WECON

PLC

(LX6V series)Programming manual



FUJIAN FUCAN WECON CO LTD

Content

This manual has a comprehensive introduction to the basic functions of WECON PLC Editor2 and the actual use. This book is completely aimed at zero-based readers, is an essential reference book for entry-level readers to quickly and fully grasp WECON PLC and WECON PLC Editor2.

This book starts from the basic product of WECON PLC and the basic concept and operation of WECON PLC Editor2. It combines with a large number of cases and graphic analysis to comprehensively and deeply explain the use of WECON PLC Editor2 Software, as well as PLC program. All rights of interpretation for this book are reserved by FUJIAN FUCAN WECON CO LTD.

✎Safety Note

Before the installation, operation, maintenance and repair of the micro programmable control, please read this manual and other related manual to ensure correct use. Please use it after you have mastered the operation method, safety information and all

✎Note:

(1) Design considerations

In the event of an abnormality in the external power supply or failure of the programmable controller, to ensure the safe operation of the entire system, be sure to install a safety circuit outside the programmable controller.

1) Be sure to install an emergency brake circuit, a protection circuit, an interlock circuit for reverse operation, such as an emergency brake circuit, a protection circuit, a forward and reverse circuit, and an interlock circuit for the upper and lower positioning limits to prevent machine damage, in the external circuit of the programmable controller.

2) When the programmable controller CPU detects abnormal conditions such as WDT errors through self-diagnosis, all outputs are shut off. In addition, when the programmable controller CPU cannot detect abnormalities in the input/output control part, etc., it cannot control the output. At this time, in order to make the machine operate safely, please design the external circuit and mechanism.

3) Due to the failure of the relay and transistor of the output unit, it is impossible to control the state of the output to ON or OFF. In order to ensure the safe operation of the machine, please design external circuits and mechanisms for output signals related to major accidents.

(2) Installation Precautions

1) Please use it in the general specification environment described in the manual.

Do not use in the following places: places with dust, oily smoke, conductive dust, corrosive gas, flammable gas; places exposed to high temperature, condensation, wind and rain; places with vibration or impact. Electric shock, fire, and misoperation can also cause product damage.

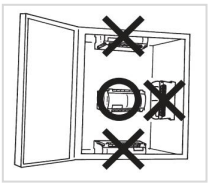
2) When processing screw holes and wiring, do not let iron filings or wire ends fall into the ventilation window of the programmable controller. It may cause fire, malfunction, or misoperation.

3) Please insert the connecting cable and display module accurately into the specified sockets. Poor contact may cause misoperation.

Note

● To prevent the temperature from rising, do not install at the bottom, top or vertical direction.Be sure to install it horizontally on the wall as shown on the right.

● Please leave a space of more than 50mm between the host and other equipment or structures. Try to avoid high-voltage lines, high-voltage equipment, and power equipment.



(3) Wiring considerations

The signal input and output lines of the programmable controller cannot pass through the same cable.

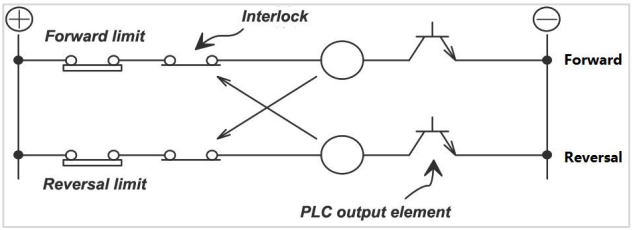
In addition, signal input lines and output lines cannot pass through the same pipeline with other power lines and output lines, and cannot be bundled together.

If implemented according to the above precautions, even if the input and output wiring is as long as 50 to 100m, there is almost no noise problem. But generally for safety, the wiring length should be within 20m.

Note

● The installation and wiring must be performed when the external power supply is cut off. Otherwise, it may cause electric shock or product damage.

● After installation and wiring, etc., the terminal cover must be installed before power-on operation to avoid electric shock.

Danger		It is very dangerous to close the positive and negative contactors at the same time. For loads like this, in addition to the interlock set by the internal program of the programmable controller, the interlock shown above must also be set outside the programmable controller.
Attention	Do not connect the empty terminal to the outside, otherwise the product will be damaged.	

● Please connect the AC power supply to the dedicated terminal according to the content in the manual.

If the AC power supply is connected to the DC input/output terminal or the DC power supply terminal, the programmable controller will be burnt out.

● Please do not supply power to the 24+ terminal of the basic unit from the external power supply, and to the empty terminal □.

Do not wire from the outside, otherwise the product will be damaged.

Please ground the ground terminal of the basic unit according to the third method. But please do not share the ground with the strong current system.

Attached

● The programmable controller will continue to work if there is an instantaneous power failure of less than 10ms.

● When the power is cut for a long time or the voltage is low, the programmable controller will stop working and the output will turn OFF, but once the power supply is restored, it will automatically restart.

(4) Precautions for startup and maintenance

Danger
● Please do not touch the terminals when the power is on, otherwise it may cause electric shock or misoperation. ● Please clean and disassemble the terminal after the power is turned off. Performing it while the power is on may cause electric shock. ● Please read the manual thoroughly and fully confirm the safety before proceeding with program changes, forced output, RUN/STOP, etc. during machine operation. Operation errors can damage the machine and cause accidents.

Caution
● Please do not disassemble or modify, otherwise it may cause malfunction, malfunction, fire. ※For repair matters, please contact Fuzhou Wecon Electronic Technology Co., Ltd. ● After the power is turned off, perform the installation and disassembly of the extension cables and other connecting cables, otherwise it may cause malfunctions.

(5) Maintenance

● Regular inspection: Whether the programmable controller is equipped with consumables with a shorter life.

● For relay output type, if the output relay works abnormally at a high frequency or drives a large-capacity load, you must pay attention to its impact on the service life.

● Check with other equipment, please pay attention to the following points:

Whether there is an abnormal increase in the temperature inside the machine due to other heating elements or direct sunlight.

Whether dust or conductive dust has penetrated into the machine.

Whether there are loose wiring and terminals or other abnormalities

Content

Content **2**

1 Execution of the program **1**

1.1 Composition of the scan 1

1.1.1 Initialization 1

1.1.2 Input and output point refresh 1

1.1.3 Operation of the program 1

1.1.4 END processing 1

1.2 Scan time 2

1.2.1 Initial scan time 2

1.3 The flow of each procedure 3

1.4 Types of program execution 4

1.4.1 Scan execution program 4

1.4.2 Event execution program 5

1.4.3 Interrupt execution program 8

1.4.4 Subroutine 18

1.4.5 Positioning instructions 19

2 Description of device **21**

2.1 User device 22

2.1.1 Input relay (X) 22

2.1.2 Output relay (Y) 22

2.1.3 Internal relay (M) 22

2.1.4 Status relay (S) 22

2.1.5 Timer (T) 22

2.1.7 Long counter (LC) 24

2.1.8 High-speed counter (HSC) 24

2.1.9 Data register (D & R) 25

2.2 System device 26

2.2.1 Special relay (SM) 26

2.2.2 Special register (SD) 26

2.3 Index register 26

2.3.1 Index Register ([D]) 26

2.4 Pointer 26

2.4.1 Pointer (P) 26

2.5 Constant 26

2.5.1 Decimal constant (K) 26

2.5.2 Hexadecimal constant (H) 27

2.5.3 Real number constant (E) 27

2.5.4 String constant 27

2.6 Power-down retention setting 27

2.7 Special use of device 28
3 Sequence control program instructions 29
3.1 Contact instructions 29
3.1.1 Operation start, series connection, parallel connection 29
3.1.2 Pulse calculation starts,pulse series connection, pulse parallel connection 33
3.2 Combining instructions 36
3.2.1 Series connection and parallel connection of ladder diagram blocks 36
3.2.2 Invert the result of operation 37
3.2.3 Pulse operation result 38
3.3 Output instructions 39
3.3.1 OUT instruction (except timers and counters) 39
3.3.2 SET instruction 41
3.3.3 RST instruction 43
3.3.4 PLF/Falling edge output 45
3.3.5 PLS/Rising edge output 46
4 Program flow instructions 47
4.1 Program jump 47
4.1.1 CJ/Conditional jump 47
4.2 Subroutine jump 52
4.2.1 CALL/Subroutine call 52
4.3 Interrupt disable, interrupt enable 55
4.3.1 DI and EI/Interrupt prohibited and allowed 55
4.3.2 SIMASK/Interrupt mask 59
4.4 Cycle instructions 61
4.4.1 FOR to NEXT/Cycle 61
4.4.2 BREAK/Break cycle 63
4.5 Watchdog reset 65
4.5.1 WDT/Watchdog timer 65
5 Timer and counter output instructions 67
5.1 Timer output instruction 67
5.1.1 OUT T/Timer output 67
5.2 Counter output instructions 69
5.2.1 OUT C/Counter output 69
5.2.2 OUT LC instruction/Long counter output 69
6 Basic instructions 71
6.1 Transfer comparison instruction 71
6.1.1 MOV/16-bit transmission 71
6.1.2 DMOV/32-bit transmission 72
6.1.3 BMOV/Batch transmission 73
6.1.4 FMOV/16-bit multicast 74

6.1.5 DFMOV/ 32-bit multicast 75
6.1.6 SMOV/Bit shift 76
6.1.7 CML/16-bit invert transmission 78
6.1.8 DCML/32-bit invert transmission 79
6.1.9 CMP/16-bit data comparison output 80
6.1.10 DCMP/32-bit data comparison output 81
6.1.11 XCH/16-bit data exchange 82
6.1.12 DXCH/32-bit data exchange 84
6.1.13 ZCP/16-bit data interval comparison 86
6.1.14 DZCP/32-bit data interval comparison 88
6.2 Cycle shift instruction 90
6.2.1 ROR/16-bit cycle shift right 90
6.2.2 DROR/32-bit cycle shift right 92
6.2.3 RCR/16-bit cycle shift right with carry 94
6.2.4 DRCR/32-bit cycle shift right with carry 96
6.2.5 ROL/16-bit cycle shift left 98
6.2.6 DROL/32-bit cycle shift left 100
6.2.7 RCL/16-bit cycle shift left with carry 102
6.2.8 DRCL/32-bit cycle shift left with carry 104
6.2.9 SFTR/n-bit shift right of n-bit data 106
6.2.10 SFTL/n-bit shift left of n-bit data 108
6.2.11 WSFR/n-word shift right of n-word data 109
6.2.12 WSFL/n-word shift left of n-word data 111
6.2.13 SFR/n-bit shift right of 16-bit data 113
6.2.14 DSFR/n word data shift right by 1 word 114
6.2.15 SFL/n-bit shift left of 16-bit data 116
6.2.16 DSFL/one word shift left of n word data 118
6.3 Arithmetic operation instructions 120
6.3.1 ADD/16-bit addition operation 120
6.3.2 DADD/32-bit addition operation 121
6.3.3 SUB/16-bit subtraction operation 124
6.3.4 DSUB/32-bit subtraction operation 126
6.3.5 MUL/16-bit multiplication 128
6.3.6 DMUL/32-bit Multiplication 129
6.3.7 DIV/16-bit division operation 130
6.3.8 DDIV/32-bit division operation 131
6.3.9 INC/16-bit data increment 132
6.3.10 DINC/32-bit data increment 133
6.3.11 DEC/16 bit data decrement 134
6.3.12 DDEC/32-bit data decrement 135
6.4 Logic operation instructions 136
6.4.1 NEG/16-bit complement 136
6.4.2 DNEG/32-bit complement 138

6.4.3 WOR/16-bit data logical OR 139
6.4.4 DOR/32-bit data logical OR 140
6.4.5 WAND/16-bit data logic AND 141
6.4.6 DAND/32-bit data logic AND 142
6.4.7 WXOR/16-bit data logic exclusive OR 143
6.4.8 DXOR/32-bit data logic exclusive OR 144
6.4.9 ANS/Alarm settings 145
6.4.10 ANR/Alarm reset 147
6.4.11 BON/16-bit data bit judgment 149
6.4.12 DBON/32-bit data bit judgment 150
6.4.13 ENCO/Encode 151
6.4.14 DECO/Decode 152
6.4.15 SUM/The ON bits of 16-bit data 153
6.4.16 DSUM/The ON bits of 32-bit data 154
6.4.17 MEAN/Average value of 16-bit data 155
6.4.18 DMEAN/Average value of 16-bit data 156
6.4.19 SQR/16-bit square root 157
6.4.20 DSQR/32-bit square root 158
6.4.21 WSUM/The sum value of 16-bit data 159
6.4.22 DWSUM/The sum value of 32-bit data 160
6.4.23 SORT/16-bit data sorting 161
6.4.24 SORT2/16-bit data sorting 165
6.4.25 DSORT2/32-bit data sorting 168
6.4.26 SWAP/16-bit data high and low byte swap 171
6.4.27 DSWAP/32-bit data high and low byte swap 172
6.4.28 BTOW/Byte unit data merge 173
6.4.29 WTOB/Byte unit data separation 175
6.4.30 DIS/4-bit separation of 16-bit data 177
6.4.31 UNI/4-bit combination of 16-bit data 178
6.4.32 ZRST/Data batch reset 179
6.4.33 ZSET/Data batch set 181
6.4.34 CRC/Cyclic redundancy check instruction 182
6.5 Matrix input instructions 185
6.5.1 MTR/Matrix input 185
6.6 Convenient instructions 187
6.6.1 ABSD/BIN 16-bit data absolute method 187
6.6.2 DABSD/BIN 32-bit data absolute method 189
6.6.3 SER/16-bit data search 191
6.6.4 DSER/32-bit data search 193
6.6.5 ALT/Bit device output inversion 195
6.6.6 INCD/BIN 16-bit data relative method 197
6.6.7 RAMP/Control ramp signal 199
6.6.8 ROTC/Rotary table proximity control 201

6.6.9 STMR/Special function timer 204
6.6.10 TTMR/Demonstration timer 206
6.7 External IO instructions 208
6.7.1 ARWS/Arrow switch 208
6.7.2 DSW/Numeric key input 211
6.7.3 HKY/Hexadecimal numeric key input 213
6.7.4 DHKY/32 System Numeric Key Input 216
6.7.5 PR/ASCII code printing 218
6.7.6 SEGD/Numeric key input 220
6.7.7 SEGL/7SEG vode hour and minute display 221
6.7.8 TKY/Numeric key input 224
6.7.9 DTKY/Numeric key input 226
6.8 Data conversion instruction 228
6.8.1 BCD/BIN → BCD 228
6.8.2 BIN/4-bit BCD → BIN 230
6.8.3 DBIN/8-bit BCD → BIN 232
6.8.4 FLT/BIN Integer → Binary floating point number 234
6.8.5 DFLT/BIN Integer → Binary floating point number 236
6.8.6 VAL/ String → BIN 16-bit data conversion 237
6.8.7 DVAL/String → BIN32-bit Data Conversion 239
6.8.8 ASCI/HEX code data →ASCII conversion 241
6.8.9 HEX/ASCII → HEX code data conversion 244
6.8.10 CCD/Check Code 247
6.8.11 GBIN/Gray code → BIN 16-bit data conversion 250
6.8.12 DGBIN/Gray code → BIN32-bit data conversion 251
6.8.13 GRY/BIN 16-bit data → Gray code conversion 252
6.8.14 DGRY/BIN 32-bit data → Gray code conversion 253
6.8.15 PRUN/Octal digit transmission (16-bit data) 254
6.8.16 DPRUN/Otal Digit Transmission (32-bit data) 256
6.9 Floating point instructions 257
6.9.1 DACOS/Single precision real number COS-1 operation 257
6.9.2 DASIN/Single precision real number SIN^{-1} operation 258
6.9.3 DATAN/Single precision real number TAN^{-1} operation 259
6.9.4 DCOS/Single precision real number COS operation 260
6.9.5 DCOSH/Single precision real number COSH operation 261
6.9.6 DSIN/Single precision real number SIN operation 262
6.9.7 DSINH/Single precision real number SINH operation 263
6.9.8 DTAN/Single precision real number TAN operation 264
6.9.9 DATANH/Single precision real number TANH operation 265
6.9.10 DDEG/Single precision real number radian → Angle conversion 266
6.9.11 DRAD/Single precision real number conversion angle → Radian conversion 267
6.9.12 DEADD/Single precision real number addition operation 268
6.9.13 DESUB/Single Precision real number subtraction operation 270

6.9.14 DEMUL/Single precision real number multiplication operation 272
6.9.15 DEDIV/Single precision real number division operation 274
6.9.16 DEMOV/Single precision real data transmission 276
6.9.17 DEBCD/Binary floating point → Decimal floating point conversion 277
6.9.18 DEBIN/Decimal floating point → Binary floating point conversion 278
6.9.19 DENEG/Single precision real number sign inversion 279
6.9.20 DECMP/Single precision real number comparison 280
6.9.21 DEZCP/Binary floating point bandwidth comparison 282
6.9.22 DESQR/Single precision real square root 284
6.9.23 DESTR/Single precision real number → String conversion 285
6.9.24 DEVAL/String → Single precision real number conversion 290
6.9.25 DEXP/Single precision real number exponent operation 294
6.9.26 INT/Single precision real number → Signed BIN 16-bit data 295
6.9.27 DINT/Single precision real number→ Signed BIN 32-bit data 296
6.9.28 DLOG10/Single precision real number common logarithmic operation 297
6.9.29 DLOGE/Single precision real number natural logarithm operation 298
6.10 Contact comparison instruction 299
6.10.1 Signed 16-bit contact comparison instruction 299
6.10.2 Signed 32-bit contact comparison instruction 301
6.10.3 Single precision real number contact comparison instruction 303
6.10.4 String comparison 305
6.11 Clock operation instruction 307
6.11.1 TADD/The addition of clock data 307
6.11.2 TSUB/The subtraction of clock data 309
6.11.3 TRD/Clock data reading 311
6.11.4 TWR/Clock data writing 312
6.11.5 HTOS/16-bit data conversion of time data (hour, minute, second → second) 314
6.11.6 DHTOS/32-bit data conversion of time data (hour, minute, second → second) 315
6.11.7 HOUR/Hour measuring 16-bit 316
6.11.8 DHOUR/Hour measuring 32 bits 318
6.11.9 STOH/16-bit data conversion of time data (second → hour, minute, second) 320
6.11.10 DSTOH/32-bit data conversion of time data (second → hour, minute, second) 321
6.11.11 TCMP/Clock data comparison 322
6.11.12 TZCP/Clock data bandwidth comparison 324
6.12 Data control instructions 326
6.12.1 BAND/BIN 16-bit data dead zone control 326
6.12.2 DBAND/BIN 32-bit data dead zone control 328
6.12.3 BINDA/BIN 16-bit data → Decimal ASCII conversion 330
6.12.4 DBINDA/BIN 32-bit Data → Decimal ASCII conversion 332
6.12.5 DABIN/Decimal ASCII → BIN conversion 333
6.12.6 DDABIN/Decimal ASCII → BIN32-bit data conversion 335
6.12.7 LIMIT/ BIN 16-bit data high and low limit control 337
6.12.8 DLIMIT/BIN 32-bit data high and low limit control 338

6.12.9 SCL/BIN 16-bit unit scale (coordinate data of each point) 339
6.12.10 DSCL/32-bit unit scale (coordinate data of each point) 342
6.12.11 SCL2/BIN 16-bit unit scale (x/y coordinate data) 345
6.12.12 DSCL2/BIN 32-bit unit scale (x/y coordinate data) 348
6.12.13 ZONE/BIN 16-bit data zone control 352
6.12.14 DZONE/BIN 32-bit data zone control 354
6.13 Data block instructions 356
6.13.1 BK+/BIN 16-bit block data addition operation 356
6.13.2 DBK+/BIN 32-bit block data addition operation 358
6.13.3 BK-/BIN 16-bit block data subtraction operation 360
6.13.4 DBK-/BIN 32-bit block data subtraction operation 362
6.13.5 BKCMP=/BIN 16-bit Block Data Comparison 364
6.13.6 DBKCMP=/BIN32-bit block data comparison 366
6.13.7 BKCMP<>/BIN 16-bit block data comparison 368
6.13.8 DBKCMP<>/BIN32-bit block data comparison 370
6.13.9 BKCMP>/BIN 16-bit block data comparison 372
6.13.10 DBKCMP>/BIN32-bit block data comparison 374
6.13.11 BKCMP>=/BIN 16-bit block data comparison 376
6.13.12 DBKCMP>=/BIN32-bit block data comparison 378
6.13.13 BKCMP</BIN 16-bit block data comparison 380
6.13.14 DBKCMP</BIN 32-bit block data 382
6.13.15 BKCMP<=/BIN16-bit block data cmparison 384
6.13.16 DBKCMP<=/BIN32-bit block data comparison 386
6.14 Data table operation instructions 388
6.14.1 SFRD/Shift read 388
6.14.2 POP/Read from the back of the data table 390
6.14.3 SFWR/Shift write 392
6.14.4 FINS/Data table data insertion 395
6.14.5 FDEL/Data deletion of data sheet 397
6.15 IO Refresh instruction 399
6.15.1 REF/IO refresh 399
6.15.2 REFF/Input refresh (With filter setting) 401
6.16 Timing measure instruction 402
6.16.1 DUTY/Clock pulse generation instruction 402
6.17 Random number instruction 404
6.17.1 RND/Random number instruction 404
6.18 Preferred instruction 405
6.18.1 DEXMN/Preferred instruction 405
7 EtherCAT 411
7.1 Description 411
7.2 Master station configuration 411

7.3 Import and add slaves 412
7.3.1 Import configurations 412
7.3.2 Add a slave station 412
7.3.3 Copy to new item 412
7.4 Slave station configuration 413
7.4.1 Delete 413
7.4.2 Update devices 413
7.4.3 Slave station parameters 414
7.5 Alias 417
7.5.1 Alias parameter configuration 417
7.5.2 Alias writes to servo 420
7.5.3 Simple configuration instance 421
7.5.4 Expert parameters 424
7.5.5 Process data 425
7.5.6 I/O mapping 425
7.5.7 Start parameters 426
7.5.8 Slave station information 426
7.6 Device scanning 427
7.7 EtherCAT instruction 430
7.7.1 ETC_READPARAMETER_COE 430
7.7.2 ETC_WRITEPARAMETER_COE 433
7.7.3 IODRVETHERCAT 436
7.8 AbrotCode exception codes 438
8 Motion control 439
8.1 Management function blocks for single-axis motion control 439
8.1.2 MC_READSTATUS 441
8.1.3 MC_READMOTIONSTATE 443
8.1.4 MC_READAXISERROR 445
8.1.5 MC_READAXISINFO 447
8.1.6 MC_READPARAMETER 450
8.1.7 MC_READBOOLPARAMETER 452
8.1.8 MC_READACTUALVELOCITY 455
8.1.9 MC_READACTUALPOSITION 457
8.1.10 MC_RESET 459
8.1.11 MC_SETPOSITION 461
8.1.12 MC_TouchProbe 463
8.1.13 MC_AbortTrigger 466
8.2 Motion Function Blocks for Single-axis Motion Control 468
8.2.1 MC_STOP 468
8.2.2 MC_HOME 470
8.2.3 MC_HALT 472
8.2.4 MC_MOVEABSOLUTE 474

8.2.5 MC_MOVERELATIVE 478
8.2.6 MC_MOVEADDITIVE 483
8.2.7 MC_MOVESUPERIMPOSED 485
8.3 Management Function Blocks for Multi-axis Motion Control 487
8.3.1 MC_CAMTABLESELECT 487
8.3.2 MC_GENERATECAMTABLE 489
8.3.3 MC_SAVECAMTABLE 491
8.4 Motion Function Block for Multi-axis Motion Control 493
8.4.1 MC_CAMIN 493
8.4.2 MC_CAMOUT 498
8.4.3 MC_GEARIN 500
8.4.4 MC_GEAROUT 504
8.4.5 MC_GEARINPOS 505
8.5 Management functions of axis group motion control 509
8.5.1 MC_ADDAXISTOGROUP 509
8.5.2 MC_REMOVEAXISFROMGROUP 511
8.5.3 MC_UNGROUPALLAXES 513
8.5.4 MC_GROUPENABLE 514
8.5.5 MC_GROUPDISABLE 516
8.5.6 MC_GROUPREDACTUALPOSITION 517
8.5.7 MC_GROUPREADACTUALVELOCITY 518
8.5.8 MC_GROUPREADERROR 519
8.5.9 MC_GROUPREADSTATUS 520
8.5.10 MC_GROUPRESET 522
8.5.11 MC_GROUPREADACTUALACCELERATION 524
8.6 Axis group motion Function Blocks 526
8.6.1 MC_GROUPSTOP 526
8.6.2 MC_GROUPHALT 528
8.6.3 MC_MOVELINEARABSOLUTE 530
8.6.4 MC_MOVELINEARRELATIVE 532
8.6.5 MC_MOVECIRCULARABSOLUTE 534
8.6.6 MC_MOVECIRCULARRELATIVE 537
8.6.7 MC_MOVEDIRECTABSOLUTE 540
8.6.8 MC_MOVEDIRECTRELATIVE 543
8.7 Other special FB 544
8.7.1 MC_MOVEJOG 545
8.7.2 MC_COMBINEAXES 547
8.7.3 MC_MOVEVELOCITY 550
8.7.4 MC _ MOVEVELOCITY _ CSV 552
8.7.5 MC _ SYNCMOVEVELOCITY 555
8.7.6 MC _ SETTORQUE 557
8.7.7 MC _ TORQUECONTROL 560

8.7.8 MC _ MOVEABSRELPP 563
8.7.9 MC _ MOVEVELOCITYPV 565
8.7.10 MC _ TORQUECONTROLPT 567
8.8 SDO instruction 570
8.8.1 MC_READSDO reads SDO data 570
8.8.2 MC_WRITESD0 writes SDO data 571
8.9 Built-in variables of pulse axis system 572
8.10 Others 578
8.10.1 Single axis motion control status diagram 578
8.10.2 Axis group motion control status diagram 579
8.10.3 Arc mode description 580
8.10.4 Transition mode description 580
8.10.5 Mode selection instructions 581
8.11 Error Code 588
9 High Speed counter 593
9.1 High-speed counter (Encoder axis) Introduction 593
9.1.1 High-speed counter axis introduction 593
9.1.2 Create encoder axis 593
9.1.3 Encoder axis user unit and conversion 594
9.1.4 Work mode setting 597
9.1.5 Setting counter parameters 600
9.2 HC Axis control 603
9.2.1 HC_COUNTER high speed counter enable 603
9.2.2 HC_PRESET 607
9.2.3 HC_TOUCHPROBE High speed counter touch probe 610
9.2.4 HC_COMPARE High speed counters comparison 615
9.2.5 HC_ARRAYCOMPARE High speed counter array comparison 618
9.2.6. HC_STEPCOMPARE high speed counters equidistant comparison 622
9.3 Encoder axis system built-in variables 626
10 Communication instruction 629
10.1 Communication port protocol setting 629
10.1.1 Communication port protocol setting 629
10.2 Modbus serial port parameter setting 631
10.2.1 PORTPARA/Modbus serial port parameter setting 631
10.3 Modbus station number setting 633
10.3.1 STATION/Modbus station number setting 633
10.4 RS Instruction 635
10.4.1 RS/External communication instruction 635
10.5 Instructions for use of RS instruction protocol 639
10.5.1 User-defined protocol description 639

10.5.2 Modbus protocol description 648
10.6 WECON Modbus protocol description 660
10.7 Communication function block 661
1. SERIAL_MODBUS_WR 661
2. SERIAL_WR 665
11 PID Control Instruction 668
11.1 PID/PID calculation 668
11.2 CCPID/CCPID calculation 671
11.3FPID/FPID calculation 672
11.4 CCPID instruction introduction manual 676
11.5 TRH/Wet and dry bulb temperature and humidity conversion 686
12 Character string instruction 688
12.1 LEN/Length detection of string 688
12.2 LEFT/Extracting from the left side of the string 689
12.3 RIGHT/Extracting from the right side of the string 691
12.4 MIDR/arbitrary extraction in the string 693
12.5 $MOV/string transfer 695
12.6 MIDW/arbitrary substitution in the string 697
12.7 STR/BIN16-bit data→string conversion 700
12.8 DSTR/BIN32-bit data→string conversion 703
12.9 $+/combination of string 706
12.10 INSTR/string search 708
12.11 ASC/ASCII data input 710
13 Ethernet communication 712
13.1 Overview of Ethernet 712
13.1.1 IP Address concept 712
13.1.2 Setting PC Network address 712
13.1.3 Testing Network connection status 713
13.1.4 Ethernet connection between PLC Editor2 and PLC 715
13.1.5 Ethernet Search Function of PLC Editor 2 716
13.2 Ethernet configuration 717
13.2.1 IP Address setting 717
13.2.2 TCP Protocol concept 718
13.3 List of Ethernet-related special devices 719
13.4 List of Ethernet Operation Error Codes 726
14 Built-in FC/FB 731
14.1 Built-in FC function 731
14.1.1 ADD/Addition operation instruction 731
14.1.2 SUB/Subtraction operation instruction 733
14.1.3 MUL/Multiplication operation instruction 735
14.1.4 DIV/Division operation instruction 736

14.1.5 MOD/Remainder operation instruction 737
14.1.6 ABS/Absolute value (Integer) 738
14.1.7 ABS/Absolute Value (Real) 738
14.1.8 ASIN/SIN^{-1} Operation 739
14.1.9 ATAN/TAN^{-1} Operation 740
14.1.10 ACOS/COS^{-1} Operation 741
14.1.11 COS/COS Operation 742
14.1.12 SIN/SIN Operation 743
14.1.13 TAN/TAN Operation 744
14.1.14 SQRT/Square root operation 745
14.1.15 EXP/Exponential operation 746
14.1.16 EXPT/Exponentiation 747
14.1.17 LOG/Logarithmic operation 748
14.1.18 LN/Natural logarithm operation 749
14.1.19 LIMIT/Upper and lower limit control 750
14.1.20 MAX/Maximum value 752
14.1.21 MIN/Minimum value 753
14.1.22 AND/Logical AND 754
14.1.23 OR/Logical OR 755
14.1.24 XOR/XOR 756
14.1.25 NOT/NOT 757
14.1.26 SHL/Shift left instruction 758
14.1.27 SHR/Shift right instruction 759
14.1.28 ROL/Rotate left 760
14.1.29 ROR/Rotate right 761
14.1.30 SEL/Selection 762
14.1.31 MUX/Multiplexer 763
14.1.32 GT/Greater than 764
14.1.33 LT/Less than 765
14.1.34 GE/Greater than or equal to 766
14.1.35 LE/Less than or equal to 767
14.1.36 EQ/Comparison (Equivalent) 768
14.1.37 NE/Comparison (not Equivalent) 769
14.1.38 SIZEOF/Get the size 770
14.1.39 TRUNC/Truncated rounding instruction 770
14.1.40 MOVE/Move instruction 771
14.1.41 BOOL_TO_<TYPE>/BOOL type conversion instruction 772
14.1.42 BYTE_TO_<TYPE>/BYTE type conversion instruction 774
14.1.43 DATE_TO_<TYPE>/DATE type conversion instruction 776
14.1.44 DINT_TO_<TYPE>/DINT type conversion instruction 778
14.1.45 DT_TO_<TYPE>/DT type conversion instruction 780
14.1.46 DWORD_TO_<TYPE>/DWORD type conversion instruction 782
14.1.47 INT_TO_<TYPE>/INT type conversion instruction 784

14.1.48 WORD_TO_<TYPE>/WORD type conversion instruction 786
14.1.49 REAL_TO_<TYPE>/REAL type conversion instruction 788
14.1.50 SINT_TO_<TYPE>/SINT type conversion instruction 790
14.1.51 TIME_TO_<TYPE>/TIME type conversion instruction 792
14.1.52 TOD_TO_<TYPE>/TOD type conversion instruction 794
14.1.53 UDINT_TO_<TYPE>/UDINT type conversion instruction 796
14.1.54 UINT_TO_<TYPE>/UINT type conversion instruction 798
14.1.55 USINT_TO_<TYPE>/USINT type conversion instruction 800
14.1.56 LREAL_TO_<TYPE>/LREAL type conversion instruction 801
14.1.57 LWORD_TO_<TYPE>/LWORD type conversion instruction 803
14.1.58 LINT_TO_<TYPE>/LINT type conversion instruction 804
14.1.59 LTIME_TO_<TYPE>/LTIME Type Conversion Instruction 805
14.1.60 TO_<TYPE> Type conversion instruction 807
14.1.61 FLOOR/Round down the first decimal place instruction 809
14.1.62 CEIL/Round up the first decimal place instruction 809
14.1.63 ROUNDUP/ Round up instruction for the first digit after the decimal point 810
14.1.64 ROUND instruction 811
14.1.65 MOVEA/Batch transfer instruction 812
14.1.66 FIND/FIND string position 813
14.1.67 INSERT/string insertion 814
14.1.68 LEFT/fetch string from left 815
14.1.69 RIGHT/fetch string from right 816
14.1.70 DELETE/string deletion 817
14.1.71 REPLACE/String Replacement 818
14.1.72 MID/Fetch arbitrarily from a string 819
14.1.73 CONCAT/Concatenate two input strings and output them 820
14.1.74 LEN/get string length 821
14.1.75 STRING_TO_<TYPE>/STRING type conversion instruction 822
14.2 Built-in function block 825
14.2.1 R_TRIG/rising edge test 825
14.2.2 F_TRIG/falling Edge Test 826
14.2.3 RS/Bistable function block (Reset First) 827
14.2.4 SR/Bistable function block (Setting First) 828
14.2.5 CTU/Count Up 829
14.2.6 CTD/Count Down 831
14.2.7 CTUD/Count Up and Down 833
14.2.8 TON/ON Delay Timer 835
14.2.9 TOF/OFF Delay Timer 836
14.2.10 TP/Pulse Timer 837
14.2.11 RTC/Real Time Clock 838
14.2.12 SETSYSTIME/Set clock 839
14.2.13 SETSYSTIMERTC/Set clock RTC 840
14.2.14 SETSYSTIMEMS/Set clock MS 841

14.2.15 GETSYSTIME/Get clock 842
14.2.16 GETSYSTIMERTC/Get clock RTC 843
14.2.17 GETSYSTIMEMS/Get clock MS 844
14.2.18 PWM/Pulse width limit 845
15 ST Help manual 848
15.1 Introduction 848
15.2 Structured text program execution sequence 848
15.2.1 Program execution sequence 848
15.2.2 Expression execution order 848
15.2.3 Operators 849
15.3 Representation and declaration of variables 850
15.3.1 Variables 850
15.3.2 Identifiers 850
15.3.3 Variable declaration 850
15.4 Introduction to basic statements 851
15.4.1 Assignment statement 852
15.4.2 Functions and function block calls 852
15.4.3 Select statements 853

```
VAR
    nDevice1,nDevice2,nDevice3:BOOL;//The start signal of three device
    nState:BYTE;//The signal of current signal
END_VAR
```

```
CASE nState OF
    0,5:
    nDevice1:= TRUE;
    nDevice3:= FALSE;
    2:
    nDevice2:= FALSE;
    nDevice3:= TRUE;
    10..20:
    nDevice1:= TRUE;
    nDevice3:= TRUE;
ELSE
    nDevice1:= FALSE;
    nDevice2:= FALSE;
    nDevice3:= FALSE;
END_CASE
```

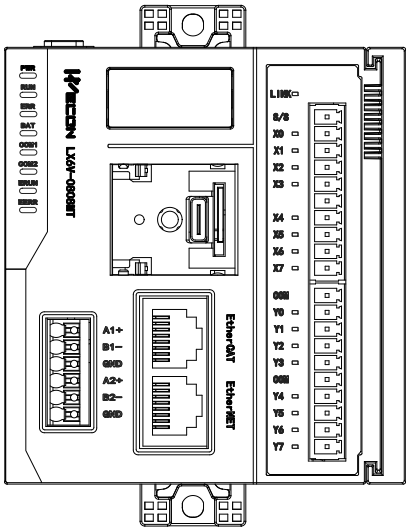
15.4.4 Iteration statements 856
15.4.5 Jump statement 859
15.4.6 RETURN statement 860
15.4.7 Empty statements 860
15.4.8 Comment statements 860
15.5 Introduction to the REF series instructions 861
15.6 ST Application instruction 863
15.6.1 OUT T/ Timer output 863
15.6.2OUT C/Counter output 865
15.6.3 OUT LC/ Long counter output 866

15.6.4 DI, EI/Disable Interrupt, Enable Interrupt 867
15.6.5 ZRST/Data batch reset 868
15.6.6 ZSET/Data batch set 870
15.6.7 RST/ Data reset 871
Appendixes 872
Appendix 1 Special Relay (SM) 872
Error message 872
System information 872
Clock Information 873
Scan information 873
Instruction related 873
Interrupt prohibited 876
High-speed input and output 876
Communication 878
List of Ethernet-related special devices 879
Appendix 2 Special register (SD) 880
Error message 880
System information 881
Clock information 882
Scan information 882
Instruction related 884
Interrupt prohibited 884
High-speed input and output 885
Right Expansion module 888
Input filter 888
Communication 888
List of Ethernet-related special devices 891
Log information 891
Appendix 3 Error codes sorting 893
PLC Hardware error 893
Ladder diagram execution error 893
PLC Parameter error 893
EtherCAT Configuration error 896
PLC Communication error 897
PLC Operation error 899
Appendix 4 ASCII code comparison table 908
ASCII Code comparison table 908

1 Execution of the program

1.1 Composition of the scan

The scan configuration of the CPU module is as follows.



1.1.1 Initialization

The initialization based on the status of the CPU module is as follows.

√: execute. **×**: not execute

Processing item	Status of the CPU module		
	When the power is ON	STOP	When STOP RUN
Initialization of input and output modules	√	×	×
CPU parameter check	√	×	×
Check of system parameters	√	×	×
Device initialization	√	×	√
Error clear	√	×	√

1.1.2 Input and output point refresh

Perform the following before starting program calculation.

Update the actual input point of the PLC to the input relay X.

The following is executed after the END instruction is executed.

Update the PLC output relay Y to the actual output point.

Key points

When performing a constant scan, the I/O refresh is performed after the waiting time of the constant scan.

1.1.3 Operation of the program

According to the program setting, the execution starts from step 0 of each program to the END instruction. This program is called the main program.

1.1.4 END processing

Perform the following processing.

① Completion processing of partial instructions

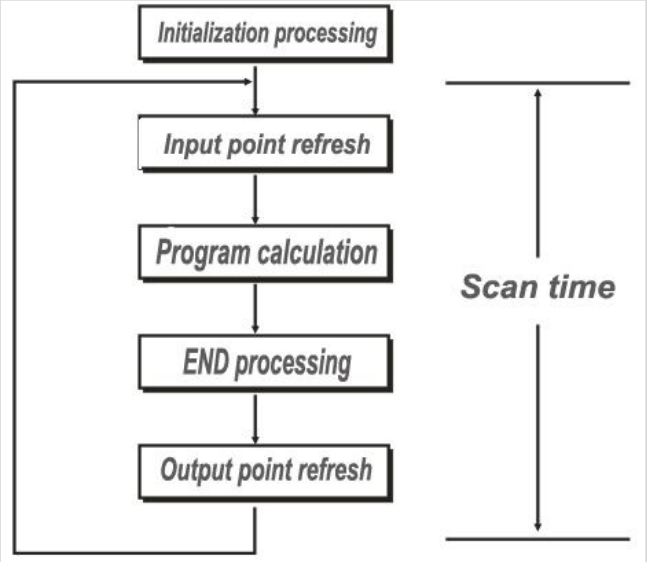
② Watchdog timer reset

③ Communication processing

④ Setting the value of special relay/special register (when the setting timing is END processing)

1.2 Scan time

The CPU module repeats the following processing, and the scan time is the total of the following processing and execution time.



The initial scan time indicates the time including this processing.

1.2.1 Initial scan time

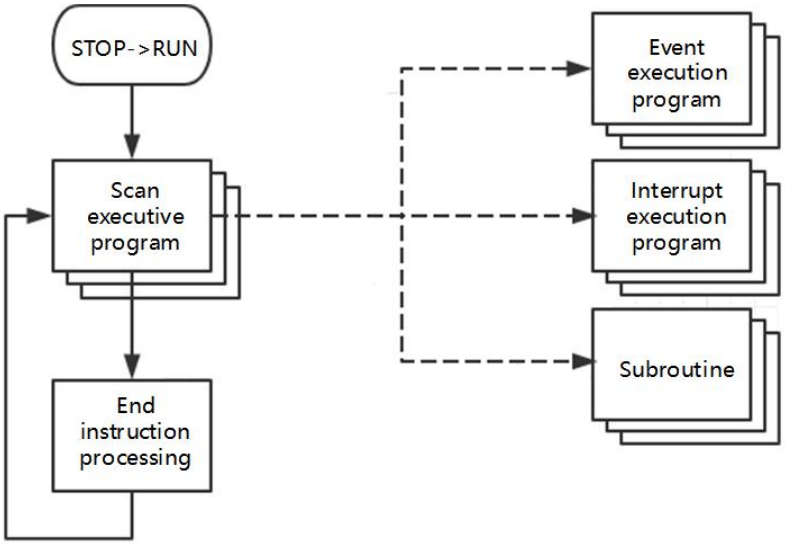
It is the first scan time of the CPU module in RUN.

Process as the following way:

The value stored in SD134 (initial scan time (ms unit)) and SD135 (initial scan time (s unit)).

1.3 The flow of each procedure

When the CPU module changes to the RUN state, the programs are executed in sequence according to the program execution type and execution sequence settings.



Key points

When the execution types of the programs are the same, they are executed in the order set in the execution order.

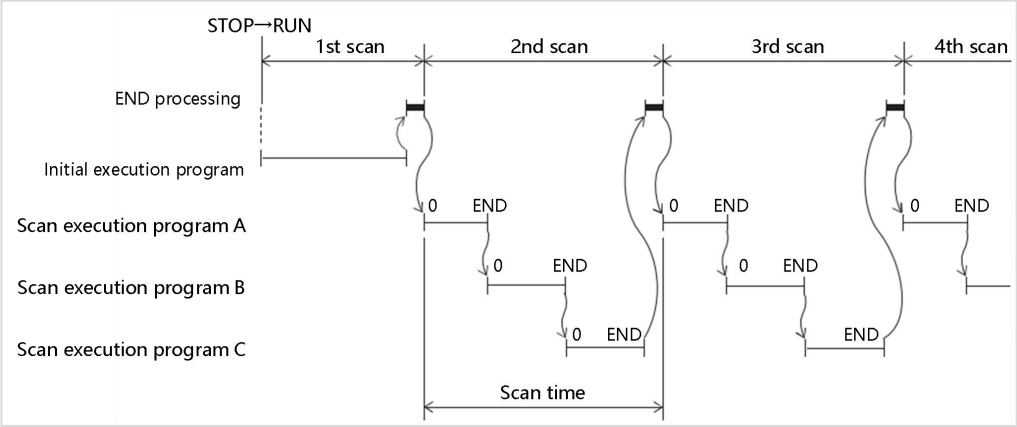
✎Note:

When executing instructions that can be completed with multiple cycles (such as OUT T, RAMP, RS, etc.), they should be programmed in the scan program. If it is used in event execution type programs and mid-stage execution type programs, these instructions may not be executed in multiple scan cycles, causing actual results to be different from the ideal results. Therefore, unless events, interrupts and subroutines can be executed in each scan cycle. It is not recommended to use multi-cycle execution instructions in other situations.

1.4 Types of program execution

1.4.1 Scan execution program

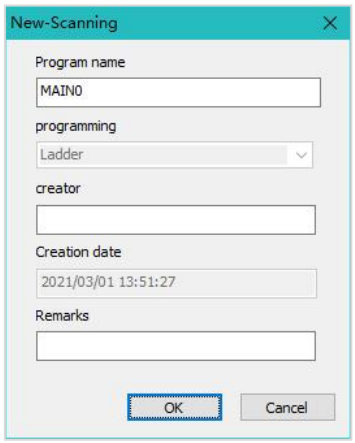
Each scan is executed only once from the next scan of the initial execution type program.



When multiple scan execution type programs are executed, the execution time of the scan execution type program is the time until all scan execution type programs are executed. In addition, before the execution of the scan execution type program is completed, if an interrupt program/event execution type program/subroutine is executed, the execution time will also be included.

Creation of multiple scanners

"Project Management"→ "Program"→ "Scan"→ Right click to create

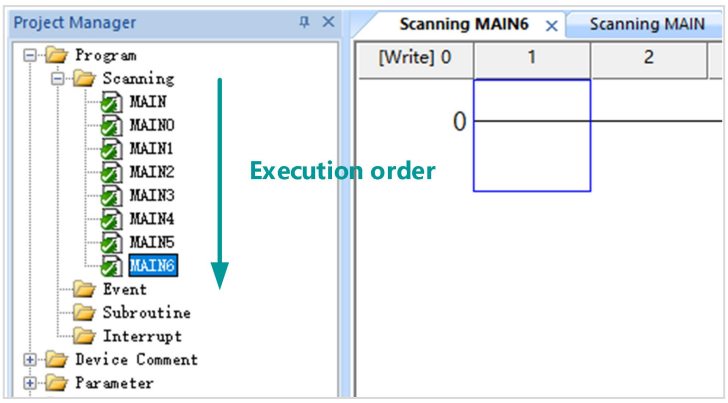


① Scan the program name: the program name requires to match case, and the program name cannot use the same name with device name (the device name does not match case).

② The input of /%$@&=~`^<>?:{}[],;!*.\\'" is not supported. It cannot exceed 64 characters. The default name is MAINx.

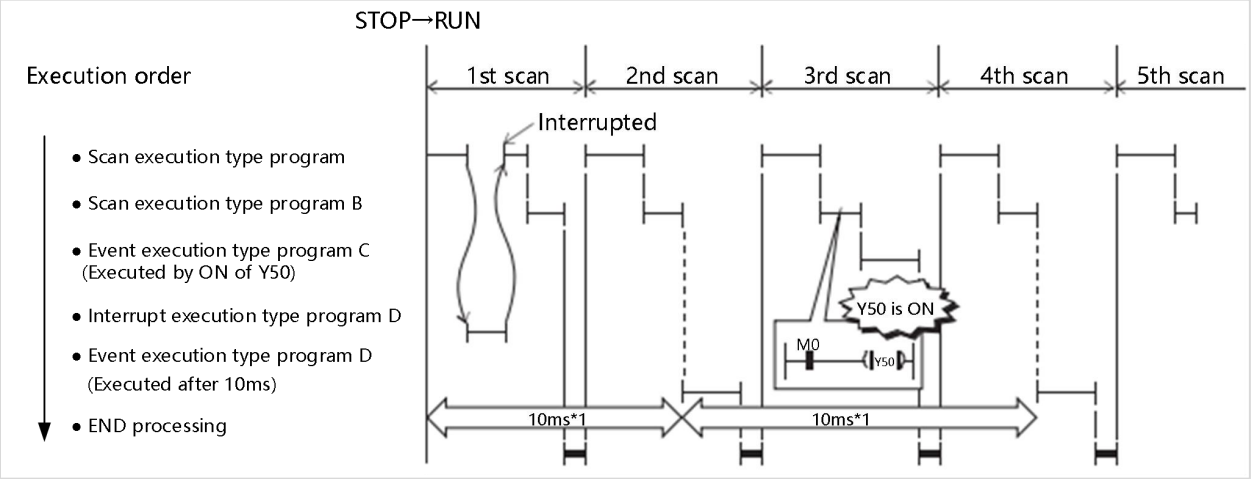
③ The number of scan programs that can be built is limited to 100.

④ The execution sequence runs from top to bottom in the order of creation.



1.4.2 Event execution program

It uses the event specified by the user as a condition to trigger the program to start execution.



Trigger type

The trigger of event execution type program is as follows.

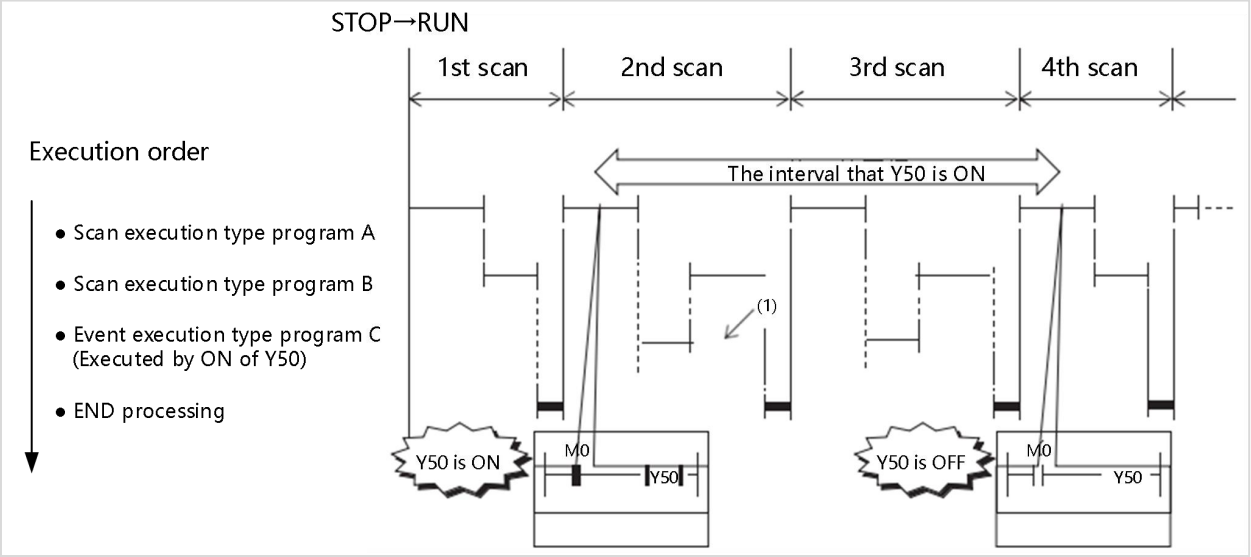
(1) ON event of bit data (TRUE)

① After the ON event is specified, if the contact that sets the trigger condition in the ON event is turned ON during the scan program, the ON event program will be executed in the scan program page*1 or before the END instruction is executed.

② The ON event program will only be executed once in a single scan cycle.

③ After the ON event is executed, you can set whether to clear the current value of the output (Y) and timer (T) used in the program.

*1: Scan program paging: multiple scan programs are established, and each scan program is called a paging. After scan program A is executed, before scan program B is executed, it will be judged whether an event program needs to be executed.



When it is the turn of the execution sequence of event execution type program C and Y50 is ON, the program is executed.

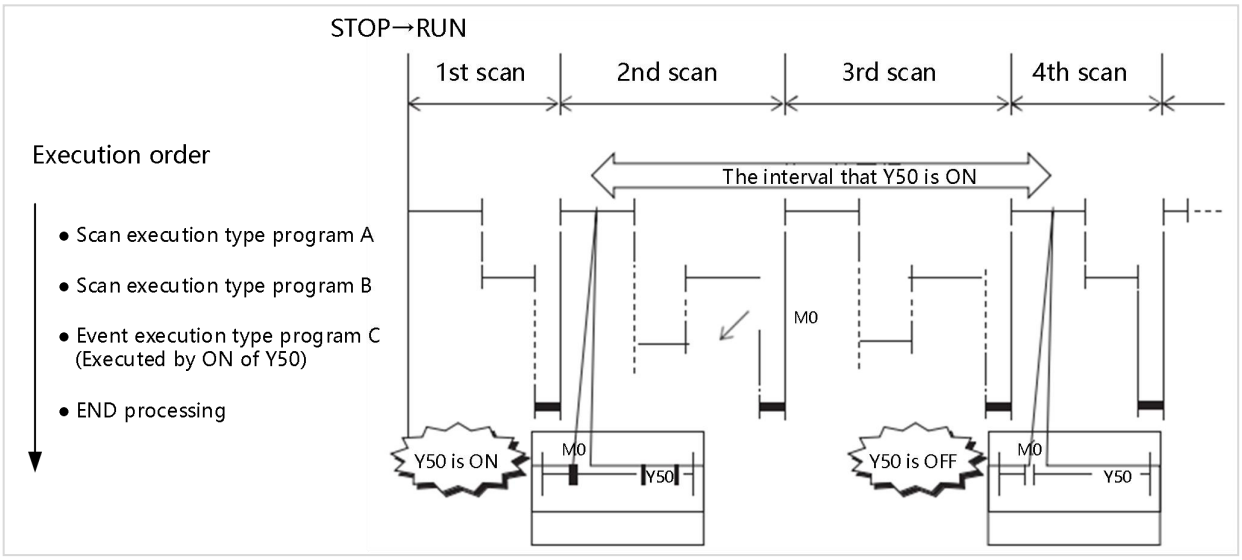
The devices that can be specified are as follows.

Project		Content
Device *1	Bit Device	X, Y, M, SM
	Bit specification of word device	D.b

*1 The indexed device cannot be specified.

(2) TIME event

After the program is to RUN state and the specified time has elapsed, event is executed one time when it comes to the execution sequence of the first corresponding program. For the second and subsequent executions, the time is re-measured from the start of the last event execution type program. After the specified time has elapsed, the program is executed repeatedly when it comes to the execution sequence of the first corresponding program. In addition, in the next scan after the corresponding program is executed, the current value of the output (Y) and timer (T) used in the corresponding program can be cleared. It can be used for programs that do not need to respond in a fixed period of time.



After the specified time has elapsed, when it comes to the first execution sequence, the event execution type program C is executed.

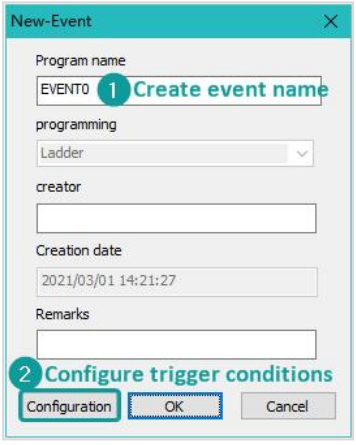
Key points

When set to clear the current value of output and timer, and the scan time is longer than the set value of elapsed time, the current value of output and timer will not be cleared.

Operational steps

(1) New event

Project management→ Program→ Event → Right click to create.



Event program name:

① The program name requires to match case, and the program name with the same name as the device cannot be used (the device name does not match case).

② The program name does not support /%$@&=~`^<>?:{}[],;!*.\\'" character input.

③ The length of the program name cannot exceed 64 characters. The default name is EVENTx.

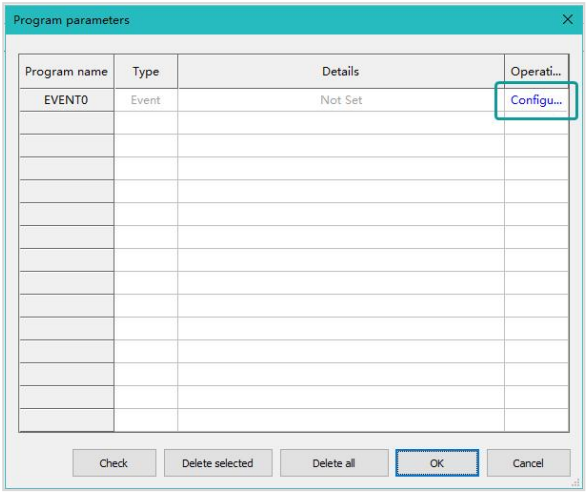
A maximum of 100 new event programs could be created.

(2) Execution type

There are two ways to configure the event execution type:

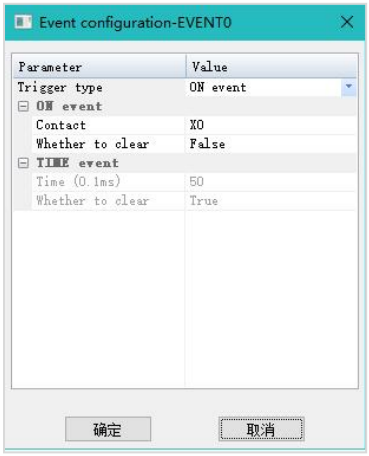
1) Configure when creating a new event program, as shown in the figure above.w

2) Project management→ Program→ Parameter → Program parameter→Configuration



Configuration instructions:

1) Configuration interface:



2) Parameter content:

Project		Content	Setting range	Default
Execution type		Select event type	Not set/ON event/TIME event	Not set
ON event	Contact	The event type can be set when ON event is selected. Set the bit device as the trigger condition.	X/Y/M/SM/D.b	
	Whether to clear	When the bit device of the trigger condition set by the ON event is turned OFF, whether to clear the current value of the output (Y) and timer (T) used in the execution program of the ON event in the next cycle.	True False	False
TIME event	Time	Set how long to execute the event program once.	1 to 2147483647 (100us unit)	
	Whether to clear	When the TIME event is executed, if the event is not executed in the next scan cycle, select whether to clear the output (Y) used in the TIME event execution program and the current value of the timer (T).	True False	False

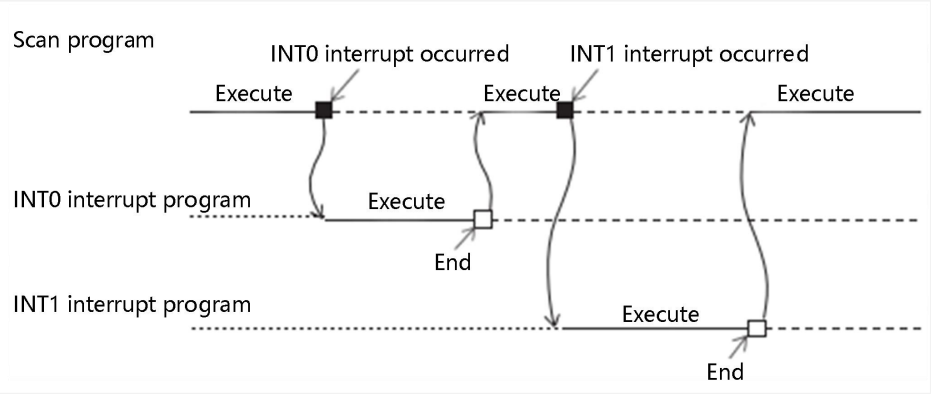
Key points

When "ON event" or "TIME event" is specified, if "Clear or not" is set to "Clear", the event program will not be executed in one scan cycle, and all the internal outputs (Y) and current value of timer (T) will all be cleared (except for the cumulative type and subroutine type T). If the time set by the TIME event is less than the scan period, it is equivalent to executing the TIME event every scan period. Even if the clear output is set, the output and timer data in the event program will not be cleared.

1.4.3 Interrupt execution program

In the process of executing the scan program, the program that can interrupt the priority execution of the scan program is called an interrupt execution program.

- When an interrupt cause occurs, the interrupt program corresponding to the interrupt pointer number will be executed. However, the execution needs to be set to the interrupt enabled state by the EI instruction.



- An interrupt name corresponds to an interrupt program, and the interrupt name cannot be repeated. Each interrupt has its own trigger condition and execution program, and each interrupt program ends with END.
- Interrupt has the characteristic of interrupting the original execution program and executing the interrupt first, but it cannot interrupt the interrupt program being executed.
- The interrupt program has the concept of priority. The smaller the priority value, the more priority the response. The priority setting range is 0 to 2.

The actions when an interruption cause occurs are as follows:

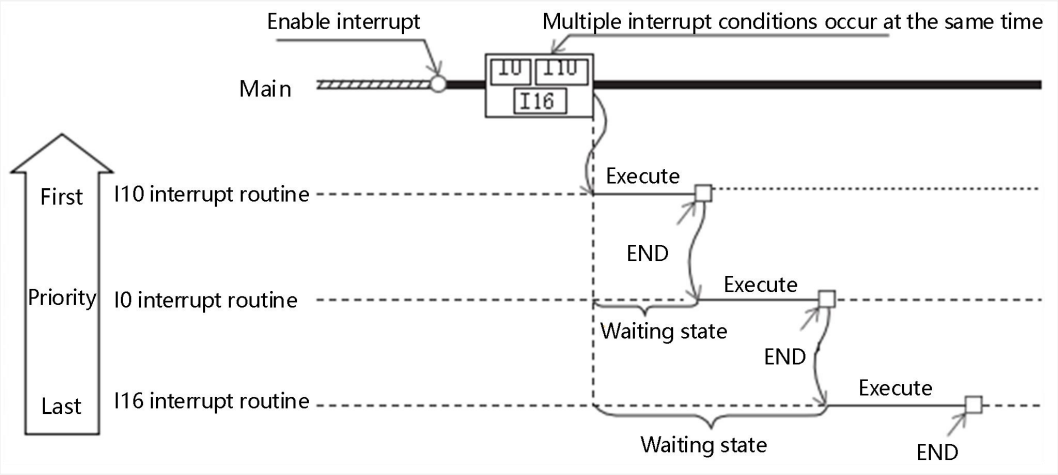
1) Interrupt prohibition (DI) When an interruption cause occurs.

If the interrupt execution condition is triggered in a program that is forbidden by DI, the interrupt will not be executed. Even if the subsequent program uses the EI instruction to allow interruption, the previously shielded interrupt program will not be executed. Only the interrupt execution condition is triggered again. The interrupt program will be executed.

2) When multiple interrupt causes occur simultaneously in the interrupt enabled state.

The interrupt program with higher priority will be executed sequentially. In addition, when multiple interrupts with the same priority occur at the same time, the actions are executed in the order of interrupt priority.

If three interrupt programs I0, I10, I16 are created, the priority of I0 is 1, the priority of I10 is 0, and the priority of I16 is 1. The execution logic is shown in the figure below: I10 has the smallest priority and is executed first; I0 and I16 have the same priority and are executed in the order of program establishment.



3) When an interrupt occurs during the waiting time when performing constant scan.

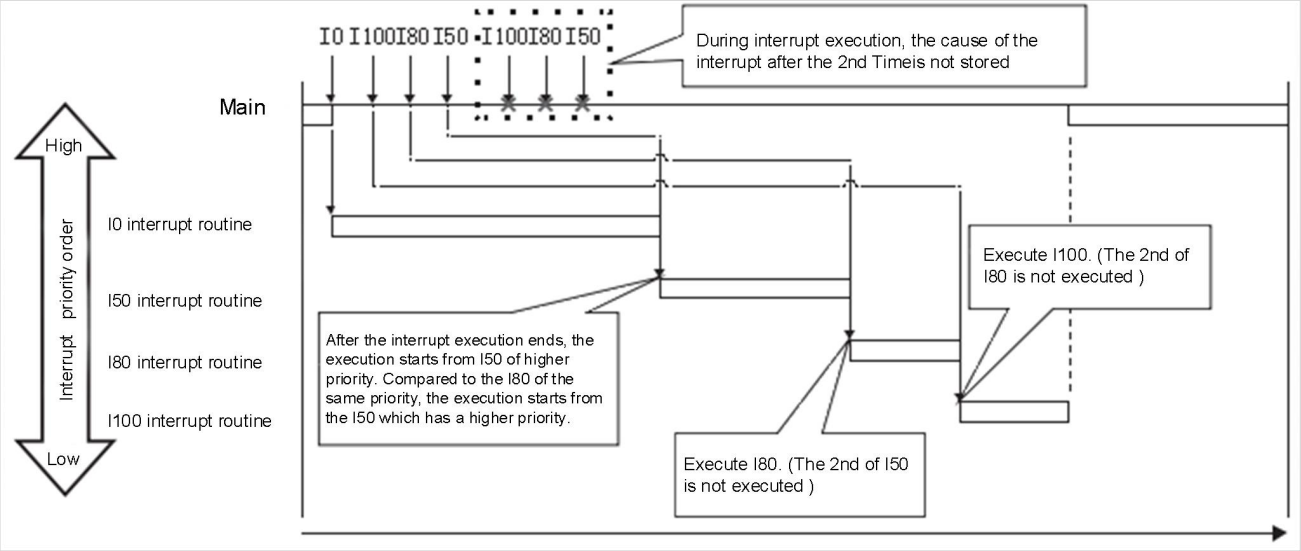
Execute the interrupt program for this interrupt.

4) When other interrupts occur during the execution of the interrupt program.

In the interrupt program (including the specification when the interrupt occurs in the event execution program), when other interrupts occur, the original interrupt execution program will not be interrupted. After the original interrupt execution program is completed, the new interrupt program is executed. After the execution is completed Then return to the scanning procedure.

5) During the execution of the interrupt program, when an interrupt cause with a low priority or the same priority occurs.

The interruption cause that occurred is stored, and after the interrupt program in execution ends, the interrupt program corresponding to the stored interruption cause is executed. Even if the same interruption cause occurs multiple times, the interruption cause is stored only once.



6) When the same interruption cause occurs during the execution of the interrupt program;

The interruption cause that occurred is stored, and after the interrupt program in execution ends, the interrupt program corresponding to the stored interruption cause is executed. Even if the same interruption cause occurs multiple times, the interruption cause is stored only once.

Interrupt trigger condition classification

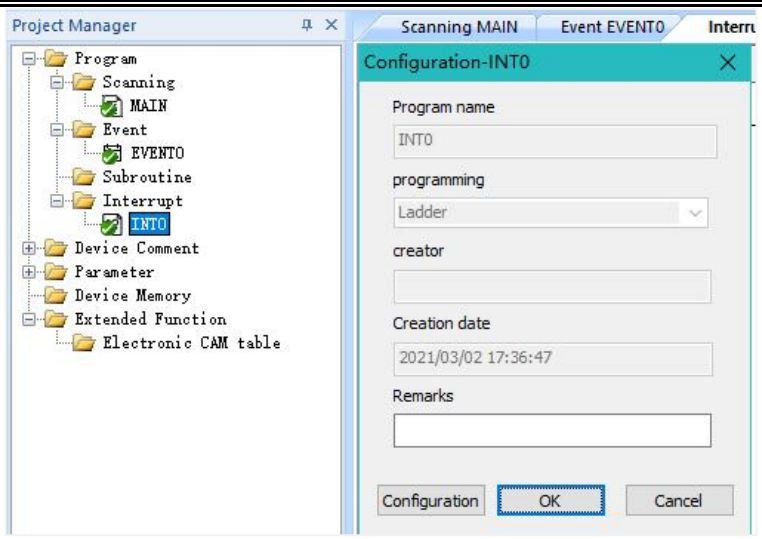
(1) External input (X) interrupt

1) Description of external input interrupt

- The external input interrupt is triggered by the rising or falling edge of the fixed X point input.
- Supports the rising and falling edge interrupts of a total of 8 input points of X0 to X7, and supports a total of 16 external input interrupts.
- The same interrupt trigger condition cannot create multiple interrupt programs.
- External input interrupt and high-speed counter cannot use the same X point.
- You must use EI in the scan program to allow interrupts before the interrupt execution program will be executed.

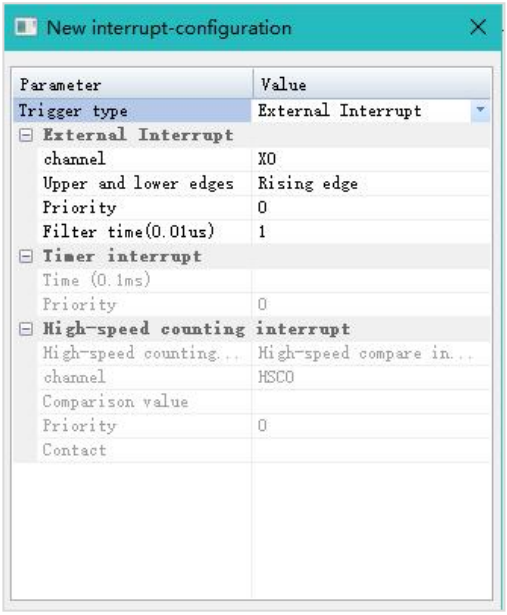
2) External input interrupt steps.

Project management→ Program→ Interrupt→ right click to create



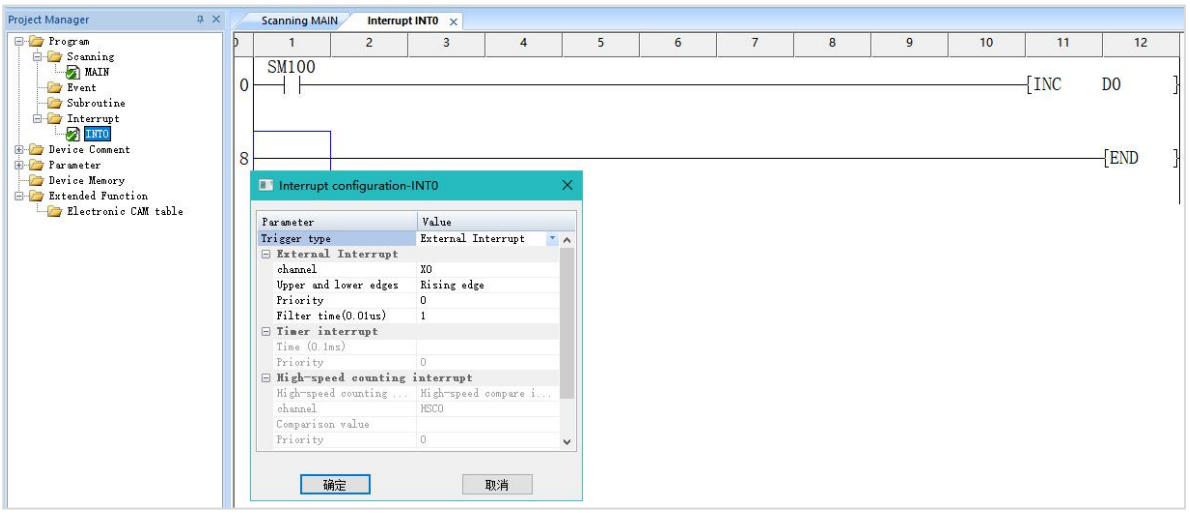
- The interrupt program name requires to match case, and the program name with the same name as the device cannot be used (the device name does not match case),
- The interrupt program name does not support the input of /%$@&=~`^<>?:{}[],;!*.\\'" characters,
- The length of the interrupt program name cannot exceed 64 characters and cannot be typed. The default name is INTx.

Click Configure, and select external interrupt for execution type, as shown in the figure below (it can also be configured in "program parameters" in "parameters" in project management):

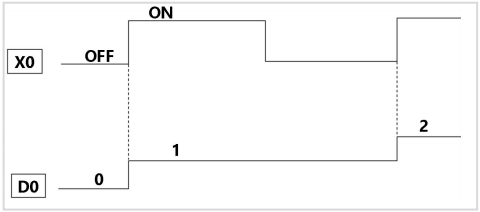


Project		Content	Setting range	Default
Execution type		Select the type of interrupt	Not set, External input interrupt, Timer event, high-speed counter interrupt	Not set
External input interrupt	channel	Select the channel for external input interrupt	X0 to X7	X0
	Trigger edge type	Choose to trigger on rising edge or falling edge	Rising edge; Falling edge	Rising edge
	priority	When multiple interrupts arrive at the same time, the order of priority execution, the smallest value is executed first	0 to 2	0
	Filter time (0.01us)	Set the filter time of X point, the unit is 0.01us. Note: X rising edge interrupt and X falling edge interrupt use the same X filter, so after the filter setting is changed in the X rising edge configuration, the X falling edge will also change. If the filter time is set to 1000, you must ensure that the high level and low level of the input signal are maintained for more than 10 us before the interrupt can be triggered.	0 to 1700	1

3) Write interrupt execution program



Double-click the newly created interrupt program in the project management to start writing the interrupt execution program. As shown in the figure above, a newly created interrupt program is INT_X0_UP, and the trigger condition is configured to execute the interrupt program when the X0 rising edge is configured. If the EI instruction is used in the main program to allow interrupts, all programs in INT_X0_UP will be executed whenever X0 changes from OFF to ON , That is, D0 will increment once.



(2) Timer interrupt

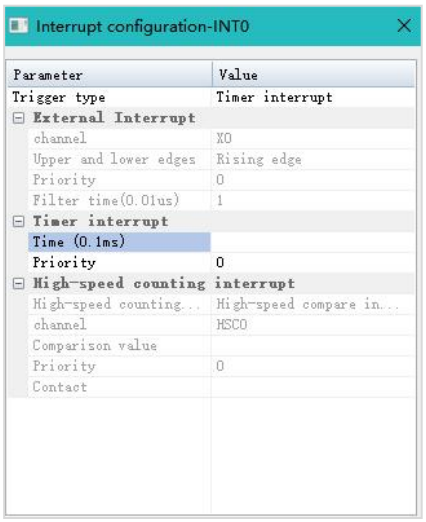
1) Timer interrupt description

- Timer interrupt is based on the set time, execute the interrupt program every this time, the minimum time interval can reach 100us.

- Up to 100 timer interrupt execution programs can be created.
- Each timer interrupt program is independent of each other and does not affect each other.
- Each timer interrupt program should be configured with priority. When triggered at the same time, it is executed in the order of priority, but when the priority is the same, it is executed in the order of the established program.
- The interrupt execution program is executed only after EI is used in the scanner to allow the interrupt

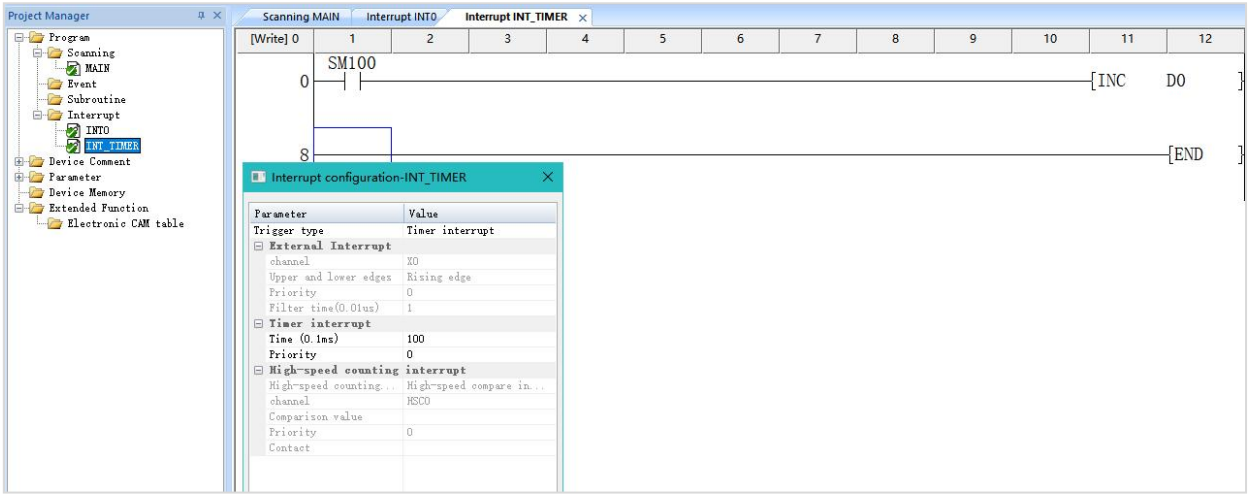
2) Timer interrupt step

Project management ⇨ Program ⇨ Interrupt ⇨ Right click to create. Enter the program name. The program name only supports the combination of English letters, numbers, and underscores, and must start with an English letter. The default is INTx. Click Configure and select Timer Interrupt as the execution type, as shown in the figure below (it can also be configured in "Program parameters" in "Parameter" in project management).



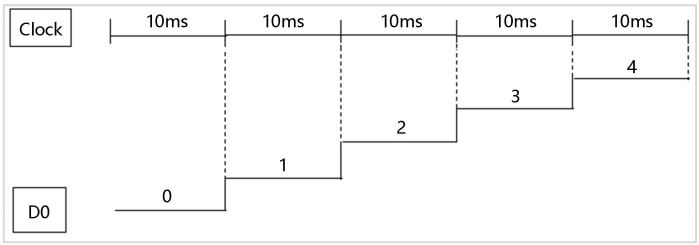
Project		Content	Setting range	Default
Execution type		Select the type of interrupt	Not set/External input interrupt/Timer event/high-speed counter interrupt	Not set
Timer interrupt	Time	Set the interval time for interrupt triggering	1 to 2147483647 (100us unit)	
	Priority	When multiple interrupts arrive at the same time, the order of priority execution, the smallest value is executed first	0 to 2	0

3) Write interrupt execution program



Double-click the newly created timer interrupt program in the project management to start writing the interrupt execution program. As shown in the figure above, a newly created timer interrupt program is INT0, and the trigger condition is configured to execute the interrupt program every 10ms. If the main program uses EI to enable interrupts, all instruction programs in INT0 will be executed

every 10ms, namely D0 It will add 1 to 10ms.



(3) High-speed counter interrupt

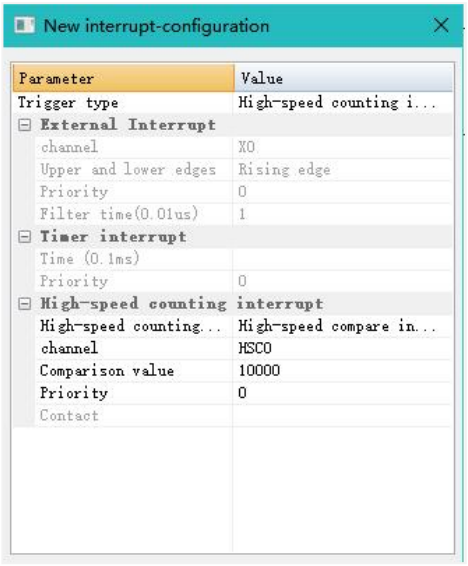
1) Description of high-speed counter interrupt

- The high-speed counter interrupt triggers an interrupt condition after the set value of the high-speed counter HSC0 to HSC7 provided by the PLC and executes the interrupt program.
- It can support up to 100 high-speed counter interrupt programs, but the number that can be supported by each channel does not need to be fixed.
- When using the high-speed counter interrupt, project must configure the high-speed counter and use the OUT HSC instruction to enable the corresponding counting channel to count before it can be used (see the high-speed counter description section for the specific configuration method).
- Each high-speed counter interrupt program should be configured with priority. When triggered at the same time, it will be executed in the order of priority. When the priority is the same, it will be executed in the order of channels HSC0-HSC7. When the channel is also the same, it is executed in order according to the creation promise.
- Project must use EI in the scan program to allow interrupts before the interrupt execution program will be executed.

✎**Note:** Both the HSC channel and the external input interrupt channel must use the PLC input point X. It should be noted that it cannot be reused during configuration. For details, please refer to the configuration chapter of the high-speed counter.

2) high-speed counter interrupt steps

"Project management"⇨"Program⇨"Interrupt"⇨Right click to create. Enter the program name. The program name only supports the combination of English letters, numbers, and underscores, and must start with an English letter. The default is INTx. Click Configure, select high-speed interrupt for execution type, as shown in the figure below (it can also be configured in "Program parameters" in "Parameters" in project management).



Project		Content	Setting range	Default
Execution type		Select the type of interrupt	Not set, External input interrupt, Timer event, high-speed counter interrupt	Not set
High count interrupt	Mode	Select the type of high-speed counter interrupt: (1) High-speed comparison interrupt: The interrupt program is executed after the trigger condition is reached. (2) High-speed comparison setting: After reaching the trigger condition, the set contact is set. (3) High-speed comparison reset: reset the set contact after reaching the trigger condition.	High-speed compare interrupt High-speed comparison set High-speed comparison reset Not set	High-speed compare interrupt
	Channel	Select the high-speed counter channel used	HSC0 to HSC7	HSC0
	Comparison value	Set the comparison value of the high-speed counter. When the high-speed counter value of the set channel passes this value, the trigger condition is reached.	-2147483648 to 2147483647	
	Priority	When multiple interrupts arrive at the same time, the order of priority execution, the smallest value is executed first	0 to 2	0
	Contact	It is available when selecting high-speed comparison set and high-speed comparison reset. The contact is set or reset immediately after the trigger condition is reached.	Y/M/D.b	

3) Description of triggering rules

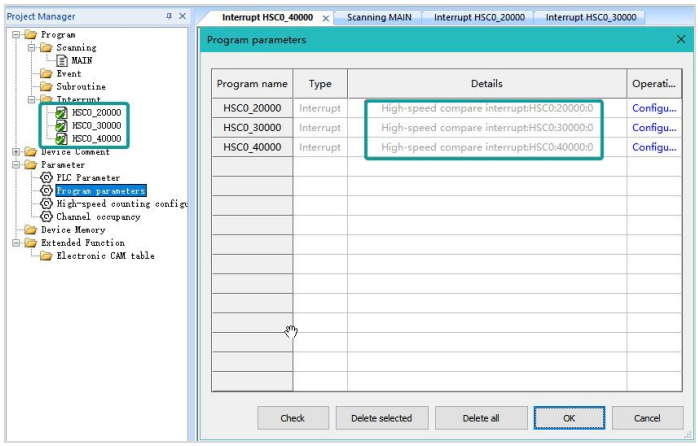
Mode	Configuration	The current value	Action
High-speed compare interrupt (INT0)	Comparison value: 10000	9999 → 10000	Execute all programs in interrupt INT0
		10001 → 10000	Execute all programs in interrupt INT0
High-speed comparison set (INT1)	Comparison value: -50,000 Contact: Y10	-50001 → -50000	Y10 is immediately set and mapped to the actual output (not affected by the scan period) The program in INT1 will not be executed
		-49999 → -50000	Y10 is immediately set and mapped to the actual output (not affected by the scan period) The program in INT1 will not be executed
High-speed comparison reset (INT2)	Comparison value: 400000 Contact: Y10	399999 → 400000	Y10 is reset immediately and mapped to the actual output (not affected by the scan period) The program in INT2 will not be executed
		400001 → 400000	Y10 is reset immediately and mapped to the actual output (not affected by the scan period) The program in INT2 will not be executed

✎**Note:** Both HSC channel and external input interrupt channel need to use the INPUT point X, so it should be noted that it cannot be reused in configuration. For details, please refer to the configuration section of high-speed counter.

4) Write interrupt execution program

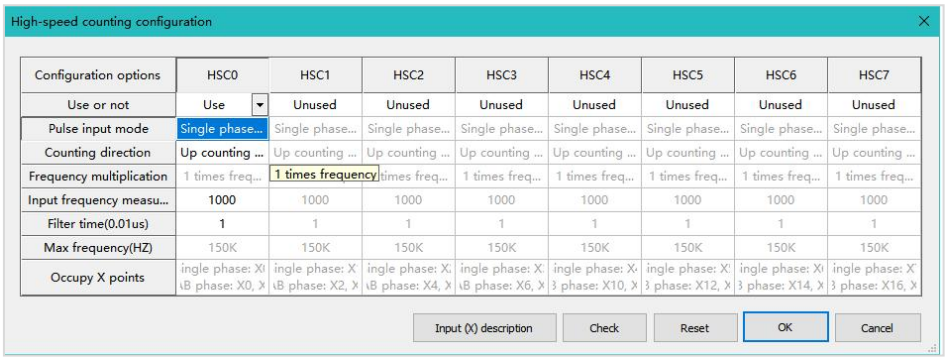
① New interrupt program

Create three new interrupt programs under the interrupt of project management, namely HSC0_20000, HSC0_30000, HSC0_40000. Configure the interrupt program in the "program parameters", as shown in the figure below.

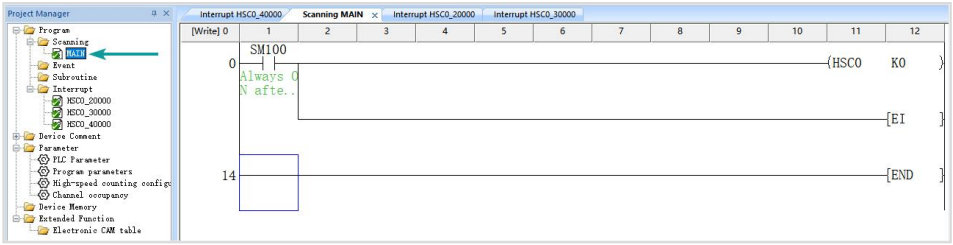


② High-speed counter configuration

Configure HSC0 for use in the high-speed counter configuration. After selecting the working mode, click the "Check" button. After the correct configuration box pops up, click Enter.



Call the high-speed counter in the main program and enable interrupts:



Program operation:

Assuming that the High-speed counter channel 0 has been receiving pulses:

When the count value of HSC0 accumulates from 0 to 20000, all procedures of HSC0_20000 are executed.

When the count value of HSC0 is accumulated from 20000 to 30000, all procedures of HSC0_30000 are executed.

When the count value of HSC0 is accumulated from 30000 to 40000, all procedures of HSC0_40000 are executed.

Mask interrupt

(1) Mask through application instructions

The PLC interrupt is in the shielded state by default when it is powered on, and can only be used after the interrupt is allowed through the EI instruction.

The interrupt mask instruction DI masks all interrupts without parameters, and masks some priority interrupts with parameters (refer to the program flow instruction DI/EI for details).

(2) Mask through special registers SM and SD

1) External input interrupt mask register

External input interrupt mask register			
Special register number	**Type of interrupt**	**Instruction**	**Defaults**
SM352	X0 rising edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM353	X0 falling edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM354	X1 rising edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM355	X1 falling edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM356	X2 rising edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM357	X2 falling edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM358	X3 rising edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM359	X3 falling edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM360	X4 rising edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM361	X4 falling edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM362	X5 rising edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM363	X5 falling edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM364	X6 rising edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM365	X6 falling edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM366	X7 rising edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF
SM367	X7 falling edge interrupt	ON: shield interrupts; OFF: interrupt allowed	OFF

2) Timer interrupt mask register

Timer interrupt mask register			
Special register number	**Type of interrupt**	**Instruction**	**Default**
SD350	1st to 16th timer interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD351	17th to 32th timer interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD352	33th to 48th timer interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD353	49th to 64th timer interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD354	65th to 80th timer interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD355	81st to 96th timer interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD356	97th to 100th timer interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0

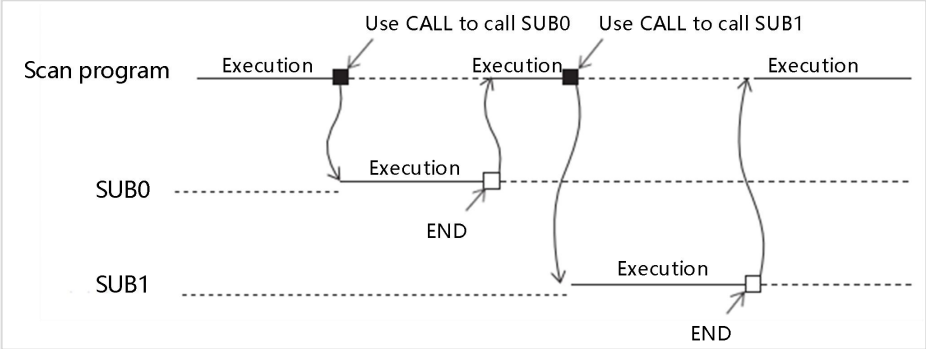
3) high-speed counter interrupt mask register

High-speed counter interrupt mask register			
Special register number	**Type of interrupt**	**Instruction**	**Default**
SD382	1st to 16th high-speed counter interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD383	17th to 32nd high-speed counter interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD384	33th to 48th high-speed counter interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD385	49th to 64th high-speed counter interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD386	65th to 80th high-speed counter interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD387	81st to 96th high-speed counter interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0
SD388	97th to 100th high-speed counter interrupt	Each bit can control the mask of an interrupt. ON: shield interrupts; OFF: interrupt allowed	0

1.4.4 Subroutine

During the execution of the scan program, the executed program can be called by the CALL instruction. You can create up to 100 new subprograms.

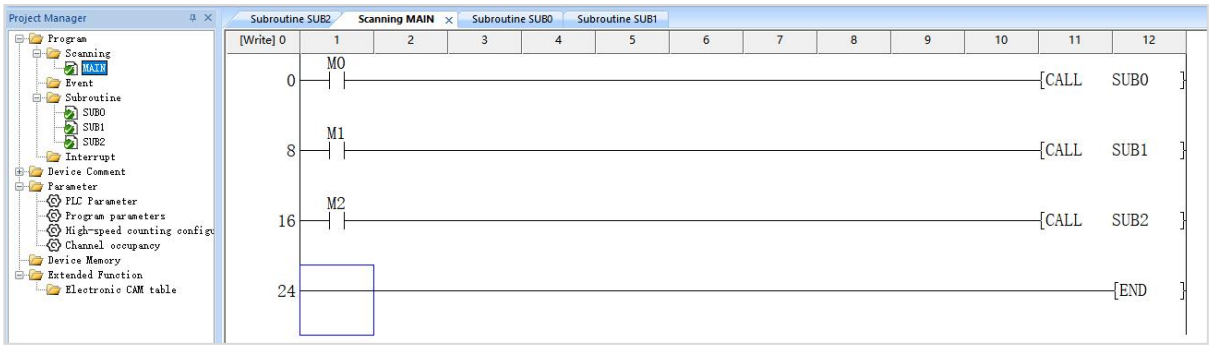
A subroutine is to split a certain module in the main program for the main program to call, which is conducive to the modularization of the program. Such as other high-level language functions, but this function has no parameters and no return value.



(1) Instructions for calling subroutines

After a new subroutine is created, the content of the program is not executed. It is executed only when the CALL(P) instruction is used to call the subroutine in the scan, event, and interrupt programs, and the call is executed once. Three new subroutines SUB0, SUB1, SUB2 are created as shown in the figure below. In the main program MAIN, the subprogram can be called by using the CALL(P) subprogram program name.

Through this programming method, the use of the same logic program for different conditions can reduce the number of ladder diagram steps and improve the readability of the ladder diagram.



✎Note:

① When using the timer (OUT T), note that the output will not be reset when the subroutine is not called, and a specific subroutine register must be used.

② It is not allowed to call recursively between subprograms, that is, call SUB1 in SUB0, and then call SUB0 in SUB1. This is not allowed.

③ The subroutine can be nested up to 32 levels. If the level exceeds 32 levels, a serious error will be reported and the ladder diagram operation will be forcibly stopped.

1.4.5 Positioning instructions

(1) Event

1) ON event

If the high-speed pulse instruction is turned on during the ON event, the high-speed pulse instruction will be sent as normal. If the ON contact of the trigger event in the scan period is turned OFF during the pulse sending, select whether to continue sending the pulse or stop the pulse according to the unscanned processing sign bit.

Output axis	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
Not scanned sign bit	SM899	SM959	SM1019	SM1079	SM1139	SM1199	SM1259	SM1319

When the sign bit is 0 (continue to send pulse), if the instruction is not scanned in the current scan cycle, continue to send pulses until it stops. At this time, it should be noted that if the trigger event OFF contact turns ON after the pulse is sent, the pulse will be sent again.

When the sign bit is 1 (stop sending pulses), if the trigger event ON contact turns OFF in a certain scan period, it will decelerate and stop.

2) TIME event

If the high-speed pulse instruction is turned on in the TIME event, the high-speed pulse instruction will be sent as normal. If the instruction is not scanned in a certain scan period during the pulse transmission, select whether to continue sending the pulse or stop the pulse according to the non-scanned sign bit.

Output axis	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
Not scanned sign bit	SM899	SM959	SM1019	SM1079	SM1139	SM1199	SM1259	SM1319

When the sign bit is 0 (continue to send pulse), if the instruction is not scanned in the current scan cycle, the pulse will continue to be sent until it stops. In the TIME event, it is impossible to ensure that the instruction is scanned in every scan cycle, so you should avoid using high-speed pulse instructions in the TIME time, otherwise the pulse will be sent again after the pulse is sent.

When the sign bit is 1 (stop sending pulses), if the instruction is not scanned in the current scan cycle, it will decelerate and stop. In the TIME event, if the sign bit is set to 1 (stop sending pulses), there will be no pulse sending.

(2) Subroutine

If the high-speed pulse instruction is turned on in the subroutine, the high-speed pulse instruction will be sent as normal. If the scanning period is closed during pulse sending, select whether to continue sending or stop the pulse according to the non-scanned sign bit.

Output axis	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
Not scanned sign bit	SM899	SM959	SM1019	SM1079	SM1139	SM1199	SM1259	SM1319

When the sign bit is 0 (continue to send pulse), if the instruction is not scanned in the current scan cycle, the pulse will continue to be sent until it stops. At this time, it should be noted that if the subroutine is called again after the pulse is sent, the pulse will be sent again.

When the sign bit is 1 (stop sending pulses), if the subroutine is closed during high-speed pulse sending, the speed will decelerate and stop. If the subroutine is closed before sending the pulse, then no pulse is sent.

(3) Interrupt

1) External interrupt

If the high-speed pulse instruction is enabled in the external interrupt, the high-speed pulse instruction selects whether the pulse continues to be sent or the pulse stops according to the non-scanned sign bit.

Output axis	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
Not scanned sign bit	SM899	SM959	SM1019	SM1079	SM1139	SM1199	SM1259	SM1319

When the sign bit is 0 (continue to send pulse), continue to send high-speed pulses until it stops.

When the sign bit is 1 (stop sending pulse), the high-speed pulse decelerates and stops.

2) Timer interrupt

If the high-speed pulse instruction is turned on in the timer interruption, the high-speed pulse instruction is sent as normal. If the instruction is not scanned in a certain scan period in the pulse transmission, the pulse continues to be sent or the pulse stops is selected according to the non-scanned sign bit.

Output axis	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
Not scanned sign bit	SM899	SM959	SM1019	SM1079	SM1139	SM1199	SM1259	SM1319

When the sign bit is 0 (continue to send pulse), if the instruction is not scanned in the current scan cycle, continue to send pulses until it stops. In the timer interrupt, it is impossible to ensure that the instruction is scanned in every scan cycle, so it is necessary to avoid using high-speed pulse instructions in the T timer interrupt. Otherwise, after the pulse transmission is completed, the pulse will be sent again.

When the sign bit is 1 (stop sending pulses), if the instruction is not scanned in the current scan cycle, it will decelerate and stop. In the TIME event, if the sign bit is set to 1 (stop sending pulses), there will be no pulse sending.

3) High-speed comparison interrupt

If the high-speed pulse instruction is enabled in the high-speed comparison interrupt, the high-speed pulse instruction selects whether the pulse continues to be sent or the pulse stops according to the non-scanned sign bit.

Output axis	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
Not scanned sign bit	SM899	SM959	SM1019	SM1079	SM1139	SM1199	SM1259	SM1319

When the sign bit is 0 (continue to send pulse), continue to send high-speed pulses until it stops.

When the sign bit is 1 (stop sending pulse), the high-speed pulse decelerates and stops.

2 Description of device

Device list

Classification	Type	Device name	Sign	Range	Mark
User device	Bit	Input	X	0 to 1777	Octal number
	Bit	Output	Y	0 to 1777	Octal number
	Bit	Internal relay	M	0 to 7999	Decimal number
	Bit	Step relay	S	0 to 4095	Decimal number
	Bit/word	Timer	T	0 to 511	Decimal number
	Bit/word	Counter	C	0 to 255	Decimal number
	Bit/double word	Long counter	LC	0 to 255	Decimal number
	Bit/double word	High-speed counter	HSC	0 to 15	Decimal number
	Word	Data Register	D	0 to 7999	Decimal number
	Word	Data Register	R	0 to 29999	Decimal number
System device	Bit	Special	SM	0 to 4095	Decimal number
	Word	Special register	SD	0 to 4095	Decimal number
Index register	Word	Index register	[D]	0 to 7999	Decimal number
	Word	Index register	V	0 to 7	Decimal number
	Double word	Long index register	Z	0 to 7	Decimal number
Nested	Bit	Nested	N	0 to 7	Decimal number
Pointer	-	Pointer	P	0 to 4095	Decimal number
Constant	-	Decimal constant	K	-	Decimal number
	-	Hexadecimal constant	H	-	Hexadecimal number
	Single precision floating point	Real constant	E	-	-

2.1 User device

2.1.1 Input relay (X)

The input relay represents the original PLC external input signal status, and the external signal status is detected through the input X port. 0 represents the external signal is open, and 1 represents the external signal is closed.

Using the program instruction method, the state of the input relay cannot be modified, and its node signals (normally open, normally closed) can be used indefinitely in the user program.

The relay signal is identified by Signs such as X0, X1,.....X7, X10, X11..... and its serial number is numbered in octal.

When an expansion module is connected, the extended X point will also use the X point as the component of the input signal state, and the occupied X point is the starting position of the X point used by the PLC with 0 as the end of the X point, such as PLC Occupy 17 to 24 X points (X0 to X21, X0 to X27), at this time the X points of the expansion module will be stored starting from X30.

2.1.2 Output relay (Y)

The output relay is a Devices directly connected to the hardware port of the external user control device, and logically corresponds to the physical output port of the PLC. After the PLC scans the user program each time, the component status of the Y relay will be transmitted to the hardware port of the PLC. 0 means the output port is open; 1 means the output port is closed.

Y relay numbers are identified by Signs such as Y0, Y1,...Y7, Y10, Y11, etc., and their serial numbers are numbered in octal format. Y relay components can be used indefinitely as promised by users .

In terms of hardware, according to the different output components, it can be divided into relay type, transistor type, solid state relay type, etc. If there are output expansion module ports, they are numbered in sequence starting from the main module.

2.1.3 Internal relay (M)

The auxiliary relay M element is used as an intermediate variable in the execution of the user program, just like the auxiliary relay in the actual electronic control system, used for the transmission of status information, and multiple M variables can also be combined into word variables. M variables and external ports There is no direct connection, but you can copy X to M through program statements, or copy M to Y to connect with the outside world. An M variable can be used unlimited times.

The auxiliary relay M is identified by Signs such as M0, M1....., M7999, and its serial number is numbered in decimal system.

2.1.4 Status relay (S)

The state relay S is used for the design and execution of the step program. The STL step instruction is used to control the transfer of the step state S, simplifying the programming design. If STL programming is not used, S can be used as an M variable. State S variables are identified by Signs such as S0, S1...S4095, and their serial numbers are numbered in decimal system.

2.1.5 Timer (T)

The timer T is equivalent to the time relay in the relay system and is used to complete the timing function. The timer is an addition expression. When the timer expires, the current value and the set value are the same value.

The measurement starts when the timer coil turns ON. When the current value of the timer is consistent with the set value, it will become the time limit, and the timer contact will turn ON. When the timer coil is turned OFF, the current value will become 0, and the timer contact will also become OFF.

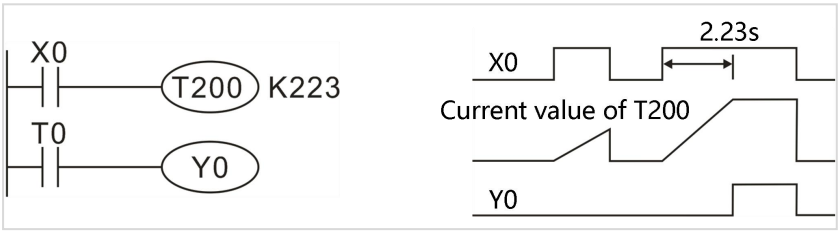
The T value range of the timer is 0 to 32767.

When the timer coil (OUT T instruction) is executed, the timer coil is turned on/off, the current value is updated, and the contact is turned on/off.

Device number	Timer	Device number	Timer
T0 to T191	100ms timer	T246 to T249	1ms accumulative timer
T192 to T199	100ms subroutine timer (used in the subroutine, even if the subroutine is not called, it will still be updated)	T250 to T255	10ms cumulative timer
		T256 to T383	1ms timer

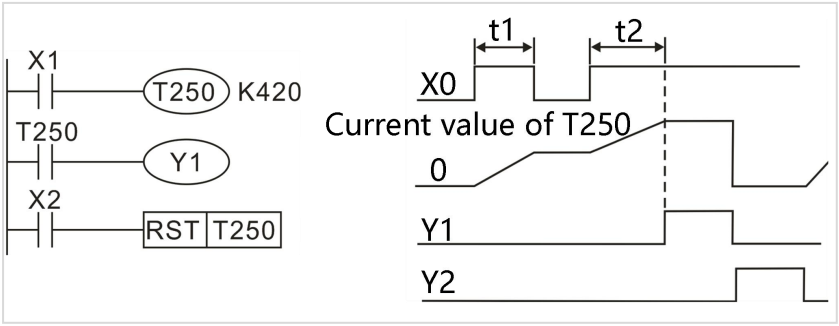
Device number	Timer	Device number	Timer
T200 to T245	10ms timer	T384 to T511	0.1ms timer

(1) General-purpose timer



As shown in the figure above: when the normally open contact of X0 is turned on, the current value counter of T200 starts timing from zero and counts up the 10ms clock pulse. When the current value is equal to the set value 223, the timer's normally open contact is turned on and the normally closed contact is turned off, that is, the output contact of T200 will act after its coil is driven for 2.23s. After the normally open contact of X0 is disconnected, T200 is reset because the coil is de-energized. After reset, its normally open contact is disconnected, and the normally closed contact is connected, and the current value returns to zero.

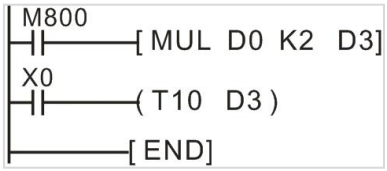
(2) Accumulative timer



When the X1 normally open contact in Figure b is turned on, the current value counter of T250 accumulates the 10ms clock pulse. When the normally open contact of X1 is disconnected or stopped, the counting stops, and the current value remains unchanged. When the normally open contact of X1 is turned on again, counting continues. When the accumulated time t1+t2 is 4.2s, the current value is equal to the set value of 420, the normally open contact of T250 is turned on and the normally closed contact is turned off. When the normally open contact of X2 is turned on, T250 will reset (because the coil of the accumulative timer will not reset when the power is off, you need to use the normally open contact of X2 and the reset instruction to force T250 to reset).

(3) Setting value

The timer time can use the constants (K, H) in the program memory as the set value, or can be specified indirectly by the content of the data register (D).



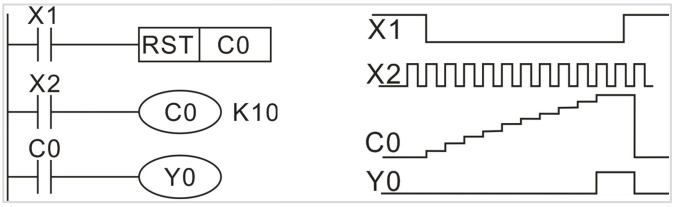
After PLC is powered on, multiplication is performed, D3=D0*2. Use the data of D3 as the timing time value of T10.

2.1.6 Counter (C)

The counter is used to complete the counting function. Each counter contains a coil, a contact, and a timer value register. Whenever the driving signal of the counter coil changes from OFF to ON, the counter reading value increases by 1. If the timer value reaches the preset time value, its contact action, a contact (NO contact) is closed, b contact (NC contact) is opened; If the timing value is cleared, the output a contact will be opened, and b contact (NC contact) will be closed. Some timers have features such as power-down retention, accumulation, etc., and maintain the value before power-down after power-on again.

The counters are identified by C0, C1,..., C255, and the order is numbered in decimal.

The counter (C) is a 16-bit counter.



The setting value of the 16-bit up counter is 1 to 32767. As shown in the working process of the up counter in Figure c, after the normally open contact of X1 in the figure is turned on, C0 is reset, its corresponding bit storage unit is set to 0, the normally open contact of C0 is disconnected, and the normally closed contact Point is turned on, and its current counter value is set to 0 at the same time. X2 provides a counting input signal. When the reset input circuit of the counter is disconnected and the counting input circuit changes from disconnected to connected (that is, the rising edge of the counting pulse), the current value of counter C0 is increased by 1. After 10 count pulses, the current value of C0 is equal to the set value of 10, and its corresponding bit storage unit is set to 1, and the Y0 contact is turned on at this time. When counting pulses again, the current value does not change until the reset input signal is turned on, and the current value of the counter is set to 0.

2.1.7 Long counter (LC)

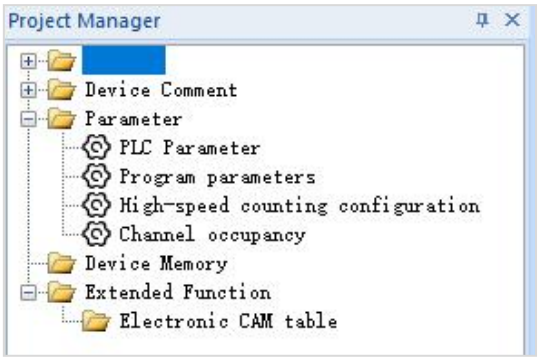
The long counter (LC) is basically the same as the counter (C), but compared to the counter (C), the long counter (LC) is a 32-bit register, and the range of values that can be counted is larger.

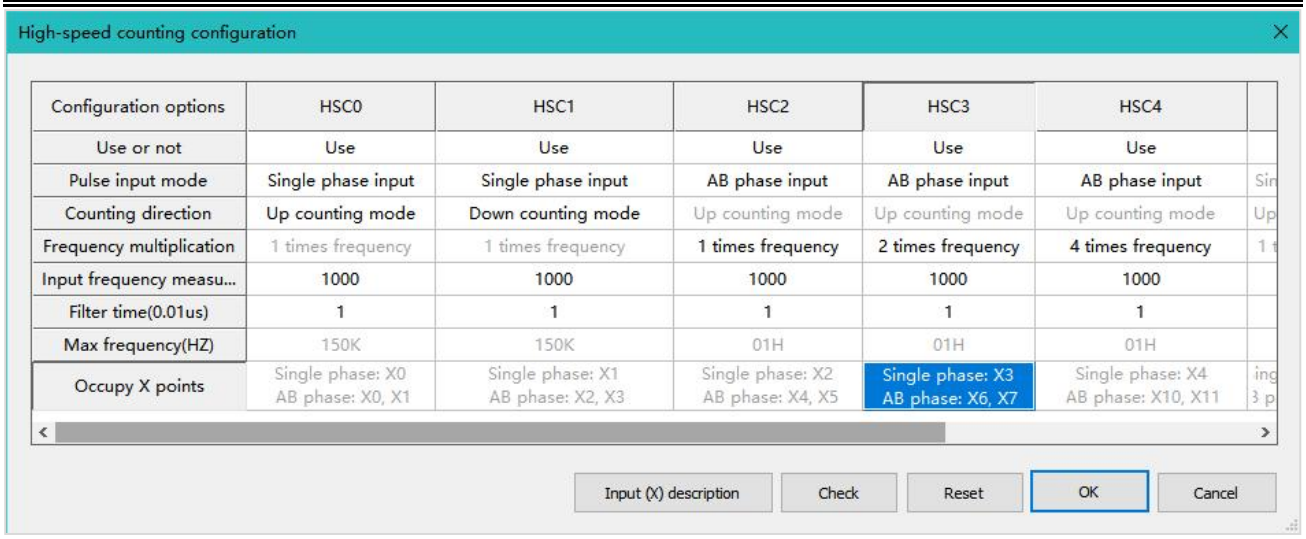
The long counter is identified by LC0, LC1,...,LC255, and the sequence is numbered in decimal.

2.1.8 High-speed counter (HSC)

High-speed counter (HSC) is a device used for counting through external input of high-speed pulse signals. HSC is a 32-bit register.

The corresponding parameter configuration can be configured through: “project management” -> “parameters” -> “high-speed counter configuration”:





2.1.9 Data register (D & R)

Registers are used for data calculation and storage, such as the calculation and calculation of timers, counters, and analog parameters. The width of each register is 16 bits. If 32bit instructions are used, the adjacent registers are automatically formed into 32bit registers for use, the lower address is the low byte, and the higher address is the high byte.

The address range of D register: D0 to D7999; the address range of R register: R0 to R29999.

The data involved in operations in most of our series PLC instructions are processed as signed numbers. For 16-bit registers, bit15 is the sign bit (0 represents a positive number, 1 represents a negative number); for a 32-bit register, the high byte bit15 It is the sign bit, and the value range is -32768 to 32767.

When 32-bit data needs to be processed, the two adjacent D registers can be formed into a 32-bit double word. For example, when accessing D100 in 32-bit format, use the high address D101 register as the high word and the high byte bit 15 as The sign bit of a double word can handle values from -2147483648 to 2147483647.

2.2 System device

2.2.1 Special relay (SM)

The special relay SM is an internal relay with a certain specification inside the programmable controller, so it cannot be used in the program like ordinary internal relays. It can be turned ON/OFF as needed to control the PLC.

For details, please refer to Special relays (SM) list.

2.2.2 Special register (SD)

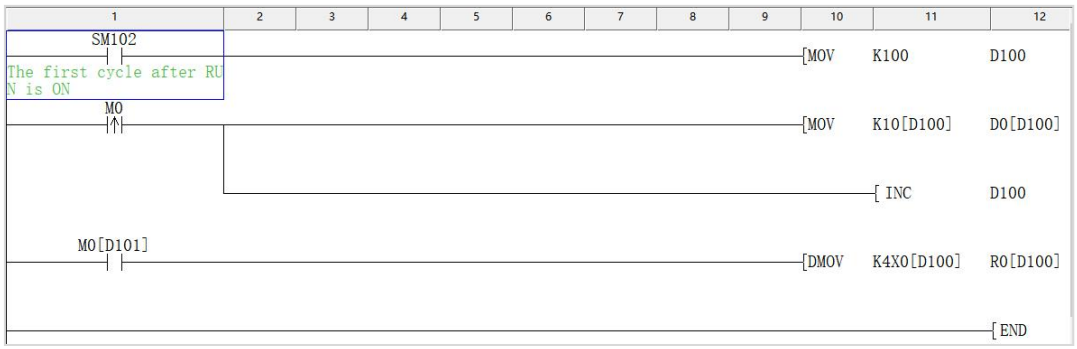
The special register SD is an internal register whose specifications are determined within the programmable controller, so it cannot be used in the program like a normal internal register, and the corresponding data can be written as needed to control the PLC.

For details, please refer to Special register (SD) list.

2.3 Index register

2.3.1 Index Register ([D])

The index register is used to modify the index of the Devices. [D] The index register is actually the same as the data register D, ranging from D0 to D7999. The input method is as follows, just add [D] directly after the Devices:



The supported devices for index modification are as follows:

- Constant K, H plus index modification, such as D0 = 10, K10[D0] result = 10 + 10 = 20.
- Constant E and character strings do not support index modification.
- Add index modification to the data device, such as D0 = 10, the result of D10[D0] is the value of D20. Even if D10[D0] is used in a double word instruction, the double word value is the value of D20 (low word) and D21 (high word).
- Bit device plus index modification, such as D0 = 10, the result of M0[D0] is the value of M10.
- Bits are combined into words with index modification. For example, D0 = 10, K4M10[D0] first takes M10 offset by 10 addresses, and then combines them. The result is equivalent to K4M10.

Whether the index modification can be used depends on whether each instruction supports the format, you can check the "offset modification" in the description of the available device for each instruction.

2.4 Pointer

2.4.1 Pointer (P)

The pointer is the device used in the jump instruction (CJ instruction).

At present, the CALLinstruction directly uses the subroutine name to call, and no longer uses the P pointer.

2.5 Constant

The constants are explained below.

2.5.1 Decimal constant (K)

“K” is a Sign that represents a decimal integer and is specified by K☐ (for example: K123). It is mainly used to designate the set value

of a timer or counter or the value in the operand of an application instruction. In 16bit instructions, the value range of constant K is -32768 to 32767; in 32bit instructions, the value range of constant K is -2147483648 to 2147483647.

2.5.2 Hexadecimal constant (H)

"H" is the Sign of hexadecimal number, specified by H□ (example: H123), mainly used to designate the value of the operand of the application instruction. The value range of the constant H is 0000 to FFFF; in the 32-bit instruction, the value range of the constant K is 0000, 0000 to FFFF, FFFF.

2.5.3 Real number constant (E)

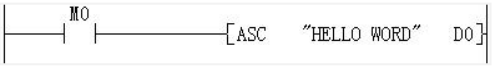
"E" is the single-precision floating-point number representation Sign, specified by E□ (example: E1.23), mainly used to specify the value of the operand of the application instruction, the value range of the single-precision floating-point number E is ±1.175495*10 -38 to ±3.402823*10+38 (±1.175495 E-38 to ±3.402823 E+38) and 0 (7 effective digits).



(The address occupies D1 and D0)

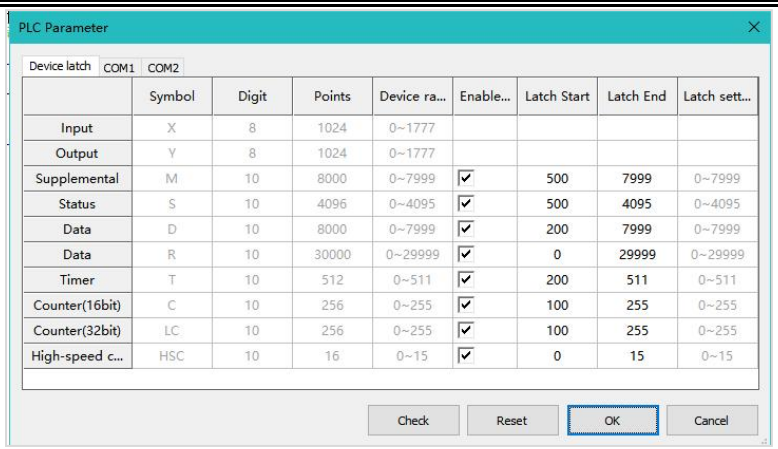
2.5.4 String constant

The character string constant is the device that specifies the character string, and only supports the ASCII code character set, and any character string ends with a NULL character (00H). To use string devices, you must use double quotation marks to modify the characters, as follows to convert the string to ASCII characters and fill in the device starting with D0:



2.6 Power-down retention setting

The user can freely configure the power-off storage range within the range of the Devices. The constant configuration is located in: "Project Management"→"Parameters"→"PLC Parameters"→"Device Latch".



✎**Note:** The X and Y registers do not support the power-down save function.

2.7 Special use of device

(1) Use bits to form words

Format: KnB

K is a fixed character.

The value of n is 1 to 8, which means that (n * 4) bits are combined into a word, such as K4M0 is a combination of M0 to M15.

B is the bit device number.

Example: Set a total of 32 bits M0 to M31 at the same time.



✎**Note:** KnB type can also support index modification.

(2) Take the bit in the word

Format: D.b

D is the number of data device D (R is not available).

b is the bit number that needs to be taken, hexadecimal, and the value range is 0 to F.

Example: bit14 in D2000 is set and Y0 is output



✎**Note:** D.b type can also support index modification.

3 Sequence control program instructions

3.1 Contact instructions

3.1.1 Operation start, series connection, parallel connection

LD, LDI, AND, ANI, OR, ORI

• **LD:** Normally open contact instruction. **LD**I: Normally closed contact instruction.

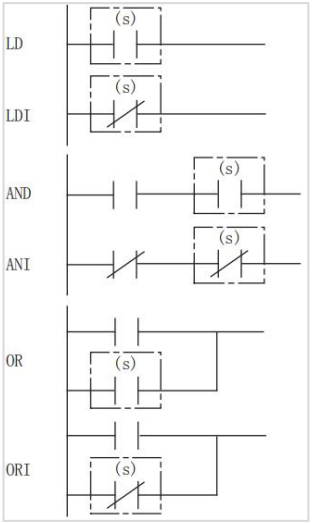
Extract the ON/OFF information of the device specified in (s) as the result of the calculation.

• **AND:** Normally open contact series connection instruction. **ANI:** Normally closed contact series connection instruction.

Extract the ON/OFF information of the device specified in (s), and perform an AND operation with the calculation result so far as the calculation result.

• **OR:** Parallel connection instruction of 1 normally open contact. **ORI:** Parallel connection instruction of 1 normally closed contact

Extract the ON/OFF information of the device specified in (s), and perform an OR operation with the result of the operation so far as the result of the operation.



Content, range and data type

Parameter	Content	Range	Data type	Data type (tag)
(s)	Device used as contact	---	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																							Index modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
LD	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
LDI	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
AND	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
ANI	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
OR	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
ORI	Parameter 1	●	●	●	●	●	●	●	●	●	●														●

Features

- **LD, LDI**

• The LD instruction is a normally open contact instruction, and the LDI instruction is a normally closed contact instruction. The ON/OFF information *1 of the specified device is extracted as the operation result.

*1: When the bit of the word device is specified, it is turned on/off according to 1/0 of the specified bit.

- **AND, ANI**

• The AND instruction is a normally open contact serial connection instruction, and the ANI instruction is a normally closed contact serial connection instruction. It extracts the ON/OFF information*1 of the specified bit device and performs an AND operation with the result of the operation so far. This value is used as the result of the operation.

*1: When the bit of the word device is specified, it is turned on/off according to 1/0 of the specified bit.

• There is no limit to the number of serial contacts, and this instruction can be used any time continuously.

• After the OUT instruction, it is called cascade output through the contact OUT to other coils. As long as the sequence is good, it can be repeated any number of times.

- **OR, ORI**

• The OR instruction is a parallel connection instruction for a normally open contact, and the ORI instruction is a parallel connection instruction for a normally closed contact. It extracts the ON/OFF information*1 of the specified device and compares it with the calculation result so far. Perform an OR operation and use the value as the result of the operation.

*1: When the bit of the word device is specified, it is turned on/off according to 1/0 of the specified bit.

• OR and ORI instructions start from the step where the instruction is located, and connect in parallel to the step where the previous LD and LDI instructions are located.

• There is no limit to the number of parallel connections.

Key point

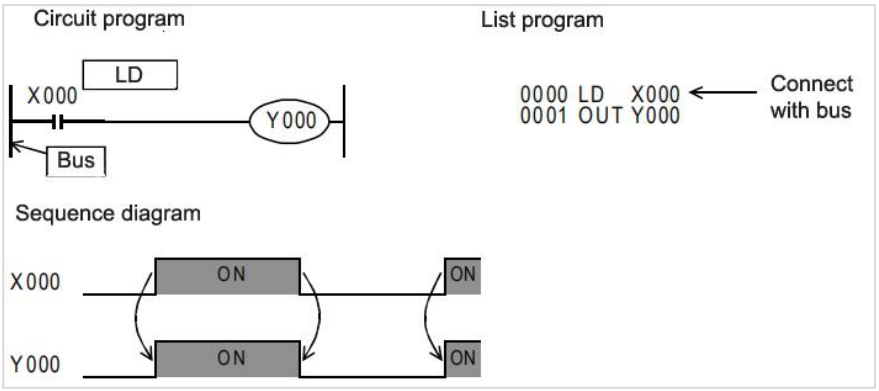
When specifying the bit of a word device, the bit is specified with a hexadecimal number. (For example, b11 of D0 will become "D0.B")

Error code

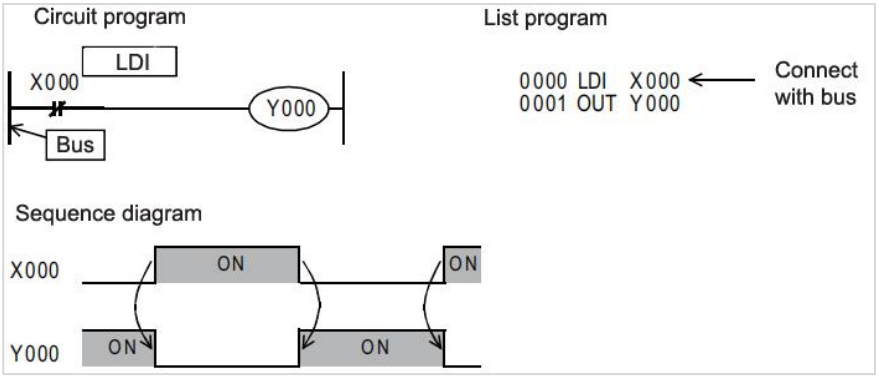
Error code	Content
4085H	(S) read address exceeds the device range

Example

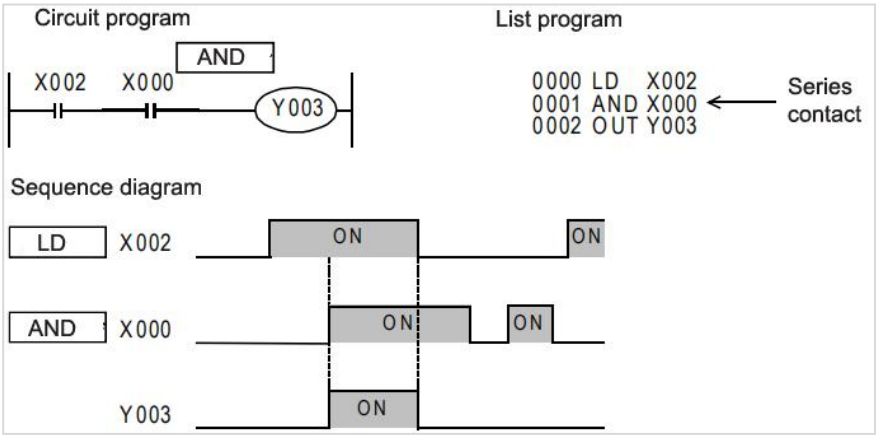
1) LD instruction (the logic operation of a contact starts)



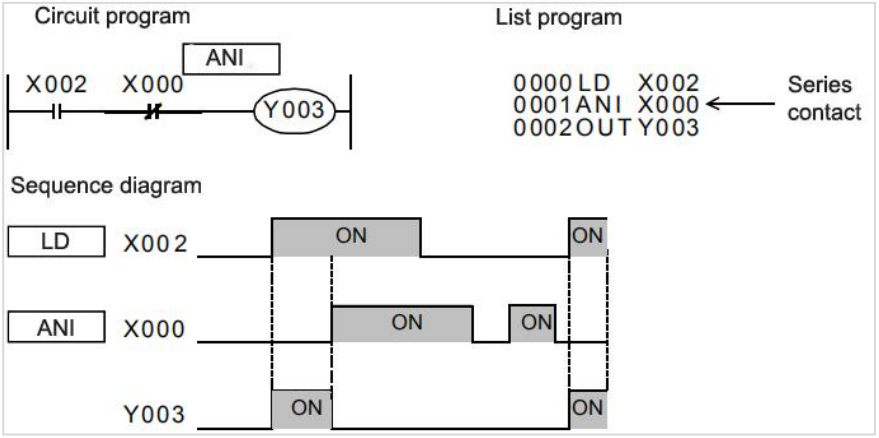
2) LDI instruction (the logic operation of contact b starts)



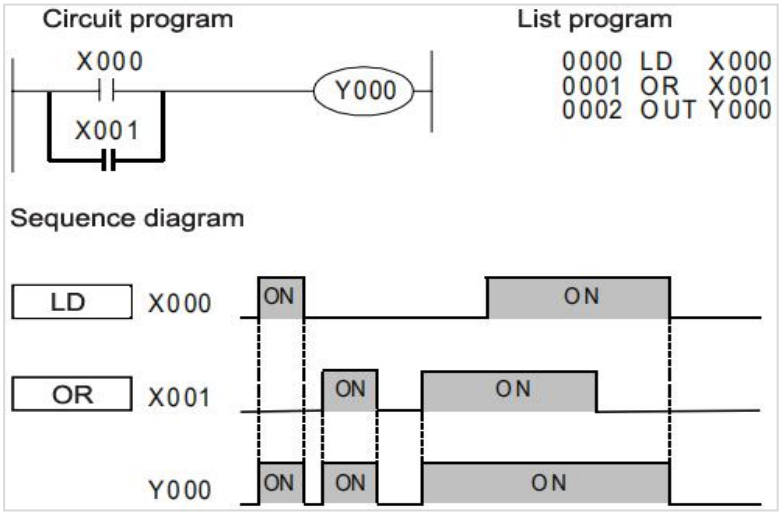
3) AND instruction (a contact in series)



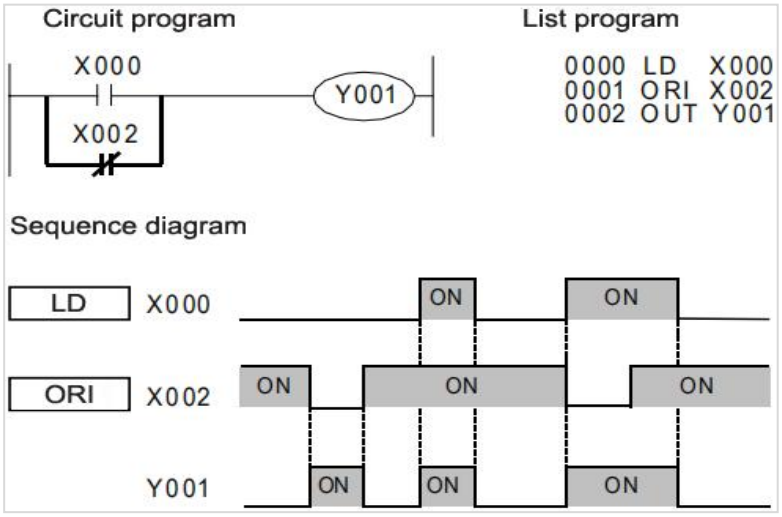
4) ANI instruction (series b contact)



5) OR instruction (a contact in parallel)



6) ORI instruction (a contact in parallel)



7) Index modification

The devices used in the LD, LDI, AND, ANI, OR, ORI instructions can all be indexed with D data devices (the status register S cannot be modified).

D0 to D7999 can be used in index modification.

When the devices are input (X) and output (Y), the value of the index register is converted into an octal number and then added.

Example



When the value of D0 is 10, X012 determines LD contact ON (conduction)/OFF (non-conduction).

8) Bit specification in the data register

Among the devices used in the LD, LDI, AND, ANI, OR, and ORI instructions, the bits of the data register (D) can be specified.

When executing the bit specification of the data register, enter "." after the number of the data register (D), and then enter the bit number (0 to F). The usable data registers are specified in bits, but only 16-bit data registers are valid.

Please specify the bit number in the order of 0,1,2,…9,A,B,…F starting from the lower bit.

Example



The third bit of D0 determines the LD contact ON (conduction)/OFF (non-conduction).

3.1.2 Pulse calculation starts,pulse series connection, pulse parallel connection

LDP, LDF, ANDP, ANDF, ORP, ORF

• LDP: Rising edge pulse operation start instruction.

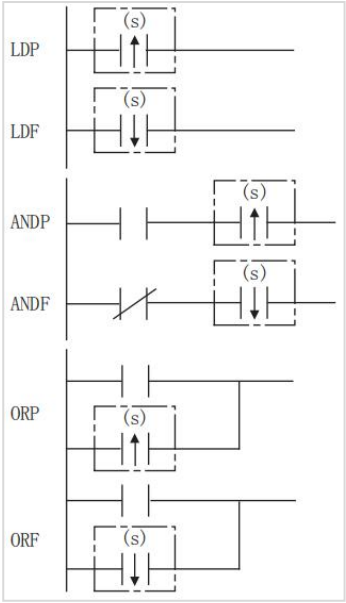
Turns on only at the rising edge (OFF→ON) of the bit device specified in (s).

• LDF: Falling edge pulse operation start instruction.

Turns on only at the falling edge (ON→OFF) of the bit device specified in (s).

• ANDP: Rising edge pulse series connection instruction, ANDF: Falling edge pulse series connection instruction. The previous operation result up to that time is ANDed with the bit device specified in (s) as the operation result.

• ORP: Parallel connection instruction for rising edge pulse/ORF: Parallel connection instruction for falling edge pulse. The operation result up to that time is ORed with the bit device specified in (s) as the operation result.



Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Devices used as contacts	-	Bit	ANY_BOOL

Device used

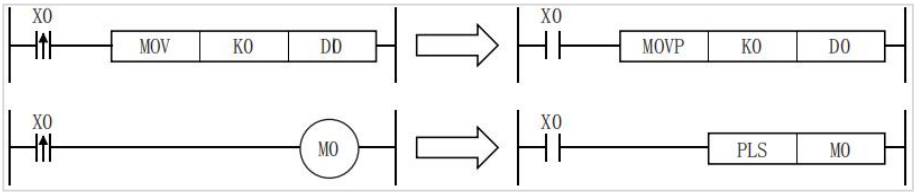
instruction	Parameter	Devices																							Index modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
LD	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
LDI	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
AND	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
ANI	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
OR	Parameter 1	●	●	●	●	●	●	●	●	●	●														●
ORI	Parameter 1	●	●	●	●	●	●	●	●	●	●														●

Features

- **LDP, LDF**

• The LDP instruction is a rising edge pulse operation start instruction, which turns on only at the rising edge (OFF→ON) of the specified bit device. When the bit of the word device is specified, it turns on only when the specified bit changes from 0→1. In the case of only the LDP instruction, it is the same as the pulsed instruction (P) of the instruction executed while ON.

When the circuit that uses the LDP instruction is replaced with a circuit that does not use the LDP instruction, the situation is as follows.



• The LDF instruction is a falling edge pulse instruction, which turns on at the falling edge (ON→OFF) of the specified bit device. When the bit of the word device is specified, it turns on only when the specified bit changes from 1→0.

- **ANDP, ANDF**

• The ANDP instruction is a series connection instruction for rising edge pulses, and the ANDF instruction is a series connection instruction for falling edge pulses. The AND operation is performed with the operation result up to that time as the operation result. The ON/OFF information used in ANDP instructions and ANDF instructions is shown in the table below.

Device specified in ANDP, ANDF		ANDP status	ANDF status
Bit device	Bit specification of word device		
OFF→ON	0→1	ON	OFF
OFF	0	OFF	OFF
ON	1	OFF	OFF
ON→OFF	1→0	OFF	ON

- **ORP, ORF**

• The ORP instruction is a parallel connection instruction for rising edge pulses, and an ORF instruction is a parallel connection instruction for falling edge pulses. The OR operation is performed with the operation result as of that time , with the result being used as the operation result. The ON/OFF information used in ORP instructions and ORF instructions is shown in the table below.

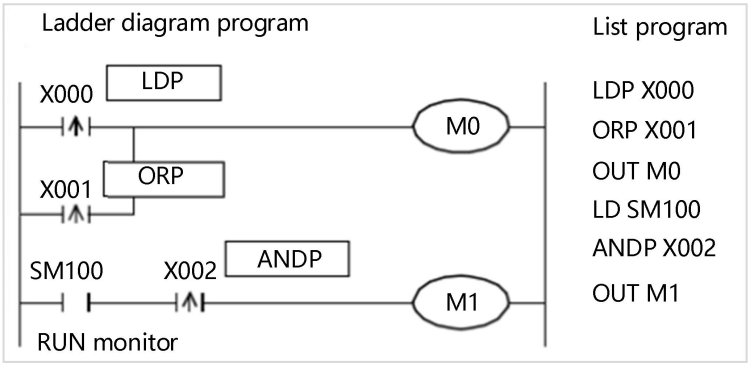
Device specified in ORP, ORF		ORP status	ORF status
Bit device	Bit specification of word device		
OFF→ON	0→1	ON	OFF
OFF	0	OFF	OFF
ON	1	OFF	OFF
ON→OFF	1→0	OFF	ON

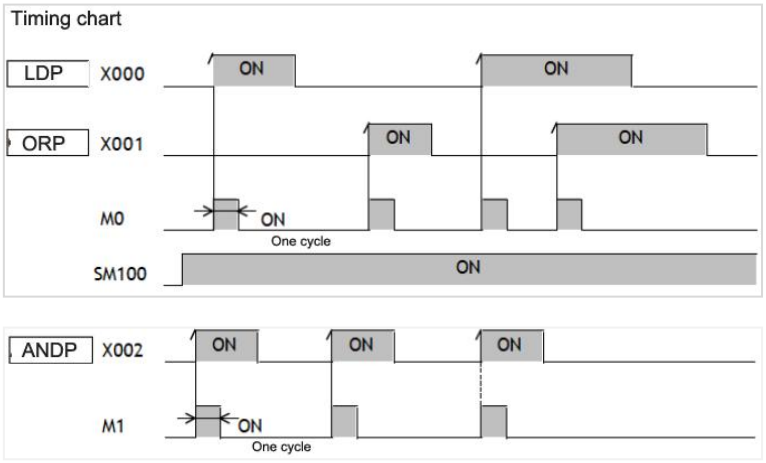
Error code

There is no operation error.

Example

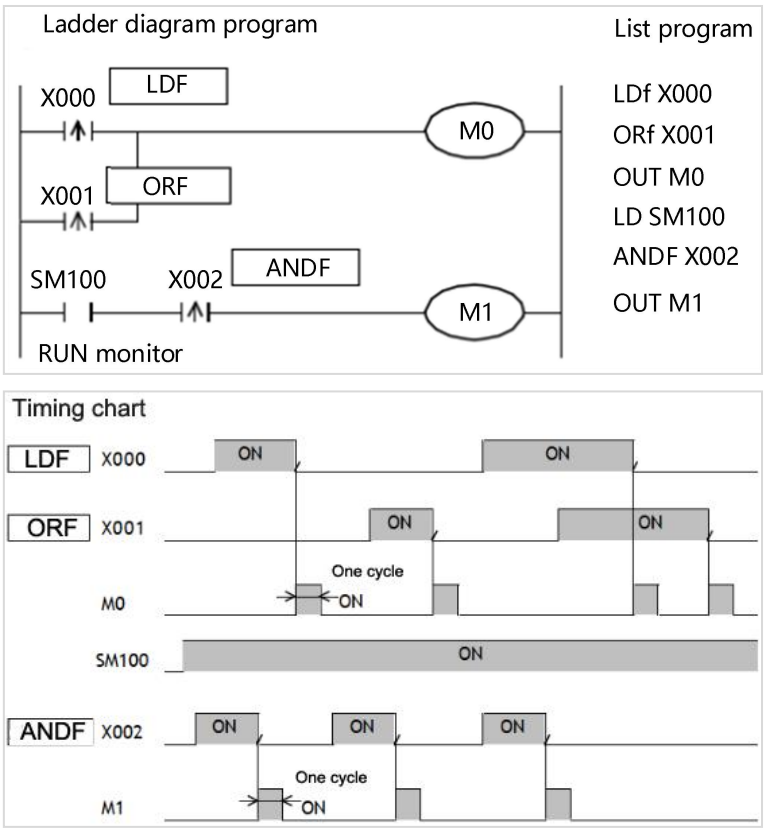
1) LDP, ANDP, ORP instructions (calculation starts when a rising edge is detected, serial connection, parallel connection)





In the above figure, when X000 to X002 changes from OFF to ON, M0 or M1 only maintains ON for 1 operation cycle.

2) LDF, ANDF, ORF instructions (calculation starts when a falling edge is detected, serial connection, parallel connection)



In the above figure, when X000 to X002 changes from ON to OFF, M0 or M1 only maintains ON for 1 operation cycle.

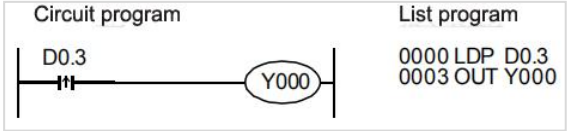
3) Bit specification of data register (D)

In the devices used for LDP, LDF, ANDP, ANDF, ORP, ORF instructions, the bits of the data register (D) can be specified.

To specify the bit of the data register, enter "." after the number of the data register (D), and then enter the bit number (0 to F). The usable data registers are specified in bits, but only 16-bit data registers are valid.

Please specify the bit number in the order of 0,1,2,...9,A,B,...F starting from the low order.

Example:



The third bit of D0 determines the LDP contact ON (conduction)/OFF (non-conduction) when it changes from OFF to ON.

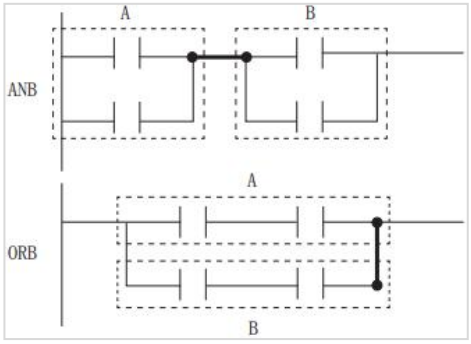
3.2 Combining instructions

3.2.1 Series connection and parallel connection of ladder diagram blocks

ANB, ORB

Perform AND operation or OR operation between block A and block B, and use it as the result of the operation.

Ladder diagram



Features

- **ANB**

• Perform AND operation of block A and block B and use it as the result of the operation.

• The Sign of the ANB instruction is not a contact Sign, but a connection Sign.

- **ORB**

• Perform OR operation of block A and block B, and use it as the result of the operation.

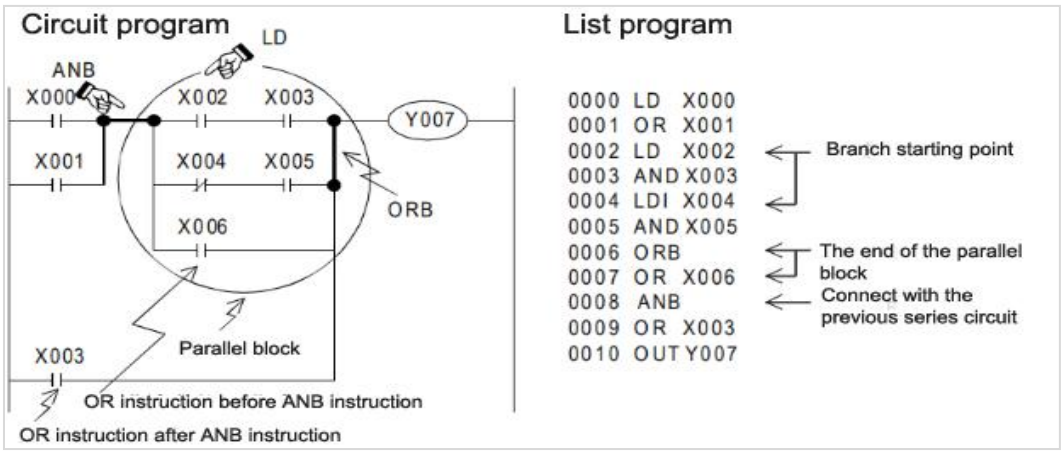
• The ORB instruction connects circuit blocks with 2 or more contacts in parallel. The parallel connection of only 1 contact uses OR instruction and ORI instruction, without ORB instruction.

• The Sign of the ORB instruction is not a contact Sign, but a connection sign.

Error code

There is no operation error.

Example

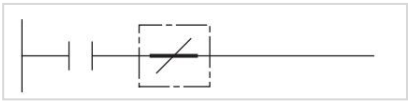


3.2.2 Invert the result of operation

INV

Invert the results of operations up to the INV instruction.

Ladder diagram



Features

Invert the results of operations up to the INV instruction.

Operation result before INV instruction	Operation result after INV instruction is executed
OFF	ON
ON	OFF

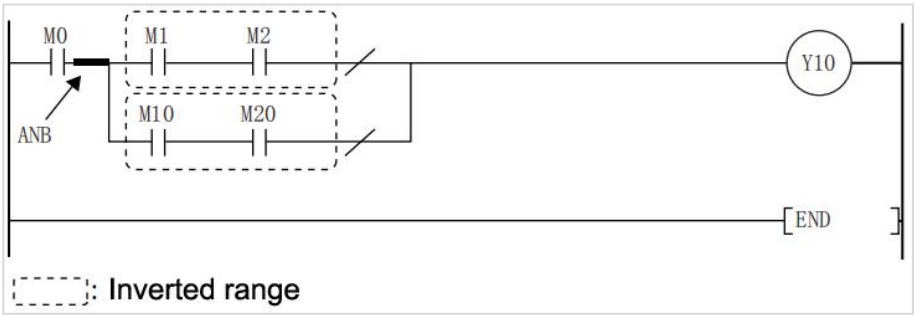
Error code

There is no operation error.

Point

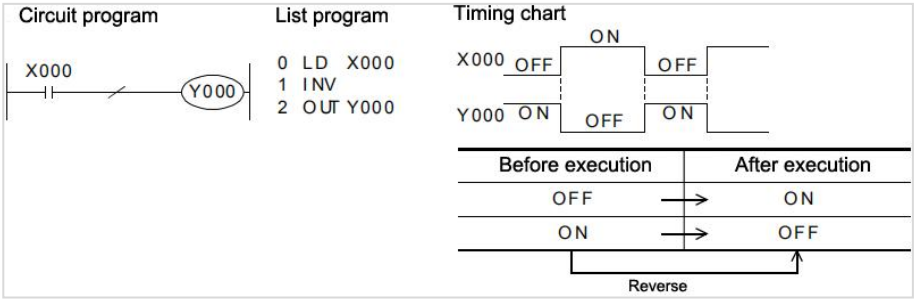
• The INV instruction executes the operation as a result of the previous operation, so it should be used in the same position as the AND instruction. INV instruction cannot be used in the position of LD and OR instructions.

• When the INV instruction and ANB instruction are used together for ladder diagram operation, pay attention to the inverted range.



Example

INV instruction (reverse operation result)



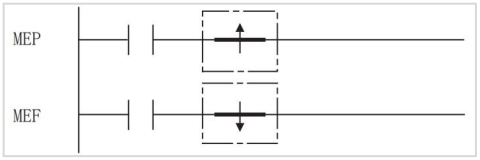
3.2.3 Pulse operation result

MEP, MEF

• MEP: Turns on when the operation result before the MEP instruction is a rising edge, and turns off when it is not a rising edge.

• MEF: Turns on when the operation result before MEF instruction is a falling edge, and turns off when it is not a falling edge.

Ladder diagram



Features

- **MEP**

• When the operation result before the MEP instruction is a rising edge (OFF→ON), it becomes ON (conduction state). When the operation result before the MEP instruction is not the rising edge, it turns off (non-conduction state).

• When using the MEP instruction, if multiple contacts are connected in series, pulse processing will be easier.

- **MEF**

• When the operation result before the MEF instruction is a falling edge (ON→OFF), it becomes ON (conduction state). When the operation result before the MEF instruction is not the falling edge, it turns OFF (non-conduction state).

• When using the MEFinstruction, if multiple contacts are connected in series, pulse processing will be easier.

Error code

There is no operation error.

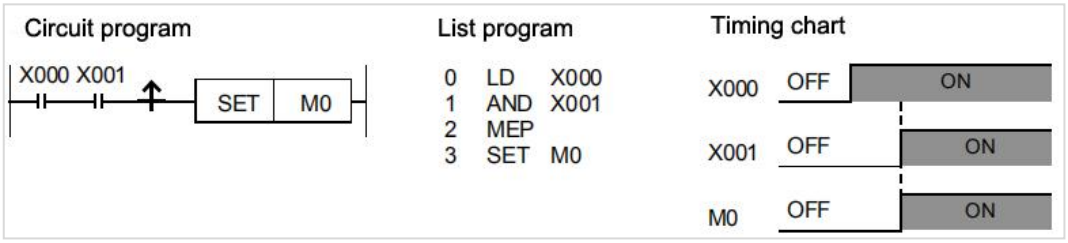
Point

• For MEP instructions and MEF instructions, if the indexed contacts are pulsed by subroutines, FOR to NEXT instructions, etc., they may not operate normally.

• The MEP instruction and MEF instruction perform actions based on the previous calculation results, so they should be used in the same position as the AND instruction. The MEP instruction and MEF instruction cannot be used in the position of LD instruction and OR instruction.

Example

1) MEP instruction (ON at the rising edge of the operation result)



2) MEF instruction (ON at the falling edge of the operation result)



3.3 Output instructions

3.3.1 OUT instruction (except timers and counters)

Output the results of the previous OUT instruction to the specified device.

Ladder diagram



Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	ON/OFF device number	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																								Index modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
OUT	Parameter 1	●	●	●	●	●					●					●	●				●	●				●

- Refer to OUT T instruction when using T;
- Refer to OUT C instruction when using C, LC, HSC;
- Index modification cannot be used when using S device.

Features

Output the results of the previous OUT instruction to the specified device.

Condition	Calculation result	Coil / specified bit
When using bit devices	OFF	OFF
	ON	ON
When using word devices	OFF	0
	ON	1

Error code

Error code	Content
4086H	(D) write address exceeds the device range

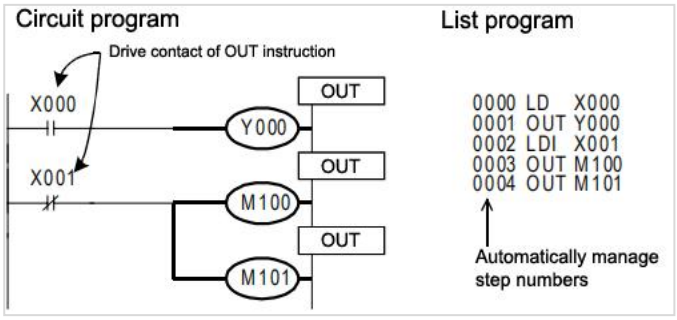
Example

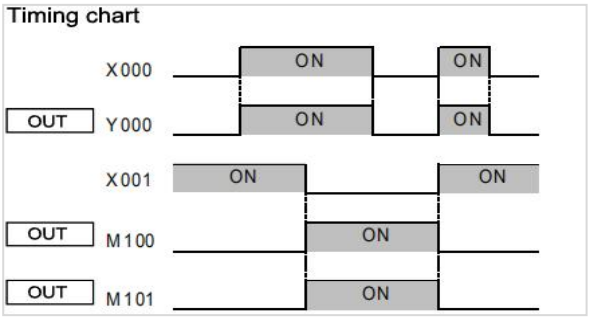
1) When using bit devices

The device programmed with the OUT instruction executes ON/OFF according to the state of the drive contact, and the parallel OUT instruction can be used continuously for many times.

In the following program example, this is the case for the OUT M101 followed by OUT M100.

However, if multiple OUT instructions are used for the same device number, it will become a dual output (double coil). Please be careful.





2) Index modification

All the devices used in the OUT instruction can be indexed with the D data device (the status register S cannot be modified).

D0 to D7999 can be used in index modification.

When the devices used are input (X) and output (Y), the value of the index register is converted to an octal number and then added.

Example:



When the value of D0 is 10, when X0 is ON (conducting), the Y12 contact is ON (conducting).

3) Bit specification in the data register

Among the devices used in the OUT instruction, the bit of the data register (D) can be specified. To specify the bit of the data register, enter "." after the number of the data register (D), and then enter the bit number (0 to F). The usable data registers are specified in bits, but only 16-bit data registers are valid.

Please indicate the positioning number in the order of 0,1,2,...9,A,B,...F starting from the low order.

Example:



In the example on the left, the state of X1 determines the ON (conducting)/OFF (non-conducting) of the third bit in D0.

3.3.2 SET instruction

When the execution instruction turns ON, the device specified in (d) will be in the following state.

• Bit device: Turn on the coil and contact.

• Bit specification of word device: Set the specified position to 1.



Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	Set (ON) bit device number/bit specification of word device	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																							Index modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
SET	Parameter 1		•	•	•	•					•															•

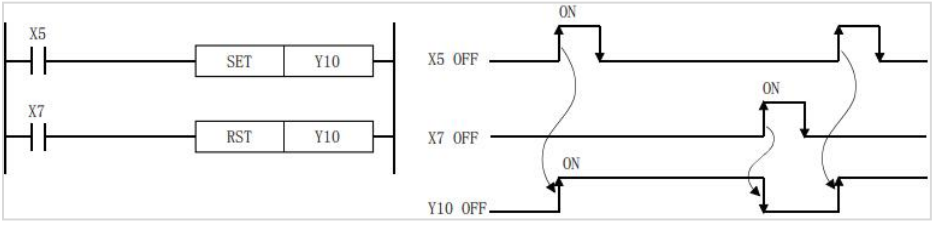
*1: Index modification cannot be used when using S devices.

Features

When the execution instruction turns ON, the device specified in (d) will be in the following state.

Devices	Device status
Bit device	Turn on the coil and contact
Bit specification of word device	Set the specified position to 1

The device that is turned on will remain on even if the execution instruction turns off. The device that is turned ON by the SET instruction can be turned OFF by the RST instruction.



✎Note:

For the output relay (Y), if the SET instruction and the RST instruction are executed in the same operation, the instruction result close to the END instruction (end of program) will be output.

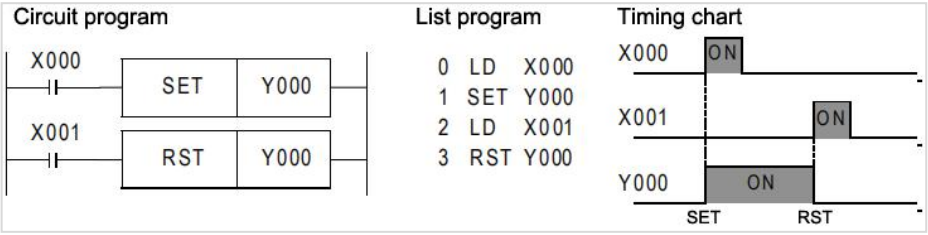
Error code

Error code	Content
4086H	(d) In the case of using offset, the offset address exceeds the device range

Example

1) When using bit devices

The parallel SET instruction could be used multiple times in succession. In the following program example, this is the case for the program with SET Y000 followed by RST Y000.



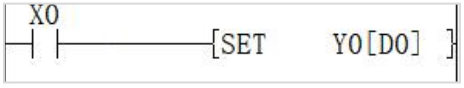
2) Index modification

All the devices used in the SET instruction can be indexed with D data devices (the status register S cannot be modified).

D0 to D7999 can be used in index modification.

When the devices used are input (X) and output (Y), the value of the index register is converted into octal number and then added.

Example:



When the value of D0 is 10, when X0 is ON (conduction), the Y12 contact is ON (conduction), X0 is OFF (non-conduction), and the Y12 contact remains unchanged.

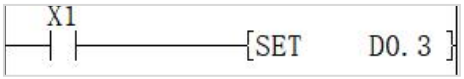
3) Bit specification in the data register

Among the devices used in the SET instruction, the bits of the data register (D) can be specified.

To specify the bit of the data register, enter "." after the number of the data register (D), and then enter the bit number (0 to F). The usable data registers are specified in bits, but only 16-bit data registers are valid.

Please specify the bit number in the order of 0,1,2,…9,A,B,…F starting from the lower bit.

Example:



In the example on the left, the state of X1 is ON (conduction), and the third bit in D0 is ON (conduction). The state of X1 is OFF (non-conduction), and the state of the third bit in D0 remains unchanged.

3.3.3 RST instruction

When the RST input turns ON, the device specified in (d) will change to the following state.

• Bit device: Turn off the coil and contact.

• Timers and counters: Set the current value to 0, and set the coil and contact to OFF.

• Bit specification of word device: Set the specified position to 0.

• Word device, module access device, index register: Set the content to 0.



Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	Reset bit device number/bit specification of word device or reset word device number	-	Bit/word/double word	ANY_ELEMENTARY

Device used

instruction	Parameter	Devices																							Index modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
RST	Parameter 1	•	•	•	•	•	•	•	•	•	•					•	•	•	•	•	•	•				•

*1: Index modification cannot be used when using S devices.

Features

When the execution instruction is ON, the specified device will be in the following state.

Devices	Device status
Bit Device	Turn on the coil and contact
Timer, counter	Set the current value to 0, set the coil and contact to OFF
Bit specification of word device	Set the specified position to 0
Word device	Set the content to 0

When the execution instruction is OFF, the device status does not change.

The function when specifying a word device with the RST instruction is the same as the following ladder diagram.



✎Note:

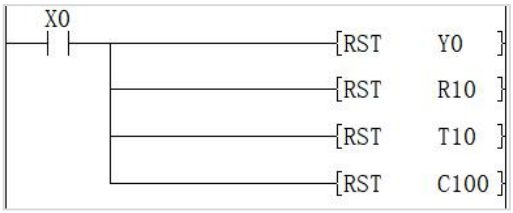
For timers and counters, when the RST instruction is executed in the program, subroutine, and interrupt program where the RST instruction is jumped, the timer and counter may remain unchanged after reset, and the timer and counter will not operate.

Error code

Error code	Content
4086H	(d) write address exceeds the device range

Example

1) Use bit device



When X0 is ON (conducting), Y0 is set to OFF, R10 is set to 0, the word device of T10 is set to 0, the bit device is set to OFF, and the word device of C100 is set to 0. The device is turned off.

When X0 is OFF (non-conduction), all states remain unchanged.

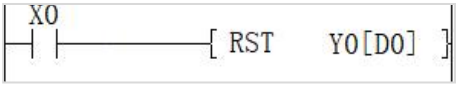
2) Index modification

All the devices used in the RST instruction can be indexed with D data devices. (The status register S could not be modified)

D0 to D7999 can be used in index modification.

When the devices used are input (X) and output (Y), the value of the index register is converted into an octal number and then added.

Example



When the value of D0 is 10, when X0 is ON (conduction), the Y12 contact is OFF (conduction), X0 is OFF (non-conduction), and the Y12 contact remains unchanged.

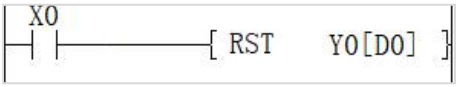
3) Bit specification in the data register

Among the devices used in the RST instruction, the bits of the data register (D) can be specified.

To specify the bit of the data register, enter "." after the number of the data register (D), and then enter the bit number (0 to F). The usable data registers are specified in bits, but only 16-bit data registers are valid.

Please specify the bit number in the order of 0,1,2,…9,A,B,…F starting from the lower bit.

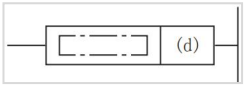
Example



In the example on the left, the state of X1 is ON (conduction), and the third bit in D0 is OFF (conduction). The state of X1 is OFF (non-conduction), and the state of the third bit in D0 remains unchanged.

3.3.4 PLF/Falling edge output

When the PLF instruction is ON→OFF, one scan of the device specified in (d) is ON, and when it is other than ON→OFF, it is OFF.



Content, range and data type

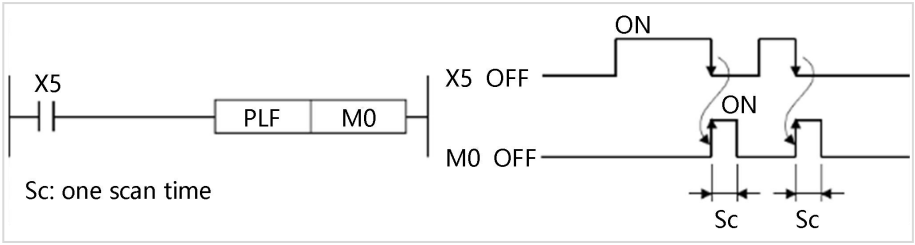
Parameter	Content	Range	Data type	Data type (label)
(d)	Pulsed device	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																							Index modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
PLF	Parameter 1	•	•	•	•	•					•															

Features

When the execution instruction is ON→OFF, the specified device is turned ON, and when the execution instruction is other than ON→OFF, it is turned OFF. When there is one PLF instruction for the device specified in (d) in one scan, the specified device will turn on one scan.



✎Note:

If the PLF instruction is jumped by the CJ instruction, or the executed subroutine is not called by the CALL(P) instruction, the device specified in (d) may be turned on for more than one scan. Be careful.

Error code

No Error code

Example

PLF instruction (differential output of falling edge)



In the above figure, when X000 changes from ON to OFF, only one operation cycle of M1 is ON.

3.3.5 PLS/Rising edge output

When the PLS instruction is OFF→ON, one scan of the device specified in (d) is turned ON, and when it is other than OFF→ON, it is turned OFF.



Content, range and data type

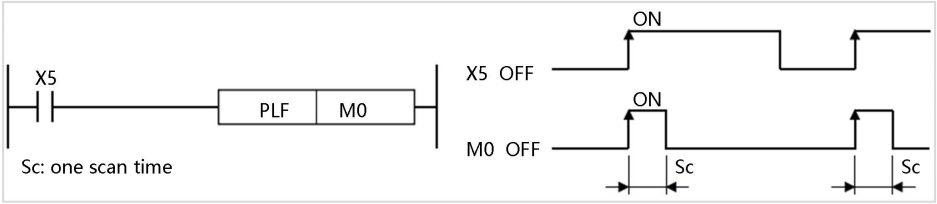
Parameter	Content	Range	Data type	Data type (label)
(d)	Pulsed device	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																								Index modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
PLS	Parameter 1	•	•	•	•	•					•															

Features

When the PLS instruction is OFF→ON, one scan of the specified device is turned on, and when it is other than OFF→ON, it is turned off. When there is one PLS instruction for the device specified in (d) in one scan, the specified device turns on one scan.



✎Note:

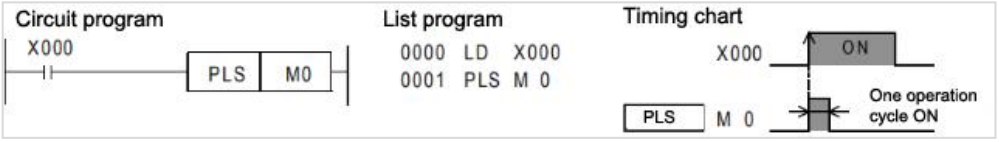
If the PLS instruction is jumped by the CJ instruction, or the executed subroutine is not called by the CALL(P) instruction, the device specified in (d) may be turned on for more than one scan. Be careful.

Error code

No Error code

Example

PLS instruction (differential output on rising edge)



In the above figure, when X000 changes from OFF to ON, only one operation cycle of M0 is ON.

4 Program flow instructions

4.1 Program jump

4.1.1 CJ/Conditional jump

When the jump instruction is ON, the program with the specified pointer number in the same program file is executed.

-[CJ (P) (P)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(P)	The pointer number of the jump target	P0 to P4095	Device name	POINTER

Device used

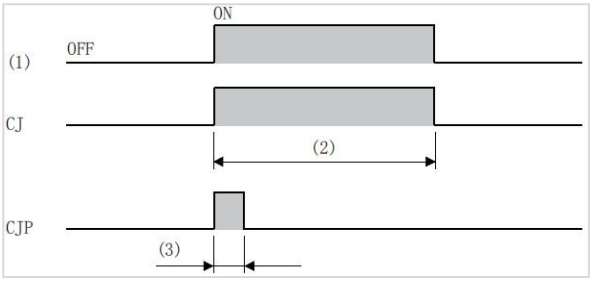
instruction	Parameter	Devices																								Index modification	Pulse extension	Other
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP	P
CJ	Parameter 1																										•	•

Features

- CJ(P)

When the execution instruction is ON, the program with the specified pointer number is executed.

When the execution instruction is OFF, execute the next program.



1) Execute instructions.

2) Each scan is executed.

3) One scan is executed.

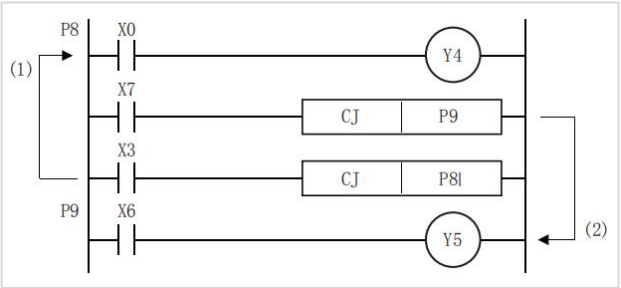
✎Note:

After turning ON the coil of the timer, if the timer whose coil is ON is jumped by the CJ(P) instruction, the measurement will not be performed normally.

When the OUT instruction is jumped by the CJ(P) instruction, the scan time will be shorter.

When the CJ(P) instruction is used to jump backward, the scan time will be longer.

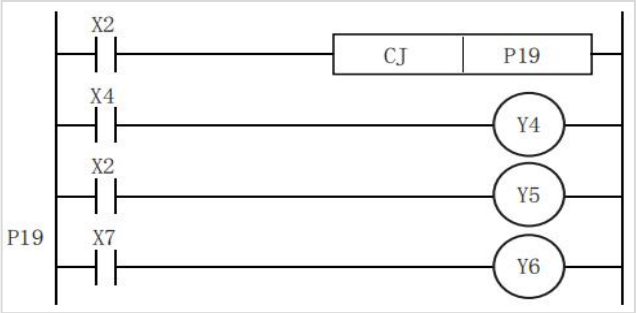
For the CJ(P) instruction, you can jump to a step smaller than the step number being executed. However, in order to avoid the time limit of the watchdog timer, a method of jumping out of the loop during this period should be considered.



(1) While X3 is ON, the loop is executed.

(2) When X7 is set to ON, it jumps out of the loop.

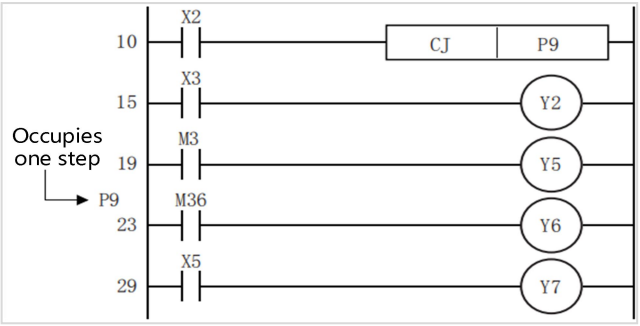
• The device skipped by the CJ(P) instruction does not change.



When X2 is ON, jump to the label of P19.

Even if X2 and X4 turn ON/OFF during CJ instruction execution, Y4 and Y5 will not change.

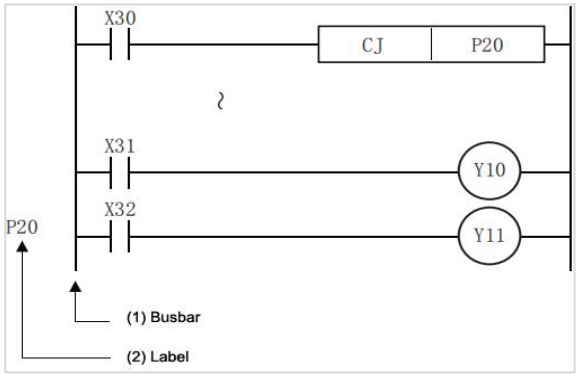
• The label (P□) occupies 1 step.



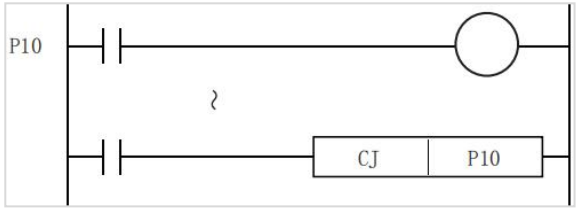
The jump instruction can only specify the pointer number in the same program file.

When jumping to the pointer number within the jump range during jump operation, the program after the jump destination pointer number is executed.

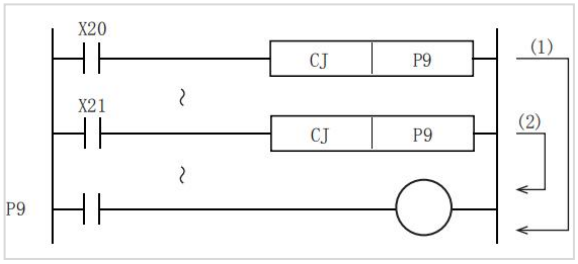
The label procedure is shown below. When creating a loop program, move the cursor to the left of the bus bar of the ladder diagram, and enter the label (P) at the beginning of the loop block.



It is also possible to program the label at the position where the step number is less than the CJ instruction, but if the scan time becomes more than 200ms (default setting), a watchdog timer error will occur, which requires attention.



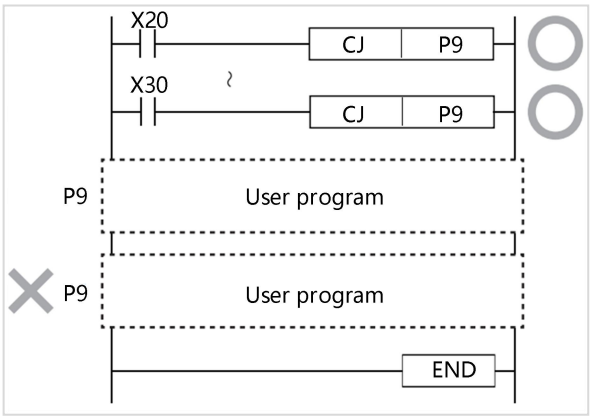
When the pointer number in the operand is the same and the label is one, the operation is as follows.



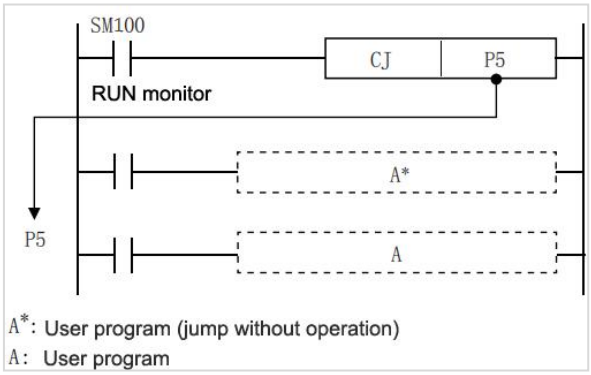
(1) When X20 is ON, jump from the CJ instruction of X20 to label P9.

(2) When X20 is OFF and X21 is ON, jump from the CJ instruction of X21 to label P9.

If the tag number is reused, it will become an error state.



SM100 is always ON during the operation of the CPU module, so the usage method shown below will jump unconditionally.



The pointer number P63 of LX3V represents the jump to the END instruction. The P63 pointer of LX5V no longer provides this function. If you need to use this function, please use the GOEND instruction.

Error code

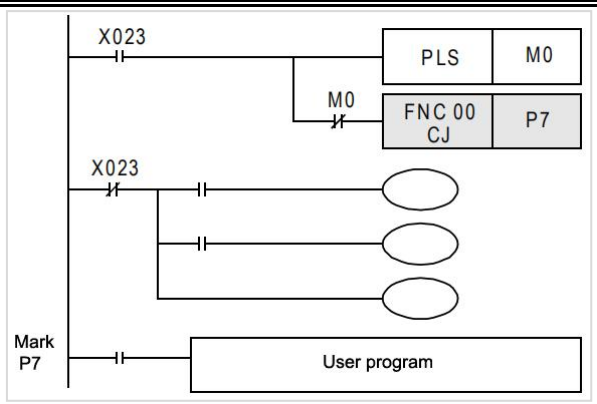
No error message

Example

1) The situation to jump after OFF processing

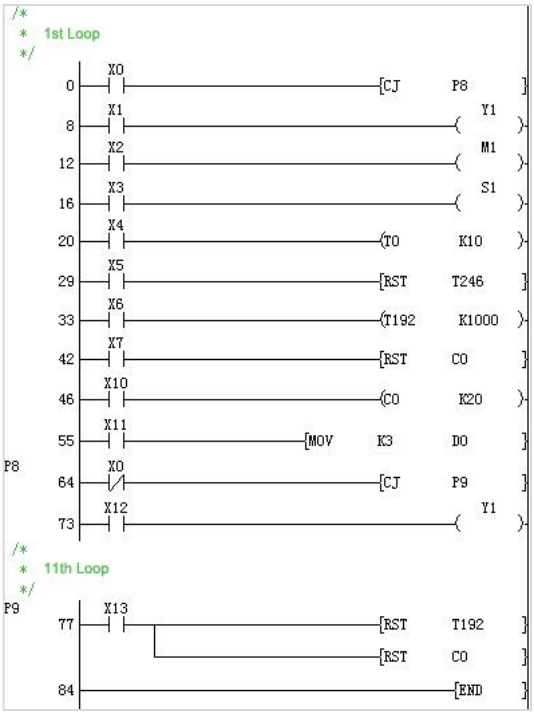
After one operation cycle when X023 changes from OFF to ON, the CJ P7 instruction is valid.

With this method, the output between CJ P7 instruction and mark P7 can be turned off before jumping.



2) CJ instruction and action of contact coil

In the following program example, when X000 is ON, jump from the CJ instruction of the first loop to the mark P8. When X000 is OFF, no jump is performed, but the program is executed in order from step 1, and the CJ instruction in the 11th loop jumps to mark P9. The jumped instruction is not executed.



Double-coil action of Y001 output:

When X000=OFF, it will act through X001.

When X000=ON, it will act through X012.

Even if the program is distinguished by conditional jump, if the same coil (Y000) is programmed twice or more within or outside the jump, it will be treated as normal double coil processing.

The action of the subroutine timer (T192 to T199):

After the coil is driven, the action continues even if it jumps, and the output contact also operates.

If using the high-speed counter (HSC0 to HSC7) operation

After the coil is driven, the action continues even if it jumps, and the output contact also operates.

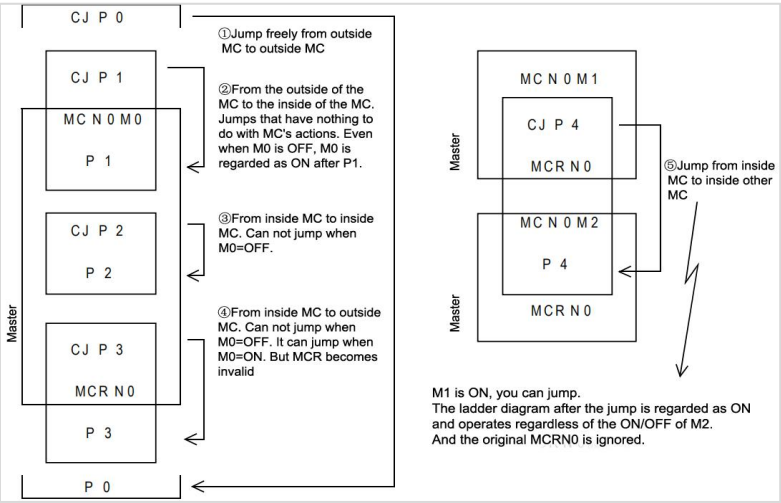
In the above program, if each input changes during the jump, the action of each coil is shown in the following table.

Content	Contact state before jump	Coil action in jump
Y,M,S (Y1, M1, S1)	X1, X2, X3 OFF	Y1, M1, S1 OFF
	X1, X2, X3 ON	Y1, M1, S1 ON
1ms, 10ms, 100ms timer (T0)	X4 OFF	Timer not working
	X4 ON	Timer interrupt (continue after X0 OFF)
Program timer (T192)	X5 OFF, X6 OFF	Timer not working, but the timer is reset when X13 is ON
	X5 OFF, X6 ON	Timing continues (contact action after X0 OFF)
Counter (C0)	X7 OFF, X10 OFF	Counting interrupt, but it is reset when X13 is ON
	X7 OFF, X10 ON	Count interruption (continue after X0 OFF)
application instructions (MOV)	X11 OFF	Single-cycle application instructions are not executed in the jump
	X11 ON	Multi-cycle application instructions are partially executable (such as high-speed pulse instructions)

3) The relationship between CJ instruction and MC to MCR jump

The relationship between the main control instruction and the jump instruction and the action content are as follows.

However, since the operation of ②, ④, and ⑤ will become complicated, please avoid using them.



4.2 Subroutine jump

4.2.1 CALL/Subroutine call

When the jump instruction is ON, the program with the specified pointer number in the same program file is executed.

-[CALL (P) (P)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(P)	Subroutine name	-	Pointer	POINTER

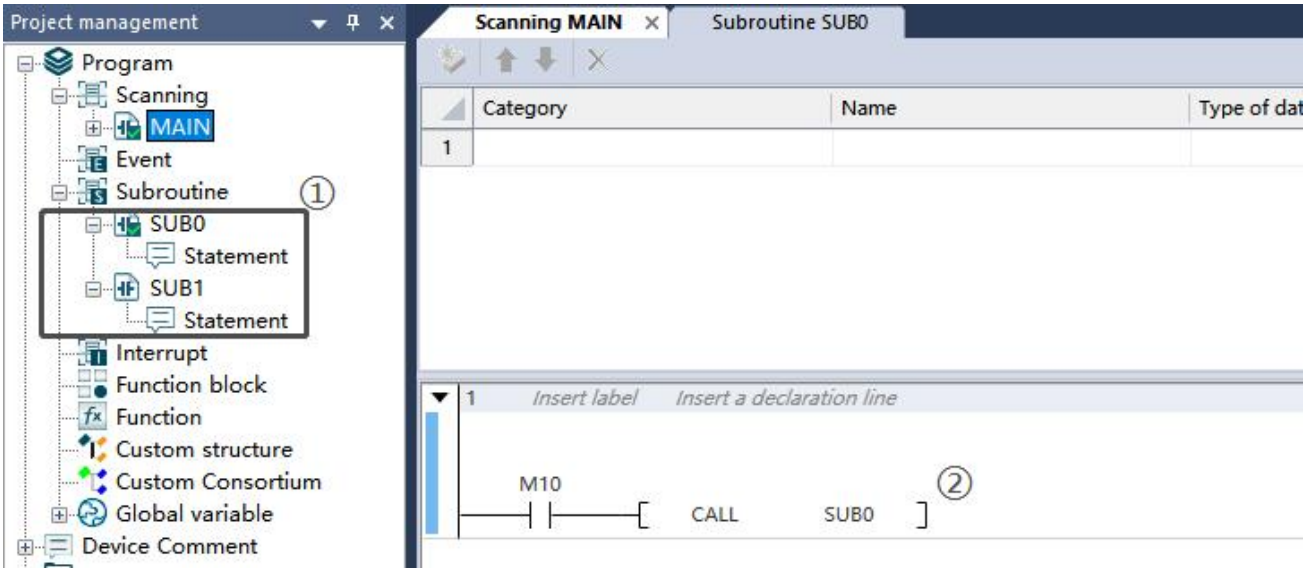
Device used

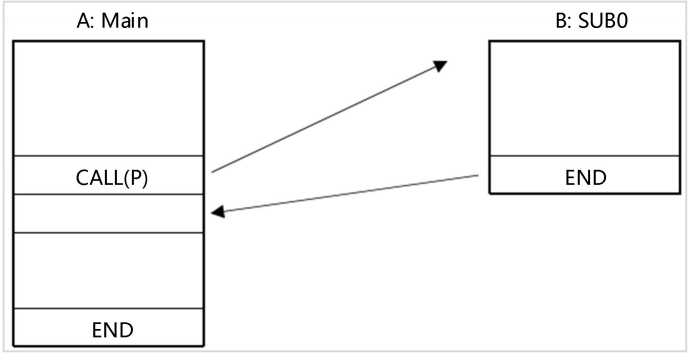
instruction	Parameter	Devices																							Index modification	Pulse extension	other
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP	Subroutine name
CALL	Parameter 1																									•	•

Parameter 1 can only use the subroutine name.

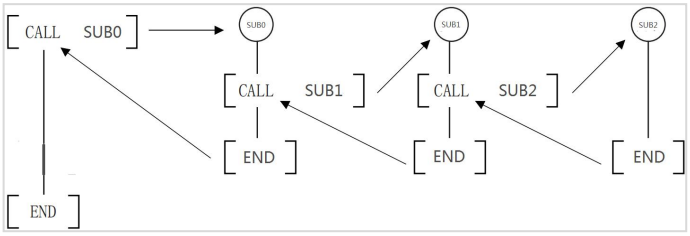
Features

When the CALL(P) instruction is executed, the subroutine of the pointer (P) will be executed. (P) can only write the name of the newly created subprogram, if the program name does not exist, the ladder diagram compilation fails.





CALL(P) instructions can be nested up to 32 levels.



✎Note:

• Multiple CALL(P) instructions can call the same subprogram, but subprograms with the same program name are not allowed.

• Use program timers in subroutines (the same applies to interrupt programs). This timer counts when the coil instruction or the END instruction is executed. If it reaches the timer setting value, the output contact will act when the coil instruction or END instruction is executed. Generally, the timer only counts when the coil instruction is executed, so if it is used in a subroutine that executes the coil instruction under certain conditions, it will not count.

• If the 1ms accumulative timer is used in a subroutine (the same in an interrupt program), when it reaches the set value, the output contact will act when the first coil instruction is executed (when the subroutine is executed), so be careful.

• The devices that are turned on in the subprogram (the same in the interrupt program) will be retained after the program ends. Therefore, these devices should be reset in the main program after the end of the program.

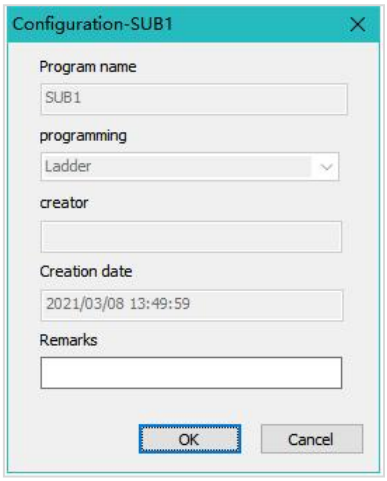
Error code

Error code	Content
4102H	CALL(P) instruction exceeds 32 levels of nesting structure

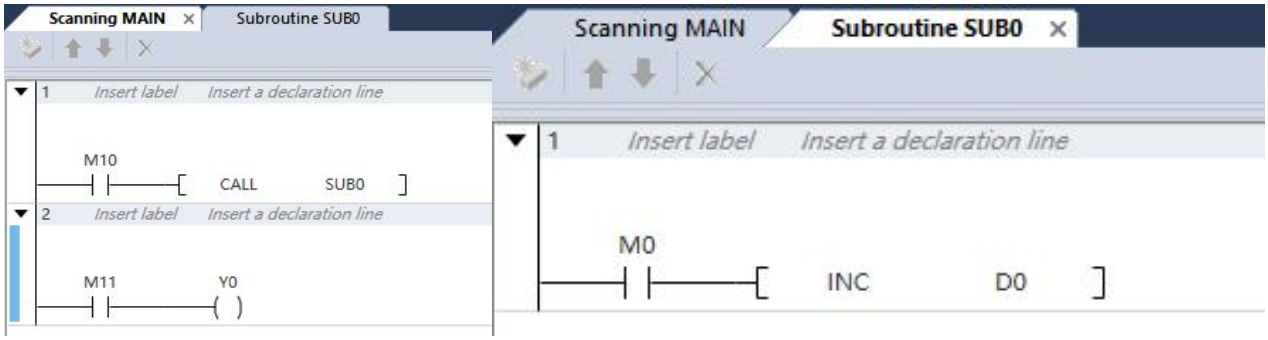
Example

1) New subroutine

Project management→Subroutine→ Scan→Right click to create

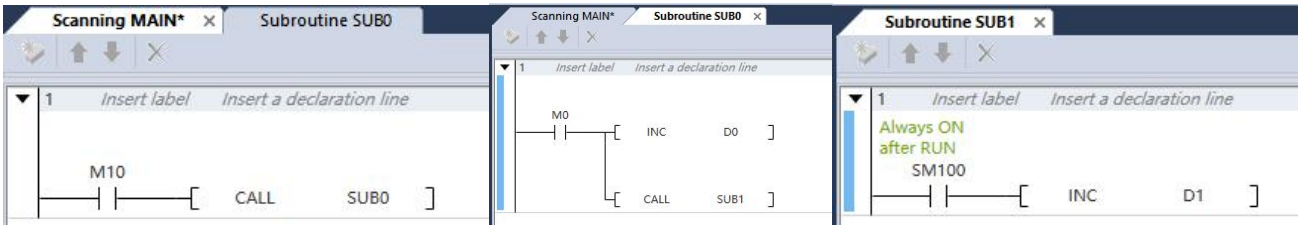


2) Subroutine call



In the scan program, turn on M10 to call the subroutine SUB0, execute the ladder diagram in the subroutine SUB0, until the END instruction of the subroutine is executed, return to the scan program MAIN to execute LD M11.

3) Subroutine nesting



In the above figure, the subroutine SUB0 is called in the scan program, and the subroutine SUB1 is called in SUB0. So when the scan program M10 is turned on, after the CALL instruction is executed, the subroutine SUB0 will be executed first.And after the CALL instruction of SUB0 is executed, SUB1 will be executed first. After executing the END instruction of SUB1, return to SUB0 for execution. After executing the END instruction of SUB0, return to the scan program MAIN. The program has only 2 levels of nesting, and the number of nesting levels cannot be greater than 32.

4.3 Interrupt disable, interrupt enable

4.3.1 DI and EI/Interrupt prohibited and allowed

The CPU module is usually interrupt disabled. This instruction can make the CPU module into the interrupt enabled state (EI instruction), and then become disabled again (DI instruction).

• DI: It is forbidden to interrupt program execution.

• EI: Release the interrupt prohibition state.

-[DI (s)]

-[EI]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(P)	Subroutine name	-	Pointer	POINTER

instruction	Parameter	Devices																										Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E			[D]	XXP
DI	Parameter 1																												●

Features

● DI

• Even if the execution interrupt condition is triggered in the program, prohibit the interrupt program execution before executing the EI instruction.

• When the PLC is powered on or after STOP, it will become the state after DI instruction is executed, and the interrupt program cannot be executed.

• The DI instruction can choose whether to use parameters. When there is no parameter, it means that all interrupt programs are prohibited. With parameters, according to the value in parameter s1, interrupt programs with this priority and lower priority are prohibited.

• The priority of the interrupt ranges from 0 to 2. The smaller the value, the higher the response priority of the interrupt. That is, the interrupt with priority 0 is the fastest to be responded.

• If there is no EI instruction before the DI instruction, the DI instruction is invalid.

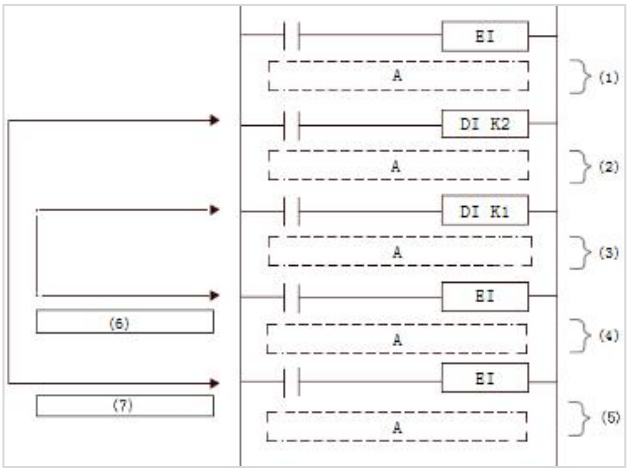
● EI

• Release the interrupt prohibition state when DI instruction is executed, and allow interrupt program to run.

• When the EI and DI instructions are not enabled, they all maintain the original enabled or forbidden interrupt program execution status. The currently disabled interrupt priority can be viewed in SD151.

SD151	Currently disabled interrupt priority	According to the interrupt prohibition instruction (DI instruction), the interrupt prohibition instruction (DI instruction) below the specified priority, and the interrupt enable instruction (EI instruction), the priority of the interrupt prohibition will be stored. 0: All priority interrupts are disabled (default); 1: Priority 1 and 2 interrupts are prohibited; 2: Priority 2 interrupt is prohibited; 3: All priority interrupts are allowed	R(read only)

- DI, EI nested structure



A: Sequence control program

(1) Interrupt allowable intervals of all priority levels;

(2) Interrupt forbidden zone below priority 2 (interrupt allowable zone above priority 1);

(3) Interrupt forbidden interval below priority 1 (interrupt allowable interval above priority 0);

(4) Interrupt prohibition zone below priority 2 (interrupt enable zone above priority 1);

(5) Interrupt allowable intervals of all priority levels;

(6) EI paired with [DI K1];

(7) EI paired with [DI K2].

• Interrupts (requests) that occur after the DI instruction are processed after the EI instruction is executed.

• When the DI instruction is executed multiple times and the priority of the argument is specified to be higher than the priority currently being prohibited, interrupts below the priority of the argument are disabled.

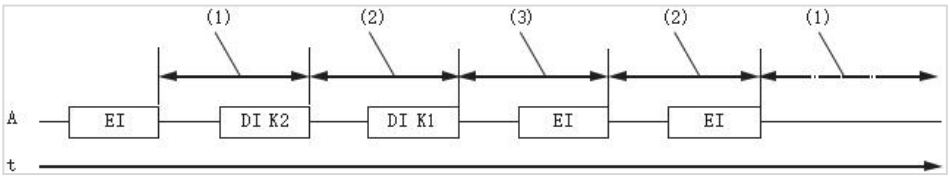
• When the DI instruction is executed multiple times and the priority of the argument is specified to be lower than the priority currently being disabled, the interrupt disable status will not be changed.

• The nesting of DI instructions can be up to 16 levels.

• The interrupt priority of the interrupt pointer can be set by the properties of the interrupt program. Refer to the description of the interrupt program for details.

• The interrupt prohibition interval when DI instruction and EI instruction are executed is as follows.

1) When the DI instruction is executed multiple times (when the interrupt with priority higher than the currently prohibited interrupt priority is prohibited and specified)



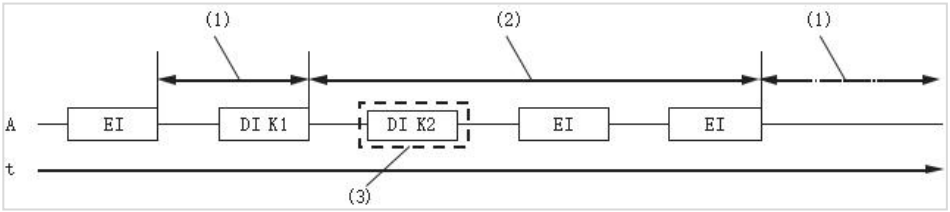
Scan execution type program

① Interrupt allowable intervals of all priority levels;

② Interrupt prohibition interval below priority 2 (interrupt allowable interval above priority 1);

③ Interrupt prohibition section below priority 1 (interrupt enable section above priority 0).

2) When the DI instruction is executed multiple times (when the interrupt priority is lower than the currently prohibited interrupt priority is prohibited and specified)



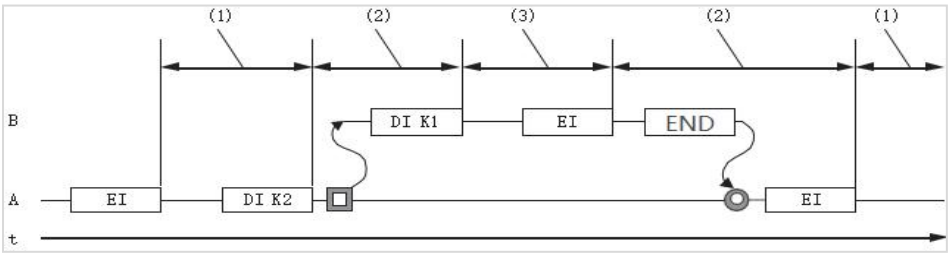
Scan execution type program

① Interrupt allowable intervals of all priority levels;

② Interrupt prohibited interval below priority 1 (interrupt allowable interval above priority 0);

③ The interrupts below priority 1 are already in the disabled state, so the interrupt disable priority will not be changed.

3) When DI instruction is executed through interrupt program



A: Scan execution type program

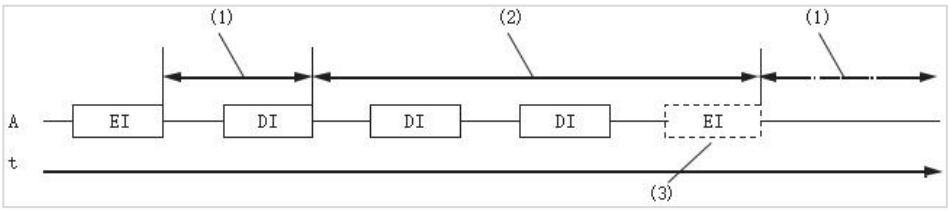
B: interrupt program

① Interrupt allowable intervals of all priority levels;

② Interrupt prohibited interval below priority 3 (interrupt allowable interval above priority 1);

③ Interrupt prohibition section below priority 2 (interrupt enable section above priority 0).

4) When only DI instructions without arguments are executed



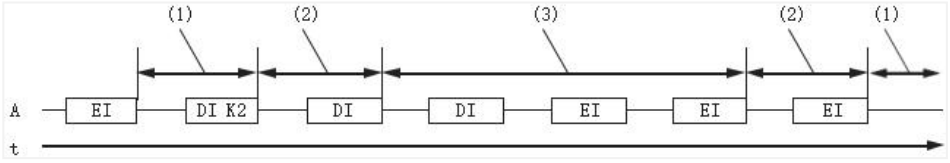
A: Scan execution type program

① Interrupt allowable intervals of all priority levels;

② Interrupt prohibition interval below priority 1 (all interrupt prohibition intervals);

③ Because the DI instruction with no argument is set to interrupt prohibition, by executing the EI instruction once, all priority interrupts are set to allow.

5) In the case of executing DI instructions with arguments and DI instructions without arguments (when executing in the order of DI instructions with arguments → DI instructions without arguments)



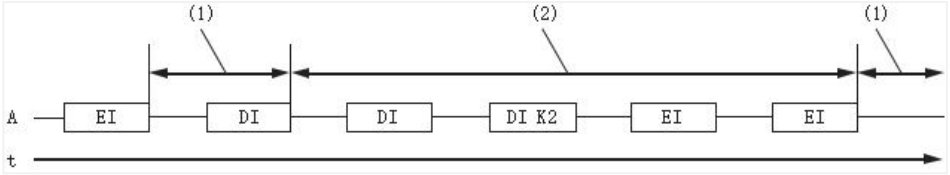
A: Scan execution type program

① Interrupt allowable intervals of all priority levels;

② Interrupt prohibition interval below priority 2 (interrupt allowable interval above priority 1);

③ Interrupt prohibition section below priority 1 (all interrupt prohibition sections).

6) In the case of executing DI instructions with arguments and DI instructions without arguments (in the case of execution in the order of DI instructions with no arguments → DI instructions with arguments)



A: Scan execution type program

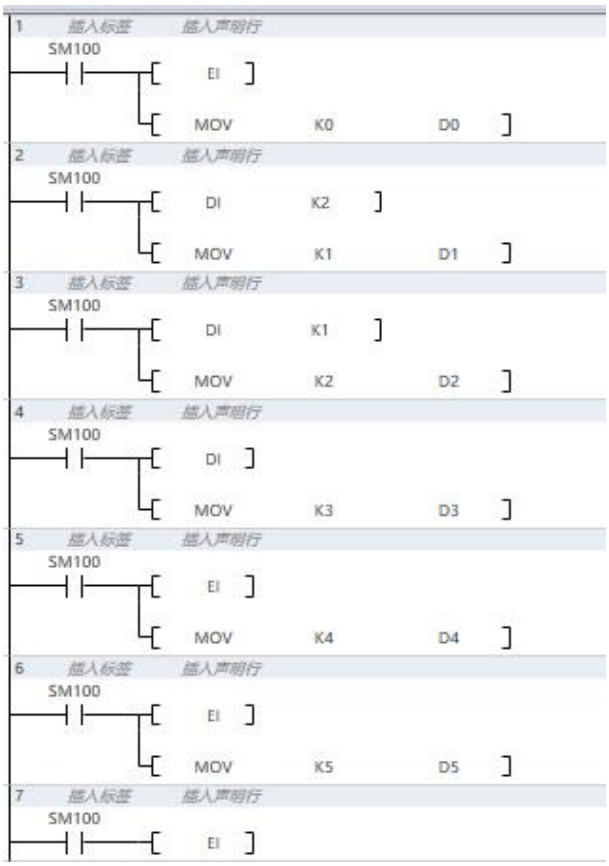
① Interrupt allowable intervals of all priority levels;

② Interrupt prohibition section below priority 1 (all interrupt prohibition sections).

Error code

Error code	Content
4085H	(S) read address exceeds the device range
4084H	The data set in (S) exceeds 0 to 2
4185H	When the nesting of DI instructions exceeds 16 levels

Example



All interrupt programs can be triggered

Can trigger interrupt programs of priority 0 and priority 1

Can trigger interrupts with priority 0

Cannot trigger any interrupts

Can trigger an interrupt program with a priority of 0

Can trigger interrupt programs with priority 0 and priority 1

4.3.2 SIMASK/Interrupt mask

Set interrupt pointer No. specified in (I) to the execution permission state/execution prohibition state according to the value of (s).

-[SIMASK (I) (s)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(I)	Interrupt program name	-	Program name	POINTER
(s)	Specify the enable/disable of interrupt	0: Allow. 1: Prohibited	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SIMASK	Parameter 1	Only support interrupt program name																								
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●	●	●	

Features

• The interrupt program of the interrupt program name specified in (I) is set to the execution permission state/execution prohibited state according to the data specified in (s).

• When (s) is 0: Interrupt program execution permission status

• When (s) is 1, the execution of the interrupt program is prohibited

• Regarding the interrupt program when the power is turned on or after STOP→RUN, all interrupt programs will be executed.

• After setting interrupt prohibition, the prohibition state will be saved even if the instruction is disconnected. To restore it, write 0 to (S), turn on the instruction again, or execute STOP→RUN.

• The interrupted execution permission status/execution prohibition status will be stored in SM or SD, details as following:

(1) External interrupt

Register	Content	Register	Content	Register	Content	Register	Content
SM352	X0 rising edge interrupt	SM356	X2 rising edge interrupt	SM360	X4 rising edge interrupt	SM364	X6 rising edge interrupt
SM353	X0 falling edge interrupt	SM357	X2 falling edge interrupt	SM361	X4 falling edge interrupt	SM365	X6 falling edge interrupt
SM354	X1 rising edge interrupt	SM358	X3 rising edge interrupt	SM362	X5 rising edge interrupt	SM366	X7 rising edge interrupt
SM355	X1 falling edge interrupt	SM359	X3 falling edge interrupt	SM363	X5 falling edge interrupt	SM367	X7 falling edge interrupt

(2) Timer interrupt

Register	Content
SD350 to SD356	Timer interrupt mask, each bit represents an interrupt, a total of 100

(3) High-speed counter interrupt

Register	Content
SD382 to SD388	high-speed counter interrupt mask, each bit represents an interrupt, a total of 100

Error code

Error code	Content
4084H	Data beyond 0 and 1 is input in the application instruction(s)
4085H	(S) in the read application instruction exceeds the device range
4189H	The SIMASK instruction specifies an interrupt program name that is not set

Example

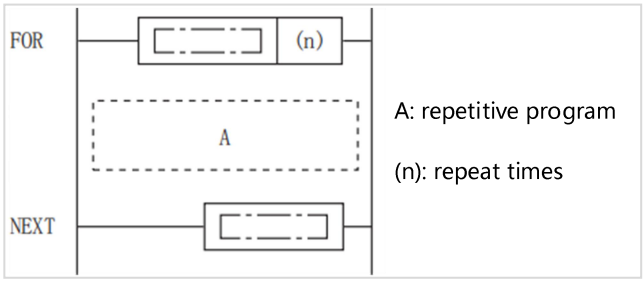


As shown in the figure: when M10 is turned on, the three interrupt programs of INT10, INT91 and INT70 are prohibited from running.

4.4 Cycle instructions

4.4.1 FOR to NEXT/Cycle

When the processing between the FOR to NEXT instruction is executed unconditionally (n) times, the next processing of the NEXT instruction will be performed.



Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(n)	Number of repetitions between FOR to NEXT instructions	1 to 32767	Signed BIN 16 bit	ANY16

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
FOR	Parameter 1											●	●	●	●	●	●	●	●	●			●	●	●	●	

Features

• When the processing between the FOR to NEXT instruction is executed unconditionally (n) times, the next processing of the NEXT instruction will be performed.

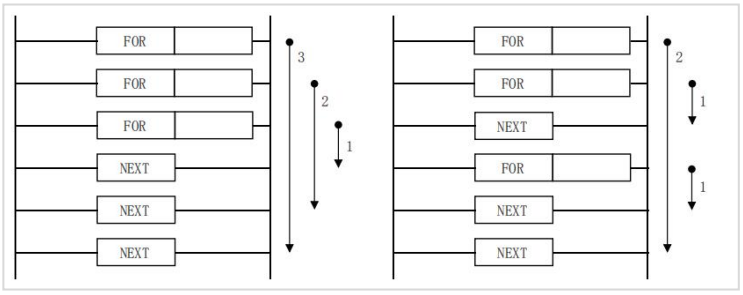
• (n) can be specified in the range of 1 to 32767. When specifying -32768 to 0, the same processing as (n)=1 will be performed.

• If you do not want to execute the processing between the FOR and NEXT instructions, use the CJ instruction to jump.

• The FOR instruction can be nested up to 5 levels.

✎Note:

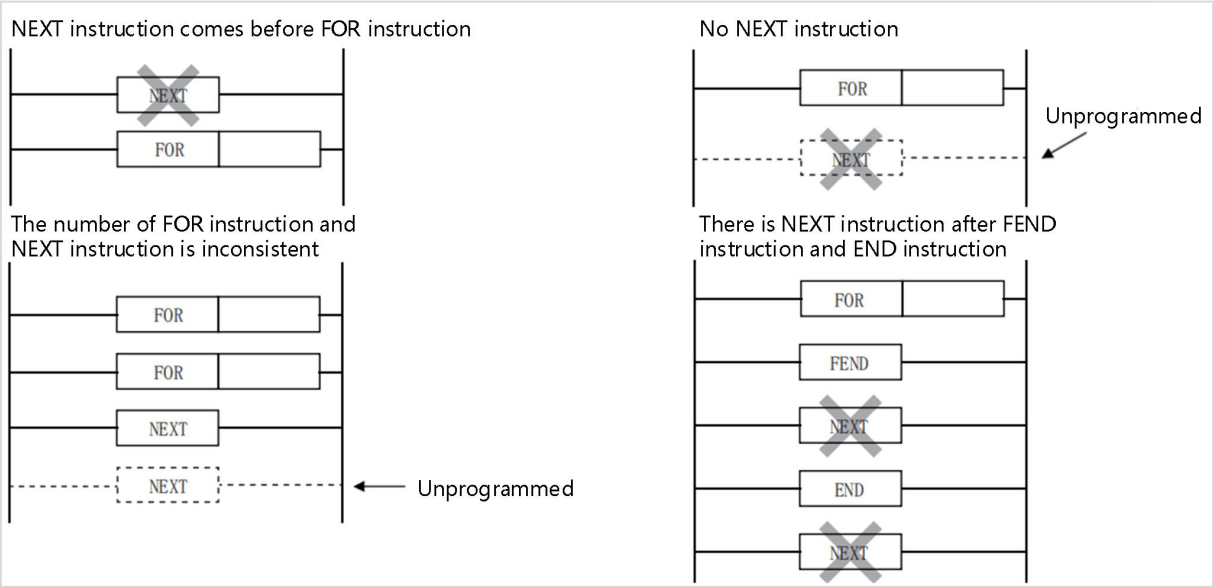
• In the case of FOR to NEXT instruction programming with nesting between FOR to NEXT instructions, up to 5 levels can be achieved.



• Do not use IRET, SRET, RET, FEND, END and other instructions to block between FOR to NEXT instructions.

• If the number of repetitions is too large, the cycle time (operation cycle) becomes longer and the watchdog timer error occurs, you need to change the watchdog timer time or reset the watchdog timer.

• The following program will become an error.

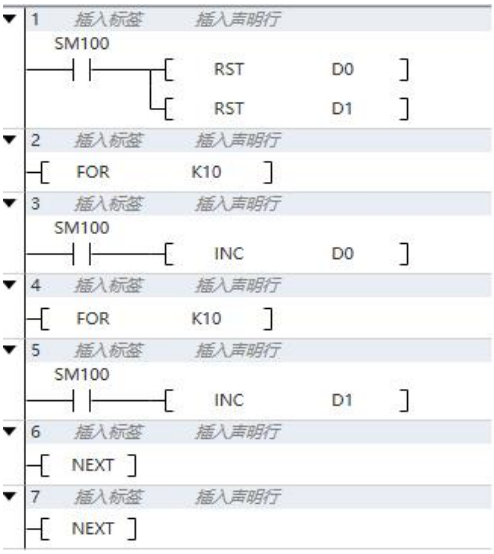


• If the FOR to NEXT instruction is repeatedly executed and ends midway, use the BREAK instruction.

Error code

Error code	Content
4085H	(s) read address exceeds the device range
4100H	When the nesting of FOR to NEXT instructions exceeds 5 levels or the number of FOR to NEXT does not correspond

Example



The program INC D0 will be executed 10 times, and INC D1 will be executed 100 times.

After execution, D0 will be equal to 10 and D1 will be equal to 100.

4.4.2 BREAK/Break cycle

When the processing between the FOR to NEXT instruction is executed unconditionally (n) times, the next processing of the NEXT instruction will be performed.

-[BREAK]

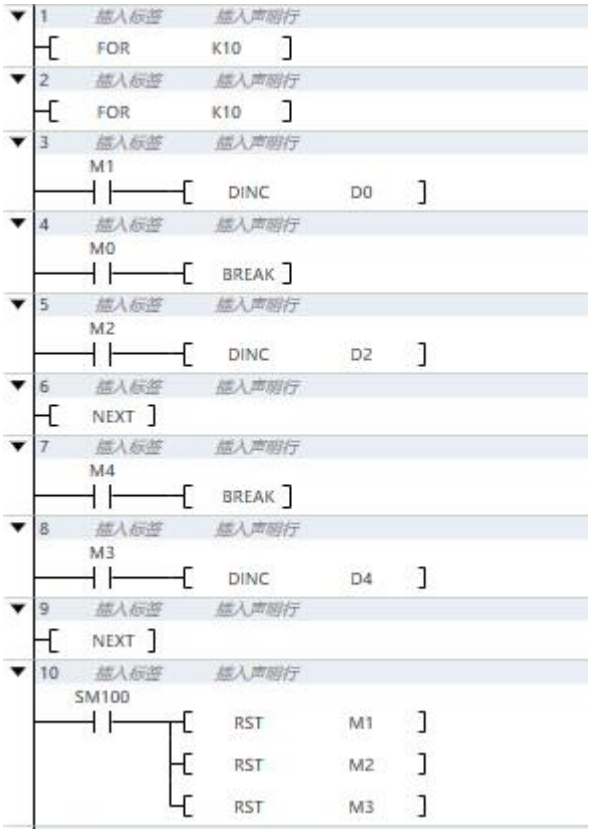
Features

• Forcibly end the repeated processing by FOR to NEXT instructions.

• This instruction can only be between FOR to NEXT, otherwise an operation error will be reported.

• The BREAK instruction can only jump out of the loop nesting structure where the instruction itself is located.

• When the contact is connected, the loop structure of the FOR to NEXT instruction where it is located is forced to end, as shown in the figure below.



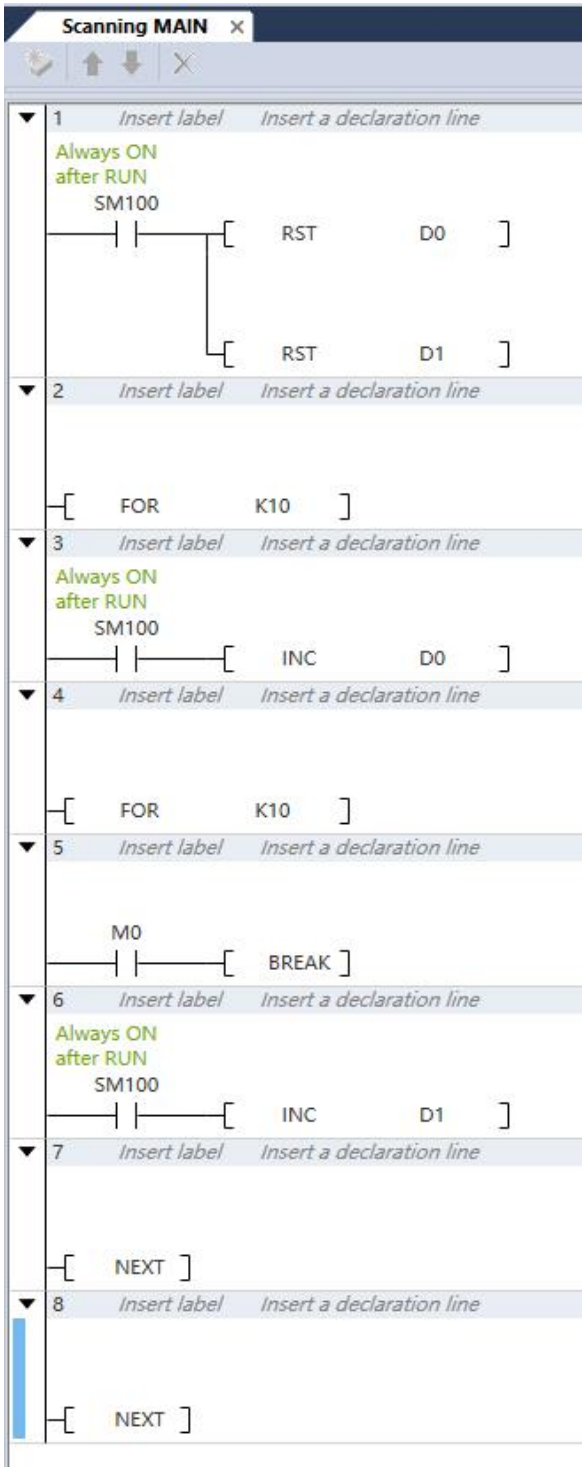
When M0 is set to ON, no matter how many cycles are left, it will directly jump to the seventh grid to execute the program.

When M4 is set to ON, no matter how many cycles are left, it will directly jump to the tenth grid to execute the program.

Error code

Error code	Content
4186H	BREAK instruction is not used between FOR to NEXT instructions

Example



The program INC D0 will be executed 10 times, and INC D1 will be executed 100 times.

When M0 is OFF, D0 will be equal to 10 and D1 will be equal to 100 after execution.

When M0 is ON, the BREAK instruction is executed, and the current loop is exited. The INC D1 instruction will not be executed, and the result D1=0.

4.5 Watchdog reset

4.5.1 WDT/Watchdog timer

The watchdog timer is reset by the program.

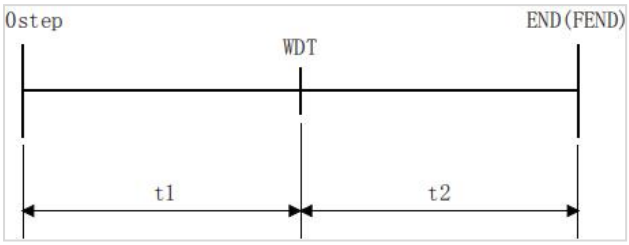
-[WDT]

Features

• Reset the watchdog timer through the program.

• Use when the scan time exceeds the set value of the watchdog timer depending on conditions.

• For t1 from step 0 to WDT instruction, and from WDT instruction to END instruction, do not exceed the set value of the watchdog timer.



• WDT instruction can be used more than twice in one scan.

✎Note:

• The watchdog timeout time can be set in the special register SD122. The default is 200ms.

• Use the special relay SM122 to control whether to turn on the watchdog timer function. The WDT instruction will be invalid after closing.



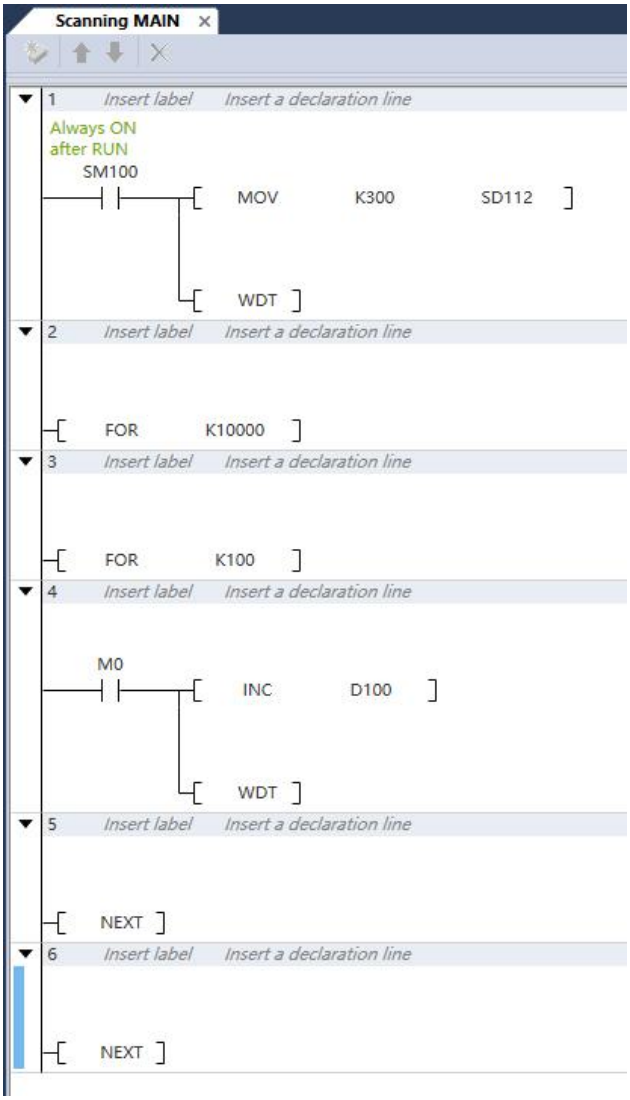
① The watchdog timer time is set to 300ms;

② Refresh the watchdog timer.

Error code

There is no operation error.

Example



The FOR to NXET instruction loop takes a long scan period for many times, which may exceed the set watchdog timer 300ms, causing the PLC to report an error and cannot continue to run. After turning on M0, the WDT instruction will run, and the watchdog timer is updated every cycle , So that it will not report an error to execute the program normally.

5 Timer and counter output instructions

5.1 Timer output instruction

5.1.1 OUT T/Timer output

When the calculation result before the OUT instruction is ON, the coil of the timer/retentive timer specified in (d) will be ON and measurement will be performed until the set value is reached. If the time limit expires, the normally open contact will conduct and the normally closed contact will become non-conductive.

-[OUT (d) (value)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	Timer device number	-	Counter	ANY
(value)	Timer setting value	0 to 32767	Unsigned BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
OUT T	Parameter 1															●										
	Parameter 2											●	●	●	●		●	●	●	●			●	●		

Features

When the operation result before the OUT instruction is ON, the coil of the timer specified in (d) will be ON and measurement will be performed until the set value is reached. If the count reaches (current value ≥ set value), the normally open contact will be conductive, and the normally closed contact will become non-conductive.

When the operation result before the OUT instruction changes from ON to OFF, the situation is as follows.

Timer type	Timer coil	The current value of the timer	Before the time limit		After the time limit	
			Normally open contact	Normally closed contact	Normally open contact	Normally closed contact
Timer	OFF	0	Non-conductive	Conduction	Non-conductive	Conduction
Cumulative timer	OFF	Keep current value	Non-conductive	Conduction	Conduction	Non-conductive

• After the time limit expires, clear the current value of the accumulative timer and turn off the contact with the RST instruction.

• When the setting value is 0, the time limit will expire when the OUT instruction is executed.

• While the OUT T instruction is ON, if the OUT T instruction is skipped by the CJ instruction, etc., the current value update and contact ON/OFF will not be performed.

• If the same OUT T instruction is executed more than twice in the same scan, the current value will be updated according to the number of executions.

• Description of each timer:

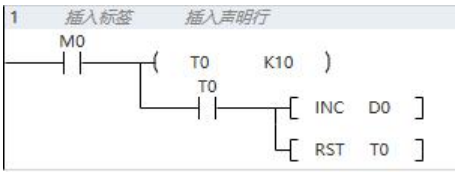
Device number	Timer specifications	Device number	Timer specifications
T0 to T191	100ms timer	T246 to T249	1ms accumulative timer
T192 to T199	100ms subroutine timer (used in the subroutine, even if the subroutine is not called, it will still be updated)	T250 to T255	10ms cumulative timer
		T256 to T383	1ms timer
T200 to T245	10ms timer	T384 to T511	0.1ms timer

Error code

Error code	Content
4084H	The parameter setting in (value) is out of range

Example

Using timing, D0 increases by 1 after every 1S:



5.2 Counter output instructions

5.2.1 OUT C/Counter output

16-bit counter instruction: When the operation result before the OUT instruction changes from OFF to ON, the current value of the counter specified in (d) will be +1. If the count reaches, the normally open contact will be turned on and the normally closed contact will become Non-conductive.

-[OUT (d) (value)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	Counter device number	-	Counter	ANY
(value)	Counter setting value	0 to 32767	Unsigned BIN 16 bit	ANY_INT

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
OUT C	Parameter 1																●										
	Parameter 2											●	●	●	●	●		●	●	●			●	●	●		

Features

• When the calculation result before the OUT instruction changes from OFF to ON, the current value (count value) of the counter specified in (d) will be +1. If the count reaches (current value ≥ set value), the normally open contact will be turned on , The normally closed contact becomes non-conductive.

• If the calculation result is ON, no counting is performed. (Counting input does not need to be pulsed.)

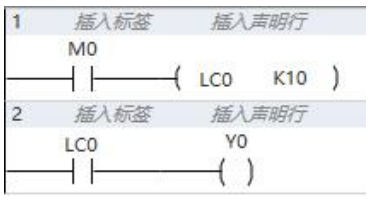
• After the count is reached, the count value and the state of the contact do not change before the RST instruction is executed.

• When the setting value is 0, the processing is the same as when it is 1.

Error code

Error code	Content
4084H	The parameter setting in (value) is out of range
4085H	The (value) parameter exceeds the device range

Example



Every time M0 changes from OFF→ON, C0 will increase by 1. When the value of C0 is added to K10, the normally open contact of C0 is closed and Y0 is output. At this time, M0 continues from OFF→ON, and the value of C0 will not change anymore.

The contact of C0 can only be turned OFF by RST/ZRST instruction and communication.

5.2.2 OUT LC instruction/Long counter output

32-bit counter instruction: When the operation result before the OUT instruction changes from OFF to ON, the current value of the long counter specified in (d) will be +1. If counted, the normally open contact will be turned on and the normally closed contact will change It is non-conductive.

-[OUT (d) (value)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	Long counter device number	-	Counter	ANY
(value)	Long counter setting value	0 to 4294967295	Unsigned BIN 32 bit	ANY_INT

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
OUT LC	Parameter 1																				●						
	Parameter 2											●	●	●	●	●		●	●	●			●	●	●		

Features

• When the calculation result before the OUT instruction changes from OFF to ON, the current value (count value) of the long counter specified in (d) will be +1. If the count reaches (current value ≥ set value), the normally open contact will turn on On, the normally closed contact becomes non-conductive.

• If the calculation result is ON, no counting is performed. (Counting input does not need to be pulsed.)

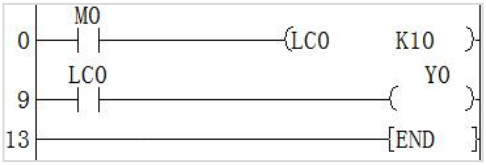
• After the count is reached, the count value and contact status will not change before the RST instruction or ZRST instruction is executed.

• When the setting value is 0, the processing is the same as when it is 1.

Error code

Error code	Content
4085H	The (value) parameter exceeds the device range

Example



Each time M0 changes from OFF to ON, LC0 will increase by 1. When the value of LC0 is added to K10, the normally open contact of LC0 is closed and Y0 is output. At this time, M0 continues from OFF→ON, and the value of LC0 will not change anymore.

The contact of LC0 can only be turned OFF by RST/ZRST instruction and communication.

6 Basic instructions

6.1 Transfer comparison instruction

6.1.1 MOV/16-bit transmission

MOV(P)

Transfer the BIN 16-bit data of the device specified in (s) to the device specified in (d).

-[MOV (s) (d)]

Content, range and data type

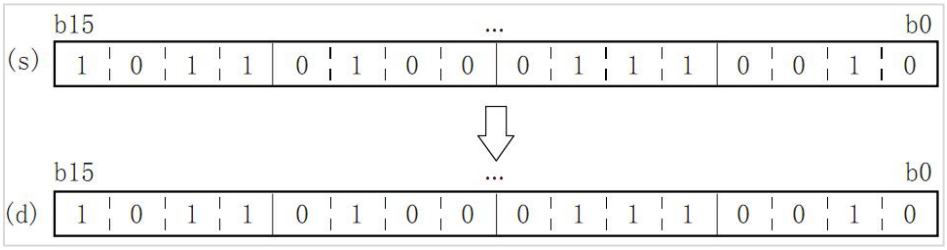
Parameter	Content	Range	Data type	Data type (label)
(s)	Transmit source data or the device number stored data	-32768 to 32767	Signed BIN16	ANY16_S
(d)	Transmit destination device number	-	Signed BIN16	ANY16_S

Device used

Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MOV	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Features

• Transfer the BIN 16-bit data specified in (s) to the device specified in (d).



Error code

Error code	Content
4085H	The output result of (s) in read application instruction exceeds the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example



When M0 is set, the value of D0 is transferred to the value of D2: (D0)→(D2).

6.1.2 DMOV/32-bit transmission

DMOV(P)

Transfer the BIN 32-bit data of the device specified in (s) to the device specified in (d).

-[DMOV (s) (d)]

Content, range and data type

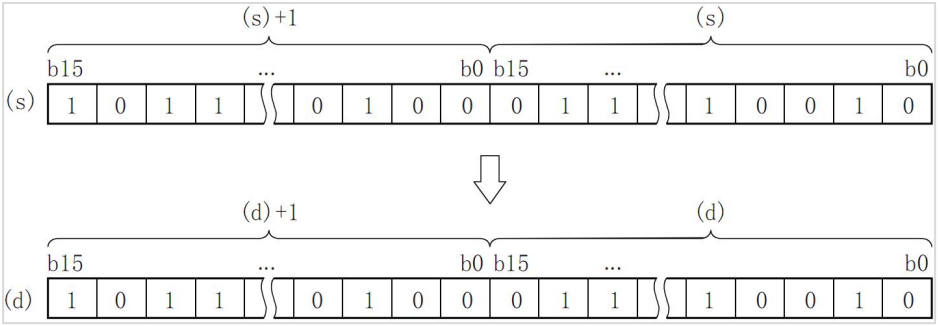
Parameter	Content	Range	Data type	Data type (label)
(s)	Transmit source data or the device number stored data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(d)	Transmit destination device number	-	Signed BIN32	ANY32_S

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DMOV	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●

Features

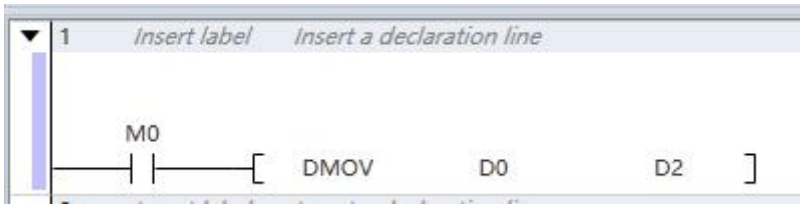
Transfer the BIN 16-bit data specified in (s) to the device specified in (d).



Error code

Error code	Content
4085H	The output result of (s) in read application instruction exceeds the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example



When M0 is set, the value of (D1, D0) is transferred to the value of (D3, D2): (D1, D0) → (D3, D2).

6.1.3 BMOV/Batch transmission

BMOV(P)

The (n) point BIN 16-bit data starting from the device specified in (s) is sequentially transmitted to the device specified in (d).

-[BMOV (s) (d) (n)]

Content, range and data type

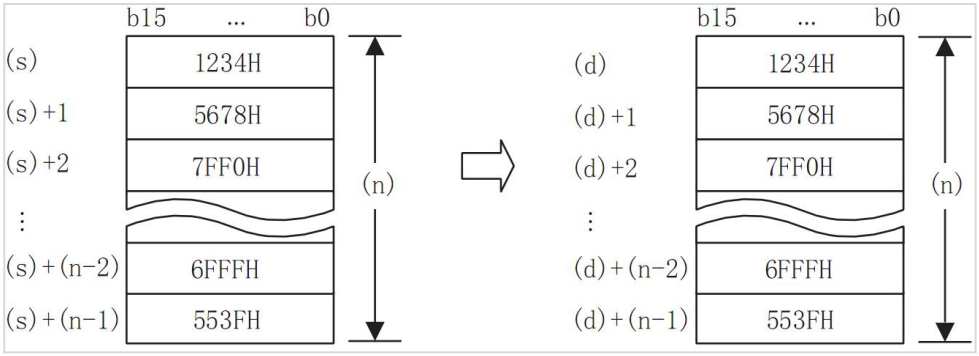
Parameter	Content	Range	Data type	Data type (label)
(s)	The start device that stores the transmission data	-	Signed BIN16	ANY16_S
(d)	The start device that transmit target	-	Signed BIN16	ANY16_S
(n)	Number of transmission	1 ≤ n ≤ 512	Signed BIN16	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BMOV	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Batch transfer the BIN 16-bit data of point (n) starting from the device specified in (s) to the device specified in (d).



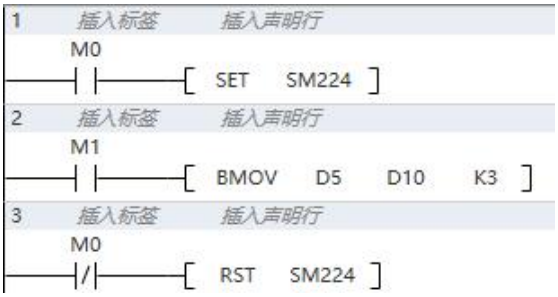
When the device number exceeds the range, it will be transferred within the allowable range.

By controlling the direction reversal flag (SM224) of the BMOV instruction, the BIN 16-bit data at point (n) starting from the device specified in (d) can be batch transferred to the device specified in (s).

Error code

Error code	Content
4084H	In application instruction (n) input the data exceeds the specified range
4085H	The output results of (s) and (n) in read application instruction exceed the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example



When M0 is set, set M1, then (D5)→(D10); (D6)→(D11); (D7)→(D12);

When M0 is reset, set M1, then (D10)→(D5); (D11)→(D6); (D12)→(D7).

6.1.4 FMOV/16-bit multicast

FMOV(P)

Transfer the BIN 16-bit data of the device specified in (s1) to the device specified in (d) at (n) points (that is, transfer the same data to multiple addresses).

-[FMOV (s) (d) (n)]

Content, range and data type

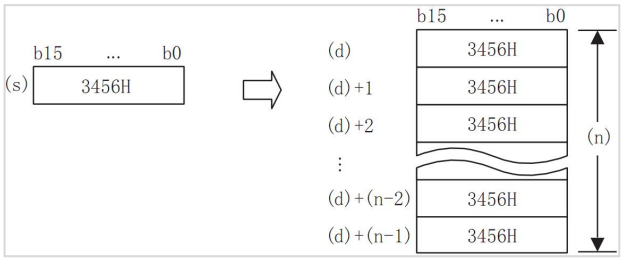
Parameter	Content	Range	Data type	Data type (label)
(s)	The start device that stores the transmission data	-32768 to 32767	Signed BIN16	ANY16_S
(d)	The start device that transmit target	-	Signed BIN16	ANY16_S
(n)	Number of transmission	[K1 ≤ n ≤ 512]	Signed BIN16	ANY16_S

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
FMOV	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

The same data as the BIN 16-bit data of the device specified in (s) is transferred to the device specified in (d) at (n) points.



When the number specified in (n) exceeds the device number range, transfer is performed within the allowable range.

When a constant (K) is specified for the transmission source (s), it will be automatically converted to BIN.

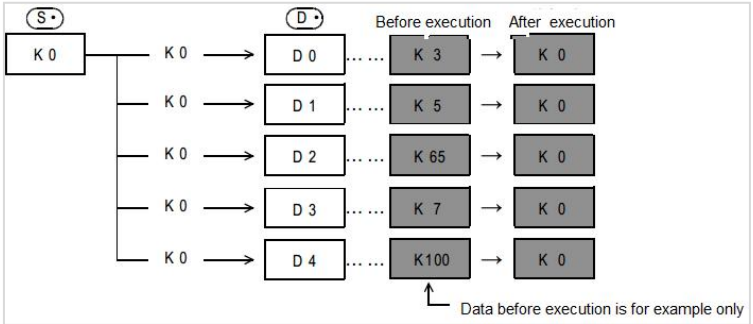
Error code

Error code	Content
4084H	(s) and(n) input the data In application instruction exceed the specified range
4085H	The output results of (s) and (n) in read application instruction exceed the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example



When M0 is set, the value of D0 to D4 is set to 0.



6.1.5 DFMOV/ 32-bit multicast

DFMOV(P)

Transfer the BIN 32-bit data of the device specified in (s1) to the device specified in (d) at (n) points (that is, transfer the same data to multiple addresses).

-[FMOV (s) (d) (n)]

Content, range and data type

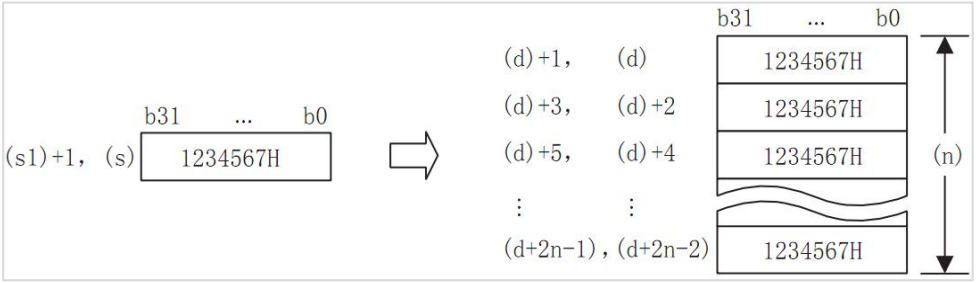
Parameter	Content	Range	Data type	Data type (label)
(s)	Transfer data or start device storing transfer data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(d)	Start device of transfer destination	-	Signed BIN32	ANY32_S
(n)	Number of transfers	[1≤n≤512]	Signed BIN32	ANY32_S

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DFMOV	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●				●	●	●	●

Features

The same data as the BIN 32-bit data of the device specified in (s) is transferred to the device specified in (d) at (n) points.



When the number specified in (n) exceeds the device number range, transfer is performed within the allowable range.

When a constant (K) is specified for the transmission source (s), it will be automatically converted to BIN.

Error code

Error code	Content
4084H	(s) and (n) input the data In application instruction exceed the specified range
4085H	The output results of (s) and (n) in read application instruction exceed the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example



When M0 is set, the value of (D1, D0), (D3, D2), (D5, D4), (D7, D6), (D9, D8) is set to 0.

6.1.6 SMOV/Bit shift

SMOV(P)

A instruction for distributing and synthesizing data in units of digits (4 bits).

-[SMOV (s) (n1) (n2) (d) (n3)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The word device number that stores the data whose bit is to be moved		Signed BIN16	ANY16_S
(n1)	Transfer destination device number	1 to 4	Signed BIN16	ANY16_S
(n2)	The number of digits to move	1 to 4	Signed BIN16	ANY16_S
(d)	The word device number that stores data for bit shifting		Signed BIN16	ANY16_S
(n3)	The starting position of the moving target	1 to 4	Signed BIN16	ANY16_S

Device used

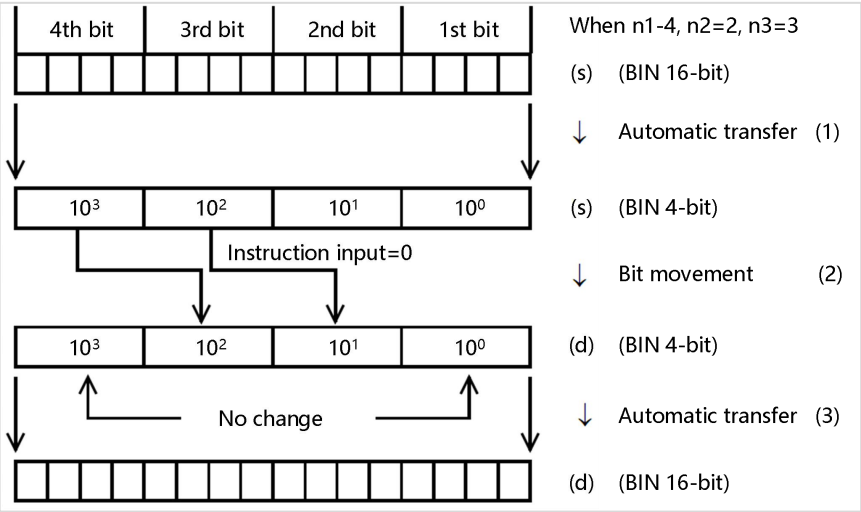
instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SMOV	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 4												●	●	●	●	●	●	●	●						●	●
	Parameter 5											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

The data is distributed/combined in units of digits (4 bits). The contents of the transmission source (s) and the transmission destination (d) are converted into 4-digit BCD (0000 to 9999), and the (n1) bits are transferred to the lower (n2) bits and the (n3) bits of the transmission destination (d) (combined) After reaching the starting position, it is converted to BIN and stored in the transfer destination (d).

When the instruction input is OFF, the transfer destination (d) does not change.

When the instruction input is ON, the data of the transmission source (s) and the number of digits other than the transmission specification of the transmission destination (d) do not change.



(1) Perform BIN→BCD conversion on

(2) Transfer (synthesize) the (n1)th bit to the lower (n2), (d), (n3)th bit to the (n2)th bit counted from the previous. (D), the first and fourth digits start from (s), and the transmission will not be affected.

(3) Convert the synthesized data (BCD) into BIN and store it in (d).

Extended function

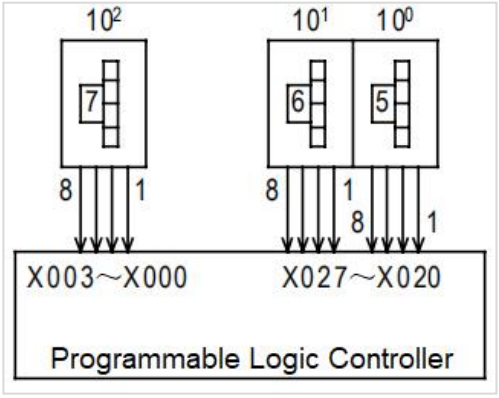
If the SMOV instruction is executed after SM168 is turned ON, the BIN→BCD conversion will not be performed. The bit shift is performed in 4-bit units.

Error code

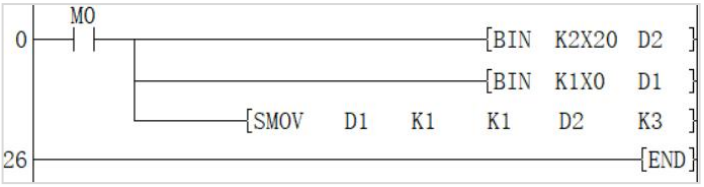
Error code	Content
4084H	(n1), (n2) and (n3) input data that exceed the specified range in the application instruction or does not satisfy the relationship of n2≤n1 and n2≤n3.
4085H	The output result of (s), (n1) (n2), (d) and (n3) in the read application instruction exceeds the device range
4086H	The output result of (d) in write application instructions exceeds the device range

Example

After synthesizing the data of the 3-digit digital switch, it is stored in D2 in binary.



Combine data of 3 digital switches connected to non-continuous input terminals.



When M0 is set,

(X020 to X027) BCD 2 digits → D 2 (binary);

(X000 to X003) BCD 1 digit → D 1 (binary);

Store the 1 digit of D1 into the 3 digit of D2, and synthesize a 3-digit value.

6.1.7 CML/16-bit invert transmission

CML(P)

After the BIN 16-bit data specified in (s) is inverted bit by bit, the result is transferred to the device specified in (d).

-[CML (s) (d)]

Content, range and data type

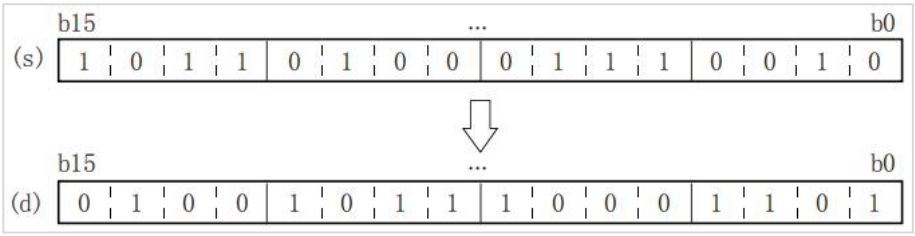
Parameter	Content	Range	Data type	Data type (label)
(s)	Inverted data or the device number that stores data	-32768 to 32767	Signed BIN16	ANY16_S
(d)	The device number that stores the inversion result	-	Signed BIN16	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	S M	T(bit)	C(bit)	LC(bit)	HSC(bit)	D. b	KnX	KnY	KnM	KnS	T	C	D	R	S D	L C	HS C	K	H	E	[D]	XXP
CML	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Features

After inverting the BIN 16-bit data specified in (s) bit by bit, the result is transferred to the device specified in (d).



When the number of digits of the device with the specified digit is 4 points, other digits are not affected.

Error code

Error code	Content
4085H	The output result of (s) in read application instruction exceeds the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example

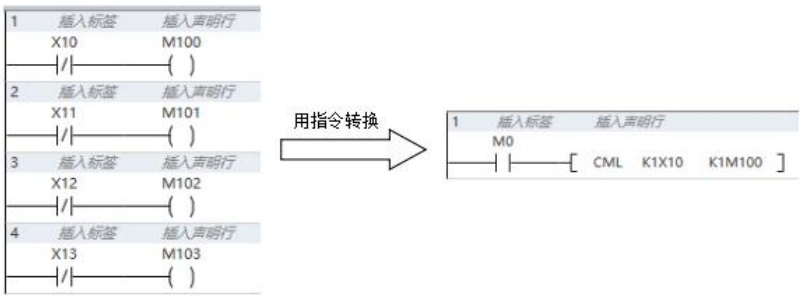
Example 1:



When M0 is set, the value of D0 is inverted and transferred to the value of D2.

Example 2:

Invert input acquisition:



6.1.8 DCML/32-bit invert transmission

DCML(P)

After the BIN 32-bit data specified in (s) is inverted bit by bit, the result is transferred to the device specified in (d).

-[CML (s) (d)]

Content, range and data type

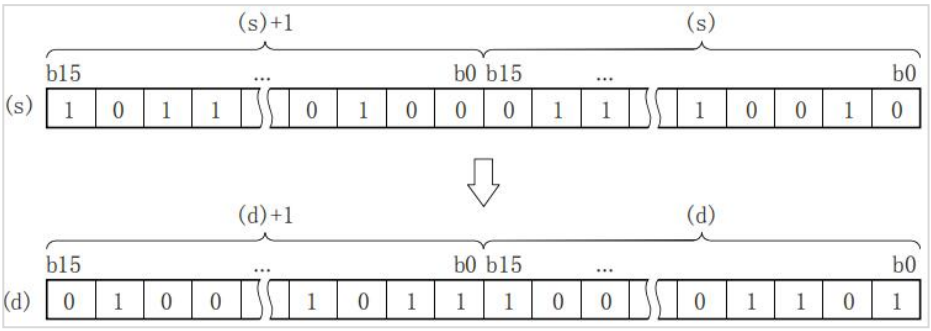
Parameter	Content	Range	Data type	Data type (label)
(s)	Inverted data or the device number that stores data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(d)	The device number that stores the inversion result	-	Signed BIN32	ANY32_S

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DCML	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●

Features

After inverting the BIN 32-bit data specified in (s) bit by bit, the result is transferred to the device specified in (d).



When the number of digits of the device with the specified digit is 4 points, other digits are not affected.

Error code

Error code	Content
4085H	The output result of (s) in read application instruction exceeds the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example



When M0 is set, the value of (D1, D0) is reversed and transferred to the value of (D3, D2).

6.1.9 CMP/16-bit data comparison output

CMP(P)

Compare the BIN 16-bit data of the device specified in (s1) and (s2).

-[CML (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparison value data or the device storing the comparison value data	-32768 to +32767	Signed BIN16	ANY16_S
(s2)	Comparison source data or the device storing the comparison source data	-32768 to 32767	Signed BIN16	ANY16_S
(d)	Start bit device for output comparison result		Bit	ANYBIT_ARRAY

Device used

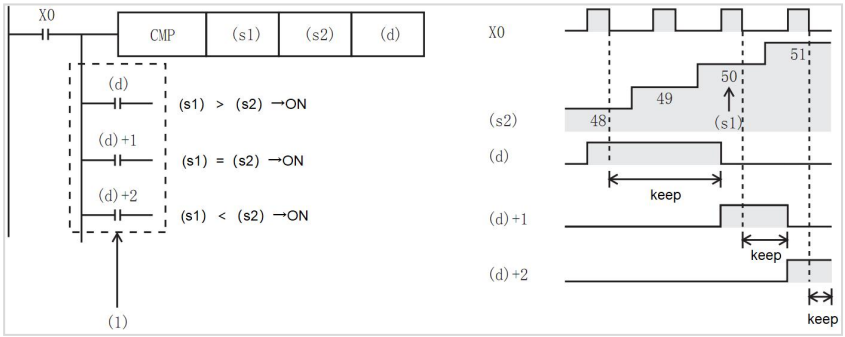
instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CMP	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3		●	●	●	●					●															●	

Features

Compare the BIN 16-bit data of the device specified in (s1) with the BIN 16-bit data of the device specified in (s2). According to the result (less than, consistent, greater than), (d), (d)+1, (d) One of)+2 will turn ON.

(s1) and (s2) are handled as BIN values within the above setting data range.

Use algebraic methods for size comparison.



(1): Even if the instruction input is OFF and the CMP instruction is not executed, (d) to (d)+2 will keep the state before the instruction input changed from ON to OFF.

✎Note:

Occupy the device specified in 3 points (d) at the beginning, please be careful not to overlap with the device used for other control.

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in read application instruction exceed the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example



When M0 is set, compare the values of D0 and D2:

If (D0)> (D2) then Y0 is ON.

If (D0) = (D2) then Y1 is ON. If (D0) <(D2) then Y2 is ON.

6.1.10 DCMP/32-bit data comparison output

DCMP(P)

Compare the BIN 32-bit data of the device specified in (s1) and (s2).

-[DCML (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparison value data or the device storing the comparison value data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(s2)	Comparison source data or the device storing the comparison source data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(d)	Start bit device for output comparison result		Bit	ANYBIT_ARRAY

Device used

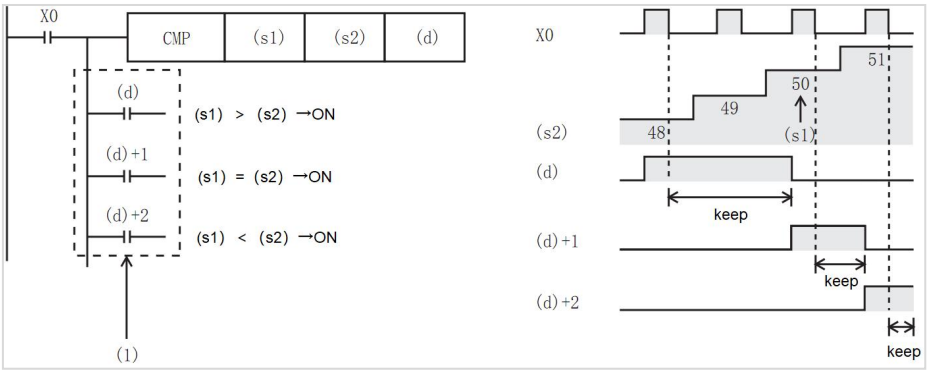
instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DCMP	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 3		●	●	●	●					●															●	

Features

• Compare the BIN 16-bit data of the device specified in (s1) with the BIN 16-bit data of the device specified in (s2). According to the result (less than, consistent, greater than), (d), (d)+1, (d) One of)+2 will turn ON.

• (s1) and (s2) are handled as BIN values within the above setting data range.

• Use algebraic methods for size comparison.



(1): Even if the instruction input is OFF, the DCMP instruction is not executed, (d) to (d)+2 will keep the state before the instruction input changed from ON to OFF.

✎Note:

Occupy the device specified in 3 points (d) at the beginning. Please be careful not to overlap with other control devices.

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in read application instruction exceed the device range
4086H	The output result of (d) in write application instruction exceeds the device range

Example



When M0 is set, compare the values of (D1, D0) and (D3, D2):

If (D1, D0)> (D3, D2) then Y0 is ON.

If (D1, D0) = (D3, D2) then Y1 is ON.

If (D1, D0) <(D3, D2) then Y2 is ON.

6.1.11 XCH/16-bit data exchange

XCH(P)

Exchange the BIN 16-bit data of (d1) and (d2).

-[XCH (d1) (d2)]

Content, range and data type

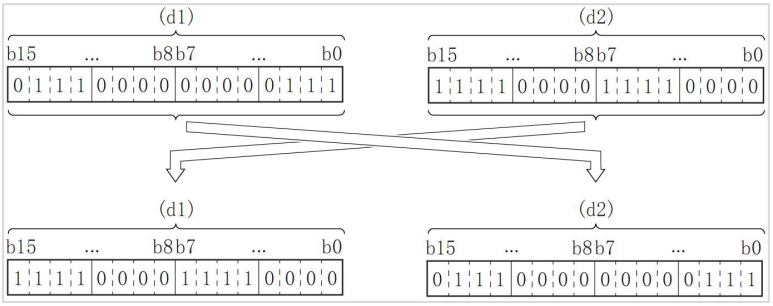
Parameter	Content	Range	Data type	Data type (label)
(d1)	The start device that stores the exchange data	-32768 to 32767	Signed BIN16	ANY16_S
(d2)	The start device that stores the exchange data	-32768 to 32767	Signed BIN16	ANY16_S

Device used

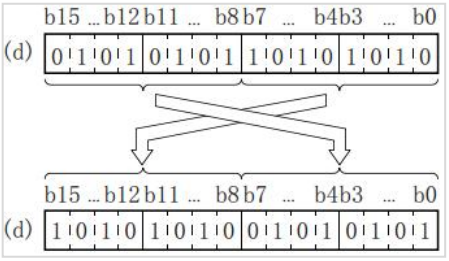
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
XCH	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Features

• Exchange the BIN 16-bit data of (d1) and (d2).



• When executing instructions with SM160 ON, if the device numbers of (d1) and (d2) are the same. Exchange the upper 8 bits (byte) and lower 8 bits (byte) of the word device.



✎**Note:** If continuous execution instructions are used, conversion will be performed every operation cycle.

Error code

Error code	Content
4084H	In exchange mode, the devices in (d1) and (d2) are different
4085H	The output results of (d1) and (d2) in the read application instruction exceed the device range
4086H	The output results of (d1) and (d2) in the writing application instruction exceed the device range

Example

When M0 is reset, set M1: the value of D0 and the value of D2 are exchanged.



When M0 is set, M1 is set: the upper 8 bits (bytes) and lower 8 bits (bytes) of D0 are exchanged with each other.



6.1.12 DXCH/32-bit data exchange

DXCH(P)

Exchange (d1) and (d2) BIN 32-bit data.

-[DXCH (d1) (d2)]

Content, range and data type

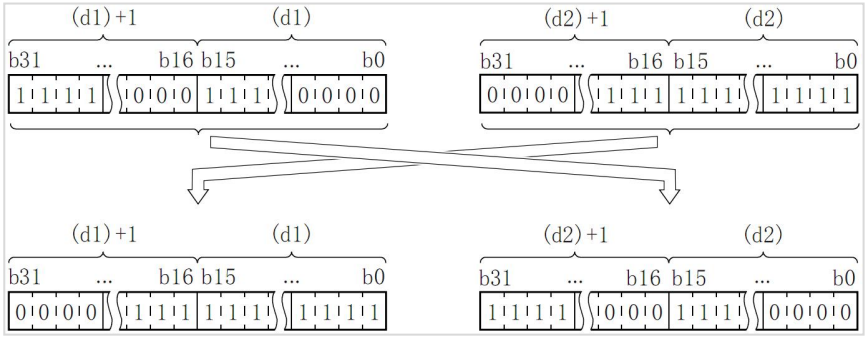
Parameter	Content	Range	Data type	Data type (label)
(d1)	The start device that stores the exchange data	-2147483647 to 2147483647	Signed BIN32	ANY32_S
(d2)	The start device that stores the exchange data	-2147483647 to 2147483647	Signed BIN32	ANY32_S

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DXCH	Parameter 1												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●

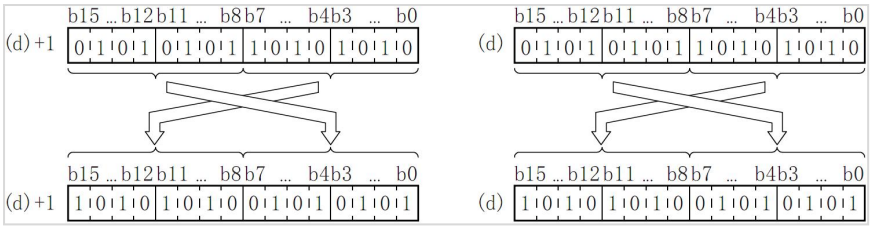
Features

• Exchange the BIN 32-bit data of (d1), (d1)+1 and (d2), (d2)+1.



• When executing instruction

s with SM160 ON, if the device numbers of (d1) and (d2) are the same. Exchange the upper 8 bits (byte) and lower 8 bits (byte) of the word device (d1) and (d1+1).



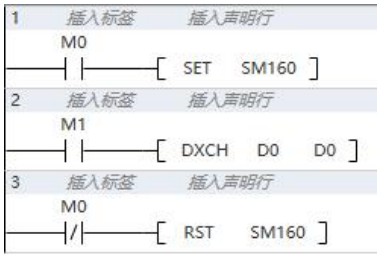
✎**Note:** If continuous execution instructions are used, conversion will be performed every operation cycle.

Error code

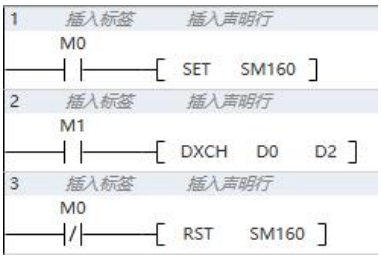
Error code	Content
4084H	In exchange mode, the devices in (d1) and (d2) are different
4085H	The output results of (d1) and (d2) in the read application instruction exceed the device range
4086H	The output results of (d1) and (d2) in the writing application instruction exceed the device range

Example :

When M0 is set, M1 is set: the high 8 bits (byte) and low 8 bits (byte) of the D0 Devices are exchanged, and the high 8 bits (byte) and low 8 bits (byte) of the D1 Devices) Exchange each other.



When M0 is reset, set M1: the value of (D1, D0) and the value of (D3, D2) are exchanged.



6.1.13 ZCP/16-bit data interval comparison

ZCP(P)

Compare the BIN 16-bit data of the device specified in (s1) and the value (bandwidth) of the BIN 16-bit data of the device specified in (s2) with the BIN 16-bit data of the device specified in the comparison source (s3), Output the result (bottom, area, top) to the device specified in (d) and later.

-[ZCP (s1) (s2) (s3) (d)]

Content, range and data type

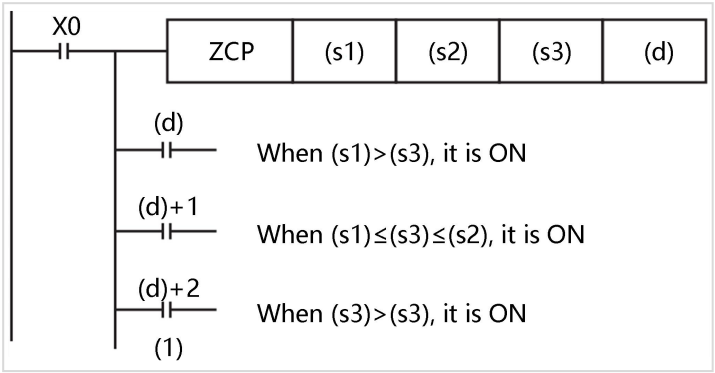
Parameter	Content	Range	Data type	Data type (label)
(s1)	The comparison value data of low limit or the device that stores the comparison value data	-32768 to 32767	Signed BIN16	ANY16_S
(s2)	The comparison value data of high limit or the device that stores the comparison value data	-32768 to 32767	Signed BIN16	ANY16_S
(s3)	Comparison source data or the device that stores the comparison source data	-32768 to 32767	Signed BIN16	ANY16_S
(d)	The start bit device of output comparison result		Bit	ANYBIT_ARRAY

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ZCP	Parameter 1											●	●	●	●	●	●	●	●	●			●	●	●	●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●	●	●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●	●	●	●
	Parameter 4		●	●	●	●					●															●	

Features

• Compare the BIN 16-bit data of the device specified in (s1) and the value (bandwidth) of the BIN 16-bit data of the device specified in (s2) with the BIN 16-bit data of the device specified in the comparison source (s3) , According to the result (bottom, area, top), one of (d), (d)+1, (d)+2 will be turned ON. (s1), (s2), (s3) are treated as BIN values within the above-mentioned setting data range. Use algebraic methods for size comparison.

• Use algebraic methods for size comparison.



(1): Even if the instruction input is OFF and the ZCP instruction is not executed, (d) to (d)+2 will keep the state before the instruction input turns from ON to OFF.

✎**Note:**

• Please set the lower comparison value (s1) to a value smaller than the upper comparison value (s2).

• When (s1) is greater than (s2), it will be processed as (s2)=(s1).

• The device specified in 3 points (d) is occupied at the beginning. Please be careful not to overlap with other control devices.

Error code

Error code	Content
4085H	The output results of (s1), (s2) and (s3) in the read application instruction exceed the device range
4086H	The output result of (d) in write application instructions exceeds the device range

Example



When M0 is set, compare whether D0 is between 0 and 1000:

If (D0)> (1000), then Y0 is ON.

If (0) ≤ (D0) ≤ (1000), then Y1 is ON.

If (D0) <(0), then Y2 is ON.

6.1.14 DZCP/32-bit data interval comparison

DZCP(P)

Compare the BIN 32-bit data of the device specified in (s1) and the value (bandwidth) of the BIN 32-bit data of the device specified in (s2) with the BIN 32-bit data of the device specified in the comparison source (s3), Output the result (bottom, area, top) to the device specified in (d) and later.

-[DZCP (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The comparison value data of low limit or the device that stores the comparison value data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(s2)	The comparison value data of high limit or the device that stores the comparison value data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(s3)	Comparison source data or the device that stores the comparison source data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(d)	The start bit device of output comparison result		Bit	ANYBIT_ARRAY

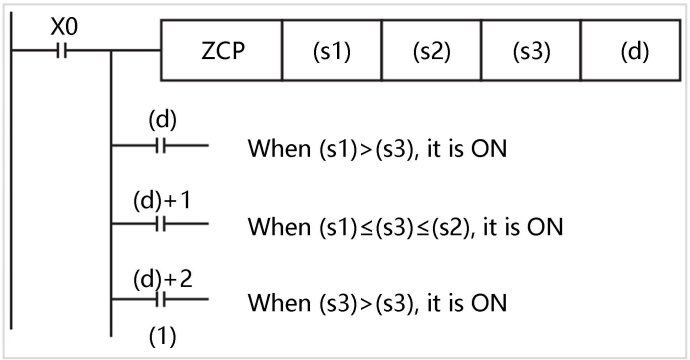
Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DZCP	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	
	Parameter 3											●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	
	Parameter 4		●	●	●	●					●															●	

Features

• Compare the BIN 32-bit data of the device specified in (s1) and the value (bandwidth) of the BIN 32-bit data of the device specified in (s2) with the BIN 32-bit data of the device specified in the comparison source (s3) , According to the result (bottom, area, top), one of (d), (d)+1, (d)+2 will be turned ON. (s1), (s2), (s3) are treated as BIN values within the above-mentioned setting data range. Use algebraic methods for size comparison.

• Use algebraic methods for size comparison.



(1): Even if the instruction input is OFF and the ZCP instruction is not executed, (d) to (d)+2 will keep the state before the instruction input turns from ON to OFF.

✎Note:

• Please set the lower comparison value (s1) to a value smaller than the upper comparison value (s2).

• When (s1) is greater than (s2), it will be processed as (s2)=(s1).

• The device specified in 3 points (d) is occupied at the beginning. Please be careful not to overlap with other control devices.

Error code

Error code	Content
4085H	The output results of (s1), (s2) and (s3) in the read application instruction exceed the device range
4086H	The output results of (d) in the write application instruction exceeds the device range

Example



When M0 is set, compare D0 with whether it is between 0 and 100000:

If (D0)> (100000), then Y0 is ON.

If (0) ≤ (D0) ≤ (100000), then Y1 is ON.

If (D0) <(0), then Y2 is ON.

6.2 Cycle shift instruction

6.2.1 ROR/16-bit cycle shift right

ROR(P)

Shift the 16-bit data of the device specified in (d) to the right by (n) bits without including the carry flag.

-[ROR (d) (n)]

Content, range and data type

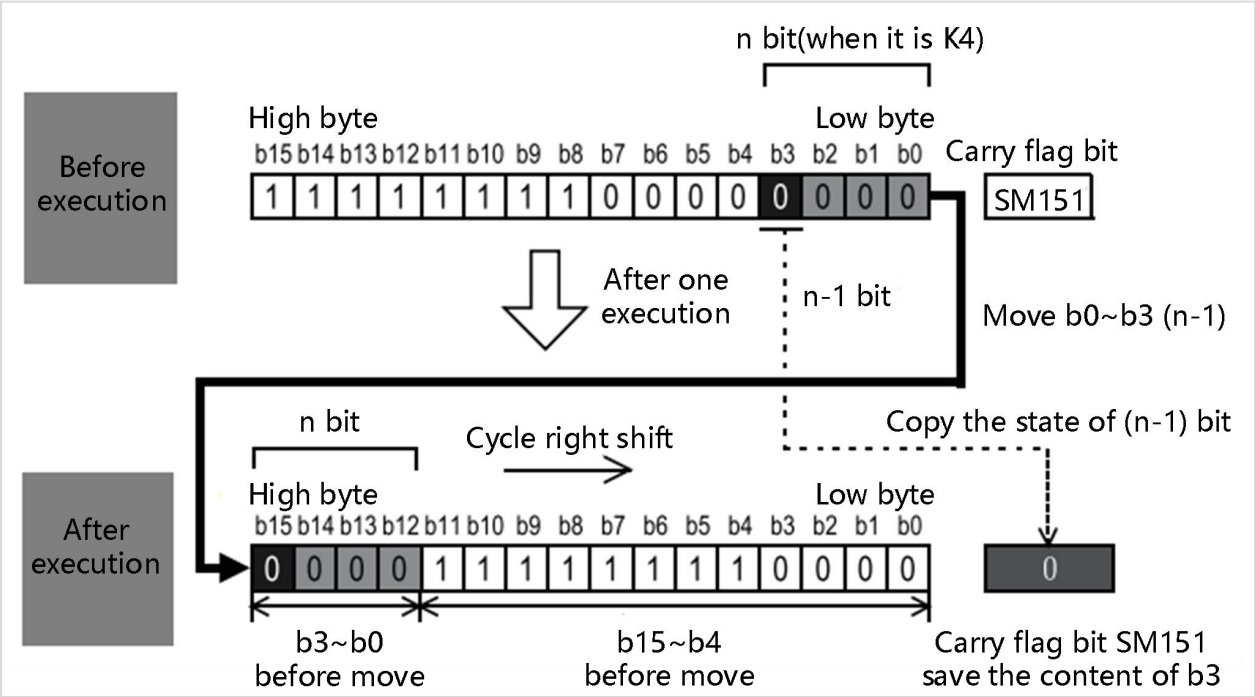
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number for cycle shift right	-	Signed BIN 16 bit	ANY16
(n)	The number of times to cycle shift right	0 to 15	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ROR	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

·The 16-bit data of the device specified in (d) is shifted right by (n) bits without including the carry flag. The carry flag is in the ON or OFF state according to the state before the ROR(P) is executed.



(n) Specifies 0 to 15. When a value of 16 or more is specified in (n), the remainder value of (n)÷16 is shifted to the right. For example, when (n)=18, 18÷16=1 and the remainder is 2, so a 2-bit right shift is performed.

Related device

Device	Name	Content
SM151	Carry	It turns ON when the last bit shifted from the lowest is 1.

✎Note:

Do not set the number of digits (n) shifted right to a negative value.

In the case of continuous execution type instructions (ROR, RCR), the right shift will be executed every scan time (operation cycle), so be careful.

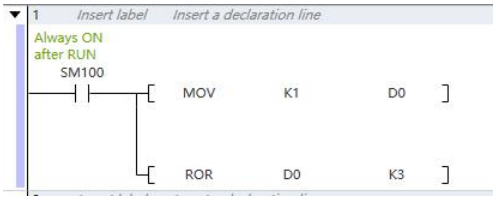
When specifying the number of digits to specify the device in (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For

example, K4Y10, K8M0).

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Shift the 1 in the D0 device by 3 bits to the right to get 8192.

6.2.2 DROR/32-bit cycle shift right

DROR(P)

Shift the 32-bit data of the device specified in (d) to the right by (n) bits without including the carry flag.

-[DROR (d) (n)]

Content, range and data type

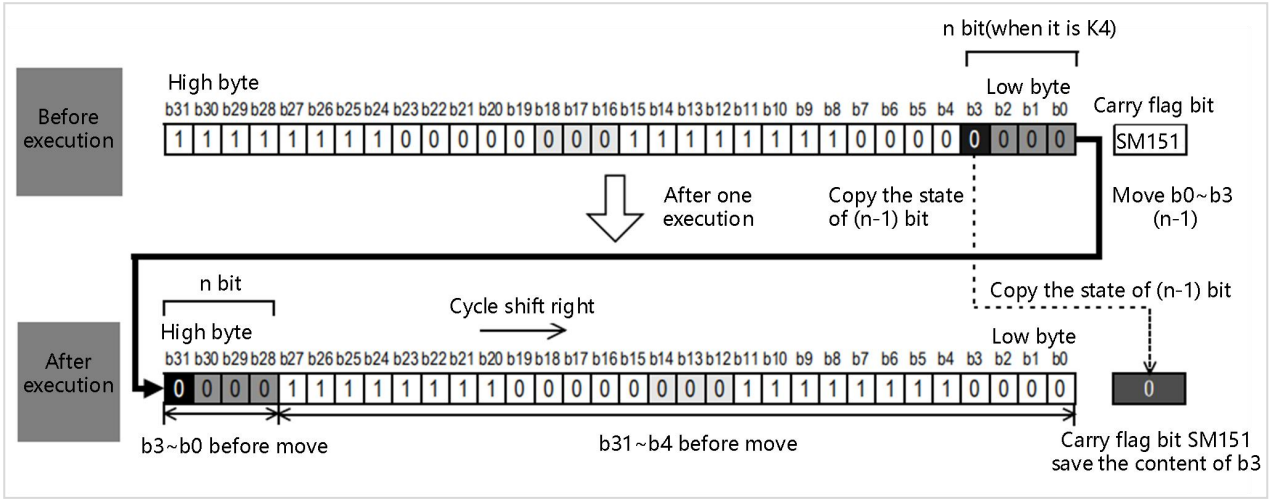
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number for cycle shift right	-	Signed BIN 32 bit	ANY32
(n)	The number of times to cycle shift right	0 to 31	Signed BIN 32 bit	ANY32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DROR	Parameter 1												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●

Features

・The 32-bit data of the device specified in (d) is shifted right by (n) bits without including the carry flag. The carry flag is on or off according to the state before DROR(P) is executed.



(n) Specifies 0 to 31. When a value of 32 or more is specified in (n), the remainder of (n)÷32 is shifted to the right. For example, when (n)=34, 34÷32=1 and the remainder is 2, so a 2-bit right shift is performed.

Related device

Device	Name	Content
SM151	Carry	It turns ON when the last bit shifted from the lowest is 1.

✎Note:

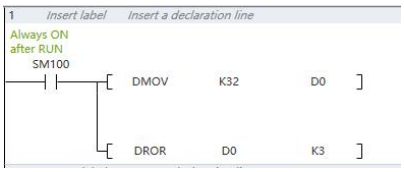
Do not set the number of digits (n) shifted right to a negative value.

In the case of continuous execution type instructions (ROR, RCR), the right shift will be executed every scan time (operation cycle), so be careful. When specifying the number of digits to specify the device in (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y10, K8M0).

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



After the rising edge of M1 is triggered, the value 32 of the D0 device is shifted right by 3 bits to get 4.

6.2.3 RCR/16-bit cycle shift right with carry

RCR(P)

Shift the 16-bit data of the device specified in (d) to the right by (n) bits with the carry flag included.

-[RCR (d) (n)]

Content, range and data type

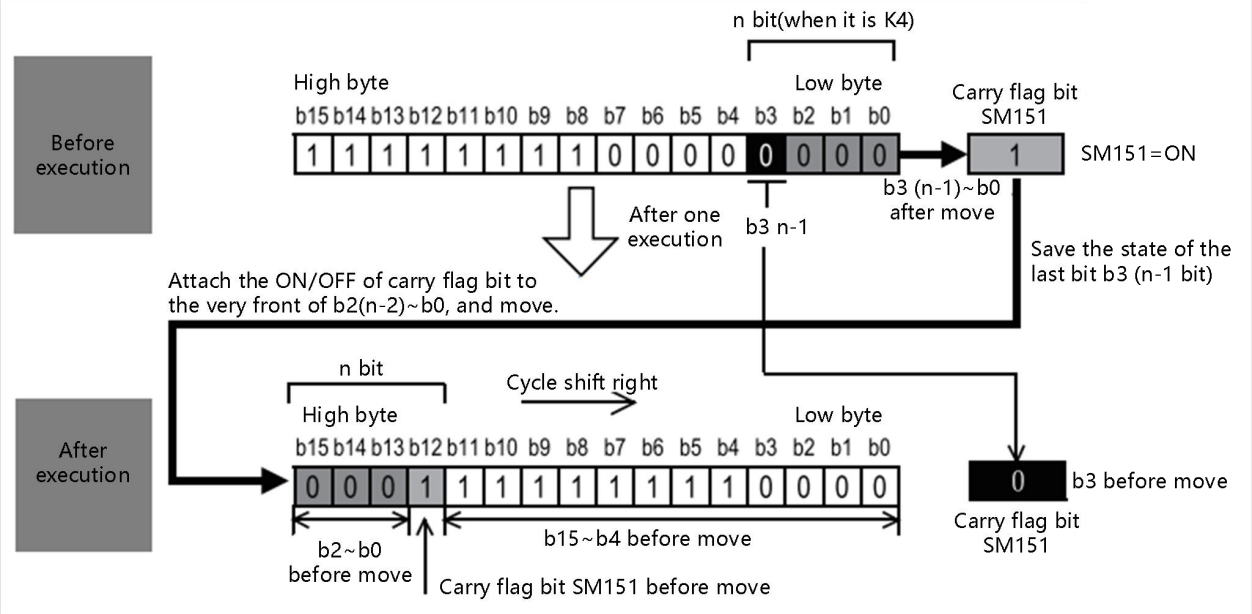
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number for cycle shift right	-	Signed BIN 16 bit	ANY16
(n)	The number of times to cycle shift right	0 to 15	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RCR	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Shift the BIN 16-bit data of the device specified in (d) to the right by (n) bits with the carry flag included. The carry flag is on or off according to the state before the RCR(P) is executed.



(n) Specifies 0 to 15. When a value of 16 or more is specified in (n), the remainder value of (n)÷16 is shifted to the right. For example, when (n)=18, 18÷16=1 and the remainder is 2, so a 2-bit right shift is performed.

Related device

Device	Name	Content
SM151	Carry	It turns ON when the last bit shifted from the lowest is 1.

✎Note:

Do not set the number of digits (n) shifted right to a negative value.

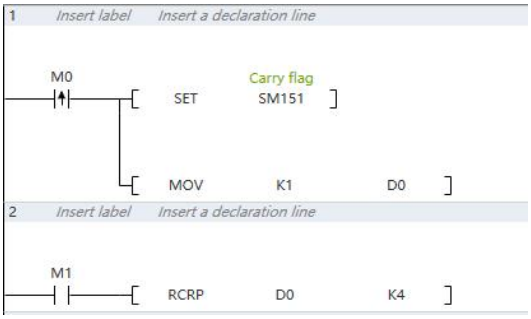
In the case of continuous execution type instructions (ROR, RCR), the right shift will be executed every scan time (operation cycle), so be careful.

When specifying the number of digits to specify the device in (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y10, K8M0).

Error code

Error code	Content
4084H	A negative value is specified in (n)
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



After the rising edge of M0 is triggered, the carry flag SM151 turns ON, and D0 is assigned the value 1. When M1=ON, the value in the D0 device is shifted right by 4 bits to get 12288.

6.2.4 DRCR/32-bit cycle shift right with carry

DRCR(P)

Shift the 32-bit data of the device specified in (d) to the right by (n) bits with the carry flag included.

-[DRCR (d) (n)]

Content, range and data type

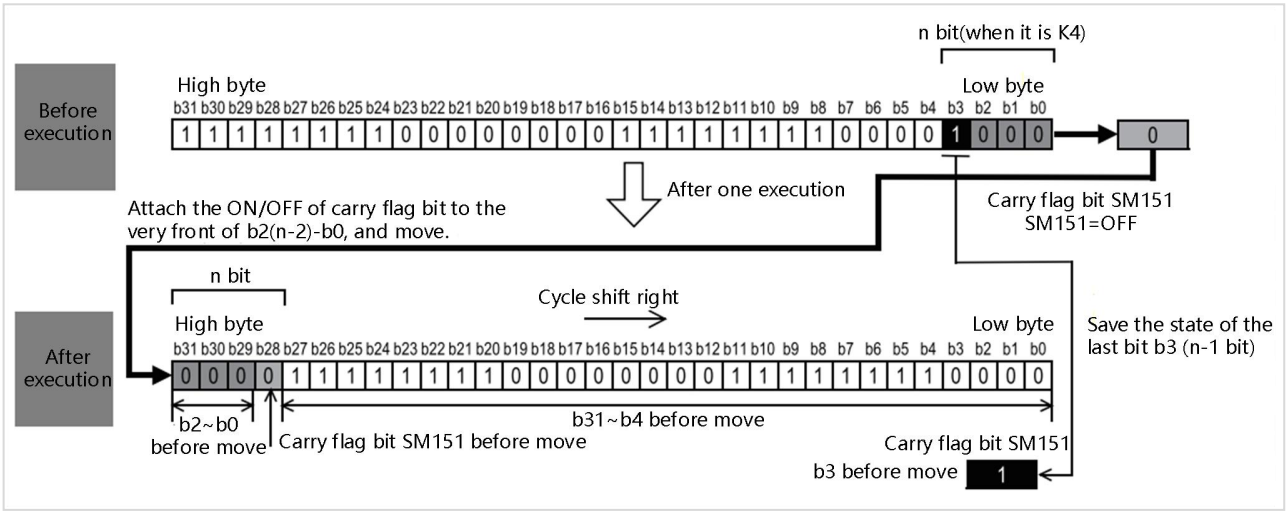
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number for cycle shift right	-	Signed BIN 32 bit	ANY32
(n)	The number of times to cycle shift right	0 to 31	Signed BIN 32 bit	ANY32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DRCR	Parameter 1												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●

Features

·The BIN 32-bit data of the device specified in (d) is shifted right by (n) bits with the carry flag included. The carry flag is in the ON or OFF state according to the state before DRCR(P) is executed.



(n) Specifies 0 to 31. When a value of 32 or more is specified in (n), the remainder value of (n)÷32 is shifted to the right. For example, when (n)=34, 34÷32=1 and the remainder is 2, so a 2-bit right shift is performed.

Related device

Devices	Name	Content
SM151	Carry	It turns ON when the last bit shifted from the lowest is 1.

✎Note:

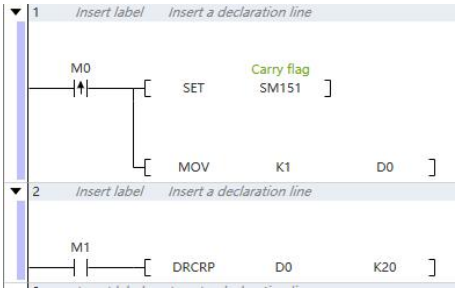
Do not set the number of bits (n) to turn right to a negative value.

In the case of continuous execution type instructions (DROR, DRCR), the right shift will be executed every scan time (operation cycle), so be careful. When specifying the number of digits to specify the device in (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y10, K8M0).

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



After the rising edge of M0 is triggered, the carry flag SM151 turns ON, and D0 is assigned the value 1. When M1=ON, the value in the D0 device is shifted

6.2.5 ROL/16-bit cycle shift left

ROL(P)

Shift the 16-bit data of the device specified in (d) to the left by (n) bits without including the carry flag.

-[ROL (d) (n)]

Content, range and data type

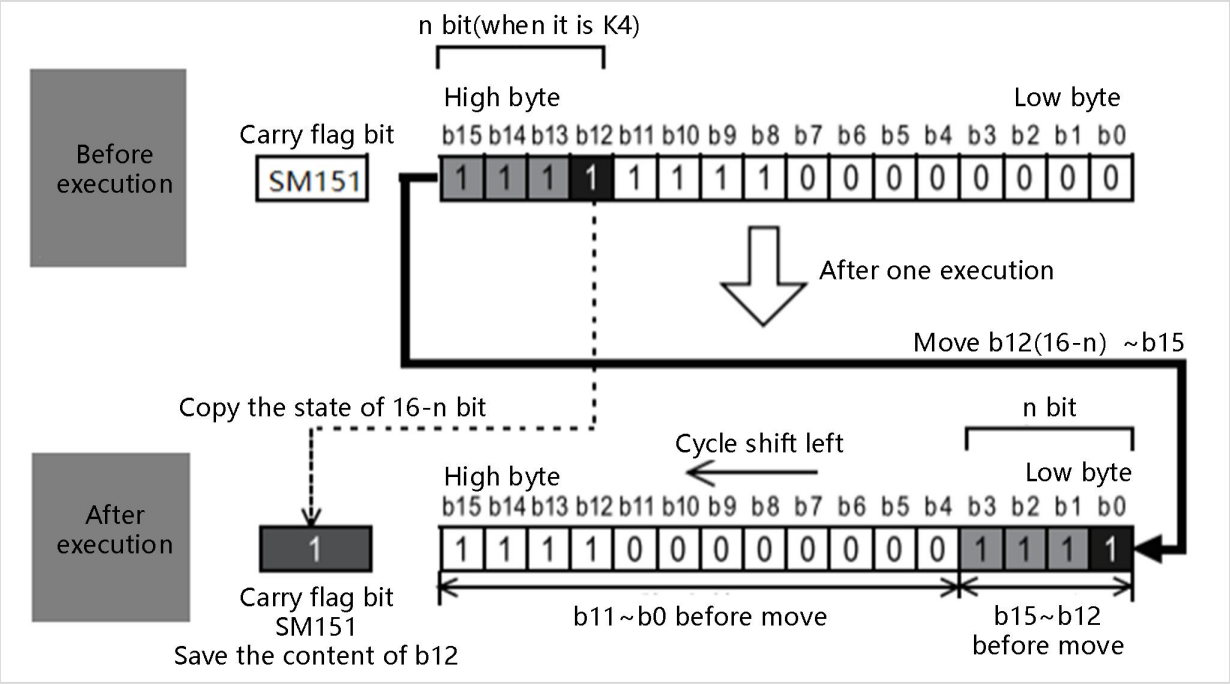
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number for cycle shift left	-	Signed BIN 16 bit	ANY16
(n)	The number of times to cycle shift left	0 to 15	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ROL	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

·The 16-bit data of the device specified in (d) is shifted to the left by (n) bits without including the carry flag. The carry flag is in the ON or OFF state according to the state before ROL(P) is executed.



(n) Specify 0 to 15. When a value of 16 or more is specified in (n), the remainder value of (n)÷16 is shifted to the left. For example, when (n)=18, 18÷16=1 and the remainder is 2, so a 2-bit left shift is performed.

Related device

Device	Name	Content
SM151	Carry	It turns ON when the last bit shifted from the highest is 1.

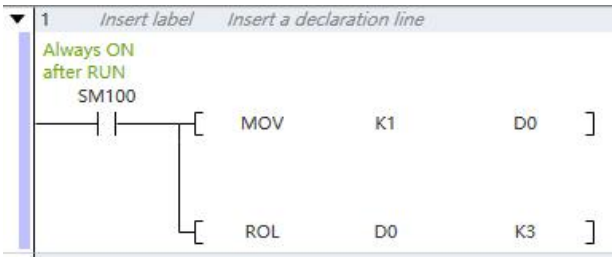
✎Note:

Do not set the number of digits (n) shifted to the left to a negative value. In the case of continuous execution type instructions (ROL, RCL), the shift to the left will be executed every scan time (operation cycle), so be careful. When specifying the number of digits to specify the device in (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y10, K8M0).

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Shift 1 in the D0 device to the left by 3 bits to get 8.

6.2.6 DROL/32-bit cycle shift left

DROL(P)

Shift the 32-bit data of the device specified in (d) to the left by (n) bits without including the carry flag.

-[DROL (d) (n)]

Content, range and data type

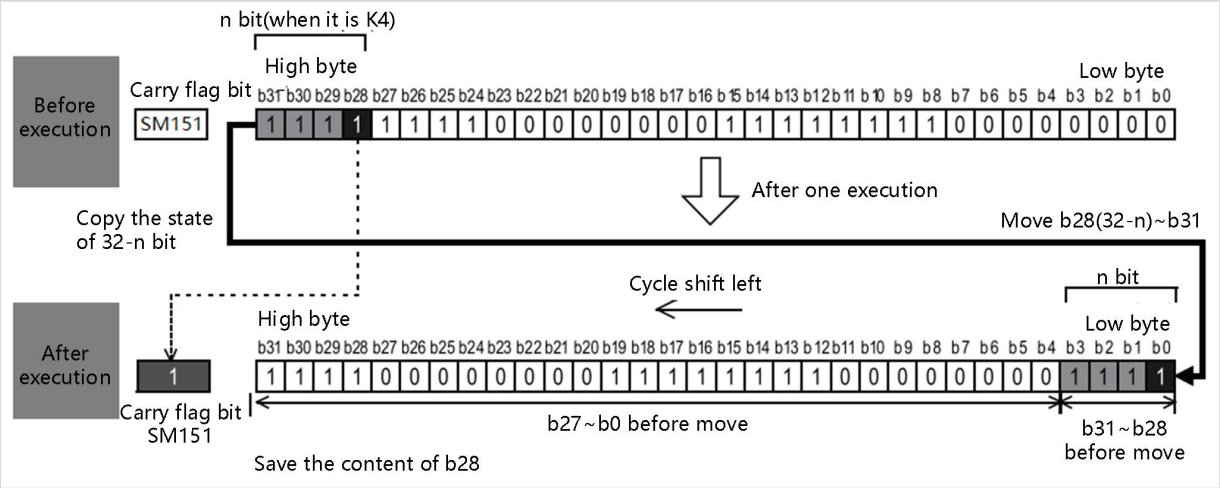
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number for cycle shift left	-	Signed BIN 32 bit	ANY32
(n)	The number of times to cycle shift left	0 to 31	Signed BIN 32 bit	ANY32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DROL	Parameter 1												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●

Features

・The 32-bit data of the device specified in (d) is shifted left by (n) bits without including the carry flag. The carry flag is on or off according to the state before DROL(P) is executed.



(n) Specifies 0 to 31. When a value of 32 or more is specified in (n), the remainder of (n)÷32 is shifted to the left. For example, when (n)=34, 34÷32=1 and the remainder is 2, so a 2-bit left shift is performed.

Related device

Device	Name	Content
SM151	Carry	It turns ON when the last bit shifted from the highest is 1.

✎Note:

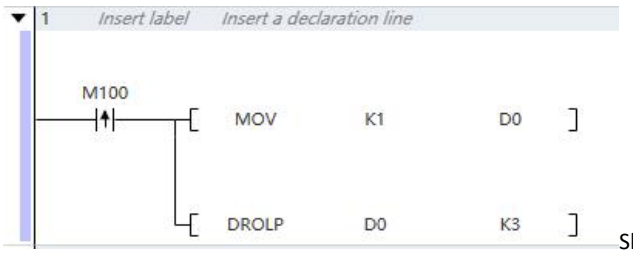
Do not set the number of digits (n) shifted to the left to a negative value.

In the case of continuous execution type instructions (ROL, RCL), the shift to the left will be executed every scan time (operation cycle), so be careful. When specifying the number of digits to specify the device in (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y10, K8M0).

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Shift 1 in the D0 device to the left by 3 bits to get 8.

6.2.7 RCL/16-bit cycle shift left with carry

RCL(P)

Shift the 16-bit data of the device specified in (d) to the left by (n) bits with the carry flag included.

-[RCL (d) (n)]

Content, range and data type

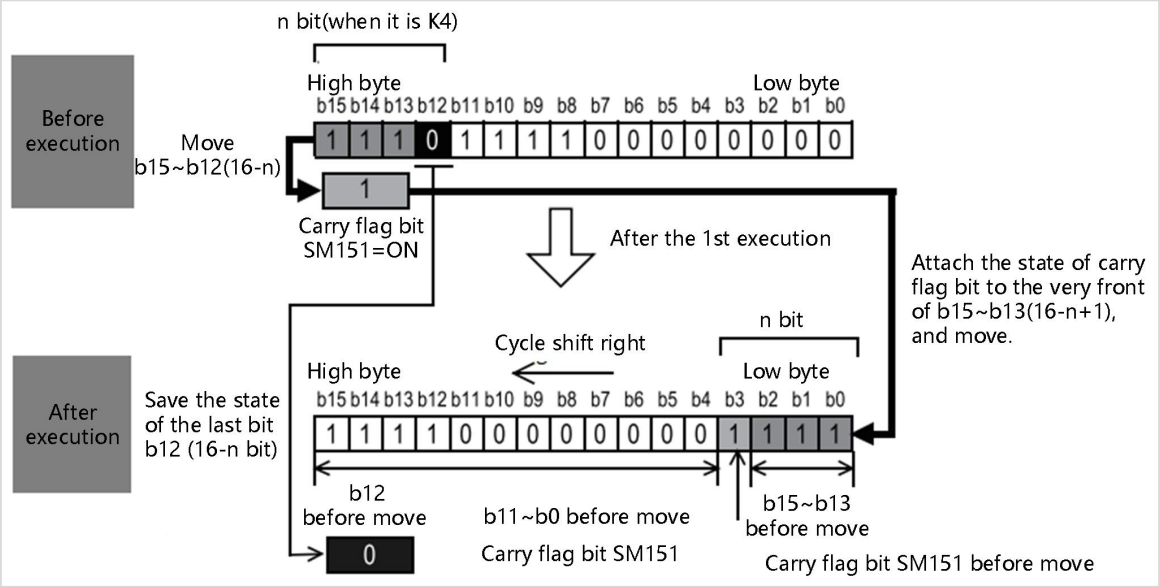
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number for cycle shift left	-	Signed BIN 16 bit	ANY16
(n)	The number of times to cycle shift left	0 to 15	Signed BIN 16 bit	ANY16

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RCL	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

·The 16-bit data of the device specified in (d) is shifted (n) to the left with the carry flag included. The carry flag is on or off according to the state before RCL(P) is executed.



(n) Specifies 0 to 15. When a value of 16 or more is specified in (n), the remainder value of (n)÷16 is shifted to the left. For example, when (n)=18, 18÷16=1 and the remainder is 2, so a 2-bit left shift is performed.

Related device

Device	Name	Content
SM151	Carry	It turns ON when the last bit shifted from the highest is 1.

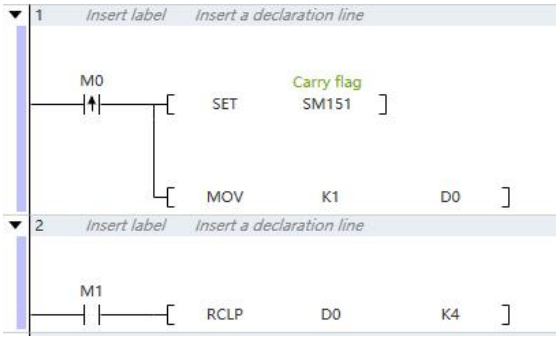
✎Note:

Do not set the number of digits (n) shifted to the left to a negative value. In the case of continuous execution type instructions (ROL, RCL), the shift to the left will be executed every scan time (operation cycle), so be careful. When specifying the number of digits to specify the device in (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y10, K8M0).

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



After the rising edge of M0 is triggered, the carry flag SM151 turns ON, and D0 is assigned the value 1.

6.2.8 DRCL/32-bit cycle shift left with carry

DRCL(P)

Move the 32-bit data of the device specified in (d) to the left by (n) bits with the carry flag included.

-[DRCL (d) (n)]

Content, range and data type

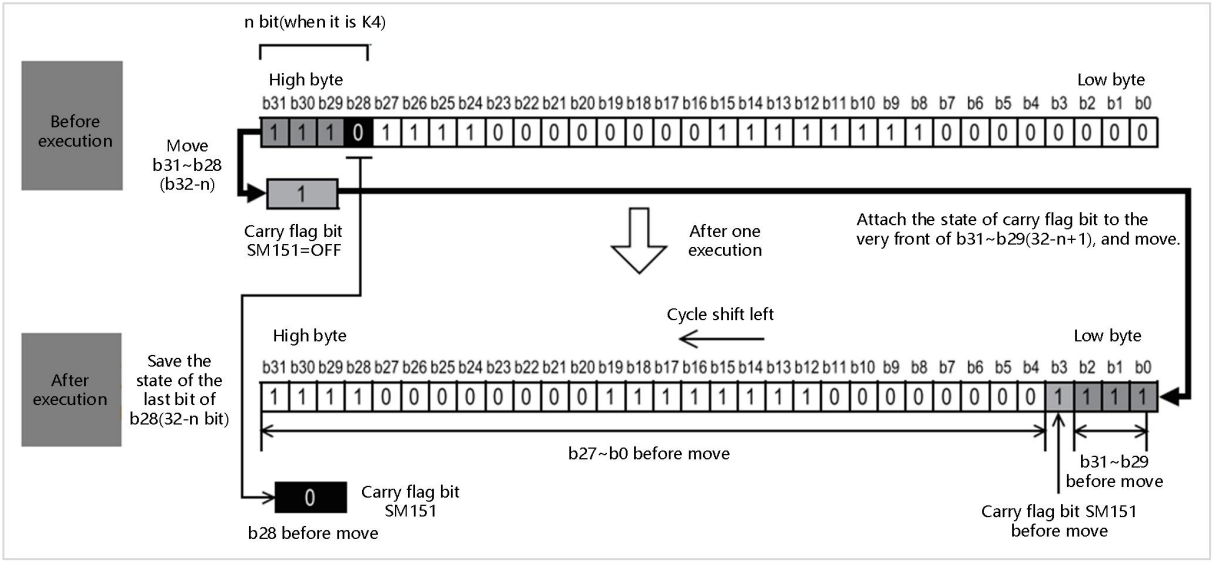
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number for cycle shift left	-	Signed BIN 32 bit	ANY32
(n)	The number of times to cycle shift left	0 to 31	Signed BIN 32 bit	ANY32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DRCL	Parameter 1												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●

Features

The 32-bit data of the device specified in (d) is shifted (n) to the left with the carry flag included. The carry flag is on or off according to the state before RCL(P) is executed.



(n) Specifies 0 to 31. When a value of 32 or more is specified in (n), the remainder of (n)÷32 is shifted to the left. For example, when (n)=34, 34÷32=1 and the remainder is 2, so a 2-bit left shift is performed.

Related device

Devices	Name	Content
SM151	Carry	Turns ON when the last bit shifted from the highest is 1.

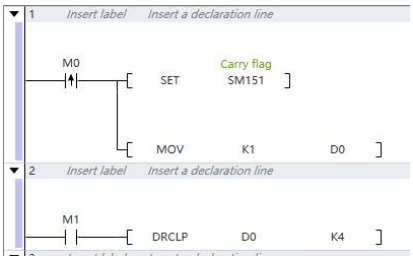
✎Note:

Do not set the number of digits (n) shifted to the left to a negative value. In the case of continuous execution type instructions (ROL, RCL), the shift to the left will be executed every scan time (operation cycle), so be careful. When specifying the number of digits to specify the device in (d), only K4 (16-bit instruction) or K8 (32-bit instruction) is valid. (For example, K4Y10, K8M0).

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



After the rising edge of M0 is triggered, the carry flag SM151 turns ON, and D0 is assigned the value 1. When M1=ON, carry the value in the D0 device to the left by 4 bits to get 24.

6.2.9 SFTR/n-bit shift right of n-bit data

SFTR(P)

Shift (n2) the data of the start (n1) bits of the device specified in (d) to the right.

-[SFTR (s) (d) (n1) (n2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of the device storing the shifted data after shifting	-	Bit	ANY_BOOL
(d)	The shifted device start number	-	Bit	ANY_BOOL
(n1)	The length of shifted data	0 to 32767	Signed BIN 16 bit	ANY16
(n2)	Number of shifts	0 to 32767	Signed BIN 16 bit	ANY16

Device used

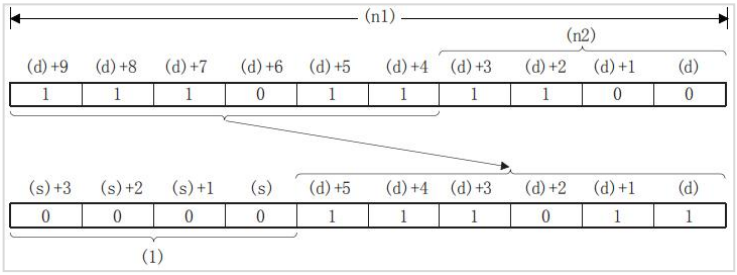
instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SFTR	Parameter 1	●	●	●	●	●					●															●	●
	Parameter 2		●	●	●	●					●															●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Shift (n2) the data of the start (n1) bits of the device specified in (d) to the right. After shifting, the point (n2) starting from (s) is transferred to the point (n2) starting from (d) + (n1 to n2).

When K0 is specified in (s), the bit of the (d) + (n1 to n2) starting point (n2) after the shift is set to 0.

When K1 is specified in (s), the bit of the (d) + (n1 to n2) starting point (n2) after the shift is set to 1.



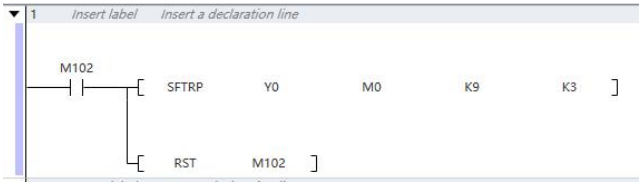
(1): When (s)=K0, it becomes 0.

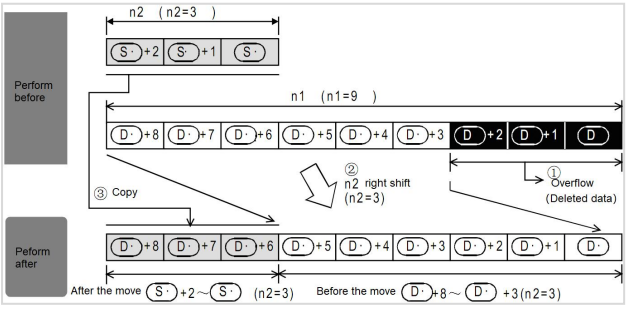
Error code

Error code	Content
4084H	When the value specified in (n1) and (n2) exceeds the range of 0 to 32767
	When the value specified in (n1) and (n2) is (n1)<(n2)
4085H	When the device specified in read application instructions (s), (d), (n1) and (n2) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

For n1=9 bits (the length of the shift register) data starting with M0, right shift n2=3 bits. After shifting, transfer n2=3 bits from Y0 to n2=3 bits from M6.




n2 (n2=3)
S·+2
S·+1
S·
Perform before
n1 (n1=9)
D·+8
D·+7
D·+6
D·+5
D·+4
D·+3
D+2
D+1
D
② n2 right shift (n2=3)
① Overflow (Deleted data)
③ Copy
Peform after
After the move S·+2～S· (n2=3)
Before the move D·+8～D·+3(n2=3)

6.2.10 SFTL/n-bit shift left of n-bit data

SFTL(P)

Shift the start (n1) bit data of the device specified in (d) to the left by (n2) bits.

-[SFTL (s) (d) (n1) (n2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of the device storing shifted data after shifting	-	Bit	ANY_BOOL
(d)	The shifted device start number	-	Bit	ANY_BOOL
(n1)	The length of shifted data	0 to 32767	Signed BIN 16 bit	ANY16
(n2)	Number of shifts	0 to 32767	Signed BIN 16 bit	ANY16

Device used

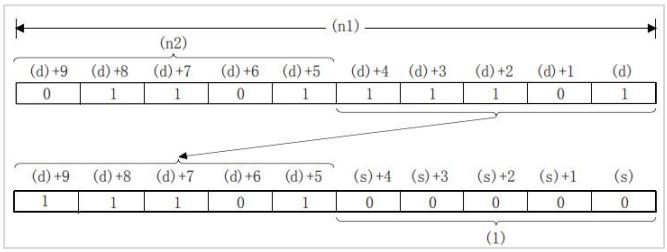
instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SFTL	Parameter 1	●	●	●	●	●					●															●	●
	Parameter 2		●	●	●	●					●															●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Shift (n2) bits of the data at the beginning (n1) bits of the device specified in (d). After shifting, the point (n2) starting from (s) is transferred to the point (n2) starting from (d) + (n1 to n2).

When K0 is specified in (s), the bit of the (d) + (n1 to n2) starting point (n2) after the shift is set to 0.

When K1 is specified in (s), the bit of the (d) + (n1 to n2) starting point (n2) after the shift is set to 1.

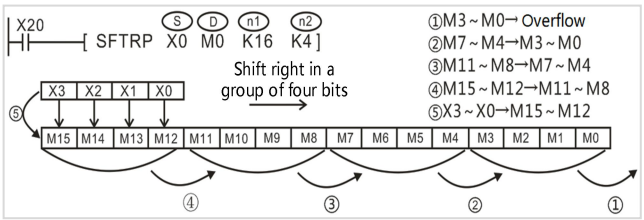


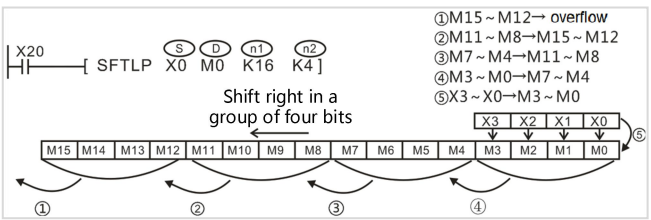
(1): When (s)=K0, it becomes 0.

Error code

Error code	Content
4084H	When the value specified in (n1) and (n2) exceeds the range of 0 to 32767
	When the value specified in (n1) and (n2) is (n1)<(n2)
4085H	When the device specified in read application instructions (s), (d), (n1) and (n2) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example





6.2.11 WSFR/n-word shift right of n-word data

WSFR(P)

Shift (n2) the data of the start (n1) bits of the device specified in (d) to the right.

-[WSFR (s) (d) (n1) (n2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of the device storing shifted data after shifting	-	word	ANY_BOOL
(d)	The shifted device start number	-	word	ANY_BOOL
(n1)	The length of shifted data	0 to 32767	Signed BIN 16 bit	ANY16
(n2)	Number of shifts	0 to 32767	Signed BIN 16 bit	ANY16

Device used

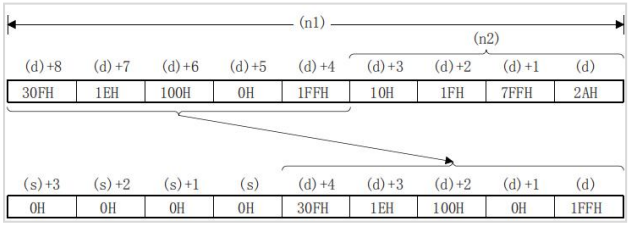
instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SFTR	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Shift (n2) the data of the beginning (n1) word of the device specified in (d) to the right. After shifting, the point (n2) starting from (s) is transferred to the point (n2) starting from (d) + (n1 to n2).

When K is specified in (s), the device at (d) + (n1 to n2) starting (n2) point after shifting is set to the specified value.

If the value specified in (n1) or (n2) is 0, it will be no processing.



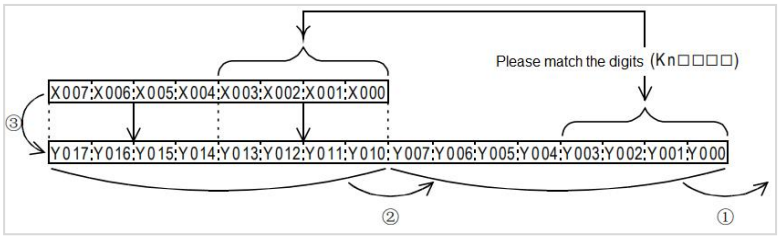
Error code

Error code	Content
4084H	When the value specified in (n1) and (n2) exceeds the range of 0 to 32767
	When the value specified in (n1) and (n2) is (n1)<(n2)
	When (s) and (d) both specify KnM, KnX, and KnS, the value of n varies.
4085H	When the device specified in read application instructions (s), (d), (n1) and (n2) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

(s), (d) specify the same multiple in the digit specified device. This program realizes to shift Y0 to Y7 bits right, shift Y10 to Y17 right to Y0 to Y7, and then store X0 to X7 to Y10 to Y17.





6.2.12 WSFL/n-word shift left of n-word data

WSFL(P)

Shift the start (n1) bit data of the device specified in (d) to the left by (n2) bits.

-[WSFL (s) (d) (n1) (n2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of the device storing shifted data after shifting	-	Word	ANY_BOOL
(d)	The shifted device start number	-	Word	ANY_BOOL
(n1)	The length of shifted data	0 to 32767	Signed BIN 16 bit	ANY16
(n2)	Number of shifts	0 to 32767	Signed BIN 16 bit	ANY16

Device used

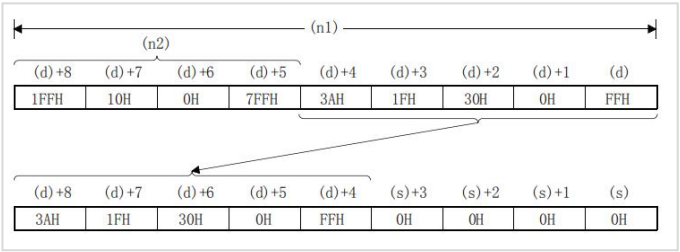
instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SFTR	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Shift (n2) the data of the beginning (n1) word of the device specified in (d) to the left. After shifting, transfer the point (n2) starting from (s) to the point (n2) starting from (d).

When K is specified in (s), the device at (d) + (n1 to n2) starting (n2) point after shifting is set to the specified value.

If the value specified in (n1) or (n2) is 0, it will be no processing.

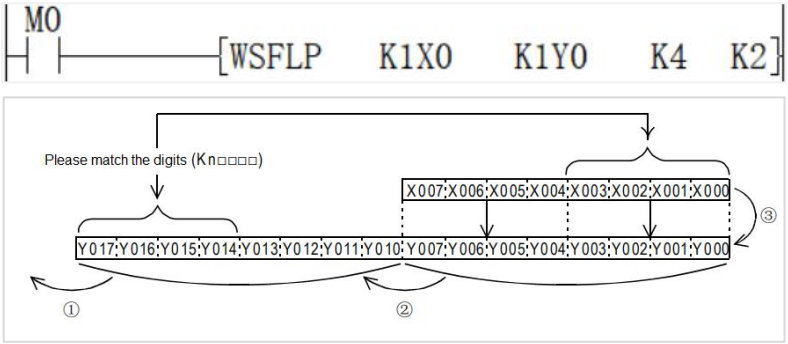


Error Code

Error code	Content
4084H	When the value specified in (n1) and (n2) exceeds the range of 0 to 32767
	When the value specified in (n1) and (n2) is (n1)<(n2)
	When (s) and (d) both specify KnM, KnX, and KnS, the value of n varies.
4085H	When the device specified in read application instructions (s), (d), (n1) and (n2) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

(S), (d) Do the same multiple specification in the digit specification device. This program realizes to remove the high bits of Y10 to Y17 left, move Y0 to Y7 left to Y10 to Y17, and then store X0 to X7 to Y0 to Y7.



6.2.13 SFR/n-bit shift right of 16-bit data

SFR(P)

Shift the 16-bit data of the device specified in (d) right by (n) bits.

-[SFR (d) (n)]

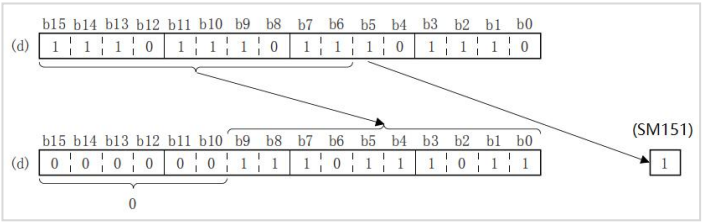
Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	The start number of the device storing the shifted data	-	Signed BIN 16 bit	ANY16
(n)	Number of shifts	0-15	Signed BIN 16 bit	ANY16

Device used

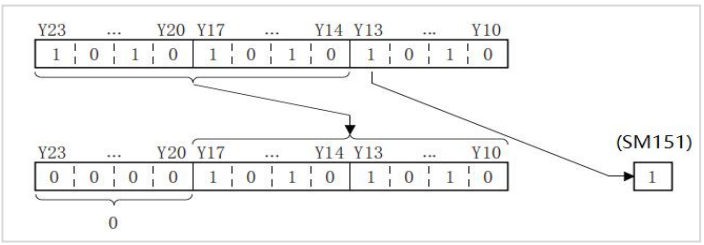
instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SFR	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●

Features



When (N)=6

Shift the 16-bit data of the device specified in (d) to the right (n) bits from the highest bit. The (n) bit from the most significant bit will become 0.



When (N)=6

When a bit device is specified in (d), the device range specified in the digit specification is shifted to the right.

(n) Specifies 0 to 15. When a value of 16 or more is specified in (n), the remainder of (n)÷16 is shifted to the left. For example, when (n)=18, 18÷16=1 and the remainder 2, so it is shifted by 2 bits to the right.

Related device

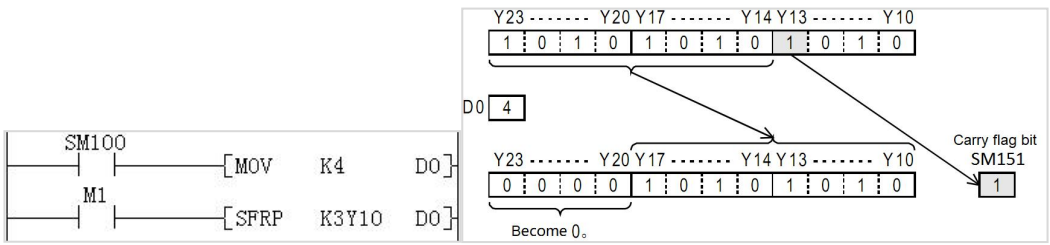
Device	Name	Content
SM151	Carry	Set to ON/OFF according to the state of n-1 bit (1/0)

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example

When M1 is ON, the contents of Y10 to Y23 are shifted to the right by the number of digits specified in D0.



6.2.14 DSFR/n word data shift right by 1 word

DSFR(P)

Shift the data at the start (n) point of the device specified in (d) to the right by 1 word.

-[DSFR (d) (n)]

Content, range and data type

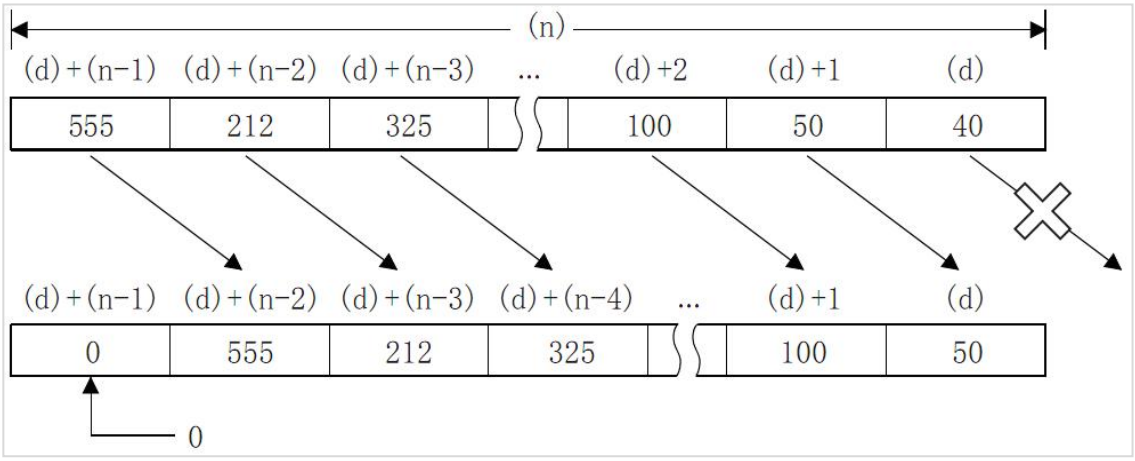
Parameter	Content	Range	Data type	Data type (label)
(d)	The start number of the device storing the shifted data	-	Signed BIN 16 bit	ANY16
(n)	Number of shifts	0 to 32767	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSFR	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

• Shift the data at the start (n) point of the device specified in (d) by 1 word to the right.



• The device specified in (d)+(n-1) will become 0.

✎Note:

In (d), when specifying the device number by specifying the number of bits of the bit device, the device number should be a multiple of 16 (0, 16, 32, 64...), and only K4 should be specified for the number of bits. When the number of bits is not K4, K4 is used for processing.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the range of 0 to 32767
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example

When M1 is ON, shift the contents of D0 to D4 to the right by 1 word (D1→D0, D2→D1, D3→D2, D4→D3, D4 is set to 0).



Before execution:

Devices	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
D1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
D2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
D3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4
D4	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	5
D5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

After execution:

Devices	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
D1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
D2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4
D3	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	5
D4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.2.15 SFL/n-bit shift left of 16-bit data

SFL(P)

Shift the 16-bit data of the device specified in (d) to the left by (n) bits.

-[SFL (d) (n)]

Content, range and data type

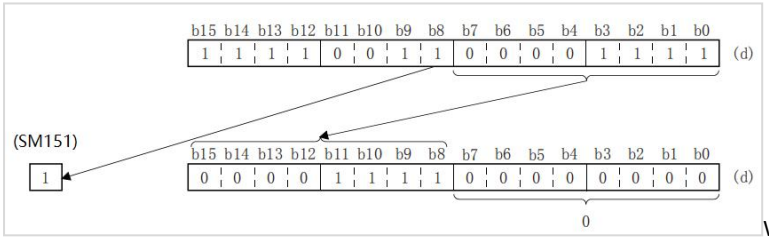
Parameter	Content	Range	Data type	Data type (label)
(d)	The start number of the device storing the shifted data	-	Signed BIN 16 bit	ANY16
(n)	Number of shifts	0 to 15	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SFL	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●

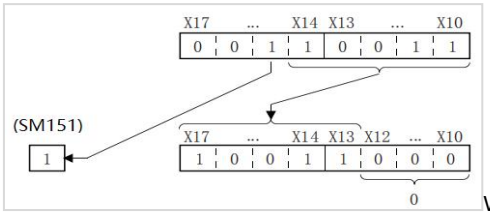
Features

Shift the 16-bit data of the device specified in (d) to the left (n) bits from the lowest bit. The (n) bit from the lowest bit will become 0.



When (n)=8, it is as follows.

When a bit device is specified in (d), the left shift is performed in the device range specified in the digit specification.



When (n)=3, it is as follows.

(n) Specify 0 to 15. When a value of 16 or more is specified in (n), the remainder of (n)÷16 is shifted to the left. For example, when (n)=18, 18÷16=1 remainder 2, so it is shifted by 2 bits to the left.

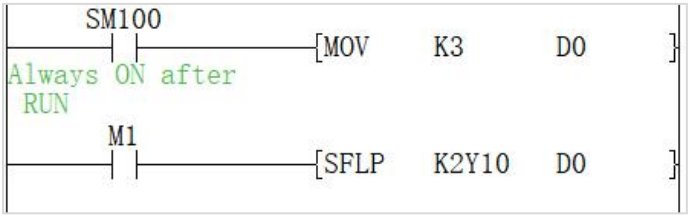
Related device

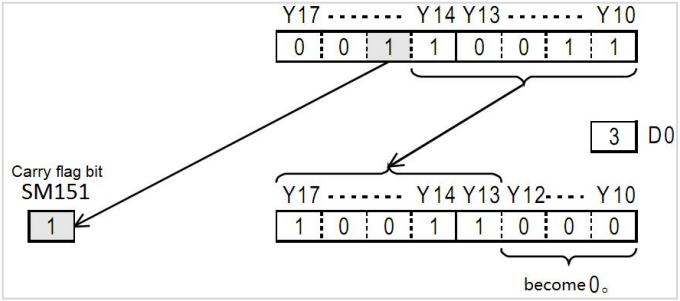
Device	Name	Content
SM151	Carry	Turn ON/OFF according to the state of n+1 bit (1/0)

Error code

Error code	Content
4084H	A negative value is specified in (n).
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example: When M1 is ON, the contents of Y10 to Y17 are shifted to the left by the number of digits specified in D0.





6.2.16 DSFL/one word shift left of n word data

DSFL(P)

Move the data at the beginning (n) point of the device specified in (d) by 1 word to the left.

-[DSFL (d) (n)]

Content, range and data type

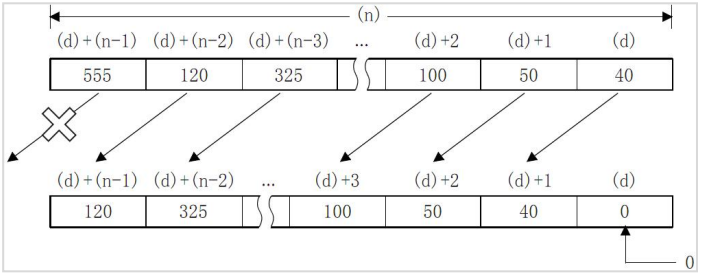
Parameter	Content	Range	Data type	Data type (label)
(d)	The start number of the device storing the shifted data	-	Signed BIN 16 bit	ANY16
(n)	Number of shifts	0 to 32,767	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSFL	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Shift the data at the start (n) point of the device specified in (d) to the left by 1 word.



The device specified in (d) will become 0.

✎Note:

In (d), when specifying the device number by specifying the number of bits of the bit device, the device number should be a multiple of 16 (0, 16, 32, 64...), and only K4 should be specified for the number of bits. When the number of bits is not K4, K4 is used for processing.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the range of 0 to 32,767
4085H	The output results of (d) and (n) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example

When M1 is ON, shift the contents of D0 to D4 to the left by 1 word (D3→D4, D2→D3, D1→D2, D0→D1, D0 is set to 0).



Before execution:

Devices	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
D1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
D2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
D3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4
D4	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	5
D5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

After execution:

Devices	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
D2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
D3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
D4	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4
D5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.3 Arithmetic operation instructions

6.3.1 ADD/16-bit addition operation

ADD(P)

Add the BIN 16-bit data specified in (s1) and the BIN 16-bit data specified in (s2), and store the result in the device specified in (d).

-[ADD (s1) (s2) (d)]

Content, range and data type

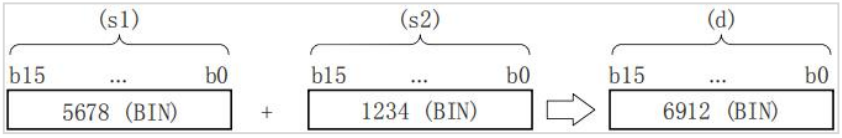
Parameter	Content	Range	Data type	Data type (label)
(s1)	Addition operation data or the device storing the addition data	-32768 to 32767	Signed BIN16	ANY16_S
(s2)	Addition operation data or the device storing the addition data	-32768 to 32767	Signed BIN16	ANY16_S
(d)	Device for storing operation results		Signed BIN16	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ADD	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

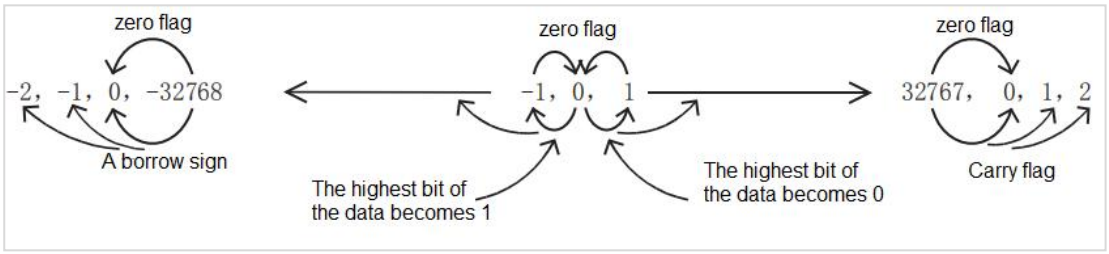
Features

Add the BIN 16-bit data specified in (s1) and the BIN 16-bit data specified in (s2), and store the result of the addition in the device specified in (d).



Related device

Devices	Name	Content
SM151	Carry	When the operation result exceeds 32,767, the carry flag will be (ON).
SM152	Borrow	When the operation result is less than -32,768, the borrow flag will be (ON).
SM153	Zero point	When the operation result is 0, the zero flag will be (ON).



✎Note:

① When the source operand and destination operand are specified as the same device:

The source operand and destination operand can also be specified as the same device number.

In this case, if you use continuous execution instructions (ADD, DADD), the result of the addition operation will change every operation cycle.

② The difference between the ADD instruction and the INC instruction using the +1 addition operation program:

ADD[P] means that every time X001 changes from OFF to ON, the content of D0 is added by one operation.

Although this instruction is very similar to the INCP instruction described later, there are some differences in the following content.

			ADD/ADDP/DADD/DADDP instructions	INC/INCP/DINC/DINCP instructions
Sign bit (zero, borrow, carry)			Action	No action
Calculation result	16-bit operation result	(S) + (+1) = (d)	32767 → 0 → +1 → +2 →	32767→-32768→-32767
		(S) + (-1) = (d)	← -2 ← -1 ← 0 ← -32768	——
	32-bit operation result	(S) + (+1) = (d)	2147483647 → 0 → +1 → +2 →	2147483647 → -2147483648→ -2147483647
		(S) + (-1) = (d)	← -2 ← -1 ← 0 ← -2147483648	——

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Add 10 to the data in (D0), and store the operation result in (D2), that is, (D0) + 10 → (D2).

6.3.2 DADD/32-bit addition operation

DADD(P)

Add the BIN32-bit data specified in (s1) and the BIN32-bit data specified in (s2), and store the result in the device specified in (d).

-[DADD (s1) (s2) (d)]

Content, range and data type

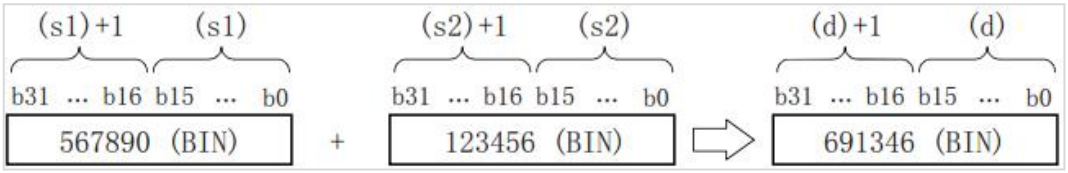
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s1)	Addition data or the device storing the addition data	-2147483648 to 2147483647	Signed BIN32	ANY32_S	DINT
(s2)	Addition data or the device storing the addition data	-2147483648 to 2147483647	Signed BIN32	ANY32_S	DINT
(d)	Device for storing operation results		Signed BIN32	ANY32_S	DINT

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DADD	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

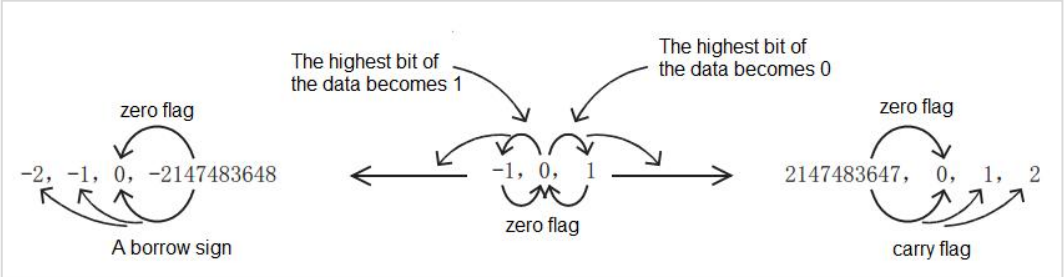
Features

Add the BIN32-bit data specified in (s1) and the BIN32-bit data specified in (s2), and store the result of the addition in the device specified in (d).



Related device

Devices	Name	Content
SM151	Carry	When the operation result exceeds 32,767, the carry flag will be (ON).
SM152	Borrow	When the operation result is less than -32,768, the borrow flag will be (ON).
SM153	Zero point	When the operation result is 0, the zero flag will be (ON).



✎Note:

① When the source operand and destination operand are specified as the same device:

The source operand and destination operand can also be specified as the same device number.

In this case, if you use continuous execution instructions (ADD, DADD), the result of the addition operation will change every operation cycle. Please note.

② The difference between the ADD instruction and the INC instruction using the +1 addition operation program:

ADD[P] means that every time X001 changes from OFF to ON, the content of D0 is added by one operation.

Although this instruction is very similar to the INCP instruction described later, there are some differences in the following content.

			ADD/ADDP/DADD/DADDP instructions	INC/INCP/DINC/DINCP instructions
sign bit (zero, borrow, carry)			Action	No action
Calculation result	16-bit Operation result	(S) + (+1) = (d)	32767→0→+1→+2→	32767→-32768→-32767
		(S) + (-1) = (d)	←-2←-1←0←-32768	——
	33-Bit operation result	(S) + (+1) = (d)	2147483647→0→+1→+2→	2147483647→-2147483648→-2147483647
		(S) + (-1) = (d)	←-2←-1←0←-2147483648	——

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Add 100000 to the data in (D1, D0), and store the result of the operation in (D3, D2), that is, (D1, D0) + 100000 → (D3, D2).

6.3.3 SUB/16-bit subtraction operation

SUB(P)

Subtract the BIN 16-bit data specified in (s1) and the BIN 16-bit data specified in (s2), and store the result in the device specified in (d).

-[SUB (s1) (s2) (d)]

Content, range and data type

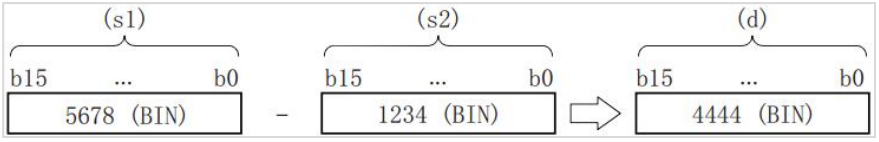
Parameter	Content	Range	Data type	Data type (label)
(s1)	The subtraction data or the device storing the subtraction data	-32768 to 32767	Signed BIN16	ANY16_S
(s2)	The subtraction data or the device storing the subtraction data	-32768 to 32767	Signed BIN16	ANY16_S
(d)	Device for storing calculation results		Signed BIN16	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SUB	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

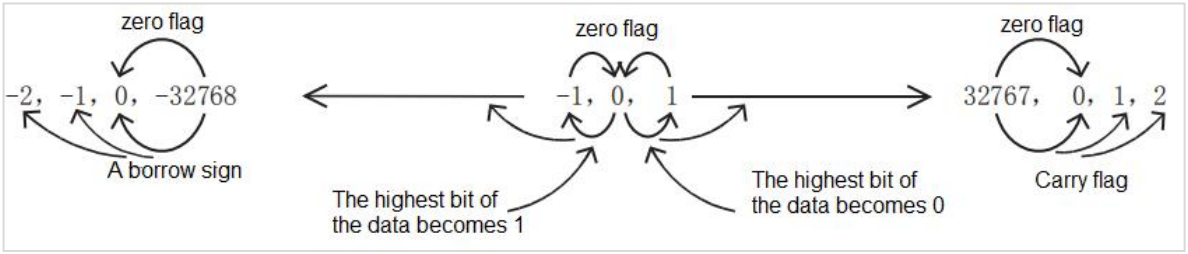
Features

Subtract the BIN 16-bit data specified in (s1) and the BIN 16-bit data specified in (s2), and store the result of the operation in the device specified in (d).



Related device

Devices	Name	Content
SM151	Carry	When the operation result exceeds 32,767, the carry flag will be (ON).
SM152	Borrow	When the operation result is less than -32,768, the borrow flag will be (ON).
SM153	Zero point	When the operation result is 0, the zero flag will be (ON).



✎Note:

① When the source operand and destination operand are specified as the same device:

The source operand and destination operand can also be specified as the same device number.

In this case, if continuous execution type instructions (SUB, DSUB) are used, the result of the subtraction operation will change every operation cycle. Please be careful.

② The difference between the SUB(P) instruction and the -(P) instruction and DEC(P) instruction executed by the -1 subtraction program

SUB(P) instruction every time X1 changes from OFF to ON, the program of D0 content -1 is similar to -(P) instruction and DEC(P) instruction described later, but the following contents are different.

			SUB/SUBP/DSUB/DSUBP instructions	DEC/DECP/DDEC/DDECP instructions
Sign bit (zero, borrow, carry)			Action	No action
Calculation result	16-bit operation result	(S)-(+1)=(d)	←-2←-1←0←-32768	-32768→+32767→32766
		(S)-(-1)=(d)	+32767→0→+1→+2→	——
	32-bit operation result	(S)-(+1)=(d)	←-2←-1←0←-2147483648	-2147483648→2147483647→2147483646
		(S)-(-1)=(d)	2147483647→0→+1→+2→	——

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Subtract 10 from the data in D0, and store the calculation result in D2, that is, (D0)-10 → (D2).

6.3.4 DSUB/32-bit subtraction operation

DSUB(P)

Subtract the BIN32-bit data specified in (s1) and the BIN32-bit data specified in (s2), and store the result in the device specified in (d).

-[DSUB (s1) (s2) (d)]

Content, range and data type

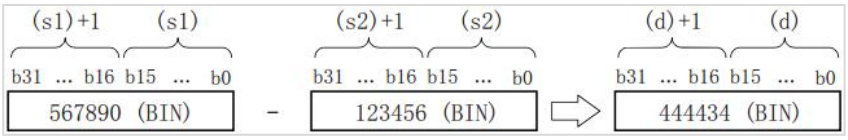
Parameter	Content	Range	Data type	Data type (label)
(s1)	The subtraction data or the device storing the subtraction data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(s2)	The subtraction data or the device storing the subtraction data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(d)	Device for storing calculation results		Signed BIN32	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSUB	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

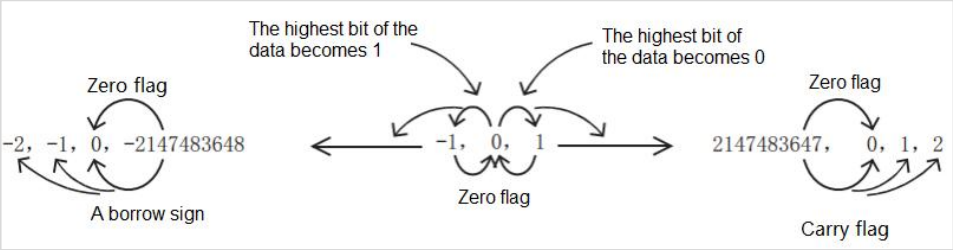
Features

Subtract the BIN32-bit data specified in (s1) and the BIN32-bit data specified in (s2), and store the result of the operation in the device specified in (d).



Related device

Devices	Name	Content
SM151	Carry	When the operation result exceeds 2,147,483,647, the carry flag will be ON.
SM152	Borrow	When the operation result is less than -2,147,483,648, the borrow flag will be ON.
SM153	Zero point	When the operation result is 0, the zero flag will be ON.



✎Note:

① When the source operand and destination operand are specified as the same device:

The source operand and destination operand can also be specified as the same device number.

In this case, if continuous execution type instructions (SUB, DSUB) are used, the result of the subtraction operation will change every operation cycle. Please be careful.

② The difference between the SUB(P) instruction and the -(P) instruction and DEC(P) instruction executed by the -1 subtraction program

SUB(P) instruction every time X1 changes from OFF to ON, the program of D0 content -1 is similar to -(P) instruction and DEC(P) instruction described later, but the following contents are different.

			SUB/SUBP/DSUB/DSUBP instructions	DEC/DECP/DDEC/DDECP instructions
Sign bit (zero, borrow, carry)			Action	No action
Calculation result	16-bit operation result	(S)-(+1)=(d)	←-2←-1←0←-32768	-32768→32767→32766
		(S)-(-1)=(d)	+32767→0→+1→+2→	——
	32-bit operation result	(S)-(+1)=(d)	←-2←-1←0←-2147483648	-2147483648→2147483647→2147483646
		(S)-(-1)=(d)	+2147483647→0→+1→+2→	——

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Subtract 100000 from the data in (D1,D0), and store the result of the operation in (D3,D2), that is, (D1,D0)-10000 → (D3,D2).

6.3.5 MUL/16-bit multiplication

MUL(P)

Multiply the BIN16 bits specified in (s1) with the BIN16 bits specified in (s2), and store the result in the device specified in (d).

-[MUL (s1) (s2) (d)]

Content, range and data type

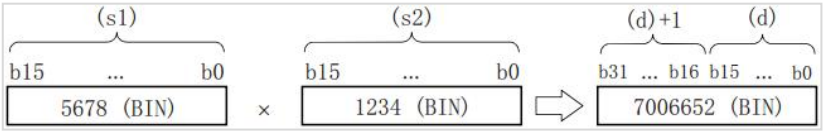
Parameter	Content	Range	Data type	Data type (label)
(s1)	Multiplication operation data or the device storing multiplication operation data	-32768 to 32767	Signed BIN 16 bit	ANY16_S
(s2)	Multiplication operation data or the device storing multiplication operation data	-32768 to 32767	Signed BIN 16 bit	ANY16_S
(d)	Device for storing calculation results		Signed BIN 32 bit	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MUL	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

Multiply the BIN 16-bit data specified in (s1) with the BIN 16-bit data specified in (s2), and store the result of the operation in the device specified in (d).



(d) is the multiplication result in the case of bit device

• K1: lower 4 bits (b0 to b3)

• K4: Lower 16 bits (b0 to b15)

• K8: Lower 32 bits (b0 to b31)

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Multiply the data in (D0) by (D2), and store the operation result in (D5, D4), that is, (D0) × (D2) → (D5, D4).

6.3.6 DMUL/32-bit Multiplication

DMUL(P)

Multiply the 32-bit BIN specified in (s1) and the 32-bit BIN specified in (s2), and store the result in the device specified in (d).

-[DMUL (s1) (s2) (d)]

Content, range and data type

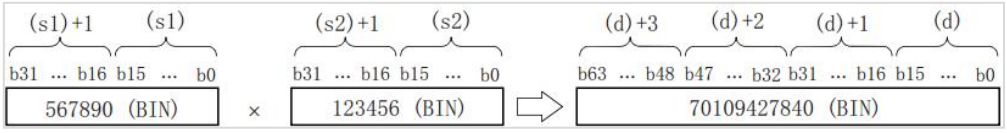
Parameter	Content	Range	Data type	Data type (label)
(s1)	Multiplication operation data or device storing multiplication operation data	-2147483648 to 2147483647	Signed BIN 32 bit	ANY32_S
(s2)	Multiplication operation data or device storing multiplication operation data	-2147483648 to 2147483647	Signed BIN 32 bit	ANY32_S
(d)	Device for storing calculation results		Signed BIN64 bit	ANY64_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DMUL	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

Multiply the BIN32-bit data specified in (s1) and the BIN32-bit data specified in (s2), and store the result of the operation in the device specified in (d).



(d) is the multiplication result in the case of bit device

• K1: lower 4 bits (b0 to b3)

• K4: Lower 16 bits (b0 to b15)

• K8: Lower 32 bits (b0 to b31)

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Multiply the data in (D1, D0) by (D3, D2), and store the result of the operation in ((D7, D6), (D5, D4)), ie (D1, D0) × (D3, D2) → ((D7, D6), (D5, D4)).

6.3.7 DIV/16-bit division operation

DIV(P)

Divide the BIN 16-bit data specified in (s1) with the BIN 16-bit data specified in (s2), and store the result in the device specified in (d).

-[DIV (s1) (s2) (d)]

Content, range and data type

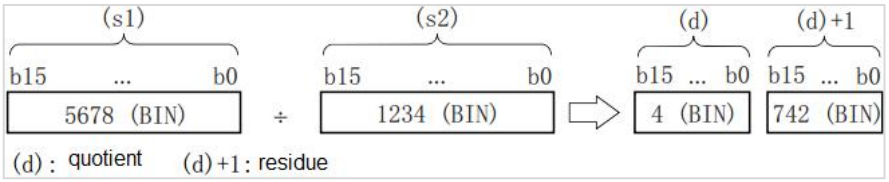
Parameter	Content	Range	Data type	Data type (label)
(s1)	Division operation data or device storing division operation data	-32768 to 32767	Signed BIN 16 bit	ANY16_S
(s2)	Division operation data or device storing division operation data	-32768 to 32767	Signed BIN 16 bit	ANY16_S
(d)	Device for storing calculation results		Signed BIN 32 bit	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DIV	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

Divide the BIN 16-bit data specified in (s1) with the BIN 16-bit data specified in (s2), and store the result of the operation in the device specified in (d).



In the case of a word device, the division result uses a 32-bit storage quotient and remainder, and in the case of a bit device, only a 16-bit storage quotient is used.

• Quotient is stored in the lower 16 bits.

• The remainder is stored in the upper 16 bits. (Can only be stored in the case of word devices.)

✎Note

① About the opearation result

• The highest bit of the quotient and remainder represents the sign of positive (0) and negative (1).

• When one of (s1) or (s2) is negative, the quotient becomes negative. When (s1) is negative, the remainder becomes negative.

② The device specified by (d)

• With the digit specification function, when specifying a bit device, the remainder cannot be obtained.

Error code

Error code	Content
4080H	The input of divisor (s2) is 0
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Divide the data in (D0) by (D2), and store the result of the calculation: the quotient is stored in (D4), and the remainder is stored in

(D5), ie (D0)/ (D2) → (D4(quotient)) (D5(remainder)).

6.3.8 DDIV/32-bit division operation

DDIV(P)

Divide the BIN32-bit data specified in (s1) with the BIN32-bit data specified in (s2), and store the result in the device specified in (d).

-[DDIV (s1) (s2) (d)]

Content, range and data type

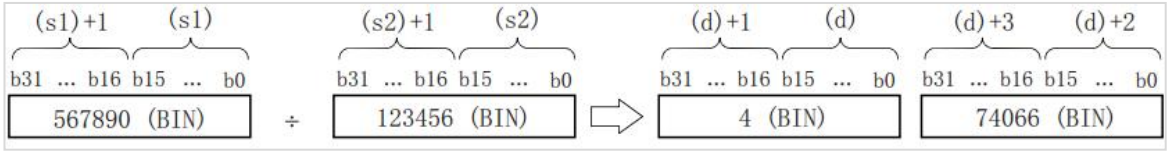
Parameter	Content	Range	Data type	Data type (label)
(s1)	Division operation data or device storing division operation data	-2147483648 to 2147483647	Signed BIN 32 bit	ANY32_S
(s2)	Division operation data or device storing division operation data	-2147483648 to 2147483647	Signed BIN 32 bit	ANY32_S
(d)	Device for storing calculation results		Signed BIN64 bit	ANY64_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DDIV	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

Divide the BIN32-bit data specified in (s1) with the BIN32-bit data specified in (s2), and store the result of the operation in the device specified in (d).



In the case of word devices, the division result uses BIN64 bits to store the quotient and remainder. In the case of bit devices, only the BIN 32-bit storage quotient is used.

✎Note:

① About the operation result

• The highest bit of the quotient and remainder represents the sign of positive (0) and negative (1).

• When one of (s1) or (s2) is negative, the quotient becomes negative. When (s1) is negative, the remainder becomes negative.

② The specified device of (d)

• With the digit specification function, when a bit device is specified, the remainder cannot be obtained.

Error code

Error code	Content
4080H	The input of divisor (s2) is 0
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Divide the data in (D1, D0) by (D3, D2), and store the result of the calculation: the quotient is stored in (D5, D4), and the remainder is stored in (D7, D6), that is (D1, D0)/ (D3, D2) → (D5, D4) (quotient) (D7, D6) (remainder).

6.3.9 INC/16-bit data increment

INC(P)

Add one to the device (BIN 16-bit data) specified in (d).

-[INC (d)]

Content, range and data type

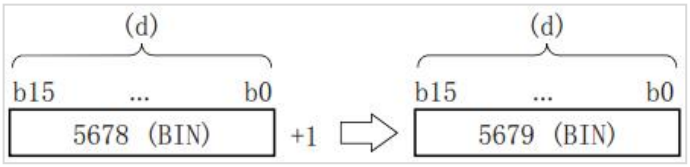
Parameter	Content	Range	Data type	Data type (label)
(d)	The word device number that stores the data added by one	-32768 to 32767	Signed BIN 16 bit	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
INC	Parameter 1												●	●	●	●	●	●	●	●						●	●

Features

Add one to the device (BIN 16-bit data) specified in (d).



• If the INC(P) instruction is executed when the content of the device specified in (d) is 32767, -32768 will be stored in the device specified in (d).

• Flags (zero, borrow, carry) do not perform actions.

✎Note:

If the continuous execution (INC) instruction is used, the addition operation will be performed every operation cycle, so care should be taken.

Error code

Error code	Content
4085H	The output results of (d) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Add one to the device value specified in D0, that is, (D0) + 1 → (D0).

6.3.10 DINC/32-bit data increment

DINC(P)

Add one to the device (BIN 32-bit data) specified in (d).

-[DINC (d)]

Content, range and data type

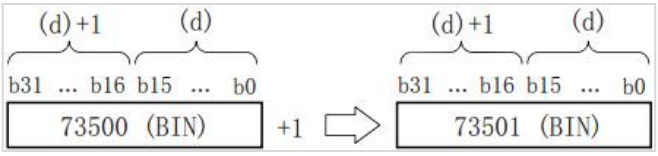
Parameter	Content	Range	Data type	Data type (label)
(d)	The word device number that stores the data added by one	-2147483648 to 2147493647	Signed BIN 32 bit	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DINC	Parameter 1												●	●	●	●	●	●	●	●	●	●				●	●

Features

Add one to the device (BIN 32-bit data) specified in (d).



• When the DINC(P) instruction is executed when the content of the device specified in (d) is 2147483647, -2147483648 will be stored in the device specified in (d).

• Flags (zero, borrow, carry) do not perform actions.

✎Note:

If the continuous execution (INC) instruction is used, the addition operation will be performed every operation cycle, so care should be taken.

Error code

Error code	Content
4085H	The output results of (d) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Add one to the device value specified in (D1, D0), that is, (D1, D0) + 1 → (D1, D0).

6.3.11 DEC/16 bit data decrement

DEC(P)

Minus one for the device (BIN 16-bit data) specified in (d).

-[DEC (d)]

Content, range and data type

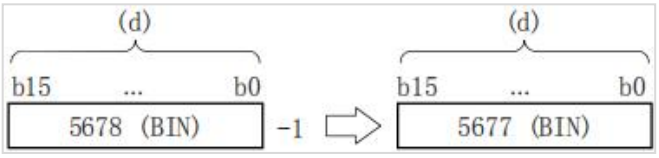
Parameter	Content	Range	Data type	Data type (label)
(d)	The word device number that stores the data minus by one	-32768 to 32767	Signed BIN 16 bit	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEC	Parameter 1												●	●	●	●	●	●	●	●						●	●

Features

Minus one for the device (BIN 16-bit data) specified in (d).



• If the DEC(P) instruction is executed when the content of the device specified in (d) is -32768, 32767 will be stored in the device specified in (d).

• Flags (zero, borrow, carry) do not perform actions.

✎**Note:**

If using continuous execution (DEC) instructions, subtraction will be performed every operation cycle, so care should be taken.

Error code

Error code	Content
4085H	The output results of (d) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Each time M0 is set, the value of the device specified in D0 will be -1, (D0)-1 → (D0).

6.3.12 DDEC/32-bit data decrement

DDEC(P)

Minus one for the device (BIN 32-bit data) specified in (d).

-[DDEC (d)]

Content, range and data type

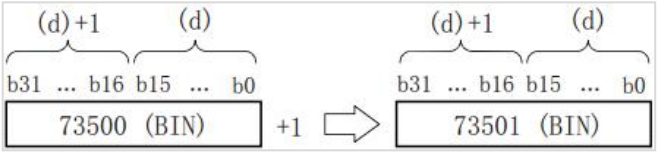
Parameter	Content	Range	Data type	Data type (label)
(d)	The word device number that stores the data minus by one	-2147483648 to 2147483647	Signed BIN 32 bit	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DDEC	Parameter 1												●	●	●	●	●	●	●	●	●	●				●	●

Features

Minus one for the device (BIN 32-bit data) specified in (d).



If the DDEC(P) instruction is executed when the content of the device specified in (d) is 0, minus one will be stored in the device specified in (d).

• Flags (zero, borrow, carry) do not perform actions.

✎Note:

If using continuous execution (DEC) instructions, subtraction will be performed every operation cycle.

Error code

Error code	Content
4085H	The output results of (d) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Minus one on the device value specified in (D1, D0), that is, (D1, D0)-1 → (D0).

6.4 Logic operation instructions

6.4.1 NEG/16-bit complement

NEG(P)

After inverting the sign of the BIN 16-bit device specified in (d), store it in the device specified in (d).

-[NEG (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	The start device that stores the data complement of 2	-32768 to 32767	Signed BIN16	ANY16_S

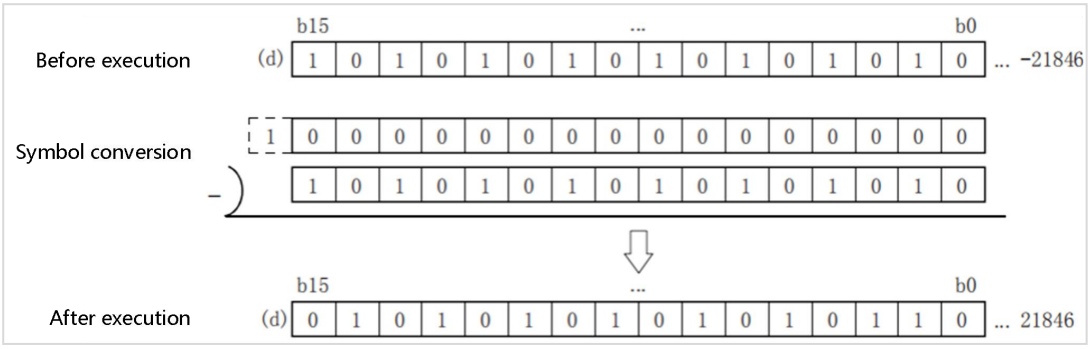
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
NEG	Parameter 1											●	●	●	●	●	●	●	●	●						●	●

Features

• Invert the sign of the BIN 16-bit device specified in (d), and store it in the device specified in (d).

• Used when inverting positive and negative signs.



✎**Note:** If the continuous execution (NEG) instruction is used, every operation cycle will be inverted, so care should be taken.

Error code

Error code	Content
4085H	The output results of (d) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example

In the two examples below, if D2=K4 and D4=K8, or D2=K8 and D10 is always K4.

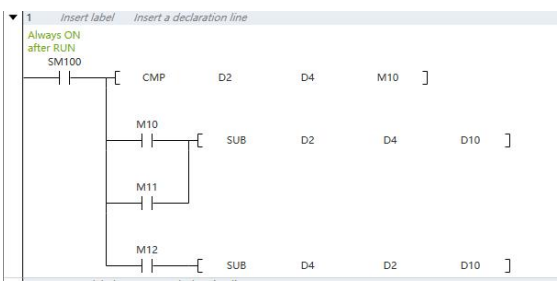
Each time M0 is set, the device value specified in D0 is reversed.



Take the absolute value of the difference of the subtraction operation.

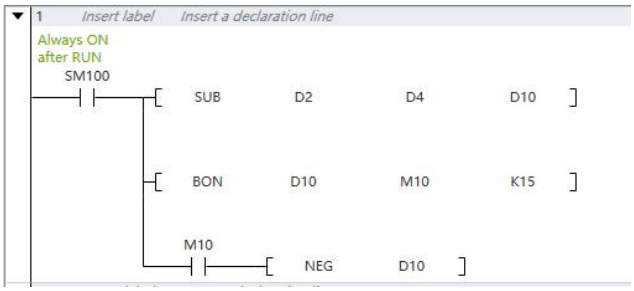
If D2>D4, M10=On. If D2=D4, M11=On. If D2 <D4, M12=On. This ensures that D10 is positive.

It can also be represented by the following program:



When bit15 of D10 is "1" (indicating that D10 is a negative number), M10 = On, use NEG instruction to complement D10 to obtain the absolute value of D10.

In the above two examples, if D2=K4, D4=K8; or D2=K8, D4=K4, the result of D10 is K4.



6.4.2 DNEG/32-bit complement

DNEG(P)

After inverting the sign of the BIN 32-bit device specified in (d), store it in the device specified in (d).

-[DNEG (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	The start device that stores the data complement of 2	-2147483648 to 2147483647	Signed BIN16	ANY16_S

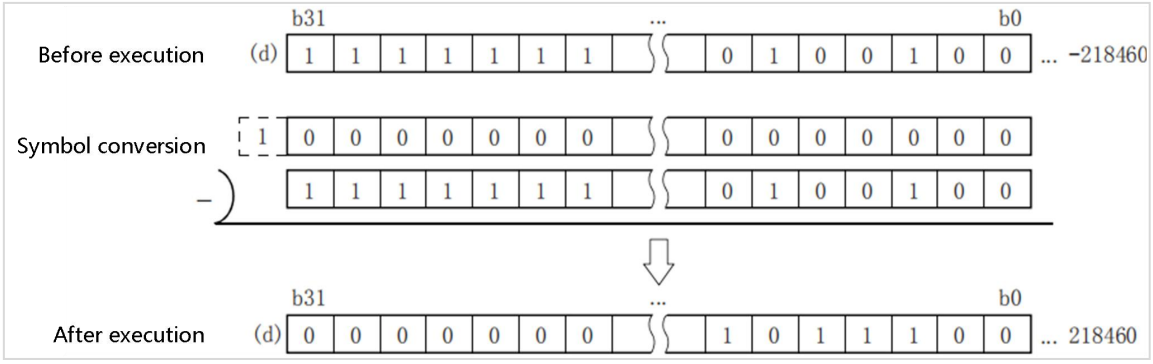
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DNEG	Parameter 1											●	●	●	●	●	●	●	●	●	●	●				●	●

Features

• Invert the sign of the BIN 32-bit device specified in (d) and store it in the device specified in (d).

• Used when inverting positive and negative signs.



✎Note:

If you use continuous execution (DNEG) instructions, every operation cycle will be inverted, so care should be taken.

Error code

Error code	Content
4085H	The output results of (d) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



Each time M0 is set, the device value specified in (D1, D0) is reversed.

6.4.3 WOR/16-bit data logical OR

WOR(P)

Perform a logical OR operation on the BIN 16-bit data of the device specified in (s1) and the BIN 16-bit data of the device specified in (s2), and store the result in the device specified in (d).

-[WOR (s1) (s2) (d)]

Content, range and data type

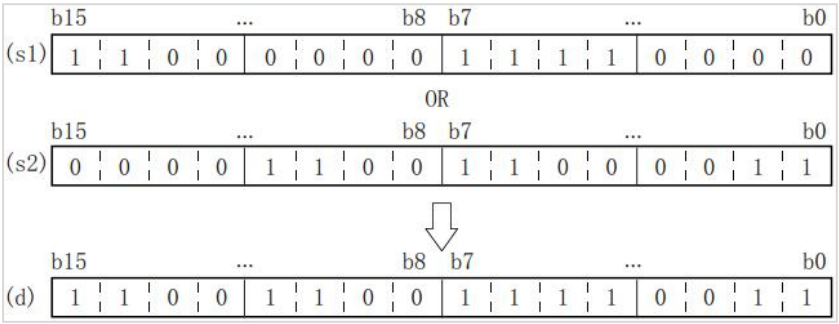
Parameter	Content	Range	Data type	Data type (label)
(s1)	Stores data for logical OR operation or a device that stores data	-32768 to 32767	Signed BIN16	ANY16_S
(s2)	Stores data for logical OR operation or a device that stores data	-32768to 32767	Signed BIN16	ANY16_S
(d)	Device for storing logic or result		Signed BIN16	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
WOR	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

Features

• Perform a logical OR operation on the BIN 16-bit data of the device specified in (s1) and the BIN 16-bit data of the device specified in (s2), and store the result in the device specified in (d).



In the case of bit devices, bit devices after the number of points specified by the number of digits will be calculated as 0.

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



When M0 is set, (D0) and (D2) are logically performed, and the value is stored in (D4), that is (D0)∨(D2) → (D4)

6.4.4 DOR/32-bit data logical OR

DOR(P)

After inverting the sign of the BIN 32-bit device specified in (d), store it in the device specified in (d).

-[DOR (s1) (s2) (d)]

Content, range and data type

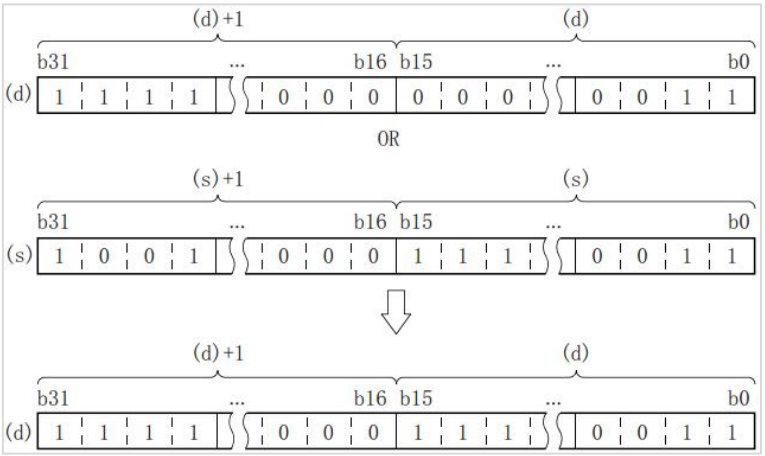
Parameter	Content	Range	Data type	Data type (label)
(s1)	Stores data for logical OR operation or a device that stores data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(s2)	Stores data for logical OR operation or a device that stores data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(d)	Device for storing logic or result		Signed BIN32	ANY32_S

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DOR	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

Perform a logical OR operation on the BIN 32-bit data of the device specified in (s1) and the BIN 32-bit data of the device specified in (s2), and store the result in the device specified in (d).



In the case of bit devices, bit devices after the number of points specified by the number of digits will be calculated as 0.

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



When M0 is set, (D1, D0) and (D3, D2) are logically performed, and the value is stored in (D5, D4), that is, (D1, D0)∨(D3, D2) → (D5, D4)).

6.4.5 WAND/16-bit data logic AND

WAND(P)

Perform a logical AND operation on each bit of the BIN 16-bit data of the device specified in (s1) and the BIN 16-bit data of the device specified in (s2), and store the result in the device specified in (d).

-[WAND (s1) (s2) (d)]

Content, range and data type

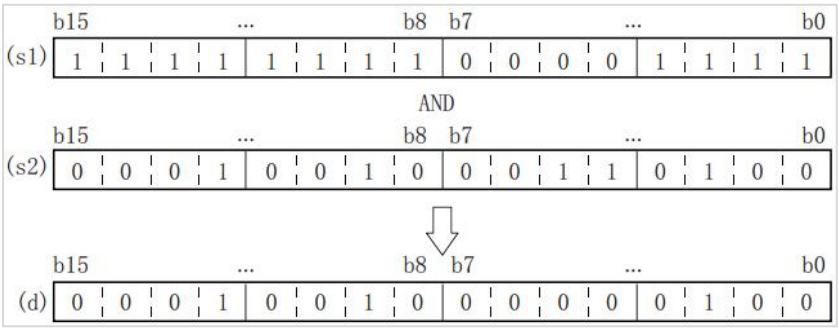
Parameter	Content	Range	Data type	Data type (label)
(s1)	Store the data for logical AND operation or the device storing the data	-32768to 32767	Signed BIN16	ANY16_S
(s2)	Store the data for logical AND operation or the device storing the data	-32768 to 32767	Signed BIN16	ANY16_S
(d)	Device for storing logic and result		Signed BIN16	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
WAND	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

Features

Perform a logical AND operation on each bit of the BIN 16-bit data of the device specified in (s1) and the BIN 16-bit data of the device specified in (s2), and store the result in the device specified in (d).



In the case of bit devices, bit devices after the number of points specified by the number of digits will be calculated as 0.

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



When M0 is set, the logical AND operation of (D0) and (D2) is performed, and the value is stored in (D4), that is, (D0) ∧ (D2) → (D4).

6.4.6 DAND/32-bit data logic AND

DAND(P)

Perform a logical AND operation on each bit of the BIN 32-bit data of the device specified in (s1) and the BIN 32-bit data of the device specified in (s2), and store the result in the device specified in (d).

-[DAND (s1) (s2) (d)]

Content, range and data type

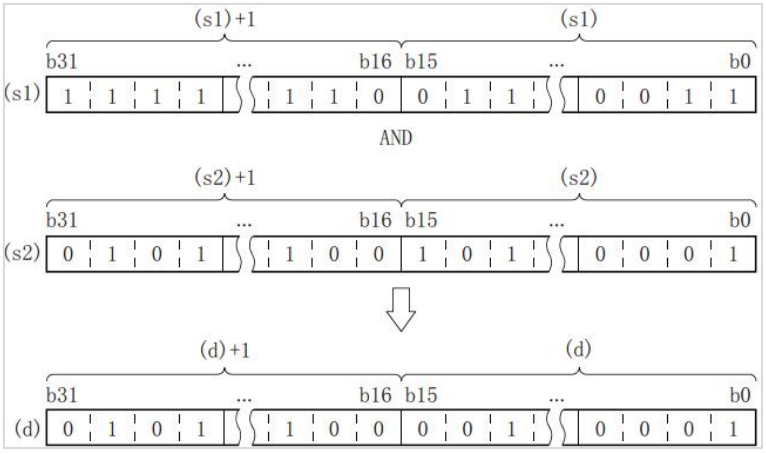
Parameter	Content	Range	Data type	Data type (label)
(s1)	Store the data for logical AND operation or the device storing the data	-2147483648 to +2147483647	Signed BIN32	ANY32_S
(s2)	Store the data for logical AND operation or the device storing the data	-2147483648 to +2147483647	Signed BIN32	ANY32_S
(d)	Device for storing logic and result		Signed BIN32	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DAND	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

Perform a logical AND operation on each bit of the BIN 32-bit data of the device specified in (s1) and the BIN 32-bit data of the device specified in (s2), and store the result in the device specified in (d).



In the case of bit devices, bit devices after the number of points specified by the number of digits will be calculated as 0.

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



When M0 is set, perform logical AND operation of (D1, D0) and (D3, D2), and store the value in (D5, D4), (D1, D0) ∧ (D3, D2) → (D5, D4) .

6.4.7 WXOR/16-bit data logic exclusive OR

WXOR(P)

Perform an exclusive OR operation on the BIN 16-bit data of the device specified in (s1) and the BIN 16-bit data of the device specified in (s2), and store the result in the device specified in (d).

-[WXOR (s1) (s2) (d)]

Content, range and data type

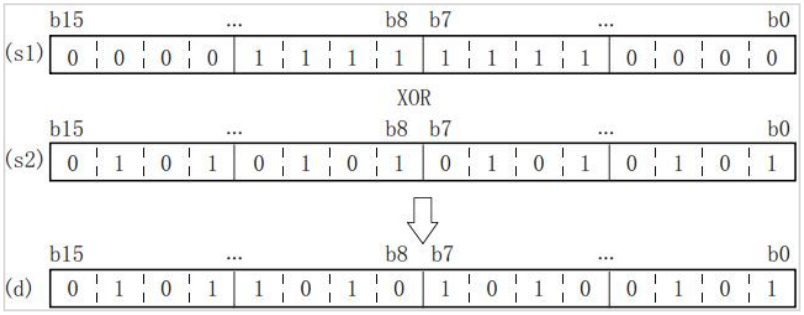
Parameter	Content	Range	Data type	Data type (label)
(s1)	Store the data for exclusive OR operation or the device storing the data	-32768 to 32767	Signed BIN16	ANY16_S
(s2)	Store the data for exclusive OR operation or the device storing the data	-32768 to +32767	Signed BIN16	ANY16_S
(d)	Device for storing XOR result		Signed BIN16	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
WXOR	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

Features

• Perform logical exclusive OR operation on the BIN 16-bit data of the device specified in (s1) and the BIN 16-bit data of the device specified in (s2), and store the result in the device specified in (d).



In the case of bit devices, bit devices after the number of points specified by the number of digits will be calculated as 0.

Error code

Error code	Content
4085H	The output results of (s1) and (s2) in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example

Example 1: When M0 is set, (D0) and (D2) are XOR operation, and the value is stored in (D4), (D0)∀(D2)→(D4).



Example 2: When used with the CML instruction, it can realize the logic exclusive OR (XORNOT) operation:



6.4.8 DXOR/32-bit data logic exclusive OR

DXOR(P)

Perform an exclusive OR operation on the BIN 32-bit data of the device specified in (s1) and the BIN 32-bit data of the device specified in (s2), and store the result in the device specified in (d).

-[DXOR (s1) (s2) (d)]

Content, range and data type

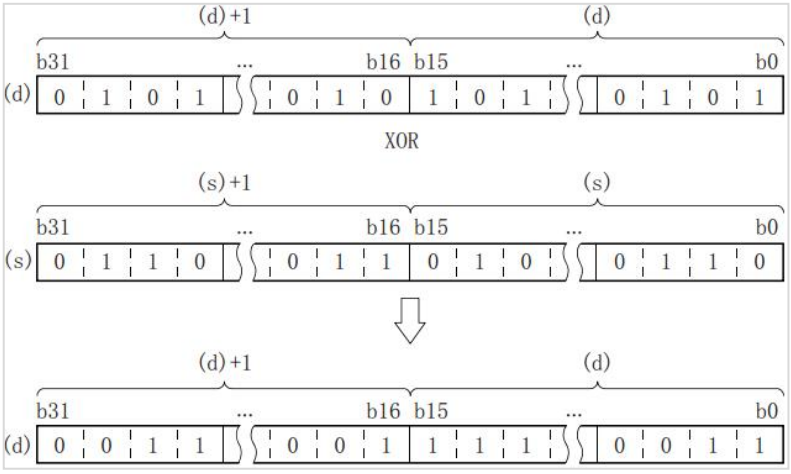
Parameter	Content	Range	Data type	Data type (label)
(s1)	Store the data for exclusive OR operation or the device storing the data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(s2)	Store the data for exclusive OR operation or the device storing the data	-2147483648 to 2147483647	Signed BIN32	ANY32_S
(d)	Device for storing XOR result		Signed BIN32	ANY32_S

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DXOR	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

Perform an exclusive OR operation on the BIN 32-bit data of the device specified in (s1) and the BIN 32-bit data of the device specified in (s2), and store the result in the device specified in (d).



In the case of bit devices, bit devices after the number of points specified by the number of digits will be calculated as 0.

Error code

Error code	Content
4085H	The output results of (s1) and (s2)in the read application instruction exceed the device range
4086H	The output result of (d) in the write application instruction exceeds the device range

Example



When M0 is set, (D1, D0) and (D3, D2) are XOR operation, and the value is stored in (D5, D4), that is, (D1, D0) ∀ (D3, D2) → (D5, D4))

6.4.9 ANS/Alarm settings

ANS(P)

Used to set alarm instructions.

-[ANS (s) (n) (d)]

Content, range and data type

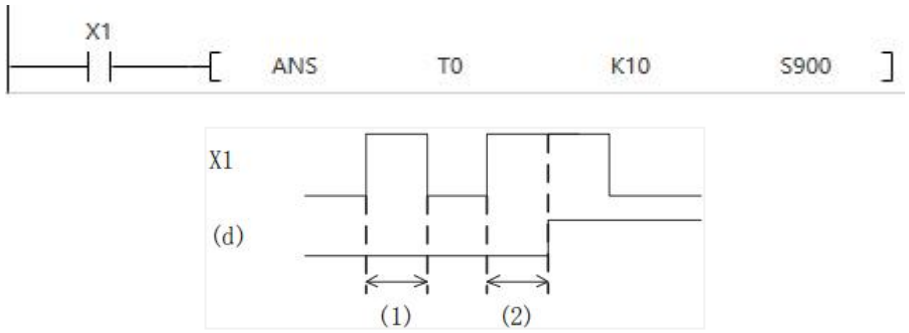
Parameter	Content	Range	Data type	Data type (label)
(s)	Timer number for judging time	-	Signed BIN 16 bit	ANY16
(n)	Data that judges time	1 to 32767	Signed BIN 16 bit	ANY16
(d)	The set alarm device	-	Bit	ANY16_BOOL

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ANS	Parameter 1															●										●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3				●																					●	●

Features

When the instruction input continues to be ON for the judgment time [(n)×100ms, timer (s)], set (d). If the instruction time turns off below the judgment time [(n)×100ms], the current value of the judgment timer (s) is reset, and (d) is not set. In addition, if the instruction input turns off, the judgment timer will be reset.



① Judge the time ((n)X 100ms or less)

② Judgment time or more (inclusive) ((n) X 100ms or more (inclusive))

Related device

Devices	Name	Content
SM249	Signal alarm is valid	After SM249 is ON, the following SM248 and SD249 act.
SM248	Signal alarm action	SM249 is ON, when any one of the states S900 to S999 is active, SM248 is ON
SD249	Signal alarm ON state minimum number	Save the smallest number of actions in S900 to S999.

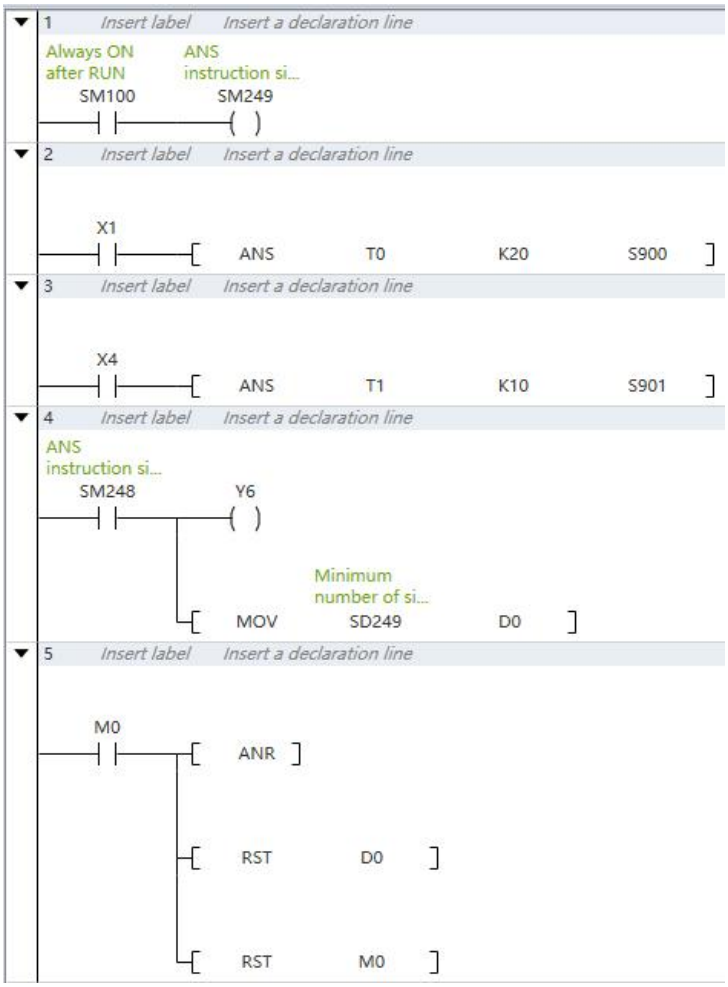
Error code

Error code	Content
4084H	The value specified in (n1) and (n2) exceeds the range of 0 to 32767
	The timer number is not in the range of T0 to T199.
	The signal alarm is not in the range of S900 to S999.
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

The fault number is displayed by the signal alarm.

As shown below, when you write a program for diagnosing external faults, such as monitoring the content of SM249 (the smallest number in the ON state), the smallest number in the ON state among S900 to S999 will be displayed. When multiple faults occur at the same time, the next fault number can be obtained after eliminating the fault with the smallest number.



Monitoring is effective after SM249 is turned ON

Detect X1 for 2 seconds, turn ON, set S900

X4 is detected for 1 second, turn ON, set S901

SM248 will act after any one of S900 to S999 is ON, and the output fault display YY6 will act

Display the fault number to the D0 device

Through the external fault diagnosis program, use the reset button M0 to turn off the activated state. Each time M0 turns ON, the action status of the new number is set in turn, and the new number that is already ON is reset.

6.4.10 ANR/Alarm reset

ANR(P)

The instruction to reset the small number that is ON in the alarm.

-[ANR]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
No	No parameter setting	-	-	-

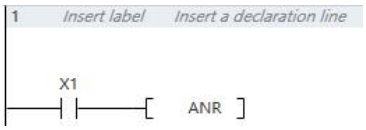
Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ANR	No	No object device																								

Features

If the instruction input is ON, reset the active alarm in the alarm.

If multiple alarms are operating, reset the smaller number. If the input instruction is turned ON again, the next small number in the alarm that is operating will be reset.



Related device

Devices	Name	Content
SM249	Signal alarm is valid	After SM249 is ON, the following SM248 and SD249 act.
SM248	Signal alarm action	SM249 is ON, when any one of the states S900 to S999 is active, SM248 is ON.
SD249	Signal alarm ON state minimum number	Save the smallest number of actions in S900 to S999.

✎Note:

If you use the ANR instruction, reset in sequence every cycle.

If the ANRP instruction is used, it will be executed in only one operation cycle.

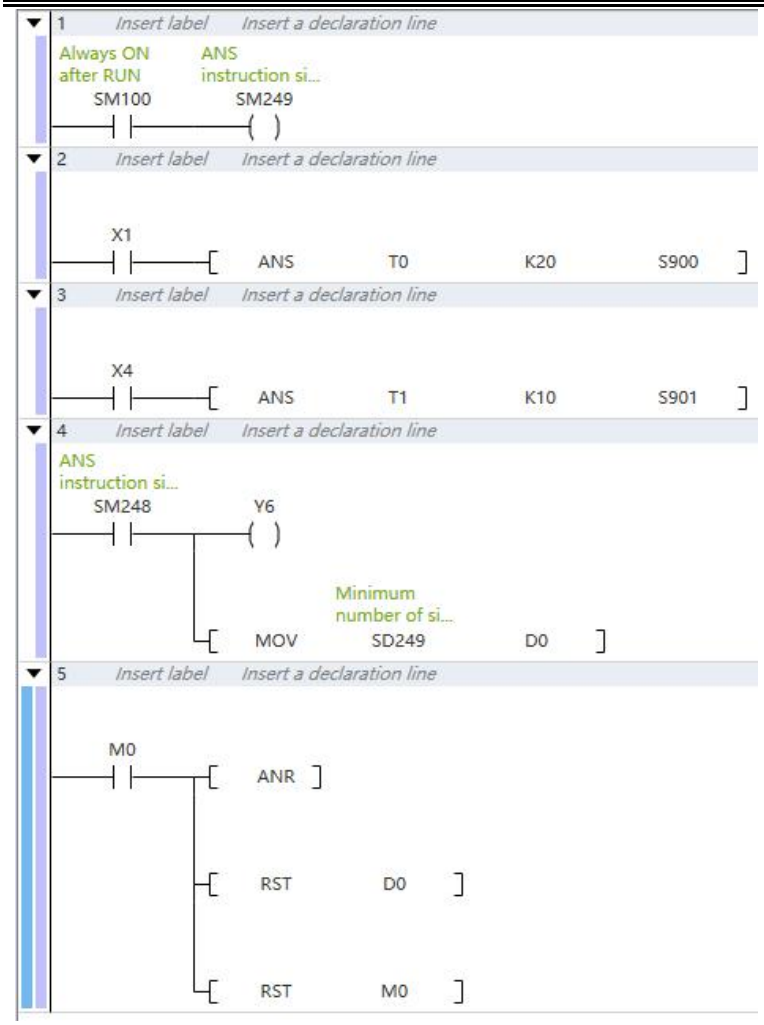
Error code

No operation error.

Example

The fault number is displayed by the signal alarm.

As shown below, when you write a program for diagnosing external faults, such as monitoring the content of SM249 (the smallest number in the ON state), the smallest number in the ON state among S900 to S999 will be displayed. When multiple faults occur at the same time, the next fault number can be obtained after eliminating the fault with the smallest number.



Monitoring is effective after SM249 is turned ON

Detect X1 for 2 seconds, turn ON, set S900

X4 is detected for 1 second, turn ON, set S901

SM248 will act after any one of S900 to S999 is ON, and the output fault display YY6 will act

Display the fault number to the D0 device

Through the external fault diagnosis program, use the reset button M0 to turn off the activated state. Each time M0 turns ON, the action status of the new number is set in turn, and the new number that is already ON is reset.

6.4.11 BON/16-bit data bit judgment

BON(P)

Check whether the state of the BIN 16-bit data (n) bit of the device specified in (s) is ON or OFF, and output the result to the device specified in (d).

-[BON (s) (n) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Data storage destination word device number	-	Signed BIN 16 bit	ANY16
(d)	Bit device number of drive	-	Bit	ANY16_BOOL
(n)	The position of the bit to be judged	0 to 15	Signed BIN 16 bit	ANY16

Device used

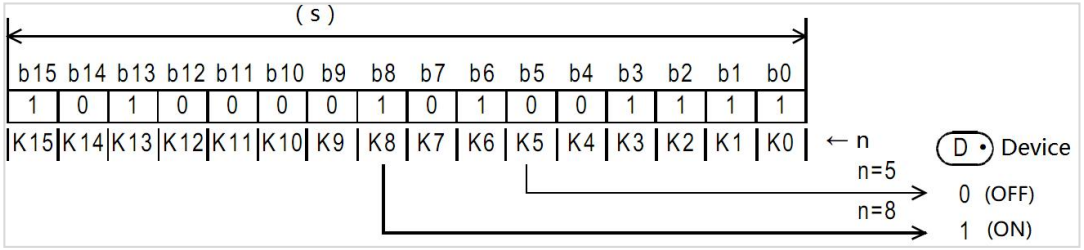
Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BON	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2		●	●	●	●					●							●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Check whether the state of the BIN 16-bit data (n) bit of the device specified in (s) is ON or OFF, and output the result to the device specified in (d).

If the above result is ON, then (d)=ON, if it is OFF, then (d)=OFF.

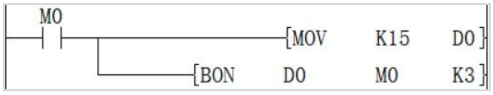
If a constant (K) is specified in the device specified in (s), it will be automatically converted to BIN.



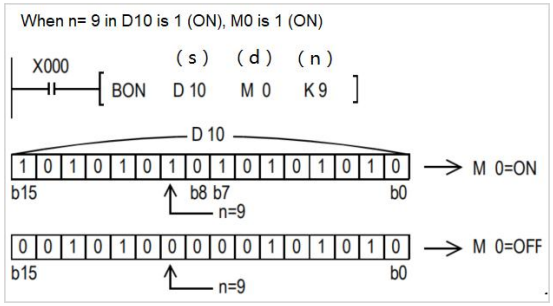
Error code

Error code	Content
4084H	The data input in (n) exceeds the specified range of 0 to 15.
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When n in D0 = the third bit is 1 (ON), M0 is set to 1 (ON).



6.4.12 DBON/32-bit data bit judgment

DBON(P)

Check whether the state of the BIN 32-bit data (n) bit of the device specified in (s) is ON or OFF, and output the result to the device specified in (d).

-[DBON (s) (n) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Data storage destination word device number	-	Signed BIN 32 bit	ANY32
(d)	Bit device number of drive	-	Bit	ANY32_BOOL
(n)	The position of the bit to be judged	0 to 31	Signed BIN 32 bit	ANY32

Device used

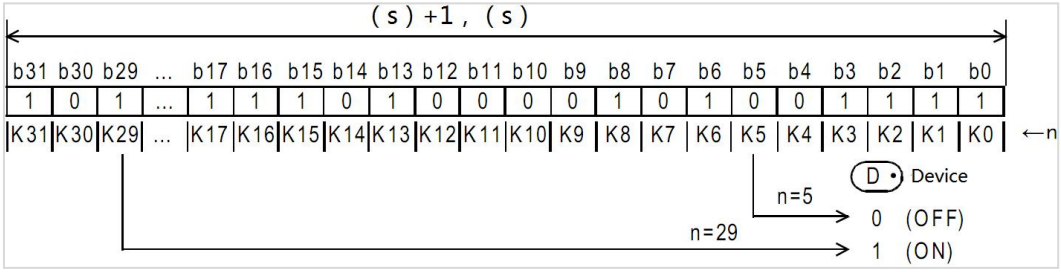
Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBON	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2		●	●	●	●					●							●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●

Features

Check whether the BIN 32-bit data (n) bit status of the device specified in (s) is ON or OFF, and output the result to the device specified in (d).

If the above result is ON, then (d)=ON, if it is OFF, then (d)=OFF.

If a constant (K) is specified in the device specified in (s), it will be automatically converted to BIN.

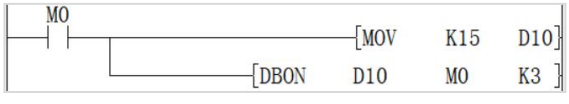


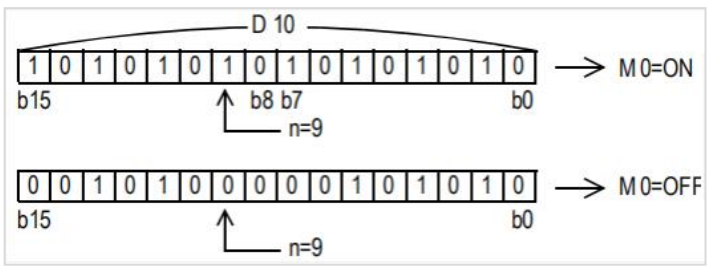
Error code

Error code	Content
4084H	The data input in (n) exceeds the specified range of 0 to 31.
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

When n in D0 = the third bit is 1 (ON), M0 is set to 1 (ON).





6.4.13 ENCO/Encode

ENCO(P)

Encode the data of the 2th (n)th power from (s) and store it in (d).

-[ENCO (s) (n) (d)]

Content, range and data type

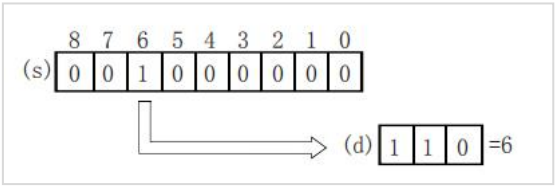
Parameter	Content	Range	Data type	Data type (label)
(s)	Start device for storing coded data	-	Bit/Signed BIN 16 bit	ANY_ELEMENTARY
(d)	Device number storing the encoding result	-	Signed BIN 16 bit	ANY_ELEMENTARY
(n)	Effective bit length	0 to 8	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ENCO	Parameter 1	●	●	●	●	●										●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

The BIN value corresponding to the bit from $2^{(n)}$ bits of (s) to 1 is stored in (d).



When (n)=0, it will be no processing, and the content of the device specified in (d) will not change.

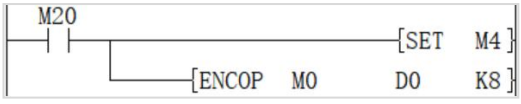
Bit devices are treated as 1 bit, and word devices are treated as 16 bits.

When multiple digits are 1, it will be processed at the upper position.

Error code

Error code	Content
4084H	In the bit device specification of (s), when (n) is other than 0 to 8.
	In the word device specification of (s), when (n) is other than 0 to 4.
	When the data of $2^{(n)}$ bits starting from (s) are all 0.
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M20 is turned ON, the D0 device is 16 after encoding.

6.4.14 DECO/Decode

DECO(P)

Decode the lower (n) bits of the device specified in (s), and store the result in the 2 (n)th power of the device specified in (d).

-[DECO (s) (n) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Decoded data or the device number storing the decoded data	-	Bit/Signed BIN 16 bit	ANY_ELEMENTARY
(d)	The start device storing the decoding result	-	Signed BIN 16 bit	ANY_ELEMENTARY
(n)	Effective bit length	0 to 8	Signed BIN 16 bit	ANY16

Device used

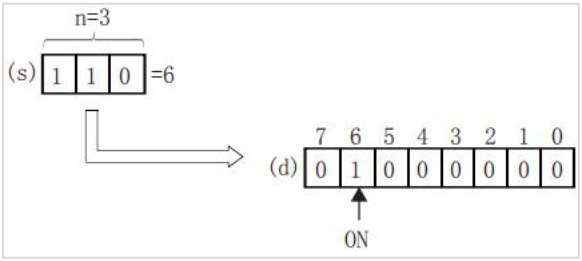
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DECO	Parameter 1	●	●	●	●	●							●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2		●	●	●	●										●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Turn ON the position of (d) corresponding to the BIN value specified in the lower (n) bit of (s).

When (n)=0, it will be no processing, and the content of the device specified in (d) will not change.

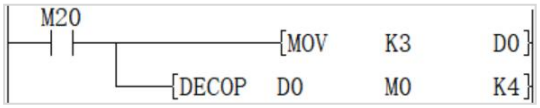
Bit devices are treated as 1 bit, and word devices are treated as 16 bits.



Error code

Error code	Content
4084H	In the bit device specification of (d), when (n) is other than 0 to 8.
	In the word device specification of (d), when (n) is other than 0 to 4.
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M20 is ON, M3 will be turned ON.

6.4.15 SUM/The ON bits of 16-bit data

SUM(P)

Store the total number of bits at 1 in the BIN 16-bit data of the device specified in (s) to the device specified in (d).

-[SUM (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number that counts the total number of bits at 1	-	Signed BIN 16 bit	ANY16
(d)	The device start number of the total number of storage bits	-	Signed BIN 16 bit	ANY16

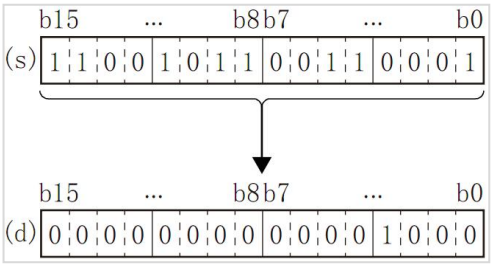
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SUM	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Features

Store the total number of bits at 1 in the BIN 16-bit data of the device specified in (s) to the device specified in (d).

When the BIN 16-bit data of the device specified in (s) is all 0, the zero flag (SM153) turns on



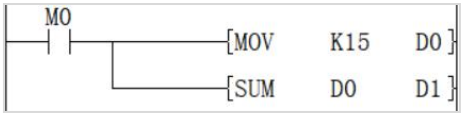
The total number of 1 (ON) is stored in BIN.

There are 8 in the example on the left.

Error code

Error code	Content
4085H	When the device specified in the read application instructions (s) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M0 is ON, the number of ON bits in D0 is counted and stored in D1. The value after D1 is executed is 4.

6.4.16 DSUM/The ON bits of 32-bit data

DSUM(P)

Store the total number of bits at 1 in the BIN 32-bit data of the device specified in (s) to the device specified in (d).

-[SUM (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number that counts the total number of bits at 1	-	Signed BIN 32 bit	ANY32
(d)	The device start number of the total number of storage bits	-	Signed BIN 32 bit	ANY32

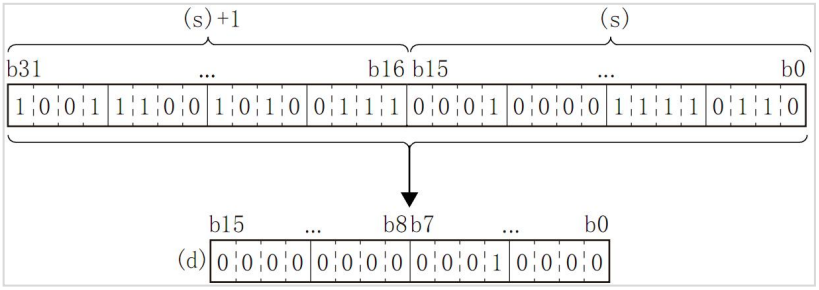
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSUM	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●

Features

Store the total number of bits at 1 in the BIN 32-bit data of the device specified in (s) to the device specified in (d).

When the BIN 32-bit data of the device specified in (s) is all 0 (OFF), the zero flag (SM153) turns on.



The total number of 1 (ON) is stored in BIN.

There are 16 in the example on the left.

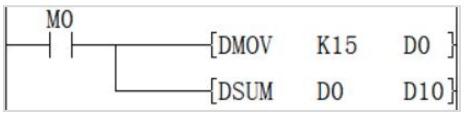
✎Note:

When the instruction input is OFF, the instruction will not be executed, and the output of the ON digits of the action will remain the same as before.

Error code

Error code	Content
4085H	When the device specified in the read application instructions (s) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M0 is ON, the number of ON bits in D0 is counted and stored in D10, and the value after D10 is executed is 4.

6.4.17 MEAN/Average value of 16-bit data

MEAN(P)

Store the total number of bits at 1 in the BIN 16-bit data of the device specified in (s) to the device specified in (d).

-[MEAN (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number storing the data for average calculation	-	Signed BIN 16 bit	ANY16
(d)	The device start number storing the average value	-	Signed BIN 16 bit	ANY16
(n)	Number of data or the device number storing the number of data	1 to 32767	Signed BIN 16 bit	ANY16

Device used

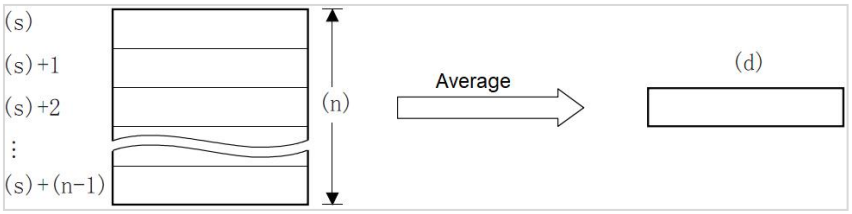
Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MEAN	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Calculate the average value of the 16-bit data at (n) points starting from the device specified in (s) and store it in the device specified in (d).

The total is calculated from the algebraic sum and divided by (n).

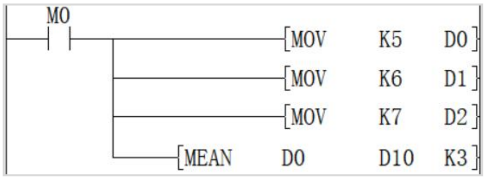
The remainder is rounded off.



Error code

Error code	Content
4084H	The data input by (n) in the application instruction exceeds the specifiable range. N≤0
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



Add the data of D0, D1, and D2 and save the value obtained after dividing by 3 in D10. The calculated average value is 6.

6.4.18 DMEAN/Average value of 16-bit data

DMEAN(P)

Store the total number of bits at 1 in the BIN 32-bit data of the device specified in (s) to the device specified in (d).

-[DMEAN (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number storing the data for average calculation	-	Signed BIN 32 bit	ANY32
(d)	The device start number storing the average value	-	Signed BIN 32 bit	ANY32
(n)	Number of data or the device number storing the number of data	1 to 2147483647	Signed BIN 32 bit	ANY32

Device used

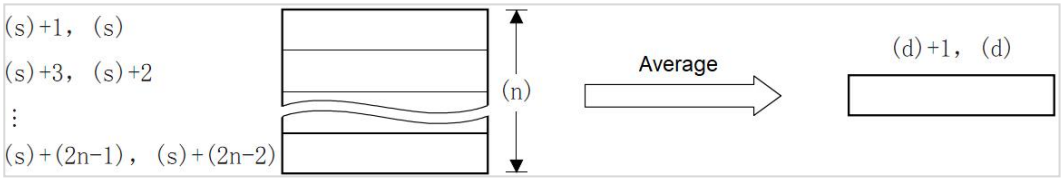
Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DMEAN	Parameter 1											●	●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 3											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●

Features

Calculate the mean value of BIN 32-bit data at (n) points starting from the device specified in (s) and store it in the device specified in (d).

The total is calculated from the algebraic sum and divided by (n).

The remainder is rounded off.



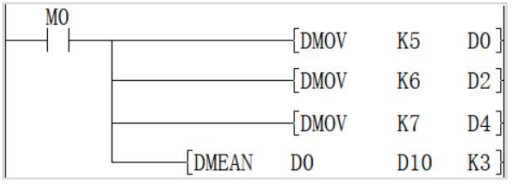
✎Note:

When the device number exceeds, (n) is handled as a smaller value within the allowable range.

Error code

Error code	Content
4084H	The data input in (n) exceeds the specifiable range. N≤0
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



Add the data of D0, D2, and D4, and save the value obtained after dividing by 3 in D10 and D11, and the calculated average value is 6.

6.4.19 SQR/16-bit square root

SQR(P)

Calculate the square root of the BIN 16-bit data specified in (s), and store the calculation result in (d).

-[SQR (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The data device storing for square root calculation	0 to +32767	Signed BIN 16 bit	ANY16
(d)	The device storing the calculated square root	-	Signed BIN 16 bit	ANY16

Device used

Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SQR	Parameter 1																	●	●	●			●	●		●	●
	Parameter 2																	●	●	●						●	●

Features

Calculate the square root of the BIN 16-bit data specified in (s), and store the calculation result in (d).

$$\sqrt{(s)} \rightarrow (d)$$

✎Note:

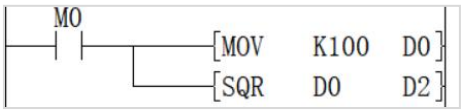
The decimal point of operation result will be rounded off and become an integer. If rounding occurs, SM152 (borrow flag) turns ON.

When the operation result is really 0, SM153 (zero flag) turns ON.

Error code

Error code	Content
4084H	When a negative value is specified in (s).
4085H	When the device specified in the read application instructions (s) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



The square root of D0 is stored in D2, and the value of D0 is 100, so the value of D2 is 10.

6.4.20 DSQR/32-bit square root

DSQR(P)

Calculate the square root of the BIN 32-bit data specified in (s), and store the calculation result in (d).

-[DSQR (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The data device storing for square root calculation	0 to 2147483647	Signed BIN 32 bit	ANY32
(d)	The device storing the calculated square root	-	Signed BIN 32 bit	ANY32

Device used

Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSQR	Parameter 1																	●	●	●	●	●	●	●		●	●
	Parameter 2																	●	●	●	●	●				●	●

Features

Calculate the square root of the BIN 32-bit data specified in (s) and store the calculation result in (d).

$$\sqrt{(s)+1,\ (s)} \rightarrow (d)+1,\ (d)$$

✎Note:

The decimal point of operation result will be rounded off and become an integer. If rounding occurs, SM152 (borrow flag) turns ON.

When the operation result is really 0, SM153 (zero flag) turns on.

Error code

Error code	Content
4084H	When a negative value is specified in (s).
4085H	When the device specified in the read application instructions (s) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



The square root of D0 is stored in D2, and the value of D0 is 110, so the value in the D2 device is 10 (the fractional part is discarded), and the borrow flag SM152 is turned ON.

6.4.21 WSUM/The sum value of 16-bit data

WSUM(P)

After adding all the BIN 16-bit data of point (n) starting from the device specified in (s), it is stored in the device specified in (d).

-[WSUM (s) (d) (n)]

Content, range and data type

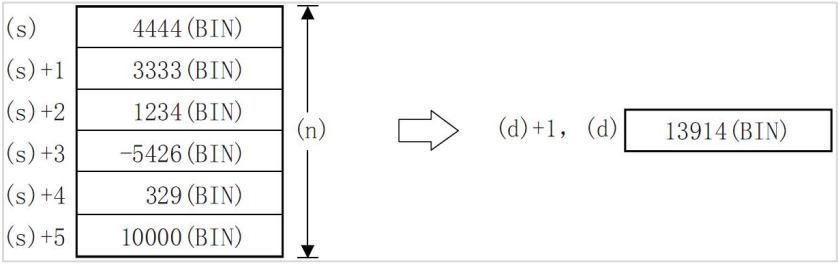
Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number storing the data for sum value calculation	-	Signed BIN 16 bit	ANY16
(d)	The device start number storing the sum value	-	Signed BIN 32 bit	ANY32
(n)	Number of data	-	Signed BIN 16 bit	ANY16

Device used

Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
WSUM	Parameter 1															●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

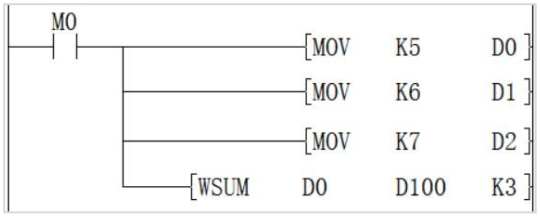
After adding all the BIN 16-bit data of point (n) starting from the device specified in (s), it is stored in the device specified in (d).



Error code

Error code	Content
4084H	When a negative value is specified in (n).
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M0=ON, the total of 16-bit data of D0 to D2 is saved in [D100, D101], and the accounting result is 18.

6.4.22 DWSUM/The sum value of 32-bit data

DWSUM(P)

Add all the 32-bit BIN data of point (n) starting from the device specified in (s) and store it in the device specified in (d).

-[DWSUM (s) (d) (n)]

Content, range and data type

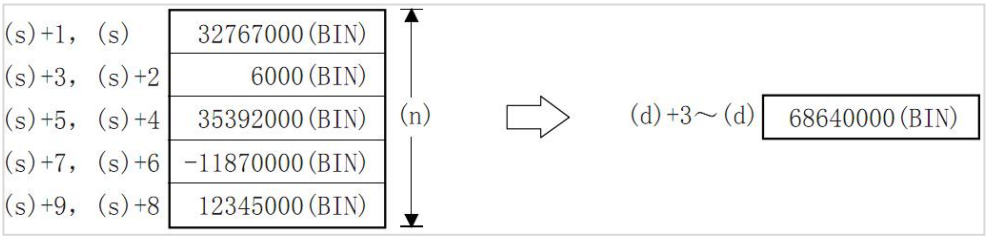
Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number storing the data for total value calculation	-	Signed BIN 32 bit	ANY32
(d)	The device start number storing the total value	-	Signed BIN64 bit	ANY64
(n)	Number of data	-	Signed BIN 32 bit	ANY32

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DWSUM	Parameter 1															●	●	●	●	●	●	●				●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 3											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●

Features

Add all the 32-bit BIN data of point (n) starting from the device specified in (s) and store it in the device specified in (d).



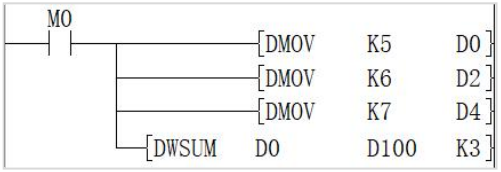
✎Note:

When the number of bits is specified in (d), the value of n ranges from 1 to 8, such as K8 (32-bit instructions, such as K8M0) without K16 (64-bit instructions).

Error code

Error code	Content
4084H	When a negative value is specified in (n).
4085H	When the device specified in the read application instructions (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M0=ON, the total of 16-bit data of D0 to D2 is saved in [D100, D101], and the accounting result is 18.

6.4.23 SORT/16-bit data sorting

SORT

Sort the data rows in ascending order based on the group data of column (n3) in the BIN 16-bit data table (sorting source) of (n1×n2) points specified in (s) and store them in the specified in (d) (N1×n2) points in the BIN 16-bit data table (after sorting).

-[SORT (s) (n1) (n2) (d) (n3)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device number storing the data table	-	Signed BIN 16 bit	ANY16
(n1)	Number of data (rows)	1 to 32	Signed BIN 16 bit	ANY16
(n2)	Number of group data (columns)	1 to 6	Signed BIN 16 bit	ANY16
(d)	The start device number storing the operation result	-	Signed BIN 16 bit	ANY16
(n3)	The column number of the group data (column) as the sorting basis	-	Signed BIN 16 bit	ANY16

Device used

Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SORT	Parameter 1															●	●	●	●	●						●	
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 4															●	●	●	●	●						●	
	Parameter 5											●	●	●	●	●	●	●	●	●			●	●		●	

Features

The BIN 16-bit data table (sorting source) of (n1×n2) points specified in (s), based on the group data of column (n3), sort the data rows in ascending order, and store them in (d). The (n1×n2) point of the BIN 16-bit data table (after sorting).

Take (n1)=K3, (n2)=K4 in the sort source as an example, the data table structure is as follows. In the case of a sorted data table, (s) should be replaced with (d).

		Number of groups (n2) ((n2)=K4)			
		Column NO. 1	Column NO. 2	Column NO. 3	Column NO. 4
		Management number	Height	Weight	Age
When the number of data (n1)=3	Line NO.1	(s)	(s) +3	(s) +6	(s) +9
	Line NO.2	(s)+1	(s) +4	(s) +7	(s) +10
	Line NO.3	(s)+2	(s) +5	(s) +8	(s) +11

Data alignment starts when instruction input is ON, data alignment ends after (n1) scan, instruction execution end flag SM229 is set to ON. According to the source data sorted as follows, an example of the operation is shown below. In addition, by putting serial numbers such as management numbers in the first column in advance, the original row number can be judged based on the content, which is very convenient.

		Number of groups (n2) ((n2)=K4)			
		Column NO. 1	Column NO. 2	Column NO. 3	Column NO. 4
		Management number	Height	Weight	Age
When the number of data (n1) = 5	Line NO.1	(s)	(s) +5	(s) +10	(s) +15
		1	150	45	20
	Line NO.2	(s)+1	(s) +6	(s) +11	(s) +16
		2	180	50	40
	Line NO.3	(s)+2	(s) +7	(s) +12	(s) +17
		3	160	70	30
	Line NO.4	(s) +3	(s) +8	(s) +13	(s) +18
		4	100	20	8
	Line NO.5	(s) +4	(s) +9	(s) +14	(s) +19
		5	150	50	45

Press (n3)=K2 (column number 2) to execute the sorting result.

		Number of groups (n2) ((n2)=K4)			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	Height	Weight	Age
When the number of data (n1) = 5	Line NO.1	(d)	(d) +5	(d) +10	(d) +15
		4	100	20	8
	Line NO.2	(d) +1	(d) +6	(d) +11	(d) +16
		1	150	45	20
	Line NO.3	(d) +2	(d) +7	(d) +12	(d) +17
		5	150	50	45
	Line NO.4	(d) +3	(d) +8	(d) +13	(d) +18
		3	160	70	30
	Line NO.5	(d) +4	(d) +9	(d) +14	(d) +19
		2	180	50	40

Press (n3)=K3 (column number 3) to execute the sorting result.

		Number of groups (n2) ((n2)=K4)			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	Height	Weight	Age
When the number of data (n1) = 5	Line NO.1	(d)	(d) +5	(d) +10	(d) +15
		4	100	20	8
	Line NO.2	(d) +1	(d) +6	(d) +11	(d) +16
		1	150	45	20
	Line NO.3	(d) +2	(d) +7	(d) +12	(d) +17
		2	180	50	40
	Line NO.4	(d) +3	(d) +8	(d) +13	(d) +18
		5	150	50	45
	Line NO.5	(d) +4	(d) +9	(d) +14	(d) +19
		3	160	70	30

✎**Note:**

Only ascending order is supported by SORT instruction.

Do not change the operand and data content during operation.

When executing again, the instruction input should be turned OFF once.

SORT instruction can drive at most one in the program.

When the same device is specified in (s) and (d), the source data is rewritten to the sorted data order. Please pay special attention not to change the content of (s) before the end of execution.

Error code

Error code	Content
4084H	When the value specified in (n1) exceeds the range of 1 to 32
	When the value specified in (n2) exceeds the range of 1 to 6
	When the value specified in (n3) exceeds the range of 1 to n2
4085H	When the device specified in read application instruction (s), (n1), (n2)and (n3) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range
4087H	When the (d) parameter in the application instruction uses an unsupported device
4089H	The number of application instructions exceeds the limit.

Example

Refer to the function description example.



6.4.24 SORT2/16-bit data sorting

SORT2(P)

Sort the data rows in ascending or descending order based on the group data in column (n3), and store them in (d), based on the BIN 16-bit data table (sorting source) of (n1×n2) points specified in (s) In the BIN 16-bit data table (after sorting) of the specified (n1×n2) points.

-[SORT2 (s) (n1) (n2) (d) (n3)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device number storing the data table	-	Signed BIN 16 bit	ANY16
(n1)	Number of data (rows)	1 to 32	Signed BIN 16 bit	ANY16
(n2)	Number of group data (columns)	1 to 6	Signed BIN 16 bit	ANY16
(d)	The start device number storing the operation result	-	Signed BIN 16 bit	ANY16
(n3)	The column number of the group data (column) as the sorting basis	-	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SORT2	Parameter 1															●	●	●	●	●						●	
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 4															●	●	●	●	●						●	
	Parameter 5											●	●	●	●	●	●	●	●	●			●	●		●	

Features

Sort the data rows in ascending or descending order based on the group data in column (n3) and store them in (d) (N1×n2) point specified in the BIN 16-bit data table (after sorting).

Take (n1)=K3, (n2)=K4 in the sort source as an example, the data table structure is as follows. In the case of a sorted data table, (s) should be replaced with (d).

		When the number of groups (n2) (n2) = K4			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	Height	Weight	Age
When the number of data (n1)=3	Line NO.1	(s)	(s)+1	(s) +2	(s) +3
	Line NO.2	(s) +4	(s) +5	(s) +6	(s) +7
	Line NO.3	(s) +8	(s) +9	(s) +10	(s) +100

Sequence is set by the ON/OFF status of SM165

	Sort order setting instruction
SM165=ON	Descending
SM165=OFF	Ascending

Data alignment starts when instruction input is ON, data alignment ends after (n1) scan, instruction execution end flag SM229 is set to ON.

According to the source data sorted as follows, an example of the operation is shown below. In addition, by putting serial numbers such as management numbers in the first column in advance, the original row number can be judged based on the content, which is very convenient.

		When the number of groups (n2) (n2) = K4			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	Height	Weight	Age
When the number of data (n1) = 5	Line NO.1	(s)	(s)+1	(s) +2	(s) +3
		1	150	45	20
	Line NO.2	(s) +4	(s) +5	(s) +6	(s) +7
		2	180	50	40
	Line NO.3	(s) +8	(s) +9	(s) +10	(s) +100
		3	160	70	30
	Line NO.4	(s) +12	(s) +13	(s) +14	(s) +15
		4	100	20	8
	Line NO.5	(s) +16	(s) +17	(s) +18	(s) +19
		5	150	50	45

Press (n3)=K2 (column number 2) to execute the sorting result (SM165=OFF in the case of ascending order)

		When the number of groups (n2) (n2) = K4			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	Height	Weight	Age
When the number of data (n1) = 5	Line NO.1	(d)	(d) +1	(d) +2	(d) +3
		4	100	20	8
	Line NO.2	(d) +4	(d) +5	(d) +6	(d) +7
		1	150	45	20
	Line NO.3	(d) +8	(d) +9	(d) +10	(d) +100
		5	150	50	45
	Line NO.4	(d) +12	(d) +13	(d) +14	(d) +15
		3	160	70	30
	Line NO.5	(d) +16	(d) +17	(d) +18	(d) +19
		2	180	50	40

Press (n3)=K3 (column number 3) to execute the sorting result (SM165=ON in the case of ascending order)

		When the number of groups (n2) (n2) = K4			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	Height	Weight	Age
When the number of data (n1) = 5	Line NO.1	(d)	(d) +1	(d) +2	(d) +3
		3	160	70	30
	Line NO.2	(d) +4	(d) +5	(d) +6	(d) +7
		2	180	50	40
	Line NO.3	(d) +8	(d) +9	(d) +10	(d) +100
		5	150	50	45
	Line NO.4	(d) +12	(d) +13	(d) +14	(d) +15
		1	150	45	20
	Line NO.5	(d) +16	(d) +17	(d) +18	(d) +19
		4	100	20	8

✎Note:

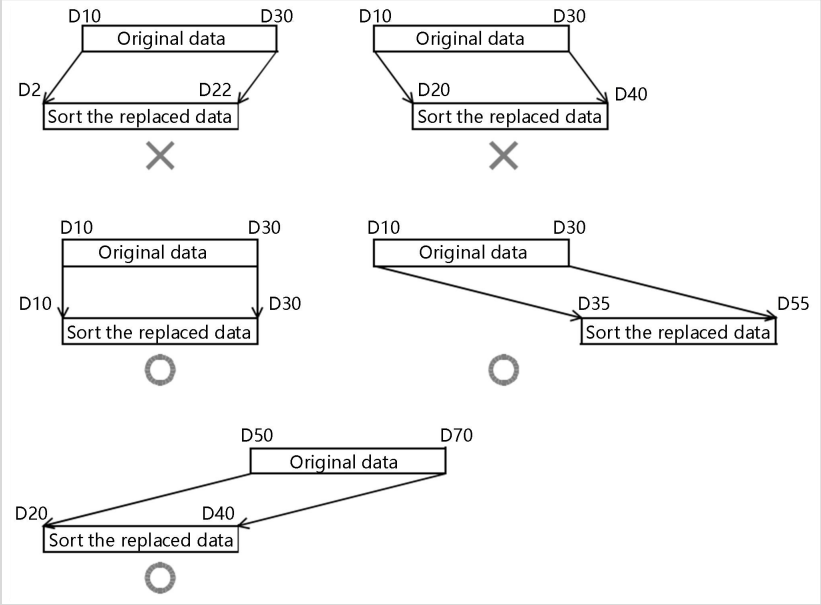
Do not change the operand and data content during operation.

When executing again, the instruction input should be turned OFF once.

The SORT2 instruction can only be written in the program to drive 2 at most.

When the same device is specified in (s) and (d), the source data is rewritten to the sorted data order. Please pay special attention not to change the content of (s) before the end of execution.

Do not overlap the source data and the sorted data.

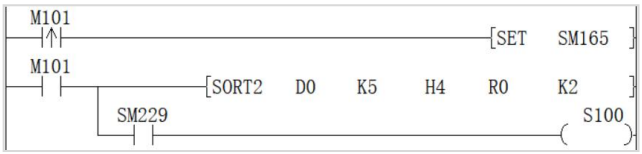


Error code

Error code	Content
4084H	When the value specified in (n1) exceeds the range of 1 to 32
	When the value specified in (n2) exceeds the range of 1 to 6
	When the value specified in (n3) exceeds the range of 1 to n2
4085H	When the device specified in read application instruction (s), (d), (n1), (n2) and (n3) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range
4089H	The number of application instructions exceeded the limit.

Example

Refer to the function description example.



6.4.25 DSORT2/32-bit data sorting

DSORT2(P)

Sort the data rows in ascending or descending order based on the group data of column (n3) in the BIN 32-bit data table (sorting source) of (n1×n2) points specified in (s) and store them in (d) The specified (n1×n2) point BIN 32-bit data table (after sorting).

-[DSORT2 (s) (n1) (n2) (d) (n3)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device number storing the data table	-	Signed BIN 32 bit	ANY32
(n1)	Number of data (rows)	1 to 32	Signed BIN 32 bit	ANY32
(n2)	Number of group data (columns)	1 to 6	Signed BIN 32 bit	ANY32
(d)	The start device number storing the operation result	-	Signed BIN 32 bit	ANY32
(n3)	The column number of the group data (column) as the sorting basis	-	Signed BIN 32 bit	ANY32

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSORT2	Parameter 1															●	●	●	●	●	●	●				●	
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
	Parameter 3											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
	Parameter 4															●	●	●	●	●	●	●				●	
	Parameter 5											●	●	●	●	●	●	●	●	●	●	●	●	●		●	

Features

Sort the data rows in ascending or descending order based on the group data in the (n3) column of the (n1×n2) point BIN 32-bit data table (sorting source) specified in (s), and store to (d) (N1×n2) specified in the BIN 32-bit data table (after sorting).

Take (n1)=K3, (n2)=K4 in the sort source as an example, the data table structure is as follows. In the case of a sorted data table, (s) should be replaced with (d).

		When the number of groups (n2) (n2) = K4			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	Height	Weight	Age
When the number of data (n1)=3	Line NO.1	(s)+1, (s)	(s)+3, (s)+2	(s)+5, (s)+4	(s) +7, (s) +6
	Line NO.2	(s) +9, (s) +8	(s)+11, (s)+10	(s) +13, (s) +12	(s) +15, (s) +14
	Line NO.3	(s) +17, (s) +16	(s) +19, (s) +18	(s) +21, (s) +20	(s) +23, (s) +22

Sequence is set by the ON/OFF status of SM165

	Sort order setting instructions
SM165=ON	Descending
SM165=OFF	Ascending

Data alignment starts when instruction input is ON, data alignment ends after (n1) scan, instruction execution end flag SM229 is set to ON.

According to the source data sorted as follows, an example of the operation is shown below. In addition, by putting serial numbers such as management numbers in the first column in advance, the original row number can be judged based on the content, which is very convenient.

		When the number of groups (n2) (n2) = K4			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	height	body weight	age
When the number of data (n1) = 5	Line NO.1	(s)+1, (s)	(s)+3, (s)+2	(s)+5, (s)+4	(s) +7, (s) +6
		1	150	45	20
	Line NO.2	(s) +9, (s) +8	(s)+11, (s)+10	(s) +13, (s) +12	(s) +15, (s) +14
		2	180	50	40
	Line NO.3	(s) +17, (s) +16	(s) +19, (s) +18	(s) +21, (s) +20	(s) +23, (s) +22
		3	160	70	30
	Line NO.4	(s) +25, (s) +24	(s) +27, (s) +26	(s) +29, (s) +28	(s) +31, (s) +30
		4	100	20	8
	Line NO.5	(s) +33, (s) +32	(s) +35, (s) +34	(s) +37, (s) +36	(s) +39, (s) +38
		5	150	50	45

Press (n3)=K2 (column NO.2) to execute the sorting result (SM165=OFF in the case of ascending order)

		When the number of groups (n2) (n2) = K4			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	height	body weight	age
When the number of data (n1) = 5	Line NO.1	(s)+1, (s)	(s)+3, (s)+2	(s)+5, (s)+4	(s) +7, (s) +6
		4	100	20	8
	Line NO.2	(s) +9, (s) +8	(s)+11, (s)+10	(s) +13, (s) +12	(s) +15, (s) +14
		1	150	45	20
	Line NO.3	(s) +17, (s) +16	(s) +19, (s) +18	(s) +21, (s) +20	(s) +23, (s) +22
		5	150	50	45
	Line NO.4	(s) +25, (s) +24	(s) +27, (s) +26	(s) +29, (s) +28	(s) +31, (s) +30
		3	160	70	30
	Line NO.5	(s) +33, (s) +32	(s) +35, (s) +34	(s) +37, (s) +36	(s) +39, (s) +38
		2	180	50	40

Press (n3)=K3 (column NO.3) to execute the sorting result (SM165=ON in the case of ascending order)

		When the number of groups (n2) (n2) = K4			
		Column NO.1	Column NO.2	Column NO.3	Column NO.4
		Management number	height	body weight	age
When the number of data (n1) = 5	Line NO.1	(s)+1, (s)	(s)+3, (s)+2	(s)+5, (s)+4	(s) +7, (s) +6
		3	160	70	30
	Line NO.2	(s) +9, (s) +8	(s)+11, (s)+10	(s) +13, (s) +12	(s) +15, (s) +14
		2	180	50	40
	Line NO.3	(s) +17, (s) +16	(s) +19, (s) +18	(s) +21, (s) +20	(s) +23, (s) +22
		5	150	50	45
	Line NO.4	(s) +25, (s) +24	(s) +27, (s) +26	(s) +29, (s) +28	(s) +31, (s) +30
		1	150	45	20
	Line NO.5	(s) +33, (s) +32	(s) +35, (s) +34	(s) +37, (s) +36	(s) +39, (s) +38
		4	100	20	8

✎Note:

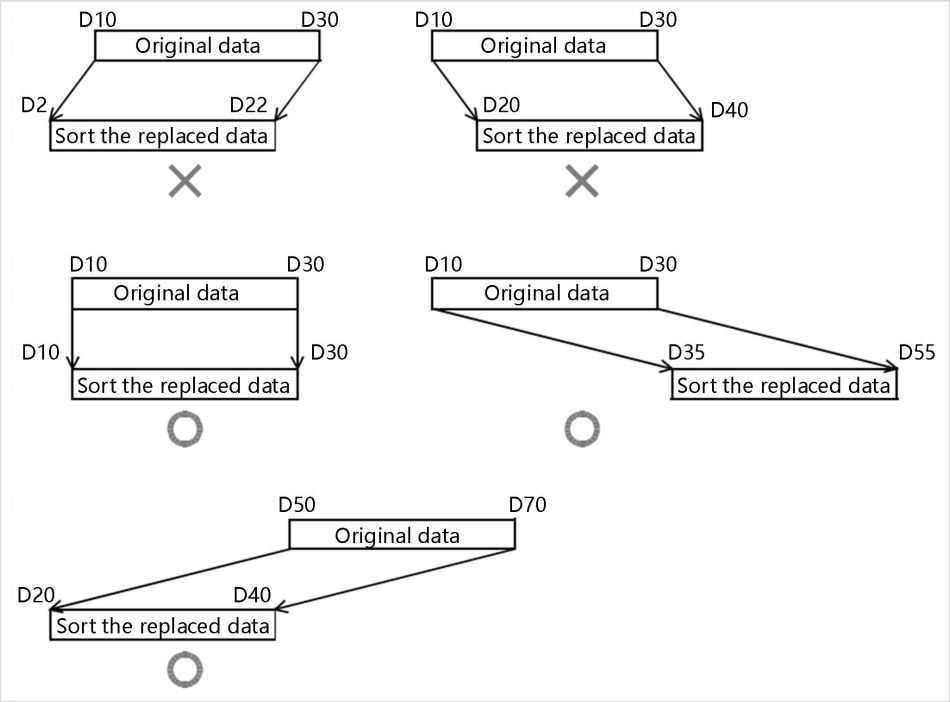
Do not change the operand and data content during operation.

When executing again, the instruction input should be turned OFF once.

The SORT2 instruction can only be written twice in the program.

When the same device is specified in (s) and (d), the source data is rewritten to the sorted data order. Please pay special attention not to change the content of (s) before the end of execution.

Do not overlap the source data and the sorted data.



Error code

Error code	Content
4084H	When the value specified in (n1) exceeds the range of 1 to 32
	When the value specified in (n2) exceeds the range of 1 to 6
	When the value specified in (n3) exceeds the range of 1 to n2
4085H	When the device specified in read application instruction (s), (d), (n1), (n2)and (n3) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range
4089H	The number of application instructions exceeded the limit.

Example

Refer to the function description example.



6.4.26 SWAP/16-bit data high and low byte swap

SWAP(P)

Swap the high and low 8-bit value of the device specified in (d).

-[SWAP (d)]

Content, range and data type

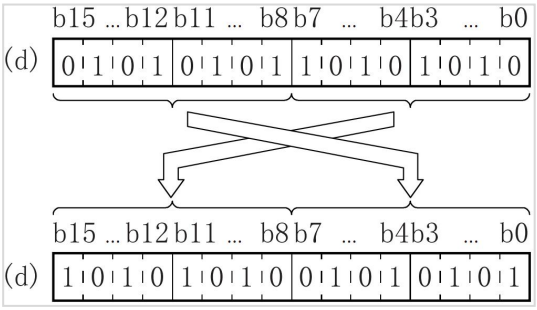
Parameter	Content	Range	Data type	Data type (label)
(d)	Word device with high and low byte swap	-	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SWAP	Parameter 1												●	●	●	●	●	●	●	●						●	●

Features

Convert the high and low 8-bit value of the device specified in (d).



Error code

Error code	Content
4085H	When the device specified in the read application instruction (d) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When the rising edge of M0 is triggered, swap the low 8 bits and high 8 bits of D0 to get H8F2A.

6.4.27 DSWAP/32-bit data high and low byte swap

DSWAP(P)

The devices specified in (d) and (d)+1 will be converted to the high and low 8-bit values respectively.

-[DSWAP (d)]

Content, range and data type

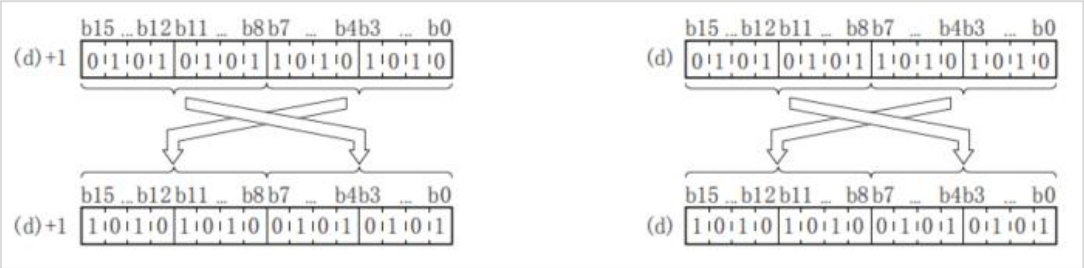
Parameter	Content	Range	Data type	Data type (label)
(d)	Word device with high and low byte swap	-	Signed BIN 32 bit	ANY32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSWAP	Parameter 1												●	●	●	●	●	●	●	●	●	●				●	●

Features

The devices specified in (d) and (d)+1 will be converted to the upper and lower 8-bit values respectively.



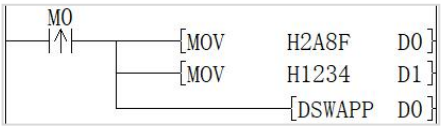
✎Note:

If continuous execution instructions are used, conversion will be performed every scan cycle.

Error code

Error code	Content
4085H	When the device specified in the read application instruction (d) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When the rising edge of M0 is triggered, the low 8 bits and the high 8 bits of D0 and D1 are swapped, and D0=H8F2A, D1=H3412 are obtained.



Devices	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D0	0	1	0	1	0	1	0	0	1	1	1	1	0	0	0	1	8F2A
D1	0	1	0	0	1	0	0	0	0	0	1	0	1	1	0	0	3412
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

6.4.28 BTOW/Byte unit data merge

BTOW(P)

Combine the low 8 bits of (n) bytes of BIN 16-bit data stored after the device number specified in (s) into word units and store it after the device number specified in (d).

-[BTOW (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device that stores the data merging in byte units	-	Signed BIN 16 bit	ANY16
(d)	The start device that stores the result of merging in byte units	-	Signed BIN 16 bit	ANY16
(n)	Number of byte data merged	0-32767	Signed BIN 16 bit	ANY16

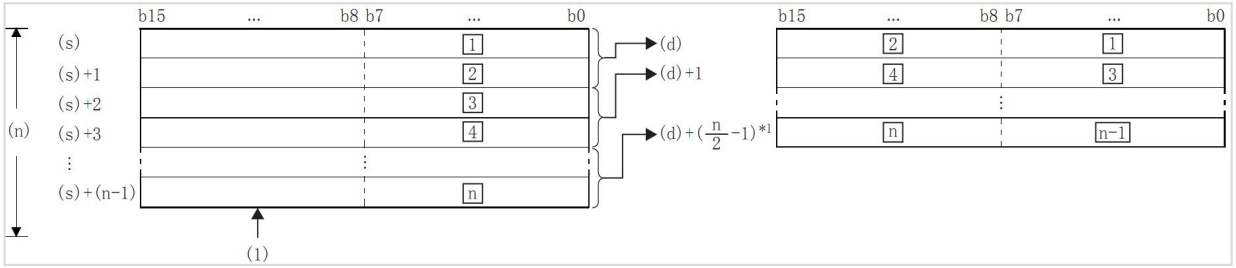
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BTOW	Parameter 1															●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

After the device number specified in (s), the lower 8 bits of the 16-bit BIN data stored in (n) bytes are combined into word units and stored in the device number specified in (d) or later.

The upper 8 bits of (n) word data stored after the device number specified in (s) will be ignored. In addition, when (n) is an odd number, 0 is stored in the upper 8 bits of the device storing the (n)th byte of data.



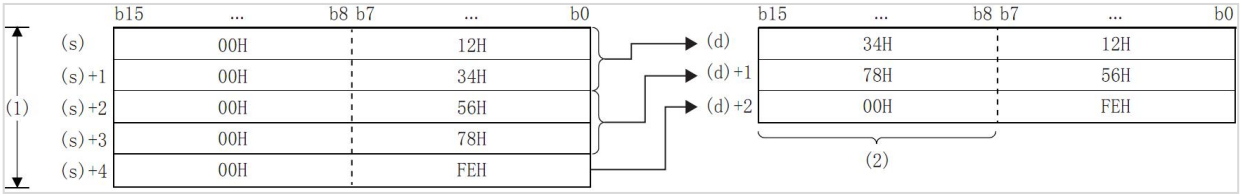
☐: the ☐th byte data;

(1): Ignore the high byte

*1: Carry below the decimal point.

Example

When (n)=5, the data up to the lower 8 bits of (s)+(s)+4 is stored in (d)+(d)+2.



(1): When (n)=5

(2): Change to 00H

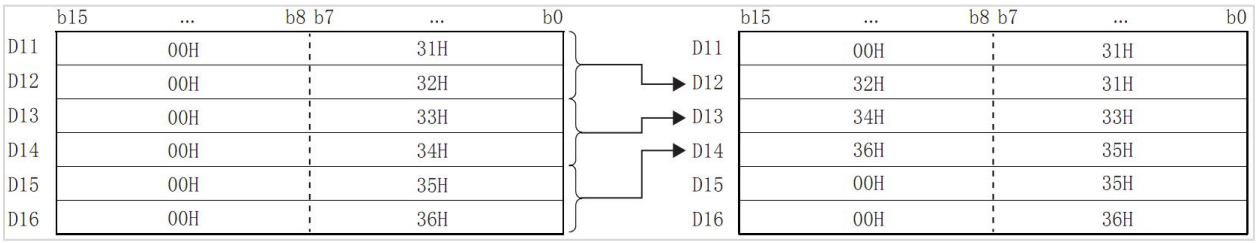
By setting the number of bytes in (n), the range of byte data specified in (s) and the range of the device storing the combined data specified in (d) will be automatically determined.

When the number of bytes specified in (n) is 0, no processing is performed.

The upper 8 bits of the byte data storage device specified in (s) will be ignored, and the lower 8 bits will be the target.

Example

When the low 8 bits of D11 to D16 is stored in D12 to D14.



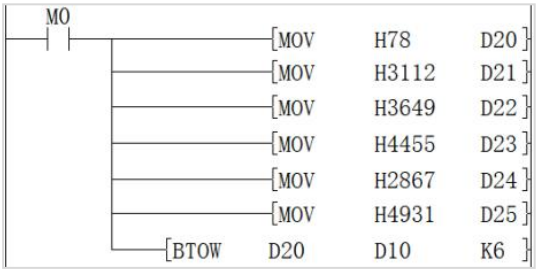
Even if the device range storing the data before merging overlaps the device rangestoring merged data, it will be handled as normal.

Device range storing the data before merging	Device range for storing merged data
(S)+0 to (s)+(n)-1	(D) to (d) + (n/2-1)

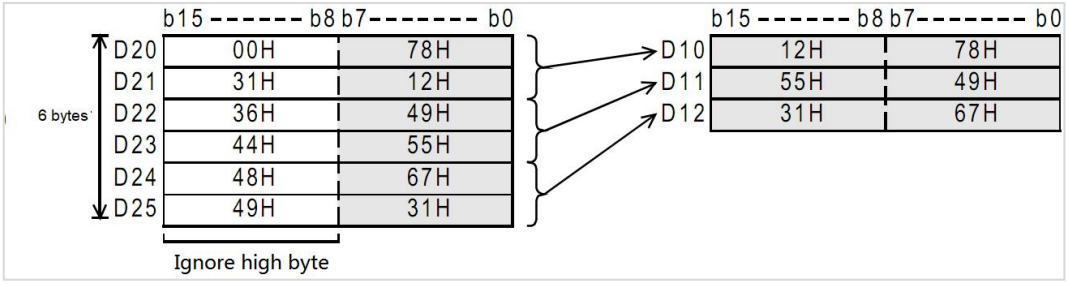
Error code

Error code	Content
4084H	The value specified in (n) exceed range of 0 to 32767
4085H	When the device specified in the write application instruction (s),(d) and (n) exceeds the corresponding device range

Example



When M0 is ON, the data of D20 to D25 is separated according to byte units, and then stored in D10 to D12.



6.4.29 WTOB/Byte unit data separation

WTOB(P)

After separating the BIN 16-bit data stored after the device number specified in (s) into (n) bytes, store it after the device number specified in (d).

-[WTOB (s) (d) (n)]

Content, range and data type

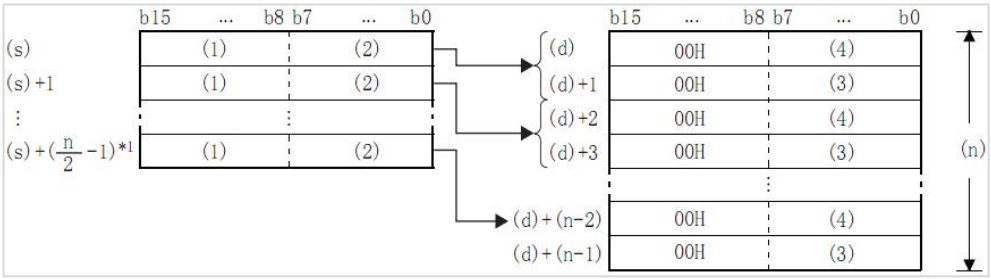
Parameter	Content	Range	Data type	Data type (label)
(s)	The start device that stores the data separation in byte unit	-	Signed BIN 16 bit	ANY16
(d)	The start device that stores the result of separation in byte unit	-	Signed BIN 16 bit	ANY16
(n)	Number of byte data separated	0-32767	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
WTOB	Parameter 1															●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3																	●	●	●			●	●		●	●

Features

After separating the BIN 16-bit data stored after the device number specified in (s) into (n) bytes, store it after the device number specified in (d).



(1)High byte;

(2)Low byte;

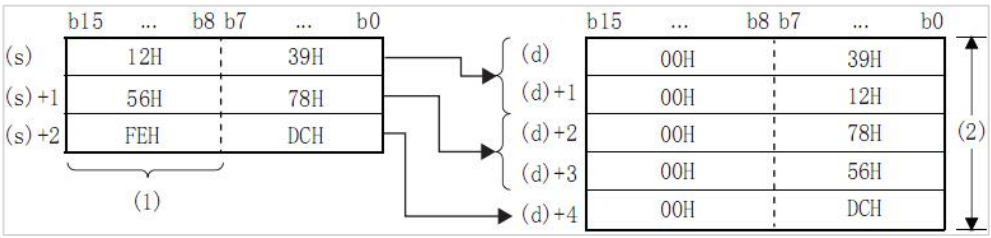
(3)High byte data;

(4)Low byte data;

(5)*1: Carry below the decimal point.

Example

In the case of (n)=5, store the data up to the lower 8 bits of (s) to (s)+2 in (d) to (d)+4:



（1）(N)=5 is ignored.

（2）(N)=5.

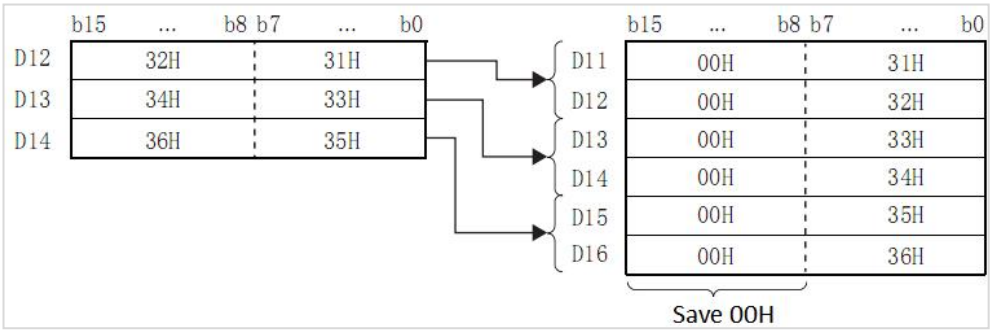
By setting the number of bytes in (n), the range of BIN 16-bit data specified in (s) and the range of the device storing the byte data specified in (d) will be automatically determined.

When the number of bytes specified in (n) is 0, no processing is performed.

00H is automatically stored in the upper 8 bits of the byte data storage device specified in (d).

Example

When D12 to D14 is stored in the low 8 bits of D11 to D16



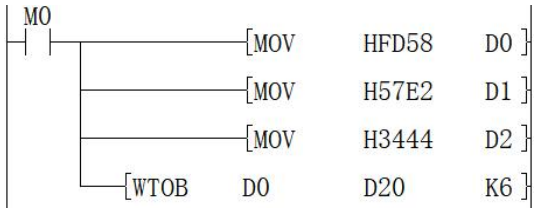
Even if the device range storing the data before merging overlaps the device rangestoring merged data, it will be handled as normal.

Device range storing the data before merging	Device range storing separated data
(s) to (s) + (n/2-1)	(d)+0 to (d)+(n)-1

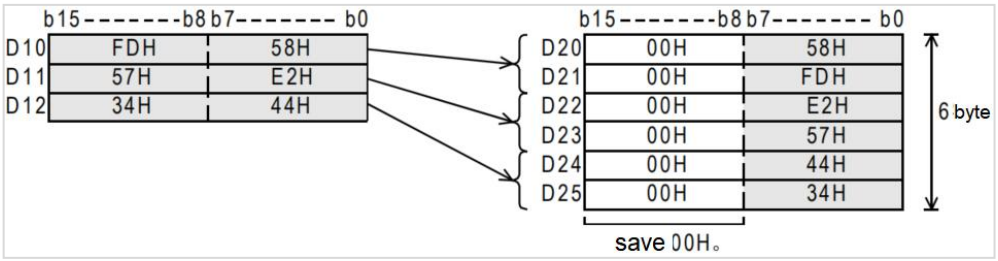
Error code

Error code	Content
4084H	The value specified by (n) exceed the range of 0 to 32767
4085H	When the device specified in read application instruction (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M0 is ON, the data of D10 to D12 are separated according to byte units, and then stored in D20 to D25.



6.4.30 DIS/4-bit separation of 16-bit data

DIS(P)

Store the data of the low (n) bits (1 bit of 4 bits) of the BIN 16-bit data specified in (s) into the low 4-bit of the (n) point starting from the device specified in (d).

-[DIS (s) (d) (n)]

Content, range and data type

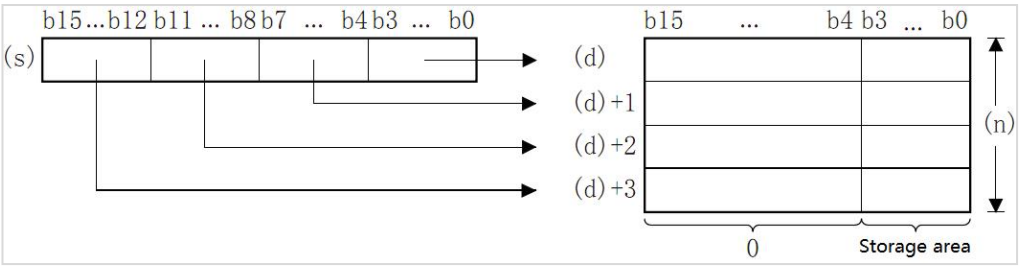
Parameter	Content	Range	Data type	Data type (label)
(s)	The start device storing the data before separation	-	Signed BIN 16 bit	ANY16
(d)	The start device storing separated data	-	Signed BIN 16 bit	ANY16
(n)	Separation number (0 means no processing)	0-4	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DIS	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Store the low-(n) bit (1 bits of 4 bits) of the BIN 16-bit data specified in (s) in the low 4-bit of the (n) point starting from the device specified in (d).



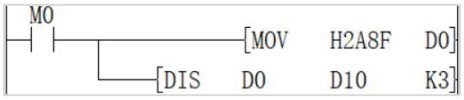
The hig-12 bit of the point (n) starting from the device specified in (s) will become 0.

When (n)=0, it will become no processing, and the content of point (n) starting from the device of (d) will not change.

Error code

Error code	Content
4084H	The data in (n) exceed the range of 0 to 4
4085H	When the device specified in read application instruction (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M0 is ON, D0 is separated every 4 bits and stored in D10 to D12. The result is D10 = HF, D11 = H8, D12 = HA.

6.4.31 UNI/4-bit combination of 16-bit data

UNI(P)

Combine the low 4 bits of the BIN 16-bit data of point (n) starting from the device specified in (s) into the BIN 16-bit device specified in (d).

-[UNI (s) (d) (n)]

Content, range and data type

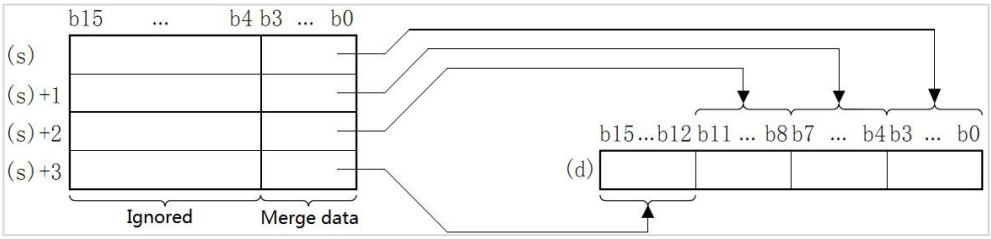
Parameter	Content	Range	Data type	Data type (label)
(s)	The start device storing the data before merging	-	Signed BIN 16 bit	ANY16
(d)	The start device storing the merged data	-	Signed BIN 16 bit	ANY16
(n)	Number of merger	0-4	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
UNI	Parameter 1															●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Combine the low 4 bits of the BIN 16-bit data at point (n) starting from the device specified in (s) into the BIN 16-bit device specified in (d).



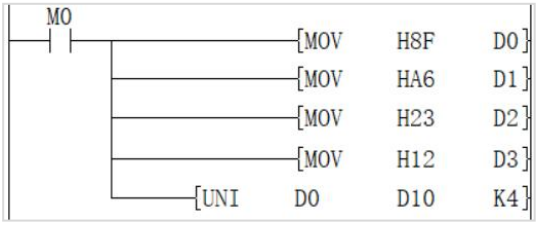
The high (4-n) bits of the device specified in (d) will become 0.

When (n)=0, it will become no processing, and the content of the device in (d) will not change.

Error code

Code	Content
4084H	The data in (n) exceed the range of 0 to 4
4085H	When the device specified in read application instruction (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M0 is ON, the low 4 bits of D0 to D3 are combined and stored in D10, the value is H236F.

6.4.32 ZRST/Data batch reset

ZRST(P)

Perform a batch reset between the devices specified in (d1) and (d2) of the same type. It is used when interrupting operation, performing initial operation, or resetting control data.

-[ZRST (d1) (d2)]

Content, range and data type

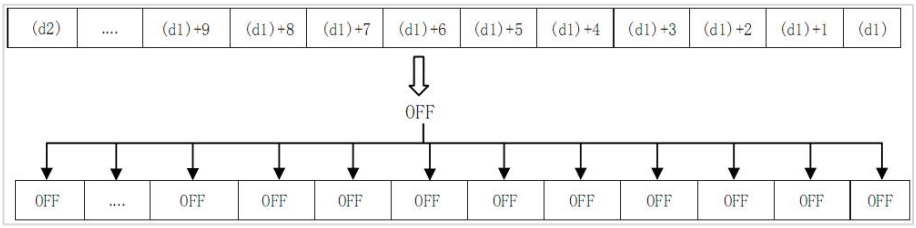
Parameter	Content	Range	Data type	Data type (label)
(d1)	The start bit or word device number of batch reset	-	Bit/Signed BIN 16 bit	ANY_ELEMENTARY
(d2)	The final bit or word device number of batch reset	-	Bit/Signed BIN 16 bit	ANY_ELEMENTARY

Device used

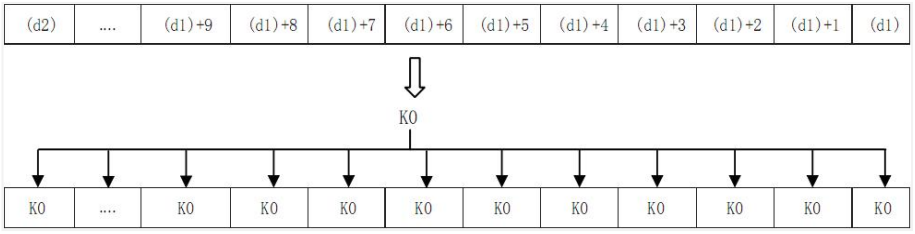
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ZRST	Parameter 1		●	●	●	●						●	●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 2		●	●	●	●						●	●	●	●	●	●	●	●	●	●	●				●	●

Features

Perform batch reset between the devices specified in (d1) and (d2) of the same type.

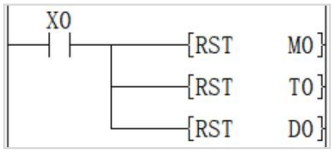


When (d1) and (d2) are bit devices, write OFF (reset) in the entire device range of (d1) to (d2).



When (d1) and (d2) are word devices, write K0 in the entire device range of (d1) to (d2).

As a separate reset instruction for the device, the RST instruction can be used for bit devices or word devices.



Reset M0

Reset the current value of T0

Reset D0

The batch write instruction of constant (for example: K0) has FMOV (P) instruction, which can write 0 to word devices (including bit device specification).



Write K0 in D0 to D99.

✎Note:

Please specify the same type number for (d1) and (d2), and make (d1) number <(d2) number. When (d1) number ≥ (d2) number, only 1 point will be reset for the device specified in (d1).

ZRST(P) instruction is a 16-bit instruction, which can specify (LC) and (HSC) devices for (d1) and (d2).

Error code

Error code	Content
4084H	When the device type specified in (d1) is different from the device type specified in (d2).
4085H	When the device specified in the read application instruction (d1) and (d2) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d1) exceeds the corresponding device range

Example



The function of this ladder diagram instruction is to set the value of the D0 to D100 device to 0.

6.4.33 ZSET/Data batch set

ZSET(P)

Perform a batch set between the devices specified in (d1) and (d2) of the same type.



Content, range and data type

Parameter	Content	Range	Data type	Data type(label)
(d1)	The start bit device number of batch set	-	Bit	ANY_BOOL
(d2)	The final bit device number of batch set	-	Bit	ANY_BOOL

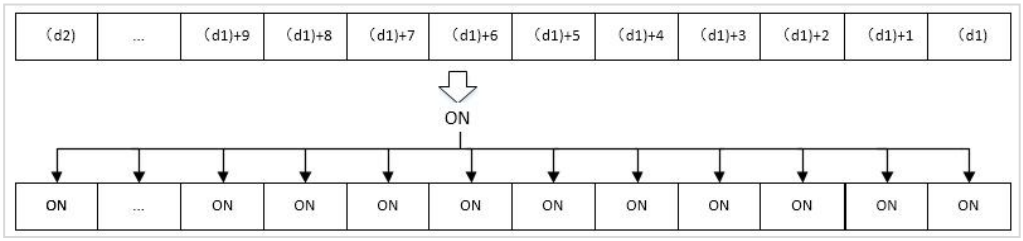
Device used

instruction	Parameter	Devices																							Index modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
ZSET	Parameter 1		●	●	●	●					●															●
	Parameter 2		●	●	●	●					●															●

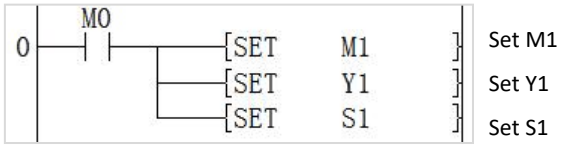
Features

·Perform a batch set between the devices specified in (d1) and (d2) of the same type.

·Write ON (set) in the entire device range of (d1) to (d2)



·As a separate set instruction for the device, the SET instruction can be used for bit devices.



✎Note:

Please specify the same type number for (d1) and (d2), and make (d1) number <(d2) number. When (d1) number ≥ (d2) number, only 1 point will be set for the device specified in (d1).

Error code

Error code	Content
4084H	When the device type specified in (d1) is different from the device type specified in (d2).
4085H	When the device specified in the read application instruction (d1) and (d2) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d1) exceeds the corresponding device range
4087H	When the device type specified in (d1) and (d2) are not bit device.

Example



The function of this LAD instruction is to set the value of the M1 to M4 device to ON.

6.4.34 CRC/Cyclic redundancy check instruction

CRC(P)

Calculate the CRC (Cyclic Redundancy Check) value, which is one of the error checking methods used in communications. In addition to CRC, error checking methods include parity and

Sum check (checksum), calculate horizontal parity check value and sum check value can use CCD(P) instruction . And this instruction is used in the generator polynomial that generates the CRC value (CRC-16)

"X 16 +X 15 +X 2 +1".

-[CRC(P) (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number storing the data of CRC value generated objects	-	Signed BIN16	ANY16
(d)	The destination device number of the generated CRC value	-	Signed BIN16	ANY16
(n)	The number of 8-bit data (bytes) for calculating the CRC value or the number of the device storing the number of data	1 to 256	Unsigned BIN16	ANY16_U

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CRC	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●			●

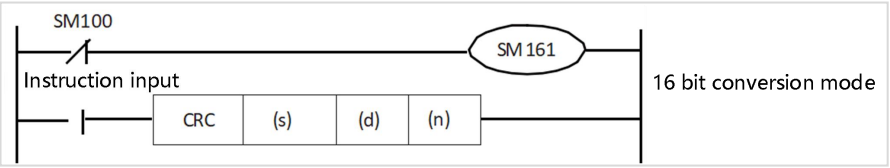
Features

Start with the device specified in (s), generate the CRC value of 8-bit data (byte unit) at (n) point, and store it in (d).

The mode used by this instruction in calculation includes 16-bit conversion mode and 8-bit conversion mode. For the operation of each mode, please refer to the following content.

(1) 16-bit conversion mode (when SM161=OFF)

Calculate the upper 8 bits (byte) and lower 8 bits (byte) of the (s) device. The result is stored in 16 bits of 1 point of the device specified in (d). In the case of the following program, perform the conversion as shown below.



			Example (s)=D100, (d)=D0, (n)=6		
			Devices	Content of object data	
				8-bit	16-bit
CRC value generation target data storage destination	(s)	Low byte	D100 low	01H	0301H
		High byte	D100 high	03H	
	(s)+1	Low byte	D101 low	03H	0203H
		High byte	D101 high	02H	
	(s)+2	Low byte	D102 low	00H	1400H
		High byte	D102 high	14H	
	...				
	(s)+(n)/2-1	Low byte			
		High byte			

CRC value storage target	(d)	Low byte	D0 low	E4H	41E4H
		High byte	D0 high	41H	

(2) 8-bit conversion mode (when SM8161=ON)

In 8-bit conversion mode, only the lower 8 bits (lower byte) of the (s) device are operated on. As a result, 2 points are used starting from the device specified in (d), the lower 8 bits (bytes) are stored in (d), and the upper 8 bits (bytes) are stored in (d)+1.

In the case of the following program, perform the conversion as shown below.



			Example) (s)=D100, (d)=D0, (n)=6	
			Devices	Content of object data
CRC value generation target data storage destination	(s)	Low byte	D100 low	01H
	(s)+1	Low byte	D101 low	03H
	(s)+2	Low byte	D102 low	03H
	(s)+3	Low byte	D103 low	02H
	(s)+4	Low byte	D104 low	00H
	(s)+5	Low byte	D105 low	14H
	...			
	(s)+(n)-1	Low byte		
CRC value storage target	(d)	Low byte	D0	E4H
	(d)+1	Low byte	D1	41H

In the CRC(P) instruction, the generator polynomial of the CRC value (CRC-16) uses "X16+X15+X2+1", but there are also many standardized generator polynomials for the CRC value. If the generator polynomial is different, it will become a completely different CRC value, which should be noted. The main CRC value generator polynomials are shown below.

Name	Generator polynomial
CRC-12	$X^{12}+X^{11}+X^3+X^2+X+1$
CRC-16	$X^{16}+X^{15}+X^2+1$
CRC-32	$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1$
CRC-CCITT	$X^{16}+X^{12}+X^5+1$

✎**Note:**

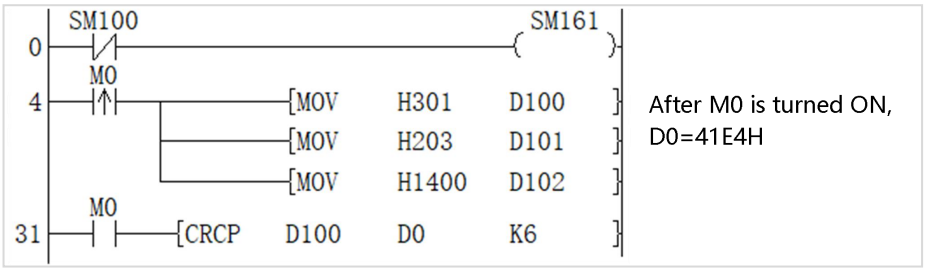
When (s1) use KnX, KnY, KnM, KnS, n must be specified as 4.

Error code

Error code	Content
4084H	The range of (n) exceeds 1 to 256
4085H	The data address of (s) to be converted exceeds the device range
4086H	The (d) write address exceeds the device range
4087H	Unsupported device type is used by (s) and (d)

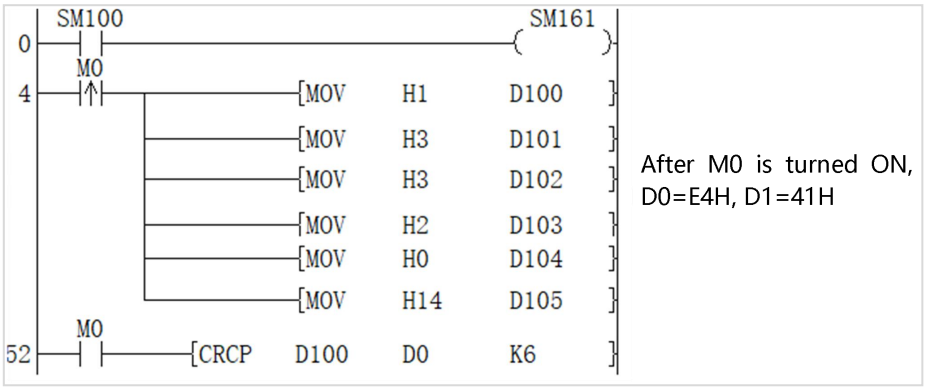
Example

① 16-bit conversion mode



After M0 is turned ON, D0=41E4H

② 8-bit conversion mode



After M0 is turned ON, D0=E4H, D1=41H

6.5 Matrix input instructions

6.5.1 MTR/Matrix input

MTR

The instruction to read the input signal (switch) of 8 points multiply by n columns in the time division method of 8 input and (n) output (transistor).

-[MTR (s) (d1) (d2) (n)]

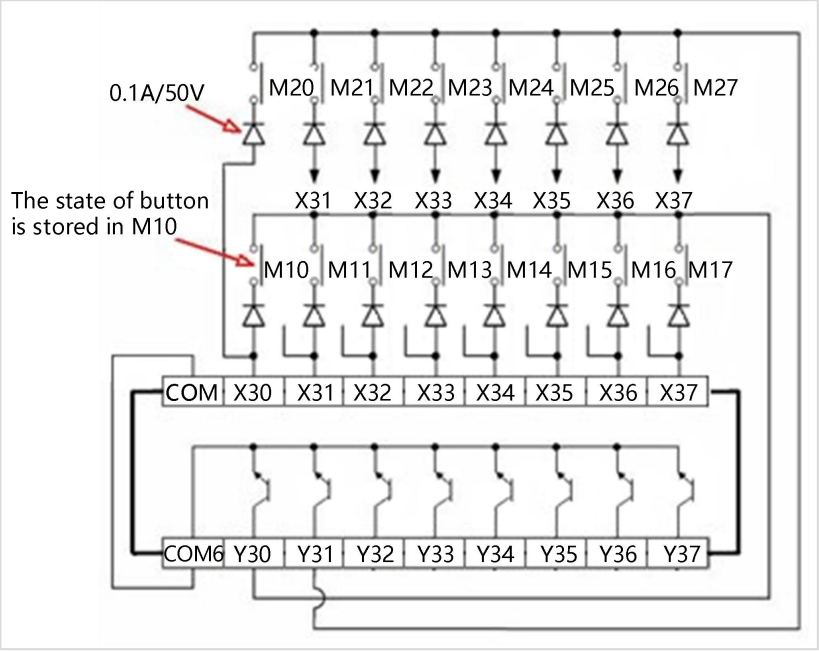
Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device (X) number X000, X010, X020 of the row signal input of the matrix is up to the final input X number. 8 consecutively occupied.	The lowest bit number of X can only be 0	Bit	ANY_BOOL
(d1)	The starting device (Y) number of the column signal output of the matrix is Y000, Y010, Y020... to the final output Y number. 8 consecutively occupied.	The lowest bit number of Y can only be 0	Bit	ANY_BOOL
(d2)	The start device (Y, M, S) number of the ON output destination address is Y000, Y010, Y020..., M000, M010, M020..., S000, S010, S020... until the final Y, M, S number. Y occupies 8*(n) continuously, and the others occupy 10*(n) continuously.	-	Bit	ANY_BOOL
(n)	Set the number of columns in the matrix input.	2 to 8	Unsigned BIN 16 bit	ANY16_U

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MTR	Parameter 1	●																									
	Parameter 2		●																								
	Parameter 3		●	●	●	●																					
	Parameter 4																						●	●			

Features



This instruction generally uses the normally ON contact SM100.



According to the example in the figure:

M10 will turn ON when Y30 and X30 are connected, M14 will be ON when Y30 and X34 are connected, M26 will be ON when Y31 and X36 are connected

(D2) is recommended to use a minimum of 0, mainly when using an address such as M4, the first start is M4, and then it will continue to occupy M11, which is inconvenient to calculate and view, so it is recommended to use a software with a minimum of 0 element.

Special device used

Devices	Content
SM229	SM229 will turn ON after one cycle of execution is completed

✎Note:

The MTR instruction can only run one instruction at the same time.

Error code

Error code	Content
4085H	The read address of (s) and (n) exceeds the device range
	(s) use the numbered device whose low bit is not 0
4086HA	The write address of (d1) and (d2) exceeds the device range
	(d2) use the numbered device whose low bit is not 0
4084H	(n) is not in the range of 2 to 8
4089H	Multiple MTR instructions are executed at the same time

6.6 Convenient instructions

6.6.1 ABSD/BIN 16-bit data absolute method

ABSD

Create multiple output modes corresponding to the current counter (BIN 16-bit value).

-[ABSD (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The start device number storing the data table (rising edge point and falling edge point)	-	Signed BIN 16 bit	ANY16
(s2)	The counter number used for monitoring of the current value compared to the data table	-	Signed BIN 16 bit	ANY16
(d)	The number of points of the output start device	-	Bit	ANY16_BOOL
(n)	Number of table rows and output bit device points	1 to 64	Signed BIN 16 bit	ANY16

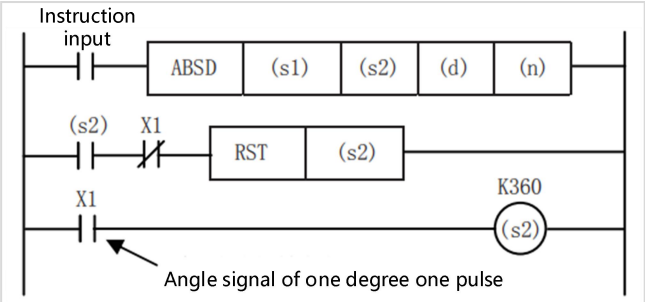
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ABSD	Parameter 1											●	●	●	●	●	●	●	●	●						●	
	Parameter 2																●									●	
	Parameter 3		●	●	●	●					●															●	
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	

Features

Take the turntable to rotate 1 revolution (0 to 360 degrees) to control the output ON/OFF as an example. (1 degree, 1 pulse angle signal)

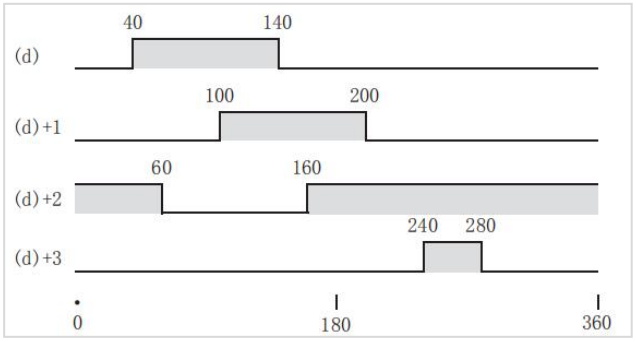
Compare the data table of row (n) starting from (s1) (row (n) multiply by 2 points) with the current value of the counter (s2), from (d) to continuous (n) in the course of one revolution The output is ON/OFF control up to the point.



Use the transfer instruction to write the following data into (s1) to (s1)+2(n)-1 in advance. For example, the rising edge point data stores 16-bit data to even-numbered devices in advance, and the falling edge point data stores 16-bit data to odd-numbered devices in advance.

Rising edge point		Falling edge point		Object output
-	Data value (example)	-	Data value (example)	
(S1)	40	(S1)+1	140	(D)
(S1)+2	100	(S1) +3	200	(D) +1
(S1) +4	160	(S1) +5	60	(D) +2
(S1) +6	240	(S1) +7	280	(D) +3
...	-	...	-	...
(S1)+2(n)-2		(S1)+2(n)-1		(D) +n-1

If the instruction input is set to ON, (d) is the start, (n) point is the output mode as shown below. Each rising edge point and falling edge point can be individually changed by rewriting the data from (s1) to (s1)+2(n)-1.



✎Note:

When specifying the number of bit devices in (s1), the device number should be a multiple of 16 (0, 16, 32, 64...), and only K4 should be specified for the number of bits.

The number of target output points is determined by the value of (n). (1≤(n)≤64)

Even if the instruction input is turned off, the output does not change.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the range of 1 to 64
4085H	When the device specified in the read application instruction (s1), (s2)and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

Refer to the example in the function description.

6.6.2 DABSD/BIN 32-bit data absolute method

DABSD

Create multiple output modes corresponding to the current counter (BIN 32-bit value).

-[DABSD (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The start device number storing the data table (rising edge point and falling edge point)	-	Signed BIN 32 bit	ANY32
(s2)	The counter number used for monitoring of the current value compared to the data table	-	Signed BIN 32 bit	ANY32
(d)	The number of points of the output start device	-	Bit	ANY16_BOOL
(n)	Number of table rows and output bit device points	1 to 64	Signed BIN 32 bit	ANY32

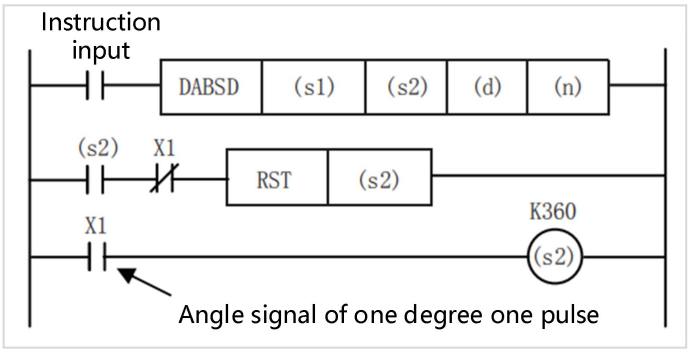
Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DABSD	Parameter 1											●	●	●	●	●	●	●	●	●	●	●				●	
	Parameter 2																●				●	●				●	
	Parameter 3		●	●		●					●															●	
	Parameter 4											●	●	●	●	●	●	●	●	●	●	●	●	●		●	

Features

Take the turntable to rotate 1 revolution (0 to 360 degrees) to control the output ON/OFF as an example. (1 degree, 1 pulse angle signal)

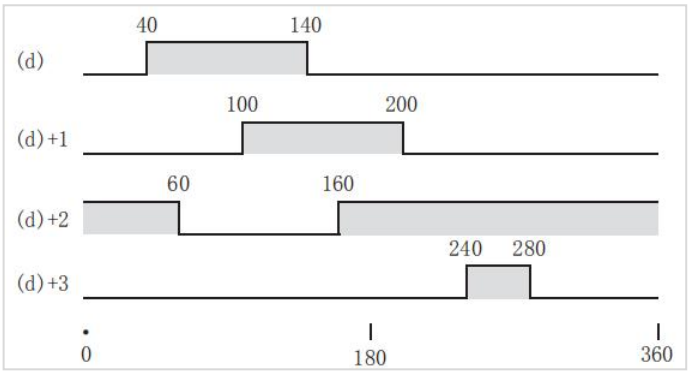
Compare the data table of row (n) starting from (s1) (row (n) × 4 points) with the current value of the counter (s2), from (d) to continuous (n) in the course of one revolution The output is ON/OFF control up to the point.



Use the transfer instruction to write the following data into (s1), (s1)+1 to (s1)+4(n)-2, (s1)+4(n)-1 in advance. For example, the rising edge point data stores 32-bit data to even-numbered devices in advance, and the falling edge point data stores 32-bit data to odd-numbered devices in advance.

Rising edge point		Falling edge point		Object output
-	Data value (example)	-	Data value (example)	
(S1)+1, (S1)	40	(S1)+3, (S1)+2	140	(D)
(S1)+5, (S1)+4	100	(S1) +7, (S1) +6	200	(D) +1
(S1) +9, (S1) +8	160	(S1)+11, (S1)+10	60	(D) +2
(S1) +13, (S1) +12	240	(S1) +15, (S1) +14	280	(D) +3
...	-	...	-	...
(S1)+4(n)-3, (S1)+4(n)-4		(S1)+4(n)-1, (S1)+4(n)-2		(D) +n-1

If the instruction input is set to ON, (d) is the start, (n) point is the output mode as shown below. Each rising edge point and falling edge point can be individually changed by rewriting the data from (s1) to (s1)+2(n)-1.



✎Note:

The high-speed counter can be specified in the DABSD instruction. When a high-speed counter is specified, the current value of the counter will have a response delay due to the scan cycle in the output mode.

When specifying the number of bit devices in (s1), the device number should be a multiple of 16 (0, 16, 32, 64...), and only K8 should be specified for the number of bits.

The number of target output points is determined by the value of (n). (1≤(n)≤64)

Even if the instruction input is turned off, the output does not change.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the range of 1 to 64
4085H	When the device specified in the read application instruction (s1), (s2)and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

Refer to the example in the function description.

6.6.3 SER/16-bit data search

SER(P)

Search the same data and the maximum and minimum values from the data table.

-[SER (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Search for the start device number of the same data, maximum value, and minimum value	-	Signed BIN 16 bit	ANY16
(s2)	Search for the value of the same data or its storage destination device number	-	Signed BIN 16 bit	ANY16
(d)	Search for the same data, maximum value, minimum value and store the start device number	-	Signed BIN 16 bit	ANY16
(n)	Search the number of same data, maximum and minimum	1 to 256	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SER	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

For (s1) as the first (n) data, search for the same data as the BIN 16-bit data of (s2), and store the result in (d) to (d)+4.

In the case of the same data, the number of the same data, the first/final position, and the maximum and minimum positions of the same data are stored in the device with the first 5 points (d).

If there is no identical data, the number of identical data, the first/final position, and the maximum and minimum positions of the same data are stored in the device with the first 5 points (d). However, in (d) is the first 3 points of the device (the number of the same data, the first \\ final position), 0 is stored.

• The structure and data examples of the search result table are as follows. (N=10)

The searched device (s1)	The value of the searched data (s1)	Comparison data (S2) value	Data location	Search results		
				Maximum value (d) +4	Consistent (d)	Minimum value (d+3)
(s1)	K100	K100	0		○(First time)	
(s1)+1	K111		1			
(s1)+2	K100		2		○	
(s1) +3	K98		3			
(s1) +4	K123		4			
(s1) +5	K66		5			○
(s1) +6	K100		6		○ (final)	
(s1) +7	K95		7			
(s1) +8	210		8	○		
(s1) +9	K88		9			

• The search result table based on the above example is shown below.

Device number	Content	Search result items
(d)	3	Number of identical data
(d) +1	0	The position of the same data (first time)
(d) +2	6	The position of the same data (last time)
(d) +3	5	The final position of the minimum
(d) +4	8	The final position of maximum

✎Note:

Perform algebraic size comparison. (-10<2)

When there are multiple minimum and maximum values in the data, the positions behind each are stored.

If driven by this instruction, the search result (d) occupies 5 points of (d), (d)+1, (d)+2, (d)+3, (d)+4. Be careful not to overlap with the device used for machine control.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the range of 0 to 256
4085H	When the device specified in read application instruction (s1), (s2), (d) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

Refer to the example in the function description.

6.6.4 DSER/32-bit data search

DSER(P)

Search the same data and the maximum and minimum values from the data table.

-[DSER (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Search for the start device number of the same data, maximum value, and minimum value	-	Signed BIN 32 bit	ANY32
(s2)	Search for the value of the same data or its storage destination device number	-	Signed BIN 32 bit	ANY32
(d)	Search for the same data, maximum value, minimum value and store the start device number	-	Signed BIN 32 bit	ANY32
(n)	Search the number of same data, maximum and minimum	1 to 128	Signed BIN 32 bit	ANY32

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSER	Parameter 1											●	●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 4											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●

Features

For (s1)+1, (s1) as the initial (n) data, search for the same data as the BIN 32-bit data of (s2)+1, (s2), and store the result in (d)+1, (D) to (d) +9, (d) +8.

In the case of the same data, the number of the same data, the first/final position and the maximum and minimum values are stored in a 5-point BIN 32-bit data device starting with (d)+1 and (d) position.

In the case of no identical data, the number of identical data, the first/final position and the maximum and minimum values are stored in the device with (d)+1 and (d) as the starting BIN 32-bit data with 5 points position. However, 0 is stored in the 32-bit 3-point device (the number of the same data, the first\\last position) with (d)+1 and (d) as the starting BIN.

• The structure and data examples of the search result table are as follows. (N=10)

The searched device (S1)	The value of the searched data (S1)	Comparison data (S2) value	Data location	search results		
				Maximum value (d) +4	Consistent (d)	Minimum value (d+3)
(S1)+1, (S1)	K100	K100	0		○ (First time)	
(S1)+3, (S1)+2	K111		1			
(S1)+5, (S1)+4	K100		2		○	
(S1) +7, (S1) +6	K98		3			
(S1) +9, (S1) +8	K123		4			
(S1)+11, (S1)+10	K66		5			○
(S1) +13, (S1) +12	K100		6		○ (final)	
(S1) +15, (S1) +14	K95		7			
(S1) +17, (S1) +16	210		8	○		
(S1) +19, (S1) +18	K88		9			

• The search result table based on the above example is shown below.

Device number	Content	Search result items
(d)+1, (d)	3	Number of identical data
(d)+3, (d)+2	0	The position of the same data (first time)
(d) +5, (d) +4	6	The position of the same data (last time)
(d) +7, (d) +6	5	The final position of the minimum
(d) +9, (d) +8	8	The final position of maximum

✎Note:

Perform algebraic size comparison. (-10<2)

When there are multiple minimum and maximum values in the data, the positions behind each are stored.

If driven by this instruction, the search result (d) occupies [(d)+1, (d)], [(d)+3, (d)+2,], [(d)+5, (d)+ 4], [(d)+7, (d)+6], [(d)+9, (d)+8] 5 points. Be careful not to overlap with the device used for machine control.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the range of 0 to 128
4085H	When the device specified in read application instruction (s1), (s2), (d) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

Refer to the example in the function description.

6.6.5 ALT/Bit device output inversion

ALT(P)

If the input turns ON, the bit device is inverted (ON→OFF).

-[ALT (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	Alternate output device number	-	Bit	ANY16_BOOL

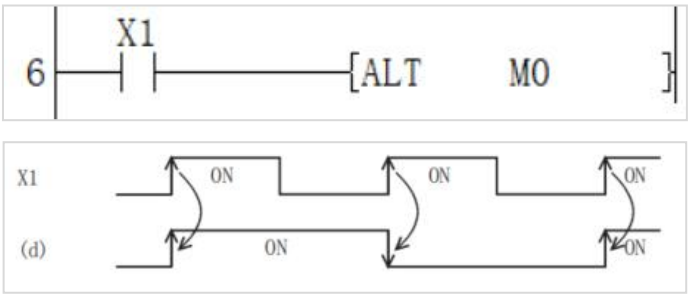
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ALT	Parameter 1		●	●	●	●					●															●	●

Features

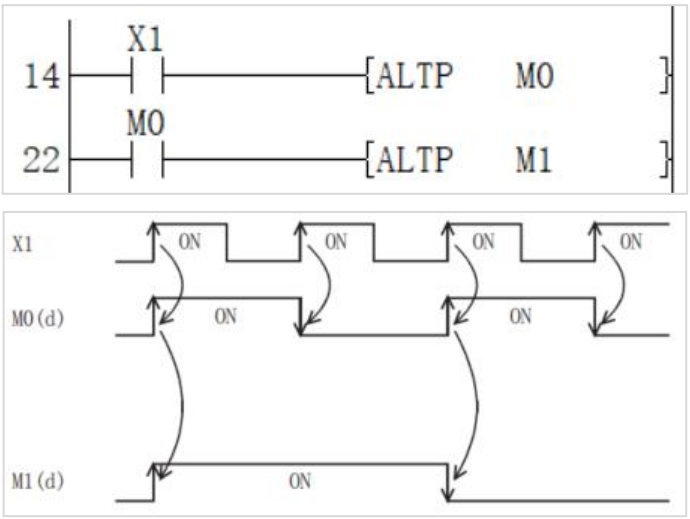
Alternating output (level 1)

Each time the instruction input changes from OFF→ON, the bit device specified in (d) is turned OFF→ON inverted.



Divided frequency output (through alternate output (2 levels))

Combine multiple ALTP instructions to perform frequency division output.



✎Note:

If you program with the ALT instruction, the action will be reversed every operation cycle. To reverse the action by the instruction ON→OFF, use the ALT instruction (pulse execution type) or set the instruction contact to LDP (pulse execution type).

Error code

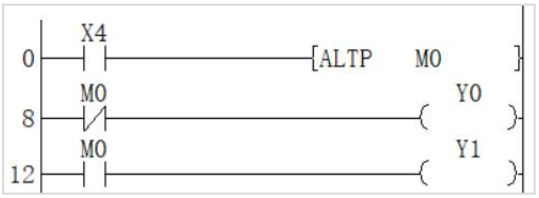
Error code	Content
4085H	When the device specified in the read application instruction (d) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

(1) Start/stop via an input.

1) After pressing the button X4, start the action of output Y1 and stop the action of Y0.

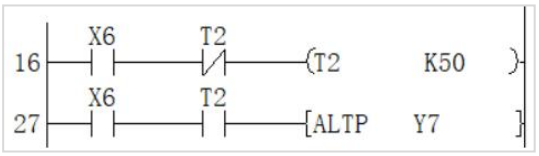
2) After pressing the button X4 again, stop the action of output Y1 and start the action of Y0.

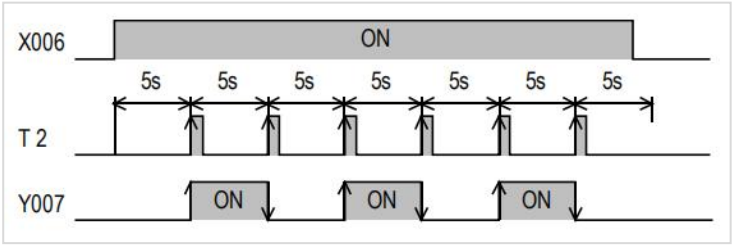


(2) Flashing action

1) When input X6 is ON, the contact of timer T2 will act instantaneously every 5 seconds.

2) The contact of T2 makes the output Y7 alternately ON/OFF every time it is ON.





6.6.6 INCD/BIN 16-bit data relative method

INCD

Use a pair of counters to create multiple output modes.

-[INCD (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The start device number storing the set value	-	Signed BIN 16 bit	ANY16
(s2)	The start number of counter for current value monitoring	-	Signed BIN 16 bit	ANY16
(d)	The start bit device number of output	-	Bit	ANY16_BOOL
(n)	Number of output bit device points	1 to 64	Signed BIN 16 bit	ANY16

Device used

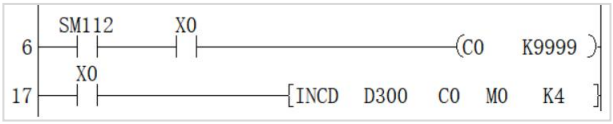
instruction	Parameter	Devices																									Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
INCD	Parameter 1											●	●	●	●	●	●	●	●	●							●	
	Parameter 2																●										●	
	Parameter 3		●	●	●	●					●																●	
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●			●	

Features

Compare the data table of row (n) starting from (s1) (row (n) × 2 points occupied) with the current value of the counter (s2), reset if they match, and control the output on/off in turn.

Example

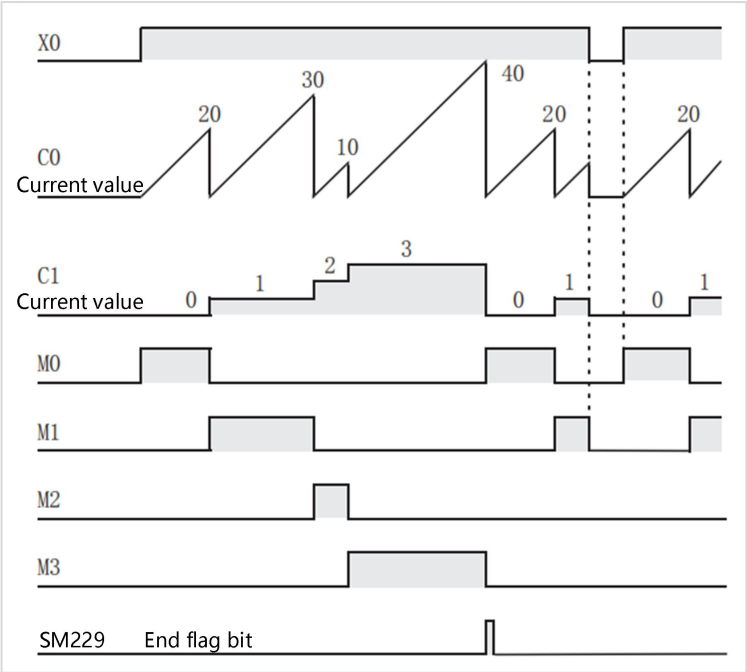
The operation is explained by the following circuit example. (S2) Take up 2 points. C0 and C1 are equivalent to this in the following timing chart.



• It is assumed that the following data is written using the transfer instruction in advance.

Storage device		Output	
-	Data value (example)	-	Example
(S1)	D300=20	(D)	M0
(S1)+1	D301=30	(D) +1	M1
(S1)+2	D302=10	(D) +2	M2
(S1) +3	D303=40	(D) +3	M3
...	....	...	...
(S1)+(n)-1	-	(D) +n-1	-

Timing diagram



If the instruction contact turns on, the M0 output turns on.

The output (M0) is reset when the current value of C0 reaches the comparison value D300, the count value of the process counter C1 is +1, and the current value of the counter C0 is also reset.

The next output M1 turns ON.

Compare the current value of C0 with the comparison value D301. When the comparison value is reached, the output M1 is reset, the count value of the process counter C1 is +1, and the current value of the counter C0 is also reset.

Compare the same to the point (K4) specified in (n). (1≤(n)≤64)

After the final process specified in (n) is completed, the execution end flag SM229 turns ON for 1 operation cycle. SM229 is the instruction execution end flag used in multiple instructions, so it should be used as a contact after the instruction to execute the end flag dedicated to the instruction.

Return to the beginning and repeat output.

✎**Note:**

In (s1), when specifying the device number by specifying the digits of the bit device, the device number should be a multiple of 16 (0, 16, 32, 64...).

Up to 4 INCD instructions can be driven simultaneously in the program.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the range of 1 to 64
4085H	When the device specified in read application instruction (s1), (s2), (d) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (s2) and (d) exceeds the corresponding device range
4089H	The number of instruction drives exceeds the limit.

Example

Refer to the example in the function description.

6.6.7 RAMP/Control ramp signal

RAM(P)

Obtain data that changes between the start (initial value) and end (target value) two values specified (n) times.

-[RAMP (s1) (s2) (d) (n)]

Content, range and data type

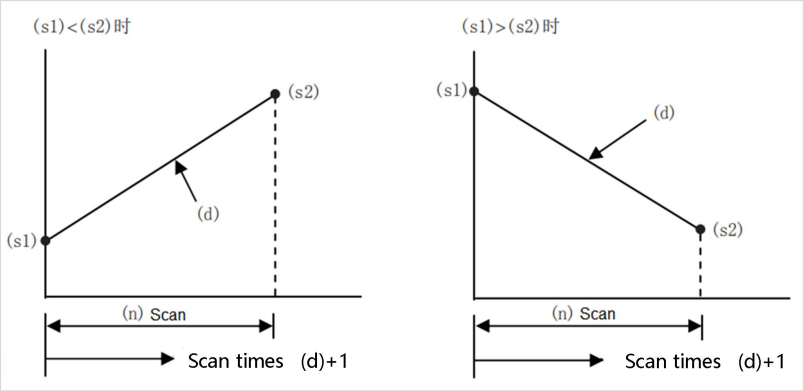
Parameter	Content	Range	Data type	Data type (label)
(s1)	The device number that stores the initial value of the set ramp	-	Signed BIN 16 bit	ANY16
(s2)	The device number that stores the set ramp target value	-	Signed BIN 16 bit	ANY16
(d)	The device number that stores the current value data of ramp	-	Signed BIN 16 bit	ANY16
(n)	Ramp transition time (scan period)	1-32767	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RAMP	Parameter 1											●	●	●	●	●	●	●	●	●						●	
	Parameter 2											●	●	●	●	●	●	●	●	●						●	
	Parameter 3															●	●	●	●	●						●	
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	

Features

Specify the start value (s1) and the value to end (s2) in advance. If the instruction input is turned ON, the value divided by the number of times specified in (n) will be added to (s1) in sequence in each operation cycle The value of is stored in (d). This instruction and analog output can be combined to output soft start/stop instructions.

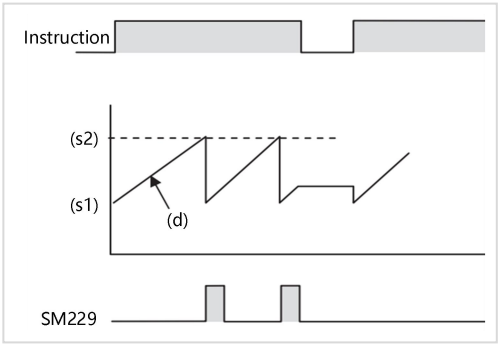


(d)+1 stores the number of scans (0→n times).

The time from the start to the end value requires operation cycle×(n) scan.

If the input instruction is turned OFF during operation, it will be in the execution interrupt state ((d): current value data retention. (d)+1 scan times clear), if it is turned ON again, (d) will be cleared (S1) Restart the action.

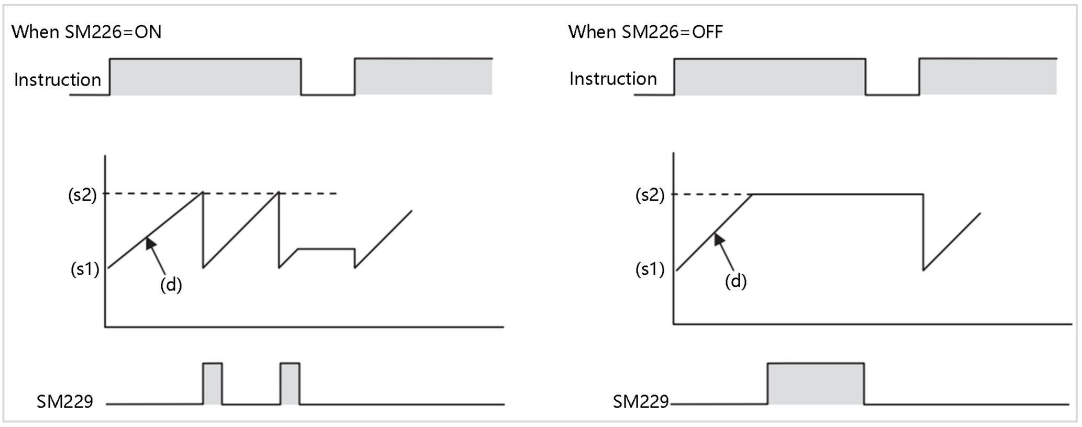
After the transition is completed, the instruction execution completed flag SM229 will act, and the value of (d) will return to the value of (s1).



In the case of obtaining the calculation result at a certain time interval (constant scan mode), write the specified scan time to SD120 (a value slightly longer than the actual scan time), and turn on SM120. For example, when the value is specified as 20 ms and n=100 times, the value of (d) changes from (s1) to (s2) in 2 seconds.

The value of the constant scan mode can also be set by the parameter setting of the engineering tool (the constant scan execution interval setting of the CPU parameter).

According to the ON/OFF action of the mode flag SM226, the content of (d) is changed as shown below.



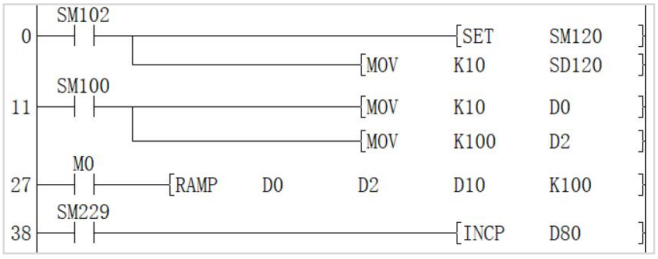
✎Note:

When the power failure retention device (retention area) is specified in (d), the instruction input remains ON. When the CPU module is set to RUN (start), clear (d) in advance.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the specified range of 1 to 32767
4085H	When the device specified in read application instruction (s1), (s2), (d) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



As in the above procedure, turn SM120 ON, and the program will run with a constant scan cycle (the value in SD120 is 10ms). When M0=ON, it changes from 10 to 100 within 100×10ms.

6.6.8 ROTC/Rotary table proximity control

ROTC

In order to take out the items on the rotating table, take out the window according to the requirements, and make the rotating table rotate nearby.

-[ROTC (s) (n1) (n2) (d)]

Content, range and data type

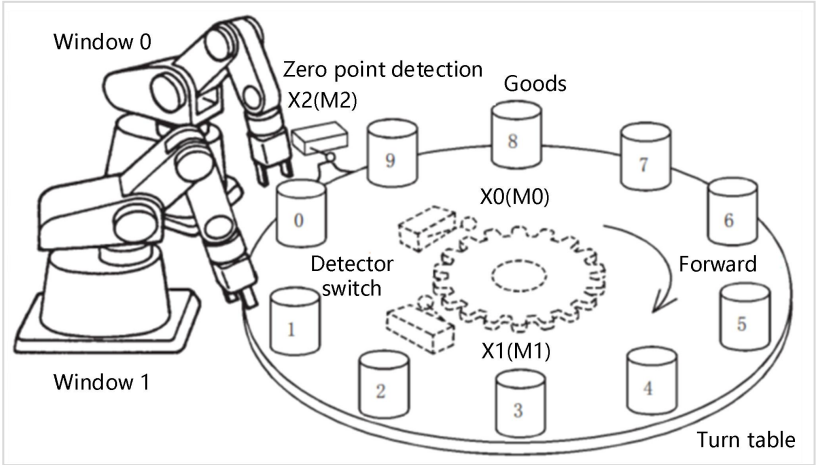
Parameter	Content		Range	Data type	Data type (label)
(s)	The specified register of the calling condition (pre-set according to the transfer instruction)	(s)+0: Register for counting	-	Signed BIN 16 bit	ANY16
		(s)+1: Call the window number setting			
		(s)+2: Call the item number setting			
(n1)	Number of divisions		2 to 32767	Signed BIN 16 bit	ANY16
(n2)	Singular in low speed zone		0 to 32767	Signed BIN 16 bit	ANY16
(d)	The specified bit of the calling condition (constitutes an internal contact circuit driven in advance from the input signal (X))	(d): phase A signal	-	Signed BIN 16 bit	ANY16
		(d)+1: phase B signal			
		(d)+2: zero point detection signal			
		(d)+3: high-speed forward rotation			
		(d)+4: low speed forward rotation			
		(d)+5: stop			
		(d)+6: low speed reverse rotation			
		(d)+7: high-speed reverse rotation			

Device used

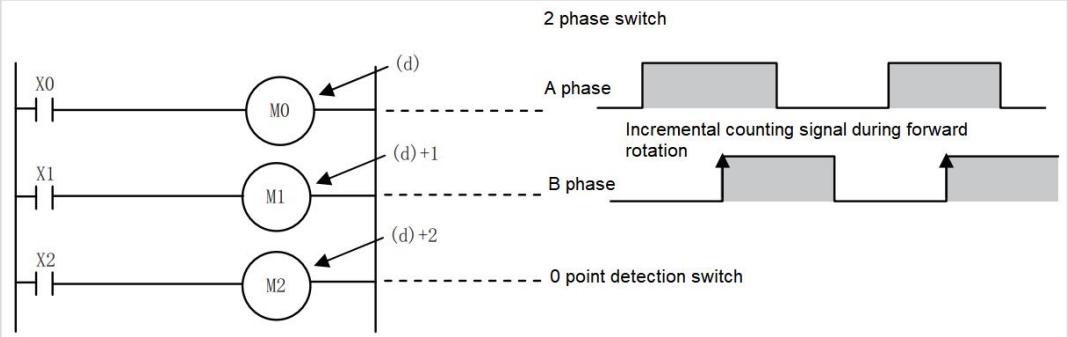
instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ROTC	Parameter 1																	●	●	●						●	
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 4	●	●	●	●						●															●	

Features

In order to take out the items on the rotating table divided into n1 (=10) as shown in the figure below, take out the inserted window as required, and rotate the rotating table nearby under the condition of n2 or (s), (d) . If the following operating conditions are specified, (d)+3 to (d)+7 can be used for forward/reverse, high-speed/low-speed/stop output.



Set up the switch X2 that is used to detect the two-phase shape (X0, X1) of the forward/reverse rotation of the rotary table and window 0. Replace X0 to X2 with (d) to (d) +2 internal contacts. The start device number specified in X or (d) can be arbitrary.



(s) is a counter, which counts how many items come to window 0.

(s)+1 set the number of the window to be called.

(s)+2 sets the number of the recalled item.

Specify the number of divisions (n1) and low-speed operation section (n2) of the rotary table.

✎Note:

If the instruction input is turned ON to drive the instruction, the result of (d)+3 to (d)+7 will be automatically obtained. If the instruction input is turned off, (d)+3 to (d)+7 will turn off.

As an example, when the rotation detection signal ((d) to (d)+2) is set to 10 actions within 1 division interval, the division number setting, calling window number setting, and article number setting should all be 10 Times the value. In this way, the setting value of the low-speed section can be set to the middle value of the number of divisions, etc.

When the instruction input is ON and the 0 point detection signal (M2) is turned ON, the content of the counting register (s) is cleared to 0. It is necessary to perform this clear operation in advance before starting operation.

At most 4 ROTC instructions can be executed at the same time.

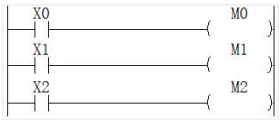
Error code

Error code	Content
4084H	When the value specified in (n1) exceeds the range of 2 to 32767
	When the value specified in (n2) exceeds the range of 0 to 32767
	When the values specified in (n1) and (n2) meet the condition of (n1)<(n2)
	When one of (s), (s)+1 and (s)+2 is negative.
	When one of (s), (s)+1 and (s)+2 is (n1) or more.
4085H	When the device specified in read application instruction (s1), (n1), (n2) and (d) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (s2) and (d) exceeds the corresponding device range
4089H	The number of instructions to be executed exceeds the limit.

Example



Variable	Features	Instructions	Variable	Function	Instructions
D200	Used as a counting register	The 3 units are pre-set by the user program	M3	Run at forward direction at high velocity	When X0 is ON, the result of M3 to M7 could be automatically obtained.
D201	Call window		M4	Run at forward	

<table>
<tr><td></td><td>number setting</td><td rowspan="2"></td><td></td><td>direction at low velocity</td><td rowspan="2"></td></tr>
<tr><td>D202</td><td>Call work piece number setting</td><td>M5</td><td>Stop</td></tr>
<tr><td>M0</td><td>Phase A signal</td><td rowspan="3">The user program executes before each scan of this statement:
 
</td><td>M6</td><td>Run at reverse direction at low velocity</td><td rowspan="3">When X0 is OFF, M3 to M7 are all OFF.</td></tr>
<tr><td>M1</td><td>Phase B signal</td><td>M7</td><td>Run at reverse direction at high velocity</td></tr>
<tr><td>M2</td><td>Zero point detection signal</td><td></td><td></td></tr>
</table>

6.6.9 STMR/Special function timer

STMR

Use the 4 points starting from the device specified in (d) to perform 4 types of timer output.

-[STMR (s1) (s2) (d)]

Content, range and data type

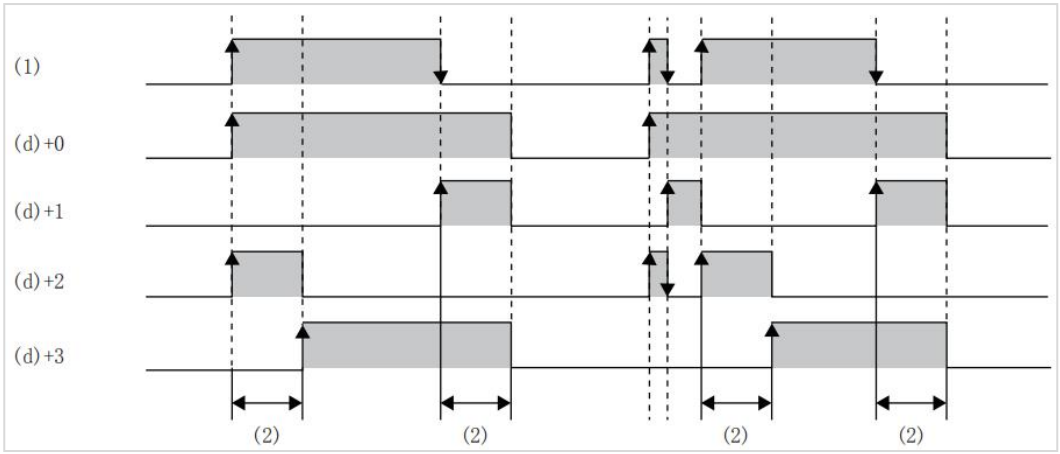
Parameter	Content	Range	Data type	Data type (label)
(s1)	Timer number used: T0 to T511 (100ms timer)	-	Device Name	ANY16
(s2)	Timer setting value	1-32767	Signed BIN 16 bit	ANY16
(d)	The start bit number of the output (occupies 4 points)	-	Bit	ANYBIT_ARRAY

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
STMR	Parameter 1															●										●	
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 3		●	●	●	●					●															●	

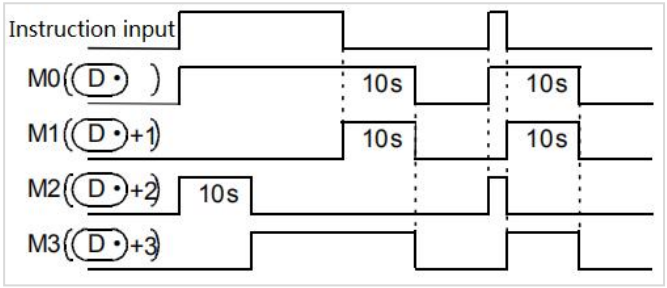
Features

Use the 4 points starting from the device specified in (d) to perform 4 types of timer output.



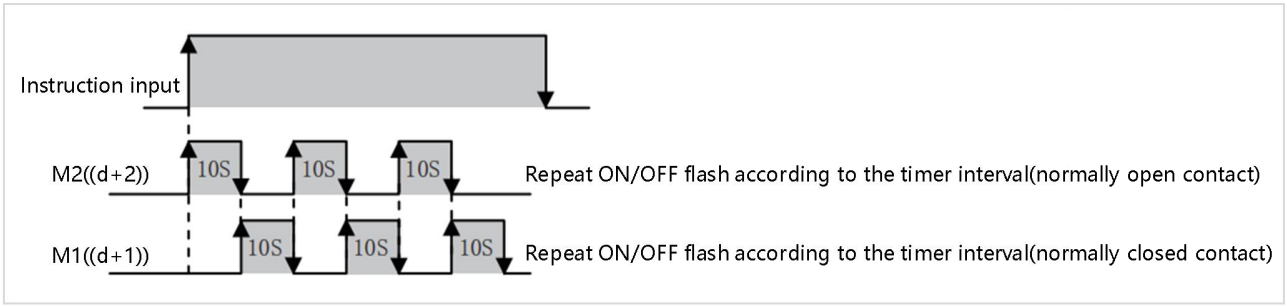
（1） The instruction of STMR instruction

（2） The setting value specified in (S2)



The blink will be in (d)+3 normally closed contact through the following program which turns on/off the STMR instruction (T10 is allocated in (s1), K100 is allocated in (s2), and M0 is allocated in (d)) Output to (d)+1, (d)+2.





The setting value of (S2) can be specified in the range of 1 to 32767 (1 to 3276.7 seconds).

✎Note:

The timer number specified by this instruction cannot be reused with other general circuits (OUT instructions, etc.). In the case of repetition, the timer action cannot be executed correctly.

The timer specified in (s1) is regarded as a 100ms timer, starting from the rising edge of the instruction contact.

Occupy the device specified in 4 points (d) at the beginning. Be careful not to overlap with the device used for machine control.

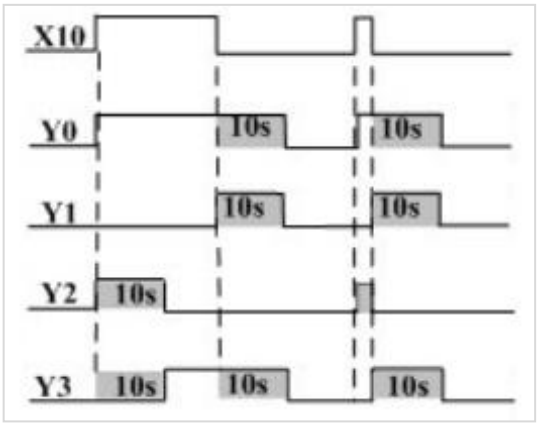
When the instruction contact is turned off, (d), (d)+1, (d)+3 will turn off after the set time. (D) +2 and timer (s1) are reset immediately.

Error code

Error code	Content
4084H	When the value specified in (s2) is less or equal to 0
4085H	When the device specified in the read application instruction (s2) and (d) exceeds the corresponding device range

Example





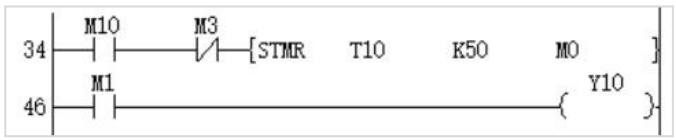
Y0: When X10 changes from Off→On, Y0=On, when X10 changes from On→Off, Y0=Off after a delay of 10 seconds.

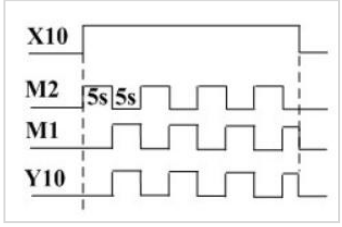
Y1: When X10 changes from On→Off, make Y1=On output once for 10 seconds.

Y2: When X10 changes from Off to On, output Y2=On once for 10

Y3: When X10 changes from Off to On, Y3=On after 10 seconds of delay. When X10 changes from On to Off, Y3=Off after 10 seconds

If the component (d)+3 is introduced into the instruction stream, the oscillator output can be easily realized (this function can also be realized by the ALT instruction), as shown in the following figure:





6.6.10 TTMR/Demonstration timer

TTMR

Test the time when the TTMR instruction is ON. It is used when adjusting the timer setting time with buttons.

-[TTMR (d) (s)]

Content, range and data type

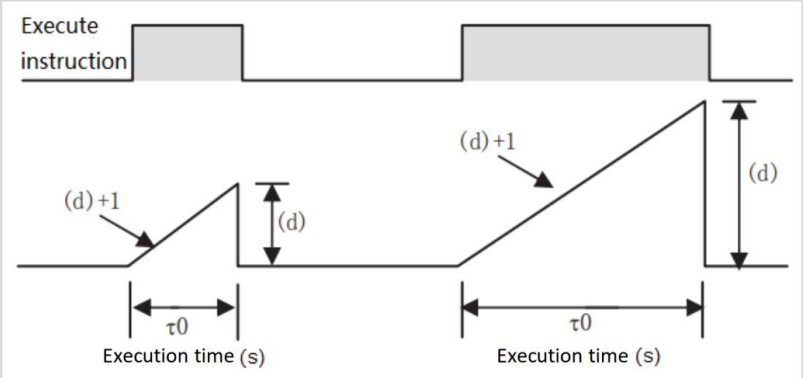
Parameter	Content	Range	Data type	Data type (label)
(d)	Device for storing teaching data	-	Signed BIN 16 bit	ANY16
(s)	Multiplying ratio of teaching data	0-2	Signed BIN 16 bit	ANY16

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TTMR	Parameter 1															●	●	●	●	●						●	
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	

Features

Measure the time of the execution instruction (button) in seconds, multiply it by the magnification (10^S) specified in (s) and store it in the device specified in (d).



For the time stored in (d), when the execution time is $\tau 0$ (unit: second), the actual value of (d) is as follows according to the magnification specified in (s).

(s)	Magnification	(D)
K0	$\tau 0$	(D) ×1
K1	$10\tau 0$	(D)×10
K2	$100\tau 0$	(D) ×100

(s)	(d)	(d)+1 (unit: 100 milliseconds)
K0 (unit: second)	$1\times\tau 0$	(d)+1 =(d)×10
K1 (unit: 100 milliseconds)	$10\times\tau 0$	(d)+1 =(d)
K2 (unit: 10 milliseconds)	$100\times\tau 0$	(d)+1 =(d)/10

✎Note:

If the instruction contact turns from ON→OFF, the current value of the execution time (d)+1 is cleared, and the teaching time (d) does not change.

Occupy the device specified in the 2 teaching time (d) at the beginning. Be careful not to overlap with the device used for machine control.

Error code

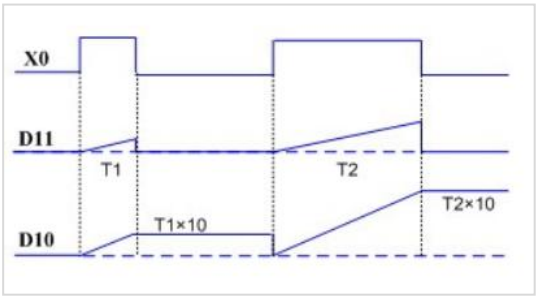
Error code	Content
4084H	When the value specified in (n) exceeds the range of 0 to 2
4085H	When the device specified in read application instruction (d) and (s) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example

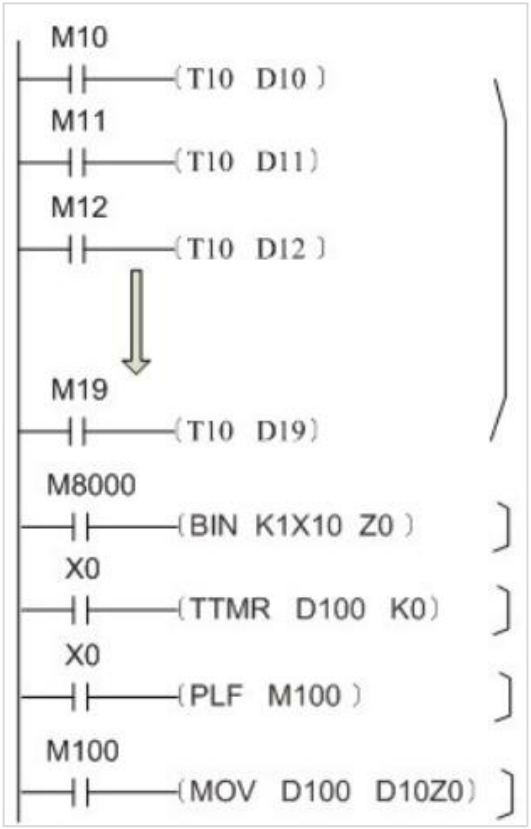
Example 1



When X0 is closed, D10=D11; when X0 is opened, the value of D10 remains unchanged, while D11 becomes 0.



Example 2



Use the TTMR instruction to write 10 sets of setting time, write the setting value into D10 to D19 in advance, reorganize the timer bit 100ms type timer, so 1/10 of the teaching data is the actual operating time (seconds)

Connect the 1-digit DIP switch to X10 to X13, use the BIN instruction to convert the setting value of the DIP switch into BIN value and store it in Z0

X0 is On, store the time (seconds) in D100

M100 demonstrates a scan cycle pulse generated by the release of the timer button X0

Use the setting number of the DIP switch as an indirect specified pointer, and then transfer the content of D100 to D10Z0 (D10 to D19)

6.7 External IO instructions

6.7.1 ARWS/Arrow switch

ARWS

Use the arrow switches for digit movement and increase or decrease of digit values to input data instructions.

-[ARWS (s) (d1) (d2) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device number that input	-	BIN16 bit	ANY_BOOL
(d1)	The word device number storing BCD conversion data	-	BIN16 bit	ANY_BOOL
(d2)	The start bit device (Y) that connect the display of the 7-segment digital tube	0 to 9999	BIN16 bit	ANY16_S
(n)	Specify the number of digits displayed by the 7-segment digital tube (Setting range: K0 to K3)	0 to 3	BIN16 bit	ANY16_S

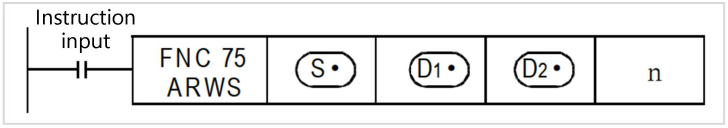
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ARWS	Parameter 1	●	●	●	●	●					●															●	
	Parameter 2															●	●	●	●	●						●	
	Parameter 3		●																							●	
	Parameter 4																						●	●			

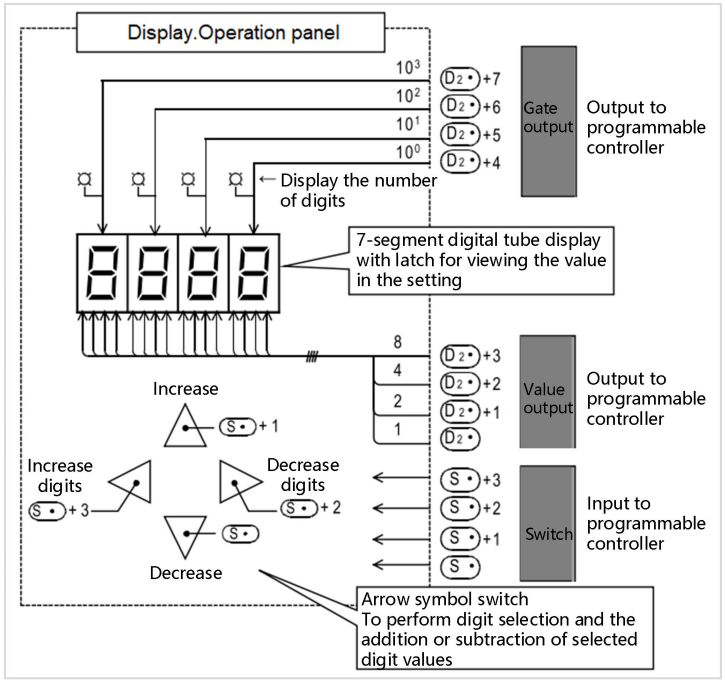
Features

16-bit operation (ARWS). The 16-bit BIN value from 0 to 9999 is stored in D+1. For the sake of convenience, the following description is displayed in BCD conversion.

When the instruction input is ON, the ARWS instruction will operate as shown below



Display and operation part of the content



(1) The digit specification of n displayed by the 7-segment digital tube with BCD code

A 4-digit (10^3 digit) is used as an example in the following operation description, .

(2) The action of the digit selection switch (S+2, S+3)

1) The action when input S+2 with reduced digits is ON.Each time the switch is pressed, the number of digits specification is changed according to $10^3 \rightarrow 10^2 \rightarrow 10^1 \rightarrow 10^0 \rightarrow 10^3$.

2) The action when the input S+3 with increased digits is ON. Each time the switch is pressed, the number of digits specification is changed according to $10^3 \rightarrow 10^0 \rightarrow 10^1 \rightarrow 10^2 \rightarrow 10^3$.

(3) The action of the LED for displaying the selected digits (D2+4 to D2+7). The specified number of digits can be displayed by LED by strobe signal D2+4 to D2+7.

(4) The operation of the data change switch in units of digits (S, S+1). The data is changed for the number of digits specified by the "digit selection switch" above.

1) Increase the action when the input is ON. Each time the switch is pressed, the content of D1 changes according to 0→1→2→...→8→9→0→1.

2) Reduce the action when the input is ON. Each time the switch is pressed, the content of D1 changes according to 0→9→8→7...1→0→9.

These contents can be displayed in the 7-segment digital tube display.

As shown above, through a series of operations, you can write the target value into D1 while viewing the 7-segment display.

✎Note:

① The setting of parameter n

Please refer to the parameter setting of SEGL (FNC 74) instruction. The setting range is 0 to 3.

② The output format of the programmable controller, please use a transistor output type programmable controller.

③ About scan time (operation cycle) and display timing

④ The ARWS instruction is executed synchronously with the scan time (operation cycle) of the programmable controller.

⑤ In order to perform a series of displays, the scan time of the programmable controller needs to exceed 10ms.

⑥ When it is less than 10ms, please use the constant scan mode and run with a scan time longer than 10ms.

⑦ Number of occupied points of the device

- The input of the device s occupies 4 points.
- The output of the device d2 occupies 8 points.

⑧ Restrictions on the times of the uses of instructions

Only one ARWS instruction can be used in the program.

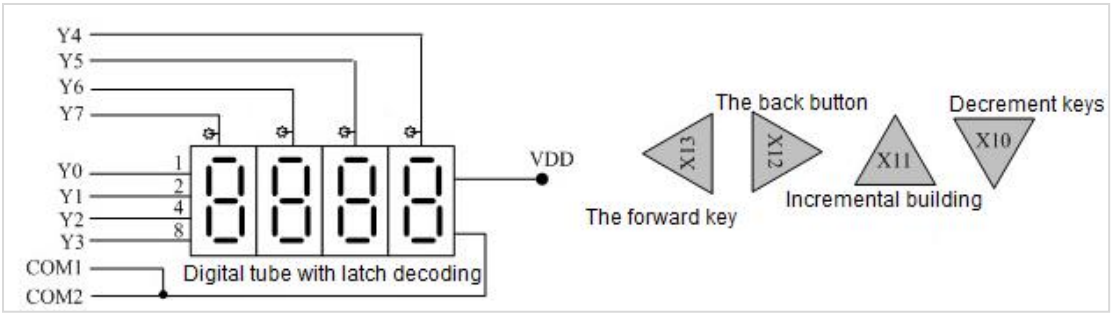
Error code

Error code	Content
4084H	The data input in the application instruction (d1) and (d2) exceeds the specified range
4085H	The output result of the read application instruction (s), (d1) and (d2) exceeds the device range
4086H	The output result of the write application instruction (d1) and (d2) exceeds the device range

Example



The corresponding hardware wiring is shown in the figure below, and the PLC should be transistor output type:



(1) The digital tube in the figure shows the value of D0. Press X10 to X13 to modify the value. The value of D0 can only be between 0 and 9999.

(2) When X20 is ON, the cursor position is thousands. Each time the back key (X12) is pressed, the specified position is switched in the order of "thousands → hundred → ten → pieces → thousand"; if the forward key (X13) is pressed, the switching sequence is reversed; the cursor position is determined by the strobe pulse signal (YO04 to YOO7) LED indication of connection.

(3) For the cursor position, each time you press the increment key (X11), the content of the position changes by 0 → 1→ 2→……8→9→0→1, and when you press the decrement key (X10), press 0 →9→8→7→…… 1→0→9 changes, the modified value takes effect immediately.

6.7.2 DSW/Numeric key input

DSW

This instruction is to read the state of the matrix type setting switch, with 4 BCD setting switches as a group, and store the setting value in the specified unit after reading it. Up to 2 groups of setting switches can be read.

-[DSW (s) (d1) (d2) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device (X) number connected to the digital switch (occupies 4 points)	-	Bit	ANY_BOOL
(d1)	The start device (Y) number that strobe signal outputed (occupies 4 points)	-	Bit	ANY_BOOL
(d2)	The device number that stores the value of the digital switch (occupies n points)	0 to 9999	Signed BIN16	ANY16_S
(n)	Number of groups of digital switches (4 digits a group) (n=1 or 2)	1 to 2	Signed BIN16	ANY16_S

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSW	Parameter 1	●																							●	
	Parameter 2		●																						●	
	Parameter 3															●	●	●	●	●					●	
	Parameter 4																						●	●		

Features

This instruction is to read the state of the matrix type setting switch, with 4 BCD setting switches as a group, and store the setting value in the specified unit after reading it. Up to 2 groups of setting switches can be read.

(1) About the input value (d1)

4 digits from 0 to 9,999 could be read.

Data is saved in BIN (binary number) value.

The first group is saved in (d2), and the second group is saved in (d2)+1.

(2) specification of the number of groups n

① When using 4 digits/1 group×1 [n=K1] pass the strobe signal

From (s) to [(s)+3], sequentially read the BCD 4-digit digital switches connected in (d1) to [(d1)+3], and save the value as BIN value in (d2).

② When using 4 digits/1 group×2 [n=K2] pass the strobe signal

From (s) to [(s)+3], sequentially read the BCD 4-digit digital switches connected in (d1) to [(d1)+3], and save the value as BIN value in (d2).

Through the strobe signal (d1) to [(d1)+3], read the BCD 4-digit digital switch connected in (s)+4 to [(s)+7] in turn, and save its value as a BIN value To (d2)+1.

✎**Note:**

(1) When the instruction contact is OFF

Even if it is OFF, the content of (d2) does not change, but from (d1) to [(d1)+3] all become OFF.

(2) Occupied points of the device

1) When using 4 digits 2 groups (n=K2), 2 points starting from (d2) are occupied.

2) When it is 4 digits and 1 group (s), 4 points are occupied, and when it is 4 digits and 2 groups, 8 points are occupied.

(3) When connecting a digital switch with less than 4 digits

For unused digits, the strobe signal <output for specified digits> (d1) does not need to be wired, but even if there are unused digits, its output is already occupied by this instruction, so it cannot be used for other purposes. Be sure to leave unused output empty.

(4) It is recommended to use transistor output type

In order to read the value of the digital switch continuously, be sure to use a transistor output type programmable controller.

(5) About digital switches

Please use a digital switch of BCD output type.

(6) About the read timing of keyboard input

In order to prevent reading omissions caused by the filter delay of keyboard input, please use the “Constant Scan Mode” and “Timer Interrupt” functions flexibly.

(7) The limit number of instructions

A maximum of two can be used at the same time

Related device

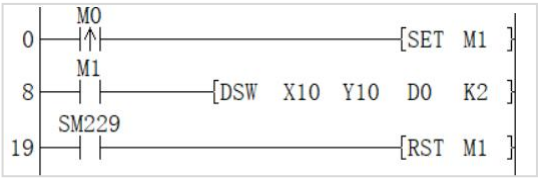
Devices	Name	Content
SM229	End of instruction execution	After a reading cycle is over, SM229 will be set for a scan cycle

Error code

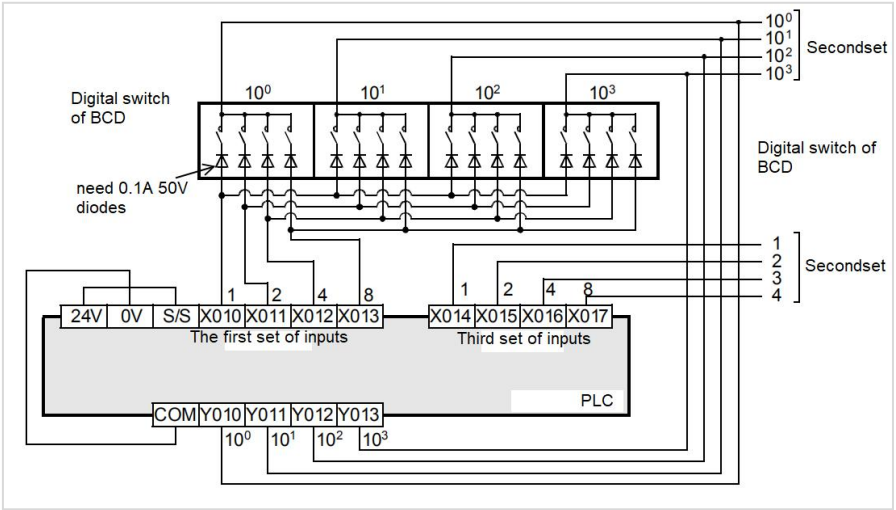
Error code	Content
4084H	The data input in the application instruction (n) and (d2) exceeds the specified range
4085H	The output result of the read application instruction (s) and (d2) exceeds the device range
4086H	The output result of the write application instruction (d1) and (d2) exceeds the device range
4089H	The number of application instructions exceeds the limit

Example

Program



Wiring diagram



① DSW operates while M1 (digital switch read input) is ON.

② DSW will operate until the end of one cycle of operation and the instruction execution end flag (SM229) turns ON.

6.7.3 HKY/Hexadecimal numeric key input

HKY

Use the keyboard (16 keys) of 0 to F to input, set the numerical value (0 to 9) and operating conditions (A to F function keys) and other instructions for data input.

When the extended function is ON, the hexadecimal number of the 0 to F keys could be used for keyboard input.

-[HKY (s) (d1) (d2) (d3)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start bit device (X) number that input 16-key (occupies 4 points)	-	Bit	ANY_BOOL
(d1)	The start device (Y) number that outputs (occupies 4 points)	-	Bit	ANY_BOOL
(d2)	The device number that stores the value input from the 16 keys	0 to 9999	BIN16 bit	ANY16_S
(d3)	The start bit device number whose key is ON (occupies 8 points)	-	BIN16 bit	ANY16_S

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
HKY	Parameter 1	●																							●	
	Parameter 2		●																						●	
	Parameter 3															●	●	●	●	●					●	
	Parameter 4		●	●	●	●					●														●	

Features

16-bit operation (HKY)

Scan the input [S to S+3] and column output [D1 to D1+3] signals connected with 16 keys (0 to F), press the 0 to 9 keys, the value will be saved in D2, and the keyboard detection will be output to D3 +7 in.

In addition, after pressing the A to F keys, the key information corresponding to the keyboard [D3 to D3+5] is ON, and the keyboard detection is output to D3+6.

(1) About using the keys 0 to 9 to input the values D3, D3+7

If it is more than 9,999, overflow from the high digit. The entered value is stored in D2 as BIN (binary number).

When any key from 0 to 9 is pressed, the keyboard detection output D3+7 is ON.

(2) Information about A to F keys D3 to D3+6

Corresponding to the A to F keys, the first 6 o'clock of D3 is ON. When any key from A to F is pressed, the keyboard detection output D3+6 is ON.

Keyboard	Key information
A	D3
B	D3+1
C	D3+2
D	D3+3
E	D3+4
F	D3+5

Extensions

After SM167 is ON and the extended function becomes valid, the data of the hexadecimal keys from 0 to F is saved in BIN mode.

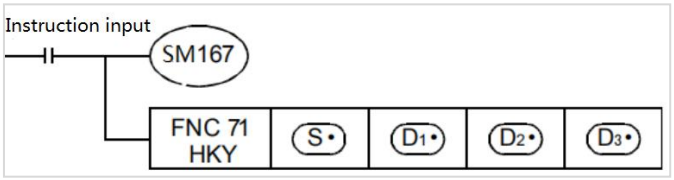
Except for the following, it is the same as the above-mentioned [Function and Operation Description].

The hexadecimal data input using the 0 to F keys is written into D2 as it is.

(1) Regarding the numerical input using the 0 to F keys D2

When it is FFFF or more, overflow from the upper digits.

For example, when inputting 1→2→3→B→F, “23BF" is saved in D2 in BIN mode. When F is input, 1 overflows.



✎Note

① Restrictions on the number of uses of instructions

HKY instructions, only one of them can be used in the program.

② When the keyboard is pressed simultaneously

When multiple keys are pressed at the same time, the key pressed first is effective.

③ When the instruction contact is OFF

Even if it is OFF, the content of D2 does not change, but D3 to D3 +7 all become OFF.

④ Number of occupied points of the device

When 16 keys are connected, 4 points from the start device S of input (X) are occupied.

When 16 keys are connected, 4 points from the start device D1 of output (Y) are occupied.

It occupies 8 points from the start device D3 for key information output.

Please do not to overlap with the devices used in other controls of the machine.

D3 to D3+5: A to F key key information

D3+6: Keyboard detection output of A to F keys

D3+7: 0-9 key keyboard detection output

⑤ About the read timing of keyboard input

HKY instruction is executed synchronously with the operation cycle of the programmable controller.

It takes 8 operation cycles to complete a series of keyboard scans.

In order to prevent reading omissions caused by the filter delay of keyboard input, please use the [Constant Scan Mode] and [Timer Interrupt] functions flexibly.

⑥ Output form

Please use a transistor output type programmable controller.

Related device

Devices	Name	Content
SM229	End of instruction execution	OFF: (d1) to (d1)+3 is being scanned, or the instruction is not executed ON: (d1) to (d1)+3 cyclic output operation (1 to 4 digit scan) and then turn ON

Error code

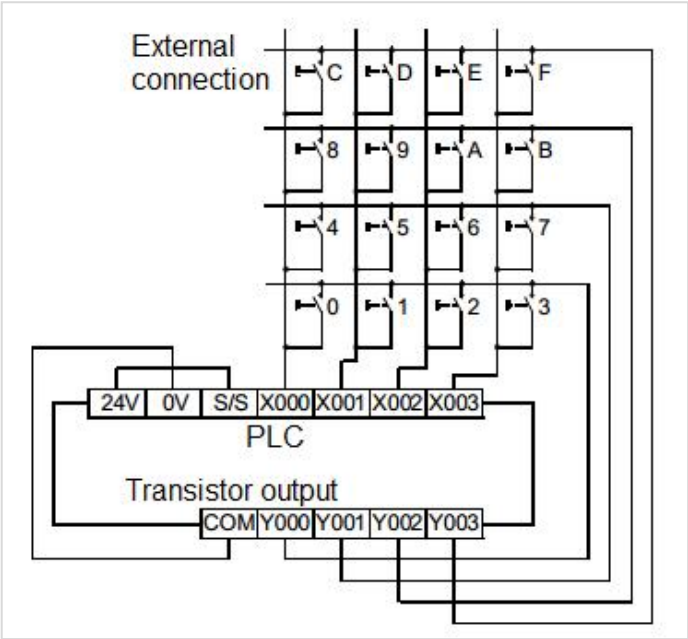
Error code	Content
4085H	The output result of the read application instruction (s) and (d2) exceeds the device range
4086H	The output result of the write application instruction (d1), (d2) and (d3) exceeds the device range

Example

Program



Wiring diagram



When inputting [1]→[2]→[3]→[B]→[F], save "23BF" in D0 in BIN mode.

When [F] is input, [1] overflows.

6.7.4 DHKY/32 System Numeric Key Input

DHKY

Use the keyboard (16 keys) of 0 to F to input, set numerical value (0 to 9) and operating conditions (A to F function keys) and other instructions for data input.

When the extended function is ON, the hexadecimal number of 0 to F key can be used for keyboard input.

-[DHKY (s) (d1) (d2) (d3)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start bit device (X) number that input 16-key (occupies 4 points)	-	Bit	ANY_BOOL
(d1)	The start device (Y) number that outputs (occupies 4 points)	-	Bit	ANY_BOOL
(d2)	The device number that stores the value input from the 16 keys	0 to 99999999	BIN32 bit	ANY32_S
(d3)	The start bit device number whose key is ON (occupies 8 points)	-	BIN16 bit	ANY16_S

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DHKY	Parameter 1	●																							●	
	Parameter 2		●																						●	
	Parameter 3															●	●	●	●	●	●	●				●	
	Parameter 4		●	●	●	●					●															●	

Features

32-bit operation (DHKY)

Scan the input [S to S+3] and column output [D1 to D1+3] signals connected with 16 keys (0 to F), press the 0 to 9 keys, and the value will be saved in [D2+1, D2] , The keyboard detection is output to D3+7.

In addition, after pressing the A to F keys, the key information corresponding to the keyboard [D3 to D3+5] is ON, and the keyboard detection is output to D3+6.

(1) Regarding the use of keys from 0 to 9 to input values [D2+1, D2], D3+7

If it is 99,999,999 or more, overflow from the high digit.

The entered value is stored in [D2+1, D2] as BIN (binary number).

When any key from 0 to 9 is pressed, the keyboard detection output D3+7 is ON.

(2) Button information about A to F keys D3 to D3+6

For keyboard press information, please refer to 16-bit operation (HKY) on the previous page

extensions

After SM167 is ON and the extended function becomes valid, the data of the hexadecimal keys from 0 to F is saved in BIN mode.

Except for the following, it is the same as the above-mentioned “Function and Operation Description”.

The hexadecimal data input using the 0 to F keys are written in [D2+1, D2] as they are.

(1) Regarding the numerical input using 0 to F keys [D2+1, D2]

-When it is FFFFFFFF or more, overflow from the upper digits.

For example, when inputting [9]→[2]→[3]→[B]→[F]→[A]→[F], save "923BFAF" in [D2+1, D2] in BIN mode.

✎Note

① Restrictions on the number of uses of instructions

Only one of the DHKY instructions can be used in the program.

② When the keyboard is pressed simultaneously

When multiple keys are pressed at the same time, the key pressed first is effective.

③ When the instruction contact is OFF

Even if it is OFF, the content of D2 does not change, but D3 to D3 +7 all become OFF.

④ Number of occupied points of the device

When 16 keys are connected, 4 points from the start device S of input (X) are occupied.

When 16 keys are connected, 4 points from the start device D1 of output (Y) are occupied.

It occupies 8 points from the start device D3 for key information output.

Please be careful not to overlap with the devices used in other controls of the machine.

D3 to D3+5: A to F key key information

D3+6: Keyboard detection output of A to F keys

D3+7: 0-9 key keyboard detection output

⑤ About the read timing of keyboard input

The DHKY instruction is executed synchronously with the operation cycle of the programmable controller.

It takes 8 operation cycles to complete a series of keyboard scans.

In order to prevent reading omissions caused by the filter delay of keyboard input, please use the “Constant Scan Mode” and “Timer Interrupt” functions flexibly.

⑥ Output form

Please use a transistor output type programmable controller.

Related device

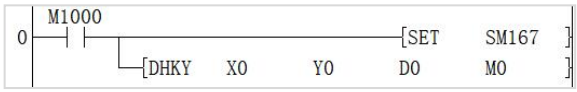
Devices	Name	Content
SM229	End of instruction execution	OFF: (d1) to (d1)+3 is being scanned, or the instruction is not executed ON: (d1) to (d1)+3 cyclic output operation (1 to 4 digit scan) and then turn ON

Error code

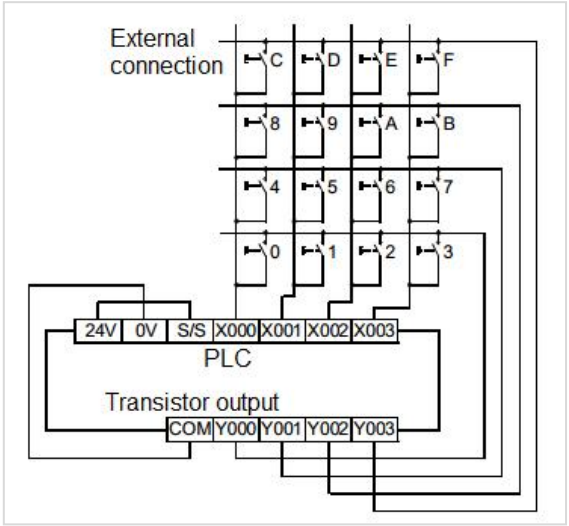
Error code	Content
4085H	The output result of the read application instruction (s) and (d2) exceeds the device range
4086H	The output result of the write application instruction (d1), (d2) and (d3) exceeds the device range

Example

Program



.Wiring diagram



When inputting 1→2→3→B→F→5→7→6, save "123BF576" in BIN to [D1,D0].

6.7.5 PR/ASCII code printing

PR

This instruction is to output ASCII data in parallel to the output (Y).

-[PR (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Start number of the device storing ASCII code data		String (ASCII code only)	ANY_ASC
(d)	The start number Y of output ASCII code data	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PR	Parameter 1															●	●	●	●	●						●	
	Parameter 2		●																							●	

Features

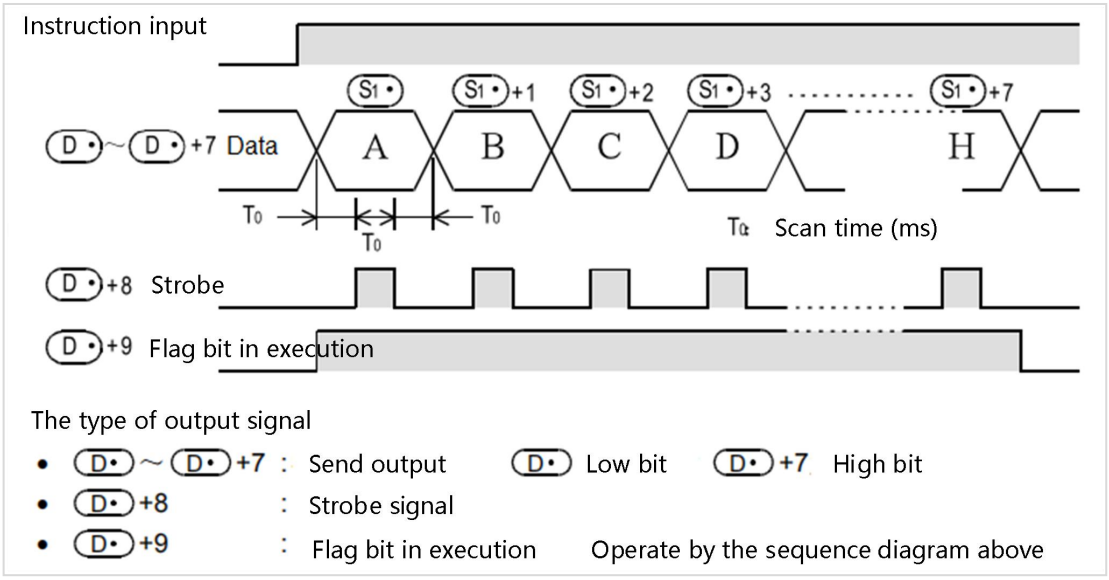
The ASCII code stored in the lower 8 bits (1 byte) of (S) to (S)+7 is output to (D) to (D)+7 character by character in a time division manner.

The ASCII code saved in is shown below, and the following timing diagram is based on this example.

The sequence of sending starts from (S) = "A", and ends with (S) + 7 = "H" for this purpose, sending eight bytes.

S	S+1	S+2	S+3	S+4	S+5	S+6	S+7
A(H41)	B(H42)	C(H43)	D(H44)	E(H45)	F(H46)	G(H47)	H(H48)

Timing diagram



Related device

Devices	Name	Content
SM227	PR mode	OFF: 8 bytes serial output (fixed to 8 characters) ON: 16 bytes serial output (1 to 16 characters)

✎Note

① instruction input and instruction action

instruction input=ON: Even if the instruction is continuously ON or the pulse instruction is executed, as long as the output of one cycle ends, the execution ends.

SM229 only works when SM227=ON.

instruction input=OFF: all outputs are OFF.

② Relationship with scan time (operation time)

The instruction is executed synchronously with the scan time.

When the scan time is short, you can use the constant scan mode to drive; when the scan time is longer, you can use the timer interrupt drive.

③ About the output of the programmable controller

Please use a transistor output type programmable controller.

④ When 00H (NUL) exists in the data (when SM227=ON)

After the instruction is executed, the remaining data is not output.

In addition, SM229 maintains an operation cycle ON.

⑤ Restrictions on the number of uses of instructions

Only one PR instruction can be used in the program.

Error code

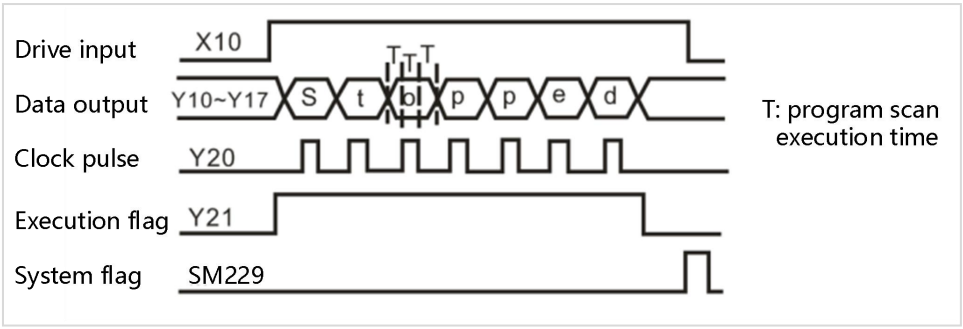
Error code	Content
4085H	The output result of the read application instruction (s) exceeds the device range
4086H	The output result of the write application instruction (d) exceeds the device range

Example

Program



If the ASCII code in D200 to D203 is "Stopped", the corresponding output port signal and its timing are as follows:



6.7.6 SEGD/Numeric key input

SEGD(P)

instruction to light up the 7-segment digital tube (1 digit).

-[SEGD (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Decoded start word device	-32767 to 32767	Bit	ANY_BOOL
(d)	Word device number for storing 7-segment display data	-32767 to 32767	Bit	ANY_BOOL

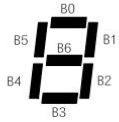
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SEGD	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Features

Decode the low 4-digit (1 digit) of 0 to F (hexadecimal number) of (S) into 7-segment display data and save it in the low 8-digit of (d).

1.7-segment code decode table

(S·) HEX	b3	b2	b1	b0	Seven segment code	(D·) B15	…	B8	B7	B6	B5	B4	B3	B2	B1	B0	Display
0	0	0	0	0		—		—	0	0	1	1	1	1	1	1	0
1	0	0	0	1		—		—	0	0	0	0	0	1	1	0	1
2	0	0	1	0		—		—	0	1	0	1	1	0	1	1	2
3	0	0	1	1		—		—	0	1	0	0	1	1	1	1	3
4	0	1	0	0		—		—	0	1	1	0	0	1	1	0	4
5	0	1	0	1		—		—	0	1	1	0	1	1	0	1	5
6	0	1	1	0		—		—	0	1	1	1	1	1	0	1	6
7	0	1	1	1		—		—	0	0	1	0	0	1	1	1	7
8	1	0	0	0		—		—	0	1	1	1	1	1	1	1	8
9	1	0	0	1		—		—	0	1	1	0	1	1	1	1	9
A	1	0	1	0		—		—	0	1	1	1	0	1	1	1	A
B	1	0	1	1		—		—	0	1	1	1	1	1	0	0	b
C	1	1	0	0		—		—	0	0	1	1	1	0	0	1	C
D	1	1	0	1		—		—	0	1	0	1	1	1	1	0	d
E	1	1	1	0		—		—	0	1	1	1	1	0	0	1	E
F	1	1	1	1		—		—	0	1	1	1	0	0	0	1	F

↑ The start of the bit device, or the lowest bit of the word device is B0

✎Note

Number of occupied points of the device: The low 8 bits of the output of the device (S) are occupied, and the high 8 bits do not change.

Error code

Error code	Content
4085H	The output result of the read application instruction (s) and (d) exceeds the device range
4086H	The output result of the write application instruction (d) exceeds the device range

Example



When M0 is set, the lower 4 bits of the data in D0 are decoded and output to the Y10 to Y17 ports. The corresponding table for translation is shown in the above table (7-segment code decoding table). The table does not need to be prepared by the user, and the comparison table is already available in the PLC system.

6.7.7 SEGL/7SEG vode hour and minute display

SEGL

Control 1 or 2 groups of 4-digit 7-segment digital tube display instructions with latch.

-[SEGL (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Start word device for BCD conversion	0 to 9999	BIN16 bit	ANY16
(d)	The starting Y number to be output	-	Bit	ANY_BOOL
(n)	Parameter number [Setting range: K0(H0) to K7(H7)]	0 to 7	BIN16/32 bit	ANY16_U

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SEGL	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 2		●																							●	
	Parameter 3																						●	●			

Features

Convert the 4-bit value of (s) into BCD data, and use the time-division method to sequentially output each 1 digit to a 7-segment digital tube with BCD decoding. (s) is valid when BIN data in the range of 0 to 9999.

The parameter (n) should be set as follows based on the positive and negative logic on the programmable controller side and the positive and negative logic on the 7-segment side.

Programmable controller output logic	Data input	Strobe signal	Parameter n	
			4 digits in 1 group	4 digits in 2 groups
Negative logic	Negative logic (consistent)	Negative logic (consistent)	0	4
		Positive logic (inconsistent)	1	5
	Positive logic (inconsistent)	Negative logic (consistent)	2	6
		Positive logic (inconsistent)	3	7
Positive logic	Positive logic (consistent)	Negative logic (consistent)	0	4
		Positive logic (inconsistent)	1	5
	Negative logic (inconsistent)	Negative logic (consistent)	2	6
		Positive logic (inconsistent)	3	7

(1) When using 4 digits in 1 group (n=K0 to 3)

After converting the 4-digit value of (s) from BIN→BCD, use the time division method to output each digit in turn from (d) to (d)+3. In addition, the strobe signal output (d)+4 to (d)+7 is also output in a time-division manner, locked to the 7-terminal display of the first group of 4 digits

(2) When using 4 digits in 2 groups (n=K4 to 7)

1) 4-digit group 1

After converting the 4-digit value of (s) from BIN→BCD, use the time division method to output each digit in turn from (d) to (d)+3. The strobe signal output (d)+4 to (d)+7 is output in time-division manner in turn, locked to the 7-segment display of the first group of 4 digits.

2) 4-digit group 2

After converting the 4-digit value of (s)+1 from BIN+BCD, use the time division method to output each digit in turn from (d)+10 to (d)+13. The strobe signal output (d)+4 to (d)+7 is output in a time-division manner in turn, locked to the 7-segment display of the

second group of digits.

✎Note

① About the time required to update the 7-segment 4-digit display

The time required to update the 4-digit display (1 group or 2 groups) is 12 times the scan time (operation time).

② Action when instruction input is OFF

When the instruction input is ON, the action is repeated. However, if the instruction contact turns off during an action, the action will be interrupted. When it is ON again, it will start from the original action.

③ Occupied points of the device

When using 4 digits in 1 group: 1 point from the start device specified in S is occupied.

Occupy 8 points from the start device specified in D. Even when the number of bits is small, the occupied points cannot be used for other purposes.

When using 4 digits 2 groups: 2 points from the start device specified in S are occupied.

Occupy 12 points from the start device specified in D. Even when the number of bits is small, the occupied points cannot be used for other purposes.

④ About scan time (operation cycle) and display timing

The SEGL instruction is executed synchronously with the scan time (operation cycle) of the programmable controller.

In order to perform a series of displays, the scan time of the programmable controller needs to exceed 10ms.

When it is less than 10ms, please use the constant scan mode and run with a scan time longer than 10ms.

⑤ Regarding the output format of the programmable controller

Please use a transistor output type programmable controller.

⑥ Limit number of instructions

At most 2 instructions can be used at the same time.

Related device

Devices	Name	Content
SM229	End of instruction execution	After the processing is completed, SM229 is ON for one scan cycle

Error code

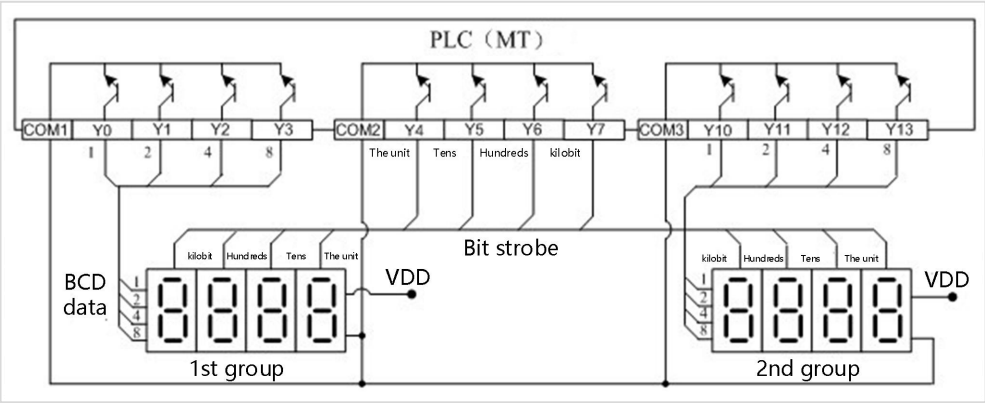
Error code	Content
4084H	The data input in the application instruction (n) exceeds the specified range
4085H	The output result of the read application instruction (s) exceeds the device range
4086H	The output result of the write application instruction (d) exceeds the device range
4089H	The number of application instructions exceeds the limit

Example

Program



The corresponding hardware wiring is shown in the following figure. The content of D0 is displayed on the first group of digital tubes, and the content of D1 is displayed on the second group of digital tubes. If the reading of D0 or D1 exceeds 9999, the program will run into an error:



The digital tube used in the wiring diagram has its own display data latch, 7-segment decoding and driving, and 7-segment digital of negative logic type (when the input port is low, it means that the input data is 1, or is strobed) Show tube. During display processing, PLC's Y4 to Y7 ports will scan automatically, and only one port is ON each time as a bit strobe signal. At this time, the data on Y0 to Y3 ports is the BCD code data sent to the corresponding bit. When the bit strobe signal turns from ON→OFF, it is latched into the latch in the digital tube. After internal decoding and driving, the digital tube displays the number. The PLC system cyclically processes Y4 to Y7 in turn, until all 4 bits are processed. In the same way, Y10 to Y13 are the data output ports of the second group of 4-digit digital tubes, which share the bit strobe lines of Y4 to Y7. The processing methods are the same, and the display processing of the two groups is performed at the same time. In the example, if D0=K2468 and D1=K9753, the first group will display 2 4 6 8 and the second group will display 9 7 5 3.

It takes 12 scan cycles to complete a display refresh. After the processing is completed: According to the positive and negative logic of the programmable controller, the positive and negative logic of the seven-segment code, etc., select according to the following principles:

For a group of 4 digits, n=0 to 3. When two groups of 4 digits, n= 4 to 7.

Display group number	Group 1				Group 2			
Y data output polarity	PNP		NPN		PNP		NPN	
Strobe and data polarity	Identical	Opposite	Identical	Opposite	Identical	Opposite	Identical	Opposite
the value of n	0	1	2	3	4	5	6	7

6.7.8 TKY/Numeric key input

TKY

Use the keyboard (number keys) of 0 to 9 to input instructions for setting data such as timers and counters.

-[TKY (s) (d1) (d2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start bit device that input the numeric key (occupies 10 points)	-	Bit	ANY_BOOL
(d1)	Word device number for storing data	0 to 9999	Signed BIN16	ANY16_S
(d2)	The start bit device number whose key start bit device is ON (occupies 11 points)	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TKY	Parameter 1	●	●	●	●	●					●															●	
	Parameter 2												●	●	●	●	●	●	●	●						●	
	Parameter 3		●	●	●	●					●															●	

Features

Input [(s) to +9] to the connected number keys and press the keyboard, save the input value in (d1), and output in (d2) to +10 Keyboard input information and detected keyboard output.

(1) About the input value (d1)

If it is more than 9,999, overflow from the high digit.

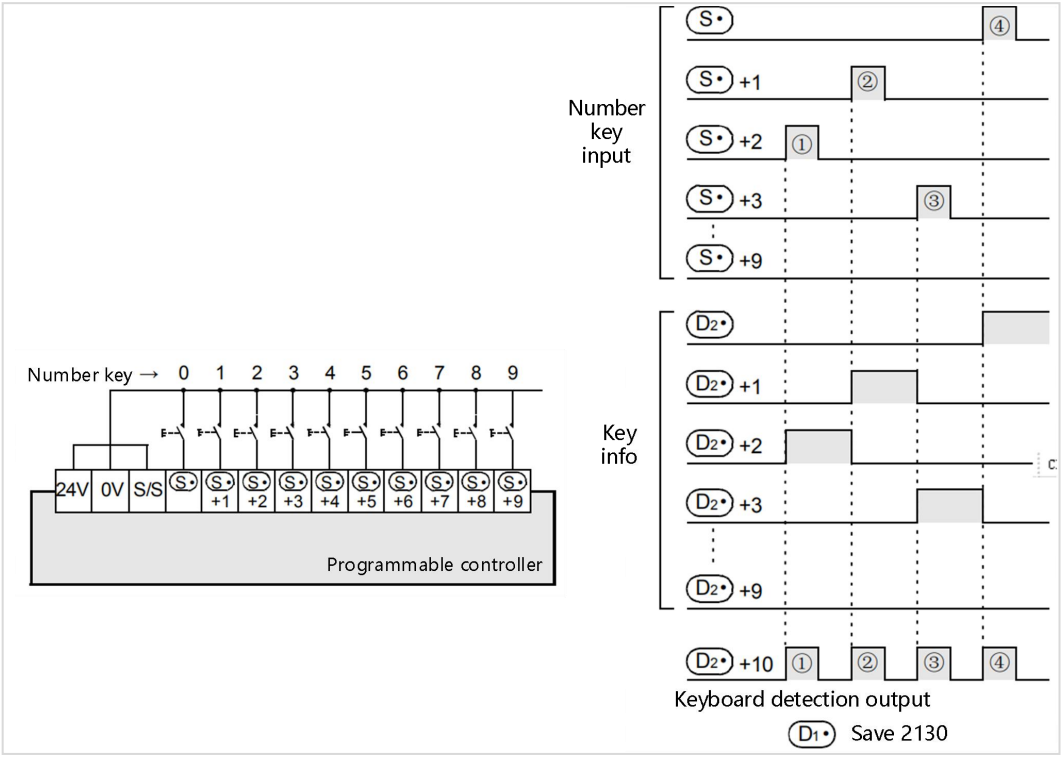
The entered value is saved in BIN (binary number).

After pressing the number keys in the order of ①, ②, ③, ④, it is stored as 2130 in (d1).

(2) About (d2) to 10 of key information

(d2) to 9 key information, according to the pressed key ON/OFF.

When any key from 0 to 9 is pressed, the keyboard detection output of (d2) +10 is ON.



✎Note

① When the keyboard is pressed simultaneously

When multiple keys are pressed at the same time, only the key pressed first is effective.

② When the instruction contact is OFF

Even if it is OFF, the content of (d2) will not change, but (d2) to (d2)+10 will be OFF.

③ Occupied points of the device

✎ Connect the input of the number keys, occupying 10 points from (s).

Even when the number key is not connected (not used), since (d2) is already occupied, it cannot be used for other purposes.

✎ It occupies 11 points from the start device (d2) for key information output.

Please be careful not to overlap with the devices used in other controls of the machine.

(D2) to (d2)+9: Turn ON according to the input of number keys 0 to 9.

(D2)+10: It is ON when any key between 0 to 9 is pressed. (Keyboard detection output)

④ Restrictions on the number of uses of instructions

Only one of the TKY instruction or DTKY instruction can be used in the program.

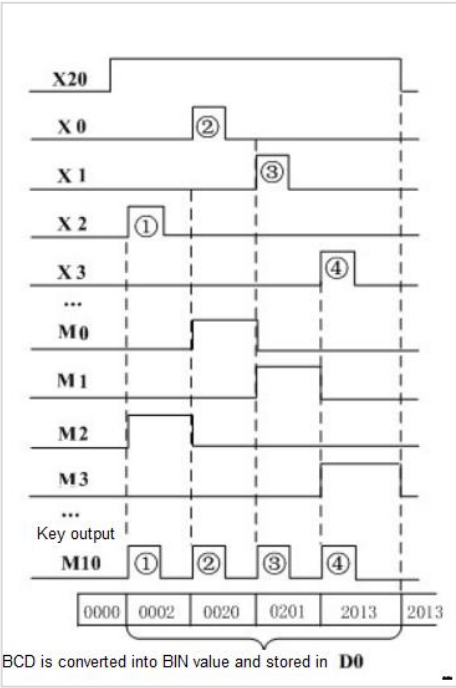
Error code

Error code	Content
4085H	The output result of the read application instruction (s) exceeds the device range
4086H	The output result of the write application instruction (d) exceeds the device range

Example



To input the number "2013", press the keys 2, 0, 1, 3 (X2, X0, X1, X3) in order. The operation of the PLC internal variables is shown in the figure below.



According to the parameter setting in the instruction, X0toX11 correspond to 0to9 numeric keys; M0toM9 correspond to the state of the keys; when any key is pressed, the key output unit M10 will be set;

The key value (such as 2013) is converted to BIN format and stored in the specified D1 unit D0; (D0=0x7DD), even if the power flow of the drive turns OFF, D0 will not change;

When multiple keys are pressed, the first detected key is valid; when the input number exceeds 4 digits, the first input number changes overflow, leaving only the last 4 numbers input.

6.7.9 DTKY/Numeric key input

DTKY

Use the 4 points starting from the device specified in (d) to perform 4 types of timer output.

-[STMR (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start bit device that input the numeric key (occupies 10 points)	-	Bit	ANY_BOOL
(d1)	Word device number for storing data	0 to 99999999	Signed BIN32	ANY32_S
(d2)	The start bit device number whose key start bit device is ON [occupies 11 points]	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DTKY	Parameter 1	●	●	●	●	●					●															●	
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	
	Parameter 3		●	●	●	●					●															●	

Features

Input [(s) to +9] to the connected number keys and press the keyboard, save the input value in (d1), and output in (d2) to +10

Keyboard input information and detected keyboard output.

(1) About the input value (d1)

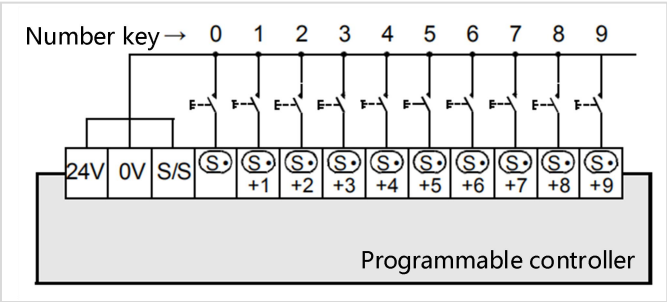
If it is more than 9,999, overflow from the high digit.

The entered value is saved in BIN (binary number).

(2) (d2) to 10 of key information

(d2) to +9 key information, according to the pressed key ON/OFF.

When any key from 0 to 9 is pressed, the keyboard detection output of (d2) +10 is ON.



✎Note

① When the keyboard is pressed simultaneously

② When multiple keys are pressed at the same time, only the key pressed first is effective.

③ When the instruction contact is OFF

Even if it is OFF, the content of (d2) will not change, but (d2) to (d2)+10 will be OFF.

④ Occupied points of the device

✎ Connect the input of the number keys, occupying 10 points from (s).

Even when the number key is not connected (not used), since (d2) is already occupied, it cannot be used for other purposes.

✎ It occupies 11 points from the start device (d2) for key information output.

Please be careful not to overlap with the devices used in other controls of the machine.

(D2) to (d2)+9: Turn ON according to the input of number keys 0 to 9.

(D2)+10: It is ON when any key between 0 to 9 is pressed. (Keyboard detection output)

⑤ Restrictions on the number of uses of instructions

Only one of the TKY instruction or DTKY instruction can be used in the program.

Error code

Error code	Content
4085H	The output result of the read application instruction (s) and (d1) exceeds the device range
4086H	The output result of the write application instruction (d1) and (d2) exceeds the device range

Example



When X20 is on, if you want to input the number "20205689", press 2, 0, 2, 0, 5, 6, 8, 9 (X2, X0, X2, X0, X5, X6, X10, X11) in sequence , Then (the value in (D1,D0) is 20205689)

6.8 Data conversion instruction

6.8.1 BCD/BIN → BCD

BCD(P)

Convert the BIN data of the device specified in (s) to BCD, and store it in the device specified in (d).

The calculation of the CPU module uses BIN (binary number) data for processing, which is used to display values in a 7-segment display equipped with a BCD decoder.

-[BCD (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	BIN data or start device storing BIN data	0 to 9999	Signed BIN16	ANY16
(d)	Start device for storing BCD data	-	BCD 4 digits	ANY16

Device used

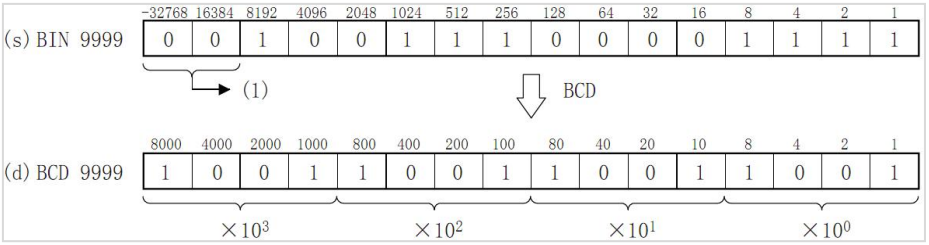
instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BCD	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●			●	●		●	●

Features

The BIN 16-bit data (0 to 9999) of the device specified in (s) is converted to BCD 4-bit data and stored in the device specified in (d).

The data specified in (s) can be converted within the range of 0 to 9999 (BCD).

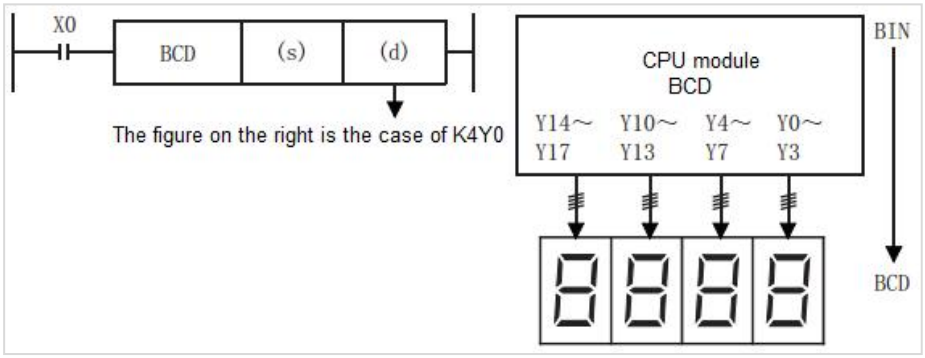
When the data specified in (s) or (d) is digit specification, the conditions are as shown in the table below.



(1): Must be set to 0.

The data specified in (s) can be converted in the range of K0 to K9999 by BCD (decimal number).

When the data specified in (s) or (d) is digit specification, the conditions are as shown in the table below.



(d)	Digits	Data range
K1Y0	1-bit	0 to 9
K2Y0	2-bit	00 to 99
K3Y0	3-bit	000 to 999
K4Y0	4-bit	0000 to 9999

✎Note

The four arithmetic operations (+-×÷), increment, decrement instructions and other operations in the CPU module are all performed by BIN (binary number). Therefore, when sending BCD (decimal) digital switch information to the CPU module, please use the BIN(P) instruction (BCD→BIN conversion transfer instruction). In addition, when outputting to the 7-segment display of BCD (decimal number), please use the BCD(P) instruction (BIN→BCD conversion transmission).

Error code

Error code	Content
4084H	The data input in the application instruction (s) exceeds the specified range
4085H	The output result of the read application instruction (s) exceeds the device range
4086H	The output result of the write application instruction (d) exceeds the device range

Example



When M0 is set, the BIN value of D200 is converted into BCD and stored in K1Y0.

6.8.2 BIN/4-bit BCD → BIN

BIN(P)

Convert the BCD data of the device specified in (s) to BIN and store it in the device specified in (d).

Similar to the digital switch, it converts the value set in BCD (decimal number) to BIN (binary number) that can be operated by the CPU module and is used for reading.

-[BIN (s) (d)]

Content, range and data type

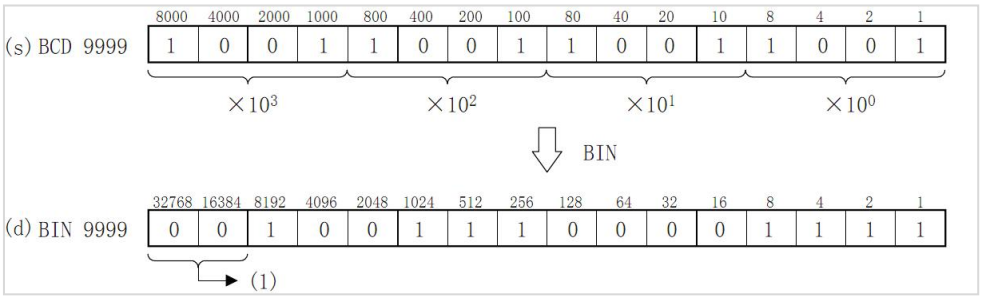
Parameter	Content	Range	Data type	Data type (label)
(s)	BCD data or start device storing BIN data	0 to 9999	BCD 4 digits	ANY16
(d)	Start device for storing BIN data	-	Signed BIN16	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BIN	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Features

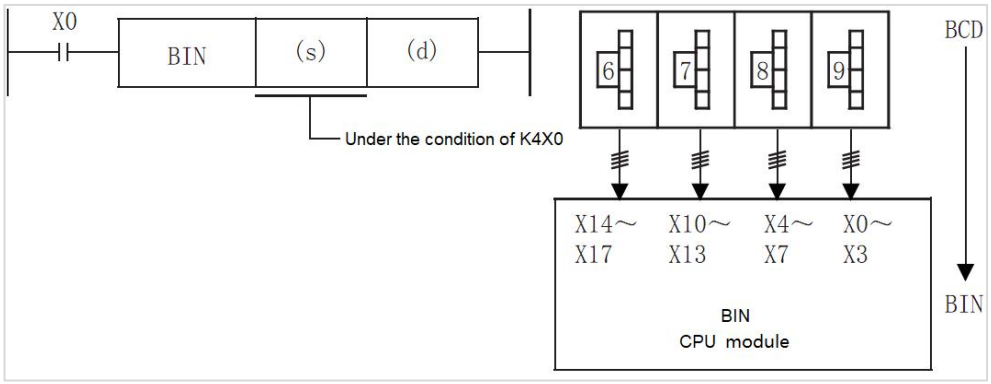
BCD 4-bit data (0 to 9999) of the device specified in (s) is converted into BIN 16-bit data and stored in the device specified in (d).



(1): Must become 0.

The data specified in (s) can be converted within the range of 0 to 9999 (BCD).

When the data specified in (s) or (d) is digit specification, the conditions are as shown in the table below.



(d)	Digits	Data range
K1X0	1-bit	0 to 9
K2X0	2-bit	00 to 99
K3X0	3-bit	000 to 999
K4X0	4-bit	0000 to 9999

✎Note

The calculations in the CPU module such as the four arithmetic operations (+-×÷), increment and decrement instructions are all

performed by BIN (binary number). Therefore, when sending BCD (decimal) digital switch information to the CPU module, please use the BIN(P) instruction (BCD→BIN conversion transfer instruction). In addition, when outputting to the 7-segment display of BCD (decimal number), please use the BCD(P) instruction (BIN→BCD conversion transmission).

Error code

Error code	Content
4084H	The data input in the application instruction (s) exceeds the specified range
4085H	The output result of the read application instruction (s) exceeds the device range
4086H	The output result of the write application instruction (d) exceeds the device range

Example



When M0 is set, the BCD value of K1Y0 is converted into BIN and stored in D200.

6.8.3 DBIN/8-bit BCD → BIN

DBIN(P)

Convert the BCD data of the device specified in (s) to BIN and store it in the device specified in (d).

Similar to the digital switch, it converts the value set in BCD (decimal number) to BIN (binary number) that can be operated by the CPU module and is used for reading.

-[DBIN (s) (d)]

Content, range and data type

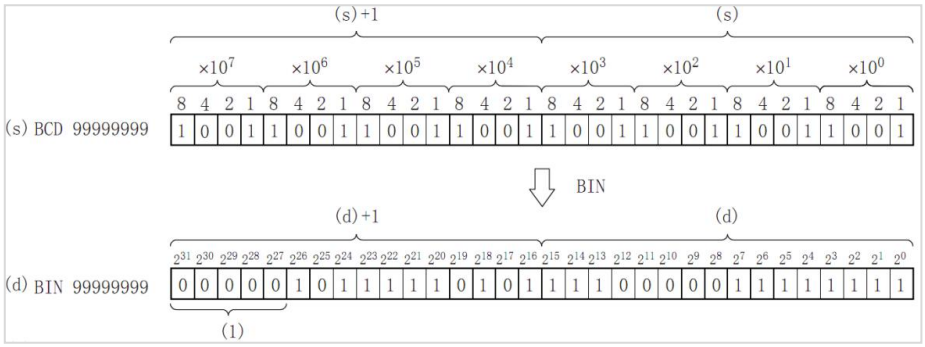
Parameter	Content	Range	Data type	Data type (label)
(s)	BCD data or start device storing BIN data	0 to 99999999	BCD 8 digits	ANY32
(d)	Start device for storing BIN data	-	Signed BIN32	ANY32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBIN	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●							●	●

Features

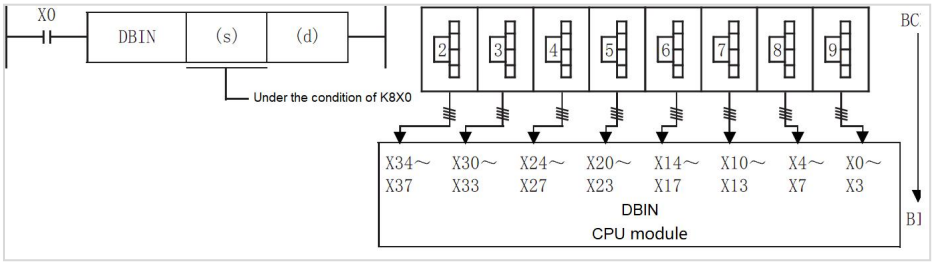
BCD 8-bit data (0 to 99999999) of the device specified in (s) is converted to BIN 32-bit data and stored in the device specified in (d).



(1): Must become 0.

The data specified in (s) can be converted within the range of 0 to 99999999 (BCD).

When the data specified in (s) or (d) is digit specification, the conditions are as shown in the table below.



(d)	Bit	Data range
K1X0	1-bit	0 to 9
K2X0	2-bit	00 to 99
K3X0	3-bit	000 to 999
K4X0	4-bit	0000 to 9999
K5X0	5-bit	00000 to 99999
K6X0	6-bit	000000 to 999999
K7X0	7-bit	0000000 to 9999999
K8X0	8-bit	00000000 to 99999999

✎Note

The calculations in the CPU module such as the four arithmetic operations (+-×÷), increment and decrement instructions are all performed by BIN (binary number). Therefore, when sending BCD (decimal) digital switch information to the CPU module, please use the BIN(P) instruction (BCD→BIN conversion transfer instruction). In addition, when outputting to the 7-segment display of BCD (decimal number), please use the BCD(P) instruction (BIN→BCD conversion transmission).

Error code

Error code	Content
4084H	The data input in the application instruction (s) exceeds the specified range
4085H	The output result of the read application instruction (s) exceeds the device range
4086H	The output result of the write application instruction (d) exceeds the device range

Example



When M0 is set, the BCD value of K8Y0 is converted into BIN and stored in D200.

6.8.4 FLT/BIN Integer → Binary floating point number

FLT(P)

An instruction to convert a BIN 16-bit integer value into a binary floating point number (real number).

-[FLT (s) (d)]

Content, range and data type

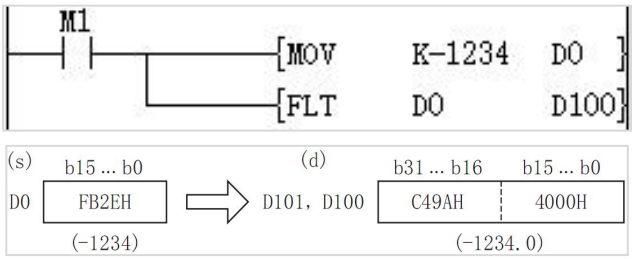
Parameter	Content	Range	Data type	Data type (label)
(s)	The data register number that saves the BIN integer value	-	Signed BIN 16 bit	ANY16
(d)	The data register number that saves the binary floating-point number (real number)	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
FLT	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

The signed 16-bit data specified in (s) is converted into a binary floating point data and stored in (d)+1, (d).



✎Note

In each binary floating point number (real number) operation instruction, the specified K and H values will be automatically converted into a binary floating point number (real number), so there is no need to use the FLT instruction for conversion.

The inverse conversion instruction of this instruction is INT (convert a binary floating point value into a BIN integer).

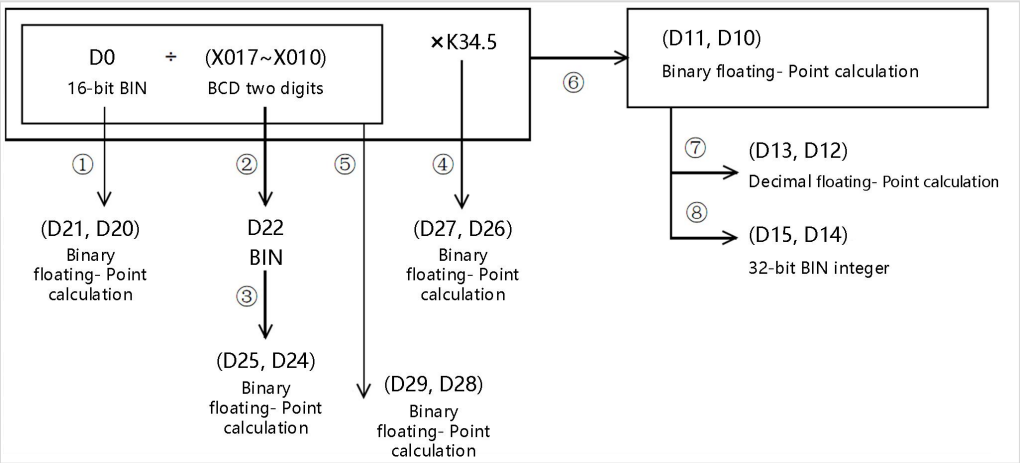
Error code

Error code	Content
4085H	When the device specified in the read application instruction (s) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

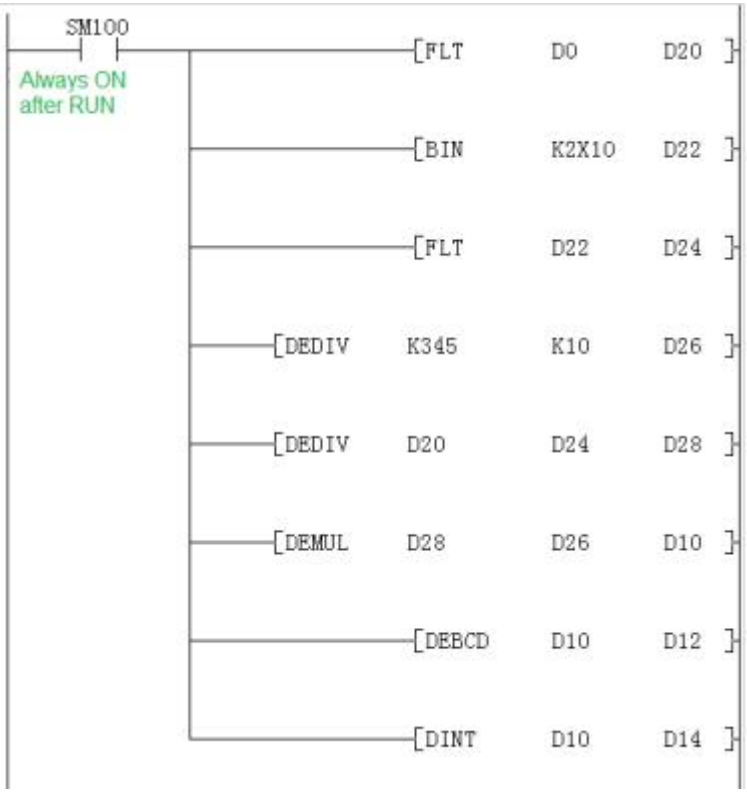
Example

Four arithmetic using binary floating point operations

(1) Calculation example



(2) Sequence control program



(D0) → (D21, D20)
BIN binary floating point operations

(X17 to X10) → (D22)
BCD BIN

(D22) → (D25, D24)
BIN binary floating point operations

K345÷K10 → (D27, D26)
Binary floating point operations

(D21,D20)÷(D25,D24) → (D29,D28)
Binary floating-point number division operation
→Binary floating-point number operation

(D29,D28)×(D27,D26) → (D11,D10)
Binary floating-point number multiplication

(D11,D10) → (D13,D12)
Binary floating-point calculations →
Decimal floating-point calculations monitoring

(D11,D10) → (D13,D12)
Binary floating point operations 32-bit BIN integer

6.8.5 DFLT/BIN Integer → Binary floating point number

DFLT(P)

An instruction to convert a BIN 32-bit integer value into a binary floating point number (real number).

-[DFLT (s) (d)]

Content, range and data type

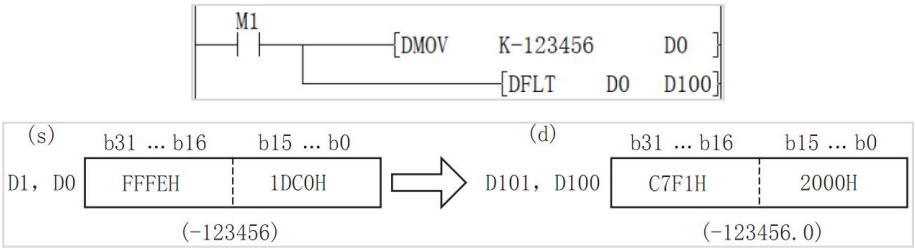
Parameter	Content	Range	Data type	Data type (label)
(s)	The data register number that saves the BIN32 integer value	-	Signed BIN 32 bit	ANY32
(d)	The data register number that saves the binary floating-point number (real number)	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DFLT	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

Convert the signed BIN 32-bit data specified in (s) to binary floating point data and store them in (d)+1, (d).



✎Note

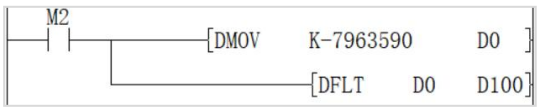
In each binary floating-point number (real number) operation instruction, the specified K and H values are automatically converted into a binary floating-point number (real number), so there is no need to use the DFLT instruction for conversion.

The inverse conversion instruction of this instruction is INT (convert a binary floating point value into a BIN integer).

Error code

Error code	Content
4085H	When the device specified in the read application instruction (s) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range

Example



When M2=ON, convert the BIN 32-bit integer -7963590 in [D1, D0] into a single-precision floating point number -7963590.0 and store it in the [D101, D100] device.

6.8.6 VAL/ String → BIN 16-bit data conversion

VAL(P)

After converting the character string stored in the device number specified in (s) and later into BIN 16-bit data, store the number of digits in (d1) and store the BIN data in (d2).

[VAL (s) (d1) (d2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The character string converted to BIN data or the start device that stores the character string	-	String	ANYSTRING_SINGLE
(d1)	The start device that stores the number of digits of converted BIN data	-	Signed BIN 16 bit	ANY16_S_ARRAY
(d2)	Start device for storing converted BIN data	-	Signed BIN 16 bit	ANY16_S

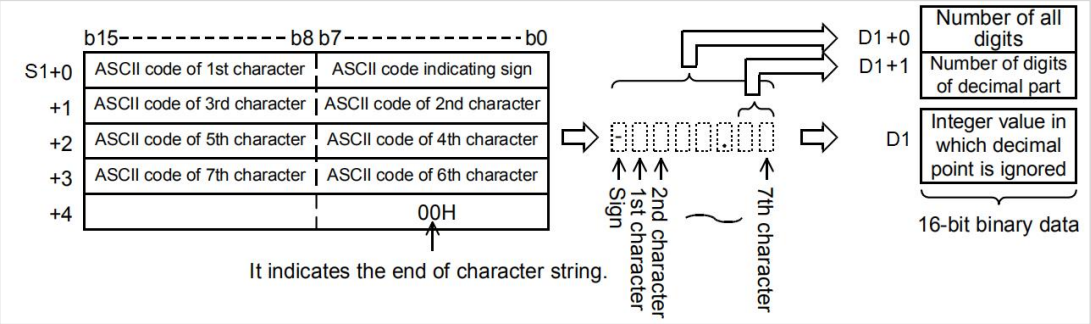
Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
VAL	Parameter 1															●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

Features

After converting the character string stored in the device number specified in (s) and later into BIN 16-bit data, store the number of digits in (d1) and store the BIN data in (d2). In the conversion from character string to BIN, the data from the device number specified in (s) to the device number storing 00H is treated as a character string.

The total number of digits stored in (d1) stores the number of all characters (including signs and decimal points) representing the value. The number of decimal places stored in (d1)+1 stores the number of characters representing the decimal part after 2EH(.). For the BIN 16-bit data stored in (d2), the character string ignoring the decimal point is converted into a BIN value and stored.

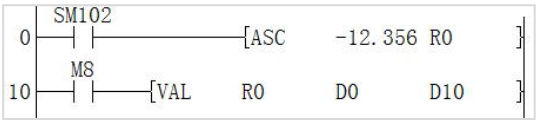


Error code

Error code	Content
4082H	The character string specified by (s) could not be converted into a numeric value For example: The first character is not a negative sign or a space, space appears in the middle of the number, decimal point appears twice. Except for the first character that appears non-characters and decimal points, the number in the symbolic string with the decimal point is removed and the range between -32768 and 32767 is exceeded Except for the first character, there are non-character and decimal Signs For example, 3.4000 is 34000 after removing the decimal point, which is out of range.
4085H	(s) read address exceeds the device range
408AH	When the character number of character string the specified in (s) is other than 2 to 8.

Error code	Content
408BH	The maximum range of the device is read when (s) taking character string, but 00H is not found as the end
4086H	When using offset, the offset address of (d) exceeds the device range

Example



The result obtained above:

D0 corresponds to str length is 7.

D1 corresponds to a decimal point length of 3.

D0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	7
D1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

D10 corresponds to -12356 ignoring the decimal point.

Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D10	0	0	1	1	1	1	0	1	1	1	1	1	0	0	1	1	-12356

6.8.7 DVAL/String → BIN32-bit Data Conversion

DVAL(P)

After converting the character string stored in the device number specified in (s) into BIN 32-bit data, store the number of digits in (d1) and store the BIN data in (d2).

-[DVAL (s) (d1) (d2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The character string converted to BIN data or the start device that stores the character string	-	String	ANYSTRING_SINGLE
(d1)	The start device that stores the number of digits of converted BIN data	-	Signed BIN 16 bit	ANY16_S_ARRAY
(d2)	Start device for storing converted BIN data	-	Signed BIN 32 bit	ANY32_S

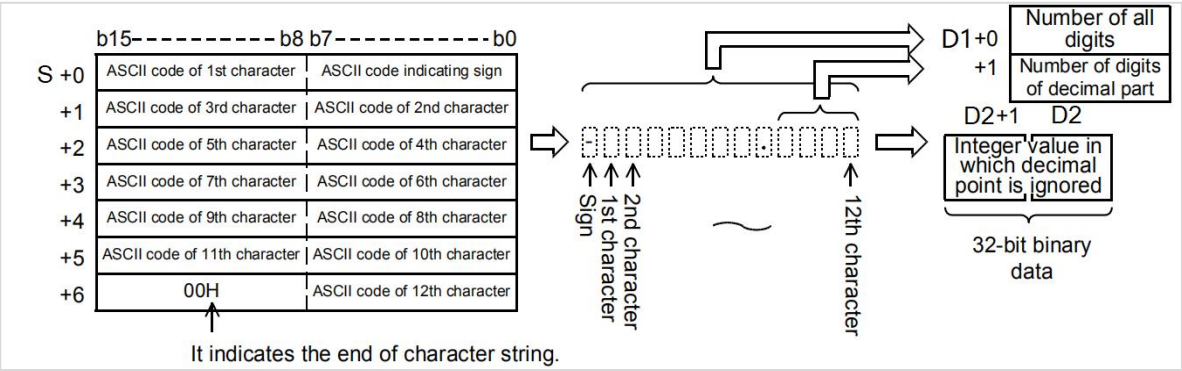
Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DVAL	Parameter 1															●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

After converting the character string stored in the device number specified in (s) into BIN 32-bit data, store the number of digits in (d1) and store the BIN data in (d2). In the conversion from character string to BIN, the data from the device number specified in (s) to the device number storing 00H is treated as a character string.

The total number of digits stored in (d1) stores the number of all characters (including signs and decimal points) representing the value. The number of decimal places stored in (d1)+1 stores the number of characters representing the decimal part after 2EH(.). For the BIN 32-bit data stored in (d2), the character string ignoring the decimal point is converted into a BIN value and stored.

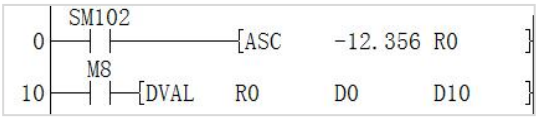


Error code

Error code	Content
4082H	The character string specified by (s) could not be converted into a numeric value. For example: The first character is not a negative sign or a space, space appears in the middle of the number, decimal point appears twice. Except for the first character that appears non-characters and decimal points, the number in the symbolic string with the decimal point is removed and the range between -2147483648 and 2147483647 is exceeded Except for the first character, there are non-character and decimal Signs For example, 3.000000000 is 3000000000 after removing the decimal point, which is out of range.
4085H	(s) read address exceeds the device range

Error code	Content
408AH	When the character number of character string the specified in (s) is other than 2 to 13.
408BH	The maximum range of the device is read when (d1) and (d2) taking character string, but 00H is not found as the end
4086H	When using offset, the offset address of (d) exceeds the device range

Example



The result obtained above

D0 corresponds to str length is 7.

D1 corresponds to a decimal point length of 3.

D0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	7
D1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	3

D10 corresponds to -12356 ignoring the decimal point

Device	+0	+1	+2	+3
D0	196615	0	0	0
D8	0	-12356	0	0

6.8.8 ASCI/HEX code data →ASCII conversion

ASCI(P)

After the n characters (bits) in the HEX code data specified in (s) are converted into ASCII codes, they are stored after the device number specified in (d).

-[ASCI (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of the device storing the HEX code to be converted	-	BIN16 bit	ANY16
(d)	The start number of the device storing the converted ASCII code	-	String	ANYSTRING_SINGLE
(n)	The number of characters (digits) of the HEX code to be converted	1to256	BIN16 bit	ANY16_U

Device used

Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ASCI	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3																	●	●	●			●	●			●

Features

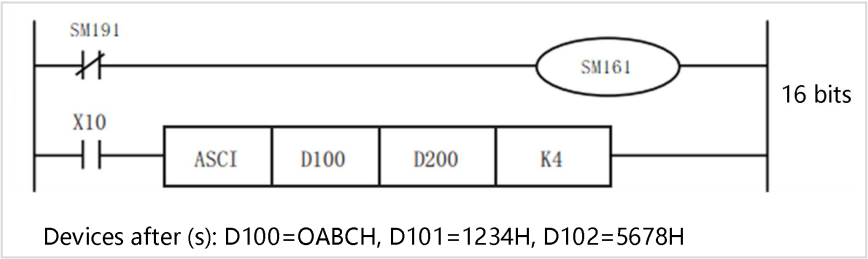
The number of characters (bits) specified by (n) in the HEX code data specified in (s) is converted into ASCII code and stored in the device number specified in (d) or later.

ASCI(P) instruction uses 16-bit mode and 8-bit mode when converting. For the operation of each mode, please refer to the following content.

(1) 16-bit conversion mode (when SM8161=OFF)

Convert the digits of the HEX code after the device specified in (s) into ASCII, and transfer to the upper and lower 8 bits (bytes) of the device specified in (d). When using in 16-bit conversion mode, SM161 should always be turned OFF.

In the case of the following program, perform the conversion as shown below.

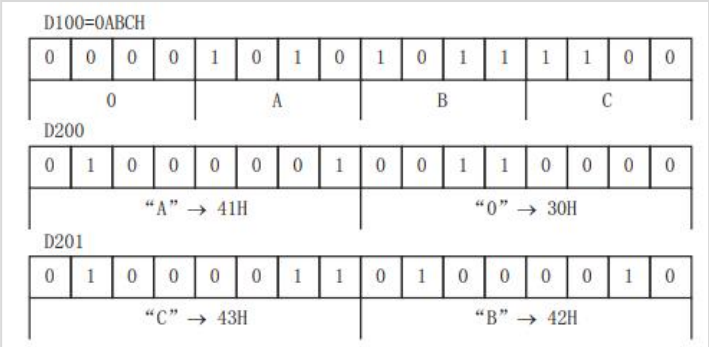


Devices after (s): D100=OABCH, D101=1234H, D102=5678H

Specify the number of bits (characters) and the conversion result

(n) / (d)	K1	K2	K3	K4	K5	K6	K7	K8	K9
Under D200	C	B	C	0	4	3	2	1	8
D200 on		C	B	C	0	4	3	2	1
Under D201			C	B	C	0	4	3	2
D201 on				C	B	C	0	4	3
Under D202					C	B	C	0	4
D202 on			Unchanged			C	B	C	0
Under D203							C	B	C
D203 on								C	B
Under D204									C

Bit structure in the case of (n)=K4



ASCII code:

"0"=30H "1"=32H "5"=35H

"A"=41H "2"=32H "6"=36H

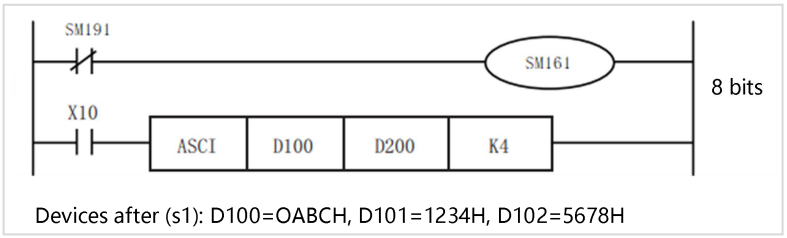
"B"=42H "3"=33H "7"=37H

"C"=43H "4"=34H "8"=38H

(2) 8-bit conversion mode (when SM161=ON)

Convert the digits of the HEX code after the device specified in (s) into ASCII, and transfer to the lower 8 bits (bytes) of the device specified in (d). When using in 8-bit conversion mode, SM161 should always be set to ON for use.

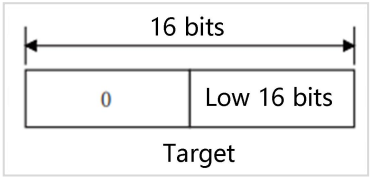
In the case of the following program, perform the conversion as shown below.



Devices after (s1): D100=OABCH, D101=1234H, D102=5678H

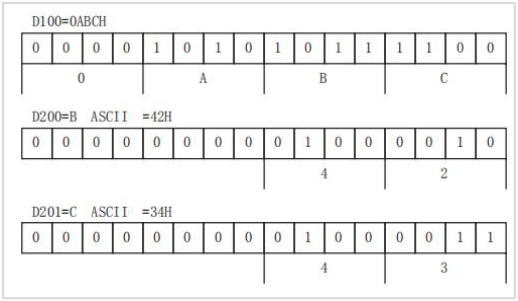
If SM161 is set to ON, it will become 8-bit mode,

Perform conversion processing as shown below.



(n) / (d)	K1	K2	K3	K4	K5	K6	K7	K8	K9
D200	C	B	C	0	4	3	2	1	8
D201		C	B	C	0	4	3	2	1
D202			C	B	C	0	4	3	2
D203				C	B	C	0	4	3
D204					C	B	C	0	4
D205		Unchanged				C	B	C	0
D206							C	B	C
D207								C	B
D208									C

Bit structure in the case of (n)=K2



ASCII

"0"=30H "1"=31H "5"=35H

"A"=41H "2"=32H "6"=36H

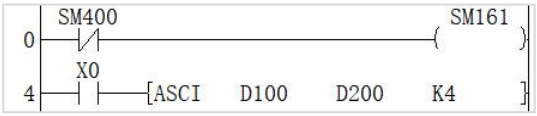
"B"=42H "3"=33H "7"=37H

"C"=43H "4"=34H "8"=38H

Error code

Error code	Content
4085H	When the specified device range is read to exceed the corresponding device range
4086H	When the specified device range is written to exceed the corresponding device range
4084H	When the value specified in (n) exceeds the range of 1 to 256

Example



• 16-bit conversion mode (when SM161=OFF)

Convert the digits of the HEX code after the device specified in d100 into ASCII, and transfer to the upper and lower 8 bits (bytes) of the device specified in d200. When using in 16-bit conversion mode, SM161 should always be turned OFF.

6.8.9 HEX/ASCII → HEX code data conversion

HEX(P)

After the device number specified in (s), the ASCII data stored in the number of characters specified in (n) is converted to HEX code, and then stored in the device number specified in (d) or later.

-[HEX (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device that stores the ASCII data converted to HEX code	-	String	ANYSTRING_SINGLE
(d)	The start device that stores converted HEX code	-	BIN16 bit	ANY16
(n)	Number of characters (bytes) of converted ASCII data	1 to 256	BIN16 bit	ANY16_U

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
HEX	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3																	●	●	●			●	●			●

Features

• After the device number specified in (s), the ASCII data stored in the number of characters specified in (n) is converted to HEX code, and then stored in the device number specified in (d) or later. The HEX(P) instruction uses 16-bit conversion mode and 8-bit conversion mode when converting. For the operation of each mode, please refer to the following content.

(1) 16-bit conversion mode (when SM161=OFF)

After converting the ASCII data stored in the upper and lower 8 digits (bytes) of the device specified in (s) into HEX code, it transmits every 4 digits to the device specified in (d). The number of characters to be converted is specified in (n).

SM161 is shared with ASC, ASCI, BCC, CCD and CRC instructions. When using in 16-bit conversion mode, please always set SM161 to OFF.

SM161 is cleared when RUN→STOP.

In addition, it is necessary to store the ASCII data in the 16-bit conversion mode in the upper 8 bits of the device specified in (s).

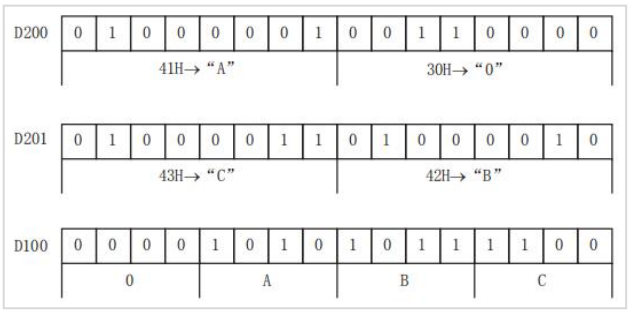
In the following program, the conversion will be performed in the following manner.



Transform the source data

(s)	ASCII data	HEX conversion
Under D200	30H	0
D200 on	41H	A
Under D201	42H	B
D201 on	43H	C
Under D202	31H	1
D202 on	32H	2
Under D203	33H	3
D203 on	34H	4
Under D204	35H	5

Bit structure in the case of (n)=K4



The number of characters specified and the conversion result becomes 0.

(n)		1	2	3	4	5	6	7	8	9
(d)	D102	Unchanged								...OH
	D101					...OH	..OAH	.OABH	OABCH	ABC1H
	D100	...OH	..OAH	.OABH	OABCH	ABC1H	BC12H	C123H	1234H	2345H

(2) 8-bit conversion mode (when SM161=ON)

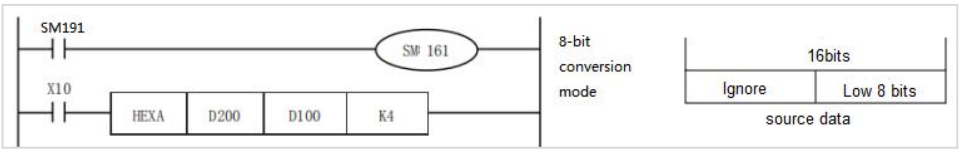
After converting the ASCII data stored in the lower 8 digits of the device specified in (s) into HEX code, it will be transmitted to the device specified in (d) every 4 digits.

The number of characters to be converted is specified in (n).

SM161 is shared with ASC, ASCI, BCC, CCD and CRC instructions. When using in 8-bit conversion mode, please always turn on SM161.

SM161 is cleared when RUN→STOP.

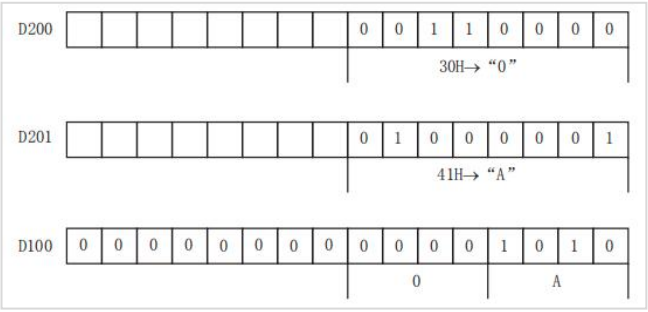
In the following program, the conversion will be performed in the following manner.



Transform the source data

(s)	ASCII data	HEX conversion
D200	30H	0
D201	41H	A
D202	42H	B
D203	43H	C
D204	31H	1
D205	32H	2
D206	33H	3
D207	34H	4
D208	35H	5

Bit structure in the case of (n)=K2



The number of characters specified and the conversion result becomes 0.

(n)		1	2	3	4	5	6	7	8	9
(d)	D102	Unchanged								...OH
	D101					...OH	..OAH	.OABH	OABCH	ABC1H
	D100	...OH	..OAH	.OABH	OABCH	ABC1H	BC12H	C123H	1234H	2345H

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the range.
	When ASCII codes other than 30H to 39H and 41H to 46H are set in (s).
4085H	When the specified device range is read to exceed the corresponding device range
4086H	When the specified device range is written to exceed the corresponding device range

Example



After converting the ASCII data stored in the upper and lower 8 digits (bytes) of the device specified in (s) into HEX code, it transmits every 4 digits to the device specified in (d). The number of characters to be converted is specified in (n).

SM161 is shared with ASC, ASCI, BCC, CCD and CRC instructions. When using in 16-bit conversion mode, please always set SM161 to OFF.

6.8.10 CCD/Check Code

CCD(P)

Calculate the horizontal parity value and the sum check value of the error checking method used in communication and the like. In addition to these error checking methods, there are CRC (Cyclic

Redundancy Check). To calculate the CRC value, use the CRC(P) instruction.

-[CCD (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of object device	-	BIN16 bit	ANY16
(d)	The start number of the storage destination device of the calculated data	-	BIN16 bit	ANY16_ARRAY (number of elements: 2)
(n)	Number of data	1 to 256	BIN16 bit	ANY16_U

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CCD	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3																	●	●	●			●	●			●

Features

Calculate the addition data and horizontal parity data of the data stored in (s) to (s)+(n)-1, and store the addition data in (d), horizontal parity

The data is stored in (d)+1. The modes used by this instruction in calculation are 16-bit mode and 8-bit mode. For the operation of each mode, please refer to the following content.

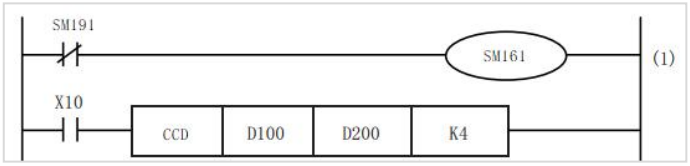
(1) 16-bit conversion mode (when SM161=OFF)

Regarding the data at point (n) starting with (s), the addition data and horizontal parity data of the high and low 8-bit data are stored in the Devicess (d) and (d)+1.

SM161 is shared with ASC, ASCI, BCC, CCD and CRC instructions. When using in 16 bits, always set to OFF for use.

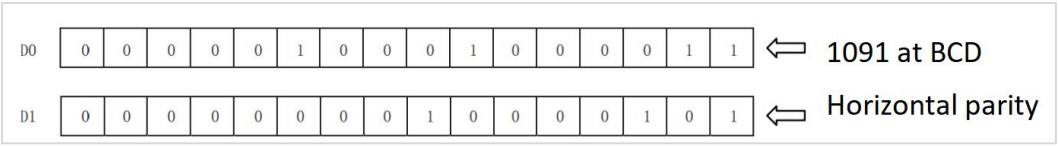
SM161 is cleared when RUN→STOP.

In the case of the following program, perform the conversion as shown below.



16-bit conversion mode

(s)	Example of the content of the data	
D100 DOWN	K100 = 0 1 1 0 0 1 0 0	
D100 UP	K111 = 0 1 1 0 1 1 1 ①	←
D101 DOWN	K100 = 0 1 1 0 0 1 0 0	
D101 UP	K 98 = 0 1 1 0 0 0 1 0	
D102 DOWN	K123 = 0 1 1 1 1 0 1 ①	← If the number of 1 is odd, the horizontal parity is 1
D102 UP	K 66 = 0 1 0 0 0 0 1 0	If the number of 1 is even, the horizontal parity is 0
D103 DOWN	K100 = 0 1 1 0 0 1 0 0	
D103 UP	K 95 = 0 1 0 1 1 1 1 ①	←
D104 DOWN	K210 = 1 1 0 1 0 0 1 0	
D104 UP	K 88 = 0 1 0 1 1 0 0 0	
Total	K1091	
Horizontal parity	1 0 0 0 0 1 0 ①	←



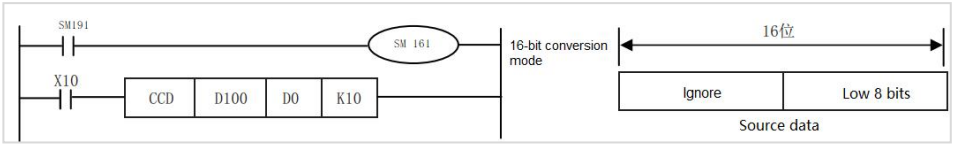
(2) 8-bit conversion mode (when SM161=ON)

Regarding (s) as the starting point (n) data (lower 8 bits only), its addition data and horizontal parity data are stored in the devices (d) and (d)+1.

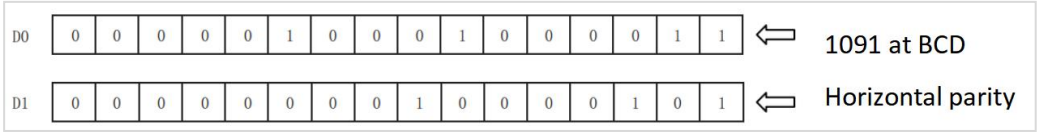
SM161 is shared with ASC, ASCI, BCC, CCD and CRC instructions. If it is used in 8 bits, it should always be set to ON for use.

SM161 is cleared when RUN→STOP.

In the case of the following program, perform the conversion as shown below.



(s)	Example of the content of the data	
D100	K100 = 0 1 1 0 0 1 0 0	
D101	K111 = 0 1 1 0 1 1 1 ①	←
D102	K100 = 0 1 1 0 0 1 0 0	
D103	K 98 = 0 1 1 0 0 0 1 0	
D104	K123 = 0 1 1 1 1 0 1 ①	← If the number of 1 is odd, the horizontal parity is 1
D105	K 66 = 0 1 0 0 0 0 1 0	If the number of 1 is even, the horizontal parity is 0
D106	K100 = 0 1 1 0 0 1 0 0	
D107	K 95 = 0 1 0 1 1 1 1 ①	←
D108	K210 = 1 1 0 1 0 0 1 0	
D109	K 88 = 0 1 0 1 1 0 0 0	
Total	K1091	
Horizontal parity	1 0 0 0 0 1 0 ①	←



Error code

Error code	Content
4084H	When the value specified in (n) exceed the range of 1 to 256.
4085H	When the specified device range is read to exceed the corresponding device range
4086H	When the specified device range is written to exceed the corresponding device range

Example



Regarding D10 as the initial 10-point data, the addition data and horizontal parity data of the high and low 8-bit data are stored in the Devices of D0 and D0+1.

SM161 is shared with ASC, ASCI, BCC, CCD and CRC instructions. When using in 16 bits, always set to OFF for use.

6.8.11 GBIN/Gray code → BIN 16-bit data conversion

GBIN(P)

Convert the BIN 16-bit Gray code data stored in the device specified in (s) into BIN 16-bit data, and store it in the device specified in (d).

-[GBIN (s) (d)]

Content, range and data type

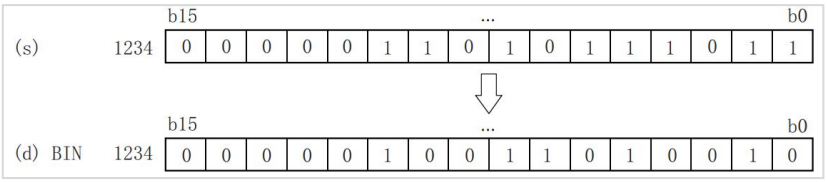
Parameter	Content	Range	Data type	Data type (label)
(s)	Gray code data or the start device that stores Gray code	0 to 32767	BIN16 bit	ANY16_S
(d)	The start device that stores the converted BIN data	-	BIN16 bit	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
GBIN	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Features

Convert the BIN 16-bit Gray code data stored in the device specified in (s) into BIN 16-bit data, and store it in the device specified in (d).



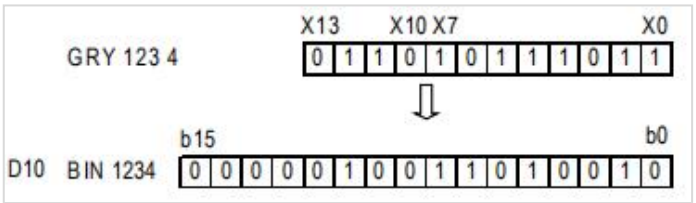
GRY→BIN Mathematical Algorithm: Starting from the second bit from the left, XOR each bit with the decoded value of the left bit as the decoded value of the bit (the leftmost bit remains unchanged).

Error code

Error code	Content
4084H	When the value specified in (s) exceeds the range
4085H	When the specified device range is read to exceed the range of the corresponding device
4086H	When the specified device range is written to exceed the range of the corresponding device

Example





It could be used when the encoder of Gray code method is used to detect the absolute position.

For S, the numerical are valid in the range of 0 to 32767.

6.8.12 DGBIN/Gray code → BIN32-bit data conversion

DGBIN(P)

Convert the BIN32-bit Gray code data stored in the device specified in (s) to BIN 32-bit data and store it in the device specified in (d).

Content, range and data type

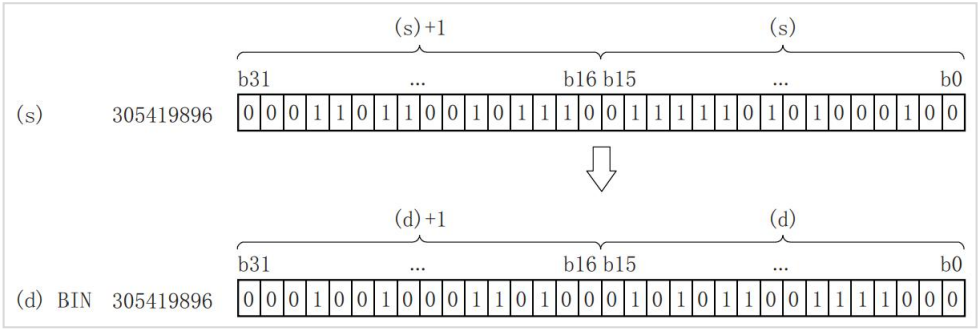
Parameter	Content	Range	Data type	Data type (label)
(s)	Gray code data or the start device that stores Gray code	0 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device that stores converted BIN data	-	BIN32 bit	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DGBIN	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●

Features

Convert the BIN32-bit Gray code data stored in the device specified in (s) into BIN 32-bit data, and store it in the device specified in (d).



(s)+1: high 16 bits

(s): low 16 bits

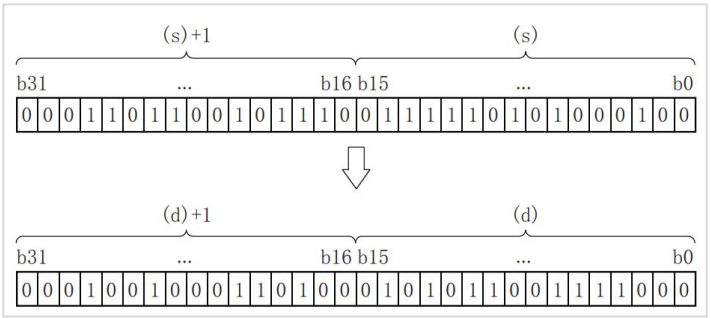
GRY→BIN Mathematical Algorithm: Starting from the second bit from the left, XOR each bit with the decoded value of the left bit as the decoded value of the bit (the leftmost bit remains unchanged).

Error code

Error code	Content
4084H	When the value specified in (s) exceeds the range
4085H	When the specified device range is read to exceed the corresponding device range
4086H	When the specified device range is written to exceed the corresponding device range

Example





6.8.13 GRY/BIN 16-bit data → Gray code conversion

GRY(P)

After converting the BIN 16-bit data of the device specified in (s) to BIN 16-bit Gray code data, it is stored in the device specified in (d).

-[GRY (s) (d)]

Content, range and data type

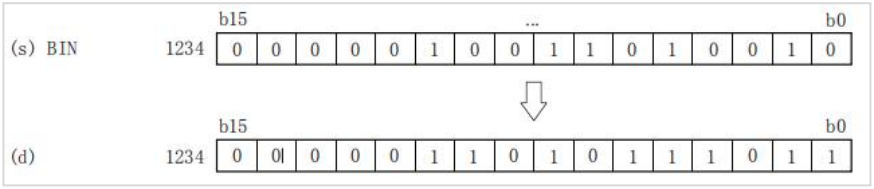
Parameter	Content	Range	Data type	Data type (label)
(s)	BIN data or the start device that stores BIN data	0 to 32767	BIN16 bit	ANY16_S
(d)	The start device that stores the converted Gray code	-	BIN16 bit	ANY16_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
GRY	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Features

Convert the BIN 16-bit data specified in (s) into BIN 16-bit Gray code, and store it in the device specified in (d).



BIN→GRY Mathematical Algorithm: Starting from the rightmost bit, XOR each bit with the left bit as the value corresponding to the GRY bit, and the leftmost bit remains unchanged (equivalent to 0 on the left) .

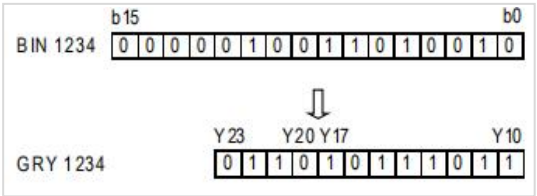
Error code

Error code	Content
4084H	When the value specified in (s) exceeds the range
4085H	When the specified device range is read to exceed the corresponding device range
4086H	When the specified device range is written to exceed the corresponding device range

Example



As shown in the above ladder diagram:



For S, the range of 0 to 32767 is valid.

6.8.14 DGRY/BIN 32-bit data → Gray code conversion

DGRY(P)

After converting the BIN 16-bit data of the device specified in (s) to BIN 16-bit Gray code data, it is stored in the device specified in (d).

-[GRY (s) (d)]

Content, range and data type

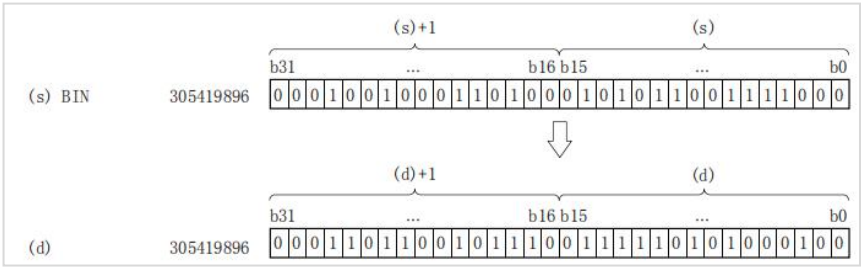
Parameter	Content	Range	Data type	Data type (label)
(s)	BIN data or the start device that stores BIN data	0 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device that stores the converted Gray code	-	BIN32 bit	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DGRY	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●

Features

Convert the BIN32-bit data specified in (s) into BIN32-bit Gray code and store it in the device specified in (d)



(s)+1: high 16 bits

(s): low 16 bits

BIN→GRY Mathematical Algorithm: Starting from the rightmost bit, XOR each bit with the left bit as the value corresponding to the GRY bit, and the leftmost bit remains unchanged (equivalent to 0 on the left) .

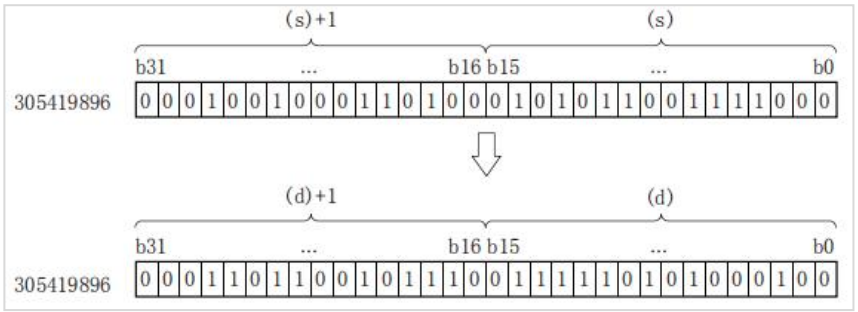
Error code

Error code	Content
4084H	When the value specified in (s) exceeds the range
4085H	When the specified device range is read to exceed the corresponding device range
4086H	When the specified device range is written to exceed the corresponding device range

Example



As shown in the above ladder diagram:



6.8.15 PRUN/Octal digit transmission (16-bit data)

PRUN(P)

After processing the device numbers of (s) and (d) with specified digits as octal numbers, transfer the data.

-[PRUN (s) (d)]

Content, range and data type

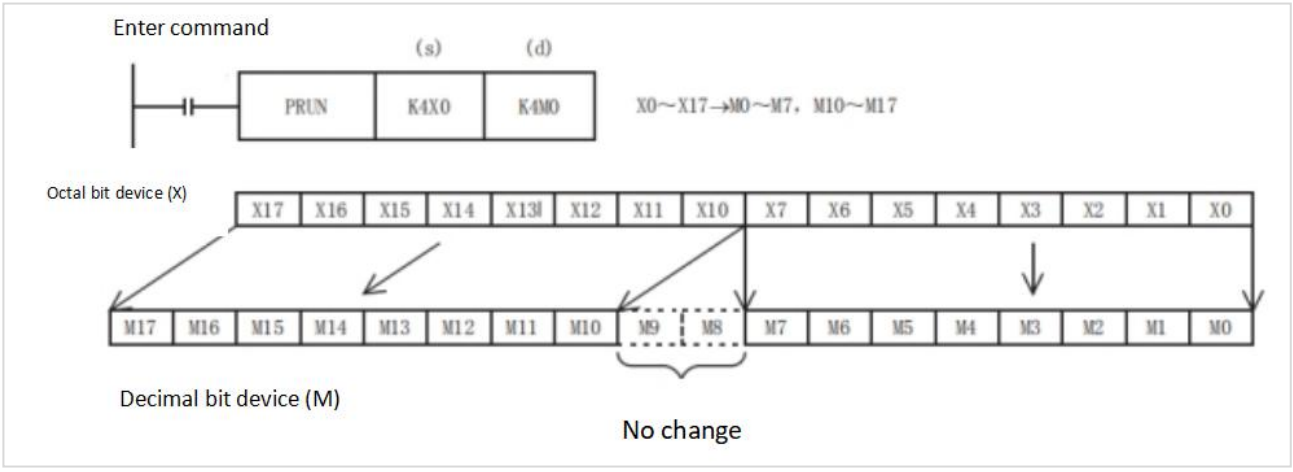
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Digit specification*1	-	BIN16 bit	ANY16	--
(d)	Transfer target device number*1	-	BIN16 bit	ANY16	--

Device used

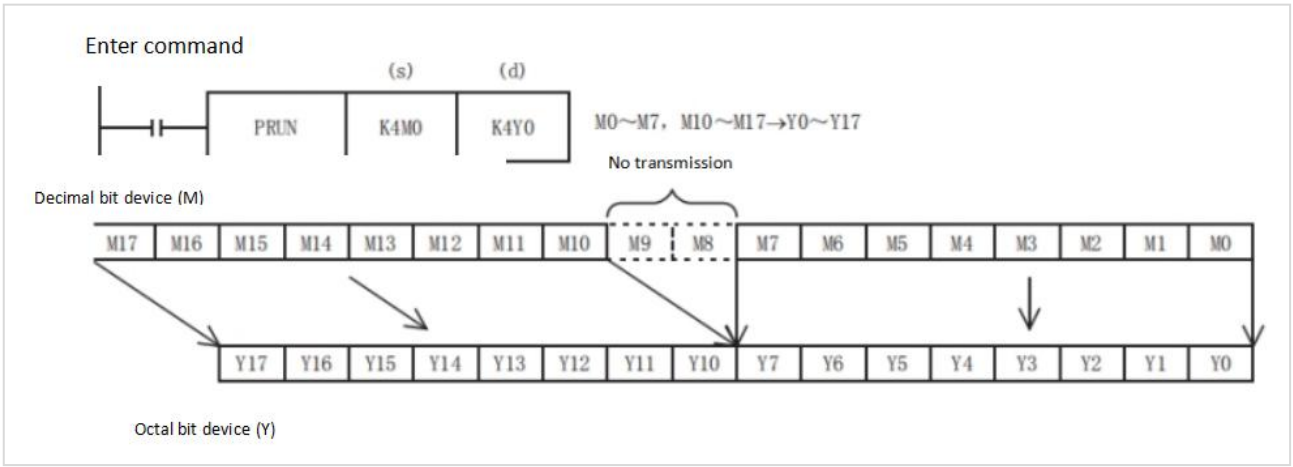
Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	Kn X	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PRUN	Parameter 1											•		•											•	•
	Parameter 2												•	•											•	•

Function

• Octal digit device→ decimal digit device



• Decimal digit device → octal digit device



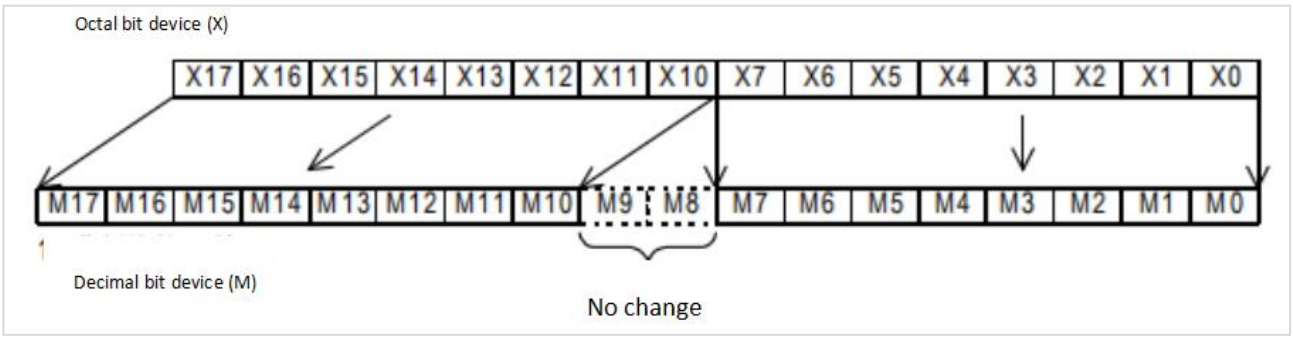
Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



As shown in the above ladder diagram: X0~X17 takes the value of octal digits and pass it to the devices corresponding to M.



6.8.16 DPRUN/Otal Digit Transmission (32-bit data)

DPRUN(P)

After processing the device numbers of (s) and (d) with specified digits as octal numbers, transfer the data.

-[PRUN (s) (d)]

Content, range and data type

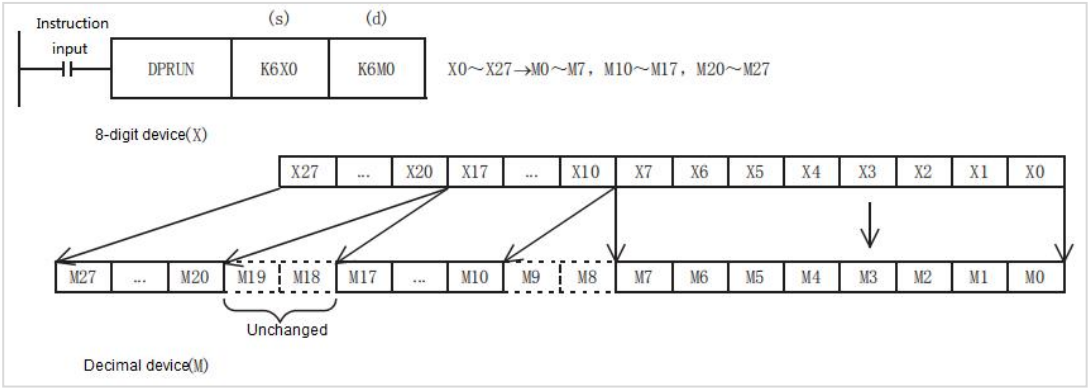
Parameter	Content	Range	Data type	Data type (label)
(s)	Digit specification*1	-	BIN32 bit	ANY32
(d)	Transfer destination device number*1	-	BIN32 bit	ANY32

Device used

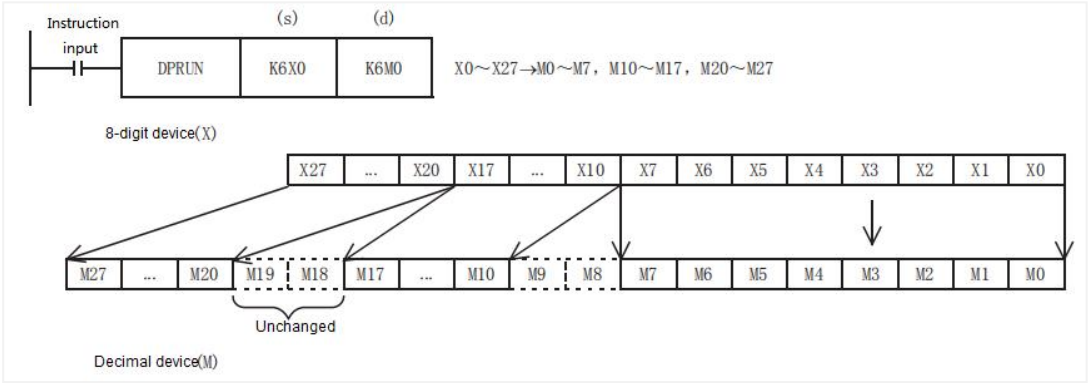
instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DPRUN	Parameter 1											●		●												●	●
	Parameter 2												●	●												●	●

Features

• Octal digit device to decimal digit device



• Decimal digit device → octal digit device



Error code

Error code	Content
4085H	When the specified device range is read to exceed the corresponding device range
4086H	When the specified device range is written to exceed the corresponding device range

Example



As shown in the above ladder diagram:

X0 to X27 take the value of octal digits and pass them to the Devices corresponding to M.

6.9 Floating point instructions

6.9.1 DACOS/Single precision real number COS-1 operation

DACOS(P)

After calculating the COS $^{-1}$ (arc cosine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DACOS (s) (d)]

Content, range and data type

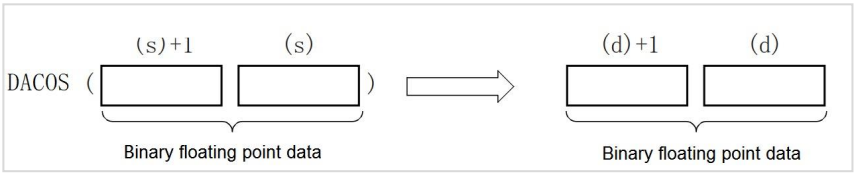
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for COS $^{-1}$ (arc cosine) calculation or the start device number that stores the angle data	0, $2^{-126} \leq \|(s)\| < 1$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	0 to π	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DACOS	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

After calculating the COS $^{-1}$ (arc cosine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).



The COS value specified in (s) can be set within the range of -1.0 to 1.0.

The angle (calculation result) stored in (d) stores the value from 0 to π in radians.

Related device are as follows:

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolutevalue of operation result<2 $^{-126}$	The value of (d) becomes the minimum value of 32-bit real numbers (2 $^{-126}$), and the borrow flag (SM152) turns ON.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number, ±∞ and exceeds -1.0 to 1.0

Example



Calculate the arc cosine value of 0.4 and the result is 1.159279.



6.9.2 DASIN/Single precision real number SIN^{-1} operation

DASIN(P)

After calculating the SIN -1 (arc sine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DASIN (s) (d)]

Content, range and data type

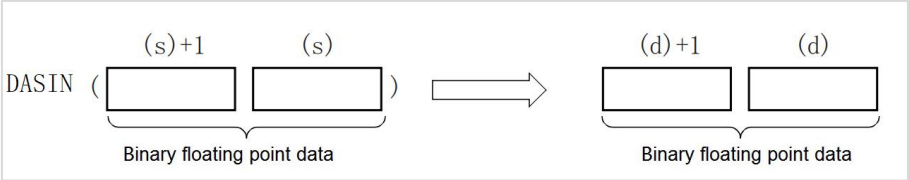
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for SIN $^{-1}$ (arcsine) calculation or the start device number that stores the angle data	0, $2^{-126} \le \|(s)\| < 1$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	-π/2 to π/2	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DASIN	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

• After calculating the SIN-1 (arc sine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).



The SIN^{-1} value specified in (s) can be set within the range of -1.0 to 1.0.

The angle (calculation result) stored in (d) is stored in the unit of radians (-π/2) to (π/2).

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result<2 $^{-126}$	The value of (d) becomes the minimum value of 32-bit real numbers (2 $^{-126)}$, and the borrow flag (SM152) turns ON.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number, ±∞ and exceeds -1.0 to 1.0

Example



Calculate the arc sine of 0.4 and the result is 0.4115168.



6.9.3 DATAN/Single precision real number TAN^{-1} operation

DATAN(P)

After calculating the TAN -1 (arctangent) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DATAN (s) (d)]

Content, range and data type

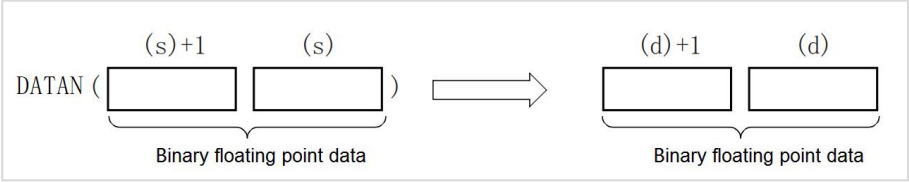
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for TAN $^{-1}$ (arctangent) calculation or the start device number that stores the angle data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	-π/2 to π/2	Single precision real number	ANYREAL_32

Device use

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DATAN	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

Calculate the TAN -1 ((arctangent) value of the angle specified in (s), and store the calculation result in the device number specified in (d).



The angle (calculation result) stored in (d) is stored in the unit of radians (-π/2) to (π/2).

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result<2 $^{-126}$	The value of (d) becomes the minimum value of 32-bit real numbers (2 $^{-126}$), and the borrow flag (SM152) turns ON.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Calculate the arctangent value of 4.6 and the result is 1.356736



6.9.4 DCOS/Single precision real number COS operation

DCOS (P)

After calculating the COS (cosine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DCOS (s) (d)]

Content, range and data type

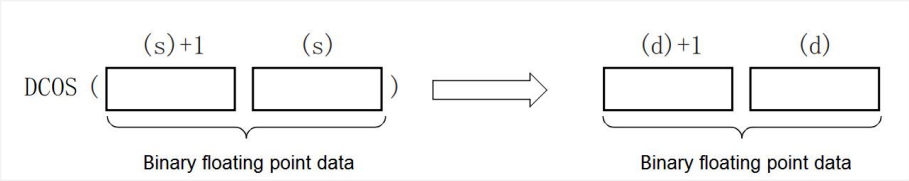
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for COS (cosine) calculation or the start device number that stores the angle data	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DCOS	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

After calculating the COS (cosine) value of the angle specified in (s), store the calculation result in the device number specified in (d).



For the angle specified in (s), set it in radians (angle×π÷180).

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result<2^{-126}	The value of (d) becomes the minimum value of 32-bit real numbers (2^{-126}), and the borrow flag (SM152) turns ON.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Calculate the cosine value of 1.3 and the result is 2.674989E-1



6.9.5 DCOSH/Single precision real number COSH operation

DCOSH(P)

After calculating the DCOSH (hyperbolic cosine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DCOSH (s) (d)]

Content, range and data type

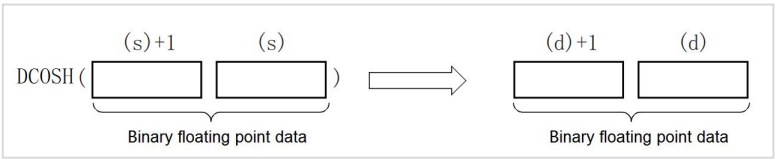
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for DCOSH (hyperbolic cosine) calculation or the device start number that stores the angle data	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	-	Single precision real number	ANYREAL_32

Device used

Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DCOSH	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

• After calculating the DCOSH (hyperbolic cosine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).



This instruction is to take the COSH value of a binary floating point number. The calculation formula is cosh value=(e s +e -s)/2.

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM151	carry	The absolute value of the operation result>2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Calculate the hyperbolic cosine value of 2.5, and the result is 6.132289



6.9.6 DSIN/Single precision real number SIN operation

DSIN(P)

After calculating the SIN (sine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DSIN (s) (d)]

Content, range and data type

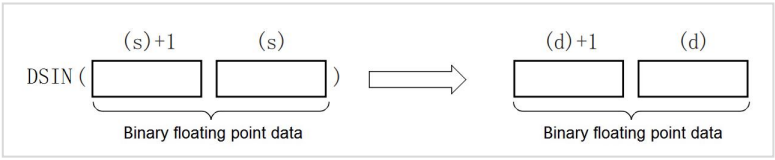
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for SIN (sine) calculation or the device start number that stores the angle data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSIN	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

After calculating the SIN (sine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).



For the angle specified in (s), set it in radians (angle×π÷180).

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2 -126	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Calculate the sine of 1.4 and the result is 0.9854497



6.9.7 DSINH/Single precision real number SINH operation

DSINH(P)

After calculating the SINH (hyperbolic sine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DSINH (s) (d)]

Content, range and data type

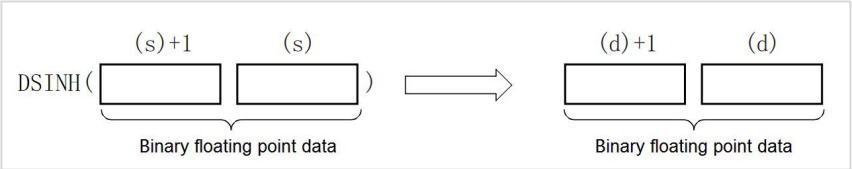
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for SINH (hyperbolic sine) calculation or the device start number that stores the angle data	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSINH	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

After calculating the SINH (hyperbolic sine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).



The instruction is to take the SINH value from a binary floating point number. The calculation formula is sinh value = (e s -e -s)/2.

The related devices are shown below.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result > 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Calculate the hyperbolic sine value of 3.2 and the result is 12.24588



6.9.8 DTAN/Single precision real number TAN operation

DTAN(P)

After calculating the TAN (tangent) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DTAN (s) (d)]

Content, range and data type

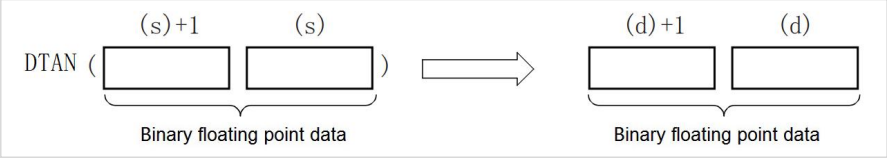
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for TAN (tangent) calculation or the device start number that stores the angle data	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DTAN	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

After calculating the TAN (tangent) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).



For the angle specified in (s), set it in radians (angle×π÷180).

The related devices are shown below.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Calculate the tangent of 1.4 and the result is 5.797883



6.9.9 DATANH/Single precision real number TANH operation

DTANH(P)

After calculating the DTANH (hyperbolic tangent) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).

-[DTANH (s) (d)]

Content, range and data type

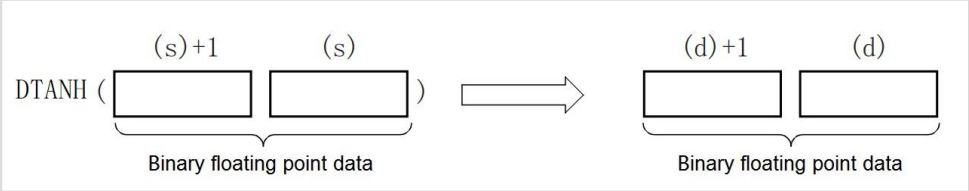
Parameter	Content	Range	Data type	Data type (label)
(s)	The angle data for DTANH (hyperbolic tangent) calculation or the device start number that stores the angle data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start device number that stores operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DTANH	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

After calculating the DTANH (hyperbolic tangent) value of the angle specified in (s), the calculation result is stored in the device number specified in (d).



The instruction is to take the TANH value of a binary floating point number. The calculation formula is tanh value=$(e^{s}-e^{-s})/(e^{s}+e^{-s})$.

The related devices are shown below.

Device	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Calculate the hyperbolic tangent of 2.5, and the result is 0.9866143



6.9.10 DDEG/Single precision real number radian → Angle conversion

DDEG(P)

Convert the size unit of the angle from the radian unit specified in (s) to the degree unit (DEG. unit), and store it in the device number specified in (d).

-[DDEG (s) (d)]

Content, range and data type

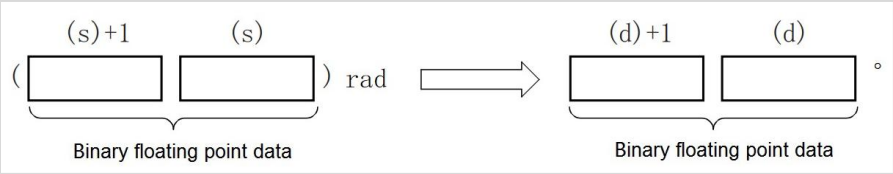
Parameter	Content	Range	Data type	Data type (label)
(s)	The radian angle that converts the degree unit or the device start number that stores the radian angle	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The device start number that stores the value converted in degrees	-π/2 to π/2	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DDEG	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

The angle size unit is converted from the radian unit specified in (s) to the degree unit (DEG. unit), and then stored in the device number specified in (d).



The conversion from degree unit to radian unit is performed as follows.

Radian unit = degree unit * 180/π

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result of is zero (when the mantissa part is zero)	The zero flag (SM153) turns ON.
SM151	Carry	The absolute value of the operation result>2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



The result is 194.8057



6.9.11 DRAD/Single precision real number conversion angle → Radian conversion

DRAD(P)

The angle size unit is converted from the degree unit (DEG. unit) specified in (s) to the radian unit and stored in the device number specified in (d).

-[DRAD (s) (d)]

Content, range and data type

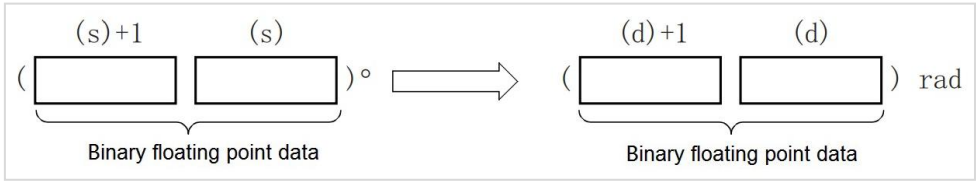
Parameter	Content	Range	Data type	Data type (label)
(s)	The radian angle that converts the degree unit or the device start number that stores the angle	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The device start number that stores the value converted in degrees	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DRAD	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	

Features

The angle size unit is converted from the degree unit (DEG. unit) specified in (s) to the radian unit and stored in the device number specified in (d).



Degree unit → radian unit

The conversion is performed as follows.

Radian unit = degree unit*π/180

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



The result is 1.047197



6.9.12 DEADD/Single precision real number addition operation

DEADD(P)

Add the binary floating point data specified in (s1) and the binary floating point data specified in (s2), and store the result in the device specified in (d).

-[DEADD (s1) (s2) (d)]

Content, range and data type

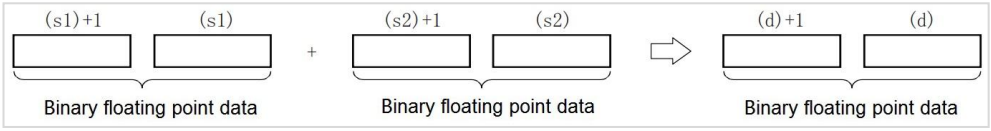
Parameter	Content	Range	Data type	Data type (label)
(s1)	The added data or the device start number that stores the added data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s2)	Addition data or the device start number that stores the addition data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	the device start number that stores the operation result	-	Single precision real number	ANYREAL_32

Device used

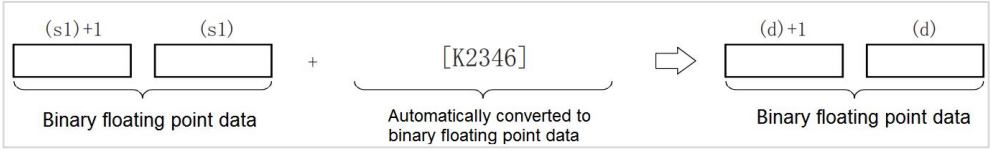
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEADD	Parameter 1															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 2															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 3															●	●	●	●	●	●	●				●	●

Features

Add the binary floating point data specified in (s1) and the binary floating point data specified in (s2), and store the result of the addition in the device specified in (d).



When constants (K, H) to (s1), (s2) are specified, the value is automatically converted to a binary floating point data.



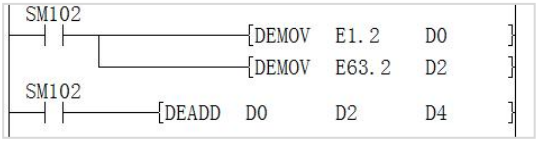
• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s1) and (s2) exceed the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s1) and (s2) is an irregular number, a non-number and ±∞

Example



The result is 1.2 + 63.2 = 64.4

D0	1.2	6.32E+1	6.44E+1

6.9.13 DESUB/Single Precision real number subtraction operation

DESUB(P)

Subtract the binary floating point data specified in (s1) and the binary floating point data specified in (s2), and store the result in the device specified in (d).

-[DESUB (s1) (s2) (d)]

Content, range and data type

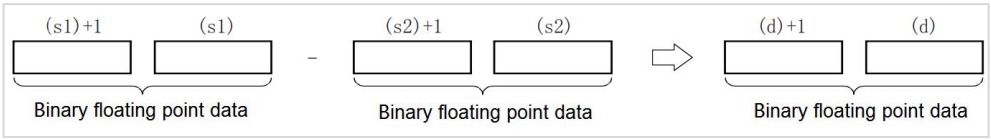
Parameter	Content	Range	Data type	Data type (label)
(s1)	The subtracted data or the device start number that stores the subtracted data	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s2)	subtract data or the device start number that stores the subtracted data	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	the device start number that stores the operation result	-	Single precision real number	ANYREAL_32

Device used

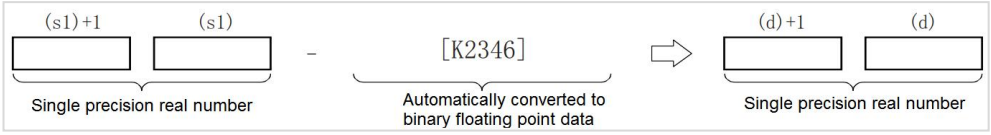
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DESUB	Parameter 1															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 2															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 3															●	●	●	●	●	●	●				●	●

Features

• Subtract the binary floating point data specified in (s1) and the binary floating point data specified in (s2), and store the subtraction result in the device specified in (d).



When constants (K, H) to (s1), (s2) are specified, the value is automatically converted to a binary floating point data.



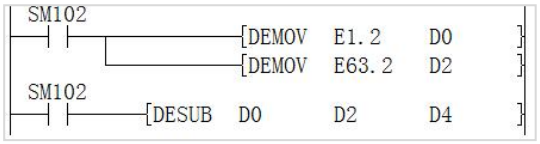
• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result $<2^{-126}$	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s1) and (s2) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s1) and (s2) is an irregular number, a non-number and ±∞

Example



The calculation result is 1.2-63.2 = -62

D0	1.2	6.32E+1	-6.2E+1

6.9.14 DEMUL/Single precision real number multiplication operation

DEMUL(P)

Multiply the binary floating point data specified in (s1) and the binary floating point data specified in (s2), and store the result in the device specified in (d).

-[DEMUL (s1) (s2) (d)]

Content, range and data type

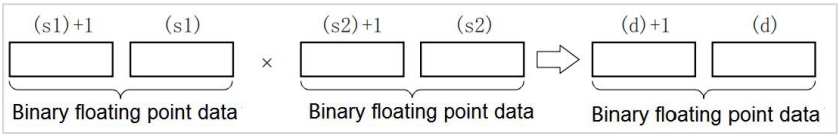
Parameter	Content	Range	Data type	Data type (label)
(s1)	The multiplication data or the device start number that stores the multiplication data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s2)	Multiplication operation data or the device start number that stores the multiplication data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	the device start number that stores the operation result	-	Single precision real number	ANYREAL_32

Device used

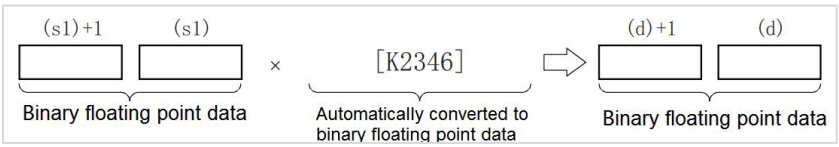
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEMUL	Parameter 1															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 2															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 3															●	●	●	●	●	●	●				●	●

Features

Multiply the binary floating point data specified in (s1) and the binary floating point data specified in (s2), and store the multiplication result in the device specified in (d).



When constants (K, H) to (s1), (s2) are specified, the value is automatically converted to a binary floating point data.



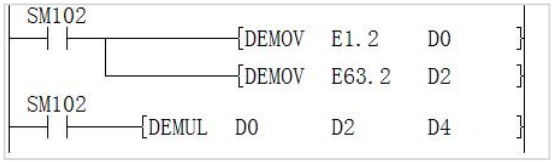
• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s1) and (s2) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s1) and (s2) is an irregular number, a non-number and ±∞

Example



The calculated result: 1.2*63.2 = 75.84

Device	+0	+2	+4	+6
D0	1.200000E+000	6.320000E+001	7.584000E+001	0.000000E+000

6.9.15 DEDIV/Single precision real number division operation

DEDIV(P)

Divide the binary floating point data specified in (s1) and the binary floating point data specified in (s2), and store the result in the device specified in (d).

-[DEDIV (s1) (s2) (d)]

Content, range and data type

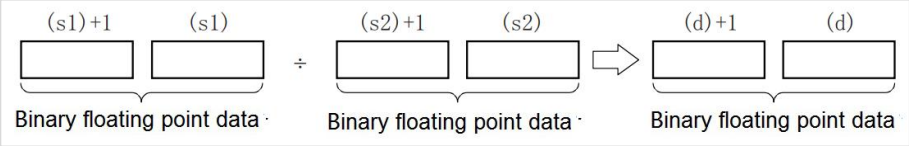
Parameter	Content	Range	Data type	Data type (label)
(s1)	The divided data or the device start number that stores the devided data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s2)	Division operation data or the device start number that stores the division operation data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	the device start number that stores the operation result	-	Single precision real number	ANYREAL_32

Device used

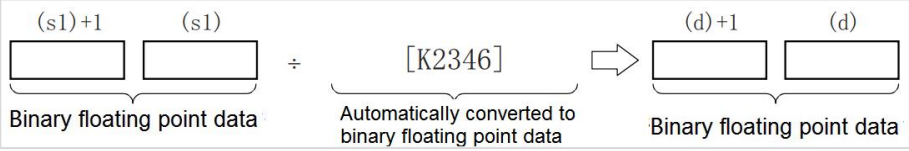
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEDIV	Parameter 1															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 2															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 3															●	●	●	●	●	●	●				●	●

Features

Divide the binary floating point data specified in (s1) and the binary floating point data specified in (s2), and store the result of the division in the device specified in (d).



When constants (K, H) to (s1), (s2) are specified, the value is automatically converted to a binary floating point data.



• The related devices are as follows.

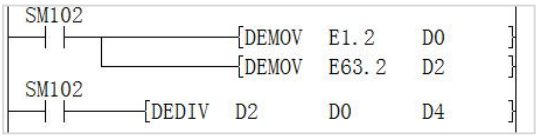
Devices	Name	Content	
		Condition	Operating
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s1) and (s2) exceeds the device range
4086H	The write address in (d) exceeds the device range

Error code	Content
4084H	When the content of the device specified by (s1) and (s2) is an irregular number, a non-number and ±∞
4080H	(s2) value is 0

Example



Get the calculation result: 63.2 / 1.2 = 52.66666667

D0	1.2	6.32E+1	5.266666E+1

6.9.16 DEMOV/Single precision real data transmission

DEMOV(P)

Transfer the binary floating point data data stored in the device specified in (s) to the device specified in (d).

-[DEMOV (s) (d)]

Content, range and data type

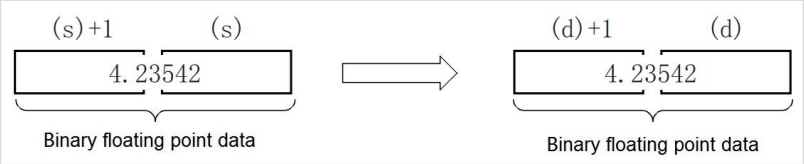
Parameter	Content	Range	Data type	Data type (label)
(s)	The transmitted data or the device that stores the transmitted data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The device number that stores the transmit destination data	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEMOV	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

Transfer the binary floating point data data stored in the device specified in (s) to the device specified in (d).



Error code

Error code	Content
4085H	(s) read address exceeds the device range
4086H	(d) write address exceeds the device range

Example



Assign 3.265 to R10

Device	+0	+1
R8	0.000000E+000	3.256

6.9.17 DEBCD/Binary floating point → Decimal floating point conversion

DEBCD(P)

After converting the binary floating point specified in (s) into a decimal floating point, it is stored in the device specified in (d).

-[DEBCD (s) (d)]

Content, range and data type

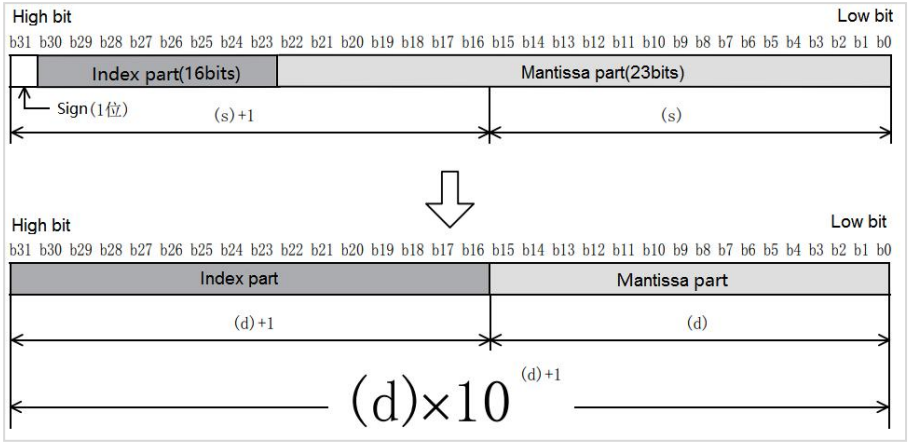
Parameter	Content	Range	Data type	Data type (label)
(s)	The device number that stores the binary floating point data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The device number that stores the converted decimal floating point data	-	Real number	ANY32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEBCD	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

After converting the binary floating point specified in (s) into a decimal floating point, it is stored in the device specified in (d).



✎Note

All floating-point operations are performed in binary floating-point. However, the binary floating point is a difficult-to-understand value (special monitoring method), so by converting it into a decimal floating point operation, it is convenient for peripheral equipment to monitor and so on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Get the result: 5600 ×10 -5

D0	0	0	0	0	0	1	1	1	1	0	1	0	1	0	0	0	5600
D1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	-5

6.9.18 DEBIN/Decimal floating point → Binary floating point conversion

DEBIN(P)

Convert the decimal floating point specified in (s) to binary floating point and store it in the device specified in (d).

-[DEBIN (s) (d)]

Content, range and data type

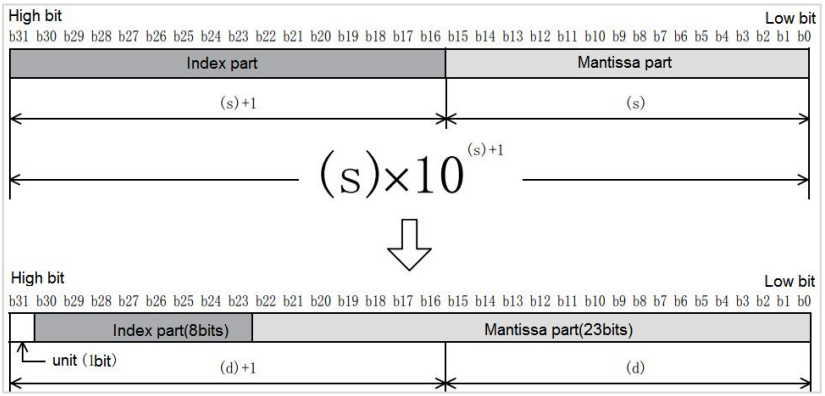
Parameter	Content	Range	Data type	Data type (label)
(s)	The device number that stores the decimal floating point data	-	Real	ANY32
(d)	The device number that stores the converted binary floating point data	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEBIN	Parameter 1															●	●	●	●	●	●	●				●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

Convert the decimal floating point specified in (s) to binary floating point and store it in the device specified in (d).



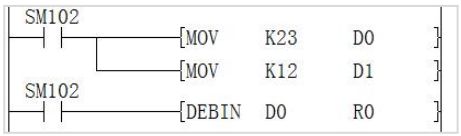
• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	(s) read address exceeds the device range
4086H	(d) write address exceeds the device range

Example



The result after conversion:



6.9.19 DENEG/Single precision real number sign inversion

DENEG(P)

After inverting the sign of the single precision real number of the device specified in (d), it is stored in the device specified in (d).

-[DEBEG (d)]

Content, range and data type

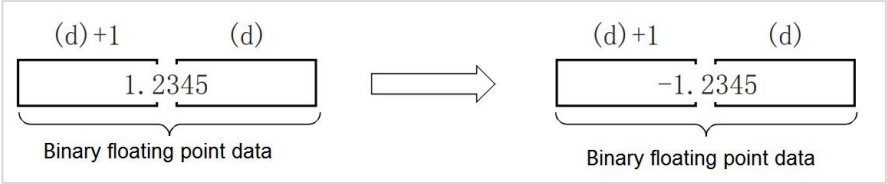
Parameter	Content	Range	Data type	Data type (label)
(d)	The device start number that stores the sign-inverted binary floating point data	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DENEG	Parameter 1															●	●	●	●	●	●	●				●	●

Features

The sign of the binary floating point data of the device specified in (d) is inverted and stored in the device specified in (d).

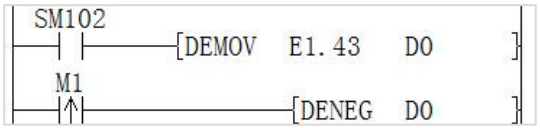


Used when inverting positive and negative signs.

Error code

Error code	Content
4086H	The write address in (d) exceeds the device range

Example



It becomes -1.43 after conversion



6.9.20 DECMP/Single precision real number comparison

DECMP(P)

Compare two data (binary floating point data), and output their large, small, and consistent results to the bit device (3 points).

-[DECMP (s1) (s2) (d)]

Content, range and data type

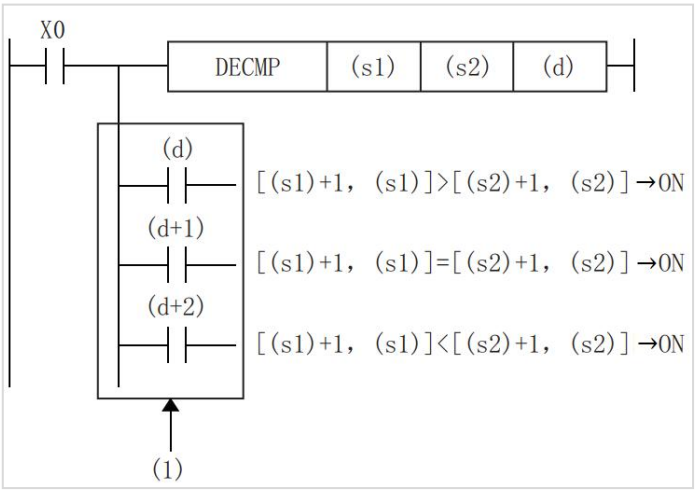
Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparison data or the device number that stores the comparison data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s2)	Comparison data or the device number that stores the comparison data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start bit device number that outputs the comparison result (occupies 3 points)	-	Bit	ANYBIT_ARRAY

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DECMP	Parameter 1															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 2															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 3		●	●	●	●					●															●	●

Features

Compare the comparison value (s1) and the comparison source (s2) as a floating point comparison. According to the result of small, consistent, and large, one of (d), (d)+1, (d)+2 turns ON.



(1) : Even if the instruction input is turned OFF and the DECMP instruction is not executed, (d) to (d)+2 will keep the state before X0 is turned OFF.

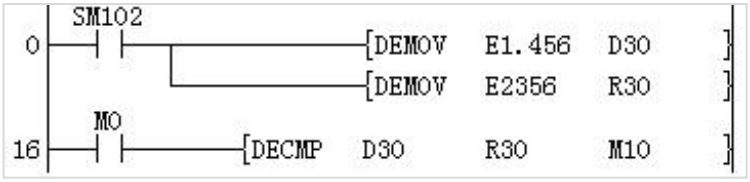
When the constant (K, H) to the device specified in (s1), (s2) is specified, the value BIN→binary floating point data conversion is processed automatically

✎**Note:** The device specified in (d) occupies 3 points [(d), (d)+1, (d)+2]. Please be careful not to overlap with devices used for other purposes.

Error code

Error code	Content
4085H	The write address in (s1) or (s2) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) or (s2) is an irregular number, a non-number and ±∞

Example



Since the floating point number in R30 is greater than the floating point number in D30, M12 turns ON.

M10	0
M11	0
M12	1

6.9.21 DEZCP/Binary floating point bandwidth comparison

DEZCP(P)

Compare the comparison range and data (binary floating point) of high and low 2 points, and output the result of its large, small, and bandwidth to the bit device (3 points).

-[DEZCP (s1) (s2) (s3) (d)]

Content, range and data type

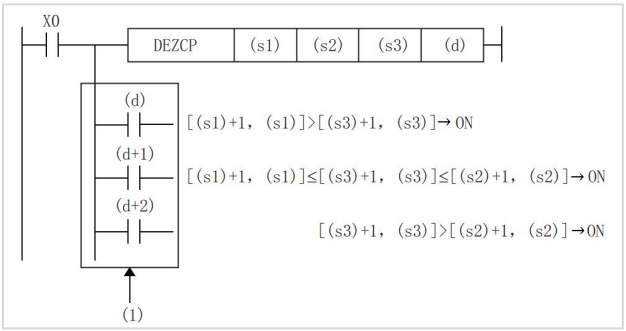
Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparison data or the device number that stores the comparison data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s2)	Comparison data or the device number that stores the comparison data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s3)	Comparison data or the device number that stores the comparison data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The start bit device number that outputs the comparison result (occupies 3 points)	-	Bit	ANYBIT_ARRAY

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEZCP	Parameter 1															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 2															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 3															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 4		●	●	●	●					●															●	●

Features

Compare the comparison value (s1), (s2) and the comparison source (s3) as a floating point comparison, according to its small, range, and large result, one of (d), (d)+1, (d)+2 The bit turns ON.



(1): Even if the instruction input is turned OFF and DEZCP instruction is not executed, (d) to (d)+2 will keep the state before X0 is turned OFF.

When the constant (K, H) to the device specified in (s1), (s2), (s3) is specified, the value is automatically converted from BIN to binary floating point for processing.

✎Note

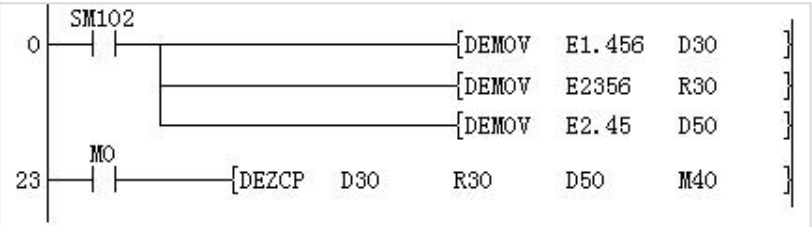
The device specified in (d) occupies 3 points [(d), (d)+1, (d)+2]. Please be careful not to overlap with devices used for other purposes.

Please set the size relationship of the comparison data as [(s1)+1,(s1)]≤[(s2)+1,(s2)]. In the case of [(s1)+1,(s1)]>[(s2)+1,(s2)], it is regarded as the value of [(s2)+1,(s2)] and [(s1)+1, (s1)] Same for comparison.

Error code

Error code	Content
4085H	The write address in (s1), (s2) and (s3) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s1), (s2) and (s3) is an irregular number, a non-number and ±∞

Example



Since 2.45 is greater than 1.456 and 2.45 is less than 2356, M41 is set to ON

M40	0
M41	1
M42	0

6.9.22 DESQR/Single precision real square root

DESQR(P)

After the square root of the value specified in (s) is calculated, the calculation result is stored in the device specified in (d).

-[DESQR (s) (d)]

Content, range and data type

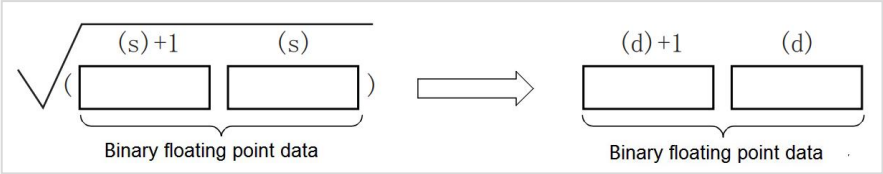
Parameter	Content	Range	Data type	Data type (label)
(s)	The data for square root operation or the device start number that stores the data	0, $2^{-126} \le \lvert(s)\rvert < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The device start number stores operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DESQR	Parameter 1															●	●	●	●	●	●	●	●	●	●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

• After the square root of the value specified in (s) is calculated, the calculation result is stored in the device number specified in (d).



The value specified in (s) can only be set to a positive number. (Cannot perform operations with negative numbers.)

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Get the result: D0 is a floating point number 2

Device	+0	+1
D0	2	0.000000E+000

6.9.23 DESTR/Single precision real number → String conversion

DESTR(P)

Convert the binary floating point data data stored in the device specified in (s1) into a character string according to the display specification stored after the device number specified in (s2), and store it in the device number specified in (d) or later .

-[DESTR (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Converted single precision real number data or the device start number that stores the data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s2)	Display the specified device start number that stores the converted value. The device specified in (s1) is used as the start, and (s2)+2 is used	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(d)	Start number of the device storing the converted character string	-	String	ANYSTRING_SINGLE

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DESTR	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2											●	●	●	●	●	●	●	●	●						●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

Features

Convert the binary floating point data data stored in the device specified in (s1) into a character string according to the display specification stored after the device number specified in (s2), and store it in the device number specified in (d) after. You can also directly specify the real number to (s1).

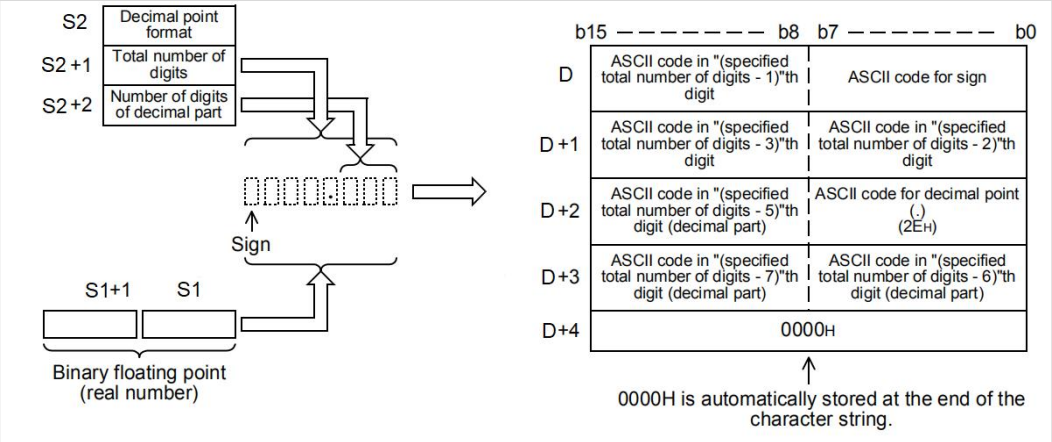
• The converted data differs according to the display specification specified in (s2).

Unit	Features
(s2)	0: Decimal point form 1: Exponential form
(s2)+1	All digits (total number of strings). Range: 2 to 24
(s2)+2	The number of decimal digits. Range: 0 to 7

The range in the above table will change the value range according to the conversion form and other information used

Decimal form

If 0 is specified in (s2), it will be in decimal form.

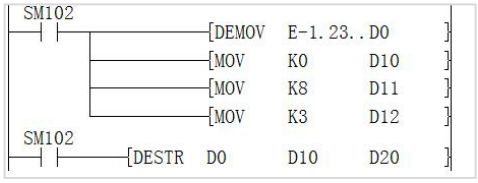


Corresponding digit range in decimal form:

Unit	Features
(s2)	0: Decimal point form
(s2)+1	All digits (total number of strings). Range: 2 to 24. When (s2)+2 is not 0: digits ≥ (number of decimal places + 3).
(s2)+2	The number of decimal places. Range 0 to 7, When (s2)+2 is not 0: digits ≥ (number of decimal places + 3).

Example: The total number of digits is 8, the number of decimal places is 3, and when -1.235 is specified, (d) will be stored in the following way.

When displaying character strings, display character strings in normal order from left to right for convenience.



Converted string

D20	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	
D21	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	1
D22	0	1	1	1	0	1	0	0	0	1	0	0	1	1	0	0	.2
D23	1	1	0	0	1	1	0	0	1	0	1	0	1	1	0	0	35

The corresponding ASCII code is:

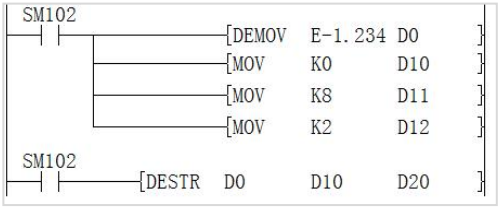
D20	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	2020	Automatically added Spaces
D21	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	3120	31H(1) 20H (blank)
D22	0	1	1	1	0	1	0	0	0	1	0	0	1	1	0	0	322E	32H(2) 2EH(.)
D23	1	1	0	0	1	1	0	0	1	0	1	0	1	1	0	0	3533	35H(5) 33H(3)

The first one is the sign bit. In the sign, when binary floating point data data is positive, 20H (blank) is stored, and when it is negative, 2DH(-) is stored.

If the actual number of digits is less than all digits during conversion, 20H (blank) will be added between the sign and the first number

If the decimal part of the binary floating point data data cannot be accommodated in the decimal part, the lower decimal part will be rounded off.

2. Example: The total number of digits is 8, the number of decimal places is 2, and when -1.234 is specified, (d) will be stored in the following way.



The converted string:

D20	1	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	-
D21	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	
D22	1	0	0	0	1	1	0	0	0	1	1	1	0	1	0	0	1.
D23	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	23

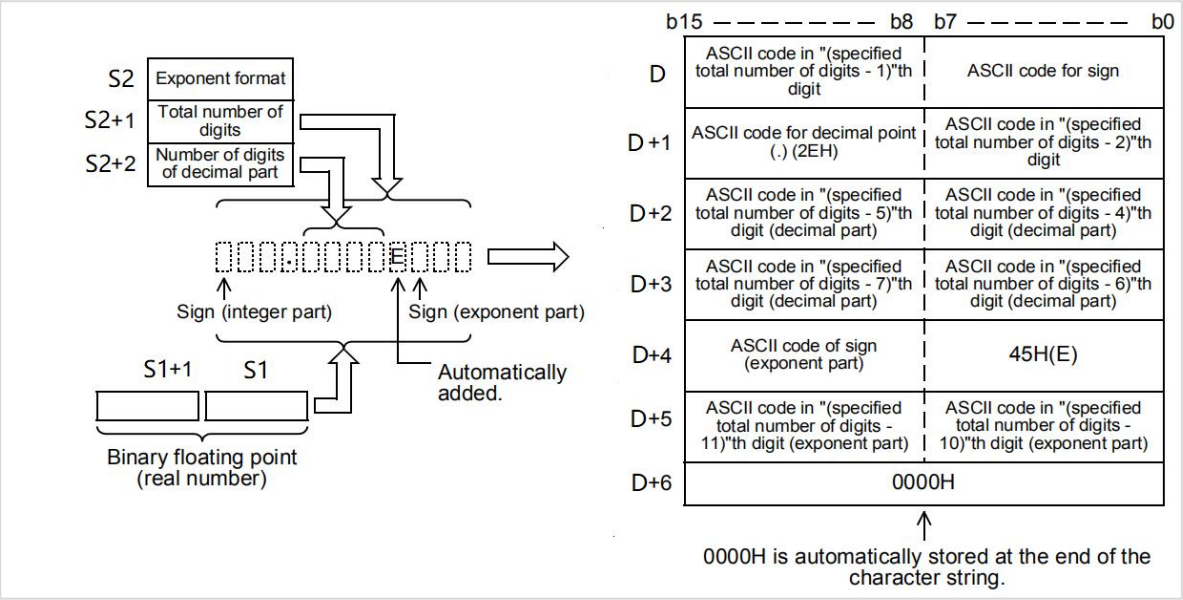
The corresponding ASCII code is:

D20	1	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	202D
D21	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	2020
D22	1	0	0	0	1	1	0	0	0	1	1	1	0	1	0	0	2E31
D23	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	3332
D24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

In the above example: the low byte of D20 stores the negative sign 2DH(-). Then due to insufficient number of digits, the high byte of D20 and D21 are both 20H (blank). Finally, D22 to D23 store numeric characters 1.23

Exponential form

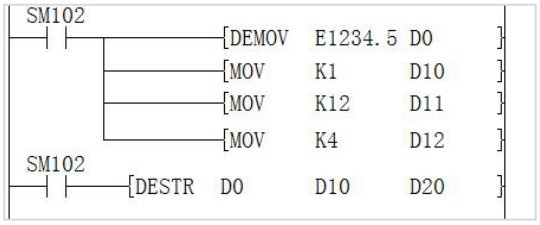
When 1 is specified in (s2), it will be in exponential format.



The corresponding digit range in exponential form:

Unit	Features
(s2)	1: Exponential form
(s2)+1	All digits (total number of strings). Range: 2 to 24. (s2)+2 when non-zero: digits ≥ (number of decimal places + 7)
(s2)+2	The number of decimal places. Range 0 to 7 (s2)+2 when non-zero: digits ≥ (number of decimal places + 7)

For example 3, all digits are 12, decimal place is 4, and 1234.5 is specified, (d) and later will be stored as follows.



The converted string:

D20	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	
D21	1	0	0	0	1	1	0	0	0	1	1	1	0	1	0	0	1.
D22	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	23
D23	0	0	1	0	1	1	0	0	1	0	1	0	1	1	0	0	45
D24	1	0	1	0	0	0	1	0	1	1	0	1	0	1	0	0	E+
D25	0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	03
D26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	..

The corresponding ASCII code is:

D20	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	2020
D21	1	0	0	0	1	1	0	0	0	1	1	1	0	1	0	0	2E31
D22	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	3332
D23	0	0	1	0	1	1	0	0	1	0	1	0	1	1	0	0	3534
D24	1	0	1	0	0	0	1	0	1	1	0	1	0	1	0	0	2B45
D25	0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	3330
D26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

In the sign of the integer part, when the binary floating point data data is positive, 20H (blank) is stored, and when it is negative,

2DH(-) is stored.

The integer part is fixed to 1 digit. 20H (blank) is stored between the integer part and the Sign.

If the decimal part of the binary floating point data data cannot be accommodated in the decimal part, the lower decimal part will be rounded off.

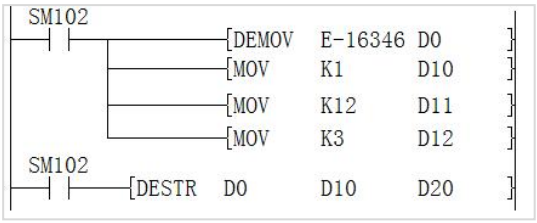
When the number of decimal places is set to other than 0, 2EH(.) is automatically stored in the number of specified decimal places+1 digit. When the decimal place is 0, 2EH(.) is not stored.

In the sign of the exponent, 2BH(+) is stored when the exponent is positive, and 2DH(-) is stored when it is negative.

The exponent is fixed to 2 digits. When the exponent part is a 1-digit number, 30H(0) is stored between the signs of the exponent part.

00H is automatically stored at the end of the converted character string.

Example 4: All digits are 12, decimal places are 3, and -16346 is specified, (d) will be stored in the following way.



The converted string:

D20	1	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	-
D21	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	1
D22	0	1	1	1	0	1	0	0	0	1	1	0	1	1	0	0	.6
D23	1	1	0	0	1	1	0	0	1	0	1	0	1	1	0	0	35
D24	1	0	1	0	0	0	1	0	1	1	0	1	0	1	0	0	E+
D25	0	0	0	0	1	1	0	0	0	0	1	0	1	1	0	0	04
D26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	..

The corresponding ASCII code is:

D20	1	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	202D
D21	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	3120
D22	0	1	1	1	0	1	0	0	0	1	1	0	1	1	0	0	362E
D23	1	1	0	0	1	1	0	0	1	0	1	0	1	1	0	0	3533
D24	1	0	1	0	0	0	1	0	1	1	0	1	0	1	0	0	2B45
D25	0	0	0	0	1	1	0	0	0	0	1	0	1	1	0	0	3430
D26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

As in the above example:

The low byte of D20 stores the negative sign 2DH(-).

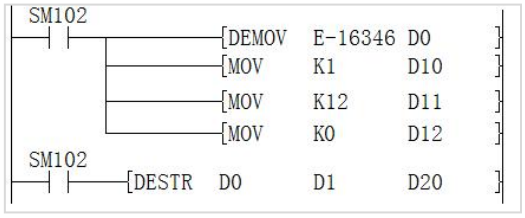
Then due to insufficient number of digits, the high byte of D20 and the low bit of D21 are both 20H (blank).

16346 becomes the string 1.635E+04, in which the last digit "6" of 16346 is rounded.

The exponent part is 34H(4) with only one bit, then add 30H(0) between the Signs 2DH(-) and 34H(4).

Finally D26 automatically stores 00H

Example 5: All digits are 12, and the number of decimal places is 0. If -16346 is specified, (d) will be stored as follows.



The converted string:

D20	1	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	-
D21	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	
D22	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	
D23	0	0	0	0	0	1	0	0	0	1	0	0	1	1	0	0	2
D24	1	0	1	0	0	0	1	0	1	1	0	1	0	1	0	0	E+
D25	0	0	0	0	1	1	0	0	0	0	1	0	1	1	0	0	04
D26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	..

The corresponding ASCII code is:

D20	1	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	202D
D21	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	2020
D22	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	2020
D23	0	0	0	0	0	1	0	0	0	1	0	0	1	1	0	0	3220
D24	1	0	1	0	0	0	1	0	1	1	0	1	0	1	0	0	2B45
D25	0	0	0	0	1	1	0	0	0	0	1	0	1	1	0	0	3430
D26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

This example mainly shows that if the decimal place is set to 0, the decimal point 2EH(.) will be automatically omitted.

✎Note

When the binary floating-point data is converted, the more digits, the lower the accuracy of the digits, the worse the accuracy of the digits, and the conversion value may be inaccurate due to the progress.

Error code

Error code	Content
4085H	The read address of (s1) and (s2) exceeds the device range
4086H	The write address of (d) exceeds the device range
4084H	When the content of the specified device (s1) and (s2) is an irregular number, a non-number, or ±∞
	When the format specified in (s2) is other than 0 or 1
	When all the digits specified in (s1) +1 exceeds the value of 24
	When the number of decimal places specified in (s2) +2 exceeds the range of 0 to 7
	In the decimal form, when (s2) is 0. ① When the number of decimal places is 0: [(s2)+1]<2 ② When the number of decimal places is other than 0: [(s2)+1]<(number of decimal places+3)
	In the exponential form, when (s2) is 0. ① When the number of decimal places is 0: [(s2)+1]<6 ② When the number of decimal places is other than 0: [(s2)+1]<(number of decimal places+7)

6.9.24 DEVAL/String → Single precision real number conversion

DEVAL(P)

The character string stored in the device number specified in (s) and later is converted to a binary floating point data, and then stored in the device specified in (d).

-[DEVAL (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	String data for single precision real number or the device start number that stores the string data	-	String	ANYSTRING_SINGLE
(d)	The device start number that stores the converted single precision real number	-	Single precision real number	ANYREAL_32

Device used

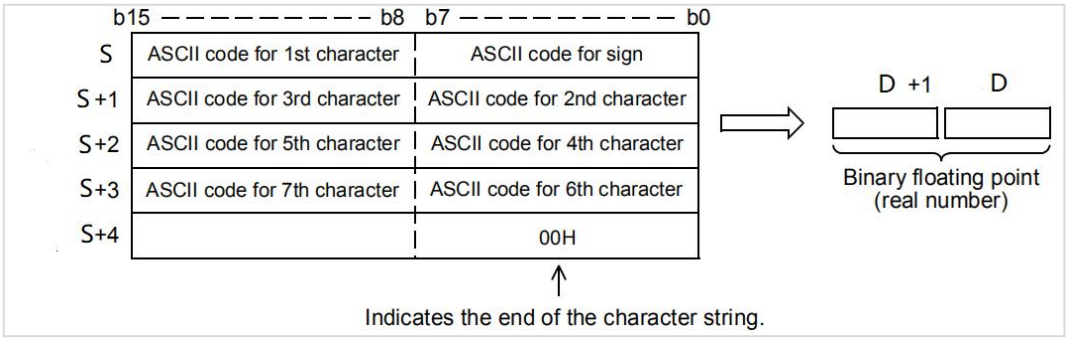
instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEVAL	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

The character string stored in the device number specified in (s) and later is converted to a binary floating point data, and then stored in the device specified in (d).

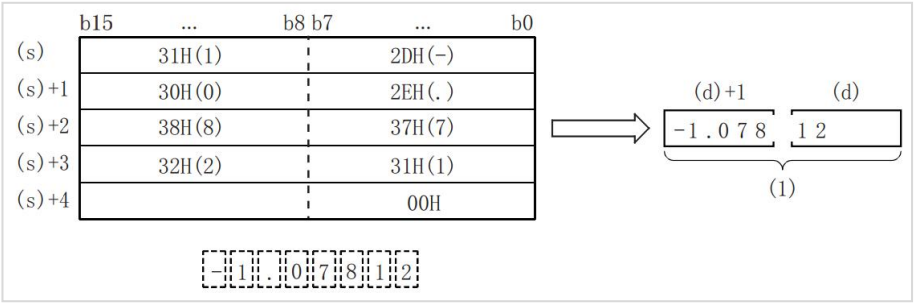
Whether the specified string is in decimal form or exponential form, it can be converted to a binary floating point data.

Up to 24 characters can be set for the string. 20H (blank) and 30H (0) in the character string are also counted as 1 character.



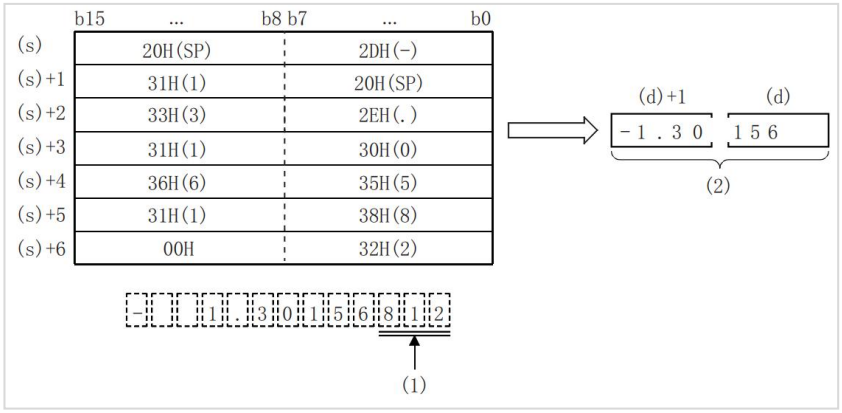
(1) Decimal form

1) When the character string specified in (s) is in decimal format, the following is the case.



(1): single-precision real number

2) In the character string specified in (s), for the character string to be converted to a binary floating point data, the 6 digits after the sign, decimal point, and exponent are valid, and the 7th digit and later will be discarded during conversion.

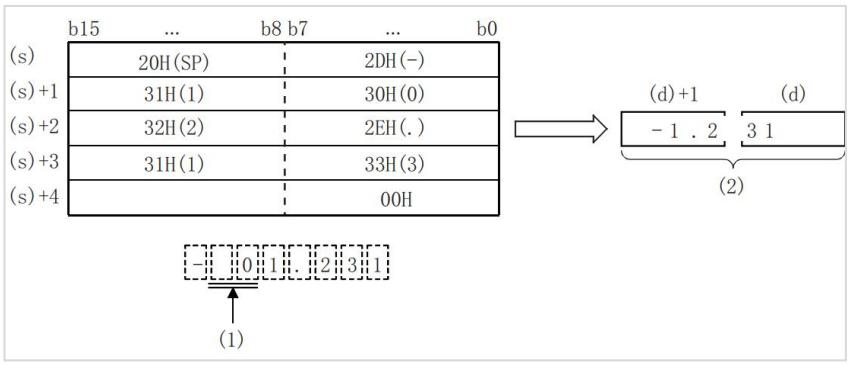


(1): discarded

(2): single-precision real number

When the sign is specified as 2BH(+) or omitted in the decimal point format, it will be converted as a positive value. In addition, when the sign is specified as 2DH(-), it will be converted as a negative value.

3) If there are 20H (blank) or 30H (0) in the character string specified in (s) other than the first 0, 20H and 30H will be ignored during conversion.

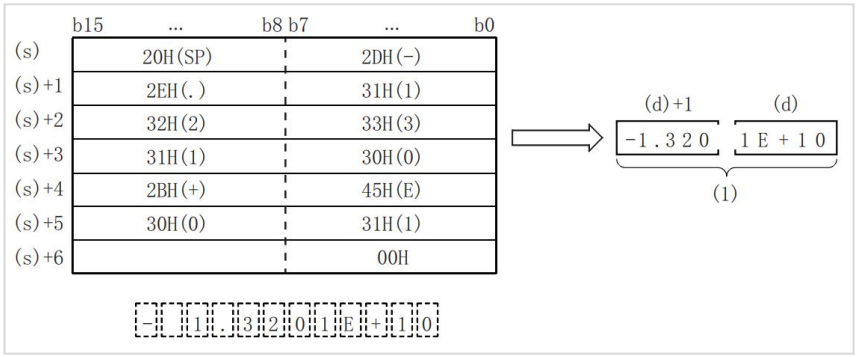


(1): ignore

(2): single-precision real number

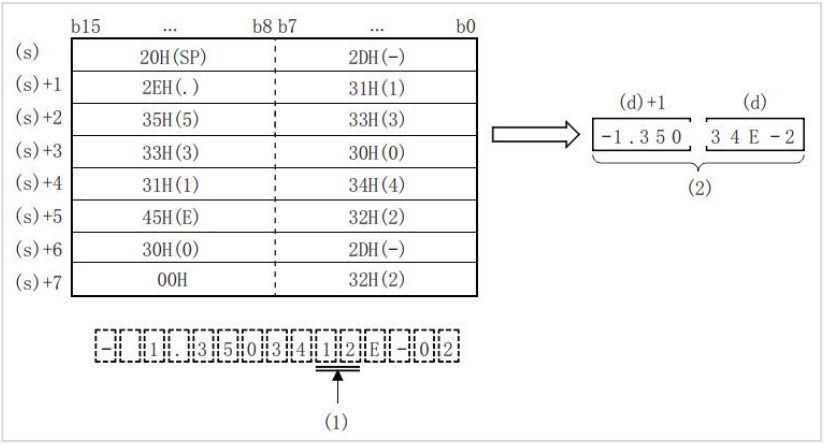
(2) In the case of exponential form

1) When the character string specified in (s) is in exponential form, it is executed as follows.



(1): single-precision real number

Among the character strings specified in (s), for the character string to be converted to a binary floating point data, the 6 digits after the sign, decimal point, and exponent are valid, and the 7th digit and later will be discarded during conversion.



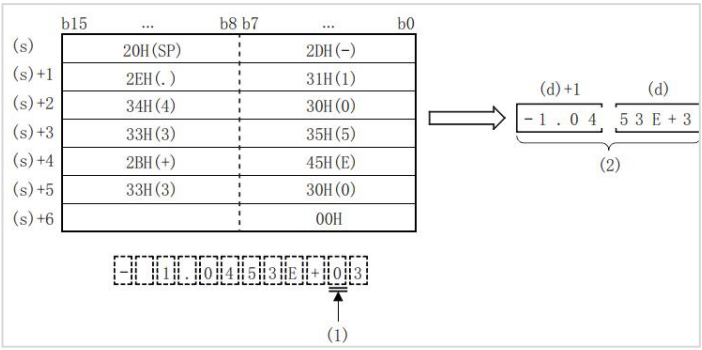
(1): discarded

(2): single-precision real number

If the sign of the exponent part is specified as 2BH(+) or omitted in the exponential form, it will be converted as a positive value. When the sign of the exponent is specified as 2DH(-), it will be converted as a negative value.

2) If there is 20H (blank) or 30H (0) in the character string specified in (s) other than the first 0, 20H and 30H will be ignored during conversion.

In the exponential character string, if 30H (0) is stored between "E" and the value, 30H will be ignored during conversion.



(1): Ignore.

(2): Single precision real number.

The related devices are shown below.

Devices	Name	Content	
		Condition	Operating
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The read address of (s) exceeds the device range
4086H	The write address of (d) exceeds the device range
408AH	The string is not read by (s), or the string length exceeds 24
408BH	When (s) reading a character string, the maximum range of the device is read, but 00H is not found and the end
4084H	When there are characters other than 2BH (+), 2DH (-), 20H (space), 2EH (.), 45H (E), 65H(e), and 30 H(0) to 39H (9) in the string specified in (s)
	When there are two or more 2EH(.) characters in the character string specified in (s).

Error code	Content
	When there are characters other than 45H(E), 2BH(+), 2DH(-), and 30 H(0) to 39H (9) in the exponent part specified in (s), or if there are multiple exponent parts, or exponent In some cases, 2BH(+) or 2DH(-) occurred twice or more.
	2BH(+) or 2DH(-) appears twice or more before the first digit of the string specified in (s).

Example



Device	+0	+1	+2	+3	+4
D0	5.	24	67	E+	12

The stored character string of D0 is: 5.2467E+12

The resulting floating point number is: 5.2467E+12

Device	+0	+1	+2
D96	0.000000E+000	0.000000E+000	5.2467E+12

6.9.25 DEXP/Single precision real number exponent operation

DEXP(P)

After performing the exponential calculation of the value specified in (s), the calculation result is stored in the device specified in (d).

-[DEXP (s) (d)]

Content, range and data type

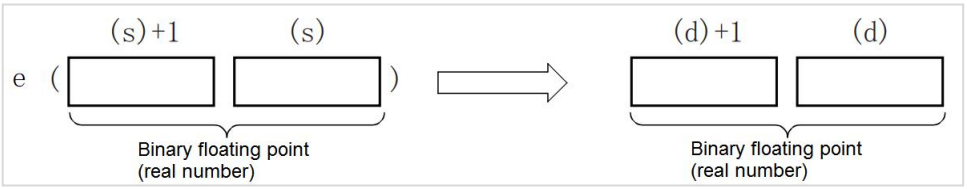
Parameter	Content	Range	Data type	Data type (label)
(s)	Data for exponential calculation or the device start number that stores the data	0, $2^{-126} \le \lvert(s)\rvert < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The device start number that stores the operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEXP	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

After performing the exponential calculation of the value specified in (s), the calculation result is stored in the device number specified in (d).



In exponential calculation, the base (e) is calculated as "2.71828".

The related devices are shown below.

Devices	Name	Content	
		Condition	Operating
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Calculate the result:



6.9.26 INT/Single precision real number → Signed BIN 16-bit data

INT(P)

Convert the specified single precision real number into signed BIN 16-bit data.

-[INT (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Single precision real number or the start device storing single precision real number	-32768 to 32767	Single precision real number	ANYREAL_32
(d)	Signed device for storing BIN data	-	BIN16 bit	ANY16_S

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
INT	Parameter 1															●	●	●	●	●	●	●				●	●
	Parameter 2															●	●	●	●	●						●	●

Features

• Convert the single precision real number specified in (s) into signed BIN 16-bit data and store it in the device specified in (d).

• The converted data will be rounded to the first digit below the decimal point of the single precision real number specified in (s).

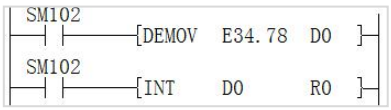
• When setting the input value with the engineering tool, rounding errors may occur.

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operation
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	Decimal places are rounded off when converting	During conversion ((s) – (d))> (2^{-126}) ,borrow (SM152) turns ON
SM151	Carry	Conversion result is out of range	The value of (s) is out of the range -32768 to 32767 or the value of (s) is less than the minimum value of 32-bit real numbers (2^{-126}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞



Get the conversion result:

Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
R0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	34

And the borrow means turn ON

SM151	0
SM152	1
SM153	0

6.9.27 DINT/Single precision real number→ Signed BIN 32-bit data

DINT(P)

Convert the specified single precision real number into signed BIN 32-bit data.

-[DINT (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Single precision real number or the start device storing single precision real number	-2147483648 to 2147483647	Single precision real number	ANYREAL_32
(d)	The start device storing BIN data	-	Signed BIN 32 bit	ANY32_S

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DINT	Parameter 1															●	●	●	●	●	●	●				●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

• Convert the binary floating point data specified in (s) into signed BIN 32-bit data and store it in the device specified in (d).

• The converted data will be rounded to the first digit below the decimal point of the binary floating point data specified in (s).

• When setting the input value with the engineering tool, rounding errors may occur.

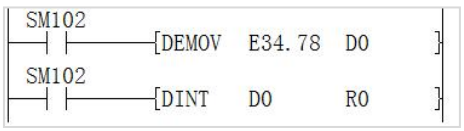
• The related devices are as follows.

Devices	Name	Content	
		Condition	Operating
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Get the conversion result:

Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
R0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	34

And the borrow means turn ON

SM151	0
SM152	1
SM153	0

6.9.28 DLOG10/Single precision real number common logarithmic operation

DLOG10(P)

Calculate the common logarithm (base 10 logarithm) of the value specified in (s), and store the result of the operation in the device specified in (d).

-[DLOG10 (s) (d)]

Content, range and data type

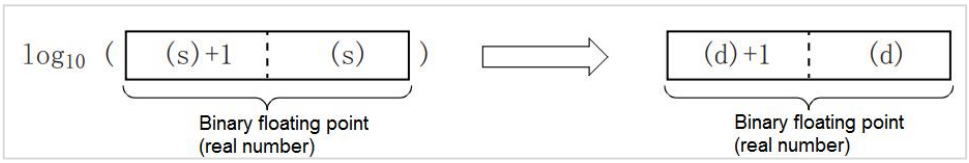
Parameter	Content	Range	Data type	Data type (label)
(s)	Data for common logarithmic operations or the device start number storing the data	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	The device start number storing operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DLOG10	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

Calculate the common logarithm (base 10 logarithm) of the value specified in (s), and store the result of the calculation in the device number specified in (d).



The value specified in (s) can only be set to a positive number. (Cannot perform operations with negative numbers.)

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operating
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result <2^{-126}	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



Get calculation results



6.9.29 DLOGE/Single precision real number natural logarithm operation

DLOGE(P)

After calculating the logarithm when the natural logarithm e of the value specified in (s) is the base, store the calculation result in the device specified in (d).

-[DLOGE (s) (d)]

Content, range and data type

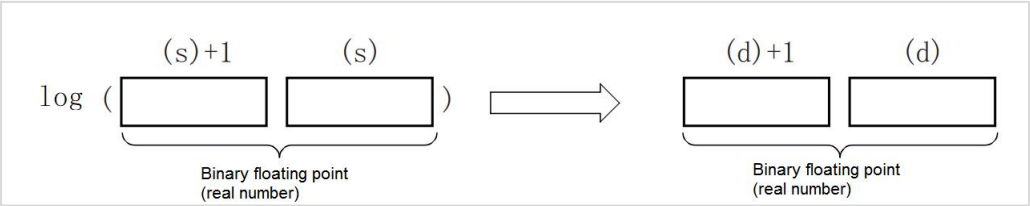
Parameter	Content	Range	Data type	Data type (label)
(s)	Data for logarithm operation or the device start number storing the data	0, $2^{-126} \leq \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(d)	the device start number storing operation result	-	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DLOGE	Parameter 1															●	●	●	●	●	●	●			●	●	●
	Parameter 2															●	●	●	●	●	●	●				●	●

Features

• After calculating the logarithm when the natural logarithm e of the value specified in (s) is the base, store the result of the calculation in the device number specified in (d).



• The value specified in (s) can only be set to a positive number. (Cannot perform operations with negative numbers.)

• The related devices are as follows.

Devices	Name	Content	
		Condition	Operating
SM153	Zero	The operation result is zero	The zero flag (SM153) turns ON.
SM152	Borrow	The absolute value of operation result $<2^{-126}$	The value of (d) becomes the minimum value of a 32-bit real number (2^{-126}), and the borrow flag (SM152) turns on.
SM151	Carry	The absolute value of operation result> 2^{128}	The value of (d) becomes the maximum value of 32-bit real numbers (2^{128}), and the carry flag (SM151) turns on.

Error code

Error code	Content
4085H	The write address in (s) exceeds the device range
4086H	The write address in (d) exceeds the device range
4084H	When the content of the device specified by (s) is an irregular number, a non-number and ±∞

Example



The result is as below:



6.10 Contact comparison instruction

6.10.1 Signed 16-bit contact comparison instruction

LD ☐, AND ☐, OR ☐

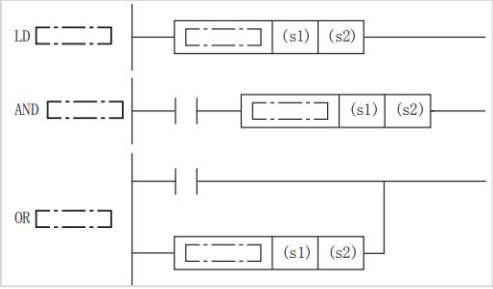
The BIN 16-bit data of the device specified in (s1) and the BIN 16-bit data of the device specified in (s2) are compared by normal open contact processing.

LD☐: Normally open contact comparison instruction

AND☐: Normally open contact series connection comparison instruction

OR☐: Normally open contact parallel connection comparison instruction

Ladder diagram



(You can enter "=", "<>", ">", "<", ">=", "<=" in "☐")

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparison data or device storing comparison data	-32768 to 32767	Signed BIN 16 bit	ANY16_S
(s2)	Comparison data or device storing comparison data	-32768 to 32767	Signed BIN 16 bit	ANY16_S

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LD=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
LD>	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
LD<	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
LD>=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
LD<=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
LD<>	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
AND=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
AND>	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
AND<	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
AND>=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
AND<=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
AND<>	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
OR=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
OR>	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
OR<	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
OR>=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
OR<=	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	
OR<>	s1, s2											●	●	●	●	●	●	●	●	●			●	●		●	

Features

The BIN 16-bit data of the device specified in (s1) and the BIN 16-bit data of the device specified in (s2) are compared by normal open contact processing.

The comparison operation result of each instruction is shown below.

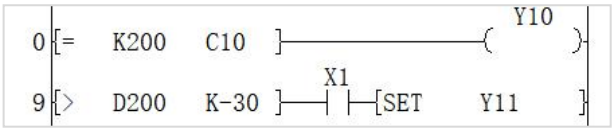
instruction Sign	Condition	Comparison operation result	instruction Sign	Condition	Comparison operation result
=	(s1)=(s2)	On state	=	(s1)≠(s2)	Non-conduction state
<>	(s1)≠(s2)		<>	(s1)=(s2)	
>	(s1)>(s2)		>	(s1)≤(s2)	
<	(s1)<(s2)		<	(s1)≥(s2)	
>=	(s1)≥(s2)		>=	(s1)<(s2)	
<=	(s1)≤(s2)		<=	(s1)>(s2)	

Error code

Error code	Content
4085H	(s) read address exceeds the device range

Example

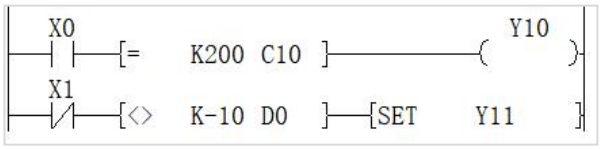
(1) LD□ instruction:



When the current value of counter C10 is 200, Y10 is set

When the content of D200 is above -29, and X1 is ON, Y11 is set

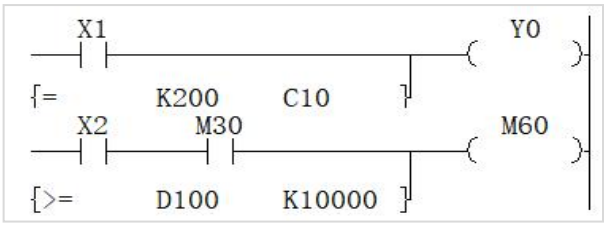
(2) AND□ instruction:



When X0 is ON, or when the current value of counter C10 is 200, Y10 is set

When X1 is OFF, and the content of data counter D0 is not -10, Y11 is set

(3) OR□ instruction:



When X1 is ON, or when the current value of counter C10 is 200, Y0 is set

When X2 and M30 are ON, or the content of register D100 is K10000 and above, M60 is set

6.10.2 Signed 32-bit contact comparison instruction

LDD ☐, ANDD ☐, ORD ☐

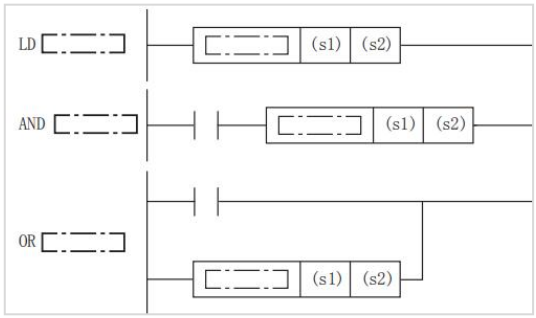
The BIN 32-bit data of the device specified in (s1) and the BIN 32-bit data of the device specified in (s2) are compared by normal open contact processing.

LDD☐: Normally open contact comparison instruction

ANDD☐: Normally open contact series link comparison instruction

ORD☐: Normally open contact parallel link comparison instruction

Ladder diagram



"=", "<>", ">", "<", ">=", "<=" can be input in "☐"

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparison data or device storing comparison data	-2147483648 to 2147483647	Signed BIN 32 bit	ANY32_S
(s2)	Comparison data or device storing comparison data	-2147483648 to 2147483647	Signed BIN 32 bit	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LDD=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
LDD>	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
LDD<	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
LDD>=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
LDD<=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
LDD<>	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ANDD=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ANDD>	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ANDD<	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ANDD>=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ANDD<=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ANDD<>	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ORD=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ORD>	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ORD<	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ORD>=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ORD<=	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	
ORD<>	s1, s2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	

Features

The BIN 32-bit data of the device specified in (s1) and the BIN 32-bit data of the device specified in (s2) are compared by normal open contact processing.

The comparison operation result of each instruction is shown below.

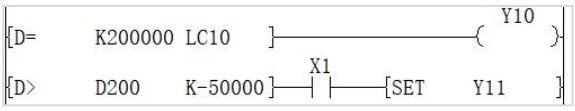
instruction Sign	Condition	Comparison operation result	instruction Sign	Condition	Comparison operation result
=	(s1)=(s2)	On state	=	(s1)≠(s2)	Non-conduction state
<>	(s1)≠(s2)		<>	(s1)=(s2)	
>	(s1)>(s2)		>	(s1)≤(s2)	
<	(s1)＜(s2)		<	(s1)≥(s2)	
>=	(s1)≥(s2)		>=	(s1)＜(s2)	
<=	(s1)≤(s2)		<=	(s1)>(s2)	

Error code

Error code	Content
4085H	(S) read address exceeds the device range

Example

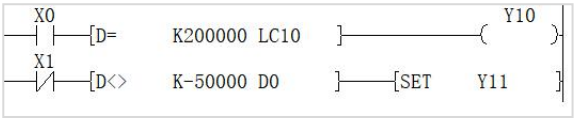
(1) LDD□ instruction:



When the data of LC10 is 200000, Y10 is set, otherwise Y10 is reset.

When the 32-bit data composed of D201 and D200 exceeds -5000, and X1 is ON, Y11 is turned ON.

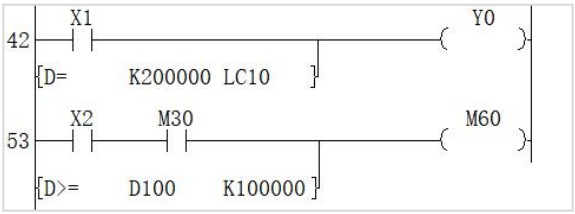
(2) ANDD□ instruction:



When X0 is ON and the value of LC10 is 200000, Y10 is set, otherwise it is reset.

When X1 is OFF and the 32-bit data composed of D1 and D0 is not equal to K-50000, Y11 is set.

(3) ORD□ instruction:



When X1 is ON, or the data of LC10 is equal to the data of LC10 is equal to 200000, Y0 is set.

When X2 and M30 are set, or the double word data composed of D101 and D100 is greater than or equal to 100000, M60 is set.

6.10.3 Single precision real number contact comparison instruction

LDE ☐, ANDE ☐, ORE ☐

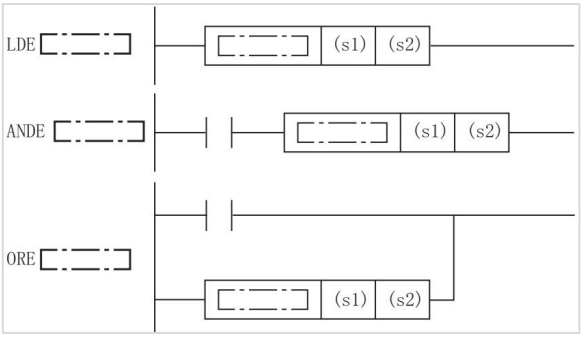
The single precision real number of the device specified in (s1) and the single precision real number of the device specified in (s2) are compared by normal open contact processing.

LDE□: Normally open contact comparison instruction

ANDE□: Normally open contact series link comparison instruction

ORE□: Normally open contact parallel link comparison instruction

Ladder diagram



"=", "<>", ">", "<", ">=", "<=" can be input in "☐"

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparison data or the device start number storing the comparison data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32
(s2)	Comparison data or the device start number storing the comparison data	0, $2^{-126} \le \|(s)\| < 2^{128}$	Single precision real number	ANYREAL_32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LDE=	s1, s2															●	●	●	●	●	●	●			●	●	
LDE>	s1, s2															●	●	●	●	●	●	●			●	●	
LDE<	s1, s2															●	●	●	●	●	●	●			●	●	
LDE>=	s1, s2															●	●	●	●	●	●	●			●	●	
LDE<=	s1, s2															●	●	●	●	●	●	●			●	●	
LDE<>	s1, s2															●	●	●	●	●	●	●			●	●	
ANDE=	s1, s2															●	●	●	●	●	●	●			●	●	
ANDE>	s1, s2															●	●	●	●	●	●	●			●	●	
ANDE<	s1, s2															●	●	●	●	●	●	●			●	●	
ANDE>=	s1, s2															●	●	●	●	●	●	●			●	●	
ANDE<=	s1, s2															●	●	●	●	●	●	●			●	●	
ANDE<>	s1, s2															●	●	●	●	●	●	●			●	●	
ORE=	s1, s2															●	●	●	●	●	●	●			●	●	
ORD>	s1, s2															●	●	●	●	●	●	●			●	●	
ORE<	s1, s2															●	●	●	●	●	●	●			●	●	
ORE>=	s1, s2															●	●	●	●	●	●	●			●	●	
ORE<=	s1, s2															●	●	●	●	●	●	●			●	●	
ORE<>	s1, s2															●	●	●	●	●	●	●			●	●	

Features

The single precision real number of the device specified in (s1) and the single precision real number of the device specified in (s2) are compared by normal open contact processing.

The comparison operation result of each instruction is shown below.

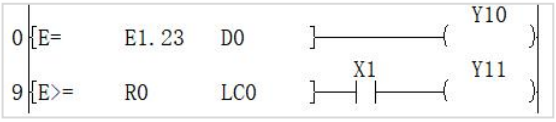
instruction Sign	Condition	Comparison operation result	instruction Sign	Condition	Comparison operation result
E=	(s1)=(s2)	On state	E=	(s1)≠(s2)	Non-conduction state
E<>	(s1)≠(s2)		E<>	(s1)=(s2)	
E>	(s1)>(s2)		E>	(s1)≤(s2)	
E<	(s1)＜(s2)		E<	(s1)≥(s2)	
E>=	(s1)≥(s2)		E>=	(s1)＜(s2)	
E<=	(s1)≤(s2)		E<=	(s1)>(s2)	

Error code

Error code	Content
4084H	When the content of the specified device by (s1) and (s2) is an irregular number, a non-number, or ±∞
4085H	The read address of (s1) and (s2exceeds the device range

Example

(1) LDE□ instruction:



When the real number input in D0 is equal to E1.23, Y10 is ON, otherwise Y10 is OFF.

When the real number in R0 is greater than or equal to the real number in LC0, Y11 is ON, otherwise it is OFF.

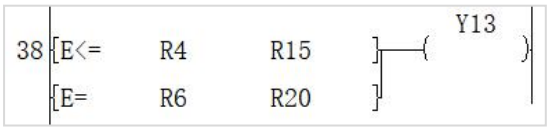
If the input in D0, R0, LC0 is not a real number, it will report H4084 error.

(2) ANDE□ instruction:



Only when M0 is ON and D2 real number is not equal to E1.23 and R2 real number is less than real number LC2 , Y12 is ON, otherwise all are OFF.

(3) ORE□ instruction:



When the real number of R4 is less than or equal to the real number of R15, or the real number R6 is equal to the real number R20, Y13 is ON, otherwise Y13 is OFF.

6.10.4 String comparison

LDS☐. ANDS☐. ORS☐

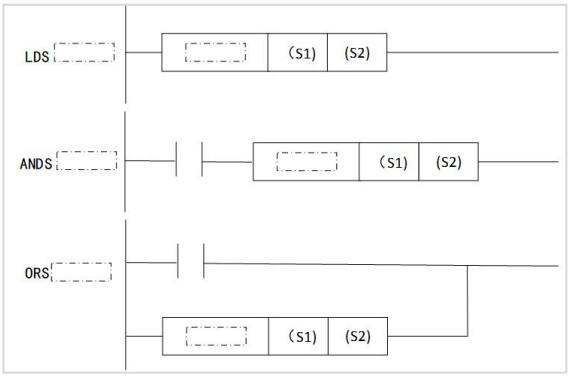
Compare the string stored after the device number specified in (s1) with the string stored after the device number specified in (s2).

LDS☐: String comparison instruction

ANDS☐: String serial connection comparison instruction

ORS☐: String parallel connection comparison instruction

Ladder diagram



"=" and "<>" could be entered in "☐"

Content, range and data type

Parameter	Content	Range	Date type	Date type(label)
(S1)	Connection data or the device start number storing the data or the string specified directly	-	String	ANYSTRING_SINGLE
(S1)	Connection data or the device start number storing the data or the string specified directly	-	String	ANYSTRING_SINGLE

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LDS=	s1. s2											●	●	●	●	●	●	●	●	●						●	
LDS<>	s1. s2											●	●	●	●	●	●	●	●	●						●	
ANDS=	s1. s2											●	●	●	●	●	●	●	●	●						●	
ANDS<>	s1. s2											●	●	●	●	●	●	●	●	●						●	
ORS=	s1. s2											●	●	●	●	●	●	●	●	●						●	
ORS<>	s1. s2											●	●	●	●	●	●	●	●	●						●	

Features

• Compare the string stored after the device number specified in (s1) with the string stored after the device number specified in (s2).

• The comparison operation result of each instruction is shown below.

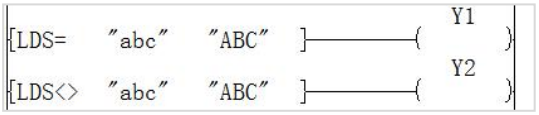
instruction sign	Condition	Comparison operation result	instruction sign	Condition	Comparison operation result
=	(s1)＝(s2)	On stat	=	(s1)≠(s2)	Non-conduction state
<>	(s1)≠(s2)		<>	(s1)＝(s2)	

Error code

Error code	Content
4085H	The read address of (s1) or (s2) exceeds the device range
408AH	The length of the read string of (s1) or (s2) exceeds, and the continuous length of the string exceeds 400 characters.
408BH	When (s1) or (s2) reading the string, the maximum range of the device is read but 00H is not found as the end.

Example

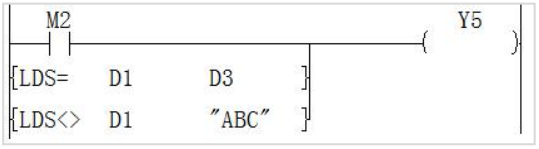
(1) 1. LDS☐ instruction



(2) ANDS☐ instruction



(3) ORS☐ instruction



6.11 Clock operation instruction

6.11.1 TADD/The addition of clock data

TADD(P)

Add the time data stored after the device number specified in (s1) and the time data stored after the device number specified in (s2), and store the result of the addition operation after the device number specified in (d) .

-[TADD (s1) (s2) (d)]

Content, range and data type

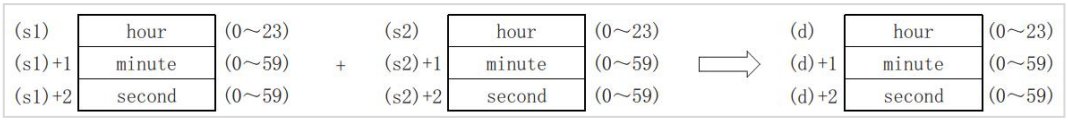
Parameter	Content	Range	Data type	Data type (label)
(s1)	The device start number that stores the added time data	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(s2)	The device start number that stores the addition operation time (time) data	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(d)	The device start number that stores the time (time) data of the addition operation result	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TADD	Parameter 1															●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3															●	●	●	●	●						●	●

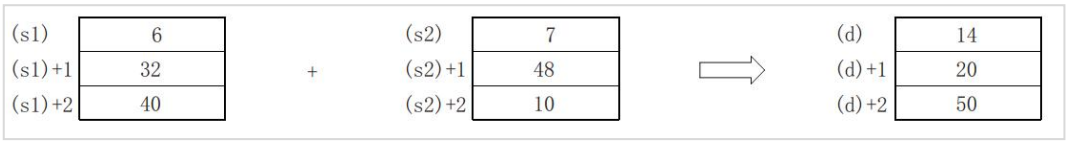
Features

Add the time data specified in (s1) and the time data specified in (s2), and store the result of the addition in the device number specified in (d) or later.

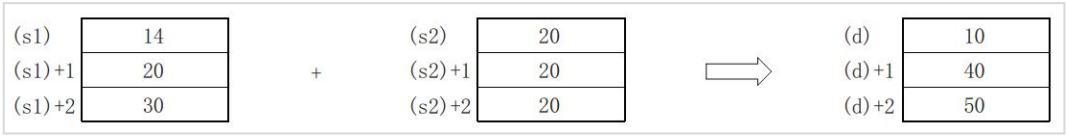


Example

When 6:32:40 and 7:48:10 are added together



When the calculation result time exceeds 24 o'clock, the carry flag turns ON, and the value after 24 hours is subtracted becomes the calculation result. For example, when 14:20:30 and 20:20:20 are added, the result is not 34:40:50, but 10:40:50.



When the calculation result is 0 (0 hour, 0 minute, 0 second), the zero flag turns on.

When 23:59:59 and 1 second are added, the result of the calculation is 0:00:00, and the carry flag and zero flag are turned on.

Related device are as follows:

Devices	Name	Content
SM151	Carry	It is ON when the result of the TADD(P) instruction exceeds the maximum clock data value of 23:59:59

Devices	Name	Content
SM153	Zero	It is ON when the result of the TADD(P) instruction is 0:00:00

✎Note

The devices specified in (s1), (s2), (d) occupy 3 points respectively. Be careful not to overlap with the device used for machine control.

When using the clock data time (hour, minute, second) of the built-in real-time clock of the CPU module, use the TRD(P) instruction to read the value of the special register and assign the word device to each operand.

Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the values specified in (s1) and (s2) are other than 0 to 23 When the values specified in (s1)+1, (s2)+1, (s1)+2 and (s2)+2 are other than 0 to 59

Example



Set D0 time to 16:30:00 and D10 time to 4:30:0

D0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	16
D1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	30
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

D10	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4
D11	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	30
D12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

After the coil is turned on, the D20 time is 21:0:0

D20	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	21
D21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.11.2 TSUB/The subtraction of clock data

TSUB(P)

Subtract the time data stored after the device number specified in (s1) and the time data stored after the device number specified in (s2), and store the subtraction result in the device number specified in (d) or later .

-[TSUB (s1) (s2) (d)]

Content, range and data type

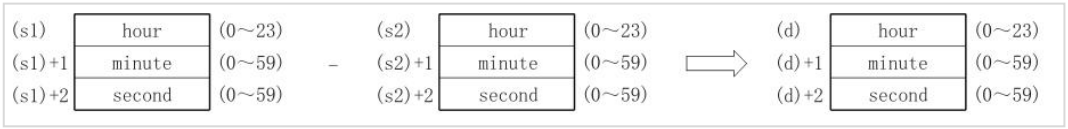
Parameter	Content	Range	Data type	Data type (label)
(s1)	The device start number that stores the subtracted time data	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(s2)	The device start number that stores the subtraction operation time (time) data	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(d)	The device start number that stores the time (time) data of the subtraction result	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TSUB	Parameter 1															●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3															●	●	●	●	●						●	●

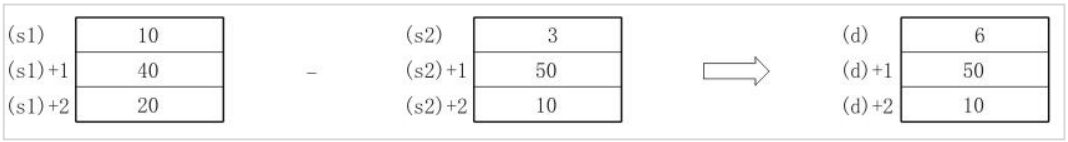
Features

Subtract the time data specified in (s1) and the time data specified in (s2), and store the subtraction result in the device number specified in (d) or later.

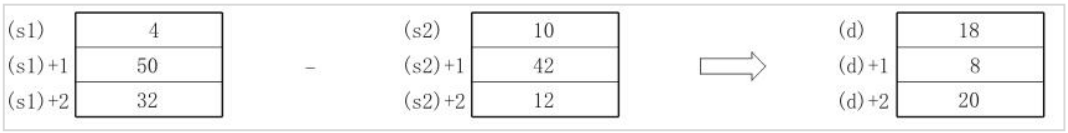


Example

When subtracting 10:40:20 and 3:50:10



When the calculation result time is a negative number, the borrow flag turns on and the data +24 is the calculation result. For example, in the case of subtracting 4:50:32 and 10:42:12, the result is not -6:8:20, but 18:8:20.



When the calculation result is 0 (0 hour, 0 minute, 0 second), the zero flag turns on.

Related device are as follows:

Devices	Name	Content
SM152	Borrow	It is ON when the result of the TSUB(P) instruction is less than 0:00:00
SM153	Zero	It is ON when the result of the TSUB(P) instruction is at the time of 0:00:00:00

✎Note

• The devices specified in (s1), (s2), and (d) occupy 3 points respectively. Be careful not to overlap with the device used for machine control.

• When using the clock data time (hour, minute, second) of the built-in real-time clock of the CPU module, use the TRD(P) instruction to read the value of the special register and assign the word device to each operand.

Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the values specified in (s1) and (s2) are other than 0 to 23 When the values specified in (s1)+1, (s2)+1, (s1)+2 and (s2)+2 are other than 0 to 59

Example



Set D0 time to 16:30:00 and D10 time to 4:30:0

D0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	16
D1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	30
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

D10	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4
D11	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	30
D12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

After the coil is turned on, the D20 time is 12:00:00

D20	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	12
D21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.11.3 TRD/Clock data reading

TRD(P)

Read the clock data of the built-in real-time clock of the CPU module.

-[TRD (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	Read destination and start device number of clock data	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 7)

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TRD	Parameter 1															●	●	●	●							●	●

Features

Read the clock data (SD100 to SD106) of the real-time clock built into the CPU module into (d) to (d)+6 in the following format.

Parameter	Element	Project	Clock data		Element	Project
Special register	SD105	Year (Gregorian)	2000 to 2099	→	(d)	Year (Gregorian)
	SD104	Month	1 to 12	→	(d)+1	Month
	SD103	Day	1 to 31	→	(d)+2	Day
	SD102	Hour	0 to 23	→	(d)+3	Hour
	SD101	Minute	0 to 59	→	(d)+4	Minute
	SD100	Seconds	0 to 59	→	(d)+5	Seconds
	SD106	Week	0 (Sun) to 6 (Sat)	→	(d)+6	Week

• The related devices are shown below. The clock data of these special registers are updated through END processing.

Devices	Content
SD100	The second data of the clock data is stored in BIN code.
SD101	The sub-data of the clock data is stored in BIN code.
SD102	Time data of clock data is stored in BIN code.
SD103	The daily data of the clock data is stored in BIN code.
SD104	The monthly data of the clock data is stored in BIN code.
SD105	The year data of the clock data is stored in a 4-digit BIN code of the Gregorian calendar.
SD106	The week data of the clock data is stored in BIN code. (0: day, 1: one, ..., 6: six) are stored in BIN code.

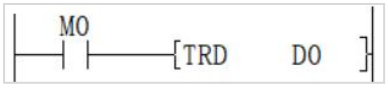
✎Note：

• The device specified in (d) occupies 7 points. Be careful not to overlap with the device used for machine control.

Error code

Error code	Content
4086H	When reading the specified device range exceeds the corresponding device range

Example



After the M0 coil is turned on, the current date and time are read as 2020-2-19 13:10:38 Wednesday

Device	+0	+1	+2	+3	+4	+5	+6
D0	2020	2	19	13	10	38	3

6.11.4 TWR/Clock data writing

TWR(P)

Write the clock data of the built-in real-time clock of the CPU module.

-[TWR (s)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Clock data write source, start device number	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 7)

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TWR	Parameter 1															●	●	●	●							●	●

Features

Write the set clock data (s) to (s)+6 to the clock data (SD100 to SD106) of the real-time clock built into the CPU module.

Set data at all times				Special register	
Element	Project	Clock data		Element	Project
(s)	Year (Gregorian)	2000 to 2099 or 0 to 99	→	SD105	Year (Gregorian)
(s)+1	Month	1 to 12	→	SD104	Month
(s)+2	Day	1 to 31	→	SD103	Day
(s)+3	Hour	0 to 23	→	SD102	Hour
(s)+4	Minute	0 to 59	→	SD101	Minute
(s)+5	Seconds	0 to 59	→	SD100	Seconds
(s)+6	Week	0 (Sun) to 6 (Sat)	→	SD106	Week

• If the TWR(P) instruction is executed, the clock data of the real-time clock is changed immediately. Therefore, the clock data after a few minutes should be transferred to the set clock data (s) to (s)+6 in advance, and the instruction will be executed when the correct time is reached.

• If the year in (s) is in the range of 0 to 99, it will be automatically treated as 2000 to 2099.

• When a value indicating an impossible time is set, the clock data will not be updated. Set the correct clock data and write again.

• The day of the week (SD100) is automatically corrected.

• The related devices are shown below.

Devices	Content
SD100	The second data of the clock data is stored in BIN code.
SD101	The sub-data of the clock data is stored in BIN code.
SD102	Time data of clock data is stored in BIN code.
SD103	The daily data of the clock data is stored in BIN code.
SD104	The monthly data of the clock data is stored in BIN code.
SD105	The year data of the clock data is stored in a 4-digit BIN code of the Gregorian calendar.
SD106	The week data of the clock data is stored in BIN code. (0: day, 1: one, ..., 6: six) are stored in BIN code.

✎Note

The device specified in (s) occupies 7 points. Be careful not to overlap with the device used for machine control.

Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range

Example

Set D0 date and time to 2020-2-19 12:36:00 in advance



At the moment when the time 12:36:00 arrives, turn on the M0 coil and write the time.

Device	+0	+1	+2	+3	+4	+5	+6
D0	2020	2	19	12	36	0	0

6.11.5 HTOS/16-bit data conversion of time data (hour, minute, second → second)

HTOS(P)

Convert the time data stored after the device number specified in (s) into seconds and store the conversion result as BIN 16-bit data in the device specified in (d).

-[HTOS (s) (d)]

Content, range and data type

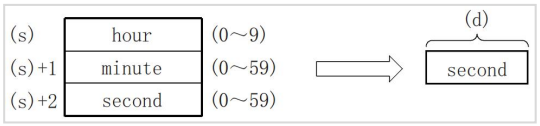
Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number that stores the data of the subtracted time	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(d)	The device start number that stores the converted clock data	-	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
HTOS	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

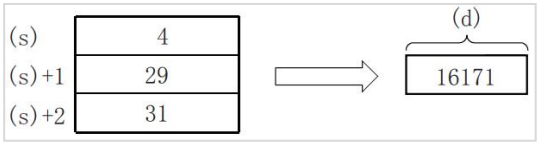
Features

Convert the time data stored after the device number specified in (s) into seconds and store the conversion result in the device specified in (d).



Example

When 4 hours, 29 minutes and 31 seconds are specified in (s)



Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the calculation result is not in the range of 0 to 32767 When the value specified in (s) is not in the range of 0 to 9 When the value specified in (s)+1 and (s)+2 is not in the range of 0 to 59

Example



D0 time is set to 5:36:53

D0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	5
D1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	36
D2	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	53

The time of D10 after the M0 coil is turned on is as below.

D10	1	0	1	0	1	1	1	1	0	1	1	1	0	0	1	0	20213

6.11.6 DHTOS/32-bit data conversion of time data (hour, minute, second → second)

DHTOS(P)

Convert the time data stored after the device number specified in (s) into seconds and store the conversion result as BIN 32-bit data in the device specified in (d).

-[DHTOS (s) (d)]

Content, range and data type

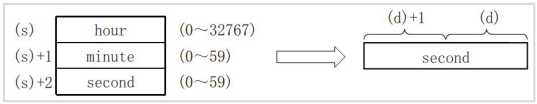
Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number that stores the data of the subtracted time	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(d)	The device start number that stores the converted clock data	-	Signed BIN 32 bit	ANY32

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DHTOS	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

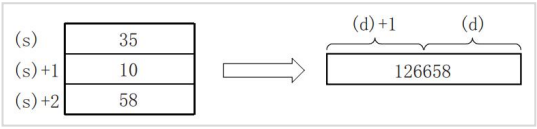
Features

Convert the time data stored after the device number specified in (s) into seconds and store the conversion result in the device specified in (d).



Example

When 35 hours, 10 minutes and 58 seconds are specified in (s)



Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the calculation result is not in the range of 0 to 32767 When the value specified in (s)+1 and (s)+2 is not in the range of 0 to 59

Example



D0 time is set to 15:33:24

Device	+0	+1	+2
D0	15	33	24

The second of D10 after the M0 coil is turned on is

Device	+0	+1
D0	2162703	24
D8	0	56004

6.11.7 HOUR/Hour measuring 16-bit

HOUR(P)

The time for the input contact to be ON is measured in units of one hour.C

-[HOUR (s) (d1) (d2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The time when the alarm (d2) is turned ON (set by one hour)	K0 to K32767	Signed BIN 16 bit	ANY16
(d1)	Device that stores the current value of measurement (specified data register for power failure retention)	-	Unsigned BIN 16 bit	ANY16_ARRAY (Number of elements: 2)
(d2)	Device that turns ON when the time limit expires (alarm output)	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
HOUR	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2																	●	●	●						●	●
	Parameter 3		●	●	●	●					●															●	●

Features

The input contact ON time is measured in units of 1 hour. When the cumulative ON time exceeds the time (BIN 16-bit data) specified in (s), the device specified in (d2) is turned on .

• In (s), set the time until the alarm (d2) turns ON in units of 1 hour.

• (d1) stores the current measured value in units of 1 hour.

• If the median value of (d1) exceeds 32767, it will be modified to 32767.

• (d1)+1 stores the current measured value (in units of 1 second) that is less than 1 hour.

• (d2) turns on when the current value (d1) exceeds the time specified in (s).

• In order to continue to use the current value data even after the power of the CPU module is turned off, specify the data register for power failure retention to (d1). If you use general-purpose data registers, the current value data will be cleared by powering off the CPU module and STOP→RUN operations.

• After the alarm output specified in (d2) turns ON, measurement will continue.

• The measurement stops when the current value reaches the 16-bit maximum. To continue the measurement, clear the current value of (d1) to (d1)+1.

✎Note

• The device specified in (d1) occupies 2 points. Be careful not to overlap with the device used for machine control.

• After the instruction stops running, the measurement stops and the output continues to be maintained.

Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the value of (s) is negative

Example



When M0 = ON, the duration of the state is accumulated, the time is recorded in D0, and the seconds less than 1 hour are recorded in D1. When the accumulated time of D0 reaches 98 hours, the Y0 output state is ON. When the timing conditions are met, after reaching the specified value (K98), the accumulated timing will continue and the reading will continue to increase; the current time value D0 reaches the maximum value of 32767 hours and D1 reaches 3599 seconds, the timing measurement will stop. The current time values D0 and D1 are cleared to 0.

6.11.8 DHOUR/Hour measuring 32 bits

DHOUR(P)

The time for the input contact to be ON is measured in units of one hour.

-[DHOUR (s) (d1) (d2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The time when the alarm (d2) is turned ON (set by one hour)	0 to 2147483647	Signed BIN 32 bit	ANY32
(d1)	Device that stores the current value of measurement (specified data register for power failure retention)	-	Unsigned BIN 32 bit	ANY32_ARRAY (Number of elements: 2)
(d2)	Device that turns ON when the time limit expires (alarm output)	-	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DHOUR	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2																	●	●	●						●	●
	Parameter 3		●	●	●	●					●															●	●

Features

• The input contact ON time is measured in units of 1 hour. When the cumulative ON time exceeds the time (BIN 32-bit data) specified in (s), the device specified in (d2) is set to ON.

• In (s)+1, (s), set the time until the alarm (d2) turns ON in units of 1 hour.

• (D1)+1 and (d1) store the current value measured in units of 1 hour. ((d1)+1: high bit, (d1): low bit)

• If the median of (d1)+1 and (d1) exceeds 2147483647, it will be modified to 2147483647.

• (D1)+2 stores the current value (in units of 1 second) of the measurement that is less than 1 hour.

• (D2) turns on when the current value (d1)+1, (d1) exceeds the time specified in (s).

• In order to continue to use the current value data even after the power of the CPU module is turned off, specify the data register for power failure retention to (d1). If you use general-purpose data registers, the current value data will be cleared by powering off the CPU module and STOP→RUN operations.

• After the alarm output specified in (d2) turns on, the measurement will continue.

• The measurement stops when the current value reaches the 32-bit maximum. To continue the measurement, clear the current value of (d1) to (d1)+2.

✎Note

• The device specified in (d1) occupies 3 points. Be careful not to overlap with the device used for machine control.

• After the instruction stops running, the measurement stops and the output continues to be maintained.

Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the value of (s) is negative

Example



When M0=ON, the duration of this state is accumulated, the time is recorded in D1, D0, and the seconds less than 1 hour are recorded in D2. When the accumulated time of D1, D0 reaches 1000 hours, the Y0 output state is ON. When the timing conditions are met, after reaching the specified value (K1000), the accumulated timing will continue, and the reading will continue to increase; the current time values D1 and D0 reach the maximum value of 2147483647 hours, and when D2 reaches 3599 seconds, the timing measurement will stop and the timing should be restarted. The current time values D0, D1, and D2 must be cleared to 0.

6.11.9 STOH/16-bit data conversion of time data (second → hour, minute, second)

STOH(P)

Convert the second 16-bit data stored in the device number specified in (s) into hour, minute, and second, and store the conversion result in the device specified in (d) and later.

-[STOH (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number that stores the clock data before conversion	0 to 32767	Signed BIN 16 bit	ANY16
(d)	The device start number that stores the converted clock data	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
STOH	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

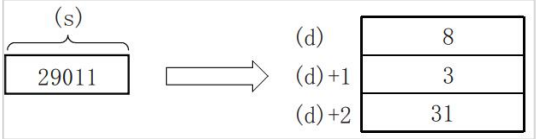
Features

Convert the second data stored after the device number specified in (s) into hour, minute, and second, and store the conversion result in the device specified in (d) and later.



Example

When 29,011 seconds are specified in (s)



Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the value of (s) exceeds the range

Example



Set D0 seconds to 12537

The hour, minute and second of D10 after the M0 coil is turned on are

D10	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
D11	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	28
D12	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	57

6.11.10 DSTOH/32-bit data conversion of time data (second → hour, minute, second)

DSTOH(P)

Convert the second 32-bit data of second stored in the device number specified in (s) into hour, minute, and second, and store the conversion result in the device specified in (d) and later.

-[DSTOH (s) (d)]

Content, range and data type

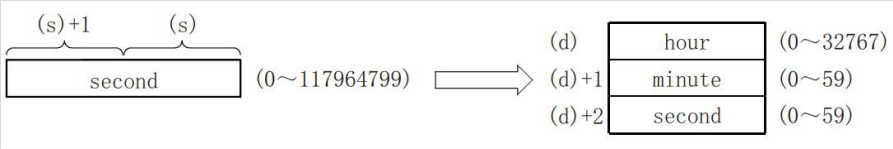
Parameter	Content	Range	Data type	Data type (label)
(s)	The device start number that stores the clock data before conversion	0 to 117964799	Signed BIN 32 bit	ANY32
(d)	The device start number that stores the converted clock data	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSTOH	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

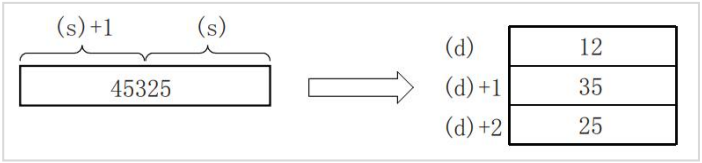
Features

Convert the second data stored after the device number specified in (s) into hour, minute, and second, and store the conversion result in the device specified in (d) and later.



Example

When 45,325 seconds is specified in (s)



Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the value of (s) exceeds the range

Example



Set D0 seconds to 2152537

The hour, minute and second of D10 after the M0 coil is turned on are

D10	1	0	1	0	1	0	1	0	0	1	0	0	0	0	0	0	597
D11	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	55
D12	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	37

6.11.11 TCMP/Clock data comparison

TCMP(P)

Compare the comparison time specified in (s1), (s2), (s3) with the time data specified in (s4), and turn the bit device specified in (d) ON/OFF according to their size match.

-[TCMP (s1) (s2) (s3) (s4) (d)]

Content, range and data type

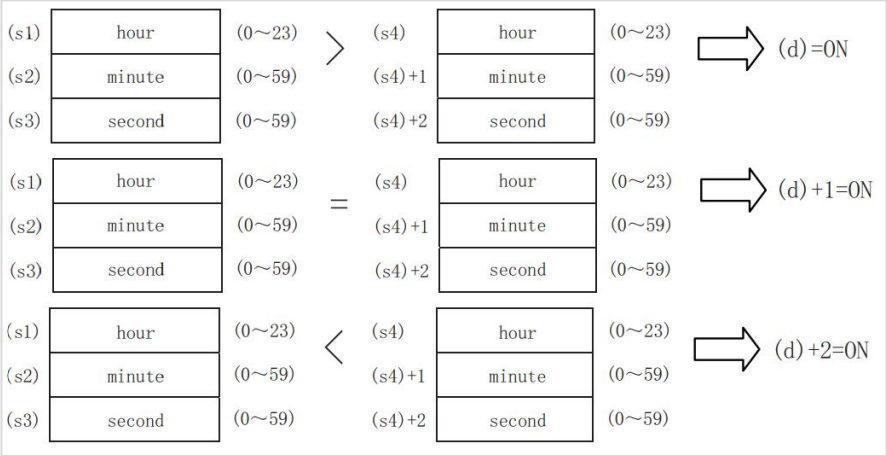
Parameter	Content	Range	Data type	Data type (label)
(s1)	Specify the "hour" of the comparison base time.	0 to 23	Signed BIN 16 bit	ANY16
(s2)	Specify the "minute" of the comparison base time.	0 to 59	Signed BIN 16 bit	ANY16
(s3)	Specify the "second" of the comparison base time.	0 to 59	Signed BIN 16 bit	ANY16
(s4)	Specify the "hour" of the time data (hour, minute, second).	-	Signed BIN 16 bit	ANY16_ARRAY
(d)	The bit device is turned ON/OFF according to the comparison result.	-	Bit	ANYBIT_ARRAY

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TCMP	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 4															●	●	●	●	●						●	●
	Parameter 5		●	●	●	●					●															●	●

Features

Compare the time of the reference time (hour, minute, second) [(s1), (s2), (s3)] with the time data (hour, minute, second) [(s4), (s4)+1, (s4) +2] Compare the size and turn on/off the 3 points from (d) according to the result of the same size.



✎Note

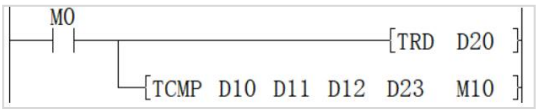
The device specified in (s4) and (d) occupies 3 points. Be careful not to overlap with the device used for machine control.

When using the clock data time (hour, minute, second) of the built-in real-time clock of the CPU module, use the TRD(P) instruction to read the value of the special register and assign the word device to each operand.

Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the value specified in (s) and (s4) is not in the range of 0 to 23 When the value specified in (s2),(s3) (s4)+1 and (s4)+2 is not in the range of 0 to 59

Example



Set D10 to 1, D11 to 30, D12 to 0

When M0 is turned on, the time that D23 will come is 0:31:27

D23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D24	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	31
D25	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	27

M10 is turned ON

M10	1
M11	0
M12	0

6.11.12 TZCP/Clock data bandwidth comparison

TZCP(P)

Compare the comparison time of the high and low points specified in (s1) and (s2) with the time data specified in (s3), and turn the bit device specified in (d) ON/OFF according to its size and bandwidth.

-[TZCP (s1) (s2) (s3) (d)]

Content, range and data type

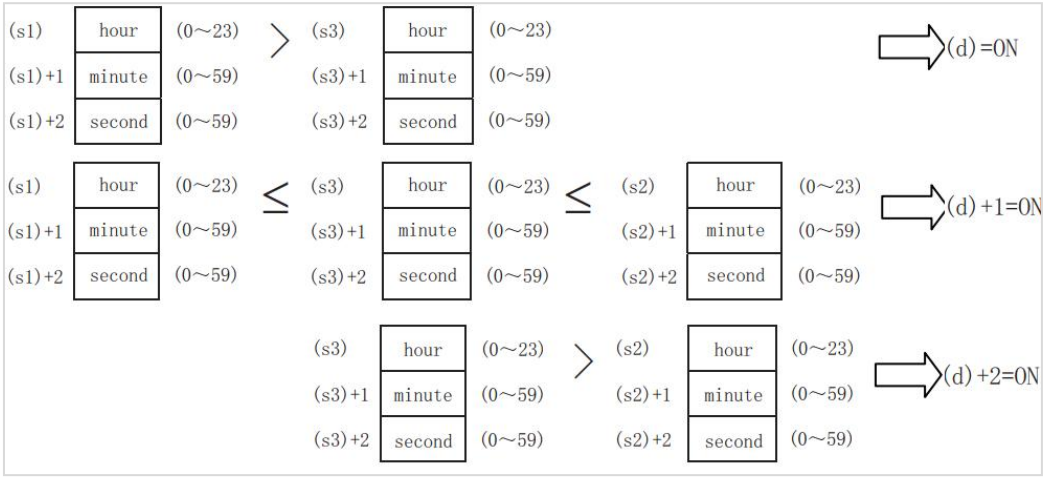
Parameter	Content	Range	Data type	Data type (label)
(s1)	Specify the "hour" of the lower limit time (hour, minute, second)	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(s2)	Specify the "hour" of the lower limit time (hour, minute, second)	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(s3)	Specify "hour" of time data (hour, minute, second)	-	Signed BIN 16 bit	ANY16_ARRAY (number of elements: 3)
(d)	The bit device is turned ON/OFF according to the comparison result.	-	Bit	ANY16_ARRAY (number of elements: 3)

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TZCP	Parameter 1															●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3															●	●	●	●	●						●	●
	Parameter 4		●	●	●	●					●															●	●

Features

Compare the comparison time of the high and low points specified in (s1) and (s2) with the time data specified in (s3), and turn the bit device specified in (d) ON/OFF according to its size and bandwidth.



✎Note

• The devices specified in (s1), (s2), (s3), (d) occupy 3 points. Be careful not to overlap with the device used for machine control.

• When using the clock data time (hour, minute, second) of the built-in real-time clock of the CPU module, use the TRD(P) instruction to read the value of the special register and assign the word device to each operand.

• When (s1)> (s2), two of (d), (d)+1, (d)+2 are ON/OFF.

Error code

Error code	Content
4085H	When reading the specified device range exceeds the corresponding device range
4086H	When writing the specified device range exceeds the corresponding device range
4084H	When the value specified in (s1), (s2) and (s3) is not in the range of 0 to 23 When the value specified in (s1)+1, (s2)+1, (s3)+1, (s1)+2, (s2)+2 and (s3)+2 is not in the range of 0 to 59

Example



Set D0 time to 16:30:00 and D10 time to 4:30:0

D0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	16
D1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	30
D2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

D10	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4
D11	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	30
D12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

After the coil is turned on, the reading time to D20 time is 8:30:00

D20	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	8
D21	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	30
D22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

M10/M12 is ON

M10	1
M11	0
M12	1

6.12 Data control instructions

6.12.1 BAND/BIN 16-bit data dead zone control

BAND(P)

The input value (BIN 16-bit value) specified in (s3) controls the output value stored in the device specified in (d) according to the upper and lower limits of the dead zone specified in (s1) and (s2).

-[BAND (s1)　(s2)　(S3)　(d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Lower limit of dead zone (no output zone)	-32,768 to +32,767	Signed BIN 16 bit	ANY16_S
(s2)	Upper limit of dead zone (no output zone)	-32,768 to +32,767	Signed BIN 16 bit	ANY16_S
(S3)	Input value controlled by dead zone control	-32768 to +32,767	Signed BIN 16 bit	ANY16_S
(D)	The start number of the device that stores the output value controlled by the dead zone control	-	Signed BIN 16 bit	ANY16_S

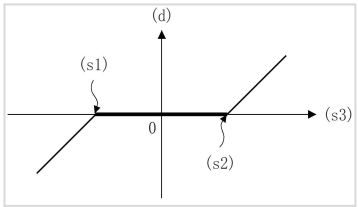
Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BAND	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3											●	●	●	●	●	●	●	●	●						●	●
	Parameter 4												●	●	●	●	●	●	●	●						●	●

Features

The input value (BIN 16-bit value) specified in (s3) controls the output value stored in the device specified in (d) according to the upper and lower limits of the dead zone specified in (s1) and (s2). The output value is controlled as follows.

Condition	The value stored in the output value
When dead zone low limit (s1)> input value (s3)	Input value (s3)-Dead zone low limit (s1)
When dead zone high limit (s1) <input value (s3)	Input value (s3)- Dead zone high limit (s2)
When dead zone low limit (s1) ≤ input value (s3) ≤ dead zone low limit (s2)	0



•When the output value stored in (d) is a signed BIN 16-bit value, and the operation result exceeds the range of -32768 to 32767, the situation is shown in the following example.

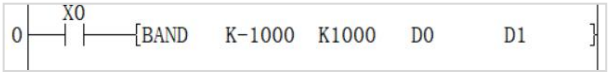
For example, when (s1) is 10 and (s3) is -32768,

the output value = -32768-10=8000H-000AH=7FFFH=32758.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device.
4086H	When the specified device range for writing exceeds the range of the corresponding device.
4084H	When the low limit specified in (s1) is greater than the high limit specified in (s2).

Example



When X000 is ON, when D0<(-1,000), the value of (D0)-(-1,000) is stored in (D1).

- When -1,000≦ D0≦1,000, 0 is stored in D1.
- When D0<1,000, the value of (D0)-1,000 is stored in D1.

6.12.2 DBAND/BIN 32-bit data dead zone control

DBAND(P)

The input value (BIN 32-bit value) specified in (s3) controls the output value stored in the device specified in (d) according to the upper and lower limits of the dead zone specified in (s1) and (s2).

-[DBAND (s1) (s2) (S3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Dead zone low limit (no output zone)	-2,147,483,648 to +2,147,483,647	Signed BIN 32 bit	ANY32_S
(s2)	Dead zone high limit (no output zone)	-2,147,483,648 to +2,147,483,647	Signed BIN 32 bit	ANY32_S
(S3)	Input value controlled by dead zone control	-2,147,483,648 to +2,147,483,647	Signed BIN 32 bit	ANY32_S
(d)	The start number of the device that stores the output value controlled by the dead zone control	-	Signed BIN 32 bit	ANY32_S

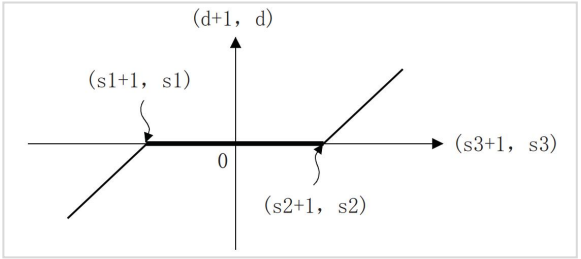
Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBAND	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3											●	●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 4												●	●	●	●	●	●	●	●	●	●				●	●

Features

The input value (BIN 32-bit value) specified in (s3) controls the output value stored in the device specified in (d) according to the upper and lower limits of the dead zone specified in (s1) and (s2). The output value is controlled as follows.

Condition	The value stored in the output value
When dead zone low limit ((s1), (s1)+1)> input value ((s3), (s3)+1)	Input value ((s3), (s3)+1)-dead zone low limit ((s1), (s1)+1)
When dead zone high limit ((s1), (s1)+1) <input value ((s3), (s3)+1)	Input value ((s3), (s3)+1)-dead zone high limit ((s2), (s2)+1)
When dead zone low limit ((s1), (s1)+1) ≤ input value ((s3), (s3)+1) ≤ dead zone high limit ((s2), (s2)+1)	0



• When the output value stored in (d) is a signed BIN 32-bit value, and the operation result exceeds the range of -2,147,483,648 to 2,147,483,647, the situation is as the following example. For example, When (s1) and (s1)+1 are 1000, (s3) and (s3)+1 are -2,147,483,648, then the output value=

-2,147,483,648-1000=80000000H-000003E8H=7FFFFC18H=2,147,482,648.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device.
4086H	When the specified device range for writing exceeds the range of the corresponding device.
4084H	When the low limit specified in (s1) is greater than the high limit specified in (s2).

Example



• When (D1, D0)<(-10,000), the value of (D1, D0)-(-10,000) is stored in (D11, D10).

• When -10,000≦(D1, D0)≦10,000, 0 is stored in (D11, D10).

• When 10,000< (D1, D0), the value of (D1, D0)-10,000 is stored in D1.

6.12.3 BINDA/BIN 16-bit data → Decimal ASCII conversion

BINDA(P)

Convert the BIN 16-bit data specified in (s) and the value of each digit in decimal numbers into ASCII codes and store them after the device number specified in (d).

-[BINDA(s)(d)]

Content, range and data type

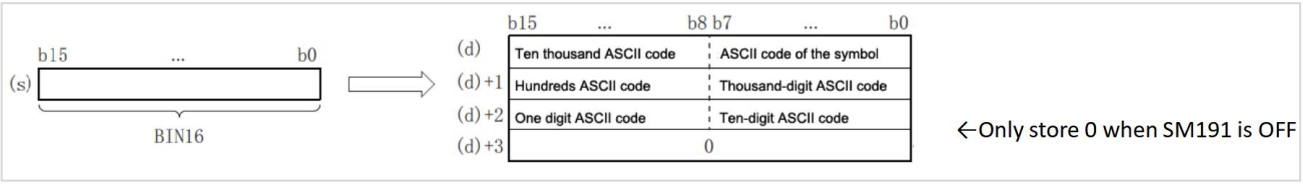
Parameter	Content	Range	Data type	Data type (label)
(s)	BIN data for ASCII conversion	-32768 to +32767	Signed BIN 16 bit	ANY16_S
(d)	The start number of the device storing the conversion result	-	String	ANYSTRING_SINGLE

Device used

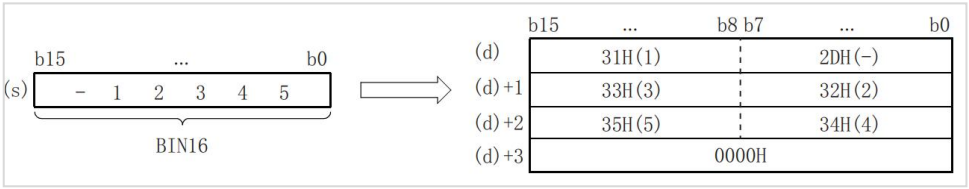
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BINDA	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2																	●	●	●						●	●

Features

Convert the BIN 16-bit data specified in (s) and the value of each digit in decimal numbers into ASCII codes and store them after the device number specified in (d).



For example, when -12,345 is specified in (s) (in the case of specifying signed)



The calculation result stored in (d) will be as below.

• In "Sign", 20H is stored when the BIN data is positive, and 2DH is stored when it is negative.

• In the 0 to the left of the effective digit, 20H is stored. (Suppress 0.) For example, in the case of "00325", "00" becomes 20H, and "325" becomes the effective digit.

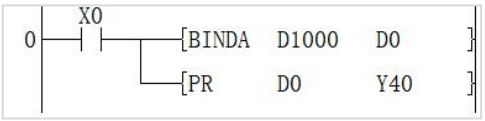
• When storing data to the device specified in (d)+3, when SM191 (output character number switching signal) is OFF, 0 is stored, and it does not change when it is ON.

✎**Note:** The number of occupied points of (d) is 3 when SM191 is ON, and it is 4 when SM191 is OFF.

Error code

Error code	Content
4085H	The read address of (s) exceeds the device range.
4086H	The write address of (d) exceeds the device range.

Example



When X000 is ON, convert the value of 16-bit data (BIN) D1000 into decimal ASCII code, and then use PR instruction to output the converted ASCII code character by character to the program of Y040 to Y057.

6.12.4 DBINDA/BIN 32-bit Data → Decimal ASCII conversion

DBINDA(P)

Convert the BIN 32-bit data specified in (s) and the value of each bit in decimal numbers into ASCII codes and store them after the device number specified in (d).

-[DBINDA(s)(d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	BIN data for ASCII conversion	-2,147483648 to 2147483647	Signed BIN 32 bit	ANY32_S
(d)	The start number of the device storing the conversion result	-	String	ANYSTRING_SINGLE

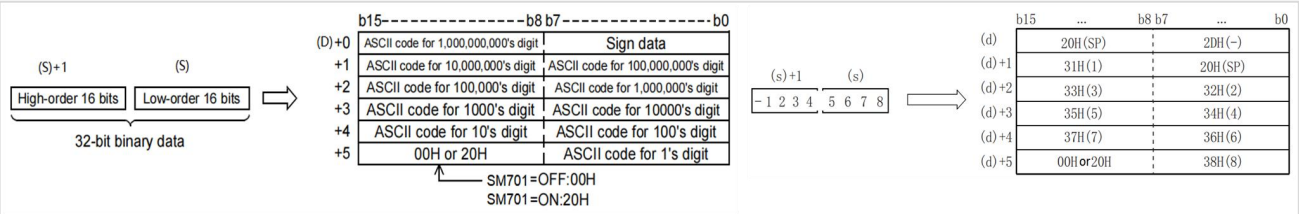
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBINDA	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2																	●	●	●						●	●

Features

Convert the BIN 32-bit data specified in (s) and the value of each bit when expressed in decimal numbers into ASCII codes, and store them after the device number specified in (d).

For example, when -12345678 is specified in (s). (in the case of specifying signed)



The calculation result stored in (d) will be as below.

• In "Sign", 20H is stored when the BIN data is positive, and 2DH is stored when it is negative.

• 20H is stored at 0 to the left of the effective number of digits. (Suppress 0.) For example, in the case of "0012034560", "00" becomes 20H, and "12034560" becomes effective digits.

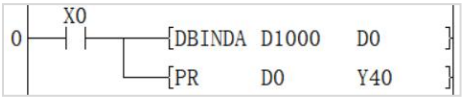
• For the data stored in the upper 8 bits of the device specified in (d)+5, 0 will be stored when SM191 (output character switching signal) is OFF, and 20H will be stored when it is ON.

✎**Note:** (d) Occupies 6 points.

Error code

Error code	Content
4085H	(s) read address exceeds the device range
4086H	(d) write address exceeds the device range

Example



When X000 is ON, convert the value of 32-bit data (BIN) D1000 into decimal ASCII code, and then use PR (FNC 77) instruction to output the converted ASCII code character by character to the program in Y040 to Y051 in time and time division.

6.12.5 DABIN/Decimal ASCII → BIN conversion

DABIN(P)

Digital ASCII code (30H to 39H) is a instruction to convert real data into BIN data.

-[DABIN (s) (d)]

Content, range and data type

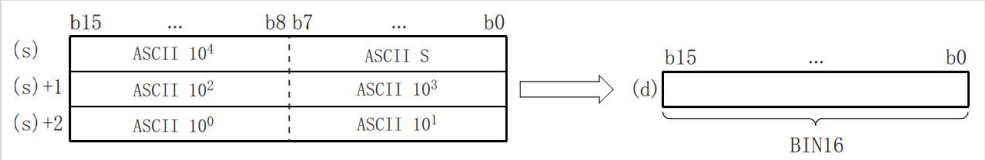
Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of the device that stores the data (ASCII code) to be converted into a BIN value	-	String	ANYSTRING_SINGLE
(d)	The device number for storing conversion result	-	BIN16 bit	ANY16_S

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DABIN	Parameter 1																	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

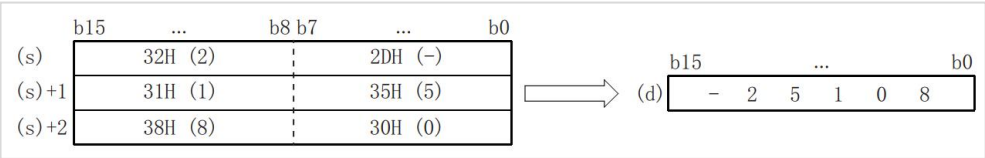
Features

The decimal ASCII data stored after the device number specified in (s) is converted into BIN 16-bit data and stored in the device specified in (d).



ASCII S	ASCII 10^0	ASCII 10^1	ASCII 10^2	ASCII 10^3	ASCII 10^4
ASCII code	Units of ASCII code	Tens of ASCII code	Hundreds of ASCII code	Thousands of ASCII code	Ten thousands of ASCII code

For example, When -25,108 is specified in (s)



• The ASCII data specified in (s) to (s)+2 is within the range of -32,768 to +32,767.

• In “Sign”, set 20H when the converted data is positive, and set 2DH when it is negative. (When other than 20H or 2DH is set, it will be treated as positive data. (DABIN(P))

• The range of the ASCII code set in each digit is 30H to 39H.

• When the ASCII code set in each bit is 20H or 00H, it will be treated as 30H.

Error code

Error code	Content
4084H	When the Sign data exceeds the range of 30H to 39H, 20H, 00H, 2DH; When the ASCII code of each bit specified in (s) to (s)+2 exceeds the range of 30H to 39H, 20H, 00H; When the ASCII data specified in (s) to (s)+2 is other than -32,768 to +32,767.
4085H	The read address of (s) exceeds the device range.
4086H	The write address of (d) exceeds the device range.

Example



When X000 is ON, the Signs set in D20 to D22 and the ASCII code data of 5-digit decimal numbers are converted into BIN values, and then stored in the program of D0.

6.12.6 DDABIN/Decimal ASCII → BIN32-bit data conversion

DDABIN(P)

The decimal ASCII data stored after the device number specified in (s) is converted into BIN 32-bit data and stored in the device number specified in (d).

-[DDABIN (s) (d)]

Content, range and data type

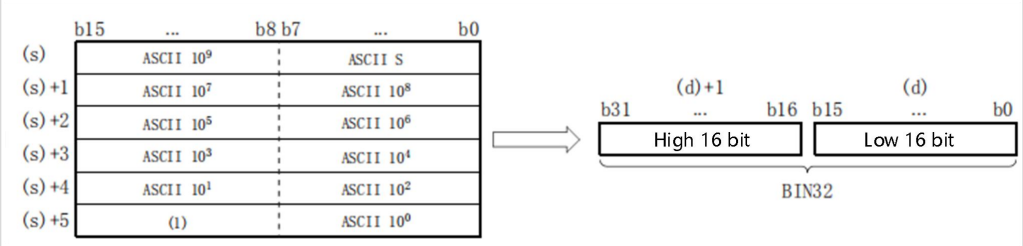
Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of the device that stores data (ASCII code) to be converted into a BIN value	-	String	ANYSTRING_SINGLE
(d)	The device number for storing conversion result	-	Signed BIN 32 bit	ANY32_S

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DDABIN	Parameter 1																	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●	●	●				●	●

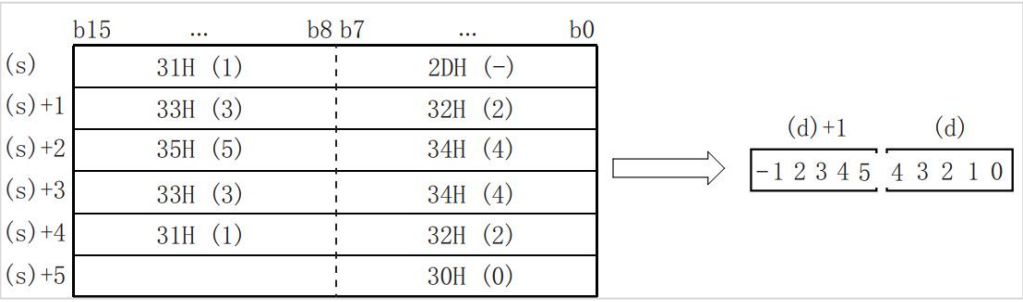
Features

The decimal ASCII data stored after the device number specified in (s) is converted into BIN 32-bit data and stored in the device specified in (d).



ASCII S	ASCII code	ASCII S	ASCII code
ASCII 10^0	Units of ASCII code	ASCII 10^5	Hundred thousands of ASCII code
ASCII 10^1	Tens of ASCII code	ASCII 10^6	Millions of ASCII code
ASCII 10^2	Hundreds of ASCII code	ASCII 10^7	Ten millions of ASCII code
ASCII 10^3	Thousands of ASCII code	ASCII 10^8	Hundred millions of ASCII code
ASCII 10^4	Tens thousands of ASCII code	ASCII 10^9	billions of ASCII code

When -1234543210 is specified in (s) (When signed is specified)



• The ASCII data specified in (s) to (s)+5 is within the range of -2,147,483,648 to +2,147,483,647. In addition, the data stored in the high byte of (s)+5 will be ignored.

• In the Sign data, set 20H when the converted data is positive, and set 2DH when it is negative. (When other than 20H or 2DH is set, it will be treated as positive data. (DABIN(P))

• The range of ASCII code set in each digit is 30H to 39H.

• When the ASCII code set in each bit is 20H or 00H, it will be treated as 30H.

Error code

Error code	Content
4084H	When the Sign data exceeds the range of 30H to 39H, 20H, 00H and 2DH; When the ASCII code of each bit specified in (s) to (s)+5 exceeds the range of 30H to 39H, 20H and 00H; When the ASCII data specified in (s) to (s)+5 exceeds the range of -2,147,483,648 to +2,147,483,647
4085H	The read address of (s) exceeds the device range.
4086H	The write address of (d) exceeds the device range.

Example



When X000 is ON, the Signs set in to D20 to D25 and the ASCII code data of 10-digit decimal numbers are converted into BIN values and then saved to the program in D0 to D1.

6.12.7 LIMIT/ BIN 16-bit data high and low limit control

LIMIT(P)

The input value (BIN 16-bit value) specified in (s3) controls the output value stored in the device specified in (d) according to the upper and lower limit value ranges specified in (s1) and (s2).

-[LIMIT (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
s1	Low limit value (minimum output limit value)	-32,768 to 32,767	BIN16 bit	ANY16_S
s2	High limit value (maximum output limit value)	-32,768 to 32,767	BIN16 bit	ANY16_S
s3	Input value controlled by high and low limit control	-32,768 to 32,767	BIN16 bit	ANY16_S
d	The start number of device that stores the output value controlled by high and low limit control	-	BIN16 bit	ANY16_S

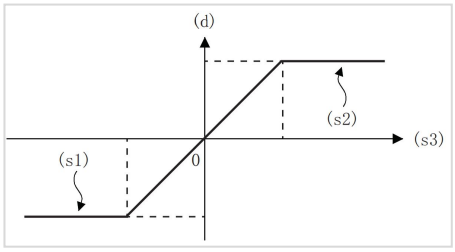
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LIMIT	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3											●	●	●	●	●	●	●	●	●						●	●
	Parameter 4												●	●	●	●	●	●	●	●						●	●

Features

The input value (BIN 16-bit value) specified in (s3) controls the output value stored in the device specified in (d) according to the high and low limit value ranges specified in (s1) and (s2). The output value is controlled as follows.

Condition	The value stored in the output value
Low limit value (s1)>input value (s3)	Low limit value (s1)
High limit value (s1) <input value (s3)	High limit value (s2)
Low limit value (s1) ≤ input value (s3) ≤ high limit value (s2)	Input value (s3)



•Only in the case of controlling high limit value, set the minimum value of data range in the low limit value specified in (s1).

• Only in the case of controlling low limit value, set the maximum value of data range in the high limit value specified in (s2).

Error code

Error code	Content
4085H	The read address exceeds the device range
4086H	The write address exceeds the device range
4084H	High limit <low limit

Example



When X000 is ON

- When D0 <500, D1 is 500.
- When 500 ≤ D0 ≤ 5,000, D1 is the value of D0.
- When 5,000 <D0, D1 is 5,000.

6.12.8 DLIMIT/BIN 32-bit data high and low limit control

DLIMIT(P)

The input value (BIN 32-bit value) specified in (s3) controls the output value stored in the device specified in (d) according to the range of high and low limit values specified in (s1) and (s2).

-[DLIMIT (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
s1	Low limit value (minimum output limit value)	-2,147,483,648 to 2,147,483,647	BIN32 bit	ANY32_S
s2	High limit value (maximum output limit value)	-2,147,483,648 to 2,147,483,647	BIN32 bit	ANY32_S
s3	Input value controlled by high and low limit control	-2,147,483,648 to 2,147,483,647	BIN32 bit	ANY32_S
d	The start number of the device that stores the output value controlled by high and low limit control	-	BIN32 bit	ANY32_S

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DLIMIT	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3											●	●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 4												●	●	●	●	●	●	●	●	●	●				●	●

Features

The input value (BIN 32-bit value) specified in (s3) controls the output value stored in the device specified in (d) according to the range of high and low limit values specified in (s1) and (s2). The output value is controlled as follows.

Condition	The value stored in the output value
Low limit value ((s1), (s1)+1)> input value ((s3), (s3)+1)	Low limit value ((s1), (s1)+1)
High limit value ((s2), (s2)+1) <input value ((s3), (s3)+1)	High limit value ((s2), (s2)+1)
Low limit value ((s1), (s1)+1) ≤ input value ((s3), (s3)+1) ≤ high limit value ((s2), (s2)+1)	Input value ((s3), (s3)+1)

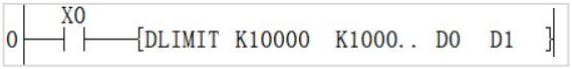
•Only in the case of controlling high limit value, set the minimum value of data range in the low limit value specified in (s1).

• Only in the case of controlling low limit value, set the maximum value of data range in the high limit value specified in (s2).

Error code

Error code	Content
4085H	The read address exceeds the device range
4086H	The write address exceeds the device range
4084H	High limit <low limit

Example



Operation:

• When (D1, D0) <10,000, (D11, D10) is 10,000.

• When 10,000 ≤ (D1, D0) ≤ 1,000,000, (D11, D10) is the value of (D1, D0).

• When 1,000,000 <(D1, D0), (D11, D10) is 1,000,000.

6.12.9 SCL/BIN 16-bit unit scale (coordinate data of each point)

SCL(P)

The scaling conversion data (16-bit data unit) specified in (s2) is scaled from the input value specified in (s1), and the calculation result is stored in the device specified in (d).

-[SCL (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The input value for scaling or the start number of device storing the input value	-32,768 to 32,767	Signed BIN 16 bit	ANY16_S
(s2)	The start number of the device storing conversion data for scaling	-	Signed BIN 16 bit	ANY16_S
(d)	The start number of the device that stores the output value controlled by scaling	-	Signed BIN 16 bit	ANY16_S

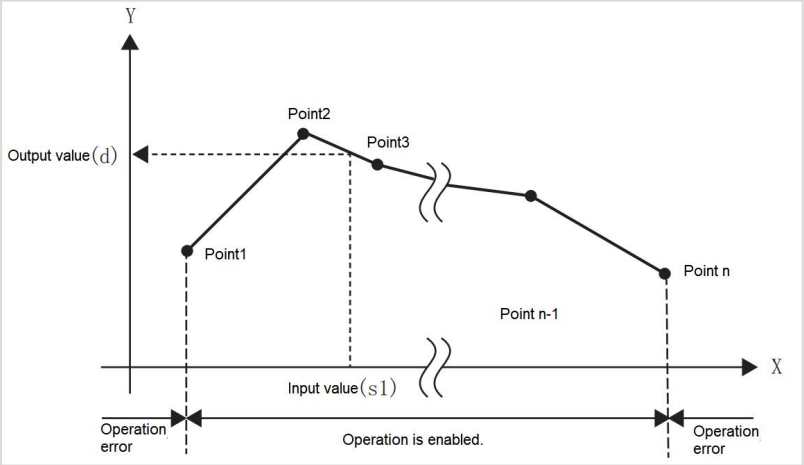
Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SCL	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2																	●	●	●						●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

Features

For the scale conversion data (16-bit data unit) specified in (s2), scale by the input value specified in (s1), and store the operation result in the device specified in (d). Scale conversion is performed based on the scale conversion data stored after the device specified in (s2).

Setting items (n represents the number of coordinate points specified in (s2))		Device allocation
Coordinate points		(s2)
Point 1	X coordinate	(s2)+1
	Y coordinate	(s2)+2
Point 2	X coordinate	(s2)+3
	Y coordinate	(s2)+4
.....		
Point n	X coordinate	(s2)+2n-1
	Y coordinate	(s2)+2n



• If the operation result is not an integer value, round the first digit below the decimal point.

• The X coordinate data of the conversion data for scaling should be set in ascending order.

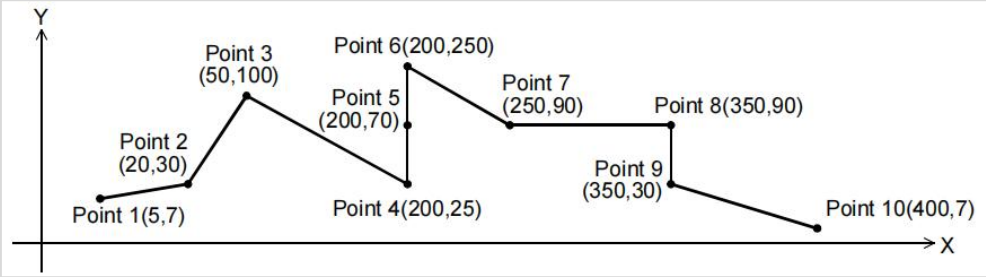
• (s1) should be set within the range of conversion data for scaling (device value of (s2)).

• If the same X coordinate is specified for multiple points, the Y coordinate value of the second point will be output.

• Set the number of coordinate points of the conversion data for scaling within the range of 1 to 32,767.

• Setting example of conversion table for scaling.

In the case of scaling conversion characteristics as shown in the figure below, set it as the following data sheet.

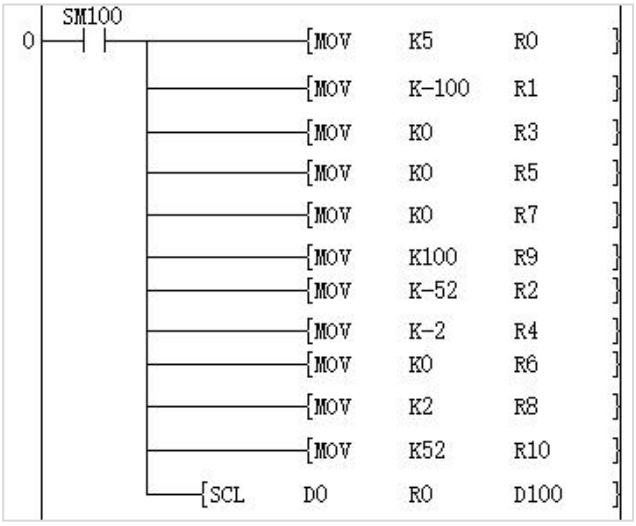


Set items		Sett device and content			Remarks
		When R0 is specified in (s2)		Set content	
Coordinate points		(s2)	R0	K10	
Point 1	X coordinate	(s2)+1	R1	K5	
	Y coordinate	(s2)+2	R2	K20	
Point 2	X coordinate	(s2)+3	R3	K30	
	Y coordinate	(s2)+4	R4	K50	
Point 3	X coordinate	(s2)+5	R5	K100	
	Y coordinate	(s2)+6	R6	K200	
Point 4	X coordinate	(s2)+7	R7	K25	If the coordinates are specified by 3 points, the intermediate value could be the output value. In this example, the output value (median value) is specified by the Y coordinate of point 5. When the X coordinate is the same at 3 points or more, the value of the 2nd point is also output.
	Y coordinate	(s2)+8	R8	K200	
Point 5	X coordinate	(s2)+9	R9	K70	
	Y coordinate	(s2)+10	R10	K200	
Point 6	X coordinate	(s2)+11	R11	K250	
	Y coordinate	(s2)+12	R12	K250	
Point 7	X coordinate	(s2)+13	R13	K250	
	Y coordinate	(s2)+14	R14	K90	
Point 8	X coordinate	(s2)+15	R15	K350	If the coordinates are specified by two points, the output value is the value of the Y coordinate of the next point. In this example, the output value is specified by the Y coordinate of point 9.
	Y coordinate	(s2)+16	R16	K90	
Point 9	X coordinate	(s2)+17	R17	K350	
	Y coordinate	(s2)+18	R18	K30	
Point 10	X coordinate	(s2)+19	R19	K400	
	Y coordinate	(s2)+20	R20	K7	

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device.
4086H	When the specified device range for writing exceeds the range of the corresponding device.
4084H	When the Xn data of data table is not sorted in ascending order. However, the instruction will be executed until the position where the error occurs; When the input value specified in (s1) exceeds the range of the set scale conversion data; When the number of start coordinate points of device (s2) is less than 0.

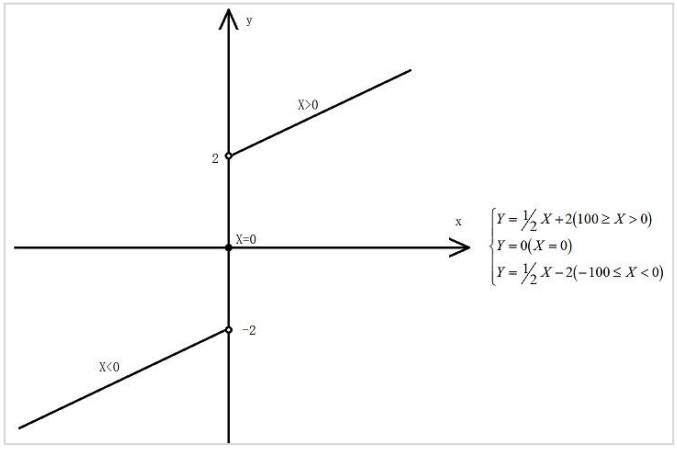
Example



When -100 ≤ D0(X) < 0, D100(Y)= $\frac{1}{2}X-2$

when D0(X)=0, D100(Y)=0;

when 0< D0(X) ≤ 100, D100(Y)= $\frac{1}{2}X+2$



6.12.10 DSCL/32-bit unit scale (coordinate data of each point)

DSCL(P)

The conversion data (32-bit data unit) for scaling specified in (s2) is scaled by the input value specified in (s1), and the operation result is stored in the device specified in (d).

-[DSCL (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The input value for scaling or the start number of the device storing the input value	-2,147,483,648 to 2,147,483,647	Signed BIN 32 bit	ANY32_S
(s2)	The start number of the device storing conversion data for scaling	-	Signed BIN 32 bit	ANY32_S
(d)	The start number of the device that stores the output value controlled by scaling	-	Signed BIN 32 bit	ANY32_S

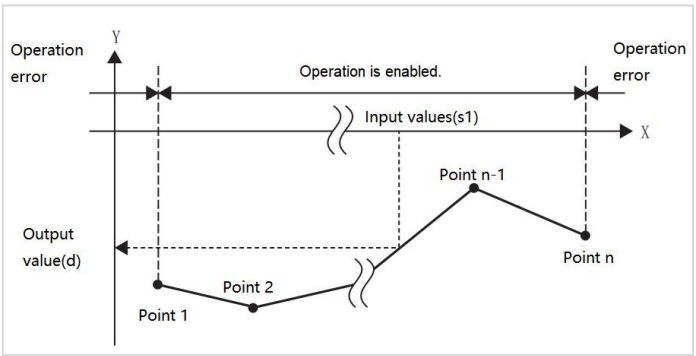
Device used

instruction	Parameter	Device																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSCL	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2																	●	●	●						●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

The conversion data (32-bit data unit) for scaling specified in (s2) is scaled by the input value specified in (s1), and the operation result is stored in the device number specified in (d). The scale conversion is performed based on the scale conversion data stored after the device specified in (s2).

Set items (n represents the number of coordinate points specified in (s2))		Device allocation
Coordinate points		(s2)+1, (s2)
Point 1	X coordinate	(s2)+3, (s2)+2
	Y coordinate	(s2)+5, (s2)+4
Point 2	X coordinate	(s2)+7, (s2)+6
	Y coordinate	(s2)+9, (s2)+8
.....		
Point n	X coordinate	(s2)+4n-1, (s2)+4n-2
	Y coordinate	(s2)+4n+1, (s2)+4n



• If the calculation result is not an integer value, round the first digit below the decimal point.

• The X coordinate data of the conversion data for scaling should be set in ascending order.

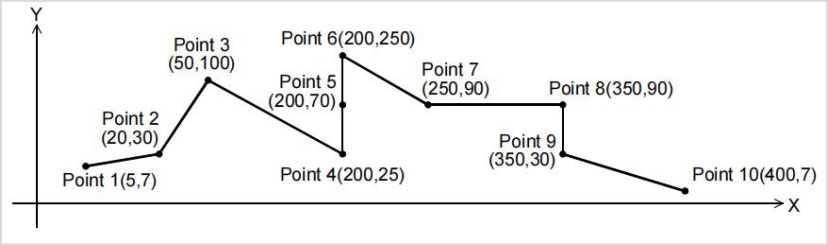
• For (s1), set within the range of the conversion data for scaling ((s2), (s2) + 1 device value).

• If the same X coordinate is specified for multiple points, the Y coordinate value of the second point will be output.

• Set the number of coordinate points of conversion data for scaling within the range of 1 to 2,147,483,647.

• Setting example of conversion table for scaling.

In the case of scaling conversion characteristics as shown in the figure below, set it as the following data sheet.

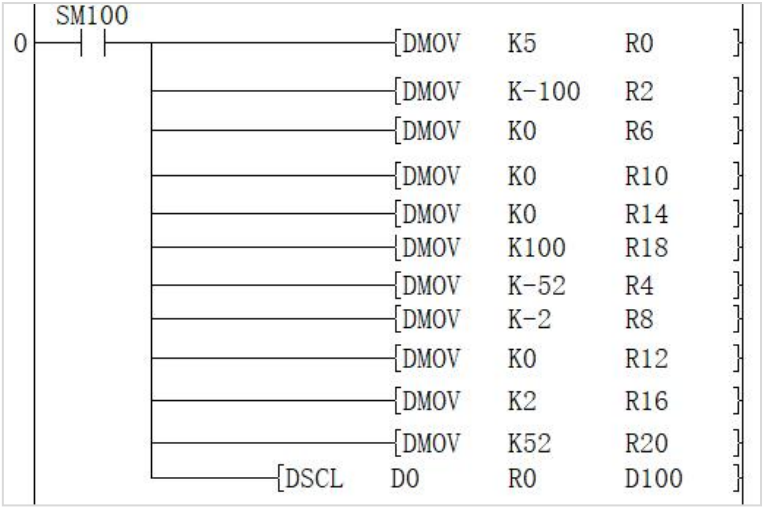


Set items		Set device and content			Remarks
		When R0 is specified in (s2)		Set content	
Coordinate points		(s2)+1, (s2)	R1, R0	K10	
Point 1	X coordinate	(s2)+3, (s2)+2	R3, R2	K5	
	Y coordinate	(s2)+5, (s2)+4	R5, R4	K7	
Point 2	X coordinate	(s2)+7, (s2)+6	R7, R6	K20	
	Y coordinate	(s2)+9, (s2)+8	R9, R8	K30	
Point 3	X coordinate	(s2)+11, (s2)+10	R10, R11	K50	
	Y coordinate	(s2)+13, (s2)+12	R13, R12	K100	
Point 4	X coordinate	(s2)+15, (s2)+14	R15, R14	K200	if the coordinates are specified by 3 points, the intermediate value could be the output value. In this example, the output value (median value) is specified by the Y coordinate of point 5. When the X coordinate is the same at 3 points or more, the value of the 2nd point is also output.
	Y coordinate	(s2)+17, (s2)+16	R17, R16	K25	
Point 5	X coordinate	(s2)+19, (s2)+18	R19, R18	K200	
	Y coordinate	(s2)+21, (s2)+20	R21, R20	K70	
Point 6	X coordinate	(s2)+23, (s2)+22	R23, R22	K200	
	Y coordinate	(s2)+25, (s2)+24	R25, R24	K250	
Point 7	X coordinate	(s2)+27, (s2)+26	R27, R26	K250	
	Y coordinate	(s2)+29, (s2)+28	R29, R28	K90	
Point 8	X coordinate	(s2)+31, (s2)+30	R31, R30	K350	If the coordinates are specified by two points, the output value is the value of the Y coordinate of the next point. In this example, the output value is specified by the Y coordinate of point 9.
	Y coordinate	(s2)+33, (s2)+32	R33, R32	K90	
Point 9	X coordinate	(s2)+35, (s2)+34	R35, R34	K350	
	Y coordinate	(s2)+37, (s2)+36	R37, R36	K30	
Point 10	X coordinate	(s2)+39, (s2)+38	R39, R38	K400	
	Y coordinate	(s2)+41, (s2)+40	R41, R40	K7	

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device.
4086H	When the specified device range for writing exceeds the range of the corresponding device.
4084H	When the Xn data of data table is not sorted in ascending order. However, the instruction will be executed until the position where the error occurs; When the input value specified in (s1) exceeds the range of the set scale conversion data; When the number of start coordinate points of device (s2) is less than 0.

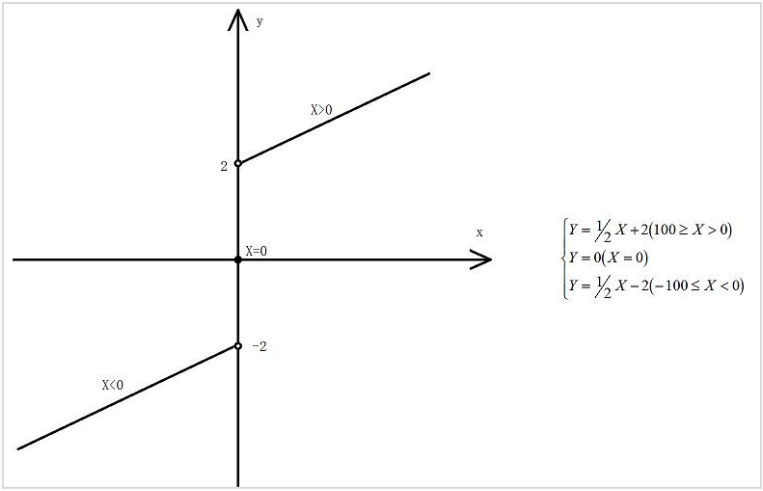
Example



When -100 ≤ D0(X) < 0, D100(Y)= $\frac{1}{2}X-2$

When D0(X)=0, D100(Y)=0;

When 0 < D0(X) ≤ 100, D100(Y)= $\frac{1}{2}X+2$



6.12.11 SCL2/BIN 16-bit unit scale (x/y coordinate data)

SCL2(P)

The conversion data (16-bit data unit) for scaling specified in (s2) is scaled by the input value specified in (s1), and the operation result is stored in the device specified in (d).

-[SCL2 (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The input value for scaling or the start number of the device storing the input value	-32,768 to 32,767	Signed BIN 16 bit	ANY16_S
(s2)	The start number of the device storing conversion data for scaling	-	Signed BIN 16 bit	ANY16_S
(d)	The start number of the device that stores the output value controlled by scaling	-	Signed BIN 16 bit	ANY16_S

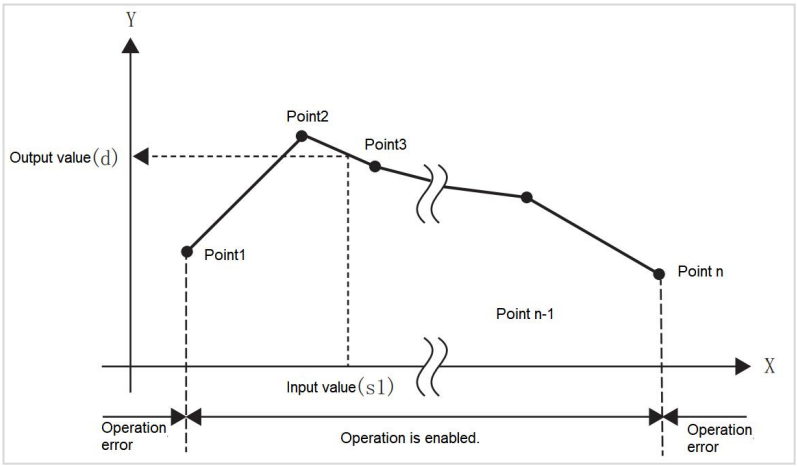
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SCL2	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2																	●	●	●						●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

Features

The conversion data (16-bit data unit) for scaling specified in (s2) is scaled by the input value specified in (s1), and the operation result is stored in the device number specified in (d). The scale conversion is performed based on the scale conversion data stored after the device specified in (s2).

Set items (n represents the number of coordinate points specified in (s2))		Device allocation
Coordinate points		(s2)
X coordinate	Point 1	(s2)+1
	Point 2	(s2)+2
	.....	.....
	Point n	(s2)+n
Y coordinate	Point 1	(s2)+n+1
	Point 2	(s2)+n+2
	.....	.....
	Point n	(s2)+2n



• If the operation result is not an integer value, round the first digit below the decimal point.

• The X coordinate data of the conversion data for scaling should be set in ascending order.

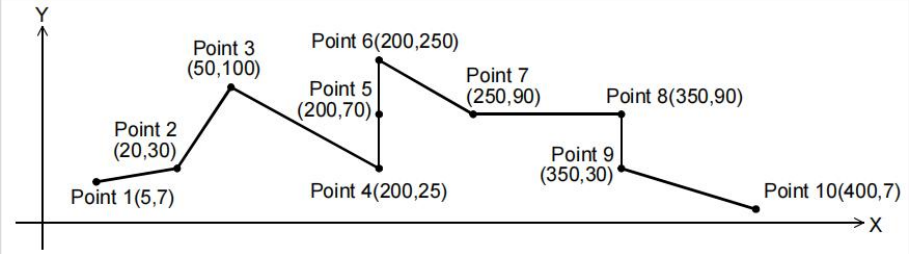
• For (s1), set within the range of the conversion data for scaling ((s2), (s2) + 1 device value).

• If the same X coordinate is specified for multiple points, the Y coordinate value of the second point will be output.

• Set the number of coordinate points of conversion data for scaling within the range of 1 to 32,767.

• Setting example of conversion table for scaling.

In the case of scaling conversion characteristics as shown in the figure below, set it as the following data sheet.



Set items		Set device and content			Remarks
		When R0 is specified in (s2)		Set content	
Coordinate points		(s2)	R0	K10	
X coordinate	Point 1	(s2)+1	R1	K5	
	Point 2	(s2)+2	R2	K20	
	Point 3	(s2)+3	R3	K50	
	Point 4	(s2)+4	R4	K200	Refer to ✎1
	Point 5	(s2)+5	R5	K200	
	Point 6	(s2)+6	R6	K200	
	Point 7	(s2)+7	R7	K250	
	Point 8	(s2)+8	R8	K350	Refer to ✎2
	Point 9	(s2)+9	R9	K350	
	Point 10	(s2)+10	R10	K400	
Y coordinate	Point 1	(s2)+11	R11	K7	
	Point 2	(s2)+12	R12	K30	
	Point 3	(s2)+13	R13	K100	
	Point 4	(s2)+14	R14	K25	Refer to✎1
	Point 5	(s2)+15	R15	K70	
	Point 6	(s2)+16	R16	K250	
	Point 7	(s2)+17	R17	K90	
	Point 8	(s2)+18	R18	K90	Refer to ✎2
	Point 9	(s2)+19	R19	K30	
	Point 10	(s2)+20	R20	K7	

✎**1** Like points 4, 5, and 6, if the coordinates are specified by 3 points, the intermediate value could be the output value.

In this example, the output value (median value) is specified by the Y coordinate of point 5.

When the X coordinate is the same at 3 or more points, the value of the second point is also output.

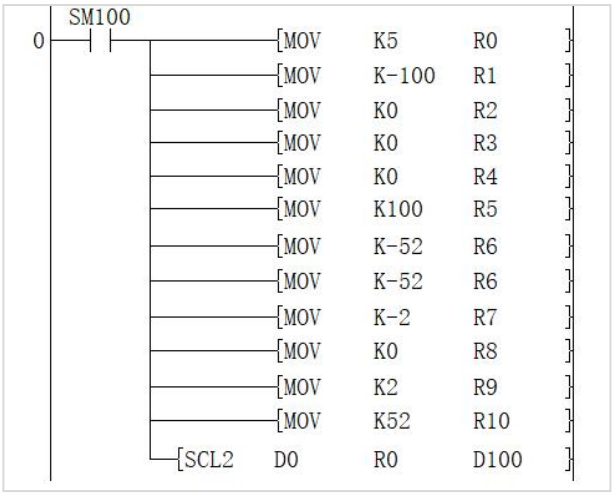
✎**2** Like points 8 and 9, if the coordinates are specified by 2 points, the output value is the value of the Y coordinate of next point.

In this example, the output value is specified by the Y coordinate of point 9.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device.
4086H	When the specified device range for writing exceeds the range of the corresponding device.
4084H	When the Xn data of data table is not sorted in ascending order. However, the instruction will be executed until the position where the error occurs; When the input value specified in (s1) exceeds the range of the set scale conversion data; When the number of start coordinate points of device (s2) is less than 0.

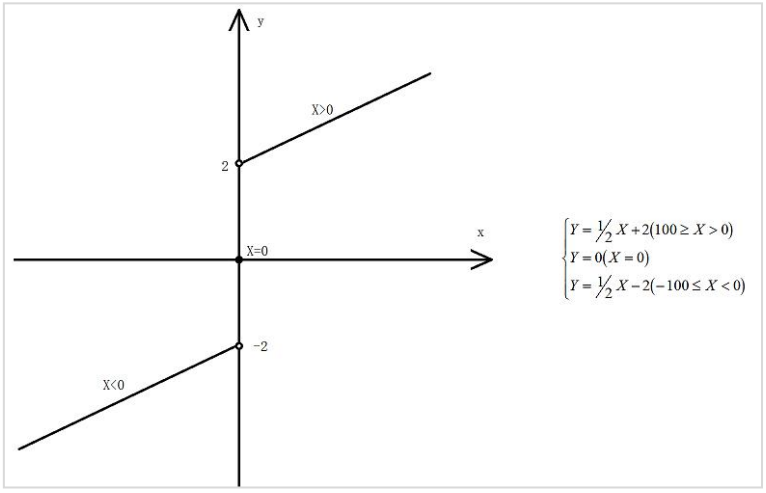
Example



When -100 ≤D0 (X) < 0, D100(Y)= $\frac{1}{2}X-2$

when D0(X)=0, D100(Y)=0;

when 0 < D0(X) ≤ 100, D100(Y)= $\frac{1}{2}X+2$



6.12.12 DSCL2/BIN 32-bit unit scale (x/y coordinate data)

DSCL2(P)

The conversion data (32-bit data unit) for scaling specified in (s2) is scaled by the input value specified in (s1), and the operation result is stored in the device specified in (d).

-[DSCL2 (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The input value for scaling or the start number of the device storing the input value	-2,147,483,648 to +2,147,483,647	Signed BIN 32 bit	ANY32_S
(s2)	The start number of the device storing conversion data for scaling	-	Signed BIN 32 bit	ANY32_S
(d)	The start number of the device that stores the output value controlled by scaling	-	Signed BIN 32 bit	ANY32_S

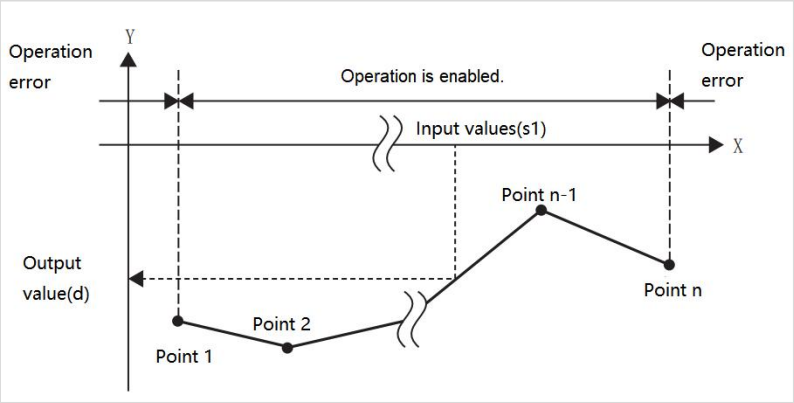
Device used

Instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSCL2	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2																	●	●	●						●	●
	Parameter 3												●	●	●	●	●	●	●	●	●	●				●	●

Features

The conversion data (32-bit data unit) for scaling specified in (s2) is scaled by the input value specified in (s1), and the operation result is stored in the device number specified in (d). The scale conversion is performed based on the scale conversion data stored after the device specified in (s2).

Set items (n represents the number of coordinate points specified in (s2))		Device allocation
Coordinate points		(s2)+1, (s2)
X coordinate	Point 1	(s2)+3, (s2)+2
	Point 2	(s2)+5, (s2)+4
	.....	.....
	Point n	(s2)+2n+1, (s2)+2n
Y coordinate	Point 1	(s2)+2n+3, (s2)+2n+2
	Point 2	(s2)+2n+5, (s2)+2n+4
	.....	.....
	Point n	(s2)+4n+1, (s2)+4n



• If the operation result is not an integer value, round the first digit below the decimal point.

• The X coordinate data of the conversion data for scaling should be set in ascending order.

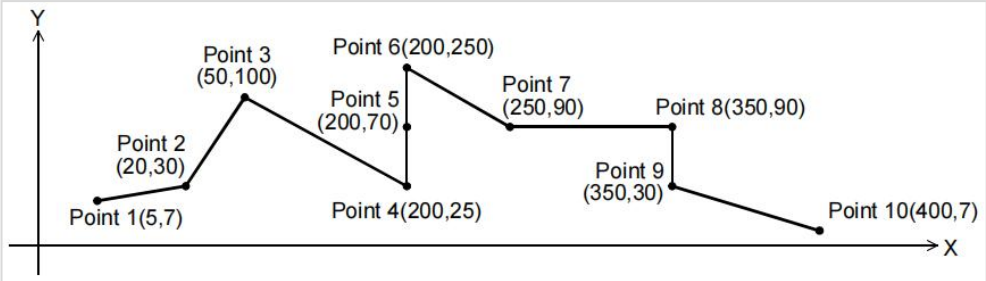
• For (s1), set within the range of the conversion data for scaling ((s2), (s2) + 1 device value).

• If the same X coordinate is specified for multiple points, the Y coordinate value of the second point will be output.

• Set the number of coordinate points of conversion data for scaling within the range of 1 to 2,147,483,647.

• Setting example of conversion table for scaling.

In the case of scaling conversion characteristics as shown in the figure below, set it as the following data sheet.



Set items		Set device and content			Remarks
		When R0 is specified in (s2)		Set content	
Coordinate points		(s2)+1, (s2)	R1, R0	K10	
X	Point 1	(s2)+3, (s2)+2	R3, R2	K5	
	Point 2	(s2)+5, (s2)+4	R5, R4	K20	
	Point 3	(s2)+7, (s2)+6	R7, R6	K50	
	Point 4	(s2)+9, (s2)+8	R9, R8	K200	Refer to ✎1
	Point 5	(s2)+11, (s2)+10	R10, R11	K200	
	Point 6	(s2)+13, (s2)+12	R13, R12	K200	
	Point 7	(s2)+15, (s2)+14	R15, R14	K250	
	Point 8	(s2)+17, (s2)+16	R17, R16	K350	Refer to ✎2
	Point 9	(s2)+19, (s2)+18	R19, R18	K350	
	Point 10	(s2)+21, (s2)+20	R21, R20	K400	

Set items		Set device and content			Remarks
		When R0 is specified in (s2)		Set content	
Y coordinate	Point 1	(s2)+23, (s2)+22	R23, R22	K7	
	Point 2	(s2)+25, (s2)+24	R25, R24	K30	
	Point 3	(s2)+27, (s2)+26	R27, R26	K100	
	Point 4	(s2)+29, (s2)+28	R29, R28	K25	Refer to ✎1
	Point 5	(s2)+31, (s2)+30	R31, R30	K70	
	Point 6	(s2)+33, (s2)+32	R33, R32	K250	
	Point 7	(s2)+35, (s2)+34	R35, R34	K90	
	Point 8	(s2)+37, (s2)+36	R37, R36	K90	Refer to ✎2
	Point 9	(s2)+39, (s2)+38	R39, R38	K30	
	Point 10	(s2)+41, (s2)+40	R41, R40	K7	

✎**1** Like points 4, 5, and 6, if the coordinates are specified by 3 points, the intermediate value could be the output value.

In this example, the output value (median value) is specified by the Y coordinate of point 5.

When the X coordinate is the same at 3 or more points, the value of the second point is also output.

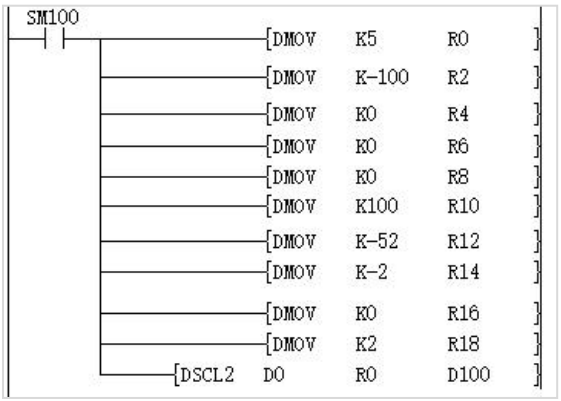
✎**2** Like points 8 and 9, if the coordinates are specified by 2 points, the output value is the value of the Y coordinate of the next point.

In this example, the output value is specified by the Y coordinate of point 9.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device.
4086H	When the specified device range for writing exceeds the range of the corresponding device.
4084H	When the Xn data of data table is not sorted in ascending order. However, the instruction will be executed until the position where the error occurs; When the input value specified in (s1) exceeds the range of the set scale conversion data; When the number of start coordinate points of device (s2) is less than 0.

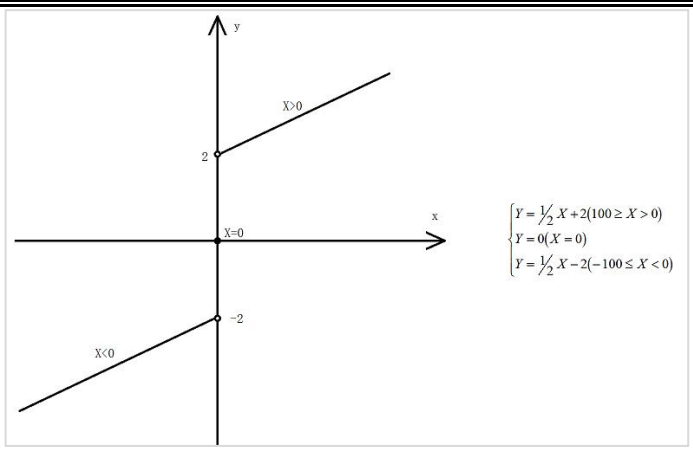
Example



When -100 ≤D0(X) < 0, D100(Y)= $\frac{1}{2}X-2$;

when D0(X)=0, D100(Y)=0;

when 0< D0(X) ≤ 100, D100(Y)= $\frac{1}{2}X+2$.


y
X>0
2
x
X=0
-2
X<0
$\begin{cases} Y = \frac{1}{2}X + 2(100 \geq X > 0) \\ Y = 0(X = 0) \\ Y = \frac{1}{2}X - 2(-100 \leq X < 0) \end{cases}$

6.12.13 ZONE/BIN 16-bit data zone control

ZONE(P)

After adding the offset value specified in (s1) or (s2) to the input value specified in (s3), it is stored in the device number specified in (d).

-[ZONE (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
s1	The negative offset value for addition in the input value	-32,768 to 32,767	BIN16 bit	ANY16_S
s2	The positive offset value for addition in the input value	-32,768 to 32,767	BIN16 bit	ANY16_S
s3	Input value for zone control`	-32,768 to 32,767	BIN16 bit	ANY16_S
d	The start number of the device storing the output value controlled by zone control	-	BIN16 bit	ANY16_S

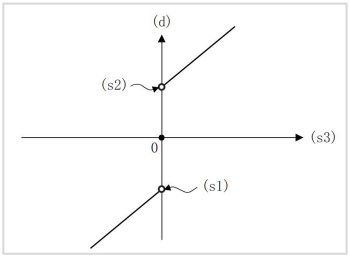
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ZONE	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3											●	●	●	●	●	●	●	●	●						●	●
	Parameter 4												●	●	●	●	●	●	●	●						●	●

Features

After adding the offset value specified in (s1) or (s2) to the input value (BIN 16-bit value) specified in (s3), it is stored in the device number specified in (d). The offset value is controlled as follows.

Condition	The value stored in the output value
When input value (s3)<0	Input value (s3) + negative offset value (s1)
When input value (s3)=0	0
When input value (s3)>0	Input value (s3) + positive offset value (s2)



• When the output value stored in (d) is a signed BIN 16-bit value, and the operation result exceeds the range of -32,768 to 32,767, the situation is shown in the following example.

For example, when (s1) is -100 and (s3) is -32,768,

the output value = -32768+(-100)=8000H-FF9CH=7F9CH=32668.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X000 is ON

- When D0<0, the value of (D0)+(-1,000) is stored in D1.
- When D0=0, 0 is stored in D1.
- When 0<D0, the value of (D0)+(1,000) is stored in D1.

6.12.14 DZONE/BIN 32-bit data zone control

DZONE(P)

After adding the offset value specified in (s1) or (s2) to the input value specified in (s3), it is stored in the device number specified in (d).

-[DZONE (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The negative offset value for addition in the input value	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s2)	The positive offset value for addition in the input value	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s3)	Input value for zone control	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(d)	The start number of the device storing the output value controlled by zone control	-	BIN32 bit	ANY32_S

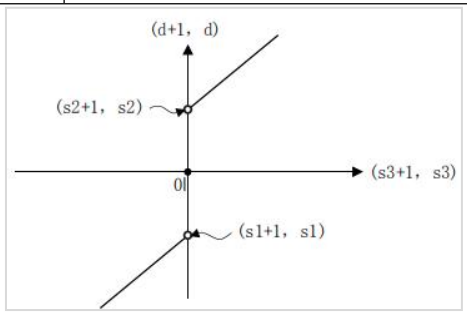
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DZONE	Parameter 1											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3											●	●	●	●	●	●	●	●	●	●	●				●	●
	Parameter 4												●	●	●	●	●	●	●	●	●	●				●	●

Features

After adding the offset value specified in (s1) or (s2) to the input value (BIN 32-bit value) specified in (s3), it is stored in the device number specified in (d). The offset value is controlled as follows.

Condition	The value stored in the output value
When input value ((s3), (s3)+1)<0	Input value ((s3), (s3)+1) + negative offset value (s1), (s1)+1
When input value ((s3), (s3)+1)=0	0
When input value ((s3), (s3)+1)>0	Input value ((s3), (s3)+1) + positive offset value (s2), (s2)+1



• When the output value stored in (d) and (d)+1 is a signed BIN 32-bit value, and the operation result exceeds the range of -2,147,483,648 to 2,147,483,647, the situation is shown in the following example.

For example, (s1), (s1)+1 is -1,000, (s3), (s3)+1 is -2,147,483,648, then the output value=-2,147,483,648+(-1000)=80000000H+FFFFFC18H=2,147,482,648.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



- When (D1, D0)<0, the value of (D1, D0)+(-10,000) is stored in (D11, D10).
- When (D1, D0)=0, 0 is stored in (D11, D10).
- When 0<(D1, D0), the value of (D1, D0)+10,000 is stored in (D11, D10).

6.13 Data block instructions

6.13.1 BK+/BIN 16-bit block data addition operation

BK+(P)

Add the BIN 16-bit data of point (n) starting from the device specified in (s1) and the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[BK+ (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
s1	The start device that stores the addition operation data	-32,768 to +32,767	BIN16 bit	ANY16_S
s2	Addition data or the starting device that stores the addition data	-32,768 to +32,767	BIN16 bit	ANY16_S
d	The start device that stores the addition operation result	-	BIN16 bit	ANY16_S
n	The number of addition operation data	0 to 65,535	BIN16 bit	ANY16

Device used

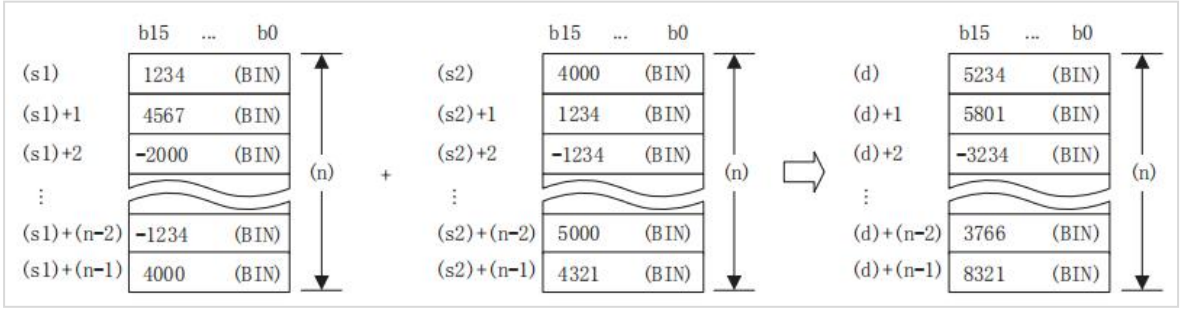
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BK+	Parameter 1															●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●			●	●		●	●
	Parameter 3															●	●	●	●	●						●	●
	Parameter 4																	●	●	●			●	●			●

Features

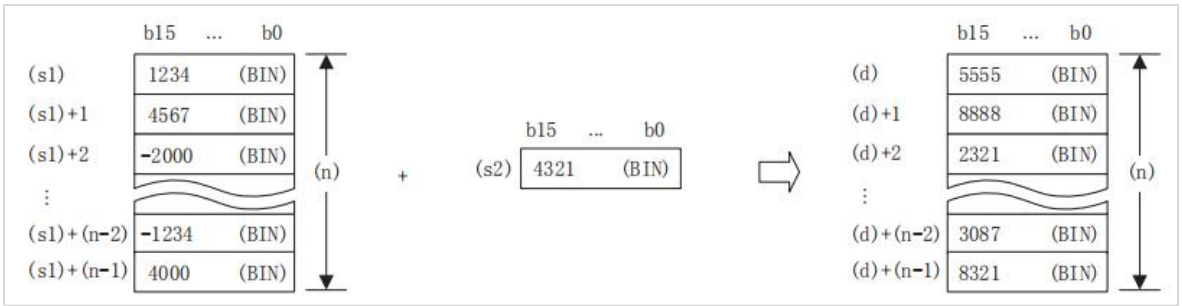
Add the BIN 16-bit data of point (n) starting from the device specified in (s1) and the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the result in the device specified in (d).

• Block addition operations are performed in 16-bit units.

When a device is specified in (s2) (when specified with a sign)

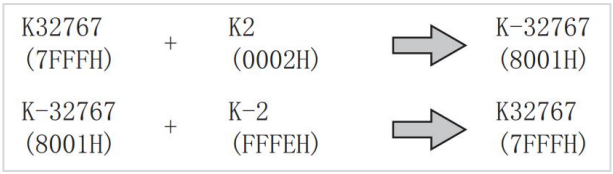


When a constant is specified in (s2) (when specified with a sign)



• When an underflow or overflow occurs in the operation result, the conditions are as follows. In this case, the carry flag does not change to ON.

When specifying sign:



When specifying unsigned:



Error code

Error code	Content
4084H	The device range of point (n) starting from (s1) or (s2) is partially consistent with the device range of point (n) starting from (d). (duplicate)
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



As shown in the above ladder program:

When X0 is ON, add the device data starting from D100 (the number of device points is the value stored in D0), and the number of devices starting from D150(the number of device points is the value stored in D0), and save the result to the program after D200.

6.13.2 DBK+/BIN 32-bit block data addition operation

DBK+(P)

Add the BIN 32-bit data of point (n) starting from the device specified in (s1) and the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[DBK+ (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
s1	The start device that stores addition operation data	-2147483648 to +2147483647	BIN32 bit	ANY32_S
s2	Addition data or the starting device that stores addition operation data	-2147483648 to +2147483647	BIN32 bit	ANY32_S
d	The start device that stores the operation data	-	BIN32 bit	ANY32_S
n	The number of addition operation data	0 to 65535	BIN32 bit	ANY32_S

Device used

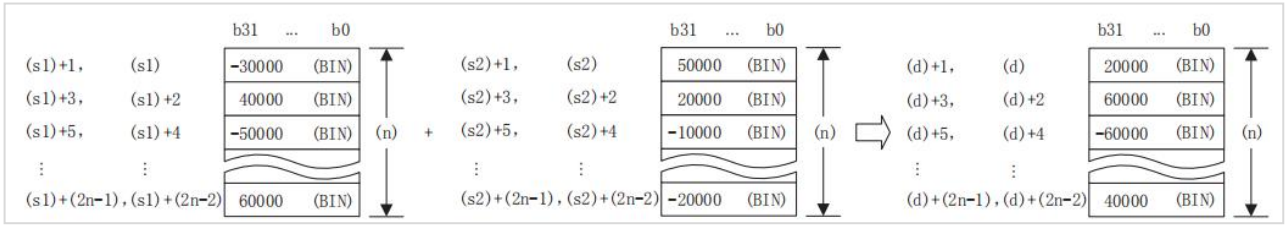
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBK+	Parameter 1															●	●	●	●	●	●	●				●	●
	Parameter 2															●	●	●	●	●	●	●	●	●		●	●
	Parameter 3															●	●	●	●	●	●	●				●	●
	Parameter 4																	●	●	●			●	●			●

Features

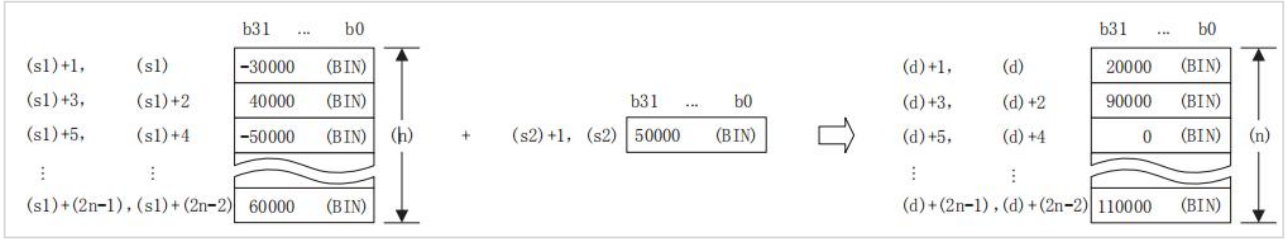
Add the BIN 32-bit data of point (n) starting from the device specified in (s1) and the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

• Block addition operations are performed in 32-bit units.

When a device is specified in (s2) (when specified with a sign)



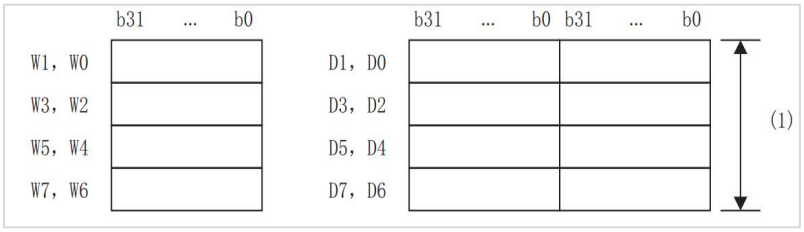
When a constant is specified in (s2) (when specified with a sign)



• When (s1) or (s2) and (d) are specified with the same device (completely consistent), operation could be performed. However, if the device range of point (n) starting from (s1) or (s2) partially matches (overlaps) the device range of point (n) starting from (d), an error occurs.

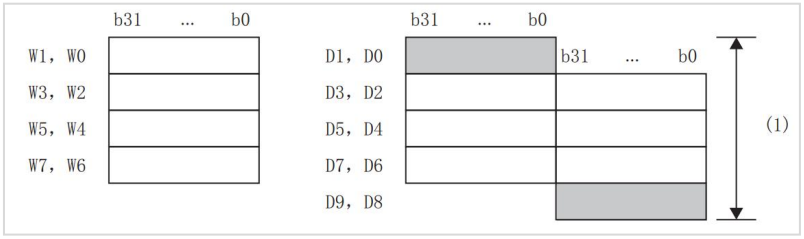
Example

When the first 4 points of the device of (s2) and (d) are completely consistent.



(1) Due to the complete consistence, operation could be executed.

When (s2) and (d) the first 4 points of the device are partially consistent.

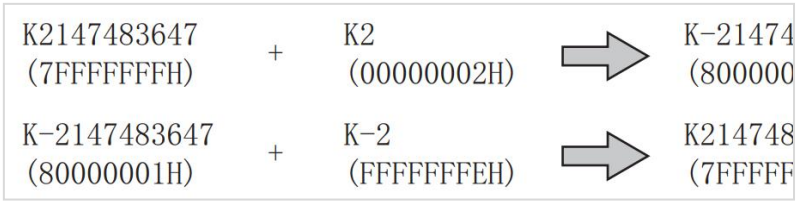


(1) Due to partial consistence, an operation error occurs.

• If the value specified in (n) is 0, it will be no processing.

• When an underflow or overflow occurs in the operation result, the conditions are as follows. In this case, the carry flag does not change to ON.

When a Sign is specified:



When specifying unsigned:



Error code

Error code	Content
4084H	The device range of point (n) starting from (s1) or (s2) is partially consistent with the device range of point (n) starting from (d). (duplicate)
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



As shown in the above ladder program:

When X0 is ON, add the device data starting from D100 (the number of device points is the value stored in D0), and the number of devices starting from D150(the number of device points is the value stored in D0), the result is saved to the program in the device after D200.

6.13.3 BK-/BIN 16-bit block data subtraction operation

BK-(P)

Subtract the BIN 16-bit data of point (n) starting from the device specified in (s1) and the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[BK- (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The start device that stores the subtracted data	-32768 to +32767	BIN16 bit	ANY16_S
(S2)	Subtraction data or the start device that stores the subtraction data	-32768 to +32767	BIN16 bit	ANY16_S
(d)	The start device that stores the operation result	-	BIN16 bit	ANY16_S
(n)	The number of subtraction operation data	0 to 65,535	BIN16 bit	ANY16

Device used

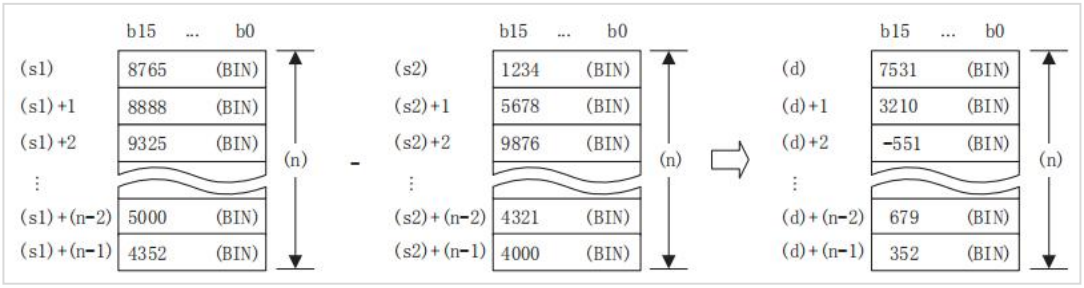
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BK-	Parameter 1															●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●			●	●		●	●
	Parameter 3															●	●	●	●	●						●	●
	Parameter 4																	●	●	●			●	●			●

Features

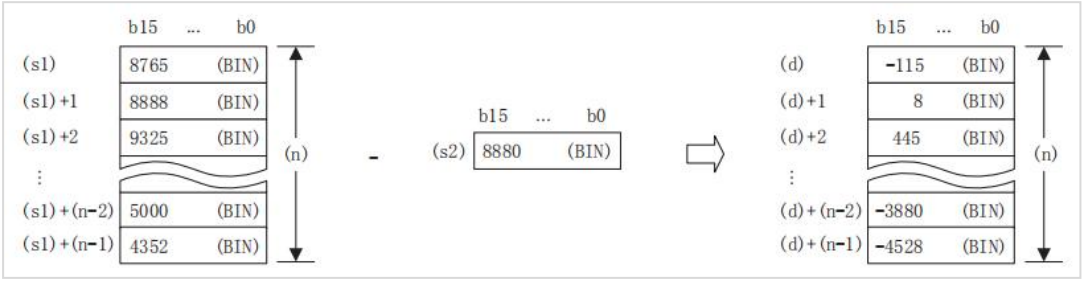
Subtract the BIN 16-bit data of point (n) starting from the device specified in (s1) and the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the result in the device specified in (d).

• Block subtraction operations are performed in 16-bit units.

When a device is specified in (s2)

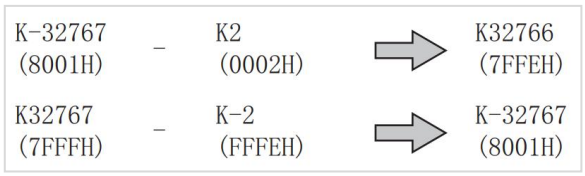


When a constant is specified in (s2)



• When an underflow or overflow occurs in the operation result, the conditions are as follows. In this case, the carry flag does not change to ON.

When a Sign is specified:



When specifying unsigned:



Error code

Error code	Content
4084H	The device range of point (n) starting from (s1) or (s2) is partially consistent with the device range of point (n) starting from (d). (duplicate)
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



As shown in the ladder program above:

When X010 is ON, after subtracting the 3 point data from D100 and the constant 8765, the result is saved to the program in the device after D200.

6.13.4 DBK-/BIN 32-bit block data subtraction operation

DBK-(P)

Subtract the BIN 32-bit data of point (n) starting from the device specified in (s1) and the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[DBK- (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	The start device that stores the subtracted data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s2)	Subtraction data or the start device that stores the subtraction data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device that stores the operation result	-	BIN32 bit	ANY32_S
(n)	The number of subtraction operation data	0 to 65,535	BIN32 bit	ANY32_S

Device used

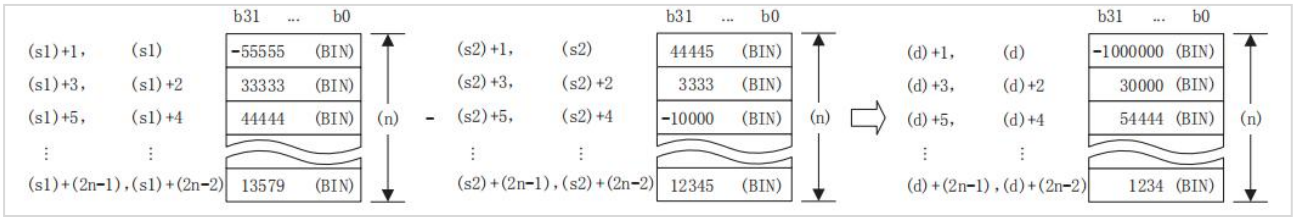
instruction	Parameter	Devices																					Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBK-	Parameter 1															●	●	●	●	●	●	●				●	●
	Parameter 2															●	●	●	●	●	●	●	●	●		●	●
	Parameter 3															●	●	●	●	●	●	●				●	●
	Parameter 4																	●	●	●			●	●			●

Features

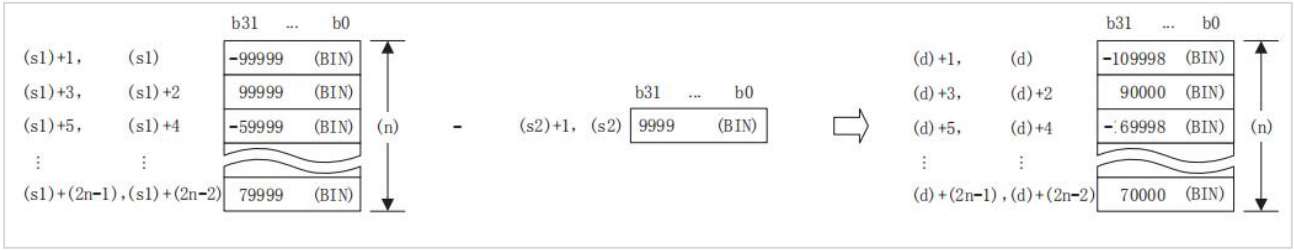
Subtract the BIN 32-bit data of point (n) from the device specified in (s1) and the BIN 32-bit data of point (n) from the device specified in (s2), and store the result in the device specified in (d).

• Block subtraction operations are performed in 32-bit units.

When a device is specified in (s2) (when specified with a sign)



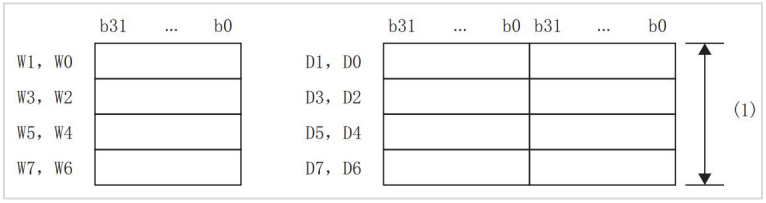
When a constant is specified in (s2) (when specified with a sign)



• When (s1) or (s2) and (d) are specified with the same device (completely consistent), operation could be performed. However, if the device range of point (n) starting from (s1) or (s2) partially matches (overlaps) the device range of point (n) starting from (d), an error occurs.

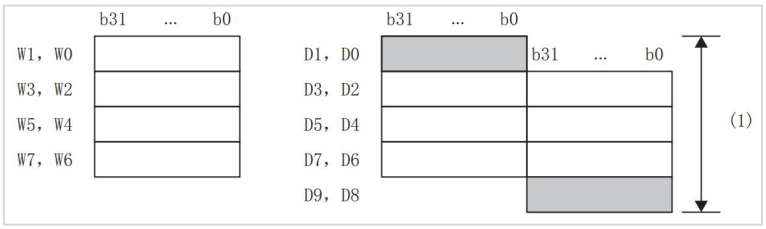
Example

When the first 4 points of the device of (s2) and (d) are completely consistent.



(1)Due to the complete consistency, operation could be executed.

When the first 4 points of the device of (s2) and (d) are partially consistent.

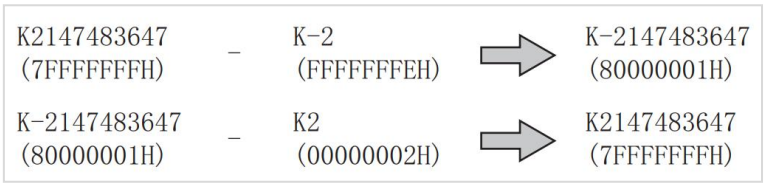


(1)Due to the partial consistency, an operation error occurs.

• If the value specified in (n) is 0, it will be no processing.

• When an underflow or overflow occurs in the operation result, the conditions are as follows. In this case, the carry flag does not change to ON.

When specifying Signed:



When specifying unsigned:



Error code

Error code	Content
4084H	The device range of point (n) starting from (s1) or (s2) is partially consistent with the device range of point (n) starting from (d). (duplicate)
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



As shown in the ladder program above:

When X010 is ON, after subtracting the 3-point data starting from D100 with the constant 987,654,321, save the result to the program in the device after D200.

6.13.5 BKCMP=/BIN 16-bit Block Data Comparison

BKCMP=(P)

Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[BKCMP= (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-32768 to 32767	BIN16 bit	ANY16_S
(s2)	The device storing the comparison source data	-32768 to 32767	BIN16 bit	ANY16_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN16 bit	ANY16

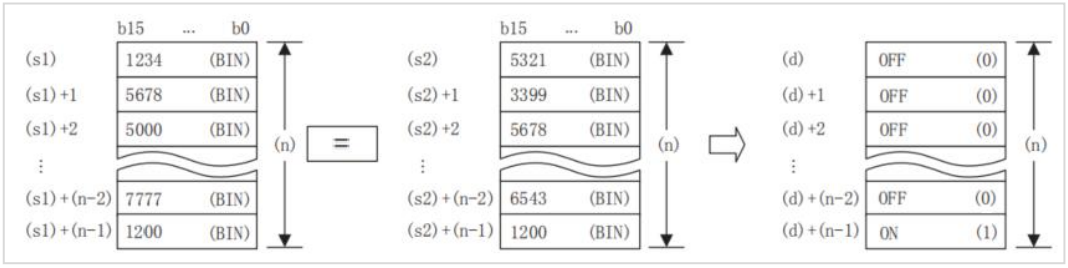
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BKCMP =	Parameter 1															●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3	●	●	●	●						●															●	●
	Parameter 4																	●	●				●	●			●

Features

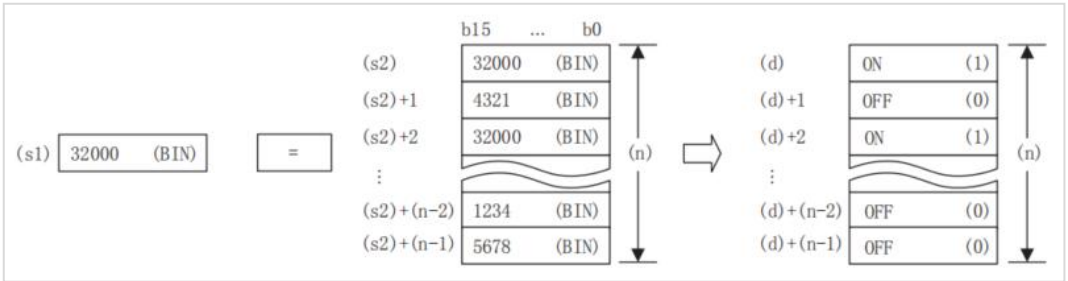
Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the comparative result in the point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 16-bit units.

• (s1) could specify a direct constant.



instruction sign	Condition	Comparative result
BKCMP=	(s1)=(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use “BKCMP=” instruction to compare the 4-point 16-bit data (BIN) starting from D100 and the 4-point 16-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.6 DBKCMP=/BIN32-bit block data comparison

DBKCM=(P)

Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[DBKCMP= (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s2)	The device storing the comparison source data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65,535	BIN32 bit	ANY32_S

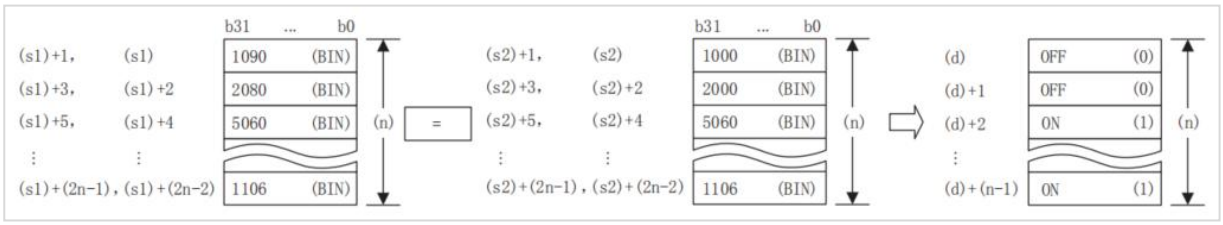
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBKCMP=	Parameter 1															●	●	●	●	●	●	●	●	●		●	●
	Parameter 2															●	●	●	●	●	●	●				●	●
	Parameter 3		●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

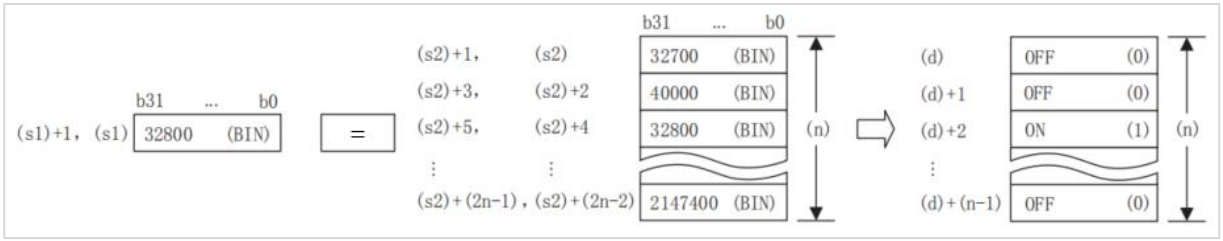
Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the comparison result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 32-bit units.

• (s1) could specify a direct constant.



• (d) is specified outside the device range of point (n) starting from (s1) and (s2).

• The comparison operation result of each instruction is shown below.

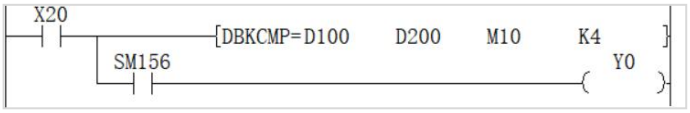
instruction sign	Condition	Comparative results
DBKCMP=	(s1)=(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4084H	When (n) is out of range
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use DBKCMP= instruction to compare the 4 points 32-bit data (BIN) starting from D100 and the 4 points 32-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.7 BKCMP<>/BIN 16-bit block data comparison

BKCMP<>(P)

Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

Ladder

-[BKCMP<> (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s1)	Comparative data or the device storing comparative data	-32768 to 32767	BIN16 bit	ANY16_S	--
(s2)	The device storing the comparison source data	-32768 to 32767	BIN16 bit	ANY16_S	--
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL	--
(n)	The number of comparative data	0 to 65535	BIN16 bit	ANY16	U

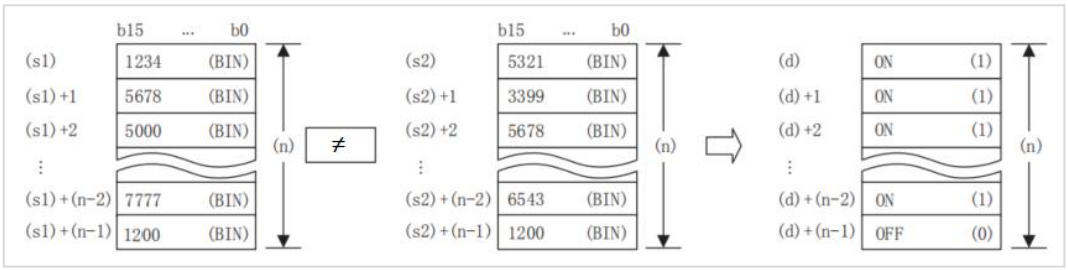
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BKCMP<>	Parameter 1															●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3		●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

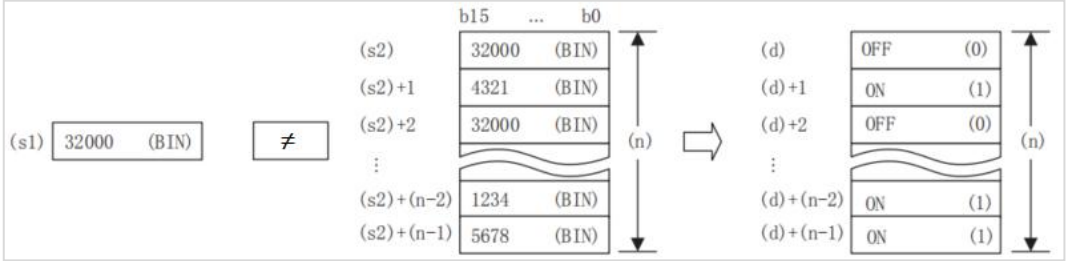
Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the comparison result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 16-bit units.

• (s1) could specify a direct constant.



instruction Sign	condition	Comparative results
BKCMP<>	(s1)≠(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use BKCMP<> instruction to compare the 4-point 16-bit data (BIN) starting from D100 and the 4-point 32-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.8 DBKCMP<>/BIN32-bit block data comparison

DBKCMP<>(P)

Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[DBKCMP<> (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s2)	The device storing the comparison source data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN32 bit	ANY32_S

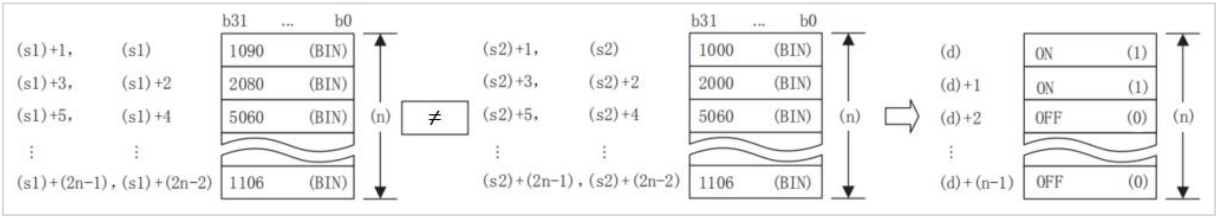
Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBKCMP<>	Parameter 1															●	●	●	●	●	●	●	●	●		●	●
	Parameter 2															●	●	●	●	●	●	●				●	●
	Parameter 3		●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

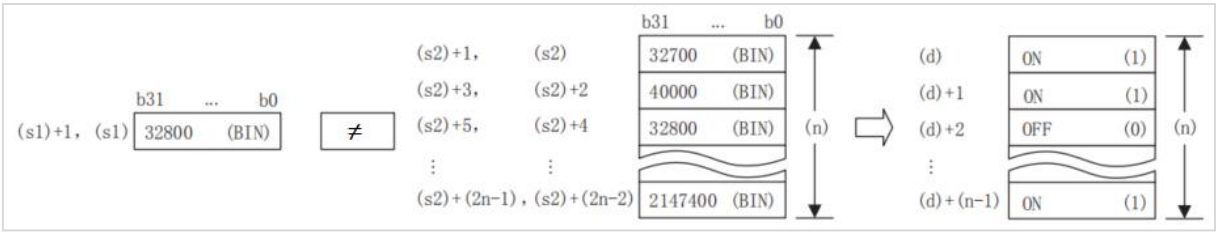
Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the comparison result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 32-bit units.

• (s1) could specify a direct constant.



• (d) is specified outside the device range of point (n) starting from (s1) and (s2).

• The comparative operation result of each instruction is shown below.

instruction sign	Condition	Comparative results
DBKCMP<>	(s1)≠(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4084H	When (n) is out of range
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use DBKCMP<> instruction to compare the 4-point 32-bit data (BIN) starting from D100 and the 4-point 32-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.9 BKCMP>/BIN 16-bit block data comparison

BKCMP>(P)

Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[BKCMP> (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-32768 to 32767	BIN16 bit	ANY16_S
(s2)	The device storing the comparison source data	-32768 to 32767	BIN16 bit	ANY16_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN16 bit	ANY16

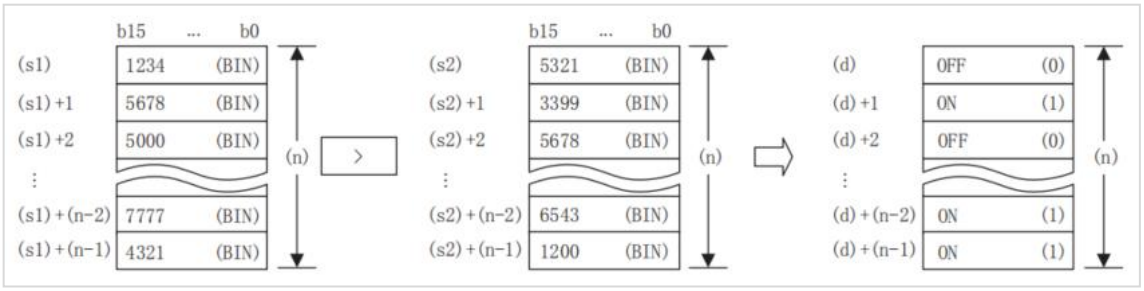
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BKCMP>	Parameter 1															●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3		●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

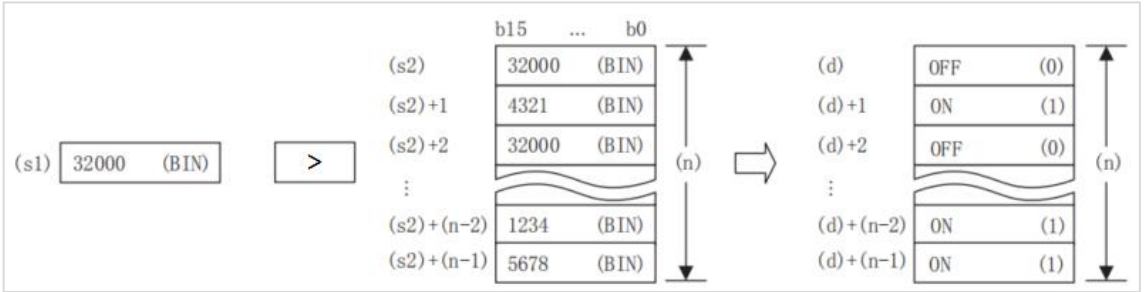
Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the comparison result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 16-bit units.

• (s1) could specify a direct constant.



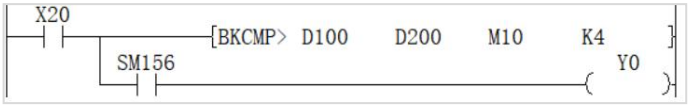
instruction sign	Condition	Comparative results
BKCMP>	(s1)>(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use BKCMP> instruction to compare the 4-point 16-bit data (BIN) starting from D100 and the 4-point 16-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.10 DBKCMP>/BIN32-bit block data comparison

DBKCMP>(P)

Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the operation result in (d) In the specified device.

-[DBKCMP> (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s2)	The device storing the comparison source data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN32 bit	ANY32_S

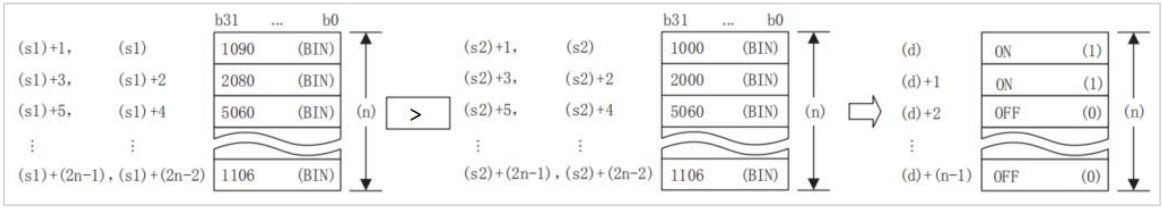
Device used

Instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBKCMP>	Parameter 1															●	●	●	●	●	●	●	●	●		●	●
	Parameter 2															●	●	●	●	●	●	●				●	●
	Parameter 3		●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

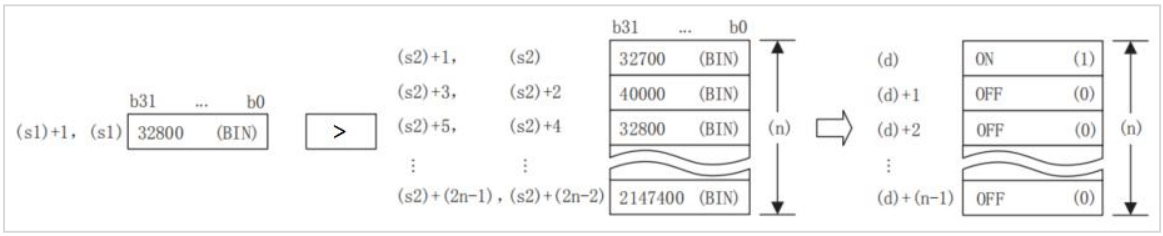
Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the comparison result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 32-bit units.

• (s1) could specify a direct constant.



• (d) is specified outside the device range of point (n) starting from (s1) and the device range of point (n) starting from (s2).

• The comparison operation result of each instruction is shown below.

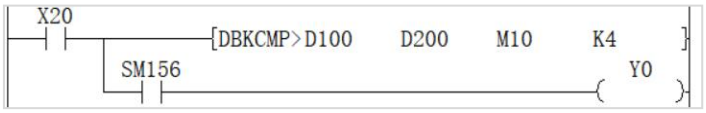
instruction sign	Condition	Comparative results
DBKCMP>	(s1)>(s2)	ON

• When all the comparative operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4084H	When (n) is out of range
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use DBKCMP> instruction to compare the 4-point 32-bit data (BIN) starting from D100 and the 4-point 32-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.11 BKCMP>=/BIN 16-bit block data comparison

BKCMP>=(P)

Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[BKCMP>= (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-32768 to 32767	BIN16 bit	ANY16_S
(s2)	The device storing the comparison source data	-32768 to 32767	BIN16 bit	ANY16_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65,535	BIN16 bit	ANY16

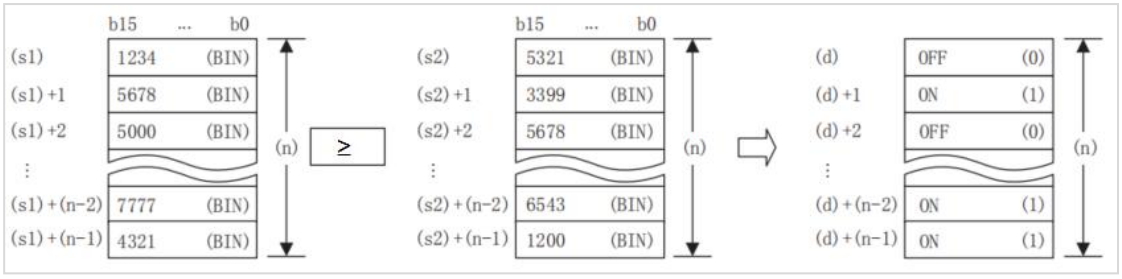
Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BKCMP>=	Parameter 1															●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3		●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

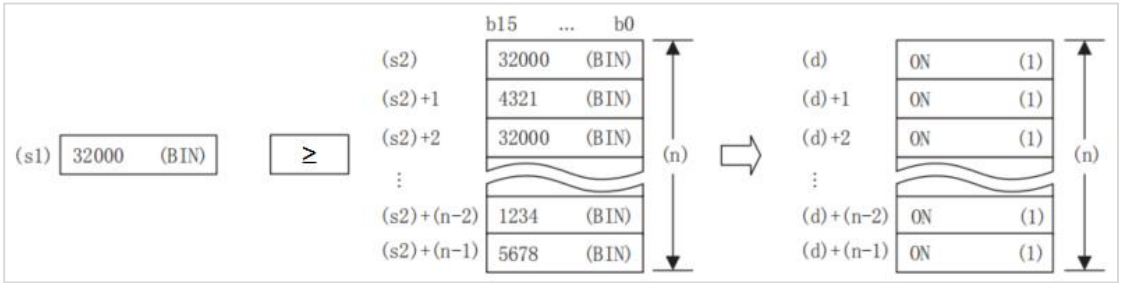
Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the comparative result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 16-bit units.

• (s1) could specify a direct constant.



instruction sign	Condition	Comparative results
BKCMP>=	(s1)>=(s2)	ON

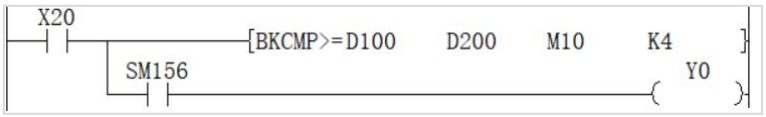
• When all the comparative operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would

turn ON.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use BKCMP>= instruction to compare the 4-point 16-bit data (BIN) starting from D100 and the 4-point 16-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.12 DBKCMP>=/BIN32-bit block data comparison

DBKCMP>=(P)

Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[DBKCMP>= (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s2)	The device storing the comparison source data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN32 bit	ANY32_S

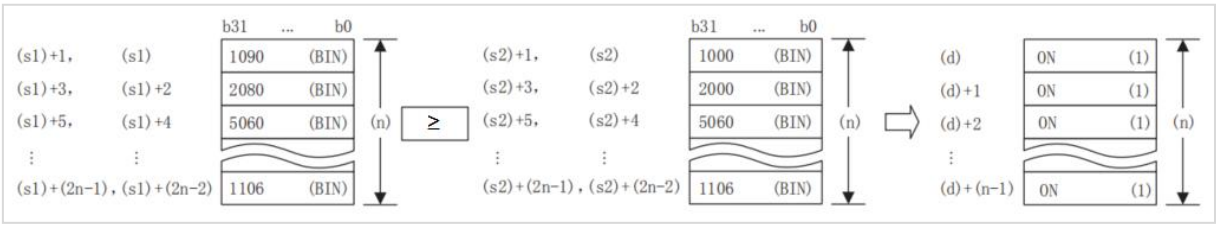
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBKCMP>=	Parameter 1															●	●	●	●	●	●	●	●	●		●	●
	Parameter 2															●	●	●	●	●	●	●				●	●
	Parameter 3		●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

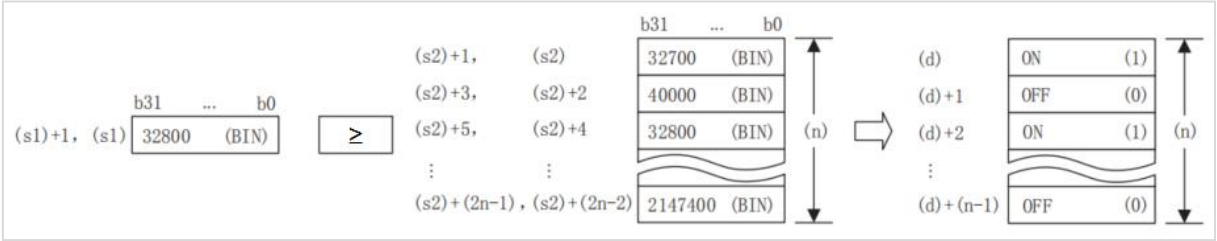
Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the comparative result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 32-bit units.

• (s1) could specify a direct constant.



• (d) is specified outside the device range of point (n) starting from (s1) and the device range of point (n) starting from (s2).

• The comparison operation result of each instruction is shown below

instruction sign	Condition	Comparative results
DBKCMP>=	(s1)>=(s2)	ON

• When all the comparison operation results stored in point (n) at the beginning of (d) are ON (1), SM349 (block comparison signal) will turn ON.

Error code

Error code	Content
4084H	When (n) is out of range
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use DBKCMP>= instruction to compare the 4-point 32-bit data (BIN) starting from D100 and the 4-point 32-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.13 BKCMP</BIN 16-bit block data comparison

BKCMP<(P)

Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[BKCMP< (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-32768 to 32767	BIN16 bit	ANY16_S
(s2)	The device storing the comparison source data	-32768 to 32767	BIN16 bit	ANY16_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN16 bit	ANY16

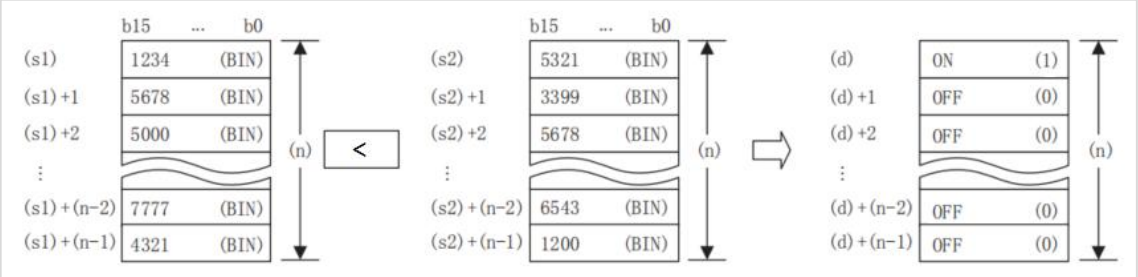
Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BKCMP<	Parameter 1															●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3	●	●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

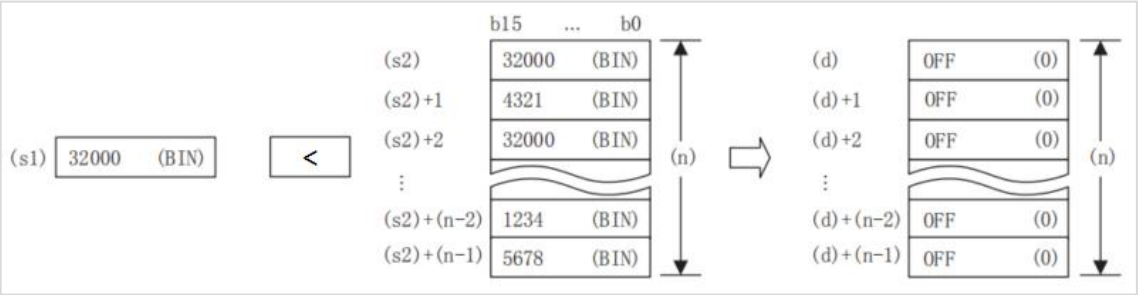
Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the comparative result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 16-bit units.

• (s1) could specify a direct constant.



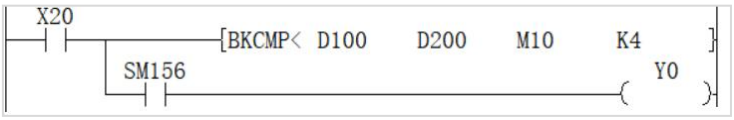
instruction sign	Condition	Comparative results
BKCMP<	(s1)<(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use BKCMP< instruction to compare the 4-point 16-bit data (BIN) starting from D100 and the 4-point 16-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.14 DBKCMP</BIN 32-bit block data

DBKCMP<(P)

Convert the n characters (bit) in the HEX code data specified in (s) to ASCII codes, and store then after the device number specified in (d).

-[DBKCMP< (P) (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s2)	The device storing the comparison source data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN32 bit	ANY32_S

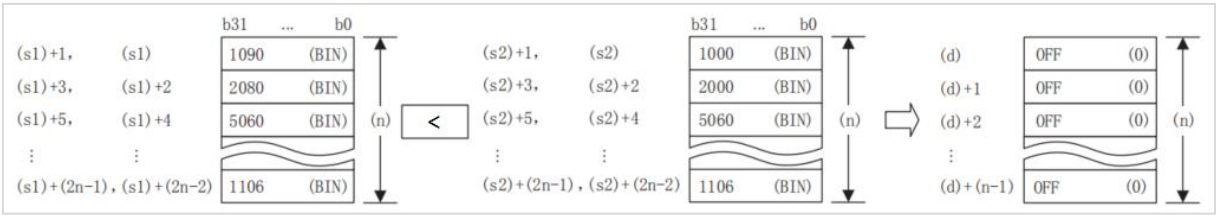
Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DBKCMP<	Parameter 1															●	●	●	●	●	●	●	●	●		●	●
	Parameter 2															●	●	●	●	●	●	●				●	●
	Parameter 3		●	●	●	●					●															●	●
	Parameter 4																	●	●				●	●			●

Features

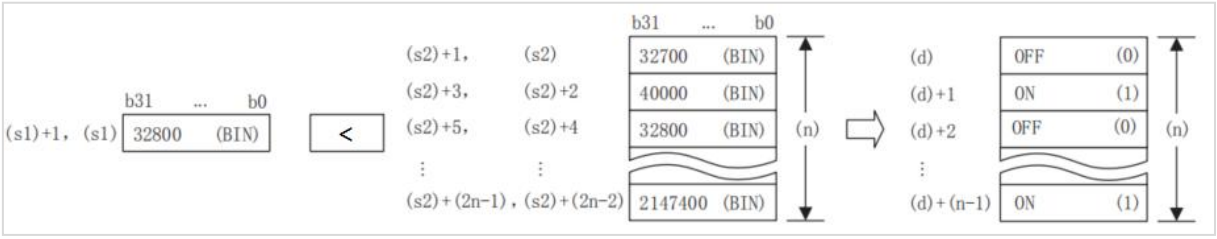
Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the comparative result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 32-bit units.

• (s1) could specify a direct constant.



• (d) is specified outside the device range of point (n) starting from (s1) and the device range of point (n) starting from (s2).

• The comparison operation result of each instruction is shown below.

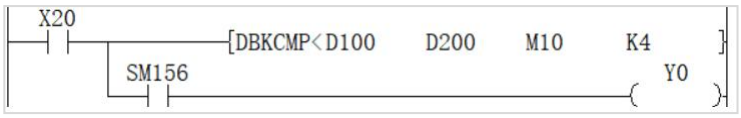
instruction sign	Condition	Comparative results
DBKCMP<	(s1)>=(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM349 (block comparison signal) would turn ON.

Error code

Error code	Content
4084H	When (n) is out of range
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use DBKCMP< instruction to compare the 4-point 32-bit data (BIN) starting from D100 and the 4-point 32-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.15 BKCMP<=/BIN16-bit block data cmparison

BKCMP<=(P)

Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[BKCMP<= (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-32768 to 32767	BIN16 bit	ANY16_S
(s2)	The device storing the comparison source data	-32768 to 32767	BIN16 bit	ANY16_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN16 bit	ANY16

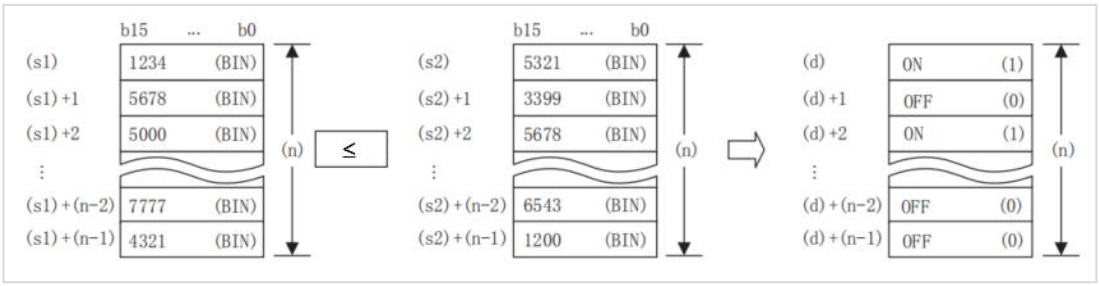
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BKCMP<=	Parameter 1															●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3	●	●	●	●						●															●	●
	Parameter 4																	●	●				●	●			●

Features

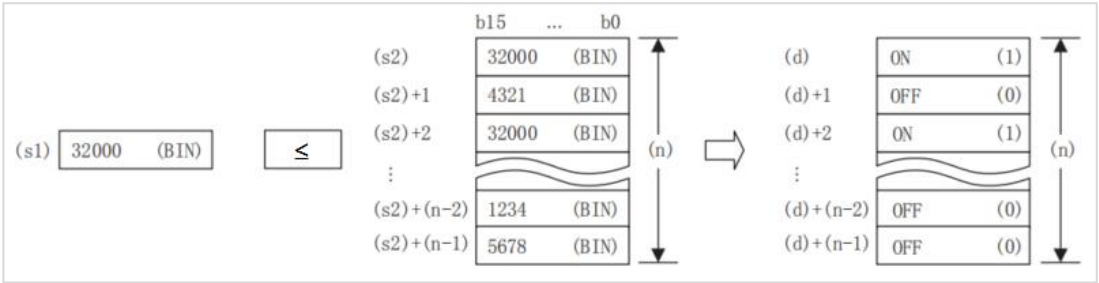
Compare the BIN 16-bit data of point (n) starting from the device specified in (s1) with the BIN 16-bit data of point (n) starting from the device specified in (s2), and store the comparative result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 16-bit units.

• (s1) could specify a direct constant.



• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

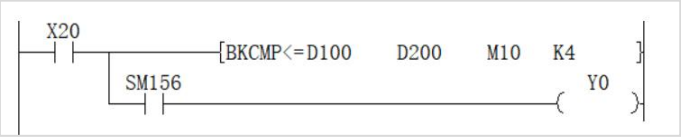
instruction sign	Condition	Comparative results
BKCMP<=	(s1)>=(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use BKCMP<= instruction to compare the 4-point 16-bit data (BIN) starting from D100 and the 4-point 16-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.13.16 DBKCMP<=/BIN32-bit block data comparison

DBKCMP<=(P)

Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the operation result in the device specified in (d).

-[DBKCMP<= (s1) (s2) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Comparative data or the device storing comparative data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(s2)	The device storing the comparison source data	-2147483648 to 2147483647	BIN32 bit	ANY32_S
(d)	The start device storing the comparative result	-	Bit	ANY_BOOL
(n)	The number of comparative data	0 to 65535	BIN32 bit	ANY32_S

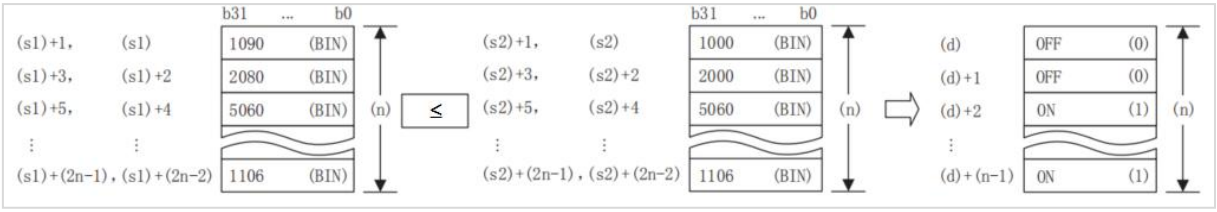
Device used

instruction	Iarameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BKCMP<=	Parameter 1															●	●	●	●	●	●	●	●	●		●	●
	Parameter 2															●	●	●	●	●	●	●				●	●
	Parameter 3	●	●	●	●						●															●	●
	Parameter 4																	●	●				●	●			●

Features

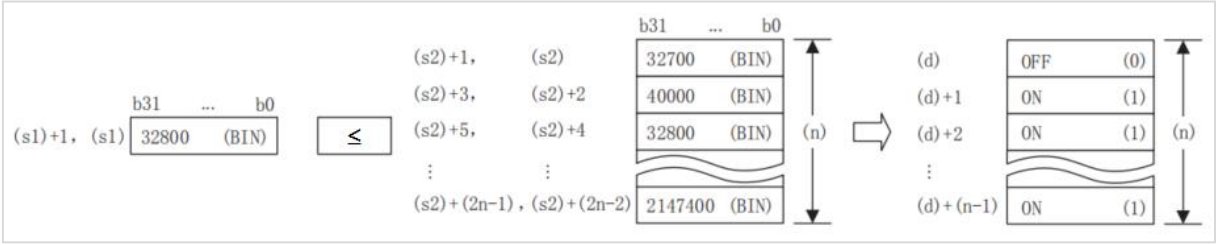
Compare the BIN 32-bit data of point (n) starting from the device specified in (s1) with the BIN 32-bit data of point (n) starting from the device specified in (s2), and store the comparative result in point (n) starting from the device specified in (d).

• The corresponding device at point (n) starting from the device specified in (d) turns on when the comparison condition is satisfied, and turns off when the comparison condition is not satisfied.



• Comparison operations are performed in 32-bit units.

• (s1) could specify a direct constant.



• (d) is specified outside the device range of point (n) starting from (s1) and the device range of point (n) starting from (s2).

• The comparison operation result of each instruction is shown below.

instruction sign	Condition	Comparative results
DBKCMP<=	(s1)>=(s2)	ON

• When all the comparison operation results stored in point (n) starting from (d) are ON (1), SM156 (block comparison signal) would turn ON.

Error code

Error code	Content
4084H	When (n) is out of range
4085H	When the specified device range for reading exceeds the range of the corresponding device
4086H	When the specified device range for writing exceeds the range of the corresponding device

Example



When X020 is ON, use DBKCMP<= instruction to compare the 4-point 32-bit data (BIN) starting from D100 and the 4-point 32-bit data starting from D200, and save the result to the program in the 4-point of the device starting from M10.

In addition, when the comparative results (4 points starting from M10) are all ON (1), Y000 turns ON.

6.14 Data table operation instructions

6.14.1 SFRD/Shift read

SFRD(P)

Data read instructions for first-in, first-out and control.

-[SFRD (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start word device number storing data (The start is a pointer, and the data starts from (s)+1)	-	Signed BIN 16 bit	ANY16
(d)	The word device number storing the first-out data	-	Signed BIN 16 bit	ANY16
(n)	It should be specified as the value of the number of points + 1 of the stored data. +1 is pointer	2 to 512	Signed BIN 16 bit	ANY16

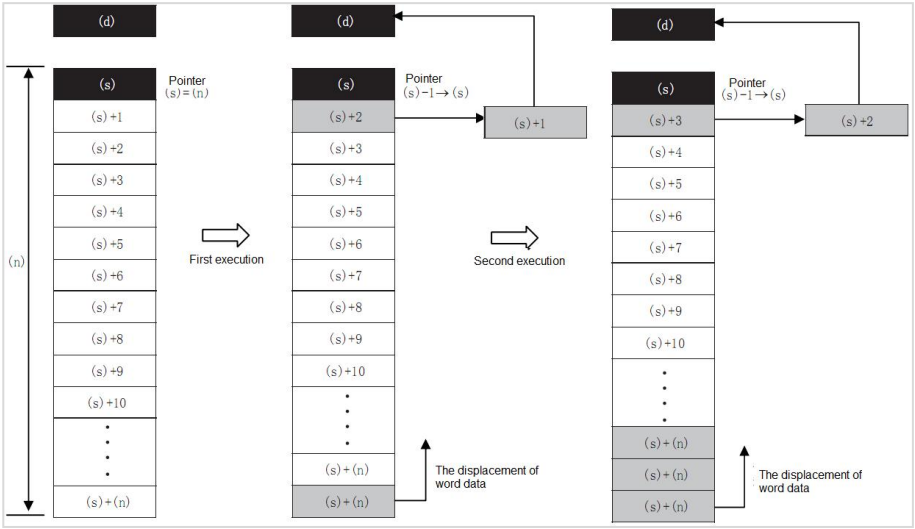
Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SFRD	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Transfer (s)+1 written sequentially to (d) by SFWR instruction, shift up each point (n)-1 by one word from (s)+1. The number of stored data of (s) subtracts one.

The content of (s)+1 is transferred (read) to (d). At the same time, the content of the pointer (s) decreases, and the data is shifted up by 1 word each. (In the continuous execution instruction SFRD, each operation cycle will shift, so the pulse execution instruction SFRDP should be used for programming).



Related device

Devices	Name	Content
SM153	Zero bit	Data readout usually starts from (s)+1, but when the pointer (s) is 0, the zero bit SM153 will operate.

✎Note:

The data after reading would not change the content of (s)+(n) due to reading.

In the case of continuous execution (SFRD) instructions, each scan time (operation cycle) will be read sequentially, but the content of (s) + (n) would not change. When the pointer (s) is 0, it would become no processing, and the content of (d) would not change.

Error code

Error code	Content
4084H	When the value set in (n) is other than the following. 2≤(n)≤512
	A negative value is specified in (s).
4085H	When the device specified in the read application instruction (s) and (n) exceeds the corresponding device range.
4086H	When the device specified in the write application instruction (s) and (d) exceeds the corresponding device range.

Example

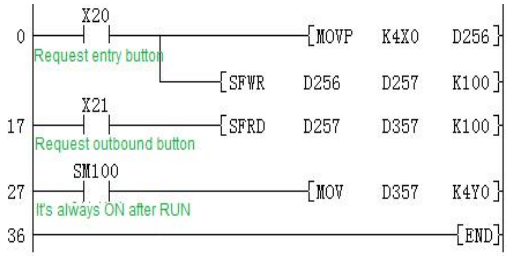
The following examples illustrate the use of shift write (SFWR) and shift read (SFRD) instructions.

(1) Action content

1) While registering the product number, in order to realize the first-in-first-out principle, the following introduces an example of a ladder ladder diagram that outputs the current product number.
2) The product number is a hexadecimal number with 4 digits or less, and the maximum inventory is below 99 points.

(2) Program

1) Program 1

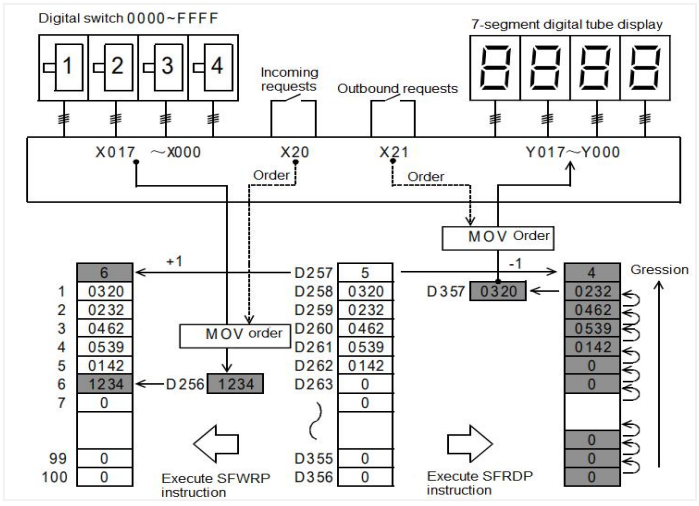


Enter the product number from X0 to X17 and transfer it to D256

Data register D258 to D356 (99 points) for pointer D257 storing product number

Output the first-in product number to D357 according to the requirements

Output the product number that should be taken out in hex 4 digits to Y0 to Y17



2) Program 2





Turn X0 from OFF to ON, and this instruction acts according to the following numbers 1 to 3. (The content of D10 remains unchanged),

① The content of D2 is read out and sent to D20.
② D10 to D3 all shift one register to the right.
③ The contents of pointer D1 are reduced by 1.

6.14.2 POP/Read from the back of the data table

POP(P)

Read the last data written by the shift write instruction (SFWR) for first in first out/first in last out control.

-[POP (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start device number that stores the first-in data (including pointer data) (the start word device number that stores data)	-	Signed BIN 16 bit	ANY16
(d)	The device number that stores the last-out data	-	Signed BIN 16 bit	ANY16
(n)	The points of stored data	2 to 512	Signed BIN 16 bit	ANY16

Device used

instruction	parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
POP	Parameter 1												●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

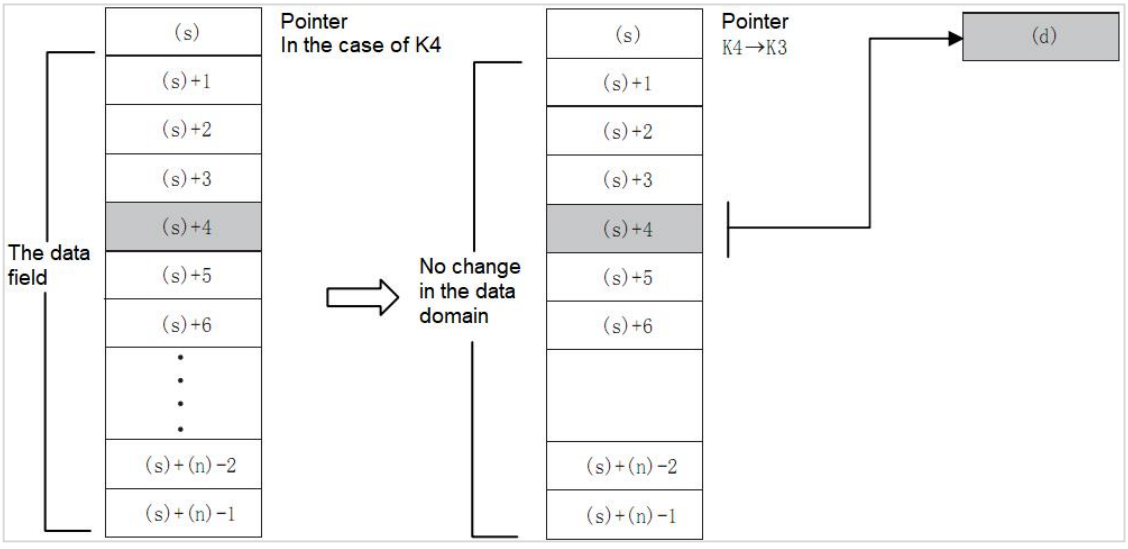
Features

For the word device of "(s) to (s)+(n)-1", the device of "(s) + instruction data (s)" will be read to (d) (The last data written by the shift write instruction (SFWR) for first-in first-out control is read to (d)). (N) Specifies 2 to 512.

The value of pointer data (s) is reduced by one.

First-in-last-out control data

	Content
(S)	Pointer data (the number of stored data)
(S)+1	Data field (Data entered by shift write instruction (SFWR))
(S) +2	
(S) +3	
.....	
(S)+(n)-3	
(S)+(n)-2	
(S)+(n)-1	



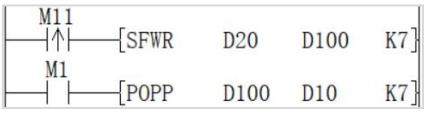
✎**Note:**

- If the POP(P) instruction is programmed in continuous execution type, the instruction will be processed per cycle. Therefore, it may not be possible to achieve the desired action. Generally, POP(P) instruction programming should be executed with "pulse execution type" or "pulse specified contact".
- When the current value of pointer (s) is 0, the zero flag SM153 turns on, and the POP(P) instruction becomes no processing.
- When the current value of pointer (s) is 1, write 0 to (s), and the zero flag SM153 turns on.

Error code

Error code	Content
4084H	(s)>(n)-1
	(s)<0
	Data outside the specified range is entered in (n). 2≤(n)≤512
4085H	When the device specified in the read application instruction (s) and (n) exceeds the corresponding device range
4086H	When the device specified in the write application instruction (s) and (d) exceeds the corresponding device range

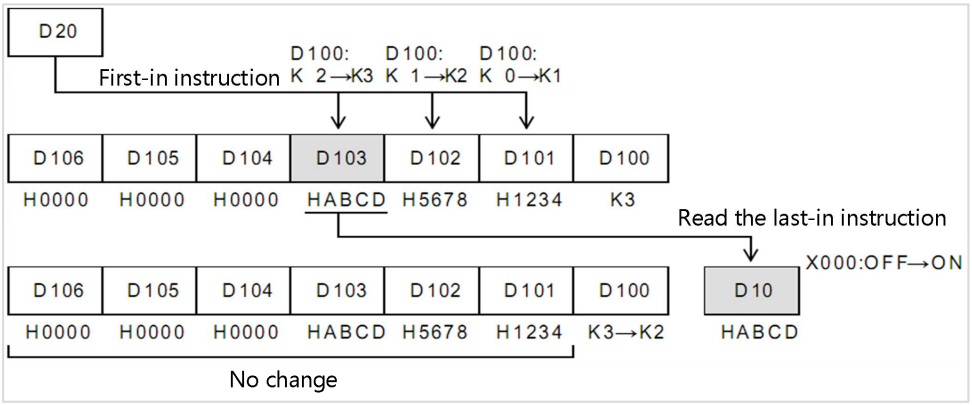
Example



Each time M1 is ON, for the values of D20 input first in D101 to D106, the last saved value would be saved in D10, and then the data saved number (pointer D100) will be reduced by 1.

When the data entered first is the content in the table below.

Pointer	D100	K3
Data	D101	H1234
	D102	H5678
	D103	HABCD
	D104	H0000
	D105	H0000
	D106	H0000



6.14.3 SFWR/Shift write

SFWR(P)

Data writing instructions for first-in-first-out and control.

-[SFWR (s) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The word device number that stores the data you want to enter first	-	Signed BIN 16 bit	ANY16
(d)	The start word device number for storing data and shifting (the start is pointer, and data starts from (d)+1)	-	Signed BIN 16 bit	ANY16
(n)	The value of points + 1 of stored data should be specified	2 to 512	Signed BIN 16 bit	ANY16

Device used

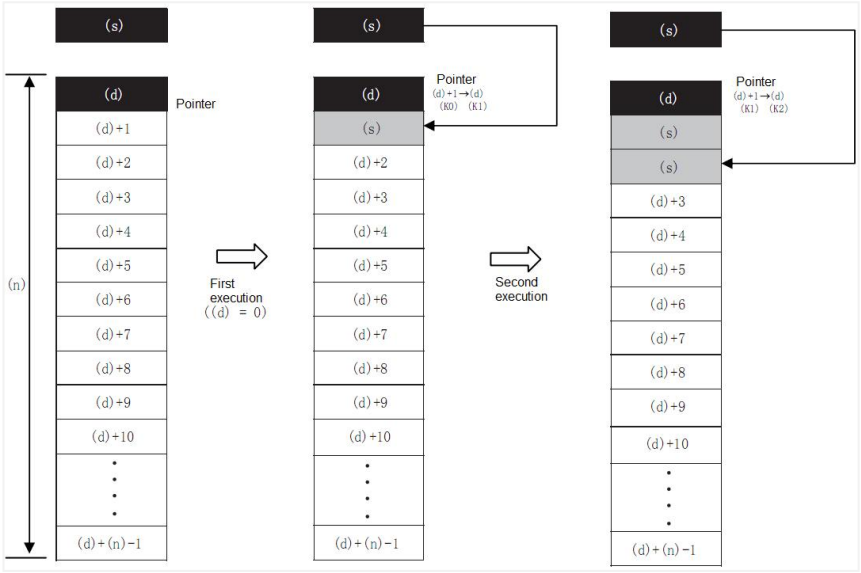
instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SFWR	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Start from (d)+1, write the contents of (s) to point (n)-1, and the stored data of (d) add one. For example, if (d)=0, write to (d) +1, (d)=1, write to (d)+2.

Through the first execution, the content of (s) is stored to (d)+1 and becomes the value of (s).

If the content of (s) is changed and executed for the second time, the content of (s) is stored to (d)+2, and the content of (d)+2 becomes (s) (in the continuous execution instruction SFWR, each operation cycle will be stored sequentially, so the pulse execution instruction SFWRP should be used for programming). After that, the data will be filled in sequentially from the right, and the number of data storage points is displayed by the content of the pointer (d).



Related device

Devices	Name	Content
SM151	carry	When the content of pointer (s) exceeds (n)-1, it becomes no processing (no writing), and the carry flag SM151 turns ON.

✎Note:

In the continuous execution type (SFWR) instruction, you should be noted that each scan time (operation cycle) will be stored

(overwritten) sequentially.

Error code

Error code	Content
4084H	When the value set in (n) is other than the following. 2≤(n)≤512
	A negative value is specified in (d).
4085H	When the device specified in the read application instruction (s), (d) and (n) exceeds the corresponding device range.
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range.

Example

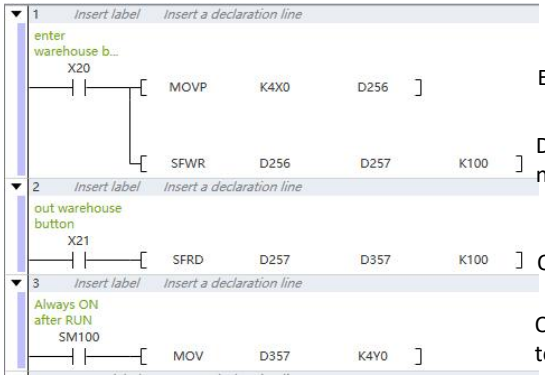
The following examples illustrate the use of shift write (SFWR) and shift read (SFRD) instructions.

7.14.4 (1) Action content

1) While registering the product number, in order to realize the first-in-first-out principle, the following introduces an example of a ladder ladder diagram that outputs the current product number.
2) The product number is a hexadecimal number with 4 digits or less, and the maximum inventory is below 99 points.

(2) Program

1) Program 1

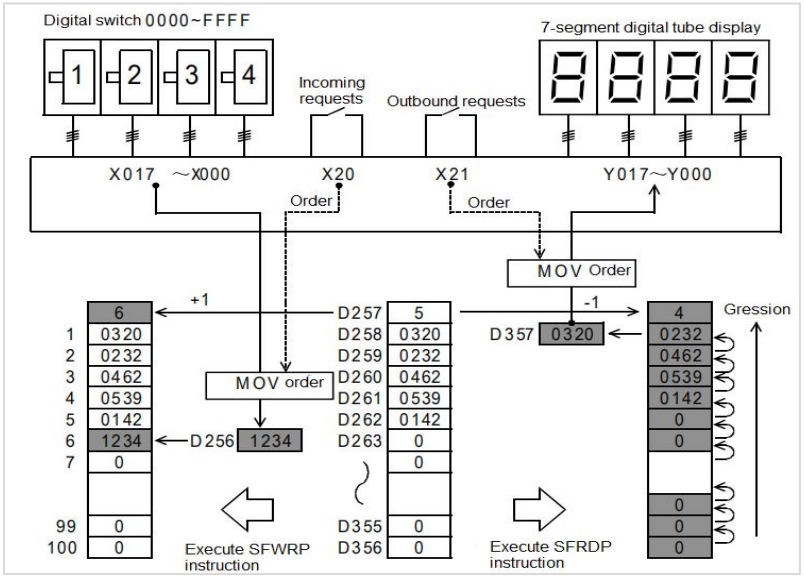


Enter the product number from X0 to X17 and transfer it to D256

Data register D258 to D356 (99 points) for pointer D257 storing product number

Output the first-in product number to D357 according to the requirements

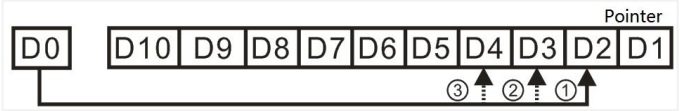
Output the product number that should be taken out in hex 4 digits to Y0 to Y17



2) Program 2



First-in-first-out data read instruction



When X0=1, the content of D0 is stored in D2, and the content of D1 becomes 1. When X0 changes from OFF to ON again, the content of D0 is stored in D3, the content of D1 becomes 2, and so on. If the content of D1 exceeds n-1, the instruction is not processed, and the carry flag M8022 will be set to 1.

6.14.4 FINS/Data table data insertion

FINS(P)

Insert the BIN 16-bit data specified in (s) into the number (n) of the data table specified in (d). After the instruction is executed, the data starting with number (n) in the data table will be postponed downward one by one.

-[FINS (s) (d) (n)]

Content, range and data type

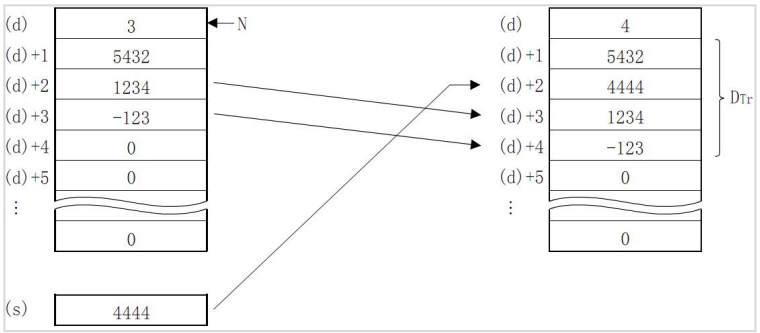
Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of device storing the inserted data	-	Signed BIN 16 bit	ANY16
(d)	The starting number of table	-	word	ANY16
(n)	The position of the inserted table	1-512	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
FINS	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Insert the BIN 16-bit data specified in (s) into the number (n) of the data table specified in (d). After the instruction is executed, the data starting with number (n) in the data table will be postponed downward one by one.



① N: the number of data storage;

② DTr: data table range;

③ When (n)=2, it will be inserted into (d)+2.

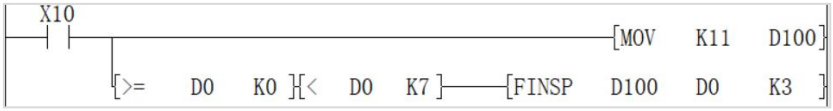
✎Note:

- The range of device used in the data table is managed by user.
- The range of the data table is (d) started from device (d) +1) after the number of data (d).

Error code

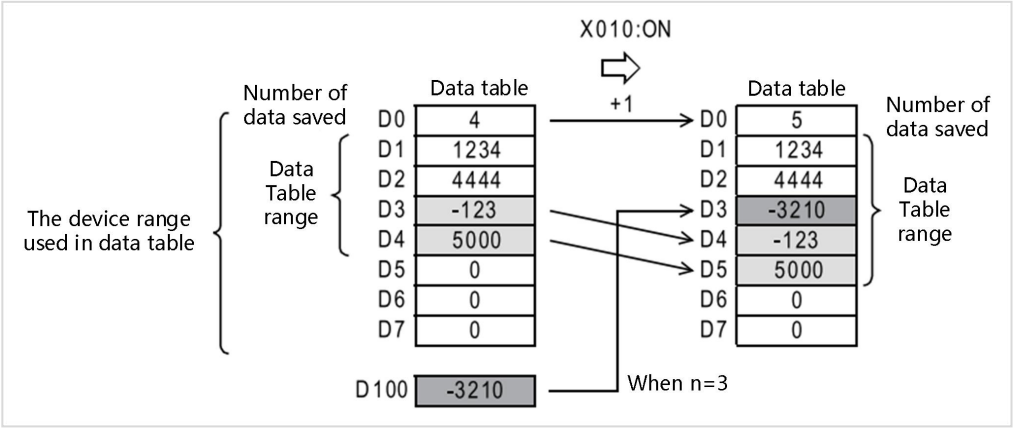
Error code	Content
4084H	FINS(P) instruction is executed when the value of (d) is 0.
	The storage data of the table of (d) exceeds 512.
	When the data set in (n) is other than the following, 1≤(n)≤512.
	When the FINS(P) instruction is executed, the table position (n) of the inserted data is greater than data storage number.
4085H	When the device specified in the read application instruction (s), (d) and (n) exceeds the corresponding device range.
4086H	When the device specified in the write application instruction (d) exceeds the corresponding device range.

Example



When X10=ON, insert the data of D100 into No. 3 of the data table of D0 to D4.

However, when the number of saved data exceeds 7, the FINS(P) instruction is not executed (the device range used in the data table is D0 to D7).



6.14.5 FDEL/Data deletion of data sheet

FDEL(P)

Delete the (n)th data of the data table specified in (d) and store it in the device specified in (s). After the instruction is executed, the data after (n)+1 in the data table will be postponed forward one by one.

-[FDEL (s) (d) (n)]

Content, range and data type

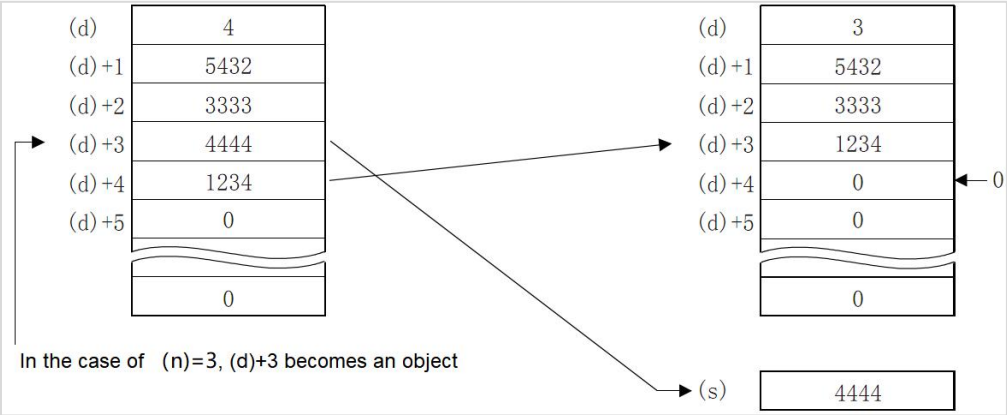
Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of the device stored deleted data	-	Signed BIN 16 bit	ANY16
(d)	The starting number of the table	-	word	ANY16
(n)	The position of the deleted table	1-512	Signed BIN 16 bit	ANY16

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	S M	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
FDEL	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●			●	●		●	●

Features

Delete the (n)th data of the data table specified in (d) and store it in the device specified in (s). After the instruction is executed, the data after (n)+1 in the data table will be postponed forward one by one.



✎Note:

- The user is responsible for the management of the device range used in the data sheet.
- The range of the data table is (d) after the data storage number (d) of the device ((d) + 1).

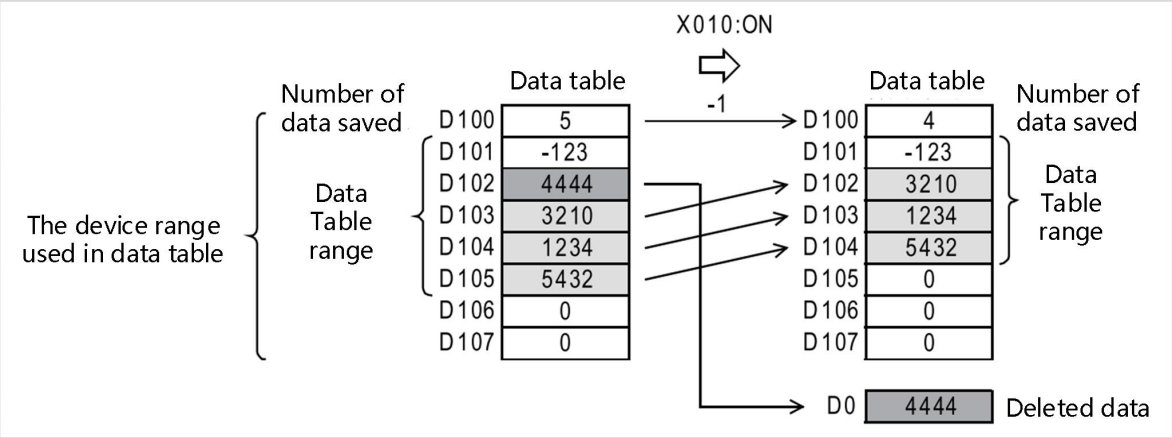
Error code

Code	Content
4084H	FDEL(P) instruction is executed when the value of (d) is 0.
	The storage data of the table of (d) exceeds 512.
	When the data set in (n) is other than the following, 1≤(n)≤512.
	When the FDEL(P) instruction is executed, the table position (n) of the deleted data is greater than data storage number.
4085H	When the device specified in the read application instruction (d) and (n) exceeds the corresponding device range.
4086H	When the device specified in the write application instruction (s) and (d) exceeds the corresponding device range.

Example



When X10 is ON, delete the second data in the data table of D100 to D105, and save the deleted data in D0. However, when the number of data saved is 0, do not execute the FDEL instruction. (The device range used in the data table is D100 to D107.)



6.15 IO Refresh instruction

6.15.1 REF/IO refresh

REF(P)

Perform a batch reset between the devices specified in (d1) and (d2) of the same type. It is used when interrupting operation, performing initial operation, or resetting control data.

Refresh n points at the beginning of the device specified in (s) to obtain or output external inputs

-[REF (s) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The start number of refreshed device	When using X and Y: The lowest bit number could only be 0; When using HSC: HSC0 to HSC7	Bit	ANY_BOOL
(n)	Refresh points	When using X and Y: It can only be the multiples of 8; When using HSC: 1 to 8	Unsigned BIN 16 bit	ANY16_U

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
REF	Parameter 1	●	●																								●
	Parameter 2												●	●	●	●	●	●	●	●			●	●			●

Features

- It is a function that only refreshes the corresponding device during a scan, and obtains or outputs external inputs.
- Acquisition of input and external output are performed in batches only after the END instruction of the program is executed, so pulse signals could not be output to the outside in one scan. When the I/O refresh instruction is executed, the corresponding input (X) or output (Y) will be forced to refresh during program execution, so pulse signals could be output to the outside in one scan.
- It can be used between FOR to NEXT and CJ instructions.
- It can be used to refresh the input and output in the interrupt subroutine to obtain the latest input information and output the operation result in time.
- The actual input port state change delay is determined by the filter time of the input components.
- The actual output port status change delay is determined by the response time of the output components (such as relays). The output contact during output refresh will act after the response time of the output relay (transistor).
- The response lag time of the relay output type is about 10ms (maximum 20ms), the transistor output type high-speed output port is about 10us, and the ordinary point output port is about 0.5ms.
- There will still be a certain delay when X0 to X17 filter time is set to 0.
- REF instruction could also refresh the value of high-speed counter HSC device. The value of high-speed counter is updated every 100us in normal use, if you need to get the latest values of high-speed counter immediately, you could use the REF instruction to do a count refresh. After the instruction is executed, the value in HSC device is the latest high-speed counter.

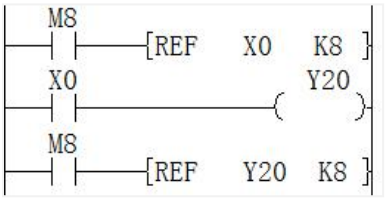
Error code

Error code	Content
4085H	The read address of (s) and (n) exceeds the device range. ✎**Note:** If (s)+(n) exceeds the maximum range of the device corresponding (s), an error will be reported
	(s) use numbered device whose low bit is not 0
4084H	When (s) use X and Y, (n) is not the multiples of 8 When (s) use HSC: (n) exceeds the range of K1 to K8

Error code	Content
2585H	Use REF instruction to refresh high-speed counter value, but there is no OUT HSC instruction to open the high-speed counter of the channel.

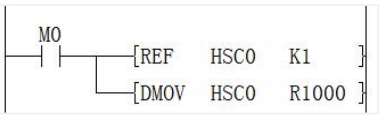
Example

(1) REF refreshes the X input or Y output



As in the example above, X0 to X7 can quickly update the input signal after M8 is turned on. After X0 triggers Y20, output Y20 to Y27 quickly through the next REF Y20 K8 instruction.

(2) REF refreshes the high-speed counter HSC



As in the example above, turn M0 OM, and refresh the current input pulse of high-speed counter, and store the latest high-speed counter value in HSC0, and store the current high-speed counter value in R1000 address.

6.15.2 REFF/Input refresh (With filter setting)

REFF(P)

Temporarily change the filter effect of the digital filter of X0 to X17 to (n)ms. (n) The range is 0 to 60ms.

-[REFF (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(n)	Filter value of X0 to X17	0 to 60	BIN16 bit	ANY16

Device used

instruction	Parameter	Devices																								Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
REFF	Parameter 1	●	●															●	●	●			●	●		●	●

Features

In programmable controller, X0 to X17 use a digital filter. The default filter time constant is set by SD2280 and SD2281, and the filter could be temporarily changed to 0 to 60ms through the REFF instruction.

When the high-speed counter or X input terminal interrupt function is used, the filter time of the relevant port is automatically the shortest time, and the filter time of the irrelevant port is still the original set value.

MOV instruction could also be used to directly assign to SD2280 and SD2281 to change the filter time, but it would not change the value of SD2280 and SD2281.

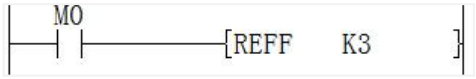
✎Note:

The X point filtering before this instruction may be out of control (if SD2280 and SD2281 are set to 0, the X point before the instruction will be completely out of control).

Error code

Error code	Content
4085H	The (n) read address exceeds the device range
4084H	(n) is not in the range of 0 to 60

Example



After M0 is turned on , the filter wave of X0 to X17 in the ladder program after the REFF instruction will temporarily be 3ms, and SD2280 and SD2281 would not change.

6.16 Timing measure instruction

6.16.1 DUTY/Clock pulse generation instruction

DUTY

Set the user's timing clock output destination (SM340 to SM344) specified in (d) to ON according to the number of scans specified in (n1), and set it OFF according to the number of scans specified in (n2).

-[DUTY (n1) (n2) (d)]

Content, range and data type

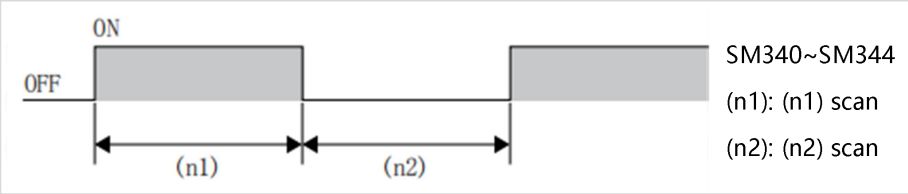
Parameter	Content	Range	Data type	Data type (label)
(n1)	The number of scans that set to ON	0 to 32,767	Unsigned BIN16	ANY16
(n2)	The number of scans that set to ON	0 to 32,767	Unsigned BIN16	ANY16
(d)	Special register for timing clock output destination	SM340 to SM344	Bit	ANY_BOOL

Device used

instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DUTY	Parameter 1											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	
	Parameter 3					●																				●	

Features

Set the user's timing clock output destination (SM340 to SM344) specified in (d) to ON according to the number of scans specified in (n1), and set it OFF according to the number of scans specified in (n2).



• The output destination special relay of the timing clock specified in (d) should be SM340 to SM344.

• Store the count value of the number of scans in SD340 to SD344 corresponding to the output destination special relay of the timing clock specified in (d).

• The count value of the number of scans, SD340 to SD344, becomes (n1) +(n2) or reset when the instruction input (instruction) is turned ON.

Special relay (d) for timing clock output	Device for counting the number of scans
SM340	SD340
SM341	SD341
SM342	SD342
SM343	SD343
SM344	SD344

• The operation starts at the rising edge of instruction input, and the output destination special relay of the timing clock is turned ON/OFF by the END instruction. Even if the instruction input is disconnected, the operation would not stop. It stops when it is STOP or the power is off.

• The cases where (n1) and (n2) are set to 0 are as follows.

The status of (n1) and (n2)	The ON/OFF status of (d)
(n1)=0, (n2)≥0	Fixed as (d)=OFF
(n1)>0, (n2)=0	Fixed as (d)=ON

• The related devices are shown below.

Special relay	Name	Content
SM340	Timing clock output 1	Timing clock output of DUTY instruction
SM341	Timing clock output 2	
SM342	Timing clock output 3	
SM343	Timing clock output 4	
SM344	Timing clock output 5	

Special register	Name	Content
SD340	Timing clock output 1 counts with scan number	DUTY instruction timing clock output 1 scan count count value
SD341	Timing clock output 2 counts with scan number	DUTY instruction timing clock output 2 scan count count value
SD342	Timing clock output 3 counts with scan number	DUTY instruction timing clock output 3 scan count count value
SD343	Timing clock output 4 counts with scan number	DUTY instruction timing clock output 4 scan count count value
SD344	Timing clock output 5 counts with scan number	DUTY instruction timing clock output 5 scan count count value

✎**Note:**

The DUTY instruction could be used up to 5 times (dots). However, the same timing clock output destination could not be used in multiple DUTY instructions.

Error code

Error code	Content
4084H	The written value of (n1) and (n2) exceed the range
4085H	The device address of (n1) and (n2) exceed the range
4086H	(d) is not in SM340 to SM344
408EH	(d) of multiple DUTY instructions use the same SM device

Example

Use the DUTY instruction to make Y0 flip once every cycle.



Set M0, SM340 will be ON for one cycle and OFF for one cycle.

6.17 Random number instruction

6.17.1 RND/Random number instruction

RND(P)

A pseudo-random number from 0 to 32767 is generated, and the value is stored as a random number in the device specified in (d).

-[RND (P) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(d)	The start number of the device storing random number	---	Signed BIN16	ANY16

Device used

instruction	Parameter	Devices																						Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RND	Parameter 1												●	●	●	●	●	●	●	●						●	●

Features

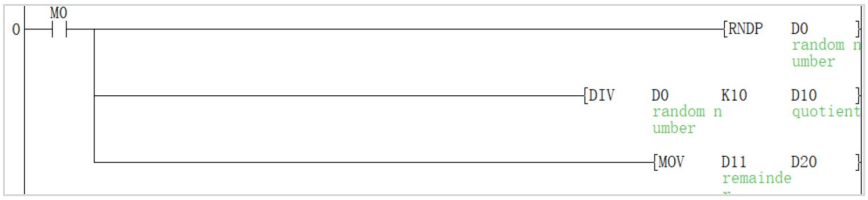
A pseudo-random number from 0 to 32767 is generated, and the value is stored as a random number in the device specified in (d).

Error code

Error code	Content
4086H	The write address of (d) exceeds the device range

Example

Pseudo-random numbers from 0 to 9 would be generated.



Turn on M0 to generate a pseudo-random number between 0-9.

6.18 Preferred instruction

6.18.1 DEXMN/Preferred instruction

DEXMN(P)

The position of the given value that is closest to the target value in multiple given values is selected through calculation.

-[DEXMN (s1) (s2) (s3) (d1) (d2)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Input data parameter array start device number		Form type	LIST
(s2)	Select the maximum number of data and the start device number of the output mode		Form type	LIST
(s3)	Target value	0 to 16777215	Unsigned BIN32	ANY32_U
(d1)	Select result array start device number		Form type	LIST
(d2)	Operation result array start device number		Form type	LIST

Device used

Instruction	Parameter	Devices																							Index modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DEXMN	Parameter 1																	●	●								●
	Parameter 2																	●	●								●
	Parameter 3																	●	●								●
	Parameter 4																	●	●								●
	Parameter 5																	●	●								●

Feature

(1) instruction function description

From the data set given by S1, Select the data combination whose number is less than or equal to S2 (select the maximum number of data) and the sum value is closest to S3 (target value). The selected result is stored in array D1 according to the position corresponding to array S1.

The error code of the instruction execution is stored in D2, and the number of the selected is stored in D2+2, and the difference between the selected array and the target value is stored in D2+4.

(2) Detailed parameter description

Input parameter S1				
Unit	Number of bytes	Features	Description	Range
s1	Double word	The number of input data	Specify the number of input data	1 to 32
s1+1				
s1+2	Double word	The first data	Input data	0 to 16777215
s1+3				
s1+4	Double word	The second data		
s1+5				
. . .	. . .	. . .		
s1+64	Double word	The 32nd data		
s1+65				

Input parameter S3				
Unit	Number of bytes	Features	Parameter Description	Range
s2 s2+1	Double word	Specify the maximum number of selected data	Specify the maximum number of selected data	1 to s1. Due to the time limit, please refer the notes.
s2+2 s2+3	Double word	Output mode	Output mode selection: 0: the 0 value in the input array is not added to the output combination 1. Add the 0 value in the input array to the output combination	

Input parameter S3				
Unit	Number of bytes	Features	Parameter Description	Range
s3 s3+1	Double word	Specify target data	Specify the selected target data	0 to 16777215

Parameter d1				
Unit	Number of bytes	Features	Parameter Description	Range
d1 d1+1	Double word	Selection result, each bit represents a data	Use bits to indicate the position of data Bit 0 corresponds to S1+2 Bit 1 corresponds to S1+4	Read-only
d1+2 d1+3	Double word	Select the position of the 1st data	The position of the data, relative to the offset of S1+2, 0 means the data is in S1+2 3 means the data is in S1+8 And so on	Read-only
d1+4 d1+5	Double word	Select the position of the 2nd data		
d1+N*2+2 d1+N*2+3	Double word	Select the position of the Nth data		
d1+64 d1+65	Double word	Select the position of the 32th data		

Parameter d2				Remarks	Remarks
Unit	Number of bytes	Features	Parameter Description		
d2 d2+1	Double word	Select the execution result OR Error code	Error code	≥0: execute correctly <0: Error code	-1: The number entered is out of range -2: The number of selected data is out of range -3: The target data is out of range -4: The input data is out of range (If there are repeated errors, the top error will be reported first. For example, the number of inputs is 0 and the target data is -30. At this time, an error of -1 will be reported)
d2+2	Double	The number	The number of Actually	Read-only	

Parameter d2				Remarks	Remarks
Unit	Number of bytes	Features	Parameter Description		
d2+3	word	selected	selected data		
d2+4	Double word	Minimum deviation	The current result minus the target value	Read-only	
d2+5					
d2+6	Double word	Operation time	Total time used (ms)		
d2+7					
d2+8	Double word	Reserved	Number of combinations (where the result is)		
d2+9					
d2+10 to d1+73	Double word	Internal use	Cache for internal calculation		

✎Note:

In the case of a large number of data, a watchdog timeout may occur. This is because the calculation takes a lot of time.

The current timetable for this instruction is as below. Please use the maximum number of data selected according to the timetable.

The number of arrays	1	2	3	4	5	6	7	8
The number of selected	Time unit (ms)							
1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
2		<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
3			<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
4				<0.1	<0.1	<0.1	<0.1	<0.1
5					<0.1	<0.1	<0.1	<0.1
6						<0.1	<0.2	<0.2
7							<0.2	<0.2
8								<0.2

The number of arrays	9	10	11	12	13	14	15	16
The number of selected	Time unit (ms)							
1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
2	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
3	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.2	<0.2
4	<0.1	<0.1	<0.2	<0.2	<0.2	<0.4	<0.4	<0.4
5	<0.2	<0.2	<0.3	<0.4	<0.5	<0.6	<0.9	1.1
6	<0.2	<0.3	<0.3	<0.6	<0.9	1.2	1.8	2.5
7	<0.2	<0.4	<0.6	<0.9	1.4	2.2	3.4	5.9
8	<0.3	<0.4	<0.7	1.4	2.0	3.3	5.5	8.9
9	<0.3	<0.5	<0.9	1.5	2.6	4.6	7.9	13.4
10		<0.6	<1.0	1.7	3.2	5.8	10.4	18.4
11			1.1	2.2	3.7	6.9	12.8	23.3
12				2.3	4.4	8.2	15.2	28.8

13					4.9	9.3	17.5	32.8
14						10.5	19.8	37.5
15							23.0	43.0
16								46.8
The number of arrays	**17**	**18**	**19**	**20**	**211**	**22**	**23**	**24**
The number of selected	**Time unit (ms)**							
1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
2	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
3	<0.3	<0.3	<0.3	<0.3	<0.3	<0.3	<0.3	<0.3
4	<0.5	<0.6	<0.7	<0.8	<0.9	1.2	1.3	1.5
5	1.4	1.8	2.3	2.9	3.7	4.6	5.7	6.9
6	3.6	5.0	6.8	9.9	12.2	15.9	20.8	26.8
7	7.7	11.4	16.5	23.5	32.9	45.7	62.5	84.4
8	14.2	22.9	33.4	51.2	75.8	110.6	158.9	225.1
9	22.6	37.2	60.2	95.3	148.6	227.6	342.9	
10	32.2	55.4	93.8	156.9	255.2			
11	42.4	75.3	132.2	222.8				
12	51.8	94.9	171.9					
13	61.4	114.3	221.2					
14	70.7	133.2						
15	80.6	151.9						
16	89.4	170.5						
17	98.6	189.7						
18		207.7						

The number of arrays	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**
The number of selected	**Time unit (ms)**							
1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
2	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
3	<0.3	<0.3	<0.4	<0.4	<0.5	<0.5	<0.6	<0.6
4	1.7	1.9	2.3	2.6	3.0	3.5	3.8	4.4
5	8.4	10.2	12.3	14.7	17.5	20.6	24.2	28.3
6	34.2	43.3	54.2	15.9	83.3	102.2	124.3	150.5
7	112.6	148.6	194.2	251.4	322.6	410.5	515.4	649.8
8	314.4	433.7	591.6					
9								

✎**Note: R**ed text is the limit of exceeding the default scan cycle.

Error code

Error code	Content
4084H	Data range error. For details, see the error code of parameter d2
4085H	The device addresses of (s1), (s2) and (s3) are out of range
4086H	The device addresses of (d1) and (d2) are out of range

Example



When D200 = 8, D4 = 2, it means to take out two data from 8 groups of data, and the sum of the two data is closest to the data in D8.

Array data of D200 (S1):

D200	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	8
D201	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D202	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	150
D203	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D204	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	99
D205	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D206	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	56
D207	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D208	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	200
D209	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D210	0	1	1	0	1	0	1	0	0	0	0	0	0	0	0	86
D211	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D212	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	76
D213	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D214	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	160
D215	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D216	1	0	0	1	1	0	0	1	0	0	0	0	0	0	0	153
D217	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Select a combination with a sum close to 300 from the data above, and the results selected by D300 (D1) are as below:

Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D300	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	10
D301	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D302	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
D303	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D304	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
D305	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D306	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D307	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D308	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D309	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D310	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D311	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D312	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit 1 and Bit 3 of D300 are 1, and the data positions of 1 and 3 are currently selected. The indicated positions are D204 (99) and D208 (200).

D400 (D2) running results are as below:

Device	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
D400	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D401	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D402	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
D403	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D404	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1
D405	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1
D406	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
D407	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D408	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	255
D409	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D410	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D411	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D412	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D413	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D414	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D415	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D416	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D417	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D418	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D419	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

If D400 is 0, the execution is correct.

If D402 is 2, the number of selected is 2.

If D404 is -1, the selected data combination sum value minus the target value difference is -1.

If D406 is 2, the use time is 2ms.

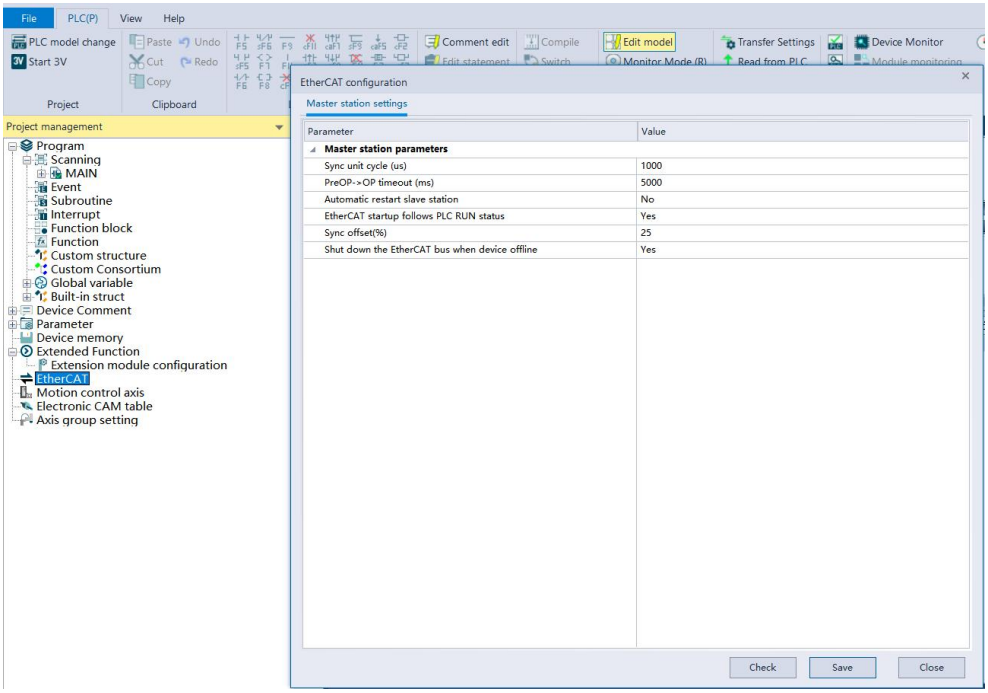
7 EtherCAT

7.1 Description

EtherCAT is an open industrial field technology based on Ethernet, which has the characteristics of short communication refresh cycle, small synchronization jitters, and low hardware cost.

7.2 Master station configuration

Double-click the EtherCAT node of the project management tree to pop up the EtherCAT configuration, modify the configuration and save it.



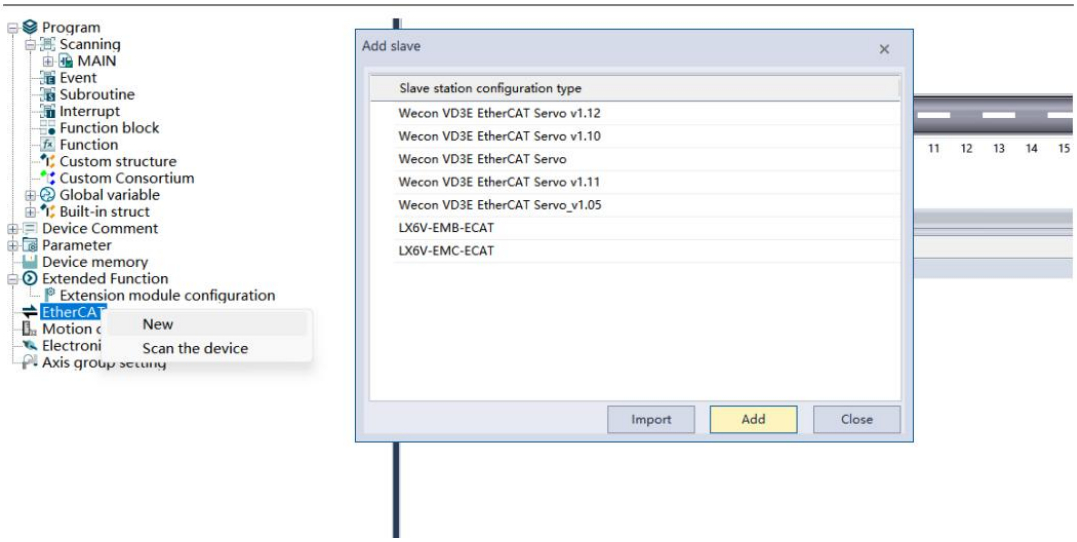
Configuration description

- Synchronous unit cycle: how long does it take to synchronize data.
- PreOP->OP timeout: the timeout from PreOP state to OP state during the initialization of OP;
- Automatically restart slave station: after it is enabled, if the slave is offline or powered off, it will try to recover to OP status;
- EtherCAT enables following PLC RUN status: after it is enabled, PLC RUN triggers EtherCAT to execute the process to OP;
- Synchronous offset: the percentage of time offset allowed during synchronization;
- Offline shutdown of EtherCAT bus: if a device is offline, the EtherCAT function will be disabled and all stations will switch to PreOP state. If automatic restart of slaves is enabled, it will be all restarted.

7.3 Import and add slaves

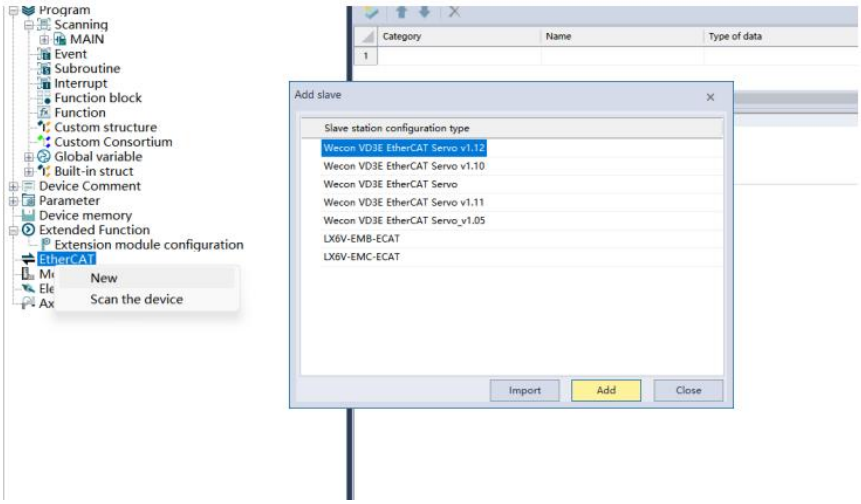
7.3.1 Import configurations

Right-click the EtherCAT node in the project management tree, click New, click the Import button in the pop-up Add Slave window, and select the EtherCAT xml file of the corresponding manufacturer to import.



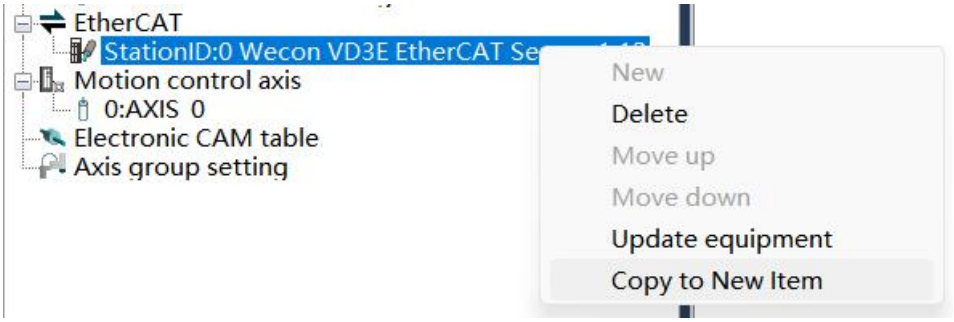
7.3.2 Add a slave station

Right-click the EtherCAT node in the project management, click New, select a slave type and click Add, a new slave node will be added to the EtherCAT node in the project management, and click Close to end the operation of adding a slave.



7.3.3 Copy to new item

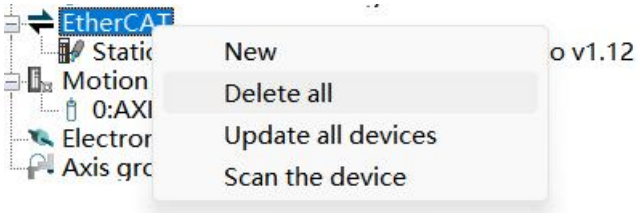
Right-click to select the slave node under the EtherCAT node in the project management, click “Copy to New Item”, and create a new slave node with exactly the same configuration as the selected node.



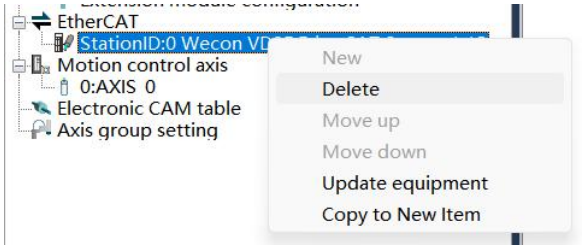
7.4 Slave station configuration

7.4.1 Delete

① Right-click the EtherCAT node, select [Delete all], and delete all currently created slave station configurations.

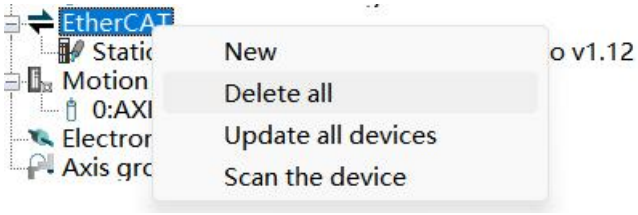


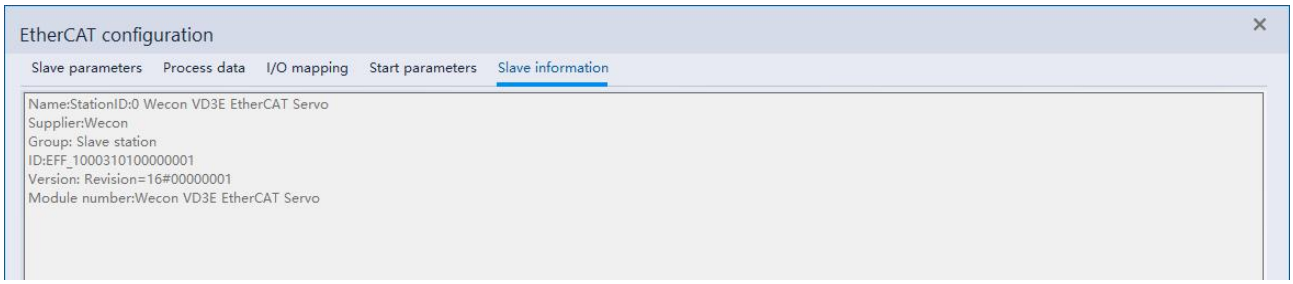
② Right-click the slave station under the EtherCAT node, select Delete, and delete the current slave station configuration.



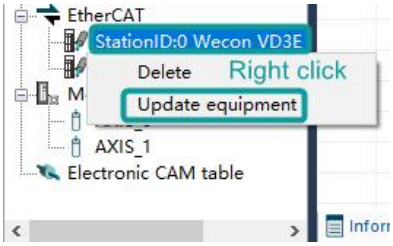
7.4.2 Update devices

① Right-click the EtherCAT node and select [Update All Devices] to update the configuration information of the current EtherCAT node to the newly imported EtherCAT configuration file information.

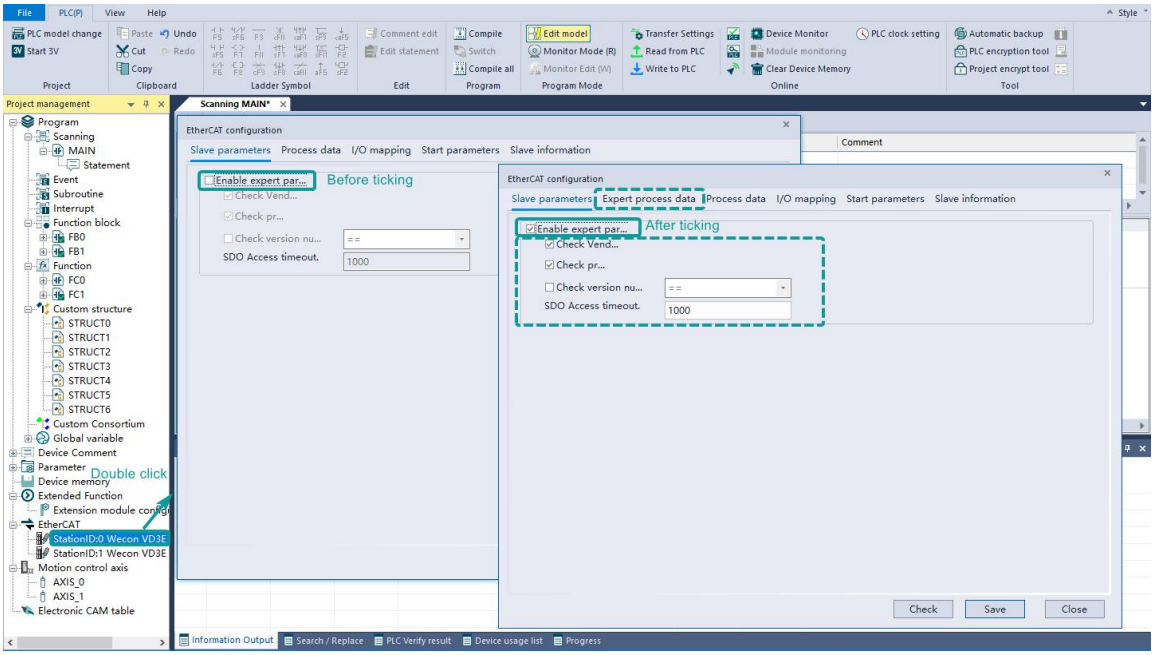




② Right-click the slave station under the EtherCAT node, select [Update equipment], and update the configuration information of the corresponding slave station node to the newly imported EtherCAT configuration file information.



7.4.3 Slave station parameters



Double-click the slave station under the EtherCAT node, and the EtherCAT configuration dialog box will pop up. In the slave station parameter interface, check the enable expert parameter to use the expert process data interface and the check box and edit box below.

▪Check Vendor ID: During activation, check whether the downloaded slave configuration is consistent with the slave vendor ID connected to the PLC;

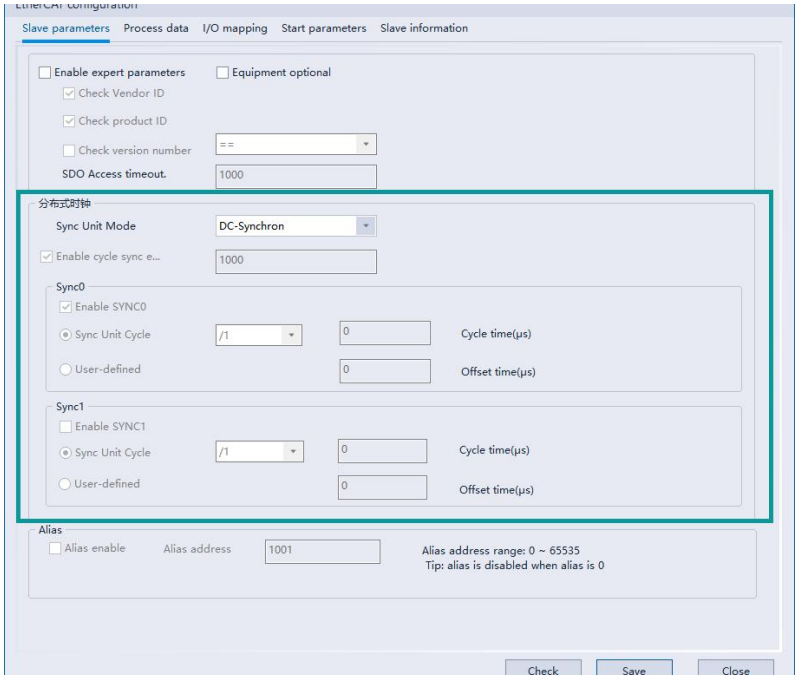
▪Check Product ID: During activation, check whether the downloaded slave configuration is consistent with the slave product ID connected to the PLC;

▪Check version number: Check whether the version numbers are consistent when activated;

▪SDO access timeout: Set the SDO access time.

7.4.3.1 Distributed clock

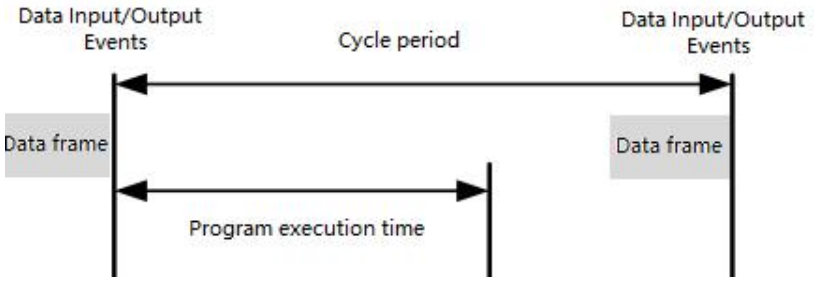
This part is used to set the synchronous operation mode of the slave station. The interface is as follows:



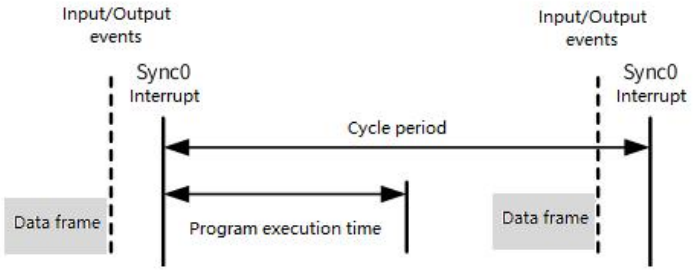
Sync unit mode: Generally, EtherCAT slave stations have FreeRun mode, running mode synchronized with input and output events

(SM-Synchron), and running mode synchronized with distributed clock (DC-Synchron). Depending on the selected slave station, the options supported by the synchronous unit mode will be different.

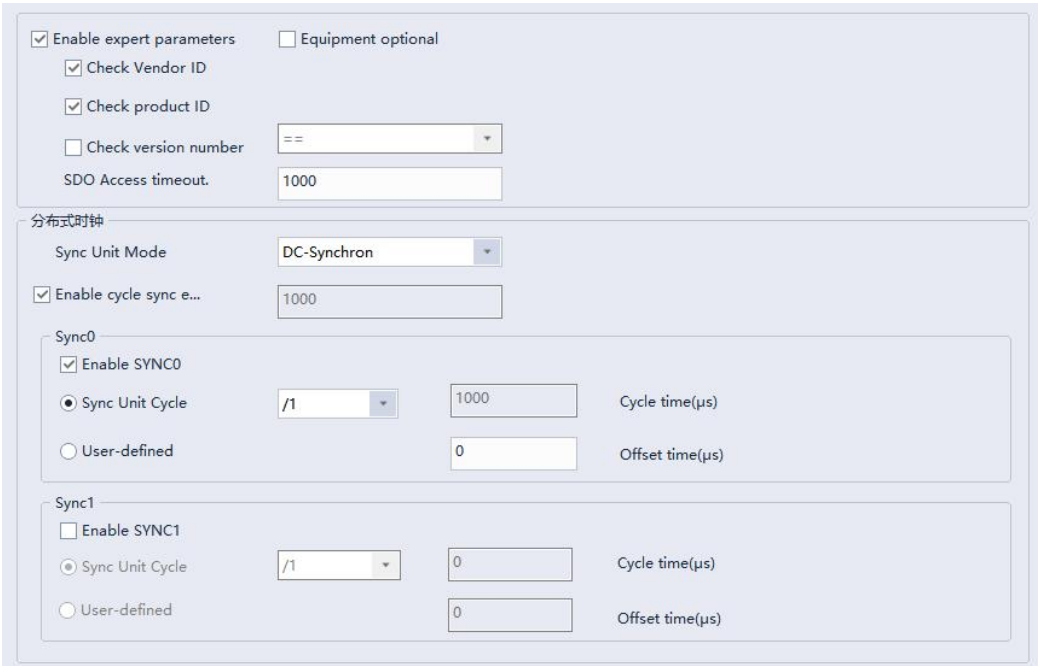
Take the Wecon VD3E EtherCAT Servo V1.11 module as an example. This module supports SM-Synchron mode, and neither SYNC0 nor SYNC1 can be set. There is only one interrupt of data input and output events inside the clock slave station. The processing logic inside the slave station is shown in the following figure:



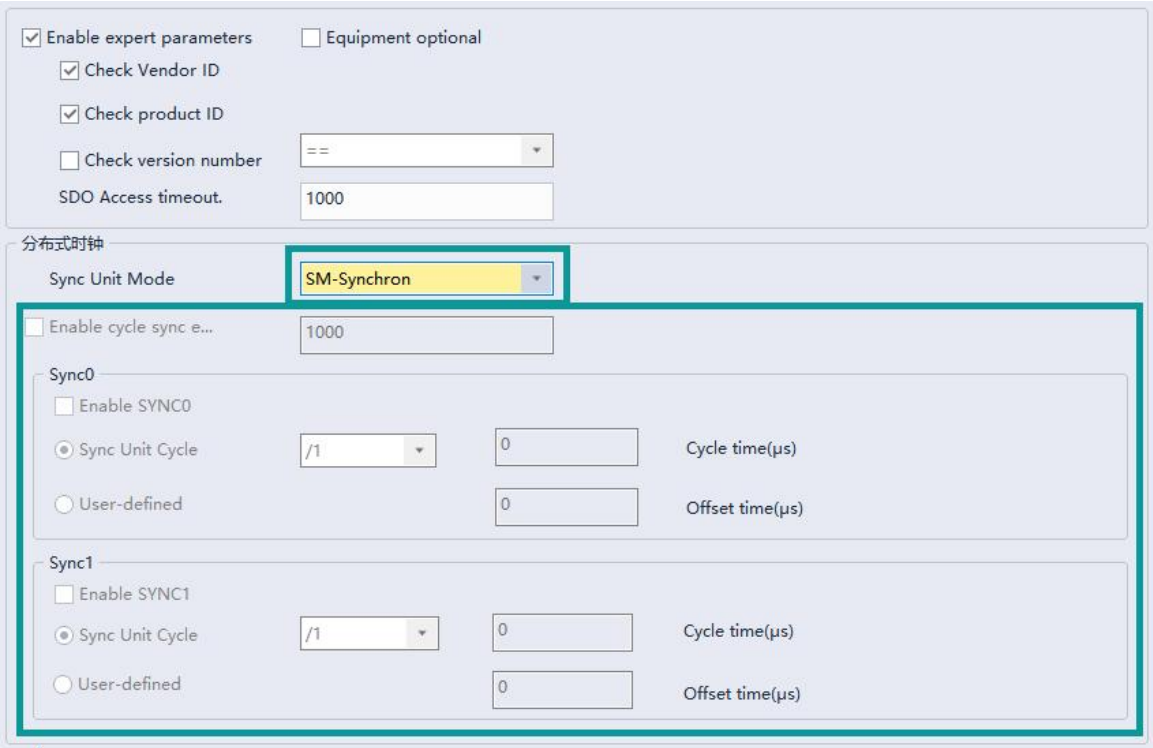
At the same time, this module also supports DC-Synchron mode. In this mode, the Sync interrupt of the slave station can be configured. DC synchronization events are enabled by default, and SYNC0 interrupt is enabled. The cycle of SYNC0 interrupt is the same as the synchronization cycle of the EtherCAT master station. SYNC1 interrupt is not enabled.



When "Enable expert parameters" is checked, select DC mode in the drop-down box, such as "DC-Synchron", and check "Enable cycle synchronization event". SYNC0 and SYNC1 groups will be restored from the disabled state to the enabled status, and SYNC0 enable button will be checked by default. The configuration is shown in the figure below:



The drop-down box selects a non-DC mode, such as "SM-Synchron", the SYNC0 and SYNC1 groups are all grayed out, and the cycle time and offset time are both 0.

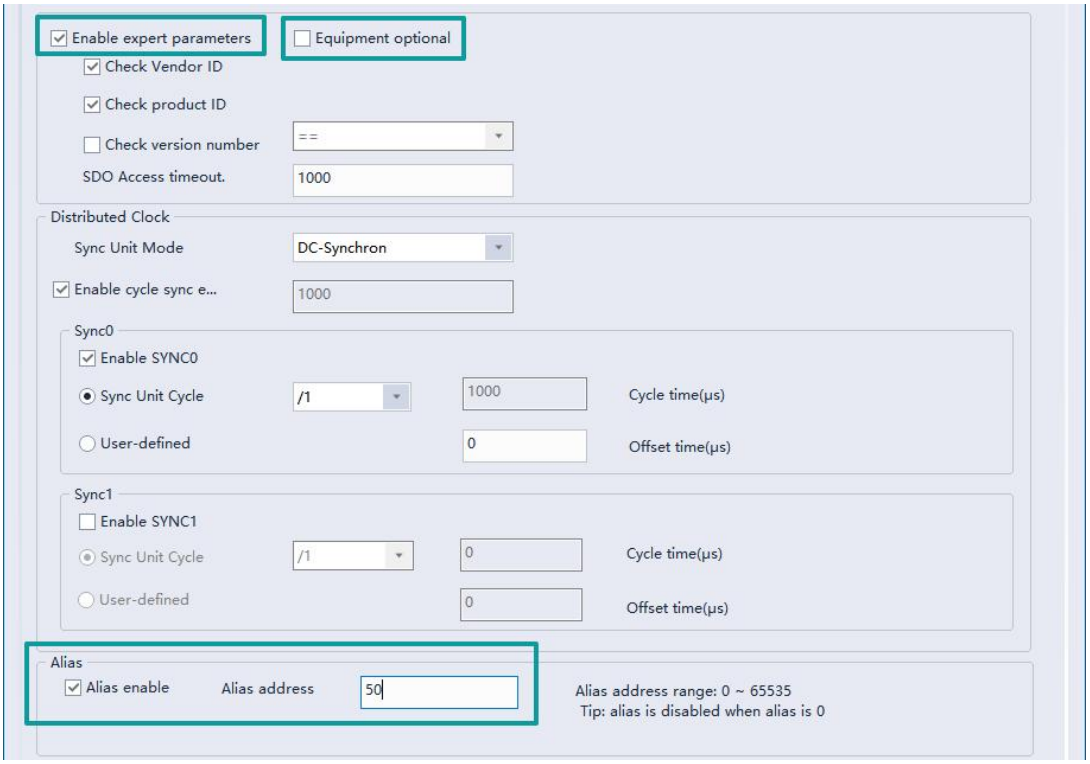

Enable expert parameters
Equipment optional
Check Vendor ID
Check product ID
Check version number
==
SDO Access timeout.
1000
分布式时钟
Sync Unit Mode
SM-Synchron
Enable cycle sync e...
1000
Sync0
Enable SYNC0
Sync Unit Cycle
/1
0
Cycle time(μs)
User-defined
0
Offset time(μs)
Sync1
Enable SYNC1
Sync Unit Cycle
/1
0
Cycle time(μs)
User-defined
0
Offset time(μs)

7.5 Alias

7.5.1 Alias parameter configuration

Check [Enable expert parameters] and [Equipment optional] to use [Alias] function.

After [Expert parameters] enables, you can choose whether you need alias. After [Equipment optional] enables, alias forcibly enables.



[Equipment optional] means that this device is an optional device and may not exist. When the device cannot be found, it will not run. If MC_POWER runs, it will not be successful and report error. [Alias enable] is forcibly turned on, that is, when the corresponding device must be found through alias and the product information must be checked, the product must be matched before it can be recognized normally.

If it is an alias set in [Expert mode], even if the alias is not found, the equipment of the corresponding model can run normally if it is found in order.

Example

① 5 devices: a, b, c, d, e. The link order is e, d, a, c, b, which will be searched according to aliases, and all of them correspond to configuration order. It will run normally.

Configuration Order	Alias	Link Order	Alias
a	1	e	5
b	2	d	4
c	3	a	1
d	4	c	3
e	5	b	2

② 5 devices: a, b, c, d, e. Check [Equipment optional], and the order of links is d, a, c, and b. When using e, the corresponding FB will report an error, but it will not affect the use and operation of other devices;

Configuration Order	Alias	Link Order	Alias
a	1		
b	2	d	4
c	3	a	1
d	4	c	3
e	5	b	2

③ 5 devices: a, b, c, d, e. C and E turn on [Device optional], and the link order is e, d, c, b. Since a does not turn on [Device optional] and cannot be found at the same time, When OP, it cannot be OP, the error "Device not found" will be reported.

Configuration Order	Alias	Link Order	Alias
a	1	e	5
b	2	d	4
c	3		
d	4	c	3
e	5	b	2

✎Note:

- If it is an alias set in expert mode, even if the alias is not found, the device of the corresponding model can run normally if it is found in order.

Example

5 devices: A, B, C, D, E. The link order is e, d, a, c, b.

① The aliases b, d, and e are all corresponding. At this time, a and c will be found if they do not match. (After b, d, and e are matched, pull out these three units from both sides, and then it becomes: Configure the order, a, c, and find the link sequence so that a and c can be found.

Configuration Order	Alias	Link Order	Alias
a	1	e	5
b	2	d	4
c	3	a	10
d	4	c	30
e	5	b	2

② If the aliases of a, d, and e correspond, b and c cannot be found if they do not match. (After the same removal, configure the order, b, c, search link sequence. So, c, b can't be found.)

Configuration Order	Alias	Link Order	Alias
a	1	e	5
b	2	d	4
c	3	a	1
d	4	c	30
e	5	b	20

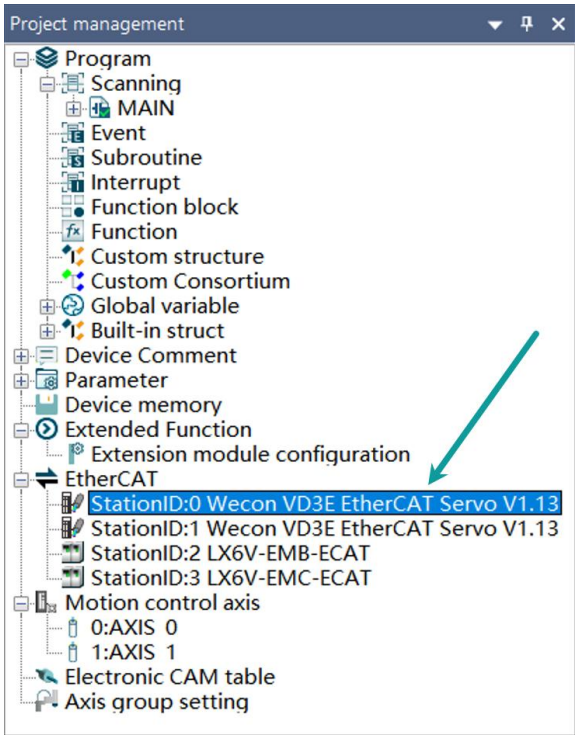
✎Note:

- If [Expert mode] and [Equipment optional] are both checked, it will run according to [Equipment optional];
- At present, the first version does not support the situation that there are more linked devices than the configured ones, and there are more devices that can be linked normally;
- For example, link 5 devices configured5 units, if one of them is set with [Device Optional] but there is no correspondence, it cannot be enabled at this time;

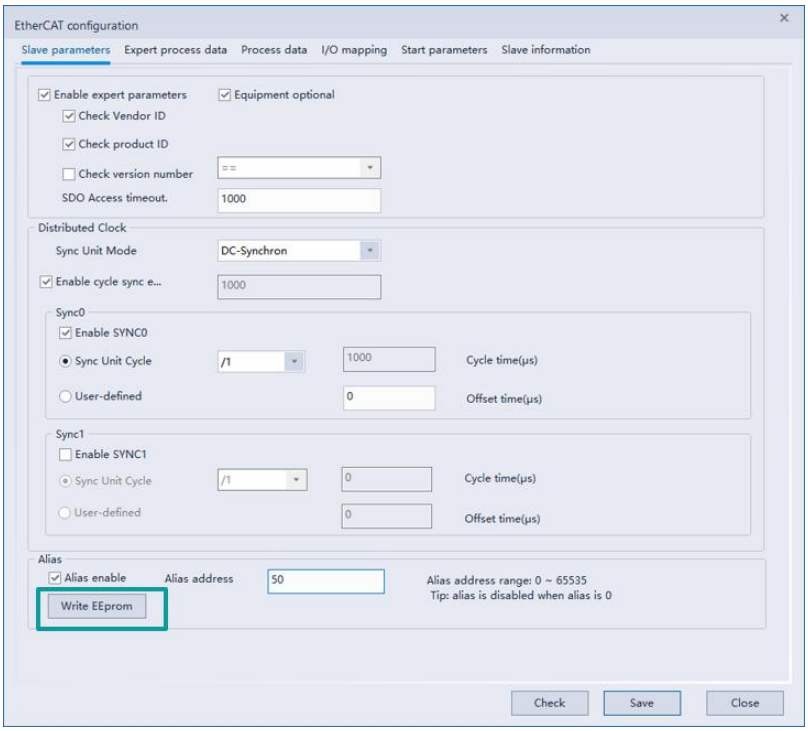
- In addition, uncheck [Manufacturer id] and [Model check], although in some cases you can only check alias correspondence, and sequential matches are forced. But there is no guarantee that it will operate normally;
- Except 0 aliases, aliases that can be normally corresponded are not allowed to be duplicated. Before OP, this section will be checked (SM1900 off->on Instantaneous detection), for example, during the process, it is allowed to set two identical aliases to the slave station, but after setting OP again(instruction or STOP RUN Errors will also be reported.

7.5.2 Alias writes to servo

In monitoring mode, click on the EtherCAT device interface in the project management interface.



At this time, the following interface appears. Different from the non-monitoring mode, [Write EEprom] appears. This button writes the alias to the corresponding servo.



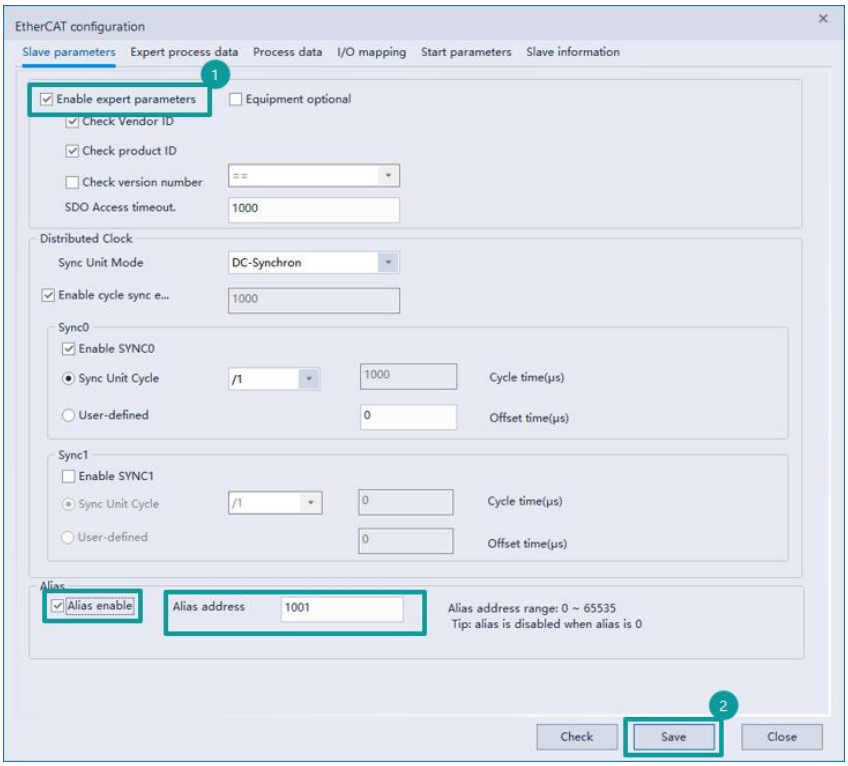
The execution rules for this key are:

- After OP, (SM1902 ON) will be set according to the device corresponding to the configuration. If there is no correspondence, it will prompt that the setting failed;
- Before OP, (SM1902 OFF) is aliased according to the link sequence Regardless of whether the model corresponds or not, set it directly. If there is no correspondence, the setting will fail.

7.5.3 Simple configuration instance

When an EtherCAT device configuration exists, open the EtherCAT configuration of the corresponding slave station.

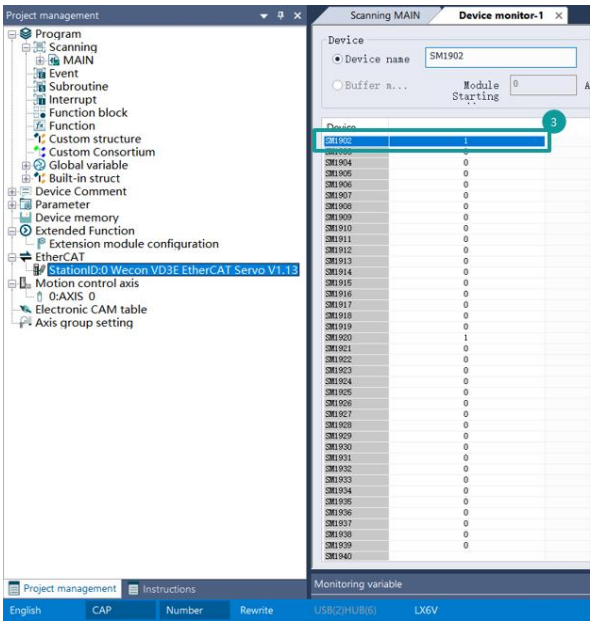
1Check [Start Expert Parameters] and [Alias Enable], and then configure the corresponding alias to be written into the "Alias Address"; 2 Save the configuration and download it to Run in PLC;

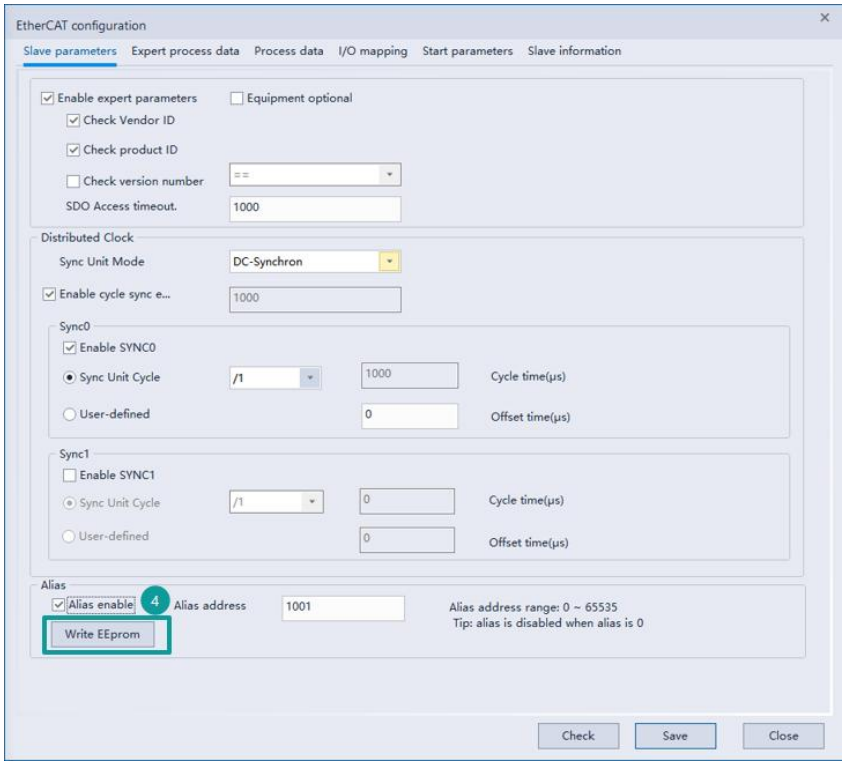


When the link sequence corresponds, the corresponding servo can still be matched.

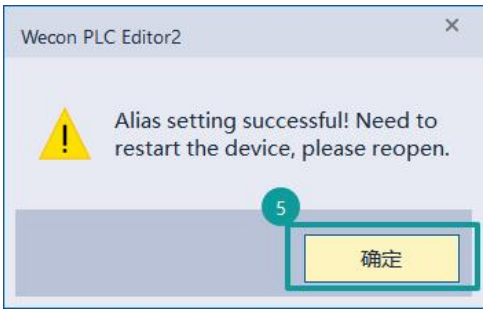
③ Open the software device monitoring interface and observe that SM1902 is ON;

④ Open the EtherCAT configuration interface, click [Write EEprom], and write the alias to Servo;

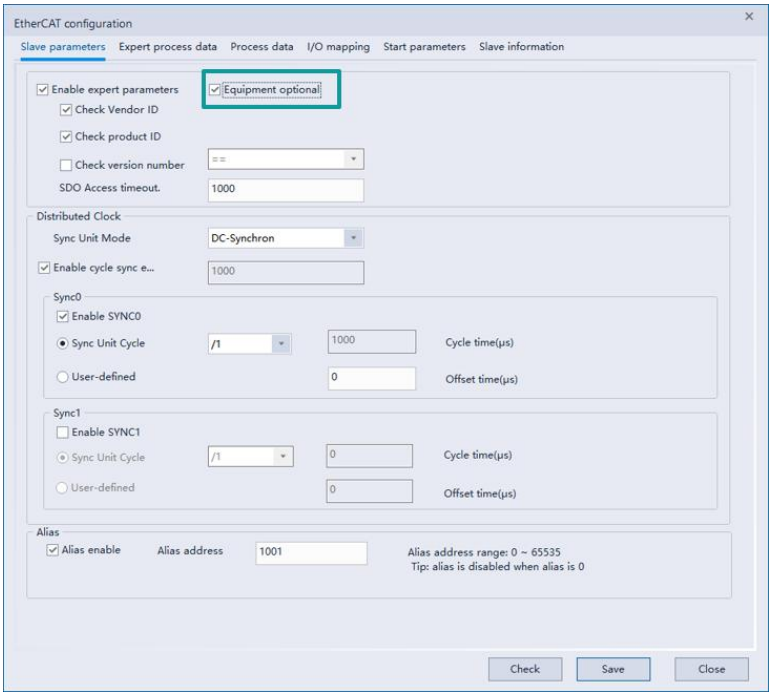




⑤ Restart the servo and PLC can be matched by alias.



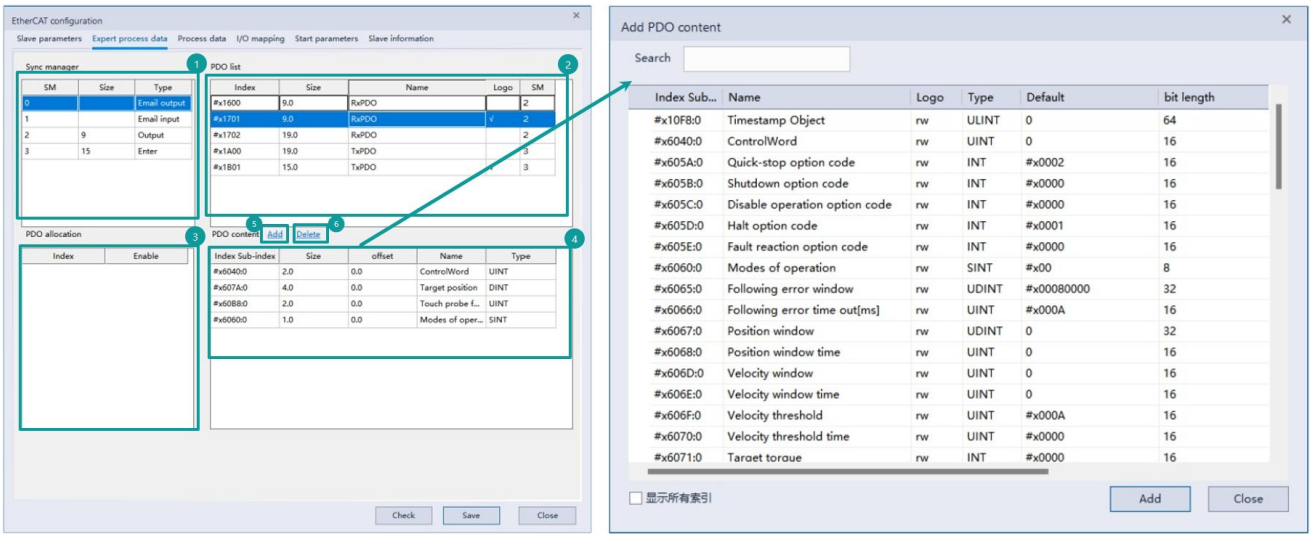
Then check [Device optional] in edit mode.



After removing the linked EtherCAT device and re-downloading the ladder diagram that checks the optional device, the ladder diagram can still start normally. However, MC_POWER corresponding to the axis cannot be started (but no error will be reported), and the IO mapping does not work.

Similarly, in the ladder diagram, connect the corresponding device, then stop->run the PLC again and the corresponding device can be used normally.

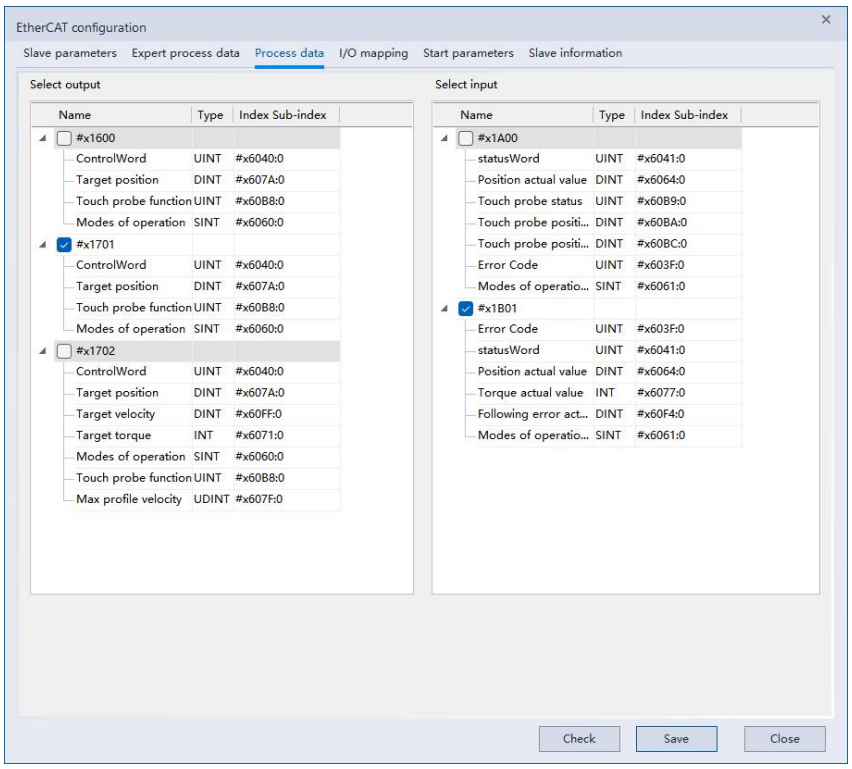
7.5.4 Expert parameters



- Select ① Synchronization Manager table area, select input and output, ② PDO allocation table area will be updated to synchronization manager output or the PDO allocation list corresponding to input;
- Check or uncheck “enable ② PDO allocation table area”, which will be fed back to the ③ PDO list;. Select the checked row in the ③ PDO list to update the ④ PDO content table;
- In the ④ PDO content table, right-click to select Add or click the ⑤ Add to add Select the content to be added in the PDO content interface and click Add;
- After selecting the content to be deleted in the ④ PDO content table, right-click the menu to select Delete or directly click the ⑥ Delete button to delete it.

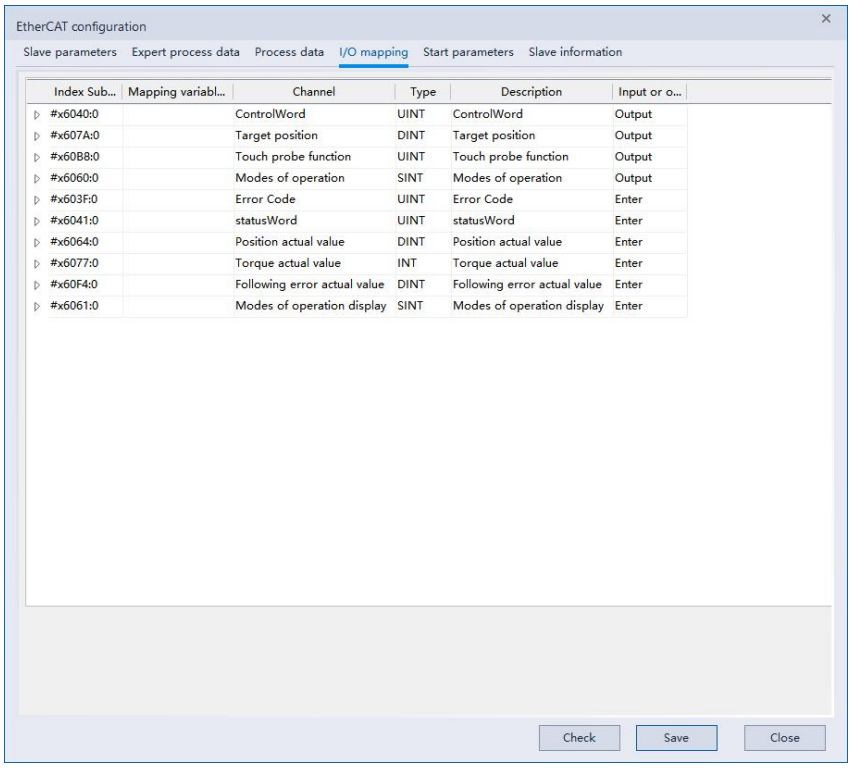
7.5.5 Process data

The process data interface is basically consistent with the expert process parameter functions except that PDO content cannot be customized.

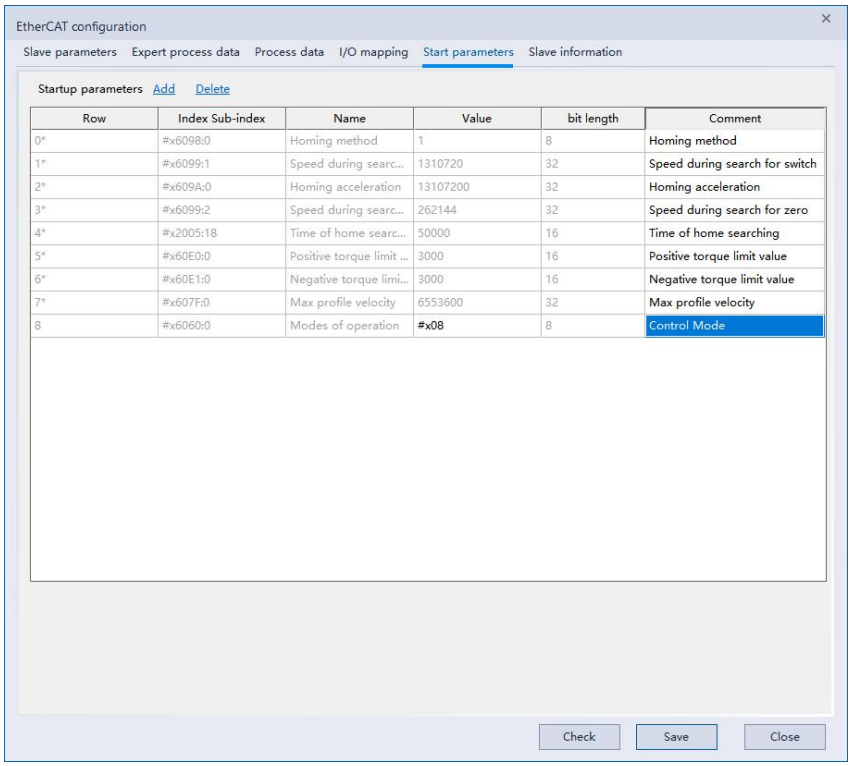


7.5.6 I/O mapping

For PDO content set by process parameters, you can select device or variables to map.



7.5.7 Start parameters



① Add

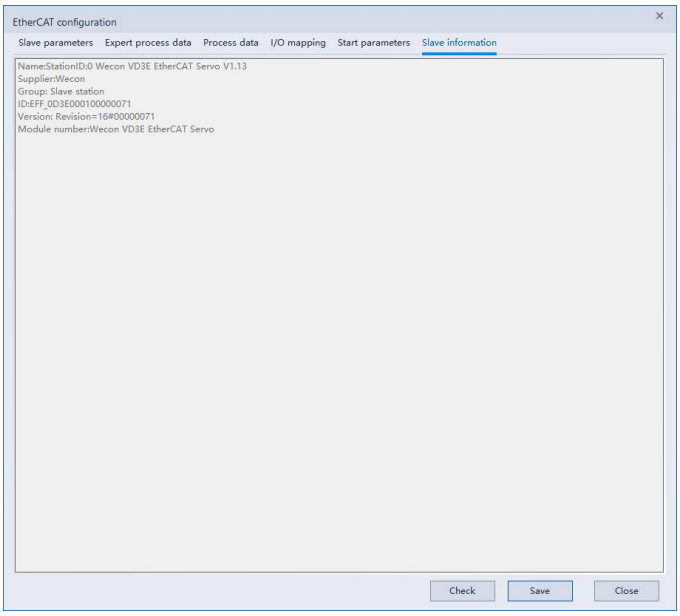
Right-click the menu to select [Add] or directly click [Add] to enter the interface for adding startup parameters, select the corresponding item, and add startup parameters.

② Delete

After selecting the parameter content to be deleted, right-click the menu to select delete or directly click “Delete”.

7.5.8 Slave station information

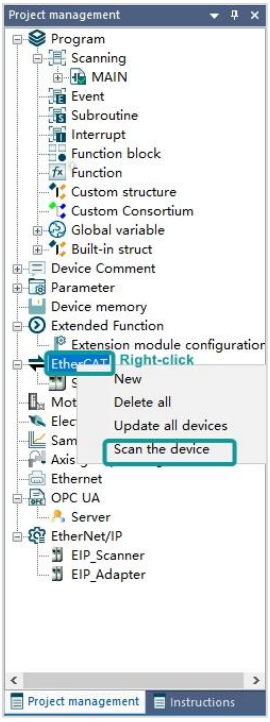
Display slave station information.



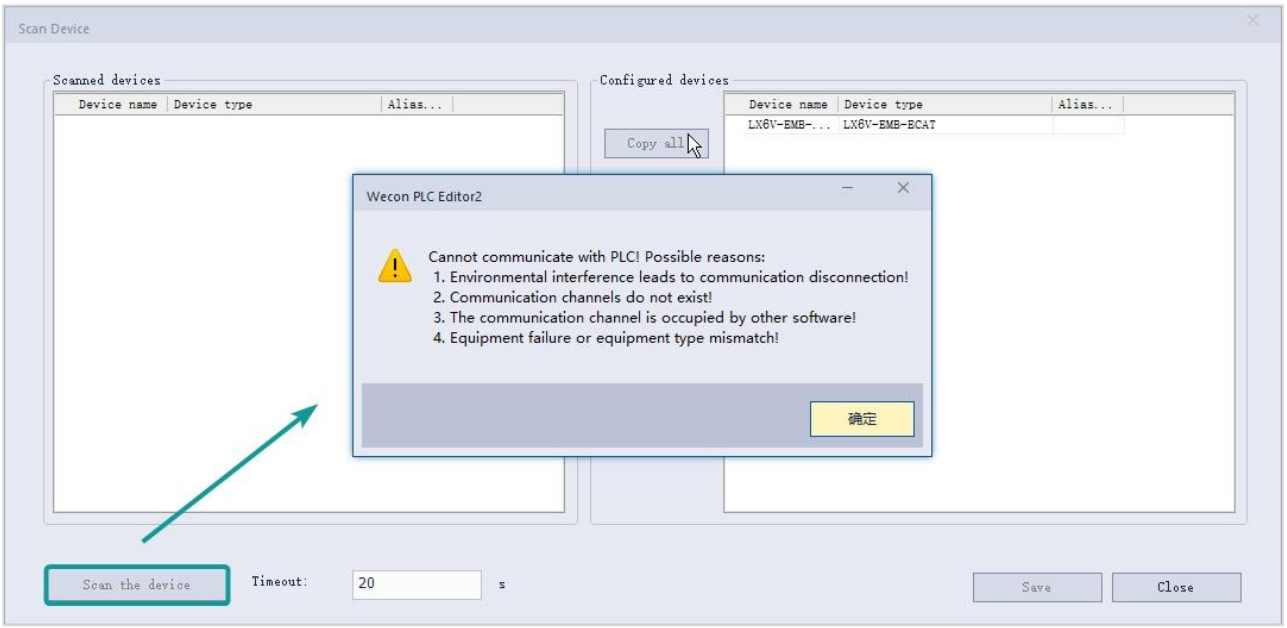
7.6 Device scanning

EtherCAT's device scanning supports direct scanning after connecting or unloading the device.

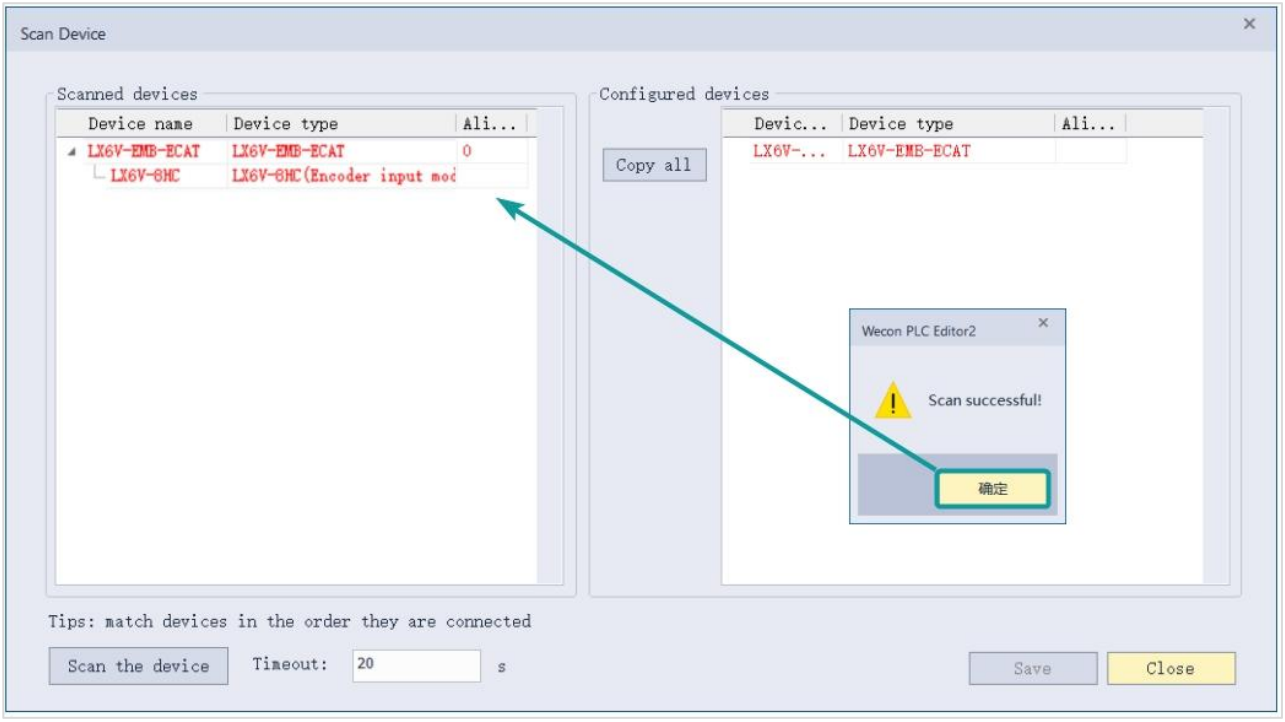
① Select the EtherCAT label, right-click the mouse button and select the scanning device in the pop-up menu.



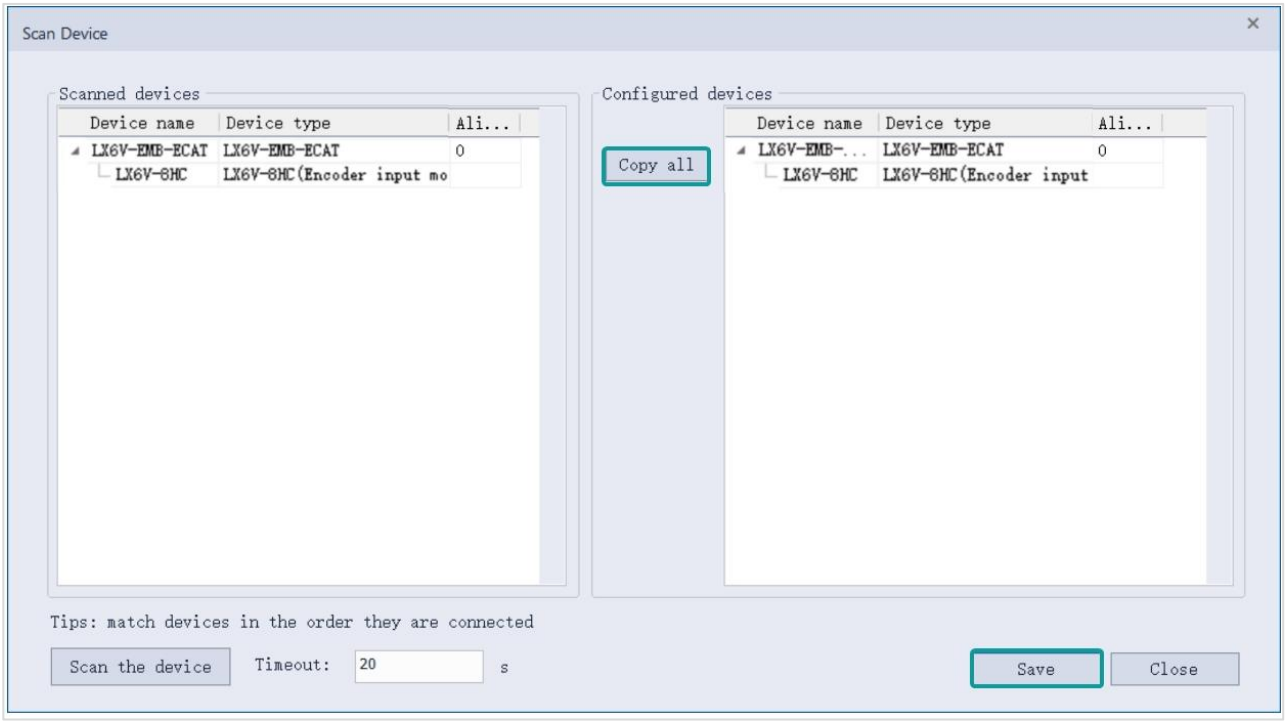
② Click the "Scan the device" button in the pop-up window. If the PLC is not connected, an error will be prompted, as shown below;



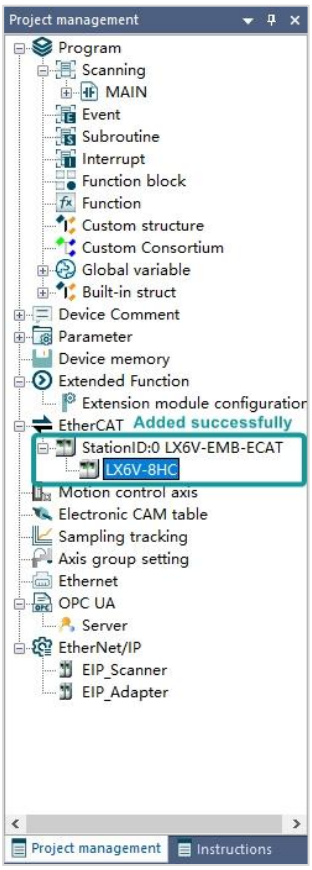
③ After waiting for the scan to complete, the scanned slave devices can be seen in the left side view.



④ Click [Copy all] to update the scanned devices to the currently configured configuration list, and then click [Save] to update the scanned devices to the configuration list of the project. After saving, click [Close] to update to the configuration list.

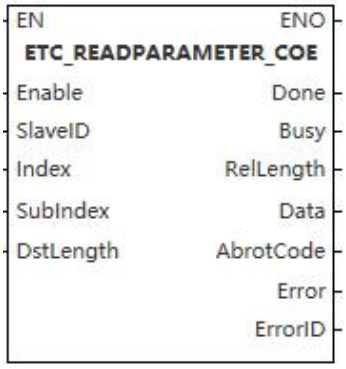


✎**Note:** Close the window if it is not saved, the scanned devices will not be added to the configuration list of the project!


Project management
Program
Scanning
MAIN
Event
Subroutine
Interrupt
Function block
Function
Custom structure
Custom Consortium
Global variable
Built-in struct
Device Comment
Parameter
Device memory
Extended Function
Extension module configuration
EtherCAT
Added successfully
StationID:0 LX6V-EMB-ECAT
LX6V-8HC
Motion control axis
Electronic CAM table
Sampling tracking
Axis group setting
Ethernet
OPC UA
Server
EtherNet/IP
EIP_Scanner
EIP_Adapter
Project management
Instructions

7.7 EtherCAT instruction

7.7.1 ETC_READPARAMETER_COE



This function is mainly used to read the SDO parameters of the servo.

Content, range, data type

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
ETC_READPARAMETER_COE	Input	Enable	BOOL	No	TRUE/FALSE	Read Servo SDO enable
		SlaveID	WORD	No	0 to range of EtherCAT slaves that can be linked	Set according to the ID configured in the interface. If the device is optional, an error will be reported even if the device does not actually exist
		Index	WORD	No	0 to 65535	Index, fill in according to the information provided by the servo
		SubIndex	WORD	No	0 to 65535	Sub-index, fill in according to the information provided by the servo
		DstLength	WORD	No	-	Data length, fill in according to the information provided by the servo.
	Output	Done	BOOL	Yes	TRUE/FALSE	When the value is obtained, Done will be set to TRUE, and if Done is not read, it will be set to FALSE
		Busy	BOOL	Yes	TRUE/FALSE	Busy is TRUE during reading
		RelLength	WORD	Yes	-	Actual length
		Data	DWORD	Yes	-	Read data
		AbrotCode	DWORD	Yes	TRUE/FALSE	Interrupt code returned by the servo, refer to the EtherCAT standard document for information
		Error	BOOL	Yes	TRUE/FALSE	Error
		ErrorID	DWORD	Yes	See error code for details	Error code

Device used

Instruction	Parameter	Device																				Offset modification	Double word extension	Pulse-type expansion
		X	Y	M	S	SM	T(Bit)	HSC(Bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	DXX	XXP
ETC_READ PARAMETER_COE	Enable	●	●	●	●	●																				
	SlaveID																									
	Index																									
	SubIndex																									
	DstLength																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	RelLength																									
	Data																									
	AbrotCode																									
	Error		●	●	●	●																				
	ErrorID																									

- Used to read the address values of the servo index and subindex, with the correct length.
- Function:

① When Enable is continuously set to ON, it will continuously read the SDO value into Data. Upon receiving a value, it will set Done to ON and append the new value to Data. If no value is read, Done will be set to OFF.

② When Enable is triggered by an edge, it will read data once. During this process, Busy is TRUE. After reading the data, it will write the obtained value to Data, at which point Done will be set to ON.

③ If the corresponding slave cannot be found after Enable is triggered, a 0x27c3 error will be reported.

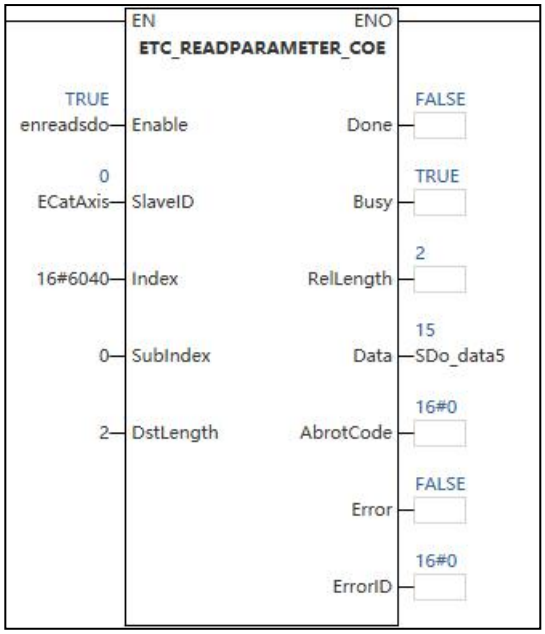
④ If the corresponding index, sub-index, or length is incorrect after Enable is triggered, an error will occur. The specific error needs to be checked according to AbrotCode.

✎**Note：** Reading data takes time, and reading too much data at once will take a long time to update the data.

Error code

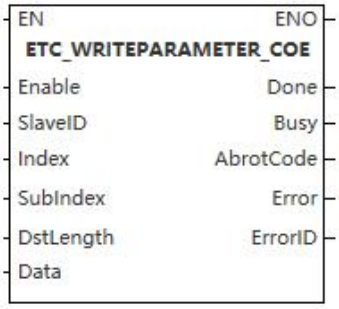
Error code	Content
0x27c3	Axis not found (Check if the input parameters are correct)
0x27D8	SDO access length error (Check the length used in SDO read and write instructions)
0x27CD	SDO read error. Check the address, retry the write operation, or increase the timeout period.
0x27CC	SDO write error. Check the address, retry the write operation, or increase the timeout period.

Example



Read the value from Index 0x6064, Subindex 0, with a length of 2, and display the read value in the Data field.

7.7.2 ETC_WRITEPARAMETER_COE



This function is mainly used to write SDO parameters to the servo drive.

Content, range, data types

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
ETC_WRITEPARAMETER_COE	Input	Enable	BOOL	No	TRUE/FALSE	Read Servo SDO enable
		SlaveID	WORD	No	0 to range of EtherCAT slaves that can be linked	Set according to the ID configured in the interface. If the device is optional, an error will be reported even if the device does not actually exist
		Index	WORD	No	0~65535	Index, fill in according to the information provided by the servo
		SubIndex	WORD	No	0~65535	Sub-index, fill in according to the information provided by the servo
		DstLength	WORD	No	-	Data length, fill in according to the information provided by the servo.
	Output	Done	BOOL	Yes	TRUE/FALSE	When the value is obtained, Done will be set to TRUE, and if Done is not read, it will be set to FALSE
		Busy	BOOL	Yes	TRUE/FALSE	Busy is TRUE during reading
		AbrotCode	DWORD	Yes	TRUE/FALSE	Interrupt code returned by the servo, refer to the EtherCAT standard document for information
		Error	BOOL	Yes	TRUE/FALSE	Error
		ErrorID	DWORD	Yes	See error code for details	Error code

Device used

Instruction	Parameter	Device																					Offset modification	Double word extension	Pulse-type expansion
		X	Y	M	S	SM	T(Bit)	HSC(Bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	DXX	XXP
ETC_WRITE PARAMETER_COE	Enable	●	●	●	●	●																			
	SlaveID																								
	Index																								
	SubIndex																								
	DstLength																								
	Done		●	●	●	●																			
	Busy		●	●	●	●																			
	AbrotCode																								
	Error		●	●	●	●																			
	ErrorID																								

- Used to read the address values of the servo index and subindex, with the correct length.
- Function:

① When Enable is continuously set to ON, it will continuously read the SDO value into Data. Upon receiving a value, it will set Done to ON and append the new value to Data. If no value is read, Done will be set to OFF.

② When Enable is triggered by an edge, it will read data once. During this process, Busy is TRUE. After reading the data, it will write the obtained value to Data, at which point Done will be set to ON.

③ If the corresponding slave cannot be found after Enable is triggered, a 0x27c3 error will be reported.

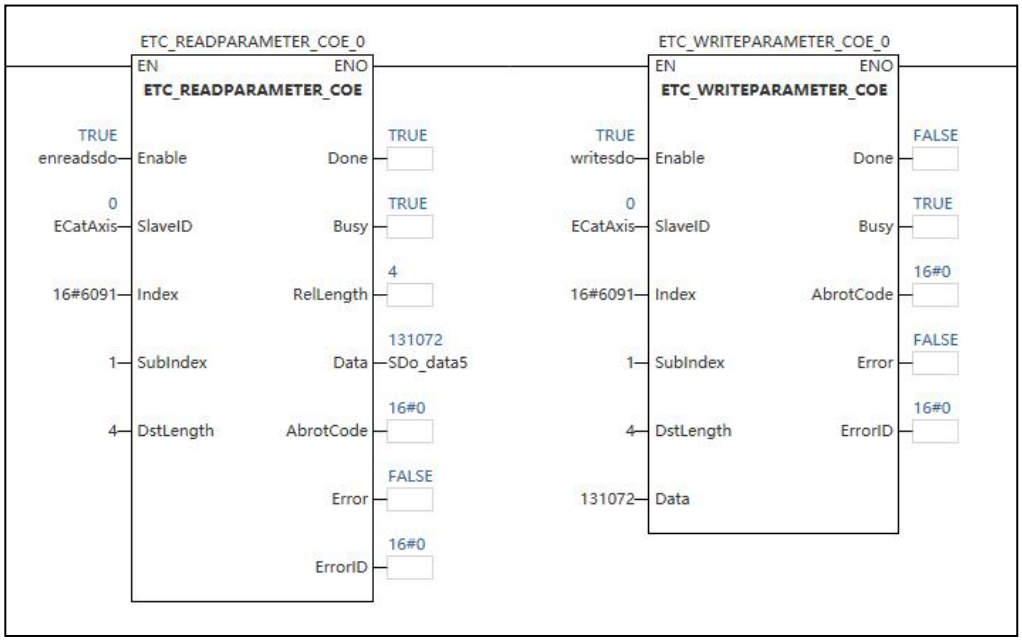
④ If the corresponding index, sub-index, or length is incorrect after Enable is triggered, an error will occur. The specific error needs to be checked according to AbrotCode.

✎**Note**：Reading data takes time, and reading too much data at once will take a long time to update the data.

Error code

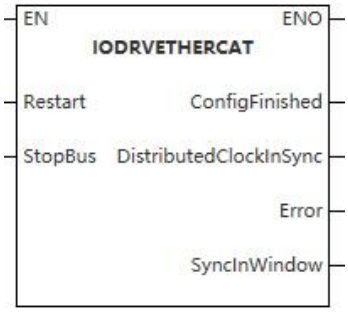
Error code	Content
0x27c3	Corresponding slave station not found (Check if the input parameters are correct)
0x27D8	SDO access length error (Check the length used in SDO read and write instructions)
0x27CD	SDO read error. Check the address, retry the write operation, or increase the timeout period.
0x27CC	SDO write error. Check the address, retry the write operation, or increase the timeout period.

Example



Write the value 131072 to the address of index 0x6091, subindex 1 and data length 4, read it back via the ETC_READPARAMETER_COE.

7.7.3 IODRVETHERCAT



This function is mainly used to start, restart, and shut down the EtherCAT bus.

Content, range, data type

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
IODRV ETHERCAT	Input	Restart	BOOL	No	TRUE/FALSE	Trigger restart/start of the EtherCAT bus
		StopBus	BOOL	No	TRUE/FALSE	Trigger shutdown of the EtherCAT bus
	Output	ConfigFinished	BOOL	Yes	TRUE/FALSE	Configuration completed (TRUE)
		DistributedClockInSync	BOOL	Yes	TRUE/FALSE	Sync clock initialization completed (TRUE)
		Error	BOOL	Yes	TRUE/FALSE	Error occurred (TRUE)
		SyncInWindow	BOOL	Yes	See error code for details	Sync window initialization completed (TRUE)

Device used

Instruction	Parameter	Device																					Offset modification	Double word extension	Pulse-type expansion
		X	Y	M	S	SM	T(Bit)	HSC(Bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	DXX	XXP
IODRV ETHERCAT	Restart	●	●	●	●	●																				
	StopBus	●	●	●	●	●																				
	ConfigFinished		●	●	●	●																				
	Distributed ClockInSync		●	●	●	●																				
	Error		●	●	●	●																				
	SyncInWindow		●	●	●	●																				

- This function is mainly used to start, restart, and shut down the EtherCAT bus.
- Function：

① Setting Restart to TRUE or triggering it on a rising edge will execute the EtherCAT bus restart procedure. ConfigFinished, DistributedClockInSync, and SyncInWindow will be set to TRUE according to the initialization order. An initialization exception (Error) will be set to TRUE.

② Setting StopBus to TRUE will shut down the EtherCAT bus. ConfigFinished, DistributedClockInSync, and SyncInWindow will be set to FALSE.

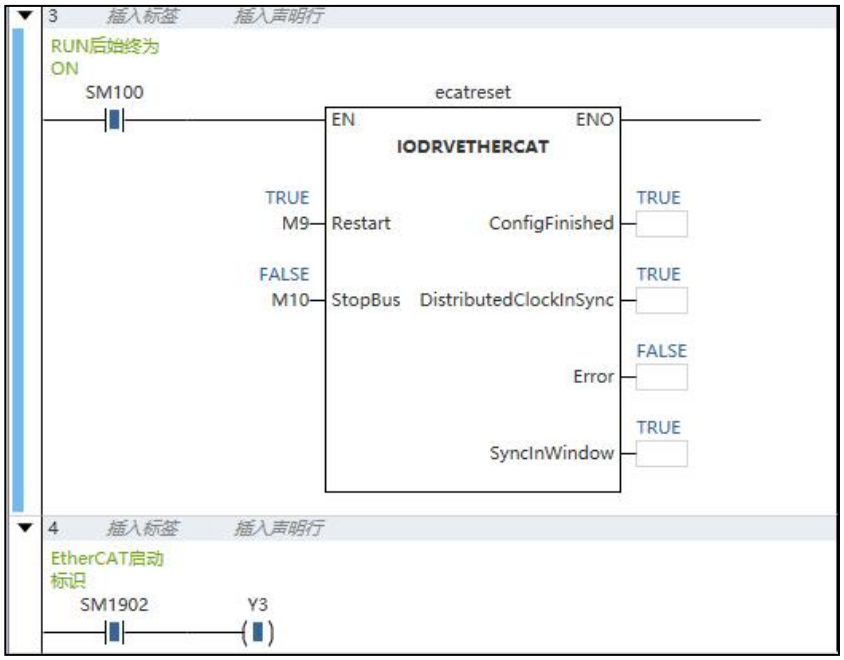
③ Setting both Restart and StopBus to TRUE simultaneously will be considered an exception, and Error will be set to TRUE. The error code will be output from the PLC error address.

✎**Note:** Reading data takes time, and reading too much data at once will take a long time to update the data.

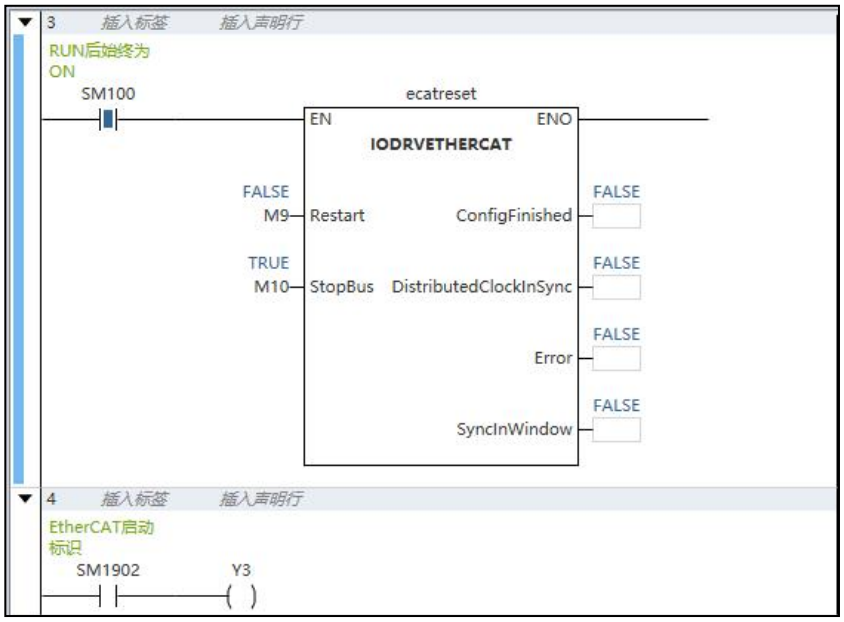
Error code

Error code	Content
0x27c3	Corresponding slave station not found (Check if the input parameters are correct)
0x27D8	SDO access length error (Check the length used in SDO read and write instructions)
0x27CD	SDO read error. Check the address, retry the write operation, or increase the timeout period.
0x27CC	SDO write error. Check the address, retry the write operation, or increase the timeout period.

Example



Setting Restart to TRUE starts the EtherCAT master, and SM1902 is set to TRUE after successful startup.



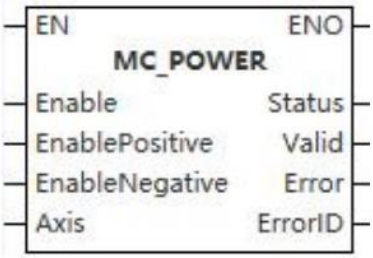
Setting StopBus to TRUE shuts down the EtherCAT master, and SM1902 is set to FALSE synchronously.

7.8 AbrotCode exception codes

AbrotCode exception codes	Corresponding exception
0x00000000	No SDO error
0x05030000	Toggle bit not changed
0x05040000	SDO timeout
0x05040001	Instruction specifier unknown
0x05040005	Insufficient memory event explanation
0x06010000	Access not supported
0x06010001	Write-only entry
0x06010002	Read-only entry
0x06010003	Entry cannot be written because Subindex 0 is not 0
0x06010004	Object cannot be accessed
0x06020000	Object does not exist
0x06040041	Object cannot be mapped to PDO
0x06040042	Mapped object exceeds PDO
0x06040043	Parameter incompatible
0x06040047	Device incompatible
0x06060000	Hardware error
0x06070010	Parameter length error
0x06070012	Parameter too long
0x06070013	Parameter too short
0x06090011	Subindex (entry) does not exist
0x06090030	Value exceeded
0x06090031	Value too large
0x06090032	Threshold too small
0x06090033	Detected module identification list (0xF030) does not match configured module identification list
0x06090036	Less than minimum value
0x08000000	General error
0x08000020	Data not readable/writable event explanation
0x08000021	Data inaccessible due to local control
0x08000022	Event explanation: unable to read or write data in current state
0x08000023	Object not in object dictionary

8 Motion control

8.1 Management function blocks for single-axis motion control



This function block is a power-on software switch for the motion control system.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_POWER	Enable	●	●	●	●	●																				
	EnablePositive	●	●	●	●	●																				
	EnableNegative	●	●	●	●	●																				
	Axis																									
	Status		●	●	●	●																				
	Valid		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R, SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Variable type	Variable type	Can be empty or not	Range	Description
MC_POWER	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		EnablePositive	BOOL	No	TRUE/FALSE	Allow movement in the positive direction
		EnableNegative	BOOL	No	TRUE/FALSE	Allow movement in negative direction
		Axis	WORD	No	[0,65535]	Axis number
	Output	Status	BOOL	Yes	TRUE/FALSE	Enable status
		Valid	BOOL	Yes	TRUE/FALSE	Data valid or not
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (See [8.11 Error code comparison table])

Function

- This function block must be enabled if an axis is to participate in motion.

- The function is as follows:

① When the Enable input is TRUE, the power to the drive is enabled, not to the function block itself;

② When the Enable input is TRUE, this function block is called. If the state of the axis is in the Disable

③ state, then the state of the axis is switched to the state of StandStill, waiting for the motion control of the axis;

④ If a fault occurs when the Enable input is TRUE, a timer function block and an AND logic function can make the state of its axis remain at FAISE. It indicates that there is a fault at the hardware level;

⑤ If the PLC power supply fails during the motion, the state of the axis will switch to the ErrorStop state;

⑥ The input signals EnablePositive and EnableNegative are level controlled. When they are TRUE, axis motion in the positive and negative directions are allowed. Two input signals can be set at the same time, indicating that the allowed motion direction can be either positive or negative;

⑦ Each axis can only have one MC_POWER function block for powering up the axis. If multiple MC_POWER control the same axis, the last function block is valid;

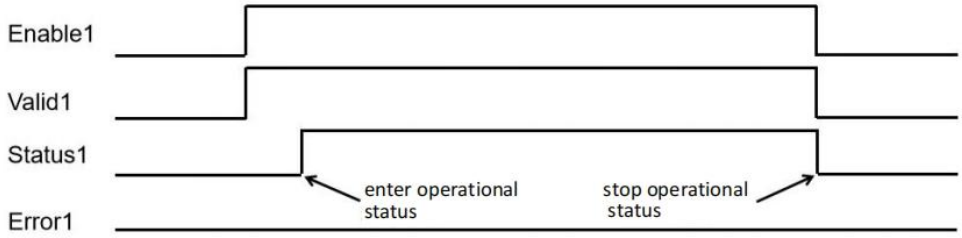
⑧ The Status output signal indicates the status of the driver power supply, and 1 means power on; 0 means not power on. It is also not a simulated value if the axis drive is a simulated axis. Busy is the operating status of the function block, 1 means busy; 0 means idle. 0 means uncontrolled. Valid indicates whether the output signal of the function block is valid, and 1 means the output signal is valid; 0 means the output signal is invalid. Error indicates whether the function block is faulty, and 1 means faulty; 0 means no fault. ErrorID is the error code, and when a function block fault occurs, there are 16 digits indicating the fault information.

Precautions

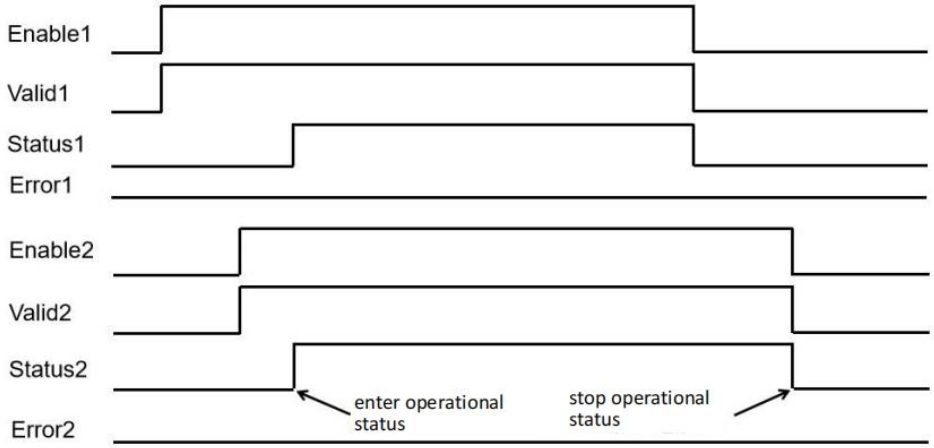
- When any error occurs when the axis is enabled, it will be fed back through Error and ErrorID;
- The PDO parameters of servo axis bound to the axis must have control word, status word, target position and current position.

Sequence diagram

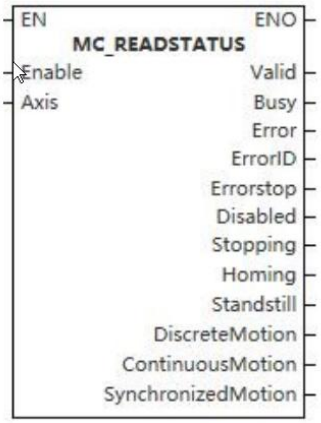
- When using a MC_POWER instruction and axis enables normally, sequence diagram is as follows:



- When axis uses two MC_POWER instructions and axis enables normally, sequence diagram is as follows:



8.1.2 MC_READSTATUS



This function block is used to read the status of the controlled axis.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_READSTATUS	Enable	●	●	●	●	●																				
	Axis																									
	Valid		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									
	Errorstop		●	●	●	●																				
	Disabled		●	●	●	●																				
	Stopping		●	●	●	●																				
	Homing		●	●	●	●																				
	Standstill		●	●	●	●																				
	DiscreteMotion		●	●	●	●																				
	ContinuousMotion		●	●	●	●																				
	SynchronizedMotion		●	●	●	●																				

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_READ STATUS	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		Axis	WORD	No	[0,65535]	Axis number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Output valid or not
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (see [8.11 Error code comparison table])
		Errorstop	BOOL	Yes	TRUE/FALSE	Errorstop flag
		Disabled	BOOL	Yes	TRUE/FALSE	Disable flag
		Stopping	BOOL	Yes	TRUE/FALSE	Stopping flag
		Homing	BOOL	Yes	TRUE/FALSE	Homing sign
		Standstill	BOOL	Yes	TRUE/FALSE	Static sign
		DiscreteMotion	BOOL	Yes	TRUE/FALSE	Discontinuous motion flag
		ContinuousMotion	BOOL	Yes	TRUE/FALSE	Continuous motion flag
		SynchronizedMotion	BOOL	Yes	TRUE/FALSE	Synchronized motion flag

Function

▪ It is used to read the status of the controlled axis, corresponding to the single-axis SynchronizedMotion, ContinuousMotion, DiscreteMotion, Stopping, Errorstop, Homing, Standstill and Disabled;

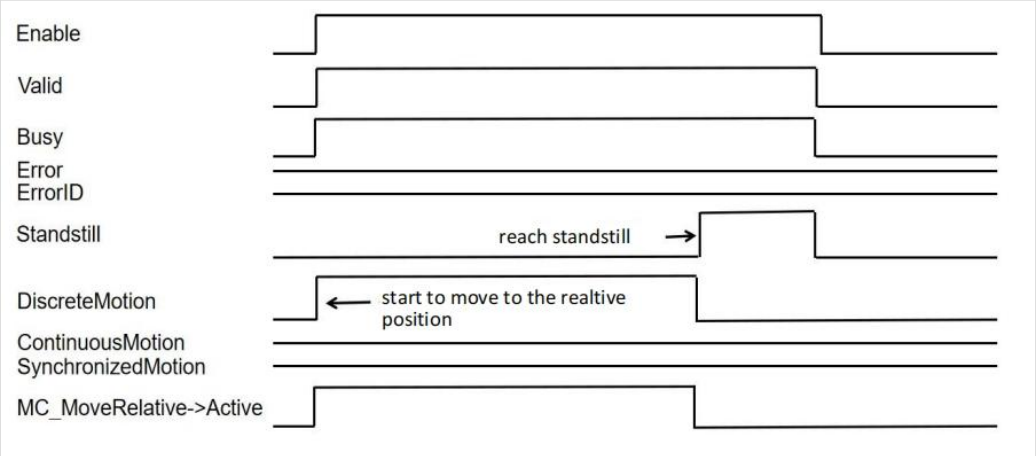
▪ When the MC_ReadStatus function block input Enable is TRUE, the status of the axis is periodically updated.

Precautions

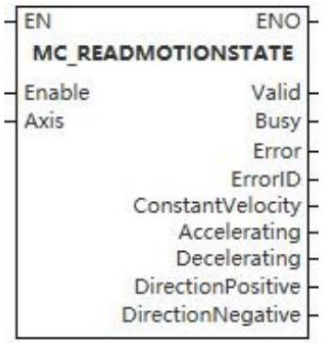
▪ This function block is not available when the axis does not exist.

Sequence diagram

▪ When MC_READSTATUS is used with the MC_MOVERELELATIVE instruction and the axis can be enabled normally, the sequence diagram is as follows:



8.1.3 MC_READMOTIONSTATE



This function block is used to read the motion status of the controlled axis, including constant speed, acceleration, deceleration, positive direction and negative direction.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																									Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E		[D]	XXP
MC_READMOTIONSTATE	Enable	●	●	●		●																						
	Axis		●	●		●																						
	Valid		●	●		●																						
	Busy		●	●		●																						
	Error																											
	ErrorID																											
	ConstantVelocity		●	●		●																						
	Accelerating		●	●		●																						
	Decelerating		●	●		●																						
	DirectionPositive		●	●		●																						
	DirectionNegative		●	●		●																						

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_READMOTIONSTATE	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		Axis	WORD	No	0 to 65535	Axis number
	Output	Valid	BOOL	No	TRUE/FALSE	Output valid or not
		Busy	BOOL	No	TRUE/FALSE	Busy sign
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code
		ConstantVelocity	BOOL	No	TRUE/FALSE	Constant velocity
		Accelerating	BOOL	No	TRUE/FALSE	Accelerating
		Decelerating	BOOL	No	TRUE/FALSE	Decelerating
		DirectionPositive	BOOL	No	TRUE/FALSE	Positive direction
		DirectionNegative	BOOL	No	TRUE/FALSE	Negative direction

Function

▪ Used to read the motion status of the controlled axis, including constant speed, acceleration, deceleration, positive direction and negative direction;

▪ When the MC_ReadStatus function block input Enable is TRUE, the status of the axis is periodically updated.

Precautions

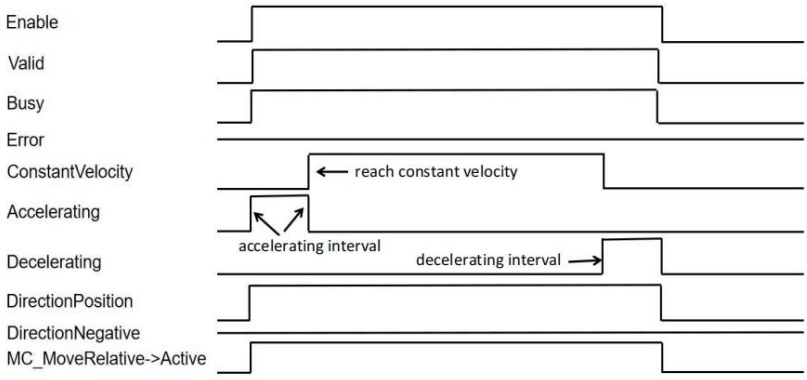
▪ This function block is not available when the axis does not exist.

Error code

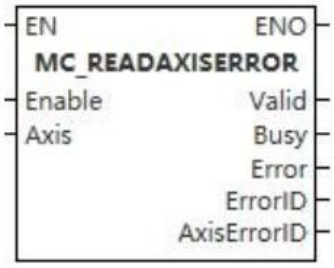
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

▪ When using MC_READMOTIONSTATE with MC_MOVERELATIVE instruction together and axis enables normally, sequence diagram is as follows:



8.1.4 MC_READAXISERROR



The function block provides fault information about the axis.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_READAXISERROR	Enable	●	●	●	●	●																					
	Axis																										
	Valid		●	●	●	●																					
	Busy		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										
	AxisErrorID																										

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_READAXISERROR	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		Axis	WORD	No	[0,65535]	Axis number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Output valid or not
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Errors occurred in this function block (see [8.11 Error code])
		AxisErrorID	DWORD	Yes	-	Read the error code of the axis

Function

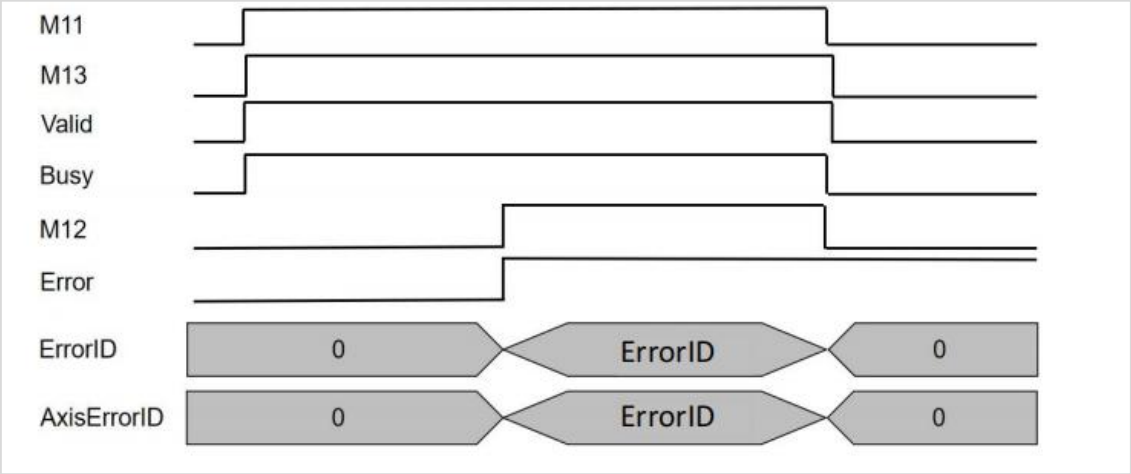
- The function block provides fault information for the axis. Typically, you can connect the axis status ErrorStop output of the MC_READSTATUS function block to the Enable input end of MC_READAXISERROR, so that the fault code of the axis can be obtained from AxisErrorID. You can also get the fault code from ErrorID.

Precautions

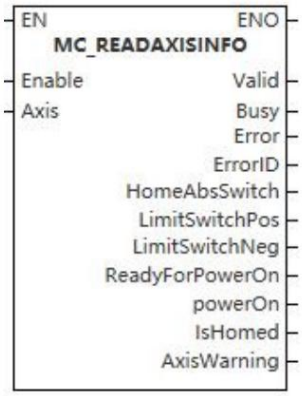
- This function block is not available when the axis does not exist.

Sequence diagram

- When using MC_READAXISINFO with MC_POWER and MC_MOVERELATIVE instructions together and axis enables normally, in which MC_POWER→EnablePosition is FALSE and MC_POWER→EnableNegative is TRUE, sequence diagram is as follows:


M11
M13
Valid
Busy
M12
Error
ErrorID
0
ErrorID
0
AxisErrorID
0
ErrorID
0

8.1.5 MC_READAXISINFO



The function block is used to provide information about the axis.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_READAXISINFO	Enable	●	●	●	●	●																					
	Axis																										
	Valid		●	●	●	●																					
	Busy		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										
	HomeAbsSwitch		●	●	●	●																					
	LimitSwitchPos		●	●	●	●																					
	LimitSwitchNeg		●	●	●	●																					
	ReadyForPowerOn		●	●	●	●																					
	powerOn		●	●	●	●																					
	IsHomed		●	●	●	●																					
	AxisWarning		●	●	●	●																					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_READAXISINFO	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		Axis	WORD	No	[0,65535]	Axis number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Output valid or not
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (See [8.11 Error code comparison table])
		HomeAbsSwitch	BOOL	Yes	TRUE/FALSE	Homing absolute position switch
		LimitSwitchPos	BOOL	Yes	TRUE/FALSE	Positive limit switch
		LimitSwitchNeg	BOOL	Yes	TRUE/FALSE	Negative limit switch
		ReadyForPowerOn	BOOL	Yes	TRUE/FALSE	Ready to enable
		Enable	BOOL	No	TRUE/FALSE	Enable
		IsHomed	BOOl	No	TRUE/FALSE	Homing successful
		AxisWarning	BOOl	No	TRUE/FALSE	Axis warning

Function

▪ It is used to provide information about the axis, including the status information of the absolute position switch during homing, the positive and negative limit switches, the power-on switch of the function block, and whether the axis has a warning or not.

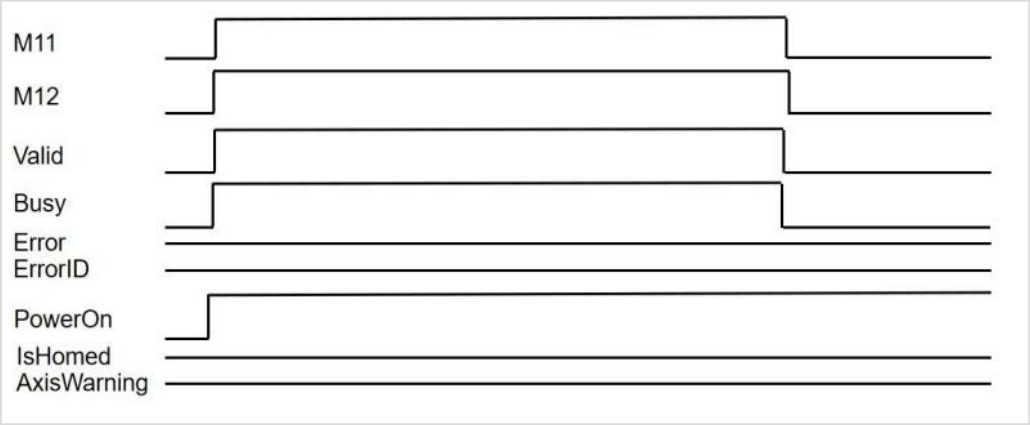
Precautions

▪ This function block is not available when the axis does not exist.

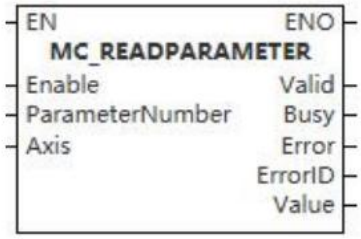
▪ HomeAbsSwitch and ReadyForPowerOn are reserved interfaces, which are both False by default.

Sequence diagram

▪ When MC_READAXISINFO is used with the MC_POWER instruction, when the axis can be enabled normally, the sequence diagram is as follows:


M11
M12
Valid
Busy
Error
ErrorID
PowerOn
IsHomed
AxisWarning

8.1.6 MC_READPARAMETER



This function block is used to read axis parameters.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification		Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	HE	[D]	XXP
MC_READPARAMETER	Enable	●	●	●	●	●																				
	ParameterNumber															●	●	●	●	●						
	Axis																									
	Valid		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									
	Value																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_READPARAMETER	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		ParameterNumber	INT	No	[1,17]	Parameter number
		Axis	WORD	No	[0,65535]	Axis number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Output valid or not
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (See [8.11 Error code table])
		Value	LREAL	Yes	-	Axis parameter value

Function

- Used to read axis parameters, as shown in following table.

Parameter number PN	Parameter name	Data type	B/E	R/W	Notes
1	InstructionedPosition	REAL	B	R	Instruction position
2	SWLimitPos	REAL	E	R/W	Position of positive software limit switch
3	SWLimitNeg	REAL	E	R/W	Positive of negative software limit switch
4	EnableLimitPos	BOOL	E	R/W	Enable positive software limit switch
5	EnableLimitNeg	BOOL	E	R/W	Enable negative software limit switch
6	EnablePosLagMonitoring	BOOL	E	R/W	Enable position lag monitoring
7	MaxPositionLag	REAL	E	R/W	Maximum position lag
8	MaxVelocitySystem	REAL	E	R	Maximum allowable axial velocity in motion system
9	MaxVelocityAppl	REAL	B	R/W	Maximum allowable axial velocity in application
10	ActualVelocity	REAL	B	R	Actual velocity
11	InstructionedVelocity	REAL	B	R	Instructioned velocity
12	MaxAccelerationSystem	REAL	E	R	Maximum allowable axial acceleration in motion system
13	MaxAccelerationAppl	REAL	E	R/W	Maximum allowable axial acceleration in application
14	MaxDecelerationSystem	REAL	E	R	Maximum allowable axial deceleration in motion system
15	MaxDecelerationAppl	REAL	E	R/W	Maximum allowable axial deceleration in application
16	MaxJerkSystem	REAL	E	R	Maximum allowable axial jerk in motion system
17	MaxJerkAppl	REAL	E	R/W	Maximum allowable axial jerk in application system

Precautions

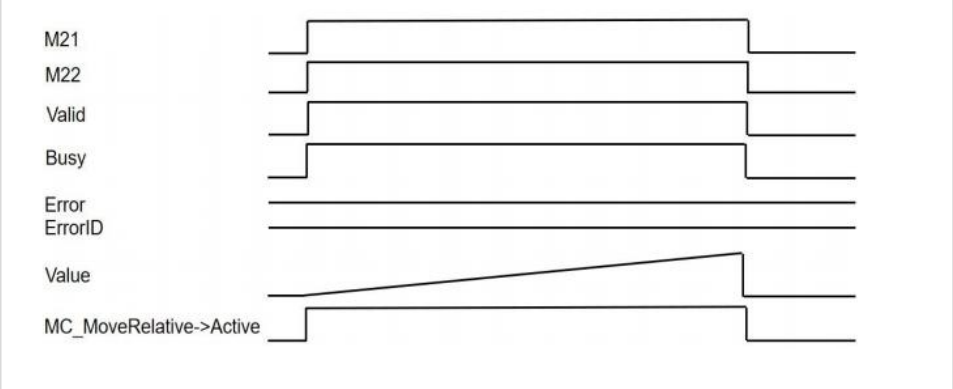
- This function block is not available when the axis does not exist.

Error code

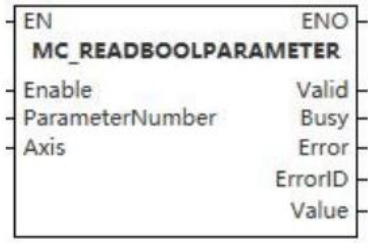
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

- When using MC_READPARAMETER with MC_MOVERELATIVE instruction together, in which ParameterNumber=1, and axis enables normally, sequence diagram is as follows:



8.1.7 MC_READBOOLPARAMETER



This function block is used to read axis Boolean parameters.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T (bit)	C (bit)	LC (bit)	HSC (bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_READBOOLPARAMETER	Enable	●	●	●	●	●																					
	ParameterNumber															●	●	●	●	●							
	Axis																										
	Valid		●	●	●	●																					
	Busy		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										
	Value		●	●	●	●																					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_READBOOLPARAMETER	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		ParameterNumber	INT	No	[1,17]	Parameter number
		Axis	WORD	No	[0,65535]	Axis number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Output valid or not
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes		Error code (See [8.11 Error code])
		Value	BOOL	Yes	TRUE/FALSE	Axis parameter

Function

- Used to read axis Boolean parameters, as shown in following table.

Parameter number PN	Parameter name	Data type	B/E	R/W	Notes
1	InstructionedPosition	REAL	B	R	Instruction position
2	SWLimitPos	REAL	E	R/W	Position of positive software limit switch
3	SWLimitNeg	REAL	E	R/W	Positive of negative software limit switch
4	EnableLimitPos	BOOL	E	R/W	Enable positive software limit switch
5	EnableLimitNeg	BOOL	E	R/W	Enable negative software limit switch
6	EnablePosLagMonitoring	BOOL	E	R/W	Enable position lag monitoring
7	MaxPositionLag	REAL	E	R/W	Maximum position lag
8	MaxVelocitySystem	REAL	E	R	Maximum allowable axial velocity in motion system
9	MaxVelocityAppl	REAL	B	R/W	Maximum allowable axial velocity in application
10	ActualVelocity	REAL	B	R	Actual velocity
11	InstructionedVelocity	REAL	B	R	Instructioned velocity
12	MaxAccelerationSystem	REAL	E	R	Maximum allowable axial acceleration in motion system
13	MaxAccelerationAppl	REAL	E	R/W	Maximum allowable axial acceleration in application
14	MaxDecelerationSystem	REAL	E	R	Maximum allowable axial deceleration in motion system
15	MaxDecelerationAppl	REAL	E	R/W	Maximum allowable axial deceleration in application
16	MaxJerkSystem	REAL	E	R	Maximum allowable axial jerk in motion system
17	MaxJerkAppl	REAL	E	R/W	Maximum allowable axial jerk in application system

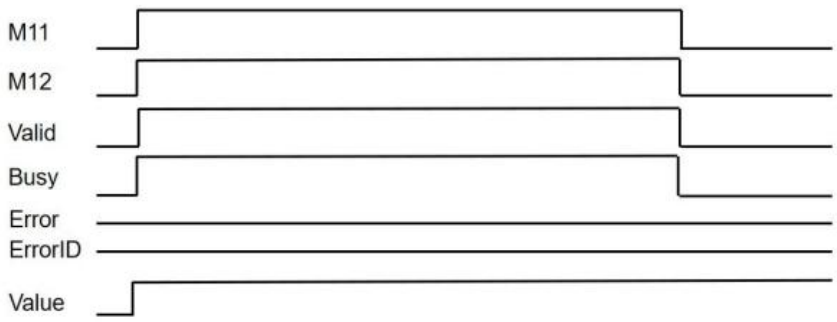
Precautions

- This function block is not available when the axis does not exist.

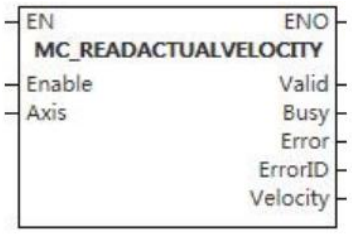
Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram



8.1.8 MC_READACTUALVELOCITY



This function block is used to read the actual velocity of the controlled axis.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_READACTUALVELOCITY	Enable	●	●	●	●	●																				
	Axis																									
	Valid		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									
	Velocity																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_READACTUALVELOCITY	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		Axis	WORD	No	[0,65535]	Axis number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Output valid or not
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (See [8.11 Error code])
		Velocity	LREAL	Yes	-	Actual velocity

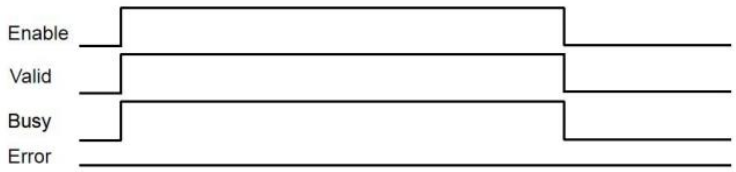
Function

- When the function block's Enable is set, the MC_READACTUALVELOCITY output returns the actual velocity of the controlled single axis.

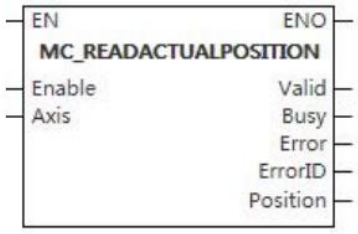
Precautions

- This function block is not available when the axis does not exist;
- When the function block's Enable is reset, the output value loses its validity, and the Valid output is reset.

Sequence diagram


Enable
Valid
Busy
Error

8.1.9 MC_READACTUALPOSITION



This function block is used to read the actual absolute position of the axis.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_READACTUALPOSITION	Enable	●	●	●	●	●																					
	Axis																										
	Valid		●	●	●	●																					
	Busy		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										
	Position																										

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_READACTUALPOSITION	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		Axis	WORD	No	[0,65535]	Axis number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Output valid or not
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (See [8.11 Error code])
		Position	LREAL	Yes	-	Actual absolute position

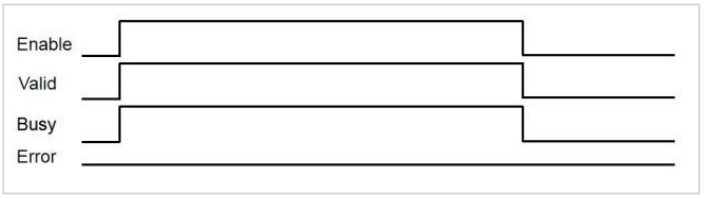
Function

- When the function block's Enable is set, the MC_READACTUALPOSITION output returns the actual absolute position of the controlled single axis.

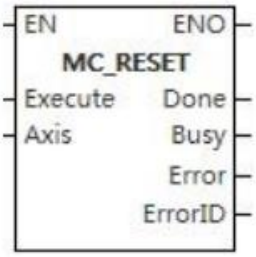
Precautions

- This function block is not available when the axis does not exist;
- 0x6064 should be configured in the bus axis, otherwise it is invalid;
- When the function block's Enable is reset, the output value loses its validity, and the Valid output is reset.

Sequence diagram



8.1.10 MC_RESET



This function block is used for clearing the faults of the controlled axis.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_RESET	Execute	●	●	●	●	●																				
	Axis																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_RESET	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Axis	WORD	No	[0,65535]	Axis number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (See [8.11 Error code])

Function

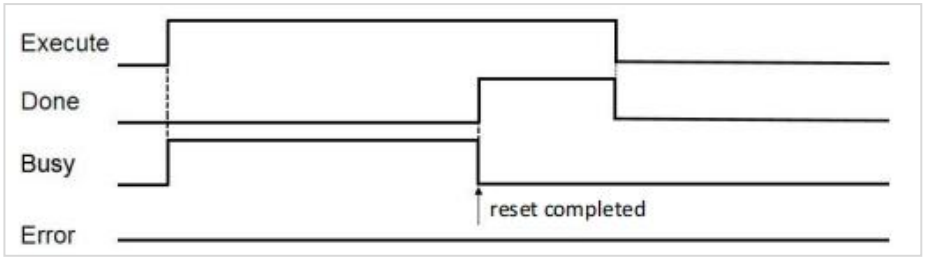
- When Execute input signal rises, it is used to clear the fault of the controlled axis. These faults include axis faults and drive side faults. If the drive fails, the reset of the drive failure should be carried out first. These fault treatments do not cause motion of the controlled axis;
- On the rising edge when Execute signal is valid, the instruction tries to reset the fault of the axis. If the reset is successful, Done output is valid; Otherwise, Error signal is valid, and ErrorID will give the reason for the reset failure;
- After a successful reset, PLCOpen state machine of the axis is in the StandStill state if the drive is in the enabled state, or the Disabled state if the drive is not enabled.

Precautions

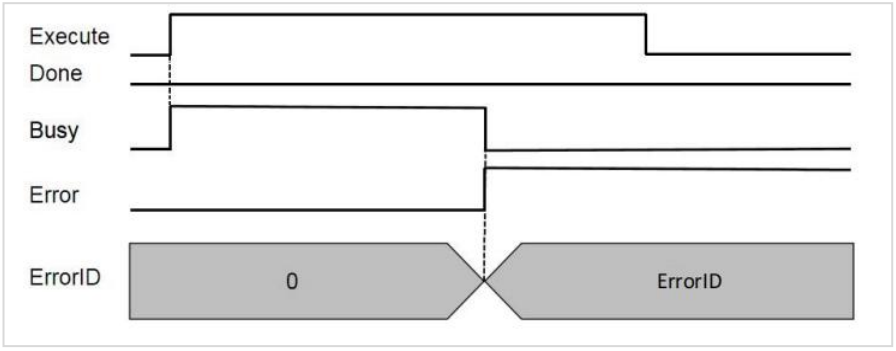
- This function block is not available when the axis does not exist.

Sequence diagram

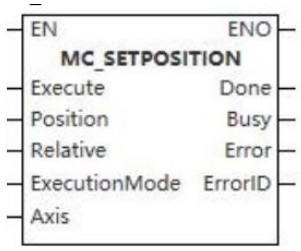
When the axis fails, MC_Reset instruction is called and the axis fault is successfully reset. The sequence diagram is shown as follows:



When a non-resettable fault occurs in the drive, the sequence diagram is as follows:



8.1.11 MC_SETPOSITION



This function block is used to set the position (absolute position or relative position) of the controlled axis.

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_SETPOSITION	Execute	●	●	●	●	●																					
	Position																										
	Relative	●	●	●	●	●																					
	Execution Mode																										
	Axis																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Device used (only a single device can be used, and device splicing and offset are not supported)

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_SETPOSITION	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Position	LREAL	Yes	[-999999999.999999, 999999999.999999]	Position
		Relative	BOOL	Yes	TRUE/FALSE	Set relative position or absolute position
		Execution Mode	WORD	Yes	[0,2]	Operation mode 0 indicates that the operation of writing the function block data is executed immediately; 1 indicates that the operation of writing function block data is delayed (Reserved); 2 indicates that the operation of writing function block data is executed in buffer mode (Reserved);
		Axis	WORD	No	[0,65535]	Axis number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes		Error code (See [8.11 Error code])

Function

- MC_SETPOSITION function block is used to set the position (absolute position or relative position) of the controlled axis. It controls the set position and actual position of the axis by moving the coordinate system so that they have the same value without causing movement, and recalibrating to obtain the same following error.
- At present, this instruction does not support triggering when the axis is in motion.

Precautions

- This function block is not available when the axis does not exist.

8.1.12 MC_TouchProbe

- MC_TouchProbe-Save position data for the current axis



- This function is used to save the position data of current axis through signal triggering.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																										Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E		[D]	XXP
MC_Touch Probe	Execute	●	●	●	●	●																							
	WIndowsOnly	●	●	●	●	●																							
	FirstPosition																												
	LastPosition																												
	TriggerInput																												
	Axis																												
	Done		●	●	●	●																							
	Busy		●	●	●	●																							
	CommandAborted		●	●	●	●																							
	Error		●	●	●	●																							
	ErrorID																												
	RecoredPosition																												

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

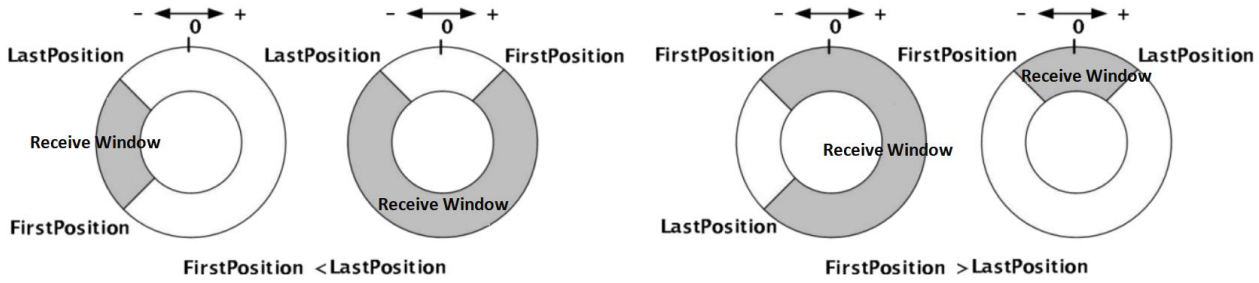
Variable Type Used

Instruction	Pin type	Parameter	Type	Empty or not	Range	Description
MC_Touch Probe	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		WindowsOnly	BOOL	Yes	TRUE/FALSE	Window mode
		FirstPosition	LREAL	Yes	[-999999999.999999, 999999999.999999]	Starting position
		LastPosition	LREAL	Yes	[-999999999.999999, 999999999.999999]	End position
		TriggerInput	MC_TRIGGER_REF	No		Associated attributes such as probe trigger signals or trigger attributes
		Axis	WORD	No	[0,65535]	Axis number
	Output	Done	BOOL	Yes	TRUE/FALSE	Touch probe trigger flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Error or not
		ErrorID	DWORD	Yes		Error code
		RecorededPosition	LREAL	Yes	[-999999999.999999, 999999999.999999]	Touch probe trigger recording position

Function

Window Mode Description

▪ FirstPosition position starts receiving, go clockwise in the window to LastPosition location, this time period is called the receiving window. The following diagram shows the relationship between the receive window and FirstPosition and LastPosition.



▪ The three pins of FirstPosition, LastPosition, and TriggerInput are vacant, that is, the window is not used (WIndowsOnly is false);

▪ MC_AbortTrigger function block is used to abort the MC_TouchProbe. A detection function of a function block;

▪ MC_TouchProbe function block is a touch probe function block that records the precise position of the controlled axis when the event is triggered. This is the edge trigger function block, So use the Execute input signal to trigger the function block to operate. And use the window to receive touch events. MC_TRIGGER_REF input/output data is a structure data type. Here are the relevant instructions.

▪ The following table describes the detailed properties of TriggerInput variable type MC_TRIGGER_REF.

Type Name	Variable	Type	Empty or not	Range	Description
Input	nTriggerNumber	WORD	No	[0,3]	Touch probe function selection. 0: Touch probe 1 rising edge latching; 1: Touch probe 1 falling edge latching; 2: Touch probe 2 rising edge latching; 3: Touch probe 2 falling edge latching.
	Single	BOOL	Yes	TRUE/FALSE	Whether it is triggered once (only triggered when the trigger signal is valid for the first time). True: Single trigger False: Continuous trigger
	FastLatChing	BOOL	Yes	TRUE/FALSE	Specify the trigger mode (currently only drive mode is supported). TRUE: Driver mode; FALSE Controller mode.
	Input	BOOL	Yes	TRUE/FALSE	Controller mode signal input. When FastLatChing is FLASE, detect this signal.
	Active	BOOL	Yes	TRUE/FALSE	Reserved

▪ Through MC_TouchProbe function block, the current position of the running axis is recorded when the relevant signal of the input parameter TriggerInput is triggered;

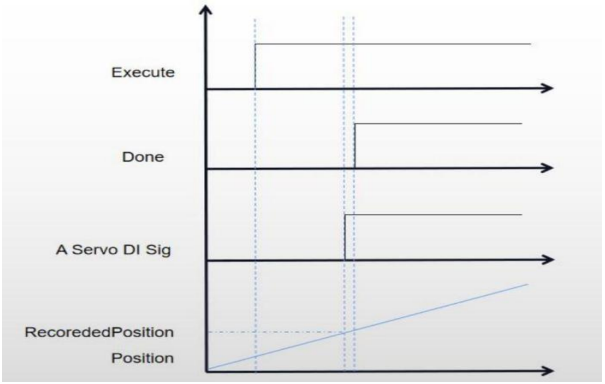
▪ Execute rising edge executes. The falling edge reset, and it can be twice triggered after resetting;

▪ When the drive is latched: The drive will record the position after receiving the latch signal and send the position to controller.

Sequence diagram

▪ Execute of the function block must be a rising edge trigger condition;

▪ Done of the function block indicates that the probe signal has been successfully launched and position information has been captured.

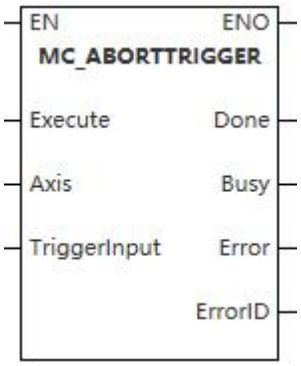


Precautions

. Configured as a single trigger, an error will be reported when triggered outside the window period, but continuous triggering will not, and the probe data outside the window period will be filtered out.

. Using the touch probe function, it needs to configure the corresponding PDO.

8.1.13 MC_AbortTrigger



The function block terminates the associated characteristics of the MC_TouchProbe related event and is used in conjunction with MC_TouchProbe.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_Abort Trigger	Execute	●	●	●	●	●																				
	TriggerInput																									
	Axis																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

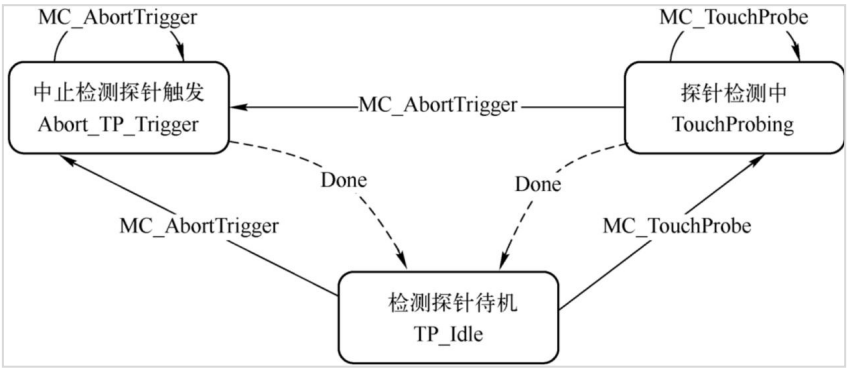
Instruction	Pin Type	Parameter	Type	Empty or not	Range	Description
MC_AbortTrigger	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		TriggerInput	MC_TRIGGER_REF	No	-	Associated attributes such as probe trigger signal or trigger attribute
		Axis	WORD	No	[0,65535]	Axis number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Where there is error or not
		ErrorID	DWORD	Yes	-	Error

Function

- The MC_AbortTrigger function block is a abort probe function block that is used to abort function blocks connected to the probe, such as the MC_TouchProbe function block
- Refer to the MC_TouchProbe function block for TriggerInput related instructions;
- The MC_AbortTrigger function block is used to terminate the associated operation of the trigger signal or property and the relevant trigger instruction;

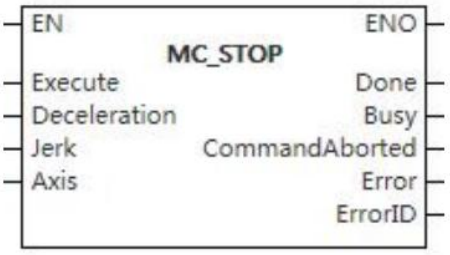
- The Execute of the function block must be a rising edge trigger condition;
- The Done of the function block indicates that the setting operation is successful;
- The Busy table of a function block indicates that the function block is currently in operation.

Trigger state machine



8.2 Motion Function Blocks for Single-axis Motion Control

8.2.1 MC_STOP



This function block is used to change the state of the controlled axis to the deceleration stopping state.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_STOP	Execute	●	●	●	●	●																				
	Deceleration																									
	Jerk																									
	Axis																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_STOP	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Deceleration	LREAL	No	0 to 9999999.00	Deceleration
		Jerk	LREAL	No	0 to 9999999.00	Jerk
		Axis	WORD	No	0 to 65535	Axis number
	Output	Done	BOOL	No	TRUE/FALSE	Completion flag
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

▪ This function block is used to change the state of the controlled axis to the deceleration stopping state. When the rising edge of the input Execute signal triggers the function block, the controlled axis decelerates with the output Decelerate value and the axis motion stops.

Precautions

▪ This function block is not available when the axis does not exist;

▪ In profile mode, triggering STOP switches to profile velocity mode;

▪ In profile mode, the STOP command requires the mapper to map the PDOs of 0x6060, 0x6061, 0x60FF, and 0x606C for normal operation;

▪ In CST/CSV mode, triggering STOP switches to synchronous velocity mode;

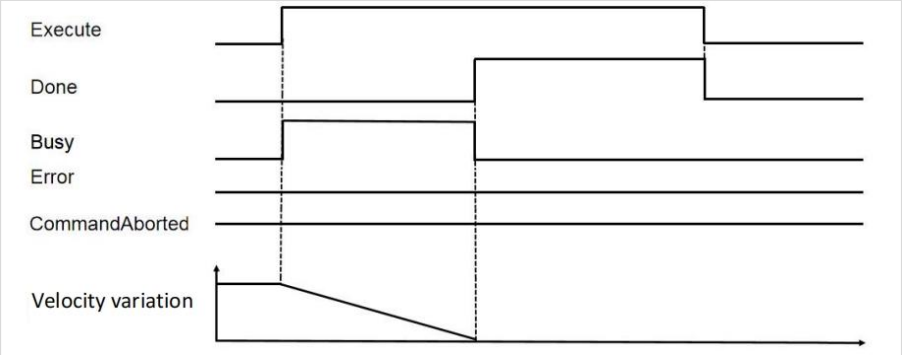
▪ In CST/CSV mode, the STOP command requires the mapper to map the PDOs of 0x6060, 0x6061, 0x60FF, and 0x606C for normal operation;

Error code

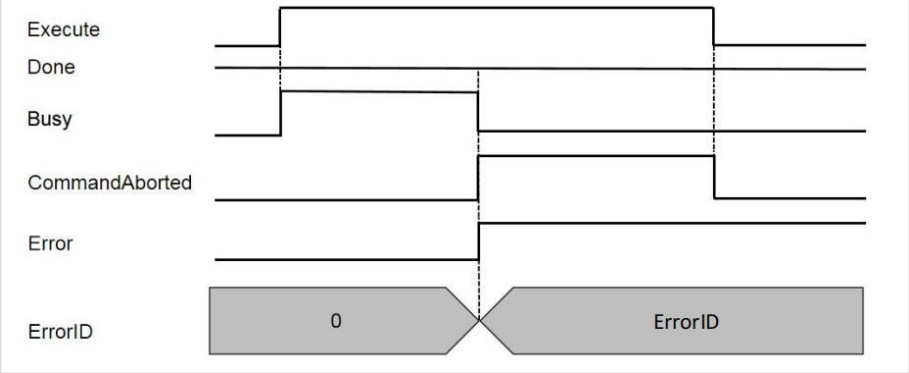
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

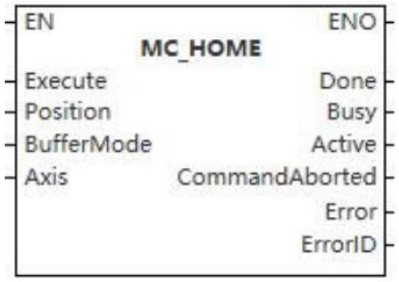
When the call is located, trigger MC_STOP instruction and sequence diagram is as follows:



▪ When driver fails during the instruction running, sequence diagram is as follows:



8.2.2 MC_HOME



This function block is used to change the state of the controlled axis to the origin state.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_HMOE	Execute	●	●	●	●	●																				
	Position																									
	BufferMode																									
	Axis																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_HOME	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Position	LREAL	No	0 to 9999999.00	Position
		BufferMode	WORD	No	0 to 5	Mode selection
		Axis	WORD	No	0 to 65535	Axis number
	Output	Done	BOOL	No	TRUE/FALSE	Completion sign
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag bit
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

- It is determined whether the controlled axis returns to its origin by detecting the relevant limit switch, absolute position origin switch or transmitted sequence pulse signal.

Precautions

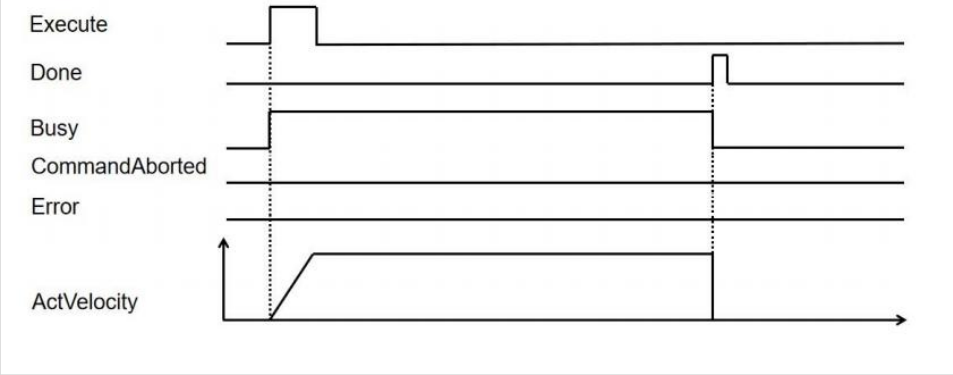
- This function block is not available when the axis does not exist.

Error code

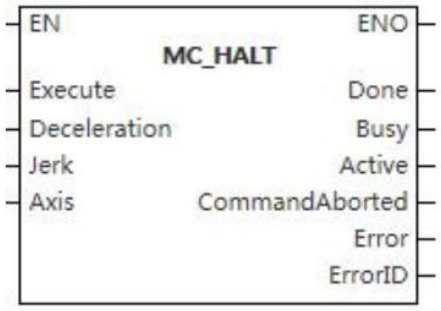
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

Start origin homing, and then driver normally executes origin homing, sequence diagram is as follows:



8.2.3 MC_HALT



This function block is often used to stop the motion of the controlled axis under normal operating conditions.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification		Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_HALT	Execute	●	●	●	●	●																				
	Deceleration																									
	Jerk																									
	Axis																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_HALT	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Deceleration	LREAL	No	0 to 9999999.00	Deceleration
		Jerk	LREAL	No	0 to 9999999.00	Jerk
		Axis	WORD	No	0 to 65535	Axis number
	Output	Done	BOOL	No	TRUE/FALSE	Data valid or not
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	No	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

- If the controlled axis is in a continuous motion state, the rising edge of the Execute signal input by the MC_Halt function block triggers the activation of the function block, so that the state of the controlled axis changes to the intermittent motion state. Similarly, if the controlled axis is in an intermittent motion state or a synchronous motion state, the activation of the MC_HALT function block will keep the controlled axis in an intermittent motion state. This function block is often used to stop the motion of the controlled axis

under normal operating conditions.

Precautions

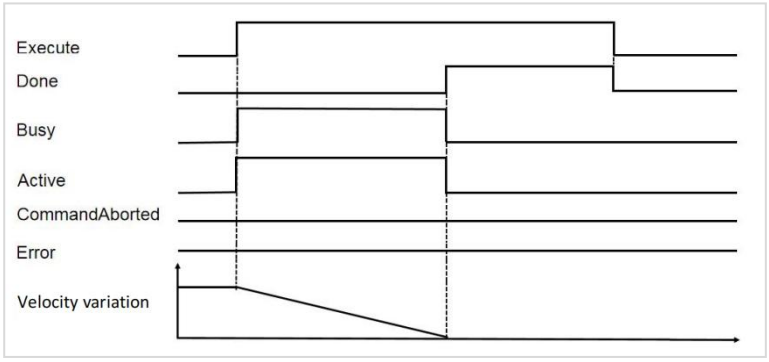
- This function block is not available when the axis does not exist.

Error code

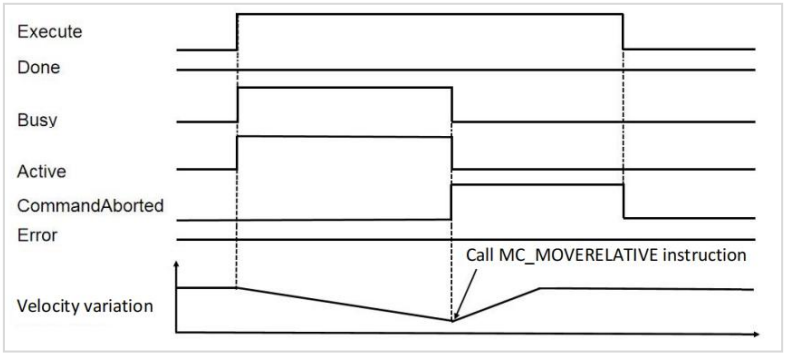
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

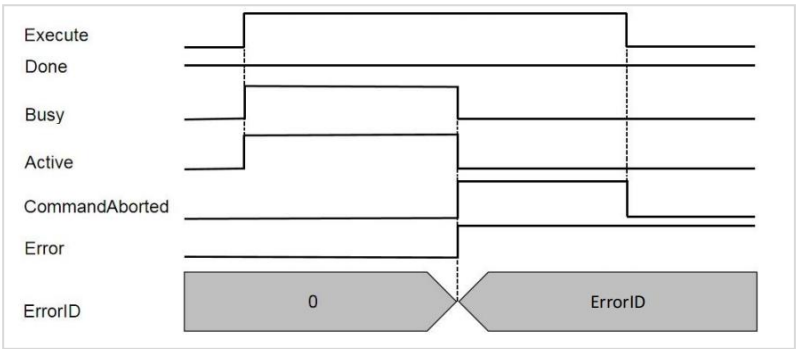
- After the call is located, trigger MC_HALT instruction, sequence diagram is as follows:



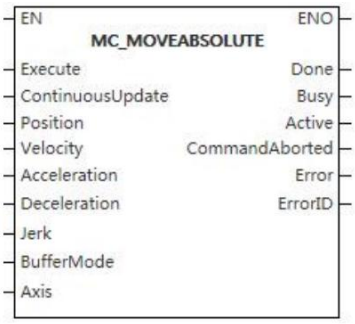
- After triggering MC_HALT instruction, call the location instruction again to interrupt the execution of MC_HALT, sequence diagram is as follows:



- When executing MC_HALT, showdown failure happens, sequence diagram is as follows:



8.2.4 MC_MOVEABSOLUTE



The function block makes the controlled axis reach the set absolute position with the input parameters.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVEABSOLUTE	Execute	●	●	●	●	●																					
	ContinuousUpdate	●	●	●	●	●																					
	Position																										
	Velocity																										
	Acceleration																										
	Deceleration																										
	Jerk																										
	BufferMode																										
	Axis																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_MOVEABSOLUTE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	No	TRUE/FALSE	Updated during operation
		Position	LREAL	No	-9999999.00~9999999.00	Absolute position
		Velocity	LREAL	No	0 to 9999999.00	Velocity
		Acceleration	LREAL	No	0 to 9999999.00	Acceleration
		Deceleration	LREAL	No	0 to 9999999.00	Deceleration
		Jerk	LREAL	No	0 to 9999999.00	Jerk
		BufferMode	WORD	No	0 to 5	Mode selection
		Axis	WORD	No	0 to 65535	Axis number
	Output	Done	BOOL	No	TRUE/FALSE	Completion flag
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	No	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

▪ The function block enables the controlled axis reach the set absolute position with input parameters (velocity, acceleration, deceleration, jerk and specified motion direction).

Precautions

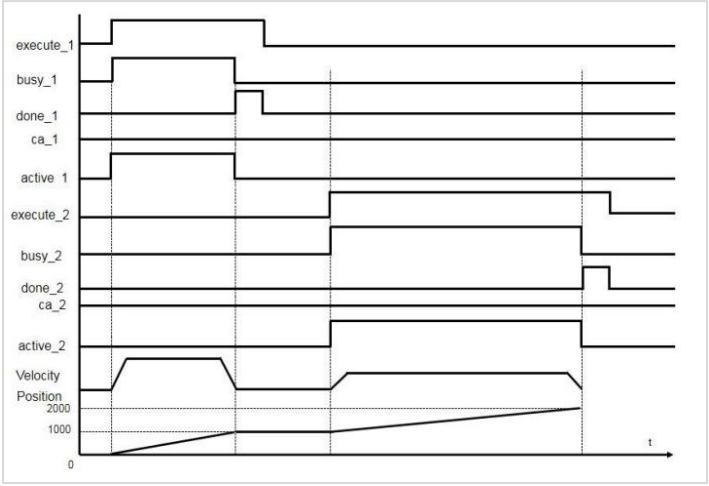
▪ This function block is not available when the axis does not exist.

Error code

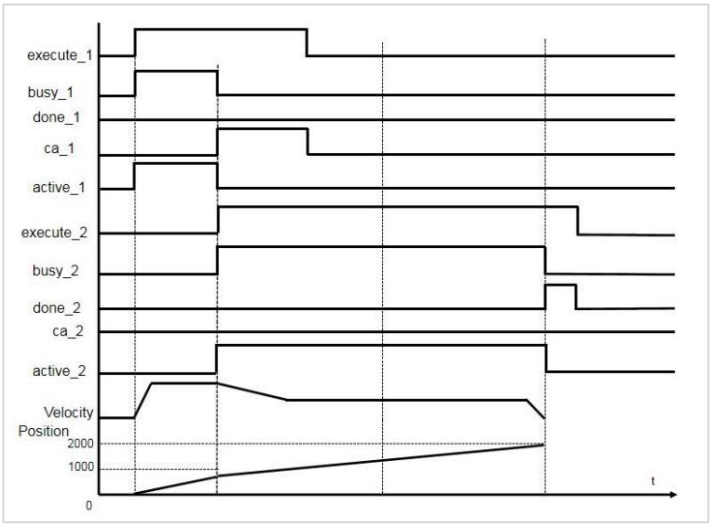
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

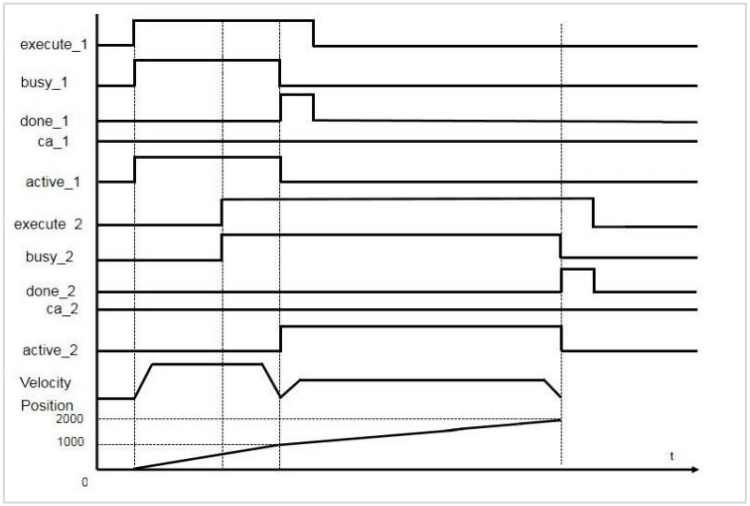
▪ Normal working condition sequence diagram (buffermode_1=0, buffermode_2=0):



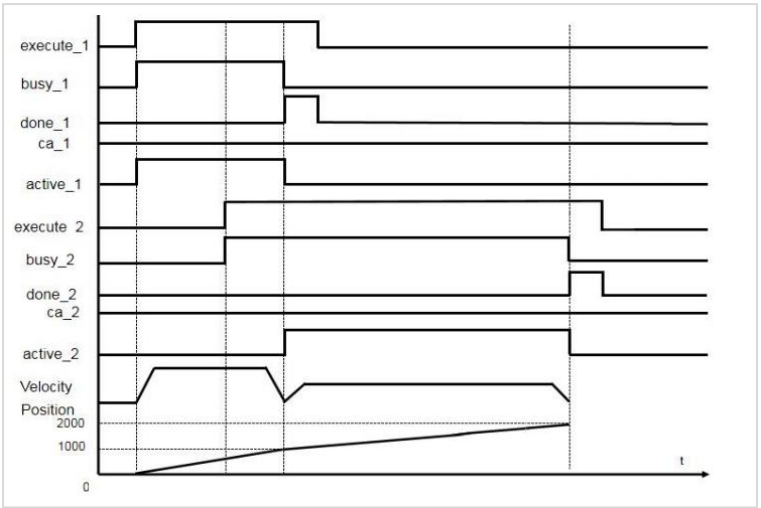
- mcAboring sequence diagram (buffermode_1=0, buffermode_2=0):



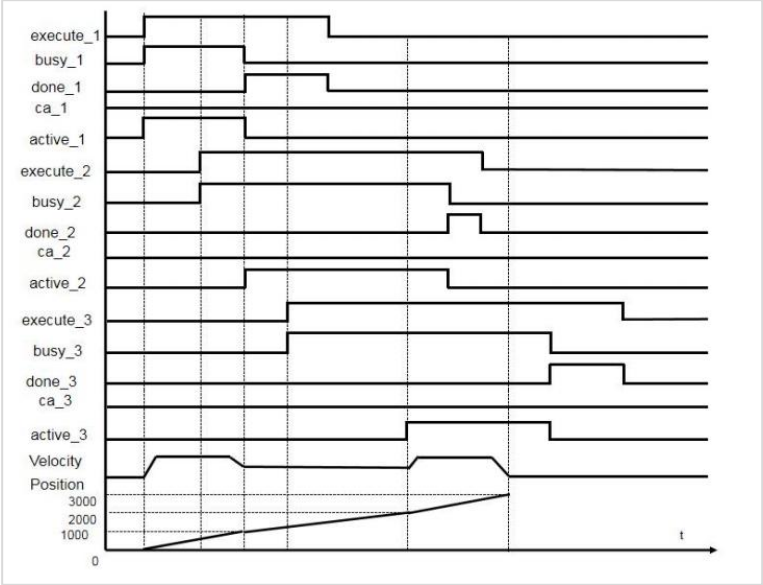
- mcBuffered sequence diagram (buffermode_1=0, buffermode_2=1):



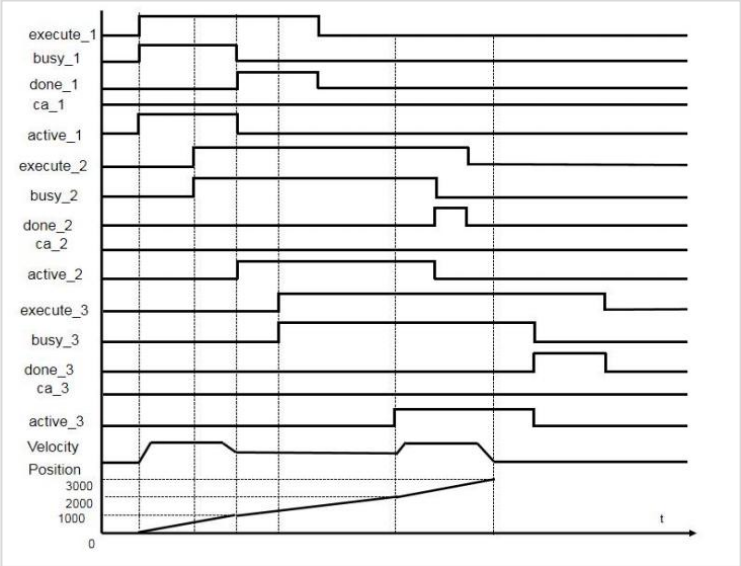
- mcBlendingLow sequence diagram (buffermode_1=0, buffermdoe_2=2, buffermode_3=2):



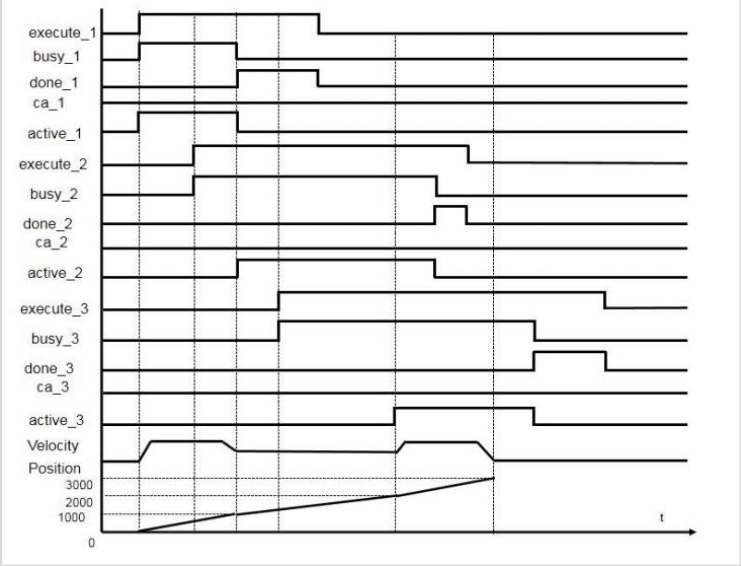
- mcBlendingPrevious sequence diagram (buffermode_1=0, buffermode_2=3, buffermode_3=3)



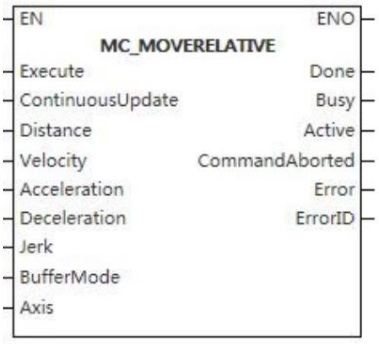
- mcBlendingNext sequence diagram (buffermode_1=0, buffermode_2=4， buffermode_3=4):



- mcBlendingHigh sequence diagram (buffermode_1=0, buffermode_2=5, buffermode_3=5):



8.2.5 MC_MOVERELATIVE



The function block makes the controlled axis reach the set absolute position with the input parameters.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVERELATIVE	Execute	●	●	●	●	●																					
	ContinuousUpdate	●	●	●	●	●																					
	Distance																										
	Velocity																										
	Acceleration																										
	Deceleration																										
	Jerk																										
	BufferMode																										
	Axis																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_MOVERELATIVE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	No	TRUE/FALSE	Updated during operation
		Distance	LREAL	No	-9999999.00~9999999.00	Relative position
		Velocity	LREAL	No	0 to 9999999.00	Velocity
		Acceleration	LREAL	No	0 to 9999999.00	Acceleration
		Deceleration	LREAL	No	0 to 9999999.00	Deceleration
		Jerk	LREAL	No	0 to 9999999.00	Jerk
		BufferMode	WORD	No	0 to 5	Mode selection
		Axis	WORD	No	0 to 65535	Axis number
	Output	Done	BOOL	No	TRUE/FALSE	Completion sign
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	No	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

The function block takes the current position as the starting point of motion and controls the controlled axis to reach the set relative position (Distance) with the set parameters (velocity, acceleration, deceleration and jerk),

It refers to the motion control that takes starting point of the motion + Distance as the end point.

Precautions

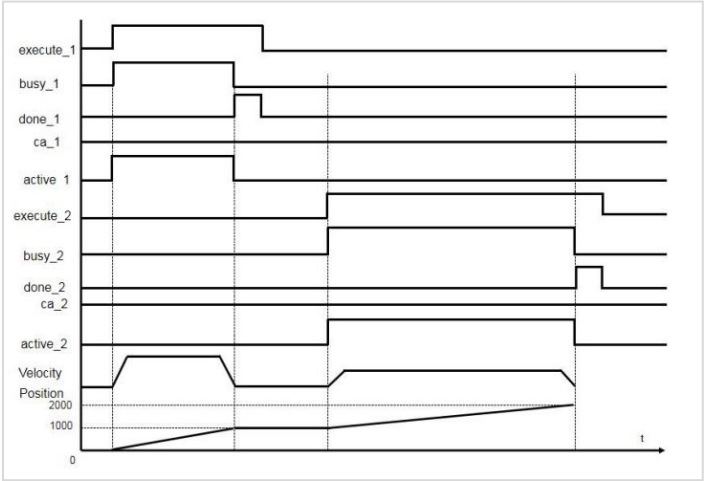
This function block is not available when the axis does not exist.

Error code

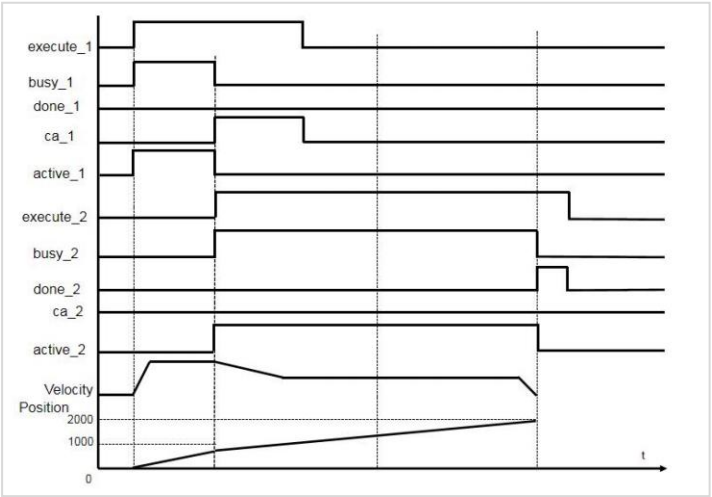
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

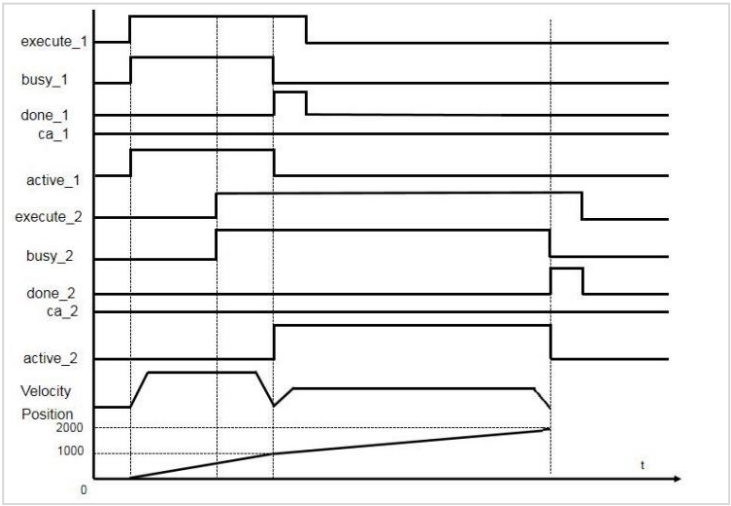
- Normal working condition sequence diagram (buffermode_1=0, buffermode_2=0):



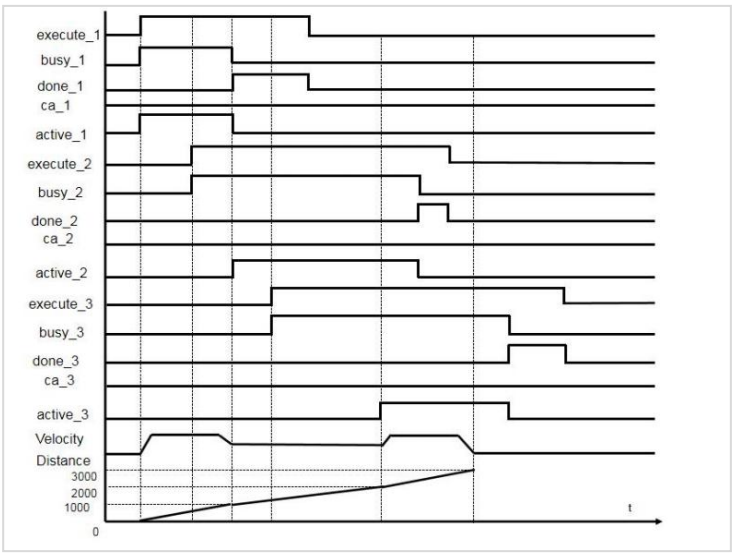
- mcAboring sequence diagram (buffermode_1=0, buffermode_2=0):



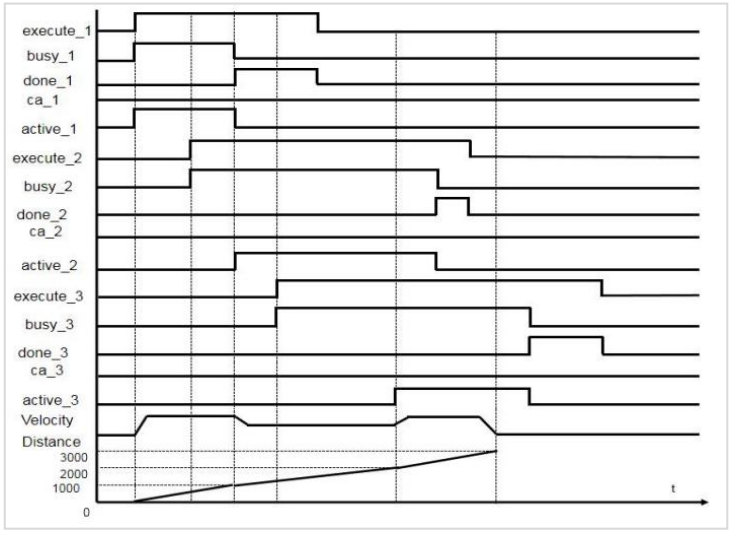
- mcBuffered sequence diagram (buffermode_1=0, buffermode_2=1):



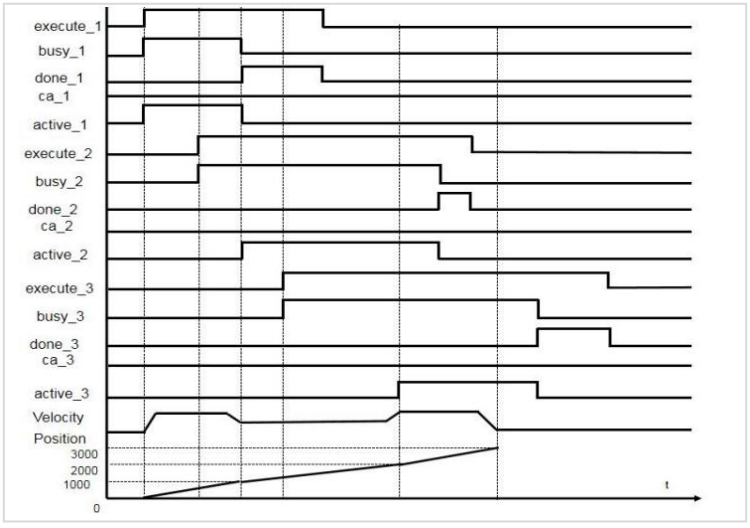
- mcBlendingLow sequence diagram (buffermode_1=0, buffermdoe_2=2, buffermode_3=2):



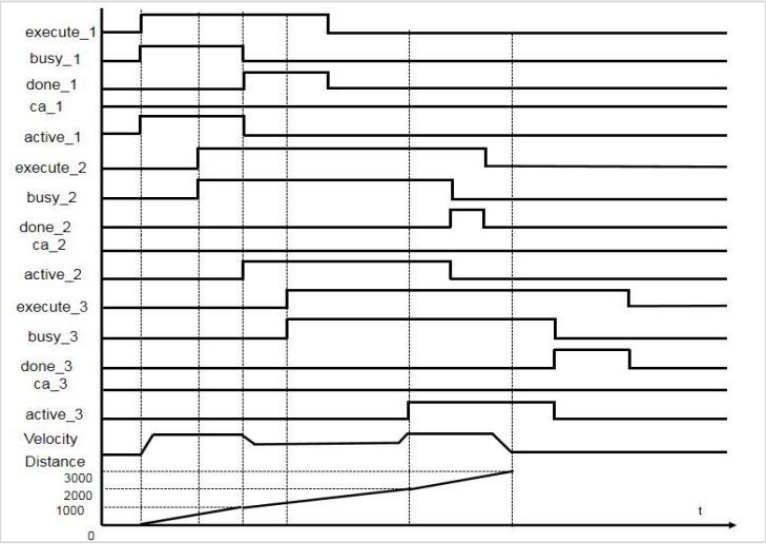
- mcBlendingPrevious sequence diagram (buffermode_1=0, buffermode_2=3, buffermode_3=3)



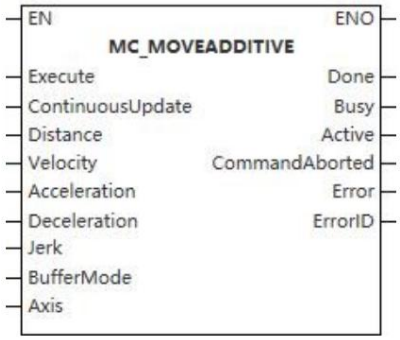
- mcBlendingNext sequence diagram (buffermode_1=0, buffermode_2=4，buffermode_3=4):



- McBlendingHigh sequence diagram (buffermode_1=0, buffermode_2=5, buffermode_3=5):


execute_1
busy_1
done_1
ca_1
active_1
execute_2
busy_2
done_2
ca_2
active_2
execute_3
busy_3
done_3
ca_3
active_3
Velocity
Distance
3000
2000
1000
0
t

8.2.6 MC_MOVEADDITIVE



This function block is a motion increment function block.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVEADDITIVE	Execute	●	●	●	●	●																					
	ContinuousUpdate	●	●	●	●	●																					
	Distance																										
	Velocity																										
	Acceleration																										
	Deceleration																										
	Jerk																										
	BufferMode																										
	Axis																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_MOVEADDITIVE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	No	TRUE/FALSE	Updated during operation
		Distance	LREAL	No	-9999999.00~9999999.00	Relative position
		Velocity	LREAL	No	0 to 9999999.00	Velocity
		Acceleration	LREAL	No	0 to 9999999.00	Acceleration
		Deceleration	LREAL	No	0 to 9999999.00	Deceleration
		Jerk	LREAL	No	0 to 9999999.00	Jerk
		BufferMode	WORD	No	0 to 5	Mode selection
		Axis	WORD	No	0 to 65535	Axis number
	Output	Done	BOOL	No	TRUE/FALSE	Completion sign
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	No	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

▪ It is used to add the input parameters (velocity, acceleration, deceleration and jerk) specified by this function block to reach the specified distance (Distance) on the basis of existing motion, i.e., adding distance.

Precautions

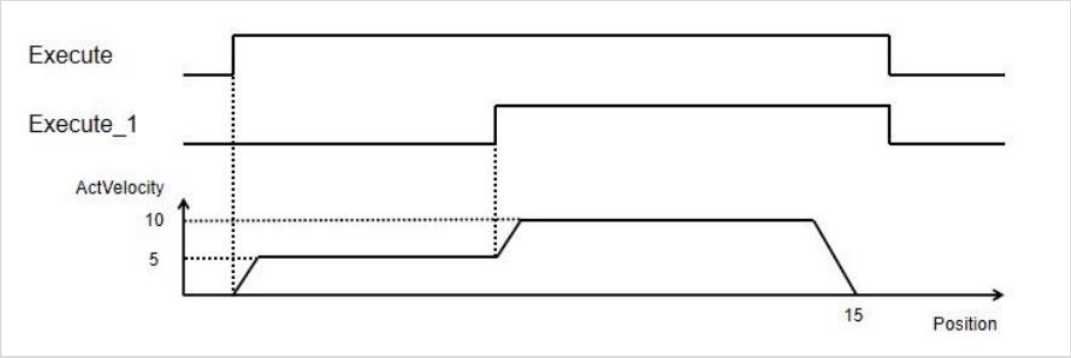
▪ This function block is not available when the axis does not exist.

Error code

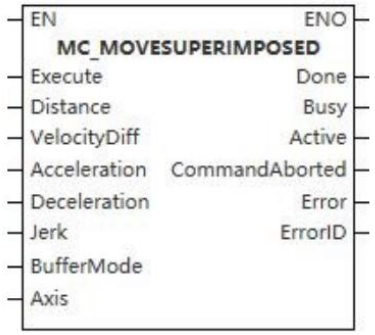
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

When using MC_MOVEABSOLUTE with MC_MOVEADDITIVE together, and when the tagret position of MC_MOVEABSOLUTE is 10 and velocity is 5, the stacked position of MC_MOVEADDITIVE is 15 and velocity is 10, the relationship between position and velocity of Axis_0 is as follows:



8.2.7 MC_MOVESUPERIMPOSED



This function block is a motion superposition function block.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVESUPERIMPOSED	Execute	●	●	●	●	●																				
	ContinuousUpdate	●	●	●	●	●																				
	Distance																									
	VelocityDiff																									
	Acceleration																									
	Deceleration																									
	Jerk																									
	BufferMode																									
	Axis																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_MOVESUPERIMPOSED	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	No	TRUE/FALSE	Updated during operation
		Distance	LREAL	No	-9999999.00~9999999.00	Relative position
		VelocityDiff	LREAL	No	0 to 9999999.00	Velocity
		Acceleration	LREAL	No	0 to 9999999.00	Acceleration
		Deceleration	LREAL	No	0 to 9999999.00	Deceleration
		Jerk	LREAL	No	0 to 9999999.00	Jerk
		BufferMode	WORD	No	0 to 5	Mode selection
		Axis	WORD	No	0 to 65535	Axis number
	Output	Done	BOOL	No	TRUE/FALSE	Completion sign
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	No	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

▪ This function block is used to superpose the motion, which means that on the basis of current motion speed, the velocity difference is added so that the input acceleration, deceleration and jerk are also increased based on the original motion values, i.e., increasing the velocity and making the controlled axis move to the specified (relative) distance (Distance).

Precautions

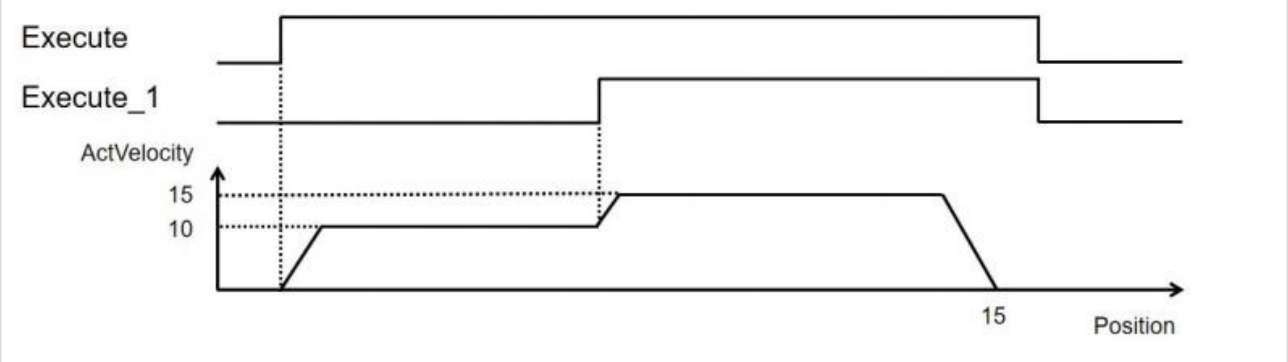
▪ This function block is not available when the axis does not exist;

▪ When it executes, the previous function block should be executed first.

Error code

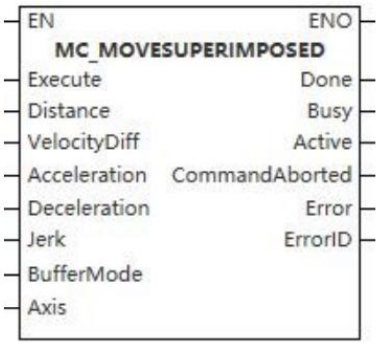
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram



8.3 Management Function Blocks for Multi-axis Motion Control

8.3.1 MC_CAMTABLESELECT



This function block is used to bind the cam table.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_CAMTABLESELECT	Execute	●	●	●	●	●																				
	Periodic	●	●	●	●	●																				
	MasterAbsolute	●	●	●	●	●																				
	SlaveAbsolute	●	●	●	●	●																				
	Master																									
	Slave																									
	CamTable																	●								
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									
	CamTableID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_CAMTABLESELECT	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Periodic	BOOL	No	TRUE/FALSE	Cyclic mode setting
		MasterAbsolute	BOOL	No	TRUE/FALSE	Master axis position mode setting
		SlaveAbsolute	BOOL	No	TRUE/FALSE	Slave axis position mode setting
		Master	WORD	No	-	Master axis
		Slave	WORD	No	-	Slave axis
		CamTable	Cam table (INT type variable)	No	-	Cam table
	Output	Done	BOOL	No	TRUE/FALSE	Completion sign
		Busy	BOOL	No	TRUE/FALSE	Busy sign
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code
		CamTableID	WORD	No	-	Cam table identification number

Function

▪ When MasterAbsolute is true, it means the master axis is under the absolute mode; when MasterAbsolute is false, it means the master axis is under the relative mode;

▪ When SlaveAbsolute is true, it means the slave axis is under the absolute mode; when SlaveAbsolute is false, it means the slave axis is under the relative mode;

▪ This function block is used to bind the cam table, that is, to bind the input master axis, slave axis and cam table. When calling MC_CAMIN function block, only the cam table bound by this function block can be used.

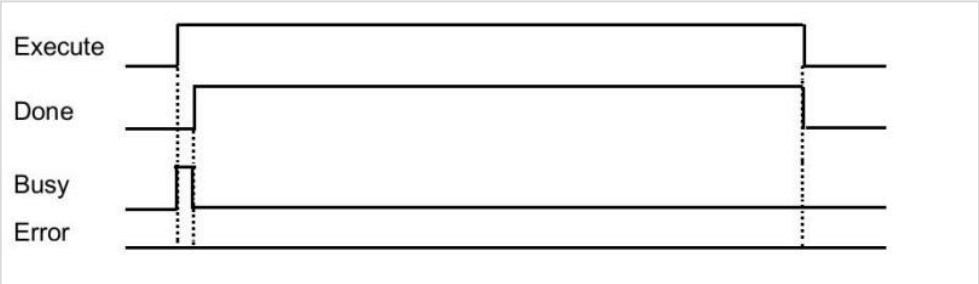
Precautions

▪ This function block is not available when the axis does not exist.

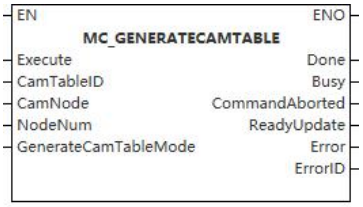
Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram



8.3.2 MC_GENERATECAMTABLE



This function block is used to update the cam table.

Device used (Only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GENERATECAMTABLE	Execute	●	●	●	●	●																				
	CamTableID																									
	CamNode																									
	NodeNum																	●								
	Generatecam TableMode																	●								
	Done	●	●	●	●	●																				
	Busy	●	●	●	●	●																				
	CommandAborted	●	●	●	●	●																				
	Error	●	●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin	Parameter	Variable type	Can it be empty	Range	Description
MC_GENERATECAMTABLE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		CamTableID	INT	No		Cam tableID
		CamNode	ARRAY	No	-	Array of cam nodes
		NodeNum	INT	No	2 ～361	Number of cam nodes
		Generate CamTable Mode	INT	No	0	Effective mode 0: Takes effect in the next cam cycle Others: Reserved
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		BusY	BOOL	Yes	TRUE/FALSE	Busy flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		ReadyUpdate	BOOL	Yes	TRUE/FALSE	Cam table update comp flag
		Error	BOOL	Yes	-	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error Code (Please check [8.11 Error Code])

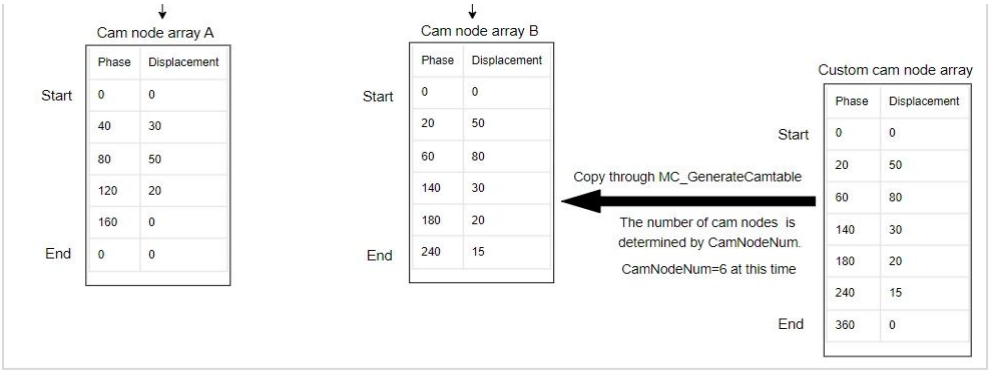
Function

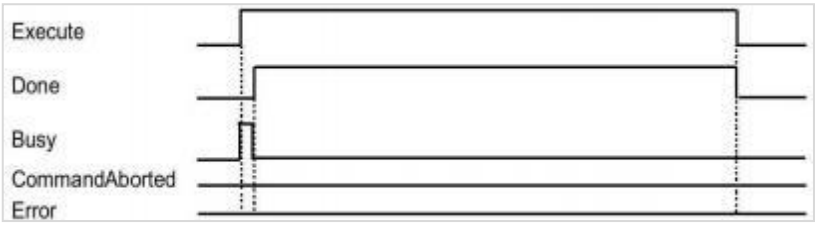
- This function block is used to update cam table. At the rising edge of startup, the cam data is calculated based on values of input variables camNode and NodeNum. The data is updated to the cam table specified by camTable, which is will be effective in the next cam cycle.

✎Note:

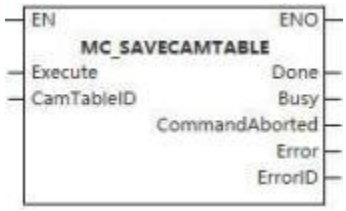
This function block is not available when the Cam table does not exist.

Sequence diagram





8.3.3 MC_SAVECAMTABLE



This function block is used to save the cam table.

Device used (Only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	S M	T(bit)	C(bit)	LC(bit)	HSC(bit)	D. b	Kn X	Kn Y	Kn M	Kn S	T	C	D	R	S D	L C	HS C	K	H	E	[D]	XXP
MC_SAVECAMTABLE	Execute	●	●	●	●	●																					
	CamTableID																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_SAVECAMTABLE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		CamTableID	WORD	No	[-1,65535]	Specify the cam table index to save; -1 indicates saving all cam tables.
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		BusY	BOOL	Yes	TRUE/FALSE	Busy flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	-	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error Code Please refer to the [8.11 Error Code]

Function

- This function block is used to save the specified cam table.

✎Note:

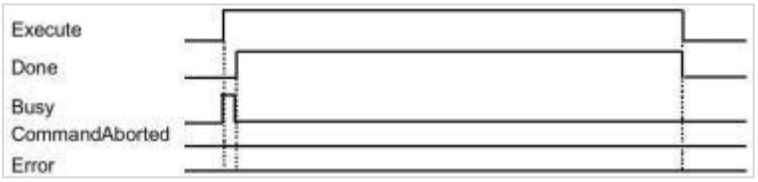
This function block is not available when the Cam table does not exist.

In the process of executing this instruction, the controller cannot be powered off. Otherwise, the data will fail to be saved and the cam data will be lost.

Error code

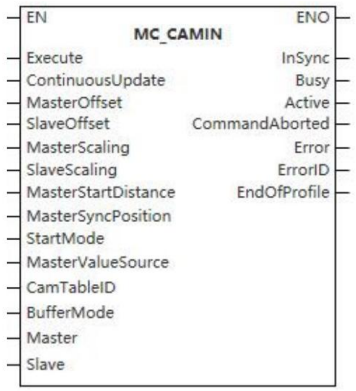
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram



8.4 Motion Function Block for Multi-axis Motion Control

8.4.1 MC_CAMIN



The function block enables the controlled axis to carry out cam motion.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_CAMIN	Execute	●	●	●	●	●																				
	ContinuousUpdate	●	●	●	●	●																				
	MasterIndex																									
	SlaveIndex																									
	MasterScaling																									
	SlaveScaling																									
	MasterStartDistance																									
	MasterSyncPosition																									
	StartMode																									
	MasterValueSource																									
	CamTableID																									
	BufferMode																									
	Master																									
	Slave																									
	InSync		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									
	EndOfProfile																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_CAMIN	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	No	TRUE/FALSE	Updated during operation
		MasterIndex	LREAL	No	-9999999.00~9999999.00	Master axis offset
		SlaveIndex	LREAL	No	-9999999.00~9999999.00	Slave axis bias
		MasterScaling	LREAL	No	-9999999.00~9999999.00	Master axis scaling factor
		SlaveScaling	LREAL	No	-9999999.00~9999999.00	Slave axis scaling factor
		MasterStartDistance	LREAL	No	-9999999.00~9999999.00 (not 0)	Master axis starting distance
		MasterSyncPosition	LREAL	No	-9999999.00~9999999.00	Master axis synchronization position
		StartMode	WORD	No	0 to 2	Start mode (0-Absolute, 1-Relative, 2-Catchup)
		MasterValueSource	WORD	No	0 to 2	Master axis synchronization source (Instruction value, set value and actual value)
		CamTableID	WORD	No	0 to 9999999.00	Electronic CAM table identification
		BufferMode	WORD	No	0	Mode selection
		Master	WORD	No	-	Master axis
		Slave	WORD	No	-	Slave axis
	Output	InSync	BOOL	No	TRUE/FALSE	Completion sign
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	No	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code
		EndOfProfile	WORD	No	-	Electronic CAM end signal

Function

▪ This function starts with the input parameter MasterSyncPosition. Before reaching the MasterSyncPosition, the catchup or transition motion is carried out according to the set parameters (absolute relative of the master axis, absolute relative of the slave axis, starting mode). After reaching the MasterSyncPosition position, the positional relationship between the master axis and the slave axis is obtained according to the camTableID, and the relationship is corrected under the set offset and scaling factors of the master axis and the slave axis, and the cam motion is carried out.

▪ MasterSyncPosition

MasterSyncPosition is the absolute position when the axis mode is absolute;

MasterSyncPosition is the relative position when the axis mode is relative;

▪ ImSync status bit is turned ON when the master axis reaches MasterSyncPosition position and the slave axis reaches the slave axis position corresponding to the cam table of the master axis. Generally speaking, when the slave axis is in relative mode, the MC_CAMIN instruction is executed, and this status bit will be turned ON immediately; when the slave axis is in absolute or tracing mode, the status bit will be turned ON immediately after the slave axis jumps or traces to the position corresponding to the master axis cam table.

▪ The EndOfProfle status bit is turned ON after the slave axis has followed the master axis through the entire cam table; The StartMode parameter, together with the MasterAbsolute/SlaverAbsolute parameters in the MC_CAMTABLESELECT instruction, determines the motion mode of the master/slave axis.

▪ The master mode is only determined by MasterAbsolute and is not affected by the value in StartMode. The slave axis mode is expressed as follows:

StartMode	SlaveAbsolute	Slave axis mode
Absolute	Relative	Relative
Absolute	Absolute	Absolute
Relative	Relative	Relative
Relative	Absolute	Relative
Catch up	Relative	Relative catch up
Catch up	Absolute	Relative catch up

Master axis mode	Slave axis mode	Result
Relative	Relative	When CAMIN is executed, slave axis position jumps to the set offset position; When master axis is executing, slave axis performs relative motion according to the start position of cam table.
	Absolute	When CAMIN is executed, slave axis position jumps to the cam table start position (i.e., 0); When the master axis is executing, slave axis moves according to the corresponding point position of the cam table.
	Relative	When CAMIN is executed, slave axis position catches up to the set offset position; When the master axis is executing, the slave axis performs relative motion according to the corresponding points of the cam table.
	Absolute catch up	When CAMIN is executed, slave axis position catches up to the cam table starting position (i.e., 0); When the master axis is executing, the slave axis moves according to the corresponding point position of the cam table.
Absolute	Relative	When CAMIN is executed, slave axis position jumps to the set offset position; When the master axis is executing, the slave axis performs relative motion according to the points corresponding to the cam table.
	Absolute	When CAMIN is executed, slave axis position jumps to the current slave axis position corresponding to the master axis in the cam table (For example, if the current position of the master axis is 100 and the slave axis point corresponding to the master axis point 100 in the cam table is 200, the slave axis jumps after CAMIN is executed;) After the master axis is executing, slave axis performs relative motion according to the corresponding points in the cam table.
	Relative catch up	When CAMIN is executed, slave axis position catches up to the set offset position; When the master axis is executing, slave axis performs relative motion according to the corresponding points of the cam table.
	Absolute catch up	When CAMIN is executed, slave axis position catches up to the current slave axis position corresponding to the master axis in the cam table (For example, if the current position of the master axis is 100 and the slave axis point corresponding to the master axis point 100 in the cam table is 200, the slave axis jumps after CAMIN is executed;) After the master axis is running, the slave axis performs relative motion to the corresponding points in the cam table.

▪ When axis is in absolute mode, if the current position of the axis is not within the range of the axis in the cam table, it will be processed by self-acting periodization.

Example: the current axis position is 110, and the axis position in the cam table is 0~100, then the default axis position after CAMIN execution is 10 (the actual axis position does not change);

▪ The master-slave axis multiplier and master-slave axis offset parameters take effect when CAMIN is executed and do not support modification in the process; unsuitable parameters will cause the slave axis position to jump.

▪ The master-slave position relationship (where CAM () indicates the corresponding slave position of the master axis on the cam table):

Master Position = Master Position * Master Multiplier + Master Offset

Slave position = Slave multiplier * CAM (spindle position) + Slave offset

▪ The master-slave multiplier cannot be 0.

Precautions

▪ This function block is not available when the axis does not exist;

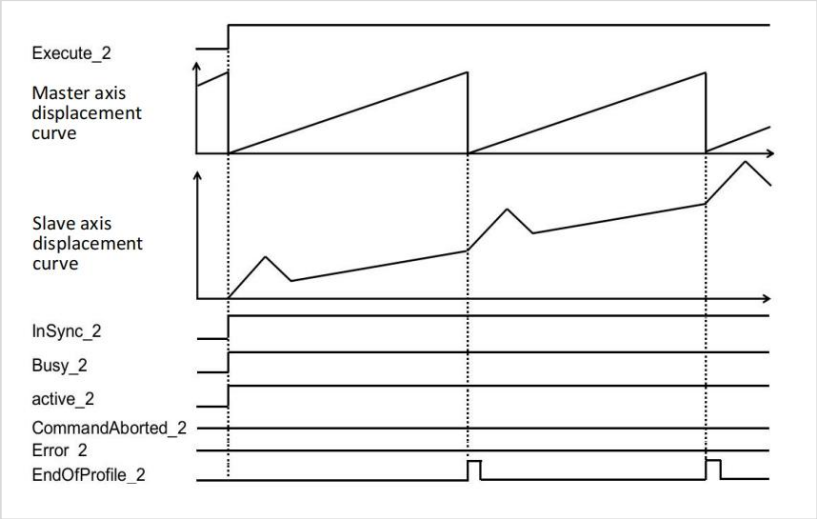
▪ The motion mode of the slave axis shall be determined according to start mode and SlaveAbsolute of CamtableSelect. See “Chapter 8.6” [Description of Several Modes in E-CAM] for details. MasterValueSource is now an actual value regardless of which mode you choose.

Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

▪ MC_CAMIN needs to be used in conjunction with MC_CAMTABLESELECT and a single-axis motion (e.g., MC_MOVEABSOLUTE), the sequence diagram is as follows.



(1) Slave motion varies depending on StartMode.

Assume that the start conditions for cam motion are as follows:

▪ In condition 1, the master axis jumps at the position of 20 (70~50), and the jumping position is the cam table position corresponding to masterSyncPosition, and the output variable InSync (in synchronization) changes to TRUE when it passes the position of 70, and the slave axis starts camming.

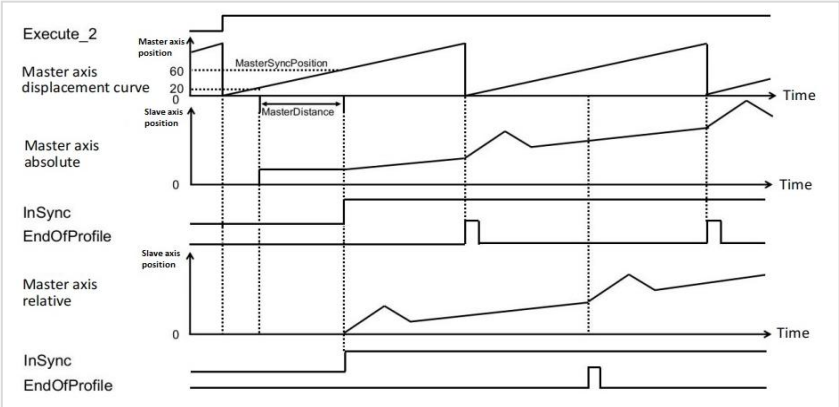
▪ In condition 2, the master axis will catch up at 20 (70~50), and the catch-up position is the cam table position corresponding to masterSyncPosition, and the output variable InSync (in synchronization) becomes TRUE when it passes the position of 70, and the slave axis starts camming.

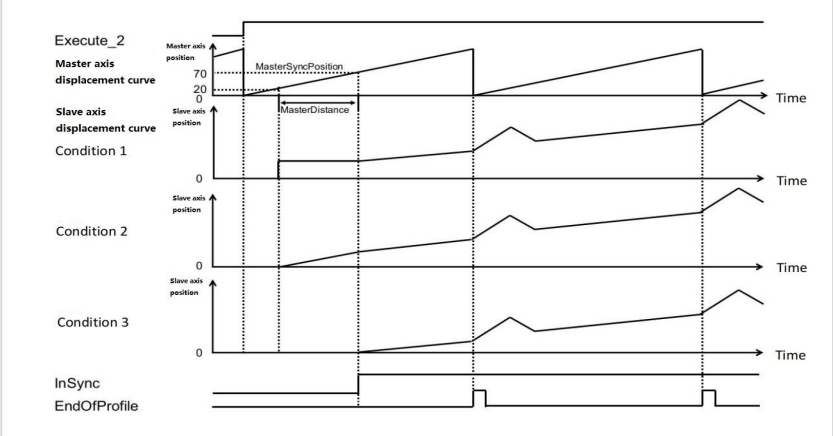
▪ In condition 3, the output variable InSync (in synchronization) becomes TRUE when the master axis passes the position of 70, and the slave axis starts camming.

Input variable	Condition 1	Condition 2	Condition 3
MasterAbsolute	True (Absolute)	True (Absolute)	True (Absolute)
SlaveAbsolute	True (Absolute)	True (Absolute)	FALSE (Relative)
StartMode	MC_START_MODE_ABSOLUTE	MC_START_MODE_RAMPIN	MC_START_MODE_RELATIVE
MasterSyncPosition	70	70	70
MasterDistace	50	50	50
MasterOffset	0	0	0
SlaveOffset	0	0	0
MasterScaling	1	1	1
SlaveScaling	1	1	1

(2) Slave motion varies depending on MasterAbsolute.







(3) Slave motion varies depending on SlaveAbsolute.

Cam table

Phase	Displacement
0	0
60	150
120	50
180	180
240	240
300	300
360	360

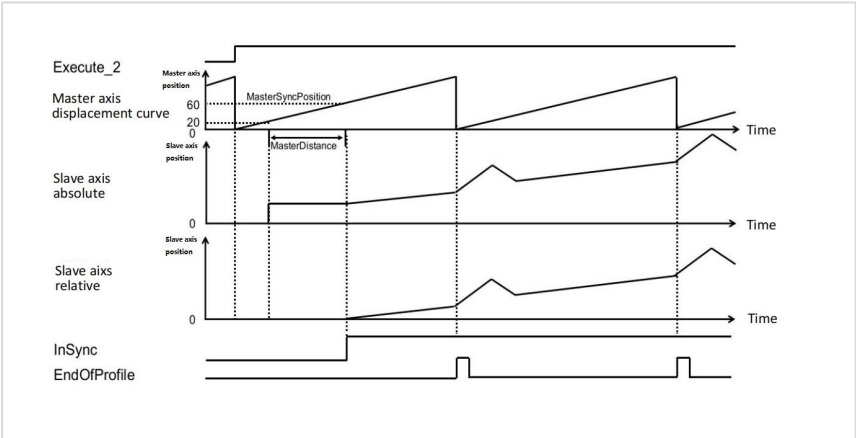
MasterSyncPosition=60

MasterAbsolute

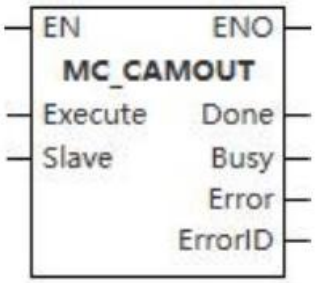
Master axis position	Slave axis position
60	150+slave axis position in cam table start
120	50+slave axis position in cam table start
180	180+slave axis position in cam table start
240	240+slave axis position in cam table start
300	300+slave axis position in cam table start
360	360+slave axis position in cam table start

SlaveAbsolute

Master axis position	Slave axis position
60	150+slave axis position in cam table start-150
120	50+slave axis position in cam table start-150
120	180+slave axis position in cam table start-150
180	240+slave axis position in cam table start-150
240	300+slave axis position in cam table start-150
300	360+slave axis position in cam table start-150



8.4.2 MC_CAMOUT



The function block enables the controlled axis to carry out uniform motion at current velocity.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Pin type	Parameter	Devices																							Index modification	Pulse expansion
			X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_CAMOUT	Input	Execute	●	●	●	●	●																				
		Slave																									
	Output	Done		●	●	●	●																				
		Busy		●	●	●	●																				
		Error		●	●	●	●																				
		ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Can be empty or not	Range	Description
MC_CAMOUT	Execute	BOOL	No	TRUE/FALSE	Enable
	Slave	WORD	No	0 to 65535	Slave axis
	Done	BOOL	No	TRUE/FALSE	Completion sign
	Busy	BOOL	No	TRUE/FALSE	Busy sign
	Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
	ErrorID	DWORD	No	-	Error code

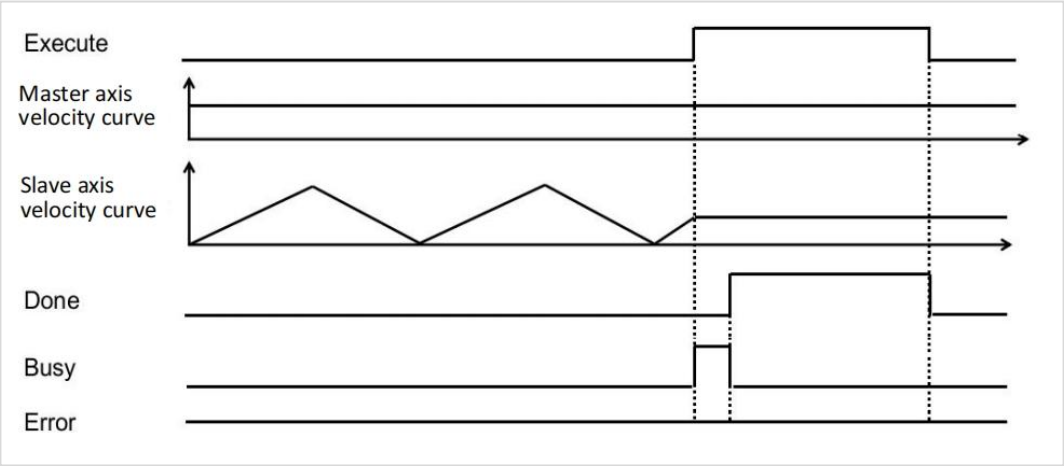
Function

- The function is used to separate the slave axis from the position relation specified for the electronic cam. Execute the MC_CamOut function block to transform the slave shaft from the synchronous state to the continuous motion state. The current velocity of the slave axis is maintained.

Precautions

- This function block is not available when the axis does not exist.

Sequence diagram



8.4.3 MC_GEARIN



The function block enables the controlled axis to carry out gear motion.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GEARIN	Execute	●	●	●	●	●																				
	ContinuousUpdate	●	●	●	●	●																				
	RatioNumerator																									
	RatioDenominator																									
	MasterValueSource																									
	Acceleration																									
	Deceleration																									
	Jerk																									
	BufferMode																									
	Master																									
	Slave																									
	InGear		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_GEARIN	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	No	TRUE/FALSE	Updated during operation
		RatioNumerator	INT16	No	-32768~32767	Gear velocity ratio (numerator)
		RatioDenominator	UINT16	No	0 to 65535	Gear velocity ratio (denominator)
		MasterValueSource	WORD	No	0 to 2	Master axis synchronization source (Instruction value, set value and actual value)
		Acceleration	LREAL	No	0 to 9999999.00	Acceleration
		Deceleration	LREAL	No	0 to 9999999.00	Deceleration
		Jerk	LREAL	No	0 to 9999999.00	Jerk
		BufferMode	WORD	No	0	Selection
		Master	WORD	No	-	Master axis
		Slave	WORD	No	-	Slave axis
	Output	InGear	BOOL	No	TRUE/FALSE	Completion flag
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	No	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

▪ The function block is used for establishing the synchronization of the velocity ratio between the master axis and the slave axis. Firstly, the slave axis takes the velocity of master axis multiplied by the velocity ratio as the target velocity and moves at the set acceleration and jerk and executes gear motion according to the velocity of master axis when reaching the target motion.

Precautions

▪ This function block is not available when the axis does not exist.

Error code

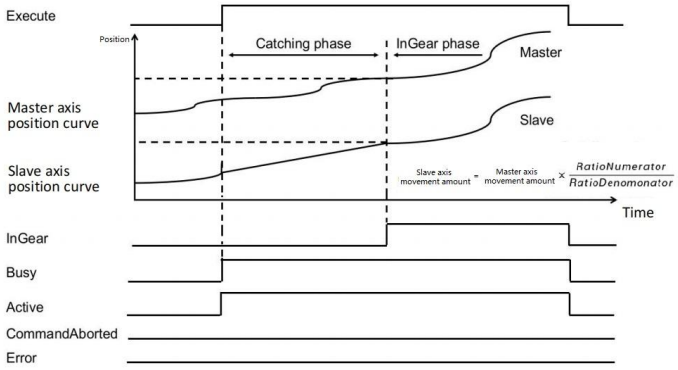
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

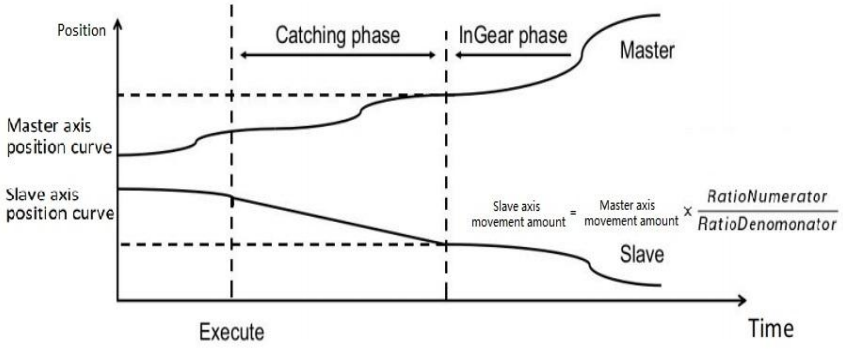
▪ After starting the motion, the slave axis performs acceleration and deceleration with the velocity obtained by multiplying the spindle speed by the gear ratio as the target speed;

▪ Before reaching target position is called the Catching phase and after reaching it is called the InGear phase;

▪ The gear ratio is positive and the slave axis moves in the same direction as the master axis.



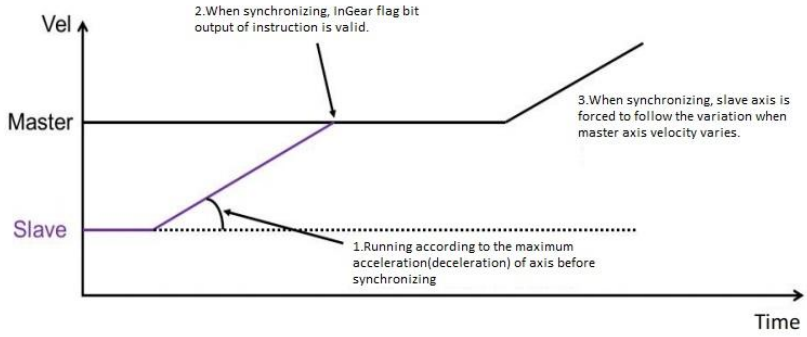
- The gear ratio is negative and the slave axis moves in the reverse direction to the master axis.



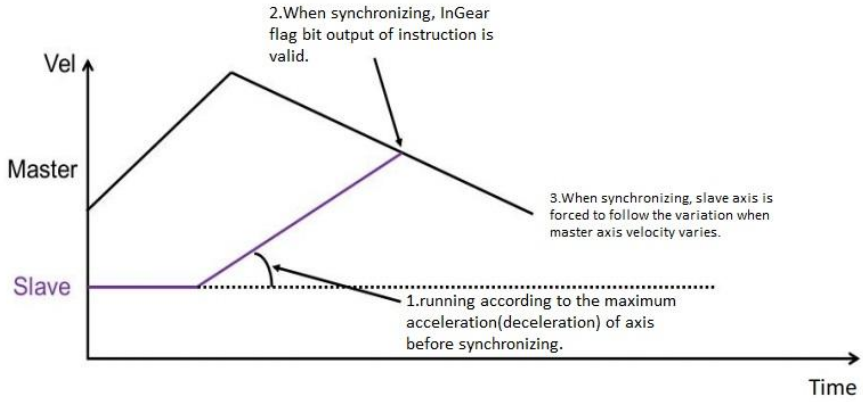
Before reaching synchronization, the slave axis moves according to the maximum acceleration set by the axis as plus or minus speed, and the slave axis velocity is recognized when it is equal to the speed of the master axis multiplied by the gear ratio.

When the speed of the slave axis equals the speed of the master axis multiplied by the gear ratio, the gears are recognized as meshing, and thereafter the slave axis follows the master axis completely.

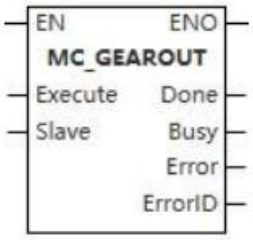
① Master axis maintains constant velocity before synchronization (Gear instruction is triggered according to 1:1 gear ratio)



② Master axis moves in variable velocity before synchronization (Gear instruction triggered according to 1:1 gear ratio)


2.When synchronizing, InGear flag bit output of instruction is valid.
Vel
Master
Slave
3.When synchronizing, slave axis is forced to follow the variation when master axis velocity varies.
1.running according to the maximum acceleration(deceleration) of axis before synchronizing.
Time

8.4.4 MC_GEAROUT



The function block enables the controlled axis to carry out uniform motion at current velocity.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GEAROUT	Execute	●	●	●	●	●																				
	Slave																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_GEAROUT	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Slave	WORD	No	0 to 65535	Axis number
	Output	Done	BOOL	No	TRUE/FALSE	Completion sign
		Busy	BOOL	No	TRUE/FALSE	Busy sign
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

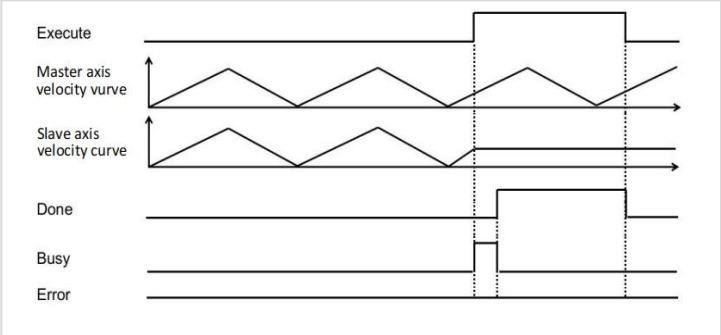
Function

- The function block is used for separating the slave axis from the master axis at the velocity ratio and maintaining the motion at the velocity when it separates.

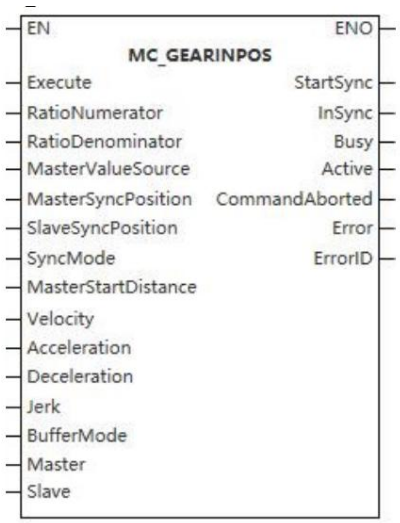
Precautions

- This function block is not available when the axis does not exist.

Sequence diagram



8.4.5 MC_GEARINPOS



The function block enables the controlled axis to carry out gear motion.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GEARINPOS	Execute	●	●	●	●	●																				
	RatioNumerator																									
	RatioDenominator																									
	MasterValueSource																									
	MasterSyncPosition																									
	SlaveSyncPosition																									
	SyncMode																									
	MasterStartDistance																									
	Velocity																									
	Acceleration																									
	Deceleration																									
	Jerk																									
	BufferMode																									
	Master																									
	Slave																									
	StartSync																									
	InSync		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
MC_GEARINPOS	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		RatioNumerator	DINT	No	-32768~32767	Gear velocity ratio (numerator)
		RatioDenominator	UINT	No	0 to 65535	Gear velocity ratio (denominator)
		MasterValueSource	WORD	No	0 to 2	Master axis synchronization source (Instruction value, set value and actual value)
		MasterSyncPosition	LREAL	No	-9999999.00~9999999.00	Master axis synchronization position
		SlaveSyncPosition	LREAL	No	-9999999.00~9999999.00	Slave axis synchronization position
		SyncMode	WORD	No	0 to 2	Synchronous mode (shortest, catchup, slowdown)
		MasterStartDistance	LREAL	No	-9999999.00~9999999.00 (not 0)	Master axis starting distance
		Velocity	LREAL	No	0~9999999.00 (not 0)	Velocity
		Acceleration	LREAL	No	0 to 9999999.00	Acceleration
		Deceleration	LREAL	No	0 to 9999999.00	Deceleration
		Jerk	LREAL	No	0 to 9999999.00	Jerk
		BufferMode	WORD	No	0	Buffer mode selection
		Master	WORD	No	-	Master axis
		Slave	WORD	No	-	Slave axis
	Output	StartSync	BOOL	No	TRUE/FALSE	Synchronous identification
		InSync	BOOL	No	TRUE/FALSE	Synchronized sign
		Busy	BOOL	No	TRUE/FALSE	Busy flag
		Active	BOOL	No	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	No	TRUE/FALSE	Abort flag
		Error	BOOL	No	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	No	-	Error code

Function

- The function block is used to achieve the specified velocity ratio between the slave axis and master axis at the specific position through a set maximum velocity, acceleration, deceleration, jerk and synchronous mode.

Precautions

- This function block is not available when the axis does not exist.

Error code

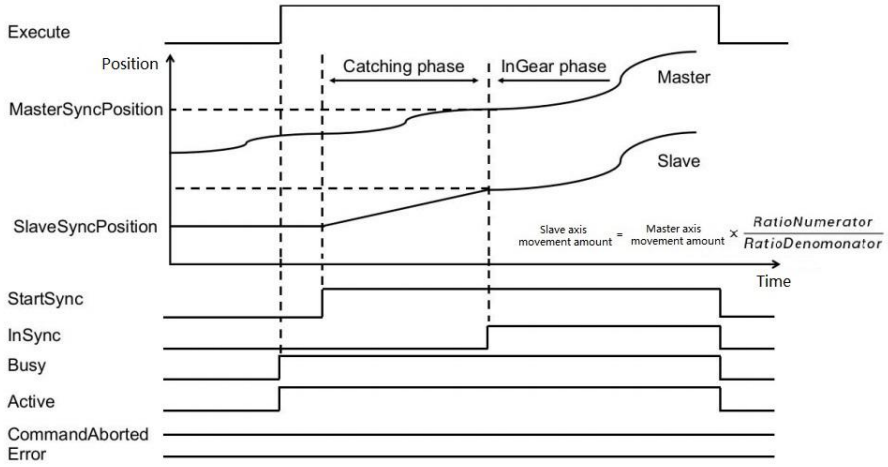
Error code	Content
4084H	Data beyond the specifiable range was entered.

Sequence diagram

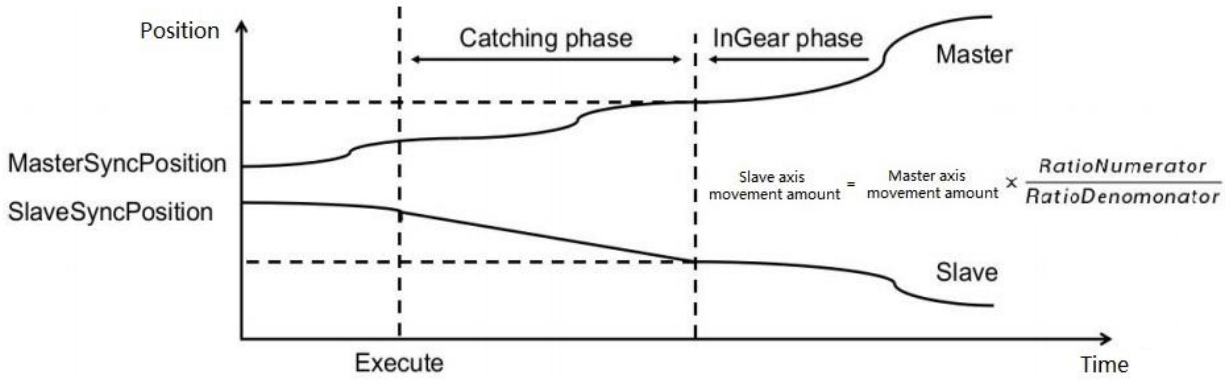
- After starting the motion, the slave axis performs acceleration and deceleration with the speed obtained by multiplying the axis velocity by the gear ratio as the target velocity;
- The position obtained by ▪ (MasterSyncPosition - MasterStartDistance) is called the start position, and after the master axis reaches

this position, the slave axis starts to catching up with the target;

- If the gear ratio is positive, the slave axis moves in the same direction as the master axis.

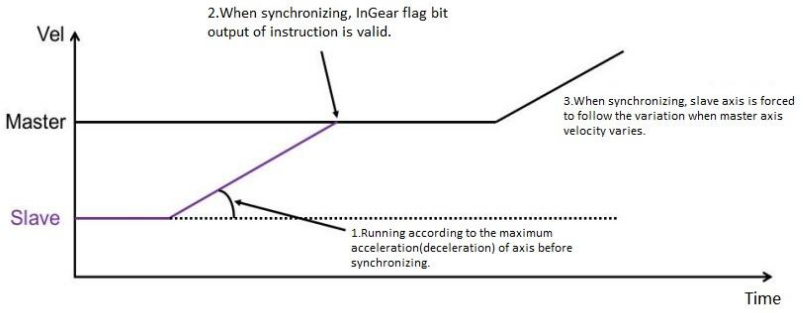


- If the gear ratio is negative, the slave axis moves in the reverse direction to the master axis.

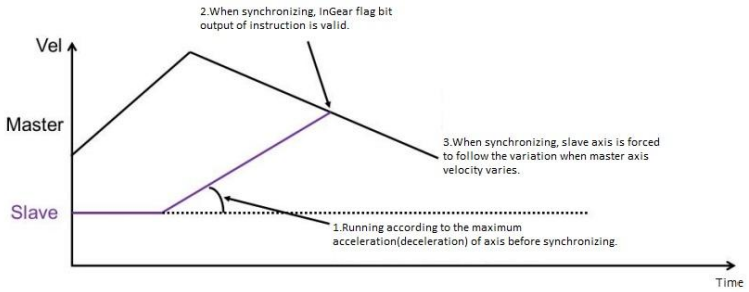


- Before reaching synchronization, the slave axis moves at the maximum acceleration set by the axis as plus or minus speed, and is recognized when the slave axis speed is equal to the speed of the master axis multiplied by the gear ratio.
- When the speed of the slave axis equals the velocity of the master axis multiplied by the gear ratio, the gears are considered to be meshed, and thereafter the slave axis follows the main axis completely.

① The master axis moves at a constant speed until synchronization is achieved (The gear instruction is triggered at a gear ratio of 1:1).



②Master axis moves in variable velocity before synchronization (Gear instruction triggered according to 1:1 gear ratio)



gearInPos carries out the pursuit motion in the following three modes, all following the spindle in gear motion after the target speed has been reached:

- Shortest: catching in constant acceleration mode;
- Catchuo: catching in S-curve;
- Slowdown: catching up in T-curve.

Note: When the amount of movement in the chase is short, there may be an abrupt change in speed from the axis to the target position and then directly to the target speed.

8.5 Management functions of axis group motion control

8.5.1 MC_ADDAXISTOGROUP



The function block is to add axis to axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																									Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_ADDAXISTOGROUP	Execute	●	●	●	●	●																						
	IdentInGroup																											
	AxesGroup																											
	Axis																											
	Done		●	●	●	●																						
	Busy		●	●	●	●																						
	Error		●	●	●	●																						

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_ADDAXISTOGROUP	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		IdentInGroup	WORD	No	[0 to 2]	Axis sequence flag
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
		Axis	WORD	No	[0 to 65535]	Axis number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 10 Error Code】

Features

▪The rising edge of Execute triggers and Axis is added to Axes Group. Instructions are not buffered and axis do not move;

▪IndentInGroup is a flag of the adding axis order to axis group.

✎**Note:**

▪If the serial number of the axis group does not exist, an error will be reported;

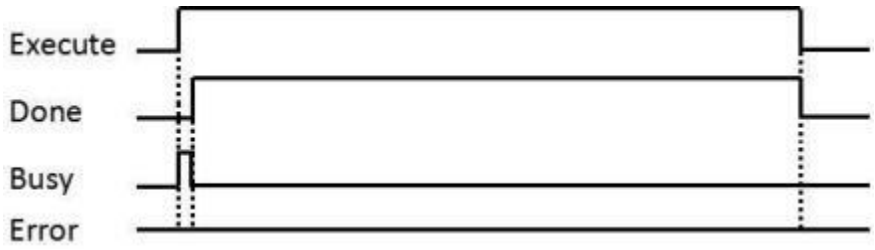
▪When the axis group is enabled, an error will be reported when add new axis;

▪Axis group cannot move when only adding axis to axis group. It is necessary to call MC_GroupEnable to enable the motion.

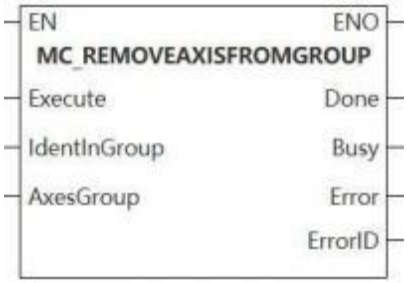
Error code

Error code	Content
4084H	Data beyond the specifiable range was input.

Sequence diagram



8.5.2 MC_REMOVEAXISFROMGROUP



This function block removes axis from an axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_REMOVEAXISFROMGROUP	Execute	●	●	●	●	●																				
	IdentInGroup																									
	AxesGroup																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_REMOVEAXISFROMGROUP	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		IdentInGroup	WORD	No	[0 to 2]	Axis sequence flag
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 10 Error Code Cross-reference Table】)

Features

▪This function block removes axis from AxesGroup with serial number marked by IdentInGroup, which are not cached and do not move.

✎**Note:**

▪If the serial number of the axis group does not exist, an error will be reported;

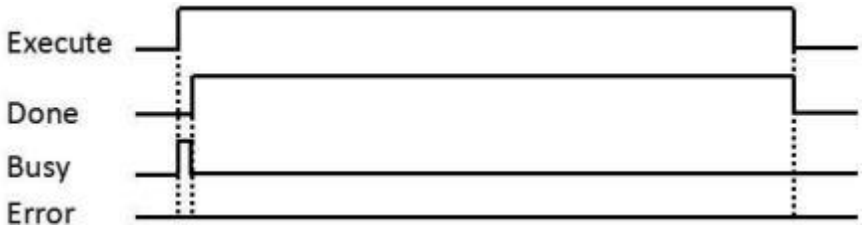
▪ Axis group is running or not disabled, and an error will be reported when removed the axis;

▪Single-axis operation can only be executed when the axis group is disabled, and whether it is removed or not will not affect the single-axis operation.

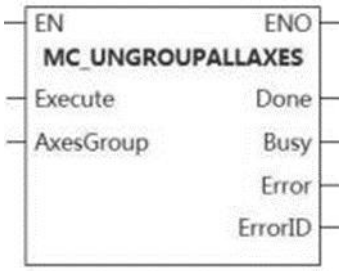
Error code

Error code	Content
4084H	Data beyond the specifiable range was input.

Sequence diagram



8.5.3 MC_UNGROUPALLAXES



This function block removes all the axes from an axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_UNGROUPALLAXES	Execute	●	●	●	●	●																				
	AxesGroup																									
	Done		●	●	●	●																				
	BusY		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_UNGROUPALLAXES	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		BusY	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 10 Error Code】)

Variable type used

Features

▪Delete all axes from AxesGroup. Not be buffered and not move.

✎Note:

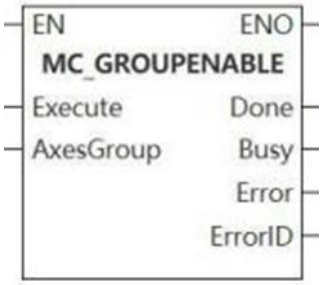
▪If the serial number of the axis group does not exist, an error will be reported;

▪ Axis group is running or not disabled, and an error will be reported when removed the axis;

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.5.4 MC_GROUPENABLE



The function block is axis group enabled.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_SAVECAM TABLE	Execute	●	●	●	●	●																				
	AxesGroup																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPENABLE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8.10 Error Code】)

Features

▪Enable axis group to run on the Execute rising edge.

✎Note:

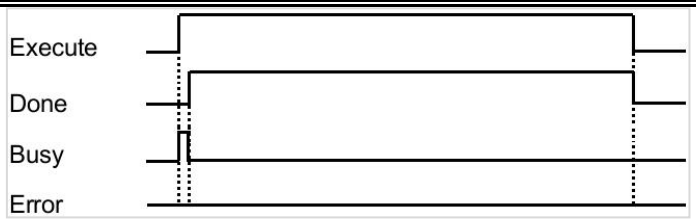
▪Axis group motion can only be running when enabled, otherwise an error will be reported;

▪When a single axis is running, enabling this block will report an error.

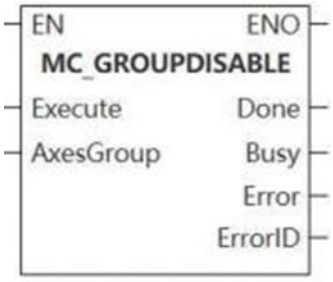
Error code

Error code	Content
4084H	Input data beyond the specifiable range

Sequence diagram


Execute
Done
Busy
Error

8.5.5 MC_GROUPDISABLE



This function block is axis group disabled.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GROUPDISABLE	Execute	●	●	●	●	●																				
	AxesGroup																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPDISABLE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 10 Error Code】)

Features

▪On the Execute rising edge, disable the axis group.

✎Note:

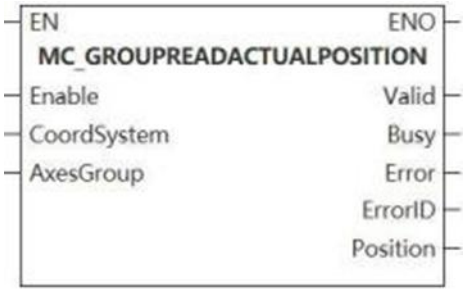
▪Single axis operation can only be executed when the group is disabled.

▪When a single axis is running, disabling this block will report an error.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.5.6 MC_GROUPREDACTUALPOSITION



This function block is to read the actual position of each axis from the axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																									Index modification	Pulse expansion
		X	Y	M	S	S M	T(bit)	C(bit)	LC(bit)	HSC(bit)	D. b	Kn X	Kn Y	Kn M	Kn S	T	C	D	R	SD	LC	HSC	K	H	E		[D]	XXP
MC_GROUPREADACTUALPOSITION	Enable	●	●	●	●	●																						
	CoordSystem																											
	AxesGroup																											
	Valid		●	●	●	●																						
	Busy		●	●	●	●																						
	Error		●	●	●	●																						
	ErrorID																											
	Position																											

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPREADACTUALPOSITION	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		CoordSystem	WORD	Yes	0	Coordinate system
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 11 Error Code】)
		Position	LREAL array	Yes		Position

Features

▪When Enable= true, the position of each axis of the selected axis group in the selected coordinate system is continuously returned.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.5.7 MC_GROUPREADACTUALVELOCITY



This function block is to read the actual velocity of each axis from the axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GROUPREADACTUALVELOCITY	Enable	●	●	●	●	●																					
	CoordSystem																										
	AxesGroup																										
	Valid		●	●	●	●																					
	Busy		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										
	Velocity																										
	PathVelocity																										

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPREADACTUALVELOCITY	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		CoordSyste	WORD	Yes	0	Coordinate system
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 11 Error Code】)
		Velocity	LREAL array	Yes		Velocity
		PathVelocit	LREAL	Yes		Path velocity

Features

▪When Enable=true, the velocity of each axis and the current TCP path velocity (the result of combined velocity) of the selected axis group in the selected coordinate system are continuously returned.

Error code

Error code	Content
4084H	Data beyond the specifiable range was input.

8.5.8 MC_GROUPREADERROR



The function block is axis group to read error.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GROUPREADERROR	Enable	●	●	●	●	●																				
	AxesGroup																									
	Valid		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									
	GroupErrorID																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPREADERROR	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 11Error Code】)
		GroupErrorID	DWROD	Yes		Axis group error code

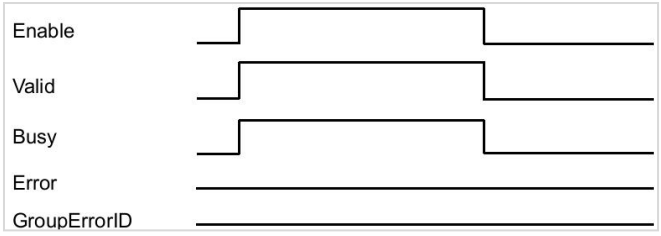
Features

When Enable= true, read the axis group error status.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

Timing diagram



8.5.9 MC_GROUPREADSTATUS



This function block reads the status of the currently activated axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GROUP READSTATUS	Enable	●	●	●	●	●																					
	AxesGroup																										
	Valid		●	●	●	●																					
	Busy		●	●	●	●																					
	GroupMoving		●	●	●	●																					
	GroupHoming		●	●	●	●																					
	GroupErrorStop		●	●	●	●																					
	GroupStandby		●	●	●	●																					
	GroupStopping		●	●	●	●																					
	GroupDisabled		●	●	●	●																					
	ConstantVelocity		●	●	●	●																					
	Accelerating		●	●	●	●																					
	Decelerating		●	●	●	●																					
	InPosition		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPREADERROR	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		AxesGroup	WORD	No	[0,65535]	Axis group number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Whether the output is valid
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		GroupMoving	BOOL	Yes	TRUE/FALSE	Axle group movement flag
		GroupHoming	BOOL	Yes	TRUE/FALSE	Axle group back to zero flag
		GroupErrorStop	BOOL	Yes	TRUE/FALSE	Axis group error stop flag
		GroupStandby	BOOL	Yes	TRUE/FALSE	Axis group static sign
		GroupStopping	BOOL	Yes	TRUE/FALSE	Axis group stop in flag
		GroupDisabled	BOOL	Yes	TRUE/FALSE	Axis group not activated flag
		ConstantVelocity	BOOL	Yes	TRUE/FALSE	Constant speed
		Accelerating	BOOL	Yes	TRUE/FALSE	Accelerate
		Decelerating	BOOL	Yes	TRUE/FALSE	Decelerate
		InPosition	BOOL	Yes	TRUE/FALSE	Reach the target position
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 11 Error Code Cross-Reference Table】)

Features

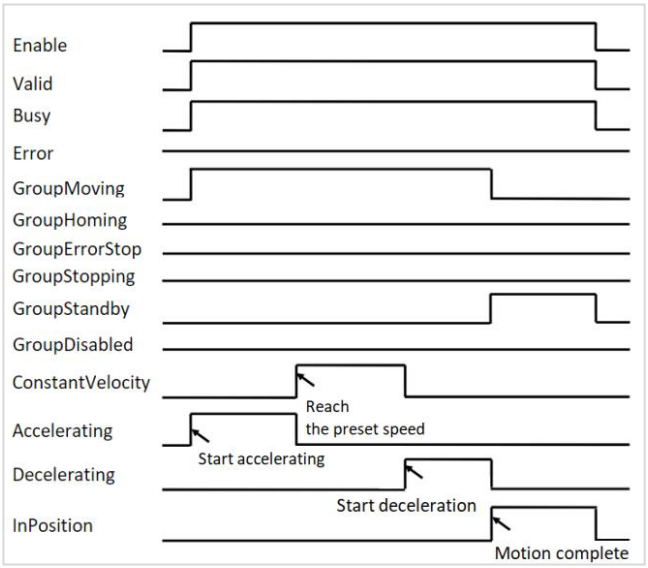
▪When Enable= true, the status of the enabled axis group is continuously returned, and its status is set and marked by the output flag;

▪ConstantVelocity is set to ON when moving at constant velocity on the instruction path.、Accelerating and decelerating is set to ON when the velocity increases or decreases on the instruction path.

▪ InPosition is set to ON when reach the target position.

Error code

Error code	Content
4084H	Input data beyond the specifiable range



8.5.10 MC_GROUPRESET



The function block is axis group reset.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GROUPRESET	Execute	●	●	●	●	●																				
	AxesGroup																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPRESET	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FAL	Whether there is an error or not

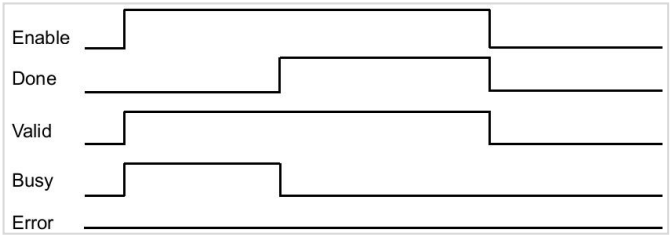
					SE	
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 10 Error Code Cross-reference Table】)

Features

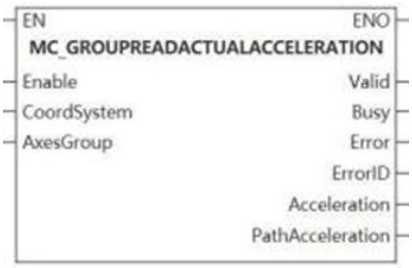
▪Reset the axis group from the error status to the enabled status when on the Execute rising edge, which is similar with the single-axis MC_Reset.

Error code

Error code	Content
4084H	Input data beyond the specifiable range



8.5.11 MC_GROUPREADACTUALACCELERATION



This function block is to read the actual acceleration of each axis from the axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GROUPREADACTUALACCELERATION	Enable	●	●	●	●	●																				
	CoordSystem																									
	AxesGroup																									
	Valid		●	●	●	●																				
	Busy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									
	Acceleration																									
	PathAcceleration																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

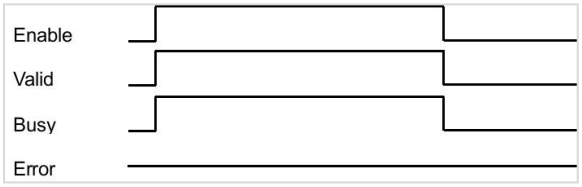
Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPREADACTUALACCELERATION	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		CoordSystem	WORD	Yes	0	Coordinate system
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Valid	BOOL	Yes	TRUE/FALSE	Whether the input is valid or not
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 11 Error Code】)
		Acceleration	LREAL array	Yes		Acceleration
		PathAcceleration	LREAL	Yes		Path acceleration

Features

▪When Enable=true, the acceleration of each axis and the current TCP path acceleration (The result of combined acceleration) of the selected axis group in the selected coordinate system are continuously returned.

Error code

Error code	Content
4084H	Data beyond the specifiable range was input.


Enable
Valid
Busy
Error

8.6 Axis group motion Function Blocks

8.6.1 MC_GROUPSTOP



The function block is use to stop axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GROUPSTOP	Execute	●	●	●	●	●																				
	Deceleration																									
	Jerk																									
	AxesGroup																									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPSTOP	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Deceleration	LREAL	No	[0 to 999999999.999999]	Deceleration
		Jerk	LREAL	Yes	[0 to 999999999.999999]	Jerk
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 11 Error Code】)

Features

▪When on the Execute rising edge, the axis group decelerate to stop and any functional block being executed is stopped. When Execute=true, the axis group is on the stop status and cannot execute other axis group motion instruction.

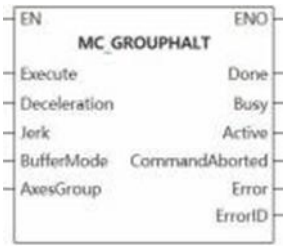
▪The velocity of axis group is zero. Done is set to ON. If Execute=false, return to the normal axis group enabling status;

▪Only MC_GroupDisable can be used to interrupt axis group.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.6.2 MC_GROUPHALT



The function block is to halt axis group.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_GROUPHALT	Execute	●	●	●	●	●																					
	Deceleration																										
	Jerk																										
	AxesGroup																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_GROUPHALT	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Deceleration	LREAL	No	[0 to 999999999. 999999]	Deceleration
		Jerk	LREAL	Yes	[0 to 999999999. 999999]	Jerk
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 10 Error Code】)

Features

▪When on the Execute rising edge, the axis group decelerate to stop, any function block being executed is stopped, and the axis group velocity is 0. Done is set to on;

▪Unbuffered mode: During axis group deceleration, if another function block with mcAborting buffer mode is executed, the current function block is interrupted and the new function block is executed immediately.

Error code

Error code	Content
4084H	Data beyond the specifiable range was input.

8.6.3 MC_MOVELINEARABSOLUTE



This function block is absolute position circular interpolation.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																									Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVELINEARABSOLUTE	Execute	●	●	●	●	●																						
	Position																											
	Velocity																											
	Acceleration																											
	Deceleration																											
	Jerk																											
	CoordSystem																											
	BufferMode																											
	TransitionMode																											
	TransitionParameter																											
	AxesGroup																											
	Done		●	●	●	●																						
	Busy		●	●	●	●																						
	Active		●	●	●	●																						
	CommandAborted		●	●	●	●																						
	Error		●	●	●	●																						
	ErrorID																											

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_MOVELINEARABSOLUTE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Position	LREAL array	No	-	Position
		Velocity	LREAL	No	[0 to 999999999.999999]	Velocity
		Acceleration	LREAL	No	[0 to 999999999.999999]	Acceleration
		Deceleration	LREAL	No	[0 to 999999999.999999]	Deceleration
		Jerk	LREAL	Yes	[0 to 999999999.999999]	Jerk is valid in S curve only.
		CoordSystem	WORD	Yes	0	Coordinate system
		BufferMode	WORD	Yes	[0 to 1]	Please check [8.9.5 Mode selection description] for more details.
		TransitionMode	WORD	Yes	[0 to 2]	Please check [8.9.4 Transition mode description] for more details.
		TransitionParameter	LREAL	Yes	[0 to 999999999.999999]	Please check [8.9.4 Transition mode description] for more details on transition parameter.
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 10 Error Code】)

Features

▪This function block executes linear interpolation motion, and puts the actual TCP position to the absolute position specified by a specific CoordSystem.

✎Note:

▪The axis group can be moved only when being enabled;

▪All velocities and accelerations are in a safe range, and subject to the maximum velocity and acceleration that can be reached (Affected by the planning velocity, the limited velocity and acceleration set in the FB). If all the velocities and accelerations exceed the limit, they should be reduced to the limit.

▪Linear motion is allowed to be interrupted;

▪Jerk is valid in S-curve only.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.6.4 MC_MOVELINEARRELATIVE



This function block is relative position circular interpolation.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Pin type	Parameter	Devices																								Index modification	Pulse expansion
			X	Y	M	S	S M	T(bit)	C(bit)	LC(bit)	HSC(bi t)	D. b	Kn X	Kn Y	Kn M	Kn S	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVELINEARRELATIVE	Input	Execute	●	●	●	●	●																					
		Distance																										
		Velocity																										
		Acceleration																										
		Deceleration																										
		Jerk																										
		CoordSystem																										
		BufferMode																										
		TransitionMode																										
		TransitionParamet er																										
		AxesGroup																										
	Output	Done	●	●	●	●																						
		Busy	●	●	●	●																						
		Active	●	●	●	●																						
		CommandAborte d	●	●	●	●																						
		Error	●	●	●	●																						
		ErrorID																										

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_MOVELINEARRELATIVE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Distance	LREAL array	No	-	Position
		Velocity	LREAL	No	[0 to 999999999.999999]	Velocity
		Acceleration	LREAL	No	[0 to 999999999.999999]	Acceleration
		Deceleration	LREAL	No	[0 to 999999999.999999]	Deceleration
		Jerk	LREAL	Yes	[0 to 999999999.999999]	Jerk: Only the axis group S-curve is valid.
		CoordSystem	WORD	Yes	-	Coordinate system
		BufferMode	WORD	Yes	[0 to 1]	Please check [8.9.5 Mode selection description] for more details.
		TransitionMode	WORD	Yes	[0 to 2]	Please check [8.9.4 Transition mode description] for more details.
		TransitionParameter	LREAL	Yes	[0 to 999999999.999999]	Please check [8.9.4 Transition mode description] for more details on transition parameter.
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check【8. 10 Error Code Cross-reference Table】)

Features

▪This function block executes line interpolation motion, and puts the actual TCP position to the absolute position specified by a specific coordsystem.

✎Note:

▪The axis group can be moved only when being enabled;

▪All velocities and accelerations are in a safe range, and subject to the maximum velocity and acceleration that can be reached (Affected by the planning velocity, the limited velocity and acceleration set in the FB). If all the velocities and accelerations exceed the limit, they should be reduced to the limit.

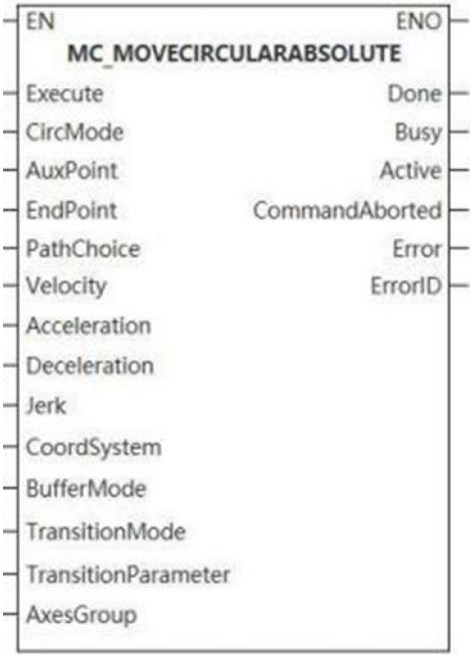
▪Linear motion is allowed to be interrupted;

▪Jerk is valid in S-curve only.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.6.5 MC_MOVECIRCULARABSOLUTE



The function block is absolute position circular interpolation.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																									Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVECIRCULARABSOLUTE	Execute	●	●	●	●	●																						
	CircMode																											
	AuxPoint																											
	EndPoint																											
	PathChoice																											
	Velocity																											
	Acceleration																											
	Deceleration																											
	Jerk																											
	CoordSystem																											
	BufferMode																											
	TransitionMode																											
	TransitionParamet																											
	AxesGroup																											
	Done		●	●	●	●																						
	Busy		●	●	●	●																						
	Active		●	●	●	●																						
	CommandAborted		●	●	●	●																						
	Error		●	●	●	●																						
	ErrorID																											

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_MOVECIRCULARABSOLUTE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		CircMode	WORD	No	[0 to 2]	Please check [8.9.3 Arc mode description] for more details on arc construction mode.
		AuxPoint	LREAL array	No	-	Please check [8.9.3 Arc mode description] for more details on auxiliary point.
		EndPoint	LREAL array	No	-	Please check [8.9.3 Arc mode description] for more details on end point.
		PathChoice	WORD	No	[0 to 1]	Please check [8.9.3 Arc mode description] for more details on arc path selection.
		Velocity	LREAL	No	[0 to 999999999.999999]	Velocity
		Acceleration	LREAL	No	[0 to 999999999.999999]	Acceleration
		Deceleration	LREAL	No	[0 to 999999999.999999]	Deceleration
		Jerk	LREAL	Yes	[0 to 999999999.999999]	Jerk is valid in S-curve only.
		CoordSystem	WORD	Yes	0	Coordinate system
		BufferMode	WORD	Yes	[0 to 1]	Please check [8.9.5 Mode selection description] for more details.
		TransitionMode	WORD	Yes	[0 to 2]	Please check [8.9.4 Transition mode description] for more details.
		TransitionParameter	LREAL	Yes	[0 to 999999999.999999]	Please check [8.9.4 Transition mode description] for more details on transition parameter.
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check [8. 10 Error Code Cross-reference Table])

Features

▪This function block executes interpolation according to the circular arc set by AuxPoint, EndPoint and CircMode. Axis Group moves to the absolute position EndPoint according to the circular arc. Arc motion direction is determined by PathChoice.

▪Check details for transition mode. Transition parameters are additional parameters of transition modes, which take effect according to different transition modes.

▪Set by coordsystem, Auxpoint is the position where function block is activated and Endpoint is the position which is in the coordinate system.

✎Note:

▪The axis group can be moved only when being enabled;

▪All velocities and accelerations are in a safe range, and subject to the maximum velocity and acceleration that can be reached (Affected by the planning velocity, the limited velocity and acceleration set in the FB). If all the velocities and accelerations exceed the limit, they should be reduced to the limit.

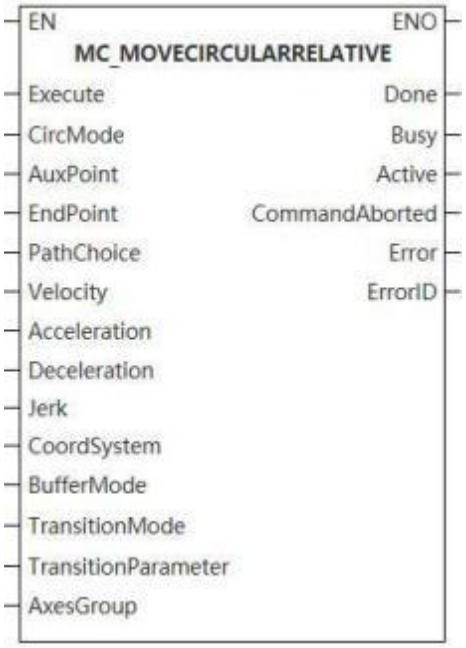
▪Arc motion is allowed to be interrupted;

▪Jerk is valid in S-curve only.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.6.6 MC_MOVECIRCULARRELATIVE



Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVECIRCULARRELATIVE	Execute	●	●	●	●	●																					
	CircMode																										
	AuxPoint																										
	EndPoint																										
	PathChoice																										
	Velocity																										
	Acceleration																										
	Deceleration																										
	Jerk																										
	CoordSystem																										
	BufferMode																										
	TransitionMode																										
	TransitionParamet																										
	AxesGroup																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_MOVELINEARRELATIVE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		CircMode	WORD	No	[0 to 2]	Please check [8.9.3 Arc mode description] for more details on arc construction mode.
		AuxPoint	LREAL array	No	-	Please check [8.9.3 Arc mode description] for more details on auxiliary point.
		EndPoint	LREAL array	No	-	Please check [8.9.3 Arc mode description] for more details on end point.
		PathChoice	WORD	No	[0 to 1]	Please check [8.9.3 Arc mode description] for more details on arc path selection.
		Velocity	LREAL	No	[0 to 999999999.999999]	Velocity
		Acceleration	LREAL	No	[0 to 999999999.999999]	Acceleration
		Deceleration	LREAL	No	[0 to 999999999.999999]	Deceleration
		Jerk	LREAL	Yes	[0 to 999999999.999999]	Jerk:is valid in S-curve only.
		CoordSystem	WORD	Yes	-	Coordinate system
		BufferMode	WORD	Yes	[0 to 1]	Please check [8.9.5 Mode selection description] for more details.
		TransitionMode	WORD	Yes	[0 to 2]	Please check [8.9.4 Transition mode description] for more details.
		TransitionParameter	LREAL	Yes	[0 to 999999999.999999]	Please check [8.9.4 Transition mode description] for more details on transition parameter.
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check [8. 10 Error Code Cross-reference Table])

Features

▪This function block performs circular interpolation according to the circular arc specified d by AuxPoint, EndPoint and CircMode. Axis group moves to the relative position Endpoint according to the circular arc. Arc motion direction is determined by PathChoice.

▪Check details for transition mode. Transition parameters are additional parameters of transition modes, which take effect according to different transition modes.

▪Set by coordsystem, Auxpoint is the position where function block is activated and Endpoint is the position which is in the coordinate system.

✎Note:

▪The axis group can be moved only when being enabled;

▪All velocities and accelerations are in a safe range, and subject to the maximum velocity and acceleration that can be reached (Affected by the planning velocity, the limited velocity and acceleration set in the FB). If all the velocities and accelerations exceed the limit, they should be reduced to the limit.

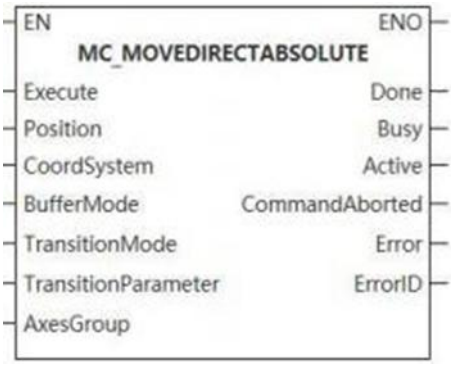
▪Arc motion is allowed to be interrupted;

▪Jerk is valid in the S-curve only.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.6.7 MC_MOVEDIRECTABSOLUTE



This function block is the fast positioning of absolute position.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVEDIRECTABSOLUTE	Execute	●	●	●	●	●																					
	Position																										
	CoordSystem																										
	BufferMode																										
	TransitionMode																										
	TransitionParameter																										
	AxesGroup																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_MOVEDIRECTABSOLUTE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Position	LREAL array	No	-	Please check [8.9.3 Arc mode description] for more details on arc construction mode.
		CoordSystem	WORD	Yes	0	Coordinate system
		BufferMode	WORD	Yes	[0 to 1]	Please check [8.9.5 Mode selection description] for more details.
		TransitionMode	WORD	Yes	[0 to 2]	Please check [8.9.4 Transition mode description] for more details.
		TransitionParameter	LREAL	Yes	[0 to 999999999.999999]	Please check [8.9.4 Transition mode description] for more details on transition parameter
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check [8. 10 Error Code])

Features

▪ The function block puts the axis group to the absolute position specified in the specific coordinate system, and does not emphasize on how to reach the target position;

▪The performance of each axis in the axis group, such as velocity, acceleration, deceleration, jerk, does not change when moving.

✎Note:

▪The axis group can be moved only when being enabled;

▪All velocities and accelerations are in a safe range, and subject to the maximum velocity and acceleration that can be reached (Affected by the planning velocity, the limited velocity and acceleration set in the FB). If all the velocities and accelerations exceed the limit, they should be reduced to the limit.

▪Linear motion is allowed to be interrupted;

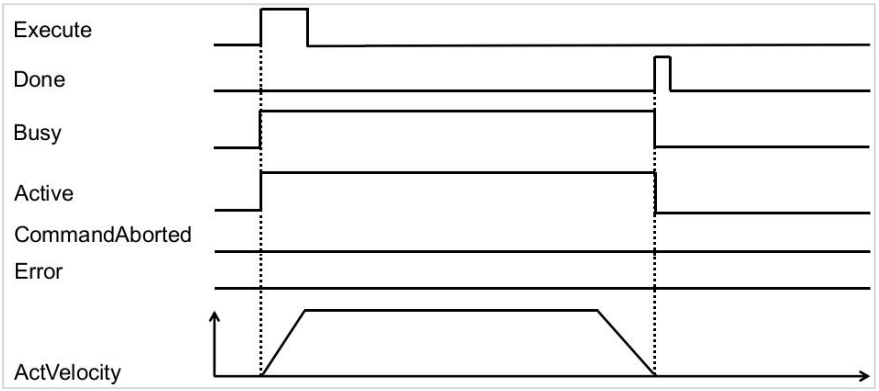
▪The instruction running velocity and acceleration are associated with the minimum value of each axis associated with the axis group;

▪Jerk is valid in S-curve only.

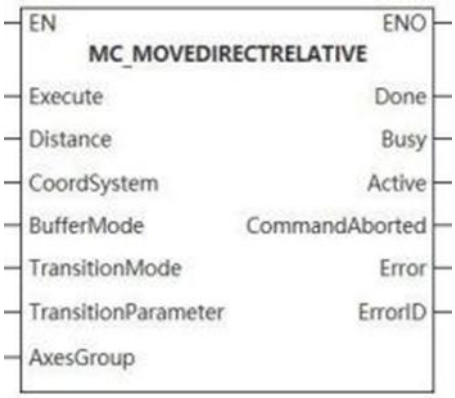
Error code

Error code	Content
4084H	Data beyond the specifiable range was input.

Sequence diagram



8.6.8 MC_MOVEDIRECTRELATIVE



The function block executes the fast positioning of the relative position.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVEDIRECTRELATIVE	Execute	●	●	●	●	●																					
	Distance																										
	CoordSystem																										
	BufferMode																										
	TransitionMode																										
	TransitionParamet																										
	AxesGroup																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_MOVEDIRECTRELATIVE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Distance	LREAL array	No	-	Please check [8.9.3 Arc mode description] for more details on arc construction
		CoordSystem	WORD	Yes	0	Coordinate system
		BufferMode	WORD	Yes	[0 to1]	Please check [8.9.5 Mode selection description] for more details.
		TransitionMode	WORD	Yes	[0 to 2]	Please check [8.9.4 Transition mode description] for more details.
		TransitionParameter	LREAL	Yes	[0 to 999999999.999999]	Please check [8.9.4 Transition mode description] for more details on transition
		AxesGroup	WORD	No	[0 to 65535]	Axis group number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check [8. 10 Error Code])

Features

▪ The function block puts the axis group to the relative position specified in the specific coordinate system, and does not emphasize on how to reach the target position;

▪The performance of each axis in the axis group, such as velocity, acceleration, deceleration, jerk, does not change when moving.

✎Note:

▪The axis group can be moved only when being enabled;

▪All velocities and accelerations are in a safe range, and subject to the maximum velocity and acceleration that can be reached (Affected by the planning velocity, the limited velocity and acceleration set in the FB). If all the velocities and accelerations exceed the limit, they should be reduced to the limit.

▪Linear motion is allowed to be interrupted;

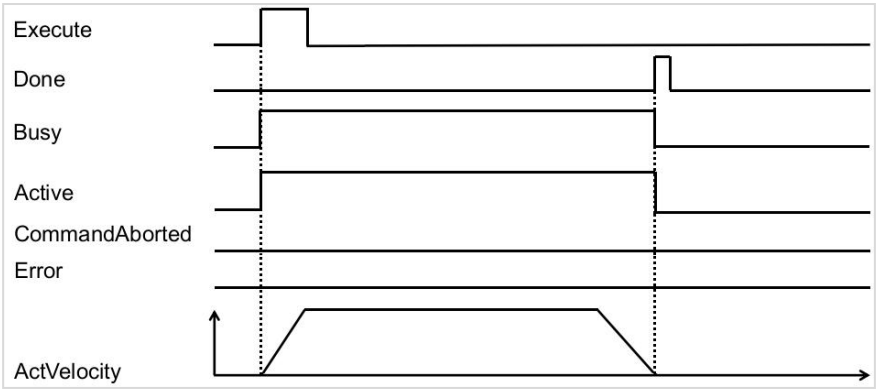
▪The instruction running velocity and acceleration are associated with the minimum value of each axis associated with the axis group;

▪Jerk is valid in S-curve only.

Error code

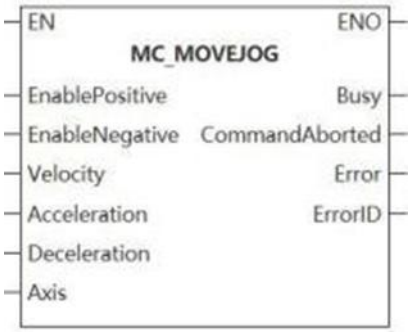
Error code	Content
4084H	Input data beyond the specifiable range

Sequence diagram



8.7 Other special FB

8.7.1 MC_MOVEJOG



The function block makes the controlled axis execute micro motion at the set target velocity.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																									Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVEJOG	EnablePositive	●	●	●	●	●																						
	EnableNegative	●	●	●	●	●																						
	Velocity																											
	Acceleration																											
	Deceleration																											
	Axis																											
	Busy		●	●	●	●																						
	CommandAborted		●	●	●	●																						
	Error		●	●	●	●																						
	ErrorID																											

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_MOVEJOG	Input	EnablePositive	BOOL	No	TRUE/FALSE	Positive motion
		EnableNegative	BOOL	No	TRUE/FALSE	Negative motion
		Velocity	LREAL	No	[0 to 999999999.	Velocity
		Acceleration	LREAL	No	[0 to 999999999.	Acceleration
		Deceleration	LREAL	No	[0 to 999999999.	Deceleration
		Axis	WORD,	No	[0 to 65535]	Axis number
	Output	Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		CommandAbort	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check [8. 10 Error Code])

Features

▪When input Enablepositive signal on the rising edge, the controlled axis executes positive motion according to the input velocity.

▪When input EnableNegative signal on the rising edge, the controlled axis executes negative motion according to the input velocity.

▪In the process of positive motion, modifying the Enablepositive to FALSE causes deceleration until the axis stops. In the process of deceleration, if Enablepositive is changed to TRUE, the axis will continue to move until reaching the specified velocity. In the process

of deceleration, if EnableNegative is changed to TRUE, the axis will decrease its velocity to 0 and then execute negative motion until its velocity reaches the specified velocity.

✎Note:

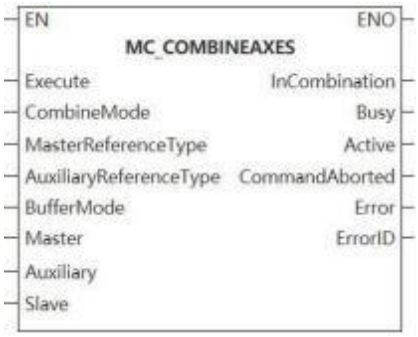
▪This function is not available when the axis does not exist.

▪When EnablePositive or EnableNegative is TRUE, modifying velocity does not change the velocity at this time.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.7.2 MC_COMBINEAXES



The function block adds and subtracts the positions of the master axis and the auxiliary axis as the position of the slave axis.

Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_COMBINEAXES	Execute	●	●	●	●	●																					
	ConbineMode																										
	MasterReferenceType																										
	AuxiliaryReferenceType																										
	BufferMode																										
	Master																										
	Auxiliary																										
	Slave																										
	InCombiantion		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_COMBINEAXES	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ConbineMode	WORD	Yes	[0 to 1]	Addition and subtraction operation selection 0: Addition 1: Subtraction
		MasterReferenceType	WORD	Yes	[1 to 2]	Master axis position type 1: Feedback position 2: Instruction position
		AuxiliaryReferenceType	WORD	Yes	[1 to 2]	Auxiliary axis position type 1: Feedback position 2: Instruction position
		BufferMode	LREAL	No	0	Mode selection
		Master	WORD, AXIS INFO T	No	[0 to 65535]	Master axis number
		Auxiliary	WORD, AXIS INFO T	No	[0 to 65535]	Auxiliary axis number
		Slave	WORD, AXIS INFO T	No	[0 to 65535]	Slave axis number
	Output	InCombiantion	BOOL	Yes	TRUE/FALSE	Addition and subtraction
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check [8. 10 Error Code])

Features

▪On the Execute rising edge, the addition and subtraction positioning operation is performed.

✎Note:

▪Unable to restart this instruction;

▪To end this instruction, MC_construction is needed;

▪This instruction does not execute in-place inspection;

▪When using the instruction position, the set values of master axis number and auxiliary axis number must be less than the slave axis number;

▪This instruction is not affected by the abnormal status of the main axis and auxiliary axis. When the abnormal status is released, the slave axis continues to execute the addition and subtraction positioning operation. In addition, when this instruction is enabled and the axis is abnormal, this instruction is interrupted, but it has no effect on the master axis and the slave axis;

▪When MC_SETPOSITION instruction is executed for the master axis or the auxiliary axis, it may happen that the slave axis catches at a large velocity, which is very dangerous.

Error code

Error code	Content
7000OA	Axis is in a bound status.
7000OB	Axis cannot be bound at run time.
800014	The master axis is consistent with the slave axis.

Error code	Content
800015	The master axis is consistent with the auxiliary axis.
800016	The auxiliary axis is consistent with the slave axis.
800017	Master axis (auxiliary axis) and slave axis numbers are not ascending order.

8.7.3 MC_MOVEVELOCITY



Device used (Only a single device can be used, and device splicing and index are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_MOVEVELOCITY	Execute	●	●	●	●	●																				
	ContinuousUpdate	●	●	●	●	●																				
	Velocity																									
	Acceleration																									
	Deceleration																									
	Jerk																									
	Direction																									
	BufferMode																									
	Axis																									
	InVelocity		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Precautions for use of devices

▪LC and HSC are signed 32-bit data types;

▪T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC_MOVEVELOCITY	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	No	TRUE/FALSE	Whether update continuously or not
		Velocity	LREAL	No	[0 to 999999999.999999]	The set velocity
		Acceleration	LREAL	No	[0 to 999999999.999999]	The set acceleration
		Deceleration	LREAL	No	[0 to 999999999.999999]	The set deceleration
		Jerk	LREAL	Yes	[0 to 999999999.999999]	The set jerk is only valid in the S-curve.
		Direction	WORD	No	[1]	Direction flag 1: Positive direction 2: Shortest path 3: Negative direction 4: Current direction
		BufferMode	WORD	Yes	[0]	Buffer mode
		Axis	WORD, AXIS INFO T	No	[0 to 65535]	Axis number and axis in-built variable
	Output	InVelocity	BOOL	Yes	TRUE/FALSE	Reach the set velocity flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please check [8. 10 Error Code])

Features

▪MC_MOVEVELOCITY function block is used for continuing the motion of the previous function block and moving according to the velocity specified by the previous function block. The input values of acceleration, deceleration and jerk are used to monitor whether the corresponding values of the controlled axis are within the input values.

▪Since there is no input signal of position or distance, the function block will move until other function block sends a new instruction to stop the operation of this function block;

▪When the velocity of the controlled axis reaches the set value of input parameter Velocity, the output parameter Invelocity is set to ON.

▪Once the input Execute signal detects that the falling edge or other function block makes this function block abort, the output parameter Invelocity is reset.

▪MC_MOVEVELOCITY function block should be used when the controlled axis is in a stop status or a single-axis running status. When the function block is triggered, the controlled axis is switched to continuous Motion status.

▪If the controlled axis is not in the above status, the function block is not executed. Erroroutput is set;

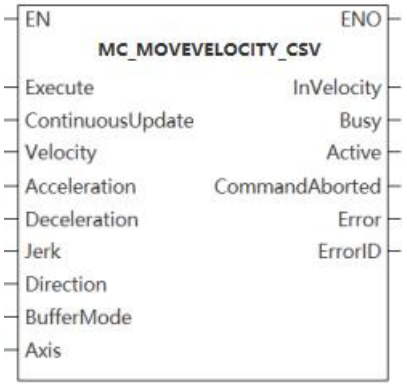
▪MC_MOVEVELOCITY function block can be triggered repeatedly.

Error code

Error code	Content
4084H	Input data beyond the specifiable range

8.7.4 MC _ MOVEVELOCITY _ CSV

MC _ MOVEVELOCITY _ CSV-Sets motion according to the speed specified by the function block



The function block is used to continue the motion of the previous function block and move according to the speed specified by the function block.

Device used (only a single device can be used, device splicing and index are not supported)

Instruction	Parameter	Device																							Index Modification	Pulse Expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC _ MOVEVELOCITY _ CSV	Execute	●	●	●	●	●																				
	ContinuousUpdate	●	●	●	●	●																				
	Velocity																									
	Acceleration																									
	Deceleration																									
	Jerk																									
	Direction																									
	BufferMode																									
	Axis																									
	InVelocity		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	Error ID																									

✎Notes:

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can it be empty	Range	Description
MC _ MOVE VELOCITY _ CSV	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	Yes	TRUE/FALSE	Whether update continuously or not
		Velocity	LREAL	No	[0,999999999.999999]	Setting velocity
		Acceleration	LREAL	No	[0,999999999.999999]	Setting acceleration
		Deceleration	LREAL	No	[0,999999999.999999]	Set deceleration
		Jerk	LREAL	Yes	[0,999999999.999999]	The set jerk is only effective in S-curve.
		Direction	WORD	No	[1,4]	Direction flag bit 1. Positive direction; 2. Negative direction; 3. The current direction; 4. The shortest path (reserved).
		BufferMode	WORD	Yes	[0]	BufferMode
		Axis	WORD	No	[0,65535]	Axis number and built-in variable of axis
	Output	InVelocity	BOOL	Yes	TRUE/FALSE	Reach the set velocity flag bit
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Activity flag
		Command aborted	BOOL	Yes	TRUE/FALSE	Interrupt flag bit
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		Error ID	DWORD	Yes	-	Error code (Please refer to [8.11 Error code comparison table])

Function

▪ The MC_ MOVEVELOCITY_CSV function block is used to continue the movement of the previous function block and move according to the speed specified by the function block. The input values of acceleration, deceleration and jerk are used to monitor whether the corresponding values of the controlled axis are within the input values.

▪ Since there is no input signal of position or distance, the function block will move until other function block sends a new command to stop movement of this function block;

▪ When the speed of the controlled axis reaches the input parameter 'Velocity' setpoint, the output parameter 'InVelocity' is set. Once the input 'Execute' signal detects a falling edge or other function block to abort this function block, the output parameter InVelocity is reset;

▪ The MC_ MOVEVELOCITY_CSV function block shall be in a stopped motion state or a single-axis operation state when the controlled axis is operated. After this function block is triggered, the controlled axis transitions to the 'Continuous Motion' state. If the controlled axis is not in the above state, the function block is not executed, Error output is set;

▪ The MC_ MOVEVELOCITY_CSV function block can be triggered repeatedly multiple times.

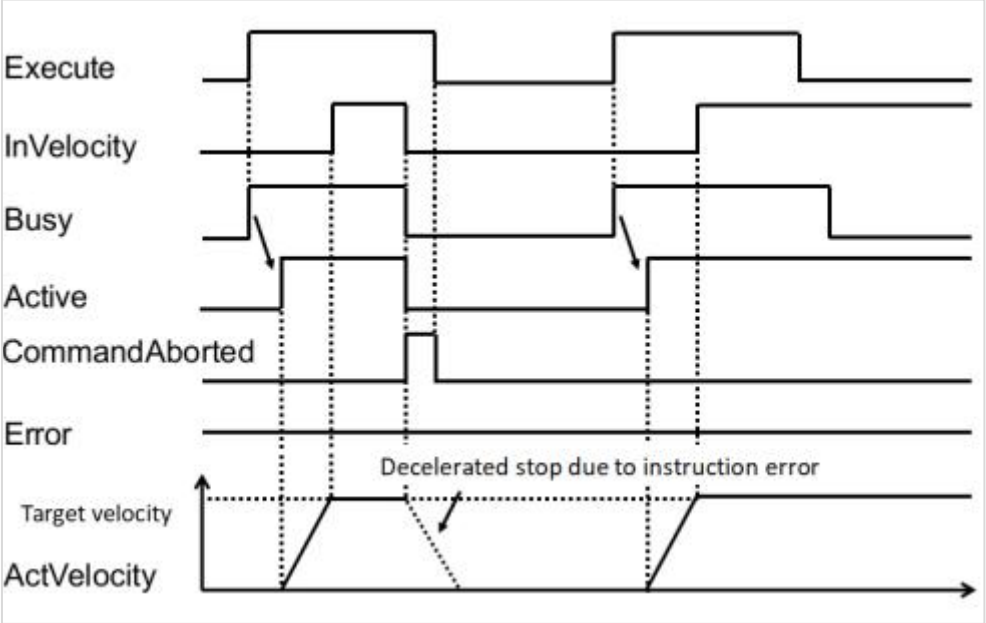
Error code

Error code	Content
4084H	Input data beyond the specifiable range

Precautions

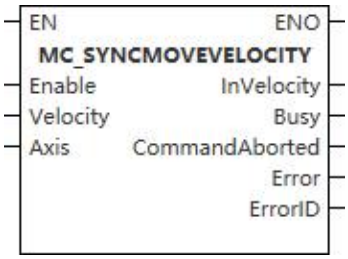
- Different from the MC _ MOVEVELOCITY function block, this function block will automatically switch the servo mode to CSV (Synchronous Cycle Speed) mode;
- Using this function block requires adding four object dictionaries, H60FF, H6060, H6061, and H606C, to the PDO.

Sequence diagram



8.7.5 MC _ SYNCMOVEVELOCITY

MC _ SYNCMOVEVELOCITY-Sets motion according to the speed specified by the function block



The function block is used to continue the motion of the previous function block and move according to the speed specified by the function block.

Device used (only a single device can be used, and device splicing and Index are not supported)

Instruction	Parameter	Device																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC _ SYNCMOVE VELOCITY	Execute	●	●	●	●	●																				
	Velocity																									
	Axis																									
	InVelocity		●	●	●	●																				
	Busy		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Notes for using devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin type	Parameter	Variable type	Empty or not	Range	Description
MC _ SYNCMOVE VELOCITY	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Velocity	LREAL	No	[-999999999.999999,999999999.999999]	Setting velocity
		Axis	WORD	No	[0,65535]	Axis number and built-in variable of axis
	Output	InVelocity	BOOL	Yes	TRUE/FALSE	Reach the set velocity flag bit
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Whether there is an error or not
		ErrorID	DWORD	Yes	-	Error code (Please refer to 8.11 Error Code

Function

- The MC_SYNCMOVEVELOCITY function block is used to continue the motion of the previous function block and move at the speed specified by this function block;

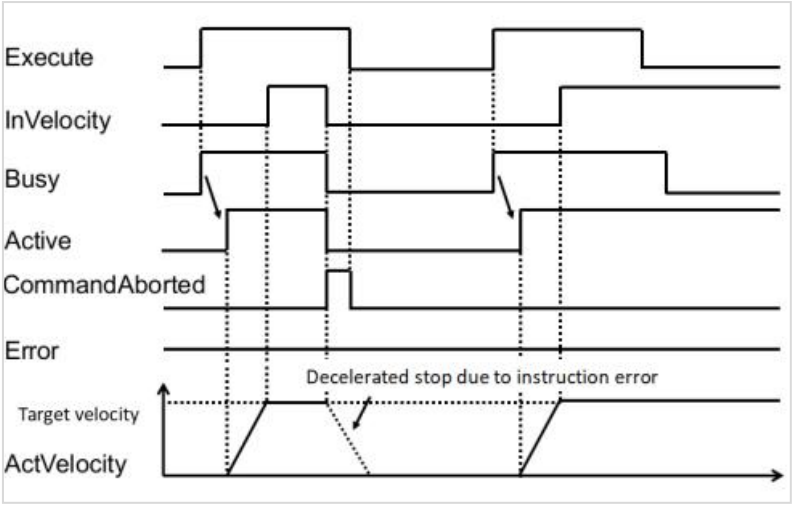
- Since there is no input signal for position or distance, this function block will keep moving until a new command from another function block stops it;
- When the controlled axis reaches the speed set by the input parameter Velocity, the output parameter InVelocity is set. Once the input Execute signal detects a falling edge or the controlled axis stops due to other function blocks, the output parameter InVelocity is reset;
- The MC_SYNCMOVEVELOCITY function block should be triggered when the controlled axis is stationary or in single - axis running state. After this function block is triggered, the controlled axis switches to the continuous motion (Continuus Motion) mode. If the controlled axis is not in the above mentioned state, this function block will not be executed, and its Error output is set;
- The MC_SYNCMOVEVELOCITY function block can be triggered multiple times repeatedly.
-

✎ **Note:** After using MC_SYNCMOVEVELOCITY, the servo mode will be automatically switched to the synchronous cycle velocity mode.

Error Code

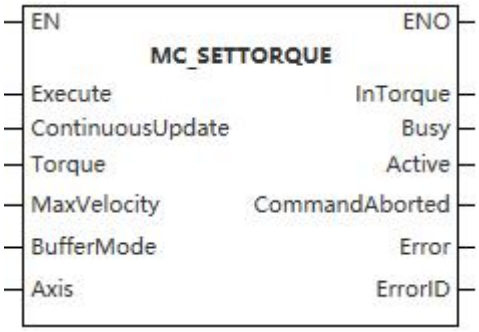
Error code	Content
4084H	Input data beyond the specifiable range

Timing diagram



8.7.6 MC _ SETTORQUE

MC _ SETTORQUE-Synchronous torque instruction



Device used (only a single device can be used, and device splicing and Index are not supported)

Instruction	Parameter	Device																							Index Modification	Pulse Expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC _ SETTORQUE	Execute	●	●	●	●	●																				
	TORQUE																									
	MaxVelocity																									
	BufferMode																									
	Axis																									
	InTorque		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Notes for using devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin Type	Parameter	Variable type	Empty or not	Range	Description
MC _ SETTORQUE	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	Yes	TRUE/FALSE	Updated during operation
		Torque	LREAL	No	[-32768,32767]	Target torque (unit is 0.1%)
		MaxVelocity	LREAL	No	[0,999999999.999999]	Maximum speed
		BufferMode	WORD	Yes	[0]	Currently only interrupts are supported
		Axis	WORD	No	[0,65535]	Axis number
	Output	InTorque	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Error or not
		ErrorID	DWORD	Yes	-	Error code (Please refer to [8.11 Error code comparison table])

Function

- This instruction is used to implement the synchronous torque control function and is only used for the bus servo axis. The rising edge and high level of the command are valid, and the virtual axis mode is not supported.
- The following PDOs need to be configured in CST mode;

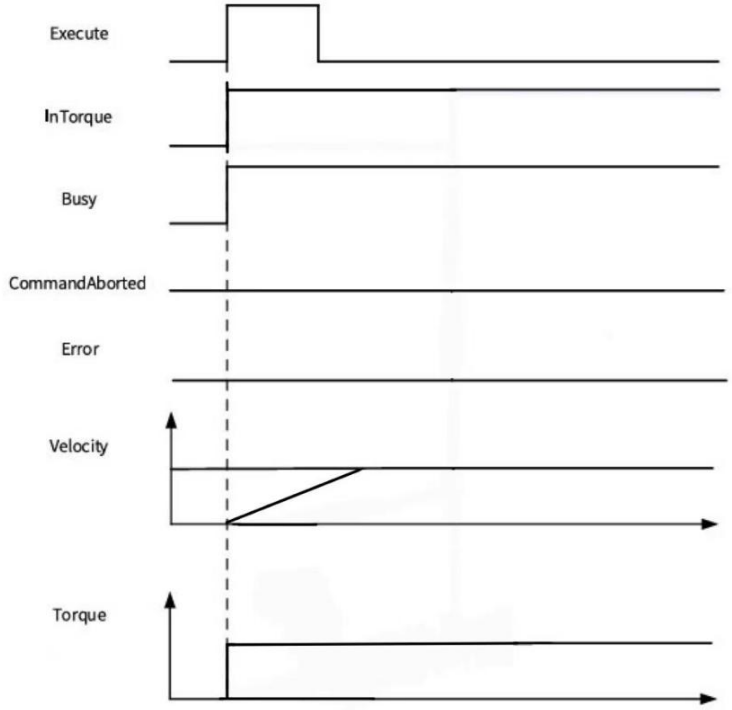
  0x1600: 0x6060, 0x6071, 607f (optional), 6040

  0x1A00: 0x6061, 0x6077, 0x6041
- If you use VD3E, you need to configure the startup parameter 2001: 9 value to 2;
- When Enable = ON, the function block converts TarTorque and MaxVelocity from user units to pulse units and sends them to the servo driver from time to time. The shaft is in the Continuous Motion state and performs synchronous torque motion. When the falling edge of Eanble, the function block performs synchronous torque motion with the TarTorque and Velocity latched during the high level, and it is invalid to modify the TarTorque and Velocity in real time;
- This instruction can only be invoked if the axis is switched to the enabled state using the MC _ Power instruction;
- This instruction uses the driver synchronous torque mode to realize the torque control function. When Enable = ON, the function block converts TarTorque and Velocity from user units to pulse units and sends them to the servo driver from time to time. The shaft is in the Continuous Motion state and performs synchronous torque motion;

  TarTorque: The target torque, the unit is 0.1%. One decimal place after the decimal point is valid in the program, and the latter ones are directly discarded. The actual torque of the driver is limited by the maximum positive and negative torques set in the configuration parameters.

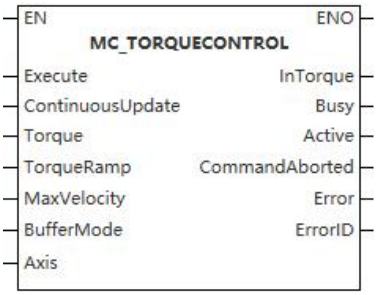
✎Note

- This function block is not available when the axis does not exist.
- Sequence diagram


Execute
InTorque
Busy
CommandAborted
Error
Velocity
Torque

8.7.7 MC _ TORQUECONTROL

MC _ TORQUECONTROL-Torque control command



Device used (only a single device can be used, and device splicing and Index are not supported)

Instruction	Parameter	Device																							Index Modification	Pulse Expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC _ TORQUE CONTROL	Execute	●	●	●	●	●																				
	ContinuousUpdate	●	●	●	●	●																				
	TORQUE																									
	TorqueRamp																									
	MaxVelocity																									
	BufferMode																									
	Axis																									
	InTorque		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Notes for using devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin Type	Parameter	Variable type	Empty or not	Range	Description
MC _ TORQUECONTROL	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		ContinuousUpdate	BOOL	Yes	TRUE/FALSE	Updated during operation
		Torque	LREAL	No	[-32768,32767]	Target torque (unit is 0.1%)
		MaxVelocity	LREAL	No	[0,999999999.999999]	Maximum speed
		TorqueRamp	LREAL	No	[0,999999999.999999]	Torque slope
		BufferMode	WORD	Yes	[0]	Only interrupt mode supported
		Axis	WORD	No	[0,65535]	Axis number
	Output	InTorque	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag

Instruction	Pin Type	Parameter	Variable type	Empty or not	Range	Description
		Error	BOOL	Yes	TRUE/FALSE	Error or not
		ErrorID	DWORD	Yes	-	Error code (Please refer to [8.11 Error code comparison table])

Function

- This instruction is used to realize the torque control function and is only used for the EtherCAT bus axis. The rising edge of the instruction is valid, and the imaginary axis mode is not supported.
- The following PDOs need to be configured in CST mode:

  0x1600: 0x6060, 0x6071, 607f (optional), 6040

  0x1A00: 0x6061, 0x6077, 0x6041
- If VD3E is used, the value of startup parameter 2001:9 is 2;

Function description

- This instruction can only be invoked if the axis is switched to the enabled state using the MC _ Power instruction;
- This instruction uses the driver synchronous torque mode to realize the torque control function;
- On the rising edge of the instruction, the function block latches the input parameters of TarTorque, TorqueSlope and MaxVelocity, and the axis is in the ContinuousMotion state and performs torque motion;

  TarTorque: Target torque, the unit is 0.1%, one decimal place after the decimal point is valid in the program, and the latter ones are discarded. The actual torque of the driver is limited by the maximum positive and negative torques set in the configuration parameters;

  TorqueSlope: Moment slope, the unit is 1%, one decimal place after the decimal point is valid in the program, and the latter ones are directly discarded;
- Speed control in torque mode

  For servo drives, if 0x607f is mapped in the axis parameter, this instruction limits the maximum speed of the servo motor by 0x607f, if 0x607f is not mapped, the speed limit is invalid;

  On the rising edge of Execute, the instruction converts the speed limit set by Velocity into pulse units and writes it into 0x607f through PDO;

  When the torque instruction is interrupted by others, the maximum speed of the axis will be limited by the "Maximum speed" in the configuration interface;

  For third-party drives, Velocity can only be used as a speed limit under the following conditions:

  The maximum speed of the servomotor can be limited by 0x607F;

  0x607F can be configured into the PDO;

  The unit of 0x607F is pulse unit, non-speed unit.
- Stop control on torque mode

  In the torque mode, the MC _ Stop instruction is called to perform the shut down, and after receiving the Stop instruction, the driver switches to the synchronous position mode, and performs deceleration according to the deceleration set by the Stop instruction.
- Interrupt

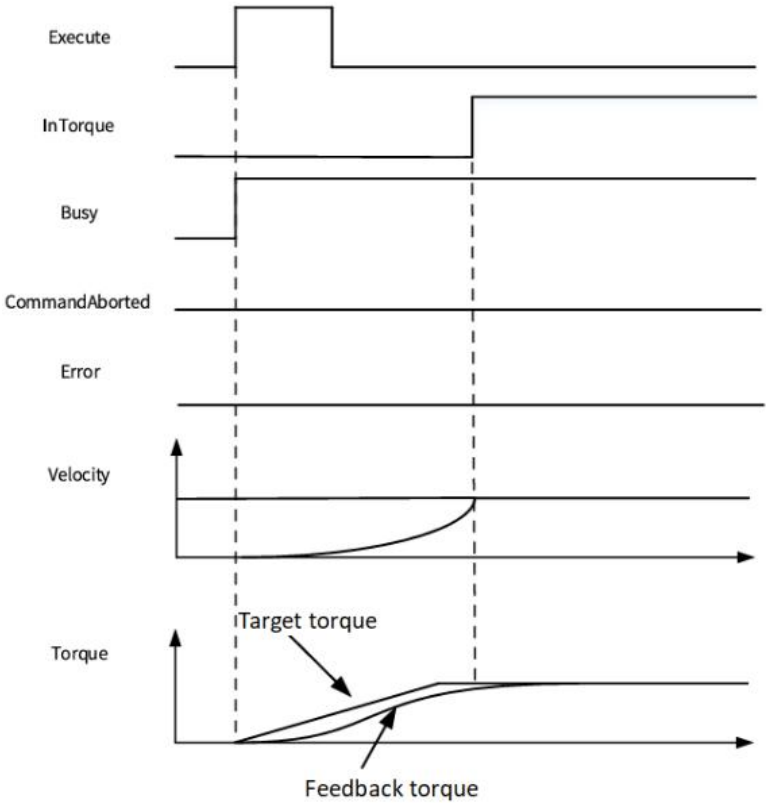
  In PLCOpen, when the instruction is valid, the axis will be in the Continuous Motion state. Other instructions that can make the axis in Discrete motion or comply with PLCOpen state machine switching can interrupt the instruction. When the instruction is interrupted, CommandAboarted is valid;

  CommandAboarted is valid when Execute=ON and Done signal is invalid and is about to hit the limit signal execution needs to perform deceleration;

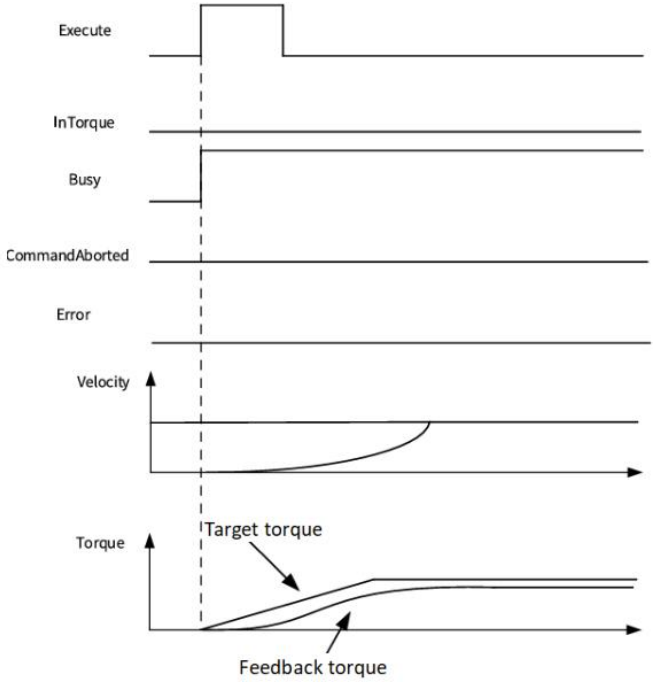
The axis de-enable CommandAboarted is valid when Execute=ON and the Done signal is invalid.

Timing diagram

After setting the target torque, the command is triggered, and the actual output torque can reach the target torque.

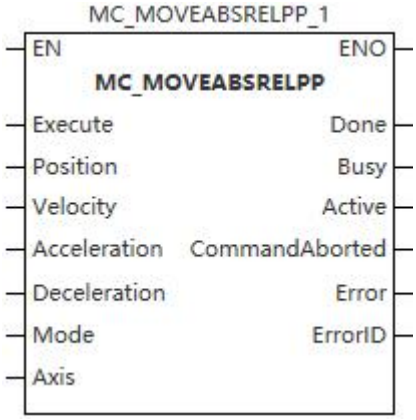


After setting the target torque, the command is triggered, and finally the actual output torque cannot reach the target torque.



8.7.8 MC _ MOVEABSRELPP

MC _ MOVEABSRELPP-Positioning instruction based on contour position mode



Device used (only a single device can be used, and device splicing and Index are not supported)

Instruction	Parameter	Device																						Index modification	Pulse Expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC _ MOVEABSRELPP	Execute	●	●	●	●	●																					
	Position																										
	Velocity																										
	Acceleration																										
	Deceleration																										
	Mode																										
	Axis																										
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Notes for using devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin Type	Parameter	Variable type	Empty or not	Range	Description
MC _ MOVEABSRELPP	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Position	LREAL	No	-999999999.999999 999999999.999999	Target location
		Velocity	LREAL	No	（0,999999999.999999]	Velocity
		Acceleration	LREAL	No	（0,999999999.999999]	Acceleration
		Deceleration	LREAL	No	（0,999999999.999999]	Deceleration
		Mode	INT	No	[0,1]	1: Absolute mode. 1: Relative mode

Instruction	Pin Type	Parameter	Variable type	Empty or not	Range	Description
		Axis	WORD	No	[0,65535]	Axis number
	Output	Done	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Error or not
		ErrorID	DWORD	Yes	-	Error code (Please refer to [8.11 Error code comparison table])

Function

- This command is used to control the EtherCAT bus contour axis to realize the absolute/relative positioning function in contour position mode. In the basic settings of axis configuration, the contour mode option needs to be checked before it can be called, and the rising edge is valid.
- Contour Position Mode requires the following PDOs:

  0x1600: 0x6081, 0x6083, 0x6084, 0x607A, 0x6060, 0x6040

  0x1A00: 0x6061, 0x6064, 0x6041

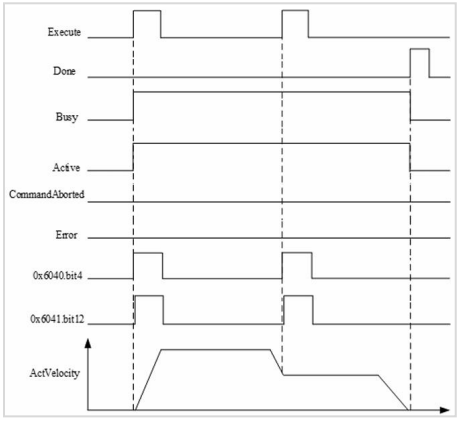
Function description

- This instruction can only be invoked if the axis is switched to the enabled state using the MC _ Power instruction;
- At the rising edge of the Execute input, the instruction latches the input parameters on the left side such as Position and Velocity, triggers the absolute/relative positioning function, and switches the PLCOpen state machine of the axis to the DiscreteMotion state;
- When Mode = 0, the positioning Mode is absolute positioning Mode, and Position is used to set the target Position of absolute positioning. If the axis Mode is configured as rotation Mode, the target Position is positioned according to the shortest distance;
- When Mode = 1, the positioning Mode is a relative positioning Mode, and Position is used to set the target distance of relative positioning;
- Interrupt

  When this instruction is valid in PLCOpen, the axis will be in the Discretemotion state. Other outline mode instructions that can make the axis in Discretemotion or comply with PLCOpen state machine switching can interrupt this instruction. CommandAboarted is valid when the instruction is interrupted. The instruction of periodic synchronization mode cannot be called, otherwise a fault will be reported;

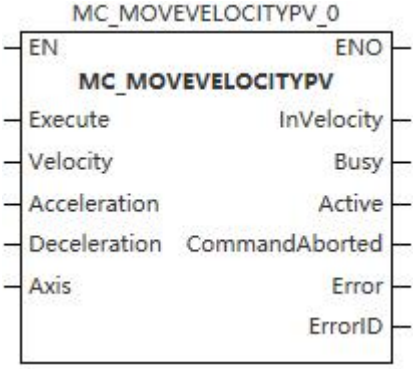
  CommandAboarted is valid when the MC _ Power instruction is invalid when Execute = TRUE and the Busy signal is valid causes the axis to be de-enabled.

Sequence diagram



8.7.9 MC _ MOVEVELOCITYPV

MC _ MOVEVELOCITYPV-Contour Speed Command



Device used (only a single device can be used, and device splicing and Index are not supported)

Instruction	Parameter	Device																								Index Modification	Pulse Expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC _ MOVEVELOCITYPV	Execute	●	●	●	●	●																					
	Velocity																										
	Acceleration																										
	Deceleration																										
	Axis																										
	InVelocity		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	CommandAborted		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																										

Notes for using devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin type	Parameter	Variable Type	Can you Is empty	Range	Description
MC _ MOVEVELOCITYPV	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Velocity	LREAL	No	-999999999.999999 999999999.999999	Velocity
		Acceleration	LREAL	No	（0,999999999.999999]	Acceleration
		Deceleration	LREAL	No	The QPS Exceeds Maximum Limit 5	Deceleration
		Axis	WORD	No	[0,65535]	Axis number
	Output	InVelocity	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag

		Error	BOOL	Yes	TRUE/FALSE	Error or not
		ErrorID	DWORD	Yes	-	Error code (Please refer to [8.11 Error code comparison table])

Function

- This command is used to realize the continuous motion function of EtherCAT bus contour axis in contour speed mode. In the basic settings of axis configuration, the contour mode option needs to be checked before it can be called, and the rising edge is valid;
- This instruction is a simulated velocity instruction based on the profile velocity (PV) mode;
- The following PDOs are required for the profile speed mode:

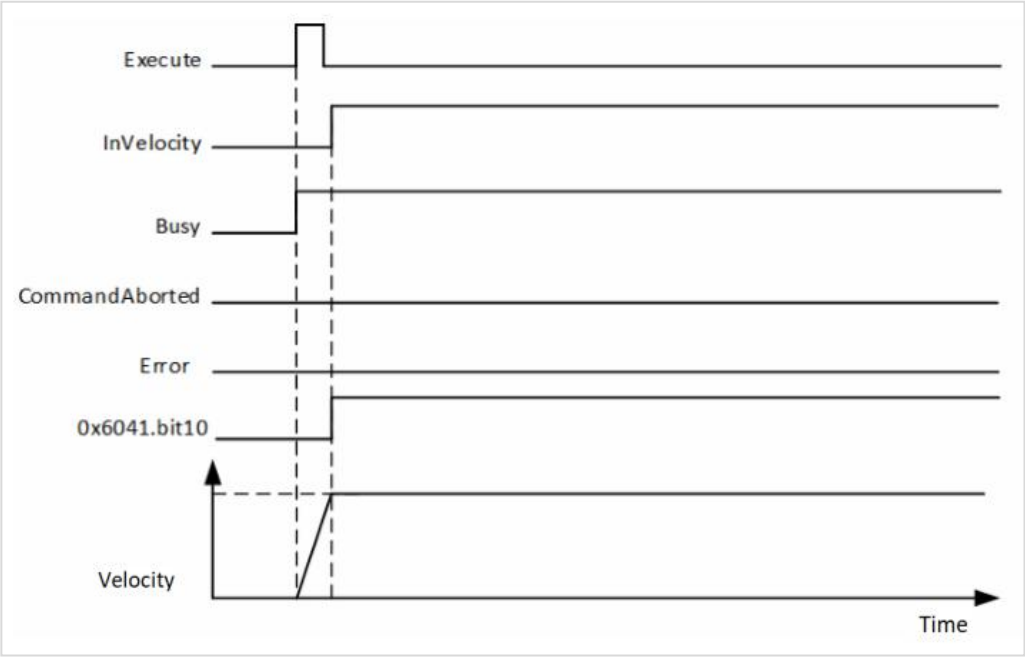
  0x1600: 0x6083, 0x6084, 0x60FF, 0x6060, 0x6040

  0x1A00: 0x6061, 0x606c, 0x6041

Function description

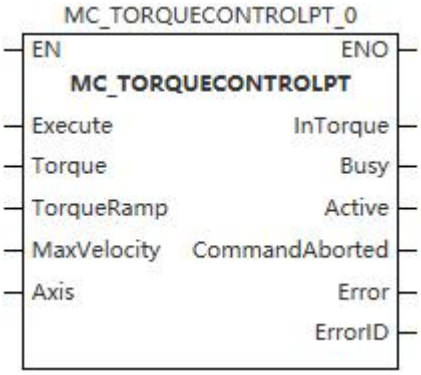
- This instruction can only be invoked if the axis is switched to the STANDSTILL state using the MC _ Power instruction;
- When Execute = TRUE, the instruction latches the input parameters on the left side such as Velocity and Acceleration, the axis continuously runs according to the target speed set by Velocity, and switches the PLCOpen state machine of the axis to the ContinuousMotion state.

Sequence diagram



8.7.10 MC _ TORQUECONTROLPT

MC _ TORQUECONTROLPT-Contour torque control command



Device used (only a single device can be used, and device splicing and Index are not supported)

Instruction	Parameter	Device																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC _ TORQUE CONTROLPT	Execute	●	●	●	●	●																				
	TORQUE																									
	TorqueRamp																									
	MaxVelocity																									
	Axis																									
	InTorque		●	●	●	●																				
	Busy		●	●	●	●																				
	Active		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Notes for using devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable type used

Instruction	Pin Type	Parameter	Variable type	Empty or not	Range	Description
MC_ TORQUECONTROLPT	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Torque	LREAL	No	[-32768,32767]	Target torque (unit is 0.1%)
		TorqueRamp	LREAL	No	（0,999999999.999999]	Torque slope
		MaxVelocity	LREAL	No	[0,999999999.999999]	Maximum speed
		Axis	WORD	No	[0,65535]	Axis number
	Output	InTorque	BOOL	Yes	TRUE/FALSE	Completion flag
		Busy	BOOL	Yes	TRUE/FALSE	Busy flag
		Active	BOOL	Yes	TRUE/FALSE	Valid flag
		CommandAborted	BOOL	Yes	TRUE/FALSE	Abort flag
		Error	BOOL	Yes	TRUE/FALSE	Error or not
		ErrorID	DWORD	Yes	-	Error code (Please refer to [8.11 Error code comparison table])

Function

- This instruction is used to control the EtherCAT bus profile axis to realize the absolute/relative positioning function in profile position mode. In the basic settings of axis configuration, the profile mode option needs to be checked before it can be called, and the rising edge is valid;
- This instruction is used to realize the torque control function and is only used for the EtherCAT bus axis. The rising edge of the instruction is valid and the virtual axis mode is not supported;
- Profile torque mode needs to be configured with the following PDOs:
  0x1600: 0x6087 (optional), 0x6060, 0x6071, 0x607f (optional), 0x6040
  0x1A00: 0x6061, 0x6077, 0x6041

Function description

- This instruction can only be invoked if the axis is switched to the enabled state using the MC _ Power instruction. Otherwise, report a fault;
- This instruction uses the driver synchronous torque mode to realize the torque control function;
- On the rising edge of the instruction, the function block latches the input parameters of TarTorque, TorqueSlope and MaxVelocity, and the axis is in the ContinuousMotion state and performs torque motion;

  TarTorque: Target torque, the unit is 0.1%, one decimal place after the decimal point is valid in the program, and the latter ones are directly discarded. The actual torque of the driver is limited by the maximum positive and negative torques set in the configuration parameters;

  TorqueRamp: Torque ramp, the unit is 1%, one decimal place after the decimal point is valid in the program, and the latter ones are directly discarded.

- Speed control in torque mode

  For servo drives, if 0x607f is mapped in the axis parameter, this instruction limits the maximum speed of the servo motor by 0x607f, if 0x607f is not mapped, the speed limit is invalid;

  On the rising edge of Execute, the instruction converts the speed limit set by Velocity into pulse units and writes it into 0x607f through PDO;

  When the torque instruction is interrupted by others, the maximum speed of the axis will be limited by the "Maximum speed" in the configuration interface;

  For third-party drives, Velocity can only be used as a speed limit under the following conditions:

  The maximum speed of the servomotor can be limited by 0x607F;

  0x607F can be configured into the PDO;

  The unit of 0x607F is pulse unit, non-speed unit.

- Stop control on torque mode

  In the torque mode, the MC _ Stop instruction is called to perform the shut down, and after receiving the Stop instruction, the driver switches to the synchronous position mode, and performs deceleration according to the deceleration set by the Stop instruction.

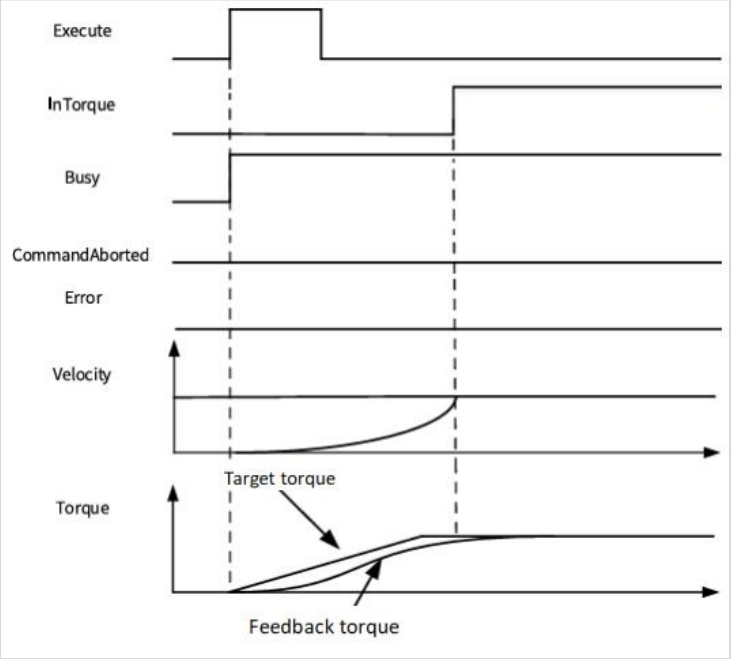
- Interrupt

  In PLCOpen, when the instruction is valid, the axis will be in the Continuous Motion state. Other instructions that can make the axis in Discrete motion or comply with PLCOpen state machine switching can interrupt the instruction. When the instruction is interrupted, CommandAboarted is valid;

  CommandAboarted is valid when Execute=ON and Done signal is invalid and is about to hit the limit signal execution needs to perform deceleration;

  The axis de-enable CommandAboarted is valid when Execute=ON and the Done signal is invalid.

Sequence diagram


Execute
InTorque
Busy
CommandAborted
Error
Velocity
Target torque
Torque
Feedback torque

8.8 SDO instruction

8.8.1 MC_READSDO reads SDO data

MC_READSDO

This instruction is used to read the value of the servo object dictionary.

- [MC_READSDO (S1) (S2) (S3) (D1) (S4)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (Table)
(S1)	Object dictionary index	-	Unsigned BIN16	ANY16_U
(S2)	Object dictionary subindex	-	Unsigned BIN16	ANY16_U
(S3)	Length (BYTE unit)	-	Unsigned BIN16	ANY16_U
(D1)	The read device	-	Define the size according to S3.	–
(S4)	Specify the servo station number	-	Signed BIN16	ANY16_S

Device used

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_READSDO	Parameter 1																	●	●	●	●	●	●				
	Parameter 2																	●	●	●	●	●	●				
	Parameter 3																	●	●	●	●	●	●				
	Parameter 4																	●	●	●	●						
	Parameter 5																	●	●	●	●	●	●				

Features

▪Read the value according to the servo station number (S4), object dictionary index (S1), object dictionary subindex (S2), and length (S3) to the set device (D1).

✎Note:

▪When there is the corresponding object dictionary set in PDO, there may be a situation where SDO cannot read it.

8.8.2 MC_WRITESD0 writes SDO data

This instruction is used to write the value of the servo object dictionary.

-[MCWRITESDO (S1) (S2) (S3) (S4) (S5)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (Table)
(S1)	Object dictionary index	-	Unsigned BIN16	ANY16_U
(S2)	Object dictionary sub-index	-	Unsigned BIN16	ANY16_U
(S3)	Length (BYTE unit)	-	Unsigned BIN16	ANY16_U
(S4)	The value to be written	-	Define the size according to S3.	-
(S5)	Specify the servo station number	-	Signed BIN16	ANY16_S

Device used

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MC_READSDO	Parameter 1																	●	●		●	●	●	●			
	Parameter 2																	●	●		●	●	●	●			
	Parameter 3																	●	●		●	●	●	●			
	Parameter 4																	●	●		●	●					
	Parameter 5																	●	●		●	●	●	●			

Features

▪The value of the device (S4) is written to the positions of the servo station number (S5), the object dictionary index (S1), the object dictionary sub-index (S2), and the length (S3).

✎**Note:**

▪When there is the corresponding object dictionary set in PDO, there may be a situation where SDO cannot write to it.

8.9 Built-in variables of pulse axis system

Name	Data type	Function
Cfg	AXIS_INFO_CFG_T	Axis configuration
wAxisID	WORD	Axis number (Read only)
wVersion	WORD	Version number (Read only)
wAxisType	WORD	Axis type (Read only): Bus servo axis; Bus encoder axis
nEtherCATNo	DINT	EtherCAT axis number
nLocalID	DINT	Local axis ID. Y0/Y1 is 0; Y2/Y3 is 1; Y4/Y5 is 2; Y6/Y7 is 3.
bVirtualAxis	BOOL	Virtual axis or not
Cmd	AXIS_INFO_CMD_T	Setting value
fSetPosition	LREAL	Setting position (Read only)
fSetVelocity	LREAL	Set speed (Read only)
fSetAcceleration	LREAL	Set acceleration (Read only)
fSetDeceleration	LREAL	Actual deceleration (Read only)
fSetJerk	LREAL	Set jerk (Read only)
fSetTorque	LREAL	Set torque (Read only) Reserved
nLocalAxisSetPos	DINT	Local pulse axis setting position (Read only)
Act	AXIS_INFO_ACT_T	Current value
fActPosition	LREAL	Current position (Read only)
fActVelocity	LREAL	Current speed (Read only)
fActAcceleration	LREAL	Actual acceleration (Read only)
fActDeceleration	LREAL	Actual deceleration (Read only)
fActJerk	LREAL	Actual jerk (Read only)
fActTorque	LREAL	Current torque (Read only) Reserved
fFollowPos	LREAL	Following error (Read only)
nLocalAxisActPos	DINT	Local pulse axis current position (Read only)
nLocalAxisActFrequency	DINT	Local pulse axis current frequency (Read only)
Status	AXIS_INFO_STATUS_T	Axis status
wAxisState	WORD	Axis status (Read only)
wAxisError	DWORD	Axis error code (Read only)
wMotionState	WORD	Motion status (Read only)
bPowerState	BOOL	Axis enabled status (Read only)

Name	Data type	Function
bphLimit	BOOL	Hardware positive limit (Read only)
bnhLimit	BOOL	Hardware reverse limit (Read only)
bHomeState	BOOL	Homing limit (Read only)
bPsLimit	BOOL	Software positive limit (Read only)
bnsLimit	BOOL	Software reverse limit (Read only)
bTarDir	BOOL	Direction of motion (Read only)
Dbg	AXIS_DEBUG_T	Axis debugging information
bAxisDebug	BOOL	Axis debugging switch
bPower	BOOL	Axis enable
bStop	BOOL	Axis stop
bReset	BOOL	Axis reset
bJogP	BOOL	JOG+
bJopN	BOOL	JOG-
bHome	BOOL	Homing
bSetPos	BOOL	Set the current position
bAbsPos	BOOL	Absolute positioning
bRevPos	BOOL	Reciprocating motion
bRelPos	BOOL	Relative positioning
bVelocity	BOOL	Continuous motion
bTorque	BOOL	Torque Mode Reserved
fPresetPosition	LREAL	Preset position
fJogVelocity	LREAL	JOG speed
fPositionOffser	LREAL	Homing mode offset
fTarPosition	LREAL	Target position
fTarVelocity	LREAL	Target velocity
fTarAcceleration	LREAL	Target acceleration
fTarDeceleration	LREAL	Target deceleration
fTarJerk	LREAL	Target jerk
fTarPosition2	LREAL	Target position 2
fTarVelocity2	LREAL	Target speed 2
fTarAcceleration2	LREAL	Target accelerate 2
fTarDeceleration2	LREAL	Target deceleration 2

Name	Data type	Function
fTarJerk2	LREAL	Target accelerate 2
fTarTorque	LREAL	Target torque Reserved
fTarTorqueSlop	LREAL	Torque ramp Reserved
fLimitVelocity	LREAL	Limit the speed. Reserved
Pdo	AXIS_INFO_PDO_T	PDO information
wControlWord	WORD	Control word
wStatusWord	WORD	Status word
nSetPosition	DINT	Setting position
nActPosition	DINT	Current position
nSetVelocity	DINT	Setting speed
nActVelocity	DINT	Current speed
nSetTorque	DINT	Setting torque (Read only) Reserved
nActTorque	DINT	Current torque Reserved
nDO	DINT	Digital output Reserved
nDI	DINT	Digital input Reserved
wModesOfPperation	WORD	Control mode
wModesOfPperationDisplay	WORD	Current control mode
wTouchFunction	WORD	Touch probe function settings
wTouchStaus	WORD	Probe touch status
nTouch1PPos	DINT	Rising edge position of touch probe 1
nTouch1NPos	DINT	Falling edge position of touch probe 1
nTouch2PPos	DINT	Rising edge position of touch probe 2
nTouch2NPos	DINT	Falling edge position of touch probe 2
nMaxVelocity	DINT	Maximum velocity Reserved
pParam	AXIS_MODEPARAM_INFO_T	Parameter information
nEncoderMode	BYTE	Encoder model 0: Incremental mode 1: Absolute mode.
nModeSet	BYTE	Mode settings. 0: Linear mode 1: Rotation mode.
dRotatePeriod	LREAL	Rotation cycle
nSpeedADType	BYTE	Speed acceleration and deceleration type. 0: Default mode; 1: T-curve; 2: S-curve.

Name	Data type	Function
bEnSwLimit	BOOL	Whether to enable software limit
dSwPLimit	LREAL	Positive limit unit
dSwNLimit	LREAL	Reverse limit unit
dFollowThreShold	LREAL	Following error threshold unit
dAxisMaxSpeed	LREAL	Axis maximum Velocity unit/s
dAxisMaxASpeed	LREAL	Axis maximum acceleration unit/s^2
dAxisMaxJogSpeed	LREAL	Axis Jog Maximum Speed unit/s
nPMaxTorque	DINT	Positive torque maximum*0.1%
nNMaxTorque	DINT	Negative torque maximum*0.1%
bNoInErrorStop	BOOL	Don't enter ErrorStop after hitting the limit status. Reserved
bHdPLimit	BOOL	Hardware postive limit 0: Positive logic; 1: Negative logic; Reserved: Not in use yet.
bHdNLimit	BOOL	Hardware reverse limit (Read-only) 0: Positive logic; 1: Negative logic; Reserved: Not in use yet.
bEnProbe1	BOOL	Touch probe 1 enable Reserved
bEnProbe2	BOOL	Touch probe 2 enable Reserved
sProbe1	ARRAY[O..5] OF BYTE	Touch probe 1, X0 to X7.
sProbe2	ARRAY[O..5] OF BYTE	Touch probe 2, X0 to X7.
nOutputWay	DWORD	Output mode 0: Pulse+Direction; 1: CW/CCW.
nCounterMode	DWORD	Counting mode 0: A/B phase 1 times frequency; 1: Phase A/B 2 times frequency; 2: Phase A/B 4 times frequency; 3: CW/CCW; 4: Pulse+Direction; 5: Single phase count.
nSignalSource	DWORD	Signal source 0: X0-Phase A; 1: X1-Phase B; 2: X2-Phase A; 3: X3-Phase B.
bEnPreSet	BOOL	Preset enable
sInputTermial	ARRAY[O..5] OF BYTE	Input terminal
bEnCmpOutput	BOOL	Compare output enable
dPulseWidth	LREAL	Pulse width
sOutputTermial	ARRAY[O..5] OF BYTE	Output terminal
sCmpOutputUnit	ARRAY[O..4] OF BYTE	Compare output units ms/Unit

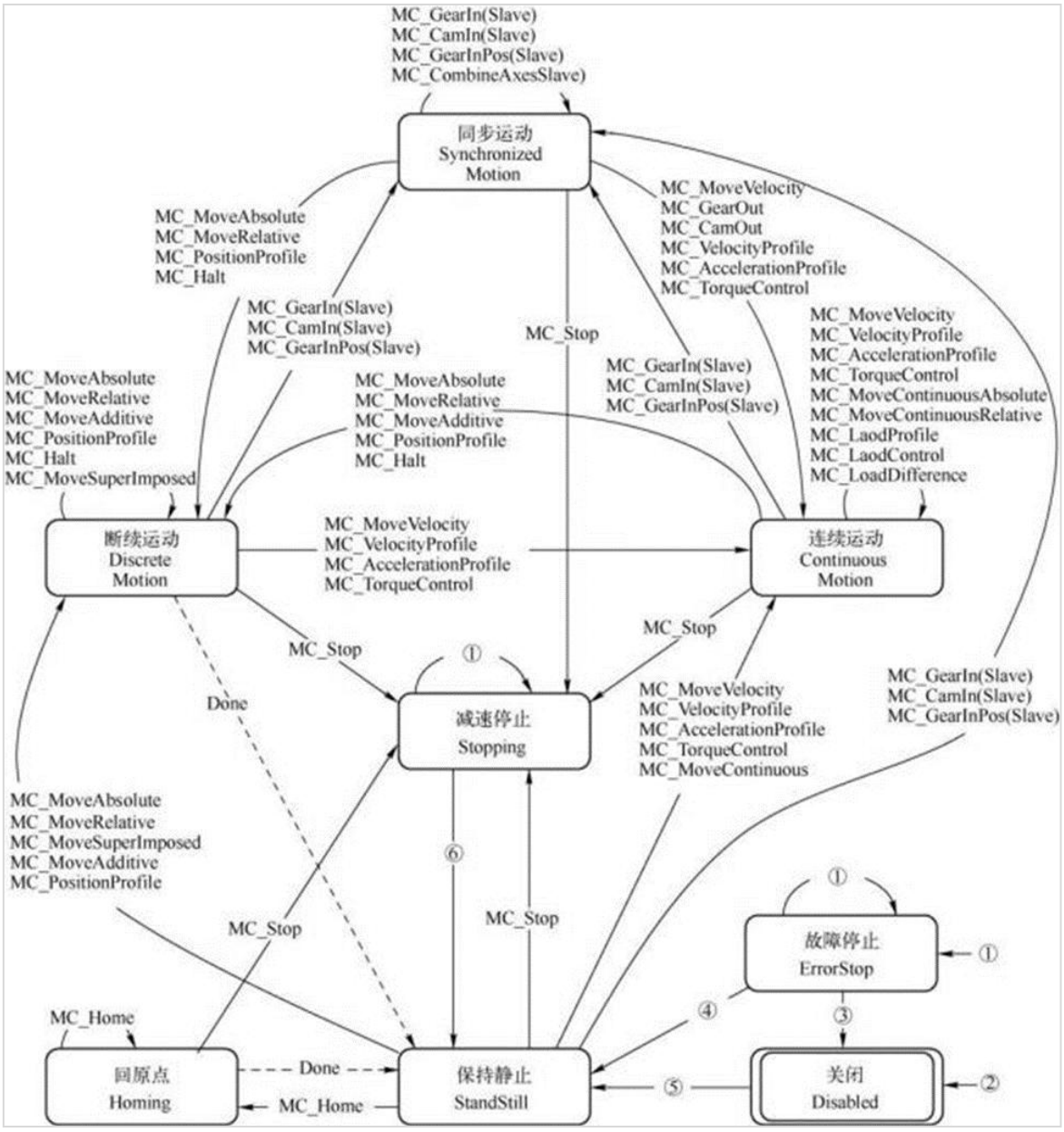
Name	Data type	Function
nFiltTime10ns	DINT	Filtering time design 10us unit
nFreqFilterTime	DINT	Velocity filtering time constant
pOrigin	AXIS_ORIGINRET_INFO_T	Homing information
nOriginSignal	BYTE	Homing signal 0: Not assigned; 1: Not used; 2: Use.
nZSingal	BYTE	Z signal. 0: Not assigned; 1: Not used; 2: In use.
nPLimit	BYTE	Positive limit 0: Not assigned; 1: Not used; 2: In use.
nNLimit	BYTE	Negative limit 0: Not assigned; 1: Not used; 2: In use.
nDirection	BYTE	Homing direction: 0: Not assigned; 1: Forward; 2: Reverse
nInputCheckDirection	BYTE	Enter the detection direction. 0: Not assigned; 0: Positive 2: Reverse
nReturnList	BYTE	Homing list Homing mode 1 to 35
nReturnZero	BYTE	Homing mode 0: Absolute homing 1: Relative homing.
dReturnSpeed	LREAL	Homing speed Unit/s
dReturnAccelerate	LREAL	Homing acceleration Uint/s^2
dReturnNearSpeed	LREAL	Homing approach velocity Unit/s
nReturnTimeOut	DINT	Homing overtime time ms
bEnPLimit	BOOL	Positive limit enables
bEnNLimit	BOOL	Negative limit enable
bEnSignalLimit	BOOL	Homing signal enable
sPLimitAddr	ARRAY[O..5] OF BYTE	Positive limit address X0~X1023. (The local pulse axis currently only supports hardware limit X0-X7)
sNLimitAddr	ARRAY[O..5] OF BYTE	Reverse limit address X0~X1023. (The local pulse axis currently only supports hardware limit X0-X7)
sSignalLimitAddr	ARRAY[O..5] OF BYTE	Homing signal address X0 to X1023 (The local pulse axis currently only supports hardware limit X0-X7)
nPLimitLogic	BOOL	Positive limit logic 0: Forward logic; 1: Reverse logic.
nNLimitLogic	BOOL	Negative limit logic. 0: Positive logic; 1: Negative logic.
nSignalLogic	BOOL	Homing signal logic

Name	Data type	Function
		0: Positive logic; 1: Negative logic.
pUnit	AXIS_UNITCONV_INFO_T	Unit information
bReverse	BOOL	Reversed or not
bEnGearBox	BOOL	Enable gearbox
nRotatePulseNum	DINT	The number of pulses for one revolution of the motor/encoder
dRotateMoveAmount1	LREAL	Movement amount 1 in one rotation
dRotateMoveAmount2	LREAL	Movement amount 2 in one rotation
nGearRatioM	INT	Gear ratio molecule
nGearRatioD	INT	Gear ratio denominator

The current state of the axis can be monitored in the program by its system variables.

8.10 Others

8.10.1 Single axis motion control status diagram



✎**Note**: Continuous line arrows indicate transitions via command. Dashed line arrows indicate automatic transitions, which occur when a command on an axis ends or a related system transition occurs (e.g., a related error).

① The transition condition for any state to fault stop or deceleration stop is: a fault occurs on the axis.

② The transition condition for any state to off is: MC_Power.Enable = FALSE and there is no fault on the axis.

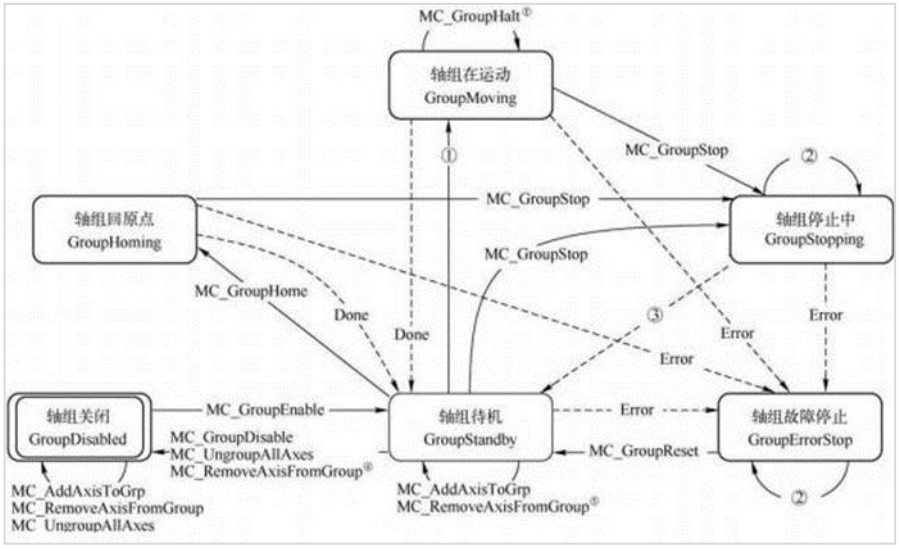
③ MC_Reset = FALSE and MC_Power.Status = FALSE.

④ MC_Reset = TRUE, MC_Power.Status = TRUE, and MC_Power.Enable = TRUE.

⑤ MC_Power.Enable = TRUE and MC_Power.Status = TRUE.

⑥ MC_Stop.Done = TRUE and MC_Stop.Execute = FALSE.

8.10.2 Axis group motion control status diagram



✎**Note**: For all states in the diagram could be transitioned to GroupDisabled state by MC_GroupDisable or MC_UngroupAllAxes.

① Applies to all non-administrative (i.e., motion control) function blocks.

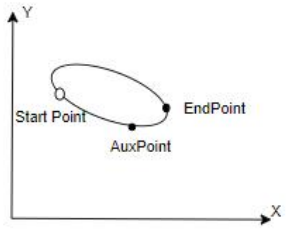
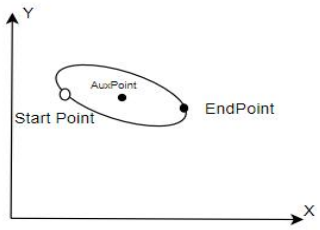
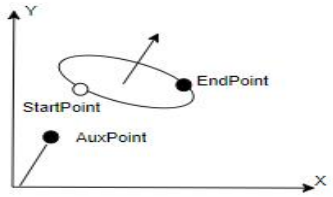
② In the GroupErrorStop or GroupStopping state, all function blocks can be invoked, although they are not executed (except that MC_GroupReset is executed in the GroupErrorStop state and any other errors are generated). In exceptional cases, GroupErrorStop or GroupStopping will transition to GroupStandby or GroupErrorStop accordingly.

③ MC_GroupStop.Done = TRUE and MC_GroupStop.Execute = FALSE.

④ The state transition is available if the last axis is removed from the axis group.

⑤ The transition is available when the axis group is non-empty.

8.10.3 Arc mode description

Mode	Border (CircMode=0)	Circle center (Circmode=1)	Diameter (CircMode=2)
Features	The arc composed of starting point, auxiliary point and ending point is regarded as the trajectory circle of motion.	The auxiliary point is defined as the center of the motion trajectory, so the auxiliary point should be on the perpendicular line connecting the starting point and the ending point.	According to the right-hand grip rule, the ending point and the vector perpendicular to the plane of the circle are defined, and the length of the vector corresponds to the radius of the circle. The direction of the vector determines the direction of CW and CCW circles. The major arc and minor arc are determined by path choice.
Description			
Auxpoint	Arc path point	Center point of arc	Normal vector of the plane where the space arc lies
Path choice	Invalid	CW and CCW circles CCW mode CW mode	Major and minor arcs 1: Minor arc 0: Major arc
Characteristics	Restriction of angle less than 2π for single instruction	Restriction of angle less than 2π and nor equal to π for single instruction	Restriction of angle less than 2π for single instruction The vector perpendicular to the circular plane must be calculated.

8.10.4 Transition mode description

Detailed explanation of transition mode: Transition parameters are additional parameters of transition modes, which take effect according to different transition modes.

TransitionMode	0 (No buffer mode)	1 (Largest velocity to buffer)	2 (Specified velocity buffer)
TransitionParameter	Invalid	Invalid	Specified velocity

8.10.5 Mode selection instructions

No.	MC_BUFFER_MODE	Description
0	mcAborting	The agreed mode: immediately enable the next function block.
1	mcBuffered	Enable the next function block when the motion of the current block finished.
2	mcBlendingLow	The blending velocity is the lower of the velocity of the previous and the next function blocks.
3	maBlendingPrevious	The blending velocity is the velocity of the previous between two function blocks.
4	mcBlendingNext	The blending velocity is the velocity of the next between two function blocks.
5	mcBlendingHigh	The blending velocity is the higher of the velocity of the previous and the next function blocks.

CAUTION

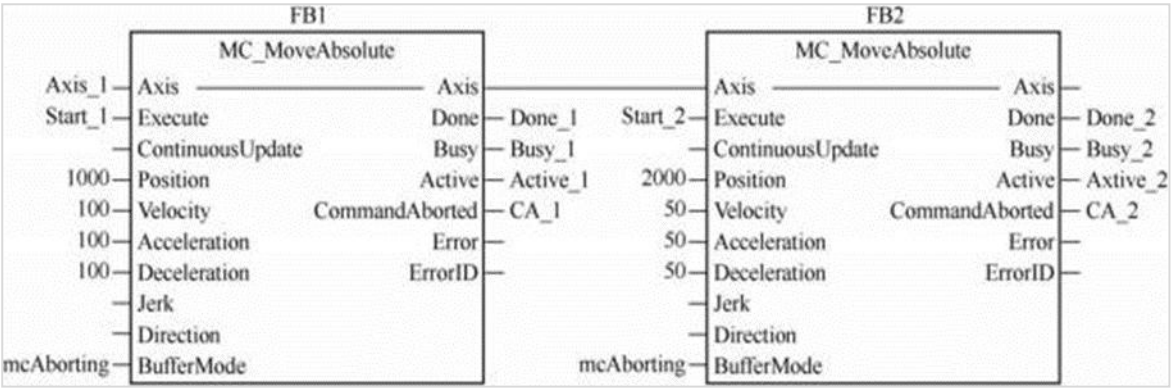
It should be noted that if the moving function block is interrupted by another function block, the following error will occur: braking distance is insufficient due to the limitation of deceleration;
Note that in all the sequence diagrams explained below, some of the sequence diagrams for the status word feedback are incorrect. For example, in Figure 8-2, the start signal is not automatically set by the system after triggering, but controlled by the user. Function block triggers execution logic when on the rising edge.
Likewise, the Done signal is not set automatically. When the function block ends, Done will be set until the falling edge of Execute is generated!

(1) mcAborting

It is an agreed pattern without buffering. When the status of the application axis is changed to "ErrorStop", all motion control function blocks with buffer mode will enter mcAborting.

At this time, the function block output "Error" in mcAborting is set. Its subsequent instructions are rejected because the output Error is set, so Active is reset.

The following figure illustrates the function connection of the front and back function block instances in mcAborting.



As shown in Figure 8-1, the previous and next motion control function blocks are MC_MoveAbsolute, and both adopt mcAborting.

The following Figure 8-2 shows the sequence diagrams of function block instances FB1 and FB2. Figure 8-2(a) shows the sequence diagram of two absolute position motion function blocks under normal working conditions, i.e., the motion controlled by FB2 is carried out after the motion controlled by FB1 is completed.

When on the Start_1=TRUE rising edge, the function block instance FB1 is activated. At the same time, the outputs Busy_1 and Active_1 are set. The motion process of the axis is as follows: the velocity increases to 100u/s according to the acceleration of 100u/s2, and keep the axis move at constant velocity. Then the velocity decreases to 0 with deceleration of 100u/s2, thus moving the axis to 1000u.

Then the output Done 1 is set, and Busy 1 and Active 1 are reset.

Axis position is 1000u.

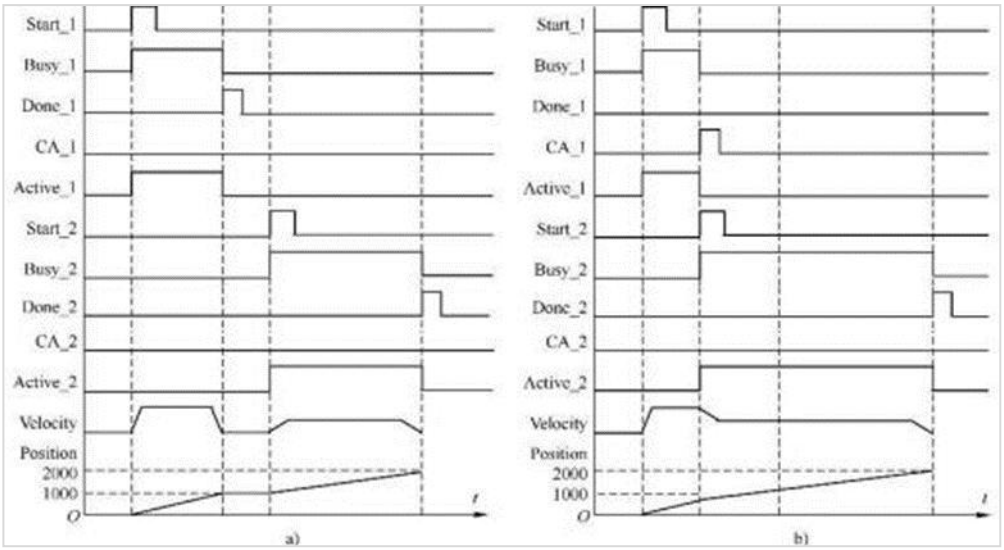


Figure8-2 The sequence diagram of normal working condition and mcAborting

When on the Start_2=TRUE rising edge, the second function block instance FB2 is activated. The output Busy_2 and Active 2 are set. According to the acceleration of 50u/s2, the axis velocity increases to the constant velocity of 50u/s. Then the axis velocity decreases to 0 at the deceleration of 50u/s. The axis position is from 1000u to 2000u.

Since the absolute position has been reached, output Done_2 is set and Busy_2 and Active_2 are reset. Axis position is 2000u.

Figure 8-2(b) shows the sequence diagrams of two absolute position motion function blocks in mcAborting. On the rising edge of start1 = TRUE, the function block instanceFB1is activated.

At the same time, the output Busy_2 and Active_1 are set. The axis motion velocity increases according to the acceleration of 100u/s2. But in this process, the output start 2 of the function block instance FB2 is set, which abort the motion of the previous function block instance FB1. At this time, the axis dose not reach the required position 1000u and is aborted to stop.

As the function block instance FB2 is activated, Busy_2 and Active_2 are set. According to the requirement of the function block instance FB2, the velocity of the axis decreases to the input value of 50u/s2 according to its deceleration of 50u/s2. Keep it move at the constant velocity of 50u/s so that it will finally reach the position 2000u.

It can be seen that the activation of the next function block causes the previous function block to stop its motion, and the aborting instruction immediately affects the motion of the axis.

This buffer mode is called mcAborting.

(2) mcBuffered

The instruction of the next function block is stored in the buffer area. The instruction of the buffer area is executed only after the motion of the previous function block is completed. Therefore, this mode is called mcBuffered.

The BufferMode input for the next function block is set to be buffered mode (mcBuffered). Figure 8-3shows the setting for the connection of two function blocks.

The sequence diagram of mcBuffered is shown in Figure 8-4. In the figure, FB1 is activated when the input start 1=TRUE of the previous function block instance is on the rising edge. When this function block instance FB1 control the axis to move, the start 2 of the next function block instance FB2 is set. Due to the FB2 buffer mode is set to mcBuffered, the activation of FB2 does not abort the control of FB1 to the axis. Although the output Busy_2 of FB2 is set, the function block is not activated, namely, the output Active_2 is still FALSE.

Therefore, FB1 continues to control the motion of the axis until the axis reaches the set position 1000u. Then FB2 is immediately activated to work, and the velocity of the axis is not delayed. Increase the velocity to 50u/s based on the acceleration of 50u/s2 directly and then keep it move at the constant velocity of 50u/s, so that it will finally reach the position of 2000.

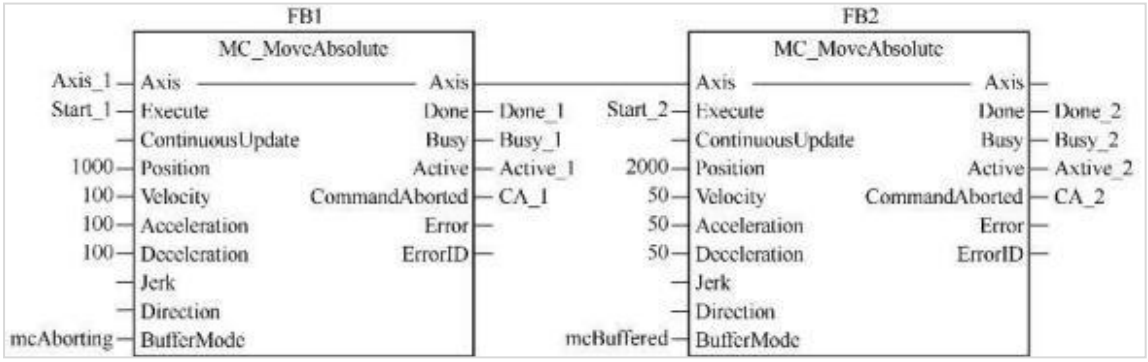
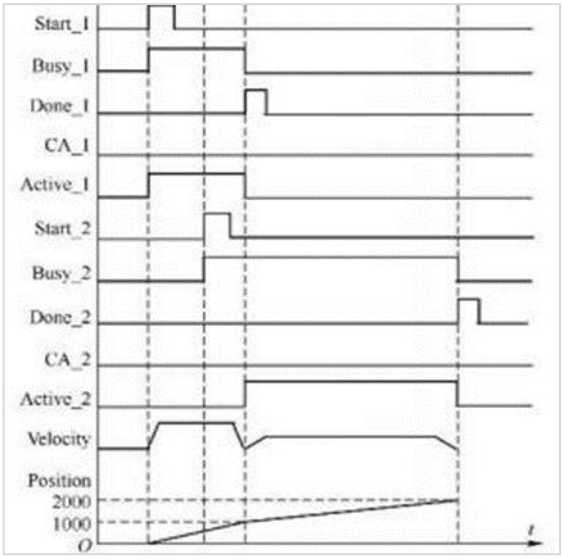


Figure8-3 Connections in mcBuffered

It can be seen from the figure that the buffer mode of the rear function block is set to mcBuffered, which means that the rear function block cannot interrupt the motion of the previous function block. And once the previous function block completes its task. and Done_1 is set, the work of the next function block will start immediately.

The mcBuffered indicates that the motion instruction of the next function block has been stored in the buffer area, which does not affect the control and operation of previous motion control function block over the axis. It can execute subsequent instructions after the executed function block outputs Done or Inxxx becomes TRUE.



mcBlending

There are four options for mcBlending of the previous and next function blocks. The mcBlending represents the blending of the velocity of the previous and next functional blocks. In this case, the previous function block interrupts its motion, and the next function block controls the subsequent motion, but its initial velocity is called blending velocity or mixing velocity.

The mixing velocity can adopt the velocity of two function blocks (whichever is lower or higher) or it can be directly specified that the velocity of the previous function block or the velocity of the next function block is adopted.

Therefore, the mcBlending is divided into four types.

(1) mcBlendingLow.

The blending velocity is the lower of the set velocity of the two function blocks. This mcBlending can be random. If the set velocity of the previous function block is lower than that of the next function block, the blending velocity adopts the velocity of the previous function block.

On the contrary, if the set velocity of the previous function block is higher than that of the next function block, the blending velocity adopts the velocity of the next function block.

When the buffer mode of the next function block is set to be mcBlendingLow, it means that the next function block will not interrupt the work of previous function block if the work of the previous function block is not completed; when the velocity of the

previous function block decreases to the lower velocity of the previous and next function blocks, the next function block can continue to control the motion of the axis.

Figure 8-5 shows the connection diagram of three motion blocks, and the mode of latter two function blocks are set to be mcBlendingLow.

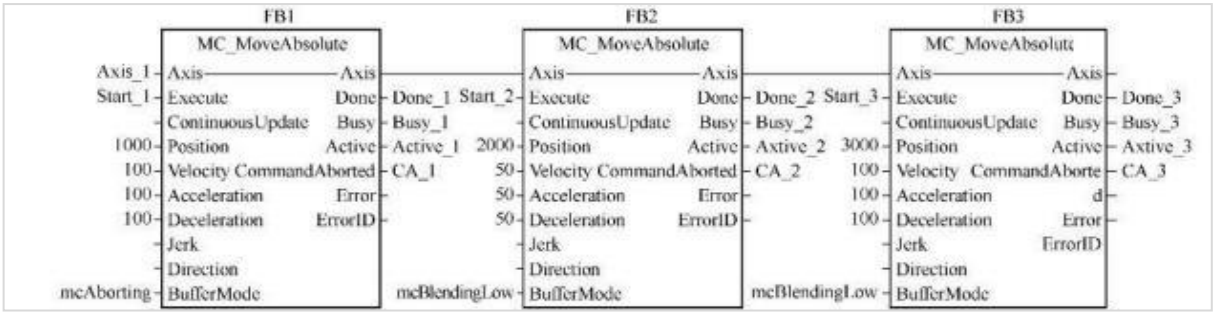


Figure 8-5 The connection of mcBlendingLow

Figure 8-6 shows the sequence diagram of mcBlendingLow. To illustrate maBlendingLow, two subsequent function block instance FB2 and FB3 are used in the figure. In the figure, FB1 is activated when start_1 is set on the rising edge and then its BusY_1 is set. When FB1 controls the axis to move, start=2 of the FB2 instance is also set on the rising edge. FB1 does not complete the axis motion, so the Buffer-Mode of FB2 is set to mcBlendingLow.

Therefore, FB1 adopts the minimum velocity after blending FB1 and FB2, namely, the set velocity of FB2 of 50u/s.

It means that in the subsequent motion of FB1, the velocity of the axis is reduced to 50u/s according to the deceleration of FB2. When the axis velocity drops to the mixed low velocity, the FB1 control axis moves to the set position of 1000u, and the output Done_1 of FB1 is set, which indicates the end of its controlled motion.

FB2 immediately controls the axis to move at a constant velocity of 50u/s, and the axis is moving to specified position of 2000u.

At this time, if the Start_3 of the function block instance FB3 is set, since FB2 has not reached its position of 2000u, and the Buffer-Mode of FB3 is set to be mcBlendingLow, when the Start_3 of FB3 is set, the moving axis will control the axis according to the set velocity of FB2 and FB3 (whichever is lower) until the position of the axis reaches the set position of 2000u, then the output Active_2 is reset when FB2 is activated.

At this time, if the Start_3 of the function block instance FB3 is set, since FB2 has not reached its position of 2000u, and the Buffer-Mode of FB3 is set to be mcBlendingLow, when the Start_3 of FB3 is set, the moving axis will control the axis according to the set velocity of FB2 and FB3 (whichever is lower) until the position of the axis reaches the set position of 2000u, then the output Active_2 is reset when FB2 is activated. Then under the control of the FB3 function block instance, increase the velocity of the axis to 100u/s; then keep the axis move at constant velocity, and finally reduce the velocity to 0u/s; at this time the axis reaches the set position of 3000u.

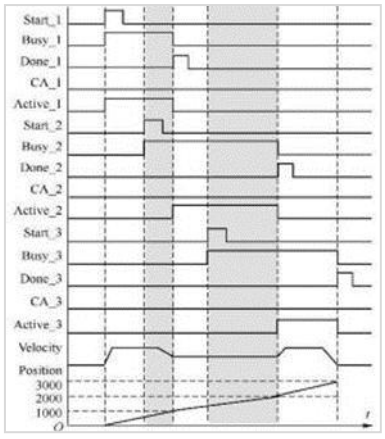


Figure 8-6 The sequence diagram of mcBlendingLow

mcBlending means that due to the difference in the set velocity of the previous and next function blocks, when the Execute rising edge input of the next function block is set, the system will determine what velocity should be adopted to control the axis according

to the BufferMode parameter set by the next function block.

The aforesaid mcBlendingLow adopts the set velocity of the two function blocks (whichever is lower).

The previous function block will complete the subsequent motion under the control of blending velocity until it reaches the set position, and the subsequent function block will continue to control the axis. mcBlendingPrevious: the blending velocity adopts the velocity set by the previous function block.

mcBlendingPrevious

As shown in Figure 8-5, if the BufferMode parameter of FB2 and FB3 is set to be mcBlendingPrevious, the blending velocity directly adopts the velocity of the previous function block.

Figure 8-7 shows the timing diagram of the mcBlendingPrevious.

When analyzing the sequence diagram, pay attention to the change of blending velocity. As shown in Figure 8-7, in the timing diagram of the mcBlendingPrevious, when FB1 is in buffer mode, the axis moves according to the velocity of the previous function block FB1, i.e., 100u/s, until it reaches the set position of 1000u.

When FB2 is in buffer mode, the axis moves according to the velocity of the previous function block FB2, i.e., 50u/s, until it reaches the set position of 2000u.

Therefore, the axis velocity is100u/s when FB1 output Active is reset, and the axis velocity is 50u/s when FB2 output Active is reset.

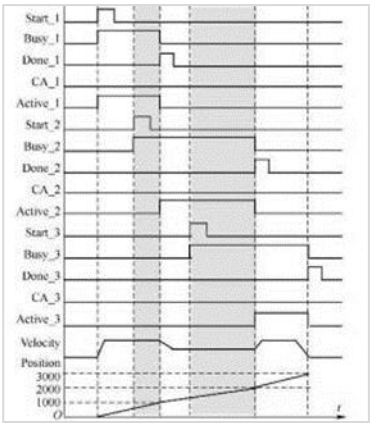


Figure 8-7 Sequence diagram of mcBlendingPrevious

mcBlendingNext

As shown in Figure 8-5, if the BufferMode parameter of FB2 and FB3 is set to be mcBlendingNext, the blending velocity directly adopts the velocity of the next function block.

Figure 8-8 shows the timing diagram of the mcBlendingNext.

As shown in Figure 8-8, in the timing diagram of the mcBlendingNext, when FB1 is in buffer mode, the axis moves according to the velocity of the next function block FB2, i.e., 50u/s,

until it reaches the set position of 1000u. When FB2 is in buffer mode, the axis moves at 100U/S according to the velocity of the next function block FB3, until it reaches the set position of 2000U. Therefore, the axis velocity is 50u/s when the FB1 output Active is reset, and the axis velocity is 50u/s when FB2 output Active is reset.

As shown in Figure 8-6 of the sequence diagram of the mcBlendingLow, when FB1 is in buffer mode, the axis buffers at the low blending velocity and moves at 50U/S according to the velocity of FB2, until it reaches the set position of 1000U.

When FB2 is in buffer mode, the axis buffers at the low blending velocity and moves at 50U/S according to the velocity of the FB2, until it reaches the set position of 2000U.

Therefore, the axis velocity is 50u/s when the FB1 output Active is reset, and the axis velocity is 50u/s when FB2 output Active is reset.

until it reaches the set position of 1000u. While when FB2 is in buffer mode, the axis moves according to the velocity of the next function block FB3, i.e. 100u/s, until it reaches the set position of 2000u. Therefore, the axis velocity is 50u/s when FB1 output Active is reset, and the axis velocity is 100u/s when FB2 output Active is reset.

As shown in Figure 8-6, in the timing diagram of mcBlendingLow, when FB1 is in buffer mode, the axis buffers at the low blending velocity and moves according to the velocity of FB2, i.e. 50u/s, until it reaches the set position of 1000u. While when FB2 is in buffer mode, the axis buffers at the low blending velocity and moves according to the velocity of the FB2, i.e. 50u/s, until it reaches the set position of 2000u. Therefore, the axis velocity is 50u/s when FB1 output Active is reset, and the axis velocity is 50u/s when FB2 output Active is reset.

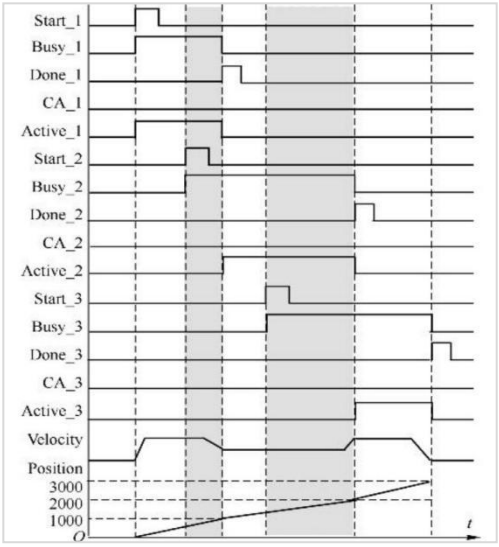


Figure 8-8 Sequence diagram of mcBlendingNext

2) mcBlendingHigh

As shown in Figure 8-5, if BufferMode parameter of FB2 and FB3 is set to be mcBlendingHigh, the blending velocity directly adopts the higher of the set velocity of the previous and next function blocks.

As shown in Figure 8-9, in the timing diagram of mcBlendingHigh, when FB1 of the previous function block instance is in buffer mode, the axis buffers at the high blending velocity and moves according to the velocity of FB1, i.e. 100u/s, until it reaches the set position of 1000u. While when FB2 is in buffer mode, the axis buffers at the high blending velocity and moves according to the velocity of the FB3, i.e. 100u/s, until it reaches the set position of 2000u. Therefore, the axis velocity is 100u/s when FB1 output Active is reset and will be reduced to 50u/s. The axis velocity is 100u/s when FB2 output Active is reset.

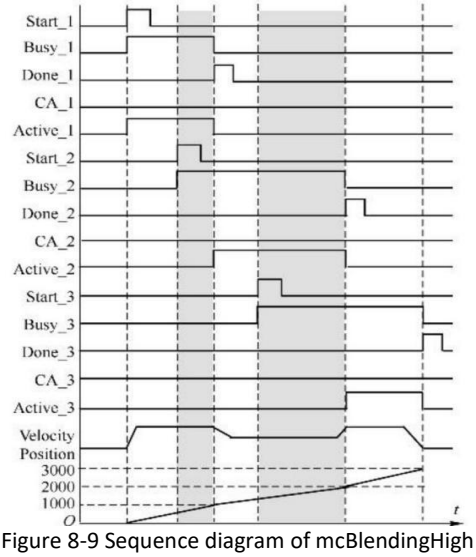


Figure 8-9 Sequence diagram of mcBlendingHigh

Description of modes of E-CAM

StartMode parameter, together with the MasterAbsolute/SlaveAbsolute parameter in the CAMTABLESELECT instruction, determines the motion mode of the master/slave axis, and the master axis mode is determined only by MasterAbsolute and is not affected by the value in StartMode. The slave axis mode is represented as follows:

Start Mode	SlaveAbsolute	Slave axis mode
Absolute	Relative	Relative
Absolute	Absolute	Absolute
Relative	Relative	Relative

Relative	Absolute	Relative
Catchup	Relative	Relative catchup
Catchup	Absolute	Absolute catchup

The result of master-slave axis absolute/relative mode when executing CAMIN instruction.

Master axis mode	Slave axis mode	Result
Relative	Relative	After CAMIN is executed, the position of slave axis is transitioned to the set offset position; After master axis runs, the slave axis performs relative motion according to the starting point of the cam table.
	Absolute	After CAMIN is executed, the position of slave axis is transitioned to the starting position of the cam table (i.e. 0); After master axis runs, the slave axis moves according to the corresponding point of the cam table.
	Relative catchup	After CAMIN is executed, the position of slave axis is moved to the set offset position via catch-up; After master axis runs, the slave axis performs relative motion according to the corresponding point of the cam table.
	Absolute catchup	After CAMIN is executed, the position of slave axis is moved to the starting position of the cam table (i.e. 0) via catch-up; After master axis runs, the slave axis moves according to the corresponding point of the cam table.
Absolute	Relative	After CAMIN is executed, the position of slave axis is transitioned to the set offset position; After master axis runs, the slave axis performs relative motion according to the corresponding point of the cam table.
	Absolute	After CAMIN is executed, The slave axis position jumps to the position corresponding to the current master axis of the cam table. (Example: The current position of the master axis is 100, the slave axis point corresponding to the master axis point 100 in the cam table is 200, and the slave axis transition after CAMIN is executed); After the main axis runs, the slave axis moves relatively according to the corresponding point of the cam table
	Relative catch up	After CAMIN is executed, the position of slave axis is transitioned to the set offset position; After master axis runs, the slave axis performs relative motion according to the corresponding point of the cam table.
	Absolute catch up	After CAMIN is executed, The slave axis position jumps to the position corresponding to the current master axis of the cam table (Example: The current position of the master axis is 100, the slave axis point corresponding to the master axis point 100 in the cam table is 200, and the slave axis transition after CAMIN is executed); After the main axis runs, the slave axis moves relatively according to the corresponding point of the cam table

Precautions

When the master axis is in absolute mode, if the current position of the master axis is not within the range of the master axis of the cam table, the periodization process will be automatically performed, for example: The current position of the master axis is 110 and the position of the master axis in the cam table is 0 to 100, then the default master axis position after CAMIN is executed is 10 (the actual master axis position does not change);

The magnification and offset parameter of master and slave are effective when CAMIN is executed; and it don't support modified during running. Unsuitable parameters will lead to slave axis position transition. The position relationship between master and slave axis is (where CAM () represents the corresponding slave axis position of the master axis on the CAM table) :

Slave axis position = slave axis magnification*CAM ((master axis position * master axis magnification + master axis offset)) + slave axis offset

. Magnification of the master and slave axis cannot be 0.

8.11 Error Code

Error Code	Error Definition	Interpretation
Normal		
0x00	MC_ERROR_ID_SUCCESS	No error
Run-time error		
0x00200000	MC_ERROR_ID_SCURVE_NOTFEASIBLE	Not feasible to compute S-curve
0x00200001	MC_ERROR_ID_SCURVE_MAXVEL_NOTREACHED	Not reach maximum velocity
0x00200002	MC_ERROR_ID_SCURVE_MAXACC_NOTREACHED	Not reach maximum acceleration
0x00200003	MC_ERROR_ID_SCURVE_FINDMAXACC_FAILED	Fail to find maximum velocity
0x00200004	MC_ERROR_ID_SCURVE_INVALIDINPUT	Invalid constraint for S-curve planning
PLCOpen State-machine error		
0x00300000	MC_ERROR_ID_AXIS_STATE_VIOLATION	Invalid axis state violation
0x00300001	MC_ERROR_ID_POWER_ON_OFF_FROM_ERROR_STOP	Try to power on the servo at ErrorStop
0x00300002	MC_ERROR_ID_INVALID_STATE_FROM_STOPPING	Invalid state transition at Stopping
0x00300003	MC_ERROR_ID_INVALID_STATE_FROM_ERROR_STOP	Invalid state transition at ErrorStop
0x00300004	MC_ERROR_ID_INVALID_STATE_FROM_DISABLED	Invalid state transition at Disabled
0x00300005	MC_ERROR_ID_INVALID_STATE_FROM_POWERED	Invalid state transition at Powered
0x00300006	MC_ERROR_ID_INVALID_STATE_FROM_SYNC	Invalid state transition at Sync
0x00300007	MC_ERROR_ID_INVALID_STATE_NOT_HOME	Invalid state transition when not home
0x00300008	MC_ERROR_ID_INVALID_STATE_TO_DISABLED	Invalid state transition to Disabled
0x00300009	MC_ERROR_ID_INVALID_STATE_TO_ERRORSTOP	Invalid state transition to ErrorStop
0x0030000A	MC_ERROR_ID_INVALID_STATE_TO_STANDSTILL	Invalid state transition to StandStill
0x0030000B	MC_ERROR_ID_INVALID_STATE_TO_HOMING	Invalid state transition to Homing
0x0030000C	MC_ERROR_ID_INVALID_STATE_TO_STOPPING	Invalid state transition to Stoppping
0x0030000D	MC_ERROR_ID_INVALID_STATE_TO_DISCRETE	Invalid state transition to DiscreteMotion
0x0030000E	MC_ERROR_ID_INVALID_STATE_TO_SYNC	Invalid state transition to SynchronizedMotion
0x0030000F	MC_ERROR_ID_INVALID_STATE_TO_CONTINUOUS	Invalid state transition to ContinuousMotion
0x00300010	MC_ERROR_ID_INVALID_STATE_FROM_HOMING	Invalid state transition at Homing
0x00300011	MC_ERROR_ID_INVALID_STATE_FROM_DISCRETE	Invalid state transition at DiscreteMotion
0x00300012	MC_ERROR_ID_INVALID_STATE_FROM_ CONTINUOUS	Invalid state transition at ContinuousMotion

Error due to user/system limitations		
0x00400000	MC_ERROR_ID_MOTION_LIMIT_ERROR	Velocity, acceleration or position over limit
0x00400001	MC_ERROR_ID_INVALID_DIRECTION_POSITIVE	No positive direction motion allowed
0x00400002	MC_ERROR_ID_INVALID_DIRECTION_NEGATIVE	No negative direction motion allowed
0x00400003	MC_ERROR_ID_VELOCITY_OVER_LIMIT	Velocity over limit
0x00400004	MC_ERROR_ID_ACCELERATION_OVER_LIMIT	Acceleration over limit
0x00400005	MC_ERROR_ID_POSITION_OVER_POSITIVE_LIMIT	Position over positive limit
0x00400006	MC_ERROR_ID_POSITION_OVER_NEGATIVE_LIMIT	Position over negative limit
0x00400007	MC_ERROR_ID_POSITION_LAG_OVERLIMIT	Position lag over limit
0x00400008	MC_ERROR_ID_TORQUE_OVER_LIMIT	Torque over limit

Error Stop		
0x00500000	MC_ERROR_ID_AXIS_ERROR_STOP	Axis error stopped
Error from servo		
0x00600000	MC_ERROR_ID_AXIS_ERROR_STOP	Servo has no error
0x00600001	MC_ERROR_ID_SERVO_FIELDBUS_INIT_ERROR	Servo fieldbus initialize error
0x00600002	MC_ERROR_ID_SERVO_POWER_ERROR	Servo power error
0x00600003	MC_ERROR_ID_SERVO_POWERING_ON_ERROR	Servo has error during powering on
0x00600004	MC_ERROR_ID_SERVO_ERROR_WHEN_POWERED_ON	Servo has error after powered on
0x00600005	MC_ERROR_ID_SERVO_POWERING_OFF_ERROR	Servo has error during powering off
0x00600006	MC_ERROR_ID_SERVO_PDO_NOT_MAP	Servo PDO not map
0x00600007	MC_ERROR_ID_SERVO_PDO_READ_ERROR	Servo PDO read error
0x00600008	MC_ERROR_ID_SERVO_PDO_WRITE_ERROR	Servo PDO write error
0x00600009	MC_ERROR_ID_SERVO_POWER_OFF	Servo can’t run in power off
0x0060000A	MC_ERROR_ID_SERVO_ERROR_HOME_STATE	Servo has error when homing state
0x0060000B	MC_ERROR_ID_SERVO_HOME_MODE_SWITCH	Servo mode switch error
0x0060000C	MC_ERROR_ID_SERVO_HOME_CTRLWORD	Servo set homing mode error
0x0060000D	MC_ERROR_ID_SERVO_HOME	Servo homing error
0x0060000E	MC_ERROR_ID_SERVO_PDO_READ_BUSY	Servo PDO read busy
0x0060000F	MC_ERROR_ID_SERVO_TOUCHPROBE_NOTFIND	Fail to find Servo TouchProbe
0x00600010	MC_ERROR_ID_SERVO_PARAREAD	Read Servo parameter error
0x00600011	MC_ERROR_ID_SERVO_PARAWRITE	Write Servo parameter error
0x00600012	MC_LOCAL_PULSE_HOME_TIMEOUT	Time out when origin regression
0x00600013	MC_LOCAL_PULSE_HOME_NOT_TOUCH_LIMIT	The execution has not encountered the limit
0x00600014	MC_ERROR_ID_LOCAL_PULSE_HOME_VELOCITY_ OVER_LIMIT	Pulse shaft HOME Velocity over limit
0x00600015	MC_ERROR_ID_LOCAL_PULSE_HOME_ACC_ OVER_LIMIT	Pulse shaft HOME Acceleration over limit
0x00600020	MC_ERROR_ID_MODE_SWITCH_NEEDPDO_ NOTMAP	There is no required PDO for the mode corresponding to mode switching.
0x006000FF	MC_ERROR_ID_SERVO_LOST	Servo lost

Axis's error		
0x00700000	MC_ERROR_ID_AXIS_GENERAL_ERROR	Axis's general error
0x00700001	MC_ERROR_ID_AXIS_ADD_QUEUE_ERROR	Axis' add error
0x00700002	MC_ERROR_ID_AXIS_IS_UNHEALTHY	Axis is unhealthy
0x00700003	MC_ERROR_ID_AXIS_IS_OVER_LIMIT	Axis's motion instruction is over limit
0x00700004	MC_ERROR_ID_AXIS_FALLINGEDGE_ERROR	Axis error when on the falling edge
0x00700005	MC_ERROR_ID_AXIS_NOTFIND	Not find the axis
0x00700006	MC_ERROR_ID_AXIS_IS_INGROUP	Axis is in Group
0x00700007	MC_ERROR_ID_AXIS_CAN_NOT_SETPOS	Axis cannot be set position in current state
0x00700008	MC_ERROR_ID_AXIS_NOT_POWER	Axis does not power on yet
0x00700009	MC_ERROR_ID_AXIS_NOT_SUPPORTED_TOUCH	Not support the bound touch probe number
0x0070000A	MC_ERROR_ID_AXIS_IS_COMBINED	Axis is combined
0x0070000B	MC_ERROR_ID_AXIS_CAN_NOT_COMBINE_WHILE_RUN	Axis can't combine while running
0x0070000C	MC_ERROR_ID_AXIS_GROUP_NOTFIND	Not find axis group number
0x0070000D	MC_ERROR_ID_AXIS_CAN_NOT_RUN_PLURAL_ HOME	FB does not support this curve
0x0070000E	MC_ERROR_ID_AXIS_TYPE_WRONG	This instruction cannot be executed and cannot be superimposed
General FB error		
0x00800000	MC_ERROR_ID_FB_GENERAL_ERROR	User's FB run error
0x00800001	MC_ERROR_ID_FB_READ_PARAM_ERROR	User's FB read error
0x00800002	MC_ERROR_ID_FB_STOP_ERROR	User's FB stop error
0x00800003	MC_ERROR_ID_FB_NOT_BIND_AXIS	User's FB does not bind the logic axis
0x00800004	MC_ERROR_ID_FB_NOT_SET_PARAMETER_ID	User's FB dose not set the read parameter
0x00800005	MC_ERROR_ID_FB_NOT_IN_QUEUE	User's FB has already done and can't update
0x00800006	MC_ERROR_ID_FB_PARAMETER_ERROR	User's FB parameter error
0x00800007	MC_ERROR_ID_FB_PLAN_FAILED	User's FB error due to parameter when executing planning
0x00800008	MC_ERROR_ID_FB_REPLACE_AXIS_IN_RUN	User's FB replace axis in running
0x00800009	MC_ERROR_ID_FB_HALTSUPERIMPOSED_ERROR	Stop virtual superimposed error
0x0080000A	MC_ERROR_ID_FB_NOT_BIND_AXIS_GROUP	User's FB isn't binding axis in axis group
0x0080000B	MC_ERROR_ID_FB_BIND_AXIS_IS_ERROR	The bound axis has error
0x0080000C	MC_ERROR_ID_FB_MINE_AXIS_IS_ERROR	Fb's planning axis has error
0x0080000D	MC_ERROR_ID_FB_NOT_BIND_MASTER	FB does not bind master axis
0x0080000E	MC_ERROR_ID_FB_NOT_BIND_SLAVE	FB does not bind slave axis
0x0080000F	MC_ERROR_ID_FB_CAN_NOT_BIND_ENCODERAXIS	FB can't bind encoder axis
0x00800010	MC_ERROR_ID_FB_NOT_BIND_TRIGGER_INFO	FB does not bind trigger information
0x00800011	MC_ERROR_ID_FB_NOT_SUPPORTED_TRINUM	Bound trigger number is not supported
0x00800012	MC_ERROR_ID_FB_CAN_NOT_CHANGE_WHILE_RUN	FB' bound parameter cannot change at running
0x00800013	MC_ERROR_ID_FB_RANGE_SET_ERROR	FB parameter range has error. Maximum value may be less than minimum value.
0x00800014	MC_ERROR_ID_FB_MASTER_SLAVE_ARE_SAME	FB master and slave axis are consistent
0x00800015	MC_ERROR_ID_FB_MASTER_AUXILIARY_ARE_SAME	FB master and auxiliary axis are consistent
0x00800016	MC_ERROR_ID_FB_AUXILIARY_SLAVE_ARE_SAME	FB auxiliary and slave axis are consistent

0x00800017	MC_ERROR_ID_FB_MASTER_SLAVE_NOT_ASCENDING	FB master and slave id are not in ascending order
0x00800018	MC_ERROR_ID_FB_POSITIVE_AND_NEGATIVE_IS_TURE	FB enablepositive and enablenegative is true
0X00800019	MC_ERROR_ID_FB_DOES_NOT_SUPPORTED_THIS_CURVE	FB does not support this curve.
0X00800020	MC_ERROR_ID_FB_CAN_NOT_SUPERIMPOSED	This instruction cannot be executed and cannot be superimposed
0X00800021	MC_ERROR_ID_FB_TRIGGERED_BUT_MISS_WINDOW	
General axis group error		
0x00900000	MC_ERROR_ID_AXIS_GROUP_GENERAL_ERROR	Group general error
0x00900001	MC_ERROR_ID_AXIS_GROUP_ADD_QUEUE_ERROR	Group fb add error
0x00900002	MC_ERROR_ID_AXIS_GROUP_REMOVE_ERROR	Group remove logic axis error
0x00900003	MC_ERROR_ID_AXIS_GROUP_ENABLE_ERROR	Group enable error
0x00900004	MC_ERROR_ID_AXIS_GROUP_DISABLE_ERROR	Group disable error
0x00900005	MC_ERROR_ID_AXIS_GROUP_STOP_ERROR	Group stop error
0x00900006	MC_ERROR_ID_AXIS_GROUP_AXIS_INSINGLEMOVE	Group is to run, but axis is in single axis controlled
0x00900007	MC_ERROR_ID_AXIS_GROUP_AXIS_INGROUPMOVE	Axis is bound in axis group
0x00900008	MC_ERROR_ID_AXIS_GROUP_IS_UNHEALTHY	Group's any axis is error or any error in group
0x00900009	MC_ERROR_ID_AXIS_GROUP_CAN_NOT_SETPOT	Group can't set position
0x0090000A	MC_ERROR_ID_AXIS_GROUP_INVALID_STATE_TRANS	Group is to switch to the unallowed state
0x0090000B	MC_ERROR_ID_AXIS_GROUP_NOT_BIND_AXIS	Group does not bind axis
0x0090000C	MC_ERROR_ID_AXIS_GROUP_IS_DISABLED	Group is disabled
Single axis planner error		
0x00A00000	MC_ERROR_ID_PLANNER_DIST_IS_ZERO	Planner position is 0
0x00A00001	MC_ERROR_ID_PLANNER_SPPED_IS_ZERO	Planner velocity is 0
0x00A00002	MC_ERROR_ID_PLANNER_ACC_DES_IS_ZERO	Planner acceleration is 0
0x00A00003	MC_ERROR_ID_PLANNER_JERK_IS_ZERO	Group general error
0x00A00004	MC_ERROR_ID_PLANNER_CURRENT_DIFF_LAST	Planner synchronization error, current position is inconsistent with the last position
0x00A00005	MC_ERROR_ID_PLANNER_SPPED_OVER_LIMIT	Planner input velocity is over limit
0x00A00006	MC_ERROR_ID_PLANNER_ACC_DES_OVER_LIMIT	Planner input acceleration and deceleration is over limit
0x00A00007	MC_ERROR_ID_PLANNER_JERK_OVER_LIMIT	Planner input Jerk is over limit
0x00AFFFFF	MC_ERROR_ID_PLANNER_SG_CULCULATE	Planner single axis calculate error
Axis group planner error		
0x00B00000	MC_ERROR_ID_PLANNER_SP_SAME_START_END	Planner space curve' start and end are the same
0x00B00000	MC_ERROR_ID_PLANNER_SP_NOT_IN_SAME_PLANE	Planner space curve is not in the same plane
0x00BFFFFF	MC_ERROR_ID_PLANNER_SP_CULCULATE	Planner space curve calculate error
Cam Gear Ratio		
0x00C00000	MC_ERROR_ID_AXIS_MASTER_AND_SLAVE_IS_SAME	Planner master and slave is the same

0x00C00001	MC_ERROR_ID_AXIS_MAX_ACC_OR_DEC_OR_VEL_IS_ZERO	Planner axis's max acc/dec/vel is zero
0x00C00002	MC_ERROR_ID_CAM_TABLE_IS_INEXISTENCE	Planner the camtable is inexistence
0x00C00003	MC_ERROR_ID_CAM_NODENUM_OVER_NODE_ SIZE	Planner the cam node num is over actual cam node size
0x00C00004	MC_ERROR_ID_CAM_NODENUM_OR_NODESIZE_IS_ERROR	Planner the cam node num or cam node size is error
0x00C00005	MC_ERROR_ID_CAM_FIRST_NODE_POS_NON_ ZERO	Planner the cam first node's master position or slave position non-zero
0x00C00006	MC_ERROR_ID_CAM_NODE_PARAMETER_IS_ ERROR	Planner the cam node parameter is error
0x00C00007	MC_ERROR_ID_CAM_UPDATE_CAM_TABLE_IS_ ERROR	Planner the update cam table is error
0x00C00008	MC_ERROR_ID_CAM_SAVE_CAM_TABLE_IS_ ERROR	Planner the save cam table is error
0x00C00009	MC_ERROR_ID_CAM_GET_AXIS_VEL_AND_POS_ ERROR	Planner get master and slave vel/pos is error
0x00C0000A	MC_ERROR_ID_BUFFER_MODE_IS_NOT_ ABORTING	Planner the buffer mode is not aborting
0x00C0000B	MC_ERROR_ID_CAM_AXIS_SCALING_IS_ZERO	Planner the cam masterscaling or slavescaling is zero
0x00C0000C	MC_ERROR_ID_CAM_TABLE_IS_NOT_BIND	Planner the cam table is not bind
0x00C0000D	MC_ERROR_ID_THE_AXIS_IS_NOT_IN_CAM_STATE	Planner the axis is not in the cam state
0x00C0000E	MC_ERROR_ID_THE_AXIS_IS_NOT_IN_GEAR_ STATE	Planner the axis is not in the gear state
0x00C0000F	MC_ERROR_ID_CAM_IN_PARAMETER_IS_ERROR	Planner the camin parameter is error
0x00C00010	MC_ERROR_ID_GEAR_IN_PARAMETER_IS_ERROR	Planner the gearin parameter is error
0x00C00011	MC_ERROR_ID_GEAR_IN_POS_PARAMETER_IS_ ERROR	Planner the gearinpos parameter is error
0x00C00012	MC_ERROR_ID_CAM_TABLE_IS_UPDATE	Planner the cam table is updating
0x00C00013	MC_ERROR_ID_GET_CAM_TABLE_POS_IS_ERROR	planner the get cam table position is error
0x00C00014	MC_ERROR_ID_CAM_TABLE_PHASE_IS_OUT_OF_ RANGE	planner the get cam table phase is out of range
0x00C00015	MC_ERROR_ID_CAM_TABLE_NODE_IS_NOT_ ADJOIN	planner the cam table node is not adjoin

9 High Speed counter

9.1 High-speed counter (Encoder axis) Introduction

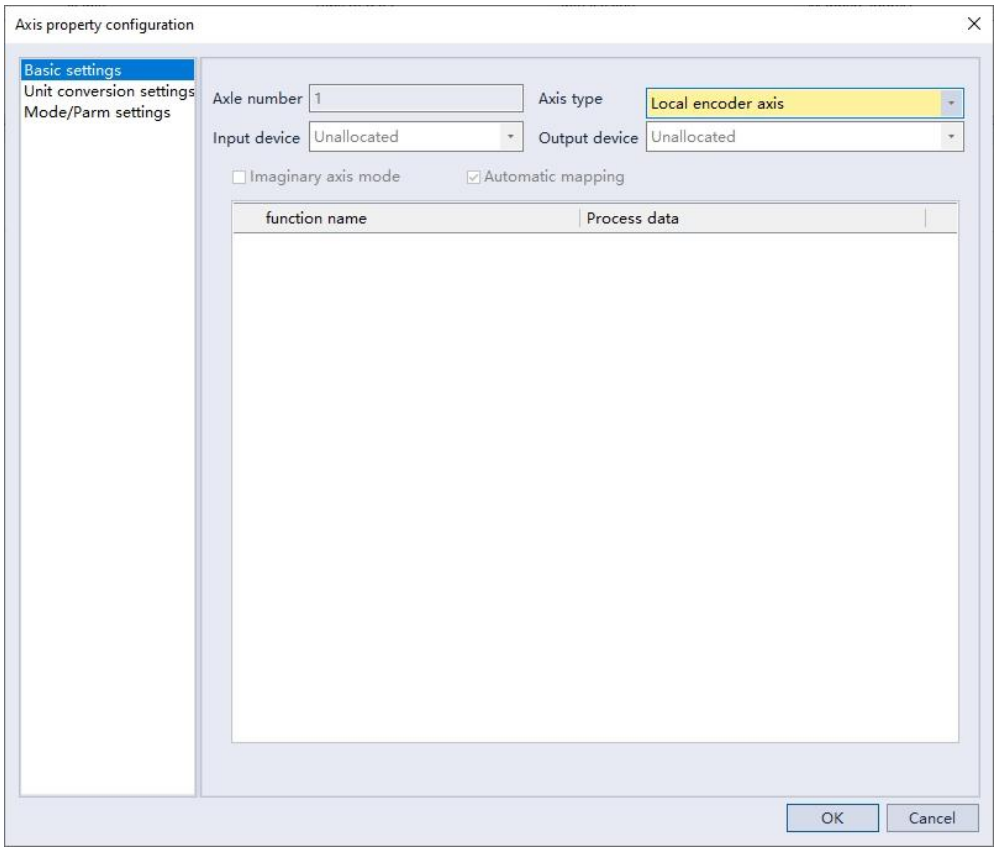
9.1.1 High-speed counter axis introduction

The high-speed counter is implemented in the form of encoder axis in LX6 series PLC, subsequent collectively referred to as the encoder axis. It supports 4-axis 32-bit high-speed counter, AB phase 1/2/4 times frequency, pulse/direction and single-phase counting. The counting signal support external pulse input or internal 100us/1ms/10ms clock counting, and supports touch probe and high-speed comparison output functions.

9.1.2 Create encoder axis

Step: "Project Management"->"Motion control axis"->"Right click"->"Add Axis"

"Axis type" select "Local encoder Axis"



9.1.3 Encoder axis user unit and conversion

The encoder axis decoding the encoder signal by pulse unit, and the encoder axis instruction uses common measurement units such as millimeters, degrees, inches, etc., which we call it user unit. The number of pulses can be converted into user unit through conversion, and user unit can be defined as equipment-related units (millimeters, revolutions, etc.) according to actual applications.

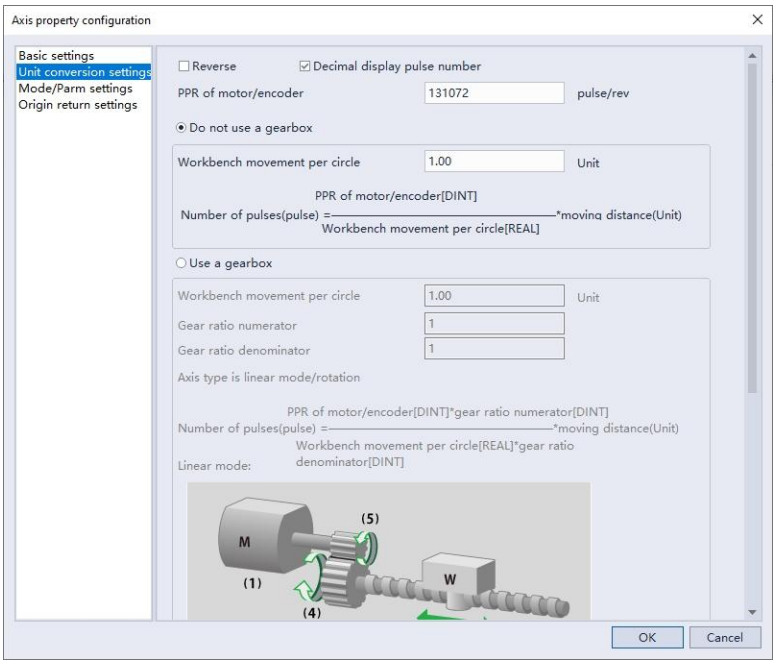
The parameters to be set for unit conversion are as follows:

Parameter name	Function
Pulses of motor/encoder for one rotation	Set pulses of motor for one rotation according to encoder resolution
With gearbox or not	Specify whether or not to use gearbox
Movement amount of motor/encoder for one rotation	Workpiece movement amount of motor for one rotation without gearbox
Movement amount of worktable for one rotation	Workpiece movement amount of motor for one rotation with gearbox
Gear ratio numerator	Set the gear ratio of workpiece side
Gear ratio denominator	Set the gear ratio of motor side

In gear drive, the drive axis is usually called input axis, and the driven axis is called output axis. Therefore, the numerator of the gear ratio represents the speed of the input axis (usually also represents the gear driven by the input axis), and the denominator represents the speed of the output axis (usually also represents the gear driven by the output axis). If the gear ratio is a: b, it means that for a revolution of the input axis, the output axis will rotate b revolutions.

For example, servo motor drives worktable to move by connecting screw rod through reducer, counts encoder to feedback worktable position by PLC controller, and counts encoder pulse by counter, taking pulse as unit; If the counter axis represents the table position, it is in millimeters. Therefore, in the program, the user unit of the counter axis is uniformly expressed in the user unit.

The conversion relationship between user unit and pulse is as follows:



The number of pulses for one revolution of the motor/encoder: In the input box, 16# indicates a hexadecimal number. If the encoder output 10000 pulses per revolution, the corresponding hexadecimal value is 2710, then enter 16#2710.

Working stroke setting: The working stroke can be set without or with a gearbox.

Without gearbox:

When the gearbox is not used, the conversion formula from user unit to pulse unit is as follows:

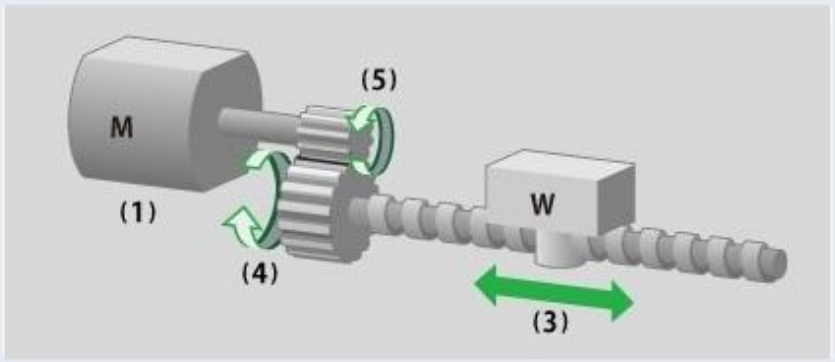
$$\text{Number of pulses(pulse)} = \frac{\text{PPR of motor/encoder[DINT]}}{\text{Workbench movement per circle[REAL]}} * \text{moving distance(Unit)}$$

For example, if the encoder rotates one circle and the corresponding working axis rotates one circle, the user unit is one revolution, then the working stroke of the motor/encoder one circle rotation can be set to 1.

·With gearbox:

The calculation formula from user unit to pulse unit is as follows:

Linear mode:



Wherein (1) is a servo motor, (3) is a workpiece, (4) is the gear ratio denominator, and (5) is the gear ratio numerator.

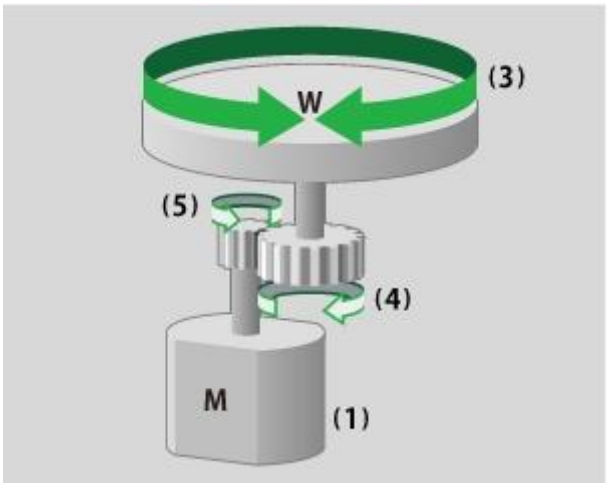
The calculation formula from user unit to pulse unit is as follows:

For instance: The servo motor drives the worktable to move through the reducer connected by the screw. The working stroke of the screw rotating once is 5mm, and the reduction ratio is 20:10. The corresponding input axis (motor) needs to rotate 20 turns every time the output axis rotates once. Therefore, the gear ratio numerator can be set as 20, and the gear ratio denominator can be set as 10. Before the motor rotation is transmitted to the output axis (screw), the amplification of the gear can ensure the required distance for the worktable to move. The settings are as follows:

Workbench movement per circle	5.00	Unit
Gear ratio numerator	10	
Gear ratio denominator	20	
Axis type is linear mode/rotation		

✎**Note:** Besides reduction ratio and gear ratio, the power of the servo motor should be determined according to the performance parameters of the servo motor and the load-characteristic of the worktable, and the required motor speed and torque should be calculated according to the reduction ratio and gear ratio to ensure that the system can work properly.

Typical working condition in rotation mode is shown in the following figure:



Wherein (1) is a servo motor, (3) is a workpiece, (4) is the gear ratio denominator, and (5) is the gear ratio numerator.

The calculation formula from user unit to pulse unit is as follows:

Linear mode:

$$\text{Number of pulses(pulse)} = \frac{\text{PPR of motor/encoder[DINT]*gear ratio numerator[DINT]}}{\text{Workbench movement per circle[REAL]*gear ratio denominator[DINT]}} * \text{moving distance(Unit)}$$

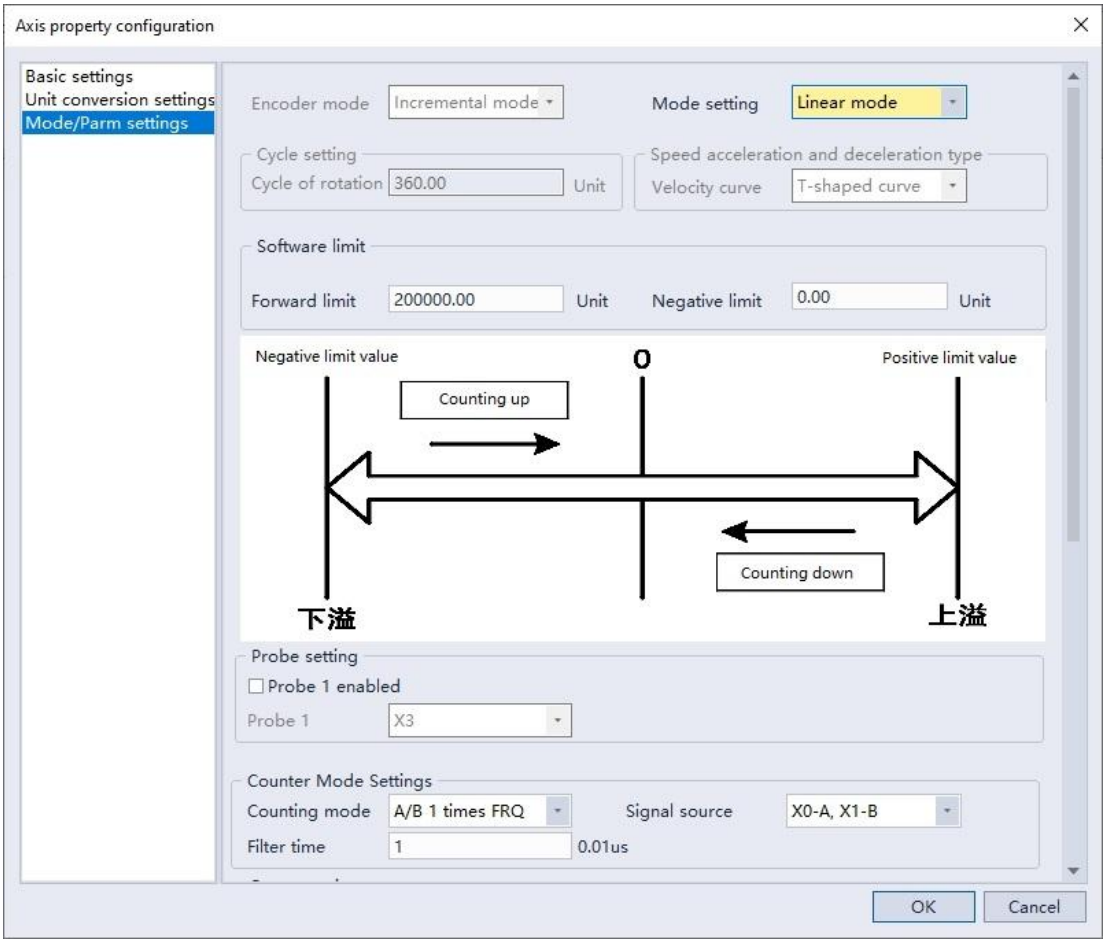
9.1.4 Work mode setting

Linear mode

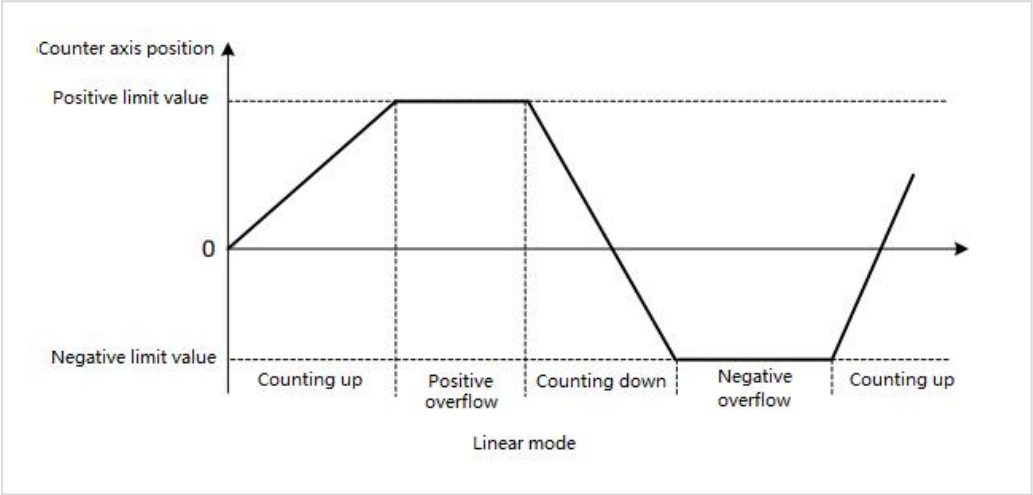
The position of the encoder axis changes between the negative limit value and the positive limit value. After the position of the encoder axis reaches the limit value, if continue to input the same direction pulse, the counter axis will report an overflow error, and the position of the counter axis will remain unchanged. After encoder report an overflow error, if input reverse pulse, the encoder axis will reverse count and overflow error removed.

In linear mode, the negative and positive position limits of the encoder axis can be set in the interface, and the position unit is the user unit. The negative limit value must be less than or equal to 0, and the positive limit value must be greater than or equal to 0.

Since the high-speed counter is a 32-bit counter, the negative limit value and the positive limit value must be in the 32-bit integer range [-2147483648, 2147483647] after being converted into pulse units.

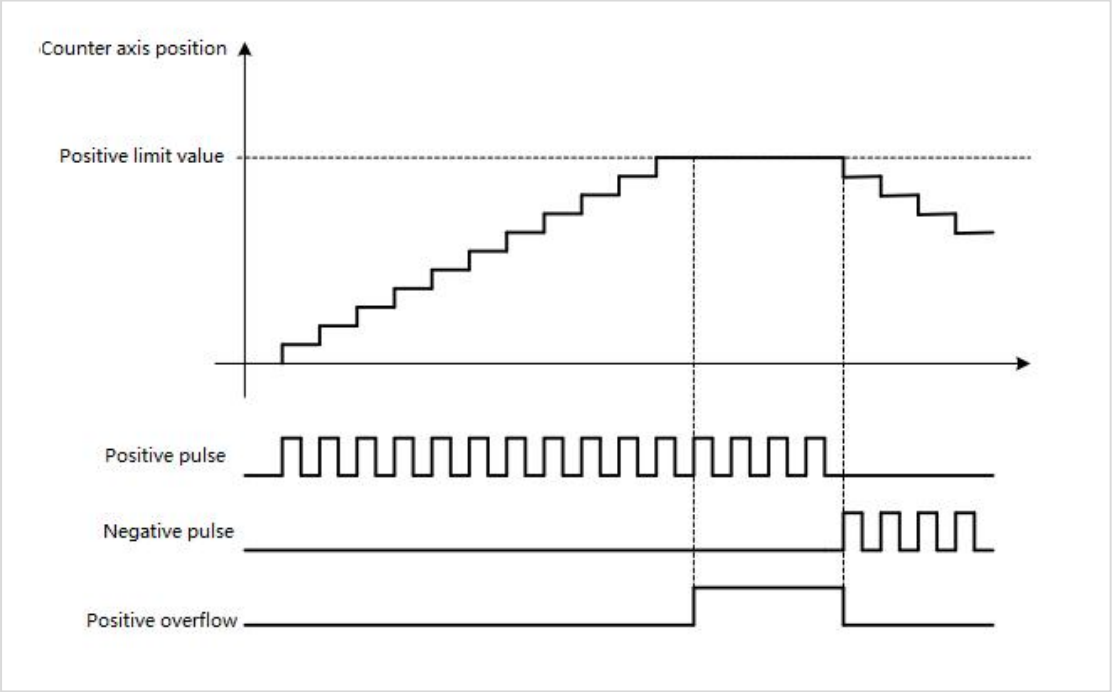


In linear mode, the high-speed counter works in the closed range of [negative limit value, positive limit value]. When the direction is negative, the count value decreases in the negative direction, and after reaching the negative limit value, the count value no longer decreases; When the direction is positive, the count value increases in the positive direction, and after reaching the positive limit value, the count value no longer increases.



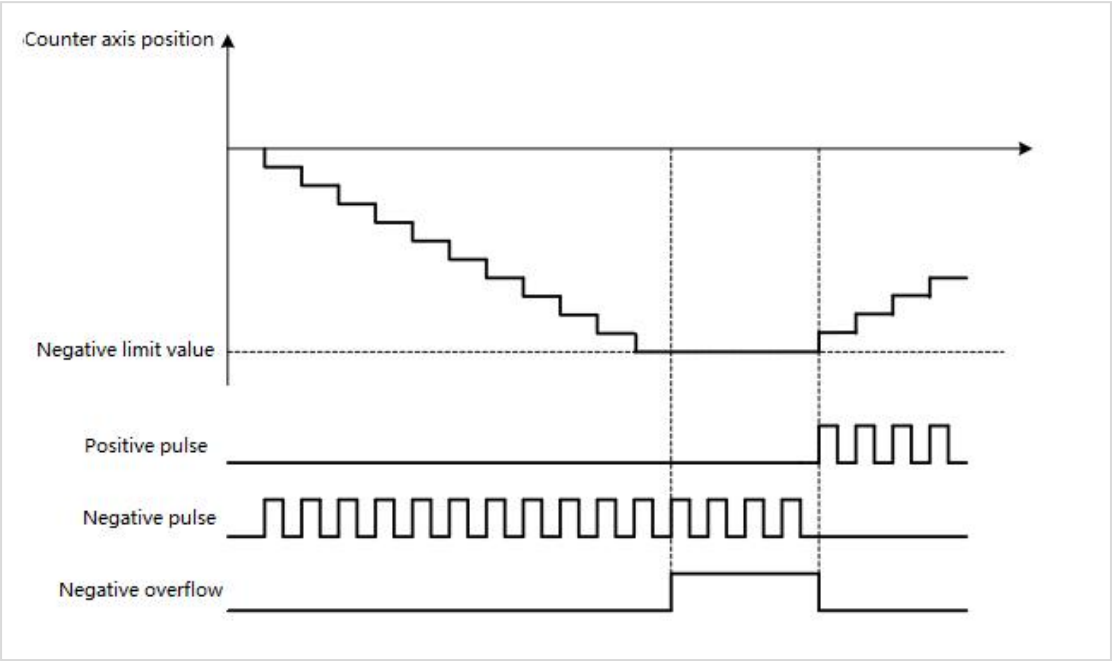
Forward pulse counting

In linear mode, input positive pulse, after the encoder axis position increment count reaches the limit value, if continue to input positive pulse will cause the encoder axis reports a positive overflow error, and the encoder axis position value remains unchanged. Input negative pulse, encoder axis position will count down and positive overflow error will be removed.



Reverse pulse counting

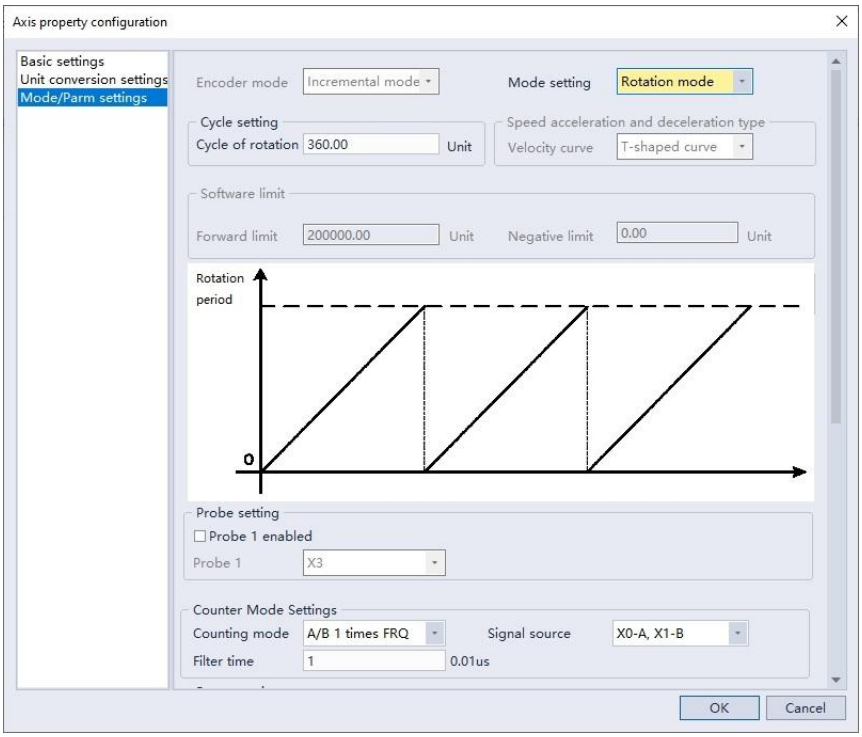
In linear mode, input negative pulse, after the encoder axis position countdown reaches the limit value, if continue to input negative pulse will cause the encoder axis reports a negative overflow error, and the encoder axis position value remains unchanged. Input positive pulse, encoder axis position will count up and negative overflow error will be cleared.



Rotation mode

The position of the encoder axis changes cyclically in the rotation period. When counting up, the position of the encoder axis reaches the maximum of the rotation period and then becomes 0. When counting down, the position of the encoder axis reaches 0 and then decreases from the maximum of the rotation period. In rotation mode, the rotation period of the counter axis can be set in the interface, and the period unit is the user unit.

Since the high-speed counter is 32-bit counter, the rotation period must be in the 32-bit integer range [-2147483648.2147483647, 2147483647] after being converted into pulse units.

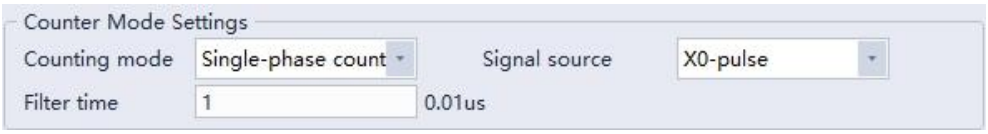


9.1.5 Setting counter parameters

Overview

Local encoder axis supports multiple signal counting modes, A/B phase (1/2/4 times frequency), pulse + direction, single phase counting.

Signal source: According to different counting modes, user can select different signal sources.



The signal source input ports supported in each counting mode are shown in the following table. Different local encoder axis can select the same signal source input port:

Counting mode	AB phase	Pulse + Direction	Single phase counting
X0	A phase	Pulse	Pulse
X1	B phase	Direction	Pulse
X2	A phase	Pulse	Pulse
X3	B phase	Direction	Pulse
X4	A phase	Pulse	Pulse
X5	B phase	Direction	Pulse
X6	A phase	Pulse	Pulse
X7	B phase	Direction	Pulse
100us	×	×	Pulse
1ms	×	×	Pulse
10ms	×	×	Pulse

When using touch probe or preset function, encoder axis is divided into two groups, namely, X0 ~ X3 is counting group 1 and X4 ~ X7 is counting group 2. The same signal source in counting group can only be used in one encoder axis, that is, when the touch probe function of counting group 1 is enabled, X3 is occupied and cannot be used as signal source of other counting modes. Touch probes and preset functions cannot be used by internal signal sources.

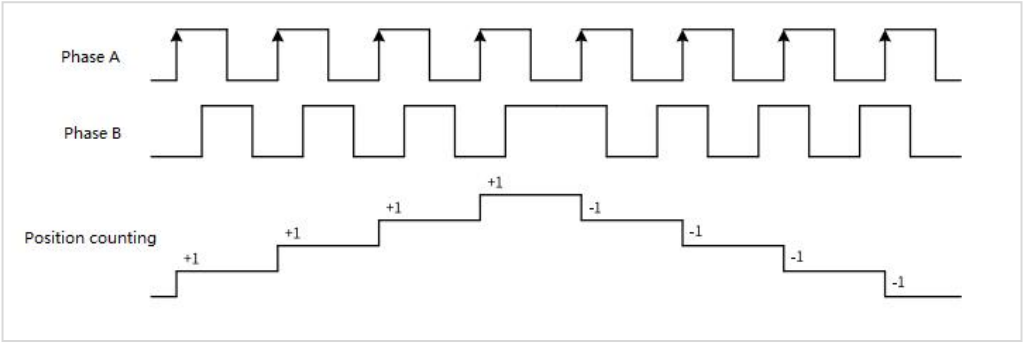
Counting mode	AB phase	Pulse + Direction	Single phase counting
X0	A phase	Pulse	Pulse
X1	B phase	Direction	×
X2	Touch probe	Touch probe	Touch probe
X3	Preset	Preset	Preset
X4	A phase	Pulse	Pulse
X5	B phase	Direction	Pulse
X6	Touch probe	Touch probe	Touch probe
X7	Preset	Preset	Preset
100us	×	×	Pulse
1ms	×	×	Pulse
10ms	×	×	Pulse

A/B phase mode

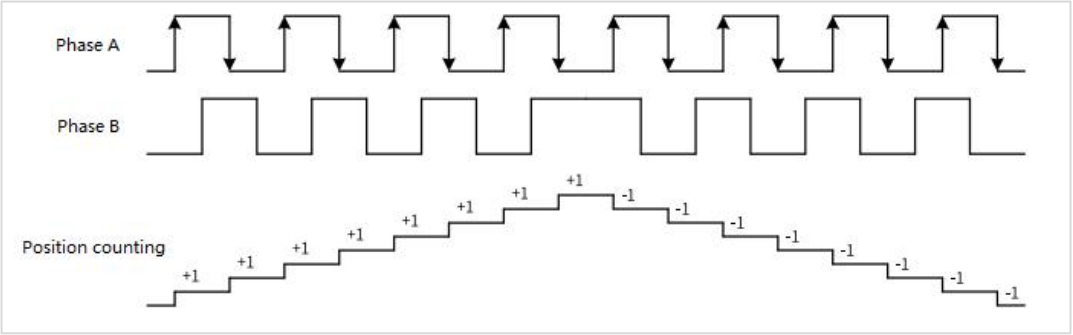
In the A/B phase mode, the encoder generates two quadrature phase pulse signals with a phase difference of 90 degrees, namely A-phase signal and B-phase signal. When the A-phase signal advances the B-phase signal, the counter increases the count; When the B-phase signal advances the A-phase signal, the counter counts down.

A/B phase pulse can be set to work in 1 time frequency mode, 2 times frequency mode or 4 times frequency mode.

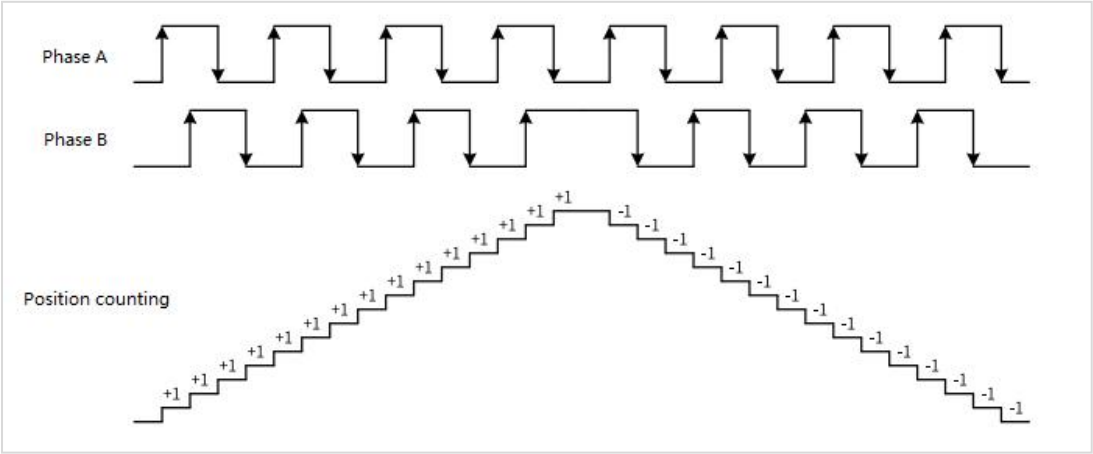
In A/B phase 1 time frequency mode, only the rising edge of phase A pulse is counted, as shown in the following figure:



In A/B phase 2 times frequency mode, the rising/falling edge of phase A pulse is counted, as shown in the following figure:

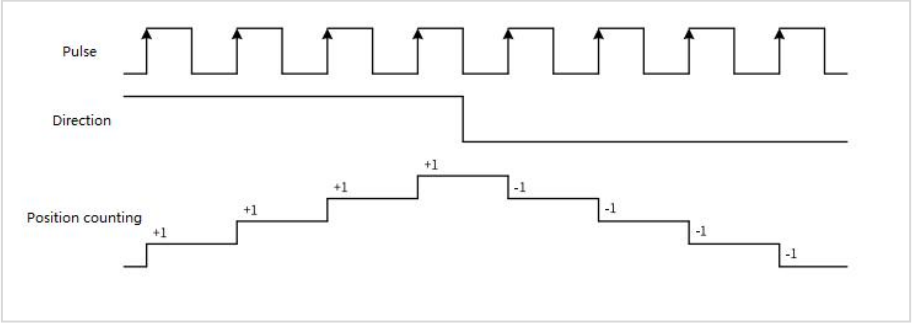


In A/B phase 4 times frequency mode, the rising/falling edge of phase A pulse is counted, as shown in the following figure:



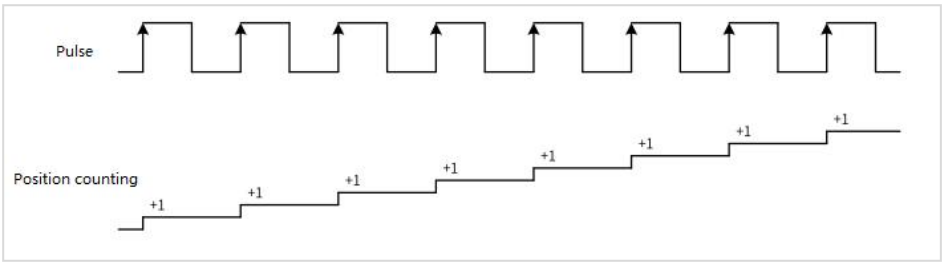
Pulse + direction mode

In this mode, when the direction signal is ON, the high-speed counter increases the count of the pulse signal, and when the direction signal is OFF, the high-speed counter decreases the count of the pulse signal, as shown in the following figure:



Single-phase counting mode

In this mode, the high-speed counter increases the count of the pulse signal, on the rising edge of the input pulse, the position count is increased by 1.



Touch probe setting

Each encoder axis supports one external input to latch the current value of the counter to realize the touch probe function. The counting group 1 fixed use terminal X3, and the counting group 2 fixed use terminal X7.

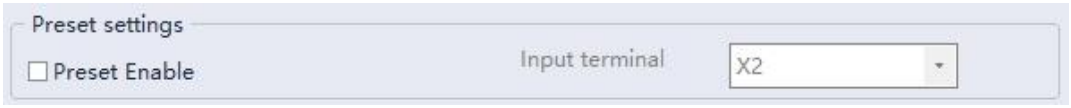
After the touch probe is enabled, the current count value will be saved after the touch probe is triggered, and the touch probe position of the counter axis will be read through the HC_TouchProbe function block instruction.



5.5 Preset terminal input

By checking the preset enable to enable the external input preset counter value, the counting group one fixed use terminal X2, and the counting group two fixed use terminal X6.

After the preset function is enabled, the encoder axis position is preset by external input through HC_Preset function block instruction.



Compare output terminal settings

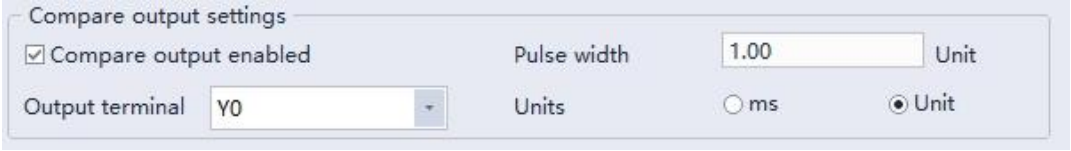
After "comparison output enabled" is checked, hardware can output when comparison is equal, which can be realized without software processing, with high real-time performance and output response up to microsecond level.

● After the comparison output function is checked and cooperated with function block instruction, the output is controlled to be ON by hardware circuit when the comparison is equal, Y0~Y7 can be arbitrarily selected as the output terminal, and the pulse width of the output to be ON can be selected as time unit or user unit.

● Each local encoder axis is equipped with a comparison output function, which can configure the input terminal and output pulse width according to the demand.

● After the configuration is completed, through HC_Compare, HC_ArrayCompare and HC_StepCompare function block instructions, the axis position comparison output is realized.

● When ms is selected as the unit, the time setting range is 0.1~6553.5ms. When Unit is selected, the set value should be in the range of 1~65535 after being converted into pulse Unit.

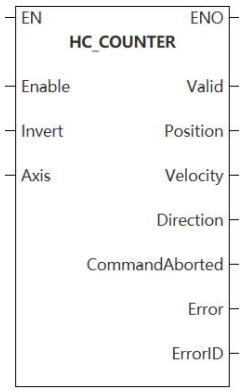


The comparison output is directly output through the hardware control port, not through the software processing, so the status of comparison output cannot be displayed through the Y device in the program. The Y device and the comparison output are controlled as OR relationship to the output port. If the Y device is continuously controlled as ON state, the actual port output is kept as ON.

9.2 HC Axis control

9.2.1 HC_COUNTER high speed counter enable

HC_COUNTER



This function block is used to control the start or stop of the high speed counter.

Content, range and data type

Instruction	Parameter	Variable type	Can it be empty	Range	Description
HC_COUNTER	Enable	BOOL	No	TRUE FALSE	High speed counter enable
	Invert	BOOL	Yes	TRUE FALSE	Negative counting
	Axis	WORD	No	0~65535	Axis name/axis number: specify the local encoder axis to operate
	Valid	BOOL	Yes	TRUE FALSE	Valid status: entering the counting status is ON
	Position	LREAL	Yes	-	Current position: user unit
	Velocity	LREAL	Yes	-	Current velocity: user unit
	Direction	BOOL	Yes	TRUE FALSE	Counting direction (TRUE: Forward, FALSE: Reverse)
	CommandAborted	BOOL	Yes	TRUE FALSE	Abort flag
	Error	BOOL	Yes	TRUE FALSE	Error
	ErrorID	DWORD	Yes	Check [ErrorID] for details	Error code

Device used

Instruction	Parameter	Devices																					Index modification
		X	Y	M	S	SM	T (Bit)	HSC (Bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
HC_COUNTER	Enable	●	●	●	●	●																		
	Invert	●	●	●	●	●																		
	Axis															●	●							
	Valid		●	●	●	●																		
	Position																							
	Velocity																							
	Direction		●	●	●	●																		
	CommandAborted		●	●	●	●																		
	Error		●	●	●	●																		
	ErrorID																							

Function

The HC_COUNTER function block can count the current counter axis position, measure the current velocity in real time and display the current counting direction.

The counter axis position is set only according to the current counting mode and varies within the range of the counter axis mode, and the position unit is Unit.

✎Note:

(1) Update the current axis position, velocity and counting direction if and only if the function block Enable is enabled.

(2) If two function blocks are counting at the same time in the scanning program and the encoder axis parameter Axis is specified to be the same, the execution of the first enabled function block will be interrupted by the second enabled function block. The former commandAborted is set to True, and the valid status Valid is set to False.

(3) When Function Block generates an error, Enable must be re-enabled to clear the error. Otherwise, the function block does not execute counting.

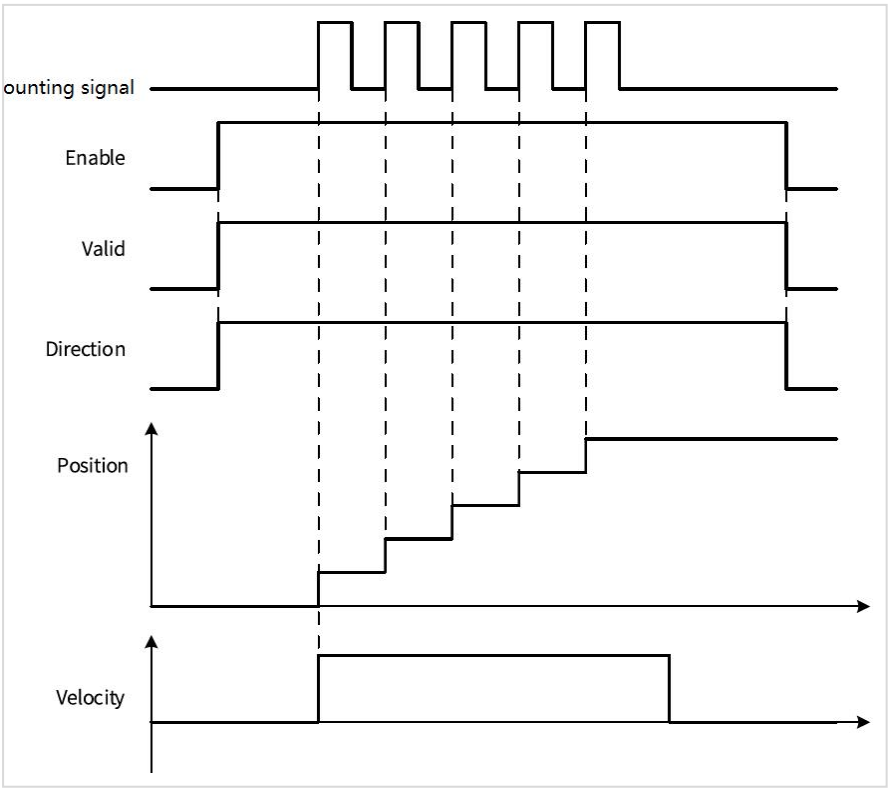
Invert parameter (Negative counting)

The counting direction of the counter is set by the Invert parameter. The definition of counting directions for different counting modes are shown in the following figure. When changing the setting of Invert, it will take effect directly if the function block is enabled.

Invert	Phase A/B	Pulse + Direction	CW/CCW	Single phase counting
FALSE	Phase A is ahead of Phase B to count up. Phase B is ahead of Phase A to count down.	Direction signal Vil count down Direction signal VIH count up	Phase A counting up. Phase B counting down.	Counting up
TRUE	Phase A is ahead of Phase B to count down. Phase B is ahead of Phase A to count up.	Direction signal Vil count up Direction signal Vol count down	Phase A counting down. Phase B counting up.	Counting down

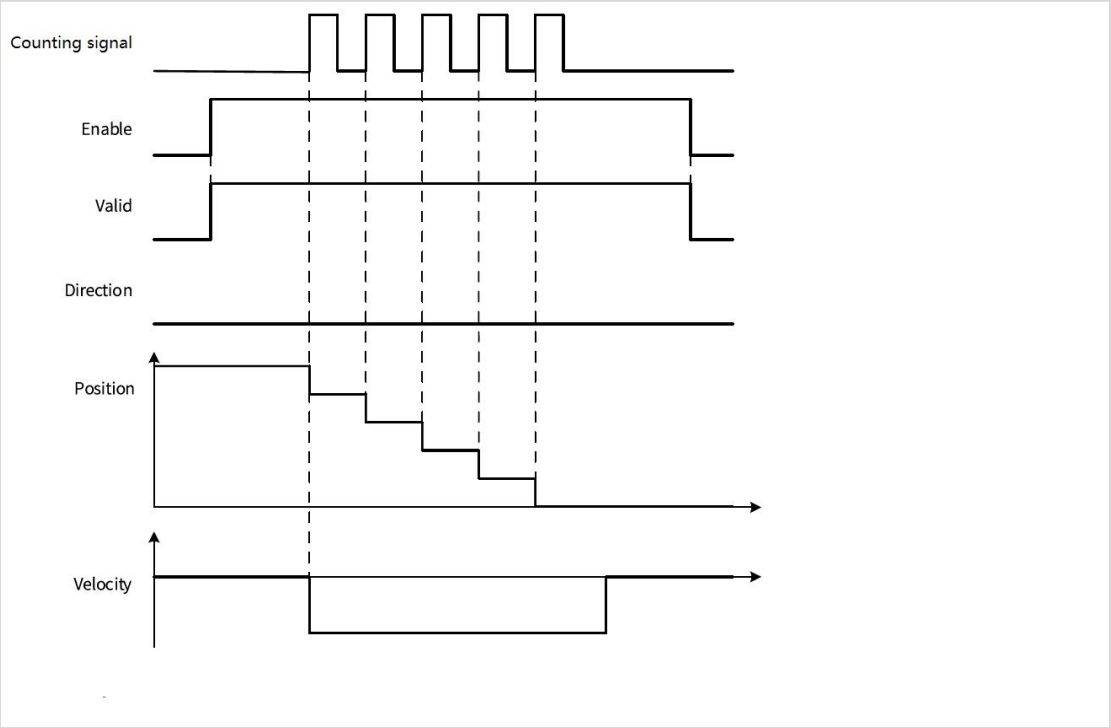
Description

Take the "pulse + direction" mode as an example. When the direction signal = ON/Invert=FALSE or the direction signal = OFF/Invert=TRUE, the counter is counting up, as shown in the following figure:



Take the "pulse + direction" mode as an example. When the direction signal = ON/Invert=TRUE or the direction signal =

OFF/Invert=FALSE, the counter is counting down, as shown in the following figure:



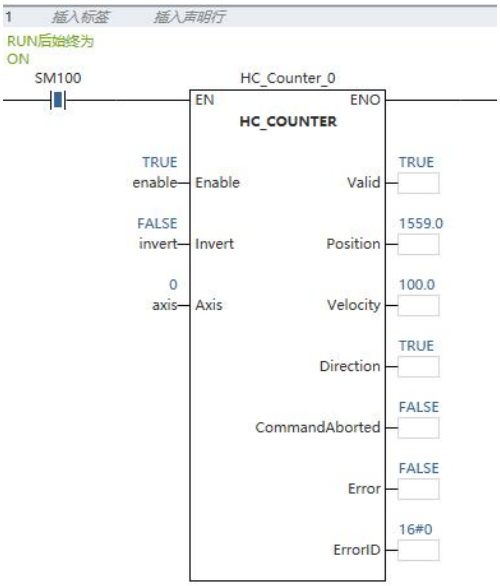
Error code

Error code	Content
0x00400005	Counting position exceeds the positive limit value.
0x00400006	Counting position exceeds negative limit value.
0x00700005	Axis undefined or invalid axis number
0x00800008	Modify the axis number when the function block is running.

Example

Application example

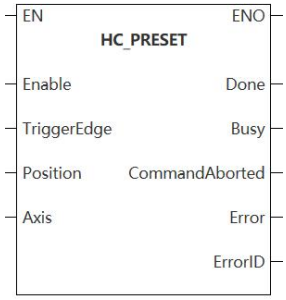
The counter axis speed is the real-time speed, and the speed unit is Unit/s. The minimum speed that can be measured by the counter axis is the speed corresponding to one pulse of the counter in 1s. For instance, when one pulse of the counter corresponds to 0.01 Unit, the minimum speed that can be measured is 0.01 Unit/s.



For example: When the current function block Enable in the above figure is ON, update the current encoder axis position and velocity, display the counting direction, and measure the current input pulse speed in the AXIS_0 is 100 Unit/s and the current position is 1559.0 Unit.

9.2.2 HC_PRESET

High speed counter preset value



HC_PRESET

Set the counter value to the preset value according to the preset trigger condition set.

Content, range and data type

Instruction	Parameter	Variable type	Can it be empty	Range	Description
HC_PRESET	Enable	BOOL	No	TRUE FALSE	High speed counter preset value function block enabled
	TriggerEdge	WORD	Yes	0~1	Trigger edge 0: Instruction Enable rising edge triggers 1: External X rising edge triggers
	Position	LREAL	No	-	Preset position: user unit
	Axis	WORD	No	0~65535	Axis name/axis number: specify the local encoder axis to operate
	Done	BOOL	Yes	TRUE FALSE	Completed
	Busy	BOOL	Yes	TRUE FALSE	Executing
	CommandAborted	BOOL	Yes	TRUE FALSE	Abort flag
	Error	BOOL	Yes	TRUE FALSE	Error
	ErrorID	DWORD	Yes	Check "ErrorID" for details	Error code

Device used

Instruction	Parameter	Devices																						Offset modification	Double word	Pulse Mode Expansion
		X	Y	M	S	SM	T (Bit)	HSC (Bit)	D. b	KnX	KnY	KnM	KnS	TC	D	R	SD	LC	HSC	K	H	E		[D]	DXX	XXP
HC_PRESET	Enable	●	●	●	●	●																				
	TriggerEdge															●	●									
	Position																									
	Axis															●	●									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Function

Enable HC_PRESET function block and the counter axis position is assigned according to the preset trigger condition.

TriggerEdge parameter

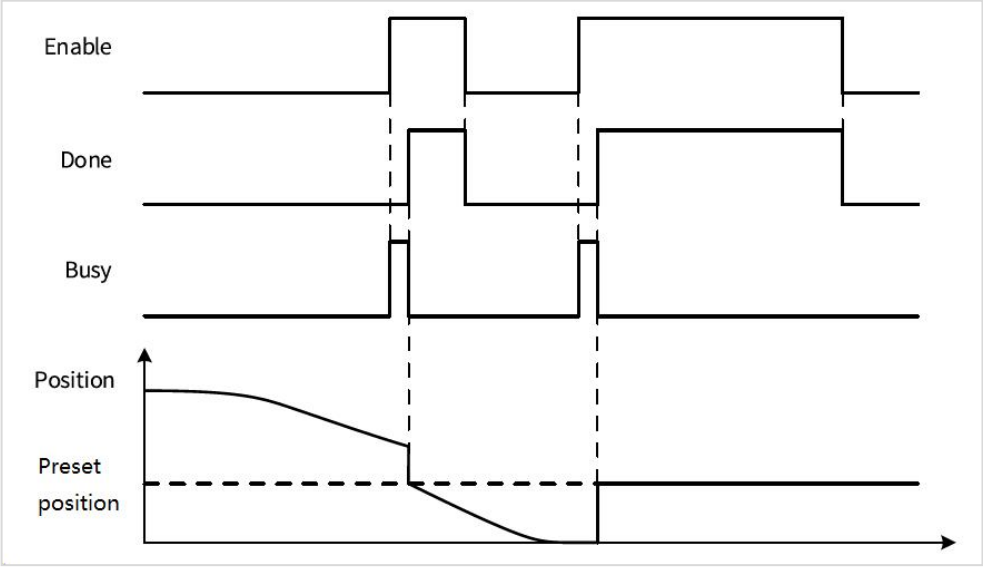
Select the rising edge of instruction enable or external X input to trigger preset function.

Parameter	Settings	Definition
TriggerEdge	0	Enable rising edge triggering
	1	External X rising edge trigger

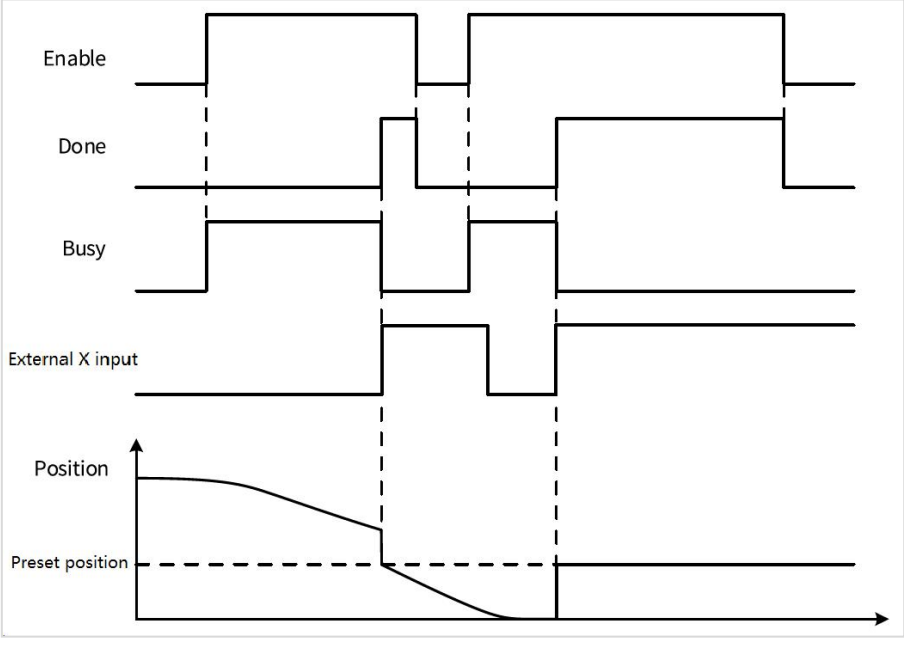
When selecting external X input trigger as preset condition, it needs to see preset function in counter parameter setting. Select input terminal and trigger condition, X2 and X6 for input terminal and rising edge trigger for trigger condition.

Description

The instruction Enable is triggered on the rising edge (TriggerEdge=0), and the instruction sequence diagram is shown in the following figure:



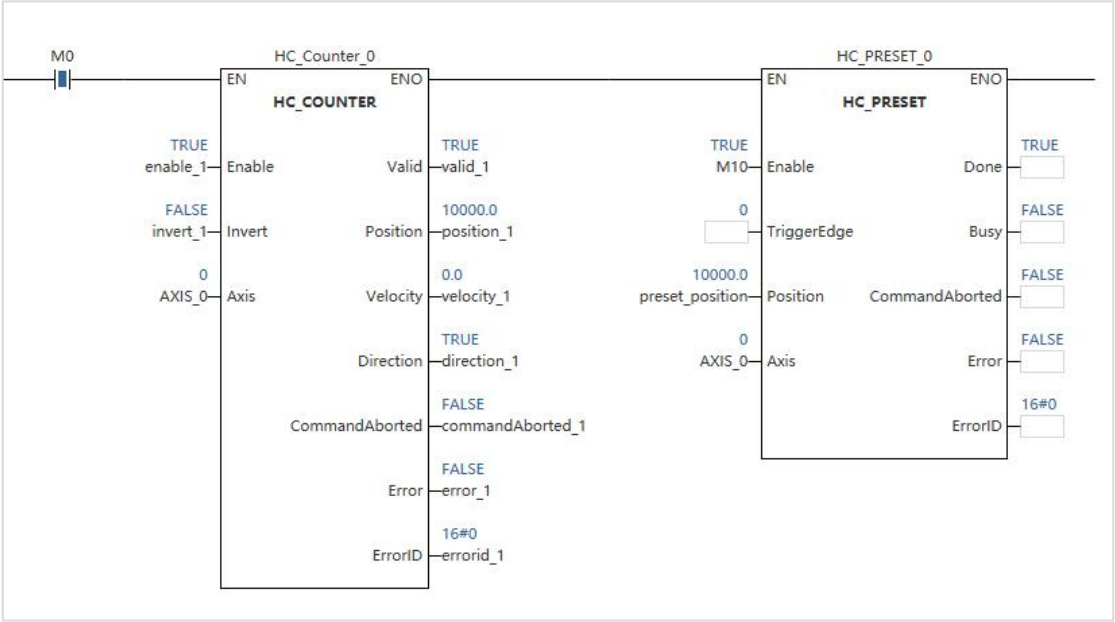
External X interrupt triggers. Take rising edge trigger (TriggerEdge=1) as an example, and the instruction sequence diagram is shown in the following figure:



Error code

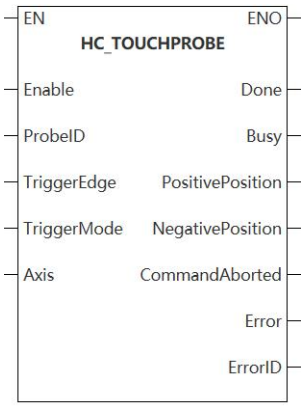
Error code	Content
0x007000000	Set the trigger edge to external X edge trigger, but the preset function of axis configuration interface is not enabled
0x00700005	Axis undefined or invalid axis number
0x00700008	High speed counter of local encoder axis is not enabled
0x00800006	Set the preset position out-of-limit value/set the trigger edge parameter out-of-range
0x00800008	Modify the axis number when the function block is running.

Application example



Set the preset trigger condition as Enable rising edge trigger (TriggerEdge=0). When the preset condition is met, modify the current HC_COUNTER axis position as the preset position. At this time, the current position read by the HC_COUNTER function block is the preset position.

9.2.3 HC_TOUCHPROBE High speed counter touch probe



According to the set trigger mode and conditions, record the counter value when the signal is triggered.

Content, range and data type

Instruction	Parameter	Variable type	Can it be empty	Range	Description
HC_TOUCHPROBE	Enable	BOOL	No	TRUE FALSE	High speed counter preset enable
	ProbeID	WORD	No	0	Touch probe ID 0: Touch probe 1
	TriggerEdge	WORD	No	1~3	Trigger edge 1: External X rising edge triggers 2: External X falling edge triggers 3: External X rising or falling edge triggers
	TriggerMode	WORD	Yes	0~1	Trigger mode 0: Single triggers 1: Continuous triggers
	Axis	WORD	No	0~65535	Axis name/axis number: specify the local encoder axis to operate
	Done	BOOL	Yes	TRUE FALSE	Completed
	Busy	BOOL	Yes	TRUE FALSE	Executing
	PositivePosition	LREAL	Yes	-	Rising edge latch position: user unit
	NegativePosition	LREAL	Yes	-	Falling edge latch position: user unit
	CommandAborted	BOOL	Yes	TRUE FALSE	Abort flag
	Error	BOOL	Yes	TRUE FALSE	Error
	ErrorID	DWORD	Yes	Check "ErrorID" for details	Error code

Device used

Instruction	Parameter	Devices																				Index modification
		X	Y	M	S	SM	T (Bit)	HSC (Bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
HC_TOUCHPROBE	Enable	●	●	●	●	●																		
	ProbeID															●	●							
	TriggerEdge															●	●							
	TriggerMode															●	●							
	Axis															●	●							
	Done		●	●	●	●																		
	Busy		●	●	●	●																		
	PositivePosition																							
	NegativePosition																							
	CommandAborted		●	●	●	●																		
	Error		●	●	●	●																		
	ErrorID																							

Function

Enable the HC_TOUCHPROBE function block instruction to latch the counter axis position value when the external input trigger condition is valid.

Each counter axis only supports one touch probe. When using, it needs to check the corresponding touch probe function in the counter parameter setting. Select the input terminal and trigger mode and condition. The input terminal can be set as X3 or X7.

Description

TriggerEdge parameter

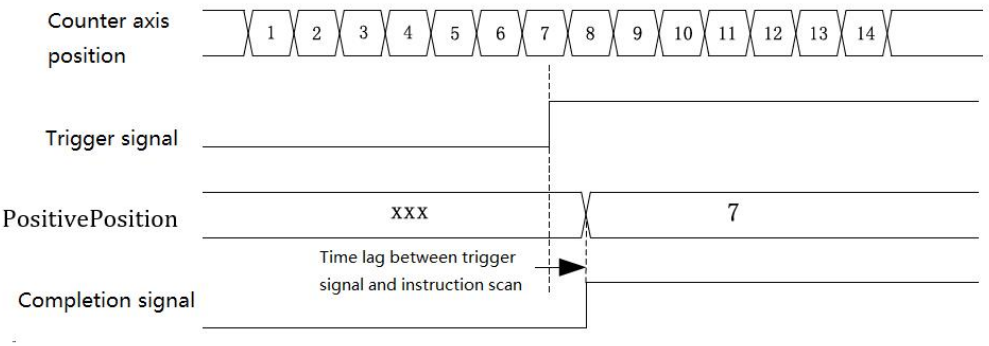
The touch probe trigger edge can be set. The rising edge trigger position is latched in the output parameter PositivePosition. The falling edge trigger position is latched in the output parameter NegativePosition.

Parameter	Settings	Definition
TriggerEdge	1	External X rising edge triggers
	2	External X falling edge triggers
	3	External X rising or falling edge triggers

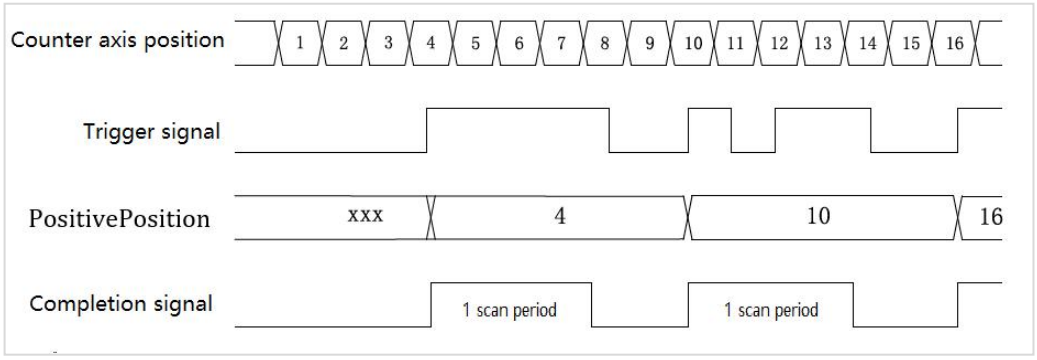
TriggerMode parameter

Single trigger and continuous trigger modes can be set.

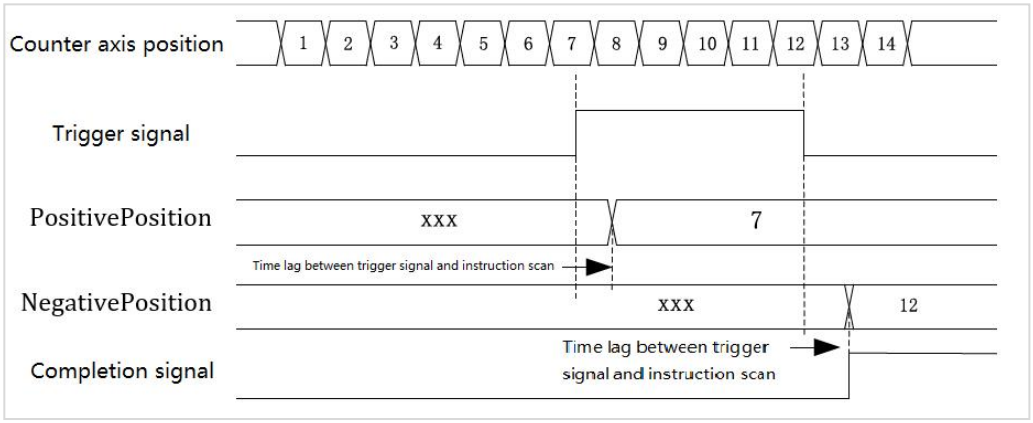
In single trigger mode, when the function block Enable is valid and external input trigger condition disappears, the counter axis position is latched and the completion signal is output. The touch probe position latches the counter axis position in real time according to the trigger edge, which is not affected by program execution. According to the scan period, the latch position is updated to the instruction output parameter when the program scanning executes to the latch instruction.

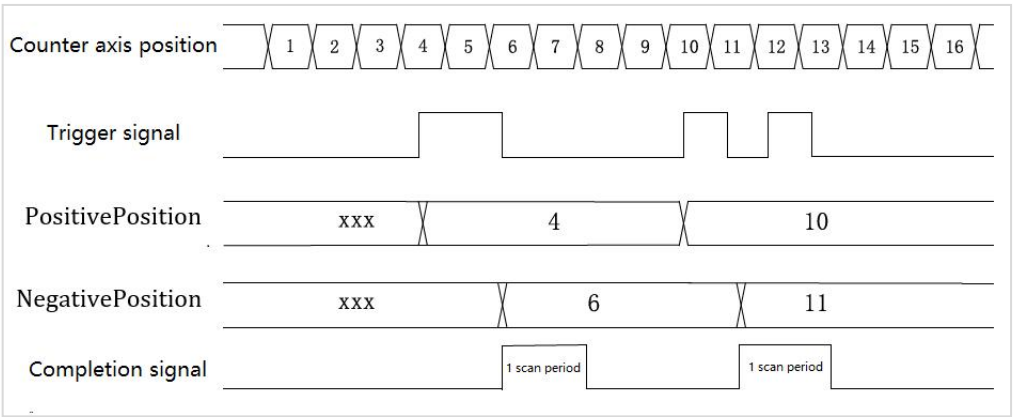


In continuous trigger mode, when the function block instruction Enable is valid and the external input trigger condition disappears, the counter axis position is latched and the completion signal is output. The valid time is 1 scan period. When the completion signal becomes OFF and the external input trigger condition is valid, the counter axis position will continue to be latched, and the completion signal with valid time of 1 scan period will be output. The counter axis position is not latched if an external input trigger condition is valid during the 1 scan period when the completion signal is valid.



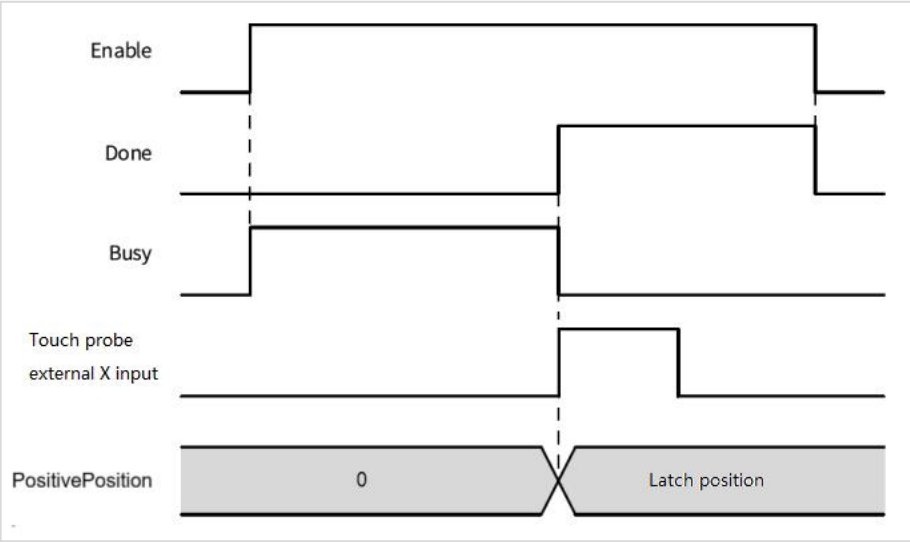
When using the dual-edge trigger mode, the completion signal is output when both the rising and the falling edges are triggered to complete the latch. In single trigger mode, the completion signal lasts until the instruction is completed. In continuous trigger mode, the valid time of completion signal is 1 scan period. The trigger latch signal is not responded to within 1 scan period when the completion signal is valid.



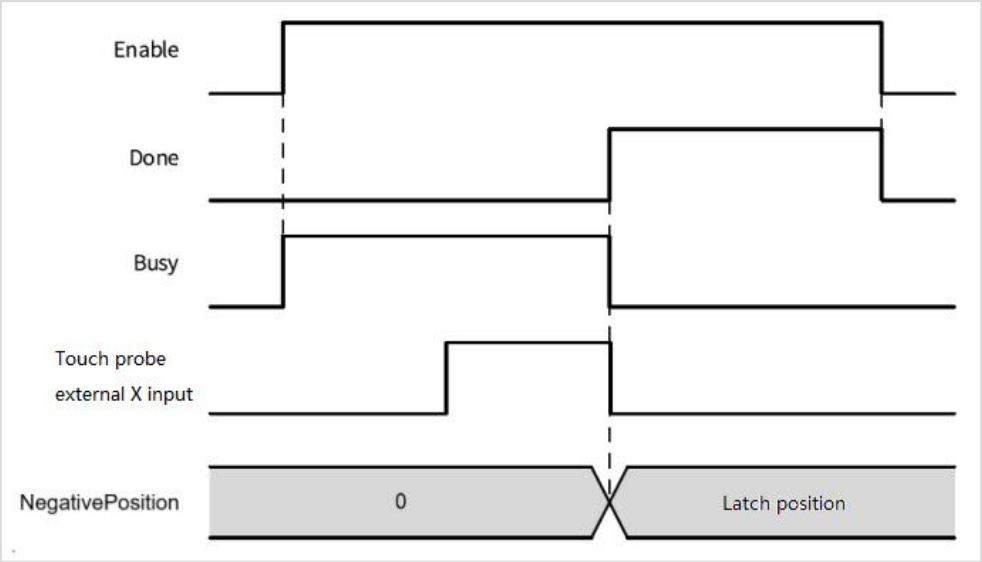


Sequence diagram

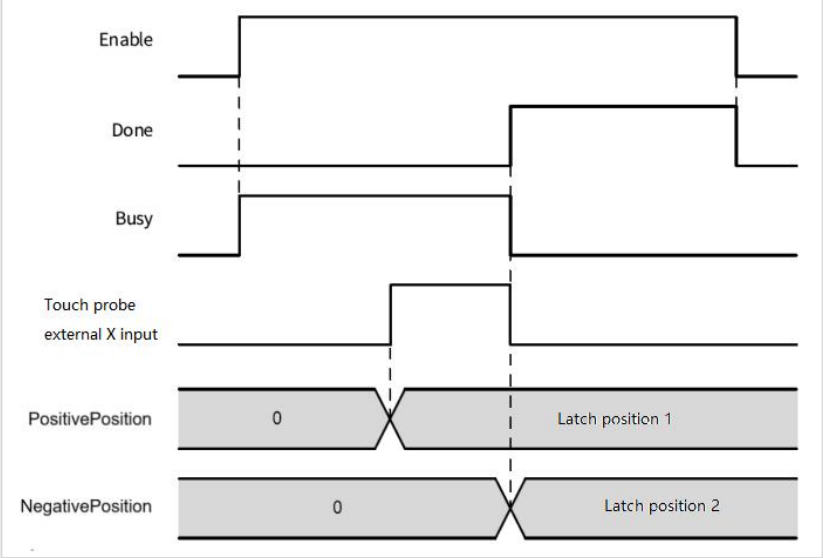
External X rising edge trigger (TriggerEdge=1), single trigger mode (TriggerMode=0), and the instruction sequence diagram is shown in the following figure.



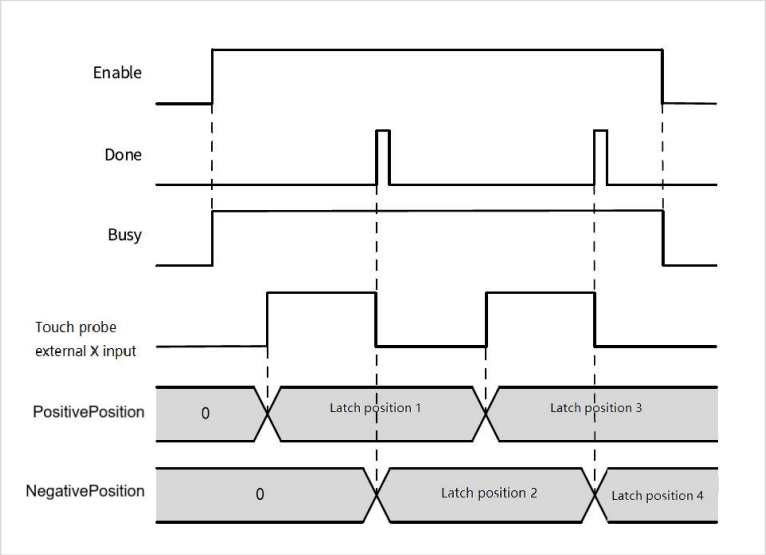
External X falling edge trigger (TriggerEdge=2), single trigger mode (TriggerMode=0), and the instruction sequence diagram is shown in the following figure.



External X rising and falling edges triggering (TriggerEdge=3), single trigger mode (TriggerMode=0), and the instruction sequence diagram is shown in the following figure.



External X rising and falling edges triggering (TriggerEdge=3), continuous trigger mode (TriggerMode=1), and the instruction sequence diagram is shown in the following figure.

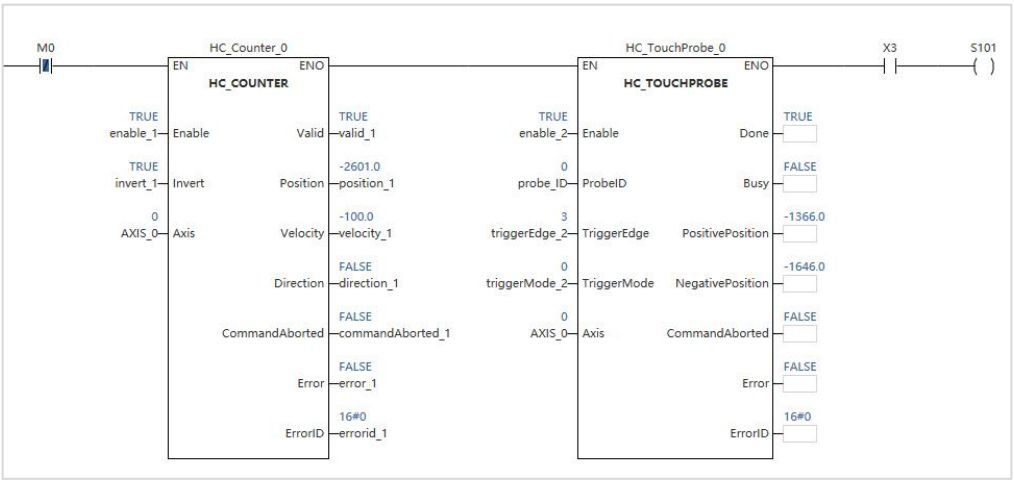


Error code

Error code	Content
0x007000000	Touch probe of axis configuration interface is not enabled
0x00700005	Axis undefined or invalid axis number
0x00800006	Setting errors from trigger edge/trigger mode/touch probe ID
0x00800008	Modify the axis number when the function block is running.

Example

Application example

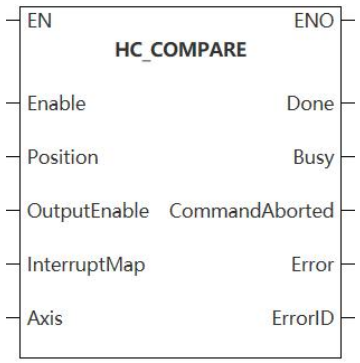


① In the axis configuration interface, the touch probe is enabled and the configuration pin is X3. Set the trigger mode to single trigger (TriggerMode = 0), and the trigger edge is external X rising and falling edges trigger (TriggerEdge = 3)

② When the function block Enable enables, Busy is set to TRUE.

3. When X3 detects the rising edge, latch position to PositivePosition; When X3 detects the falling edge, latch position to NegativePosition. At this time, the execution of this function block is completed, and the Busy status is set to False and Done is set to TRUE.

9.2.4 HC_COMPARE High speed counters comparison



HC_COMPARE

The touch probe counter count reaches the specified value.

Content, range and data type

Instruction	Parameter	Variable type	Can it be empty	Range	Description
HC_COMPARE	Enable	BOOL	No	TRUE FALSE	High speed counter comparison enabled
	Position	LREAL	No	-	Comparison position: user unit
	OutputEnable	BOOL	Yes	TRUE FALSE	Hardware output enabled FALSE: No hardware output TRUE: Using hardware output
	InterruptMap	POINTER	Yes	-	Interrupt mapping: interrupts are generated and associated when the comparison value matches the count value. The interrupt name can be input directly, for instance: IRQ0: associate interrupt IRQ0 None: not associate interrupt
	Axis	WORD	No	0~65535	Axis name/axis number: specify the local encoder axis to operate
	Done	BOOL	Yes	TRUE FALSE	Completed
	Busy	BOOL	Yes	TRUE FALSE	Executing
	CommandAborted	BOOL	Yes	TRUE FALSE	Abort flag
	Error	BOOL	Yes	TRUE FALSE	Error
	ErrorID	DWORD	Yes	Check "ErrorID" for details	Error code

Device used

Instruction	Parameter	Devices																						Index modification	DWORD expansion	Pulse expansion
		X	Y	M	S	SM	T (Bit)	HSC (Bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	DXX	XXP
HC_COMPARE	Enable	●	●	●	●	●																				
	Position																									
	OutputEnable	●	●	●	●	●																				
	InterruptMap																									
	Axis															●	●									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Function

Enabling HC_COMPARE to achieve a single position comparison of the counter axis.

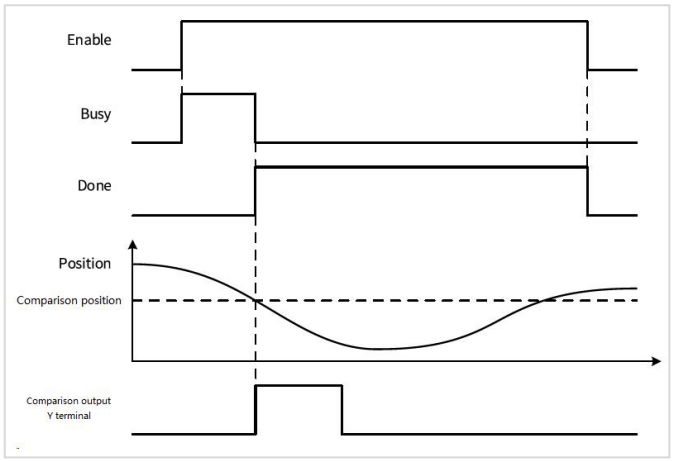
✎Note:

（1）When using this function block, if the InterruptMap parameter specifies the program name of mapping interrupt, the EI instruction should be enabled in the scanning program. Otherwise, the interrupt program cannot be executed.

（2）If there are two high speed comparation function blocks, whose Axis parameters specify the same local encoder axis, the execution of the former executed function block will be interrupted by the latter function block. The ComandAborted execution interrupt will be set to ON.

Description

Using hardware output (OutputEnable=TRUE), the instruction sequence diagram is shown in the following figure.

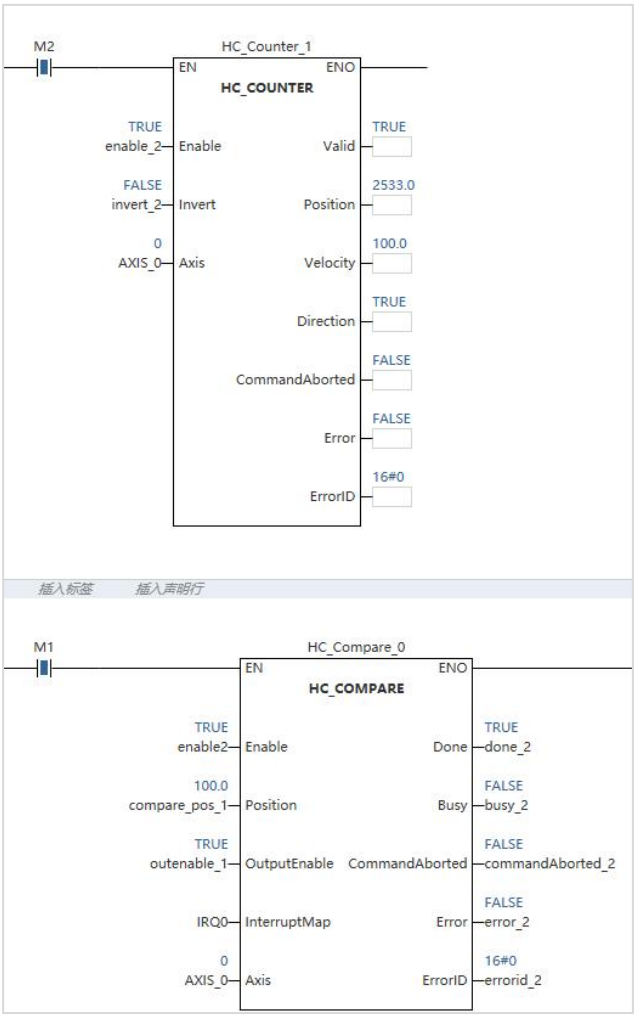


Error code

Error code	Content
0x007000000	OutEnable is ON, but axis configuration interface comparison output is not enabled.
0x00700005	Axis undefined or invalid axis number
0x00800006	The set comparison position is out of limit.
0x00800008	Modify the axis number when the function block is running.

Example

Application example

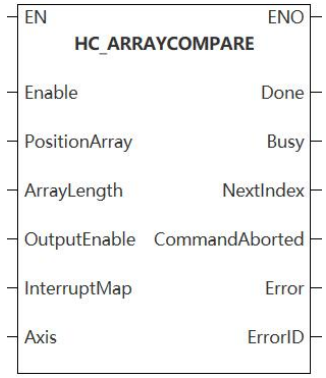


(1) Set the comparison position as 100, configure the interrupt mapping program (InterruptMap = IRQ0), configure the hardware output enabled (outputEnable = TRUE), and enable the function block (Enable = TRUE);

(2) When the counter axis position reaches 100, the comparison interrupt is triggered, and the interrupt IRQ0 is called. If the hardware output is enabled at this time, the pulse of set pulse width is output by the corresponding Y output point.

9.2.5 HC_ARRAYCOMPARE High speed counter array comparison

HC_ARRAYCOMPARE



Sequentially detect whether the counter count value is equal to the specified array sequence.

Content, range and data type

Instruction	Parameter	Variable type	Can it be empty	Range	Description
HC_ARRAYCOMPARE	Enable	BOOL	No	TRUE FALSE	High speed counter comparison enabled
	PositionArray	ARRAY OF LREAL	No	-	Comparison position array: user unit
	ArrayLength	WORD	No	1 to 32	Array length
	OutputEnable	WORD	Yes	TRUE FALSE	Hardware output enabled FALSE: No hardware output TRUE: Using hardware output
	InterruptMap	POINTER	Yes	-	Interrupt mapping: when the comparison value matches the count value, an interrupt is generated and associated. Interrupt name can be input directly. IRQ0: associate interrupt IRQ0 None: not associate interrupt
	Axis	WORD	No	0 to 65535	Axis name/axis number: specify the local encoder axis to operate
	Done	BOOL	Yes	TRUE FALSE	Completed
	Busy	BOOL	Yes	TRUE FALSE	Executing
	NextIndex	WORD	Yes	0 to 32	Next comparison position index
	CommandAborted	BOOL	Yes	TRUE FALSE	Abort flag
	Error	BOOL	Yes	TRUE FALSE	Error
	ErrorID	DWORD	Yes	Check "Error code" for details	Error code

Device used

Instruction	Parameter	Devices																				Index modification	DWORD expansion	Pulse expansion
		X	Y	M	S	SM	T (Bit)	HSC (Bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	DXX	XXP
HC_ARRAYCOMPARE	Enable	●	●	●	●	●																				
	PositionArray																									
	ArrayLength															●	●									
	OutputEnable	●	●	●	●	●																				
	InterruptMap																									
	Axis															●	●									
	Done		●	●	●	●																				
	Busy		●	●	●	●																				
	NextIndex															●	●									
	CommandAborted		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorID																									

Function

Enable HC_ARRAYCOMPARE to achieve continuous multi-position comparison of the counter axis.

Description

When the function block Enable is valid, the counter axis position is compared with the first position of the array. When the comparison is equal, it is compared with the next position of the array. The ArrayLength parameter in the function block is used to set the array length. When the comparison of all array positions set by the array length is completed, the signal is continuously output and the multi-position continuous comparison is completed.

The output parameter NextIndex indicates the next comparison position index, namely the number of equal comparisons that have been completed.

✎Note:

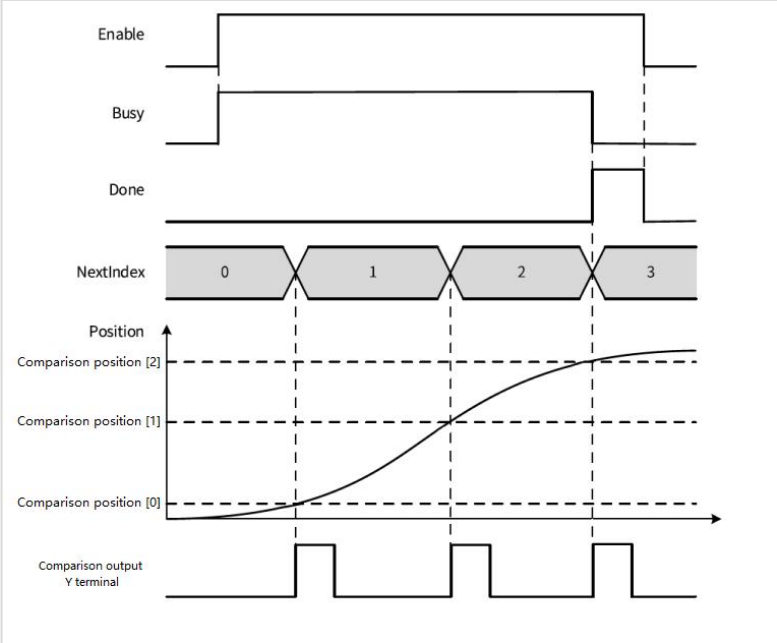
(1) When using this function block, if the InterruptMap parameter specifies the program name of mapping interrupt, the EI instruction should be enabled in the scanning program. Otherwise, the interrupt program cannot be executed.

(2) When inputting the PositionArray parameter of comparison position array, it must be a monotonically increasing array or a monotonically decreasing array, and two adjacent comparison positions cannot be repeated.

(2) If there are two high speed comparation function blocks, whose Axis parameters specify the same local encoder axis, the execution of the former executed function block will be interrupted by the latter function block. The ComandAborted execution interrupt will be set to ON.

Sequence diagram

Compare the three positions (ArrayLength=3), using hardware output (OutputEnable=TRUE), and the instruction sequence diagram is shown in the following figure.

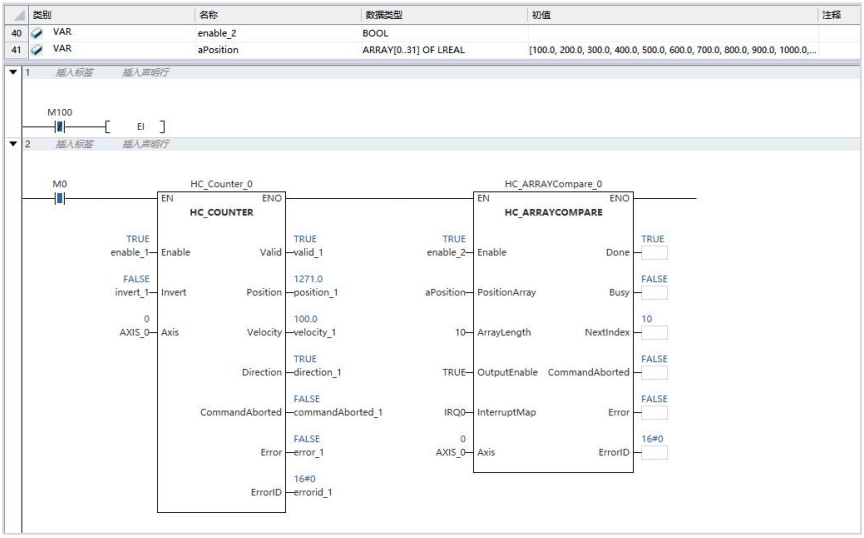


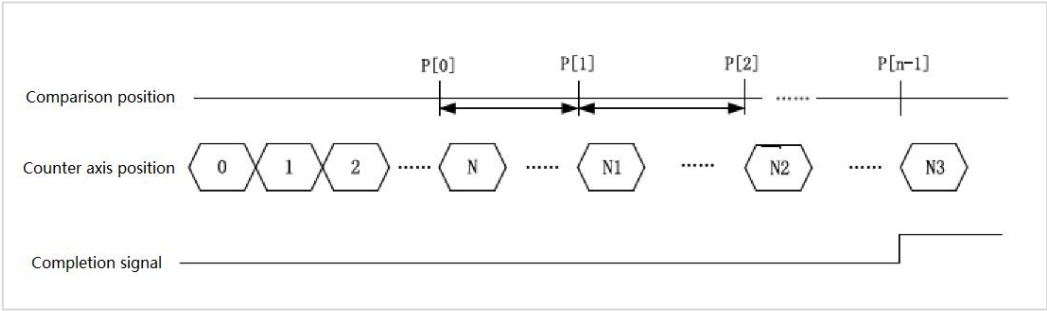
Error code

Error code	Content
0x007000000	OutEnable is ON, but axis configuration interface comparison output is not enabled.
0x00700005	Axis undefined or invalid axis number.
0x00800006	The comparison array is not set or the length of the array is set incorrectly.
0x00800008	Modify the axis number when the function block is running.

Example

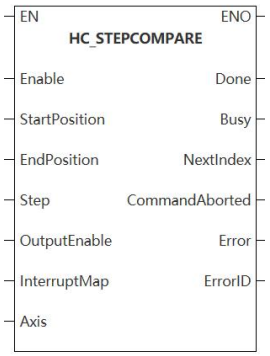
Application example





(1) Set array parameters of current comparison position and array length n (ArrayLength=32 is allowed at the longest). When the function block Enable is enabled, the current comparison position is set to PositionArray Index 0 position value. When the current count value position is equal to the comparison position, the comparison interrupt is triggered, and the interrupt program IRQ0 specified by InterruptMap parameter is called. If the hardware output is enabled (OutputEnable=TRUE), pulse of a specific width pulse is generated in the corresponding Y point. If the comparison of all positions in the array is completed, Done completion flag will continue to be TRUE.

9.2.6. HC_STEPCOMPARE high speed counters equidistant comparison



HC_STEPCOMPARE

Sequentially and continuously compares the counter count values with equidistant consecutive values within a specified range.

Content, range and data type

Instruction	Parameter	Variable type	Can it be empty	Range	Description
HC_STEPCOMPARE	Enable	BOOL	No	TRUE FALSE	High speed counter comparison enabled
	StartPosition	LREAL	No	-	Start position: user unit
	EndPosition	LREAL	No	-	End position: user unit
	Step	LREAL	No	Positive LREAL	Spacing: user unit
	OutputEnable	BOOL	Yes	TRUE FALSE	Hardware output enabled FALSE: No hardware output TRUE: Using hardware output
	InterruptMap	POINTER	Yes	-	Interrupt mapping: when the comparison value matches the count value, an interrupt is generated and associated. Interrupt name can be input directly. IRQ0: associate interrupt IRQ0 None: not associate interrupt
	Axis	WORD	No	0~65535	Axis name/axis number: specify the local encoder axis to operate
	Done	BOOL	Yes	TRUE FALSE	Completed
	Busy	BOOL	Yes	TRUE FALSE	Executing
	NextIndex	WORD	Yes	0~65535	Next comparison position index
	CommandAborted	BOOL	Yes	TRUE FALSE	Abort flag
	Error	BOOL	Yes	TRUE FALSE	Error
	ErrorID	DWORD	Yes	Check "Error code" for details	Error code

Function

Enable HC_STEPCOMPARE to achieve the comparison between counter axis and equidistant continuous position.

Description

When the function block Enable is valid, the counter axis position is compared with the StartPosition position. When the comparison is equal, the comparison continues after increasing or decreasing the Step spacing in the comparison position. When the last comparison position is completed in equidistant comparison, the signal is continuously output.

Output parameter NextIndex

Indicate the index of the next comparison position, and the index starts from 0. (0 represents the first comparison position.)

Therefore, the index is exactly equal to the number of comparison positions that have been completed.

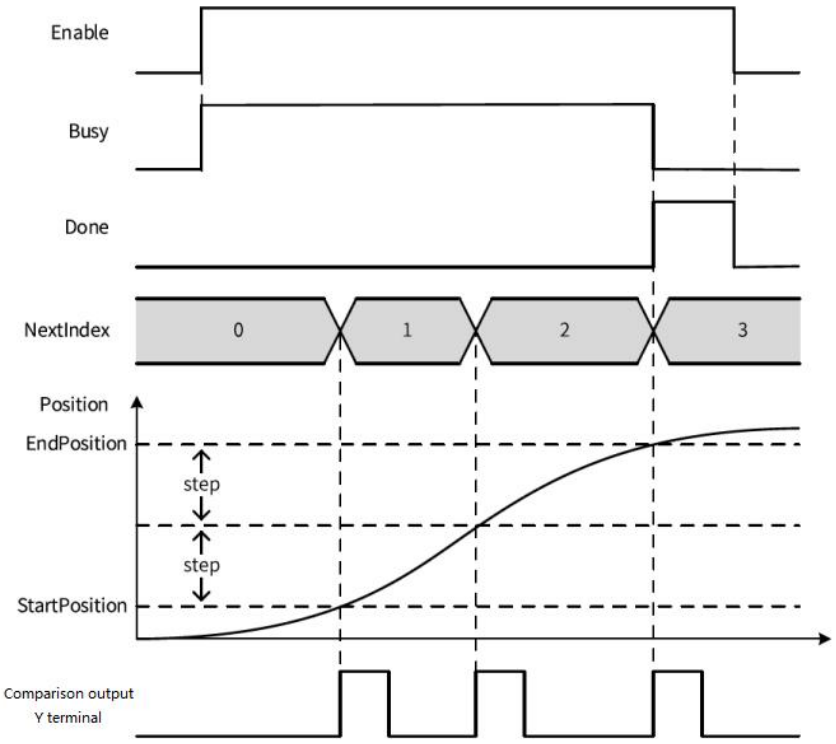
✎Note:

(1) When using this function block, if the InterruptMap parameter specifies the interrupt mapping program, the EI instruction should be enabled in the scanning program. Otherwise, the interrupt program cannot be executed.

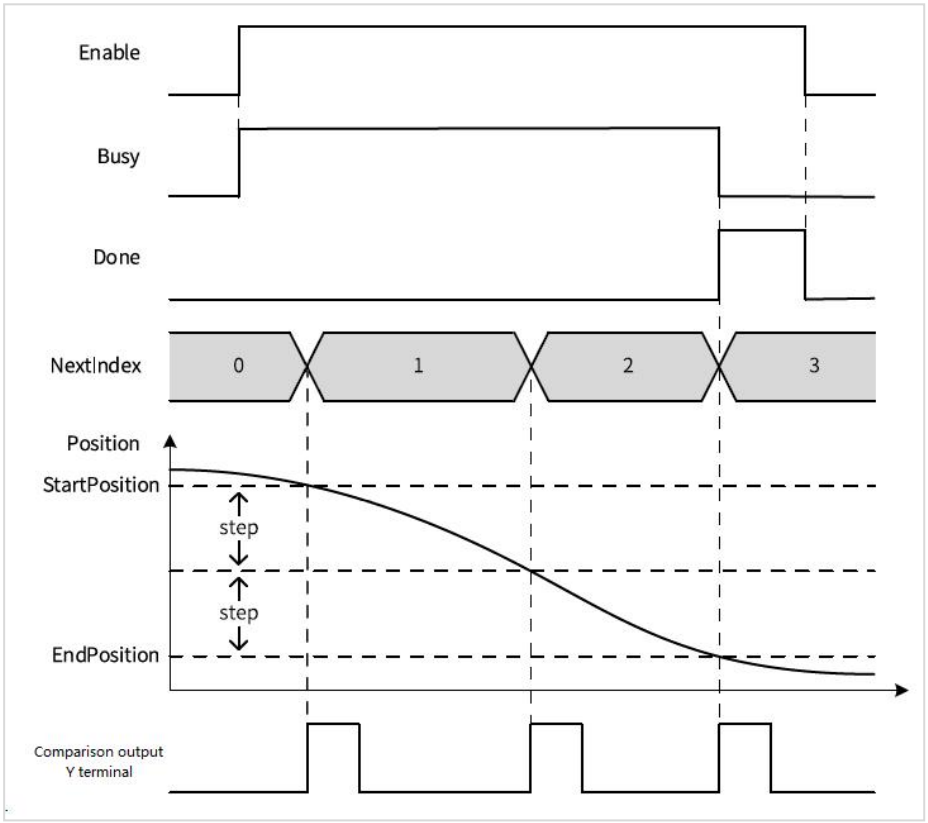
(2) If there are two high speed comparation function blocks, whose Axis parameters specify the same local encoder axis, the execution of the former executed function block will be interrupted by the latter function block. The ComandAborted execution interrupt will be set to ON.

Sequence diagram

Using hardware output (OutputEnable=TRUE), StartPosition<EndPosition, the instruction sequence diagram is shown in the following figure.



Using hardware output (OutputEnable=TRUE), StartPosition<EndPosition, the instruction sequence diagram is shown in the following figure.

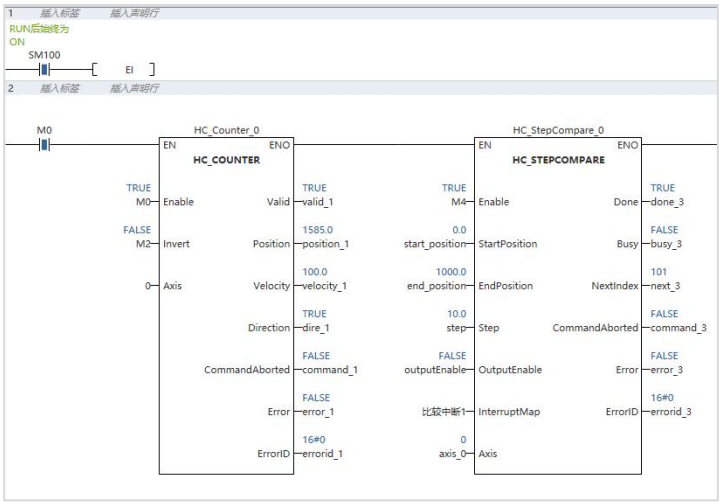


Error code

Error code	Content
0x007000000	OutEnable is ON, but axis configuration interface comparison output is not enabled
0x00700005	Axis undefined or invalid axis number
0x00800000	The number of comparison positions exceeds 65535, and the equidistant comparison is not completed.
0x00800006	Setting start position, end position and spacing error
0x00800008	Modify the axis number when the function block is running.

Example

Application example



When the function block is enabled, the current comparison position is set to the specified position value of StartPosition. When the current count value position is equal to the comparison position, the comparison interrupt is triggered, and the interrupt program specified by the InterruptMap parameter is called. If the hardware output is enabled (OutputEnable=TRUE), the corresponding Y point will generate a pulse with a specific pulse width. When the current position exceeds the end position, the completion signal is continuously output.

9.3 Encoder axis system built-in variables

The system built-in variables for the local encoder axis are shown in the table below.

Name	Data type	Function
param	HC_PARAM_INFO_T	High speed counter information
byCountMode	BYTE	High-speed counting mode 0: A/B Phase 1 time frequency; 1: A/B Phase 2 times frequency; 2: A/B Phase 4 times frequency; 3: CW/CCW(Reserved); 4: Pulse+Direction; 5: Single phase count.
byCountSource	BYTE	High-speed counting signal source A/B phase 1 frequency: 0: X0-Phase A, X1-Phase B; 1: X2-Phase A, X3-Phase B; 2: X4-Phase A, X5-Phase B; 3: X6-Phase A, X7- Phase B. A/B phase 2 times frequency: 0: X0-Phase A, X1-Phase B; 1: X2-Phase A, X3-Phase B; 2: X4-Phase A, X5-Phase B; 3: X6-Phase A, X7- Phase B. A/B phase 4 times frequency: 0: X0-Phase A, X1-phase; 1: X2-Phase A, X3-Phase B; 2: X4-Phase A, X5-Phase B; 3: X6-Phase A, X7-Phase B. Pulse+Direction: 0: X0-pulse, X1-direction; 1: X2-pulse, X3-direction; 2: X4-pulse, X5-direction; 3: X6-pulse, X7-direction. Single phase counting: 0~7: X0-pulse~X7-pulse; 8: Internal 100us signal; 9: Internal 1ms signal; 10: Internal 10ms signal.
bRotategMode	BOOL	Rotation mode 0: Linear mode; 1: Rotation mode.
bLimitEnable	BOOL	High speed count limit enabled. Enable: TRUE; Not enabled: FALSE
nFiltTime10ns	DINT	High-speed counter filter value
nFreqFilterTime	DINT	High-speed counter frequency measurement time
dRotateCycle	LREAL	Rotation cycle
dLimitPosP	LREAL	Positive limit position\ User unit
dLimitPosN	LREAL	Negative limit position\user unit
Preset	HC_PRESET_INFO_T	High speed counter preset information
bEnable	BOOL	High speed counter preset enabled

Name	Data type	Function
bPresetState	BOOL	Preset trigger status
byPin	BYTE	Preset the external interrupt pin. 2: X2; 6: X6.
byTriggerEdge	BYTE	External interrupt triggers the edge. 1: Rising edge; 2: Falling edge 3: Double edge.
nPresetPos	DINT	Default position \ Pulse unit
dPresetUintPos	LREAL	Preset position \ User unit
Probe	HC_TOUCH_PROBE_T	High-speed counter touch probe informatio
bEnable	BOOL	High speed counter touch probe enable
byTouchProbeId	BYTE	High speed counter touch probe ID
byExtiPin	BYTE	Preset function external interrupt input pin 3: X3; 7: X7.
byTriggerEdge	BYTE	External interrupt triggers the edge. 1: Rising edge; 2: Falling edge 3: Double edge.
nPostivePos	DINT	Rising edge latch position \ pulse unit
nNegativePos	DINT	Falling edge latch position \ pulse unit
dPositiveUintPos	LREAL	Rising edge latch position \ user unit
dNegativeUintPos	LREAL	Falling edge latch position \ user unit
Compare	HC_COMPARISON_INFO_T	High speed counter comparison information
byOutputPin	BYTE	High speed comparison output pin
by InterruptEnable	BOOL	High-speed comparison interrupt enable Reserved
byOutputEnable	BOOL	High speed counter comparison output enable
by InterruptMap	BYTE	High-speed comparison interrupt code Reserved
byOutputMode	BYTE	High speed comparison output pin 0: 1ms; 1: 1 unit.
dCompareUnitPos	LREAL	Compare location\User unit
dOutputWidth	LREAL	Output width dOutputWidth*Current output mode unit
Cfg	AXIS_INFO_CFG_T	Axis basic configuration (Read only)
wAxisID	WORD	Axis number (Read only)
wVersion	WORD	Version number (Read only)
wAxisType	WORD	Axis type (Read only) 0: Bus servo axis 1: Bus encoder axis 2: Local pulse axis 3: Local encoder axis
nEtherCATNo	DINT	EtherCAT axis number

Name	Data type	Function
wLocalID	DINT	Local axis ID. Y0/Y1 is 0; Y2/Y3 is 1; Y4/Y5 is 2; Y6/Y7 is 3.
bVirtualAxis	BOOL	Virtual axis: TRUE Non-virtual axis: FALSE
Act	HC_ENCODER_ACT_T	Axis current value
nCounterPos	DINT	Current position\Pulse unit
nCounterVel	DINT	Current Speed\ Pulse Unit
dActPos	LREAL	Current user unit position
dActVel	LREAL	Current user unit speed
Status	HC_ENCODER_STATUS_T	Axis status
bPowerStatus	BOOL	Enable status
bDirStatus	BOOL	Counting direction 0: Negative; 1: Forward.
bLimitStatusP	BOOL	Forward limit state
bLimitStatusN	BOOL	Reverse limit status
Unit:	AXIS_UNITCONV_INFO_T	Unit conversion information
bReverse	BOOL	Reverse or not
bEnGearBox	BOOL	Enable gearbox
nRotatePulseNum	DINT	The number of pulses for one rotation of the motor/encoder
dRotateMoveAmount1	LREAL	Movement amount 1 in one rotation
dRotateMoveAmount2	LREAL	Movement amount 2 in one rotation
nGearRatioM	INT	Gear ratio molecule
nGearRatioD	INT	Gear ratio denominator

10 Communication instruction

10.1 Communication port protocol setting

10.1.1 Communication port protocol setting

PROTOCOL(P)

Set (s) protocol for (n) communication port.

-[PROTOCOL (s) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Protocol number to be set	-	Unsigned BIN16 bits	ANY16	INT
(n)	Which communication port to be set, 0 means COM1, 1 means COM2	0, 1	Unsigned BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	N	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PROTOCOL	Parameter 1												●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●			●	●		●	●

Function

This instruction is mainly used to set the protocol during RUN, and can also be used in the first cycle of RUN.

The value of the protocol address (COM1 SD2542, COM2 SD2592) and the protocol modification sign (COM1 SD2543, COM2 SD2593) can be directly set according to the instruction parameters.

The specific calculation formula for setting the protocol modification sign (COM1 SD2543, COM2 SD2593) is: (parameter setting value + offset of the corresponding serial port's initial special device+ 10)*2

For example, setting the protocol to 2, i.e. (2 +2593-2590 +10)*2, which is (2+3+10) *2=30. At this time, SD2592 will be set to 1, and SD2593 will be set to 32.

·**Note:** Whether the value is set by self-calculation and then modifying the identification (COM1 SD2543, COM2 SD2593) to the protocol, or by using this instruction, the protocol can be modified when the PLC is in the RUN state.

Protocol modification sign (COM1 SD2543 and COM2 SD2593): during the RUN process, when the protocol and modification sign are set correctly (the correctness will be judged at the End of the program), it will be deemed as the correct first cycle; then the protocol and modification sign (COM1 SD2543, COM2 SD2593) will be cleared upon completion of modification and the same value that is consequently set will not be processed. It mainly aims to avoid repeated settings for multiple times.

Protocol number

Protocol number	Content
0H	Wecon Modbus slave station
2H	Modbus RTU slave station
3H	Modbus ASCII slave station
10H	User-defined protocol
20H	Modbus RTU master

Protocol number	Content
30H	Modbus ASCII master

Related devices

Devices	Content
SD2542	COM1 protocol settings
SD2543	COM1 protocol modification sign
SD2592	COM2 protocol settings
SD2593	COM2 protocol modification sign

Precautions

The setting of communication parameters will affect the overall communication. The parameters in PLC are modified when there is no communication or after a round of communication is completed. Attention shall be paid to this point.

If the set protocol does not match the provided protocol number, the protocol modification flag (COM1 SD2543, COM2 SD2593) will not be cleared after setting. At this time, the protocol will not be set successfully, and it will run according to the original protocol.

Error code

Error code	Content
4085H	The read address (s) (n) exceeds the device range.
4084H	(n) is not 1.

Example



As shown in above example, M1 turns from OFF to ON during RUN process.

SD2594 will be set to 32 (20H), SD2595 will be set to 90 and then it will be cleared. At this time, it has been set successfully.

SD2592	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	32
SD2593	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

If M1 turns from OFF to ON again,

SD2592 will be set to 32 (20H), SD2595 will be set to 90, which will not be cleared.

SD2592	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	32
SD2593	0	1	0	1	1	0	1	0	0	0	0	0	0	0	0	0	90

10.2 Modbus serial port parameter setting

10.2.1 PORTPARA/Modbus serial port parameter setting

PORTPARA(P)

Set (s) serial port parameters for (n) communication port.

-[PORTPARA (s) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Serial port parameters to be set	-	Unsigned BIN16 bits	ANY16	INT
(n)	Which communication port to be set, 0 means COM1, 1 means COM2	0, 1	Unsigned BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PORTPARA	Parameter 1												●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 2												●	●	●	●	●	●	●	●			●	●		●	●

Function

This instruction is mainly used to set serial port parameters during RUN, and can also be used in the first cycle of RUN.

The value of serial port parameter address (COM1 SD2540, COM2 SD2590) and the serial parameter modification sign (COM1 SD2541, COM2 SD2591) can be set directly according to the instruction parameters.

The specific calculation formula for setting the serial port parameter modification sign (COM1 SD2541, COM2 SD2591) is: (parameter setting value + offset of the corresponding serial port's initial special device + 10)*2

For example, setting the COM1 serial port parameter to 193 (HC1), i.e. (193 +2541-2540 +10)*2 which is (193 +1+10) *2=408. At this time, SD2540 is set to be 193(HC1) and SD2541 is set to be 408.

·Note: Whether the value is set by self-calculation and then modifying the sign (COM1 SD2541, COM2 SD2591) to the serial port parameters, or by using this instruction, the serial port parameters can be modified when the PLC is in the RUN state.

·Serial port parameter modification sign (COM1 SD2541 and COM2 SD2591): during the RUN process, when the serial port parameter and modification sign are set correctly (the correctness will be judged at the End of the program), it will be deemed as the correct first cycle; then the serial port parameter and modification sign (COM1 SD2541, COM2 SD2591) will be cleared upon completion of modification and the same value that is consequently set will not be processed. It mainly aims to avoid repeated settings for multiple times.

Serial port parameter setting table

Bit number	Name	Content	
		0 (bit OFF)	1 (bit ON)
b0	Data length	7 digits	8 digits
b1 b2	Parity check	b2, b1 (0, 0): None (0,1): Odd parity (ODD) (1,1): Even parity (EVEN)	
b3	Stop bit	1 bit	2 bit

Bit number	Name	Content	
		0 (bit OFF)	1 (bit ON)
b4 b5 b6 b7 b8 b9	Baud rate(bps)	b9,b8,b7,b6,b5,b4 (0, 0, 0, 1, 1, 1): 4800 (0, 0, 1, 0, 0, 0): 9600 (0, 0, 1, 0, 0, 1): 19200 (0, 0, 1, 0, 1, 0): 38400 (0, 0, 1, 0, 1, 1): 57600	b9,b8,b7,b6,b5,b4 (0, 0, 1, 1, 0, 0): 115200 (0, 0, 1, 1, 0, 1): 187500 (0, 0, 1, 1, 1, 0): 230400 (0, 0, 1, 1, 1, 1): 460800 (0, 1, 0, 0, 0, 0): 921600
b10	STX	Disable STX function	Enable STX function, the specific value is set by D8124
b11	ETX	Disable ETX function	Enable the ETX function, the specific value is set by D8125
b12	--	--	--
b13	--	--	--
b14	--	--	--
b15	--	--	--

Related devices

Devices	Content
SD2540	COM1 serial port parameter setting
SD2541	COM1 serial port parameter modification sign
SD2590	COM2 serial port parameter setting
SD2591	COM2 serial port parameter modification sign

Precautions

The setting of communication parameters will affect the overall communication. The parameters in PLC are modified when there is no communication or after a round of communication is completed. Attention shall be paid to this point.

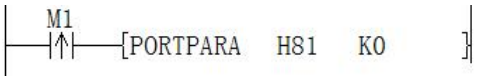
STX function and ETX function are only useful in the case of user-defined protocol.

If the set protocol does not match the provided protocol number, the serial port parameter modification sign (COM1 SD2541, COM2 SD2591) will not be cleared after setting. At this time, the protocol will not be set successfully, and it will run according to the original protocol.

Error code

Error code	Content
4085H	The read address (s) and (n) exceeds the device range.
4084H	(n) is not 0 or 1.

Example



As shown in the above example,

M1 turns from OFF to ON during RUN process.

SD2540 will be set to 129 (H81, baud rate: 9600, stop bit: 1, data bit: 8, parity bit: none), SD2541 will be set to 280 and then cleared. At this time, it has been set successfully

SD2540	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	129
SD2541	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

If M1 turns from OFF to ON again,

SD2540 will be set to 129 (H81), SD2541 will be set to 280, which will not be cleared.

SD2540	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	129
SD2541	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	280

10.3 Modbus station number setting

10.3.1 STATION/Modbus station number setting

STATION (P)

Under the Modbus slave station protocol. Set the station number (s) for the (n) communication port.

-[STATION (s) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Station number to be set	0~255	Unsigned BIN16 bits	ANY16	INT
(n)	Which communication port to set, 0 means COM1, 1 means COM2	0, 1	Unsigned BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
STATION	Parameter 1												•	•	•	•	•	•	•	•			•	•		•	•
	Parameter 2												•	•	•	•	•	•	•	•			•	•		•	•

Function

This instruction is mainly used to set the station number during RUN, and can also be used in the first cycle of RUN.

●The value of the station number address (COM1 SD2544, COM2 SD2594) and the station number modification sign (COM1 SD2545, COM2 SD2595) can be directly set according to the instruction parameters.

●The specific calculation formula for setting the station number modification sign (COM1 SD2545, COM2 SD2595) is: (parameter setting value + offset of the corresponding serial port's initial special device + 10) *2

For example, setting COM2 station number to 1, i.e. (1 +2595 - 2590 +10) *2, which is (1 +5+10) *2=32. At this time, SD2594 will be set to 1, and SD2595 will be set to 32.

Note: Whether the value is set by self-calculation and then modifying the sign (COM1 SD2545, COM2 SD2595) to the station number, or by using this instruction, the station number can be modified when the PLC is in the RUN state.

Station number modification sign (COM1 SD2545 and COM2 SD2595): During RUN, when the station number and modification sign are set correctly (the correctness will be judged at the End of the program), it will be deemed as the correct first cycle; then the protocol and modification sign (COM1 SD2545, COM2 SD2595) will be cleared upon completion of modification and the same value that is consequently set will not be processed. It mainly aims to avoid repeated settings for multiple times.

Related devices

Devices	Content
SD2544	COM1 station number settings
SD2545	COM1 station number modification sign
SD2594	COM2 station number settings
SD2595	COM2 station number modification sign

Error code

Error code	Content
4085H	The read address (s) (n) exceeds the device range.
4084H	(s) not within the range of 0~255.
	(n) is not 0 or 1.

Precautions

The setting of communication parameters will affect the overall communication. The parameters in PLC are modified when there is no communication or after a round of communication is completed. Attention shall be paid to this point.

Example



As shown in the above example,

M1 turns from OFF to ON during RUN process.

SD2594 will be set to 2, SD2595 will be set to 34 and then it will be cleared. At this time, it has been set successfully.

SD2594	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2
SD2595	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

If M1 turns from OFF to ON again,

SD2594 will be set to 2, SD2595 will be set to 34 but will not be cleared.

SD2594	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2
SD2595	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	34

10.4 RS Instruction

10.4.1 RS/External communication instruction

RS

In Modbus master station protocol: the instruction is the setting interface for the master station to send protocol frames. According to the station number function code (s), slave station address (m), length (d) set by the instruction and the function code, it can be determined whether n data is required to automatically combine the transmission and reception of protocol frames. If it is a read type function code, the data will be written into (n).

-[RS (s) (m) (d) (n)]

Content, range and data type

In the case of user-defined protocol:

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	The starting address of the register area where the data to be sent is stored.	-	Unsigned BIN16 bits	ANY16	--
(m)	The length (in bytes) of the data to be sent.	0~523	Unsigned BIN16 bits	ANY16	--
(d)	The starting number of the device that stores written data.	-	Unsigned BIN16 bits	ANY16	--
(n)	Volume of data written (in bytes).	0~523	Unsigned BIN16 bits	ANY16	--

In Modbus protocol:

Parameter	Content	Range	Data type	Data type (label)
(s)	The high byte stores the station number of the slave station, and the low byte stores the Modbus function code.	-	Unsigned BIN16 bits	ANY16
(m)	Slave station address. The address provided by the slave station will read or write data from the address of slave station.	-	Unsigned BIN16 bits	ANY16
(d)	Length, the data length read or written by Modbus, and the unit are determined by the function code.	-	Unsigned BIN16 bits	ANY16
(n)	The starting address for data to be read or written.	-	Unsigned BIN16 bits	ANY16

Device used

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RS	Parameter 1															●	●	●	●				●	●			
	Parameter 2															●	●	●	●				●	●			
	Parameter 3															●	●	●	●				●	●			
	Parameter 4															●	●	●	●				●	●			

(1) **User-defined protocol**

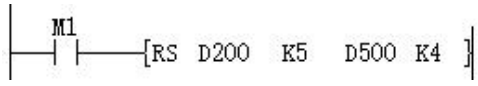
When the communication protocol is set as a user-defined protocol. When the contact in front of RS instruction is turned on and SM2591 is also turned on at the same time, if the sending length (m) is not 0, (m) bytes of the data of (s) will be sent out, and then it will enter the mode of waiting for reception. When the data is received, (N) bytes of data will be stored in (d). SM2593 will be turned ON after receiving (n) data.

In addition, when the sending length (m) is 0, it will be in the receive-only mode. When the receiving length (n) is 0, it will be in the send-only mode.

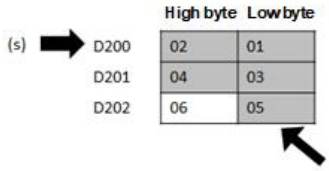
To enable the start character (STX) and the end character (ETX) modes of RS instruction, the status of the 10th and 11th bits of the special address SD2600 must be set. See the table below for detailed settings:

Bit number	Name	Content	
		0 (bit OFF)	1 (bit ON)
b0	Data length	7 digits	8 digits
b1 b2	Parity check	b2, b1 (0, 0): None (0,1): Odd parity (ODD) (1,1): Even parity (EVEN)	
b3	Stop bit	1 bit	2 bit
b4 b5 b6 b7 b8 b9	Baud rate (bps)	b9,b8,b7,b6,b5,b4 (0, 0, 0, 1, 1, 1): 4800 (0, 0, 1, 0, 0, 0): 9600 (0, 0, 1, 0, 0, 1): 19200 (0, 0, 1, 0, 1, 0): 38400 (0, 0, 1, 0, 1, 1): 57600	b9,b8,b7,b6,b5,b4 (0, 0, 1, 1, 0, 0): 115200 (0, 0, 1, 1, 0, 1): 187500 (0, 0, 1, 1, 1, 0): 230400 (0, 0, 1, 1, 1, 1): 460800 (0, 1, 0, 0, 0, 0): 921600
b10	STX	Disable STX function	Enable STX function, the specific value is set by D8124
b11	ETX	Disable ETX function	Enable the ETX function, the specific value is set by D8125
b12	--	--	--
b13	--	--	--
b14	--	--	--
b15	--	--	--

Example

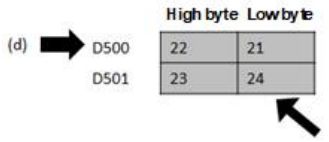


When M1 is ON, the sent and received data of communication are stored as follows after the instruction is executed:



As shown in the figure, the sent data are 01H, 02H, 03H, 04H and 05H in turn

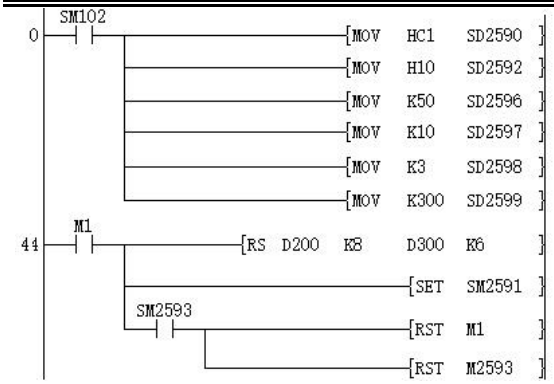
The number of dark backgrounds used is the sent data length (m).



As shown in the figure, the received data are 21H, 22H, 23H and 24H in turn

The number of dark backgrounds used is the received data length (n).

Some configuration and preparation of serial communication are needed for actual programming in order to achieve communication as expected. For instance, sending and receiving mode of serial port, baud rate, number of bits, parity, software protocol settings, timeout judgment conditions, and data preparation for the transceiver buffer, sending and receiving sign processing shall be set. A relatively complete RS communication setting program is shown as follows:



Set the protocol to a user-defined protocol

Sending interval: 5ms

Reception timeout: 100ms

Retries: 3 times

Character interval timeout: 30ms

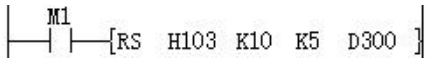
Sending starting address D200 of user-defined protocol: sending length 8

Directly trigger sending upon being

For detailed user-defined protocol instructions, please refer to "9.5.1 User-defined Protocol Description".

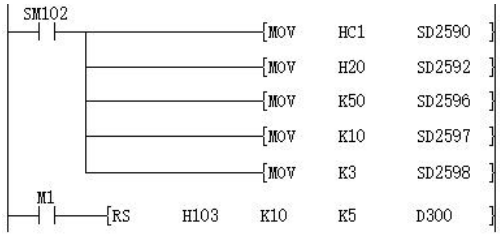
⑵ Modbus protocol

When the protocol is set to Modbus master station protocol (whether it is RTU or ASCII), When the contact in front of the RS instruction is set ON. RS2 instruction will judge whether the data of (N) is needed to send the combined data frame according to station number function code (s), slave station address (m), the length (d) and function code.



After receiving, directly close the instruction and reset the

Parameter	Content
(s)	The high byte stores the station number of slave station, and the low byte stores the function code of Modbus.
(m)	Slave station address. The address provided by the slave station will read or write data from the address of slave station.
(d)	Length: the length read or written by Modbus. The unit is determined by function code.
(n)	The starting address for data to be read or written.



Communication protocol setting:

Sending interval: 5ms

Reception timeout: 100ms

Timeout retry times: 3 times

Station number 01H, function code 03H, slave station

As shown in above ladder diagram:

When M1 is turned ON, PLC will send data (hexadecimal) from COM2 of PLC: 01 03 00 0A 00 05 A5 C8

01: represents slave address, the high 8 bits of (s);

03: Modbus instruction code, the low 8 bits of (s), which means to read the slave register;

00 0A: the address of slave register to be read, the value of (m);

00 05: the number of registers to be read, the value of (d),

A5 C8: CRC parity code.

For detailed user-defined protocol instructions, please refer to "9.5.2 Modbus Protocol Description".

Precautions

1. Although the RS instruction currently allows all the parameters of the instruction to use constants such as K and H, different protocols have different restrictions.
2. When the protocol is a user-defined protocol, S and D cannot be constants, otherwise it will report (3189H) error.
3. When the protocol is Modbus protocol, n cannot be a constant, otherwise it will report (3189H) error.
4. The combination of RTU protocol and 7-bit data bits cannot be set.
5. If the serial port parameter settings are different, it may still be able to communicate. At present, it is normal.
6. Currently, the RS comand cannot be used in interrupts and events.

Error code

Error code	Content
4085H	(s), (m), (d) and (n) read address exceeds the range of the device.
3180H	COM2 data reception error. There may be interference on the communication cable, it is recommended to connect GND.
3181H	COM2 data reception timeout. Check the wiring, whether the serial port parameter settings correspond to master and slave, and whether there is interference. Check whether it is caused by the late response from the slave station. It is recommended to increase the sending interval SD2546.
3182H	COM2 CRC parity error. There may be interference on the communication cable, it is recommended to connect GND.
3183H	COM2 LRC parity error. There may be interference on the communication cable, it is recommended to connect GND.
3184H	The COM2 station number is incorrectly configured. Check the slave station number setting, and check whether there is any problem with the receiving and sending mechanism from slave station.
3185H	COM2 send buffer overflow If this error message appears, please contact technical personnel
3186H	COM2 function code is wrong. Check whether the set function code is a function code supported by PLC.
3187H	COM2 address is wrong. Check whether the slave station has this address. (Please refer to Modbus exception 02).
3188H	The length of COM2 is wrong. Check whether the communication length exceeds the length range specified by Modbus protocol or user-defined protocol.
3189H	COM2 data error. Check whether parameters of the instruction are wrong. Check whether the set value is supported by the slave station. (Please refer to Modbus exception 03).
318AH	COM2 slave station is busy. Message returned from slave station: Slave station is busy. (Please refer to Modbus exception 06).
318BH	COM2 slave station does not support function codes. Check whether this function code is supported by the slave station. (Please refer to Modbus exception 01).
318CH	COM2 slave station is faulty. Message returned from slave station: Slave station failed, please check whether slave station failed. (Please refer to Modbus exception 04).
318DH	COM2 slave station confirmation. Message returned from slave station: slave station confirmation. (Please refer to Modbus exception 05).
318EH	At present, COM2 does not support this instruction or function. When the slave protocol is set, the communication instructions related to master station or master station functions cannot be used. Please change the protocol or close the contact before the corresponding instruction or close the corresponding communication function.
318FH	COM2 sending timeout. If this error message appears, please contact technical personnel.
31A0H	No available gateway for COM2 Message returned from the slave station: no available gateway. (Please refer to Modbus exception 0A).
31A1H	COM2 indicates that no response was obtained from the target device. Message returned from the slave station: The device is not networked. (Please refer to Modbus Exception 0B).

10.5 Instructions for use of RS instruction protocol

10.5.1 User-defined protocol description

10.5.1.1 Introduction

The function of user-defined protocol: it can directly transmit data with equipment without any processing or the customer can set corresponding protocol to communicate with other equipment.

At present, the user-defined protocol of 6V PLC is configured and sent by COM2, and the instruction used is RS instruction. Protocol and serial port parameters need to be configured through devices.

The current user-defined protocol is generally similar to 3V.

10.5.1.2 Basic Configuration

(1) Instruction

The RS instruction itself is used in the same way as the previous RS instruction at 3V, and R devices can be used.

-[RS (s) (m) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom Variable Type
(s)	The starting address of the register area where the data to be sent is stored	-	Unsigned BIN16 bits	ANY16	--
(m)	The length (in bytes) of the data to be sent	0~528	Unsigned BIN16 bits	ANY16	--
(d)	The start number of device that stores written data	-	Unsigned BIN16 bits	ANY16	--
(n)	Volume of data written	0~528	Unsigned BIN16 bits	ANY16	--

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RS	Parameter 1															●	●	●	●				●	●			
	Parameter 2															●	●	●	●				●	●			
	Parameter 3															●	●	●	●				●	●			
	Parameter 4															●	●	●	●				●	●			

Note: Although the RS instruction currently allows all the parameters of the instruction to use constants such as K and H, different protocols have different restrictions.

When the protocol is user-defined protocol, S and D cannot be constants, otherwise an error will be reported.

(2) Special device setting

Special address table

COM2 special D device (SD)		COM2 special M device (SM)	
2590	Communication port settings	2590	Sending control on
2591	COM1 serial port parameter modification sign	2591	Send control/send prompt
2592	Protocol settings	2592	Reception control on
2593	Protocol modification sign	2593	Accept control/receive prompt
2594	Station number settings	2594	8-bit mode (for user-defined protocol)
2595	Station number modification sign	2595	

COM2 special D device (SD)		COM2 special M device (SM)	
2596	Sending interval (0.1ms) 0~32767 Set 0 to 10 (1ms)	2596	
2597	Communication timeout setting (10ms) 0~32767 Set 0 to 10 (100ms)	2597	
2598	Timeout retry times 0~32767	2598	
2599	Character interval timeout setting (for custom protocol) (0.1ms) 0 to 32767 Set 0 to 10 (1ms)	2599	
2600	STX value	2600	
2601	ETX value	2601	
...		...	
2610	Volume of data received	2610	Communication completion sign
2611	Last error	2611	Receiving sign
2612	Current error	2612	Retry
2613	Error steps	2613	Communication error
2614	Error station number	2614	Communication timeout
2615	Cumulative error times	2615	

The contents that will be used or set by the user-defined protocol have been marked with “underline” and “in bold”.

In addition, the devices to be used as judgment conditions have been marked “in bold” in the table.

Note: at present, COM1 doesn’t support user-defined protocol.

① Communication port settings SD2590

Bit number	Name	Content	
		0 (bit OFF)	1 (bit ON)
b0	Data length	7 digits	8 digits
b1 b2	Parity check	b2, b1 (0, 0): None (0,1): Odd parity (ODD) (1,1): Even parity (EVEN)	
b3	Stop bit	1 bit	2 bit
b4 b5 b6 b7 b8 b9	Baud rate (bps)	b9,b8,b7,b6,b5,b4 (0, 0, 0, 1, 1, 1): 4800 (0, 0, 1, 0, 0, 0): 9600 (0, 0, 1, 0, 0, 1): 19200 (0, 0, 1, 0, 1, 0): 38400 (0, 0, 1, 0, 1, 1): 57600	b9,b8,b7,b6,b5,b4 (0, 0, 1, 1, 0, 0): 115200 (0, 0, 1, 1, 0, 1): 187500 (0, 0, 1, 1, 1, 0): 230400 (0, 0, 1, 1, 1, 1): 460800 (0, 1, 0, 0, 0, 0): 921600
b10	STX	Disable STX function	Enable STX function, the specific value is set by D8124
b11	ETX	Disable ETX function	Enable the ETX function, the specific value is set by D8125

According to the bit setting provided in the above table, the serial port parameter is set to SD2590. At present, the setting is basically the same as 3V. As a higher baud rate can be set, the bits used for enabling STX and ETX need to be moved back by two bits.

For instance, set serial port parameters: baud rate: 115200, stop bit: 1, data bit: 8, parity bit: none, enable STX.

Then set the value H4C1 (K1217) on SD2590. At present, the parameters directly set to SD2590 are only valid in the first cycle of PLC RUN, and if they need to be modified during RUN, they can be set by PORTPARAM instruction.

② Protocol settings SD2592

Protocol settings		Protocol settings	
0 H	Wecon Modbus slave station	10 H	User-defined protocol
2 H	Modbus RTU slave station	20 H	Modbus RTU master station
3 H	Modbus ASCII slave station	30 H	Modbus ASCII master station

The corresponding protocol can be set by setting the corresponding value in SD2592.

At present, the parameters directly set to SD2590 are only valid in the first cycle of PLC RUN.

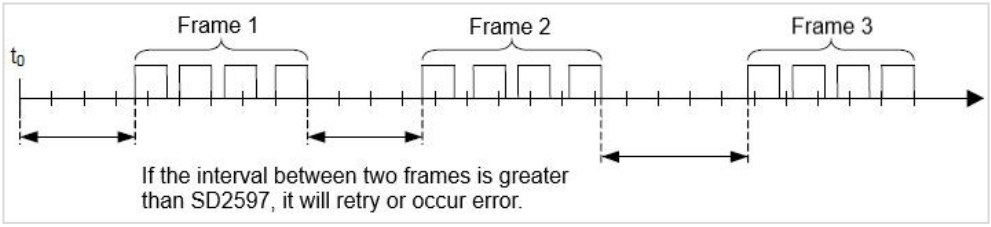
If the parameters need to be modified during RUN, they can be set by PROTOCOL instruction.

③ Sending interval SD2596

The main function of sending interval is: to determine how long does it take to wait to send the next instruction when one instruction is completed. When 0 is set, it will not wait for sending at specified interval, but it will be affected by scanning period.

The unit of transmission interval is 0.1 ms, i.e. the interval time is 10ms when the value is set to be 100.

④ Communication timeout SD2597 and timeout retry SD2598



The main function of communication timeout is: to determine when the retry is required or an error needs to be reported when the data is not received after being sent by PLC. When 0 is set, it is 100ms by default.

The unit of transmission interval is 10ms, i.e. the timeout judgment time is 100ms when the value is set to be 10.

When receiving timeout occurs, it will judge whether there are retries and the current retry times. If the retry times are greater than or equal to SD2598, an error will be reported.

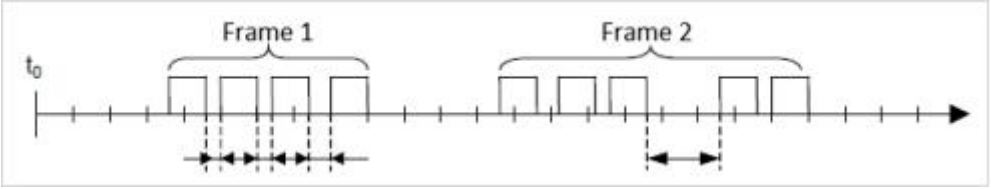
If SD2598 is set to 0, no retry is required.

When it is set to 1, send the data once and then send it again after timeout.

⑤ Character interval timeout SD2599

At present, this setting is available for user-defined protocol.

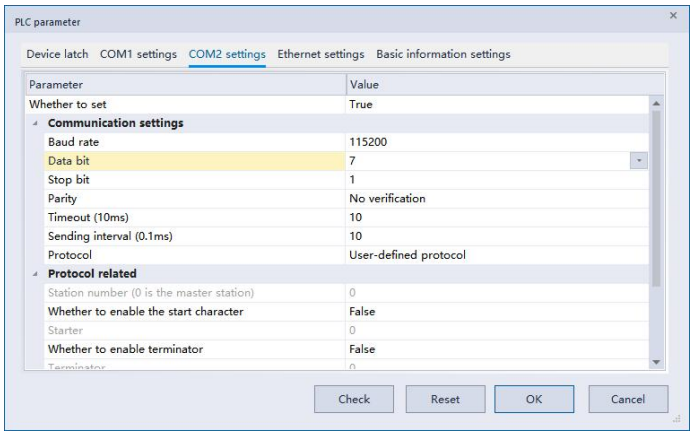
The main function of character interval timeout is: after receiving at least one character before there is no communication timeout, judge whether the interval time between the following two characters exceed the time limit, report an error if it exceeds the time limit, and end communication.



This setting is mainly designed after considering that some devices may send data slowly or there are other specific sending requirements. No retry is required in case of character interval timeout.

⑶ Serial port parameter setting instruction

① Upper computer interface settings



Through the PLC parameter setting in the upper computer interface, if the protocol is switched to "no protocol", it means the user-defined protocol.

Serial port parameters mainly include baud rate, data bits and other serial port parameters.

Start/end characters can be sent when start characters are enabled. Start/end characters will be distinguished when they are received. See the instructions in the sending and receiving process for details.

It is important to note that the characters here are actually decimal values.

② PORTPARA instruction

-[PORTPARA (s) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Serial port parameters to be set	0 to 256	Unsigned BIN16 bits	ANY16	--
(n)	Which communication port to set, 0 means COM1, 1 means COM2	0, 1	Unsigned BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PORTPARA	Parameter 1												•	•	•	•	•	•	•	•			•	•		•	•
	Parameter 2												•	•	•	•	•	•	•	•			•	•		•	•

Function: used to set communication parameters at the time of RUN.

For details, please refer to "PORTPARA Instruction Description".

③ PROTOCOL instruction

-[PROTOCOL (s) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Protocol number to be set	0 to 65535	Unsigned BIN16 bits	ANY16	--
(n)	Which communication port to set, 0 means COM1, 1 means COM2*1	1	Unsigned BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PROTOCOL	Parameter 1												•	•	•	•	•	•	•	•			•	•		•	•
	Parameter 2												•	•	•	•	•	•	•	•			•	•		•	•

Function

It is used to set the communication protocol at the time of RUN, and the above instruction is actually used to set a specific value for the parameter modification sign.

●The specific calculation formula is: (parameter setting value + offset of the corresponding serial port's initial special device + 10)*2

For example, setting the protocol to Modbus RTU master station (H20), i.e. (0x20 +2593-2590 +10) * 2, which is (32 +3 +10) *2=90, which mainly aims to avoid random modification of serial port parameters during operation.

For user-defined protocol, if the set parameters are triggered, the value will be changed only after the instruction is completely completed.

For Modbus master station protocol, if the set parameters are triggered, the value will be changed only after the instruction is completely completed.

For Modbus slave station protocol, if the set parameters are triggered, switching can be performed when the received data is not being processed.

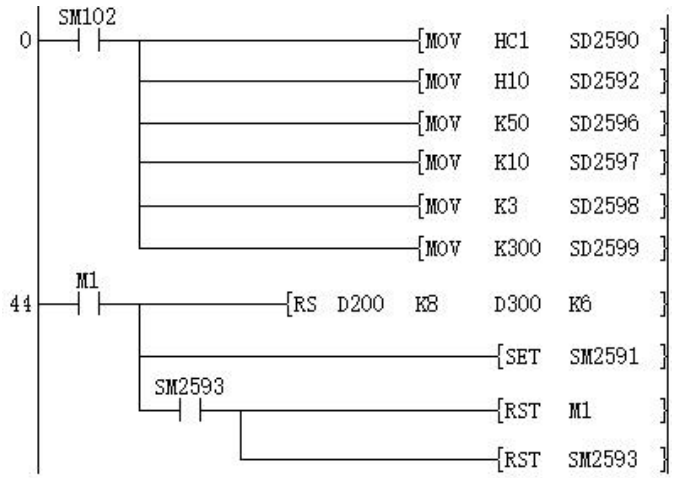
For details, please refer to "PROTOCOL Instruction Description".

④ Priority description of serial port parameters

The priority of serial port parameter setting is listed below. At present, serial port parameters are saved at the time of power failure.

Serial port parameter setting instruction setting = using ladder diagram MOV instruction to set corresponding SD device > upper computer downloading parameters > previous data saved at the time of power outage.

⑷Basic ladder diagram



Setting serial port parameters:
Baud rate: 115200, stop bit: 1, data bit: 8, parity

Set the protocol to a user-defined protocol

Sending interval: 5ms

Reception timeout: 100ms

Retries: 3 times

Character interval timeout: 30ms

Sending starting address D200 of user-defined protocol: sending length 8

Directly trigger sending upon being

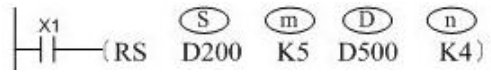
After receiving, directly close the instruction and reset the

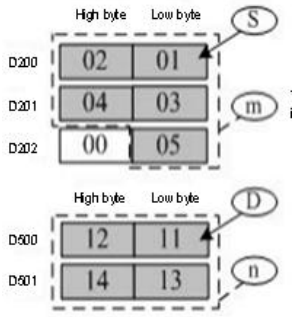
10.5.1.3 Sending and receiving mechanism

①No start character (STX) and end character (ETX)

16-bit (SM2594 is OFF)

Devices will be divided into high and low directly and then the data in the low address will be sent first, followed by the data in the high address.





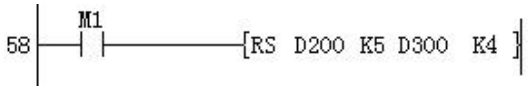
For instance, the sent data are 01, 02, 03, 04 and 05

When X1 is ON, the sent and received data of communication are stored as follows after the instruction is executed.

For instance, the received data are 11, 12, 13 and 14

8-bit (SM2594 is ON)

Get the value of the low address of the device directly and send it



D200	0	0	0	0	1	1	1	0	1	0	1	1	1	1	1	0	7D70
D201	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	00E1
D202	1	0	0	0	1	1	0	0	1	0	1	0	1	0	0	0	1531
D203	0	1	0	0	0	1	1	0	0	0	0	1	1	0	0	0	1862
D204	0	0	1	1	0	1	1	0	0	1	1	1	1	0	0	0	1E6C
D205	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
D206	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

As shown in above ladder diagram and under above data transmission, the sent data should be: 70 E1 31 62 6C.

D300	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0002
D301	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0096
D302	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0042
D303	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0053
D304	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

The received data will also be stored at low address.

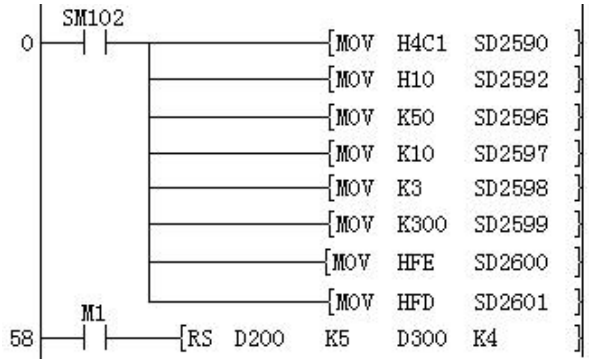
②Opening and setting the start character STX

The value of the start character SD2600 is valid only when it is at low bit.

When STX is opened, the data will start with the start character STX at the time of sending, and the data starting with STX will be received, but no STX will be shown.

When the data starting with STX is received in the half way,the reception process will restart.

If the first character is not received, it will be judged as time out in case of the timeout of the first character; if the first character is received, no matter what kind of data is received, the character timeout will be counted since the last character is received.



D200	0	0	0	0	1	1	1	0	1	0	1	1	1	1	1	0	7D70
D201	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	00E1
D202	1	0	0	0	1	1	0	0	1	0	1	0	1	0	0	0	1531
D203	0	1	0	0	0	1	1	0	0	0	0	1	1	0	0	0	1862
D204	0	0	1	1	0	1	1	0	0	1	1	1	1	0	0	0	1E6C
D205	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
D206	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

As shown in above ladder diagram and under above data transmission, the sent data should be FE 70 7D E1 00 31.

If FE 51 26 34 15 is sent to PLC, the following data will be received.

If FE 25 63 FE 51 26 34 15 is sent to PLC, the following data will be received.

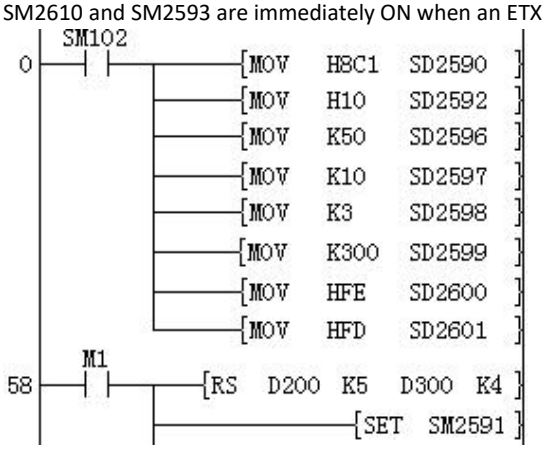
D300	1	0	0	0	1	0	1	0	0	1	1	0	0	1	0	0	2651
D301	0	0	1	0	1	1	0	0	1	0	1	0	1	0	0	0	1534

②Opening and setting the end character ETX

The value of the end character SD2601 is valid only when it is at low bit.

When ETX is opened, the data will end with ETX at the end of sending.

SM2610 and SM2593 are immediately ON when an ETX is received, no matter whether sufficient length of data is received.



D200	0	0	0	0	1	1	1	0	1	0	1	1	1	1	1	0	7D70
D201	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	00E1
D202	1	0	0	0	1	1	0	0	1	0	1	0	1	0	0	0	1531
D203	0	1	0	0	0	1	1	0	0	0	0	1	1	0	0	0	1862
D204	0	0	1	1	0	1	1	0	0	1	1	1	1	0	0	0	1E6C
D205	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
D206	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

As shown in above ladder diagram and under above data transmission, the result is 70 7D E1 00 31 FD.

As shown in above ladder diagram, if 01 06 FD is sent, the following data will be received. If FD is sent directly, it will be judged as "end" directly.

D300	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0601
D301	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
D302	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
D303	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

①Both start character (STX) and end character (ETX) are opened

When both start character and end character are opened, it is generally deemed as the combination of above single open state.

However, if only ETX data is received when both characters are opened, it will not end immediately, but it will change to judge the character interval time, without starting to receive data.

⑵Receiving after sending (similar to Modbus master station mechanism)

① OFF (default) state of sending control switch (SM2590) and receiving control switch (SM2592)

When the contact point of RS instruction is set ON, SM2591 will be sent immediately when it is set ON, and SM2591 will be automatically set OFF after sending.

After sending, judge the state of SM2593. If it is ON, it will stop in this instruction all the time and wait until it is set OFF and then receive data; if it is OFF, it will start receiving data.

After the first character timeout time (SD2597), if no character is received, it is judged as timeout; if retry time is set, retry it. If the communication cannot be achieved after retry, an error will be reported and SM2593 and SM2614 are set to be on.

After receiving the first character, judge whether the interval between characters exceeds the time set by SD2599, otherwise an error will be reported. (Please refer to "Character Interval Timeout" for details.)

If there are two or more instructions, after SM2591 is set to be ON, RS instruction for setting the first ON will be sent, and then if this RS instruction is completed* 1and set to be ON, SM2591 will switch to the next RS instruction in the ladder diagram sequence* 2.

If there are two or more instructions, another data will be sent when the data is half received, and the original instruction is

continued to be executed for receiving until it is completed.

Note * 1: Completion refers to receiving data or reporting errors.

Note * 2: If there is no other RS instruction opened after the ladder diagram sequence, it will return to execute the first RS instruction opened in the ladder diagram sequence.

② Sending mode when the sending control switch (SM2590) is ON

When the sending control switch (SM2590) is ON, there is no need to control SM2591 to be set as ON.

At this time, as long as the contact of RS instruction is triggered, it can be sent, and SM2591 is automatically set as ON.

② Receiving mode when the receiving control switch (SM2592) is ON

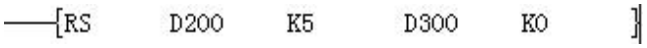
When the receiving control switch (SM2592) is ON, SM2593 is automatically turned to OFF when the instruction is executed.

When receiving, the status of SM2593 will not be judged again, and SM2593 will still be set as ON after receiving.

① ON state of sending control switch (SM2590) and receiving control switch (SM2592)

Under above circumstance, the data can be sent only by triggering the contact in front of the RS instruction, which will automatically turn to the waiting for receiving state, and when the reception is completed, it will turn to the next opened RS instruction.

⑶Send only



When the sending length is set to a non-0 value, the receiving length is set to 0. This is send-only mode.

① OFF state of sending control switch (SM2590)

When SM2591 is set as ON, it sends data and does not receive data.

If there are two or more instructions, after SM2591 is set to be ON, RS instruction for setting the first ON will be sent, and then if this RS instruction is completed and set to be ON, SM2591 will switch to the next RS instruction in the ladder diagram.

② ON state of sending control switch (SM2590)

The data will be sent when RS instruction is triggered.

When several instructions are triggered, the data will be set in circular manner, and during the circular sending, the data will be sent at intervals according to the set sending intervals.

⑷Receive-only



When the sending length is set to be 0, and the receiving length is set to be non-0, it is in receive-only mode.

①OFF state of SM2592 receiving control switch

When receiving, SM2593 will be set as ON, and the data can be received continuously only after it is set as OFF under the control of ladder diagram. SM2593 will not be set as ON if no data is received.

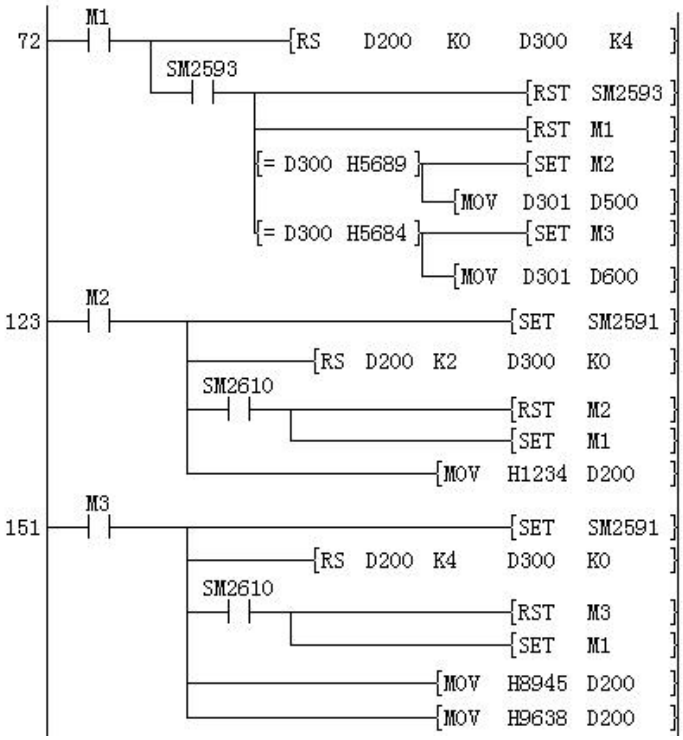
When several instructions are opened at the same time, the receiving position of “receive-only”: the data will be received first according to the first opened instruction; after receiving or after timeout, SM2593 is set as ON and then SM2593 is set as OFF for circular receiving.

①ON state of SM2592 receiving control switch

The data can be normally received, without being controlled by the sign bit. SM2593 receiving sign is used as the receiving prompt sign, and SM2593 will not be set as ON if no data is received.

If several RS receive-only instructions are opened at the same time, the receiving position will be uncertain because the data will be switched to next data due to the judgement of timeout.

⑸ Sending after receiving by the receive-only and send-only mode (similar to Modbus slave station mechanism)



Receive-only: when it is judged as completing reception, the instruction to be triggered will be judged according to the received contents; meanwhile, the received data will be saved.

Send-only: the sent contents are different after M2 is triggered and M3 is triggered. After the trigger is over, they will return to trigger the receive-only part of M1.

10.5.1.4 Error information

The current error code will be displayed in theSD7, SD0 and SD2611 and SD2612.

The error codes that will occur in user-defined protocols are mainly shown as follows:

Error code	Content
3181H	Data reception timeout
3188H	Length error
3189H	COM2 data error. “Check whether the parameters of the instruction are wrong. Check whether the set value is supported by the slave station. (Please refer to Modbus exception 03).”
318EH	At present, COM2 does not support this instruction or function. When the slave protocol is set, the communication instructions related to master station or master station functions cannot be used. Please change the protocol or close the contact before the corresponding instruction or close the corresponding communication function.
318FH	COM2 sending timeout. If this error message appears, please contact technical personnel.
3190H	COM2 receives data beyond the cache limit.

After resetting the protocol or communication parameters, the reported errors will be cleared.

In addition, in case of communication completion/communication error/communication timeout, the sign bit will be set after executing the instruction.

10.5.1.5 Difference with Mitsubishi

Currently known difference:

⑴ For STX and ETX, Mitsubishi can set a maximum of 4 bytes, but we only have one byte.

⑵Support accumulation, parity, CR and LF.

⑶Mitsubishi's instruction control is to execute the instruction on a fixed basis when the instruction is triggered first. It will not be switched to other instructions midway unless the contact of this instruction is closed.

10.5.2 Modbus protocol description

10.5.2.1 Introduction

The current Modbus master station protocol is generally similar to LX3V. The address is modified in the slave station part.

10.5.2.2 Basic Configuration

(1) Instruction

The RS instruction itself is used in the same way as the previous RS instruction at 3V. R device can be used.

-[RS (s) (m) (d) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	The high byte stores the station number of slave station, and the low byte stores the function code of Modbus.	-	Unsigned BIN16 bits	ANY16
(m)	Slave station address. The address provided by the slave station will read or write data from the address of slave station.	-	Unsigned BIN16 bits	ANY16
(d)	Length: the length read or written by Modbus. The unit is determined by function code.	-	Unsigned BIN16 bits	ANY16
(n)	The starting address for data to be read or written.	-	Unsigned BIN16 bits	ANY16

Device used

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RS	Parameter 1															●	●	●	●				●	●			
	Parameter 2															●	●	●	●				●	●			
	Parameter 3															●	●	●	●				●	●			
	Parameter 4															●	●	●	●								

Note: Although the RS instruction currently allows all the parameters of the instruction to use constants such as K and H, different protocols have different restrictions. When the protocol is Modbus master station protocol, n cannot be a constant, otherwise it will report an error.

(2) Special device setting

Special address table

COM1 special D device (SD)	
2540	Communication port settings
2541	COM1 serial port parameter modification sign
2542	
2543	
2544	Station number settings
2545	Station number modification sign

*Note: COM1 doesn't support modification of protocol.

COM2 special D device (SD)		COM2 special M device (SM)	
2590	Communication port settings	2590	Sending control on
2591	COM1 serial port parameter modification sign	2591	Send control/send prompt
2592	Protocol settings	2592	Reception control on
2593	Protocol modification sign	2593	Accept control/receive prompt

COM2 special D device (SD)		COM2 special M device (SM)	
2594	Station number settings	2594	8-bit mode (for user-defined protocol)
2595	Station number modification sign	2595	
2596	Sending interval (0.1ms) 0-32767 Set 0 to 10 (1ms)	2596	
2597	Communication timeout setting (10ms) 0-32767 Set 0 to 10 (100ms)	2597	
2598	Timeout retry times 0-32767	2598	
2599	Character interval timeout setting (for custom protocol) (0.1ms) 0 to 32767 Set 0 to 10 (1ms)	2599	
2600	STX value	2600	
2601	ETX value	2601	
...		..	
2610	Volume of data received	2610	Communication completion sign
2611	Last error	2611	Receiving sign
2612	Current error	2612	Retry
2613	Error steps	2613	Communication error
2614	Error station number	2614	Communication timeout
2615	Cumulative error times	2615	

The contents that will be used or set by the Modbus protocol have been marked with “underline” and “in bold”.

In addition, the devices to be used as judgment conditions have been marked “in bold” in the table.

①Communication port settings SD2590

Bit number	Name	Content	
		0 (bit OFF)	1 (bit ON)
b0	Data length	7 digits	8 digits
b1 b2	Parity check	b2, b1 (0, 0): None (0,1): Odd parity (ODD) (1,1): Even parity (EVEN)	
b3	Stop bit	1 bit	2 bit
b4 b5 b6 b7 b8 b9	Baud rate (bps)	b9,b8,b7,b6,b5,b4 (0, 0, 0, 1, 1, 1): 4800 (0, 0, 1, 0, 0, 0): 9600 (0, 0, 1, 0, 0, 1): 19200 (0, 0, 1, 0, 1, 0): 38400 (0, 0, 1, 0, 1, 1): 57600	b9,b8,b7,b6,b5,b4 (0, 0, 1, 1, 0, 0): 115200 (0, 0, 1, 1, 0, 1): 187500 (0, 0, 1, 1, 1, 0): 230400 (0, 0, 1, 1, 1, 1): 460800 (0, 1, 0, 0, 0, 0): 921600

According to the bit setting provided in the above table, the serial port parameter is set to SD2590.

At present, the setting is basically the same as 3V. As a higher baud rate can be set, the bits used for enabling STX and ETX need to be moved back by two bits.

For instance, set serial port parameters: baud rate: 115200, stop bit: 1, data bit: 8, parity bit: none.

Then set the value HC1 (K1217) on SD2590.

At present, the parameters directly set to SD2590 are only valid in the first cycle of PLC RUN.

If the parameters need to be modified during RUN, they can be set by PORTPARAM instruction.

②Protocol settings SD2592

Protocol settings			
0H	Wecon Modbus slave station	10H	User-defined protocol
2H	Modbus RTU slave station	20H	Modbus RTU master station
3H	Modbus ASCII slave station	30H	Modbus ASCII master station

The corresponding protocol can be set by setting the corresponding value in SD2592.

At present, the parameters directly set to SD2590 are only valid in the first cycle of PLC RUN.

If the parameters need to be modified during RUN, they can be set by PROTOCOL instruction.

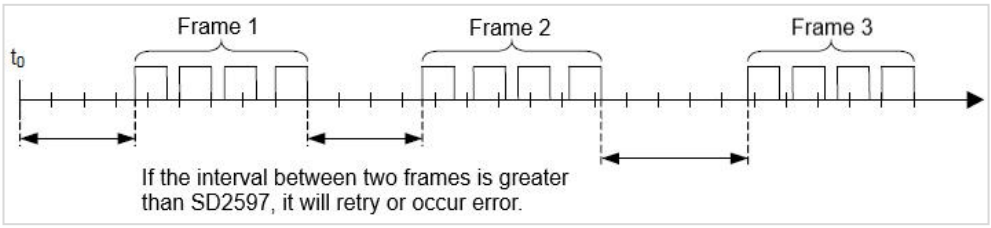
At present, COM1 cannot support the protocol other than W Modbus slave protocol.

③Sending interval SD2596

The main function of sending interval is: to determine how long does it take to wait to send the next instruction when one instruction is completed. When 0 is set, it will not wait for sending at specified interval, but it will be affected by scanning period.

The unit of transmission interval is 0.1 ms, i.e. the interval time is 10ms when the value is set to be 100.

④Communication timeout SD2597 and timeout retry SD2598



The main function of communication timeout is: to determine when the retry is required or an error needs to be reported when the data is not received after being sent by PLC. When 0 is set, it is 100ms by default.

The unit of transmission interval is 10ms, i.e. the timeout judgment time is 100ms when the value is set to be 10.

When receiving timeout occurs, it will judge whether there are retries and the current retry times. If the retry times are greater than or equal to SD2598, an error will be reported.

If SD2598 is set to 0, no retry is required.

When it is set to 1, send the data once and then send it again after timeout.

(3) Serial port parameter setting

① PORTPARA instruction

-[PORTPARA (s) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Serial port parameters to be set 0~256	-	Unsigned BIN16 bits	ANY16	INT
(n)	Which communication port to be set, 0 means COM1, 1 means COM2	-2147483648~ 2147483647	Unsigned BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PORTPARA	Parameter 1												•	•	•	•	•	•	•	•			•	•		•	•
	Parameter 2												•	•	•	•	•	•	•	•			•	•		•	•

Function

Used to set communication parameters at the time of RUN.

For details, please refer to "PORTPARA Instruction Description".

② PROTOCOL instruction

-[PROTOCOL (s) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Protocol number to be set	0 to 65535	Unsigned BIN16 bits	ANY16	INT
(n)	Which communication port to be set, 0 means COM1, 1 means COM2	1	Unsigned BIN16 bits	ANY16	INT

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PROTOCOL	Parameter 1												•	•	•	•	•	•	•	•			•	•		•	•
	Parameter 2												•	•	•	•	•	•	•	•			•	•		•	•

Function

It is used to set the communication protocol at the time of RUN, and the above instruction is actually used to set a specific value for the parameter modification sign.

The specific calculation formula is: (parameter setting value + offset of the corresponding serial port's initial special device + 10)*2

For example, setting the protocol to Modbus RTU master station (H20), i.e. (0x20 +2593-2590 +10) * 2, which is (32 +3 +10) *2=90, which mainly aims to avoid random modification of serial port parameters during operation.

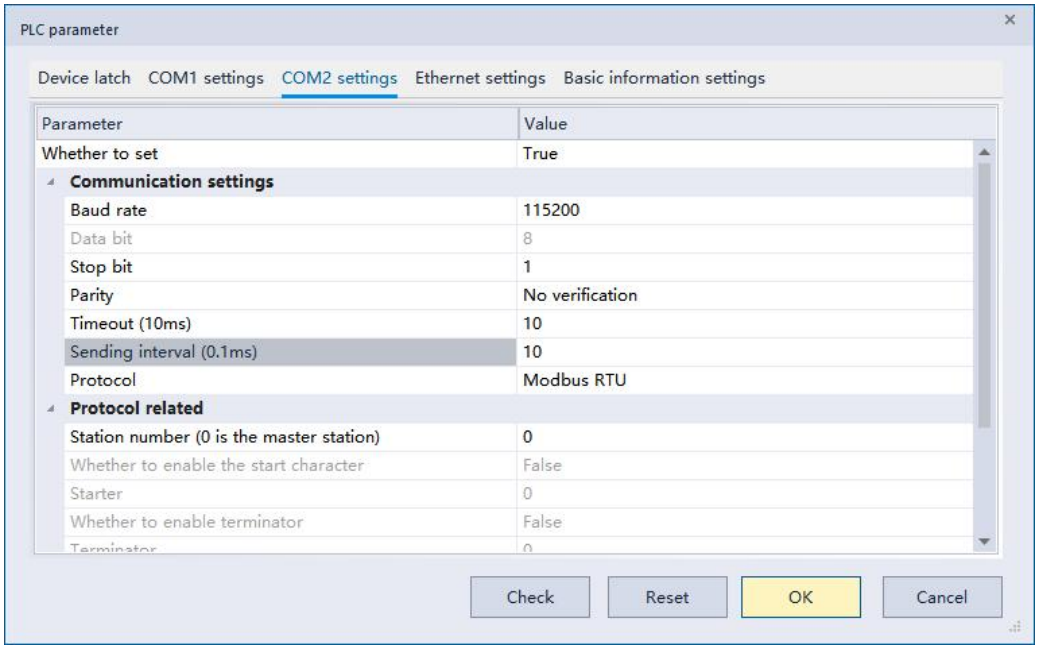
For Modbus master station protocol, if the set parameters are triggered, the value will be changed only after the instruction is completely completed.

For Modbus slave station protocol, if the set parameters are triggered, switching can be performed when the received data is not being processed.

For details, please refer to "PROTOCOL Instruction Description".

③Upper computer settings

The PLC parameter setting of the upper computer can set the corresponding serial port parameters.



Specific protocol station number cannot be 0

Station number under Modbus RTU and Modbus ASCII protocol is 0: Protocol sets the master station.

Station number under Modbus RTU and Modbus ASCII protocol is non-0: Protocol sets the slave station.

Serial port parameters are filled in according to the contents in the table.

Note: RTU protocol cannot set 7-bit data bits.

④Priority description of serial port parameters

The priority of serial port parameter settings is listed below.

At present, serial port parameters are saved at the time of power failure.

Serial port parameter setting instruction setting = using ladder diagram MOV instruction to set corresponding SD device > upper computer downloading parameters > previous data saved at the time of power outage.

(4) Basic ladder diagram



Serial port parameter setting
Baud rate: 115200, data bit: 8, parity bit: none, stop ...
Communication protocol setting:
Sending interval: 5ms
Reception timeout: 100ms
Timeout retry times: 3 times
Station number 01H, function code 03H, slave station

10.5.2.3 Sending and receiving process

(1) Modbus master station

When programming, in front of each RS (Modbus mode) instruction, each operand unit should be assigned with a certain value according to the address of the communication operation object, operation type, operation register address, data number, sending or receiving unit. Once the execution starts, the system program will automatically calculate CRC parity, organize communication frames, and complete the operation of sending data and receiving response.

If Modbus-ASC protocol is used for communication (SD2592 is set to H30), the HEX-ASC format for sending and receiving data is automatically changed by PLC system program, and the method of using RS (Modbus mode) instruction by users is exactly the same as that of using Modbus-RTU protocol.

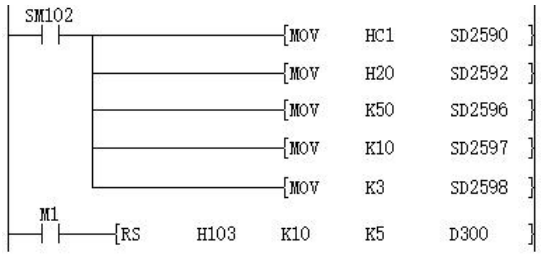
In the plc program, if multiple RS (Modbus Mode) instructions are driven, when the system program is executed, the next RS

instruction will be processed in the same way after the links of “sending, waiting for response, receiving, checking, analyzing and storing" of one RS instruction are completed, until all RS instructions are executed and started again. Users don't need to care about the timing and process of its execution, which simplifies PLC programming design. This is the advantage of Modbus instruction.

List of Modbus Master Station Function

Function code	Function name	Details
0x01	Coil state readout	Coil state readout (several points)
0x02	Input readout	Input readout (several)
0x03	Holding register readout	Holding register readout (several points)
0x04	Input register readout	Input register readout (several points)
0x05	1 Coil written	Coil written (only one point)
0x06	1 Register written	Holding register written (only one point)
0x0F	Batch coil written	Multi-point coil written
0x10	Batch register written	Multi-point holding register written

For instance:



Serial port parameter setting
Baud rate: 115200, data bit: 8, parity bit: none, stop
Communication protocol setting:
Sending interval: 5ms
Reception timeout: 100ms
Timeout retry times: 3 times
Station number 01H, function code 03H, slave station

As shown in above ladder diagram:

When M1 is turned ON, PLC will send data (hexadecimal) from COM2 of PLC: 01 03 00 0A 00 05 A5 C8

01: represents slave address, the high 8 bits of (s);

03: Modbus instruction code, the low 8 bits of (s), which means to read the slave register;

00 0A: the address of slave register to be read, the value of (m);

00 05: The number of registers to be read, the value of (d),

A5 C8: CRC parity code

Notes

With 1 coil written (0x05), if 1 is written, the used value is 0xFF00; if 0 is written, the used value is 0x0000.

With batch coil written (0x0F), pay attention to the high and low bytes switching. For example, when writing 1 length, the fourth parameter value is 1 (0x0001), which means sending 0, and only when 256 (0x0100) is written can 1 be written as the target address.

(2) Modbus slave station

When plc is used as Modbus slave station, it supports communication operation instructions such as 0x01, 0x03, 0x05, 0x06, 0x0f and 0x10 of Modbus; through these instructions, the coils that can read and write PLC have variables such as M, S, T, C, X (read-only) SM, Y, LC, HSC, etc. Register variables include D, T, C, R, SD, LC and HSC.

When the Modbus communication host accesses (reads or rewrites) the internal variables of PLC slave, it must follow the following definition of communication instruction frame and variable address index method, so as to carry out normal communication operation.

①Instruction code 0x01 (01): read coil

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x01 (instruction code)	1 byte	Read coil
3	Start address of coil	2 bytes	High bit in the front and low bit in the rear, see the addressing of the coil
4	Coil number	2 bytes	High bit in the front and low bit in the rear (N)
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Response frame format: slave address + 0x01 + bytes + coil status + CRC parity

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x01 (instruction code)	1 byte	Read coil
3	Number of bytes	1 byte	Value: [(N+7)/8]
4	Coil state	[(N+7)/8] bytes	Every 8 coils are combined into one byte. If the last coil is less than 8 bits, the undefined part is filled with 0. The first 8 coils are in the first byte, and the coil with the smallest address is in the lowest bit. And so forth.
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Error response: see error response frame

② Instruction code 0x03 (03): read register

Request frame format: slave address+ 0x03 + register start address + register number + CRC parity.

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x03 (instruction code)	1 byte	Read register
3	Register start address	2 bytes	High bit in the front and low bit in the rear, see the addressing of the register
4	Register number	2 bytes	High bit in the front and low bit in the rear (N)
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x03 (instruction code)	1 byte	Read register
3	Number of bytes	1 byte	Value: N*2
4	Register value	N*2 bytes	Every two bytes represents a register value, with high bit in the front and low bit in the rear. Register with small address comes first

No.	Data (byte) meaning	Number of bytes	Description
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Response frame format: slave address+0x03+bytes+register value +CRC parity.

Error response: see error response frame

③ Instruction code 0x05 (05): write single coil

Response frame format: slave address + 0x05+ bytes + coil address + CRC parity

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x05 (instruction code)	1 byte	Write single coil
3	Coil address	2 bytes	High bit in the front and low bit in the rear, see the addressing of the coil
4	Coil state	2 bytes	High bit in the front and low bit in the rear. If the value is not 0, it is valid.
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Response frame format: slave address+0x05+coil address+ coil state+ CRC parity.

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594.
2	0x05 (instruction code)	1 byte	Write single coil
3	Coil address	2 bytes	High bit in the front and low bit in the rear, see the addressing of the coil
4	Coil state	2 bytes	High bit in the front and low bit in the rear. If the value is not 0, it is valid.
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Error response: see error response frame

Wherein the coil state 0xFF00 means being set as ON and 0x0000 means being set as OFF.

④ Instruction code 0x06 (06): write single register

Request frame format: slave address+ 0x06 + register address + register value + CRC parity.

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x06 (instruction code)	1 byte	Write single register
3	Register address	2 bytes	High bit in the front and low bit in the rear, see the addressing of the register
4	Register value	2 bytes	High bit in the front and low bit in the rear. If the value is not 0, it is valid.
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Response frame format: slave address+0x06+register address+ register value+ CRC parity.

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x06 (instruction code)	1 byte	Write single register
3	Register address	2 bytes	High bit in the front and low bit in the rear, see the addressing of the register
4	Register value	2 bytes	High bit in the front and low bit in the rear. If the value is not 0, it is valid.
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Error response: see error response frame

⑤ Instruction code 0x0f (15): write multiple coils

Request frame format: slave address + 0x 0f+ coil start address + coil number+ number of bytes+ coil state+ CRC parity.

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x 0f (instruction code)	1 byte	Write multiple coils
3	Start address of coil	2 bytes	High bit in the front and low bit in the rear, see the addressing of the coil
4	Coil number	2 bytes	High bit in the front and low bit in the rear. N, maximum value: 1968
5	Number of bytes	1 byte	Value: [(N+7)/8]
6	Coil state	[(N+7)/8] bytes	Every 8 coils are combined into one byte. If the last coil is less than 8 bits, the undefined part is filled with 0. The first 8 coils are in the first byte, and the coil with the smallest address is in the lowest bit. And so forth.
7	CRC parity	2 bytes	High bit in the front and low bit in the rear

Response frame format: slave address+ 0x 0f +coil start address+ coil number+ CRC parity

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x 0f (instruction code)	1 byte	Write multiple coils
3	Start address of coil	2 bytes	High bit in the front and low bit in the rear, see the addressing of the coil
4	Coil number	2 bytes	High bit in the front and low bit in the rear.
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Error response: see error response frame

⑥ Instruction code 0x10 (16): write multiple registers

Request frame format: slave address+ 0x10 + register start address + register number +number of bytes+ register value+ CRC parity.

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x10 (instruction code)	1 byte	Write multiple registers
3	Register start address	2 bytes	High bit in the front and low bit in the rear, see the addressing of the register
4	Register number	2 bytes	High bit in the front and low bit in the rear. N, maximum value: 120
5	Number of bytes	1 byte	Value: N*2
6	Register value	N*2 (N*4)	
7	CRC parity	2 bytes	High bit in the front and low bit in the rear

Response frame format: slave address+0x10+ register start address+ register number + CRC parity.

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	0x10 (instruction code)	1 byte	Write multiple registers
3	Register start address	2 bytes	High bit in the front and low bit in the rear, see the addressing of the register
4	Register number	2 bytes	High bit in the front and low bit in the rear. N, maximum value: 120
5	CRC parity	2 bytes	High bit in the front and low bit in the rear

Error response: see error response frame

⑦ Error response frame

Error response: slave address + (instruction code + 0x80) + error code + CRC parity.

No.	Data (byte) meaning	Number of bytes	Description
1	Slave address	1 byte	Take value 1~247, which is set by SD2544 and SD2594
2	Instruction code+0x80	1 byte	Error instruction code
3	Error code	1 byte	1 to 4
4	CRC parity	2 bytes	High bit in the front and low bit in the rear

No.	Error code	Description
1	01	Not supported function codes
2	02	Wrong address or function code
3	03	Wrong length
4	04	Incomplete instruction
5	05	Address not allowed

⑧Slave station address table

Word address				
Address type	Occupation	Address range	Decimal address	Total reserved address size
T0 to T511	512 WORD	0x0000 to 0x01ff	0	1536
C0~C255	256 WORD	0x0600 to 0x06ff	1536	1024
LC0 to LC255	512 WORD	0x0A000 to 0x0BFF	2560	1024
HSC0 to HSC15	128 WORD	0x0E00 to 0x0E1F	3584	512
D0~D7999	8000 WORD	0x1000 to 0x2F3F	4096	16384
SD0 to SD4095	4096 WORD	0x5000 to 0x5FFF	20480	12288
R0~R30000	30000 WORD	0x8000 to 0xF52F	32768	30000

Bit address				
Address type	Occupation	Address range	Decimal address	Total reserved address size
T0 to T511	512 bit	0x0000 to 0x01ff	0	1536
C0~C255	256 bit-	0x0600 to 0x06ff	1536	1024
LC0 to LC255	256 bit-	0x0A00 to 0x0AFF	2560	1024
HSC0 to HSC15	64 bit	0x0E00 to 0x0E0F	3584	512
M0 to M8000	8192 bit	0x1000 to 0x2F3F	4096	16384
SM0~SM4095	4096 bit	0x5000 to 0x5FFF	20480	12288
Reserved		0x8000 to 0xBFFF		16383
S0 to S4095	4096 bit	0xC000 to 0xCFFF	49152	8192
X0 to X1777	1024 bit-	0xE000 to 0xE3FF	57344	4096
Y0 to Y1777	1024 bit-	0xF000 to 0xF3FF	61440	4096

10.5.2.4 Error information

The current error code will be displayed in the SD7, SD0, SD2611 and SD2612.

The error codes that will occur in Modbus protocols are mainly shown as follows:

Error code	Content
4085H	(s), (m), (d) and (n) read address exceeds the range of the device (this error is displayed in SD7 and SD0 only).
3180H	COM2 data reception error. There may be interference on the communication cable, it is recommended to connect GND.
3181H	COM2 data reception timeout. Check the wiring, whether the serial port parameter settings correspond to master and slave, and whether there is interference. Check whether it is caused by the late response from slave station. It is recommended to increase the sending interval SD2546.
3182H	COM2 CRC parity error. There may be interference on the communication cable, it is recommended to connect GND.
3183H	COM2 LRC parity error. There may be interference on the communication cable, it is recommended to connect GND.
3184H	The COM2 station number is incorrectly configured. Check the slave station number setting, and check whether there is any problem with the receiving and sending mechanism from slave station.
3185H	COM2 send buffer overflow If this error message appears, please contact technical personnel.
3186H	COM2 function code is wrong. Check whether the set function code is a function code supported by PLC.
3187H	COM2 address is wrong. Check whether the slave station has this address. (Please refer to Modbus exception 02).
3188H	The length of COM2 is wrong. Check whether the communication length exceeds the length range specified by Modbus protocol or user-defined protocol.
3189H	COM2 data error. Check whether parameters of the instruction are wrong. Check whether the set value is supported by the slave station. (Please refer to Modbus exception 03).
318AH	COM2 slave station is busy. Message returned from slave station: Slave station is busy. (Please refer to Modbus exception 06).
318BH	COM2 slave station does not support function codes. Check whether this function code is supported by the slave station. (Please refer to Modbus exception 01).
318CH	COM2 slave station is faulty. Message returned from slave station: Slave station failed, please check whether slave station failed. (Please refer to Modbus exception 04).
318DH	COM2 slave station confirmation. Message returned from slave station: slave station confirmation. (Please refer to Modbus exception 05).
318EH	At present, COM2 does not support this instruction or function. When the slave protocol is set, the communication instructions related to master station or master station functions cannot be used. Please change the protocol or close the contact before the corresponding instruction or close the corresponding communication function.
318FH	COM2 sending timeout. If this error message appears, please contact technical personnel.
3190H	COM2 receives data beyond the cache limit.
31A0H	No available gateway for COM2 Message returned from the slave station: no available gateway. (Please refer to Modbus exception 0A).
31A1H	COM2 indicates that no response was obtained from the target device. Message returned from the slave station: The device is not networked. (Please refer to Modbus Exception 0B).

After resetting the protocol or communication parameters, the reported errors will be cleared.

In addition, in case of communication completion/communication error/communication timeout, the sign bit will be set after executing the instruction.

10.6 WECON Modbus protocol description

At present, WECON Modbus protocol description (special protocol) is modified based on Modbus RTU protocol.

Therefore, 7-bit data bits cannot be used in serial port parameters.

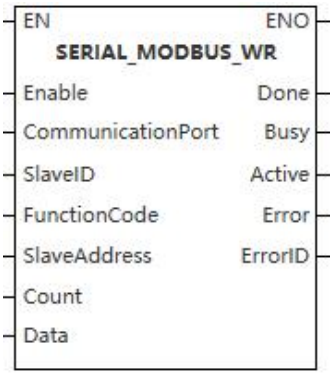
This protocol is fully compatible with Modbus RTU protocol, and the address is the same as the default address of Modbus RTU protocol of LX6V PLC.

The extended function is mainly used to communicate with PLC Editor 2.

10.7 Communication function block

1. SERIAL_MODBUS_WR

SERIAL_MODBUS_WR - Modbus master serial communication



This function block is the setting function block for sending protocol frames by the Modbus master station (whether RTU or ASCII). When the contact in front of Enable is turned ON, according to the communication port, station number, function code, slave station address, quantity and mapped data address set by the function block, the data frame is combined and sent.

Device used

Instruction	Parameter	Device																								Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SERIAL_ MODBUS_WR	Enable	●	●	●	●	●																					
	CommunicationPort															●	●	●	●	●							
	SlaveID															●	●	●	●	●							
	FunctionCode															●	●	●	●	●							
	SlaveAddress															●	●	●	●	●							
	Count															●	●	●	●	●							
	Data																	●	●								
	Done		●	●	●	●																					
	Busy		●	●	●	●																					
	Active		●	●	●	●																					
	Error		●	●	●	●																					
	ErrorID																	●	●								

Variable Type Used

Instruction	Pin Type	Parameter	Variable Type	Can you Is empty	Range	Description
SERIAL MODBUS_WR	Input	Enable	BOOL	No	TRUE/FALSE	Enable sending Modbus master serial communication data
		CommunicationPort	UINT	No	[0,1]	Communication port 0: Serial port 1 1: Serial port 2
		SlaveID	UINT	No	[0,255]	Station number of slave station
		FunctionCode	UINT	No	1/2/3/4/5/6 /15/16	Function code

Instruction	Pin Type	Parameter	Variable Type	Can you Is empty	Range	Description
		SlaveAddress	UINT	No	[0,65535]	The slave address (the address provided by the slave station) will read or write data from this address of the slave station.
		Count	UINT	No	-	Number The number of Modbus reads or writes, and the unit is determined by the function code.
		Data	ANY	No	-	The starting address for reading or writing data
	Output	Done	BOOL	Yes	TRUE/FALSE	Instruction completed
		Busy	BOOL	Yes	TRUE/FALSE	Serial port busy
		Active	BOOL	Yes	TRUE/FALSE	Instruction working
		Error	BOOL	Yes	TRUE/FALSE	Error
		ErrorID	DWORD	Yes	Refer to ErrorID for details	Error code

Function

Enable the MB_SERIAL function block to send the specified Modbus master serial communication data for reading from or writing to a slave station.

Notice

(1) If and only if the function block Enable is enabled, update the current serial communication instruction sending status;

(2) If two or more serial port instructions use the same serial port in the scanning program, the corresponding serial port instructions will wait for the execution of the previous serial port instruction to be completed according to the instruction order, and then execute their own communication instructions;

(3) If two or more serial port instructions use different serial ports in the scanning program, the corresponding serial port instructions will be executed in sequence according to the instruction order without affecting each other;

(4) Busy means that the serial port is busing and Active means that the current instruction is being executed.

Modbus protocol

When the protocol is set to Modbus master protocol (whether RTU or ASCII). When the contact in front of the MB_SERIAL function block is ON, the MB_SERIAL function block will send a combined data frame at the designated communication port according to the station number, function code, slave station address, quantity, and data starting address.

Error code

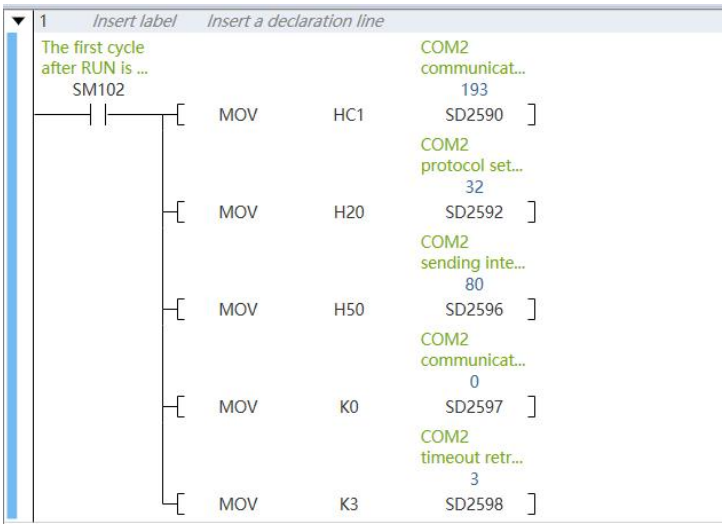
Error code	Content
4084H	CommunicationPort input error (0: Serial port 1, 1: Serial port 2).
408FH	The firmware does not support this instruction, and needs to be upgraded.
4096H	The settings of Count and Data are outside the range of variables.
3180H	COM2 reception error There may be interference on the communication cable, it is recommended to connect GND.
3181H	COM2 data reception timeout. Check the wiring, whether the serial port parameter settings correspond to master and slave, and whether there is interference. Check whether it is caused by the late response from the slave station. It is recommended to increase the sending interval SD2546.
3182H	COM2 CRC parity error. There may be interference on the communication cable, it is recommended to connect GND.
3183H	COM2 LRC parity error. There may be interference on the communication cable, it is recommended to connect GND.

Error code	Content
3184H	The COM2 station number is configuration error Check the slave station number setting, and check whether there is any problem with the receiving and sending mechanism from slave station.
3185H	COM2 send buffer overflow If this error message appears, please contact technical personnel.
3186H	COM2 function code is error. Check whether the set function code is a function code supported by PLC.
3187H	COM2 address error. Check whether the slave station has this address. (Please refer to Modbus exception 02)
3188H	The length of COM2 is wrong. Check whether the communication length exceeds the length range specified by Modbus protocol or user-defined protocol.
3189H	COM2 data error. Check whether parameters of the command are wrong. Check whether the set value is supported by the slave station. (Please refer to Modbus exception 03).
318AH	COM2 slave station is busy. Slave station return information: Slave station is busy (Please refer to Modbus exception 06).
318BH	COM2 slave station does not support function codes. Check whether this function code is supported by the slave station. (Please refer to Modbus exception 01)
318CH	COM2 slave failure. Slave station return information: Slave station failure, please check whether the slave station fails. (Please refer to Modbus exception 04)
318DH	COM2 slave station confirmation. Slave station return information: Slave confirmation (Please refer to Modbus exception 05)
318EH	At present, COM2 does not support this command or function. When the slave protocol is set, the communication commands related to master station or master station functions cannot be used. Please change the protocol or close the contact in front of the corresponding command or close the corresponding communication function.
318FH	COM2 sending timeout. If this error message appears, please contact technical personnel.
31A0H	No available gateway for COM2 Slave station return information: Unavailable gateway (Please refer to Modbus exception 0A)
31A1H	COM2 indicates that no response was obtained from the target device. Slave station return information: The device is not in the network. (Please refer to Modbus exception 0B)

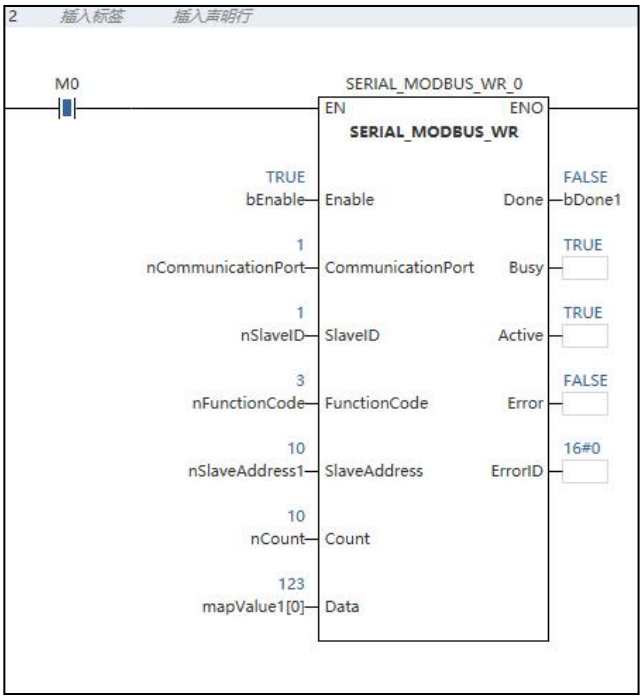
Example

Application example

Set the COM2 communication parameters through the SD device.

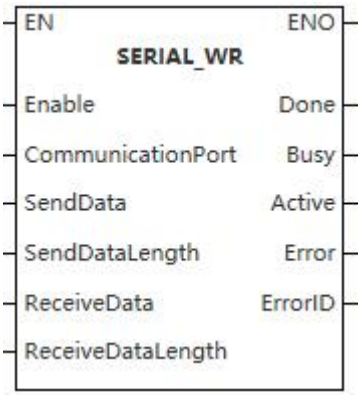


Enable the function block and its Enable pin, send Modbus serial port master data and wait for the slave respond. Done indicates that the current instruction has been executed, Busy indicates that the serial port is busy, and Active indicates that the current instruction is running (waiting for data to be returned from the slave station).



2. SERIAL_WR

SERIAL_WR - Custom protocol serial communication



This function block is a setting function block for user defined protocol serial port to send protocol frames. When the contact in front of Enable is turned ON. According to the communication port, sending data mapping address, sending data length, receiving data mapping address, and receiving data length set by the function block, the data frame is sent and waits for the data of the specified length to be returned.

Device used

Instruction	Parameter	Device																				Offset Modification	Pulse Expansion
		X	Y	M	S	SM	T(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SERIAL_WR	Enable	●	●	●	●	●																			
	CommunicationPort													●	●	●	●	●							
	SendData															●	●								
	SendDataLength													●	●	●	●	●							
	ReceiveData															●	●								
	ReceiveDataLength													●	●	●	●	●							
	Done		●	●	●	●																			
	Busy		●	●	●	●																			
	Active		●	●	●	●																			
	Error		●	●	●	●																			
	ErrorID															●	●								

Variable Type Used

Instruction	Pin Type	Parameter	Variable Type	Can it be empty	Range	Description
SERIAL_WR	Input	Enable	BOOL	No	TRUE/FALSE	Enable sending protocol-less serial communication data
		CommunicationPort	UINT	No	[0,1]	Communication port, 0: Serial port 1, 1: Serial port 2
		SendData	ANY	No	-	Starting address for sending data
		SendDataLength	UINT	No	[0,523]	Send data length, in bytes
		ReceiveData	ANY	No	-	Starting address for receiving data
		ReceiveDataLength	UINT	No	[0,523]	Length of received data, in bytes
	Output	Done	BOOL	Yes	TRUE/FALSE	Instruction completed
		Busy	BOOL	Yes	TRUE/FALSE	Serial port busy

Instruction	Pin Type	Parameter	Variable Type	Can it be empty	Range	Description
		Active	BOOL	Yes	TRUE/FALSE	Instruction working
		Error	BOOL	Yes	TRUE/FALSE	Error
		ErrorID	DWORD	Yes	Refer to ErrorID for details	Error code

Function

▪ Enable the PROTOCOLLESS_SERIAL function block, which can send the serial communication data of the specified custom protocol master station and wait for the data return of the slave station.

Notice

▪ If and only if the function block Enable is enabled, updating the current serial communication instruction sending state;

▪ If two or more serial communication instructions are used simultaneously in the scanning program, the corresponding communication instructions are sequentially executed according to the instruction sequence;

▪ Busy indicates that the serial port is busy, and Active indicates that the current instruction is being executed.

User-defined protocol

When the protocol is set to a custom protocol. When the contact in front of the PROTOCOLLESS_SERIAL function block is ON， The PROTOCOLLESS_ SERIAL function block will send the data starting address and sending data length in the specified communication port to send the combined data frame, and wait for the data of the received data length to be filled in the received data starting address.

▪ When the length of the transmitted data is set to 0, the mapping address of the transmitted data may not be filled in. At this time, the PROTOCOLLESS_SERIAL function block only receives data;

▪ When the received data length is set to 0, the received data mapping address may not be filled in. At this time, the PROTOCOLLESS_SERIAL function block only sends data.

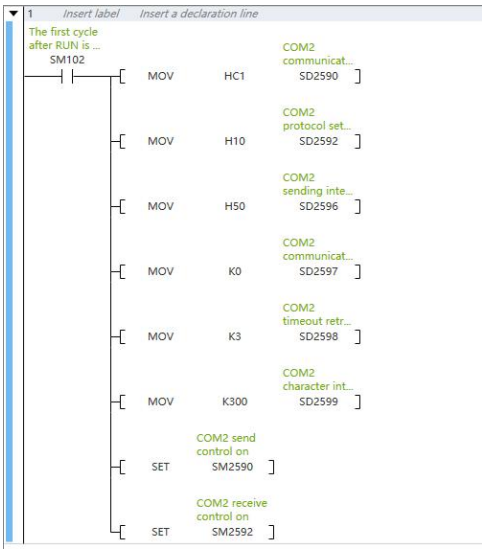
Error code

Error code	Content
4084H	CommunicationPort input error (0: Serial port 1, 1: Serial port 2).
5:	The setting of SendDataLength and SendData or ReceiveDataLength and ReceiveData is outside the scope of the variable.
3181H	Data reception timeout
318EH	At present, COM2 does not support this command or function. When set to the slave protocol, the communication instructions related to the master station or the master station function cannot be used. Please change the protocol or close the contact in front of the corresponding command or close the corresponding communication function.
318FH	COM2 data reception timeout. ▪ Check wiring; ▪ Check whether the serial port parameter settings are master-slave corresponding; ▪ Check whether there is any interference; ▪ Check whether it is caused by the late response from the slave station. It is recommended to increase the sending interval SD2546.
3190H	COM2 receives data beyond the cache limit.

Example

Application example

- Set the COM2 communication parameters through the SD device.



Enables the function block and the Enable pin of the function block, sends custom protocol data and waits for the slave to respond. Done indicates that the current instruction has been executed, Busy indicates that the serial port is busy, and Active indicates that the current instruction is running (waiting for data to be returned from the slave station).

11 PID Control Instruction

11.1 PID/PID calculation

PID

This instruction is used to perform PID control that changes the output value according to the amount of input change.

-[PID (s1)(s2)(s3)(d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Device number for storing the target value (SV)	-32767 to 32767	Signed BIN 16 bit	ANY16
(s2)	Device number for storing the measured value (PV)	-32767 to 32767	Signed BIN 16 bit	ANY16
(s3)	Device number for storing parameters	1 to 32767	Signed BIN 16 bit	ANY16
(d)	Device number for storing output value (MV)	-32767 to 32767	Signed BIN 16 bit	ANY16

Device used

Instruction	Parameter	Devices																							Offset modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PID	Parameter 1																	●	●	●							
	Parameter 2																	●	●	●							
	Parameter 3																	●	●	●							
	Parameter 4																	●	●	●							

Features

This instruction is to complete the PID operation and is used to control the parameters of the closed-loop control system. PID control has a wide range of applications in mechanical equipment, pneumatic equipment, constant pressure water supply and electronic equipment, etc. among them:

(S1) is the target value of PID control;

(S2) is the measured feedback value;

(S3) The starting address of the buffer area for setting parameters required for PID operation and saving intermediate results, occupies a total of 26 variable units in the subsequent addresses, the value range is D0 to D7974, it is best to specify the power failure retention area, which will remain after the power is OFF Set the value, otherwise the buffer area needs to be assigned value before starting the operation for the first time. The function and parameter description of each unit are described in this section;

(D) is the storage unit of the PID calculation result. Please designate (D) as a non-battery holding area, otherwise it needs to be initialized and cleared before starting the calculation for the first time.

Programming example



The parameter description is as follows:

What is stored in D9 is the target value of PID adjustment, and D10 is the closed-loop feedback value. Note that D9 and D10 must be of the same dimension, such as both 0.01MPa units, or 1℃ units, etc.;

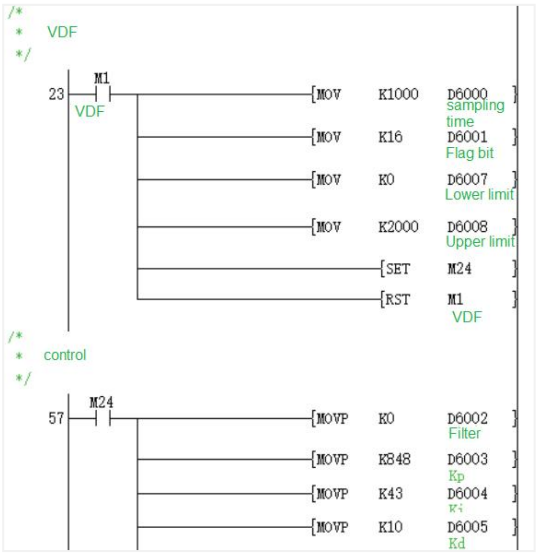
A total of 26 units of D200 to D225 are used to store the set value and process value of PID operation. These values must be set item by item before the first PID calculation;

The D130 unit is used to store the calculated control output value to control the execution of the action.

The function and setting method of the parameter value of each unit about starting of (S3) are described in the following table:

Unit	Features	Setting instructions
(S3)	Sampling time (TS)	The setting range is 1 to 32767 (ms), but it needs to be greater than the PLC program scan period
(S3)+1	Action direction (ACT)	bit0: 0 = positive action; 1 = reverse action bit3: 0 = unidirectional; 1 = bidirectional bit4: 0 = auto-tuning does not work; 1 = auto-tuning is executed, others cannot be used.
(S3)+2	Maximum ascent rate (DeltaT)	Setting range 0 to 32000 is the threshold of integral increment
(S3)+3	Proportional gain (Kp)	Setting range 0 to 32767, note that this value is enlarged by 256 times, the actual value is Kp/256
(S3)+4	Integral gain (Ki)	Setting range 0 to 32767, Ki=16384Ts/Ti, Ti is the integral time
(S3)+5	Differential gain (Kd)	Setting range 0 to 32767, Kd≈Td/Ts, Td is the derivative time
(S3)+6	Filtering (C0)	Setting range 0 to 1023, integral part filtering
(S3)+7	Output lower limit	Recommended setting range -2000 to 2000 When bit3 of S3+1=0, set to 0; When bit3 of S3+1=1, set to -2000;
(S3)+8	Output upper limit	Recommended setting value 2000
(S3)+9	Reserved	Reserved
⋮	⋮	⋮
(S3)+25	Reserved	Reserved

Auto tuning example



✎Note:

• When multiple instructions are used, the device number of (d) cannot be repeated.

• During the execution of auto-tuning, the (s3) parameter space cannot be modified.

• The instruction occupies 26 point devices from the device specified in (s3).

• PID instruction can be used multiple times in the program and can be executed at the same time, but the variable area used in each PID instruction should not overlap; it can also be used in step instructions, jump instructions, timing interrupts, and subroutines, in this case When executing the PID instruction, the (s3)+9 cache unit must be cleared in advance.

• The maximum error of the sampling time Ts is -(1 operation cycle +1ms) +(1 operation cycle). If the sampling time Ts ≤ 1 operation cycle of the programmable controller, the following PID operation error (4D86H) will occur, and the PID operation will be executed with TS = operation cycle. In this case, it is recommended to use constant scan mode or use PID instruction in timer interrupt.

Error code

Error code	Content
4085H	When the device specified in the read application instructions (s1), (s2), (s3), (d) exceeds the range of the corresponding device.
4086H	When the device specified in the write application instruction (s3) and (d) exceeds the range of the corresponding device.
4D80H	The sampling time is out of range. $Ts \le 0$
4D81H	Input filter constant (Co) is out of range (. $Co < 0$Or $Co \ge 1023$)
4D82H	The maximum ascent rate (ΔT) is out of range. $\Delta T < 0$Or$\Delta T > 32000$
4D83H	The proportional gain (Kp) is out of range. $Kp < 0$
4D84H	The integral gain (Ki) is out of range. $Ki < 0$
4D85H	The differential gain (Kd) is out of range. $Kd < 0$
4D86H	The sampling time (Ts) is less than the operation cycle. $Ts <$ Scan cycle

11.2 CCPID/CCPID calculation

CCPID

This instruction is used to perform PID control that changes the output value according to the amount of input change.

-[CCPID (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s1)	Device number for storing the target value (SV)	-32767 to 32767	Signed BIN 16 bit	ANY16
(s2)	Device number for storing the measured value (PV)	-32767 to 32767	Signed BIN 16 bit	ANY16
(s3)	Device number for storing parameters	1 to 32767	Signed BIN 16 bit	ANY16
(d)	Device number for storing output value (MV)	-32767 to 32767	Signed BIN 16 bit	ANY16

Device used

Instruction	Parameter	Devices																								Offset modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CCPID	Parameter 1																	●	●	●							
	Parameter 2																	●	●	●							
	Parameter 3																	●	●	●							
	Parameter 4																	●	●	●							

Features

After setting target value (s1), measured value (s2), parameter (s3) to (s3) +12 and executing the program, the calculation result (MV) will be stored to the output according to the first sampling time (s3) in the parameter Value (d).

✎Note:

It can be executed multiple times at the same time (there is no limit to the number of loops), but please note that the device numbers (s3) and (d) used in the calculation cannot be repeated.

The instruction occupies 52 points of devices starting from the device specified in (s3).

During the execution of auto-tuning, the (s3) parameter space cannot be modified.

Error code

Error code	Content
4085H	When the device specified in the read application instructions (s1), (s2), (s3), (d) exceeds the range of the corresponding device.
4086H	When the device specified in the write application instruction (s3) and (d) exceeds the range of the corresponding device.
4D80H	The sampling time is out of range. $Ts \leq 0$
4D81H	Input filter constant (Co) is out of range ($Co < 0 \mathrm{Or} Co \geq 1023$)
4D82H	The maximum ascent rate (ΔT) is out of range. $\Delta T < 0 \mathrm{OR}\ \Delta T > 32000$
4D86H	The sampling time (Ts) is less than the operation cycle. $Ts < \text{Scan cycle}$
4D87H	The proportional gain (Kp) is out of range. $Kp < 1 \mathrm{Or} Kp > 30000$
4D88H	The integral time constant (Ti) is out of range. $Ti < 0 \mathrm{Or} Ti > 3600$
4D89H	The differential time constant (Td) is out of range. $Td < 0 \mathrm{Or} Td > 1000$
4D90H	The upper limit of CCPID output is less than the lower limit.

Example

See "CCPID Instruction Manual".

11.3FPID/FPID calculation

FPID

The function of this instruction is to adjust PID control parameters by fuzzy algorithm.

-[FPID (s) (d1) (d2) (d3)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)
(s)	Store the start number of the device of the fuzzy parameter table (no input required)	-	Signed BIN 16 bit	ANY16
(d1)	Start number of the device storing the initialization parameters	-	Signed BIN 16 bit	ANY16
(d2)	Store the start number of the device of the input PID parameter	-	Signed BIN 16 bit	ANY16
(d3)	The start number of the device that stores the adjusted PID parameters	-	Signed BIN 16 bit	ANY16

Device used

Instruction	Parameter	Devices																								Offset modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
FPID	Parameter 1																	●	●	●	●						
	Parameter 2																	●	●	●	●						
	Parameter 3																	●	●	●	●						
	Parameter 4																	●	●	●	●						

Features

This instruction needs to be used in conjunction with the PID instruction. It completes the fuzzy calculation of the adjustments of the three parameters of PID, Kp, Ki, and Kd. By passing in the three parameters of the PID, the new three parameters are calculated and substituted into the PID for output control.

Parameter Description:

S parameter setting					
Parameter	Offset address	Name	Format	Instruction	Range
Parameter 1	S	-	-	-	-

d1 parameter setting					
Parameter	Offset address	Name	Format	Instruction	Range
Parameter 1	d1	em domain	Floating point	Temperature difference	>0
	d1+1				
Parameter 2	d1+2	ecm domain	Floating point	Temperature difference	>0
	d1+3				
Parameter 3	d1+4	kpm coefficient	Floating point	0.5 (fixed) (not set)	-
	d1+5				
Parameter 4	d1+6	kim coefficient	Floating point	1 (fixed) (not set)	-
	d1+7				
Parameter 5	d1+8	kdm coefficient	Floating point	1 (fixed) (not set)	-
	d1+9				
Parameter 6	d1+10	EM	32-bit integer	6 (fixed) (not set)	-
	d1+11				

Parameter 7	d1+12 d1+13	ECM	32-bit integer	6 (fixed) (not set)	-
Parameter 8	d1+14 d1+15	UM	32-bit integer	6 (fixed) (not set)	-
Parameter 9	d1+16 d1+17	Size_x	32-bit integer	13 (fixed) (not set)	-
Parameter 10	d1+18 d1+19	Size_y	32-bit integer	13 (fixed) (not set)	-
Parameter 11	d1+20	Kpm reserved for internal use	Reserved	Reserved	-
Parameter 12	d1+21	Kim reserved for internal use	Reserved	Reserved	-
Parameter 13	d1+22	Kdm reserved for internal use	Reserved	Reserved	-
Parameter 14	d1+23	Kukp reserved for internal use	Reserved	Reserved	-
Parameter 15	d1+24	Kuki reserved for internal use	Reserved	Reserved	-
Parameter 16	d1+25	Kukd reserved for internal use	Reserved	Reserved	-
⋮	⋮	⋮	⋮	Reserved	-
Parameter 20	d1+37	Reserved for internal use	Reserved	Reserved	-

d2 parameter setting					
Parameter	**Offset address**	**Name**	**Format**	**Instruction**	**Range**
Parameter 1	d2	Current Temperature	16-bit integer	Current test temperature	-
Parameter 2	d2+1	set temperature	16-bit integer	Set temperature	-
Parameter 3	d2+2	Calculation period	16-bit integer	Take an integer multiple of the pid sampling time, usually the same	-
Parameter 4	d2+3	Kp	16-bit integer	PID initial Kp value	-
Parameter 5	d2+4	KI	16-bit integer	PID initial Ki value	-
Parameter 6	d2+5	KD	16-bit integer	PID initial Kd value	-
Parameter 7	d2+6	Sampling cycle	16-bit integer	No need to enter	-
Parameter 8	d2+7	Initialization flag	16-bit integer	Reserved for internal use	-
Parameter 9	d2+8 d2+9	Last calculation time	32-bit integer	View usage (not operable)	-
Parameter 10	d2+10	Last temperature	16-bit integer	View usage (not operable)	-
Parameter 11	d2+11	Reserved	16-bit integer	Reserved	

d3 parameter setting					
Parameter	**Offset address**	**Name**	**format**	**Instruction**	**Range**
Parameter 1	d3	Current Temperature	16-bit integer	Current test temperature	-
Parameter 2	d3+1	set temperature	16-bit integer	Set temperature	-
Parameter 3	d3+2	Calculation period	16-bit integer	Take an integer multiple of the pid sampling time, usually the same	-
Parameter 4	d3+3	Kp	16-bit integer	Kp value of PID after adjustment	-
Parameter 5	d3+4	KI	16-bit integer	Ki value of PID after adjustment	-
Parameter 6	d3+5	KD	16-bit integer	Kd value of PID after adjustment	-

d3 parameter setting					
Parameter 7	d3+6	Sampling cycle	16-bit integer	No need to enter	-
Parameter 8	d3+7	Reserved	16-bit integer	Reserved	-

✎Note:

The instruction starts from the device specified in (d1) and occupies 38 points of the device, and initializes the parameters. Normally, it only needs to be initialized once before calling (some parameters are fixed) (occupies 38 words space).

The instruction starts with the device specified in (d2) and occupies 12 points of the device, input parameters, and input the first 6 parameters, where Kp, Ki, Kd are the initial values of the PID control parameters (occupies 12 words space) .

The instruction starts from the device specified in (d3) and occupies 8 points of soft elements and output parameters, among which Kp, Ki, Kd are the parameter values after fuzzy adaptive calculation, which can be input to the designated position of PID (occupy 8 words space).

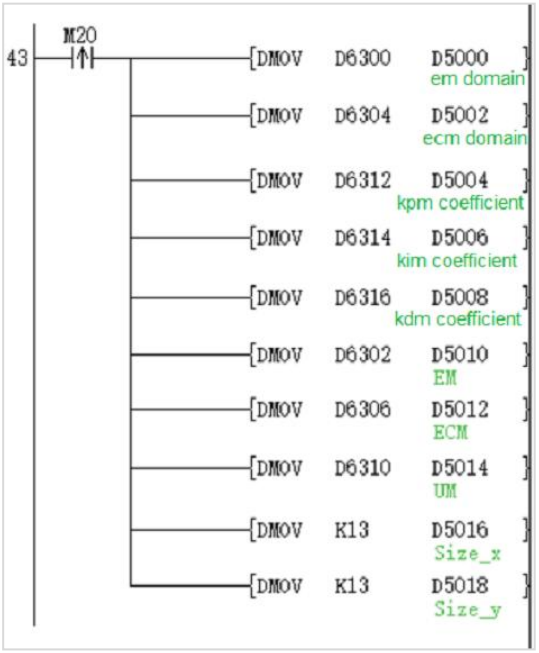
The FPID instruction occupies 58 words. The address of each operand must have a specified interval interval, which cannot be occupied by other instructions.

Error code

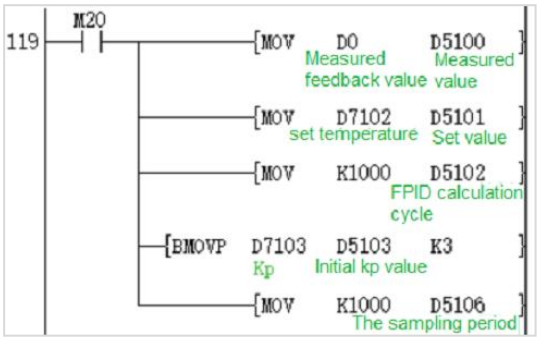
Error code	Content
4085H	When the device specified in the read application instructions (d1), (d2), (d3) exceeds the range of the corresponding device.
4086H	When the device specified in the write application instructions (d1), (d2), (d3) exceeds the range of the corresponding device.
4D91H	FPID calculation cycle is less than or equal to 0
4D92H	FPID parameter range error
4D93H	FPID initial flag error

Example

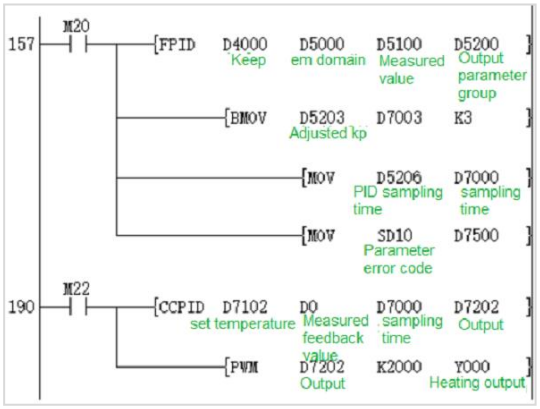
1. Parameter d1



2. Parameter d2



3. Invoke FPID



11.4 CCPID instruction introduction manual

(1) Background and purpose

Background:

PID (proportion, integral, derivative) controller has been the earliest practical controller for nearly a hundred years, and it is still the most widely used industrial controller. The PID controller is simple and easy to understand, and does not require precise system models and other prerequisites in use, making it the most widely used controller.

Purpose:

You might not be familiar with the parameter settings in the new series of CCPID for the first time, this manual could let you quickly understand the meaning of each parameter in the CCPID and the influence on the control effect, so that you can quickly learn the CCPID.

(2) Description of the host CCPID instruction

Instruction description

Content, range and data type

Name	Features	Bits (bits)	Whether pulse type	Instruction format	Step count
CCPID	PID Operation	16	No	CCPID (S1) (S2) (S3) (D)	9

Device used

Instruction	Parameter	Devices																								Offset modification	Pulse extension
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CCPID	Parameter 1																	●	●	●							
	Parameter 2																	●	●	●							
	Parameter 3																	●	●	●							
	Parameter 4																	●	●	●							

(S1) is the target value (SV) of PID control;

(S2) is the measured feedback value (PV);

(S3) is the start address of the buffer area for setting parameters required for PID operation and saving intermediate results, occupying a total of 52 variable units of subsequent addresses (recommended to reserve 100 continuous spaces).The value range is D0 to D7,948, it is better to specify power failure retention, and the setting value remains after power supply is off. Otherwise,the buffer needs to be assigned value before starting the calculation for the first time. The function and parameter description of each unit are described in this section;

(D) is the storage unit (MV) of the PID calculation result. Please specify it as a non-battery retentive area, otherwise it needs to be initialized and cleared before the first start of calculation.



(3) Programming example

The parameter description is as follows:

In D9, the target value of PID adjustment is stored, and D10 is the closed-loop feedback value. Note that D9 and D10 must be of the same dimension, such as both 0.01MPa units, or 1℃ units, etc.;

A total of 52 units of D200 to D224 are used to store the set value and process value of PID operation. These values must be set item by item before the first PID calculation;

D130 unit is used to store the calculated control output value to control the execution of the action.

The functions and setting methods of the parameter values of each unit used by (S3) are described in the following table:

(S3) to (S3)+14 is the parameter range that can be set (parameters set when CCPID is executed).

(S3)+15 to (S3)+21 is the space used internally by CCPID control.

(S3)+22 to (S3)+51 is the parameter space used in the auto-tuning process.

Unit	Features	Setting instructions	Supplement
S3	Sample time (TS)	The set range is 1 to 32,767 (ms), but greater than PLC program scan cycle.	It is how often the instruction calculates and updates the output value (MV). When TS is less than one scan time, PID instruction is executed with one scan time and alarm 4D86H. When TS ≤ 0, alarm 4D80H and no execution.
S3+1	Action direction (ACT)	bit0: 0=positive action; 1=reverse action bit2: auto-tuning transition zone switch. 0=not open;1=open bit3: 0=unidirection; 1=bidirection Bit4: 0=auto-tuning does not execute; 1=execute auto-tuning [Bit6:0=Two-stage auto-tuning does not execute. 1=Execute two-stage auto-tuning (bit4 must be set to 1). bit7: 0=Three-stage auto-tuning does not execute. 1=Execute three-stage auto-tuning (bit4 must be set to 1)] The Others cannot be used.	bit0: Positive action: similar heating system, when the temperature is lower than the set value, increases the output ; Reverse action: similar cooling system, when the temperature is greater than the set value, increases the output. bit2: Self-tuning transition zone switch. There is a transition zone size of 1.5℃ when opened. bit3: Bidirection indicates that outputs the positive and negative values to the heating system or the cooling system to control two external systems by one PID. bit4: ✎When bit4=1 and bit6 and bit7 are not 1, auto-tuning is not executed. ✎When bit4=0 and one of bit6 and bit7 is 1, auto-tuning is not executed. ✎When bit4=1 and bit6 and bit7 are both 1, auto-tuning is executed
S3+2	Filter coefficient	The first-order inertia filter of feedback amount (0 to 100%) has a range of 0 to100	When the value is greater than or equal to 100, it will be executed as 0, that is, no filtering will be executed;
S3+3	Proportional gain(Kp)	Set range: 0 to 30,000[%]	Overrun error 4D87H
S3+4	Integration time (Ti)	Ti is integration time, and the range is 0 to 3,600 (s)	Overrun error 4D88H
S3+5	Differential time (Td)	Td is derivative time, and the range is 0 to 1,000 (s)	Overrun error 4D89H
S3+6	Working interval	Operating temperature setting enabled by PID (0 indicates no effect) The range is 0 to 1,000	It is recommended to be greater than 5°C, that is, 50 (precision 0.1°C). If it exceeds the range, the boundary value will be taken.
S3+7	Output low limit	Range: -10,000 to 10,000. Recommended setting range: -2,000 or 0 (when S3+1 bit3=0, the lower limit = 0; when bit3=1, the lower limit = -2,000)	1. Self-tuning initialization: ① Unidirection control: the lower limit is 0; ② Bidirection control: If the lower limit is greater than 0, adjust the lower limit to 0; if the upper limit and the lower limit are equal to 0, the default lower limit is -2,000. ✎Note: If set to -2,000, and the output value (MV) is less than -2,000, it will output -2,000. 2. During the control process, the lower limit is dynamically adjustable. If the lower limit is greater than or equal to the upper limit, error 4D90H will be reported.

Unit	Features	Setting instructions	Supplement
S3+8	Output upper limit	Value range: -10,000 to 10,000. Recommended setting value is 2,000	1. Self-tuning initialization: ① Unidirection control: If the upper limit is less than 0, the default upper limit is 2,000; ② Bidirection control: If the upper limit is less than 0, adjust the upper limit to 0; if the upper limit and the lower limit are equal to 0, the default upper limit is -2,000. ✎Note: If set to -2,000 and the output value (MV) is greater than -2,000, it will output 2,000. 2. During the control process, the upper limit is dynamically adjustable. If the lower limit is greater than or equal to the upper limit, error 4D90H will be reported.
S3+9	Mode setting	0: Overshoot allowed 1: Slight overshoot or no overshoot 2: Dynamic setting	0:Overshoot allowed (ukd = 100) 1: Slight overshoot or no overshoot mode (ukd = 300)
S3+10	Scale factor (ukp)	Typically sets value to 100 (default 100) [enabled when S3+9 is set to 2].The range is 1 to 500.	When the value is less than or equal to 0, or greater than 500, the boundary value will be taken.
S3+11	Integral coefficient (uki)	Typically sets value to 50 (default 50) [enabled when S3+9 is set to 2]. The range is 1 to 300.	When the value is less than or equal to 0, or greater than 300, the boundary value will be taken.
S3+12	Differential coefficient (ukd)	Typically sets value to 50 (default 100. 300 to 400 can be set when slight overshoot is required) [Enable when S3+9 is set to 2]. The range is 1 to 500.	When the value is less than or equal to 0, or greater than 500, the boundary value will be taken.
S3+13	Maximum ascent rate (DeltaT)	The range is 0 to 32,000, which is the threshold of integral increment	Overrun error 4D82H
S3+14	Filtering (C0)	The range is 0 to 1,023, integral part filtering	Overrun error 4D81H
S3+15 ⋮ S3+21	reserved for internal control	Internal control space occupation	
S3+22 ⋮ S3+51	used space for self-tuning	New self-tuning space for internal use	

The auto-tuning process occupies the space of S3+22 to S3+51. When the auto-tuning is successful, the adjusted parameters will be written into the space of S3+2 to S3+21.

S3+2 filter coefficient α: Processing in first-order inertial filter

Formula: $\mathbf{T_{now}=(100-\alpha)\times T_{\alpha}+\alpha\times T_{old}}$

T_{α} is the currently measured temperature. T_{old} is the temperature that participated in the PID calculation last time. T_{now} is the temperature used for the current PID calculation. α is the filter coefficient (when α=0, no filtering is performed, and the range of α is 0 to 100.

(If there is a temperature with a small overshoot but a long stabilization time, the parameter can be set to 80, and analyze the specific problems in detail)

S3+6 work range: T_{work}(example: 170 represents 17℃)

Positive action: OUT= { 100% power output: PV<SV-T_{work} ; PidOut: PV≥SV-T_{work}

Reverse action: OUT= { 100% power output: PV>SV+T_{work} ; PidOut: PV≤SV+T_{work}

S3+9 working mode:

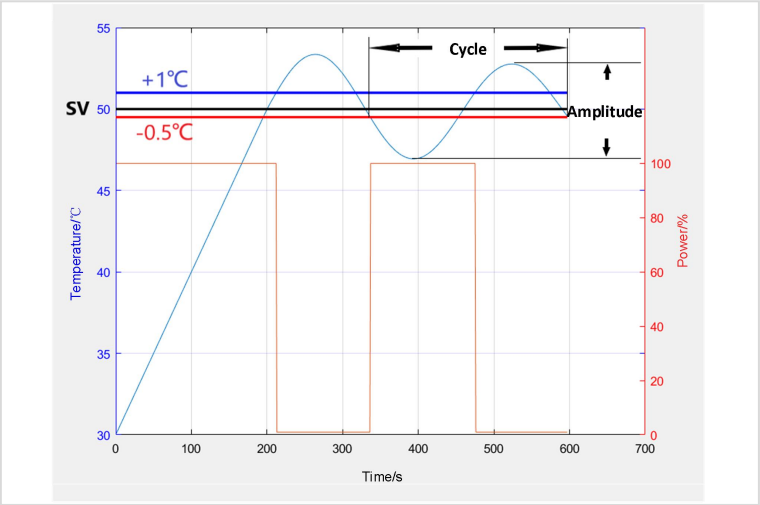
0: Working mode that allows overshoot

1: Slight overshoot or no overshoot working mode

2: Custom settings; to achieve by setting S3+10, S3+11, S3+12 three coefficients.

S3+1 bit2 self-tuning transition zone switch: (upper limit 1℃, low limit 0.5℃)

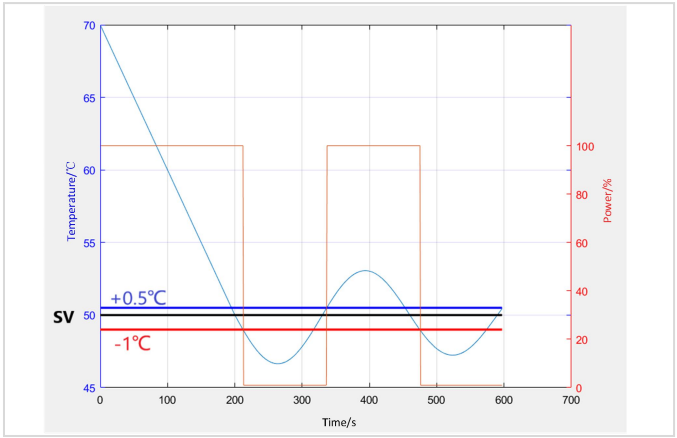
① The transition zone description in forward control:



In the heating process, when PV≤SV+1℃, 100% power output; when PV＞SV+1℃, no output.

In the cooling process, when PV＜SV-0.5℃, 100% power output; When PV≥SV-0.5℃, no output.

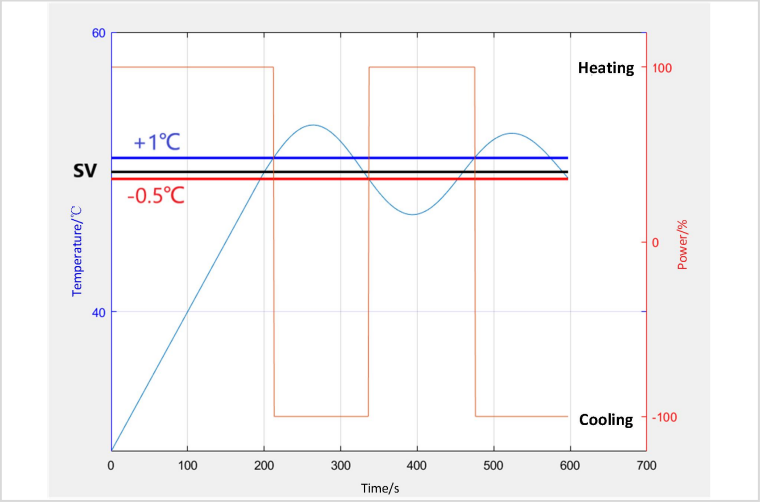
② The transition zone description in reverse control:



In the cooling process, when PV≥SV-1℃, 100% power output; when PV＜SV-1℃, no output.

In the heating process, when PV＞SV+0.5℃, 100% power output; When PV≤SV+0.5℃, no output.

③ The transition zone description in bidirectional control:



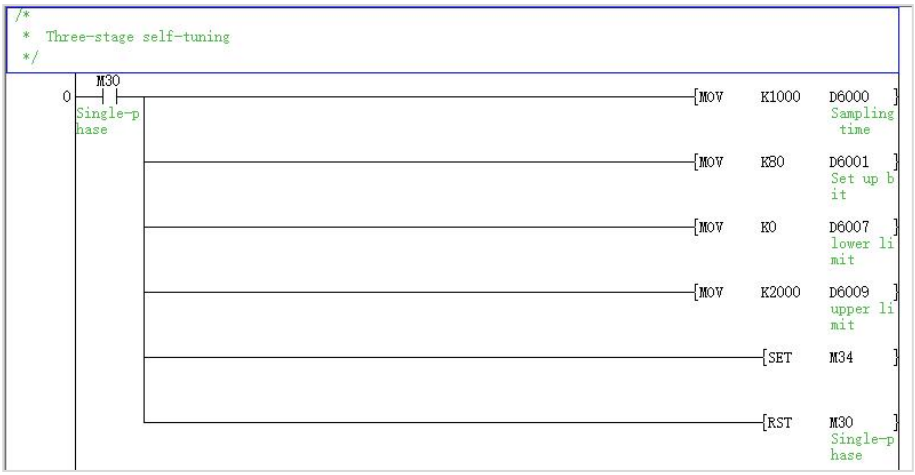
In the heating process, when PV≤SV+1℃, 100% power heating output; when PV>SV+1℃, 100% power cooling output.

In the cooling process, when PV＜SV-0.5℃, 100% power heating output. When PV≥SV-0.5℃, 100% power cooling output

(4) Programming case

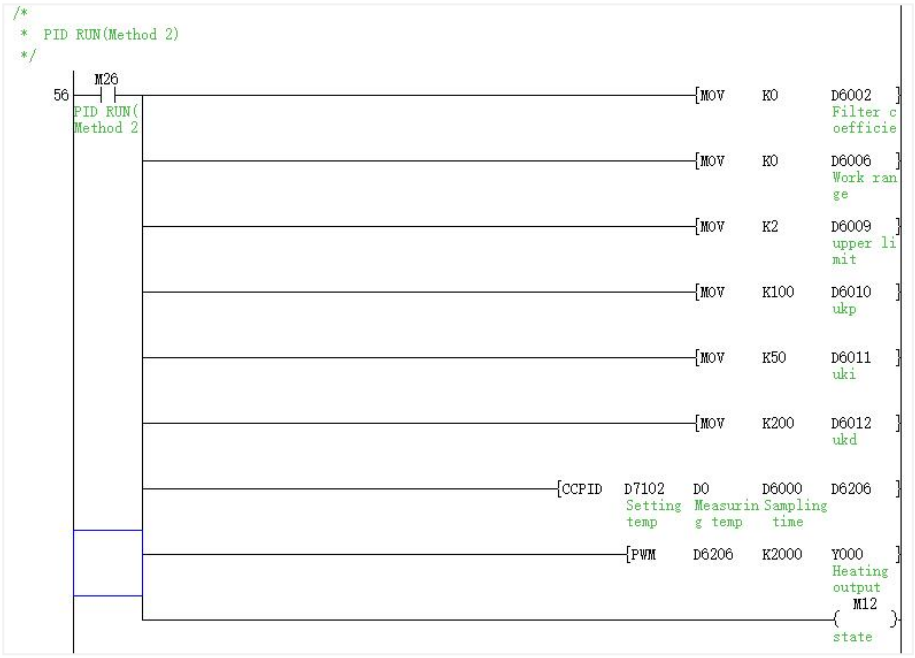
CCPID application configuration

① Parameter setting

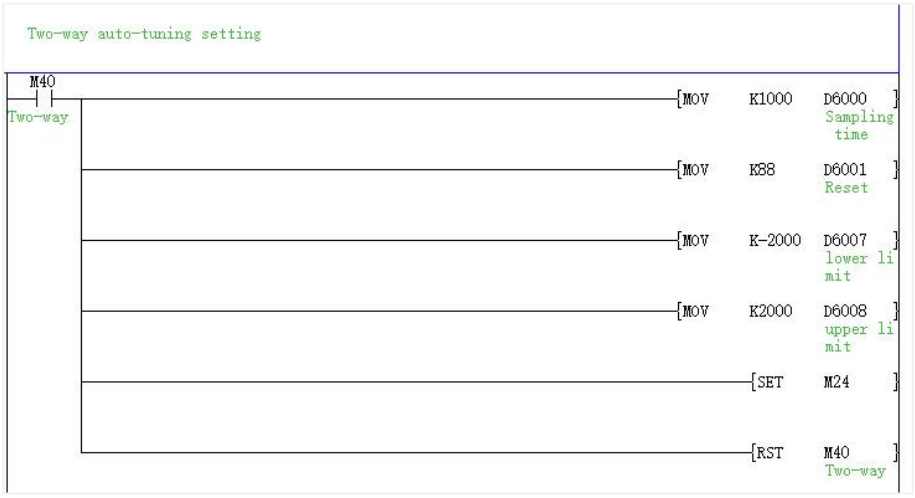


② CCPID control process setting





③ Bidirection control



✎Note:

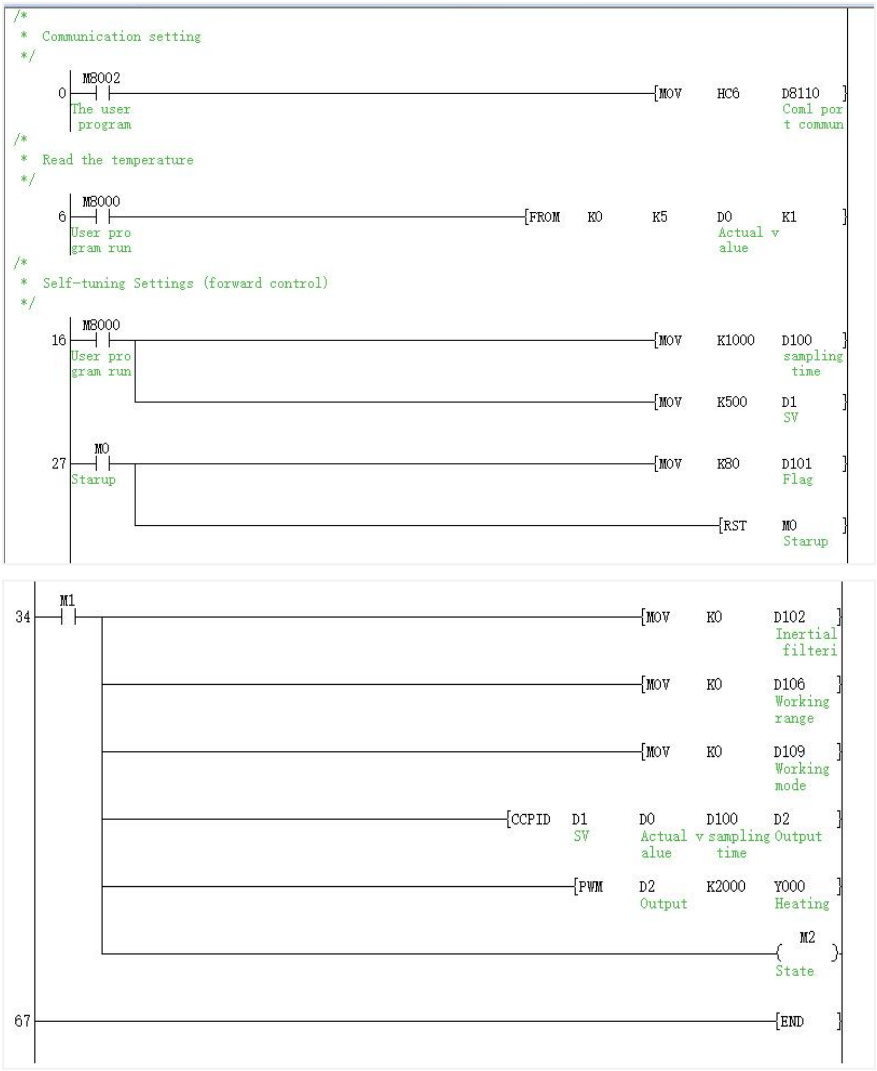
1. CCPID is a special instruction for operation control. CCPID operation will be executed only after the sample time is reached.
2. There is no limit to the number of times the CCPID instruction can be used, but+51 cannot be repeated.
3. Before CCPID instruction is executed, CCPID parameters need to be set.

(5) Case analysis

Control requirements

The control environment of this example is a kettle. The configuration is controlled by PLC-5V2416 host with 4PT module, and PI8070 screen is used for data storage and process curve viewing.

① **Sample program**



② **Parameter description**

PLC device	Control instructions
M0	Set auto tuning
M1	CCPID instruction calculation start
M2	CCPID operating status
Y0	Pulse output with adjustable pulse width
D0	Temperature measured value
D1	Temperature setting value
D100	sample time
D101	Control detail settings
D102	First-order inertial filter coefficient
D106	Working interval
D109	Working mode

(6) Parameter control effect description

① **Boiling water experiment**

a. Auto-tuning process and control process (no transition zone setting), take two-stage auto-tuning as an example

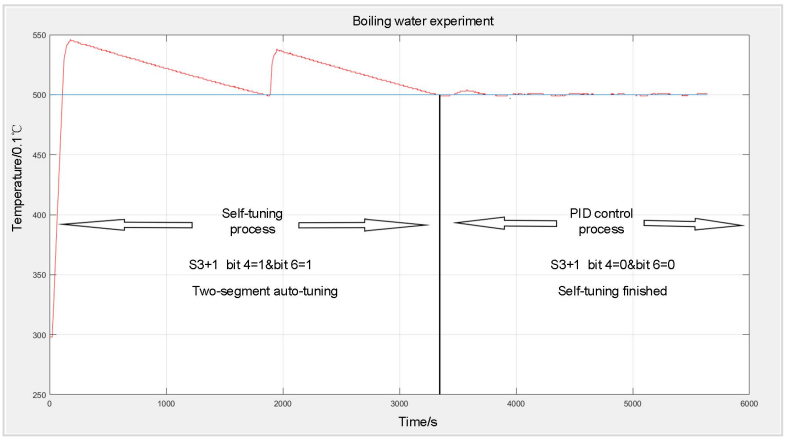


Figure 1 Auto-tuning process curve without transition zone

When the control system is a single temperature control system or a system where environmental interference does not cause large fluctuations. Generally the automatic tuning without transition zone is selected, so that the self-tuning process can be completed more quickly than the method with transition zone.

b. Self-tuning process and control process (transition zone setting)

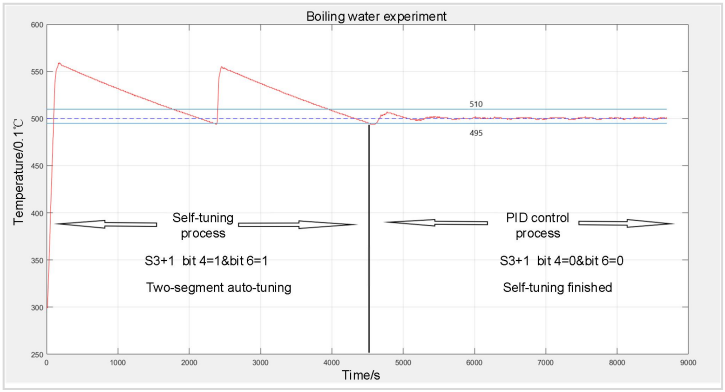


Figure 2 Self-tuning process curve with transition zone

It is more suitable in a two-way control system with transition zone self-tuning process. The transition zone has a range of 1.5°C. The upper limit is 1°C, and the lower limit is 0.5°C.

② **Difference in working interval setting**

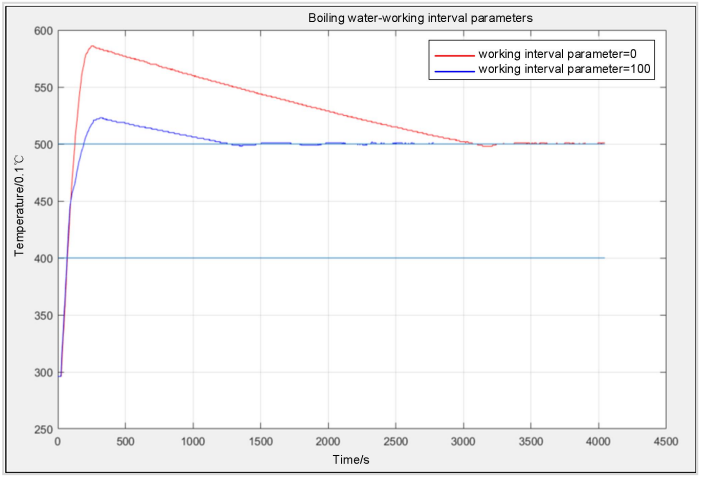


Figure 3 Process curve under different working interval parameters

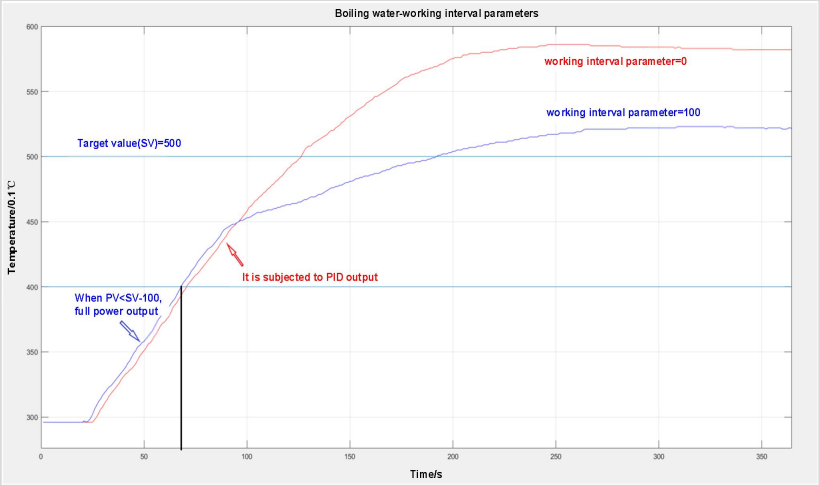


Figure 4 Process curve without different working interval parameters (heating process diagram)

It can be seen from the partially enlarged graph that the parameters of the working interval have a certain influence on the overshoot and the stable time. In the case of allowing overshoot, setting the working interval parameters can make the overshoot smaller. This is because the deviation E of PID starting to work is relatively small, and the integration accumulation will not quickly saturate.

③ **Result of filter coefficient setting**

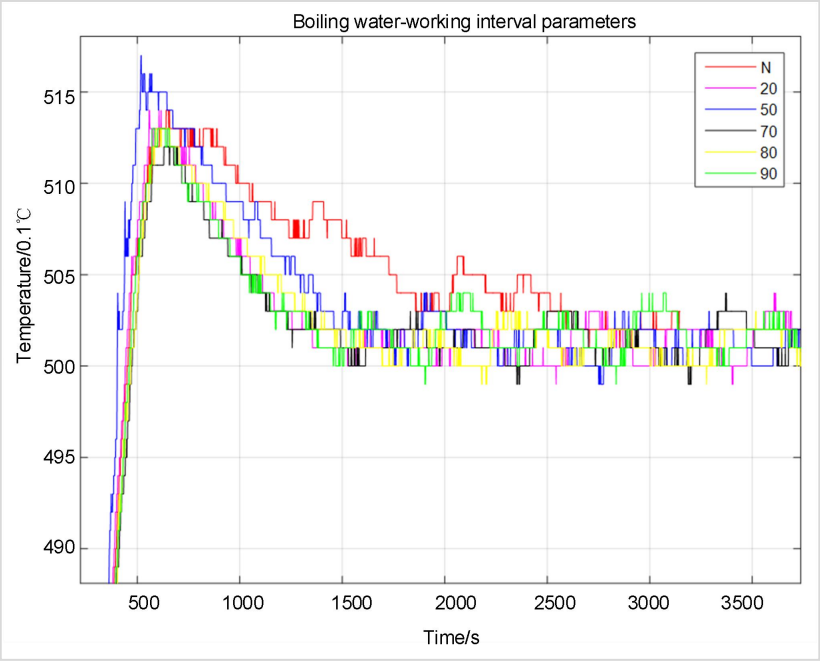


Figure 5 Process curve under different filtering parameters

The figure above is the experimental result under the small overshoot coefficient, the sample time is 1s. The coefficients of the first-order inertial filtering are (20, 50, 70, 80, 90). After adding the inertia coefficient, the stability time of system control is greatly accelerated, and it is increased by about 6 minutes for the boiling water experiment. The overshoot is about 1.2°C to 1.7°C.

Therefore, the introduction of first-order inertial filtering could greatly improve the PID environment where the temperature fluctuates to a certain extent and increase stabilization time.

✎**Note:** This parameter of filter coefficient is helpful for systems with not very large hysteresis or the control effect of the phenomenon that the control amount fluctuates back and forth has been greatly improved.

④ **The difference in mode selection**

0: Overshoot allowed(ukd = 100)

1: Small overshoot or no overshoot (ukd = 300)

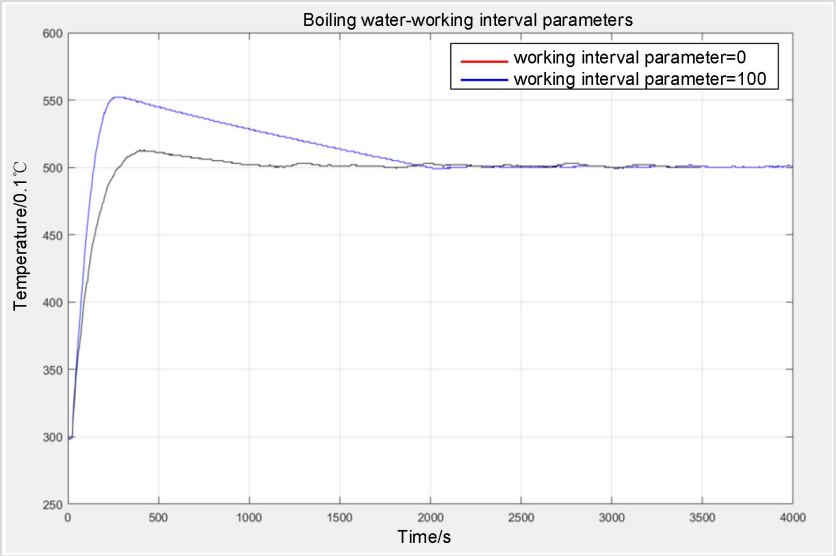


Figure 6 Process curves in different working modes

When selecting mode 1 (small overshoot or no overshoot), the stable temperature may be slightly higher than the set temperature (fluctuates above the set temperature).

⑤ The function of the coefficient

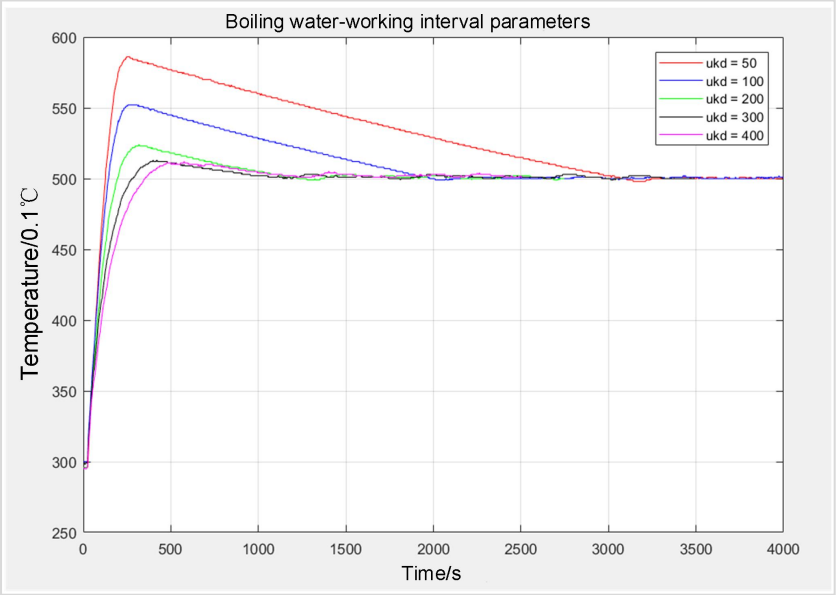


Figure 7 Process curve under dynamic setting

When selecting working mode 2, there are three corresponding adjustable parameters: ukp[S3+10], uki[S3+11], ukd[S3+12]. Usually, the default parameters can be used for ukp and uki. Adjust the value of ukd could achieve the control effect.

Ukp is adjusted when the value of Kp reaches the maximum value, and the default value is usually 100.

Uki is adjusted when periodic oscillations occur. Gradually increase the value of uki to track the control effec

11.5 TRH/Wet and dry bulb temperature and humidity conversion

TRH

This instruction completes the conversion of dry bulb temperature, wet bulb temperature and corresponding humidity.

-[TRH (d1) (s) (d2) (n)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(d1)	Humidity	0~100	Single-precision floating point	ANYREAL_32	REAL
(s)	Dry bulb temperature	-	Single-precision floating point	ANYREAL_32	REAL
(d2)	Wet bulb temperature	-	Single-precision floating point	ANYREAL_32	REAL
(n)	Mode	0 to 1	Signed BIN 32 bit	ANY32	DINT

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TRH	Parameter 1															●	●	●	●	●						●	
	Parameter 2															●	●	●	●	●						●	
	Parameter 3															●	●	●	●	●						●	
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	

Function

There are two modes to choose from (n):

Mode 0: Calculate the corresponding humidity by wet bulb temperature and dry bulb temperature.

Mode 1: Calculate the corresponding wet bulb temperature by dry bulb temperature and humidity.

The conversion process formula is as follows:

Assuming that the wet bulb temperature is A, the dry bulb temperature is B, and the corresponding current humidity is C, which meet the following conditions:

$$EXP\{(A\times 17.27)/(A+237.36)\}\times 611 = x \qquad (1)$$

$$EXP\{(B\times 17.27)/(B+237.36)\}\times 611 = y \qquad (2)$$

$$z = x - C\times y/100 \qquad (3)$$

$$A = B - z/65.566 \qquad (4)$$

Precautions

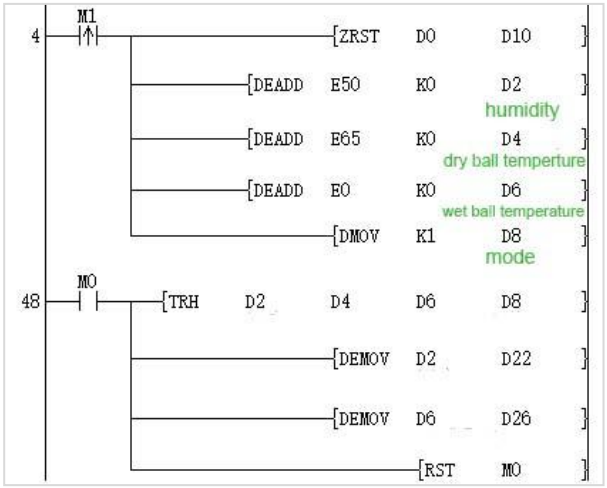
·The wet bulb temperature is not greater than the dry bulb temperature. When they are the same, the humidity reaches the maximum value 100%.

·The unit of dry and wet bulb temperature is (°C).

·The general value range of dry bulb is between 0~100°C, and the instruction does not judge its range, so pay special attention when using this instruction.

Error code

Error code	Content
4084H	When the value specified in (n) exceeds the following range. 0 to 1
	The value specified in (d1) exceeds the following range. 0 to 100
	A negative value is specified in (s).
	A negative value is specified in (d2).
4085H	The output result of (d1)(s)(d2)(n) in the read application instruction exceeds the device range.
4086H	The output result of the write application instruction (d1) and (d2) exceeds the device range.



Dry and wet bulb temperature/humidity conversion table

Dry/wet ball temperature and humidity conversion table

B / C / A		25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
10.0	2.55	3.06	3.58	4.09	4.58	5.07	5.54	6.02	6.49	6.95	7.41	7.86	8.29	8.73	9.16	9.59	10.00
11.0	3.24	3.78	4.32	4.85	5.37	5.88	6.38	6.87	7.36	7.84	8.31	8.77	9.24	9.69	10.13	10.57	11.00
12.0	3.94	4.50	5.06	5.62	6.15	6.68	7.21	7.72	8.23	8.72	9.21	9.70	10.17	10.64	11.10	11.56	12.00
13.0	4.62	5.21	5.79	6.38	6.93	7.49	8.04	8.57	9.09	9.61	10.12	10.62	11.12	11.59	12.07	12.54	13.00
14.0	5.30	5.92	6.53	7.13	7.72	8.29	8.85	9.42	9.96	10.50	11.02	11.54	12.05	12.55	13.05	13.52	14.00
15.0	5.98	6.62	7.26	7.89	8.50	9.10	9.68	10.26	10.83	11.38	11.93	12.47	12.99	13.50	14.02	14.51	15.00
16.0	6.64	7.32	7.99	8.64	9.28	9.90	10.51	11.11	11.69	12.27	12.83	13.38	13.93	14.47	14.98	15.50	16.00
17.0	7.31	8.02	8.72	9.39	10.05	10.70	11.34	11.95	12.56	13.16	13.73	14.31	14.87	15.42	15.95	16.48	17.00
18.0	7.98	8.72	9.43	10.13	10.82	11.50	12.15	12.80	13.42	14.03	14.64	15.23	15.80	16.37	16.93	17.46	18.00
19.0	8.64	9.40	10.15	10.89	11.59	12.29	12.97	13.64	14.28	14.92	15.54	16.15	16.75	17.33	17.90	18.45	19.00
20.0	9.30	10.09	10.87	11.63	12.37	13.09	13.79	14.49	15.16	15.81	16.45	17.07	17.69	18.28	18.87	19.44	20.00
21.0	9.95	10.78	11.59	12.38	13.14	13.89	14.61	15.33	16.02	16.69	17.35	17.99	18.62	19.24	19.84	20.43	21.00
22.0	10.60	11.47	12.31	13.12	13.92	14.69	15.44	16.17	16.88	17.58	18.26	18.92	19.56	20.19	20.81	21.41	22.00
23.0	11.25	12.14	13.02	13.86	14.68	15.48	16.26	17.02	17.75	18.46	19.16	19.84	20.50	21.15	21.77	22.40	23.00
24.0	11.89	12.83	13.73	14.61	15.46	16.28	17.08	17.86	18.61	19.35	20.06	20.76	21.44	22.11	22.75	23.39	24.00
25.0	12.53	13.51	14.44	15.35	16.22	17.08	17.90	18.70	19.48	20.24	20.97	21.68	22.38	23.06	23.73	24.37	25.00
26.0	13.18	14.18	15.15	16.09	16.99	17.87	18.73	19.54	20.34	21.13	21.88	22.62	23.33	24.02	24.70	25.36	26.00
27.0	13.82	14.86	15.83	16.84	17.76	18.67	19.55	20.39	21.21	22.01	22.79	23.53	24.26	24.98	25.67	26.35	27.00
28.0	14.46	15.53	16.57	17.57	18.54	19.46	20.37	21.24	22.08	22.90	23.70	24.46	25.20	25.94	26.64	27.33	28.00
29.0	15.10	16.21	17.28	18.31	19.31	20.26	21.20	22.09	22.95	23.79	24.61	25.39	26.15	26.90	27.61	28.32	29.00
30.0	15.73	16.88	17.99	19.05	20.08	21.07	22.02	22.94	23.82	24.68	25.51	26.31	27.10	27.85	28.58	29.30	30.00
31.0	16.37	17.56	18.70	19.80	20.85	21.87	22.84	23.78	24.69	25.57	26.42	27.24	28.04	28.82	29.56	30.29	31.00
32.0	17.00	18.22	19.41	20.54	21.62	22.67	23.67	24.63	25.56	26.47	27.33	28.17	28.99	29.76	30.54	31.27	32.00
33.0	17.63	18.90	20.12	21.28	22.40	23.47	24.50	25.48	26.43	27.35	28.24	29.10	29.93	30.73	31.51	32.27	33.00
34.0	18.26	19.58	20.83	22.02	23.18	24.28	25.32	26.33	27.31	28.25	29.15	30.03	30.87	31.69	32.49	33.25	34.00

12 Character string instruction

12.1 LEN/Length detection of string

LEN(P)

Detect the length of the string specified in (S), and store it after the device number specified in (d).

Data from the device number specified in (s) to the device number storing 00H is treated as a character string.

-[LEN (s) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	String or start number of device that stores string	-	String	ANYSTRING_SINGLE	--
(d)	Device number that stores the detected string length	-	Signed BIN16 bits	ANY16	INT

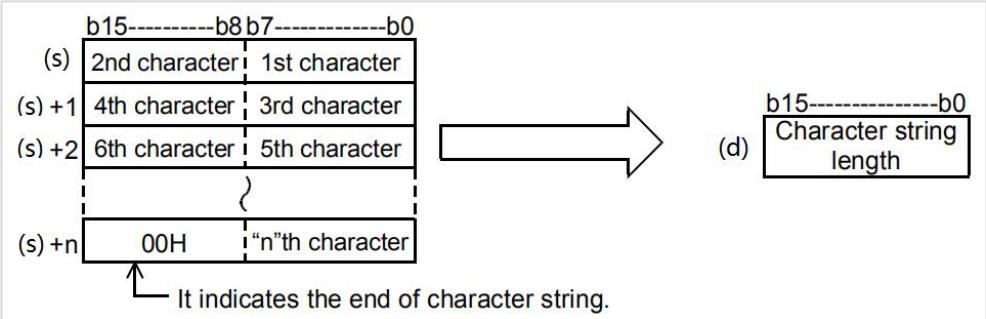
Device used

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LEN	Parameter 1											•	•	•	•	•	•	•	•	•						•	•
	Parameter 2												•	•	•	•	•	•	•	•						•	•

Function

Detect the length of the string specified in (S), and store it after the device number specified in (d).

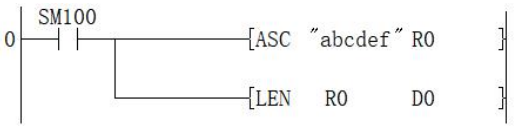
Data from the device number specified in (s) to the device number storing 00H is treated as a character string.



Error code

Error code	Content
4085H	(s) read address exceeds the device range.
408AH	The length of the read string of (s) exceeds the limit, and the continuous length of the string exceeds 400 characters.
408BH	When (s) reading a character string, the maximum range of the device is read, but 00H is not found as the end.
4086H	When using offset, the offset address of (d) exceeds the device range.

Example



As shown in the above ladder program:

Use the asc instruction to write the string abcdef to the address starting from R0.

Then use LEN instruction to determine the length. At this point, D0 will display 6.

12.2 LEFT/Extracting from the left side of the string

LEFT(P)

For the character string data stored after device number specified in (s), (n) Character data starting from the left side of the string (the start of the string) is stored after device number specified in (d).

-[LEFT (s) (d) (n)]

Content, range and data type

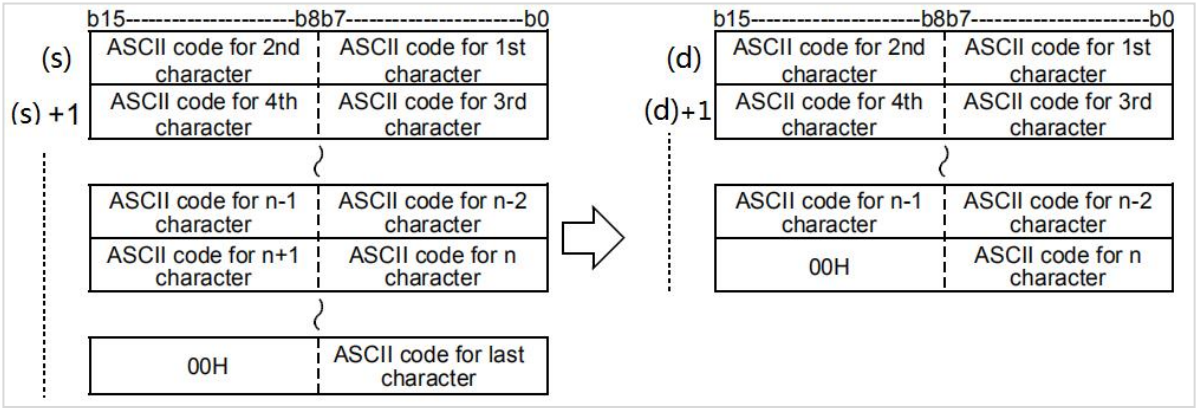
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	String or start number of device that stores string	-	String	ANYSTRING_SINGLE	--
(d)	Store the start number of the device of the string of (n) character starting fro m the left side of (s)	-	String	ANYSTRING_SINGLE	--
(n)	Number of characters extracted	1 to 400	Signed BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																					Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LEFT	Parameter 1											•	•	•	•	•	•	•	•	•						•	•
	Parameter 2												•	•	•	•	•	•	•	•						•	•
	Parameter 3												•	•	•	•	•	•	•	•			•	•		•	•

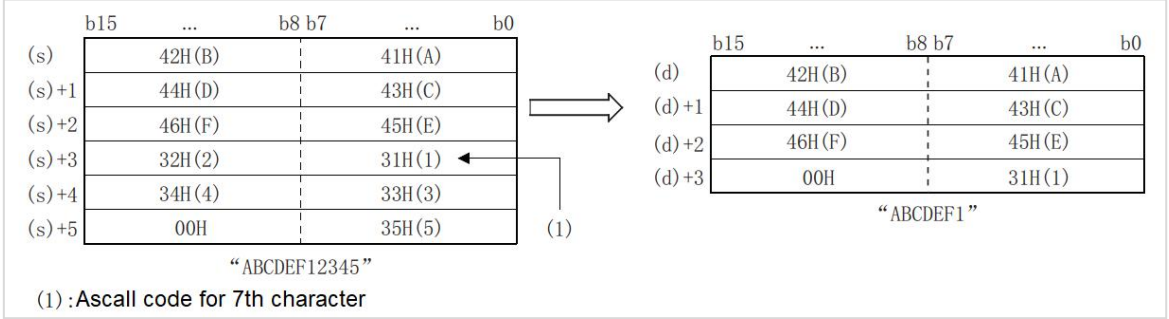
Function

For the character string data stored after device number specified in (s), (n) character data starting from the left side of the string (the start of the string) is stored after device number specified in (d).



The string specified in (s) refers to the data in bytes from the specified device to the position where "00H" is first detected.

(N)=7:



The final NULL code (00H) representing the string is automatically appended to the end of the string data.

If the number of extracted characters is odd, “00H” will be stored in the high byte of the device storing the final character. If the number of extracted characters is even, “0000H” will be stored in the low byte of the device storing the final character.

If the number of characters specified in (n) is 0, a NULL code (00H) is stored in (d).

Precautions

Attention shall be paid to the following points when handling the character codes other than ASCII code.

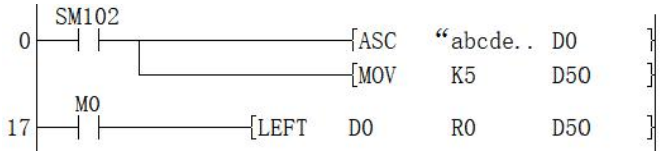
The number of characters is processed in bytes (8 bits). Therefore, like the shifted JIS code, the character code of 1 character is represented by 2 bytes and the number of characters of 1 character is “2”.

When extracting the string from the string for which the character code of 1 character is represented by 2 bytes like the shifted JIS code, the number of the characters extracted shall be determined by taking the character code of 1 character as the unit. Please note that if only 1 byte of a 2-byte character code is extracted, it will not be the desired character code.

Error code

Error code	Content
4085H	(s) read address exceeds the device range.
408AH	The length of the read string of (s) exceeds the limit, and the continuous length of the string exceeds 400 characters.
408BH	When (s) reading a character string, the maximum range of the device is read, but 00H is not found as the end.
4084H	(n)<1 or (n)> string length
4086H	(d) write address exceeds the device range.

Example



Take five characters from the left of “abcdef” starting from D0 into R0, and the string of R0 is “abcde”.

R0	1	0	0	0	0	1	1	0	0	1	0	0	0	1	1	0	ab
R1	1	1	0	0	0	1	1	0	0	0	1	0	0	1	1	0	cd
R2	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	e.
R3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

12.3 RIGHT/Extracting from the right side of the string

RIGHT(P)

For the character string data stored after device number specified in (s), (n) character data starting from the right side of the string (the end of the string) is stored after device number specified in (d).

-[RIGHT (s) (d) (n)]

Content, range and data type

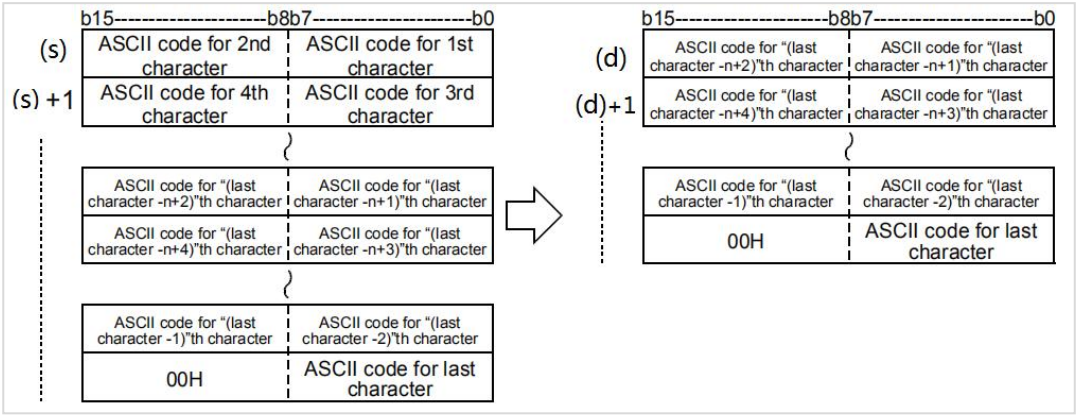
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	String or start number of device that stores string	-	String	ANYSTRING_SINGLE	--
(d)	Store the start number of the device of the string of (n) character starting from the right side of (s)	-	String	ANYSTRING_SINGLE	--
(n)	Number of characters extracted	1 to 400	Signed BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RIGHT	Parameter 1											•	•	•	•	•	•	•	•	•						•	•
	Parameter 2												•	•	•	•	•	•	•	•						•	•
	Parameter 3												•	•	•	•	•	•	•	•			•	•		•	•

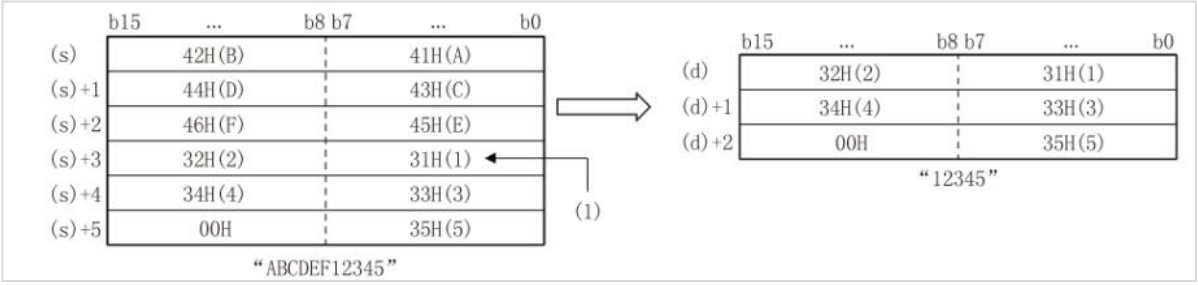
Function

For the character string data stored after device number specified in (s), (n) character data starting from the right side of the string (the end of the string) is stored after device number specified in (d).



The string specified in (s) refers to the data in bytes from the specified device to the position where "00H" is first detected.

(n)=5:



The final NULL code (00H) representing the string is automatically appended to the end of the string data.

If the number of extracted characters is odd, “00H” will be stored in the high byte of the device storing the final character. If the number of extracted characters is even, “0000H” will be stored in the device after the device storing the final character.

If the number of characters specified in (n) is 0, a NULL code (00H) is stored in (d).

Precautions

Attention shall be paid to the following points when handling the character codes other than ASCII code.

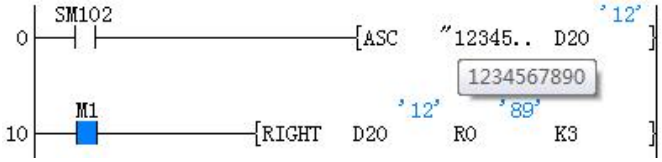
The number of characters is processed in bytes (8 bits). Therefore, like the shifted JIS code, the character code of 1 character is represented by 2 bytes and the number of characters of 1 character is “2”.

When extracting the string from the string for which the character code of 1 character is represented by 2 bytes like the shifted JIS code, the number of the characters extracted shall be determined by taking the character code of 1 character as the unit. Please note that if only 1 byte of a 2-byte character code is extracted, it will not be the desired character code.

Error code

Error code	Content
4085H	The read address of (s) and (n) exceeds the device range.
408AH	The length of the read string of (s) exceeds the limit, and the continuous length of the string exceeds 400 characters.
408BH	When (s) reading a character string, the maximum range of the device is read, but 00H is not found as the end.
4084H	(n)<1 or (n)> string length
4086H	(d) write address exceeds the device range.

Example



From the string “1234567890”, take three characters “890” from the right and store them into R0.

R0	0	0	0	1	1	1	0	0	1	0	0	1	1	1	0	0	89
R1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0.

12.4 MIDR/arbitrary extraction in the string

MIDR(P)

Store the data at any position in the character string data to the device number specified in (d).

-[MIDR (s1) (d) (s2)]

Content, range and data type

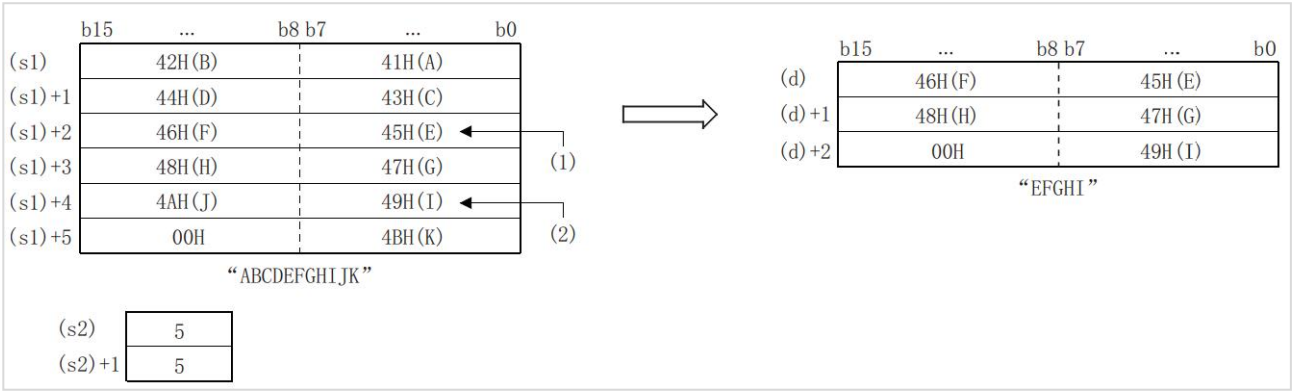
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s1)	String or start number of device that stores string	-	String	ANYSTRING_SINGLE	--
(d)	The start number of device that stores operation result string data	-	String	ANYSTRING_SINGLE	--
(s2)	The start number of device that stores start character position and number of characters (s2)+0: The position of the start character, (s2) +1: the number of characters is signed	-	Signed BIN16 bits	ANY16_ARRAY	--

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MIDR	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●						●	●

Function

For the character string data stored after device number specified in (s1), character data designated in (s2)+1 starting from the designated position of (s2) is stored after device number specified in (d).



The string specified in (s1) refers to the data in bytes from the specified device to the position where "00H" is first detected.

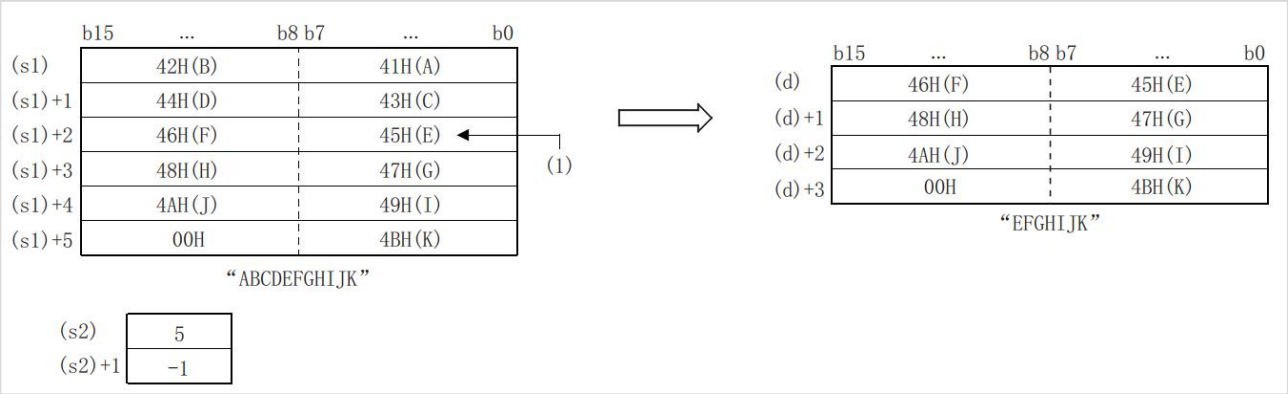
The final NULL code (00H) representing the string is automatically appended to the end of the string data.

If the number of extracted characters "(s2)+1" is odd, "00H" will be stored in the high byte of the device storing the final character.

If the number of extracted characters "(s2)+1" is even, "0000H" will be stored in the device after the device storing the final character.

When the number of characters specified in (s2)+1 is 0, no processing is performed.

If the character designated in (s2)+1 is -1, the data ending at the position of the final character data designated in (s1) is stored after device number specified in (d).



Precautions

Attention shall be paid to the following points when handling the character codes other than ASCII code:

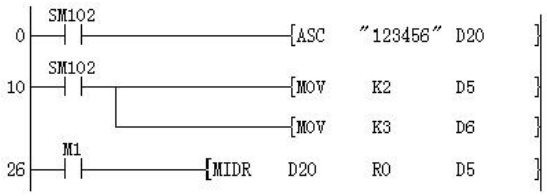
① The number of characters is processed in bytes (8 bits). Therefore, like the shifted JIS code, the character code of 1 character is represented by 2 bytes and the number of characters of 1 character is "2".

② When extracting the string from the string for which the character code of 1 character is represented by 2 bytes like the shifted JIS code, the number of the characters extracted shall be determined by taking the character code of 1 character as the unit. Please note that if only 1 byte of a 2-byte character code is extracted, it will not be the desired character code.

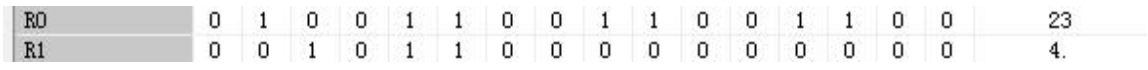
Error code

Error code	Content
4085H	(s1) and (s2) read address exceeds range of the device.
408AH	(s1) The length of the read string of (s) exceeds the limit, and the continuous length of the string exceeds 400 characters.
408BH	When (s1) reading a character string, the maximum range of the device is read, but 00H is not found as the end.
4084H	The value of (s2)+1 is less than -2 (including -2); The value of (s2) exceeds the number of characters of (s1); The value of (s2) is negative; The value of (s2)+1 exceeds the number of characters of (s1); The value obtained from the additive operation of (s2) and (s2)+1 exceeds the number of characters of (s1).
4086H	(d) Write address exceeds the device range.

Example



From the string "123456", take three characters "234" from the second character and store them into R0.



12.5 $MOV/string transfer

$MOV(P)

Transfer the string data specified in (s) to the device number specified in (d).

-[$MOV (s) (d)]

Content, range and data type

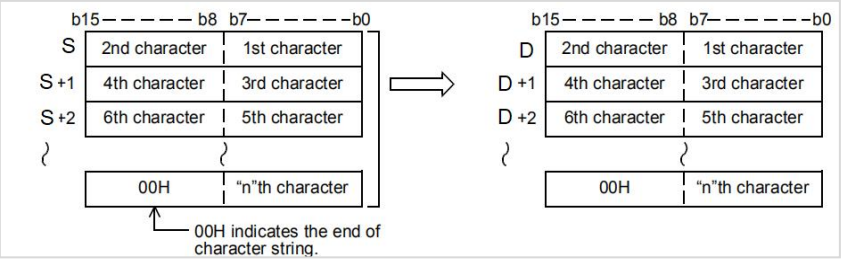
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	The start number of device that transfers string (max. 255 characters) or that stores string	-	String	ANYSTRING_SINGLE	--
(d)	The start number of device that stores and transfers the string	-	String	ANYSTRING_SINGLE	--

Device used

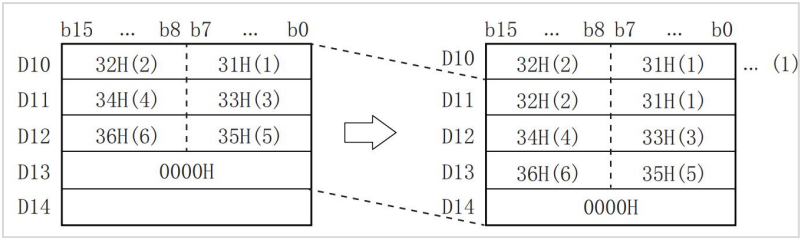
Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
$MOV	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2												●	●	●	●	●	●	●	●						●	●

Function

Transfer the string data specified in (s) to the device number specified in (d). In the transfer of a string, a string enclosed by “” (double quotation marks) specified in (s) or a string starting from the device number to the device number that stores 00H is transferred once.

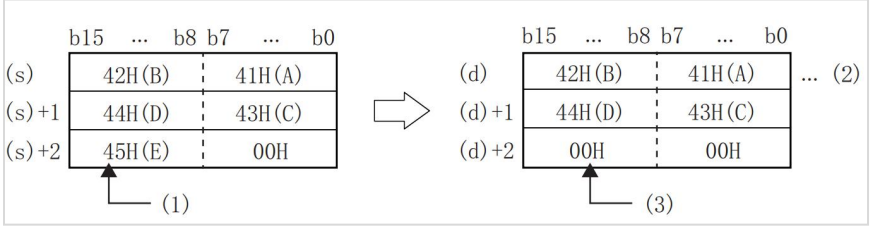


Even if the device range (s)~(s)+n that stores the transmitted string data repeates with the device range (d)~(d)+n that stores the string data after transmission , it can be processed normally. For example, when the string stored in D10~D13 is transmitted to D11~D14, the situation is shown as follows.



(1): Directly changed to the string before transmission.

If 00H is stored in the lower byte of (s) + n, 00H will be stored in both the higher byte and the lower byte of (d) + n.



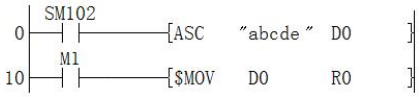
(1): High byte cannot be transmitted.

(2): Directly changed to the string before transmission.

(3): 00H is automatically stored in high bytes.

Error code	Content
4085H	(s) read address exceeds the device range.
408AH	The length of the read string of (s) exceeds the limit, and the continuous length of the string exceeds 400 characters.
408BH	When (s) reading a character string, the maximum range of the device is read, but 00H is not found as the end.
4086H	(d) Write address exceeds the device range.

Example



Copy the string “abcde” in the D0 to R0.

R0	1	0	0	0	0	1	1	0	0	1	0	0	0	1	1	0	ab
R1	1	1	0	0	0	1	1	0	0	0	1	0	0	1	1	0	cd
R2	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	e.

12.6 MIDW/arbitrary substitution in the string

MIDW(P)

For the string data stored after device number specified in (s1), store the data of the character specified in (s2)+1 to the position designated in (s2) of the string data stored after the device number designated in (d).

-[MIDW (s1) (d) (s2)]

Content, range and data type

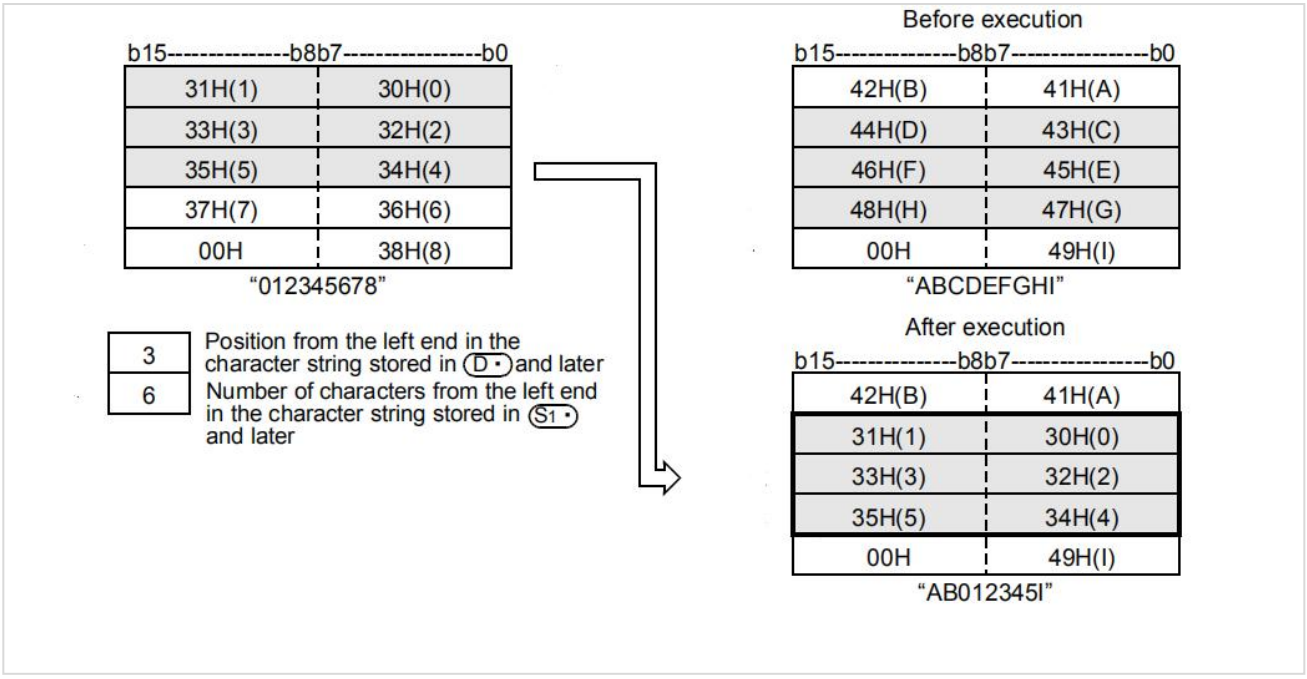
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s1)	String or start number of device that stores string	-	String	ANYSTRING_SINGLE	--
(d)	The start number of device that stores operation result string data	-	String	ANYSTRING_SINGLE	--
(s2)	The start number of device that stores start character position and number of characters (s2)+0: The position of the start character, (s2) +1: The number of characters is signed	-	Signed BIN16 bits	ANY16_ARRAY	--

Device used

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MIDW	Parameter 1											•	•	•	•	•	•	•	•	•						•	•
	Parameter 2												•	•	•	•	•	•	•	•						•	•
	Parameter 3											•	•	•	•	•	•	•	•	•						•	•

Function

For the string data stored after device number specified in (s1), store the data of the character specified in (s2)+1 to the position designated in (s2) of the string data stored after the device number designated in (d).

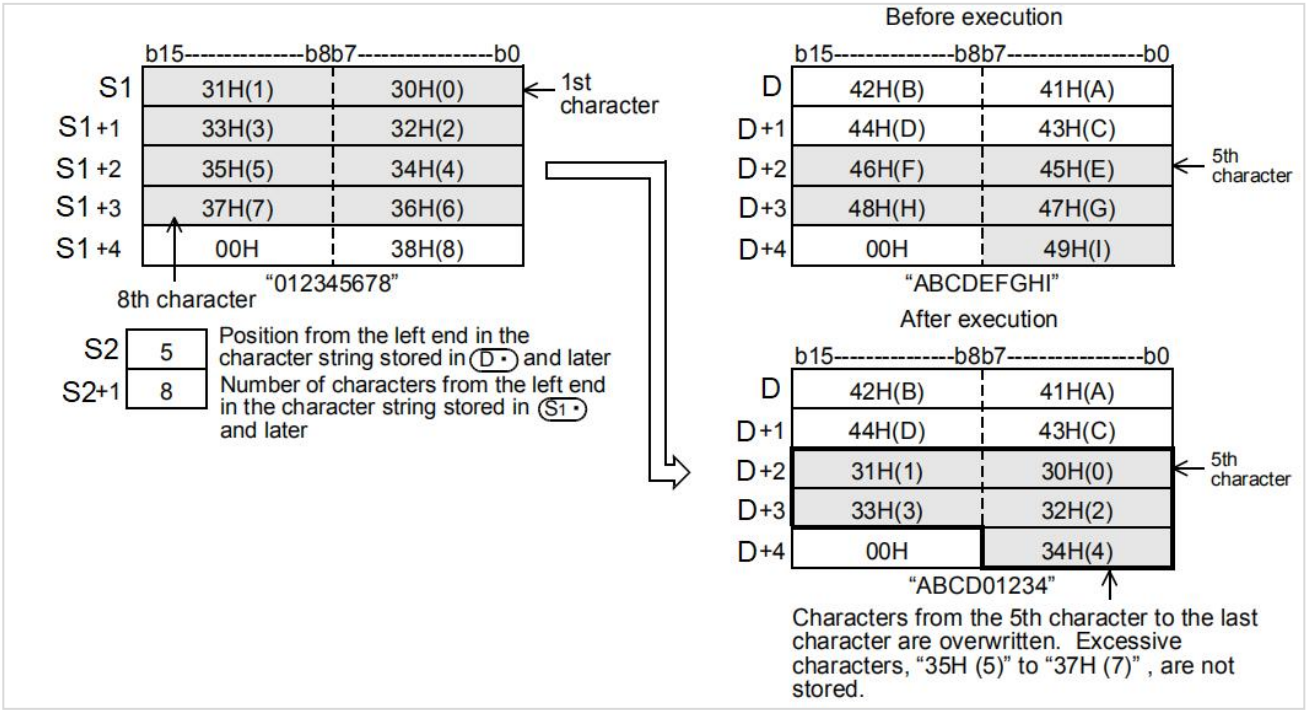


The string specified in (s1) or (d) refers to the data in bytes from the specified device to the position where "00H" is first detected.

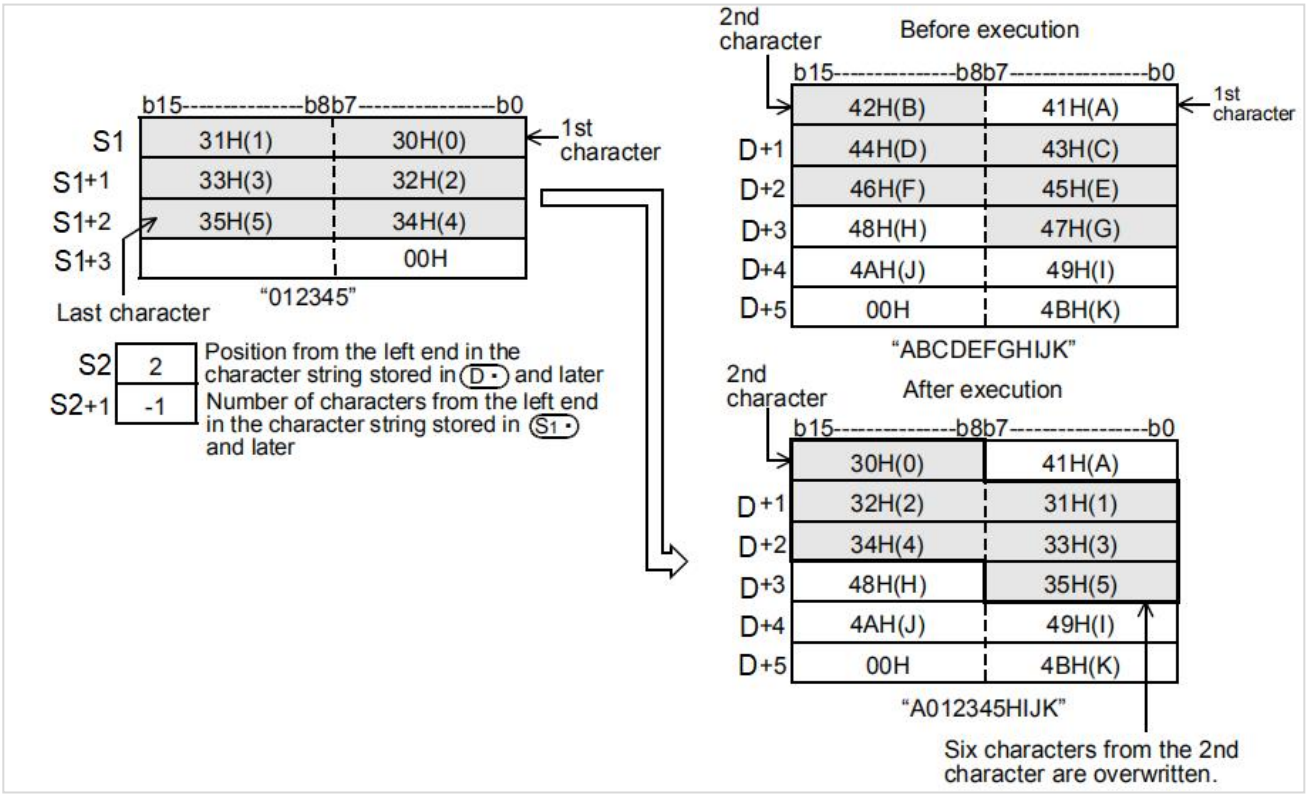
The final NULL code (00H) representing the string is automatically appended to the end of the string data.

When the number of characters specified in (s2)+1 is 0, no processing is performed.

If the number of characters designated in (s2)+1 exceeds the final character of the string data designated in (d), the data ending with the final character of (d) is stored.



If the character designated in (s2)+1 is -1, the data ending at the position of the final character data designated in (s1) is stored after device number specified in (d).



Precautions

Attention shall be paid to the following points when handling the character codes other than ASCII code.

The number of characters is processed in bytes (8 bits). Therefore, like the shifted JIS code, the character code of 1 character is represented by 2 bytes and the number of characters of 1 character is "2".

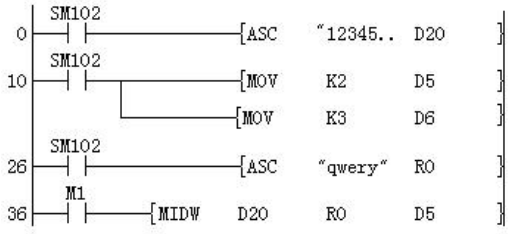
When extracting the string from the string for which the character code of 1 character is represented by 2 bytes like the shifted JIS code, the number of the characters extracted shall be determined by taking the character code of 1 character as the unit. Please

note that if only 1 byte of a 2-byte character code is extracted, it will not be the desired character code.

Error code

Error code	Content
4085H	(s1)(s2) (d) read address exceeds the range of the device.
408AH	(s1) (d) The length of the read string exceeds the limit, and the continuous length of the string exceeds 400 characters.
408BH	When (s1) (d) reading a character string, the maximum range of the device is read, but 00H is not found as the end.
4084H	The value of (s2)+1 is less than -2 (including -2); The value of (s2) exceeds the number of characters of (d); The value of (s2) is negative; The value of (s2)+1 exceeds the number of characters of (s1).
4086H	(d) Write address exceeds the device range.

Example



Replace the character with length of three characters starting from the second character in the string “qwery” stored in R0 with the first three characters in D20.

The result of R0 is “q123y”.

R0	1	0	0	0	1	1	1	0	1	0	0	0	1	1	0	0	q1
R1	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	23
R2	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	y.

12.7 STR/BIN16-bit data→string conversion

STR(P)

The BIN16-bit data specified in (s2) is converted to a character string with a decimal point at the position specified in (s1) and stored after device number specified in (d).

-[STR (s1) (s2) (d)]

Content, range and data type

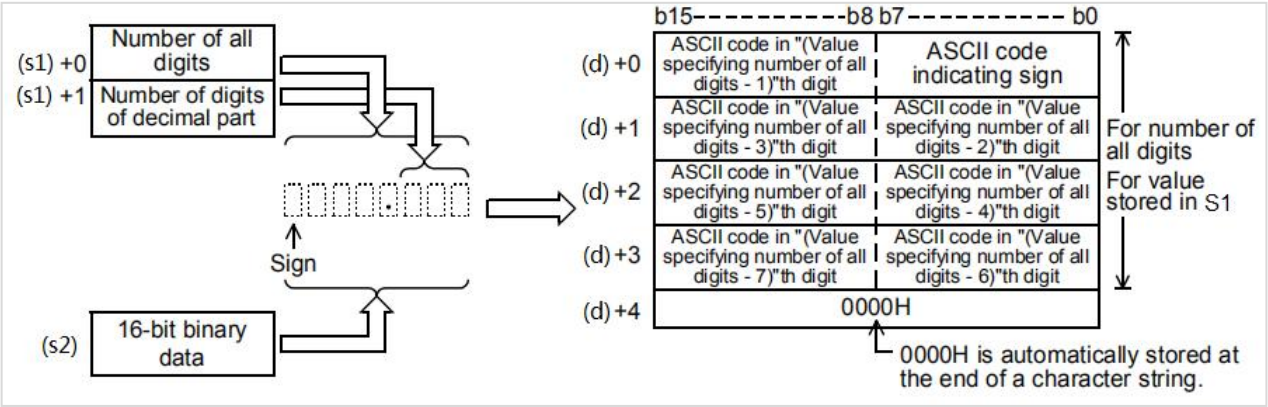
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s1)	The start number of device that stores the bits of the converted data	-	Signed BIN16 bits	ANY16_S_ARRAY	--
(s2)	Converted BIN data	-32768~+32767	Signed BIN16 bits	ANY16_S	INT
(d)	The start number of device that stores the converted string	-	String	ANYSTRING_SINGLE	--

Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
STR	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●			●	●		●	●
	Parameter 3															●	●	●	●	●						●	●

Function

The BIN16-bit data specified in (s2) is converted to a character string with a decimal point at the position specified in (s1) and stored after device number specified in (d).



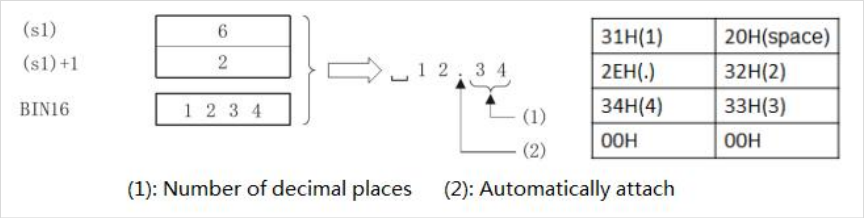
The total digits that can be specified in (s1) are 2~8 digits.

The number of the decimal parts that can be specified in (s1)+1 are 0~5 digits. However, the setting should meet the condition that the number of decimal parts ≤ (all digits-3).

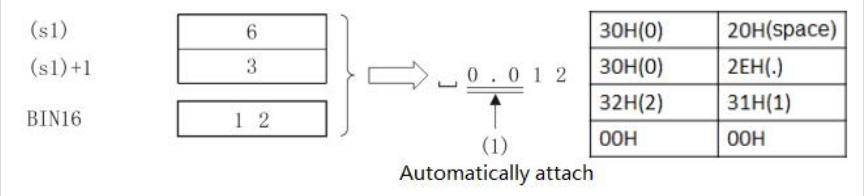
The string data after conversion will be stored to the device number after (d) in following ways.

•In the symbol, when BIN16-bit data is positive, 20H (blank) is stored; and when it is negative, 2DH(-) is stored.

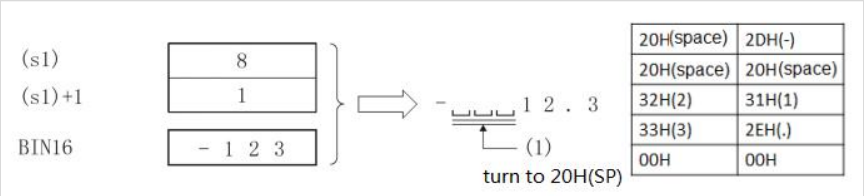
When the number of decimal places is set to any value other than 0, 2EH(.) is automatically stored in the number of specified decimal places+1 digit. . When the decimal place is 0, 2EH(.) is not stored. .



If the value of the number of decimal place is greater than the number of bits of BIN16-bit data, it will be automatically attached with 0 and then aligned to the right and converted to “0.***”.



If the symbol is removed from the value of all bits and the value of the number of decimal place is greater than the number of bits of BIN16-bit data, 20H (blank) will be stored between the symbol and the value. If the number of bits of BIN 16-bit data is large, it will become an error state.



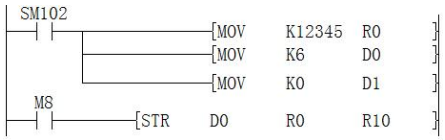
00H is automatically stored at the end of the converted character string.

•If the number of total digits is even, “0000H” will be stored in the device after the device storing the final character. If the number of total digits is odd, “00H” will be stored in the high byte (8 bits) of the device storing the final character.

Error code

Error code	Content
4085H	(s1) and (s2) read address exceeds the range of the device.
4084H	(s1) or (s1+1) parameter setting value exceeds the range. For example: 1. The value of (s1) is not within the range of 2-8. 1. The value of (s1+1) is not within the range of 0-5. 3. The value of (s1+1) is greater than the value of (s1) minus 3. 4. When (s1+1) is 0, the number of bits specified in (s1) is less than the bit number of BIN16-bit data specified in (s2)+1. 4. When (s1+1) is not 0, the number of bits specified in (s1) is less than the bit number of BIN16-bit data specified in (s2)+2. (Number of digits of (s1))< number of bits of BIN 16-bit data without symbols of (s2) + the number of bits of symbols (+ or-) + the decimal point (.) Number of digits
4086H	When using offset, the offset address of (d) exceeds the device range.

Example



After M8 is set as ON, it is converted into a string “12345” (a space in front of 1) according to the setting of all digits (6 decimal places and 0 digit).

R10	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	1
R11	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	23
R12	0	0	1	0	1	1	0	0	1	0	1	0	1	1	0	0	45

12.8 DSTR/BIN32-bit data→string conversion

DSTR(P)

The BIN32-bit data specified in (s2) is converted to a character string with a decimal point at the position specified in (s1) and stored after device number specified in (d).

-[DSTR (s1) (s2) (d)]

Content, range and data type

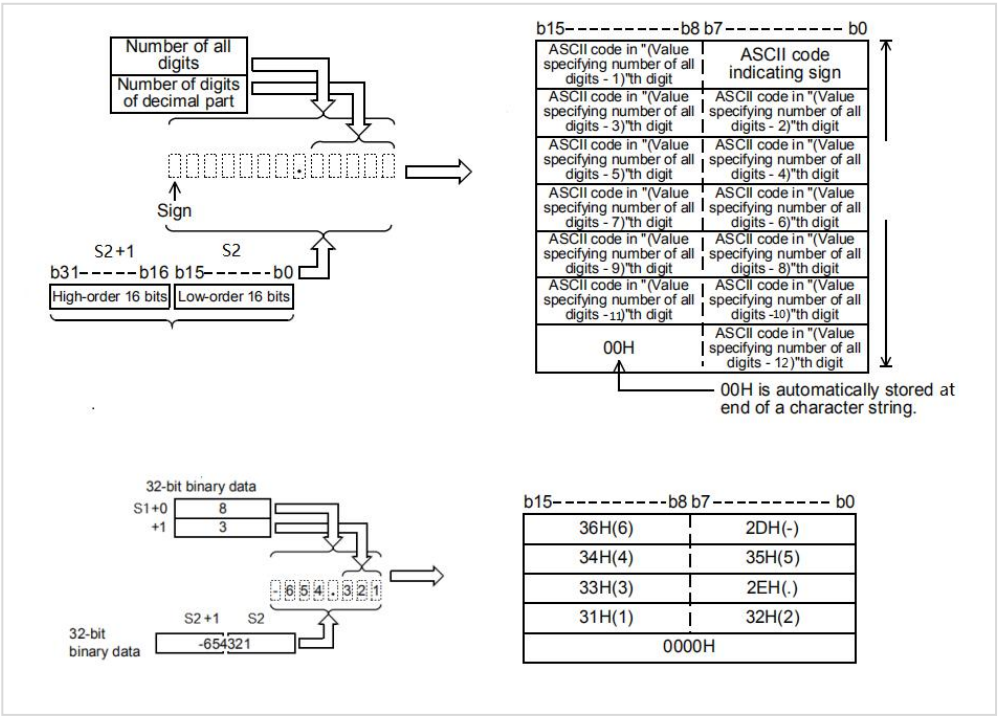
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s1)	The start number of device that stores the bits of the converted data	-	Signed BIN16 bits	ANY16_S_ARRAY	--
(s2)	Converted BIN data	-2147483648 to 2147483647	Signed BIN 32 bit	ANY32_S	DINT
(d)	The start number of device that stores the converted string	-	String	ANYSTRING_SINGLE	--

Device used

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DSTR	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●	●	●	●	●		●	●
	Parameter 3															●	●	●	●	●						●	●

Function

The BIN32-bit data specified in (s2) is converted to a character string with a decimal point at the position specified in (s1) and stored after device number specified in (d).



When -654.321 is specified in (s2):

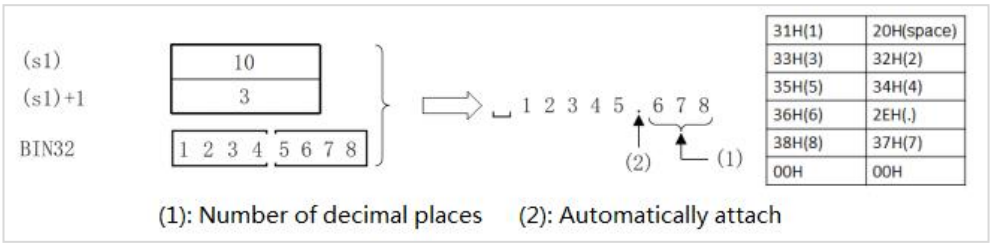
The total digits that can be specified in (s1) are 2~13 digits.

The number of the decimal places that can be specified in (s1)+1 are 0~10 digits. However, the setting should meet the condition that the number of decimal parts ≤ (all digits-3).

The string data after conversion will be stored to the device number after (d) in following ways.

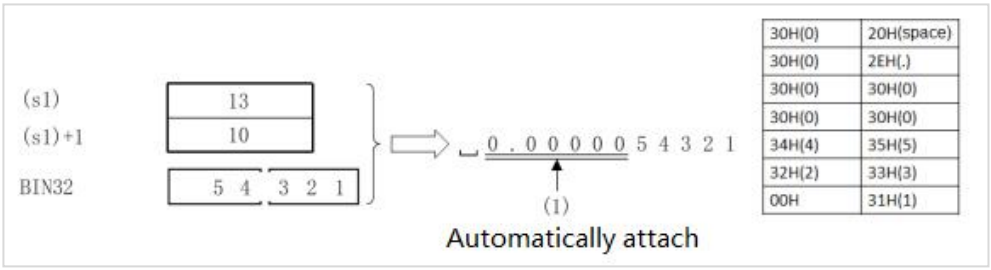
• In the symbol, when BIN32-bit data is positive, 20H (blank) is stored; and when it is negative, 20DH(-) is stored.

• When the number of decimal places is set to any value other than 0, 2EH(.) is automatically stored in the number of specified decimal places+1 digit. . When the decimal place is 0, 2EH(.) is not stored.



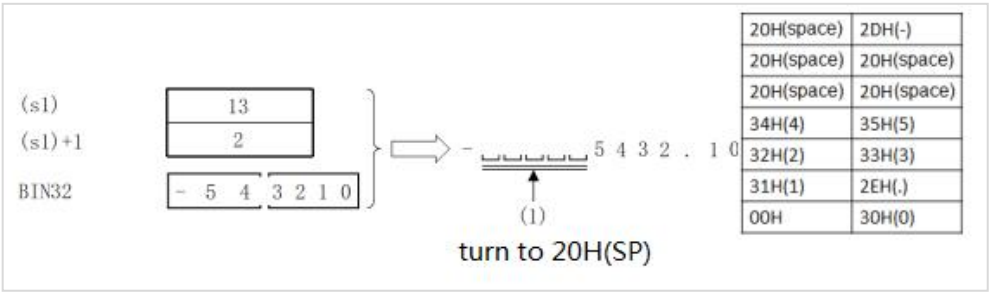
31H(1)	20H(space)
33H(3)	32H(2)
35H(5)	34H(4)
36H(6)	2EH(.)
38H(8)	37H(7)
00H	00H

•If the value of the number of decimal place is greater than the number of bits of BIN32-bit data, it will be automatically attached with 0 and then aligned to the right and converted to “0.***”.



30H(0)	20H(space)
30H(0)	2EH(.)
30H(0)	30H(0)
30H(0)	30H(0)
34H(4)	35H(5)
32H(2)	33H(3)
00H	31H(1)

•If the symbol is removed from the value of all bits and the value of the number of decimal place is greater than the number of bits of BIN32-bit data, 20H (blank) will be stored between the symbol and the value. If the number of bits of BIN 16-bit data is large, it will become an error state.



20H(space)	2DH(-)
20H(space)	20H(space)
20H(space)	20H(space)
34H(4)	35H(5)
32H(2)	33H(3)
31H(1)	2EH(.)
00H	30H(0)

•00H is automatically stored at the end of the converted character string.

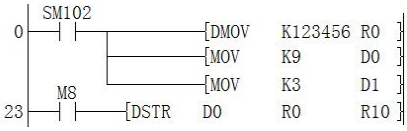
•If the number of total digits is even, “0000H” will be stored in the device after the device storing the final character. If the number of total digits is odd, “00H” will be stored in the high byte (8 bits) of the device storing the final character.

Error code

Error code	Content
4085H	(s1) and (s2) read address exceeds the range of the device.
4084H	(s1) or (s1+1) parameter setting value exceeds the range. For example: 1. The value of (s1) is not within the range of 2-13. 2. The value of (s1+1) is not within the range of 0-10. 3. The value of (s1+1) is greater than the value of (S1) minus 3. 4. When (s1+1) is 0, the number of bits specified in (s1) is less than the bit number of BIN16-bit data specified in (s2)+1. 4. When (s1+1) is not 0, the number of bits specified in (s1) is less than the bit number of BIN16-bit data specified in (s2)+2. (Number of digits of (s1))< number of bits of BIN 16-bit data without symbols of (s2) + the number of bits of symbols (+ or-) + the decimal point (.) Number of digits

Error code	Content
4086H	When using offset, the offset address of (d) exceeds the device range.

Example



As shown in the example, we need to convert 123456 to a floating-point string with three digits after nine decimal places.

The result of the conversion should be: 123.456, and the preceding value will have two spaces to supplement the insufficient number.

R10	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	
R11	1	0	0	0	1	1	0	0	0	1	0	0	1	1	0	0	12
R12	1	1	0	0	1	1	0	0	0	1	1	1	0	1	0	0	3.
R13	0	0	1	0	1	1	0	0	1	0	1	0	1	1	0	0	45
R14	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	6.

12.9 $+/combination of string

$+(P)

Connect the string stored after the device number specified in (s2) to the string data stored after the device number specified in (s1), and store it after the device number specified in (d) .

-[$+ (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s1)	Connect data or the device start number storing the data or the string specified directly	-	String	ANYSTRING_SINGLE	--
(s2)	Connected data or the start number of device that stores the connected data or the string specified directly	-	String	ANYSTRING_SINGLE	--
(d)	The start number of device that stores the connection result	-	String	ANYSTRING_SINGLE	--

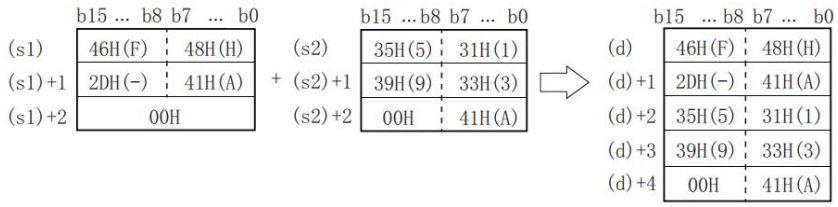
Device used

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
$+	Parameter 1											●	●	●	●	●	●	●	●	●						●	●
	Parameter 2											●	●	●	●	●	●	●	●	●						●	●
	Parameter 3												●	●	●	●	●	●	●	●						●	●

Function

Connect the string stored after the device number specified in (s2) to the string data stored after the device number specified in (s1), and store it after the device number specified in (d) .

The string of (s1) and (s2) starts from the designated device number and ends at the device number that stores 00H.



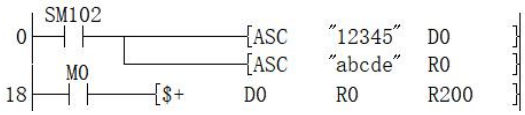
When strings are merged, 00H representing the end of the string specified in (s1) is ignored, and the string specified in (s2) is connected at the final character of (s1).

If strings are merged, 00H will be automatically attached at the end. If the number of connected characters is odd, “00H” will be stored in the high byte of the device storing the final character. If the number of extracted characters is even, “0000H” will be stored in the device after the device storing the final character.

Error code

Error code	Content
4085H	The read address of (s1) or (s2) exceeds the device range.
408AH	The length of the read string of (s1) or (s2) exceeds the limit, and the continuous length of the string exceeds 400 characters.
408BH	When (s1) or (s2) reading string, the maximum range of device is read, but 00H is not found as the end.
4086H	(d) Write address exceeds the device range.

Example



After the string “12345” and string “abcde” are merged, the result is “12345abcde”.

R200	1	0	0	0	1	1	0	0	0	1	0	0	1	1	0	0	12
R201	1	1	0	0	1	1	0	0	0	0	1	0	1	1	0	0	34
R202	1	0	1	0	1	1	0	0	1	0	0	0	0	1	1	0	5a
R203	0	1	0	0	0	1	1	0	1	1	0	0	0	1	1	0	bc
R204	0	0	1	0	0	1	1	0	1	0	1	0	0	1	1	0	de

12.10 INSTR/string search

INSTR(P)

Start from the left (s3) character of the string data stored after device number specified in (s2), search the string data stored after the device number designated in (s1) and store the searched result in the device designated in (d).

-[INSTR (s1) (s2) (d) (s3)]

Content, range and data type

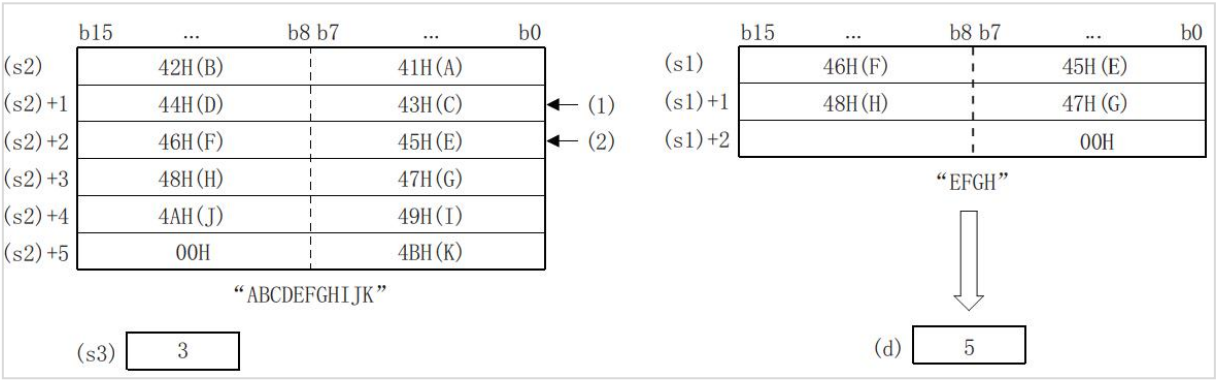
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s1)	Search string or the start number of device that stores the search string	-	String	ANYSTRING_SINGLE	--
(s2)	Searched string or the start number of device that stores the searched string	-	String	ANYSTRING_SINGLE	--
(d)	The start number of device that stores the search result	-	Signed BIN16 bits	ANY16	--
(s3)	Search start position	1 to 400	Signed BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
INSTR	Parameter 1															●	●	●	●	●						●	●
	Parameter 2															●	●	●	●	●						●	●
	Parameter 3											●	●	●	●	●	●	●	●	●						●	●
	Parameter 4											●	●	●	●	●	●	●	●	●			●	●		●	●

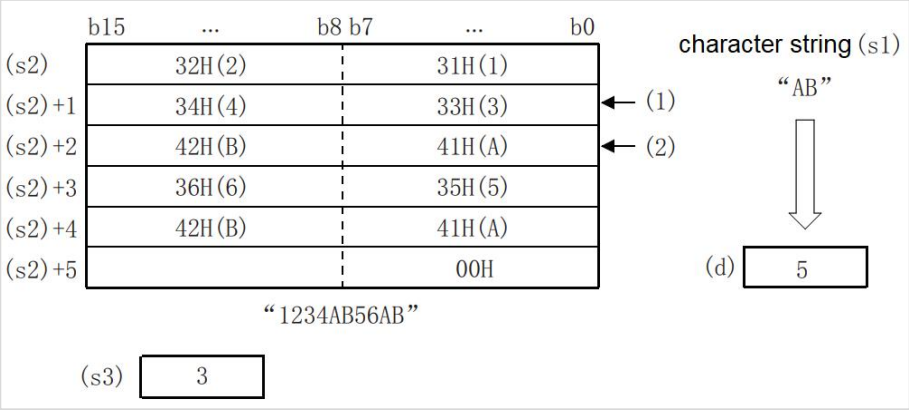
Function

Start from the left (s3) character of the string data stored after device number specified in (s2), search the string data stored after the device number designated in (s1) and store the searched result in the device designated in (d). The search result will store the first xx character of the start character of the string data specified in (s2).



If there is no consistent string data, 0 will be stored in (d).

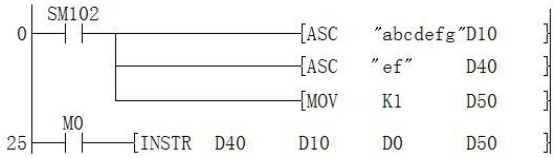
When the search start position (s3) is "0", no processing is performed. The searched string (s1) can directly designate the string.



Error code

Error code	Content
4085H	The write address in (s1), (s2) and (s3) exceeds the device range.
408AH	The length of the read string of (s1) and (s2) exceeds the limit, and the continuous length of the string exceeds 400 characters.
408BH	When (s1) and (s2) reading the string, the maximum range of device is read, but 00H is not found as the end.
4086H	(d) Write address exceeds the device range.
4084H	(s3) <0 or (s3)>= string length

Example



When searching “ef” string from the continuous string “abcdefg”, start searching from the first character and locate it at the 5th place.

12.11 ASC/ASCII data input

ASC

Instruction used to convert half-width/English numeric strings into ASCII codes. Used to select multiple messages for display in an external display.

-[ASC(s)|(d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	32-character half-width English numerals input by the computer	-	String (only ASCI code)	ANY_ASC	--
(d)	Start character device number that saves ASCII data	-	BIN16 bit	ANY16_S	--

Device used

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ASC	Parameter 1																										
	Parameter 2															●	●	●	●	●						●	

Function

1. 16-bit operation (ASC)

•Convert half-width, English and numeric strings specified in (S) into ASCII codes and transmit them to (D).

(1) In (S), half-width characters of A ~ Z, 0 ~ 9 and symbols are processed. (Full-width strings are not processed.) When programming with a programming tool, enter a string.

(2) The converted ASCII code is stored in (D) every 2 charters/1 byte in the order of lower 8 bits and higher 8 bits.

2. Extended function

•After SM162 is set as ON, the extended function becomes valid; then convert half-width, English and numeric strings specified in (S) into ASCII codes and transmit them to the lower 8 bits of (D) (1 byte) in proper order.

Precautions

1. Occupied places of the device

(1) When the extended function is OFF, the number of characters occupied by D÷ 2 places (when it is not divisible, the decimal place is carried.)

(2) When the extended function is ON, the places occupied by D are the same as the number of characters occupied.

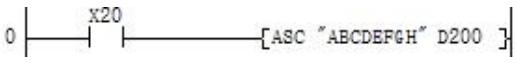
2. When using instructions such as ASC, ASCI, BCC, CCD and CRC, the extended function sign bit SM161 is the sign bit commonly used for other instructions.

When using the above instructions and ASC instructions, please note that the program SM161 ON or OFF is written in front of the ASC instructions so as to avoid adverse impact.

Error code

Error code	Content
4085H	The output result of the read application instruction (s) exceeds the device range.
4086H	The output result of the write application instruction (d) exceeds the device range.

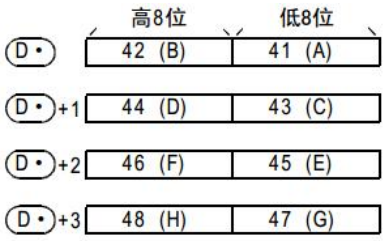
Example



As shown in the following figure, the high byte of each variable is filled with 0 (hexadecimal):

	D•		S
	(High 8 bits)	(Low 8 bits)	String
D•	00	41	A
D• +1	00	42	B
D• +2	00	43	C
D• +3	00	44	D
D• +4	00	45	E
D• +5	00	46	F
D• +6	00	47	G
D• +7	00	48	H

When X20=ON, the assignment of D200~D203 is shown as follows:



When the special register SM161 is set as ON,

each ASCII character occupies one 16-bit variable after conversion.

13 Ethernet communication

13.1 Overview of Ethernet

13.1.1 IP Address concept

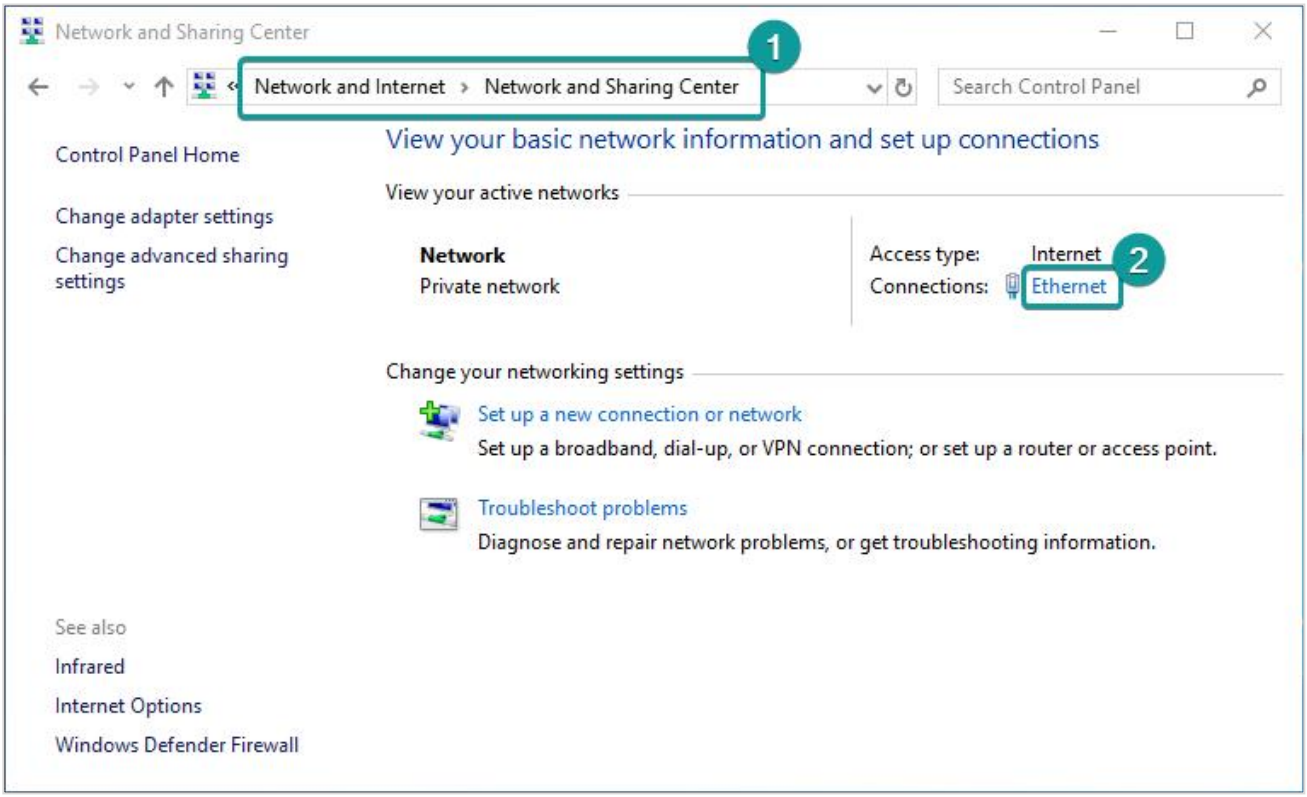
IP addresses consist of network addresses and host addresses, which are distinguished by subnet masks. If a programming device (such as a PC) needs to be connected to a factory LAN via a network card, the programming device and the PLC must be in the same subnet. The IP address is combined with a subnet mask to specify the subnet of the device.

No.	Network equipment 1			Network equipment 2			Network connection
	IP	Subnet mask	Network address	IP	Subnet mask	Network address	
1	192.168.0.1	255.255.255.0	192.168.0.0	192.168.0.10	255.255.255.0	192.168.0.0	Yes
2	192.168.0.1	255.255.255.0	192.168.0.0	192.168.1.10	255.255.255.0	192.168.1.0	No
3	192.168.0.1	255.255.255.1	192.168.0.1	192.168.0.10	255.255.255.1	192.168.0.0	No

When logical operation of IP address and subnet mask is performed, the network address can be calculated. If the addresses are in the same network, it means that communication can be carried out. For instance:

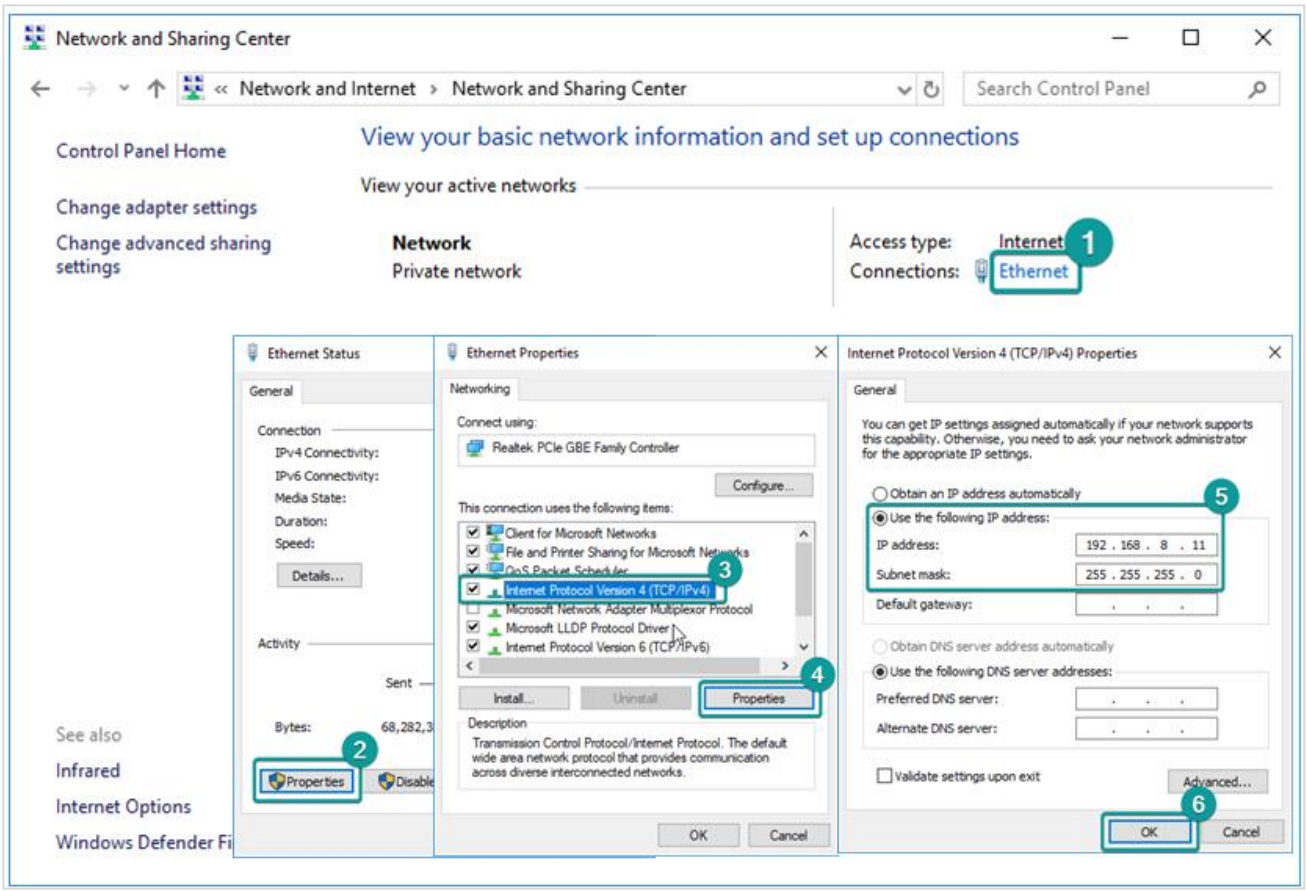
13.1.2 Setting PC Network address

1. Open the control panel->Network and Internet->Network and sharing center;



2. Click on the local link->Attributes->Internet Protocol Version 4;

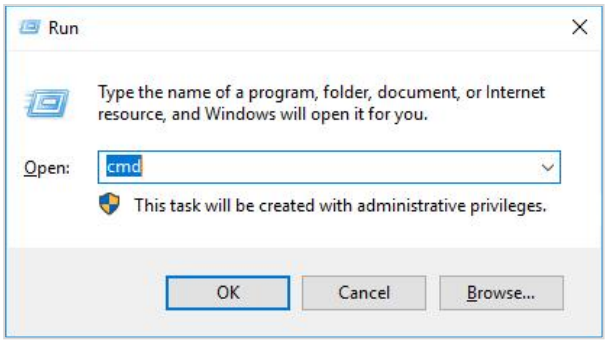
3. Set the IP address and subnet mask that are within the same network address as PLC. Note that IP addresses that have been used in LAN cannot be set. If the IP of PLC is 192.168. 8.8 and the subnet mask is 255.255. 255.0, the IP address as shown in the following figure can be set to connect the PC to PLC.



13.1.3 Testing Network connection status

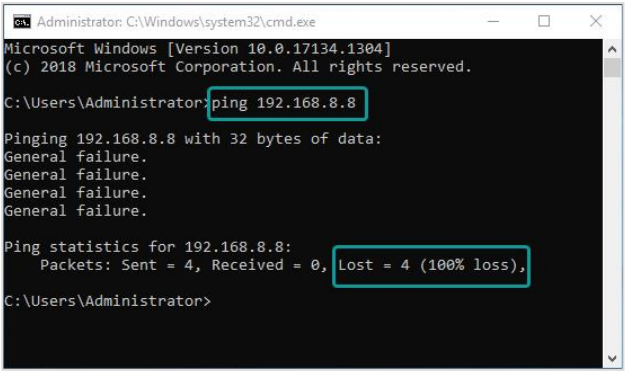
Test the connection status of PC and PLC via ping instruction.

1. win+R key, input cmd;

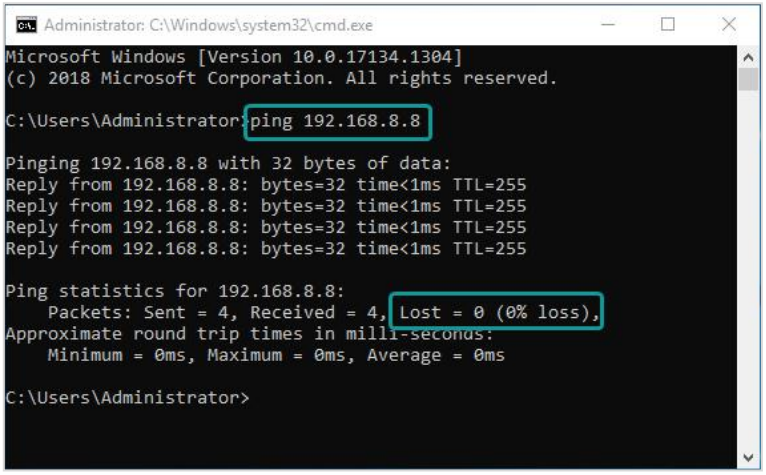


2. If PLC's IP address is 192.168.8.8;

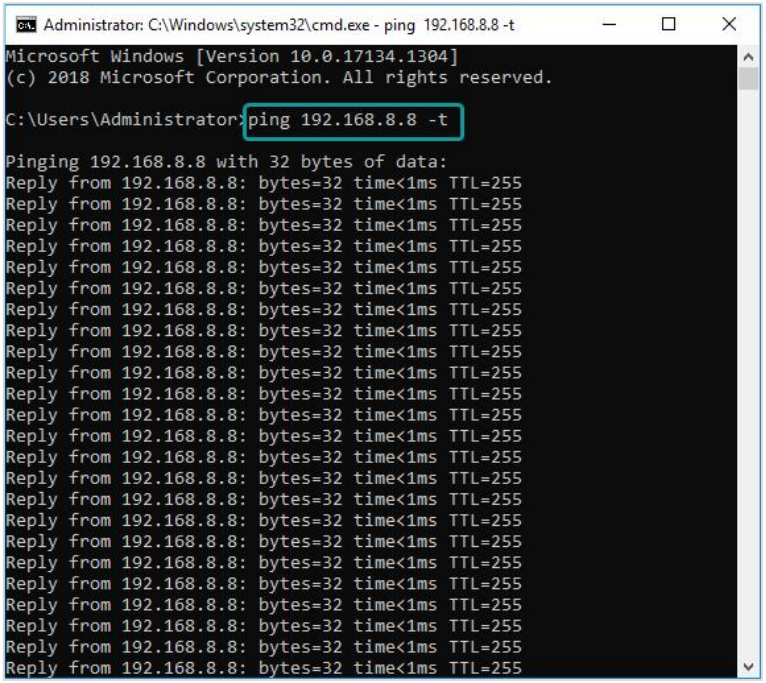
(1) Enter ping 192.168. 8.8 instruction and press Enter. If 100% missing is displayed, it means that it cannot be connected with PLC.



(2) Enter ping 192.168. 8.8 instruction and press Enter. If 0% missing is displayed, it means that it can be connected with PLC.

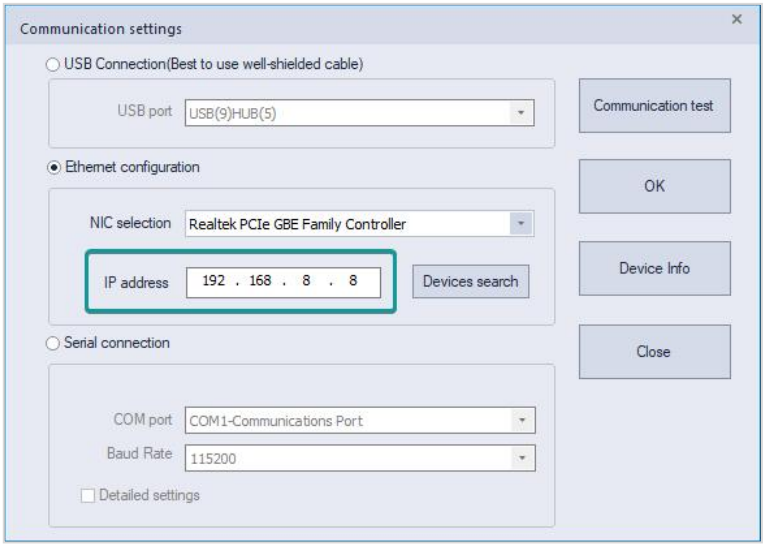


(3) “ping network equipment ip”instruction can only be used for four times . If it needs to further perform ping , please use the instruction “ping network equipmentip -t”, as shown in following figure:

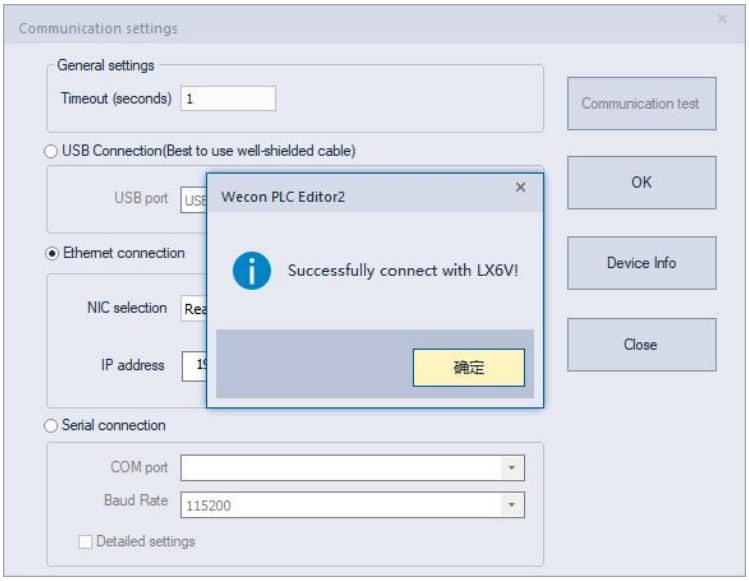


13.1.4 Ethernet connection between PLC Editor2 and PLC

1. Transmission settings->Ethernet configuration->Input the IP address of PLC; (note: the address of the network card should be in the same network segment with PLC)



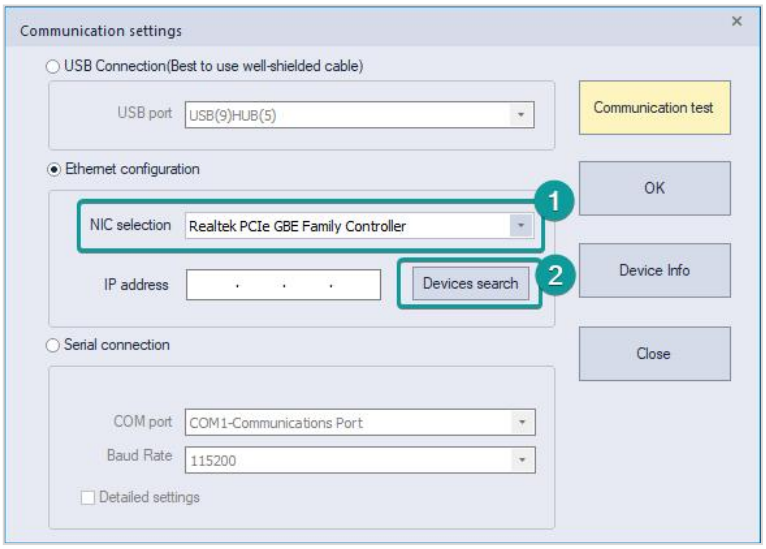
3. Click the communication test to confirm the communication;



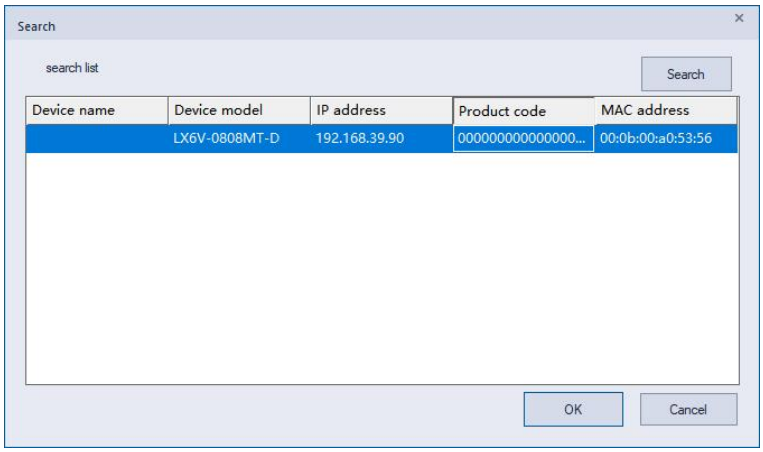
3. After successful connection, click “select” to operate PLC.

13.1.5 Ethernet Search Function of PLC Editor 2

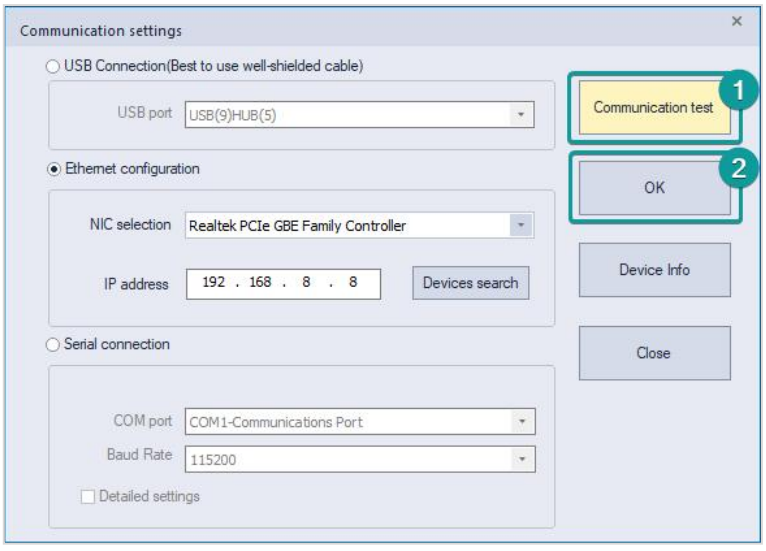
1. Transmission settings->Network card selection->Equipment search (note: the address of the network card should be in the same network segment with PLC), select “connect PLC” at 1 or the network card that is within the same network segment with PLC, and click 2 to search PLC equipment in LAN.



2. Search interface is shown as follows. Click search to display the PLC equipment in the LAN, select the corresponding equipment and click OK to communicate.



3. IP address at 1 is automatically filled in. Click the communication test to test whether the communication is normal. if it is normal, click “select” button to complete the connection configuration.



13.2 Ethernet configuration

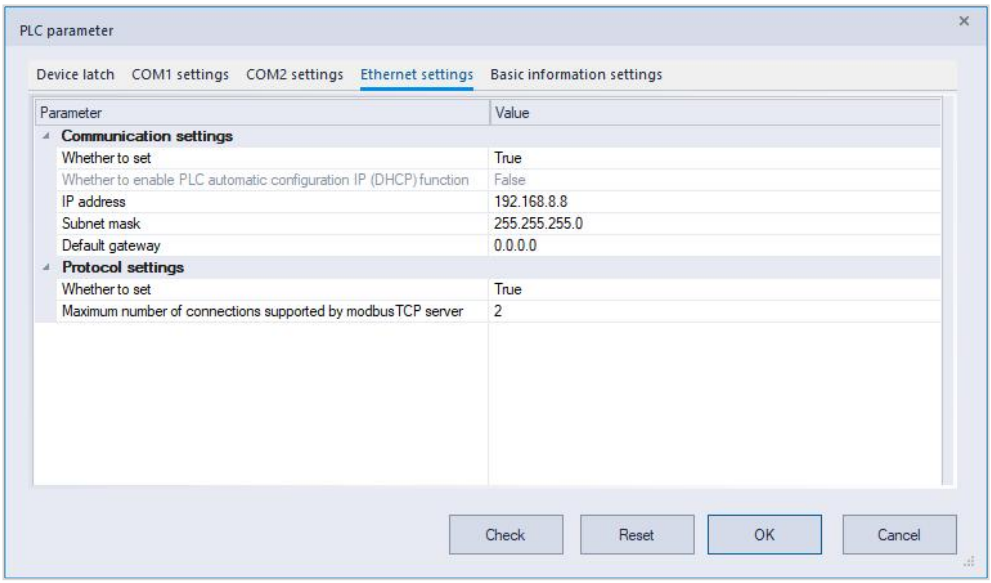
13.2.1 IP Address setting

(1) Setting via programming software

Project management->parameters->PLC parameter->Ethernet configuration.

After modification, check the parameter items through PLC download; after download, click STOP->RUN to complete.

Note: Modbus TCP Server supports maximum connections, which is used to set the maximum number of external Modbus TCP clients that can be connected to PLC simultaneously, with the range being 2~8.



(2) Setting via special device

Write IP address, subnet mask and default gateway in SD2680~SD2691;

When SM2680 is set as ON, open static IP function;

When SM2683 is set as ON, the sign IP can be modified;

STOP->RUN or the new IP address takes effect after power-on again.

SM sign	Name	Content	R/W	SD label	Name
SM2680	Statically set IP switch	ON: static setting OFF: The IP address is automatically configured by router DHCP, and the IP cannot be modified STOP->RUN	R/W	SD2680	The first byte of the IP address
SM2681	Display current network information	Refresh the current IP gateway subnet mask after being set as ON. OFF after refresh is completed.	R/W	SD2681	The second byte of the ip address
SM2682	Display current MAC information (reserved)	Refresh the current MAC after being set as ON. OFF after refresh is completed.	R/W	SM2682	The third byte of the ip address
SM2683	IP, subnet mask, gateway modification sign	ON: can be modified OFF: cannot be modified (When being set as ON, STOP->RUN; it can be automatically changed to OFF after modification)	R/W	SD2683	The fourth byte of the ip address
SM2684				SD2684	The first byte of subnet mask
SM2685				SD2685	The second byte of subnet mask

SM2686				SD2686	The third byte of subnet mask
SM2687				SD2687	The fourth byte of subnet mask
SM2688				SD2688	Default gateway first byte
SM2689				SD2689	Default gateway second byte
SM2690				SD2690	Default gateway third byte
SM2691				SD2691	Default gateway fourth byte

13.2.2 TCP Protocol concept

TCP protocol is a Transport Control Protocol, which is a connection-oriented and reliable transport layer protocol. Connection-oriented means that a normal TCP transmission needs to be completed by establishing a specific virtual circuit connection between TCP client and TCP server. To transmit data over TCP, a connection must be established between hosts at both ends.

13.3 List of Ethernet-related special devices

SM sign	Name	Content	R/W	Power-down save
SM2681	Display current network information	Refresh the current IP, subnet mask and default gateway after being set as ON; and then being set as OFF upon completion of refresh	R/W	×
SM2682	Display current MAC information	Refresh the current MAC after being set as ON. OFF after refresh is completed.	R/W	×
SM2683	IP, subnet mask, gateway modification sign	ON: can be modified OFF: cannot be modified (When being set as ON, STOP->RUN; it can be automatically changed to OFF after modification)	R/W	√
SM2684	Network card connection status	ON: network connected OFF: network not connected; check if the network cable is connected	R	√
SM2692	MAC address modification sign	ON: can be modified OFF: cannot be modified (When being set as ON, STOP->RUN; it can be automatically changed to OFF after modification)	R/W	√
SM2700	Modbus TCP keep-alive mechanism	ON: Open OFF: Close (default)	R/W	√
SM2701	Forced shutdown of Modbus TCP server	ON: Open OFF: Close (default) (after being enabled, it will be automatically changed into OFF after shutdown)	R/W	×
SM2710	Ethernet error sign	ON: there is an Ethernet error, please check SD2710 and SD2711 OFF: there is no Ethernet error	R	×
SM2740	ModbusTCP server connection status 1	ON: Client side has been connected OFF: Client side hasn't been connected	R	×
SM2760	ModbusTCP server connection status 2	ON: Client side has been connected OFF: Client side hasn't been connected	R	×
SM2780	ModbusTCP server connection status 3	ON: Client side has been connected OFF: Client side hasn't been connected	R	×
SM2800	ModbusTCP server connection status 4	ON: Client side has been connected OFF: Client side hasn't been connected	R	×
SM2820	ModbusTCP server connection status 5	ON: Client side has been connected OFF: Client side hasn't been connected	R	×
SM2840	ModbusTCP server connection status 6	ON: Client side has been connected OFF: Client side hasn't been connected	R	×
SM2860	ModbusTCP server connection status 7	ON: Client side has been connected OFF: Client side hasn't been connected	R	×
SM2880	ModbusTCP server connection status 8	ON: Client side has been connected OFF: Client side hasn't been connected	R	×

SD label	Name	Content	R/W	Power-down save
SD2680	The first byte of the IP address	Local IP address	R/W	√
SD2681	The second byte of the ip address		R/W	√
SM2682	The third byte of the ip address		R/W	√
SD2683	The fourth byte of the ip address		R/W	√
SD2684	The first byte of subnet mask	Local subnet mask	R/W	√
SD2685	The second byte of subnet mask		R/W	√
SD2686	The third byte of subnet mask		R/W	√
SD2687	The fourth byte of subnet mask		R/W	√
SD2688	Default gateway first byte	Local default gateway	R/W	√
SD2689	Default gateway second byte		R/W	√
SD2690	Default gateway third byte		R/W	√
SD2691	Default gateway fourth byte		R/W	√
SD2692	The first byte of MAC address	Local MAC address	R/W	√
SD2693	The second byte of MAC address		R/W	√
SD2694	The third byte of MAC address		R/W	√
SD2695	The fourth byte of MAC address		R/W	√
SD2696	The fifth byte of MAC address		R/W	√
SD2697	The sixth byte of MAC address		R/W	√
SD2700	Communication speed display	0: 100Mbps/half duplex 1: 100Mbps/full duplex 2: 10Mbps/half duplex 3: 10Mbps/full duplex	R	×
SD2702	The modbusTCP server supports the maximum number of connections	Local modbusTCP server supports the maximum number of clients to make connections at the same time	R/W	×

SD label	Name	Content	R/W	Power-down save
SD2703	Modbus TCP connections	Local Modbus TCP connections	R	×
SD2710	Error code	Ethernet error code	R	×
SD2711	Socket ID of the error	-1: system default Modbus TCP server 0~5: Customer custom socket error	R	×
SD2720	Input the low bit of the number of ping request	External input ping instruction count	R	×
SD2721	Input the high bit of the number of ping request		R	×
SD2722	Input the low bit of the number of ping response	Number of replies after receiving external ping instruction	R	×
SD2723	Input the high bit of the number of ping response		R	×
SD2724	Output the low bit of the number of ping request	Send ping instruction count	R	×
SD2725	Output the high bit of the number of ping request		R	×
SD2726	Output the low order of the number of ping response	Number of replies after receiving sent ping instruction	R	×
SD2727	Output the high bit of the number of ping response		R	×
SD2728	Number of arp packets sent	Count of number of arp packets sent	R	×
SD2729	Number of arp packets received	Count of number of arp packets received	R	×
SD2730	Number of ip packets sent	Count of number of ip packets sent	R	×
SD2731	Number of ip packets received	Count of number of arp packets received	R	×
SD2732	Number of tcp packets sent	Count of number of tcp packets sent	R	×
SD2733	The number of tcp packets received	Count of number of tcp packets received	R	×
SD2734	Number of udp packets sent	Count of number of udp packets sent	R	×
SD2735	Number of udp packets received	Count of number of udp packets received	R	×
SD2740	Connection 1 local port number	Connection information and error information about the first Modbus TCP client's connection to the PLC.	R	×
SD2741	Connection 1 IP address first byte		R	×

SD label	Name	Content	R/W	Power-down save
SD2742	Connection 1 IP address second byte		R	×
SD2743	Connection 1 IP address third byte		R	×
SD2744	Connection 1 IP address fourth byte		R	×
SD2745	Connection 1 peer port number		R	×
SD2746	Reserved		R	×
SD2747	Reserved			
SD2748	Connection 1 error code		R	×
SD2749	Connection 1 error communication times low word		R	×
SD2750	Connection 1 error communication times high word		R	×
SD2760	Connection 2 local port number	Connection information and error information about the second Modbus TCP client's connection to the PLC.	R	×
SD2761	Connection 2 IP address first byte		R	×
SD2762	Connection 2 IP address second byte		R	×
SD2763	Connection 2 IP address third byte		R	×
SD2764	Connection 2 IP address fourth byte		R	×
SD2765	Connection 2 peer port number		R	×
SD2766	Reserved		R	×
SD2767	Reserved			
SD2768	Connection 2 error code		R	×
SD2769	Connection 2 error communication times low word		R	×
SD2770	Connection 2 error communication times high word		R	×
SD2780	Connection 3 local port number	Connection information and error information about the third Modbus TCP client's connection to the PLC.	R	×
SD2781	Connection 3 IP address first byte		R	×

SD label	Name	Content	R/W	Power-down save
SD2782	Connection 3 IP address second byte		R	×
SD2783	Connection 3 IP address third byte		R	×
SD2784	Connection 3 IP address fourth byte		R	×
SD2785	Connection 3 peer port number		R	×
SD2786	Reserved		R	×
SD2787	Reserved			
SD2788	Connection 3 error code		R	×
SD2789	Connection 3 error communication times low word		R	×
SD2780	Connection 3 error communication times high word		R	×
SD2800	Connection 4 local port number	Connection information and error information about the fourth Modbus TCP client's connection to the PLC.	R	×
SD2801	Connection 4 IP address first byte		R	×
SD2802	Connection 4 IP address second byte		R	×
SD2803	Connection 4 IP address third byte		R	×
SD2804	Connection 4 IP address fourth byte		R	×
SD2805	Connection 4 peer port number		R	×
SD2806	Reserved		R	×
SD2807	Reserved			
SD2808	Connection 4 error code	Error information about the fourth Modbus TCP client's connection to the PLC.	R	×
SD2809	Connection 4 error communication times low word		R	×
SD2810	Connection 4 error communication times high word		R	×
SD2820	Connection 5 local port number	Connection information and error information about the fifth Modbus TCP client's connection to the PLC.	R	×
SD2821	Connection 5 IP address first byte		R	×

SD label	Name	Content	R/W	Power-down save
SD2822	Connection 5 IP address second byte		R	×
SD2823	Connection 5 IP address third byte		R	×
SD2824	Connection 5 IP address fourth byte		R	×
SD2825	Connection 5 peer port number		R	×
SD2826	Reserved		R	×
SD2827	Reserved			
SD2828	Connection 5 error code		R	×
SD2829	Connection 5 error communication times low word		R	×
SD2830	Connection 5 error communication times high word		R	×
SD2840	Connection 6 local port number	Connection information and error information about the sixth Modbus TCP client's connection to the PLC.	R	×
SD2841	Connection 6 IP address first byte		R	×
SD2842	Connection 6 IP address second byte		R	×
SD2843	Connection 6 IP address third byte		R	×
SD2844	Connection 6 IP address fourth byte		R	×
SD2845	Connection 6 peer port number		R	×
SD2846	Reserved		R	×
SD2847	Reserved			
SD2848	Connection 6 error code		R	×
SD2849	Connection 6 error communication times low word		R	×
SD2850	Connection 6 error communication times high word		R	×
SD2860	Connection 7 local port number	Connection information and error information about the seventh Modbus TCP client's connection to the PLC.	R	×
SD2861	Connection 7 IP address first byte		R	×

SD label	Name	Content	R/W	Power-down save
SD2862	Connection 7 IP address second byte		R	×
SD2863	Connection 7 IP address third byte		R	×
SD2864	Connection 7 IP address fourth byte		R	×
SD2865	Connection 7 peer port number		R	×
SD2866	Reserved		R	×
SD2867	Reserved			
SD2868	Connection 7 error code		R	×
SD2869	Connection 7 error communication times low word		R	×
SD2870	Connection 7 error communication times high word		R	×
SD2880	Connection 8 local port number	Connection information and error information about the eighth Modbus TCP client's connection to the PLC.	R	×
SD2881	Connection 8 IP address first byte		R	×
SD2882	Connection 8 IP address second byte		R	×
SD2883	Connection 8 IP address third byte		R	×
SD2884	Connection 8 IP address fourth byte		R	×
SD2885	Connection 8 peer port number		R	×
SD2866	Reserved	Connection information and error information about the eighth Modbus TCP client's connection to the PLC.		
SD2867	Reserved		R	×
SD2888	Connection 8 error code		R	×
SD2889	Connection 8 error communication times low word		R	×
SD2890	Connection 8 error communication times high word		R	×

13.4 List of Ethernet Operation Error Codes

Error code	Description	Action	Processing approach	Test time
3680	Ethernet data reception error	Continue to run	Check whether there is any interference in the environment	When the instruction is executed
3681	Ethernet data reception timeout	Continue to run	Check if the network cable is loosened 1. Check whether the network peer is faulty, which results in failure in sending data. Check whether the network peer does not respond in time, which results in any delay in data. Under such circumstance, the reception timeout in the socket configuration can be increased.	When the instruction is executed
3684	Modbus TCP station number configuration error	Continue to run	Check slave station number setting. Check whether there is any problem with the receiving and sending mechanism of slave station.	When the instruction is executed
3685	Modbus TCP send buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
3686	Modbus TCP function code error	Continue to run	Check whether the set function code is a function code supported by PLC	When the instruction is executed
3687	Modbus TCP address error	Continue to run	Check whether the slave station has this address. (Please refer to Modbus exception 02).	When the instruction is executed
3688	Modbus TCP length error	Continue to run	Check whether the communication length exceeds the Modbus range	When the instruction is executed
3689	Modbus TCP data error	Continue to run	Check whether parameters of the instruction are wrong. Check whether the set value is supported by the slave station. (Please refer to Modbus exception 03).	When the instruction is executed
368A	Modbus TCP slave is busy	Continue to run	Message returned from slave station: Slave station is busy. (Please refer to Modbus exception 06).	When the instruction is executed

Error code	Description	Action	Processing approach	Test time
368B	Modbus TCP slave station does not support function code	Continue to run	Check whether this function code is supported by the slave station. (Please refer to Modbus exception 01).	When the instruction is executed
368C	Modbus TCP slave failure	Continue to run	Message returned from slave station: Slave station failed, please check whether slave station failed. (Please refer to Modbus exception 04).	When the instruction is executed
368D	Modbus TCP slave confirmation	Continue to run	Message returned from slave station: slave station confirmation. (Please refer to Modbus exception 05).	When the instruction is executed
368E	The current protocol of Modbus TCP does not support this instruction Instruction	Continue to run	The RS instruction cannot be used when the slave protocol is set, please change the protocol or close the contact before the RS instruction	When the instruction is executed
368F	Network port send timeout	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
3690	Receive buffer overflow	Continue to run	Check if the other device is sending data all the time	When the instruction is executed
36A0	Unavailable gateway for Modbus TCP	Continue to run	Message returned from the slave station: no available gateway. (Please refer to Modbus exception 0A).	When the instruction is executed
36A1	Modbus TCP indicates that no response was obtained from the target device. It usually means the device is not on the network.	Continue to run	Message returned from the slave station: The device is not in the network. (Please refer to Modbus exception 0B)	When the instruction is executed
36C0	ModbusTCP transaction identifier error	Continue to run	Check if there is network congestion and data can not be received in time.	When the instruction is executed
36C1	Modbus TCP server available links are full	Continue to run	Check if SD2702 and SD2703 have too many clients to link	When the instruction is executed
36C8	Insufficient space in the Ethernet stack	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36C9	Number of connections exceeds the limit	Continue to run	Confirm whether the total number of connections exceeds the limit	When the instruction is executed

Error code	Description	Action	Processing approach	Test time
36CA	The last send is not completed	Continue to run	Use the sending completion sign to determine that the current sending is completed, and then send the next one	When the instruction is executed
36CB	TCP write exception	Continue to run	Use the sign bit device to judge whether the current connection status is normal. If it is not normal, no data transmission will be performed. For example, after the closing sign is set, data transmission will not be performed.	When the instruction is executed
36CC	TCP output exception	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36CD	IP address is used	Continue to run	Please check if a connection exists with the same address information	When the instruction is executed
36CE	Error in server's acceptance of connection	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36CF	TCP receive buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36D0	TCP link failed	Continue to run	It may be caused by opening the TCP client when the network cable is not connected.	When the instruction is executed
36D1	Exception when actively closing the connection	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36D2	Abnormal shutdown occurred inside the protocol stack	Continue to run	May be shut down due to no answering for a long time. Check whether the peer is online and can ping.	When the instruction is executed
36D3	The peer initiates an RST link	Continue to run	1. Check whether the peer initiates abnormal shutdown. 2. As a client, the number of peer connections is full, or the peer port is not open.	When the instruction is executed
36D4	The protocol stack has a single-ended shutdown	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36D5	There is an IP address conflict	Continue to run	The same IP device exists in the LAN, please change the IP address	When the instruction is executed

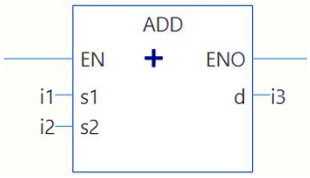
Error code	Description	Action	Processing approach	Test time
36D6	There is a MAC address conflict	Continue to run	The same MAC device exists in the LAN, please change the MAC address	When the instruction is executed
36D7	TCP send buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36D8	UDP connection exception	Continue to run	IP address or port number may has been used.	When the instruction is executed
36D9	UDP send buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36DA	Insufficient memory space when sending UDP	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36DB	UDP send failed	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36DC	UDP memory release failed	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36DD	UDP receive buffer overflow	Continue to run	The length of the data received by UDP exceeds the limit value of 512.	When the instruction is executed
4084	Data beyond the specified range is entered in the application instruction (for example, parameter 1 is specified as 0~1, setting 2)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4085	The output result of the read application instruction exceeds the device range. (For example, the maximum output of the D device is D7999, but D8000 is used)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4086	The output result of the write application instruction exceeds the device range. (For example, the maximum output of the D device is D7999, but D8000 is used)	Continue to run	Modify application instruction parameters	When the application instruction is executed
5080	The Ethernet socket has been linked and cannot be opened repeatedly.	Continue to run	Check if SOCOPEN instruction is executed repeatedly	When the application instruction is executed

Error code	Description	Action	Processing approach	Test time
5081	Ethernet socket is not open, which results in the failure in operation	Continue to run	Check whether set ON the connected sign bit of the SOCOPEN instruction (d2) parameter	When the application instruction is executed
5082	Socket ID input by Ethernet instruction is out of range.	Continue to run	Modify application instruction parameters	When the application instruction is executed
5083	Failed to create TCP server	Continue to run	Check if the connection is full	When the application instruction is executed
5084	Failed to create link	Continue to run	Check if the connection is full	When the application instruction is executed
5086	The socket ID used by the Ethernet instruction is not configured by the upper computer or is not enabled after configuration.	Continue to run	Check the Ethernet configuration of upper computer	When the application instruction is executed
5087	SOCRECV instruction reception timeout	Continue to run	1. Check if the network connection is normal. 2. Check whether there is data sent by the network peer.	When the application instruction is executed
5088	The SOCMTCP instruction specifies that the socket uses the configuration mode of a non-TCP client	Continue to run	Check the Ethernet configuration of upper computer	When the application instruction is executed
5089	When configuring the TCP server for the Ethernet socket, specify the local port as 502	Continue to run	Port 502 has been occupied internally, please modify the local port.	When the application instruction is executed
508A	The UDP port is specified as 1092	Continue to run	UDP port: 1092 is occupied by the system and cannot be used, please modify the local port number.	When the application instruction is executed
5090	Network cable connection exception	Continue to run	Check if the network cable is connected	When the application instruction is executed

14 Built-in FC/FB

14.1 Built-in FC function

14.1.1 ADD/Addition operation instruction



The devices or variables specified in s1 and s2 are added and output to d.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ADD	Input s1															•	•	•	•	•	•	•					
	Input s2															•	•	•	•	•	•	•					
	Output d															•	•	•	•	•	•	•					

Precautions for use of devices

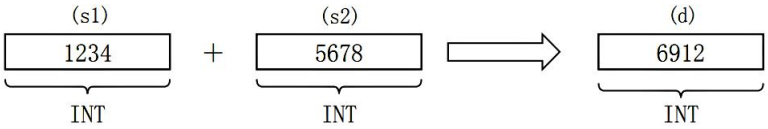
- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
ADD	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME

Function

- Perform additive operation of the data specified in (s1) and the data specified in (s2) and store the additive operation result in (d) and output the operation result from (d) based on the data type that is same to (s).
- When data type is INT type:



Precautions

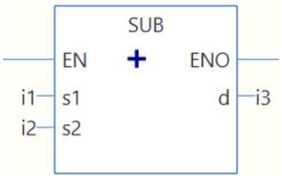
- It is best to ensure that the output data and the input data are of the same type, otherwise, it may cause high-bit data loss. If the length of the output result is larger than the length of (d), data truncation will be carried out, and the high-bit data will be discarded, with the data of (d) length being left;
- If the integer type overflows or underflows during operation, it will be counted cyclically;
- If the real type overflows or underflows during operation, the operation error will occur and the output value is indefinite;
- NAN: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞;

▪ INF: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. If the result is positive, the output is < INF; if the result is negative, the output is-INF;

When the calculated exponent (e) and mantissa (f) are both 0, the sign becomes ON. When the absolute value of the result is less than the minimum value that floating-point data can express, the calculation result also outputs 0, under which circumstance, the underflow sign becomes ON.

▪ Addition rules between TIME types: TIME + TIME = TIME, TOD + TIME = TOD, DT + TIME = DT, DATE + TIME = DATE. Except for DATE and DT types, which are accurate to seconds (s), other TIME and TOD types are accurate to milliseconds (ms). Therefore, when adding data types with different accuracy, it is necessary to convert milliseconds (ms) into seconds (s), and then carry out addition operation. Then check the accuracy of the output type to see if it is necessary to convert units for output.

14.1.2 SUB/Subtraction operation instruction



Perform subtraction operation of the data specified in (s1) and the data specified in (s2) and store the operation result in (d).

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SUB	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d															●	●	●	●	●	●	●					

Precautions for use of devices

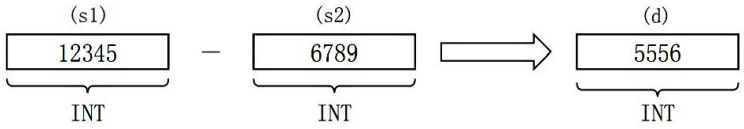
- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
SUB	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME

Function

- Perform subtraction operation of the data specified in (s1) and the data specified in (s2) and store the operation result in (d). Output the operation result from (d) based on the data type that is same to (s).
- When data type is INT type:



Precautions

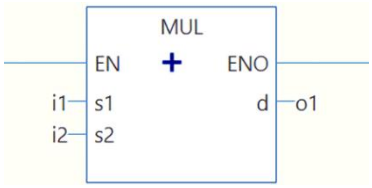
- It is best to ensure that the output data and the input data are of the same type, otherwise, it may cause high-bit data loss. If the length of the output result is larger than the length of (d), data truncation will be carried out, and the high-bit data will be discarded, with the data of (d) length being left;
- If the integer type overflows or underflows during operation, it will be counted cyclically;
- If the real type overflows or underflows during operation, the operation error will occur and the output value is indefinite;
- nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞;
- inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. If the result is positive, the output is < INF; if the result is negative, the output is-INF;
- Subtraction rules between TIME types: TIME-TIME = TIME, DATE-DATE = TIME, DATE-TIME = DATE, TOD-TIME = TOD, TOD-TOD = TIME, DT-TIME = DT, DT-DT = TIME. DATE and DT types are accurate to seconds (s), other TIME and TOD types are accurate to milliseconds (ms). It can be divided into three types:

① The input and output accuracy are all the same. For example: TIME-TIME = TIME, TOD-TIME = TOD, TOD-TOD = TIME, without the need of unit conversion; direct subtraction and direct output can be completed;

② The input accuracy is the same, but the output is different. For example: DATE-DATE = TIME, DT-DT = TIME. After this subtraction, it is necessary to judge the accuracy of the type. The data of DATE and DT types in seconds (s) need to be converted into milliseconds (ms) for output;

③ The input accuracy is different. For example: DT-TIME = DT and DATE-TIME = DATE. In case that the accuracy of two types of data that need to be subtracted is inconsistent, it is necessary to convert the milliseconds (ms) into seconds (s) for subtraction, and then judge the accuracy of the output type to see whether it needs to convert the units for output.

14.1.3 MUL/Multiplication operation instruction



Perform multiplication operation of the data specified in (s1) and the data specified in (s2) and store the operation result in (d).

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MUL	Input s1															•	•	•	•	•	•	•					
	Input s2															•	•	•	•	•	•	•					
	Output d															•	•	•	•	•	•	•					

Precautions for use of devices

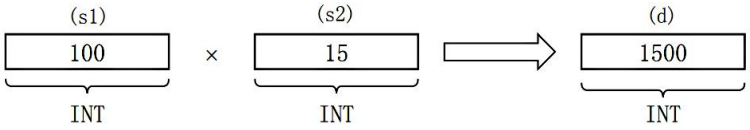
- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
MUL	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME

Function

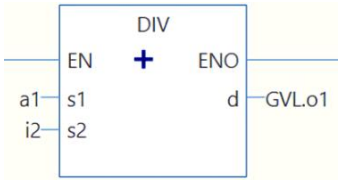
- Perform multiplication operation of the data specified in (s1) and the data specified in (s2) and store the operation result in (d).
- When data type is INT type:



Precautions

- Even if the operation result under INT type exceeds the data range of INT type, the data of INT type is also output. (The result of the operation is changed to DINT type, but the data of INT type in which the high-order 16 bits are deleted is output). Or DINT type is used to receive the output value;
- nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞;
- inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. It the result is positive, the output is + INF; if the result is negative, the output is - INF.

14.1.4 DIV/Division operation instruction



Perform division operation of the data specified in (s1) and the data specified in (s2) and store the operation result in (d).

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DIV	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d															●	●	●	●	●	●	●					

Precautions for use of devices

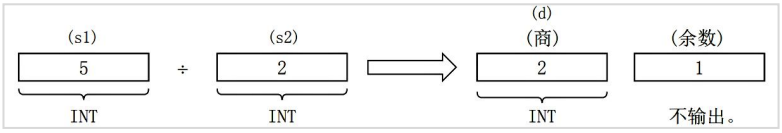
- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
DIV	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME

Function

- Perform division operation of the data specified in (s1) and the data specified in (s2) and output the operation result from (d) based on the data type that is same to (s).
- When data type is INT type:



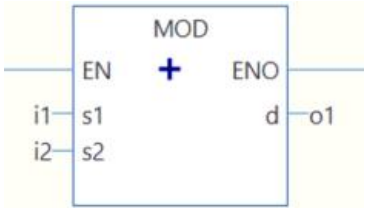
Precautions

- Only the quotient of integer can be output under the operation of INT/DINT type;
- Under the operation of REAL type, the operation result of REAL number will be output;
- nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞;
- inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. It the result is positive, the output is + INF; if the result is negative, the output is - INF.

Error code

Error code	Content
4080H	The divisor (s2) is input as 0.

14.1.5 MOD/Remainder operation instruction



Output the remainder of the input value ((s1) (s2)).

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MOD	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d															●	●	●	●	●	●	●					

Precautions for use of devices

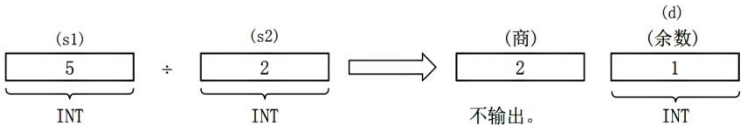
- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
MOD	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT

Function

- Perform the division operation ((s1) (s2)) of the data of INT type/DINT type input in (s1) and (s2) and output the remainder of the operation result from (d) based on the data type that is same to (s).



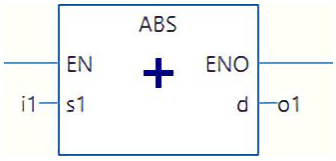
Precautions

- nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞;
- inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. It the result is positive, the output is + INF; if the result is negative, the output is - INF.

Error code

Error code	Content
4080H	The divisor (s2) is input as 0.

14.1.6 ABS/Absolute value (Integer)



Calculate the absolute value of s1 and output it to d.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ABS	Input s1															●	●	●	●	●	●	●					
	Output d															●	●	●	●	●	●	●					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
ABS	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT

Function

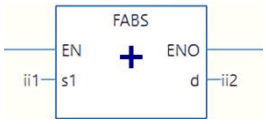
- Calculate the absolute value of s1 and output it to d.

  d = |s1|

Precautions

- When the data type of (s1) is INT and -32768 is input, -32768 is output from (d);
- When the data type of (s1) is DINT and -2147483648 is input, -2147483648 is output from (d);

14.1.7 ABS/Absolute Value (Real)



Calculate the absolute value of s1 and output it to d.

Unsupported device

Variable Type Used

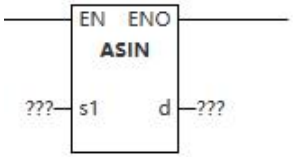
Instruction	Parameter	Variable type
FABS	Input s1	LREAL
	Output d	LREAL

Function

- Calculate the absolute value of s1 and output it to d.

  d = |s1|

14.1.8 ASIN/SIN^{-1} Operation



After calculating the SIN^{-1} (arc sine) value of the angle specified in (s1), the calculation result is stored in (d).

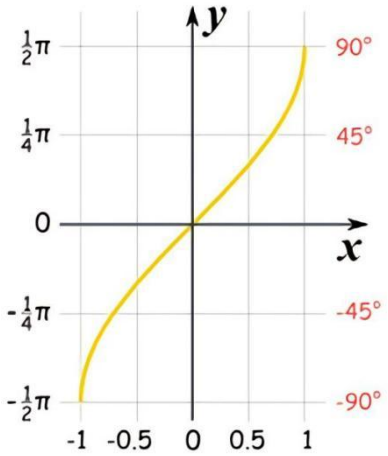
Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
ASIN	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

Function

- After calculating the SIN^{-1} (arc sine) value of the angle specified in (s1), the calculation result is stored in the device number specified in (d).
- The SIN^{-1} value specified in (s1) can be set within the range of -1.0~1.0.
- The angle (calculation result) stored in (d) is stored in the unit of radians (-π/2)~(π/2).

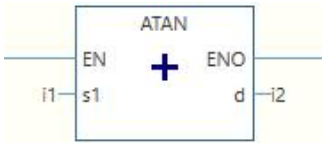


ASIN graph

Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

14.1.9 ATAN/TAN^{-1} Operation



After calculating the TAN^{-1} (arc tangent) value of the angle specified in (s1), the calculation result is stored in (d).

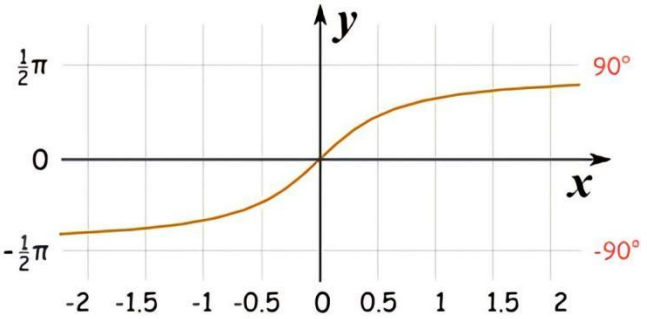
Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
ATAN	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

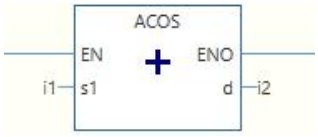
Function

- Calculate the TAN^{-1} ((arc tangent) value of the angle specified in (s1), and store the calculation result in (d).
- The TAN^{-1} value specified in (s1) can be set within the range of 1.17549 -38~3.40282+38.
- The angle (calculation result) stored in (d) is stored in the unit of radians (-π/2)~(π/2).



ATAN graph

14.1.10 ACOS/COS^{-1} Operation



After calculating the COS^{-1} (arc cosine) value of the angle specified in (s1), the calculation result is stored in (d).

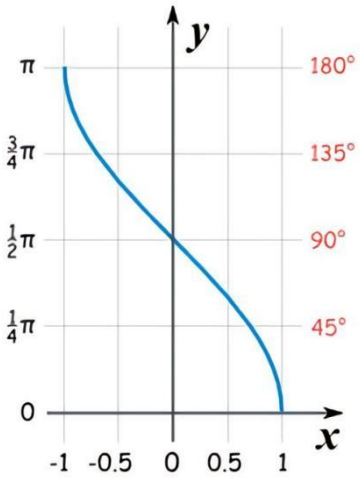
Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
ACOS	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

Function

- After calculating the COS^{-1} (arc cosine) value of the angle specified in (s), the calculation result is stored in the device number specified in (d);
- The COS value specified in (s) can be set within the range of -1.0~1.0.
- The angle (calculation result) stored in (d) stores the value from 0~π in radians.

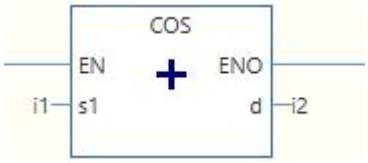


ACOS graph

Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

14.1.11 COS/COS Operation



After calculating the SIN^{-1} (cosine) value of the angle specified in (s1), the calculation result is stored in the device number specified in (d).

Unsupported device

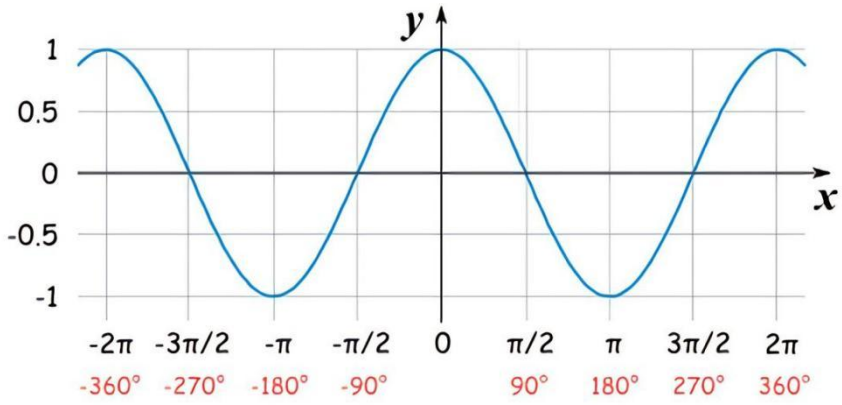
Variable Type Used

Instruction	Parameter	Variable type
COS	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

Function

After calculating the COS (cosine) value of the angle specified in (s1), store the calculation result in the device number specified in (d).

For the angle specified in (d), set it in radians (angle×π÷180).



COS graph

14.1.12 SIN/SIN Operation



After calculating the SIN (sine) value of the angle specified in (s1), the calculation result is stored in the device number specified in (d).

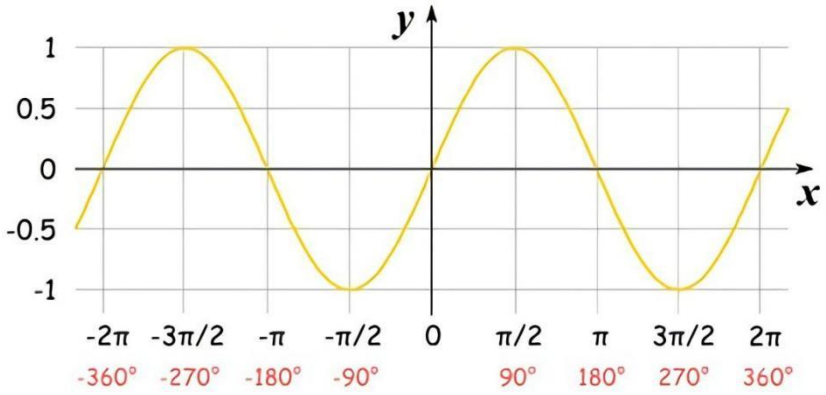
Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
SIN	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

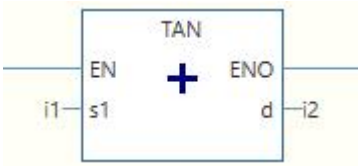
Function

- After calculating the SIN (sine) value of the angle specified in (s1), the calculation result is stored in the device number specified in (d);
- For the angle specified in (s), set it in radians (angle×π÷180).



SIN graph

14.1.13 TAN/TAN Operation



After calculating the TAN (tangent) value of the angle specified in (s1), the calculation result is stored in the device number specified in (d).

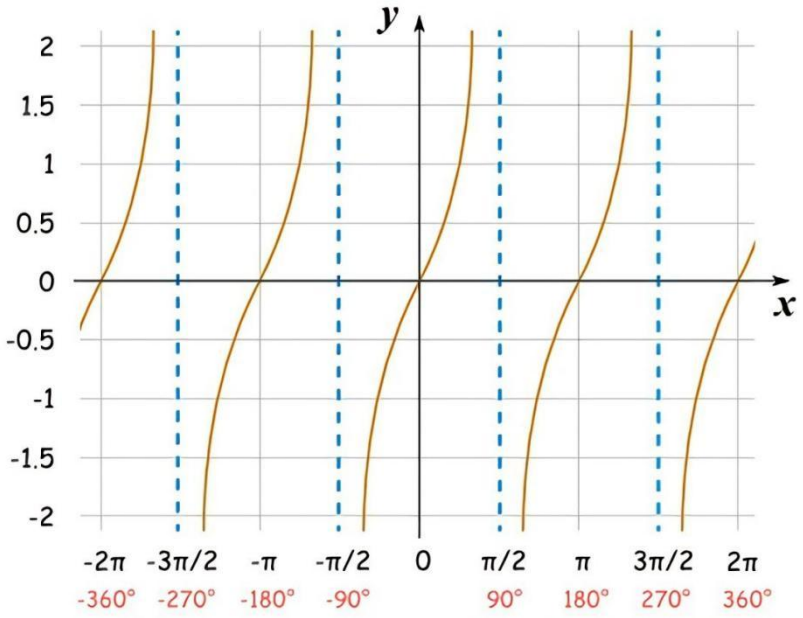
Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
SIN	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

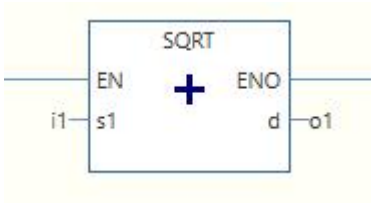
Function

- After calculating the TAN (tangent) value of the angle specified in (s1), the calculation result is stored in the device number specified in (d).
- For the angle specified in (s1), set it in radians (angle×π÷180).



TAN graph

14.1.14 SQRT/Square root operation



Perform square root operation on (s1).

Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
SQRT	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

Function

- Output the square root of the REAL data input in (s1) from (d);
- If the input value is set to A and the operation output value is set to B, the situation is as follows:

$$B = \sqrt{A}$$

- The input value of (s1) is a data value of REAL or long LREAL type and is within the range of positive numbers.

Precautions

- s1 cannot be negative; otherwise, the operation error may occur;
- nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞;
- inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. It the result is positive, the output is + INF; if the result is negative, the output is - INF.

Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

14.1.15 EXP/Exponential operation



After performing the exponential calculation of the value specified in (s1), the calculation result is stored in the device specified in (d).

Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
EXP	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

Function

▪ After performing the exponential calculation of the value specified in (s), the calculation result is stored in the device number specified in (d).

▪ In exponential calculation, the base (e) is calculated as “2.718282”.

▪ If the input value is set to A and the operation output value is set to B, the situation is as follows:

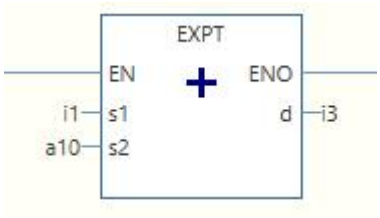
$B=e^{A}$

Precautions

▪ nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞.

▪ inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. It the result is positive, the output is + INF; if the result is negative, the output is - INF.

14.1.16 EXPT/Exponentiation



After performing the (s2) exponentiation of the value specified in (s1), the calculation result is stored in the device specified in (d).

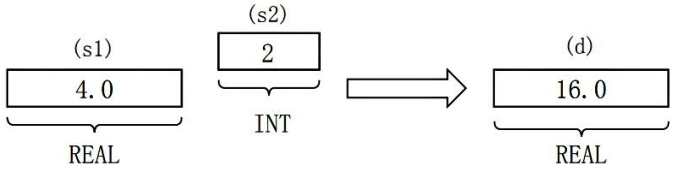
Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
EXPT	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

Function

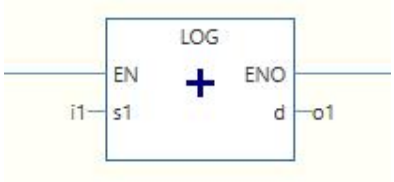
- After performing the exponentiation of the value specified in (s1), the calculation result is stored in the device specified in (d).



Precautions

- nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞.
- inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. It the result is positive, the output is + INF; if the result is negative, the output is - INF.

14.1.17 LOG/Logarithmic operation



After performing the logarithmic operation of the value specified in (s1) based on 10, the calculation result is stored in (d).

Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
LOG	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

Function

- After performing the logarithmic operation of the value specified in (s1) based on 10, the calculation result is stored in (d);
- If the input value is set to A and the operation output value is set to B, the situation is as follows:

  $B = \log 10^{A}$
- The input value of (s1) is a data value of REAL type;
- The value specified in (s1) can only be set to a positive number. (Cannot perform operations with negative numbers.)
- When the operation result is-0 or underflow occurs, the operation result is output as 0.

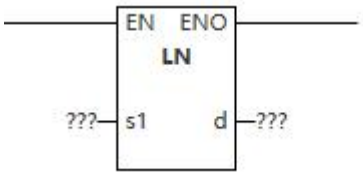
Precautions

- nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞;
- inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. It the result is positive, the output is + INF; if the result is negative, the output is - INF.

Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

14.1.18 LN/Natural logarithm operation



After performing the logarithmic operation of the value specified in (s1) based on e, the calculation result is stored in (d).

Unsupported device

Variable Type Used

Instruction	Parameter	Variable type
LN	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL
	Output d	LREAL

Function

- After performing the logarithmic operation of the value specified in (s1) based on e, the calculation result is stored in (d);
- If the input value is set to A and the operation output value is set to B, the situation is as follows:

  $B = \log e^{A}$
- In natural logarithm operation, the base (e) is calculated as “2.71828”.

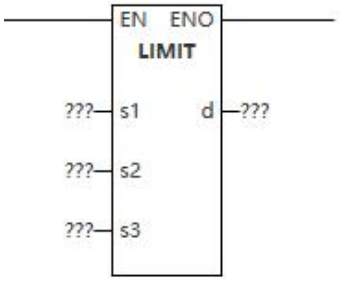
Precautions

- nan: not a number, which means “non-numerical value”, which is generated when the calculation result is not equal to a number or infinite, such as 0.0/0.0, ∞/∞, or ∞-∞;
- inf: infinite, which means “infinity”. When the absolute value of the result is greater than the maximum value that floating-point data can express, it is overflow, and the result outputs ± INF. It the result is positive, the output is + INF; if the result is negative, the output is - INF.

Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

14.1.19 LIMIT/Upper and lower limit control



Judge whether the input data is between the minimum value and the maximum value. If the input data is between the minimum value and the maximum value, the input data is directly output as output data. If the input data is greater than the maximum value, the maximum value is taken as the output value. If the input data is less than the minimum value, the minimum value is taken as the output value.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LIMIT	Input s1															•	•	•	•	•	•	•					
	Input s2															•	•	•	•	•	•	•					
	Input s3															•	•	•	•	•	•	•					
	Output d															•	•	•	•	•	•	•					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Parameter	Variable type
LIMIT	Input s1	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Input s2	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Input s3	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Output d	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME

Function

- Judge whether the input data is between the minimum value and the maximum value. If the input data is between the minimum value and the maximum value, the input data is directly output as output data. If the input data is greater than the maximum value, the maximum value is taken as the output value. If the input data is less than the minimum value, the minimum value is taken as the output value;
- (s1) is a minimum value, (s3) is a maximum value, and (s2) is a comparison value;
- When the input value of (s2) is greater than the input value of (s3), output the input value of (s3) from (d);
- When the input value of (s2) is less than the input value of (s1), output the input value of (s1) from (d);
- When the input value of (s1) is less than the input value of (s2) and is less than the input value of (s3), output the input value of (s2)

from (d).

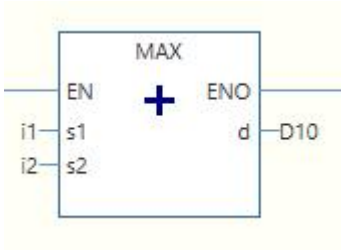
Precautions

- The types of input and output should be same; otherwise, data loss may occur;
- The value of (s3) needs to be greater than the value of (s1), otherwise the operation error will occur and d1 will not output the value.

Error code

Error code	Content
4084H	Data beyond the specifiable range was entered.

14.1.20 MAX/Maximum value



The maximum value of two input data is taken as the output value.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MAX	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d															●	●	●	●	●	●	●					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
MAX	Input s1	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Input s2	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Output d	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME

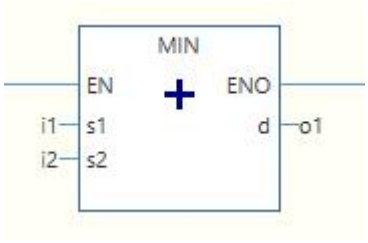
Function

- The maximum value of two input data is taken as the output value;
- Output and input are of the same type.

Precautions

- Please select the data of same type for comparison.

14.1.21 MIN/Minimum value



The minimum value of two input data is taken as the output value.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MIN	Input s1															•	•	•	•	•	•	•					
	Input s2															•	•	•	•	•	•	•					
	Output d															•	•	•	•	•	•	•					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Parameter	Variable type
MIN	Input s1	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Input s2	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Output d	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME

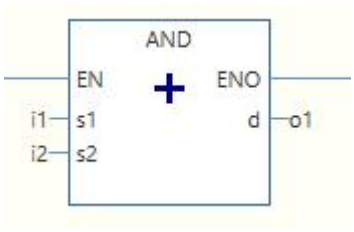
Function

- The minimum value of two input data is taken as the output value;
- Output and input are of the same type.

Precautions

- Please select the data of same type for comparison.

14.1.22 AND/Logical AND



AND operation of variables or constants.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
AND	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d															●	●	●	●	●	●	●					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

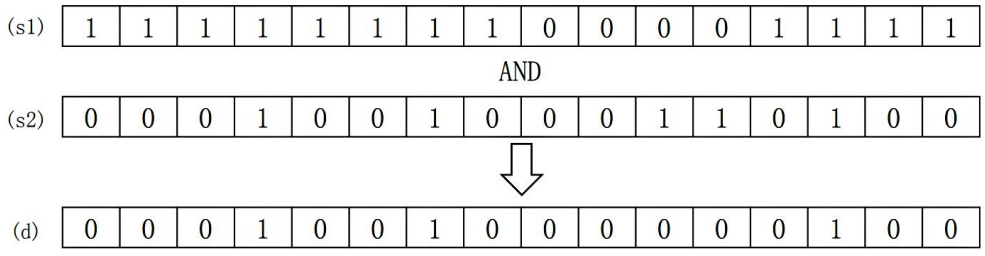
Variable Type Used

Instruction	Parameter	Variable type
AND	Input s1	BOOL, BYTE, WORD, DWORD and LWORD
	Input s2	BOOL, BYTE, WORD, DWORD and LWORD
	Output d	BOOL, BYTE, WORD, DWORD and LWORD

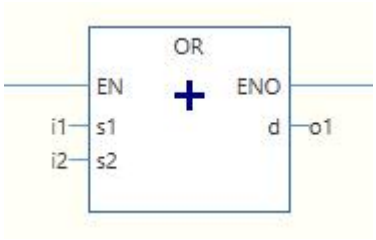
Function

- Perform logic product operation of data of BOOL type/WORD type/DWORD type input in (s1)(s2) based on different bits and output the operation result from (d) based on the data type that is same to (s).

When data type is WORD:



14.1.23 OR/Logical OR



OR operation of variables or constants.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
OR	Input s1															•	•	•	•	•	•	•					
	Input s2															•	•	•	•	•	•	•					
	Output d															•	•	•	•	•	•	•					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

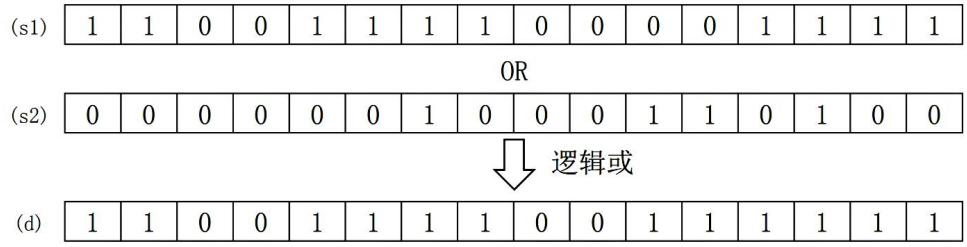
Variable Type Used

Instruction	Parameter	Variable type
OR	Input s1	BOOL, BYTE , WORD, DWORD and LWORD
	Input s2	BOOL, BYTE , WORD, DWORD and LWORD
	Output d	BOOL, BYTE , WORD, DWORD and LWORD

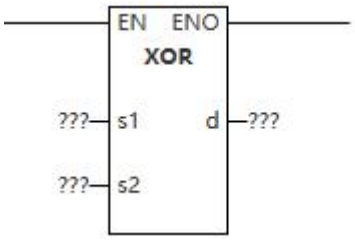
Function

- Perform logical OR operation of data of BOOL type/WORD type/DWORD type input in (s1)(s2) based on different bits and output the operation result from (d) based on the data type that is same to (s).

When data type is WORD:



14.1.24 XOR/XOR



XOR operation of variables or constants.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
XOR	Input s1															•	•	•	•	•	•	•					
	Input s2															•	•	•	•	•	•	•					
	Output d															•	•	•	•	•	•	•					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

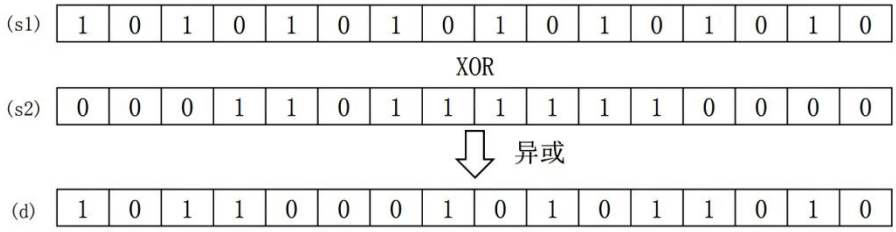
Variable Type Used

Instruction	Parameter	Variable type
XOR	Input s1	BOOL, BYTE, WORD, DWORD and LWORD
	Input s2	BOOL, BYTE, WORD, DWORD and LWORD
	Output d	BOOL, BYTE, WORD, DWORD and LWORD

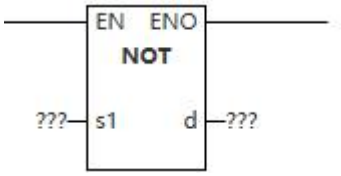
Function

- Perform XOR operation of data of BOOL type/WORD type/DWORD type input in (s1)(s2) based on different bits and output the operation result from (d) based on the data type that is same to (s).

When data type is WORD:



14.1.25 NOT/NOT



NOT operation of variables or constants. Perform NOT operation bit by bit.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
NOT	Input s1															●	●	●	●	●	●	●					
	Output d															●	●	●	●	●	●	●					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
NOT	Input s1	BOOL, BYTE , WORD, DWORD and LWORD
	Output d	BOOL, BYTE , WORD, DWORD and LWORD

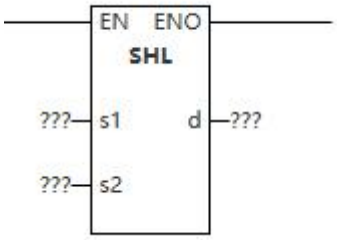
Function

- Perform logical NOT operation of data of BOOL type/WORD type/DWORD type input in (s1)(s2) based on different bits and output the operation result from (d) based on the data type that is same to (s).

When data type is WORD:

(s)	0	1	1	0	1	0	1	1	0	0	0	0	1	1	1	1
	NOT															
(d)	1	0	0	1	0	1	0	0	1	1	1	1	0	0	0	0

14.1.26 SHL/Shift left instruction



The operand is shifted to the left by bit, and the bits moved out on the left are not processed, and the right is automatically filled with 0.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SHL	Input s1																									
	Input s2															●	●	●	●	●						
	Output d																									

Precautions for use of devices

- T, C, D, R and SD are signed 16-bit data types;

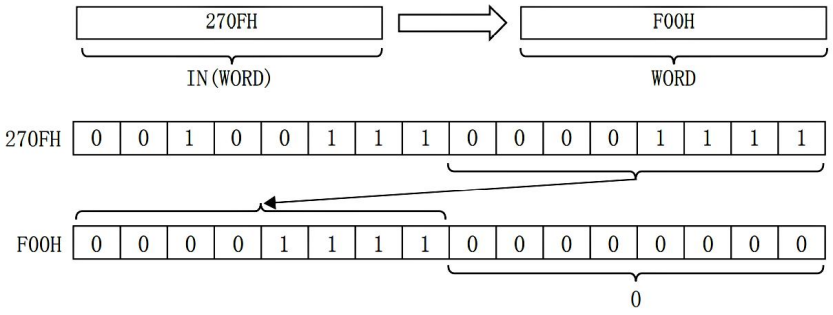
Variable Type Used

Instruction	Parameter	Variable type
SHL	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT
	Input s2	INT
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT

Function

- Shift the 16-bit data/32-bit data (WORD type/DWORD type) input in (s1) to left for (s2) bit and output the result from (d) based on the data type that is same to (s1).
- The (s2) bit from the lowest bit will become 0;
- The input value of (s1) is the 16-bit data/32-bit data (WORD type/DWORD type).

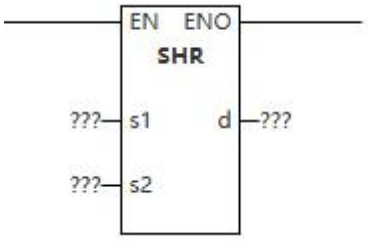
When the data of (s1) is the 16-bit data (WORD type), and the input value of (s2) is 8:



Precautions

- The input value to (n) (specified by the number of carry digits) is INT data value and is within the following range:
  ① When the input is 16-bit data (s2), the range is 0~15. The remainder operation will be performed of the excess part;
  ② When the input is 32-bit data (s2), the range is 0~31. The remainder operation will be performed of the excess part.
- When the value s2 of the displacement<0, it can be converted to s2 = s2 + output data type bits. For example, when the input and output type is a 16-bit int type, shifting s2 =-3 to the right, which is equivalent to shifting s2 = 16 + (-3) = 13.

14.1.27 SHR/Shift right instruction



The operand is shifted to the right by bit, and the bits moved out on the right are not processed, and the left is automatically filled with 0.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SHR	Input s1																										
	Input s2																●	●	●	●		●					
	Output d																										

Precautions for use of devices

- T, C, D, R and SD are signed 16-bit data types;

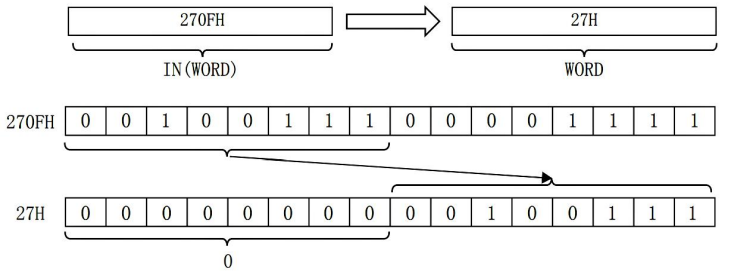
Variable Type Used

Instruction	Parameter	Variable type
SHR	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT
	Input s2	INT
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT

Function

- Shift the 16-bit data/32-bit data (WORD type/DWORD type) input in (s1) to the right for (s2) bit and output the result from (d) based on the data type that is same to (s1);
- The number of bits (n) shifted to the right is the value of (s2);
- The (s2) bit from the highest bit will become 0;
- The input value to (s1) is the 16-bit data/32-bit data (WORD type/DWORD type).

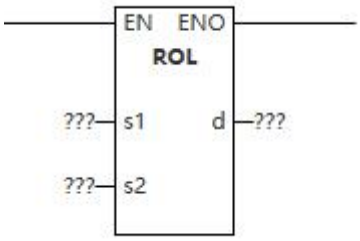
When the data of (s1) is the 16-bit data (WORD type), and the input value of (s2) is 8:



Precautions

- The input value to (s2) (specified by the number of carry digits) is INT data value and is within the following range:
  ① When the input is 16-bit data (s2), the range is 0~15. The remainder operation will be performed of the excess part;
  ② When the input is 32-bit data (s2), the range is 0~31. The remainder operation will be performed of the excess part.
- When the value s2 of the displacement<0, it can be converted to s2 = s2 + output data type bits. For example, when the input and output type is a 16-bit int type, shifting s2 =-3 to the right, which is equivalent to shifting s2 = 16 + (-3) = 13.

14.1.28 ROL/Rotate left



The operand is rotated to the left by bit, and the bits moved out on the left are directly supplemented to the lowest bits on the right.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ROL	Input s1																										
	Input s2																●	●	●	●							
	Output d																										

Precautions for use of devices

- T, C, D, R and SD are signed 16-bit data types

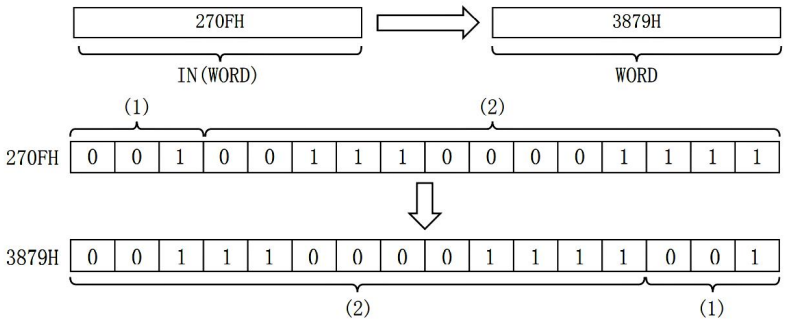
Variable Type Used

Instruction	Parameter	Variable type
ROL	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT
	Input s2	INT
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT

Function

- Rotate the 16-bit data/32-bit data (WORD type/DWORD type) input in (s1) to the left for (s2) bit and output the result from (d) based on the data type that is same to (s1);
- The number of bits (s2) rotated to the left is the value input to (n);
- The input value to (s1) is the 16-bit data/32-bit data (WORD type/DWORD type).

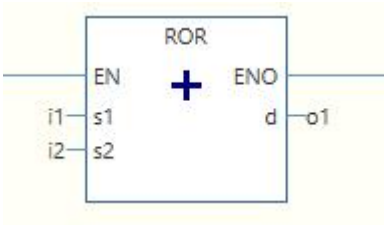
When the data of (s1) is the 16-bit data (WORD type), and the input value of (s2) is 3: (Rotate to left for 3 bits.)



Precautions

- The input value to (s2) (specified by the number of carry digits) is INT data value and is within the following range:
  ① 1. When the input is 16-bit data (s2), the range is 0~15. The remainder operation will be performed of the excess part;
  ② 2. When the input is 32-bit data (s2), the range is 0~31. The remainder operation will be performed of the excess part.
- When the value s2 of the displacement<0, it can be converted to s2 = s2 + output data type bits. For example, when the input and output type is a 16-bit int type, shifting s2 =-3 to the right, which is equivalent to shifting s2 = 16 + (-3) = 13.

14.1.29 ROR/Rotate right



The operand is rotated to the right by bit, and the bits moved out on the right are directly supplemented to the highest bits on the left.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ROR	Input s1																										
	Input s2															•	•	•	•	•							
	Output d																										

Precautions for use of devices

- T, C, D, R and SD are signed 16-bit data types;

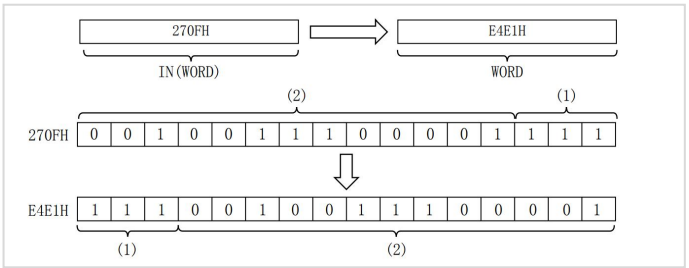
Variable Type Used

Instruction	Parameter	Variable type
ROR	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT
	Input s2	INT
	Output d	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT

Function

- Rotate the 16-bit data/32-bit data (WORD type/DWORD type) input in (s1) to the right for (s2) bit and output the result from (d) based on the data type that is same to (s1);
- The number of bits (s2) rotated to the right is the value input to (s2);
- The input value to (s1) is the 16-bit data/32-bit data (WORD type/DWORD type).

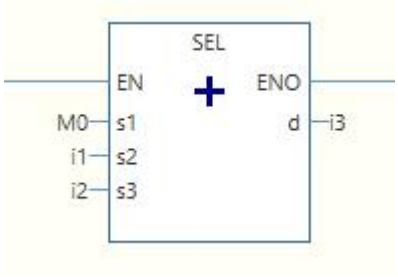
When the data of (s1) is the 16-bit data (WORD type), and the input value to (s2) is 3: (Rotate to right for 3 bits.)



Precautions

- The input value to (s2) (specified by the number of carry digits) is INT data value and is within the following range:
  ① When the input is 16-bit data (s2), the range is 0~15. The remainder operation will be performed of the excess part;
  ② When the input is 32-bit data (s2), the range is 0~31. The remainder operation will be performed of the excess part.
- When the value s2 of the displacement<0, it can be converted to s2 = s2 + output data type bits. For example, when the input and output type is a 16-bit int type, shifting s2 =-3 to the right, which is equivalent to shifting s2 = 16 + (-3) = 13.

14.1.30 SEL/Selection



Select one of the two input data via the selection switch as the output; when the selection switch is FALSE, the output value is the first input data; when the selection switch is TRUE, the output value is the second input data.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SEL	Input s1	●	●	●	●	●																				
	Input s2															●	●	●	●	●	●	●				
	Input s3															●	●	●	●	●	●	●				
	Output d															●	●	●	●	●	●	●				

Precautions for use of devices

- X, Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types.

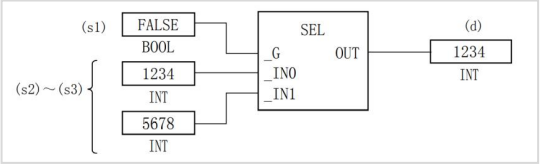
Variable Type Used

Instruction	Parameter	Variable type
SEL	Input s1	BOOL
	Input s2	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Input s3	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Output d	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME

Function

- According to the input value that is input to (s1), output the value that is input to (s2) and (s3) from (d) based on the data type that is same to (s2) and (s3);
- When the input value of (s1) is FALSE(=0), output the input value of (s2) from (d);
- When the input value of (s1) is TRUE(=1), output the input value of (s3) from (d).

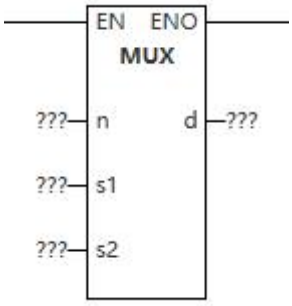
When data type of (s2) and (s3) is INT type ((s2) and (s3) of the independent variable name correspond to the bit value (0 or 1) of (s1).)



Precautions

- If data (d) varies from s2 and s3, it may result in data loss.

14.1.31 MUX/Multiplexer



According to the input value that is input to (n), output the value that is input to (s1)~ (s28) from (d) based on the data type that is same to (s).

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MUX	Input n															●	●	●	●	●							
	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Input s2	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d		●	●	●	●										●	●	●	●	●	●	●					

Precautions for use of devices

- X, Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types.

Variable Type Used

Instruction	Parameter	Variable type
MUX	Input n	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT and LINT
	Input s1	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Input s2	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Output d	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME

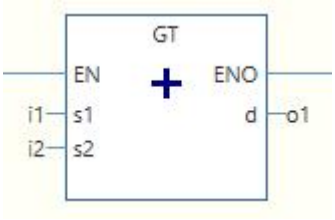
Function

- According to the input value that is input to (n), output the value that is input to (s1)~ (s28) from (d) based on the data type that is same to (s).
- n=0, no output;
- n≠0, output s(n).

Precautions

- If data (d) varies from input data, it may result in data loss.

14.1.32 GT/Greater than



Judge the size of two operands, when the first number is greater than the second number, output is TRUE, otherwise the output is FALSE.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
GT	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d		●	●	●	●																					

Precautions for use of devices

- Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types.

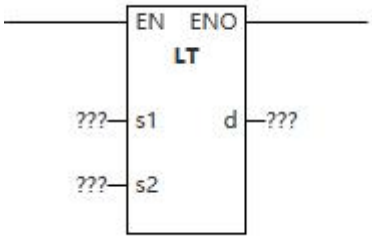
Variable Type Used

Instruction	Parameter	Variable type
GT	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Output d	BOOL

Function

- Judge the size of two operands, when the first number is greater than the second number, output is TRUE, otherwise the output is FALSE.

14.1.33 LT/Less than



Judge the size of two operands, when the first number is less than the second number, output is TRUE, otherwise the output is FALSE.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LT	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d		●	●	●	●																					

Precautions for use of devices

- Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types.

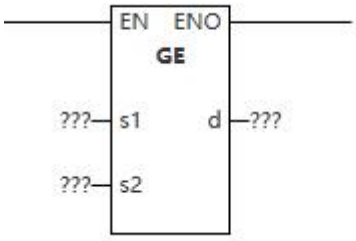
Variable Type Used

Instruction	Parameter	Variable type
LT	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Output d	BOOL

Function

- Judge the size of two operands, when the first number is less than the second number, output is TRUE, otherwise the output is FALSE.

14.1.34 GE/Greater than or equal to



Judge the size of two operands, when the first number is greater than or equal to the second number, output is TRUE, otherwise the output is FALSE.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
GE	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d		●	●	●	●																					

Precautions for use of devices

- Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types.

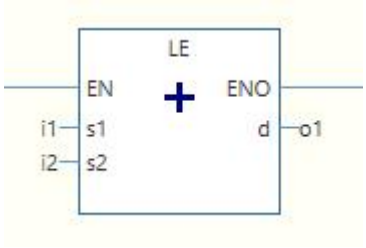
Variable Type Used

Instruction	Parameter	Variable type
GE	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Output d	BOOL

Function

- Judge the size of two operands, when the first number is greater than or equal to the second number, output is TRUE, otherwise the output is FALSE.

14.1.35 LE/Less than or equal to



Judge the size of two operands, when the first number is less than or equal to the second number, output is TRUE, otherwise the output is FALSE.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LE	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d		●	●	●	●																					

Precautions for use of devices

- Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types.

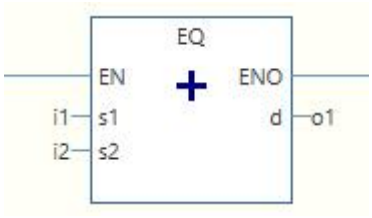
Variable Type Used

Instruction	Parameter	Variable type
LE	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Output d	BOOL

Function

- Judge the size of two operands, when the first number is less than or equal to the second number, output is TRUE, otherwise the output is FALSE.

14.1.36 EQ/Comparison (Equivalent)



Judge the size of two operands, when the first number is equivalent to the second number, output is TRUE, otherwise the output is FALSE.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
EQ	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d		●	●	●	●																					

Precautions for use of devices

- Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types.

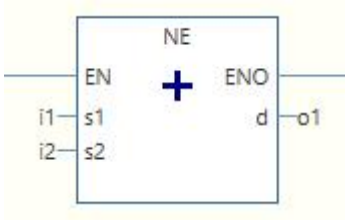
Variable Type Used

Instruction	Parameter	Variable type
EQ	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Output d	BOOL

Function

- Judge the size of two operands, when the first number is equivalent to the second number, output is TRUE, otherwise the output is FALSE.

14.1.37 NE/Comparison (not Equivalent)



Judge the size of two operands, when the first number is not equivalent to the second number, output is TRUE, otherwise the output is FALSE.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
NE	Input s1															●	●	●	●	●	●	●					
	Input s2															●	●	●	●	●	●	●					
	Output d		●	●	●	●																					

Precautions for use of devices

- Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types.

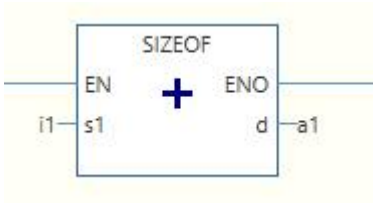
Variable Type Used

Instruction	Parameter	Variable type
NE	Input s1	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME and LTIME
	Output d	BOOL

Function

- Judge the size of two operands, when the first number is not equivalent to the second number, output is TRUE, otherwise the output is FALSE.

14.1.38 SIZEOF/Get the size



Gets the size of the specified variable (s1).

Unsupported device

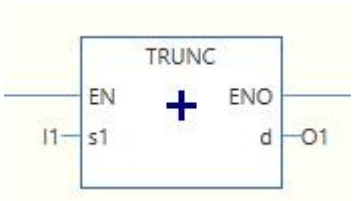
Variable Type Used

Instruction	Parameter	Variable type
SIZEOF	Input s1	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Output d	WORD

Function

- Gets the size of the specified variable (s1).

14.1.39 TRUNC/Truncated rounding instruction



Rounding.

Unsupported device

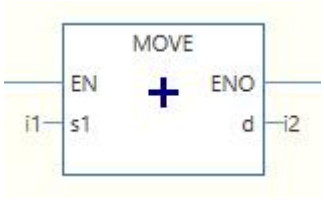
Variable Type Used

Instruction	Parameter	Variable type
TRUNC	Input s1	LREAL
	Output d	INT、DINT

Function

- Round the value of s1 and truncate the data after the decimal place.

14.1.40 MOVE/Move instruction



Move the value of the variable specified in (s1) into the variable specified in (d).

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MOVE	Input s1															●	●	●	●	●	●	●					
	Output d															●	●	●	●	●	●	●					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type
MOVE	Input s1	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Output d	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME, TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME

Function

- Move the value of the variable specified in (s1) into the variable specified in (d);
- For (s1) and (d), BOOL type/INT type/DINT type/WORD type/DWORD type/LWORD type/REAL type/TIME type can be specified. In addition, (s1) and

(d) can only designate the same data type.

14.1.41 BOOL_TO_<TYPE>/BOOL type conversion instruction

BOOL_TO_<TYPE>

Function

- Convert the BOOL data type to other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BOOL_TO_BYTE	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_DATE	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_DINT	Input s1	●	●	●	●	●																				
	Output d																				●	●				
BOOL_TO_DT	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_DWORD	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_INT	Input s1	●	●	●	●	●																				
	Output d															●	●	●	●	●						
BOOL_TO_REAL	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_SINT	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_TIME	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_TOD	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_UDINT	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_UINT	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_USINT	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_WORD	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_LREAL	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_LWORD	Input s1	●	●	●	●	●																				
	Output d																									
BOOL_TO_LINT	Input s1	●	●	●	●	●																				
	Output d																									

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BOOL_TO_LTIME	Input s1	●	●	●	●	●																				
	Output d																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
BOOL_TO_BYTE	Input s1	BOOL	BOOL_TO_TOD	Input s1	BOOL
	Output d	BYTE		Output d	TIME_OF_DAY
BOOL_TO_DATE	Input s1	BOOL	BOOL_TO_UDINT	Input s1	BOOL
	Output d	DATE		Output d	UDINT
BOOL_TO_DINT	Input s1	BOOL	BOOL_TO_UINT	Input s1	BOOL
	Output d	DINT		Output d	UINT
BOOL_TO_DT	Input s1	BOOL	BOOL_TO_USINT	Input s1	BOOL
	Output d	DATE_AND_TIME		Output d	USINT
BOOL_TO_DWORD	Input s1	BOOL	BOOL_TO_WORD	Input s1	BOOL
	Output d	DWORD		Output d	WORD
BOOL_TO_INT	Input s1	BOOL	BOOL_TO_LREAL	Input s1	BOOL
	Output d	INT		Output d	LREAL
BOOL_TO_REAL	Input s1	BOOL	BOOL_TO_LWORD	Input s1	BOOL
	Output d	REAL		Output d	LWORD
BOOL_TO_SINT	Input s1	BOOL	BOOL_TO_LINT	Input s1	BOOL
	Output d	SINT		Output d	LINT
BOOL_TO_TIME	Input s1	BOOL	BOOL_TO_LTIME	Input s1	BOOL
	Output d	TIME		Output d	LTIME
BOOL_TO_STRING	Input s1	BOOL			
	Output d	STRING			

InstructionsSTRING

- If BOOL is equal to FALSE, the output value is 0; if BOOL is TRUE, the output value is 1;
- When BOOL turns to DATE, DATE is timed in seconds at the bottom layer. Therefore, no matter what the input value of BOOL is, it is displayed as 1970-01-01;
- When BOOL turns to DATE, DATE_AND_TIME is timed in seconds at the bottom layer. When BOOL is TRUE, the output is 1970-01-01-0: 0:1.

14.1.42 BYTE_TO_<TYPE>/BYTE type conversion instruction

BYTE_TO_<TYPE>

Function

- Convert BYTE data type to other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
BYTE_TO_BOOL	Input s1																										
	Output d		●	●	●	●																					
BYTE_TO_DINT	Input s1																										
	Output d																				●	●					
BYTE_TO_INT	Input s1																										
	Output d															●	●	●	●	●							

The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
BYTE_TO_BOOL	Input s1	BYTE	BYTE_TO_TOD	Input s1	BYTE
	Output d	BOOL		Output d	TIME_OF_DAY
BYTE_TO_DATE	Input s1	BYTE	BYTE_TO_UDINT	Input s1	BYTE
	Output d	DATE		Output d	UDINT
BYTE_TO_DINT	Input s1	BYTE	BYTE_TO_UINT	Input s1	BYTE
	Output d	DINT		Output d	UINT
BYTE_TO_DT	Input s1	BYTE	BYTE_TO_USINT	Input s1	BYTE
	Output d	DATE_AND_TIME		Output d	USINT
BYTE_TO_DWORD	Input s1	BYTE	BYTE_TO_WORD	Input s1	BYTE
	Output d	DWORD		Output d	WORD
BYTE_TO_INT	Input s1	BYTE	BYTE_TO_LREAL	Input s1	BYTE
	Output d	INT		Output d	LREAL
BYTE_TO_REAL	Input s1	BYTE	BYTE_TO_LWORD	Input s1	BYTE
	Output d	REAL		Output d	LWORD
BYTE_TO_SINT	Input s1	BYTE	BYTE_TO_LINT	Input s1	BYTE
	Output d	SINT		Output d	LINT
BYTE_TO_TIME	Input s1	BYTE	BYTE_TO_LTIME	Input s1	BYTE
	Output d	TIME		Output d	LTIME
BYTE_TO_STRING	Input s1	BYTE			
	Output d	STRING			

Instructions

- When BYTE is converted to BOOL, if the input value is 0, the output value is FALSE; otherwise, if the input value is not 0, the output value is TRUE;

- When BYTE is converted to DATE, DATE is timed in seconds at the bottom layer. Therefore, no matter what the input value of BYTE is, it is displayed as 1970-01-01;
- When BYTE is converted to DT, DATE_AND_TIME is timed in seconds at the bottom layer. When the BYTE input is 61, the output is 1970-01-01-0: 1: 1.

14.1.43 DATE_TO_<TYPE>/DATE type conversion instruction

DATE_TO_<TYPE>

Function

- The date is saved as a DWORD internally (the seconds since January 1, 1970);
- Convert DATE data type to other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DATE_TO_BOOL	Input s1																										
	Output d		•	•	•	•																					
DATE_TO_DINT	Input s1																										
	Output d																				•	•					
DATE_TO_INT	Input s1																										
	Output d															•	•	•	•	•							

The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
DATE_TO_BYTE	Input s1	DATE	DATE_TO_TOD	Input s1	DATE
	Output d	BYTE		Output d	TIME_OF_DAY
DATE_TO_BOOL	Input s1	DATE	DATE_TO_UDINT	Input s1	DATE
	Output d	BOOL		Output d	UDINT
DATE_TO_DINT	Input s1	DATE	DATE_TO_UINT	Input s1	DATE
	Output d	DINT		Output d	UINT
DATE_TO_DT	Input s1	DATE	DATE_TO_USINT	Input s1	DATE
	Output d	DATE_AND_TIME		Output d	USINT
DATE_TO_DWORD	Input s1	DATE	DATE_TO_WORD	Input s1	DATE
	Output d	DWORD		Output d	WORD
DATE_TO_INT	Input s1	DATE	DATE_TO_LREAL	Input s1	DATE
	Output d	INT		Output d	LREAL
DATE_TO_REAL	Input s1	DATE	DATE_TO_LWORD	Input s1	DATE
	Output d	REAL		Output d	LWORD
DATE_TO_SINT	Input s1	DATE	DATE_TO_LINT	Input s1	DATE
	Output d	SINT		Output d	LINT
DATE_TO_TIME	Input s1	DATE	DATE_TO_LTIME	Input s1	DATE
	Output d	TIME		Output d	LTIME
DATE_TO_STRING	Input s1	DATE			

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
	Output d	STRING			

Instructions

- When DATE is converted to BOOL,. DATE_TO_BOOL(D#1970-01-01), the output value is FALSE;
- When DATE is converted to INT, DATE_TO_INT(D#1970-01-15), the output value is 29952;
- When DATE is converted to TIME, DATE is converted to TOD, and DATE is converted to LTIME, DATE is timed in seconds at the bottom layer, TIME and TIME_OF_DAY are timed in milliseconds (ms), and LTIME is timed in nanoseconds (ns), so corresponding unit conversion is required.

Precautions

- If DATE is converted from other types, there may be values stored internally. When an INT type data 10 is converted into DATE through INT_TO_DATE, the output is displayed as 1970-01-01, but it is actually saved internally for 10 seconds. When this value passes through DATE_TO_INT, it is displayed as 10 instead of 0, and when this value passes through DATE_TO_BOOL, it is displayed as TRUE.

14.1.44 DINT_TO_<TYPE>/DINT type conversion instruction

DINT_TO_<TYPE>

Function

- It is used to convert DINT data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DINT_TO_BYTE	Input s1																				●	●					
	Output d																										
DINT_TO_DATE	Input s1																				●	●					
	Output d																										
DINT_TO_BOOL	Input s1																				●	●					
	Output d		●	●	●	●																					
DINT_TO_DT	Input s1																				●	●					
	Output d																										
DINT_TO_DWORD	Input s1																				●	●					
	Output d																										
DINT_TO_INT	Input s1																				●	●					
	Output d															●	●	●	●	●							
DINT_TO_REAL	Input s1																				●	●					
	Output d																										
DINT_TO_SINT	Input s1																				●	●					
	Output d																										
DINT_TO_TIME	Input s1																				●	●					
	Output d																										
DINT_TO_TOD	Input s1																				●	●					
	Output d																										
DINT_TO_UDINT	Input s1																				●	●					
	Output d																										
DINT_TO_UINT	Input s1																				●	●					
	Output d																										
DINT_TO_USINT	Input s1																				●	●					
	Output d																										
DINT_TO_WORD	Input s1																				●	●					
	Output d																										
DINT_TO_LREAL	Input s1																				●	●					
	Output d																										
DINT_TO_LWORD	Input s1																				●	●					

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
	Output d																										
DINT_TO_LINT	Input s1																				•	•					
	Output d																										
DINT_TO_LTIME	Input s1																				•	•					
	Output d																										

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
DINT_TO_BYTE	Input s1	DINT	DINT_TO_TOD	Input s1	DINT
	Output d	BYTE		Output d	TIME_OF_DAY
DINT_TO_DATE	Input s1	DINT	DINT_TO_UDINT	Input s1	DINT
	Output d	DATE		Output d	UDINT
DINT_TO_BOOL	Input s1	DINT	DINT_TO_UINT	Input s1	DINT
	Output d	BOOL		Output d	UINT
DINT_TO_DT	Input s1	DINT	DINT_TO_USINT	Input s1	DINT
	Output d	DATE_AND_TIME		Output d	USINT
DINT_TO_DWORD	Input s1	DINT	DINT_TO_WORD	Input s1	DINT
	Output d	DWORD		Output d	WORD
DINT_TO_INT	Input s1	DINT	DINT_TO_LREAL	Input s1	DINT
	Output d	INT		Output d	LREAL
DINT_TO_REAL	Input s1	DINT	DINT_TO_LWORD	Input s1	DINT
	Output d	REAL		Output d	LWORD
DINT_TO_SINT	Input s1	DINT	DINT_TO_LINT	Input s1	DINT
	Output d	SINT		Output d	LINT
DINT_TO_TIME	Input s1	DINT	DINT_TO_LTIME	Input s1	DINT
	Output d	TIME		Output d	LTIME
DINT_TO_STRING	Input s1	DINT			
	Output d	STRING			

Instructions

- When DINT is converted to BOOL, the non-0 output is TRUE;
- When DINT is converted to DATE, DATE is timed in seconds at the bottom layer. When DINT_TO_DATE input is 86400, the output is 1970-01-02;
- When DINT is converted to DT, DATE_AND_TIME is timed in seconds at the bottom layer. When the DINT input is 61, the output is 1970-01-01-0:01:1.

14.1.45 DT_TO_<TYPE>/DT type conversion instruction

DT_TO_<TYPE>

Function

- The date is saved as a DWORD internally (the seconds since January 1, 1970);
- It is used to convert DATE_AND_TIME data type to other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DT_TO_DINT	Input s1																									
	Output d																				●	●				
DT_TO_BOOL	Input s1																									
	Output d		●	●	●	●																				
DT_TO_INT	Input s1																									
	Output d															●	●	●	●	●						

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
DT_TO_BYTE	Input s1	DATE_AND_TIME	DT_TO_UINT	Input s1	DATE_AND_TIME
	Output d	BYTE		Output d	UINT
DT_TO_DATE	Input s1	DATE_AND_TIME	DT_TO_USINT	Input s1	DATE_AND_TIME
	Output d	DATE		Output d	USINT
DT_TO_DINT	Input s1	DATE_AND_TIME	DT_TO_WORD	Input s1	DATE_AND_TIME
	Output d	DINT		Output d	WORD
DT_TO_BOOL	Input s1	DATE_AND_TIME	DT_TO_LREAL	Input s1	DATE_AND_TIME
	Output d	BOOL		Output d	LREAL
DT_TO_DWORD	Input s1	DATE_AND_TIME	DT_TO_USINT	Input s1	DATE_AND_TIME
	Output d	DWORD		Output d	USINT
DT_TO_INT	Input s1	DATE_AND_TIME	DT_TO_WORD	Input s1	DATE_AND_TIME
	Output d	INT		Output d	WORD
DT_TO_REAL	Input s1	DATE_AND_TIME	DT_TO_LREAL	Input s1	DATE_AND_TIME
	Output d	REAL		Output d	LREAL
DT_TO_SINT	Input s1	DATE_AND_TIME	DT_TO_LWORD	Input s1	DATE_AND_TIME
	Output d	SINT		Output d	LWORD
DT_TO_TIME	Input s1	DATE_AND_TIME	DT_TO_LINT	Input s1	DATE_AND_TIME
	Output d	TIME		Output d	LINT
DT_TO_TOD	Input s1	DATE_AND_TIME	DT_TO_LTIME	Input s1	DATE_AND_TIME
	Output d	TIME_OF_DAY		Output d	LTIME
DT_TO_STRING	Input s1	DT			
	Output d	STRING			

Instructions

▪ When DT is converted to BOOL, the input value is 1970-01-01-0:0:0 and the output value is FALSE; otherwise, the output value is TRUE;

▪ When DT is converted to DATE, the input value is 1970-01-01-1:1:1 and the input result is 1970-01-01, but there is a value stored internally;

▪ When DT is converted to TOD, DT is converted to TIME, and DT is converted to LTIME, DATE_AND_TIME is timed in seconds at the bottom layer, TIME and TIME_OF_DAY are timed in milliseconds (ms), and LTIME is timed in nanoseconds (ns), so corresponding unit conversion is required.

14.1.46 DWORD_TO_<TYPE>/DWORD type conversion instruction

DWORD_TO_<TYPE>

Function

- It is used to convert DWORD data type to other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DWORD_TO_DINT	Input s1																									
	Output d																				•	•				
DWORD_TO_BOOL	Input s1																									
	Output d		•	•	•	•																				
DWORD_TO_INT	Input s1																									
	Output d															•	•	•	•	•						

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
DWORD_TO_BYTE	Input s1	DWORD	DWORD_TO_TOD	Input s1	DWORD
	Output d	BYTE		Output d	TIME_OF_DAY
DWORD_TO_DATE	Input s1	DWORD	DWORD_TO_UDINT	Input s1	DWORD
	Output d	DATE		Output d	UDINT
DWORD_TO_DINT	Input s1	DWORD	DWORD_TO_UINT	Input s1	DWORD
	Output d	DINT		Output d	UINT
DWORD_TO_DT	Input s1	DWORD	DWORD_TO_USINT	Input s1	DWORD
	Output d	DATE_AND_TIME		Output d	USINT
DWORD_TO_BOOL	Input s1	DWORD	DWORD_TO_WORD	Input s1	DWORD
	Output d	BOOL		Output d	WORD
DWORD_TO_INT	Input s1	DWORD	DWORD_TO_LREAL	Input s1	DWORD
	Output d	INT		Output d	LREAL
DWORD_TO_REAL	Input s1	DWORD	DWORD_TO_LWORD	Input s1	DWORD
	Output d	REAL		Output d	LWORD
DWORD_TO_SINT	Input s1	DWORD	DWORD_TO_LINT	Input s1	DWORD
	Output d	SINT		Output d	LINT
DWORD_TO_TIME	Input s1	DWORD	DWORD_TO_LTIME	Input s1	DWORD
	Output d	TIME		Output d	LTIME
DWORD_TO_STRING	Input s1	DWORD			
	Output d	STRING			

Instructions

- When DWORD is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE;
- When DWORD is converted to DATE, DATE is timed in seconds at the bottom layer. When the input value is 86400, the output is

1970-01-02;

▪ When DWORD is converted to DT, DATE_AND_TIME is timed in seconds at the bottom layer. When the input value is 86401, the output is 1970-01-02-0:0:1.

14.1.47 INT_TO_<TYPE>/INT type conversion instruction

INT_TO_<TYPE>

Function

- It is used to convert INT data type to other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
INT_TO_BYTE	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_DATE	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_DINT	Input s1															●	●	●	●	●							
	Output d																				●	●					
INT_TO_DT	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_DWORD	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_BOOL	Input s1															●	●	●	●	●							
	Output d		●	●	●	●																					
INT_TO_REAL	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_SINT	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_TIME	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_TOD	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_UDINT	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_UINT	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_USINT	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_WORD	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_LREAL	Input s1															●	●	●	●	●							
	Output d																										
INT_TO_LWORD	Input s1															●	●	●	●	●							

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
	Output d																									
INT_TO_LINT	Input s1															•	•	•	•	•						
	Output d																									
INT_TO_LTIME	Input s1															•	•	•	•	•						
	Output d																									

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
INT_TO_BYTE	Input s1	INT	INT_TO_TOD	Input s1	INT
	Output d	BYTE		Output d	TIME_OF_DAY
INT_TO_DATE	Input s1	INT	INT_TO_UDINT	Input s1	INT
	Output d	DATE		Output d	UDINT
INT_TO_DINT	Input s1	INT	INT_TO_UINT	Input s1	INT
	Output d	DINT		Output d	UINT
INT_TO_DT	Input s1	INT	INT_TO_USINT	Input s1	INT
	Output d	DATE_AND_TIME		Output d	USINT
INT_TO_DWORD	Input s1	INT	INT_TO_WORD	Input s1	INT
	Output d	DWORD		Output d	WORD
INT_TO_BOOL	Input s1	INT	INT_TO_LREAL	Input s1	INT
	Output d	BOOL		Output d	LREAL
INT_TO_REAL	Input s1	INT	INT_TO_LWORD	Input s1	INT
	Output d	REAL		Output d	LWORD
INT_TO_SINT	Input s1	INT	INT_TO_LINT	Input s1	INT
	Output d	SINT		Output d	LINT
INT_TO_TIME	Input s1	INT	INT_TO_LTIME	Input s1	INT
	Output d	TIME		Output d	LTIME
INT_TO_STRING	Input s1	INT			
	Output d	STRING			

Instructions

- When INT is converted to BOOL, if the output value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE;
- When INT is converted to DT, DATE_AND_TIME is timed in seconds at the bottom layer. When the input value is 10, the output is 1970-01-01-0:0:10.

14.1.48 WORD_TO_<TYPE>/WORD type conversion instruction

WORD_TO_<TYPE>

Function

- It is used to convert WORD data type to other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
WORD_TO_DINT	Input s1																									
	Output d																				•	•				
WORD_TO_INT	Input s1																									
	Output d															•	•	•	•	•						
WORD_TO_BOOL	Input s1																									
	Output d		•	•	•	•																				

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
WORD_TO_BYTE	Input s1	WORD	WORD_TO_TOD	Input s1	WORD
	Output d	BYTE		Output d	TIME_OF_DAY
WORD_TO_DATE	Input s1	WORD	WORD_TO_UDINT	Input s1	WORD
	Output d	DATE		Output d	UDINT
WORD_TO_DINT	Input s1	WORD	WORD_TO_UINT	Input s1	WORD
	Output d	DINT		Output d	UINT
WORD_TO_DT	Input s1	WORD	WORD_TO_USINT	Input s1	WORD
	Output d	DATE_AND_TIME		Output d	USINT
WORD_TO_DWORD	Input s1	WORD	WORD_TO_BOOL	Input s1	WORD
	Output d	DWORD		Output d	BOOL
WORD_TO_INT	Input s1	WORD	WORD_TO_LREAL	Input s1	WORD
	Output d	INT		Output d	LREAL
WORD_TO_REAL	Input s1	WORD	WORD_TO_LWORD	Input s1	WORD
	Output d	REAL		Output d	LWORD

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
WORD_TO_SINT	Input s1	WORD	WORD_TO_LINT	Input s1	WORD
	Output d	SINT		Output d	LINT
WORD_TO_TIME	Input s1	WORD	WORD_TO_LTIME	Input s1	WORD
	Output d	TIME		Output d	LTIME
WORD_TO_STRING	Input s1	WORD			
	Output d	STRING			

Instructions

- When WORD is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE;
- When WORD is converted to DT, DATE_AND_TIME is timed in seconds at the bottom layer. When WORD is TRUE, the output is 1970-01-01-0:0:1.

14.1.49 REAL_TO_<TYPE>/REAL type conversion instruction

REAL_TO_<TYPE>

Function

- It is used to convert REAL data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
REAL_TO_DINT	Input s1																									
	Output d																				•	•				
REAL_TO_INT	Input s1																									
	Output d															•	•	•	•	•						
REAL_TO_BOOL	Input s1																									
	Output d		•	•	•	•																				

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
REAL_TO_BYTE	Input s1	REAL	REAL_TO_TOD	Input s1	REAL
	Output d	BYTE		Output d	TIME_OF_DAY
REAL_TO_DATE	Input s1	REAL	REAL_TO_UDINT	Input s1	REAL
	Output d	DATE		Output d	UDINT
REAL_TO_DINT	Input s1	REAL	REAL_TO_UINT	Input s1	REAL
	Output d	DINT		Output d	UINT
REAL_TO_DT	Input s1	REAL	REAL_TO_USINT	Input s1	REAL
	Output d	DATE_AND_TIME		Output d	USINT
REAL_TO_DWORD	Input s1	REAL	REAL_TO_WORD	Input s1	REAL
	Output d	DWORD		Output d	WORD
REAL_TO_INT	Input s1	REAL	REAL_TO_LREAL	Input s1	REAL
	Output d	INT		Output d	LREAL
REAL_TO_BOOL	Input s1	REAL	REAL_TO_LWORD	Input s1	REAL
	Output d	BOOL		Output d	LWORD
REAL_TO_SINT	Input s1	REAL	REAL_TO_LINT	Input s1	REAL
	Output d	SINT		Output d	LINT
REAL_TO_TIME	Input s1	REAL	REAL_TO_LTIME	Input s1	REAL
	Output d	TIME		Output d	LTIME
REAL_TO_STRING	Input s1	REAL			
	Output d	STRING			

Instructions

- When REAL is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE;
- When REAL is converted to INT, when REAL is 1.5, the output is 2, and when REAL is 1.4, the output is 1;

- When REAL is -1.5, the output is -2, and when REAL is -1.4, the output is -1.

14.1.50 SINT_TO_<TYPE>/SINT type conversion instruction

SINT_TO_<TYPE>

Function

- It is used to convert SINT data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SINT_TO_DINT	Input s1																									
	Output d																				•	•				
SINT_TO_INT	Input s1																									
	Output d															•	•	•	•	•						
SINT_TO_BOOL	Input s1																									
	Output d		•	•	•	•																				

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
SINT_TO_BYTE	Input s1	SINT	SINT_TO_TOD	Input s1	SINT
	Output d	BYTE		Output d	TIME_OF_DAY
SINT_TO_DATE	Input s1	SINT	SINT_TO_UDINT	Input s1	SINT
	Output d	DATE		Output d	UDINT
SINT_TO_DINT	Input s1	SINT	SINT_TO_UINT	Input s1	SINT
	Output d	DINT		Output d	UINT
SINT_TO_DT	Input s1	SINT	SINT_TO_USINT	Input s1	SINT
	Output d	DATE_AND_TIME		Output d	USINT
SINT_TO_DWORD	Input s1	SINT	SINT_TO_WORD	Input s1	SINT
	Output d	DWORD		Output d	WORD
SINT_TO_INT	Input s1	SINT	SINT_TO_LREAL	Input s1	SINT
	Output d	INT		Output d	LREAL
SINT_TO_REAL	Input s1	SINT	SINT_TO_LWORD	Input s1	SINT
	Output d	REAL		Output d	LWORD

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
SINT_TO_BOOL	Input s1	SINT	SINT_TO_LINT	Input s1	SINT
	Output d	BOOL		Output d	LINT
SINT_TO_TIME	Input s1	SINT	SINT_TO_LTIME	Input s1	SINT
	Output d	TIME		Output d	LTIME
SINT_TO_STRING	Input s1	SINT			
	Output d	STRING			

Instructions

- When SINT is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE;
- When SINT is converted to DT, DATE_AND_TIME is timed in seconds at the bottom layer. When the SINT is 61, the output is 1970-01-01-0:1:1.

14.1.51 TIME_TO_<TYPE>/TIME type conversion instruction

TIME_TO_<TYPE>

Function

- The unit of TIME is milliseconds;
- It is used to convert TIME data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TIME_TO_DINT	Input s1																									
	Output d																				•	•				
TIME_TO_INT	Input s1																									
	Output d															•	•	•	•	•						
TIME_TO_BOOL	Input s1																									
	Output d		•	•	•	•																				

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
TIME_TO_BYTE	Input s1	TIME	TIME_TO_TOD	Input s1	TIME
	Output d	BYTE		Output d	TIME_OF_DAY
TIME_TO_DATE	Input s1	TIME	TIME_TO_UINT	Input s1	TIME
	Output d	DATE		Output d	UINT
TIME_TO_DINT	Input s1	TIME	TIME_TO_USINT	Input s1	TIME
	Output d	DINT		Output d	USINT
TIME_TO_DT	Input s1	TIME	TIME_TO_WORD	Input s1	TIME
	Output d	DATE_AND_TIME		Output d	WORD
TIME_TO_DWORD	Input s1	TIME	TIME_TO_LREAL	Input s1	TIME
	Output d	DWORD		Output d	LREAL
TIME_TO_INT	Input s1	TIME	TIME_TO_LWORD	Input s1	TIME
	Output d	INT		Output d	LWORD
TIME_TO_REAL	Input s1	TIME	TIME_TO_ LINT	Input s1	TIME
	Output d	REAL		Output d	LINT
TIME_TO_SINT	Input s1	TIME	TIME_TO_LTIME	Input s1	TIME
	Output d	SINT		Output d	LTIME

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
TIME_TO_BOOL	Input s1	TIME	TIME_TO_UDINT	Input s1	TIME
	Output d	BOOL		Output d	UDINT
TIME_TO_STRING	Input s1	TIME			
	Output d	STRING			

Instructions

- When TIME is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE.
- When DT is converted to TOD, DT is converted to TIME, and DT is converted to LTIME, DATE_AND_TIME is timed in seconds at the bottom layer, TIME and TIME_OF_DAY are timed in milliseconds (ms), and LTIME is timed in nanoseconds (ns), so corresponding unit conversion is required.

14.1.52 TOD_TO_<TYPE>/TOD type conversion instruction

TOD_TO_<TYPE>

Function

- It is used to convert TIME_OF_DAY data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TOD_TO_DINT	Input s1																									
	Output d																				•	•				
TOD_TO_INT	Input s1																									
	Output d															•	•	•	•	•						
TOD_TO_BOOL	Input s1																									
	Output d		•	•	•	•																				

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
TOD_TO_BYTE	Input s1	TIME_OF_DAY	TOD_TO_UDINT	Input s1	TIME_OF_DAY
	Output d	BYTE		Output d	UDINT
TOD_TO_DATE	Input s1	TIME_OF_DAY	TOD_TO_UINT	Input s1	TIME_OF_DAY
	Output d	DATE		Output d	UINT
TOD_TO_DINT	Input s1	TIME_OF_DAY	TOD_TO_USINT	Input s1	TIME_OF_DAY
	Output d	DINT		Output d	USINT
TOD_TO_DT	Input s1	TIME_OF_DAY	TOD_TO_WORD	Input s1	TIME_OF_DAY
	Output d	DATE_AND_TIME		Output d	WORD
TOD_TO_DWORD	Input s1	TIME_OF_DAY	TOD_TO_LREAL	Input s1	TIME_OF_DAY
	Output d	DWORD		Output d	LREAL
TOD_TO_INT	Input s1	TIME_OF_DAY	TOD_TO_ LWORD	Input s1	TIME_OF_DAY
	Output d	INT		Output d	LWORD
TOD_TO_REAL	Input s1	TIME_OF_DAY	TOD_TO_ LINT	Input s1	TIME_OF_DAY
	Output d	REAL		Output d	LINT
TOD_TO_SINT	Input s1	TIME_OF_DAY	TOD_TO_LTIME	Input s1	TIME_OF_DAY
	Output d	SINT		Output d	LTIME
TOD_TO_TIME	Input s1	TIME_OF_DAY	TOD_TO_UDINT	Input s1	TIME_OF_DAY
	Output d	TIME		Output d	UDINT
TOD_TO_BOOL	Input s1	TIME_OF_DAY	TOD_TO_UINT	Input s1	TIME_OF_DAY
	Output d	BOOL		Output d	UINT
TOD_TO_STRING	Input s1	TIME_OF_DAY			

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
	Output d	STRING			

- When TOD is converted to BOOL and when the input value is 00:00:00, the output is FALSE; otherwise, the output is TRUE;
- When TOD is converted to INT and when the input value is 01:05:05, the output is -27160
- When TOD is converted to DT, DATE_AND_TIME is timed in seconds at the bottom layer. When the input value is 01:05:05, the output is 1970-01-01-01:05:05;
- When TOD is converted to DATE, TIME is converted to DT, and TOD is converted to LTIME, TOD is timed in milliseconds (ms), and LTIME is timed in nanoseconds (ns), DATE_AND_TIME and DATE are timed in seconds at the bottom layer, so corresponding unit conversion is required.

14.1.53 UDINT_TO_<TYPE>/UDINT type conversion instruction

UDINT_TO_<TYPE>

Function

- It is used to convert UDINT data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
UDINT_TO_DINT	Input s1																										
	Output d																				•	•					
UDINT_TO_INT	Input s1																										
	Output d															•	•	•	•	•							
UDINT_TO_BOOL	Input s1																										
	Output d		•	•	•	•																					

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
UDINT_TO_BYTE	Input s1	UDINT	UDINT_TO_TOD	Input s1	UDINT
	Output d	BYTE		Output d	TIME_OF_DAY
UDINT_TO_DATE	Input s1	UDINT	UDINT_TO_UINT	Input s1	UDINT
	Output d	DATE		Output d	UINT
UDINT_TO_DINT	Input s1	UDINT	UDINT_TO_USINT	Input s1	UDINT
	Output d	DINT		Output d	USINT
UDINT_TO_DT	Input s1	UDINT	UDINT_TO_WORD	Input s1	UDINT
	Output d	DATE_AND_TIME		Output d	WORD
UDINT_TO_DWORD	Input s1	UDINT	UDINT_TO_LREAL	Input s1	UDINT
	Output d	DWORD		Output d	LREAL
UDINT_TO_INT	Input s1	UDINT	UDINT_TO_ LWORD	Input s1	UDINT
	Output d	INT		Output d	LWORD
UDINT_TO_REAL	Input s1	UDINT	UDINT_TO_ LINT	Input s1	UDINT
	Output d	REAL		Output d	LINT
UDINT_TO_SINT	Input s1	UDINT	UDINT_TO_LTIME	Input s1	UDINT
	Output d	SINT		Output d	LTIME
UDINT_TO_TIME	Input s1	UDINT	UDINT_TO_UDINT	Input s1	UDINT
	Output d	TIME		Output d	UDINT
UDINT_TO_BOOL	Input s1	UDINT	UDINT_TO_UINT	Input s1	UDINT
	Output d	BOOL		Output d	UINT
UDINT_TO_STRING	Input s1	UDINT			
	Output d	STRING			

Instructions

- When UDINT is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE.

14.1.54 UINT_TO_<TYPE>/UINT type conversion instruction

UINT_TO_<TYPE>

Function

- It is used to convert UINT data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
UINT_TO_DINT	Input s1																										
	Output d																				•	•					
UINT_TO_INT	Input s1																										
	Output d															•	•	•	•	•							
UINT_TO_BOOL	Input s1																										
	Output d		•	•	•	•																					

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
UINT_TO_BYTE	Input s1	UINT	UINT_TO_TOD	Input s1	UINT
	Output d	BYTE		Output d	TIME_OF_DAY
UINT_TO_DATE	Input s1	UINT	UINT_TO_BOOL	Input s1	UINT
	Output d	DATE		Output d	BOOL
UINT_TO_DINT	Input s1	UINT	UINT_TO_USINT	Input s1	UINT
	Output d	DINT		Output d	USINT
UINT_TO_DT	Input s1	UINT	UINT_TO_WORD	Input s1	UINT
	Output d	DATE_AND_TIME		Output d	WORD
UINT_TO_DWORD	Input s1	UINT	UINT_TO_LREAL	Input s1	UINT
	Output d	DWORD		Output d	LREAL
UINT_TO_INT	Input s1	UINT	UINT_TO_LWORD	Input s1	UINT
	Output d	INT		Output d	LWORD
UINT_TO_REAL	Input s1	UINT	UINT_TO_ LINT	Input s1	UINT
	Output d	REAL		Output d	LINT
UINT_TO_SINT	Input s1	UINT	UINT_TO_LTIME	Input s1	UINT
	Output d	SINT		Output d	LTIME
UINT_TO_TIME	Input s1	UINT	UINT_TO_UDINT	Input s1	UINT
	Output d	TIME		Output d	UDINT
UINT_TO_STRING	Input s1	UINT			
	Output d	STRING			

Instructions

- When UINT is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE.

14.1.55 USINT_TO_<TYPE>/USINT type conversion instruction

USINT_TO_<TYPE>

Function

- It is used to convert USINT data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
USINT_TO_DINT	Input s1																									
	Output d																				●	●				
USINT_TO_INT	Input s1																									
	Output d															●	●	●	●	●						
USINT_TO_BOOL	Input s1																									
	Output d		●	●	●	●																				

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instructions

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
USINT_TO_BYTE	Input s1	USINT	USINT_TO_TOD	Input s1	USINT
	Output d	BYTE		Output d	TIME_OF_DAY
USINT_TO_DATE	Input s1	USINT	USINT_TO_UDINT	Input s1	USINT
	Output d	DATE		Output d	UDINT
USINT_TO_DINT	Input s1	USINT	USINT_TO_UINT	Input s1	USINT
	Output d	DINT		Output d	UINT
USINT_TO_DT	Input s1	USINT	USINT_TO_BOOL	Input s1	USINT
	Output d	DATE_AND_TIME		Output d	BOOL
USINT_TO_DWORD	Input s1	USINT	USINT_TO_WORD	Input s1	USINT
	Output d	DWORD		Output d	WORD
USINT_TO_INT	Input s1	USINT	USINT_TO_LREAL	Input s1	USINT
	Output d	INT		Output d	LREAL
USINT_TO_REAL	Input s1	USINT	USINT_TO_LWORD	Input s1	USINT
	Output d	REAL		Output d	LWORD
USINT_TO_SINT	Input s1	USINT	USINT_TO_LINT	Input s1	USINT
	Output d	SINT		Output d	LINT
USINT_TO_TIME	Input s1	USINT	USINT_TO_LTIME	Input s1	USINT
	Output d	TIME		Output d	LTIME
USINT_TO_STRING	Input s1	USINT			
	Output d	STRING			

- When USINT is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE.

14.1.56 LREAL_TO_<TYPE>/LREAL type conversion instruction

LREAL_TO_<TYPE>

Function

- It is used to convert LREAL data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																									Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LREAL_TO_DINT	Input s1																											
	Output d																				●	●						
LREAL_TO_INT	Input s1																											
	Output d															●	●	●	●	●								
LREAL_TO_BOOL	Input s1																											
	Output d	●	●	●	●																							

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
LREAL_TO_BYTE	Input s1	LREAL	LREAL_TO_TOD	Input s1	LREAL
	Output d	BYTE		Output d	TIME_OF_DAY
LREAL_TO_DATE	Input s1	LREAL	LREAL_TO_UDINT	Input s1	LREAL
	Output d	DATE		Output d	UDINT
LREAL_TO_DINT	Input s1	LREAL	LREAL_TO_UINT	Input s1	LREAL
	Output d	DINT		Output d	UINT
LREAL_TO_DT	Input s1	LREAL	LREAL_TO_USINT	Input s1	LREAL
	Output d	DATE_AND_TIME		Output d	USINT
LREAL_TO_DWORD	Input s1	LREAL	LREAL_TO_WORD	Input s1	LREAL
	Output d	DWORD		Output d	WORD
LREAL_TO_INT	Input s1	LREAL	LREAL_TO_REAL	Input s1	LREAL
	Output d	INT		Output d	REAL
LREAL_TO_BOOL	Input s1	LREAL	LREAL_TO_LWORD	Input s1	LREAL
	Output d	BOOL		Output d	LWORD
LREAL_TO_SINT	Input s1	LREAL	LREAL_TO_LINT	Input s1	LREAL
	Output d	SINT		Output d	LINT
LREAL_TO_TIME	Input s1	LREAL	LREAL_TO_LTIME	Input s1	LREAL
	Output d	TIME		Output d	LTIME
LREAL_TO_STRING	Input s1	LINT			
	Output d	STRING			

Instructions

- When LREAL is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE;
- When LREAL is converted to INT, when LREAL is 1.5, the output is 2, and when LREAL is 1.4, the output is 1;

  When LREAL is -1.5, the output is -2, and when LREAL is -1.4, the output is -1.

14.1.57 LWORD_TO_<TYPE>/LWORD type conversion instruction

LWORD_TO_<TYPE>

Function

- It is used to convert LWORD data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																									Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LREAL_TO_DINT	Input s1																											
	Output d																				●	●						
LREAL_TO_INT	Input s1																											
	Output d															●	●	●	●	●								
LREAL_TO_BOOL	Input s1																											
	Output d	●	●	●	●																							

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
LWORD_TO_BYTE	Input s1	LWORD	LWORD_TO_TOD	Input s1	LWORD
	Output d	BYTE		Output d	TIME_OF_DAY
LWORD_TO_DATE	Input s1	LWORD	LWORD_TO_UDINT	Input s1	LWORD
	Output d	DATE		Output d	UDINT
LWORD_TO_DINT	Input s1	LWORD	LWORD_TO_UINT	Input s1	LWORD
	Output d	DINT		Output d	UINT
LWORD_TO_DT	Input s1	LWORD	LWORD_TO_USINT	Input s1	LWORD
	Output d	DATE_AND_TIME		Output d	USINT
LWORD_TO_DWORD	Input s1	LWORD	LWORD_TO_WORD	Input s1	LWORD
	Output d	DWORD		Output d	WORD
LWORD_TO_INT	Input s1	LWORD	LWORD_TO_REAL	Input s1	LWORD
	Output d	INT		Output d	REAL
LWORD_TO_BOOL	Input s1	LWORD	LWORD_TO_LREAL	Input s1	LWORD
	Output d	BOOL		Output d	LREAL
LWORD_TO_SINT	Input s1	LWORD	LWORD_TO_LINT	Input s1	LWORD
	Output d	SINT		Output d	LINT
LWORD_TO_TIME	Input s1	LWORD	LWORD_TO_LTIME	Input s1	LWORD
	Output d	TIME		Output d	LTIME

Instructions

- When LWORD is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE.

14.1.58 LINT_TO_<TYPE>/LINT type conversion instruction

LINT_TO_<TYPE>

Function

- It is used to convert LINT data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LINT_TO_DINT	Input s1																										
	Output d																				●	●					
LINT_TO_INT	Input s1																										
	Output d															●	●	●	●	●							
LINT_TO_BOOL	Input s1																										
	Output d		●	●	●	●																					

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
LINT_TO_BYTE	Input s1	LINT	LINT_TO_TOD	Input s1	LINT
	Output d	BYTE		Output d	TIME_OF_DAY
LINT_TO_DATE	Input s1	LINT	LINT_TO_UDINT	Input s1	LINT
	Output d	DATE		Output d	UDINT
LINT_TO_DINT	Input s1	LINT	LINT_TO_UINT	Input s1	LINT
	Output d	DINT		Output d	UINT
LINT_TO_DT	Input s1	LINT	LINT_TO_USINT	Input s1	LINT
	Output d	DATE_AND_TIME		Output d	USINT
LINT_TO_DWORD	Input s1	LINT	LINT_TO_WORD	Input s1	LINT
	Output d	DWORD		Output d	WORD
LINT_TO_INT	Input s1	LINT	LINT_TO_REAL	Input s1	LINT
	Output d	INT		Output d	REAL
LINT_TO_BOOL	Input s1	LINT	LINT_TO_LREAL	Input s1	LINT
	Output d	BOOL		Output d	LREAL
LINT_TO_SINT	Input s1	LINT	LINT_TO_LWORD	Input s1	LINT
	Output d	SINT		Output d	LWORD
LINT_TO_TIME	Input s1	LINT	LINT_TO_LTIME	Input s1	LINT
	Output d	TIME		Output d	LTIME
LWORD_TO_STRING	Input s1	LWROD			
	Output d	STRING			

Instructions

- When LINT is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE.

14.1.59 LTIME_TO_<TYPE>/LTIME Type Conversion Instruction

LTIME_TO_<TYPE>

Function

- It is used to convert LTIME data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LTIME_TO_DINT	Input s1																										
	Output d																				●	●					
LTIME_TO_INT	Input s1																										
	Output d															●	●	●	●	●							
LTIME_TO_BOOL	Input s1																										
	Output d		●	●	●	●																					

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;

Variable type used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
LTIME_TO_BYTE	Input s1	LTIME	LTIME_TO_TOD	Input s1	LTIME
	Output d	BYTE		Output d	TIME_OF_DAY
LTIME_TO_DATE	Input s1	LTIME	LTIME_TO_UDINT	Input s1	LTIME
	Output d	DATE		Output d	UDINT
LTIME_TO_DINT	Input s1	LTIME	LTIME_TO_UINT	Input s1	LTIME
	Output d	DINT		Output d	UINT
LTIME_TO_DT	Input s1	LTIME	LTIME_TO_USINT	Input s1	LTIME
	Output d	DATE_AND_TIME		Output d	USINT
LTIME_TO_DWORD	Input s1	LTIME	LTIME_TO_WORD	Input s1	LTIME
	Output d	DWORD		Output d	WORD
LTIME_TO_INT	Input s1	LTIME	LTIME_TO_REAL	Input s1	LTIME
	Output d	INT		Output d	REAL
LTIME_TO_BOOL	Input s1	LTIME	LTIME_TO_LREAL	Input s1	LTIME
	Output d	BOOL		Output d	LREAL
LTIME_TO_SINT	Input s1	LTIME	LTIME_TO_LWORD	Input s1	LTIME
	Output d	SINT		Output d	LWORD
LTIME_TO_TIME	Input s1	LTIME	LTIME_TO_LINT	Input s1	LTIME
	Output d	TIME		Output d	LINT
LINT_TO_STRING	Input s1	LINT			
	Output d	STRING			

Instructions

- When LTIME is converted to BOOL, if the input value is equal to 0, the output value is FALSE; otherwise, the output value is TRUE;
- When LTIME is converted to DATE, LTIME is converted to DT, and LTIME is converted to TOD and LTIME is converted to TIME, LTIME is timed in nanoseconds (ns), TOD is timed in milliseconds (ms), and DATE_AND_TIME and DATE are timed in seconds at the bottom layer,

so corresponding unit conversion is required.

14.1.60 TO_<TYPE> Type conversion instruction

TO_<TYPE>

Function

- It is used to convert a certain data type into other data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TO_BYTE	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_DATE	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_DINT	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																				●	●					
TO_DT	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_DWORD	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_INT	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d															●	●	●	●	●							
TO_BOOL	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d		●	●	●	●																					
TO_SINT	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_TIME	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_TOD	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_UDINT	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_UINT	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_USINT	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_WORD	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_LREAL	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_REAL	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TO_LWORD	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_LINT	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_LTIME	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										
TO_STRING	Input s1	●	●	●	●	●										●	●	●	●	●	●	●					
	Output d																										

- The other instructions do not support the use of devices.

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types;
- ANY: BOOL, BYTE, DATE, DATE_AND_TIME, SINT, TIME, TIME_OF_DAY, UDINT, UINT, USINT, WORD, DWORD, INT, DINT, REAL, LREAL, LWORD, LINT, LTIME.

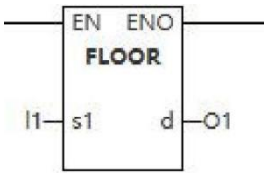
Variable Type Used

Instruction	Parameter	Variable type	Instruction	Parameter	Variable type
TO_BYTE	Input s1	ANY	TO_UDINT	Input s1	ANY
	Output d	BYTE		Output d	UDINT
TO_DATE	Input s1	ANY	TO_UINT	Input s1	ANY
	Output d	DATE		Output d	UINT
TO_DINT	Input s1	ANY	TO_USINT	Input s1	ANY
	Output d	DINT		Output d	USINT
TO_DT	Input s1	ANY	TO_WORD	Input s1	ANY
	Output d	DATE_AND_TIME		Output d	WORD
TO_DWORD	Input s1	ANY	TO_REAL	Input s1	ANY
	Output d	DWORD		Output d	REAL
TO_INT	Input s1	ANY	TO_LREAL	Input s1	ANY
	Output d	INT		Output d	LREAL
TO_BOOL	Input s1	ANY	TO_LWORD	Input s1	ANY
	Output d	BOOL		Output d	LWORD
TO_SINT	Input s1	ANY	TO_LINT	Input s1	ANY
	Output d	SINT		Output d	LINT
TO_TIME	Input s1	ANY	TO_LTIME	Input s1	ANY
	Output d	TIME		Output d	LTIME
TO_TOD	Input s1	ANY	TO_STRING	Input s1	ANY
	Output d	TIME_OF_DAY		Output d	STRING

Instructions

- According to the input variable type, the data will be automatically converted into corresponding data type internally. The detailed conversion function is the same as above.

14.1.61 FLOOR/Round down the first decimal place instruction



The first decimal place is rounded down. Device is not supported.

Variable Type Used

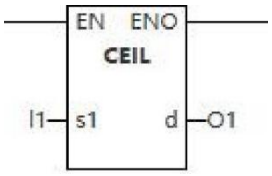
Instruction	Parameter	Variable type
FLOOR	Input s1	REAL, LREAL
	Output d	INT, DINT

Function

- Round the value of s1 and round down the data at the first decimal place.

14.1.62 CEIL/Round up the first decimal place instruction

CEIL



The first decimal place is rounded up. Device is not supported.

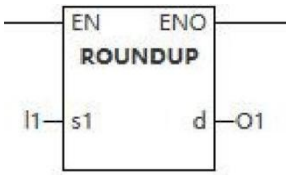
Variable Type Used

Instruction	Parameter	Variable type
CEIL	Input s1	REAL, LREAL
	Output d	INT, DINT

Function

- Round the value of s1 and round up the data at the first decimal place.

14.1.63 ROUNDUP/ Round up instruction for the first digit after the decimal point



Rounding up the first decimal place to the next integer.

Device is not supported.

Variable Type Used

Instruction	Parameter	Variable type
ROUNDUP	Input s1	REAL, LREAL
	Output d	INT, DINT

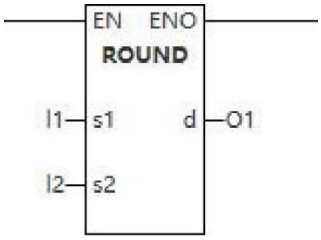
Function

Round s1 up to the next integer at the first decimal place.

Description of carry and round instructions

Input value	ROUNDUP	CEIL	FLOOR
1.6	2	2	1
1.5	2	2	1
1.4	2	2	1
2.5	3	3	2
-1.6	-2	-1	-2
-1.5	-2	-1	-2
-1.4	-2	-1	-2
-2.5	-3	-2	-3

14.1.64 ROUND instruction



Rounding half up/Round half to even Device is not supported.

Variable Type Used

Instruction	Parameter	Variable type
ROUND	Input s1	REAL, LREAL
	Input s2	INT
	Output d	INT, DINT

Function

If s2 is 0, round the value of s1 and round the data at the first decimal place.

If s2 is 1, round the value of s1 and round half the data at the first decimal place to even.

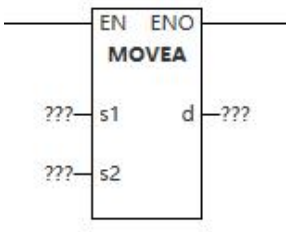
[Round half to even] Descriptions

Value of the decimal part	Disposal	Example
Less than 0.5	Discard	1.49->1 -1.49->-1
0.5	If the unit digit is even, round down; If the unit digit is odd, round up.	1.50->2 2.50->2 -1.50->-2 -2.50->-2
Greater than 0.5	Carry	1.51->2 -1.51->-2

Error code

Error code	Content
4084H	The data input in (s2) exceed the specified range.

14.1.65 MOVEA/Batch transfer instruction



Substituting the value of the variable specified in (s1) into the variable specified in (d) in batches according to the length in (s2);

Device used (only a single device can be used, and device splicing and offset are not supported)

Command	Parameter	Devices																							Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MOVEA	Input s1															●	●	●	●	●	●	●	●	●			
	Input s2															●	●	●	●	●	●	●	●	●			
	Output d															●	●	●	●	●	●	●	●	●			

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Parameter	Variable Type
MOVEA	Input s1	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME、TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME
	Input s2	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
	Output d	BOOL, BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT, REAL, LREAL, TIME、TIME_OF_DAY, DATE, DATE_AND_TIME, LTIME

Function

- Substituting the value of the variable specified in (s1) into the variable specified in (d) in batches according to the length in (s2);

14.1.66 FIND/FIND string position

FIND (P)

Finds the string S2 from the input string S1 and returns the position of the first character of the string S2 in S1.

-[FIND (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source string	-	STRING	-	STRING
Input s2	Target string	-	STRING	-	STRING
Output d	The position of the target string in the input source string	[0~255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT

Device used

Instruction	Parameter	Devices																								Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
FIND	Input s1																										
	Input s2																										
	Output d																										

Function

▪ Finds the string S2 from the input string S1, and returns the position of the first character of the string S2 in S1 and outputs it to the specified device in d.

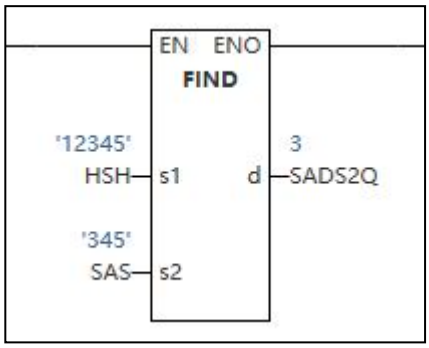
Precautions

▪ If the string to be looking for does not exist, the return value is 0;

▪ When the source string contains multiple matches, the return value is the one with the smallest position;

▪ The return value indicates the position of the search result. The position of the first character on the leftmost side of the string is 1, and the subsequent byte is added with 1.

Example



Finds the string S2 from the input string S1 and returns the position of the first character of the string S2 in S1.

14.1.67 INSERT/string insertion

INSERT (P)

Insert a string in the middle of the input source string on the left, and output the synthesized string.

-[INSERT (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source string	-	STRING	-	STRING
Input s2	Inserted string	-	STRING	-	STRING
Input s3	Insertion location	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Output d	Output String		STRING	-	STRING

Device used

Instruction	Parameter	Devices																							Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
INSERT	Input s1																									
	Input s2																									
	Input s3																									
	Output d																									

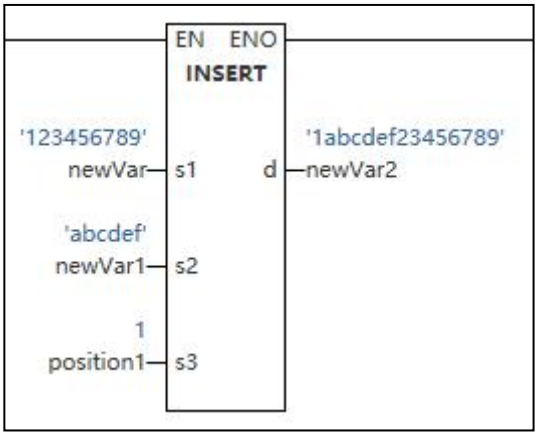
Function

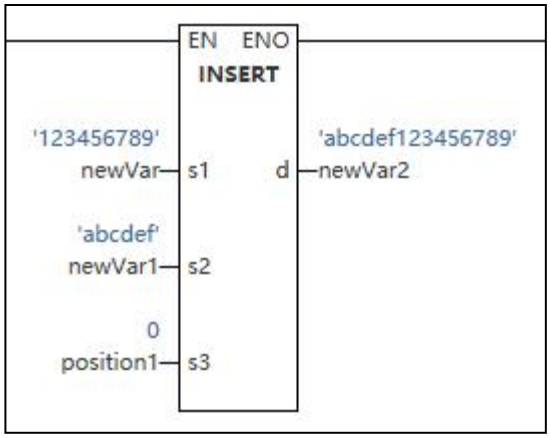
- Insert a string in the middle of the input source string on the left, and output the synthesized string.

Precautions

- S3 represents the insertion position, the position of the first character on the leftmost side of the string is 1, and the subsequent byte is added with 1;
- When the sum of the actual lengths of S1 and S2 strings is greater than 80, only the first 80 characters will be returned and report error;
- If the length of S3 is greater than the length of the source string, the source string is returned and an error is reported.

Example





14.1.68 LEFT/fetch string from left

LEFT(P)

Fetches a string of the specified length starting from the left side of the source string and returns a string of the taken length.

-[LEFT (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source string	-	STRING	-	STRING
Input s2	Take data length	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Output d	Output string	-	STRING	-	STRING

Device used

Instruction	Parameter	Devices																							Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LEFT	Input s1																									
	Input s2																									
	Output d																									

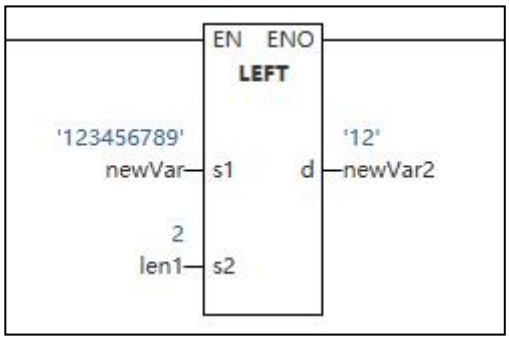
Function

- Fetches a string of the specified length starting from the left side of the source string and returns a string of the taken length.

Precautions

- If the input S2 is 0, the output string is empty;
- If the input S2 is greater than the input source string length, the output string is the original input string, and report error.

Example



14.1.69 RIGHT/fetch string from right

RIGHT(P)

Fetches a string of the specified length starting from the right side of the source string and returns a string of the fetched length.

-[RIGHT (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source string	-	STRING	-	STRING
Input s2	Take data length	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Output d	Output string	-	STRING	-	STRING

Device used

Instruction	Parameter	Devices																						Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RIGHT	Input s1																										
	Input s2																										
	Output d																										

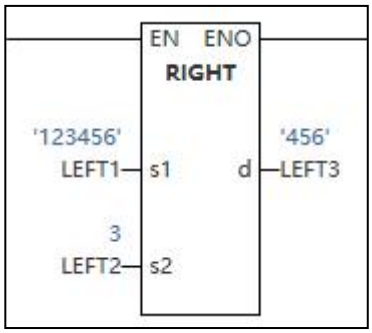
Function

- Fetches a string of the specified length starting from the right side of the source string and returns a string of the fetched length.

Precautions

- If the input S2 is 0, the output string is empty;
- If the input S2 is greater than that length of the input source string, the output string is the original input string, and report error.

Example



14.1.70 DELETE/string deletion

DELETE (P)

Delete the specified length string at the specified position in the input string, and output the processed string

-[DELETE (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source string	-	STRING	-	STRING
Input s2	Delete length	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Input s3	Delete location	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Output d	Output String	-	STRING	-	STRING

Device used

Instruction	Parameter	Devices																						Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DELETE	Input s1																										
	Input s2																										
	Input s3																										
	Output d																										

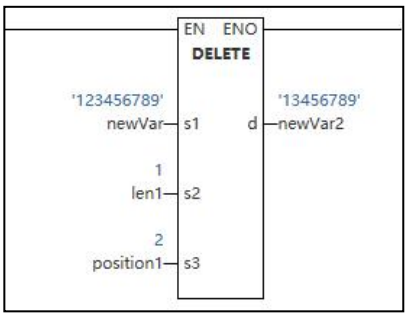
Function

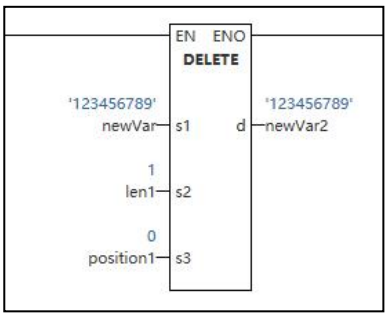
- Deletes the specified position and length string in the input string, and outputs the processed string.

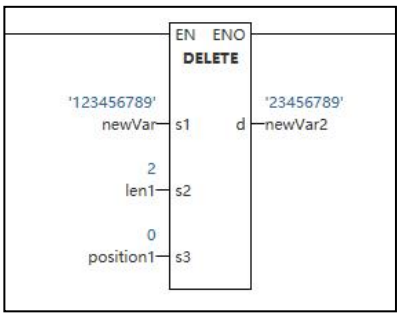
Precautions

- S3 represents the deletion position, the position of the first character on the leftmost side of the string is 1, and the subsequent byte is added with 1;
- S3 is greater than the source string length to output an empty string;
- The s2 length is greater than the source string length returns the source string.

Example







14.1.71 REPLACE/String Replacement

REPLACE (P)

Replace part of the source string with the target string, and take the new string generated after replacement as the return value.

-[REPLACE (s1) (s2) (s3) (s4) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source string	-	STRING	-	STRING
Input s2	Replacement string:	-	STRING	-	STRING
Input s3	Replacement length	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Input s4	Replacement position	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Output d	Output string	-	STRING	-	STRING

Device used

Instruction	Parameter	Devices																						Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
REPLACE	Input s1																										
	Input s2																										
	Input s3																										
	Input s4																										
	Output d																										

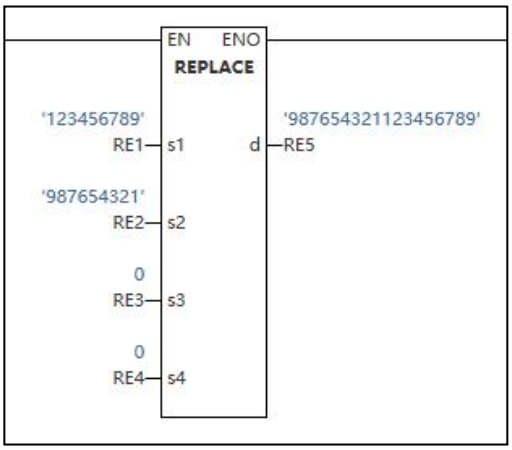
Function

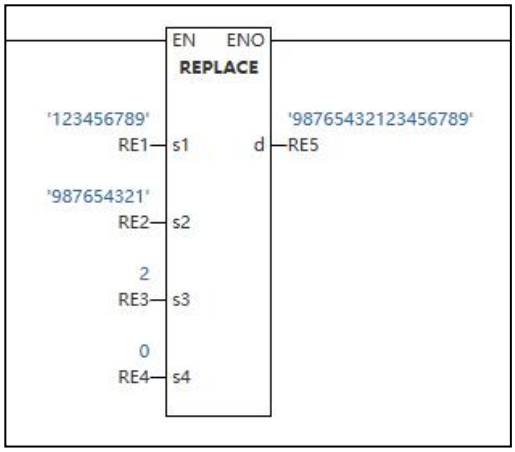
- Replace part of the source string with the target string, and take the new string generated after replacement as the return value.

Precautions

- If the actual length of the replaced string is greater than 80, only the first 80 characters will be returned and report error.
- S4 represents the replacement position, the position of the first character on the leftmost side of the string is 1, and the subsequent byte is added with 1;
- The length of S4 is greater than source string length, returns the source string.

Example





14.1.72 MID/Fetch arbitrarily from a string

MID (P)

Returns a string of the specified length starting at the position specified by the source string

-[MID (s1) (s2) (s3) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source String	-	STRING	-	STRING
Input s2	Take data length	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Input s3	Take data location	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT
Output d	Output string	-	STRING	-	STRING

Device used

Instruction	Parameter	Devices																							Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
MID	Input s1																									
	Input s2																									
	Input s3																									
	Output d																									

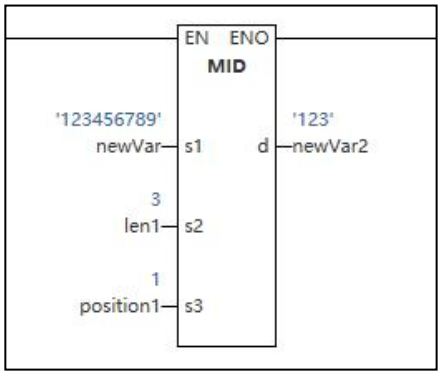
Function

Returns a string of the specified length, starting at the position specified by the source string

Precautions

- If the input S3 is 0, the output string is empty;
- If the input S3 is greater than the length of the input string, the source string is output.

Example



14.1.73 CONCAT/Concatenate two input strings and output them

CONCAT (P)

Concatenate two input strings and output them

-[CONCAT (s1) (s2) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source string 1	-	STRING	-	STRING
Input s2	Source string 2	-	STRING	-	STRING
Output d	Output string	-	STRING	-	STRING

Device used

Instruction	Parameter	Devices																							Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CONCAT	Input s1																									
	Input s2																									
	Output d																									

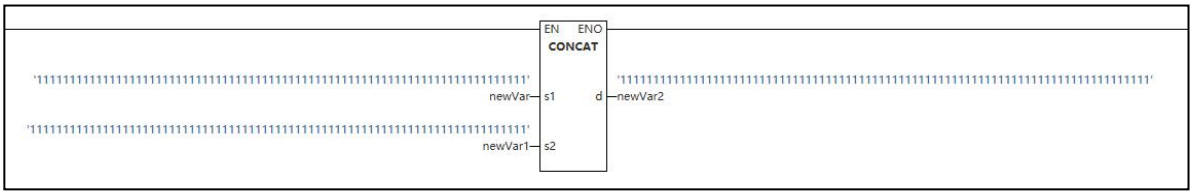
Function

- Concatenate the input two strings and output them.

Precautions

- When the sum of the actual lengths of S1 and S2 strings is greater than 80, only the output string 80 size can be returned.

Example



If the length of S1+S2 is greater than 80, the first 80 bits will be intercepted, and an error will be reported but it will not affect the result.

14.1.74 LEN/get string length

LEN(P)

Read the length of the string

-[LEN (s1) (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
Input s1	Source string 1	-	STRING	-	STRING
Output d	Output length	[0 to 255]	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT	-	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, UDINT, LINT

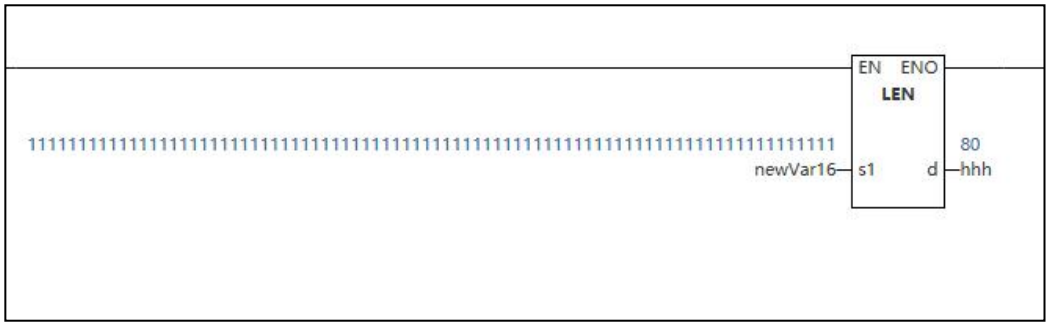
Device used

Instruction	Parameter	Devices																							Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
LEN	Input s1																									
	Output d																									

Function

- The length of the read string.

Example



14.1.75 STRING_TO_<TYPE>/STRING type conversion instruction

STRING_TO_<TYPE>

Function

- Convert the STRING data type to another data type.

Device used (only a single device can be used, and device splicing and offset are not supported)

Command	Parameter	Device																								Offset Modification	Pulse Expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
STRING_TO_BYTE	Input s1																										
	Output d																										
STRING_TO_DATE	Input s1																										
	Output d																										
STRING_TO_BOOL	Input s1																										
	Output d		•	•	•	•																					
STRING_TO_DT	Input s1																										
	Output d																										
STRING_TO_DWORD	Input s1																										
	Output d																										
STRING_TO_INT	Input s1																										
	Output d															•	•	•	•								
STRING_TO_REAL	Input s1																										
	Output d																										
STRING_TO_SINT	Input s1																										
	Output d																										
STRING_TO_TIME	Input s1																										
	Output d																										
STRING_TO_TOD	Input s1																										
	Output d																										
STRING_TO_UDINT	Input s1																										
	Output d																										
STRING_TO_UINT	Input s1																										
	Output d																										
STRING_TO_USINT	Input s1																										
	Output d																										
STRING_TO_WORD	Input s1																										
	Output d																										
STRING_TO_LREAL	Input s1																										
	Output d																										
STRING_TO_LWORD	Input s1																										

Command	Parameter	Device																						Offset Modification	Pulse Expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
	Output d																										
STRING_TO_LINT	Input s1																										
	Output d																										
STRING_TO_LTIME	Input s1																										
	Output d																										
STRING_TO_DINT	Input s1																										
	Output d																				•	•					

Precautions for use of devices

- LC and HSC are signed 32-bit data types;
- T, C, D, R and SD are signed 16-bit data types.

Variable Type Used

Instruction	Parameter	Variable Type	Instruction	Parameter	Variable Type
STRING_TO_BYTE	Input s1	STRING	STRING_TO_TOD	Input s1	STRING
	Output d	BYTE		Output d	TIME_OF_DAY
STRING_TO_DATE	Input s1	STRING	STRING_TO_UDINT	Input s1	STRING
	Output d	DATE		Output d	UDINT
STRING_TO_BOOL	Input s1	STRING	STRING_TO_UINT	Input s1	STRING
	Output d	BOOL		Output d	UINT
STRING_TO_DT	Input s1	STRING	STRING_TO_USINT	Input s1	STRING
	Output d	DATE_AND_TIME		Output d	USINT
STRING_TO_DWORD	Input s1	STRING	STRING_TO_WORD	Input s1	STRING
	Output d	DWORD		Output d	WORD
STRING_TO_INT	Input s1	STRING	STRING_TO_LREAL	Input s1	STRING
	Output d	INT		Output d	LREAL
STRING_TO_REAL	Input s1	STRING	STRING_TO_LWORD	Input s1	STRING
	Output d	REAL		Output d	LWORD
STRING_TO_SINT	Input s1	STRING	STRING_TO_LINT	Input s1	STRING
	Output d	SINT		Output d	LINT
STRING_TO_TIME	Input s1	STRING	STRING_TO_LTIME	Input s1	STRING
	Output d	TIME		Output d	LTIME

Instruction

REAL/LREAL,don't support base conversion and defaults to decimal, for 16#, 2#, 8#, only recognize the previous numbers.

① STRING_TO_DATE

The input format is date#1988-5-1 or DATE#1988-5-1. If the input format is wrong or the date does not conform to the specification, return 1970-1-1;

Format: Date#Y-M-D.

② STRING_TO_DT

The input format is DT#1988-5-1-12:12:12, blue font can be omitted;

If the input format is wrong or the date does not conform to the specification, returns 1970-1-1-0: 0: 0;

The input must contain red font to be valid.

③ STRING_TO_TIME

The input format is TIME#123ms or TIME#8d7h6m1s8ms;

For example, 'Time#152asdhad10d9ss8madi7dd' will be detected as 17d9s8m and converted to ms for output;

The input must contain red font to be valid and must have a number followed by a time unit to be recognized.

④ STRING_TO_LTIME

The input format is LTIME #123ns or LTIME#8d7h6m1s8ms8us9ns converted to ns output;

For example, 'TIME#152asdhad10d9ss8msa7di'' will be detected as 17d9s8m and converted to ns for output;

The input must contain red font to be valid and must have a number followed by a time unit to be recognized.

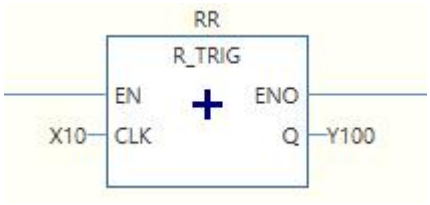
⑤ STRING_TO_TOD

Input format: TOD 12:12:12 The input must comply with the time specification, and the input must be identified;

The input must contain red font to be valid.

14.2 Built-in function block

14.2.1 R_TRIG/rising edge test



Rising edge test

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
R_TRIG	CLK	●	●	●	●	●																				
	Q		●	●	●	●																				

Precautions for use of devices

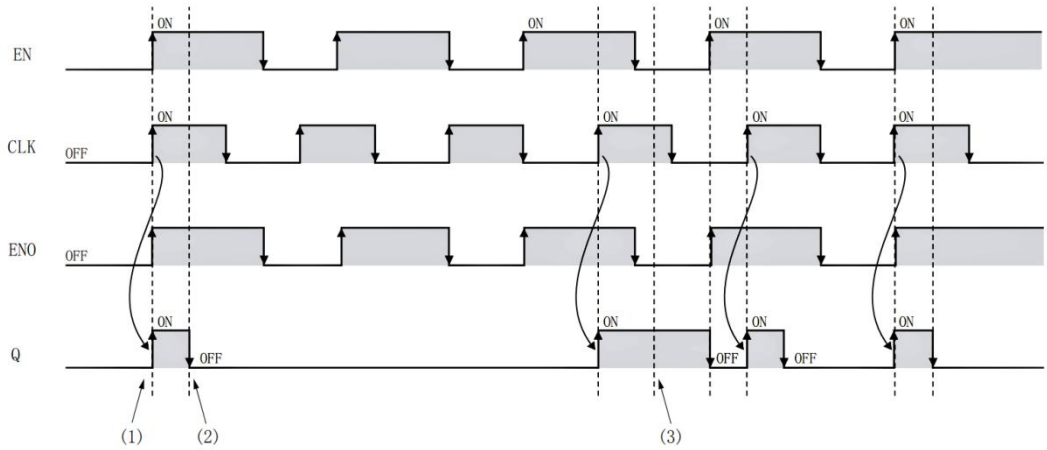
- X, Y, M, S and SM are BOOL type;
- X cannot be used as output.

Variable type used

Instruction	Parameter	Variable type
R_TRIG	CLK	BOOL
	Q	BOOL

Function

- When (CLK) is turned to ON, (Q) can be set as ON through only one scan.
- Timing diagram

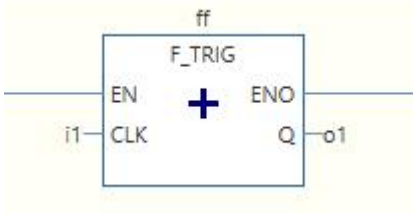


(1): When EN=ON and reaching the rising edge of (CLK), (Q) is set as ON;

(2): During next scan, (Q) is set as OFF;

(3): When EN=OFF, (Q) maintains the output of previous scan.

14.2.2 F_TRIG/falling Edge Test



F_TRIGfalling edge test

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
F_TRIG	CLK	●	●	●	●	●																				
	Q		●	●	●	●																				

Precautions for use of devices

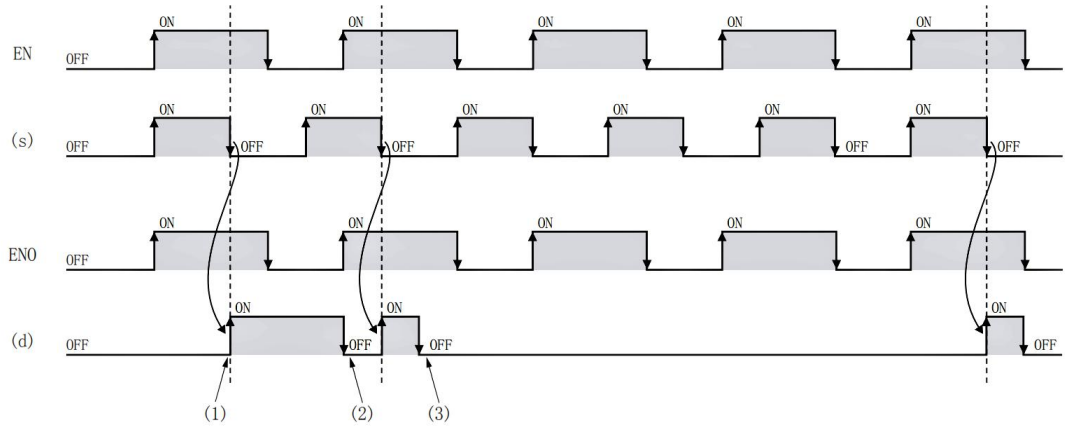
- X, Y, M, S and SM are BOOL type;
- X cannot be used as output.

Variable Type Used

Instruction	Parameter	Variable type
F_TRIG	CLK	BOOL
	Q	BOOL

Function

- When (CLK) is turned to OFF, (Q) can be set as ON through only one scan.
- Timing diagram

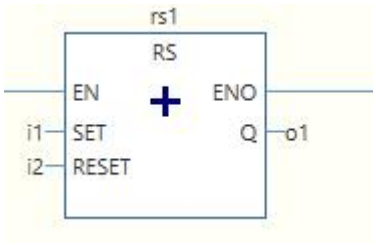


(1): When EN=ON and reaching the falling edge of (CLK), (Q) is set as ON;

(2): During next scan, (Q) is set as OFF;

(3): When EN=OFF, (Q) maintains the output of previous scan.

14.2.3 RS/Bistable function block (Reset First)



Bistable function block with reset first.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RS	SET	●	●	●	●	●																				
	RESET	●	●	●	●	●																				
	Q		●	●	●	●																				

Precautions for use of devices

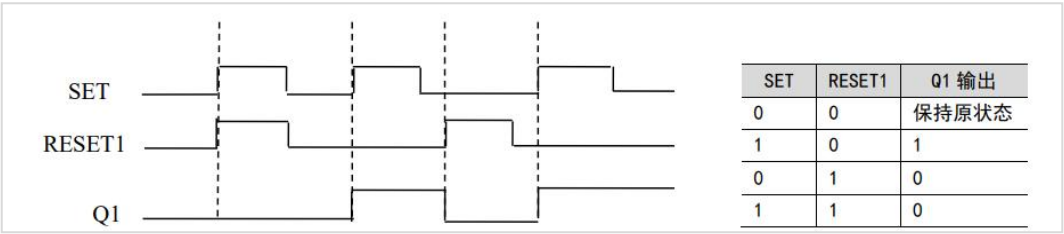
- X, Y, M, S and SM are BOOL type;
- X cannot be used as output.

Variable Type Used

Instruction	Parameter	Variable type
RS	SET	BOOL
	RESET	BOOL
	Q	BOOL

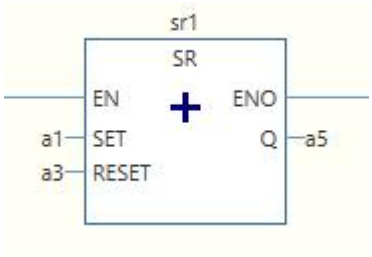
Function

- When RESET is “TRUE”, regardless of the value of SET, the output of Q is “FALSE”; when RESET is “FALSE” and when SET is “TRUE”, the output of Q is “TRUE”; when RESET and SET are “FALSE”, Q maintains its original state.



SET	RESET1	Q1 输出
0	0	保持原状态
1	0	1
0	1	0
1	1	0

14.2.4 SR/Bistable function block (Setting First)



Bistable function block with setting first.

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SR	SET	●	●	●	●	●																					
	RESET	●	●	●	●	●																					
	Q		●	●	●	●																					

Precautions for use of devices

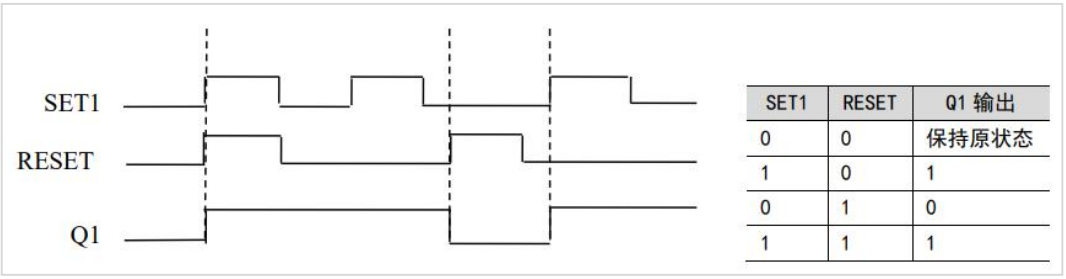
- X, Y, M, S and SM are BOOL type;
- X cannot be used as output.

Variable Type Used

Instruction	Parameter	Variable type
SR	SET	BOOL
	RESET	BOOL
	Q	BOOL

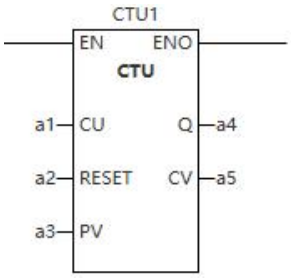
Function

- When SET is “TRUE”, regardless of the value of RESET, the output of Q is “TRUE”; when SET is “FALSE” and when RESET is “TRUE”, the output of Q is “FALSE”; when RESET and SET are “FALSE”, Q maintains its original state.



SET1	RESET	Q1 输出
0	0	保持原状态
1	0	1
0	1	0
1	1	1

14.2.5 CTU/Count Up



Count Up

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CTU	CU	●	●	●	●	●																				
	RESET	●	●	●	●	●																				
	PV															●	●	●	●	●						
	Q		●	●	●	●																				
	CV															●	●	●	●	●						

Precautions for use of devices

- X, Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types;
- X cannot be used as output.

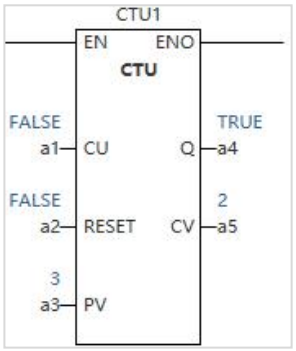
Instruction	Parameter	Variable type
CTU	CU	BOOL
	RESET	BOOL
	PV	WORD
	Q	BOOL
	CV	WORD

Variable Type Used

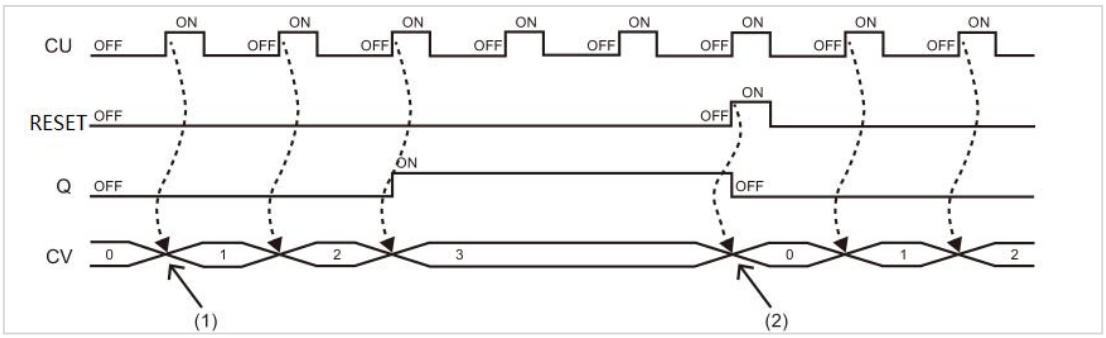
Function

- If (CU) turns from OFF to ON, (CV) is up counted (+1);
- If (CV) is greater than or equal to (PV) of the counter, (Q) becomes ON, and up counting continues until the maximum value 32767 is added; then stop counting;
- (PV) sets the target value of the counter. If (RESET) is set as ON, (CV) is set as 0.

  For example: PV is set as 3



Timing chart:



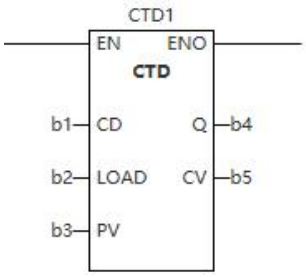
(1) On the rising edge of CU, CV is up counted;

(2) When RESET is set as ON, CV is reset.

Precautions

- Range of the counter: 0~32767;
- If the value of PV is set beyond the range of the counter, the output of Q will always be ON.

14.2.6 CTD/Count Down



Count Down

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CTD	CD	●	●	●	●	●																					
	LOAD	●	●	●	●	●																					
	PV															●	●	●	●	●							
	Q		●	●	●	●																					
	CV															●	●	●	●	●							

Precautions for use of devices

- X, Y, M, S and SM are BOOL type;
- T, C, D, R and SD are signed 16-bit data types;
- LC and HSC are signed 32-bit data types;
- X cannot be used as output.

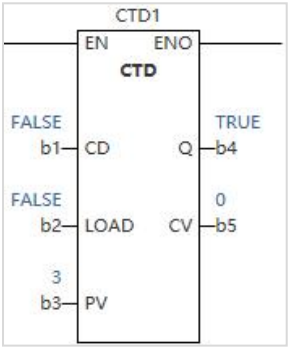
Variable Type Used

Instruction	Parameter	Variable type
CTD	CD	BOOL
	LOAD	BOOL
	PV	WORD
	Q	BOOL
	CV	WORD

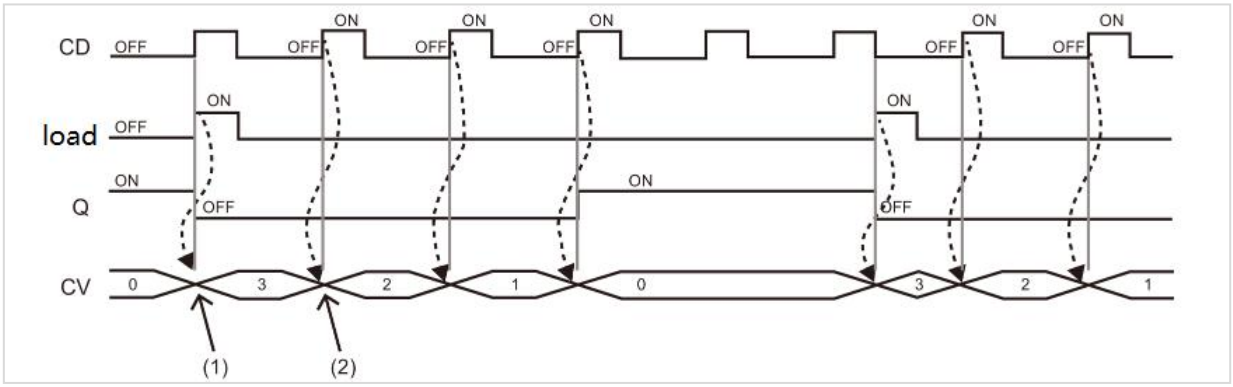
Function

- If (CD) turns from OFF to ON, (CV) is down counted (-1);
- If (CV) is equal to 0, (Q) becomes ON; then stop counting;
- (PV) sets the initial value of the counter. If (LOAD) is set as ON, (CV) is set as the value of (PV).

  For example: PV is set as 3



Timing chart:



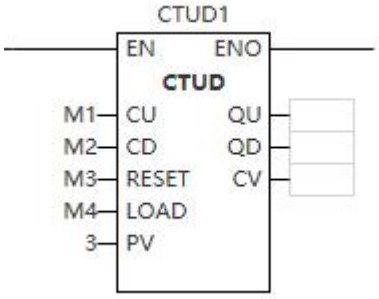
(1) When LOAD is set as ON, PV value is assigned to CV;

(2) When CD is on the rising edge, CV is down counted;

Precautions

- Range of the counter: 0~32767;
- If the value of CV is set beyond the range of the counter, the output of Q will always be ON;
- If the value of CV is set beyond the range of the counter, the count is invalid.

14.2.7 CTUD/Count Up and Down



Count Up and Down

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
CTUD	CU	●	●	●	●	●																					
	CD	●	●	●	●	●																					
	RESET	●	●	●	●	●																					
	LOAD	●	●	●	●	●																					
	PV															●	●	●	●	●							
	QU		●	●	●	●																					
	QD		●	●	●	●																					
	CV															●	●	●	●	●							

Precautions for use of devices

- T, C, D, R and SD are signed 16-bit data types;
- X, Y, M, S and SM are BOOL type;
- LC and HSC are signed 32-bit data types;
- X cannot be used as output.

Variable Type Used

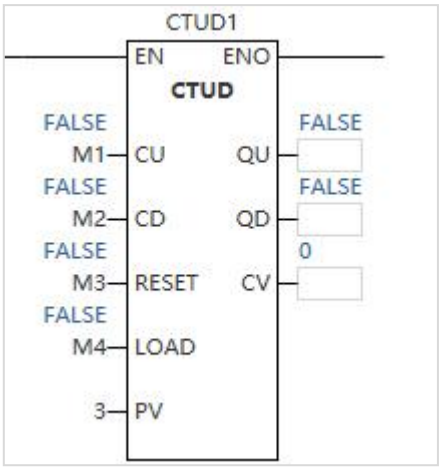
Instruction	Parameter	Variable type
CTUD	CU	BOOL
	CD	BOOL
	RESET	BOOL
	LOAD	BOOL
	PV	WORD
	QU	BOOL
	QD	BOOL
	CV	WORD

Function

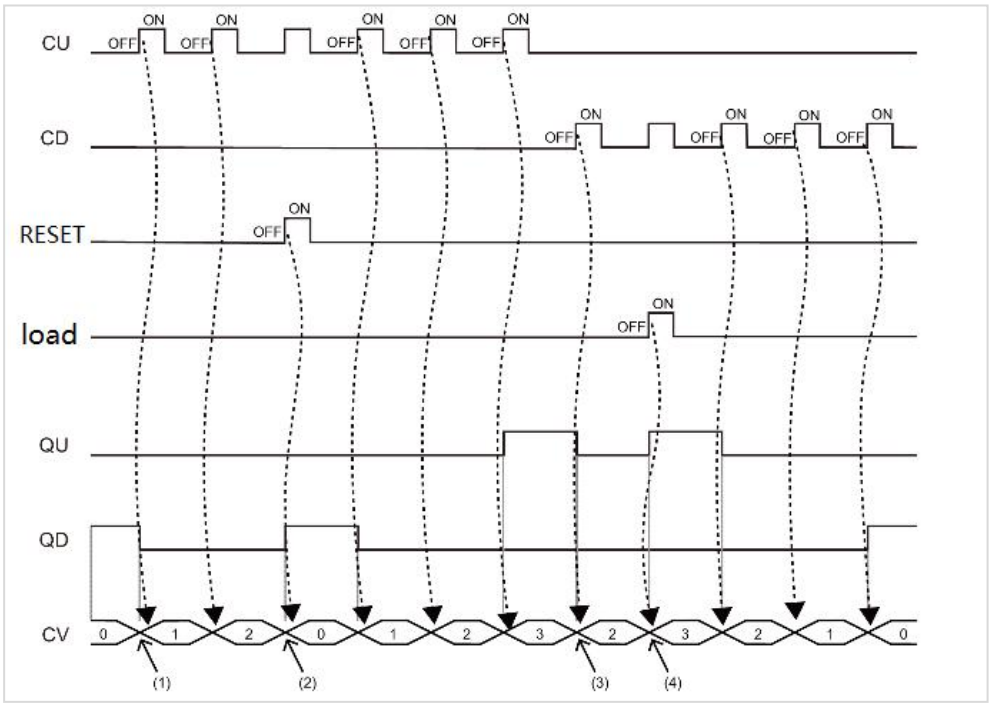
- If (CU) turns from OFF to ON, (CV) is up counted (+1);
- If (CV) is greater than or equal to (PV) of the counter, (QU) becomes ON, and up counting continues until the maximum value 32767 is added; then stop counting;
- (PV) sets the initial value of the counter. If (RESET) is set as ON, (CV) is set as 0;
- If (CD) turns from OFF to ON, (CV) is down counted (-1);

- If (CV) is equal to 0, (QD) becomes ON; then stop down counting;
- (PV) sets the target value of the counter. If (LOAD) is set as ON, (CV) is set as the value of (PV).

For example: PV is set as 3



Timing chart:



(1) On the rising edge of CU, CV is up counted;

(2) When RESET is set as ON, CV is reset;

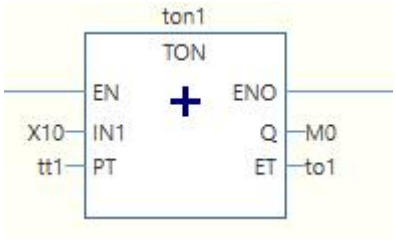
(3) On the rising edge of CD, CV is down counted;

(4) When LOAD is set as ON, PV value is assigned to CV.

Precautions

- Range of the counter: 0~32767;
- If the value of PV is set beyond the range of the counter, the output of QU will always be ON.
- If the value of CV is set beyond the range of the counter, the count up/down is invalid and the output of QD is ON.

14.2.8 TON/ON Delay Timer



ON delay timer

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TON	IN1	●	●	●	●	●																				
	PT																									
	Q		●	●	●	●																				
	ET																									

Precautions for use of devices

- X, Y, M, S and SM are BOOL type;
- X cannot be used as output.

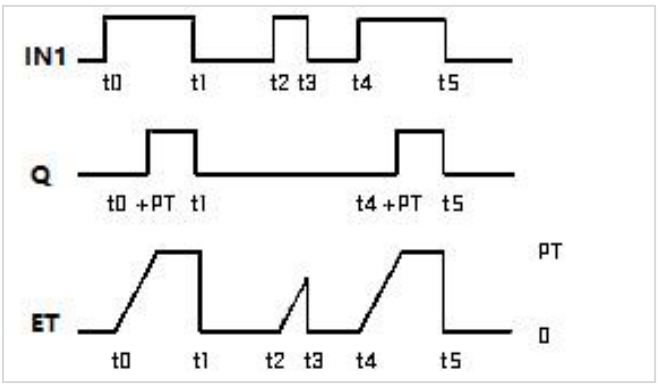
Variable Type Used

Instruction	Parameter	Variable type
TON	IN1	BOOL
	PT	TIME
	Q	BOOL
	ET	TIME

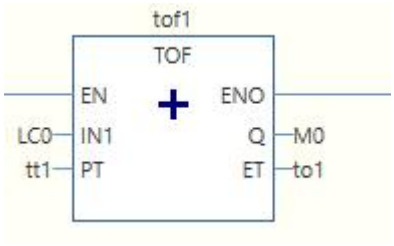
Function

- When IN1 turns from “FALSE” to “TRUE”, the timer starts. When it reaches the specified time of PT and during such period, IN1 is always under “TRUE” state, the output Q is set as “TRUE”. If the state of IN1 changes to “FALSE” during the period, the timer is reset, waiting for the next rising edge signal of IN1, and then the timer restarts;
- ET is the time elapsed after the timer is started, and the delay starts from 0 to the end of the time set by PT, and then keeps ET until IN1 changes to “FALSE” state, and ET will be reset as 0.

Timing diagram



14.2.9 TOF/OFF Delay Timer



OFF delay timer

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TOF	IN1	●	●	●	●	●																					
	PT																										
	Q		●	●	●	●																					
	ET																										

Precautions for use of devices

- X, Y, M, S and SM are BOOL type;
- X cannot be used as output.

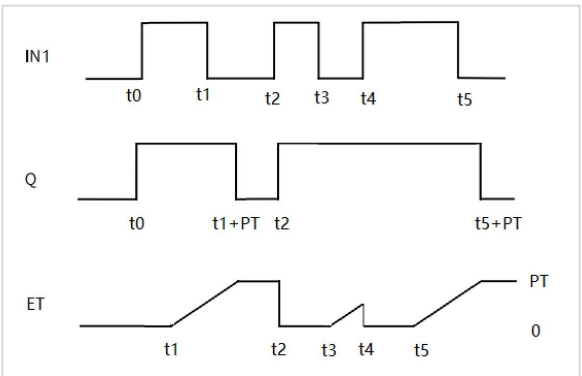
Variable Type Used

Instruction	Parameter	Variable type
TOF	IN1	BOOL
	PT	TIME
	Q	BOOL
	ET	TIME

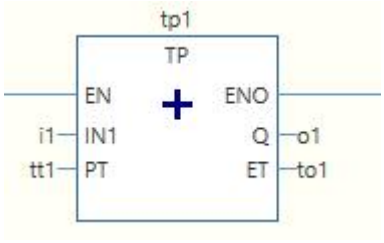
Function

- IN1 changes from “FALSE” to “TRUE”, the output signal of timer Q is “TRUE”, the starting input of the timer changes to “FALSE” and the timer starts; as long as the timer is running, its output Q will keep the state of “TRUE”; when reaching the specified time, the output Q is reset; before reaching the time of the timer, if the input IN1 of the timer is changed to “TRUE” state, the timer ET is reset and the output Q keeps “TRUE” state;
- ET is the time elapsed after the timer is started, and the delay starts from 0 to the end of the time set by PT, and then ET is kept until IN1 changes to “TRUE” state, and ET will be reset as 0. Before reaching the time set by PT, IN1 changes to “TRUE” and the output ET will be reset immediately.

Timing diagram



14.2.10 TP/Pulse Timer



Pulse timer

Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
TP	IN1	●	●	●	●	●																				
	PT																									
	Q		●	●	●	●																				
	ET																									

Precautions for use of devices

- X, Y, M, S and SM are BOOL type;
- X cannot be used as output.

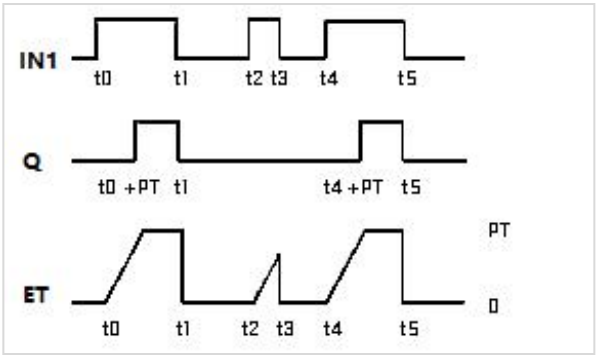
Variable Type Used

Instruction	Parameter	Variable type
TP	IN1	BOOL
	PT	TIME
	Q	BOOL
	ET	TIME

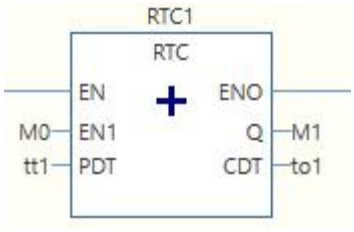
Function

- When IN1 changes from “FALSE” to “TRUE”, the timer starts; no matter how the timer input end IN1 changes, the actual running time of the timer is the PT defined by the client; when the timer is running, the output signal of the output end Q is “TRUE”;
- The output end ET provides timing time for the output end Q. The timing time starts at 0 and ends at PT. When reaching PT, ET will keep the timing time until IN1 becomes “FALSE”.

Timing diagram



14.2.11 RTC/Real Time Clock



Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
RTC	EN1	●	●	●	●	●																				
	PDT																									
	Q		●	●	●	●																				
	CDT																									

Precautions for use of devices

- X, Y, M, S and SM are BOOL type;
- X cannot be used as output.

Variable Type Used

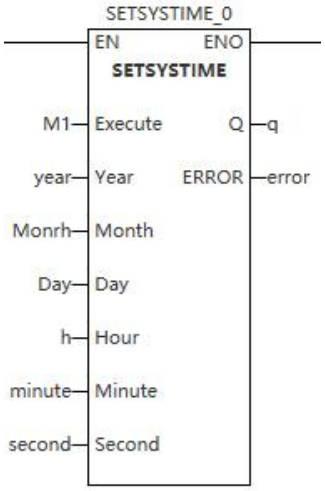
Instruction	Parameter	Variable type
RTC	EN1	BOOL
	PDT	DATE_AND_TIME
	Q	BOOL
	CDT	DATE_AND_TIME

Function

- When EN1 is “FALSE”, the output Q and CDT are both “0” and the relevant time is DT#1970-1-1-00:00:00. Once EN1 is changed to “TRUE”, it starts from the time set by PDT and counts in seconds and returns from CDT. When EN is reset to FALSE, CDT will be reset to DT#1970-1-1-00:00:00.

Precautions: Only the rising edge triggers the start counting.

14.2.12 SETSYSTIME/Set clock



Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																						Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SETSYSTIME	Execute	●	●	●	●	●																				
	Year																									
	Month																									
	Day																									
	Hour																									
	Minute																									
	Second																									
	Q		●	●	●	●																				
	ERROR		●	●	●	●																				

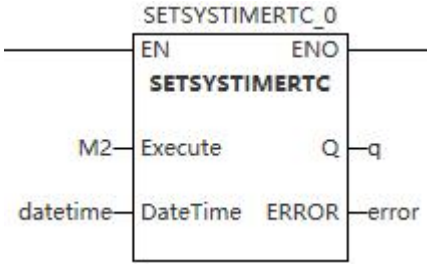
Precautions for use of devices

- LC, HSC are signed 32-bit data types;
- T, C, D, R, SD are signed 16-bit data type.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
SETSYSTIME	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		Year	WORD	No	[0,65535]	Year
		Month	WORD	No	[0,65535]	Month
		Day	WORD	No	[0,65535]	Day
		Hour	WORD	Yes	[0,65535]	Hour
		Minute	WORD	Yes	[0,65535]	Minute
		Second	WORD	Yes	[0,65535]	Second
	Output	Q	BOOL	Yes	TRUE/FALSE	Complete
		ERROR	BOOL	Yes	TRUE/FALSE	Error

14.2.13 SETSYSTIMERTC/Set clock RTC



Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																								Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SETSYSTIMERTC	Execute	●	●	●	●	●																					
	DateTime																										
	Q		●	●	●	●																					
	ERROR		●	●	●	●																					

Precautions for use of devices

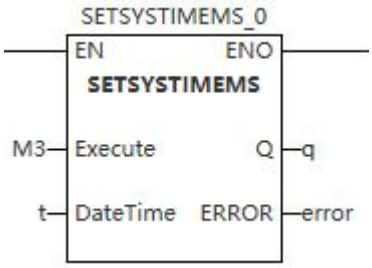
▪LC, HSC are signed 32-bit data types;

▪T, C, D, R, SD are signed 16-bit data type.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
SETSYSTIMERTC	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		DateTime	DATE_AND_TIME	No	1970-01-01-0:0:0 to 2106-02-07-06:28:15	Set Date
	Output	Q	BOOL	Yes	TRUE/FALSE	Complete
		ERROR	BOOL	Yes	TRUE/FALSE	Error

14.2.14 SETSYSTIMEMS/Set clock MS



Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
SETSYSTIMEMS	Execute	●	●	●	●	●																				
	DateTime																									
	Q		●	●	●	●																				
	ERROR		●	●	●	●																				

Precautions for use of devices

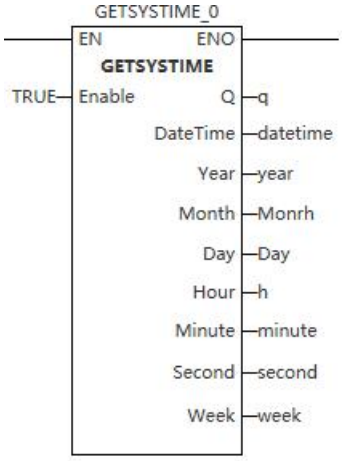
▪LC, HSC are signed 32-bit data types;

▪T, C, D, R, SD are signed 16-bit data type.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
SETSYSTIMEMS	Input	Execute	BOOL	No	TRUE/FALSE	Enable
		DateTime	DWORD	No	[0,4294927695]	Set millisecond timestamp
	Output	Q	BOOL	Yes	TRUE/FALSE	Complete
		ERROR	BOOL	Yes	TRUE/FALSE	Error

14.2.15 GETSYSTIME/Get clock



Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
GETSYSTIME	Enable	●	●	●	●	●																				
	Q		●	●	●	●																				
	Year																									
	Month																									
	Day																									
	Hour																									
	Minute																									
	Second																									
	DateTime																									
	Week																									

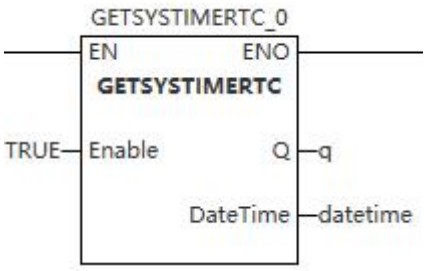
Precautions for use of devices

- LC, HSC are signed 32-bit data types;
- T, C, D, R, SD are signed 16-bit data type.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
GETSYSTIME	Input	Enable	BOOL	No	TRUE/FALSE	Enable
	Output	Year	WORD	Yes	[0,65535]	Year
		Month	WORD	Yes	[0,65535]	Month
		Day	WORD	Yes	[0,65535]	Day
		Hour	WORD	Yes	[0,65535]	Hour
		Minute	WORD	Yes	[0,65535]	Minute
		Second	WORD	Yes	[0,65535]	Second
		Q	BOOL	Yes	TRUE/FALSE	Complete
		DateTime	DATE_AND_TIME	Yes	1970-01-01-0:0:0 to 2106-02-07-06:28:15	Date
		Week	WORD	Yes	[0,65535]	Week

14.2.16 GETSYSTIMERTC/Get clock RTC



Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
GETSYSTIMERTC	Enable	●	●	●	●	●																				
	Q			●	●	●	●																				
	DateTime																									

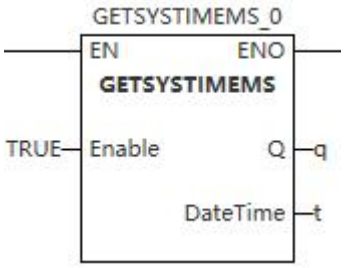
Precautions for use of devices

- LC, HSC are signed 32-bit data types;
- T, C, D, R, SD are signed 16-bit data type.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
GETSYSTIMERTC	Input	Enable	BOOL	No	TRUE/FALSE	Enable
	Output	Q	BOOL	Yes	TRUE/FALSE	Complete
		DateTime	DATE_AND_TIME	Yes	1970-01-01-0:0:0~2106-02-07-06:28:15	Date

14.2.17 GETSYSTIMEMS/Get clock MS



Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
GETSYSTIMEMS	Enable	●	●	●	●	●																				
	Q		●	●	●	●																				
	DateTime																									

Precautions for use of devices

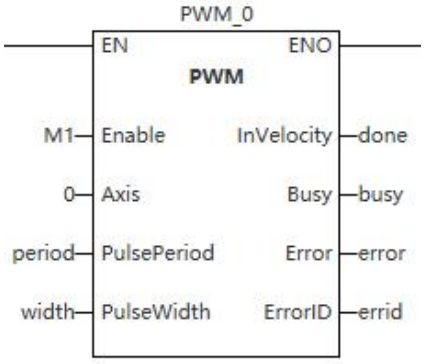
- LC, HSC are signed 32-bit data types;
- T, C, D, R, SD are signed 16-bit data type.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
GETSYSTIMEMS	Input	Enable	BOOL	No	TRUE/FALSE	Enable
	Output	Q	BOOL	Yes	TRUE/FALSE	Complete
		DateTime	DWORD	Yes	[0,4294967295]	Set millisecond timestamp

✎**Note**: An error code 4084H will be reported during operation if the input time is outside the range from 00:00:00 on January 1, 2000 to 23:59:59 on December 31, 2099.

14.2.18 PWM/Pulse width limit



Device used (only a single device can be used, and device splicing and offset are not supported)

Instruction	Parameter	Devices																							Index modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
PWM	Enable	●	●	●	●	●																				
	Axis																									
	PulsePeriod																									
	PulseWidth																									
	InVelocity		●	●	●	●																				
	Bysy		●	●	●	●																				
	Error		●	●	●	●																				
	ErrorId																									

Precautions for use of devices

- LC, HSC are signed 32-bit data types;
- T, C, D, R, SD are signed 16-bit data type.

Variable type used

Instruction	Pin type	Parameter	Variable type	Can be empty or not	Range	Description
PWM	Input	Enable	BOOL	No	TRUE/FALSE	Enable
		Axis	WORD	No	[0,65535]	Axis number
		PulsePeriod	WORD	No	[0,65535]	Period: Width of a complete square wave. Unit: 1μs
		PulseWidth	WORD	No	[0,65535]	High-level width of the square wave. Unit: 1μs
	输出	InVelocity	BOOL	Yes	TRUE/FALSE	Pulse width frequency reaches the set value
		Busy	BOOL	Yes	TRUE/FALSE	Executes PWM square wave output.
		Error	BOOL	Yes	TRUE/FALSE	TRUE: An internal error occurred in the function block.
		ErrorId	DWORD	Yes	0	Error ID. For details, refer to MC_ERROR.

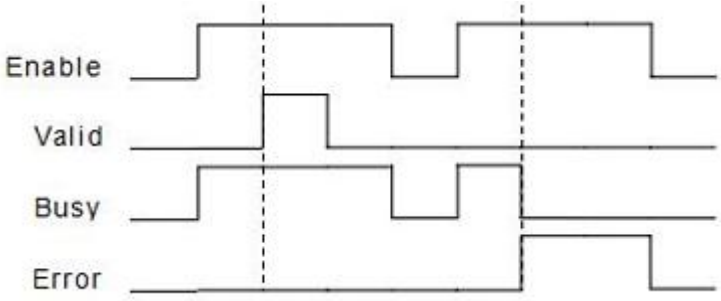
Function

(1)This function block implements dynamic pulse width modulation (PWM).

(2)During the execution of the MC_PWM function block, no other PWM commands are allowed to perform PWM output on the

same axis.

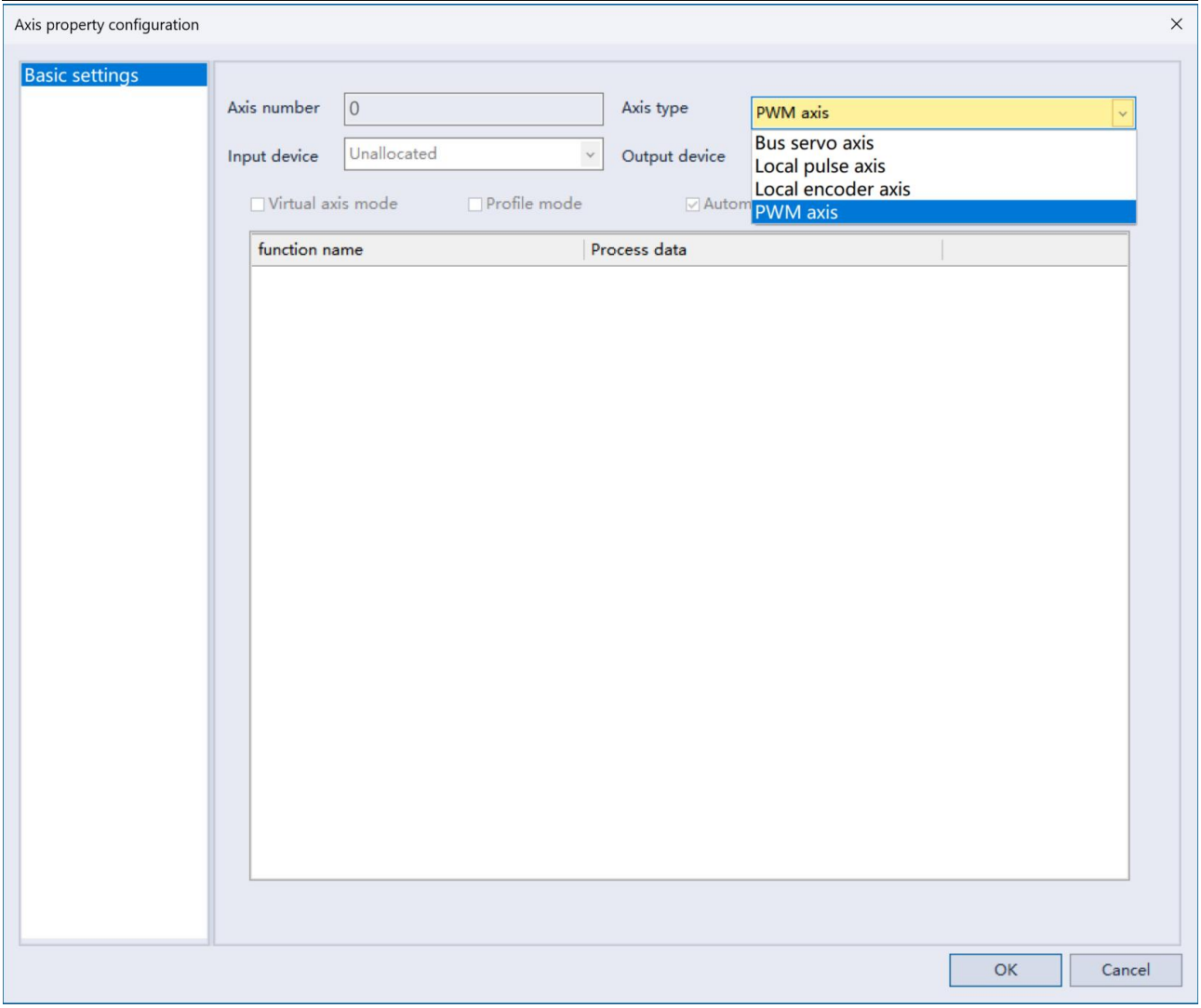
Sequence diagram



Error description:

Error code	Initial value	Description
0	MC_NO_ERROR	No error
0x00800024	MC_PWM_PORT_WAS_RUN	The corresponding port is currently running
0x00800023	MC_PWM_PERIOD_LESS_THAN_WIDTH	The pulse period is less than the high-level pulse width
0x00700005	MC_ERROR_ID_AXIS_NOTFIND	The axis does not exist

In the same axis configuration, only one of PWM mode or pulse mode can be selected; they cannot be used simultaneously. Ensure that the configured axis number is consistent with the created axis configuration.


Axis property configuration
Basic settings
Axis number
0
Axis type
PWM axis
Bus servo axis
Local pulse axis
Local encoder axis
PWM axis
Input device
Unallocated
Output device
Virtual axis mode
Profile mode
function name
Process data
OK
Cancel

15 ST Help manual

15.1 Introduction

Structured text (ST) is a high-level text language, it can be used to define the logical behavior of functions, function blocks and programs, it also can also describe the behavior of steps, actions and transitions in sequential function flow diagram.

Structure text is similar to Pascal. It is a language specially developed for industrial control applications. For personnel familiar with high-level computer programming languages, structured text is exceptionally easy to learn and use. It supports functionalities such as selection, iteration, and jump statements. In addition, structured text languages are easy to read and understand, especially when annotated with meaningful identifiers and annotations. In complex control systems, structured text can greatly reduce the amount of code and make complex system problems simple. The disadvantages are that debugging is not intuitive and compilation speed is relatively slow.

15.2 Structured text program execution sequence

15.2.1 Program execution sequence

The sequence of execution of programs using structured text is executed sequentially from top to bottom according to the "line number".

```
1  a+b+c;
2  3.14*R*R;
3  ABS(D0,var1);
```

15.2.2 Expression execution order

The expression includes operators and operands. The operands are operated according to the rules specified by the operators, and the results are obtained and returned. The operands can be variables, constants, device, functions, etc.

If there are several operators in the expression, the operators are executed in the agreed order of priority: The high-priority operator operations are executed first, and then the low-priority operator operations are executed sequentially. If there are operators with the same priority in the expression, these operators are executed from left to right in writing order.

15.2.3 Operators

Structured text supports the operators listed in the table below.

Priority	Name	Operator
1	Brackets	(expression)
2	Function/Function Block Call	Function name(parameter)
3	Retrieve Content	^
4	Negation/Bitwise NOT	-, NOT
5	Multiplication, Division, Modulo	*, /, MOD
6	Addition, Subtraction	+, -
7	Comparison	<, >, <=, >=
8	Equality/Inequality	=, <>
9	Bitwise AND	AND
10	Bitwise XOR	XOR
11	Bitwise OR	OR
12	Assignment	:=

15.3 Representation and declaration of variables

Variables can be used to represent a numeric value, a string value or an array, etc.

15.3.1 Variables

Variables are abstract data stored in memory to be processed. They are names used to identify the input/output of the PLC and the storage area inside the PLC. They can replace the use of physical addresses in programs.

The value of the data stored in the variable can be changed at any time as needed. During program execution, the value of a variable can change. Variables must be declared before using them, and the type and name of the variable must be specified. Variables have names, types and values. The data type of a variable determines the size and type of memory it represents. The variable name is the identifier in the program source code.

15.3.2 Identifiers

The identifier is the name of the variable. In defining the identifier, according to the IEC 61131-3 standard, it may consist of letters, numbers, Chinese characters and underscore characters. In addition, the following rules should be followed;

The initial letter of the identifier must be a letter or underline:

- The first character of an identifier must be a letter, a Chinese character, or an underscore.
- Identifiers are not case sensitive
- Shall not contain spaces.

For example ab_c, AB_de and _AbC are allowed identifiers, while 1abc is not allowed.

15.3.3 Variable declaration

A variable declaration is the name, type, and assignment of an initial value to a specified variable.

The declaration of variables is very important. Undeclared variables cannot be compiled, so they cannot be used in programs. Users can declare variables in the program organization unit (POU), global variable list (GVL) and automatic declaration dialog box.

The most commonly used variable declaration, variables that do not need to be associated with hardware peripherals or communication, is only used by the internal logic of the project. Ordinary declaration must subject to the following rules:

<Identifier>:<Data type>{:=<Initial value>}

{} is optional. For example: nTest: BOOL; nTest: BOOL: = TRUE;

15.4 Introduction to basic statements

The structured text statement table is shown in the following table.

Instruction Type	Instruction statement	Example
Assignment statement	:=	bFan:= TRUE;
Function block/function call	Function block/function name();	COS();

Select statement	IF	IF var1 = 1 THEN
		var1: = 2;
		END_IF

	CASE	CASE var1 OF 0: var1: = 10; 1: var1: = 20; 2: var1: = 30; END_CASE

Iteration statement	FOR	FOR var1: = 1 TO 10 DO
		var2: = var2 + 1;
		END_FOR

	WHILE	WHILE var1 = 2 DO
		var1: = 123;
		END_WHILE

	REPEAT	REPEAT
		var3: = 234;
		UNTIL var1 = 2
		END_REPEAT

Jump statement	EXIT	FOR var1: = 1 TO 10 DO
		var2: = var2 + 1;
		IF var1 = 5 THEN
		EXIT
		END_IF
		END_FOR

	CONTINUE	FOR var1: = 1 TO 10 DO
		var2: = var2 + 1;
		IF var1 = 5 THEN
		CONTINUE
		END_IF
		END_FOR

Return Statement	RETURN	IF var1 = 1 THEN
		RETURN 1;
		ELSE
		RETURN 0;
		END_IF

NULL statements	:	N/A

15.4.1 Assignment statement

Assignment statements are one of the most commonly used statements in structured text. Their function is to assign the value generated by the expression on the right side to the operand (variable or address) on the left side, using ":= "represents. The specific format is:

<Variable> :=<Expressions>;

For example: Assign values to two Boolean variables respectively, set TRUE for bFan and set FALSE for bHeater.

```
VAR
    bFan:BOOL;
    bHeater:BOOL;
END_VAR
```

```
bFan:=TRUE;
bHeater:=FALSE;
```

15.4.2 Functions and function block calls

Function control statement

After the function is called, the return value is directly assigned to the variable as the value of the expression. For example, in the "rVar1: = SIN (rData1);" statement, the sine function SIN is called and the return value is assigned to the variable rVar1. The function control statement format is as follows: Variable = function name (parameter table);

Example of function control statement:

```
rResult:=ADD(rData1,rData2);//Call ADD FC and assign the sum of two data to rResult
```

Function block control statement

In the function block call, the function block name is instantiated to implement the call, for example, Timer is the instance name of the TON function block. The specific format is:

Function block instance name (function block parameter);

If a function block needs to be called in ST, the instance name of the function block can be entered directly, and each parameter of the function block is assigned a value or variable in subsequent brackets, with the parameters separated by commas; function block calls end with semicolons.

For example, if the function block TON timer is called in structured text, assuming its instance name is TON1, the specific implementation is as follows:

```
VAR
    TON1:TON;
END_VAR
```

```
TON1(IN1:=,PT:=,Q=>,ET=>);
```

15.4.3 Select statements

IF statement

Use "IF statement" to implement single-branch selection structure. The basic format is as follows:

IF<Boolean expression>THEN

<Statement Content>;

END_IF

When use the above format, only if <Boolean expression> is TRUE, the statement content will be executed; Otherwise do not execute the IF statement <Statement Content>。

The content of the statement can be one statement or an empty statement, or multiple statements can be arranged in parallel.

Example

For example: Use PLC to determine whether the current temperature exceeds 60 degrees Celsius. If it exceeds, always turn on the fan for heat dissipation. The specific implementation code is as follows:

```
VAR
    nTemp: BYTE; //Current temperature
    bFAN: BOOL; //Control signal of fan
END_VAR
```

```
IF nTemp>60 THEN
    bFAN:= TRUE;
END_IF
```

IF.... ELSE statement

Use the IF....ELSE statement to implement the double-branch selection mechanism. The basic format is as follows:

IF<Boolean expression>THEN

<Statement Content 1>;

ELSE

<Statement Content 2>;

END_IF

The above expression first determines the value in <Boolean expression>. If it is TRUE, <statement content 1> is executed; if it is FALSE, <statement content 2> is executed.

For example: Use PLC to judge that when the temperature is less than 20 degrees Celsius, turn on the heating equipment, otherwise (the temperature is greater than or equal to 20 degrees Celsius) the heating equipment is disconnected.

```
VAR
    nTemp: BYTE; //Current temperature
    bHeating: BOOL; //Control signal of heater
END_VAR
```

```
IF nTemp<20 THEN
    bHeating:= TRUE;
    ELSE
    bHeating:= FALSE;
END_IF
```

When the program has more than one conditional judgment expression, at this time, another nested IF... ELSE statement is needed, that is, a multi-branch selection structure. The basic format is as follows:

IF<Boolean expression 1>THEN

```
IF<Boolean expression 2>THEN
    <Statement Content 1>;
ELSE
    <Statement Content 2>;
END_IF
ELSE
    <Statement content 3>;
END_IF
```

As mentioned above, put an IF... ELSE statement into IF... ELSE to implement nesting. The following is an example to illustrate the use of nesting.

Example

For the above expression , first judge the value within <Boolean expression 1>:

- If it is TRUE, continue to judge the value within <Boolean expression 2>;
- If the value of <Boolean expression 1> is FALSE, execute <Statement content 3>, return to <Boolean expression 2>judgment;
- If the value of <Boolean expression 2> is TRUE, execute <Statement Content 1>, otherwise execute <Statement Content 2>。

When the equipment enters the automatic mode, if the actual temperature is greater than 50 degrees Celsius, turn on the fan and turn off the heater;

When the temperature is less than or equal to 50 degrees Celsius, turn off the fan and turn on the heater. For example, in manual mode, both the heater and the fan not act.

```
VAR
    bAutoMode: BOOL; //Manual/Auto mode switch
    nTemp: BYTE; //Current temperature
    bFan: BOOL; //Control signal of fan
    bHeating: BOOL; //Control signal of heater
END_VAR
```

```
IF bAutoMode=TRUE THEN
    IF nTemp>50 THEN
        bFan:= TRUE;
        bHeating:= FALSE;
        ELSE
        bFan:= FALSE;
        bHeating:= TRUE;
    END_IF
ELSE
    bFan:= FALSE;
    bHeating:= FALSE;
END_IF
```

IF.. ELESIF.. ELSE Statement

In addition, the multi-branch selection structure can also be presented in the following method. The specific format is as follows:

```
IF <Boolean expression 1>THEN
    <Statement Content 1>;
ELSIF <Boolean expression 2>THEN
    <Statement Content 2>;
ELSIF <Boolean expression 3>THEN
    <Statement content 3>;
    . . .
    . . .
ELSE
    <Statement content n>;
END_IF
```

▪ If the expression <Boolean expression 1> is TRUE, then only the instruction <statement content 1> is executed, and no other instructions are executed;

▪ Otherwise, a judgment is made starting from the expression <Boolean expression 2> until one of the Boolean expressions is TRUE, and the statement content corresponding to this Boolean expression is executed;

▪ If none of the values of the Boolean expression is TRUE, only the instruction <statement content n> is executed.

Example

For example: When the value of newVar is 10, newVar1 is assigned 10;

When the value of newVar is 100, newVar1 is assigned 100;

If the value of newVar is neither 10 nor 100, newVar1 is assigned 1000.

```
VAR
    newVar:INT;
    newvar1:INT;
END_VAR
IF newVar = 10 THEN
    newVar1:=10;
ELSIF newVar = 100 THEN
    newVar1:=100;
ELSE
    newVar1:=1000;
END_IF
```

CASE Statement

The CASE statement is a multi-branch selection statement, it causes the program to select a branch for execution from a multiple branches according to the value of the expression. The basic format is as follows:

CASE<Conditional variables>OF

<Value 1>: <Statement Content 1>;

<Value 2>: <Statement Content 2>;

<Value 3, Value 4, Value 5>: <Statement content 3>;

<Value 6, Value 10>: <Statement Content 4>;

...

<Value n>: <Statement content n>;

ELSE

<ELSE statement contents>;

END_CASE;

The CASE statement executes in the following mode:

▪ If the value of <Conditional variables> is <Value i>, execute the instruction <Statement content i>。

▪ If <conditional variable> doesn't have any specified value, the instruction <ELSE statement content> is executed.

▪ If several values of a condition variable need to execute the same instruction, several values can be written together one after another and separated by commas. In this way, the common instruction is executed, as in the fourth line of the program above.

▪ If the condition variable is required to execute the same instruction within a certain range, it can be separated by two points by writing the initial and final values. In this way, the common instruction is executed, as in the fifth line of the program above.

Example

For example: When the current state is 1 or 5, the device 1 operates and the device 3 stops;

When the current state is 2, the device 2 stops and the device3 operates;

The current state is between 10 and 20, and both device 1 and device 3 are operating;

In other cases, devices 1, 2, and 3 are required to stop. The specific implementation code is as follows:

```
VAR
    nDevice1,nDevice2,nDevice3:BOOL;//The start signal of three device
    nState:BYTE;//The signal of current signal
END_VAR
```

```
CASE nState OF
    0,5:
    nDevice1:= TRUE;
    nDevice3:= FALSE;
    2:
    nDevice2:= FALSE;
    nDevice3:= TRUE;
    10..20:
    nDevice1:= TRUE;
    nDevice3:= TRUE;
ELSE
    nDevice1:= FALSE;
    nDevice2:= FALSE;
    nDevice3:= FALSE;
END_CASE
```

15.4.4 Iteration statements

Iterative statements are mainly used for programs that are executed repeatedly. Common iterative statements include FOR, REPEAT and WHILE statements. This statement is explained in detail below.

FOR loop

The FOR loop statement is used to evaluate an initialization sequence. When a condition is TRUE, the nested statement is repeatedly executed and an iterative reaching sequence is evaluated. If it is FALSE, the loop is terminated. The specific format is as follows:

Before executing the loop, system will first calculate whether <variable> is within the range of <initial> and <target>

- When the step size is positive, <statement content> is executed only if <variable> is less than or equal to <target value>, and if <variable> is greater than <target value>, it will not enter the loop;
- When step size is negative, <Statement Content> is executed only if <Variable> greater than or equal to<Target Value>, and if <Variable> less than <Target value>, it will not enter the loop.

After each execution of <Statement Content>, <Variable> changes in specified step size:

- If no step size is specified, the default step size is 1 (increased by 1);
- When the step size is positive, the loop ends when <Variable> is greater than <Target value>;
- When the step size is negative, the loop ends when <Variable> is less than <Target value>;

Example

Here's an example to demonstrate how to use the FOR loop.

For example: Use the FOR loop to implement 2 to the fifth power calculation.

```
VAR
    i: INT;//Counting value
    Var1: INT;//Output result
END_VAR
```

```
Var1:= 1;
FOR i:=1 TO 5 BY 1 DO
    Var1:= Var1 * 2
END_FOR
```

Assuming that the initial value of Var1 is 1, and after the loop ends, the value of Var1 is 32.

✎Note:

- If <Target value> equal the limit value of <Variable>, it will enter an infinite loop. Assuming that the type of in the above example is SINT (-128 to 127), When <Target value> is set to 127, the controller enters an infinite loop. Therefore, <Target value> cannot be set the limit value.

WHILE loop

The WHILE loop is used similarly to the FOR loop. The difference between the two is that the end condition of the WHILE loop can be any logical expression. That is, user can specify a condition, and when the condition is met, the loop is executed. The specific format is as follows.

```
WHILE <Boolean expression> DO
   <Statement Content>;
END_WHILE;
```

The WHILE loop is executed in the following order:

Calculate the return value of <Boolean expression>.

When the value of<Boolean expression> is TRUE, Repeat execution<Statement Content>;

When the initial value of <Boolean expression> is FALSE, the instruction <Statement Content> will not be executed, jumps to the

end of the WHILE statement.

✎Note:

If the value of <Boolean expression> is always TRUE, then an infinite loop will occur, and the infinite loop shall be avoided. The generation of an infinite loop can be avoided by changing the conditions of the loop instruction. For exampleUsing incrementing and decrementing counters to avoid infinite loops.

Example

Here's an example to demonstrate how to use the WHILE loop. For example:

As long as the value of variable i is not 0, the program the loop is always executed.

```
VAR
    i:INT;
    Var1:INT:=1;
END_VAR
```

```
WHILE i<>0 DO
    Var1:=Var1*2;
    i := i-1;
END_WHILE
```

REPEAT cycle

The REPEAT loop is different from the WHILE loop in that the REPEAT loop checks the end condition only after the instruction has been executed. This means that the loop is executed at least once, regardless of the end condition. The specific format is as follows:

REPEAT

<Statement Content>

UNTIL<Boolean expression>

END_REPEAT;

The REPEAT loop is executed in the following order:

When the value of <Boolean expression> is FALSE, execute <Statement Content>;

When the value of<Boolean expression> is TRUE, stop execution <Statement Content>;

After first execute <Statement Content>, if the value of <Boolean expression> is TRUE, <Statement Content>is executed only once.

Notice

▪ If the value of <Boolean expression> is always TRUE, then an infinite loop will occur, and the infinite loop shall be avoided. The generation of an infinite loop can be avoided by changing the conditions of the loop instruction. For example: Using incrementing and decrementing counters to avoid infinite loops.

Example

Here is an example to demonstrate how to use the REPEAT loop.

For example, the REPEAT loop stops when the counter is 100.

```
VAR
    i:BYTE;
END_VAR
```

```
REPEAT
    i:=i+1;
UNTIL i=100
END_REPEAT
```

The result of this example, each program cycle enters the REPEAT cycle, and i is BYTE (0-255), that is, 256 self-add-one calculations are performed in each cycle.

Because the previously mentioned "this means that the loop is executed at least once regardless of the end condition", whenever the REPEAT statement is entered, i is 1 first, and i is executed 256 times in each cycle: = i+1 instruction until the i variable is accumulated to an overflow of 0, jumping out of the loop. Then it is added to the overflow, and so on.

15.4.5 Jump statement

EXIT statement

If the EXIT instruction is used in the FOR, WHILE, and REPEAT loops, the inner loop stops immediately regardless of the end condition. The specific format is as follows:

EXIT;

For example: Use the EXIT instruction to avoid division by zero when using iterative statements.

```
VAR
    i:BYTE;
    Var1:INT;
    Var2:INT;
END_VAR
```

```
FOR i:=1 TO 5 BY 1 DO
    Var1 := Var1 / 2;
    IF Var1 = 0 THEN
        EXIT
    END_IF
    Var1:=Var1/Var2;
END_FOR
```

When Var1 is equal to 0, the FOR loop ends.

CONTINUE statement

This instruction is an extension instruction of IEC 61131-3 standard, and the CONTINUE instruction can be used in three cycles: FOR, WHILE and REPEAT.

The CONTINUE statement breaks the current loop, ignoring the code behind it and starting a new loop directly. When multiple loops are nested, the CONTINUE statement can only cause the loop statement that contains it directly to start a new loop once. The specific format is as follows:

CONTINUE

For example: Use the CONTINUE directive to avoid division by zero when using iterative statements.

```
VAR
    i:BYTE;
    Var1,Var2:INT;
END_VAR
```

```
FOR i:=1 TO 5 BY 1 DO
    IF Var1 = 0 THEN
        CONTINUE
    END_IF
    Var1 := Var1 / Var2;
END_FOR
```

15.4.6 RETURN statement

The RETURN instruction is used to exit the program organization unit (POU). the specific format is as follows:

RETURN;

For example: Use the IF statement as the judgment, and when the conditions are met, the execution of this program will be terminated immediately.

```
VAR
    nCounter: BYTE;
    bSwitch: BOOL;(*Switch signal*)
END_VAR
```

```
IF bSwitch=TRUE THEN
    RETURN;
END_IF
nCounter:= nCounter + 1;
```

When bSwitch is FALSE, nCounter always performs self-adding by 1. For example, when bSwitch is TRUE, nCounter keeps the value of the previous cycle and immediately exits the program organization unit (POU).

15.4.7 Empty statements

non - execution. An example is as follows:

```
;
```

15.4.8 Comment statements

Comments are a very important part of the program, it makes the program more readable without affecting the execution of the program. Anywhere in the declaration section or the execution section of the ST editor, can add comments.

In the ST language, there are two annotation methods:

Multi-line comments start with (* and end with *). For instance:

```
(*IF newVar1 = TRUE THEN
    newVar1:= FALSE;
END_IF*)
```

A single-line comment starts with "//" and ends with one line.

✎**Note:** The variable declaration area only supports comments on variables.

```
//newVar1:=65535;
```

```
VAR
    newVar: BOOL; //This is a new variable
END_VAR
```

However, the following method of using comments in the variable declaration area is not supported.

```
VAR
    //newVar:BOOL;
END_VAR
```

15.5 Introduction to the REF series instructions

The REF series instructions are specifically designed for devices,, used to access the data stored in devices according to specific data types.The table below lists the devices supported by each REF instruction (V and Z devices are not supported).

Instruction	Applicable device	Usage Description
REFBOOL	S/M/SM/X/Y	Get the value of the corresponding bit-type device
	C/LC/HSC/T	Get the value of the corresponding bit-type device
	D/R/SD	Get the low-byte value of the corresponding device
REFBYTE	C/LC/HSC/T	Access the low-byte data of the corresponding word-type device as BYTE type
	D/R/SD	Access the low-byte data of the corresponding word-type device as BYTE type
REFWORD	C/LC/HSC/T	Access the 2-byte data starting from the corresponding device as a WORD type.
	D/R/SD	Access the 2-byte data starting from the corresponding device as a WORD type.
REFDWORD	C/LC/HSC/T	Access the 4-byte data starting from the corresponding device as a DWORD type.
	D/R/SD	Access the 4-byte data starting from the corresponding device as a DWORD type.
REFLWORD	C/LC/HSC/T	Access the 8-byte data starting from the corresponding device as a LWORD type.
	D/R/SD	Access the 8-byte data starting from the corresponding device as a LWORD type.
REFSINT	C/LC/HSC/T	Access the low-byte data of the corresponding word-type device as SINT type
	D/R/SD	Access the low-byte data of the corresponding word-type device as SINT type
REFUSINT	C/LC/HSC/T	Access the low-byte data of the corresponding word-type device as USINT type
	D/R/SD	Access the low-byte data of the corresponding word-type device as USINT type
REFINT	C/LC/HSC/T	Access the 2-byte data starting from the corresponding device as a INT type
	D/R/SD	Access the 2-byte data starting from the corresponding device as a INT type
REFUINT	C/LC/HSC/T	Access the 2-byte data starting from the corresponding device as a UINT type
	D/R/SD	Access the 2-byte data starting from the corresponding device as a UINT type
REFDINT	C/LC/HSC/T	Access the 4-byte data starting from the corresponding device as a DINT type.
	D/R/SD	Access the 4-byte data starting from the corresponding device as a DINT type.
REFUDINT	C/LC/HSC/T	Access the 4-byte data starting from the corresponding device as a UDINT type.
	D/R/SD	Access the 4-byte data starting from the corresponding device as a UDINT type.
REFLINT	C/LC/HSC/T	Access the 8-byte data starting from the corresponding device as a LINT type
	D/R/SD	Access the 8-byte data starting from the corresponding device as a LINT type
REFREAL	C/LC/HSC/T	Access the 4-byte data starting from the corresponding device as a REAL type
	D/R/SD	Access the 4-byte data starting from the corresponding device as a REAL type

Instruction	Applicable device	Usage Description
REFLREAL	C/LC/HSC/T	Access the 8-byte data starting from the corresponding device as a LREAL type
	D/R/SD	Access the 8-byte data starting from the corresponding device as a LREAL type
REFTIME	C/LC/HSC/T	Access the 4-byte data starting from the corresponding device as a TIME type
	D/R/SD	Access the 4-byte data starting from the corresponding device as a TIME type
REFTOD	C/LC/HSC/T	Access the 4-byte data starting from the corresponding device as a TOD type
	D/R/SD	Access the 4-byte data starting from the corresponding device as a TOD type
REFDATE	C/LC/HSC/T	Access the 4-byte data starting from the corresponding device as a DATE type
	D/R/SD	Access the 4-byte data starting from the corresponding device as a DATE type
REFDT	C/LC/HSC/T	Access the 4-byte data starting from the corresponding device as a DT type
	D/R/SD	Access the 4-byte data starting from the corresponding device as a DT type
REFLTIME	C/LC/HSC/T	Access the 8-byte data starting from the corresponding device as a LTIME type
	D/R/SD	Access the 8-byte data starting from the corresponding device as a LTIME type

A simple code example for the REF statement is as follows:

```
PROGRAM MAIN
VAR
    res: BOOL;
END_VAR
```

```
res:= REFBOOL(D0); //Get the first byte of the D0 device. Returns TRUE if the value is 1; returns FALSE if the value is 0.
```

15.6 ST Application instruction

15.6.1 OUT T/ Timer output

When the result of the operation before the OUT instruction is ON, the coil of the timer/accumulation timer specified by (d) is ON and starts timing until the set value. When the time limit is reached, the normally open contact turns ON, and the normally closed contact turns OFF.

[OUT (d, value)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(d)	Timer device number	-	Counter	ANY	-
(value)	Timer setting value	-	Unsigned BIN16 bits	ANY16	ANY_INT

Device used

Instruction	Parameter	Devices																							Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
OUT T	Parameter 1															●											
	Parameter 2																	●	●	●			●				

Function

- When the result before the OUT instruction is ON, the coil of the timer specified in (d) turns ON and counts until the set value is reached. If the count reaches (current value ≥ set value), the normally open contact turns ON, and the normally closed contact turns OFF.
- When the result before the OUT instruction changes from ON to OFF, the situation is shown below.

Timer type	Timer coil	Current value of timer	Before the time limit is reached		After the time limit is reached	
			Normally open contact	Normally closed contact	Normally open contact	Normally closed contact
Timer	OFF	0	Non-conducting	conducting	Non-conducting	conducting
cumulative timer	OFF	Keep the current value	Non-conducting	conducting	conducting	Non-conducting

- After the time limit is reached, the current value of the accumulation timer is cleared and the contact is turned OFF by the RST instruction.
- When the set value is 0, the timer will reach the time limit as soon as the OUT instruction is executed.
- When the same OUT T instruction is executed twice or more in the same scan, the current value will be updated according to the number of executions.
- Description of each timer:

Device number	Timer specifications	Device number	Timer specifications
The QPS Exceeds Maximum Limit 5	100ms timer	T246 to T249	1ms cumulative timer
T192 to T199	100ms subroutine timer (cumulative type, used in subroutines, it will still be updated even if the subroutine is not called)	T250 to T255	10ms cumulative timer
		T256 to T383	1ms timer
T200 to T245	10ms timer	T384 to T511	0.1ms timer

Error code

Error code	Content
4084H	The parameter setting in (value) is out of range.

Example

```
1   D0:=100;
2   OUT(T0,D0);
3   IF REFBOOL(T0) THEN
4       D1:=1234;
5   END_IF
```

With the timing, the value of D1 after 10s is 1234.

15.6.2OUT C/Counter output

16-bit counter instructions: When the result of the operation before the OUT instruction changes from OFF to ON, the current value of the counter specified in (d) is incremented by 1. If the count is reached, the normally open contact becomes conductive, and the normally closed contact becomes non-conductive.

[OUT (d, value)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(d)	Counter Component Number	-	Counter	ANY	-
(value)	Counter Setting Value	[0,32767]	Unsigned BIN16 bits	ANY_INT	INT

Device used

Instruction	Parameter	Devices																						Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
OUT C	Parameter 1																●										
	Parameter 2															●		●	●	●			●	●			

Function

- When the operation result before the OUT instruction changes from OFF → ON, the current value (count value) of the counter specified in (d) is set to +1. If the count reaches (current value ≥ set value), the normally open contact becomes conductive, and the normally closed contact becomes non-conductive.
- Counting is not performed when the operation result remains ON. (The count input does not need to be pulsed.)
- After the count is reached, the count value and the contact status do not change until the RST instruction is executed.
- When the value is set to 0, the processing is the same as when it is 1.

Error code

Error code	Content
4084H	The parameter setting in (value) is out of range.
4085H	The (value) parameter is out of the range of devices.

Example

```
PROGRAM MAIN
VAR
    BOOL_CHECK:BOOL;
    UP_CHECK:R_TRIG;
END_VAR
```

```
UP_CHECK(CLK:=BOOL_CHECK,Q=>);
IF UP_CHECK.Q THEN
    OUT(LC0,K100);
END_IF
```

Each time the BOOL_CHECK variable changes from FALSE to TRUE, the value of LC0 increments by 1 until LC0 equals the value of the second parameter.

15.6.3 OUT LC/ Long counter output

32-bit counter instructions: When the result of the operation before the OUT instruction changes from OFF to ON, the current value of the long counter specified in (d) is increment by 1. If the count is reached, the normally open contact becomes conductive, and the normally closed contact becomes non-conductive.

- [OUT (d, value)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(d)	Long counter device number	-	Counter	ANY	-
(value)	Long counter setting value	[0,4294967295]	Unsigned BIN32 bits	ANY_INT	UDINT

Device used

Instruction	Parameter	Devices																						Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
OUT LC	Parameter 1																				●						
	Parameter 2															●		●	●	●			●	●			

Function

- When the calculation result before the OUT instruction changes from OFF → ON, the current value (count value) of the long counter specified in (d) is set to +1. If the count reaches (current value ≥ set value), the normally open contact becomes conductive, and the normally closed contact becomes non-conductive.
- Counting is not performed when the operation result remains ON. (The count input does not need to be pulsed.)
- After the count is reached, the count value and the contact status do not change until the RST instruction is executed.
- When the value is set to 0, the processing is the same as when it is 1.

Error code

Error code	Content
4085H	The (value) parameter is out of the range of devices.

Example

```
IF M0 THEN
    OUT(LC0,K10);
END_IF
IF REFBOOL(LC0) THEN
    Y0:=TRUE;
END_IF
```

When M0 is set, the value of LC0 will become 10, and the value monitored by REFBOOL (LC0) will turn TRUE, and the value of Y0 will be set to TRUE.

15.6.4 DI, EI/Disable Interrupt, Enable Interrupt

The CPU module is usually in an interrupt disabled state. This instruction can change the CPU module to an interrupt enabled state (EI instruction) and then switch it back to a disabled state (DI instruction).

- ▪DI: Disables the execution of interrupt programs.
- ▪EI: Release interrupt-disabled state.

- [DI () /DI (s)]
- [EI ()]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(s)	Interrupt disable priority	[0,2]	Signed BIN16 bits	ANY16	INT

Device used

Instruction	Parameter	Devices																							Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
DI	Parameter 1															●	●	●	●	●			●	●		●	

Function

■ DI

- Even if the conditions for executing an interrupt are triggered in the program, the execution of interrupt programs is disabled until the EI instruction is executed.
- When the PLC is powered on or after it enters the STOP state, it will enter the state after the DI instruction is executed, and interrupt programs cannot be executed.
- The DI instruction can optionally use parameters: if no parameter is specified, all interrupt programs are disabled; if parameters are used, according to the value in parameter s1, interrupt programs with this priority level and lower priority levels are disabled.
- The priority level range of interrupts is [0,2]. The smaller the value, the higher the interrupt response priority (an interrupt with priority level 0 is responded to the fastest).
- If there is no EI instruction executed before the DI instruction, the DI instruction will be invalid.

■ EI

- Releases the interrupt-disabled state set by the execution of the DI instruction and allows interrupt programs to run.
- When the EI and DI instructions are not enabled, the original allowed or prohibited interrupt program execution status is maintained, and the currently disabled interrupt priority levels can be viewed in SD151.

For details of the nested structure of DI and EI instructions, please refer to the section of 4.3 Disable interrupt and enable interrupt.

15.6.5 ZRST/Data batch reset

Perform a batch reset between the devices specified in (d1) and (d2) of the same type. Used during interrupt operation, initial operation, or when resetting control data.

[ZRST (d1, d2)]

Content, range and data type

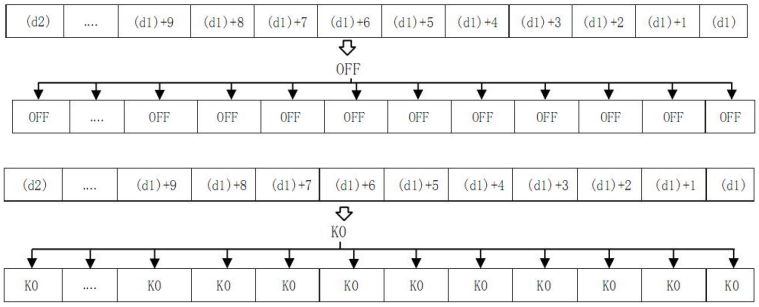
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(d1)	Start bit/word device number for batch reset	-	Bits/Signed BIN16 Bits	ANY_ELEMENTARY	-
(d2)	Start bit/word device number for batch reset	-	Bits/Signed BIN16 Bits	ANY_ELEMENTARY	-

Device used

Instruction	Parameter	Devices																						Offset modification	Pulse expansion
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]	XXP
ZRST	Parameter 1		●	●	●	●										●	●	●	●	●	●	●				●	
	Parameter 2		●	●	●	●										●	●	●	●	●	●	●				●	

Function

- Perform a batch reset between the devices specified in (d1) and (d2) of the same type.



When (d1) and (d2) are bit devices, write OFF (reset) to all devices in the range from (d1) to (d2).

When (d1) and (d2) are bit devices, write K0 to all devices in the range from

- As a single reset application instruction for a devices, the RST instruction can be used for bit devices or word devices.

```
1  RST(M0);
2  RST(D0);
3  RST(C0);
```

The instructions in the figure will reset M0, D0, C0.

Note

- Please assign the same device number for (d1) and (d2) , and make (d1) number <(d2) number. When (d1) number ≥ (d2) number, only 1 point is reset for the device specified in (d1).

Error code

Error code	Content
4084H	When the device type specified in (d1) is different from the device type specified in (d2).
4085H	When the devices specified in the read application instructions (d1) and (d2) exceed the range of the corresponding device.
4086H	When the devices specified in the write application instruction (d1) excceed the range of the corresponding device.

Example

```
1  ZRST(M0,M7999);
```

The function of the application instruction is to set the values of devices from M0 to M7999 to 0.

15.6.6 ZSET/Data batch set

Batch set between the bit devices specified in (d1) and (d2) of the same type.

[ZSET (d1, d2)]

Content, range and data type

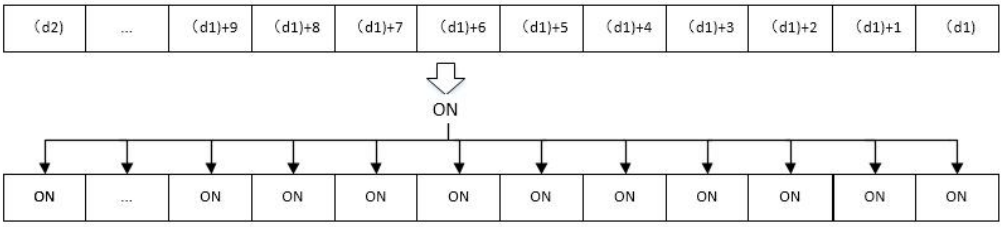
Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(d1)	Starting bit device number for batch set	-	Bit	ANY_BOOL	-
(d2)	Starting bit device number for batch set	-	Bit	ANY_BOOL	-

Device used

Instruction	Parameter	Devices																						Offset modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
ZSET	Parameter 1		●	●	●	●					●															●
	Parameter 2		●	●	●	●					●															●

Function

- Batch set between the bit devices specified in (d1) and (d2) of the same type.
- Write ON (set) over the entire range of devices from (d1) to (d2).



Note

- Please specify the same type of number for (d1), (d2) and make (d1) number <(d2) number. When (d1) number ≥ (d2) number, only 1 point is reset for the device specified in (d1).

Error code

Error code	Content
4084H	When the device type specified in (d1) is different from the device type specified in (d2).
4085H	When the devices specified in the read application instructions (d1) and (d2) exceed the range of the corresponding device.
4086H	When the devices specified in the write application instruction (d1) excceed the range of the corresponding device.
4087H	When the device type specified in (d1) is different from the device type specified in (d2).

Example

```
1  ZSET(M1,M100);
```

The function of the application instruction is to set the values of devices from M0 to M100 to 0.

15.6.7 RST/ Data reset

When the RST input turns ON, the soft element specified in (d) will change to the following state.

- Bit device: Set the coil and contact to OFF.
- Timer, counter: Set the current value to 0 and set the coil, contact to OFF.
- Word device bit specification Set the specified bit to 0.
- Word device, module access device, index register: Set the content to 0.

[RST (d)]

Content, range and data type

Parameter	Content	Range	Data type	Data type (label)	Custom variable type
(d)	Bit device number for reset /Word device bit specification/Word device number for reset	-	Bit/word/double word	ANY_ELEMENTARY	-

Device used

Instruction	Parameter	Devices																							Offset modification
		X	Y	M	S	SM	T(bit)	C(bit)	LC(bit)	HSC(bit)	D.b	KnX	KnY	KnM	KnS	T	C	D	R	SD	LC	HSC	K	H	E	[D]
RST	Parameter 1															●	●				●					●

Function

When the execution instruction is ON, the specified device will change to the following state.

Devices	Device status
Bit device	Set the coil and contact to OFF.
Timer, counter	Set the current value to 0 and set the coils and contacts to OFF
Word device bit specification	Set the specified bit to 0.
Word device	Set content to 0

- When the execution instruction is OFF, the state of the soft element does not change.

✎Note

For timers and counters, if the RST instruction is executed in a jumped program, subroutine, or interrupt program, the timers and counters may remain in the reset state and stop operating.

Error code

Error code	Content
4086H	(d) Write address exceeds the device range.

Example

```
1  RST(C1);
2  RST(T1);
3  RST(LC1);
```

In this example, the RST instruction resets the devices C1, T1, and LC1.

Appendixes

Appendix 1 Special Relay (SM)

Error message

SM sign	Name	Content	R/W	Retention on Power Loss
SM0	Latest error message	OFF: No error ON: There is an error	R	×
SM1	Reserved			
SM2	Error release	OFF→ON: Error clearing request ON→OFF: Error cleared	R/W	×
SM3	Low battery voltage	OFF: Normal ON: Low battery voltage	R	×
SM4	Low battery voltage latch	OFF: Normal ON: Low battery voltage	R	×
SM5	Reserved		R	×
SM6	PLC hardware error	OFF: No error ON: Error	R	×
SM7	PLC communication error	OFF: No error ON:Error	R	×
SM10	Parameter error	OFF: No error ON: Error	R	×
SM11	Operation error	OFF: No error ON: Error	R	×
SM14	Operation error latch	OFF: No error ON: Error	R	×

System information

SM sign	Name	Content	R/W	Power-down save
SM30	Low battery warning shield	OFF: Close (Default) ON: Open	R/W	×
SM31	Clear all the non-holding registers	OFF→ON: Clear request ON→OFF: Cleared	R/W	×
SM32	Clear all the holding registers	OFF→ON: Clear request ON→OFF: Cleared	R/W	×
SM33	All devices remain unchanged in stop state	OFF: Close (Default) ON: Open	R/W	×
SM34	All PLC outputs are OFF	OFF: Close (Default) ON: Open	R/W	×
SM35	Low battery BAT light status	OFF: BAT lamp is invalid (default) ON: Low battery BAT light is on	R/W	×
SM36~SM49	Reserved			
SM53~SM99	Reserved			

Clock Information

SM sign	Name	Content	R/W	Power-down save
SM100	Always ON after RUN		R	×
SM101	Always OFF after RUN		R	×
SM102	The first cycle after RUN is ON		R	×
SM103	The first cycle after RUN is OFF		R	×
SM104	USB power supply	It is under USB power supply mode when it is ON. At this time, it only supports downloading, clock setting, password setting and other functions.	R	×
SM105~SM106	Reserved		R	×
SM107	Clock stop and preset	Stop clock running and display	R/W	×
SM108	Clock reading display stopped	Internal running of clock, display stopped	R/W	×
SM109	1min oscillation clock	Switch state every 30s	R	×
SM110	1ms oscillation clock	Switch state every 0.5ms	R	×
SM111	10ms oscillation clock	Switch state every 5ms	R	×
SM112	100ms oscillation clock	Switch state every 50ms	R	×
SM113	1s oscillation clock	Switch state every 500ms	R	×
SM114	nms oscillator clock	Switch state every (n/2)ms, n is set by SD114	R	×
SM115	ns oscillator clock	Switch state every (n/2)s, n is set by SD115	R	×
SM116	±30s correction	Clear when the clock seconds are less than 30S, carry the digits when the clock seconds are greater than 30S	R/W	×
SM117~SM119	Reserved		R	×

Scan information

SM sign	Name	Content	R/W	Power-down save
SM120	Constant scan cycle	OFF: Not enabled (default) ON: enabled	R/W	×
SM121	RUN, STOP control	OFF: STOP ON: RUN	R/W	×
SM122	Ladder diagram watchdog function switch	OFF: Not enabled ON: enabled (default)	R/W	×

Instruction related

SM sign		Name	Content	R/W	Power-down save
SM151		Carry sign	OFF: Carry not occurred in calculation ON: Carry occurred in calculation	R	×
SM152		Abdication sign	OFF: abdication not occurred in calculation ON: abdication occurred in calculation	R	×
SM153		Zero sign	OFF: The result is not zero OFF: The result is zero	R	×
SM160		XCH exchange mode	OFF: parameter 1 is switched with parameter 2 ON: The upper eight bits are switched with eighth bits of the parameter itself	R/W	×
SM161		Bit processing mode (ASC, ASCI, BCC, CCD, CRC)	OFF: 16-bit processing mode OFF: 8-bit processing mode	R/W	×
SM165		SORT/SORT2 instruction ascending and descending sequence selection	OFF: Ascending ON: descending	R/W	×

SM sign		Name	Content	R/W	Power-down save
SM167		HKY instruction HEX data processing	OFF: Numeric key+ function key ON: Hex	R/W	×
SM168		SMOV instruction hexadecimal processing	OFF: execute BIN→BCD switch ON: do not execute BIN→BCD switch	R/W	×
SM191		BINDA output character number switching signal	OFF: Output 00H ON: No change	R/W	×
SM224		BMOV instruction direction	OFF: forward transfer ON: reverse transfer	R/W	×
SM226		RAMP instruction mode	OFF: loop execution mode ON: Hold after completion	R/W	×
SM227		HSC instruction mode	OFF: 8-byte serial output (eight characters are fixed) ON: 16-byte serial output (1-16 characters)	R/W	×
SM229		Partial application instruction execution completion sign	OFF: Instruction not executed or is executing ON: instruction is executed	R/W	×
SM240		STL: transfer prohibited	OFF: Ordinary action ON: state transfer prohibited	R/W	×
SM241		IST instruction transfer starts	OFF: IST instruction is not executed ON: IST instruction transfer starts	R/W	×
SM242		IST instruction corresponds to the pulse output of the start input	OFF: not enabled ON: enabled	R/W	×
SM243		End sign of IST instruction homing state (user program control)	OFF: homing not ended ON: homing ended	R/W	×
SM244		IST instruction detects the mechanical origin action (user program control)	OFF: Non-mechanical origin ON: mechanical origin	R/W	×
SM245		For STL instruction: Reset all output when mode switching is prohibited (user program control)	OFF: Reset all output at the time of state switching ON: No action is performed on output at the time of state switching		×
SM246		IST instruction: ON with STL status	OFF: when STL monitoring is invalid (OFF), or when STL monitoring is valid (ON) and all step relays (S devices) are OFF ON: when STL monitoring is valid (ON) and any of the step relays (S devices) is ON		×
SM247		STL monitoring valid	OFF: Invalid ON: turn STL monitoring into valid, and the status numbers (S0-S4095) in action are stored in the special auxiliary relays SD240-SD247 in order from small to large.	R/W	×
SM248		ANS instruction signal alarm action	OFF: The alarm does not act ON: The alarm acts	R/W	×
SM249		ANS instruction signal alarm is valid	OFF: alarm is not valid ON: The alarm is valid	R/W	×
SM340		DUTY timing clock output 1	Timing clock output of DUTY instruction	R	×
SM341		DUTY timing clock output 2		R	×
SM342		DUTY timing clock output 3		R	×
SM343		DUTY timing clock output 4		R	×

SM sign		Name	Content	R/W	Power-down save
SM344		DUTY timing clock output 5		R	×

Interrupt prohibited

SM sign	Name	Content	R/W	Power-down save
SM352	X0 rising edge interrupt	OFF: X0 rising edge interrupt valid ON: X0 rising edge interrupt prohibited	R/W	×
SM353	X0 falling edge interrupt	OFF: X0 falling edge interrupt valid ON: X0 falling edge interrupt prohibited	R/W	×
SM354	X1 rising edge interrupt	OFF: X1 rising edge interrupt valid ON: X1 rising edge interrupt prohibited	R/W	×
SM355	X1 falling edge interrupt	OFF: X1 falling edge interrupt valid ON: X1falling edge interrupt prohibited	R/W	×
SM356	X2 rising edge interrupt	OFF: X2 rising edge interrupt valid ON: X2 rising edge interrupt prohibited	R/W	×
SM357	X2 falling edge interrupt	OFF: X2 falling edge interrupt valid ON: X2 falling edge interrupt prohibited	R/W	×
SM358	X3 rising edge interrupt	OFF: X3 rising edge interrupt valid ON: X3 rising edge interrupt prohibited	R/W	×
SM359	X3 falling edge interrupt	OFF: X3 falling edge interrupt valid ON: X3 falling edge interrupt prohibited	R/W	×
SM360	X4 rising edge interrupt	OFF: X4 rising edge interrupt valid ON: X4 rising edge interrupt prohibited	R/W	×
SM361	X4 falling edge interrupt	OFF: X4 falling edge interrupt valid ON: X4 falling edge interrupt prohibited	R/W	×
SM362	X5 rising edge interrupt	OFF: X5 rising edge interrupt valid ON: X5 rising edge interrupt prohibited	R/W	×
SM363	X5 falling edge interrupt	OFF: X5 falling edge interrupt valid ON: X5 falling edge interrupt prohibited	R/W	×
SM364	X6 rising edge interrupt	OFF: X6 rising edge interrupt valid ON: X6 rising edge interrupt prohibited	R/W	×
SM365	X6 falling edge interrupt	OFF: X6 falling edge interrupt valid ON: X6 falling edge interrupt prohibited	R/W	×
SM366	X7 rising edge interrupt	OFF: X7 rising edge interrupt valid ON: X7 rising edge interrupt prohibited	R/W	×
SM367	X7 falling edge interrupt	OFF: X7 falling edge interrupt valid ON: X7 falling edge interrupt prohibited	R/W	×

High-speed input and output

SM sign	Name	Content	R/W	Power-down save
SM400	HSC0 contact status	OFF: The count doesn't reach the set value ON: the count reaches the set value	R	×
SM401	HSC0 motion direction	OFF: Positive direction ON: reverse direction	R	×
SM405	HSC0 counting direction	OFF: CTU ON: CTD	R/W	×
SM406~SM429	Reserved			
SM430	HSC1 contact status	OFF: The count doesn't reach the set value ON: the count reaches the set value	R	×
SM431	HSC1 motion direction	OFF: Positive direction ON: reverse direction	R	×
SM435	HSC1 counting direction	OFF: CTU ON: CTD	R/W	×
SM436~SM459	Reserved			
SM460	HSC2 contact status	OFF: The count doesn't reach the set value ON: the count reaches the set value	R	×
SM461	HSC2 motion direction	OFF: Positive direction ON: reverse direction	R	×

SM sign	Name	Content	R/W	Power-down save
SM465	HSC2 counting direction	OFF: CTU ON: CTD	R/W	×
SM466~SM489	Reserved			
SM490	HSC3 contact status	OFF: The count doesn't reach the set value ON: the count reaches the set value	R	×
SM491	HSC3 motion direction	OFF: Positive direction ON: reverse direction	R	×
SM495	HSC3 counting direction	OFF: CTU ON: CTD	R/W	×
SM496~SM519	Reserved			
SM520	HSC4 contact status	OFF: The count doesn't reach the set value ON: the count reaches the set value	R	×
SM521	HSC4 motion direction	OFF: Positive direction ON: reverse direction	R	×
SM525	HSC4 counting direction	OFF: CTU ON: CTD	R/W	×
SM526~SM549	Reserved			
SM550	HSC5 contact status	OFF: The count doesn't reach the set value ON: the count reaches the set value	R	×
SM551	HSC5 motion direction	OFF: Positive direction ON: reverse direction	R	×
SM555	HSC5 counting direction	OFF: CTU ON: CTD	R/W	×
SM556~SM579	Reserved			
SM580	HSC6 contact status	OFF: The count doesn't reach the set value ON: the count reaches the set value	R	×
SM581	HSC6 motion direction	OFF: Positive direction ON: reverse direction	R	×
SM585	HSC6 counting direction	OFF: CTU ON: CTD	R/W	×
SM586~SM609	Reserved			
SM610	HSC7 contact status	OFF: The count doesn't reach the set value ON: the count reaches the set value	R	×
SM611	HSC7 motion direction	OFF: Positive direction ON: reverse direction	R	×
SM615	HSC7 counting direction	OFF: CTU ON: CTD	R/W	×
SM616~SM639	Reserved			

Communication

SM sign	Name	Content	R/W	Power-down save
Communication COM1				
SM2540SM2540	COM1 send control on		R/W	×
SM2541	COM1 send control/send prompt		R/W	×
SM2542	COM1 receive control on		R/W	×
SM2543	COM1 accept control/receive prompt		R/W	×
SM2544	COM1 8-bit mode		R/W	×
SM2560	COM1 communication completion sign		R/W	×
SM2561	COM1 receiving sign		R/W	×
SM2562	COM1retry		R/W	×
SM2563	COM1 communication error		R/W	×
SM2564	COM1communication timeout		R/W	×
SM2565	Reserved			×
SM2566	Reserved			×
SM2567	Reserved			×
SM2568	Reserved			×
SM2569	Reserved			×
SM2570	Reserved			×
SM2571	Reserved			×
Communication COM2				
SM2590	COM2 send control on	OFF: Data transmission will be controlled by SM2591 ON: Data transmission is performed automatically	R/W	×
SM2591	COM2 send control/send prompt	OFF→ON: data transmission starts ON→OFF: data transmission ends	R/W	×
SM2592	COM2 receive control on	OFF: when data are fully received, it will be automatically stopped ON: Received normally, not affected by sign SM2593	R/W	×
SM2593	COM2 accept control/receive prompt	OFF: Data is not received fully or data is not received (according to the status of SM2592) ON: Data is received fully or data is received (according to the status of SM2592)	R/W	×
SM2594	8-bit mode of COM2 (for RS user-defined protocol)	OFF: 16-bit mode ON: 8-bit mode		×
SM2610	Communication completion sign	OFF: communication not completed ON: communication completed	R/W	×
SM2611	Receiving sign	OFF: data is not received ON: data is being received	R/W	×
SM2612	Retry	OFF: no retry occurs ON: retry occurs	R/W	×
SM2613	Communication error	OFF: No error ON: communication error occurred	R/W	×
SM2614	Communication timeout	OFF: normal communication ON: communication timeout	R/W	×
SM2615	Reserved			

List of Ethernet-related special devices

SM sign	Name	Content	R/W	Power-down save
SM2681	Display current network information	Refresh the current IP, subnet mask and default gateway after being set as ON. OFF after refresh is completed.	R/W	×
SM2682	Display current MAC information	Refresh the current MAC after being set as ON. OFF after refresh is completed.	R/W	×
SM2683	IP, subnet mask, gateway modification sign	ON: can be modified OFF: cannot be modified (When being set as ON, STOP->RUN; it can be automatically changed to OFF after modification)	R/W	√
SM2684	Network card connection status	ON: network connected OFF: network not connected; check if the network cable is connected	R	√
SM2692	MAC address modification sign	ON: can be modified OFF: cannot be modified (When being set as ON, STOP->RUN; it can be automatically changed to OFF after modification)	R/W	√

Appendix 2 Special register (SD)

Error message

SD label	Name	Content	R/W	Power-down save
SD0	Latest error code of error message	The latest self-diagnosis error code will be stored	R	X
SD1	Reserved			
SD2	Set minimum battery voltage	Default: 26 (2.6V)	R/W	X
SD3	Current battery voltage	Unit: 0.1V	R	X
SD4	Battery voltage latch value	When there is a warning showing that the battery voltage is too low, the battery voltage value under error state shall be latched, with the unit being 0.1V.	R	X
SD5	AC/DC power down times	Record the number of the power-down and re-startup of the current power supply	R	X
SD6	Error code of PLC hardware error	Error code of PLC hardware error will be stored	R	X
SD7	PLC communication error code	Error code of communication error will be stored	R	X
SD8	PLC communication error step number low word	The step number of the ladder diagram of communication error will be stored, double word	R	X
SD9	PLC communication error step number high word		R	X
SD10	Parameter error code	Error code of parameter error will be stored	R	X
SD11	Operation error code	Error code of operation error will be stored	R	X
SD12	Operation error program step number low word	The step number of the ladder diagram of operation error will be stored, double word	R	X
SD13	Operation error program step number high word		R	X
SD14	Operation error code latch	Error code of operation error will be stored and such error cannot be cleared by the error removal function	R	X
SD15	Operation error program step number latch low word	The step number of the ladder diagram of operation error will be stored, and such error cannot be cleared by the error removal function, double word	R	X
SD16	Operation error program step number latch high word		R	X
SD17	Program error code	Unpredictable errors occurred in PLC	R	X
SD18	AC220V power down recovery time	The recovery time of AC220V after power down will be stored in ms	R	X
SD19~SD29	Reserved			

System information

SD label	Name	Content	R/W	Power-down save
SD30	Model identification code	Store PLC model identification code, which cannot be modified	R	X
SD31	Software version number	Store PLC software version number, which cannot be modified	R	X
SD32	Hardware version number	Store PLC hardware version number, which cannot be modified	R	X
SD33	Input points	Store PLC input points, which cannot be modified	R	X
SD34	Output points	Store PLC output points, which cannot be modified	R	X
SD35	Number of high-speed input axis	Store number of high-speed input axis of PLC, which cannot be modified	R	X
SD36	Number of high-speed output axis	Store number of high-speed output axis of PLC, which cannot be modified	R	X
SD37 and SD38	Relay identification	Identify how many output points are relay type, use mask mode, each bit identifies one output point, and code 1 is relay type	R	X
SD40~SD47	Product unique ID (16 bytes)	Store product unique ID, which cannot be modified	R	X
SD48	Compile the link version	Store PLC compilation link module version, which cannot be modified	R	X
SD49	Production information string	Store production information and ASCII code		X
SD50~SD99	Model identification code	Store PLC model identification code, which cannot be modified	R	X

Clock information

SD label	Name	Content	R/W	Power-down save
SD100	Real time clock seconds (0~59)	PLC Built-in RTC Clock	R	X
SD101	Real time clock minutes (0~59)		R	X
SD102	Real-time clock hour (0~23)		R	X
SD103	Real time clock day (1~31)		R	X
SD104	Real time clock month (1~12)		R	X
SD105	Real-time clock Gregorian calendar year (2000~2099)		R	X
SD106	Real time clock week		R	X
SD107~SD113	Reserved			
SD114	n value of nms oscillation clock	Set the n value of SM114 oscillation clock, and the default is 500ms	R/W	X
SD115	n value of ns oscillation clock	Set the n value of SM114 oscillation clock, and the default is 2s	R/W	X
SD116~SD119	Reserved			

Scan information

SM sign	Name	Content	R/W	Power-down save
SD120	Constant scan cycle time setting (ms)	Default: 10ms	R/W	X
SD122	Watchdog timer time setting value	Unit ms, default: 200ms	R/W	X
SD128	Scan period current value ms part	• The current scan time will be stored in SD128 and SD129. (The measurement is carried out in 1μs) SD128: Store bits of ms (storage range: 0~65535) SD129: Store bits of μs (storage range: 0 ~ 999) (For instance) If the current scan time is 23.6 ms, store it as follows. SD128=23 SD129=600 • When the state changes from STOP to RUN, it will be cleared once.	R	X
SD129	Scan period current value us part		R	X
SD130	Scan period maximum value ms part	The maximum value of the scan time after removing the scan time of the initial execution program is stored in SD 130 and SD 131. (The measurement is carried out in 1μs) SD130: Store bits of ms (storage range: 0~65535) SD131: Store bits of μs (storage range: 0 ~ 999) • When the state changes from STOP to RUN, it will be cleared once.	R	X
SD131	Scan period maximum value us part		R	X
SD132	Scan period minimum value ms part	The maximum value of the scan time after removing the scan time of the initial execution program is stored in SD 133 and SD 134. (The measurement is carried out in 1μs) SD130: Store bits of ms (storage range: 0~65535) SD131: Store bits of μs (storage range: 0 ~ 999) • When the state changes from STOP to RUN, it will be cleared once.	R	X
SD133	Scan period minimum value us part		R	X

SM sign	Name	Content	R/W	Power-down save
SD134	Initial scan time ms part	• The current scan time will be stored in SD134 and SD135. (The measurement is carried out in 1μs) SD134: Store bits of ms (storage range: 0~65535) SD135: Store bits of μs (storage range: 0 ~ 999) • When the state changes from STOP to RUN, it will be cleared once.	R	X
SD135	Initial scan time us part		R	X
SD136	END processing time ms part	• After the scan program ends, the time until the next scan starts will be stored to SD136 and SD137. (The measurement is carried out in 1μs) SD136: Store bits of ms (storage range: 0~65535) SD137: Store bits of μs (storage range: 0 ~ 999) • When the state changes from STOP to RUN, it will be cleared once.	R	X
SD137	END processing time us part		R	X
SD138	Program execution time ms part	Constant scan waiting time (ms unit) • The waiting time at the time of constant scan setting is stored in SD 138 and SD 139. (The measurement is carried out in 1μs) SD138: Store bits of ms (storage range: 0~65535) SD149: Store bits of μs (storage range: 0 ~ 999) • When the state changes from STOP to RUN, it will be cleared once.	R	X
SD139	Program execution time us part		R	X
SD140	Constant scan cycle waiting time ms	•The execution time of the scan program in one scan is stored in SD 140 and SD 141. (The measurement is carried out in 1μs) SD140: Store bits of ms (storage range: 0~65535) SD141: Store bits of μs (storage range: 0 ~ 999) • When the state changes from STOP to RUN, it will be cleared once.	R	X
SD141	Constant scan cycle waiting time us		R	X
SD150	Current interrupt priority	During the execution of the interrupt program, the priority of the existence of the interrupt will be stored. 0~2: priority of the interrupt pointer of the interrupt program during execution	R	X
SD151	Current prohibited interrupt priority	According to the interrupt prohibition instruction (DI instruction), the interrupt prohibition instruction (DI instruction) below the specified priority, and the interrupt enable instruction (EI instruction), the priority of the interrupt prohibition will be stored. 0: Interrupt prohibition (default) of all priorities 1: interrupt prohibition of priorities 1 and 2; 2: interrupt prohibition of priority 2; 3: interrupt enable of all priorities	R	X

Instruction related

SD label	Name	Content	R/W	Power-down save
SD150	Current interrupt priority	During the execution of the interrupt program, the priority of the existence of the interrupt will be stored. 1~3: the priority of the interrupt pointer of the interrupt program during execution 0: Interrupt not executed (default)	R	X
SD151	Current prohibited interrupt priority	According to the interrupt prohibition instruction (DI instruction), the interrupt prohibition instruction (DI instruction) below the specified priority, and the interrupt enable instruction (EI instruction), the priority of the interrupt prohibition will be stored. 0: Interrupt prohibition (default) of all priorities 1: interrupt prohibition of priorities 1 and 2; 2: interrupt prohibition of priority 2; 3: interrupt enable of all priorities	R	X
SD240	For STL: ON status number 1	In STL, the number of S device under ON status will be saved, with up to 8 numbers being saved; if there are more than eight storage numbers, the first eight small numbers will be saved.	R	X
SD241	For STL: ON status number 2		R	X
SD242	For STL: ON status number 3		R	X
SD243	For STL: ON status number 4		R	X
SD244	For STL: ON status number 5		R	X
SD245	For STL: ON status number 6		R	X
SD246	For STL: ON status number 7		R	X
SD247	For STL: ON status number 8		R	X
SD249	Minimum number of signal alarm ON state	Store the minimum number of signal alarm ON state	R/W	X
SD340	Timing clock output 1 counts the value in scanned numbers	The timing clock output 1 of DUTY instruction counts the value in scanned numbers	R/W	X
SD341	Timing clock output 2 counts the value in scanned numbers	The timing clock output 2 of DUTY instruction counts the value in scanned numbers	R/W	X
SD342	Timing clock output 3 counts the value in scanned numbers	The timing clock output 3 of DUTY instruction counts the value in scanned numbers	R/W	X
SD343	Timing clock output 4 counts the value in scanned numbers	The timing clock output 4 of DUTY instruction counts the value in scanned numbers	R/W	X
SD344	Timing clock output 5 counts the value in scanned numbers	The timing clock output 5 of DUTY instruction counts the value in scanned numbers	R/W	X

Interrupt prohibited

SD label	Name	Content	R/W	Power-down save
SD350~SD381	Timer interrupt prohibited mask	SIMASK instruction interrupt mask, each bit represents an interrupt, see SIMAK instruction for details	R/W	X
SD382~SD388	High-speed counter interrupt prohibited mask	SIMASK instruction interrupt mask, each bit represents an interrupt, see SIMAK instruction for details	R/W	X

High-speed input and output

SD label	Name	Content	R/W	Power-down save
SD400	HSC0 current count value low bit	Store the current count value of the channel high-speed counter, to be updated every 100us	R	X
SD401	HSC0 current count value high bit		R	X
SD402	HSC0 current frequency low bit	Store the current frequency of channel high-speed counter, to be updated every 100us	R	X
SD403	HSC0 current frequency high bit		R	X
SD405	HSC0 mode (display) 0: Common IO 1: Single phase count 2: AB phase count	Default: Common IO	R	X
SD420	HSC0 frequency multiplication (display) 1: 1 frequency multiplication 2: 2 frequency multiplication 4: 4 frequency multiplication	Default: 1 frequency multiplication	R	X
SD421	HSC0 frequency sampling time (ms)	Calculate the sampling time of the channel high-speed counter frequency, with the default time being 1000ms	R/W	X
SD422	HSC0 input filter setting (0.01us)	Range 0~1700, default 0	R	X
SD423	The DHSCS, DHSCR and DHSZ instructions use the priority settings of the HSC0 channel	0~2. The highest priority is 0, and the default value is 0	R/W	X
SD430	HSC1 current count value low bit	Store the current count value of the channel high-speed counter, to be updated every 100us	R	X
SD431	HSC1 current count value high bit		R	X
SD432	HSC1 current frequency low bit	Store the current frequency of channel high-speed counter, to be updated every 100us	R	X
SD433	HSC1 current frequency high bit		R	X
SD435	HSC1 mode (display) 0: Common IO 1: Single phase count 2: AB phase count	Default: Common IO	R	X
SD450	HSC1 frequency multiplication 1: 1 frequency multiplication 2: 2 frequency multiplication 4: 4 frequency multiplication	Default: 1 frequency multiplication	R	X
SD451	HSC1 frequency sampling time (ms)	Calculate the sampling time of the channel high-speed counter frequency, with the default time being 1000ms	R/W	X
SD452	HSC1 input filter setting (0.01us)	Range 0~1700, default 0	R	X
SD453	The DHSCS, DHSCR and DHSZ instructions use the priority settings of the HSC1 channel	0~2. The highest priority is 0, and the default value is 0	R/W	X
SD460	HSC2 current count value low bit	Store the current count value of the channel high-speed counter, to be updated every 100us	R	X
SD461	HSC2 current count value high bit		R	X
SD462	HSC2 current frequency low bit	Store the current frequency of channel high-speed counter, to be updated every 100us	R	X
SD463	HSC2 current frequency high bit		R	X
SD464	Reserved			
SD465	HSC2 mode (display) 0: Common IO 1: Single phase count 2: AB phase count	Default: Common IO	R	X
SD480	HSC2 frequency multiplication 1: 1 frequency multiplication 2: 2 frequency multiplication 4: 4 frequency multiplication	Default: 1 frequency multiplication	R	X
SD481	HSC2 frequency sampling time (ms)	Calculate the sampling time of the channel high-speed counter frequency, with the default time being 1000ms	R/W	X
SD482	HSC2 input filter setting (0.01us)	Range 0~1700, default 0	R	X

SD label	Name	Content	R/W	Power-down save
SD483	The DHSCS, DHSCR and DHSZ instructions use the priority settings of the HSC2 channel	0~2. The highest priority is 0, and the default value is 0	R/W	X
SD490	HSC3 current count value low bit	Store the current count value of the channel high-speed counter, to be updated every 100us	R	X
SD491	HSC3 current count value high bit		R	X
SD492	HSC3 current frequency low bit	Store the current frequency of channel high-speed counter, to be updated every 100us	R	X
SD493	HSC3 current frequency high bit		R	X
SD494	Reserved			X
SD495	HSC3 mode (display) 0: Common IO 1: Single phase count 2: AB phase count	Default: Common IO	R	X
SD510	HSC3 frequency multiplication 1: 1 frequency multiplication 2: 2 frequency multiplication 4: 4 frequency multiplication	Default: 1 frequency multiplication	R	X
SD511	HSC3 frequency sampling time (ms)	Calculate the sampling time of the channel high-speed counter frequency, with the default time being 1000ms	R/W	X
SD512	HSC3 input filter setting (0.01us)	Range 0~1700, default 0	R	X
SD513	The DHSCS, DHSCR and DHSZ instructions use the priority settings of the HSC3 channel	0~2. The highest priority is 0, and the default value is 0	R/W	X
SD521	HSC4 current count value high bit		R	X
SD522	HSC4 current frequency low bit	Store the current frequency of channel high-speed counter, to be updated every 100us	R	X
SD523	HSC4 current frequency high bit		R	X
SD524	Reserved			
SD525	HSC4 mode (display) 0: Common IO 1: Single phase count 2: AB phase count	Default: Common IO	R	X
SD540	HSC4 frequency multiplication 1: 1 frequency multiplication 2: 2 frequency multiplication 4: 4 frequency multiplication	Default: 1 frequency multiplication	R	X
SD541	HSC4 frequency sampling time (ms)	Calculate the sampling time of the channel high-speed counter frequency, with the default time being 1000ms	R/W	X
SD542	HSC4 input filter setting (0.01us)	Range 0~1700, default 0	R	X
SD543	The DHSCS, DHSCR and DHSZ instructions use the priority settings of the HSC4 channel	0~2. The highest priority is 0, and the default value is 0	R/W	X
SD550	HSC5 current count value low bit	Store the current count value of the channel high-speed counter, to be updated every 100us	R	X
SD551	HSC5 current count value high bit		R	X
SD552	HSC5 current frequency low bit	Store the current frequency of channel high-speed counter, to be updated every 100us	R	X
SD553	HSC5 current frequency high bit		R	X
SD554	Reserved			
SD555	HSC5 mode (display) 0: Common IO 1: Single phase count 2: AB phase count	Default: Common IO	R	X

SD label	Name	Content	R/W	Power-down save
SD570	HSC5 frequency multiplication 1: 1 frequency multiplication 2: 2 frequency multiplication 4: 4 frequency multiplication	Default: 1 frequency multiplication	R	X
SD571	HSC5 frequency sampling time (ms)	Calculate the sampling time of the channel high-speed counter frequency, with the default time being 1000ms	R/W	X
SD572	HSC5 input filter setting (0.01us)	Range 0~1700, default 0	R	X
SD573	The DHSCS, DHSCR and DHSZ instructions use the priority settings of the HSC5 channel	0~2. The highest priority is 0, and the default value is 0	R/W	X
SD580	HSC6 current count value low bit	Store the current count value of the channel high-speed counter, to be updated every 100us	R	X
SD581	HSC6 current count value high bit		R	X
SD582	HSC6 current frequency low bit	Store the current frequency of channel high-speed counter, to be updated every 100us	R	X
SD583	HSC6 current frequency high bit		R	X
SD584	Reserved			
SD585	HSC6 mode (display) 0: Common IO 1: Single phase count 2: AB phase count	Default: Common IO	R	X
SD600	HSC6 frequency multiplication 1: 1 frequency multiplication 2: 2 frequency multiplication 4: 4 frequency multiplication	Default: 1 frequency multiplication	R	X
SD601	HSC6 frequency sampling time (ms)	Calculate the sampling time of the channel high-speed counter frequency, with the default time being 1000ms	R/W	X
SD602	HSC6 input filter setting (0.01us)	Range 0~1700, default 0	R	X
SD603	The DHSCS, DHSCR and DHSZ instructions use the priority settings of the HSC6 channel	0~2. The highest priority is 0, and the default value is 0	R/W	X
SD610	HSC7 current count value low bit	Store the current count value of the channel high-speed counter, to be updated every 100us	R	X
SD611	HSC7 current count value high bit		R	X
SD612	HSC7 current frequency low bit	Store the current frequency of channel high-speed counter, to be updated every 100us	R	X
SD613	HSC7 current frequency high bit		R	X
SD614	Reserved			
SD615	HSC7 mode (display) 0: Common IO 1: Single phase count 2: AB phase count	Default: Common IO	R	X
SD630	HSC7 frequency multiplication 1: 1 frequency multiplication 2: 2 frequency multiplication 4: 4 frequency multiplication	Default: 1 frequency multiplication	R	X
SD631	HSC7 frequency sampling time (ms)	Calculate the sampling time of the channel high-speed counter frequency, with the default time being 1000ms	R/W	X
SD632	HSC7 input filter setting (0.01us)	Range 0~1700, default 0	R	X
SD633	The DHSCS, DHSCR and DHSZ instructions use the priority settings of the HSC7 channel	0~2. The highest priority is 0, and the default value is 0	R/W	X

Right Expansion module

SD label	Name	Content	R/W	Power-down save
SD2081		Reserved	R	×
SD2082		Reserved	R	×
SD2083	Number of expansion modules	Number of the current expansion modules connected	R	×
SD2084~SD2115	Information about expansion module model and version number	bit0~bit9: version number bit10~bit15: model 1/2: LX6_16EX module 3/4: LX6_16EYT module	R	×

Input filter

SD label	Name	Content	R/W	Power-down save
SD2280	Input filter point setting, with the default value being 10ms	Low byte X0~X3, high byte X4~X7	R/W	×
SD2281	Input filter point setting, with the default value being 10ms	Low byte X10~X13, high byte X14~X17	R/W	×
SD2282	Input filter point setting, with the default value being 10ms	Low byte X20~X23, high byte X24~X27	R/W	×
SD2283	Input filter point setting, with the default value being 10ms	Low byte X30~X33, high byte X34~X37	R/W	×
SD2284	Input filter point setting, with the default value being 10ms	Low byte X40~X43 , high byte ~X47	R/W	×
SD2285	Input filter point setting, with the default value being 10ms	Low byte X50~X53, high byte X54~X57	R/W	×
SD2286~SD2287	Input filter point setting, with the default value being 10ms	.....	R/W	×

Communication

SD label	Name	Content	R/W	Power-down save
Communication COM1				
SD2540	COM1 communication port settings	Default: baud rate: 115200, stop bit: 1, data bit: 8, parity bit: none See PORTPARA instruction description for specific setting method	R/W	√
SD2541	COM1 serial port parameter modification sign	If the serial port parameters need to be modified in RUN, the modification sign must be calculated. Only when the correct identification is written can the modification be successful. After making successful modification, the value will be automatically cleared. Please refer to PORTPARA instruction description for the specific calculation method.	R/W	√
SD2542	COM1 protocol settings	0H : Modbus slave 2H: Modbus RTU slave 3H: Modbus ASCII slave 10H User-defined protocol 20H: Modbus RTU master 30H: Modbus ASCII master	R/W	√
SD2543	COM1 protocol modification sign	If the communication protocol needs to be modified in RUN, the modification sign must be calculated. Only when the correct identification is written can the modification be successful. After	R/W	√

SD label	Name	Content	R/W	Power-down save
Communication COM1				
		making successful modification, the value will be automatically cleared. Please refer to PROTOCOL instruction description for the specific calculation method.		
SD2544	COM1 station number settings	Range: 0~255, default: 0	R/W	√
SD2545	COM1 station number modification sign	If the communication station number needs to be modified in RUN, the modification sign must be calculated. Only when the correct identification is written can the modification be successful. After making successful modification, the value will be automatically cleared. Please refer to STATION instruction description for the specific calculation method.	R/W	√
SD2546	COM1 sending interval 0.1ms		R/W	√
SD2547	COM1 communication timeout setting 10ms		R/W	√
SD2548	COM1Timeout retry count		R/W	√
SD2549	COM1 character interval timeout setting 0.1ms		R/W	√
SD2550	COM1 STX value		R/W	√
SD2551	COM1 ETX value		R/W	√
SD2555	PLC upload and download timeout: in case of timeout and failure in data transmission, the upload and download will be interrupted.	Unit: 100ms, default: 300 (30s)		√
SD2560	The amount of data received by COM1		R	√
SD2561	COM1last error		R	√
SD2562	COM1 current error		R	√
SD2563	COM1error steps		R	√
SD2564	COM1 error station number		R	√
SD2565	COM1 cumulative error times		R	√
SD2566	COM1error steps (double word)		R	√
SD2567			R	√
SD2568	Reserved		R	√
SD2569	Reserved		R	√
SD2570	Reserved		R	√
SD2571	Reserved		R	√
Communication COM2				
SD2590	COM2 communication port settings	Default: baud rate: 115200, stop bit: 1, data bit: 8, parity bit: none See PORTPARA instruction description for specific setting method	R/W	√
SD2591	COM2 serial port parameter modification sign	If the serial port parameters need to be modified in RUN, the modification sign must be calculated. Only when the correct identification is written can the modification be successful. After making successful modification, the value will be automatically cleared. Please refer to PORTPARA instruction description for the specific calculation method.	R/W	√
SD2592	COM2 protocol settings	0H : Modbus slave 2H: Modbus RTU slave 3H: Modbus ASCII slave 10H User-defined protocol 20H: Modbus RTU master 30H: Modbus ASCII master	R/W	√
SD2593	COM2 protocol modification sign	If the communication protocol needs to be	R/W	√

SD label	Name	Content	R/W	Power-down save
Communication COM1				
		modified in RUN, the modification sign must be calculated. Only when the correct identification is written can the modification be successful. After making successful modification, the value will be automatically cleared. Please refer to PROTOCOL instruction description for the specific calculation method.		
SD2594	COM2 station number settings	Range: 0~255, default: 0	R/W	√
SD2595	COM2 station number modification sign	If the communication station number needs to be modified in RUN, the modification sign must be calculated. Only when the correct identification is written can the modification be successful. After making successful modification, the value will be automatically cleared. Please refer to STATION instruction description for the specific calculation method.	R/W	√
SD2596	COM2 sending interval	Unit: 0.1ms, default: 0	R/W	√
SD2597	COM2 communication timeout setting	Unit: 10ms, default: 100ms	R/W	√
SD2598	COM2 timeout retry times	Default value: 0	R/W	√
SD2599	COM2 character interval timeout setting	Unit: 0.1ms, default: 10 (1ms)	R/W	√
SD2600	COM2 user-defined protocol start character	Default value: 0	R/W	√
SD2601	COM2 user-defined protocol end character	Default value: 0	R/W	√
SD2610	The amount of data received by COM2	Store the amount of data received by the serial port	R	×
SD2611	COM2 last error	Store the last displayed communication error code	R	×
SD2612	COM2 current error	Store the current communication error code	R	×
SD2613	COM2 error steps	Store the ladder diagram steps of the current communication error	R	×
SD2614	COM2 error station number	Store the station number of the current communication error	R	×
SD2615	COM2 cumulative error times	Store the cumulative communication error times	R	×

List of Ethernet-related special devices

SD label	Name	Content	R/W	Power-down save
SD2680	The first byte of the IP address	Local IP address	R/W	√
SD2681	The second byte of the ip address		R/W	√
SM2682	The third byte of the ip address		R/W	√
SD2683	The fourth byte of the ip address		R/W	√
SD2684	The first byte of subnet mask	Local subnet mask	R/W	√
SD2685	The second byte of subnet mask		R/W	√
SD2686	The third byte of subnet mask		R/W	√
SD2687	The fourth byte of subnet mask		R/W	√
SD2688	Default gateway first byte	Local default gateway	R/W	√
SD2689	Default gateway second byte		R/W	√
SD2690	Default gateway third byte		R/W	√
SD2691	Default gateway fourth byte		R/W	√
SD2692	The first byte of MAC address	Local MAC address	R/W	√
SD2693	The second byte of MAC address		R/W	√
SD2694	The third byte of MAC address		R/W	√
SD2695	The fourth byte of MAC address		R/W	√
SD2696	The fifth byte of MAC address		R/W	√
SD2697	The sixth byte of MAC address		R/W	√

Log information

SD label	Name	Content	R/W	Power-down save
SD4000	Low bit of ladder diagram writing times	Store the total number of downloads of ladder diagram, and save it at the time of power-down	R	√
SD4001	High bit of ladder diagram writing times		R	√
SD4002	Low bit of PLC parameter writing times	Store the total number of downloads of parameter, and save it at the time of power-down	R	√
SD4003	High bit of PLC parameter writing times		R	√
SD4004	Low bit of password writing times	Store the total number of password writing, and save it at the time of power-down	R	√
SD4005	High bit of password writing times		R	√
SD4006	Low bit of comment writing times	Store the total number of comment download, and save it at the time of power-down	R	√
SD4007	High bit of comment writing times		R	√
SD4008	Low bit of total startup times	Store the total times of PLC startup, and save it at the time of power-down	R	√
SD4009	High bit of total startup times		R	√
SD4010	Low bit of total startup times	Store the total time of PLC startup, and save it at the time of power-down, with the unit being s	R	√
SD4011	High bit of total startup times		R	√
SD4012	Low bit of the total boot RUN time	Store the total time of RUN after PLC starts up, and save it at the time of power-down, with the unit being s	R	√
SD4013	High bit of the total boot RUN time		R	√

SD label	Name	Content	R/W	Power-down save
SD4014	Low bit of this boot RUN time	Store the RUN time after PLC starts up, with the unit being s	R	×
SD4015	High bit of this boot RUN time		R	×

Appendix 3 Error codes sorting

PLC Hardware error

Error code	Description	Action	Processing approach	Test time
1000	PLC power supply voltage is abnormal	Stop running and PWR light is off	Prompt: the power supply is abnormal, please replace the power supply.	Always
1100	Watchdog timeout	Stop running	Program scan time exceeds the watchdog timeout time setting. Modify the setting value of the timeout time (SD122) or the program.	Always
1200	FLASH writing times exceeded the limit (upper computer read time information display, SD4000 double word)	Stop running	The number of FLASH writes exceeds 20,000, and the PLC needs to be replaced	When downloading the programs
1201	Failed to read production information	Stop running	FLASH is damaged, PLC needs to be replaced	When the state changes from STOP to RUN
1380	It is detected that the battery voltage is too low, which will affect the operation of PLC clock.	Continue to run	·Confirm the connection of the battery. Replace the battery in time.	When END instruction is executed
1382	User-defined exception	Continue to run	No processing required	When END instruction is executed

Ladder diagram execution error

Error code	Description	Action	Processing approach	Test time
1400	Program exception caused by STOP→RUN	Stop running	Check whether the parameter configuration is incorrectly configured, and whether the ladder diagram uses an unsupported instruction.	When the state changes from STOP to RUN
1401	Program exception caused by STOP	Stop running		At the time of STOP
1402	Program exception caused by ladder diagram execution	Stop running		Ladder diagram running
1403	Program abnormality caused by RUN→STOP	Stop running		When the state changes from RUN to STOP
1500	The sign error when executing OUT T in END instruction during the ladder diagram conversion	Stop running	Detect OUT T instruction in ladder diagram	When END instruction is executed
1501	Null pointer error	Stop running	Check whether an undefined program name is used.	At the time of initialization

PLC Parameter error

Error code	Description	Action	Processing approach	Test time
2000	Number of I/O points allocated by program is different from the actual number of hardware I/O points	Stop running	Check I/O points configuration	When the state changes from STOP to RUN
2001	Set the parameters of the standard input and output module for the high-speed pulse input and output module	Stop running	Check input point parameter configuration	When the state changes from STOP to RUN
2002	The installed expansion module exceeds the maximum number	Stop running	Reduce the installation of expansion modules	When the state changes from STOP to RUN
2003	X point multiplexing, the same point is used as AB phase high-speed input, but also as one-way input or interrupt input	Stop running	Check input mode configuration	When the state changes from STOP to RUN
2004	Configure high-speed input IO error, CNTCFG instruction parameter write	Stop running	Check the value of parameter 1 of CNTCFG	Instruction is running

Error code	Description	Action	Processing approach	Test time
2100	Memory capacity setting error	Stop running	Check memory capacity settings	When the state changes from STOP to RUN
2101	Wrong setting of holding area	Stop running	Check holding register settings	When the state changes from STOP to RUN
2102	Comment area setting error	Stop running	Check the comment setting	When the state changes from STOP to RUN
2103	File register area setting error	Stop running	Check file storage area settings	When the state changes from STOP to RUN
2200	Inconsistent program verification	Stop running	The upper and lower computer programs are inconsistent, please upload or download again	When the state changes from STOP to RUN
2201	Inconsistent checksums of special parameters	Stop running	The upper and lower computer parameters are inconsistent, please upload or download again	When the state changes from STOP to RUN
2202	Special parameter setting error	Stop running	Check special parameters settings	When the state changes from STOP to RUN
2203	PLC EDITOR and PLC firmware versions are inconsistent	Stop running	Please re-install the PLC Editor2 or upgrade the firmware by checking the correspondence table between the PLC Editor2 and the firmware version.	When the state changes from STOP to RUN
2380	The current scan cycle exceeds the constant scan cycle set value	Continue to run	Modify the constant scan cycle setting	When END instruction is executed
2400	Event exceeds maximum range	Stop running	Check whether the event setting exceeds 100	When the state changes from STOP to RUN
2401	Event execution program is empty	Stop running	Whether to establish the correct event program	When the state changes from STOP to RUN
2402	Event clear program is empty	Stop running	Whether to establish the correct event program	When the state changes from STOP to RUN
2403	Timed interrupt exceeds the maximum range	Stop running	Check whether the timer interrupt setting exceeds 100	When the state changes from STOP to RUN
2404	Timer interrupt execution program is empty	Stop running	Whether to establish the correct timer interrupt program	When the state changes from STOP to RUN
2405	Timer interrupt priority setting error	Stop running	Check whether the timer interrupt priority is set to 0~2	When the state changes from STOP to RUN
2406	High-speed counting interrupt exceeds the maximum range	Stop running	Check whether the high-speed counting interrupt setting exceeds 100	When the state changes from STOP to RUN
2407	High-speed counting interrupt execution program is empty	Stop running	Whether to establish the correct high-speed counting interrupt program	When the state changes from STOP to RUN

Error code	Description	Action	Processing approach	Test time
2408	High-speed counting priority setting error	Stop running	Check whether the high-speed counting interrupt priority is set to 0~2	When the state changes from STOP to RUN
2409	External interrupt exceeds the maximum range	Stop running	Check whether the external interrupt setting exceeds 16	When the state changes from STOP to RUN
240A	External interrupt execution program is empty	Stop running	Whether to establish the correct external interrupt program	When the state changes from STOP to RUN
240B	External interrupt priority setting error	Stop running	Check whether the external interrupt priority is set to 0~2	When the state changes from STOP to RUN
2500	High-speed counter channel exceeds the maximum range	Stop running	Detect high-speed input configuration	When the state changes from STOP to RUN
2501	High-speed counter mode setting error	Stop running	Detect high-speed input configuration	When the state changes from STOP to RUN
2502	High-speed counter frequency multiplication setting error	Stop running	Detect high-speed input configuration	When the state changes from STOP to RUN
2503	The counting direction of the high-speed counter is set incorrectly	Stop running	Detect high-speed input configuration	When the state changes from STOP to RUN
2504	The high-speed counter interrupt is used, but the OUT HSC instruction is not used to turn on the high-speed counter, and the value of the HSC device is modified	Continue to run	Check if there is a contact that closed the OUT HSC instruction	100us interrupt execution
2580	After the high-speed counter is turned on, but the axis high-speed counter enable is not configured	Continue to run	View project management -> parameters -> high-speed counting configuration	When OUT HSC instruction is executed
2581	The high-speed counter interrupt is used, but the OUT HSC instruction is not used to turn on the high-speed counter, and the value of the HSC device is modified	Continue to run	Check if there is a contact that closed the OUT HSC instruction	100us interrupt execution
2582	The REF instruction is used to refresh the high-speed count value, but there is no OUT HSC instruction to turn on the high-speed counter of the channel	Continue to run	1. Check project management →parameters →high-speed counting configuration →whether to use 2. Check if there is a contact that closed the OUT HSC instruction	When the REF instruction is executed
2600	The input point X is not available for both high-speed counter and interrupt program.	Stop running	Detect high-speed input or external interrupt configuration	When the state changes from STOP to RUN

EtherCAT Configuration error

Error code	Description	Action	Processing approach	Test time
2781	The EtherCAT configuration file failed to load. The file is corrupted.	Continue to run	Please download the EtherCAT configuration file again.	At the time of initialization of EtherCAT
2782	The EtherCAT configuration file could not be found.	Continue to run	Please download the EtherCAT configuration file again.	At the time of initialization of EtherCAT
2783	Failed to open EtherCAT configuration file.	Continue to run	Please download the EtherCAT configuration file again.	At the time of initialization of EtherCAT
2784	Failed to parse EtherCAT configuration, and no corresponding value or parameter was found.	Continue to run	Please download the EtherCAT configuration file again, or check whether the versions of EtherCAT upper and lower computers match.	At the time of initialization of EtherCAT
2785	Initialization failed.	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT
2786	Failed to get slave device.	Continue to run	Please check whether there are linked EtherCAT slave devices.	At the time of initialization of EtherCAT
2787	Slave parameter setting failed.	Continue to run	Check whether the configuration is correct.	At the time of initialization of EtherCAT
2788	EtherCAT slave device activation failed.	Continue to run	Check whether the configuration is correct.	At the time of initialization of EtherCAT
2789	EtherCAT startup failed.	Continue to run	Check whether the configuration is correct, or contact the manufacturer.	At the time of initialization of EtherCAT
278A	The number of configured EtherCAT slaves does not match the number of linked slaves.	Continue to run	Check the settings and the number of linked EtherCAT slave devices.	At the time of initialization of EtherCAT
278B	The accessed servo model is inconsistent with the model in EtherCAT configuration table.	Continue to run	Check whether the configuration used matches the actual model.	At the time of initialization of EtherCAT
278C	Initialization failed.	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT
278D	Initialization failed.	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT
278E	ETHERCAT stop error	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT
278F	Initialization failed.	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT
2790	Initialization failed.	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT
2791	Initialization timeout	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT
2792	Initialization timeout	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT
2793	Initialization timeout	Continue to run	Please contact manufacturer.	At the time of initialization of EtherCAT

PLC Communication error

Error code	Description	Action	Processing approach	Test time
3080	COM1 data reception error	Continue to run	There may be interference on the communication cable, it is recommended to connect GND.	When the instruction is executed
3081	COM1 data reception timeout	Continue to run	1. Check the wiring and check whether the serial port parameter settings correspond to master and slave station. 2. Check whether there is interference. 3. Check whether it is caused by the late response from the slave station. It is recommended to increase the sending interval SD2546.	When the instruction is executed
3082	COM1 CRC parity error	Continue to run	There may be interference on the communication cable, it is recommended to connect GND.	When the instruction is executed
3083	COM1 LRC parity error	Continue to run	There may be interference on the communication cable, it is recommended to connect GND.	When the instruction is executed
3084	COM1 Station number configuration error	Continue to run	Check the slave station number setting, and check whether there is any problem with the receiving and sending mechanism from slave station.	When the instruction is executed
3085	COM1 send buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
3086	COM1 function code error	Continue to run	Check whether the set function code is a function code supported by PLC	When the instruction is executed
3087	COM1 wrong address	Continue to run	Check whether the slave station has this address. (Please refer to Modbus exception 02).	When the instruction is executed
3088	COM1 wrong length	Continue to run	Check whether the communication length exceeds the Modbus range	When the instruction is executed
3089	COM1 data error	Continue to run	Check whether parameters of the instruction are wrong. Check whether the set value is supported by the slave station. (Please refer to Modbus exception 03).	When the instruction is executed
308A	COM1 slave station is busy.	Continue to run	Message returned from slave station: Slave station is busy. (Please refer to Modbus exception 06).	When the instruction is executed
308B	COM1 slave station does not support function code	Continue to run	Check whether this function code is supported by the slave station. (Please refer to Modbus exception 01).	When the instruction is executed
308C	COM1 slave failure	Continue to run	Message returned from slave station: Slave station failed, please check whether slave station failed. (Please refer to Modbus exception 04).	When the instruction is executed
308D	COM1 slave confirmation	Continue to run	Message returned from slave station: slave station confirmation. (Please refer to Modbus exception 05).	When the instruction is executed
308E	COM1 does not currently support this instruction or function.	Continue to run	When the slave protocol is set, the communication instructions related to master station or master station functions cannot be used. Please change the protocol or close the contact before the corresponding instruction or close the corresponding communication function.	When the instruction is executed
308F	COM1 send timeout	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
3090	Receive buffer overflow	Continue to run	Check if the other device is sending data all the time	When the instruction is executed

Error code	Description	Action	Processing approach	Test time
30A0	No available gateway for COM1	Continue to run	Message returned from the slave station: no available gateway. (Please refer to Modbus exception 0A).	When the instruction is executed
30A1	COM1 indicates that no response was obtained from the target device. It usually means the device is not on the network.	Continue to run	Message returned from the slave station: The device is not networked. (Please refer to Modbus Exception 0B).	When the instruction is executed
3180	COM2 data reception error	Continue to run	There may be interference on the communication cable, it is recommended to connect GND.	When the instruction is executed
3181	COM2 data reception timeout	Continue to run	1. Check the wiring and check whether the serial port parameter settings correspond to master and slave station. 2. Check whether there is interference. 3. Check whether it is caused by the late response from the slave station. It is recommended to increase the sending interval SD2546.	When the instruction is executed
3182	COM2 CRC parity error	Continue to run	There may be interference on the communication cable, it is recommended to connect GND.	When the instruction is executed
3183	COM2 LRC parity error	Continue to run	There may be interference on the communication cable, it is recommended to connect GND.	When the instruction is executed
3184	COM2 station number configuration error	Continue to run	Check the slave station number setting, and check whether there is any problem with the receiving and sending mechanism from slave station.	When the instruction is executed
3185	COM2 send buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
3186	COM2 function code error	Continue to run	Check whether the set function code is a function code supported by PLC	When the instruction is executed
3187	COM2 address error	Continue to run	Check whether the slave station has this address. (Please refer to Modbus exception 02).	When the instruction is executed
3188	COM2 length error	Continue to run	Check whether the communication length exceeds the Modbus range	When the instruction is executed
3189	COM2 data error	Continue to run	Check whether parameters of the instruction are wrong. Check whether the set value is supported by the slave station. (Please refer to Modbus exception 03).	When the instruction is executed
318A	Busy COM2 slave station	Continue to run	Message returned from slave station: Slave station is busy. (Please refer to Modbus exception 06).	When the instruction is executed
318B	Function code not supported by COM2 slave station	Continue to run	Check whether this function code is supported by the slave station. (Please refer to Modbus exception 01).	When the instruction is executed
318C	Faulty COM2 slave station	Continue to run	Message returned from slave station: Slave station failed, please check whether slave station failed. (Please refer to Modbus exception 04).	When the instruction is executed
318D	COM2 slave station confirmation	Continue to run	Message returned from slave station: slave station confirmation. (Please refer to Modbus exception 05).	When the instruction is executed

Error code	Description	Action	Processing approach	Test time
318E	At present, COM2 does not support this instruction or function.	Continue to run	When the slave protocol is set, the communication instructions related to master station or master station functions cannot be used. Please change the protocol or close the contact before the corresponding instruction or close the corresponding communication function.	When the instruction is executed
318F	COM2 sending timeout.	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
31A0	No available gateway for COM2	Continue to run	Message returned from the slave station: no available gateway. (Please refer to Modbus exception 0A).	When the instruction is executed
31A1	COM2 indicates that no response was obtained from the target device. It usually means the device is not on the network.	Continue to run	Message returned from the slave station: The device is not networked. (Please refer to Modbus Exception 0B).	When the instruction is executed
31C3	The number of instructions is out of range. The current limit is 1~255.	Continue to run	Check the number of instructions	Application instruction or When the function is executed
31C6	The device used by the table instruction is out of device range.	Continue to run	Check the device used by each instruction in the table	Application instruction or When the function is executed

PLC Operation error

Error code	Description	Action	Processing approach	Test time
3680	Ethernet data reception error	Continue to run	Check whether there is any interference in the environment	When the instruction is executed
3681	Ethernet data reception timeout	Continue to run	Check if the network cable is loosened 1. Check whether the network peer is faulty, which results in failure in sending data. Check whether the network peer does not respond in time, which results in any delay in data. Under such circumstance, the reception timeout in the socket configuration can be increased.	When the instruction is executed
3684	Modbus TCP station number configuration error	Continue to run	Check slave station number setting. Check whether there is any problem with the receiving and sending mechanism of slave station.	When the instruction is executed
3685	Modbus TCP send buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
3686	Modbus TCP function code error	Continue to run	Check whether the set function code is a function code supported by PLC	When the instruction is executed

Error code	Description	Action	Processing approach	Test time
3687	Modbus TCP address error	Continue to run	Check whether the slave station has this address. (Please refer to Modbus exception 02).	When the instruction is executed
3688	Modbus TCP length error	Continue to run	Check whether the communication length exceeds the Modbus range	When the instruction is executed
3689	Modbus TCP data error	Continue to run	Check whether parameters of the instruction are wrong. Check whether the set value is supported by the slave station. (Please refer to Modbus exception 03).	When the instruction is executed
368A	Modbus TCP slave is busy	Continue to run	Message returned from slave station: Slave station is busy. (Please refer to Modbus exception 06).	When the instruction is executed
368B	Modbus TCP slave station does not support function code	Continue to run	Check whether this function code is supported by the slave station. (Please refer to Modbus exception 01)	When the instruction is executed
368C	Modbus TCP slave failure	Continue to run	Message returned from slave station: Slave station failed, please check whether slave station failed. (Please refer to Modbus exception 04).	When the instruction is executed
368D	Modbus TCP slave confirmation	Continue to run	Message returned from slave station: slave station confirmation. (Please refer to Modbus exception 05)	When the instruction is executed
368E	The current Modbus TCP protocol does not support this instruction	Continue to run	The RS instruction cannot be used when the slave protocol is set, please change the protocol or close the contact before the RS instruction	When the instruction is executed
368F	Network port send timeout	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
3690	Receive buffer overflow	Continue to run	Check if the other device is sending data all the time	When the instruction is executed
36A0	No available gateway for Modbus TCP	Continue to run	Message returned from the station: no available gateway. (Please refer to Modbus exception 0A). (Please refer to Modbus exception 0A)	When the instruction is executed
36A1	Modbus TCP indicates that no response was obtained from the target device. It usually means the device is not on the network.	Continue to run	Message returned from the slave station: The device is not in the network. (Please refer to Modbus exception 0B)	When the instruction is executed
36C0	ModbusTCP transaction identifier error	Continue to run	Check if there is network congestion and data can not be received in time.	When the instruction is executed

Error code	Description	Action	Processing approach	Test time
36C1	Modbus TCP server available links are full	Continue to run	Check if SD2702 and SD2703 have too many clients to link	When the instruction is executed
36C8	Insufficient space in the Ethernet stack	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36C9	Connection limit exceeded	Continue to run	Confirm whether the total number of connections exceeds the limit	When the instruction is executed
36CA	The last send is not completed	Continue to run	Use the sending completion sign to determine that the current sending is completed, and then send the next one	When the instruction is executed
36CB	TCP write exception	Continue to run	Use the sign bit device to judge whether the current connection status is normal. If it is not normal, no data transmission will be performed. For example, after the closing sign is set, data transmission will not be performed.	When the instruction is executed
36CC	TCP output exception	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36CD	IP address is used	Continue to run	Please check if a connection exists with the same address information	When the instruction is executed
36CE	Error in server's acceptance of connection	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36CF	TCP receive buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36D0	TCP connection failed	Continue to run	It may be caused by opening the TCP client when the network cable is not connected.	When the instruction is executed
36D1	Exception when actively closing the connection	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36D2	Abnormal shutdown occurred inside the protocol stack	Continue to run	May be shut down due to no answering for a long time. Check whether the peer is online and can ping.	When the instruction is executed
36D3	The peer initiates an RST link	Continue to run	1. Check whether the peer initiates abnormal shutdown. 2. As a client, the number of opposite connections is full, or the opposite port is not open.	When the instruction is executed
36D4	The protocol stack has a single-ended shutdown	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36D5	There is an IP address conflict	Continue to run	The same IP device exists in the LAN, please change the IP address	When the instruction is executed
36D6	There is a MAC address conflict	Continue to run	The same MAC device exists in the LAN, please change the MAC address	When the instruction is executed

Error code	Description	Action	Processing approach	Test time
36D7	TCP send buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36D8	UDP connection exception	Continue to run	IP address or port number may has been used.	When the instruction is executed
36D9	UDP send buffer overflow	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36DA	Insufficient memory space when sending UDP	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36DB	UDP send failed	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36DC	UDP memory release failed	Continue to run	If this error message appears, please contact technical personnel	When the instruction is executed
36DD	UDP receive buffer overflow	Continue to run	The length of the data received by UDP exceeds the limit value of 512.	When the instruction is executed
4080	Divisor is 0 in division instruction	Continue to run	Modify application instruction parameters	When the application instruction is executed
4081	Application instruction calculation data overflow	Continue to run	Modify application instruction parameters	When the application instruction is executed
4082	A data type that cannot be converted is entered in the application instruction	Continue to run	Modify application instruction parameters	When the application instruction is executed
4083	Any data of -0, non-normalized number, non-number, and ±∞ is input in the application instruction	Continue to run	Modify application instruction parameters	When the application instruction is executed
4084	Data beyond the specified range is entered in the application instruction (for example, parameter 1 is specified as 0~1, setting 2)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4085	The output result in the read application instruction exceeds device range (for example, the maximum output result of the D device is D7999, but D8000 is used)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4086	The output result in the write application instruction exceeds device range (for example, the maximum output result of the D device is D7999, but D8000 is used)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4087	The application instruction parameter uses an unsupported device	Continue to run	Modify application instruction parameters	When the application instruction is executed
4088	Several application instructions use the same axis at the same time and have been executed.	Continue to run	Modify application instruction parameters	When the application instruction is executed
4089	The number of application instructions exceeds the limit.	Continue to run	Check whether a restricted instruction is used in the ladder diagram and exceeds the limit	When the application instruction is executed

Error code	Description	Action	Processing approach	Test time
408A	The read length of the string exceeds the limit, the continuous length of the string exceeds the limit (currently 400) or exceeds the limit within the instruction	Continue to run	Modify the read string length	When the application instruction is executed
408B	When the character string is read, the maximum range of device is read, but 00H is not found.	Continue to run	View string end character	When the application instruction is executed
408E	Multiple application instruction parameters use the same device, but the instruction does not allow reuse of device	Continue to run	Check whether the DUTY instruction uses the same SM for output	When the application instruction is executed
408F	The firmware used does not support this instruction, please upgrade it to the latest firmware	Continue to run	Upgrade the firmware that includes the instruction	When the application instruction is executed
4090	The order of two parameters used to designate the range is abnormal. For example, it is allowed to use D10 D100 other than D100 D10 to specify the range.	Continue to run	Check and modify the parameters of instructions.	When the application instruction is executed
4100	The number of FOR~NEXT instructions does not correspond or FOR~NEXT exceeds the maximum nesting level	Stop running	Modify the corresponding relationship of the ladder diagram FOR~NEXT instructions	When NEXT and END instruction is executed
4180	No CJ and CALL jump destination address, result of indexing modification and label undefined; P63 is executed in CALL instruction beyond the range of P0~P4095. P63 cannot be used in the CALL instruction because it is the tag for jumping to END	Continue to run	Modify application instruction parameters	When the application instruction is executed
4181	CJ instruction exceeds the maximum nesting level	Continue to run	Modify application instruction parameters	When the application instruction is executed
4102	CALL instruction exceeds the maximum nesting level	Continue to run	Modify application instruction parameters	When the application instruction is executed
4183	Interrupt exceeds the maximum nesting level	Continue to run	Modify application instruction parameters	When the application instruction is executed
4185	EI instruction popping error	Continue to run	Modify application instruction parameters	When the application instruction is executed
4186	BREAK is not in the FOR~NEXT instruction	Continue to run	Modify application instruction parameters	When the application instruction is executed
4187	MC~MCR exceeds the maximum nesting range	Continue to run	View the nesting relationship of MC and MCR	When the application instruction is executed
4188	When using N in the MC nesting structure, the order from small to large is not followed	Continue to run	Modify the N nesting corresponding to MC	When the application instruction is executed
4189	SIMASK instruction specifies an unset interrupt	Continue to run	Modify the interrupt name specified by SIMASK or the interrupt configuration	When the application instruction is executed

Error code	Description	Action	Processing approach	Test time
4D80	Sampling time (Ts) exceeds the target range (Ts≤0)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D81	The input filter constant (α) exceeds the target range (α<0 or 1023<α)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D82	The maximum ascent rate (deltaT) exceeds the target range (deltaT<0 or 32000<deltaT)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D83	Proportional gain (Kp) exceeds the target range (Kp<0)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D84	Integral gain (Ki) exceeds the target range (Ki<0)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D85	Differential gain (Kd) exceeds the target range (Kd<0)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D86	Sampling time (Ts)<operation cycle	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D87	Proportional gain (Kp) exceeds the target range (Kp<1 or Kp>3000)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D88	The integration time (Ti) exceeds the target range (Ti<0 or Ti>3600)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D89	The derivative time (Td) exceeds the target range (Td<0 or Td>1000)	Continue to run	Modify application instruction parameters	When the application instruction is executed
4D90	PID output upper limit is less than lower limit	Continue to run	Modify application instruction parameters	When the application instruction is executed
4E80	E-CAM table loading error	Continue to run	Modify application instruction parameters	Instruction is running
4E81	The currently numbered table has a cam in use	Continue to run	Modify application instruction parameters	Instruction is running
4E82	Table address error	Continue to run	Modify application instruction parameters	Instruction is running
4E83	The table exceeds device range	Continue to run	Modify application instruction parameters	Instruction is running
4EC0	Electronic gear ratio setting error	Continue to run	Modify application instruction parameters	Instruction is running
4F80	Minimum range of DHSZ instruction >= maximum range	Continue to run	Modify application instruction parameters	Instruction is running
4F81	The DHSCS, DHSCR and DHSZ instructions are enabled, but the OUT HSC instruction is not used to enable the high-speed counter to count.	Continue to run	1. Check project management →parameters →high-speed counting configuration →whether to use 2. Check if there is a contact that closed the OUT HSC instruction	Instruction is running
4F90	In radius mode, the radius is not within the range.	Continue to run	Modify application instruction parameters	Instruction is running
4F92	In the center of circle/radius mode, the large deviation between the set end point coordinates and the theoretical end point on the circle causes quadrant calculation errors.	Continue to run	Modify application instruction parameters	Instruction is running

Error code	Description	Action	Processing approach	Test time
4F93	In radius mode, the chord length is greater than the diameter.	Continue to run	Modify application instruction parameters	Instruction is running
4F95	In the center of circle mode, the distance between the center and the start point, and the distance between the center and the end point are not within the range of [-10-10].	Continue to run	Modify application instruction parameters	Instruction is running
4F97	In the center of circle mode, the calculated radius distance is greater than the maximum radius, which means the range is +/-800,000 pulses.	Continue to run	Modify application instruction parameters	Instruction is running
4F98	Helical interpolation error, the Z axis is the main axis (the Z axis coordinate is greater than the number of virtual main axes in the circular plane).	Continue to run	Modify application instruction parameters	Instruction is running
4F99	Helical interpolation error, Z axis is 0.	Continue to run	Modify application instruction parameters	Instruction is running
4F9B	The lead setting is out of range (i.e. lead appears $K \leq 0$).	Continue to run	Modify application instruction parameters	Instruction is running
5080	The ethernet socket is already connected and cannot be opened again.	Continue to run	Check if SOCOPEN instruction is executed repeatedly	When the application instruction is executed
5081	Ethernet socket is not open, which results in the failure in operation	Continue to run	Check whether set ON the connected sign bit of the SOCOPEN instruction (d2) parameter.	When the application instruction is executed
5082	Socket ID input by Ethernet instruction is out of range.	Continue to run	Modify application instruction parameters	When the application instruction is executed
5083	Failed to create TCP server	Continue to run	Check if the connection is full	When the application instruction is executed
5084	Failed to create link	Continue to run	Check if the connection is full	When the application instruction is executed
5086	The socket ID used by the Ethernet instruction is not configured by the upper computer or is not enabled after configuration.	Continue to run	Check the Ethernet configuration of upper computer	When the application instruction is executed

Error code	Description	Action	Processing approach	Test time
5087	SOCRECV instruction reception timeout	Continue to run	1. Check whether the network connection is normal. 2. Check whether there is data sent by the network peer.	When the application instruction is executed
5088	The SOCMTCP instruction specifies that the socket uses the configuration mode of a non-TCP client	Continue to run	Check the Ethernet configuration of upper computer	When the application instruction is executed
5089	When configuring the TCP server for the Ethernet socket, specify the local port as 502	Continue to run	Port 502 has been occupied internally, please modify the local port.	When the application instruction is executed
508A	The UDP port is specified as 1092	Continue to run	UDP port: 1092 is occupied by the system and cannot be used, please modify the local port number.	When the application instruction is executed
5090	Abnormal network cable connection	Continue to run	Check if the network cable is connected	When the application instruction is executed

Right Expansion Module Error (Communication Error Reported)

Error code	Description	Action	Processing approach	Test time
7080	Expansion module checksum error	Continue to run	Detect the link between the expansion module and the host or whether there is external interference	Instruction is running
7081	Expansion module communication message abnormal	Continue to run	Detect the link between the expansion module and the host or whether there is external interference	Instruction is running
7082	FROM/TO instruction error	Continue to run	Detect the link between the expansion module and the host	Instruction is running
7083	Expansion module access abnormal	Continue to run	Detect the link between the expansion module and the host	Instruction is running
7084	Number of configured expansion modules does not match the actual number	Stop run	Check if the configuration is correct	Instruction is running
7110	Expansion module configuration error	Stop run	Check if the expansion module configuration is correct	Instruction is running

Appendix 4 ASCII code comparison table

ASCII Code comparison table

Bin (Binary)	Oct (Octal)	(Dec) (Decimal)	Hex (Hexadecimal)	Abbreviations/characters	Explanation
0000 0000	0	0	0x00	NUL(null)	Null character
0000 0001	1	1	0x01	SOH (start of headline)	Start of headline
0000 0010	2	2	0x02	STX (start of text)	Start of text
0000 0011	3	3	0x03	ETX (end of text)	End of text
0000 0100	4	4	0x04	EOT (end of transmission)	End of transmission
0000 0101	5	5	0x05	ENQ (enquiry)	Enquiry
0000 0110	6	6	0x06	ACK (acknowledge)	Acknowledge
0000 0111	7	7	0x07	BEL (bell)	Bell
0000 1000	10	8	0x08	BS (backspace)	Backspace
0000 1001	11	9	0x09	HT (horizontal tab)	Horizontal tab
0000 1010	12	10	0x0A	LF (NL line feed, new line)	Line feed
0000 1011	13	11	0x0B	VT (vertical tab)	Vertical tab
0000 1100	14	12	0x0C	FF (NP form feed, new page)	Form feed
0000 1101	15	13	0x0D	CR (carriage return)	Carriage return
0000 1110	16	14	0x0E	SO (shift out)	Shift out
0000 1111	17	15	0x0F	SI (shift in)	Shift in
0001 0000	20	16	0x10	DLE (data link escape)	Data link escape
0001 0001	21	17	0x11	DC1 (device control 1)	Device control 1
0001 0010	22	18	0x12	DC2 (device control 2)	Device control 2
0001 0011	23	19	0x13	DC3 (device control 3)	Device control 3
0001 0100	24	20	0x14	DC4 (device control 4)	Device control 4
0001 0101	25	21	0x15	NAK (negative acknowledge)	Negative acknowledge
0001 0110	26	22	0x16	SYN (synchronous idle)	Synchronous idle
0001 0111	27	23	0x17	ETB (end of trans. block)	End of transmission block
0001 1000	30	24	0x18	CAN (cancel)	Cancel
0001 1001	31	25	0x19	EM (end of medium)	End of medium
0001 1010	32	26	0x1A	SUB (substitute)	Substitute
0001 1011	33	27	0x1B	ESC (escape)	Escape
0001 1100	34	28	0x1C	FS (file separator)	File separator
0001 1101	35	29	0x1D	GS (group separator)	Group separator
0001 1110	36	30	0x1E	RS (record separator)	Record separator
0001 1111	37	31	0x1F	US (unit separator)	Unit separator
0010 0000	40	32	0x20	(space)	Space
0010 0001	41	33	0x21	!	Exclamation mark
0010 0010	42	34	0x22	"	Double quotation marks
0010 0011	43	35	0x23	#	Pound sign
0010 0100	44	36	0x24	$	Dollar sign
0010 0101	45	37	0x25	%	Percent sign
0010 0110	46	38	0x26	&	Ampersand

Bin (Binary)	Oct (Octal)	(Dec) (Decimal)	Hex (Hexadecimal)	Abbreviations/characters	Explanation
0010 0111	47	39	0x27	'	Single quotation mark
0010 1000	50	40	0x28	(	Opening bracket
0010 1001	51	41	0x29	)	Closing bracket
0010 1010	52	42	0x2A	*	Asterisk
0010 1011	53	43	0x2B	+	Plus sign
0010 1100	54	44	0x2C	,	**Comma**
0010 1101	55	45	0x2D	-	Subtraction sign/dash
0010 1110	56	46	0x2E	.	Full stop
0010 1111	57	47	0x2F	/	Slash
0011 0000	60	48	0x30	0	Character 0
0011 0001	61	49	0x31	1	Character 1
0011 0010	62	50	0x32	2	Character 2
0011 0011	63	51	0x33	3	Character 3
0011 0100	64	52	0x34	4	Character 4
0011 0101	65	53	0x35	5	Character 5
0011 0110	66	54	0x36	6	Character 6
0011 0111	67	55	0x37	7	Character 7
0011 1000	70	56	0x38	8	Character 8
0011 1001	71	57	0x39	9	Character 9
0011 1010	72	58	0x3A	:	Colon
0011 1011	73	59	0x3B	:	Semicolon
0011 1100	74	60	0x3C	<	Less-than sign
0011 1101	75	61	0x3D	=	Equal sign
0011 1110	76	62	0x3E	>	Greater-than sign
0011 1111	77	63	0x3F	?	Question mark
0100 0000	100	64	0x40	@	Email symbol
0100 0001	101	65	0x41	A	Capital letter A
0100 0010	102	66	0x42	B	Capital letter B
0100 0011	103	67	0x43	C	Capital letter C
0100 0100	104	68	0x44	D	Capital letter D
0100 0101	105	69	0x45	E	Capital letter E
0100 0110	106	70	0x46	F	Capital letter F
0100 0111	107	71	0x47	G	Capital letter G
0100 1000	110	72	0x48	H	Capital letter H
0100 1001	111	73	0x49	I	Capital letter I
1001010	112	74	0x4A	J	Capital letter J
0100 1011	113	75	0x4B	K	Capital letter K
0100 1100	114	76	0x4C	L	Capital letter L
0100 1101	115	77	0x4D	M	Capital letter M
0100 1110	116	78	0x4E	N	Capital letter N
0100 1111	117	79	0x4F	O	Capital letter O
0101 0000	120	80	0x50	P	Capital letter P

Bin (Binary)	Oct (Octal)	(Dec) (Decimal)	Hex (Hexadecimal)	Abbreviations/characters	Explanation
0101 0001	121	81	0x51	Q	Capital letter Q
0101 0010	122	82	0x52	R	Capital letter R
0101 0011	123	83	0x53	S	Capital letter S
0101 0100	124	84	0x54	T	Capital letter T
0101 0101	125	85	0x55	U	Capital letter U
0101 0110	126	86	0x56	V	Capital letter V
0101 0111	127	87	0x57	W	Capital letter W
0101 1000	130	88	0x58	X	Capital letter X
0101 1001	131	89	0x59	Y	Capital letter Y
0101 1010	132	90	0x5A	Z	Capital letter Z
0101 1011	133	91	0x5B	[	Open bracket
0101 1100	134	92	0x5C	/	Backslash
0101 1101	135	93	0x5D	]	Close bracket
0101 1110	136	94	0x5E	^	Caret
0101 1111	137	95	0x5F	_	Underline
0110 0000	140	96	0x60	`	Open single quotation mark
0110 0001	141	97	0x61	a	Lowercase letter a
0110 0010	142	98	0x62	b	Lowercase letter b
0110 0011	143	99	0x63	c	Lowercase letter c
0110 0100	144	100	0x64	d	Lowercase letter d
0110 0101	145	101	0x65	e	Lowercase letter e
0110 0110	146	102	0x66	f	Lowercase letter f
0110 0111	147	103	0x67	g	Lowercase letter g
0110 1000	150	104	0x68	h	Lowercase letter h
0110 1001	151	105	0x69	i	Lowercase letter i
0110 1010	152	106	0x6A	j	Lowercase letter j
0110 1011	153	107	0x6B	k	Lowercase letter k
0110 1100	154	108	0x6C	l	Lowercase letter l
0110 1101	155	109	0x6D	m	Lowercase letter m
0110 1110	156	110	0x6E	n	Lowercase letter n
0110 1111	157	111	0x6F	o	Lowercase letter o
0111 0000	160	112	0x70	p	Lowercase letter p
0111 0001	161	113	0x71	q	Lowercase letter q
0111 0010	162	114	0x72	r	Lowercase letter r
0111 0011	163	115	0x73	s	Lowercase letter s
0111 0100	164	116	0x74	t	Lowercase letter t
0111 0101	165	117	0x75	u	Lowercase letter u
0111 0110	166	118	0x76	v	Lowercase letter v
0111 0111	167	119	0x77	w	Lowercase letter w
0111 1000	170	120	0x78	x	Lowercase letter x
0111 1001	171	121	0x79	y	Lowercase letter y

Bin (Binary)	Oct (Octal)	(Dec) (Decimal)	Hex (Hexadecimal)	Abbreviations/characters	Explanation
0111 1010	172	122	0x7A	z	Lowercase letter z
0111 1011	173	123	0x7B	{	Flower bracket
0111 1100	174	124	0x7C	|	Vertical line
0111 1101	175	125	0x7D	}	Closing flower bracket
0111 1110	176	126	0x7E	~	Tilde
0111 1111	177	127	0x7F	DEL (delete)	Delete