

# User Manual



## TETA TP Series PLC

## Catalog

<b>1</b>	<b>TETA PLC EDITOR SOFTWARE OVERVIEW .....</b>	<b>1</b>
<b>2</b>	<b>BASIC OPERATION AND SCREEN INTRODUCTION .....</b>	<b>3</b>
2.1	START AND EXIT SOFTWARE .....	3
2.2	INTEGRAL STRUCTURE INTRODUCTION .....	4
2.3	QUICKACCESSBAR.....	4
2.4	MENU BAR .....	7
2.4.1	PLC .....	7
2.4.2	View.....	16
2.4.3	Help.....	17
2.5	PROJECT MANAGER .....	19
2.6	PROGRAMMING AREA .....	19
2.7	FOLDING WINDOWS.....	20
2.7.1	Information output .....	20
2.7.2	PLC Verify .....	21
2.7.3	Cross reference list.....	21
2.7.4	Device list.....	22
2.7.5	Search/ Replace .....	23
2.8	STATE BAR.....	24
<b>3</b>	<b>PROJECT MANAGEMENT.....</b>	<b>26</b>
3.1	PROJECT OPERATIONS.....	26
3.1.1	Create new project.....	26
3.1.2	Saving project .....	27
3.1.3	Closing project .....	27
3.1.4	Opening existing projects .....	28
3.1.5	Deleting projects.....	28
3.1.6	Changing PLC type of project.....	28
3.1.7	Switching programming language of project .....	29
3.2	PROJECT PROPERTY.....	29
3.2.1	Program.....	29
3.2.2	Device Comment.....	30
3.2.3	PLC parameter .....	32
3.2.4	Setting Security for Projects.....	36
<b>4</b>	<b>EDITING PROGRAMS.....</b>	<b>42</b>
4.1	PROGRAMMING.....	42
4.1.1	Ladder program .....	42
4.1.2	Instruction list programming .....	47

4.1.3	<i>Right-click menu</i> .....	48
4.2	LABELS .....	53
4.2.1	<i>Comment</i> .....	53
4.2.2	<i>Statement</i> .....	53
4.3	COMPILE .....	54
<b>5</b>	<b>TRANSFER PROJECT</b> .....	<b>56</b>
5.1	COMMUNICATION TEST .....	56
5.2	DOWNLOAD PROJECT .....	58
5.3	UPLOAD PROJECT .....	60
5.4	UPLOAD PROHIBITED .....	63
<b>6</b>	<b>DEBUG PROGRAM</b> .....	<b>64</b>
6.1	MONITOR MODE .....	64
6.2	MONITOR EDIT .....	64
<b>7</b>	<b>SHORTCUTS LIST</b> .....	<b>66</b>
7.1	COMMON SHORTCUTS LIST .....	66
7.2	SHORTCUTS LIST IN PROGRAMMING AREA .....	66
<b>1</b>	<b>PROGRAMMING MANUAL OVERVIEW</b> .....	<b>69</b>
<b>2</b>	<b>DEVICES</b> .....	<b>70</b>
2.1	INPUT RELAY X .....	70
2.2	OUTPUT REPLAY Y .....	71
2.3	AUXILIARY RELAYS M .....	71
2.4	STATE RELAYS S .....	72
2.5	TIMER .....	73
2.6	COUNTER .....	73
2.7	HIGH SPEED COUNTER .....	74
2.7.1	<i>Output Y: high speed pulse output transistor</i> .....	74
2.7.2	<i>Input X: one phase</i> .....	75
2.7.3	<i>Input X: A/B phase</i> .....	75
2.8	DATA REGISTER D .....	76
2.8.1	<i>Index registers V, Z</i> .....	76
2.8.2	<i>File registers D</i> .....	77
2.9	POINTERS REGISTERS P, I .....	77
2.10	CONSTANT K, H .....	78
2.10.1	<i>Constant K</i> .....	78
2.10.2	<i>Constant H</i> .....	78
<b>3</b>	<b>BASIC PROGRAM INSTRUCTIONS</b> .....	<b>79</b>

3.1	BASIC PROGRAM INSTRUCTION LIST .....	79
3.2	BASIC PROGRAM INSTRUCTION DESCRIPTION .....	80
3.2.1	<i>LD, LDI (Load, Load Inverse)</i> .....	80
3.2.2	<i>OUT (out)</i> .....	80
3.2.3	<i>AND, ANI (And, And Inverse)</i> .....	81
3.2.4	<i>OR, ORI (Or, Or Inverse)</i> .....	81
3.2.5	<i>LDP, LDF (Load Pulse, Load Trailing Pulse)</i> .....	82
3.2.6	<i>ANDP, ANDF (And Pulse, And Trailing Pulse)</i> .....	83
3.2.7	<i>ORP, ORF (Or Pulse, Or Trailing Pulse)</i> .....	84
3.2.8	<i>ANB, ORB (And Block)</i> .....	85
3.2.9	<i>MPS, MRD and MPP</i> .....	86
3.2.10	<i>MC, MCR</i> .....	88
3.2.11	<i>INV</i> .....	90
3.2.12	<i>PLS, PLF (Rising edge pulse and Falling edge pulse)</i> .....	91
3.2.13	<i>SET, RST</i> .....	92
<b>4</b>	<b>APPLIED INSTRUCTIONS</b> .....	<b>93</b>
4.1	APPLIED INSTRUCTION LIST .....	93
4.1.1	<i>Program Flow instruction list</i> .....	93
4.1.2	<i>Move And Compare instruction list</i> .....	93
4.1.3	<i>Arithmetic And Logical Operations instruction list</i> .....	94
4.1.4	<i>Rotation and Shift</i> .....	95
4.1.5	<i>Data operation</i> .....	96
4.1.6	<i>High Speed Processing instruction list</i> .....	97
4.1.7	<i>Handy Instructions list</i> .....	98
4.1.8	<i>External I/O Devices instruction list</i> .....	98
4.1.9	<i>External Devices instruction list</i> .....	99
4.1.10	<i>Floating Point instruction list</i> .....	99
4.1.11	<i>Positioning Instruction list</i> .....	100
4.1.12	<i>Real Time Clock Control</i> .....	100
4.1.13	<i>Gray Codes instruction list</i> .....	101
4.1.14	<i>Inline Comparisons Instruction list</i> .....	101
4.2	APPLIED INSTRUCTION DESCRIPTION .....	102
4.2.1	<i>ABSD instruction</i> .....	102
4.2.2	<i>ADD instruction</i> .....	104
4.2.3	<i>ALT instruction</i> .....	106
4.2.4	<i>ANR instruction</i> .....	108
4.2.5	<i>ANS instruction</i> .....	109
4.2.6	<i>ARWS instruction</i> .....	110

4.2.7	ASC instruction.....	112
4.2.8	ASCI instruction.....	114
4.2.9	BCD instruction.....	117
4.2.10	BIN conversion.....	117
4.2.11	BMOV instruction.....	118
4.2.12	BON instruction.....	119
4.2.13	CALL instruction.....	120
4.2.14	CCD instruction.....	122
4.2.15	CJ instruction.....	124
4.2.16	CML instruction.....	127
4.2.17	CMP instruction.....	129
4.2.18	DABS instruction.....	130
4.2.19	DCOS instruction.....	132
4.2.20	DEADD instruction.....	133
4.2.21	DEBCD instruction.....	134
4.2.22	DEBIN instruction.....	135
4.2.23	DEC instruction.....	136
4.2.24	DECMP instruction.....	137
4.2.25	DECO instruction.....	138
4.2.26	DEDIV instruction.....	139
4.2.27	DEMUL instruction.....	140
4.2.28	DESQR instruction.....	141
4.2.29	DESUB instruction.....	142
4.2.30	DEZCP instruction.....	143
4.2.31	DHSCR instruction.....	144
4.2.32	DHSCS instruction.....	145
4.2.33	DHSZ instruction.....	147
4.2.34	DIV instruction.....	148
4.2.35	DRVA instruction.....	149
4.2.36	DRVI instruction.....	152
4.2.37	DSIN instruction.....	154
4.2.38	DSW instruction.....	156
4.2.39	DTAN instruction.....	158
4.2.40	IRET、EI、DI instruction.....	160
4.2.41	ENCO instruction.....	165
4.2.42	FEND instruction.....	166
4.2.43	FLT instruction.....	167
4.2.44	FMOV instruction.....	167
4.2.45	FOR, NEXT instruction.....	168

4.2.46	FROM instruction.....	171
4.2.47	GBIN instruction.....	172
4.2.48	GRY instruction.....	173
4.2.49	HEX instruction.....	173
4.2.50	HKY instruction.....	176
4.2.51	HOUR instruction.....	177
4.2.52	INC instruction.....	178
4.2.53	INCD instruction.....	179
4.2.54	INT instruction.....	181
4.2.55	IST instruction.....	182
4.2.56	MEAN instruction.....	188
4.2.57	MOV instruction.....	189
4.2.58	MTR instruction.....	190
4.2.59	MUL instruction.....	192
4.2.60	NEG instruction.....	193
4.2.61	PID instruction.....	195
4.2.62	PLSR instruction.....	200
4.2.63	PLSV instruction.....	203
4.2.64	PLSY instruction.....	204
4.2.65	PR instruction.....	205
4.2.66	PRUN instruction.....	206
4.2.67	PTO instruction.....	208
4.2.68	PWM instruction.....	210
4.2.69	RAMP instruction.....	211
4.2.70	RCL instruction.....	212
4.2.71	RCR instruction.....	213
4.2.72	REF instruction.....	215
4.2.73	REFF instruction.....	216
4.2.74	ROL instruction.....	217
4.2.75	ROR instruction.....	218
4.2.76	ROTC instruction.....	219
4.2.77	RS instruction.....	221
4.2.78	SEGD instruction.....	222
4.2.79	SEGL instruction.....	224
4.2.80	SER instruction.....	224
4.2.81	SFRD instruction.....	226
4.2.82	SFTL instruction.....	227
4.2.83	SFTR instruction.....	227
4.2.84	SFWR instruction.....	228

4.2.85	<i>SMOV instruction</i> .....	229
4.2.86	<i>SORT instruction</i> .....	231
4.2.87	<i>SPD instruction</i> .....	233
4.2.88	<i>SQR instruction</i> .....	235
4.2.89	<i>STMR instruction</i> .....	236
4.2.90	<i>SUB instruction</i> .....	237
4.2.91	<i>SUM instruction</i> .....	238
4.2.92	<i>SWAP instruction</i> .....	239
4.2.93	<i>TADD instruction</i> .....	240
4.2.94	<i>TCMP instruction</i> .....	241
4.2.95	<i>TKY instruction</i> .....	242
4.2.96	<i>TO instruction</i> .....	245
4.2.97	<i>TRD instruction</i> .....	246
4.2.98	<i>TSUB instruction</i> .....	247
4.2.99	<i>TTMR instruction</i> .....	248
4.2.100	<i>TWR instruction</i> .....	251
4.2.101	<i>TZCP instruction</i> .....	253
4.2.102	<i>WAND instruction</i> .....	254
4.2.103	<i>WDT instruction</i> .....	255
4.2.104	<i>WOR instruction</i> .....	257
4.2.105	<i>WSFL instruction</i> .....	258
4.2.106	<i>WSFR instruction</i> .....	259
4.2.107	<i>WXOR instruction</i> .....	260
4.2.108	<i>XCH instruction</i> .....	261
4.2.109	<i>ZCP instruction</i> .....	263
4.2.110	<i>ZRN instruction</i> .....	264
4.2.111	<i>ZRST instruction</i> .....	266
4.2.112	<i>AND Comparisons instructions</i> .....	267
4.2.113	<i>LD Comparisons instructions</i> .....	268
4.2.114	<i>OR Comparisons instructions</i> .....	269
<b>5</b>	<b>STL INSTRUCTION</b> .....	<b>270</b>
5.1	STL INSTRUCTION LIST.....	270
5.2	STL INSTRUCTION DESCRIPTION.....	270
<b>6</b>	<b>SYSTEM-SPECIFIC ADDRESS LIST</b> .....	<b>274</b>
<b>7</b>	<b>ERROR CODE</b> .....	<b>290</b>
7.1	NO ERROR .....	290

7.1.1	Error code 0000 .....	290
7.2	CONFIGURATION ERROR (0***~1***) .....	290
7.2.1	Error code 0***~1***) .....	290
7.3	HARDWARE ERROR (6101~6115).....	291
7.3.1	Error code 6101 .....	291
7.3.2	Error code 6102 .....	291
7.3.3	Error code 6103 .....	292
7.3.4	Error code 6104 .....	292
7.3.5	Error code 6105 .....	292
7.3.6	Error code 6106 .....	293
7.3.7	Error code 6107 .....	293
7.3.8	Error code 6113 .....	293
7.3.9	Error code 6114 .....	293
7.3.10	Error code 6115 .....	294
7.4	PP COMMUNICATION ERROR (6201~6205) .....	294
7.4.1	Error code 6201 .....	294
7.4.2	Error code 6202 .....	295
7.4.3	Error code 6203 .....	295
7.4.4	Error code 6204 .....	295
7.4.5	Error code 6205 .....	296
7.5	SERIAL COMMUNICATION ERROR (6301~6340) .....	296
7.5.1	Error code 6301 .....	296
7.5.2	Error code 6302 .....	296
7.5.3	Error code 6303 .....	297
7.5.4	Error code 6304 .....	297
7.5.5	Error code 6305 .....	298
7.5.6	Error code 6306 .....	298
7.5.7	Error code 6307 .....	298
7.5.8	Error code 6308 .....	299
7.5.9	Error code 6309 .....	299
7.5.10	Error code 6312 .....	300
7.5.11	Error code 6313 .....	300
7.5.12	Error code 6314 .....	300
7.5.13	Error code 6320 .....	301
7.5.14	Error code 6340 .....	301
7.6	PARAMETER ERROR (6401~6421) .....	302
7.6.1	Error code 6401 .....	302
7.6.2	Error code 6402 .....	302
7.6.3	Error code 6403 .....	302



7.6.4	Error code 6404 .....	303
7.6.5	Error code 6405 .....	303
7.6.6	Error code 6406 .....	304
7.6.7	Error code 6407 .....	304
7.6.8	Error code 6407 .....	304
7.6.9	Error code 6411 .....	305
7.6.10	Error code 6412 .....	305
7.6.11	Error code 6413 .....	306
7.6.12	Error code 6420 .....	306
7.6.13	Error code 6421 .....	307
7.7	SYNTAX ERROR (6501~6512).....	307
7.7.1	Error code 6501 .....	307
7.7.2	Error code 6502 .....	308
7.7.3	Error code 6503 .....	308
7.7.4	Error code 6504 .....	309
7.7.5	Error code 6505 .....	309
7.7.6	Error code 6506 .....	310
7.7.7	Error code 6507 .....	310
7.7.8	Error code 6508 .....	310
7.7.9	Error code 6509 .....	311
7.7.10	Error code 6510 .....	311
7.7.11	Error code 6511 .....	312
7.7.12	Error code 6512 .....	312
7.8	LADDER ERROR (6601~6632) .....	313
7.8.1	Error code 6601 .....	313
7.8.2	Error code 6602 .....	313
7.8.3	Error code 6603 .....	314
7.8.4	Error code 6604 .....	314
7.8.5	Error code 6605 .....	315
7.8.6	Error code 6606 .....	315
7.8.7	Error code 6607 .....	316
7.8.8	Error code 6608 .....	316
7.8.9	Error code 6609 .....	317
7.8.10	Error code 6610 .....	317
7.8.11	Error code 6611 .....	318
7.8.12	Error code 6612 .....	318
7.8.13	Error code 6613 .....	319
7.8.14	Error code 6614 .....	319
7.8.15	Error code 6615 .....	320

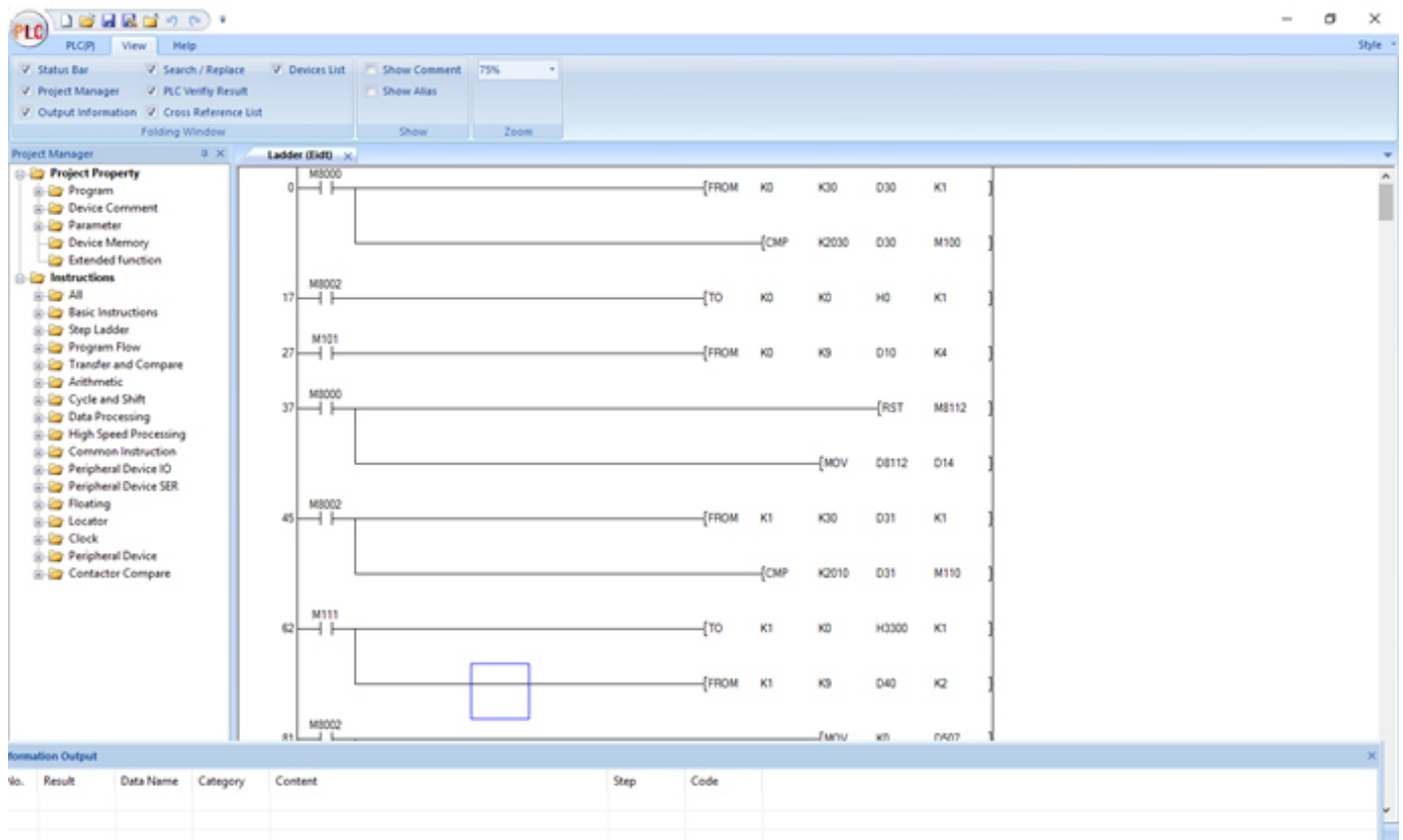
7.8.16	Error code 6616 .....	320
7.8.17	Error code 6617 .....	321
7.8.18	Error code 6618 .....	321
7.8.19	Error code 6619 .....	322
7.8.20	Error code 6620 .....	322
7.8.21	Error code 6621 .....	323
7.8.22	Error code 6622 .....	323
7.8.23	Error code 6623 .....	323
7.8.24	Error code 6624 .....	324
7.8.25	Error code 6625 .....	324
7.8.26	Error code 6626 .....	325
7.8.27	Error code 6627 .....	325
7.8.28	Error code 6628 .....	326
7.8.29	Error code 6629 .....	326
7.8.30	Error code 6630 .....	327
7.8.31	Error code 6631 .....	327
7.8.32	Error code 6632 .....	328
7.8.33	Error code 6633 .....	328
7.9	Operation error (6701~6780) .....	329
7.9.1	Error code 6701 .....	329
7.9.2	Error code 6702 .....	329
7.9.3	Error code 6703 .....	330
7.9.4	Error code 6704 .....	330
7.9.5	Error code 6705 .....	331
7.9.6	Error code 6706 .....	331
7.9.7	Error code 6707 .....	331
7.9.8	Error code 6708 .....	332
7.9.9	Error code 6709 .....	332
7.9.10	Error code 6710 .....	333
7.9.11	Error code 6712 .....	333
7.9.12	Error code 6730 .....	334
7.9.13	Error code 6732 .....	334
7.9.14	Error code 6733 .....	334
7.9.15	Error code 6734 .....	335
7.9.16	Error code 6735 .....	335
7.9.17	Error code 6736 .....	335
7.9.18	Error code 6740 .....	336
7.9.19	Error code 6742 .....	336
7.9.20	Error code 6743 .....	336

7.9.21	Error code 6744 .....	337
7.9.22	Error code 6745 .....	337
7.9.23	Error code 6746 .....	338
7.9.24	Error code 6747 .....	338
7.9.25	Error code 6748 .....	338
7.9.26	Error code 6749 .....	339
7.9.27	Error code 6750 .....	339
7.9.28	Error code 6751 .....	339
7.9.29	Error code 6752 .....	340
7.9.30	Error code 6753 .....	340
7.9.31	Error code 6754 .....	341
7.9.32	Error code 6755 .....	341
7.9.33	Error code 6756 .....	341
7.9.34	Error code 6757 .....	342
7.9.35	Error code 6758 .....	342
7.9.36	Error code 6759 .....	343
7.9.37	Error code 6760 .....	343
7.9.38	Error code 6762 .....	343
7.9.39	Error code 6763 .....	344
7.9.40	Error code 6764 .....	344
7.9.41	Error code 6765 .....	344
7.9.42	Error code 6770 .....	345
7.9.43	Error code 6771 .....	345
7.9.44	Error code 6772 .....	345
7.9.45	Error code 6772 .....	346
7.9.46	Error code 6774 .....	346
7.9.47	Error code 6780 .....	346
<b>8</b>	<b>EXAMPLE .....</b>	<b>348</b>
8.1	SETTING FOR TP SERIES PLC COM2 .....	348
8.1.1	Protocol Setting (D8126).....	348
8.1.2	Communication Format (D8120) .....	348
8.1.3	TETA PLC - MODBUS (Slave) addresses rules .....	349
8.1.4	MODBUS Function Code Introduction.....	349
8.1.5	Example .....	351
8.2	N: N APPLICATION IN TP SERIES PLC .....	355
8.2.1	N: N Instructions .....	355
8.2.2	System registers .....	356
8.2.3	Example .....	357

8.2.4 Notice..... 358

# TETA PLC Editor

## Operating manual



# 1 TETA PLC Editor Software Overview

PLC is a digital computer used for automation of typically industrial electromechanical processes; PLCs are used in many machines, in many industries.

It reads external input signals such as: the state of buttons, sensors, switches and pulse waves, and then uses a microprocessor to perform logic, sequence, timing, counting and arithmetic operations, resulting in the corresponding output signal based on the input signal status or internally stored value and pre-written program.

TETA PLC editor uses ladder and instructions list as programming language.

## Ladder

Ladder logic is widely used to program PLCs, where sequential control of a process or manufacturing operation is required. Ladder logic is useful for simple but critical control systems or for reworking old hardwired relay circuits. As programmable logic controllers became more sophisticated it has also been used in very complex automation systems. It is a graphic language evolution came in relay ladder original relay control system based on the devices used in the design, such as buttons X, intermediate relay M, time relay T, counter C, and so on similar properties contact time of electrical device. The ladder as the Figure 1-1 shows.

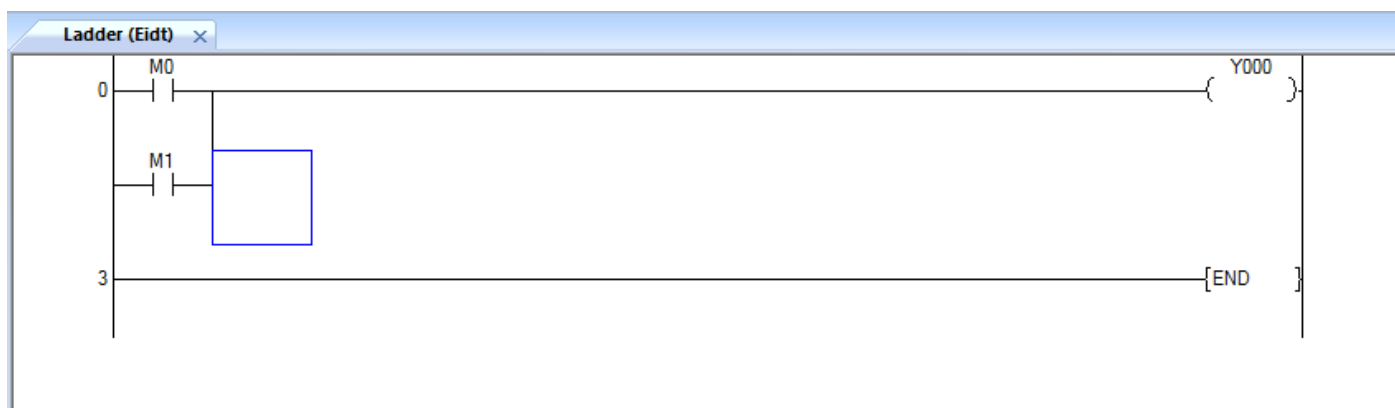
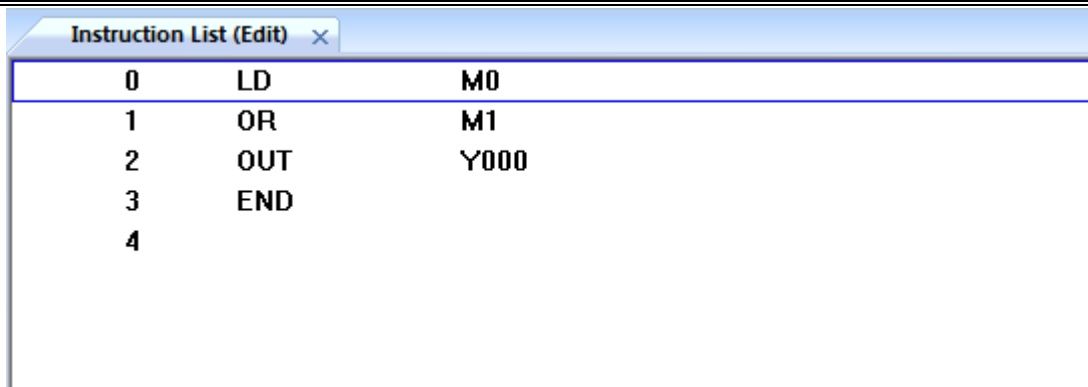


Figure 1-1

## Instructions list

Instruction List (IL) is designed for programmable logic controllers (PLCs). It is a low level language and resembles assembly. All the instructions and operands are inputted for PLC programming. The IL as the Figure 1-2 shows.



Address	Instruction	Operand
0	LD	M0
1	OR	M1
2	OUT	Y000
3	END	
4		

Figure 1-2

### Program switch

According to their own programming practice, users can switch ladder and instruction list in order to improve programming efficiency. There is switch function as Figure 1-3 shows.

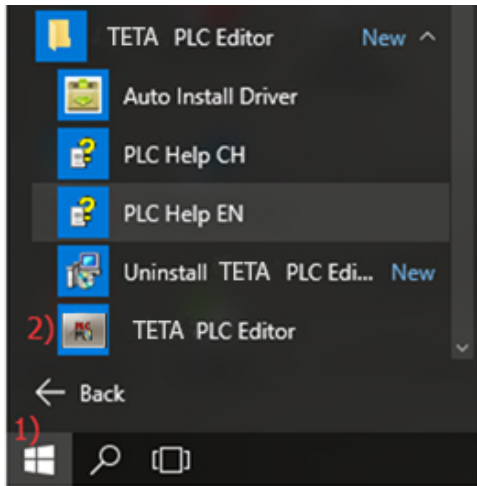


Figure 1-3

## 2 Basic operation and screen introduction

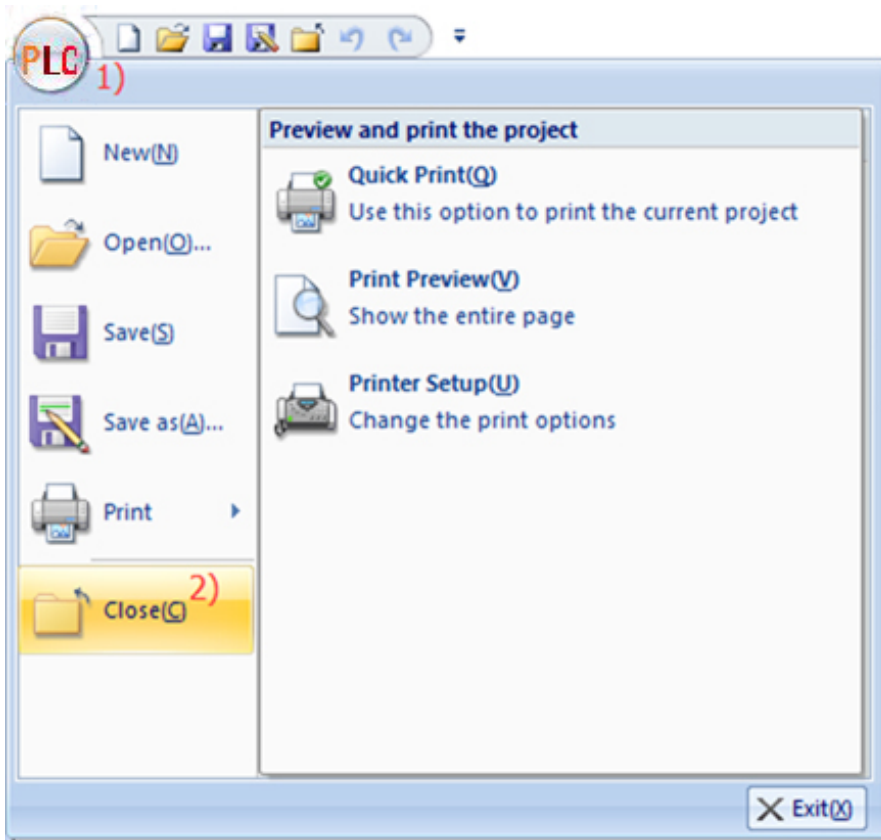
TETA PLC Editor provides full and powerful functions.

### 2.1 Start and exit software



TETA PLC Editor is a programming tool for designing, debugging, and maintaining programs on Windows, please start the software by click icon on desktop or select [Start]->[ TETA PLC Editor]. As Figure 2-1 shows.

Figure 2-1



Click [TETA logo] -> [Close] to exit software, if program is not saved, software will ask saving, as Figure 2-2 and Figure 2-3 show.

Figure 2-2



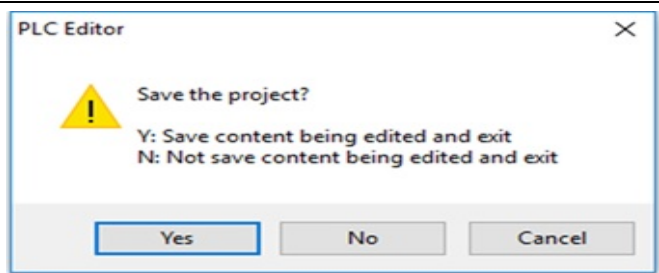


Figure 2-3

## 2.2 Integral structure Introduction

The Figure 2-4 shows TETA PLC Editor Screen, the upper area of main screen is [Menu Toolbar]; the [Project Manager] on the left of screen; the programming area is on the middle of screen. The low part lists other tools.

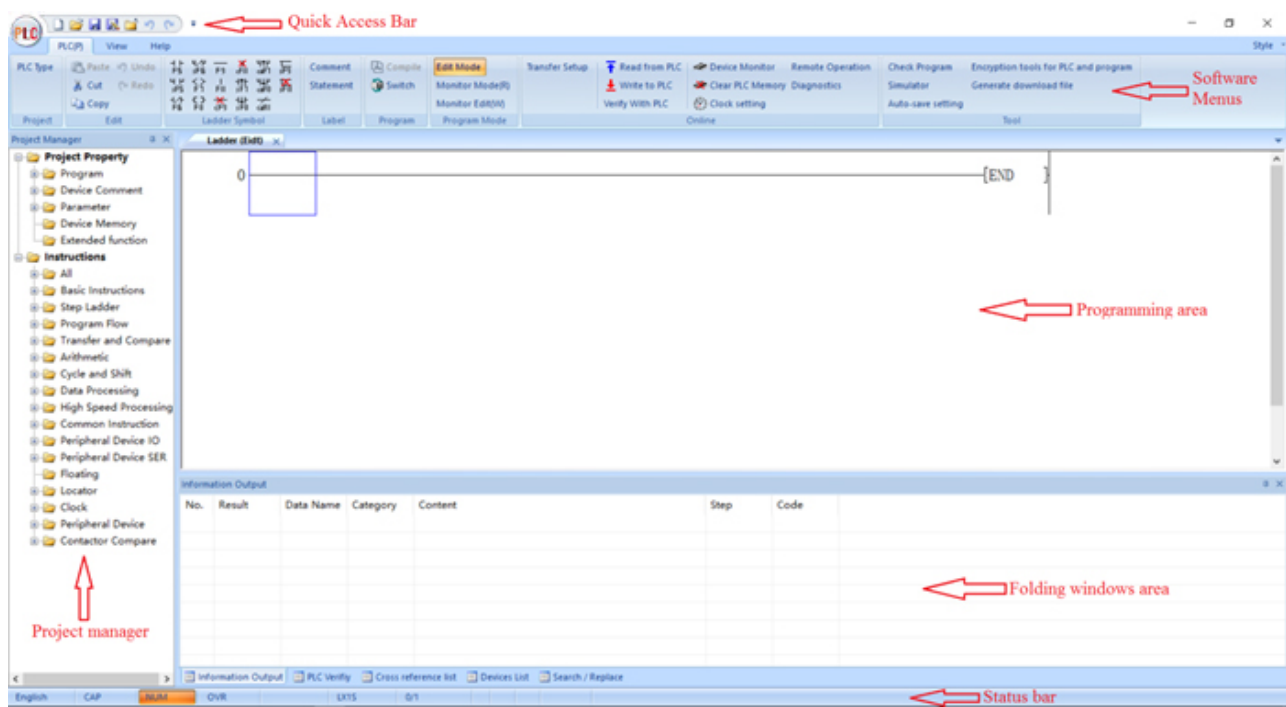


Figure 2-4

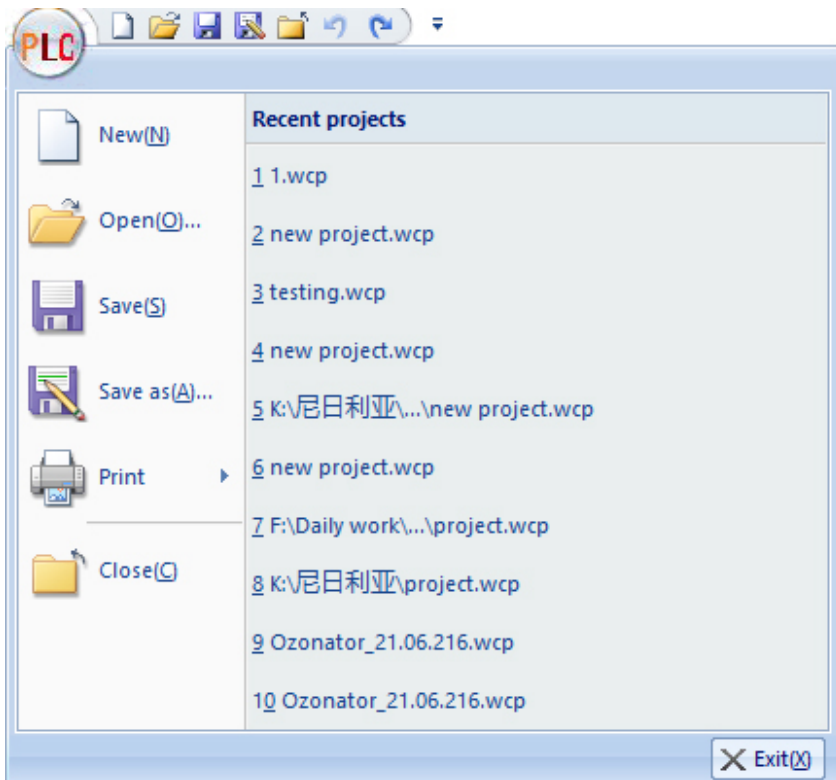
Users can modify the [Project Manager] and other tools position by themselves.

## 2.3 Quick Access Bar

Users can do the basic operation via [QuickAccessBar], the [QuickAccessBar] is introduced in front of content.










Figure 2-5



: TETA LOGO, click it to open the windows, as Figure 2-6 shows.

Figure 2-6

- : New, click it to create new program;
- : Open, click it to open current program;
- : Save, click it to save modification;
- : Save as, click it to save program in other path;

- : Close, click it to exit current program;
- : Undo;
- : Redo;

### Customize Quick Access Toolbar

Users can more easily customize the functionality as what they need by using the [Customize Quick Access Toolbar].

- 1) Access [Customize Quick Access Toolbar]. Please move the mouse cursor to drop-down icon which is on the right of [QuickAccessBar];
- 2) Open the [Customize Quick Access Toolbar], and select [More command], as Figure 2-7 shows;
- 3) After completing Step 2, then open the [custom] window, as Figure 2-8 shows.
- 4) Add Shortcut: Select any command in the [command] box on the left side. Then click [Add] so that you will find the selected item moves to the right box. And click [OK] can successfully add a custom tool.
- 5) Delete Shortcut: Select any command in the [command] box where the right side. Then click [Delete] so that you can find selected items in the right dialog box disappears. Click [OK] to complete delete the command.

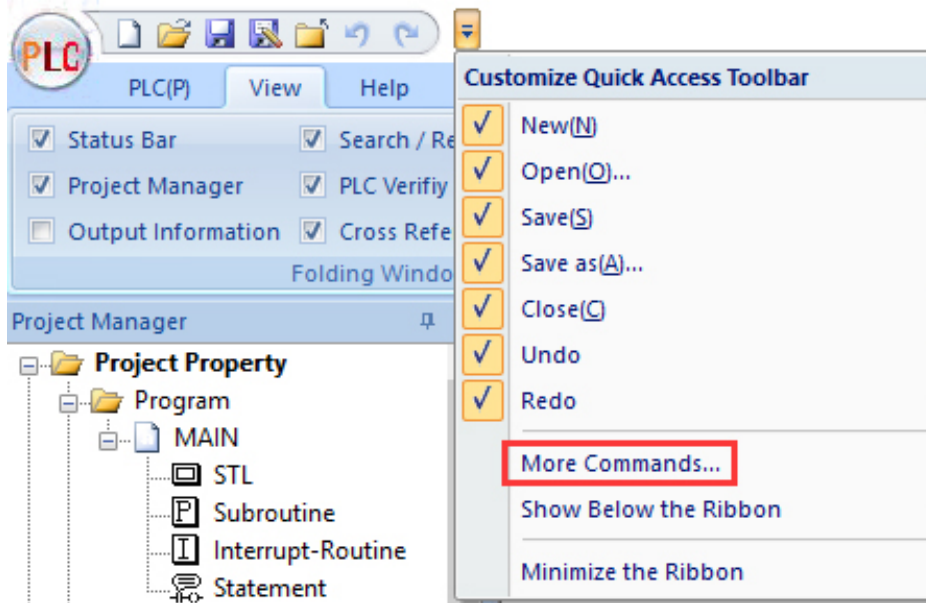


Figure 2-7

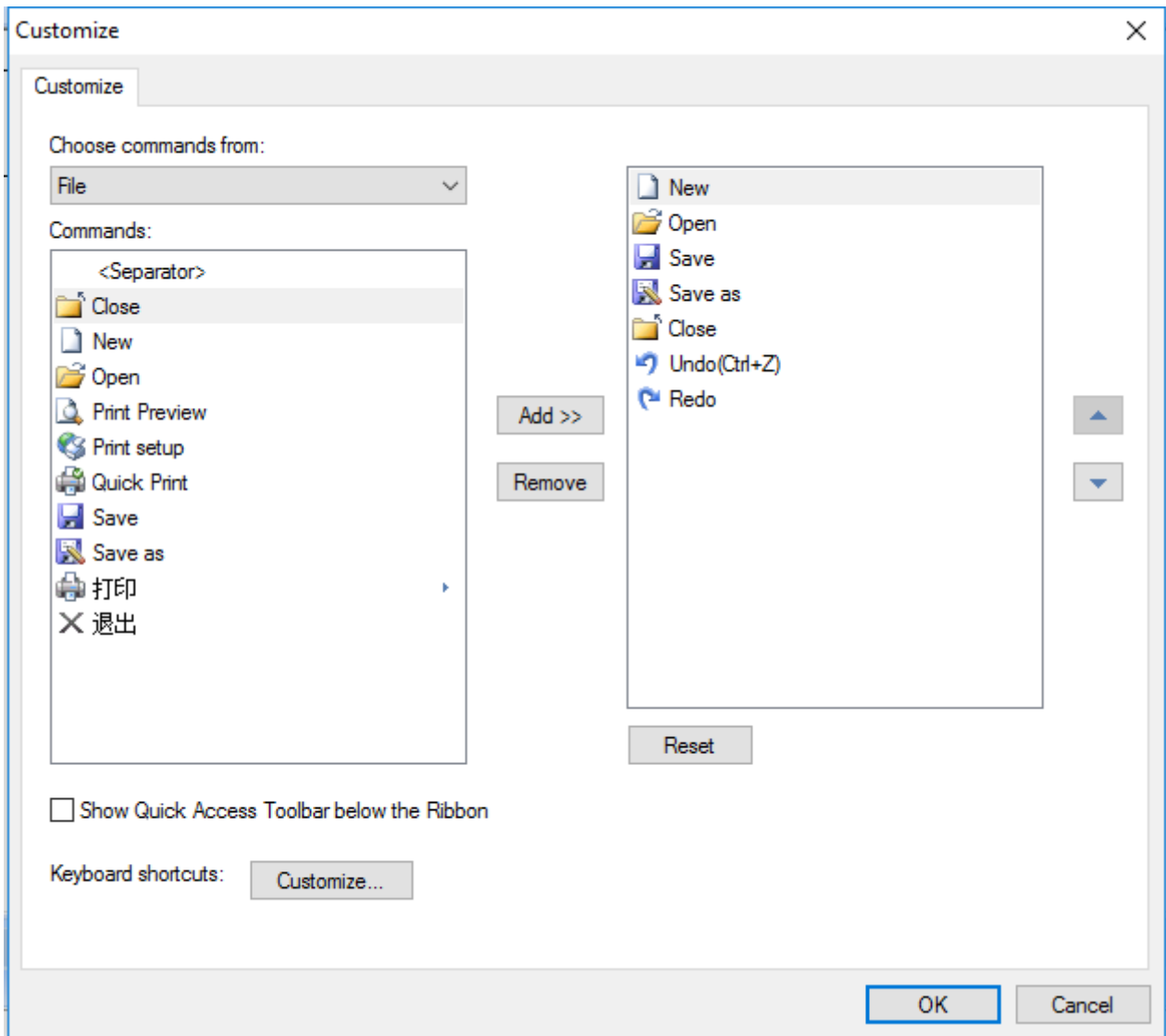


Figure 2-8

## 2.4 Menu bar

This sector introduces the software menu bar. The upper area of main screen is [Menu Toolbar]; users can easily do various operations through it, and then detailed operations, please see following instructions.

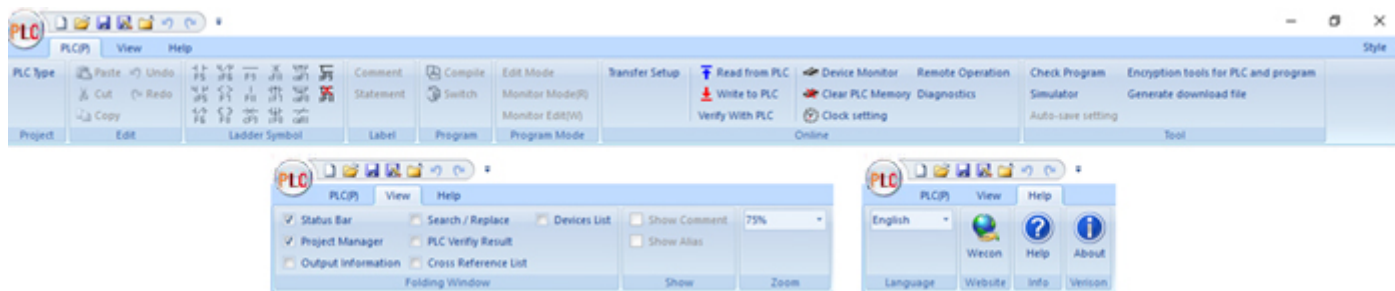


Figure 2-9

### 2.4.1 PLC

#### 2.4.1.1 Project Toolbar

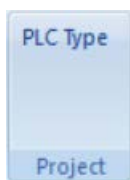


Figure 2-10

User can change the PLC type for program by click [PLC Type] in [Project] toolbar, as Figure 2-11 shows.

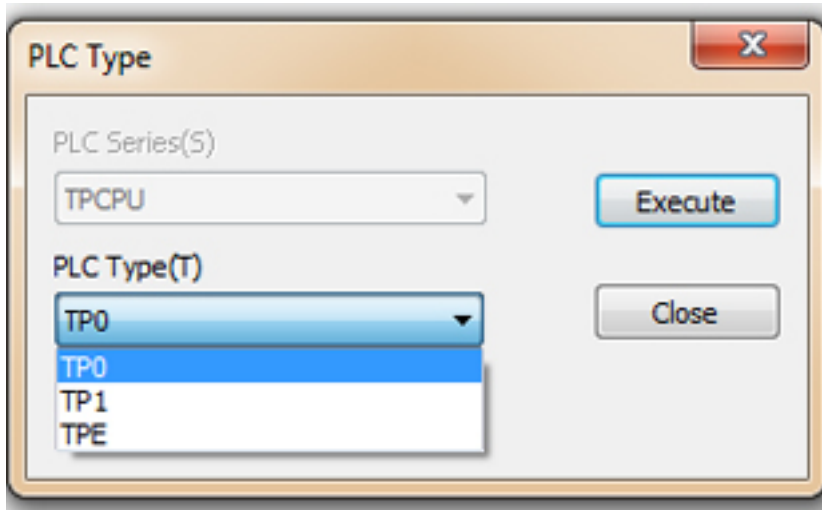


Figure 2-11

#### 2.4.1.2 Edit Toolbar

The clipboard toolbar is the basic function commonly used for editing a PLC project. This section will explain the basic operation of the clipboard.

For example: CUT, COPY, PASTE, UNDO/REDO

- 1) **Cut:** The cut command removes the selected data from its original position;
- 2) **Copy:** The copy command creates a duplicate;

3) **Paste:** Transferring text, data, files or objects from a source to a destination;

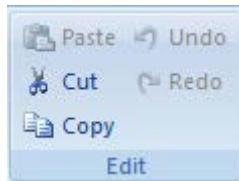


Figure 2-12

4) **Undo:** It erases the last change done to the document reverting it to an older state;

5) **Redo:** The redo command reverses the undo or advances the buffer to a more current state;

### 2.4.1.3 Ladder Symbol Toolbar

The menu in TETA PLC Editor provides the full and powerful functions, which greatly improves programming efficiency, as Figure 2-13 shows.

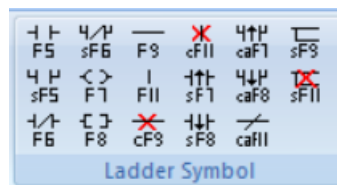



Figure 2-13

The Table 2-1 shows a detailed description of the ladder symbol

Table 2-1

Symbol	Instruction	Function	Hotkey
	LD, AND	Open contact	F5
	OR	Open branch	SHIFT+F5
	LDI, ANI	Close contact	F6
	ORI	Close branch	SHIFT+F6
	OUT	Output coil	F7
	CJ/CALL	Input application instruction	F8
		Draw horizontal line	F9
		Draw vertical line	F11
		Delete horizontal line	CTRL+F9
		Delete vertical line	CTRL+F11
	LDP, ANP	Rising pulse	SHIFT+F7
	LDF, ANF	Falling pulse	SHIFT+F8
	ORP	Rising pulse open branch	CTRL+ALT+F7
	ORF	Falling pulse close branch	CTRL+ALT+F8

	INV	Reverse operation results	CTRL+ALT+F11
---	-----	---------------------------	--------------

#### 2.4.1.4 Label Toolbar

In order to increase readability, software allows users to add comments and statements for program.

Click the text to enable this function, the following picture Figure 2-14 shows the state of each text when function is enabled.



Figure 2-14

**Note: Please run the software as Administrator in Windows 7 and later systems**

#### 2.4.1.5 Project Toolbar

This section provides a brief description of [Program] menu.

There are two functions in [Program], one is compile program, and the other is switch program.

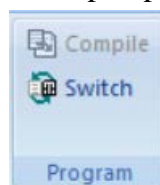


Figure 2-15

- 1) **Compile:** All the program must be compiled before download into PLC device;
- 2) **Switch:** It is used for converting program between the ladder format and the IL (Instruction List) format.

#### 2.4.1.6 Program mode Toolbar

Program mode contains three functions, [Edit Mode], [Monitor Mode (R)] and [Monitor Edit (W)].

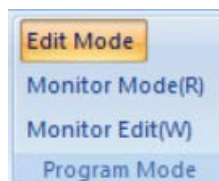


Figure 2-16

- 1) **Edit mode:** It is for editing program (ladder or IL) for PLC;
- 2) **Monitor Mode(R):** It is for monitoring program, which is running in connected PLC;
- 3) **Monitor Edit (W):** It is for modifying program when monitoring it;

#### 2.4.1.7 Online Toolbar

The Online toolbar provides nine functions for operating and accessing PLC device, such as upload/download PLC program.

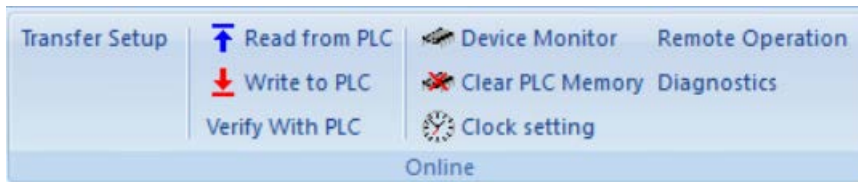


Figure 2-17

**Transfer:** It is for testing the communication between PC and PLC;

**Read from PLC:** It is for uploading the program from PLC to PC;

**Write to PLC:** It is for downloading the program from PC to PLC;

**Verify with PLC:** It is for verify the project opened by TETA PLC Editor is the same as project that running in connected PLC.

**Device monitor:** It is for monitoring and debugging all devices in PLC, when program is executed or simulated;

### Clear PLC Memory

It is for clear PLC memory, the setting window as Figure 2-18 shows;

Clear PLC memory will delete all the data in PLC device, in other word, it will format PLC.

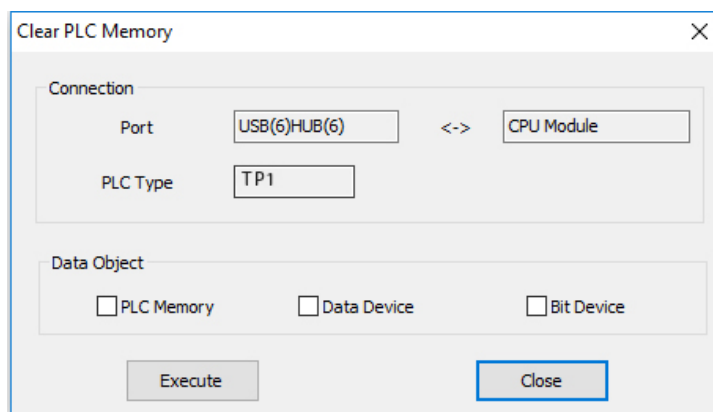


Figure 2-18

### Description

- 1) [Object]: Displays the current PLC information and parameters. They cannot be changed.
- 2) [Data Object]: Users could select clear content in this area.
  - PLC Memory: Clear PLC memory that is format the PLC
  - Data Device: Clear all the data blocks settings (device memory area of the D register default value)
  - Bit Device: Clear all data blocks setting (Device Memory) of the bit device values.

### Note:

- 1) This operation only works in PLC stop state.
- 2) This operation is not reversible.

### Clock setting

It is for setting real time for PLC device, the setting windows as Figure 2-19 shows;

Users can customize the internal PLC time by clock setting to achieve accurate calculation.

**Description**

- 1) Connection: Displays the current PLC information and parameters. They cannot be changed;
- 2) Time and Calendar: Users can easily select the time information and free to match time;
- 3) Read PC Time: Automatically read PC time without manual calibration;
- 4) Set Time: After completing the settings, click the button, then PLC will save the current settings;
- 5) Cancel: Don't save the current operating data. Exit and close the window;

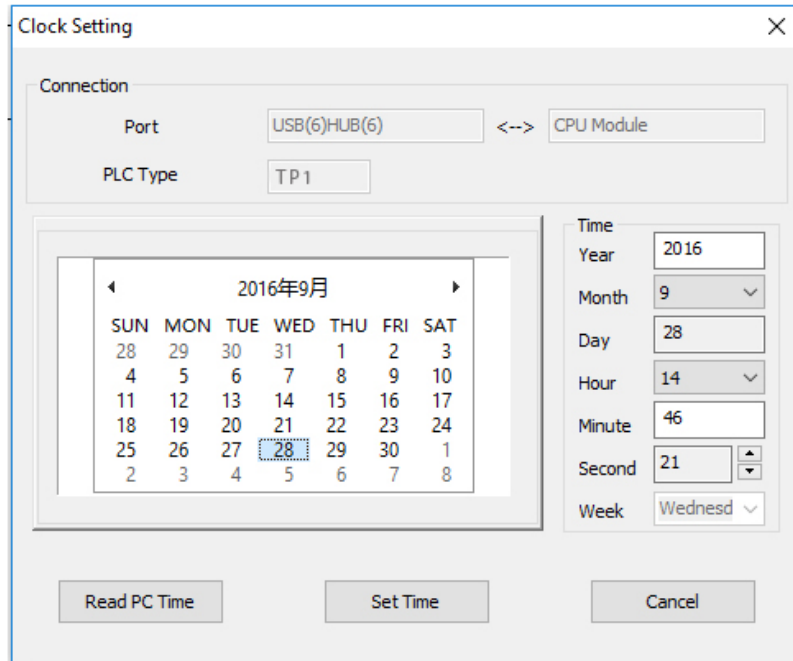


Figure 2-19

**Setting time**

- 1) Connect PLC to PC, and run TETA PLC Editor;
- 2) Switch the PLC to stop state;
- 3) Click [Online]-> [Clock setting];
- 4) Set the clock and time. The range of year is 1980~2079, the range of month is 1~12, the range of hour is 0~23, the range of minute is 0~59, the range of second is 0~59;
- 5) Users can click [Get PC time] for quickly setting;
- 6) Click [Set Clock] to save the date and time, it will pop-up tip message as following figure shows;



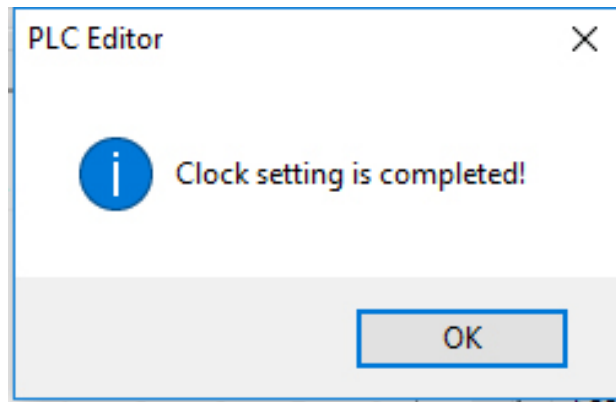


Figure 2-20

### Remote Operation

Remote operation can change the PLC states when PLC is running; it has the same functionality as PLC DIP switch.

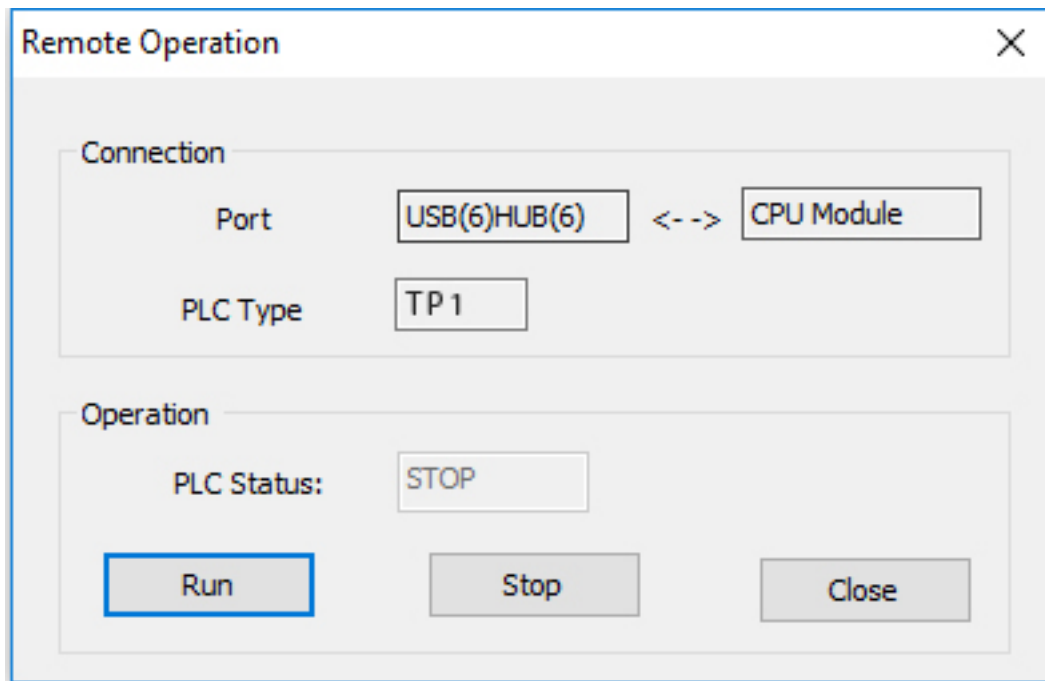


Figure 2-21

### Description

- 1) [Object]: Displays the current PLC information and parameters. They cannot be changed;
- 2) [Operation]: Displays the current status of the PLC, RUN or STOP;
  - [RUN]: PLC into RUN state;
  - [STOP]: PLC into the STOP state;
  - [CLOSE]: Close the window;

**Note:**

The device monitor requires a good communication between PLC and PC.

**Diagnostics:**

This function contains most online functions; users can view the diagnostic data in this window, and can quickly perform the operations.

**2.4.1.8 Tool Toolbar**

Tool toolbar provides five tools for users, such as check program, auto-saving as so on; it brings more convenience for programming.



Figure 2-22

**Check Program**

The function of [Check Program] is that checking the program error, and modified them timely, at the same time can quickly locate to the error. The Check Program Figure 2-23 shows.

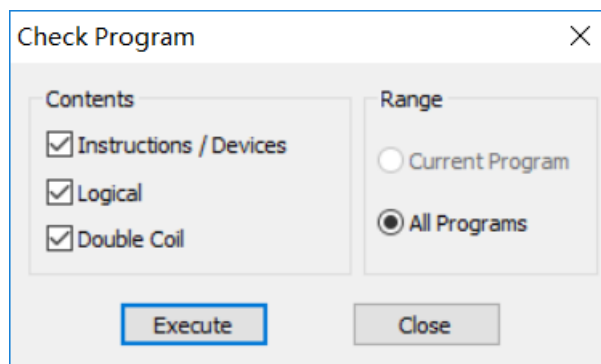


Figure 2-23

**Simulator (Not recommended)**

The function of simulator is checking the running result of program. But most instructions can be executed in simulator mode, so this function is not recommended.

The operation steps for simulator

- 1) Run TETA PLC Editor Software;
- 2) Create a PLC program and compile it;
- 3) Click [Simulator] for offline simulation, it will pop-up simulator window as Figure 2-24 shows;

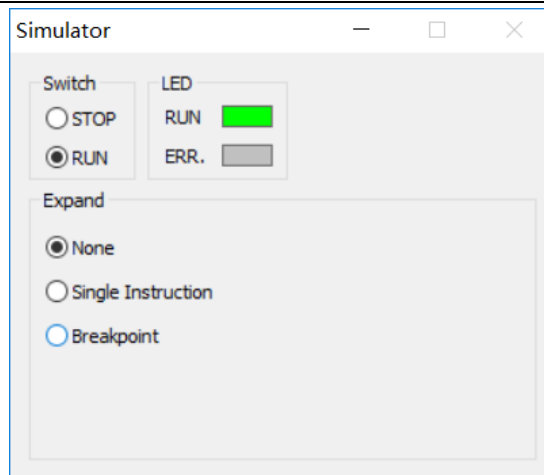


Figure 2-24

- There are two debug modes, one is Single instruction, and the other is Breakpoint.
- Single Instruction: Execute program statements one by one; interrupt occurs after executing each program statement or a process.
- Breakpoint: Users set a breakpoint in PLC program, interrupt occurs when PLC execute to this low.

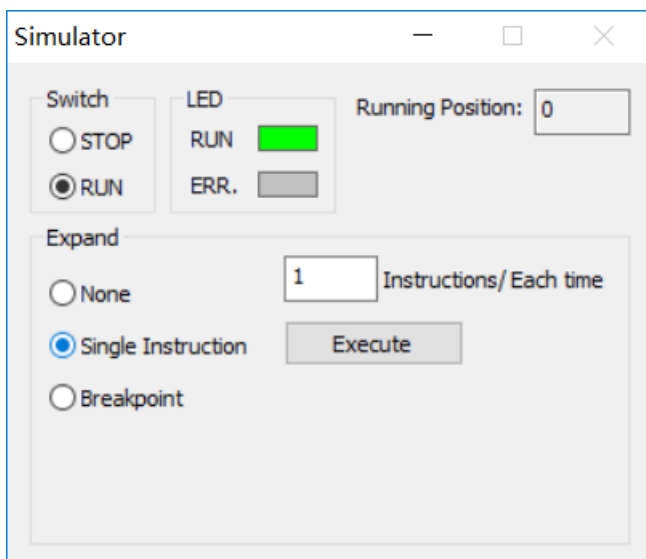


Figure 2-25

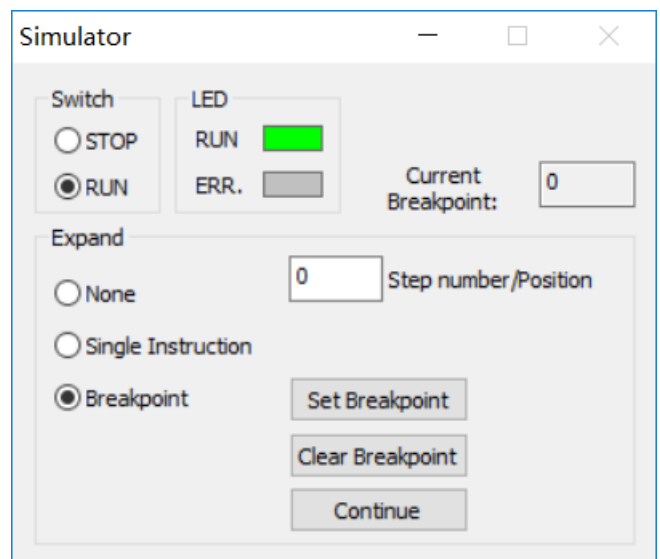


Figure 2-26

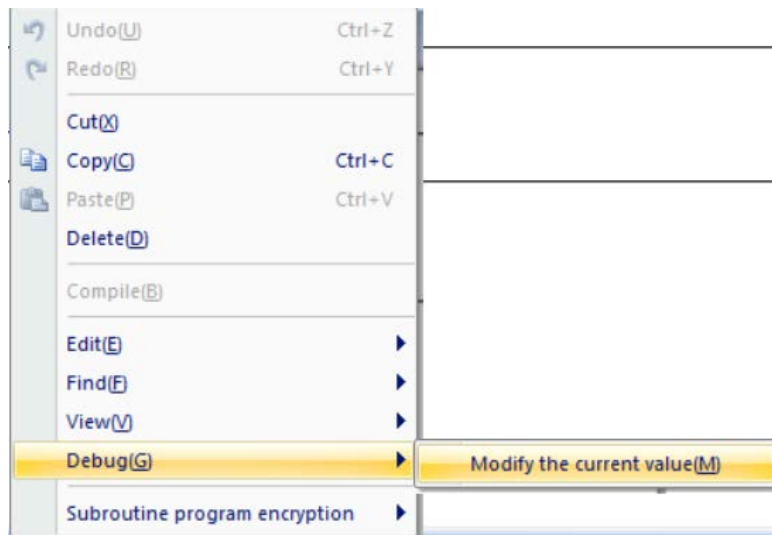


Figure 2-27

- 4) Click the register that needs to be modified, and right click it, it will pop-up a menu window.
- 5) Select [Debug]->[Modify the current value], as Figure 2-27 shows;
- 6) Users can set the value for registers as Figure 2-28 shows;
- 7) After setting the value, users can view the program running result, as Figure 2-29 shows.

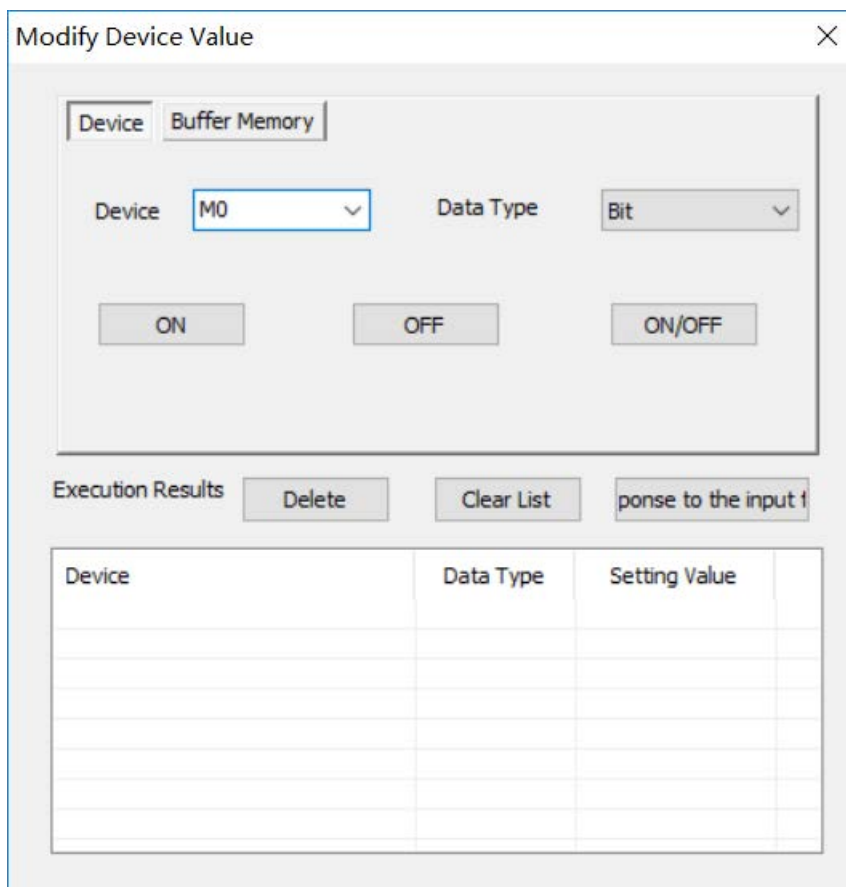


Figure 2-28

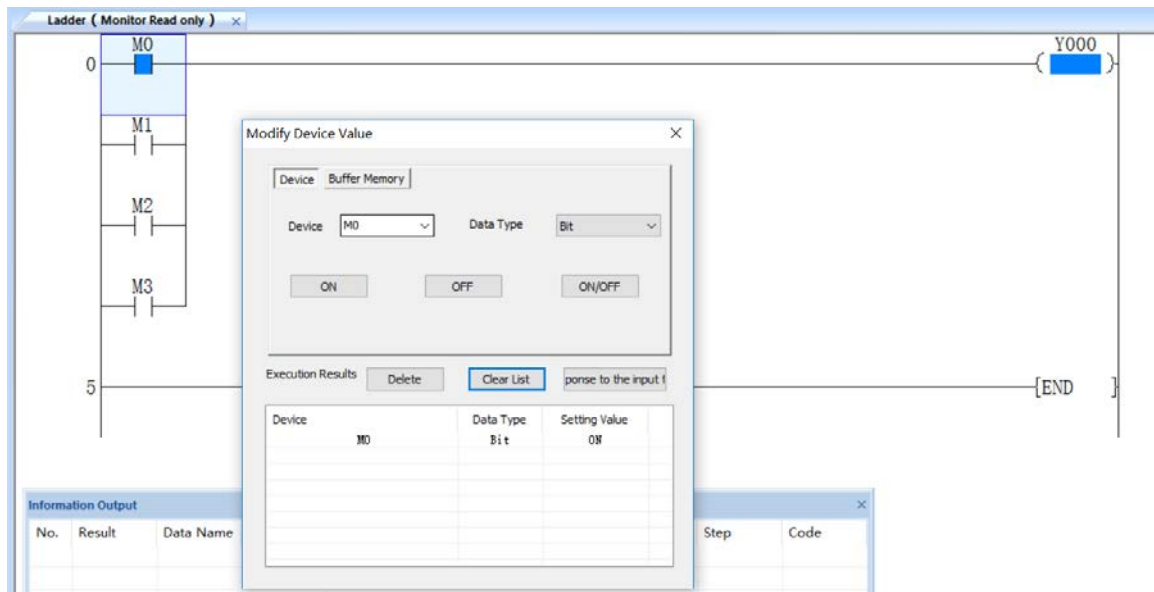


Figure 2-29

### Auto-save setting

It provides auto-save program function to avoid losing data. The default saving time is 5 seconds. The auto-save setting window as Figure 2-30 shows.

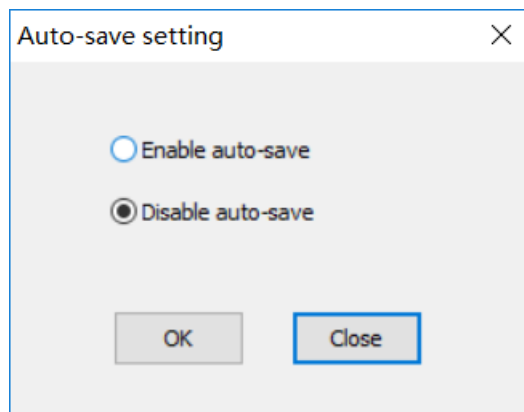


Figure 2-30

### Encryption tools for PLC and program

TETA PLC Editor provides encryption tools for users to set all kinds of password for programs. Those passwords can prevent program from being uploaded from PLC by ending users.

### Generate download file

This is for generating download file, its format is .swcp, which can't be viewed and modified by TETA PLC Editor, and so that programmer can send it to ending users directly.

### 2.4.2 View

This section introduces the [View] menu function.

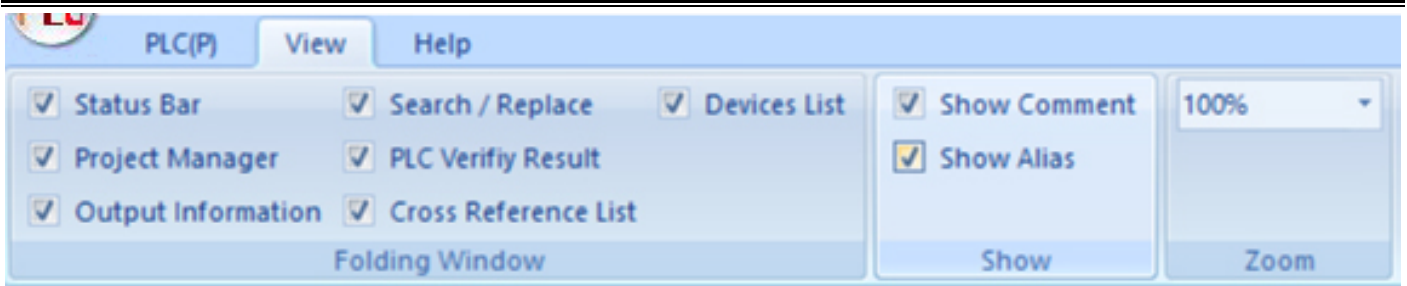


Figure 2-31

### 2.4.2.1 Folding Window toolbar

It is similar with other software; the Folding Window toolbar is used for setting the display of TETA PLC Editor, when check the option, the corresponding window will be available in software screen.

### 2.4.2.2 Show toolbar

Every devices addresses in program can be commented and aliased, check the options to show the comments are alias in programming area.

### 2.4.2.3 Zoom toolbar

The zoom only works in programming area; there are six modes as Figure 2-32 shows, users can select the display mode according to requirements.

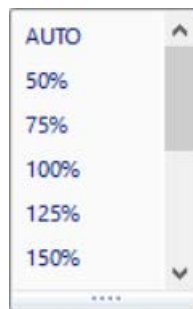


Figure 2-32

## 2.4.3 Help

This section introduces the [Help] menu function.

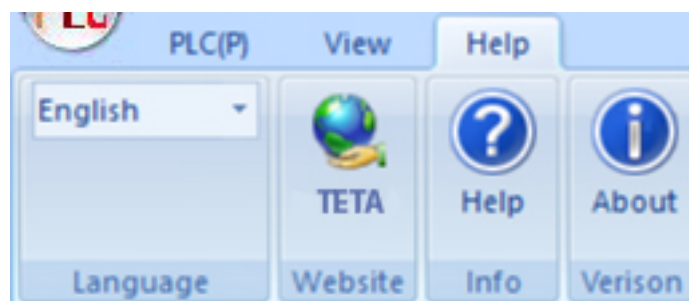


Figure 2-33

### 2.4.3.1 Language toolbar

Language toolbar is used for setting software displaying language. Simplified Chinese, Traditional

Chinese and English are supported in software.

**Note:** Please run the software as Administrator in Windows 7 and later systems for language switching.

### 2.4.3.2 Website toolbar

It is linked to TETA official webpage.

### 2.4.3.3 Info toolbar

Please click the [Help] in info toolbar for opening the helping document.

### 2.4.3.4 Version toolbar

Please click it to check the software version, the software version information as Figure 2-34 shows.

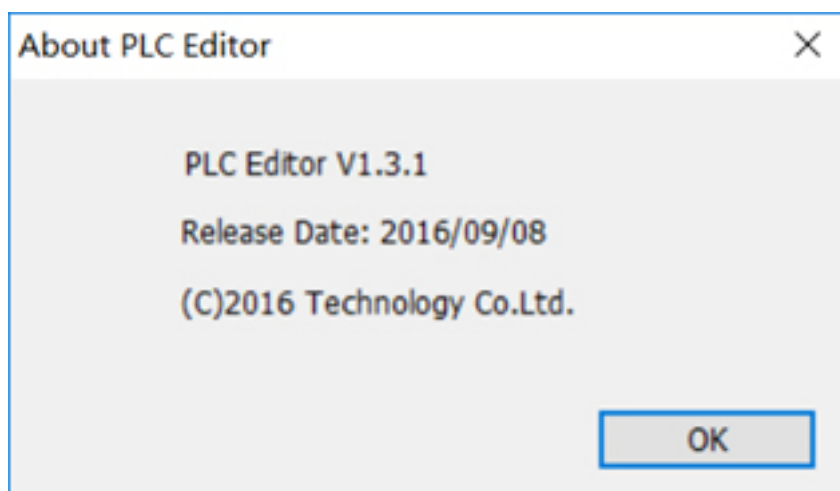


Figure 2-34

## 2.5 Project manager

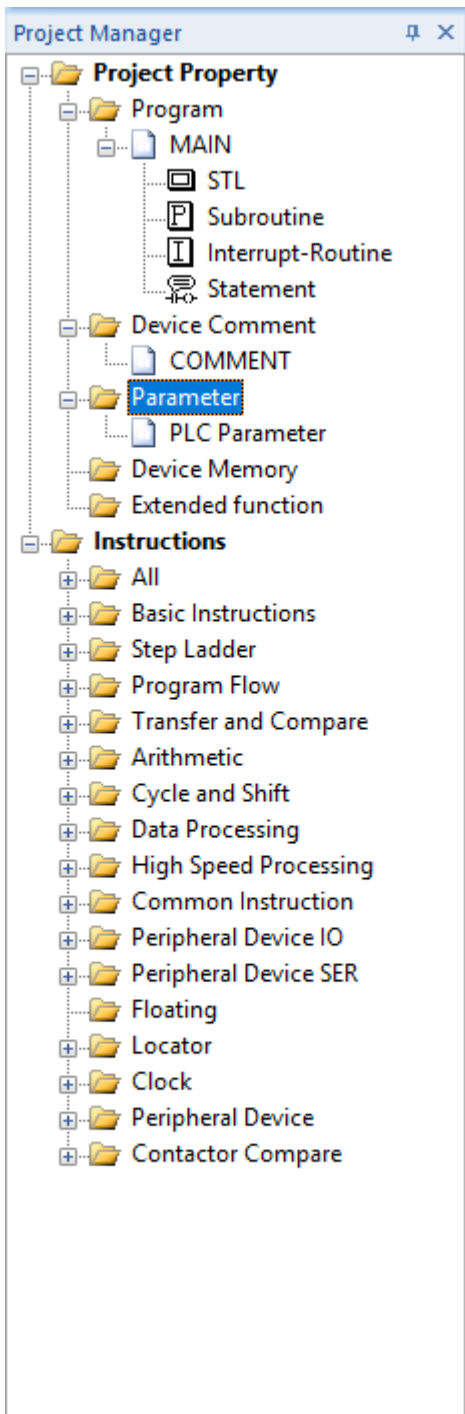


Figure 2-35

This sector introduces the [Project manager].

The left side of main screen is [Project Manager]. It shows the entire project by tree. The project includes organization structure, project name, program, device comment, parameter, device memory, extended function, all kinds of instruction table and so on.

Users can control the whole project by [Project Manager]. And [Project Manager] supports the use of right button function. As Figure 2-35 shows.

## 2.6 Programming area

Programming area is a main screen used for operations such as programming (ladder or instruction list), parameter setting, and monitoring in TETA PLC Editor.

On the top left, it shows current states and programming information.



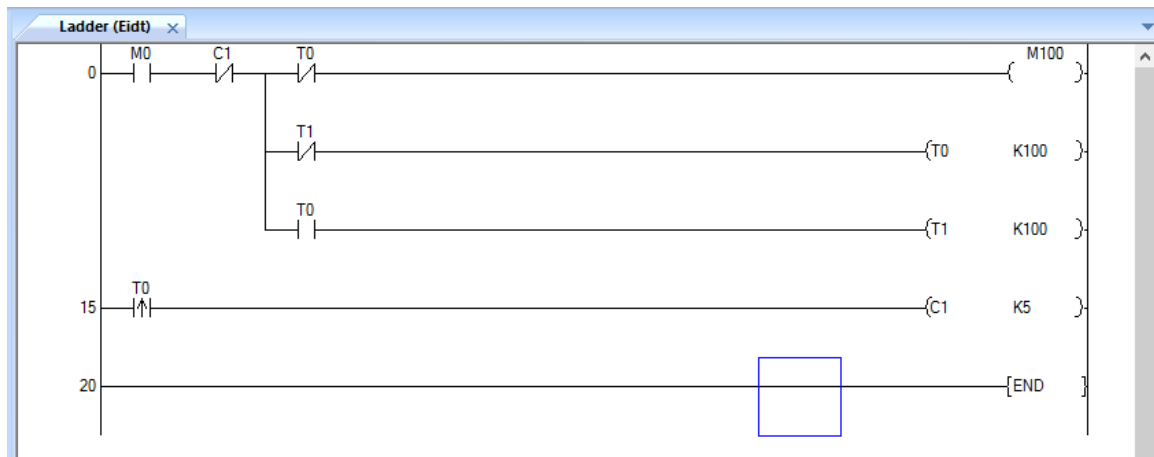


Figure 2-36

No.	Instruction	Operand 1	Operand 2	Operand 3
0	LD	M0		
1	ANI	C1		
2	MPS			
3	ANI	T0		
4	OUT	M100		
5	MRD			
6	ANI	T1		
7	OUT	T0	K100	
10	MPP			
11	AND	T0		
12	OUT	T1	K100	
15	LDP	T0		
17	OUT	C1	K5	
20	END			
21				

Figure 2-37

## 2.7 Folding windows

This sector main introduces the fives windows in folding windows. Such as [Information Output], [Search/Replace], [PLC Verify], [Cross Reference list] and [Devices List]

### 2.7.1 Information output

Information output displays compilation, communication and check results (errors and warnings).

No.	Result	Data Name	Category	Content	Step	Code
1	Error	MAIN	Check Pro...	'Y000' is double coil. It maybe can not run properly, ple...	2	0

Figure 2-38

The detailed errors will be displayed in information output windows, user can select the message and the software will automatically locate the corresponding area.

### 2.7.2 PLC Verify

PLC Verify window displays the verifying result, users can check the difference between displaying program and running program.

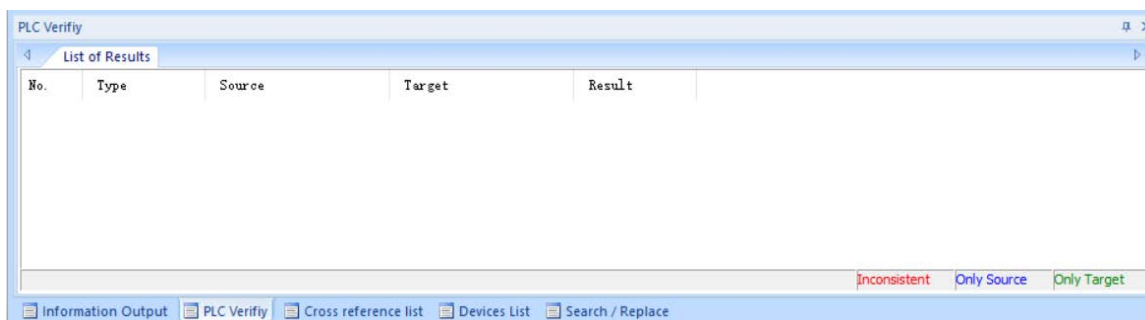


Figure 2-39

### 2.7.3 Cross reference list

Cross reference list is used for displaying specified device usage in the program, and it can also display all the devices that used in current program.

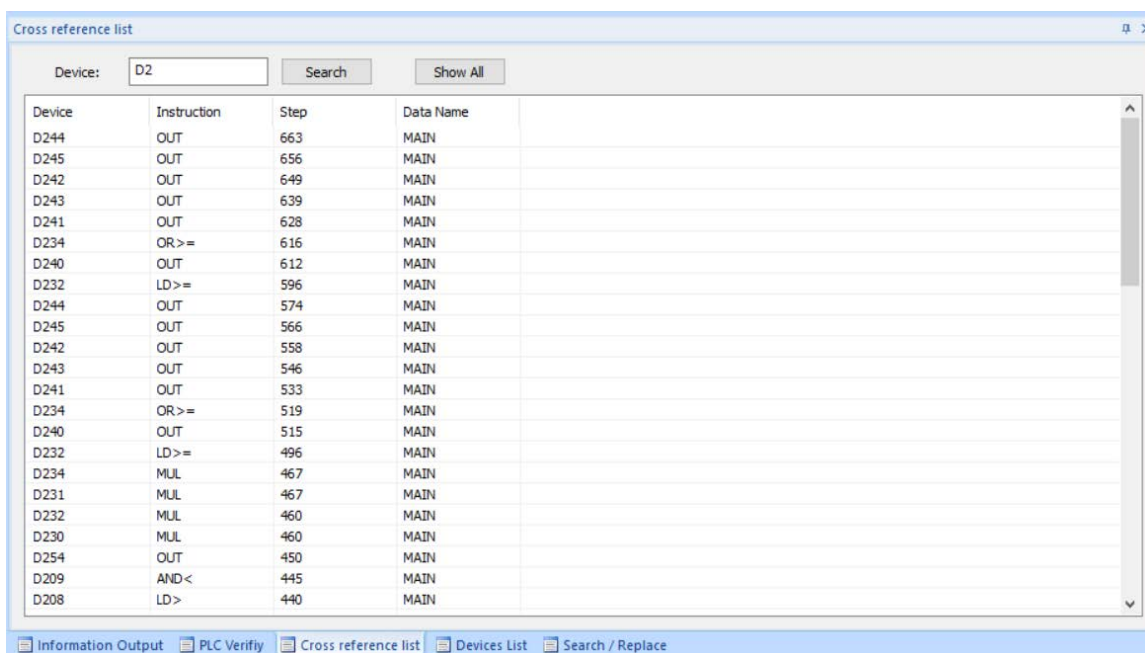


Figure 2-40

#### Description

- 1) [Device]: Users can enter the specified device in bar, and press [Enter] in keyboard, or click [Search] to display the device information. For example, enter the D2, it will display all the devices that contains D2, such as D2, D24, D203 and so on;
- 2) [Search]: Click it to execute the searching operation;
- 3) [Show All]: Click it to display all devices in current program, as Figure 2-40 shows

## 2.7.4 Device list

Device list is used for checking the used devices information in current program, such as Device, Contact, Coil (Count), Comment and Alias, and users can modify the devices in [Device list].

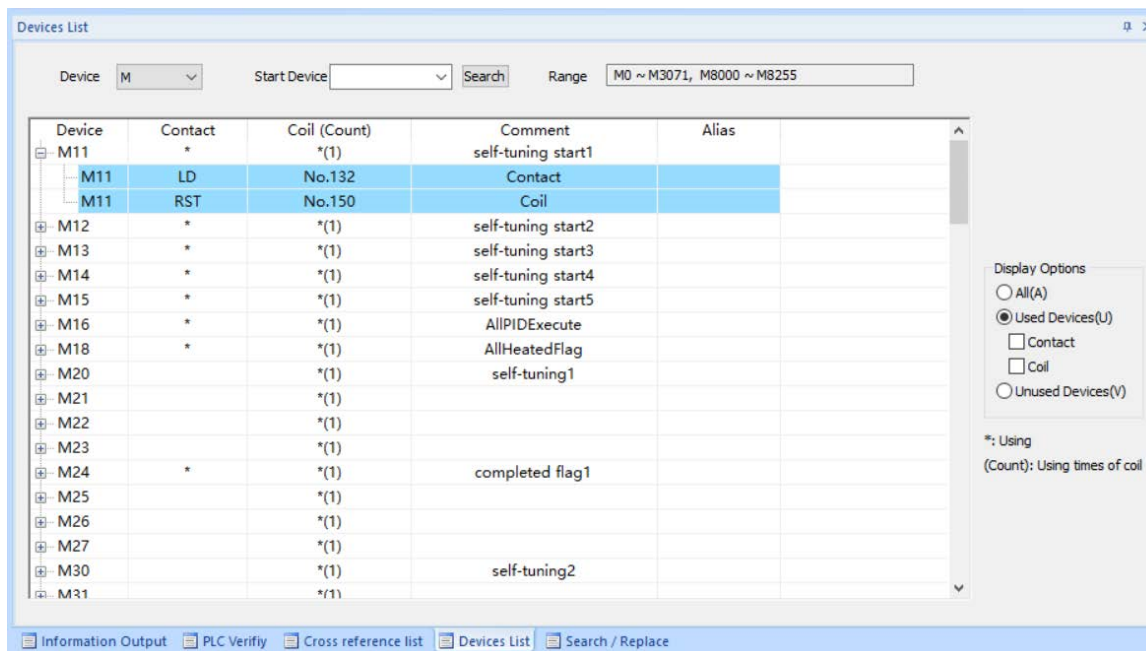


Figure 2-41

### Description

- 1) [Device]: it is drop down list menu, users can select the device type from it, such as D, C and so on, and window will display the corresponding device information.

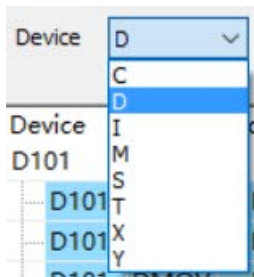


Figure 2-42

- 2) [Start Device]: It is used for setting displaying device start address; the default is from 0, such M0, C0, D0 and so on. The drop down list displays the search records.

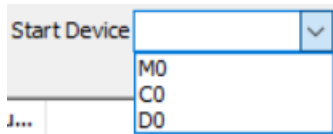


Figure 2-43

- 3) [Search]: Click it to execute search operation, it [Start Device] is empty, it will pop-up tip message.
- 4) [Range]: The searching range is displayed in this area, and it isn't editable.



Figure 2-44

- 5) [+]: Click it to view the detailed information of selected device, such as instructions, steps, and data type, double-click the item, it can jump to corresponding location.

M6	*	*(3)	Sequence control
M6	LD	No.625	Contact
M6	ANI	No.752	Contact
M6	ANI	No.860	Contact
M6	ANI	No.894	Contact
M6	ANI	No.955	Contact
M6	LDP	No.982	Contact
M6	RST	No.991	Coil
M6	ZRST	No.999	Coil
M6	ZRST	No.1007	Coil

Figure 2-45

- 6) [Comment & Alias]: The area which background is in white is used for displaying device comment and alias, users can edit the comment and alias by double-click white area.

Comment freq of sequence	Alias
Coil	
Coil	
Coil	
Coil	
Coil	
Coil	
Coil	

Figure 2-46

- 7) [Display Option]
- [All]: Check it to display all the device of selected type;
  - [Used Devices]: Check it to only display the devices that used in current program;
    - [Contact]: Only display contact that used in current program;
    - [Coil]: Only display coil that used in current program;
    - Note: if users didn't check anyone, it will display both contact and coil;
  - [Unused Device]: Check it to only display the unused device in current program;

### 2.7.5 Search/ Replace

Search/replace feature is similar to the MS OFFICE search replace function, it can locate a user-specified data block in large amounts of data, or botchily modify data, and it can greatly improve the efficiency of programming. The window as Figure 2-47 shows, there are five sub-windows.

Press [Ctrl] + [F] key in keyboard can open the window, or click [Search/Replace] in software, as Figure 2-48 shows.

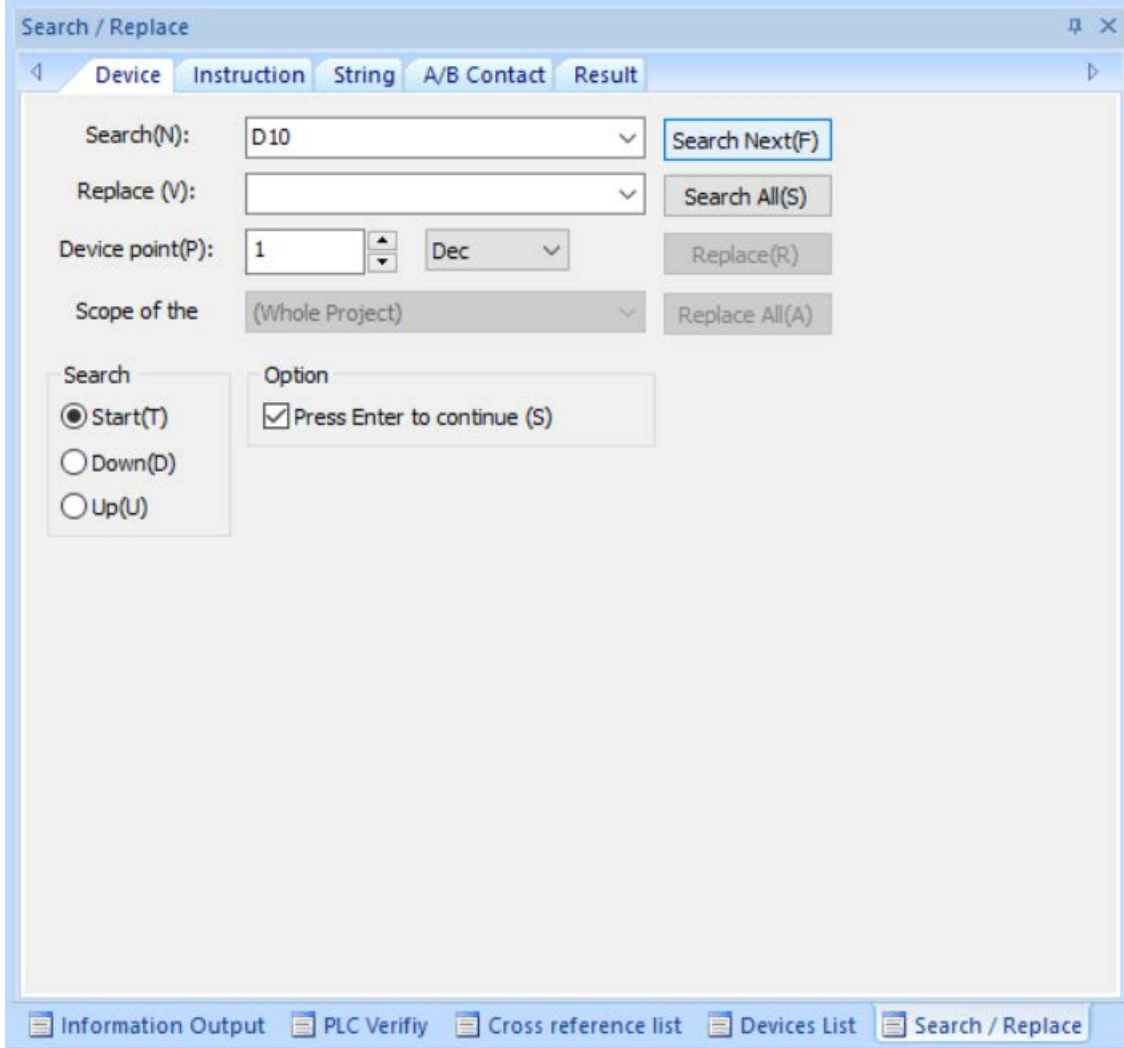


Figure 2-47

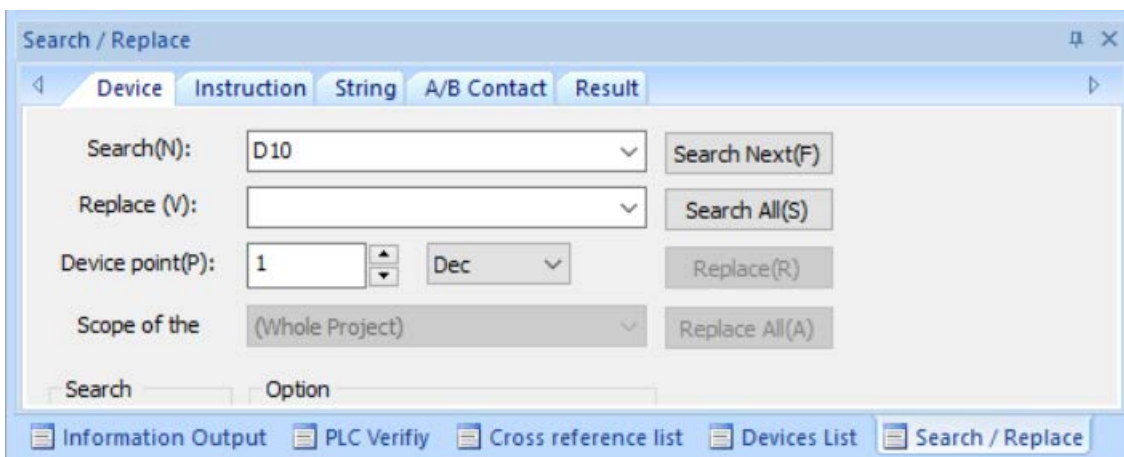


Figure 2-48

## 2.8 State bar

It display some information of software and current program, such as language, caps, state of number keyboard, communication port, PLC model, and steps information.

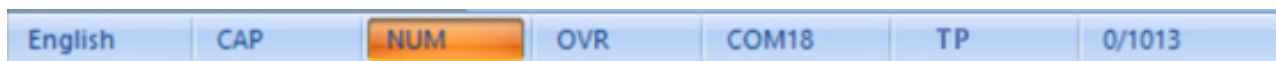


Figure 2-49

- Language: The language of software interface;
- CAP: It is for caps lock;
- NUM: It used for indicating whether the numeric keyboard is available;
- OVR: Ladder diagram edit mode.
- COM port: Software communication COM port. Click it will popup window of Communication Settings;
- TP : It is used for display PLC model;
- The steps of current program/Total steps of program: It used for displaying steps information of current program;

## 3 Project management

This chapter explains basic operations and management of projects

### 3.1 Project operations

This section explains basic operations of TETA PLC Editor such as creating, opening, and saving projects.

#### 3.1.1 Create new project

Create a new project after starting software, user need to set following parameters for new project.

- PLC Series
- PLC type
- Programming language

#### Operating procedure

- 1) Click the TETA icon on the top left corner of software interface, the menu will pop up automatically;
- 2) Select the [New];

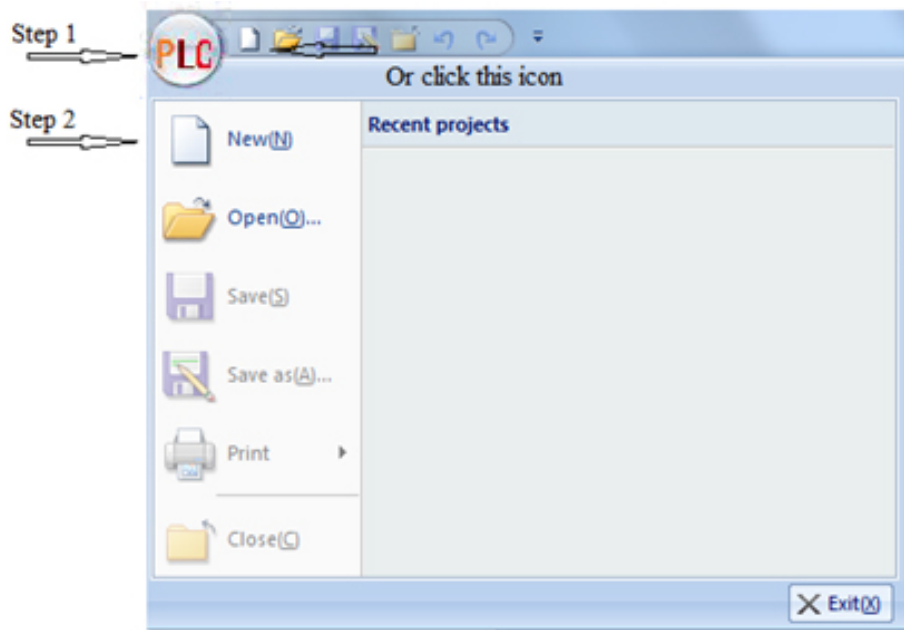


Figure 3-1

- 3) Set the parameters for new project as the Figure 3-2 shows, using TP as an example, and please click [OK] to open the main screen.

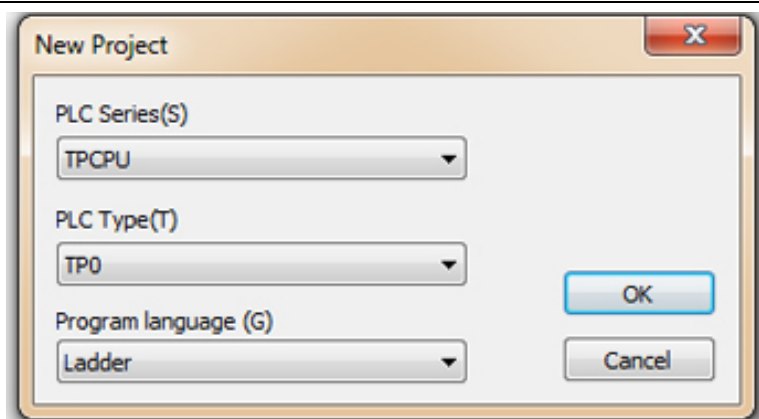


Figure 3-2

### 3.1.2 Saving project

Save a project in a hard disk or other memory media of a PC.

#### Operating procedure

- 1) Click [Save] or [Save As] in quick access bar;
- 2) Enter/ Select the folder (drive/path);
- 3) Enter project name;
- 4) Click [Save];

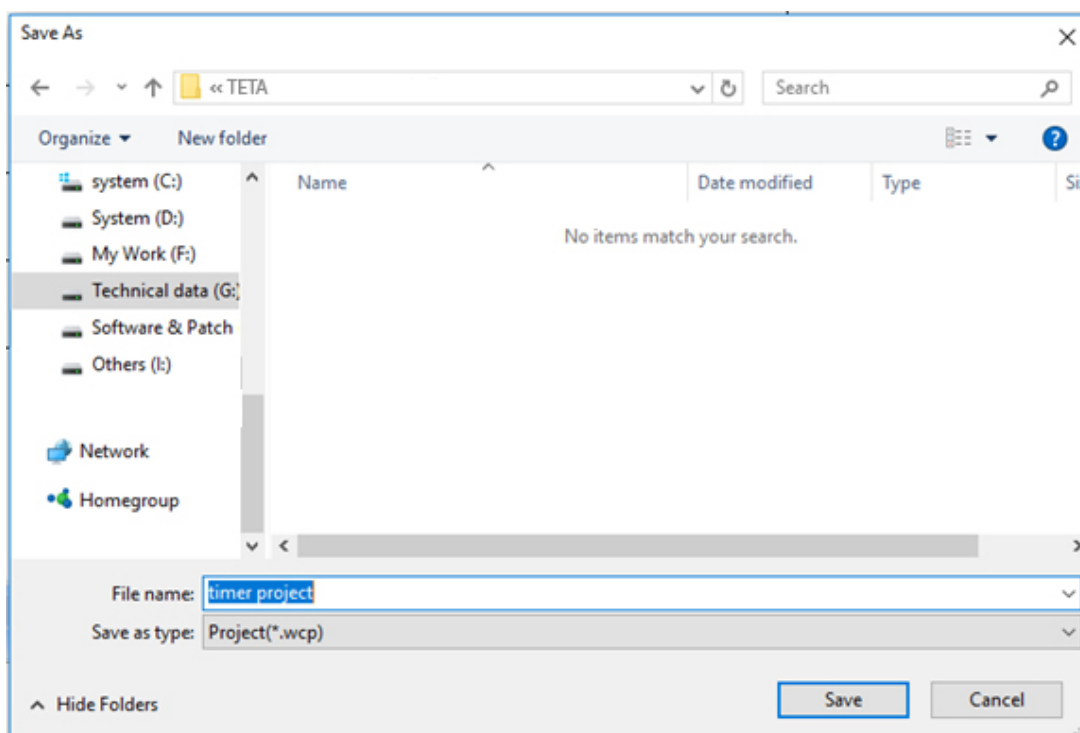


Figure 3-3

### 3.1.3 Closing project

Close an open project.



### Operating procedure

Click [Close] in quick access bar or click close button on the top right of the software main screen.

### 3.1.4 Opening existing projects

Read a project saved in a hard disk or other memory media of a PC.

#### Operating procedure

- 1) Select [Open (Ctrl+O)] in quick access bar;
- 2) Enter/ Select the folder (drive/path);
- 3) Enter/Select project name;
- 4) Click [Open];

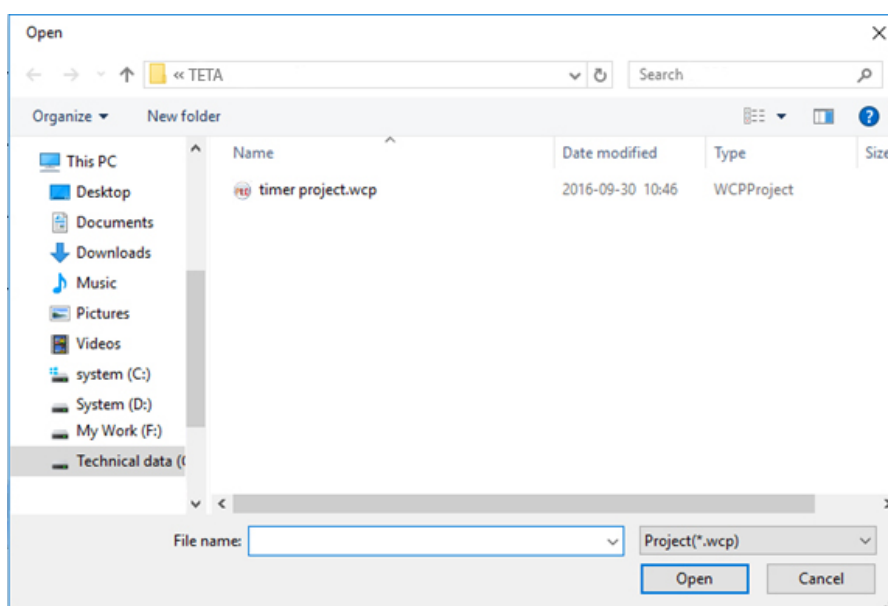


Figure 3-4

### 3.1.5 Deleting projects

Delete a project saved in a hard disk or other memory media of a PC.

#### Operating procedure

- 1) Open project saving folder;
- 2) Select projects;
- 3) Right-click and select [Delete];

### 3.1.6 Changing PLC type of project

Change the PLC type of a project being edited

#### Operating procedure

- 1) Select [Project] -> [PLC Type]
- 2) Select PLC Type

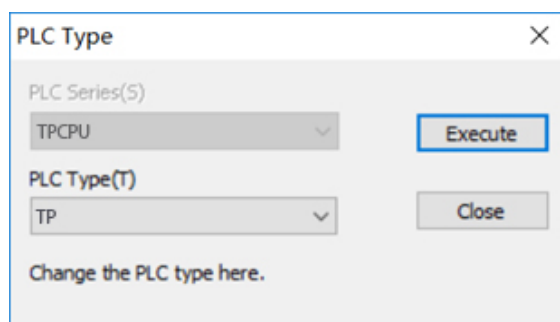


Figure 3-5

- 3) Click [Execute]
- 4) The following confirmation message is displayed. Click the [Yes] button to execute the function.

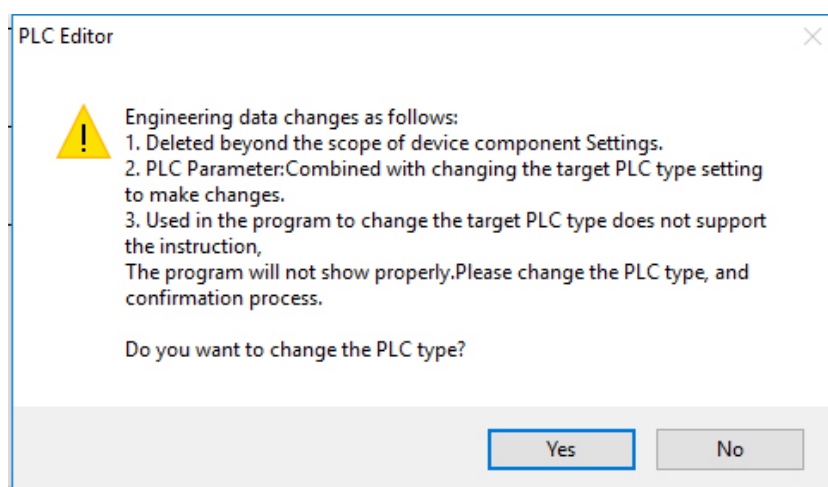


Figure 3-6

### 3.1.7 Switching programming language of project

Switch programming language of a project being edited.

#### Operating procedure

Click [Program] -> [Switch]

## 3.2 Project property

This section explains project property of TETA PLC Editor for a project.

### 3.2.1 Program

MAIN is the name of PLC program; one project only contains one program, and double click MAIN could open the programing area.

Also if there is no program, users can right click [Program] folder and select [New], as Figure 3-8 shows.

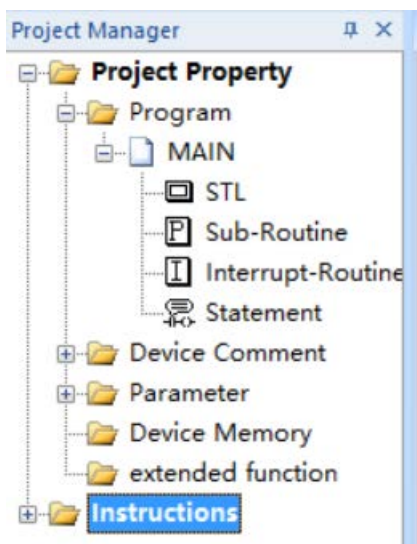


Figure 3-7

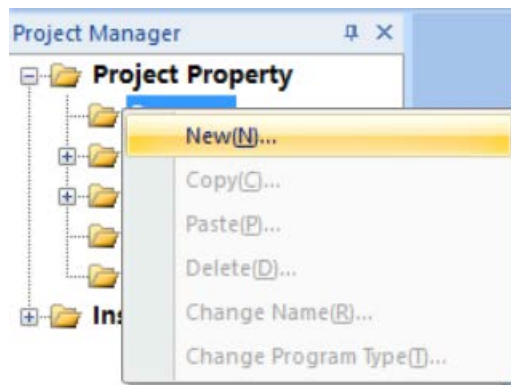


Figure 3-8

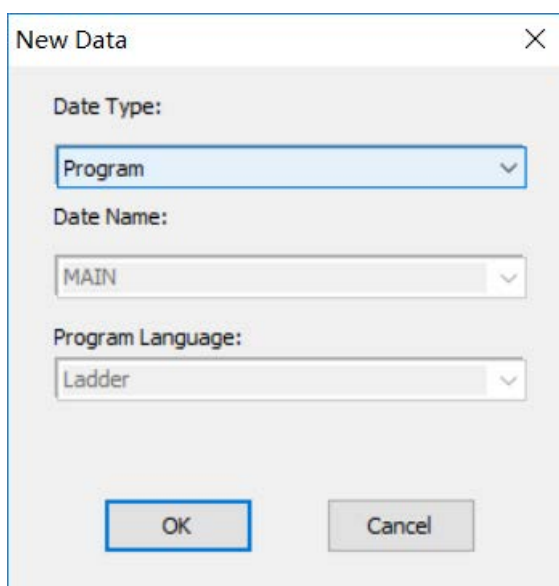


Figure 3-9

When create a new project, it will create an empty program named [MAIN] in [Program] folder, users could right click [MAIN] and delete it.

**Note: one project only has one MAIN program.**

### 3.2.2 Device Comment

One PLC project only has one [Device Comment] file, right click [COMMENT], users can delete it. And also users can create new COMMENT (The file must be named COMMENT), if there is no file in [Device Comment] folder, as Figure 3-10 shows,

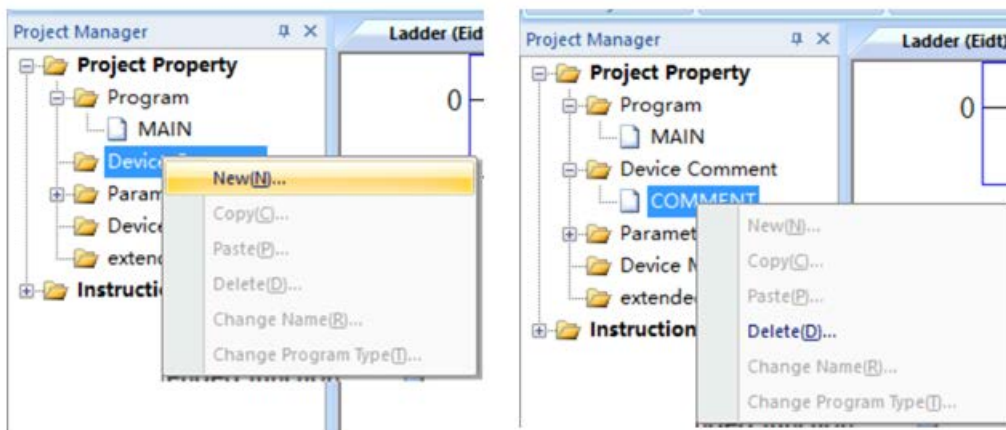


Figure 3-10

**Operating procedure**

- 1) Double click [COMMENT], the setting windows will pop up automatically, as Figure 3-11 shows;

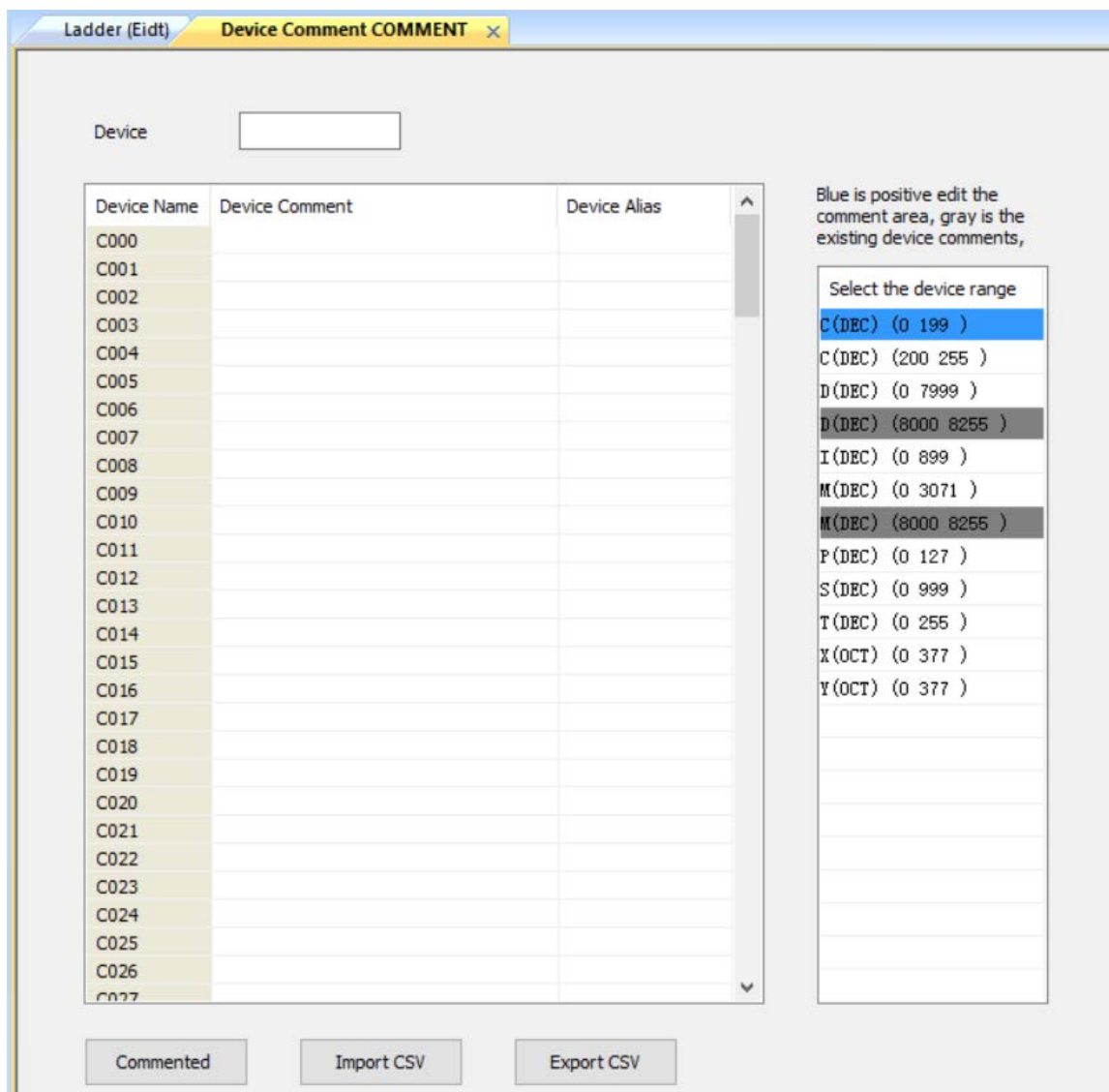


Figure 3-11

- 2) Enter the device comments and device alias one by one. Or Import the comments and alias from

existing .csv file

### Description of window

[Device]: Enter the name of the Device correctly and press [Enter] in keyboard, Device list (on the right of window) will locate the device and selected (shown in blue).

[Device display window]: It has three column, [Device Name], [Device Comment] and [Device Alias]. The column of [Device Name] is in gray, it means it is not editable, others are in white, it means it is editable. The select row in [Device display window] is in blue, and double click to edit the [Device Comment] and [Device Alias]. The maximum length for [Device Comment] is 16 characters, and [Device Alias] is 8 characters.

[Select the device range]: Software list all the devices, users can select the device in this list to change the display window. Blue indicates that the devices range is selected; black indicates that the devices have comment or alias; white indicates that the devices don't have comments section or alias. TPS and TP section of the device are different.

### 3.2.3 PLC parameter

PLC parameter is for setting PLC project parameter, one project only has one PLC parameter, users can't delete it and can't add new one.

The Figure 3-12 shows the parameter setting window, the right one is for TP.



Figure 3-12

### Operating buttons

- 1) [Check]: Check the current settings, whether the "program capacity" is greater than zero. If program capacity is less than or equal to zero, the examination is not successful;
- 2) [Default]: Click this button to set all settings to be default values;
- 3) [Finished]: Complete the PLC parameter setting;
- 4) [Cancel]: Cancel the PLC parameter settings and close the window;

#### 3.2.3.1 Memory capacity

The Memory capacity setting including Memory Capacity, Comment capacity, File registers capacity and

Program Capacity.

An error occurs when Comment capacity and File registers capacity are out of range, the modification fails.

**Project capacity= Memory capacity – Comment capacity\*500- File registers capacity\*500.** The Project capacity must be greater than zero, otherwise an error occurs and modification fails.

3.2.3.2 Device

Device window includes various types of devices. In the table the cells in white are editable; the cells in gray are not editable. The latched range is editable in TP series PLC. The setting window as Figure 3-13 shows.

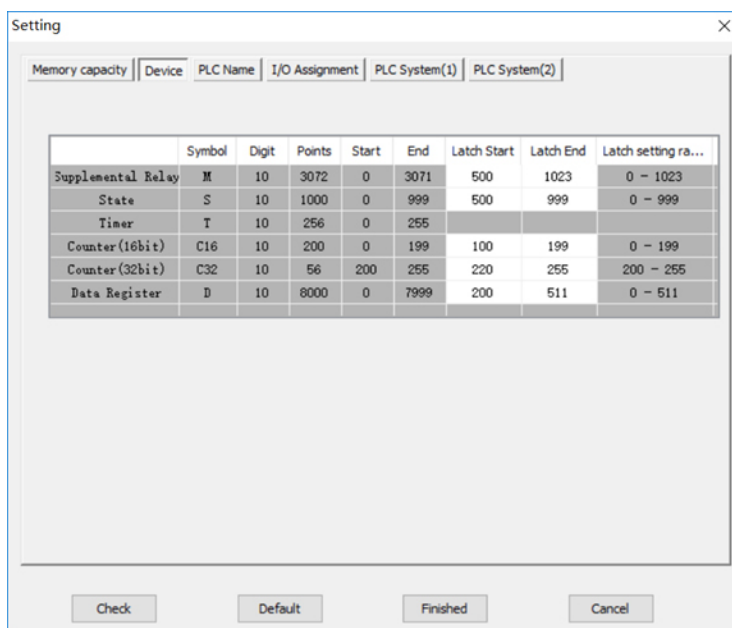


Figure 3-13

3.2.3.3 PLC name

It is used for entering the name of the PLC, it allows up to 32 bytes. The setting window as Figure 3-14 shows.

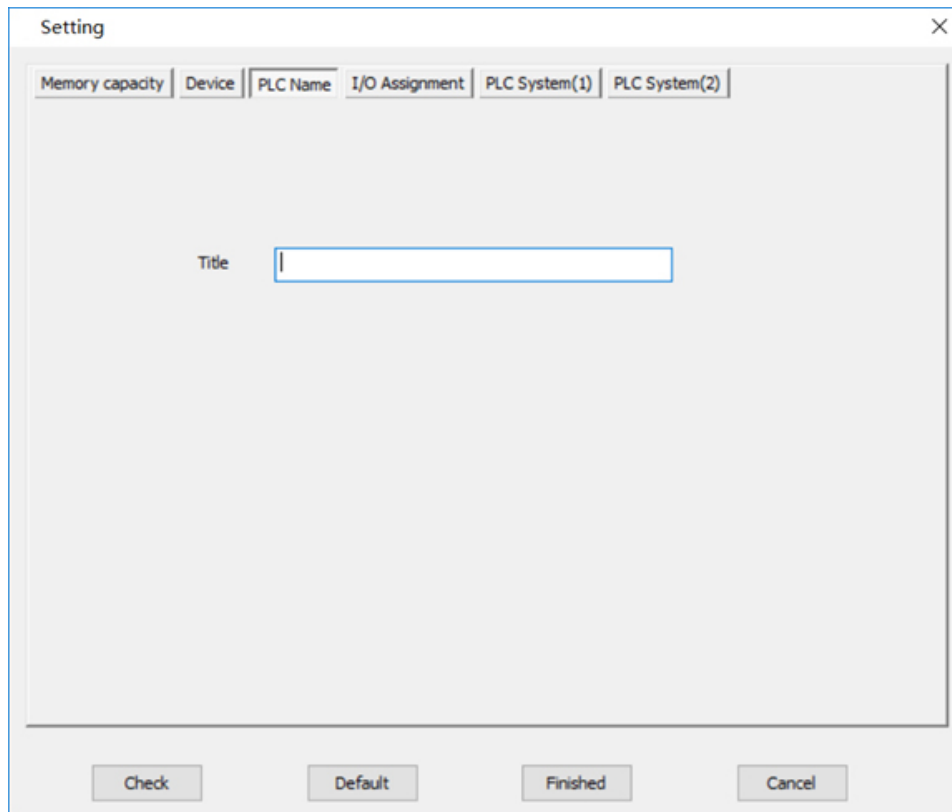


Figure 3-14

### 3.2.3.4 I/O Assignment

I/O assignment is to show the total I/O point of PLC host and extend module. The window as Figure 3-15 shows.

### 3.2.3.5 PLC System (1)

It is used for setting the PLC input terminal as RUN/STOP DIP switch. For example select X0 in this setting window, when PLC is running, if the X0 turn ON, PLC is in RUN mode, if the X0 turn off, PLC is in STOP mode, the setting windows as Figure 3-16 shows.

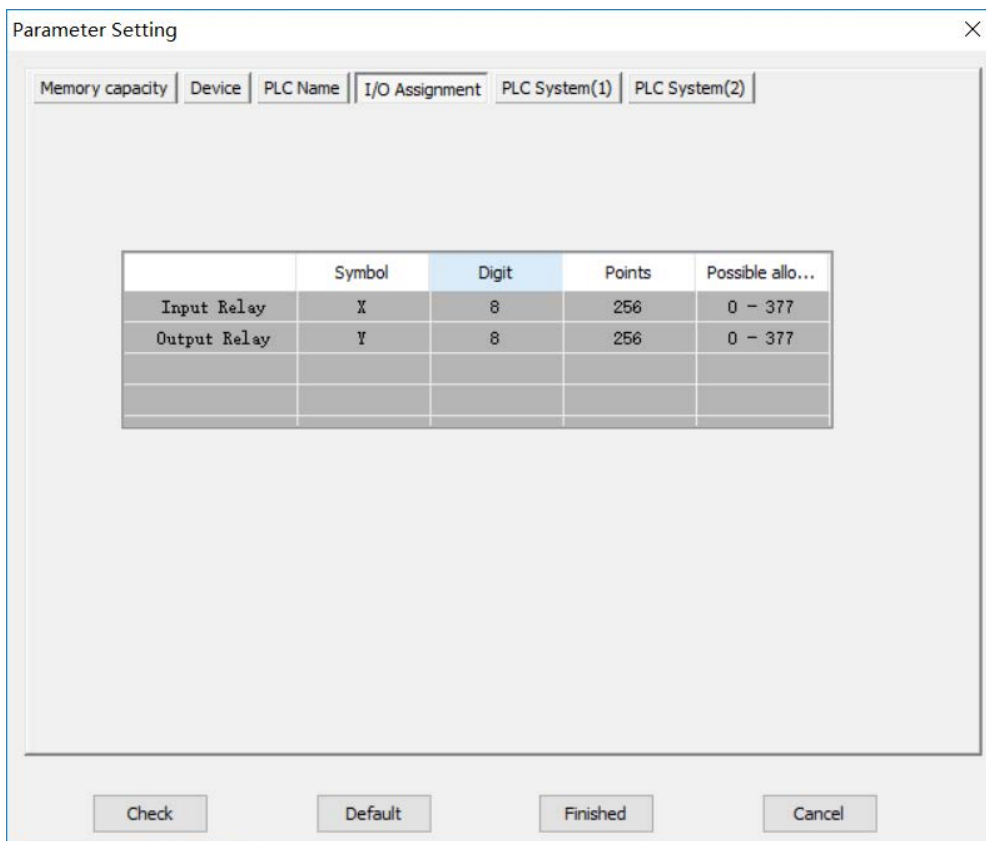


Figure 3-15

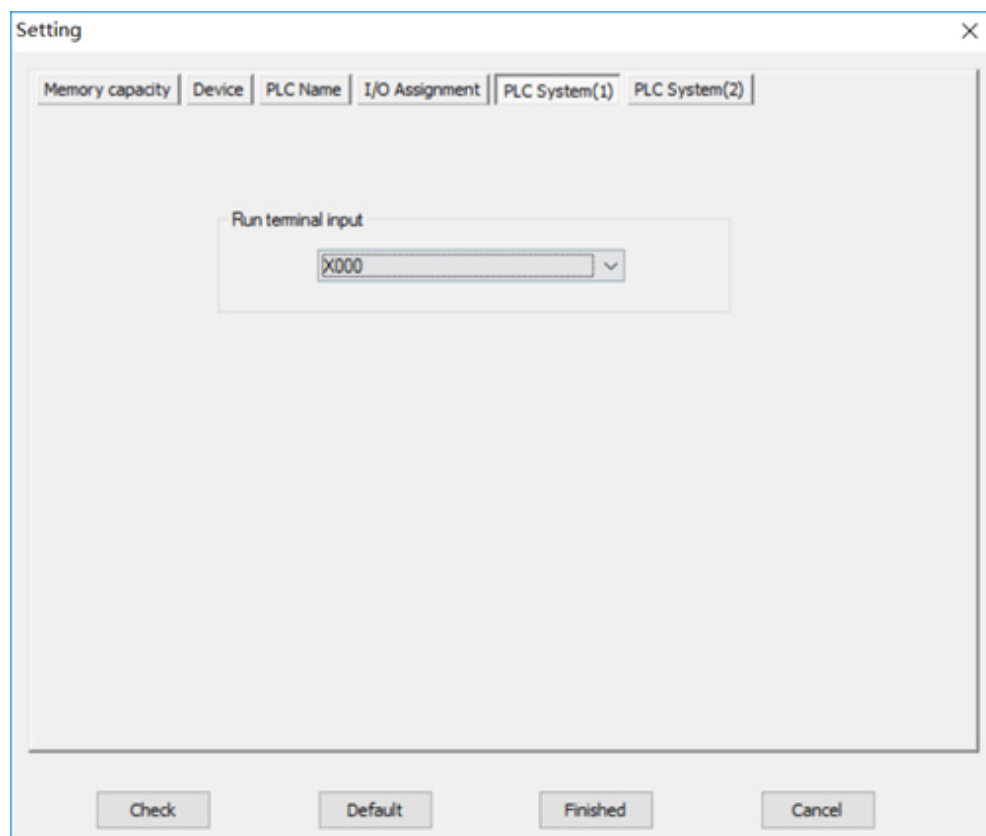


Figure 3-16



### 3.2.3.6 PLC System (2)

- 1) [Communication Setting]: Check it to enable the communication setting;
- 2) [Protocol]: Select protocol for communication
- 3) [Data Length]: This is one of communication parameter; users can select 7bit or 8bit;
- 4) [Parity Bit]: This is one of communication parameter; users can none, odd or even;
- 5) [Stop Bit]: This is one of communication parameter; users can select 1bit or 2bit;
- 6) [Transmission Rate]: This is one of communication parameter; users can select baud rate;
- 7) [Header]: Check it for enable header requirements;
- 8) [Terminator]: Check it for enable terminator requirements;
- 9) [Transmission procedure]: so far there is only one mode;
- 10) [H\W type]: It is used for select the commutation mode; users can select Regular/RS-232 and RS485
- 11) [Station number setting]: Set up the station number, range 00H~0FH (Hex)
- 12) [Control mode]: Default;
- 13) Transmission speed: Choose the transmission speed;

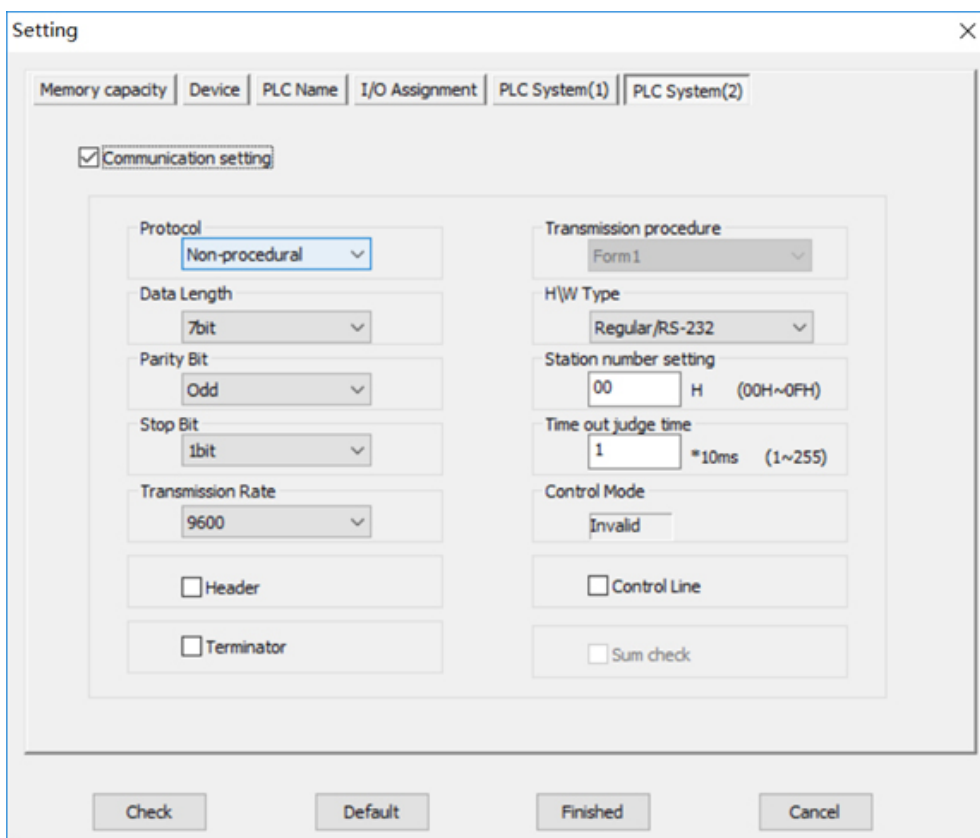


Figure 3-17

### 3.2.4 Setting Security for Projects

This section explains how to set security for projects to protect the projects themselves and the data in projects.

Setting security not only restricts an access to projects but also prevents the data, such as device comments,

and parameters, that are created by the user from erroneous modification and/or disclosure to unauthorized users.

This function can restrict writing/reading of data to/from a PLC.

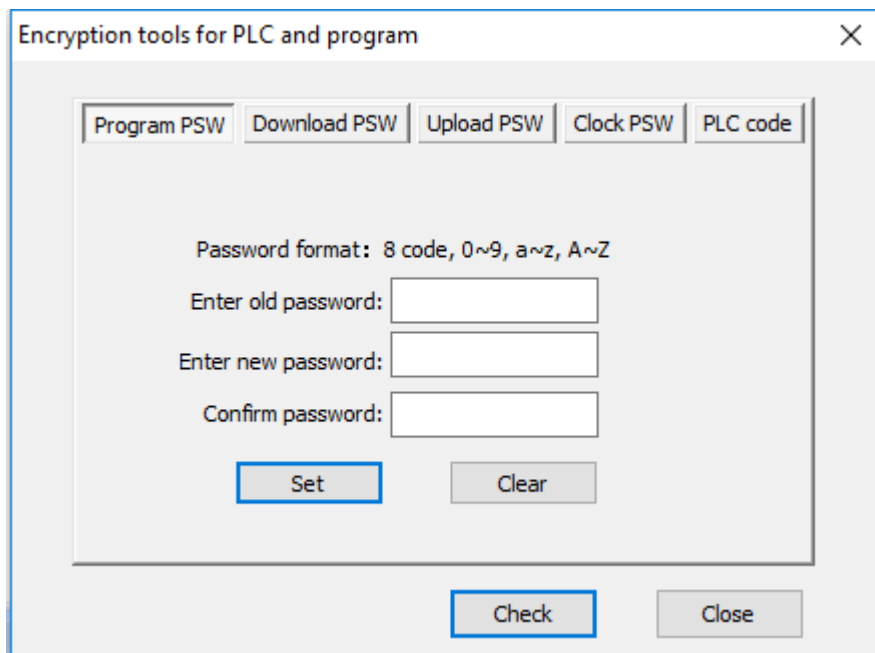


Figure 3-18

Click the [Check] button will display the current project password information, as Figure 3-19 shows.

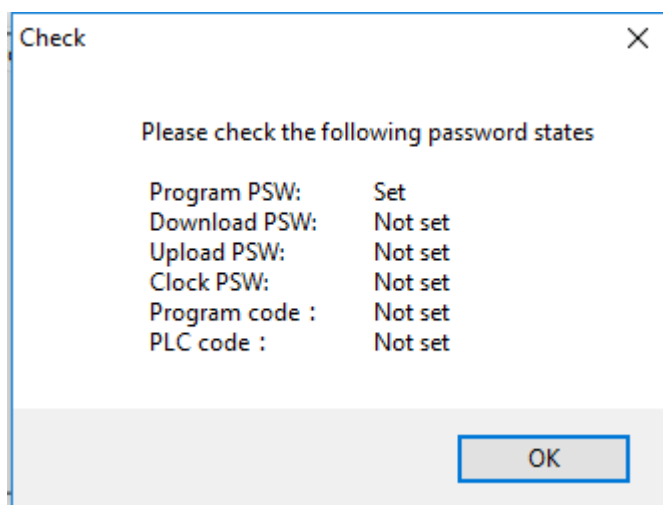


Figure 3-19

### 3.2.4.1 Program PSW

Set/Modify/Clear the program password for opening project, if project with program password, when open the project, it will pop-up password enter screen, as Figure 3-20 shows.

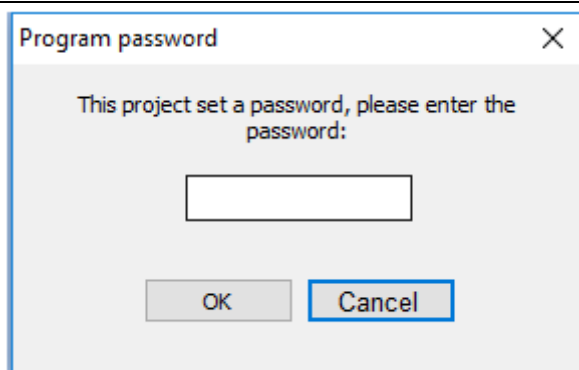


Figure 3-20

**Operating procedure for setting**

- 1) Enter the 8-code password in [Enter new password];
- 2) Enter the 8-code password in [Confirm password];
- 3) Click [Set] button;

Software will display the following two tips according to setting result.

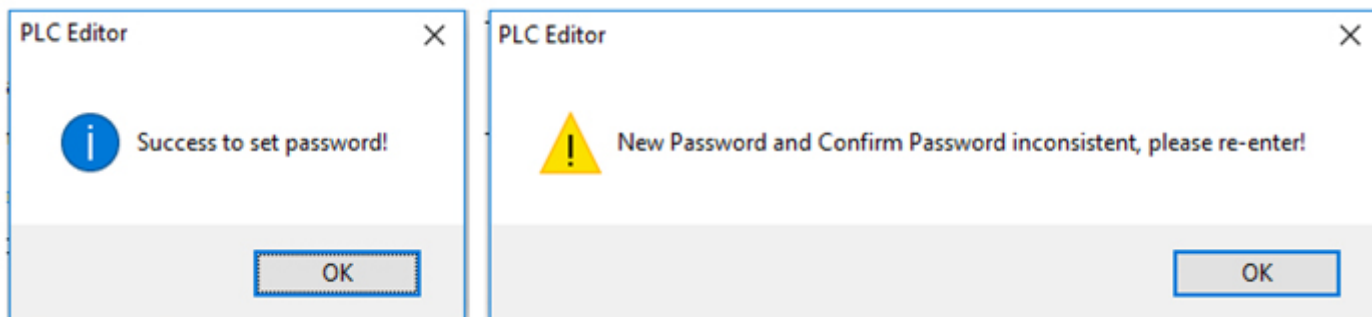


Figure 3-21

**Operating procedure for modifying**

- 1) Enter the old password in [Enter old password];
- 2) Enter the 8-code password in [Enter new password];
- 3) Enter the 8-code password in [Confirm password];
- 4) Click [Set] button;

Software will display the following three tips according to setting result, as Figure 3-22 shows.

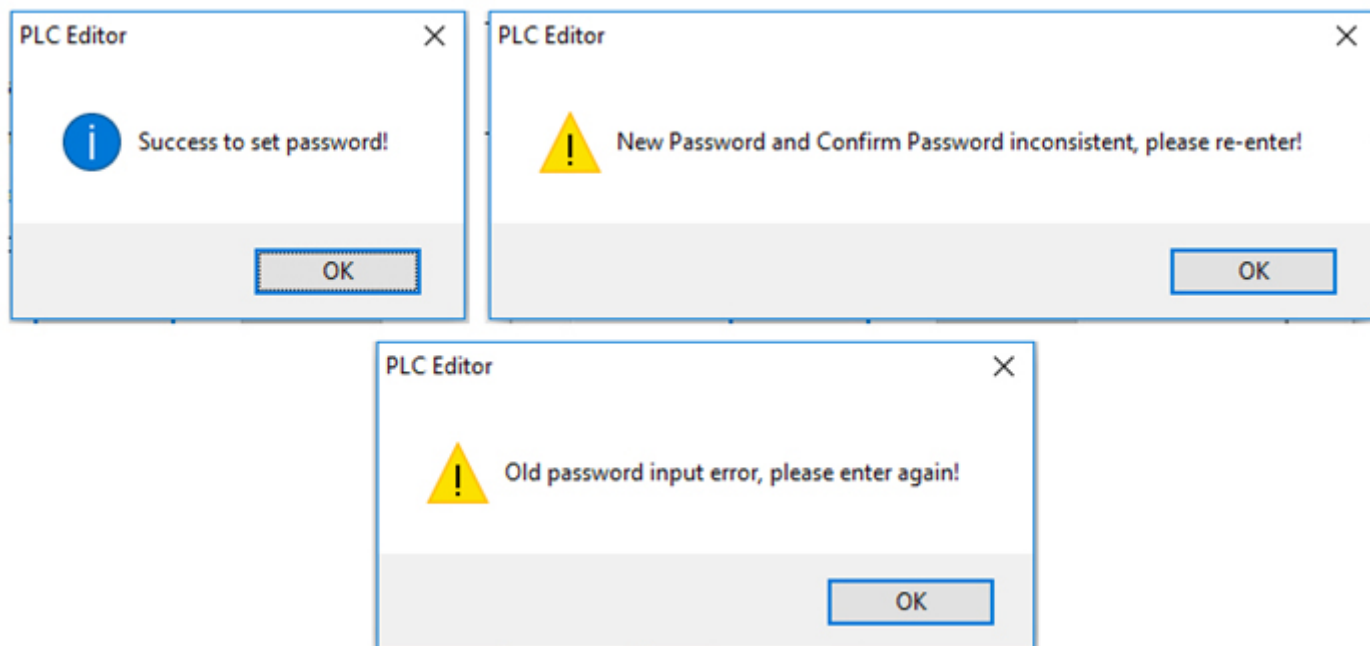


Figure 3-22

**Operating procedure for clearing**

- 1) Click [Clear] button;
- 2) Software display windows as Figure 3-23 shows;
- 3) Enter program password;
- 4) Click [OK] button;

Software will display the following two tips according to setting result, as Figure 3-34 shows.

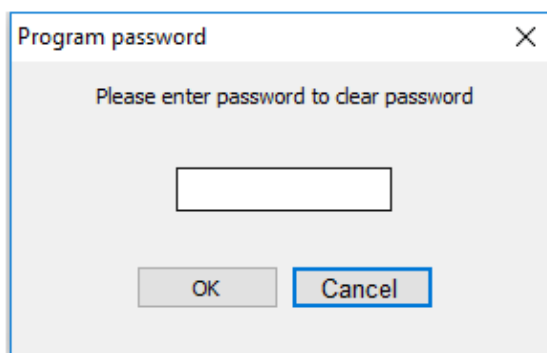


Figure 3-23

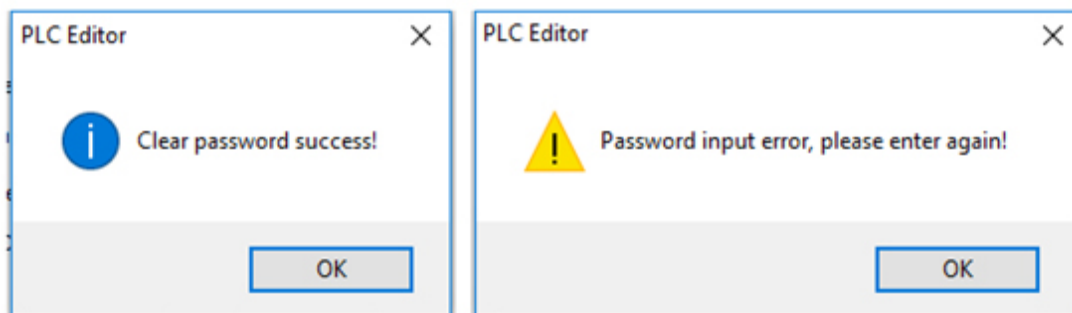


Figure 3-24

**Note:**

- 1) The password must be 8 codes;
- 2) Please make sure the new password and confirm password be the same;

**3.2.4.2 Download PSW**

Set/Modify/Clear download password for opening project, if project set download password, it requires check password before executing downloading operation.

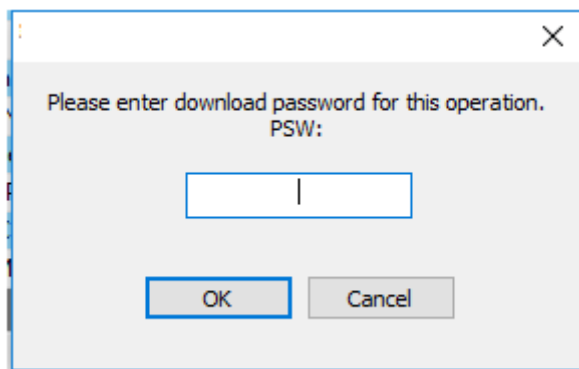


Figure 3-25

**Operating procedure**

The procedure for setting, modifying and clear are the same as program password, please refer to the

**3.2.4.3 Upload PSW**

Set/Modify/Clear upload password for opening project, if project set upload password, it requires check password before executing uploading operation.

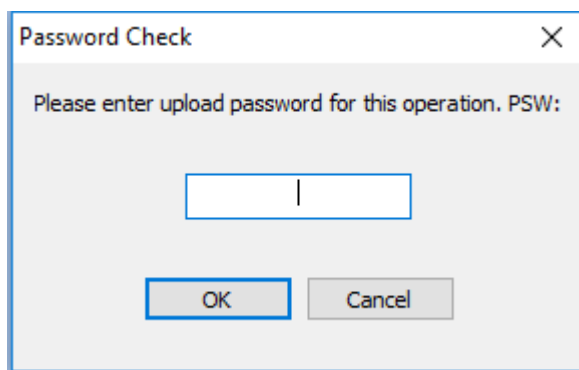


Figure 3-26

**Operating procedure**

The procedure for setting, modifying and clear are the same as program password, please refer to the [3.2.4.1](#).

**3.2.4.4 Clock PSW**

Set/Modify/Clear clock password for opening project, if project set clock password, it requires check

password before setting clock operation.

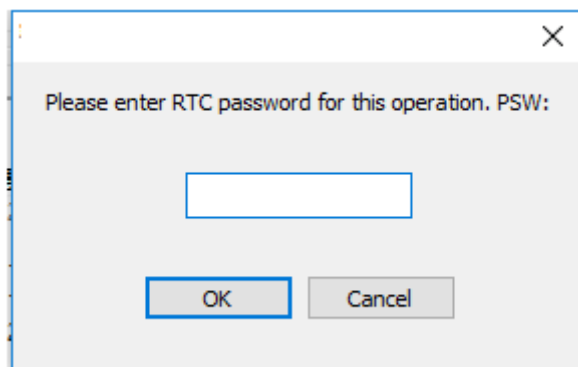


Figure 3-27

### Operating procedure

The procedure for setting, modifying and clear clock password is the same as program password, please refer to the [3.2.4.1](#).

#### 3.2.4.5 PLC code

In order to make the project can be dedicated to the designated PLC. TETA PLC Editor introduces the identification code for project and PLC device. The downloading operation only executes when the identification codes match. So the identification code can make the project only be used for specific PLC device.

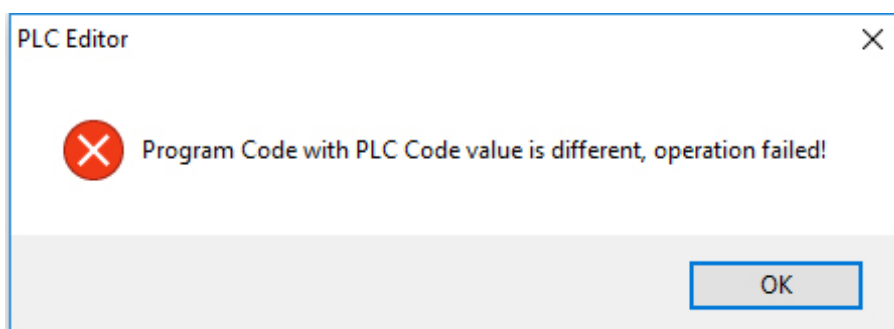


Figure 3-28

### Operating procedure

The procedure for setting, modifying and clear identification code is the same as program password, please refer to the [3.2.4.1](#).

### Note

- 1) It requires connecting PLC device for setting identification code for PLC device;
- 2) [Clear PLC Memory] can't clear PLC identification code, so please backup the identification code.

## 4 Editing programs

This chapter explains the functions of the program editors used to edit programs.

### 4.1 Programming

This sector explains the functions in programming area, including ladder program, instruction list program and right-click menu.

#### 4.1.1 Ladder program

Program the ladder program in programming area.

##### 4.1.1.1 Enter instruction in ladder

There are five methods to enter instruction in ladder.

- Direct enter
- Instruction enter
- Double-click
- Instruction help
- Ladder Symbol

##### Direct enter

- 1) Move the mouse, left-click on the cell in programming area;
- 2) Enter instruction in keyboard, such as [LD]; the [Edit] window will pop up automatically;
- 3) Enter the complete instructions and operands in [Edit] window;
- 4) Click [OK] or press [Enter] key in the keyboard;
- 5) The system will verify the enter instruction, and then generate the graphic represented by the instruction;
- 6) If the check fails, the instruction cannot be saved, please find the cause of the error;

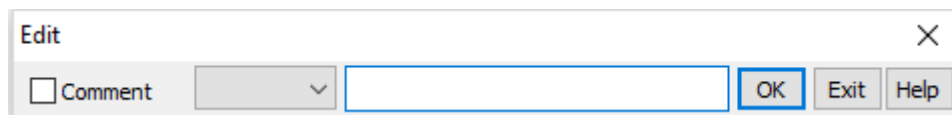


Figure 4-1

##### Instruction enter

- 1) Open the [Program Manager], which on the left of software main screen;
- 2) Open [Instruction];
- 3) Open the instruction tree;
- 4) Select the need instruction, hold the mouse to select instruction and drag it to the programming

area, placed it in a designated location;

- 5) The [Edit] window will pop up automatically with selected instruction, as Figure 4-2 shows;

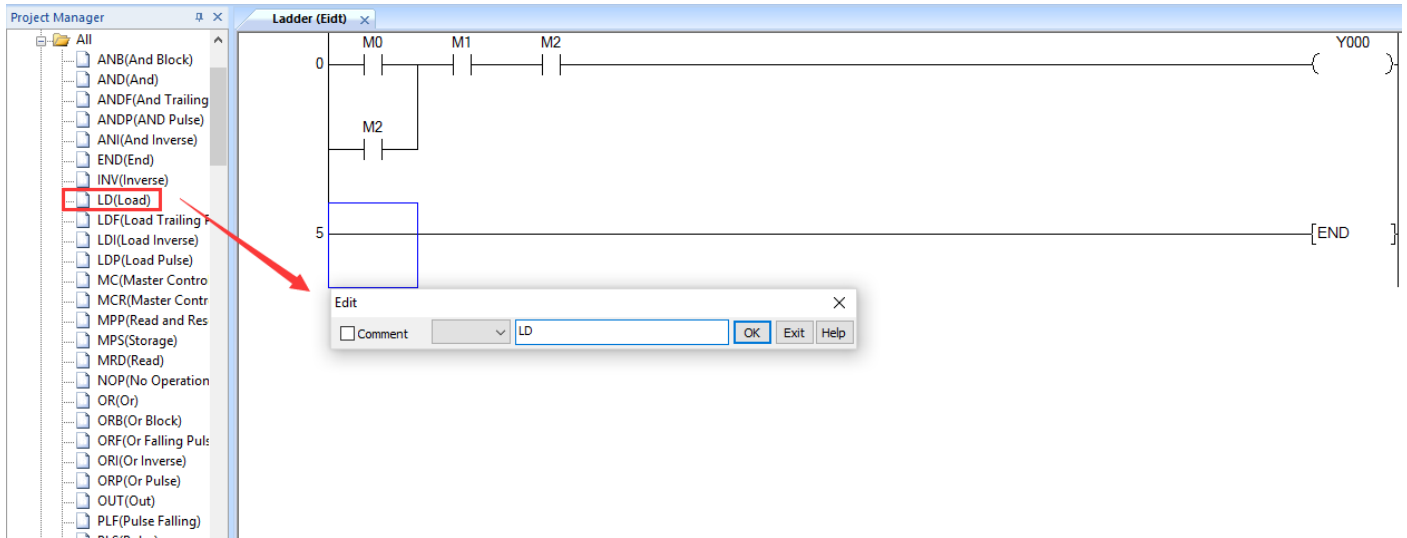


Figure 4-2

- 6) Enter the operands in [Edit] window;
- 7) Click [OK] or press [Enter] key in the keyboard;

**Double-click**

- 1) Move the mouse to the designated location, and double-click mouse;
- 2) The [Edit] window will pop up automatically;
- 3) Enter the complete instructions and operands in [Edit] window;
- 4) Click [OK] or press [Enter] key in the keyboard;
- 5) The system will verify the enter instruction, and then generate the graphic represented by the instruction;
- 6) If the check fails, the instruction cannot be saved, please find the cause of the error;

**Instruction help**

- 1) Move the mouse to the designated location, and double-click mouse;
- 2) The [Edit] window will pop up automatically;
- 3) Click [Help] in [Edit] window, as Figure 4-3 shows, it will pop up [Help] windows for all the PLC instructions, as Figure 4-4 shows;

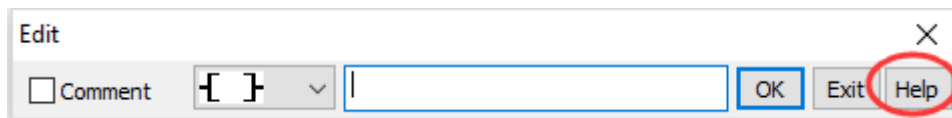


Figure 4-3



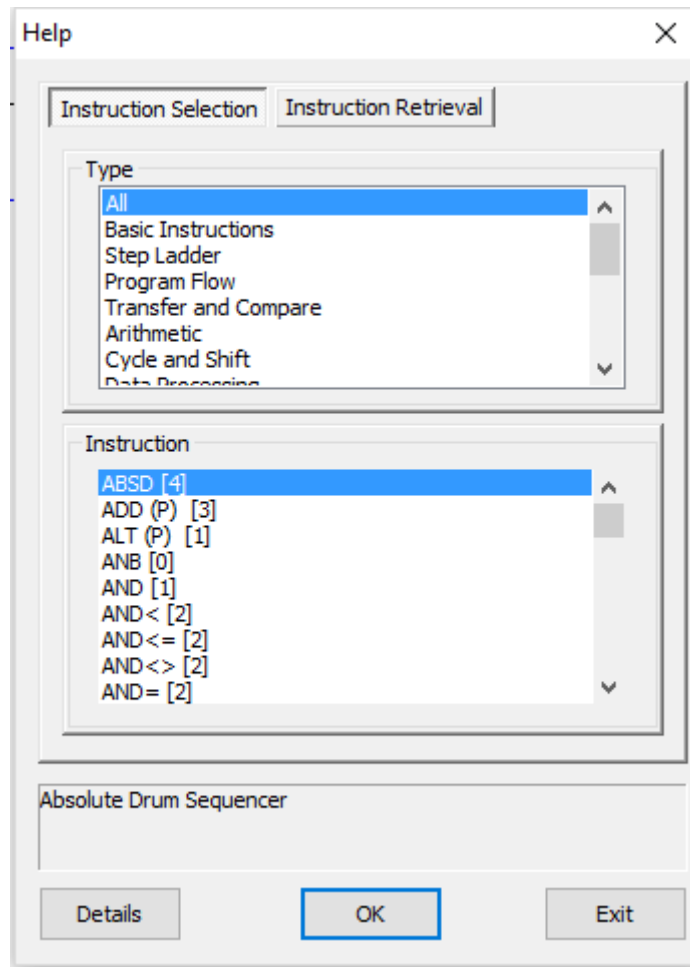


Figure 4-4

- 4) Select the needed instruction, and click [Details] to open the instruction guide window as Figure 4-5 shows;
- 5) Enter the right operands, and click [OK] to save instruction;

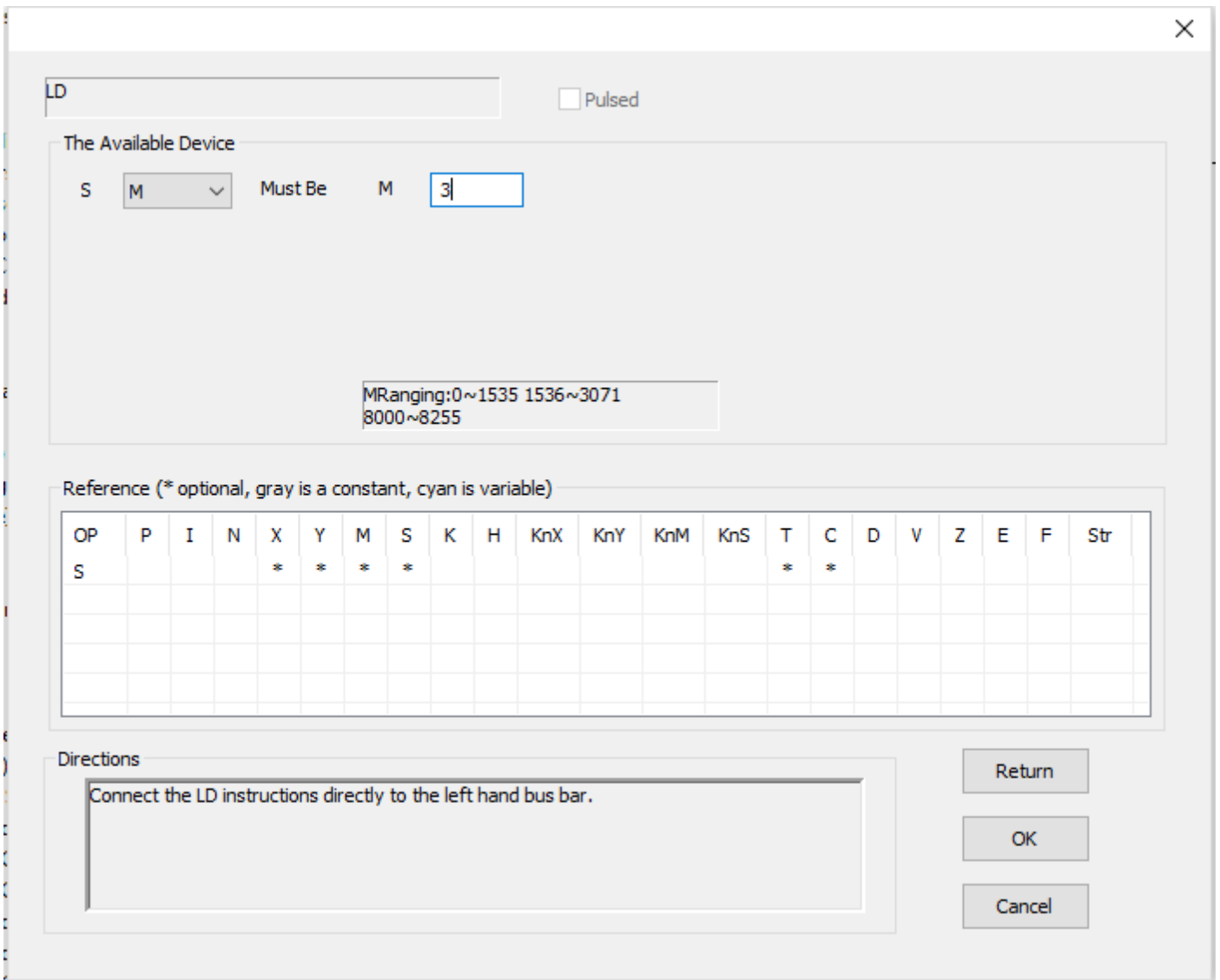
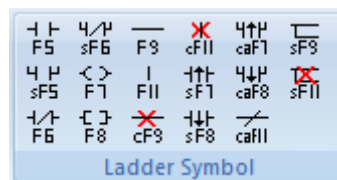


Figure 4-5

### Ladder Symbol

There is [Ladder Symbol] toolbar for ladder editing.



- 1) Click instruction icon[Ladder Symbol], the [Edit] window pop up automatically;
- 2) And the enter operand in [Edit] window
- 3) Enter the operands in [Edit] window;
- 4) Click [OK] or press [Enter] key in the keyboard;
- 5) The system will verify the enter instruction, and then generate the graphic represented by the instruction;

- 6) If the check fails, the instruction cannot be saved, please find the cause of the error;

**Note**

- 1) Some instructions only be enter in the instruction list (IL), cannot enter in the ladder, such as MPS 、MPP. When enter such instructions in ladder editing area, it can generate the graphic represented by the instruction;
- 2) Since the coils and application instructions represent the end of a line of ladder program, so users cannot enter any device behind them;
- 3) Due to the vertical line can take up two rows of space. So if there is device in second line, this vertical line cannot be entered;
- 4) The maximum number of columns is 13 in ladder program;

**4.1.1.2 Modify/Delete devices in ladder**

**Operating procedure of modify devices**

- 1) Ensure software is in [Edit Mode];
- 2) Select the device by mouse;
- 3) Double-click device or press [Enter] key in keyboard to open the [Edit] window;
- 4) Enter the new device or operand
- 5) Click [OK] to save the modification

**Note:**

- 1) Modifications must be able to verify the correct instruction, it can be saved.
- 2) Contact devices of occupying a cell (normally open, normal close, rising edge, falling edge, inversion, Stepping) should not be modified to the compared devices.
- 3) All of the contact device can't modify each other with coil or application instructions.

**Operating procedure of delete devices**

- 1) Ensure software is in [Edit Mode];
- 2) Select the delete area by mouse;
- 3) Right-click and select [Delete], or press [Delete] key in keyboard;
- 4) Delete horizontal line and vertical lines, please select the lines, and then click the corresponding icon in [Ladder Symbol] toolbar or press [F9] and [F11] in keyboard;

**4.1.1.3 Copy and paste function**

When ladder needs a large number of the same device, use copy and paste function can improve programming efficiency.

**Operating procedure**

- 1) Enter the program programming area;
- 2) Select the copy content in program programming area;
- 3) Select [Copy] in right-click Menu or in [Edit] toolbar, or click [Ctrl] + [C] in keyboard to copy the contents to the clipboard;

- 4) Paste the contents to the certain area;

**Note:**

- 1) When there is a vertical in copying area, only when the vertical across the top and bottom cells is within the range of choice. The vertical will be copied to the clipboard.
- 2) No matter to use which edit mode, when you perform paste function, only to use the overlay mode. Make the devices clear within the range of pasting, and then paste every devices of the clipboard one by one.

**4.1.2 Instruction list programming**

TETA PLC Editor Software not only supports ladder, but also supports instruction list. Instruction list is used for enter of instruction and operand. The way of enter instruction is the same as ladder. System will calculate the steps of instruction, and automatically align instruction, operands, when completed programing. The programming area as Figure 4-6 shows.

Step	Instruction	Operand
0	LD	M0
1	OR	M1
2	OUT	Y000
3		

Figure 4-6

**4.1.2.1 Entering instructions in instruction list**

Enter instruction in programming area

**Operating procedure of entering instruction**

- 1) Create a new project;
- 2) Click [Switch] in [Program] toolbar into Instruction list editing mode;



Figure 4-7

- 3) If there is uncompiled program in the ladder editing mode, please click [Compile] in [Program] toolbar firstly;
- 4) Move the mouse to the end row of instruction list editing area, and double-click to open [Edit] window;

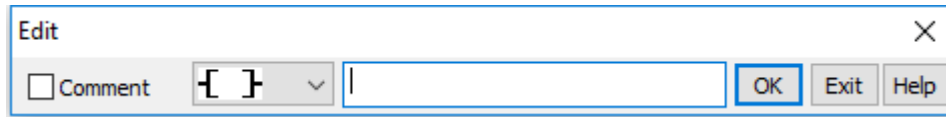


Figure 4-8

- 5) Enter the instruction and operand in [Edit] window, and click [OK] to save the instruction and operand;

#### 4.1.2.2 Copy, cut and paste instructions

This feature is completely consistent with the function of windows, the contents of the copy, cut to the clipboard, and then paste to the user specified location.

- 1) Create a new project and switch to instruction list editing mode.
- 2) Select the area or element that need to be copied or cut, and click [Ctrl] + [C] or [Ctrl] + [X] on the keyboard.
- 3) Move the mouse to user specified location. And then click [Ctrl] + [V].

#### 4.1.3 Right-click menu

In the program editing area, right click mouse will display menu, this menu contains commonly functions. It can greatly improve the development efficiency of developers.

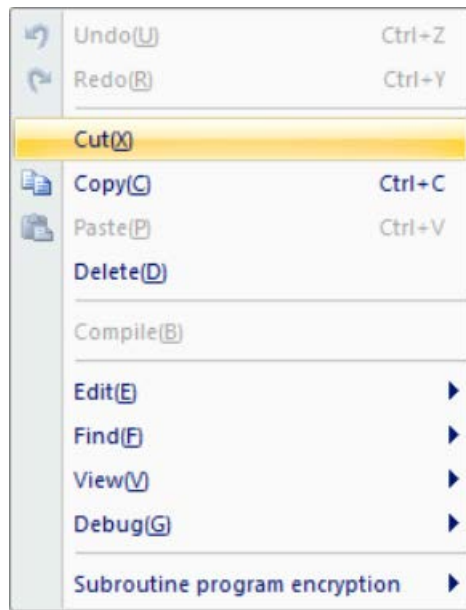


Figure 4-9

- Undo: Cancel the current operation and back the previous step;
- Redo: Undo operation;
- Cut: Cut the selected ladder in the ladder editing area. Cut the selected command line in the list of instruction. The content is store in the clipboard;
- Copy: Copy the selected ladder diagram in the ladder diagram. Copy the instruction row in the instruction list editing area. The content is stored in the clipboard;
- Paste: Paste the data in the clipboard;
- Delete: Delete the program in editing area;

- Compile: Compile the current program;

#### 4.1.3.1 Edit

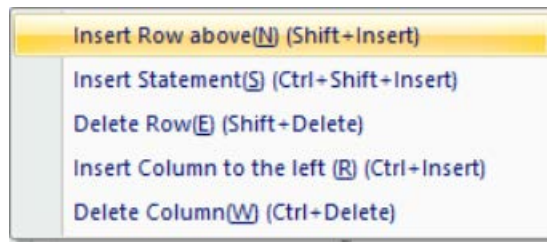


Figure 4-10

- Insert Row above: Add a new row above in the certain row;
- Insert Statement: Add a statement for certain row;
- Delete Row: Delete a certain row;
- Insert Column to the left: Add a new column on the left of certain column;
- Delete Column: Delete a certain column in the PLC program;

#### 4.1.3.2 Find

There are three options in Find menu, and all of them could open the [Search/Replace] window, but different option for different function.

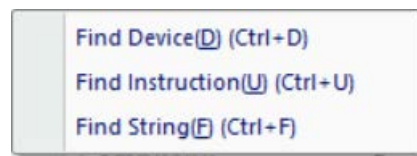


Figure 4-11

- **Find Device:** Open the [Search/Replace] window, and the default search page is Device. And if users selected device, this device will be display in [Search] bar, otherwise, [Search] bar is empty;

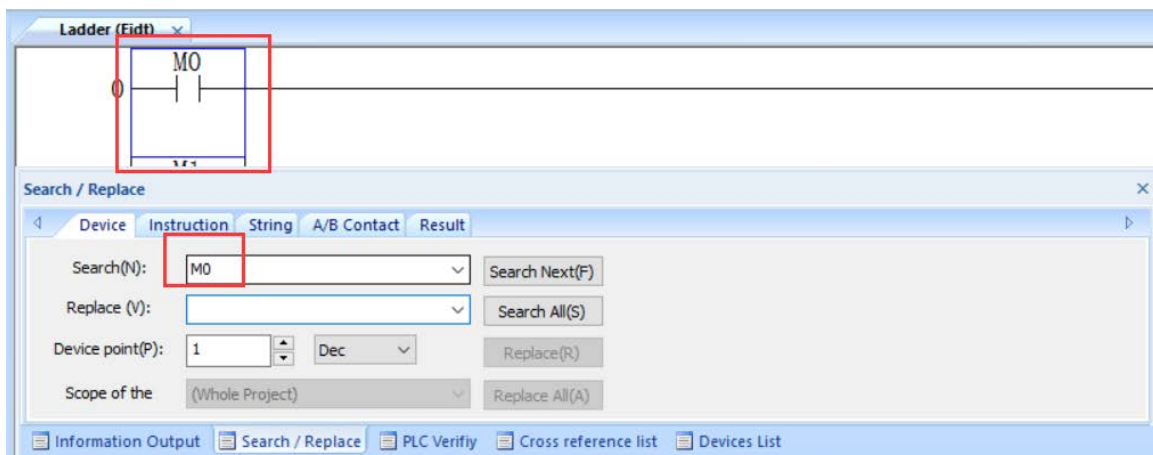


Figure 4-12

- **Find Instruction:** Open the [Search/Replace] window, and the default search page is Instruction. And if users selected instruction, this instruction will be display in [Search] bar, otherwise, [Search] bar is empty;

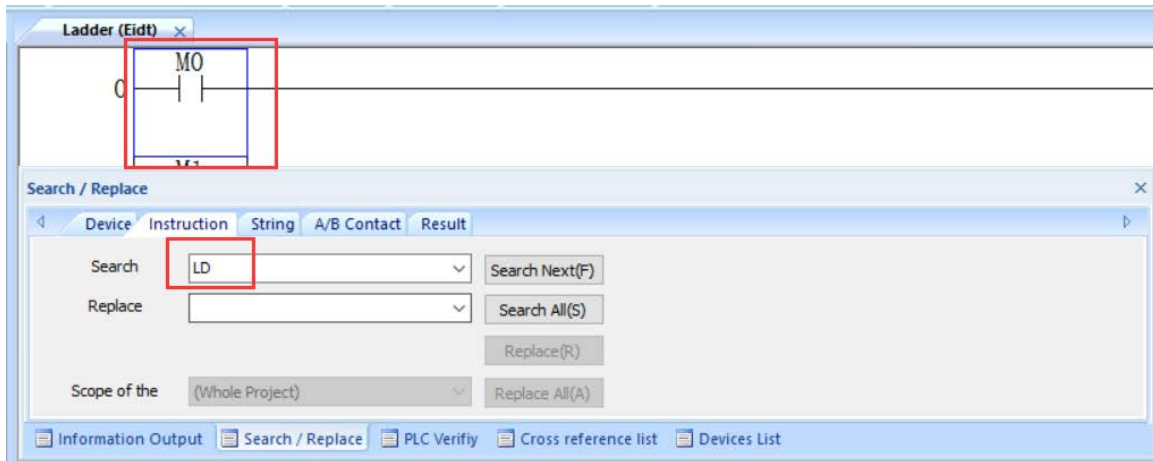


Figure 4-13

- **Find String:** Open the [Search/Replace] window, and the default search page is String. And if users selected instruction, this instruction will be display in [Search] bar as string, otherwise, [Search] bar is empty;

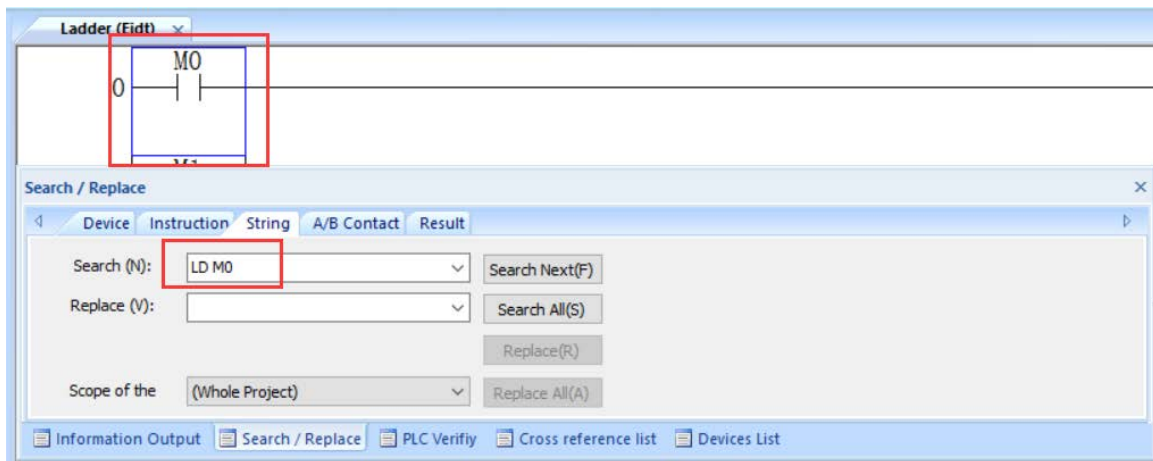


Figure 4-14

### 4.1.3.3 View

It used for setting display content, the menu as Figure 4-15 shows.

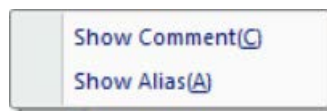


Figure 4-15

- Show Comment: The device comment is displayed in ladder (Instruction list is not available).
- Show Alias: The alias of device is displayed in ladder and instruction list.

### 4.1.3.4 Debugging



Figure 4-16

Modify the current value: Users can change the device value in monitoring mode and simulator mode (Write mode is not available).

#### 4.1.3.5 Subroutine program encryption

It is used for the part of the ladder program encryption; it can be used for encrypting the subroutine and interrupt program. TETA PLC Editor provides encryption function in [Tool] toolbar, including program password, upload password, download password; all are for the entire ladder program.



Figure 4-17

- **Encrypt**

It is used for set password for subroutines, when subroutines are encrypted; they are not displayed in programming area, as Figure 4-18 shows.

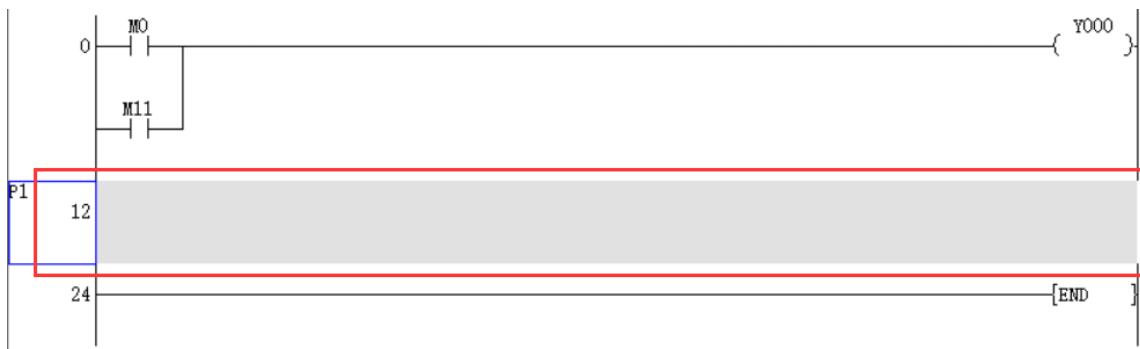


Figure 4-18

#### Operating procedure

- Select the subroutine;
- Click subroutine pointer, such as P1;
- Right click, and select [Subroutine program encryption] -> [Encrypt];
- Enter the password;
- Click [OK] button;

- **Decrypt**

Decrypt the encrypted subroutine, it requires enter password, when subroutine decrypted, it is displayed in programming area.

#### Operating procedure

- Select the encrypted subroutine;



- b) Click subroutine pointer, such as P1;
- c) Right click, and select [Subroutine program encryption] -> [Decrypt]
- d) Enter the password;
- e) Click [OK] button;

● **Modify passwords**

Modify the passwords for encrypted subroutine, it requires enter old password.

**Operating procedure**

- a) Select the encrypted subroutine;
- b) Click subroutine pointer, such as P1;
- c) Right click, and select [Subroutine program encryption] -> [Modify passwords]
- d) Enter the old password;
- e) Enter the new password;
- f) Click [OK] button;

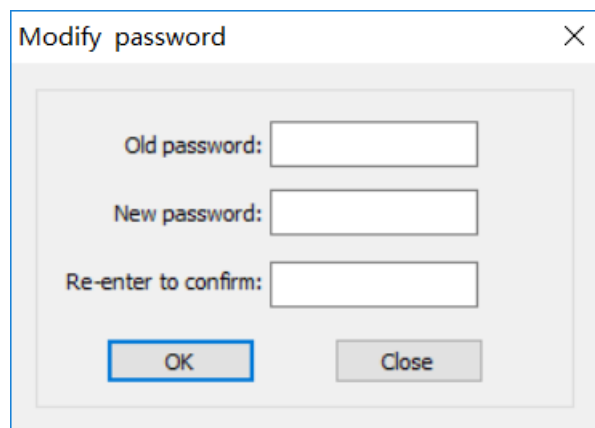


Figure 4-19

● **Show encrypted content this time**

It used for showing encrypted content in programming area, it requires enter password for displaying.

**Operating procedure**

- a) Select the encrypted subroutine;
- b) Click subroutine pointer, such as P1;
- c) Right click, and select [Subroutine program encryption] -> [Show encrypted content this time]
- d) Enter the password;
- e) Click [OK] button;

## 4.2 Labels

This sector explains how to edit labels in programming area, including device comments and program statements.

### 4.2.1 Comment

Edit comment for devices, its prerequisite is a program.

#### Operating procedure

- 1) Click [Comment], if the background color of the [Comment] changed, then this function is enabled. If the background does not change, please save the current project and restart the software as administrator;
- 2) When enable edit comment mode, and double click the device, it will pop-up [comment edit] window;
- 3) Enter the comment for this device;
- 4) Click [Execute] to save;

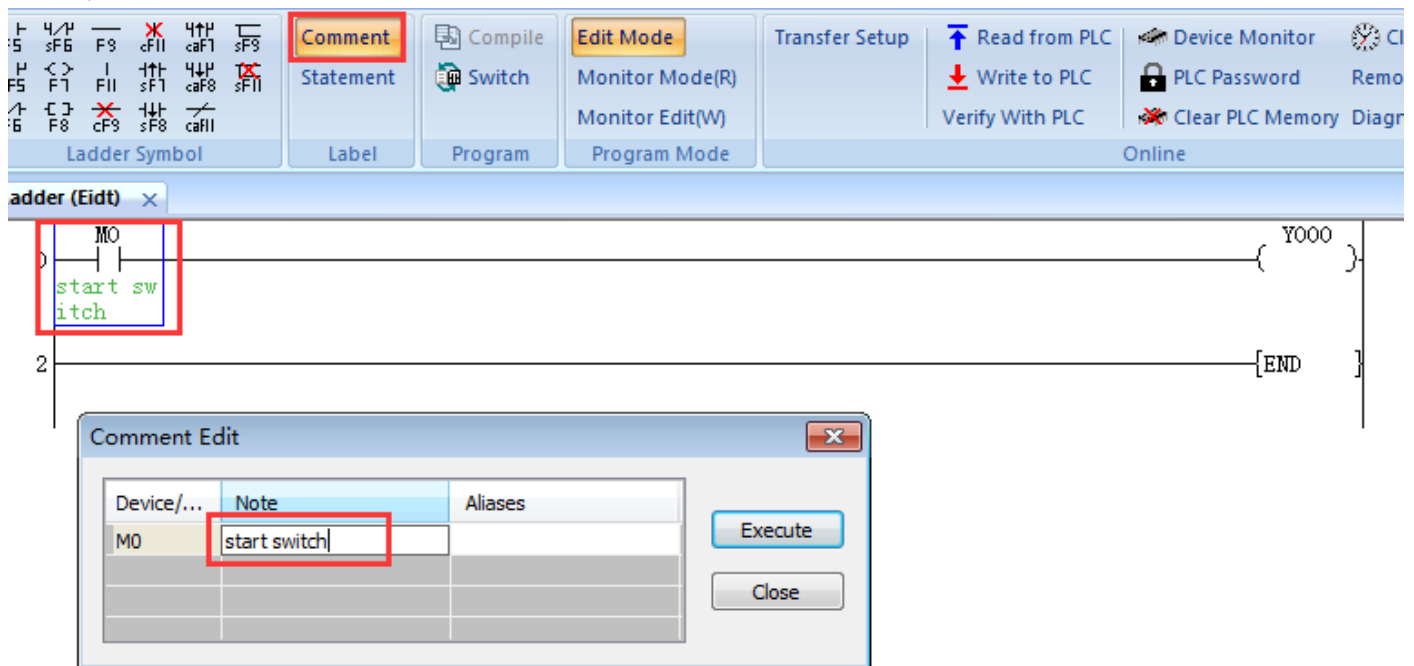


Figure 4-20

**Note:**

Comments supports maximum 27 characters and it only contains 16 characters after downloading into PLC.

### 4.2.2 Statement

Edit statements for program, its prerequisite is a program.

#### Operating procedure

- 1) Click [Statement], if the background color of the [Statement] changed, then this function is enabled.

- 1) If the background does not change, please save the current project and restart the software as administrator;
- 2) When enable edit statement mode, and double click the certain project line, it will pop-up [Statement Edit] window;
- 3) Enter the statement for this line;
- 4) Click [Execute] button to save;

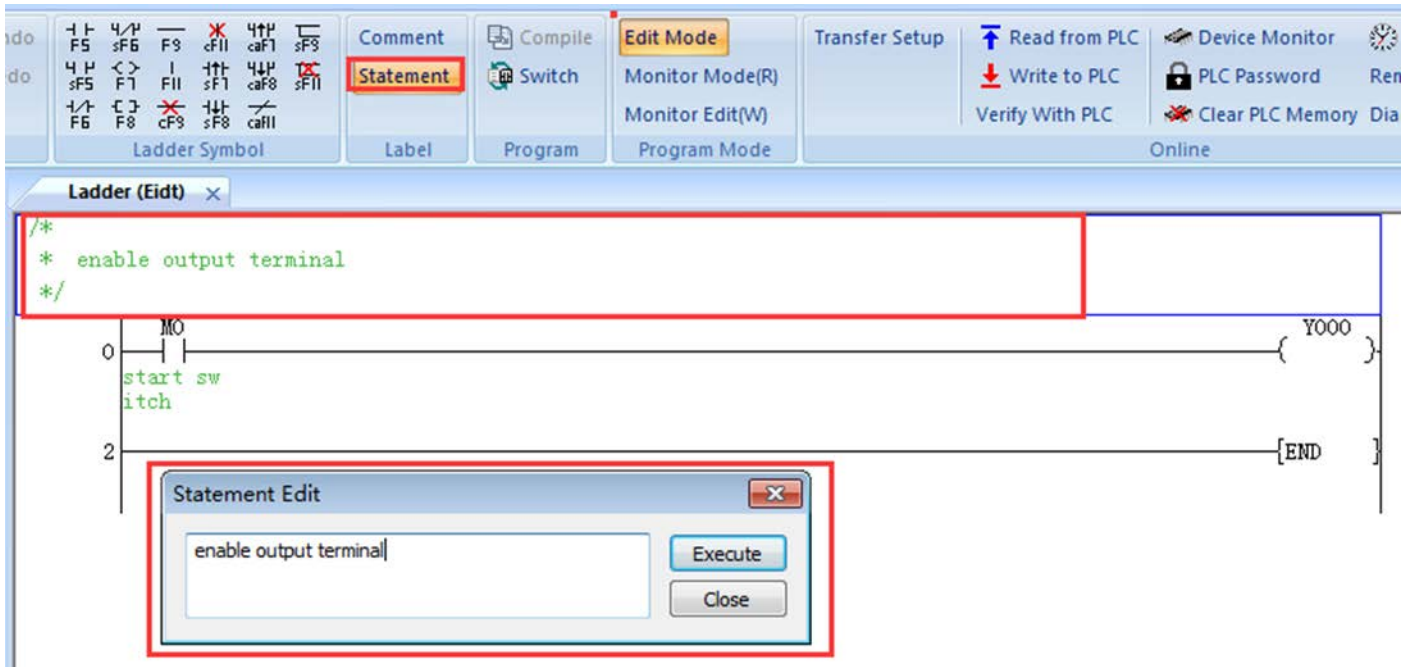


Figure 4-21

**Note:**

There is no limit of characters in statement, but the statement can't be downloaded into PLC.

### 4.3 Compile

This sector explains how to compile the program. The [Compile] is in [Program] toolbar. If the compile is successful, the [Compile] is unavailable.

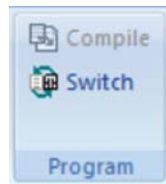


Figure 4-22

As the Figure 4-23 shows, if the program isn't compiled, the program background is in purple, the [Compile] is available. Users can click it for compiling program.



Figure 4-23

**Operating procedure**

- 1) Run the software, and then create a new program;
- 2) Click [Compile] in [Program] toolbar when complete program;
- 3) If there is no error, software will show the following tip message;

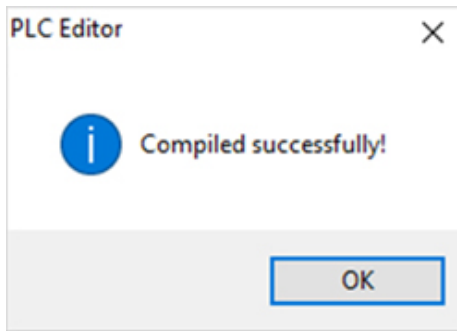


Figure 4-24

- 4) If there is any error, software will show the following tip message;

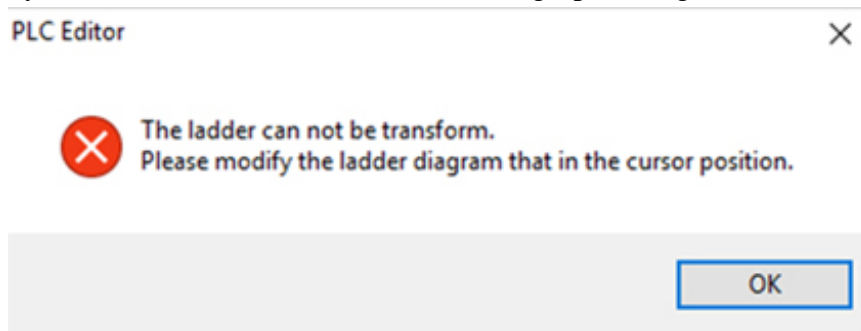


Figure 4-25

# 5 Transfer project

This chapter explains the operation methods for transfer project between PLC and PC.

## 5.1 Communication test

The main purpose of the communication test is to improve the stability of the communication between PLC and PC.

### Operating procedure

- 1) Open TETA PLC Editor software;
- 2) Click [Transfer Setup] in [Online] toolbar, as Figure 5-1 shows;

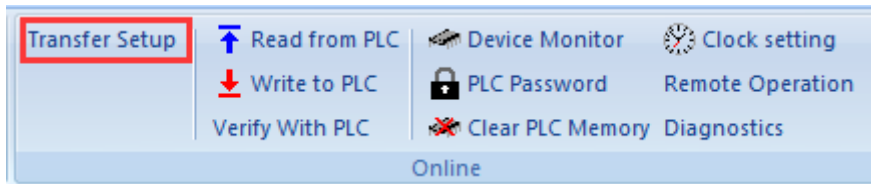


Figure 5-1

- 3) Open the [Transfer Setup] windows as Figure 5-2 shows, this [Transfer Setup] is available in V1.1.0 and later software version;

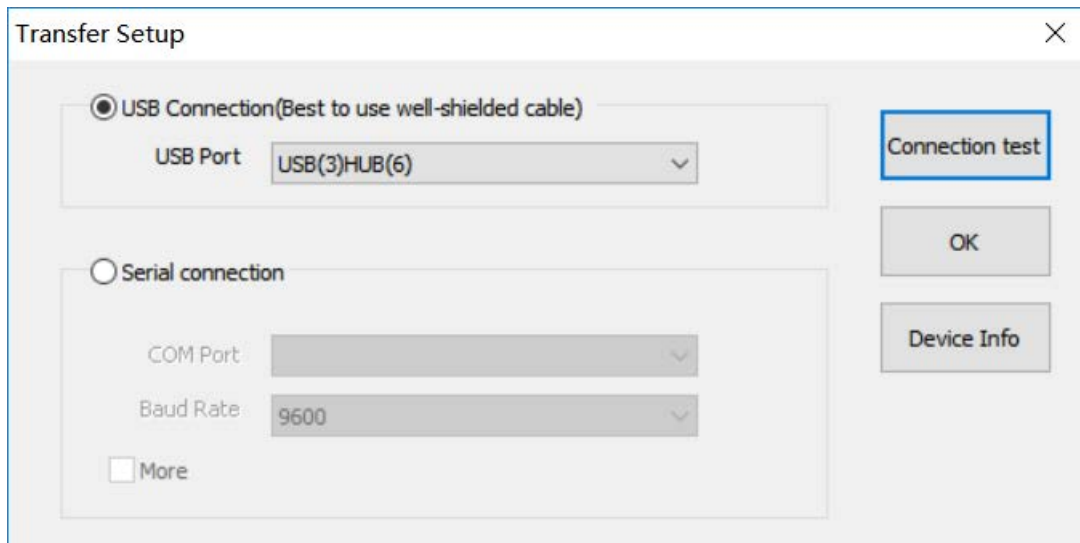


Figure 5-2

- USB Connection: It is for programming and monitor PLC via Micro USB port (PLC produced after February 2016);
- Serial Connection: it is for programming and monitor old PLC devices via Micro USB port produced before February 2016, and also for programming and monitor all PLC devices via RS422 port (round port);

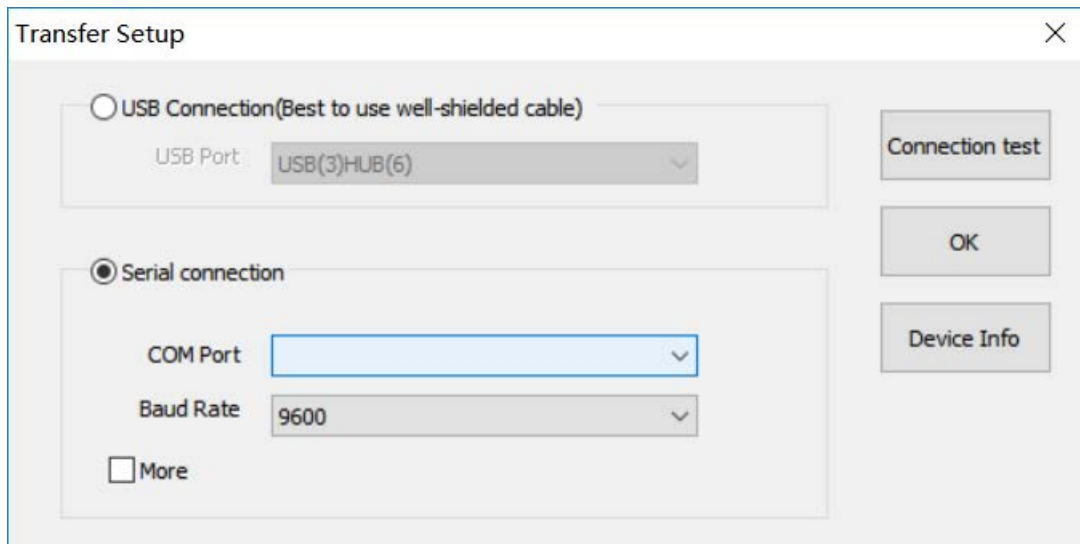


Figure 5-3

- Software selects connection mode automatically according to the PLC device, but if old PLC and new PLC were connected to PC at the same time, the software would ask user to select connection mode.
- 4) Check [More] for communication parameters setting, as following figure shows, user can set communication parameters according requirements;

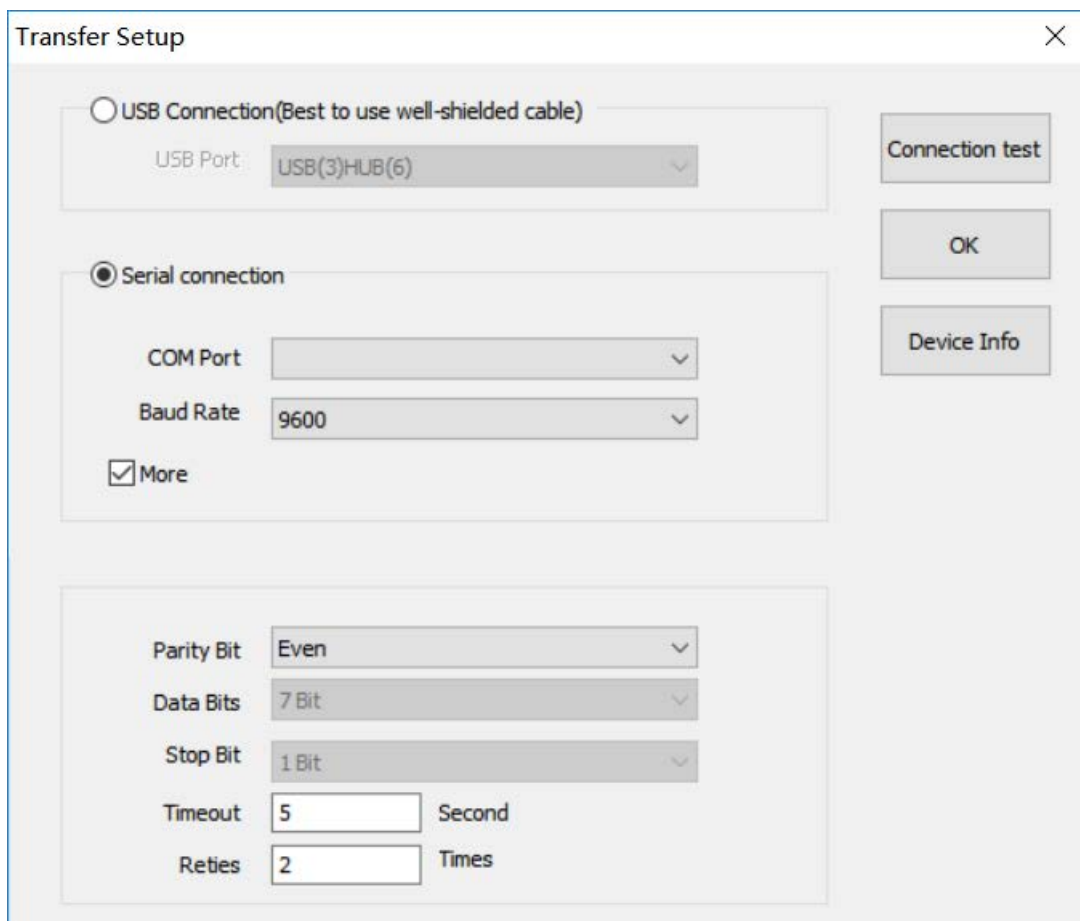


Figure 5-4

- Click [Connection test] for communication testing, the testing tip message as Figure 5-6 shows,

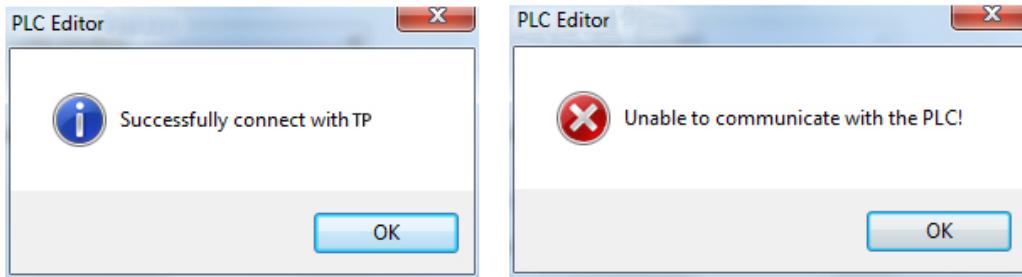


Figure 5-5

- If successfully connect with PLC, users can check PLC information by click [Device Info], as Figure 5-7 shows, the version also can be check by D8001 and D8101.

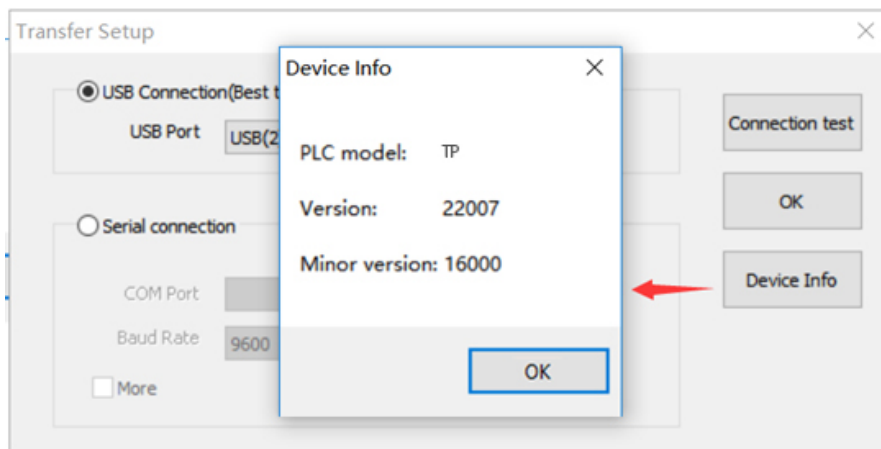


Figure 5-6

**Note:** Please use micro USB cable with strong anti-interference ability.

## 5.2 Download project

This section explains download project from PC to PLC.

### Operating procedure

- Click [Write to PLC] in [Online] toolbar;

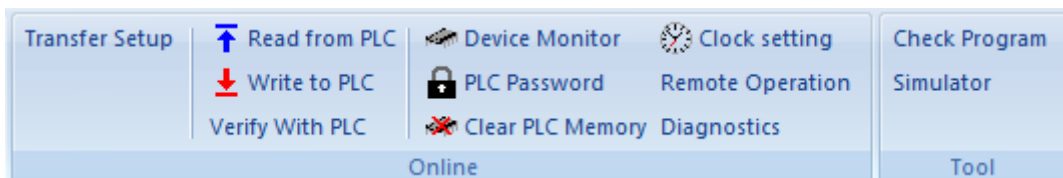


Figure 5-7

- The download window as Figure 5-8 shows:

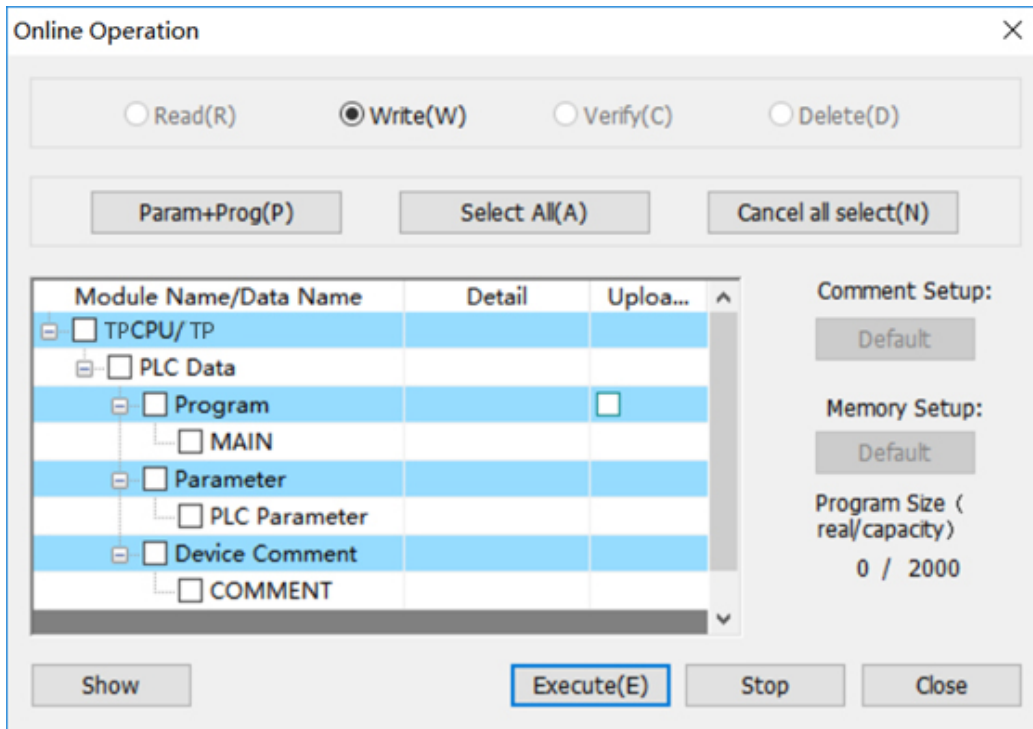


Figure 5-8

- 3) Select downloading content by click buttons, such as [Param + Prog (P)], as Figure 5-9 shows.

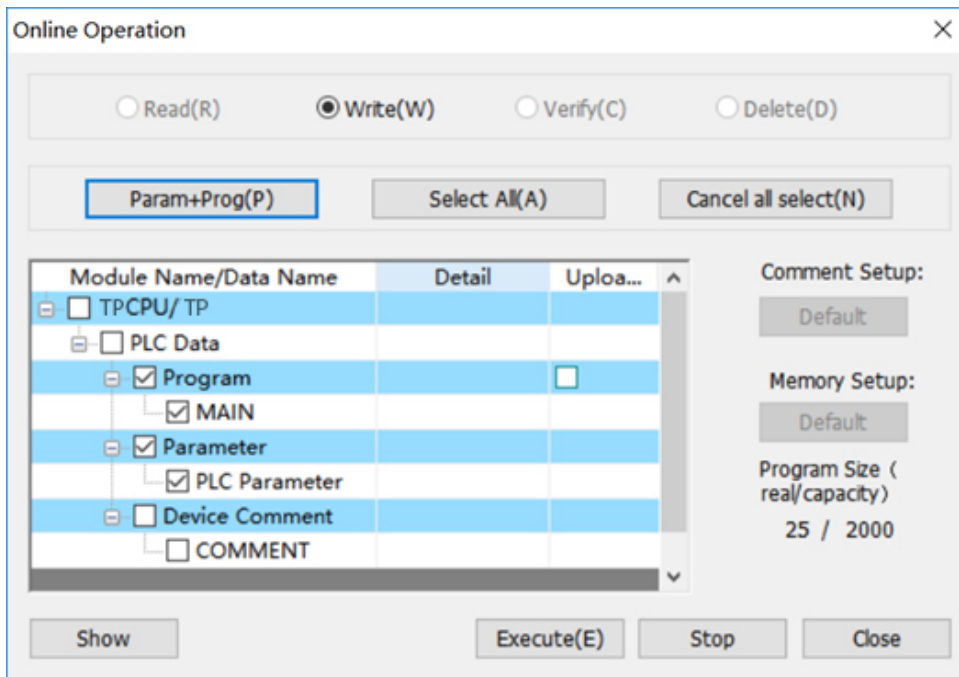


Figure 5-9

- If users need to download comments into PLC, please modify the PLC capacity in [PLC Parameters].
- 4) After clicking [Execute (E)], if PLC device still in running state, it would pop-up following tip message, users need to confirm safety and click [Yes] to continue downloading.



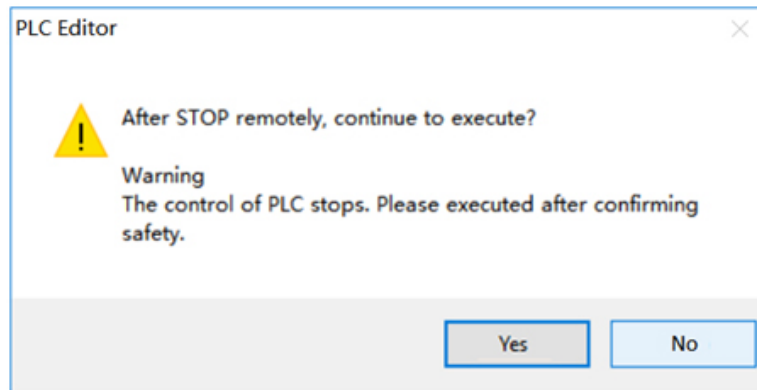


Figure 5-10

5) The information for downloading is displayed in bottom of screen, as following figure shows.

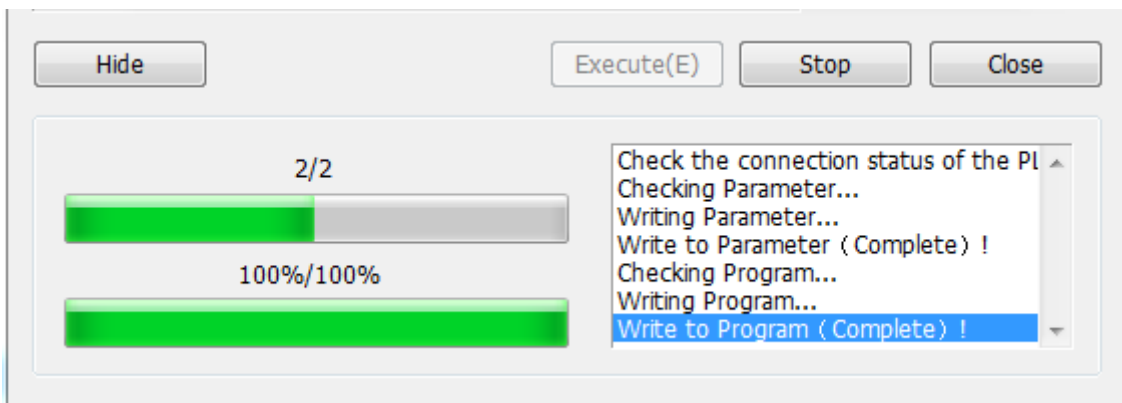


Figure 5-11

6) Complete message for downloading as following figure shows.

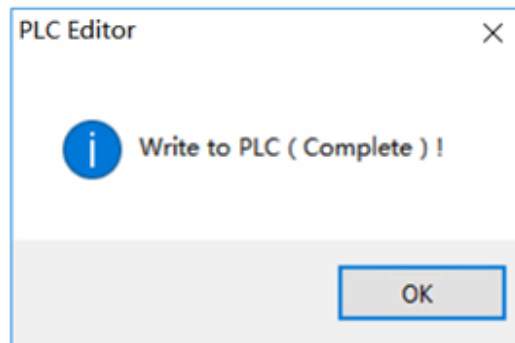


Figure 5-12

### 5.3 Upload project

This section explains upload project from PC to PLC.

[Read from PLC] is for uploading PLC project from PLC device to programming software. If there is no communication between PLC and software, this function can't work. The uploading window as Figure 5-13 shows.

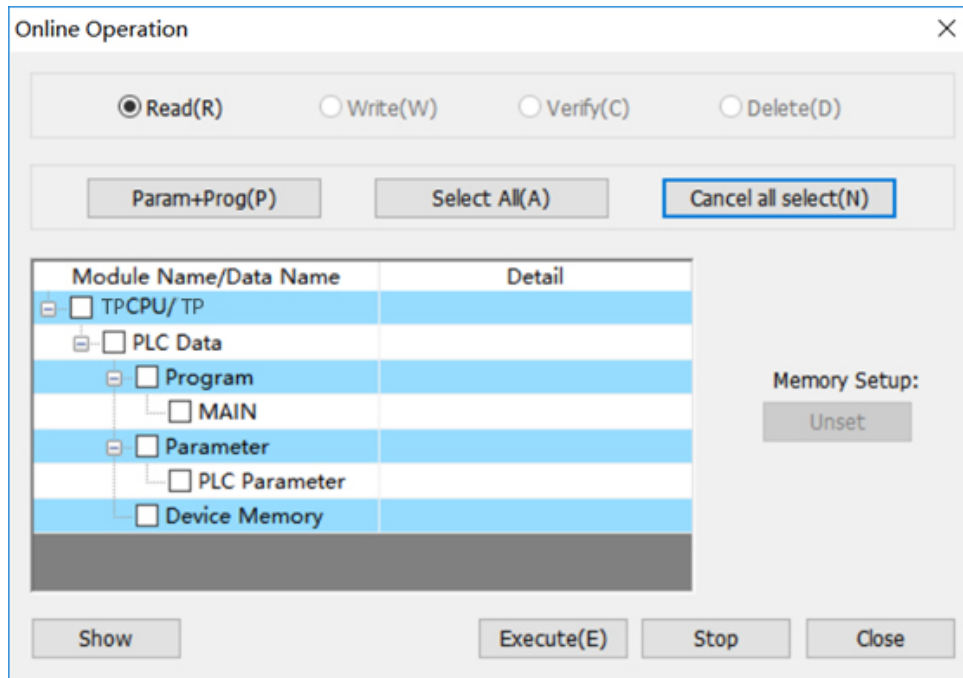


Figure 5-13

**Button description:**

- 1) [Param+Prog (P)]: Upload parameters and project from PLC;
- 2) [Select All (A)]: Upload parameters, PLC project, comments, and device memory from PLC;
- 3) [Cancel all select (N)]: Deselect all;
- 4) [Execute (E)]: Executeupload operation;
- 5) [Stop]: Terminate upload operation;
- 6) [Close]: Exit the current window;
- 7) [Show]: Shows upload progress and other information, as Figure 5-14 shows:

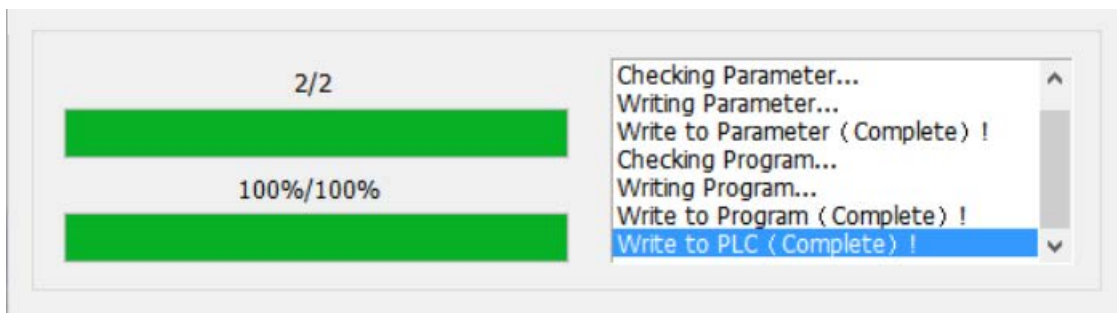


Figure 5-14

**Memory setup**

The setting for memory setup is enabled when users check [Device Memory] in [Online Operation] windows

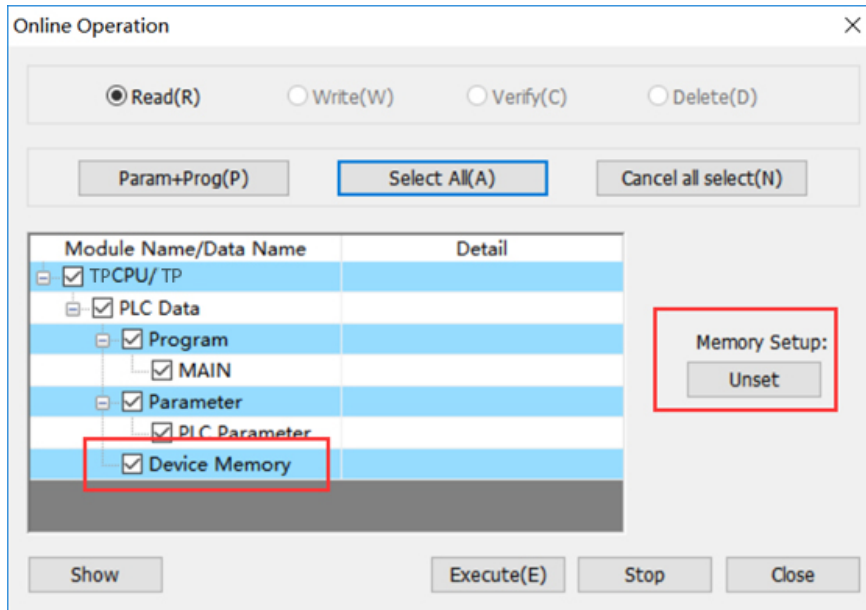


Figure 5-15

The default name of Device data is MAIN, users can rename it, also set Device data.

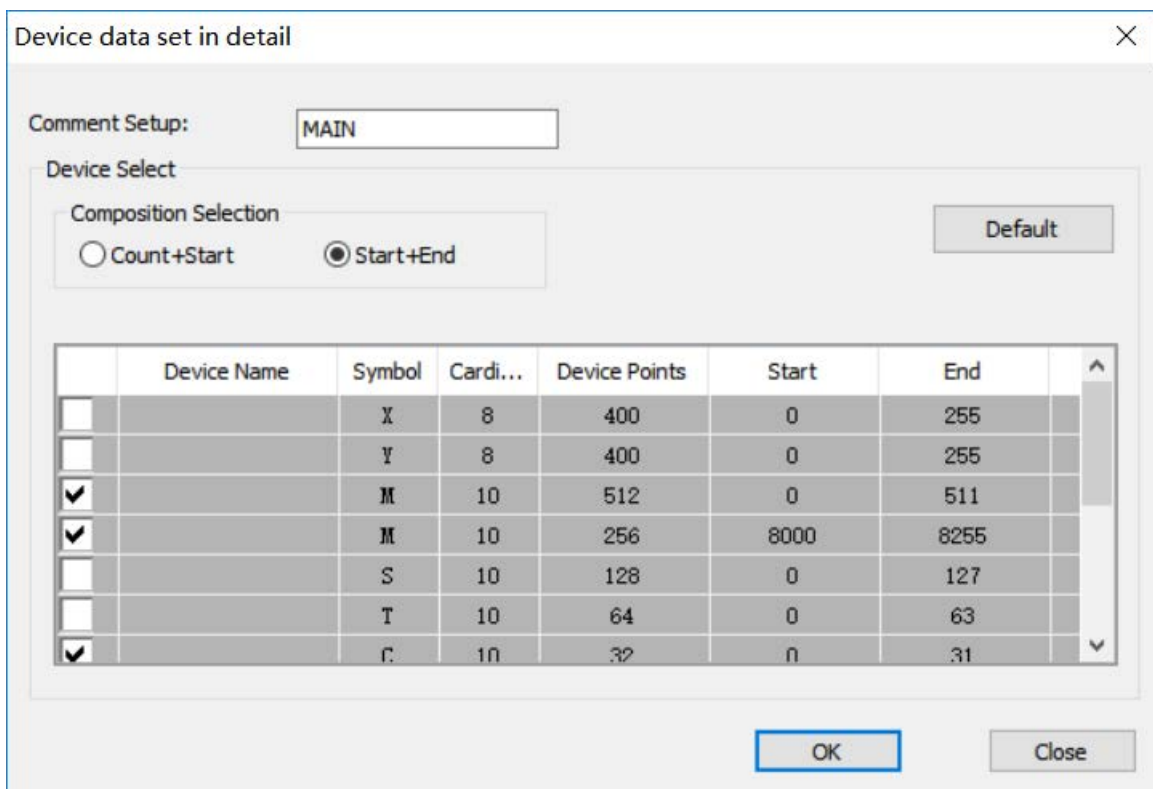


Figure 5-16

## 5.4 Upload prohibited

This section explains upload prohibited function and set upload prohibited.

Upload prohibited can ensure that the project once downloaded to the PLC, cannot be uploaded to PC.

### Operating procedure

- 1) Open downloading window;
- 2) Check [Upload prohibited] in window, as Figure 5-17 shows;

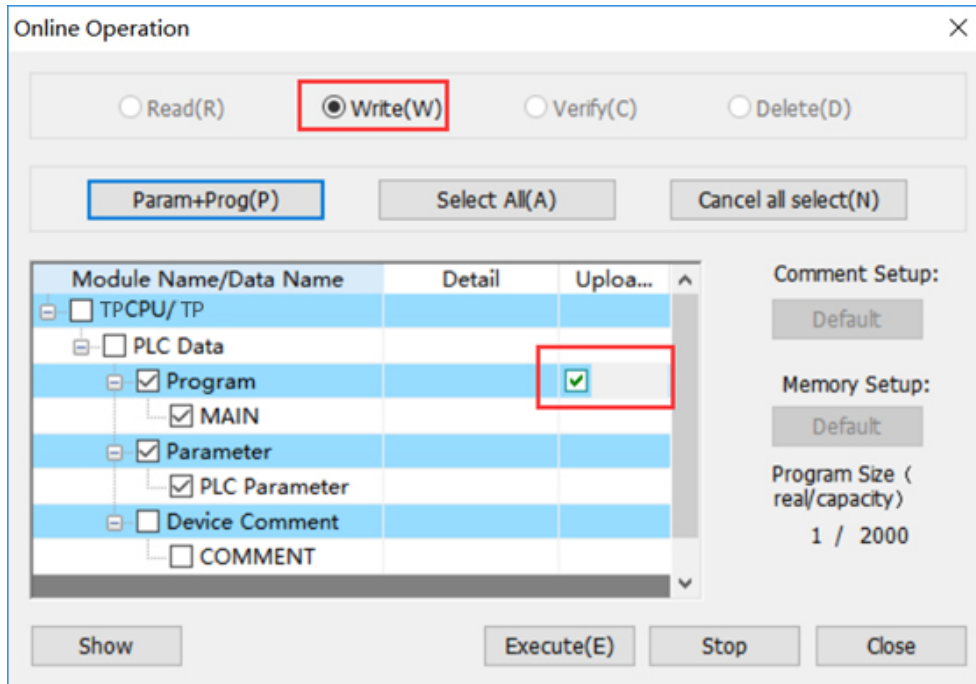


Figure 5-17

- 3) Click [Execute(E)] button;

When upload the project from PLC to PC, user will get following tip message.

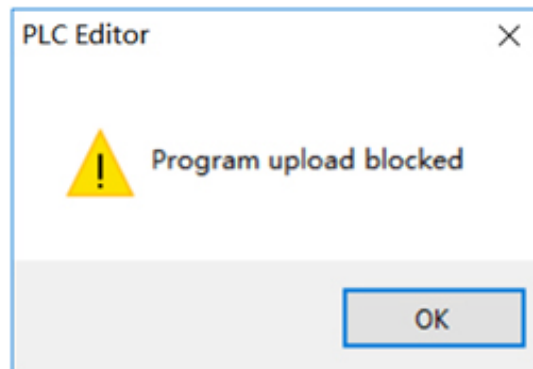


Figure 5-18

# 6 Debug program

This chapter explains how to debug sequence programs. TETA PLC Editor provides two methods to debug program.

## 6.1 Monitor mode

This section explains [Monitor mode] function, and how to debug sequence program in monitor mode. Monitor mode is used for changing the value of device, when PLC is running.

### Operating procedure

- 1) Completed program;
- 2) Connect PLC to PC;
- 3) Select [Monitor mode (R)];
- 4) Select device;
- 5) Right-click and select [Debug]-> [Modify the current value (M)]

Figure 6-1 is for bit device, Figure 6-2 is for word device.

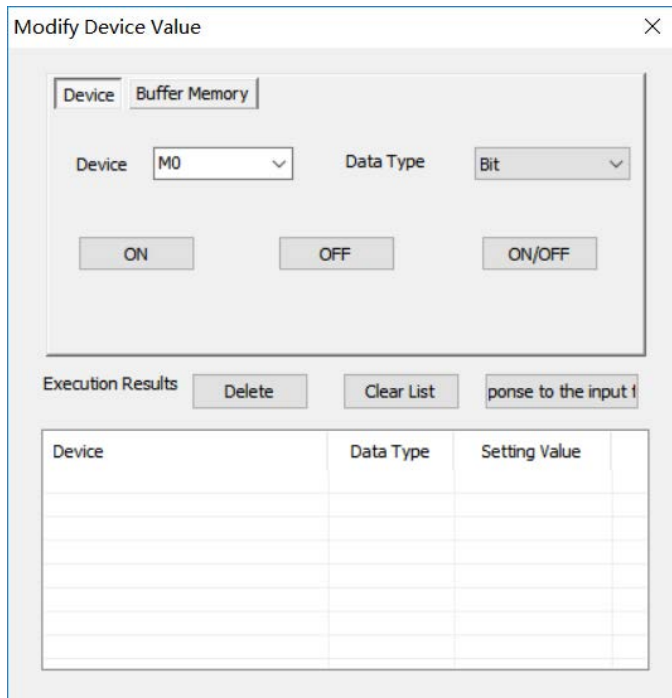


Figure 6-1

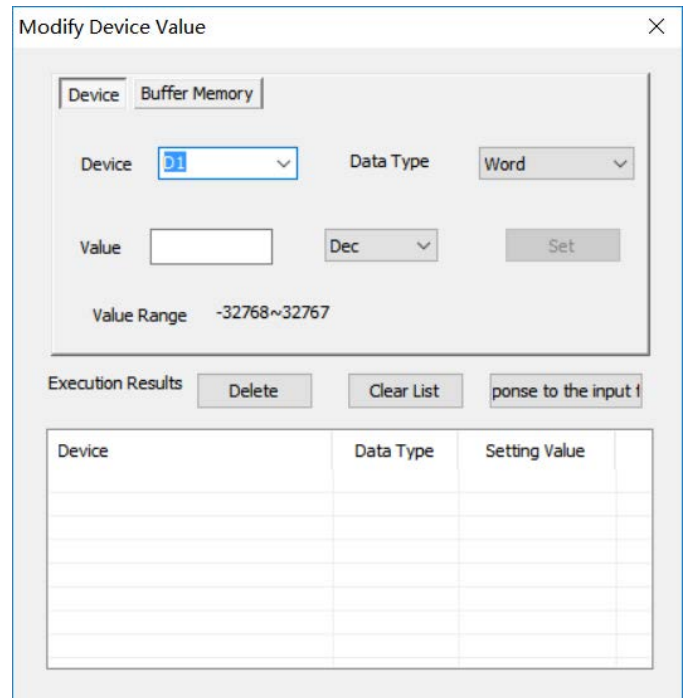


Figure 6-2

## 6.2 Monitor edit

This section explains [Monitor edit] function, and how to debug sequence program in monitor mode.

Monitor edit is not only for changing the value of device, but also changing the program, when PLC is running.

**Operating procedure of changing value**

- 1) Completed program;
- 2) Connect PLC to PC;
- 3) Select [Monitor Edit (W)];
- 4) Select device;
- 5) Right-click and select [Debug]-> [Modify the current value (M)]

Figure 6-1 is for bit device, Figure 6-2 is for word device.

**Operating procedure of changing value**

- 1) Completed program;
- 2) Connect PLC to PC;
- 3) Select [Monitor Edit (W)];
- 4) Editing the program;
- 5) Click [Compile] in [Program] toolbar, software will shows following tip message;
- 6) Click [Yes] button, when complete download, software pop-up window as Figure 6-4 shows;

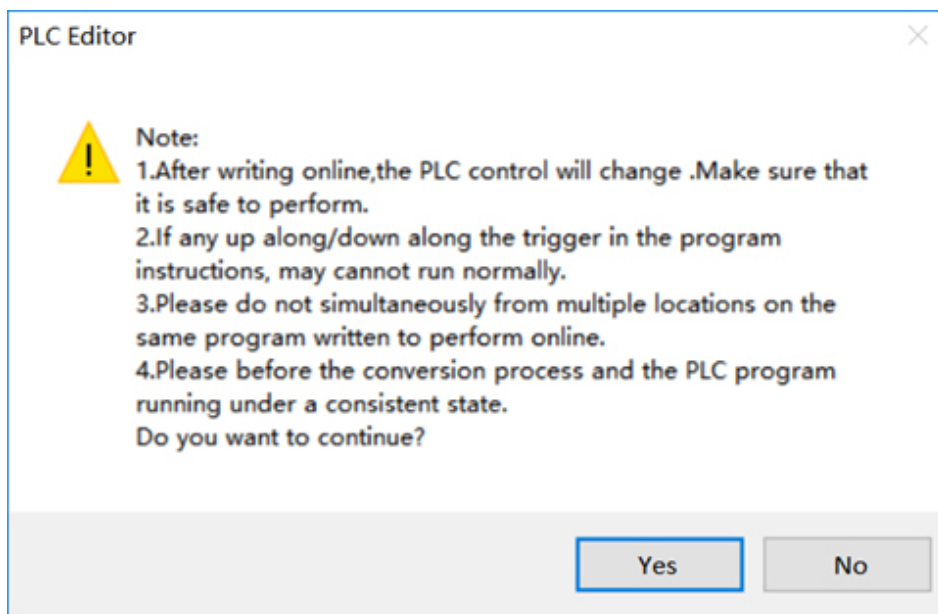


Figure 6-3

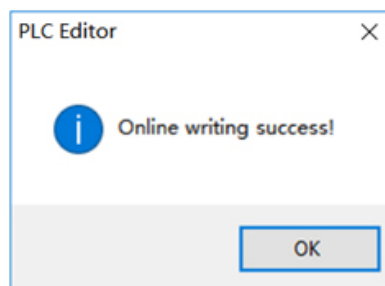


Figure 6-4

## 7 Shortcuts list

This chapter shows the list of shortcuts that can be used for TETA PLC Editor.

### 7.1 Common shortcuts list

The following table lists the common shortcuts.

Table 7-1

Shortcuts	Corresponding menu	Description
Ctrl + N	New	Create a new project
Ctrl + O	Open	Open an existing project
Ctrl + S	Save	Save the project
Ctrl + X	Cut	Cut the selected data
Ctrl + C	Copy	Copy the selected data
Ctrl + V	Paste	Paste the cut/copied data at the cursor position
Ctrl + Z	undo	Cancel the previous operation
Ctrl + Y	Redo	Perform the operation canceled by [Undo]
Ctrl + F	Find Device	Search for a device
Ctrl + F1	Show/Hide toolbar menu	Show/hide toolbar menu
F3	Start monitoring	Start monitoring the window being operated.
Ctrl + F3	Stop monitoring	Stop monitoring the window being operated
F4	Transform/transform + compilation	Compile (Transform) current program
Alt+F4	Exit	Close the project being edited and exits TETA PLC Editor

### 7.2 Shortcuts list in programming area

The following table lists the shortcuts in programming area.

Table 7-2

Shortcuts	Corresponding menu	Description
F5	Open contact	Insert an open contact at the cursor position
Shift + F5	Open branch	Insert an open contact branch at the cursor position
F6	Close contact	Insert a closed contact at the cursor position
Shift + F6	Close branch	Insert a closed contact branch at the cursor position

F7	Coil	Insert a coil at the cursor position
F8	Application instruction	Insert an application instruction at the cursor position
F9	Horizontal line	Insert a horizontal line at the cursor position
F11	Vertical line	Insert a vertical line at the cursor position
Ctrl + F9	Delete horizontal line	Delete the horizontal line at the cursor position
Ctrl + F11	Delete vertical line	Delete the vertical line at the cursor position
Shift + F7	Rising pulse	Insert a rising pulse at the cursor position
Shift + F8	Falling pulse	Insert a falling pulse at the cursor position
Ctrl + Alt + F7	Rising pulse branch	Insert a rising pulse branch at the cursor position
Ctrl + Alt + F8	Falling pulse branch	Insert a falling pulse branch at the cursor position
Ctrl + Alt + F11	Invert operation results	Insert an operation result inversion at the cursor position
Ctrl + Shift + Insert	Insert line statement	Insert statement line statement at the cursor position
Shift + Insert	Insert row	Insert a row at the cursor position
Shift + Delete	Delete row	Delete the row at the cursor position
Ctrl + Insert	Insert column	Insert a column at the cursor position
Ctrl + Delete	Delete column	Delete the column at the cursor position
Ctrl + →	Enter/Delete HLine rightward	Enter/delete a line at the right of the cursor position
Ctrl + ←	Enter/Delete HLine leftward	Enter/delete a line at the left of the cursor position
Ctrl + ↓	Enter/Delete VLine downward	Enter/delete a line at the downward of the cursor position
Ctrl + ↑	Enter/Delete VLine upward	Enter/delete a line at the upward of the cursor position
Ctrl + /	Switch open/close contact	Switch an open contact to closed contact, and vice versa
Ctrl + G	Jump	Display the specified row
Ctrl + F5	Comment	Display device comments
Ctrl + F7	Statement	Display statements
F1	Open the instructions help	Display the instructions help



# TETA TP Series PLC

## Programming manual



# 1 Programming manual overview

This manual gives details on all aspects of operation and programming for TP Series programmable controllers (PLCs). For all information relating to the PLC hardware and installation, refer to the appropriate manual supplied with the unit.

## 2 Devices

The following table lists all the devices that TETA TP series PLC supports.

Table 2-1

No.	Device	Descriptions	Page
1	X - Input	Representation of physical inputs to PLC;	<a href="#">69</a>
2	Y - Output	Representation of physical outputs from PLC;	<a href="#">70</a>
3	M - Intermediate	Common intermediate register; System special register;	<a href="#">70</a>
4	S - State	PLC internal states flag for step control;	<a href="#">71</a>
5	T - Timer	16-bit timer (1, 10 and 100ms)	<a href="#">72</a>
6	C - Counter	16-bit and 32-bit up/down counter; High speed counter;	<a href="#">72</a>
7	D – Data register	Data register ; String register; Indirect addressing address;	<a href="#">75</a>
8	P, I - Pointer	Jump pointer; Sub-program pointer; Interrupt pointer (high speed, );	<a href="#">76</a>
9	K, H - Constant	Binary, decimal, hexadecimal, floating point, etc.	<a href="#">76</a>

### 2.1 Input relay X

The input relay X represents the physical inputs to PLC. It can detect the external signal states. 0 is for open circuit, 1 is for closed circuit.

The states of input relays can't be modified by program instruction, the node signal (normally open, normally closed) can be unlimited use in the program.

If connected IO expansion module, the port starts from the main module, according to the order of the numbers. But DI is named in groups of eight. For example main module is X0~X7, X10~X14. The X0 in DI expansion module corresponds to X20, not X15.

Devices numbered in: Octal, i.e. X0 to X7, X10 to X17

**Available devices:**

Table 2-2

Type	Input	Output
TP-14MR/MT	8 point	6 point
TP-20MR/MT	12 point	8 point
TP-32MR/MT	16 point	16 point

TP-40MR/MT	24 point	16 point
TP-60MR/MT	36 point	24 point
TP -1212MR/MT	12 point	12 point
TP -1410MR/MT	14 point	10point
TP -1412MR/MT	14 point	12 point
TP -1616MR/MT	16 point	16 point
TP -2416MR/MT	24 point	16 point
TP -3624MR/MT	36 point	24 point

## 2.2 Output replay Y

The output relay Y represents physical outputs from PLC. 0 is for open circuit, 1 is for closed circuit. Depending on the output element can be divided into relay type, transistor type etc.

If connected IO expansion module, the port starts from the main module, according to the order of the numbers. But DO is named in groups of eight. For example main module is Y0~Y7, Y10~Y14. The Y0 in DO expansion module corresponds to Y20, not Y15.

Devices numbered in: Octal, i.e. Y0 to Y7, Y10 to Y17.

## 2.3 Auxiliary relays M

Auxiliary Relay M device is used as an intermediate variable during the execution of a program, as auxiliary relays in the practical power control system which is used to transfer the state messages. It can use the word variable formed by M variables. M variables is not directly linked with any external ports, but it can contact with the outside world by the manners of copying X to M or M to Y through the program coding. A variable M can be used repeatedly.

Devices **numbered in**: Decimal, i.e. M0 to M9, M10 to M19. The variables that are more than M8000 are the system-specific variables, which are used to interact with the PLC user program with the system states; part of the M variables have the feature of power-saving.

Table 2-3

PLC	General	Latched	Latched-specific	System-specific
TPS	384 ※3 (M0 – M383)	-	128 ※ (M383 – M511)	256 (M8000-M8255)
TPN	500 ※1 (M0 – M499)	524 ※ 2 (M500 – M1023)	2048 ※3 (M1024 – M3071)	256 (M8000-M8255)
TP	500 ※1 (M0 – M499)	524 ※ 2 (M500 – M1023)	2048 ※3 (M1024 – M3071)	256 (M8000-M8255)

※1, Non-latched area, it can be changed to latched area by parameter setting.

※2, Latched area, it can be changed to non-latched area by parameter setting.

※3, The non-latched or latched feature can't be changed.

A PLC has a number of special auxiliary relays. These relays all have specific functions and are classified into the following two types.

1) Using contacts of special auxiliary relays

Coils are driven automatically by the PLC. Only the contacts of these coils may be used by a user defined program

**M8000:** RUN monitor (ON during run)

**M8002:** Initial pulse (Turned ON momentarily when PLC starts)

**M8012:** 100 msec clock pulse

2) Driving coils of special auxiliary relays

A PLC executes a predetermined specific operation when these coils are driven by the user

**M8033:** All output statuses are retained when PLC operation is stopped

**M8034:** All outputs are disabled

**M8039:** The PLC operates under constant scan mode

**Note:**

M device present valid drive, and implement effective instruction END, users cannot use the special auxiliary relays that have not yet been defined.

## 2.4 State relays S

State relays S is used to design and handle step procedures, controls transfer of step by STL step instructions to simplify programming design. S also can be used as M, if there is no STL instruction. Part of the S has the feature of power-saving

Devices numbered in: Decimal, i.e. S0 to S9, S10 to S19.

Table 2-4

PLC	General			Latched			Alarm
	-	Initialized	-	-	Initialized	-	
TPS	-	-	-	128 ※3 (S0 – S127)	10 (S0 – S9)	10 (S10 –S19)	
TPN	500 ※1 (S0 – S499)	10 (S0 – S9)	10 (S10 – S19)	400 ※2 (S500 – S899)	-	-	100 ※2 (S900 – S999)
TP	500 ※1 (S0 – S499)	10 (S0 – S9)	10 (S10 – S19)	400 ※2 (S500 – S899)	-	-	100 ※2 (S900 – S999)

※1, Non-latched area, it can be changed to latched area by parameter setting.

※2, Latched area, it can be changed to non-latched area by parameter setting.

※3, The non-latched or latched feature can't be changed.

## 2.5 Timer

The timer is used to perform the timing function. Each timer contains coils, contacts, and counting time value register. A driven coil sets internal PLC contacts. Various timer resolutions are possible, from 1 to 100ms. If the coil power shuts off (insufficient power), the contacts will restore to their initial states and the value will automatically be cleared. Some timers have the feature of accumulation and power-saving.

Devices numbered in: Decimal, i.e. T0 to T9, T10 to T19.

Table 2-5

PLC	100ms 0.1 – 3276.7s	100ms 0.1 – 3276.7s 0.01 – 327.67s	10ms 0.01 – 327.67s	Latched 1ms 0.001 – 32.767s	Latched 100ms 0.1 – 3276.7s
TPS	32 (T0 – T31)	31 (T32 – T62)	31 (T32 – T62)	1 (T63)	
TPN	200 (T0 – T199)	-	46 (T200 – T245)	Interrupted 4 (T246 – T249)	6 (T250 – T255)
	Sub-program 8 (T192 – T199)				
TP	200 (T0 – T199)	-	46 (T200 – T245)	Interrupted 4 (T246 – T249)	6 (T250 – T255)
	Sub-program 8 (T192 – T199)				

## 2.6 Counter

Counter performs counting function, it contains coil, contact and count value register. The current value of the counter increases each time coil C0 is turned ON. The output contact is activated when count value reach to preset value.

Counters which are latched are able to retain their status information, even after the PLC has been powered down. This means on re-powering up, the latched counters can immediately resume from where they were at the time of the original PLC power down.

Devices numbered in: Decimal, i.e. C0 to C9, C10 to C19

Table 2-6

PLC	16bit UP Counters 0 – 32,767		32bit Bi-directional Counters -2,147,483,648 - +2,147483647	
	General	Latched	Latched-special	System-special
TPS	16 (C0 – C15) ※3	16 (C16 – C31) ※3	-	-

TPN	100 (C0-C99) ※1	100(C100 – C199) ※2	20 (C200 – C219) ※1	15 (C220 – C234) ※2
TP	100 (C0-C99) ※1	100(C100 – C199) ※2	20 (C200 – C219) ※1	15 (C220 – C234) ※2

※1, Non-latched area, it can be changed to latched area by parameter setting.

※2, Latched area, it can be changed to non-latched area by parameter setting.

※3, The non-latched or latched feature can't be changed.

## 2.7 High speed counter

Although counters C235 to C255 (21 points) are all high speed counters, they share the same range of high speed inputs. Therefore, if an input is already being used by a high speed counter, it cannot be used for any other high speed counters or for any other purpose, i.e. as an interrupt input.

The selection of high speed counters is not free, they are directly dependent on the type of counter required and which inputs are available.

### Available counter types;

- a) 1 phase with user start/reset: C235 to C240
- b) 1 phase with assigned start/reset: C241 to C245
- c) 2 phase bi-directional: C246 to C250
- d) A/B phase type: C251 to C255

Different types of counters can be used at the same time but their inputs must not coincide. Inputs X0 to X7 cannot be used for more than one counter.

Table 2-7

Input	1 phase 1 directional											2 phase bi-directional					A/B phase				
	C235	C236	C237	C238	C239	C240	C241	C242	C243	C244	C245	C246	C247	C248	C249	C250	C251	C252	C253	C244	C255
X0	U/D						U/D			U/D		U	U		U		A	A		A	
X1		U/D					R			R		D	D		D		B	B		B	
X2			U/D					U/D			U/D		R		R			R		R	
X3				U/D				R			R			U		U			A		A
X4					U/D				U/D					D		D			B		B
X5						U/D			R					R		R			R		R
X6										S					S					S	
X7											S					S					S

U: up counter input

D: down counter input

R: reset counter (input)

S: start counter (input)

A: A phase counter input

B: B phase counter input

### 2.7.1 Output Y: high speed pulse output transistor

- It supports up to 4 channels, and each channel maximum output frequency is 200K;
- The output frequency can be used for controlling inverter, stepper and servo motors and so on;

### 2.7.2 Input X: one phase

- If first two(X0, X1) terminals were for hardware counter, they can support maximum 200 KHz input signal at the same time; If first two terminals were for software counter could support maximum 100 KHz input signal at same time; they default for hardware counter, but users can change them for software counter by HSCS, HSCR, HSZ instructions;
- X2, X3, X4 and X5 are for software counter, they can support maximum 10 KHz input signal at the same time;

### 2.7.3 Input X: A/B phase

- If first two(X0, X1) terminals were for hardware counter, they can support maximum 100 KHz input signal at the same time; If first two terminals were for software counter could support maximum 50 KHz input signal at same time; they default for hardware counter, but users can change them for software counter by HSCS, HSCR, HSZ instructions;
- X2, X3, X4 and X5 are for software counter, they can support maximum 5 KHz input signal at the same time;

There are two frequency modes for 2 phase 2 input, one is 2 times, and the other is 4 times, as following table shows, users select mode in D8200.

Table 2-8

Value in D8200	Count icon
K2 (two times)	
K4 or others (four times) (default)	

**Note:**HSCS, HSCR and HSCZ can't be used with Frequency multiplication.



## 2.8 Data register D

Data registers, as the name suggests, store data. The stored data can be interpreted as a numerical value or as a series of bits, being either ON or OFF. A single data register contains 16bits or one word. However, two consecutive data registers can be used to form a 32bit device more commonly known as a double word. If the contents of the data register are being considered numerically then the Most Significant Bit (MSB) is used to indicate if the data has a positive or negative bias. As bit devices can only be ON or OFF, 1 or 0 the MSB convention used is, 0 is equal to a positive number and 1 is equal to a negative number.

In TETA TP Series PLC, most data in the instructions are signed numbers. The bit 15 in 16-bit address is sign bit (0 means positive, 1 means negative). The high bit 15 in 32-bit address is sign bit, the data range is -32,768 - +32,767.

Devices numbered in: Decimal, i.e. D0 to D9, D10 to D19

Table 2-9

PLC	General	Latched	Latched- specific		System- specific	Special
			-	Files		
TPS	128 ※3 (D0-D127)	-	128 ※3 (D128-D255)	D1000-D2499 can be used for files by parameter setting	256 (D8000-D8255)	16 (V0-V7) (Z0-Z7)
TPN	200 ※1 (D0-D199)	312 ※2 (D200-D511)	7488 ※3 (D512-D7999)	D1000-D7999 can be used for files by parameter setting	256 (D8000-D8255)	16 ※3 (V0-V7) (Z0-Z7)
TP	200 ※1 (D0-D199)	312 ※2 (D200-D511)	7488 ※3 (D512-D7999)	D1000-D7999 can be used for files by parameter setting	256 (D8000-D8255)	16 ※3 (V0-V7) (Z0-Z7)

※1, Non-latched area, it can be changed to latched area by parameter setting.

※2, Latched area, it can be changed to non-latched area by parameter setting.

※3, The non-latched or latched feature cannot be changed.

### 2.8.1 Index registers V, Z

The index registers are same as common data registers, is 16-bit registers for data reading and writing. There are totally 64 (14) registers, V0-V31 (7), Z0-Z31 (7).

The index registers can be used in combination with other registers or values by application instructions. But they cannot be used in combination with the basic instructions and step ladder diagram instruction.

### 2.8.2 File registers D

The file registers start from D1000 to D7999. File registers can be secured in the program memory in units of 500 points. File registers are actually setup in the parameter area of the PLC. For every block of 500 file registers allocated and equivalent block of 500 program steps are lost.

### 2.9 Pointers registers P, I

Pointers register P is used for entry address of jump program, and identification of sub-program starting address.

Pointer register I is used for identification of interrupted program starting address.

Devices numbered in: Decimal, i.e. P0 to P9, P10 to P19, I0 to I9, I10 to I19.

Table 2-10

PLC	Sub-program		Insert	Insert counter	Counter interrupt
	-	Jump to end			
TPS	63 (P0-P62)	1 (P63)	6 I00_(X000), I10_(X001), I20_(X002), I30_(X003), I40_(X004), I50_(X005)	-	-
TPN	127 (P0-P62) (P64-P127)	1 (P63)	6 I00_(X000), I10_(X001), I20_(X002), I30_(X003), I40_(X004), I50_(X005)	3 (I6_, I7_, I8_)	6 (I010, I020, I030, I040, I050, I060)
TP	127 (P0-P62) (P64-P127)	1 (P63)	6 I00_(X000), I10_(X001), I20_(X002), I30_(X003), I40_(X004), I50_(X005)	3 (I6_, I7_, I8_)	6 (I010, I020, I030, I040, I050, I060)

**Note:** The input X for interrupt register can't be used for [high speed counter] and [SPD] instruction as the same time.

## 2.10 Constant K, H

TP Series PLC could support five kinds of contacts for programming, the detailed as the following table shows.

Table 2-11

Format	Description
Decimal	The set value of timer and counter (K is a constant); The number of Auxiliary Relay(M), Timer(T), Counter(C), Status(S) and so on (the number of registers); The value and instruction action in the operand, which are applied (K is a constant);
Hexadecimal	As with the decimal, it is applied in the operand and the specific actions in the application instruction.
Binary	Using decimal number or hexadecimal number to design the value of the timer, counter or data register. However, in the internal PLC, these data is dealt with binary numbers. Moreover, when monitoring external devices these registers will be converted to a decimal number automatically (16 hex can be converted as well).
Octal	It is used for distribute the register number of input relay and output relay. Use the binary values of [0-7, 10-17 ... 70-77, 100-107]. [8, 9] do not exist in the octal.
BCD	Binary-coded decimal (BCD) is a class of binary encodings of decimal numbers where each decimal digit is represented by a fixed number of bits, usually four or eight. Special bit patterns are sometimes used for seven segment display controlling.
BIN float	BIN float is used for calculation in PLC internal.
Decimal float	It is only used for monitoring and improving readability.

### 2.10.1 Constant K

[K] is decimal integer symbol, mainly used for setting the value of the timer or counter or application instruction operand values. The value range in 16-bit is -32,768 – 32,767, the value range in 32-bit is -2,147,483,648 – 2,147,483,647.

### 2.10.2 Constant H

[H] is hexadecimal numbers symbol, mainly used to setting the value of application instruction operand value. The value range in 16-bit instruction is 0000-FFFF, the value range in 32-bit instruction is 0000,0000 – FFFF, FFFF.

### 3 Basic program instructions

A program is a connected series of instructions written in a language that the PLC can understand. There are two forms of program format; instruction and ladder.

#### 3.1 Basic program instruction list

Table 3-1

<b>Instruction</b>	<b>Description</b>
LD	Operation contact type NO (normally open)
LDI	Operation contact type NC (normally closed)
OUT	Final logical operation type coil drive
AND	Serial connection of NO(normally open)
ANI	Serial logical, operation contact type NC(normally closed) contacts
OR	Parallel, connection of NO (normally open) contacts
ORI	Parallel, connection of NC (normally closed) contacts
LDP	Initial logical, operation -Rising edge pulse
LDF	Initial logical, operation falling/trailing edge pulse
ANDP	Serial connection of Rising edge pulse
ANDF	Serial connection of falling/ trailing edge pulse
ORP	Parallel, connection of NO Rising edge pulse
ORF	Parallel connection of Falling/trailing edge pulse
ORB	Serial connection of multiple parallel circuits
ANB	Serial connection of multiple parallel circuits
MPS	Stores the current result of the internal PLC operations
MRD	Reads the current result of the internal PLC operations
MPP	Pops (recalls and removes) the currently stored result
MC	Denotes the start of a master control block
MCR	Denotes the end of a master control block
INV	Invert the current result of the internal PLC operations
PLS	Rising edge pulse
PLF	Falling / trailing edge pulse
SET	Sets a bit device permanently ON
RST	Resets a bit device permanently OFF

### 3.2 Basic program instruction description

#### 3.2.1 LD, LDI (Load, Load Inverse)

##### Instruction Description

Table 3-2

Name	Function	Devices	Format	Steps
LD (Load)	Initial logical Operation contact type NO (normally open)	X,Y,M,S,T,C		1
LDI (Load Inverse)	Initial logical Operation contact type NC (normally closed)	X,Y,M,S,T,C		1

##### Program example



##### Basic points to remember

- 1) Connect the LD and LDI instructions directly to the left hand bus bar.

#### 3.2.2 OUT (out)

##### Instruction Description

Table 3-3

Name	Function	Devices	Format	Steps
OUT (OUT)	Final logical operation type coil drive	X,Y,M,S,T,C		Y,M:1 Special M coils:2 C(16 bit):3 C(32 bit):5



##### Basic points to remember

- 1) Connect the OUT instruction directly to the right hand bus bar.
- 2) It is not possible to use the OUT instruction to drive 'X' type input devices.

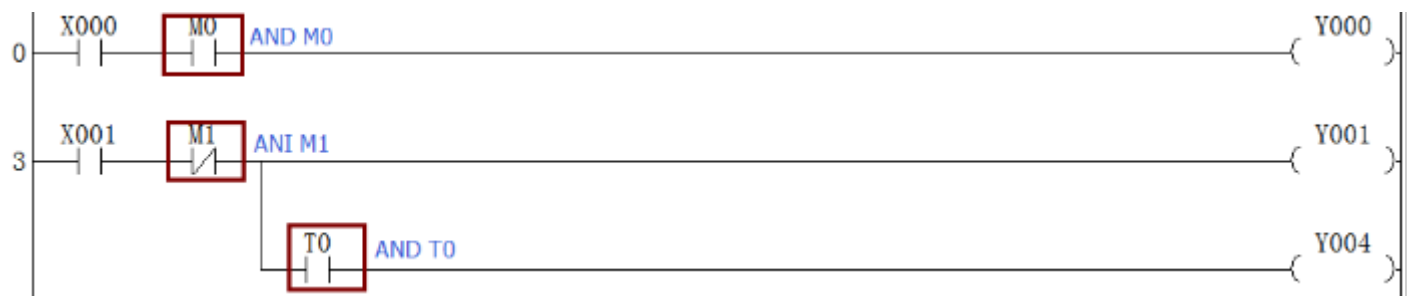
### 3.2.3 AND, ANI (And, And Inverse)

#### Instruction Description

Table 3-4

Name	Function	Devices	Format	Steps
AND (AND)	Serial connection of NO(normally open)	X,Y,M,S,T,C		1
ANI (AND Inverse)	Serial logical Operation contact type NC(normally closed) contacts	X,Y,M,S,T,C		1

#### Program example



#### Basic points to remember

- 1) Use the AND and ANI instructions for serial connection of contacts. As many contacts as required can be connected in series (see following point headed “Peripheral limitations”).
- 2) The output processing to a coil, through a contact, after writing the initial OUT instruction is called a “follow-on” output (for an example see the program above; OUT Y4). Follow on outputs are permitted repeatedly as long as the output order is correct.

### 3.2.4 OR, ORI (Or, Or Inverse)

#### Peripheral limitations

The PLC has no limit to the number of contacts connected in series or in parallel. However, some programming panels, screens and printers will not be able to display or print the program if it exceeds the limit of the hardware. It is preferable for each line or rung of ladder program to contain up to a maximum of 10 contacts and 1 coil. Also keep number of follow-on outputs to a maximum of 24.

**Instruction Description**

Table 3-5

Name	Function	Devices	Format	Steps
OR (OR)	Parallel Connection of NO (normally open) contacts	X,Y,M,S,T,C		1
ORI (OR Inverse)	Parallel Connection of NC (normally closed) contacts	X,Y,M,S,T,C		1

**Program example**



**Basic points to remember**

- 1) Use the OR and ORI instructions for parallel connection of contacts. To connect a block that contains more than one contact connected in series to another circuit block in parallel, use an ORB instruction.
- 2) Connect one side of the OR/ORI instruction to the left hand bus bar.

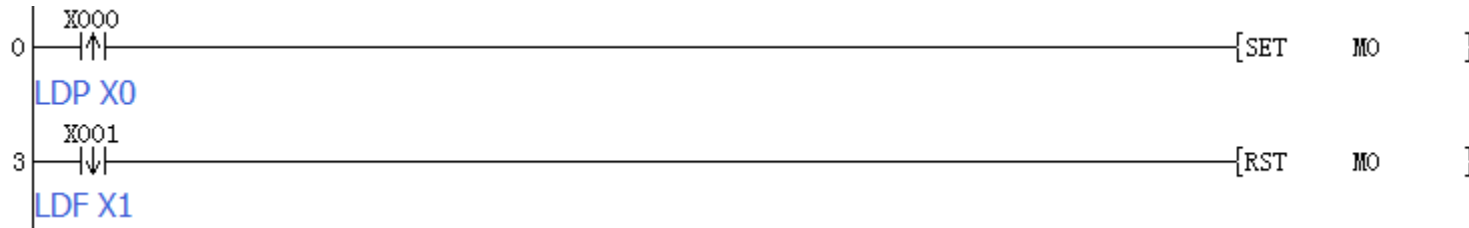
**3.2.5 LDP, LDF (Load Pulse, Load Trailing Pulse)**

**Instruction Description**

Table 3-6

Name	Function	Devices	Format	Steps
LDP (Load Pulse)	Initial logical Operation -Rising edge pulse	X,Y,M,S,T,C		2
LDF (Load Falling pulse)	Initial logical Operation falling /trailing edge pulse	X,Y,M,S,T,C		2

**Program example**



**Basic points to remember**

- 1) Connect the LDP and LDF instructions directly to the left hand bus bar.
- 2) LDP is active for one program scan after the associated device switches from OFF to ON.
- 3) LDF is active for one program scan after the associated device switches from ON to OFF.

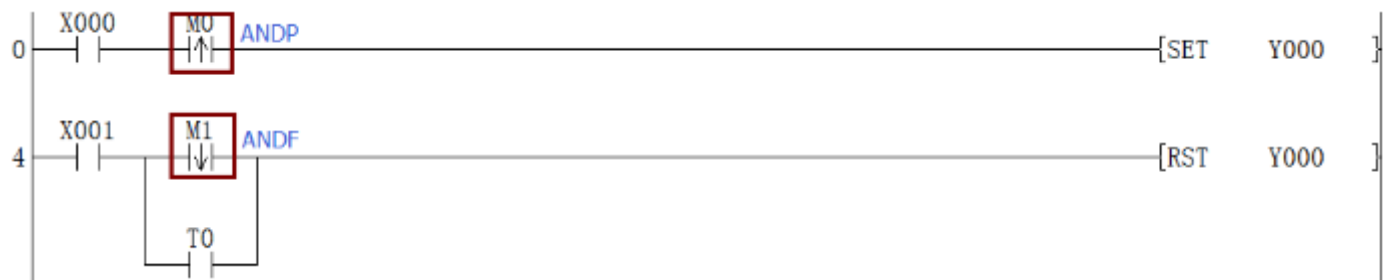
**3.2.6 ANDP, ANDF (And Pulse, And Trailing Pulse)**

**Instruction Description**

Table 3-7

Name	Function	Devices	Format	Steps
ANDP (AND Pulse)	Serial connection of Rising edge pulse	X,Y,M,S,T,C		2
ANDF (AND Falling pulse)	Serial connection of falling/ trailing edge pulse	X,Y,M,S,T,C		2

**Program example**



**Basic points to remember**

- 1) Use the ANDP and ANDF instructions for the serial connection of pulse contacts.
- 2) Usage is the same as for AND and ANI; see earlier.
- 3) ANP is active for one program scan after the associated device switches from OFF to ON.
- 4) ANF is active for one program scan after the associated device switches from ON to OFF.



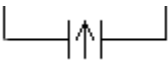
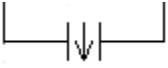
### 3.2.7 ORP, ORF (Or Pulse, Or Trailing Pulse)

#### Peripheral limitations

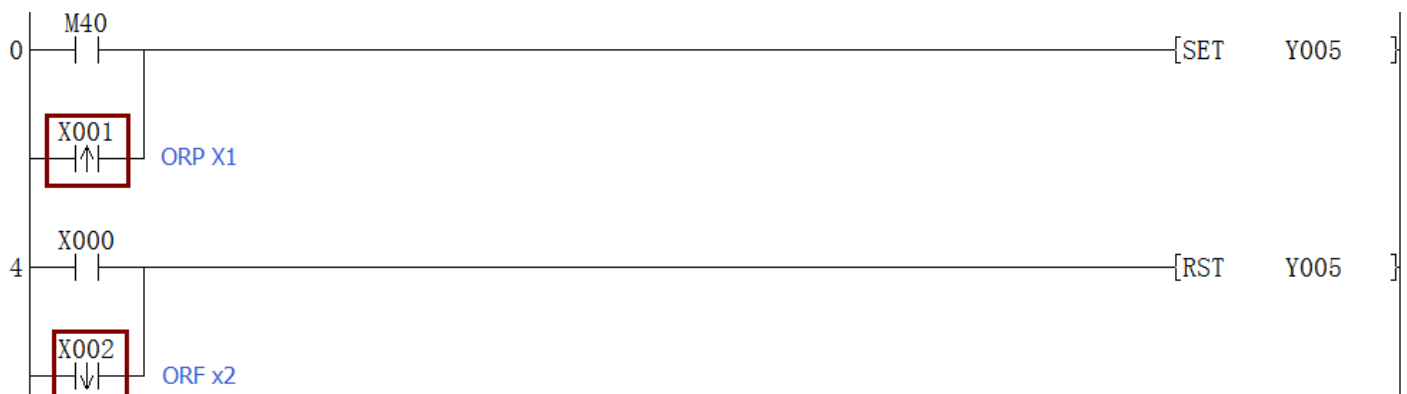
The PLC has no limit to the number of contacts connected in series or in parallel. However, some programming panels, screens and printers will not be able to display or print the program if it exceeds the limit of the hardware. It is preferable for each line or rung of ladder program to contain up to a maximum of 10 contacts and 1 coil. Also keep number of follow-on outputs to a maximum of 24.

#### Instruction Description

Table 3-8

Name	Function	Devices	Format	Steps
ORP (OR Pulse)	parallel Connection of NO Rising edge pulse	X, Y, M, S, T, C		2
ORF (OR Falling pulse)	Parallel connection of Falling/trailing edge pulse	X, Y, M, S, T, C		2

#### Program example



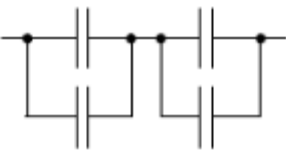
#### Basic points to remember

- 1) Use the ORP and ORF instructions for the parallel connection of pulse contacts.
- 2) Usage is the same as for OR and ORI; see earlier.
- 3) ORP is active for one program scan after the associated device switches from OFF to ON.
- 4) ORF is active for one program scan after the associated device switches from ON to OFF.

### 3.2.8 ANB, ORB (And Block)

#### Instruction Description

Table 3-9

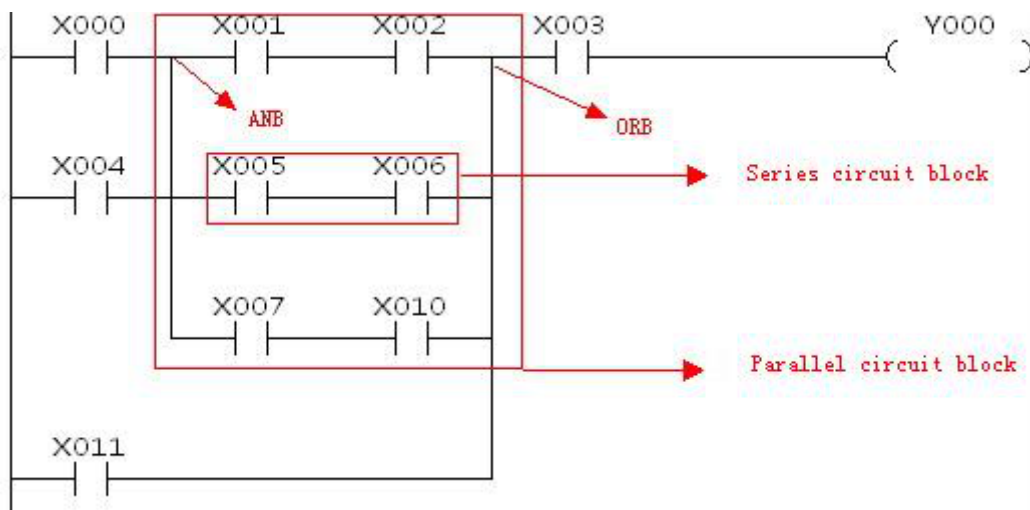
Name	Function	Devices	Format	Steps
ANB (And Block)	Serial connection of multiple parallel circuits	N/A		1

#### Sequential processing limitations

It is possible to use as many ANB instructions as necessary to connect a number of parallel circuit blocks to the preceding block in series (see the program listing).

#### Program example

Ladder mode:



Instruction List mode:

0	LD	X000
1	OR	X004
2	LD	X001
3	AND	X002
4	LD	X005
5	AND	X006
6	ORB	
7	LD	X007
8	AND	X010
9	ORB	
10	ANB	
11	OR	X011
12	AND	X003
13	OUT	Y000

ORB instruction is used in the end of each branch, not in the end of all branches, as command table above shown.

ORB and ANB instructions merely connect on the block. If the block not be used. As shown, examples for series circuits block and parallel circuits block.

**Batch processing limitations**

When using ANB instructions in a batch, use no more than 8 LD and LDI instructions in the definition of the program blocks (to be connected in parallel). Ignoring this will result in a program error (see ORB explanation for example).

**Basic points to remember**

- 1) An ANB instruction is an independent instruction and is not associated with any device number
- 2) Use the ANB instruction to connect multi-contact circuits (usually parallel circuit blocks) to the preceding circuit in series. Parallel circuit blocks are those in which more than one contact connects in parallel or the ORB instruction is used.
- 3) To declare the starting point of the circuit block, use a LD or LDI instruction. After completing the parallel circuit block, connect it to the preceding block in series using the ANB instruction.

**3.2.9 MPS, MRD and MPP**

**Instruction Description**

Table 3-10

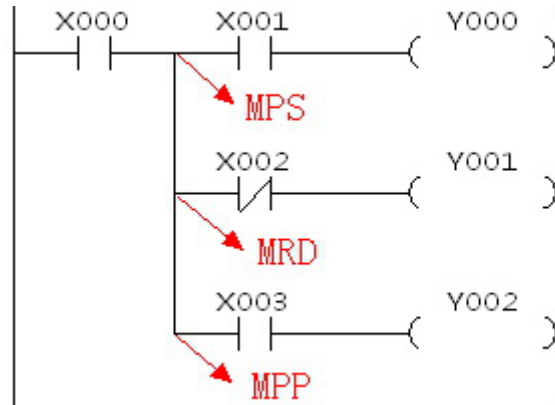
Name	Function	Devices	Format	Steps
MPS(Point Store)	Stores the current result of the internal PLC operations	N/A		1
MRD (Read)	Reads the current result of the internal PLC operations	N/A		1
MPP (Pop)	Pops (recalls and removes) the currently stored result	N/A		1

**MPS, MRD and MPP usage**

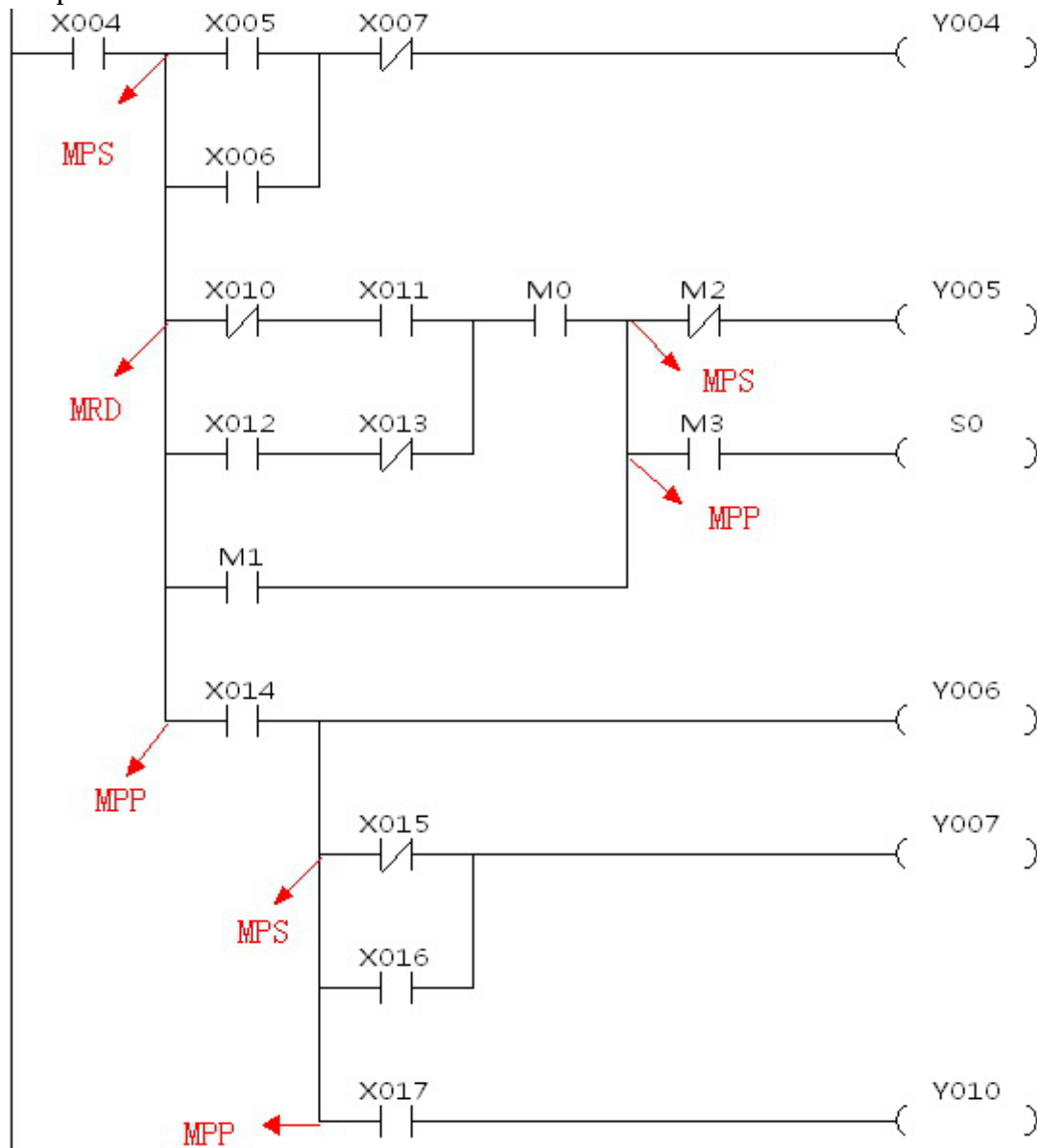
- 1) When writing a program in ladder format, programming tools automatically add all MPS, MRD and MPP instructions at the program conversion stage. If the generated instruction program is viewed, the MPS, MRD and MPP instructions are present.
- 2) When writing a program in instruction format, it is entirely

Program examples

1) Example 1



2) Example 2




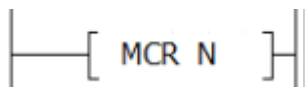
**Basic points to remember**

- 1) Use these instructions to connect output coils to the left hand side of a contact. Without these instructions connections, it can only be made to the right hand side of the last contact.
- 2) MPS stores the connection point of the ladder circuit so that further coil branches can recall the value later.
- 3) MRD recalls or reads the previously stored connection point data and forces the next contact to connect to it.
- 4) MPP pops (recalls and removes) the stored connection point. First, it connects the next contact, and then it removes the point from the temporary storage area.
- 5) For every MPS instruction there MUST be a corresponding MPP instruction.
- 6) The last contact or coil circuit must connect to an MPP instruction.
- 7) At any programming step, the number of active MPS-MPP pairs must be no greater than 11.

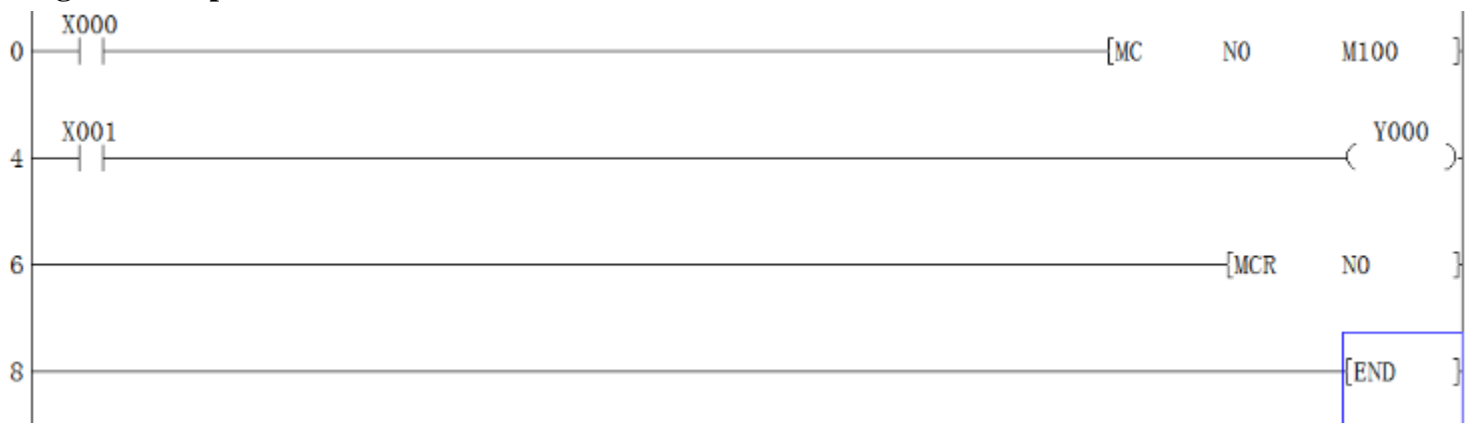
**3.2.10 MC, MCR**

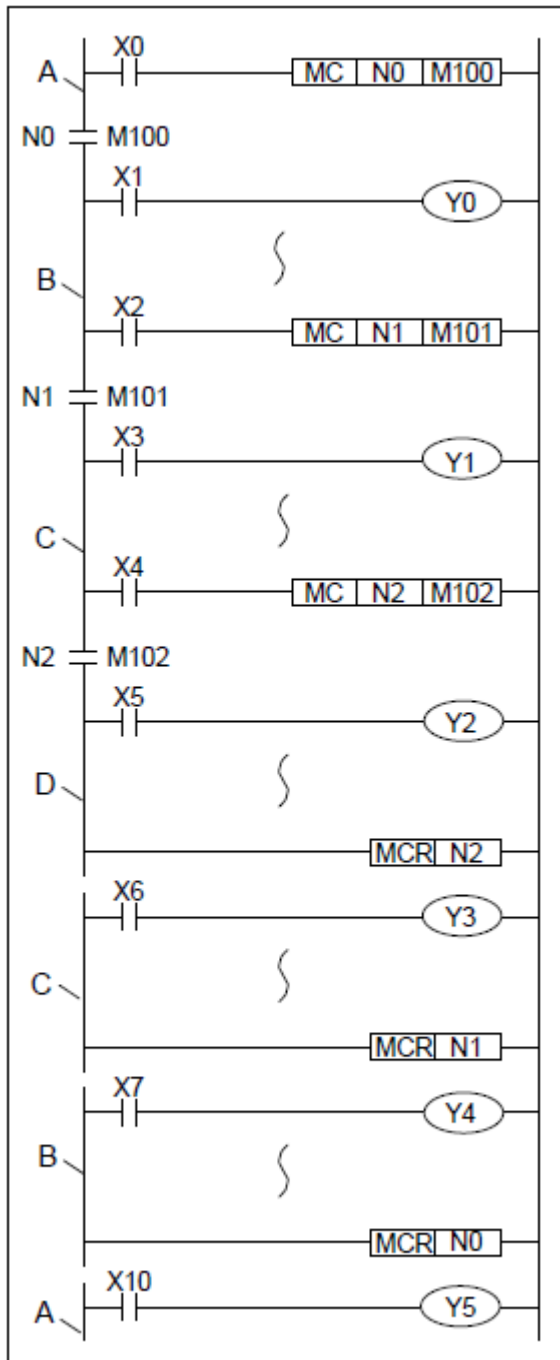
**Instruction Description**

Table 3-11

Name	Function	Devices	Format	Steps
MC (Master Control)	Denotes the start of a master control block	Y, M (no special M coils allowed) N denotes the nest level (N0 to N7)		3
MCR (Master Control Reset)	Denotes the end of a master control block	N denotes the nest level (N0 to N7) to be reset.		2

**Program example**





**Nested MC program example:**

Level N0: Bus line (B) active when X0 is ON.

Level N1: Bus line (C) active when both X0 and X2 are ON.

Level N2: Bus line (D) active when X0, X2 and X4 are ON.

Level N1: MCRN2 executes and restores bus line (C). If the MCR had reset N0 then the original bus bar (A) would now be active as all master controls below nest level 0 would reset.

Level N0: MCRN1 executes and restores bus line (B).

Initial state: MCR N0 executes and restores the initial bus line (A).

Output Y5 turns ON/OFF according to the ON/OFF state of X10, regardless of the ON/OFF status of inputs X0, X2 or X4.

**Basic points to remember**

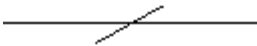
- 1) After the execution of MC instruction, the bus line (LD, LDI point) shifts to a point after the MC instruction. An MCR instruction returns this to the original bus line.
- 2) The MC instruction also includes a nest level pointer N. Nest levels are from the range N0 to N7 (8 points). The top nest level is '0' and the deepest is '7'.
- 3) The MCR instruction resets each nest level. When a nest level is reset, it also resets ALL deeper nest levels. For example, MCR N5 resets nest levels 5 to 7.
- 4) When input X0=ON, all instructions between the MC and the MCR instruction would execute.

- 5) When input X0=OFF, none of the instruction between the MC and MCR instruction execute; this resets all devices except for retentive timers, counters and devices driven by SET/RST instructions.
- 6) The MC instruction can be used as many times as necessary, by changing the device number Y and M. Using the same device number twice is processed as a double coil
- 7) Nest levels can be duplicated but when the nest level resets, all occurrences of that level reset and not just the one specified in the local MC.

### 3.2.11 INV

#### Instruction Description

Table 3-12

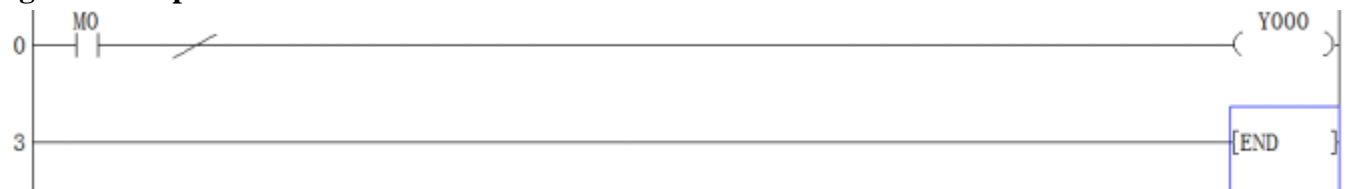
Name	Function	Devices	Format	Steps
INV(Inverse)	Invert the current result of the internal PLC operations	N/A		1

INV is the instruction which reverses the results before INV instruction and after LD, LDI, LDP, LDF instructions. And it has not operands. The instruction spends 1 process step.

#### Usages for INV

Use the invert instruction to quickly change the logic of a complex circuit. It is also useful as an inverse operation for the pulse contact instructions LDP, LDF, ANP, etc.

#### Program example





#### Basic points to remember

- 1) The INV instruction is used to change (invert) the logical state of the current ladder network at the inserted position.
- 2) Usage is the same as for AND and ANI please see [3.2.3](#)

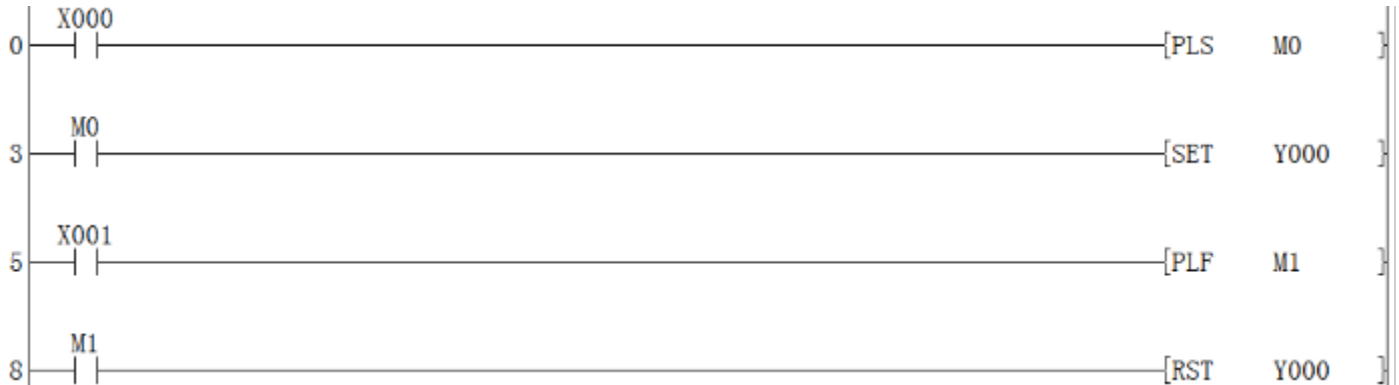
### 3.2.12 PLS, PLF (Rising edge pulse and Falling edge pulse)

#### Instruction Description

Table 3-13

Name	Function	Devices	Format	Steps
PLS (Pulse)	Rising edge pulse	Y, M (no special M coils allowed)		2
PLF(Pulse)Falling)	Falling / trailing edge pulse	Y, M (no special M coils allowed)		2

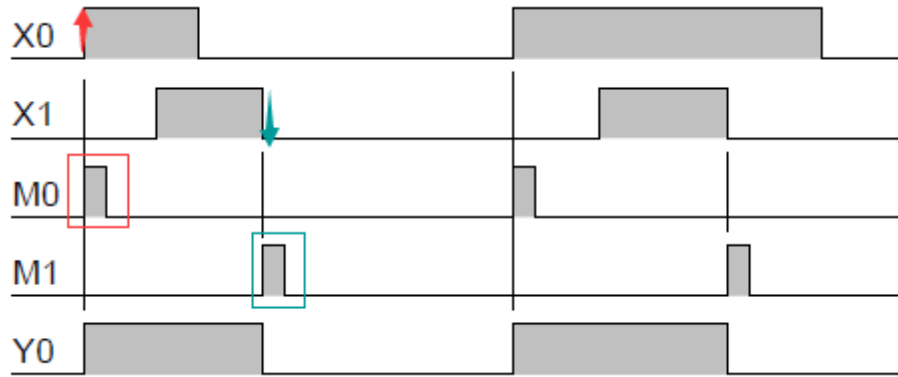
#### Program example



#### Basic points to remember

- 1) When a PLS instruction is executed, object devices Y and M operate for one operation cycle after the drive input signal has been turned ON.
- 2) When a PLF instruction is executed, object devices Y and M operate for one operation cycle after the drive input signal has been turned OFF.
- 3) When the PLC status is changed from RUN to STOP and back to RUN with the input signals still ON, PLS M0 is operated again. However, if a M coil which is battery backed (latched) was used instead of M0 it would not re-activate. For the battery backed device to be re-pulsed, its driving input (ex. X0) must be switched OFF during the RUN/STOP/RUN sequence before it will be pulsed once more.





### 3.2.13 SET, RST

#### Instruction Description

Name	Function	Devices	Format	Steps
SET (SET)	Sets a bit device permanently ON	Y, M, S	<code>—[SET device Y000 ]</code>	Y,M:1
RST (Reset)	Resets a bit device permanently OFF	Y, M, S, D, V, Z (timers and counters T,C )	<code>—[RST device Y000 ]</code>	S, special M :2 D, special D registers, V and Z:3

#### Program example



#### Basic points to remember

- 1) Turning ON X0 causes Y0 to turn ON. Y0 remains ON even after X0 turns OFF.
- 2) Turning ON X1 causes Y0 to turn OFF. Y0 remains OFF even after X1 turns OFF.
- 3) SET and RST instructions can be used for the same device as many times as necessary. However, the last instruction activated determines the current status.
- 4) It is also possible to use the RST instruction to reset the contents of data devices such as data registers, index registers etc. The effect is similar to moving 'K0' into the data device.

## 4 Applied instructions

Applied Instructions are the ‘specialist’ instructions of the TP Series PLC’s. They allow the user to perform complex data manipulations, mathematical operations while still being very easy to program and monitor. Each applied instruction has unique mnemonics and special function numbers.

### 4.1 Applied instruction list

This section lists all the applied instructions.

#### 4.1.1 Program Flow instruction list

Table 4-1

Instruction	Description
CJ	Conditional jump
CJP	Conditional jump (pulse)
CALL	Call Subroutine
CALLP	Call Subroutine (pulse)
EI	Enable Interrupt
DI	Disable Interrupt
WDT	Watchdog Timer
WDTP	Watchdog Timer (pulse)
FOR	Start of a For/Next Loop
NEXT	End a For/Next Loop

#### 4.1.2 Move And Compare instruction list

Table 4-2

Instruction	Description
CMP	16-bit compare
CMPP	16-bit compare (pulse)
DCMP	32-bit compare
DECMPP	32-bit compare (pulse)
ZCP	16-bit zone compare
ZCPP	16-bit zone compare (pulse)
DZCP	32-bit zone compare
DZCPP	32-bit zone compare (pulse)
MOV	16-bit move
MOVP	16-bit move (pulse)
DMOV	32-bit move

DMOV	32-bit move (pulse)
SMOV	shift move
SMOVP	shift move (pulse)
CML	compliment
CMLP	compliment (pulse)
BMOV	block move
BOMVP	block move (pulse)
FMOV	16-bit fill move
FMOVP	16-bit fill move (pulse)
DFMOV	32-bit fill move
DFMOVP	32-bit fill move (pulse)
XCH	16-bit exchange
XCHP	16-bit exchange (pulse)
DXCH	32-bit exchange
DXCHP	32-bit exchange (pulse)
BCD	16-bit binary coded decimal
BCDP	16-bit binary coded decimal (pulse)
DBC	32-bit binary coded decimal
DBC	32-bit binary coded decimal (pulse)
BIN	16-bit binary
BINP	16-bit binary (pulse)
DBIN	32-bit binary
DBINP	32-bit binary (pulse)

### 4.1.3 Arithmetic And Logical Operations instruction list

Table 4-3

Instruction	Description
ADD	16-bit addition
ADDP	16-bit addition (pulse)
DADD	32-bit addition
DADDP	32-bit addition (pulse)
SUB	16-bit subtraction
SUBP	16-bit subtraction (pulse)
DSUB	32-bit subtraction
DSUBP	32-bit subtraction (pulse)
MUL	16-bit multiplication
MULP	16-bit multiplication (pulse)
DMUL	32-bit multiplication

DMULP	32-bit multiplication (pulse)
DIV	16-bit division
DIVP	16-bit division (pulse)
DDIV	32-bit division
DDIVP	32-bit division (pulse)
INC	16-bit increment
INCP	16-bit increment (pulse)
DINC	32-bit increment
DINCP	32-bit increment (pulse)
DEC	16-bit decrement
DECP	16-bit decrement (pulse)
DDEC	32-bit decrement
DDECP	32-bit decrement (pulse)
WAND	16-bit word AND
WANDP	16-bit word AND (pulse)
DAND	32-bit word AND
DANDP	32-bit word AND (pulse)
WOR	16-bit word OR
WORP	16-bit word OR (pulse)
DOR	32-bit word OR
DORP	32-bit word OR (pulse)
WXOR	16-bit word exclusive OR
WXORP	16-bit word exclusive OR (pulse)
DXOR	32-bit word exclusive OR
DXORP	32-bit word exclusive OR (pulse)
NEG	16-bit negation
NEGP	16-bit negation (pulse)
DNEG	32-bit negation
DNEGP	32-bit negation (pulse)

#### 4.1.4 Rotation and Shift

Table 4-4

Instruction	Description
ROR	16-bit rotation right
RORP	16-bit rotation right (pulse)
DROR	32-bit rotation right
DRORP	32-bit rotation right (pulse)
ROL	16-bit rotation left

ROLP	16-bit rotation left (pulse)
DROL	32-bit rotation left
DROLP	32-bit rotation left (pulse)
RCR	16-bit rotation right with carry
RCRP	16-bit rotation right with carry (pulse)
DRCR	32-bit rotation right with carry
DRCRP	32-bit rotation right with carry (pulse)
RCL	16-bit rotation left with carry
RCLP	16-bit rotation left with carry (pulse)
DRCL	32-bit rotation left with carry
DRCLP	32-bit rotation left with carry (pulse)
SFTR	(bit) shift right
SFTRP	(bit) shift right (pulse)
SFTL	(bit) shift left
SFTLP	(bit) shift left (pulse)
WSFR	word shift right
WSFRP	word shift right (pulse)
WSFL	word shift left
WSFLP	word shift left (pulse)
SFWR	shift register write
SFWRP	shift register write (pulse)
SFRD	shift register read
SFRDP	shift register read (pulse)

### 4.1.5 Data operation

Table 4-5

Instruction	Description
ZRST	zone reset
ZRSTP	zone reset (pulse)
DECO	decode
DECOP	decode (pulse)
ENCO	encode
ENCOP	encode (pulse)
SUM	16-bit the sum of active bits
SUMP	16-bit the sum of active bits (pulse)
DSUM	32-bit the sum of active bits
DSUMP	32-bit the sum of active bits (pulse)
BON	16-bit check specified bit status

BONP	16-bit check specified bit status (pulse)
DBON	32-bit check specified bit status
DBONP	32-bit check specified bit status (pulse)
MEAN	16-bit mean
MEANP	16-bit mean (pulse)
DMEAN	32-bit mean
DMEANP	32-bit mean (pulse)
ANS	(timed) annunciator set
ANR	annunciator reset
ANRP	annunciator reset (pulse)
SQR	16-bit square root
SQRP	16-bit square root (pulse)
DSQR	32-bit square root
DSQRP	32-bit square root (pulse)
FLT	16-bit float
FLTP	16-bit float (pulse)
DFLT	32-bit float
DFLTP	32-bit float (pulse)
SWAP	16-bit high and low bit conversion
SWAPP	16-bit high and low bit conversion (pulse)
DSWAP	32-bit high and low bit conversion
DSWAPP	32-bit high and low bit conversion (pulse)

#### 4.1.6 High Speed Processing instruction list

Table 4-6

Instruction	Description
REF	refresh
REFP	refresh (pulse)
REFF	refresh and filter adjust
REFFP	refresh and filter adjust (pulse)
MTR	input matrix
DHSCR	high speed counter set
DHSCS	high speed counter reset
DHSZ	high speed counter zone compare
SPD	speed detect
PLSY	16-bit pulse Y output
DPLSY	32-bit pulse Y output
PWM	pulse width modulation

PLSR	16-bit ramp pulse output
DPLSR	32-bit ramp pulse output
PTO	envelope pulse output

### 4.1.7 Handy Instructions list

Table 4-7

Instruction	Description
IST	initial state
SER	16-bit search
SERP	16-bit search (pulse)
DSER	32-bit search
DSERP	32-bit search (pulse)
ABSD	16-bit absolute drum
DABSD	32-bit absolute drum
INCD	incremental drum
TTMR	teaching timer
STMR	special timer - definable
ALT	alternate state
ALTP	alternate state (pulse)
RAMP	ramp - variable value
ROTC	rotary table control
SORT	sort data

### 4.1.8 External I/O Devices instruction list

Table 4-8

Instruction	Description
TKY	16-bit ten key input
DTKY	32-bit ten key input
HKY	16-bit hexadecimal input
DHKY	32-bit hexadecimal input
DSW	digital switch (thumbwheel input)
SEGD	seven segment decoder
SEGDP	seven segment decoder (pulse)
SEGL	seven segment with latch
ARWS	arrow switch
ASC	ASCII code
PR	“print” to a display
FROM	16-bit Read from a special function block

FROMP	16-bit Read from a special function block (pulse)
DFROM	32-bit Read from a special function block
DFROMP	32-bit Read from a special function block (pulse)
TO	16-bit write to a special function block
TOP	16-bit write to a special function block (pulse)
DTO	32-bit write to a special function block
DTOP	32-bit write to a special function block (pulse)

#### 4.1.9 External Devices instruction list

Table 4-9

Instruction	Description
RS	RS communications
PRUN	16-bit octal bit transmission
PRUNP	16-bit octal bit transmission (pulse)
DPRUN	32-bit octal bit transmission
DPRUNP	32-bit octal bit transmission (pulse)
ASCI	hexadecimal to ASCII
ASCIP	hexadecimal to ASCII (pulse)
HEX	ASCII to hexadecimal
HEXP	ASCII to hexadecimal (pulse)
CCD	check code
CCDP	check code (pulse)
PID	PID control loop

#### 4.1.10 Floating Point instruction list

Table 4-10

Instruction	Description
DECOMP	Binary floating point data compare
DECMPP	Binary floating point data compare(pulse type)
DEZCP	Binary floating point zone compare
DEZCPP	Binary floating point zone compare(pulse type)
DEBCD	Binary to BCD floating point data conversion
DEBCDP	Binary to BCD floating point data conversion(pulse type)
DEBIN	BCD to Binary floating point data conversion
DEBINP	BCD to Binary floating point data conversion(pulse type)
DEADD	Binary floating point addition
DEADDP	Binary floating point addition(pulse type)
DESUB	Binary floating point subtraction



DESUBP	Binary floating point subtraction(pulse type)
DEMUL	Binary floating point multiplication
DEMULP	Binary floating point multiplication(pulse type)
DEDIV	Binary floating point division
DEDIVP	Binary floating point division(pulse type)
DESQR	Binary floating point square root
DESQRP	Binary floating point square root(pulse type)
INT	16-bit binary floating point to integer
INTP	16-bit binary floating point to integer(pulse type)
DINT	32-bit binary floating point to integer
DINTP	32-bit binary floating point to integer(pulse type)

#### 4.1.11 Positioning Instruction list

Table 4-11

Instruction	Description
DABS	Absolute current value read
ZRN	Setting of zero return speed
PLSV	Variable speed pulse output
DRVI	Relative position control
DRVA	Absolute position control

#### 4.1.12 Real Time Clock Control

Table 4-12

Instruction	Description
TCMP	time compare
TCMPP	time compare (pulse)
TZCP	time zone compare
TZCPP	time zone compare (pulse)
TADD	time add
TADDP	time add (pulse)
TSUB	time subtract
TSUBP	time subtract (pulse)
TRD	read RTC data
TRDP	read RTC data (pulse)
TWR	set RTC data
TWRP	set RTC data (pulse)
HOUR	16-bit chronograph
DHOUR	32-bit chronograph

### 4.1.13 Gray Codes instruction list

Table 4-13

Instruction	Description
GRY	16-bit Gray code conversion
GRYP	16-bit Gray code conversion (pulse)
DGRY	32-bit Gray code conversion
DGRYP	32-bit Gray code conversion (pulse)
GBIN	16-bit Gray code inverted conversion
GBINP	16-bit Gray code inverted conversion (pulse)
DGBIN	32-bit Gray code inverted conversion
DGBINP	32-bit Gray code inverted conversion (pulse)
GRY	16-bit Gray code conversion
GRYP	16-bit Gray code conversion (pulse)
DGRY	32-bit Gray code conversion
DGRYP	32-bit Gray code conversion (pulse)
GBIN	16-bit Gray code inverted conversion
GBINP	16-bit Gray code inverted conversion (pulse)

### 4.1.14 Inline Comparisons Instruction list

Table 4-14

Instruction	Description
LD=	Comparison of 16-bit data (==)
LDD=	Comparison of 32-bit data (==)
LD>	Comparison of 16-bit data (>)
LDD>	Comparison of 32-bit data (>)
LD<	Comparison of 16-bit data (<)
LDD<	Comparison of 32-bit data(<)
LD<>	Comparison of 16-bit data (<>)
LDD<>	Comparison of 32-bit data (<>)
LD<=	Comparison of 16-bit data (<=)
LDD<=	Comparison of 32-bit data (<=)
LD>=	Comparison of 16-bit data (>=)
LDD>=	Comparison of 32-bit data (>=)
AND=	Comparison of 16-bit data (==)
ANDD=	Comparison of 32-bit data (==)
AND>	Comparison of 16-bit data (>)
ANDD>	Comparison of 32-bit data (>)

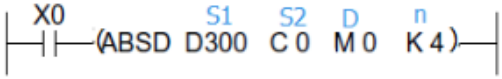
AND<	Comparison of 16-bit data (<)
ANDD<	Comparison of 32-bit data (<)
AND<>	Comparison of 16-bit data (<>)
ANDD<>	Comparison of 32-bit data (<>)
AND<=	Comparison of 16-bit data (<=)
ANDD<=	Comparison of 32-bit data (<=)
AND>=	Comparison of 16-bit data (>=)
ANDD>=	Comparison of 32-bit data (>=)
OR=	Comparison of 16-bit data (==)
ORD=	Comparison of 32-bit data (==)
OR>	Comparison of 16-bit data (>)
ORD>	Comparison of 32-bit data (>)
OR<	Comparison of 16-bit data (<)
ORD<	Comparison of 32-bit data (<)
OR<>	Comparison of 16-bit data (<>)
ORD<>	Comparison of 32-bit data (<>)
OR<=	Comparison of 16-bit data (<=)
ORD<=	Comparison of 32-bit data (<=)
OR>=	Comparison of 16-bit data (>=)
ORD>=	Comparison of 32-bit data (>=)

## 4.2 Applied instruction description

### 4.2.1 ABSD instruction

#### Instruction Description

Table 4-15

Name	Function	Devices				Format	Steps
		S1	S2	D	n		
ABSD (Absolute drum sequencer)	Generates multiple output patterns in response to counter data	KnX, KnY, KnM, KnS, (in groups of 8) T, C, D	C	Y,M,S Note: n consecutive devices are used	K,H Note: N<=6 4		ABSD: 9 steps DABS: 17 steps.
		Note: High speed are not allowed					

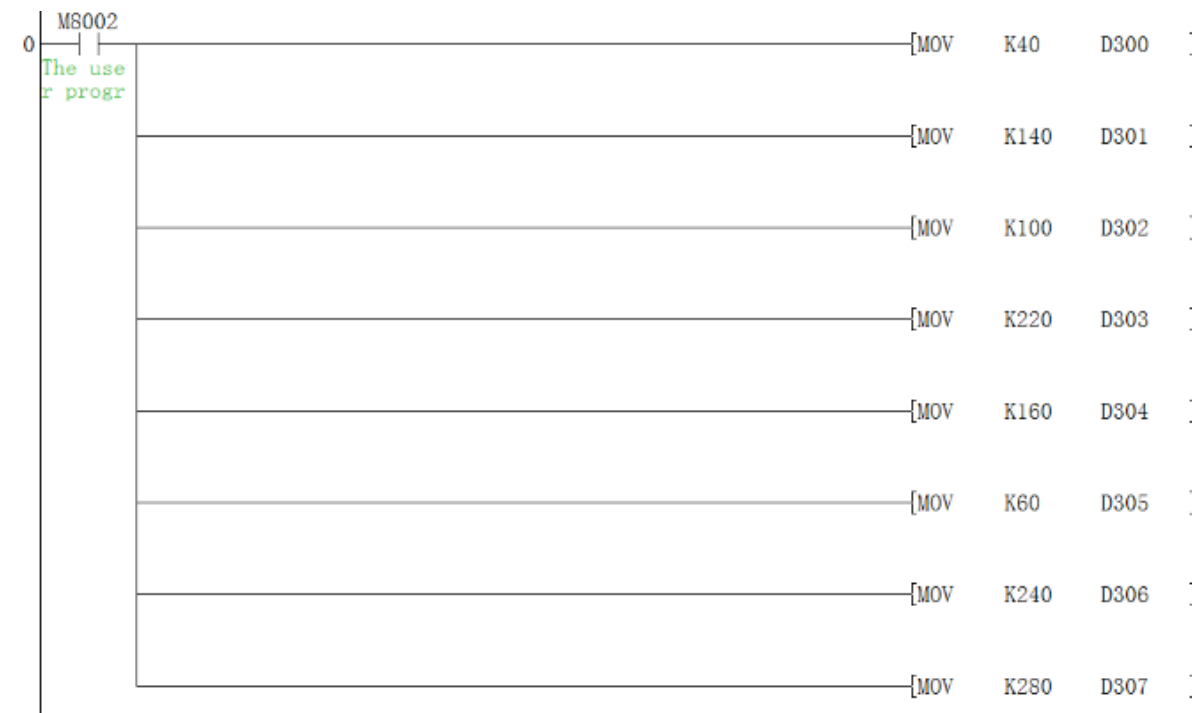
**Operation**

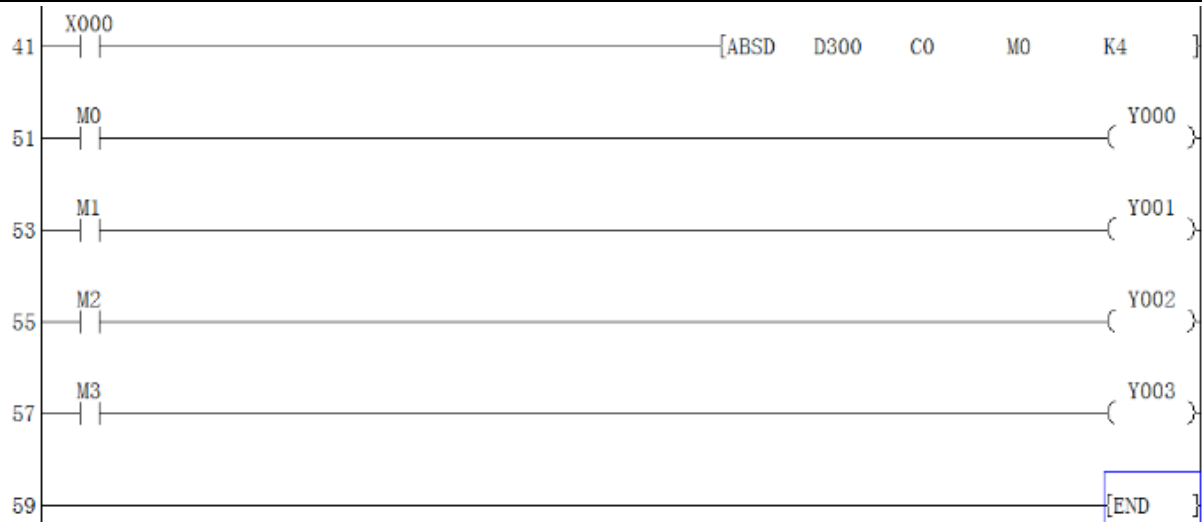
This instruction generates a variety of output patterns (there are n numbers of addressed outputs) in response to the current value of a selected counter, S2.

**Points to note**

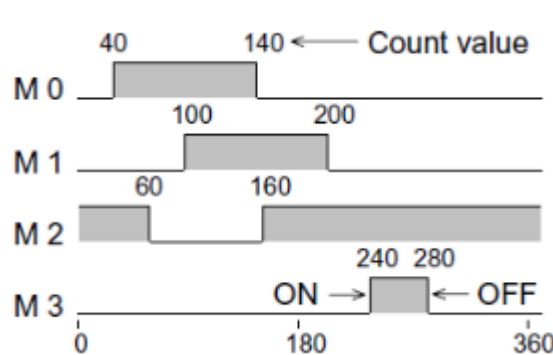
- 1) The current value of the selected counter (S2) is compared against a user defined data table. This data table has a head address identified by operand S1. S1 should always have an even device number.
- 2) For each destination bit (D) there are two consecutive values stored in the data table. The first allocated value represents the event number when the destination device (D) will be turned ON. The second identifies the reset event. The data table values are allocated as a consecutive pair for each sequential element between D and D+n.
- 3) The data table has a length equal to 2 \* n data entries. Depending on the format of the data table, a single entry can be one data word such as D300 or a group of 16 bit devices e.g. K4X000.
- 4) Values from 0 to 32,767 may be used in the data table.
- 5) The ABSD instruction may only be used **ONCE**.

**Program example**





When counter S2 equals the value below, the destination device D is		Assigned Destination device D
turned ON	turned OFF	
D300 =40	D301 - 140	M0
D302 - 100	D303 - 200	M1
D304 - 160	D305 - 60	M2
D306 - 240	D307 - 280	M3



### 4.2.2 ADD instruction

#### Instruction Description

Table 4-16

Name	Function	Devices			Format	Steps
		S1	S2	D		
ADD (Addition)	The value of the two source devices is added and the	K,H, KnX, KnY, KnM, KnS,		KnY, KnM, KnS, T, C, D, V, Z		ADD, ADDP: 7 steps  DADD,

	result stored in the destination device	T, C, D, V, Z			DADDP : 13 steps
--	---	---------------	--	--	------------------

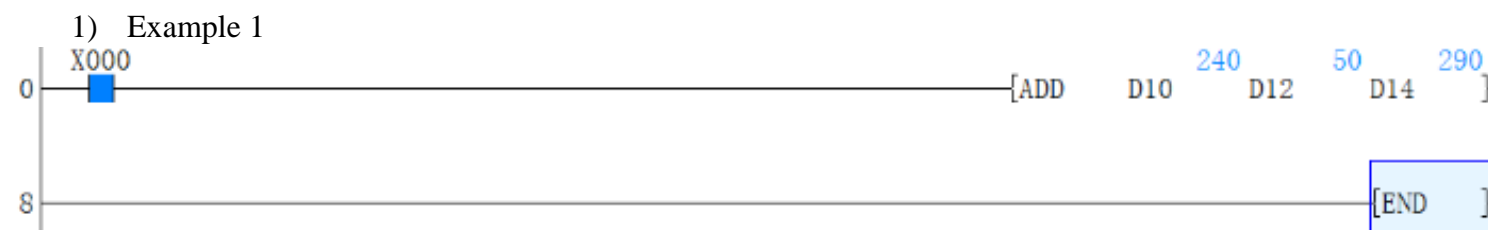
**Operation (Applicable to all units)**

The data contained within the source devices (S1,S2) is combined and the total is stored at the specified destination device (D).

**Points to note**

- 1) All calculations are algebraically processed, i.e.  $5 + (-8) = -3$ .
- 2) The same device may be used as a source (S1 or S2) and as the destination (D). If this is the case then the ADD instruction would actually operate continuously. This means on every scan the instruction would add the result of the last scan to the second source device. To prevent this from happening the pulse modifier should be used or an interlock should be programmed.
- 3) If the result of a calculation is "0" then a special auxiliary flag, M8020 is set ON.
- 4) If the result of an operation exceeds 32,767 (16 bit limit) or 2,147,483,647 (32 bit limit) the carry flag, M8022 is set ON. If the result of an operation exceeds -32,768 or -2,147,483,648 the borrow flag, M8021 is set ON. When a result exceeds either of the number limits, the appropriate flag is set ON (M8021 or M8022) and a portion of the carry/borrow is stored in the destination device. The mathematical sign of this stored data is reflective of the number limit which has been exceeded, i.e. when -32,768 is exceeded negative numbers are stored in the destination device but if 32,767 was exceeded positive numbers would be stored at D.
- 5) If the destination location is smaller than the obtained result, then only the portion of the result which directly maps to the destination area will be written, i.e if 25 (decimal) was the result, and it was to be stored at K1Y4 then only Y4 and Y7 would be active. In binary terms this is equivalent to a decimal value of 9 a long way short of the real result of 25!

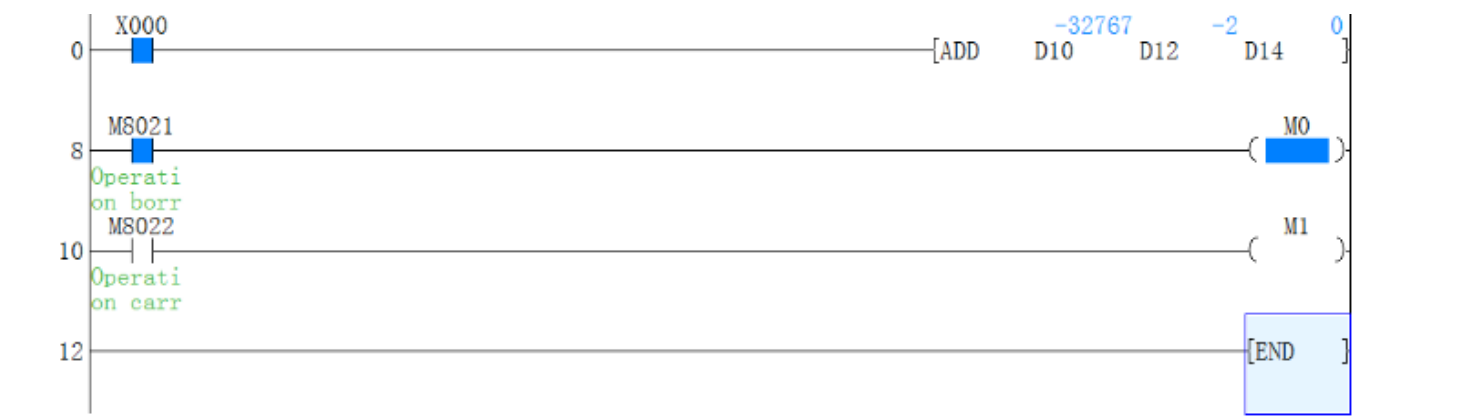
**Program example**



2) Example 2



3) Example 3



4.2.3 ALT instruction

Instruction Description

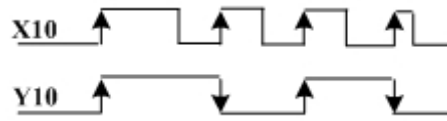
Table 4-17

Name	Function	Devices	Format	Steps
		D		
ALT (Alternate state)	The status of the assigned device is inverted on every operation of the instruction	Y, M, S		ALT, ALTP: 3 steps

Operation

- 1) The status of the destination device (D) is alternated on every operation of the ALT instruction.
- 2) This means the status of each bit device will flip-flop between ON and OFF. This will occur on every program scan unless a pulse modifier or a program interlock is used.
- 3) The ALT instruction is ideal for switching between two modes of operation e.g. start and stop, on and off etc.

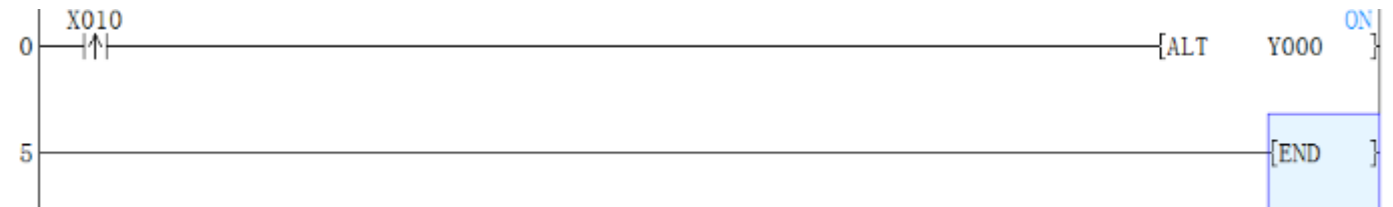
Program example



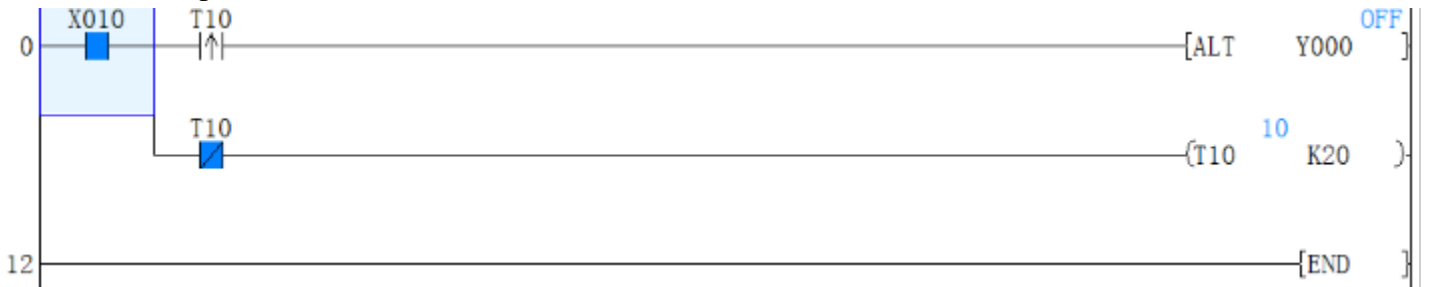
1) Example 1



The following instruction operation is the same:



2) Example 2



The following instruction operation is the same:





### 4.2.4 ANR instruction

**Instruction Description:**

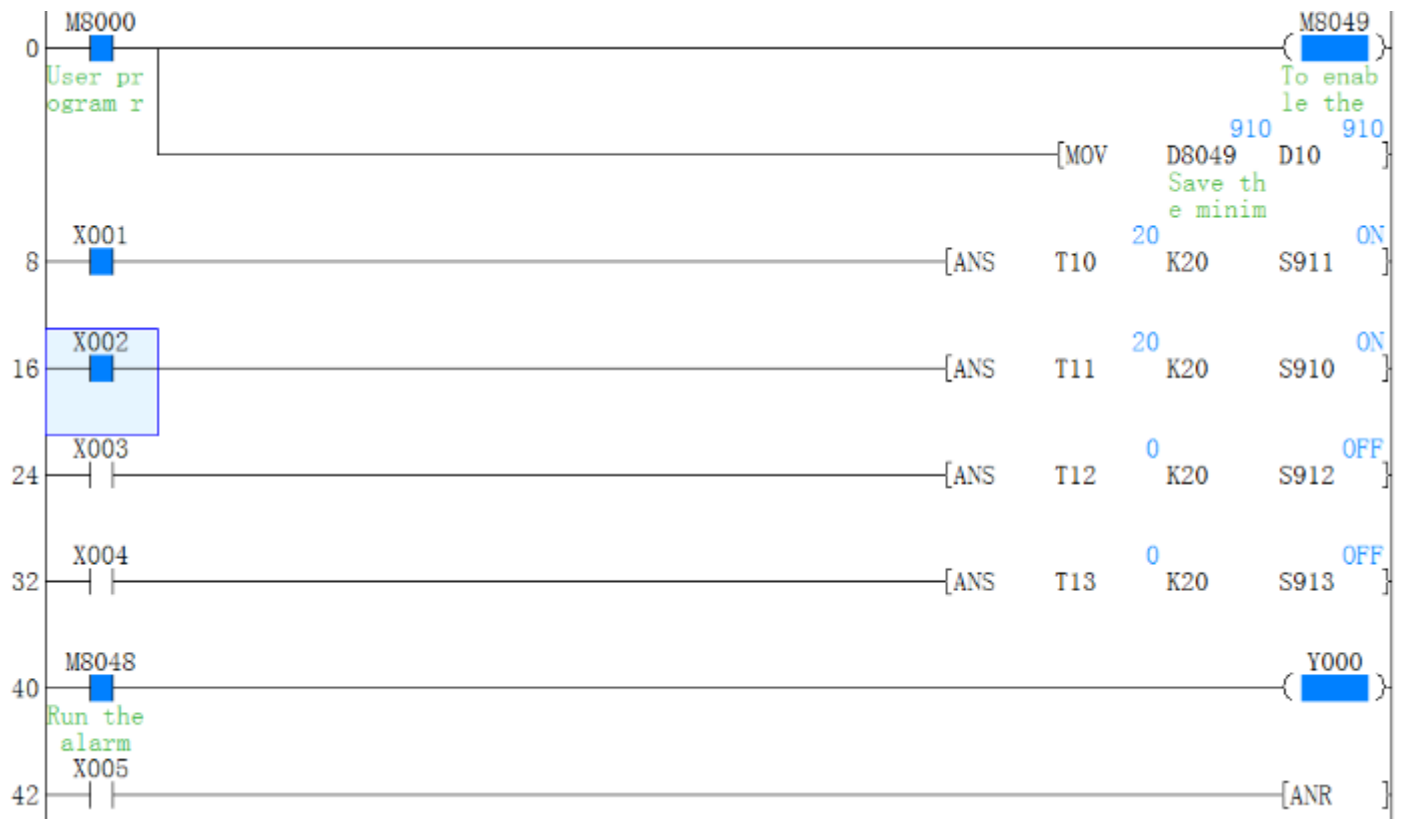
Table 4-18

Name	Function	Devices	Format	Steps
		D		
ANR (Annunciator reset)	The lowest active annunciator is reset on every operation of this instruction	N/A		ANR,ANRP: 1step

**Operation**

- 1) Annunciators which have been activated are sequentially reset one-by-one, each time the ANR instruction is operated.
- 2) If the ANR instruction is driven continuously it will carry out its resetting operation on every program scan unless it is modified by the pulse, P prefix or by a user defined program interlock.

**Program example**



If X5 set ON, S900--S999 will set off.

### 4.2.5 ANS instruction

#### Instruction Description

Table 4-19

Name	Function	Devices			Format	Steps
		S	D	n		
ANS (Timed annunciator set)	This instruction starts a timer. Once timed out the selected annunciator flag is set ON	T Note: Available range T0 to T199	S Note: Annunciator range S900 to S999	K Note: n range 1 to 32,767 - in units of 100msec		ANS: 7 steps

#### Operation

This instruction, when energized, starts a timer (S) for n,100 msec. When the timer completes its cycle the assigned annunciator (D) is set ON.

If the instruction is switched OFF during or after completion of the timing cycle the timer is automatically reset. However, the current status of the annunciator coil remains unchanged.

**Note:** This is only one method of driving annunciator coils, others such as direct setting can also be used.

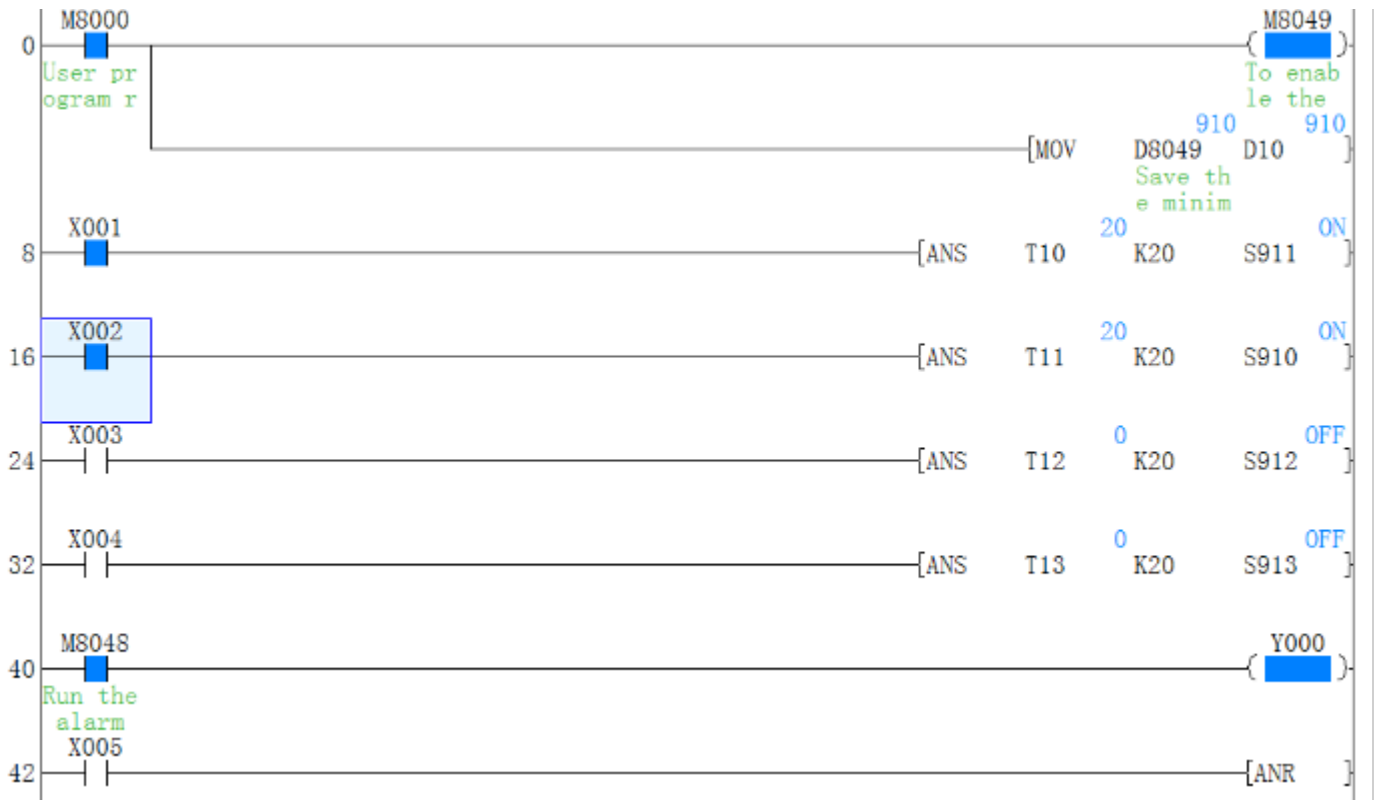
#### Program example

1) Example 1



If X1 and X2 are connected for more than 1 second, S911 is set ON. Following that, S911 will keep ON, Even if X1 or X2 is set to OFF and T10 can be reset to 0). If X1 and X2 are connected for less than 1 second, X1 or X2 set to OFF. Then the timer will reset.

2) Example 2



If M8049 (signal alarm is available) is set to ON in begin, the lowest number of S that is set ON will be saved at D8049. (the lowest S number with the ON state). when any signal in S900~S999 is ON then M8048 is set to ON (alarm operation).

4.2.6 ARWS instruction

Instruction Description

Table 4-20

Name	Function	Devices				Format	Steps
		S	D1	D2	n		
ARWS (Arrow switch)	Creates a user defined, (4 key) numeric data entry panel	X, Y, M, S Note: uses 4 consecutive devices	T, C, D, V, Z Note: data is stored in a decimal format	Y Note: uses 8 consecutive devices	K, H Note: n= 0 to 3,	$\left  \left  \left  \left( \text{ARWS } X \ 10 \ D \ 0 \ Y \ 0 \ K \ 0 \right) \right  \right  \right $	ARWS: 9 steps

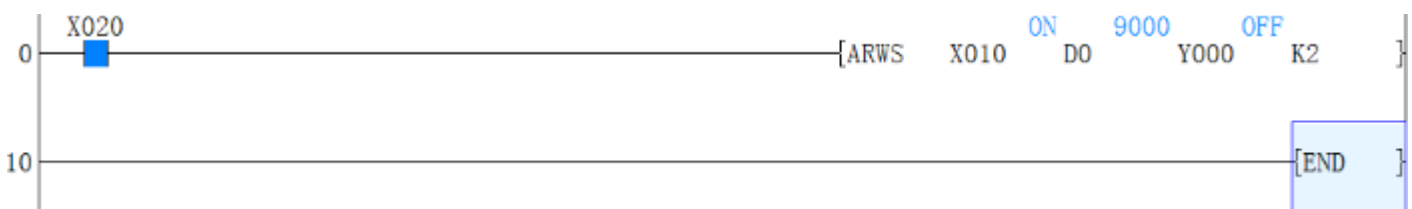
### Operation

This instruction displays the contents of a single data device D<sub>1</sub> on a set of 4 digit, seven segment displays. The data within D<sub>1</sub> is actually in a standard decimal format but is automatically converted to BCD for display on the seven segment units. Each digit of the displayed number can be selected and edited. The editing procedure directly changes the value of the device specified as D<sub>1</sub>.

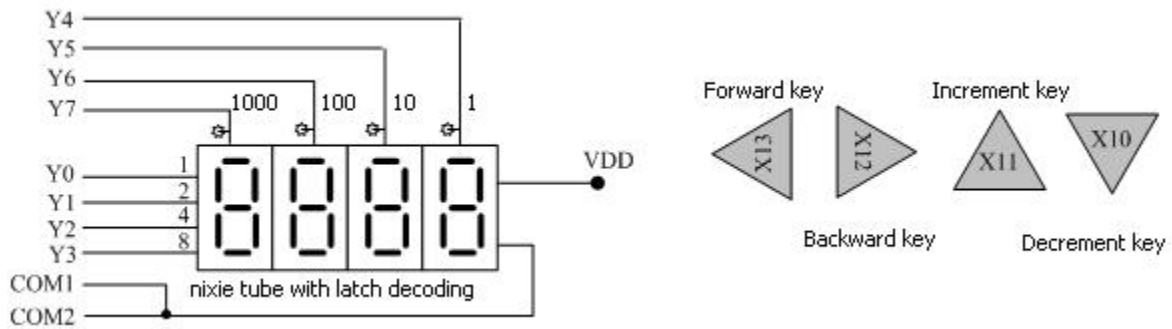
### Points to note

- 1) The data stored in destination device D<sub>1</sub> can have a value from the range 0 to 9,999 (decimal), i.e. 4 digit data. Each digit's data value, can be incremented (S<sub>+1</sub>) or decremented (S<sub>+0</sub>) by pressing the associated control keys. The edited numbers automatically 'wrap-around' from 9 - 0 - 1 and 1 - 0 - 9. The digit data is displayed by the lower 4 devices from D<sub>2</sub>, i.e. D<sub>2+0</sub> to D<sub>2+3</sub>.
- 2) On initial activation of the ARWS instruction, the digit in the numeric position 10<sub>3</sub> is currently selected. Each digit position can be sequentially 'cursor through' by moving to the left (S<sub>+2</sub>) or to the right (S<sub>+3</sub>). When the last digit is reached, the ARWS instruction automatically wraps the cursor position around, i.e. after position 10<sub>3</sub>, position 10<sub>0</sub> is selected and vice-versa. Each digit is physically selected by a different 'strobe' output.
- 3) To aid the user of an operation panel controlled with the ARWS instruction, additional lamps could be wired in parallel with the strobe outputs for each digit. This would indicate which digit was currently selected for editing.
- 4) The parameter n has the same function as parameter n of the SEGL instruction. Selecting the correct value for operand n'. Note: as the ARWS instruction only controls one set of displays only values of 0 to 3 are valid for n.
- 5) The ARWS instruction can be used **ONCE**. This instruction should only be used on transistor output PLC.

### Programming example



The corresponding hardware wiring is shown in the following figure, in which PLC is the transistor output type:



### 4.2.7 ASC instruction

#### Instruction Description

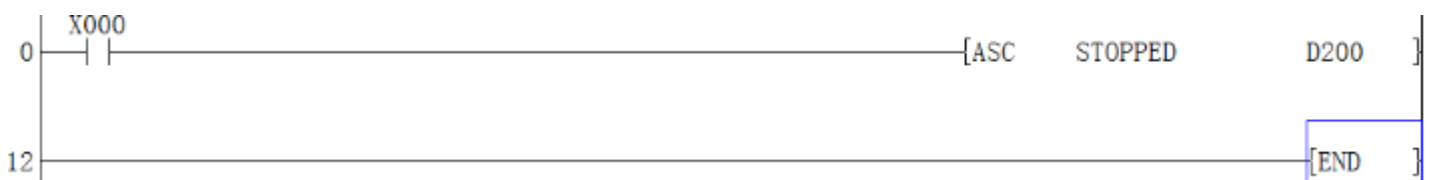
Table 4-21

Name	Function	Devices		Format	Steps
		S	D1		
ASC (ASCII code conversion)	An entered Alphanumeric string can be converted to its ASCII codes	Alphanumeric data e.g. 0-9, A-Z and a-z etc. Note: Only one, 8 character string may be entered at any one time.	T, C, D Note: uses 4 consecutive devices		ASC :7 steps

#### Operation

The source data string S consists of up to 8 characters taken from the printable ASCII character (Char) set. If less than 8 Char are used, the difference is made up with null Char (ASCII00). The source data is converted to its associated ASCII codes. The codes are then stored in the destination devices D, see example shown below.

#### Program example





The values of D200-D203 when X20=ON are shown in the left chart.

	High byte	Low byte
D200	54 (t)	53 (S)
D201	50 (p)	4F (o)
D202	45 (e)	50 (p)
D203	20	44 (d)

If the special register M8161 is set to ON, every ASCII occupies one 16bit variable after conversion, which is shown in the Figure 4-1 and the higher byte of every variable is set to 0.

	High byte	Low byte
D200	00	53 (S)
D201	00	54 (t)
D202	00	4F (o)
D203	00	50 (p)
D204	00	50 (p)
D205	00	45 (e)
D206	00	44 (d)
D207	00	20

Figure 4-1

Attached: "ASCII code parallel Table"

Table 4-22

Decimal digit	ASCII (Hex)
0	30
1	31
2	32
3	33
4	34
5	35
6	36
7	37
8	38
9	39

Table 4-23

English letter	ASCII (Hex)	English letter	ASCII (Hex)
A	41	N	4E
B	42	O	4F
C	43	P	50
D	44	Q	51
E	45	R	52

F	46	S	53
G	47	T	54
H	48	U	55
I	49	V	56
J	4A	W	57
K	4B	X	58
L	4C	Y	59
M	4D	Z	5A

Table 4-24

Code	ASCII (Hex)
STX	02
ETX	03

### 4.2.8 ASCII instruction

#### Instruction Description

Table 4-25

Name	Function	Devices			Format	Steps
		S	D	n		
ASCII (Converts HEX to ASCII)	Converts a data value from hexadecim al to ASCII	K, H, KnX, KnY, KnM, KnS T, C, D, V, Z	KnY, KnM, KnS T, C, D	K, H Note: n = 1 to 256		ASCII, ASCIP: 7 steps

#### Operation

This instruction reads n hexadecimal data characters from head source address (S) and converts them in to the equivalent ASCII code. This is then stored at the destination (D) for n number of bytes.

#### Points to note

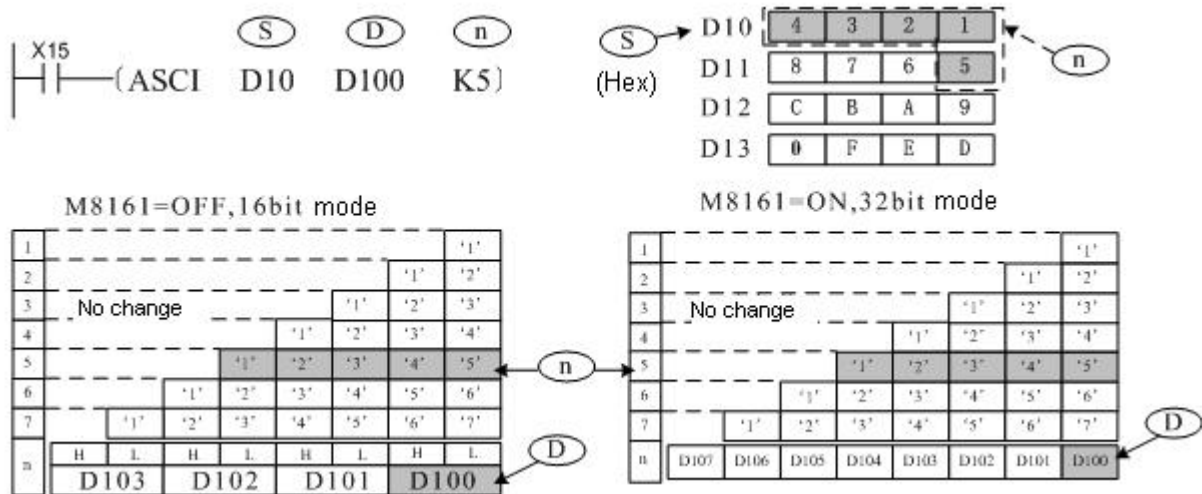
- 1) Please note that data is converted ‘as read’, i.e. using the example above with the following data in (D9, D8) ABCDH, EF26H. Taking the first n hexadecimal characters (digits) from the right (in this case n= 6) and converting them to ASCII will store values in 6 consecutive bytes from D20, i.e. D20 = (67, 68), D21 = (69, 70) and D22 = (50, 54) respectively. In true characters symbols that would be read as CDEF26.

Source (S)		Data	Destination (D)		ASCII Code		Symbol		
					HEX	DEC			
D9	b12-15	A	D20	b8-15	43	67	'C'		
	b8-11	B		D21	b0-7	44	68	'D'	
	b4-7	C			D22	b8-15	45	69	'E'
	b0-3	D				b0-7	46	70	'F'
D8	b12-15	E	D22		b8-15	32	50	'2'	
	b8-11	F		D25	b0-7	36	54	'6'	
	b4-7	2							
	b0-3	6							

Figure 4-2

- This can be shown graphically as in the table to the right. Please take special note that the source data (S) read from the most significant device to the least significant. While the destination data (D) is read in the opposite direction.
- The ASCII instruction can be used with the M8161, 8 bit/16bit mode flag. The example to the up shows the effect when M8161 is OFF.
- If M8161 was set to ON, then only the lower destination byte (b0-7) would be used to store data and hence 6 data registers would be required (D20 through D25).

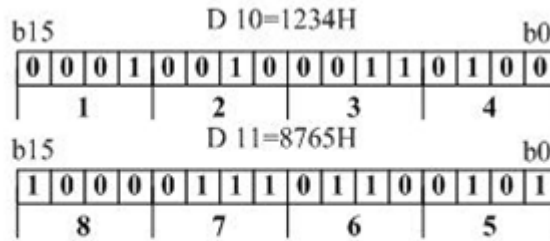
**Program example**



The M8161 flag determines the width mode of the target variable for calculation result storage. When M8161=OFF, it is 16bit mode, which means the higher byte and lower byte are saved respectively. When

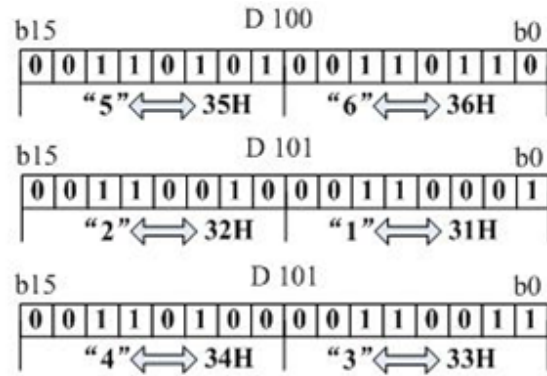
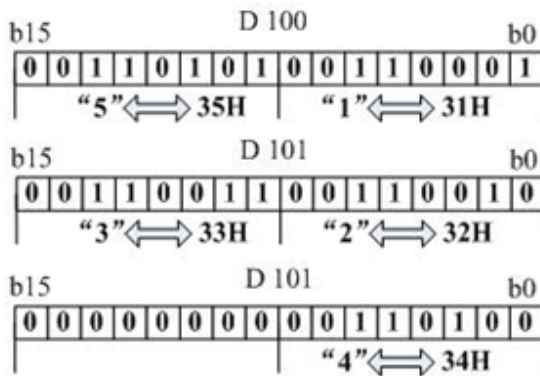


M8161=ON, it is 8bit mode, which means that only the lower byte is used to save result and the actual variable range length is longer.



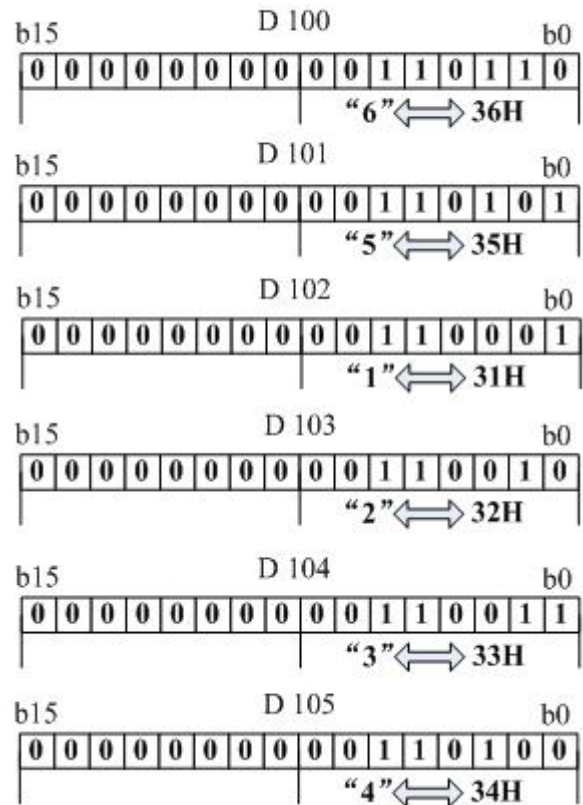
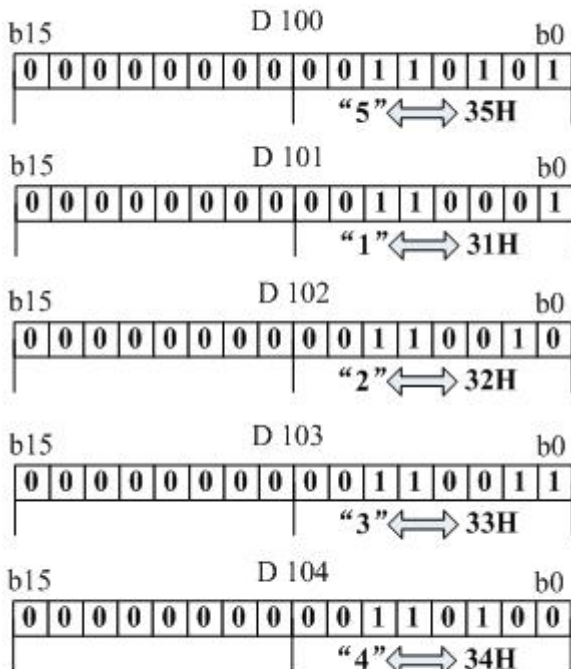
Bit component when M8161=OFF, n=5  
(D10-D11) conversion

Bit component when M8161=OFF, n=6  
(D10-D11) conversion



Bit component when M8161=ON, n=5  
(D10-D11) conversion

Bit component when M8161=ON, n=6  
(D10-D11) conversion



Note: RS/ HEX/ ASCII/ CCD instructions share the M8161 mode flag.

### 4.2.9 BCD instruction

#### Instruction Description

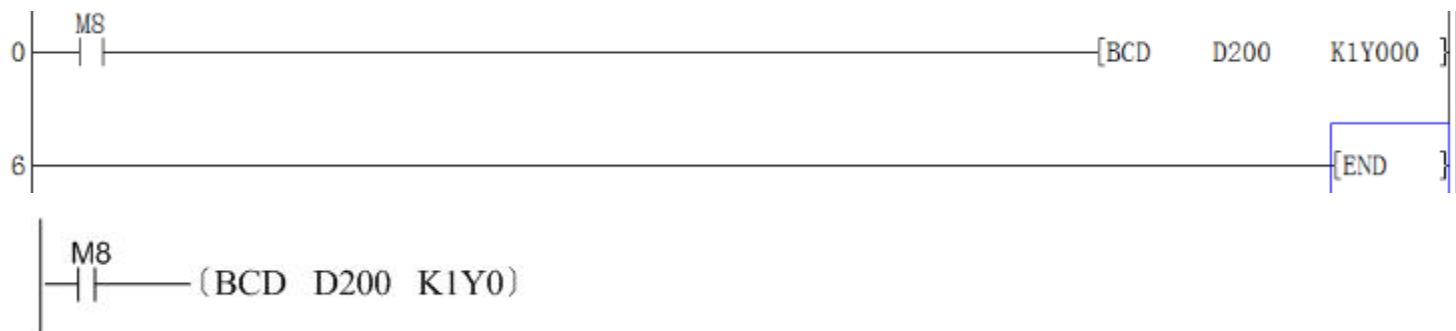
Table 4-26

Name	Function	Devices		Format	Steps
		S	D		
BCD(Binary coded decimal)	Converts binary numbers to BCD equivalents / Converts floating point data to scientific format	KnX,KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z		ANR,ANR P: 1step

#### Operation (Applicable to all units)

The binary source data (S) is converted into an equivalent BCD number and stored at the destination device (D). If the converted BCD number exceeds the operational ranges of 0 to 9,999 (16 bit operation) and 0 to 99,999,999 (32 bit operation) an error will occur. This instruction can be used to output data directly to a seven segment display.

#### Program example



The BIN value in D200 is converted to BCD value and the units of the result are saved to K1Y0 (four bit components Y0~Y3).

If D200=H000E (hex) = K14 (decimal), then Y0~Y3=0100 (BIN) after conversion.

If D200=H0028 (hex) = K40 (decimal), then Y0~Y3=0000 (BIN) after conversion.

### 4.2.10 BIN conversion

#### Instruction Description

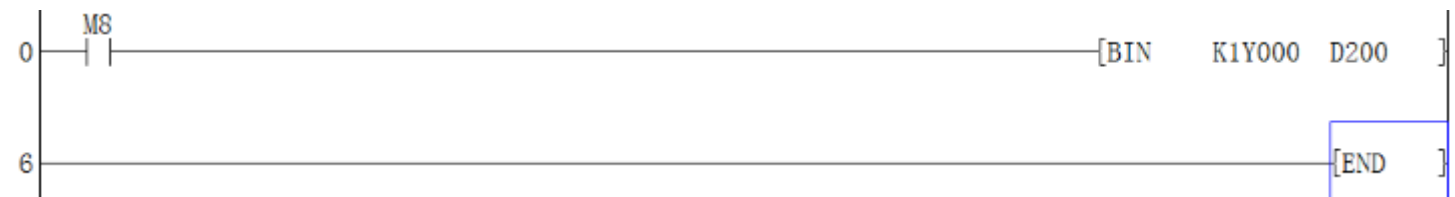
Table 4-27

Name	Function	Devices		Format	Steps
		S	D		
BIN (Binary)	Converts BCD numbers to their binary equivalent Converts scientific format data to floating point format	KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z		BIN, BINP: 5 steps  DBIN, DBINP: 9 steps

**Operation**

The BCD source data (S) is converted into an equivalent binary number and stored at the destination device (D). If the source data is not provided in a BCD format an error will occur. This instruction can be used to read in data directly from thumbwheel switches.

**Program example**



**4.2.11 BMOV instruction**

**Instruction Description**

Table 4-28

Name	Function	Devices			Format	Steps
		S	D	N		
BMOV (Block move)	Copies a specified block of multiple data elements to a new destination	KnX, KnY, KnM, KnS, T,C,D, V, Z (RAM) File registers,	KnY, KnM, KnS, T, C, D, V, Z (RAM) File registers, see noted)	K, H D  Note: n£ 51 2		BMOV, BMOV P: 7 steps

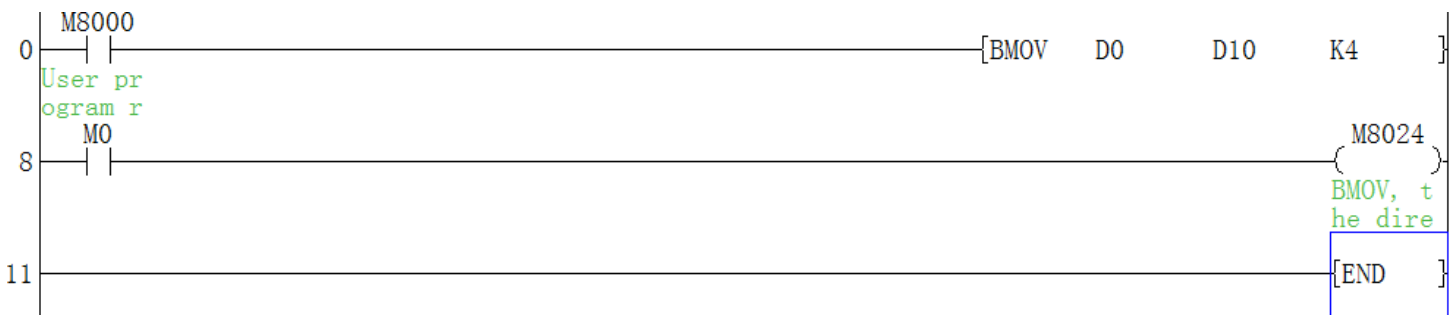
**Operation**

A quantity of consecutively occurring data elements can be copied to a new destination. The source data is identified as a device head address (S) and a quantity of consecutive data elements (n). This is moved to the destination device (D) for the same number of elements (n).

**Points to note**

- 1) If the quantity of source devices (n) exceeds the actual number of available source devices, then only those devices which fall in the available range will be used.
- 2) If the number of source devices exceeds the available space at the destination location, then only the available destination devices will be written to.
- 3) The BMOV instruction has a built in automatic feature to prevent overwriting errors from occurring when the source (S - n) and destination (D -n) data ranges coincide. This is clearly identified.

**Program example**



**Note:**

When the special variable is M8024=ON, the transmission direction is opposite.

**4.2.12 BON instruction**

**Instruction Description**

Table 4-29

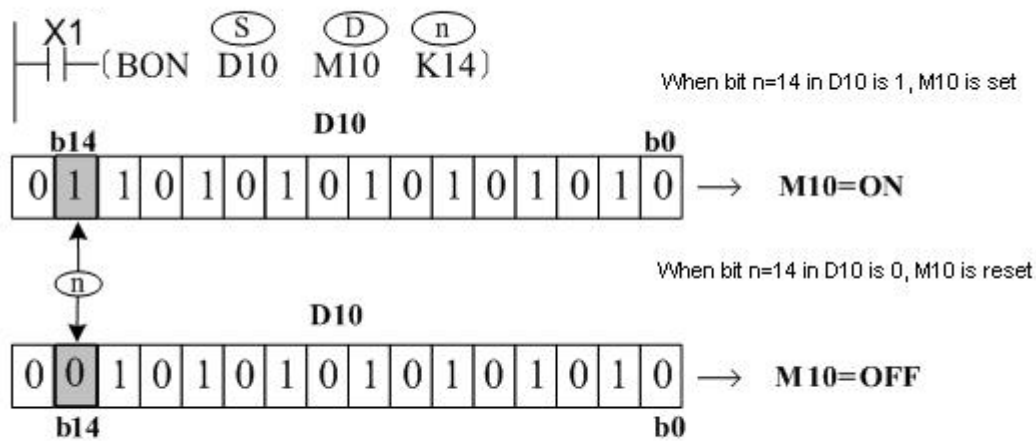
Name	Function	Devices			Format	Steps
		S	D	n		
BON (Check specified bit status)	The status of the specified bit in the source device is indicated at the destination	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	Y, M, S	K,H, Note: 16 bit operation n=0 to 15 32 bit operation n=0 to 31		BON, BONP: 7steps DBON P, DBON: 13 steps

**Operation**

- 1) A single bit position (n) is specified from within a source device/area (S). n could be regarded as a specified offset from the source head address (S), i.e. 0 (zero) being the first device (a 0 offset) where as an offset of 15 would actually be the 16th device.
- 2) If the identified bit becomes active, i.e. ON, the destination device (D) is activated to “flag” the new status. The destination device could be said to act as a mirror to the status of the selected bit source

**Program example:**

If D0=100 0000 0000(2) =1024. (b10=1) and X0=ON, then M1=1



**4.2.13 CALL instruction**

**Instruction description**

Table 4-30

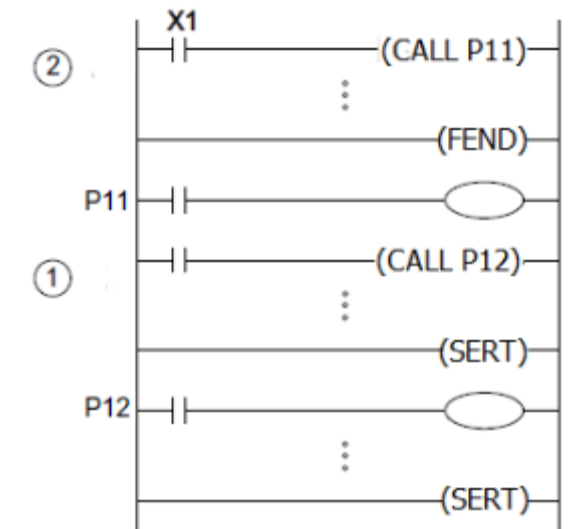
Name	Function	Devices	Format	Steps
		D		
CALL (Call subrou-tine)	Executes the subrou-tine program starting at the identified pointer position	Valid pointers from the range 0 to 62 Nest levels: 5 including the initial CALL		CALL, CALLP: 3 step Subrou-tine pointer P n: 1 steps

**Operation**

When the CALL instruction is active it forces the program to run the subroutine associated with the called pointer (area identified as subroutine P10). A CALL instruction must be used in conjunction with FEND and SRET instructions. The program jumps to the subroutine pointer (located after an FEND instruction) and processes the contents until an SRET instruction is encountered. This forces the program flow back to the line of ladder logic immediately following the original CALL instruction.

**Points to note**

- 1) Many CALL statements can reference a single subroutine.
- 2) Each subroutine must have a unique pointer number. Subroutine pointers can be selected from the range P0 to P127 (TP). Subroutine pointers and the pointers used for CJ instructions are NOT allowed to coincide.
- 3) Subroutines are not normally processed as they occur after an FEND instruction. When they are called, care should be taken not to overrun the watchdog timer setting.
- 4) Subroutines can be nested for 5 levels including the initial CALL instruction. As an example the program showed opposite shows a 2 level nest.
- 5) When X1 is activated the program calls subroutine P11. Within this subroutine is a CALL to a second subroutine P12. When both subroutines P11 and P12 are active simultaneously, they are said to be nested. Once subroutine P12 reaches its SRET instruction it returns the program control to the program step immediately following its original CALL (see ①). P11 then completes its operation, and once its SRET instruction is processed the program returns once again to the step following the CALL P11 statement (see ②).



**Program example**



**4.2.14 CCD instruction**

**Instruction description**

Table 4-31

Name	Function	Devices			Format	Steps
		S	D	n		
CCD (Check Code)	Checks the 'vertical' parity of a data stack	KnX, KnY, KnM, KnS, T, C, D	KnY, KnM, KnS, T, C, D	K, H, D Note: n = 1 to 256		CCD, CCD P: 7 steps

**Operation**

This instruction looks at a byte (8 bit) stack of data from head address (S) for n bytes and checks the vertical bit pattern for parity and sums the total data stack. These two pieces of data are then stored at the destination (D).

**Points to note**

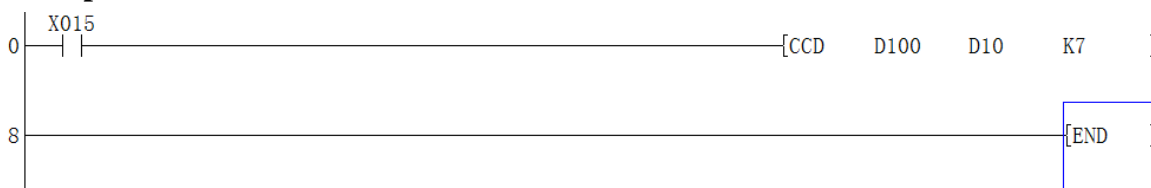
- 1) The SUM of the data stack is stored at destination D, while the Parity for the data stack is stored at D+1.
- 2) During the Parity check an even result is indicated by the use of a 0 (zero) while an odd parity is indicated by a 1 (one).
- 3) This instruction can be used with the 8 bit/ 16 bit mode flag M8161. The following results will occur under these circumstances.

M8161=OFF										
Source (S)		Bit pattern								
D100	H	FF	1	1	1	1	1	1	1	1
	L	FF	1	1	1	1	1	1	1	1
D101	H	FF	1	1	1	1	1	1	1	1
	L	00	0	0	0	0	0	0	0	0
D102	H	F0	1	1	1	1	0	0	0	0
	L	0F	0	0	0	0	1	1	1	1
Vertical parity D1			0	0	0	0	0	0	0	0
SUM D0		3FC								

M8161=ON										
Source (S)		Bit pattern								
D100	L	FF	1	1	1	1	1	1	1	1
D101	L	00	0	0	0	0	0	0	0	0
D102	L	0F	0	0	0	0	1	1	1	1
D103	L	F0	1	1	1	1	0	0	0	0
D104	L	F0	1	1	1	1	0	0	0	0
D105	L	0F	0	0	0	0	1	1	1	1
Vertical parity D1			1	1	1	1	1	1	1	1
SUM D0		2FD								

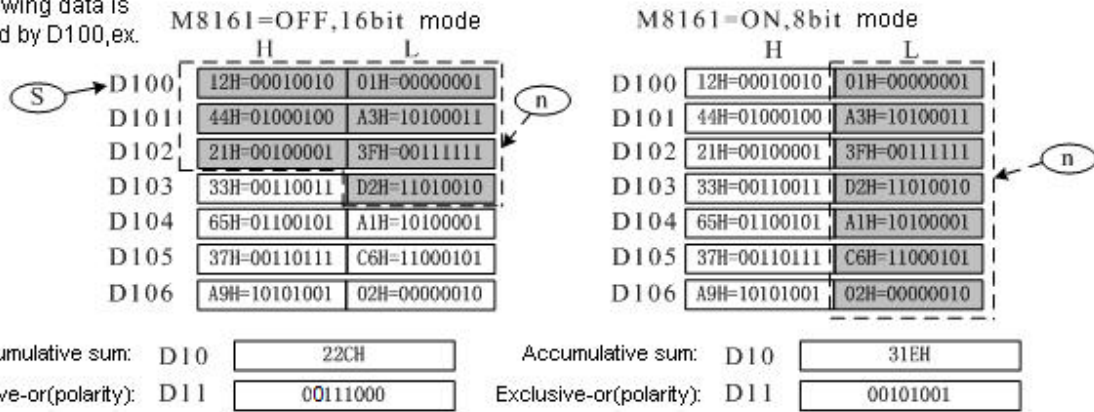
- 4) It should be noted that when M8161 is OFF 'n' represents the number of consecutive bytes checked by the CCD instruction. When M8161 is ON only the lower bytes of 'n' consecutive words are used.
- 5) The 'SUM' is quite simply a summation of the total quantity of data in the data stack. The Parity is checked vertically through the data stack as shown by the shaded areas.

**Program example**





the following data is activated by D100,ex.



RS/ HEX/ ASCI/ CCD instructions share the M8161 mode flag, which should be paid attention when programming.

### 4.2.15 CJ instruction

#### Instruction description

Table 4-32

Name	Function	Devices	Format	Steps
		D		
CJ (Conditional Jump)	Jumps to the identified pointer position	Valid pointers from the range 0 to 63(1S) Valid pointers from the range 0 to 127(2N/3V)		CJ, CJP: 3steps Jump pointer PPP: 1 step

#### Operation

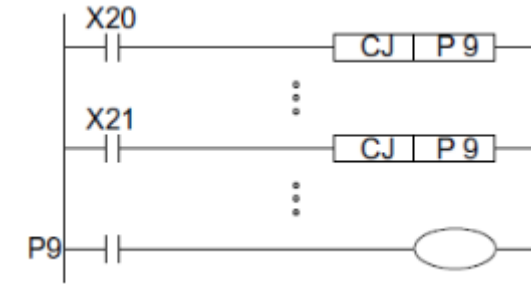
When the CJ instruction is active it forces the program to jump to an identified program marker. While the jump takes place the intervening pro-gram steps are skipped. This means they are not processed in any way. The resulting effect is to speed up the programs operational scan time.

#### Points to note

- 1) Many CJ statements can reference a single pointer.
- 2) Each pointer must have a unique number. Using pointer P63 is equivalent to jumping to the END instruction.
- 3) Any program area which is skipped, will not update output statuses even if the input devices change.

For example, the program opposite shows a situation which loads X1 to drive Y1. Assuming X1 is ON and the CJ instruction is activated the load X1, out Y1 is skipped. Now even if X1 is turned OFF Y1 will remain ON while the CJ instruction forces the program to skip to the pointer P0. The

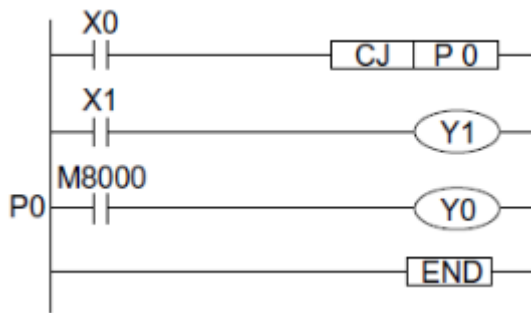
reverse situation will also apply, i.e. if X1 is OFF to begin with and the CJ instruction is driven, Y1 will not be turned ON if X1 is turned ON. Once the CJ instruction is deactivated X1 will drive Y1 in the normal manner. This situation applies to all types of outputs, e.g. SET, RST, OUT, Y, M & S devices etc.



- 4) The CJ instruction can jump to any point within the main program body or after an FEND Instruction
- 5) A CJ instruction can be used to Jump forwards through a program, i.e. towards the END instruction OR it can jump backwards towards step 0. If a backwards jump is used care must be taken not to overrun the watchdog timer setting otherwise the PLC will enter an error situation. For more information on the watchdog timer (WDT instruction)



- 6) Unconditional jumps can be entered by using special auxiliary coils such as M8000. In this situation while the PLC is in RUN the program will ALWAYS execute the CJ instruction in an unconditional manner.



**Important**

Timers and counters will freeze their current values if they are skipped by a CJ instruction. For example if Y1 in the previous program (see point c) was replaced by T0 K100 and the CJ instruction was driven, the contents of T0 would not change/increase until the CJ instruction is no longer driven, i.e. the current timer value would freeze. High speed counters are the only exception to this situation as they are processed independently of the main program.

- 1) Using applied instructions:

- Applied instructions are also skipped if they are programmed between the CJ instruction and the destination pointer. However, The PLSY and PWM instructions will operate continuously if they were active before the CJ instruction was driven, otherwise they will be processed, i.e. skipped, as standard applied instructions.
- 2) Details of using CJ with other program flow instructions
- Further details can be found about the combined use of different program flow techniques (such as master control, MC etc.).

**Program example:**





In the above example: If X0=ON and jump instruction is implemented, the coil operations in skipped instructions are listed as follows:

- 1) Y, M, S keeps the previous state, after CJ is implemented.
- 2) If T is not activated before jumping, the timer will not operate even it is activated after jumping. If T is activated, it will keep state but output coil cannot be activated. When X0 port is power OFF, the X0 contact of project will set off immediately.
- 3) If C is not activated before jumping, the counter will not operate even if it is activated after jumping.
- 4) After jumping, the function instruction will not operate.
- 5) If the reset instruction of the timer and counter is out of the jump, the timer coil and counter coil reset is effective.

### 4.2.16 CML instruction

#### Instruction description:

Table 4-33

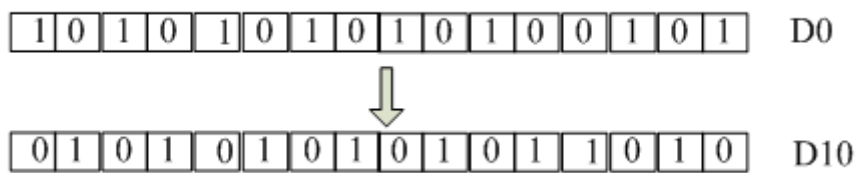
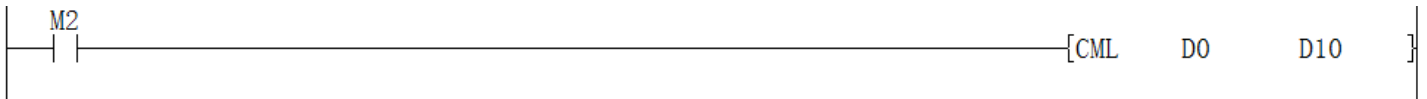
Name	Function	Devices		Format	Steps
		S	D		
CML (Complement)	Copies and inverts the source bit pattern to a specified destination	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z		CML,C MLP: 5 steps DCML, DCML P: 9 steps

#### Operation

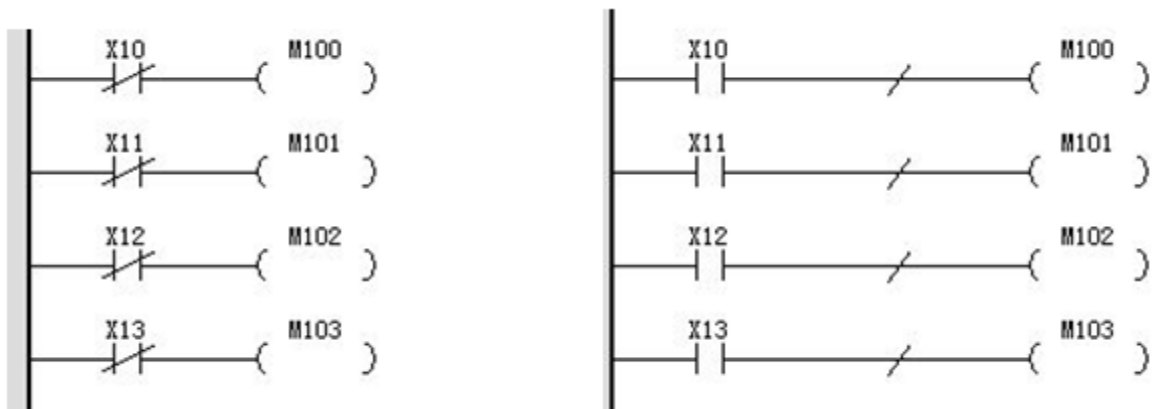
- 1) A copy of each data bit within the source device (S) is inverted and then moved to a designated destination (D).
- 2) This means each occurrence of a '1' in the source data will become a '0' in the destination data while each source digit which is '0' will become a '1'. If the destination area is smaller than the source data then only the directly mapping bit devices will be processed.

**Program example**

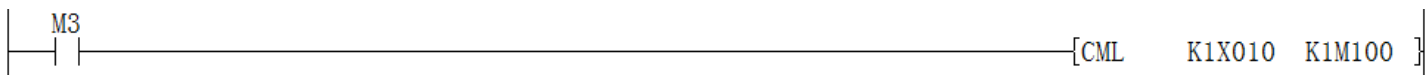
- 1) Example 1:



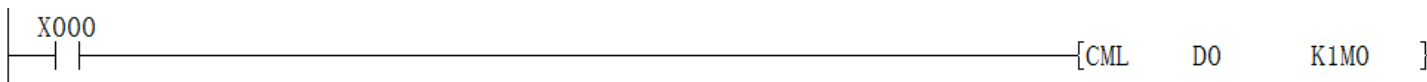
- 2) Example 2:

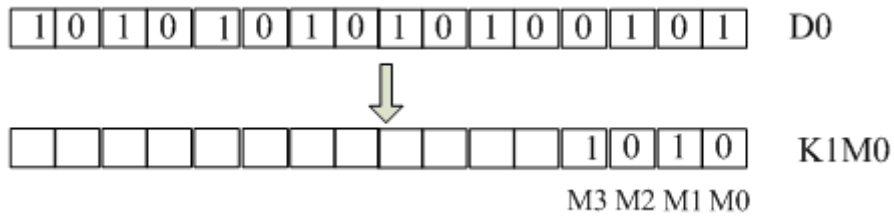


The above two programs can be replaced with the CML instruction below.



- 3) Example 3:





### 4.2.17 CMP instruction

#### Instruction description

Table 4-34

Name	Function	Devices			Format	Steps
		S1	S2	D		
CMP (Compare)	Compares two data values - results of <, = and > are given.	K, H, KnX, KnM, KnS, T, C, D, V, Z	KnY, KnS	Y, M, S Note: 3 consecutive devices are used.		CMP, CMPP: 7 steps DCMP, DCMP P: 13 steps

#### Operation

The data of S1 is compared to the data of S2. The result is indicated by 3 bit devices specified from the head address entered as D. The bit devices indicate:

S2 is less than S1 - bit device D is ON

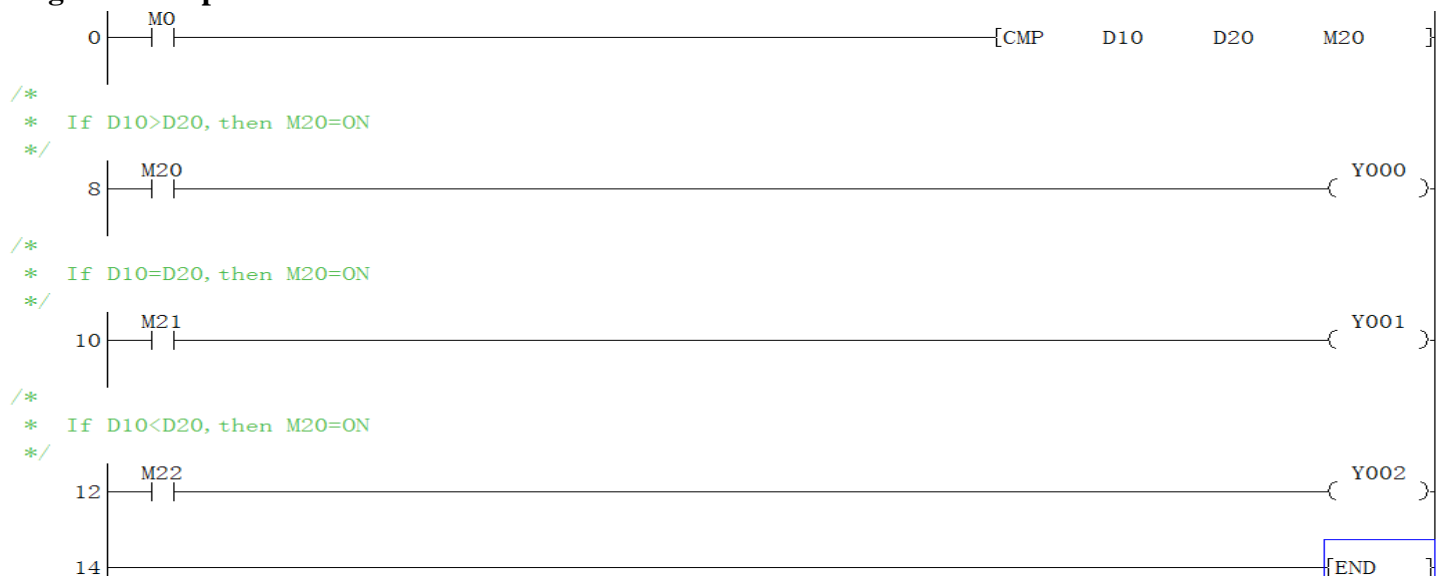
S2 is equal to S1 - bit device D+1 is ON

S2 is greater than S1 - bit device D+2 is ON

#### Note

The destination (D) device statuses will be kept even if the CMP instruction is deactivated. Full algebraic comparisons are used, i.e. -10 is smaller than +2 etc.

#### Program example



### 4.2.18 DABS instruction

#### Instruction description

Table 4-35

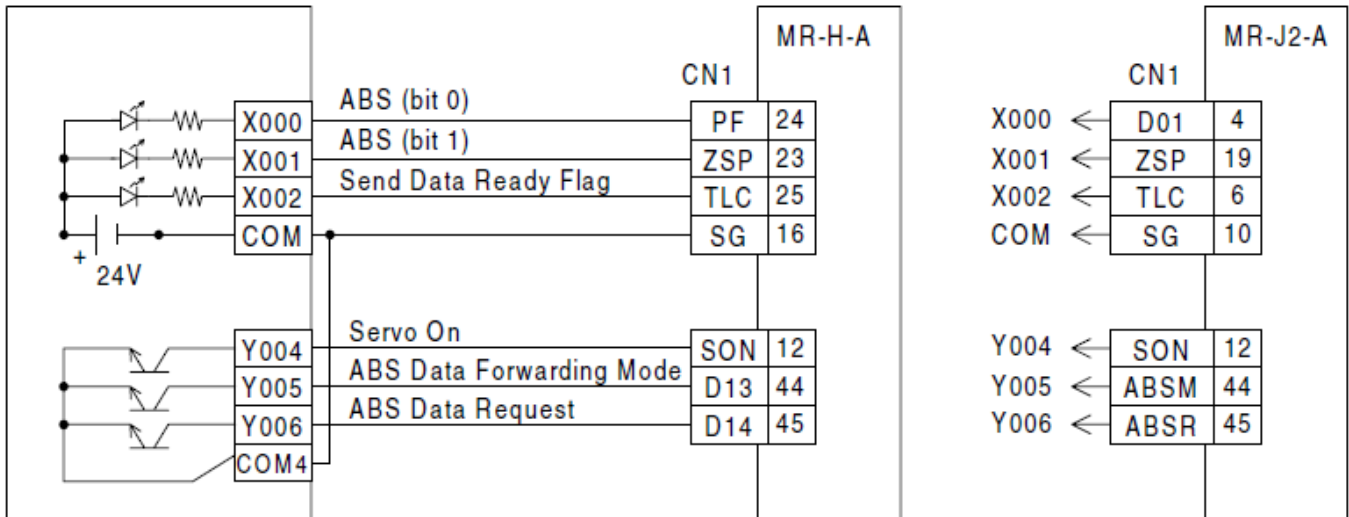
Name	Function	Devices			Format	Steps
		S	D1	D2		
ABS	Reads the absolute position from a servo motor	X,Y,M, S	Y,M,S	T,C,D, V,Z		DAB S 13 steps

#### Operation

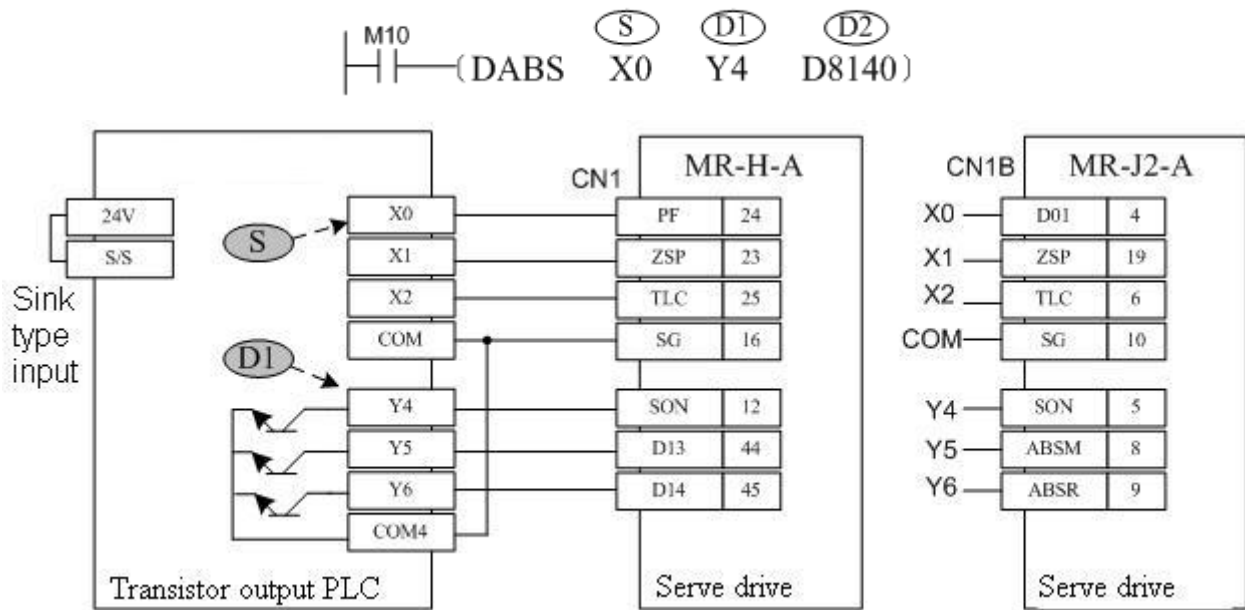
This instruction reads the absolute position data when a Mitsubishi servo motor, MR-H or MR-J2, equipped with absolute positioning function is connected. [S] is the first of three inputs used for communication flags (see drawing below), [D1] is the first of three communication outputs and [D2] is the data destination register

#### Points to note

- 1) This instruction is 32-Bit. Be sure to input as “DABS”
- 2) Read starts when the instruction drive contact turns ON. When the read is complete, the execution complete flag M8029 is energized. If the instruction drive contact is turned OFF during read, read is aborted.
- 3) When designing a system, set the servo amplifier to be ON earlier than the power of the PLC, or so that they are both powered ON at the same time.
- 4) The device [D2] to which the absolute value is read, can be set within a word device range. However, the absolute value should be transferred at some point to the correct registers (D8141 & D8140)
- 5) The DABS instruction drive contact uses an input which is always ON, even after the absolute value is read. If the instruction drive contact turns OFF after the read is complete, the servo ON (SON) signal will turn OFF and the operation disabled.
- 6) Even if the servo motor is equipped with an absolute position detection function, it is good practice to execute a zero return operation during initial system set up.

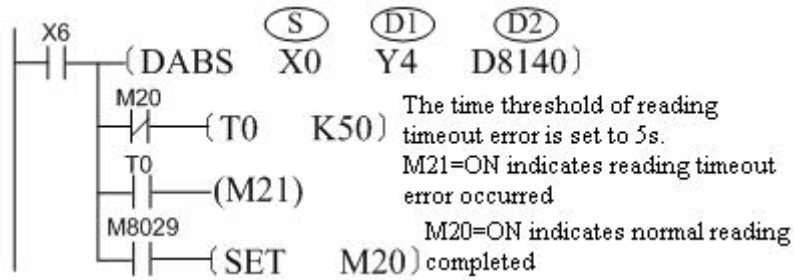


Program example

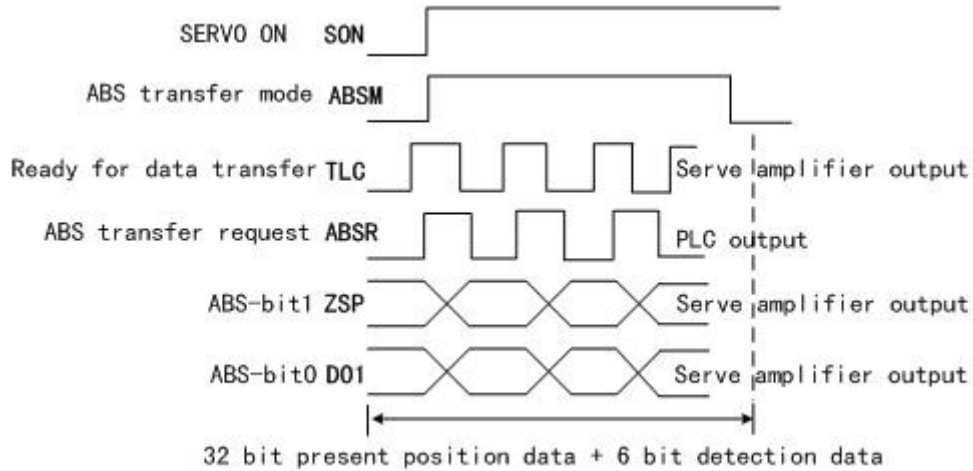


- 1) When M10 is set to ON, it begins to read. When DABS instruction is completed, the M8029 flag is set to ON;
- 2) When the instruction is running in process and the driver flag is set to OFF, the read operation will be interrupted;
- 3) The programming example for reading ABS data is as follows: when the X6 terminal is closed, it begins to read. If it is not completed in 5s, the timeout flag M21 will be set. The demo is as following:





The signal time sequence of the ABS read operation is shown in the following figure. When implementing an instruction, the PLC will automatically implement the access operation with servo driver.



### 4.2.19 DCOS instruction

#### Instruction description

Table 4-36

Name	Function	Devices		Format	Steps
		S	D		
COS (Cosine)	Calculates the cosine of a floating point value	D - must be in floating point number format (32 bits).	D - a floating point value (32 bits).		DCOS ,DCO SP: 9 steps

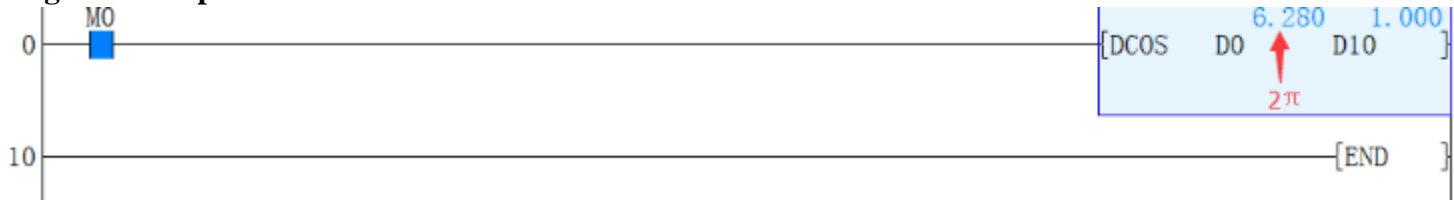
#### Operation

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

#### Points to note

All the points for the SIN instruction apply, except that COS is calculated.

Program example



4.2.20 DEADD instruction

Instruction Description

Table 4-37

Name	Function	Devices			Format	Steps
		S1	S2	D		
EADD (Floating Point Addition)	Adds two floating point numbers together	K, H - integer value automatically converted to floating point D - must be in floating point format (32 bits).		D - a floating point value (32 bits).		DEAD D, DEAD DP: 13 steps

Operation

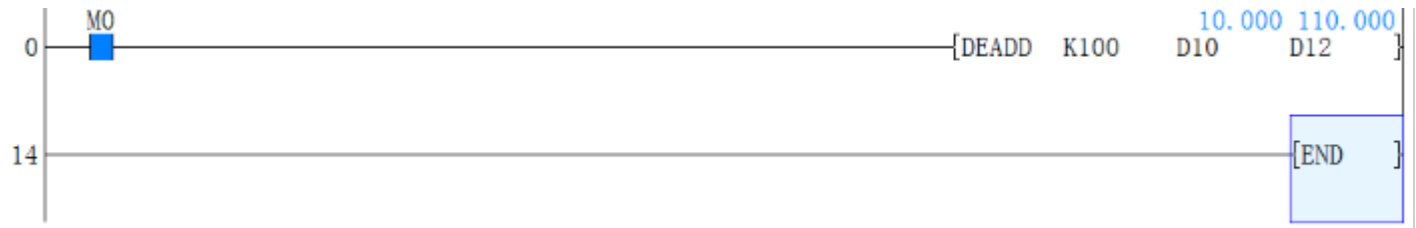
The floating point values stored in the source devices S1 and S2 are algebraically added and the result stored in the destination device D.

Points to note

- 1) The instruction must use the double word format; i.e., **DEADD** or **DEADDP**. All source data and destination data will be double word; i.e. uses two consecutive data registers to store the data (32 bits). Except for K or H, all source data will be regarded as being in floating point format and the result stored in the destination will also be in floating point format.
- 2) If a constant K or H is used as source data, the value is converted to floating point before the addition operation.
- 3) The addition is mathematically correct: i.e.  $2.3456 \times 10^2 + (-5.6 \times 10^{-1}) = 2.34 \times 10^2$
- 4) The same device may be used as a source and as the destination. If this is the case then, on continuous operation of the DEADD instruction, the result of the previous operation will be used as a new source value and a new result calculated. This will happen every program scan unless the pulse modifier or an interlock program is used.
- 5) If the result of the calculation is zero “0” then the zero flag, M8020 is set ON.

**Program example**

1) Example 1



2) Example 2



**4.2.21 DEBCD instruction**

**Instruction Description**

Table 4-38

Name	Function	Devices		Format	Steps
		S	D		
EBCD (Float to Scientific conversion)	Converts floating point number format to scientific number format	D - must be in floating point format (32 bits).	D - 2 consecutive devices are used D - mantissa D+1 - exponent.		DEB CD, DEB CDP: 9 steps

**Operation**

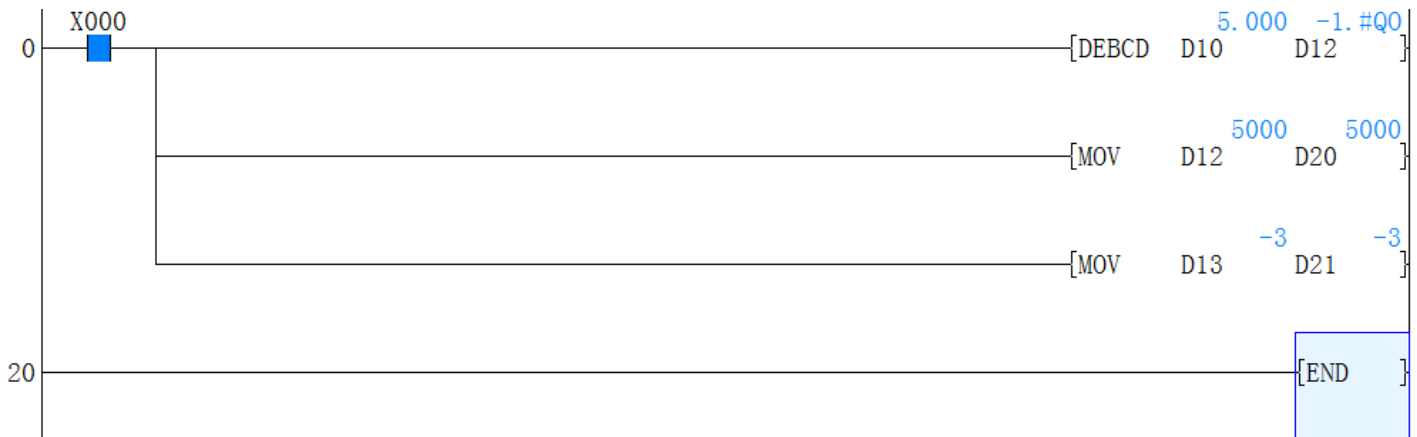
It converts a floating point value at S into separate mantissa and exponent parts at D and D+1 (scientific format).

**Points to note**

- 1) The instruction must be double word format. The destinations D and D+1 represent the mantissa and exponent of the floating point number respectively.
- 2) To provide maximum accuracy in the conversion the mantissa D will be in the range 1000 to 9999 (or 0) and the exponent D+1 corrected to an appropriate value.

3) E.g.  $S = 3.4567 \times 10^{-5}$  will become  $D = 3456, D_{+1} = -8$

**Program example**



**4.2.22 DEBIN instruction**

**Instruction description**

Table 4-39

Name	Function	Devices		Format	Steps
		S	D		
EBIN (Scientific to Float conversion)	Converts scientific number format to floating point number format	D - 2 consecutive devices are used S- mantissa S+1 - exponent.	D - a floating point value (32 bits).		DEBIN, DEBIN P: 9 steps

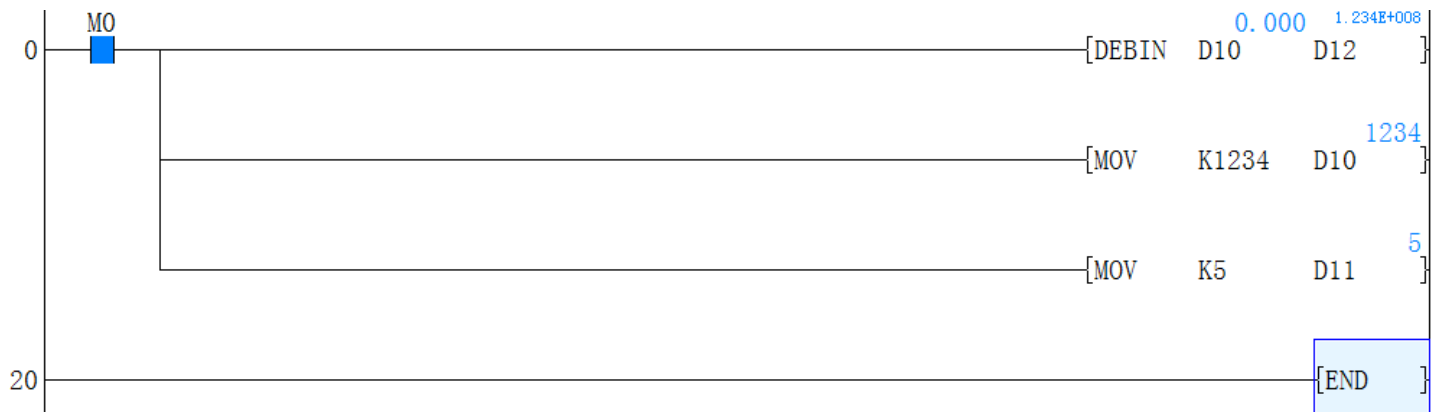
**Operation**

It generates a floating point number at D from scientific format data at source S.

**Points to note**

- 1) The instruction must be double word format. The source data Sand S+1 represent the mantissa and exponent of the floating point number to be generated.
- 2) To provide maximum accuracy in the conversion the mantissa S must be in the range 1000 to 9999 (or 0) and the exponent S+1 corrected to an appropriate value.
- 3) E.g. S= 5432, S+1= 12 will become  $D = 5.432 \times 10^9$

**Program example**



**4.2.23 DEC instruction**

**Instruction Description**

Table 4-40

Name	Function	Devices	Format	Steps
		D		
DEC (Decrement)	The designated device is decremented by 1 on every execution of the instruction	KnY, KnM, KnS, T, C, D, V, Z Standard V,Z rules apply for 32 bit operation		DEC,DE CP: 3 steps DDEC, DDECP: 5 steps

**Operation**

- 1) On every execution of the instruction the device specified as the destination D has its current value decremented (decreased) by a value of 1.
- 2) In 16 bit operation, when -32,768 is reached the next increment will write a value of +32,767 to the destination device.
- 3) In 32 bit operation, when -2,147,483,648 is reached the next increment will write a value of +2,147,483,647 to the destination device.
- 4) In both cases there is no additional flag to identify this change in the counted value.

**Program example**



### 4.2.24 DECMP instruction

#### Instruction description

Table 4-41

Name	Function	Devices			Format	Steps
		S1	S2	D		
ECMP (Floating Point Compare)	Compares two floating point values - results of <, = and > are given	K, H - integer value automatically converted to floating point D - must be in floating point format (32bits).	Y, M, S Note:3 consecutive devices are used.		DECMP , DECMP P: 13 steps	

#### Operation

- 1) The data of S1 is compared to the data of S2. The result is indicated by 3 bit devices specified with the head address entered as D. The bit devices indicate:
- 2) S2 is less than < S1 - bit device D is ON
- 3) S2 is equal to = S1 - bit device D+1 is ON
- 4) S2 is greater than > S1 - bit device D+2 is ON

#### Points to note

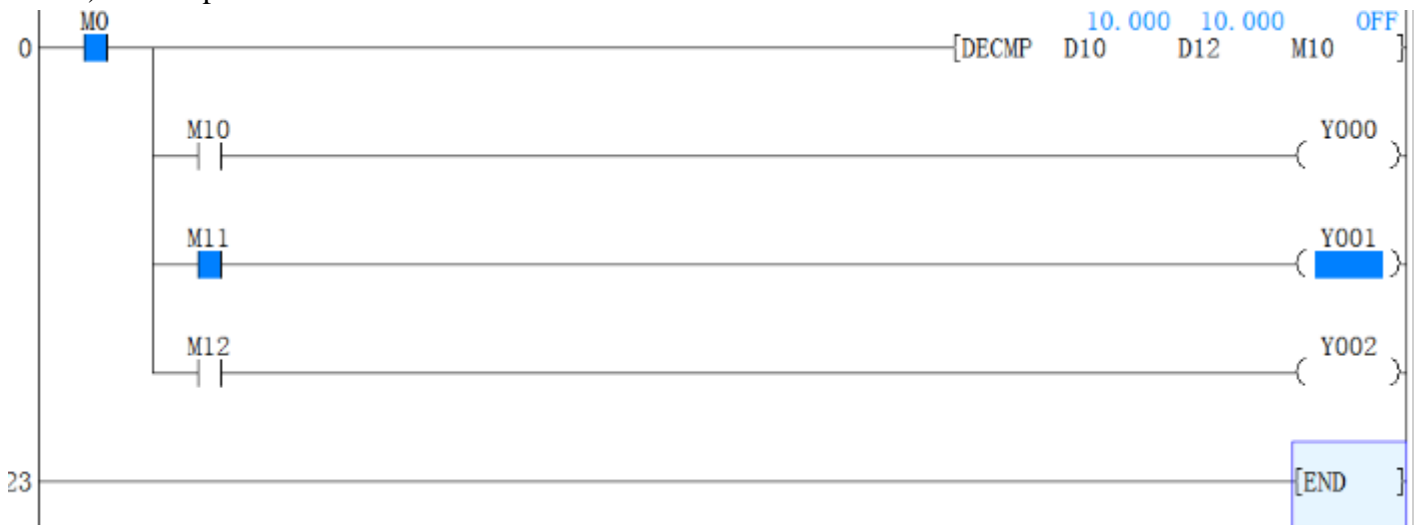
The status of the destination devices will be kept even if the ECMP instruction is deactivated.

#### Program example

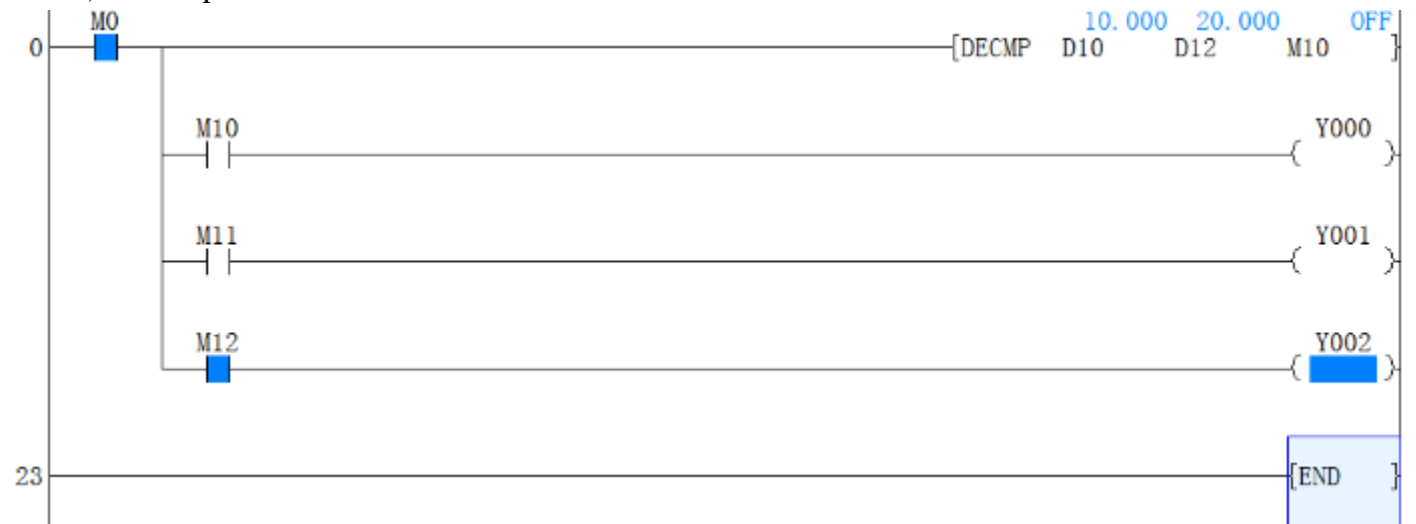
- 1) Example 1



2) Example 2



3) Example 3



4.2.25 DECO instruction

Instruction description

Table 4-42

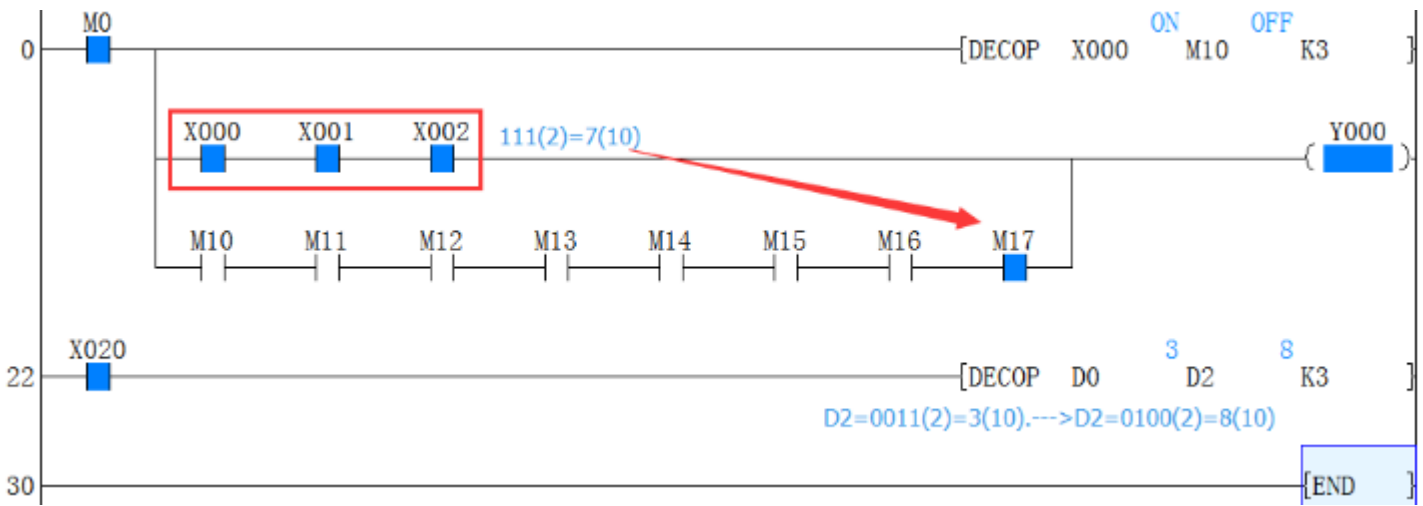
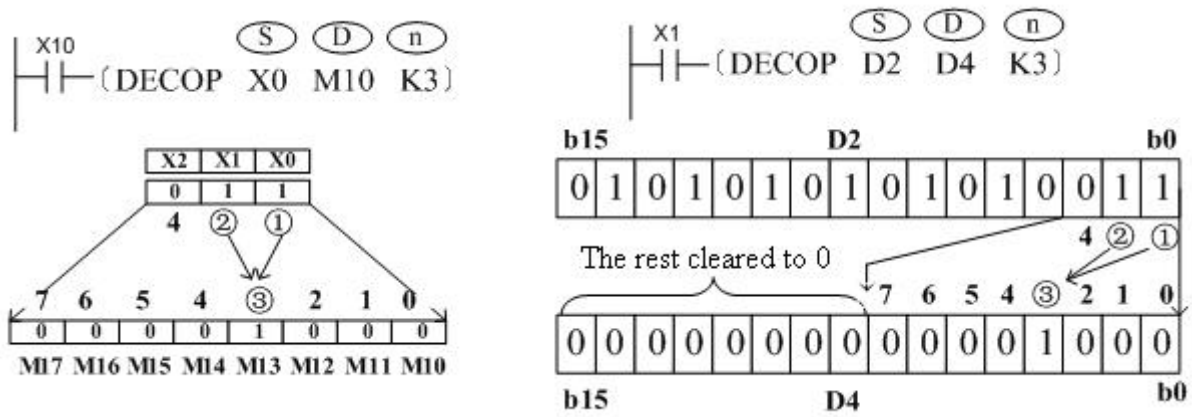
Name	Function	Devices			Format	Steps
		S	D	n		
DECO (Decode)	Source data value Q identifies the bit of the destination device which will be	K, H, X, Y, M, S, T, C, D, V, Z	Y, M, S, T, C, D	K, H, Note: D= Y,M,S then n range =1-8 D= T,C,D then		DEC O, DEC OP: 7 steps

	turned ON			n range = 1-4 n= 0, then no processing	
--	-----------	--	--	---	--

**Operation**

Source data is provided by a combination of operands S and n. Where S specifies the head address of the data and n, the number of consecutive bits. The source data is read as a single number (binary to decimal conversion) Q. The source number Q is the location of a bit within the destination device (D) which will be turned ON (see example opposite). When the destination device is a data device n must be within the range 1 to 4 as there are only 16 available destination bits in a single data word. All unused data.

**Program example**



**4.2.26 DEDIV instruction**

**Instruction description**



Table 4-43

Name	Function	Devices			Format	Steps
		S1	S2	D		
EDIV (Floating Point Division)	Divides one floating point number by another	K, H - integer value automatically converted to floating point D - must be in floating point format (32 bits)..		D - a floating point value (32 bits).		DEDIV ,DEDI VP: 13 steps

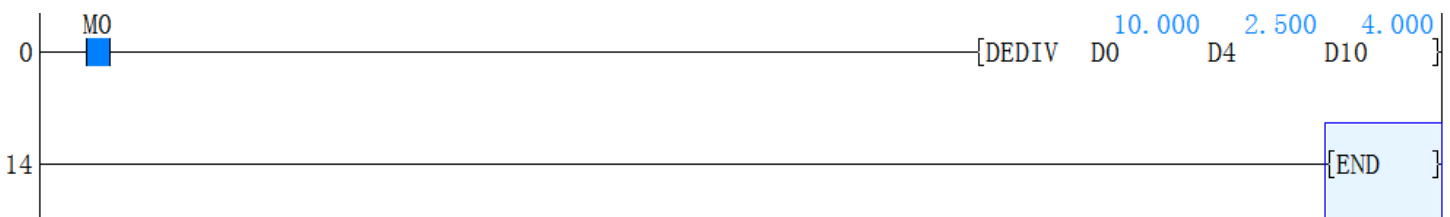
**Operation**

The floating point value of S1 is divided by the floating point value of S2. The result of the division is stored in D as a floating point value. No

**Points to note:**

Points a, b, c, d of the EADD instruction applies, except that a division is performed. If S2 is 0 (zero) then a divide by zero error occurs and the operation fails. M8067, M8068 will be set to ON. If the calculation result is 0, the 0 flag bit (M8020) will be reset.

**Program example:**



**4.2.27 DEMUL instruction**

**Instruction description**

Table 4-44

Name	Function	Devices			Format	Steps
		S 1	S2	n		
EMUL (Floating Point)	Multiplies two floating point	K, H - integer value automatically converted to		D - a floating point value		DEMUL ,DEMU LP:13 steps

Multipl ication)	numbers together	floating point D - must be in floating point format (32 bits).	(32 bits).		
---------------------	---------------------	--	---------------	--	--

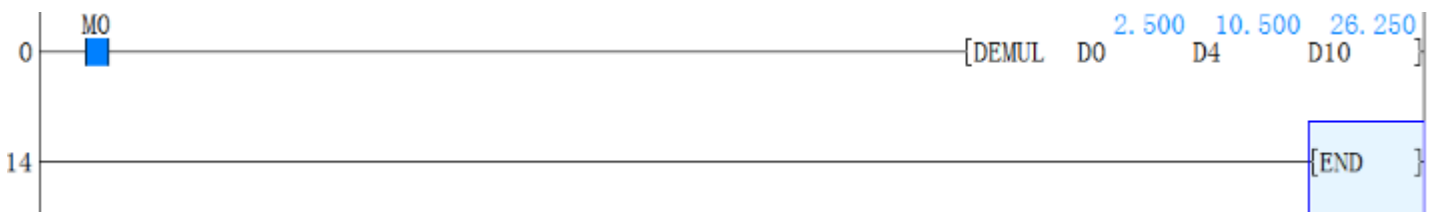
**Operation**

The floating point value of S<sub>1</sub> is multiplied with the floating point value of S<sub>2</sub>. The result of the multiplication is stored at D as a floating point value.

**Points to note**

Point a, b, c and d of the EADD instruction apply, except that a multiplication is performed. If the calculation result is 0, the 0 flag bit (M8020) will be reset.

**Program Example**



**4.2.28 DESQR instruction**

**Instruction description**

Table 4-45

Name	Function	Devices		Format	Steps
		S	D		
ESQR (Floating Point Square Root)	Calculates the square root of a floating point value.	K, H - integer value automatically converted to floating point D - must be in floating point number format (32 bits).	D - a floating point value (32 bits).		DESQL , DESQL P: 9 steps

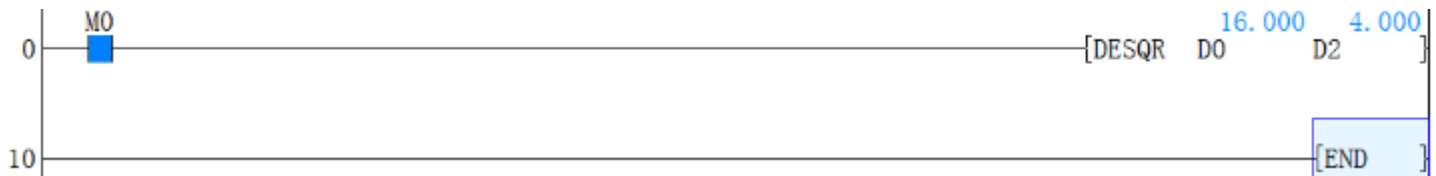
**Operation**

A square root is performed on the floating point value of Sand the result is stored in D.

**Points to note**

Points a, b, c, d of the EADD instruction applies, except that a division is performed. If S2 is 0 (zero) then a divide by zero error occurs and the operation fails. **M8067, M8068 will be set to ON. If the calculation result is 0, the 0 flag bit (M8020) will be reset.**

**Program Example:**



**4.2.29 DESUB instruction**

**Instruction description**

Table 4-46

Name	Function	Devices			Format	Steps
		S1	S2	D		
ESUB (Floating Point Sub-raction)	Subtracts one floating point number from another	K, H - integer value automatically converted to floating point number format (32 bits).	D - a floating point value (32 bits).		DESUB , DESUB P: 13 steps	

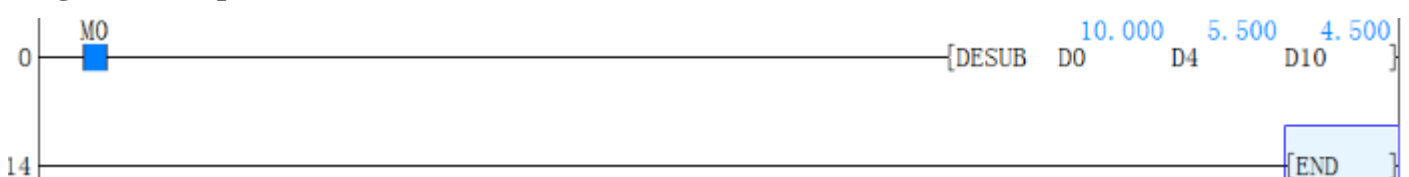
**Operation**

The floating point value of S2 is subtracted from the floating point value of S1 and the result stored in.

**Points to note**

All points of the EADD instruction apply, except that a subtraction is performed. If the calculation result is 0, the 0 flag bit (M8020) will be reset.

**Program Example**



### 4.2.30 DEZCP instruction

#### Instruction description

Table 4-47

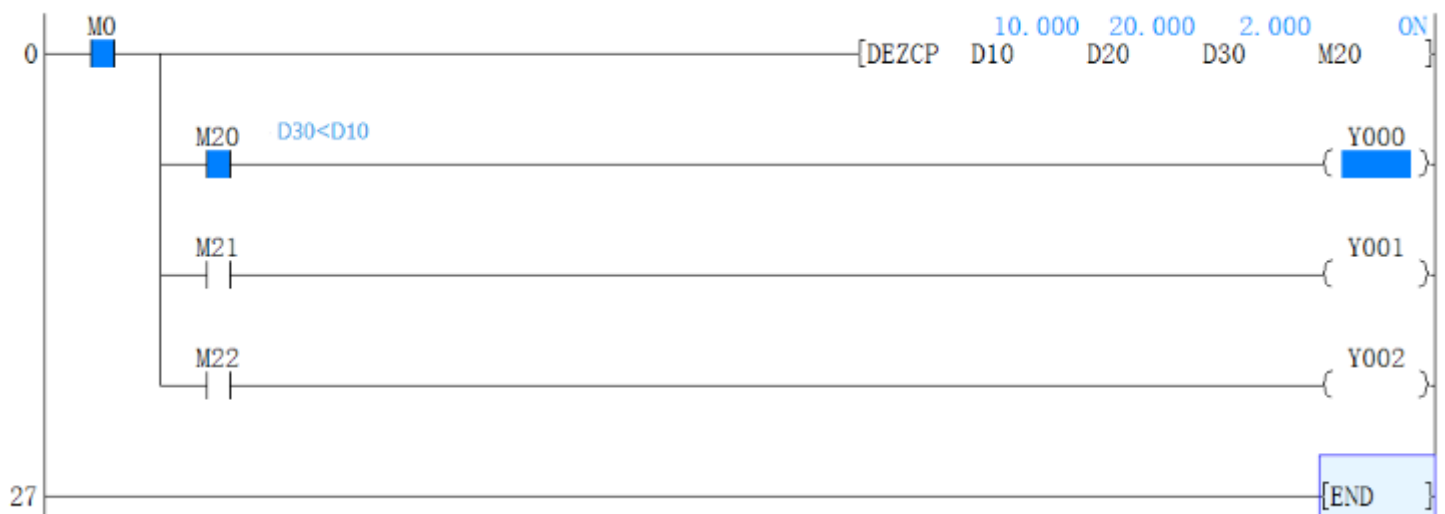
Name	Function	Devices				Format	Steps
		S 1	S2	S3	n		
EZCP ( Floating Point Zone Comparison)	Compares a float range with a float value - results of <, = and > are given	K, H - integer value automatically converted to floating point format (32 bits). <b>Note:</b> S1 must be less than S2.			Y, M, S Note: 3 consecutive devices are used.		DEZCP , DEZCP P: 13 steps

#### Operation

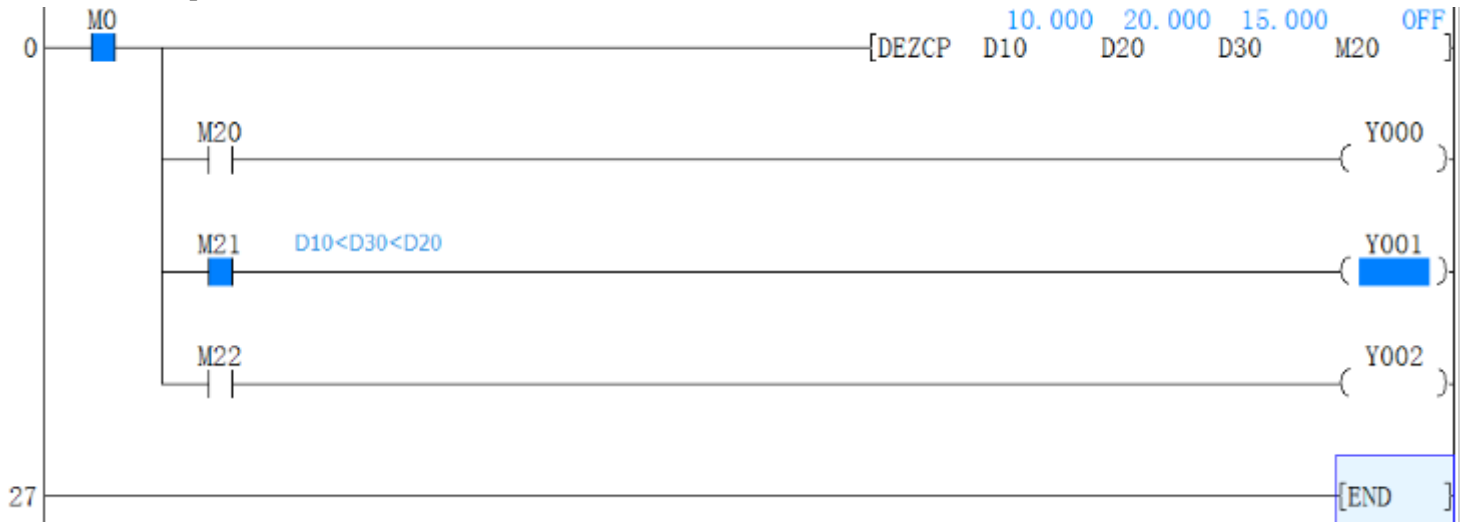
- 1) The operation is the same as the ECMP instruction except that a single data value (S3) is compared to a data range (S1 - S2).
- 2) S3 is less than S1 and S2 - bit device D is ON
- 3) S3 is between S1 and S2 - bit device D+1 is ON
- 4) S3 is greater than S2 - bit device D+2 is ON

#### Program Example

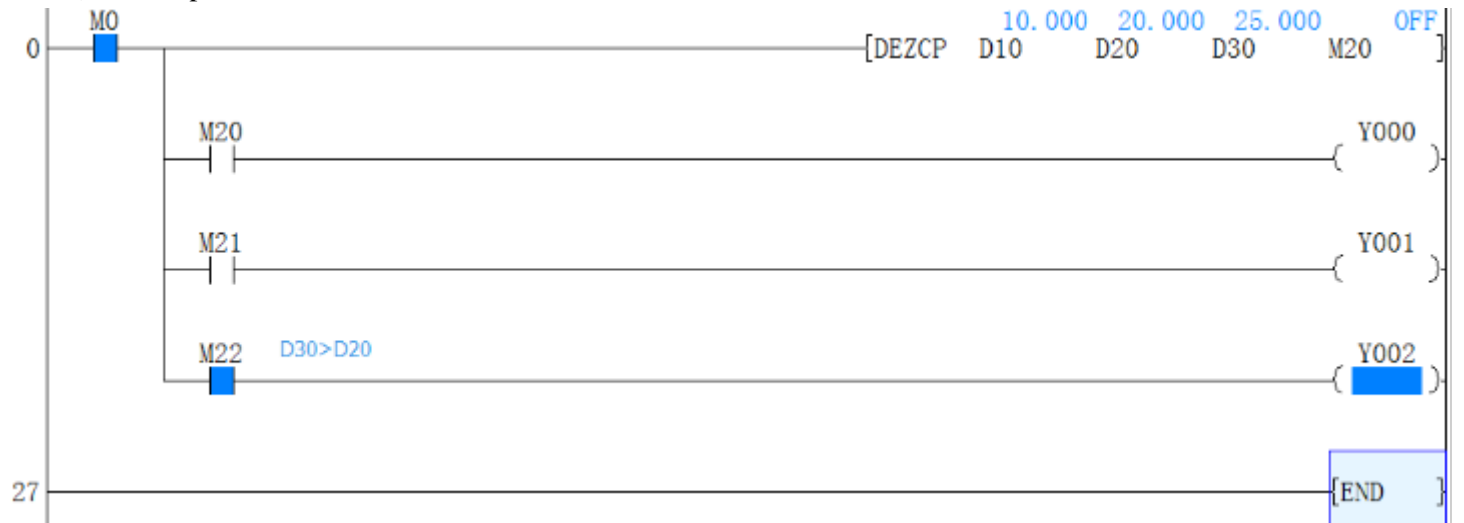
- 1) Example 1



2) Example 2



3) Example 3



4.2.31 DHSCR instruction

Instruction description

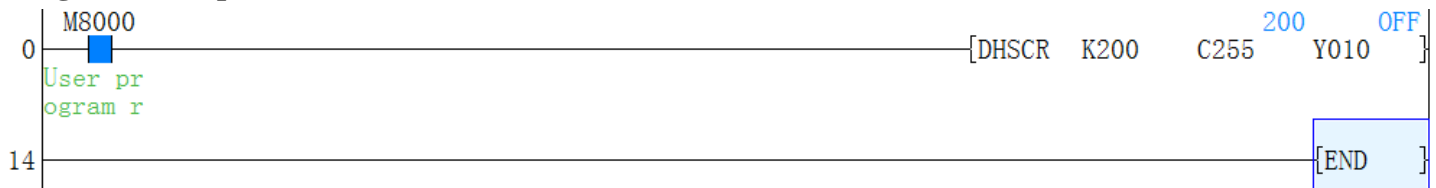
Table 4-48

Name	Function	Devices			Format	Steps
		S 1	S2	n		
HSCR (High speed counter reset)	Resets the selected output when the specified high speed counter equals the test value	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	C Note: C = C235 to C255, or available high speed counters	Y, M, S C Note: If C, use same counter as S2		DHSC R: 13 steps

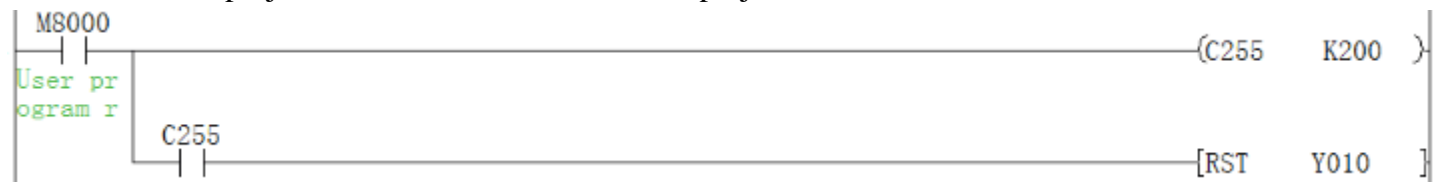
**Operation**

- 1) The HSCR, compares the current value of the selected high speed counter (S<sub>2</sub>) against a selected value (S<sub>1</sub>). When the counters current value changes to a value equal to S<sub>1</sub>, the device specified as the destination (D) is reset. In the example above, Y10 would be reset only when C255's value stepped from 199 to 200 or from 201 to 200. If the current value of C255 was forced to equal 200 by test techniques, output Y10 would **NOT** reset.
- 2) For further, general points, about using high speed counter functions, please see the subsection 'Points to note' under the HSCS. Relevant points are; a, b, and c.

**Program Example**



The above project is the same function with the project below,



**4.2.32 DHSCS instruction**

**Instruction description**

Table 4-49

Name	Function	Devices			Format	Steps
		S 1	S2	n		
HSCS (High speed counter set)	Sets the selected output when the specified high speed counter value equals the test value	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	C Note: C = 235 to 254, or available high speed counters	Y, M, S Interrupt pointers I010 to I060 can be set on TP		DHSCS : 13 steps

**Operation**

The HSCS set, compares the current value of the selected high speed counter (S<sub>2</sub>) against a selected value (S<sub>1</sub>). When the counters current value changes to a value equal to S<sub>1</sub> the device specified as the

destination (D) is set ON. The example above shows that Y10 would be set ON only when C255's value stepped from 99-100 OR 101-100. If the counters current value was forced to equal 100, output Y10 would NOT be set ON.

**Points to note**

- 1) It is recommended that the drive input used for the high speed counter functions; HSCS, HSCR, HSCZ is the special auxiliary RUN contact M8000.
- 2) If more than one high speed counter function is used for a single counter the selected flag devices (D) should be kept within 1 group of 8 devices, i.e. Y0-7, M10-17.
- 3) All high speed counter functions use an interrupt process, hence, all destination devices (D) are updated immediately.

**Use of interrupt pointers**

- 1) TP can use interrupt pointers I010 through I060 (6 points) as destination devices (D). This enables interrupt routines to be triggered directly when the value of the specified high speed counter reaches the value in the HSCS instruction.
- 2) When (D) is between I010~I060, the subprogram for interrupting 0~5 in the high-speed counter needs to be initiated. It is certain that the corresponding interrupting subprogram, the initiation of relevant interrupting permissible signal, and the overall interrupting permissible signal must be properly programmed in order to intercept the counter when necessary. M8059 that is positioned as ON prohibits all intercepting procedures over high-speed counters.

Table 4-50

Operand	Interruption Prohibiting Instruction
I010	M8059
I020	
I030	
I040	
I050	
I060	

**Program Example**

- 1) Example 1



The above project is the same with the project below.



2) Example 2



4.2.33 DHSZ instruction

Instruction description

Table 4-51

Name	Function	Devices				Format	Steps
		S 1	S2	S3	n		
HSCS (High speed counter set)	The current value of a high speed counter is checked against a specified range	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		C Note :C = 235 to 255,	Y, M, S Note: 3 consecutive devices are used		DHSZ: 17 steps

Operation

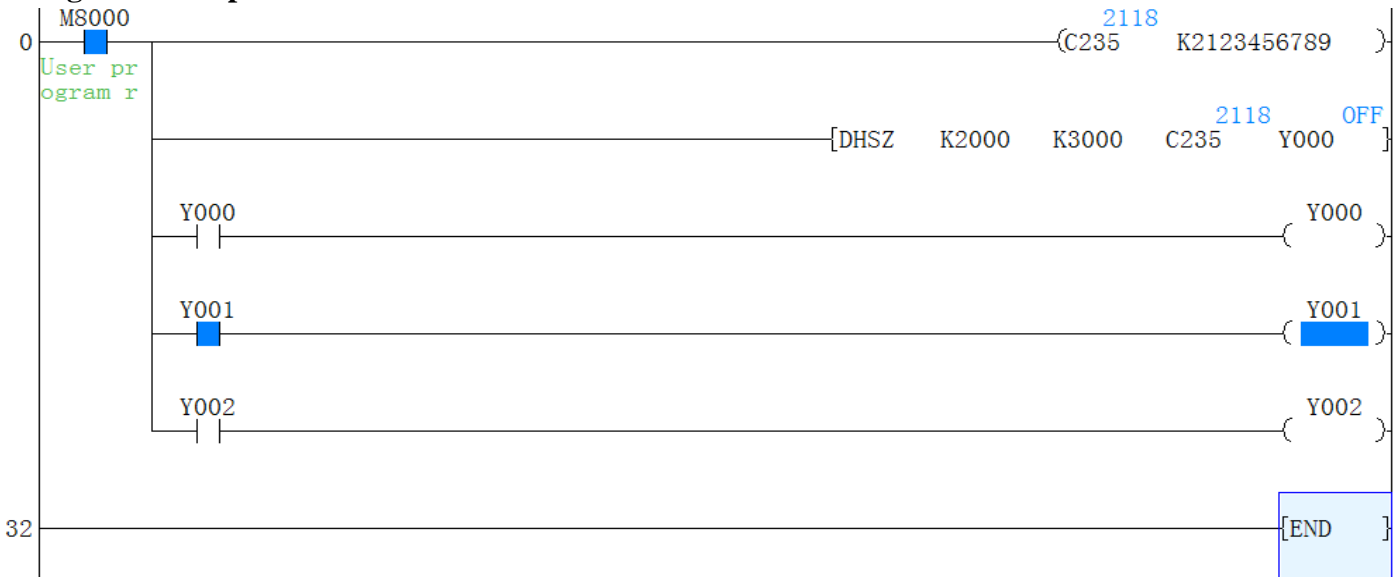
- 1) This instruction works in exactly the same way as the standard ZCP (FNC11). The only difference



is

- 2) That the device being compared is a high speed counter (specified as S3). Also, all of the outputs (D) are updated immediately due to the interrupt operation of the DHSZ. It should be remembered that when a device is specified in operand D it is in fact a head address for 3 consecutive devices. Each one is used to represent the status of the current comparison, i.e. using the above example as a basis,
- 3) For further, general points, about using high speed counter functions please see the subsection 'Points to note' under the HSCS. Relevant points are; a, b, and c. Please also reference the note about the number of high speed instructions allowable.

**Program Example**



- 1) Y0 (D) C235 is less than S1, K12000 ( $C235 < 2000$ )
- 2) Y1 (D+1) C235 is greater than S1, K2000 but less than S2, K3000 ( $2000 < C235 < 3000$ )
- 3) Y2 (D+2) C235 is greater than S2, K3000 ( $C235 > 3000$ )

**4.2.34 DIV instruction**

**Instruction Description**

Table 4-52

Name	Function	Devices			Format	Steps
		S 1	S2	n		
DIV (Division)	Divides one source value by another the result is stored in the destination device	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		KnY, KnM, KnS, T, C, D		DIV, DIV P: 7steps DDIV, DDIVP: 13 steps

**Operation**

The primary source (S<sub>1</sub>) is divided by the secondary source (S<sub>2</sub>). The result is stored in the destination (D). Note the normal rules of algebra apply.

**Points to note**

- 1) When operating the DIV instruction in 16bit mode, two 16 bit data sources are divided into each other. They produce two 16 bit results. The device identified as the destination address is the lower of the two devices used to store these results.

This storage device will actually contain a record of the number of whole times S<sub>2</sub> will divide into S<sub>1</sub> (the quotient).

The second, following destination register contains the remained left after the last whole division (the remainder). Using the previous example with some test data:

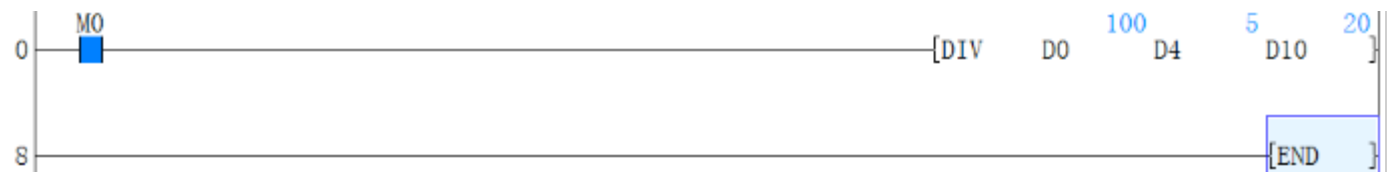
$$51(D0) \div 10(D2) = 5(D4) 1(D5)$$

This result is interpreted as 5 whole divisions with 1 left over

$$(5 \times 10 + 1 = 51).$$

- 2) When operating the DIV instruction in 32 bit mode, two 32 bit data sources are divided into each other. They produce two 32 bit results. The device identified as the destination address is the lower of the two devices used to store the quotient and the following two devices are used to store the remainder, i.e. if D30 was selected as the destination of 32 bit division operation then D30, D31 would store the quotient and D32, D33 would store the remainder. If the location of the destination device is smaller than the obtained result, then only the portion of the result which directly maps to the destination area will be written. If bit devices are used as the destination area, no remainder value is calculated.
- 3) If the value of the source device S<sub>2</sub> is 0 (zero) then an operation error is executed and the operation of the DIV instruction is cancelled.

**Program Example**



**4.2.35 DRVA instruction**

**Instruction Description**

Table 4-53

Name	Function	Devices				Format	Steps
		S 1	S2	D1	D2		
DRVA (Drive to Absolute)	Absolute positioning	K,H, KnX,KnY, KnM,KnS T,C,D,V,Z		Y Note: Y000 to Y003 only	Y,M,S		DRVA 9 steps DDRVA 17 steps

**Operation**

This instruction is for single speed positioning using a zero home point and absolute measurements.[S1] is the Number of Pulses, [S2] is the Output Frequency, [D1] is the Pulse Output Designations, and [D2] is the Rotation Direction Signal.

**Points to note**

- 1) The target position for absolute positioning [S1] can be: 16-bit -32,768 to 32,767 pulses or 32-bit -2,147,483,648 to 2,147,483,647 pulses.
- 2) Users may use output pulse frequencies [S2], 16-bit 10 to 32,767Hz or 32-bit 10 to 200kHz.  
TP -4H type: Y0 and Y1 max up to 200KHZ,Y2 and Y3 max up to 100KHZ,But Y0+Y1+Y2+Y3 total frequencies cannot beyond 400KHZ  
TP -4H/ TPE -4H type Y0,Y1,Y2,Y3 both support max up to 200KHZ in every Y0 to Y3 channels.
- 3) Only Y000 or Y001 or Y002 or Y003 can be used for the pulse output [D1].Because of the nature of the high speed output, transistor type output units should be used with this instruction. Relay type outputs will suffer a greatly reduced life, and will cause false outputs to occur.
- 4) Rotation direction signal output [D2] operated as follows: if [D2] = OFF, rotation = negative, if [D2] = ON, rotation = positive.
- 5) If the contents of an operand are changed while the instruction is executed, it is not reflected on the operation. The new contents become effective when the instruction is next driven.
- 6) If the instruction drive contact turns off while the instruction is being executed, the machine decelerates and stops. At this time the execution complete flag M8029 does not turn ON.
- 7) Once the instruction drive contact is off, re-drive of the instruction is not possible while the pulse output flag (Y000 : [M8147], Y001 : [M8148] ,Y002 : [M8149] is ON,Y003 : [M8150]) is ON.
- 8) For operation in the absolute drive method, the travel distance from the zero point is specified.
- 9) The minimum value of output pulse frequency which can be actually used is determined by the following equation

$$\sqrt{\text{Max speed}[D8147,D8146]\text{HZ}\div(2\times\text{Acc/dec time}[D8148]\text{ms}\div 1000)}$$

10) Related device numbers

D8145: Minimum speed limit when either DRVI or DRVA are executed

D8147 (upper digit) & D8146 (lower digit): Maximum speed limit when either DRVI or DRVA are executed

D8148: Acceleration/Deceleration time adopted when ZRN or DRVI or DRVA are executed

M8145: Y000 pulse output stop (immediate)

M8146: Y001 pulse output stop (immediate)

M8152: Y002 pulse output stop (immediate)

M8153: Y003 pulse output stop (immediate)

M8147: Y000 pulse output monitor (BUS/READY)

M8148: Y001 pulse output monitor (BUS/READY)

M8149: Y002 pulse output monitor (BUS/READY)

M8150: Y003 pulse output monitor (BUS/READY)

When exporting to [Y000], the current pulse register value is [D8141 (high byte), D8140 (low byte)] (in 32-bit).

When exporting to [Y001], the current pulse register value is [D8143 (high byte), D8142 (low byte)] (in 32-bit).

When exporting to [Y002], the current pulse register value is [D8151 (high byte), D8150 (low byte)] (in 32-bit).

When exporting to [Y003], the current pulse register value is [D8153 (high byte), D8152 (low byte)] (in 32-bit).

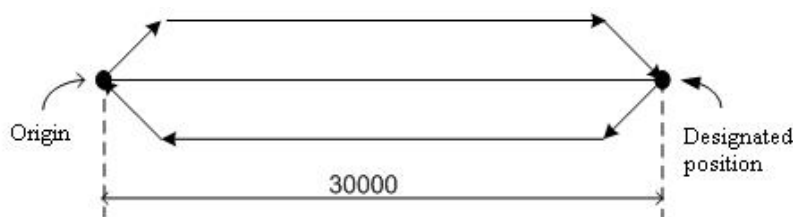
**Note:**

Attention should be paid to the instruction drive timing.

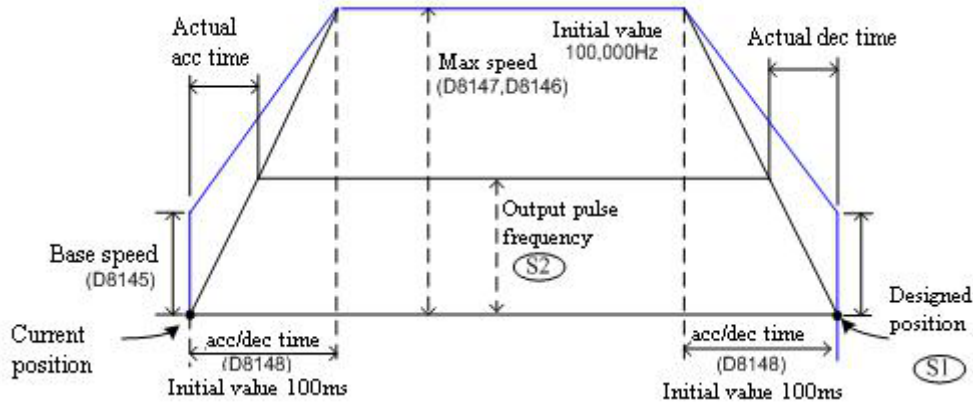
**Program Example**



The instruction is a type of control method to control the operating movement of machinery from the assigned origin toward the designated point.



During the pulse output process, the frequency will either accelerate or decelerate according to the pre-set value.



### 4.2.36 DRVI instruction

#### Instruction Description

Table 4-54

Name	Function	Devices				Format	Steps
		S 1	S 2	D1	D2		
DRVI Drive to increment	Increment positioning	K,H,KnX,KnY,KnM,KnS T,C,D,V,Z	Y Note: Y000 to Y003 only	Y,M,S		DRVI 9 steps DDRVI 17 steps	

#### Operation

This instruction is for single speed positioning in the form of incremental movements.[S1] is the Number of Pulses, [S2] is the Pulse Output Frequency, [D1] is the Pulse Output Designation, and [D2] is the Rotation Direction Signal.

#### Points to note

- 1) The maximum number of pulses [S1] available is: 16-bit -32,768 to 32,767 pulses or 32-bit -2,147,483,648 to 2,147,483,647 pulses.
- 2) Users may use output pulse frequencies [S2], 16-bit 10 to 32,767Hz or 32-bit 10 to 200kHz.  
TP -4H type: Y0 and Y1 max up to 200KHZ, Y2 and Y3 max up to 100KHZ, But Y0+Y1+Y2+Y3 total frequencies cannot beyond 400KHZ  
TP -4H/ TP E-4H type Y0,Y1,Y2,Y3 both support max up to 200KHZ in every Y0 to Y3 channels.
- 3) Only Y000 or Y001 or Y002 or Y003 can be used for the pulse output [D1]. Because of the nature of the high speed output, transistor type output units should be used with this instruction. Relay type outputs will suffer a greatly reduced life, and will cause false outputs to occur. Ensure a 'clean'

output signal when using transistor type units.

- 4) Rotation direction signal output [D2] operated as follows: if [D2] = OFF, rotation = negative, if [D2] = ON, rotation = positive.
- 5) If the contents of an operand are changed while the instruction is executed, it is not reflected on the operation. The new contents become effective when the instruction is next driven.
- 6) If the instruction drive contact turns off while the instruction is being executed, the machine decelerates and stops. At this time the execution complete flag M8029 does not turn ON.
- 7) Once the instruction drive contact is off, re-drive of the instruction is not possible while the pulse output flag (Y000: [M8147], Y001: [M8148]) is ON, Y002: [M8149] is ON, Y003: [M8150]) is ON.
- 8) For operation in the incremental drive method, the travel distance from the current position is specified with either a positive or a negative symbol.
- 9) The minimum value of output pulse frequency which can be actually used is determined by the following equation

$$\sqrt{\text{Max speed}[D8147,D8146]\text{HZ} \div (2 \times \text{Acc/dec time}[D8148]\text{ms} \div 1000)}$$

- 10) Related device numbers

D8145: Minimum speed limit when either DRVI or DRVA are executed

D8147 (upper digit) & D8146 (lower digit): Maximum speed limit when either DRVI or DRVA are executed

D8148: Acceleration/Deceleration time adopted when ZRN or DRVI or DRVA are executed

M8145: Y000 pulse output stop (immediate)

M8146: Y001 pulse output stop (immediate)

M8152: Y002 pulse output stop (immediate)

M8153: Y003 pulse output stop (immediate)

M8147: Y000 pulse output monitor (BUS/READY)

M8148: Y001 pulse output monitor (BUS/READY)

M8149: Y002 pulse output monitor (BUS/READY)

M8150: Y003 pulse output monitor (BUS/READY)

When exporting to [Y000], the current pulse register value is [D8141 (high byte), D8140 (low byte)] (in 32-bit).

When exporting to [Y001], the current pulse register value is [D8143 (high byte), D8142 (low byte)] (in 32-bit).

When exporting to [Y002], the current pulse register value is [D8151 (high byte), D8150 (low byte)] (in 32-bit).

When exporting to [Y003], the current pulse register value is [D8153 (high byte), D8152 (low byte)] (in 32-bit).

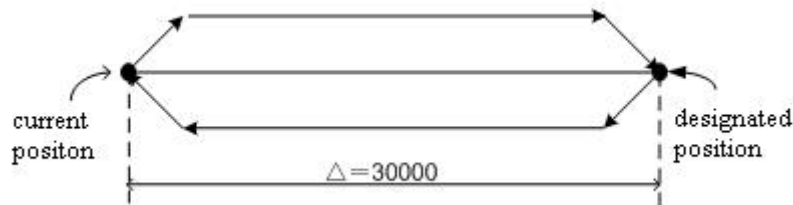
**Note:**

Attention should be paid to the instruction drive timing.

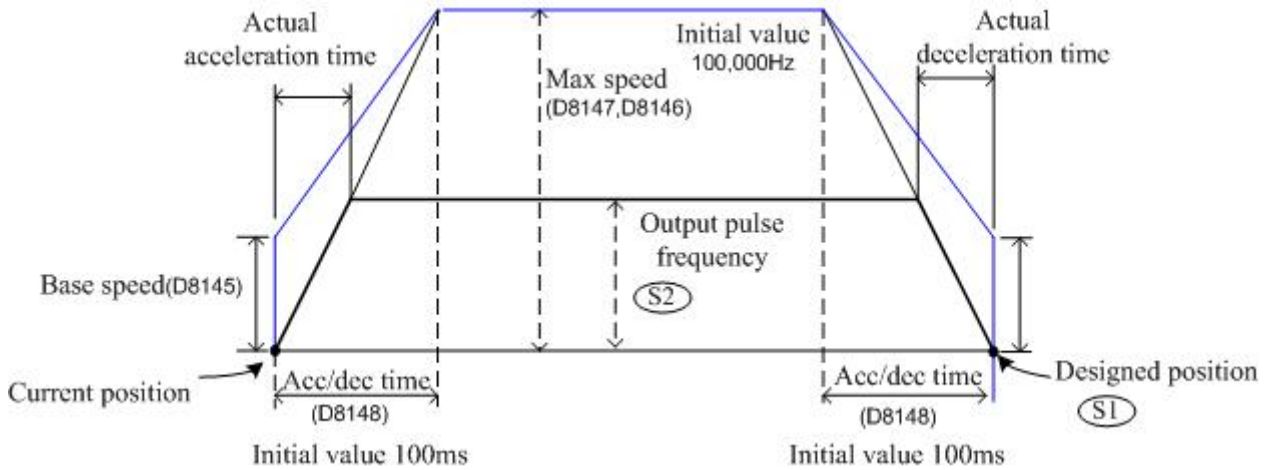
**Program Example**



With 30000 pulses exported from the Y0 port at the frequency of 4 kHz, the external server allows the machine to operate in directions that are determined by Y3.



During the pulse output process, the frequency will either accelerate or decelerate according to the present value.



**4.2.37 DSIN instruction**

**Instruction description**

Table 4-55

Name	Function	Devices		Format	Steps
		S	D		
SIN (Sine)	Calculates the sine of a floating point value	D - must be in floating point number format (32 bits).(radians)	D - a floating point value (32 bits).		DSIN, DSINP : 9 steps

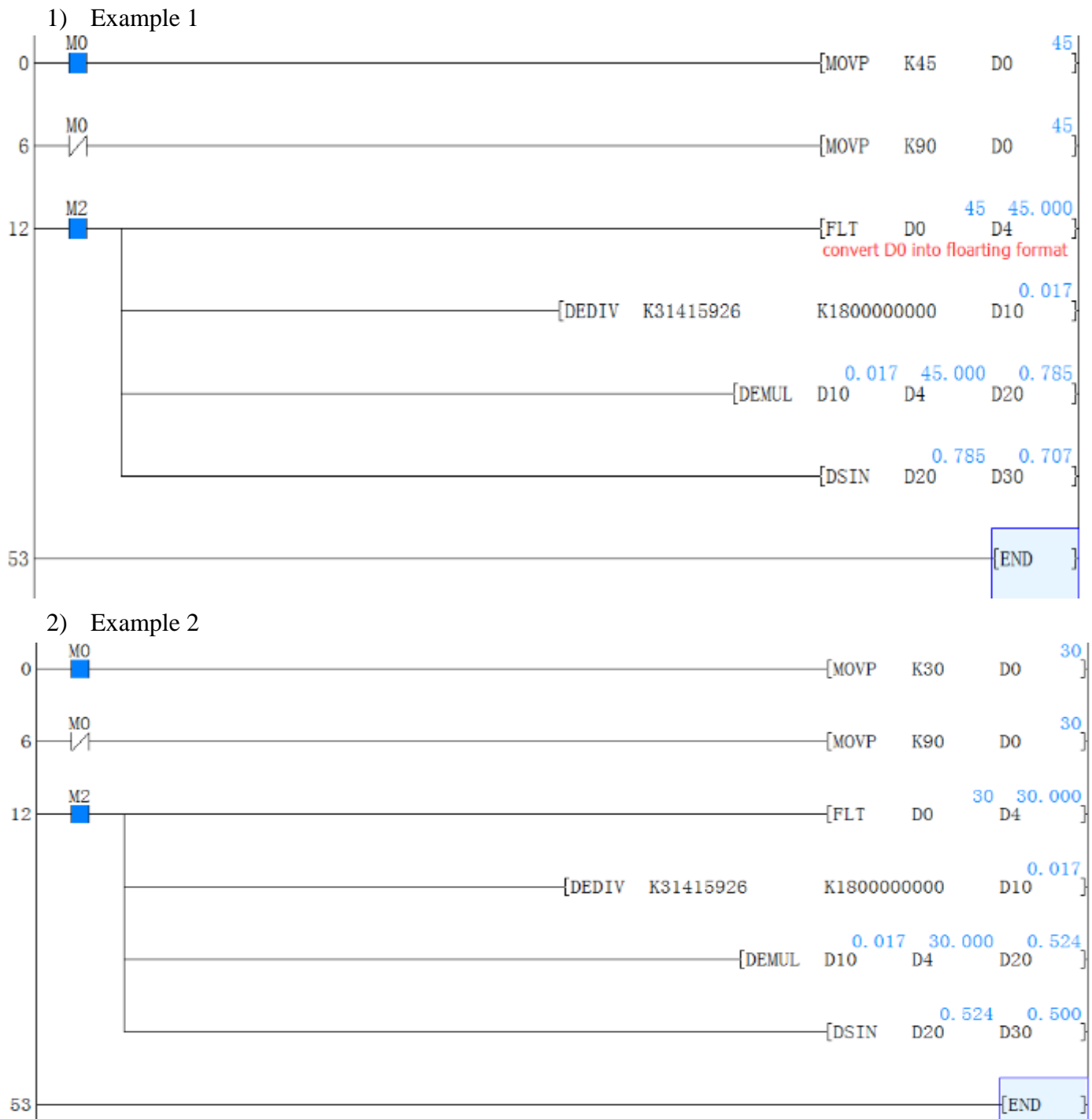
**Operation**

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

**Points to note**

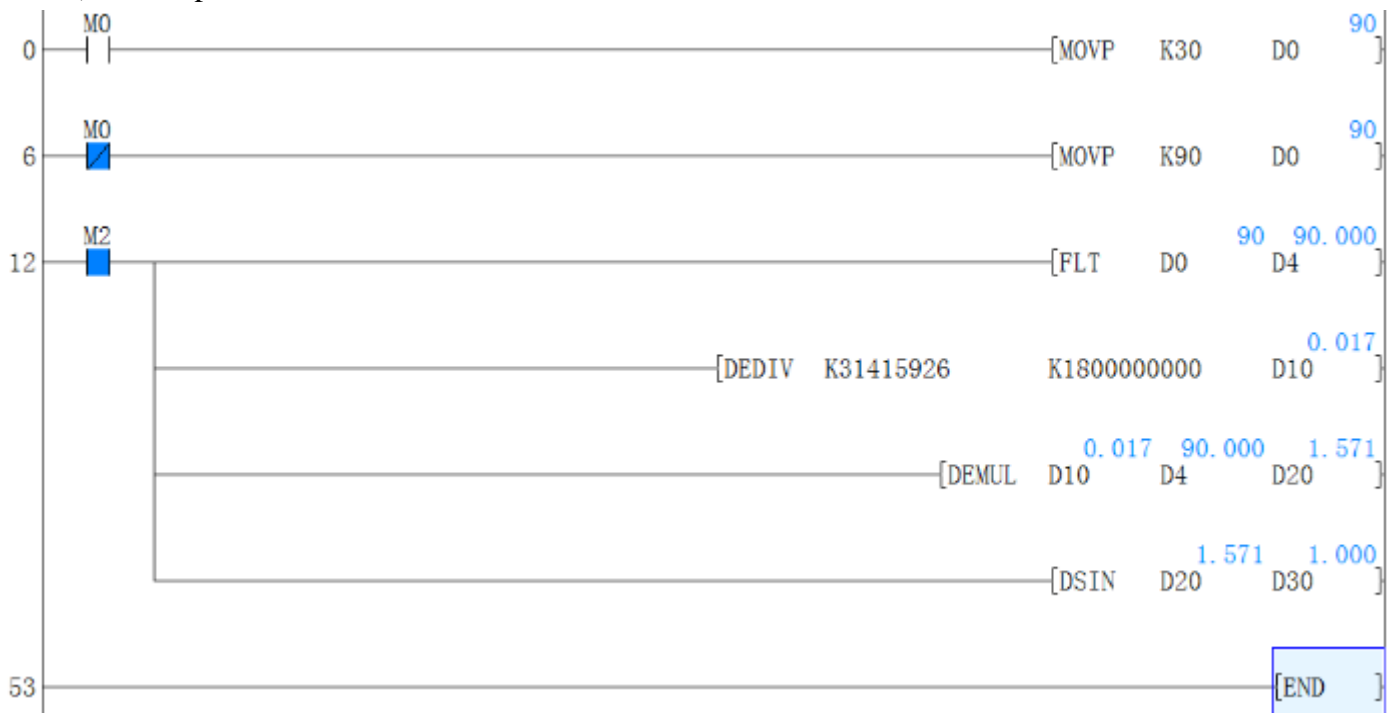
- 1) The instruction must use the double word format: i.e., **DSIN** or **DSINP**. All source and destination data will be double word; i.e., uses two consecutive data registers to store the data (32 bits).  
The source data is regarded as being in floating point format and the destination is also in floating point format.
- 2) The source value must be an angle between 0 to 360 degrees in radians; i.e.,  $0^\circ \leq S < 360^\circ$

**Program Example**





3) Example 3



4.2.38 DSW instruction

Instruction description

Table 4-56

Name	Function	Devices				Format	Steps
		S	D1	D2	n		
DSW (Digital switch)	Multiplexed reading of n sets of digital (BCD) thumb wheels	X Note: If n=2 then 8 device else 4.	Y Note: uses 4 consecutive device	T, C, D, V, Z Note: If n=2 then 2 devices else 1.	K, H ) Note: n= 1 or 2		DSW: 9 steps

Operation

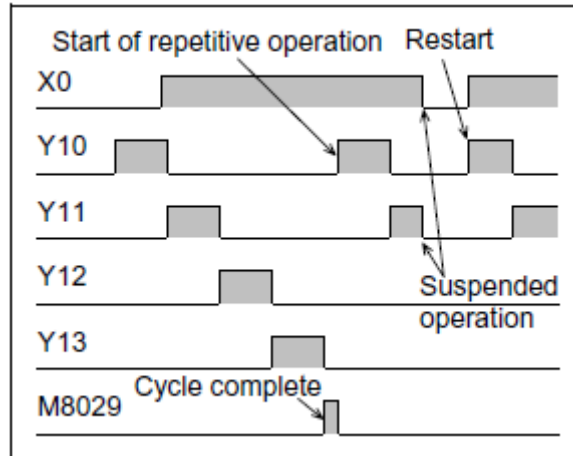
This instruction multiplexes 4 outputs (D1) through 1 or 2(n) sets of switches. Each set of switches consists of 4 thumb wheels providing a single digit input.

Points to note

- 1) When n = 1 only one set of switches are read. The multiplex is completed by wiring the thumb wheels in parallel back to 4 consecutive inputs from the head address specified in operand S. The

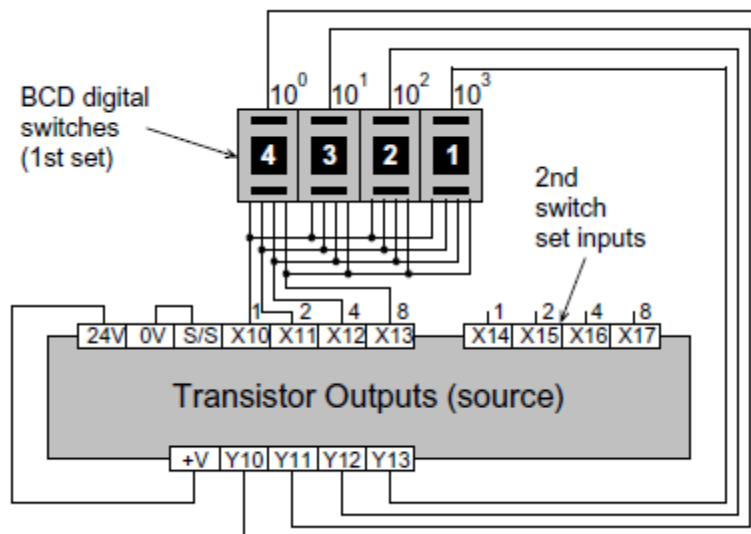
(4 digit) data read is stored in data device D2.

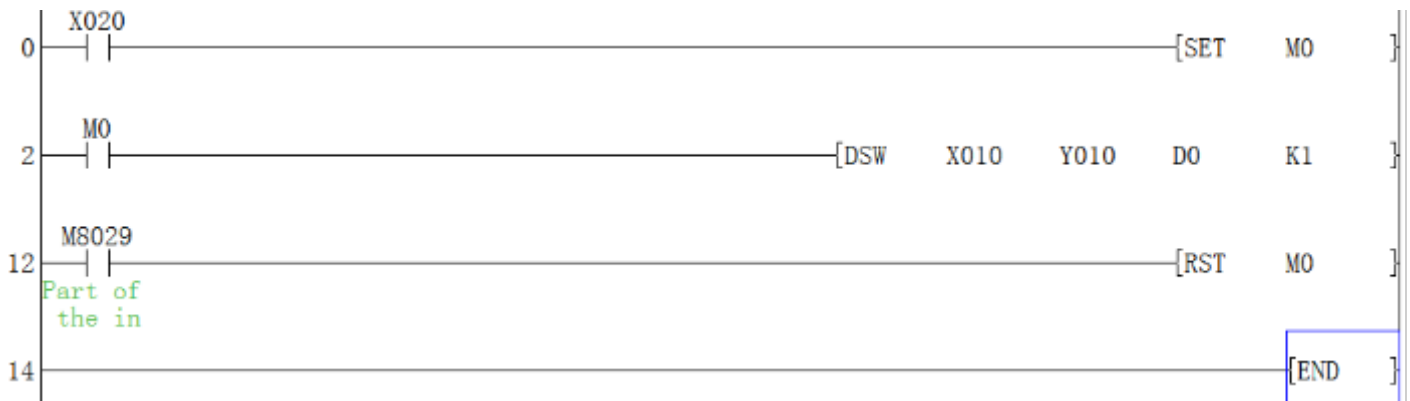
- 2) When  $n=2$ , two sets of switches are read. This configuration requires 8 consecutive inputs taken from the head address specified in operand S. The data from the first set of switches, i.e. those using the first 4 inputs, is read into data device D2. The data from the second set of switches (again 4 digits) is read into data device D<sub>2+1</sub>.
- 3) The outputs used for multiplexing (D<sub>1</sub>) are cycled for as long as the DSW instruction is driven. After the completion of one reading, the execution complete flag M8029 is set. The number of outputs used does **not** depend on the number of switches n.



- 4) If the DSW instruction is suspended during midoperation, when it is restarted it will start from the beginning of its cycle and not from its last status achieved.
- 5) It is recommended that transistor output units are used with this instruction. However, if the program technique at the right is used, relay output units can be successfully operated as the outputs will not be continually active.
- 6) Every S switch need add a diode (0.1A/50V) to X port.

**Program Example**





### 4.2.39 DTAN instruction

#### Instruction description

Table 4-57

Name	Function	Devices		Format	Steps
		S	D		
TAN (Tangent )	Calculates the tangent of a floating point value	D - must be in floating point number format (32 bits).	D - a floating point value (32 bits).		DTAN, DTANP: 9 steps

#### Operation

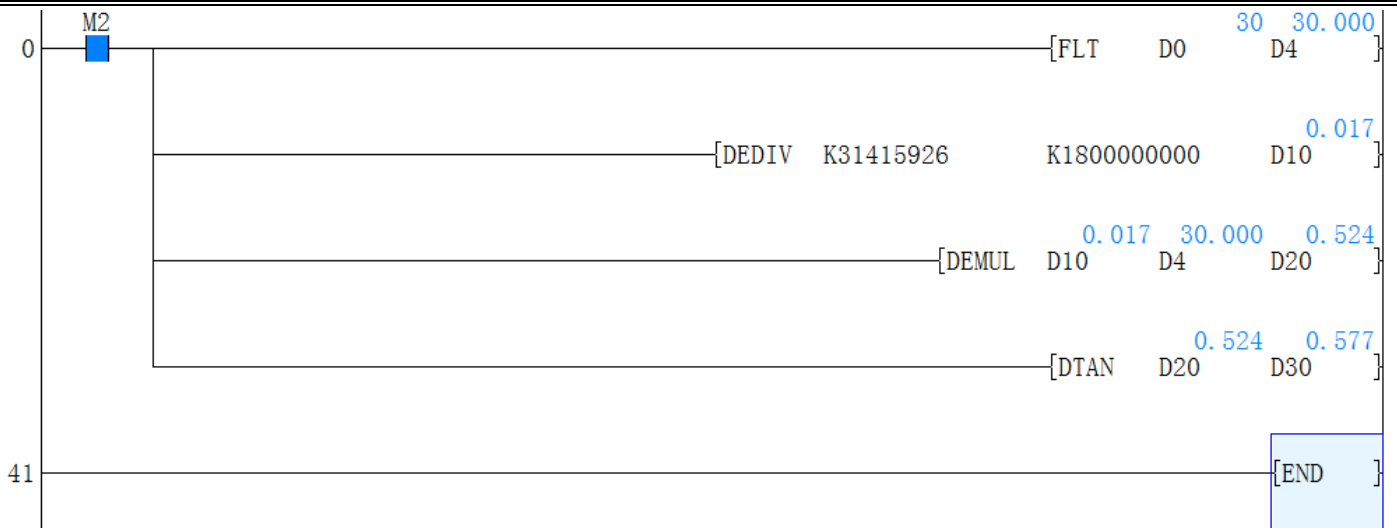
This instruction performs the mathematical TAN operation on the floating point value in S. The result is stored in D.

#### Points to note

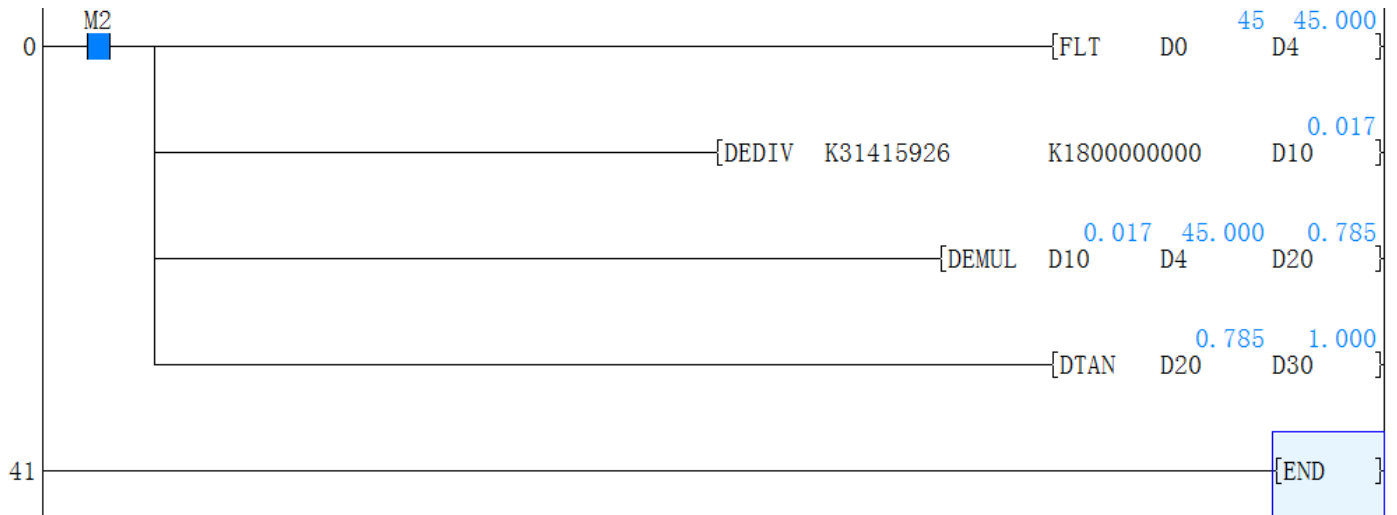
All the points for the SIN instruction apply, except that COS is calculated.

#### Program Example

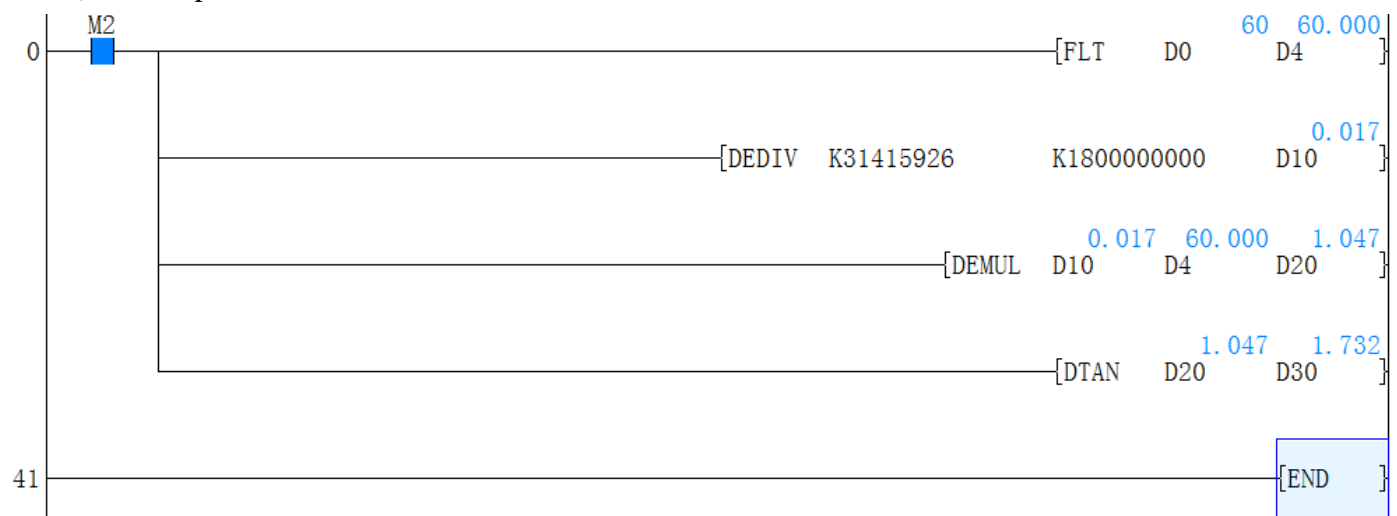
- 1) Example 1



2) Example 2



3) Example 3



### 4.2.40 IRET、EI、DI instruction

#### Instruction description

Table 4-58

Name	Function	Devices	Steps
		D	
IRET (Interrupt return)	Forces the program to return from the active interrupt routine	N/A Automatically returns to the main program step which was being processed at the time of the interrupt call.	IRET: 1 step
EI (Enable interrupts)	Enables interrupt inputs to be processed	N/A Any interrupt input being activated after an EI instruction and before FEND or DI instructions will be processed immediately unless it has been specifically disabled.	EI: 1 step
DI (Disable interrupts)	Disables the processing of interrupt routines	N/A Any interrupt input being activated after a DI instruction and before an EI instruction will be stored until the next sequential EI instruction is processed.	DI: 1 step
I (Interrupt pointer)	Identifies the beginning of an interrupt routine	A 3 digit numeric code relating to the interrupt type and operation.	IPPP: 1 step
FEND (First end)	Used to indicate the end of the main program block	N/A Note: Can be used with CJ (FNC 00), CALL (FNC 01) and interrupt routines	FEND: 1 step

#### General description of an interrupt routine

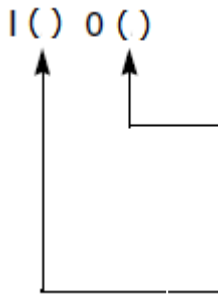
An interrupt routine is a section of program which is, when triggered, operated immediately interrupting the main program flow. Once the interrupt has been processed the main program flow continues from where it was, just before the interrupt originally occurred.

#### Operation

Interrupts are triggered by different input conditions, sometimes a direct input such as X0 is used other times a timed interval e.g. 30 msec can be used. To program and operate interrupt routines requires up to 3 dedicated instructions

##### 1) Input Interrupts

Identification of interrupt pointer number:



0: Interrupt triggered on trailing/ falling edge of input signal  
 1: Interrupt triggered on leading/ rising edge of input signal

Input number; each input number can only be used once.  
 Other units have 6 points (0 to 5 which map to X0 to X5)

Input No.	Pointer No.		Interrupt disabled instruction
	Rising edge interrupt	Falling edge interrupt	
X000	I001	I000	M8050
X001	I101	I100	M8051
X002	I201	I200	M8052
X003	I301	I300	M8053
X004	I401	I400	M8054
X005	I501	I500	M8055

**Example: I001**

The sequence programmed after the label (indicated by the I001 pointer) is executed on the leading or rising edge of the input signal X0. The program sequence returns from the interruption program when an IRET instruction is encountered.

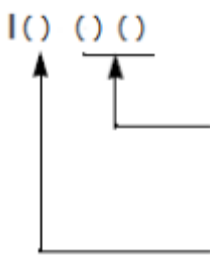
**Note:**

The following points must be followed for an interrupt to operate;

- 1) Interrupt pointers cannot have the same number in the '100s' position, i.e. I100 and I101 are not allowed.
- 2) The input used for the interrupt device must not coincide with inputs already allocated for use by other high speed instructions within the user program

**2) Timer Interrupts**

Identification of interrupt pointer number:



10 to 99 msec: the interrupt is repeatedly triggered at intervals of the specified time.

Timer interrupts number 3 points (6 to 8)

Input No.	Interrupt period (ms)	Interrupt disable instruction
I6□□	Input 1~99 to □□ in the instructions, for example, I605, which executes one timing interrupt every 5 ms	M8056
I7□□		M8057
I8□□		M8058

**Example: I610**

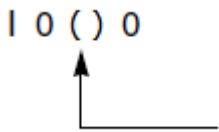
The sequence programmed after the label (indicated by the I610 pointer) is executed at intervals of 10msec. The program sequence returns from the interruption program when an IRET instruction is encountered.

**Note:**

Interrupt pointers cannot have the same number in the '100's' position, i.e. I610 and I650 are not allowed.

**3) Counter Interrupts**

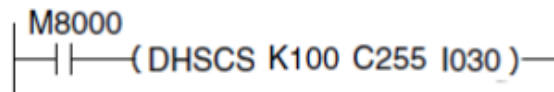
Identification of interrupt pointer number:



Counter interrupt number 6 points (1 to 6). Counter interrupts can be entered as the output devices for High Speed Counter Set (HSCS). To disable the Counter Interrupts Special Auxiliary Relay M8059 must be set ON.

Input No.	Interrupt disable instruction
I010	M8059
I020	
I030	
I040	
I050	
I060	

**Example:**



The sequence programmed after the label(indicated by the I030 pointer) is executed once the value of High Speed Counter C255 reaches/equals the preset limit of K100 identified in the example HSCS.

Note point: Please check HSCS instruction page.

**Defining an interrupt routine**

An interrupt routine is specified between its own unique interrupt pointer and the first occurrence of an

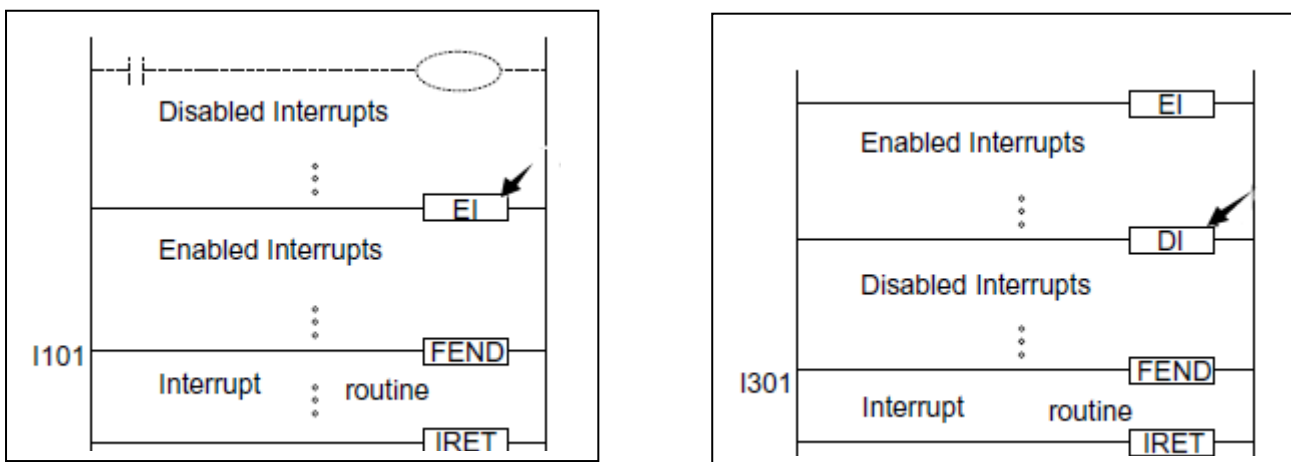
IRET instruction.

Interrupt routines are ALWAYS programmed after an FEND instruction.

The IRET instruction may only be used within interrupt routines.

### Controlling interrupt operations

The PLC has a default status of disabling interrupt operation. The EI instruction must be used to activate the interrupt facilities. All interrupts which physically occur during the program scan period from the EI instruction until the FEND or DI instructions will have their associated interrupt routines run. If these interrupts are triggered outside of the enclosed range (EI-FEND or EI-DI, see diagram below) they will be stored until the EI instruction is processed on the following scan. At this point the interrupt routine will be run.



### Nesting interrupts

- 1) Interrupts may be nested for two levels. This means that an interrupt may be interrupted during its operation.
- 2) However, to achieve this, the interrupt routine which may be further interrupted must contain the EI and DI instructions; otherwise as under normal operation, when an interrupt routine is activated all other interrupts are disabled.

### Simultaneously occurring interrupts

If more than one interrupt occurs sequentially, priority is given to the interrupt occurring first. If two or more interrupts occur simultaneously, the interrupt routine with the lower pointer number is given the higher priority.

### Using general timers within interrupt routines

PLC's have a range of special timers which can be used within interrupt routines. For more information please see Timers Used in Interrupt and 'CALL' Subroutines.



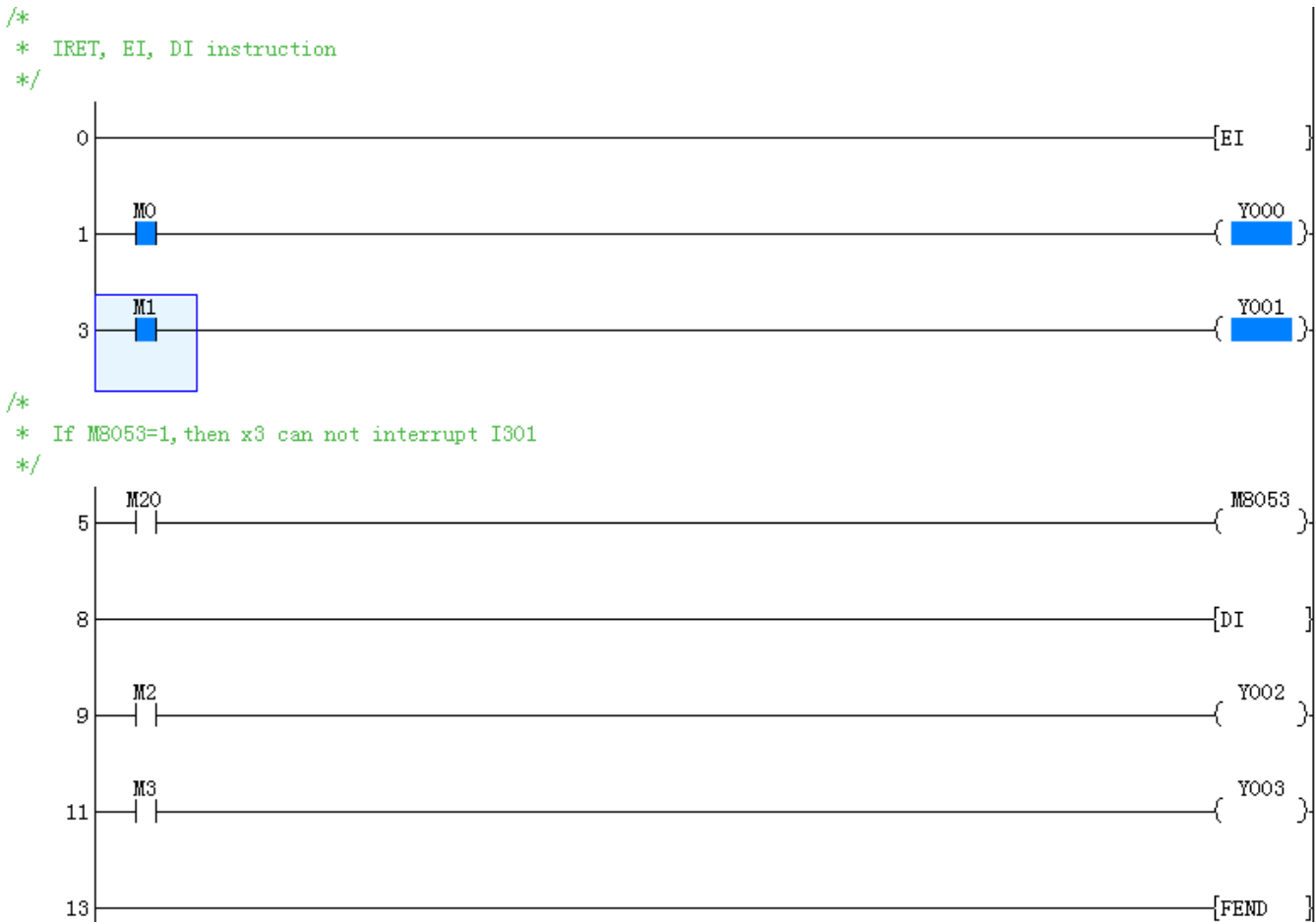
**Input triggers signals - pulse duration**

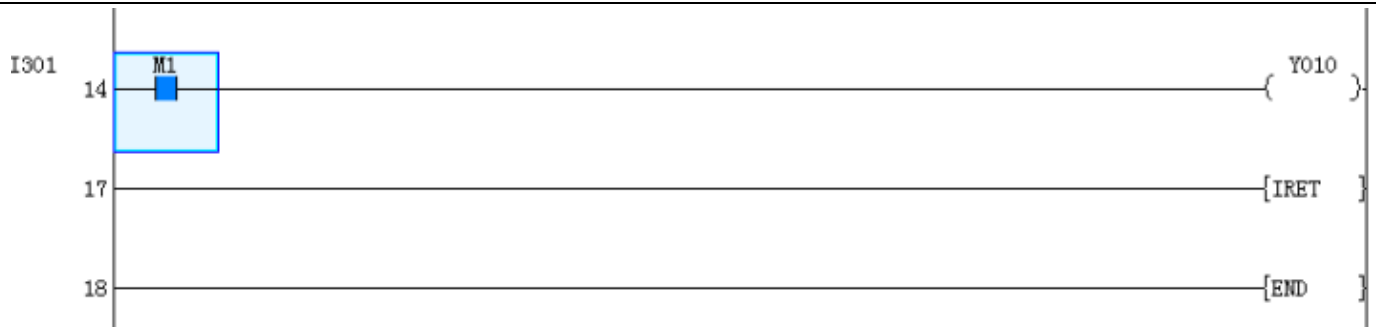
Interrupt routines which are triggered directly by interrupt inputs, such as X0 etc., require a signal duration of approximately 200µsec, i.e. the input pulse width is equal or greater than 200µsec. When this type of interrupt is selected, the hardware input filters are automatically reset to 50µsec. (under normal operating circumstances the input filters are set to 10msec).

**Pulse catch function**

Direct high speed inputs can be used to ‘catch’ short pulsed signals. When a pulse is received at an input a corresponding special M coil is set ON. This allows the ‘captured’ pulse to be used to trigger further actions, even if the original signal is now OFF. TP units require the EI instruction to activate pulse catch for inputs X0 through X5, with M8170 to M8175 indicating the caught pulse. Note that, if an input device is being used for another high speed function, then the pulse catch for that device is disabled.

**Program Example**





### 4.2.41 ENCO instruction

#### Instruction description

Table 4-59

Name	Function	Devices			Format	Steps
		S	D	n		
ENC O (Enco de)	Then location of the highest active bit is stored as a numerical position from the head address	X, Y, M, S, T, C, D, V, Z	T, C, D, V, Z.	K, H,) Note: S=X, Y, M, S then n range=1-8 S= T,C,D then n range = 1-4 n = 0, then no processing		DTAN , DTAN P: 9 steps

#### Operation

The highest active bit within the readable range has its location noted as a numbered offset from the source head address (S). This is stored in the destination register (D).

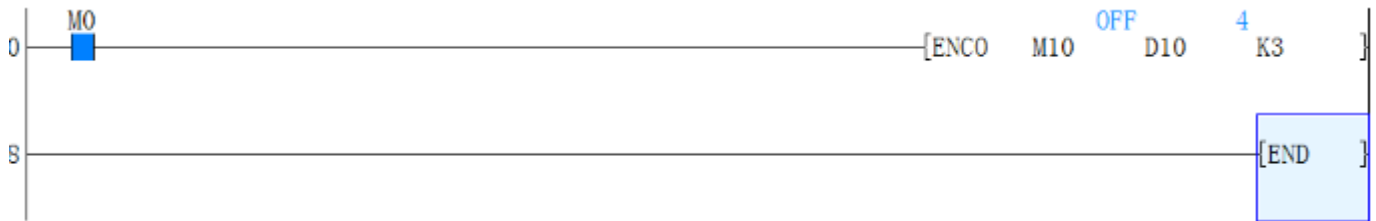
#### Points to note

- 1) The readable range is defined by the largest number storable in a binary format within the number of destination storage bits specified by n, i.e. if n was equal to 4 bits a maximum number within the range 0 to 15 can be written to the destination device. Hence, if bit devices were being used as the source data, 16 bit devices would be used, i.e. the head bit device and 15 further, consecutive devices.
- 2) If the stored destination number is 0 (zero) then the source head address bit is ON, i.e. The active bit has a 0 (zero) offset from the head address. However, if NO bits are ON within the source area, 0 (zero) is written to the destination device and an error is generated.
- 3) When the source device is a data or word device n must be taken from the range 1 to 4 as there are only 16 source bits available within a single data word.

**Program example**

If n=K3 in the below demo, then user need to check m10 m11 M12 value.

If M10=1, M11=1, M12=1(111), user need choose the highest bit m12. So D10=100(2)=4(10)



**4.2.42 FEND instruction**

**Instruction description**

Table 4-60

Name	Function	Devices	Steps
		D	
FEND (First end)	Used to indicate the end of the main program block	N/A Note: Can be used with CJ (FNC 00), CALL (FNC 01) and interrupt routines	FMOV,FMOV P: 7 steps DFMOV,DFM O VP: 13 steps

**Operation**

An FEND instruction indicates the first end of a main program and the start of the program area to be used for subroutines. Under normal operating circumstances the FEND instruction performs a similar action to the END instruction, i.e. output processing, input processing and watchdog timer refresh are all carried out on execution.

**Points to note**

- 1) The FEND instruction is commonly used with CJ-P-FEND, CALL-P-SRET and I-IRET program constructions (P refers to program pointer, I refers to interrupt pointer).Both CALL pointers/subroutines and interrupt pointers (I) subroutines are ALWAYS programmed after an FEND instruction, i.e. these program features NEVER appear in the body of a main program.
- 2) Multiple occurrences of FEND instructions can be used to separate different subroutines (see diagram above).
- 3) The program flow constructions are NOT allowed to be split by an FEND instruction.
- 4) FEND can never be used after an END instruction.

### 4.2.43 FLT instruction

#### Instruction description

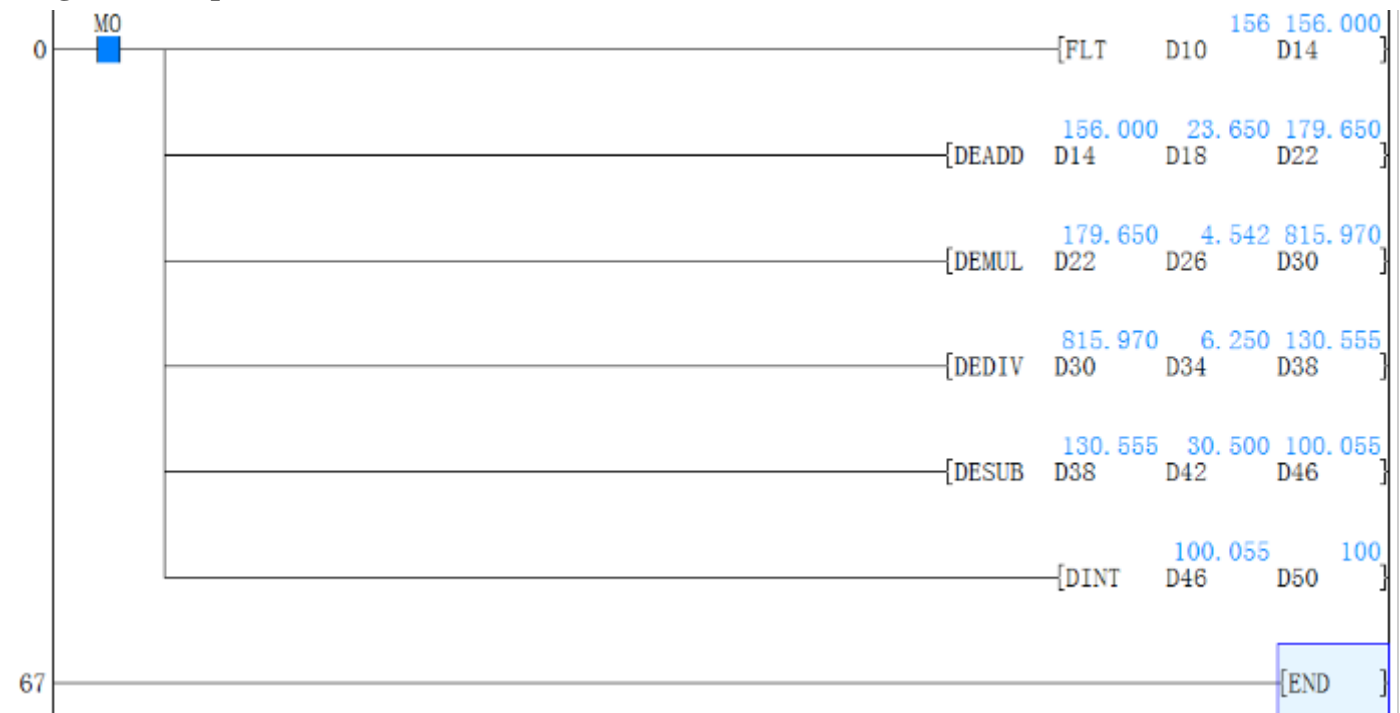
Table 4-61

Name	Function	Devices		Format	Steps
		S	D		
FLT (Floating point)	Used to convert Decimal data to floating point format	D	D		FLT, FLTP: 5 steps DFLT, DFLTP: 9 steps

#### Operation

The instruction converts the decimal data S to floating digits, and saves the result in D and D+1 units. Please note that two consecutive devices (D and D+1) will be used to store the converted float number. This is true regardless of the size of the source data (S), i.e. whether (S) is a single device (16 bits) or a double device (32 bits) has no effect on the number of destination devices (D) used to store the floating point number. (The instruction INT: Convert floating point value to decimal value)

#### Program example



### 4.2.44 FMOV instruction

#### Instruction description

Table 4-62

Name	Function	Devices			Format	Steps
		S	D	n		
FMOV (Fill move)	Copies a single data device to a range of destination devices	KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM ,KnS ,T, C, D, V, Z	K, H ) Note: n ≤ 512		FMOV,FMOV P:7 steps DFMOV,DFM OVP: 13 steps

**Operation**

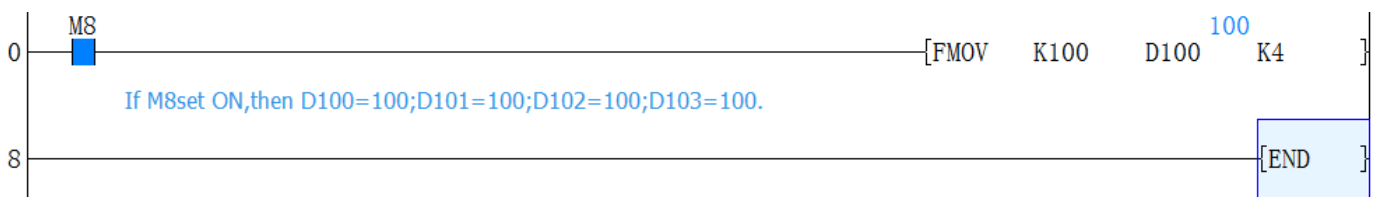
The data stored in the source device (S) is copied to every device within the destination range. The range is specified by a device head address (D) and a quantity of consecutive elements (n). If the specified number of destination devices (n) exceeds the available space at the destination location, then only the available destination devices will be written to.

**Program example**



The following operation is completed when M8=ON.

- k100 → D100
- k100 → D101
- k100 → D102
- k100 → D103



**4.2.45 FOR, NEXT instruction**

**Instruction description**

Table 4-63

Name	Function	Devices	Format	Steps
		S		
FOR (Start of a FOR-NEXT loop)	Identifies the start position and the number of repeats for the loop	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z		FOR: 3 step
NEXT (End of a FOR-NEXT loop)	Identifies the end position for the loop	N/A Note: The FOR-NEXT loop can be nested for 5 levels, i.e. 5 FOR-NEXT loops are programmed within each other.		NEXT: 1 step

**Operation**

The FOR and NEXT instructions allow the specification of an area of program, i.e. the program enclosed by the instructions, which is to be repeated S number of times.

**Points to note**

- 1) The FOR instruction operates in a 16 bit mode hence, the value of the operand S may be within the range of 1 to 32,767. If a number between the range -32,768 and 0 (zero) is specified it is automatically replaced by the value 1, i.e. the FOR-NEXT loop would execute once.
- 2) The NEXT instruction has NO operand.
- 3) The FOR-NEXT instructions must be programmed as a pair e.g. for every FOR instruction there **MUST** be an associated NEXT instruction. The same applies to the NEXT instructions, there **MUST** be an associated FOR instruction. The FOR-NEXT instructions must also be programmed in the correct order. This means that programming a loop as a NEXT-FOR (the paired NEXT instruction proceeds the associated FOR instruction) is **NOT** allowed.

Inserting an FEND instruction between the FOR-NEXT instructions, i.e. FOR-FEND- NEXT, is NOT allowed. This would have the same effect as programming a FOR without a NEXT instruction, followed by the FEND instruction and a loop with a NEXT and no associated FOR instruction.

- 4) A FOR-NEXT loop operates for its set number of times **before** the main program is allowed to finish the current program scan.

- 5) When using FOR-NEXT loops care should be not taken exceeding the PLC's watchdog timer setting. The use of the WDT instruction and/or increasing the watchdog timer value is recommended.

**Nested FOR-NEXT loops**

FOR-NEXT instructions can be nested for 5 levels. This means that 5 FOR-NEXT loops can be sequentially programmed within each other.

In the example a 3 level nest has been programmed. As each new FOR-NEXT nest level is encountered the number of times that loop is repeated is increased by the multiplication of all of the surrounding/previous loops.

For example, loop C operates 4 times. But within this loop there is a nested loop, B. For every completed cycle of loop C, loop B will be completely executed, i.e. it will loop D0Z times. This again applies between loops B and A.

The total number of times that loop A will operate for ONE scan of the program will equal;

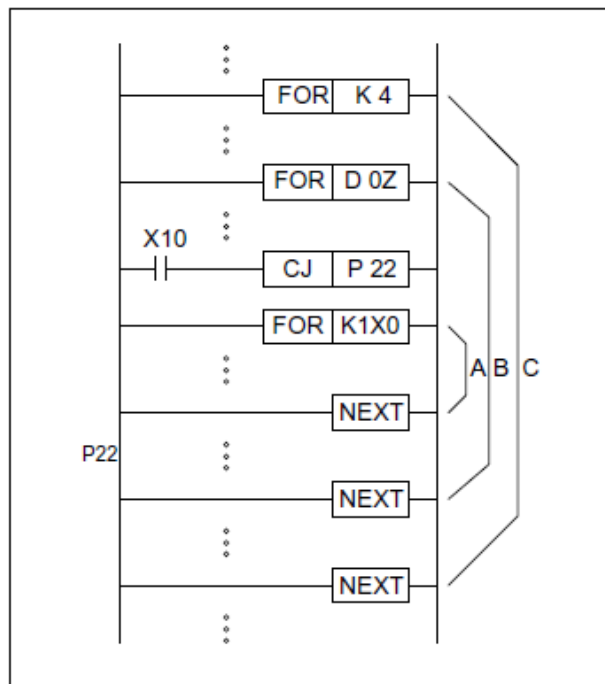
- 1) The number of loop A operations multiplied by
- 2) The number of loop B operations multiplied by
- 3) The number of loop C operations

If values were associated to loops A, B and C, e.g. 7, 6 and 4 respectively, the following number of operations would take place in ONE program scan:

Number of loop C operations = 4 times

Number of loop B operations = 24 times (C × B, 4 × 6)

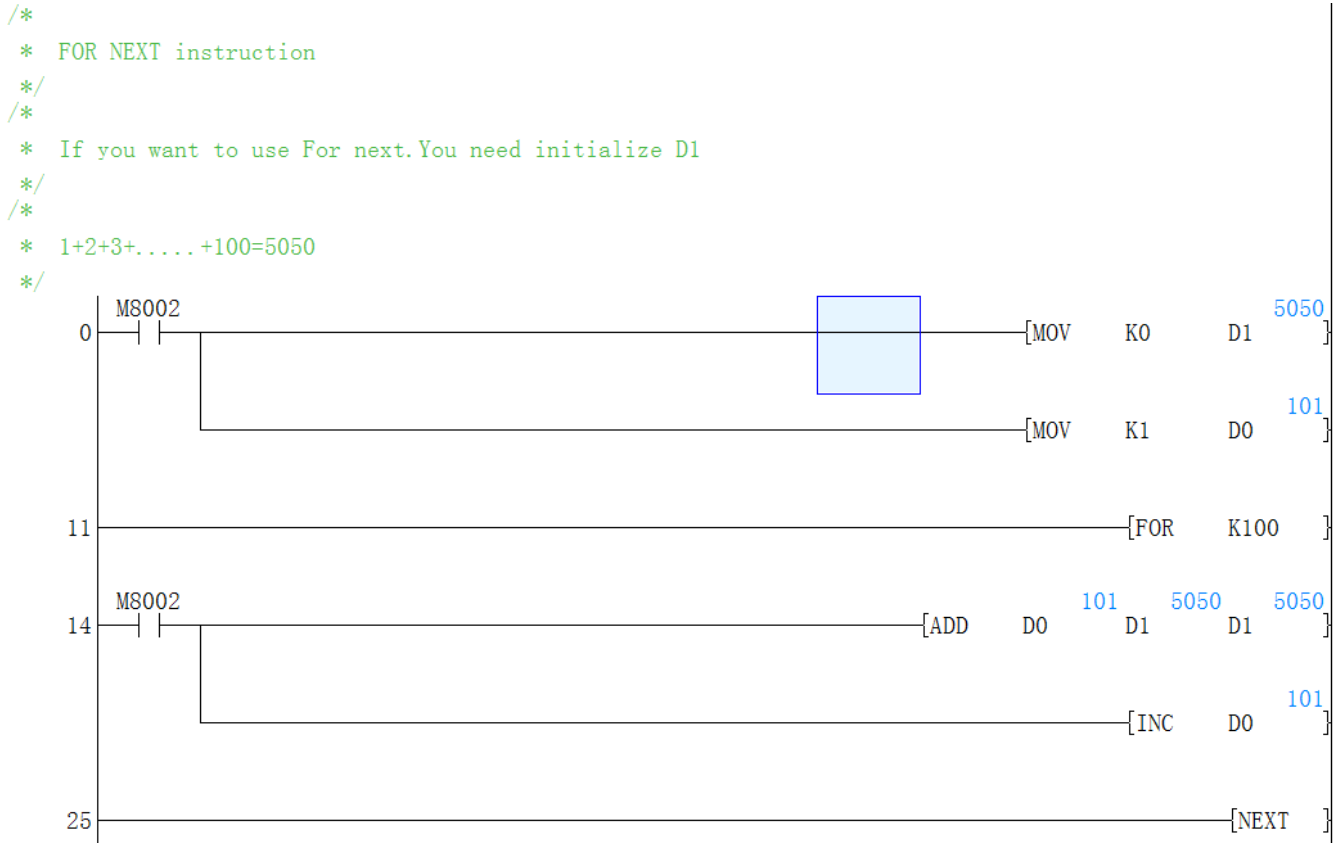
Number of loop A operations = 168 times (C × B × A, 4 × 6 × 7)



**Note:**

The use of the CJ programming feature, causing the jump to P22 allows the 'selection' of which loop will be processed and when, i.e. if X10 was switched ON, loop A would no longer operate.

**Program example**



**4.2.46 FROM instruction**

**Instruction description**

Table 4-64

Name	Function	Devices				Format	Steps
		m1	m2	D	n		
FROM (FROM)	Read data from the buffer memories of attached special function blocks	K, H) Note: m1= 0 to 15	K, H ) Note: 0 to 3276 7	KnY, KnM, KnS, T, C, D, V, Z	K, H ) Note : 0 to 3276 7	$\left  \begin{array}{c} X1 \\ \text{---} \end{array} \right  \left  \begin{array}{c} \text{---} \\ \text{---} \end{array} \right  \text{(FROM } \begin{array}{c} m1 \\ K2 \end{array} \begin{array}{c} m2 \\ K10 \end{array} \begin{array}{c} D \\ D10 \end{array} \begin{array}{c} n \\ K6 \end{array} \text{)} \left  \begin{array}{c} \text{---} \\ \text{---} \end{array} \right $	FROM, FROM P: 9 steps DFRO M, DFRO MP: 17 steps

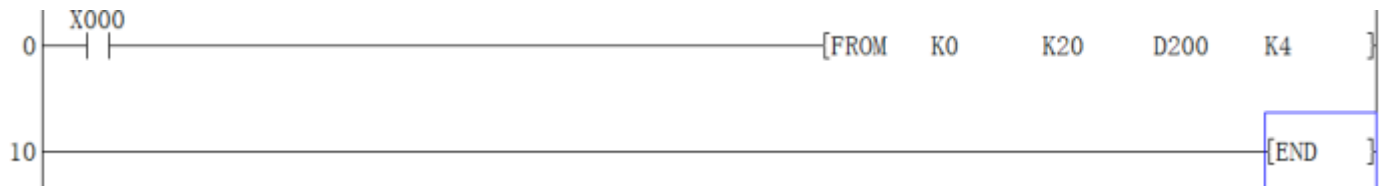


**Operation**

The FROM instruction reads the data n words of data starting from the BFM register (buffer memory address ) in the special extended module. The read data is stored in the PLC at head address D for n word devices.

- 1) m1 is the which number special module close to PLC.
- 2) m2 is the special module BFM address ID number inside the special module.
- 3) D is the storage address after reading from module.
- 4) n how many head BFM address parameters will be read from module into head address D.

**Program example**



When X0=ON, D200 reads K4 word of data starting from the zero module’s BFM 20 register (BFM20---->D200, BFM21---->D201, BFM22---->D202, BFM23---->D203)

**4.2.47 GBIN instruction**

**Instruction description**

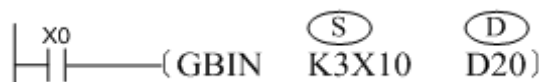
Table 4-65

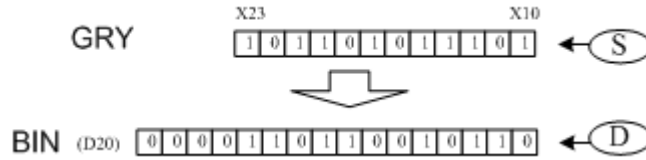
Name	Function	Devices		Format	Steps
		S	D		
GBIN (Gray Code)	Calculates the integer value of a gray code	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z		GBIN,G BINP: 5 steps DGBIN, DGBINP : 9 steps

**Operation**

The GRAY CODE value in S is converted to the normal binary equivalent and stored at D.

**Program example**





### 4.2.48 GRY instruction

#### Instruction description

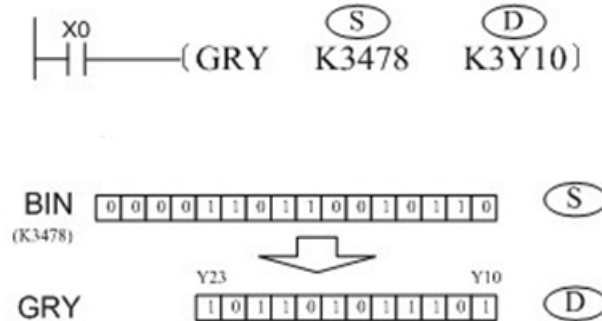
Table 4-66

Name	Function	Devices		Format	Steps
		S	D		
GRY (Gray Code)	Calculates the gray code value of an integer	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z		GRY,GRY P: 5 steps DGRY,DG RYP: 9 steps

#### Operation

The binary integer value in S is converted to the GRAY CODE equivalent and stored at D.

#### Program example



### 4.2.49 HEX instruction

#### Instruction description

Table 4-67

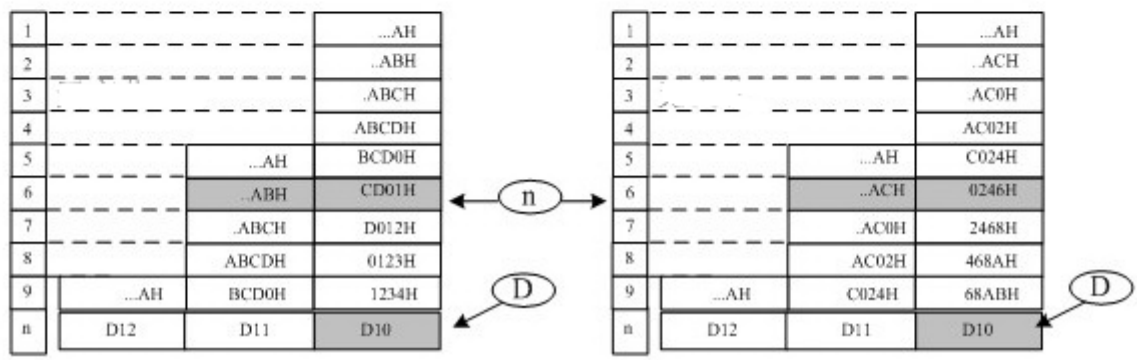
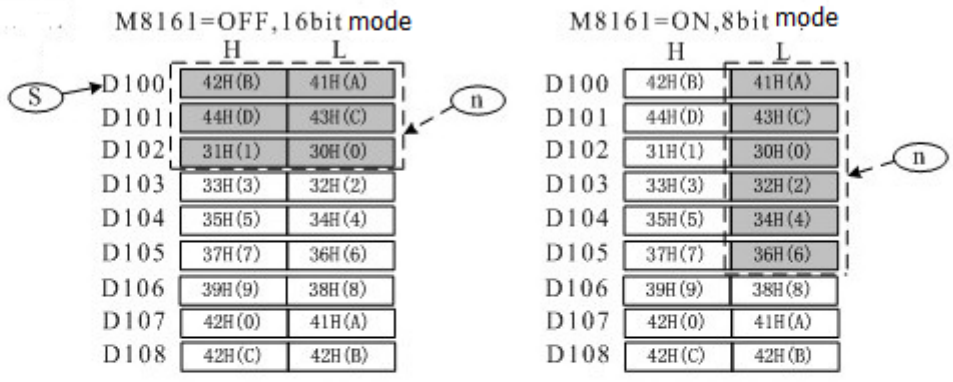
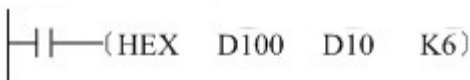
Name	Function	Devices			Format	Steps
		S	D	n		
HEX (Converts data value from ASCII to hexadecimal equivalent)	Converts a data value from ASCII to hexadecimal equivalent	K, H, KnX, KnY, KnM, KnS, T, C, D	KnY, KnM, KnS, T, C, D, V, Z	K, H Note: n = 1 to 256 )		HEX, HEXP: 7 steps

**Operation**

- 1) This instruction reads n ASCII data bytes from head source address (S) and converts them in to the equivalent Hexadecimal character. This is then stored at the destination (D) for n number of bytes.
- 2) Please note that this instruction ‘works in reverse’ to the ASCI instruction, i.e. ASCII data stored in bytes is converted into associated hexadecimal characters. The HEX instruction can be used with the M8161 8bit/16bit flag.
- 3) In this case the source data (S)is read from either the lower byte (8bits) when M8161 is ON, or the whole word when M8161 is OFF i.e. using the example above with the following data in devices D100 and D101 and D102 respectively (42H,41H) (44H,43H)(31H,30H)and assuming M8161 is ON.
- 4) The ASCII data is converted to its hexadecimal equivalent and stored sequentially digit by digit from the destination head address.

**Program Example**

Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D100	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	AB
D101	1	1	0	0	0	0	1	0	0	0	1	0	0	0	1	0	CD
D102	0	0	0	0	1	1	0	0	1	0	0	0	1	1	0	0	01
D103	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	23
D104	0	0	1	0	1	1	0	0	1	0	1	0	1	1	0	0	45
D105	0	1	1	0	1	1	0	0	1	1	1	0	1	1	0	0	67



1) Example1 (M8161=OFF,16-bit mode)



Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D10	1	0	0	0	0	0	0	0	1	0	1	1	0	0	1	1	CD01
D11	1	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	00AB

2) Example2 (M8161=ON, 8-bit mode)



Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D10	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0246
D11	0	0	1	1	0	1	0	1	0	0	0	0	0	0	0	0	00AC


**Note:**

- 1) The M8161 mode sign is shared with instructions of RS/HEX/ASCII/CCD, etc.
- 2) The source data of S must be ASCII chars, otherwise errors will occur during the conversion.

### 4.2.50 HKY instruction

#### Instruction description

Table 4-68

Name	Function	Devices				Format	Steps
		S	D1	D2	D3		
HKY (Hexa decimal key input)	Multiplexes inputs and outputs to create a numeric keyboard with 6 function keys	X, Note: uses 4 consecutive devices	Y, Note: uses 4 consecutive devices	T, C, D, V, Z Note: uses 2 consecutive devices for 32 bit operation	Y, M, S Note: uses 8 consecutive devices		HKY : 9 steps DHK Y: 17 steps

#### Operation

This instruction creates a multiplex of 4 outputs (D1) and 4 inputs (S) to read in 16 different devices. Which are the decimal 0~9 keys and the functional keys of A~F. When the keys are pressed (ON), decimal numbers of 4 bits between 0~9999 or functional keys between A~F can be entered, depending on the sequence of the key press actions. If 32bit instructions are used, decimal numbers of 8 bits between 0~99,999,999 or functional keys between A~F can be entered. Where:

(S) is the input port X of the keys, 4 X ports will be used;

(D1) is the starting port button of scanning output Y port, and it uses the four Y ports.

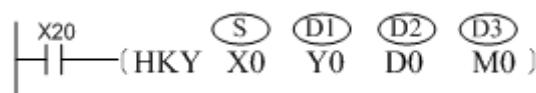
(D2) is the storage unit for the entered values from the keys, with a range of 0~9999. If 32bit instructions are used, decimal numbers of 8 bits between 0~99,999,999 can be entered.

(D3) is the address which display the entering status of the keys, which occupies a variable unit of 8 continuous bits.

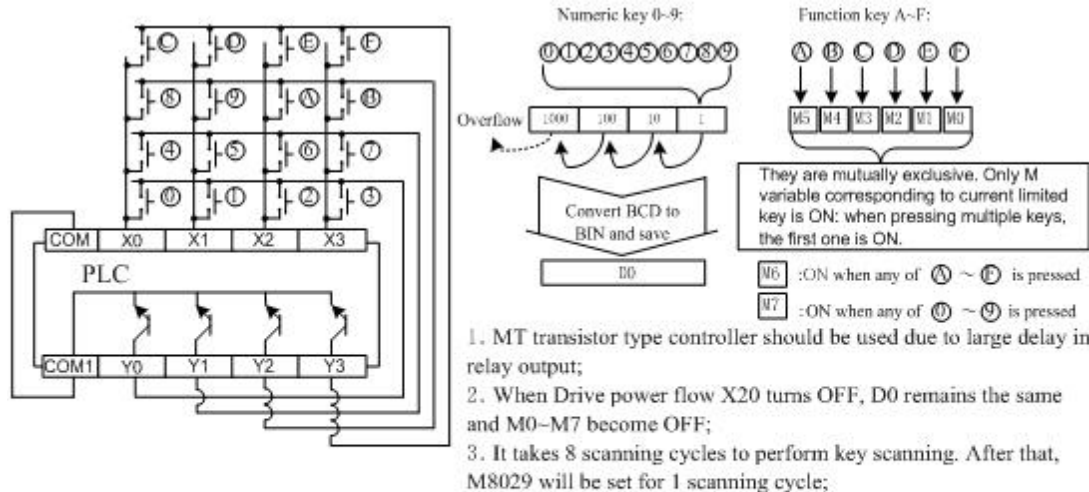
#### Note:

This instruction can only be used in transistor-output type PLC.

#### Program example



Wiring diagram and parameter response instruction as follows:



1. MT transistor type controller should be used due to large delay in relay output;
2. When Drive power flow X20 turns OFF, D0 remains the same and M0-M7 become OFF;
3. It takes 8 scanning cycles to perform key scanning. After that, M8029 will be set for 1 scanning cycle;

- Since it takes several execution period to perform key scanning, use constant scanning mode or timing interrupt processing to avoid the influence on X port filtering.
- Notice for expansion function:  
When special variable M8167 is set to ON, this instruction stores the 16bit data of keys 0-F to  $(D2)$ .

### 4.2.51 HOUR instruction

#### Instruction description

Table 4-69

Name	Function	Devices			Format	Steps
		S	D1	D2		
Hour (Hour meter)	Hour meter	K,H, KnX, KnY, KnM, KnS, T,C,D,V,Z	D Note: Data register should be backed	Z,Y, M,S	$H-(HEX \quad \bar{S} \quad \bar{D} \quad \bar{n} \quad K6)$	HOUR 7 steps D HOUR 13 steps

#### Operation 1: 16 bit instruction

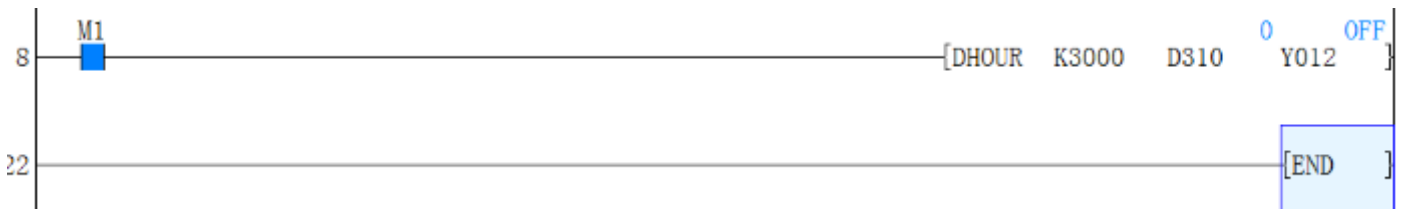
- [S] = Period of time before [D2] turns on (Hrs)
  - [D1] = Current value in Hours
  - [D1]+1 = Current value, if less than 1 hour, time is specified in seconds.
  - [D2] = Alarm output destination, turns on when [D1] exceeds [S]
- In the below example, [D2] turns on at 2000 hours and 1 second.

**Program example**



**Operation 2: 32 bit instruction**

- [S] = Period of time in which [D2] turns on (Hrs)
  - [D1] = Current value in Hours
  - [D1]+2 = Current value, if less than 1 hour.
  - [D2] = Alarm output destination, when [D1] exceeds [S]
- In the below example, [D2] turns on at 3000 hours and 1 second.



**Points to note**

- 1) In order to continuously use the current value data, even after a power OFF and ON, specify a data register which is backed up against power interruption.
- 2) The hour meter will continue operation even after the alarm output [D2] turns ON.  
Operation will stop when the value of [D1] reaches the maximum for the specified 16 or 32 bit operation.  
If continuous operation is required, clear the value stored in [D1] to [D1]+1 (16-bit) and [D1] to [D1]+2 (32-bit).

**4.2.52 INC instruction**

**Instruction Description**

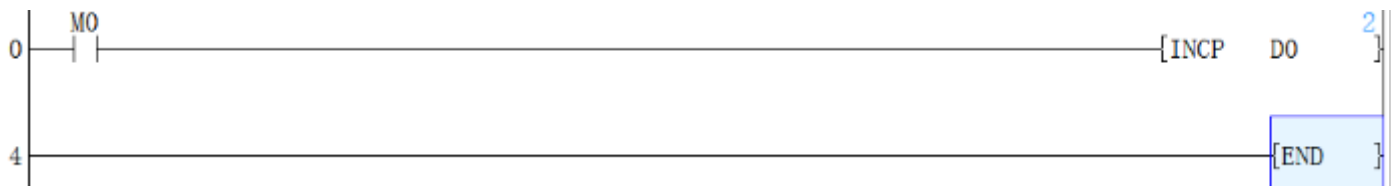
Table 4-70

Name	Function	Devices	Format	Steps
		D		
INC (Increment)	The designated device is incremented by 1 on every execution of the instruction	KnY, KnM, KnS, T, C, D, V, Z		INC, INC P: 3 steps DINC, DINCP: 5 steps

**Operation**

- 1) Every execution of the instruction the device specified as the destination D, has its current value incremented (increased) by a value of 1.
- 2) In 16 bit operation, when +32,767 is reached, the next increment will write a value of -32,768 to the destination device.
- 3) In 32 bit operation, when +2,147,483,647 is reached the next increment will write a value of -2,147,483,648 to the destination device.

**Program example**



If M0 set on, then D0 will increase to 1.If M0 set on again, then D0 increase to 2.

**4.2.53 INCD instruction**

**Instruction Description**

Table 4-71

Name	Function	Devices				Format	Steps
		S1	S2	D	n		
INCD (Incremental drum sequencer)	Generates a single output sequence in response to counter data	KnX, KnY, KnM, KnS, T, C, D	C Uses 2 consecutive counters	Y, M, S Note: n consecutive devices are used	Z, Y, M ,S		INCD: 9 steps

**Operation**

This instruction generates a sequence of sequential output patterns (there are n number of addressed outputs) in response to the current value of a pair of selected counters (S2, S2+1).

**Points to note**

- 1) This instruction uses a ‘data table’ which contains a single list of values which are to be selected and compared by two consecutive counters (S2 and S2+1). The data table is identified as having a



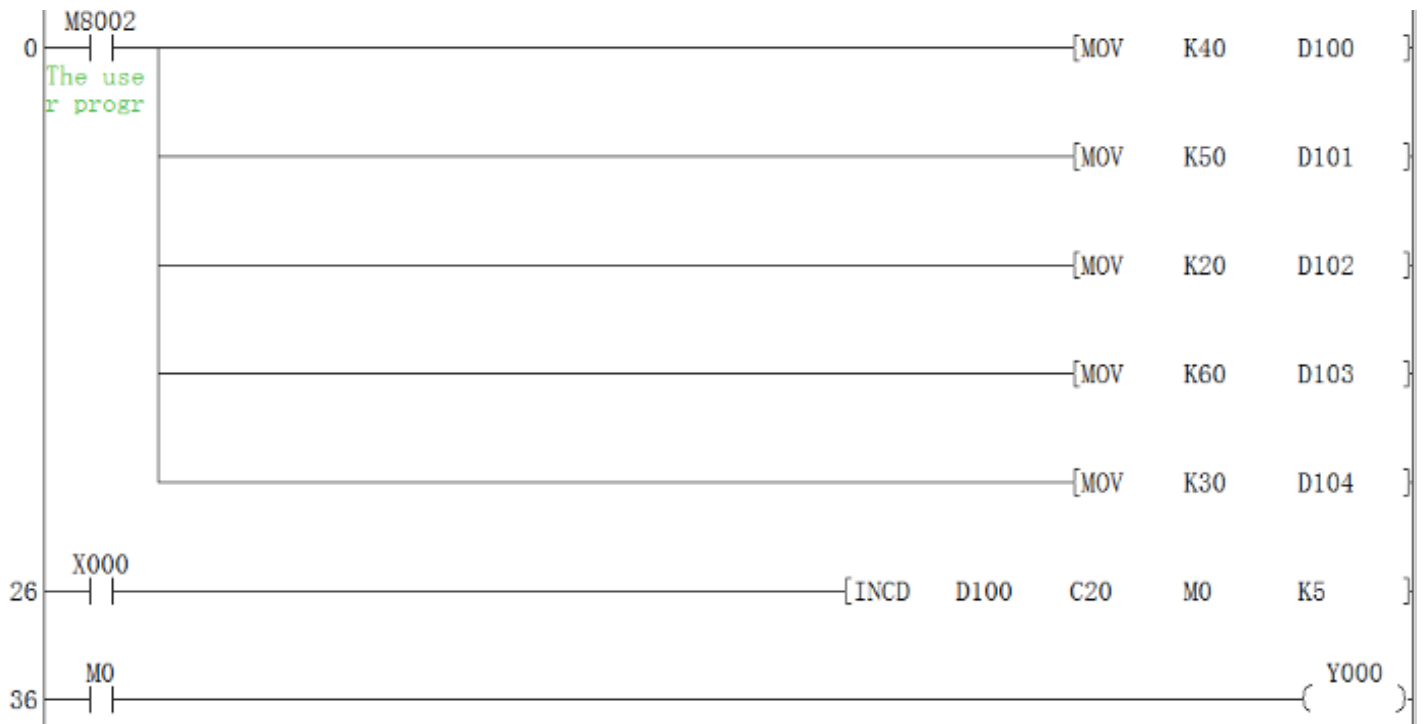
head address S1 and consists of n data elements.

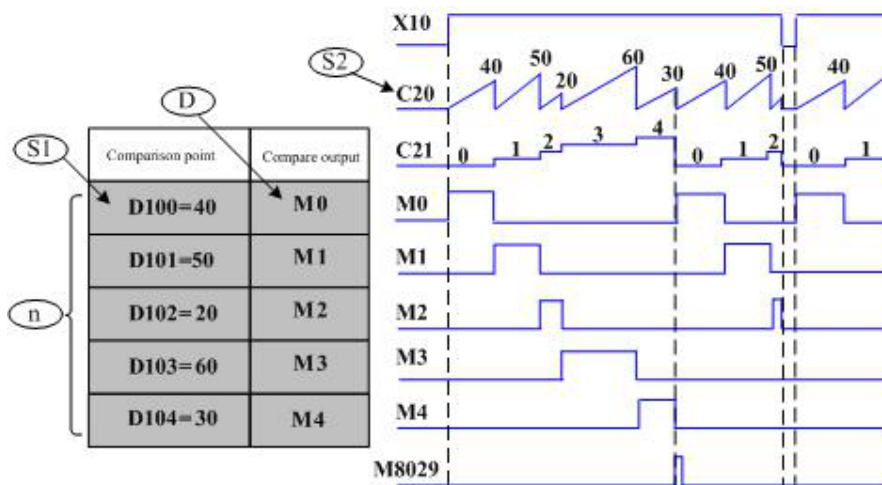
- 2) Counter S2 is programmed in a conventional way. The set value for counter S2 MUST be greater than any of the values entered into the data table. Counter S2 counts a user event and compares this to the value of the currently selected data element from the data table.

When the counter and data value are equal, S2 increments the count of counter S2+1 and resets its own current value to '0' (zero). This new value of counter S2+1 selects the new data element from the data table and counter S2 now compares against the new data elements value.

- 3) The counter S2+1 may have values from 0 to n. Once the nth data element has been processed, the operation complete flag M8029 is turned ON. This then automatically resets counter S2+1, hence, the cycle starts again with data element S1+0.
- 4) Values from 0 to 32,767 may be used in the data table.
- 5) The INCD instruction may only be used **ONCE** in a program.

**Program Example**





### 4.2.54 INT instruction

#### Instruction Description

Table 4-72

Name	Function	Devices		Format	Steps
		S	D		
INT (Float to Integer)	Converts a number from floating point format to decimal format	D - must be in floating point number format (always 32 bits).	D - decimal format for INT, INTP - 16 bits for DINT, DINTP - 32 bits		INT, INTP: 5 steps DINT, DINTP: 9 steps

**Operation**

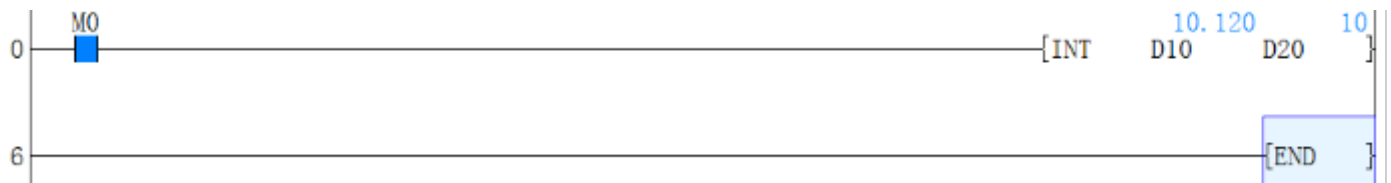
The floating point value of S is rounded down to the nearest integer value and stored in normal binary format in D.

**Points to note**

- 1) The source data is always a double (32 bit) word; a floating point value. For single word (16 bit) operation the destination is a 16 bit value. For double word (32 bit) operation the destination is a 32 bit value.
- 2) This instruction is the inverse of the FLT instruction.

**Program example**

- 1) Example 1(16-bit mode)



- 2) Example 2 (32-bit mode)



**4.2.55 IST instruction**

**Instruction description**

Table 4-73

Name	Function	Devices			Format	Steps
		S	D1	D2		
IST	Automatically sets up a multi-mode STL operating system	X, Y, M, S, Note: uses 8 consecutive devices	S, Note: TP users S20 to S899			IST: 7 steps

**Operation**

This instruction can be used to initialize the control status of a typical multi-action looping execution

mechanism and to specify parameters for the operation mode such as the input signal, action status, etc.

Where:

(S) is the component address of the starting bit variable of the input of the specified operation mode. It occupies 8 continuous address units from (S) ~ (S)+7. The special function definition for each variable is described below:

(D1) is the minimum serial number using the S status in the specified automatic operation mode.

(D2) is the maximum serial number using the S status in the specified automatic operation mode. (D1)

to (D2) are the status serial numbers of the looping action of the control system, which determine the status numbers.

For example, in the system below, the execution mechanism acts sequentially in such a way: the grabbing device drops to the position of work piece A from the base point to grab the work piece, and then it lifts the work piece to the specified height and translates to the desired position and drops. After arriving at the required position, it releases the work piece and back tracks to start the next looping action. It is possible to use the IST instruction to specify the control signal input, the control of the status transferring, etc. of the operational mechanism to achieve automatic control. In addition, it supports manual commissioning of single-step actions and base point reset, etc.

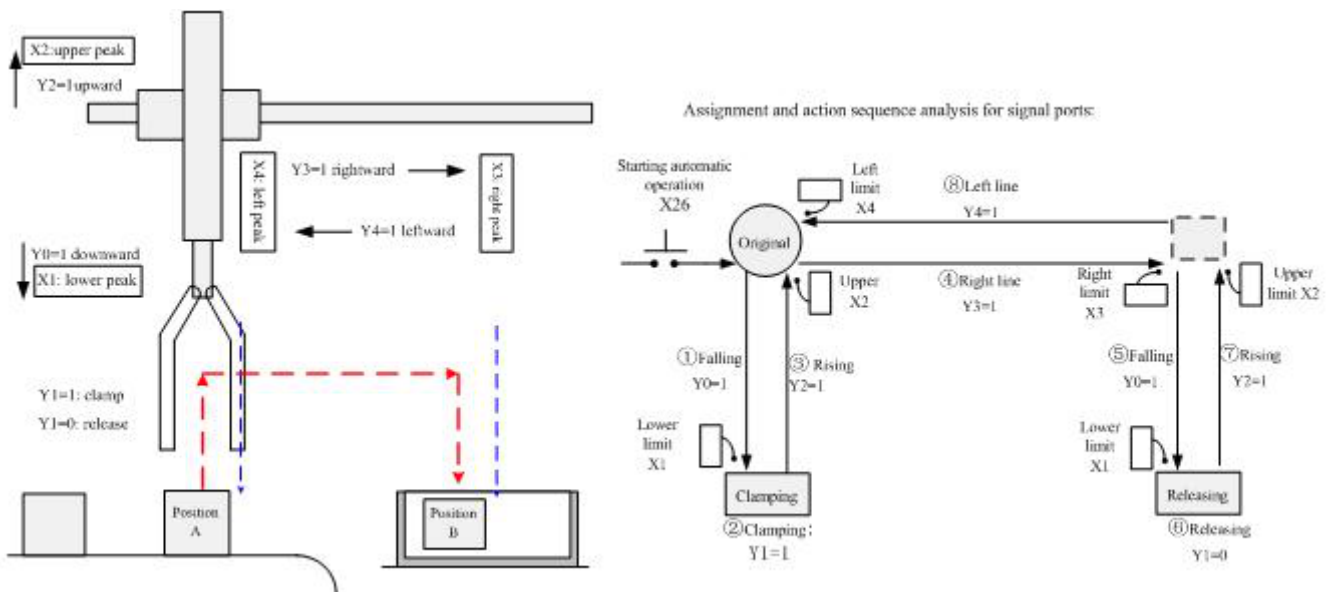


Figure 4-3

Instruction keys and status changing switches are required to control the operational mechanism using manual commissioning, single actions, and looping actions, etc. The following is a schematic diagram of the operation panel, including the key ports and their function assignments:

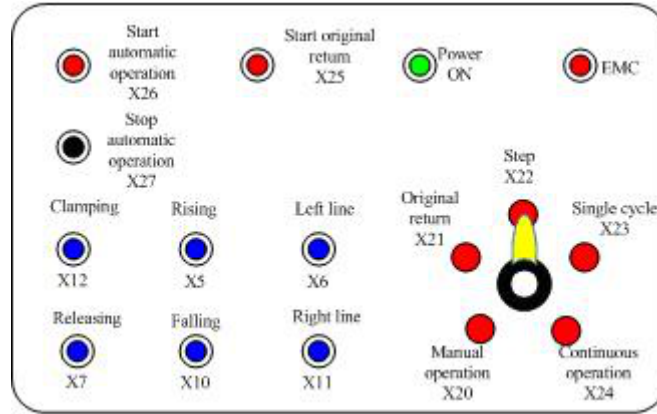
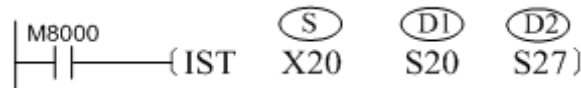


Figure 4-4

For applications like the above diagram, each complete cycle can be divided into 8 steps (i.e. 8 statuses). The following instruction clauses can be used to initialize the status of the control system:



**S** specifies X20 as the starting input of the operation mode. Therefore, the input ports X21 to X27 of the subsequent addresses will also be used. The functional action features will be defined respectively as: (it is similar for variables X, M, or Y)

X20: This is the manual operation mode to switch on/off the various control output signals using a single button.

X21: This is the base point reset mode to reset the device to the base point by pressing the base point reset button.

X22: This is the single-step operation mode to step forward a process each time the starting button is pressed.

X23: This is the one-cycle looping mode. When the start button is pressed, it will run the one-cycle looping automatically and stop at the base point. The operation can be stopped by pressing the stop button. Then, if the start button is pressed, the operation will continue and stop at the base point automatically.

X24: This is the continuous operation mode to run continuously by pressing the start button. When the stop button is pressed, it will move to the base point and stop.

X25: To start the base point rest command signal.

X26: To start the automatic command signal.

X27: To stop the automatic command signal.

**Note:**

In these port signals, the operation mode is determined by X20 to X24, for which the statuses can't be ON at the same time. Therefore, it is suggested to use rotary switches for the selection and switching of the signals.

**D1** and **D2** are used to specify the minimum and maximum serial number S20 to S27 of the service

statuses (8 for total) in the automatic operation mode. The following special variables for the definition and use requirements of the IST instruction should be noted:

When driving the IST instruction, the control of the following components will be automatically switched and can be referenced by user programs. In order to make the status switching and control of the IST instruction cooperate, the operation of certain special variables is required in the user programs. See the description in the table below:

Table 4-74

Default variables in IST instruction		Variables driven in user program	
M8040	1=disable transfer of all states	M8043	1=original return completed. In the original return mode, after a machine returns to original, the special M variable will be set by the user program.
M8041	1=transfer start	M8044	1=original condition. Detect the original condition of a machine and drive the special assistant relay. It is set in all modes.
M8042	1=start pulse	M8045	1=all output reset disabled. If a machine is switched among manual, return and automatic modes when it is not at original, all output and action states will be reset. But only action status can be reset if M8045 has been driven.
S0	Manual operation initial state	M8047	1=STL monitoring valid. After M8047 has been driven, the state number of action (S0 ~ S99) will be saved in the special assistant relay D8040 ~ D8047 in ascendent order, thus monitoring action state numbers of 8 points. In addition, if any of these states is enabled, the special assistant relay M8046 will act.
S1	Original return initial state		
S2	Automatic operation initial state		

Under the "automatic operation" mode, free conversion is possible between: single step<-->one-cycle looping<-->continuous operation.

When performing conversion between "manual operation" <--> "base point reset" <--> "automatic operation" while the machine is running, the switched mode is effective after all the outputs are reset. (Reset is not applicable for M8045 drive.)

S10 to S19 can be used for the base point reset when using the IST instruction. Therefore, don't use these statuses as common statuses. In addition; S0 to S9 are used for the initial status process, S0 to S2, as mentioned in the above manual operations, are used for the base point reset and automatic operation, and S3 to S9 can be used freely.

When programming, the IST instruction must be programmed with a higher priority than the various STL circuit, such as status S0 to S2, etc.

Rotary switches must be used to avoid the situation that X20 to X24 are ON at the same time.

When switching between each (X20), base point reset (X21), auto (X22, X23, X24) before the base point completion signal (M8043) is activated, all the outputs are switched OFF. And the automatic operation can't drive again until the base point reset is finished.

After initialization of the control instruction using the IST instruction, the action of each status of the execution mechanism and the conditions for status transferring need to be programmed, as detailed below:

- 1) System initialization: defines the conditions for base point reset and defines the input ports of the operation mode signals used in the IST instruction and the status variables of the looping actions. The program clauses used are illustrated in the following.

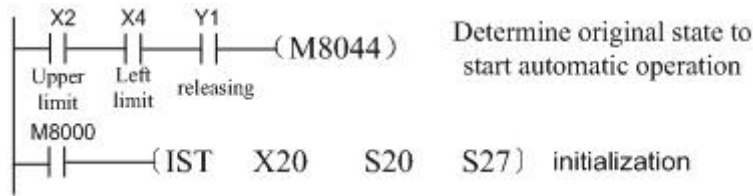


Figure 4-5

- 2) Manual operation: driven to execute by the command signals defined on the operation plate. See the program clauses of status S0 in the following diagram. This part of the program can be skipped if there is no manual mode:

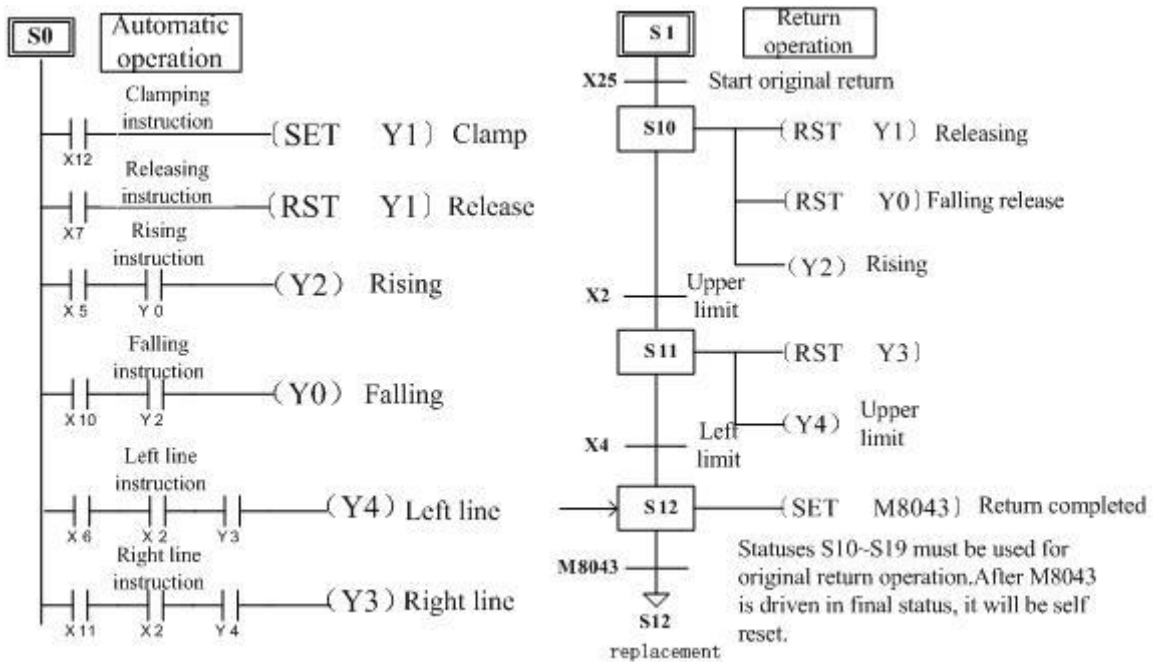
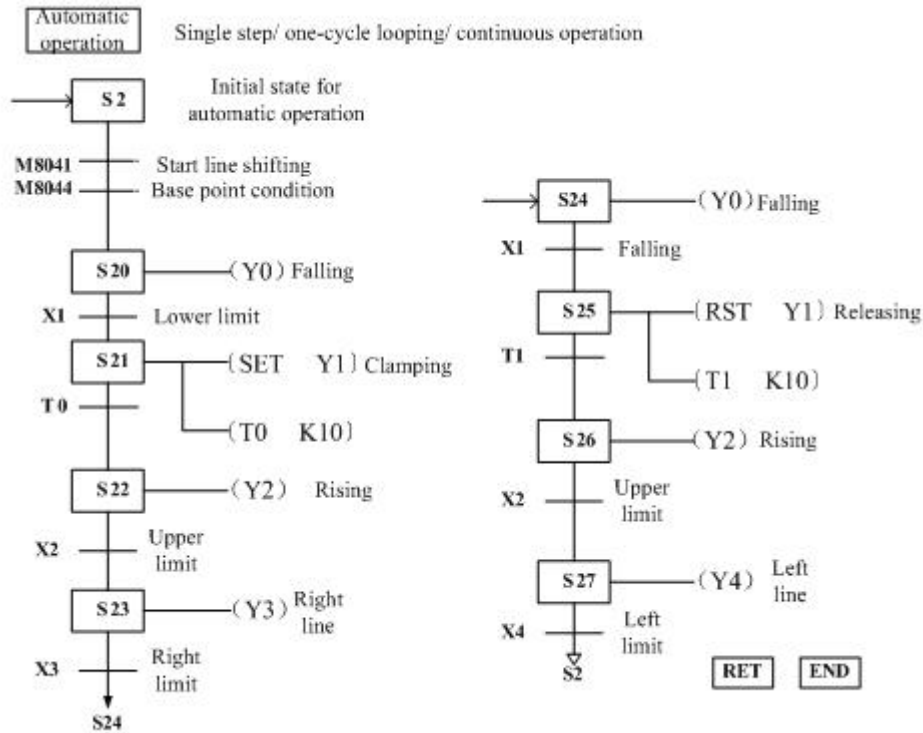


Figure 4-6

- 3) Base point reset: design reset program based on the command signal at the starting of the reset and the sequence of the reset actions, *as shown in the upper right:???????*
- 4) Automatic operation: write program based on the required action conditions and sequence and the control signal output, as shown in the diagram below:



Up to this point, the control system is allowed to complete the looping action according to the above mentioned action requirements. The above programming description uses step instructions for the convenience of reading, while the user is free to program using the equivalent ladder diagrams.

When different status numbers occur to the "automatic operation" mode in a control system, the above example can be referenced to program in modifying the setting items of (D1) and (D2) corresponding works need to be done in the "automatic operation" mode.

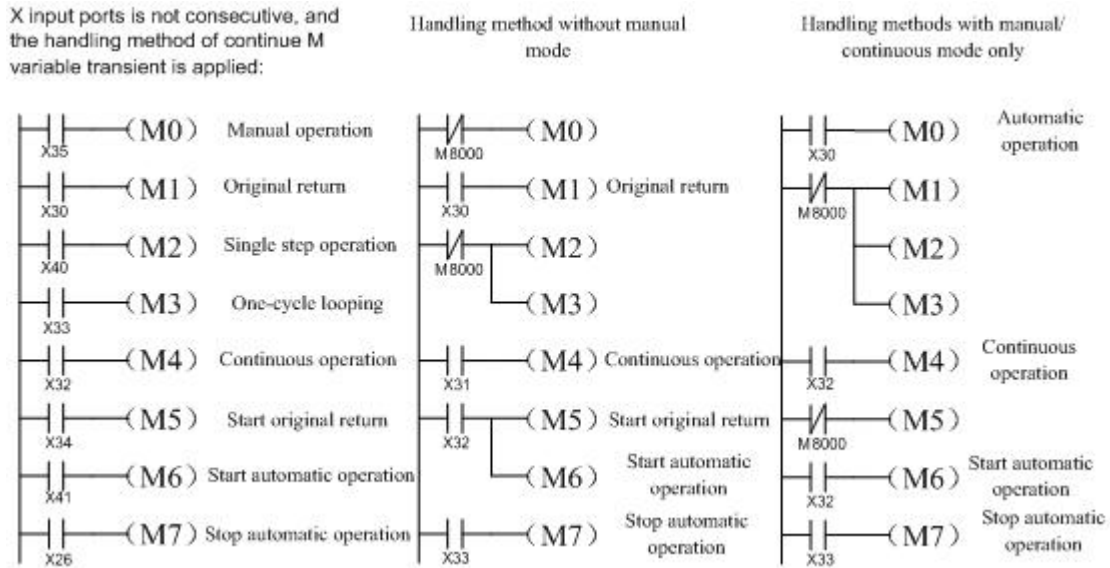
Handling methods for non-continuous X input:

If an X input port with non-continuous addresses needs to be used as the provided input of the operation mode, the M variable can be used for a "transitional" transmission. That is, the non-continuous X input status will be copied to an M variable with continuous addresses one by one using the simple OUT instruction rather than the instructions below:



Specific to the continuous M0 to M7 variable area in the IST, the programming instructions can be used to shield the non-existent control mode. For example, the corresponding relationship between X as the mode input end and the M variable in the following diagram. For un-required modes, you simply input the M variable and fix it to zero:





### 4.2.56 MEAN instruction

#### Instruction Description

Table 4-75

Name	Function	Devices			Format	Steps
		S	D	n		
MEAN (Mean)	Calculates the mean of a range of devices	KnX, KnY, KnM, KnS, T, C, D	KnY, KnM, KnS, T, C, D, V, Z	K, H, ) Note: n= 1 to 64		MEAN, MEANP: 7 steps DMEAN, DMEANP : 13steps

#### Operation

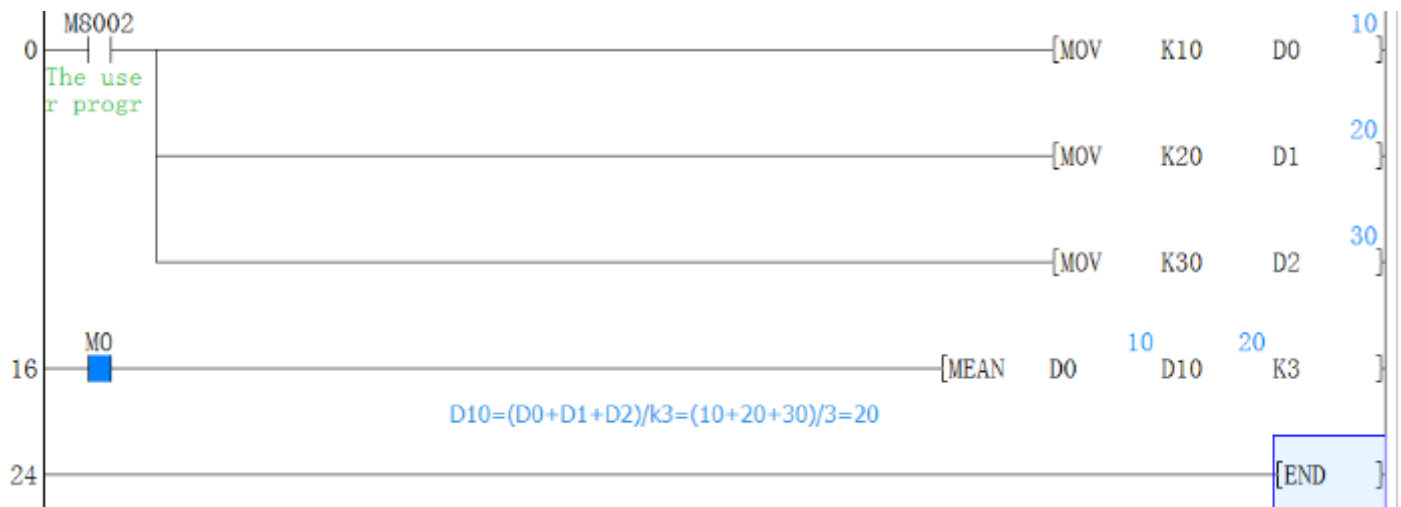
The range of source data is defined by operands Sand n. S is the head address of the source data and n specifies the number of consecutive source devices used. The value of all the devices within the source range is summed and then divided by the number of devices summed, i.e. n. This generates an integer mean value which is stored in the destination device (D). The remainder of the calculated mean is ignored.

#### Points to note

If the source area specified is actually smaller than the physically available area, then only the available devices are used. The actual value of n used to calculate the mean will reflect the used, available devices. However, the value for n which was entered into the instruction will still be displayed. This can cause confusion as the mean value calculated manually using this original n value will be different from that which is displayed.

If the value of n is specified outside of the stated range (1 to 64) an error is generated.

**Program example**



**4.2.57 MOV instruction**

**Instruction Description**

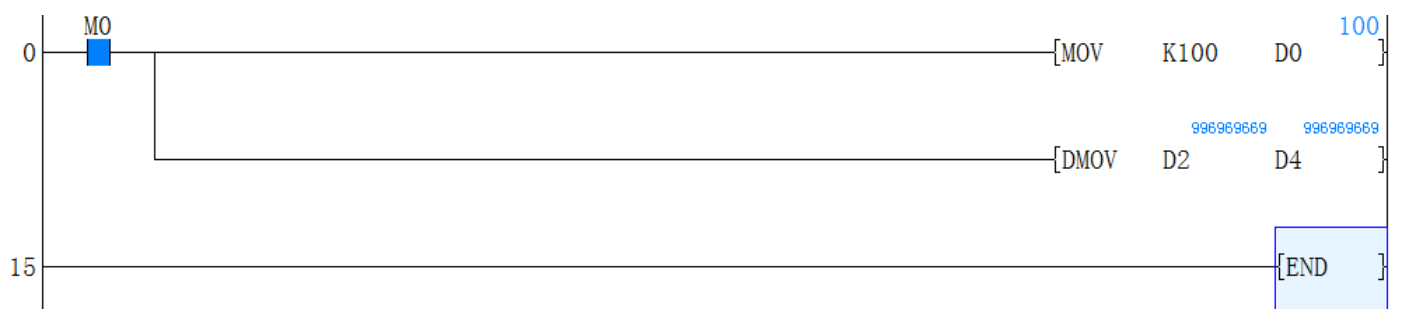
Table 4-76

Name	Function	Devices		Format	Steps
		S	D		
MOV (Move)	Moves data from one storage area to a new storage area	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z	$\left[ \begin{array}{c} X0 \\     \\ \text{---} \end{array} \right] \text{---} \left( \text{MOV} \begin{array}{c} S \\ H0050 \end{array} \begin{array}{c} D \\ D10 \end{array} \right) \text{---} \left[ \begin{array}{c}     \\ \text{---} \end{array} \right]$	MOV, MOVP: 5 steps DMOV, DMOVP: 9 steps

**Operation**

The content of the source device (S) is copied to the destination (D) device when the control input is active. If the MOV instruction is not driven, no operation takes place.

**Program example**



### 4.2.58 MTR instruction

#### Instruction Description

Table 4-77

Name	Function	Devices				Format	Steps
		S	D1	D2	n		
MTR (Input matrix)	Multiplexes a bank of inputs into a number of sets of devices. Can only be used ONCE	X	Y	Y, M, S	K, H, ) Note: n=2 to 8		MTR: 9 steps

#### Operation

This instruction allows a selection of 8 consecutive input devices (head address S) to be used multiple (n) times, i.e. each physical input has more than one, separate and quite different (D1) signal being processed. The result is stored in a matrix-table (head address D2).

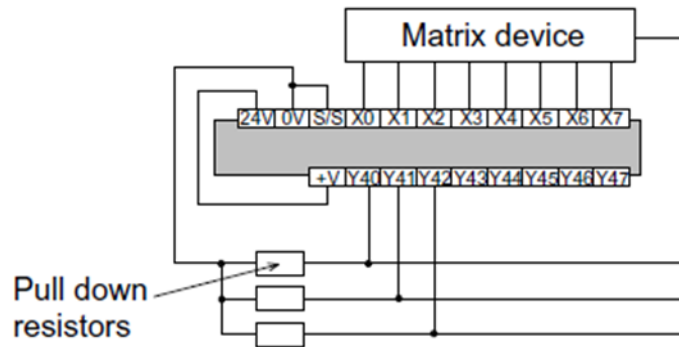
#### Points to note

- 1) The MTR instruction involves high speed input/output switching. For this reason this instruction is only recommended for use with transistor output modules.
- 2) For the MTR instruction to operate correctly, it must be driven continuously. It is recommended that special auxiliary relay M8000, the PLC RUN status flag, is used. After the completion of the first full reading of the matrix, operation complete flag M8029 is turned ON. This flag is automatically reset when the MTR instruction is turned OFF.
- 3) Each set of 8 input signals are grouped into a 'bank' (there are n number of banks).
- 4) Each bank is triggered/selected by a dedicated output (head address D1). This means the quantity of outputs from D1, used to achieve the matrix are equal to the number of banks n. As there are now additional inputs entering the PLC. Each of these will have a status which needs recording. This is stored in a matrix-table. The matrix-table starts at the head address D2. The matrix construction mimics the same 8 signal by n bank configuration. Hence, when a certain input in a selected bank is read, its status is stored in an equivalent position within the result matrix-table.
- 5) The matrix instruction operates on an interrupt format, processing each bank of inputs every 20msec. This time is based on the selected input filters being set at 10msec. This would result in an 8 bank matrix, i.e. 64 inputs (8 inputs' 8 banks) being read in 160msec.

If high speed input (ex. X0) is specified for operand S, the reading time of each bank becomes only 10msec, i.e. a halving of the reading speed. However, additional pull down resistors are required on the

drive outputs to ensure the high speed reading does not detect any residual currents from the last operation.

These should be placed in parallel to the input bank and should be of a value of approximately 3.3kΩ, 0.5W. For easier use, high speed inputs should not be specified at S.



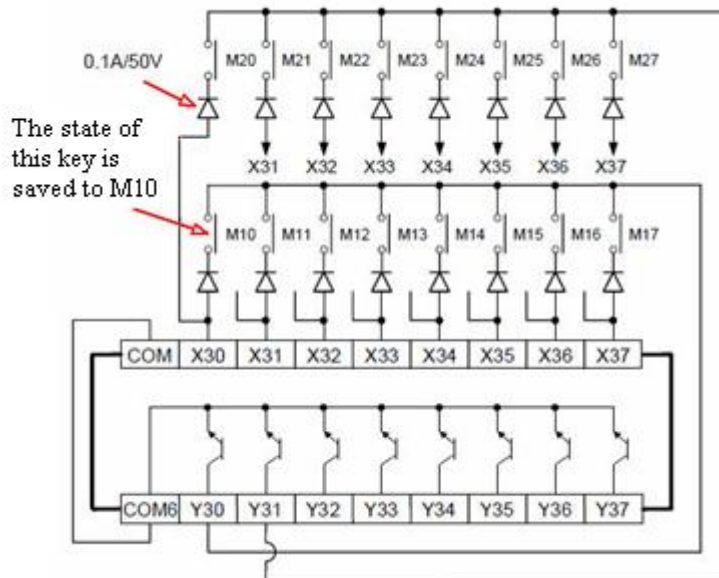
- 6) f) Because this instruction uses a series of multiplexed signals it requires a certain amount of hard wiring' to operate. The example wiring diagram to the right depicts the circuit used if the previous example instruction was programmed. As a general precaution to aid successful operation diodes should be placed after each input device (see diagram opposite). These should have a rating of 0.1A, 50V.

Notice how the resulting matrix-table does not use any of the P8 and P9 bit devices when state S or auxiliary M relays are used as the storage medium.

**Program example**

```

M8000
| | | (MTR X30 Y30 M10 K2 )
    
```



**Example Operation**

When output Y30 is ON only those inputs in the first bank are read. These results are then stored; in this

example, auxiliary coils M10 to M17. The second step involves Y30 going OFF and Y31 coming ON. This time only inputs in the second bank are read. These results are stored in devices M20 to M27. The last step of this example has Y31 going OFF and Y32 coming ON. This then allows all of the inputs in the second bank to be read and stored in devices M20 to M27. The processing of this instruction example would take  $20 \times 2 = 40$  msec.

A scanning input with a maximum of 64 points can be achieved using 8-point X output and 8-point transistor Y output. But it is not suitable for high speed input operations because it needs a time of 20ms, 8 line = 160ms to read each input. Therefore, the ports after X20 are typically used as the scanning inputs. This instruction is allowed to be used only once in the program.

### 4.2.59 MUL instruction

#### Instruction Description

Table 4-78

Name	Function	Devices			Format	Steps
		S1	S2	D		
MUL (Multiplication)	Moves data from one storage area to a new storage area	K, H, KnX, KnY, KnS, T, C, D, V, Z	KnX, KnM	KnY, KnM, T, C, D, Z(V) Note: Z(V) may NOT be used for 32 bit operation		MUL, MULP: 7 steps DMUL, DMULP: 13 steps

#### Operation

(Applicable to all units) the contents of the two source devices (S1, S2) are multiplied together and the result is stored at the destination device (D). Note the normal rules of algebra apply.

#### Points to note

- 1) When operating the MUL instruction in 16bit mode, two 16 bit data sources are multiplied together. They produce a 32 bit result. The device identified as the destination address is the lower of the two devices used to store the 32 bit result. Using the above example with some test data:  
(D0) × 7 (D2) = 35 - The value 35 is stored in (D4, D5) as a single 32 bit word.
- 2) When operating the MUL instruction in 32 bit mode, two 32 bit data sources are multiplied together. They produce a 64 bit result. The device identified as the destination address is the lower of the four devices used to store the 64 bit result.
- 3) If the location of the destination device is smaller than the obtained result, then only the portion of the result which directly maps to the destination area will be written, i.e. if a result of 72 (decimal) is to be stored at K1Y4 then only Y7 would be active. In binary terms this is equivalent to a

decimal value of 8, a long way short of the real result of 72!

**Program example**



**4.2.60 NEG instruction**

**Instruction Description**

Table 4-79

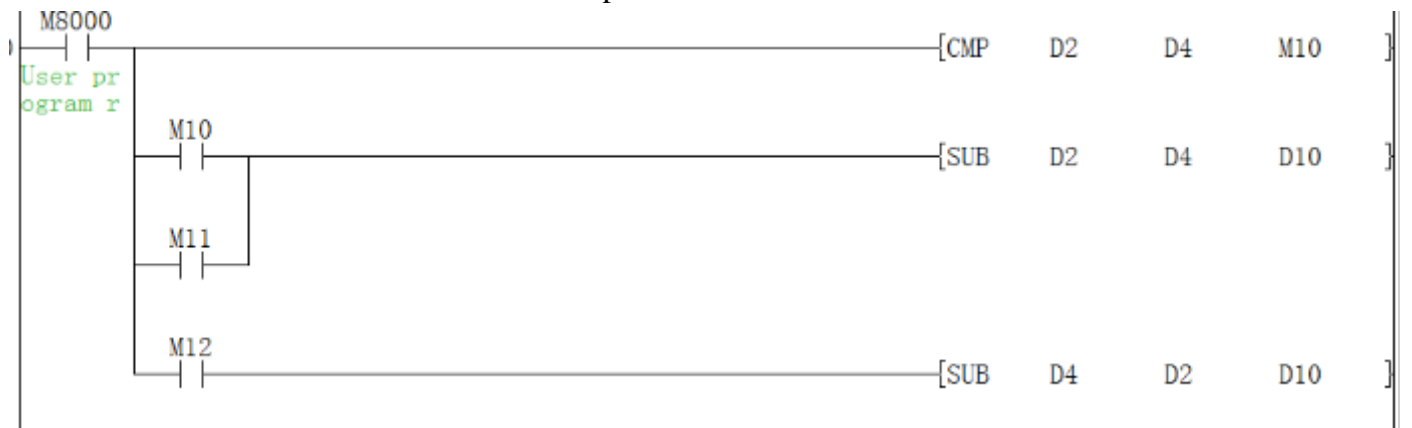
Name	Function	Devices	Format	Steps
		D		
NEG (Negation)	Logically inverts the contents of the designated device	KnY, KnM, KnS, T, C, D, V, Z		NEG, NEG GP: 3 steps DNEG, DNEG GP: 5 steps

**Operation**

The bit pattern of the selected device is inverted. This means any occurrence of a '1' becomes a '0' and any occurrence of a '0' will be written as a '1'. When this is complete, a further binary 1 is added to the bit pattern. The result is the total logical sign change of the selected devices contents, e.g. a positive number will become a negative number or a negative number will become a positive.

**Program example**

Take the absolute value of a subtraction operation



If D2>D4, then M10=On; if D2=D4, then M11=On; and if D2<D4, then M12=On. By this way it can

be ensured that the value of D10 is always positive. This program can be illustrated by the following process flow.

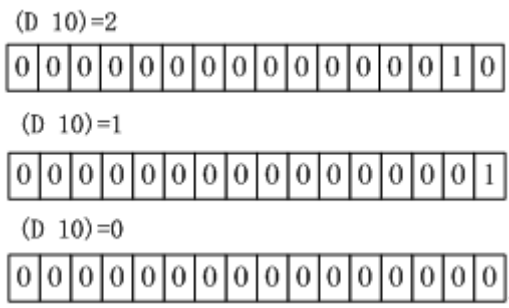


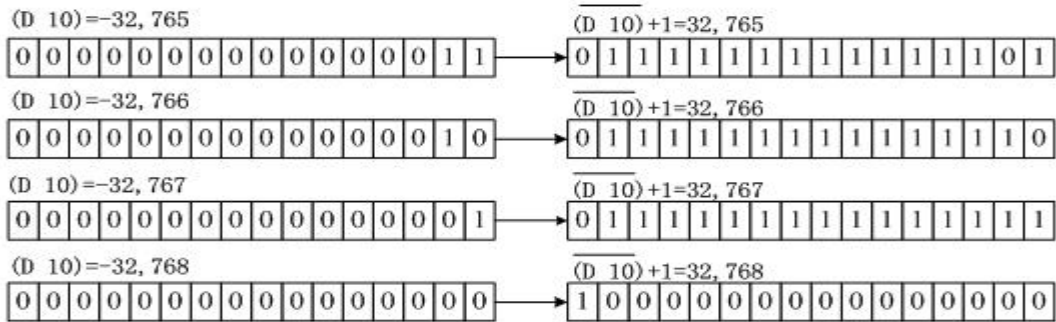
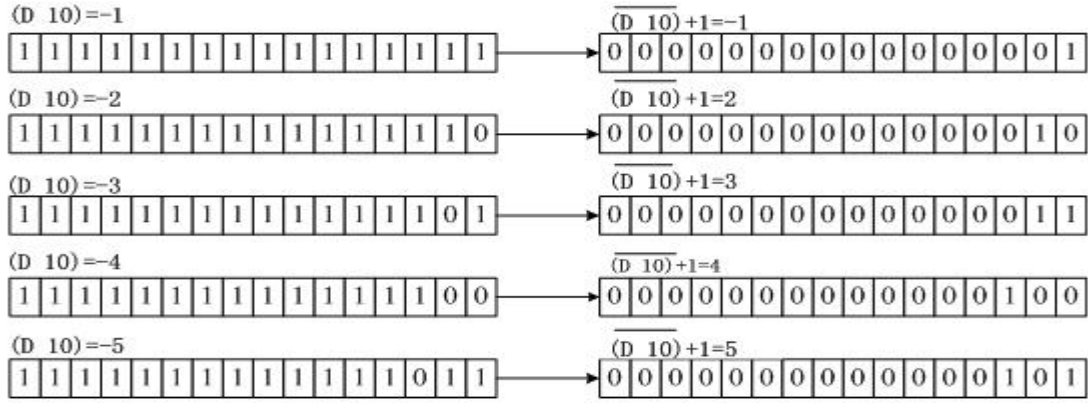
When bit15 of D10 is "1" (indicates that D10 is negative), M10=on. The complemented value of D10 by instruction NEG will be the absolute value of D10.

In both examples above, the result for D10 is K4 if D2=K4, D4= K8 or D2=K8, D4=K4.

**Additional remarks**

- 1) If a number is positive or negative, it is indicated by the value of the highest bit (the leftmost), "0" for positive and "1" for negative.
- 2) If the value of the highest bit is 1, the NEG instruction can be used to convert it into the absolute value.





Maximum absolute value can only be 32,767.

### 4.2.61 PID instruction

#### Instruction description

Table 4-80

Name	Function	Devices				Format	Steps
		S1	S2	S3	D		
PID (PID control loop) register each	Receives a data input and calculates a corrective action to a specified level based on PID control	D) Note: S1 and S2 use a single data register	D) Note: S3 uses 25 consecutive data registers	D) Note: D uses a single data register		PID: 9 steps	



**Operation**

This instruction takes a current value (S<sub>2</sub>) and compares it to a predefined set value (S<sub>1</sub>). The difference or error between the two values is then processed through a PID loop to produce a correction factor which also takes into account previous iterations and trends of the calculated error. The PID process calculates a correction factor which is applied to the current output value and stored as a corrected output value in destination device (D). The setup parameters for the PID control loop are stored in 9 consecutive data registers S<sub>3+0</sub> through S<sub>3+8</sub>.

Table 4-81

		<b>Operation parameter(S<sub>3+N</sub>)</b>
<b>Unit</b>	<b>Function</b>	<b>Description</b>
S3	Sample time(Ts)	Setting range 1~32767(ms), but must longer than scanning cycle of plc program
S3+1	Reaction direction(ACT)	bit0: 0=positive action; 1=negative action; bit3: 0=one way; 1=two way; bit4: 0=disable self-tuning; 1=enable self-tuning; Others cannot be used.
S3+2	Maximum climbing(Delta T)	Setting range 0~320
S3+3	Proportional gain(Kp)	Setting range: 0~32767, note:this value is magnified 256 times, actual value is Kp/256
S3+4	Integral gain(Ki)	Setting range: 0~32767, Ki=16384Ts/Ti, Ti is integral time
S3+5	Derivative gain(Kd)	Setting time: 0~32767, Kd≈Td/Ts, Td is derivative time
S3+6	Filter (C0)	Range: 0~1024
S3+7	Output lower limit	Recommended range: -2000~2000, when S3+1 bit3=0, please set 0; S3+1 bit3=1, please set -2000
S3+8	Output upper limit	Recommended values: 2000

**Program example**

1) Set the target value when PLC is running.



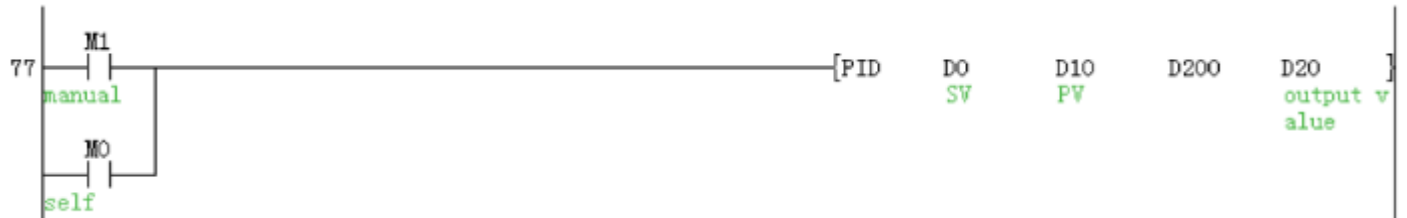
2) Use M0 for trigger self-tuning mode, set the D201 (S3+1) is H10 (10000), and set other parameters.



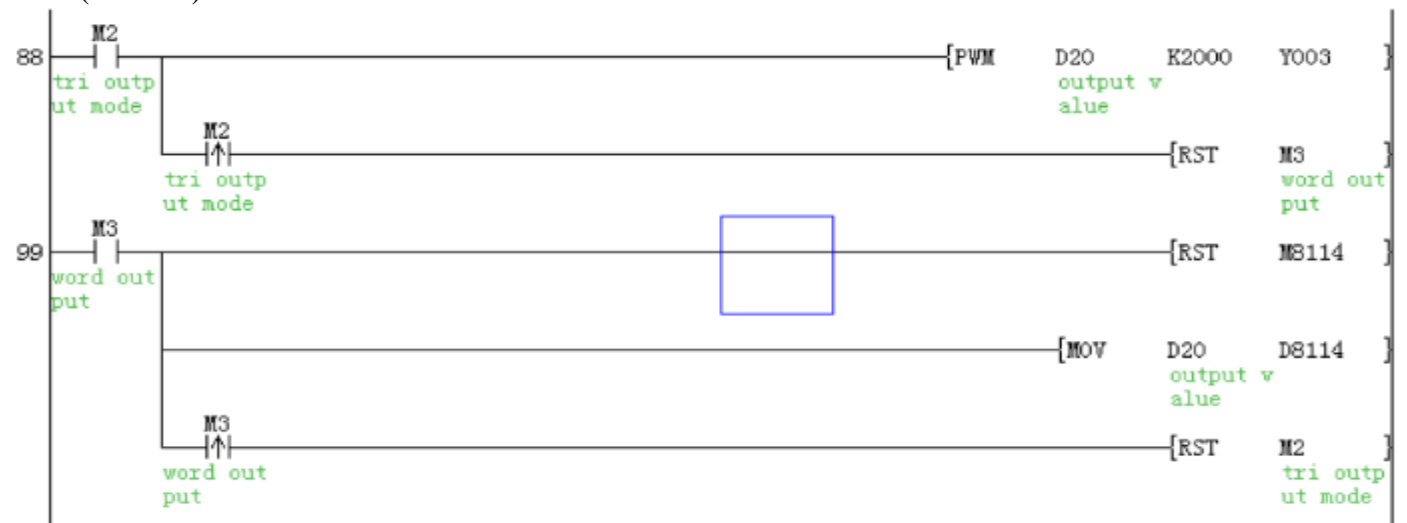
3) Use M1 for trigger manual-tuning mode, set the D201 (S3+1) is H00 (00000)



4) Trigger PID command by M0 or M1.



5) Use M2 for bit output, and use M3 for word output, in this project, it uses 2AD2DA-BD for analog output (4-20mA)



**Error code**

Controlling the setting value of parameter or the error occur on the data of PID algorithm, the algorithm error M8067 turn into ON status and save the error content D8067 into below data.

Table 4-82

Code	Error content	Process state	Processing method
K6705	Operand of application instruction outside of target soft element	PID order to stop algorithm	Pls confirm the content of control data
K6706	Operand of application instruction outside of target soft element		
K6730	(TS< 0 ) Sampling time(TS) outside of target soft element (TS< 0 )		
K6732	filter(C0)outside(C0<0 or 1024≤C0)		
K6732	Maximum rate of raise(DeltaT) outside ΔT<0 or 320≤ΔT		
K6733	Proportional gain(KP) outside of target range		
K6734	Integral gain (KI)out side of target range(KI<0)		
K6735	Derivative gain outside of target range (KD<0)		
K6740	Sampling time≤ algorithm cycle	PID order to go on operating	
K6742	Variation of measured value exceed ((PV<-32768 or 32767 < ( PV )		
K6751	Direction of Self-tuning isn't match	Self-tuning carry on	It is not match between the action direction estimated by estimated value while self-tuning start and the actual action direction while self-tuning output. pls correct the target value, self-tuning output, estimated value, then self-tuning.
K7652	Self-tuning action is improper	Self-tuning	Cannot make the proper action due to the variation of estimated value of self-tuning, pls set

			the sampling time longer much than the output change cycle, increase the output filter constant. changes set, then restart self-tuning
--	--	--	--

**Program instruction**

- 1) PID instruction can be used multiple times and executed at the same time, but variable area of PID instruction cannot overlap it also can be used in step instruction, jump instruction, timer interruption, subroutine, but pls note need to delete 9 cache unit before execute PID instruction.
- 2) The maximum error of sampling time TS is  $-(1 \text{ execution cycle} + 1 \text{ ms}) \sim +(1 \text{ execution cycle})$ . If sampling time  $TS \leq 1 \text{ execution cycle OF PLC}$ , then will have below PID operational error (K6740) ,and execute PID algorithm as  $TS = \text{execution cycle}$ , in that case, it is better to use constant scanning mode or use the PID instruction in timer interrupt (16□□~ 18□□)

**4.2.62 PLSR instruction**

**Instruction Description**

Table 4-83

Name	Function	Devices				Format	Steps
		S1	S2	S3	D		
PLSR (Pulse ramp)	Outputs a specified number of pulses, ramping up to a set frequency and back down to stop	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z Note: S1 S2			Y Y0--Y3.		PLSR: 9 steps DPLSR: 17 steps

**Operation**

A specified quantity of pulses S2 is output through device D. The output frequency is first ramped up to the maximum frequency S1 in acceleration time S3 ms, Then ramped down to stop also in S3 ms. This instruction is used to generate simple ACC/DEC curves where the quantity of outputs is of primary concern.

**Points to note**

- 1) TP -4H type: Y0 and Y1 max up to 200KHZ, Y2 and Y3 max up to 100KHZ, But Y0+Y1+Y2+Y3 total frequencies cannot beyond 400KHZ  
TP1-4H/TPE-4H type Y0, Y1, Y2, Y3 both support max up to 200KHZ in every Y0 to Y3 channels.

- 2) The maximum number of pulses: 16 bit operation: 1 to 32,767 pulses, 32 bit operation: 1 to 2,147,483,647 pulses.

Note: special auxiliary coil M8029 is turned ON when the specified number of pulses has been completed. The pulse count and completion flag (M8029) are reset when the PLSY instruction is de-energized. If "0" (zero) is specified the PLSY instruction will continue generating pulses for as long as the instruction is energized.

- 3) A single pulse is described as having a 50% duty cycle. This means it is ON for 50% of the pulse and consequently OFF for the remaining 50% of the pulse. The actual output is controlled by interrupt handling, i.e. the output cycle is NOT affected by the scan time of the program.
- 4) The data in operands S<sub>1</sub> and S<sub>2</sub> may be changed during execution. However, the new data in S<sub>2</sub> will not become effective until the current operation has been completed, i.e. The instruction has been reset by removal of the drive contact.
- 5) Two PLSY can be used at the same time in a program to output pulses to Y000 and Y001 and Y002 and Y003 respectively. Or, only one PLSY and one PLSR can be used together in the active program at once, again outputting independent pulses to Y000 and Y001 and Y002 and Y003.

It is possible to use subroutines or other such programming techniques to isolate different instances of this instructions. In this case, the current instruction must be deactivated before changing to the new instance.

- 6) Because of the nature of the high speed output, transistor output units should be used with this instruction. Relay outputs will suffer from a greatly reduced life and will cause false outputs to occur due to the mechanical 'bounce' of the contacts.
- 7) TP PLC can monitor the number of pulses when exporting to [Y000], the current pulse register value is [D8141 (high byte), D8140 (low byte)] (in 32-bit).

When exporting to [Y001], the current pulse register value is [D8143 (high byte), D8142 (low byte)] (in 32-bit).

When exporting to [Y002], the current pulse register value is [D8151 (high byte), D8150 (low byte)] (in 32-bit).

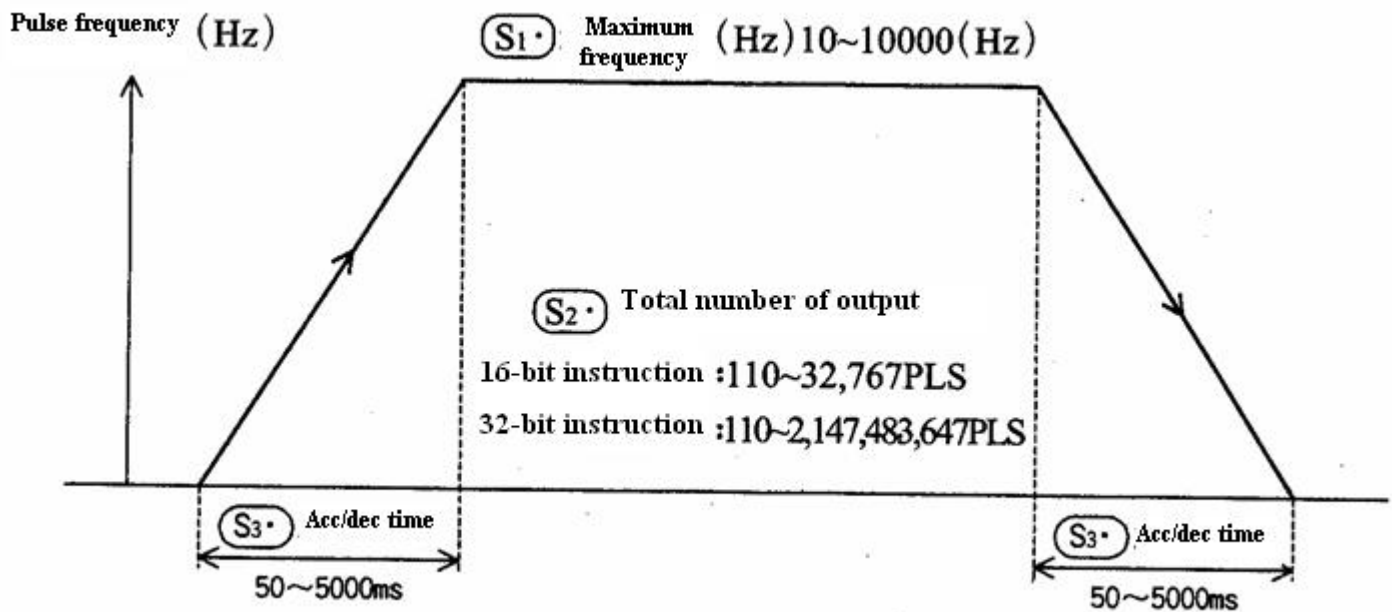
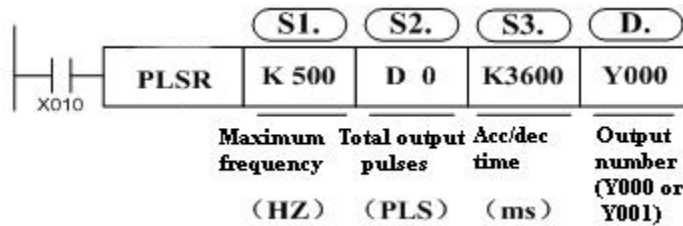
When exporting to [Y003], the current pulse register value is [D8153 (high byte), D8152 (low byte)] (in 32-bit).

- 8) The total number of pulses output can be monitored using D8136 and D8137.

The minimum value of output pulse frequency which can be actually used is determined by the following equation

$$\sqrt{S1HZ \div (2 \times (S3ms \div 1000))}$$

Program example



The special registers corresponding to each output port are listed as follow:

Table 4-84

Register		Definition	Remarks
D8140	Lower byte	Number of total pulses output to Y0 port set in the PLSY or PLSR instruction	Applicable instructions: use DMOV K0 D81xx to perform clear operation
D8141	Upper byte		
D8142	Lower byte	Number of total pulses output to Y1 port set in the PLSY or PLSR instruction	
D8143	Upper byte		
D8150	Lower byte	Number of total pulses output to Y2 port set in the PLSY or PLSR instruction	
D8151	Upper byte		
D8152	Lower byte	Number of total pulses output to Y3 port set in the PLSY or PLSR instruction	
D8153	Upper byte		
D8136	Lower byte	Accumulative value of the number of the pulses already output to Y0 and Y1	
D8137	Upper byte		

### 4.2.63 PLSV instruction

#### Instruction Description

Table 4-85

Name	Function	Devices			Format	Steps
		S	D1	D 2		
PLSV (Pulse V	Variable speed pulse output	K,H, KnX,KnY , KnM,KnS T,C,D,V,Z	Y Y0-Y 3.	Y,M,S		PLSV 9 steps DPLS V 17 steps

#### Operation

This is a variable speed output pulse instruction, with a rotation direction output. [S] is the Pulse Frequency, [D1] is the Pulse Output Designation, and [D2] is the Rotation Direction Signal.

#### Points to note

- TP -4H type: Y0 and Y1 max up to 200KHZ, Y2 and Y3 max up to 100KHZ, But Y0+Y1+Y2+Y3 total frequencies cannot beyond 400KHZ  
TP1-4H/TPE-4H type Y0, Y1, Y2, Y3 both support max up to 200KHZ in every Y0 to Y3 channels.
- Only Y000 or Y001 and Y002 and Y003 can be used for the pulse output [D1]. Because of the nature of the high speed output, transistor type output units should be used with this instruction. Relay type outputs will suffer a greatly reduced life, and will cause false outputs to occur.
- Rotation direction signal output [D2] operated as follows: if [D2] = OFF, rotation = negative, if [D2] = ON, rotation = positive.
- The pulse frequency [S] can be changed even when pulses are being output.
- Acceleration/deceleration is not performed at start/stop. If cushion start/stop is required, increase or decrease the output pulse frequency [S] using the RAMP instruction.
- If the instruction drives contact turns off while pulses are output, the machine stops without deceleration
- Once the instruction drive contact is off, re-drive of the instruction is not possible while the pulse output flag (Y000 : [M8147] Y001 : [M8148]) is ON. Y002 M8149 is ON, Y003 M8150 is ON.
- The normal or reverse direction is specified by the positive or negative sign of the output pulse frequency [S].
- Related device numbers.  
D8141 (the upper byte), D8140 (the lower byte): Y000 represents the number of the output pulses decreasing when reversing. (32-bit)  
D8143 (the upper byte), D8142 (the lower byte): Y001 represents the number of the output pulses decreasing when reversing. (32-bit)  
D8151 (the upper byte), D8150 (the lower byte): Y002 represents the number of the output pulses



decreasing when reversing. (32-bit)

D8153 (the upper byte), D8152 (the lower byte): Y003 represents the number of the output pulses decreasing when reversing. (32-bit)

M8145:Y000 represents the pulse output stopped (instantly)

M8146:Y001 represents the pulse output stopped (instantly)

M8152:Y002 represents the pulse output stopped (instantly)

M8153:Y003 represents the pulse output stopped (instantly)

M8147:Y000 represents monitoring during the pulse output process (BUSY/READY)

M8148:Y001 represents monitoring during the pulse output process (BUSY/READY)

M8149:Y002 represents monitoring during the pulse output process (BUSY/READY)

M8150:Y003 represents monitoring during the pulse output process (BUSY/READY)

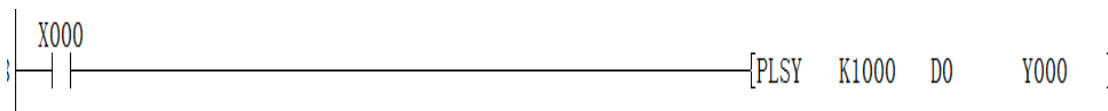
### 4.2.64 PLSY instruction

#### Instruction description

Table 4-86

Name	Function	Devices			Format	Steps
		S1	S2	D		
PLSY	Outputs a specified number of pulses at a set frequency	KnX, KnY, KnM, KnS, T, C, D, V, Z		Y	$\begin{array}{c} X010 \\ \text{--- } \text{---} \end{array} \text{[PLSY } \begin{array}{c} S1 \\ K1000 \end{array} \begin{array}{c} S2 \\ D0 \end{array} \begin{array}{c} D \\ Y000 \end{array} \text{ ]}$	PLSY: 7 steps DPLSY :13 steps.

#### Program example



Output a specified quantity of pulses at a specified frequency.

**S1: specified frequency;**

**Setting range:** 16 bits instruction range from 1 to 32767 Hz, 32 bits instruction range from 1 to 20000 Hz

When users change the value of s1, the output frequency changes as well

**S2: the quantity of pulses;**

**Setting range:** 16 bits instruction ranges from 1 to 32767, 32 bits instruction ranges from 1 to 2147483647 pulses.

If user assigns 0 to s2, there is no limit for the output pulse.

The data in s2 may be changed during execution. The new data in s2 will not become effective until the current operation has been completed.

D: output port. For TP, you can use y0 to y3, for TPS, user can use y0 to y1.

For the TP series programmable logical controller, to ensure a high frequency output pulse, a nominal load current for the output transistor is needed.

In the program example, when X0 is OFF, the output becomes 0 too, when X0 become ON again it will react initially.

A single pulse is described as having a 50% duty cycle. This means it is ON for 50% of the pulse and OFF for the 50% of the pulse. The actual is controlled by interrupt handling, i.e. the output cycle is not affected by the scan time of the program.

The pulse completion flag (M8029) is set when the PLSY instruction is done.

**The related variable in the PLSY:**

- 1) D8141 (high byte), D8140 (low byte):Y000 the count of output pulse, when the direction is reverse, Y000 decrease.(32 bits)
- 2) D8143 (high byte), D8142 (low byte):Y001 the count of output pulse, when the direction is reverse, Y000 decrease.(32 bits)
- 3) D8151 (high byte), D8150 (low byte):Y002 the count of output pulse, when the direction is reverse,Y000 decrease.(32 bits)
- 4) D8153 (high byte), D8152 (low byte):Y003 the count of output pulse, when the direction is reverse,Y000 decrease.(32 bits)
- 5) M8145: Y000 stop output pulse (immediately)
- 6) M8146: Y001 stop output pulse (immediately)
- 7) M8152: Y002 stop output pulse (immediately)
- 8) M8153: Y003 stop output pulse (immediately)
- 9) M8147: Y000 monitor in the output pulse(BUSY/READY)
- 10) M8148: Y001 monitor in the output pulse(BUSY/READY)
- 11) M8149: Y002 monitor in the output pulse(BUSY/READY)
- 12) M8150: Y003 monitor in the output pulse(BUSY/READY)

**4.2.65 PR instruction**

**Description**

Table 4-87

Name	Function	Devices		Format	Steps
		S	D		
PR	Outputs ASCII data to items such as display units	T, C, D	Y	$\begin{array}{c} \text{X000} \\ \text{---   ---} \end{array} [\text{PR} \quad \begin{array}{c} \text{S} \\ \text{D30} \end{array} \quad \begin{array}{c} \text{D} \\ \text{Y000} \end{array} ]$	5

**Operation**

- 1) Source data (stored as ASCII values) is read byte by byte from the source data devices. Each byte

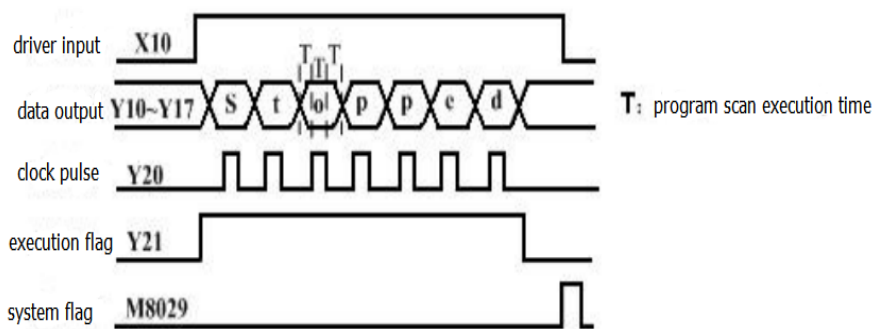
is mapped directly to the first 8 consecutive destination devices D0 to D7 and outputted by the output port Y.

- 2) S is the head address of the source data.
- 3) D is the head number of the output port.

**Program Example**



If the value stored in d200 to d203 is "Stopped", the corresponding output port and a timing signal are as shown below.



**Instructions for use**

- 1) This instruction should only be used on transistor output PLC.
- 2) The PR instruction will not automatically repeat its operation unless the drive input has been turned OFF and ON again.
- 3) 16 byte operation requires the special auxiliary flag M8027 to be driver ON, otherwise it is 8 byte mode.
- 4) In the print process, once encountered "00" character, it will automatically end the printing operation regardless of the remaining character. M8029 the execution complete flag will be set ON for a scan cycle when the drive energy flow signal is invalid.
- 5) If the scan cycle is short, then chose the constant scan mode ,otherwise, the PR instruction could be written into a timer interrupt routine.

**4.2.66 PRUN instruction**

**Description**

Table 4-88

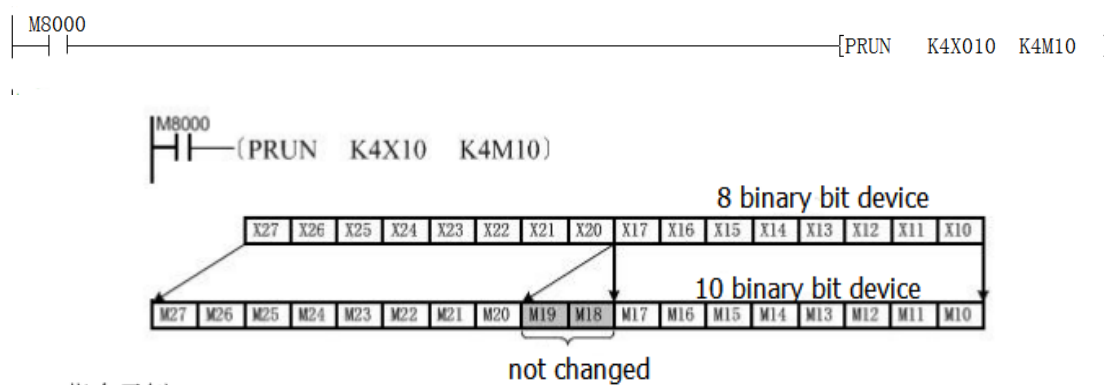
Name	Function	Devices		Format	Steps
		S	D		
PRUN	8 bit transmission	KnX, KnM	KnY,KnM		PRUN,PRU NP: 5 steps DPRUN, DPRUNP: 9 steps.

**Operation**

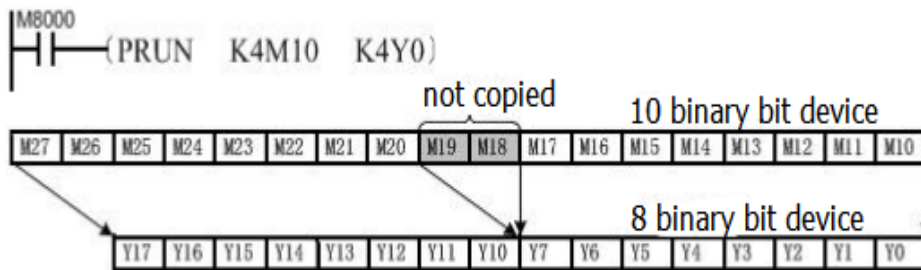
- 1) This instruction allows source data(the data unit is 8 bit wide) to be moved into the bit transmission area that begins from D.
- 2) S is the head address of the source data. For ease, the head address bit should be a multiple of 10,e.g. X10, M100 etc.
- 3) D is the destination address of source data. For ease, the destination address bit should be a multiple of 10,e.g. M100, Y30, etc. For Kn ,n=1 to 8.

**Program Example**

Program Example 1



Program Example 2



### 4.2.67 PTO instruction

#### Description

Table 4-89

Name	Function	Devices		Format	Steps
		S1	S2		
PTO	Pulse envelope output instruction	D	Y		PTO: 5 steps DPTO: 9 steps.

#### Operation

Take operator S1 as the starting address, then the data table is as below:

Table 4-90

ADDRESS OFFSET	SECTION	DESCRIPITON
0		Number of segments:1 to 255(0 is not output pulse)
1		Record the number currently being
2		The number of executions of the Envelope table(-1:doesn't execute 0:always execute )( Restart to take effect)
.....		reserved
10	#1	Initial frequency(range of frequencies)(0~200,000)
11		Frequency increment(signed ±20,000~20,000 )
12		Pulse number(1-4,294,967,295 )
13	#2	Initial frequency(range of frequencies)(0~200,000)
14		Frequency increment(signed ±20,000~20,000)
15		Pulse number(1-4,294,967,295 )
(continuous)	#3	(continuous)

When using the 32-bit instruction DPTO, accounting for two-word address, the address offset is 2.

#### Program Example



Use the PTO to control a stepper motor to achieve a simple acceleration, constant speed and deceleration or a complex process consisting of up to 255 pulses, and every waveform is acceleration, constant speed or deceleration operation. Starting and final pulse frequency is 2KHZ, the maximum pulse frequency is 10KHZ, and it requires 4000 pulses to achieve the desired number of revolutions of the motor.

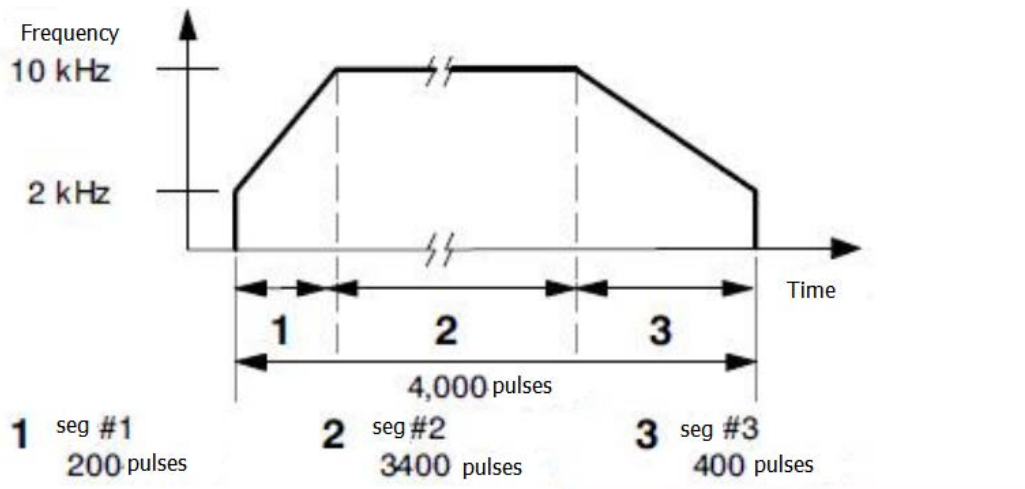


Figure 4-7

The example above required to produce a output signal contained three sections:

Stepper motor acceleration (section 1);

Stepper motor constant speed (section 2);

Stepper motor deceleration (section 3);

According to the note 4 below, we get the frequency increment of each section:

Sec 1(acceleration) frequency increment=40

Sec 2(constant speed) frequency increment=0

Sec 3(deceleration) frequency increment= -20

Considering the combination of instruction and fig 1, users get the envelope table:

Segment	Register address	value	description
Parameter setting	D0	3	Total segments
	D1	0	Record the number currently being executed
	D2	0	Number of executions of Envelope table
#1	D10	2khz	Initial frequency

#1	D11	40	Frequency increment
	D12	200	Pulse number
#2	D13	10khz	Initial frequency
	D14	0	Frequency increment
	D15	3400	Pulse number
#3	D16	10khz	Initial frequency
	D17	-20	Frequency increment
	D18	400	Pulse number

**Note:**

- 1) Take the frequency as the standard, run the command during the operation.
- 2) The range of frequency:0 to 100 kHz
- 3) If the envelope table is beyond the effective range of the device , no pulse will be sent out.
- 4) Frequency increment formula:  
Frequency increment= (final frequency - initial frequency)/ the number of pulse
- 5) The frequency interval of pulse (including inter-segment and segment) can not exceed 2000Hz, otherwise it will go wrong (the wrong number is 6780)and the instruction will not be executed.
- 6) If the frequency interval of pulse (including inter-segment and segment) exceeds 2000Hz, then PTO will not be executed:
  - Cyclic transmission mode: the last pulse of the last segment and the first pulse of the first segment are regarded as the neighboring pulse.
  - Single transmission mode: the last pulse of the last segment and the first pulse of the first segment are not regarded as the neighboring pulse.

**4.2.68 PWM instruction**

**Description**

Table 4-91

Name	Function	Devices			Format	Steps
		S1	S2	D		
PWM	PULSE WIDTH MODULATION	K,H,KnX,KnY,KnM,nS,T,C,D,V,Z	K,H,KnX,KnY,KnM,KnS,T,C,D,V,Z	Y		7

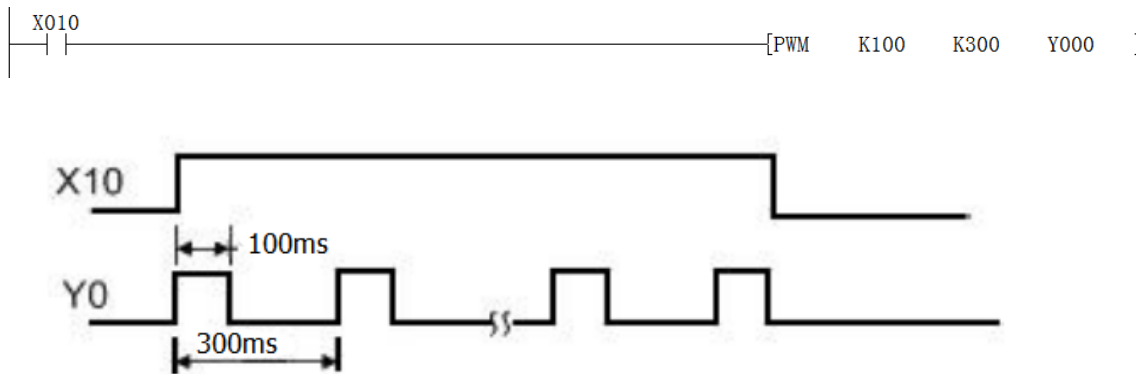
**Operation**

Since the relay is not suitable for high-frequency operation, only the transistor output PLC is suitable to use the command. S1 defines the width of the pulse.S2 defines the cycle of the pulse. D defines the output port.

Besides, S1 must be less than or equal S2,the setting range for S1 and S2 is 0~32767ms.

D is the pulse output port. For TP,D can be Y0 ~ Y3; FOR TPS, it can be Y0 ~ Y1.Don't occupy the same output port with the instruction like PLSY,PLSR.PWM is executed in the interrupt mode. When the drive energy flow signal is OFF, the output is stop. S1 and S2 can be changed during the execution.

**Program Example**



**4.2.69 RAMP instruction**

**Description**

Table 4-92

Name	Function	Devices				Format	Steps
		S1	S2	D	n		
RAM P	Ramps a device from one value to another in the specified number of steps	D	D	D	Constant 1to32 767	<pre> X010 ---   ---[RAMP S1 S2 D n               D10 D11 D12 K100                     ]                     </pre>	9

**Operation**

The RAMP instruction varies a current value (D) between the data limits set by the user(S1 AND S2).The "journey" between these extreme limits takes n program scans. Once the current value of D equals the set value of S2, the instruction is over.

S1:the start value;

S2:the end value;

D:the current value is stored in D, the current scan number is stored in D+1.

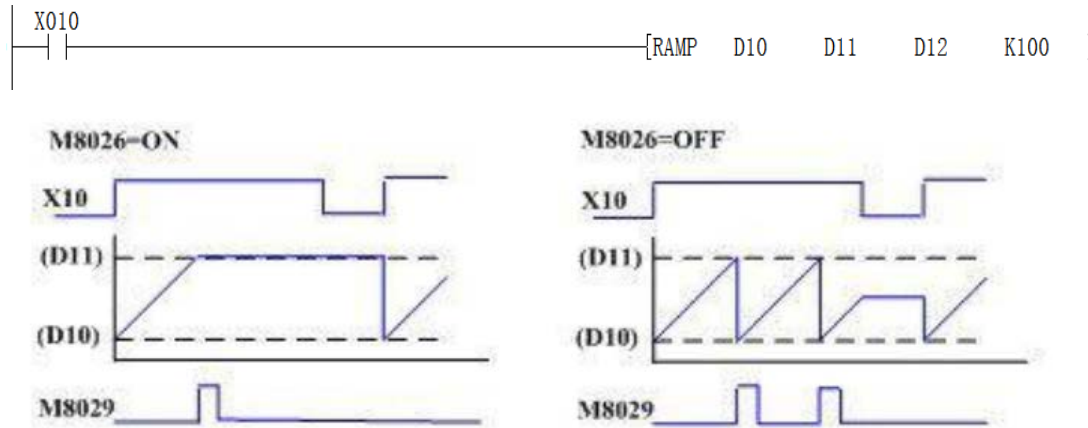
n:the number of scan. Since the output of the interpolation is executed in the normal main loop, in order to ensure linear interpolation output, the RAMP instruction must be operated with a constant scan mode.



Interpolation is calculated as integer, regardless of the decimal part.

RAMP has two modes defined by the M8026 flag. When each interpolation operation is completed, M8029 will set for a scan cycle. As shown in the following example:

**Program example**



**4.2.70 RCL instruction**

**Description**

Table 4-93

Name	Function	Devices		Format	Steps
		D	n		
RCL	The contents of the destination device are rotated left with 1 bit extracted to the carry flag	K,H,Kn X,KnY, KnM,K nS,T,C, D,V,Z	CONST ANT: n=1to16 (16 bit); n=1 to 32(32 bit)	$X000$ ───┤ ───┤ [RCL $D$ $n$ ├───┤ $D0$ $K4$	RCL,R CLP: 5 steps DRCL, DRCL P: 9 steps.

**Operation**

The contents of the D are rotated left n bit with the carry flag M8022. This instruction is generally used as pulse execution instruction, i.e. use the RCLP or DRCLP. When the instruction is 32bit, it takes 2 sequential addresses.

When D is KnY or KnM or KnS, only K4(16 bit) and K8(32 bit) are effective.

**Program Example:**

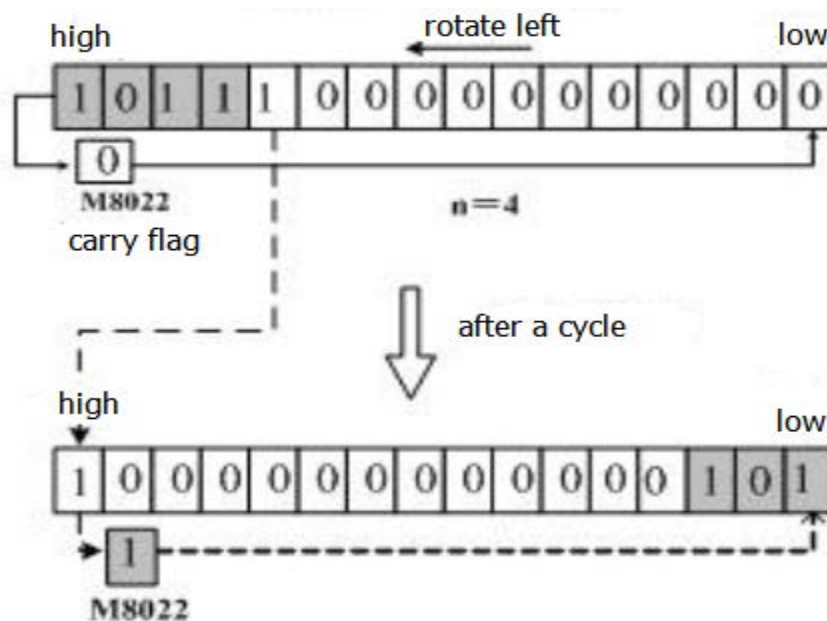
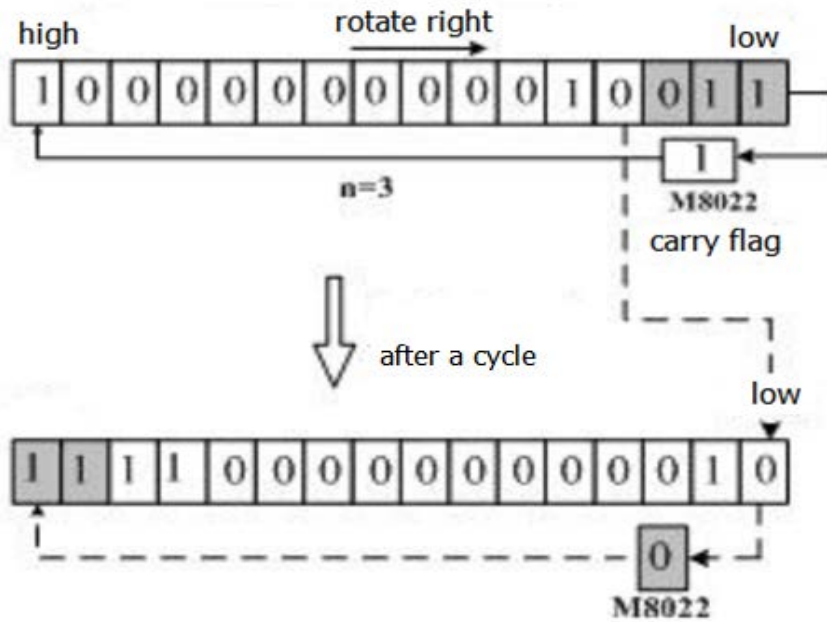
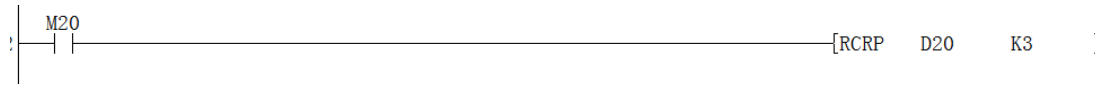




**Operation**

The contents of the D are rotated right n bit with the carry flag M8022. This instruction is generally used as pulse execution instruction, i.e. use the RCLP or DRCLP. When the instruction is 32bit, it takes 2 sequential address when D is KnY or KnM or KnS, only K4(16 bit) and K8(32 bit) are effective.

**Program Example**



### 4.2.72 REF instruction

#### Description

Table 4-95

Name	Function	Devices		Format	Steps
		D	n		
REF	Forces an immediate update of inputs or outputs as specified	X, Y	Constant from 1 to 256, always be a multiple of 8, i.e. 8, 16, 24 etc.	$\begin{array}{c} \text{X000} \\ \text{---   ---} \end{array} \text{[REF} \quad \text{D} \quad \text{n} \\ \text{X010} \quad \text{K8} \quad \text{]}$	REF: 5 steps  REFP: 5 steps.

#### Operation

Update or refresh blocks of n consecutive devices that beginning from the D device.

Because the REF instruction can only be used to update or refresh blocks of 8 (n) consecutive devices, so the head address of the refreshed devices, i.e. D should always have its last digits as a 0(zero), i.e. in units of 10, like X0, X10, Y0, Y10.

The value of n should always be a multiple of 8 (n=8 to 256)

Under normal circumstances, the state of the input port is read before the execution of scan. The refresh of the output port state will be executed in batches when the program scan is over i.e. till to the END, thus there will be a delay in IO process.

If the application requires the latest input information and the latest result of output, you can use the immediately refresh instruction REF.

REF can be used between the instruction FOR~ NEXT or CJ.

REF can be used in the interrupt subprogram to refresh the input information and the output result.

The delay of the input port state depends on the filter time of the input device. X0 to X7 have the digital filter function, the filter time is between 0 and 60 ms, the other IO ports are hardware filter that the filter time is 10 ms. The specific parameter you need to refer to the PLC manual.

The delay of the output port state change depends on the response time of the output element, such as relay. The output contact will not act until the response time of the relay or transistor is over.

The response lag time of the relay output type plc is about 10 ms (max :20ms), the high speed output port of the transistor plc is about 10 us, for the common output port of the transistor plc is about 0.5 ms. The specific parameter you need to refer to the PLC manual.

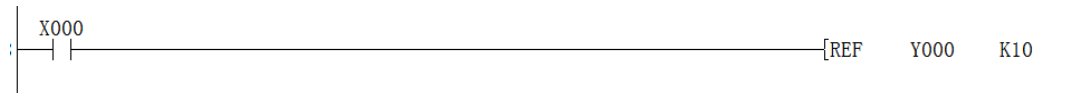
#### Program Example

- 1) Program Example 1



During the operation, once X20 is ON, the state of the input port X0 to X17 will be read immediately, the input signal will be refreshed and there is no input delay.

2) Program Example 2



During the operation, once X20 is ON, the state of the output port Y0 to Y17 will be read immediately, the output signal will be refreshed immediately rather than until the END instruction.

4.2.73 REFF instruction

Description

Table 4-96

Name	Function	Devices	Format	Steps
		n		
REFF	Input are refreshed, and their input filters are reset to the newly designated value	Constant, n=0 to 60( ms)		REFF: 3 steps REFFP: 3 steps.

Operation

"n" is the filter time constant for X0 ~ X7 input port.

In the plc, X0~X7 use digital filters, the default filter time constant is set by the D8020. D8020 can be changed to 0 ~ 60ms by REFF instruction. The remaining X ports has only hardware RC filter that the filter time constant is about 10ms and can't be changed.

When using the interrupts or high speed counting, the filter time constant of the related port reduce to minimum automatically. The unrelated ports stay as it was.

User can also use MOV instruction to change the value of D8020.

Program Example



The filter time of X0~X7 is 5ms, when X10 is ON. The filter time of X0~ X7 is 15 ms, when X10 is

OFF.

### 4.2.74 ROL instruction

#### Description

Table 4-97

Name	Function	Devices		Format	Steps
		D	n		
ROL	The bit pattern of the destination devices is rotated n places to the left on every execution	K,H,Kn X,KnY, KnM,K nS,T,C, D,V,Z	Constant , n=1 to 16(16 bit), n=1 to 32( 32 bit)		ROL,R OLP: 5 steps DROL, DROL P: 9 steps.

#### Operation

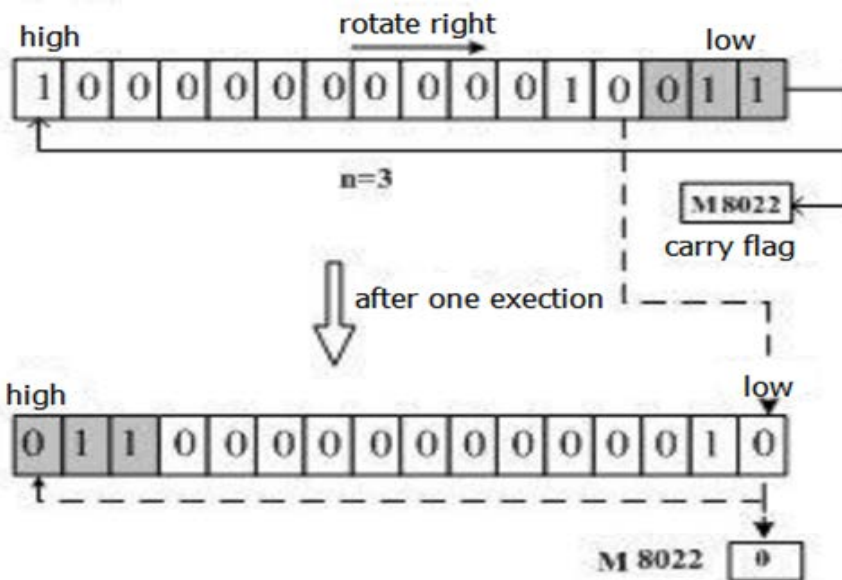
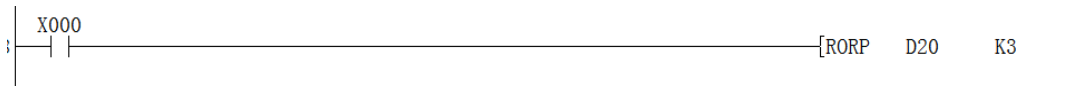
The bit pattern of D is rotated n bits to the left on every execution.

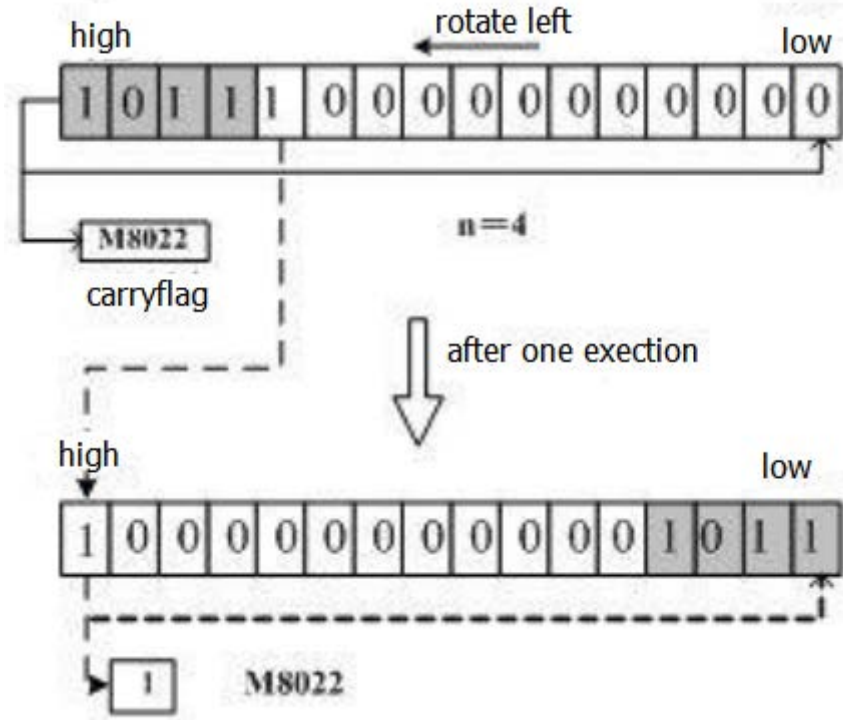
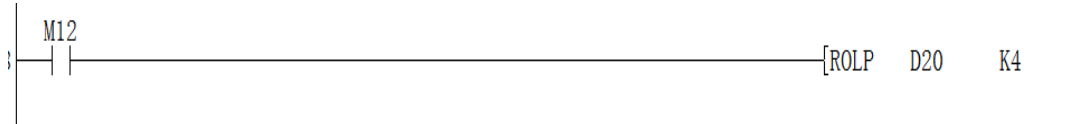
This instruction is generally used in pulse execution instruction.

When the instruction is 32 bit, it occupies the subsequent neighboring address.

When the device in D is KnY, KnM or KnS, only K4 (16-bit) and K8 (32-bit) is effective. The status of the last bit rotated is copied to carry flag M8022.

#### Program Example





### 4.2.75 ROR instruction

#### Description

Table 4-98

Name	Function	Devices		Format	Steps
		D	n		
ROR	The bit pattern of the destination devices is rotated n places to the right on every execution	K,H,Kn X,KnY, KnM,K nS,T,C, D,V,Z	Constan, n=1 to 16(16 bit), n=1 to 32( 32 bit)	$X000$ 	ROR,R ORP: 5 steps DROR, DROR P: 9 steps.

#### Operation

The bit pattern of D is rotated n bits to the right on every execution.

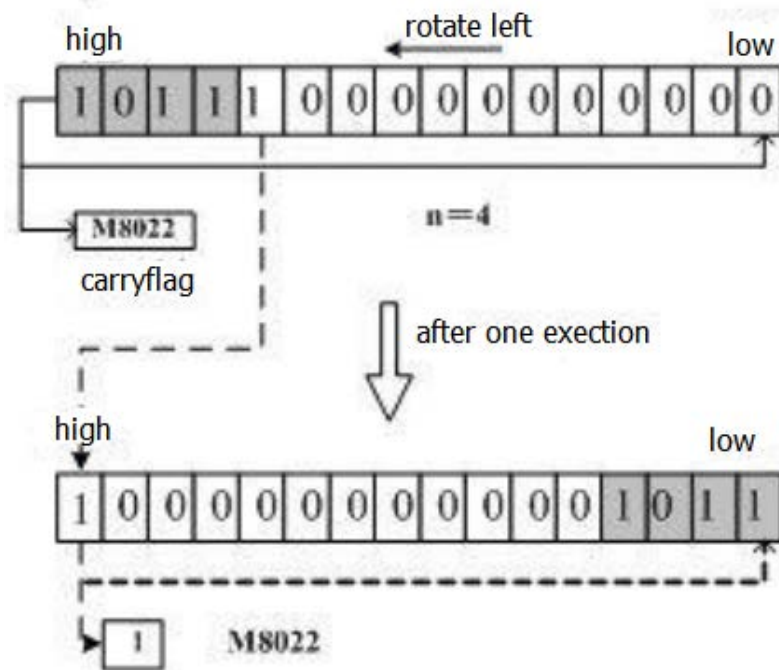
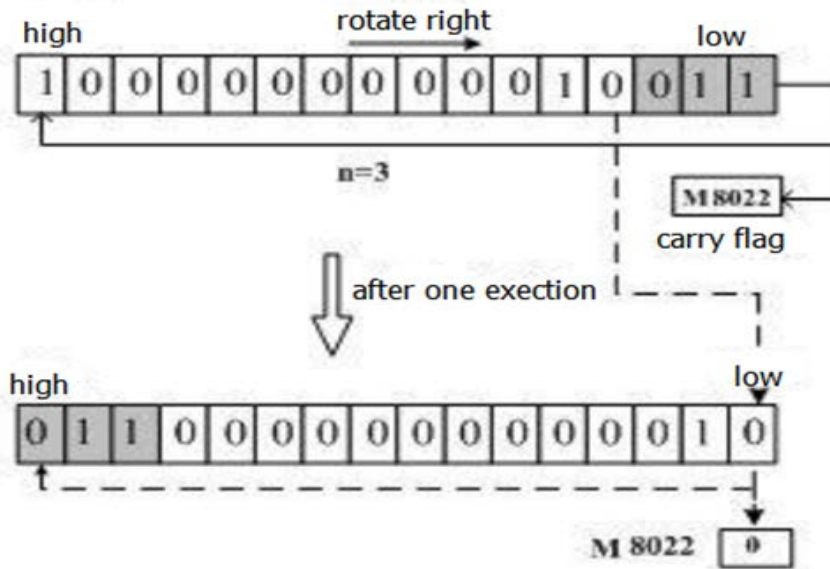
This instruction is generally used in pulse execution instruction.

When the instruction is 32 bit, it occupies the subsequent neighboring address.

When the device in D is KnY, KnM or KnS, only K4(16 bit) and K8(32 bit) is effective.

The status of the last bit rotated is copied to carry flag M8022.

Program Example



4.2.76 ROTC instruction

Description

Table 4-99

Name	Function	Devices	Format	Steps
------	----------	---------	--------	-------



		S	D	M1	M2		
ROTC	Controls a rotary tables movement is response to a requested position	D	Y,M, S	2 to 3276 7,m1 ≥m2	0 to 32767,m1 ≥m2		9

**Operation**

The ROTC instruction is used to aid the tracking and positional movement of the rotary table as it moves to a specified destination.

The position detection signal of the rotary table needed to be arranged as the specified configuration.

- 1) S: the head address of the count variable
- 2) m1: the number of stations in the rotary table,  $m1 \geq m2$
- 3) m2: the number of stations in the rotary table in low speed,  $m1 \geq m2$
- 4) D: the head address of the position detection signal of the rotary table, it takes the subsequent 8 bit position.

For example, shown as fig. below, X0 and X1 are connected with AB quadrature encoder A-phase and B-phase output signal respectively. A mechanical switch may be used to obtain the signal quadrature phase as well. X2 connects to number 0 station for the detection of input( when rotated to station No. 0 ,it is ON).Then we can get the rotational speed ,steering direction and steering station of the rotary table whereby the 3 signals.

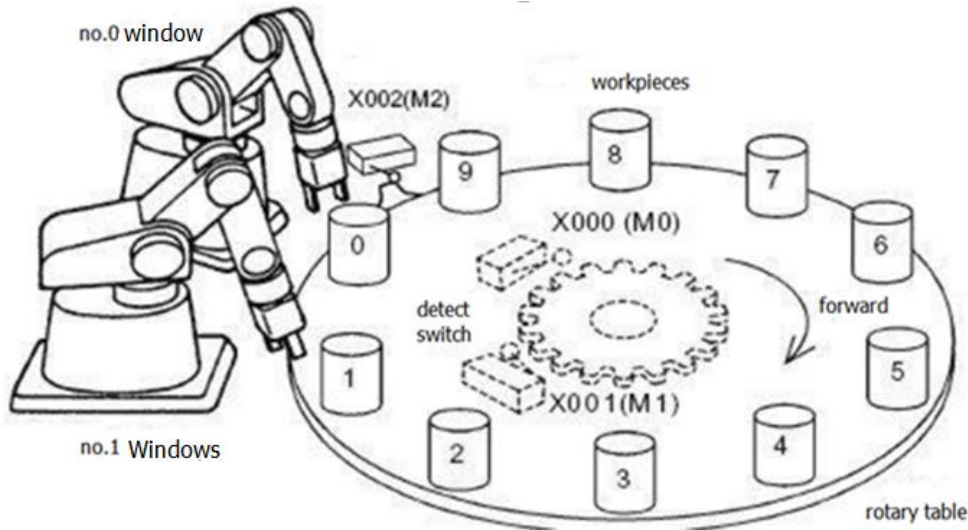


Figure 4-8

**Program Example**



Variables are as follows:

Table 4-100

Variable	Function	Operation description
D200	Count register	User pre-configured three units
D201	Windows number setting	
D202	Work piece number setting	
M0	A phase signal	executed before each scan
M1	B phase signal	
M2	0 detection signal	
M3	High-speed forward	When x10 is ON, You can automatically get the results of m0 to m7, otherwise they are OFF.
M4	low-speed forward	
M5	Stop	
M6	low-speed reverse	
M7	High-speed reverse	

In the subsequent user programs, you can control an external actuator by m0 to m7 outputted by the Y port. The instruction ROTC only can be used once in the program.

### 4.2.77 RS instruction

#### Instruction description

Table 4-101

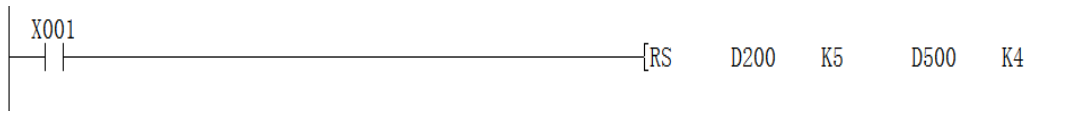
Name	Function	Devices				Format	Steps
		S	M	D	n		
RS	Used to control serial communications from/to the plc	D	K,H, D	D	K,H, D	$X001 \begin{array}{c} \text{---}   \text{---} \\ \text{---}   \text{---} \end{array} \text{[RS } \begin{array}{c} S \\ D200 \end{array} \begin{array}{c} M \\ K5 \end{array} \begin{array}{c} D \\ D500 \end{array} \begin{array}{c} n \\ K4 \end{array} ]$	9

#### Operation

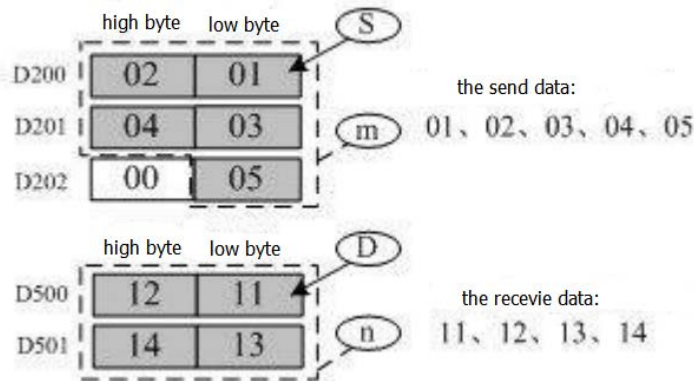
RS is a transceiver instruction that automatically sends the data stored in the specific register to the serial port sequentially and stores the data received by serial port in the specific area. It is equivalent to directly access the communication buffer. With the user program for communication transceiver buffer processing, we can implement the user-defined protocol communication.

- 1) S: the head address of the register where the to be sent data stored in
- 2) M: the length of the to be sent data (byte), 0 to 256.
- 3) D: the head address of the register where the receive data stored in
- 4) n: the length of the receive data(byte),0 to 256

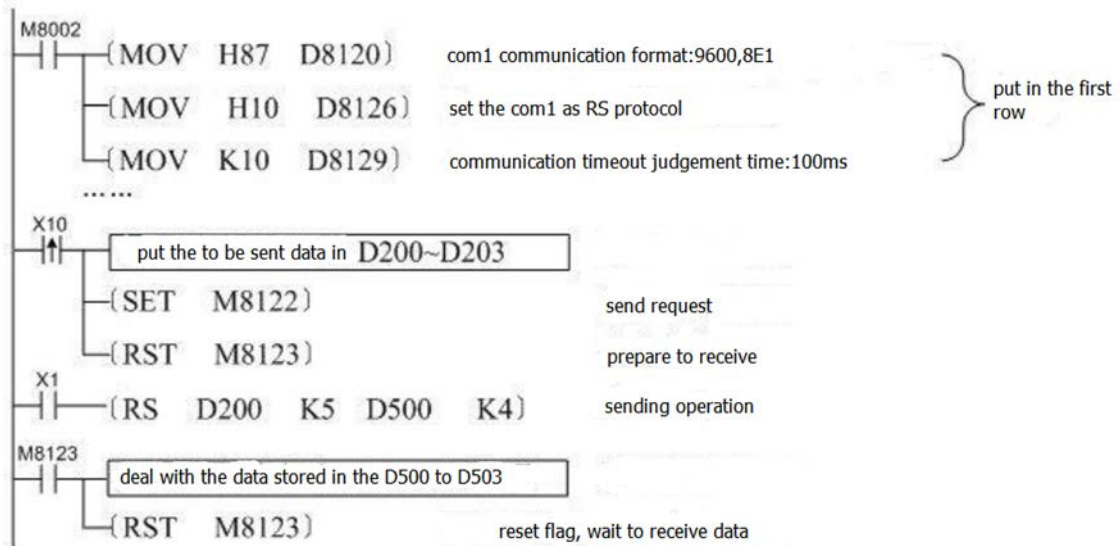
**Program Example**



When X1 is ON, the receive data and the send data is shown as below.



In actual programming, you need to do some preparation for serial communication and configuration, such as baud rate, check bit, timeout judgment condition, protocol etc. The same example, a relatively complete set of RS communication procedures are as follows.



**4.2.78 SEGD instruction**

**Instruction description**

Table 4-102

Name	Function	Devices		Format	Steps
		S1	D		
SEGD	Hex data is decoded into a format used to drive seven segment displays	K,H,Kn X, KnY, KnM,Kn S,T,C,D, V,Z	KnY,Kn M,KnS,T, C,D,V,Z	X020 —   — [SEGD S D0 K2Y010 ]	SEGD, SEGDP :5 steps

**Operation**

A single hexadecimal digit occupying the lower 4 bits of source device S is decoded into a data format used to drive a seven segment display. A representation of the hex digit is then displayed. The decoded data is stored in the lower 8 bits of destination device D. The bit devices indicate:

- 1) S: The source data remaining to be decoded (b0 to b3)
- 2) D: The variable used to store the decoded data

**Program example**



When X20 is ON, the lower 4 bits of D0 are decoded and sent to port Y0 to Y17. The segment table is shown as below. PLC system already contains this table internally.

Table 4-103

data		segments	decoded table value								decoded character
HEX	BIN		B7	B6	B5	B4	B3	B2	B1	B0	
0	0000	<p>each bit corresponds to a segment 1=ON 0=OFF</p>	0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	1
3	0011		0	1	0	0	1	1	1	1	1
4	0100		0	1	1	0	0	1	1	0	1
5	0101		0	1	1	0	1	1	0	1	1
6	0110		0	1	1	1	1	1	0	1	1
7	0111		0	0	0	0	0	1	1	1	1
8	1000		0	1	1	1	1	1	1	1	1
9	1001		0	1	1	0	1	1	1	1	1
A	1010		0	1	1	1	1	0	1	1	1
B	1011		0	1	1	1	1	1	0	0	1
C	1100		0	1	1	1	1	0	0	1	1
D	1101		0	1	0	1	1	1	1	0	1
E	1110		0	1	1	1	1	0	0	1	1
F	1111		0	1	1	1	0	0	0	1	1

### 4.2.79 SEGL instruction

#### Instruction description

Table 4-104

Name	Function	Devices			Format	Steps
		S	D	n		
SEGL	Writes data to multiplexed single digit displays -4 digits per set	K, H,KnX, KnY,KnM,KnS,T ,C,D,V,Z	Y	0 to 7		SEGL 7 steps

#### Operation

SEGL uses 8 or 12 Y port to drive 4 bits or 8 bits seven-segment digital tube. Tube is display by scan driving mode. The bit devices indicate:

- 1) S: The data to be displayed, it will not be displayed until the value is converted to BCD.
- 2) D: The beginning NO. of the Y port that used to drive digital tube.

### 4.2.80 SER instruction

#### Instruction description

Table 4-105

Name	Function	Devices				Format	Steps
		S1	S2	D	n		
SER	Generates a list of statistics about a single data value located/ found in a data stack	KnX, KnY, KnM ,KnS ,T,C, D,V, Z	K,H, KnX, KnY, KnM, KnS,T ,C,D, V,Z	KnX, KnY, KnM, KnS,T ,C,D, V,Z	K,H,D( 16bit:1 to 256;32 bit:1 to 128)		SER,S ERP: 7 steps DSER, DSERP : 17 steps.

**Operation**

SER is used to find the same data, maxim data and minimum data in a set of data. The bit devices indicate:

- 1) S1: The head address of the data set
- 2) S2: The data to be retrieved
- 3) D: The head address of the search results storage area
- 4) n:the length of data set

When using 32bit instruction, S1, S2, D and n are calculated by the width of 32 bits.

**Program Example**



Table 4-106

S1	Retrieved data set	S2	Device No.	Result
D10	(D10)=K100	D0=K100	0	Equal
D11	(D11)=K123		1	
D12	(D12)=K100		2	Equal
D13	(D13)=K98		3	
D14	(D14)=K111		4	
D15	(D15)=K66		5	Min
D16	(D16)=K100		6	Equal
D17	(D17)=K100		7	Equal
D18	(D18)=K210		8	Max
D19	(D19)=K88		9	

Search results:

Table 4-107

D	Parameter	Definition
D80	4	Number of equal parameters
D81	8	The first number of equal parameters
D82	7	The last order number of equal parameters
D83	5	The order number of min para
D84	8	The largest order number of equal para

**Note**

- 1) If there are several max or min values, display the device with the largest order number

respectively.

- 2) The search result occupies 5 sequential address that begin from D. If there are no equal data, the D80 ~ D82 are 0 in the above example.

### 4.2.81 SFRD instruction

#### Instruction description

Table 4-108

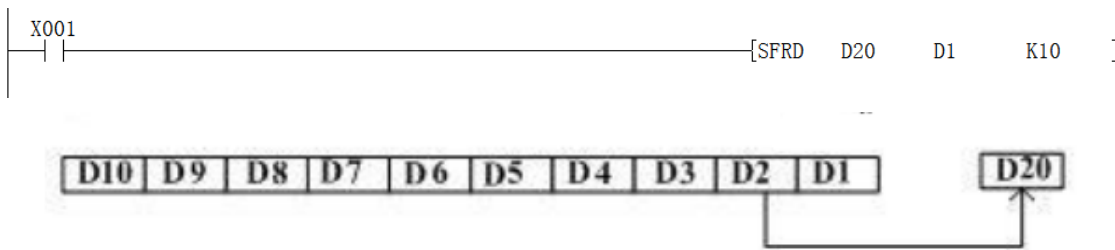
Name	Function	Devices			Format	Steps
		S1	D	n		
SFRD	This instruction reads and reduces FIFO stack	K,H,Kn X,KnY, KnM,KnS,T,C, D	KnY, KnM, KnS,T ,C,D	16bit:1 to 256; 32bit:1 to 128		SFRD: 7 steps  SFRDP : 7 steps.

#### Operation

The source device(S) identifies the head address of the FIFO stack. This instruction reads the first piece of data from the FIFO stack (register S+1), moves all of the data within the stack 'up' one position to fill the read area and decrements the contents of FIFO head address(S) by 1. The read data is written to the destination device (D).When the contents of source device (S) are equal to '0'(zero),i.e. the FIFO stack is empty, the flag M8020 is turned ON.

This instruction is generally used as pulse instruction, i.e. SFRDP.

#### Program example



When X0 switches from OFF to ON, the instruction acts as the numbers 1 to 3.(the value of D10 keeps unchanged)

- 1) The content in D2 is read out and delivered to D20.
- 2) D10 to D3 are shifted to right for a unit.
- 3) Pointer D1 minus 1.

### 4.2.82 SFTL instruction

#### Instruction description

Table 4-109

Name	Function	Devices				Format	Steps
		S	D	N1	N2		
SFTL	The status of the source devices are copied to a controlled bit stack moving the existing data to the left	X Y, M; S	Y, M; S	$n1 \leq 1$ 024	$n2 \leq 1$ $n1$		SFTL: 9 steps SFTLP: 9 steps.

#### Operation

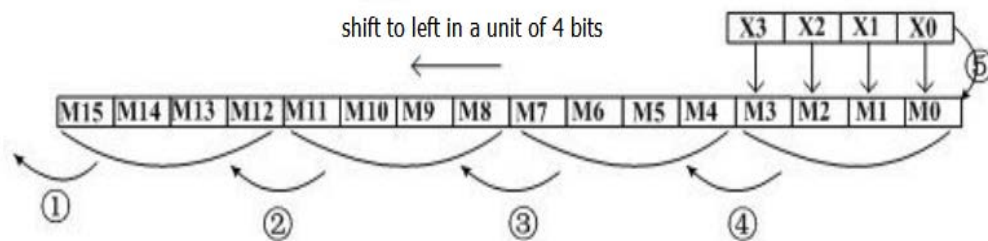
The instruction copies  $n2$  source devices beginning from  $S$  to a bit stack of length  $n1$  beginning from  $D$ . For every new addition of  $n2$  bits, the existing data within the bit stack is shifted  $n1$  bits to the left. Any bit data moving to a position exceeding the  $n1$  limit is diverted to an overflow area.

This instruction is generally used as pulse instruction, i.e. SFTLP.

#### Program Example



- 1) M15-M12 → overflow
- 2) M11-M8 → M15-M12
- 3) M7-M4 → M11-M8
- 4) M3-M0 → M7-M4
- 5) X3-X0 → M3-M0



### 4.2.83 SFTR instruction

#### Instruction description



Table 4-110

Name	Function	Devices				Format	Steps
		S1	D	N1	N2		
SFTR	The status of the source devices are copied to a controlled bit stack moving existing to the right	X Y, M; S	Y, M; S	$n1 \leq 1$ 024	$n2 \leq n$ 1	$\begin{matrix} X006 \\ -    - \end{matrix} \text{[SFTR } \begin{matrix} S \\ X000 \end{matrix} \begin{matrix} D \\ M0 \end{matrix} \begin{matrix} n1 \\ K16 \end{matrix} \begin{matrix} n2 \\ K4 \end{matrix} ]$	7

**Operation**

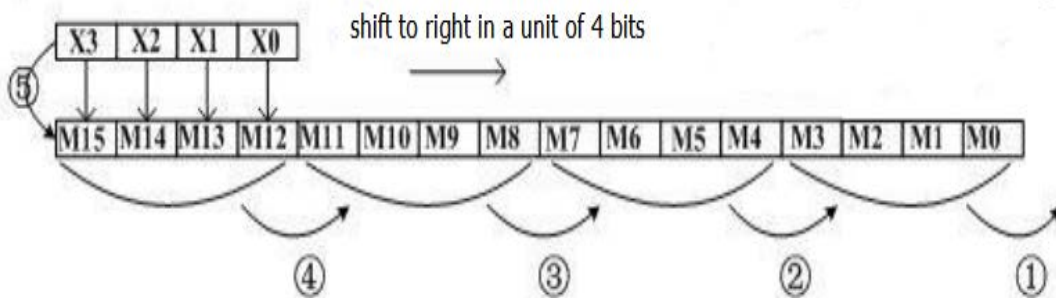
The instruction copies n2 source devices beginning from S to a bit stack of length n1 beginning from D. For every new addition of n2 bits, the existing data within the bit stack is shifted n1 bits to the right. Any bit data moving to a position exceeding the n1 limit is diverted to an overflow area.

This instruction is generally used as pulse instruction, i.e. SFTRP.

**Program Example**



- 1) M3-M0 → overflow
- 2) M7-M4 → M3-M0
- 3) M11-M8 → M7-M4
- 4) M15-M12 → M11-M8
- 5) X3-X0 → M15-M11



**4.2.84 SFWR instruction**

**Instruction description**

Table 4-111

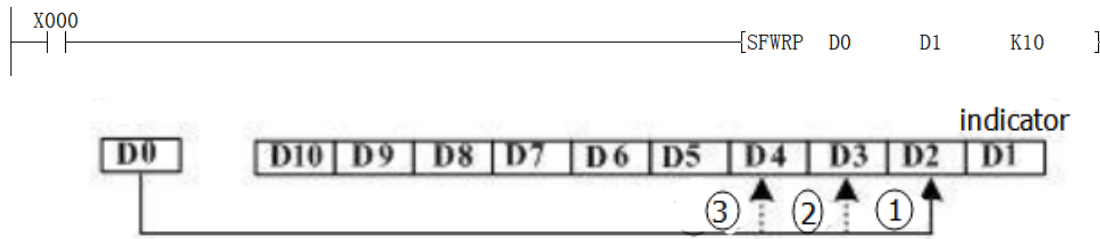
Name	Function	Devices			Format	Steps
		S1	D	n		
SFWR	This instruction creates and builds a FIFO stack n devices long	K,H, KnX, KnY, KnM, KnS,T ,C,D	KnY, KnM, KnS,T ,C,D	2048 ≥n ≥ 2	$\begin{matrix} X000 & S & D & n \\ \text{--- } & \text{--- } & \text{--- } & \text{--- } \\ \text{--- } & \text{--- } & \text{--- } & \text{--- } \\ \text{--- } & \text{--- } & \text{--- } & \text{--- } \\ \text{--- } & \text{--- } & \text{--- } & \text{--- } \\ \text{--- } & \text{--- } & \text{--- } & \text{--- } \\ \text{--- } & \text{--- } & \text{--- } & \text{--- } \end{matrix}$	SFWR 7 STEPS  SFWRP 7 STEPS

**Operation**

The contents of source device (S) are written to the FIFO stack. The position of insertion into stack is automatically calculated by the PLC. The destination device (D) is the head address of the FIFO stack. The contents of D identify where the next record will be stored (as an offset from D+1).

This instruction is generally used as pulse instruction, i.e. SFWRP.

**Program Example**




When X0 turns ON, the content of D0 will be stored in D2, the value of D1 is changed to 1 . When X0 switch from OFF to ON again, the content in D0 will be stored in D3, the value of D1 is changed to 2 and so forth. If the contents of D1 exceed the value 'n-1' (n is the length of FIFO stack) then insertion into the FIFO stack is stopped. The carry flag M8022 is turned ON to identify this situation.

**4.2.85 SMOV instruction**

**Instruction description**

Table 4-112

Name	Function	Devices					Format	Steps
		S	M1	M2	D	n		
SMOV	Takes elements of an existing 4 digit decimal number and inserts them into a new 4 digit number	K,H, KnX, KnY, KnM KnS,T ,C,D,	K,H		KnY, KnM, KnS, T, C, D, V, Z	K, H		SMOV 11 steps  SMOV 11 steps

**Operation 1**

This instruction copies a specified number of digits from a 4 digit decimal source (S) and places them at a specified location within a destination (D) number (also a 4 digit decimal). The existing data in the destination is overwritten.

**Key:**

- 1) m1 - The source position of the 1st digit to be moved
- 2) m2 - The number of source digits to be moved
- 3) n- The destination position for the first digit
- 4) S, D Range 0 to 9,999 (decimal) or 0 to 9,999 (BCD)

**Note:**

The selected destination must NOT be smaller than the quantity of source data. Digit positions are referenced by number: 1= units, 2= tens, 3= hundreds, 4=thousands.

**Operation 2**

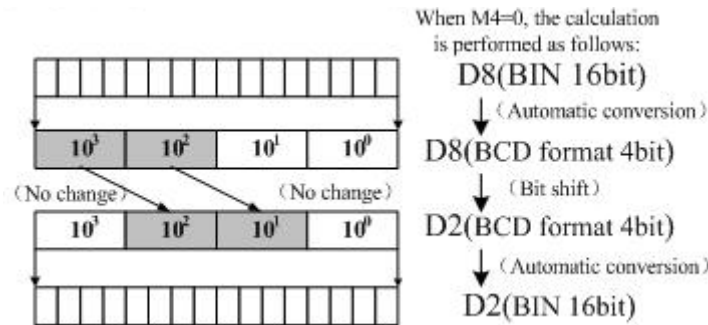
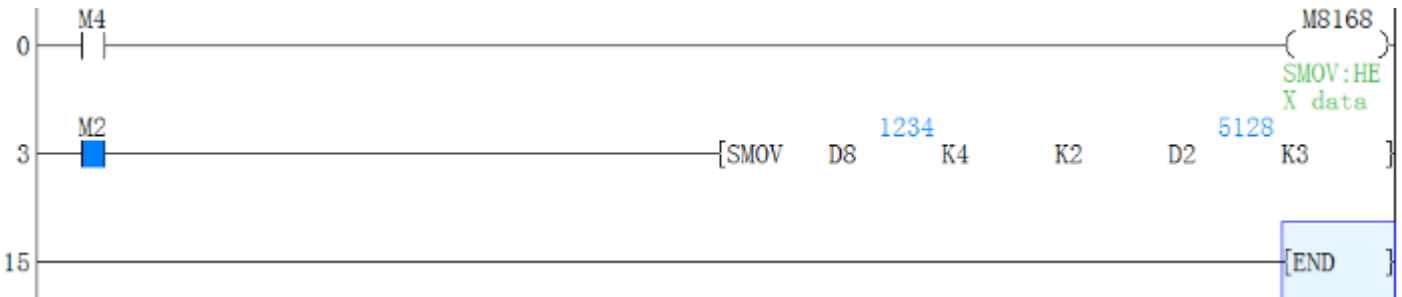
Allows BCD numbers to be manipulated in exactly the same way as the ‘normal’ SMOV manipulates decimal numbers, i.e. This instruction copies a specified number of digits from a 4 digit BCD source (S) and places them at a specified location within a destination (D) number (also a 4 digit BCD number).

To select the BCD mode the SMOV instruction is coupled with special M coil M8168 which is driven ON. Please remember that this is a ‘mode’ setting operation and will be active, i.e. all SMOV instructions will operate in BCD format until the mode is reset, i.e. M8168 is forced OFF.

**Program Example**

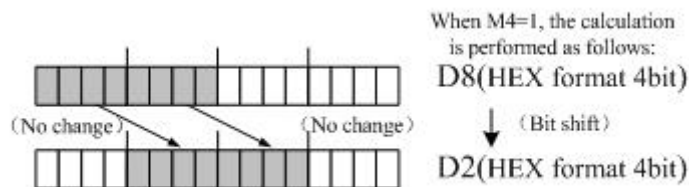
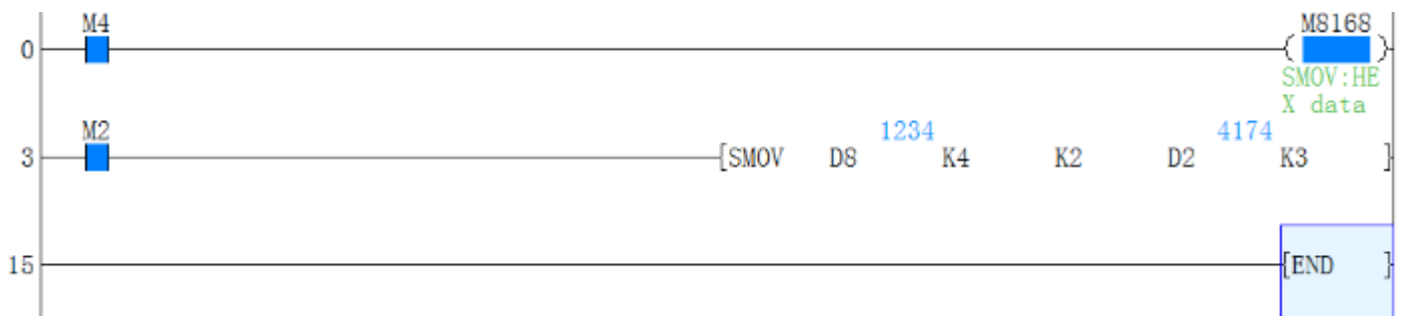
- 1) Example 1

If D8=K1234, D2=K5678, and M8168 is OFF (BCD mode),. When M2 is set to ON and D2 value is K5128;



2) Example 2

If D8=K1234=04d2(H), D2=K5678=162e (H), and M8168 is ON (BIN mode), When M2 is ON, M2 is set to ON and D2=K4174=104e (H).



4.2.86 SORT instruction

Instruction Description

Table 4-113

Name	Function	Devices					Format	Steps
		S	m1	m2	D	n		
SORT (SORT Tabulated Data)	Data in a defined table can be sorted on selected fields while retaining record integrity	D	(K, H) Note: m1= 1 to 32 m2= 1 to 6		D	K, H D) Note : n = 1 to m2		SORT : 11 steps

**Operation**

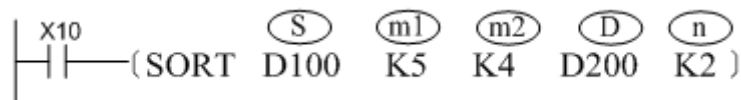
This instruction constructs a data table of m1 records with m2 fields having a start or head address of S. Then the data in field is sorted in to numerical order while retaining each individual records integrity. The resulting (new) data table is stored from destination device D.

**Points to note**

- 1) When a sort occurs each record is sorted in to ascending order based on the data in the selected sort field n.
- 2) The source (S) and destination (D) areas can be the same BUT if the areas are chosen to be different, there should be no overlap between the areas occupied by the tables.
- 3) Once the SORT operation has been completed the ‘Operation Complete Flag’ M8029 is turned ON. For the complete sort of a data table the SORT instruction will be processed m1 times.
- 4) During a SORT operation, the data in the SORT table must not be changed. If the data is changed, this may result in an incorrectly sorted table.
- 5) The SORT instruction may only be used **ONCE** in a program.

From the example instruction and the ‘data table’ The following data manipulation will occur when ‘n’ is set to the identified field

**Program Example**



When X10=ON, sort operation is implemented, and after the implementation, M8029 is set to 1 (program scan period); If it needs re-sorting, X10 should be reset from OFF to ON.

Equivalent table of the above instructions and data examples:

Data result after sorting of  $n = K2$  according to the instruction:

S		m2			
		Col no.	1	2	3
m1	Row no.	Student no.	Chinese	Mathematics	Physics
	1	D100 1	D105 85	D110 78	D115 83
	2	D101 2	D106 82	D111 91	D116 81
	3	D102 3	D107 77	D112 89	D117 88
	4	D103 4	D108 90	D113 81	D118 75
	5	D104 5	D109 87	D114 95	D119 77

D		n = K2			
		Col no.	1	2	3
Row no.	Student no.	Chinese	Mathematics	Physics	
	1	D200 3	D205 77	D210 89	D215 88
	2	D201 2	D206 82	D211 91	D216 81
	3	D202 1	D207 85	D212 78	D217 83
	4	D203 5	D208 87	D213 95	D218 77
5	D204 4	D209 90	D214 81	D219 75	

Data result after sorting of  $n = K4$  according to the instruction:

D		n = K4			
		Col no.	1	2	3
Row no.	Student no.	Chinese	Mathematics	Physics	
	1	D200 4	D205 90	D210 81	D215 75
	2	D201 5	D206 87	D211 95	D216 77
	3	D202 2	D207 82	D212 91	D217 81
	4	D203 1	D208 85	D213 78	D218 83
5	D204 3	D209 77	D214 89	D219 88	

### 4.2.87 SPD instruction

#### Instruction Description

Table 4-114

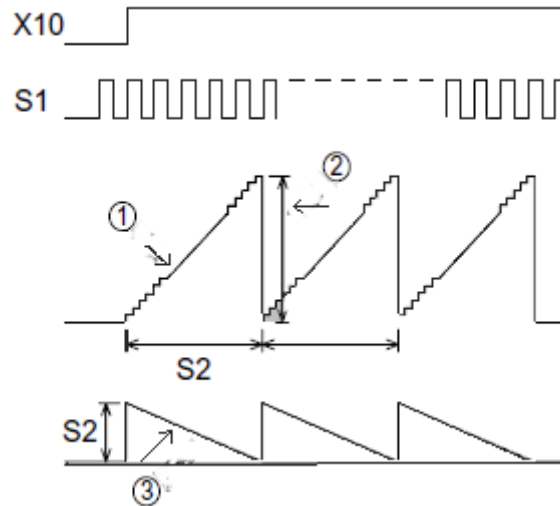
Name	Function	Devices			Format	Steps
		S1	S2	D		
SPD (Speed detection)	Detects the number of 'encoder' pulses in a given time frame. Results can be used to calculate speed	X0 to X5	K, H, KnX, KnY, KnM, KnS, T, C, D, V,Z	T, C, D, Z (V) Note: 3 consecutive devices are used. In the case Of D=Z monitor D8028,D8029 and D8030		SPD: 7 steps

#### Operation

The number of pulses received at S1 are counted and stored in D+1; this is the current count value. The counting takes place over a set time frame specified by S2 in msec. The time remaining on the current 'timed

count', is displayed in device D+2. The number of counted pulses (of S1) from the last timed count are stored in D. The timing chart opposite shows the SPD operation in a graphical sense. Note:

- 1) Current count value, device D+1
- 2) Accumulated/ last count value, device D
- 3) Current time remaining in msec, device D+2



**Point note**

- 1) When the timed count frame is completed the data stored in D+1 is immediately written to D. D+1 is then reset and a new time frame is started.
- 2) Because this is both a high speed and an interrupt process only inputs X0 to X5 may be used as the source device S1. However, the specified device for S1 must **NOT** coincide with any other high speed function which is operating, i.e. a high speed counter using the same input. The SPD instruction is considered to act as a single phase counter.
- 3) Multiple SPD instructions may be used, but the identified source devices S1 restrict this to a maximum of 6 times.
- 4) Once values for timed counts have been collected, appropriate speeds can be calculated using simple mathematics. These speeds could be radial speeds in rpm, linear speeds in M/min it is entirely down to the mathematical manipulation placed on the SPD results. The following interpretations could be used;

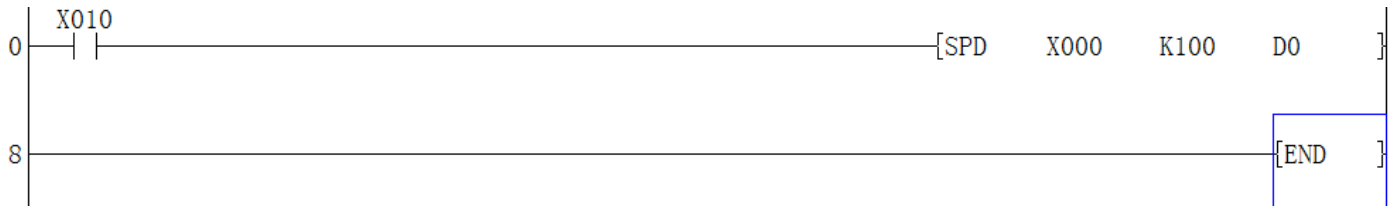
$$\text{Linear speed } N \text{ (km/h)} = \frac{3600 \times (D)}{n \times S2} \times 1000$$

Where n = the number of linear encoder divisions per kilometer.

$$\text{Radial speed } N \text{ (km/h)} = \frac{60 \times (D)}{n \times S2} \times 1000$$

Where n = the number of encoder pulses per revolution of the encoder disk.

**Program Example**



### 4.2.88 SQR instruction

#### Instruction Description

Table 4-115

Name	Function	Devices		Format	Steps
		S	D		
SQR (Square root)	Performs a mathematical square root e.g.: $D = \sqrt{S}$	K,H,D	D		SQR, SQRP: 5 steps DSQR, DSQRP:9 steps

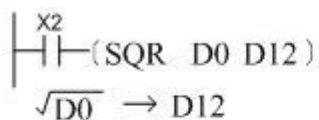
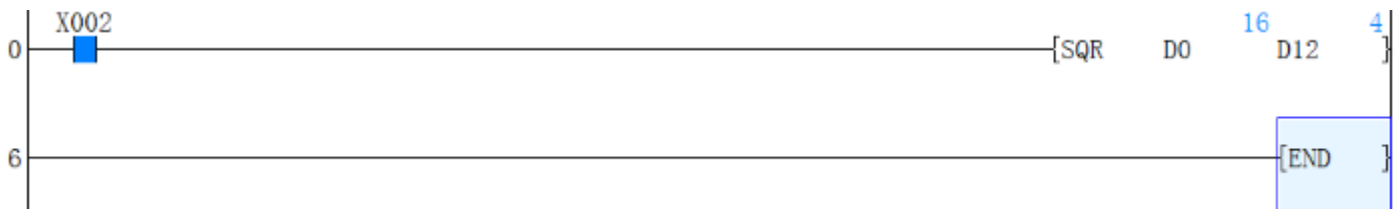
#### Operation

This instruction performs a square root operation on source data (S) and stores the result at destination device (D). The operation is conducted entirely in whole integers rendering the square root answer rounded to the lowest whole number. For example, if (S) = 154, then (D) is calculated as being 12. M8020 is set ON when the square root operation result is equal to zero. Answers with rounded values will activate M8021.

#### Note:

Performing any square root operation (even on a calculator) on a negative number will result in an error. This will be identified by special M coil M8067 being activated:

#### Program Example:



If D0=K100, D12=K10 when X2 is ON.

If D0=K110, D12=K10 with the decimals rounded off when X2 is ON.



### 4.2.89 STMR instruction

#### Instruction Description

Table 4-116

Name	Function	Devices			Format	Steps
		S	n	D		
STMR (Special timer)	Provides dedicated off-delay, one shot and flash timers	T Note: Timers 0 to 199 (100ms) ec devices )	K, H ) Note: n= 1 to 32,76 7	Y, M, S Note:uses 4 consecutive devices D+0to D+3		STM R: 7 steps

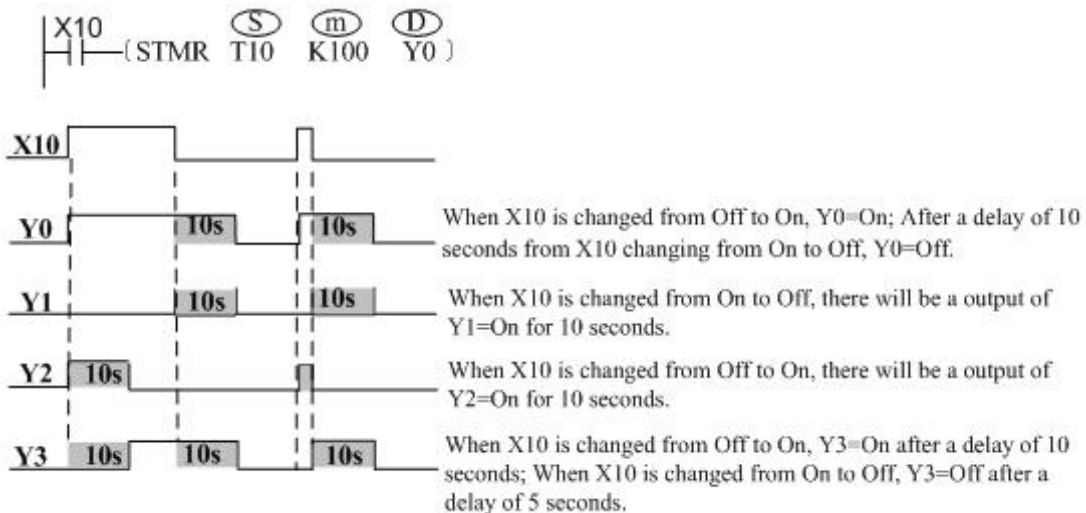
#### Operation

The designated timers S will operate for the duration n with the operational effect being flagged by devices D+0 to D+3. Device D+0 is an off-delay timer, D+1 is a one shot timer. When D+3 is used in the configuration below, D+1 and D+2 act in an alternate flashing sequence.

**Note:** The timer be used here, Then it can't be reused in other instructions.

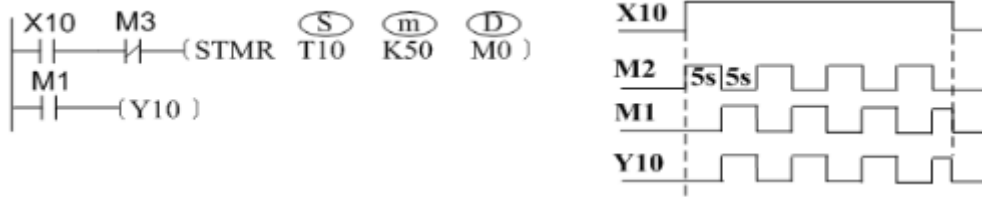
#### Program Example

1) Example 1



2) Example 2

This example also can be implemented by ALT instruction



### 4.2.90 SUB instruction

#### Instruction Description

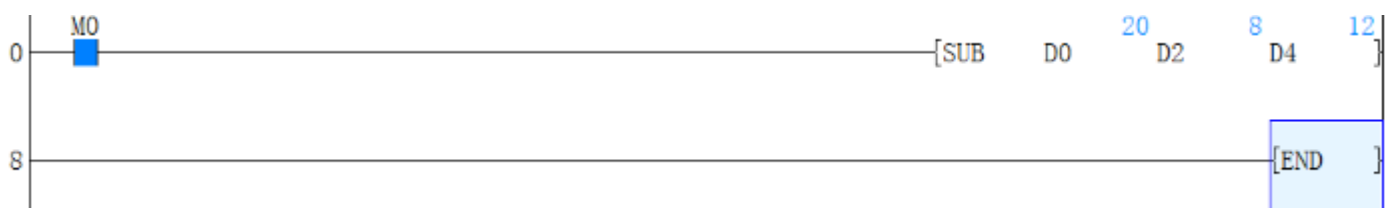
Table 4-117

Name	Function	Devices			Format	Steps
		S1	S2	D		
SUB (Subtract)	One source device is subtracted from the other - the result is stored in the destination device	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnX, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z		SUB, SUBP: 7 steps  DSUB, DSUBP: 13 steps

#### Operation: (Applicable to all units)

The data contained within the source device, S2 is subtracted from the contents of source device S1. The result or remainder of this calculation is stored in the destination device D. Note : the ‘Points to note’, under the ADD instruction (previous page) can also be similarly applied to the subtract instruction.

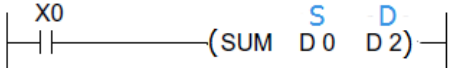
#### Program Example



### 4.2.91 SUM instruction

#### Instruction Description

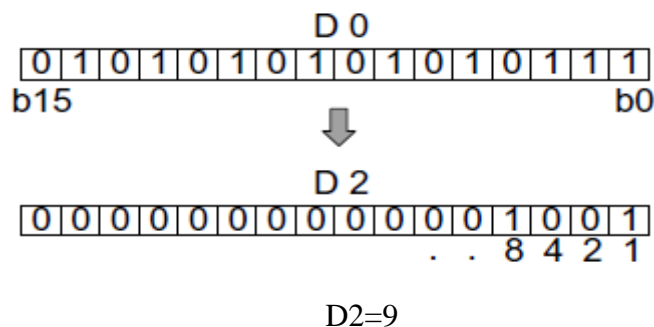
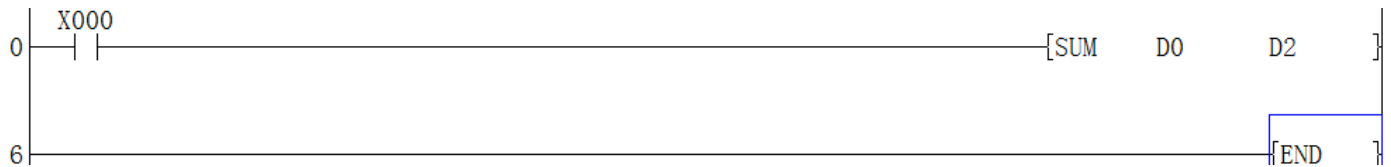
Table 4-118

Name	Function	Devices		Format	Steps
		S	D		
SUM (Sum of active bits)	The number (quantity) of active bits in the source data is stored in the Destination device	T, C, D Note: 3 consecutive devices are used to represent hours, minutes and seconds respectively.	KnY, KnM, KnS, T, C, D, V, Z		SUM,S UMP: 7 steps  DSUM , DSUM P: 9 steps

#### Operation

The numbers of active (ON) bits within the source device (S), i.e. bits which have a value of “1” are counted. The count is stored in the destination register (D). If a double word format is used, both the source and destination devices use 32 bit, double registers. The destination device will always have its upper 16 bits set to 0 (zero) as the counted value can never be more than 32.

#### Program example



### 4.2.92 SWAP instruction

#### Instruction Description

Table 4-119

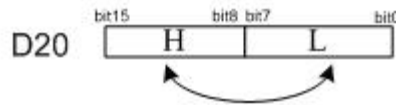
Name	Function	Devices	Format	Steps
		S		
SWAP (Byte Swap)	The high and low byte of the designated devices are exchanged	KnY, KnM, KnS, T, C, D, V, Z		SWAP, SWAPP : 5 steps  DSWAP, DSWAPP: 9 steps

#### Operation

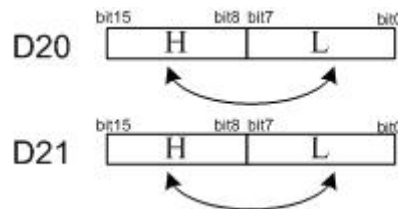
The upper byte and the lower byte of the source device are swapped.

#### Points to note

- 1) In single word (16 bit) operation the upper and lower byte of the source device are exchanged.

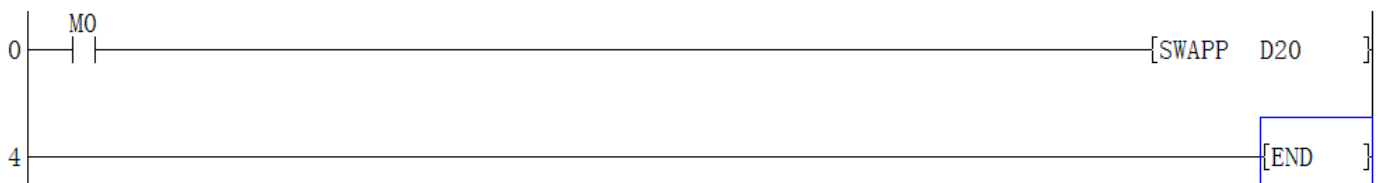


- 2) In double word (32 bit) operation the upper and lower byte of each or the two 16 bit devices are exchanged. Result of DSWAP(P) D10:



- 3) If the operation of this instruction is allowed to execute each scan, then the value of the source device will swap back to its original value every other scan. The use of the pulse modifier or an interlock program is recommended.

#### Program Example



### 4.2.93 TADD instruction

#### Instruction Description

Table 4-120

Name	Function	Devices			Format	Steps
		S1	D1	D2		
TADD (Time Addition)	Adds two time values together to give a new time	T, C, D Note: 3 consecutive devices are used to represent hours, minutes and seconds respectively.				TADD, TADD P:7 steps

#### Operation

- Each of S1, S2 and D specify the head address of 3 data devices to be used a time value. The time value in S1 is added to the time value in S2, the result is stored to D as a new time value.
- If the calculation result exceeds 24 hours, the carry flag M8022 is set to 1 and the actual displayed time will be subtracted with 24:00:00; If the calculation result is 00:00:00, zero flag M8020 is set to 1;

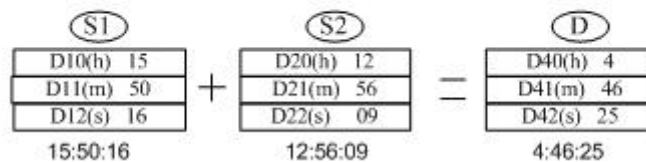
#### Points to note

- The addition is performed according to standard time values. Hours, minutes and seconds are kept within correct limits. Any overflow is correctly processed.



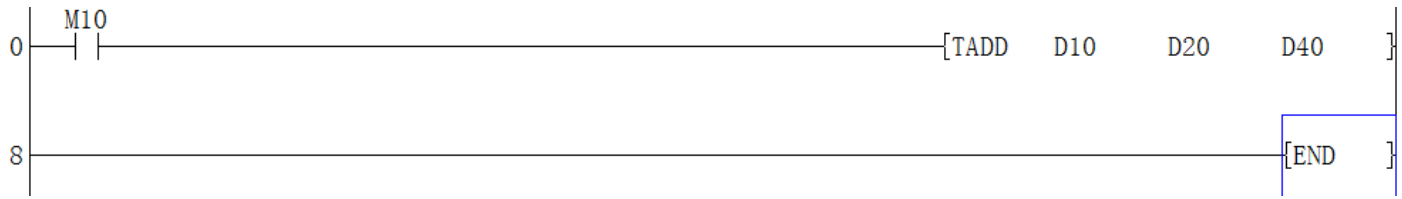
- If the addition of the two times results in a value greater than 24 hours, the value of the result is the time remaining above 24 hours. When this happens the carry flag M8022 is set ON.

If the result of addition operation is higher than 24 hours, the carry flag M8022 will be set to ON.



- If the addition of the two times results in a value of zero (0:00:00: 0 hours, 0 minutes, 0 seconds) then the zero flag M8020 is set ON.
- The same device may be used as a source (S1 or S2) and destination device. In this case the addition is continually executed; the destination value changing each program scan. To prevent this from happening, use the pulse modifier or an interlock program.

**Program Example**



**4.2.94 TCMP instruction**

**Instruction Description**

Table 4-121

Name	Function	Devices					Format	Steps
		S1	S2	S3	S	D		
TCMP (Time Compare)	Compares two times - results of <, = and > are given	X, Y, M, S Note: uses 10 Consecutive devices (identified as S+0 to S+9)			K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	T, C, D Y, M, S Note: 3 consecutive devices are used.		TCMP, TCMPP: 11 steps

**Operation**

S1, S2 and S3 represent hours, minutes and seconds respectively. This time is compared to the time value in the 3 data devices specified by the head address S. The result is indicated in the 3 bit devices specified by the head address D.

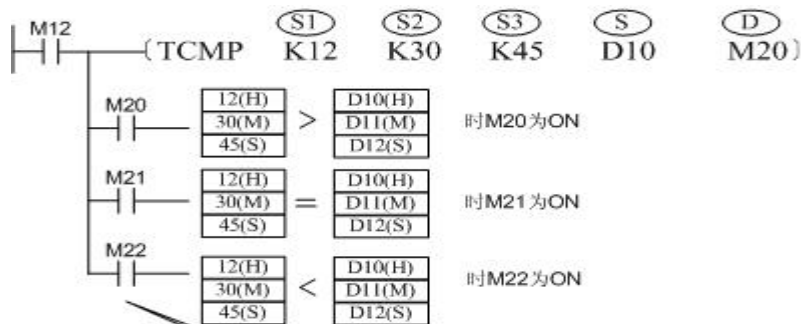
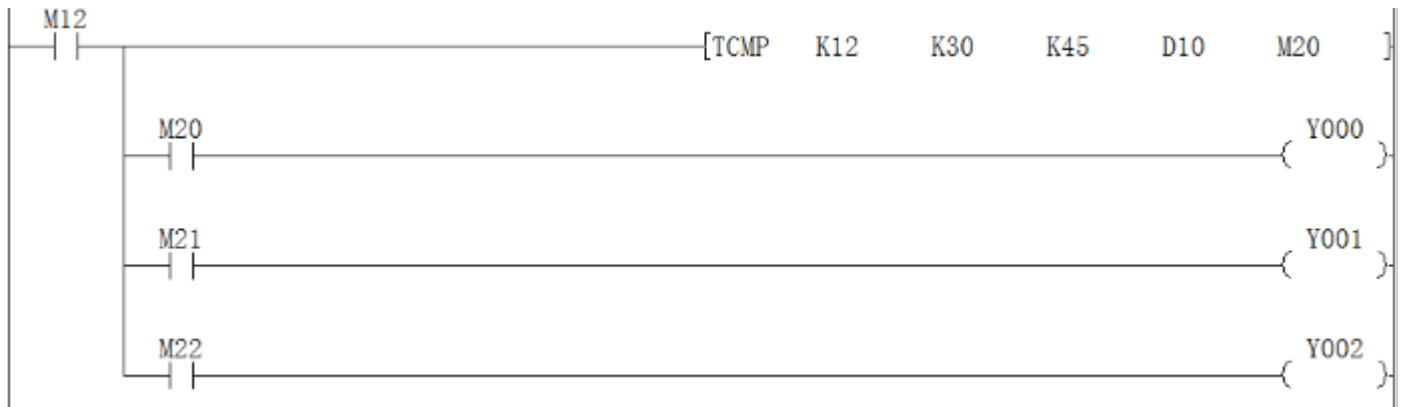
The bit devices in D indicate the following:

- 1) D+0 is set ON, when the time in S is less than the time in S1, S2 and S3.
- 2) D+1 is set ON, when the time in S is equal to the time in S1, S2 and S3.
- 3) D+2 is set ON, when the time in S is greater than the time in S1, S2 and S3.

**Points to note**

- 1) The status of the destination devices is kept, even if the TCMP instruction is deactivated.
- 2) The comparison is based on the time value specified in the source devices.
  - The valid range of values for S1 and S+0 is 0 to 23 (Hours).
  - The valid range of values for S2 and S+1 is 0 to 59 (Minutes).
  - The valid range of values for S3 and S+2 is 0 to 59 (Seconds).
- 3) The current time of the real time clock can be compared by specifying D8015 (Hours), D8014 (Minutes) and D8013 (Seconds) as the devices for S1, S2 and S3 respectively

Program example



If M12 =ON, one of M20~M22 will be set to ON. When M12 is changed from ON to OFF, TCMP instruction will not be executed, and M20/M21/M22 will remain the state before M12=OFF. RST or ZRST can be used to clear the comparison result of M20~M22. The result of  $\geq$ ,  $\leq$ ,  $\neq$  can be obtained by series or parallel-connection of M20~M22.

4.2.95 TKY instruction

Instruction Description

Table 4-122

Name	Function	Devices			Format	Steps
		S	D1	D2		
TKY (Ten key input)	Reads 10 devices with associated decimal values into a single number	X, Y, M, S Note: uses 10 Consecutive devices (identified as S+0 to S+9)	KnY, KnM, KnS, T, C, D, V, Z Note: uses 2 consecutive devices for 32 bit operation	Y, M, S Note: uses 11 consecutive devices (identified D2+0 to D2+10)		TKY: 7 steps DTK: 13 steps

**Operation**

This instruction can read from 10 consecutive devices(S+0 to S+9) and will store an entered numeric string in device D1.

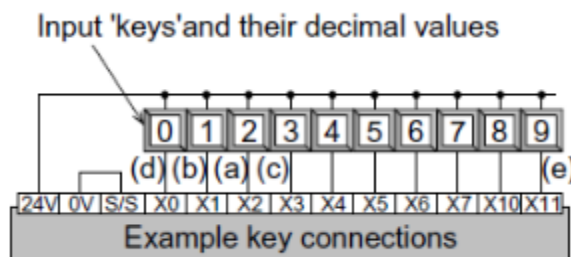
(S) is the starting input port of pressing key, occupying the following ten bit units (such as X port);

(D1) is the storage unit for inputted value;

(D2) is the temp starting unit for state of current pressing key group, occupying the following eleven bit units

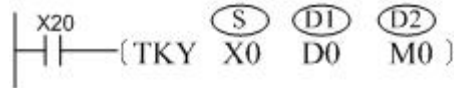
**Points to note**

- 1) When a source device becomes active its associated destination (bit) device D2 also becomes active. This destination device will remain active until another source device is operated. Each source device maps directly to its own D2 device, i.e. S+0 maps to D2+0, S+7 maps to D2+7 etc. These in turn, map directly to decimal values which are then stored in the destination data devices specified by D1.
- 2) One source device may be active at any one time. The destination device D2+10 is used to signify that a key (one of the 10 source devices) has been pressed. D2+10 will remain active for as long as the key is held down. When the TKY instruction is active, every press of a key adds that digit to the stored number in D1. When the TKY is OFF, all of the D2 devices are reset, but the data value in D1 remains intact.
- 3) When the TKY instruction is used with 16 bit operation, D1 can store numbers from 0 to 9,999 i.e. max. 4 digits. When the DTKY instruction is used (32 bit operation) values of 0 to 99,999,999 (max. 8 digits) can be accommodated in two consecutive devices D1 and D1+1. In both cases if the number to be stored exceeds the allowable ranges, the highest digits will overflow until an allowable number is reached. The overflowed digits are lost and can no longer be accessed by the user. Leading zero's are not accommodated, i.e. 0127 will actually be stored as 127 only.
- 4) The TKY instruction may only be used **ONCE**.
- 5) Using the above instruction as a brief example: If the 'keys' identified (a) to (d) are pressed in that order the number 2,130 will be entered into D1. If the key identified as (e) is then pressed the value in D1 will become 1,309. The initial '2' has been lost.

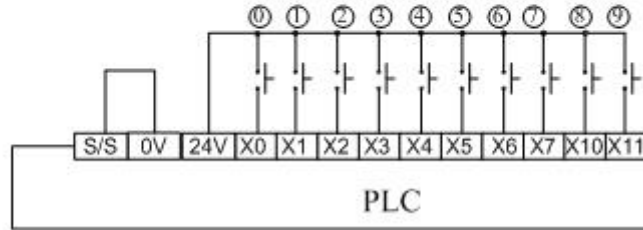




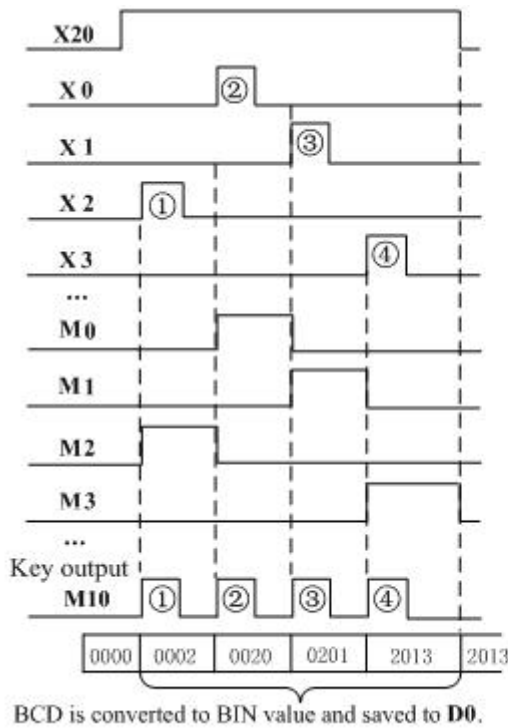
Program Example



The corresponding hardware wiring is shown in below figure.



If you want to input "2013", just pressing key 2, 0, 1, 3 in order. The operation of PLC internal variable is shown as Figure 4-9.



Set by parameters in an instruction, X0~X11 respectively correspond to numeric keys 0~9; M0~M9 correspond to the status of keys; Key output unit will be reset whenever a key is pressed.

Key values (e.g. 2013) are converted to BIN and saved to the designated **(D1)** unit D; (D0=0x7DD), D0 will never change even when the power flow turns OFF.

When several keys are pressed simultaneously, the key which is firstly detected is valid; if the number entered is more than 4 digits, the first entered number will overflow and only a 4-digit number is left.

Figure 4-9

If using 32bit instruction (DTKY), and **(D1)** occupies 32bit variable. For the above case, they are D1, D0, which is higher word and lower word respectively.

### 4.2.96 TO instruction

#### Instruction Description

Table 4-123

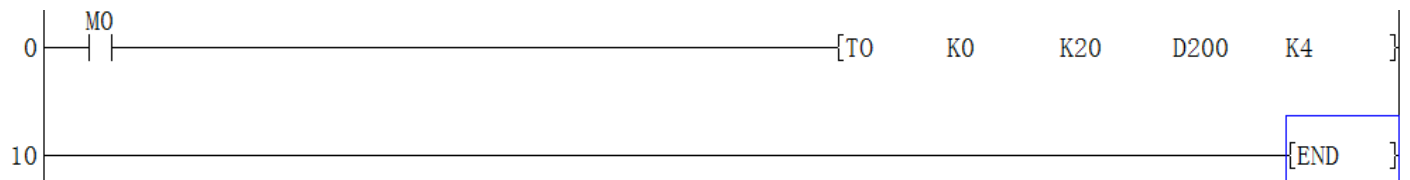
Name	Function	Devices				Format	Steps
		m1	m2	S	n		
TO (TO)	Writes data to the buffer memories of attached special function blocks	K, H ) Note: m1= 0 to 15	K, H ) Note: 0 to 3276 7	KnY, KnM, KnS, T, C, D, V, Z	K, H ) Note : 0 to 3276 7		TO, TOP: 9 steps DTO, DTOP: 17 steps

#### Operation

The TO instruction writes the data n words of data starting into the BFM register(buffer memory address) in the special extended module. The written data is taken from the PLC's head address S for n word devices.

- 1) **m1** is the which number special module close to PLC.
- 2) **m2** is the special module BFM address ID number inside the special module.
- 3) **S** is written data is taken from the PLC's head address S for n word devices.
- 4) **n** how many head address PLC's address will write into BFM address.

#### Program Example



When M0=ON, D200 reads K4 word of data starting from the zero module's BFM 20 register (D200---->BFM20, D201---->BFM21, D202---->BFM22,D203---->BFM23)

#### Note

- 1) When M8164=ON and implementing FROM/TO instruction, the content of special data register D8164 (the specified transmission number register of FROM/TO instruction) will be taken as n;
- 2) FROM/TO instruction is a time-consuming operation. When multiple FROM/TO instruments is implemented or multiple buffer memory data is transmitted, the PLC scanning period will be long. In order to prevent overtime, User can add WDT instruction for extending monitor timer cycle before FROM/TO instruction.
- 3) Please check the special Module program manual from our official website.

### 4.2.97 TRD instruction

#### Instruction Description

Table 4-124

Name	Function	Devices			Format	Steps
		S1	S2	D		
TRD (Time Read)	Reads the current value of the real time clock to a group of registers	T, C, D Note: 7 consecutive devices are used.				TRD, TRDP: 5 steps

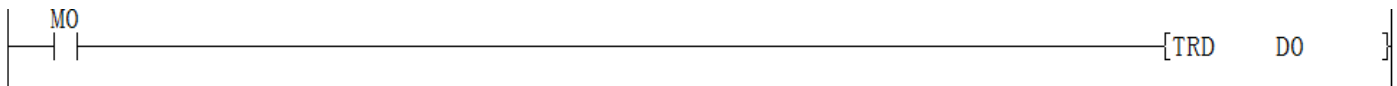
#### Operation

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

Table 4-125

Device	Meaning	Values		Device	Meaning
D8018	Year	00-99	⇒	D+0	Year
D8017	Month	01-12	⇒	D+1	Month
D8016	Date	01-31	⇒	D+2	Date
D8015	Hours	00-23	⇒	D+3	Hours
D8014	Minutes	00-59	⇒	D+4	Minutes
D8013	Seconds	00-59	⇒	D+5	Seconds
D8019	Day	0-6(Sun-Sat)	⇒	D+6	Day

#### Program example



#### Points to note

The year is read as a two digit number. This can be change to a 4 digit number by setting D8018 to 2000 during the first program scan; see following program extract.



### 4.2.98 TSUB instruction

#### Instruction Description

Table 4-126

Name	Function	Devices			Format	Steps
		S1	S2	D		
TSUB (Time Subtraction)	Subtracts one time value from another to give a new time	T, C, D Note: 3 consecutive devices are used.				TSUB, TSUBP: 7 steps

#### Operation

Each of S1, S2 and D specify the head address of 3 data devices to be used a time value. The time value in S1 is subtracted from the time value in S2, the result is stored to D as a new time

#### Points to note

- 1) The subtraction is performed according to standard time values. Hours, minutes and seconds are kept within correct limits. Any underflow is correctly processed.

S1	S2	D														
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>D10: 10 hours</td></tr> <tr><td>D11: 30 mins</td></tr> <tr><td>D12: 27 secs</td></tr> <tr><td>10:30:27</td></tr> </table>	D10: 10 hours	D11: 30 mins	D12: 27 secs	10:30:27	-	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>D20: 3 hours</td></tr> <tr><td>D21: 10 mins</td></tr> <tr><td>D22: 49 secs</td></tr> <tr><td>03:10:49</td></tr> </table>	D20: 3 hours	D21: 10 mins	D22: 49 secs	03:10:49	=	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>D30: 7 hours</td></tr> <tr><td>D31: 19 mins</td></tr> <tr><td>D32: 38 secs</td></tr> <tr><td>07:19:38</td></tr> </table>	D30: 7 hours	D31: 19 mins	D32: 38 secs	07:19:38
D10: 10 hours																
D11: 30 mins																
D12: 27 secs																
10:30:27																
D20: 3 hours																
D21: 10 mins																
D22: 49 secs																
03:10:49																
D30: 7 hours																
D31: 19 mins																
D32: 38 secs																
07:19:38																

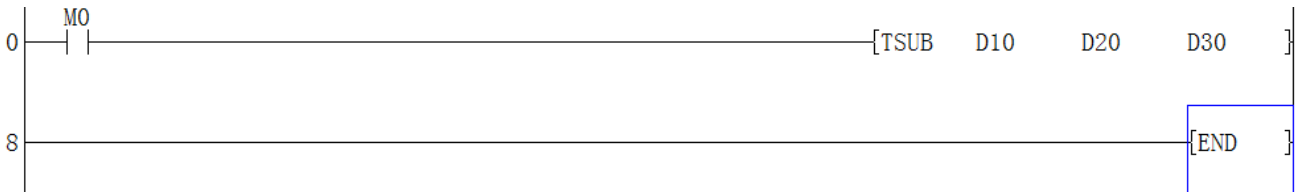
- 2) If the subtraction of the two times results in a value less than 00:00:00 hours, the value of the result is the time remaining below 00:00:00 hours.

S1	S2	D														
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>D10: 10 hours</td></tr> <tr><td>D11: 17 mins</td></tr> <tr><td>D12: 29 secs</td></tr> <tr><td>10:17:29</td></tr> </table>	D10: 10 hours	D11: 17 mins	D12: 29 secs	10:17:29	-	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>D20: 18 hours</td></tr> <tr><td>D21: 12 mins</td></tr> <tr><td>D22: 34 secs</td></tr> <tr><td>18:12:34</td></tr> </table>	D20: 18 hours	D21: 12 mins	D22: 34 secs	18:12:34	=	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>D30: 13 hours</td></tr> <tr><td>D31: 41 mins</td></tr> <tr><td>D32: 16 secs</td></tr> <tr><td>16:04:55</td></tr> </table>	D30: 13 hours	D31: 41 mins	D32: 16 secs	16:04:55
D10: 10 hours																
D11: 17 mins																
D12: 29 secs																
10:17:29																
D20: 18 hours																
D21: 12 mins																
D22: 34 secs																
18:12:34																
D30: 13 hours																
D31: 41 mins																
D32: 16 secs																
16:04:55																
		M8021 ON														

When this happens the borrow flag M8021 is set ON.

- 3) If the subtraction of the two times results in a value of zero (00:00:00 hours) then the zero flag M8020 is set ON.
- 4) The same device may be used as a source (S1 or S2) and destination device. In this case the subtraction is continually executed; the destination value changing each program scan. To prevent this from happening, use the pulse modifier or an interlock program.

**Program Example**



**4.2.99 TTMR instruction**

**Instruction Description**

Table 4-127

Name	Function	Devices		Format	Steps
		D	n		
TTMR (Teaching timer)	Monitors the duration of a signal and places the timed data into a data register	D Note: 2 devices 16 bit words are used D and D+1	K, H) Note: n= 0: (D) = (D+1) × 1 n= 1: (D) = (D+1) × 10 n= 2: (D) = (D+1) × 100		TTMR: 5 steps

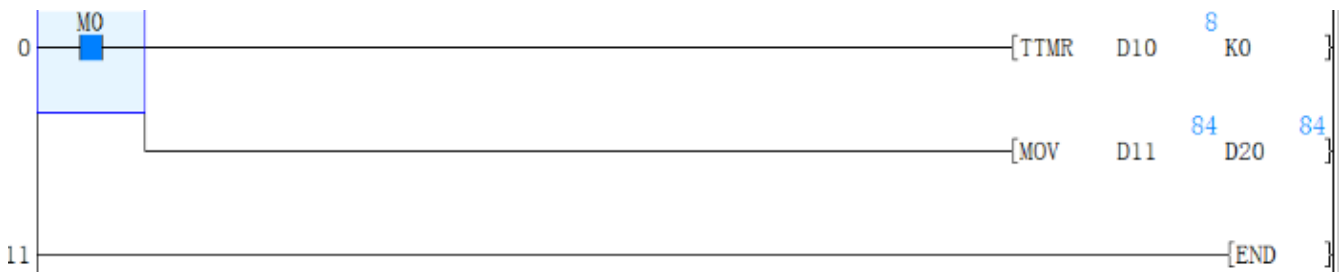
**Operation**

The duration of time that the TTMR instruction is energized, is measured and stored in device D+1 (as a count of 100ms periods). The data value of D+1 (in seconds), multiplied by the factor selected by the operand n, is moved in to register D. The contents of D could be used as the source data for an indirect timer setting or even as raw data for manipulation. When the TTMR instruction is de-energized D+1 is automatically reset (D is unchanged).

**Program Example**

1) Example 1:

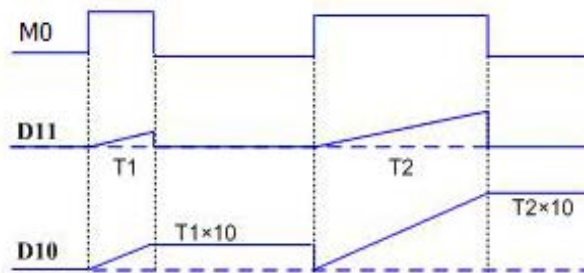
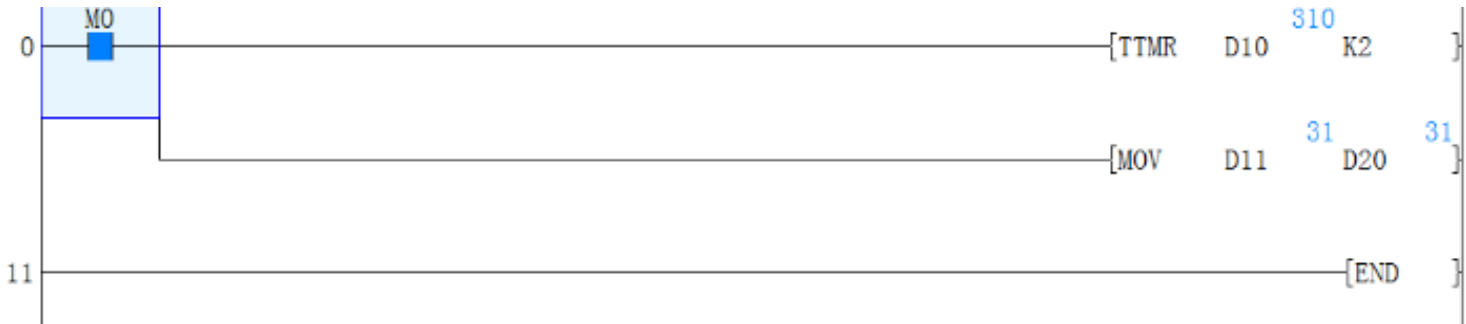
When M0=OFF, then D11 will be cleared. When M0=ON and n=0, then D10=D11\*0.1



When M0=OFF, then D11 will clear. When M0=ON and n=1, then D10=D11

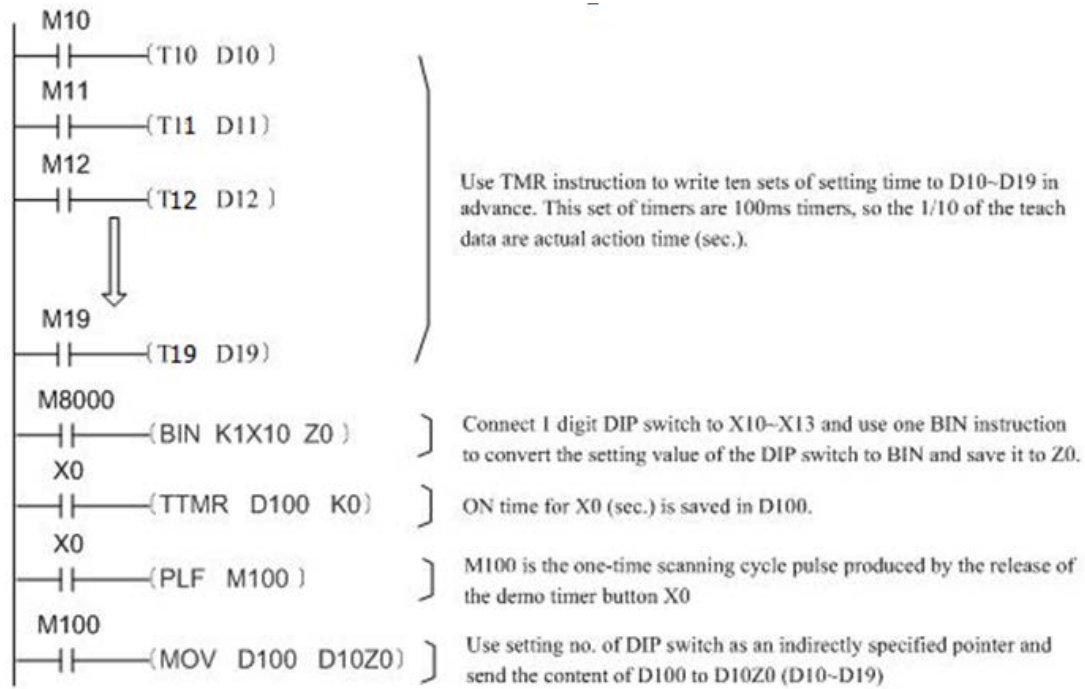


When M0=OFF, then D10 will be cleared. When M0=ON and n=2, then  $D10=D11 \times K10 \rightarrow 310=31 \times 10$



n	D 10	D 11(unit: 10 ms)
K0(unit: sec)	1×T	D 11=D10×10
K1(unit: 100 ms)	10×T	D 11=D10
K2(unit: 10 ms)	100×T	D 11=D10/10

2) Example 2 for instruction:





### 4.2.100 TWR instruction

#### Instruction Description

Table 4-128

Name	Function	Devices	Format	Steps
		D		
TWR (Time Write)	Sets the real time clock to the value stored in a group of registers	T, C, D Note: 7 consecutive devices are used.		TWR, TWRP: 5 steps

#### Operation

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

#### The seven devices

Device	Meaning	Value
S+0	Year	0--99
S+1	Month	1--12
S+2	Day	1--31
S+3	Hour	0--23
S+4	Minute	0--59
S+5	Second	0--59
S+6	Week	0--6(Sun-Sat)

- 
- 
- 
- 
- 
- 
- 

Device	Meaning
D8018	Year
D8017	Month
D8016	Day
D8015	Hour
D8014	Minute
D8013	Second
D8012	Week



**Points to note:**

This instruction removes the need to use M8015 during real time clock setting. When setting the time it is a good idea to set the source data to a time a number of minutes ahead and then drive the instruction when the real time reaches this value.

**Program example:**

1) Example 1

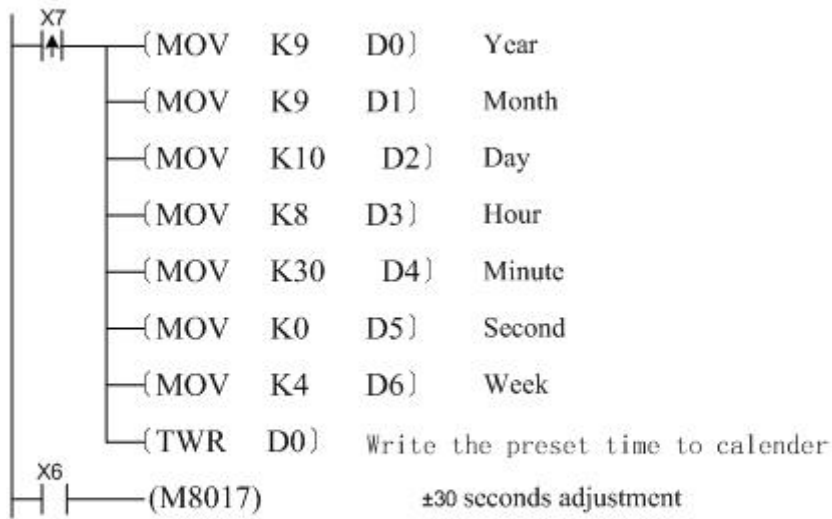
- Note that the seven data are whole written when clock is written. Any variable can not be lacked when you preset the value. If week is not written, the default is 0 for Sunday; if month source address is not written, the month variable is 0, and PLC believes that the month you provide is wrong. Thus the clock change is invalid.
- Once M8017 produce one ON, PLC internal clock does  $\pm 30$  correction action. Where the correction means that when the PLC's internal clock second hand is in 1~29, clock will be automatically classified as "0" seconds and minute hand does not act; in 30 ~9, it will also be automatically classified as "0" seconds, minutes plus 1 minute. M8015 set ON to stop the clock timing.
- In the usual case it shows only 2 digits (for example: in 2009 only show 09), If you hope that "year" shows four digits format, execute the following statements in one scan cycle:



If D8018=09 at the first time, D8018=2009 after switch. PLC internal clock is as follows.

2) Example 2 for instruction:

Change current time in PLC to Thursday, 0 second, 8:30, Sept. 10. 2009



Write the time to D0~D6 in ahead of a period of time. X7 will be turned ON to write the correct time to PLC when the actual time is due.

±30 seconds adjustment can be made as soon as M8017 is turned ON.

**Note:**

Usually users have to modify PLC clock. Write the clock into D8013~D8019 by TWR instruction. Do not use the MOV instruction for direct assignment of the D8012~D8018.

**4.2.101 TZCP instruction**

**Instruction Description**

Table 4-129

Name	Function	Devices				Format	Steps
		S1	S2	S	D		
TZCP (Time Zone Compare)	Compares a time to a specified time range - results of <, = and > are given	T, C, D S1 must be less than or equal to S2. Note: 3 consecutive devices are used for all			Y, M, S	<p>M0 ON when D0,D1,D2 &lt; D20,D21,D22 M1 ON when D20,D21,D22 ≤ D0,D1,D2 ≤ D30,D31,D32 M2 ON when D30,D31,D32 &lt; D0,D1,D2</p>	TZCP, TZC PP:9 steps

**Operation**

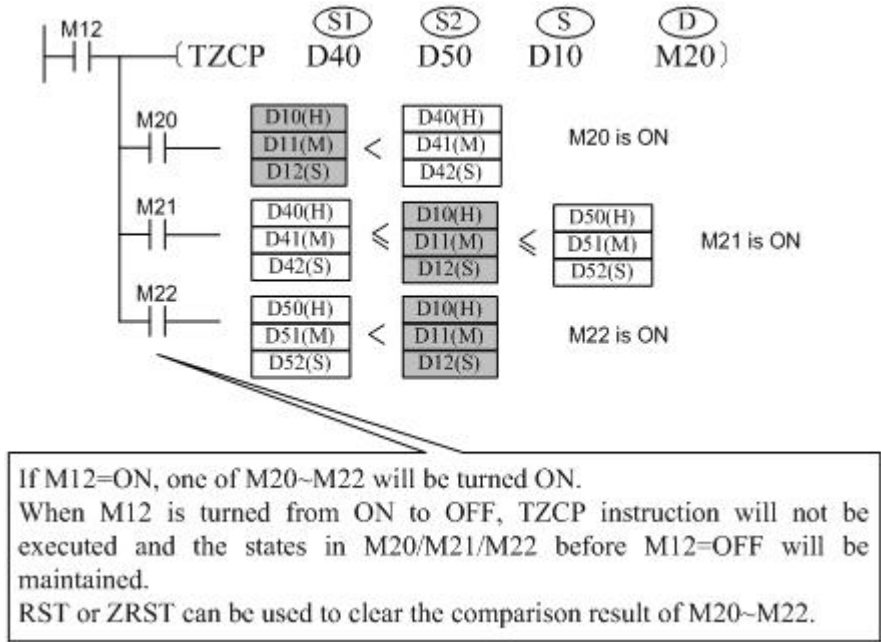
S1, S2 and S represent time values. Each specifying the head address of 3 data devices. S is compared to the time period defined by S1 and S2. The result is indicated in the 3 bit devices specified by the head address D.

The bit devices in D indicate the following:

- 1) D+0 is set ON, when the time in S is less than the times in S1 and S2.

- 2) D+1 is set ON, when the time in S is between the times in S1 and S2.
- 3) D+2 is set ON, when the time in S is greater than the times in S1 and S2.

**Program Example**



**4.2.102 WAND instruction**

**Instruction Description**

Table 4-130

Name	Function	Devices			Format	Steps
		S1	S2	D		
WAND (Logical word AND)	A logical AND is performed on the source devices -result stored at destination	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z			WAND, WANDP:7 steps DAND, DANDP:13 steps

**Operation**

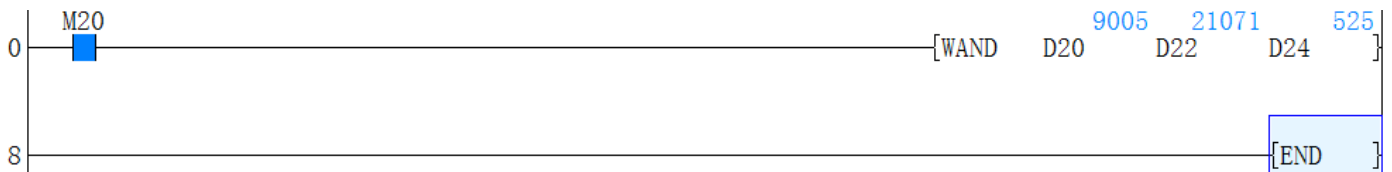
The bit patterns of the two source devices are analyzed (the contents of S2 is compared against the contents of S1). The result of the logical AND analysis is stored in the destination device (D). The following rules are used to determine the result of a logical AND operation. This takes place for every bit contained within the source devices:

General rule: (S1) Bit n WAND (S2) Bit n = (D) Bit n

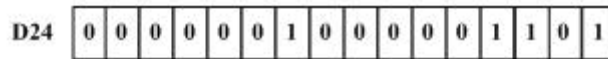
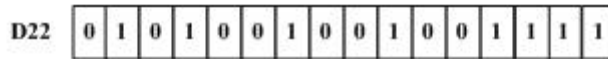
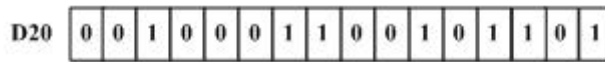
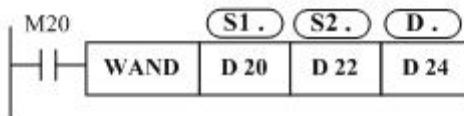
1 WAND 1 = 1 0 WAND 1 = 0

1 WAND 0 = 0 0 WAND 0 = 0

**Program Example:**



Logic AND



Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D20	1	0	1	1	0	1	0	0	1	1	0	0	0	1	0	0	9005
D21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D22	1	1	1	1	0	0	1	0	0	1	0	0	1	0	1	0	21071
D23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D24	1	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	525
D25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**4.2.103 WDT instruction**

**Instruction Description**

Table 4-131

Name	Function	Devices	Format	Steps
		D		
WDT (Watch dog timer refresh)	Used to refresh the watch dog timer during a program scan	N/A Can be driven at any time within the main program body		WDT, WDTP: 1 step

**Operation**

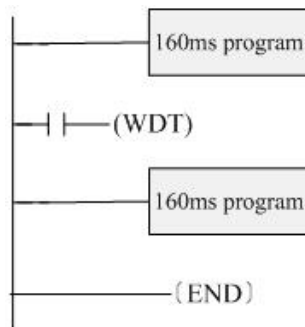
The WDT instruction refreshes the PLC’s watchdog timer. The watchdog timer checks that the program scan (operation) time does not exceed an arbitrary time limit. It is assumed that if this time limit is exceeded there is an error at some point. The PLC will then stop operation to prevent any further errors from occurring. By causing the watchdog timer to refresh (driving the WDT instruction) the usable scan (program operation) time is effectively increased.

If the operation of user's program is too complex (for example, too many Cycles of calculation), an error may occur when the implementation of programming running out . If necessary, the program can use WDT instruction (for example, between the FOR ~ NEXT instruction can insert the instructions); If the program's scan time is longer than the value of D8000 (default 200ms), we can insert the WDT instructions. The program will be divided into many pieces, Every piece’s scan time is less than 200ms or change the setting value of D8000.

**Points to note**

- 1) When the WDT instruction is used, it will operate on every program scan so long as its input condition has been made.  
To force the WDT instruction to operate for only ONE scan requires the user to program some form of interlock.
- 2) The watchdog timer has a default setting of 200 msec. This time limit may be customized to a user’s own requirement by editing the contents of data register D8000, the watchdog timer register.

**Program Example**



This program scan time is 320ms. We can divide program into two parts with the WDT instruction, so that each part of the program scan time is less than 200ms (D8000 default value).

### 4.2.104 WOR instruction

#### Instruction Description

Table 4-132

Name	Function	Devices			Format	Steps
		S1	S2	D		
WOR (Logic word OR)	A logical OR is performed on the source devices -result stored at destination	K, H, KnX, KnY, KnM, KnS, T, C, D, V, Z	KnX, KnM, KnS, T, C, D, V, Z	KnY, KnM, KnS, T, C, D, V, Z		WOR, W ORP: 7 steps DOR, DORP: 13 steps

#### Operation

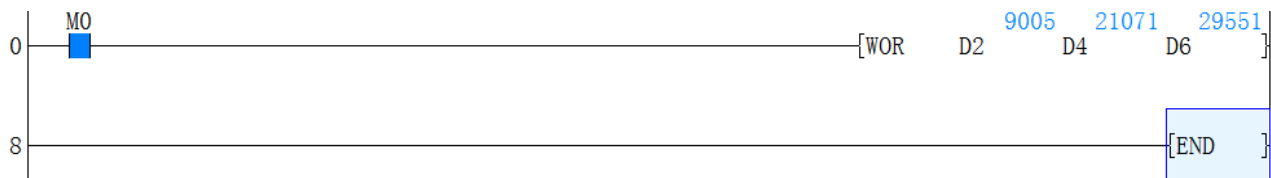
The bit patterns of the two source devices are analyzed (the contents of S2 is compared against the contents of S1). The result of the logical OR analysis is stored in the destination device (D). The following rules are used to determine the result of a logical OR operation. This takes place for every bit contained within the source devices:

General rule: (S1) Bit n WOR (S2) Bit n = (D) Bit n

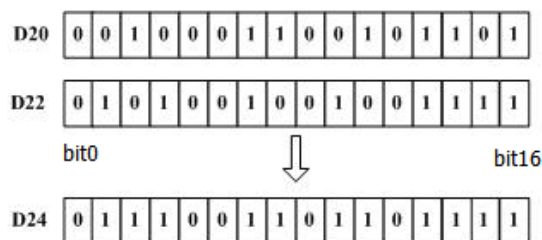
1 WOR 1 = 1 0 WOR 1 = 1

1 WOR 0 = 1 0 WOR 0 = 0

#### Program Example



Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D2	1	0	1	1	0	1	0	0	1	1	0	0	0	1	0	0	9005
D3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D4	1	1	1	1	0	0	1	0	0	1	0	0	1	0	1	0	21071
D5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D6	1	1	1	1	0	1	1	0	1	1	0	0	1	1	1	0	29551



### 4.2.105 WSFL instruction

#### Instruction Description

Table 4-133

Name	Function	Devices				Format	Steps
		S	D	n1	n2		
WSFL (Word shift left)	The value of the source devices are copied to a controlled word stack moving the existing data to the left	KnX, KnY, KnM, KnS, T, C, D	KnY, KnM, KnS,T ,C, D	K,H, Note: $n2 \leq n1$ $\leq 512$		WSFL, WSFLP: 9 steps	

#### Operation

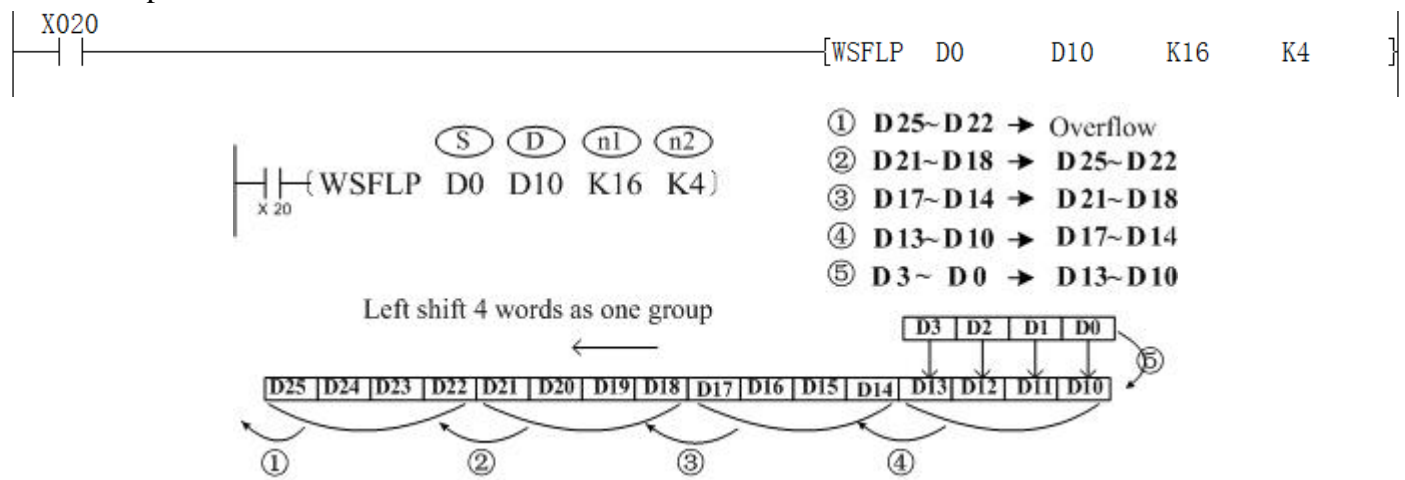
The instruction copies n2 source devices to a word stack of length n1. For each addition of n2 words, the existing data within the word stack is shifted n2 words to the left. Any word data moving to a position exceeding the n1 limit is diverted to an overflow area. The word shifting operation will occur every time the instruction is processed unless it is modified with either the pulse suffix or a controlled interlock.

#### Note:

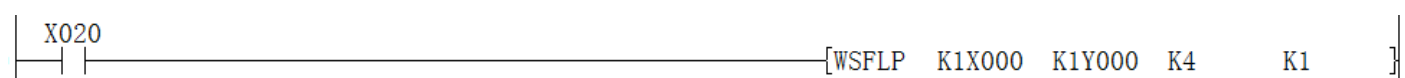
When using bit devices as source (S) and destination (D) the Kn value must be equal.

#### Program example

Example 1 for word:




Example 2 for bit :(it is similar with WSFR example 2.direction is opposite)



### 4.2.106 WSFR instruction

#### Instruction Description

Table 4-134

Name	Function	Devices				Format	Steps
		S	D	n1	n2		
WSFR (Word shift right)	The value of the source devices are copied to a controlled word stack moving the existing data to the right	KnX, KnY, KnM, KnS,T , C, D	KnY, KnM, KnS,T ,C,D	K,H, ) Note: $n2 \leq n1 \leq 512$			WSFR, WSFRP :9 steps

#### Operation

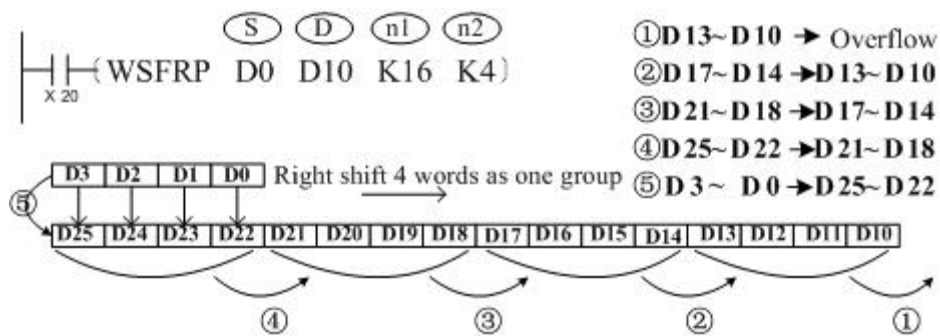
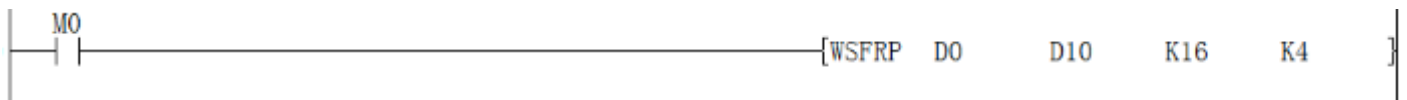
The instruction copies n2 source devices to a word stack of length n1. For each addition of n2 words, the existing data within the word stack is shifted n2 words to the right. Any word data moving to a position exceeding the n1 limit is diverted to an overflow area. The word shifting operation will occur every time the instruction is processed unless it is modified with either the pulse suffix or a controlled interlock.

#### Note:

When using bit devices as source (S) and destination (D) the Kn value must be equal.

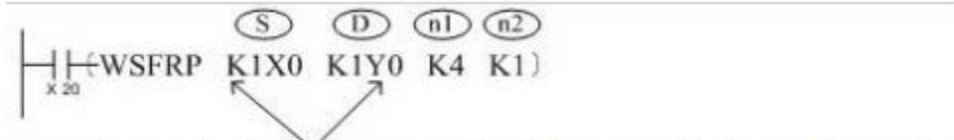
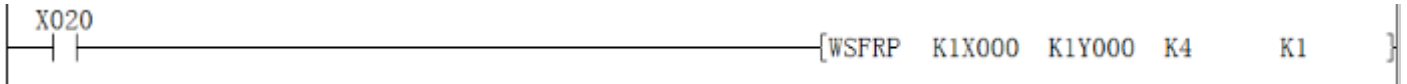
#### Program Example

Example 1 for word

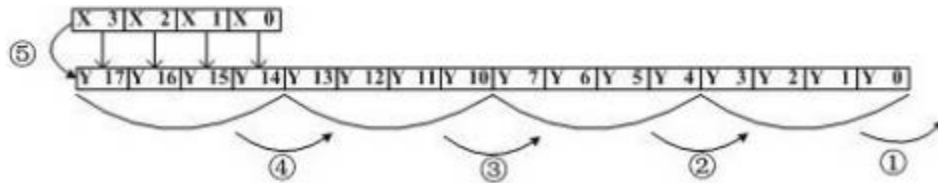




Example 2 for bit



Note: when using bit devices as source (S) and destination (D) the Kn value must be equal.



Right shift operation in one-time scanning is performed according to the following no. 1~5.

- 1: Y3~Y0 → carry bit
- 2: Y17~Y14 → Y13~Y10
- 3: Y13~Y10 → Y7~Y4
- 4: Y7~Y4 → Y3~Y0
- 5: X3~X00 → Y17~Y14 Completed

### 4.2.107 WXOR instruction

#### Instruction Description

Table 4-135

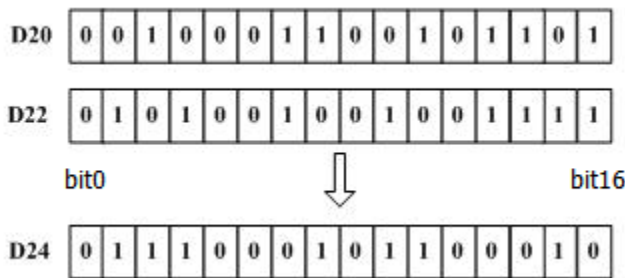
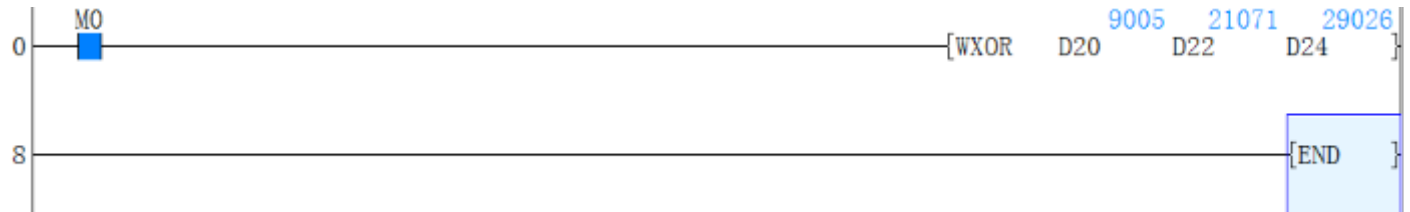
Name	Function	Devices			Format	Steps
		S1	S2	D		
WXOR (Logical Exclusive OR)	A logical XOR is performed on the source devices -result stored at destination	K, H KnX, KnY, KnM, KnS, T, C, D, V, Z		KnY, KnM, KnS, T, C, D, V, Z		WXOR, WXOR P:7 steps  DXOR, DXORP 13 steps.

#### Operation

The bit patterns of the two source devices are analyzed (the contents of S2 is compared against the contents of S1). The result of the logical XOR analysis is stored in the destination device (D). The following rules are used to determine the result of a logical XOR operation. This takes place for every bit contained within the source devices: General rule: (S1) Bit n WXOR (S2) Bit n = (D) Bit n

1 WXOR 1 = 0 0 WXOR 1 = 1  
 1 WXOR 0 = 1 0 WXOR 0 = 0

**Program Example**



Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D20	1	0	1	1	0	1	0	0	1	1	0	0	0	1	0	0	9005
D21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D22	1	1	1	1	0	0	1	0	0	1	0	0	1	0	1	0	21071
D23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D24	0	1	0	0	0	1	1	0	1	0	0	0	1	1	1	0	29026

**4.2.108 XCH instruction**

**Instruction Description**

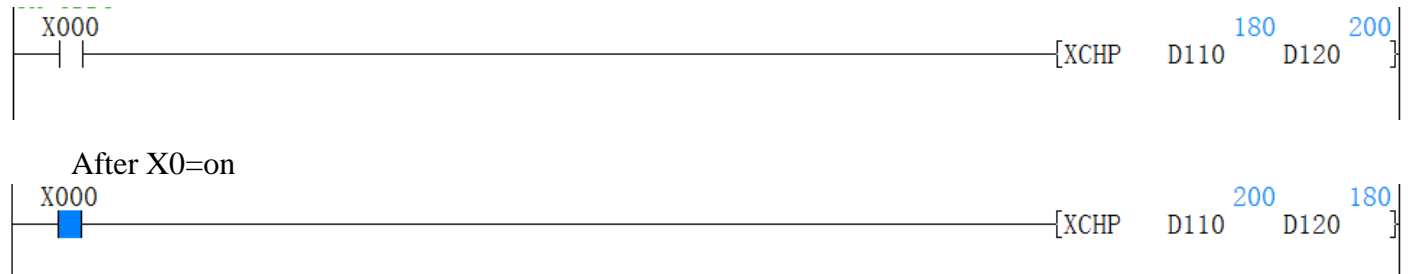
Table 4-136

Name	Function	Devices		Format	Steps
		D1	D2		
XCH (Exchange)	Data in the designated devices is exchanged	KnY, KnM, KnS, T, C, D, V, Z Note: when using the byte XCH (i.e.M8160 is ON) D1 and D2 must be the same device otherwise a program error will occur and M8067 will be turned ON			XCH, XCHP: 5 steps DXCH, DXCHP: 9 steps

**Operation 1**

The contents of the two destination devices D1 and D2 are swapped, i.e. The complete word devices are exchanged.

**Example 1**



**Operation 2**

This function is equivalent to SWAP the bytes within each word of the designated devices D1 are exchanged when 'byte mode flag' M8160 is ON. Please note that the mode will remain active until it is reset, i.e. M8160 is forced OFF.

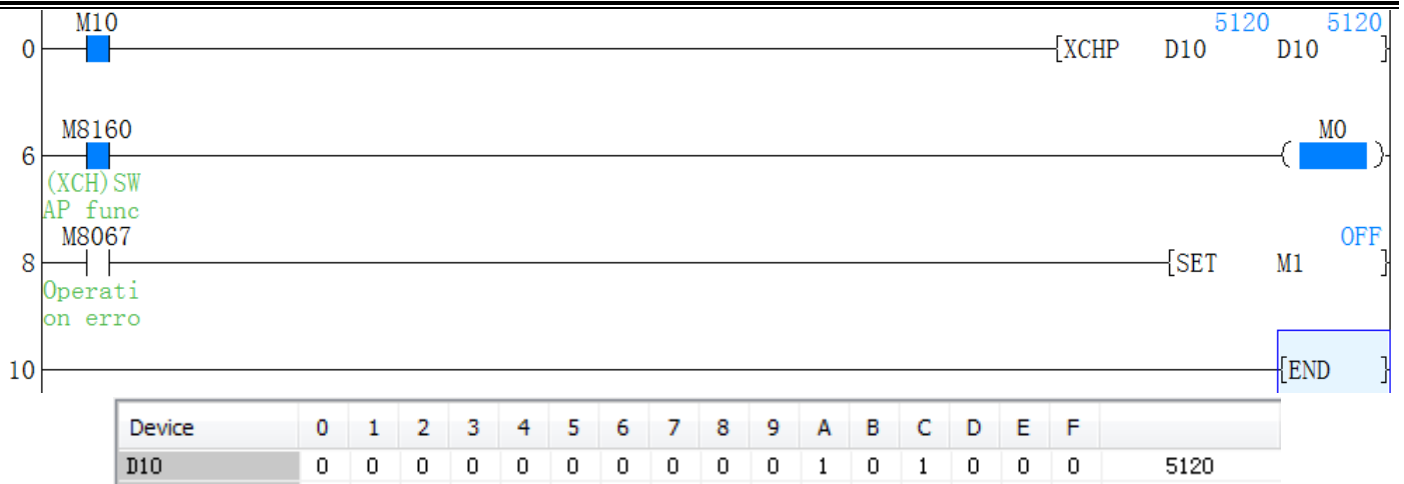
**Example 2**

Change D10 H 8bit and D10 L 8bit. M8160=ON, M10 set on in first time.



Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D10	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	20

M8160=ON, M10 set on in second time.



### 4.2.109 ZCP instruction

#### Instruction Description

Table 4-137

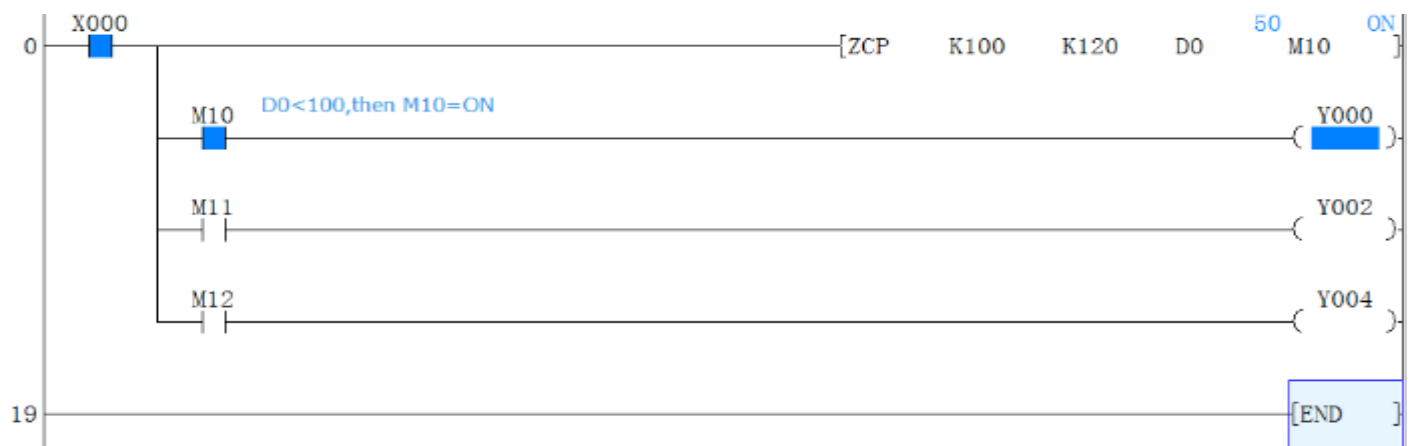
Name	Function	Devices				Format	Steps
		S 1	S2	S3	n		
ZCP (Zone compare)	Compares a data value against a data range - results of <, = and > are given.	K, H, KnX, KnM, KnS, T, C, D, V, Z Note: S1 should be less than S2	KnY,	Y, M, S Note: 3 consecutive devices are used.	(ZCP S1 S2 S3 D) Example: (ZCP K100 K120 C 30 M 3)	ZCP, ZCPP: 9 steps DZCP, DZCPP: 17 steps	

#### Operation

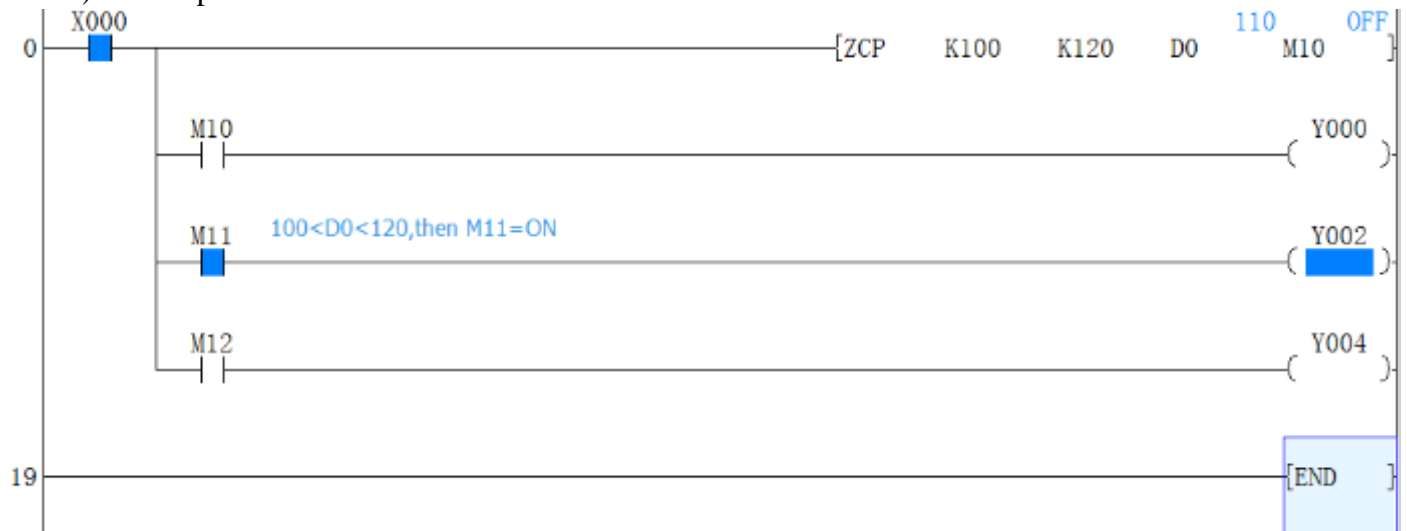
The operation is the same as the CMP instruction except a single data value (S3) is compared against a data range (S1-S2). S3 is less than S1 and S2 - bit device D is ON. S3 is equal to or between S1 and S2 - bit device D+1 is ON. S3 is greater than both S1 and S2 - bit device D+2 is ON.

#### Program Example

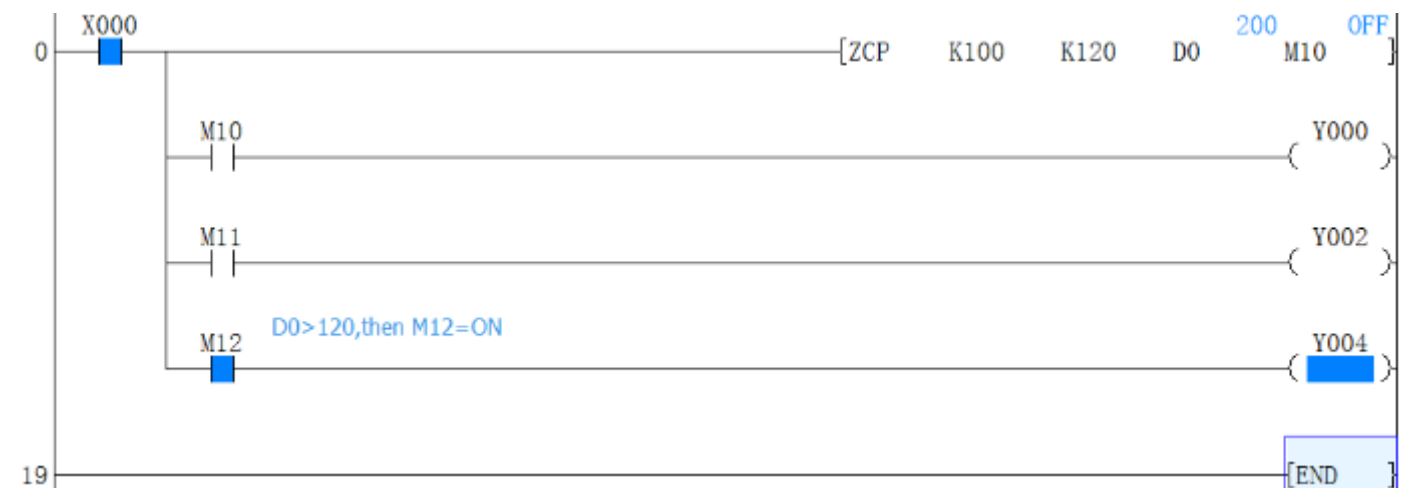
1) Example 1



2) Example 2



3) Example 3



4.2.110 ZRN instruction

Instruction Description

Table 4-138

Name	Function	Devices				Format	Steps
		S1	S2	S3	D		
ZRN Zero return	Return to zero home point after machine ON or initial setting.	K,H,KnX ,KnY, KnM,Kn S T,C,D,V, Z		X,Y, M,S	Y Note : ) Y00 0 to Y00 3		ZRN: 9 steps DZRN: 17 steps

## Operation

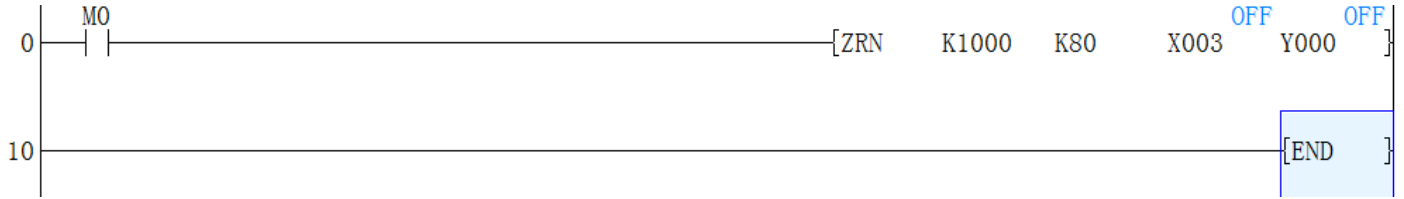
When executing incremental or absolute positioning, the PLC stores the current position values which increase or decrease during operation. Using these values, the PLC always knows the machine position. However when the power to the PLC is turned off, this data would be lost. To cope with this the machine should return to the zero point when the power is turned ON, or during initial set up, to teach the zero position.

[S1] is the Zero Return Speed, [S2] is the Creep Speed, [S3] is the Near Point Signal, and [D] is the Pulse Output Designation.

## Points to note

- 1) Users may specify zero return speed [S1] as, 16-bit 10 to 32,767Hz or 32-bit 10 to 100kHz.
- 2) Users may specify the creep speed [S2] of 10 to 32,767Hz
- 3) If any device other than an input relay (X) is specified for the Near point signal [S3] it will be affected by the operation cycle of the PLC and the dispersion of the zero point may be large.
- 4) Only Y000 to Y003 can be used for the pulse output [D]. Because of the nature of the high speed output, transistor type output units should be used with this instruction. Relay type outputs will suffer a greatly reduced life, and will cause false outputs to occur. To ensure a 'clean' output signal when using transistor type units, the load current should be 200mA or higher with the TP Series. The load current should be 10 - 100mA with the TP Series. It may be necessary to use 'pull up' resistors.
- 5) If M8140 is set to ON, the clear signal is sent to the servo motor when the return to zero point, Only when Y0/Y1 is pulse output port, M8140 is able. (If Y0 finish ZRN process, Y2 will set ON about 20ms, then Y2 set OFF. If Y1 finish ZRN process, Y2 will set ON about 20ms, then Y2 set OFF.)
- 6) Related device numbers.
  - D8141 (upper digit) & D8140 (lower digit): Current value register of Y000 (32-bit)
  - D8143 (upper digit) & D8142 (lower digit): Current value register of Y001 (32-bit)
  - D8151 (upper digit) & D8150 (lower digit): Current value register of Y002 (32-bit)
  - D8153 (upper digit) & D8152 (lower digit): Current value register of Y003 (32-bit)
  - M8145: Y000 pulse output stop (immediate)
  - M8146: Y001 pulse output stop (immediate)
  - M8147: Y000 pulse output monitor (BUS/READY)
  - M8148: Y001 pulse output monitor (BUS/READY)
  - M8149: Y002 pulse output monitor (BUS/READY)
  - M8150: Y003 pulse output monitor (BUS/READY)
  - M8151: Y002 pulse output stop (immediate)
  - M8152: Y003 pulse output stop (immediate)
  - M8129: Set ON, when to finish ZRN process.

**Program Example**



This instruction means that, after M10 turns ON, PLC send out pulses at speed of 1000Hz from Y0. Making stepper motor run back toward original point. While when X3(Near Point Signal) turns ON. The output pulse frequency turns into 80Hz creep speed, until X3(Near Point Signal) turns OFF again, and Y0 stops to output pulse,

**4.2.111 ZRST instruction**

**Instruction Description**

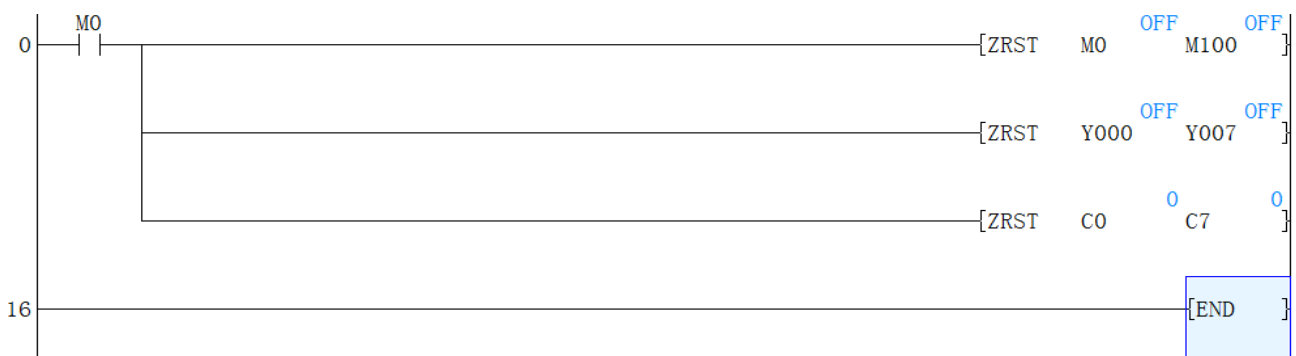
Table 4-139

Name	Function	Devices		Format	Steps
		S	D		
ZRST (Zone Reset)	Used to reset a range of like devices in one operation	Y, M,S,T, C, D Note: D1 must be less than or equal ( $\leq$ ) to D2. Standard and High speed counters cannot be mixed.		<pre>           M8002            -----  (ZRST M 500 M 599)         </pre>	ZRST, ZRSTP: 5 steps

**Operation**

The range of devices, including those specified as the two destinations are reset, i.e. for data devices the current value is set to 0 (zero) and for bit elements, the devices are turned OFF, i.e. also set to 0 (zero).The specified device range cannot contain mixed device types, i.e. C000 specified as the first destination device (D1) cannot be paired with T199 as the second destination device (D2). When resetting counters, standard and high speed counters cannot be reset as part of the same range. If D1 is greater than ( $>$ ) D2 then only device D1 is reset.

**Program Example**



### 4.2.112 AND Comparisons instructions

#### Instruction Description

Table 4-140

Name	Function	Devices		Format	Steps
		S	D		
AND (AND compa re) where is =, >, <, <>	Serial comparison contact. Active when the comparison S1 compares with S2 is true.	K,H, KnX, KnY, KnM, KnS, T, C, D, V, Z	K,H, KnX, KnY, KnM, KnS, T, C, D, V, Z		ADD, ADDP: 7 steps  DADD, DADDP : 13 steps.

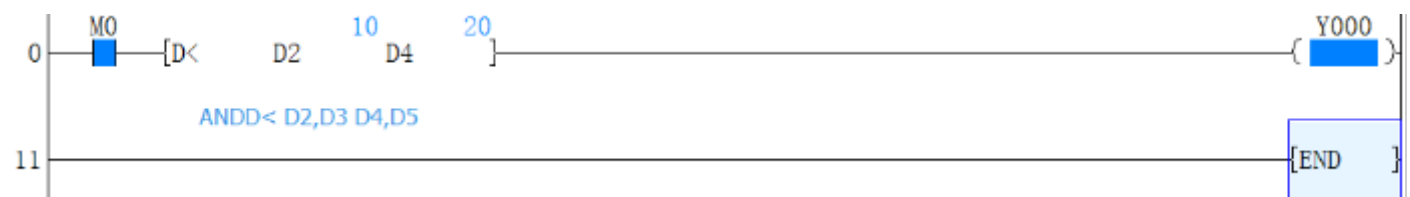
#### Operation

The value of S1 and S2 are tested according to the comparison of the instruction. If the comparison is true then the AND contact is active. If the comparison is false then the AND contact is not active.

#### Points to note

The AND comparison functions can be placed anywhere in a program that a standard AND instruction can be placed. I.e., it is a serial connection contact.

#### Program example





### 4.2.113 LD Comparisons instructions

#### Instruction Description

Table 4-141

Name	Function	Devices		Format	Steps
		S	D		
LDr (Load compare) where r is =, >, <, <>, ≤, ≥	Initial comparison contact. Active when the comparison S1 r S2 is true.	K,H, KnX, KnY, KnM, KnS, T, C, D, V, Z	K,H, KnX, KnY, KnM, KnS, T, C, D, V, Z		LDr: 5 steps DLDr: 9 steps

#### Operation

The value of S1 and S2 are tested according to the comparison of the instruction. If the comparison is true then the LD contact is active. If the comparison is false then the LD contact is not active.

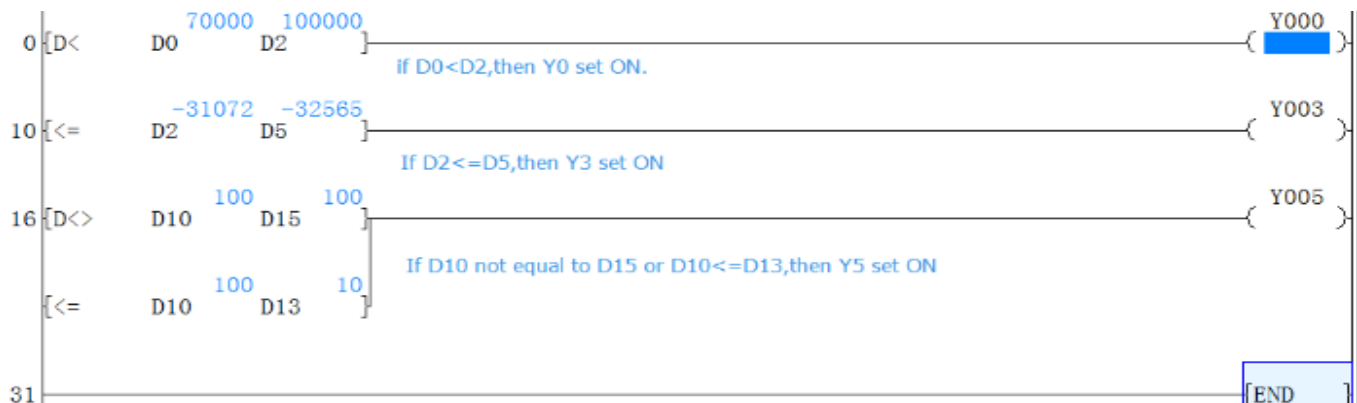
#### Points to note

The LD comparison functions can be placed anywhere in a program that a standard LD instruction can be placed. I.e., it always starts a new block.

Table 4-142

Mnemonic		Active when	Inactive when
16 bit	32 bit		
LD =	LDD =	S1 = S2	S1 ≠ S2
LD >	LDD >	S1 > S2	S1 ≤ S2
LD <	LDD <	S1 < S2	S1 ≥ S2
LD <>	LDD <>	S1 ≠ S2	S1 = S2
LD ≤	LDD ≤	S1 ≤ S2	S1 > S2
LD ≥	LDD ≥	S1 ≥ S2	S1 < S2

#### Program example



### 4.2.114 OR Comparisons instructions

#### Instruction Description

Table 4-143

Name	Function	Devices		Format	Steps
		S	D		
ORr (OR compar e) where r is =, >, <, <>, ≤, ≥	Parallel comparison contact. Active when the comparison S1 r S2 is true.	K,H, KnX, KnY, KnM, KnS, T, C, D, V, Z	K,H, KnX, KnY, KnM, KnS, T, C, D, V, Z		ORr: 5 steps DORr: 9 steps

#### Operation

The value of S1 and S2 are tested according to the comparison of the instruction. If the comparison is true then the OR contact is active. If the comparison is false then the OR contact is not active.

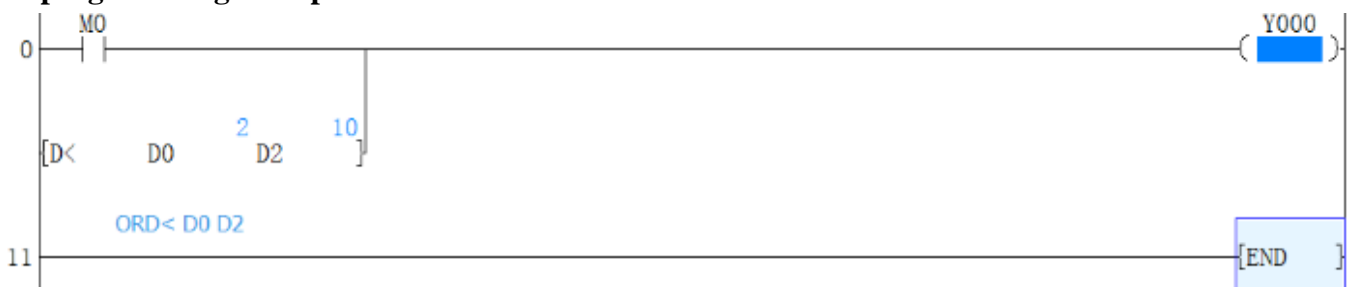
#### Points to note

The OR comparison functions can be placed anywhere in a program that a standard OR instruction can be placed. i.e., it is a parallel connection contact.

Table 4-144

Mnemonic		Active when	Inactive when
16 bit	32 bit		
OR =	ORD =	S1 = S2	S1 ≠ S2
OR >	ORD >	S1 > S2	S1 ≤ S2
OR <	ORD <	S1 < S2	S1 ≥ S2
OR <>	ORD <>	S1 ≠ S2	S1 = S2
OR ≤	ORD ≤	S1 ≤ S2	S1 > S2
OR ≥	ORD ≥	S1 ≥ S2	S1 < S2

#### The programming example



**Note:** When 32-bit counter (C200~C255) compares this instruction, be sure to use 32-bit instructions

## 5 STL instruction

STL programming is one of the basic programming instructions included in all TP Series PLC. The abbreviation STL actually means Step Ladder programming.

STL programming is a very simple concept to understand yet can provide the user with one of the most powerful programming techniques possible. The key to STL lies in its ability to allow the programmer to create an operational program which ‘flows’ and works in almost exactly the same manner as SFC. This is not a coincidence as this programming technique has been developed deliberately to achieve an easy to program and monitor system.

### 5.1 STL instruction list

Table 5-1

Instruction	Description	Page
STL	STL programming start instruction	
RET	STL programming end instruction	

### 5.2 STL instruction description

#### Instruction description

Table 5-2

Name	Function	Devices	Format	Steps
		S		
STL	Initiation of step procedure	S		1
RET	End of step procedure	N/C		

#### Operation

Step Control (STL) is controlled by several operating procedures (S0,S1.....Sn).

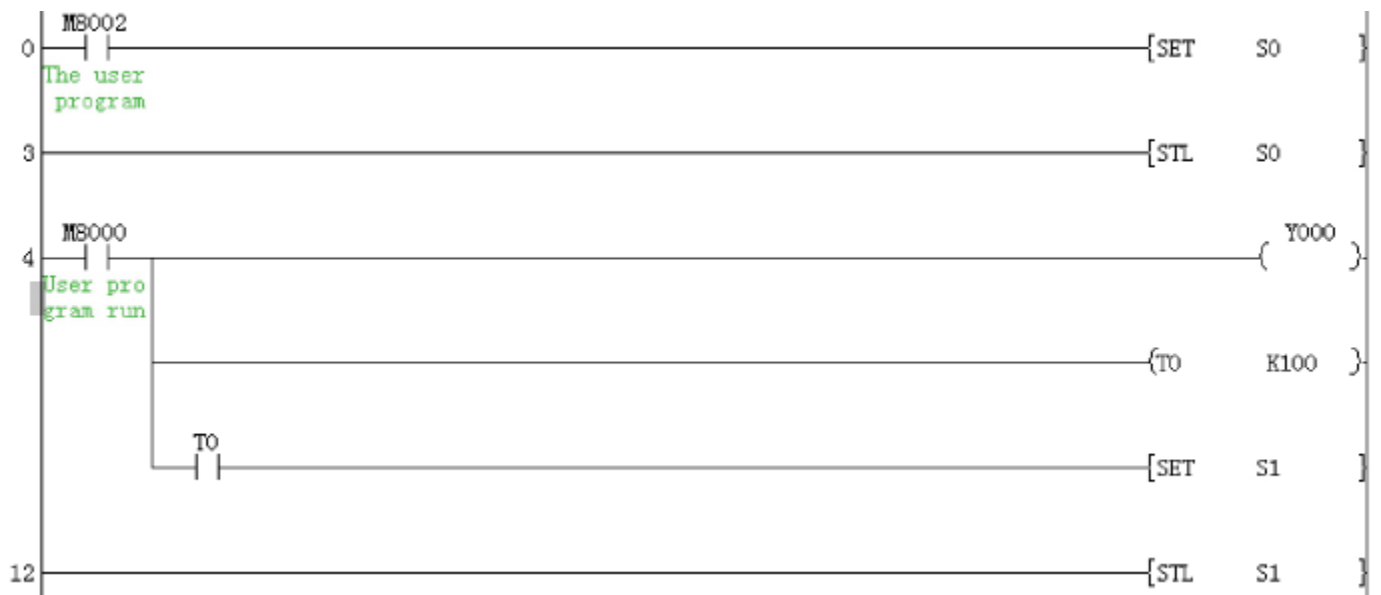
Step Control method's feature is that after taken into considerations for each control step and divided the complex procedure into successive steps, it greatly reduces the interdependence between each step and the

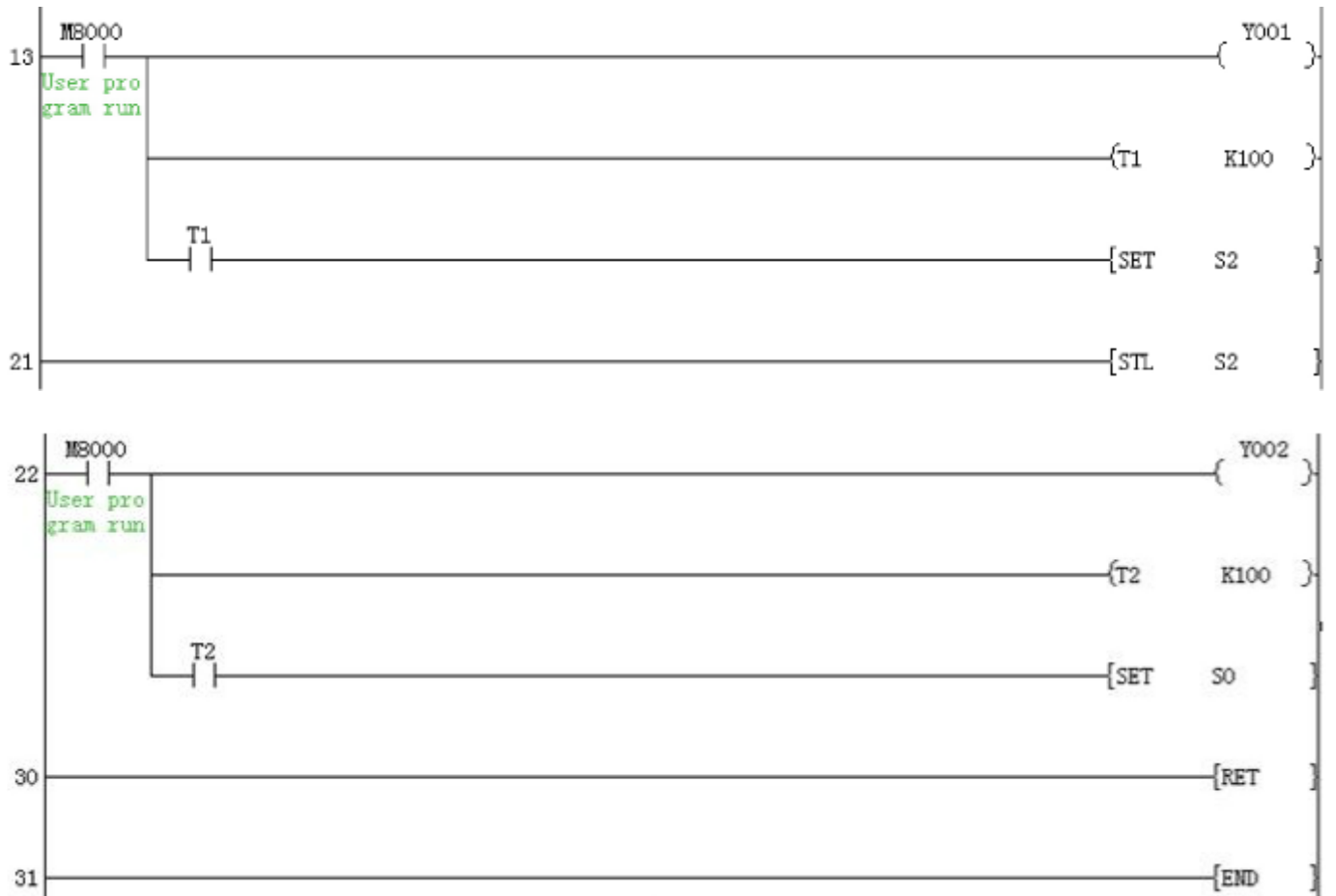




2) Automatic operation

- Y0=ON about 10 seconds. Then Y0=OFF and continue step2
- Y1=ON about 10 seconds. Then Y1=OFF and continue step3
- Y2=ON about 10 seconds. Then Y2=OFF and continue step1





# 6

## 7 System-specific address list

Table 6-1

M	Description	TPS	TPN/ TP	D	Description	TPS	TPN/ TP
<b>System operation</b>							
M8000	RUN monitor, NO contact	O	O	D8000	Watchdog timer	O	O
M8001	RUN monitor, NC contact	O	O	D8001	PLC type and version TP /3V-A2:250** TP -A1: 220** TP1: 251** ** is viewed by D8101	O	O
M8002	Initial pulse NO contact	O	O	D8002	Memory capacity 0002: 2K steps 0004: 4K steps 0008: 8K step	O	O
M8003	Initial pulse NC contact	O	O	D8003	Memory type default value is 0x10.	O	O
M8004	ON when one or more error flags from the range M8060to M8067 [except M8062]are ON	O	O	D8004	Error BCD code of M8060~M8067, the default value is 0.	O	O
M8005	Battery voltage Low	-	O	D8005	Battery voltage	-	O
M8006	Battery error latch	-	O	D8006	The level at which a battery voltage low is detected	-	O
M8007	Power loss has occurred more than 5ms, M8007&M8008 are ON	-	O	D8007	The number of time a momentary power failure has occurred since power ON.	-	O
M8008	Power loss has occurred	-	O	D8008	The time period before shutdown when a power failure occurs (default 10ms)	-	O
M8009	Power failure of 24V DC service supply	-	O	D8009	The device number of module, which affected by 24VDC power	-	O



					failure		
Clock Devices							
M8010	Reserved	O	O	D8010	Current operation cycle / scan time in units of 0.1 msec	O	O
M8011	Oscillates in 10 msec cycles	O	O	D8011	Minimum cycle/ scan time in units of 0.1 msec	O	O
M8012	Oscillates in 100 msec cycles	O	O	D8012	Maximum cycle/ scan time in units of 0.1 msec	O	O
M8013	Oscillates in 1 sec cycles	O	O	D8013	Seconds data for use with an RTC (0-59)	O	O
M8014	Oscillates in 1 min cycles	O	O	D8014	Minute data for use with an RTC (0-59)	O	O
M8015	When ON - clock stops, ON→OFF restarts clock	O	O	D8015	Hour data for use with an RTC (0-23)	O	O
M8016	When ON D8013 to 19 are frozen for display but clock continues	O	O	D8016	Day data for use with an RTC (1-31)	O	O
M8017	When pulsed ON set RTC to nearest minute	O	O	D8017	Month data for use with an RTC (1-12)	O	O
M8018	When ON Real Time Clock is installed	O	O	D8018	Year data for use with an RTC (2000-2099)	O	O
M8019	Clock data has been set out of range	O	O	D8019	Weekday data for use with an RTC (0-6)	O	O
Operation Flags							
M8020	Set when the result of an ADD or SUB is "0"	O	O	D8020	Input filter setting for devices X000 to X007 default is 10msec, (0-60)	O	O
M8021	Set when the result of a SUB is less than the min. negative number	O	O	D8021	Reserved		
M8022	Set when 'carry' occurs during an ADD or when an overflow occurs as a result of a data shift operation	O	O	D8022	Reserved		
M8023	Reserved	O	O	D8023	Reserved		

M8024	Direction of BMOV	-	O	D8024	Reserved		
M8025	HSC mode	-	O	D8025	Reserved		
M8026	RAMP mode	-	O	D8026	Reserved		
M8027	PR 16 element data string	-	O	D8027	Reserved		
M8028	Switch 100ms/10ms timer	O	-	D8028	Current value of the Z index register	O	O
M8029	Instruction execution complete such as PLSR	O	O	D8029	Current value of the V index register	O	O
<b>PLC Operation Mode</b>							
M8030	Battery voltage is low but BATT.V LED not lit	-	O	D8030	Reserved		
M8031	Clear all unsaved memory	O	O	D8031	Reserved		
M8032	Clear all the saved memory	O	O	D8032	Reserved		
M8033	The device statuses and settings are retained when the PLC changes from RUN to STOP and back into RUN	O	O	D8033	Reserved		
M8034	All of the physical switchgear for activating outputs is disabled. However, the program still operates normally.	O	O	D8034	Reserved		
M8035	Forced operation 1	O	O	D8035	Reserved		
M8036	Forced operation 2	O	O	D8036	Reserved		
M8037	Forced stop	O	O	D8037	Reserved		
M8038	Communication parameter setting flag	O	O	D8038	Reserved		
M8039	Constant scan	O	O	D8039	Constant scan time, default 0, in units of MS	O	O
<b>Step Ladder (STL) Flags</b>							
M8040	When ON STL state transfer is disabled	O	O	D8040	Up to 8 active STL states, from the range S0 to S899, are stored in D8040 to D8047 in ascending numerical order (Updated at END)	O	O
M8041	When ON STL transfer from initial state is enabled during automatic operation	O	O	D8041		O	O

M8042	A pulse output is given in response to a start input	O	O	D8042		O	O
M8043	On during the last state of ZERO RETURN mode	O	O	D8043		O	O
M8044	ON when the machine zero is detected	O	O	D8044		O	O
M8045	Disables the all output reset function when the operation mode is changed	O	O	D8045		O	O
M8046	ON when STL monitoring has been enable (M8047)	O	O	D8046		O	O
M8047	When ON D8040 to D8047 are enabled for active STL step monitoring	O	O	D8047		O	O
M8048	ON when annunciator monitoring has been enabled (M8049) and there is an active annunciator flag	-	O	D8048	Reserved		
M8049	When ON D8049 is enabled for actove annunciator state monitoring.	-	O	D8049	Stores the lowest currently active annunciator from the range S900 to S999 (Updated at END)	-	O
<b>Interrupt Control Flags</b>							
M8050	I00□ disabled	O	O	D8050	Reserved		
M8051	I10□ disabled	O	O	D8051	Reserved		
M8052	I20□ disabled	O	O	D8052	Reserved		
M8053	I30□ disabled	O	O	D8053	Reserved		
M8054	I40□ disabled	O	O	D8054	Reserved		
M8055	I50□ disabled	O	O	D8055	Reserved		
M8056	I6□□ disabled	-	O	D8056	Reserved		
M8057	I7□□ disabled	-	O	D8057	Reserved		
M8058	I8□□ disabled	-	O	D8058	Reserved		
M8059	Counters disabled	-	O	D8059	Reserved		
<b>Error Detection</b>							

M8060	I/O configuration error	-	O	D8060	The first I/O number of the unit or block causing the error	-	O
M8061	PLC hardware error	O	O	D8061	Error code for hardware error	O	O
M8062	PLC communication error	-	O	D8062	Error code for PLC Communications error	-	O
M8063	Parallel link error	O	O	D8063	Error code for parallel link error	O	O
M8064	Parameter error	O	O	D8064	Error code identifying parameter error	O	O
M8065	Syntax error	O	O	D8065	Error code identifying syntax error	O	O
M8066	Loop error	O	O	D8066	Error code identifying loop error	O	O
M8067	Operation error	O	O	D8067	Error code identifying operation error.	O	O
M8068	Operation error latch	O	O	D8068	Operation error step number latched	O	O
M8069	Reserved			D8069	Step numbers for found errors corresponding to flags M8065 to M8067	O	O
<b>High-speed ring counter</b>							
M8099	High-speed ring counter operation	O	O	D8099	High-speed ring counter, range: 0 to 32,767 in units of 0.1 msec	O	O
<b>Other functions</b>							
M8100	SPD (X000) pulse/minute	O	O	D8100	Reserved	O	O
M8101	SPD (X001) pulse/minute	O	O	D8101	Firmware sub-version TP /3VP: 160** The ** and D8001** combines a complete firmware version number	O	O
M8102	SPD (X002) pulse/minute	O	O	D8102	User program capacity	O	O
M8103	SPD (X003) pulse/minute	O	O	D8103	Reserved	O	O

	minute						
M8104	SPD (X004) pulse/ minute	O	O	D8104	The AC/DE time for DRVI, DRVA, [100 ms default value] it effected by M8135 (Y0), <b>it must be the same as D8165.</b>	O	O
M8105	SPD (X005) pulse/ minute	O	O	D8105	The AC/DE time for DRVI, DRVA, [100 ms default value] it effected by M8135 (Y1), <b>it must be the same as D8166.</b>	O	O
M8106	Reserved			D8106	The AC/DE time for DRVI, DRVA, [100 ms default value] it effected by M8135 (Y2), <b>it must be the same as D8167.</b>	O	O
M8107	Reserved			D8107	The AC/DE time for DRVI, DRVA, [100 ms default value] it effected by M8135 (Y3), <b>it must be the same as D8168.</b>	O	O
M8108	Reserved			D8108	Reserved		
M8109	Output refresh error	O	O	D8109	Output refresh error device number;	O	O
<b>COM1 communication settings</b>							
M8110	Reserved			D8110	Com1 port setting (only available in 22319, 24320, 25007 or later)	O	O
M8111	Reserved			D8111	Reserved		
M8112	Reserved			D8112	Reserved		
M8113	Reserved			D8113	Reserved		
M8114	Reserved			D8114	Reserved		
M8115	Reserved			D8115	Reserved		
M8116	Reserved			D8116	Reserved		
M8117	Reserved			D8117	Reserved		
M8118	Reserved			D8118	Reserved		
M8119	Reserved			D8119	Reserved		
<b>COM2 communication settings</b>							

M8120	Reserved			D8120	Com2 port setting, the default value is 0	O	O
M8121	Sending and waiting (RS instruction)	O	O	D8121	Station number settings, the default value is 1	O	O
M8122	Sending flag (RS instruction) Instruction execution status (MODBUS)	O	O	D8122	Amount of remaining data to be transmitted (Only for RS instruction) unit:0.1ms	O	O
M8123	Receiving complete flag (RS) Communication error flag (MODBUS)	O	O	D8123	Amount of data already received (Only to RS instruction)	O	O
M8124	Receiving (only to RS instruction)	O	O	D8124	Start character STX (Only to RS instruction)	O	O
M8125	Reserved			D8125	End character ETX (Only to RS instruction)	O	O
M8126	Reserved			D8126	Communication protocol setting, the default value is 0	O	O
M8127	Reserved			D8127	Starting address for PC protocol	O	O
M8128	Reserved			D8128	Data length for PC protocol	O	O
M8129	Timeout judgement	O	O	D8129	Timeout judgement, default value is 10 (100ms)	O	O
<b>High speed &amp; Position</b>							
M8130	Selects comparison tables to be used with the HSZ instruction	O	O	D8130	Contains the number of the current record being processed in the HSZ comparison table	O	O
M8131		O	O	D8131	HSZ&PLSY speed mode	O	O
M8132	HSZ&PLSY speed mode	O	O	D8132	HSZ&PLAY speed mode	Low	O
M8133		O	O	D8133	frequency	-	
M8134	Reserved			D8134	HSZ&PLAY speed mode	Low	O
M8135	Reserved			D8135	pulses	High	

M8136	Reserved			D8136	total output pulse of Y000&Y001	Low	O	O
M8137	Reserved			D8137		High		
M8138	Reserved			D8138	Reserved			
M8139	Reserved			D8139	Reserved			
M8140	The CLR signal output function of ZRN is valid	O	O	D8140	Accumulated value of PLSY & PLSR output pulse in Y000	Low	O	O
M8141	Accumulator register of output pulse can latched when turn ON (D8136, D8137, D8140~D8143, D8150~D8153)	O	O	D8141		High		
M8142	Reserved			D8142	Accumulated value of PLSY & PLSR output pulse in Y001	Low	O	O
M8143	Reserved			D8143		High		
M8144	Reserved			D8144	Reserved			
M8145	Stop pulse output in Y000	O	O	D8145	Bias speed of DRVI & DRVA			
M8146	Stop pulse output in Y001	O	O	D8146	Highest speed of DRVI & DRVA (default is 100,000)	Low	O	O
M8147	Monitor pulse output in Y000	O	O	D8147		High		
M8148	Monitor pulse output in Y001	O	O	D8148	ACC/DEC time of DRVI & DRVA (default is 100)			
M8149	Monitor pulse output in Y002	O	O	D8149	Reserved			
M8150	Monitor pulse output in Y003	O	O	D8150	Accumulated value of PLSY & PLSR output pulse in Y002	Low	O	O
M8151	Reserved			D8151		High		
M8152	Stop pulse output in Y002	O	O	D8152	Accumulated value of PLSY & PLSR output pulse in Y003	Low	O	O
M8153	Stop pulse output in Y003	O	O	D8153		High		
M8154	Reserved			D8154	Reserved			
M8155	Reserved			D8155	Reserved			
<b>Extend function</b>								

M8156	Reserved			D8156	Define clear signal in Y0 (ZRN) (default is 5=Y5)	O	O
M8157	Reserved			D8157	Define clear signal in Y1 (ZRN) (default is 6=Y6)	O	O
M8158	Reserved			D8158	Define clear signal in Y2 (ZRN) (default is 7=Y7)	O	O
M8159	Reserved			D8159	Define clear signal in Y3 (ZRN) (default is 8=Y10)	O	O
M8160	SWAP function is XCH	-	O	D8160	Define clear signal in Y4 (ZRN) (default is 9=Y11)	O	O
M8161	Bit processing mode of ASC/RS/ASCII/HEX/CC D	O	O	D8161	Reserved		
M8162	High-speed connection in parallel mode	O	O	D8162	Reserved		
M8163	Reserved			D8163	Reserved		
M8164	Variable transmission points mode (FROM/TO)	-	O	D8164	Special transmission points mode (FROM/TO)	O	O
M8165	Reserved			D8165	When enable acceleration and deceleration time, ensure the values is the same as D8104's	O	O
M8166	Reserved			D8166	When enable acceleration and deceleration time, ensure the values is the same as D8105's	O	O
M8167	HEX processing function of SMOV	-	O	D8167	When enable acceleration and deceleration time, ensure the values is the same as D8106's	O	O



M8168	HEX processing function of HEY	-	O	D8168	When enable acceleration and deceleration time, ensure the values is the same as D8107's	O	O
M8169	Reserved			D8169	Reserved		
<b>Pulse catch</b>				<b>Communication</b>			
M8170	X000 pulse catch	O	O	D8170	Reserved		
M8171	X001 pulse catch	O	O	D8171	Reserved		
M8172	X002 pulse catch	O	O	D8172	Reserved		
M8173	X003 pulse catch	O	O	D8173	Station number setting state	O	O
M8174	X004 pulse catch	O	O	D8174	Communication sub-station setting state	O	O
M8175	X005 pulse catch	O	O	D8175	Refresh range setting state	O	O
M8176	Reserved			D8176	Station number setting	O	O
M8177	Reserved			D8177	Communication sub-station setting	O	O
M8178	Reserved			D8178	Refresh range setting	O	O
M8179	Reserved			D8179	Retries setting	O	O
M8180	Reserved			D8180	Timeout setting	O	O
<b>Communication</b>				<b>Indexed addressing</b>			
M8181	Reserved			D8181	Reserved		
M8182	Reserved			D8182	No.2 bit device/ Content of Z1 device	O	O
M8183	Master transfers data error	O	O	D8183	No.3 bit device/ Content of V1 device	O	O
M8184	Slave 1 transfers data error	O	O	D8184	No.4 bit device/ Content of Z2 device	O	O
M8185	Slave 2 transfers data error	O	O	D8185	No.5 bit device/ Content of V2 device	O	O
M8186	Slave 3 transfers data error	O	O	D8186	No.6 bit device/ Content of Z3 device	O	O
M8187	Slave 4 transfers data error	O	O	D8187	No.7 bit device/ Content of V3 device	O	O
M8188	Slave 5 transfers data error	O	O	D8188	No.8 bit device/ Content of Z4 device	O	O

M8189	Slave 6 transfers data error	O	O	D8189	No.9 bit device/ Content of V4 device	O	O
M8190	Slave 7 transfers data error	O	O	D8190	No.10 bit device/ Content of Z5 device	O	O
M8191	Data transferring	O	O	D8191	No.11 bit device/ Content of V5 device	O	O
M8192	Reserved			D8192	No.12 bit device/ Content of Z6 device	O	O
M8193	Reserved			D8193	No.13 bit device/ Content of V6 device	O	O
M8194	Reserved			D8194	No.14 bit device/ Content of Z7 device	O	O
M8195	Reserved			D8195	No.15 bit device/ Content of V7 device	O	O
M8196	Reserved			D8196	Reserved		
M8197	Reserved			D8197	Reserved		
M8198	Reserved			D8198	Reserved		
M8199	Reserved			D8199	Reserved		
Counters states				Communication			
M8200	C200 Control	O	O	D8200	Frequency multiplication of C251 device D8200=0: 1 frequency multiplication D8200=1: 2 frequency multiplication D8200=2: 4 frequency multiplication <b>Note: HSCS, HSCR and HSCZ instructions can be used with frequency multiplication simultaneously. And this function is available in V311 or later version</b>	O	O
M8201	C201 Control	O	O	D8201	Reserved		
M8202	C202 Control	O	O	D8202	Reserved		
M8203	C203 Control	O	O	D8203	Reserved		
M8204	C204 Control	O	O	D8204	Reserved		

M8205	C205 Control	O	O	D8205	Reserved		
M8206	C206 Control	O	O	D8206	Reserved		
M8207	C207 Control	O	O	D8207	Reserved		
M8208	C208 Control	O	O	D8208	Reserved		
M8209	C209 Control	O	O	D8209	Reserved		
M8210	C210 Control	O	O	D8210	Reserved		
M8211	C211 Control	O	O	D8211	Reserved		
M8212	C212 Control	O	O	D8212	Reserved		
M8213	C213 Control	O	O	D8213	Reserved		
M8214	C214 Control	O	O	D8214	Reserved		
M8215	C215 Control	O	O	D8215	Reserved		
M8216	C216 Control	O	O	D8216	Reserved		
M8217	C217 Control	O	O	D8217	Reserved		
M8218	C218 Control	O	O	D8218	Reserved		
M8219	C219 Control	O	O	D8219	Reserved		
M8220	C220 Control	O	O	D8220	<p>D8220=1 to enable the new filtering methods (four points constitute a set of filter). When use new filtering methods, the filter time which set by D8020 is not valid. And before using this filtering methods, users need to set the filtering time for each X terminals (D8221~D8228), Filter time unit is ms.</p> <p><b>Note: This filter method only works on CPU IO, the IO in extension module is not invalid.</b></p>	O	O
M8221	C221 Control	O	O	D8221	<p>Low bits are for setting filter time of X0~X3; High bits are for setting filter time of X4~X7 Unit is ms</p>	O	O

M8222	C222 Control	O	O	D8222	Low bits are for setting filter time of X10~X13; High bits are for setting filter time of X14~X17 Unit is ms	O	O
M8223	C223 Control	O	O	D8223	Low bits are for setting filter time of X20~X23; High bits are for setting filter time of X24~X27 Unit is ms	O	O
M8224	C224 Control	O	O	D8224	Low bits are for setting filter time of X30~X33; High bits are for setting filter time of X34~X37 Unit is ms	O	O
M8225	C225 Control	O	O	D8225	Low bits are for setting filter time of X40~X43; High bits are for setting filter time of X44~X47 Unit is ms	O	O
M8226	C226 Control	O	O	D8226	Low bits are for setting filter time of X50~X53; High bits are for setting filter time of X54~X57 Unit is ms	O	O
M8227	C227 Control	O	O	D8227	Low bits are for setting filter time of X60~X63; High bits are for setting filter time of X64~X67 Unit is ms	O	O
M8228	C228 Control	O	O	D8228	Low bits are for setting filter time of X70~X73;	O	O

					High bits are for setting filter time of X74~X77 Unit is ms		
M8229	C229 Control	O	O	D8229	Reserved		
M8230	C230 Control	O	O	D8230	Reserved		
M8231	C231 Control	O	O	D8231	Reserved		
M8232	C232 Control	O	O	D8232	Reserved		
M8233	C233 Control	O	O	D8233	Reserved		
M8234	C234 Control	O	O	D8234	Reserved		
M8235	One phase one directional	C235 Control	O	O	D8235	Reserved	
M8236		C236 Control	O	O	D8236	Reserved	
M8237		C237 Control	O	O	D8237	Reserved	
M8238		C238 Control	O	O	D8238	Reserved	
M8239		C239 Control	O	O	D8239	Reserved	
M8240		C240 Control	O	O	D8240	Reserved	
M8241		C241 Control	O	O	D8241	Reserved	
M8242		C242 Control	O	O	D8242	Reserved	
M8243		C243 Control	O	O	D8243	Reserved	
M8244		C244 Control	O	O	D8244	Reserved	
M8245	C245 Control	O	O	D8245	Reserved		
M8246	2 phase bi-directional	C246 Control	O	O	D8246	Reserved	
M8247		C247 Control	O	O	D8247	Reserved	
M8248		C248 Control	O	O	D8248	Reserved	
M8249		C249 Control	O	O	D8249	Reserved	
M8250		C250 Control	O	O	D8250	Reserved	
M8251	A/B phase	C251 Control	O	O	D8251	Reserved	
M8252		C252 Control	O	O	D8252	Reserved	
M8253		C253 Control	O	O	D8253	Reserved	
M8254		C254 Control	O	O	D8254	Reserved	
M8255		C255 Control	O	O	D8255	Reserved	



## 8 Error code

Error code	Error contents
0000	No error
0***~1***	Configuration error
6101~6115	Hardware error
6201~6205	PP communication error
6301~6340	Serial port communication error
6402~6421	Parameters error
6501~6512	Grammatical errors
6601~6632	Ladder program error
6701~6780	Calculation error

### 8.1 No error

#### 8.1.1 Error code 0000

- Applicable Model: TP series CPU
- Error code (D8067):0000
- Error message: No error
- LED state (RUN): -
- LED state (PROG.E): -
- PLC acting state: -
- Diagnosis timing: -
- Abnormal content and reasons None
- Solution:

The error code “0000” means no error. No need to do any operation for it.

### 8.2 Configuration Error (0\*\*\*~1\*\*\*)

#### 8.2.1 Error code 0\*\*\*~1\*\*\*

- Applicable Model: TP series CPU
- Error code (D8060):0\*\*\*, 1\*\*\*
- Error message: I/O configuration error
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing:  
PLC is running;

When switch power OFF -> ON

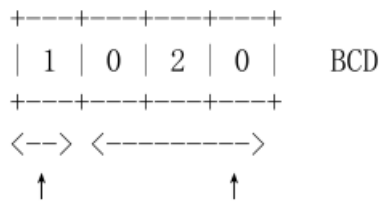
First switch STOP-> RUN after powering ON

- Abnormal content and reasons

Display the starting address of unconnected I/O

For example:

Suppose X20 is unconnected



1: input Device address 10~337

0: output

1<sup>st</sup> ~ 3<sup>rd</sup> codes mean device address;

4<sup>th</sup> code means I/O type (1=input, 0=output)

- Solution:

Change the device address;

Sometime PLC can run with error device address, but it is better to change it

## 8.3 Hardware Error (6101~6115)

### 8.3.1 Error code 6101

- Applicable Model: TP series CPU
- Error code (D8061):6101
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing: PLC power is turned ON
- Abnormal content and reasons RAM error

### 8.3.2 Error code 6102

- Applicable Model: TP series CPU
- Error code (D8061):6102
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing: PLC power is turned ON



- Abnormal content and reasons Operation ladder error

### 8.3.3 Error code 6103

- Applicable Model: TP series CPU
- Error code (D8061):6103
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing: PLC power is turned ON
- Abnormal content and reasons I/O bus error (M8069 is ON)
- Solution  
Check extension module cables.

### 8.3.4 Error code 6104

- Applicable Model: TP series CPU
- Error code (D8061):6104
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing: PLC power is turned ON
- Abnormal content and reasons Extension module 24V failure (M8069=ON)
- Solution  
Check extension cables.

### 8.3.5 Error code 6105

- Applicable Model: TP series CPU
- Error code (D8061):6105
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing: PLC power is turned ON
- Abnormal content and reasons Watch Dog Timer error.
- Solution  
Confirm user program. The scan time exceeds the value stored in D8000

### 8.3.6 Error code 6106

- Applicable Model: TP series CPU
- Error code (D8061):6106
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing: PLC power is turned ON
- Abnormal content and reasons Create I/O table error (CPU error)
- Solution

The error occurs if the 24V power is not supplied for 10 seconds or more after main power turns ON.

### 8.3.7 Error code 6107

- Applicable Model: TP series CPU
- Error code (D8061):6107
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing: PLC power is turned ON
- Abnormal content and reasons System configure error,
- Solution The numbers of connected expansion module exceed the allowed range.

### 8.3.8 Error code 6113

- Applicable Model: TP series CPU
- Error code (D8061):6113
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing  
PLC power is turned ON. Change the setting for CC-Link/LT
- Abnormal content and reasons  
The setting for CC-Link/LT can't be saved, when protect switch at ON state and install the storage tape
- Solution  
Protect switch turns OFF, and set again.

### 8.3.9 Error code 6114

- Applicable Model: TP series CPU

- Error code (D8061):6114
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing  
PLC power is turned ON. Change the setting for CC-Link/LT
- Abnormal content and reasons The setting for CC-Link/LT can't be saved on master.

### 8.3.10 Error code 6115

- Applicable Model: TP series CPU
- Error code (D8061):6115
- Error message: PLC hardware error
- LED state (RUN): OFF
- LED state (PROG.E): ON
- PLC acting state: Stop
- Diagnosis timing
  - 1) PLC power is turned ON.
  - 2) Change the setting for CC-Link/LT
- Abnormal content and reasons
  - 1) Write error in CC-Link/LT master EEPROM
  - 2) Unable to complete the configuration in the automatic mode

## 8.4 PP communication error (6201~6205)

### 8.4.1 Error code 6201

- Applicable Model: TP series CPU
- Error code (D8062):6201
- Error message: PLC/PP communication error
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving PP signal
- Abnormal content and reasons Parity/ overrun/ framing error
- Solution
  - 1) Check the cable connection between the programming panel (PP) and the PLC.
  - 2) Plug communication cable when monitoring will lead error.
  - 3) In TPN/TP Series, the basic module power transmission speed is 9.6 Kbps, but the transmission speed for peripherals is 19.2Kbps, so when error occurs when executing the programming

communication. So please clear the error flag in every channel, and set the transmission speed for peripherals to be 9.6 Kbps, when a communication error occurs at the start of the communication

#### 8.4.2 Error code 6202

- Applicable Model: TP series CPU
- Error code (D8062):6202
- Error message: PLC hardware error
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving PP signal
- Abnormal content and reasons Communications character error
- Solution
  - 1) Check the cable connection between the programming panel (PP) and the PLC.
  - 2) Plug communication cable when monitoring will lead error.

#### 8.4.3 Error code 6203

- Applicable Model: TP series CPU
- Error code (D8062):6203
- Error message: PLC hardware error
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving PP signal
- Abnormal content and reasons Communication data sum check error
- Solution
  - 1) Check the cable connection between the programming panel (PP) and the PLC.
  - 2) Plug communication cable when monitoring will lead error.

#### 8.4.4 Error code 6204

- Applicable Model: TP series CPU
- Error code (D8062):6204
- Error message: PLC hardware error
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving PP signal
- Abnormal content and reasons Data format error
- Solution

- 1) Check the cable connection between the programming panel (PP) and the PLC.
- 2) Plug communication cable when monitoring will lead error.

### 8.4.5 Error code 6205

- Applicable Model: TP series CPU
- Error code (D8062):6205
- Error message: PLC hardware error
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving PP signal
- Abnormal content and reasons Instruction error
- Solution
  - 1) Check the cable connection between the programming panel (PP) and the PLC.
  - 2) Plug communication cable when monitoring will lead error.

## 8.5 Serial communication error (6301~6340)

### 8.5.1 Error code 6301

- Applicable Model: TP series CPU
- Error code (D8063):6301
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Parity/ overrun/ framing error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.
  - 5) In TPN/TP Series, the basic module power transmission speed is 9.6 Kbps, but the transmission speed for peripherals is 19.2Kbps, so when error occurs when executing the programming communication. So please clear the error flag in every channel, and set the transmission speed for peripherals to be 9.6 Kbps, when a communication error occurs at the start of the communication

### 8.5.2 Error code 6302

- Applicable Model: TP series CPU

- Error code (D8063):6302
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Communication character error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.3 Error code 6303

- Applicable Model: TP series CPU
- Error code (D8063):6303
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Communication data sum check error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.4 Error code 6304

- Applicable Model: TP series CPU
- Error code (D8063):6304
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Communication data format error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.

- 2) Simple PLC connections, parallel connection: check the programming setting.
- 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
- 4) Check the communication cable.

### 8.5.5 Error code 6305

- Applicable Model: TP series CPU
- Error code (D8063):6305
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Instructions error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.6 Error code 6306

- Applicable Model: TP series CPU
- Error code (D8063):6306
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Watchdog timer error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.7 Error code 6307

- Applicable Model: TP series CPU
- Error code (D8063):6307
- Error message: Serial communication error 1
- LED state (RUN): ON

- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Modem initialization error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.8 Error code 6308

- Applicable Model: TP series CPU
- Error code (D8063):6308
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Parameter between simple PLC connections error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.9 Error code 6309

- Applicable Model: TP series CPU
- Error code (D8063):6309
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Program setting between simple PLC connections error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.



### 8.5.10 Error code 6312

- Applicable Model: TP series CPU
- Error code (D8063):6312
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Parallel link character error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.11 Error code 6313

- Applicable Model: TP series CPU
- Error code (D8063):6313
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Parallel link data sum check error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.12 Error code 6314

- Applicable Model: TP series CPU
- Error code (D8063):6314
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station

- Abnormal content and reasons Parallel link data format error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.13 Error code 6320

- Applicable Model: TP series CPU
- Error code (D8063):6320
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Communication with inverter error
- Solution
  - 1) Communicate with the inverter/PC, programming: check the communication parameter setting.
  - 2) Simple PLC connections, parallel connection: check the programming setting.
  - 3) Remote maintenance: ensure the modem's power is turned on, and AT instruction content is right.
  - 4) Check the communication cable.

### 8.5.14 Error code 6340

- Applicable Model: TP series CPU
- Error code (D8063):6340
- Error message: Serial communication error 1
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When receiving signal for target station
- Abnormal content and reasons Connection is abnormal in special adapter.
- Solution
 

Check the connection state of the special adapter.

## 8.6 Parameter error (6401~6421)

### 8.6.1 Error code 6401

- Applicable Model: TP series CPU
- Error code (D8064):6401
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Program sum check error
- Solution  
Turn off PLC, and set the parameters correctly.

### 8.6.2 Error code 6402

- Applicable Model: TP series CPU
- Error code (D8064):6402
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Memory capacity setting error
- Solution  
Turn off PLC, and set the parameters correctly.

### 8.6.3 Error code 6403

- Applicable Model: TP series CPU
- Error code (D8064):6403
- Error message: Parameter error

- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Latched device area setting error
- Solution
 

Turn off PLC, and set the parameters correctly.

#### 8.6.4 Error code 6404

- Applicable Model: TP series CPU
- Error code (D8064):6404
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Comment area setting error
- Solution
 

Turn off PLC, and set the parameters correctly.

#### 8.6.5 Error code 6405

- Applicable Model: TP series CPU
- Error code (D8064):6405
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON

4) The first RUN state after PLC turn ON

- Abnormal content and reasons File register area setting error
- Solution  
Turn off PLC, and set the parameters correctly.

### 8.6.6 Error code 6406

- Applicable Model: TP series CPU
- Error code (D8064):6406
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons BFM initial value data sum check error
- Solution  
Turn off PLC, and set the parameters correctly.

### 8.6.7 Error code 6407

- Applicable Model: TP series CPU
- Error code (D8064):6407
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons BFM initial value data error
- Solution  
Turn off PLC, and set the parameters correctly.

### 8.6.8 Error code 6407

- Applicable Model: TP series CPU

- Error code (D8064):6407
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons BFM initial value data error
- Solution
 

Turn off PLC, and set the parameters correctly.

### 8.6.9 Error code 6411

- Applicable Model: TP series CPU
- Error code (D8064):6411
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Built-in CC-Link / LT (specific area) parameter error
- Solution
 

Turn off PLC, and set the parameters correctly.

### 8.6.10 Error code 6412

- Applicable Model: TP series CPU
- Error code (D8064):6412
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)

- 2) Project is transferred (STOP state)
- 3) PLC power is turned ON
- 4) The first RUN state after PLC turn ON
- Abnormal content and reasons  
Built-in CC-Link / LT (special settings area) parameter sum check error
- Solution  
Turn off PLC, and set the parameters correctly.

### 8.6.11 Error code 6413

- Applicable Model: TP series CPU
- Error code (D8064):6413
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons  
Built-in CC-Link / LT (specific area) parameter sum check error
- Solution  
Turn off PLC, and set the parameters correctly.

### 8.6.12 Error code 6420

- Applicable Model: TP series CPU
- Error code (D8064):6420
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Special parameter sum check error
- Solution

Turn off PLC, and set the parameters correctly.

### 8.6.13 Error code 6421

- Applicable Model: TP series CPU
- Error code (D8064):6421
- Error message: Parameter error
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Special parameter setting error
- Solution
  - 1) Check the special parameter error, and the content of D8489
  - 2) Troubleshooting in special adapter
  - 3) Set the special parameters correctly

## 8.7 Syntax error (6501~6512)

### 8.7.1 Error code 6501

- Applicable Model: TP series CPU
- Error code (D8065):6501
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Instruction/ device symbol/device number error
- Solution

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.



### 8.7.2 Error code 6502

- Applicable Model: TP series CPU
- Error code (D8065):6502
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons No timer or counter coil before setting value
- Solution

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.3 Error code 6503

- Applicable Model: TP series CPU
- Error code (D8065):6503
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
  - 1) No setting value following either a timer or a counter coil
  - 2) Insufficient number of operands for an applied instruction
- Solution

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.4 Error code 6504

- Applicable Model: TP series CPU
- Error code (D8065):6504
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
  - 1) The same label number is used more than once
  - 2) The same interrupt input or high speed counter input is used more than once
- Solution
 

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.5 Error code 6505

- Applicable Model: TP series CPU
- Error code (D8065):6505
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
 

Device number is outside the allowable range
- Solution
 

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.6 Error code 6506

- Applicable Model: TP series CPU
- Error code (D8065):6506
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Invalid applied instruction
- Solution

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.7 Error code 6507

- Applicable Model: TP series CPU
- Error code (D8065):6507
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Invalid P assignment
- Solution

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.8 Error code 6508

- Applicable Model: TP series CPU
- Error code (D8065):6508

- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Invalid I assignment
- Solution

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.9 Error code 6509

- Applicable Model: TP series CPU
- Error code (D8065):6509
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Other error
- Solution

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.10 Error code 6510

- Applicable Model: TP series CPU
- Error code (D8065):6510
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF

- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons MC nesting (N) number error
- Solution
 

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.11 Error code 6511

- Applicable Model: TP series CPU
- Error code (D8065):6511
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Interrupt and high speed counter assigned inputs overlap
- Solution
 

During programming, each instruction is checked as it is entered. If a syntax error is detected, re-enter the instruction correctly.

### 8.7.12 Error code 6512

- Applicable Model: TP series CPU
- Error code (D8065):6512
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing

- 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons EXTR instruction is used without extend storage
  - Solution Please use TPN/ TP series PLC

## 8.8 Ladder error (6601~6632)

### 8.8.1 Error code 6601

- Applicable Model: TP series CPU
- Error code (D8066):6601
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons LD and LDI is used continuously 9 or more times in succession
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.2 Error code 6602

- Applicable Model: TP series CPU
- Error code (D8066):6602
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON

- 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
  - 1) No LD/ LDI instruction. Unauthorized use of the LD / LDI, AND / ANI instructions
  - 2) The following instructions are not connected to the active bus line: STL, RET, MCR, (P)pointer, (I)interrupt, EI, DI, SRET, IRET, FOR, NEXT, FEND and END
  - 3) When MPP is missing
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.3 Error code 6603

- Applicable Model: TP series CPU
- Error code (D8066):6603
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons MPS is used continuously more than 12 times
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.4 Error code 6604

- Applicable Model: TP series CPU
- Error code (D8066):6604
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)

- 3) PLC power is turned ON
- 4) The first RUN state after PLC turn ON

- Abnormal content and reasons Unauthorized use of the MPS/ MRD/MPP instructions
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.5 Error code 6605

- Applicable Model: TP series CPU
- Error code (D8066):6605
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
  - 1) A single STL branch drives 9 or more parallel circuits
  - 2) MC/ MCR or (I) interrupts are designated within an STL state
  - 3) RET has not been designated or is designated out of an STL state

- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.6 Error code 6606

- Applicable Model: TP series CPU
- Error code (D8066):6606
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)



- 3) PLC power is turned ON
- 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
  - 1) No (P) pointer/ (I) interrupt
  - 2) No SRET/ IRET
  - 3) An (I) interrupt/ SRET or IRET has been designated within the main body of the program
  - 4) STL/ RET/ MC or MCR have been designated within either a subroutine or an interrupt routine
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.7 Error code 6607

- Applicable Model: TP series CPU
- Error code (D8066):6607
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
  - 1) Unauthorized use of FOR - NEXT. 6 or more levels have been designated
  - 2) The following instructions have been designated within a FOR -NEXT loop: STL/ RET/ MC/ MCR/ IRET/ SRET/ FEND or END
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.8 Error code 6608

- Applicable Model: TP series CPU
- Error code (D8066):6608
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker

- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
  - 1) Unauthorized MC/ MCR relationship
  - 2) Missing MCR N0
  - 3) SRET/ IRET or an (I) interrupt has been designated within an MC/ MCR block
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.9 Error code 6609

- Applicable Model: TP series CPU
- Error code (D8066):6609
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Other error
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.10 Error code 6610

- Applicable Model: TP series CPU
- Error code (D8066):6610
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker

- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons LD and LDI is used continuously 9 or more times in succession
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.11 Error code 6611

- Applicable Model: TP series CPU
- Error code (D8066):6611
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Number of LD/LDI instructions is fewer than ANB/ORB instructions
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.12 Error code 6612

- Applicable Model: TP series CPU
- Error code (D8066):6612
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)

- 2) Project is transferred (STOP state)
- 3) PLC power is turned ON
- 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Number of LD/LDI instructions is more than ANB/ORB instructions
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.13 Error code 6613

- Applicable Model: TP series CPU
- Error code (D8066):6613
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons MPS is used continuously more than 12 times
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.14 Error code 6614

- Applicable Model: TP series CPU
- Error code (D8066):6614
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON

- Abnormal content and reasons MPS instruction missing
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.15 Error code 6615

- Applicable Model: TP series CPU
- Error code (D8066):6615
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons MPP instruction missing
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.16 Error code 6616

- Applicable Model: TP series CPU
- Error code (D8066):6616
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Unauthorized use of the MPS/ MRD/MPP instructions; possible coil missing
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.17 Error code 6617

- Applicable Model: TP series CPU
- Error code (D8066):6617
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons One of the following instructions is not connected to the active bus line: STL, RET, MCR, (P)pointer, (I)interrupt, EI, DI, SRET, IRET, FOR, NEXT, FEND and END
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.18 Error code 6618

- Applicable Model: TP series CPU
- Error code (D8066):6618
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons STL/ RET/ MC or MCR programmed within either a subroutine or an interrupt routine
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select

programming mode and correct the identified error.

### 8.8.19 Error code 6619

- Applicable Model: TP series CPU
- Error code (D8066):6619
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Invalid instruction programmed within a FOR - NEXT loop: STL/ RET/ MC/ MCR/ I/ IRET/ SRET
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.20 Error code 6620

- Applicable Model: TP series CPU
- Error code (D8066):6620
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reason FOR - NEXT nesting exceeded
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.21 Error code 6621

- Applicable Model: TP series CPU
- Error code (D8066):6621
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons Unmatched number of FOR and NEXT instructions
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.22 Error code 6622

- Applicable Model: TP series CPU
- Error code (D8066):6622
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons NEXT instruction not found
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.23 Error code 6623

- Applicable Model: TP series CPU
- Error code (D8066):6623



- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons MC instruction not found
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

#### 8.8.24 Error code 6624

- Applicable Model: TP series CPU
- Error code (D8066):6624
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons MCR instruction not found
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

#### 8.8.25 Error code 6625

- Applicable Model: TP series CPU
- Error code (D8066):6625
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF

- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
 

A single STL branch drives 9 or more parallel circuits
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.26 Error code 6626

- Applicable Model: TP series CPU
- Error code (D8066):6626
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons
 

Invalid instruction programmed within an STL - RET block: MC/ MCR/ I/ IRET/ SRET
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.27 Error code 6627

- Applicable Model: TP series CPU
- Error code (D8066):6627
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker

- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons RET instruction not found
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.28 Error code 6628

- Applicable Model: TP series CPU
- Error code (D8066):6628
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons I/ SRET/ IRET incorrectly programmed within main program
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.29 Error code 6629

- Applicable Model: TP series CPU
- Error code (D8066):6629
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)

- 2) Project is transferred (STOP state)
- 3) PLC power is turned ON
- 4) The first RUN state after PLC turn ON
- Abnormal content and reasons P or I label not found
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.30 Error code 6630

- Applicable Model: TP series CPU
- Error code (D8066):6630
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons SRET or IRET not found
- Solution
 

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.31 Error code 6631

- Applicable Model: TP series CPU
- Error code (D8066):6631
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON

- Abnormal content and reasons SRET programmed in invalid location
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.32 Error code 6632

- Applicable Model: TP series CPU
- Error code (D8066):6632
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons FEND programmed in invalid location
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

### 8.8.33 Error code 6633

- Applicable Model: TP series CPU
- Error code (D8066):6633
- Error message: Syntax error
- Display error location device: D8069
- LED state (RUN): OFF
- LED state (PROG.E): Flicker
- PLC acting state: Stop
- Diagnosis timing
  - 1) Project is changed (STOP state)
  - 2) Project is transferred (STOP state)
  - 3) PLC power is turned ON
  - 4) The first RUN state after PLC turn ON
- Abnormal content and reasons  
STL instruction not found
- Solution

A ladder error occurs if a combination of instructions is incorrect or badly specified. Select programming mode and correct the identified error.

## 8.9 Operation error (6701~6780)

### 8.9.1 Error code 6701

- Applicable Model: TP series CPU
- Error code (D8067):6701
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons
  - 1) No jump destination for CJ or CALL instructions
  - 2) Undefined index result and label, or exceed the range of P0 ~ P4095
  - 3) P63 is executed by CALL instruction, the P63 is used with END instruction, and it can't be used with CALL instruction.

- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.2 Error code 6702

- Applicable Model: TP series CPU
- Error code (D8067):6702
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons 6 or more CALL instructions have been nested together
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.3 Error code 6703

- Applicable Model: TP series CPU
- Error code (D8067):6703
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons 3 or more interrupts instructions have been nested together
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.4 Error code 6704

- Applicable Model: TP series CPU
- Error code (D8067):6704
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons 6 or more FOR~NEXT instructions have been nested together
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.5 Error code 6705

- Applicable Model: TP series CPU
- Error code (D8067):6705
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons The operand of the application instruction exceeds the range
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.6 Error code 6706

- Applicable Model: TP series CPU
- Error code (D8067):6706
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons A device has been specified outside of the allowable range for an applied instruction operand
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.7 Error code 6707

- Applicable Model: TP series CPU
- Error code (D8067):6707
- Error message: Operation error



- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons A file register has been accessed which is outside of the users specified range
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.8 Error code 6708

- Applicable Model: TP series CPU
- Error code (D8067):6708
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons FROM/TO instruction error
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.9 Error code 6709

- Applicable Model: TP series CPU
- Error code (D8067):6709
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run

- Diagnosis timing When PLC is running
- Abnormal content and reasons Other error, such as unauthorized FOR-NEXT relationship
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.10 Error code 6710

- Applicable Model: TP series CPU
- Error code (D8067):6710
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Parameter mismatch
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.11 Error code 6712

- Applicable Model: TP series CPU
- Error code (D8067):6712
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons 32 or more times high-speed compare instruction have been used
- Solution

These errors occur during the execution of an operation.

When an operation error occurs, STOP the PLC, and enter programming mode and correct the error.

Note: Operation errors can occur even when the syntax or ladder is correct, e.g. T200Z is a valid statement with TETA TP PLC. But if Z had a value of 100, the data register T300 would be attempted to be accessed. This will cause an operation error as there is no T300 device available.

### 8.9.12 Error code 6730

- Applicable Model: TP series CPU
- Error code (D8067):6730
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reason Sampling time Ts ( $T_s < 0$ )
- Solution

The identified parameter is specified outside of its allowable range

Execution ceases PID instruction must be reset before execution will be resumed

### 8.9.13 Error code 6732

- Applicable Model: TP series CPU
- Error code (D8067):6732
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reason Input filter value  $\alpha$  ( $\alpha < 0$  or  $\geq 100$ )
- Solution

The identified parameter is specified outside of its allowable range Execution ceases PID instruction must be reset before execution will be resumed

### 8.9.14 Error code 6733

- Applicable Model: TP series CPU
- Error code (D8067):6733
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF

- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Proportional gain  $K_P$  ( $K_P < 0$ )
- Solution

The identified parameter is specified outside of its allowable range Execution ceases PID instruction must be reset before execution will be resumed

### 8.9.15 Error code 6734

- Applicable Model: TP series CPU
- Error code (D8067):6734
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Integral time constant  $T_I$  ( $T_I < 0$ )
- Solution

The identified parameter is specified outside of its allowable range Execution ceases PID instruction must be reset before execution will be resumed

### 8.9.16 Error code 6735

- Applicable Model: TP series CPU
- Error code (D8067):6735
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Derivative gain  $K_D$  ( $K_D < 0$  or  $\geq 201$ )
- Solution

The identified parameter is specified outside of its allowable range Execution ceases PID instruction must be reset before execution will be resumed

### 8.9.17 Error code 6736

- Applicable Model: TP series CPU
- Error code (D8067):6736
- Error message: Operation error

- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Derivative time constant  $T_D$  ( $T_D < 0$ )
- Solution
 

The identified parameter is specified outside of its allowable range Execution ceases PID instruction must be reset before execution will be resumed

### 8.9.18 Error code 6740

- Applicable Model: TP series CPU
- Error code (D8067):6740
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Sampling time  $T_s$  is less than the program scan time
- Solution
 

$T_s$  is set to program scan time, the execution will continue.

### 8.9.19 Error code 6742

- Applicable Model: TP series CPU
- Error code (D8067):6742
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Current value  $\Delta$  exceeds its limits ( $\Delta PV < -32768$  or  $32767 < \Delta PV$ )
- Solution
 

Data affected resets to the nearest limit value. This will either be a minimum of -32768 or a maximum of +32767. Execution will continue, but user should reset PID instruction.

### 8.9.20 Error code 6743

- Applicable Model: TP series CPU

- Error code (D8067):6743
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Calculated error  $\varepsilon$  exceeds its limits ( $\varepsilon V < -32768$  or  $32767 < \varepsilon V$ )
- Solution  
Data affected resets to the nearest limit value. This will either be a minimum of -32768 or a maximum of +32767. Execution will continue, but user should reset PID instruction.

### 8.9.21 Error code 6744

- Applicable Model: TP series CPU
- Error code (D8067):6744
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Integral result exceeds its limits
- Solution  
Data affected resets to the nearest limit value. This will either be a minimum of -32768 or a maximum of +32767. Execution will continue, but user should reset PID instruction.

### 8.9.22 Error code 6745

- Applicable Model: TP series CPU
- Error code (D8067):6745
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Derivative gain over, or differential value exceeds allowable range
- Solution  
Data affected resets to the nearest limit value. This will either be a minimum of -32768 or a maximum of +32767. Execution will continue, but user should reset PID instruction.

### 8.9.23 Error code 6746

- Applicable Model: TP series CPU
- Error code (D8067):6746
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Derivative result exceeds its limits
- Solution

Data affected resets to the nearest limit value. This will either be a minimum of -32768 or a maximum of +32767. Execution will continue, but user should reset PID instruction.

### 8.9.24 Error code 6747

- Applicable Model: TP series CPU
- Error code (D8067):6747
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Total PID result exceeds its limits
- Solution

Data affected resets to the nearest limit value. This will either be a minimum of -32768 or a maximum of +32767. Execution will continue, but user should reset PID instruction.

### 8.9.25 Error code 6748

- Applicable Model: TP series CPU
- Error code (D8067):6748
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons PID output upper limit set value < PID output lower limit value

- Solution

Exchange upper limit value and the lower limit value of the PID output, and then continue the operation.

### 8.9.26 Error code 6749

- Applicable Model: TP series CPU
- Error code (D8067):6749
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Alarm set value of PID error (Alarm set value<0)
- Solution

No alarm output→continues PID operation. Please ensure all the settings are correctly

### 8.9.27 Error code 6750

- Applicable Model: TP series CPU
- Error code (D8067):6750
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Self-tuning results error
- Solution

[Self-tuning end → execute PID calculation]

Confirm measured value and target value, and execute PID again

- 1) End the self-tuning, when the difference between the measured value and the target value at the start is less than 150.
- 2) End the self-tuning, when the difference between the measured value and the target value at the start is over 1/3.

### 8.9.28 Error code 6751

- Applicable Model: TP series CPU
- Error code (D8067):6751
- Error message: Operation error
- Display error location device: D8069



- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Self-tuning direction inconsistent
- Solution

Please set target value correctly, please confirm the corresponding between output value of self-tuning, and measured value, and execute PID again.

### 8.9.29 Error code 6752

- Applicable Model: TP series CPU
- Error code (D8067):6752
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Self-tuning error
- Solution

The error fluctuation is outside the normal operation limits for the PID instruction.

Please set sampling time longer than output variation period, or increasing the input filter constant.

Execution ceases. PID instruction must be reset.

### 8.9.30 Error code 6753

- Applicable Model: TP series CPU
- Error code (D8067):6753
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons << Limit cycle method >> Output set value for self-tuning is abnormal [ULV (upper limit)  $\leq$  LLV (lower limit)]
- Solution

<< Forced end self-tuning → Doesn't perform PID operation transfer >> Make sure to set the contents of the object is correct.

### 8.9.31 Error code 6754

- Applicable Model: TP series CPU
- Error code (D8067):6754
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons << Limit cycle method >> The setting value of PV limit in self-tuning is abnormal (SHpv<0).
- Solution

<< Forced end self-tuning→ Doesn't perform PID operation transfer >> Make sure to set the contents of the object is correct.

### 8.9.32 Error code 6755

- Applicable Model: TP series CPU
- Error code (D8067):6755
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons << Limit cycle method >> The migration status in self-tuning is abnormal
- Solution

<< Forced end self-tuning→ Doesn't perform PID operation transfer >> Please check the devices for PID instruction in PLC program

### 8.9.33 Error code 6756

- Applicable Model: TP series CPU
- Error code (D8067):6756
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running

- Abnormal content and reasons << Limit cycle method >> Self-tuning measurement timeout leads to abnormal results ( $\tau_{on} > \tau$ ,  $\tau_{on} < 0$ ,  $\tau < 0$ )
- Solution
  - << Forced end self-tuning → doesn't perform PID operation transfer >>
  - Please set enough time for self-tuning.
  - The difference between lower and high limit for self-tuning output (ULV-LLV), input filter constant  $\alpha$ , and reduce PV limit value and SHpv value for solving problem.

### 8.9.34 Error code 6757

- Applicable Model: TP series CPU
- Error code (D8067):6757
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons << Limit cycle method >> Beyond proportional gain range for self-tuning ( $K_p=0\sim32767$ )
- Solution
  - << End self-tuning → doesn't perform PID operation transfer >>
  - The change of measured value (PV) is less than output value's. Please use 10 times for inputting measured value (PV) operation to increase PV change.

### 8.9.35 Error code 6758

- Applicable Model: TP series CPU
- Error code (D8067):6758
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons << Limit cycle method >>
  - Beyond integration time range for self-tuning ( $T_I=0\sim32767$ )
- Solution
  - << End self-tuning → doesn't perform PID operation transfer >> Set the enough time for self-tuning.
  - The difference between lower and high limit for self-tuning output (ULV-LLV), input filter constant  $\alpha$ , and reduce PV limit value and SHpv value for solving problem.

### 8.9.36 Error code 6759

- Applicable Model: TP series CPU
- Error code (D8067):6759
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons
  - << Limit cycle method >> Beyond integration time range for self-tuning (TI=0~32767)
- Solution << End self-tuning→ doesn't perform PID operation transfer >>
  - Set the enough time for self-tuning.
  - The difference between lower and high limit for self-tuning output (ULV-LLV), input filter constant  $\alpha$ , and reduce PV limit value and SHpv value for solving problem.

### 8.9.37 Error code 6760

- Applicable Model: TP series CPU
- Error code (D8067):6760
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Sum check of servo ABS error
- Solution
  - Please check the settings and the connections with servos

### 8.9.38 Error code 6762

- Applicable Model: TP series CPU
- Error code (D8067):6762
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running

- Abnormal content and reasons The port which specified by communication command is occupied by other devices
- Solution Please check the communication port.

### 8.9.39 Error code 6763

- Applicable Model: TP series CPU
- Error code (D8067):6763
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons The X terminal which specified by DVIT, ZRN, and DSZR instructions is occupied by others
- Solution  
Please make sure the X terminal which specified by DVIT, ZRN, and DSZR instructions is not used for following functions
  - Input Interrupt (contains delay function)
  - High-speed counter C235 ~ C255
  - Pulse catch M8050 ~ M8057
  - SPD instruction

### 8.9.40 Error code 6764

- Applicable Model: TP series CPU
- Error code (D8067):6764
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Pulse output number has been used by positioning instruction, pulse output instruction (PLSY, PWM etc.)
- Solution  
Please make sure the Y terminal which specified by pulse output instructions is not used by others.

### 8.9.41 Error code 6765

- Applicable Model: TP series CPU

- Error code (D8067):6765
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Using frequency of application instructions error
- Solution  
Application instruction is used continuously more than allowed times

#### 8.9.42 Error code 6770

- Applicable Model: TP series CPU
- Error code (D8067):6770
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons Flash writing error

#### 8.9.43 Error code 6771

- Applicable Model: TP series CPU
- Error code (D8067):6771
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons FLASH storage tape is not connected
- Solution  
Please confirm whether the storage is correctly installed

#### 8.9.44 Error code 6772

- Applicable Model: TP series CPU
- Error code (D8067):6772
- Error message: Operation error

- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons FLASH storage is in writing disabled state
- Solution  
Please change the writing disabled state, and try again

#### 8.9.45 Error code 6772

- Applicable Model: TP series CPU
- Error code (D8067):6772
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons FLASH storage is in writing disabled state
- Solution  
Please change the writing disabled state, and try again

#### 8.9.46 Error code 6774

- Applicable Model: TP series CPU
- Error code (D8067):6774
- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons High-speed pulse running, online write failure
- Solution
  1. Stop high-speed pulse, and then try to online write again;
  2. When high-speed pulse completed, and then try to online write again;

#### 8.9.47 Error code 6780

- Applicable Model: TP series CPU
- Error code (D8067):6780

- Error message: Operation error
- Display error location device: D8069
- LED state (RUN): ON
- LED state (PROG.E): OFF
- PLC acting state: Run
- Diagnosis timing When PLC is running
- Abnormal content and reasons PTO instruction parameters error: The interval of the pulse frequency can not exceed the difference 20000Hz
- Solution Reconfigure the start frequency



## 9 Example

This chapter provides some examples about TETA PLC.

### 9.1 Setting for TP Series PLC COM2

COM2 port in TP Series PLC can support TETA PLC protocol and MODBUS RTU protocol.

#### 9.1.1 Protocol Setting (D8126)

Table 8-1

Protocol	Description	Value of D8126
TETA PLC Protocol	Using TETA PLC Protocol	01H
MODBUS RTU Master	PLC is slave device	02H
MODBUS ASCII Master	PLC is slave device	03H
MODBUS RTU Slave	PLC is master device	20H
MODBUS ASCII Slave	PLC is master device	30H

#### 9.1.2 Communication Format (D8120)

Table 8-2

Item	Parameter	Bit value of D8120							
		b7	b6	b5	b4	b3	b2	b1	b0
Baud rate (Bps)	115200	1	1	0	0	-	-	-	-
	57600	1	0	1	1	-	-	-	-
	38400	1	0	1	0	-	-	-	-
	19200	1	0	0	1	-	-	-	-
	9600	1	0	0	0	-	-	-	-
	4800	0	1	1	1	-	-	-	-
Stop bit	1 bit	-	-	-	-	0	-	-	-
	2 bit	-	-	-	-	1	-	-	-
Parity	None	-	-	-	-	-	0	0	-
	Odd	-	-	-	-	-	0	1	-
	Even	-	-	-	-	-	1	1	-
Data bit	7 bit	-	-	-	-	-	-	-	0
	8 bit	-	-	-	-	-	-	-	1

Example: Communication parameters is 9600.1.8.None,  $b_7b_6b_5b_4=1000$ ,  $b_3=0$ ,  $b_2b_1=00$ ,  $b_0=1$ .  
 $D8120=81H ((10000001)_2=81H$ , 81H means hexadecimal number)

### 9.1.3 TETA PLC - MODBUS (Slave) addresses rules

Table 8-3

PLC Bit Address		
PLC Address	MODBUS Address	
	Hex	Decimal
M0 ~ M3071	0 ~ 0xBFF	0 ~ 3071
M8000 ~ M8256	0x1F40 ~ 0x2040	8000 ~ 8256
S0 ~ S999	0xE000 ~ 0xE3E7	57344 ~ 58343
T0 ~ T256	0xF000 ~ 0xF100	61440 ~ 61696
C0 ~ C255	0xF400 ~ 0xF4FF	62464 ~ 62719
X0 ~ X255	0xF800 ~ 0xF9FE	63488 ~ 63998
Y0 ~ Y255	0xFC00 ~ 0xFDFE	64512 ~ 65022
PLC Word Address		
PLC Address	MODBUS Address	
	Hex	Decimal
D0 ~ D8255	0 ~ 0x203F	0 ~ 8255
T0 ~ T255	0xF000 ~ 0xF0FF	61440 ~ 61695
C0 ~ C199	0xF400 ~ 0xF4C7	62464 ~ 62663
C200 ~ C255	0xF700 ~ 0xF7FF	63232 ~ 63487

### 9.1.4 MODBUS Function Code Introduction

#### 1) Function code 0x01(01): read coil (bit address)

**Frame format:** Station number of slave&0x01 + start address + number of coils + CRC

Table 8-4

No.	Data	Number of byte	Instruction
1	Station number of slave	1 byte	Value range 1~247, set by D8121
2	0x01 (function code)	1 byte	Read coil
3	Start address	2 bytes	
4	Number of coils	2 bytes	
5	CRC	2 bytes	

#### 2) Function code 0x03(03): read register (word address)

**Frame format:** Station number of slave&0x03 + start address+ number of registers + CRC

Table 8-5

No.	Data	Number of byte	Instruction
1	Station number of slave	1 byte	Value range 1~247, set by D8121
2	0x03 (function code)	1 byte	Read register
3	Start address	2 bytes	

4	Number of registers	2 bytes	
5	CRC	2 bytes	

**3) Function code 0x05(05): write single coil**

**Frame format:** Station number of slave&0x05 + address + state of coil + CRC

Table 8-6

No.	Data	Number of byte	Instruction
1	Station number of slave	1 byte	Value range 1~247, set by D8121
2	0x05 (function code)	1 byte	Write single coil
3	address	2 bytes	
4	State of coil	2 bytes	
5	CRC	2 bytes	

**4) Function code 0x06 (06): Write single register**

**Frame format:** Station number of slave&0x06 + address + value + CRC

Table 8-7

No.	Data	Number of byte	Instruction
1	Station number of slave	1 byte	Value range 1~247, set by D8121
2	0x06 (function code)	1 byte	Write single register
3	address	2 bytes	
4	Value of register	2 bytes	
5	CRC	2 bytes	

**5) Function code 0x0f(15): Write continuous coils**

**Frame format:** Station number of slave&0x0f + start address + number of coils + length + state of coil + CRC

Table 8-8

No.	Data	Number of byte	Instruction
1	Station number of slave	1 byte	Value range 1~247, set by D8121
2	0x0f (function code)	1 byte	Write continuous coils
3	Start address	2 bytes	
4	Number of coil	2 bytes	
5	Length	1 bytes	
6	State of coils	[(N+7)/8] bytes	
7	CRC	2 bytes	

**6) Function code 0x10 (10): Write continuous registers**

**Frame format:** Station number of slave&0x10 + start address + number of registers + length + value of

register + CRC

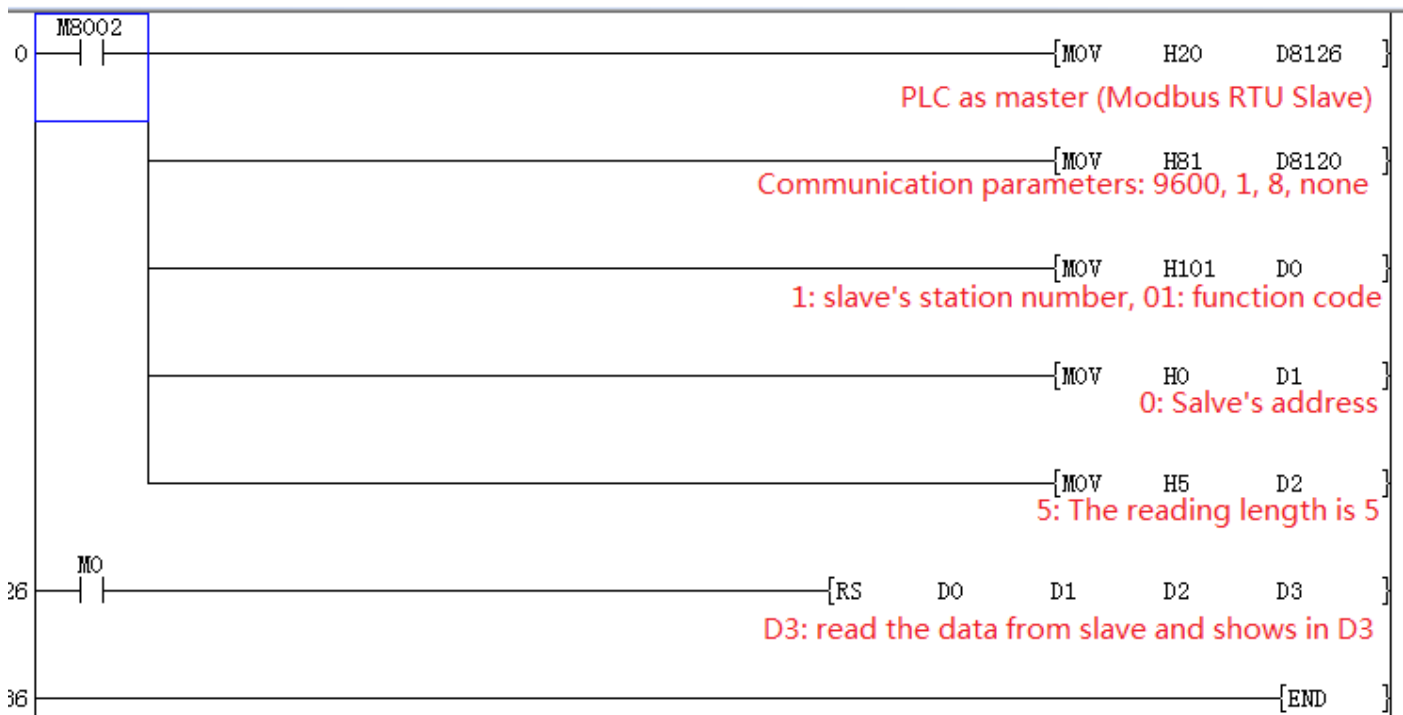
Table 8-9

No.	Data	Number of byte	Instruction
1	Station number of slave	1 byte	Value range 1~247, set by D8121
2	0x10 (function code)	1 byte	Write continuous registers
3	Start address	2 bytes	
4	Number of registers	2 bytes	
5	Length	1 bytes	
6	Value of register	N*2 bytes	
7	CRC	2 bytes	

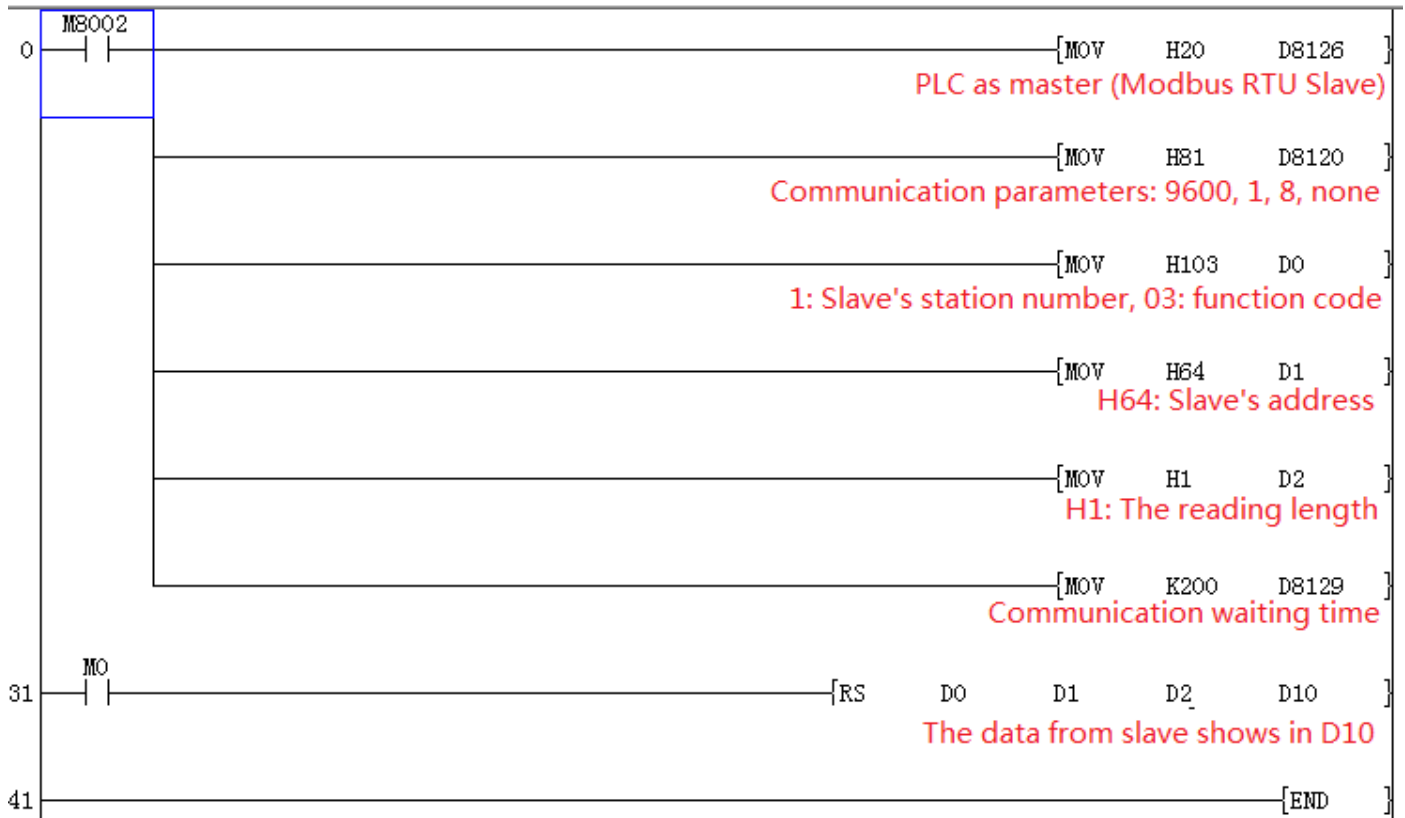
### 9.1.5 Example

PLC as master

- Read bit address from device



- Read word address from device

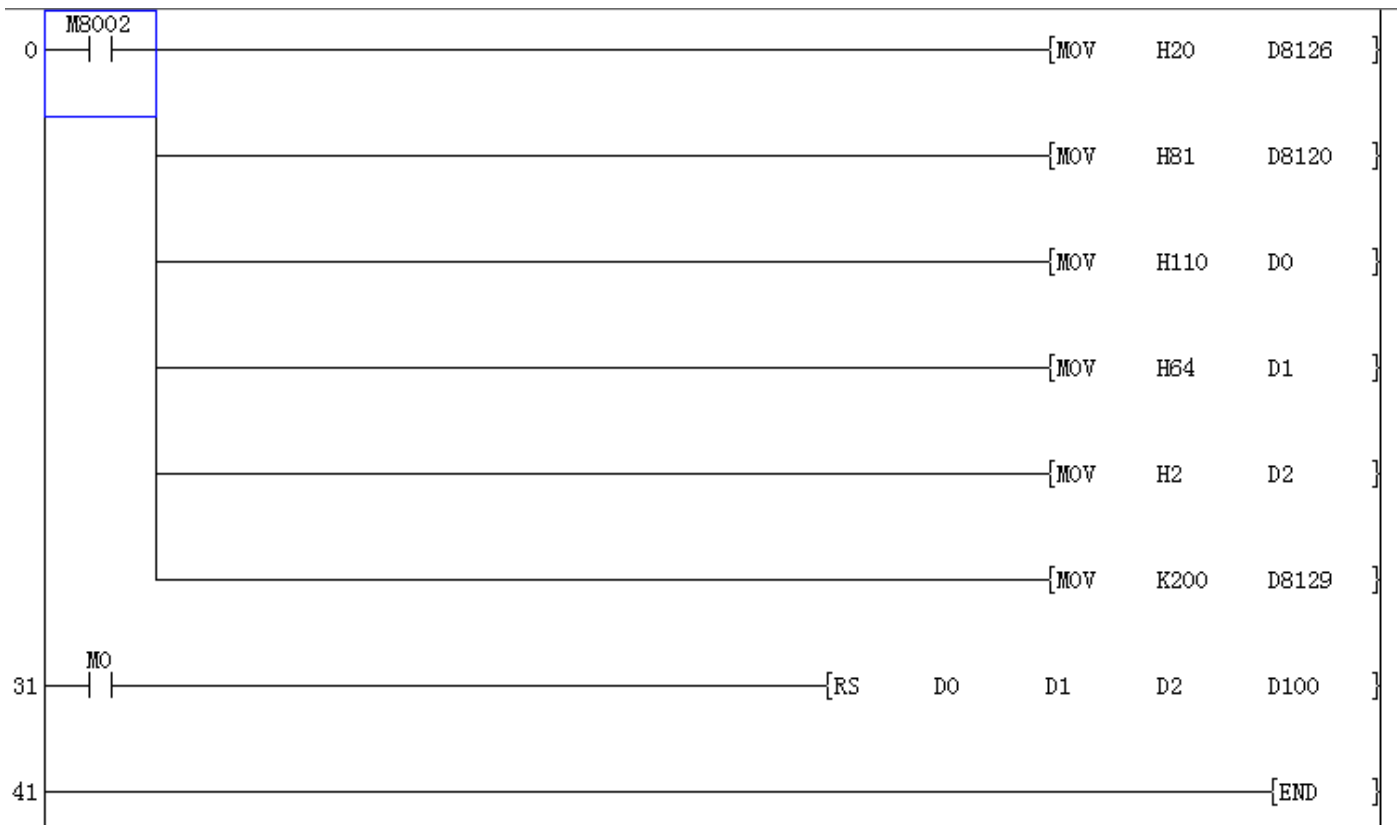


● Write single coil

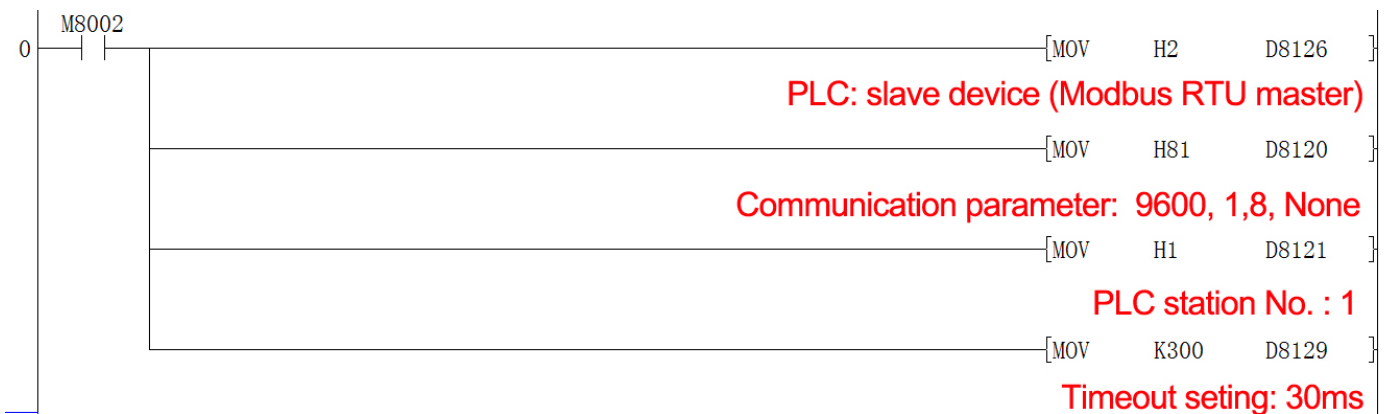




- Write continuous registers



1) PLC as slave device



2) TETA PLC Protocol (COM2)



## 9.2 N: N application in TP1 Series PLC

PLC built-in N: N connection protocol provides a effective way to exchange data among multiple PLC (Max. 8 devices). Technically, it only requires the twisted pair RS485 cable to connect with each PLC COM2 (in parallel).

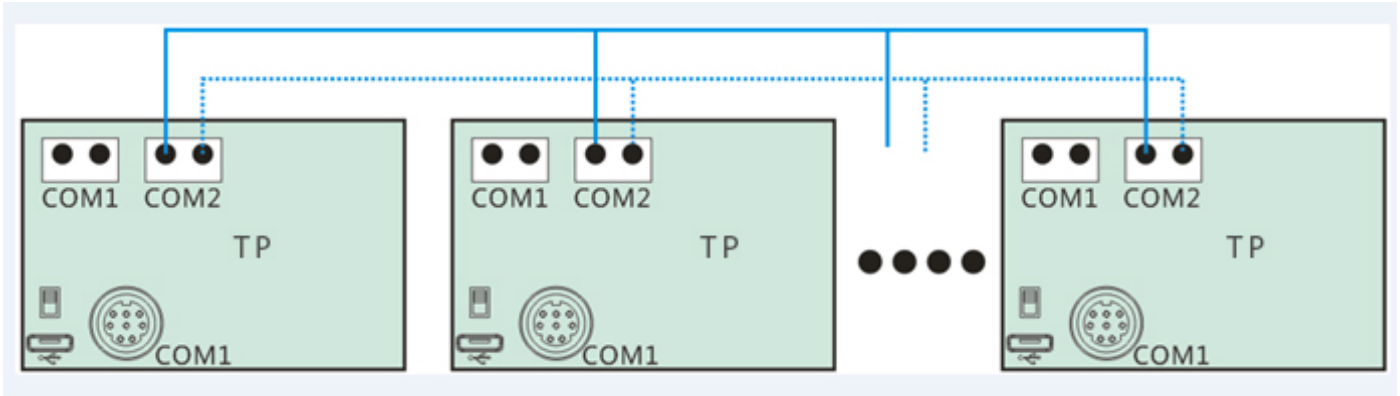


Figure 8-1

### 9.2.1 N: N Instructions

User should put data to specified register unit. Exchanging the data among PLCs, This function only requires putting data in preset register block, all the data in this register block would share with other PLCs. There are five modes for choice, according to data volume and communication speed. See the preset register block from table below.

Table 8-10

No.		Mode 0		Mode 1		Mode 2		Mode 3		Mode 4	
		Bit(M)	Word(D)	Bit(M)	Word(D)	Bit(M)	Word(D)	Bit(M)	Word(D)	Bit(M)	Word(D)
		--	station use 4 address	station use 32 address	station use 4 address	station use 32 address	station use 8 address	station use 64 address	station use 16 address	station use 64 address	station use 32 address
Slave	Master No. 0	--	D00--D03	M1000--M1031	D00--D03	M1000--M1063	D00--D07	M1000--M1063	D00--D15	M1000--M1063	D00--D31
	No. 1	--	D32--D35	M1064--M1095	D32--D35	M1064--M1127	D32--D39	M1064--M1127	D32--D47	M1064--M1127	D32--D63
	No. 2	--	D64--D67	M1128--M1159	D64--D67	M1128--M1191	D64--D71	M1128--M1191	D64--D79	M1128--M1191	D64--D95
	No. 3	--	D96--D99	M1192--M1223	D96--D99	M1192--M1255	D96--D103	M1192--M1255	D96--D111	M1192--M1255	D96--D127
	No. 4	--	D128--D131	M1256--M1287	D128--D131	M1256--M1319	D128--D135	M1256--M1319	D128--D143	M1256--M1319	D128--D159



N o. 5	--	D160-- D163	M1320- -M1351	D160-- D163	M1320- -M1383	D160-- D167	M1320- -M1383	D160-- D175	M1320- -M1383	D160-- D191	
	N o. 6	--	D192-- D195	M1384- -M1415	D192-- D195	M1384- -M1447	D192-- D199	M1384- -M1447	D192-- D207	M1384- -M1447	D192-- D223
	N o. 7	--	D224-- D227	M1448- -M1479	D224-- D227	M1448- -M1511	D224-- D231	M1448- -M1511	D224-- D239	M1448- -M1511	D224-- D255

Communicating with each PLC (up to 8 PLC), please see the connection construction below (For 3 PLC inter-connection).



### 9.2.2 System registers

Table 8-11

Register	Description
D8120	Communication format settings
D8126	COM2 communication protocol settings, 40h means N:N Master Device, 04h means N:N Slave device
D8176	Station number, from 0 to 7, 0 means master device
D8177	Total station number, from 1 to 7, only required for master device.
D8178	Register block setting, from 0 to 5, only required for master device.
D8179	Retry count settings, only required for master device.
D8180	Timeout setting, unit: 10ms only required for master device.
D8201	Current connection scan time
D8202	Maximum connection scan time
D8203	Master error counter
D8204~D8210	Slave error counter
D8211	Master N:N error code
D8212~D8218	Slave N:N error code
M8183	Master data transfer sequence error
M8183~M8190	Communication error flag: M8183 - No.0 (Master); M8184 - No. 1 (Slave 1) ..... M8190 - No. 7 (Slave 7)
M8191	Processing sending data

N: N Error code (D8211~D8218)

Table 8-12

Error	Description
1	Monitor time exceeded, no response from slave device
2	Responded by other slave device
3	Different count value between parameters and slave responses
4	Incorrect responses from slave device
17	Monitor time exceeded, Master failed to send request to next slave device
20	Incorrect responses from master device
33	Slave device no response
34	Responded by other slave device
35	Different count value between parameters and slave responses
49	Request received from mater device, while no parameters received

Communications format

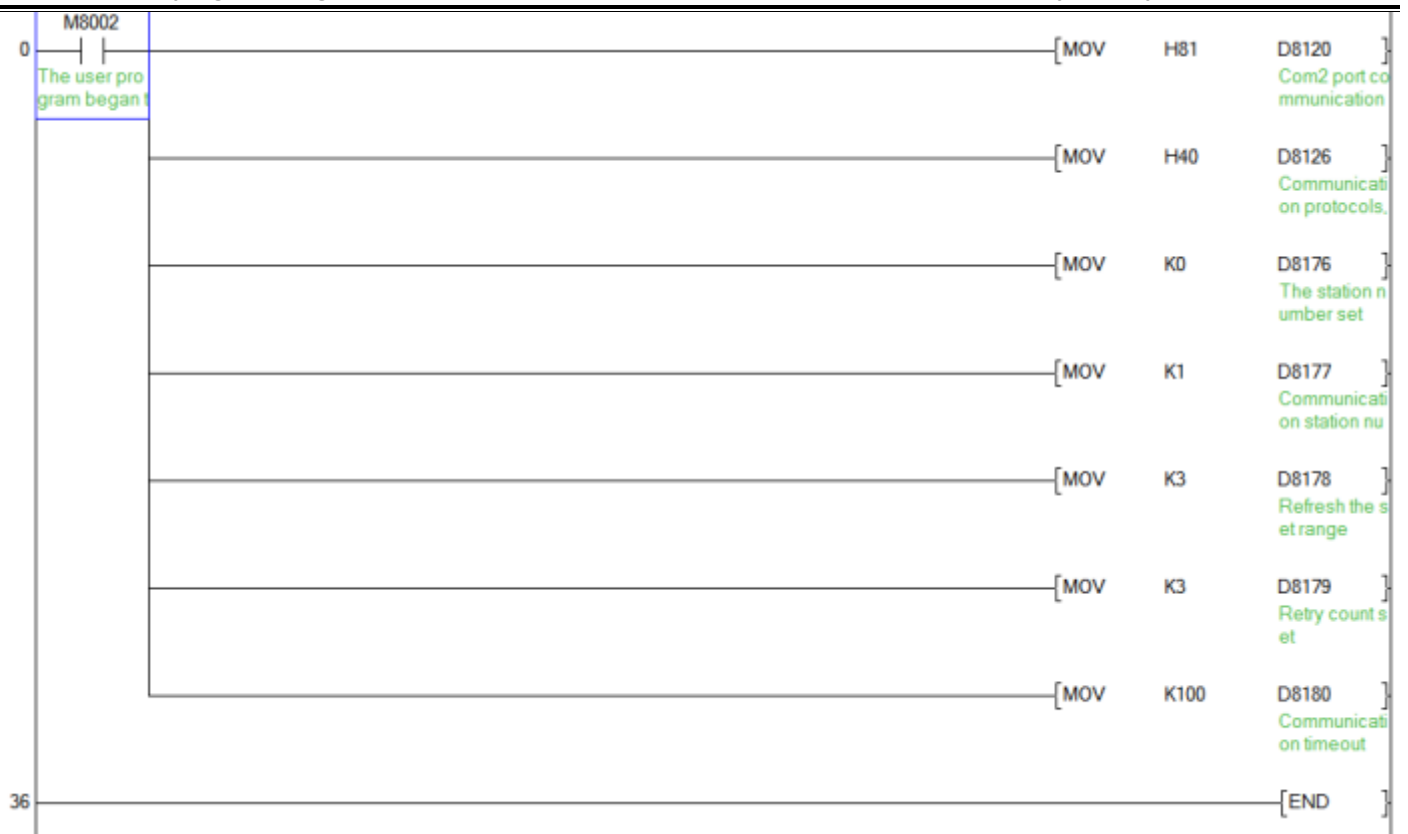
Table 8-13

D8120			
Bit	Functions	Descriptions	
		0(OFF)	1(ON)
B0	Date length	7 bits	8 bits
B2B1	Parity	00:None 01:Odd 11:Even	
B3	Stop Bit	1 bit	2 bits
B7b6b5b4	Baud rate(bps)	0111:4800 1001:19200 1011:57600	1000:9600 1010:38400 1100:115200

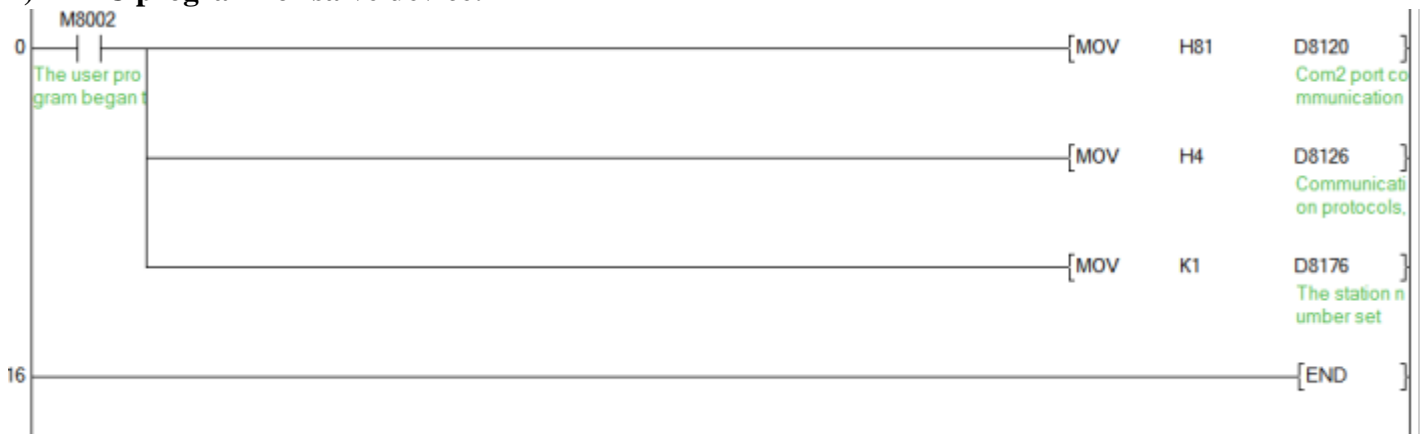
9.2.3 Example

TP0 COM2 port communication parameters: 9600, 1, 8, NONE. Register block mode: 3.

1) PLC program for Master device



2) PLC program for slave device:



9.2.4 Notice

There are two modes of N: N protocol configuration. One is TP0 built-in N:N protocol, the other one is TP0 N:N protocol (TP-2RS485 expansion card required).

In TP0 series PLC, only one kind of N: N configuration available. Second mode would be disabled when TP0 built-in N: N protocol configure.