XINJE

XS series PLCopen controller

User manual [Instruction] (XS Studio)

Wuxi XINJE Electric Co., Ltd.

Data No. PS05 20240201EN 1.2

Basic Description

- Thank you for purchasing the Xinje XS series PLCopen standard controller.
- This manual mainly introduces the relevant usage of XS series PLCopen standard controller instructions.
- Before using the product, please read this manual carefully and program it on the premise of fully understanding the content of the manual.
- Please deliver this manual to the end user.

User Information

- Only operators with a certain level of electrical knowledge can perform wiring and other operations on the product. If there are any areas where the use is unknown, please
- Consult our company's technical department.
- The examples listed in manuals and other technical materials are for user understanding and reference only, and do not guarantee certain actions.
- When using this product in combination with other products, please confirm whether it meets relevant specifications, principles, etc.
- When using this product, please confirm whether it meets the requirements and is safe.
- Please set up backup and safety functions on your own to avoid potential machine malfunctions or losses caused by the malfunction of this product.

Declaration of Responsibility

- Although the content in the manual has been carefully checked, errors are inevitable and we cannot guarantee complete consistency.
- We will regularly check the content in the manual and make corrections in subsequent versions. We welcome your valuable feedback.
- The content described in the manual is subject to change without prior notice.

Related manuals

Please refer to the following manual for hardware related and software applications of the XS series PLC.

- User Manual for XS Series PLCopen Standard Controller [Hardware]
- User Manual for XS Series PLCopen Standard Controller [Software]

WUXI XINJE ELECTRIC CO., LTD. Copy Rights Reserved

Without explicit written permission, this material and its contents shall not be copied, reproduced or used, and violators shall be liable for any losses caused. All rights provided in the patent license and registration, including utility modules or designs, are reserved.

Jan. 2023

Catalog

1. BASIC INSTRUCTIONS 1

1-1. BIT LOGIC INSTRUCTIONS 1

1-1-1. Basic bit logic instructions 1

1-1-1-1. Bit AND 1

1-1-1-2. Boolean AND 3

1-1-1-3. Bit OR 4

1-1-1-4. Boolean OR 5

1-1-1-5. Bit NOT 6

1-1-1-6. Boolean NOT 7

1-1-1-7. Bit XOR 8

1-1-1-8. Boolean XOR 9

1-1-2. Bit logic operation instruction 10

1-1-2-1. Rising edge output PLS 10

1-1-2-2. Falling edge output PLF 11

1-1-2-3. Alternating output ALT 12

1-1-2-4. Bit data output BOUT 14

1-1-2-5. Set on bit BSET 15

1-1-2-6. Reset bit BRST 16

1-1-2-7. Logical AND/Logical OR/XOR/XNOR AryAnd/AryOr/AryXor/AryXorN 18

1-1-3. Set priority and reset priority trigger instructions 21

1-1-3-1. Set priority trigger SR 21

1-1-3-2. Reset priority trigger RS 22

1-1-4. Edge detection instruction 24

1-1-4-1. Rising edge detection R_TRIG 24

1-1-4-2. Falling edge detection F_TRIG 25

1-2. TIMER 27

1-2-1. Pulse timer TP 27

1-2-2. Power on delay timer TON 28

1-2-3. Power off delay timer TOF 30

1-2-4. RTC 31

1-2-5. Accumulation timer 33

1-2-6. 100ms Timer 35

1-3. COUNTER 37

1-3-1. Up counter CTU 37

1-3-2. Down counter CTD 39

1-3-3. Up/down counter CTUD 40

1-3-4. Subtract counter CTD_XXX 42

1-3-5. Up/Down Counter CTUD_XXX 45

1-3-6. Up counter CTU_XXX 49

1-4. DATA PROCESSING INSTRUCTIONS 52

1-4-1. Selection commands 52

1-4-1-1. Binary choice instruction SEL 52

1-4-1-2. Take the maximum value MAX 53

1-4-1-3. Take the minimum value MIN 54

1-4-1-4. Limit value LIMIT 55
1-4-1-5. Multiple choice MUX 57
1-4-2. Compare Instructions *58*
1-4-2-1. Basic compare instruction EQ/NE/GT/GE/LT/LE 58
1-4-2-2. Maximum/minimum value retrieval of arrays AryMax/AryMin 60
1-4-2-3. Array retrieval ArySearch 63
1-4-2-4. Batch comparison of arrays EQ AryCmpEQ 66
1-4-2-5. Comparison of array elements EQV AryCmpEQV 68
1-4-2-6. Batch comparison of arrays GE AryCmpGE 70
1-4-2-7. Comparison of array elements GEV AryCmpGEV 72
1-4-2-8. Batch comparison of arrays GT AryCmpGT 74
1-4-2-9. Comparison of array elements GTV AryCmpGTV 76
1-4-2-10. Batch comparison of arrays LE AryCmpLE 78
1-4-2-11. Comparison of array elements LEV AryCmpLEV 80
1-4-2-12. Batch comparison of arrays LT AryCmpLT 82
1-4-2-13. Comparison of array elements LTV AryCmpLTV 84
1-4-2-14. Batch comparison of arrays NE AryCmpNE 86
1-4-2-15. Comparison of array elements NEV AryCmpNEV 88
1-4-2-16. Table comparison TableCmp 90
1-4-2-17. Zone comparison ZoneCmp 93
1-4-3. Shift instruction *95*
1-4-3-1. Shift left SHL 95
1-4-3-2. Shift right SHR 96
1-4-3-3. Rotate left ROL 98
1-4-3-4. Rotate right ROR 99
1-4-3-5. Shift register AryShiftReg 101
1-4-3-6. Left-right shift register AryShiftRegLR 104
1-4-3-7. Loop left/right shift with carry RCL/RCR 107
1-4-3-8. Shift N elements of an array left/right ArySHL/ArySHR 111
1-4-3-9. Shift N elements of an array left/right NSHLC/NSHRC 114
1-4-3-10. Copy bit data to the left/right SFTL/SFTR 116
1-4-3-11. Copy word data to the left/right WSFL/WSFR 125
1-4-3-12. First in, first out data read/write SFRD/SFWR 128
1-4-4. Move instructions *133*
1-4-4-1. Bulk transfer BMOV 133
1-4-4-2. Obtain data specified bit status BON 135
1-4-4-3. Data combination of byte unit BTOW 136
1-4-4-4. One to many data transfer FMOV 138
1-4-4-5. ON bit SUM 140
1-4-4-6. Array transfer AryMove 141
1-4-4-7. Digit transfer MoveDigit 143
1-4-4-8. Multiple bits transfer TransBits 145
1-4-4-9. Set block SetBlock 147
1-4-4-10. Bit decoder Decoder 149
1-4-4-11. Bit encoder Encoder 151
1-4-4-12. Bit count BitCnt 153

1-4-4-13. High low byte exchange SWAP 155
1-4-4-14. Data exchange XCH 156
1-4-4-15. Array data exchange AryExchange 158
1-4-4-16. Data exchange Exchange 161
1-4-5. Data separation and combination instructions 163
1-4-5-1. Byte unit data separation Dispart8Bit 163
1-4-5-2. 4-bit separate DispartDigit 165
1-4-5-3. Mantissa of real numbers, exponential separation DispartReal 167
1-4-5-4. Byte unit data combination group Unite8Bit_XXX 168
1-4-5-5. 4-digit combination group UniteDigit_XXX 170
1-4-5-6. Combine mantissa and exponents into real numbers UniteReal 172
1-4-5-7. Byte unit data separation WTOB 173
1-4-6. Data read and write instructions 175
1-4-6-1. N bits write to group WriteNbit_XXX 175
1-4-6-2. N bits read group ReadNbit_XXX 177
1-4-7. Data conversion instructions 179
1-4-7-1. BCD code and integer data conversion 179
1-4-7-2. BOOL_TO_<TYPE> boolean type conversion data 182
1-4-7-3. BYTE_TO_<TYPE> byte type conversion data 184
1-4-7-4. < Integer data > _TO_<TYPE> integer type conversion instruction 185
1-4-7-5. REAL_TO_<TYPE> Real type conversion instruction 187
1-4-7-6. TIME_TO_<TYPE> time type conversion instruction 189
1-4-7-7. DATE_TO_<TYPE> Date type conversion instruction 190
1-4-7-8. DT_TO_<TYPE> Date time type conversion instruction 192
1-4-7-9. TOD_TO_<TYPE> Time type conversion instruction 193
1-4-7-10. STRING_TO_<TYPE> Character type conversion instruction 194
1-4-7-11. Rounding TRUNC 195
1-4-7-12. Array conver to string AryToString 196
1-4-7-13. Date convert to string DateToString 198
1-4-7-14. Date time conver to string DtToString 199
1-4-7-15. Convert hexadecimal strings to integers HexStringToNum_XXX 200
1-4-7-16. Convert integers to fixed length decimal/hexadecimal strings NumToDecString/NumToHexString 202
1-4-7-17. String conver to array StringToAry 205
1-4-7-18. Time convert to string TodToString 207
1-4-7-19. Convert from byte array AryByteTo 208
1-4-7-20. Batch conversion of arrays to BCD AryToBCD 213
1-4-7-21. Batch convert arrays to bit strings AryToBin 215
1-4-7-22. Bit string conver to Gray code BinToGray_XXX 217
1-4-7-23. Bit train/bit line conver to array ColmToLine_XXX 219
1-4-7-24. Copy bit mode CopyByteToNum/CopyWordToNum/CopyDwordToNum/ CopyLwordToNum 221
1-4-7-25. Copy bit mode CopyDwordToReal/CopyLwordToLreal 223
1-4-7-26. Copy bit mode CopyRealToDWord/CopyLrealToLword 224
1-4-7-27. Copy bit mode CopyNumToXXX 225
1-4-7-28. Copy bit mode CopyNumToReal/CopyNumToLreal 227
1-4-7-29. Copy bit mode CopyRealToNum/CopyLrealToNum 228
1-4-7-30. Gray code convert to bit string GrayToBin_XXX 229

1-4-7-31. Gray code convert to angle GRAY 231
1-4-7-32. Convert fixed decimal point number to decimal strings FixNumToString....235
1-4-7-33. String convert to fixed decimal point number StringToFixNum....236
1-4-7-34. Row/column conversion LineToColm 239
1-4-7-35. Convert to byte array ToAryByte....241
1-4-8. Initialization instruction 244
1-4-8-1. Initialization Clear 244
1-4-9. Regional control instructions....246
1-4-9-1. Dead zone control command BZAND_TAB 246
1-4-9-2. Calculation of average data value MEAN_TAB....248
1-4-9-3. Slope command RAMP_TAB....250
1-4-9-4. Table coordinate acquisition instruction SCL_TAB....253
1-4-9-5. Table data sorting instruction SORT_TAB....255
1-4-9-6. Data calculation total instruction WSUM_TAB....257
1-4-9-7. Zone control instruction ZONE_TAB....259
1-4-9-8. Batch data reset instruction ZRST_TAB....261
1-5. OPERATION INSTRUCTIONS....263
1-5-1. Assignment instruction....263
1-5-1-1. Assignment instruction MOVE....263
1-5-2. Arithmetic operation instructions 264
1-5-2-1. Addition operation ADD....264
1-5-2-2. Subtraction operation SUB 266
1-5-2-3. Multiplication operation MUL....267
1-5-2-4. Division operation DIV....268
1-5-2-5. Remainder operation MOD....270
1-5-3. Mathematical operation instructions....271
1-5-3-1. Absolute value ABS....271
1-5-3-2. Square root SQRT....272
1-5-3-3. Exponent EXP 273
1-5-3-4. Natural logarithm LN....274
1-5-3-5. Logarithm with a base of 10 LOG 275
1-5-3-6. Sine function SIN....276
1-5-3-7. Cosine function COS 277
1-5-3-8. Arccosine function ACOS....278
1-5-3-9. Arcsin function ASIN....279
1-5-3-10. Tangent function TAN....280
1-5-3-11. Arctangent function ATAN....281
1-5-3-12. Floating point angle/radian conversion RAD/DEG 282
1-5-3-13. Array element addition AryAddV 284
1-5-3-14. Calculation of array elements average value AryMean....286
1-5-3-15. Standard deviation of array elements ArySD....288
1-5-3-16. Subtraction of array elements ArySubV....290
1-5-3-17. Real number check CheckReal 292
1-5-3-18. Increase/decrease Inc/Dec....294
1-5-3-19. Real remainder ModReal 295
1-5-3-20. Long real remainder ModReal_LR 297

1-5-3-21. Moving average MovingAverage....299
1-5-3-22. Real carry RoundUp....301
1-5-3-23. Generate random number RAND....302
1-5-4. Address operation instruction....304
1-5-4-1. Data type size SIZEOF....304
1-5-4-2. Address operator ADR....305
1-5-4-3. Bit address operator BITADR....307
1-6. STRING COMMAND....309
1-6-1. Sum calculation StringSum....309
1-6-2. String clear Clearstring....311
1-6-3. Get the number of bytes in the string GetByteLen....312
1-6-4. Real string read delimiter delete SubDelimiter....314
1-6-5. Long real string read delimiter delete SubDelimiter_LR....318
1-6-6. String letter conversion ToUCase/ToLCase....322
1-6-7. String left/right adjustment TrimL/TrimR....324
2. SPECIAL INSTRUCTIONS....326
2-1. HIGH SPEED COUNT INSTRUCTIONS....326
2-1-1. Function overview....326
2-1-2. Function block....326
2-1-2-1. Enable the high speed counter【XJ_Counter_Enable】....326
2-1-2-2. Compare consistent output【XJ_Counter_Compare】....329
2-1-2-3. Preset value write in【XJ_Counter_PresetValue】....331
2-1-2-4. Probe【XJ_TouchProbe】....333
2-1-2-5. Read the pulse width measurement value of the counter【XJ_MeasurePulseWidth】....335
2-1-2-6. Counter sample【XJ_Counter_Sample】....336
2-1-2-7. Multiple segments compare【XJ_Counter_CompareArray】....337
2-1-2-8. Ring counting【XJ_Counter_SetRing】....339
2-1-2-9. Reset port of compare consistent output【XJ_ResetCmpOutput】....340
2-1-2-10. Clear the error【XJ_Counter_Reset】....341
2-1-3. Parameter settings....342
2-2. XS SERIES HIGH SPEED COUNT INSTRUCTIONS....343
2-2-1. Function overview....343
2-2-2. Function block....343
2-2-2-1. High speed counter【XJ_Counter】....343
2-2-2-2. Read high speed counter【XJ_CounterGetValue】....344
2-2-2-3. Write high speed counter【XJ_CounterSetValue】....345
2-2-3. Parameter configuration....348
2-2-4. Application example....348
2-3. EXTERNAL INTERRUPT AND COMPARE CONSISTENT INTERRUPT INSTRUCTIONS....349
2-3-1. Function overview....349
2-3-2. Function block....349
2-3-2-1. Open external interrupts and compare consistent interrupts【XJ_EnableInterrupt】....349
2-3-2-2. Write in interrupt parameters【XJ_WriteInterruptParameter】....353
2-3-3. Parameter configuration....354
2-3-4. Application example....356
2-4. PID INSTRUCTIONS....358

2-4-1. Command format *358*
2-4-2. Related variables *358*
2-4-3. Function description *360*
2-4-4. Application example *361*
2-5. SYSTEM LIBRARY 368
2-5-1. Function overview *368*
2-5-2. Function block introduction *368*
2-5-2-1. CPU dominant frequency【XJ_GetCPUFrequency】 368
2-5-2-2. CPU temperature【XJ_GetCPUTemperature】 368
2-5-2-3. CPU occupancy【XJ_GetCPUUsage】 369
2-5-2-4. Memory size【XJ_GetMemSize】 369
2-5-2-5. Memory occupancy【XJ_GetMemUsage】 370
2-5-2-6. On time【XJ_GetBootTime】 370
2-5-2-7. PLC name【XJ_GetPLCName】 371
2-5-2-8. PLCID【XJ_GetPLCID】 372
2-5-2-9. Firmware version【XJ_GetPLCVersion】 373
2-5-2-10. Obtain network port information【XJ_GetNetInfo】 374
2-5-2-11. Set network port information【XJ_SetNetInfo】 376
2-5-2-12. Runtime version【XJ_GetRuntimeVersion】 377
2-5-2-13. BOOT version【XJ_GetBootVersion】 378
2-5-2-14. Obtain date and time【XJ_GetTime】 378
2-5-2-15. Set date and time【XJ_SetTime】 379
2-5-3. Parameter configuration *381*
2-6. ECAT_FROMTO 383
2-6-1. Function overview *383*
2-6-2. Function block introduction *383*
2-6-2-1. Remote IO read【XJ_ECATFromTo.XJ_EC_FROM】 383
2-6-2-2. Remote IO write【XJ_ECATFromTo.XJ_EC_TO】 384
2-6-3. Parameter configuration *385*
2-6-4. Application *387*
2-7. DATE TIME 388
2-7-1. Obtain the current system clock GetSystemDate *388*
2-7-2. Obtain system runtime GetSystemTime *390*
2-7-3. Set the current system clock SetSystemDate *391*
2-7-4. Add date and time ADD_DT_TIME *393*
2-7-5. Time addition ADD_TOD_TIME *394*
2-7-6. Leap year judgment ChkLeapYear *396*
2-7-7. Date and time combination DateStructToDt *397*
2-7-8. Date second conversion DateToSec *399*
2-7-9. Days and months conversion DaysToMonth *400*
2-7-10. Time division DIVTIME *401*
2-7-11. Date time decomposition DtToDateStruct *403*
2-7-12. Date, time, second conversion DtToSec *405*
2-7-13. Get week information GetDayOfWeek *406*
2-7-14. Obtain the number of days in the target month GetDaysOfMonth *408*
2-7-15. Obtain the current system time structure GetSystemDate_sDt *409*

2-7-16. Get week information GetWeekOfYear *410*
2-7-17. Multiplying time MULTIME *412*
2-7-18. Nanosecond time conversion NanoSecToTime *413*
2-7-19. Second date conversion SecToDate *414*
2-7-20. Second date time conversion SecToDt *416*
2-7-21. Second time conversion SecToTime *417*
2-7-22. Second time conversion SecToTod *418*
2-7-23. Date subtraction SUB_DATE_DATE *419*
2-7-24. Subtracting date and time SUB_DT_DT *420*
2-7-25. Subtracting date, time, and time SUB_DT_TIME *422*
2-7-26. Subtracting time from time SUB_TOD_TIME *423*
2-7-27. Time subtraction SUB_TOD_TOD *424*
2-7-28. Time nanosecond conversion TimeToNanoSec *426*
2-7-29. Time second conversion TimeToSec *427*
2-7-30. Time second conversion TodToSec *428*
2-7-31. Date and time rounding off TruncDt *429*
2-7-32. Time rounding off TruncTime *431*
2-7-33. Rounding off time TruncTod *432*
2-8. COMMUNICATION INSTRUCTIONS 434
2-8-1. TCP communication *434*
2-8-1-1. TCP communication data receive TCP_Receive 434
2-8-1-2. TCP communication data send TCP_Send 436
2-8-1-3. Create TCP connection TCP_Connect 437
2-8-1-4. Create TCP server-side communication service TCP_Server 438
2-8-1-5. Create TCP client communication service TCP_Client 439
2-8-1-6. Example of TCP communication instruction program 440
2-8-2. UDP communication *449*
2-8-2-1. Create UDP communication connecting UDP_Peer 449
2-8-2-2. UDP communication data reception UDP_Receive 450
2-8-2-3. UDP communication data send UDP_Send 451
2-8-2-4. Example of UDP communication 452
2-8-3. TCP/UDP communication error code *456*
2-8-4. CAN free format communication *457*
2-8-4-1. CAN free format communication specification 457
2-8-4-2. CAN free format send FreeCan_Send_CB 457
2-8-4-3. CAN free format receive FreeCan_Recv_CB 459
2-8-3-4. CAN free format communication example 461
2-9. FILE PROCESSING 463
2-9-1. Create directory DirCreate *463*
2-9-2. Delete directory DirRemove *465*
2-9-3. Open file FileOpen *467*
2-9-4. Close the file FileClose *470*
2-9-5. Find the file FileSeek *472*
2-9-6. Write in the file FileWrite *475*
2-9-7. Read the file FileRead *478*
2-9-8. Write string FilePuts *481*

2-9-9. Read string FileGets *483*
2-9-10. Write variable file FileWriteVar *486*
2-9-11. Read variable file FileReadVar *489*
2-9-12. Copy file FileCopy *492*
2-9-13. Delete file FileRemove *495*
2-9-14. Rename file FileRename *497*

3. MOTION INSTRUCTIONS 500

3-1. SINGLE AXIS 500
3-1-1. Single axis instruction overview *500*
3-1-2. Single axis instructions *501*
3-1-2-1. Axis enable [MC_Power] 501
3-1-2-2. Axis reset [MC_Reset] 503
3-1-2-3. Stop controller motion [MC_Stop] 505
3-1-2-4. Pause the motion [MC_Halt] 506
3-1-2-5. Axis homing [MC_Home] 507
3-1-2-6. Jog run [MC_Jog] 511
3-1-2-7. Absolute position [MC_MoveAbsolute] 513
3-1-2-8. Position overlay [MC_MoveAdditive] 515
3-1-2-9. Relative position [MC_MoveRelative] 519
3-1-2-10. Superimposed relative motion command [MC_MoveSuperImposed] 521
3-1-2-11. Speed control [MC_MoveVelocity] 525
3-1-2-12. Position profile [MC_PositionProfile] 527
3-1-2-13. Speed profile [MC_VelocityProfile] 531
3-1-2-14. Acceleration profile [MC_AccelerationProfile] 535
3-1-2-15. Read actual position [MC_ReadActualPosition] 537
3-1-2-16. Read current torque [MC_ReadActualTorque] 538
3-1-2-17. Read the current speed [MC_ReadActualVelocity] 539
3-1-2-18. Read axis error status [MC_ReadAxisError] 540
3-1-2-19. Read the axis bit parameter [MC_ReadBoolParameter] 541
3-1-2-20. Read the axis parameter [MC_ReadParameter] 542
3-1-2-21. Read axis instruction position [SMC_ReadSetPosition] 543
3-1-2-22. Read function block error [SMC_ReadFBError] 544
3-1-2-23. Set axis bit parameter [MC_WriteBoolParameter] 545
3-1-2-24. Set the axis parameter [MC_WriteParameter] 547
3-1-2-25. Clear the error [SMC_ClearFBError] 548
3-1-2-26. Read the error [SMC_ErrorString] 549
3-1-2-27. Recall driver/axis [SMC3_ReinitDrive] 550
3-1-2-28. Set the axis COE parameter [SMC3_ETC_WriteParameter_CoE] 551
3-1-2-29. Probe [MC_TouchProbe] 553
3-1-3. Single axis function application *557*
3-2. AXIS GROUP FUNCTION 558
3-2-1. Axis group instruction *558*
3-2-2. Axis group instructions *559*
3-2-2-1. Add axis to axis group [MC_AddAxisToGroup] 559
3-2-2-2. Remove axis from the axis group [MC_RemoveAxisFromGroup] 560
3-2-2-3. Ungroup all the axes [MC_UngroupAllAxes] 561

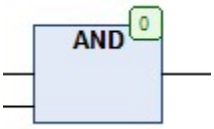
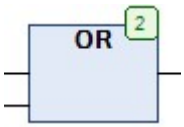
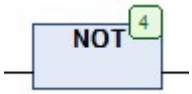
3-2-2-4. Enable the axis group [MC_GroupEnable] 562
3-2-2-5. Disable the axis group [MC_GroupDisable] 563
3-2-2-6. Axis group reset [MC_GroupReset] 564
3-2-2-7. Set axis group position [MC_GroupSetPosition] 566
3-2-2-8. Coordinate transform [MC_SetCoordinateTransform] 567
3-2-2-9. Dynamic coordinate system conversion [MC_SetDynCoordTransform] 568
3-2-2-10. Axis group continue running [MC_GroupContinue] 570
3-2-2-11. Axis group halt [MC_GroupHalt] 571
3-2-2-12. Axis group interruption [MC_GroupInterrupt] 572
3-2-2-13. Axis group stop [MC_GroupStop] 573
3-2-2-14. Kinematic coordinate transformation [MC_SetKinTransform] 575
3-2-2-15. Set axis group overshoot value [MC_GroupSetOverride] 576
3-2-2-16. Absolute arc interpolation [MC_MoveCircularAbsolute] 580
3-2-2-17. Relative arc interpolation [MC_MoveCircularRelative] 585
3-2-2-18. Absolute position quick positioning [MC_MoveDirectAbsolute] 590
3-2-2-19. Relative position quick positioning [MC_MoveDirectRelative] 592
3-2-2-20. Absolute position linear interpolation [MC_MoveLinearAbsolute] 594
3-2-2-21. Relative position linear interpolation [MC_MoveLinearRelative] 599
3-2-2-22. Read the feedback position of the axis group [MC_GroupReadActualPosition] 603
3-2-2-23. Read the feedback speed of the axis group [MC_GroupReadActualVelocity] 604
3-2-2-24. Read axis group configuration parameters [MC_GroupReadConfiguration] 605
3-2-2-25. Read axis group error [MC_GroupReadError] 607
3-2-2-26. Read the current operating status of the axis group [MC_GroupReadStatus] 608
3-2-2-27. Startup the axis group [SMC_StartupAxisGroup] 610
3-2-2-28. Enable the axis group [SMC_GroupPower] 611
3-2-2-29. Break Assignment [SMC_GroupInterruptAt] 612
3-2-2-30. Reboot after error reset [SMC_GroupEnableResumeAfterError] 613
3-2-2-31. Axis group jog run [SMC_GroupJog] 614
3-2-2-32. Axis group wait [SMC_GroupWait] 616
3-2-3. Axis group function application *618*
3-3. CAM FUNCTION 620
3-3-1. CAM instruction list *620*
3-3-2. CAM instructions *621*
3-3-2-1. Cam table designation [MC_CamTableSelect] 621
3-3-2-2. CAM binding [MC_CamIn] 622
3-3-2-3. Cam unbinding [MC_CamOut] 625
3-3-2-4. CAM action start [MC_GearIn] 626
3-3-2-5. Position specified gear action [MC_GearInPos] 628
3-3-2-6. Gear action release [MC_GearOut] 632
3-3-2-7. Master slave axis phase offset [MC_Phasing] 633
3-3-2-8. CAM range [SMC_CAMBounds] 634
3-3-2-9. CAM position range [SMC_CAMBounds_Pos] 639
3-3-2-11. CAM tappet control [SMC_CamRegister] 642
3-3-2-12. Get cam slave axis position [SMC_GetCamSlaveSetPosition] 645
3-3-2-13. Obtain tappet output value [SMC_GetTappetValue] 647
3-3-2-14. Read the cam table [SMC_ReadCAM] 648

3-3-2-15. Modify the cam table [SMC_WriteCAM] 649
3-3-3. CAM function application *650*
APPENDIX **653**

1. Basic instructions

1-1. Bit logic instructions

1-1-1. Basic bit logic instructions

Instruction	Instruction icon	Function
AND		Operator AND
OR		Operator OR
NOT		Operator NOT
XOR		Operator XOR

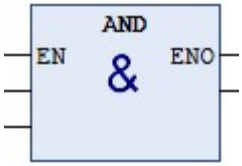
Basic bit logic instructions include AND, OR, NOT, and XOR. In XS Studio, functions can be divided into bit logic operation and Boolean logic operation.

- Bit logic operation: perform Boolean logic operation on corresponding bits of two integer data one by one, and return compatible integer results.
- Boolean logic operation: perform logical operation on two Boolean type data.

1-1-1-1. Bit AND

Function: The bit AND operation instruction compares the corresponding bits of two integers. When the corresponding bits of two numbers are both 1, the corresponding result bit returned is "1". When the corresponding bits of two integers are both "0" or one of the bits is "0", the corresponding result bit is returned as "0".

Bit AND [AND]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

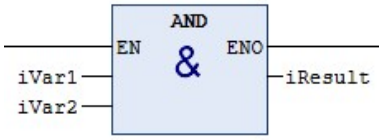
Instruction	Name	Graph	ST
AND	Bit AND		AND

AND logic relationship table

Input 1	Input 2	Result
0	0	0
0	1	0
1	0	0
1	1	1

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    iVar1  :UINT:=1;
    iVar2  :UINT:=85;
    iResult:UINT;
END_VAR
```

</td></tr>
<tr><td>Sample Program</td><td> 
</td><td>

```
1  iResult:=iVar1 and iVar2;
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

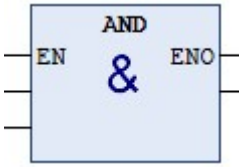
表达式	类型	值
iVar1	UINT	1
iVar2	UINT	85
iResult	UINT	1

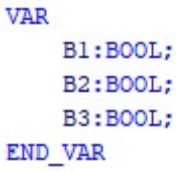
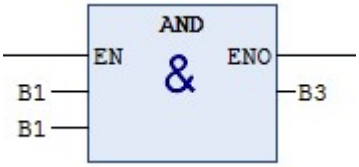
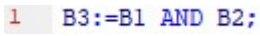
</td></tr>
<tr><td>Result analysis</td><td colspan="2">The decimal number 1 corresponds to a binary number of 0000 0001, and the decimal number 85 corresponds to a binary number of 0101 0101. According to the definition of bit AND operation, each independent bit is subjected to an AND operation one by one, resulting in the final result of 0000 0001, which is a decimal value of 1.</td></tr>
</table>

1-1-1-2. Boolean AND

Function: Boolean AND operation is used to calculate the AND results of two Boolean expression. When the result of two Boolean expression is true, it is returned as true. If one of them is false, it is returned as false.
Example: Create a POU and use the Boolean AND operation to determine the return value of the operation. The code is as follows:

Boolean AND [AND]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

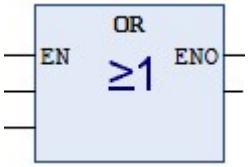
Instruction	Name	Graph	ST
AND	Boolean AND		AND

DEMO	LD	ST
Define variables		
Sample Program		
Result analysis	When both B1 and B2 are true, the running result of the program is B3 to be true.	

1-1-1-3. Bit OR

Function: The bit OR operation instruction compares the corresponding bits of two integers. When one of the corresponding bits of two numbers is "1" or both are "1", the corresponding result bit is returned as "1". When the corresponding bits of two integers are both "0", the corresponding result is returned as "0".

Bit OR [OR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

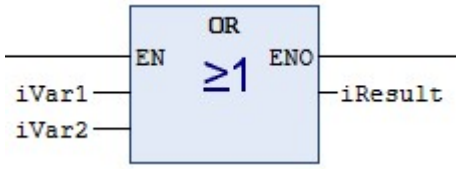
Instruction	Name	Graph	ST
OR	Bit OR		OR

OR logic relationship table

Input 1	Input 2	Result
0	0	0
0	1	1
1	0	1
1	1	1

DEMO	LD	ST

Define variables		VAR
		iVar1 :UINT:=1;
		iVar2 :UINT:=85;
		iResult:UINT;
		END_VAR

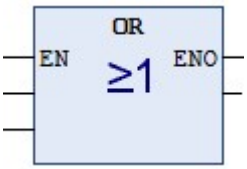
Sample Program		1 iResult:=iVar1 OR iVar2;
Operation result		
Result analysis	The final running result of the program is 85.	

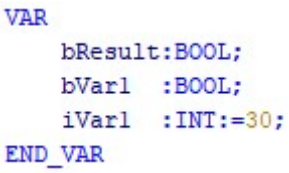
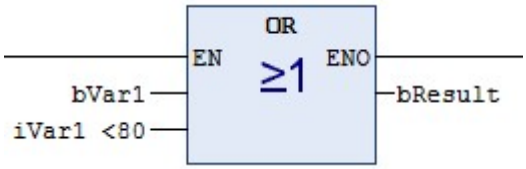
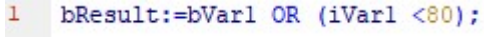
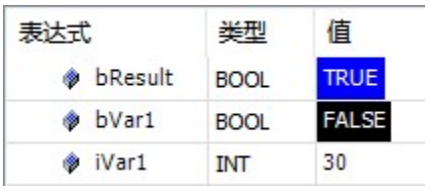
1-1-1-4. Boolean OR

Function: Boolean OR operation instruction is used to calculate the OR result of two Boolean expression. When one of the two Boolean expression returns true, the result is true. When the result of two Boolean expression is false, the result is false.

Example: Create a POU and use the Boolean OR operation to determine the return value of the operation. The code is as follows:

Boolean OR [OR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

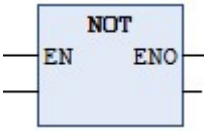
Instruction	Name	Graph	ST
OR	Boolean OR		OR

DEMO	LD	ST
Define variables		
Sample Program		
Operation result		
Result analysis	Due to the initial value of iVar1 being 30, the condition of iVar1<80 is true, while the initial value of bVar1 is "0", so it is false. The logical result of one true and one false OR is true. Therefore, the condition on the right side of the equation is true, and the running result of the program is bResult being true.	

1-1-1-5. Bit NOT

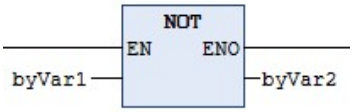
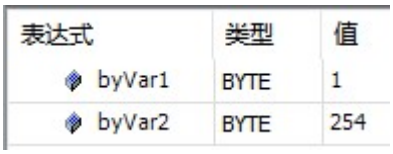
Function: Negate the logical string, changing the current value from "0" to "1" or from "1" to "0". The bit 'not' operation instruction is to take variables or constants one by one as not.

Bit NOT [NOT]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
NOT	Bit NOT		NOT

NOT logic relationship table

Input	Result
0	1
1	0

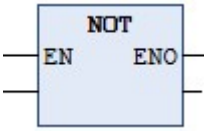
DEMO	LD	ST
Define variables		VAR byVar1:BYTE:=1; byVar2:BYTE; END_VAR
Sample Program		1 byVar2:=NOT byVar1;
Operation result		
Result analysis	Due to the value of byVar1 being 1, converting it to binary yields 0000 0001. After performing bit negation, the result is 1111 1110. The final output result is 254.	

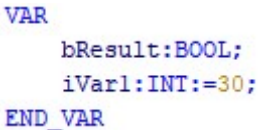
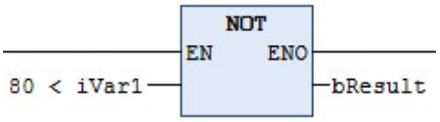
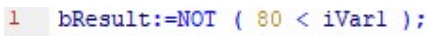
1-1-1-6. Boolean NOT

Function: Boolean "Not" operation instruction is used to calculate the result of a single Boolean expression. When the input is true, the result is false. When the input is false, the result is true.

Example: Create a POU and use the Boolean 'not' operation to determine the return value of the operation. The specific code is as follows:

Boolean NOT [NOT]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

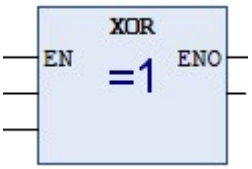
Instruction	Name	Graph	ST
NOT	Boolean NOT		NOT

DEMO	LD	ST
Define variables		
Sample Program		
Operation result		
Result analysis	80<30 this proposition is false. After using the NOT instruction to negate the Boolean expression, the result is true, so the final bResult is true.	

1-1-1-7. Bit XOR

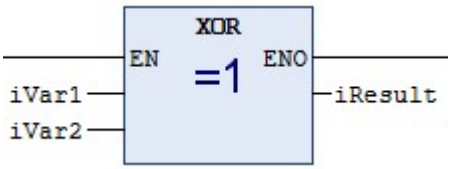
Function: Compare the corresponding bits of two integers using the bit XOR operation instruction. When the corresponding bit of two integers is one "1" and the other is "0", the corresponding result bit returned is "1". When the corresponding bits of two integers are both "1" or "0", the corresponding result bit is returned as "0".
Example: Create a POU, perform bit XOR on variables iVar1 and iVar2, and output the results. The specific implementation code is as follows:

Bit XOR [XOR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
XOR	Bit XOR		XOR

DEMO	LD	ST

Define variables	VAR
	iVar1 :INT:=1;
	iVar2 :INT:=85;
	iResult:INT;
	END_VAR	

Sample Program		1 iResult:=iVar1 XOR iVar2;
Operation result		
Result analysis	The decimal number 1 corresponds to a binary number of 0000 0001, and the decimal number 85 corresponds to a binary number of 0101 0101. According to the definition of the bit XOR operation instruction, the result is 84. Bit XOR operation. Only when the input state of one contact is "1" and the input state of the other contact is "0", the output is "1". If both contact states are "1" or "0" at the same time, the output is "0".	

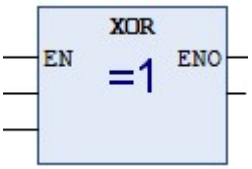
表达式	类型	值
iVar1	INT	1
iVar2	INT	85
iResult	INT	84

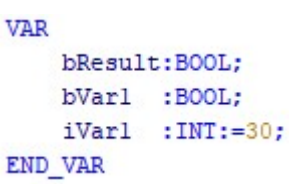
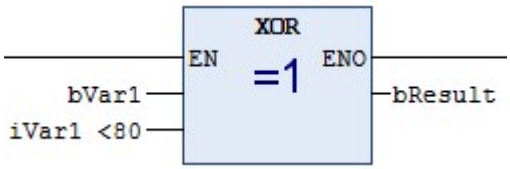
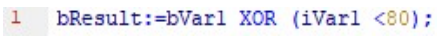
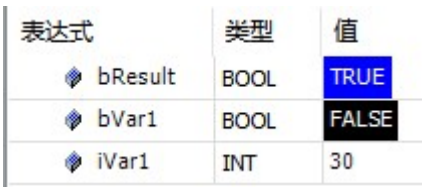
1-1-1-8. Boolean XOR

Function: Boolean XOR operation instruction is used to calculate the results of two Boolean expression. Only when one expression is true and the other is false, the result returned by the expression is true. When both expressions evaluate to true or false, the returned result is false.

Example: Create a POU and use the Boolean XOR operation instruction to determine whether the return value is TRUE or FALSE. The specific implementation code is as follows:

Boolean XOR [XOR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
XOR	Boolean XOR		XOR

DEMO	LD	ST
Define variables		
Sample Program		
Operation result		
Result analysis	The running result of the program is TRUE.	

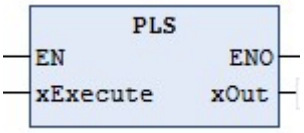
1-1-2. Bit logic operation instruction

1-1-2-1. Rising edge output PLS

(1) Instruction overview

When the PLS instruction xExecute detects a rising edge, xOut will be set to a true state, which will be set to a false state after one scan cycle.

Rising edge output [PLS]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
PLS	Rising edge output		PLS(xExecute:= , xOut=>);

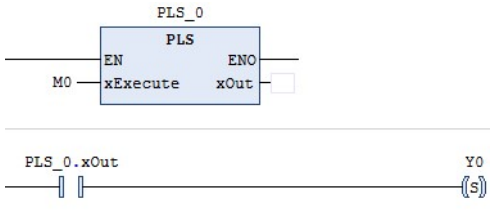
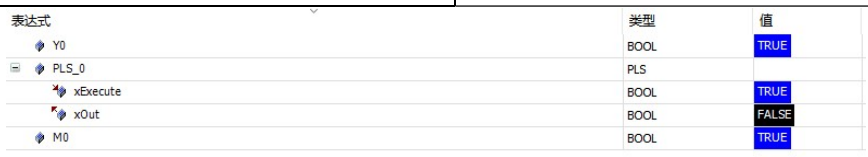
(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
xExecute	Input signal	BOOL	FALSE,TRUE	FALSE	The BOOL value to be detected
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Output signal	BOOL	FALSE,TRUE	FALSE	Output result

(3) Function

- Output a scan cycle command when the rising edge is reached.

(4) Program example

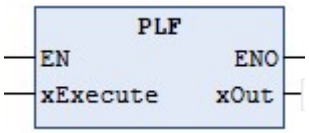
DEMO	LD	ST
Define the variable		VAR PLS_0: PLS; M0: BOOL; Y0: BOOL; END_VAR
Program example		PLS_0(xExecute:=M0, xOut=>); IF PLS_0.xOut THEN Y0:=TRUE; END_IF
Operation result		

1-1-2-2. Falling edge output PLF

(1) Instruction overview

When the PLF instruction xExecute detects a falling edge, xOut will be set to a true state, which will be set to a false state after one scan cycle.

Falling edge output [PLF]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
PLF	Falling edge output		PLF_0(xExecute:=, xOut=>);

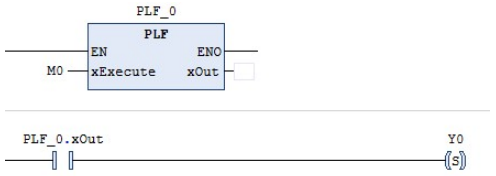
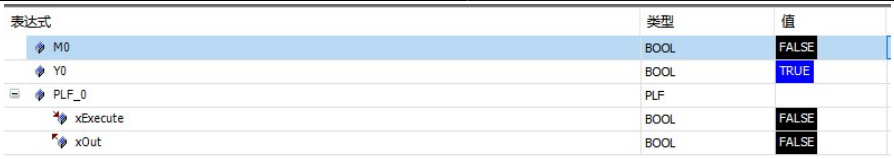
(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
xExecute	Input signal	BOOL	FALSE,TRUE	FALSE	The BOOL value to be detected
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
xOut	Output signal	BOOL	FALSE,TRUE	FALSE	Output result

(3) Function

- Output a scanning cycle command at the falling edge

(4) Program example

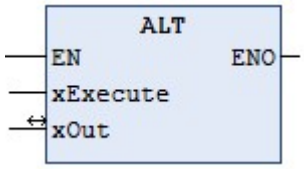
DEMO	LD	ST
Define the variable		VAR PLF_0: PLF; M0: BOOL; Y0: BOOL; END_VAR
Program example		PLF_0(xExecute:= M0, xOut=>); IF PLF_0.xOut THEN Y0:=TRUE; END_IF
Operation result		

1-1-2-3. Alternating output ALT

(1) Instruction overview

When the ALT instruction xExecute detects the rising edge, xOut will reverse its state.

Alternating output [ALT]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ALT	Alternating output		ALT_0(xExecute:= , xOut:=);

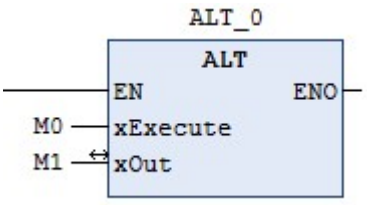
(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
xExecute	Input signal	BOOL	FALSE,TRUE	FALSE	The BOOL value to be detected
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
xOut	Output signal	BOOL	FALSE,TRUE	FALSE	If the rising edge of xExecute is detected and its state is reversed

(3) Function

- After detecting the rising edge of xExecute, the output signal state will reverse; Convert from original FALSE to True, or from True to FALSE

(4) Program example

DEMO	LD	ST
Define the variable		VAR M0: BOOL; M1: BOOL; ALT_0: ALT; END_VAR
Program example		ALT_0(xExecute:=M0 , xOut:=M1);

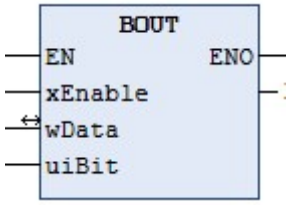
Operation result	

1-1-2-4. Bit data output BOUT

(1) Instruction overview

Set the bit status.

Bit data output [BOUT]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
BOUT	Bit vlaue output		BOUT(xEnable:= , wData:= , uiBit:=);

(2) Related variable

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
xExecute	Input signal	BOOL	FALSE,TRUE	FALSE	The BOOL value to be detected
uiBit	Input signal	UINT	As the data type	0	The bit to be set
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
wData	Output signal	Word	As the data type	0	The value to be set

(3) Function

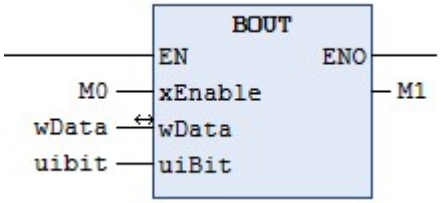
- Detect the rising edge of xExecute and set on the bit set by the uiBit corresponding to wData data.

(4) Note

- WDate is the input and output data, which needs to be set in the variable table and converted to binary. Confirm the conversion bit.

(5) Program example

DEMO	LD	ST
Define the variable	VAR M0:BOOL; wData: WORD; uibit: UINT; M1: BOOL; END_VAR	

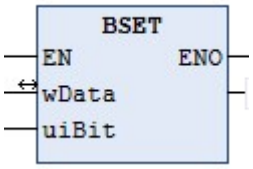
Program example		BOUT(xEnable:= M0, wData:=wData, uiBit:=uibit);
Operation result		
Result analysis	After setting the initial data of wDate to 22 (binary: 10110), uiBit equals 3 and M0 is set on, the bit3 of wDate data is set on, the wDate value changed to 30 (binary: 11110), and the return signal M1 is set to True.	

1-1-2-5. Set on bit BSET

(1) Instruction overview

Set on the bit.

Set on bit [BSET]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
BSET	Set on the bit		BSET(wData:=, uiBit:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
uiBit	Input signal	UINT	As the data type	0	The bit to be set
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
wData	Output signal	Word	As the data type	0	The value to be set

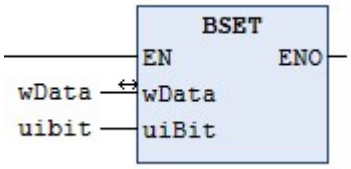
(3) Function

- Set on the bit set by uiBit corresponding to wData.

(4) Note

- WDate is the input and output data, which needs to be set in the variable table and converted to binary. Confirm the conversion bit.
- Regardless of whether the BSET command is used later or not, the ON status will always be maintained, and the BRST command can be used to set this bit to OFF.

(5) Program example

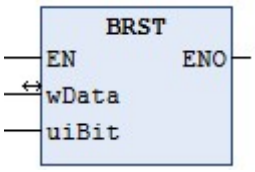
DEMO	LD	ST
Define the variable		VAR wData: WORD; uibit: UINT; END_VAR
Program example		BSET(wData:=wData, uiBit:=uibit);
Operation result		
Result analysis	The initial data of wDate is set to 22 (binary: 10110), and uiBit is equal to 3; The bit3 of wDate data is set on, the wDate value changed to 30 (binary: 11110).	

1-1-2-6. Reset bit BRST

(1) Instruction overview

Reset the bit.

Reset bit [BRST]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
BRST	Reset the bit		BRST(wData:=, uiBit:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
uiBit	Input signal	UINT	As the data type	0	The bit to be set
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
wData	Output signal	Word	As the data type	0	The value to be set

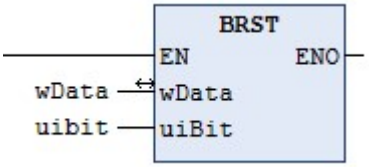
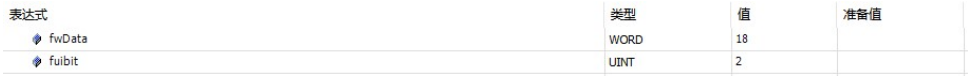
(3) Function

- Reset the bit set by uiBit corresponding to wData

(4) Note

- WDate is the input and output data that needs to be set in the variable table, and the data needs to be converted to binary. Confirm the conversion bit

(5) Program example

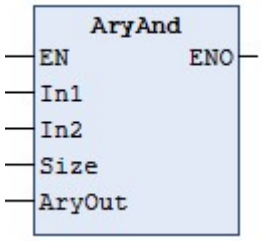
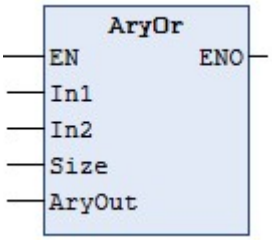
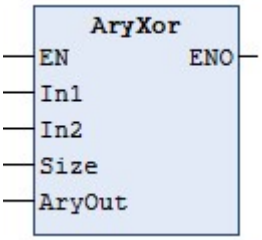
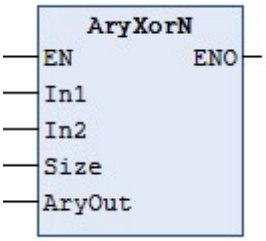
DEMO	LD	ST
Define the variable	VAR fwData: WORD; fuibit: UINT; END_VAR	
Program example		BRST(wData:=fwData, uiBit:=fuibit);
Operation result		
Result analysis	The initial data of wDate is set to 22 (binary: 10110), uiBit is equal to 2, and bit3 of wDate data is reset, wDate changed to 18 (binary: 10010).	

1-1-2-7. Logical AND/Logical OR/XOR/XNOR AryAnd/AryOr/AryXor/AryXorN

(1) Instruction overview

Perform operations on each bit of the Boolean and bit string of the elements between arrays.

Logical AND/Logical OR/XOR/XNOR [AryAnd/AryOr/AryXor/AryXorN]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryAnd	Logical and		AryAnd(In1:= , In2:= , Size:= , AryOut:=);
AryOr	Logical or		AryOr(In1:=, In2:= , Size:= , AryOut:=]);
AryXor	Exclusive or		AryXor(In1:= , In2:= , Size:= , AryOut:=);
AryXorN	Same or		AryXorN(In1:=, In2:=, Size:=, AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In1[], In2[] array	Arithmetic object array	BOOL/BYTE/ WORD/DWORD/ LWORD	As the data type	-	Arithmetic object array
Size	Data	UINT	As the data type	-	Number of operation

	elements				object data elements
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Aryout[]	Operation result array	BOOL/BYTE/ WORD/DWORD /LWORD	As the data type	-	Operation result array

(3) Function

- Perform logical operations on the corresponding bits of each of the "Size" elements starting with In1[] and In2[] in the operation object array. The operation result is saved in the corresponding element of AryOut[]. Therefore, the data types of In1[], In2[], and AryOut[] must be the same.
- The input-output relationship of the operation is as follows:

AryAnd

If one of the two elements is FALSE, the result of the operation is FALSE; Only when both elements are true, the result of the operation is true.

In1[] element bit	In1[] element bit	AryOut[] element result bit
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

AryoR

If one of the two elements is true, the result of the operation is true; Only when both elements are FALSE, the result of the operation is FALSE.

In1[] element bit	In1[] element bit	AryOut[] element result bit
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

AryXor

If two elements are the same, the result of the operation is FALSE; If two elements are different, the result of the operation is true.

In1[] element bit	In1[] element bit	AryOut[] element result bit
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE

AryXorN

If two elements are the same, the result of the operation is True; If two elements are different, the result of the operation is FALSE.

In1[] element bit	In1[] element bit	AryOut[] element result bit
FALSE	FALSE	TRUE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

(4) Note

- The data types of In1[], In2[], and AryOut[] should all be set to the same. If different, the function returns a value of FALSE, and the value of "AryOut" remains unchanged.
- The value of "Size" should not exceed any array area in In1[], In2[], or AryOut[] to avoid PLC exceptions.
- When the value of "Size" is 0, the value of AryOut[] remains unchanged, and the function returns a value of True.
- In1[] and In2[] require Boolean and bitstring types, otherwise the program will compile an error at runtime.

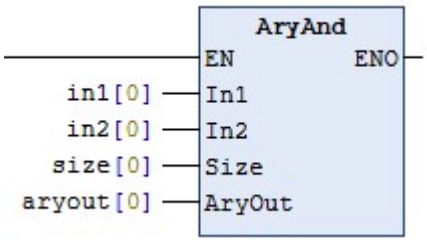
(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define the variable</td><td colspan="2">

```
VAR
    in1:ARRAY[0..5] OF BOOL;
    in2:ARRAY[0..5] OF BOOL;
    size:ARRAY[0..3] OF UINT;
    aryout:ARRAY[0..5] OF BOOL;

END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
AryAnd(In1:=in1[0] ,
    In2:=in2[0] ,
    Size:=size[0] ,
    AryOut:=aryout[0] );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值	准备值	地
in1	ARRAY [0..5] OF BO...			
in1[0]	BOOL	FALSE		
in1[1]	BOOL	FALSE		
in1[2]	BOOL	TRUE		
in1[3]	BOOL	TRUE		
in1[4]	BOOL	FALSE		
in1[5]	BOOL	FALSE		
in2	ARRAY [0..5] OF BO...			
in2[0]	BOOL	FALSE		
in2[1]	BOOL	FALSE		
in2[2]	BOOL	FALSE		
in2[3]	BOOL	TRUE		
in2[4]	BOOL	FALSE		
in2[5]	BOOL	FALSE		
size	ARRAY [0..3] OF UINT			
aryout	ARRAY [0..5] OF BO...			
aryout[0]	BOOL	FALSE		
aryout[1]	BOOL	FALSE		
aryout[2]	BOOL	FALSE		
aryout[3]	BOOL	TRUE		
aryout[4]	BOOL	FALSE		
aryout[5]	BOOL	FALSE		
fwData	WORD	0		

</td></tr>
<tr><td>Result analysis</td><td colspan="2">When in1[0] is FALSE, in2[0] is FALSE, aryout[0] result is FALSE;
When in1[0] is FALSE, in2[0] is TRUE, aryout[0] result is FALSE;
When in1[0] is TRUE, in2[0] is FALSE, aryout[0] result is FALSE;
When in1[0] is TRUE, in2[0] is TRUE, aryout[0] result is TRUE.</td></tr>
</table>

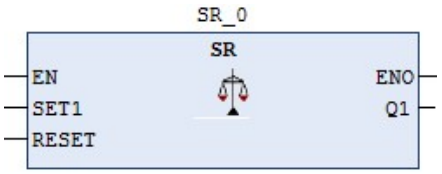
1-1-3. Set priority and reset priority trigger instructions

In a relay system, several pairs of contacts of a relay act simultaneously. In PLC, instructions are executed one by one, and the execution of instructions is sequential, without any "simultaneous" instructions. So the setting and reset commands for coil format have priority. The set and reset inputs of SR trigger and RS trigger are in the same instruction, and the set and reset inputs are executed by the one below the instruction input. The SR trigger is a "set priority" type trigger. When the set signal (SET1) and reset signal (RESET) are both 1, the trigger ultimately enters the set state. The RS trigger is a "reset priority" type trigger. When the set signal (SET) and reset signal (RESET1) are both 1, the trigger ultimately enters the reset state.

1-1-3-1. Set priority trigger SR

When both the set input signal and the reset input signal are valid. Set input signal priority.

Set priority trigger [SR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

Instruction	Instruction icon	Function
SR		SR_0(SET1:= , RESET:= , Q1=>);

Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
SET1	Set signal	BOOL	[FALSE,TRUE]	FALSE	-
RESET	Reset siganl	BOOL	[FALSE,TRUE]	FALSE	-
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q1	Output status	BOOL	[FALSE,TRUE]	FALSE	Set priority output

Function: Set bistable trigger, with set priority. Logical relationship: Q1=(NOT RESET AND Q1) OR SET1, where SET1 is the set signal and RESET is the reset signal.
Syntax: When SET1 is "1", regardless of whether RESET is "1" or not, Q1 output is "1". When SET1 is "0", if Q1 output is "1", once RESET is "1", Q1 output will immediately reset to "0". If Q1 output is "0", regardless of whether RESET is "1" or "0", Q1 output remains "0".

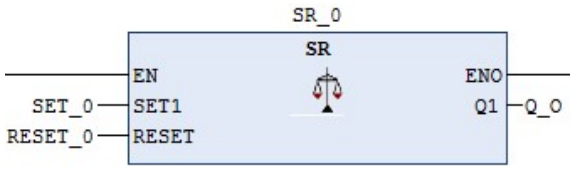
SR status table

SET1	RESET	Q1 output
0	0	Maintain the original state
1	0	1
0	1	0

1	1	1

DEMO	LD	ST
Define variables		
Sample Program		
Operation result		

```
VAR
    SR_0    : SR;
    SET_0   :BOOL:=TRUE;
    RESET_0:BOOL;
    Q_0     :BOOL;
END_VAR
```



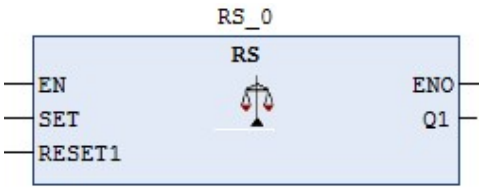

```
1  SR_0(SET1:= SET_0,
2       RESET:= RESET_0,
3       Q1=> Q_0);
```

1-1-3-2. Reset priority trigger RS

When both the set input signal and the reset input signal are valid. Reset input signal priority.

Reset priority trigger [RS]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

Instruction	Instruction icon	Function
RS		




```
RS_0(SET:= ,
     RESET1:= ,
     Q1=> );
```

Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
SET1	Set signal	BOOL	[FALSE,TRUE]	FALSE	-
RESET1	Reset siganl	BOOL	[FALSE,TRUE]	FALSE	-
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q1	Output status	BOOL	[FALSE,TRUE]	FALSE	Reset priority

					output

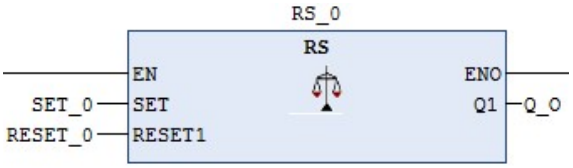
Function: Reset the bistable trigger, with reset priority. Logical relationship: Q1=NOT RESET1 AND (Q1 OR SET), where SET is the set signal and RESET1 is the reset signal.
Syntax: When RESET1 is "1", regardless of whether SET is "1" or not, Q1 output is "0". When RESET1 is "0", if Q1 output is "0", once SET is "1", Q1 output is immediately set to "1". If Q1 output is "1", regardless of whether SET is "1" or "0", Q1 output remains "1".

RS status table

SET	RESET1	Q1 output
0	0	Maintain the original state
1	0	1
0	1	0
1	1	0

DEMO	LD	ST
Define variables		
Sample Program		
Operation result		

```
VAR
    RS_0: RS;
    SET_0  :BOOL;
    RESET_0:BOOL:=TRUE;
    Q_0    :BOOL;
END_VAR
```




```
1  RS_0(SET:= SET_0,
2       RESET1:= RESET_0,
3       Q1=> Q_0);
```

表达式	类型	值
RS_0	RS	
SET_0	BOOL	FALSE
RESET_0	BOOL	TRUE
Q_0	BOOL	FALSE

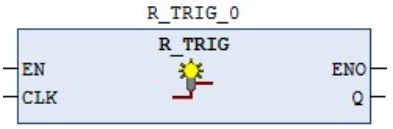
1-1-4. Edge detection instruction

The edge detection instruction is used to detect changes in the rising edge (signal from 0-->1) and falling edge (signal from 1-->0) of the BOOL signal. In each scanning cycle, the signal state is compared with its state in the previous scanning cycle. If different, it indicates a jump edge. Therefore, the signal state from the previous cycle must be stored in order to be compared with the new signal state.

1-1-4-1. Rising edge detection R_TRIG

For detecting the rising edge.

Rising edge detection [R_TRIG]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

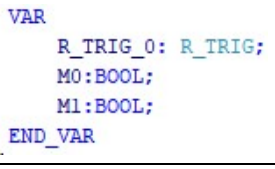
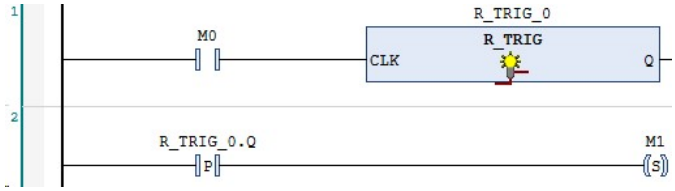
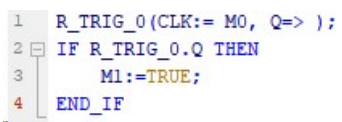
Instruction	Instruction icon	Function
R_TRIG		R_TRIG_0(CLK:= , Q=>);

Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CLK	Input signal	BOOL	[FALSE,TRUE]	FALSE	The rising edge of the signal (from 0 to 1) is effective
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Output signal	BOOL	[FALSE,TRUE]	FALSE	Detect the rising edge

Function: Used to detect the rising edge.

Syntax: When CLK changes from "0" to "1", the rising edge detector starts, and the Q output first changes from "1" to "0", lasting for one PLC operation cycle. If CLK remains at "1" or "0" continuously, Q output remains at "0".

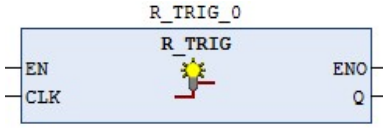
DEMO	LD	ST
Define variables		
Sample Program		

Operation result	

1-1-4-2. Falling edge detection F_TRIG

For detecting the falling edge.

Falling edge detection [F_TRIG]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

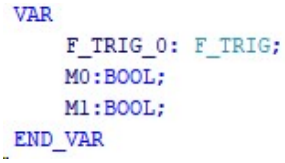
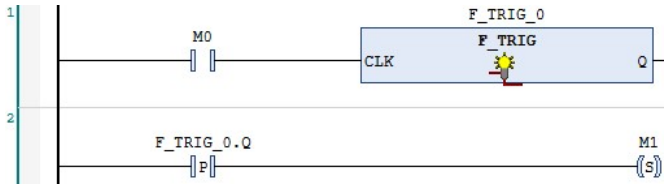
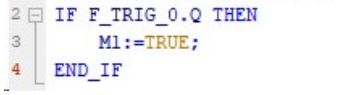
Instruction	Instruction icon	Function
F_TRIG		F_TRIG_0(CLK:= , Q=>);

Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CLK	Input signal	BOOL	[FALSE,TRUE]	FALSE	The falling edge of the signal (from 1 to 0) is effective
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Output signal	BOOL	[FALSE,TRUE]	FALSE	Detect the falling edge

Function: Used to detect the falling edge.

Syntax: When CLK changes from "1" to "0", the falling edge detector starts, and the Q output first changes from "1" to "0", lasting for one PLC operation cycle. If CLK remains at "1" or "0" continuously, Q output remains at "0".

DEMO	LD	ST
Define variables		
Sample Program		

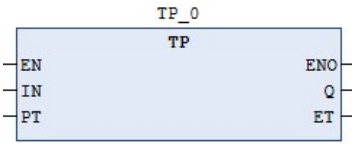
Operation result	

1-2. Timer

1-2-1. Pulse timer TP

Pulse timer.

Pulse timer [TP]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

Command	Command icon	ST
TP		TP_0(IN:= , PT:= , Q=> , ET=>);

Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN	Input signal	BOOL	[FALSE,TRUE]	FALSE	FALSE → TRUE start timing
PT	Setting time	TIME	-	T#0ms	-
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Timing completed	BOOL	[FALSE,TRUE]	FALSE	TRUE → FALSE timing completed
ET	Current timing	TIME	-	T#0ms	-

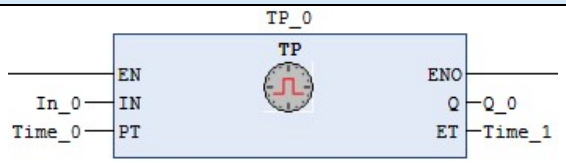
Function: Pulse timing.

Once IN becomes "True", Q becomes "True", and time will start counting in milliseconds in ET until its value equals PT and Q becomes "FALSE".

When the PT time expires, ET will maintain a timed time until IN becomes "FALSE" and returns to T#0ms. If the input IN has already changed to "FALSE" before reaching the PT timing time, the ET will immediately reset to T # 0ms after timing to the set PT time.

When IN is set to FALSE and PT=T#0ms, the timer can be reset.

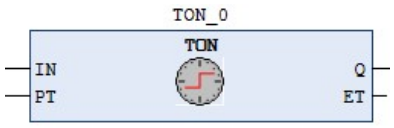
DEMO	LD	ST
Define variables		VAR TP_0 : TP; In_0 :BOOL; Time_0:TIME; Q_0 :BOOL; Time_1:TIME; END_VAR

DEMO	LD	ST
Sample Program		1 TP_0(IN:= In_0, 2 PT:= Time_0, 3 Q=> Q_0, 4 ET=> Time_1);
Operation result		

1-2-2. Power on delay timer TON

After a delay in power on for a period of time, output a timing completion signal.

Power on delay timer [TON]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

Command	Command icon	Function
TON		TON_0(IN:= , PT:= , Q=> , ET=>);

Related variables

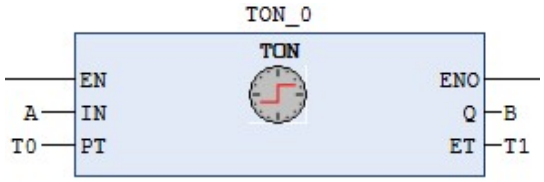
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN	Input signal	BOOL	[FALSE,TRUE]	FALSE	FALSE→TRUE start timing, keep it true during the timing process
PT	Setting time	TIME	-	T#0ms	-
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Timing completed	BOOL	[FALSE,TRUE]	FALSE	After the timing is completed, FALSE becomes True
ET	Current timing	TIME	-	T#0ms	-

Function: Power on delay timing.
When the input IN changes from "FALSE" to "True", the timer starts. When the signal IN at the input end is always maintained at "True" and ET reaches the timed time PT, its output end Q is "True".
The output end ET provides a timing time, which starts from T#0s and ends at the set PT time. When ET reaches PT, it will maintain a timed time until IN becomes "FALSE".
If the input IN becomes "FALSE" and the output ET immediately becomes T#0s before ET reaches the PT timing time. So in order to restart the timer, IN can be set to "FALSE".

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    TON_0: TON;
    A:BOOL;
    T0:TIME;
    B:BOOL;
    T1:TIME;
END_VAR
```

</td></tr>
<tr><td>Sample Program</td><td>



</td><td>

```
1  TON_0(IN:= A,
2       PT:= T0,
3       Q=> B,
4       ET=> T1);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

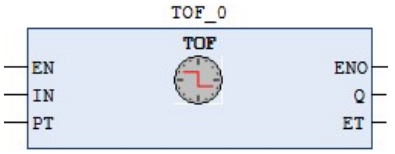
表达式	类型	值
TON_0	TON	
IN	BOOL	TRUE
PT	TIME	T#10s
Q	BOOL	TRUE
ET	TIME	T#10s
A	BOOL	TRUE
T0	TIME	T#10s
B	BOOL	TRUE
T1	TIME	T#10s

</td></tr>
</table>

1-2-3. Power off delay timer TOF

After a power outage delay for a period of time, output a timing completion signal.

Power off delay timer [TOF]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

Command	Command icon	Function
TOF		TOF_0(IN:= , PT:= , Q=> , ET=>);

Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN	Input signal	BOOL	[FALSE,TRUE]	FALSE	TRUE → FALSE start timing, keep it as FALSE during the timing process
PT	Setting time	TIME	-	T#0ms	-
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Timing completed	BOOL	[FALSE,TRUE]	FALSE	After the timing is completed, TRUE becomes FALSE
ET	Current timing	TIME	-	T#0ms	-

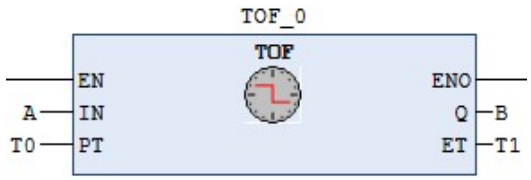
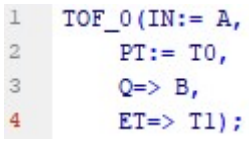
Function: Power off delay timing.

When the input terminal IN changes from "FALSE" to "True", the Q output signal of the timer is "True". When the input terminal IN starts to change to "FALSE", the timer starts. During the timer timing process, its output Q remains "True". When the timing time PT is reached, the output terminal Q resets.

The output end ET provides a timing time, which starts from T#0s and ends at the set PT time. When ET reaches PT, it will maintain a timed time until IN becomes "True".

If the input IN becomes "True" and the output ET immediately becomes T#0s before ET reaches the PT timing time. So in order to restart the timer, IN can be set to "True".

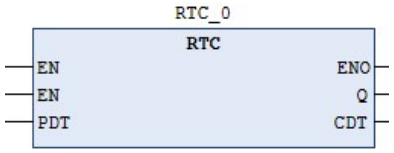
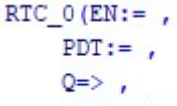
DEMO	LD	ST
Define variables	VAR TOF_0: TOF; A :BOOL; T0 :TIME; B :BOOL; T1 :TIME; END_VAR	

Sample Program		
Operation result		

1-2-4. RTC

Start timing from the set time.

RTC [RTC]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

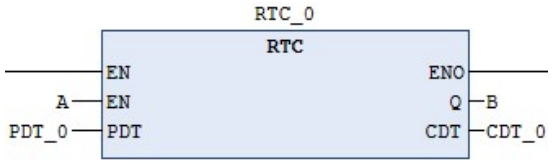
Command	Command icon	Function
RTC		

Related variable

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
EN	Enable	BOOL	[FALSE,TRUE]	FALSE	Rising edge: CDT set to PDT, start rising at CDT Falling edge: CDT set to DT#1970-01-01 00:00:00
PDT	Set the time and date when it will be launched	DATE_AND_TIME	(DT#1970-1-1-0:0:0) -(2106-2-7-6:28:15)	DT#1970-1-1-0:0:0	Preset date and time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Status output	BOOL	[FALSE,TRUE]	FALSE	CDT is TRUE when counting
CDT	Current counting time	DATE_AND_TIME	(DT#1970-1-1-0:0:0) -(2106-2-7-6:28:15)	DT#1970-1-1-0:0:0	Current counting time and date status

	and date status				

Once EN is "True", CDT will be set to the time given by PDT and will be counted in seconds. Once EN is reset to "FALSE" and the output variable Q is "FALSE", CDT will be reset to the initial value DT #1970-01-00:00:00.
PDT time is only effective on the rising edge.

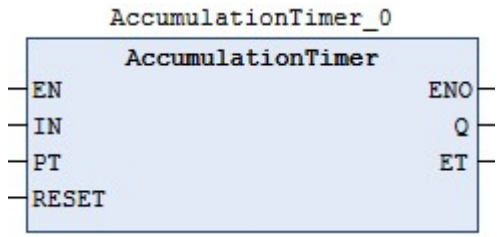
DEMO	LD	ST
Define variables	VAR RTC_0: RTC; A :BOOL; PDT_0:DATE_AND_TIME; B :BOOL; CDT_0:DATE_AND_TIME; END_VAR	
Sample Program		1 RTC_0(EN:= A, 2 PDT:= PDT_0, 3 Q=> B, 4 CDT=> CDT_0);
Operation result		

1-2-5. Accumulation timer

(1) Instruction overview

Accumulation timer.

Accumulation timer [AccumulationTimer]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
AccumulationTimer	Accumulated timer		AccumulationTimer_0(IN:= , PT:= , RESET:= , Q=> , ET=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Timer input	BOOL	FALSE/TRUE	FALSE	FALSE: timer stop TRUE: timer working
PT	Set time	TIME	As the data type	0	Maximum value of timing
Reset	Reset	BOOL	FALSE/TRUE	FALSE	FALSE: timer not reset TRUE: timer reset
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Timer output	BOOL	FASLE/TRUE	-	TRUE: “ET” reached “PT” FALSE: “ET” not reach “PT”
ET	Accumulated time	TIME	As the data type	-	Accumulated time

(3) Function

- A timer that accumulates the time when the input "In" is true. Set the time unit to ms and the timing accuracy to 100ns.
- When resetting "Reset" to FALSE and "In" changes from FALSE to True, the timer starts and the cumulative time "ET" increases simultaneously with the time.
- When "In" changes to FALSE, the timer stops. But maintain the "ET" value at that moment.
- When "In" changes to true again, the timer starts again. "ET" increases from the holding value.
- When "ET" reaches the set time "PT", the timer output "Q" changes to True. At this point, "ET" stops increasing.
- When "Reset" becomes true, the timer resets. The value of "ET" becomes 0, and "Q" becomes FALSE.

(4) Note

- "ET" and "Q" are updated during the execution of this command. Therefore, strictly speaking, the condition for "Q" to become true is not the moment when the cumulative time of timer operation equals "PT". But rather the moment when the cumulative time of the timer's operation reaches "PT" and this

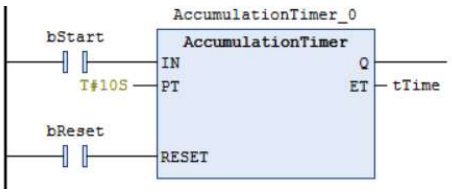
instruction is first executed. Therefore, a delay of up to one task cycle will occur.

- PT and ET are set in ms, with a timing accuracy of 100ns.
- When both "In" and "Reset" become true, "Reset" takes priority. The value of "ET" becomes 0, and "Q" becomes FALSE.
- If the value of "In" is already true at the beginning of the run, timing will start from that moment.
- When T#0ms is set for "PT", the value of "In" becomes true, and "Q" becomes true.
- Before the value of "ET" reaches the value of "PT", the value of "PT" can be changed, but the cumulative timing "ET" is still calculated according to the original "PT".

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
   AccumulationTimer_0: AccumulationTimer;
   bStart: BOOL;
   bReset: BOOL;
   tTime: TIME;
   XDONE: BOOL;
END_VAR
```

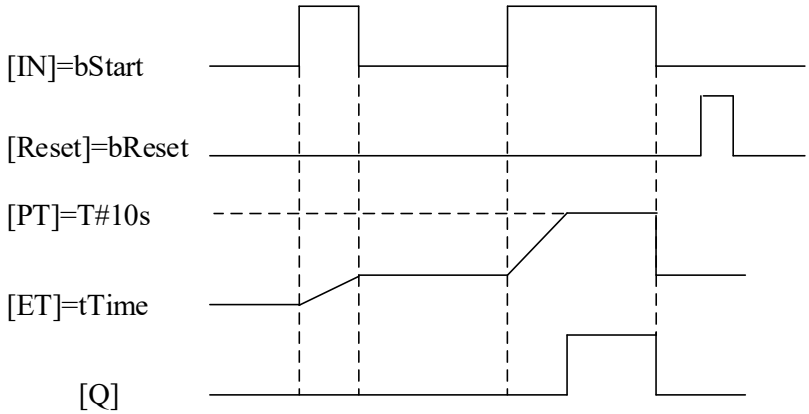
</td></tr>
<tr><td>Sample Program</td><td>



</td><td>

```
AccumulationTimer_0(
    IN:= bStart,
    PT:= T#10S,
    RESET:= bReset,
    Q=> XDONE,
    ET=> tTime);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">



When the cumulative time for variable bStart to be true reaches 10 seconds, the value of variable XDONE becomes true.

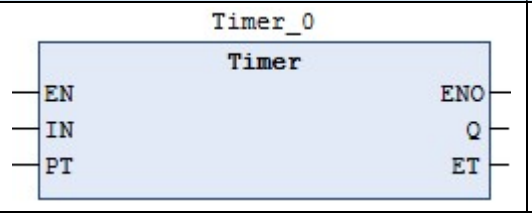
</td></tr>
</table>

1-2-6. 100ms Timer

(1) Instruction overview

100ms timer.

100ms timer [Timer]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
Timer	100ms timer		Timer_0(IN:= , PT:= , Q=> , ET=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Timer input	BOOL	FALSE/TRUE	FALSE	FALSE: timer stop TRUE: timer working
PT	Set time	UINT	As the data type	0	The maximum value of timing, unit is 100ms
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Timer output	BOOL	FASLE/TRUE	-	TRUE: “ET” reached “PT” FALSE: “ET” not reach “PT”
ET	Remaining time	UINT	As the data type	-	Remaining time

(3) Function

- After a set time has passed since startup, a timer outputs true. Set the time in 100ms units.
- If the value of "In" input to the timer becomes FALSE, the timer will reset. The remaining time "ET" will be set to the set time "PT", and the value of the timer output "Q" will become FALSE.
- If "In" changes from FALSE to True, start the timer. The value of "ET" decreases over time.
- After "ET" reaches 0, the timer output "Q" becomes True. At this point, the value of "ET" stops decreasing.
- After the timer starts, even if "ET" does not reach 0, if "In" becomes FALSE, the timer will reset.

(4) Note

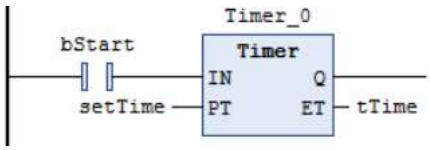
- Calling multiple TIMER within the same POU may cause exceptions in the program.
- The timing error required for "PT" and "Q" to become true is 1 task cycle. The timer determines whether the timing of "ET" for each task cycle has reached "PT". If the timing of "ET" reaches "PT" after judgment, it will delay by 1 task cycle.
- At the beginning of the run, if the value of "In" is already true, start timing immediately.
- When the value of "PT" is changed, it needs to be applied during the next timer reset; It does not take effect during the timing process.
- Manually modifying the values of internal parameter variables (ET, Q) cannot take effect.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
   Timer_0: Timer;
   bStart: BOOL;
   tTime: UINT;
   setTime: UINT:= 100;
   XDONE: BOOL;
END_VAR
```

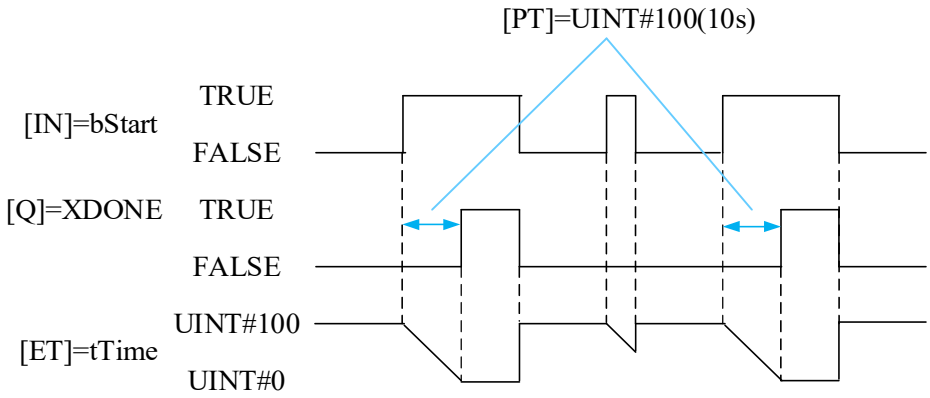
</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
Timer_0(
    IN:= bStart,
    PT:= setTime,
    Q=> XDONE,
    ET=> tTime);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">



After variable bStart becomes True for 100*100ms (10s), the value of variable XDONE becomes True.

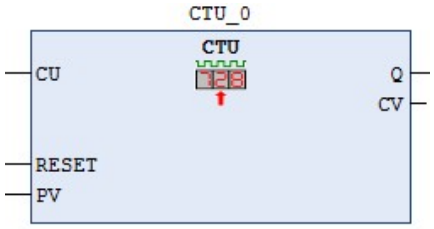
</td></tr>
</table>

1-3. Counter

1-3-1. Up counter CTU

When the rising edge signal is triggered once, the count value is increased by 1.

Up counter [CTU]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

Command name	Command icon	Function
CTU		CTU_0(CU:= , RESET:= , PV:= , Q=> , CV=>);

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CU	Counting signal input	BOOL	[FALSE,TRUE]	FALSE	Rising edge: CV count value plus 1
RESET	Reset input	BOOL	[FALSE,TRUE]	FALSE	TRUE: CV reset to 0
PV	Set upper limit for counting	WORD	0-65535	0	Set upper limit for counting
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Status output	BOOL	[FALSE,TRUE]	FALSE	When CV>=PV, it is TRUE
CV	Current counting value	WORD	0-65535	0	Current counting value

When the signal of CU changes from "FALSE" to "True", the current calculated value is increased by 1 and displayed through the output terminal CV. When the count reaches the upper limit of 65535, the counter will not increase again.

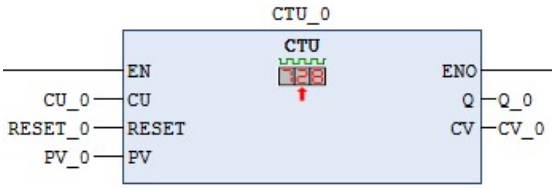
When the signal state of RESET is "True", the CV of the counter is initialized to 0 and Q is "FALSE", and the rising edge no longer has any effect on CU.

When the CV value is greater than or equal to PV, the output Q is "True". At this point, CV can still continue to accumulate, and the output Q continues to be output "True".

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    CU_0    :BOOL;
    RESET_0:BOOL;
    PV_0    :WORD := 3;
    Q_0     :BOOL;
    CV_0    :WORD;
    CTU_0   : CTU;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
CTU_0(CU:= CU_0,
    RESET:= RESET_0,
    PV:= PV_0,
    Q=> Q_0,
    CV=> CV_0);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

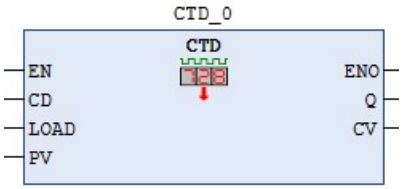
表达式	类型	值
CU_0	BOOL	TRUE
RESET_0	BOOL	FALSE
PV_0	WORD	3
Q_0	BOOL	TRUE
CV_0	WORD	3
⊞ CTU_0	CTU	

</td></tr>
</table>

1-3-2. Down counter CTD

When the rising edge signal is applied once, the count decreases by 1.

Down counter [CTD]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

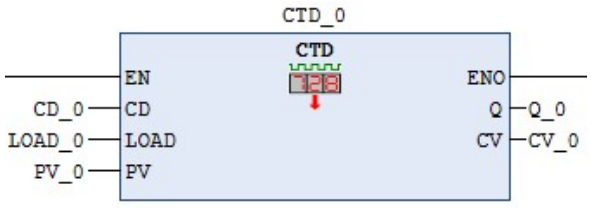
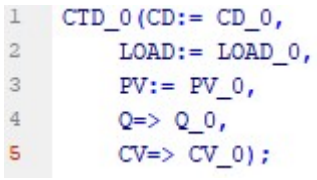
Command name	Command icon	Function
CTD		CTD_0(CD:= , LOAD:= , PV:= , Q=> , CV=>);

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CU	Counting signal input	BOOL	[FALSE,TRUE]	FALSE	Rising edge: CV count value minus 1
LOAD	Load signal	BOOL	[FALSE,TRUE]	FALSE	True: Set CV as the initial value PV
PV	Decline starting value	WORD	0-65535	0	Decline starting value
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Status output	BOOL	[FALSE,TRUE]	FALSE	CV=0, it is TRUE
CV	Current counting value	WORD	0-65535	0	Current counting value

The load signal needs to be changed from "FALSE" to "True", CV will be set to the starting value given by PV, output Q will be changed from the initial "True" to "FALSE", and the load will be changed to the state "FALSE" before the function block can take effect.

When the CD signal at the input end of the counter changes from "FALSE" to the state "True", the current count value is subtracted by 1 and displayed on the output end CV. When the value of CV decreases to 0, Q will be set to True, and the count value will no longer decrease.

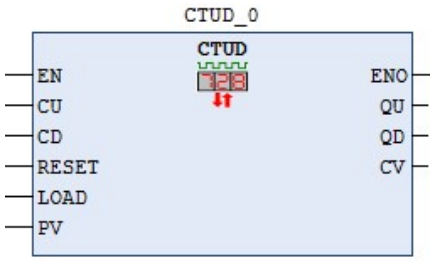
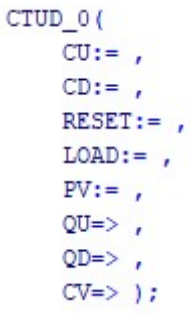
DEMO	LD	ST
Define variables	VAR CD_0 :BOOL; LOAD_0 :BOOL; PV_0 :WORD := 3; Q_0 :BOOL; CV_0 :WORD; CTD_0 : CTD; END_VAR	

Program example		
Operation result		
Result analysis	After setting LOAD_0 to True, Q_0 changes from True to FALSE and CV_0 becomes 3. Reset LOAD_0 to FALSE again. At this point, CD_0 is conducting, and the value of CV_0 is reduced by 1.	

1-3-3. Up/down counter CTUD

You can increase the count by 1, or decrease the count by 1.

Up/Down counter [CTUD]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Standard		

Command name	Command icon	Function
CTUD		

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CU	Count up input	BOOL	[FALSE,TRUE]	FALSE	Rising edge: CV count value plus 1
CD	Count down input	BOOL	[FALSE,TRUE]	FALSE	Rising edge: CV count value minus 1
RESET	Reset input	BOOL	[FALSE,TRUE]	FALSE	TRUE: CV reset to 0
LOAD	Load signal	BOOL	[FALSE,TRUE]	FALSE	True: Set CV as the initial value PV
PV	Set upper limit for counting	WORD	0-65535	0	Set upper limit for counting
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
QU	Increase count status output	BOOL	[FALSE,TRUE]	FALSE	CV>=PV, it is TRUE
QD	Countdown status output	BOOL	[FALSE,TRUE]	FALSE	CV=0, it is TRUE
CV	Current counting value	WORD	0-65535	0	Current counting value

When the CU signal at the count input changes from "FALSE" to the state "True", the current count value is increased by 1 and displayed on the output CV.

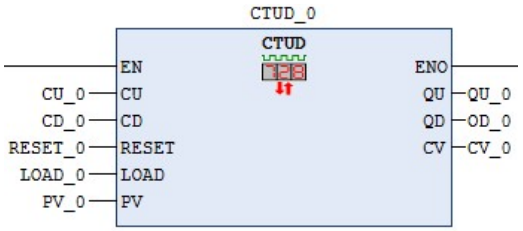
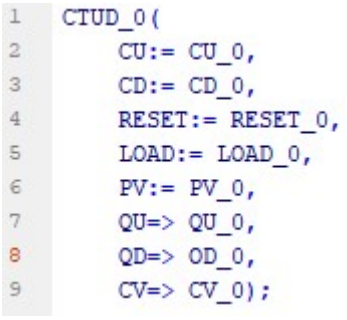
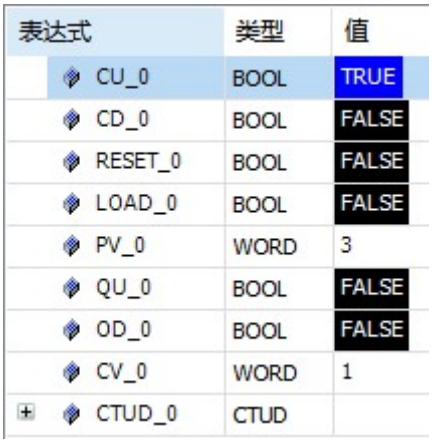
When the signal of the CD at the counting input end changes from "FALSE" to the state "True" (the loading signal load needs to be changed from "FALSE" to "True" and then to the state "FALSE" before the function block can take effect), the current count value is subtracted by 1 and displayed on the output end CV. If both inputs have rising edges, the current count value will remain unchanged.

When the counting value reaches the upper limit value of 65535, the rising edge of the counting input CU no longer works. Therefore, even if there is a rising edge on the counting input CU, its value will not increase. Similarly, when the count value reaches the lower limit value of 0, the counting input CD will not take effect. Therefore, even if the counting input CD shows an upward edge, the count value will not decrease.

When the CV value is greater than or equal to the PV value, the output QU is "True". When the CV value is equal to 0, the output QD is "True".

DEMO	LD	ST
Define variables		

```
VAR
    CU_0    :BOOL;
    CD_0    :BOOL;
    RESET_0:BOOL;
    LOAD_0 :BOOL;
    PV_0    :WORD := 3;
    QU_0    :BOOL;
    OD_0    :BOOL;
    CV_0    :WORD;
    CTUD_0: CTUD;
END_VAR
```

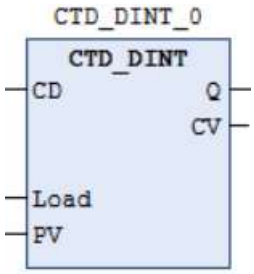
Program example		
Operation result		

1-3-4. Subtract counter CTD_XXX

(1) Instruction overview

A counter that performs subtraction operations when inputing signal of the counter. The data types for preset values and count values are any of DINT, LINT, UDIND, or ULINT.

Subtract counter [CTD_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
CTD_XXX	Subtract counter		CTD_DINT_0(CD:= , Load:= , PV:= , Q=> , CV=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CD	Counter input	BOOL	FALSE/TRUE	FALSE	Counter input
Load	Load signal	BOOL	FALSE/TRUE	FALSE	When set TRUE, load PV to CV
PV	Preset value	DINT/LINT/ UDINT/ULINT	As the data type	0	Preset value of the counter
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Counter output	BOOL	FALSE/TRUE	FALSE	TRUE: counter outputs ON FALSE: counter outputs OFF
CV	Counting value	DINT/LINT/UDINT/ULINT	As the data type	-	Counter present value

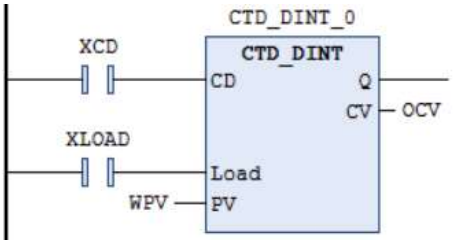
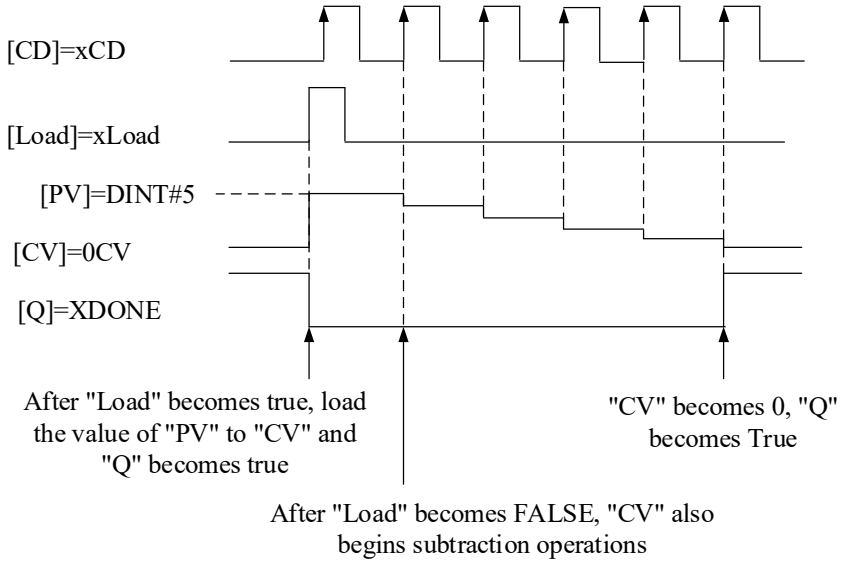
(3) Function

- Subtraction counter. The data types for preset values and count values are any of DINT, LINT, UDINT, or ULINT.
- The instruction name varies depending on the data type of "PV" and "CV". For example, when both are DINT type, the instruction name is CTD_ DINT.
- When the load signal "Load" is set to True, the preset value "PV" is loaded into the count value "CV", and the counter output "Q" becomes FALSE; When the load signal "Load" is set to True and the "PV" value is 0, the counter output "Q" becomes True.
- When the counter input signal "CD" is on the rising edge, "CV" will perform subtraction operation. When the value of "CV" is equal to 0, the value of "Q" becomes True. When the value of "CV" is manually modified to be less than 0, "Q" becomes True. In addition, the "Q" value cannot be modified during operation.
- When the value of "CV" is less than 0, even if "CD" is on the rising edge, "CV" does not perform subtraction operations.
- During the period when "Load" is true, "CV" cannot be modified and subtraction operations are not performed.

(4) Note

- When starting the counter again after the countdown is completed, please set the value of "Load" to True before setting it to FALSE.
- Please set the data types for "PV" and "CV" to be the same.
- When a negative number is set for "PV", the value of "PV" is loaded into "CV" when the value of "Load" becomes true. The value of "CV" is less than 0, so the value of "Q" will immediately become true. Afterwards, even if "CD" changes, "CV" does not perform subtraction operations.
- If the value of "PV" exceeds the range of values, "PV" will be implicitly converted to an error value, and the function will not have any exceptions.
- When the power is disconnected or the program controls the action mode in the state where the value of "CD" is FALSE, and the instruction is restarted, if the value of "CD" becomes true, "CV" performs one subtraction operation.
- When using this instruction through a ladder program, if an abnormality occurs in the front section of the circuit of this instruction, the value of "Q" will become FALSE.

(5) Program example

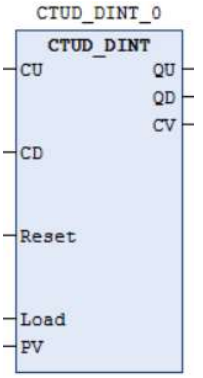
DEMO	LD	ST
Define variables	VAR CTD_DINT_0: CTD_DINT; XCD: BOOL; XLOAD: BOOL; WPV: DINT:= 5; OCV: DINT; XDONE: BOOL; END_VAR	
Sample program		CTD_DINT_0(CD:= XCD, Load:= XLOAD, PV:= WPV, Q=> XDONE, CV=>OCV);
Principle analysis		

1-3-5. Up/Down Counter CTUD_XXX

(1) Instruction overview

A counter that performs addition and subtraction operations based on the input of the addition counter and the subtraction counter. The data types for preset values and count values are any of DINT, LINT, UDIND, or ULINT.

Up/Down counter [CTUD_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
CTUD_XXX	Up/down counter		CTUD_DINT_0(CU:= , CD:= , Reset:= , Load:= , PV:= , QU=> , QD=> , CV=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CU	Addition counter input	BOOL	FALSE/TRUE	FALSE	Addition counter input
CD	Subtraction counter input	BOOL	FALSE/TRUE	FALSE	Subtraction counter input
Reset	Reset signal	BOOL	FALSE/TRUE	FALSE	When set to true, reset 'CV' to 0
Load	Load signal	BOOL	FALSE/TRUE	FALSE	When set to true, load 'PV' to 'CV'
PV	Preset value	DINT/LINT/ UDINT/ULINT	As the data type	0	The preset value of the counter
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
QU	Addition counter output	BOOL	FALSE/TRUE	FALSE	TRUE: up counter output ON FALSE: up counter output OFF
QD	Subtraction counter output	BOOL	FALSE/TRUE	FALSE	TRUE: down counter output ON FALSE: down counter output OFF
CV	Count value	DINT/LINT/ UDINT/ULINT	As the data type	-	The current value of the counter

(3) Function

- A counter that performs addition and subtraction operations based on the input signals of the addition counter and the subtraction counter.
- It has the function of both an addition counter and a subtraction counter.
- The data types for preset values and count values are any of DINT, LINT, UDINT, or ULINT.
- The instruction name varies depending on the data type of "PV" and "CV". For example, when both are LINT type, the instruction name is CTUD_Lint.
- The function of the addition counter
- When the reset signal "Reset" is set to True, the value of the count value "CV" becomes 0, and the output of the addition counter "QU" becomes FALSE.
- When the input signal "CU" of the addition counter is on the rising edge, make "CV" perform the addition operation. When the value of "CV" is equal to the preset value of "PV", the value of "QU" becomes true. If the value of "CV" is manually modified to be greater than the value of "PV", the value of "QU" becomes true.
- When the value of "CV" is greater than the value of "PV", even if the input signal "CU" is triggered again, "CV" will not change.
- When "CV" is less than or equal to 0, "QD" becomes True.
- The function of the subtraction counter
- When the load signal "Load" is set to True, the preset value "PV" is loaded into the count value "CV", and the subtraction counter output "QD" becomes FALSE.
- When the input signal "CD" of the subtraction counter is on the rising edge, subtract "CV". When the value of "CV" is equal to 0, the value of "QD" becomes true. Manually modify the value of "CV" to be greater than or equal to the value of "PV", and the value of "QU" becomes true. Manually modify the value of "CV" to be less than 0, and the value of "QD" becomes true.
- When the value of "CV" is less than 0, even if the input signal "CD" is triggered again, "CV" will not change.
- Common functions of addition and subtraction counters
- During the period when "Load" or "Reset" is true, ignore "CU" and "CD". "CV" does not perform addition and subtraction operations.
- When "CU" and "CD" are both on the rising edge, "CV" will not change.
- When both "Reset" and "Load" are true, "Reset" takes priority and the value of "CV" becomes 0.
- When "Reset" is set to true, the value of "CV" becomes 0, so the value of "QD" becomes true.
- When "Load" is set to True, the value of "CV" is equal to "PV", so the value of "QU" becomes True.

“Reset”	“Load”	“CV”	“QU”	“QD”	Action
FALSE	FALSE	Below 0	FALSE	TRUE	Only perform addition counter actions When "CU" is on the rising edge, "CV" performs addition operations When CD is on the rising edge, no subtraction operation is performed
		Between 0 and “PV”	FALSE	FALSE	Perform addition/subtraction counter actions When "CU" is on the rising edge, "CV" performs addition operations When CD is on the rising edge, CV performs subtraction operation

		Larger than “PV”	TRUE	FALSE	Only perform subtraction counter action When "CU" is on the rising edge, no addition operation is performed When CD is on the rising edge, CV performs subtraction operation
TRUE	FALSE	0	FALSE	TRUE	
FALSE	TRUE	“PV”	TRUE	FALSE	The reset state of the addition counter, with the value of "CV" set to 0 Reset state of subtraction counter, set the value of "CV" to "PV"
TRUE	TRUE	0	FALSE	TRUE	The reset status of the addition counter, "Reset" takes priority over "Load", and the value of "CV" is set to 0

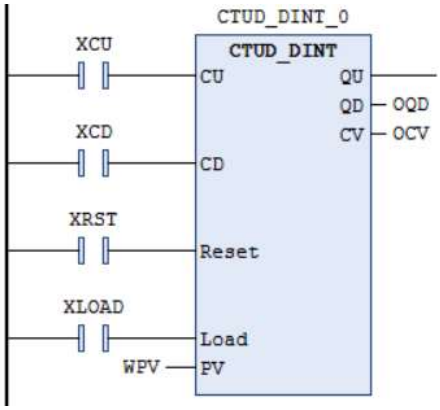
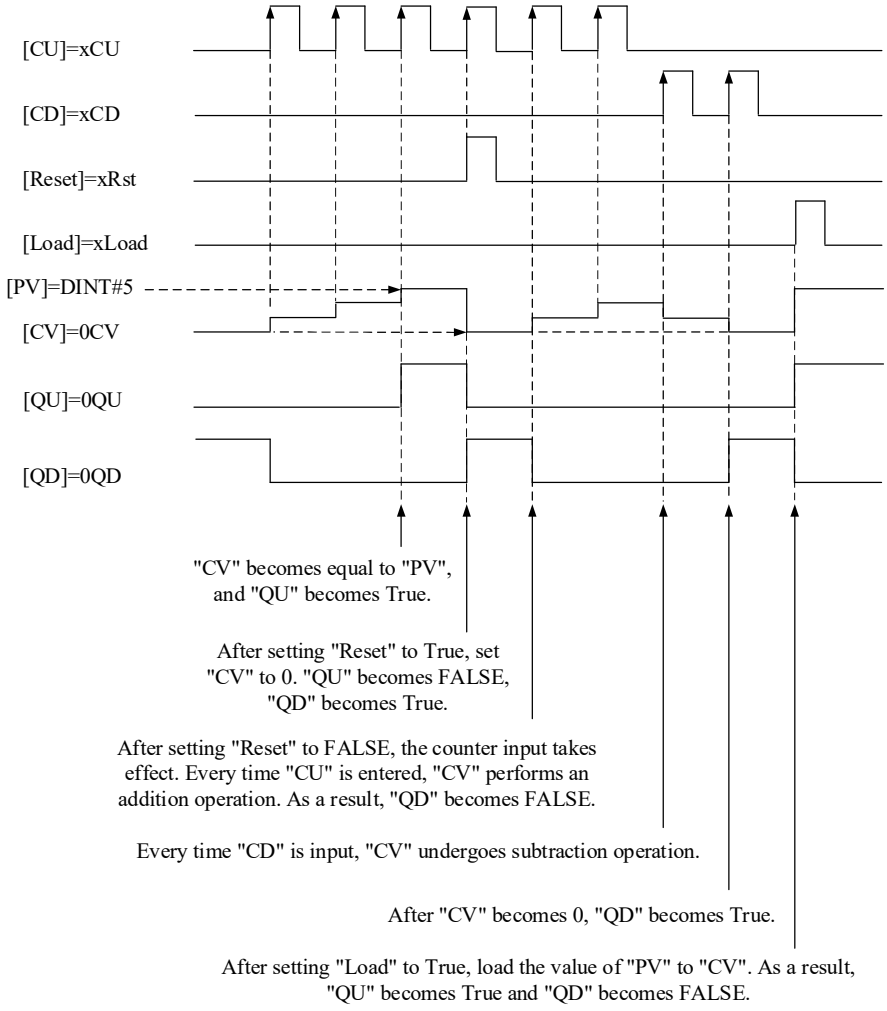
(4) Note

- Please note that to reset the addition counter, when the value of "Reset" is set to True, the value of "QU" becomes FALSE, and the value of "QD" becomes True.
- Please note that to reset the subtraction counter, when the value of "Load" is set to True, the value of "QD" becomes FALSE, and the value of "QU" becomes True.
- When a negative number is set for "PV", the value of "PV" is loaded into "CV" when the value of "Load" becomes true. The value of "CV" is less than 0, so the value of "QD" will immediately become true. Afterwards, even if "CD" changes, "CV" does not perform subtraction operations. The value of "Reset" becomes True, and the value of "CV" becomes 0. The value of "CV" is greater than the value of "PV", so the value of "QU" will immediately become true. Afterwards, even if "CU" changes, "CV" does not perform addition operations.
- If the value of "PV" exceeds the range of values, "PV" will be implicitly converted to an error value, and the function will not have any exceptions.
- Please set the data types for "PV" and "CV" to be the same.
- When the value of "CU" or "CD" is FALSE and the power is disconnected or the program controls the action mode, and the instruction is restarted, if the value of "CU" or "CD" becomes true, "CV" performs addition or subtraction operation once.
- The "Q" value cannot be modified during operation.
- The preset value of "PV" can be set to a negative value, and after triggering "CU"/"CD", it can be added/subtracted normally until it reaches the value of "PV".

(5) Program example

DEMO	LD	ST
Define variables		

```
VAR
   CTUD_DINT_0: CTUD_DINT;
   XCU: BOOL;
   XCD: BOOL;
   XRST: BOOL;
   XLOAD: BOOL;
   WPV: DINT:= 5;
   OQD: BOOL;
   OCV: DINT;
   OQU: BOOL;
END_VAR
```

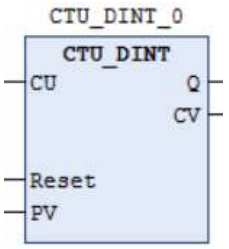
Sample program		``` CTUD_DINT_0(CU:= XCU, CD:= XCD, Reset:= XRST, Load:= XLOAD, PV:= WPV, QU=> OQU, QD=> OQD, CV=> OCV); ```
Principle analysis		

1-3-6. Up counter CTU_XXX

(1) Instruction overview

A counter that performs addition operations when inputing counter input signal. The data types for preset values and count values are any of DINT, LINT, UDIND, or ULINT.

Up counter [CTU_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
CTU_XXX	Up counter		CTU_DINT_0(CU:= , Reset:= , PV:= , Q=> , CV=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CU	Counter input	BOOL	FALSE/TRUE	FALSE	Counter input
Reset	Load signal	BOOL	FALSE/TRUE	FALSE	When set to true, reset to 0 towards' CV '
PV	Preset value	DINT/LINT/ UDINT/ULINT	As the data type	0	The counter preset value
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Q	Counter output	BOOL	FALSE/TRUE	FALSE	TRUE: counter output ON FALSE: counter output OFF
CV	Count value	DINT/LINT/ UDINT/ULINT	As the data type	-	The counter current value

(3) Function

- Addition counter. The data types for preset values and count values are any of DINT, LINT, UDINT, or ULINT.
- The instruction name varies depending on the data type of "PV" and "CV". For example, when both are DINT type, the instruction name is CTU_DINT.
- When the reset signal "Reset" is set to True, the value of the count value "CV" becomes 0, and the counter output "Q" becomes FALSE.
- When the counter input signal "CU" is on the rising edge, make "CV" perform addition operation. When the value of "CV" is equal to the preset value of "PV", the value of "Q" becomes true. When the value of "CV" is manually modified to be greater than "PV", the value of "Q" becomes true. In addition, during the execution process, the "Q" value cannot be modified.
- When the value of "CV" is greater than the value of "PV", even if a larger value of "CU" is entered, "CV"

will not change.

- During the period when "Reset" is true, ignore "CU". CV does not perform addition operations.
- During the period when "Reset" is true, "CV" cannot be modified and does not perform addition operations.

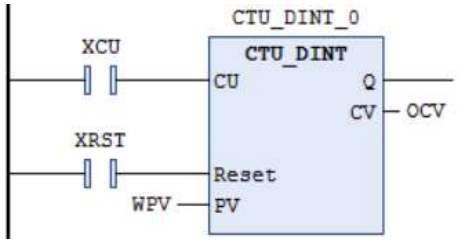
(4) Note

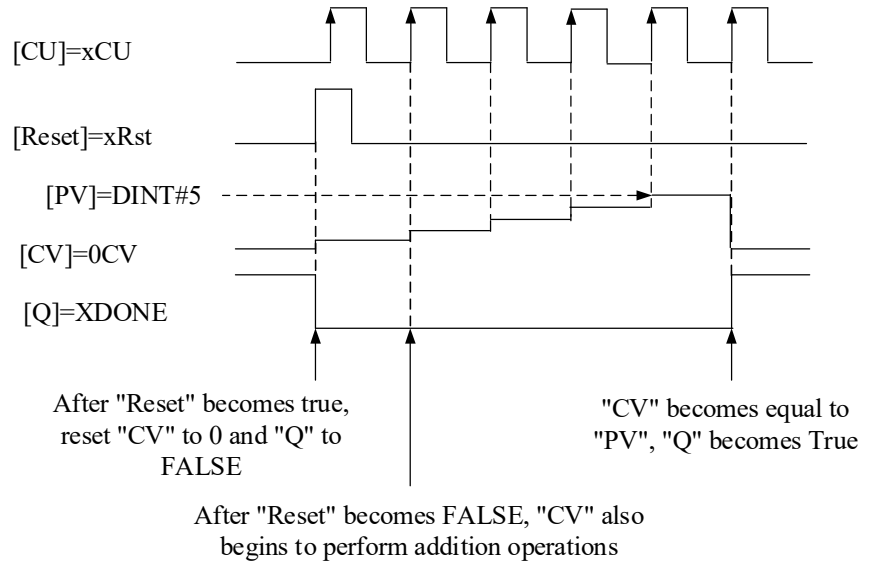
- When starting the counter again after counting, please set the value of "Reset" to True then set to FALSE.
- When a negative number is set for "PV", when the value of "Reset" becomes true, the value of "CV" becomes 0. The value of "CV" is greater than the value of "PV", so the value of "Q" will immediately become true. Afterwards, even if "CU" changes, "CV" does not perform addition operations.
- If the value of "PV" exceeds the range of values, "PV" will be implicitly converted to an error value, and the function will not have any exceptions.
- Please set the data types for "PV" and "CV" to be the same.
- When the value of "Reset" is FALSE, if the value of "PV" changes, its action is shown in the following figure:

"PV" value	Meaning
"CV" greater than that moment	Continue counting
"CV" less than that moment	Count ended. The value of "Q" becomes true, while the value of "CV" remains unchanged at that moment

- When the value of "CU" is in the FALSE state and the power is disconnected or the program control action mode, when this instruction is restarted, if the value of "CU" becomes true, "CV" performs addition operation once.
- When using this instruction through a ladder program, if an abnormality occurs in the front section of the circuit of this instruction, the value of "Q" will become FALSE.

(5) Program example

DEMO	LD	ST
Define variables		VAR CTU_DINT_0: CTU_DINT; XCU: BOOL; XRST: BOOL; WPV: DINT:= 5; OCV: DINT; XDONE: BOOL; END_VAR
Sample program		CTU_DINT_0(CU:= XCU, Reset:= XRST, PV:= WPV, Q=> XDONE, CV=> OCV);

Principle analysis	

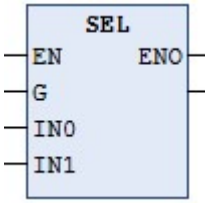
1-4. Data Processing Instructions

1-4-1. Selection commands

1-4-1-1. Binary choice instruction SEL

When the selection switch is FALSE, the output is the first input data, and when the selection switch is True, the output is the second data.

Binary choice [SEL]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

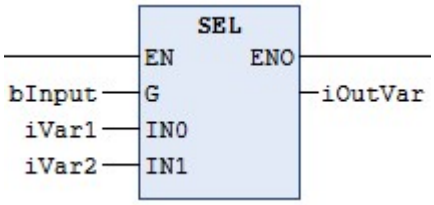
Command	Command icon	Function
SEL		SEL

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
G	Input signal	BOOL	[FALSE,TRUE]	FALSE	FALSE: output IN0 data TRUE: output IN1 data
IN0	Input variable	Any type	As the data type	0	G is FALSE: output IN0 data
IN1	Input variable	Any type	As the data type	0	G is TRUE: output IN1 data
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Output variable	Any type	As the data type	0	Output variable

By selecting a switch, select one of the two input data as the output. If G is FALSE, the result of the return value is IN0; if G is True, the result of the return value is IN1.

DEMO	LD	ST
Define the variable		

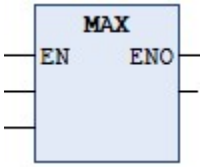
```
VAR
    iVar1  :REAL:=3.3;
    iVar2  :REAL:=3.4;
    iOutVar:REAL;
    bInput :BOOL;
END_VAR
```

DEMO	LD	ST
Sample program		1 iOutVar:=SEL(bInput,iVar1,iV
Operation result		
Result analysis	When the input value bInput is FALSE, the output is 3.3. Conversely, when it is true, the output is 3.4.	

1-4-1-2. Take the maximum value MAX

Choose the maximum value between two values to output.

Take the maximum value [MAX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Command	Command icon	Function
MAX		MAX

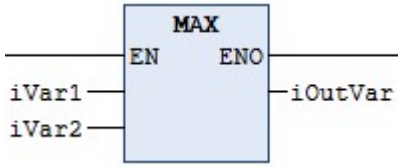
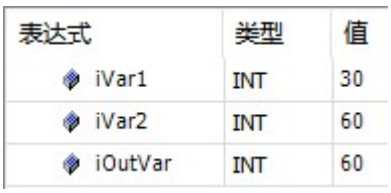
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN0	Input variable	Any type	As the data type	0	Input data 0
INn	Input variable	Any type	As the data type	0	Input data 1
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description

OUT	Output variable	Any type	As the data type	0	Output variable

Given two values, take the maximum value between them.

DEMO	LD	ST

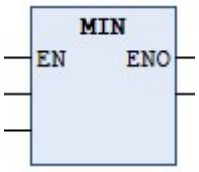
Define the variable	VAR
	iVar1 :INT:=30;
	iVar2 :INT:=60;
	iOutVar:INT;
	END_VAR	

Sample program		1 iOutVar:=MAX(iVar1,iVar2);
Operation result		

1-4-1-3. Take the minimum value MIN

Choose the minimum value between two values to output.

Take the minimum value [MIN]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Command	Command icon	Function
MIN		MIN

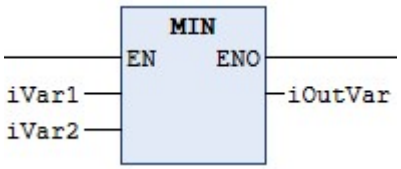
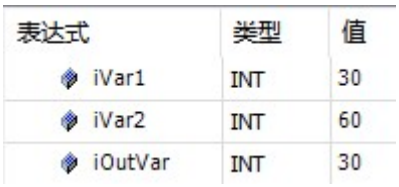
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN0	Input variable	Any type	As the data type	0	Input data 0
INn	Input variable	Any type	As the data type	0	Input data 1

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Output variable	Any type	As the data type	0	Output variable

Given two values, take the minimum value between them.

DEMO	LD	ST

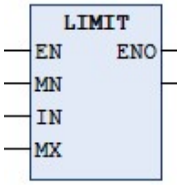
Define the variable	VAR
	iVar1 :INT:=30;
	iVar2 :INT:=60;
	iOutVar:INT;
	END_VAR	

Sample program		1 iOutVar:=MIN(iVar1,iVar2);
Operation result		

1-4-1-4. Limit value LIMIT

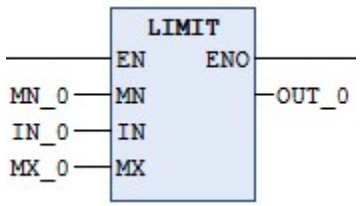
If the input data is between the upper and lower limit values, output the value. If it is below the lower limit value, output the lower limit value. If it is above the upper limit value, output the upper limit value.

Limit value [LIMIT]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Command	Command icon	Function
LIMIT		LIMIT

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
MN	Lower limit value	Any type	As the data type	0	Minimum value
IN	Input variable	Any type	As the data type	0	Input data
MX	Upper limit value	Any type	As the data type	0	Maximum value
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Output variable	Any type	As the data type	0	Output variable

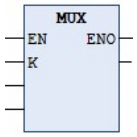
Limit value output. Determine if the input data is between the minimum and maximum values. If the input data is between the two, output it directly as output data. When the input data is greater than the maximum value, the maximum value is taken as the output value. When the input data is less than the minimum value, the minimum value is used as the output value.

DEMO	LD	ST
Define the variable		VAR MN_0 :INT:=20; IN_0 :INT:=90; MX_0 :INT:=60; OUT_0:INT; END_VAR
Sample program		1 OUT_0:=LIMIT(MN_0,IN_0,MX_0);
Operation result		

1-4-1-5. Multiple choice MUX

MUX selects the Kth value from a set of values. The first value is K=0. If K is greater than the number of other inputs (n), XS Studio passes the last value.

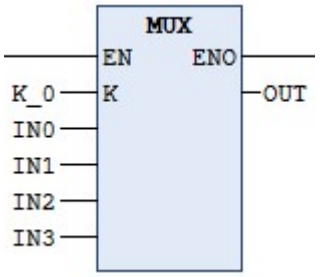
Multiple choice [MUX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Command	Command icon	Function
MUX		MUX

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
K	Lower limit value	BYTE, WORD, DWORD, LWORD, SINT, USINT, INT, UINT, DINT, LINT, ULINT, UDINT	As the data type	0	Control number
IN0	Input variable	Any type	As the data type	0	Input data 0
INn	Upper limit value	Any type	As the data type	0	Input data n
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Output variable	Any type	As the data type	0	Output data

MUX selects the Kth data output from the variable group.
When the value of K is less than 0 or greater than the number of input variables, output the last variable value.

DEMO	LD	ST
Define the variable		VAR K_0:INT:=2; IN0:INT:=10; IN1:INT:=11; IN2:INT:=20; IN3:INT:=16; OUT:INT; END_VAR

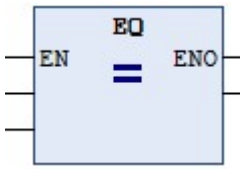
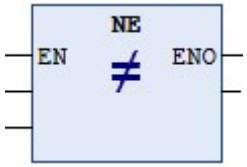
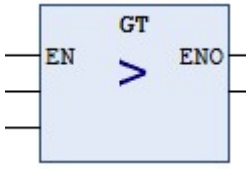
Sample program		1 OUT:=MUX(K_0,IN0,IN1,IN2,IN3)
Operation result	表达式 / 类型 / 值 K_0 INT 2 IN0 INT 10 IN1 INT 11 IN2 INT 20 IN3 INT 16 OUT INT 20	

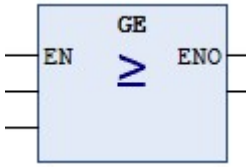
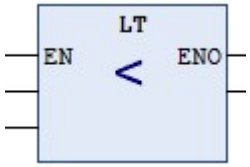
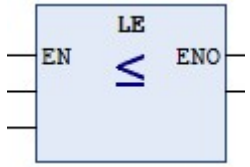
1-4-2. Compare Instructions

1-4-2-1. Basic compare instruction EQ/NE/GT/GE/LT/LE

Data size comparison instruction.

Basic compare instruction [EQ/NE/GT/GE/LT/LE]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Command name	Command icon	Function
EQ		=
NE		<>
GT		>

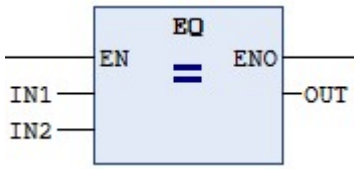
Command name	Command icon	Function
GE		>=
LT		<
LE		<=

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN0	Input variable	Any type	As the data type	0	Comparison value
IN1	Input variable	Any type	As the data type	0	Comparison value
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Compare result	BOOL	[FALSE,TRUE]	0	Output data

Basic comparison instructions comparison situation

Instruction	Compare	Result
EQ	IN0=IN1	TRUE
NE	IN0<>IN1	TRUE
GT	IN0>IN1	TRUE
GE	IN0>=IN1	TRUE
LT	IN0<IN1	TRUE
LE	IN0<=IN1	TRUE

DEMO	LD	ST
Define the variable		VAR IN1:INT:=10; IN2:INT:=10; OUT:BOOL; END_VAR

Sample program		1 OUT:=IN1=IN2;
Operation result	表达式 / 类型 / 值 IN1 / INT / 10 IN2 / INT / 10 OUT / BOOL / TRUE	

1-4-2-2. Maximum/minimum value retrieval of arrays AryMax/AryMin

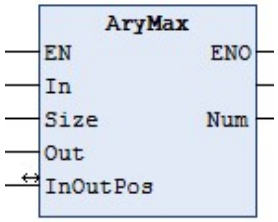
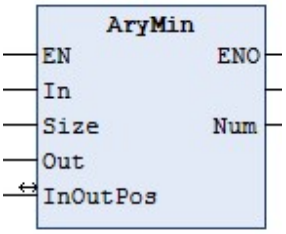
(1) Instruction overview

Retrieve the maximum/minimum values of a 1-dimensional array.

AryMax: Retrieves the maximum value of a 1-dimensional array element.

AryMin: Retrieve the minimum value of a 1-dimensional array element.

Maximum/minimum value retrieval of arrays [AryMax/AryMin]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryMax	Maximum value retrieval of arrays		Out:=AryMax(In:= , Size:= , Out:= , InOutPos:= , Num=>);
AryMin	Minimum value retrieval of arrays		Out:=AryMin(In:= , Size:= , Out:= , InOutPos:= , Num=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In[] array	Retrieve Object Array	USINT/UINT/UDINT /ULINT/SINT/INT/DINT/ LINT/REAL/LREAL/TIME/ DATE/TOD/DT/STRING	As the data type	(*)	Retrieve Object Array

Size	Retrieve the number of elements in an object array	UINT	As the data type	1	In [], the number of elements in the retrieval object array
Out	Search results	Consistent with In[] element data type	As the data type	(*)	Search results
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Num	Number of retrieved values	UINT	As the data type	-	Number of retrieved values
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOutPos	retrieve elements	UINT	As the data type	-	Array element numbers for retrieved values

(3) Function

- Retrieve "Size" array elements starting from In[] set in the retrieval object array In.
- Assign the search result to "Out", the offset of the search element from the input element position to "InOutPos", and the number of search values to "Num". When Num is greater than 1, the value of InOutPos becomes the lowest element number in the retrieved value.
- The judgment of the size relationship between values of data types other than integers and real numbers is shown in the table below:

Data type	Size relationship
TIME	The value with a larger value is judged as large
DATE, TOD, DT	The latter date or time is judged to be larger
STRING	Firstly, compare the character encoding of the first character of each string. If the character encoding is different, the size of the character encoding is the size of the string. If the character encoding of the first character is the same, continue comparing the second and third characters until a different character encoding is found. If the length of each string is different, a NULL character (16#00) will be added at the end of the shorter string based on the longer string, and then compared.

(4) Note

- The value of "Size" should not exceed the array area of In[].
- In[] must be defined as a one-dimensional array.
- Please ensure that "Out" and the elements of the retrieval object array In[] are of the same data type.
- Retrieve the element number "InOutPos" as the offset of the input element position.
- When In[] is a real number, the expected result may not be obtained due to the influence of errors.
- When the value of "Size" is 0, the value of "Num" becomes 0, the values of "InOutPos" and "Out" remain unchanged, and the function returns a value of True.
- When In[] is an unsupported data type, it cannot be executed properly.
- When In[] and "Out" are STRING types, the area size of In[] should not be greater than that of "Out". Otherwise, the function returns a value of FALSE and "Out" remains unchanged.
- When In[] and "Out" are STRING types, the valid range supports a maximum of 1985 characters.

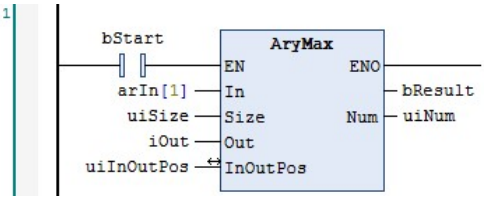
Otherwise, the return value returns FALSE.

- In[] requires integers, real numbers, and date types. Otherwise, the return value returns FALSE, while "InOutPos" and "Out" remain unchanged.

(5) Program example

DEMO	LD	ST
Define variables		
Sample program		
Operation result		

```
VAR
    bStart:BOOL;
    arIn:ARRAY[0..7]OF INT:=[99,3,6,20,5,11,4,66];
    uiSize    :UINT:=6;
    iOut      :INT;
    uiInOutPos:UINT;
    uiNum     :UINT;
    bResult   :BOOL;
END_VAR
```




```
IF bStart THEN
    bResult:=AryMax(In:= arIn[1],
        Size:= uiSize,
        Out:= iOut,
        InOutPos:= uiInOutPos,
        Num=> uiNum);
    bStart:=FALSE;
END_IF
```

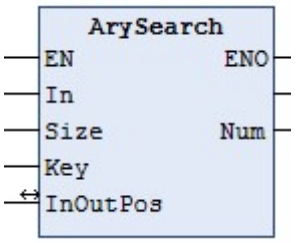
表达式	类型	值
bStart	BOOL	FALSE
arIn	ARRAY [0..7] OF INT	
arIn[0]	INT	99
arIn[1]	INT	3
arIn[2]	INT	6
arIn[3]	INT	20
arIn[4]	INT	5
arIn[5]	INT	11
arIn[6]	INT	4
arIn[7]	INT	66
uiSize	UINT	6
iOut	INT	20
uiInOutPos	UINT	2
uiNum	UINT	1
bResult	BOOL	TRUE

1-4-2-3. Array retrieval ArySearch

(1) Instruction overview

Retrieve the specified value in a one-dimensional array.

Array retrieval [ArySearch]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ArySearch	Array retrieval		Out:=ArySearch(In:= , Size:= , Key:= , InOutPos:= , Num=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In[] array	Retrieve Object Array	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/LINT/ REAL/LREAL/TIME/DATE/ TOD/DT/STRING	As the data type	(*)	Retrieve Object Array
Size	Retrieve the number of elements in an object array	UINT	As the data type	1	Retrieve the number of elements in an object array
Key	Search keywords	Consistent with In[] element data type	As the data type	(*)	Retrieve value
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Num	Number of retrieved values	UINT	As the data type	0	Number of retrieved values
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOutPos	Retrieve element numbers	UINT	As the data type	-	Retrieved value is the offset value of the input element position

(3) Function

- Retrieve elements with the same value as the retrieval keyword "Key" from the "Size" array elements starting from the In[] set in the 1-dimensional retrieval object array.

- The values for retrieving element number "InOutPos" and number of retrieved values "Num" are shown in the table below:

The existence of elements with the same value as "Key"	“InOutPos”	“Num”
Exist	The offset of the input position of the lowest element in an element with the same value as "Key"	Number of elements with the same value as "Key"
Not exist	Not change	0

- The judgment of the size relationship between values of data types other than integers and real numbers is shown in the table below:

Data type	Size relationship
TIME	The one with a larger value is judged as large
DATE	The latter one is judged as large
STRING	Firstly, compare the character encoding of the first character of each string. If the character encoding is different, the size of the character encoding is the size of the string. If the character encoding of the first character is the same, continue comparing the second and third characters until a different character encoding is found. If the length of each string is different, a NULL character (16#00) will be added at the end of the shorter string based on the longer string, and then compared.

(4) Note

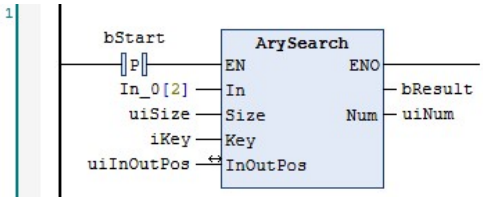
- The value of "Size" should not exceed the array area of In[].
- In[] must be defined as a one-dimensional array.
- Please ensure that "Key" and the elements of the retrieval object array In[] are of the same data type, otherwise the function returns a value of FALSE.
- Please ensure that the input parameter for "Key" is a variable.
- Retrieve the element number "InOutPos" as the offset of the input element position.
- When the value of "Size" is 0, the value of "Num" is initialized to 0, the value of "InOutPos" remains unchanged, and the function returns a value of True.
- When In[] and "Key" are STRING types, the valid range supports a maximum of 1985 characters. When it exceeds the valid range, the value returned by the function is FALSE, and the value of "Num" is initialized to 0.

(5) Program example

DEMO	LD	ST

Define variables	VAR
	bStart :BOOL;
	In_0 :ARRAY[0..9] OF INT:=[0,1,2,3,4,5,6,7,8,9];
	uiSize :UINT:=7;
	iKey :INT:=5;
	uiInOutPos:UINT;
	uiNum :UINT;
	bResult :BOOL;
	END_VAR	

Sample program

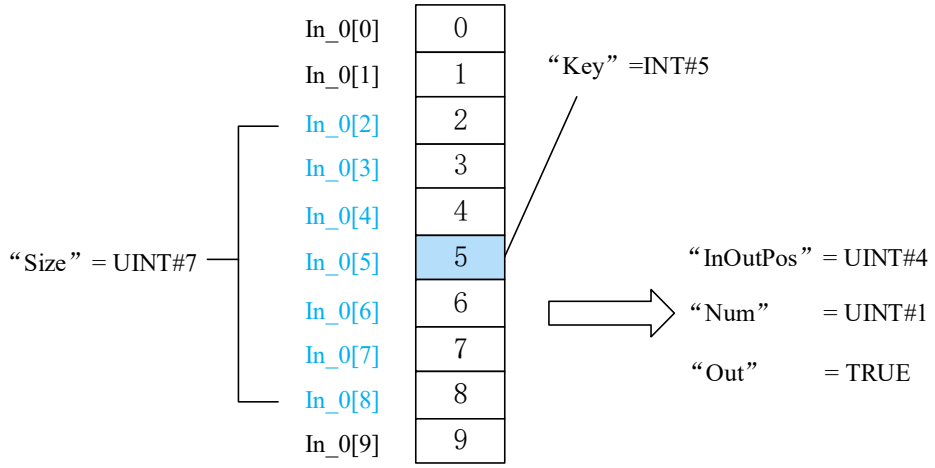



```
IF bStart THEN
    bResult:=ArySearch(In:= In_0[2],
        Size:= uiSize,
        Key:= iKey,
        InOutPos:= uiInOutPos,
        Num=> uiNum);
    bStart:=FALSE;
END_IF
```

Operation result

表达式	类型	值
bStart	BOOL	TRUE
In_0	ARRAY [0..9] OF INT	
In_0[0]	INT	0
In_0[1]	INT	1
In_0[2]	INT	2
In_0[3]	INT	3
In_0[4]	INT	4
In_0[5]	INT	5
In_0[6]	INT	6
In_0[7]	INT	7
In_0[8]	INT	8
In_0[9]	INT	9
uiSize	UINT	7
iKey	INT	5
uiInOutPos	UINT	3
uiNum	UINT	1
bResult	BOOL	TRUE

Principle analysis

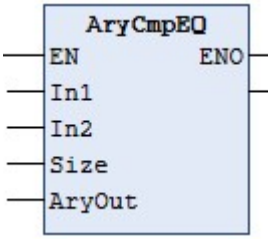


1-4-2-4. Batch comparison of arrays EQ AryCmpEQ

(1) Instruction overview

Compare whether the elements of two arrays are equal.

Batch comparison of arrays EQ [AryCmpEQ]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpEQ	Batch comparison of arrays EQ		bResult:=AryCmpEQ(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array1	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array1
In2[] array	Compare array2	An array with the same data type as In1[]	As the data type	-	Compare array2
Size	the compare elements numbers	UINT	As the data type	1	the compare elements numbers
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

(3) Function

- Compare the Size elements starting from In1[] and the Size elements starting from In2[] of two arrays to see if they are equal. The comparison results are saved to the corresponding element numbers of the consecutive Size elements starting from AryOut[].
- When In1[i]=In2[i], AryOut[i] is True, otherwise it is FALSE.

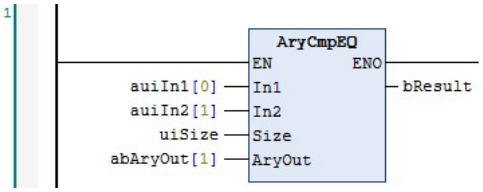
(4) Note

- The data types of In1[] and In2[] should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2[] are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[], In2[], or AryOut[]

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	auiIn2 :ARRAY[0..4] OF UINT:=[0,99,1,22,33];
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=4;
	bResult :BOOL;
	END_VAR	

Sample program		bResult:=AryCmpEQ(In1:= auiIn1[0], In2:= auiIn2[1], Size:= uiSize, AryOut:= abAryOut[1]);
Operation result		

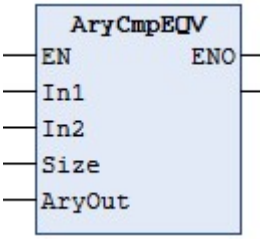
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
auiIn2	ARRAY [0..4] OF UINT	
auiIn2[0]	UINT	0
auiIn2[1]	UINT	99
auiIn2[2]	UINT	1
auiIn2[3]	UINT	22
auiIn2[4]	UINT	33
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	FALSE
abAryOut[2]	BOOL	TRUE
abAryOut[3]	BOOL	TRUE
abAryOut[4]	BOOL	FALSE
uiSize	UINT	4
bResult	BOOL	TRUE

1-4-2-5. Comparison of array elements EQV AryCmpEQV

(1) Instruction overview

Compare whether the elements of two arrays are equal.

Comparison of array elements EQV [AryCmpEQV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpEQV	Comparison of array elements EQV		bResult:=AryCmpEQV(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array
In2[] array	Compare variable	Same data type as In1[] element	As the data type	-	Compare variable
Size	Compare element numbers	UINT	As the data type	1	Compare element numbers
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

Note: In2 can also be defined as a variable.

(3) Function

- Compare the consecutive Size elements starting from array In1[] and variable "In2", whether they are equal (or compare the specified elements of array In1 and array In2). The comparison results are saved to the corresponding element numbers of the consecutive Size elements starting from AryOut[].
- When In1[i]=In2, AryOut[i] is True, otherwise it is FALSE.

(4) Note

- The data types of In1[] and In2 should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2 are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].

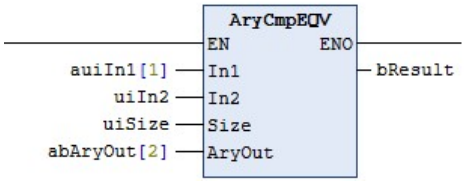
- The value of "Size" should not exceed any array area in In1[] or AryOut[].
- In1[] and In2[] need to use Boolean, bitstring, integer, or real types, otherwise the runtime function returns FALSE and AryOut[] remains unchanged.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    auiIn1  :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
    uiIn2   :UINT:=57;
    abAryOut:ARRAY[0..4] OF BOOL;
    uiSize  :UINT:=3;
    bResult :BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
1  bResult:=AryCmpEQV(In1:= auiIn1[1],
2       In2:= uiIn2,
3       Size:= uiSize,
4       AryOut:= abAryOut[2]);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
uiIn2	UINT	57
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	FALSE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	FALSE
abAryOut[4]	BOOL	TRUE
uiSize	UINT	3
bResult	BOOL	TRUE

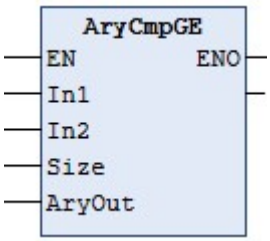
</td></tr>
</table>

1-4-2-6. Batch comparison of arrays GE AryCmpGE

(1) Instruction overview

Compare the size of each element of two arrays (greater than or equal to).

Batch comparison of arrays GE [AryCmpGE]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpGE	Batch comparison of arrays GE		bResult:=AryCmpGE(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array1	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/REAL/ LREAL	As the data type	-	Compare array1
In2[] array	Compare array2	An array with the same data type as In1[]	As the data type	-	Compare array2
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

(3) Function

- Compare the consecutive Size elements starting from In1[] and the consecutive Size elements starting from In2[] in two arrays. The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut[].
- When In1[i]>=In2[i], AryOut[i] is True, otherwise it is FALSE.

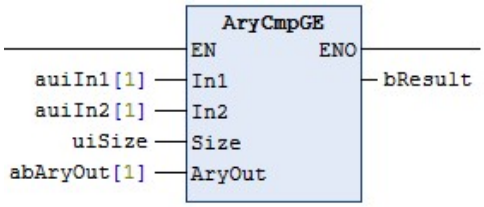
(4) Note

- The data types of In1[] and In2[] should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2[] are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[], In2[], or AryOut[].

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	auiIn2 :ARRAY[0..4] OF UINT:=[0,99,1,22,33];
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=4;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpGE(In1:= auiIn1[1], 2 In2:= auiIn2[1], 3 Size:= uiSize, 4 AryOut:= abAryOut[1]);
Operation result		

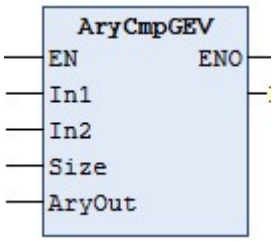
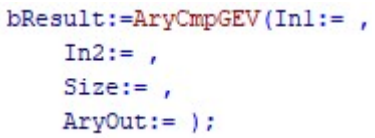
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
auiIn2	ARRAY [0..4] OF UINT	
auiIn2[0]	UINT	0
auiIn2[1]	UINT	99
auiIn2[2]	UINT	1
auiIn2[3]	UINT	22
auiIn2[4]	UINT	33
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	FALSE
abAryOut[2]	BOOL	TRUE
abAryOut[3]	BOOL	TRUE
abAryOut[4]	BOOL	TRUE
uiSize	UINT	4
bResult	BOOL	TRUE

1-4-2-7. Comparison of array elements GEV AryCmpGEV

(1) Instruction overview

Compare the size of each element and variable in the array (greater than or equal to).

Comparison of array elements GEV [AryCmpGEV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpGEV	Comparison of array elements GEV		


```
bResult:=AryCmpGEV(In1:= ,
    In2:= ,
    Size:= ,
    AryOut:= );
```

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array
In2[] array	Compare variable	Same data type as In1[] element	As the data type	-	Compare variable
Size	Compare element numbers	UINT	As the data type	1	Compare element numbers
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

Note: In2 can also be defined as a variable.

(3) Function

- Compare the consecutive Size elements starting from the two arrays In1[] and variable "In2" (greater than or equal to). The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result arrangement of AryOut[].
- When In1[i]>=In2, AryOut[i] is True, otherwise it is FALSE.

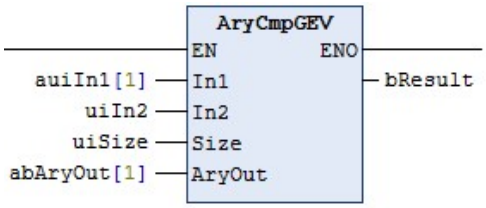
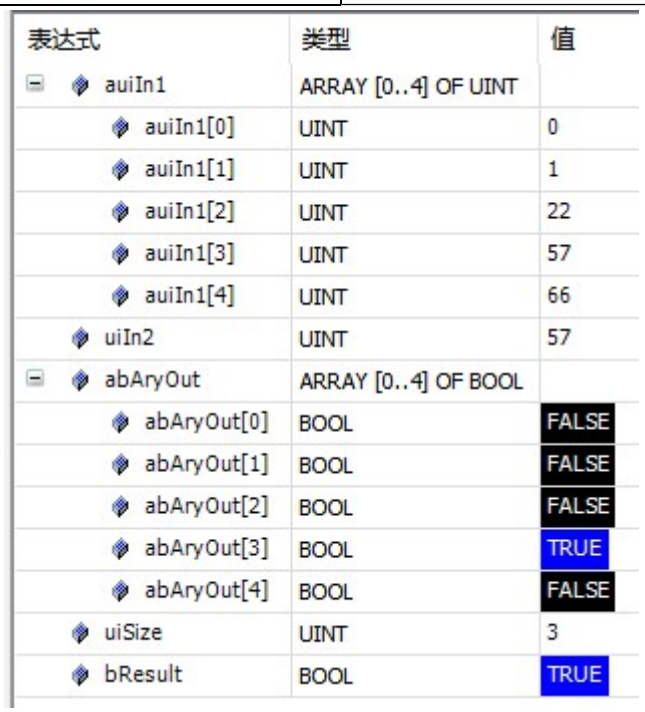
(4) Note

- The data types of In1[] and In2 should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2 are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[] or AryOut[].

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	uiIn2 :UINT:=57;
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=3;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpGEV(In1:= auiIn1[1], 2 In2:= uiIn2, 3 Size:= uiSize, 4 AryOut:= abAryOut[1]);
Operation result		

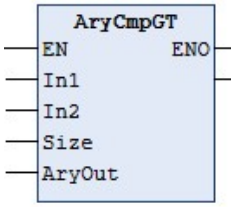
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
uiIn2	UINT	57
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	FALSE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	TRUE
abAryOut[4]	BOOL	FALSE
uiSize	UINT	3
bResult	BOOL	TRUE

1-4-2-8. Batch comparison of arrays GT AryCmpGT

(1) Instruction overview

Compare the size of each element of two arrays (greater than).

Batch comparison of arrays GT [AryCmpGT]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpGT	Batch comparison of arrays GT		bResult:=AryCmpGT(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array1	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array1
In2[] array	Compare array2	An array with the same data type as In1[]	As the data type	-	Compare array2
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

(3) Function

- Compare the consecutive Size elements of In1[] and In2[] (greater than). The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut[].
- When In1 [i]>In2 [i], AryOut[i] is True, otherwise it is FALSE.

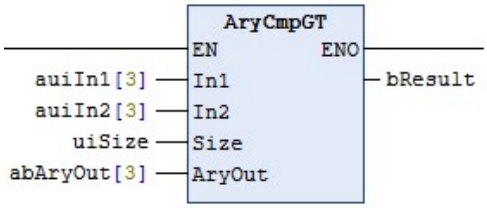
(4) Note

- The data types of In1[] and In2[] should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2[] are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[], In2[], or AryOut[].

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	auiIn2 :ARRAY[0..4] OF UINT:=[0,99,1,22,33];
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=2;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpGT(In1:= auiIn1[3], 2 In2:= auiIn2[3], 3 Size:= uiSize, 4 AryOut:= abAryOut[3]);
Operation result		

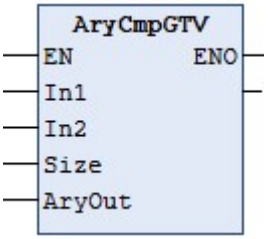
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
auiIn2	ARRAY [0..4] OF UINT	
auiIn2[0]	UINT	0
auiIn2[1]	UINT	99
auiIn2[2]	UINT	1
auiIn2[3]	UINT	22
auiIn2[4]	UINT	33
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	FALSE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	TRUE
abAryOut[4]	BOOL	TRUE
uiSize	UINT	2
bResult	BOOL	TRUE

1-4-2-9. Comparison of array elements GTV AryCmpGTV

(1) Instruction overview

Compare the size of each element and variable in the array (greater than).

Comparison of array elements GTV [AryCmpGTV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpGTV	Comparison of array elements GTV		bResult:=AryCmpGTV(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array1	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array
In2[] array	Compare array2	Same data type as In1[] element	As the data type	-	Compare variable
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

(3) Function

- Compare the consecutive Size elements starting from In1[] and variable "In2"(greater than). The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut[].
- When In1 [i]>In2, AryOut [i] is True, otherwise it is FALSE.

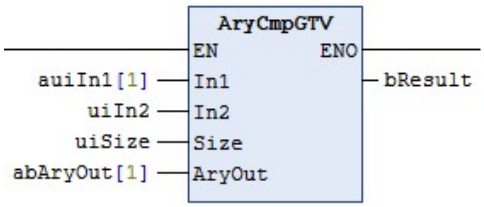
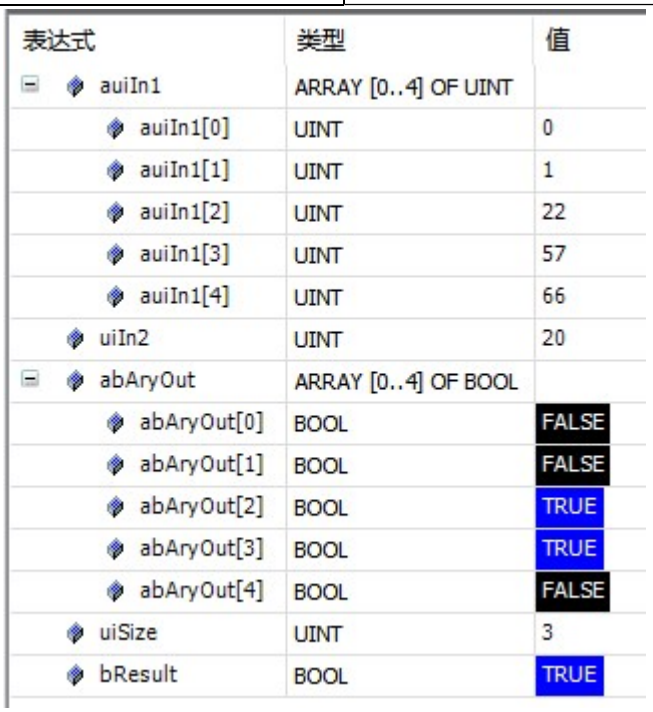
(4) Note

- The data types of In1[] and In2 should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2 are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[] or AryOut[].

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	uiIn2 :UINT:=20;
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=3;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpGTV(In1:= auiIn1[1], 2 In2:= uiIn2, 3 Size:= uiSize, 4 AryOut:= abAryOut[1]);
Operation result		

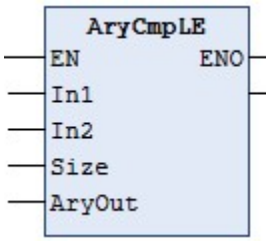
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
uiIn2	UINT	20
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	FALSE
abAryOut[2]	BOOL	TRUE
abAryOut[3]	BOOL	TRUE
abAryOut[4]	BOOL	FALSE
uiSize	UINT	3
bResult	BOOL	TRUE

1-4-2-10. Batch comparison of arrays LE AryCmpLE

(1) Instruction overview

Compare the size of each element of two arrays (less than or equal to).

Batch comparison of arrays LE [AryCmpLE]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpLE	Batch comparison of arrays LE		bResult:=AryCmpLE(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array1	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array1
In2[] array	Compare array2	An array with the same data type as In1[]	As the data type	-	Compare array2
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

(3) Function

- Compare the continuous Size elements starting from In1[] and the continuous Size elements starting from In2[] (less than or equal to). The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut [].
- When In1[i]<=In2[i], AryOut[i] is True, otherwise it is FALSE.

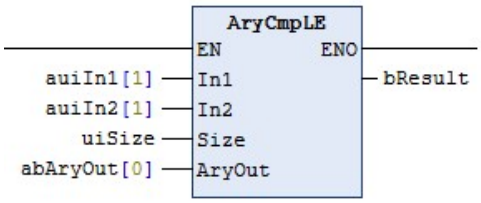
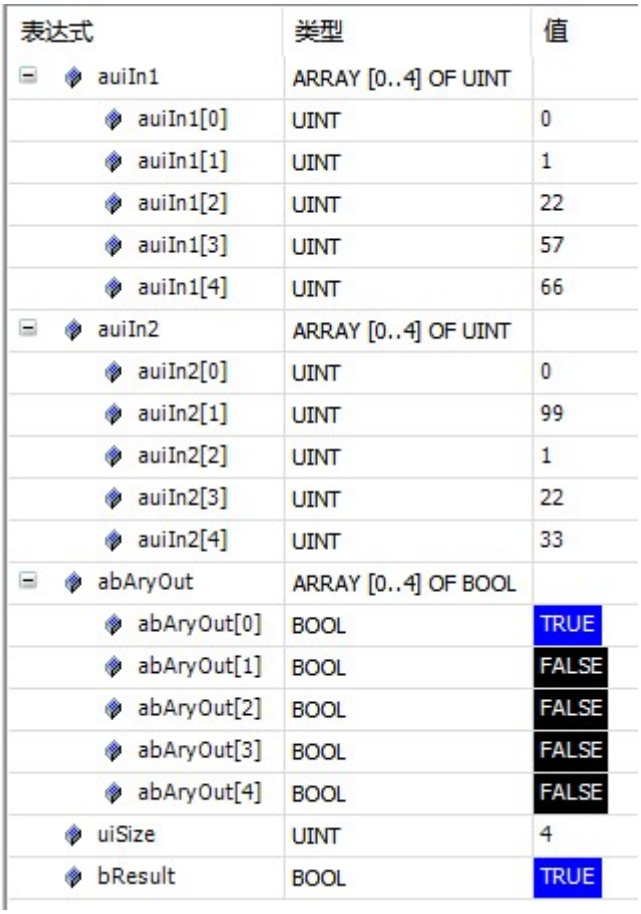
(4) Note

- The data types of In1[] and In2[] should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2[] are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[], In2[], or AryOut[].

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	auiIn2 :ARRAY[0..4] OF UINT:=[0,99,1,22,33];
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=4;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpLE(In1:= auiIn1[1], 2 In2:= auiIn2[1], 3 Size:= uiSize, 4 AryOut:= abAryOut[0]);
Operation result		

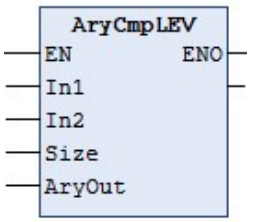
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
auiIn2	ARRAY [0..4] OF UINT	
auiIn2[0]	UINT	0
auiIn2[1]	UINT	99
auiIn2[2]	UINT	1
auiIn2[3]	UINT	22
auiIn2[4]	UINT	33
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	TRUE
abAryOut[1]	BOOL	FALSE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	FALSE
abAryOut[4]	BOOL	FALSE
uiSize	UINT	4
bResult	BOOL	TRUE

1-4-2-11. Comparison of array elements LEV AryCmpLEV

(1) Instruction overview

Compare the size of each element and variable in an array (less than or equal to).

Comparison of array elements LEV [AryCmpLEV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpLEV	Comparison of array elements LEV		bResult:=AryCmpLEV(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array
In2[] array	Compare variable	Same data type as In1[] element	As the data type	-	Compare variable
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

Note: In2 can also be defined as a variable.

(3) Function

- Compare the consecutive Size elements starting from the comparison array In1[] and variable "In2" (less than or equal to). The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut[].
- When In1[i]<=In2, AryOut[i] is True, otherwise it is FALSE.

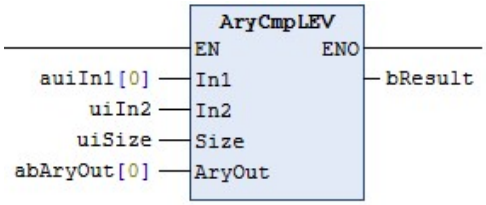
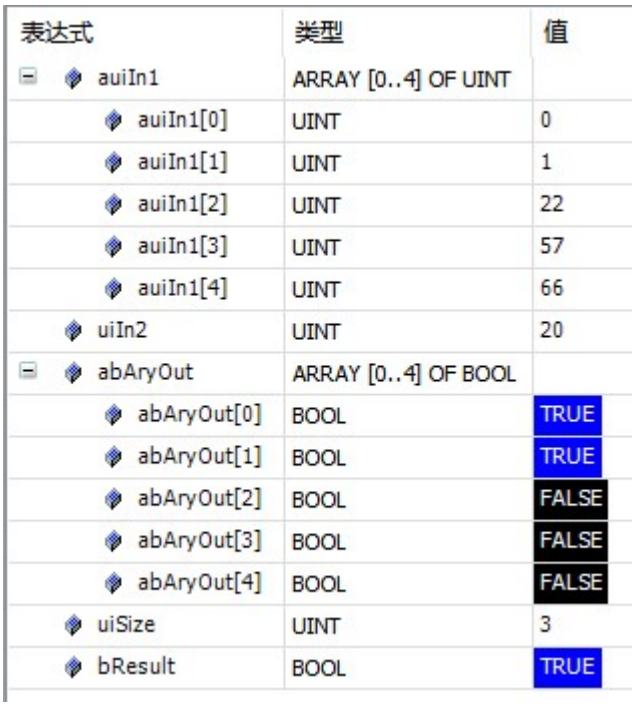
(4) Note

- The data types of In1[] and In2 should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2 are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[] or AryOut[].

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	uiIn2 :UINT:=20;
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=3;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpLEV(In1:= auiIn1[0], 2 In2:= uiIn2, 3 Size:= uiSize, 4 AryOut:= abAryOut[0]);
Operation result		

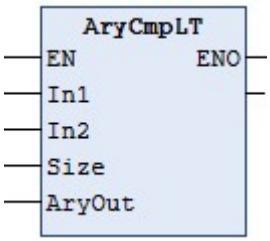
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
uiIn2	UINT	20
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	TRUE
abAryOut[1]	BOOL	TRUE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	FALSE
abAryOut[4]	BOOL	FALSE
uiSize	UINT	3
bResult	BOOL	TRUE

1-4-2-12. Batch comparison of arrays LT AryCmpLT

(1) Instruction overview

Compare the size of each element of two arrays (less than).

Batch comparison of arrays LT [AryCmpLT]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpLT	Batch comparison of arrays LT		bResult:=AryCmpLT(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array1	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array1
In2[] array	Compare array2	An array with the same data type as In1[]	As the data type	-	Compare array2
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

(3) Function

- Compare the consecutive Size elements starting from In1[] and consecutive Size elements starting from In2[](less than). The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut[].
- When In1[i]<In2[i], AryOut[i] is True, otherwise it is FALSE.

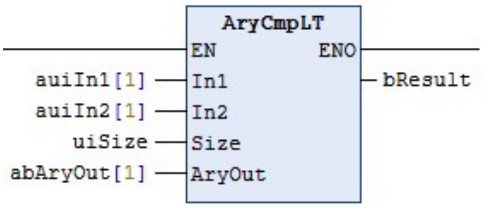
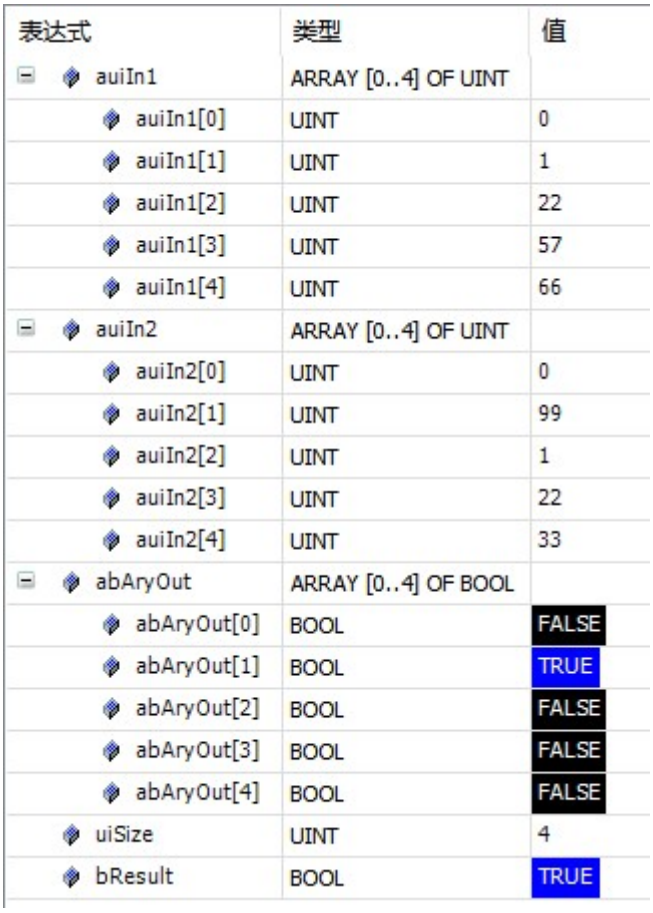
(4) Note

- The data types of In1[] and In2[] should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2[] are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[], In2[], or AryOut[].

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	auiIn2 :ARRAY[0..4] OF UINT:=[0,99,1,22,33];
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=4;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpLT(In1:= auiIn1[1], 2 In2:= auiIn2[1], 3 Size:= uiSize, 4 AryOut:= abAryOut[1]);
Operation result		

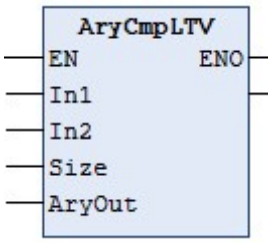
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
auiIn2	ARRAY [0..4] OF UINT	
auiIn2[0]	UINT	0
auiIn2[1]	UINT	99
auiIn2[2]	UINT	1
auiIn2[3]	UINT	22
auiIn2[4]	UINT	33
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	TRUE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	FALSE
abAryOut[4]	BOOL	FALSE
uiSize	UINT	4
bResult	BOOL	TRUE

1-4-2-13. Comparison of array elements LTV AryCmpLTV

(1) Instruction overview

Compare the size of each element and variable in the array (less than).

Comparison of array elements LTV [AryCmpLTV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpLTV	Comparison of array elements LT		bResult:=AryCmpLTV(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Compare array
In2[] array	Compare variable	Same data type as In1[] element	As the data type	-	Compare variable
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

Note: In2 can also be defined as a variable.

(3) Function

- Compare the consecutive Size elements starting from array In1[0] and the comparison variable "In2" (less than). The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut[].
- When In1[i]<In2, AryOut[i] is True, otherwise it is FALSE.

(4) Note

- The data types of In1[] and In2 should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2 are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].

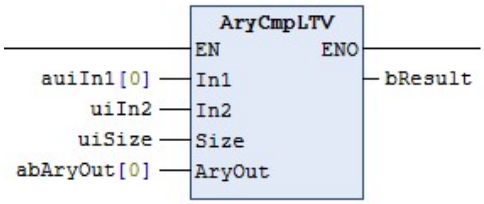
- The value of "Size" should not exceed any array area in In1[] or AryOut[].

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    auiIn1  :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
    uiIn2   :UINT:=20;
    abAryOut:ARRAY[0..4] OF BOOL;
    uiSize  :UINT:=3;
    bResult :BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
1  bResult:=AryCmpLTV(In1:= auiIn1[0],
2      In2:= uiIn2,
3      Size:= uiSize,
4      AryOut:= abAryOut[0]);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
uiIn2	UINT	20
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	TRUE
abAryOut[1]	BOOL	TRUE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	FALSE
abAryOut[4]	BOOL	FALSE
uiSize	UINT	3
bResult	BOOL	TRUE

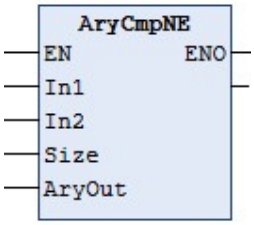
</td></tr>
</table>

1-4-2-14. Batch comparison of arrays NE AryCmpNE

(1) Instruction overview

Compare whether the elements of two arrays are different.

Batch comparison of arrays NE [AryCmpNE]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpNE	Batch comparison of arrays NE		bResult:=AryCmpNE(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array1	BOOL/BYTE/WORD/ DWORD/LWORD/USINT/ UINT/UDINT/ULINT/SINT/ INT/DINT/LINT/REAL/LREAL	As the data type	-	Compare array1
In2[] array	Compare array2	An array with the same data type as In1[]	As the data type	-	Compare array2
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

(3) Function

- Compare the consecutive Size elements starting from array In1[] and consecutive Size elements starting from array In2[]. The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut[].
- When In1[i]<>In2[i], AryOut[i] is True, otherwise it is FALSE.

(4) Note

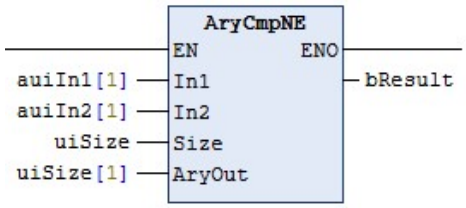
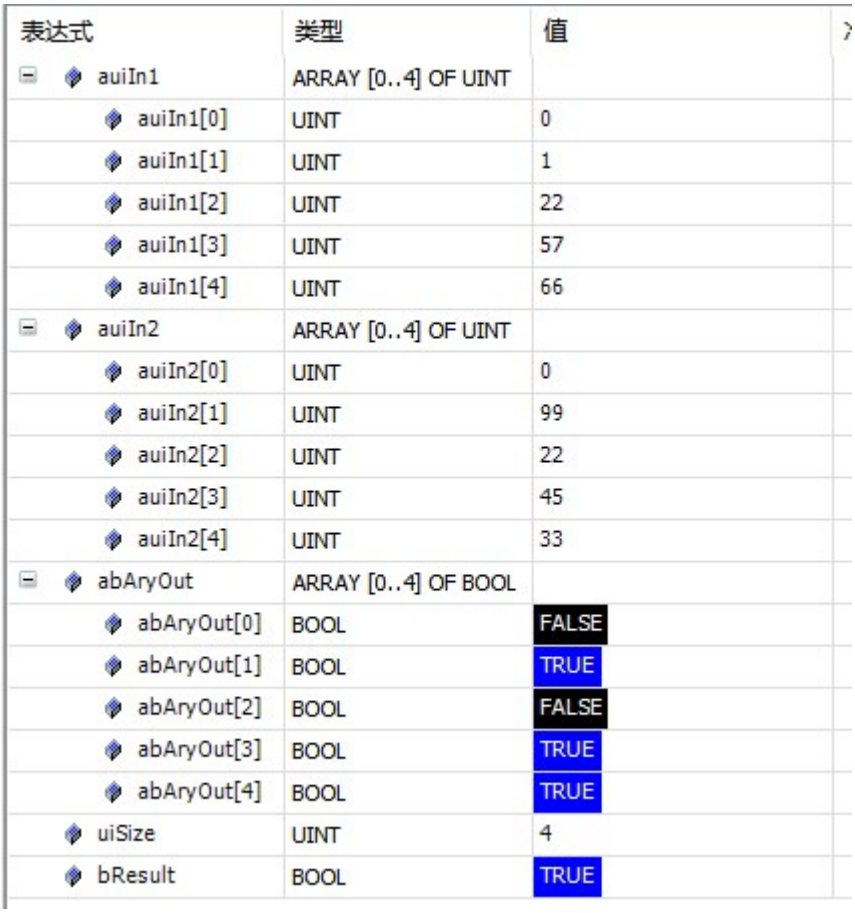
- The data types of In1[] and In2[] should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- In1[] and In2[] need to use Boolean, bitstring, integer, or real types, otherwise the return value returns FALSE, and AryOut[] remains unchanged.
- When In1[] and In2[] are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.

- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The value of "Size" should not exceed any array area in In1[], In2[], or AryOut[].

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,22,57,66];
	auiIn2 :ARRAY[0..4] OF UINT:=[0,99,22,45,33];
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=4;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpNE(In1:= auiIn1[1], 2 In2:= auiIn2[1], 3 Size:= uiSize, 4 AryOut:= abAryOut[1]);
Operation result		

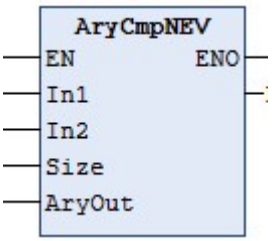
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	22
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
auiIn2	ARRAY [0..4] OF UINT	
auiIn2[0]	UINT	0
auiIn2[1]	UINT	99
auiIn2[2]	UINT	22
auiIn2[3]	UINT	45
auiIn2[4]	UINT	33
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	TRUE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	TRUE
abAryOut[4]	BOOL	TRUE
uiSize	UINT	4
bResult	BOOL	TRUE

1-4-2-15. Comparison of array elements NEV AryCmpNEV

(1) Instruction overview

Compare whether the elements of two arrays are different.

Comparison of array elements NEV [AryCmpNEV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryCmpNEV	Comparison of array elements NEV		AryCmpNEV (In1 :=, In2 :=, Size := , AryOut :=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1[] array	Compare array	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/LINT/ REAL/LREAL	As the data type	-	Compare array
In2[] array	Compare variable	Same data type as In1[] element	As the data type	-	Compare variable
Size	Compare element number	UINT	As the data type	1	Compare element number
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Comparison result array	BOOL	As the data type	-	Comparison result array

Note: In2 can also be defined as a variable.

(3) Function

- Compare the continuous Size elements of compare array In1[] and comparison variable "In2" to see if they are different. The comparison results are saved to the corresponding element numbers for the consecutive Size elements starting from the comparison result array AryOut [].
- When In1[i]<>In2, AryOut[i] is True, otherwise it is FALSE.

(4) Note

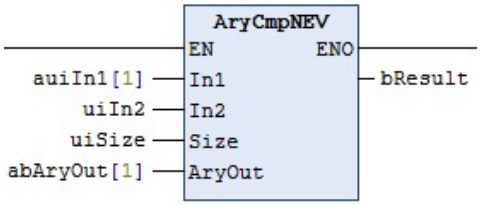
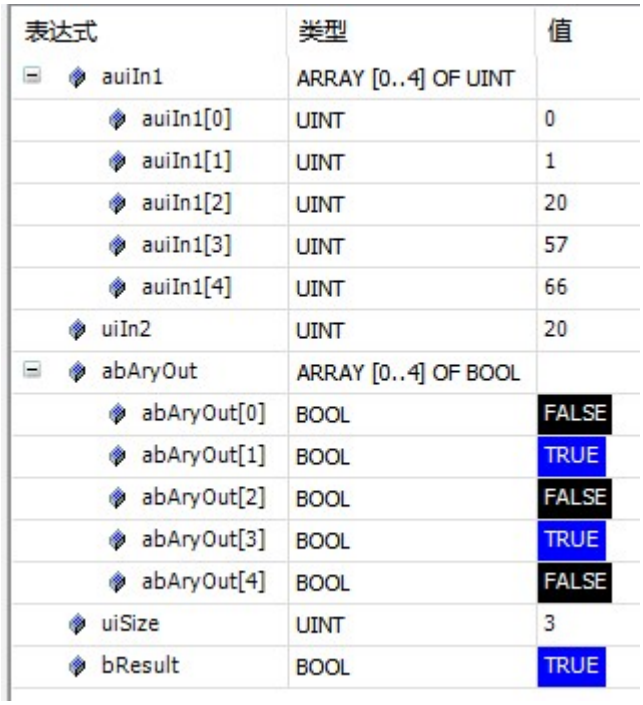
- The data types of In1[] and In2 should be consistent, otherwise the return value will return FALSE and AryOut[] will remain unchanged.
- When In1[] and In2 are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].

- The value of "Size" should not exceed any array area in In1[] or AryOut[].
- In1[] and In2[] need to use Boolean, bitstring, integer, or real types, otherwise the return value returns FALSE, and AryOut[] remains unchanged.

(5) Program example

DEMO	LD	ST

Define variables	VAR
	auiIn1 :ARRAY[0..4] OF UINT:=[0,1,20,57,66];
	uiIn2 :UINT:=20;
	abAryOut:ARRAY[0..4] OF BOOL;
	uiSize :UINT:=3;
	bResult :BOOL;
	END_VAR	

Sample program		1 bResult:=AryCmpNEV(In1:= auiIn1[1], 2 In2:= uiIn2, 3 Size:= uiSize, 4 AryOut:= abAryOut[1]);
Operation result		

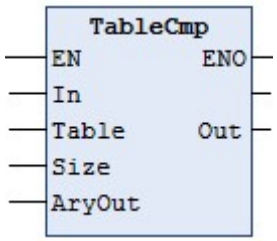
表达式	类型	值
auiIn1	ARRAY [0..4] OF UINT	
auiIn1[0]	UINT	0
auiIn1[1]	UINT	1
auiIn1[2]	UINT	20
auiIn1[3]	UINT	57
auiIn1[4]	UINT	66
uiIn2	UINT	20
abAryOut	ARRAY [0..4] OF BOOL	
abAryOut[0]	BOOL	FALSE
abAryOut[1]	BOOL	TRUE
abAryOut[2]	BOOL	FALSE
abAryOut[3]	BOOL	TRUE
abAryOut[4]	BOOL	FALSE
uiSize	UINT	3
bResult	BOOL	TRUE

1-4-2-16. Table comparison TableCmp

(1) Instruction overview

Compare multiple defined intervals specified in the comparison data and comparison table.

Table comparison [TableCmp]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
TableCmp	Table comparison		bResult:=TableCmp(In:= , Table:= , Size:= , AryOut:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Compare data	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL	As the data type	-	Value to be compared
Table[] 2D array	Compare table	A 2D array with elements of the same data type as In	As the data type	-	A 2D array with defined intervals as elements
Size	Compare size	UINT	As the data type	1	The number of elements in Table[] compared to "In"
AryOut[] array	Individual comparison result array	BOOL	[FALSE,TRUE]	FALSE	Each elements comparison results in Table[] True: Consistent FALSE: Inconsistent
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Return value	BOOL	[FALSE,TRUE]	FALSE	Comparison result

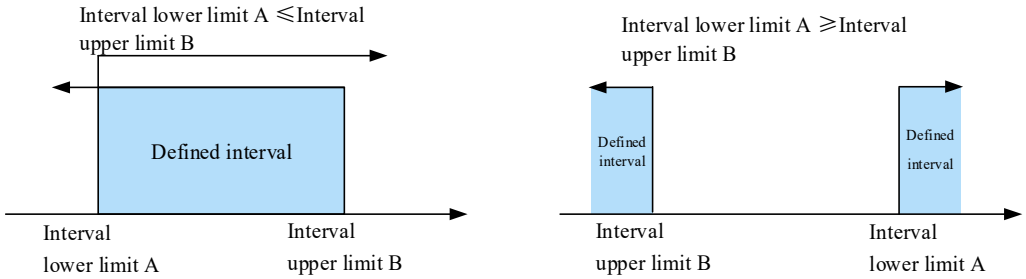
(3) Function

- Compare the defined group “Size” interval in the comparison data "In" and the comparison table[].
- Table[] is a 2-dimensional array, where the 1-dimensional element is the number of the defined interval, the 0th element in the 2-dimensional array represents the lower limit value A of the defined interval, and the 1st element represents the upper limit value B of the defined interval.

Defined intervals	Interval lower limit value A	Interval upper limit value B
Interval 0	Table[0,0]	Table[0,1]
Interval 1	Table[1,0]	Table[1,1]

...	...	...
Interval “Size”-1	Table[“Size”-1,0]	Table[“Size”-1,1]

- The method of specifying the defined interval using the lower limit value A and the upper limit value B of the interval is as follows. The values of the lower limit value A and the upper limit value B of the interval are included in the defined interval.



- The comparison results of "In" and Table[] are saved in the individual comparison result array AryOut[]. If "In" is within the i-th defined interval, then the value of AryOut[i] is True, otherwise it is FALSE. If all the "Size" elements of AryOut[] are true, then the value of the comparison result "Out" is true, otherwise it is false.

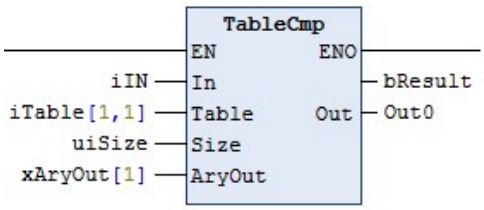
(4) Note

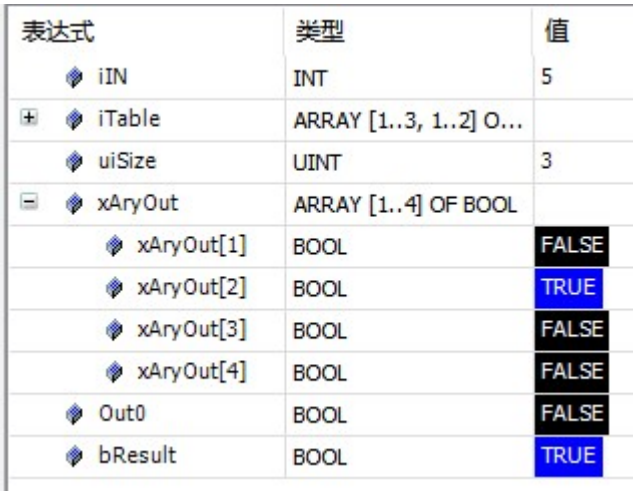
- The data types of "In" and Table[] elements should be consistent, otherwise the function return value will be FALSE.
- Please make sure to use a 2-dimensional array for Table[] and the size of the 2nd dimensional array is 2, otherwise the output result will be unexpected.
- When the size of the AryOut[] array is greater than "Size", the comparison results will be saved to the corresponding element numbers of the consecutive Size at the beginning of AryOut[]. Other array elements remain unchanged.
- When comparing real numbers and including division results that cannot be fully divided, the processing results may differ from the expected ones due to rounding errors.
- When the value of "Size" is 0, the value of "Out" is FALSE, AryOut[] remains unchanged, and the function returns a value of True.
- When the value of "Size" exceeds the size of the AryOut[] array or the first dimensional array of Table[], it may cause program execution exceptions and even PLC crashes.

(5) Program example

DEMO	LD	ST

Define variables	VAR
	iIN :INT:=5;
	iTable :ARRAY[1..3,1..2] OF INT:=[0,1,4,10,10,20];
	uiSize :UINT:=3;
	xAryOut:ARRAY[1..4] OF BOOL;
	Out0 :BOOL;
	bResult:BOOL;
	END_VAR	

Sample program		1 bResult:=TableCmp(In:= iIN, 2 Table:= iTable[1,1], 3 Size:= uiSize, 4 AryOut:= xAryOut[1], 5 Out=> Out0);

Operation result	

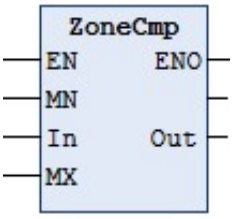
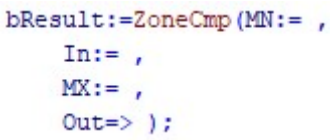
表达式	类型	值
iIN	INT	5
iTable	ARRAY [1..3, 1..2] O...	
uiSize	UINT	3
xAryOut	ARRAY [1..4] OF BOOL	
xAryOut[1]	BOOL	FALSE
xAryOut[2]	BOOL	TRUE
xAryOut[3]	BOOL	FALSE
xAryOut[4]	BOOL	FALSE
Out0	BOOL	FALSE
bResult	BOOL	TRUE

1-4-2-17. Zone comparison ZoneCmp

(1) Instruction overview

Determine whether the comparison data is between the specified upper and lower limits.

Zone comparison [ZoneCmp]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ZoneCmp	Zone comparison		


```
bResult:=ZoneCmp(MN:= ,
    In:= ,
    MX:= ,
    Out=> );
```

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
MN	Lower limit value	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/REAL/ LREAL/TIME/DATE/TOD/DT	As the data type	-	Lower limit value
In	Compare data	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/REAL/ LREAL/TIME/DATE/TOD/DT	As the data type	-	Value to be compared
MX	Upper limit value	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/REAL/ LREAL/TIME/DATE/TOD/DT	As the data type	1	Upper limit value
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Out	Comparison result	BOOL	[FALSE,TRUE]	FALSE	Comparison result

(3) Function

- Determine whether the comparison data "In" is between the upper limit value "MX" and the lower limit value "MN".
- When "MX"≥"In"≥"MN", the value of "Out" is true, while others are false.
- The determination of the size relationship between values other than integers and real numbers is as follows:

Data type	Size relationship
TIME	The value with a larger value is judged as large
DATA, TOD, DT	The date or time later is judged as large

(4) Note

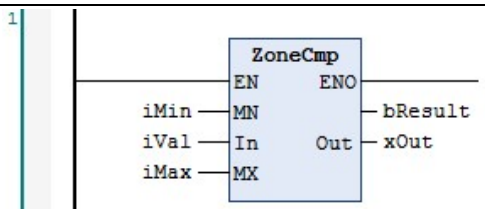
- When the data types of "In", "MX", and "MN" are different, the return value returns FALSE and Out remains unchanged.

- When "In", "MX", and "MN" are real numbers and include inexhaustible division results, the processing results may differ from the expected ones due to rounding errors.
- Determine whether in the range of +∞ or -∞ are equal.
- When the value of "MN" is greater than the value of "MX", the program encounters an exception and the function returns a value of FALSE.
- "MX" and "MN" should not be set to non numerical values, otherwise the program will encounter an exception and the function will return a value of FALSE.
- "In", "MX", and "MN" need to use integers, real numbers, and dates. Otherwise, the return value will return FALSE and Out will remain unchanged.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    iMin:INT:=10;
    iVal:INT:=20;
    iMax:INT:=30;
    xOut:BOOL;
    bResult:BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>


</td><td>

```
1  bResult:=ZoneCmp(MN:= iMin,
2       In:= iVal,
3       MX:= iMax,
4       Out=> xOut);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
iMin	INT	10
iVal	INT	20
iMax	INT	30
xOut	BOOL	TRUE
bResult	BOOL	TRUE

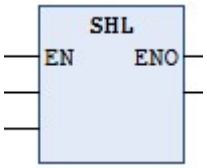
</td></tr>
</table>

1-4-3. Shift instruction

1-4-3-1. Shift left SHL

Shift the operand bit by bit to the left, leaving the left out bit unprocessed, and automatically filling the right empty bit with 0.

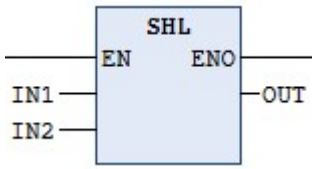
Shift left [SHL]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Command name	Command icon	Function
SHL		SHL

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN1	Input data	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT	As the data type	0	Input data
IN2	Shift bits	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT	As the data type	0	Shift bits
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Data after shifting	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT	As the data type	0	Data after shifting

The instruction can shift the data in input IN1 to the left by IN2 bits, and assign the output result to OUT. Shifting the binary number to the left by one bit is equivalent to multiplying the original number by 2. If the value of IN2 is greater than the width of the data type in IN1, the value of IN1 will be filled in as 0.

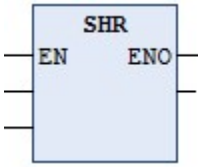
DEMO	LD	ST
Define variables		VAR IN1:WORD:=10; IN2:WORD:=2; OUT:WORD; END_VAR

DEMO	LD	ST
Sample program		1 OUT:=SHL(IN1,IN2);
Operation result		

1-4-3-2. Shift right SHR

Shift the operand bit by bit to the right, and do not process the right out bit.

Shift right [SHR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Command name	Command icon	Function
SHR		SHR

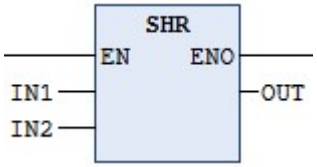
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN1	Input data	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT	As the data type	0	Input data
IN2	Shift bits	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT	As the data type	0	Shift bits
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Data after shifting	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT	As the data type	0	Data after shifting

The instruction can shift the data in input IN1 to the right by IN2 bits, assign the output result to OUT, and shift the binary number to the right by one bit is equivalent to dividing the original number by 2. If the IN2 value is greater than the width of the data type in IN1, the IN1 value will be filled with 0.
After the unsigned data type is shifted to the right, since there is no sign bit, the high bit can be filled with 0. Encountering signed integer data, the high bit needs to be shifted to the right by filling in the sign bit.

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    IN1:INT:=16#F100;
    IN2:WORD:=2;
    OUT:INT;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>

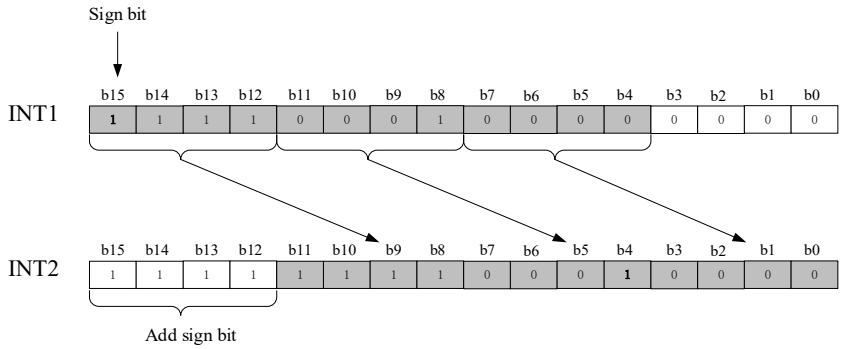


</td><td>

```
1   OUT:=SHR(IN1,IN2);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

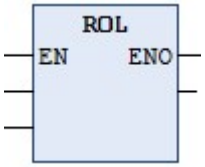
表达式	类型	值
IN1	INT	2#1111000100000000
IN2	WORD	2#0000000000000010
OUT	INT	2#1111110001000000

</td></tr>
<tr><td>Result analysis</td><td colspan="2">Due to INT is signed bit data with valid values ranging from -32768 to 32767, IN1 is hexadecimal signed data F100, and the highest bit b15 is the sign bit, after being moved 4 bits to the right, data needs to be added. As the source data sign bit is 1, the high 4 bits are added with four 1 (i.e. sign bits). Therefore, the final result of program operation is 16#FF10. The specific shift process is shown in the following figure:

</td></tr>
</table>

1-4-3-3. Rotate left ROL

Rotate the operand in a bitwise loop to the left, and the bits removed from the left are directly added to the lowest bit on the right.

Rotate left [ROL]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

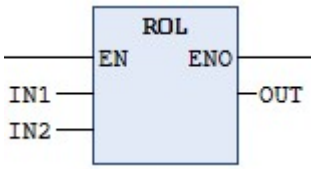
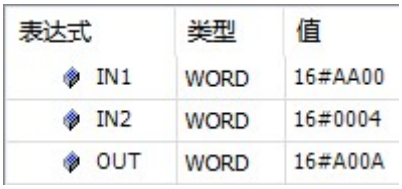
Command name	Command icon	Function
ROL		ROL

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN1	Input data	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	0	Input data
IN2	Shift bits	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	0	Shift bits
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Data after shifting	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	0	Data after shifting

This instruction can be used to cycle all the contents of input IN1 to the left bit by bit, and the empty bits are filled with the signal status of the shifted bits. The input parameter IN2 provides a numerical value representing the number of bits to rotate, and OUT is the result of the rotation operation.
After the unsigned data type is shifted to the right, since there is no sign bit, the high bit can be filled with 0. Encountering signed integer data, the high bit needs to be shifted to the right by filling in the sign bit.

DEMO	LD	ST
Define variables		

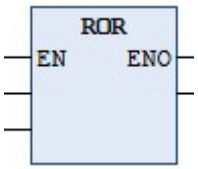
```
VAR
    IN1:WORD:=16#AA00;
    IN2:WORD:=4;
    OUT:WORD;
END_VAR
```

Sample program		1 OUT:=ROL(IN1,IN2);
Operation result		
Result analysis	The use of the output OUT after a right shift loop does not fill in the empty bit by 0, but rather fills in the 1010 from b12 to b15 in the input data IN1 to bits b0 to b3.	

1-4-3-4. Rotate right ROR

Rotate the operands bit by bit to the right, and the bits removed from the right are directly added to the highest position on the left.

Rotate right [ROR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

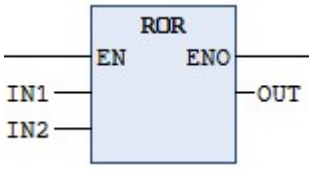
Command name	Command icon	Function
ROR		ROR

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
IN1	Input data	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	0	Input data
IN2	Shift bits	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	0	Shift bits
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
OUT	Data after shifting	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SINT /INT/DINT/LINT	As the data type	0	Data after shifting

This instruction can be used to cycle all the contents of input IN1 to the left bit by bit, and the empty bits are filled with the signal status of the shifted bits. The input parameter IN2 provides a numerical value representing the number of bits to rotate, and OUT is the result of the rotation operation.
After the unsigned data type is shifted to the right, since there is no sign bit, the high bit can be filled with 0. Encountering signed integer data, the high bit needs to be shifted to the right by filling in the sign bit.

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    IN1:WORD:=16#AA00;
    IN2:WORD:=4;
    OUT:WORD;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td> 
</td><td>

```
1   OUT:=ROR(IN1,IN2);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN1	WORD	16#AA00
IN2	WORD	16#0004
OUT	WORD	16#0AA0

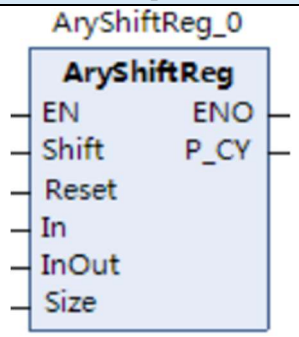
</td></tr>
</table>

1-4-3-5. Shift register AryShiftReg

(1) Instruction overview

Shift the entire array elements by 1 bit to the left, and insert the input value in the lowest position.

Shift register [AryShiftReg]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryShiftReg	shift register		AryShiftReg_0(Shift:= , Reset:= , In:= , InOut:= , Size:= , P_CY=> , ENO=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Shift	Shift	BOOL	[FALSE,TRUE]	FALSE	When changing from FALSE to True, perform a shift
Reset	Reset	BOOL	[FALSE,TRUE]	FALSE	When changing from FALSE to True, perform a reset
In	Input value	BOOL	[FALSE,TRUE]	FALSE	The value inserted in the lowest position of InOut[]
Size	The number of elements in a positional array	UINT	As the data type	0	The number of elements used in the shift register of InOut[]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
P_CY	carry flag	BOOL	As the data type	-	The value saved in the carry flag
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOut[] array	positional array	BOOL/BYTE/WORD /DWORD/LWORD	As the data type	-	positional array

(3) Function

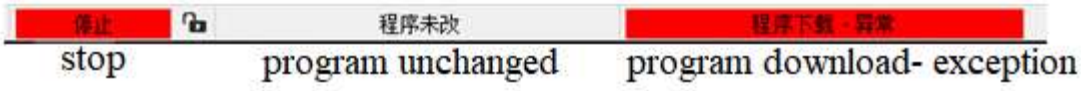
- When "Shift" changes from FALSE to True, shift the "Size" array elements starting from the InOut[] set on

pin InOut to the left (high bit direction) by 1 bit.

- Insert the input value "In" in the lowest position. The highest bit overflowing from the array will be output to the carry (CY) flag "P_CY".
- When "Reset" is true, set all bits of "Size" elements starting from the InOut[] set on pin InOut and carry (CY) flags to FALSE.

(4) Note

- When "Reset" is true, the array InOut[] cannot be assigned a value, and even if "Shift" changes from FALSE to true, no shift is performed.
- When the value of "Size" is 0, InOut[] remains unchanged, and the function returns a value of True.
- When the value of "Size" exceeds the array area of InOut[], the PLC will stop prompting an exception, as shown in the following figure:

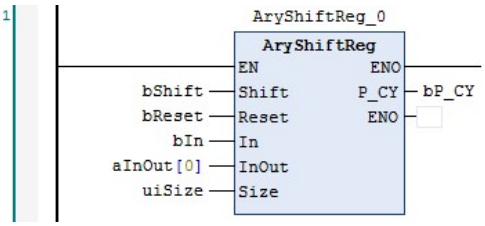


- InOut[] requires the use of Boolean and bit string types, otherwise ENO will be false when the program runs.

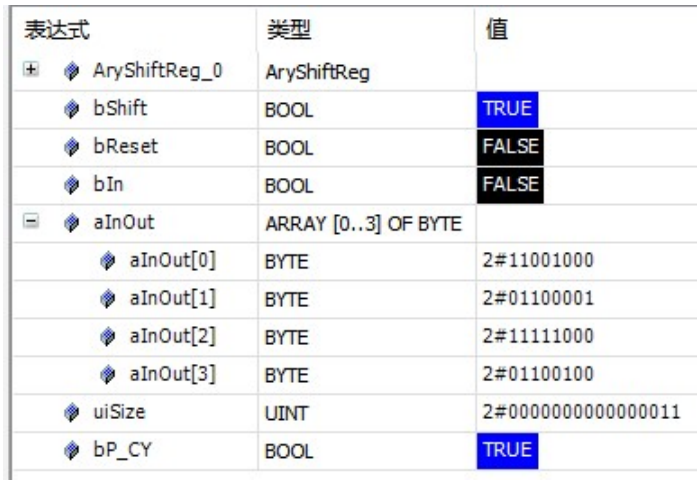
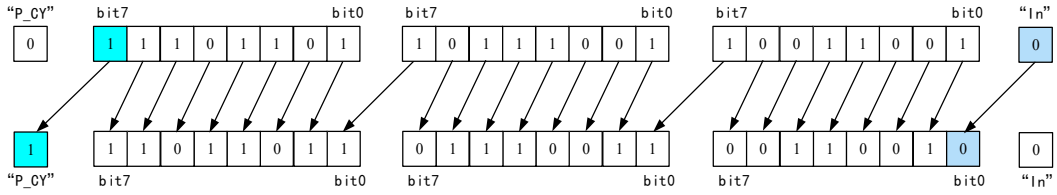
(5) Program example

DEMO	LD	ST
Define the variable		
Sample program		

```
1  PROGRAM AryShiftReg_ST
2  VAR
3      AryShiftReg_0:AryShiftReg;
4      bShift:BOOL;
5      bReset:BOOL;
6      bIn:BOOL;
7      aInOut:ARRAY[0..3]OF BYTE:=[2#11100100,2#00110000,2#11111100,2#01100100];
8      uiSize:UINT;
9      bP_CY:BOOL;
10 END_VAR
```




```
1  AryShiftReg_0(
2      Shift:= bShift,
3      Reset:= bReset,
4      In:= bIn,
5      InOut:= aInOut[0],
6      Size:= uiSize,
7      P_CY=> bP_CY,
8      ENO=> );
```

Operation result	
Principle analysis	

表达式	类型	值
AryShiftReg_0	AryShiftReg	
bShift	BOOL	TRUE
bReset	BOOL	FALSE
bIn	BOOL	FALSE
aInOut	ARRAY [0..3] OF BYTE	
aInOut[0]	BYTE	2#11001000
aInOut[1]	BYTE	2#01100001
aInOut[2]	BYTE	2#11111000
aInOut[3]	BYTE	2#01100100
uiSize	UINT	2#0000000000000011
bP_CY	BOOL	TRUE

1-4-3-6. Left-right shift register AryShiftRegLR

(1) Instruction overview

Shift the entire array elements by 1 bit to the left (or right), and insert the input value at the lowest (or highest) position.

Left-right shift register [AryShiftRegLR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryShiftRegLR	left-right shift register	AryShiftRegLR_0 AryShiftRegLR EN ENO ShiftL P_CY ShiftR Reset In InOut Size	AryShiftRegLR_0(ShiftL:= , ShiftR:= , Reset:= , In:= , InOut:= , Size:= , P_CY=> , ENO=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
ShiftL	Shift left	BOOL	[FALSE,TRUE]	FALSE	When changing from FALSE to True, perform a left shift
ShiftR	Shift right	BOOL	[FALSE,TRUE]	FALSE	When changing from FALSE to True, perform a right shift
Reset	Reset	BOOL	[FALSE,TRUE]	FALSE	When changing from FALSE to True, perform reset
In	Input value	BOOL	[FALSE,TRUE]	FALSE	The value inserted into the lowest or highest position of InOut[]
Size	The number of elements in a positional array	UINT	As the data type	0	The number of elements used in the shift register of InOut[]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
P_CY	carry flag	BOOL	As the data type	-	The value saved in the carry flag

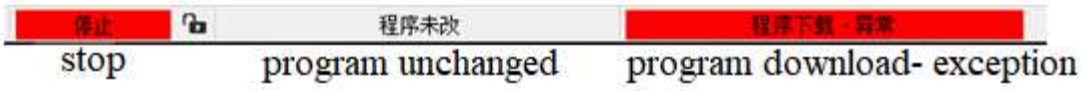
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOut[] array	positional array	BOOL/BYTE/WORD /DWORD/LWORD	As the data type	-	positional array

(3) Function

- When "ShiftL" changes from FALSE to True, shift the "Size" array elements starting from the InOut[] set on pin InOut to the left (high bit direction) by 1 bit. Insert the input value "In" in the lowest position. The highest bit overflowing from the array will be output to the carry (CY) flag "P_CY".
- When "ShiftR" changes from FALSE to True, shift the "Size" array elements starting from the InOut[] set on pin InOut to the right (low bit direction) by 1 bit. Insert the input value "In" into the highest position. The lowest bit overflowing from the array will be output to the carry (CY) flag "P_CY".
- When "Reset" is true, set all bits of "Size" elements starting from the InOut[] set on pin InOut and carry (CY) flags to FALSE.

(4) Note

- When "Reset" is true, even if "ShiftL" or "ShiftR" changes from FALSE to true, no shift is performed.
- When "ShiftL" and "ShiftR" both change from FALSE to True, no shift is performed.
- When the value of "Size" is 0, InOut[] remains unchanged, and the function returns a value of True.
- When the value of "Size" exceeds the array area of InOut[], the PLC will stop prompting an exception, as shown in the following figure:

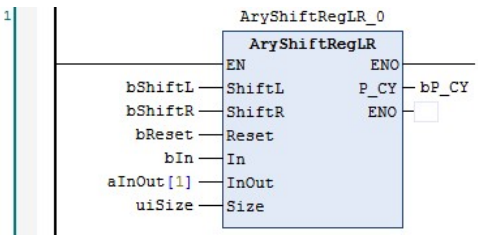


- InOut[] requires Boolean and bit string types, otherwise ENO will be false at runtime.

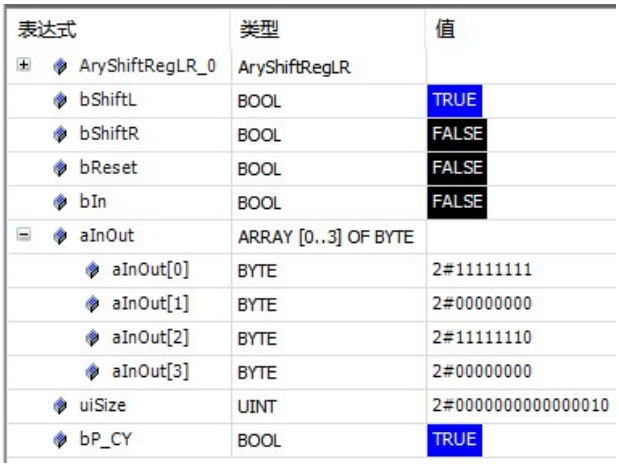
(5) Program example

DEMO	LD	ST
Define the variable		
Sample program		

```
VAR
    AryShiftRegLR_0:AryShiftRegLR;
    bShiftL:BOOL;
    bShiftR:BOOL;
    bReset:BOOL;
    bIn:BOOL;
    aInOut:ARRAY[0..3]OF BYTE:=[2#11111111,2#00000000,2#11111111,2#00000000];
    uiSize:UINT:=2;
    bP_CY:BOOL;
END_VAR
```




```
1  AryShiftRegLR_0(
2      ShiftL:= bShiftL,
3      ShiftR:= bShiftR,
4      Reset:= bReset,
5      In:= bIn,
6      InOut:= aInOut[1],
7      Size:= uiSize,
8      P_CY=> bP_CY,
9      ENO=> );
```

Operation result	

表达式	类型	值
AryShiftRegLR_0	AryShiftRegLR	
bShiftL	BOOL	TRUE
bShiftR	BOOL	FALSE
bReset	BOOL	FALSE
bIn	BOOL	FALSE
aInOut	ARRAY [0..3] OF BYTE	
aInOut[0]	BYTE	2#11111111
aInOut[1]	BYTE	2#00000000
aInOut[2]	BYTE	2#11111110
aInOut[3]	BYTE	2#00000000
uiSize	UINT	2#0000000000000010
bP_CY	BOOL	TRUE

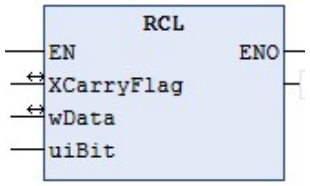
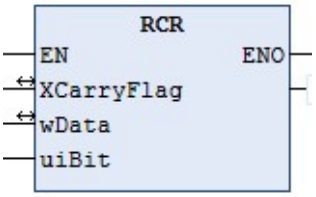
1-4-3-7. Loop left/right shift with carry RCL/RCR

(1) Instruction overview

RCL: After execution, the operand is cyclically shifted to the left with a carry flag, and the bit moved out on the left is directly filled to the carry flag. The original carry flag is shifted to the lowest position on the right.

RCR: After execution, the operand is cyclically shifted to the right with a carry flag, and the bit moved to the right is directly filled to the carry flag. The original carry flag is shifted to the highest position on the left.

Loop left/right shift with carry [RCL/RCR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
RCL	Loop left shift with carry		RCL(XCarryFlag:= , wData:= , uiBit:=);
RCR	Loop right shift with carry		RCR(XCarryFlag:= , wData:= , uiBit:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
uiBit	Loop shift number	UINT	1-1024	-	Loop shift number
VAR_IN_OUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
xCarryFlag	Data carry flag	BOOL	-	-	Data carry flag
wData	data sources	WORD	1-1024	-	data sources

(3) Function

- Shift the WORD type data in "wData" to the left and right according to binary display.
- Specify the number of elements to be converted of "In[]" in "Size".

(4) Program example

① RCL

When the input signal "source data" is 2 # 111000000 and the "shift number" is 1:

ST:

表达式	类型	值	准备值	地址	注释
R_TRIG_0	R_TRIG				
CLK	BOOL	TRUE			Boolean signal to be checked
Q	BOOL	FALSE			``TRUE``: Rising edge detected
源数据	WORD	2#1100000000000000			
循环移位数	UINT	2#0000000000000001			
进位标志位	BOOL	TRUE			
Execute	BOOL	FALSE			

```
1  R_TRIG_0(CLK:=,Q FALSE =>Execute FALSE );
2  IF Execute FALSE THEN
3      RCL(xCarryFlag:=进位标志位 TRUE ,wData:=源数据 2#1100000000000000 ,uiBit:=循环移位数 2#0000000000000001 );
4  END_IF RETURN
```

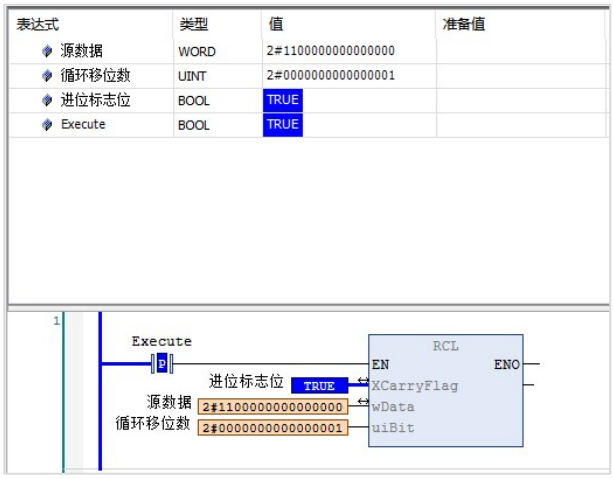
Triggering for the first time, the output signal "result data" is 2 # 11000000000. The valid data is shifted to the left by 1 bit to the carry flag, and the original carry flag is shifted to the lowest position on the right.

表达式	类型	值	准备值	地址	注释
R_TRIG_0	R_TRIG				
CLK	BOOL	TRUE			Boolean signal to be checked
Q	BOOL	FALSE			``TRUE``: Rising edge detected
源数据	WORD	2#1000000000000001			
循环移位数	UINT	2#0000000000000001			
进位标志位	BOOL	TRUE			
Execute	BOOL	FALSE			

```
1  R_TRIG_0(CLK:=,Q FALSE =>Execute FALSE );
2  IF Execute FALSE THEN
3      RCL(xCarryFlag:=进位标志位 TRUE ,wData:=源数据 2#1000000000000001 ,uiBit:=循环移位数 2#0000000000000001 );
4  END_IF RETURN
```

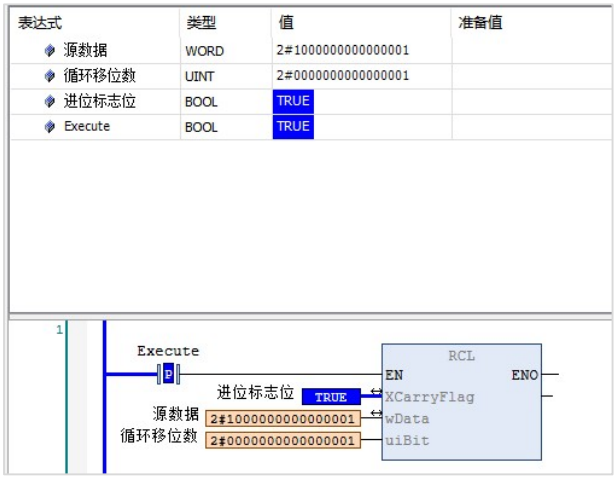
Triggering for the second time, the output signal "data result" is 2 # 1000000000001. The valid data is shifted to the left by 1 bit to the carry flag, and the original carry flag is shifted to the lowest position on the right.

LD:



表达式	类型	值	准备值
源数据	WORD	2#1100000000000000	
循环移位数	UINT	2#0000000000000001	
进位标志位	BOOL	TRUE	
Execute	BOOL	TRUE	

Triggering for the first time, the output signal "result data" is 2 # 11000000000. The valid data is shifted to the left by 1 bit to the carry flag, and the original carry flag is shifted to the lowest position on the right.

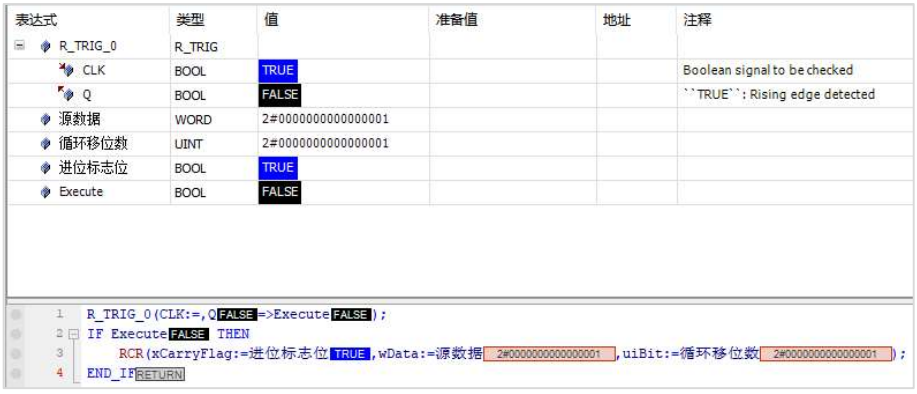


Triggering for the second time, the output signal "data result" is 2 # 1000000000001. The valid data is shifted to the left by 1 bit to the carry flag, and the original carry flag is shifted to the lowest position on the right.

② RCR

When the input signal "source data" is 2 # 000000 0000011 and the "shift number" is 1:

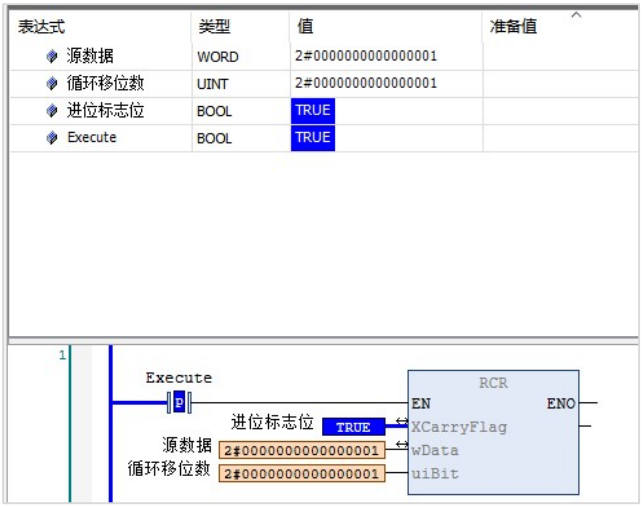
ST:



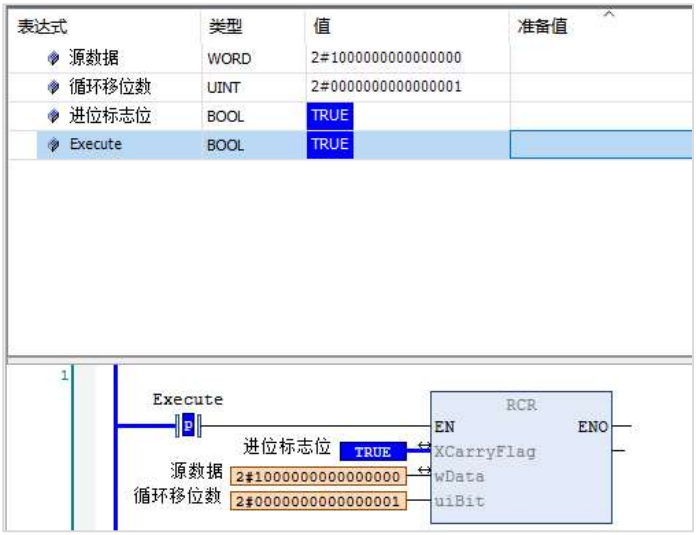
Triggering for the first time, the output signal "result data" is 2 # 1000000000001. The valid data is shifted 1 bit to the right to the carry flag, and the original carry flag is shifted to the highest position on the left.

Triggering for the second time, the output signal "data result" is 2 # 100000000000. The valid data is shifted 1 bit to the right to the carry flag, and the original carry flag is shifted to the highest position on the left.

LD:



Triggering for the first time, the output signal "result data" is 2 # 1000000000001. The valid data is shifted 1 bit to the right to the carry flag, and the original carry flag is shifted to the highest position on the left.



Triggering for the second time, the output signal "data result" is 2 # 100000000000. The valid data is shifted 1 bit to the right to the carry flag, and the original carry flag is shifted to the highest position on the left.

1-4-3-8. Shift N elements of an array left/right ArySHL/ArySHR

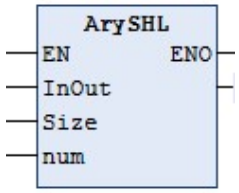
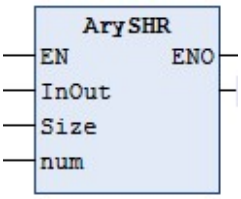
(1) Instruction overview

Shift multiple array elements.

ArySHL: Shift multiple elements in an array to the left (high order direction).

ArySHR: Shift multiple elements in an array to the right (low order direction).

Shift N elements of an array left/right [ArySHL/ArySHR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ArySHL	Left shift of N elements in an array		ArySHL(InOut:= , Size:= , num:=);
ArySHR	Right shift of N elements in an array		ArySHR(InOut:= , Size:= , num:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Size	The number of elements in the shifted object array	UINT	As the data type	1	The number of elements in the shifted object array
Num	Number of shifted digits	UINT	As the data type	1	Number of shifted digits
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOut[] array	Shift Object Array	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT /ULINT/SINT/INT/DINT/ LINT/REAL/LREAL/TIME/ DATE/TOD/DT/STRING	As the data type	-	Shift Object Array

(3) Function

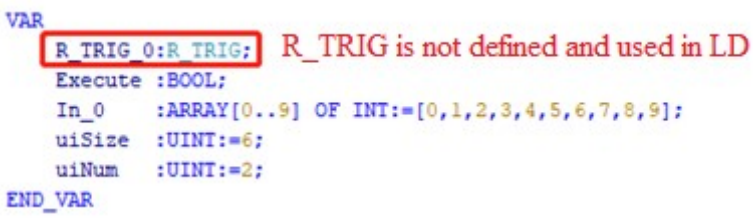
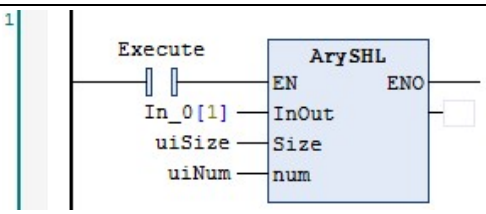
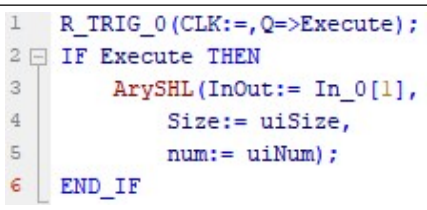
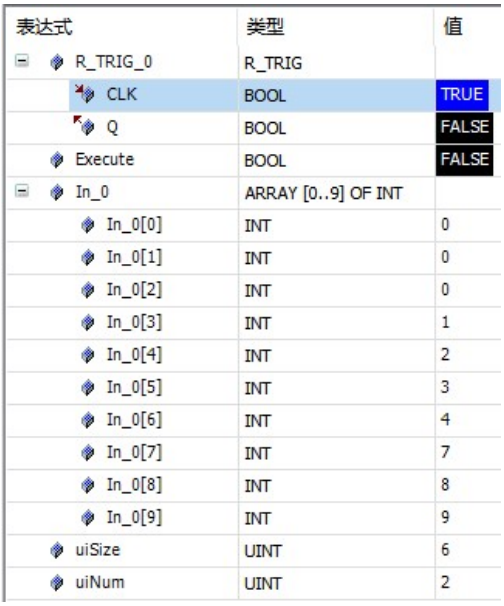
- Shift the "Num" bits left/right for the low/high "Size" elements of the shift object array InOut[].
- Elements are only shifted within "Size", and after shifting, overflow elements from the shifted object array are cleared.
- After shifting, initialize the empty elements of the object array InOut[], that is, assign the default initial value of the corresponding data type to the empty elements.
- The default initial values for each data type are shown in the table below:

Data type	Initial value
BOOL	FALSE
BYTE, WORD, DWORD, LWORD	16#0
USINT, UINT, UDINT, ULINT, SINT, INT, DINT, LINT, REAL, LREAL	0
TIME	T#0ms
DATE	D#1970-1-1
TOD	TOD#0:0:0
DT	DT#1970-1-1-0:0:0
STRING	‘’

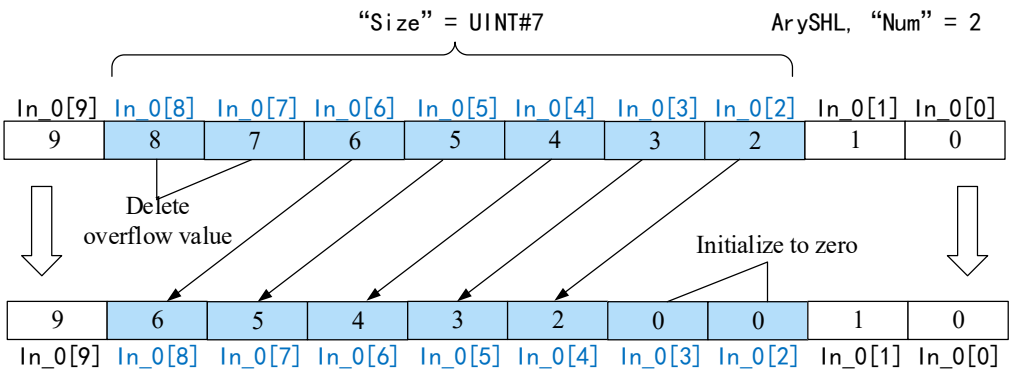
(4) Note

- The value of "Size" should not exceed the array area of InOut[].
- When the value of "Num" or "Size" is 0, no shift action is taken, and the value returned by the function is true, while InOut[] remains unchanged.
- When the value of "Num" is greater than "Size", consecutive Size values starting from InOut[] are initialized to zero.

(5) Program example

DEMO	LD	ST
Define the variable		
Sample program		
Operation result		

表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	TRUE
Q	BOOL	FALSE
Execute	BOOL	FALSE
In_0	ARRAY [0..9] OF INT	
In_0[0]	INT	0
In_0[1]	INT	0
In_0[2]	INT	0
In_0[3]	INT	1
In_0[4]	INT	2
In_0[5]	INT	3
In_0[6]	INT	4
In_0[7]	INT	7
In_0[8]	INT	8
In_0[9]	INT	9
uiSize	UINT	6
uiNum	UINT	2

Principle analysis	

1-4-3-9. Shift N elements of an array left/right NSHLC/NSHRC

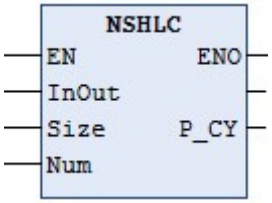
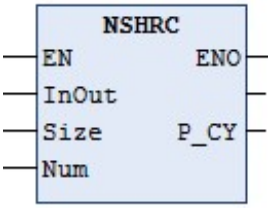
(1) Instruction overview

Move multiple bits along with the carry (CY) flag.

NSHLC: Shift to the left (high bit direction).

NSHRC: Shift to the right (low position direction).

Shift N elements of an array left/right [NSHLC/NSHRC]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
NSHLC	Left shift of N elements in an array		xResult:=NSHLC(InOut:= , Size:= , Num:= , P_CY=>);
NSHRC	Right shift of N elements in an array		xResult:=NSHRC(InOut:= , Size:= , Num:= , P_CY=>);

(2) Related variable

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Size	The number of bits of the shifted object	USINT	As the data type	1	The number of bits of the shifted object
Num	Shift bits	USINT	As the data type	1	Shift bits
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
P_CY	Carry flag	BOOL	As the data type	-	Carry flag
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOut[] array	Shift Object Array	BOOL/BYTE/WORD /DWORD/LWORD	As the data type	-	Shift Object Array

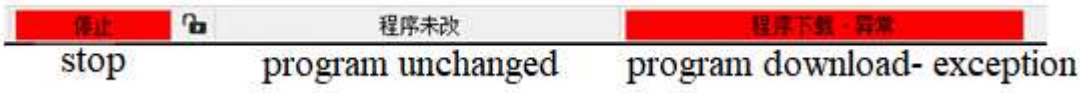
(3) Function

- Move the "Size" array elements starting from the InOut [0] of the shifted object array InOut [] by "Num" bits. The final value of the overflow bit will be output to the carry (CY) flag, and the bit inserted on the opposite side will be filled with 0.
- NSHLC: Shift to the left (from low to high).

- NSHRC: Shift to the right (from high to low).

(4) Note

- When the value of "Num" is 0, no shift is performed, the function returns a value of True, InOut[] remains unchanged, and the value of "P_CY" is FALSE.
- When the value of "Num" is greater than "Size", the value from bit0 of InOut[] to the "Size" bits is FALSE. In addition, the value of the carry (CY) flag is also FALSE.
- When the value of "Size" exceeds the array area of InOut[], a PLC exception will occur as shown in the figure below. This incorrect usage should be avoided.

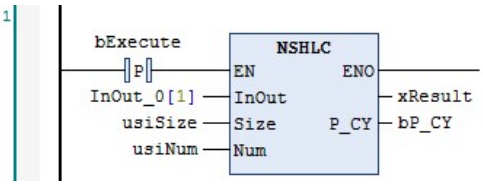


- "InOut[]" requires Boolean and bitstring types, otherwise the runtime function returns FALSE.

(5) Program sample

DEMO	LD	ST
Define the variable		
Sample program		
Operation result		

```
VAR
    bExecute:BOOL;
    InOut_0:ARRAY[0..4] OF BYTE:=[2(2#11110000),3(2#00001111)];
    usiSize:USINT:=15;
    usiNum :USINT:=2;
    bP_CY  :BOOL;
    xResult:BOOL;
END_VAR
```




```
1  R_TRIG_0(CLK:=,Q=>bExecute);
2  IF bExecute THEN
3      xResult:=NSHLC(InOut:= InOut_0[1],
4          Size:= usiSize,
5          Num:= usiNum,
6          P_CY=> bP_CY);
7  END_IF
```

表达式	类型	值
bExecute	BOOL	TRUE
InOut_0	ARRAY [0..4] OF BYTE	
InOut_0[0]	BYTE	2#11110000
InOut_0[1]	BYTE	2#11000000
InOut_0[2]	BYTE	2#00111111
InOut_0[3]	BYTE	2#00001111
InOut_0[4]	BYTE	2#00001111
usiSize	USINT	2#00001111
usiNum	USINT	2#00000010
bP_CY	BOOL	FALSE
xResult	BOOL	TRUE

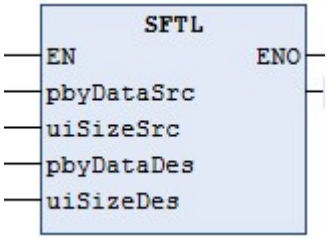
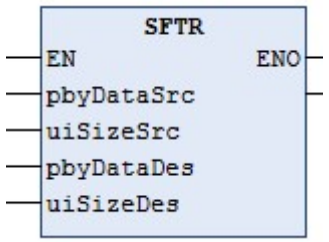
1-4-3-10. Copy bit data to the left/right SFTL/SFTR

(1) Instruction overview

SFTL: Copy the bit data of the specified length from the source array continuously to the left into the target array.

SFTR: Copy the bit data of the specified length from the source array continuously to the right into the target array.

Copy bit data to the left/right [SFTL/SFTR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
SFTL	Copy bit data to the left		SFTL(pbyDataSrc:= , uiSizeSrc:= , pbyDataDes:= , uiSizeDes:=);
SFTR	Copy bit data to the right		SFTR(pbyDataSrc:= , uiSizeSrc:= , pbyDataDes:= , uiSizeDes:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
pbyDataSrc	Source data	POINTER_TO_BYTE	-	-	Source data
pbyDataDes	Target data	POINTER_TO_BYTE	-	-	Target data
uiSizeSrc	Data source numbers	BYTE	-	-	Data source numbers
uiSizeDes	Target data numbers	BYTE	-	-	Target data numbers

(3) Function

- uiSizeSrc must larger than uiSizeDes, otherwise, the PLC will shut down abnormally.

(4) Program sample

① SFTL

Copy the consecutive uiSizeSrc (number of elements in a single copy) bits of data starting from the first address of pointer pbyDataSrc (source array) to the low range of the first address of pbyDataDes (target array). Before copying, all uiSizeDes (offset target) bits of data in pbyDataDes are moved to the high range by uiSizeSrc bits, and only within the range of uiSizeDes (offset target) bits of data in pbyDataDes (target array).

ST：

```
PROGRAM SFTL_ST
VAR
    R_TRIG_0:R_TRIG;
    uiSizeSrc:BYTE;
    uiSizeDes:BYTE;
    源数据:ARRAY[0..3]OF BOOL;
    目标数据:ARRAY[0..5]OF BOOL;
    Execute:BOOL;
END_VAR
```

```
R_TRIG_0(CLK:=,Q=>Execute);
IF Execute=1 THEN
    SFTL(pbyDataSrc:=ADR(源数据[0]),
        uiSizeSrc:=uiSizeSrc,
        pbyDataDes:=ADR(目标数据[0]),
        uiSizeDes:=uiSizeDes);
END_IF
```

Define variables and programming;

表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	FALSE
Q	BOOL	FALSE
uiSizeSrc	BYTE	2
uiSizeDes	BYTE	5
源数据	ARRAY [0..3] OF BO...	
源数据[0]	BOOL	TRUE
源数据[1]	BOOL	TRUE
源数据[2]	BOOL	FALSE
源数据[3]	BOOL	FALSE
目标数据	ARRAY [0..5] OF BO...	
目标数据[0]	BOOL	FALSE
目标数据[1]	BOOL	FALSE
目标数据[2]	BOOL	FALSE
目标数据[3]	BOOL	FALSE
目标数据[4]	BOOL	FALSE
目标数据[5]	BOOL	FALSE
Execute	BOOL	FALSE

```
R_TRIG_0(CLK:=,Q FALSE =>Execute FALSE );
IF Execute FALSE =1 THEN
    SFTL(pbyDataSrc:=ADR(源数据[0] TRUE ),
        uiSizeSrc:=uiSizeSrc 2 ,
        pbyDataDes:=ADR(目标数据[0] FALSE ),
        uiSizeDes:=uiSizeDes 5 );
END_IF RETURN
```

Initial assignment;

表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	TRUE
Q	BOOL	FALSE
uiSizeSrc	BYTE	2
uiSizeDes	BYTE	5
源数据	ARRAY [0..3] OF BO...	
源数据[0]	BOOL	TRUE
源数据[1]	BOOL	TRUE
源数据[2]	BOOL	FALSE
源数据[3]	BOOL	FALSE
目标数据	ARRAY [0..5] OF BO...	
目标数据[0]	BOOL	TRUE
目标数据[1]	BOOL	TRUE
目标数据[2]	BOOL	FALSE
目标数据[3]	BOOL	FALSE
目标数据[4]	BOOL	FALSE
目标数据[5]	BOOL	FALSE
Execute	BOOL	FALSE

```
1  R_TRIG_0(CLK:=,Q FALSE =>Execute FALSE );
2  IF Execute FALSE =1 THEN
3      SFTL(pbyDataSrc:=ADR(源数据[0] TRUE ),
4          uiSizeSrc:=uiSizeSrc 2 ,
5          pbyDataDes:=ADR(目标数据[0] TRUE ),
6          uiSizeDes:=uiSizeDes 5 );
7  END_IF RETURN
```

Triggering for the first time, as the target data is all FALSE, shifting 2 bits towards the high bit is still FALSE; Source data [0], source data [1] are passed to target data [0], and the target data [1] is true;

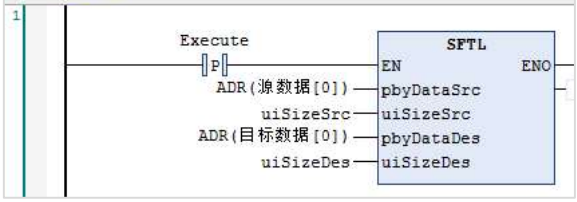
表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	TRUE
Q	BOOL	FALSE
uiSizeSrc	BYTE	2
uiSizeDes	BYTE	5
源数据	ARRAY [0..3] OF BO...	
源数据[0]	BOOL	TRUE
源数据[1]	BOOL	TRUE
源数据[2]	BOOL	FALSE
源数据[3]	BOOL	FALSE
目标数据	ARRAY [0..5] OF BO...	
目标数据[0]	BOOL	TRUE
目标数据[1]	BOOL	TRUE
目标数据[2]	BOOL	TRUE
目标数据[3]	BOOL	TRUE
目标数据[4]	BOOL	FALSE
目标数据[5]	BOOL	FALSE
Execute	BOOL	FALSE

```
1  R_TRIG_0(CLK:=,Q FALSE =>Execute FALSE );
2  IF Execute FALSE =1 THEN
3      SFTL(pbyDataSrc:=ADR(源数据[0] TRUE ),
4          uiSizeSrc:=uiSizeSrc 2 ,
5          pbyDataDes:=ADR(目标数据[0] TRUE ),
6          uiSizeDes:=uiSizeDes 5 );
7  END_IF RETURN
```

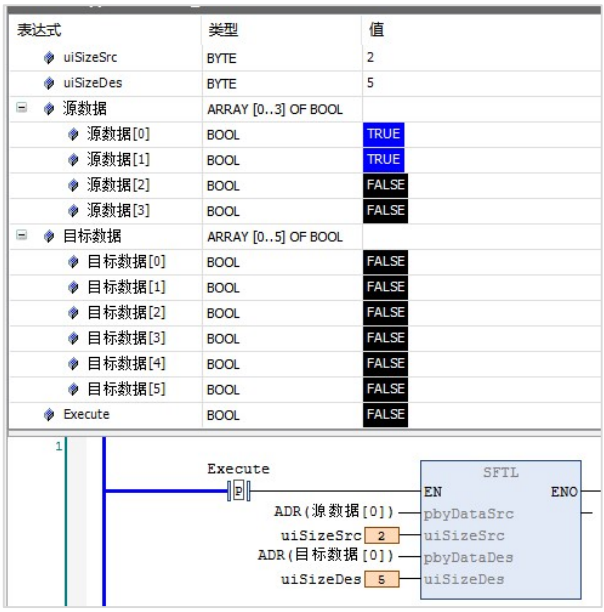
Triggering the second time, the value of the original target data [0] - target data [4] is moved 2 bits to the high position, to the target data [2] - target data [4], and the overflow 2 bits are moved to the target data [0] - target data [1]; Next, the value of the source data [0] and source data [1] are passed to the target data [0] and target data [1].

LD：

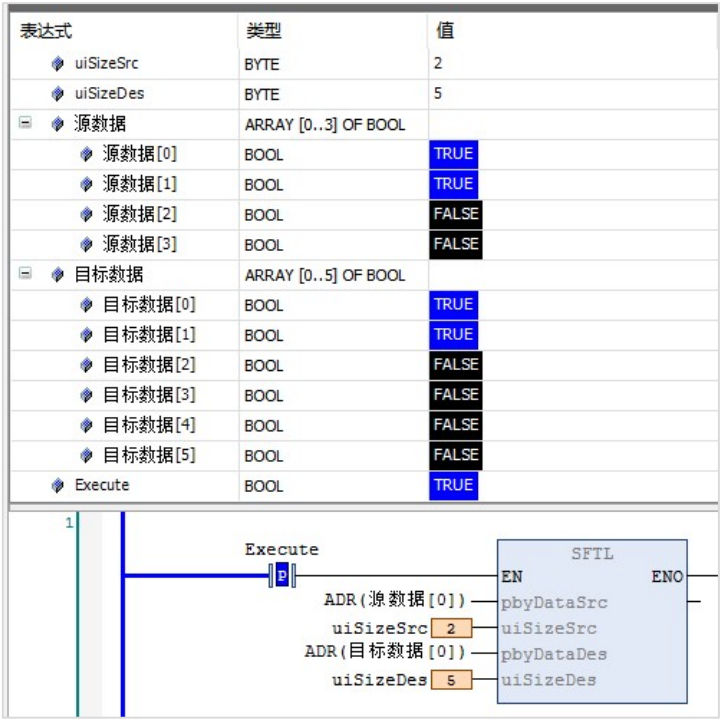
```
PROGRAM SFTL_LD
VAR
    uiSizeSrc:BYTE;
    uiSizeDes:BYTE;
    源数据:ARRAY[0..3]OF BOOL;
    目标数据:ARRAY[0..5]OF BOOL;
    Execute:BOOL;
END_VAR
```



Defining variables and programming.

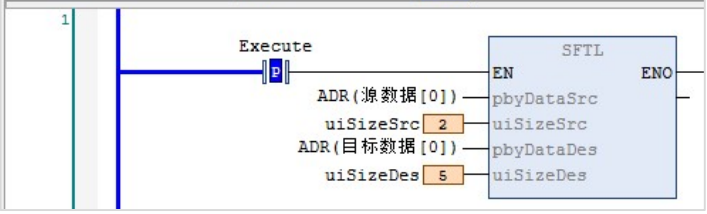


Initial assignment.



Triggering for the first time, as the target data is all FALSE, shifting 2 bits towards the high bit is still FALSE; Source data [0], source data [1] are passed to target data [0], and the target data [1] is true.

表达式	类型	值
uiSizeSrc	BYTE	2
uiSizeDes	BYTE	5
源数据	ARRAY [0..3] OF BOOL	
源数据[0]	BOOL	TRUE
源数据[1]	BOOL	TRUE
源数据[2]	BOOL	FALSE
源数据[3]	BOOL	FALSE
目标数据	ARRAY [0..5] OF BOOL	
目标数据[0]	BOOL	TRUE
目标数据[1]	BOOL	TRUE
目标数据[2]	BOOL	TRUE
目标数据[3]	BOOL	TRUE
目标数据[4]	BOOL	FALSE
目标数据[5]	BOOL	FALSE
Execute	BOOL	TRUE



Triggering the second time, the value of the original target data [0] - target data [4] is moved 2 bits to the high position, to the target data [2] - target data [4], and the overflow 2 bits are moved to the target data [0] - target data [1]; Next, the value of the source data [0] and source data [1] is passed to the target data [0] and target data [1].

② SFTR

Copy the consecutive uiSizeSrc (number of array elements in a single copy) bit data starting from the first address of pointer pbyDataSrc (source array) to the high bit area of pbyDataDes (target array). Before copying, all uiSizeSrc (offset target) bit data of pbyDataDes will be shifted low by uiSizeSrc bits, and only within the range of uiSizeDes (offset target) data of pbyDataDes (target array).

ST：

```
PROGRAM SFTL_ST
VAR
    R_TRIG_0:R_TRIG;
    uiSizeSrc:BYTE;
    uiSizeDes:BYTE;
    源数据:ARRAY[0..3]OF BOOL;
    目标数据:ARRAY[0..5]OF BOOL;
    Execute:BOOL;
END_VAR
```

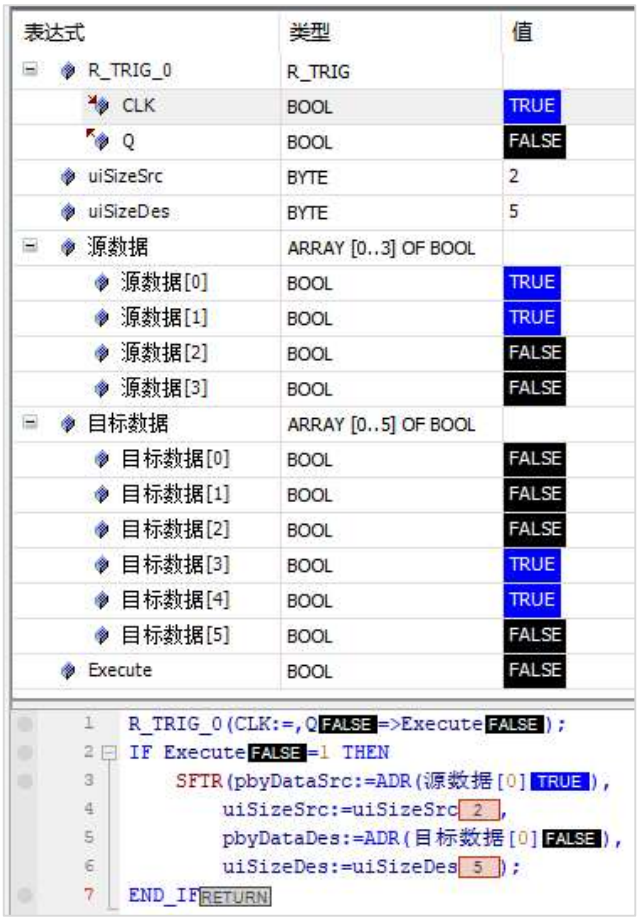
```
R_TRIG_0(CLK:=,Q=>Execute);
IF Execute=1 THEN
    SFTR(pbyDataSrc:=ADR(源数据[0]),
        uiSizeSrc:=uiSizeSrc,
        pbyDataDes:=ADR(目标数据[0]),
        uiSizeDes:=uiSizeDes);
END_IF
```

Defining variables and programming.

表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	FALSE
Q	BOOL	FALSE
uiSizeSrc	BYTE	2
uiSizeDes	BYTE	5
源数据	ARRAY [0..3] OF BOOL	
源数据[0]	BOOL	TRUE
源数据[1]	BOOL	TRUE
源数据[2]	BOOL	FALSE
源数据[3]	BOOL	FALSE
目标数据	ARRAY [0..5] OF BOOL	
目标数据[0]	BOOL	FALSE
目标数据[1]	BOOL	FALSE
目标数据[2]	BOOL	FALSE
目标数据[3]	BOOL	FALSE
目标数据[4]	BOOL	FALSE
目标数据[5]	BOOL	FALSE
Execute	BOOL	FALSE

```
1  R_TRIG_0(CLK:=,Q FALSE =>Execute FALSE );
2  IF Execute FALSE =1 THEN
3      SFTR(pbyDataSrc:=ADR(源数据[0] TRUE ),
4          uiSizeSrc:=uiSizeSrc 2 ,
5          pbyDataDes:=ADR(目标数据[0] FALSE ),
6          uiSizeDes:=uiSizeDes 5 );
7  END_IF RETURN
```

Initial assignment.



表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	TRUE
Q	BOOL	FALSE
uiSizeSrc	BYTE	2
uiSizeDes	BYTE	5
源数据	ARRAY [0..3] OF BOOL	
源数据[0]	BOOL	TRUE
源数据[1]	BOOL	TRUE
源数据[2]	BOOL	FALSE
源数据[3]	BOOL	FALSE
目标数据	ARRAY [0..5] OF BOOL	
目标数据[0]	BOOL	FALSE
目标数据[1]	BOOL	FALSE
目标数据[2]	BOOL	FALSE
目标数据[3]	BOOL	TRUE
目标数据[4]	BOOL	TRUE
目标数据[5]	BOOL	FALSE
Execute	BOOL	FALSE

```
1  R_TRIG_0(CLK:=,Q FALSE =>Execute FALSE );
2  IF Execute FALSE =1 THEN
3      SFTR(pbyDataSrc:=ADR(源数据[0] TRUE ),
4          uiSizeSrc:=uiSizeSrc 2 ,
5          pbyDataDes:=ADR(目标数据[0] FALSE ),
6          uiSizeDes:=uiSizeDes 5 );
7  END_IF RETURN
```

Triggering for the first time, as the target data is all FALSE, shifting 2 bits towards the low bit is still FALSE; Source data [0], source data [1] are passed to target data [3], and the target data [4] is true.

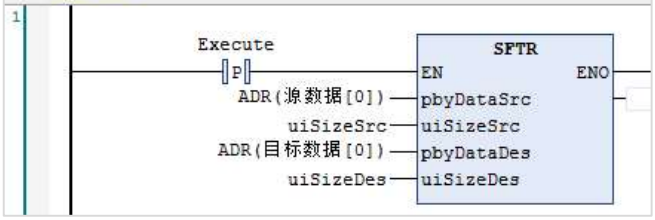
表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	TRUE
Q	BOOL	FALSE
uiSizeSrc	BYTE	2
uiSizeDes	BYTE	5
源数据	ARRAY [0..3] OF BOOL	
源数据[0]	BOOL	TRUE
源数据[1]	BOOL	TRUE
源数据[2]	BOOL	FALSE
源数据[3]	BOOL	FALSE
目标数据	ARRAY [0..5] OF BOOL	
目标数据[0]	BOOL	FALSE
目标数据[1]	BOOL	TRUE
目标数据[2]	BOOL	TRUE
目标数据[3]	BOOL	TRUE
目标数据[4]	BOOL	TRUE
目标数据[5]	BOOL	FALSE
Execute	BOOL	FALSE

```
1  R_TRIG_0(CLK:=,Q FALSE =>Execute FALSE );
2  IF Execute FALSE =1 THEN
3      SFTR(pbyDataSrc:=ADR(源数据[0] TRUE ),
4          uiSizeSrc:=uiSizeSrc 2 ,
5          pbyDataDes:=ADR(目标数据[0] FALSE ),
6          uiSizeDes:=uiSizeDes 5 );
7  END_IF RETURN
```

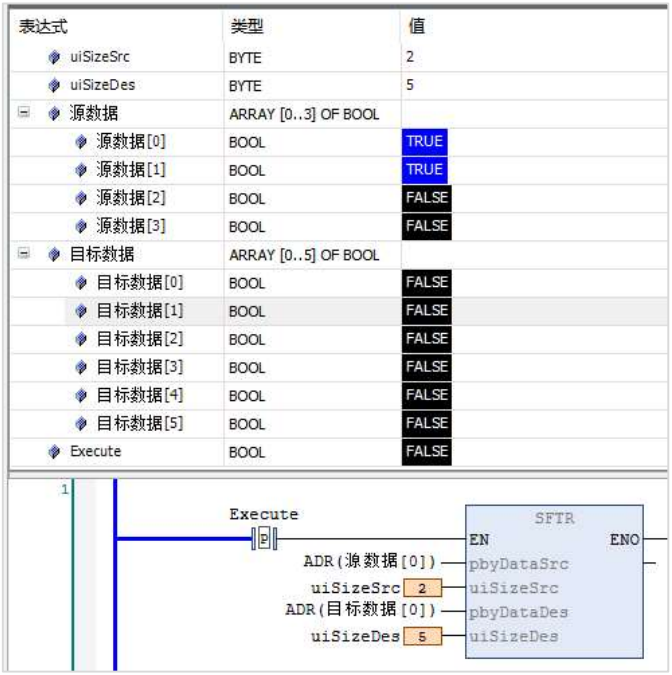
Triggering the second time, the value of the original target data [0] - target data [4] is moved 2 bits to the low position, to the target data [0] - target data [2], and the overflow 2 bits are moved to the target data [3] - target data [4]; Next, the value of the source data [0] and source data [1] is passed to the target data [3] and target data [4].

LD：

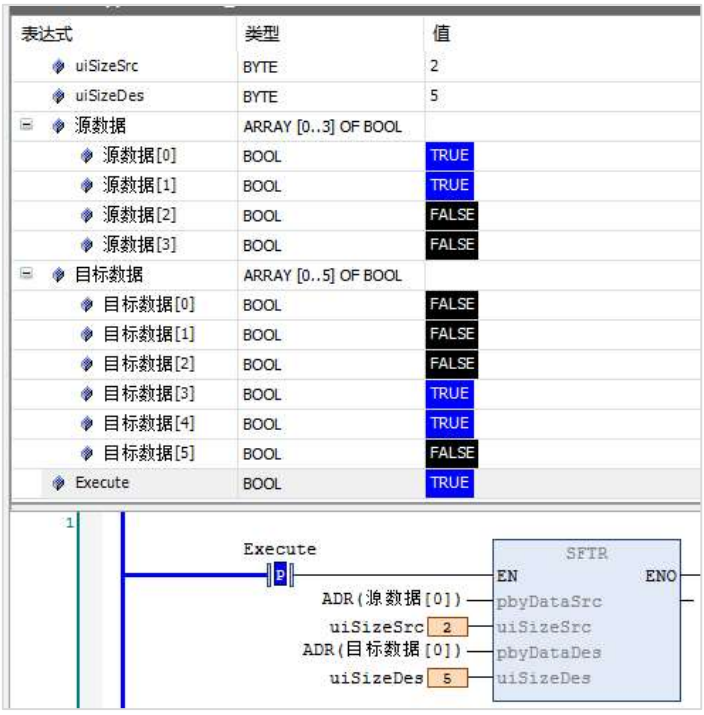
```
1  PROGRAM SFTL_LD
2  VAR
3      uiSizeSrc:BYTE;
4      uiSizeDes:BYTE;
5      源数据:ARRAY[0..3]OF BOOL;
6      目标数据:ARRAY[0..5]OF BOOL;
7      Execute:BOOL;
8  END_VAR
```



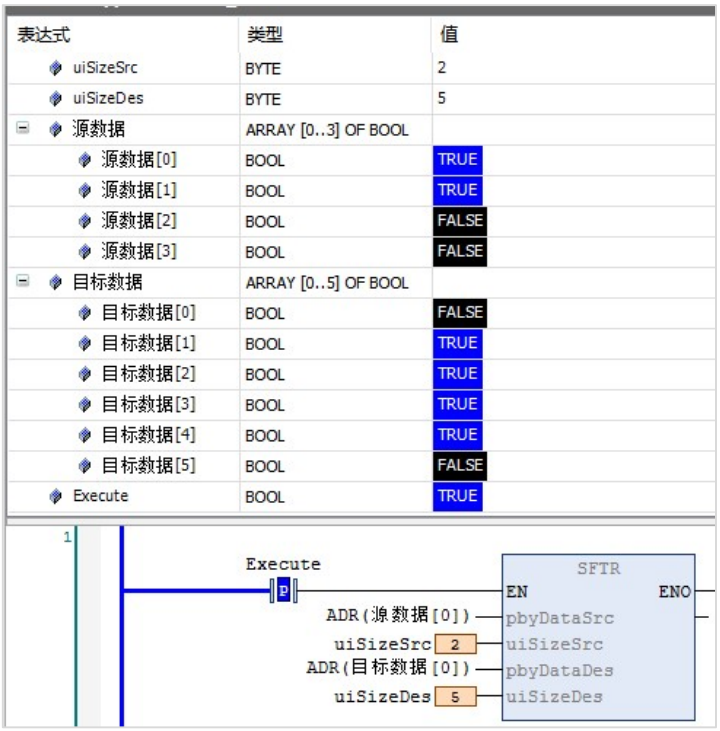
Defining variables and programming.



Initial assignment.



Triggering for the first time, as the target data is all FALSE, shifting 2 bits towards the low bit is still FALSE; Source data [0], source data [1] are passed to target data [3], and the target data [4] is true.



Triggering the second time, the value of the original target data [0] - target data [4] is moved 2 bits to the low position, to the target data [0] - target data [2], and the overflow 2 bits are moved to the target data [3] - target data [4]; Next, the value of the source data [0] and source data [1] is passed to the target data [3] and target data [4].

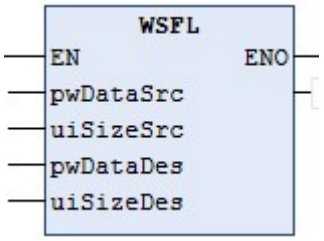
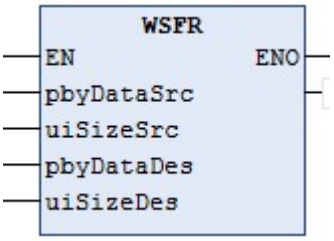
1-4-3-11. Copy word data to the left/right WSFL/WSFR

(1) Instruction overview

WSFL: Copy word data of the specified length from the source array continuously to the left into the target array.

WSFR: Copy word data of the specified length from the source array continuously to the right into the target array.

Copy word data to the left/right [WSFL/WSFR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
WSFL	Copy word data to the left		WSFL(pwDataSrc:= ,uiSizeSrc:= , pwDataDes:= , uiSizeDes:=);
WSFR	Copy word data to the right		WSFR(pwDataSrc:= , uiSizeSrc:= , pwDataDes:= , uiSizeDes:=);

(2) Related variable

VAR_INPUT	Name	Data type	Effective range	Initial value
pbyDataSrc	Source data	POINTER_TO_WORD	-	-
pbyDataDes	Target data	POINTER_TO_WORD	-	-
uiSizeSrc	Source data numbers	UINT	1-1024	-
uiSizeDes	Target data numbers	UINT	1-1024	-

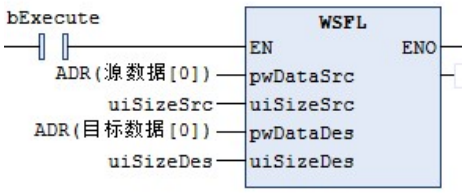
(3) Note

- UiSizeSrc must not be greater than uiSizeDes, otherwise the PLC will shut down abnormally.

(4) Sample program

DEMO	LD	ST
Define variables	VAR R_TRIG_0:R_TRIG; bExecute: BOOL; 源数据:ARRAY[0..3] OF INT:=[10,20,30,40]; 目标数据:ARRAY[0..9]OF INT:=[1,2,3,4,5,6,7,8,9,10]; uiSizeSrc:UINT; uiSizeDes:UINT; END_VAR	

Sample program




```
R_TRIG_0(CLK:=,Q=>bExecute);
IF bExecute=1 THEN
    WSFL(pwDataSrc:= ADR(源数据[0]),
         uiSizeSrc:= uiSizeSrc,
         pwDataDes:= ADR(目标数据[0]),
         uiSizeDes:= uiSizeDes);
END_IF
```

Variable assignment

表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	FALSE
Q	BOOL	FALSE
bExecute	BOOL	FALSE
源数据	ARRAY [0..3] OF INT	
源数据[0]	INT	10
源数据[1]	INT	20
源数据[2]	INT	30
源数据[3]	INT	40
目标数据	ARRAY [0..9] OF INT	
目标数据[0]	INT	1
目标数据[1]	INT	2
目标数据[2]	INT	3
目标数据[3]	INT	4
目标数据[4]	INT	5
目标数据[5]	INT	6
目标数据[6]	INT	7
目标数据[7]	INT	8
目标数据[8]	INT	9
目标数据[9]	INT	10
uiSizeSrc	UINT	2
uiSizeDes	UINT	6

Operation twice	

表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	TRUE
Q	BOOL	FALSE
bExecute	BOOL	FALSE
源数据	ARRAY [0..3] OF INT	
源数据[0]	INT	10
源数据[1]	INT	20
源数据[2]	INT	30
源数据[3]	INT	40
目标数据	ARRAY [0..9] OF INT	
目标数据[0]	INT	10
目标数据[1]	INT	20
目标数据[2]	INT	10
目标数据[3]	INT	20
目标数据[4]	INT	1
目标数据[5]	INT	2
目标数据[6]	INT	7
目标数据[7]	INT	8
目标数据[8]	INT	9
目标数据[9]	INT	10
uiSizeSrc	UINT	2
uiSizeDes	UINT	6

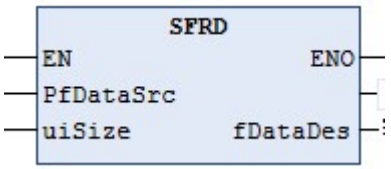
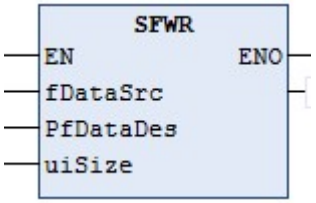
1-4-3-12. First in, first out data read/write SFRD/SFWR

(1) Instruction overview

SFRD: Read the content of a specified length of data after execution.

SFWR: Write the content of a specified length of data after execution.

First in, first out data read/write [SFRD/SFWR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
SFRD	First in, first out data reading		SFRD(pfDataSrc:=, uiSize:=, fDataDes=>);
SFWR	First in, first out data writing		SFWR(fDataSrc:=, pfDataDes:=, uiSize:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value
PfDataSrc	Source data	POINTER_TO_REAL	-	-
uiSize	The number of data floating point elements or the source data queue	UINT	-	-
fDataDes	Read data	REAL	-	-
VAR_OUTPUT	Name	Data type	Effective range	Initial value
fDataSrc	Source data	REAL	-	-
PfDataDes	Target data	POINTER_TO_REAL	-	-
uiSize	The number of data floating point elements or the source data queue	UINT	-	-

(3) Note

- The first element of the source data needs to be assigned in advance, otherwise the execution will be abnormal.

(4) Sample program

① SFRD

Read the first item of the first in, first out queue pfDataSrc (pointer pointing to source data) into fDataDes (target data), and then move the queue pfDatasrc to the right one word word by word, decreasing the queue pointer. Using the first numbered device as the pointer, when the instruction is executed, the content value of the pointer is first reduced by 1, and then the content value of the device specified by pfDataSrc is written to the

position specified by the pointer in the first in, first out fDataDes data string. If the pointer is already 0, the instruction does not process the aforementioned operation.

ST：

表达式	类型	值
源数据	ARRAY [0..9] OF REAL	
源数据[0]	REAL	1
源数据[1]	REAL	1
源数据[2]	REAL	2
源数据[3]	REAL	3
源数据[4]	REAL	4
源数据[5]	REAL	5
源数据[6]	REAL	6
源数据[7]	REAL	7
源数据[8]	REAL	8
源数据[9]	REAL	9
数据长度	UINT	10
结果数据	REAL	0
bExecute	BOOL	FALSE
R_TRIG_0	R_TRIG	
CLK	BOOL	FALSE
Q	BOOL	FALSE

```
1  源数据[0] 1 :=1;
2  R_TRIG_0(CLK:= , Q FALSE => bExecute FALSE );
3  IF bExecute FALSE =1 THEN
4      SFRD(PfDataSrc:= ADR(源数据),
5          uiSize:= 数据长度 10 ,
6          fDataDes=> 结果数据 0 );
7  END_IF RETURN
```

Initial status.

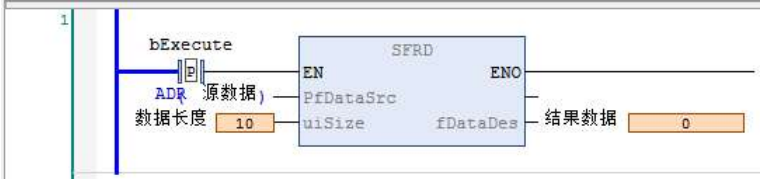
表达式	类型	值
源数据	ARRAY [0..9] OF REAL	
源数据[0]	REAL	1
源数据[1]	REAL	4
源数据[2]	REAL	5
源数据[3]	REAL	6
源数据[4]	REAL	7
源数据[5]	REAL	8
源数据[6]	REAL	9
源数据[7]	REAL	9
源数据[8]	REAL	9
源数据[9]	REAL	9
数据长度	UINT	10
结果数据	REAL	3
bExecute	BOOL	FALSE
R_TRIG_0	R_TRIG	
CLK	BOOL	TRUE
Q	BOOL	FALSE

```
1  源数据[0] 1 :=1;
2  R_TRIG_0(CLK:= , Q FALSE => bExecute FALSE );
3  IF bExecute FALSE =1 THEN
4      SFRD(PfDataSrc:= ADR(源数据),
5          uiSize:= 数据长度 10 ,
6          fDataDes=> 结果数据 3 );
7  END_IF RETURN
```

After triggering three times.

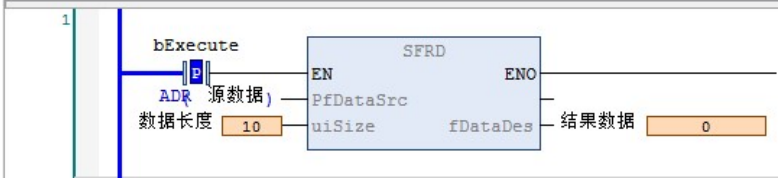
LD：

表达式	类型	值	准备值
源数据	ARRAY [0..9] OF REAL		
源数据[0]	REAL	1	
源数据[1]	REAL	1	
源数据[2]	REAL	2	
源数据[3]	REAL	3	
源数据[4]	REAL	4	
源数据[5]	REAL	5	
源数据[6]	REAL	6	
源数据[7]	REAL	7	
源数据[8]	REAL	8	
源数据[9]	REAL	9	
数据长度	UINT	10	
结果数据	REAL	0	
bExecute	BOOL	FALSE	



Initial status.

表达式	类型	值	准备值
源数据	ARRAY [0..9] OF REAL		
源数据[0]	REAL	0	
源数据[1]	REAL	2	
源数据[2]	REAL	3	
源数据[3]	REAL	4	
源数据[4]	REAL	5	
源数据[5]	REAL	6	
源数据[6]	REAL	7	
源数据[7]	REAL	8	
源数据[8]	REAL	9	
源数据[9]	REAL	9	
数据长度	UINT	10	
结果数据	REAL	0	
bExecute	BOOL	TRUE	



After triggering three times.

② SFWR

Write the value of fDataSrc (source data) into a "first in, first out" queue starting from the address of pfDataDes (target data) with a number of uiSize (data length), using the first numbered device as the pointer. When the instruction is executed, the pointer's content value first increases by 1, and then the content value of the device specified by fDataSrc is written to the position specified by the pointer in the first in, first out pfDataDes data string.

ST：

表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	FALSE
Q	BOOL	FALSE
源数据	REAL	1
数据长度	UINT	10
结果数据	ARRAY [0..9] OF REAL	
结果数据[0]	REAL	0
结果数据[1]	REAL	0
结果数据[2]	REAL	0
结果数据[3]	REAL	0
结果数据[4]	REAL	0
结果数据[5]	REAL	0
结果数据[6]	REAL	0
结果数据[7]	REAL	0
结果数据[8]	REAL	0
结果数据[9]	REAL	0
bExecute	BOOL	FALSE

```
1  R_TRIG_0(CLK:= , Q=> bExecute);
2  IF bExecute=1 THEN
3      SFWR(fDataSrc:= 源数据[1],
4          PfDataDes:= ADR(结果数据),
5          uiSize:= 数据长度[10]);
6  END_IF
```

Initial status.

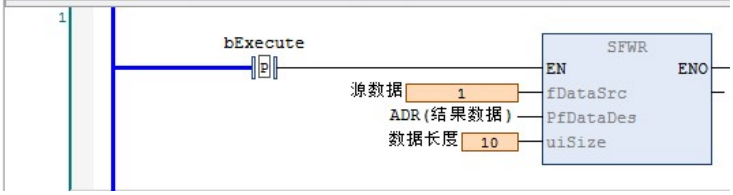
表达式	类型	值
R_TRIG_0	R_TRIG	
CLK	BOOL	TRUE
Q	BOOL	FALSE
源数据	REAL	1
数据长度	UINT	10
结果数据	ARRAY [0..9] OF REAL	
结果数据[0]	REAL	3
结果数据[1]	REAL	1
结果数据[2]	REAL	1
结果数据[3]	REAL	1
结果数据[4]	REAL	0
结果数据[5]	REAL	0
结果数据[6]	REAL	0
结果数据[7]	REAL	0
结果数据[8]	REAL	0
结果数据[9]	REAL	0
bExecute	BOOL	FALSE

```
1  R_TRIG_0(CLK:= , Q=> bExecute);
2  IF bExecute=1 THEN
3      SFWR(fDataSrc:= 源数据[1],
4          PfDataDes:= ADR(结果数据),
5          uiSize:= 数据长度[10]);
6  END_IF
```

After triggering three times.

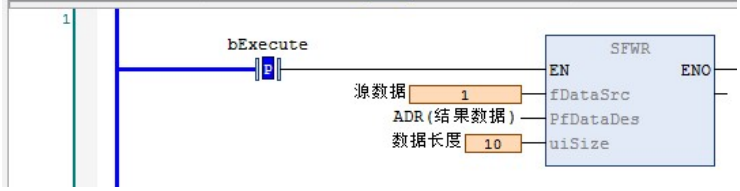
LD：

表达式	类型	值	准备值
源数据	REAL	1	
数据长度	UINT	10	
结果数据	ARRAY [0..9] OF REAL		
结果数据[0]	REAL	0	
结果数据[1]	REAL	0	
结果数据[2]	REAL	0	
结果数据[3]	REAL	0	
结果数据[4]	REAL	0	
结果数据[5]	REAL	0	
结果数据[6]	REAL	0	
结果数据[7]	REAL	0	
结果数据[8]	REAL	0	
结果数据[9]	REAL	0	
bExecute	BOOL	FALSE	



Initial status.

表达式	类型	值	准备值
源数据	REAL	1	
数据长度	UINT	10	
结果数据	ARRAY [0..9] OF REAL		
结果数据[0]	REAL	3	
结果数据[1]	REAL	1	
结果数据[2]	REAL	1	
结果数据[3]	REAL	1	
结果数据[4]	REAL	0	
结果数据[5]	REAL	0	
结果数据[6]	REAL	0	
结果数据[7]	REAL	0	
结果数据[8]	REAL	0	
结果数据[9]	REAL	0	
bExecute	BOOL	TRUE	



After triggering three times.

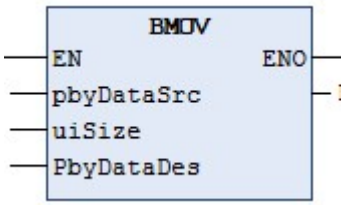
1-4-4. Move instructions

1-4-4-1. Bulk transfer BMOV

(1) Instruction overview

Copy the source data to the target data at the specified length.

Bulk transfer [BMOV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
BMOV	Batch transmission of data		BMOV(pbyDataSrc:= , uiSize:= , PbyDataDes:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value
pbyDataSrc	Starting address for data replication	POINTER TO BYTE	-	-
uiSize	Number of variables	UINT	As the data type	0
VAR_IN_OUT	Name	Data type	Effective range	Initial value
PbyDataDes	Starting address for data storage	POINTER TO BYTE	-	-

(3) Function

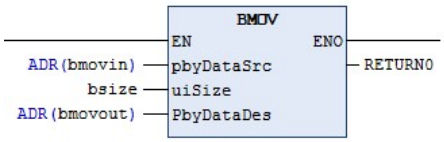
- This instruction is contact driven, copying the uiSize variable values of the specified starting address of pbyDataSrc to the uiSize units of the specified starting address of pbyDataDes. The range of uiSize values is 1-1024. Return value, True: Execution successful, FALSE: Execution error.

(4) Note

- Uisize is byte numbers.

(5) Sample program

DEMO	LD	ST
Define variables		VAR bmovin:ARRAY[0..5] OF INT; bmovout:ARRAY[0..5] OF INT; bsize:UINT; RETURN0: BOOL; END_VAR

<table>
<tr><td>Sample program</td><td> 
</td><td>

```
BMOV(pbyDataSrc:=ADR(bmovin) ,
     uiSize:=bsize ,
     PbyDataDes:=ADR(bmovout) );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
bmovin	ARRAY [0..5] OF INT	
bmovin[0]	INT	111
bmovin[1]	INT	222
bmovin[2]	INT	333
bmovin[3]	INT	444
bmovin[4]	INT	555
bmovin[5]	INT	666
bmovout	ARRAY [0..5] OF INT	
bmovout[0]	INT	111
bmovout[1]	INT	222
bmovout[2]	INT	333
bmovout[3]	INT	444
bmovout[4]	INT	555
bmovout[5]	INT	0
bsize	UINT	10
RETURN0	BOOL	TRUE

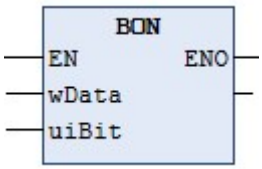
</td></tr>
<tr><td>Result analysis</td><td colspan="2">Uisize represents the number of bytes, with data set to 10; The array element is defined as INT, which is 2 bytes. Therefore, it transfers the data from bmovin[0] - bmovin[4] to bmovout[0] - bmovout[4].</td></tr>
</table>

1-4-4-2. Obtain data specified bit status BON

(1) Instruction overview

Determine the status of the uiBit bit of data wData, store the result in BON, and output it from the right side.

Obtain data specified bit status [BON]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
BON	Obtain data specified bit status		BON(wData:= , uiBit:=);

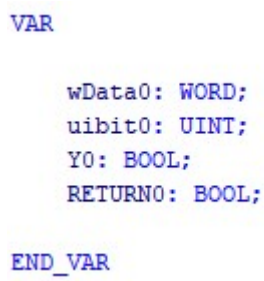
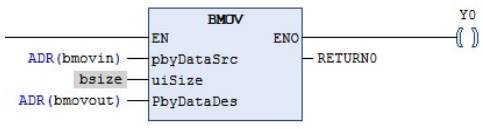
(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
wData	Data that needs to be judged	WORD	遵照数据类型	0	Data that needs to be judged (decimal)
uiBit	The number of data bits	UINT	遵照数据类型	0	The number of data bits (binary bit numbers)

(3) Function

- Determine the status of the uiBit bit of data wData, store the result in BON, and output it from the right side.

(4) Sample program

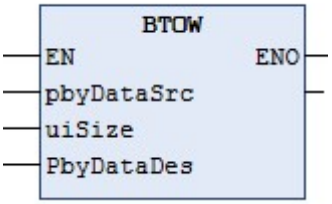
DEMO	LD	ST
Define variables		
Sample program		Y0:=BON(wData:=wData0, uiBit:=uisize0);
Operation result		
Result analysis	Set the value of wData0 to 22 (binary display 10110), set uibit0 to 2, that is, bit2, with a value of 1, and output the result as True.	

1-4-4-3. Data combination of byte unit BTOW

(1) Instruction overview

An instruction that combines the low 8 bits (low bytes) of continuous 16 bits data together.

Data combination of byte unit [BTOW]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
BTOW	Data combination of byte unit		BTOW(pbyDataSrc:= , uiSize:= , PbyDataDes:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
pbyDataSrc	Starting address of combined data	POINTER TO BYTE	-	-	Starting address of combined data
uiSize	Number of combined data	UINT	As the data type	0	Number of combined data
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
PbyDataDes	Target address for storing combined data	POINTER TO BYTE	-	-	Target address for storing combined data

(3) Function

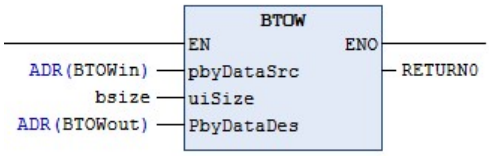
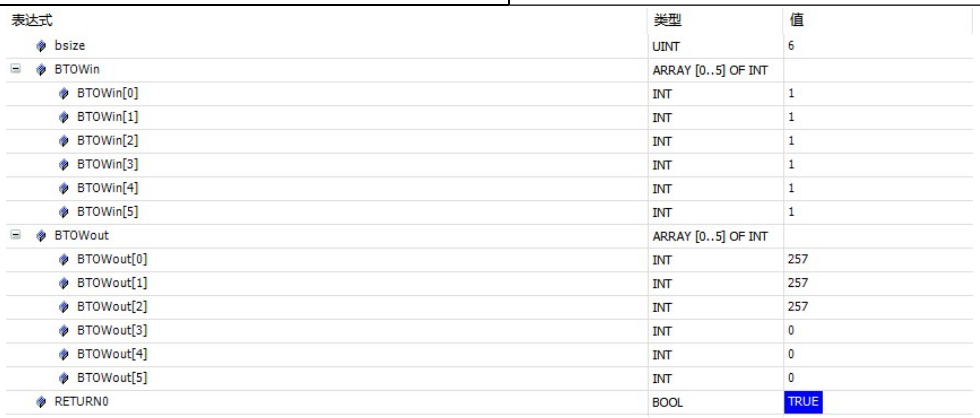
- Enter the source data in byDataSrc and the number of combined data in uiSize. The instruction combines the low 8 bits of multiple 16 bits data in the input source data to form a new 16 bits data. The newly generated data is saved to the starting address of pwDataDes, and the high 8 bits of pbyDataSrc are ignored.

(4) Note

- When the SIZE value is 0, the function returns a value of True.

(5) Sample program

DEMO	LD	ST

Define variables	VAR bsize:UINT; BTOWin:ARRAY[0..5] OF INT; BTOWout:ARRAY[0..5] OF INT; RETURN0: BOOL; END_VAR
Sample program	 BTOW(pbyDataSrc:=ADR(BTOWin) , uiSize:=bsize , PbyDataDes:=ADR(BTOWout));
Operation result	
Result analysis	Bsize represents the number of bytes, and the data is set to 6; Convert the two low octets of BTOWin [0] - BTOWin [5] in order, i.e. 01 data, into a new 16 bits data (0101) and store it in BTOWout [0] - BTOWout [2] in sequence.

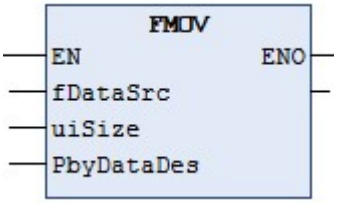
表达式	类型	值
bsize	UINT	6
BTOWin	ARRAY [0..5] OF INT	
BTOWin[0]	INT	1
BTOWin[1]	INT	1
BTOWin[2]	INT	1
BTOWin[3]	INT	1
BTOWin[4]	INT	1
BTOWin[5]	INT	1
BTOWout	ARRAY [0..5] OF INT	
BTOWout[0]	INT	257
BTOWout[1]	INT	257
BTOWout[2]	INT	257
BTOWout[3]	INT	0
BTOWout[4]	INT	0
BTOWout[5]	INT	0
RETURN0	BOOL	TRUE

1-4-4-4. One to many data transfer FMOV

(1) Instruction overview

Copy the source data array specified bit to multiple bits of the target array, such as transferring the value of A1 [1] to A2 [0,1,2].

One to many data transfer [FMOV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
FMOV	One to many data transfer		FMOV(fDataSrc:= , uiSize:= , byDataDes:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
fDataSrc	Starting address for data replication	REAL	As the data type	0	Starting address for data replication
uiSize	Number of variables	UINT	As the data type	0	Number of variables
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
PbyDataDes	Starting address for data storage	POINTER TO BYTE	-	-	Starting address for data storage

(3) Function

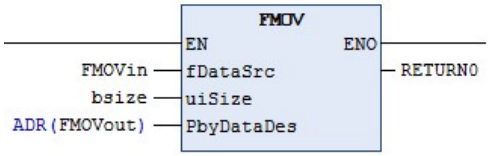
- Transfer the data from fDataSrc to uiSize units at the specified starting address of PbyDataDes, where the uiSize data range is from 0 to 1024.

(4) Note

- UiSize represents the number of transfers.

(5) Sample program

DEMO	LD	ST
Define variables	VAR FMOVin: REAL; FMOVout:ARRAY[0..10] OF REAL; bsize:UINT; RETURN0: BOOL; END_VAR	

Sample program		FMOV(fDataSrc:=FMOVin , uiSize:=bsize , PbyDataDes:=ADR(FMOVout));
Operation result		

表达式	类型	值
FMOVin	REAL	22
FMOVout	ARRAY [0..10] OF R...	
FMOVout[0]	REAL	22
FMOVout[1]	REAL	22
FMOVout[2]	REAL	22
FMOVout[3]	REAL	22
FMOVout[4]	REAL	22
FMOVout[5]	REAL	22
FMOVout[6]	REAL	0
FMOVout[7]	REAL	0
FMOVout[8]	REAL	0
FMOVout[9]	REAL	0
FMOVout[10]	REAL	0
bsize	UINT	6
RETURN0	BOOL	TRUE

1-4-4-5. ON bit SUM

(1) Instruction overview

Count the number of "1" in binary data, store the statistical results in SUM, and output them from the right side.

ON bit sum [SUM]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
SUM	Total ON bits	SUM EN ENO wData	SUM(wData:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
wData	Data that needs to be judged	WORD	As the data type	0	Data to be judged (decimal)

(3) Function

- Count the number of "1" in binary data, and store the statistical results in SUM.

(4) Sample program

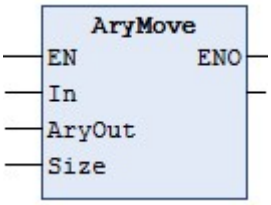
DEMO	LD	ST
Define variables	VAR Sumin: WORD; sumnum: INT; END_VAR	
Sample program	SUM EN ENO Sumin wData Sumnum	SUMnum:=SUM(wData:=SUMin);
Operation result	表达式 / 类型 / 值 Sumin / WORD / 22 sumnum / INT / 3	
Result analysis	Sumin data input 22 (binary display 10110), so Sumnum displays 3.	

1-4-4-6. Array transfer AryMove

(1) Instruction overview

Transfer multiple array elements.

Array transfer [AryMove]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryMove	Array transfer		AryMove(In:= , AryOut:= , Size:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	The array to be transferred	BOOL/BYTE/WORD /DWORD/LWORD/ USINT/UINT/UDINT /ULINT/SINT/INT/DINT /LINT/REAL/LREAL/TIME /DATE/TOD/DT/STRING	As the data type	-	The array to be transferred
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut	Transfer result array	BOOL/BYTE/WORD /DWORD/LWORD/ USINT/UINT/UDINT /ULINT/SINT/INT/DINT /LINT/REAL/LREAL/TIME /DATE/TOD/DT/STRING	As the data type	-	Transfer result array
Size	Number of elements to be transmitted	UINT	As the data type	0	Number of elements to be transmitted

(3) Function

- Transfer the "Size" elements starting from In[0] of the transfer object array In[] to the elements starting from AryOut[0] of the transfer result array AryOut[].

(4) Note

- The data types of In[] and AryOut[] elements should be consistent, otherwise the runtime function return value is FALSE and AryOut[] remains unchanged.
- When the value of "Size" is 0, the function returns a value of True and there is no change in AryOut[].
- The length of In[] should not be greater than the memory size of AryOut[] to avoid PLC exceptions.
- The value of "Size" should not exceed the array area of In[] and AryOut[] to avoid PLC exceptions.

(5) Sample program

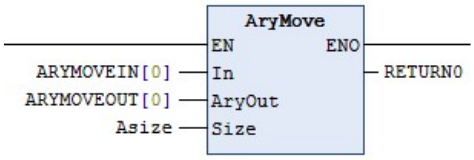
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR

    ARYMOVEIN:ARRAY[0..5] OF INT;
    ARYMOVEOUT:ARRAY[0..5] OF INT;
    Asize: UINT;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
AryMove(In:=ARYMOVEIN[0],
    AryOut:=ARYMOVEOUT[0] ,
    Size:=bsize );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
ARYMOVEIN	ARRAY [0..5] OF INT	
ARYMOVEIN[0]	INT	11
ARYMOVEIN[1]	INT	22
ARYMOVEIN[2]	INT	33
ARYMOVEIN[3]	INT	44
ARYMOVEIN[4]	INT	55
ARYMOVEIN[5]	INT	66
ARYMOVEOUT	ARRAY [0..5] OF INT	
ARYMOVEOUT[0]	INT	11
ARYMOVEOUT[1]	INT	22
ARYMOVEOUT[2]	INT	33
ARYMOVEOUT[3]	INT	44
ARYMOVEOUT[4]	INT	55
ARYMOVEOUT[5]	INT	66
Asize	UINT	6
RETURN0	BOOL	TRUE

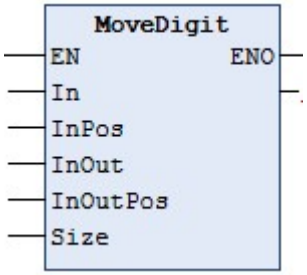
</td></tr>
</table>

1-4-4-7. Digit transfer MoveDigit

(1) Instruction overview

Transfer multiple digits in a sequence (1 digit is 4 bits).

Digit transfer [MoveDigit]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
MoveDigit	Digit transfer		MoveDigit(In:= , InPos:= , InOut:= , InOutPos:= , Size:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Sender	BYTE/WORD/ DWORD/LWORD	As the data type	-	Sender
Inpos	Number position of the sender	USINT	As the data type	0	The position of the number transmitted in "In"
InOut	Receiver	BYTE/WORD/ DWORD/LWORD	As the data type	-	Receiver
InOutPos	Receiver's numerical position	USINT	As the data type	0	The numerical position of the receiver in "Out"
Size	Number of digits to be transmitted	USINT	As the data type	1	Number of digits to be transmitted

(3) Function

- Transfer the "Size" digits starting from the number position "InPos" of the sender "In" to the number position "InOutPos" of the receiver "InOut", where one digit is 4 bits.

(4) Note

- If the specified receiver digit position exceeds the highest position of "InOut", the excess data will be saved to the lowest position of "InOut".
- If the specified number position of the sender exceeds the highest position of "In", the excess data will be transmitted to the lowest position of "In".
- When the value of "Size" is 0, the function returns a value of True and "InOut" remains unchanged.
- An exception will occur in the following situations. The return value of the function becomes FALSE, and "InOut" remains unchanged.
  - When the value of "InPos" exceeds the valid range.

- When the value of "InOutPos" exceeds the valid range.
- When the value of "Size" exceeds the valid range.

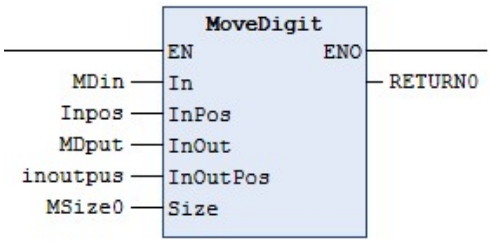
(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    MDin:WORD;
    MDput:WORD;
    Inpos: USINT;
    inoutpus: USINT;
    MSize0: USINT;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
MoveDigit(In:= MDin,
    InPos:= Inpos,
    InOut:=MDput,
    InOutPos:= inoutpus,
    Size:=MSize0 );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
MDin	WORD	16#1234
MDput	WORD	16#9934
Inpos	USINT	16#00
inoutpus	USINT	16#00
MSize0	USINT	16#02
RETURN0	BOOL	TRUE

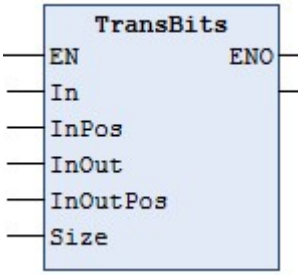
</td></tr>
</table>

1-4-4-8. Multiple bits transfer TransBits

(1) Instruction overview

Transfer multiple bits of a sequence.

Multiple bits transfer [TransBits]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
TransBits	Multiple bits transfer		TransBits(In:= , InPos:= , InOut:= , InOutPos:= , Size:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Sender	BYTE/WORD/ DWORD/LWORD	As the data type	-	Sender
Inpos	Bit position of the sender	USINT	0~'in' bits -1	-	The position of the transmitted bit in "In"
InOut	Receiver	BYTE/WORD/ DWORD/LWORD	As the data type	-	Receiver
InOutPos	Bit position of the receiver	USINT	0~'inout' bits -1	-	The position of receiver bit in "Out"
Size	Number of digits to be transmitted	USINT	0~'in' and 'inout' bits -1	-	Number of digits to be transmitted

(3) Function

- Transfer the "Size" bits starting from bit position "InPos" of the sender "In" to bit position "InOutPos" of the receiver "InOut".

(4) Note

- When the value of "Size" is 0, it is not transmitted and the function returns a value of True.
- The bits in "InOut" that are not related to transmission remain unchanged.
- In the following situations, an exception will occur, with the function returning a value of FALSE and InOut remaining unchanged.
  - When the value of "InPos" exceeds the valid range.
  - When the value of "InOutPos" exceeds the valid range.
  - When the value of "Size" exceeds the valid range.
  - When the specified values of "InPos" and "Size" exceed the number of digits of "In".
  - When the specified values of "InOutPos" and "Size" exceed the number of digits of "InOut".

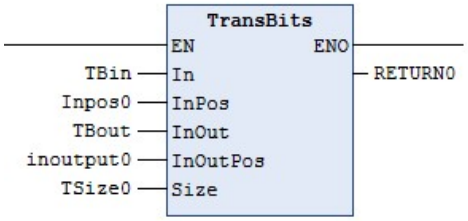
(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR

    RETURN0: BOOL;
    TBin:WORD;
    TBout:WORD;
    Inpos0: USINT;
    inoutput0: USINT;
    TSize0: USINT;

END_VAR
```

</td></tr>
<tr><td>Sample program</td><td> 
</td><td>

```
TransBits(In:= TBin,
    InPos:= Inpos0,
    InOut:=TBout ,
    InOutPos:=inoutput0 ,
    Size:=TSize0 );
```

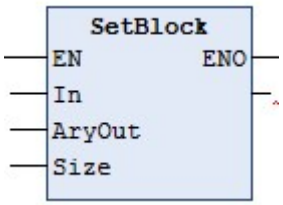
</td></tr>
<tr><td>Operation result</td><td colspan="2">
</td></tr>
<tr><td>Result analysis</td><td colspan="2">Set TBin (data set to 0), starting from input0 (data set to 0), two pieces of data are transferred to TBout (initial data bit 2#11111), starting from inoutpos0 (value 0), two pieces of data are transferred to TBout (TBout data is transferred to 2#11100), and the function returns a value of TRUE.</td></tr>
</table>

1-4-4-9. Set block SetBlock

(1) Instruction overview

Transfer the values of variables and constants to multiple array elements.

Set block [SetBlock]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
SetBlock	Set block		SetBlock(In:= , AryOut:= , Size:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Transmission source	BOOL/BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME/DATE/TOD/DT/STRING	As the data type	-	The array to be transferred
Size	Number of elements to be transmitted	UINT	As the data type	0	Number of elements to be transmitted
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut	Transfer target array	BOOL/BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME/DATE/TOD/DT/STRING	As the data type	-	Transfer result array

(3) Function

- Transfer the value of the transfer source "In" to "Size" positions starting from the AryOut[0] of the transfer target array AryOut[].

(4) Note

- When the data types of "In" and "AryOut" are different, the value returned by the function is FALSE, while "AryOut" remains unchanged.
- The area sizes of "In" and "AryOut" should be consistent to avoid PLC anomalies.
- When the value of "Size" is 0, the value returned by the function is true, and AryOut[] remains unchanged.
- The value of "Size" should not exceed the array area of AryOut[] to avoid PLC exceptions.

- When "In" and AryOut[] are unsupported data types, an exception will occur during compilation.

(5) Sample program

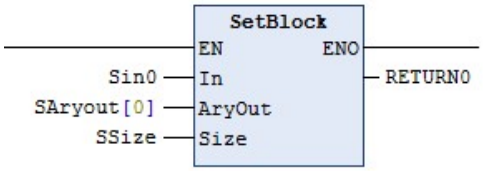
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR

    RETURN0: BOOL;
    Sin0:INT;
    SAryout:ARRAY[0..5] OF INT;
    SSize:UINT;

END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
SetBlock(In:= Sin0,
    AryOut:= SAryout[0],
    Size:=SSize );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
RETURN0	BOOL	TRUE
Sin0	INT	11
SAryout	ARRAY [0..5] OF INT	
SAryout[0]	INT	11
SAryout[1]	INT	11
SAryout[2]	INT	11
SAryout[3]	INT	11
SAryout[4]	INT	0
SAryout[5]	INT	0
SSize	UINT	4

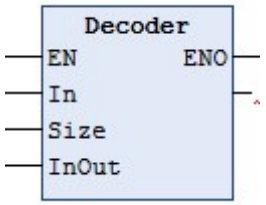
</td></tr>
</table>

1-4-4-10. Bit decoder Decoder

(1) Instruction overview

Set one of the arranged elements consisting of up to 256 bits as True and the other bits as FALSE.

Bit decoder [Decoder]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
Decoder	Bit decoder		Decoder(In:= , Size:= , InOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Conversion bit position	BYTE	As the data type	0	Conversion bit position
Size	Convert bits	USINT	0~8	1	Convert bits
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOut	Convert Object Array	BOOL/BYTE/WORD /DWORD/LWORD	As the data type	-	Convert Object Array

(3) Function

- Set the specified bit of the 2^"Size" bits starting from element InOut[0] in the transformation object array InOut[] to True, and set the other bits to FALSE.
- Among them, the bit position to be set as True is specified by the conversion bit position "In".
- The input parameters passed to InOut[] should be specified along with the element number, such as array[3].

(4) Note

- The value of "Size" should not exceed the number of digits of the "InOut" array element to avoid PLC exceptions.
- The value of "Size" should not exceed the valid range to avoid PLC anomalies.
- When the value of "Size" is 0, the function returns a value of True and InOut[] remains unchanged.
- InOut[] requires the use of Boolean and bitstring types, otherwise the program will compile an error at runtime.

(5) Sample program

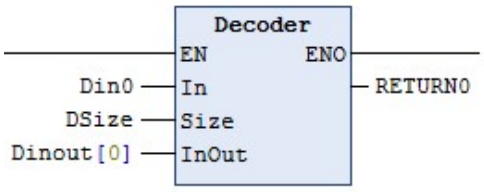
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR

    Din0: BYTE;
    DSize: USINT;
    DAryout:ARRAY[0..5] OF BYTE;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
Decoder(In:= Din0,
    Size:=DSize ,
    InOut:=Dinout[0] );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
Din0	BYTE	2#00000010
DSize	USINT	2#00000100
Dinout	ARRAY [0..5] OF BYTE	
Dinout[0]	BYTE	2#00000100
Dinout[1]	BYTE	2#00000000
Dinout[2]	BYTE	2#11111111
Dinout[3]	BYTE	2#11111111
Dinout[4]	BYTE	2#11111111
Dinout[5]	BYTE	2#11111111
RETURN0	BOOL	TRUE

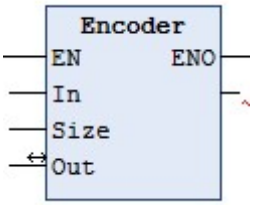
</td></tr>
<tr><td>Result analysis</td><td colspan="2">Set the Dinout array data to 2#11111111; Run the command to set the 2^4(DSize) bits starting from Dinout[0] data to 0, except bit2(Din0) data which is set to 1.</td></tr>
</table>

1-4-4-11. Bit encoder Encoder

(1) Instruction overview

Calculate the bit position with a value of True in a permutation element composed of up to 256 bits.

Bit decoder [Encoder]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
Encoder	Bit encoder		Encoder(In:= , Size:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Convert object arrangement	BOOL/BYTE/WORD /DWORD/LWORD	As the data type	-	Convert object arrangement
Size	Convert bits	USINT	0~8	1	Convert bits
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut	Convert result	BYTE	As the data type	0	Convert result

(3) Function

- Calculate the bit position with a value of True within any range of the transformation object arrangement In[].
- The range for calculating bit positions is 2^"Size" bits starting from In[0]. The positions of the bits with a value of True within this range are represented in binary and saved in the conversion result "Out". Then, save FALSE in the remaining bits of "Out".
- If there are multiple bits with a value of True within the specified range, calculate the position of the highest bit among them.
- The input parameters passed to In[] should be specified together with the element number, such as Array[3].
- When "Size"=USINT#4 and In[] is a BYTE arrangement, since "Size"=USINT#4, the object range for calculating the TRUE bit position is 2^4=16 starting from In[0].

(4) Note

- When the value of "Size" is 0, the function returns true and "Out" remains unchanged.
- When "Size" exceeds the valid range, avoid PLC anomalies.
- When the value of "Size" exceeds the number of digits of In[] arranged elements, an incorrect value may be obtained.
- When all the values specified in "Size" in the In[] bit are FALSE, the program encounters an exception and the function returns a value of FALSE, while the value of "Out" remains unchanged.

- In[] requires the use of Boolean and bit string types, otherwise the program compilation will report an error at runtime.

(5) Sample program

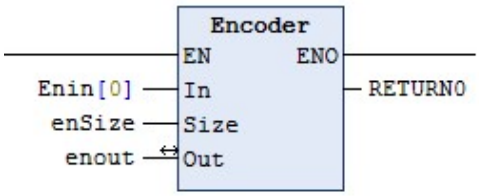
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR

    Enin:ARRAY[0..5] OF BYTE;
    enSize: USINT;
    enout: BYTE;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
Encoder(In:=enin[0] ,
    Size:=enSize ,
    Out:=enout );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
Enin	ARRAY [0..5] OF BYTE	
Enin[0]	BYTE	2#11111111
Enin[1]	BYTE	2#11111111
Enin[2]	BYTE	2#11111111
Enin[3]	BYTE	2#11111111
Enin[4]	BYTE	2#11111111
Enin[5]	BYTE	2#11111111
enSize	USINT	2#00000100
enout	BYTE	2#00001111
RETURN0	BOOL	TRUE

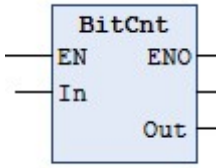
</td></tr>
</table>

1-4-4-12. Bit count BitCnt

(1) Instruction overview

Calculate the total number of bits in a string with a value of True.

Bit count [BitCnt]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
BitCnt	Bit count		BitCnt(In:= , Out=>);

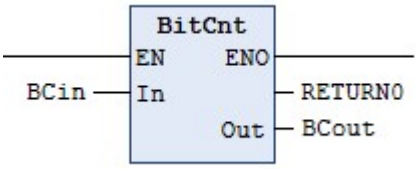
(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Count object	BYTE/WORD/ DWORD/LWORD	As the data type	-	Count object
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Out	Calculate the bits	USINT	As the data type	0	Count the number of bits in the object that are true

(3) Function

- Calculate the total number of bits in the count object "In" where the value is true.

(4) Sample program

DEMO	LD	ST
Define variables		VAR BCin: WORD; BCout: USINT; RETURN0: BOOL; END_VAR
Sample program		BitCnt(In:=BCin , Out=> BCout);

Operation result	

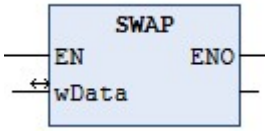
表达式	类型	值
BCin	WORD	17
BCout	USINT	2
RETURN0	BOOL	TRUE

1-4-4-13. High low byte exchange SWAP

(1) Instruction overview

Swap the high and low 8 bits of 16-bit type data.

High low byte exchange [SWAP]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
SWAP	High low byte exchange		SWAP(wData:=);

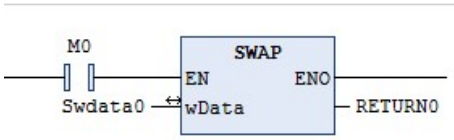
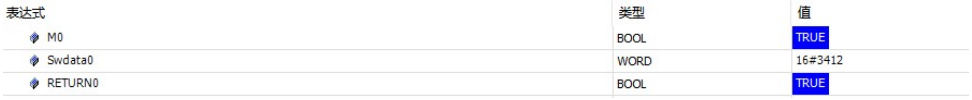
(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
wData	Data that needs to be operated	Word	As the data type	-	Data that needs to be operated

(3) Function

- Swap the high and low 8 bits of 16-bit type data.

(4) Sample program

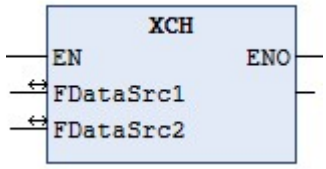
DEMO	LD	ST
Define variables	VAR M0:BOOL; Swdata0: WORD; RETURN0: BOOL; END_VAR	
Sample program		IF M0 THEN SWAP(wData:=Swdata0); END_IF
Operation result		
Result analysis	M0 is set to True, and the high and low bits of Swdata0 will be converted from 16#1234 to 16#3412; M0 is constantly triggered, and the high and low positions will continue to transition.	

1-4-4-14. Data exchange XCH

(1) Instruction overview

Swap two floating-point data.

Data exchange [XCH]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
XCH	Data exchange		`XCH(FDataSrc1:= , FDataSrc2:= );`

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
FDataSrc1	Data1 to be exchanged	REAL	As the data type	0	Data to be exchanged
FDataSrc1	Data2 to be exchanged	REAL	As the data type	0	Data to be exchanged

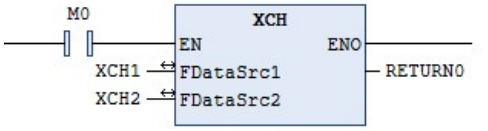
(3) Function

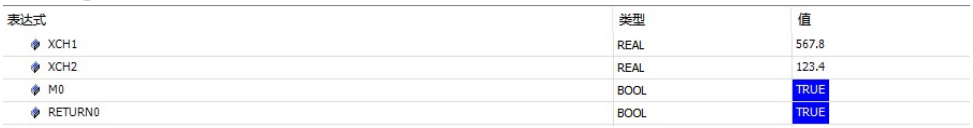
- Swap two floating-point data.

(4) Sample program

DEMO	LD	ST

Define variables		`VAR`
		`XCH1: REAL;`
		`XCH2: REAL;`
		`M0:BOOL;`
		`RETURN0: BOOL;`
		`END_VAR`

Sample program		`IF M0 THEN`
		`XCH(FDataSrc1:= XCH1,`
		`FDataSrc2:=XCH2 );`
		`END_IF`

Operation result	Before operation After operation 	

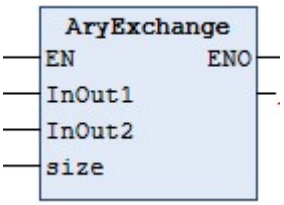
Result analysis	Set M0 to True, the data of XCH1 and XCH2 will be exchanged, with data 123.4 in XCH1 being exchanged for data 567.8 in XCH2; M0 is constantly triggered, and XCH1 and XCH2 will continue to exchange.

1-4-4-15. Array data exchange AryExchange

(1) Instruction overview

Swap the elements of two arrays.

Array data exchange [AryExchange]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryExchange	Array data exchange		AryExchange(InOut1:= , InOut2:= , size:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	The array to be transferred	BOOL/BYTE/WORD/ DWORD/LWORD/ USINT/UINT/UDINT/ ULINT/SINT/INT/DINT /LINT/REAL/LREAL TIME/DATE/TOD/DT/ STRING	As the data type	-	The array to be transferred
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut	Transfer result array	BOOL/BYTE/WORD/ DWORD/LWORD/ USINT/UINT/UDINT/ ULINT/SINT/INT/DINT /LINT/REAL/LREAL TIME/DATE/TOD/DT/ STRING	As the data type	-	Transfer result array
Size	Number of elements to be transmitted	UINT	As the data type	0	Number of elements to be transmitted

(3) Function

- Exchange the values of "InOut1" and "InOut2".
- Swap the "Size" elements starting from InOut1[0] in the swappable object array InOut1[] with the "Size" elements starting from InOut2[0] in InOut2[].

(4) Note

- The data types of InOut1[] and InOut2[] elements should be consistent, otherwise the function return value is FALSE, and InOut1[] and InOut2[] remain unchanged.
- When the value of "Size" is 0, the function returns a value of True, and InOut1[] and InOut2[] remain unchanged.
- The InOut1[] and InOut2[] elements are STRING type, and the string length of one should not be greater than the memory size of the other to avoid PLC exceptions.
- The value of "Size" should not exceed the array area of InOut1[] or InOut2[] to avoid PLC exceptions.

(5) Sample program

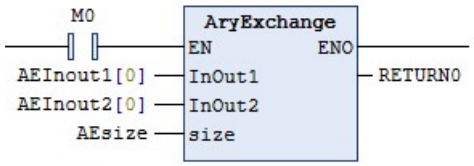
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR

    AEInout1:ARRAY[0..5] OF BYTE;
    AEInout2:ARRAY[0..5] OF BYTE;
    AEsize: UINT;
    M0:BOOL;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
IF M0 THEN
    AryExchange(InOut1:= AEInout1[0],
        InOut2:= AEInout2[0],
        size:=AEsize );
END_IF
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">
Before operation

表达式	类型	值
AEInout1	ARRAY [0..5] OF BYTE	
AEInout1[0]	BYTE	11
AEInout1[1]	BYTE	22
AEInout1[2]	BYTE	33
AEInout1[3]	BYTE	44
AEInout1[4]	BYTE	55
AEInout1[5]	BYTE	66
AEInout2	ARRAY [0..5] OF BYTE	
AEInout2[0]	BYTE	77
AEInout2[1]	BYTE	88
AEInout2[2]	BYTE	99
AEInout2[3]	BYTE	123
AEInout2[4]	BYTE	234
AEInout2[5]	BYTE	245
AEsize	UINT	5
M0	BOOL	FALSE

After operation

表达式	类型	值
AEInout1	ARRAY [0..5] OF BYTE	
AEInout1[0]	BYTE	77
AEInout1[1]	BYTE	88
AEInout1[2]	BYTE	99
AEInout1[3]	BYTE	123
AEInout1[4]	BYTE	234
AEInout1[5]	BYTE	66
AEInout2	ARRAY [0..5] OF BYTE	
AEInout2[0]	BYTE	11
AEInout2[1]	BYTE	22
AEInout2[2]	BYTE	33
AEInout2[3]	BYTE	44
AEInout2[4]	BYTE	55
AEInout2[5]	BYTE	245
AEsize	UINT	5
M0	BOOL	TRUE
RETURN0	BOOL	TRUE

</td></tr>
</table>

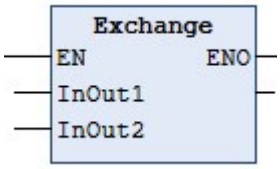
Result analysis	Set M0 to True, and the AEsize data of AEInOut1 and AEinout2 will be exchanged; M0 is constantly triggered, and XCH1 and XCH2 will continue to exchange.

1-4-4-16. Data exchange Exchange

(1) Instruction overview

Swap the values of two variables.

Data exchange [Exchange]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
Exchange	Data exchange		Exchange(InOut1:= , InOut2:=);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOut1	Exchange Object 1	BOOL/BYTE/WORD/ DWORD/LWORD/USINT/UINT/ UDINT/ULINT/SINT/INT/DINT /LINT/REAL/LREAL/TIME/DATE/ TOD/DT/STRING	As the data type	0	Exchange Object 1
InOut2	Exchange Object 2	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/LINT/REAL/ LREAL/TIME/DATE/TOD/DT/STRING	As the data type	0	Exchange Object 2

(3) Function

- Exchange the values of "InOut1" and "InOut2".

(4) Note

- The data types of "InOut1" and "InOut2" should be consistent, otherwise the runtime function returns a value of False.
- When using this instruction in the ST program, the return value "Out" is not used.
- "InOut1" and "InOut2" are STRING types. If the length of any string does not match the size of the other party, it will not be swapped.

(5) Sample program

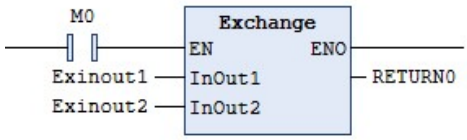
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR

    Exinout1: WORD;
    Exinout2: WORD;
    M0:BOOL;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
IF M0 THEN
    Exchange(InOut1:=Exinout1,
        InOut2:=Exinout2 );
END_IF
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">
Before operation

表达式	类型	值
Exinout1	WORD	123
Exinout2	WORD	456
M0	BOOL	FALSE
RETURN0	BOOL	FALSE

After operation

表达式	类型	值
Exinout1	WORD	456
Exinout2	WORD	123
M0	BOOL	TRUE
RETURN0	BOOL	TRUE

</td></tr>
<tr><td>Result analysis</td><td colspan="2">Set M0 to True, and the data of ExInOut1 and Exinout2 will be exchanged; M0 is constantly triggered, and ExInOut1 and Exinout2 will always switch.</td></tr>
</table>

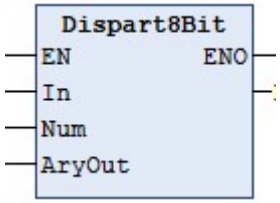
1-4-5. Data separation and combination instructions

1-4-5-1. Byte unit data separation Dispart8Bit

(1) Instruction overview

Separate bit strings in units of 1 byte.

Byte unit data separation [Dispart8Bit]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
Dispart8Bit	Byte unit data separation		bResult:=Dispart8Bit(In:= , Num:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Separate objects	BYTE/WORD/ DWORD/LWORD	As the data type	-	Bit string of separate object
Num	Separate bytes	USINT	As the data type	1	Number of bytes to be separated
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Separate result array	BYTE	As the data type	-	Separate result array

Note: The input and output variable AryOut requires specifying array elements.

(3) Function

- Separate the separated object "In" in units of 1 byte and save it in the separation result array AryOut[ ].
- Firstly, separate "In" in units of 1 byte. Then, save the lowest byte in the starting address of AryOut[] set on pin AryOut. Save the next byte in the AryOut[]+1 address. Perform the same processing on the number of separated bytes "Num".

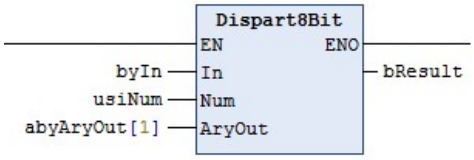
(4) Note

- When the value of "Num" exceeds the valid range, the return value of the function is FALSE, and AryOut[] remains unchanged.
- The value of "Num" should not exceed the array area of AryOut[].

(5) Sample program

DEMO	LD	ST
Define variables		
Sample program		
Operation result		

```
VAR
    byIn      :LWORD:=16#FFFF00FF;
    usiNum    :USINT:=4;
    abyAryOut:ARRAY[0..4] OF BYTE:=[0,0,0,0,0];
    bResult   :BOOL;
END_VAR
```




```
1  bResult:=Dispart8Bit(In:= byIn,
2      Num:= usiNum,
3      AryOut:= abyAryOut[1]);
```

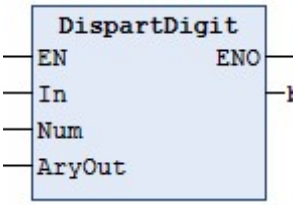
表达式	类型	值
byIn	LWORD	16#00000000FFFF00FF
usiNum	USINT	16#04
abyAryOut	ARRAY [0..4] OF BYTE	
abyAryOut[0]	BYTE	16#00
abyAryOut[1]	BYTE	16#FF
abyAryOut[2]	BYTE	16#00
abyAryOut[3]	BYTE	16#FF
abyAryOut[4]	BYTE	16#FF
bResult	BOOL	TRUE

1-4-5-2. 4-bit separate DispartDigit

(1) Instruction overview

Separate bit strings in units of 4 bits.

4-bit separate [DispartDigit]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
DispartDigit	4-bit separate		bResult:=DispartDigit(In:= , Num:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Separate objects	BYTE/WORD/ DWORD/LWORD	As the data type	-	Bit strings of separate object
Num	Separate bits	USINT	As the data type	1	Number of digits to be separated
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AryOut[] array	Separate result array	BYTE	As the data type	-	Separate result array

Note: The input and output variable AryOut requires specifying array elements.

(3) Function

- Separate the "In" of the separated object in units of 4 digits (digital units) and save it in the separation result array AryOut[].
- Firstly, separate "In" in units of 4 digits. Then, save the lowest 4 bits in the starting address of the AryOut[] set on pin AryOut. At this point, add 16#0 in bits 4 to 7 of the BYTE type AryOut[].
- Perform the same treatment on the number of separation digits "Num".

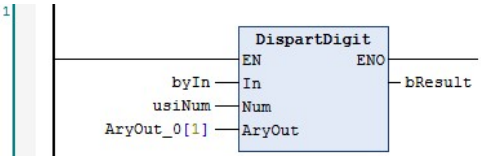
(4) Note

- When the value of "Num" is 0, the function returns a value of True, and the value of AryOut[] remains unchanged.
- When the value of "Num" exceeds the valid range, the return value of the function is FALSE, and AryOut[] remains unchanged.
- The value of "Num" should not exceed the array area of AryOut[].

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    byIn:LWORD:=16#FF00FF;
    usiNum:USINT:=3;
    AryOut_0:ARRAY[0..5] OF BYTE:=[0,0,0,0,0,0];
    bResult:BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>


</td><td>

```
1  bResult:=DispartDigit(In:= byIn,
2      Num:= usiNum,
3      AryOut:= AryOut_0[1]);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
byIn	LWORD	16#0000000000FF00FF
usiNum	USINT	16#03
AryOut_0	ARRAY [0..5] OF BYTE	
AryOut_0[0]	BYTE	16#00
AryOut_0[1]	BYTE	16#0F
AryOut_0[2]	BYTE	16#0F
AryOut_0[3]	BYTE	16#00
AryOut_0[4]	BYTE	16#00
AryOut_0[5]	BYTE	16#00
bResult	BOOL	TRUE

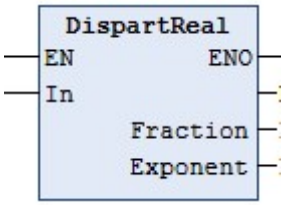
</td></tr>
</table>

1-4-5-3. Mantissa of real numbers, exponential separation DispartReal

(1) Instruction overview

Decompose real numbers into signed mantissa parts and exponential parts.

Mantissa of real numbers, exponential separation [DispartReal]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
DispartReal	Mantissa of real numbers, exponential separation		bResult:=DispartReal(In:= , Fraction=> , Exponent=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Real number	REAL/LREAL	As the data type	-	Real number to be separated
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Fraction	Signed mantissa part	LINT	As the data type	0	Signed mantissa part
Exponent	Exponential part	INT	As the data type	0	Exponential part

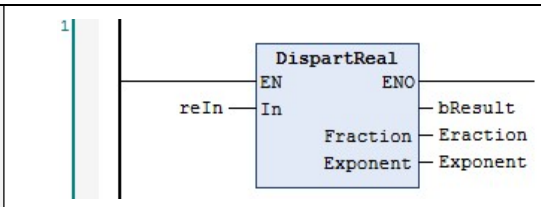
(3) Function

- Decompose the real number "In" into a signed mantissa part "Fraction" and an exponential part "Exponent".
- When "In" is a REAL type, "Fraction" is a 7-bit integer. When "In" is LREAL type, "Fraction" is a 15 bits integer. The effective range of "Fraction" varies depending on the data type of "In".

“In” data type	“Fraction” effective range
REAL	-9999999～9999999
LREAL	-99999999999999～99999999999999

(4) Sample program

DEMO	LD	ST
Define variables		VAR reIn:REAL:=12.34567E+8; Eraction:LINT; Exponent:INT; bResult:BOOL; END_VAR

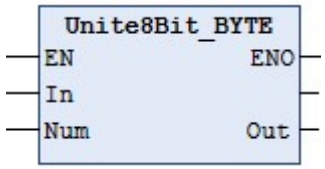
Sample program		1 bResult:=DispartReal(In:= reIn, 2 Fraction=> Eraction, 3 Exponent=> Exponent);
Operation result		

1-4-5-4. Byte unit data combination group Unite8Bit_XXX

(1) Instruction overview

Combine data in units of 1 byte into a bit string.

Byte unit data combination group [Unite8Bit_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
Unite8Bit_XXX	Byte unit data combination group		bResult:=Unite8Bit_BYTE(In:= , Num:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In[] array	Combining Object Array	BYTE	As the data type	-	Array to be combined
Num	Combining bytes	USINT	As the data type	1	Bytes to be combined
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Combining result	BYTE/WORD/ DWORD/LWORD	As the data type	-	Bit string of combined results

Note: The input variable In needs to pass in the specified element of the array.

(3) Function

- Combine the elements of the combination object array In[] to generate the bit string "Out" of the combination result.
- The number of array elements to be combined is specified by the combined byte count "Num". Firstly,

combine In[] with consecutive "Num" bytes to generate a bit string of "Num" bytes. On its high bit, take the 16#00 bit string of the combination {(bytes of "Out") - "Num"} as "Out".

- The instruction name varies depending on the data type of "Out". For example, when "Out" is DWORD type, the instruction name is Unite8Bit_DWORD.

(4) Note

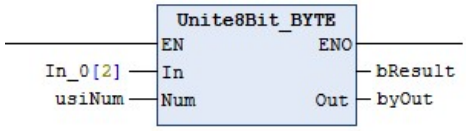
- When the value of "Num" is 0, the value of "Out" is initialized to 0, and the function returns a value of True.
- When the value of "Num" exceeds the valid range, the function returns a value of FALSE and initializes the value of "Out" to 0.
- The value of "Num" cannot exceed the array range of In[] and cannot exceed the range of "Out".
- The input needs to pass in the specified elements of the array, and the array length cannot be obtained. Otherwise, the function returns False.

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    In_0     :ARRAY[1..10] OF BYTE:=[10(1)];
    usiNum   :USINT:=1;
    byOut    :BYTE;
    bResult  :BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
1  bResult:=Unite8Bit_BYTE(In:= In_0[2],
2      Num:= usiNum,
3      Out=> byOut);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
In_0	ARRAY [1..10] OF BYTE	
In_0[1]	BYTE	16#01
In_0[2]	BYTE	16#01
In_0[3]	BYTE	16#01
In_0[4]	BYTE	16#01
In_0[5]	BYTE	16#01
In_0[6]	BYTE	16#01
In_0[7]	BYTE	16#01
In_0[8]	BYTE	16#01
In_0[9]	BYTE	16#01
In_0[10]	BYTE	16#01
usiNum	USINT	16#01
byOut	BYTE	16#01
bResult	BOOL	TRUE

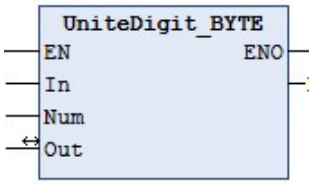
</td></tr>
</table>

1-4-5-5. 4-digit combination group UniteDigit_XXX

(1) Instruction overview

Combine 4-bit data into a bit string.

4-digit combination group [UniteDigit_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
UniteDigit_XXX	4-digit combination group		bResult:=UniteDigit_BYTE(In:= , Num:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In[] array	Combining Object Array	BYTE	As the data type	-	Array to be combined
Num	Combining bits	USINT	As the data type	1	Bits to be combined
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Combining result	BYTE/WORD/ DWORD/LWORD	As the data type	-	Bit string of combined results

(3) Function

- Combine the elements of the combination object array In[] by the low 4 bits (1 digit=4 bits) to generate the bit string "Out" of the combination result.
- The number of array elements to be combined is specified by the combination digit "Num". Firstly, combine each low 4 bits of the consecutive "Num" starting from In[] to generate a bit string of "Num" bits. At its high position, take the 16#0 bit string of the combination {(number of "Out" bits) - "Num"} as "Out".
- The instruction name varies depending on the data type of "Out". For example, when "Out" is WORD type, the instruction name is UniteDigit_WORD.

(4) Note

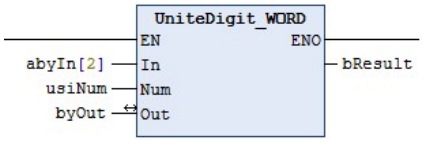
- When the value of "Num" is 0, the value of "Out" is 0, and the function returns a value of True.
- When the value of "Num" exceeds the valid range, the function returns a value of FALSE and "Out" remains unchanged.
- The value of "Num" cannot exceed the array range of In[] and cannot exceed the range of "Out".
- Input In requires passing in the specified elements of the array, and the array length cannot be obtained. Otherwise, the function returns False.

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    abyIn:ARRAY[1..10] OF BYTE:=[10(1)];
    usiNum:USINT:=2;
    byOut:WORD;
    bResult:BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
1  bResult:=UniteDigit_WORD(In:= abyIn[2],
2      Num:= usiNum,
3      Out:= byOut);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
abyIn	ARRAY [1..10] OF BYTE	
abyIn[1]	BYTE	16#01
abyIn[2]	BYTE	16#01
abyIn[3]	BYTE	16#01
abyIn[4]	BYTE	16#01
abyIn[5]	BYTE	16#01
abyIn[6]	BYTE	16#01
abyIn[7]	BYTE	16#01
abyIn[8]	BYTE	16#01
abyIn[9]	BYTE	16#01
abyIn[10]	BYTE	16#01
usiNum	USINT	16#02
byOut	WORD	16#0011
bResult	BOOL	TRUE

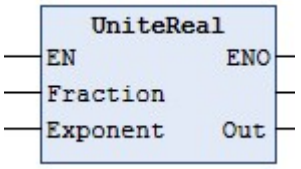
</td></tr>
</table>

1-4-5-6. Combine mantissa and exponents into real numbers UniteReal

(1) Instruction overview

Combine the signed mantissa and exponential parts to obtain real numbers.

Combine mantissa and exponents into real numbers [UniteReal]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
UniteReal	Combine mantissa and exponents into real numbers		bResult:=UniteReal(Fraction:= , Exponent:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Fraction	Signed mantissa part	DINT/LINT	As the data type	-	Signed mantissa part
Exponent	Exponential part	INT	As the data type	0	Exponential part
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Real number	LREAL	As the data type	0	Real numbers obtained by combining

Note: When the data type of "Fraction" is DINT, the data type of "Out" is REAL; When the data type of "Fraction" is LINT, the data type of "Out" is LREAL.

(3) Function

- Combine the signed mantissa part "Fraction" and the exponential part "Exponent" to obtain the real number "Out".

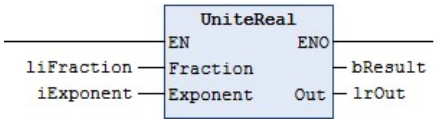
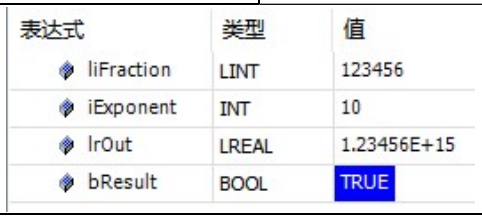
(4) Note

- Due to the different values of "Fraction" and "Exponent", there may be errors when converting integers to real numbers.
- When the combination result exceeds the valid range of "Out", if "Exponent" is a positive number, then the value of "Out" is infinite with the same sign as "Fraction". If "Exponent" is negative, then the value of "Out" is 0.
- "Fraction" requires the use of Dint and Lint types. When the type is determined to be outside the integer type, a compilation error is reported. Otherwise, the runtime function returns a value of FALSE, and the value of "Out" is 0.

(5) Sample program

DEMO	LD	ST

Define variables		VAR
		liFraction:LINT:=123456;
		iExponent :INT:=10;
		lrOut :LREAL;
		bResult :BOOL;
		END_VAR

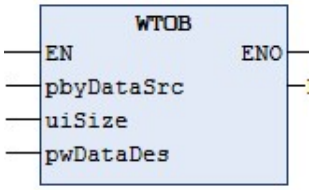
Sample program		1 bResult:=UniteReal(Fraction:= liFraction, 2 Exponent:= iExponent, 3 Out=> lrOut);
Operation result		

1-4-5-7. Byte unit data separation WTOB

(1) Instruction overview

Separate continuous 16 bits data in byte units.

Byte unit data separation [WTOB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
WTOB	Byte unit data separation		bResult:=WTOB(pbyDataSrc:= , uiSize:= , pwDataDes:=);

(2) Related variable

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
pbyDataSrc	Starting address for separating data	POINTER_TO_BYTE	As the data type	-	Starting address for separating data
uiSize	Number of separated data	UINT	As the data type	0	Number of separated data
pwDataDes	The final storage destination address	POINTER_TO_WORD	As the data type	-	The final storage destination address of separating data

(3) Function description

- Save the 16 bits data starting from the pbyDataSrc address, in bytes, to the low 8 bits of the uiSize*2 starting from the pwDataDes address, and the high 8 bits to 16#00.

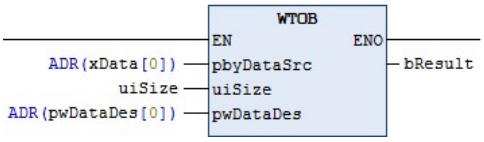
(4) Note

- When the SIZE value is 0, the function returns a value of True.

(5) Program example

DEMO	LD	ST
Define variables		
Sample program		
Operation result		

```
VAR
    bResult:BOOL;
    xData:ARRAY[0..8] OF BYTE:=[9(16#11)];
    uiSize:UINT:=3;
    pwDataDes:ARRAY[0..10] OF WORD;
END_VAR
```




```
1  bResult:=WTOB(pbyDataSrc:= ADR(xData[0]),
2      uiSize:= uiSize,
3      pwDataDes:= ADR(pwDataDes[0]));
```

表达式	类型	值
bResult	BOOL	TRUE
xData	ARRAY [0..8] OF BYTE	
xData[0]	BYTE	16#11
xData[1]	BYTE	16#11
xData[2]	BYTE	16#11
xData[3]	BYTE	16#11
xData[4]	BYTE	16#11
xData[5]	BYTE	16#11
xData[6]	BYTE	16#11
xData[7]	BYTE	16#11
xData[8]	BYTE	16#11
uiSize	UINT	16#0003
pwDataDes	ARRAY [0..10] OF WORD	
pwDataDes[0]	WORD	16#0011
pwDataDes[1]	WORD	16#0011
pwDataDes[2]	WORD	16#0011
pwDataDes[3]	WORD	16#0011
pwDataDes[4]	WORD	16#0011
pwDataDes[5]	WORD	16#0011
pwDataDes[6]	WORD	16#0000
pwDataDes[7]	WORD	16#0000
pwDataDes[8]	WORD	16#0000
pwDataDes[9]	WORD	16#0000
pwDataDes[10]	WORD	16#0000

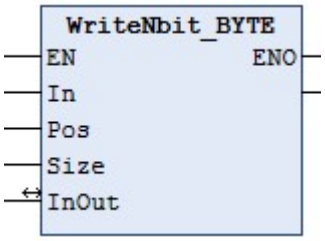
1-4-6. Data read and write instructions

1-4-6-1. N bits write to group WriteNbit_XXX

(1) Instruction overview

Write multiple bits into a bit train.

N bits write to group [WriteNbit_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
WriteNbit_XXX	N bits write to group		WriteNbit_BYTE(In:= , Pos:= , Size:= , InOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Read source	BYTE/WORD/ DWORD/LWORD	As the data type	-	This bit train is used to read the value to be written to "InOut"
Pos	Write in position	USINT	As the data type	0	The position of the bit to be written
Size	Write in size	USINT	As the data type	0	The number of bits to be written
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
InOut	Write in object	Same as "In" type	As the data type	-	Write in the result

(3) Function

- Firstly, read out the low bit "Size" of the source "In". Then, write the read value to the write position "Pos" of the write object "InOut".
- The instruction name varies depending on the data type of "In" and "InOut". For example, when "In" and "InOut" are BYTE types, the instruction name type is WriteNbit_BYTE.

(4) Note

- Please set the data types of "In" and "InOut" to be the same.
- When the value of "Size" is 0, the value of "InOut" remains unchanged, and the function returns a value of True.
- When using this instruction in the ST program, the return value "Out" is not used.
- When "Size" and "Pos" exceed the valid range, the program encounters an exception, and the function returns a value of FALSE, while the value of "InOut" remains unchanged.

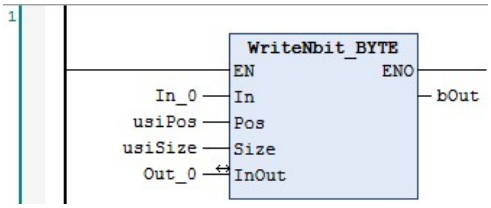
- When the number of bits in "InOut" is insufficient from the position shown in "Pos" to the position shown in "Size", the program encounters an exception, and the function returns a value of FALSE, while the value of "InOut" remains unchanged.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    In_0   :BYTE:=16#76;
    usiPos :USINT:=1;
    usiSize:USINT:=4;
    Out_0  :BYTE;
    bOut   :BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



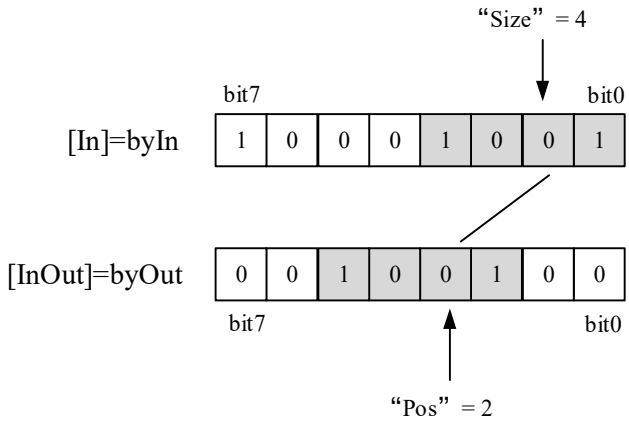
</td><td>

```
1  WriteNbit_BYTE(In:= In_0,
2      Pos:= usiPos,
3      Size:= usiSize,
4      InOut:= Out_0);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
In_0	BYTE	16#76
usiPos	USINT	16#01
usiSize	USINT	16#04
Out_0	BYTE	16#0C
bOut	BOOL	TRUE

</td></tr>
<tr><td>Principle analysis</td><td colspan="2">



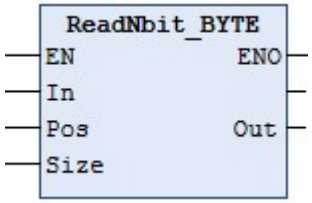
</td></tr>
</table>

1-4-6-2. N bits read group ReadNbit_XXX

(1) Instruction overview

Read multiple bits in the bit train.

N bits read group [ReadNbit_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ReadNbit_XXX	N bits read group		ReadNbit_BYTE(In:= , Pos:= , Size:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Read source	BYTE/WORD/ DWORD/LWORD	As the data type	-	This bit train is used to read the value to be written to "InOut"
Pos	Read position	USINT	As the data type	0	To read the position of bit
Size	Read size	USINT	As the data type	0	The number of bits to be read
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Read result	Same as "In" type	As the data type	-	Read result

(3) Function

- Substitute the "Size" bits starting from the reading position "Pos" of the source bit column "In" into the reading result "Out".
- The instruction name varies depending on the data type of "In" and "Out". For example, when "In" and "Out" are BYTE types, the instruction name is ReadNbit_BYTE.

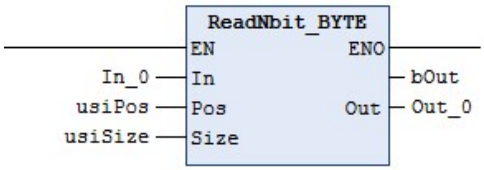
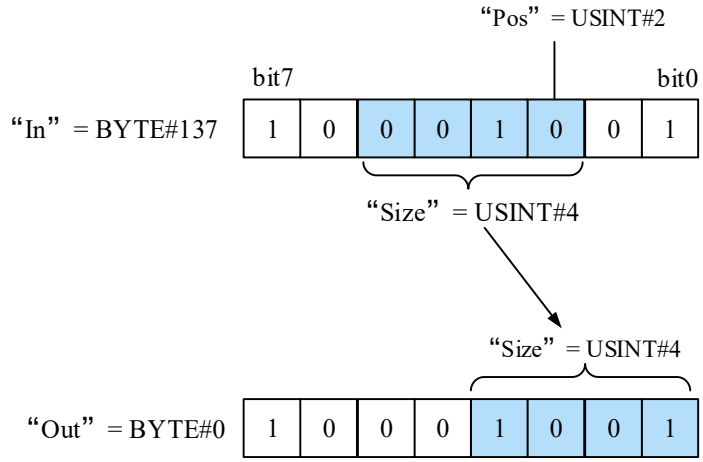
(4) Note

- It is recommended to unify the data types for "In" and "Out".
- When the value of "Size" is 0, the value of "Out" is 16#0.
- In the following situations, an exception will occur, with the function returning a value of FALSE and the value of "Out" being 16#0.
  - When the value of "Pos" exceeds the valid range.
  - When the value of "Size" exceeds the valid range.
  - When the specified values of "Pos" and "Size" exceed the number of digits of "In".

(5) Program example

DEMO	LD	ST

Define variables		VAR
		In_0 :BYTE:=16#76;
		usiPos :USINT:=1;
		usiSize:USINT:=4;
		Out_0 :BYTE;
		bOut :BOOL;
		END_VAR

Sample program		1 ReadNbit_BYTE(In:= In_0, 2 Pos:= usiPos, 3 Size:= usiSize, 4 Out=> Out_0);
Operation result		
Principle analysis		

1-4-7. Data conversion instructions

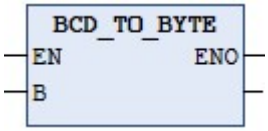
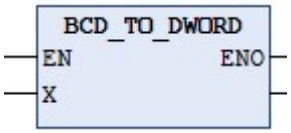
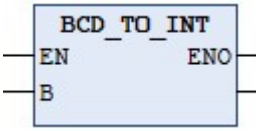
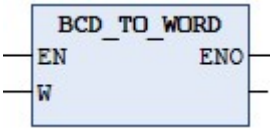
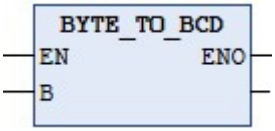
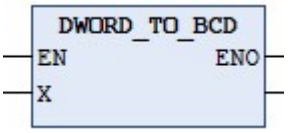
- It is strictly prohibited to implicitly convert "larger" data types to "smaller" data types for use, as there is a risk of losing information when transitioning from larger data types to smaller data types.
- If the converted value exceeds the storage range of the target data type, the high byte of this number will be ignored. Example: Convert INT type to BYTE type or DINT type to WORD type.

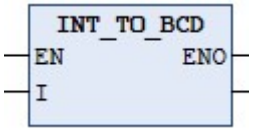
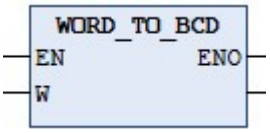
1-4-7-1. BCD code and integer data conversion

(1) Instruction overview

BCD code is converted to BYTE, DWORD, INT, and WORD data.

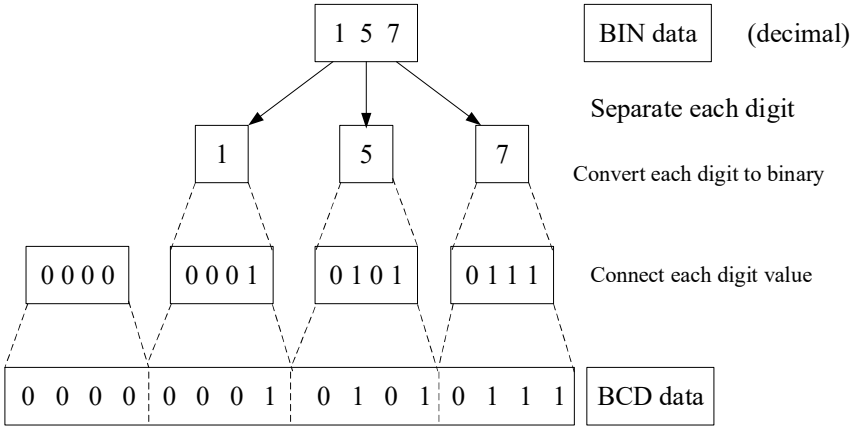
BCD code and integer data conversion [BCD_TO_XXX/XXX_TO_BCD]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	Util		

Instruction	Name	Graph	ST
BCD_TO_BYTE	BCD convert to BYTE		`BCD_TO_BYTE(B:= );`
BCD_TO_DWORD	BCD convert to DWORD		`BCD_TO_DWORD(X:= );`
BCD_TO_INT	BCD convert to INT		`BCD_TO_INT(B:= );`
BCD_TO_WORD	BCD convert to WORD		`BCD_TO_WORD(W:= );`
BYTE_TO_BCD	BYTE convert to BCD		`BYTE_TO_BCD(B:= );`
DWORD_TO_BCD	DWORD convert to BCD		`DWORD_TO_BCD(X:= );`

Instruction	Name	Graph	ST
INT_TO_BCD	INT convert to BCD		INT_TO_BCD(I:=);
WORD_TO_BCD	WORD convert to BCD		WORD_TO_BCD(W:=);

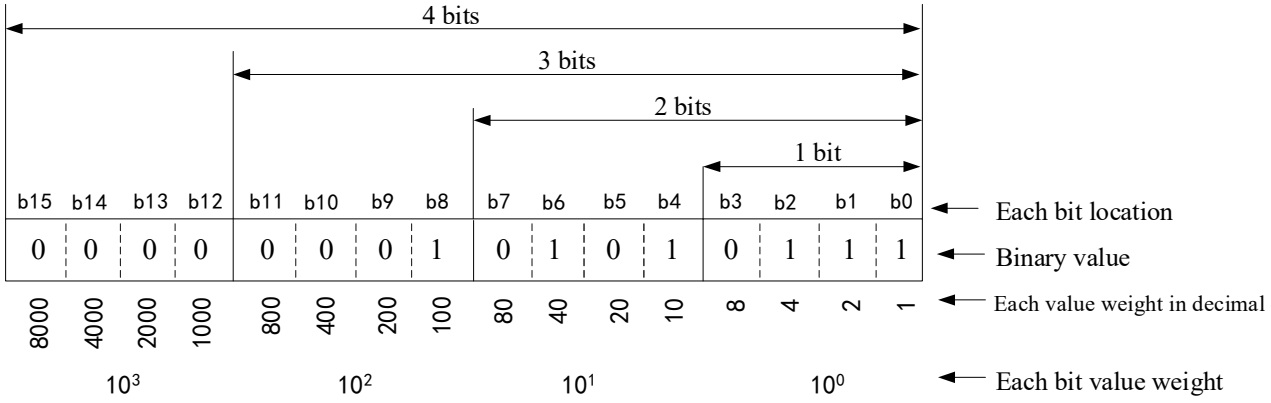
(2) Function

BCD (Binary Coded Decimal... BCD) refers to the use of 4-bit binary numbers to represent the values of each digit in a decimal number in parallel. For example, in BIN data, BCD data 0000 0001 0101 0111 (343) is used to represent the decimal number "157" in the following way.



BCD example

When BCD data is stored in 16 bits memory, it can handle values ranging from 0 to 9999 (the maximum value of 4 bits). The weights of each bit are shown in the following figure:



Weights of each value represented by BCD in decimal system

(3) Note

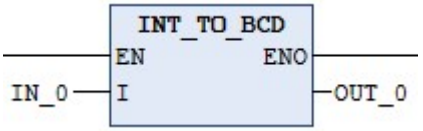
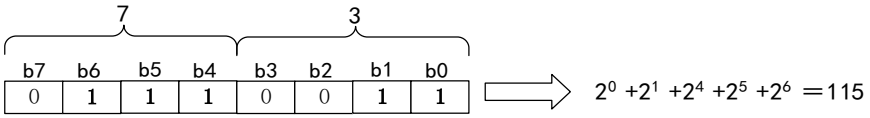
BCD code and integer conversion instruction data types

Command	Input type	Return value type
BCD_TO_BYTE	BYTE	BYTE
BCD_TO_DWORD	DWORD	DWORD

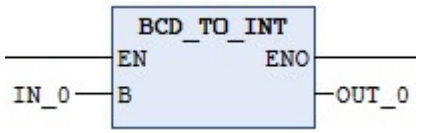
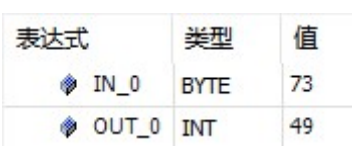
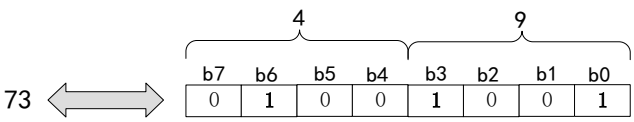
Command	Input type	Return value type
BCD_TO_INT	BYTE	INT
BCD_TO_WORD	WORD	WORD
BYTE_TO_BCD	BYTE	BYTE
DWORD_TO_BCD	DWORD	DWORD
INT_TO_BCD	INT	BYTE
WORD_TO_BCD	WORD	WORD

(4) Program example

① Convert integer data 73 to BCD code

DEMO	LD	ST
Define variables		VAR IN_0 :INT:=73; OUT_0:BYTE; END_VAR
Sample program		1 OUT_0:=INT_TO_BCD(I:= IN_0);
Operation result		
Principle analysis		

② Convert BCD code 73 to integer data

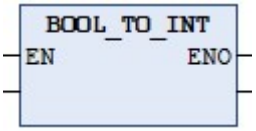
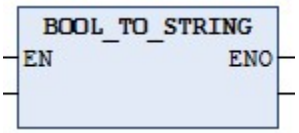
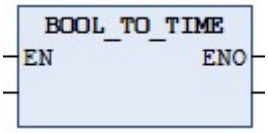
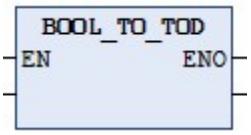
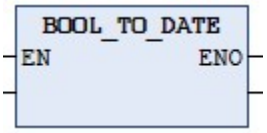
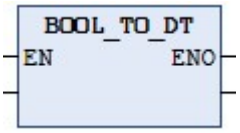
DEMO	LD	ST
Define variables		VAR IN_0 :BYTE:=73; OUT_0:INT; END_VAR
Sample program		1 OUT_0:=BCD_TO_INT(B:= IN_0);
Operation result		
Principle analysis		

1-4-7-2. BOOL_TO_<TYPE> boolean type conversion data

(1) Instuction overview

Convert Boolean data types to other data types.

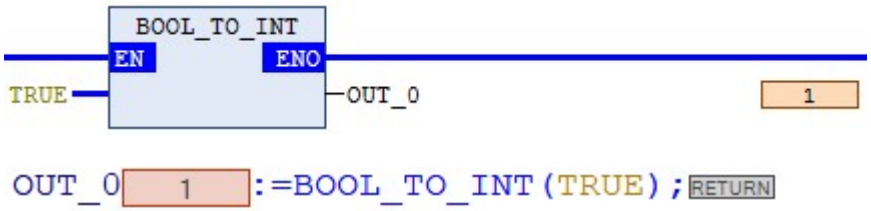
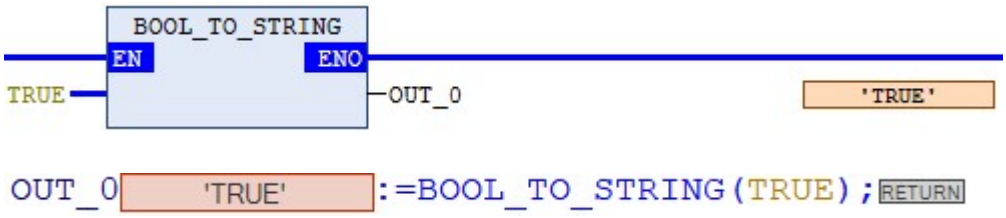
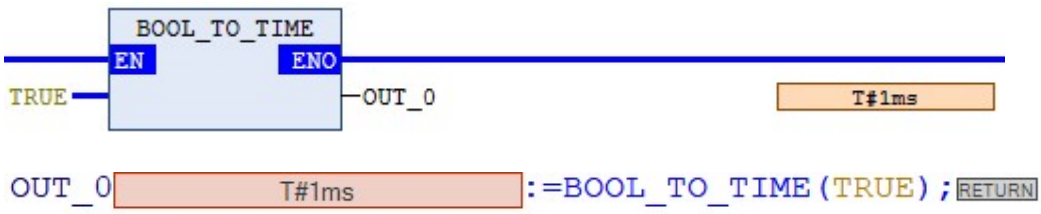
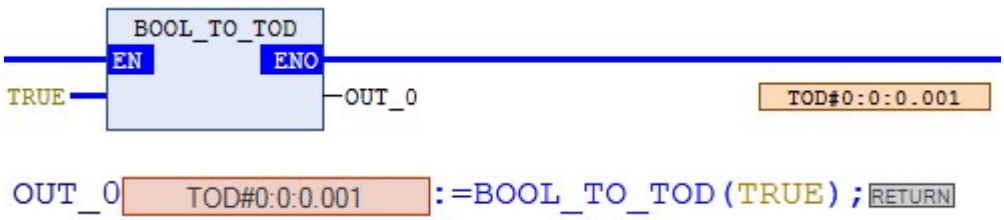
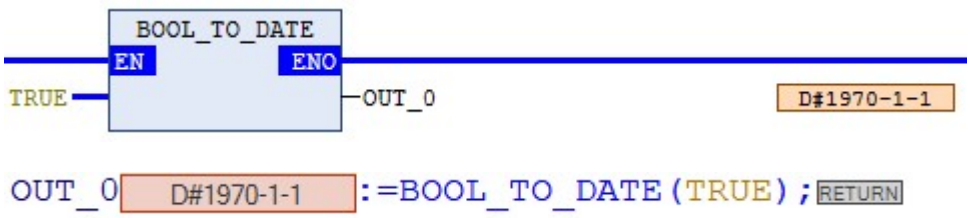
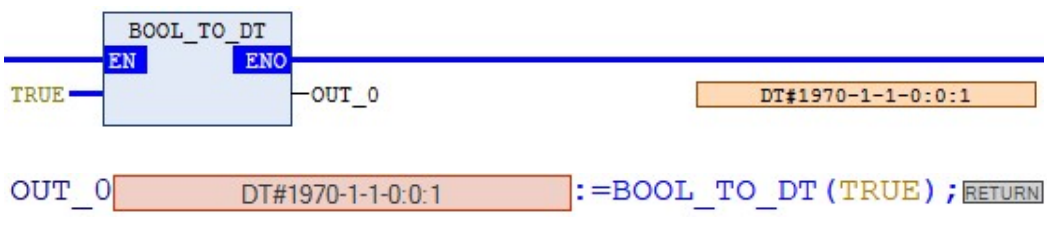
boolean type conversion data [BOOL_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
BOOL_TO_INT	BOOL convert to INT		BOOL_TO_INT
BOOL_TO_STRING	BOOL convert to STRING		BOOL_TO_STRING
BOOL_TO_TIME	BOOL convert to TIME		BOOL_TO_TIME
BOOL_TO_TOD	BOOL convert to TOD		BOOL_TO_TOD
BOOL_TO_DATE	BOOL convert to DATE		BOOL_TO_DATE
BOOL_TO_DT	BOOL convert to DT		BOOL_TO_DT

(2) Function

Supported data types: BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, DT and STRING.

(3) Program example

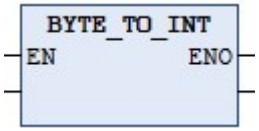
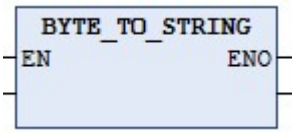
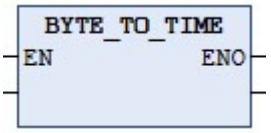
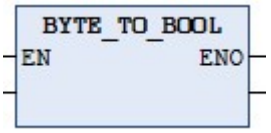
DEMO	LD/ST
Program example	
	
	
	
	
	

1-4-7-3. BYTE_TO_<TYPE> byte type conversion data

(1) Instruction overview

Convert byte types to other data types.

byte type conversion data [BYTE_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
BYTE_TO_INT	BYTE convert to INT		BYTE_TO_INT
BYTE_TO_STRING	BYTE convert to STRING		BYTE_TO_STRING
BYTE_TO_TIME	BYTE convert to TIME		BYTE_TO_TIME
BYTE_TO_BOOL	BYTE convert to BOOL		BYTE_TO_BOOL
.....			

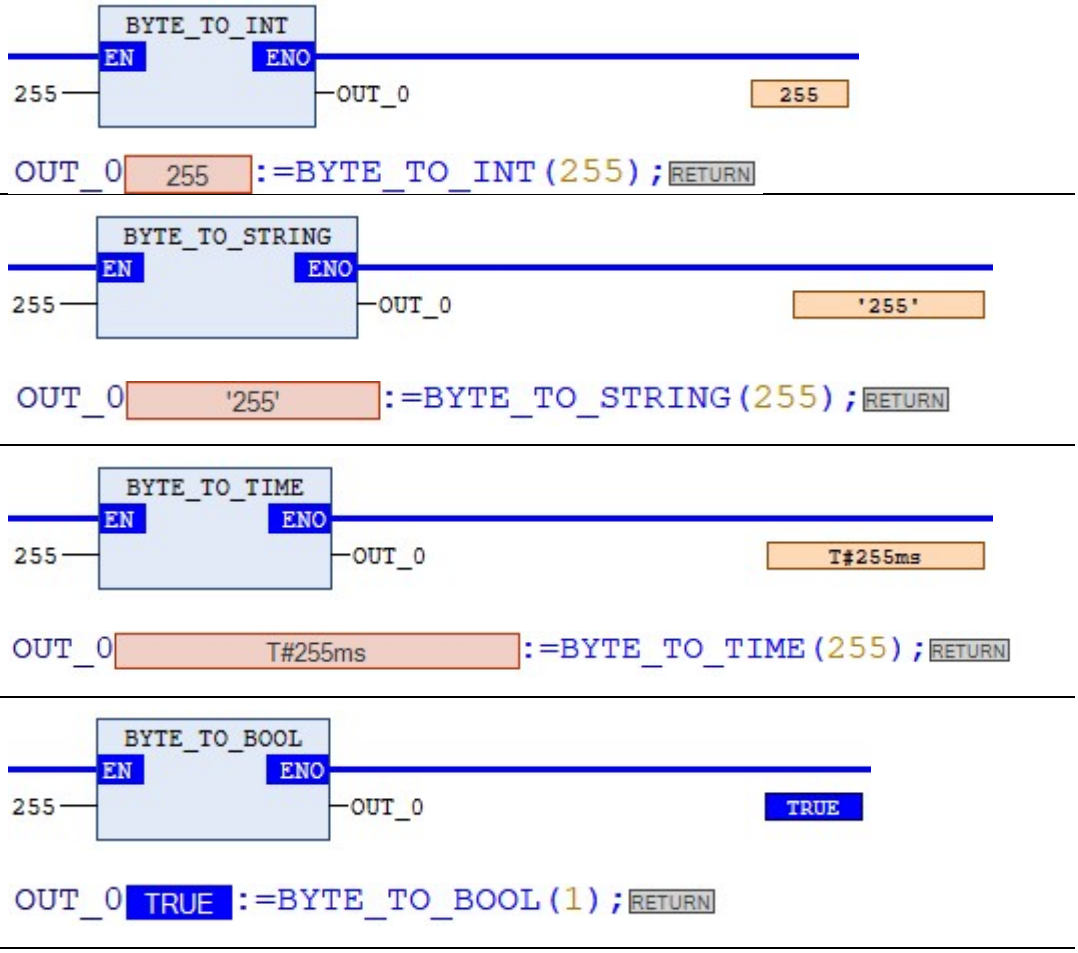
(2) Function

- Support data types: BOOL, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, DT, and STRING.
- When the output is BOOL: When the input is not equal to 0, the output is True. When the input is equal to 0, the output is FALSE.
- When the output is TIME or TOD: the input will be converted in milliseconds.
- When outputting as DATE or DT: The input will be converted in seconds.

(3) Note

- More byte type conversion instructions can be found in the software's online help.

(4) Program example

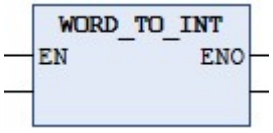
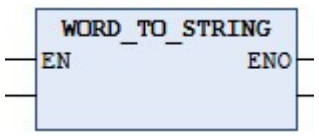
DEMO	LD/ST
Program example	

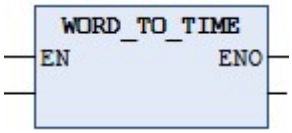
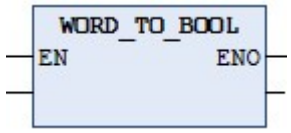
1-4-7-4. < Integer data > _TO_<TYPE> integer type conversion instruction

(1) Instruction overview

Convert integer data to other data types.

integer type conversion instruction [< Integer data >_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
WORD_TO_INT	WORD convert to INT		WORD_TO_INT
WORD_TO_STRING	WORD convert to STRING		WORD_TO_STRING

Instruction	Name	Graph	ST
WORD_TO_TIME	WORD convert to TIME		WORD_TO_TIME
WORD_TO_BOOL	WORD convert to BOOL		WORD_TO_BOOL
.....			

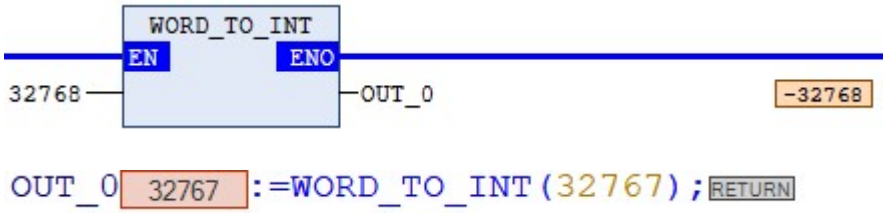
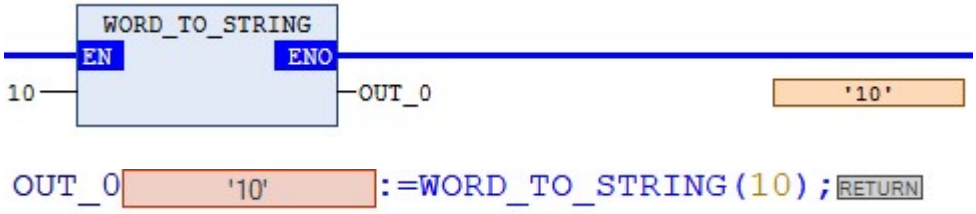
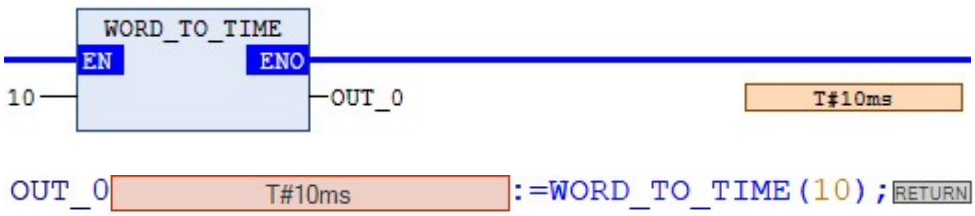
(2) Function

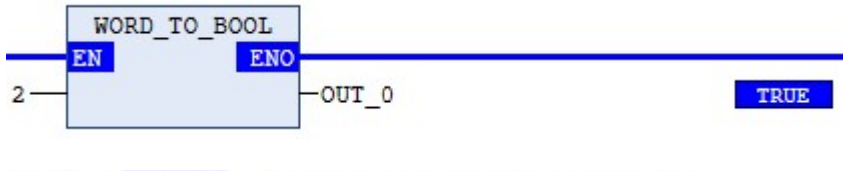
- Supports data types BOOL, BYTE, SINT, WORD, DWORD, USINT, INT, UINT, force, UDINT, REAL, TIME, DATE, TOD, DT, and strings.
- When the output is BOOL: if the input is not equal to 0, the output is TRUE and the input is equal to 0, the output is FALSE.
- When output time or TOD, the input is converted in milliseconds.
- When the output is a date or DT, the input is converted in seconds.

(3) Note

See the software's online help for more integer conversion instructions.

(4) Program example

DEMO	LD/ST
Program example	
	
	

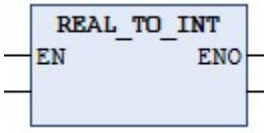
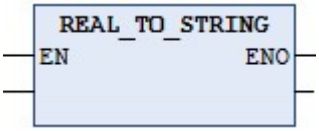
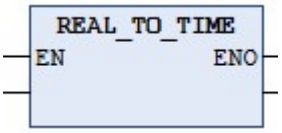
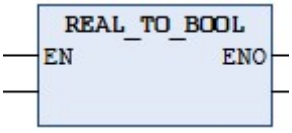


1-4-7-5. REAL_TO_<TYPE> Real type conversion instruction

(1) Instruction overview

Convert floating-point numbers to other types of data. When converting a floating-point number to another type of data, round the value to an integer value (except when converting to STRING) and then convert to the new quantity type.

Real type conversion instruction [REAL_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
REAL_TO_INT	REAL convert to INT		REAL_TO_INT
REAL_TO_STRING	REAL convert to STRING		REAL_TO_STRING
REAL_TO_TIME	REAL convert to TIME		REAL_TO_TIME
REAL_TO_BOOL	REAL convert to BOOL		REAL_TO_BOOL
.....			

(2) Function

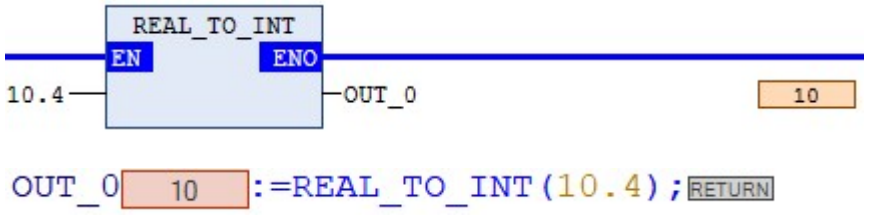
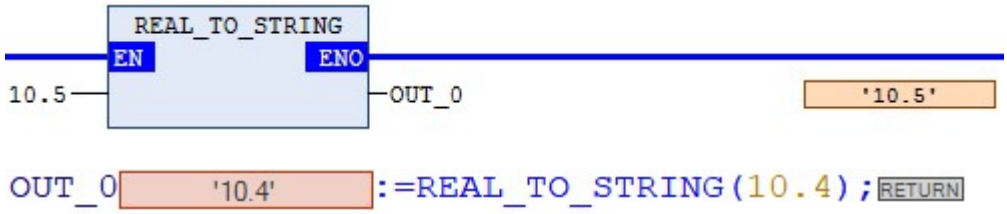
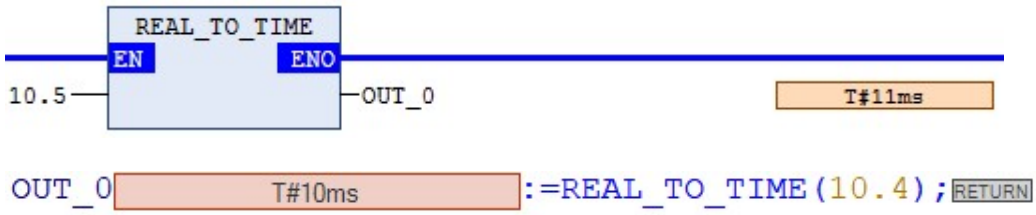
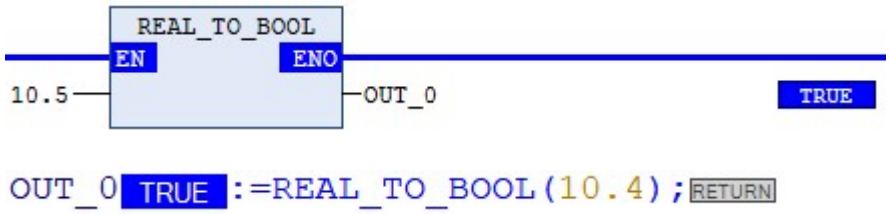
- Support data types BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, DT, and STRING.

- When the output is BOOL: The output is TRUE if the input is not equal to 0. When the input is 0, the output is FALSE.
- When the output is TIME or TOD, the input is converted in milliseconds.
- When the output is DATE or DT, the input is converted in seconds.

(3) Note

See the software's online help for more floating-point number type conversion instructions.

(4) Program example

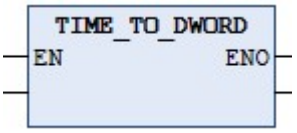
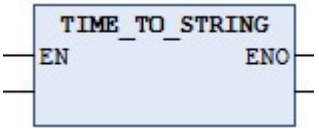
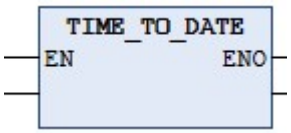
DEMO	LD/ST
Program example	   

1-4-7-6. TIME_TO_<TYPE> time type conversion instruction

(1) Instruction overview

Converts time-based data to other types, and the time is stored internally as a DWORD type in milliseconds (starting at 00:00 am for the TIME_OF_DAY variable).

time type conversion instruction [TIME_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
TIME_TO_DWORD	TIME convert to DWORD		TIME_TO_DWORD
TIME_TO_STRING	TIME convert to STRING		TIME_TO_STRING
TIME_TO_DATE	TIME convert to DATE		TIME_TO_DATE
.....			

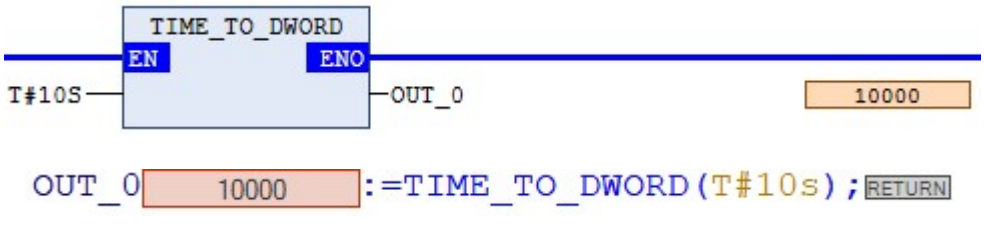
(2) Function

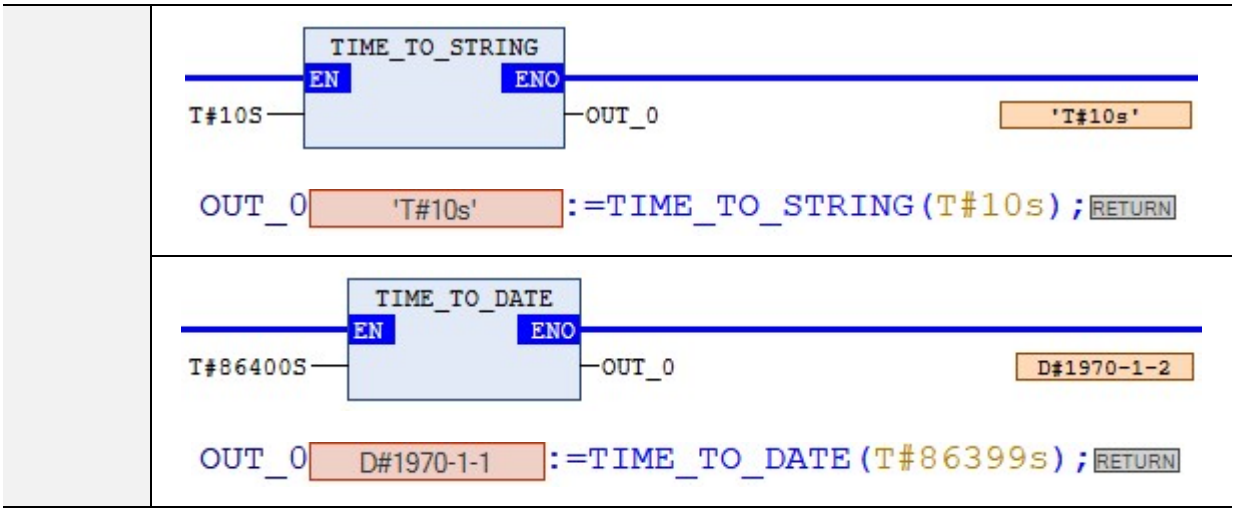
- Supports data types BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, DT, and STRING.

(3) Note

- See the software's online help for more time type conversion instructions.

(4) Program example

DEMO	LD/ST
Program example	



1-4-7-7. DATE_TO_<TYPE> Date type conversion instruction

(1) Instruction overview

Converts date-type data to other types of data, and dates are stored internally in seconds, starting on January 1, 1970.

Date type conversion instruction [DATE_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
DATE_TO_INT	DATE convert to INT	DATE_TO_INT EN ENO	DATE_TO_INT
DATE_TO_STRING	DATE convert to STRING	DATE_TO_STRING EN ENO	DATE_TO_STRING
DATE_TO_BOOL	DATE convert to BOOL	DATE_TO_BOOL EN ENO	DATE_TO_BOOL
.....			

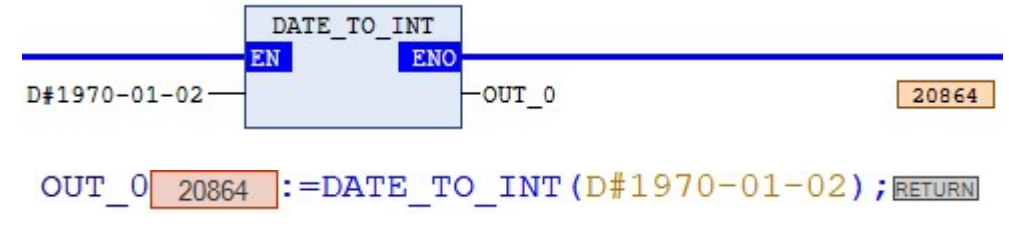
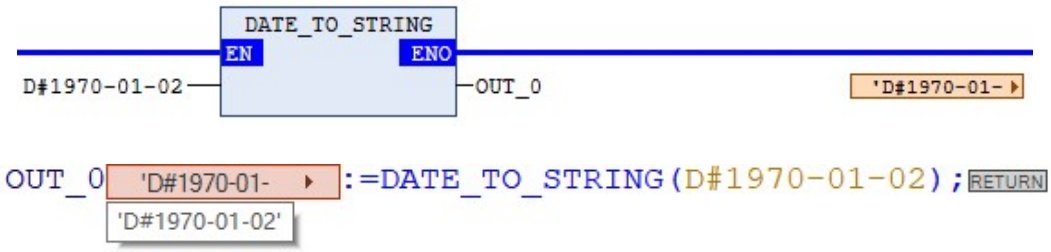
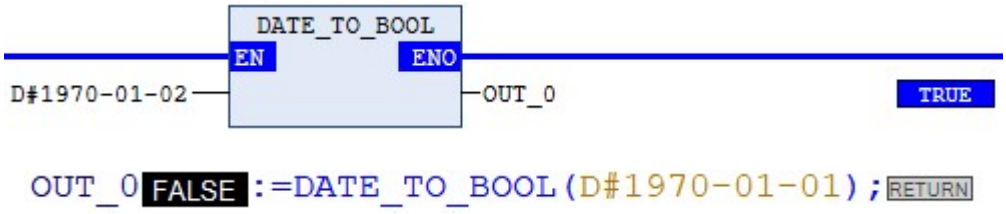
(2) Function

- Supports data types BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, DT, and STRING.

(3) Note

- See the software's online help for more date type conversion instructions.

(4) Program example

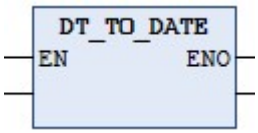
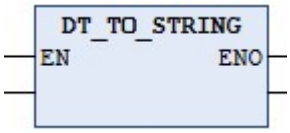
DEMO	LD/ST
Program example	  

1-4-7-8. DT_TO_<TYPE> Date time type conversion instruction

(1) Instruction overview

Converts date-time data to other types of data, with dates stored internally in seconds, starting on January 1, 1970.

Date time type conversion instruction [DT_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
DT_TO_DATE	DT convert to DATE		DT_TO_DATE
DT_TO_STRING	DT convert to STRING		DT_TO_STRING
.....			

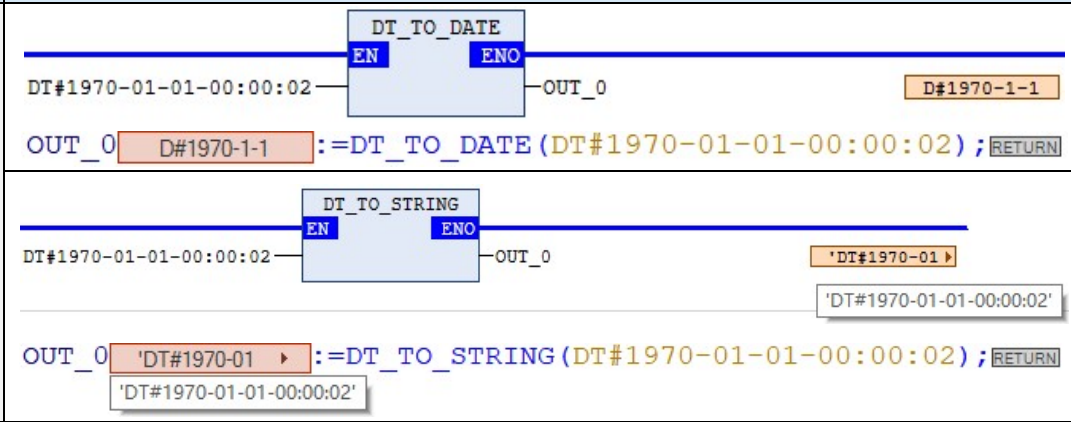
(2) Function

- Supports data types BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, DT, and STRING.

(3) Note

- See the software's online help for more date-time type conversion instructions.

(4) Program example

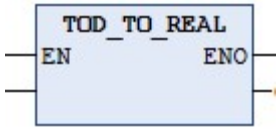
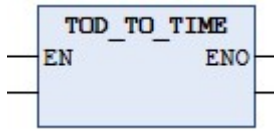
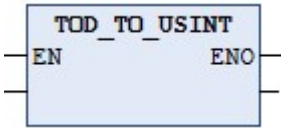
DEMO	LD/ST
Program example	

1-4-7-9. TOD_TO_<TYPE> Time type conversion instruction

(1) Instruction overview

Converts time-based data to other types of data, with dates internally converted in milliseconds.

Time type conversion instruction [TOD_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
TOD_TO_REAL	TOD convert to REAL		TOD_TO_REAL
TOD_TO_TIME	TOD convert to TIME		TOD_TO_TIME
TOD_TO_USINT	TOD convert to USINT		TOD_TO_USINT
.....			

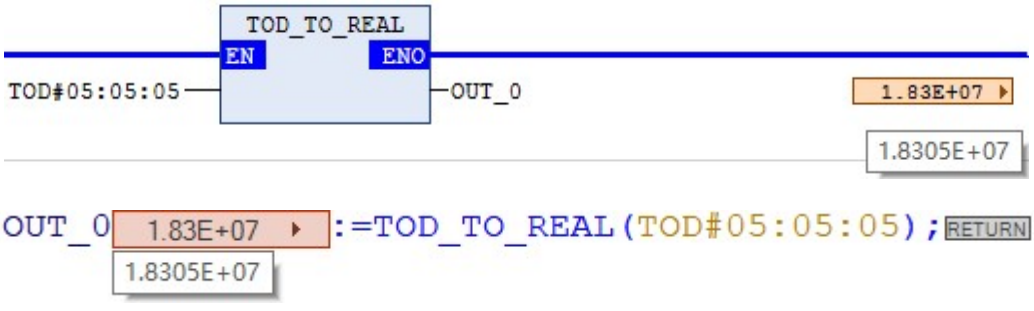
(2) Function

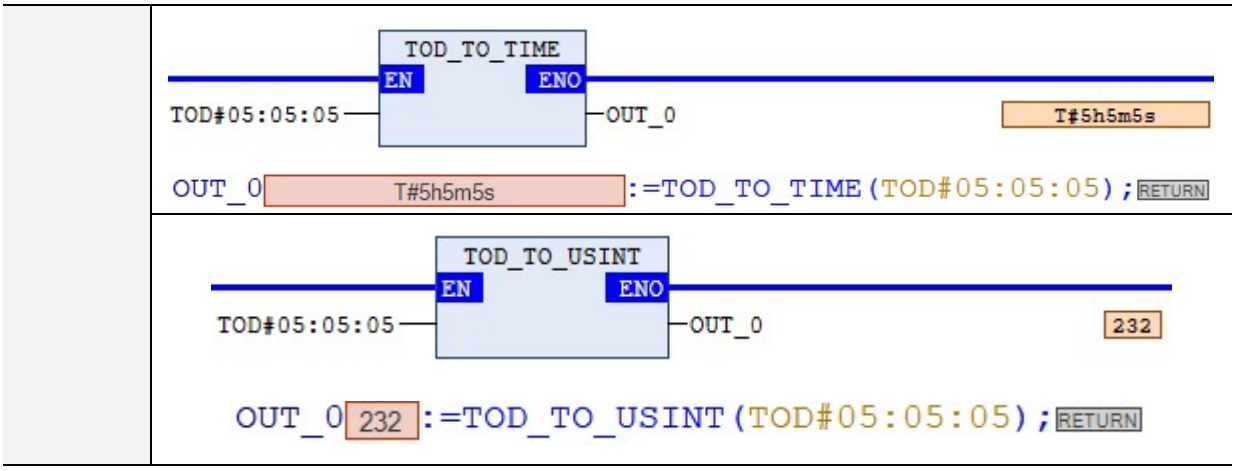
- Supports data types BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, DT, and STRING.

(3) Note

More time type data conversion instructions can be found in the software's online help.

(4) Program example

DEMO	LD/ST
Program example	



1-4-7-10. STRING_TO_<TYPE> Character type conversion instruction

(1) Instruction overview

To convert a string to another type of data, the string variable must contain a valid target variable value, otherwise the conversion result will be 0.

Character type conversion instruction [STRING_TO_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
STRING_TO_WORD	STRING convert to WORD	STRING_TO_WORD EN ENO	STRING_TO_WORD
STRING_TO_TIME	STRING convert to TIME	STRING_TO_TIME EN ENO	STRING_TO_TIME
.....			

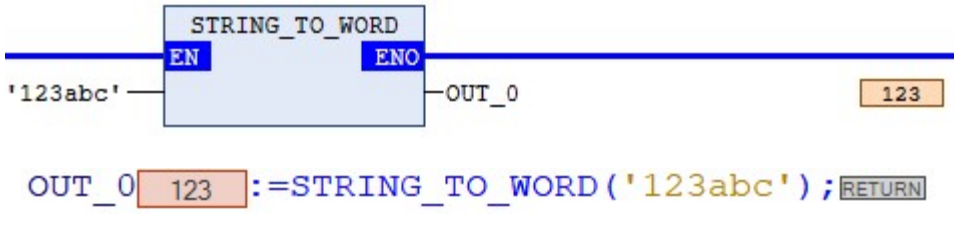
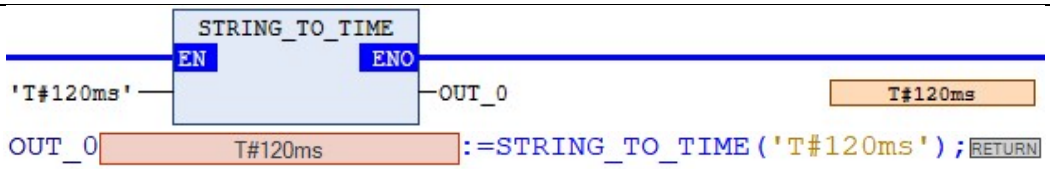
(2) Function

- Supports data types BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, TIME, DATE, TOD, DT, and STRING.

(3) Note

- See the software's online help for more string conversion instructions.

(4) Program example

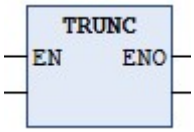
DEMO	LD/ST
Program example	 

1-4-7-11. Rounding TRUNC

(1) Instruction overview

Truncate the decimal part of the data and only retain the integer part.

Rounding [TRUNC]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
TRUNC	Rounding		TRUNC

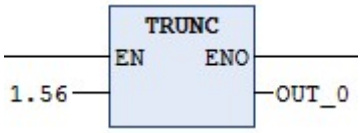
(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input data	REAL	As the data type	0	Input data
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
OUT	Data after rounding	DINT	As the data type	0	Data after rounding

(3) Note

- See the software's online help for more string conversion instructions.
- When it change from a larger data type to a smaller data type, it may lose information.
- This directive only cuts an integer. If you want to round it up, use the REAL_TO_INT directive.

(4) Program example

DEMO	LD	ST
Define variable	VAR OUT_0:DINT; END_VAR	
Program example		1 OUT_0:=TRUNC(-3.56);
Operation result		1 OUT_0 -3 :=TRUNC(-3.56);

1-4-7-12. Array conver to string AryToString

(1) Instruction overview

Converts a BYTE array to a string.

Array conver to string [AryToString]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryToString	Array conver to string		XDONE:=AryToString(In:= , Size:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Converted array source	BYTE	As the data type	-	BYTE array, the maximum number of elements is 1985
Size	Convert size	UINT	As the data type	1	The number of elements of the array In[] to be converted
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	STRING	1985 bytes	-	String

(3) Function

- The element values after the BYTE array "In[X]" (inclusive) are treated as character encoding and saved to the string "Out". Add a NULL character at the end of "Out".
- Specify the number of elements of "In[]" to be converted in "Size".
- If there are NULL characters between In[X] and In[X+“Size-1”], only the content before the NULL character is saved to Out.

(4) Note

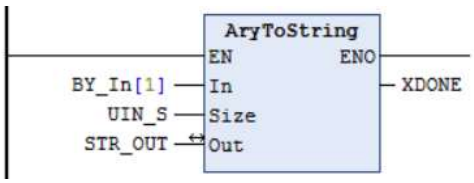
- If the value of Size exceeds the array area of In[], avoid a PLC exception.
- In[] must be a BYTE array; otherwise, the function returns FALSE and Out remains unchanged.
- If the value of "Size" is 0, "Out" is a null-only string, and the function returns True.
- If Size exceeds 1985 bytes, truncated it, the function returns True.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
    BY_In: ARRAY[0..5] OF BYTE:= [51,56,57,68,69,70];
    UIN_S: UINT:= 4;
    STR_OUT: STRING(1985);
    XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:=AryToString(In:= BY_In[1],
    Size:= UIN_S,
    Out:= STR_OUT);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
BY_In	ARRAY [0..5] OF BYTE	
BY_In[0]	BYTE	16#33
BY_In[1]	BYTE	16#38
BY_In[2]	BYTE	16#39
BY_In[3]	BYTE	16#44
BY_In[4]	BYTE	16#45
BY_In[5]	BYTE	16#46
UIN_S	UINT	16#0004
STR_OUT	STRING(1985)	'89DE'
XDONE	BOOL	TRUE

</td></tr>
</table>

1-4-7-13. Date convert to string DateToString

(1) Instruction overview

Convert the date to a string.

Date convert to string [DateToString]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
DateToString	Date convert to string		xDone:=DateToString(In:= , Out=>);

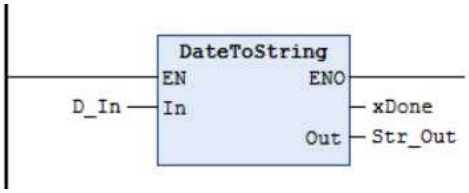
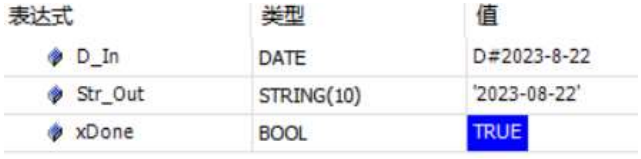
(2) Related variable

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert source	DATE	As the data type	D#1970-1-1	Date
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	STRING[10]	10 bytes	-	Convert result string

(3) Function

- Convert the date "In" to a string.
- Add NULL character at the end of the string "Out".

(4) Program example

DEMO	LD	ST
Define variable	VAR D_In: DATE := D#2023-8-22; Str_Out: STRING(10); xDone: BOOL; END_VAR	
Program example		xDone := DateToString(In:= D_In, Out=> Str_Out);
Operation result		

1-4-7-14. Date time conver to string DtToString

(1) Instruction overview

Convert date and time to a string.

Date time convert to string [DtToString]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
DtToString	Date time conver to string		xDone:=DtToString(In:= , Out=>);

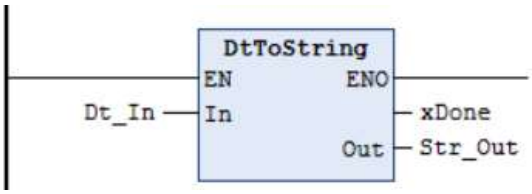
(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert source	DT	As the data type	D#1970-1-1-0:0:0	Date time
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	STRING[19]	19 bytes	-	string

(3) Function

- Convert the date time "In" to a string.
- Add NULL character at the end of the string "Out".

(4) Program example

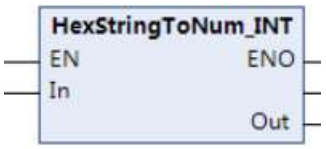
DEMO	LD	ST
Define variable	VAR Dt_In: DATE_AND_TIME:= DT#2023-8-22-13:15:50; Str_Out: STRING(19); xDone: BOOL; END_VAR	
Program example		xDone := DtToString(In:= Dt_In, Out=> Str_Out);
Operation result		

1-4-7-15. Convert hexadecimal strings to integers HexStringToNum_XXX

(1) Instruction overview

Convert hexadecimal string format to integers.

Convert hexadecimal strings to integers [HexStringToNum_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
HexStringToNum_XXX	hexadecimal strings →integer		xDone:=HexStringToNum_INT(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert source	STRING	As the data type	-	hexadecimal strings
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	USINT/UINT/UDINT/ULINT SINT/INT/DINT/LINT	As the data type	-	Integer type

(3) Function

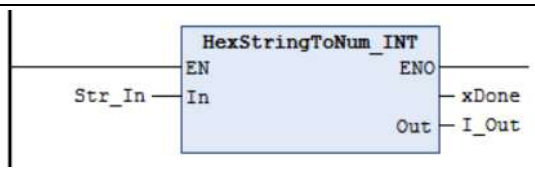
- Convert the string "In" in hexadecimal format to an integer. Regardless of the high half space (16#20) and '0' (16#30). Simultaneously ignore the underline in the string (16#5F).
- The instruction name varies depending on the data type of the "Out" output variable. For example, when "Out" is of type INT, the instruction name is HexStringToNum_INT.

(4) Note

- Even if the conversion result exceeds the valid range of "Out", no exception will occur, but illegal values are saved in "Out".
- When the content of "In" does not end with a NULL character, it may cause conversion exceptions.
- When the content of "In" is a character that cannot be converted to a numerical value, the program encounters an exception and the function returns a value of FALSE, while "Out" remains unchanged.

(5) Program example

DEMO	LD	ST
Define variable		VAR Str_In: STRING(1985); I_Out: INT; xDone: BOOL; END_VAR

Program example		xDone := HexStringToNum_INT(In:= Str_In, Out=> I_Out);
Operation result	表达式 / 类型 / 值 Str_In / STRING(1985) / '200' I_Out / INT / 512 xDone / BOOL / TRUE 表达式 / 类型 / 值 Str_In / STRING(1985) / '-F' I_Out / INT / -15 xDone / BOOL / TRUE	

1-4-7-16. Convert integers to fixed length decimal/hexadecimal strings

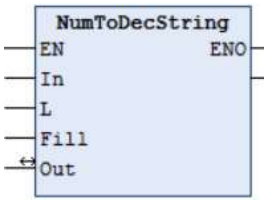
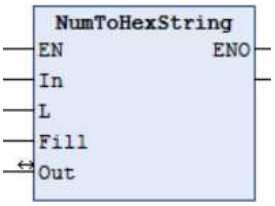
NumToDecString/NumToHexString

(1) Instruction overview

NumToDecString: Converts an integer to a fixed length decimal string format.

NumToHexString: Converts an integer to a fixed length hexadecimal string format.

Convert integers to fixed length decimal/hexadecimal strings [NumToDecString/NumToHexString]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
NumToDecString	Fixed length decimal string conversion		xDone:=NumToDecString(In:= , L:= , Fill:= , Out:=);
NumToHexString	Fixed length hexadecimal string conversion		xDone:=NumToHexString(In:= , L:= , Fill:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert source	USINT/UINT/UDINT/ULINT SINT/INT/DINT/LINT	As the data type	-	Integer
L	Characters	UINT	0-1985	1	Out characters
Fill	Add characters	_eFiLL_CHR	1 (_BLANK) 2 (_ZERO)	1 (_BLANK)	Add characters
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	STRING	As the data type	-	String

(3) Function

- NumToDecString: Converts the integer "In" to a decimal format string of UTF-8 half width English numerals. When "In" is a negative number, add a "-" (minus sign) at the beginning.
- NumToHexString: Converts the integer "In" to a hexadecimal format string of UTF-8 half width English numerals. When "In" is a negative number, it is expressed in the form of a complement of 2 (taking the inverse then add 1).
- The above instructions set the number of characters in the string "Out" to "L". When the number of characters is insufficient, add the character specified by "Fill" to the high position.
- When the number of characters in the conversion result is more than "L", only the "L" character is taken into "Out" from the lowest position of the conversion result.
- The data type of "Fill" is the enumerator_eFilL_CHR. The meaning of the enumeration body is shown in

the following figure:

Enumerating elements	Meaning
1 (_BLANK)	‘’ (Half width blank character)
2 (_ZERO)	‘0’

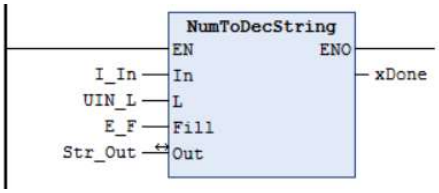
(4) Note

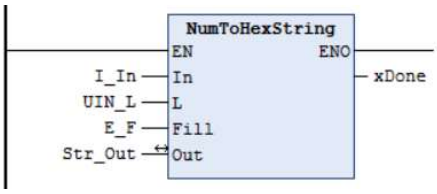
- When the value of "L" is 0, the value of "Out" remains unchanged.
- When the number of characters in the conversion result exceeds "L", save the "L" bits in "Out" from the low position of the conversion result, as shown in the following example:

Instruction	“In” value	“L” value	“Out” value
NumToDecString	128	2	28
NumToHexString			80

- When the value of "L" exceeds the valid range, the program encounters an exception, and the function returns a value of FALSE with no change in "Out".
- Compilation exception occurs when the value of "Fill" exceeds the valid range.
- When the conversion result exceeds the range of "Out", the output is an error value.

(5) Program example

NumToDecString	LD	ST
Define variable		VAR I_In: INT; UIN_L: UINT; E_F: _eFILL_CHR; Str_Out: STRING(1985); xDone: BOOL; END_VAR
Program example		xDone:=NumToDecString(In:= I_In, L:= UIN_L, Fill:= E_F, Out:= Str_Out);
Operation result		

NumToHexString	LD	ST
Define variable		VAR I_In: INT; UIN_L: UINT; E_F: _eFILL_CHR; Str_Out: STRING(1985); xDone: BOOL; END_VAR
Program example		xDone:=NumToHexString(In:= I_In, L:= UIN_L, Fill:= E_F, Out:= Str_Out);
Operation result		

表达式	类型	值
I_In	INT	128
UIN_L	UINT	8
E_F	_EFILL_CHR	_BLANK
Str_Out	STRING(1985)	' 80'
xDone	BOOL	TRUE

表达式	类型	值
I_In	INT	-128
UIN_L	UINT	8
E_F	_EFILL_CHR	_BLANK
Str_Out	STRING(1985)	'FFFFFF80'
xDone	BOOL	TRUE

1-4-7-17. String conver to array StringToAry

(1) Instruction overview

Convert the string to a BYTE array.

String conver to array [StringToAry]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
StringToAry	String conver to array		XDONE:=StringToAry(In:= , AryOut:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert source	STRING	Maximum 1985 characters	-	String
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
AryOut	Array after converting	BYTE	As the data type	-	BYTE type array
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Bytes after converting	UINT	0-1985	-	Bytes after converting

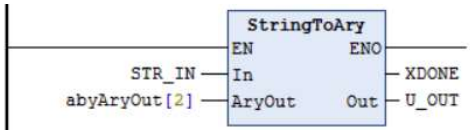
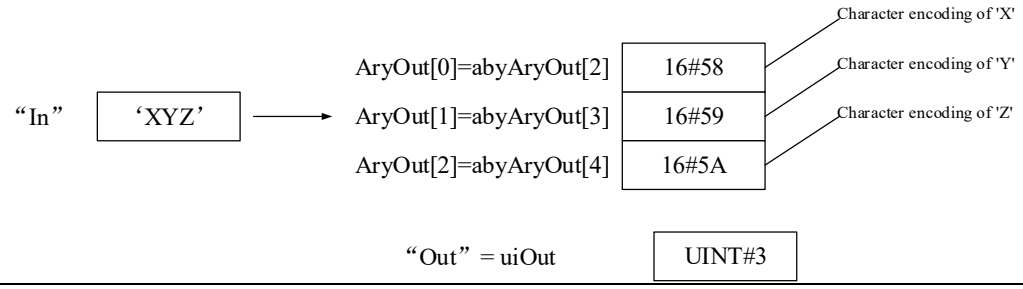
(3) Function

- Treat the character encoding of the string "n" as a numerical value and save each character one by one in the BYTE arrangement "AryOut[]". Save the number of converted bytes in "Out".

(4) Note

- The NULL character at the end of "In" will not be saved in "AryOut[]".
- When "In" is a string with only NULL characters, the value of "Out" is 0, and "AryOut[]" remains unchanged.
- When the number of bytes in "In" is greater than the number of elements in "AryOut[]", avoid PLC exceptions.
- When the number of bytes in "In" exceeds the valid range (1985), characters outside the valid range will be truncated, and the function will convert 1985 characters normally.

(5) Program example

DEMO	LD	ST
Define variable	VAR STR_IN: STRING(1985):= 'XYZ'; abyAryOut: ARRAY[0..5] OF BYTE; U_OUT: UINT; XDONE: BOOL; END_VAR	
Program example		XDONE:=StringToAry(In:= STR_IN, AryOut:= abyAryOut[2], Out=> U_OUT);
Operation result		
Principle analysis		

1-4-7-18. Time convert to string TodToString

(1) Instruction overview

Time convert to string.

Time convert to string [TodToString]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
TodToString	Time convert to string		xDone:=TodToString(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert source	TOD	As the data type	TOD#0:0:0	Hour minute second
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	STRING[13]	13 bytes	-	Conversion result string

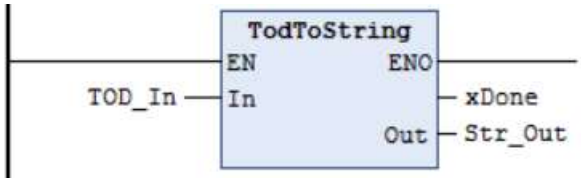
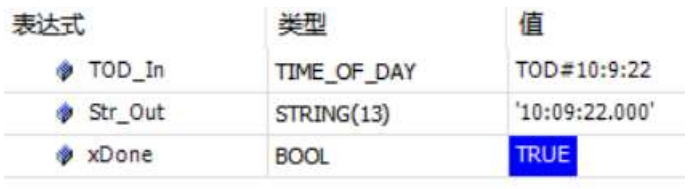
(3) Function

- Convert the time "In" to a string. Add NULL character at the end of "Out".

(4) Note

- "Out" is expressed in milliseconds.

(5) Program example

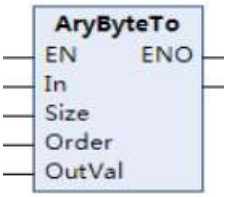
DEMO	LD	ST
Define variable		VAR TOD_In: TIME_OF_DAY; Str_Out: STRING(13); xDone: BOOL; END_VAR
Program example		xDone:=TodToString(In:= TOD_In, Out=> Str_Out);
Operation result		

1-4-7-19. Convert from byte array AryByteTo

(1) Instruction overview

Combine BYTE type array elements and save them to variables.

Convert from byte array [AryByteTo]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryByteTo	Convert from byte array		XDONE:=AryByteTo(In:= , Size:= , Order:= , OutVal:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object array	BYTE	As the data type	-	Convert object array
Size	Convert object element sizes	UINT	As the data type	-	The number of In[] elements to convert
Order	Convert order	_eBYTE_ORDER	As the data type	_LOW_HIGH	The position to be converted
OutVal	Convert result	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/REAL/LREAL/ TIME/DATE/TOD/DT/STRING	As the data type	-	Convert result

(3) Function

- Combine the starting "Size" elements of the conversion object array "In[X]" according to the data type size of the conversion result "OutVal", and then save them to "OutVal".
- When combining the "In[]" element, specify the order in the conversion order "Order"; The data type of "Order" is enumeration_eBYTE_ORDER; The meaning of enumeration values is as follows.

Enum	Meaning
_LOW_HIGH	First low byte, then high byte
_HIGH_LOW	First high byte, then low byte

- The following two situations introduce the functions of AryByteTo.

(1) The data type of "OutVal" is 1 byte

When the data type of "OutVal" is 1 byte, save "In[]" in "OutVal" in units of 1 byte.

There are several types of 1-byte data:

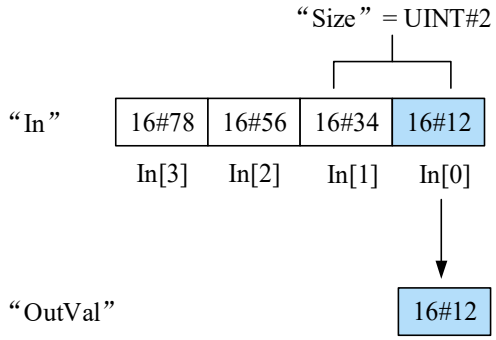
Type	Data type
Bit string	BYTE

Integer	USINT, SINT
Real	None
Time, duration, date, string	None

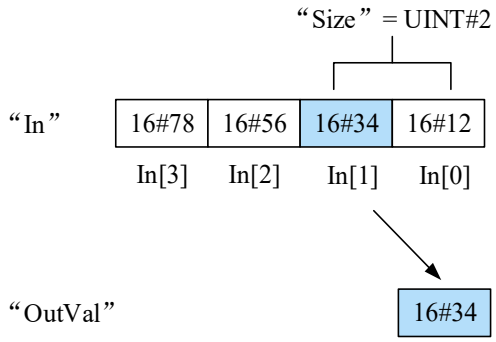
The saving method is as follows:

“Order” value	“OutVal” data type	Method for saving to "OutVal"
LOW_HIGH	BOOL	Save the value of the 0th bit of In[0] to "OutVal"
	Not BOOL	Save the value of In[0] to "OutVal"
HIGH_LOW	BOOL	Save the value of the 0th bit of In[1] to "OutVal"
	Not BOOL	Save the value of In[1] to "OutVal"

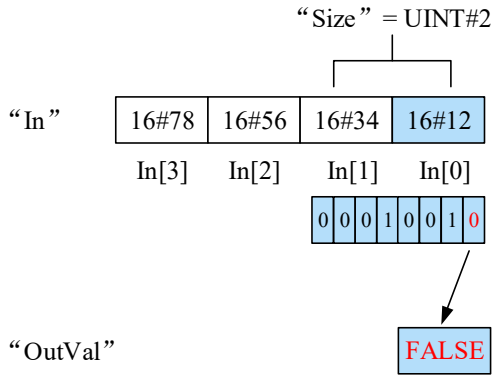
The example of "OutVal" being a SINT type variable with "Size"=UINT#2 and "Order"=_LOW_HIGH is as follows:



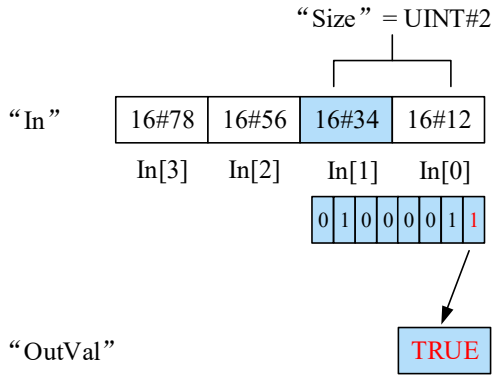
The example of "OutVal" being a SINT type variable with "Size"=UINT#2 and "Order"= _HIGH_LOW is as follows:



The example of "OutVal" being a BOOL type variable with "Size"=UINT#2 and "Order"= _LOW_HIGH is as follows:



The example of "OutVal" being a BOOL type variable with "Size"=UINT#2 and "Order"= _HIGH_LOW is as follows:



(2) The data type of “OutVal” is 2 bytes or more

When the data type of "OutVal" is 2 bytes or more, combine "In[]" in units of the data type size of "OutVal" and save it to "OutVal".

There are several types of data with 2 bytes or more:

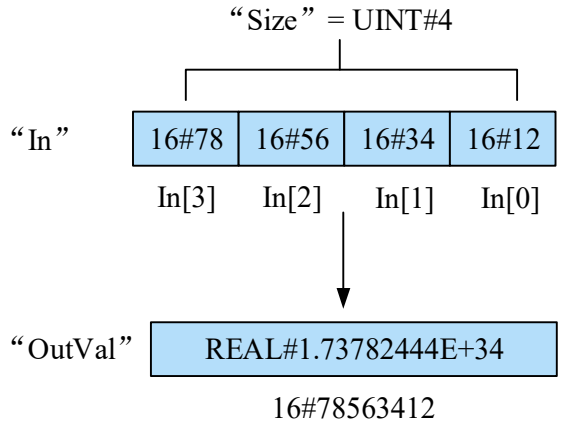
Type	Data type
Bit string	WORD, DWORD, LWORD
Integer	UINT, UDINT, ULINT, INT, DINT, LINT
Real	REAL, LREAL
Time, duration, date, string	TIME, DATE, TOD, TD, STRING of 2 bytes or more

The saving method is as follows:

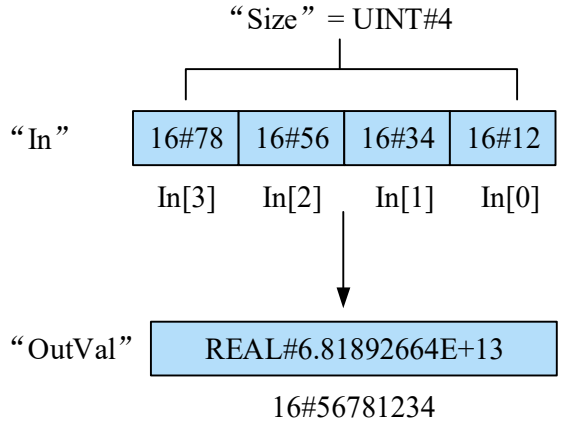
- Create 1 word (2 bytes) of data based on the value of "Order" combined with "In[0]" and "In[1]". If the value of "Order" is _LOW_HIGH, then the high byte is "In[1]" and the low byte is "In[0]"; If the value of "Order" is _HIGH_LOW, then the high byte is "In[0]" and the low byte is "In[1]". Combine the values after "In[2]" and "In[3]" using the same method to create multiple 1-word data.
- Based on the data type size of "OutVal", combine the multiple 1-word data. For example, when "OutVal" is a DWORD type, combine four 1-word data. If "OutVal" is the entire array, save "In[i]" (where i is even) to "OutVal[i+1]". Save "In[i]" (where i is odd) to "OutVal[i-1]". If the value of "Size" is odd, it will be saved until "OutVal ["Size"]", and 16#00 will be saved

in "OutVal[" Size "-1]". Save the obtained data to "OutVal".

The example of "OutVal" being a REAL type variable with "Size"=UINT#4 and "Order"=_LOW_HIGH is as follows:



The example of "OutVal" being a REAL type variable, with "Size"=UINT#4 and "Order"=_HIGH_LOW is as follows:



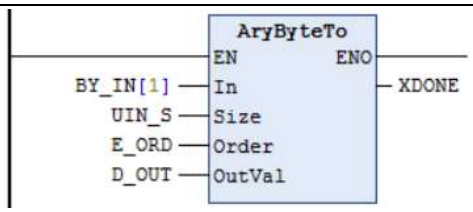
(4) Note

- When the value of "Size" exceeds the array area of "In[]", it is necessary to avoid PLC exceptions.
- "OutVal" cannot be the entire BOOL type array, otherwise the program running result will be incorrect.
- When the value of "Size" is 0, the value of "Out" is true, and "OutVal" remains unchanged.

(5) Program example

DEMO	LD	ST
Define variable	VAR BY_IN: ARRAY[0..5] OF BYTE:= [16#21,16#52,16#62,16#81,16#66,16#55]; UIN_S: UINT:= 3; E_ORD: _eBYTE_ORDER:= _eBYTE_ORDER._LOW_HIGH; D_OUT: DWORD; XDONE: BOOL; END_VAR	

Program example




```
XDONE:=AryByteTo(
    In:= BY_IN[1],
    Size:= UIN_S,
    Order:= E_ORD,
    OutVal:= D_OUT);
```

Operation result

表达式	类型	值
BY_IN	ARRAY [0..5] OF BYTE	
BY_IN[0]	BYTE	16#21
BY_IN[1]	BYTE	16#52
BY_IN[2]	BYTE	16#62
BY_IN[3]	BYTE	16#81
BY_IN[4]	BYTE	16#66
BY_IN[5]	BYTE	16#55
UIN_S	UINT	16#0003
E_ORD	_EBYTE_ORDER	_LOW_HIGH
D_OUT	DWORD	16#00816252
XDONE	BOOL	TRUE

1-4-7-20. Batch conversion of arrays to BCD AryToBCD

(1) Instruction overview

Convert the elements of an unsigned integer array to BCD.

Batch conversion of arrays to BCD [AryToBCD]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryToBCD	Batch conversion of arrays to BCD		AryToBCD(In:=, Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert array source	USINT/UINT/ UDINT/ULINT	As the data type	-	Unsigned integer array
Size	Convert size	UINT	As the data type	1	The number of In[] elements to convert
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
AryOut	Result array	Consistent with In	As the data type	-	BCD array

(3) Function

- Convert the "Size" elements starting from In[0] of the unsigned integer array In[] to BCD and output them to the BCD array AryOut[].
- The effective range of In[] and AryOut[] varies depending on the combination of element data types.

In[] element data type	In[]effective range	AryOut[] effective range
USINT	0-99	16#00-16#99(BCD)
UINT	0-9999	16#0000-16#9999(BCD)
UDINT	0-99999999	16#0000_0000-16#9999_9999(BCD)
ULINT	0-9999999999999999	16#0000_0000_0000_0000-16#9999_9999_9999_9999(BCD)

(4) Note

- Unable to convert signed BCD. Therefore, please set the data type of In[] to any one of the unsigned integers (USINT, UINT, UDINT, ULINT).
- When the value of "Size" is 0, the function returns a value of True, and the value of AryOut[] remains unchanged.
- When the value of In[] exceeds the valid range, the function returns a value of FALSE, and the value of AryOut[] remains unchanged.

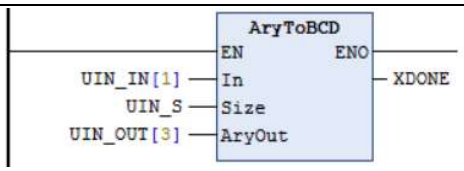
- When the value of "Size" exceeds the array area of In[] or AryOut[], it is necessary to avoid PLC exceptions.
- The "In[] array" needs to use an integer type. If it is determined that it does not belong to this type, a compilation error will be reported.
- The "In[] array" type is not USINT/UINT/UDINT/ULINT, the runtime function returns a value of FALSE, and the value of AryOut[] remains unchanged.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   UIN_IN: ARRAY[0..5] OF UINT:= [5,16,26,168,12,512];
   UIN_S: UINT:= 3;
   UIN_OUT: ARRAY[0..5] OF UINT;
   XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:=AryToBCD(
     In:= UIN_IN[1],
     Size:= UIN_S,
     AryOut:= UIN_OUT[3]);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
UIN_IN	ARRAY [0..5] OF UINT	
UIN_IN[0]	UINT	16#0005
UIN_IN[1]	UINT	16#0010
UIN_IN[2]	UINT	16#001A
UIN_IN[3]	UINT	16#00A8
UIN_IN[4]	UINT	16#000C
UIN_IN[5]	UINT	16#0200
UIN_S	UINT	16#0003
UIN_OUT	ARRAY [0..5] OF UINT	
UIN_OUT[0]	UINT	16#0000
UIN_OUT[1]	UINT	16#0000
UIN_OUT[2]	UINT	16#0000
UIN_OUT[3]	UINT	16#0016
UIN_OUT[4]	UINT	16#0026
UIN_OUT[5]	UINT	16#0168
XDONE	BOOL	TRUE

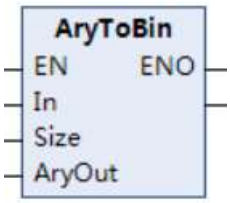
</td></tr>
</table>

1-4-7-21. Batch convert arrays to bit strings AryToBin

(1) Instruction overview

Convert the elements of the BCD array into bit strings.

Batch convert arrays to bit strings [AryToBin]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryToBin	Batch convert arrays to bit strings		XDONE:=AryToBin(In:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert array source	BYTE/WORD/ DWORD/LWORD	As the data type	-	BCD array
Size	Number of Conversion Elements	UINT	As the data type	1	The number of In[] elements to convert
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
AryOut	Result array	Consistent with In	As the data type	-	Converted bit string array

(3) Function

- Convert the "Size" elements starting from In[0] of the BCD array In[] to bit strings, and output the bit string array AryOut[].
- The effective range of In[] and AryOut[] varies depending on the combination of element data types.

The data type of the In[] element	The effective range of In[]	The effective range of AryOut[]
BYTE	16#00-16#99(BCD)	0-99
WORD	16#0000-16#9999(BCD)	0-9999
DWORD	16#0000_0000-16#9999_9999(BCD)	0-99999999
LWORD	16#0000_0000_0000_0000-16#9999_9999_9999_9999(BCD)	0-9999999999999999

(4) Note

- Unable to convert signed BCD. Therefore, please set the data type of AryOut[] to any one of the bit strings (BYTE, WORD, DWORD, LWORD).
- When the value of "Size" is 0, the function returns a value of True, and the value of AryOut[] remains unchanged.
- When the value of In[] exceeds the valid range, the function returns a value of FALSE, and the value of

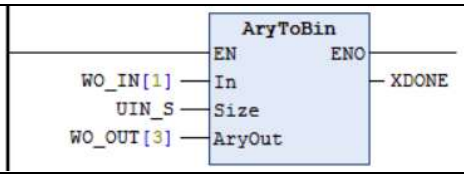
AryOut[] remains unchanged.

- When the value of "Size" exceeds the array area of In[] or AryOut[], it is necessary to avoid PLC exceptions.
- The data types of In[] and AryOut[] should be consistent, otherwise the return value will return FALSE, while the value of AryOut[] remains unchanged.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   WO_IN: ARRAY[0..5] OF WORD:= [16#9,16#16,16#26,16#168,16#12,16#512];
   UIN_S: UINT:= 3;
   WO_OUT: ARRAY[0..5] OF WORD;
   XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td> 
</td><td>

```
XDONE:=AryToBin(
    In:= WO_IN[1],
    Size:= UIN_S,
    AryOut:= WO_OUT[3]);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
WO_IN	ARRAY [0..5] OF WO...	
WO_IN[0]	WORD	16#0009
WO_IN[1]	WORD	16#0016
WO_IN[2]	WORD	16#0026
WO_IN[3]	WORD	16#0168
WO_IN[4]	WORD	16#0012
WO_IN[5]	WORD	16#0512
UIN_S	UINT	16#0003
WO_OUT	ARRAY [0..5] OF WO...	
WO_OUT[0]	WORD	16#0000
WO_OUT[1]	WORD	16#0000
WO_OUT[2]	WORD	16#0000
WO_OUT[3]	WORD	16#0010
WO_OUT[4]	WORD	16#001A
WO_OUT[5]	WORD	16#00A8
XDONE	BOOL	TRUE

</td></tr>
</table>

1-4-7-22. Bit string conver to Gray code BinToGray_XXX

(1) Instruction overview

Bit string conver to Gray code.

Bit string conver to Gray code [BinToGray_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

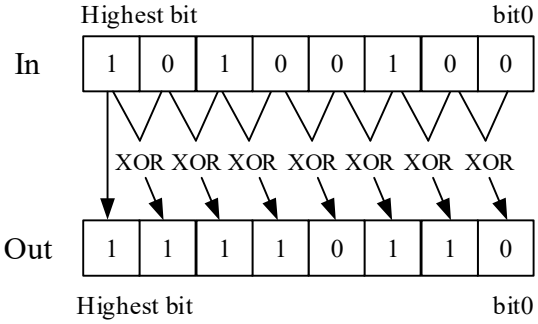
Instruction	Name	Graph	ST
BinToGray_XXX	Bit string conver to Gray code		xDone:=BinToGray_BYTE(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object	BYTE/WORD/DWORD/LWORD	As the data type	0	Bit string
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	Consistent with In	As the data type	-	Gray code

(3) Function

- Convert the bit string object "In" to Gray code.
- The instruction name varies depending on the data type of "In" and "Out". For example, when "In" and "Out" are WORD types, the instruction name is BinToGray_WORD.
  - Taking "In" and "Out" as BYTE types as an example, explain the conversion steps below:
    (1) The highest bit (7th bit) of "In" is directly used as the highest bit (7th bit) of "Out".
    (2) The XOR of the 7th value of "In" and the 6th value of "In" is the 6th value of "Out".
    (3) Using the same method, determine the value to the lowest (0th) position of "Out".



(4) Note

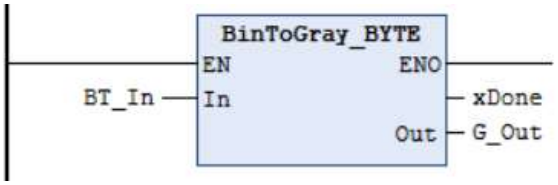
- It is recommended to unify the data types for "In" and "Out".

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
    BT_In: BYTE;
    G_Out: BYTE;
    xDone: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
xDone:=BinToGray_BYTE(
    In:= BT_In,
    Out=> G_Out);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
BT_In	BYTE	2#10010110
G_Out	BYTE	2#11011101
xDone	BOOL	TRUE

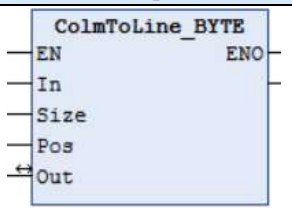
</td></tr>
</table>

1-4-7-23. Bit train/bit line conver to array ColmToLine_XXX

(1) Instruction overview

Extract the value of the bits at the specified position of each array element and output it as a bit string.

Bit train/bit line conver to array [ColmToLine_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ColmToLine_XXX	Bit train/bit line conver to array		xDone:=ColmToLine_BYTE(In:= , Size:= , Pos:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object array	BYTE/WORD/ DWORD/LWORD	As the data type	-	Convert object array
Size	Number of elements for converting objects	USINT	As the data type	1	The number of elements in In[] that serve as conversion objects
Pos	The position to be converted	USINT	As the data type	0	The position to be converted
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	BYTE/WORD/ DWORD/LWORD	As the data type	-	Convert result

(3) Function

- Extract the values of the bits at the specified positions of each element in the array, arrange them, and output them as bit strings.
- Firstly, extract the "Size" elements of the transformation object array In[] starting from the input array element In[]. Then, only extract the value of the "Pos" position of the above elements. Arrange it into a bit string of "Size" bits and save it in the lowest bit position of the conversion result "Out". Save FALSE in the remaining bits of "Out".
- The instruction name varies depending on the data type of "Out". For example, when "Out" is of BYTE type, the instruction name is ColmToLine_BYTE.

(4) Note

- When the value of "Size" is 0, the function returns a value of True, and all bits of "Out" are FALSE.
- When the value of "Size" exceeds the valid range, the function returns a value of FALSE and "Out" remains unchanged.
- When the value of "Pos" exceeds the valid range, the function returns a value of FALSE and "Out" remains

unchanged.

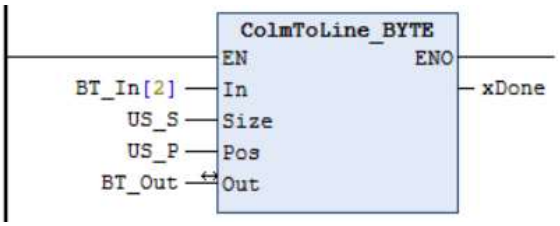
- When the value of "Size" exceeds the array area of In[], it is necessary to avoid PLC exceptions.
- When the data type passed in by "In[]" does not belong to the bit string range, an error occurs during compilation.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   BT_In: ARRAY[0..5] OF BYTE:= [82,176,13,47,125,34];
   US_S: USINT:= 4;
   US_P: USINT:= 1;
   BT_Out: BYTE;
   xDone: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
xDone:=ColmToLine_BYTE(
    In:= BT_In[2],
    Size:= US_S,
    Pos:= US_P,
    Out:= BT_Out);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
BT_In	ARRAY [0..5] OF BYTE	
BT_In[0]	BYTE	2#01010010
BT_In[1]	BYTE	2#10110000
BT_In[2]	BYTE	2#00001101
BT_In[3]	BYTE	2#00101111
BT_In[4]	BYTE	2#01111101
BT_In[5]	BYTE	2#00100010
US_S	USINT	2#00000100
US_P	USINT	2#00000001
BT_Out	BYTE	2#00001010
xDone	BOOL	TRUE

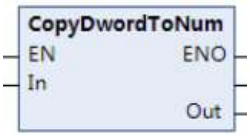
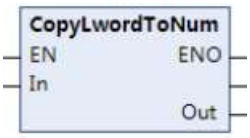
</td></tr>
</table>

1-4-7-24. Copy bit mode CopyByteToNum/CopyWordToNum/CopyDwordToNum/ CopyLwordToNum

(1) Instruction overview

Copy the content of the bit string directly into a signed integer.

Copy bit mode [CopyByteToNum/CopyWordToNum/CopyDwordToNum/ CopyLwordToNum]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
CopyByteToNum	Copy bit mode		XDONE:=CopyByteToNum(In:= , Out=>);
CopyWordToNum	Copy bit mode		XDONE:=CopyWordToNum(In:= , Out=>);
CopyDwordToNum	Copy bit mode		XDONE:=CopyDwordToNum(In:= , Out=>);
CopyLwordToNum	Copy bit mode		XDONE:=CopyLwordToNum(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Copy source	BYTE/WORD/DWORD/LWORD	As the data type	0	Copy source
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Paste target	SINT/INT/DINT/LINT	As the data type	（*）	Paste target

(3) Function

- Copy the content of the source without any processing, and directly copy it to the paste target "Out".
- There are four types of instruction names due to different combinations of data types for "In" and "Out".

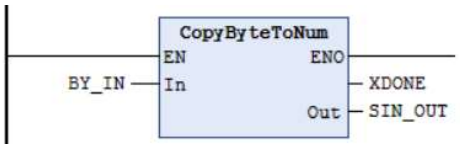
“In”	“Out”	Command name
BYTE	SINT	CopyByteToNum
WORD	INT	CopyWordToNum
DWORD	DINT	CopyDwordToNum
LWORD	LINT	CopyLwordToNum

(4) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   BY_IN: BYTE:= 20;
   XDONE: BOOL;
   SIN_OUT: SINT;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:=CopyByteToNum(
    In:= BY_IN,
    Out=> SIN_OUT);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
BY_IN	BYTE	2#00010100
XDONE	BOOL	TRUE
SIN_OUT	SINT	2#00010100

</td></tr>
</table>

1-4-7-25. Copy bit mode CopyDwordToReal/CopyLwordToLreal

(1) Instruction overview

Copy the content of the bit string directly into the real number.

Copy bit mode [CopyDwordToReal/CopyLwordToLreal]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
CopyDwordToReal	Copy bit mode		XDONE:=CopyDwordToReal(In:= , Out=>);
CopyLwordToLreal	Copy bit mode		XDONE:=CopyLwordToLreal(In:= , Out=>);

(2) Related variables

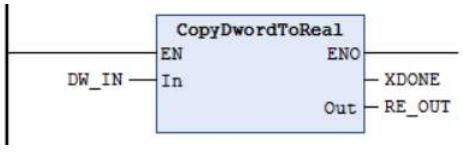
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Copy source	DWORD/LWORD	As the data type	0	Copy source
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Paste target	REAL/LREAL	As the data type	(*)	Paste target

(3) Function

- Copy the content of the source without any processing, and directly copy it to the paste target "Out".
- There are two types of instruction names due to different combinations of data types for "In" and "Out".

"In"	"Out"	Command name
DWORD	REAL	CopyDwordToReal
LWORD	LREAL	CopyLwordToLreal

(4) Program example

DEMO	LD	ST
Define variable		VAR DW_IN: DWORD:= 20; XDONE: BOOL; RE_OUT: REAL; END_VAR
Program example		XDONE:=CopyDwordToReal(In:= DW_IN, Out=> RE_OUT);
Operation result	XDONE TRUE :=CopyDwordToReal(In:= DW_IN 2#00000000000000000000000000010100 , Out=> RE_OUT 2#00000000000000000000000000010100);	

1-4-7-26. Copy bit mode CopyRealToDWord/CopyLrealToLword

(1) Instruction overview

Copy the content of real numbers directly into the bit string.

Copy bit mode [CopyRealToDWord/CopyLrealToLword]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
CopyRealToDWord	Copy bit mode		XDONE:=CopyRealToDWord(In:= , Out=>);
CopyLrealToLword	Copy bit mode		XDONE:=CopyLrealToLword(In:= , Out=>);

(2) Related variables

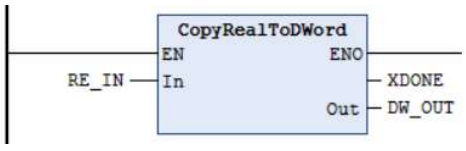
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Copy source	REAL/LREAL	As the data type	0	Copy source
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Paste target	DWORD/LWORD	As the data type	（*）	Paste target

(3) Function

- Copy the content of the source without any processing, and directly copy it to the paste target "Out".
- There are two types of instruction names due to different combinations of data types for "In" and "Out".

“In”	“Out”	Command name
REAL	DWORD	CopyRealToDWord
LREAL	LWORD	CopyLrealToLword

(4) Program example

DEMO	LD	ST
Define variable		VAR RE_IN: REAL:= 8.9; DW_OUT: DWORD; XDONE: BOOL; END_VAR
Program example		XDONE:=CopyRealToDWord(In:= RE_IN, Out=> DW_OUT);
Operation result	XDONE TRUE :=CopyRealToDWord(In:= RE_IN 2#01000001000011100110011001100110 , Out=> DW_OUT 2#01000001000011100110011001100110);	

1-4-7-27. Copy bit mode CopyNumToXXX

(1) Instruction overview

Copy the contents of signed integers directly into real numbers.

Copy bit mode [CopyNumToXXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
CopyNumToByte	Copy bit mode		XDONE:=CopyNumToByte(In:= , Out=>);
CopyNumToWord	Copy bit mode		XDONE:=CopyNumToWord(In:= , Out=>);
CopyNumToDword	Copy bit mode		XDONE:=CopyNumToDword(In:= , Out=>);
CopyNumToLword	Copy bit mode		XDONE:=CopyNumToLword(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Copy source	SINT/INT/DINT/LINT	As the data type	0	Copy source
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Paste target	BYTE/WORD/DWORD/LWORD	As the data type	（*）	Paste target

(3) Function

- Copy the content of the source without any processing, and directly copy it to the paste target "Out".
- There are four types of instruction names due to different combinations of data types for "In" and "Out".

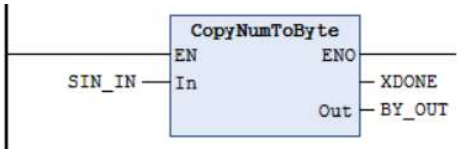
“In”	“Out”	Command name
SINT	BYTE	CopyNumToByte
INT	WORD	CopyNumToWord
DINT	DWORD	CopyNumToDword
LINT	LWORD	CopyNumToLword

(4) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
    SIN_IN: SINT:= 20;
    BY_OUT: BYTE;
    XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:=CopyNumToByte(
    In:= SIN_IN,
    Out=> BY_OUT);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
SIN_IN	SINT	2#00010100
BY_OUT	BYTE	2#00010100
XDONE	BOOL	TRUE

</td></tr>
</table>

1-4-7-28. Copy bit mode CopyNumToReal/CopyNumToLreal

(1) Instruction overview

Copy the contents of signed integers directly into real numbers.

Copy bit mode [CopyNumToReal/CopyNumToLreal]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
CopyNumToReal	Copy bit mode		XDONE:=CopyNumToReal(In:= , Out=>);
CopyNumToLreal	Copy bit mode		XDONE:=CopyNumToLreal(In:= , Out=>);

(2) Related variables

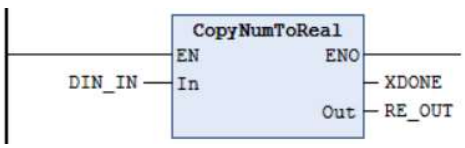
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Copy source	DINT/LINT	As the data type	0	Copy source
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Paste target	REAL/LREAL	As the data type	（*）	Paste target

(3) Function

- Copy the content of the source without any processing, and directly copy it to the paste target "Out".
- There are two types of instruction names due to different combinations of data types for "In" and "Out".

“In”	“Out”	Command name
DINT	REAL	CopyNumToReal
LINT	LREAL	CopyNumToLreal

(4) Program example

DEMO	LD	ST
Define variable		VAR DIN_IN: DINT:= 20; RE_OUT: REAL; XDONE: BOOL; END_VAR
Program example		XDONE:=CopyNumToReal(In:= DIN_IN, Out=> RE_OUT);
Operation result	XDONE TRUE :=CopyNumToReal(In:= DIN_IN 2#00000000000000000000000000010100 , Out=> RE_OUT 2#00000000000000000000000000010100);	

1-4-7-29. Copy bit mode CopyRealToNum/CopyLrealToNum

(1) Instruction overview

Copy the contents of real numbers directly into integers.

Copy bit mode [CopyRealToNum/CopyLrealToNum]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
CopyRealToNum	Copy bit mode		XDONE:=CopyRealToNum(In:= , Out=>);
CopyLrealToNum	Copy bit mode		XDONE:=CopyLrealToNum(In:= , Out=>);

(2) Related variables

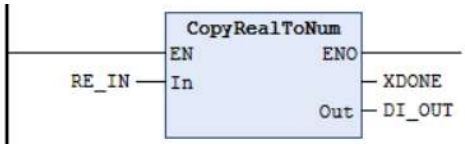
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Copy source	REAL/LREAL	As the data type	0	Copy source
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Paste target	DINT/LINT	As the data type	（*）	Paste target

(3) Function

- Copy the content of the source without any processing, and directly copy it to the paste target "Out".
- There are two types of instruction names due to different combinations of data types for "In" and "Out".

“In”	“Out”	Command name
REAL	DINT	CopyRealToNum
LREAL	LINT	CopyLrealToNum

(4) Program example

DEMO	LD	ST
Define variable		VAR RE_IN: REAL:= 8.8; DI_OUT: DINT; XDONE: BOOL; END_VAR
Program example		XDONE:=CopyRealToNum(In:= RE_IN, Out=> DI_OUT);
Operation result	XDONE TRUE :=CopyRealToNum(In:= RE_IN 2#01000001000011001100110011001101 , Out=> DI_OUT 2#01000001000011001100110011001101);	

1-4-7-30. Gray code convert to bit string GrayToBin_XXX

(1) Instruction overview

Convert Gray code to bit string.

Gray code convert to bit string [GrayToBin_XXX]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

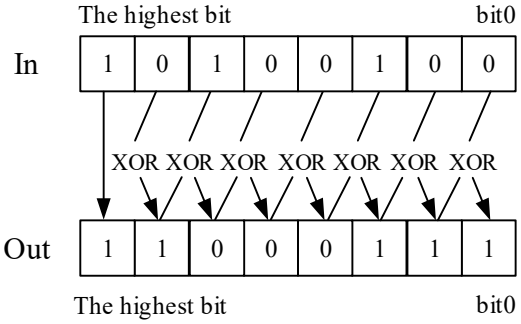
Instruction	Name	Graph	ST
GrayToBin_XXX	Gray code convert to bit string		XDONE:=GrayToBin_BYTE(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object	BYTE/WORD/DWORD/LWORD	As the data type	0	Gray code
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	Consistent with In	As the data type	-	Bit string

(3) Function

- Convert the Gray code conversion object "In" to a bit string.
- The instruction name varies depending on the data type of "In" and "Out". For example, when "In" and "Out" are WORD types, the instruction name is GrayToBin_WORD.
- Taking "In" and "Out" as BYTE types as an example, explain the conversion steps below:
  - ① The highest bit (7th bit) of "In" is directly used as the highest bit (7th bit) of "Out";
  - ② The XOR of the 6th value of "In" and the 7th value of "Out" is the 6th value of "Out";
  - ③ Determine the value to the lowest (0th) position of "Out" using the same method.



(4) Note

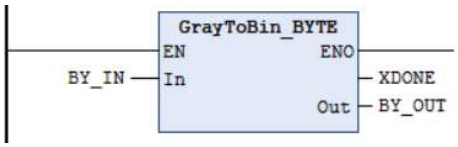
- It is recommended to unify the data types for "In" and "Out".

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   BY_IN: BYTE:= 2#10011101;
   BY_OUT: BYTE;
   XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:=GrayToBin_BYTE(
    In:= BY_IN,
    Out=> BY_OUT);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
BY_IN	BYTE	2#10011101
BY_OUT	BYTE	2#11101001
XDONE	BOOL	TRUE

</td></tr>
</table>

1-4-7-31. Gray code convert to angle GRAY

(1) Instruction overview

Convert Gray code to angle.

Gray code convert to angle [Gray]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
GRAY	Gray code convert to angle		XDONE:=GRAY(In:= , Resolution:= , ERC:= , ZPC:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object	WORD	As the data type	0	Gray code
Resolution	Resolution	_eGRY_RESOLUTION	As the data type	_R256	Resolution
ERC	Encoder residual correction value	UINT	As the data type	0	Encoder residual correction value
ZPC	Origin correction value	UINT	As the data type	0	Origin correction value
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	LREAL	As the data type	-	Convert result

(3) Function

- Convert the output value "In" of the rotary encoder represented by Gray code to an angle value. The unit of conversion result "Out" is "degrees".
- The data type of "Resolution" is enumeration type _eGRY_RESOLUTION. The meaning of enumeration values is as follows:

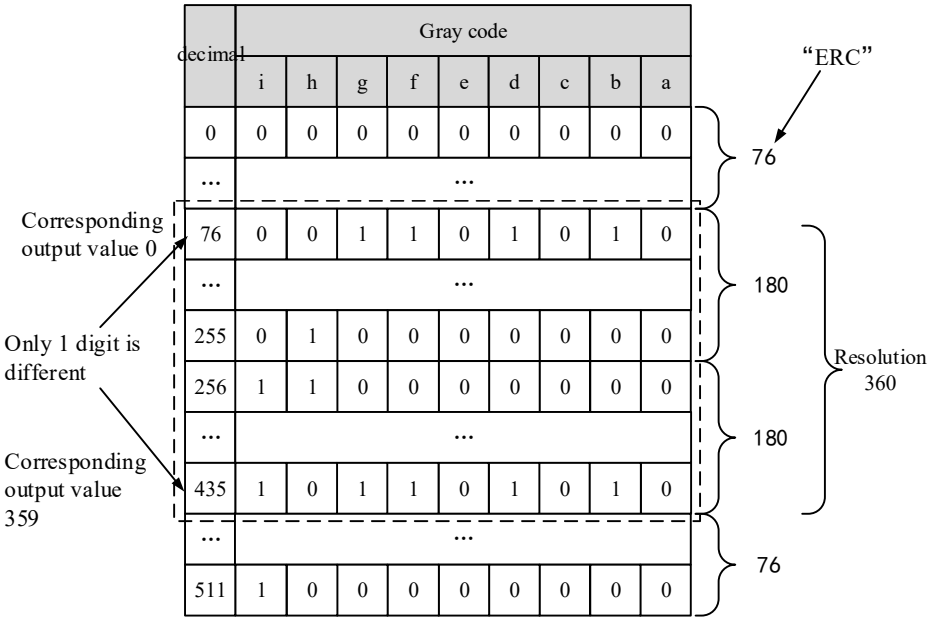
Enum	Meaning	Enum	Meaning
_R256	256	_R10B	10 bits (1024)
_R1B	1 bit (2)	_R11B	11 bits (2048)
_R2B	2 bits (4)	_R12B	12 bits (4096)
_R3B	3 bits (8)	_R13B	13 bits (8192)
_R4B	4 bits (16)	_R14B	14 bits (16384)
_R5B	5 bits (32)	_R15B	15 bits (32768)
_R6B	6 bits (64)	_R360	360
_R7B	7 bits (128)	_R720	720

_R8B	8 bits (256)	_R1024	1024
_R9B	9 bits (512)		

- Gray code refers to a type of alternating binary code. Its characteristic is that any two adjacent codes have only one binary digit different.
- Gray code is commonly used for the output of absolute encoders, etc.
- The following are 4-bit binary codes and Gray codes.

Decimal number	Binary code				Gray code			
	2^3	2^2	2^1	2^0	d	c	b	a
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

- If Gray code is used, when the output value of the encoder increases or decreases by 1, only one bit of the value changes, thus preventing the output of incorrect values. The encoder residual correction value "ERC" is the correction value specified within the Gray code range to ensure that the maximum and minimum output values of the encoder are only one bit different in the Gray code when the encoder resolution is not a power of 2.
- For example, using an absolute encoder with a resolution of 360. The number of digits in the Gray code is 9. The range of values that can be represented by 9 bits is 0-511. At this point, within a range of 180 degrees from the center of 0-511, i.e. 76-435, use the Gray code. Therefore, the Gray code representing output value 0 is 001101010 (decimal number 76), and the Gray code representing output value 359 is 101101010 (decimal number 435), with only one difference between the two.
- At this point, the value of the encoder residual correction value "ERC" is 76.



- The origin correction value "ZPC" is set when it is necessary to offset the origin angle of the rotary encoder. For example, when the origin of a rotary encoder with a resolution of 256 is offset by 90°, the value of "ZPC" is 256 x (90/360)=64.
- When “In”=WORD#16#256, “Resolution”=_R10B, “ERC”=UINT#0, “ZPC”=UINT#2, the examples are as the following:

  Firstly, due to the resolution of 10 bits, one unit of the Gray code is 360/1024=0.3515625°.

  The value of the decimal number corresponding to Gray code 16#256 is 923. Therefore, the angle value before correction is 0.3515625° ×923=324.492188°

  The value of "ERC" is 0, and the value of "ZPC" is 2. Therefore, the corrected angle value is 324.492188° - (0+2) ×0.3515625° =323.7890625°

  The value of "Out" is LREAL #323.7890625.

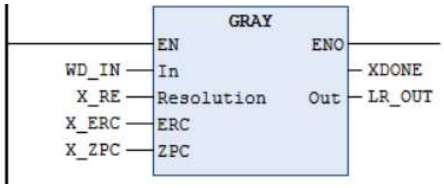
(4) Note

An exception will occur in the following situations. The return value of the function becomes FALSE, and "Out" is initialized to 0.

- When the value of "ERC" is greater than the resolution specified in "Resolution".
- When the value of "ZPC" is greater than the resolution specified in "Resolution".
- When the value after converting "In" to a rank is less than the value of "ERC".
- When the value corrected by "ERC" is greater than the resolution specified in "Resolution".

(5) Program example

DEMO	LD	ST
Define variable	VAR WD_IN: WORD:= 16#256; X_RE: _eGRY_RESOLUTION:= _eGRY_RESOLUTION._R10B; X_ERC: UINT:= 0; X_ZPC: UINT:= 2; LR_OUT: LREAL; XDONE: BOOL; END_VAR	

Program example		XDONE:=GRAY(In:= WD_IN, Resolution:= X_RE, ERC:= X_ERC, ZPC:= X_ZPC, Out=> LR_OUT);
Operation result		

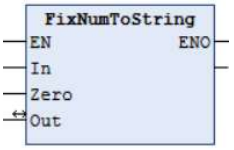
表达式	类型	值
WD_IN	WORD	598
X_RE	_EGRY_RESOLUTION	_R10B
X_ERC	UINT	0
X_ZPC	UINT	2
LR_OUT	LREAL	323.7890625
XDONE	BOOL	TRUE

1-4-7-32. Convert fixed decimal point number to decimal strings FixNumToString

(1) Instruction overview

Convert signed fixed decimal points to decimal string format.

Convert fixed decimal point number to decimal strings [FixNumToString]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	XS: V2.2.0 and up XSA: -	Software	XS StudioV2.2.0 and up (XS models)
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
FixNumToString	Convert fixed decimal point number to decimal strings		XDONE:=FixNumToString(In:= , Zero:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object	DWORD	As the data type	0	Signed fixed decimal point number
Zero	Zero display	BOOL	TRUE/FALSE	FALSE	Display when the number of decimal places is less than 3 True: Fill in "0" FALSE: Do not add "0"
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	STRING	As the data type	-	10 base string

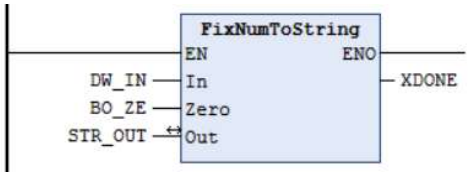
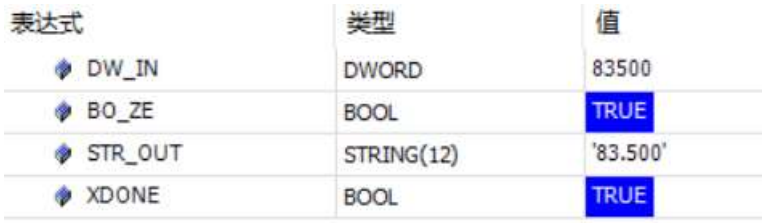
(3) Function

- Convert the signed fixed decimal point number "In" to a decimal string. The conversion steps are as follows:
  ① Convert "In" in hexadecimal format to decimal format.
  ② Divide the value by 1000.
- When the number of decimal places of "In" is less than 3, zero is displayed. "Zero" specifies whether to fill in "0" before the 3rd decimal place of "Out". If the value of "Zero" is true, then fill in "0"; If it is FALSE, do not fill in "0".
- "Out" ends with a NULL character.

(4) Note

- When the conversion result exceeds the valid range of "Out", the function returns a value of FALSE and "Out" remains unchanged.

(5) Program example

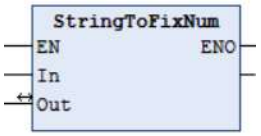
DEMO	LD	ST
Define variable	VAR DW_IN: DWORD:= 16#1462C; BO_ZE: BOOL:= TRUE; STR_OUT: STRING(12); XDONE: BOOL; END_VAR	
Program example		XDONE:=FixNumToString(In:= DW_IN, Zero:= BO_ZE, Out:= STR_OUT);
Operation result		

1-4-7-33. String convert to fixed decimal point number StringToFixNum

(1) Instruction overview

Convert a decimal string to a signed fixed decimal point format.

String convert to fixed decimal point number [StringToFixNum]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
StringToFixNum	String convert to fixed decimal point number		XDONE:=StringToFixNum(In:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object	STRING	As the data type	0	10 base string
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	DWORD	As the data type	-	Fixed decimal places number

(3) Function

- Convert the decimal string "In" to a fixed number of decimal points. The conversion steps are as follows:

① Multiply the value represented by "In" by 1000.
② Round off the decimal point of this value.
③ Convert the value to 32-bit hexadecimal format (DWORD type).

Name	Format
Symbol part	◆ Ignore a single '+' or '-' that appears after the consecutive ' ' (spaces 16#20) that appear at the beginning of the string. ◆ Compliance can be omitted. ◆ Simultaneously ignore consecutive spaces after symbols.
Integer part	◆ Take the consecutive numerical values ('0' to '9') from the symbol to the decimal point as the integer part. Sometimes symbols are also omitted. Sometimes there may be consecutive spaces between the symbol and integer parts. ◆ When there is no decimal point or decimal part, the exponential part is preceded by the integer part. ◆ When there is no decimal point, decimal and exponential parts, the integer part is until the end of the string. ◆ The integer part cannot be omitted. ◆ The maximum number of digits in the integer part is the maximum string length of 1986 subtracting the symbol, decimal point, decimal part, exponential part, and total bytes of space before and after the symbol.
Decimal point	◆ The single '.' after the integer part (dot) is a decimal point. ◆ When there is no decimal part, please omit the decimal point.
Decimal part	◆ Take the consecutive numerical values ('0' to '9') from the decimal point to the exponential part as the decimal part. ◆ When there is no exponential part, it is the decimal part until the end of the string. ◆ Decimal parts can be omitted. When there is no decimal point, there is also no decimal part. ◆ The maximum number of decimal places is 15.
Exponent part	◆ The single 'e' or 'E' after the decimal part, and the consecutive numerical values ('0' to '9') from the single '+' or ' - ' to the end of the string are the exponential part. ◆ When there is no decimal part, the string after the decimal point is the exponential part. ◆ When there is no decimal point or decimal part, the above string after the integer is the exponential part. ◆ The exponential part can be omitted. ◆ The maximum number of digits in the exponential part is 3.

(4) Note

- Round off after 4 decimal places for "In".
- Ignoring the underline (16#5F) in the "In" string.
- When "In" does not end with a NULL character, the function returns a value of FALSE while "Out" remains unchanged.
- When the content of "In" is a character that cannot be converted to a numerical value, the function returns a value of FALSE while "Out" remains unchanged.
- When the content of "In" has a decimal point but no decimal part, the function returns a value of FALSE while "Out" remains unchanged.

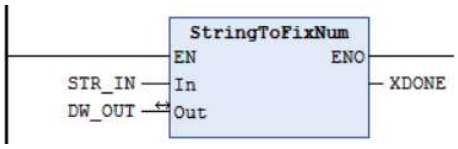
- When the conversion result exceeds the valid range of "Out", the function returns a value of FALSE and "Out" remains unchanged.
- In the symbol section, the "+" sign can be omitted, and negative numbers are used for overflow processing. When two or more symbols appear, the runtime function returns FALSE.
- When there are spaces in the middle and at the end of the input data, the function returns a value of FALSE and "Out" remains unchanged.

(5) Progarm example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   STR_IN: STRING(1985);
   DW_OUT: DWORD;
   XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:=StringToFixNum(
    In:= STR_IN,
    Out:= DW_OUT);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
STR_IN	STRING(1985)	'2.2563'
DW_OUT	DWORD	2256
XDONE	BOOL	TRUE

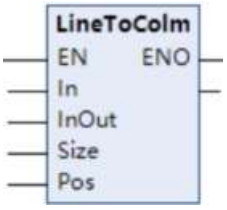
</td></tr>
</table>

1-4-7-34. Row/column conversion LineToColm

(1) Instruction overview

Decompose the bit string and output it to the specified bit position of the array element.

Row/column conversion [LineToColm]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
LineToColm	Row/column conversion		XDONE:=LineToColm(In:= , InOut:= , Size:= , Pos:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object	BYTE/WORD DWORD/LWORD	As the data type	-	Convert object
Size	The number of elements in the conversion result	UINT	As the data type	1	The number of elements in the conversion result
Pos	Conversion bit position	UINT	As the data type	0	The position to be converted
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
InOut	Conversion result array	BYTE/WORD DWORD/LWORD	As the data type	-	Conversion result array

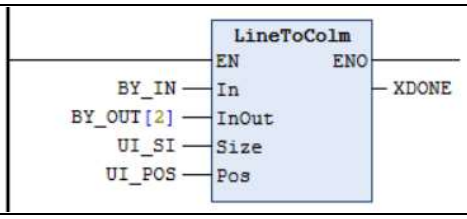
(3) Function

- Decompose the bit string and output it to the specified bit position of the array element.
- Firstly, extract the "Size" bit from the lowest bit of the transformation object "In" and decompose it into each bit.
- Then, save it in the "Pos" position of each element starting from InOut[0] in the conversion result array InOut[]. There are "Size" saved array elements.
- Unsaved value bits are used to maintain the value.

(4) Note

- When the value of "Size" is 0, the value of InOut[] remains unchanged, and the function returns a value of True.
- When the value of "Size" exceeds the valid range, the value of InOut[] remains unchanged, and the function returns a value of FALSE.
- When the value of "Pos" exceeds the valid range, the value of InOut[] remains unchanged, and the function returns a value of FALSE.
- When the value of "Size" exceeds the array area of InOut[], it is necessary to avoid PLC exceptions.

(5) Program example

DEMO	LD	ST
Define variable		VAR BY_IN: BYTE:= 173; BY_OUT: ARRAY[0..5] OF BYTE; UI_SI: UINT:= 3; UI_POS: UINT:= 1; XDONE: BOOL; END_VAR
Program example		XDONE:=LineToColm(In:= BY_IN, InOut:= BY_OUT[2], Size:= UI_SI, Pos:= UI_POS);
Operation result		

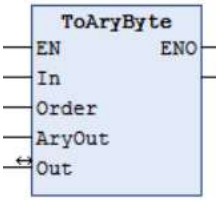
表达式	类型	值
BY_IN	BYTE	2#10101101
BY_OUT	ARRAY [0..5] OF BYTE	
BY_OUT[0]	BYTE	2#00000000
BY_OUT[1]	BYTE	2#00000000
BY_OUT[2]	BYTE	2#00000010
BY_OUT[3]	BYTE	2#00000000
BY_OUT[4]	BYTE	2#00000010
BY_OUT[5]	BYTE	2#00000000
UI_SI	UINT	2#0000000000000011
UI_POS	UINT	2#0000000000000001
XDONE	BOOL	TRUE

1-4-7-35. Convert to byte array ToAryByte

(1) Instruction overview

Divide variables by 1 byte and save them to a BYTE type array.

Convert to byte array [ToAryByte]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ToAryByte	Convert to byte array		XDONE:=ToAryByte(In:= , Order:= , AryOut:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object	BOOL/ BYTE/WORD/DWORD/LWORD / USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/ REAL/LREAL/ TIME/DATE/TOD/DT	As the data type	-	Convert object
Order	Conversion order	_eBYTE_ORDER	As the data type	_LOW_HIGH	The position to be converted
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
AryOut	Conversion result array	BYTE	As the data type	-	Conversion result array
Out	Conversion result element number	UINT	As the data type	-	Conversion result element number

(3) Function

- Split the conversion object "In" in units of 1 byte and save it sequentially from the AryOut[] array of conversion results, starting from AryOut [0]. Save the number of elements already saved in AryOut[] to the number of elements in the conversion result "Out".
- The order in which the value of "In" is converted in units of 1 byte is specified by the conversion order "Order". The data type of "Order" is the enumerator_eBYTEE_ORDER; The meaning of enumeration elements is as follows:

Enum	Meaning
_LOW_HIGH	Low byte before high byte
_HIGH_LOW	High byte before low byte

- When the data type of "In" is more than 2 bytes in size; Divide "In" by 1 byte and save it in AryOut[]. The data types with a size of 2 bytes or more are as follows:

Type	Data type
Bit string	WORD, DWORD, LWORD
Integer	UINT, UDINT, ULINT, INT, DINT, LINT
Real number	REAL, LREAL
Time, duration, date, string	TIME, DATE, TOD, TD

- The processing steps are as follows:
  ① Firstly, divide the value of "In" in units of 1 word (2 bytes);
  ② Divide the word with the lowest bit by 1 byte (2 bytes);
  ③ If the value of "Order" is _LOW_HIGH, save the low byte in AryOut[0] and the high byte in AryOut[1]. If the value of "Order" is _HIGH_LOW, save the high byte in AryOut[0] and the low byte in AryOut[1].
  ④ Similarly, divide the next word by 1 byte and save it in AryOut[2] and AryOut[3]. Perform the same processing until the end of the In value.

- When the data type of "In" is 1 byte, it is still saved in AryOut[] in units of 1 byte. The data types with a size of 1 byte are as follows:

Type	Data type
Bit string	BYTE
Integer	USINT, SINT
Real number	None
Time, duration, date, string	None

- The storage method is as follows:

“Order” value	Method to save to "AryOut[]"
LOW_HIGH	Save the value of "In" in AryOut[0]
HIGH_LOW	Save the value of "In" in AryOut[1]; Save 16#00 in AryOut[0].

- When the data type of "In" is BOOL (1 bit), the method to save to AryOut[] is as follows:

“Order” value	Method to save to "AryOut[]"
LOW_HIGH	Save the value of "In" with the logical sum of 16#00 in AryOut[0]
HIGH_LOW	Connect the value of In[0] and save it in AryOut[0]; Connect the value of In[1], save it in AryOut[1], and then use the same method to save it. Be sure to set the value of "Out" to an even number.

(4) Note

- Please make sure to set the input parameters transmitted to "In" as variables. If a constant is transmitted, an exception will occur during compilation.
- When the value of "Order" is _HIGH_LOW and the total size of "In" is odd bytes, add 16#00 at the end of "In", set it to even bytes, and then start the conversion.

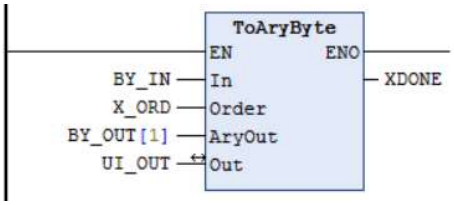
- When the value of "Order" exceeds the valid range, the function returns FALSE, while "Out" and "AryOut[] remain unchanged.
- When the conversion result exceeds the array area of AryOut[], it is necessary to avoid PLC exceptions.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   BY_IN: DWORD:= 16#5121C168;
   X_ORD: _eBYTE_ORDER:=_eBYTE_ORDER._LOW_HIGH;
   BY_OUT: ARRAY[0..5] OF BYTE;
   UI_OUT: UINT;
   XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:=ToAryByte(
    In:= BY_IN,
    Order:= X_ORD,
    AryOut:= BY_OUT[1],
    Out:= UI_OUT);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
BY_IN	DWORD	16#5121C168
X_ORD	_EBYTE_ORDER	_LOW_HIGH
BY_OUT	ARRAY [0..5] OF BYTE	
BY_OUT[0]	BYTE	16#00
BY_OUT[1]	BYTE	16#68
BY_OUT[2]	BYTE	16#C1
BY_OUT[3]	BYTE	16#21
BY_OUT[4]	BYTE	16#51
BY_OUT[5]	BYTE	16#00
UI_OUT	UINT	16#0004
XDONE	BOOL	TRUE

</td></tr>
</table>

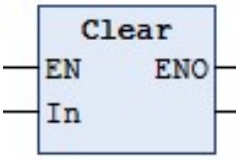
1-4-8. Initialization instruction

1-4-8-1. Initialization Clear

(1) Instruction overview

Initialize variables.

Initialization [Clear]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
Clear	Initialization		`Clear(In:= );`

(2) Related variables

VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
In	Initialize Object	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/LINT/ REAL/LREAL/TIME/DATE/ TOD/DT/STRING	As the data type	-	Initialize Object

(3) Function

- Initialize the value of the initialization object "In" to the default initial value of each data type.
- The default initial values for each data type are shown in the table below:

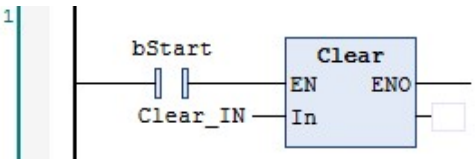
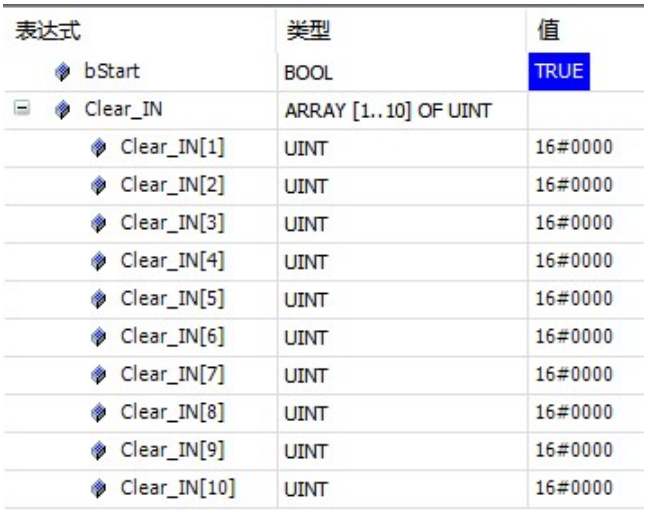
Data type	Initial value
BOOL	FALSE
BYTE, WORD, DWORD, LWORD	16#0
USINT, UINT, UDINT, ULINT, SINT, INT, DINT, LINT, REAL, LREAL	0
TIME	T#0ms
DATE	D#1970-1-1
TOD	TOD#0:0:0
DT	DT#1970-1-1-0:0:0
STRING	‘’

- When "In" represents the entire array, with one element and one variable in the array, the following table shows the processing:

“In”	Processing content
The entire array	Initialize all elements of an array
An element of an array	Only initialize this element
The entire structure	Initialize all structural elements of the structure
One structural element of a	Only initialize this structural element

structure	

(4) Program example

DEMO	LD	ST
Define variable	VAR bStart:BOOL; Clear_IN:ARRAY[1..10] OF UINT:=[66,77,52,3,5,789,56,366,32,75]; END_VAR	
Program example		1 Clear(In:= Clear_IN);
Operation result		

表达式	类型	值
bStart	BOOL	TRUE
Clear_IN	ARRAY [1..10] OF UINT	
Clear_IN[1]	UINT	16#0000
Clear_IN[2]	UINT	16#0000
Clear_IN[3]	UINT	16#0000
Clear_IN[4]	UINT	16#0000
Clear_IN[5]	UINT	16#0000
Clear_IN[6]	UINT	16#0000
Clear_IN[7]	UINT	16#0000
Clear_IN[8]	UINT	16#0000
Clear_IN[9]	UINT	16#0000
Clear_IN[10]	UINT	16#0000

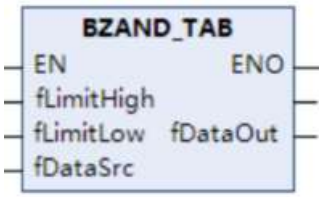
1-4-9. Regional control instructions

1-4-9-1. Dead zone control command BZAND_TAB

(1) Instruction overview

Determine whether the input value is within the specified upper and lower limits of the dead zone, in order to control the output value.

Dead zone control command [BZAND_TAB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
BZAND_TAB	Dead zone control command		XDONE:=BZAND_TAB(fLimitHigh:= , fLimitLow:= , fDataSrc:= , fDataOut=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
fLimitHigh	Dead zone upper limit value	REAL	As the data type	0	Dead zone upper limit value
fLimitLow	Dead zone lower limit value	REAL	As the data type	0	Dead zone lower limit value
fDataSrc	Source data	REAL	As the data type	0	Input value through dead zone
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
fDataOut	Output data	REAL	As the data type	-	Output value through dead zone

(3) Function

- Control the output value by determining whether the input value is within the specified upper and lower limits of the dead zone; Set the dead zone range in fLimitLow, fLimitHigh to output the input value fDatasrc outside the dead zone range to fDataOut.
- When fLimitLow>fDatasrc, fDataOut = fDataSrc - fLimitLow；
- When fLimitHigh<fDatasrc, fDataOut = fDataSrc - fLimitHigh；
- When fLimitLow ≤ fDataSrc ≤ fLimitHigh, fDataOut = 0.

(4) Note

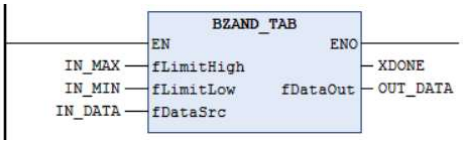
- When fDataSrc input is a non numeric value, FDataOut output is 0.
- When fLimitLow is a non numerical value and fDataSrc>fLimitHigh, When fDataOut=fDataSrc-fLimitHigh; fDataSrc<fLimitHigh, FDataOut is 0.

- When fLimitHigh is a non numerical value and fDataSrc<fLimitLow, When fDataOut=fDataSrc fLimitLow; fDataSrc>fLimitLow, FDataOut is 0.
- When fLimitHigh, fLimitLow are all non numeric, FDataOut is 0.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
    IN_MAX: REAL:= 10;
    IN_MIN: REAL:= 1;
    IN_DATA: REAL:= -2;
    OUT_DATA: REAL;
    XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td> 
</td><td>

```
XDONE:= BZAND_TAB(
    fLimitHigh:= IN_MAX,
    fLimitLow:= IN_MIN,
    fDataSrc:= IN_DATA,
    fDataOut=> OUT_DATA);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN_MAX	REAL	10
IN_MIN	REAL	1
IN_DATA	REAL	-2
OUT_DATA	REAL	-3
XDONE	BOOL	TRUE

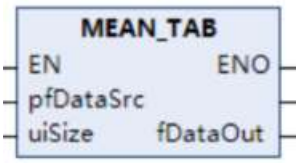
</td></tr>
</table>

1-4-9-2. Calculation of average data value MEAN_TAB

(1) Instruction overview

Calculate the average value of the specified data.

Calculation of average data value [MEAN_TAB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
MEAN_TAB	Calculation of average data value		XDONE:=MEAN_TAB(pfDataSrc:= , uiSize:= , fDataOut=>);

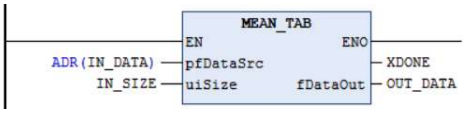
(2) Related variables

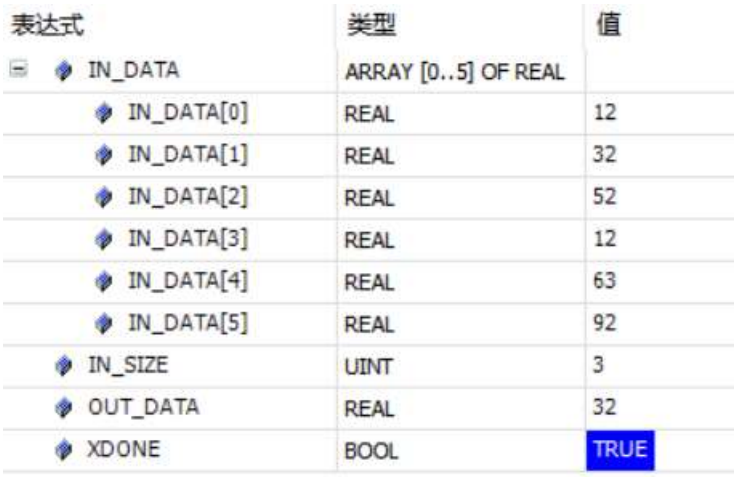
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
pfDataSrc	Source data	POINTER TO REAL	-	-	Input data address
uiSize	Data length	UINT	1-1024	0	Data length
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
fDataOut	Output data	REAL	As the data type	-	Output average

(3) Function

- Perform the average operation on the number of uiSize starting from pfDataSrc, and store the result in fDataOut.

(4) Program example

DEMO	LD	ST
Define variable		VAR IN_DATA: ARRAY[0..5] OF REAL:= [12,32,52,12,63,92]; IN_SIZE: UINT:= 3; OUT_DATA: REAL; XDONE: BOOL; END_VAR
Program example		XDONE:= MEAN_TAB(pfDataSrc:= ADR(IN_DATA), uiSize:= IN_SIZE, fDataOut=> OUT_DATA);

Operation result	

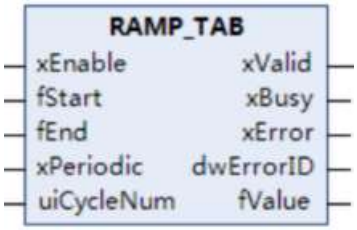
表达式	类型	值
IN_DATA	ARRAY [0..5] OF REAL	
IN_DATA[0]	REAL	12
IN_DATA[1]	REAL	32
IN_DATA[2]	REAL	52
IN_DATA[3]	REAL	12
IN_DATA[4]	REAL	63
IN_DATA[5]	REAL	92
IN_SIZE	UINT	3
OUT_DATA	REAL	32
XDONE	BOOL	TRUE

1-4-9-3. Slope command RAMP_TAB

(1) Instruction overview

Slope command, with a specified interval, the output value changes from the lowest value to the highest value, or from the highest value to the lowest value, performing linear interpolation between high and low values.

Slope command [RAMP_TAB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

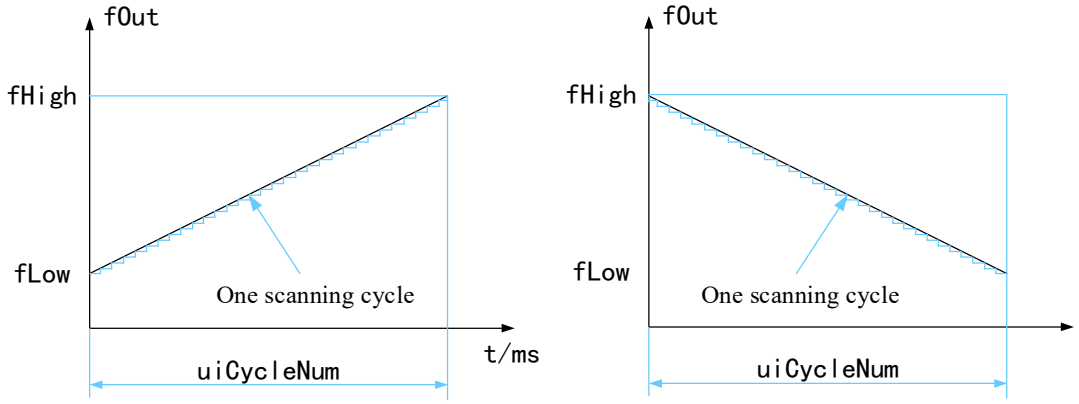
Instruction	Name	Graph	ST
RAMP_TAB	Slope command		RAMP_TAB_0(xEnable:= , fStart:= , fEnd:= , xPeriodic:= , uiCycleNum:= xValid=> , xBusy=> , xError=> , dwErrorID=> , fValue=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
xEnable	Execute	BOOL	FALSE/TRUE	FALSE	Input high-level start command
fStart	Start value	REAL	As the data type	0	Start value
fEnd	End value	REAL	As the data type	0	End value
xPeriodic	Cyclic output	BOOL	FALSE/TRUE	FALSE	FALSE: Execute once True: Loop execution
uiCycleNum	Number of completed cycles	UINT	As the data type	0	Number of completed cycles
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
xValid	Completed	BOOL	FALSE/TRUE	FALSE	True: Instruction execution completed FALSE: Instruction not completed
xBusy	Executing	BOOL	FALSE/TRUE	FALSE	True: Instruction execution in progress FALSE: Instruction not in progress
xError	Error	BOOL	FALSE/TRUE	FALSE	True: Instruction execution error FALSE: No error occurred during instruction execution
dwErrorID	Error code	DWORD	-	0	Error code
fValue	Current value	REAL	-	0	Current output value

(3) Function

- The function of the instruction is to perform linear interpolation between two given data points within a specified time interval, and output process values sequentially according to the scanning execution time until the endpoint value at the end of the interval.



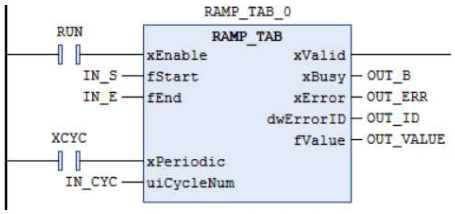
(4) Note

- xEnable changes from FALSE to True, The fValue value starts to change again, xBusy is True, after the run ends, xValid, xBusy is True.
- After xEnable changes from True to False, fValue value is maintained.
- Only xEnable is False, fStart, fEnd, xPeriodic, uiCycleNum can be written and take effect when xEnable is true.
- When xEnable is False, the value of fValue can be modified, but it does not take effect.
- When xEnable is True, the values of all input and output variables can be modified, but it cannot take effect.
- When uiCycleNum is set to 0, When xEnable is True, XError is True, dwErrorID is 1, xValid, xBusy is False.

(5) Program example

DEMO	LD	ST
Define variable		VAR RAMP_TAB_0: RAMP_TAB; RUN: BOOL; IN_S: REAL:= 5; IN_E: REAL:=100; XCYC: BOOL; IN_CYC: UINT:=95; OUT_B: BOOL; OUT_ERR: BOOL; OUT_ID: DWORD; OUT_VALUE: REAL; XDONE: BOOL; END_VAR

Program example




```
RAMP_TAB_0(
   xEnable:= RUN,
   fStart:= IN_S,
   fEnd:= IN_E,
   xPeriodic:= XCYC,
   uiCycleNum:= IN_CYC,
   xValid=> XDONE,
   xBusy=> OUT_B,
   xError=> OUT_ERR,
   dwErrorID=> OUT_ID,
   fValue=> OUT_VALUE);
```

Operation result

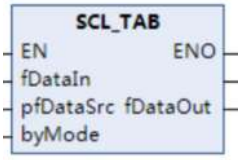
表达式	类型	值
⊞ RAMP_TAB_0	RAMP_TAB	
RUN	BOOL	TRUE
IN_S	REAL	5
IN_E	REAL	100
XCYC	BOOL	FALSE
IN_CYC	UINT	95
OUT_B	BOOL	TRUE
OUT_ERR	BOOL	FALSE
OUT_ID	DWORD	0
OUT_VALUE	REAL	100
XDONE	BOOL	TRUE

1-4-9-4. Table coordinate acquisition instruction SCL_TAB

(1) Instruction overview

Pre store table data, input values, and then obtain corresponding output values based on the table data.

Table coordinate acquisition instruction [SCL_TAB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
SCL_TAB	Table coordinate acquisition instruction		XDONE:=SCL_TAB(fDataIn:= , pfDataSrc:= , byMode:= , fDataOut=>);

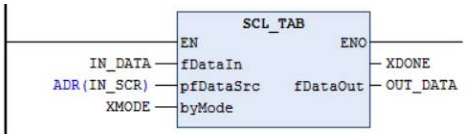
(2) Related variables

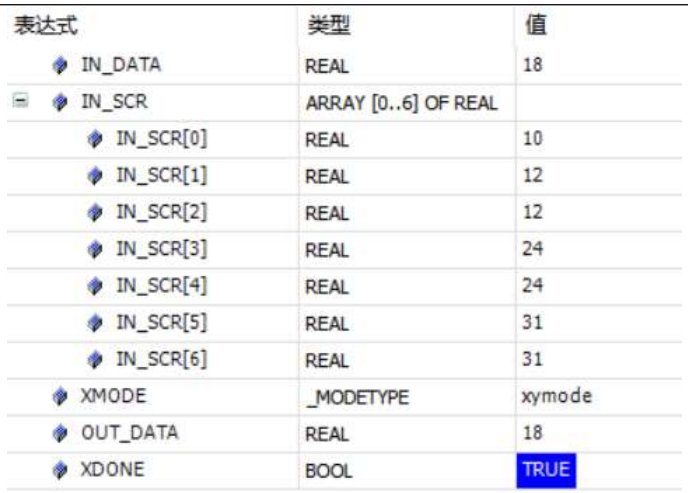
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
fDataIn	Input value	REAL	As the data type	0	Input data
pfDataSrc	Source data	POINTER TO REAL	-	-	Table data
byMode	Mode selection	_modeType	1/2	_xymode	1：x1,y1;x2,y2;....xn,yn 2：x1,x2,...xn;y1,y2...yn
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
fDataOut	Output value	REAL	-	-	Output value obtained by referring to table data

(3) Function

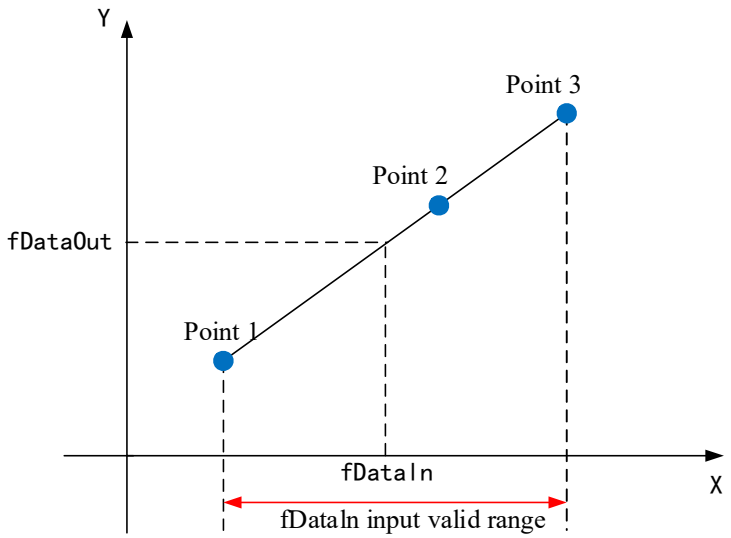
- Based on the graph determined by the table in pfDatasrc, find the output value fDataOut corresponding to the input value fDataIn.

(4) Program example

DEMO	LD	ST
Define variable	VAR IN_DATA: REAL; IN_SCR: ARRAY[0..6] OF REAL:= [10,12,12,24,24,31,31]; XMODE: _modeType:= 1; OUT_DATA: REAL; XDONE: BOOL; END_VAR	
Program example		XDONE:= SCL_TAB(fDataIn:= IN_DATA, pfDataSrc:=ADR(IN_SCR), byMode:= XMODE, fDataOut=> OUT_DATA);

Operation result	

The principle analysis of the example program is as follows:

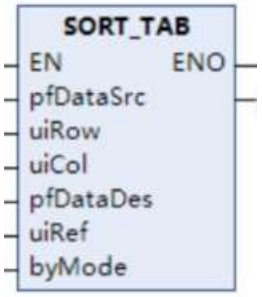
Principle analysis		
Table Data Diagram		
Source data (pfDataSrc)		Table data address
Coordinate point data value		(pfDataSrc)^
Point 1	X1 coordinate	(pfDataSrc+1)^
	Y1 coordinate	(pfDataSrc+2)^
Point 2	X2 coordinate	(pfDataSrc+3)^
	Y2 coordinate	(pfDataSrc+4)^
Point 3	X3 coordinate	(pfDataSrc+5)^
	Y3 coordinate	(pfDataSrc+6)^

1-4-9-5. Table data sorting instruction SORT_TAB

(1) Instruction overview

Sort the array of uiRow rows × uiCol columns by the uiRef column or row parameter, and store the sorting results in the variable area starting with the pfDataDes unit.

Table data sorting instruction [SORT_TAB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
SORT_TAB	Table data sorting instruction		XDONE:=SORT_TAB(pfDataSrc:= , uiRow:= , uiCol:= , pfDataDes:= , uiRef:= , byMode:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
pfDataSrc	Source data	POINTER TO REAL	-	-	Arrange data objects
uiRow	Row	UINT	1-1024	0	Data row
uiCol	column	UINT	1-1024	0	Data column
pfDataDes	Output data	POINTER TO REAL	-	-	Data output
uiRef	Row/column	UINT	1-1024	0	Based on rows or columns
byMode	Arrangement method	BYTE	1-4	0	Arrangement method 1: Column reference ascending order 2: Column reference descending order 3: Row reference ascending order 4: Row Reference Descending

(3) Function

- pfDatasrc is the starting unit of the first variable in the first row (or the first record).
- The number of rows in the uiRow array, also known as the number of records.
- The number of columns in the uiCol array, also known as the number of columns per record.
- The starting unit stored after sorting in pfDataDes occupies the same number of subsequent variable units as the number of array variables before sorting.
- uiRef is a row/column number based on sorting.
- byMode sorting mode 1: column reference ascending, 2: column reference descending, 3: row reference

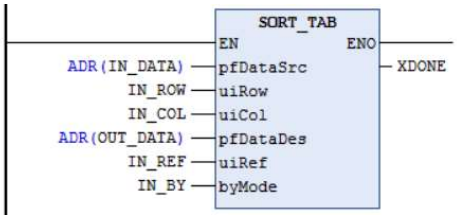
ascending, 4: row reference descending.

(4) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   IN_DATA: ARRAY[0..8] OF REAL:= [1,2,3,4,5,6,7,8,9];
   IN_ROW: UINT:= 3;
   IN_COL: UINT:= 3;
   OUT_DATA: ARRAY[0..8] OF REAL;
   IN_REF: UINT:=2;
   IN_BY: BYTE:= 2;
   XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:= SORT_TAB(
    pfDataSrc:= ADR(IN_DATA),
    uiRow:= IN_ROW,
    uiCol:= IN_COL,
    pfDataDes:= ADR(OUT_DATA),
    uiRef:= IN_REF,
    byMode:= IN_BY);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN_DATA	ARRAY [0..8] OF REAL	
IN_DATA[0]	REAL	1
IN_DATA[1]	REAL	2
IN_DATA[2]	REAL	3
IN_DATA[3]	REAL	4
IN_DATA[4]	REAL	5
IN_DATA[5]	REAL	6
IN_DATA[6]	REAL	7
IN_DATA[7]	REAL	8
IN_DATA[8]	REAL	9
IN_ROW	UINT	3
IN_COL	UINT	3
OUT_DATA	ARRAY [0..8] OF REAL	
OUT_DATA[0]	REAL	3
OUT_DATA[1]	REAL	2
OUT_DATA[2]	REAL	1
OUT_DATA[3]	REAL	6
OUT_DATA[4]	REAL	5
OUT_DATA[5]	REAL	4
OUT_DATA[6]	REAL	9
OUT_DATA[7]	REAL	8
OUT_DATA[8]	REAL	7
IN_REF	UINT	2
IN_BY	BYTE	2
XDONE	BOOL	TRUE

</td></tr>
</table>

- The principle analysis of the example program is as follows: perform descending processing on the second column.

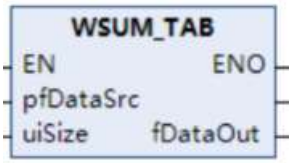
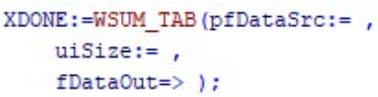
Data before execution			
-	Column 1	Column 2	Column 3
Row 1	pfDataSrc[0]=1	pfDataSrc[3]=4	pfDataSrc[6]=7
Row 2	pfDataSrc[1]=2	pfDataSrc[4]=5	pfDataSrc[7]=8
Row 3	pfDataSrc[2]=3	pfDataSrc[5]=6	pfDataSrc[8]=9
Data after execution			
-	Column 1	Column 2	Column 3
Row 1	pfDataSrc[0]=3	pfDataSrc[3]=6	pfDataSrc[6]=9
Row 2	pfDataSrc[1]=2	pfDataSrc[4]=5	pfDataSrc[7]=8
Row 3	pfDataSrc[2]=1	pfDataSrc[5]=4	pfDataSrc[8]=7

1-4-9-6. Data calculation total instruction WSUM_TAB

(1) Instruction overview

Calculate the total value of a specified amount of table data.

Data calculation total instruction [WSUM_TAB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
WSUM_TAB	Data calculation total instruction		


```
XDONE:=WSUM_TAB(pfDataSrc:= ,
    uiSize:= ,
    fDataOut=> );
```

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
pfDataSrc	Source data	POINTER TO REAL	-	-	Input data address
uiSize	Data length	UINT	1-1024	0	Data length
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
fDataOut	Output data	REAL	As the data type	-	Output total value

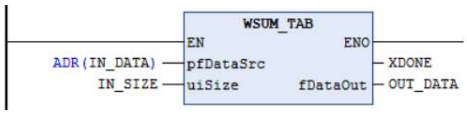
(3) Function

- Calculate the total length of uiSize data in the input data pfDataSrc and output it to fDataOut.

(4) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
   IN_DATA: ARRAY[0..5] OF REAL:= [12,16,22,10,17,31];
   IN_SIZE: UINT:= 3;
   OUT_DATA: REAL;
   XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td> 
</td><td>

```
XDONE:= WSUM_TAB(
    pfDataSrc:= ADR(IN_DATA),
    uiSize:= IN_SIZE,
    fDataOut=> OUT_DATA);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN_DATA	ARRAY [0..5] OF REAL	
IN_DATA[0]	REAL	12
IN_DATA[1]	REAL	16
IN_DATA[2]	REAL	22
IN_DATA[3]	REAL	10
IN_DATA[4]	REAL	17
IN_DATA[5]	REAL	31
IN_SIZE	UINT	3
OUT_DATA	REAL	50
XDONE	BOOL	TRUE

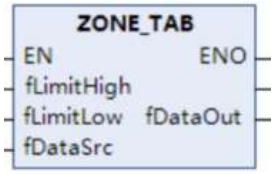
</td></tr>
</table>

1-4-9-7. Zone control instruction ZONE_TAB

(1) Instruction overview

Control the output value with the specified deviation value based on whether the input value is positive or negative (add the deviation value to the input value).

Zone control instruction [ZONE_TAB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
ZONE_TAB	Zone control instruction		XDONE:=ZONE_TAB(fLimitHigh:= , fLimitLow:= , fDataSrc:= , fDataOut=>);

(2) Related variables

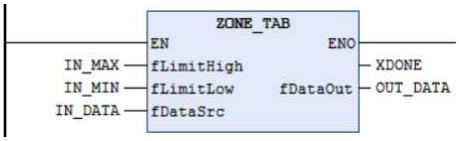
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
fLimitHigh	Positive deviation	REAL	As the data type	0	Positive deviation, the deviation value added to the input value
fLimitLow	Negative deviation	REAL	As the data type	0	Negative deviation, the deviation value added to the input value
fDataSrc	Input value	REAL	As the data type	0	Source data
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
fDataOut	Output value	REAL	As the data type	-	Output value

(3) Function

- Determine the sign of the input value fDataSrc, add fLimitHigh or fLimitLow, and save the execution result to fDataOut.
- When fDataSrc<0, fDataOut = fDataSrc + fLimitLow；
- When fDataSrc>0, fDataOut = fDataSrc + fLimitHigh；
- When fDataSrc=0, fDataOut = 0.

(4) Program example

DEMO	LD	ST
Define variable		VAR IN_MAX: REAL:= 6; IN_MIN: REAL:= 2; IN_DATA: REAL; OUT_DATA: REAL; XDONE: BOOL; END_VAR

Program example		XDONE:= ZONE_TAB(fLimitHigh:= IN_MAX, fLimitLow:= IN_MIN, fDataSrc:= IN_DATA, fDataOut=> OUT_DATA);
Operation result	表达式 / 类型 / 值 IN_MAX / REAL / 6 IN_MIN / REAL / 2 IN_DATA / REAL / -5 OUT_DATA / REAL / -3 XDONE / BOOL / TRUE	

1-4-9-8. Batch data reset instruction ZRST_TAB

(1) Instruction overview

Batch reset the data in the table.

Batch data reset instruction [ZRST_TAB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
ZRST_TAB	Batch data reset instruction		XDONE:=ZRST_TAB(pbyData:= , uiSize:=);

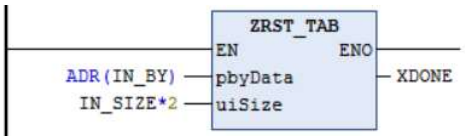
(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
pbyData	Source data	POINTER TO BYTE	-	-	Data header address
uiSize	Data length	UINT	1-1024	0	Data length

(3) Function

- Batch reset input source data of specified length.

(4) Program example

DEMO	LD	ST
Define variable	VAR IN_BY: ARRAY[0..5] OF WORD:= [9,5,3,2,4,6]; IN_SIZE: UINT:= 5; XDONE: BOOL; END_VAR	
Program example		XDONE:= ZRST_TAB(pbyData:= ADR(IN_BY), uiSize:= IN_SIZE*2);

Operation result	

表达式	类型	值
IN_BY	ARRAY [0..5] OF WO...	
IN_BY[0]	WORD	0
IN_BY[1]	WORD	0
IN_BY[2]	WORD	0
IN_BY[3]	WORD	0
IN_BY[4]	WORD	0
IN_BY[5]	WORD	6
IN_SIZE	UINT	5
XDONE	BOOL	TRUE

1-5. Operation instructions

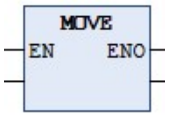
1-5-1. Assignment instruction

1-5-1-1. Assignment instruction MOVE

(1) Instruction overview

Function: Assign the value of a constant or variable to another variable.

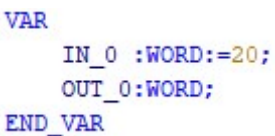
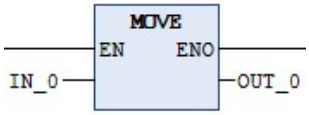
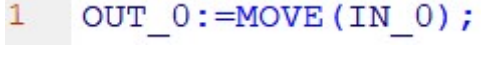
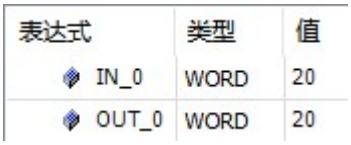
Assignment instruction [MOVE]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
MOVE	Assignment instruction		MOVE

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Send data	BOOL/BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME/DATE/TOD/DT/STRING	As the data type	-	Variables for transmitting data
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Receive data	BOOL/BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME/DATE/TOD/DT/STRING	As the data type	-	Variables receiving data

(3) Program example

DEMO	LD	ST
Define variable		
Program example		
Operation result		

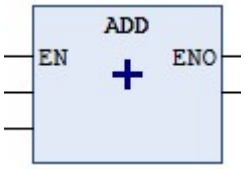
1-5-2. Arithmetic operation instructions

1-5-2-1. Addition operation ADD

(1) Instruction overview

Function: Addition operation instruction, adding two (or more) variables or constants. Two time variables can also be added, resulting in another time variable.

Addition operation [ADD]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
ADD	Addition operation		+

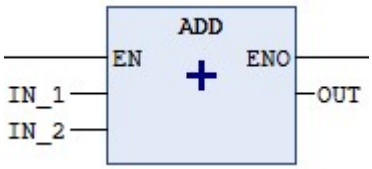
(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN1	Addend 1	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME	As the data type	-	Addend 1
IN2	Addend 2	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME	As the data type	-	Addend 2
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Sum	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME	As the data type	-	Sum

(3) Program example

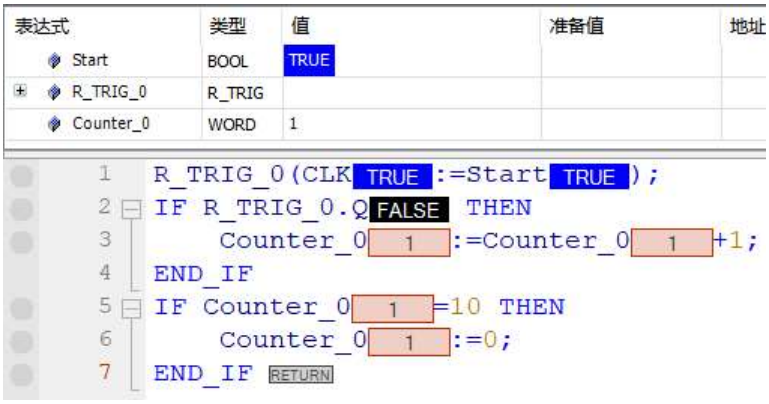
① Add the values of two INT variables:

DEMO	LD	ST
Define variable		VAR IN_1:INT:=10; IN_2:INT:=20; OUT :INT; END_VAR

Program example		1 OUT:=IN_1+IN_2;
Operation result		

表达式	类型	值
IN_1	INT	10
IN_2	INT	20
OUT	INT	30

② Using the ST programming language, the rising edge trigger function block accumulates the addend Counter_0. When the number accumulates to 10, the cumulative variable is reset to zero.

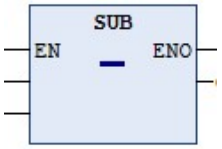
DEMO	ST
Define variable	VAR Start : BOOL; R_TRIG_0 :R_TRIG; Counter_0:WORD; END_VAR
Program example	1 R_TRIG_0(CLK:=Start); 2 IF R_TRIG_0.Q THEN 3 Counter_0:=Counter_0+1; 4 END_IF 5 IF Counter_0=10 THEN 6 Counter_0:=0; 7 END_IF
Operation result	

1-5-2-2. Subtraction operation SUB

(1) Instruction overview

Function: Subtraction operation instruction, subtracting two variables or constants.

Subtraction operation [SUB]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
SUB	Subtraction operation		-

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN1	Minuend	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME/TOD	As the data type	-	Minuend
IN2	Subtrahend	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME/TOD	As the data type	-	Subtrahend
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Difference	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME	As the data type	-	Difference

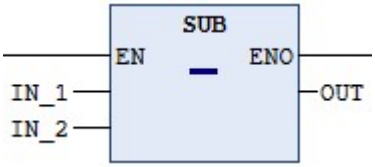
(3) Note

TOD type variables can also use subtraction function, where two TOD types are subtracted to obtain a new TIME type data.

Example: TOD#20:10:0- TOD#20:9:59=T#1s, but the time result cannot have a negative value.

(4) Program example

DEMO	LD	ST
Define variable		VAR IN_1:INT:=20; IN_2:INT:=3; OUT :INT; END_VAR

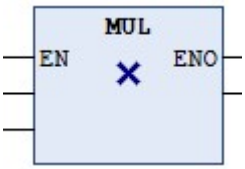
Program example		1 OUT:=IN_1-IN_2;
Operation result		

1-5-2-3. Multiplication operation MUL

(1) Instruction overview

Function: Multiplication operation instruction, multiplying two (or more) variables or constants.

Multiplication operation [MUL]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

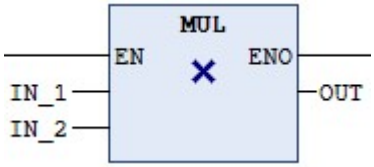
Instruction	Name	Graph	ST
MUL	Multiplication operation		*

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN1	Multiplicator 1	BYTE/WORD/DWORD/LWORD /USINT/UINT/UDINT/ULINT/SI NT/INT/DINT/LINT/REAL/LRE AL/TIME/TOD	As the data type	-	Multiplicator 1
IN2	Multiplicator 2	BYTE/WORD/DWORD/LWORD /USINT/UINT/UDINT/ULINT/SI NT/INT/DINT/LINT/REAL/LRE AL/TIME/TOD	As the data type	-	Multiplicator 2
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Product	BYTE/WORD/DWORD/LWORD	As the data	-	Product

		/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME/TOD	type		

(3) Program example

DEMO	LD	ST
Define variable	VAR IN_1:INT:=20; IN_2:INT:=3; OUT :INT; END_VAR	
Program example		1 OUT:=IN_1*IN_2;
Operation result		

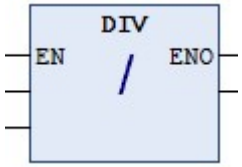
表达式	类型	值
IN_1	INT	20
IN_2	INT	3
OUT	INT	60

1-5-2-4. Division operation DIV

(1) Instruction overview

Function: Division operation instruction, dividing two variables or constants.

Division operation [DIV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
DIV	Division operation		/

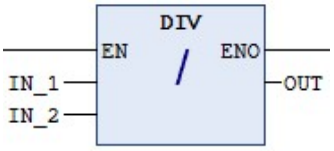
(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN1	Dividend	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SINT /INT/DINT/LINT/REAL/LREAL	As the data type	-	Dividend
IN2	Divisor	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SINT /INT/DINT/LINT/REAL/LREAL	As the data type	-	Divisor
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Quotient	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SINT /INT/DINT/LINT/REAL/LREAL	As the data type	-	Quotient

(3) Note

When using the DIV command in engineering, CheckDivByte, CheckDivWord, CheckDivDWord, and CheckDivReal can be used to check whether the divisor is zero, avoiding the phenomenon of divisors being zero.

(4) Program example

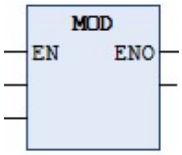
DEMO	LD	ST
Define variable	VAR IN_1:INT:=20; IN_2:INT:=5; OUT :INT; END_VAR	
Program example		1 OUT:=IN_1/IN_2;
Operation result		

1-5-2-5. Remainder operation MOD

(1) Instruction overview

Function: Divide variables or constants to obtain remainder, and the result is the remainder after dividing two numbers, which is an integer data.

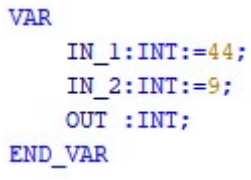
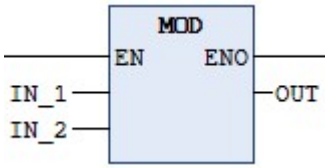
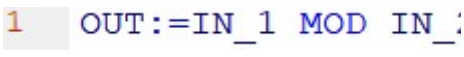
Remainder operation [MOD]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
MOD	Remainder operation		MOD

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN1	Dividend	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Dividend
IN2	Divisor	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Divisor
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Remainder	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Remainder

(3) Program example

DEMO	LD	ST
Define variable		
Program example		

Operation result	

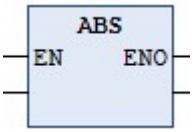
1-5-3. Mathematical operation instructions

1-5-3-1. Absolute value ABS

(1) Instruction overview

Function: This function instruction is used to calculate the absolute value of a number. It has nothing to do with the sign of positive and negative numbers.

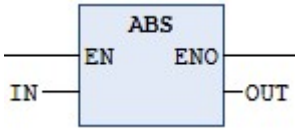
Absolute value [ABS]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
ABS	Absolute value		ABS

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input variables	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SIN T/INT/DINT/LINT/REAL/LREAL	As the data type	-	Parameters to be taken as absolute values
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SIN T/INT/DINT/LINT/REAL/LREAL	As the data type	-	Output value

(3) Program example

DEMO	LD	ST
Define variable	VAR IN :INT:=-12; OUT:INT; END_VAR	
Program example		1 OUT:=ABS(IN);
Operation result		

1-5-3-2. Square root SQRT

(1) Instruction overview

Function: The square root of non negative real numbers.

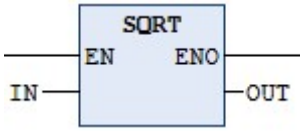
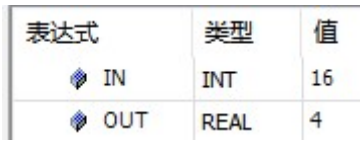
Square root [SQRT]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
SQRT	Square root		SQRT

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Calculation results	REAL/LREAL	As the data type	-	Result

(3) Program example

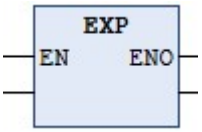
DEMO	LD	ST
Define variable	VAR IN :INT:=16; OUT:REAL; END_VAR	
Program example		1 OUT:=SQRT(IN);
Operation result		

1-5-3-3. Exponent EXP

(1) Instruction overview

Function: return the power of e (the base of Natural logarithm). e is a constant of 2.71828.

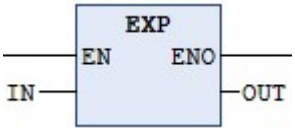
Exponent [EXP]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
EXP	Exponent		EXP

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

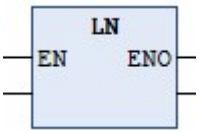
DEMO	LD	ST
Define variable	VAR IN :REAL:=2; OUT:REAL; END_VAR	
Program example		1 OUT:=EXP(IN);
Operation result		

1-5-3-4. Natural logarithm LN

(1) Instruction overview

Function: returns the Natural logarithm of a number. The base of Natural logarithm is the constant term e (2.71828182845904).

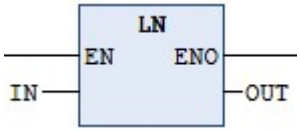
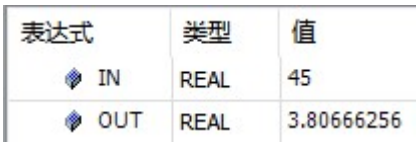
Natural logarithm [LN]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
LN	Natural logarithm		LN

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

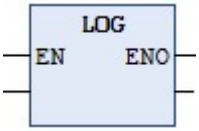
DEMO	LD	ST
Define variable	VAR IN :REAL:=45; OUT:REAL; END_VAR	
Program example		1 OUT:=LN(IN);
Operation result		

1-5-3-5. Logarithm with a base of 10 LOG

(1) Instruction overview

Function: Returns the logarithm of a base of 10.

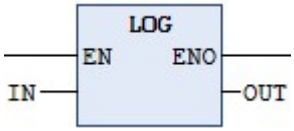
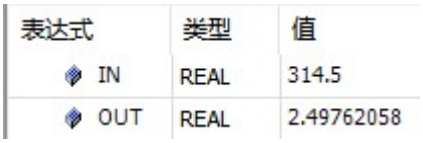
Logarithm with a base of 10 [LOG]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
LOG	Logarithm with a base of 10		LOG

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

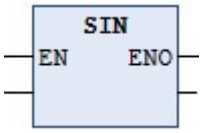
DEMO	LD	ST
Define variable	VAR IN :REAL:=314.5; OUT:REAL; END_VAR	
Program example		1 OUT:=LOG(IN);
Operation result		

1-5-3-6. Sine function SIN

(1) Instruction overview

Function: Sine function.

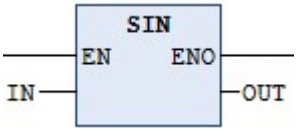
Sine function [SIN]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
SIN	Sine function		SIN

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SINT/ INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

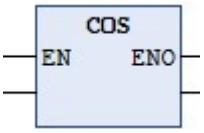
DEMO	LD	ST
Define variable	VAR IN :REAL:=0.5; OUT:REAL; END_VAR	
Program example		1 OUT:=SIN(IN);
Operation result	表达式 / 类型 / 值 IN / REAL / 0.5 OUT / REAL / 0.47942555	

1-5-3-7. Cosine function COS

(1) Instruction overview

Function: Cosine function.

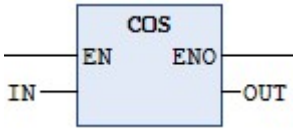
Cosine function [COS]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
COS	Cosine function		COS

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SIN T/INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

DEMO	LD	ST
Define variable	VAR IN :REAL:=0.5; OUT:REAL; END_VAR	
Program example		1 OUT:=COS(IN);
Operation result		

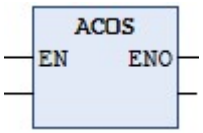
表达式	类型	值
IN	REAL	0.5
OUT	REAL	0.87758255

1-5-3-8. Arccosine function ACOS

(1) Instruction overview

Cosine radian (arccosine function).

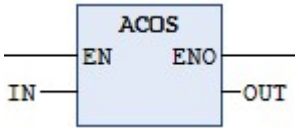
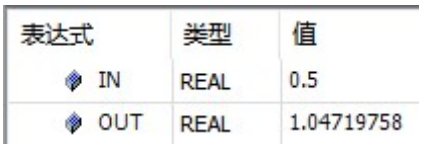
Arccosine function [ACOS]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
ACOS	Arccosine function		ACOS

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SINT /INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

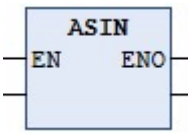
DEMO	LD	ST
Define variable	VAR IN :REAL:=0.5; OUT:REAL; END_VAR	
Program example		1 OUT:=ACOS(IN);
Operation result		

1-5-3-9. Arcsin function ASIN

(1) Instruction overview

Function: Sine radian (Arcsine function).

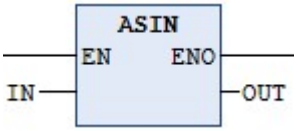
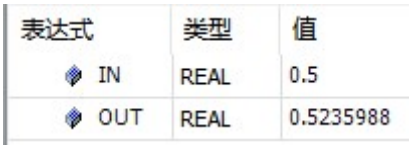
Arcsin function [ASIN]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
ASIN	Arcsin function		ASIN

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/SINT /INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

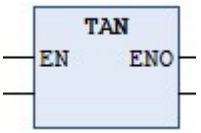
DEMO	LD	ST
Define variable	VAR IN :REAL:=0.5; OUT:REAL; END_VAR	
Program example		1 OUT:=ASIN(IN);
Operation result		

1-5-3-10. Tangent function TAN

(1) Instruction overview

Function: Tangent function.

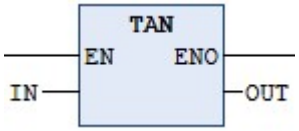
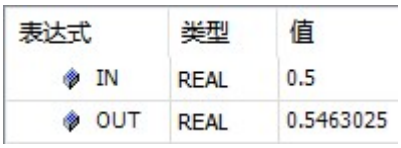
Tangent function [TAN]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
TAN	Tangent function		TAN

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

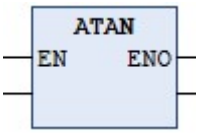
DEMO	LD	ST
Define variable	VAR IN :REAL:=0.5; OUT:REAL; END_VAR	
Program example		1 OUT:=TAN(IN);
Operation result		

1-5-3-11. Arctangent function ATAN

(1) Instruction overview

Function: Tangent radian (Arctangent function).

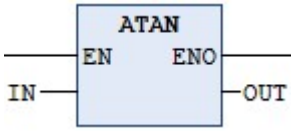
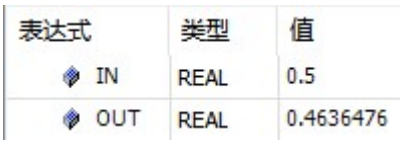
Arctangent function [ATAN]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
ATAN	Arctangent function		ATAN

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Input parameter	BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL	As the data type	-	Variable to be calculated
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Output value	REAL/LREAL	As the data type	-	Output result

(3) Program example

DEMO	LD	ST
Define variable	VAR IN :REAL:=0.5; OUT:REAL; END_VAR	
Program example		1 OUT:=ATAN(IN);
Operation result		

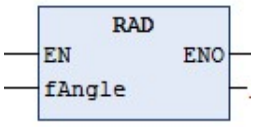
1-5-3-12. Floating point angle/radian conversion RAD/DEG

(1) Instruction overview

RAD: The operation of converting floating-point angles to radians. The calculation formula is [radian unit=angle unit ×л /180].

DEG: The operation of converting floating-point numbers from radians to angles. The calculation formula is [angle unit=radian unit ×л /180].

Floating point angle/radian conversion [RAD/DEG]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
RAD	Convert angles to radians		RAD(fAngle:=);
DEG	Convert radians to angles		DEG(fRadian:=);

(2) Related variables

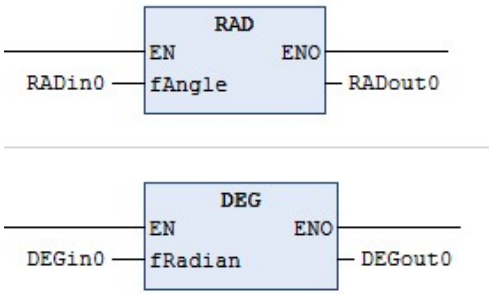
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
fAngle	Input angle	LREAL	As the data type	0	Input angle
	Output radian	LREAL	As the data type	0	Output radian

fRadian	Input radian	LREAL	As the data type	0	Input radian
	Output angle	LREAL	As the data type	0	Output angle

(3) Function

- RAD: The operation of converting floating-point angles to radians. The calculation formula is [radian unit=angle unit ×л /180].
- DEG: The operation of converting floating-point numbers from radians to angles. The calculation formula is [angle unit=radian unit ×л /180].

(4) Program example

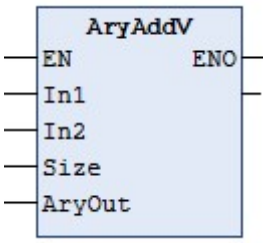
DEMO	LD	ST
Define variable	VAR RADout0: LREAL; RADin0: LREAL; DEGin0: LREAL; DEGout0: LREAL; END_VAR	
Program example		RADout0:=RAD(fAngle:= RADin0); DEGout0:=DEG(fRadian:= DEGin0);
Operation result		

1-5-3-13. Array element addition AryAddV

(1) Instruction overview

Add the same value to each element of the arrangement.

Array element addition [AryAddV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryAddV	Array element addition		AryAddV(In1:= , In2:= , Size:= , AryOut:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In1[]	Additive arrangement array	USINT/UINT/UDINT/ ULINT/SINT/INT/ DINT/LINT	As the data type	-	Additive arrangement array
In2	Additive value	USINT/UINT/UDINT/ ULINT/SINT/INT/ DINT/LINT	As the data type	-	Additive value
Size	The number of In1[] element to be added	UINT	As the data type	1	The number of In1[] element to be added
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
AryOut	Add result arrangement	USINT/UINT/UDINT/ ULINT/SINT/INT/ DINT/LINT	As the data type	-	Add result arrangement

(3) Function

- Add the additive value "In2" on the "Size" elements starting from In1[0] in the additive arrangement In1[], and output it to the additive result arrangement AryOut[].

(4) Note

- The value of "Size" should not exceed the array area of In1[] or AryOut[] to avoid PLC exceptions.
- The data types for "In[]", "In2", and "AryOut" should be consistent. If different, the value returned by the runtime function is FALSE, AryOut[] remains unchanged.
- If the sum result exceeds the valid range of AryOut[], the element of AryOut[] has an incorrect value. At this point, no abnormalities will occur. In addition, it will not damage the adjacent storage area of the element.

- When the value of "Size" is 0, the value of AryOut[] remains unchanged, and the function returns a value of True.
- In1, In2, AryOut[] supports both integer and real types, otherwise the runtime function returns FALSE.

(5) Program example

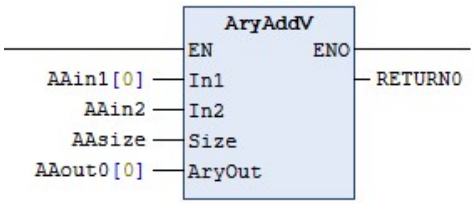
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR

    AAin1:ARRAY[0..5] OF INT;
    AAsize: UINT;
    AAin2: INT;
    AAout0:ARRAY[0..5] OF INT;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
AryAddV(In1:=AAin1[0] ,
    In2:= AAin2,
    Size:= AAsize ,
    AryOut:=AAout0[0] );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
AAin1	ARRAY [0..5] OF INT	
AAin1[0]	INT	11
AAin1[1]	INT	22
AAin1[2]	INT	33
AAin1[3]	INT	44
AAin1[4]	INT	55
AAin1[5]	INT	66
AAsize	UINT	4
AAin2	INT	12
AAout0	ARRAY [0..5] OF INT	
AAout0[0]	INT	23
AAout0[1]	INT	34
AAout0[2]	INT	45
AAout0[3]	INT	56
AAout0[4]	INT	0
AAout0[5]	INT	0
RETURN0	BOOL	TRUE

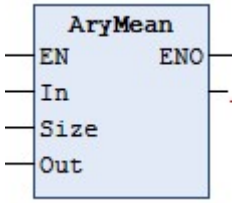
</td></tr>
</table>

1-5-3-14. Calculation of array elements average value AryMean

(1) Instruction overview

Calculate the average value of array elements.

Calculation of array elements average value [AryMean]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
AryMean	Calculation of array elements average value		AryMean(In:= , Size:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In[]	Arithmetic object array	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL/TIME	As the data type	-	Arithmetic object array
Size	In[] element numbers	UINT	As the data type	-	In[] element numbers
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Operation result	USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/ LINT/REAL/LREAL/TIME	As the data type	-	Operation result

(3) Function

- Calculate the average value of array elements.

(4) Note

- The four arithmetic operations are not supported when the value of In[] is +∞/-∞ and a non number.
- When the data types of In[] and "Out" are different, the function returns a value of FALSE while "Out" remains unchanged.
- The value of "Size" should not exceed the array area of In[] to avoid PLC exceptions.
- When "In" and "Out" are integers, the number after the decimal point of the average is rounded off.
- When the value of "Size" is 0, the value of "Out" is 0, and the function returns a value of True.
- In[], "Out" needs to use integers, real numbers, and TIME types, otherwise the runtime function returns a value of FALSE while "Out" remains unchanged.

(5) Program example

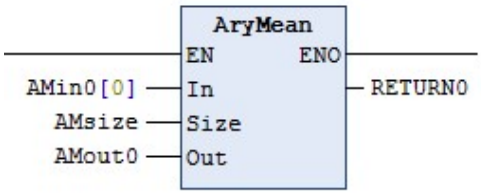
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR

    AMin0:ARRAY[0..5] OF INT;
    AMsize: UINT;
    AMout0: INT;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
AryMean(In:= AMin0[0],
    Size:= AMsize,
    Out:= AMout0);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
AMin0	ARRAY [0..5] OF INT	
AMin0[0]	INT	12
AMin0[1]	INT	22
AMin0[2]	INT	32
AMin0[3]	INT	42
AMin0[4]	INT	0
AMin0[5]	INT	0
AMsize	UINT	4
AMout0	INT	27
RETURN0	BOOL	TRUE

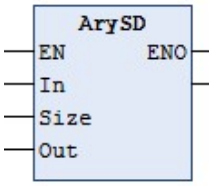
</td></tr>
</table>

1-5-3-15. Standard deviation of array elements ArySD

(1) Instruction overview

Calculate the standard deviation of the arranged elements.

Standard deviation of array elements [ArySD]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ArySD	Standard deviation of array elements		ArySD(In:= , Size:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In[]	Arithmetic object array	REAL/LREAL	As the data type	-	Arithmetic object array
Size	In[] element numbers	UINT	As the data type		In[] element numbers
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	standard deviation	REAL/LREAL	As the data type	-	standard deviation

(3) Function

- Calculate the standard deviation of the "Size" elements starting from In[0] in the operation object arrangement In[].

  i: The suffix of ln[] (0~"Size" -1);

  LnM: The average value of ln[0]~ln["Size" -1].

(4) Note

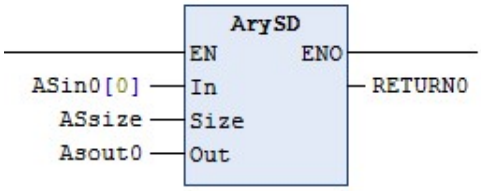
- When the value of "Size" is 0 or 1, the value of "Out" is 0, and the function returns a value of FALSE.
- When the intermediate value during the operation exceeds the valid range of In[], the value of "Out" is an illegal value. At this point, no abnormalities will occur.
- The value of "Size" should not exceed the array area of In[] to avoid PLC exceptions.
- In[] and Out data types should be consistent, otherwise the runtime function returns FALSE, Out remains unchanged.
- In[] supports real number types, otherwise the program compilation will report errors.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
    ASin0:ARRAY[0..5] OF REAL;
    ASsize: UINT;
    Asout0: REAL;
    RETURN0: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
ArySD(In:= ASin0[0],
    Size:= ASsize,
    Out:= ASout0[0] );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
ASin0	ARRAY [0..5] OF REAL	
ASin0[0]	REAL	5
ASin0[1]	REAL	6
ASin0[2]	REAL	7
ASin0[3]	REAL	8
ASin0[4]	REAL	9
ASin0[5]	REAL	10
ASsize	UINT	6
Asout0	REAL	1.87082875
RETURN0	BOOL	TRUE

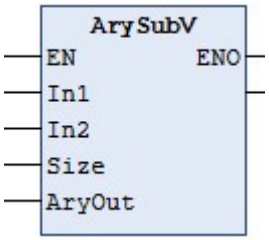
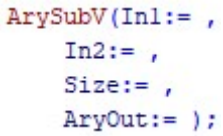
</td></tr>
</table>

1-5-3-16. Subtraction of array elements ArySubV

(1) Instruction overview

Subtract the same value from each element of the arrangement.

Subtraction of array elements [ArySubV]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ArySubV	Subtraction of array elements		


```
ArySubV(In1:= ,
    In2:= ,
    Size:= ,
    AryOut:= );
```

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In1[]	Subtracted arrangement	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	-	Subtracted arrangement
In2	Subtracted value	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	-	Subtracted value
Size	The number of elements to subtract from In1[]	UINT	As the data type	1	The number of elements to subtract from In1[]
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
AryOut	Subtraction result arrangement	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	-	Subtraction result arrangement

(3) Function

- Subtract "In2" from the "Size" elements starting from In1[0] in the subtracted group In1[] and output it to the subtraction result array AryOut[].

(4) Note

- The value of "Size" should not exceed the array area of In1[] or AryOut[] to avoid PLC exceptions.
- The data types for "In[]", "In2", and "AryOut" should be consistent. If different, the value returned by the runtime function is FALSE, AryOut[] remains unchanged.
- When the subtraction result exceeds the valid range of AryOut[], the element of AryOut[] is an unexpected value. At this point, there will be no abnormalities or damage to the adjacent storage area of the element.
- When the value of "Size" is 0, the value of AryOut[] remains unchanged, and the function returns a value of True.

- In1, In2, AryOut[] supports integer types as input, otherwise the program will compile an error.

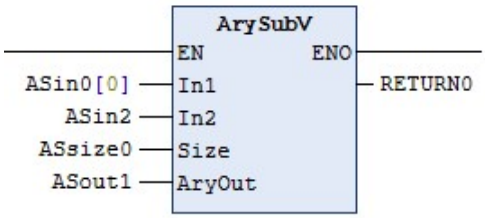
(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
    ASin1:ARRAY[0..5] OF INT;
    ASsize0: UINT;
    ASin2: INT;
    ASout1:ARRAY[0..5] OF INT;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
ArySubV(In1:= ASin1[0],
    In2:= ASin2[0],
    Size:=ASsize0 ,
    AryOut:= ASout1[0]);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
ASin1	ARRAY [0..5] OF INT	
ASin1[0]	INT	23
ASin1[1]	INT	34
ASin1[2]	INT	45
ASin1[3]	INT	56
ASin1[4]	INT	67
ASin1[5]	INT	78
ASsize0	UINT	6
ASin2	INT	12
ASout1	ARRAY [0..5] OF INT	
ASout1[0]	INT	11
ASout1[1]	INT	22
ASout1[2]	INT	33
ASout1[3]	INT	44
ASout1[4]	INT	55
ASout1[5]	INT	66
RETURN0	BOOL	TRUE

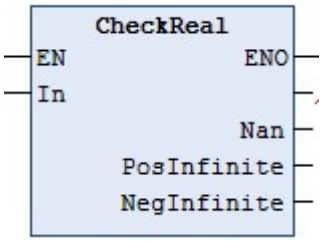
</td></tr>
</table>

1-5-3-17. Real number check CheckReal

(1) Instruction overview

Determine if the input value is non numeric, positive infinity, negative infinity.

Real number check [CheckReal]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
CheckReal	Real number check		CheckReal(In:= , Nan=> , PosInfinite=> , NegInfinite=>);

(2) related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Real number	REAL/LREAL	As the data type	-	Real number
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Nan	Non numerical judgment	BOOL	[FALSE,TRUE]	FALSE	True: Non numerical value FALSE: Not a non-numerical value
PosInfinite	Positive infinity judgment	BOOL	[FALSE,TRUE]	FALSE	True: Positive infinity FALSE: Not positive infinity
NegInfinite	Negative infinity judgment	BOOL	[FALSE,TRUE]	FALSE	True: Negative infinity FALSE: Not negative infinity

(3) Function

- Check if the real number "In" is non numerical, positive infinity, and negative infinity, and output the results to "Nan", "PosInfinite", and "NegInfinite" respectively.

(4) Note

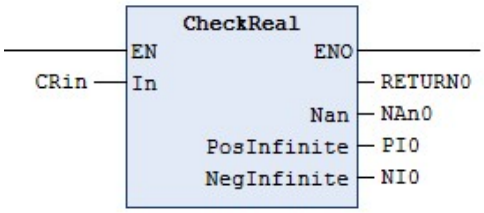
- When the parameter passed to "In" is of a data type other than real, an error occurs during compilation.
- When the judgment result is non numerical, positive infinity, or negative infinity, "Nan", "PosInfinite", and "NegInfinite" return True respectively, and the function returns True.
- When the judgment result is a normal value, the function returns a value of FALSE.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
    CRin: REAL;
    NAn0: BOOL;
    PI0: BOOL;
    NI0: BOOL;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Program example</td><td> 
</td><td>

```
    CRin:=CRin/1E-38;
CheckReal(In:= CRin,
    Nan=> NAn0,
    PosInfinite=>PI0 ,
    NegInfinite=> NI0);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
CRin	REAL	Infinity
NAn0	BOOL	FALSE
PI0	BOOL	TRUE
NI0	BOOL	FALSE

</td></tr>
</table>

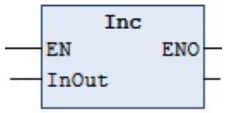
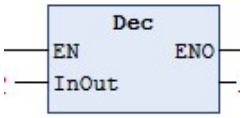
1-5-3-18. Increase/decrease Inc/Dec

(1) Instruction overview

Inc: Increase integer values.

Dec: Decrease integer value.

Increase/decrease [Inc/Dec]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
Inc	Increase		`Inc(InOut:= );`
Dec	Decrease		`Dec(InOut:= );`

(2) Related variables

VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Inout	Object data	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT	As the data type	-	Object data

(3) Function

- Inc: Increment the object data "InOut". Finally, when the maximum value of "InOut" is exceeded, it returns to the minimum value.
- Dec: Decrement the object data "InOut". Finally, when the minimum value of "InOut" is exceeded, it returns to the maximum value.

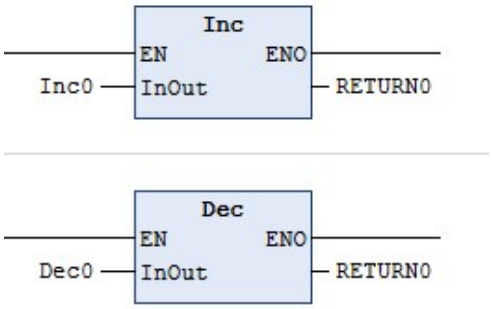
(4) Note

- When using this instruction in the ST program, the return value "Out" is not used.

(5) Program sample

DEMO	LD	ST

Define variable		`VAR`
		`  Inc0: INT:=100;`
		`  Dec0: INT:=100;`
		`  RETURN0: BOOL;`
		
		`END_VAR`

Program example		`Inc(InOut:= Inc0);` `Dec(InOut:= Dec0);`
Operation result		
Result analysis	Command conduction, the Inc instruction will continue to increase; The Dec command will continue to decrease.	

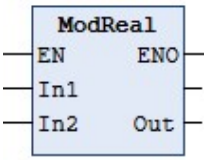
1-5-3-19. Real remainder ModReal

(1) Instruction overview

Calculate the remainder obtained by dividing the dividend "In1" by the divisor "In2".

Real remainder [ModReal]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST

ModReal	Real remainder		`ModReal(In1:= ,`
			`In2:= ,`
			`Out=> );`

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In1	Dividend	REAL	As the data type	(*)	Dividend
In2	Divisor	REAL	As the data type	(*)	Divisor
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Remainder	REAL	As the data type	(*)	Remainder

(3) Function

- Calculate the remainder obtained by dividing the dividend "In1" by the divisor "In2".
- The operation of this instruction is carried out according to the following expression.
- "Out"="In1" - ("In1"/"In2") * "In2", where the division operation in parentheses is the integer value rounded down.

- Examples of values for "In1", "In2", and "Out" are as follows:

“In1” value	“In2” value	“Out” value
8.8	3.2	2.4
8.8	-3.2	-0.8
-8.8	3.2	0.8
-8.8	-3.2	-2.4

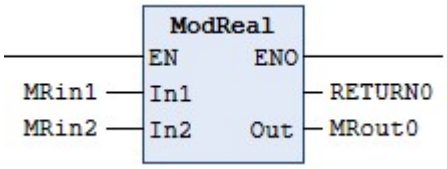
(4) Note

- According to the different combinations of values for "In1" and "In2", the values for "Out" are shown in the table below:

0		“In1”				
		Value	+∞	-∞	Non numerical value	
“In2”	0	Non numerical value	Non numerical value	Non numerical value	Non numerical value	Non numerical value
	Value	0	The remainder of dividing "In1" by "In2"	Non numerical value	Non numerical value	Non numerical value
	+∞	0	“In1” value	Non numerical value	Non numerical value	Non numerical value
	-∞	0	“In1” value	Non numerical value	Non numerical value	Non numerical value
	Non numerical value	Non numerical value	Non numerical value	Non numerical value	Non numerical value	Non numerical value

- When the parameters passed to "In1" and "In2" are data types other than real numbers, an exception will occur during compilation.

(5) Program sample

DEMO	LD	ST
Define variable	VAR MRin1: REAL; MRin2: REAL; MRout0: REAL; RETURN0: BOOL; END_VAR	
Program example		ModReal(In1:= MRin1, In2:= MRin2, Out=> MRout0);

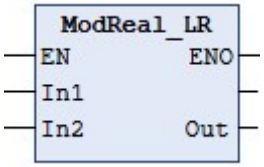
Operation result	

1-5-3-20. Long real remainder ModReal_LR

(1) Instruction overview

Calculate the remainder obtained by dividing the dividend "In1" by the divisor "In2".

Long real remainder [ModReal_LR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ModReal_LR	Long real remainder		ModReal_LR(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In1	Dividend	LREAL	As the data type	(*)	Dividend
In2	Divisor	LREAL	As the data type	(*)	Divisor
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Remainder	LREAL	As the data type	(*)	Remainder

(3) Function

- Calculate the remainder obtained by dividing the dividend "In1" by the divisor "In2".
- The operation of this instruction is carried out according to the following expression.
- "Out"="In1" - ("In1"/"In2") * "In2", where the division operation in parentheses is the integer value rounded down.
- Examples of values for "In1", "In2", and "Out" are as follows:

“In1” value	“In2” value	“Out” value
8.8	3.2	2.4
8.8	-3.2	-0.8
-8.8	3.2	0.8
-8.8	-3.2	-2.4

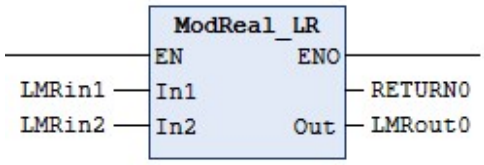
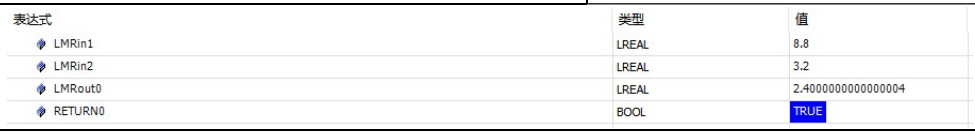
(4) Note

- According to the different combinations of values for "In1" and "In2", the values for "Out" are shown in the table below:

0		“In1”				
		Value	+∞	-∞	Non numerical value	
“In2”	0	Non numerical value	Non numerical value	Non numerical value	Non numerical value	Non numerical value
	Value	0	The remainder of dividing "In1" by "In2"	Non numerical value	Non numerical value	Non numerical value
	+∞	0	“In1” value	Non numerical value	Non numerical value	Non numerical value
	-∞	0	“In1” value	Non numerical value	Non numerical value	Non numerical value
	Non numerical value	Non numerical value	Non numerical value	Non numerical value	Non numerical value	Non numerical value

- When the parameters passed to "In1" and "In2" are data types other than real numbers, an exception will occur during compilation.

(5) Program example

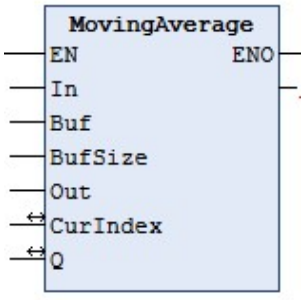
DEMO	LD	ST
Define variable		VAR LMRin1: LREAL; LMRin2: LREAL; LMRout0: LREAL; RETURN0: BOOL; END_VAR
Program example		ModReal_LR(In1:= LMRin1, In2:= LMRin2, Out=> LMRout0);
Operation result		

1-5-3-21. Moving average MovingAverage

(1) Instruction overview

Calculate the moving average.

Moving average [MovingAverage]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
MovingAverage	moving average		MovingAverage(In:= , Buf:= , BufSize:= , Out:= , CurIndex:= , Q:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Used to calculate the average value data	USINT/UINT/UDINT/ ULINT/SINT/INT/ DINT/LINT/REAL/ LREAL	As the data type	-	Used to calculate the average value data
Buf	Used to save the arrangement of "In"	USINT/UINT/UDINT/ ULINT/SINT/INT/ DINT/LINT/REAL/ LREAL	As the data type	1	Used to save the arrangement of "In"
BufSize	Used to calculate the max element number of average value	UINT	As the data type	-	Used to calculate the max element number of average value
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
CurIndex	Buf[] location for saving "In"	UINT	As the data type	0	Buf[] location for saving "In"
Q	True: The number of values saved to Buf[] is greater than "BufSize"; FALSE: The number of values saved to Buf[] is less than "BufSize"	BOOL	[FALSE,TRUE]	FALSE	True: The number of values saved to Buf[] is greater than "BufSize"; FALSE: The number of values saved to Buf[] is less than "BufSize"

VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Operation result	USINT/UINT/UDINT/ ULINT/SINT/INT/ DINT/LINT/REAL/ LREAL	As the data type	0	Operation result

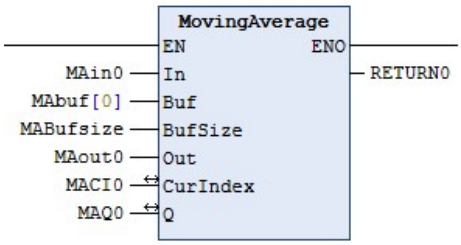
(3) Function

- Every time this command is executed, the input value "In" will be saved to the input value save arrangement Buf[]. The average value of the saved values is saved in the operation result "Out". The maximum number of element used to calculate the average is specified in "BufSize".

(4) Note

- The data types of "In", "Out", and Buf[] elements should be consistent, otherwise the function return value will be FALSE, Out[] remains unchanged.
- Even if the operation result exceeds the valid range of "Out", no exception will occur, and illegal values will be saved in "Out".
- When the value of "BufSize" is 0, the values of "Out" and "CurIndex" are 0, and the value of "Q" becomes True.
- After changing the value of "BufSize", "CurIndex" will maintain its current value.
- The value of "BufSize" should not exceed the size of Buf[] to avoid PLC anomalies.

(5) Program example

DEMO	LD	ST
Define variable	VAR MAbuf:ARRAY[0..5] OF INT; MAin0: INT; MABufsize: UINT; MAout0: INT; MACI0: UINT; MAQ0: BOOL; RETURN0: BOOL; END_VAR	
Program example		MovingAverage(In:= MAin0, Buf:= MAbuf[0], BufSize:= MABufsize, Out:= MAout0, CurIndex:= MACI0, Q:= MAQ0);

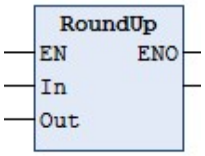
Operation result	
Result analysis	Every time the instruction is executed, the data of MAin0 will be saved to the starting array of MAbuf[0]; Execute MABufsize a total of times and save MABufsize data in MAbuf.

1-5-3-22. Real carry RoundUp

(1) Instruction overview

Carry the first place of decimal part of a real number and take the whole number.

Real carry [RoundUp]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
RoundUp	Real carry		RoundUp(In:= , Out:=);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Convert object	REAL/LREAL	As the data type	-	Convert object
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Convert result	DINT/LINT	As the data type	-	Convert result

(3) Function

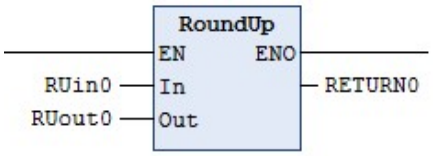
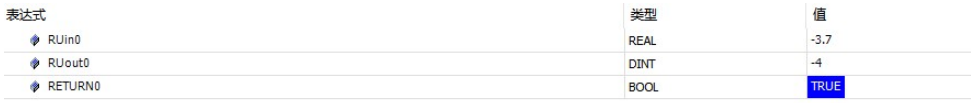
- Process the first decimal place of the real number "In" to make it an integer.

Input value	Output value	Input value	Output value
REAL#1.6	DINT#2	REAL#-1.4	DINT#-2
REAL#1.5	DINT#2	REAL#-2.5	DINT#-3
REAL#1.4	DINT#2	REAL#3.2	DINT#4
REAL#2.5	DINT#3	REAL#3.7	DINT#4
REAL#-1.6	DINT#-2	REAL#-3.2	DINT#-4
REAL#-1.5	DINT#-2	REAL#-3.7	DINT#-4

(4) Note

- When the conversion result exceeds the valid range of "Out", the value of "Out" is an incorrect value.
- The data types of "In" and "Out" should match, for example, REAL corresponds to DINT data, otherwise the function returns a value of FALSE and "Out" remains unchanged.
- "Out" requires the use of Dint and Lint types, otherwise the runtime function returns FALSE and "Out" remains unchanged.

(5) Program example

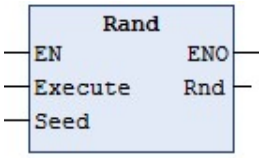
DEMO	LD	ST
Define variable		VAR RUin0: REAL; RUout0: DINT; END_VAR
Program example		RoundUp(In:= RUin0, Out:=RUout0);
Operation result		

1-5-3-23. Generate random number RAND

(1) Instruction overview

Generate pseudo-random numbers.

Generate random number [RAND]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
RAND	Generate random number		RAND_0(Execute:= , Seed:= , Rnd=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
Execute	Trigger	BOOL	[FALSE,TRUE]	FALSE	Trigger generation of random

					numbers
Seed	Random number mode	UINT	As the data type	0	Specify a sequence of random numbers 0: Not specified
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Rnd	Random number	LREAL	0.00000000000000e+0～1.00000000000000e+0	-	Random number

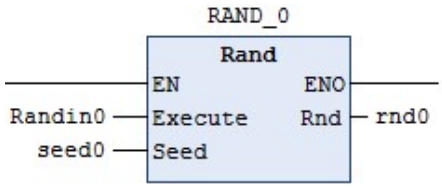
(3) Function

- Generate a random number "Rnd". The value of "Rnd" changes after each instruction execution.
- The random number mode "Seed" is used to specify a sequence of random numbers. If the value of "Seed" is the same, the same random number sequence will be generated every time the power is turned on. Therefore, reproducible random numbers can be generated.
- When the value of "Seed" is 0, a non repetitive random number will be generated. If you do not want to generate the same random number sequence every time the power is turned on, please set the value of "Seed" to 0.

(4) Note

- The value of "Rnd" is a real number between 0 and 1.

(5) Program example

DEMO	LD	ST
Define variable	VAR RAND_0: Rand; Randin0: BOOL; seed0: UINT; rnd0: LREAL; END_VAR	
Program example		RAND_0(Execute:= Randin0, Seed:= seed0, Rnd=> rnd0);
Operation result		

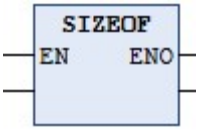
1-5-4. Address operation instruction

1-5-4-1. Data type size SIZEOF

(1) Instruction overview

Function: Execute this function to determine the number of bytes required for the given data type. Simply put, its function is to return the number of memory bytes occupied by an object or type.

Data type size [SIZEOF]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
SIZEOF	Data type size		SIZEOF

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Target data	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/LINT/ REAL/LREAL/TIME/DATE/TOD/ DT/STRING	As the data type	-	Target data
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Data length	USINT/UINT/UDINT/ULINT	As the data type	-	Data length

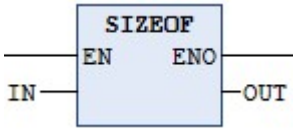
(3) Function

The return value of SIZEOF is an unsigned value, and the return value of type will be used to find the size of variable IN, the output value of OUT is in bytes, IN can be of any data type.

Return data type of SIZEOF

SIZEOF return value	Implicit data type
0 <= size of x < 256	USINT
256 <= size of x < 65536	UINT
65536 <= size of x < 4294967296	UDINT
4294967296 <= size of x	ULINT

(4) Program example

DEMO	LD	ST
Define variable	VAR IN :ARRAY[0..4] OF INT; OUT:UINT; END_VAR	
Program example		1 OUT:=SIZEOF(IN);
Operation result		
Result analysis	The program assigns the result to OUT, and the final OUT equals 10, because the IN array consists of 5 INT integer elements, the result unit of SIZEOF is BYTE, so the final program running result is a total of 10 BYTEs, indicating that IN occupies 10 bytes of memory.	

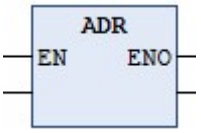
1-5-4-2. Address operator ADR

(1) Instruction overview

ADR: Obtain the memory address of the input variable, assign the result to the output variable, operator, which is an extension of the IEC 61131-3 standard.

^: Obtain the address content of the input variable and assign the result to the output variable.

Address operator [ADR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
ADR	Obtain address instruction		ADR
^	Obtain address content instruction	-	^

(2) Related variables

ADR

VAR_INOUT	Name	Data type	Valid range	Initial value	Description
IN	Target data	BOOL/BYTE/WORD/DWORD/ LWORD/USINT/UINT/UDINT/ ULINT/SINT/INT/DINT/LINT/ REAL/LREAL/TIME/DATE/ TOD/DT/STRING	As the data type	-	Target data
OUT	Target data address	POINTER TO	-	-	Target data address

^

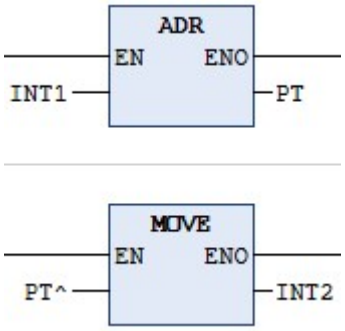
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
OUT	Point to address content	BOOL/BYTE/WORD/DWORD/LWORD/USINT/UINT/UDINT/ULINT/SINT/INT/DINT/LINT/REAL/LREAL/TIME/DATE/TOD/DT/STRING	As the data type	-	Point to address content

(3) Function

The return value of ADR is only the memory address of the variable. The length of data that can be stored in this memory address is 1 BYTE. The content in the corresponding address can be extracted using the content operator "^". To obtain the memory address of var_int1, assign its address to a pointer variable, and then extract the specific content in its corresponding address using the "^" operator and assign it to var_int2. The implementation program is as follows:

```
pt := ADR(var_int1);
var_int2:= pt^;
```

(4) Program example

DEMO	LD	ST
Define variable	VAR INT1:INT:=10; PT :POINTER TO INT; INT2:INT; END_VAR	
Program example		1 PT:=ADR(INT1); 2 INT2:=PT^;

Operation result	

1-5-4-3. Bit address operator BITADR

(1) Instruction overview

Function: Returns the bit address information offset of the allocation variable.

Bit address operator [BITADR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	-		

Instruction	Name	Graph	ST
BITADR	Bit address operator		BITADR

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
IN	Target data	BOOL	FALSE,TRUE	-	Target data
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
OUT	Point to the target address	DWORD	As the data type	-	Point to the target address

(3) Function

BITADR returns a bit offset numerical address as a DWORD variable type. Note that the offset value depends on whether the option type address can be obtained from the target system. The maximum value of DWORD defines the memory area as shown in the table below:

BITADR offset addresses for each address area

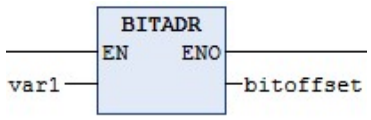
Address area	Start address	Explanation
Memory	16x40000000	%M
Input	16x80000000	%I
Output	16xC0000000	%Q

(4) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR
    var1 AT %IX2.3:BOOL;
    bitoffset     : DWORD;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
1   bitoffset:=BITADR(var1);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
var1	BIT	FALSE
bitoffset	DWORD	16#80000013

</td></tr>
<tr><td>Result analysis</td><td colspan="2">

The running result is 80000013 in hexadecimal, where "2" in %IX2.3 is 2 bytes and ". 3" is the 4th bit, so its address is equal to 2*8+4=20. Convert decimal 20 to hexadecimal 14. Also, because the first address corresponding to Zone I is stored starting from 80000000, it is not difficult to understand that the actual address corresponding to hexadecimal 14 is 16#80000013. The schematic diagram is shown in the table below:

BITADR example

Offset value	Data content
16#80000000	
16#80000001	
16#80000002	
...	
...	
16#80000011	
16#80000012	
16#80000013	bitoffset

</td></tr>
</table>

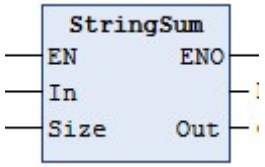
1-6. String command

1-6-1. Sum calculation StringSum

(1) Instruction overview

Calculate the SUM value of a string.

Sum calculation [StringSum]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
StringSum	Sum calculation		StringSum(In:= , Size:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Object string	STRING	As the data type	-	Object string
Size	Byte size	USINT	1,2	1	Byte size of Sum value
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Sum value	STRING	The number of bytes represented in 'size'	-	The string after Sum value conversion

(3) Function

- Calculate the SUM value of the object string "In" (the sum of character encodings for each character). The SUM value "Out" is the number of bytes specified in the byte size "Size". "Out" is represented as a string of hexadecimal numbers, with a NULL character saved at the end.

(4) Note

- When the total character encoding of "In" exceeds the number of bits that "Size" can represent, the high bits will be rounded off, and the function will return a value of FALSE.
- When the value of "Size" exceeds the valid range, the program encounters an exception and the function returns a value of FALSE.
- When the byte count of "In" is 0 (NULL), the program encounters an exception and the function returns a value of FALSE.
- When "In" does not end with a NULL, it may result in abnormal results.

(5) Program example

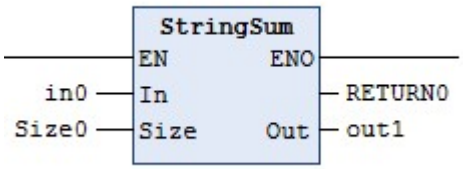
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR

    Size0: USINT;
    in0:STRING;
    out1: STRING(2);
    RETURN0: BOOL;


END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
StringSum(In:=in0 ,
    Size:=size0,
    Out=>out1);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
Size0	USINT	2
in0	STRING	'1234'
out1	STRING(2)	'CA'
RETURN0	BOOL	TRUE

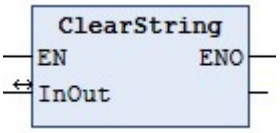
</td></tr>
<tr><td>Result analysis</td><td colspan="2">The character data in In is converted to hexadecimal, and the low bits are added, the high bits are added, to obtain the calculated value. If the low bit sum exceeds 16#F, it adds the high bit by 1. If the high bit sum exceeds 16#F, the data is not displayed; for example '1234'=31 32 33 34 =3+3+3+3 1+2+3+4=CA.</td></tr>
</table>

1-6-2. String clear Clearstring

(1) Instruction overview

Clear the string.

String clear [Clearstring]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
Clearsteing	String clear		`ClearString(InOut:= );`

(2) Related variables

VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
InOut	Clear the string	STRING	As the data type	-	The string needs to be cleared

(3) Function

- Clear the string "InOut". Save NULL characters in all regions of "InOut".

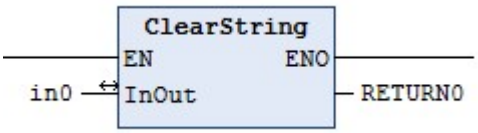
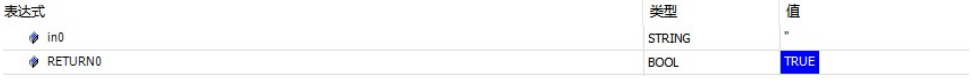
(4) Note

- The return value remains true.

(5) Program example

DEMO	LD	ST

Define variable		`VAR`
		`in0:STRING;`
		`RETURN0: BOOL;`
		`END_VAR`

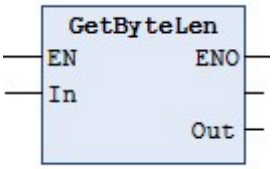
Program example		`RETURN0:=ClearString(` `InOut:=string0);`
Operation result		
Result analysis	The value of In0 is '1234', and the string data is cleared after writing.	

1-6-3. Get the number of bytes in the string GetByteLen

(1) Instruction overview

Calculate the number of bytes in a string.

Get the number of bytes in the string [GetByteLen]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
GetByteLen	Get the number of bytes in the string		GetByteLen(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Object string	STRING	As the data type	-	The object string to calculate the number of bytes
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	String byte numbers	UINT	0~1985	-	Byte numbers

(3) Function

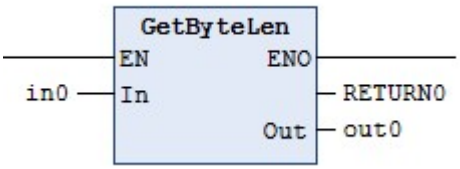
- Calculate the number of bytes from the beginning to the end of the object string "In". The NULL character at the end of the string is not within the number of bytes.

(4) Note

- The length of the input string is limited by the data type STRING.

(5) Program example

DEMO	LD	ST
Define variable		VAR in0:STRING; out0: UINT; END_VAR

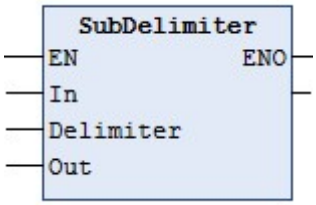
Program example		GetByteLen(In:=in0, Out=>out0);
Operation result		

1-6-4. Real string read delimiter delete SubDelimiter

(1) Instruction overview

Read data separated by delimiters from a string and save it as a real number array.

Real string read delimiter delete [SubDelimiter]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
SubDelimiter	Real string read delimiter delete		SubDelimiter(In:= , Delimiter:= , Out:=);

(2) Related variables

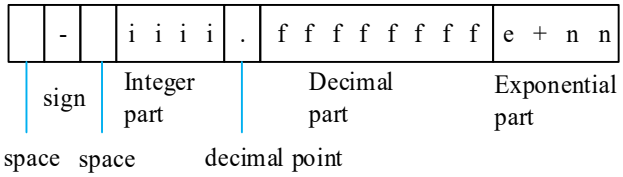
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Input string	STRING	Maximum 1985 bytes	-	String to be read separated by delimiters
Delimiter	Delimiter	_eDELIMITER	As the data type	_eDELIMITER_COMMA	Delimiter
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Save position real number array	REAL	As the data type	-	Save long real arrays of converted data

(3) Function

- Convert the string separated by the delimiter "Delimiter" in the input string "In" to a storage position real number array "Out", and save it sequentially as the values of each array element.
- The data type of the delimiter "Delimiter" is the enumeration type _eDELIMITER, and the meaning of the enumeration value is as follows:

Enum value	Meaning
_COMMA	‘,’ (comma)
_TAB	‘$T’ (Tab)
_SEMICOLON	‘;’ (semicolon)
_SPACE	‘ ’ (space)

  - When the number of data separated by delimiters in "In" is greater than the number of members in "Out", excess data will be ignored.
  - When the number of data separated by delimiters in "In" is less than the number of members in "Out", the value of members with insufficient data remains unchanged.
  - The data separated by delimiters in "In" is of string type. The string format for representing real type data is as follows:



Name	Format
Sign	After ignoring the consecutive ' ' (spaces) at the beginning of the string, consider the individual occurrences of '+' or '-' as symbols. The '+' symbol can be omitted. The consecutive spaces after the symbol will be ignored.
Integer part	The integer part is from the end of the symbol to the front of the decimal point. However, the consecutive spaces after the symbol are not within the integer part. In addition, symbols may be omitted. When there is no decimal point or decimal part, consider the front of the exponential part as the integer part. When there is no decimal point, decimal part, or exponential part, treat the entire string as an integer part. The integer part is composed of numbers ranging from 0 to 9. The integer part cannot be omitted. The maximum number of digits in the integer part refers to the maximum string length of 1985 minus the number of digits in the symbol, decimal point, decimal part, exponential part, and spaces before and after the symbol.
Decimal point	A separate '.' (period) followed by the integer part is a decimal point. When there is no decimal part, please omit the decimal point.
Decimal part	The decimal part starts from after the decimal point and ends before the exponential part. When there is no exponential part, the entire string after the decimal point is the decimal part. The decimal part is composed of numbers ranging from 0 to 9. The decimal part can be omitted. The maximum number of decimal places is 15. When there is no decimal point, the decimal part also does not exist.
Exponential part	Starting from the individual 'e' followed by the decimal part and the individual '+' or '-' after it, until the end of the string, it is considered as the exponential part. When there is no decimal part, consider the above string after the decimal point as the exponential part. When there is no decimal point or decimal part, consider the above string after the integer part as the exponential part. The numerical part of the exponential consists of numbers ranging from 0 to 9. The exponential part can be omitted. The maximum number of digits in the numerical part of the exponential is 3.

An example of conversion rules is as follows:

Format	STRING	REAL
Signed, with decimal point, with decimal part, without exponential part	‘-123.4567’	-123.4567

Signed, with decimal point, with decimal part, with exponential part	‘+123.4567e+02’	12345.67
Without sign, with decimal point, with decimal part, with exponential part	‘123.4567e-02’	1.234567
Without sign, without decimal point, without decimal part, without exponential part	‘1’	1

Conversion precautions:

- When the numerical value represented by the input string exceeds the valid range of REAL data (1.0E-44~3.402823E+38), if it is a positive number, the conversion result is Infinity; If it is a negative number, the conversion result is -Infinity.
- The input string cannot contain any other characters that are not real strings, otherwise the conversion will encounter an exception and the function will return a value of FALSE.
- When there are two or more consecutive "_" (underline) at any middle position of the input string, the conversion is normal.
- When there is an '_' (underline) between the '-' (negative sign) or '+' (positive sign) at the beginning of the input string and a number, the conversion is normal.
- When there is an '_' (underline) at the beginning or end of the input string, the conversion exception occurs (the function returns FALSE).
- When there is a separate '_' (underline) in any middle position of the input string, it will be ignored.
- When the content of the input string exceeds the precision that can be represented by the REAL data type, it will be rounded.
- When the content of the input string is closer to 0 than the value that can be represented by the data type of REAL, the conversion result is 0.
- When the decimal part of the input string exceeds the maximum of 15 digits, the program will not execute and cannot be converted. The function returns FALSE.
- When the exponential part of the input string exceeds the maximum number of 3 digits, the program will not execute and cannot be converted. The function returns FALSE.
- When the input string contains integers, decimals, and exponents with numbers other than 0~... 9, the program does not execute and cannot be converted. The function returns FALSE.
- The input string has no decimal part, but the decimal point is not omitted. The program does not execute and cannot be converted. The function returns FALSE.
- The exponential part of the input string has no numerical part, but there are symbols "e+"/"e -". The program does not execute and cannot be converted. The function returns FALSE.
- If other input strings are any non regular numbers, the program will not execute and cannot be converted. The function returns FALSE.
- If there is a space character before and after the exponential part of the input string, the program will not execute and cannot be converted. The function returns FALSE.
- The continuous space characters at the beginning and end of the input string (including before and after the delimiter) will be ignored, and the instruction will execute normally.
- When the exponential part is a decimal, the program does not execute and cannot be converted. The function returns FALSE.

(4) Note

- When connecting with delimiters in "In", there is no data separated by delimiters. If the data does not exist, the value of the corresponding member in Out[] is REAL#0.

- When the string in "In" contains data that does not conform to the real data format, illegal data and subsequent data are not converted. The function returns a value of FALSE, and the successfully converted values remain unchanged.
- Do not use separators for purposes other than separation in "In". Even if used for purposes other than separation, it will still be recognized as a delimiter in this instruction.
- When the conversion result exceeds the array area of Out[], it is necessary to avoid PLC exceptions, and data that does not exceed the array area of Out[] will be saved.
- When the input string is empty, the function returns True and Out remains unchanged.

(5) Program example

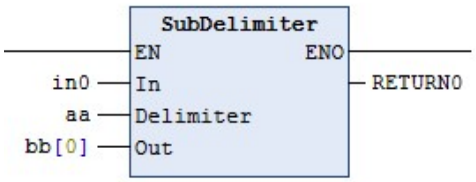
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR

    in0:STRING;
    aa: XJ_DataUtils._eDELIMITER;
    bb:ARRAY[0..10] OF REAL;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
SubDelimiter(In:=in0,
    Delimiter:=aa,
    Out:=bb[0]);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
in0	STRING	'123.45,2e+02,123.456e-02'
aa	_EDELIMITER	_COMMA
bb	ARRAY [0..10] OF R...	
bb[0]	REAL	123.45
bb[1]	REAL	200
bb[2]	REAL	1.23456
bb[3]	REAL	0
bb[4]	REAL	0
bb[5]	REAL	0
bb[6]	REAL	0
bb[7]	REAL	0
bb[8]	REAL	0
bb[9]	REAL	0
bb[10]	REAL	0
RETURN0	BOOL	TRUE

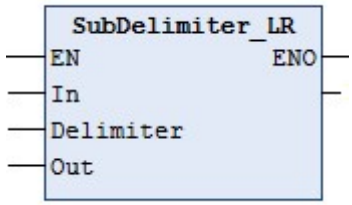
</td></tr>
</table>

1-6-5. Long real string read delimiter delete SubDelimiter_LR

(1) Instruction overview

Read data separated by delimiters from a string and save it as a long real number array.

Long real string read delimiter delete [SubDelimiter_LR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
SubDelimiter_LR	Long real string read delimiter delete		SubDelimiter_LR(In:= , Delimiter:= , Out:=);

(2) Related variables

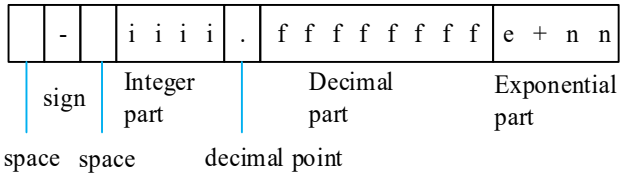
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Input string	STRING	Maximum 1985 bytes	-	String to be read separated by delimiters
Delimiter	Delimiter	_eDELIMITER	As the data type	_eDELIMITER._COMMA	Delimiter
VAR_IN_OUT	Name	Data type	Valid range	Initial value	Description
Out	Save position real number array	LREAL	As the data type	-	Save long real arrays of converted data

(3) Function

- Convert the string separated by the delimiter "Delimiter" in the input string "In" to a storage position real number array "Out", and save it sequentially as the values of each array element.
- The data type of the delimiter "Delimiter" is the enumeration type _eDELIMITER, and the meaning of the enumeration value is as follows:

Enum value	Meaning
_COMMA	‘,’ (comma)
_TAB	‘$T’ (Tab)
_SEMICOLON	‘;’ (semicolon)
_SPACE	‘ ’ (space)

  - ◆ When the number of data separated by delimiters in "In" is greater than the number of members in "Out", excess data will be ignored.
  - ◆ When the number of data separated by delimiters in "In" is less than the number of members in "Out", the value of members with insufficient data remains unchanged.
  - ◆ The data separated by delimiters in "In" is of string type. The string format for representing real type data is as follows:



Name	Format
Sign	After ignoring the consecutive ' ' (spaces) at the beginning of the string, consider the individual occurrences of '+' or '-' as symbols. The '+' symbol can be omitted. The consecutive spaces after the symbol will be ignored.
Integer part	The integer part is from the end of the symbol to the front of the decimal point. However, the consecutive spaces after the symbol are not within the integer part. In addition, symbols may be omitted. When there is no decimal point or decimal part, consider the front of the exponential part as the integer part. When there is no decimal point, decimal part, or exponential part, treat the entire string as an integer part. The integer part is composed of numbers ranging from 0 to 9. The integer part cannot be omitted. The maximum number of digits in the integer part refers to the maximum string length of 1985 minus the number of digits in the symbol, decimal point, decimal part, exponential part, and spaces before and after the symbol.
Decimal point	A separate '.' (period) followed by the integer part is a decimal point. When there is no decimal part, please omit the decimal point.
Decimal part	The decimal part starts from after the decimal point and ends before the exponential part. When there is no exponential part, the entire string after the decimal point is the decimal part. The decimal part is composed of numbers ranging from 0 to 9. The decimal part can be omitted. The maximum number of decimal places is 15. When there is no decimal point, the decimal part also does not exist.
Exponential part	Starting from the individual 'e' followed by the decimal part and the individual '+' or '-' after it, until the end of the string, it is considered as the exponential part. When there is no decimal part, consider the above string after the decimal point as the exponential part. When there is no decimal point or decimal part, consider the above string after the integer part as the exponential part. The numerical part of the exponential consists of numbers ranging from 0 to 9. The exponential part can be omitted. The maximum number of digits in the numerical part of the exponential is 3.

An example of conversion rules is as follows:

Format	STRING	REAL
Signed, with decimal point, with decimal part, without exponential part	‘-123.4567’	-123.4567

Signed, with decimal point, with decimal part, with exponential part	‘+123.4567e+02’	12345.67
Without sign, with decimal point, with decimal part, with exponential part	‘123.4567e-02’	1.234567
Without sign, without decimal point, without decimal part, without exponential part	‘1’	1

Conversion precautions:

- When the numerical value represented by the input string exceeds the valid range of LREAL data (4.94065645841247E-324~1.7976931348623157E+308), if it is a positive number, the conversion result is Infinity; If it is a negative number, the conversion result is -Infinity.
- When the input string does not represent a long real number value, the conversion result is 0, and the function return value is FALSE.
- When there are two or more consecutive "_" (underline) at any middle position of the input string, the conversion is normal.
- When there is an '_' (underline) between the '-' (negative sign) or '+' (positive sign) at the beginning of the input string and a number, the conversion is normal.
- When there is an '_' (underline) at the beginning or end of the input string, the conversion exception occurs (the function returns FALSE).
- When there is a separate '_' (underline) in any middle position of the input string, it will be ignored.
- When the content of the input string exceeds the precision that can be represented by the LREAL data type, it will be rounded.
- When the content of the input string is closer to 0 than the value that can be represented by the LREAL data type, the conversion result is 0.
- When the decimal part of the input string exceeds the maximum of 15 digits, the program will not execute and cannot be converted. The function returns FALSE.
- When the exponential part of the input string exceeds the maximum number of 3 digits, the program will not execute and cannot be converted. The function returns FALSE.
- When the input string contains integers, decimals, and exponents that are not between 0 and 9, the program does not execute and cannot be converted. The function returns FALSE.
- The input string has no decimal part, but the decimal point is not omitted. The program does not execute and cannot be converted. The function returns FALSE.
- The exponential part of the input string has no numerical part, but there are symbols 'e+'/'e-'. The program does not execute and cannot be converted. The function returns FALSE.
- If other input strings are any non regular numbers, the program will not execute and cannot be converted. The function returns FALSE.
- If there is a space character before and after the ‘+/-’ of exponential part of the input string, the program will not execute and cannot be converted. The function returns FALSE.
- The continuous space characters at the beginning and end of the input string (including before and after the delimiter) will be ignored, and the instruction will execute normally.
- When the exponential part is a decimal, the program does not execute and cannot be converted. The function returns FALSE.

(4) Note

- "In" contains data separated by delimiters. If the data does not exist, the value of the corresponding member in Out[] is LREAL#0.

- When the string in "In" contains data that does not conform to the real data format, illegal data and subsequent data are not converted. The function returns a value of FALSE, and the successfully converted values remain unchanged.
- Do not use separators for purposes other than separation in "In". Even if used for purposes other than separation, it will still be recognized as a delimiter in this instruction.
- When the conversion result exceeds the array area of Out[], it is necessary to avoid PLC exceptions, and data that does not exceed the array area of Out [] will be saved.
- When the input string is empty, the function returns a value of True, Out has no change.

(5) Program example

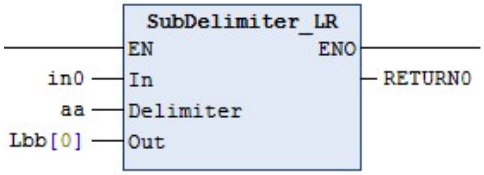
<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR

    in0:STRING;
    aa: XJ_DataUtils._eDELIMITER;
    Lbb:ARRAY[0..10] OF LREAL;
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
SubDelimiter_LR(In:=in0,
    Delimiter:=aa ,
    Out:=lbb[0] );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
in0	STRING	'123.45,2e+02,123.456e-02'
aa	_EDELIMITER	_COMMA
Lbb	ARRAY [0..10] OF L...	
Lbb[0]	LREAL	123.45
Lbb[1]	LREAL	200
Lbb[2]	LREAL	1.23456
Lbb[3]	LREAL	0
Lbb[4]	LREAL	0
Lbb[5]	LREAL	0
Lbb[6]	LREAL	0
Lbb[7]	LREAL	0
Lbb[8]	LREAL	0
Lbb[9]	LREAL	0
Lbb[10]	LREAL	0
RETURN0	BOOL	TRUE

</td></tr>
</table>

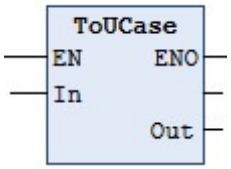
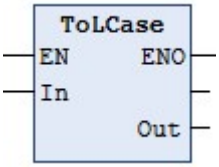
1-6-6. String letter conversion ToUCase/ToLCase

(1) Instruction overview

ToUCase: Converts all half width letters in a string to uppercase letters.

ToLCase: Convert all half width letters in a string to lowercase letters.

String letter conversion [ToUCase/ToLCase]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
ToUCase	String uppercase conversion		ToUCase(In:= , Out=>);
ToLCase	String lowercase conversion		ToLCase(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Conversion object	STRING	Maximum 1985 bytes	-	Converted object string
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Conversion result	STRING	Maximum 1985 bytes	-	Converted string

(3) Function

- ToUCase: Convert the half width letters of the converted object string "In" to uppercase letters.
- ToLCase: Convert the half width letters of the conversion object string "In" to lowercase letters.
- All instructions contain a NULL character at the end of the string when outputting. Characters with non half width letters are not affected.

(4) Note

- Full width letters do not belong to the conversion object.
- When the text code of "In" is abnormal, an exception may occur, and the "Out" value is inaccurate.
- When "In" does not end with a NULL character, it may cause abnormal conversion results.
- When the conversion result exceeds the size of "Out", only the conversion value of Out length will be displayed.

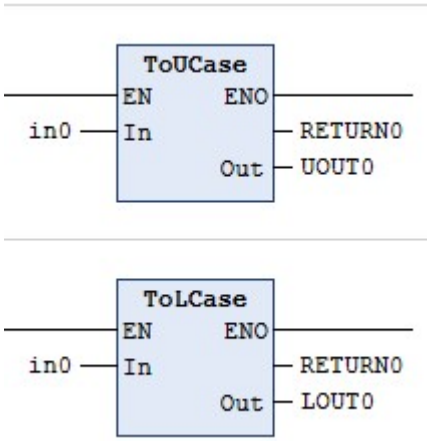
(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variable</td><td colspan="2">

```
VAR

    in0:STRING;
    UOUT0: STRING(1985);
    LOUT0: STRING(1985);
    RETURN0: BOOL;

END_VAR
```

</td></tr>
<tr><td>Program example</td><td>


</td><td>

```
ToLCase(In:=in0 ,
    Out=>Lout0);

ToUCase(In:=in0 ,
    Out=>Uout0);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

</td></tr>
</table>

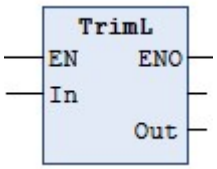
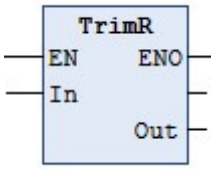
1-6-7. String left/right adjustment TrimL/TrimR

(1) Instruction overview

TrimL: Remove spaces at the beginning of the string.

TrimR: Remove spaces at the end of the string.

String left/right adjustment [TrimL/TrimR]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_DataUtils		

Instruction	Name	Graph	ST
TrimL	Left adjustment of string		TrimL(In:= , Out=>);
TrimR	Right adjustment of string		TrimR(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
In	Delete object	STRING	Maximum 1985 bytes	-	Object string deleted
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Out	Delete result	STRING	Maximum 1985 bytes	-	String after deletion

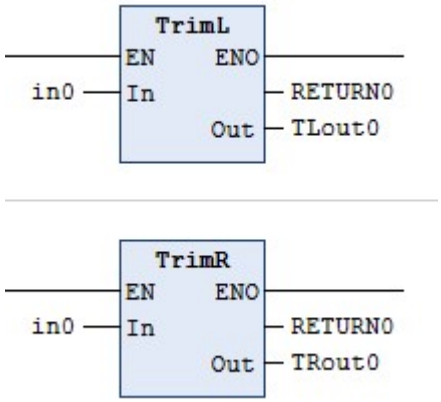
(3) Function

- TrimL: Remove spaces from the beginning of the object string "In" to the first character that appears. When there are no spaces at the beginning, no processing is done.
- TrimR: Delete spaces from the last character of the object string "In" to the end. When there are no spaces at the end, no processing is done.
- Both instructions will output results including the NULL character at the end of the string.

(4) Note

- Both instructions will output results including the NULL character at the end of the string.

(5) Program example

DEMO	LD	ST
Define variable	VAR in0:STRING; TLout0: STRING(1985); TRout0: STRING(1985); END_VAR	
Program example	 	TrimL(In:=string0 , Out=>outl[2]); TrimR(In:=string0 , Out=>outl[3]);
Operation result		

表达式	类型	值
in0	STRING	' ABC '
TLout0	STRING(1985)	'ABC '
TRout0	STRING(1985)	' ABC'

2. Special instructions

2-1. High speed count instructions

2-1-1. Function overview

By selecting different counters, high-speed input signals such as measurement sensors and rotary encoders can be measured, with a maximum measurement frequency of 1MHz. The high-speed counting instruction libraries for XS3-26T4, XSDH-60A32-E, and XSLH-30A32 can be found in chapters 2-2.

2-1-2. Function block

The reason why the graphical representation and ST representation of some instructions are marked with red wavy lines is that VAR_IN_OUT'Counter must be assigned a value when called in order to compile and download the program.

2-1-2-1. Enable the high speed counter【XJ_Counter_Enable】

High speed counter enable [XJ_Counter_Enable]			
Execution condition	Normally ON/OFF triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_Counter_Enable	Enable the high speed counter	XJ_Counter_Enable_0 XJ_Counter_Enable EN → ENO Counter → diValue xEnable → udiFrequency eMode → udiRPM xDirection → xBusy xError eErrorID	XJ_Counter_Enable_0(Counter:= , xEnable:= , eMode:= , xDirection:= , diValue=> , udiFrequency=> , udiRPM=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

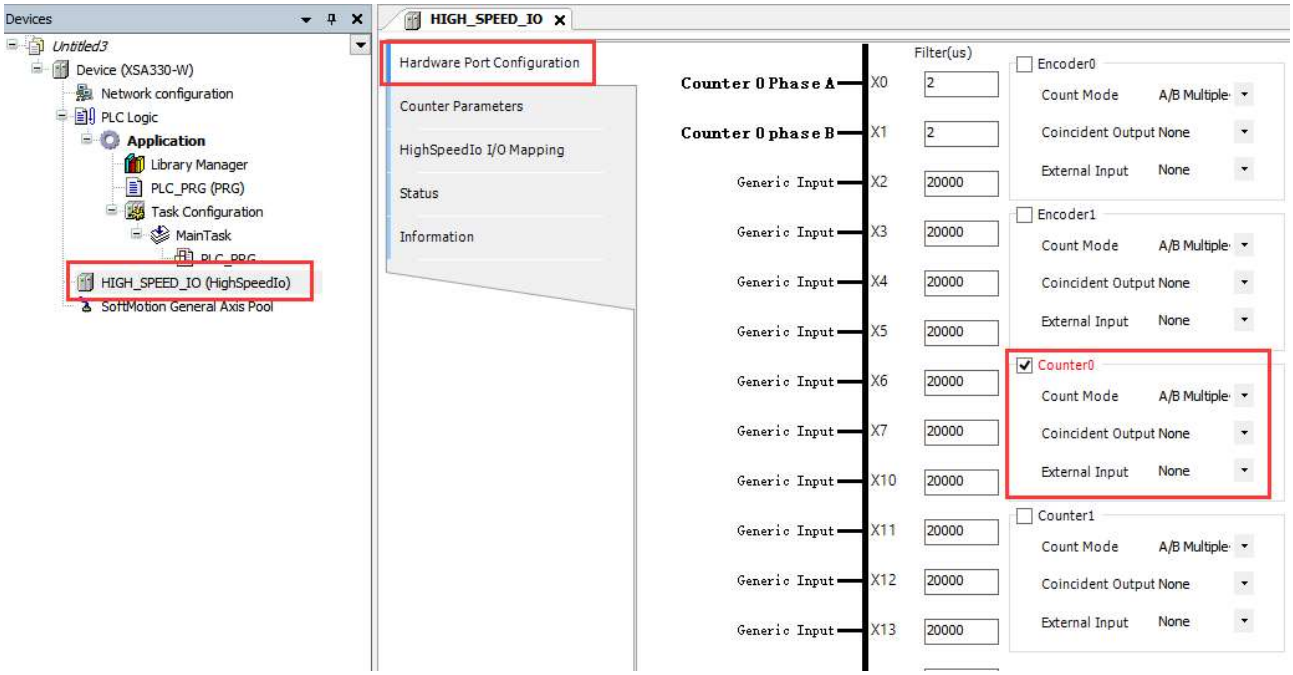
Input variable	Name	Data type	Effective range	Initial value	Description
Counter	High speed counting input port	XJ_COUNTER_REF	-	-	After configuring the high-speed counter input in the high-speed IO interface, it will be automatically instantiated. Select the corresponding instantiation name of the high-speed counter
xEnable	Enable	BOOL	TRUE, FALSE	FALSE	Normally ON the enable to count

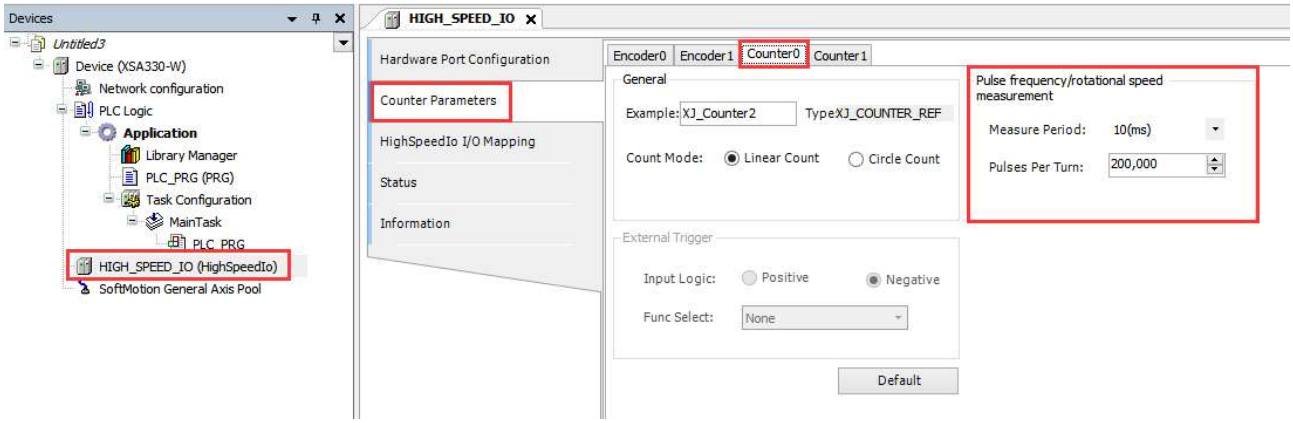
Input variable	Name	Data type	Effective range	Initial value	Description
eMode	Counting mode	HSC_EDGE_MODE	PosEdge, NegEdge, BothEdge	PosEdge	PosEdge: rising edge counting NegEdge: falling edge counting BothEdge: double edge counting
xDirection	Direction	BOOL	TRUE, FALSE	FALSE	FALSE: up counting TRUE: down counting

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
diValue	High speed counting value	DINT	Data range	0	High speed counting value
udiFrequency	Pulse frequency measurement value	UDINT		0	Unit: Hz, if it is a low frequency, it can be used in conjunction with the measurement period through the interface
udiRPM	Rotation speed per minute	UDINT		0	Unit: r/min, used in conjunction with the number of pulses per turn in the interface configuration
xBUSY	Busy	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

To use high-speed counting, it is necessary to check and configure the corresponding high-speed counter in the hardware parameters, set the counting method and pulse frequency/rotation speed measurement related parameters for the counter parameters.





Note:

(1) Please assigning a value to VAR_IN_OUT'Counter' when calling XJ_Counter_Enable, then can the program be compiled and downloaded.

(2) Notes when setting xDirection:
- When single-phase or AB phase counting, counting up/down based on the xDirection terminal status.
- When in P+D mode, the direction terminal FALSE and the xDirection terminal FALSE are counting up. Direction terminal FALSE, xDirection terminal TRUE, represents the subtraction count. The direction terminal is TRUE, while the xDirection terminal is FALSE, indicating a subtractive count. The direction terminal is true, and the xDirection terminal is true, indicating the addition count.

(3) The meaning of measurement cycle is to collect the number of pulses during this period for calculation:
- The measurement cycle is 1000ms, and the minimum measurable frequency is 1Hz. The measurement cycle is 1ms, and the minimum measurable frequency is 1kHz.
- The measurement cycle is 10ms, and the minimum measurable frequency is 100Hz. The measurement cycle is 100ms, and the minimum measurable frequency is 10Hz. The maximum measurement frequency is the frequency that can be received by high-speed counting.

(4) The unit of rotation speed per minute is r/min, and the measured values may fluctuate back and forth due to hardware influence.

(5) The difference between linear counting and circular counting:
- Linear counting: (DownLimitValue, UpLimitValue)
- Circular counting: [iRingDownValue, iRingUpValue]
- Counter 0-3: The linear counting range is (-2147483648, 2147483647), not include -2147483648 and 2147483647.
- Ring counting range is [-2147483648, 2147483647], include -2147483648 and 2147483647.

(6) When Counter0/Counter1 selects AB phase counting, only A-phase frequency is displayed. Set in single phase, AB phase interface.

(7) If the count value reaches near the boundary value, the counter overflow may not report an error due to the influence of the scanning cycle.

(8) When downloading the program, the counter value is not cleared to zero.

(9) Please enable the XJ_Counter_Enable before using XJ_Counter_Sample function block.

2-1-2-2. Compare consistent output【XJ_Counter_Compare】

Set consistent output. When high-speed counting interrupt function is required, XJ_EnableInterrupt needs to be called to turn on the interrupt.

Multi segment comparison and single segment comparison are executed in the triggering order, and the later triggered will not interrupt the first triggered. The execution triggered later will take effect after the execution triggered earlier is completed.

Compare consistent output [XJ_Counter_Compare]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_Counter_Compare	Compare consistent output	XJ_Counter_Compare_0 XJ_Counter_Compare EN ENO Counter xDone xExcute xBusy diCompareValue xError uiImRefreshCycle eErrorID	XJ_Counter_Compare_0(Counter:= , xExcute:= , diCompareValue:= , uiImRefreshCycle:= , xDone=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variable

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
xExecute	Trigger	BOOL	TRUE,FALSE	FALSE	Trigger
diCompareValue	appoint the compare value	DINT		0	
uiImRefreshCycle	Hardware direct output time	UINT		0	Unit: 100us, maximum output time is 3000ms.

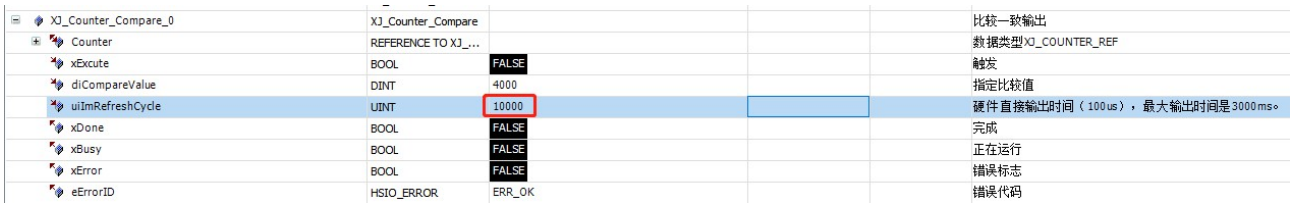
(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After the instruction execution is completed, the flag is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErr1orID	Error type	HSIO_ERROR	-	0	

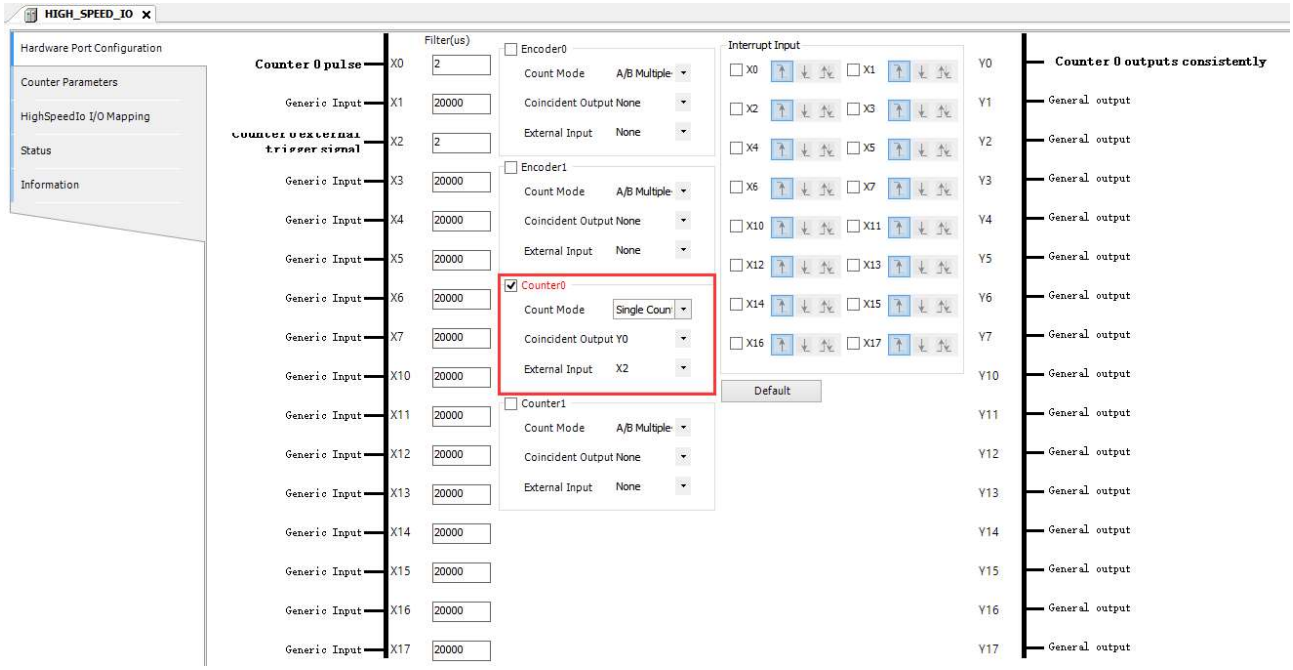
Note:

(1) The control of Y0-Y7 can be configured in the background, and each counter can freely select Y0-Y7, with a certain delay in output. The output time is 100us-3000ms. If ImRefreshCycle is 0, call XJ_Reset CmpOutput to lower the output.
(2) ImRefresh=1: Hardware immediately outputs, Y0-Y7 cannot be freely selected, there is no delay, and the output time is set through ImRefreshCycle. The output time is 0-3000ms.
(3) XJ_EnableInterrupt needs to be called in advance to open compare consistent interrupt.
(4) After hot reset and cold reset, maintain the previous comparison value.

Set compare consistent output and call XJ_EnableInterrupt to open the interrupt.
If hardware output is not set, it is necessary to turn on compare consistent interrupt. If hardware output is set, compare consistent interrupt may not be turned on.
For example, if the output time is set to 1 second, the hardware output of Y will be 1 second after comparing the values, and the software status of Y will remain unchanged, it will not display any output. Execute interrupt program after external trigger input.

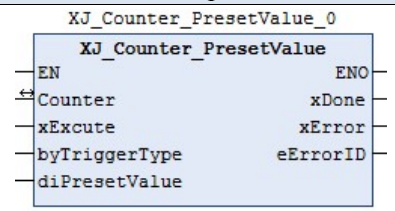
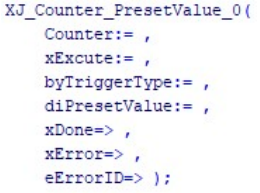


XJ_Counter_Compare_0	XJ_Counter_Compare				比较一致输出
Counter	REFERENCE TO XJ_...				数据类型XJ_COUNTER_REF
xExcute	BOOL	FALSE			触发
diCompareValue	DINT	4000			指定比较值
uiImRefreshCycle	UINT	10000			硬件直接输出时间（100us），最大输出时间是3000ms。
xDone	BOOL	FALSE			完成
xBusy	BOOL	FALSE			正在运行
xError	BOOL	FALSE			错误标志
eErrorID	HSIO_ERROR	ERR_OK			错误代码



2-1-2-3. Preset value write in【XJ_Counter_PresetValue】

Preset value write in [XJ_Counter_PresetValue]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_Counter_PresetValue	Preset value write in		

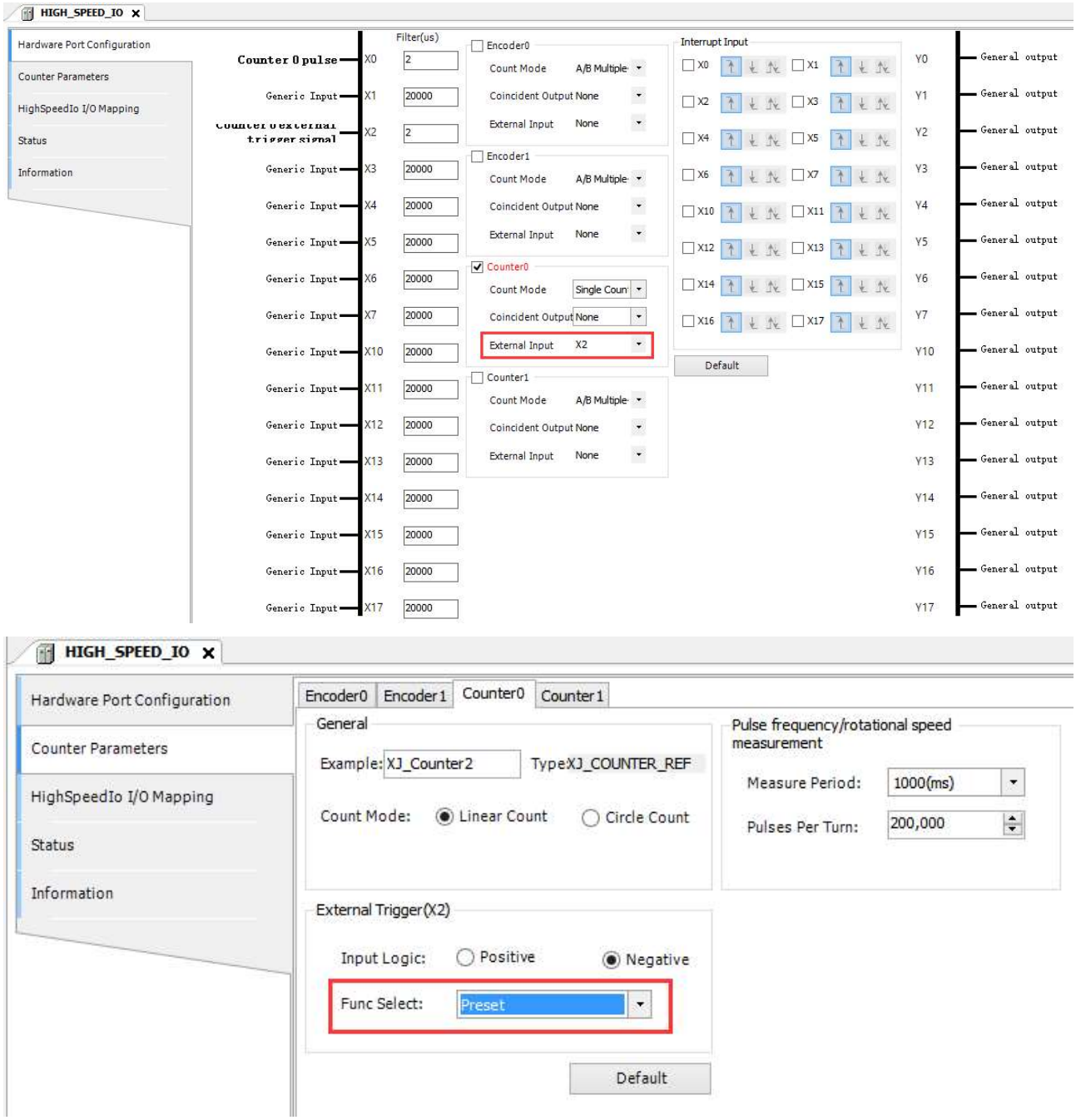
(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
xExecute	Trigger	BOOL	TRUE,FALSE	FALSE	Trigger
byTrigger Type	Trigger type	HSC_PSV_TIGGLE		0	0: Rising edge triggered write; 1: External input triggering; 2: When comparing consistent outputs, it is preset and triggered at the rising edge.
diPresetVa lue	Preset value	DINT	Data range	0	Write high-speed count preset value

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the write, the flag is TRUE
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

Note: If the value displayed for ErrorID is 2, it is because the CounterID range is not between 0 and 3.
0: Rising edge triggered writing. After the instruction is triggered, the preset value can be written into the high-speed count.
1: External input triggering. After the command is triggered, wait for the set external input terminal signal, and if there is a signal, the preset value can be written in.



2: When comparing consistent outputs, it is preset and triggered at the rising edge.

2-1-2-4. Probe【XJ_TouchProbe】

Probe [XJ_TouchProbe]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_TouchProbe	Probe	XJ_TouchProbe_0 XJ_TouchProbe EN Counter xExcute ENO xDone xBusy diTouchValue xError eErrorID	XJ_TouchProbe_0(Counter:= , xExcute:= , xDone=> , xBusy=> , diTouchValue=> , xError=> , eErrorID=>);

(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
xExcute	Trigger	BOOL	TRUE,FALSE	FALSE	Trigger

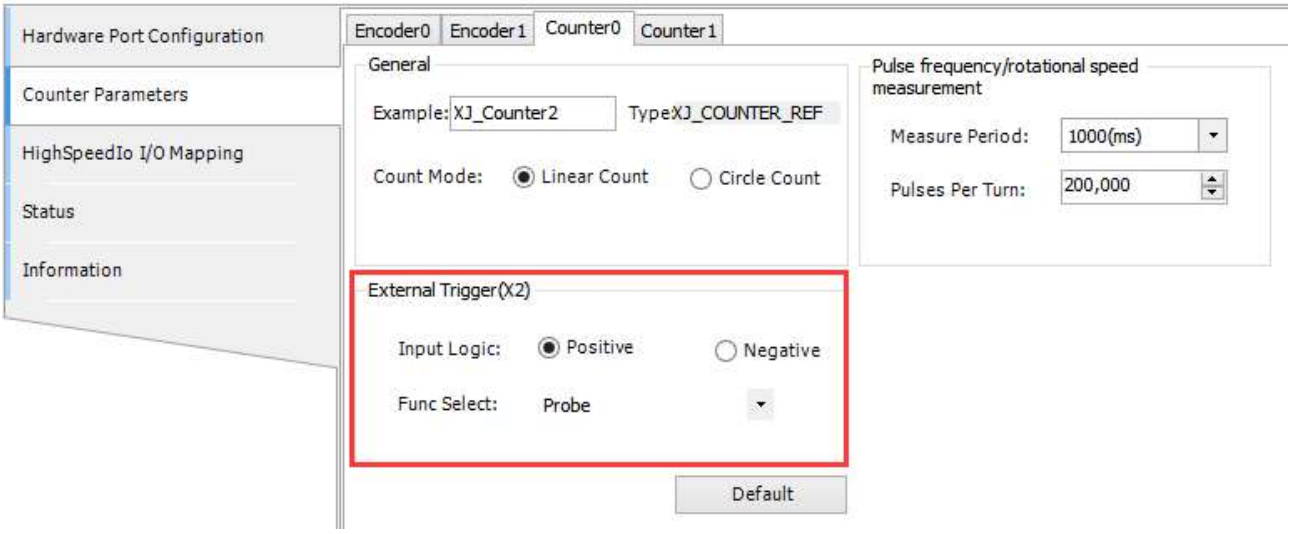
(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the write, the flag is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
diTouchVaule	Latch value	DINT		0	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

Set the terminal for the probe function. After triggering this terminal, the current high-speed count value can be locked, as shown in the following figure. The positive logic, takes effect after the rising edge is triggered.

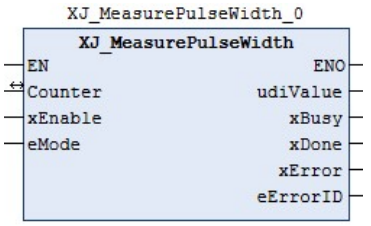
XJ_TouchProbe_0	XJ_TouchProbe				
Counter	REFERENCE TO XJ_COUNTER_REF				数据类型XJ_COUNTER_REF
xExcute	BOOL	FALSE			触发
xDone	BOOL	FALSE			完成标志
xBusy	BOOL	FALSE			正在运行
diTouchValue	DINT	0			锁存值
xError	BOOL	FALSE			错误标志
eErrorID	HSIO_ERROR	ERR_OK			错误代码

Hardware Port Configuration
Counter Parameters
HighSpeedIo I/O Mapping
Status
Information
Filter(us)
Counter 0 pulse
Generic Input
Counter 0 external trigger signal
Encoder0
Count Mode
A/B Multiple
Coincident Output
None
External Input
Encoder1
Counter0
Single Coun
Y0
External Input
X2
Counter1
Interrupt Input
Default
Counter 0 outputs consistently
General output


Hardware Port Configuration
Counter Parameters
HighSpeedIo I/O Mapping
Status
Information
Encoder0
Encoder1
Counter0
Counter1
General
Example:
XJ_Counter2
Type:
XJ_COUNTER_REF
Count Mode:
Linear Count
Circle Count
Pulse frequency/rotational speed measurement
Measure Period:
1000(ms)
Pulses Per Turn:
200,000
External Trigger(X2)
Input Logic:
Positive
Negative
Func Select:
Probe
Default

2-1-2-5. Read the pulse width measurement value of the counter【XJ_MeasurePulseWidth】

Read the pulse width measurement value of the counter [XJ_MeasurePulseWidth]			
Execution condition	Normally ON/OFF triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_MeasurePulseWidth	Read the pulse width measurement value of the counter		XJ_MeasurePulseWidth_0(Counter:= , xEnable:= , eMode:= , udiValue=> , xBusy=> , xDone=> , xError=> , eErrorID=>);

(1) Input variables

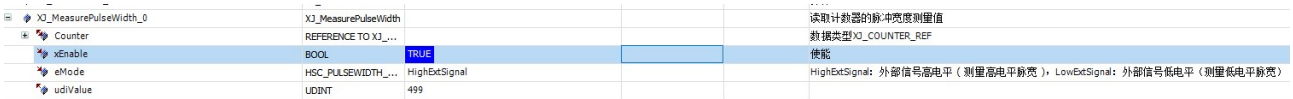
Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
xExecute	Trigger	BOOL	TRUE,FALSE	FALSE	Trigger
eMode	Measure high/low level pulse width	HSC_PULSEWIDTH_TYPE	0,1	0	0: External signal high level (measuring high level pulse width) 1: External signal low-level (measuring low-level pulse width)

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
udiValue	Measuring value	UDINT		0	
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the write, the flag is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

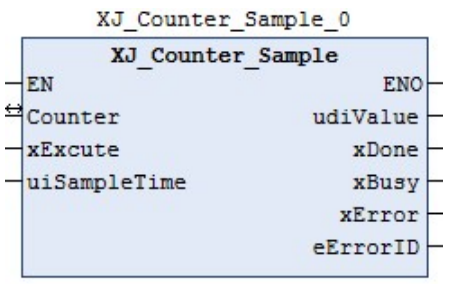
Note: Measure the duration of high and low levels in us microseconds.

Example: If the frequency is 1kHz, it is equivalent to sending a pulse every 1ms, and if there is a high or low level every 0.5ms, the measured high or low level is about 500us.



2-1-2-6. Counter sample【XJ_Counter_Sample】

Counter sample [XJ_Counter_Sample]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_Counter_Sample	Counter sample		XJ_Counter_Sample_0(Counter:= , xExcute:= , uiSampleTime:= , udiValue=> , xDone=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
eExecute	Trigger	BOOL	TRUE,FALSE	FALSE	Trigger
uiSampleTime	Sampl time	UINT		0	Sample time (10ms~65535ms)

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
udiValue	Sample value	UDINT		0	
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the write, the flag is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

Note:

(1) XJ_Counter_Sample can run after XJ_Counter_Enable is enabled.

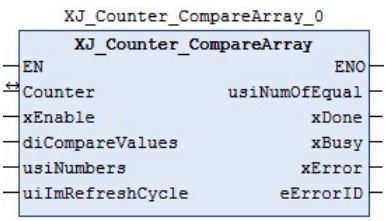
(2) After the command is triggered, the sampled signal count value will not change in real time, but will be displayed after the command is completed, displaying the high-speed count value collected within the set time.

2-1-2-7. Multiple segments compare【XJ_Counter_CompareArray】

Compare Consistent Output: Up to 100 comparison values can be set, and the Done signal is the output.

> Multi segment comparison and single segment comparison are executed in the triggering order, and the later triggered will not interrupt the first triggered. The execution triggered later will take effect after the execution triggered earlier is completed.

Multiple segments compare [XJ_Counter_CompareArray]			
Execution condition	Normally ON/OFF triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_Counter_CompareArray	Multiple segments compare		XJ_Counter_CompareArray_0(Counter:= , xEnable:= , diCompareValues:= , usiNumbers:= , uiImRefreshCycle:= , usiNumOfEqual=> , xDone=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
xEnable	Enable	BOOL	TRUE,FALSE	FALSE	Enable
diCompareValues	Comparison value one-dimensional array	ARRAY		0	A one-dimensional array of comparison values for the set counters, supporting a maximum of 100
usiNumbers	set actual value	USINT		0	The actual value set, allowed to be set to 1, with a maximum of 100
uiImRefreshcycle	hardware direct output time	UINT		0	The unit is 100us, and the maximum output time is 3000ms. For example, setting 10000 means 1000ms

(2) Output variables

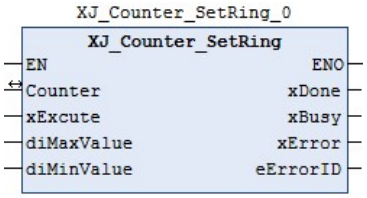
Output variables	Name	Data type	Effective range	Initial value	Description
usiNumOfEqual	Equal numbers	USINT		0	
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the write, the flag is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

- Multiple comparison values can be set, and when the comparison value is reached, it will enter the task execution instruction of external events, with hardware output.
- Difference from single segment comparison: It is necessary to open a compare consistent interrupt, otherwise the instruction will report an error.
- As shown in the figure below, set the corresponding parameters as needed:

XJ_Counter_CompareArray_0	XJ_Counter_CompareArray				多段比较
Counter	REFERENCE TO XJ_COUNTER_REF				数据类型XJ_COUNTER_REF
xEnable	BOOL	FALSE			使能
diCompareValues	ARRAY [0..99] OF DINT				设置的计数器的比较值一维数组，最大支持 100 个
usiNumbers	USINT	0			设置的实际值，最大100个
uiImRefreshCycle	UINT	20000			硬件直接输出时间（单位：100us），最大输出时间是3000ms。
usiNumOfEqual	USINT	0			相等个数
xDone	BOOL	FALSE			完成标志
xBusy	BOOL	FALSE			正在运行
xError	BOOL	FALSE			错误标志
eErrorID	HSIO_ERROR	ERR_OK			错误代码

2-1-2-8. Ring counting【XJ_Counter_SetRing】

Ring counting [XJ_Counter_SetRing]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_Counter_SetRing	Set ring counter		XJ_Counter_SetRing_0(Counter:= , xExcute:= , diMaxValue:= , diMinValue:= , xDone=> , xBusy=> , xError=> , eErrorID=>);

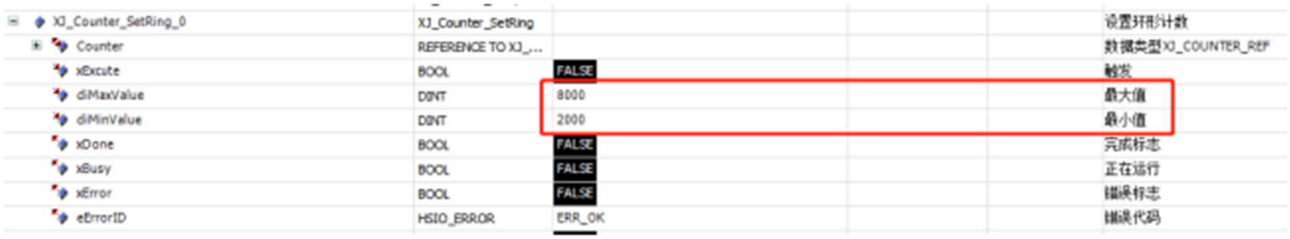
(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
xExcute	Trigger	BOOL	TRUE,FALSE	FALSE	Trigger
diMaxValue	Maximum value	DINT		0	Maximum value
diMinValue	Minimum value	DINT		0	Minimum value

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the write, the flag is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

- The current count value needs to be within the set value range, otherwise the command will report an error. After the instruction is executed, no instruction can switch back to linear counting. If switching back to linear technology is required, the counting range needs to be changed to a linear counting range.



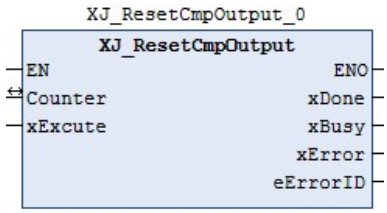
- The minimum value for linear counting is -2147483647, and the maximum value is 2147483646. For example, adding one signal input after increasing the count to 2147483646 will change to -2147483647, and then continue counting upwards. It will not reach the minimum value of double word signed numbers -2147483648 and the maximum value of 2147483647.

- There are two configuration methods for ring counting: firstly, when not logged in, in the "High speed IO"

interface - "Counter parameter settings", you can click "Ring counting" and set the range. The second is to use the XJ_Counter_SetRing instruction setting, which can directly switch linear counting to circular counting during login.

2-1-2-9. Reset port of compare consistent output【XJ_ResetCmpOutput】

Reset port of compare consistent output [XJ_ResetComOutput]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_ResetCmpOutput	Reset the compare consistent output port		XJ_ResetCmpOutput_0(Counter:= , xExcute:= , xDone=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
xExcute	Trigger	BOOL	TRUE,FALSE	FALSE	Trigger

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the setting, the flag is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

Release all compare consistent output ports. At this time, Y, which was originally set as a compare consistent output, can be used as a normal Y. As for whether the Y-point is used as a compare consistent output or a regular output, it depends on which port instruction is triggered first for the compare consistent output command and reset the compare consistent output command.

2-1-2-10. Clear the error【XJ_Counter_Reset】

Clear errors related to high-speed counting instructions.

Clear the error [XJ_Counter_Reset]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_Counter_Reset	Clear the error	XJ_Counter_Reset_0 XJ_Counter_Reset EN Counter xExcute ENO xDone xBusy xError eErrorID	XJ_Counter_Reset_0(Counter:= , xExcute:= , xDone=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	High speed counter input port	XJ_COUNTER_REF	-	-	High speed counter, it is necessary to define the high-speed counting input terminal (see functional description)
xExcute	Trigger	BOOL	TRUE,FALSE	FALSE	Trigger

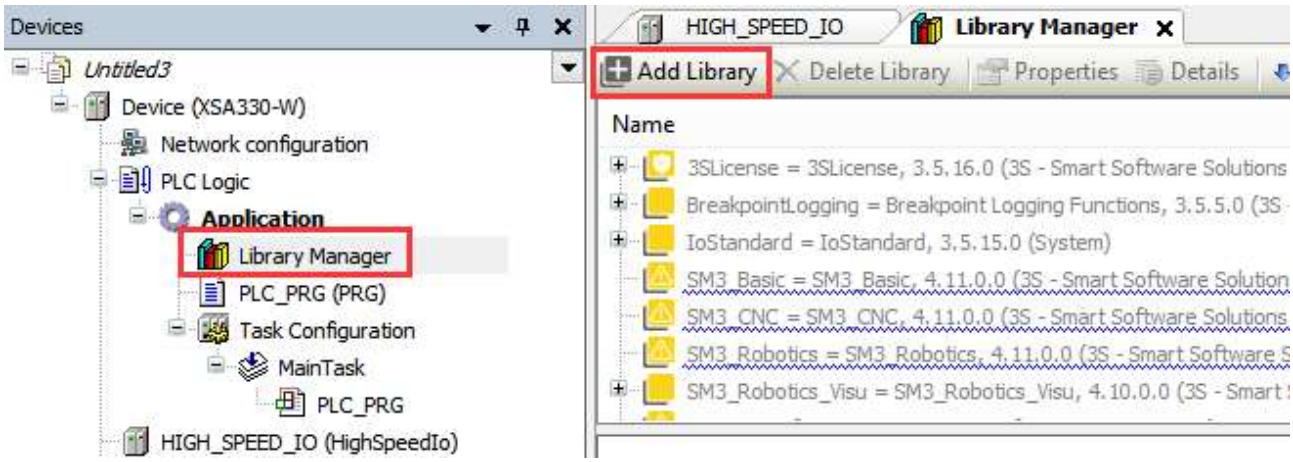
(2) Output variables

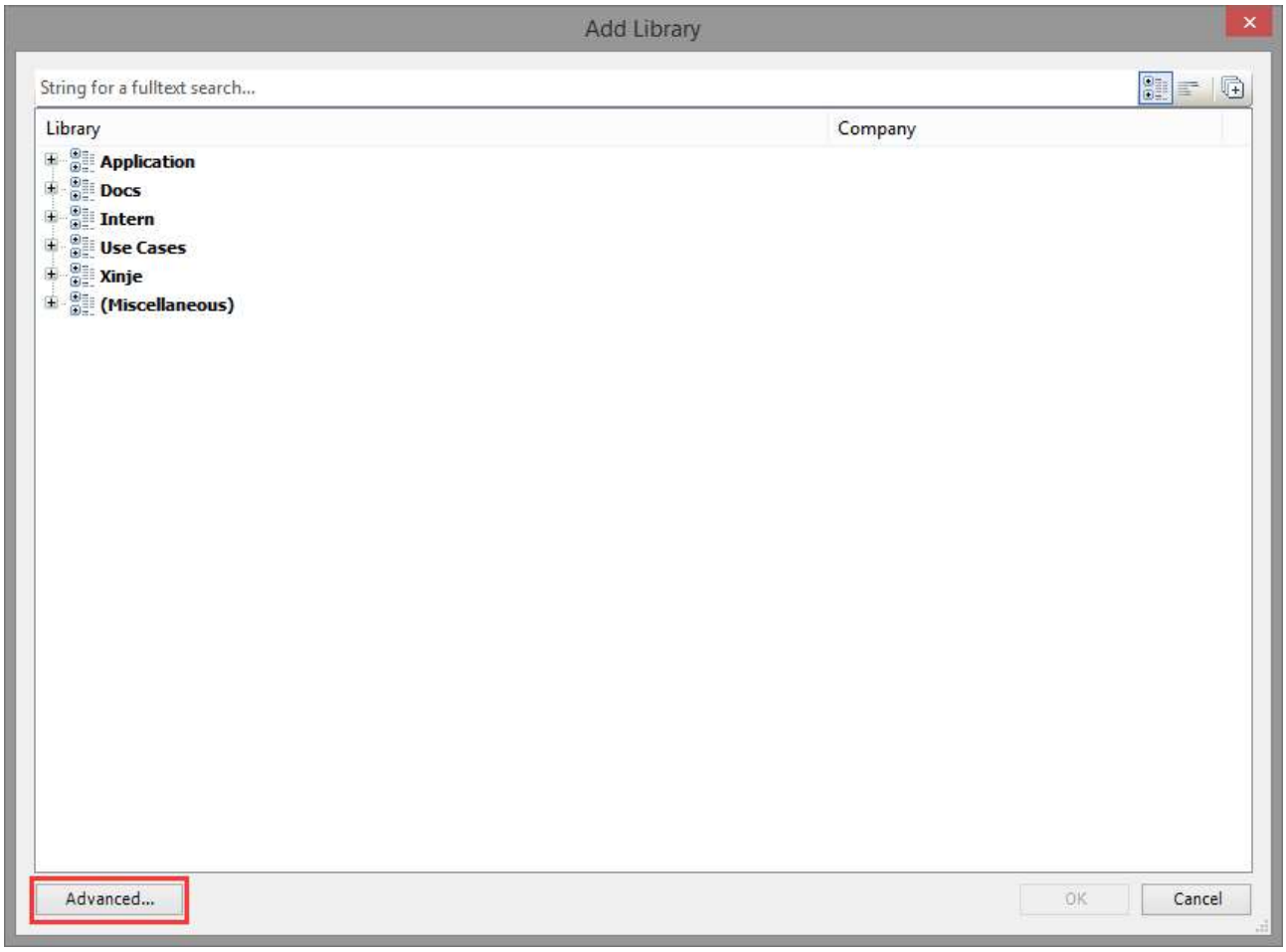
Output variables	Name	Data type	Effective range	Initial value	Description
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the setting, the flag is TRUE

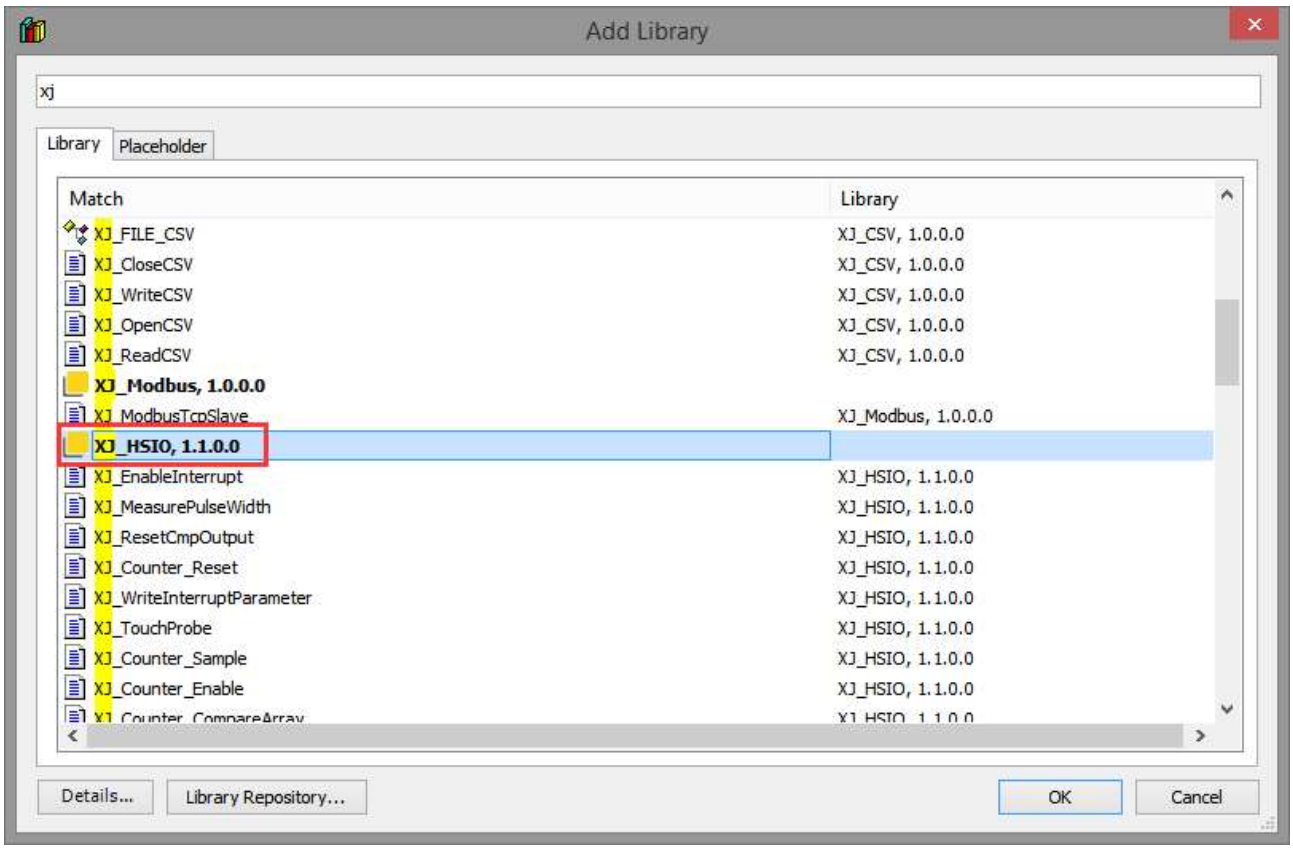
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

2-1-3. Parameter settings

Add Library File: Library Manager - Add Library - Advanced - Add XJ_HSIO.







2-2. XS series high speed count instructions

2-2-1. Function overview

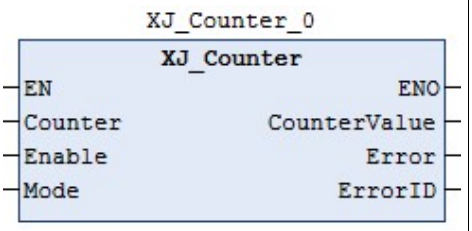
The XS series PLC has a high-speed counting function, which enables the measurement of high-speed input signals such as measurement sensors and rotary encoders by selecting different counters. Its maximum measurement frequency can reach 200kHz.

This chapter's instruction library only supports XS3-26T4, XSDH-60A32-E, and XSLH-30A32.

2-2-2. Function block

2-2-2-1. High speed counter【XJ_Counter】

High speed counter [XJ_Counter]			
Execution condition	Normally ON/OFF	Suitable model	XS3, XSDH, XSLH-30A32
Firmware	-	Software	-
Library	XinjeCnt		

Instruction	Name	Graph	ST language
XJ_Counter	High speed count		XJ_Counter(Counter:= , Enable:= , Mode:= , CounterValue=> , Error=> , ErrorID=>);

(1) Input variables

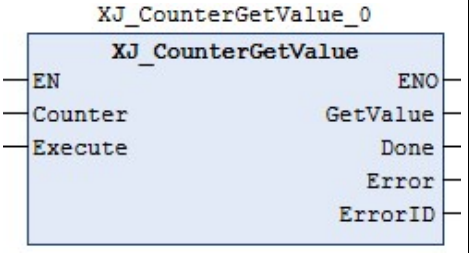
Input variables	Name	Data type	Effective range	Initial value	Description
Counter	Counter	COUNTER_REF	-	-	High speed counter, which specifies the input terminal and initial value of high-speed counting
Enable	Enable	BOOL	TRUE,FALSE	FALSE	Normally open the enable to start counting
Mode	Count mode	Mode	AB_Mode，Single_Mode	FALSE	High speed count mode: MODE=XJ.AB_Mode is AB phase high speed count. MODE=XJ.Single_Mode is single phase high speed count.

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
CounterValue	Count value	DINT	Data range	0	High speed count value
Error	Error flag	BOOL	TRUE,FALSE	FALSE	
ErrorID	Error type	UINT	-	0	

2-2-2-2. Read high speed counter【XJ_CounterGetValue】

Read high speed counter [XJ_CounterGetValue]			
Execution condition	Normally ON/OFF	Suitable model	XS3, XSDH, XSLH-30A32
Firmware	-	Software	-
Library	XinjeCnt		

Instruction	Name	Graph	ST language
XJ_CounterGetValue	Read high speed counter		XJ_CounterGetValue(Counter:= , Execute:= , GetValue=> , Done=> , Error=> , ErrorID=>);

(1) Input variables

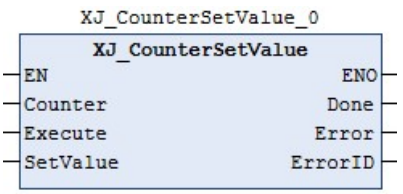
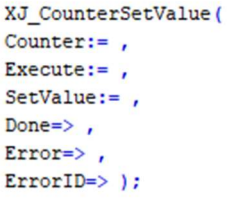
Input variables	Name	Data type	Effective range	Initial value	Description
Counter	Counter	COUNTER_REF	-	-	High speed counter, which specifies the input terminal and initial value of high-speed counting
Execute	Enable	BOOL	TRUE,FALSE	FALSE	Triggered by the rising edge, reading the current high-speed count value

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetValue	Read value	DINT	Data range	0	Present count value
Done	Completion	BOOL	TRUE,FALSE	FALSE	After completing the read, the flag is TRUE
Error	Error flag	BOOL	TRUE,FALSE	FALSE	
ErrorID	Error type	UINT	-	0	

2-2-2-3. Write high speed counter【XJ_CounterSetValue】

Write high speed counter [XJ_CounterSetValue]			
Execution condition	Normally ON/OFF	Suitable model	XS3, XSDH, XSLH-30A32
Firmware	-	Software	-
Library	XinjeCnt		

Instruction	Name	Graph	ST language
XJ_CounterSetValue	Write high speed counter		


```
XJ_CounterSetValue(
Counter:= ,
Execute:= ,
SetValue:= ,
Done=> ,
Error=> ,
ErrorID=> );
```

(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Counter	Counter	COUNTER_REF	-	-	High speed counter, which specifies the input terminal and initial value of high-speed counting
Execute	Enable	BOOL	TRUE,FALSE	FALSE	Trigger by rising edge, write high-speed count value, write SetValue in CounterValue
SetValue	Write in value	DINT	Data range	0	Write high-speed counting set

					value

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
Done	Completion	BOOL	TRUE,FALSE	FALSE	After completing the write, the flag bit is TRUE
Error	Error flag	BOOL	TRUE,FALSE	FALSE	
ErrorID	Error type	UINT	-	0	

Note: If the value displayed for ErrorID is 2, it is because the CounterID range is not between 0 and 3.

(3) Function description

- The high-speed counting function has three functional blocks, namely the high-speed counting function block, the read high-speed counting function block, and the write high-speed counting function block. XS3 series high-speed input can only receive differential signal (DIFF), but cannot receive Open collector signal (OC). Please be sure to select differential signal encoder. XSDH series high-speed input is to receive Open collector signal (OC).
- Counter is COUNTER_REF data type:
  COUNTER_REF specific description is as follows:

Member	Name	Data type	Effective range	Initial value	Description
CounterID	Counter terminal	INT	0,1,2,3	0	Select high-speed counter input port
CounterValue	Counter initial value	DINT	data range	0	Setting the initial value of the counter

- The XS3 and XSDH series high-speed counting functions have two modes, namely single phase incremental mode and AB phase mode.

(1) Mode= Single_Mode

In this mode, the input pulse signal is counted, and the count value increases with the rising edge of each pulse signal.

(2) Mode=AB_Mode

In this mode, the high-speed counting value is incremented or decremented based on the pulse signal (A phase and B phase) with a phase difference of 90°. The default counting mode is 4 times frequency.

- XS series high-speed counting input port allocation

XS3-26T4								
	Single phase mode				AB phase mode			
CounterID	0	1	2	3	0	1	2	3
Max frequency	200k	200k	200k	200k	200k	200k	200k	200k
X0+	U+				A+			
X0-	U-				A-			
X1+					B+			
X1-					B-			
X2								
X3+		U+				A+		
X1-		U-				A-		
X4+						B+		

XS3-26T4								
	Single phase mode				AB phase mode			
X2-						B-		
X5								
X6+			U+				A+	
X6-			U-				A-	
X7+							B+	
X7-							B-	
X10								
X11+				U+				A+
X11-				U-				A-
X12+								B+
X12-								B-
X13								

XSDH-60A32-E								
	Single phase mode				AB phase mode			
CounterID	0	1	2	3	0	1	2	3
Max frequency	200k	200k	200k	200k	100k	100k	100k	100k
X0	U				A			
X1					B			
X2								
X3		U				A		
X4						B		
X5								
X6			U				A	
X7							B	
X10								
X11				U				A
X12								B
X13								

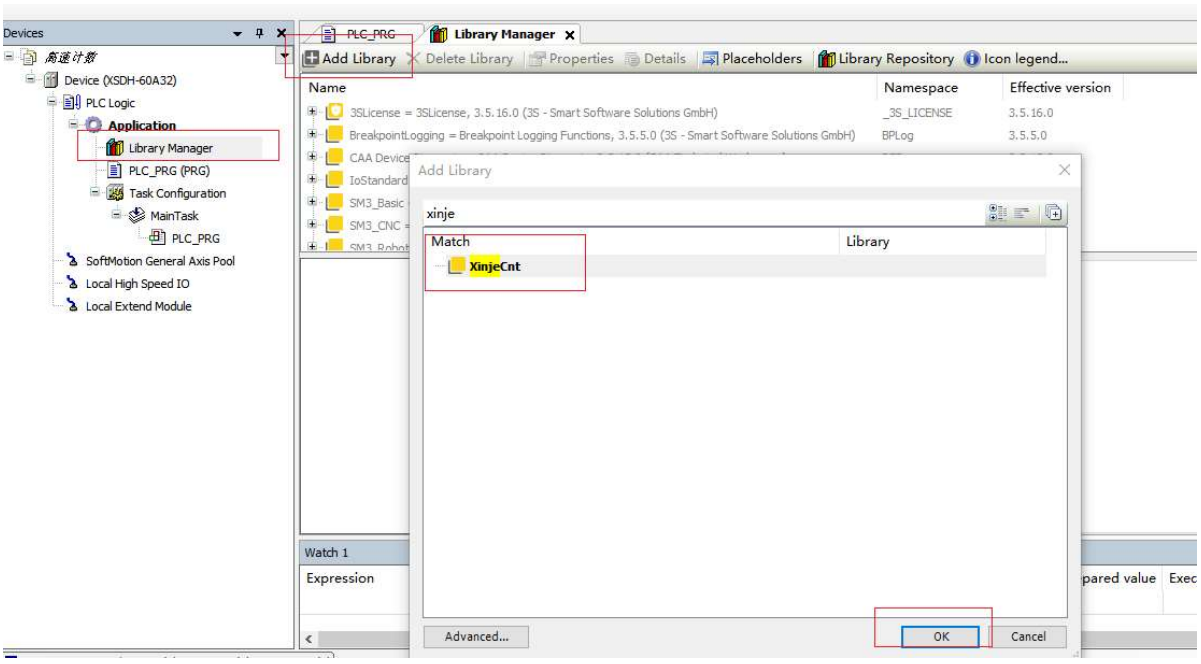
XSLH-30A32								
	Single phase mode				AB phase mode			
CounterID	0	1	2	3	0	1	2	3
Max frequency	200k	200k	80k	80k	100k	100k	50k	50k
X0+	U+				A+			
X0-	U-				A-			
X1+					B+			
X1-					B-			
X2								
X3+		U+				A+		

XSLH-30A32								
	Single phase mode				AB phase mode			
X1-		U-				A-		
X4+						B+		
X2-						B-		
X5								
X6			U				A	
X7							B	
X10								
X11				U				A
X12								B
X13								
X14								
X15								

2-2-3. Parameter configuration

Add library file:

Add "XinjeCnt" in the "Library Manager", and after adding it, you can use the high-speed counting function.



2-2-4. Application example

Example 1: Use the first channel of high-speed count and read the current count value in the count to modify the current high-speed count value.

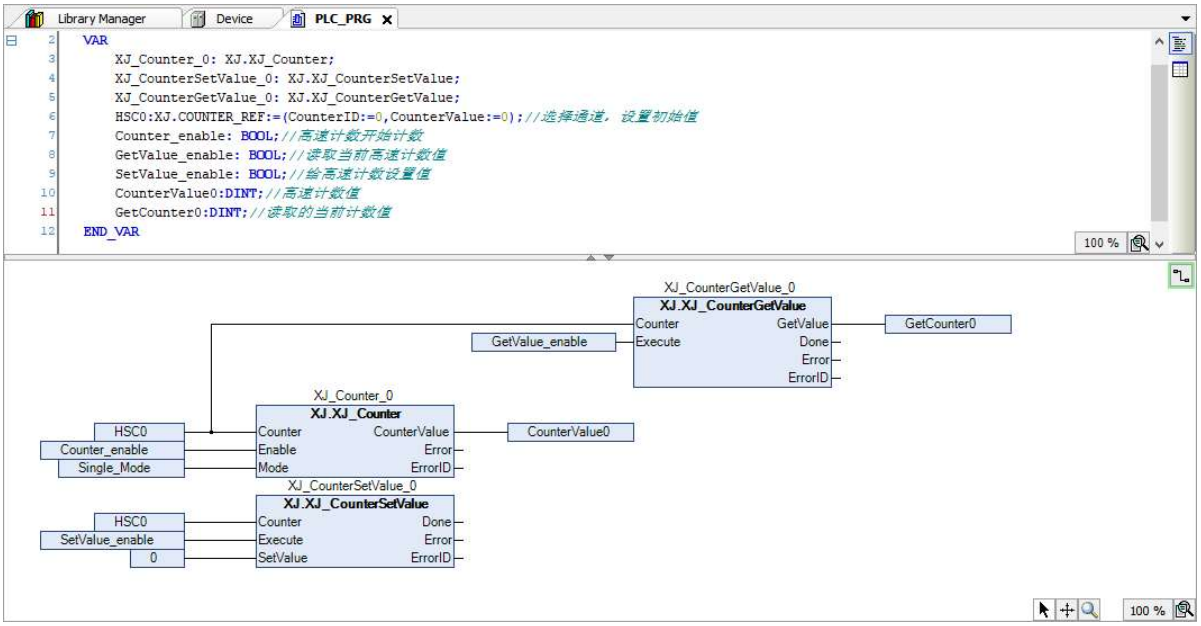
Program operation:

(1) Install the required libraries according to the steps in sections 2-1-3.

(2) Write a high-speed counting program.

Programming: Use the function blocks "XJ.XJ_Counter", "XJ.XJ_CounterGetValue", and

"XJ.XJ_CounterSetValue" to set the high-speed counting port, high-speed counting mode, and high-speed counting value used in the program.



2-3. External interrupt and compare consistent interrupt instructions

2-3-1. Function overview

The XSA series PLC supports X-terminal interrupt, and the same terminal supports rising and falling edge interrupts. In Codesys, interrupts are used through external events in the task type. Like X2R_TRIG represents X2 rising edge interrupt, X2F_TRIG represents the falling edge interrupt, and the number and type of interrupts supported by each model can be found in the "External event" option.

XS3-26T4, XSDH-60A32-E, and XSLH-30A32 currently do not support this instruction library. For external interrupts, please refer to the "XS Series PLCopen Standard Controller User Manual [Software Section]" manual.

2-3-2. Function block

2-3-2-1. Open external interrupts and compare consistent interrupts【XJ_EnableInterrupt】

Open external interrupts and compare consistent interrupts [XJ_EnableInterrupt]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_EnableInterrupt	Open external interrupts and compare consistent interrupts	XJ_EnableInterrupt_0 XJ_EnableInterrupt EN / ENO xEnable / xValid udiExternal / xBusy uiCompare / xError eErrorID	XJ_EnableInterrupt(xEnable:= , udiExternal:= , uiCompare:= , xValid=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
xEnable	Enable	BOOL	TRUE,FALSE	FALSE	Enable
udiExternal	Open external input interrupt	UINT		0	
uiCompare	Open compare consistent interrupt	UINT		0	For example, 4 means CounterID:=2

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
xValid	Interruption take effect	BOOL	TRUE,FALSE	FALSE	After the interrupt takes effect, the flag bit is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

- **Open external input interrupt:**

① Select external interrupt terminal, such as X1

② Open XJ_EnableInterrupt to add interrupt task, where 1 represents X0, 2 represents X1, 3 represents X0, X1, and so on. As shown in the following figure, external interrupt tasks are executed when the rising edge of X1 is effective.

That is, the udiExternal opens the external output interrupt and displays it as a binary address, and then gives a

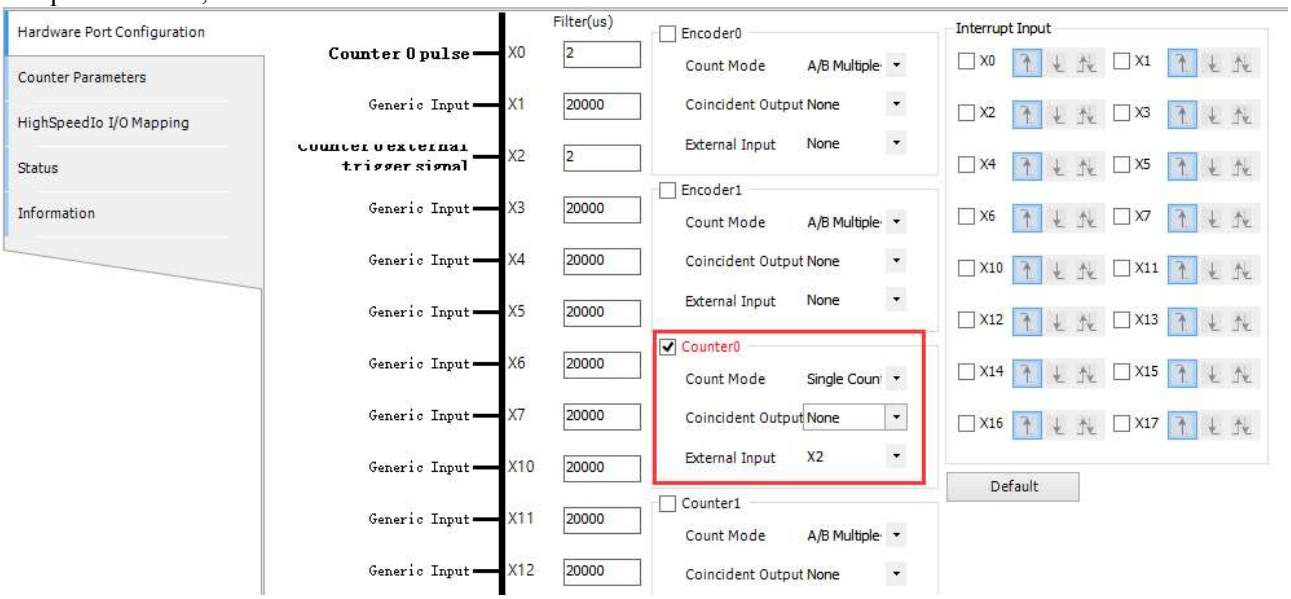
value. For example, if you fill in binary 2#11 or decimal 10#3, then open X0 and X1. In the "Hardware Parameter Configuration", you also need to check the required X0 and X1, which can be selected as "edge" or "double edge". If there is a signal from X0 and X1, the task configured as "X0" or "X1" will execute the program.

XJ_EnableInterrupt_0	XJ_EnableInterrupt				打开外部中断和比较一致中断
xEnable	BOOL	FALSE			使能
udiExternal	UINT	2			打开外部输入中断
uiCompare	UINT	4			打开比较一致中断
xValid	BOOL	FALSE			中断生效
xBusy	BOOL	FALSE			正在运行
xError	BOOL	FALSE			错误标志
eErrorID	HSIO_ERROR	ERR_OK			错误代码

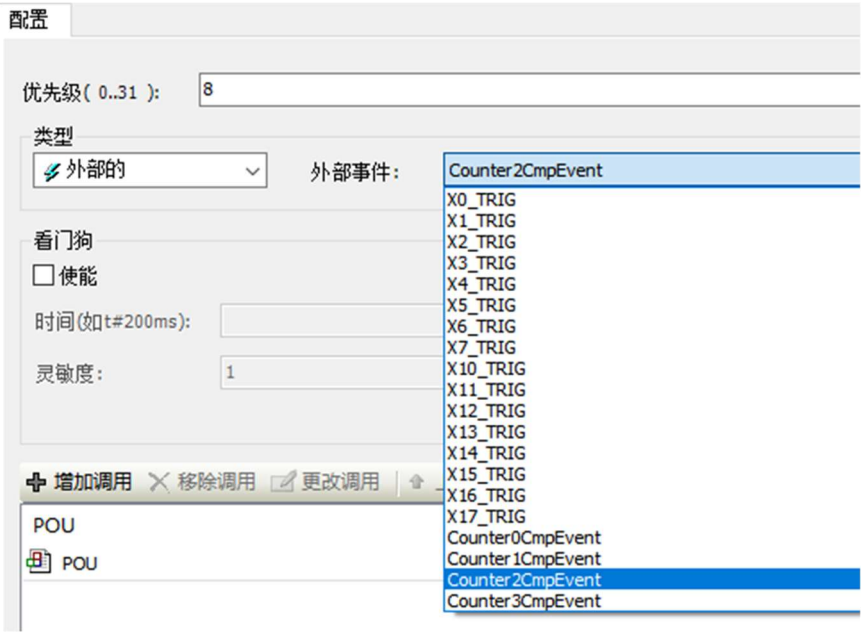
The port needs to be configured on both the interface and instructions to take effect.

- **Open compare consistent interrupt:**

Set the counter ID that needs to be opened, such as 3 indicating CounterID:=0 and CounterID:=1. In the case, M210 and CounterID:=2 are used, so opening a compare consistent interrupt to write 4. After reaching the comparison value, it will enter the task execution instruction of external events.



XJ_EnableInterrupt_0	XJ_EnableInterrupt				打开外部中断和比较一致中断
xEnable	BOOL	FALSE			使能
udiExternal	UINT	0			打开外部输入中断
uiCompare	UINT	4			打开比较一致中断
xValid	BOOL	FALSE			中断生效
xBusy	BOOL	FALSE			正在运行
xError	BOOL	FALSE			错误标志
eErrorID	HSIO_ERROR	ERR_OK			错误代码


配置
优先级(0..31):
8
类型
外部的
外部事件:
Counter2CmpEvent
X0_TRIG
X1_TRIG
X2_TRIG
X3_TRIG
X4_TRIG
X5_TRIG
X6_TRIG
X7_TRIG
X10_TRIG
X11_TRIG
X12_TRIG
X13_TRIG
X14_TRIG
X15_TRIG
X16_TRIG
X17_TRIG
Counter0CmpEvent
Counter1CmpEvent
Counter2CmpEvent
Counter3CmpEvent
看门狗
使能
时间(如t#200ms):
灵敏度:
1
增加调用
移除调用
更改调用
POU
POU

2-3-2-2. Write in interrupt parameters【XJ_WriteInterruptParameter】

Write in interrupt parameters [XJ_WriteInterruptParameter]			
Execution condition	Edge triggering	Suitable model	XSA, XSLH-24
Firmware	-	Software	-
Library	XJ_HSIO		

Command	Name	Graph	ST language
XJ_WriteInterruptParameter	Interrupt parameter writing	XJ_WriteInterruptParameter… XJ_WriteInterruptParameter EN ENO Port xDone xExcute xBusy byValue xError eErrorID	XJ_WriteInterruptParameter(Port:= , xExcute:= , byValue:= , xDone=> , xBusy=> , xError=> , eErrorID=>);

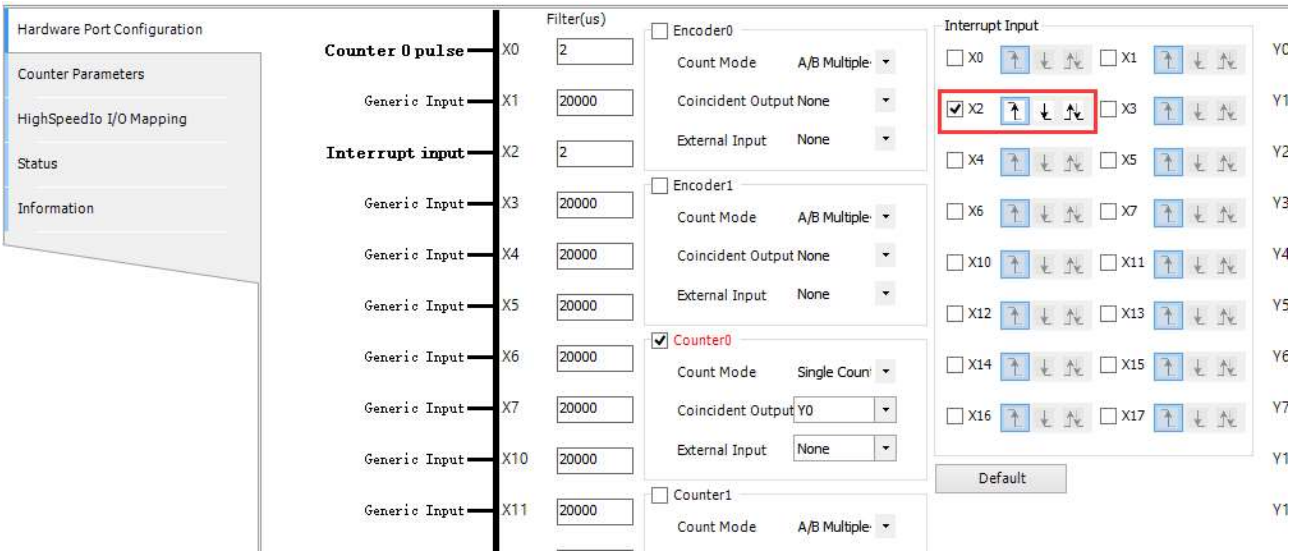
(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Port	Port no.	UINT		0	Binary, such as 2#11 representing X0 and X1
xExcute	Trigger	BOOL	TRUE,FALSE	FALSE	
byValue	Value	BYTE		0	

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
xDone	Completed flag	BOOL	TRUE,FALSE	FALSE	After completing the setting, the flag bit is TRUE
xBusy	Running	BOOL	TRUE,FALSE	FALSE	
xError	Error flag	BOOL	TRUE,FALSE	FALSE	
eErrorID	Error type	HSIO_ERROR	-	0	

External interrupt parameters can be configured on the interface or modified using the command.



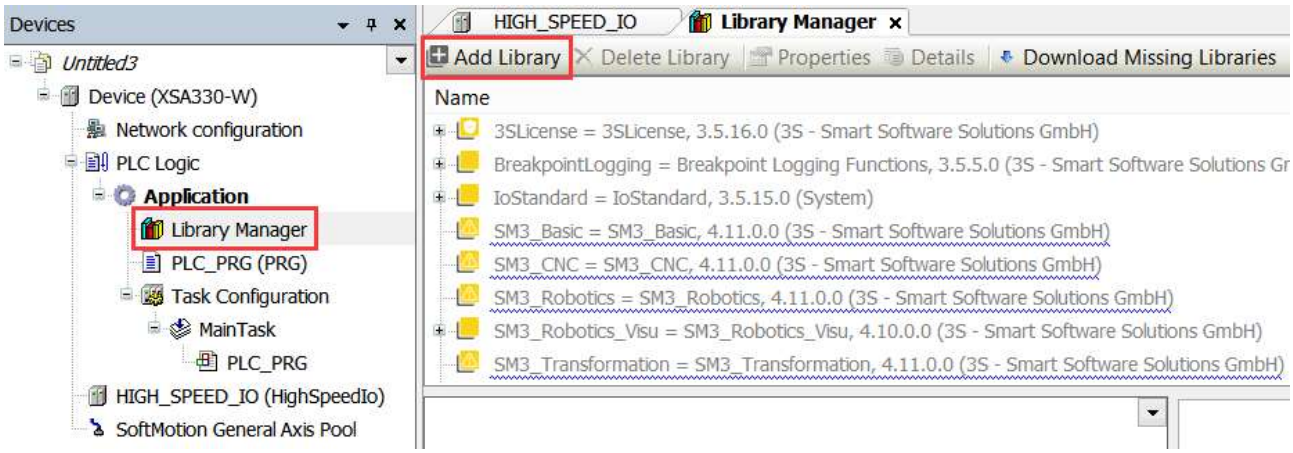
XJ_WriteInterruptParameter_0	XJ_HSIO.XJ_WriteInterruptParameter				中断参数写入
Port	UINT	4			端口号二进制如：2#11代表X0、X1
xExcute	BOOL	FALSE			触发
byValue	BYTE	0			值(0:上升沿，1为下降沿，2为上升沿+下降沿)
xDone	BOOL	FALSE			完成标志
xBusy	BOOL	FALSE			正在运行
xError	BOOL	FALSE			错误代码
eErrorID	HSIO_ERROR	ERR_OK			错误标志

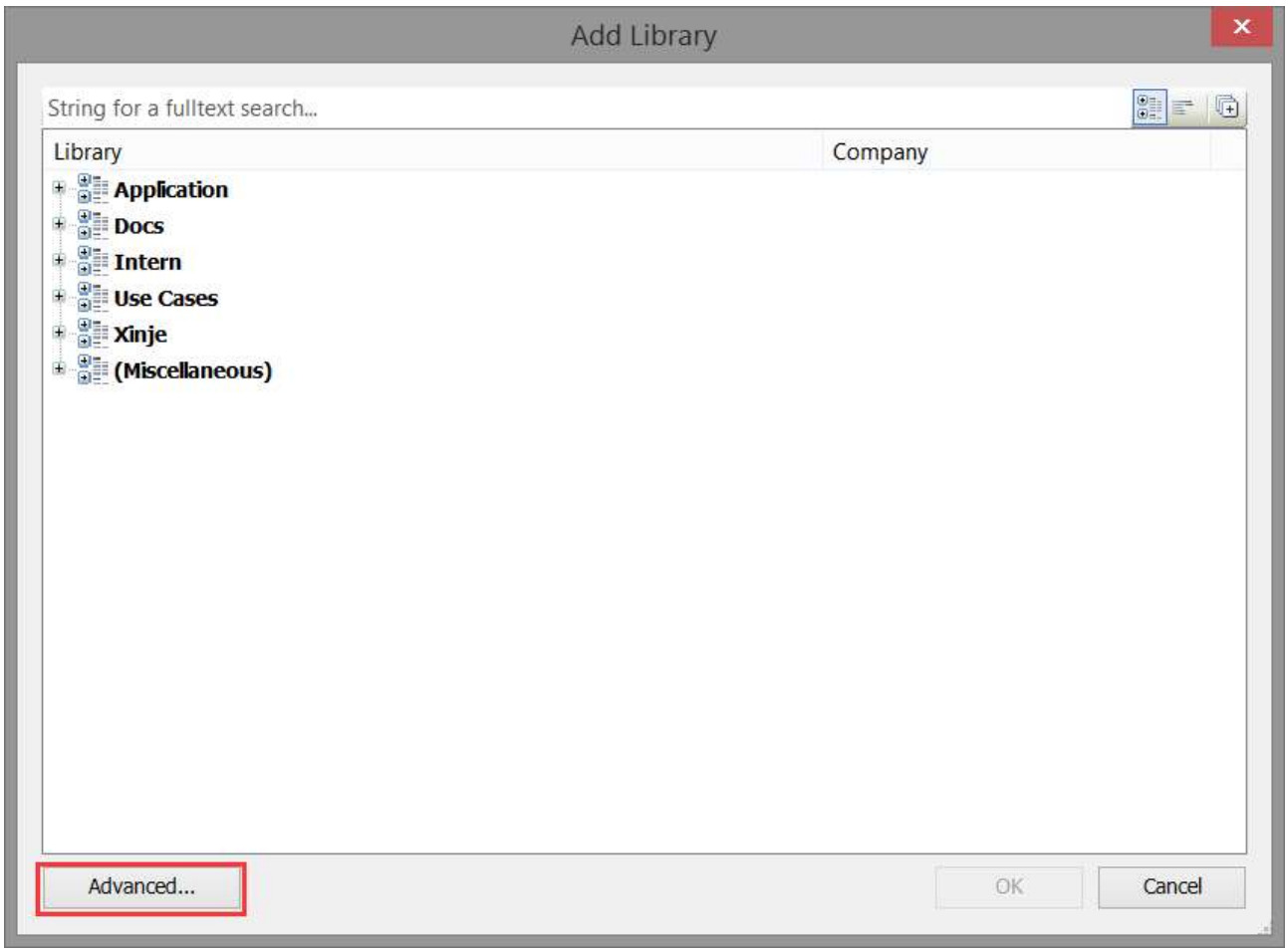
- XJ_WriteInterruptParameter executed before XJ_EnableInterrupt, then XJ_WriteInterruptParameter parameters are valid.
- XJ_EnableInterrupt executed before XJ_WriteInterruptParameter, background interrupt parameters are valid. XJ_WriteInterruptParameter executed again, then XJ_WriteInterruptParameter parameters are valid.
- When executing this command, it is also necessary to open external input interrupts.

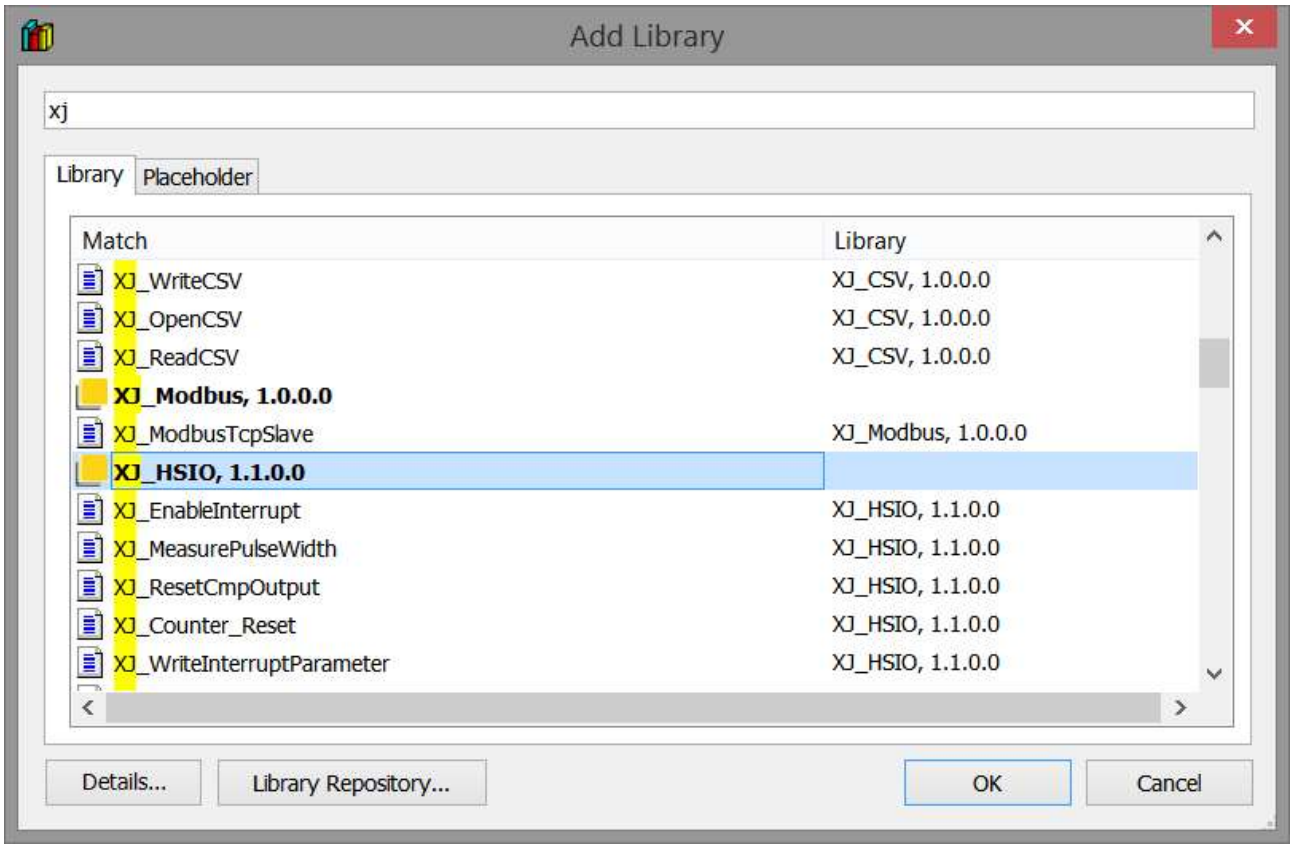
XJ_EnableInterrupt_0	XJ_EnableInterrupt				打开外部中断和比较一致中断
xEnable	BOOL	FALSE			使能
udiExternal	UINT	4			打开外部输入中断
uiCompare	UINT	4			打开比较一致中断
xValid	BOOL	FALSE			中断生效
xBusy	BOOL	FALSE			正在运行
xError	BOOL	FALSE			错误标志
eErrorID	HSIO_ERROR	ERR_OK			错误代码

2-3-3. Parameter configuration

Add Library File: Library Manager - Add Library - Advanced - Add XJ_HSIO.

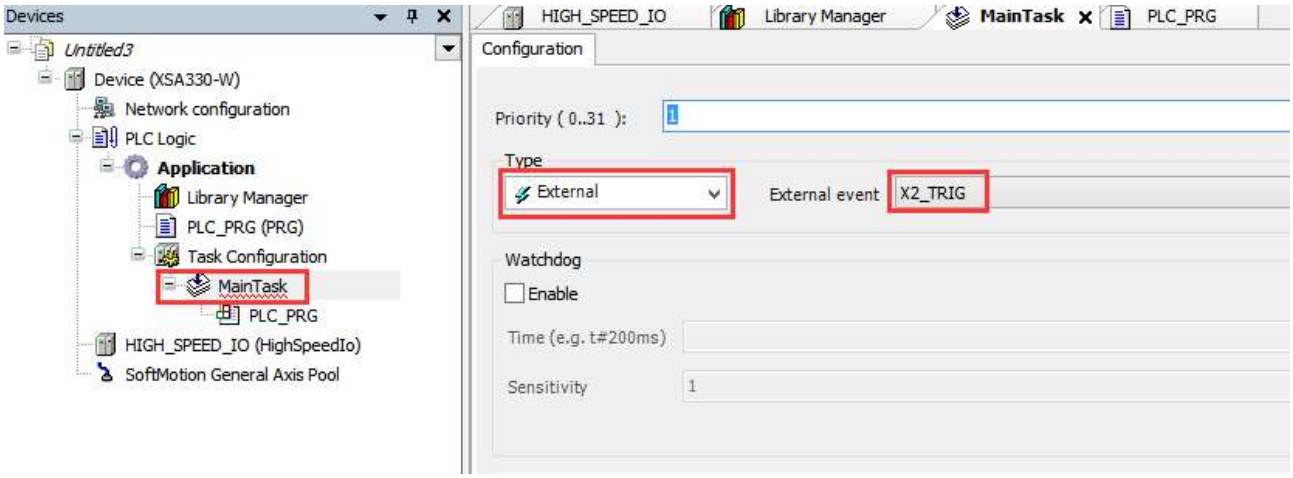



Add Library
String for a fulltext search...
Library
Company
Application
Docs
Intern
Use Cases
Xinje
(Miscellaneous)
Advanced...
OK
Cancel

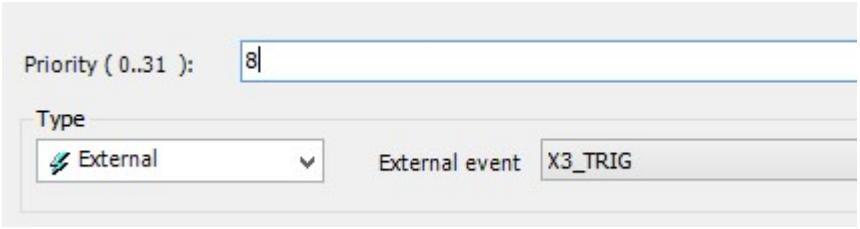

Add Library
xj
Library
Placeholder
Match
Library
XJ_WriteCSV
XJ_CSV, 1.0.0.0
XJ_OpenCSV
XJ_CSV, 1.0.0.0
XJ_ReadCSV
XJ_CSV, 1.0.0.0
XJ_Modbus, 1.0.0.0
XJ_ModbusTcpSlave
XJ_Modbus, 1.0.0.0
XJ_HSIO, 1.1.0.0
XJ_EnableInterrupt
XJ_HSIO, 1.1.0.0
XJ_MeasurePulseWidth
XJ_HSIO, 1.1.0.0
XJ_ResetCmpOutput
XJ_HSIO, 1.1.0.0
XJ_Counter_Reset
XJ_HSIO, 1.1.0.0
XJ_WriteInterruptParameter
XJ_HSIO, 1.1.0.0
Details...
Library Repository...
OK
Cancel

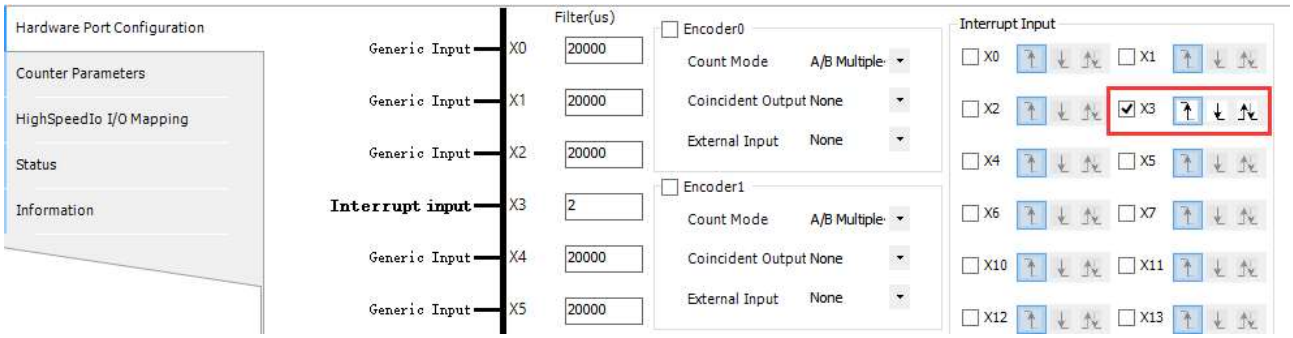
2-3-4. Application example

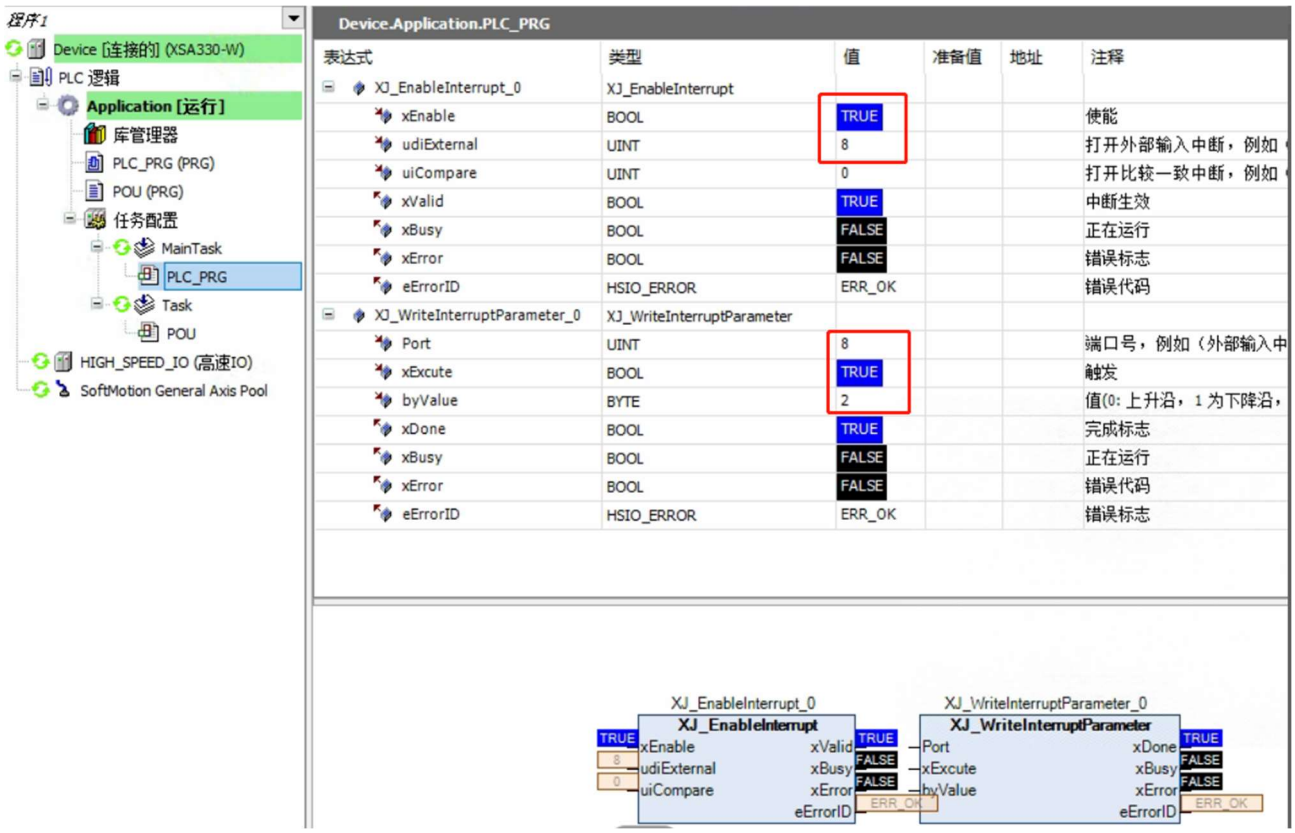
Double click “Task”, the type to "External" in the pop-up interface - use terminal X for external interrupts, and you can also set the priority of external interrupt events.

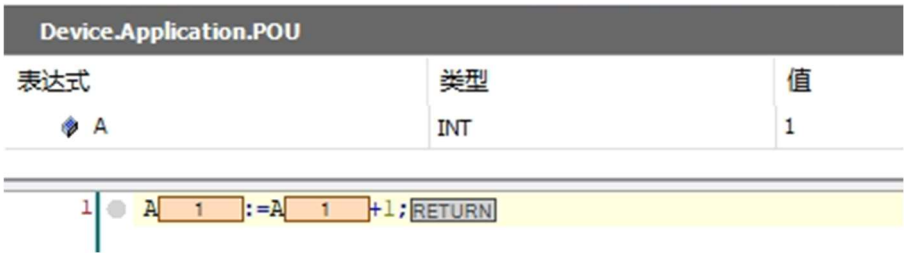


Example 1: Use the 【XJ_EnnableInterrupt】【XJ_WriteInterruptParameter】instructions. Set X3 as an external interrupt input, take its bilateral edge signal, which can be configured in the hardware parameter configuration interface or using XJ_WriteInterruptParameter instruction. Execute the self adding 1 instruction in the POU program under another task (configured as external, X3_TRIG) once the edge signal of X3 is given. The parameter configuration and instructions are shown in the following figure.






程序1
Device [连接的] (XSA330-W)
PLC 逻辑
Application [运行]
库管理器
PLC_PRG (PRG)
POU (PRG)
任务配置
MainTask
PLC_PRG
Task
POU
HIGH_SPEED_IO (高速IO)
SoftMotion General Axis Pool
Device.Application.PLC_PRG
表达式 类型 值 准备值 地址 注释
XJ_EnableInterrupt_0 XJ_EnableInterrupt
xEnable BOOL TRUE 使能
udiExternal UINT 8 打开外部输入中断，例如
uiCompare UINT 0 打开比较一致中断，例如
xValid BOOL TRUE 中断生效
xBusy BOOL FALSE 正在运行
xError BOOL FALSE 错误标志
eErrorID HSIO_ERROR ERR_OK 错误代码
XJ_WriteInterruptParameter_0 XJ_WriteInterruptParameter
Port UINT 8 端口号，例如（外部输入中
xExcute BOOL TRUE 触发
byValue BYTE 2 值(0: 上升沿，1 为下降沿，
xDone BOOL TRUE 完成标志
xBusy BOOL FALSE 正在运行
xError BOOL FALSE 错误代码
eErrorID HSIO_ERROR ERR_OK 错误标志
XJ_EnableInterrupt_0
XJ_EnableInterrupt
TRUE
8
0
xEnable
udiExternal
uiCompare
xValid
xBusy
xError
eErrorID
TRUE
FALSE
FALSE
ERR_OK
XJ_WriteInterruptParameter_0
XJ_WriteInterruptParameter
Port
xExcute
byValue
xDone
xBusy
xError
eErrorID
TRUE
FALSE
FALSE
ERR_OK

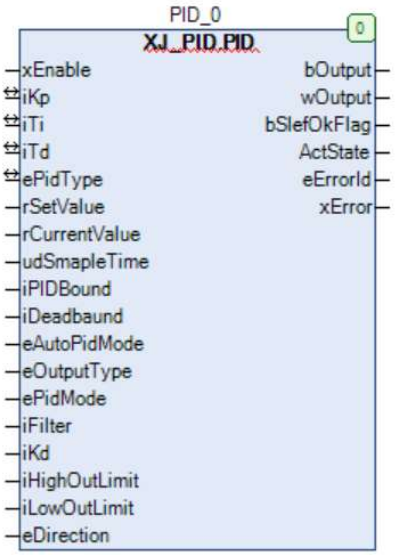

Device.Application.POU
表达式 类型 值
A INT 1
1 A[1]:=A[1]+1;RETURN

2-4. PID instructions

The PID function block is copyrighted and can only be used on Xinje's PLC! Simulation not available!

2-4-1. Command format

PID function [PID]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA model
Firmware	-	Software	-
Library	XJ_PID		

Command	Name	Graph	ST language
XJ_PID.PID	PID instruction		XJ_PID.PID(xEnable:= , iKp:= , iTi:= , iTd:= , ePidType:= , rSetValue:= , rCurrentValue:= , udSmapleTime:= , iPIDBound:= , iDeadbaund:= , eAutoPidMode:= , eOutputType:= , ePidMode:= , iFilter:= , iKd:= , iHighOutLimit:= , iLowOutLimit:= , eDirection:= , bOutput=> , wOutput=> , bSlefOkFlag=> , ActState=> , eErrorId=> , xError=>);

2-4-2. Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
xEnable	Enable	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
iKp	Proportional gain	INT	0~32767	0	Proportional gain
iTi	Integral time	INT	0~32767	0	Integral time (*100ms)

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
iTd	Differential time	INT	0~32767	0	Differential time (*10ms)
ePidType	Tuning mode	PIDTYPE	-	-	0: Manual mode; 1: Oscillation Self-tuning mode
rSetValue	Target value	REAL	Data range	0	*Target setting value
rCurrentValue	Feedback value	REAL	Data range	0	*Present feedback value
udSmapleTime	Sample time	UDINT	Data range	1000	* Sampling time (ms) Suggest setting between 1000 and 2000
iPIDBound	Operational range	INT	0~32767	0	PID operation range
iDeadbaund	Control Deadband	INT	0~32767	0	PID control dead zone
eAutoPidMode	Self-tuning PID control mode	PID CONTRLMODE	-	-	Self-tuning PID mode 0: PID control; 1: PI control; 2: P control
eOutputType	Output mode	OUTPUTTYPE	-	-	0: IO channel output; 1: Digital output
ePidMode	Control mode	PIDMODE	-	-	Control mode: general mode, advanced mode
iFilter	Input filtering constant	INT	0~100	0	Input filtering constant, available in advanced mode from 0 to 100
iKd	Differential gain	INT	-	50	Differential gain, default is 50, available in advanced mode
iHighOutLimit	Output upper limit setting value	INT	0~32767	0	Output upper limit setting value (default: 4095)
iLowOutLimit	Output lower limit setting value	INT	0~32767	0	Output lower limit setting value (default: 0)
eDirection	Reverse action/forward action	DIRECTIONPARAMETER	-	-	ATHWART: reverse action (heating); POSITIVE: Positive action (cooling)
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
bOutpu	Boolean output	BOOL	TRUE/FALSE	FALSE	IO output is TRUE
wOutput	Digital output	INT	TRUE/FALSE	FALSE	Digital output is true
bSlefOkFlag	Self-tuning completion flag	BOOL	TRUE/FALSE	FALSE	The completion of Self-tuning is TRUE
ActState	Present mode	AUTOPIDSTATE	-	-	Present mode
eErrorId	Error code	ERRID_PID	-	-	When normal, the value is 0. When an exception occurs, an error code is output
xError	Error flag	BOOL	TRUE/FALSE	FALSE	True when an exception occurs

2-4-3. Function description

- Direction of action

Positive action: The action where the output value MV of the operation increases with the increase of the measured value PV, usually used for cooling control.
Reverse action: An action in which the output value MV decreases as the measured value PV increases, usually used for heating control.

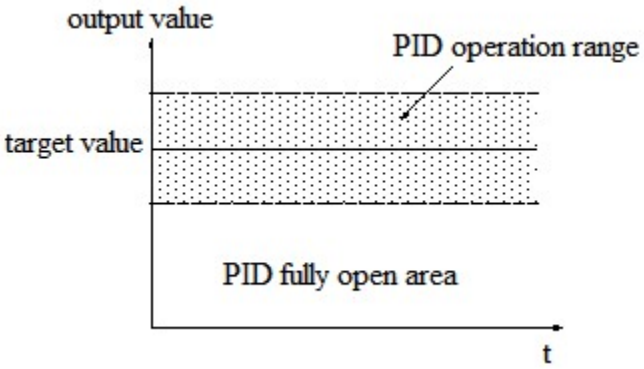
- Advanced mode setting: enable advanced mode, and users can set parameters such as filtering time, differential gain, and output upper and lower limits.

- Sample time

The system samples the current value at a certain time interval and compares it with the output value. This time interval is the sampling time T. When DA is output, T has no limit. When the port outputs, T must be greater than 1 PLC program scan cycle. The value of T should be within the range of 100-1000 PLC scanning cycles.

- PID operation range

When the system is running, it is initially in the PID fully open stage, which approaches the target value at the fastest speed (default is 4095). When it reaches the operating range of the PID, the parameters Kp, TI, and TD begin to take control. As shown in the following figure:



When the user is not clear about the specific set value of PID parameters, the Self-tuning mode can be selected to enable the system to automatically find the best control parameters (proportional gain Kp, integration time Ti, differential time TD).

- Control object applicable to Self-tuning mode: temperature and pressure. Unsuitable control objects: liquid level and flow rate, etc.

- Self-tuning is the process of extracting PID parameters. Sometimes Self-tuning can not find the best parameters once, and it requires multiple Self-tuning. Oscillation in the process is normal. After Self-tuning is completed and the best parameters are found, it is necessary to switch to manual PID. If the control object is unstable during the manual PID process and cannot be controlled at a constant target value, it may be caused by poor parameter adjustment, and it is necessary to adjust the PID parameters again to achieve stable control.

- At the beginning of Self-tuning in critical oscillation method, the user needs to set the PID control cycle (sampling time) in advance. Reference value: Generally, slow response systems can be set to 1000ms, while fast response systems can be set to 10ms-100ms.

- By using the critical oscillation method for Self-tuning, the system can start from any state. For temperature control objects, the current measured temperature does not need to be consistent with the ambient temperature. It can be below or above the target temperature.

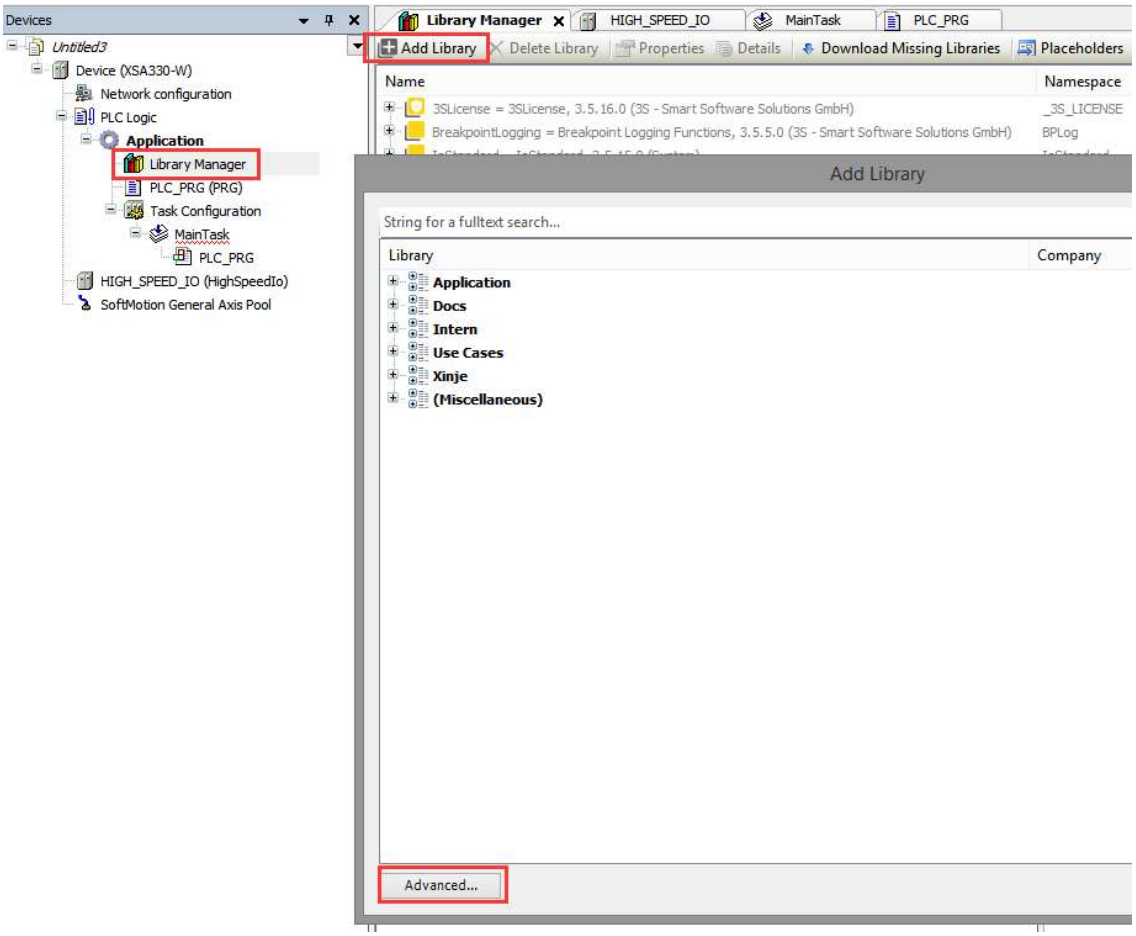
- For Self-tuning mode, it is necessary to set "ePidType" to TRUE, and select the pid mode (p, pi, pid). When the setting is completed, switch xEnable status to TRUE, PLC will enter the Self-tuning status, ActState status is AutoPidBusy, when "ActState" is AutoPidDone, it means Self-tuning is successful, and "bSlefOkFlag" is true. After maintaining a cycle, PID status will automatically switch to CommonPID, at this time, PID controls the control system in manual mode with Self-tuning parameters.

- Switching of output values. If the output value required by the user is a digital quantity, set the

eOutputType value to 1. If the output value required by the user is a switching quantity, set the eOutputType value to 0.

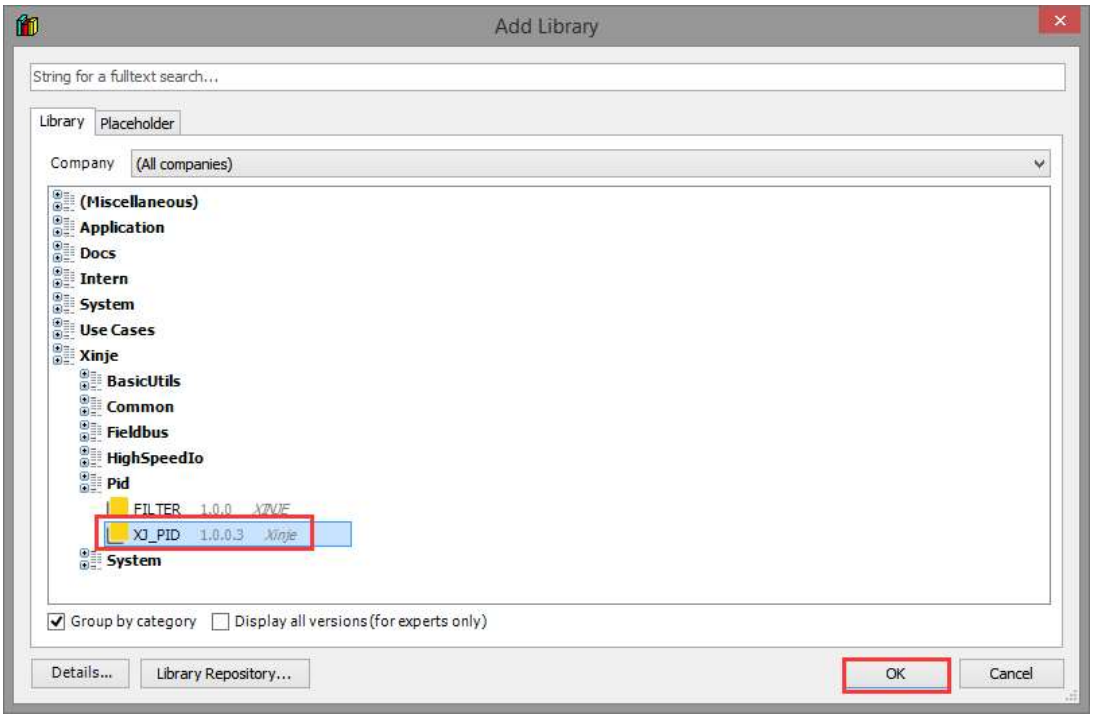
2-4-4. Application example

Double click Library manager → add library →advanced.



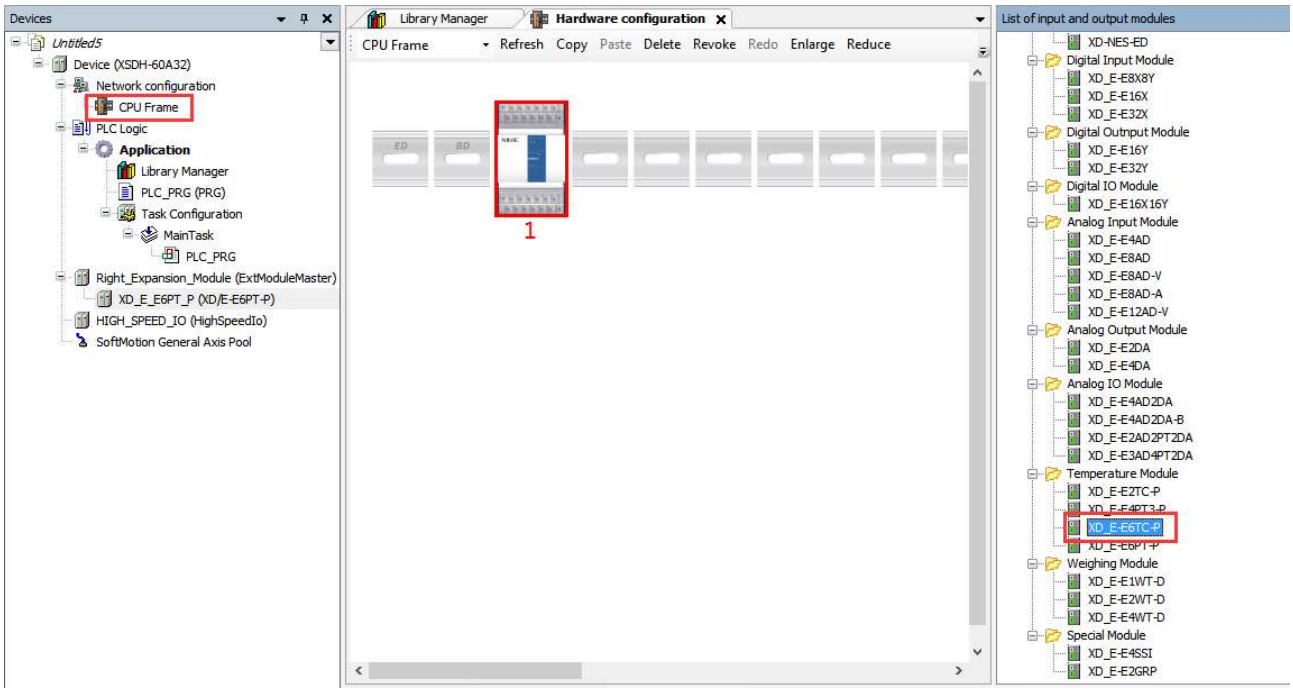
The PID function block is protected by copyright and can only be used on Xinje's PLC! Simulation not available!

Add XJ_PID.

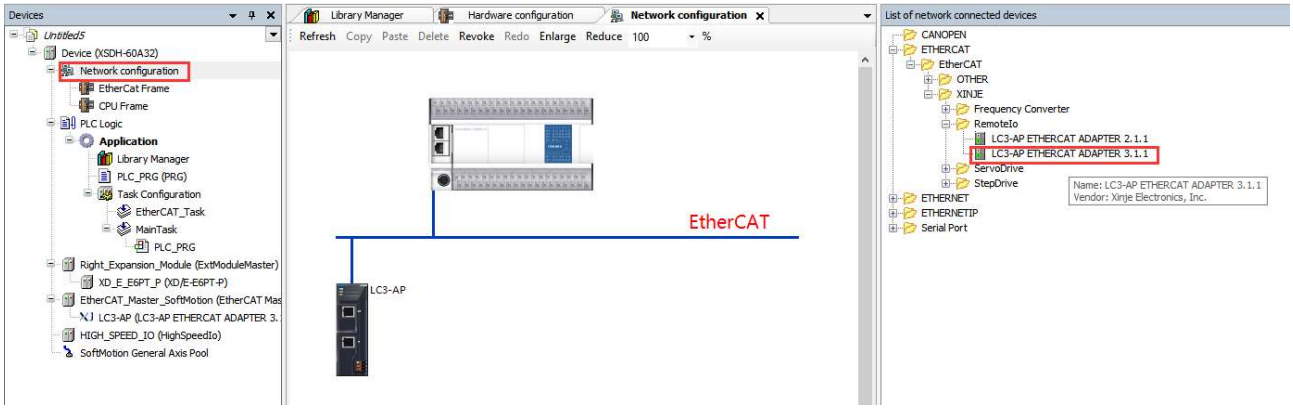


Add temperature expansion module.

- Double click CPU Frame, select position 1, then double click XD-E6TC-P.

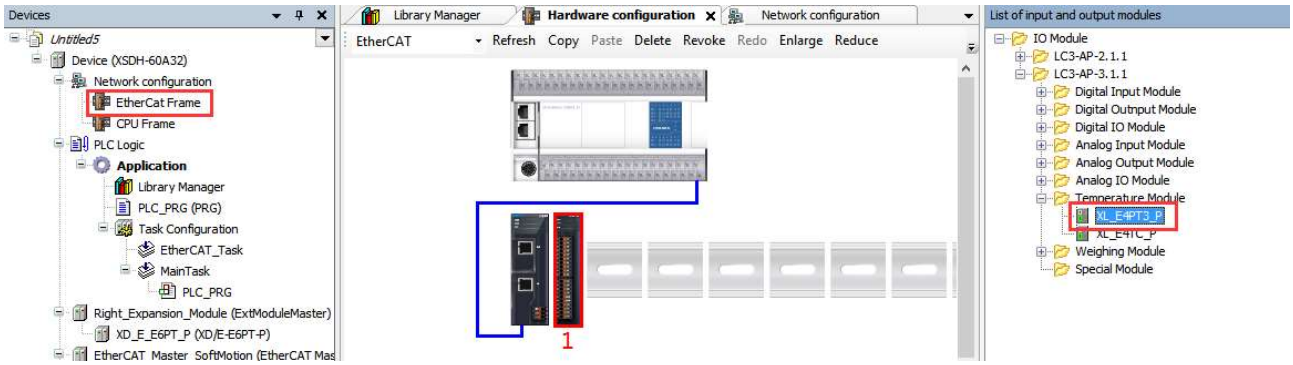


- Remote IO module: double click Network configuration, double click LC3-AP Ethercat adapter 3.1.1.

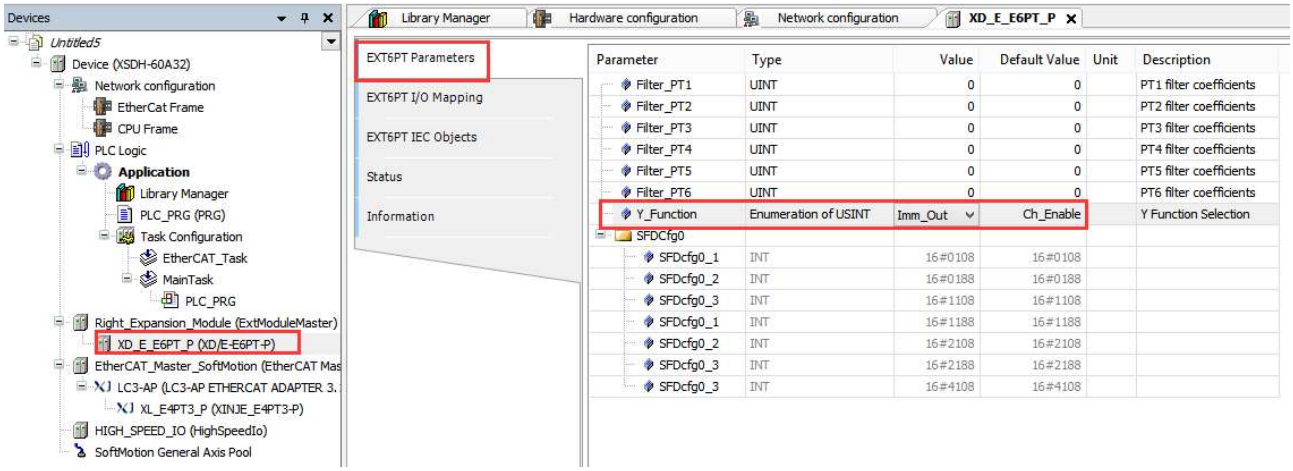


Note: Currently, only LC3-AP V3.1 and above support EC_From TO instruction.

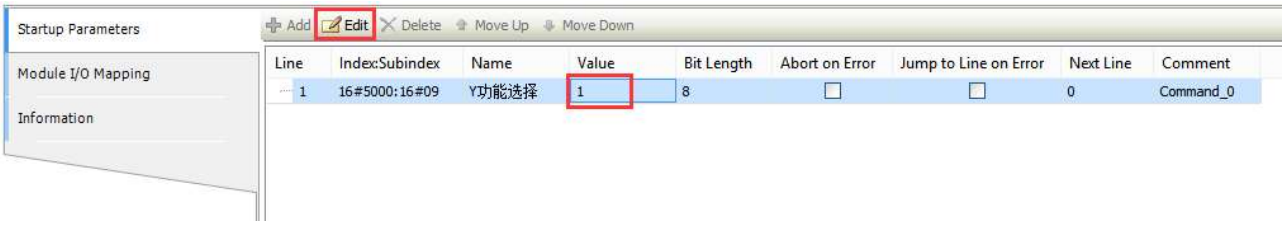
Double click EtherCat Frame, select position 1, double click XL-E4PT3-P.



If a switch quantity control relay is needed, the module on the main body needs to set the Y_Function to 'Imm Out'.

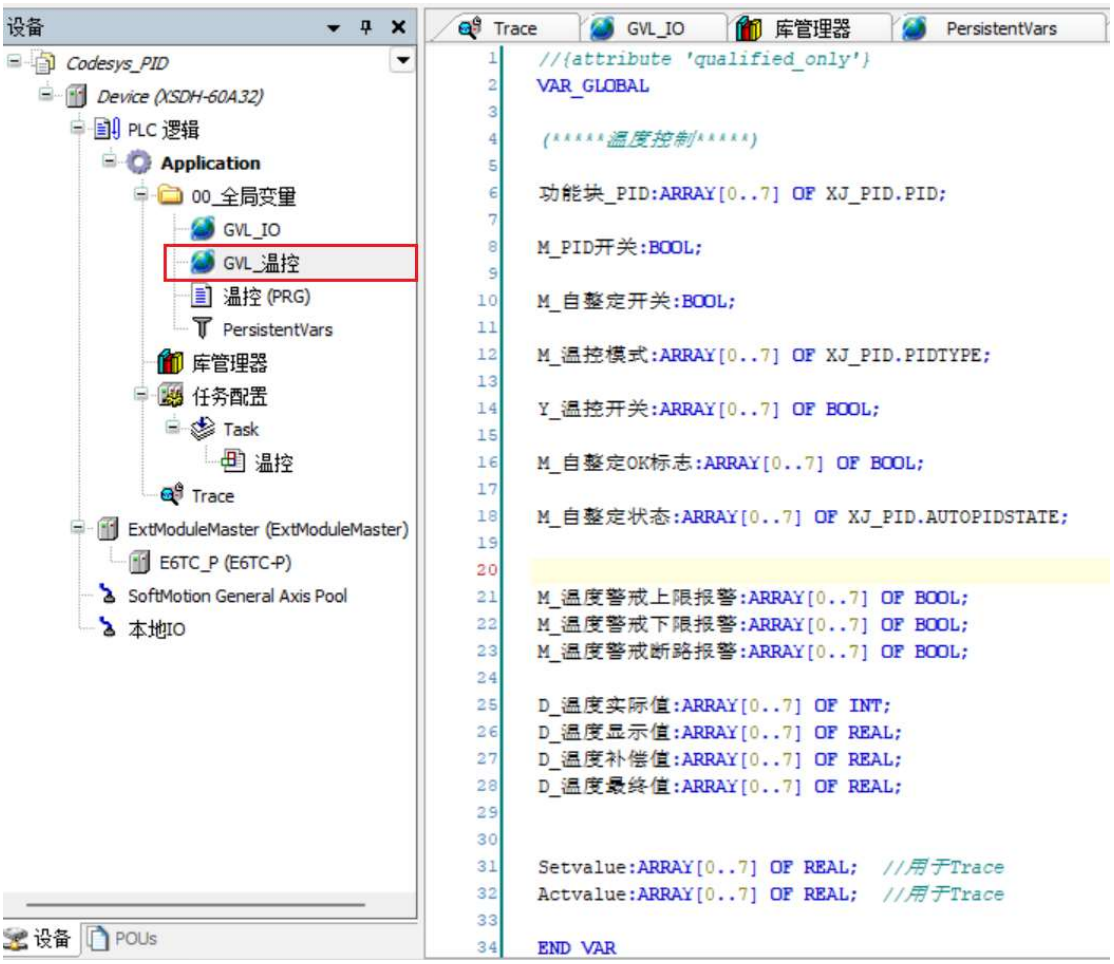


The module on remote IO needs to add the corresponding startup parameter to the default value of 1.

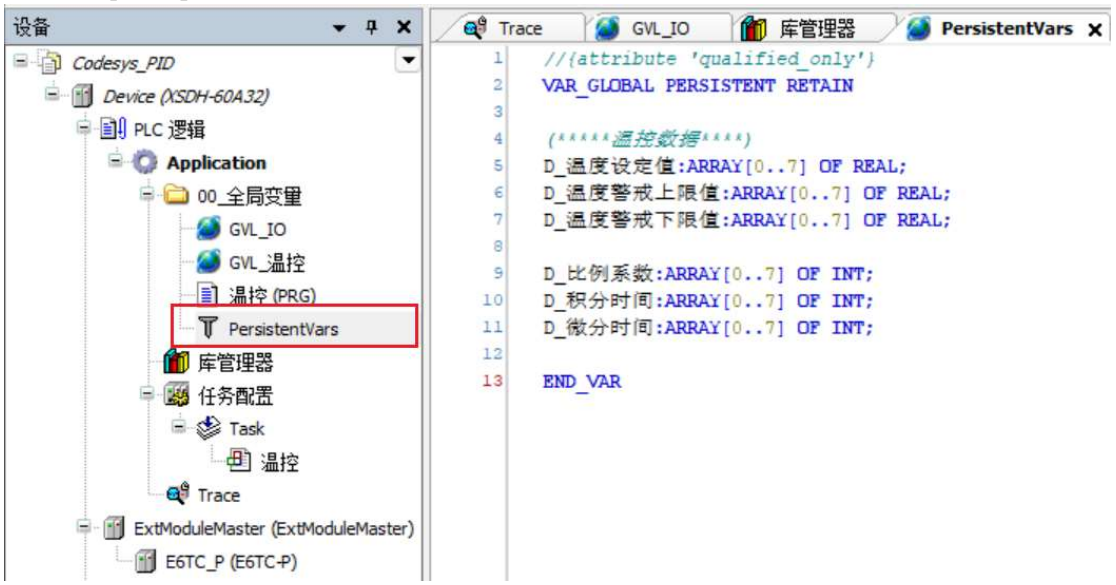


After modifying the module, the parameters need to be powered on again to take effect.

Define the required Global variable.



8. Define the required power-off retention variables so that data will not be lost after the PLC restarts.



9. Make the PID program.

① Convert the actual temperature value obtained from the temperature module into the correct temperature display value. For example, if the transmitted data is 289, it needs to be divided by 10 to convert it into a Floating-point number of 28.9, representing 28.9°, which is convenient for subsequent personnel to understand and maintain. Due to the possibility of different on-site conditions requiring additional compensation, an additional compensation value is added to obtain a final value. Use this final value to adjust the PID.

```
//注：在手动模式下，写入参数需要将PID功能块使能关掉。

(*****温度实际值转换*****)
FOR a:=0 TO 2 BY 1 DO

    D_温度显示值[a] :=TO_REAL(D_温度实际值[a]) / 10;
    D_温度最终值[a] := D_温度显示值[a] + D_温度补偿值[a];

    Setvalue[a]:=D_温度设定值[a];  //用于Trace采集
    Actvalue[a]:=D_温度最终值[a];  //用于Trace采集

END_FOR
```

② Temperature alarm: When the actual temperature is too high or too low, as well as when there is an open circuit, an alarm is required.

```
(*****温度超限报警*****)
FOR b:=0 TO 2 BY 1 DO
    //温度低下限报警
    IF D_温度最终值[b] < D_温度警戒下限值[b] THEN
        M_温度警戒下限报警[b]:= TRUE;
    ELSE
        M_温度警戒下限报警[b]:= FALSE;
    END_IF

    //温度超上限报警
    IF D_温度最终值[b] > D_温度警戒上限值[b] THEN
        M_温度警戒上限报警[b]:= TRUE;
    ELSE
        M_温度警戒上限报警[b]:= FALSE;
    END_IF

    //温度断路报警
    IF D_温度显示值[b] = -0.1 THEN
        M_温度警戒断路报警[b]:= TRUE;
    ELSE
        M_温度警戒断路报警[b]:= FALSE;
    END_IF

END_FOR
```

③ One key Self-tuning, when the "M_ Self-tuning switch" is on (it can be instantaneous, and it is forbidden to be always on), all PID function blocks are in Self-tuning mode, which is convenient for batch debugging.

```
(*****启用自整定模式*****)
IF M_自整定开关 THEN
    M_温控模式[0]:= 1;
    M_温控模式[1]:= 1;
    M_温控模式[2]:= 1;
END_IF
```

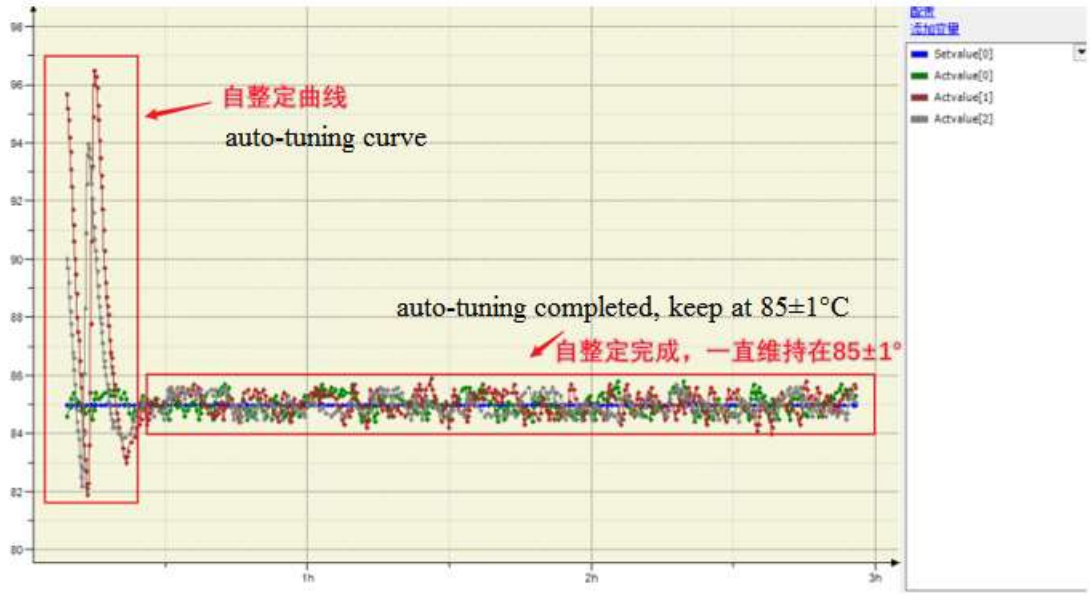
④ To enable Enable the PID function block, simply set ON M_PID. (Note: When modifying PID parameters, be sure to turn off the PID enable!)

```
IF NOT M_温度警戒上限报警[0] THEN
  功能块_PID[0](
    xEnable:= M_PID开关,  //启用PID功能块
    iKp:= D_比例系数[0],  //比例系数，范围0~32767，单位%
    iTi:= D_积分时间[0],  //积分时间，范围0~32767，单位100ms
    iTd:= D_微分时间[0],  //微分时间，范围0~32767,单位10ms
    rSetValue:=  D_温度设定值[0],  //要维持的温度值
    rCurrentValue:= D_温度最终值[0],  //温度传感器检测到的实际值
    ePidType:= M_温控模式[0], //0:手动模式；1：自整定模式
    udSmapleTime:= 1000,  //采样时间，建议设置在1000~2000
    iPIDBound:= ,  //PID运算范围，0~32767
    iDeadbaund:= ,  //PID控制死区，0~32767
    eAutoPidMode:= 0,//自整定模式，0：PID控制；1：PI控制；2：P控制
    eOutputType:= XJ_PID.OUTPUTTYPE.BoolType,  //输出模式，开关量或模拟量
    ePidMode:= 0,  //控制模式：通用模式，高级模式
    iFilter:= ,  //采样温度滤波参数，0~100，单位%，高级模式模式下可用
    iKd:= ,  //微分增益，默认50，高级模式模式下可用
    iHighOutLimit:= ,  //输出上限设定值，0~32767，高级模式模式下可用
    iLowOutLimit:= ,  //输出下限设定值，0~32767，高级模式模式下可用
    eDirection:= 0,  //0：逆动作（加热）；1：正动作（冷却）
    bOutput=> Y_温控开关[0],  //开关量输出
    wOutput=> ,  //模拟量输出
    bSlefOkFlag=> M_自整定OK标志[0] ,
    ActState=> M_自整定状态[0],  //自整定状态
    eErrorId=> ,
    xError=>  );
END_IF
```

⑤ The PID Self-tuning time is about 15~25 minutes. After the Self-tuning is completed, iKp, iTi and iTd will all have values. ePidype will automatically switch to "ManualType", the bSelfOKFlag flag will be on, and ActState will display "AutoPIDrea" to facilitate the next Self-tuning.

```
(*****PID*****)
PID[0](
    xEnable TRUE := M_PID开关 TRUE,   //启用PID功能块
    iKp 31929 := D_比例系数[0] 31929,   //比例系数，范围0~32767，单位%
    iTi 1636 := D_积分时间[0] 1636,   //积分时间，范围0~32767，单位100ms
    iTd 4091 := D_微分时间[0] 4091,   //微分时间，范围0~32767,单位10ms
    rSetValue 85 := D_温度设定值[0] 85,   //要维持的温度值
    rCurrentValue 91.2 := D_温度最终值[0] 91.2,   //传感器检测到的实际值
    ePidType 0 := M_温控模式[0] ManulType, //0:手动模式; 1: 自整定模式
    udSmapleTime 1000 := 1000,   //采样时间
    iPIDBound:= ,   //PID运算范围，0~32767
    iDeadbaund:= ,   //PID控制死区，0~32767
    eAutoPidMode PIDMode := 0,//自整定模式，0：PID控制；1：PI控制；2：P控制
    eOutputType BoolType := XJ_PID.OUTPUTTYPE.BoolType,   //输出模式，开关量或模拟量
    ePidMode Common := 0,   //控制模式：通用模式，高级模式
    iFilter:= ,   //采样温度滤波参数，0~100，单位%，高级模式模式下可用
    iKd:= ,   //微分增益，默认50，高级模式模式下可用
    iHighOutLimit:= ,   //输出上限设定值，0~32767，高级模式模式下可用
    iLowOutLimit:= ,   //输出下限设定值，0~32767，高级模式模式下可用
    eDirection Athwart := 0,   //0：逆动作（加热）；1：正动作（冷却）
    bOutput FALSE => Y_温控开关[0] FALSE,   //开关量输出
    wOutput=> ,   //模拟量输出
    bSlefOkFlag TRUE => M_自整定OK标志[0] TRUE ,
    ActState AutoPIDRea => M_自整定状态[0] AutoPIDRea,   //自整定状态
    eErrorId=> ,
    xError=>  ,
    PreI_Item 0 => PreI_Item[0] 0,
    PreD_Item 293 => PreD_Item[0] 293,
    PrePID_Item 0 => PrePID_Item[0] 0,
    Overshoot 7 => Overshoot[0] 7)
;
```

10. The PID accuracy is around 2% of the set temperature. After actual measurement, the set temperature is 85 °C. With PID adjustment, the temperature has been maintained at 85±1 °C



2-5. System library

2-5-1. Function overview

Users can read or write system parameters of the controller through instructions. Currently only XSA330-W supports this function.

2-5-2. Function block introduction

2-5-2-1. CPU dominant frequency【XJ_GetCPUFrequency】

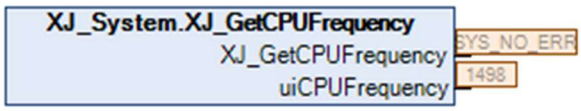
CPU main frequency [XJ_GetCPUFrequency]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetCPUFrequency	CPU dominant frequency		`XJ_GetCPUFrequency(uiCPUFrequency=> );`

(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetCPUFrequency	Error code	SYS_XJ_ERR	-	0	Error code
CPUFrequency	CPU dominant frequency	UINT	-	0	Unit is M

The execution results are as follows:



2-5-2-2. CPU temperature【XJ_GetCPUTemperature】

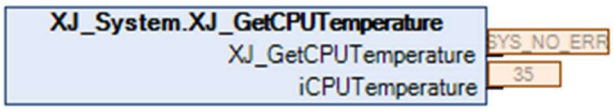
CPU temperature [XJ_GetCPUTemperature]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetCPUTemperature	CPU temperature		`XJ_GetCPUTemperature(iCPUTemperature=> )`

(1) Output variables

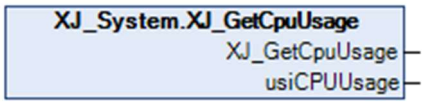
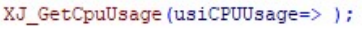
Output variables	Name	Data type	Effective range	Initial value	Description
GetCPUTemperature	Error code	SYS_XJ_ERR	-	0	Error code
CPUTemperature	CPU temperature	INT	-	0	CPU temperature

The execution results are as follows:



2-5-2-3. CPU occupancy【XJ_GetCPUUsage】

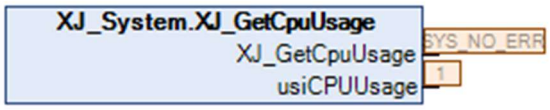
CPU usage [XJ_GetCPUUsage]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetCPUUsage	CPU occupancy		

(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetCPUUsage	Error code	SYS_XJ_ERR	-	0	Error code
CPUUsage	CPU occupancy	USINT	-	0	Return percentage, 0~100

The execution results are as follows:



2-5-2-4. Memory size【XJ_GetMemSize】

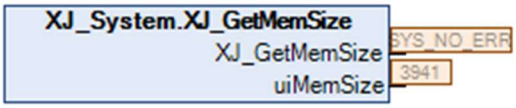
Memory size [XJ_GetMemSize]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetMemSize	Memory size	XJ_System.XJ_GetMemSize XJ_GetMemSize uiMemSize	XJ_GetMemSize(uiMemSize=>);

(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetMemSize	Error code	SYS_XJ_ERR	-	0	Error code
MemSize	Memory size	UINT	-	0	Unit is MB

The execution results are as follows:



2-5-2-5. Memory occupancy【XJ_GetMemUsage】

Memory occupancy [XJ_GetMemUsage			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetMemUsage	Memory occupancy	XJ_System.XJ_GetMemUsage XJ_GetMemUsage usiMemUsage	XJ_GetMemUsage(usiMemUsage=>);

(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetMemUsage	Error code	SYS_XJ_ERR	-	0	Error code
MemUsage	Memory occupancy	USINT	-	0	Return percentage, 0~100

The execution results are as follows:

2-5-2-6. On time【XJ_GetBootTime】

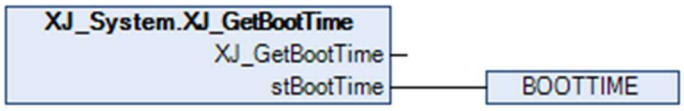
On time [XJ_GetBootTime]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetBootTime	On time	XJ_System.XJ_GetBootTime XJ_GetBootTime stBootTime	XJ_GetBootTime(stBootTime=>);

(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetBootTime	Error code	SYS_XJ_ERR	-	0	Error code
BootTime	On time	SYS_XJ_TIME	-	0	structural morphology

The execution results are as follows:




```
PROGRAM POU
VAR
    // 获取网口信息
    XJ_GetNetInfo_0: XJ_System.XJ_GetNetInfo;
    PLCID: DWORD;
    BOOTTIME: XJ_SYSTEM.SYS_XJ_TIME;
    AA: BOOL;
    NETID: BYTE;
    XJ_SetNetInfo_0: XJ_System.XJ_SetNetInfo;
    runtimeversion: DWORD;
    STTIME: XJ_SYSTEM.SYS_XJ_TIME;
    timezone: INT;
    SetTIME: XJ_System.SYS_XJ_TIME;
    itimezone: INT;
END_VAR
```

BOOTTIME	XJ_SYSTEM.SYS_XJ_TIME				
Year	UINT	2022			年
Month	UINT	11			月
Day	UINT	19			日
Hour	UINT	1			时
Minute	UINT	55			分
Second	UINT	43			秒
Milliseconds	UINT	0			微妙
DayOfWeek	UINT	6			周

2-5-2-7. PLC name【XJ_GetPLCName】

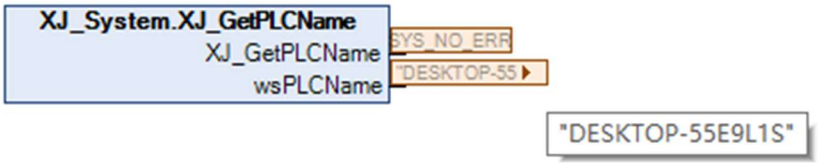
PLC name [XJ_GetPLCName]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetPLCName	PLC name	XJ_System.XJ_GetPLCName XJ_GetPLCName wsPLCName	XJ_GetPLCName(wsPLCName=>);

(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetPLCName	Error code	SYS_XJ_ERR	-	0	Error code
PLCName	PLC name	WSTRING	-	0	PLC name

The execution results are as follows:



2-5-2-8. PLCID【XJ_GetPLCID】

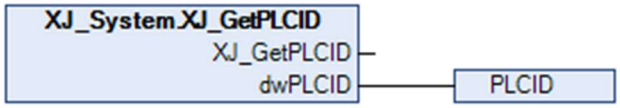
PLCID [XJ_GetPLCID]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetPLCID	PLCID	XJ_System.XJ_GetPLCID XJ_GetPLCID dwPLCID	XJ_GetPLCID(dwPLCID=>);

(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetPLCID	Error code	SYS_XJ_ERR	-	0	Error code
PLCID	PLCID	DWORD	-	0	PLCID information, hexadecimal display

The execution result is as follows, and the PLCID information is displayed in hexadecimal:



```
PROGRAM POU
VAR
    // 获取网口信息
    XJ_GetNetInfo_0: XJ_System.XJ_GetNetInfo;
    PLCID: DWORD;
    BOOTTIME: XJ_SYSTEM.SYS_XJ_TIME;
    AA: BOOL;
    NETID: BYTE;
    XJ_SetNetInfo_0: XJ_System.XJ_SetNetInfo;
    runtimeversion: DWORD;
    STTIME: XJ_SYSTEM.SYS_XJ_TIME;
    timezone: INT;
    SetTIME: XJ_System.SYS_XJ_TIME;
    itimezone: INT;
END_VAR
```

表达式	类型	值	准
XJ_GetNetInfo_0	XJ_System.XJ_GetNetInfo		
PLCID	DWORD	16#17070200	
BOOTTIME	XJ_SYSTEM.SYS_XJ_TIME		
AA	BOOL	FALSE	
NETID	BYTE	16#00	
XJ_SetNetInfo_0	XJ_System.XJ_SetNetInfo		
runtimeversion	DWORD	16#03050F28	
STTIME	XJ_SYSTEM.SYS_XJ_TIME		
timezone	INT	16#0000	
SetTIME	XJ_System.SYS_XJ_TIME		
itimezone	INT	16#0000	

2-5-2-9. Firmware version【XJ_GetPLCVersion】

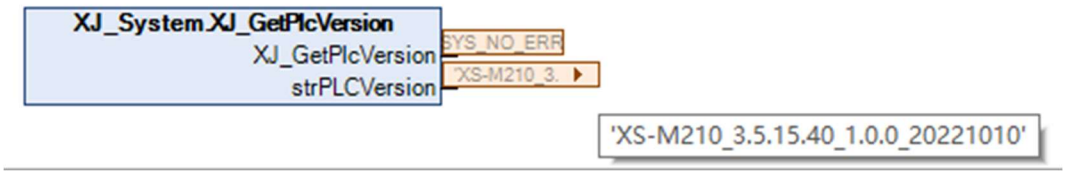
Firmware version [XJ_GetPLCVersion]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetPLCVersion	PLC firmware version	XJ_System.XJ_GetPlcVersion XJ_GetPlcVersion strPLCVersion	XJ_GetPlcVersion(strPLCVersion=>);

(1) Output variables

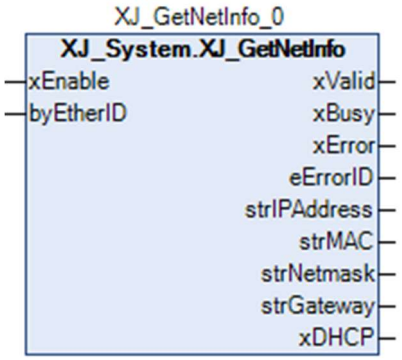
Output variables	Name	Data type	Effective range	Initial value	Description
GetPLCVersion	Error code	SYS_XJ_ERR	-	0	Error code
PLCVersion	PLC firmware version	STRING	-	0	PLC firmware version

The execution result is as follows:



2-5-2-10. Obtain network port information【XJ_GetNetInfo】

Obtain network port information [XJ_GetNetInfo]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetNetInfo	Obtain network port information		

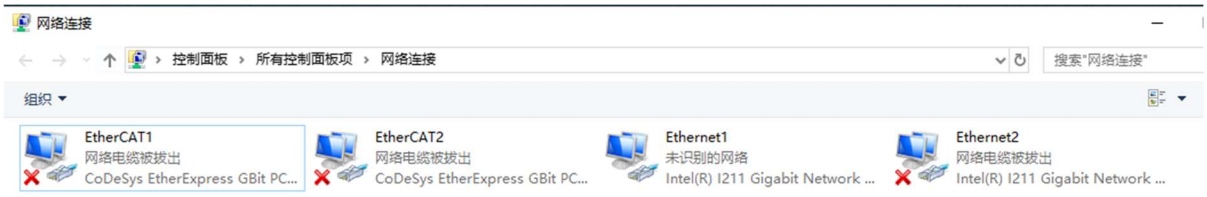
(1) Input variables

Input variables	Name	Data type	Effective range	Initial value	Description
Enable	Enable	BOOL	TRUE,FALSE	FALSE	Enable
EtherID	Network port ID	BYTE	-	0	Corresponding network port no. on the display screen

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
Valid	Output valid	BOOL	TRUE,FALSE	FALSE	Output valid
Busy	Executing	BOOL	TRUE,FALSE	FALSE	Executing
Error	Error flag	BOOL	TRUE,FALSE	FALSE	Error flag
ErrorID	Error code	SYS_XJ_ERR	-	0	Error code
IPAddress	IP address	STRING	-	0	IP address
MAC	MAC address	STRING	-	0	MAC address
Netmask	Subnet address	STRING	-	0	Subnet address
Gateway	Gateway Information	STRING	-	0	Gateway Information
DHCP	DHCP properties	BOOL	TRUE，FALSE	FALSE	Is it automatic property acquisition

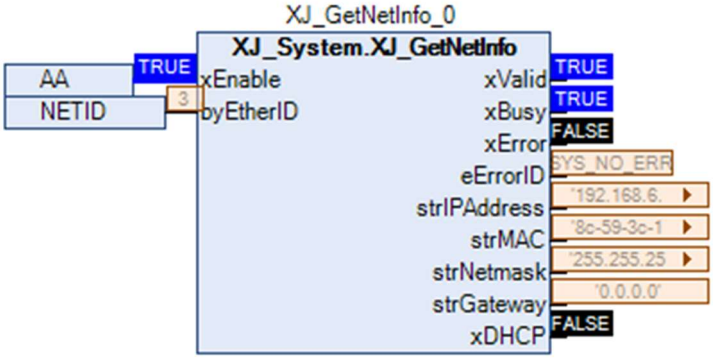
The network port IDs of the industrial computer are arranged in 0-3 order from left to right in the network adapter, as shown in the figure Ethernet1, and the IDs are written in 2.



The definition of XSA-L, also known as the XSA industrial computer network port ID for Linux systems, is that the network port ID near the DP port is 0, increasing in order of 0, 1, 2, 3, 4, and 5.

```
PROGRAM POU
VAR
    // 获取网口信息
    XJ_GetNetInfo_0: XJ_System.XJ_GetNetInfo;
    PLCID: DWORD;
    BOOTTIME: XJ_SYSTEM.SYS_XJ_TIME;
    AA: BOOL;
    NETID: BYTE;
END_VAR
```

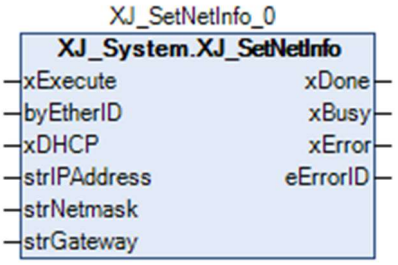
The execution result is as follows：



XJ_GetNetInfo_0	XJ_System.XJ_GetNetInfo			获取网口信息
xEnable	BOOL	TRUE		功能块使能
byEtherID	BYTE	3		网口 Id
xValid	BOOL	TRUE		输出有效
xBusy	BOOL	TRUE		正在执行
xError	BOOL	FALSE		错误
eErrorID	SYS_XJ_ERR	SYS_NO_ERR		错误 ID
strIPAddress	STRING(17)	'192.168.6.6'		IP地址
strMAC	STRING(17)	'8c-59-3c-10-a9-...		MAC地址
strNetmask	STRING(17)	'255.255.255.0'		子网地址
strGateway	STRING(17)	'0.0.0.0'		网关信息
xDHCP	BOOL	FALSE		是否为自动获取属性

2-5-2-11. Set network port information【XJ_SetNetInfo】

Set network port information [XJ_SetNetInfo]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_SetNetInfo	Set network port information		XJ_SetNetInfo_0(xExecute:= , byEtherID:= , xDHCP:= , strIPAddress:= , strNetmask:= , strGateway:= , xDone=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

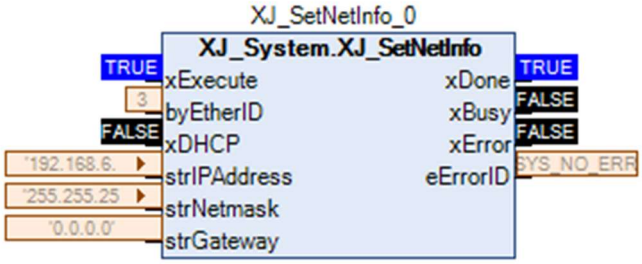
Input variables	Name	Data type	Effective range	Initial value	Description
Excute	Excute	BOOL	TRUE,FALSE	FALSE	Execute the function block at the rising edge
EtherID	Network port ID	BYTE	-	0	Corresponding network port on the display screen
DPCH	DPCH property	BOOL	TRUE,FALSE	FALSE	Is it automatic property acquisition
IPAddress	IP address	STRING	-	0	IP address
Netmask	Subnet address	STRING	-	0	Subnet address
Gateway	Gateway information	STRING	-	0	Gateway information

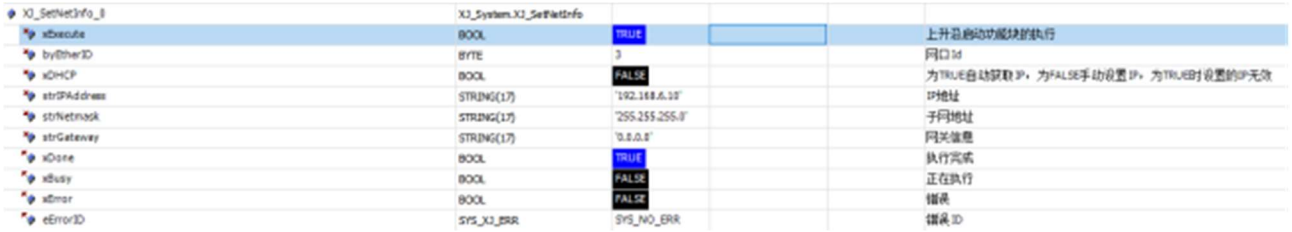
(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
Done	Execution completed	BOOL	TRUE,FALSE	FALSE	Execution completed
Busy	Executing	BOOL	TRUE,FALSE	FALSE	Executing
Error	Error flag	BOOL	TRUE,FALSE	FALSE	Error flag
ErrorID	Error code	SYS_XJ_ERR	-	0	Error code

The modification will only take effect after powering off and restarting the PLC or RTE.

The definition of XSA-L, also known as the XSA industrial computer network port ID for Linux systems, is that the network port ID near the DP port is 0, increasing in order of 0, 1, 2, 3, 4, and 5.





2-5-2-12. Runtime version【XJ_GetRuntimeVersion】

Runtime version [XJ_GetRuntimeVersion]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetRuntimeVersion	Runtime version	XJ_System.XJ_GetRuntimeVersion XJ_GetRuntimeVersion dwRuntimeVersion	XJ_GetRuntimeVersion(dwRuntimeVersion=>);

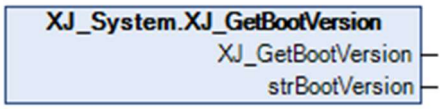
(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetRuntimeVersion	Error code	SYS_XJ_ERR	-	0	Error code
RuntimeVersion	Runtim version	DWORD	-	0	Runtim version

The runtime version is viewed in hexadecimal.

2-5-2-13. BOOT version【XJ_GetBootVersion】

BOOT version [XJ_GetBootVersion]			
Execution condition	-	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetBootVersion	BOOT version		XJ_GetBootVersion(strBootVersion=>);

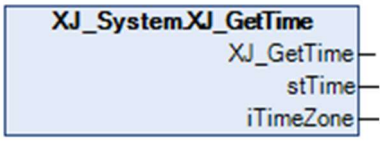
(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetBootVersion	Error code	SYS_XJ_ERR	-	0	Error code
BootVersion	BOOT version	STRNG	-	0	BOOT version

The execution results are as follows:

2-5-2-14. Obtain date and time【XJ_GetTime】

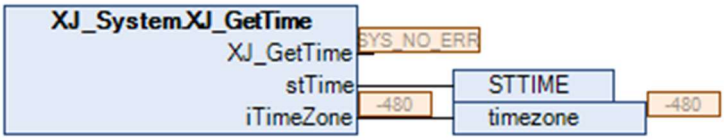
Obtain date and time [XJ_GetTime]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_GetTime	Obtain date and time		XJ_GetTime(stTime=> , iTimeZone=>)

(1) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description
GetTime	Error code	SYS_XJ_ERR	-	0	Error code
Time	Time	SYS_XJ_TIME	-	0	Structural morphology
TimeZone	Get time zone	INT	-	-	Unit is minute

The execution results are as follows:




```
PROGRAM POU
VAR
    // 获取网口信息
    XJ_GetNetInfo_0: XJ_System.XJ_GetNetInfo;
    PLCID: DWORD;
    BOOTTIME: XJ_SYSTEM.SYS_XJ_TIME;
    AA: BOOL;
    NETID: BYTE;
    XJ_SetNetInfo_0: XJ_System.XJ_SetNetInfo;
    runtimeversion: DWORD;
    STTIME: XJ_SYSTEM.SYS_XJ_TIME;
    timezone: INT;
    SetTIME: XJ_System.SYS_XJ_TIME;
    itimezone: INT;
END_VAR
```

STTIME	XJ_SYSTEM.SYS_XJ_TIME				
Year	UINT	2022			年
Month	UINT	11			月
Day	UINT	18			日
Hour	UINT	13			时
Minute	UINT	9			分
Second	UINT	8			秒
Milliseconds	UINT	909			微秒
DayOfWeek	UINT	5			周

2-5-2-15. Set date and time【XJ_SetTime】

Set date and time [XJ_SetTime]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	V1.1.0 and up	Software	-
Library	XJ_System		

Command	Name	Graph representation	ST language
XJ_SetTime	Set date and time	XJ_System.XJ_SetTime stTime iTimeZone XJ_SetTime	XJ_SetTime(stTime:= , iTimeZone:=);

(1) Input variables

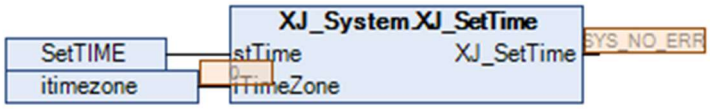
Input variables	Name	Data type	Effective range	Initial value	Description
Time	Set time	SYS_XJ_TIME	-	0	Set time
TimeZone	Set time zone	INT	-	-480	Unit is minute

(2) Output variables

Output variables	Name	Data type	Effective range	Initial value	Description

SetTime	Error code	SYS_XJ_ERR	-	0	Error code

The change in time takes effect immediately after writing. Due to the lack of conduction conditions, the time remains unchanged after writing the time. The execution result is as follows:




```
PROGRAM POU
VAR
    // 获取网口信息
    XJ_GetNetInfo_0: XJ_System.XJ_GetNetInfo;
    PLCID: DWORD;
    BOOTTIME: XJ_SYSTEM.SYS_XJ_TIME;
    AA: BOOL;
    NETID: BYTE;
    XJ_SetNetInfo_0: XJ_System.XJ_SetNetInfo;
    STTIME: XJ_SYSTEM.SYS_XJ_TIME;
    runtimeversion: DWORD;
    timezone: INT;
    SetTIME: XJ_System.SYS_XJ_TIME;
    itimezone: INT;
END_VAR
```

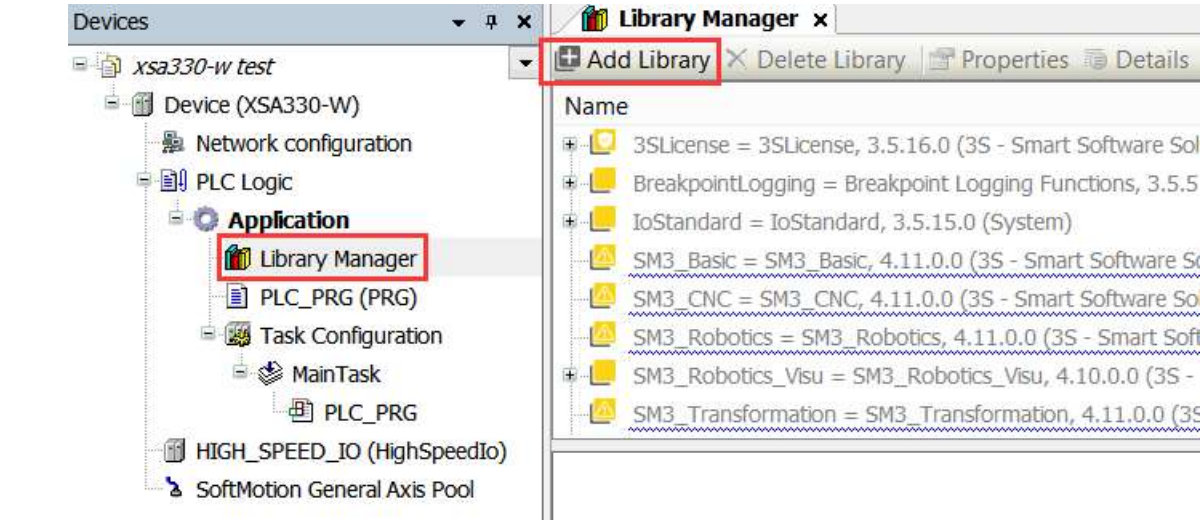
SetTIME	XJ_System.SYS_XJ_TIME				
Year	UINT	2022			年
Month	UINT	11			月
Day	UINT	19			日
Hour	UINT	13			时
Minute	UINT	30			分
Second	UINT	10			秒
Milliseconds	UINT	855			微妙
DayOfWeek	UINT	5			周

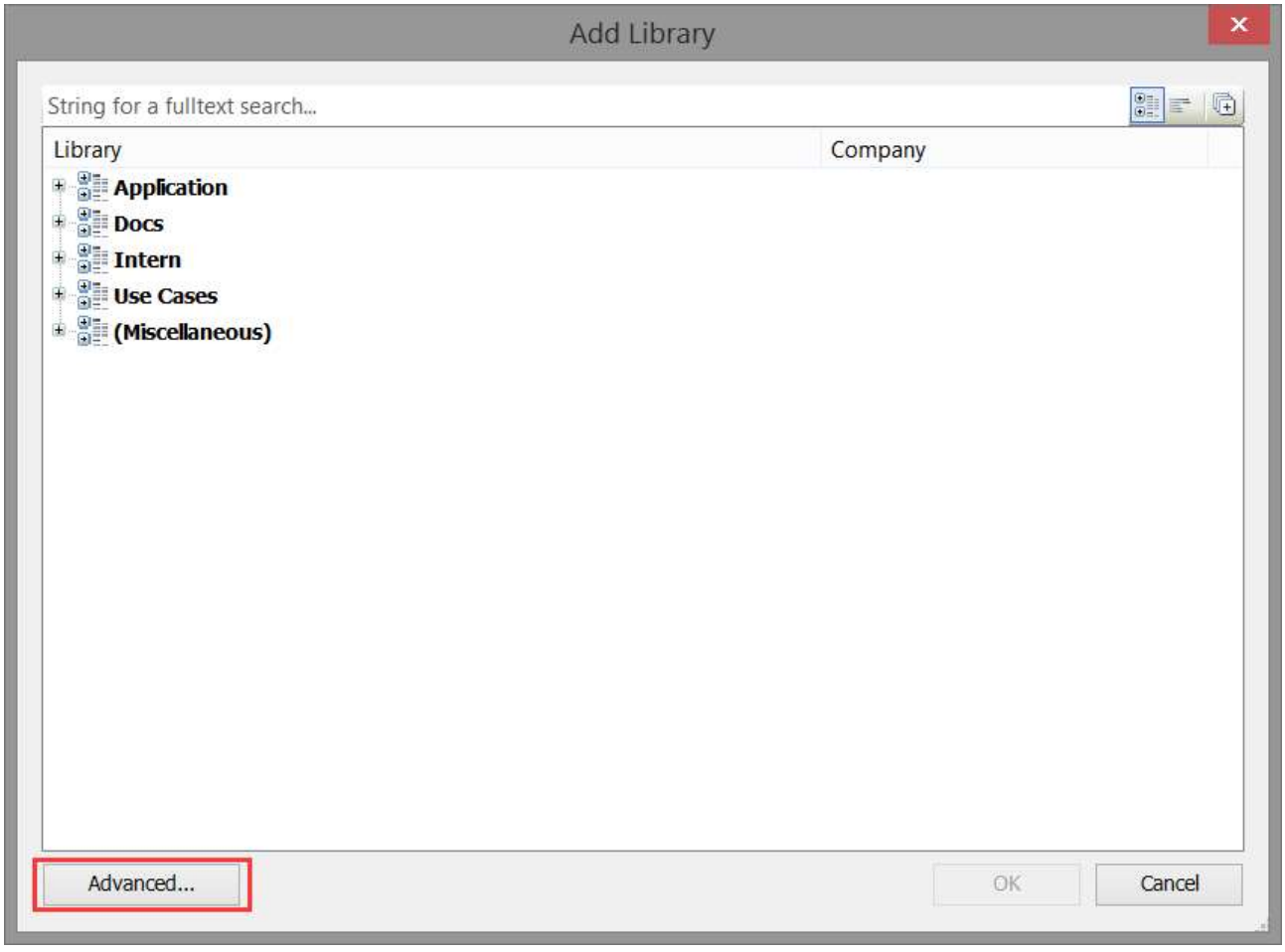
The obtained time is consistent with the set time.

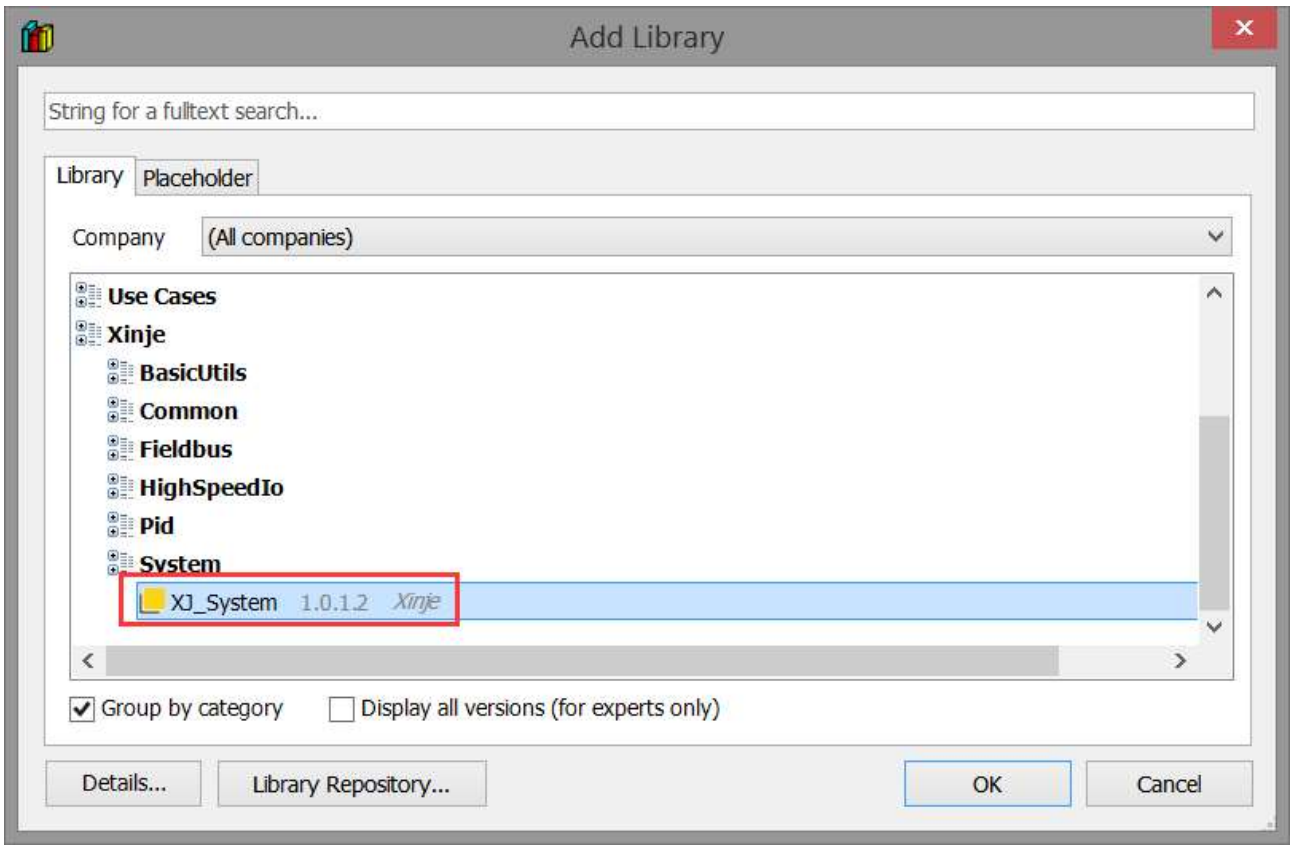
STTIME	XJ_SYSTEM.SYS_XJ_TIME				
Year	UINT	2022			年
Month	UINT	11			月
Day	UINT	19			日
Hour	UINT	13			时
Minute	UINT	30			分
Second	UINT	11			秒
Milliseconds	UINT	938			微妙
DayOfWeek	UINT	6			周

2-5-3. Parameter configuration

Add Library File: Library Manager - Add Library - Advanced - Add XJ_System.






Add Library
String for a fulltext search...
Library
Placeholder
Company
(All companies)
Use Cases
Xinje
BasicUtils
Common
Fieldbus
HighSpeedIo
Pid
System
XJ_System 1.0.1.2 Xinje
Group by category
Display all versions (for experts only)
Details...
Library Repository...
OK
Cancel

2-6. ECAT_FROMTO

2-6-1. Function overview

Used for reading and writing module parameters on remote IO, for temperature modules and weighing modules.

2-6-2. Function block introduction

2-6-2-1. Remote IO read【XJ_ECATFromTo.XJ_EC_FROM】

Remote IO read [XJ_ECATFromTo.XJ_EC_FROM]			
Execution condition	Rising edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_ECATFromTo		

Command	Name	Graphical representation	ST language
XJ_ECATFromTo.XJ_EC_FROM	Remote IO read	XJ_EC_FROM_0 XJ_ECATFromTo.XJ_EC_FROM EN, awValue, xExcute, iStationNumber, iModuleID, iModuleAddress, iNum ENO, xDone, xBusy, xError, eErrorID	XJ_ECATFromTo.XJ_EC_FROM_0(awValue:= , xExcute:= , iStationNumber:= , iModuleID:= , iModuleAddress:= , iNum:= , xDone=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
awValue	Read value	POINTER TO WORD	-	-	The longest array length [0..100]
xExcute	Rising edge trigger	BOOL	-	-	
iStationNumber	Station no.	UINT		-	ECAT station no., such as 1001
iModuleID	Module no.	INT	0-15	-	One remote IO with a maximum of 16 modules
iModuleAddress	Module address	DWORD	-	-	Refer to the module address in the expansion module manual
iNum	Register numbers	INT	-	-	Number of consecutive Word reads

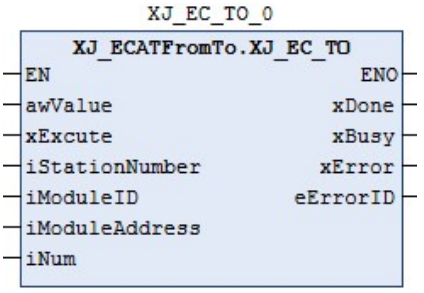
(2) Output variables

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
xDone	Boolean output	BOOL	TRUE,FALSE	FALSE	Read completed is TRUE
xBusy	Boolean output	BOOL	TRUE,FALSE	FALSE	Reading is TRUE
xError	Boolean output	BOOL	TRUE,FALSE	FALSE	True when an exception occurs

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
eErrorID	Present mode	EC_FT_ERRID	-	-	When normal, the value is 0. When an exception occurs, an error code is output

2-6-2-2. Remote IO write【XJ_ECATFromTo.XJ_EC_TO】

Remote IO write [XJ_ECATFromTo.XJ_EC_TO]			
Execution condition	Rising edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_ECATFromTo		

Command	Name	Graphical representation	ST language
XJ_ECATFromTo.XJ_EC_TO	Remote IO write		XJ_ECATFromTo.XJ_EC_TO_0(awValue:= , xExcute:= , iStationNumber:= , iModuleID:= , iModuleAddress:= , iNum:= , xDone=> , xBusy=> , xError=> , eErrorID=>);

(1) Input variables

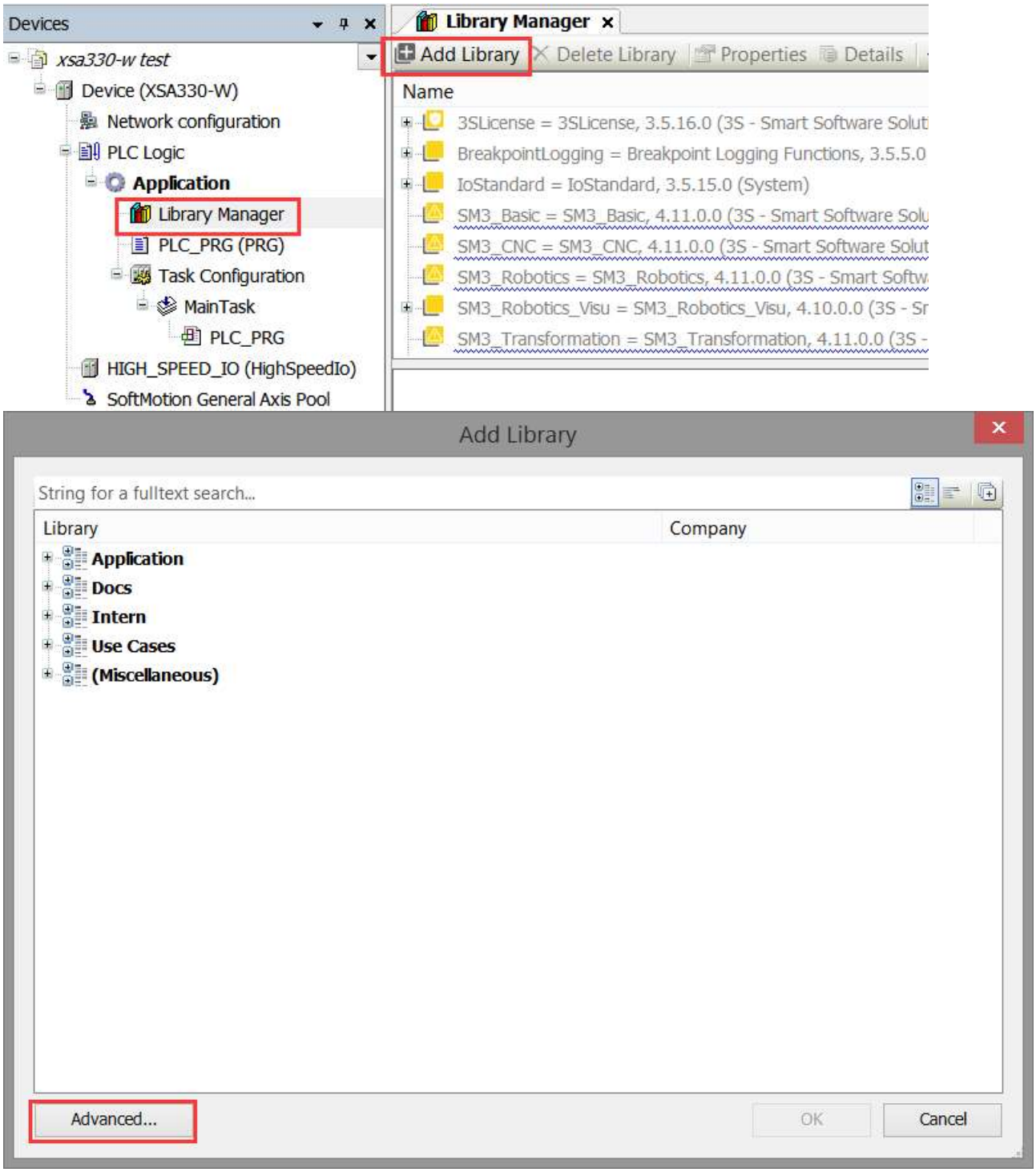
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
awValue	write in value	POINTER TO WORD	-	-	The longest array length [0..100]
xExcute	Rising edge trigger	BOOL	-	-	
iStationNumber	Station no.	UINT		-	ECAT station no., such as 1001
iModuleID	Module no.	INT	0-15	-	One remote IO with a maximum of 16 modules
iModuleAddress	Module address	DWORD	-	-	Refer to the module address in the expansion module manual
iNum	Register numbers	INT	-	-	Number of consecutive Word writes

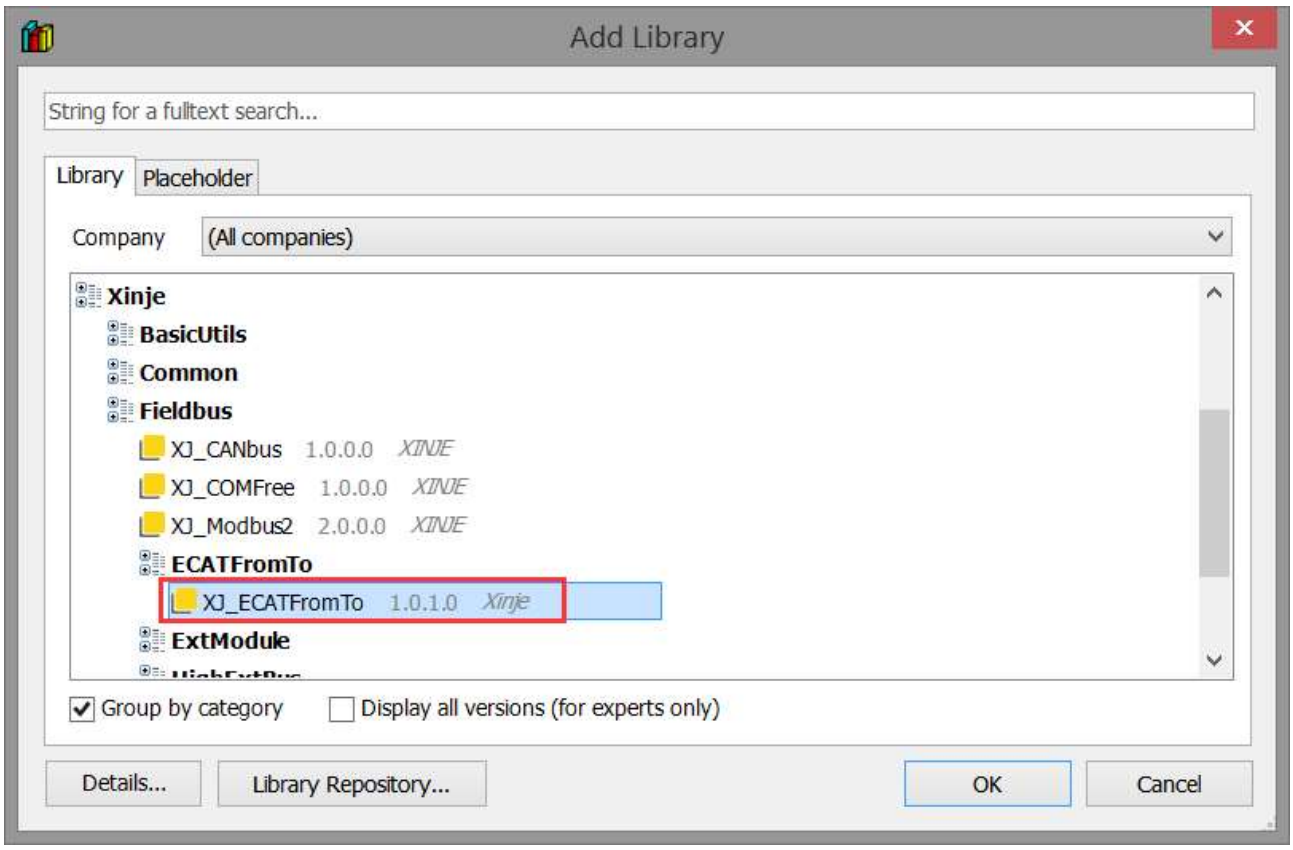
(2) Output variables

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
xDone	Boolean output	BOOL	TRUE,FALSE	FALSE	Write completed is TRUE
xBusy	Boolean output	BOOL	TRUE,FALSE	FALSE	Writing is TRUE

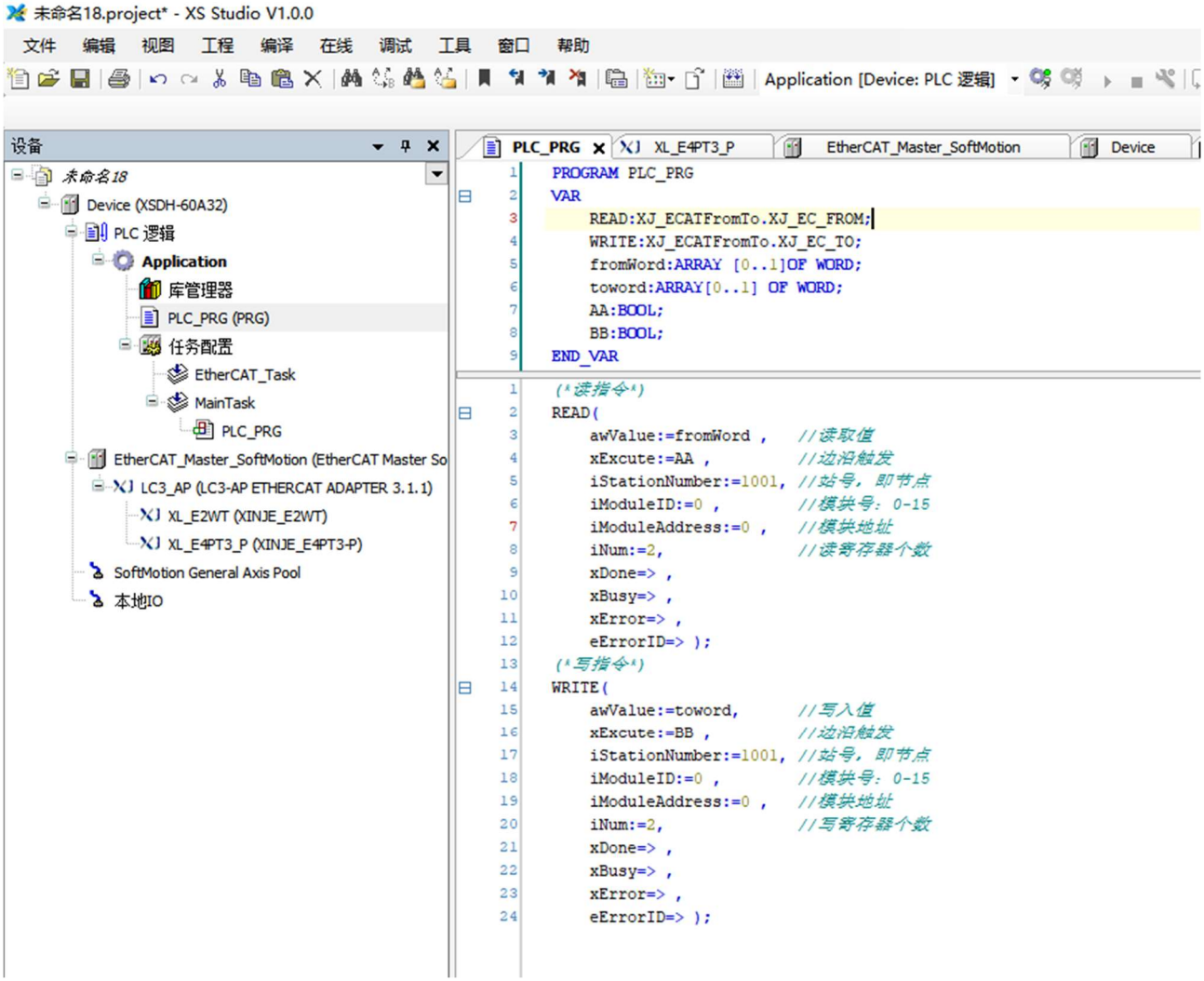
xError	Boolean output	BOOL	TRUE,FALSE	FALSE	True when an exception occurs
eErrorID	Present mode	EC_FT_ERRID	-	-	When normal, the value is 0. When an exception occurs, an error code is output

2-6-3. Parameter configuration




Add Library
String for a fulltext search...
Library
Placeholder
Company
(All companies)
Xinje
BasicUtils
Common
Fieldbus
XJ_CANbus 1.0.0.0 XINJE
XJ_COMFree 1.0.0.0 XINJE
XJ_Modbus2 2.0.0.0 XINJE
ECATFromTo
XJ_ECATFromTo 1.0.1.0 Xinje
ExtModule
Group by category
Display all versions (for experts only)
Details...
Library Repository...
OK
Cancel

2-6-4. Application



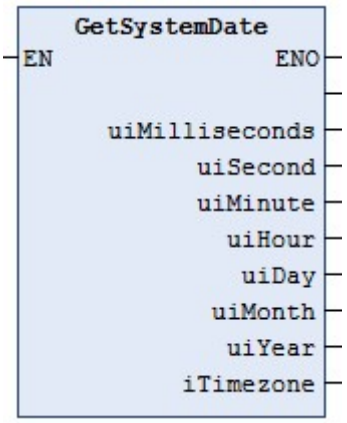
2-7. Date time

2-7-1. Obtain the current system clock GetSystemDate

(1) Instruction overview

Get the current system's time zone, date, and time.

Obtain the current system clock [GetSystemDate]			
Execution condition	-	Suitable model	XS, XSA models
Firmware	XS: V2.2.0 and up XSA: -	Software	XS StudioV2.2.0 and up (XS models)
Library	XJ_FunctionUtils		

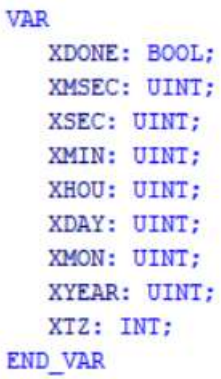
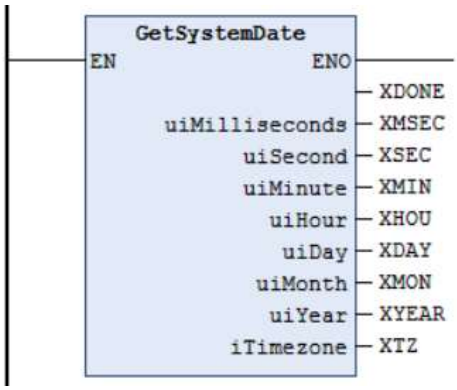
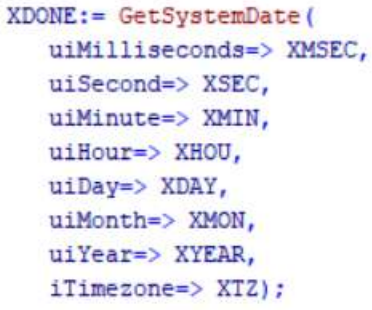
Command	Name	Graphical representation	ST language
GetSystemDate	Obtain the current system clock		GetSystemDate(uiMilliseconds=> , uiSecond=> , uiMinute=> , uiHour=> , uiDay=> , uiMonth=> , uiYear=> , iTimezone=>);

(2) Related variables

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
uiMilliseconds	Millisecond	UINT	-	-	Millisecond
uiSecond	Second	UINT	-	-	Second
uiMinute	Minute	UINT	-	-	Minute
uiHour	Hour	UINT	-	-	Hour
uiDay	Day	UINT	-	-	Day
uiMonth	Month	UINT	-	-	Month
uiYear	Year	UINT	-	-	Year
iTimezone	Time zone	INT	-	-	Time zone

(3) Program example

DEMO	LD	ST

Define variables	
Program example	 
Operation result	

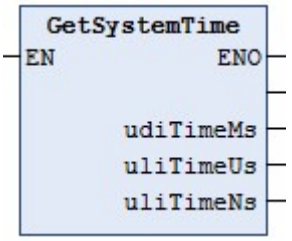
表达式	类型	值
XDONE	BOOL	TRUE
XMSEC	UINT	358
XSEC	UINT	39
XMIN	UINT	11
XHOU	UINT	14
XDAY	UINT	7
XMON	UINT	9
XYEAR	UINT	2023
XTZ	INT	-480

2-7-2. Obtain system runtime GetSystemTime

(1) Instruction overview

Obtain the running time after startup.

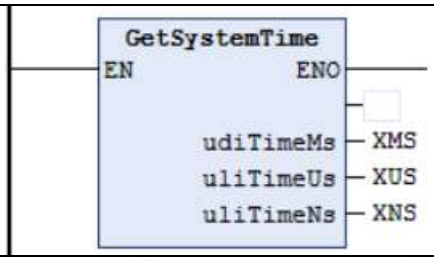
Obtain system runtime [GetSystemTime]			
Execution condition	-	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
GetSystemTime	Obtain system runtime		GetSystemTime(udiTimeMs=> , uliTimeUs=> , uliTimeNs=>);

(2) Related variables

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
udiTimeMs	Runtime ms	UDINT	-	-	Runtime ms
uliTimeUs	Runtime us	ULINT	-	-	Runtime us
uliTimeNs	Runtime ns	ULINT	-	-	Runtime ns

(3) Program example

DEMO	LD	ST
Define variables		VAR XMS: UDINT; XUS: ULINT; XNS: ULINT; END_VAR
Program example		GetSystemTime(udiTimeMs=> XMS, uliTimeUs=> XUS, uliTimeNs=> XNS);
Operation result		

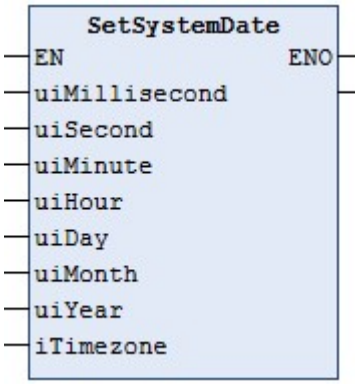
2-7-3. Set the current system clock SetSystemDate

(1) Instruction overview

Set the current system's time zone, date, and time.

Set the current system clock [SetSystemDate]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	XS: V2.2.0 and up XSA: -	Software	XS StudioV2.2.0 and up (XS models)
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language

SetSystemDate	Set the current system clock		`SetSystemDate(`
			`uiMillisecond:= ,`
			`uiSecond:= ,`
			`uiMinute:= ,`
			`uiHour:= ,`
			`uiDay:= ,`
			`uiMonth:= ,`
			`uiYear:= ,`
			`iTimezone:= );`

(2) Related variables

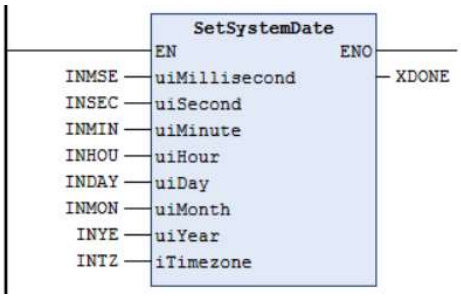
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
uiMilliseconds	Millisecond	UINT	-	-	Millisecond
uiSecond	Second	UINT	-	-	Second
uiMinute	Minute	UINT	-	-	Minute
uiHour	Hour	UINT	-	-	Hour
uiDay	Day	UINT	-	-	Day
uiMonth	Month	UINT	-	-	Month
uiYear	Year	UINT	-	-	Year
iTimezone	Time zone	INT	-	-	Time zone

(3) Program example

DEMO	LD	ST

Define variables		`VAR`
		`XDONE: BOOL;`
		`INMSE: UINT;`
		`INSEC: UINT:= 3;`
		`INMIN: UINT:= 31;`
		`INHOU: UINT:= 8;`
		`INDAY: UINT:= 9;`
		`INMON: UINT:= 9;`
		`INYE: UINT:= 2023;`
		`INTZ: INT;`
		`END_VAR`

Program example




```
XDONE:= SetSystemDate(
            uiMillisecond:= INMSE,
            uiSecond:= INSEC,
            uiMinute:= INMIN,
            uiHour:= INHOU,
            uiDay:= INDAY,
            uiMonth:= INMON,
            uiYear:= INYE,
            iTimezone:= INTZ);
```

Operation result

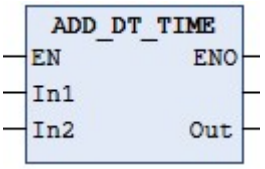
表达式	类型	值
XDONE	BOOL	TRUE
INMSE	UINT	0
INSEC	UINT	3
INMIN	UINT	31
INHOU	UINT	8
INDAY	UINT	9
INMON	UINT	9
INYE	UINT	2023
INTZ	INT	0

2-7-4. Add date and time ADD_DT_TIME

(1) Instruction overview

Add the date and time.

Add date and time [ADD_DT_TIME]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
ADD_DT_TIME	Add date and time		ADD_DT_TIME(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Date to be added	DT	As the data type	-	Date to be added
In2	Time to be added	TIME	As the data type	-	Time to be added
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	The result	DT	As the data type	-	The result

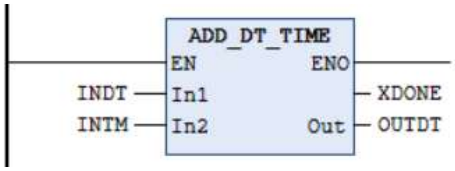
(3) Function

- The date time "In1" and time "In2" are added together, and the sum result "Out" is the date time.

(4) Note

- When the sum result exceeds the valid range of "Out", the program will not have any exceptions. Execute the following actions:
  DT#2106-2-7-6:28:15 + T#1s → DT#1970-1-1-0:0:0

(5) Program example

DEMO	LD	ST
Define variables		VAR INDT: DT:= DT#2023-9-9-9:11:15; INTM: TIME:= T#10S; OUTDT: DT; XDONE: BOOL; END_VAR
Program example		XDONE:= ADD_DT_TIME(In1:= INDT, In2:= INTM, Out=> OUTDT);

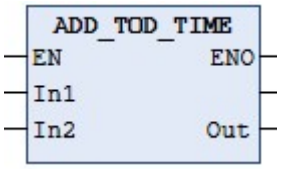
Operation result	表达式	类型	值
	INDT	DATE_AND_TIME	DT#2023-9-9-9:11:15
	INTM	TIME	T#10s
	OUTDT	DATE_AND_TIME	DT#2023-9-9-9:11:25
	XDONE	BOOL	TRUE

2-7-5. Time addition ADD_TOD_TIME

(1) Instruction overview

Time addition.

Time addition [ADD_TOD_TIME]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
ADD_TOD_TIME	Time addition		ADD_TOD_TIME(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Added time	TOD	As the data type	-	Added time
In2	Additive time	TIME	As the data type	-	Additive time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Sum result time	TOD	As the data type	-	Sum result time

(3) Function

- Add time "In1" and time "In2", and the sum result "Out" is the time.

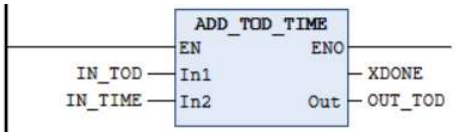
(4) Note

- When the sum result exceeds the valid range of "Out", the program will not have any exceptions. Execute the following actions:
  TOD#23:59:59.999 + T#1s1ms → TOD#0:0:1

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    IN_TOD: TOD:= TOD#10:11:16.003;
    IN_TIME: TIME:= T#2M16S;
    OUT_TOD: TOD;
    XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td> 
</td><td>

```
XDONE:= ADD_TOD_TIME(
    In1:= IN_TOD,
    In2:= IN_TIME,
    Out=> OUT_TOD);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN_TOD	TIME_OF_DAY	TOD#10:11:16.003
IN_TIME	TIME	T#2m16s
OUT_TOD	TIME_OF_DAY	TOD#10:13:32.003
XDONE	BOOL	TRUE

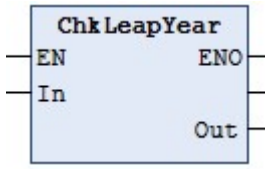
</td></tr>
</table>

2-7-6. Leap year judgment ChkLeapYear

(1) Instruction overview

Determine whether the specified year is a leap year.

Leap year judgment [ChkLeapYear]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
ChkLeapYear	Leap year judgment		`ChkLeapYear(In:= , Out=> );`

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Year	UINT	1970-2106	-	Year
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Judgment result	BOOL	FALSE/TRUE	-	True: leap year FALSE: Non leap year

(3) Function

- Determine whether the year "In" is a leap year. If it is a leap year, the value of the judgment result "Out" is true. If not, it is FALSE.

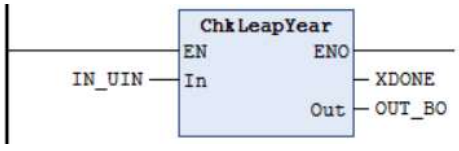
(4) Note

- When the value of "In" exceeds the valid range, the program will encounter an exception. The output value of "Out" is FALSE, and the function return value is FALSE.

(5) Program example

DEMO	LD	ST

Define variables		`VAR`
		`  IN_UIN: UINT:= 2020;`
		`  OUT_BO: BOOL;`
		`  XDONE: BOOL;`
		`END_VAR`

Program example		`XDONE:= ChkLeapYear(`
		`  In:= IN_UIN,`
		`  Out=> OUT_BO);`

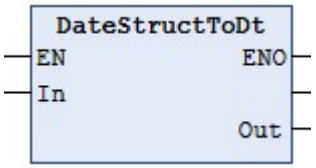
Operation result	表达式	类型	值
	IN_UIN	UINT	2020
	OUT_BO	BOOL	TRUE
	XDONE	BOOL	TRUE

2-7-7. Date and time combination DateStructToDt

(1) Instruction overview

Combine dates and moments that are decomposed into years, months, days, hours, minutes, and seconds.

Date and time combination [DateStructToDt]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
DateStructToDt	Date and time combination		DateStructToDt(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Date time	_sDT	As the data type	-	Date time before combining
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Date time	DT	DT#1970-1-1-0:0:0-DT#2106-2-7-6:28:15	DT#1970-1-1-0:0:0	Date and time after combining

(3) Function

- Combine the date and time "In" that are decomposed into "year", "month", "day", "hour", "minute", and "second".
- The members of the _sDT structure are as follows:

Variable	Name	Data type	Valid range	Initial value	Description
uiYear	Year	UINT	1970-2106	1970	Year
uiMonth	Month	UINT	1-12	1	Month
uiDay	Day	UINT	1-31	1	Day
uiHour	Hour	UINT	0-23	0	Hour
uiMinute	Minute	UINT	0-59	0	Minute
uiSecond	Second	UINT	0-59	0	Second
uiMillisecond	Millisecond	UINT	0-999	0	Millisecond
uiDayOfWeek	Week	UINT	1-7	1	Week

(4) Note

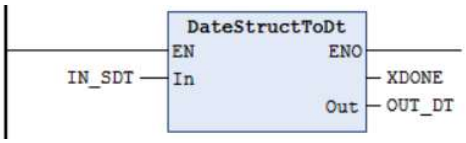
- The valid range for outputting the date and time "Out" is DT#1970-1-1-0:0:0 - DT#2106-2-7-6:28:15.
- If the output "Out" exceeds the valid range, the function returns a value of FALSE and the calculated result is DT#1970-1-1-0:0:0.
- If the input "In" exceeds the valid range of each element in the "_sDT" structure, the function returns a value of FALSE and the calculated result is DT#1970-1-1-0:0:0.
- The initial values of each element in the "_sDT" structure are the decomposition values of DT#1970-1-1-0:0:0.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    IN_SDT: _sDT;
    OUT_DT: DT;
    XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:= DateStructToDt(
    In:= IN_SDT,
    Out=> OUT_DT);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN_SDT	_sDT	
uiYear	UINT	2023
uiMonth	UINT	9
uiDay	UINT	9
uiHour	UINT	10
uiMinute	UINT	31
uiSecond	UINT	15
uiMillisecond	UINT	256
uiDayOfWeek	UINT	6
OUT_DT	DATE_AND_TIME	DT#2023-9-9-10:31:15
XDONE	BOOL	TRUE

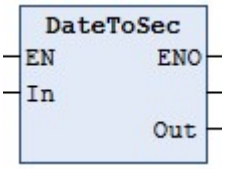
</td></tr>
</table>

2-7-8. Date second conversion DateToSec

(1) Instruction overview

Convert the input date to a number of seconds starting from January 1, 1970, 00:00:00.

Date second conversion [DateToSec]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
DateToSec	Date second conversion		`DateToSec(In:= , Out=> );`

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Date	DATE	As the data type	D#1970-1-1	Date
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	LINT	0-4294967295	-	Conversion result

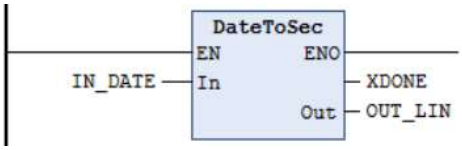
(3) Function

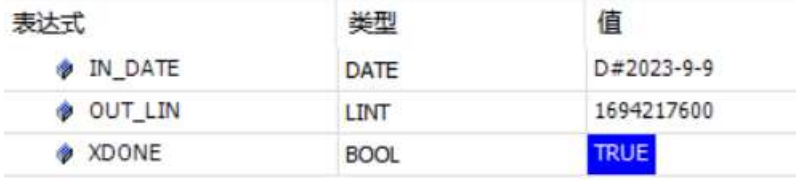
- Convert the 0:00:00s of the date "In" to the number of seconds starting from 0:00:00s on January 1, 1970. The converted value is measured in seconds.

(4) Program example

DEMO	LD	ST

Define variables		`VAR`
		`  IN_DATE: DATE:= D#2023-9-9;`
		`  OUT_LIN: LINT;`
		`  XDONE: BOOL;`
		`END_VAR`

Program example		`XDONE:= DateToSec(`
		`  In:= IN_DATE,`
		`  Out=> OUT_LIN);`

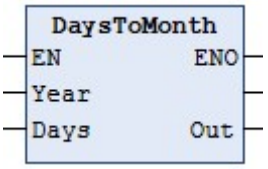
Operation result		

2-7-9. Days and months conversion DaysToMonth

(1) Instruction overview

Calculate the month of the date according to the days starting from January 1st.

Days and months conversion [DaysToMonth]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
DaysToMonth	Days and months conversion		DaysToMonth(Year:= , Days:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Year	Year	UINT	1970-2106	1970	Year
Days	Days	UINT	1-365 (When 'Year' is a leap year, it is 1-366)	1	Days from January 1st
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Month	USINT	1-12	-	Month

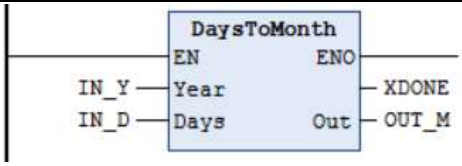
(3) Function

- Calculate the month of the day based on the number of days starting from January 1st of the year.

(4) Note

- When the value of "Year" exceeds the valid range, no exception will occur, and the value of "Out" is an illegal value.
- When the value of "Days" exceeds the valid range, the function returns a value of FALSE and "Out" becomes 0.

(5) Program example

DEMO	LD	ST
Define variables		VAR IN_Y: UINT:= 2023; IN_D: UINT:= 156; OUT_M: USINT; XDONE: BOOL; END_VAR
Program example		XDONE:= DaysToMonth(Year:= IN_Y, Days:= IN_D, Out=> OUT_M);

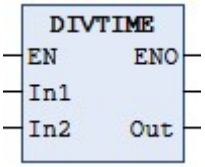
Operation result	表达式	类型	值
	IN_Y	UINT	2023
	IN_D	UINT	156
	OUT_M	USINT	6
	XDONE	BOOL	TRUE

2-7-10. Time division DIVTIME

(1) Instruction overview

Divide time by the specified divisor.

Time division [DIVTIME]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
DIVTIME	Time division		DIVTIME(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Dividend time	TIME	As the data type	-	Dividend time
In2	Divisor	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT REAL/LREAL	As the data type	-	Divisor
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Division result	TIME	As the data type	-	Division result time

(3) Function

- The time "In1" is divided by the divisor "In2", and the division result "Out" is the time.

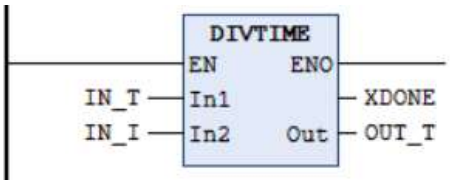
(4) Note

- When "In2" is negative, the program will encounter an exception. The output value of "Out" is T#0ms, and the function returns a value of FALSE.
- When the division result exceeds the valid range of "Out", the output value of "Out" is T#0ms, and the return value of the function is FALSE.

- When the input value of "In2" causes the output value of "Out" to exceed the upper limit of the TIME type, the output value of "Out" is T#0ms, and the function returns a value of FALSE.
- When "In2" is of integer type and has a value of 0, the function returns a value of FALSE and "Out" is 0.
- When "In2" is a real number, those less than 1ms in the division result will be rounded off.
- When "In2" is a real number, the division result with more significant digits may result in an error of several milliseconds.
- When "In2" is a real number and the value of "In2" is 0, +∞, -∞, and is not a numerical value, the function returns a value of True, and the value of "Out" is as follows:

“In2” input value	“Out” output value
0	T#49d17h2m47s295ms
+∞	T#0ms
-∞	T#0ms
numerical value	T#0ms

(5) Program example

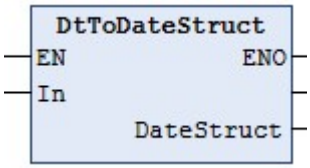
DEMO	LD	ST
Define variables		VAR IN_T: TIME:= T#2D16H8M6S200MS; IN_I: INT:= 2; OUT_T: TIME; XDONE: BOOL; END_VAR
Program example		XDONE:= DIVTIME(In1:= IN_T, In2:= IN_I, Out=> OUT_T);
Operation result		

2-7-11. Date time decomposition DtToDateStruct

(1) Instruction overview

Decompose the date and time into years, months, days, hours, minutes, and seconds.

Date time decomposition [DtToDateStruct]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
DtToDateStruct	Date time decomposition		DtToDateStruct(In:= , DateStruct=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Date time	DT	DT#1970-1-1-0:0:0- DT#2106-2-7-6:28:15	DT#1970-1-1-0:0:0	Date and time before decomposition
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Date time	_sDT	As the data type	-	Decomposed date and time

(3) Function

- Decompose the date and time "In" into "year", "month", "day", "hour", "minute", and "second", and output them in the format of the _sDT structure.
- The members of the _sDT structure are as follows:

Variable	Name	Data type	Effective range	Initial value	Description
uiYear	Year	UINT	1970-2106	1970	Year
uiMonth	Month	UINT	1-12	1	Month
uiDay	Day	UINT	1-31	1	Day
uiHour	Hour	UINT	0-23	0	Hour
uiMinute	Minute	UINT	0-59	0	Minute
uiSecond	Second	UINT	0-59	0	Second
uiMillisecond	Millisecond	UINT	0-999	0	Millisecond
uiDayOfWeek	Week	UINT	1-7	1	Week

(4) Note

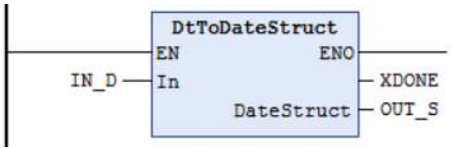
- The valid range for entering the date and time In is DT#1970-1-1-0:0:0 - DT#2106-2-7-6:28:15.
- If the input In exceeds the valid range, the function returns a value of FALSE and calculates the decomposition result of DT#1970-1-1-0:0:0.
- The initial values of each element in the structure of the _sDT are the decomposition values of DT#1970-1-1-0:0:0.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    IN_D: DT:= DT#2023-9-9-11:31:21;
    OUT_S: _sDT;
    XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:= DtToDateStruct(
    In:= IN_D,
    DateStruct=> OUT_S);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN_D	DATE_AND_TIME	DT#2023-9-9-11:31:21
OUT_S	_sDT	
uiYear	UINT	2023
uiMonth	UINT	9
uiDay	UINT	9
uiHour	UINT	11
uiMinute	UINT	31
uiSecond	UINT	21
uiMillisecond	UINT	0
uiDayOfWeek	UINT	6
XDONE	BOOL	TRUE

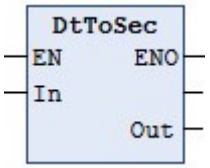
</td></tr>
</table>

2-7-12. Date, time, second conversion DtToSec

(1) Instruction overview

Convert the date and time to seconds starting from January 1, 1970, 00:00:00.

Date, time, second conversion [DtToSec]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
DtToSec	Date, time, second conversion		DtToSec(In:= , Out=>);

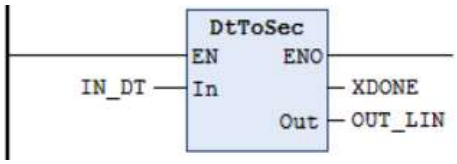
(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Date time	DT	As the data type	DT#1970-1-1-0:0:0	Date time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	LINT	0-4294967295	-	Conversion result

(3) Function

- Convert the date and time "In" to the number of seconds starting from January 1, 1970, 00:00:00. The converted value is measured in seconds. Values less than 1 second will be rounded off.

(4) Program example

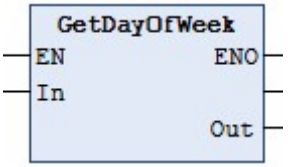
DEMO	LD	ST
Define variables	VAR IN_DT: DT:= DT#2023-9-9-11:36:22; OUT_LIN: LINT; XDONE: BOOL; END_VAR	
Program example		XDONE:= DtToSec(In:= IN_DT, Out=> OUT_LIN);
Operation result		

2-7-13. Get week information GetDayOfWeek

(1) Instruction overview

Get the week information for the specified month, month, and day.

Get week information [GetDayOfWeek]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
GetDayOfWeek	Get week information		`GetDayOfWeek(In:= , Out=> );`

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Date	DATE/DT	DATE#1970-1-1-DATE#2106-2-7 DT#1970-1-1-0:0:0-DT#2106-2-7-6:28:15	-	Year month day
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Week	_eDAYOFWEEK	As the data type	_MON	Week information

(3) Function

- Obtain the week information of the year, month, and day shown in "In".
- The data type of "Out" is the enumeration type _eDAYOFWEEK, and the meaning of the enumeration value is as follows:

_eDAYOFWEEK enum value	Meaning
_MON	Monday
_TUE	Tuesday
_WED	Wednesday
_THU	Thursday
_FRI	Friday
_SAT	Saturday
_SUN	Sunday

(4) Note

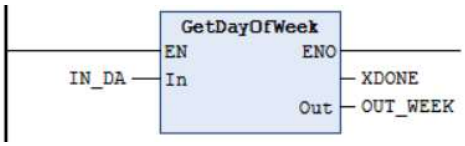
- When the value of "In" exceeds the valid range, the program will encounter an exception. The output value of "Out" is _MON, and the function returns a value of FALSE.
- The input parameter passed to "In" must be a variable. If a constant is passed, a compilation error occurs.
- "In" requires the use of DATA and DT types. For data types TIME and TOD, compile without error, and the function returns FALSE after running; For other types of data, compile an error.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    IN_DA: DATE:= DATE#2023-9-9;
    OUT_WEEK: _eDAYOFWEEK;
    XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:= GetDayOfWeek(
    In:= IN_DA,
    Out=> OUT_WEEK);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN_DA	DATE	D#2023-9-9
OUT_WEEK	_EDAYOFWEEK	_SAT
XDONE	BOOL	TRUE

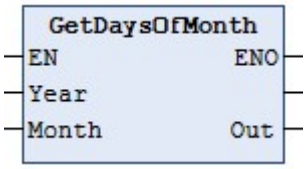
</td></tr>
</table>

2-7-14. Obtain the number of days in the target month GetDaysOfMonth

(1) Instruction overview

Get the number of days in the specified month.

Obtain the number of days in the target month [GetDaysOfMonth]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
GetDaysOfMonth	Obtain the number of days in the target month		GetDaysOfMonth(Year:= , Month:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Year	Year	UINT	1970-2106	-	Year
Month	Month	USINT	1-12	-	Month
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Days	USINT	28-31	-	Days

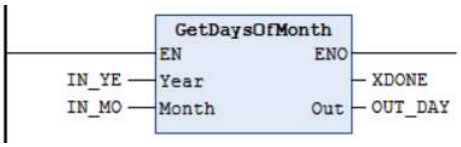
(3) Function

- Obtain the number of days for the target year "Year" and target month "Month".

(4) Note

- When the value of "Year" exceeds the valid range, the program will encounter an exception. The output value of "Out" is 0, and the function return value is FALSE.
- When the value of "Month" exceeds the valid range, the program will encounter an exception. The output value of "Out" is 0, and the function return value is FALSE.

(5) Program example

DEMO	LD	ST
Define variables		VAR IN_YE: UINT:= 2023; IN_MO: USINT:= 9; OUT_DAY: USINT; XDONE: BOOL; END_VAR
Program example		XDONE:= GetDaysOfMonth(Year:= IN_YE, Month:= IN_MO, Out=> OUT_DAY);

Operation result	表达式	类型	值
	IN_YE	UINT	2023
	IN_MO	USINT	9
	OUT_DAY	USINT	30
	XDONE	BOOL	TRUE

2-7-15. Obtain the current system time structure GetSystemDate_sDt

(1) Instruction overview

Obtain the current system time structure.

Obtain the current system time structure [GetSystemDate_sDt]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
GetSystemDate_sDt	Obtain the current system time structure		GetSystemDate_sDt(stSystemDate =>);

(2) Related variables

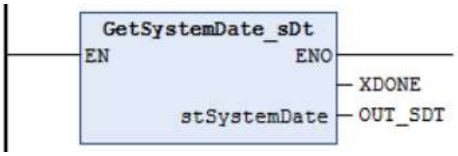
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
stSystemDate	System time	_sDT	-	-	System time structure

(3) Function

- Read the current system time and output it in the format of the _sDT structure.
- The members of the _sDT structure are as follows:

Variables	Name	Data type	Effective range	Initial value	Description
uiYear	Year	UINT	1970-2106	1970	Year
uiMonth	Month	UINT	1-12	1	Month
uiDay	Day	UINT	1-31	1	Day
uiHour	Hour	UINT	0-23	0	Hour
uiMinute	Minute	UINT	0-59	0	Minute
uiSecond	Second	UINT	0-59	0	Second
uiMillisecond	Millisecond	UINT	0-999	0	Millisecond
uiDayOfWeek	Week	UINT	1-7	1	Week

(4) Program example

DEMO	LD	ST
Define variables		VAR OUT_SDT: _sDT; XDONE: BOOL; END_VAR
Program example		XDONE:= GetSystemDate_sDt(stSystemDate=> OUT_SDT);
Operation result		

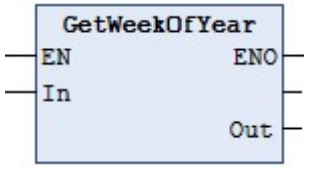
表达式	类型	值
OUT_SDT	_sDT	
uiYear	UINT	2023
uiMonth	UINT	9
uiDay	UINT	9
uiHour	UINT	12
uiMinute	UINT	21
uiSecond	UINT	7
uiMillisecond	UINT	995
uiDayOfWeek	UINT	6
XDONE	BOOL	TRUE

2-7-16. Get week information GetWeekOfYear

(1) Instruction overview

Calculate the specified year, month, and day as the week of the current year.

Get week information [GetWeekOfYear]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
GetWeekOfYear	Get week information		GetWeekOfYear(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Year month day	DATE/DT	DATE#1970-1-1-DATE#2106-2-7 DT#1970-1-1-0:0:0-DT#2106-2-7-6:28:15	-	Year month day

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Week	USINT	1-54	-	Weeks of the target year

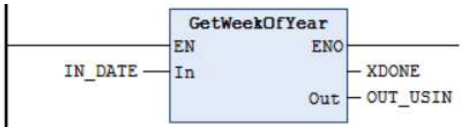
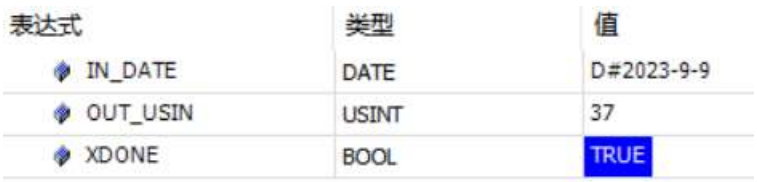
(3) Function

- Calculate the weeks of the target year, month, day “In”.
- Monday to Sunday is considered as one week, and when it changes from Sunday to Monday, it is counted as two weeks.
- January 1st must be the first week.

(4) Note

- When the value of "In" exceeds the valid range, the program will encounter an exception. The output value of "Out" is 0, and the function return value is FALSE.
- The input parameter passed to "In" must be a variable. If a constant is passed, a compilation error occurs.

(5) Program example

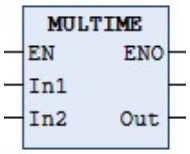
DEMO	LD	ST
Define variables		VAR IN_DATE: DATE:= DATE#2023-9-9; OUT_USIN: USINT; XDONE: BOOL; END_VAR
Program example		XDONE:= GetWeekOfYear(In:= IN_DATE, Out=> OUT_USIN);
Operation result		

2-7-17. Multiplying time MULTIME

(1) Instruction overview

Multiply time by the specified multiplier.

Multiplying time [MULTIME]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
MULTIME	Multiplying time		MULTIME(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Time to be multiplied	TIME	As the data type	-	Time to be multiplied
In2	Multiplier	USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/ REAL/LREAL	As the data type	-	Multiplier
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Multiplication result	TIME	As the data type	-	Multiplication result

(3) Function

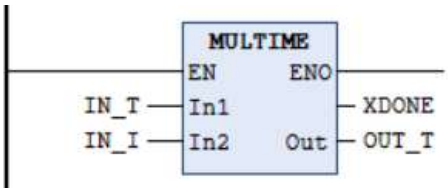
- The time "In1" is multiplied by the multiplier "In2", and the multiplication result "Out" is the time.

(4) Note

- When "In2" is negative, the program will encounter an exception. The output value of "Out" is T#0ms, and the function return value is FALSE.
- When the multiplication result exceeds the valid range of "Out", the output value of "Out" is T#0ms, and the return value of the function is FALSE.
- When the input value of "In2" causes the output value of "Out" to exceed the upper limit of the TIME type, the output value of "Out" is T#0ms, and the function returns a value of FALSE.
- When "In2" is a real number, the multiplication result with less than 1ms will be rounded off.
- When "In2" is a real number, multiplying result with more significant digits may result in an error of several milliseconds.
- The value of "In2" is +∞, -∞, and when it is not a numerical value, the value of "Out" is as follows:

“In2” input value	“Out” output value
+∞	T#0ms
-∞	T#0ms
Non numerical value	T#0ms

(5) Program example

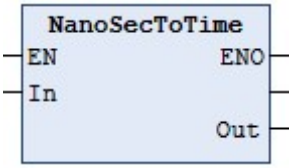
DEMO	LD	ST
Define variables		VAR IN_T: TIME:= T#5D3H10M16S; IN_I: INT:= 2; OUT_T: TIME; XDONE: BOOL; END_VAR
Program example		XDONE:= MULTIME(In1:= IN_T, In2:= IN_I, Out=> OUT_T);
Operation result		

2-7-18. Nanosecond time conversion NanoSecToTime

(1) Instruction overview

Convert nanoseconds to time.

Nanosecond time conversion [NanoSecToTime]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
NanoSecToTime	Nanosecond time conversion		NanoSecToTime(In:= , Out=>);

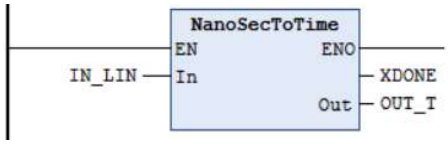
(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Nanoseconds	LINT	0-4294967295000000	0	Nanoseconds
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	TIME	As the data type	-	Time after converted

(3) Function

- Convert nanosecond "In" to time.

(4) Program example

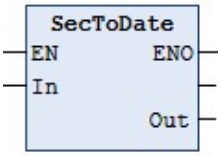
DEMO	LD	ST
Define variables		VAR IN_LIN: LINT:= 65535000000; OUT_T: TIME; XDONE: BOOL; END_VAR
Program example		XDONE:= NanoSecToTime(In:= IN_LIN, Out=> OUT_T);
Operation result		

2-7-19. Second date conversion SecToDate

(1) Instruction overview

Convert the number of seconds counted from January 1, 1970, 00:00:00, to a date.

Second date conversion [SecToDate]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SecToDate	Second date conversion		SecToDate(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Seconds	LINT	0-4294967295	0	The number of seconds counted from 00:00:00 on January 1, 1970
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	DATE	As the data type	-	Date after conversion

(3) Function

- Convert the number of seconds "In" starting from 00:00:00 on January 1, 1970 to a date. Values less than 1 day will be rounded off.

(4) Note

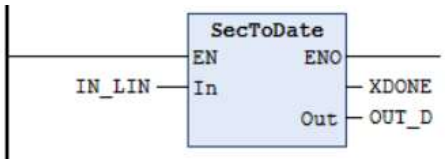
- When the value of "In" exceeds the valid range, the function return value becomes FALSE, and the value of "Out" becomes D#1970-1-1.

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    IN_LIN: LINT:= 1694217600;
    OUT_D: DATE;
    XDONE: BOOL;
END_VAR
```

</td></tr>
<tr><td>Program example</td><td>



</td><td>

```
XDONE:= SecToDate(
    In:= IN_LIN,
    Out=> OUT_D);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
IN_LIN	LINT	1694217600
OUT_D	DATE	D#2023-9-9
XDONE	BOOL	TRUE

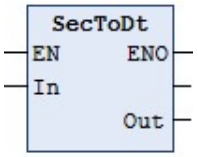
</td></tr>
</table>

2-7-20. Second date time conversion SecToDt

(1) Instruction overview

Convert the number of seconds starting from 00:00:00 on January 1, 1970 to date time.

Second date time conversion [SecToDt]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SecToDt	Second date time conversion		`SecToDt(In:= , Out=> );`

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Seconds	LINT	0-4294967295	0	The number of seconds starting from 00:00:00 on January 1, 1970
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	DT	As the data type	-	Date and time of conversion

(3) Function

- Convert the number of seconds "In" starting from 00:00:00 on January 1, 1970 to a date time.

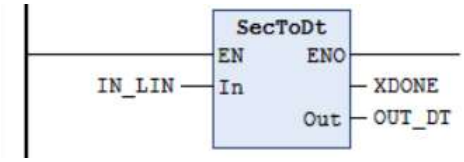
(4) Note

- When the value of "In" exceeds the valid range, the function return value becomes FALSE, and the value of "Out" becomes DT#1970-1-1-0:0:0.

(5) Program example

DEMO	LD	ST

Define variables		`VAR`
		`  IN_LIN: LINT:= 1694263261;`
		`  OUT_DT: DT;`
		`  XDONE: BOOL;`
		`END_VAR`

Program example		`XDONE:= SecToDt(`
		`  In:= IN_LIN,`
		`  Out=> OUT_DT);`

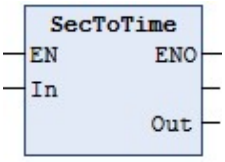
Operation result		

2-7-21. Second time conversion SecToTime

(1) Instruction overview

Convert seconds to time.

Second time conversion [SecToTime]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SecToTime	Second time conversion		SecToTime(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Seconds	LINT	0-4294967295	0	Seconds
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	TIME	As the data type	-	Time after conversion

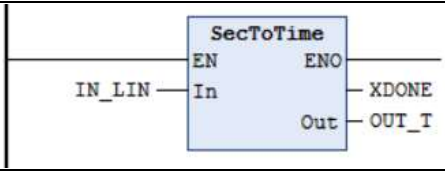
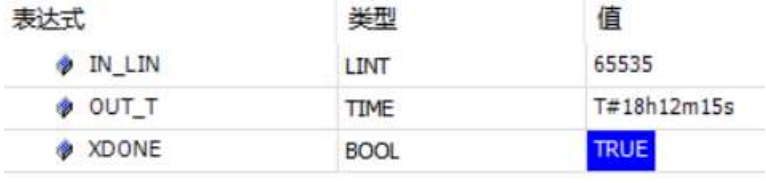
(3) Function

- Convert the number of seconds "In" to time.

(4) Note

- The unit of "In" is seconds, and the unit of "Out" is milliseconds.
- When the value of "In" exceeds the valid range, the function returns a value of FALSE and "Out" becomes 0.

(5) Program example

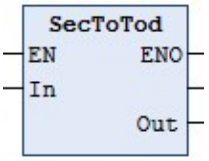
DEMO	LD	ST
Define variables		VAR IN_LIN: LINT:= 65535; OUT_T: TIME; XDONE: BOOL; END_VAR
Program example		XDONE:= SecToTime(In:= IN_LIN, Out=> OUT_T);
Operation result		

2-7-22. Second time conversion SecToTod

(1) Instruction overview

Convert the number of seconds starting from 0:0:00s to time.

Second time conversion [SecToTod]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SecToTod	Second time conversion		SecToTod(In:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Seconds	LINT	0-4294967295	0	The number of seconds counted from 0:0:00s
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	TOD	As the data type	-	The time after conversion

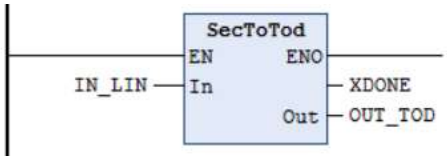
(3) Function

- Convert the number of seconds "In" starting from 0:0:00s to a time. When the value of "In" is greater than 24 hours, divide the value of "In" by 24 hours, and then convert the remainder to time.

(4) Note

- When the value of "In" exceeds the valid range, the return value of the function becomes FALSE, and the value of "Out" becomes TOD#0:0:0.

(5) Program example

DEMO	LD	ST
Define variables		VAR IN_LIN: LINT:= 65535; OUT_TOD: TIME_OF_DAY; XDONE: BOOL; END_VAR
Program example		XDONE:= SecToTod(In:= IN_LIN, Out=> OUT_TOD);

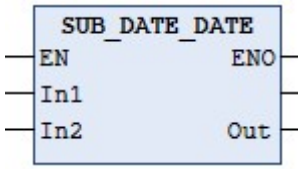
Operation result	

2-7-23. Date subtraction SUB_DATE_DATE

(1) Instruction overview

Subtract the date from the date.

Date subtraction [SUB_DATE_DATE]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SUB_DATE_DATE	Date subtraction		SUB_DATE_DATE(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Date to be subtracted	DATE	As the data type	-	Date to be subtracted
In2	Subtraction date	DATE	As the data type	-	Subtraction date
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Subtraction result time	TIME	As the data type	-	Subtraction result time

(3) Function

- Subtract the date "In1" from the date "In2", and the subtraction result "Out" is the time.

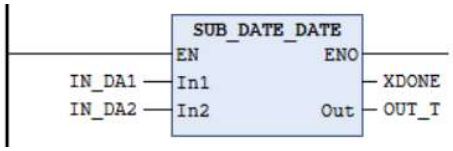
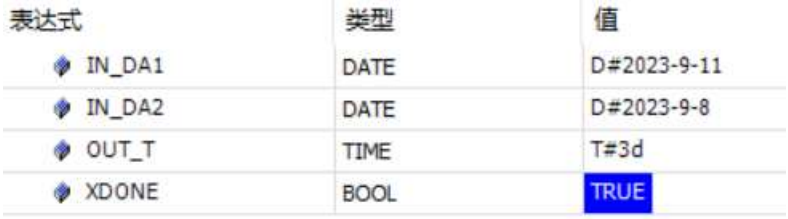
(4) Note

- When the subtraction result exceeds the valid range of "Out", the program will encounter an exception. The output value of "Out" is T#0ms, and the function return value is FALSE.
- The difference between the date "In1" and the date "In2" must be within the upper limit of the Time type (T#49d17h2m47s295ms), and the value of "In1" must be greater than "In2", otherwise the program will encounter an exception and the function will return a value of "FALSE".

Note: "In1" (close to the lower limit value) minus "In2" (close to the upper limit value), and the difference between "In1" and "In2" is within the upper limit value of Time type (T#49d17h2m47s295ms), the program will not have any exceptions. For example, subtracting the value of the lower limit (DATE#1970-1-1) from the value

of the upper limit (DATE#2106-2-7) yields (T#1d).

(5) Program example

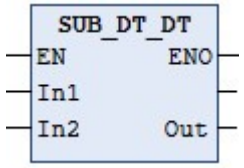
DEMO	LD	ST
Define variables		VAR IN_DA1: DATE:= DATE#2023-9-11; IN_DA2: DATE:= DATE#2023-9-8; OUT_T: TIME; XDONE: BOOL; END_VAR
Program example		XDONE:= SUB_DATE_DATE(In1:= IN_DA1, In2:= IN_DA2, Out=> OUT_T);
Operation result		

2-7-24. Subtracting date and time SUB_DT_DT

(1) Instruction overview

Subtract the date time from the date time.

Subtracting date and time [SUB_DT_DT]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SUB_DT_DT	Subtracting date and time		SUB_DT_DT(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Date time to be subtracted	DT	As the data type	-	Date time to be subtracted
In2	Subtraction date and time	DT	As the data type	-	Subtraction date and time
VAR_OUTPUT	Name	Data type	Effective range	Initial	Description

				value	
Out	Subtraction result time	TIME	As the data type	-	Subtraction result time

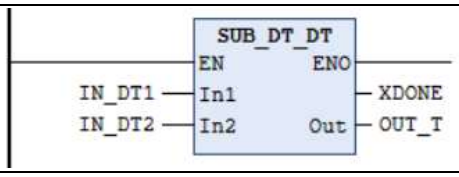
(3) Function

- Subtract the date time "In1" from the date time "In2", and the subtracted result "Out" is the time.

(4) Note

- When the subtraction result exceeds the valid range of "Out", the program will encounter an exception. The output value of "Out" is T#0ms, and the function return value is FALSE.
- If the value of the subtracted date "In1" is smaller than the value of the subtracted date "In2", the program will not have any exceptions. Execute the following action. DT#1970-1-1-0:0:0 - DT#2106-2-6-6:28:15 → T#1d1s.

(5) Program example

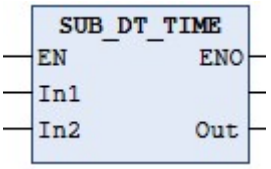
DEMO	LD	ST
Define variables	VAR IN_DT1: DT:= DT#2023-9-11-8:15:22; IN_DT2: DT:= DT#2023-9-8-9:25:18; OUT_T: TIME; XDONE: BOOL; END_VAR	
Program example		XDONE:= SUB_DT_DT(In1:= IN_DT1, In2:= IN_DT2, Out=> OUT_T);
Operation result		

2-7-25. Subtracting date, time, and time SUB_DT_TIME

(1) Instruction overview

Subtract time from date and time.

Subtracting date, time, and time [SUB_DT_TIME]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SUB_DT_TIME	Subtracting date, time, and time		SUB_DT_TIME(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Date time to be subtracted	DT	As the data type	-	Date time to be subtracted
In2	Subtraction time	TIME	As the data type	-	Subtraction time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Subtraction result date and time	DT	As the data type	-	Subtraction result date and time

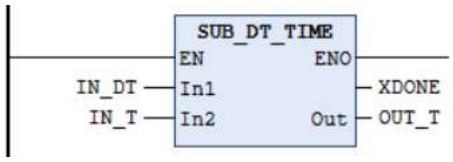
(3) Function

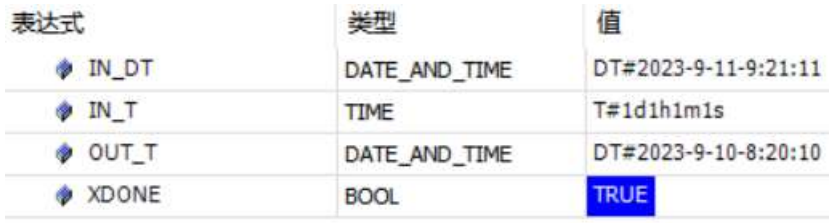
- Subtract time "In2" from date and time "In1", and the result of subtraction "Out" is the date and time.

(4) Note

- When the subtraction result exceeds the valid range of "Out", the program will not have any exceptions. Execute the following action. DT#1970-1-1-0:0:0 - T#1d → DT#2106-2-6-6:28:15.
- The minimum unit of "In2" value needs to be set to s (greater than or equal to 1000ms). When "In2" is less than 1000ms, the value of "Out" is consistent with the value of "In1".

(5) Program example

DEMO	LD	ST
Define variables		VAR IN_DT: DT:= DT#2023-9-11-9:21:11; IN_T: TIME:= T#1D1H1M1S; OUT_T: DT; XDONE: BOOL; END_VAR
Program example		XDONE:= SUB_DT_TIME(In1:= IN_DT, In2:= IN_T, Out=> OUT_T);

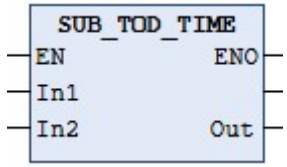
Operation result	

2-7-26. Subtracting time from time SUB_TOD_TIME

(1) Instruction overview

Subtract time from time.

Subtracting time from time [SUB_TOD_TIME]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SUB_TOD_TIME	Subtracting time from time		SUB_TOD_TIME(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Time to be subtracted	TOD	As the data type	-	Time to be subtracted
In2	Subtraction time	TIME	As the data type	-	Subtraction time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Subtraction result time	TOD	As the data type	-	Subtraction result time

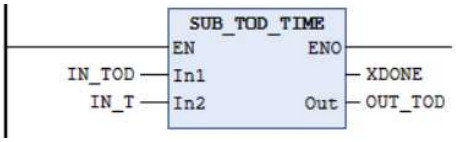
(3) Function

- Subtract time "In2" from time "In1", and the subtracted result "Out" is the time.

(4) Note

- TOD#0:0:0 - T#1ms → TOD#23:59:59:999。When the subtraction result exceeds the valid range of "Out", the program will not have any exceptions. Execute the following action. TOD#0:0:0 - T#1ms → TOD#23:59:59:999.

(5) Program example

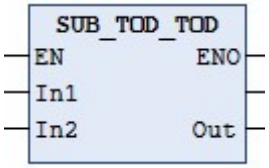
DEMO	LD	ST
Define variables		VAR IN_TOD: TOD:= TOD#18:31:29; IN_T: TIME:= T#3H11M2S; OUT_TOD: TOD; XDONE: BOOL; END_VAR
Program example		XDONE:= SUB_TOD_TIME(In1:= IN_TOD, In2:= IN_T, Out=> OUT_TOD);
Operation result		

2-7-27. Time subtraction SUB_TOD_TOD

(1) Instruction overview

Subtract time from time.

Time subtraction [SUB_TOD_TOD]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
SUB_TOD_TOD	Time subtraction		SUB_TOD_TOD(In1:= , In2:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In1	Time to be subtracted	TOD	As the data type	-	Time to be subtracted
In2	Subtraction time	TOD	As the data type	-	Subtraction time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Subtraction result time	TIME	As the data type	-	Subtraction result

					time

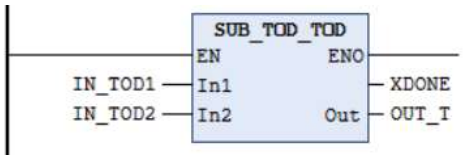
(3) Function

- Subtract time "In1" from time "In2", and the subtracted result "Out" is the time.

(4) Note

- When the value of the subtracted time "In1" is smaller than the value of the subtracted time "In2", the program will not have any exceptions. Execute the following actions:
  TOD#13:34:56.789 - TOD#13:35:56.789 → T#23h59m

(5) Program example

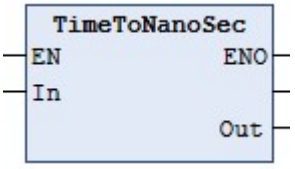
DEMO	LD	ST
Define variables	VAR IN_TOD1: TOD:= TOD#19:30:21; IN_TOD2: TOD:= TOD#10:12:56; OUT_T: TIME; XDONE: BOOL; END_VAR	
Program example		XDONE:= SUB_TOD_TOD(In1:= IN_TOD1, In2:= IN_TOD2, Out=> OUT_T);
Operation result		

2-7-28. Time nanosecond conversion TimeToNanoSec

(1) Instruction overview

Convert time to nanoseconds.

Time nanosecond conversion [TimeToNanoSec]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TimeToNanoSec	Time nanosecond conversion		`TimeToNanoSec(In:= , Out=> );`

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Time	TIME	As the data type	T#0s	Time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	LINT	As the data type	-	Nanoseconds

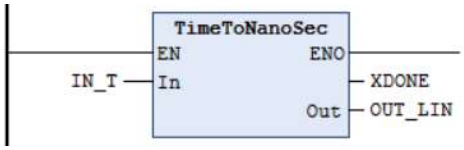
(3) Function

- Convert time "In" to nanoseconds.

(4) Program example

DEMO	LD	ST

Define variables		`VAR`
		`  IN_T: TIME:= T#3H11M26S;`
		`  OUT_LIN: LINT;`
		`  XDONE: BOOL;`
		`END_VAR`

Program example		`XDONE:= TimeToNanoSec(`
		`  In:= IN_T,`
		`  Out=> OUT_LIN);`

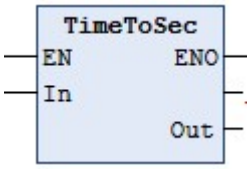
Operation result		

2-7-29. Time second conversion TimeToSec

(1) Instruction overview

Convert time to seconds.

Time second conversion [TimeToSec]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TimeToSec	Time second conversion		TimeToSec(In:= , Out=>);

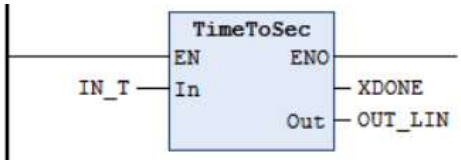
(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Time	TIME	As the data type	T#0s	Time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	LINT	As the data type	-	Seconds

(3) Function

- Convert the time "In" to seconds. Values less than 1 second will be rounded off.

(4) Program example

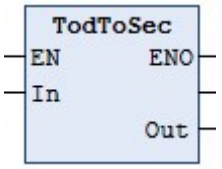
DEMO	LD	ST
Define variables		VAR IN_T: TIME:= T#3M10S; OUT_LIN: LINT; XDONE: BOOL; END_VAR
Program example		XDONE:= TimeToSec(In:= IN_T, Out=> OUT_LIN);
Operation result		

2-7-30. Time second conversion TodToSec

(1) Instruction overview

Convert the time to the number of seconds starting from 0:0:00s.

Time second conversion [TodToSec]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TodToSec	Time second conversion		TodToSec(In:= , Out=>);

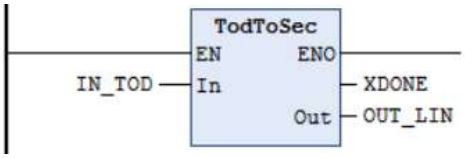
(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Time	TOD	As the data type	TOD#0:0:0	Time
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Conversion result	LINT	0-86399	-	Seconds

(3) Function

- Convert the time "In" to the number of seconds starting from 0:0:0:0. The converted value is measured in seconds. Values less than 1 second will be rounded off.

(4) Program example

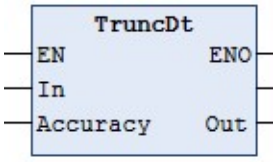
DEMO	LD	ST
Define variables		VAR IN_TOD: TOD:= TOD#2:1:10; OUT_LIN: LINT; XDONE: BOOL; END_VAR
Program example		XDONE:= TodToSec(In:= IN_TOD, Out=> OUT_LIN);
Operation result		

2-7-31. Date and time rounding off TruncDt

(1) Instruction overview

Round off values smaller than the specified unit in DT type variables.

Date and time rounding off [TruncDt]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TruncDt	Date and time rounding off		TruncDt(In:= , Accuracy:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Target date and time	DT	As the data type	-	Abandoning objects
Accuracy	Rounding unit	_eSUBSEC	As the data type	_SEC	Minimum time unit reserved
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	The processed result	DT	As the data type	-	Date time after rounding off

(3) Function

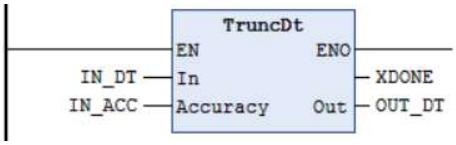
- In the object date time "In", values smaller than the rounding unit "Accuracy" will be rounded off, and the rounded value will be saved in the rounded date time "Out".
- The data type of "Accuracy" is the enumeration type _eSUBSEC, and the meaning of the enumeration value is as follows:

Enum	Meaning
_MILLISEC	Millisecond
_SEC	Second
_MINUTE	Minute
_HOUR	Hour

(4) Note

- The enumeration body value cannot be set to _MILLISEC, otherwise the program will encounter an exception and the function return value will be FALSE.

(5) Program example

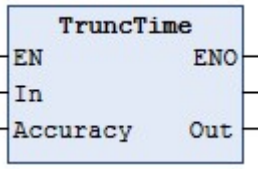
DEMO	LD	ST
Define variables	VAR IN_DT: DT:= DT#2023-9-11-10:30:21; IN_ACC: _eSUBSEC:=_eSUBSEC._HOUR; OUT_DT: DT; XDONE: BOOL; END_VAR	
Program example		XDONE:= TruncDt(In:= IN_DT, Accuracy:= IN_ACC, Out=> OUT_DT);
Operation result		

2-7-32. Time rounding off TruncTime

(1) Instruction overview

Round off values smaller than the specified unit in TIME type variables.

Time rounding off [TruncTime]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TruncTime	Time rounding off		TruncTime(In:= , Accuracy:= , Out=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Target time	TIME	As the data type	-	Rounding off target unit time
Accuracy	Rounding unit	_eSUBSEC	As the data type	_SEC	Minimum time unit retained
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Result after processing	TIME	As the data type	-	Time after rounding off

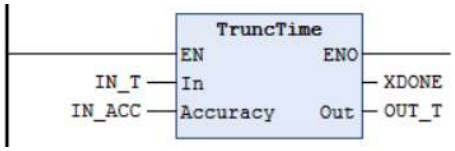
(3) Function

- In the object time "In", values smaller than the rounding unit "Accuracy" will be rounded off, and the rounded value will be saved in the rounding time "Out".
- The data type of "Accuracy" is the enumeration type _eSUBSEC, and the meaning of the enumeration value is as follows:

Enum	Meaning
_MILLISEC	Millisecond
_SEC	Second
_MINUTE	Minute
_HOUR	Hour

(4) Program example

DEMO	LD	ST
Define variables	VAR IN_T: TIME:= T#2D6H32M11S231MS; IN_ACC: _eSUBSEC:= _eSUBSEC._HOUR; OUT_T: TIME; XDONE: BOOL; END_VAR	

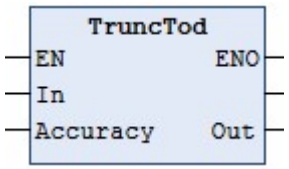
Program example		XDONE:= TruncTime(In:= IN_T, Accuracy:= IN_ACC, Out=> OUT_T);
Operation result		

2-7-33. Rounding off time TruncTod

(1) Instruction overview

Round off values smaller than the specified unit in TOD type variables.

Rounding off time [TruncTod]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TruncTod	Rounding off time		TruncTod(In:= , Accuracy:= , Out=>);

(2) Related variables

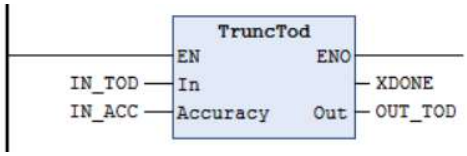
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
In	Target time	TOD	As the data type	-	Rounding off target unit time
Accuracy	Rounding off unit	_eSUBSEC	As the data type	_SEC	Minimum time unit retained
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Out	Result after processing	TOD	As the data type	-	Time after rounding off

(3) Function

- In the object time "In", values smaller than the rounding unit "Accuracy" will be rounded off, and the rounded value will be saved in the time "Out" after rounding.
- The data type of "Accuracy" is the enumeration type _eSUBSEC, and the meaning of the enumeration value is as follows:

Enum	Meaning
_MILLISEC	Millisecond
_SEC	Second
_MINUTE	Minute
_HOUR	Hour

(4) Program example

DEMO	LD	ST
Define variables		VAR IN_TOD: TOD:= TOD#11:12:29; IN_ACC: _eSUBSEC:= _eSUBSEC._HOUR; OUT_TOD: TOD; XDONE: BOOL; END_VAR
Program example		XDONE:= TruncTod(In:= IN_TOD, Accuracy:= IN_ACC, Out=> OUT_TOD);
Operation result		

2-8. Communication instructions

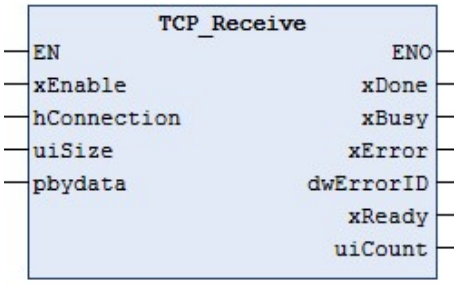
2-8-1. TCP communication

2-8-1-1. TCP communication data receive TCP_Receive

(1) Instruction overview

TCP communication data receive.

TCP communication data receive [TCP_Receive]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TCP_Receive	TCP communication data receive		TCP_Receive_0(xEnable:= , hConnection:= , uiSize:= , pbydata:= , xDone=> , xBusy=> , xError=> , dwErrorID=> , xReady=> , uiCount=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
xEnable	Command enable	BOOL	FALSE/TRUE	FALSE	Input high-level start command
hConnection	Connection handle	DWORD	As the data type	-	Connection handle for TCP communication
uiSize	Receiving data size	UINT	As the data type	-	Received data area size (BYTE)
pbyData	Point to the receive area	POINTER TO BYTE	-	-	Pointer to the beginning byte of the receiving area
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
xDone	End signal	BOOL	FALSE/TRUE	FALSE	Data reception termination signal
xBusy	Busy flag	BOOL	FALSE/TRUE	FALSE	Busy status
xError	Error flag	BOOL	FALSE/TRUE	FALSE	Error status
dwErrorID	Error code	DWORD	As the data type	-	Error code
xReady	Successfully established connection flag	BOOL	FALSE/TRUE	FALSE	Read data from the buffer, and if the data is not empty, set on the flag for one scan

					cycle
uiCount	Connection handle	_UXINT	-	-	Read the actual number of data from the receive buffer

(3) Function

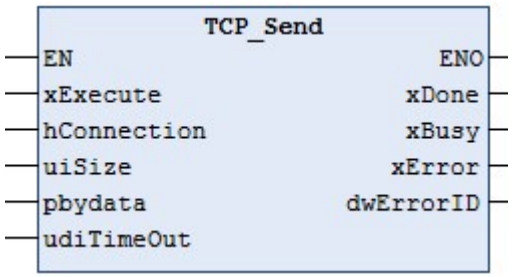
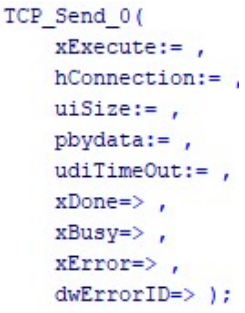
- When the TCP_Receive instruction xEnable detects a high level of power, it will attempt to read data from the TCP communication buffer and set xBusy to True.
- If the data is successfully read, the read data will be placed in an array with pbyData as the first address, and uiCount will output the actual number of data read. xReady will be set to TURE, and the two variables will maintain a scanning cycle.
- If there is an abnormality in network communication during the reading period, causing the reading to fail, xDone will be set to True, and the data reading function will fail. xEnable=True needs to be triggered again. Even trigger the xEnable=True of TCP_Connection or TCP_Client function block again.

2-8-1-2. TCP communication data send TCP_Send

(1) Instruction overview

TCP communication data send.

TCP communication data send [TCP_Send]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TCP_Send	TCP communication data send		


```
TCP_Send_0(
    xExecute:= ,
    hConnection:= ,
    uiSize:= ,
    pbydata:= ,
    udiTimeOut:= ,
    xDone=> ,
    xBusy=> ,
    xError=> ,
    dwErrorID=> );
```

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
xExecute	Command enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge start command
hConnection	Connection handle	_XWORD	-	-	Connection handle for TCP communication
uiSize	Send data size	UINT	As the data type	-	Send data area size (BYTE)
pbyData	Point to send area	POINTER TO BYTE	-	-	Pointer to the starting byte of the sending area
UditimeOut	Timeout	UDINT	As the data type	500000	Send data timeout, unit μs, default is 500ms
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
xDone	Completed signal	BOOL	FALSE/TRUE	FALSE	Data transmission completion signal
xBusy	Busy signal	BOOL	FALSE/TRUE	FALSE	Busy status
xError	Error flag	BOOL	FALSE/TRUE	FALSE	Error status
dwErrorID	Error code	DWORD	As the data type	-	Error code

(3) Function

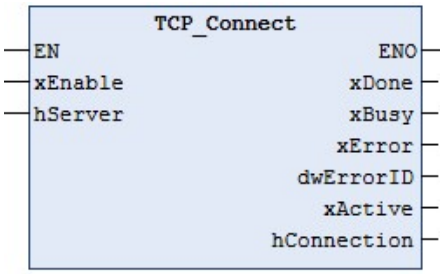
- When the TCP_Send instruction xExecute detects a rising edge, it will send data with a length of uiSize starting from the user set send buffer pbyData to the target device. If the transmission is successful within the timeout period, xDone is set to True.

2-8-1-3. Create TCP connection TCP_Connect

(1) Instruction overview

Create a TCP connection and connect to the server.

Create TCP connection [TCP_Connect]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TCP_Connect	Create a TCP connection and connect to the server		TCP_Connect_0(xEnable:= , hServer:= , xDone=> , xBusy=> , xError=> , dwErrorID=> , xActive=> , hConnection=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
xEnable	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level start command
hServer	Server-side handle	_XWORD	-	-	TCP server-side handle
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
xDone	Termination signal	BOOL	FALSE/TRUE	FALSE	Terminate listening signal
xBusy	Busy flag	BOOL	FALSE/TRUE	FALSE	Busy status
xError	Error flag	BOOL	FALSE/TRUE	FALSE	Error status
dwErrorID	Error code	DWORD	As the data type	-	Error code
xActive	Connection flag	BOOL	FALSE/TRUE	FALSE	Server side listening to the remote client connection handle flag
hConnection	Connection handle	_XWORD	-	-	Connection handle for establishing communication between server and client

(3) Function

- When the TCP_Connect command xEnable detects high power levels, the local server listens for connection requests from remote clients. After the client successfully connects to the server, a communication connection handle hConnection will be created between the server and the remote client.

(4) Note

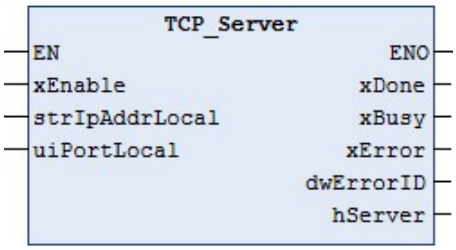
- When communication is established, xActive is set to True, and when communication is disconnected, xActive automatically resets.
- Communication monitoring requires the cooperation of the TCP_Receive function block, and can only be successfully monitored when TCP_Receive.xBusy is set to True.

2-8-1-4. Create TCP server-side communication service TCP_Server

(1) Instruction overview

Create TCP server-side communication service.

Create TCP server-side communication service [TCP_Server]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TCP_Server	Create TCP server-side communication service		TCP_Server_0(xEnable:= , strIpAddrLocal:= , uiPortLocal:= , xDone=> , xBusy=> , xError=> , dwErrorID=> , hServer=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
xEnable	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level start command
strIpAddrLocal	Local IP address	STRING	As the data type	-	The IP address of the local PLC communication network port
uiPortLocal	Local port	UINT	As the data type	-	Port number for local communication
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
xDone	Termination Flag	BOOL	FALSE/TRUE	FALSE	Server end communication termination flag
xBusy	Busy flag	BOOL	FALSE/TRUE	FALSE	Busy status
xError	Error flag	BOOL	FALSE/TRUE	FALSE	Error status
dwErrorID	Error code	DWORD	As the data type	-	Error code

hServer	Server handle	_XWORD	-	-	TCP server handle

(3) Function

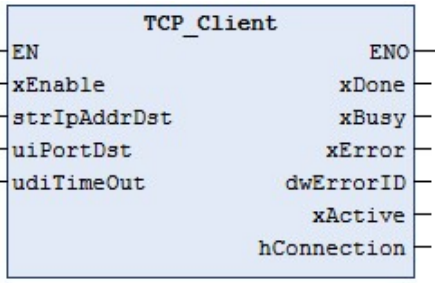
- Create a TCP server-side communication service instruction. When the TCP_Server instruction xEnable is true, a valid handle for TCP communication between the local server and the remote client will be created (the value of hServer is not equal to 0), and xBusy will remain true. If there is an error in the service related function during the communication process, xDone will be set to True, and the server-side related function blocks cannot be used, and TCP_Server needs to be triggered again.

2-8-1-5. Create TCP client communication service TCP_Client

(1) Instruction overview

Create TCP client communication service.

Create TCP client communication service [TCP_Client]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graphical representation	ST language
TCP_Client	Create TCP client communication service		TCP_Client_0(xEnable:= , strIpAddrDst:= , uiPortDst:= , udiTimeOut:= , xDone=> , xBusy=> , xError=> , dwErrorID=> , xActive=> , hConnection=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
xEnable	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level start command
strIpAddrDst	Remote device IP address	STRING	As the data type	-	Remote device IP address
uiPortDst	Remote device port number	UINT	As the data type	-	Remote device port number
udiTimeOut	Timeout	UDINT	As the data type	500000	The timeout time for requesting a connection, in μs, defaults to 500ms
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description

xDone	Termination signal	BOOL	FALSE/TRUE	FALSE	Client communication termination flag
xBusy	Busy flag	BOOL	FALSE/TRUE	FALSE	Busy status
xError	Error flag	BOOL	FALSE/TRUE	FALSE	Error status
dwErrorID	Error code	DWORD	As the data type	-	Error code
xActive	Successfully established connection flag	BOOL	FALSE/TRUE	FALSE	Flag bit for successful connection between client and remote server
hConnection	Connection handle	_XWORD	-	-	Connection handle for establishing communication between server and client

(3) Function

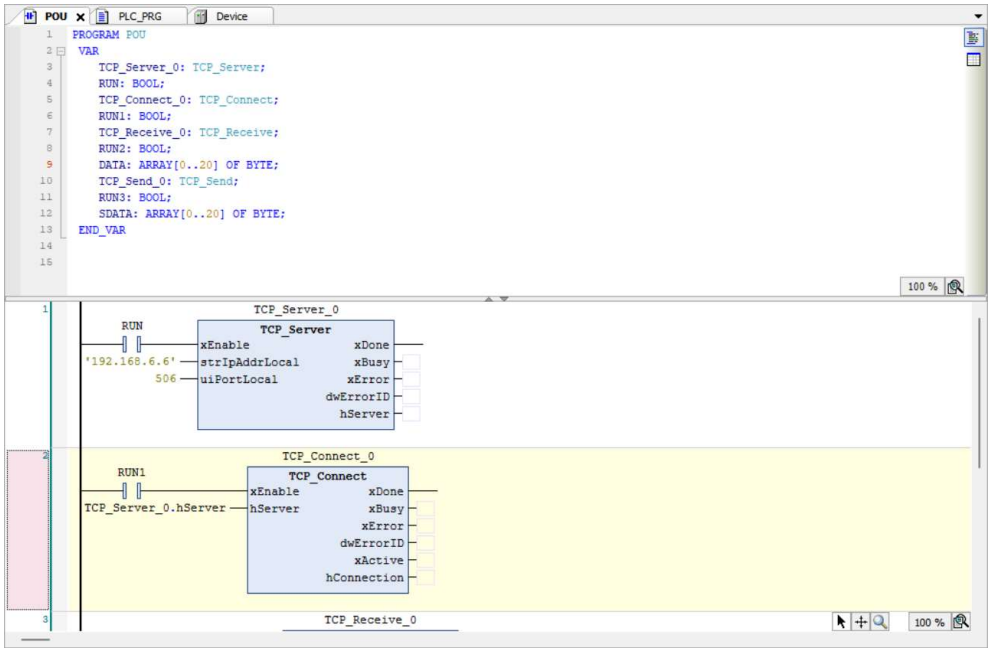
- Create a TCP client communication service instruction. When the TCP_Client instruction xEnable detects high level power, the local client requests to connect to the remote server. After the client successfully connects to the server, a communication connection handle hConnection will be created between the server and the remote client, and xActive will be set to True.

2-8-1-6. Example of TCP communication instruction program

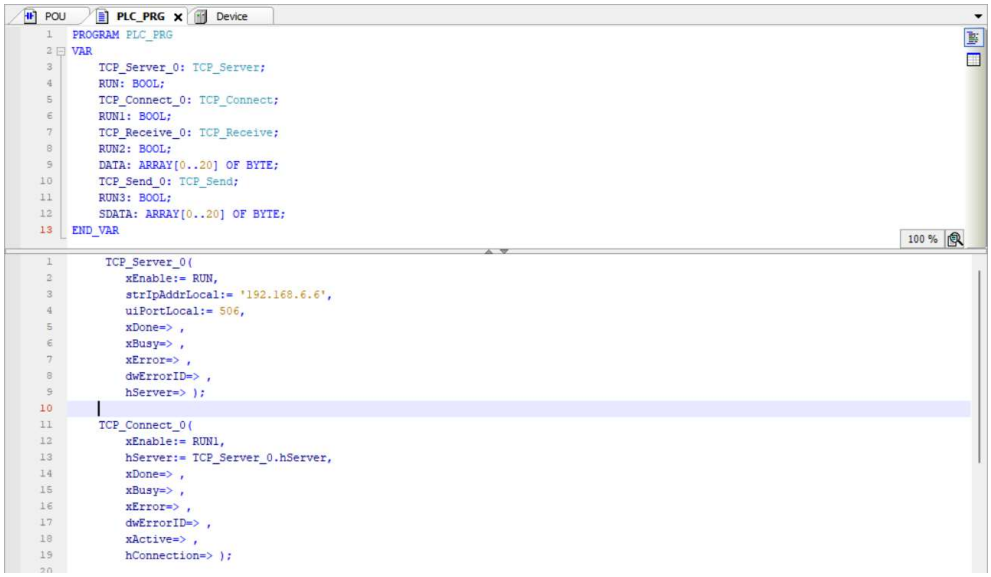
XSDH-60A32 as server, debugging assistant as client

The IP address of XSDH-60A32 is 192.168.6.6

1. Create TCP server-side communication services and establish connections between clients and servers; Trigger the TCP Server and TCP Connect instructions, where the input pin hServer of the TCP Connect instruction is the output pin hServer of TCP Server.
2. When triggering the debugging assistant connection, be sure to keep the IP address and port number consistent; The port number in the example is 506.



LD form

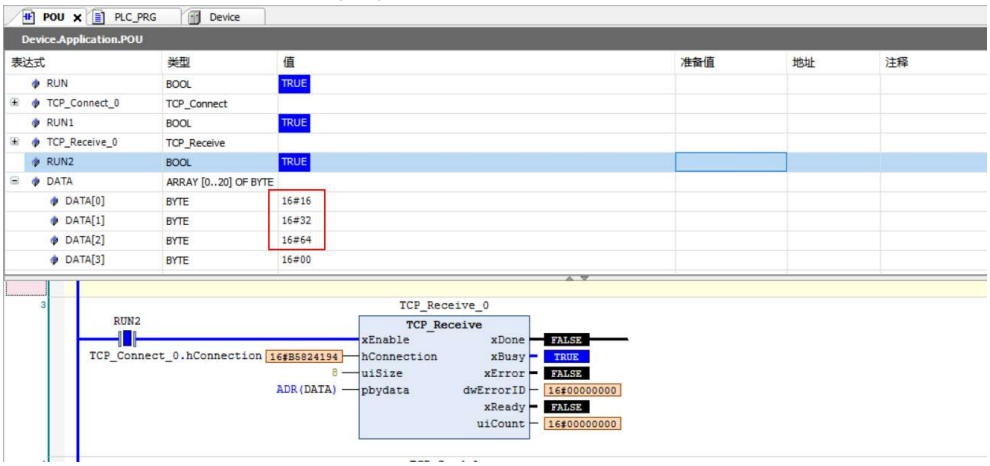


ST form

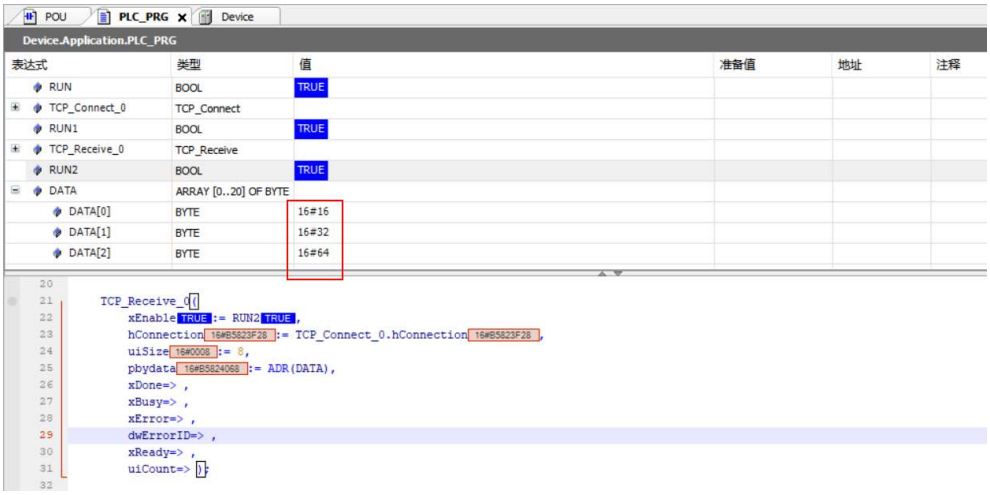
3. The server receives data sent by the client debugging assistant.

① Sending hexadecimal data 16#16 16#32 16#64 using the debugging assistant.

② Trigger the TCP_Receive instruction, and the received data is stored at the address pointed to by the pbyData pointer, as shown in the following figure. Received 16#16, 16#32, 16#64.



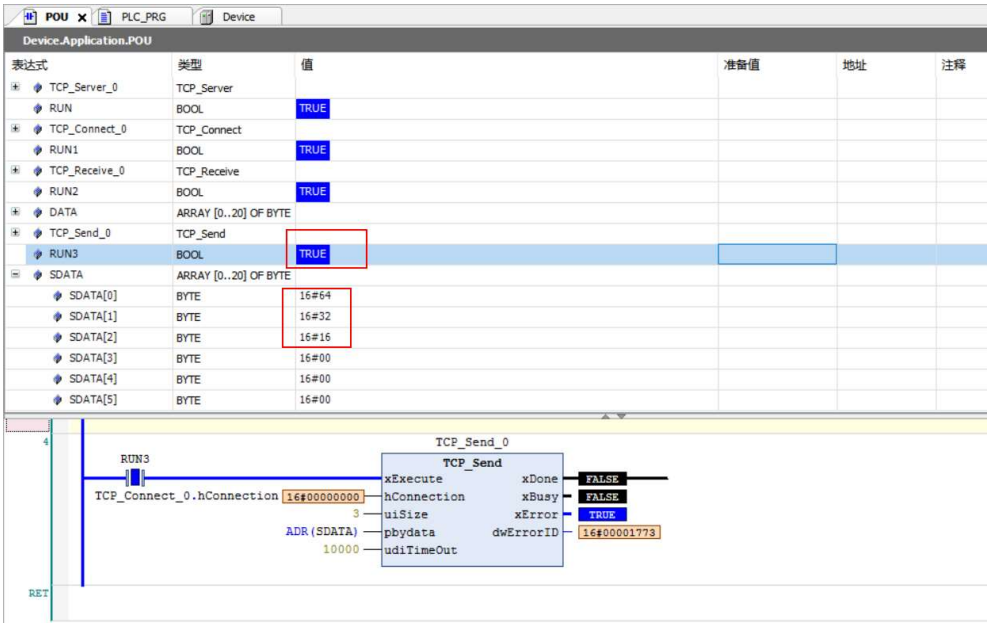
LD form



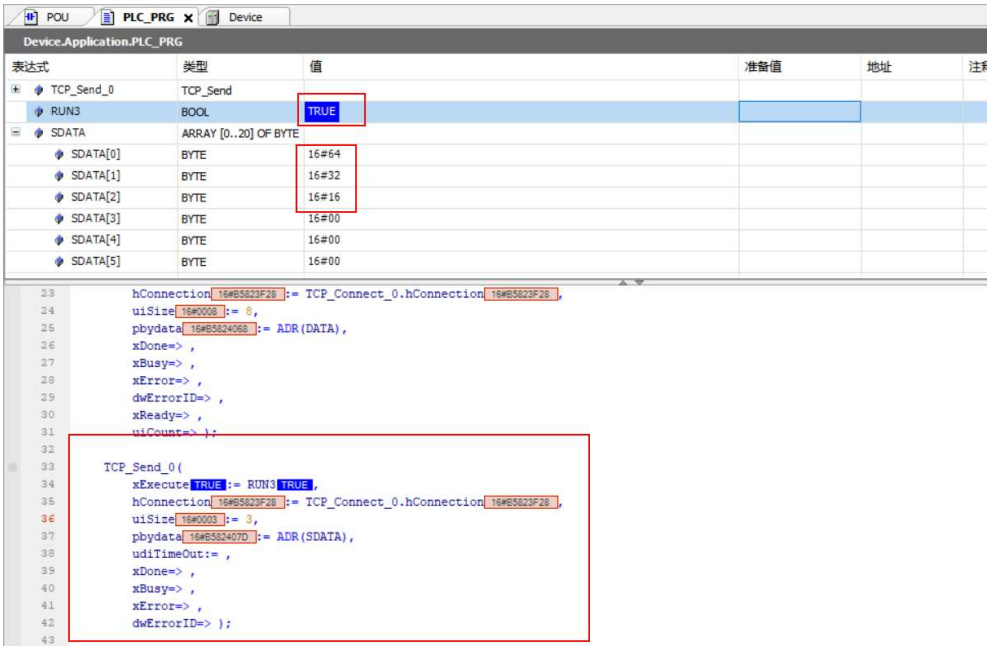
ST form

4. The server sends data to the client debugging assistant.

① Trigger the TCP_Send command to send data stored at the address pointed to by the pbyData pointer, as shown in the following figure. Send data 16#64 16#32 16#16.



LD form



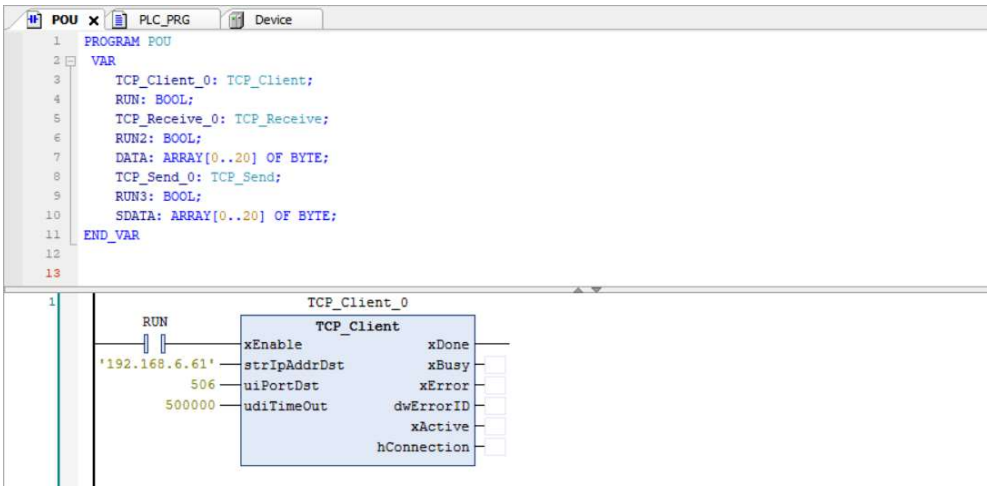
ST form

② The following figure shows the client debugging assistant receiving 16#64 16#32 16#16 data sent by the PLC server.

XSDH-60A32 as the client, debugging assistant as the server

1. Create a TCP client communication service and establish a connection with the debugging assistant.

① Trigger the TCP_Client command, and note that the IP address and port number are both the remote server address and port (the debugging assistant IP '192.168.6.1' is the IP address of the PC port connected to the PLC); Open the debugging assistant server before triggering the command.



LD form

```
PROGRAM PLC_PRG
VAR
    TCP_Client_0: TCP_Client;
    RUN: BOOL;
    TCP_Receive_0: TCP_Receive;
    RUN2: BOOL;
    DATA: ARRAY[0..20] OF BYTE;
    TCP_Send_0: TCP_Send;
    RUN3: BOOL;
    SDATA: ARRAY[0..20] OF BYTE;
END_VAR
```

```
TCP_Client_0(
    xEnable:= RUN,
    strIpAddrDst:= '192.168.6.61',
    uiPortDst:= 506,
    udiTimeOut:= 500000,
    xDone=> ,
    xBusy=> ,
    xError=> ,
    dwErrorID=> ,
    xActive=> ,
    hConnection=> );
```

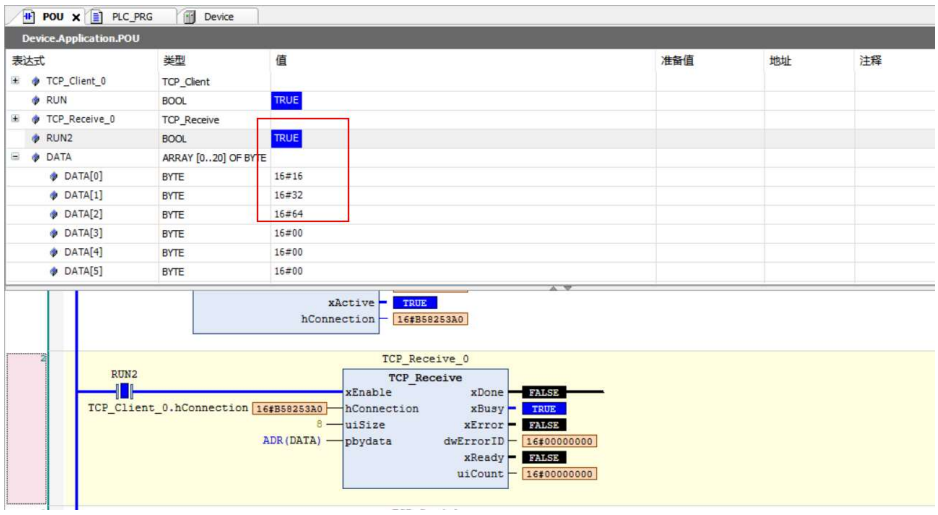
ST form

2. First, open the debugging assistant server, and then trigger the TCP_Client command mentioned above to create a client communication service. The connection has been successfully established as shown in the following figure:

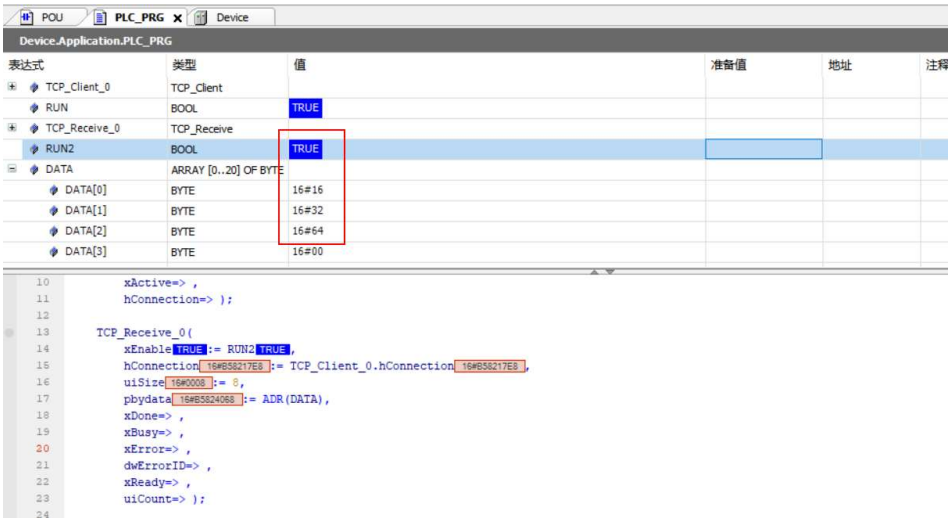
3. The client receives data sent by the server-side debugging assistant.

① Sending hexadecimal data 16#16 16#32 16#64 using the debugging assistant.

② Trigger the TCP_Receive instruction, and the received data is stored at the address pointed to by the pbyData pointer; the client received data 16#16 16#32 16#64 sent by the server, as shown in the following figure, and the reception was successful.



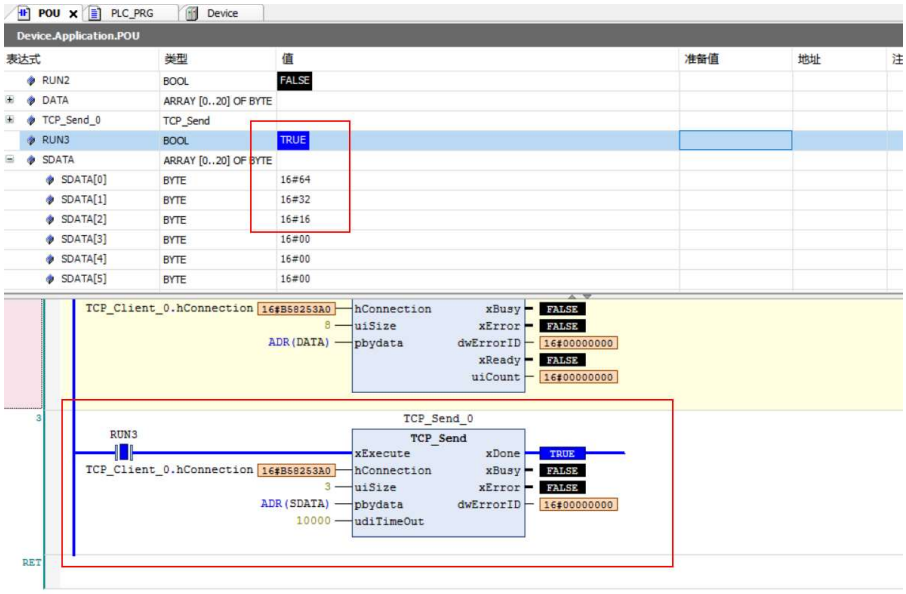
LD form



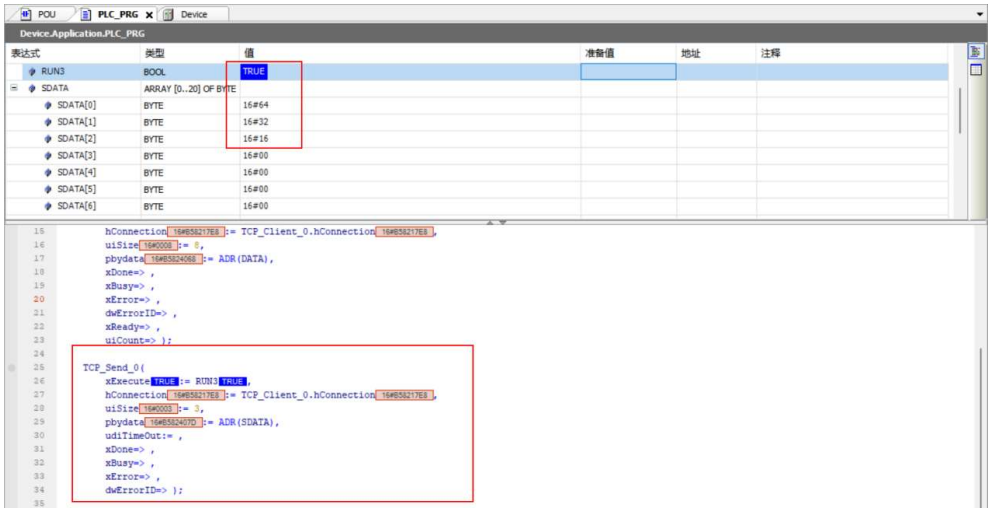
ST form

4. The client sends data to the server-side debugging assistant.

① Trigger the TCP_Send instruction to send data stored at the address pointed to by the pbyData pointer.



LD form



ST form

② The debugging assistant on the server side received data 16#64 16#32 16#16 sent by the PLC client, as shown in the following figure, and the transmission was successful.

TCP/UDP 网络调试助手
编码格式: UTF-8
通讯模式
TCP Client
TCP Server
UDP Mode
通讯参数
目的IP: 192.168.6.6
目的端口: 53236
本地端口: 506
关闭
自动覆盖
显示发送
接收字节数: 3
发送字节数: 9
保存文件
计数清零
接收清零
03:22:48 443 : [192.168.6.6:53236]: 64 32 16
自动换行
十六进制接收
显示时间
客户端列表: 192.168.6.6:53236
发送区设置
定时 100 ms
自动回复（发送框内容）
自动回复（发送方内容）
十六进制
发送后自加
自加起始字节 0
步长值 0
发送

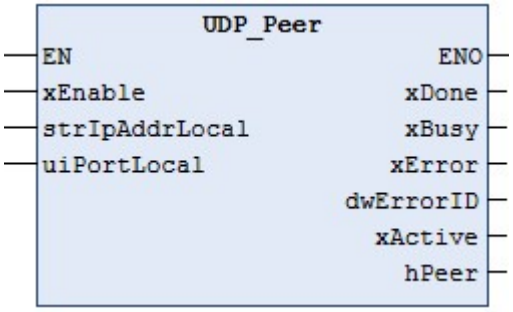
2-8-2. UDP communication

2-8-2-1. Create UDP communication connecting UDP_Peer

(1) Instruction overview

Create UDP communication connection.

Create UDP communication connection [UDP_Peer]			
Execution condition	Normally open/close triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graph	ST
UDP_Peer	Create UDP communication connection		UDP_Peer_0(xEnable:= , strIpAddrLocal:= , uiPortLocal:= , xDone=> , xBusy=> , xError=> , dwErrorID=> , xActive=> , hPeer=>);

(2) Related variables

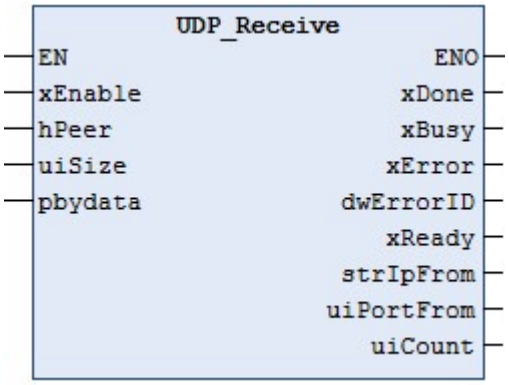
VAR_INPUT	Name	Data type	Valid range	Initial value	Description
xEnable	Command enable	BOOL	FALSE/TRUE	FALSE	Input high-level to start command
strIpAddrLocal	Local IP address	STRING(80)	As the data type	-	The IP address of the local PLC communication network port
uiPortLocal	Local port no.	UINT	As the data type	-	Port number for local communication
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
xDone	Termination Flag	BOOL	FALSE/TRUE	FALSE	Communication termination flag
xBusy	Busy flag	BOOL	FALSE/TRUE	FALSE	Busy flag
xError	Error flag	BOOL	FALSE/TRUE	FALSE	Error flag
dwErrorID	Error code	DWORD	As the data type	-	Error code
xActive	Communication success flag	BOOL	FALSE/TRUE	FALSE	Create UDP communication success flag
hpeer	Communication handle	_XWORD	-	-	UDP communication handle

2-8-2-2. UDP communication data reception UDP_Receive

(1) Instruction overview

UDP communication data reception.

UDP communication data reception [UDP_Receive]			
Execution condition	Normally open/close triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graph	ST
UDP_Receive	UDP communication data reception		UDP_Receive_0(xEnable:= , hPeer:= , uiSize:= , pbydata:= , xDone=> , xBusy=> , xError=> , dwErrorID=> , xReady=> , strIpFrom=> , uiPortFrom=> , uiCount=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
xEnable	Command enable	BOOL	FALSE/TRUE	FALSE	Input high-level to start command
hPeer	Communication handle	_XWORD	-	-	UDP communication handle
uiSize	Receive data size	UINT	As the data type	-	Receive data area size (BYTE)
pbyData	Point to receiving area	POINTER TO BYTE	-	-	Pointer to the beginning byte of the receiving area
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
xDone	Termination Flag	BOOL	FALSE/TRUE	FALSE	Data reception termination signal
xBusy	Busy flag	BOOL	FALSE/TRUE	FALSE	Busy status
xError	Error flag	BOOL	FALSE/TRUE	FALSE	Error status
dwErrorID	Error code	DWORD	As the data type	-	Error code
xReady	Received successfully flag	BOOL	FALSE/TRUE	FALSE	Read data from the buffer, and if the data is not empty, set the flag for one scan cycle
strIpFrom	Current packet	STRING(80)	As the data	-	Current packet source IP

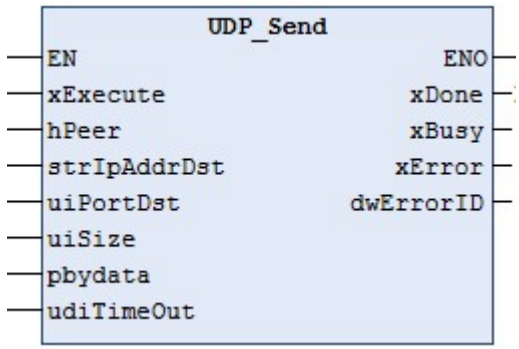
	source IP		type		
uiPortFrom	Current packet source port	UINT	As the data type	-	The source port of the current packet
uiCount	Number of data	_UXINT	-	-	Read the actual number of data from the buffer

2-8-2-3. UDP communication data send UDP_Send

(1) Instruction overview

UDP communication data send.

UDP communication data send [UDP_Send]			
Execution condition	Edge triggering	Suitable model	XS, XSA models
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Command	Name	Graph	ST
UDP_Send	UDP communication data send		UDP_Send_0(xExecute:= , hPeer:= , strIpAddrDst:= , uiPortDst:= , uiSize:= , pbydata:= , udiTimeOut:= , xDone=> , xBusy=> , xError=> , dwErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
xExecute	Command enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
hPeer	Communication handle	_XWORD	-	-	UDP communication handle
strIpAddrDst	Target IP	STRING(80)	As the data type	-	Target IP for sending data
uiPortDst	Target port	UINT	As the data type	-	Target port for sending data
uiSize	Send data size	UINT	As the data type	-	Send data area size (BYTE)
pbyData	Point to the sending area	POINTER TO BYTE	-	-	Pointer to the starting byte of the sending area
UditimeOut	Timeout time	UDINT	As the data	500000	Send data timeout, unit μs,

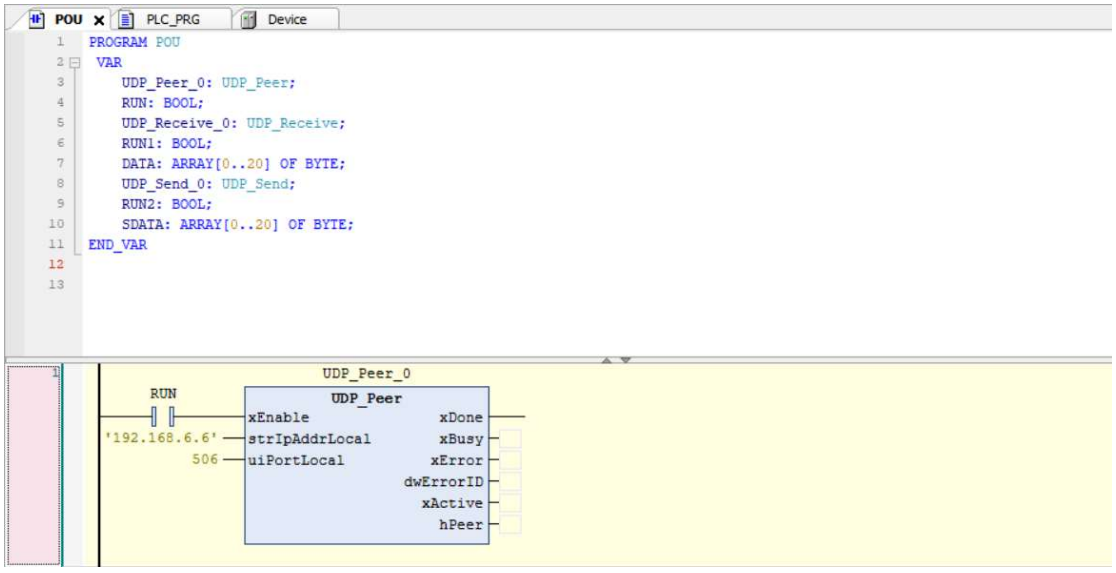
			type		default is 500ms
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
xDone	Completion signal	BOOL	FALSE/TRUE	FALSE	Data sending completion signal
xBusy	Busy flag	BOOL	FALSE/TRUE	FALSE	Busy status
xError	Error flag	BOOL	FALSE/TRUE	FALSE	Error status
dwErrorID	Error code	DWORD	As the data type	-	Error code

2-8-2-4. Example of UDP communication

XSDH-60A32 establishes UDP communication with computer debugging assistant

Wherein: debugging assistant (PC port) IP address 192.168.6.61, port number 506
PLC end IP address 192.168.6.6, port number 506

1. Establish a UDP connection between the PLC and the debugging assistant. In the UDP communication connection instruction UDP_Peer, the strIpAddrLocal input pin is the local PLC communication network port address, and the uiPortLocal input pin is the local PLC port number.



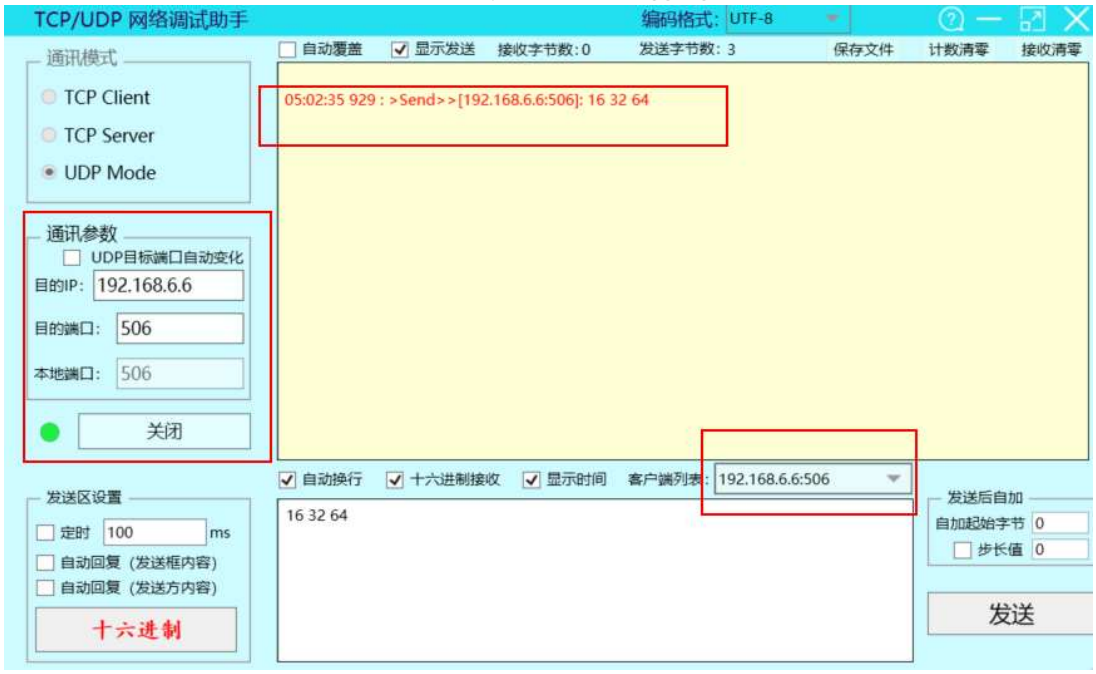
LD form

```
PROGRAM PLC_PRG
VAR
    UDP_Peer_0: UDP_Peer;
    RUN: BOOL;
    UDP_Receive_0: UDP_Receive;
    RUN1: BOOL;
    DATA: ARRAY[0..20] OF BYTE;
    UDP_Send_0: UDP_Send;
    RUN2: BOOL;
    SDATA: ARRAY[0..20] OF BYTE;
END_VAR
```

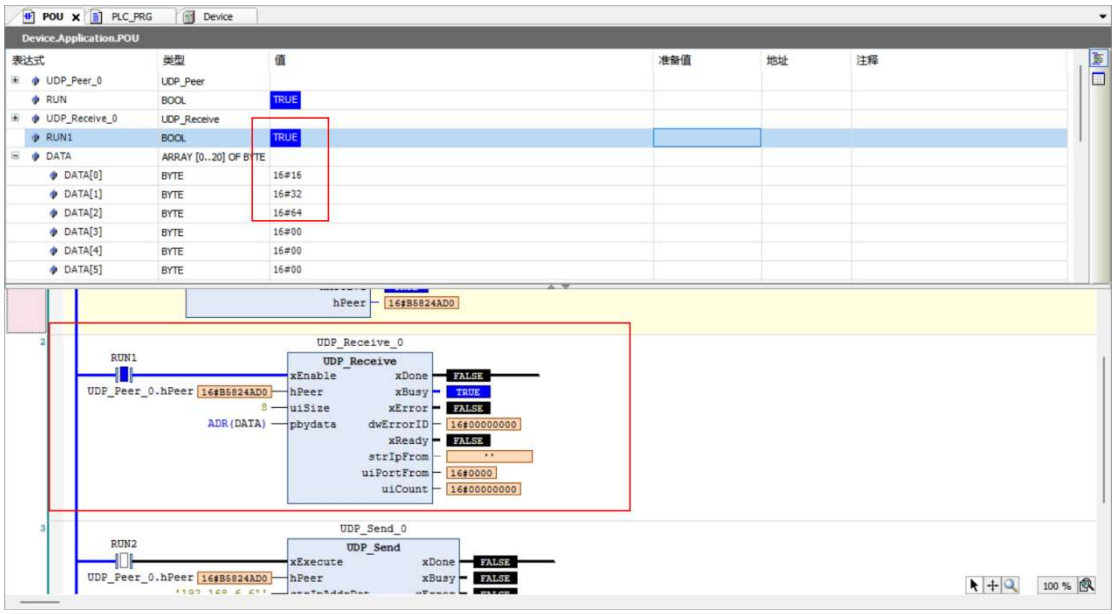
```
UDP_Peer_0(
    xEnable:= RUN,
    strIpAddrLocal:= '192.168.6.6',
    uiPortLocal:= 506,
    xDone=> ,
    xBusy=> ,
    xError=> ,
    dwErrorID=> ,
    xActive=> ,
    hPeer=> );
```

ST form

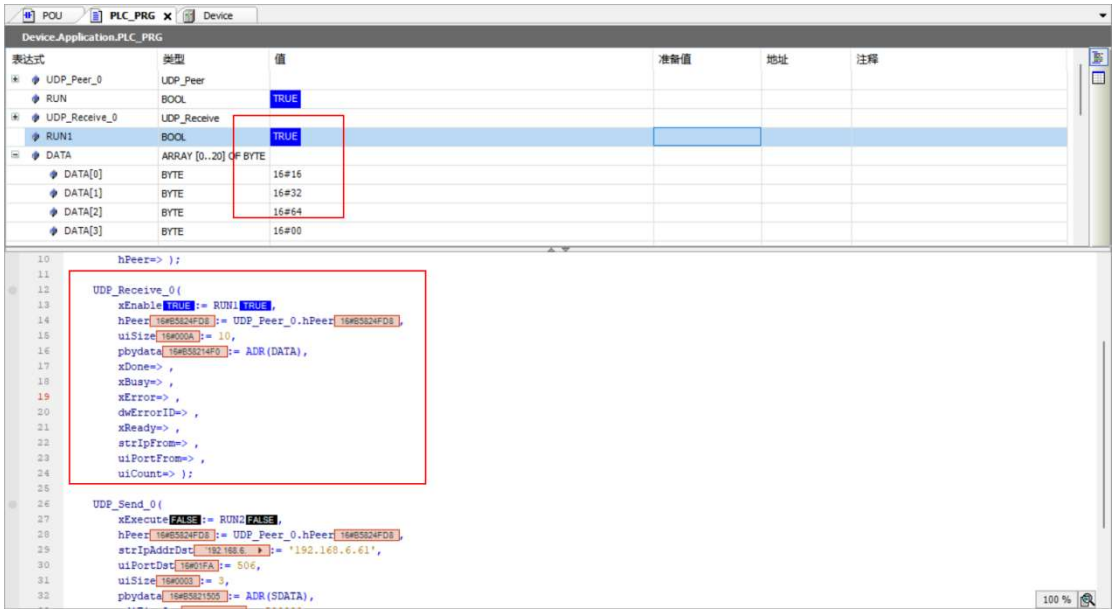
2. Receive data, the PLC receives data sent by the PC debugging assistant.



Triggering the UDP_Receive instruction, the received data is stored at the address pointed to by the pbyData pointer, as shown in the following figure. Received 16#16, 16#32, 16#64, and received successfully.



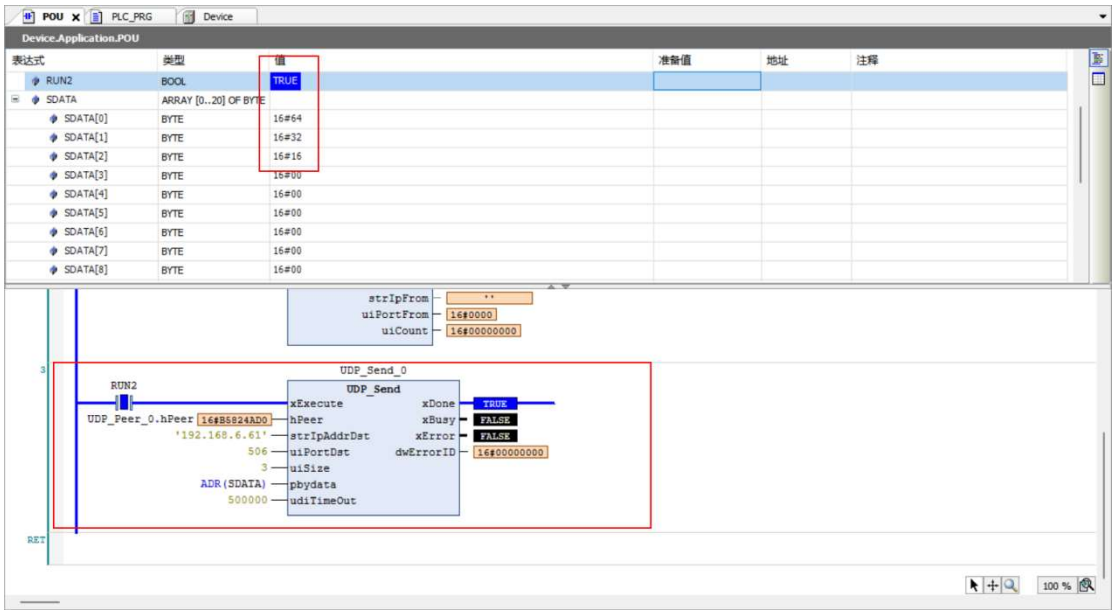
LD form



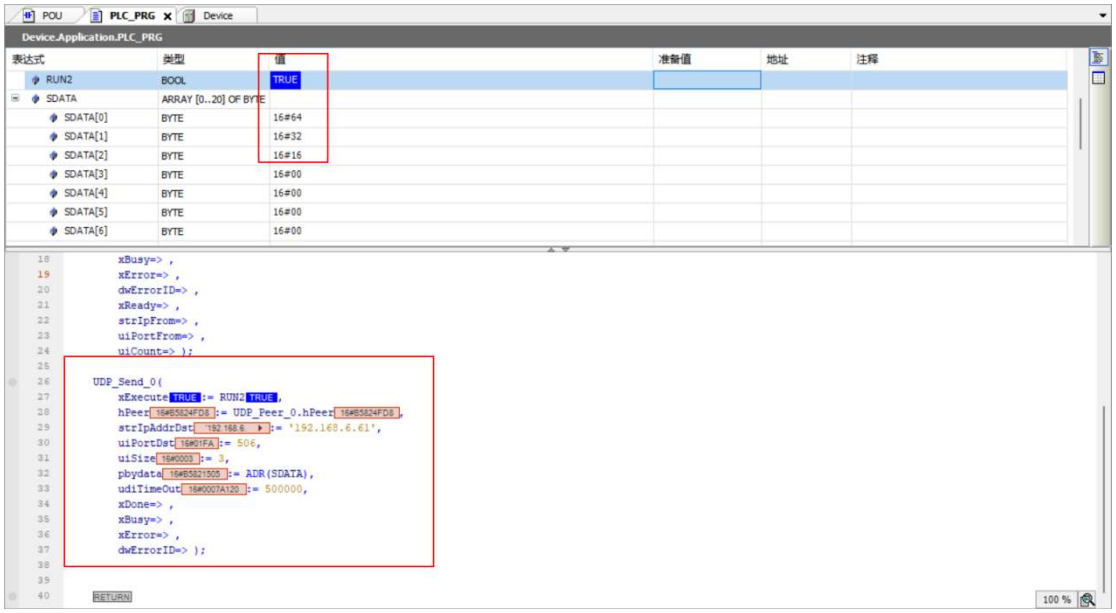
ST form

3. Send data, PLC sends data to PC debugging assistant.

Trigger the UDP_Send command to send data stored at the address pointed to by the pbyData pointer, as shown in the following figure. Send data 16#64 16#32 16#16.

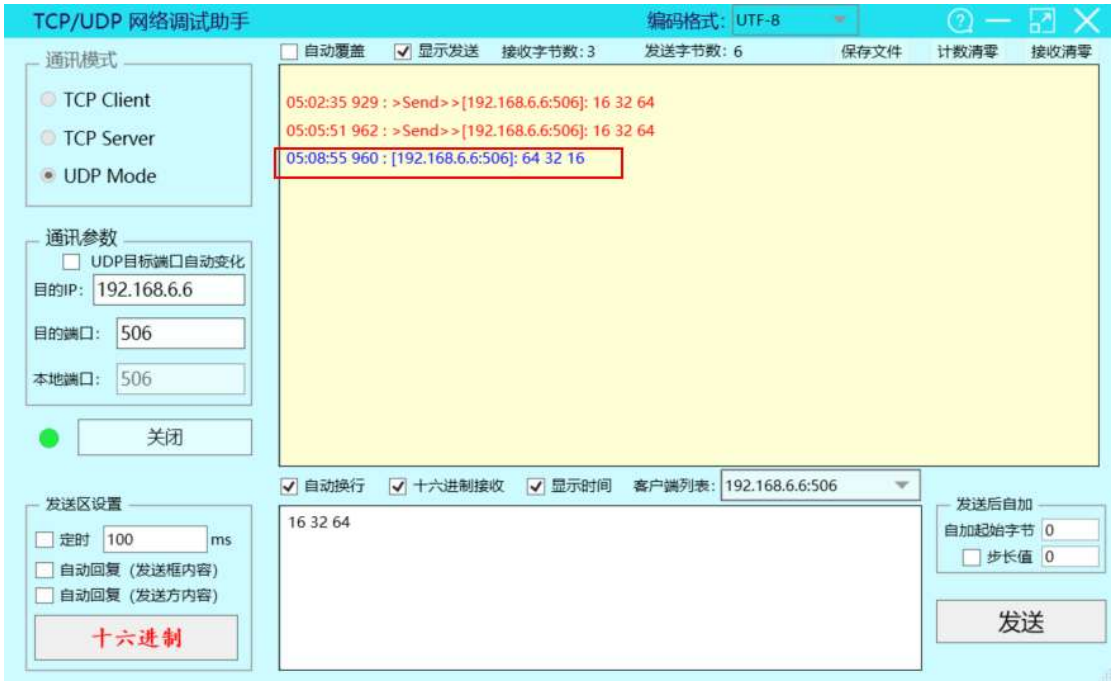


LD form



ST form

As shown in the figure below, the PC debugging assistant received data 16#64 16#32 16#16, and the PLC data was successfully sent.



2-8-3. TCP/UDP communication error code

Error ID	Error code	Description
NO_ERROR	0	No error
FIRST_ERROR	6000	Reserved
TIME_OUT	6001	Timeout
INVALID_ADDR	6002	IP invalid
INVALID_HANDLE	6003	Handle invalid
INVALID_DATAPOINTER	6004	Data pointer invalid
INVALID_DATASIZE	6005	Data length invalid
UDP_RECEIVE_ERROR	6006	UDP receive error
UDP_SEND_ERROR	6007	UDP send error
UDP_SEND_NOT_COMPLETE	6008	UDP sending not complete
UDP_OPEN_ERROR	6009	UDP open error
UDP_CLOSE_ERROR	6010	UDP close error
TCP_SEND_ERROR	6011	TCP send error
TCP_RECEIVE_ERROR	6012	TCP receive error
TCP_OPEN_ERROR	6013	TCP open error
TCP_CONNECT_ERROR	6014	TCP connection error
TCP_CLOSE_ERROR	6015	TCP close error
TCP_SERVER_ERROR	6016	TCP server error
WRONG_PARAMETER	6017	Parameter error
ERROR_UNKNOWN	6018	Unknown error
TCP_NO_CONNECTION	6019	TCP no connection
IOCTL_ERROR	6020	IOCTL error
FIRST_MF	6050	Reserved
LAST_ERROR	6099	Last time error

2-8-4. CAN free format communication

2-8-4-1. CAN free format communication specification

Item	CAN interface features
Connector	3-PIN terminal
Transmission cable	Two communication cables, one shielding cable, and one grounding cable
Supported standard protocols	CAN 2.0A (11-bit identifier) CAN 2.0B (29-bit identifier)
Baud rate	10,20,50,100,125,250,500,800,1000 kbit/s
Protocol	CANBUS 2.0A, CANBUS 2.0B

2-8-4-2. CAN free format send FreeCan_Send_CB

(1) Instruction overview

Write the contents of the specified register in this machine to the CAN device with the specified ID number.

CANBUS communication send data [FreeCan_Send_CB]			
Execution condition	Edge triggering	Suitable model	XSLH-30A32
Firmware	-	Software	-
Library	XJ_CANbus		

Command	Name	Graph	ST
FreeCan_Send_CB	CAN free format send	send FreeCan_Send_CB EN, Execute, CanID, SendBuff, SendDataLen, Port, Protocl ENO, Done, Busy, Error, ErrorID	FreeCan_Send_CB_0(Execute:= , CanID:= , SendBuff:= , SendDataLen:= , Port:= , Protocl:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
Execute	Edge triggering	BOOL	FALSE/TRUE	FALSE	Command conduction conditions
CanID	Specify remote communication ID number	UDINT	CAN2.0A：0-2047 CAN2.0B：0-536870911	0	Remote communication ID number
SendBuff	Specify the first address for sending variables	POINTER TO WORD	As the data type	-	The first address for sending variables

SendDataLen	Specify the number of bytes to send data	BYTE	As the data type	-	Bytes of data sent
Port	Specified CAN port	BYTE	As the data type	-	Communication CAN port
Protocl	Specified CANBUS protocol	BYTE	As the data type	-	CANBUS communication protocol
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
Done	Processing completion flag	BOOL	FALSE/TRUE	FALSE	Processing completion flag
Busy	Function block activation flag	BOOL	FALSE/TRUE	FALSE	Function block activation flag
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Error flag
ErrorID	Error code	XJ_FreeCan_ Error_Type	As the data type	-	Error code

(3) Function

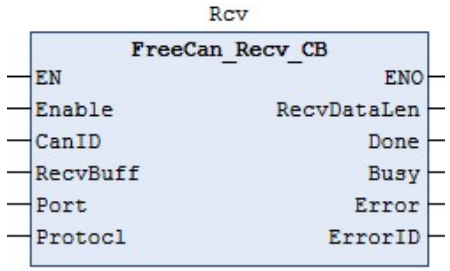
- Execute: edge triggered to send data.
- CanID: CAN2.0A: corresponds to the hexadecimal range H000- H7FF; CAN2.0B: corresponds to the hexadecimal range H0000 0000- H1FFF FFFF. It is necessary to limit the corresponding ID range based on the selected protocol type.
- SendBuf: specifies the first address of the variable for sending cached data.
- SendDataLen: The number of bytes sent is K0~K8, and the input byte count needs to be limited to a maximum of 8 bytes.
- Port: Communication CAN port range, K1: PLC first CAN channel, K2: PLC second CAN channel(reserved)
- Protocol: CAN communication protocol K0: CAN2.0A protocol, K1: CAN2.0B protocol.
- Done: instruction processing completion flag, set Done to true when sending is completed, and set FALSE when the falling edge of Execute is detected.
- Busy: Executing function block.
- Error: True: An error occurred, the function block terminated the operation, FALSE: No error.
- ErrorID: Error code.

2-8-4-3. CAN free format receive FreeCan_Recv_CB

(1) Instruction overview

The instruction to receive data from the CAN device with the specified ID number to the specified address within the local machine.

CANBUS communication receive data [FreeCan_Recv_CB]			
Execution condition	Normally open/close triggering	Suitable model	XSLH-30A32
Firmware	-	Software	-
Library	XJ_CANbus		

Command	Name	Graph	ST
FreeCan_Recv_CB	CAN free format receive		FreeCan_Recv_CB_0(Enable:= , CanID:= , RecvBuff:= , Port:= , Protocl:= , RecvDataLen=> , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
Enable	Command normally open/close coil conduction	BOOL	FALSE/TRUE	FALSE	Normally open/close coil conduction
CanID	Specify remote communication ID number	UDINT	As the data type	-	Remote communication ID number
RecvBuff	Specify the first address number for receiving data locally	POINTER TO WORD	As the data type	-	The first address number for receiving data locally
Port	Specify the CAN port	BYTE	As the data type	-	Communication CAN port
Protocl	Specify CANBUS protocol	BYTE	As the data type	-	CANBUS communication protocol
VAR_OUTPUT	Name	Data type	Valid range	Initial value	Description
RecvDataLen	The actual length of received data	BYTE	As the data type	-	Data reception termination signal
Done	Instruction	BOOL	FALSE/TRUE	FALSE	Instruction

VAR_INPUT	Name	Data type	Valid range	Initial value	Description
	processing completion flag				processing completion flag
Busy	Function block activation flag	BOOL	FALSE/TRUE	FALSE	Function block activation flag
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Error flag
ErrorID	Error code	XJ_FreeCan_Error_Type	As the data type	-	Error code

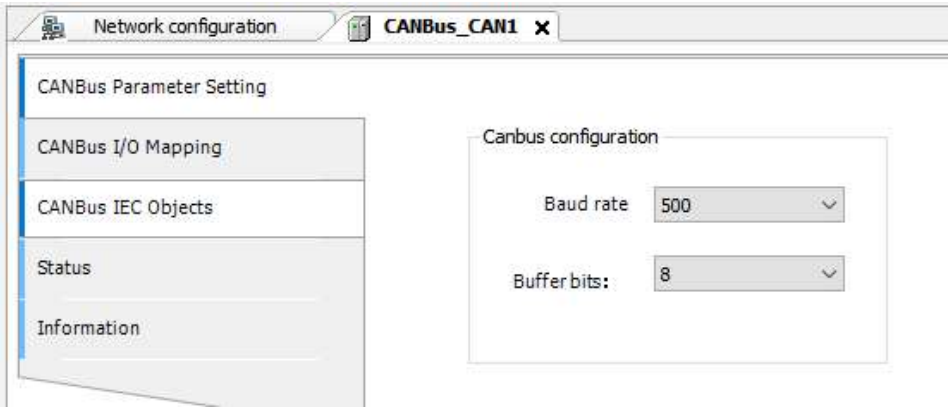
(3) Function

- Enable: Normally open/close coil waiting for reception.
- CanID: CAN2.0A: hexadecimal range H000- H7FF; CAN2.0B: Hexadecimal range H0000 0000- H1FFF FFFF. The ID number K0 serves as the identifier for enabling CAN free format broadcast reception under the CAN2.0A or CAN2.0B protocol. Other range ID numbers must comply with the standard receiving data of the sender's ID number and be limited according to the selected protocol type.
- RecvBuff: The first address of the received data byte.
- When the ID is 0, it is used as broadcast reception. CAN2.0A defaults to occupying 18 bytes after the specified receiving variable's first address. The first two bytes are used to store the broadcast reception ID number, and the last 16 bytes are used to receive data. CAN2.0B defaults to occupying the 20 bytes after the first address of the specified receiving variable. The first 4 bytes are used to store the ID number of broadcast reception, and the last 16 bytes are used to receive data.
- When receiving data with an ID other than 0, it defaults to occupying 16 bytes after the specified receiving address.
- Port: Communication CAN port range: K1, K2. K1: First CAN channel of PLC, K2: Second CAN channel of PLC (reserved)
- Protocol: CAN communication protocol K0: CAN2.0A protocol K1: CAN2.0B protocol.
- Done: Instruction processing completion flag, received data Done set to true.
- RecvDataLen: The actual length of data received.
- Busy: Function block activation.
- Error: True: An error occurred, the function block terminated the operation, FALSE: No error.
- ErrorID: Error code.

2-8-3-4. CAN free format communication example

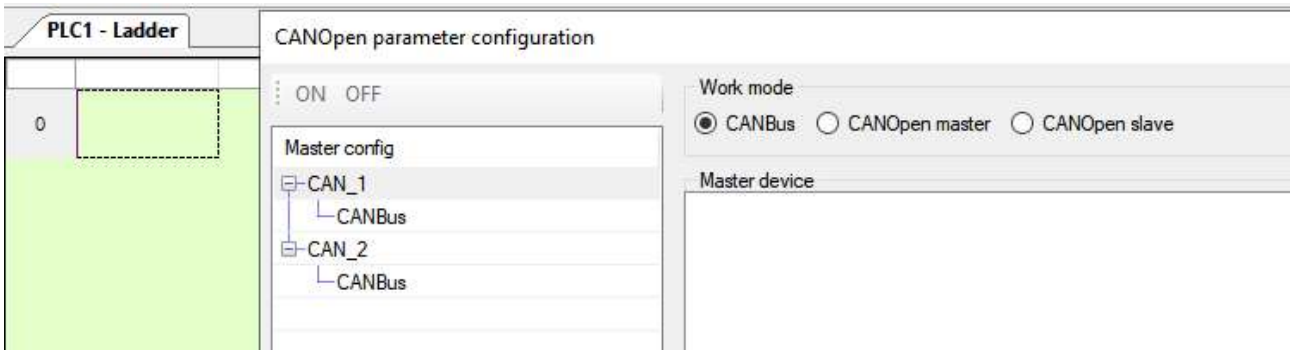
Establishing CAN free format communication between XSLH-30A32 and XL5N-32T

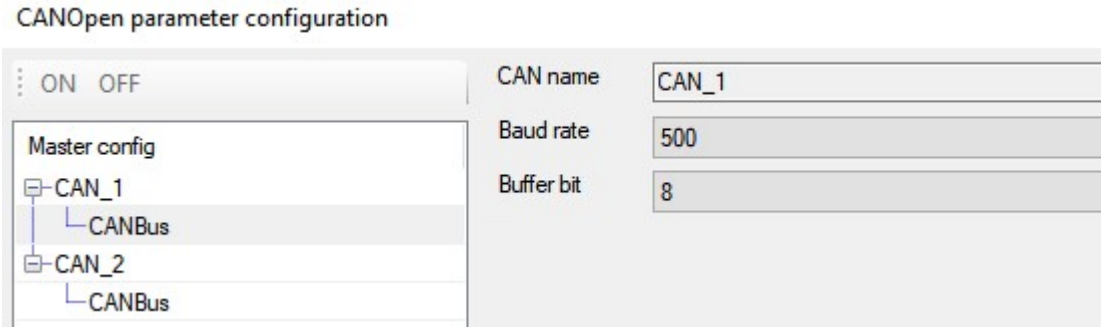
The default parameters for the CAN port of XSLH-30A32 are a baud rate of 500Kbit/s and 8 cache bits. The parameters can be modified on the configuration interface as shown in the following figure:

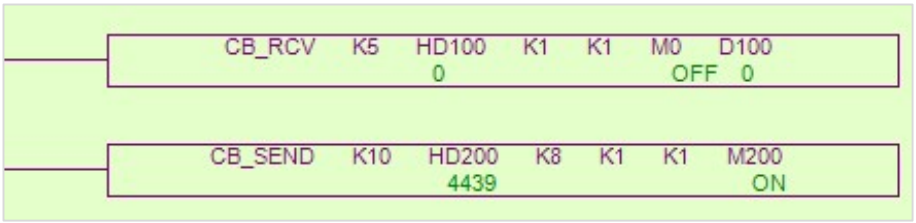


1. XSLH-30A32 is used to write FreeCan_Send_CB and FreeCan_Recv_CB instructions, configure the remote communication ID number, communication address, communication byte count, communication CAN port, and protocol for the two instructions. The sample program is shown in the following figure:

2. XL5N-32T is used to configure the CAN port protocol and baud rate, and then write the CB_SEND and CB_RCV commands. The configuration corresponding to the XSLH-30A32 command is written with the corresponding remote communication ID number, communication address, communication byte count, communication CAN port and protocol. The sample program is shown in the following figure:







3. After configuring the instructions and parameters, download and trigger the instructions to receive and receive data.

2-9. File processing

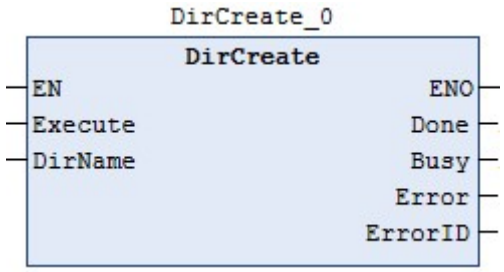
Due to the different file processing methods between Windows and Linux systems, there may be some differences in using the same file processing instruction on different operating systems.

2-9-1. Create directory DirCreate

(1) Instruction overview

Create a directory with the specified name in the SD card.

Create directory [DirCreate]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
DirCreate	Create directory		DirCreate_0(Execute:= , DirName:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Command enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
DirName	Specify directory name	STRING	Maximum 1985 characters	-	The directory name to be created
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

- Generate a directory with the specified name in "DirName" in the SD card.

(4) Note

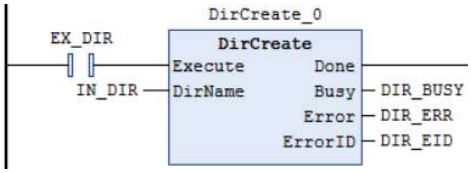
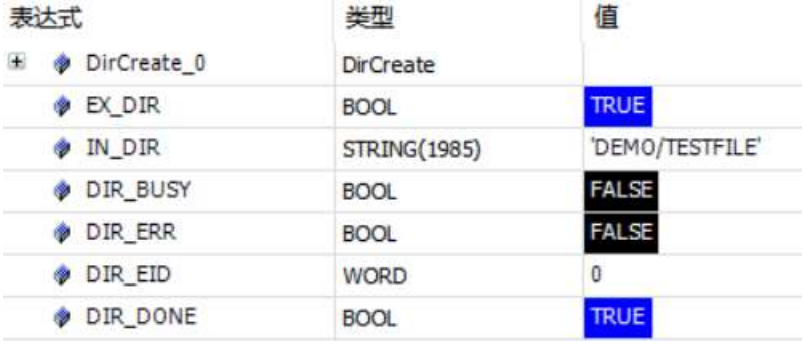
- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can

be confirmed by whether the value of "Done" has changed to true.

- When the SD storage card is unplugged while the file is open, the file will remain open. But when reinstalling the SD memory card, it will not be possible to read or write files. To read and write the file, please reopen the file.
- The directory name size specified in "DirName", with a maximum support of 255 characters for Linux systems with prefixes added, will be truncated if exceeded; The Windows system supports a maximum of 260 characters with prefix added, and an error will be reported if it exceeds this limit.
  - An exception will occur in the following situations. "Error" is true.

◆ When the SD card is not in a usable state.
◆ The SD card is write protected.
◆ When the remaining space of the SD card is insufficient.
◆ When the number of directories that can be created exceeds.
◆ When the directory specified in "DirName" already exists.
◆ When the value of "DirName" is an illegal directory name.
◆ When accessing the SD card and unable to access due to an exception.

(5) Sample program

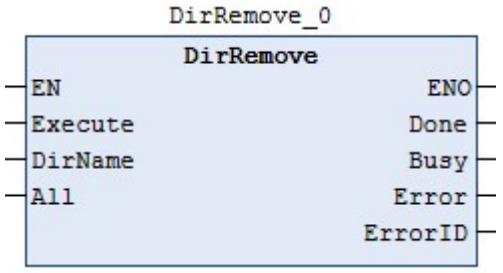
DEMO	LD	ST
Define the variables	VAR DirCreate_0: DirCreate; EX_DIR: BOOL; IN_DIR: STRING(1985):= 'DEMO/TESTFILE'; DIR_BUSY: BOOL; DIR_ERR: BOOL; DIR_EID: WORD; DIR_DONE: BOOL; END_VAR	
Sample program		DirCreate_0(Execute:= EX_DIR, DirName:= IN_DIR, Done=> DIR_DONE, Busy=> DIR_BUSY, Error=> DIR_ERR, ErrorID=> DIR_EID);
Operation result		

2-9-2. Delete directory DirRemove

(1) Instuction overview

Delete the specified directory in the SD card.

Delete directory [DirRemove]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
DirRemove	Delete directory		DirRemove_0(Execute:= , DirName:= , All:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
DirName	Delete directory name	STRING	Maximum 1985 characters	-	Delete directory name
All	Select all	BOOL	FALSE/TRUE	FALSE	Operation when there are files/subdirectories in the directory True: Delete FALSE: Do not delete
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

- Delete the specified directory in the SD storage card specified in the directory name "DirName".
- If there are files or subdirectories in the directory specified by "DirName", depending on the value of "All" selected, the process is as follows.

“All” value	Operation
TRUE	Delete specified directory, including files and entire subdirectories
FALSE	Not deleting specified directory, exception occurred

(4) Note

- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle,

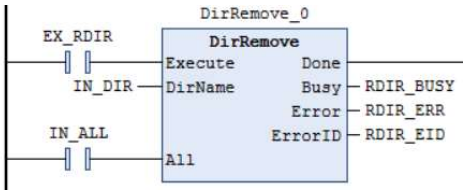
this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.

- When the SD storage card is unplugged while the file is open, the file will remain open. But when reinstalling the SD memory card, it will not be possible to read or write files. To read and write the file, please reopen the file.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - The SD storage card is write protected.
  - When the value of "All" is true and the directory specified in "DirName" is being accessed through other instructions.
  - When the value of "All" is FALSE and there are files or directories in the directory specified in "DirName".
  - When the directory specified in "DirName" is read-only.
  - When the file specified in "DirName" does not exist.
  - When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

DEMO	LD	ST
Define the variables		
Sample program		
Operation result		

```
VAR
    DirRemove_0: DirRemove;
    EX_RDIR: BOOL;
    IN_DIR: STRING(1985):= 'DEMO/TESTFILE';
    IN_ALL: BOOL;
    RDIR_BUSY: BOOL;
    RDIR_ERR: BOOL;
    RDIR_EID: WORD;
    RDIR_DONE: BOOL;
END_VAR
```




```
DirRemove_0(
    Execute:= EX_RDIR,
    DirName:= IN_DIR,
    All:= IN_ALL,
    Done=> RDIR_DONE,
    Busy=> RDIR_BUSY,
    Error=> RDIR_ERR,
    ErrorID=> RDIR_EID);
```

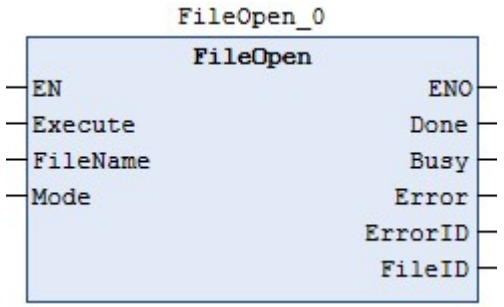
表达式	类型	值
DirRemove_0	DirRemove	
EX_RDIR	BOOL	TRUE
IN_DIR	STRING(1985)	'DEMO/TESTFILE'
IN_ALL	BOOL	TRUE
RDIR_BUSY	BOOL	FALSE
RDIR_ERR	BOOL	FALSE
RDIR_EID	WORD	0
RDIR_DONE	BOOL	TRUE

2-9-3. Open file FileOpen

(1) Instruction overview

Open the specified file in the SD storage card.

Open file [FileOpen]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileOpen	Open file		FileOpen_0(Execute:= , FileName:= , Mode:= , Done=> , Busy=> , Error=> , ErrorID=> , FileID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileName	Specify a filename	STRING	Maximum 1985 characters	-	The file name to be opened
Mode	Open mode	ACCESS_MODE	As the data type	-	Open file mode
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID
FileID	File ID	DWORD	As the data type	-	The file ID of the opened file

(3) Function

- Open the file in the SD storage card specified in "FileName" using the mode specified in "Mode".
- As the result of opening the file, the file ID "FileID" will be output. "FileID" can be used in the FileRead and FileWrite instructions to specify a file.
- The data type of "Mode" is enumeration type ACCESS-MODE. The meaning of enumeration values is as follows:

Enum	Meaning
AM_READ	If the target path file exists, open the existing file as read-only. If it does not exist, return an error value
AM_WRITE	If the target path file exists, open the file by rewriting it. If it does not exist,

	a new file with write permission will be automatically created
AM_APPEND	If the target path file exists, open the existing file in a write only manner and append the content to the end of the file. If it does not exist, return an error value
AM_READ_PLUS	If the target path file exists, open the existing file for reading and writing. If it does not exist, return an error value
AM_WRITE_PLUS	If the target path file exists, open the file as read-write. If it does not exist, a new file with read/write permissions will be automatically created
AM_APPEND_PLUS	If the target path file exists, open the existing file as read-write and append the content to the end of the file. If it does not exist, a new file with read/write permissions will be automatically created

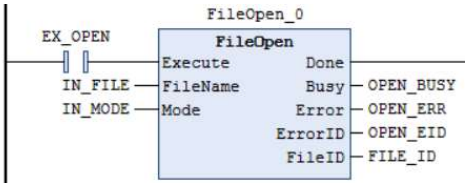
(4) Note

- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- This instruction needs to be executed before the FileSeek instruction, FileWrite instruction, FileRead instruction, FilePuts instruction, and FileGets instruction.
- For files opened with this command, please execute the FileClose command to close the file.
- When this instruction is completed, save the value in "FileID". When the value of "Done" changes from FALSE to True.
- When the SD storage card is unplugged while the file is open, the file will remain open. But when reinstalling the SD memory card, it will not be possible to read or write files. To read and write the file,
- A file opened in the form of AM_WRITE, when using SysFileRead in the Linus system to obtain character length, will display an error message with an error ID of 1; In the Windows system, using SysFileRead to obtain character length will display an error message with an error ID of 2.
- A file opened in AM_APPEND format, when using SysFileRead in the Linus system to obtain character length, will display an error message with an error ID of 1; Using SysFileRead to obtain character length in Windows system can execute normally.
- Files opened through SysFileOpen can be deleted under the RUNTIME path in the Linus system, but cannot be deleted under the RUNTIME path in the Windows system due to permission issues.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - The SD storage card is write protected.
  - When the value of "Mode" is AM_READ, AM_APPEND or AM_READ_PLUS, and the file specified in "FileName" does not exist.
  - When the value of "FileName" is an illegal file name.
  - When the number of files and directories that can be created exceeds.
  - When accessing the file specified in "FileName".
  - When the file specified in "FileName" is read-only.
  - When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define the variables</td><td colspan="2">

```
VAR
   FileOpen_0: FileOpen;
   EX_OPEN: BOOL;
   IN_FILE: STRING(1985):= 'DEMO/TESTFILE/VERSION.TXT';
   IN_MODE: XJ_FunctionUtils.ACCESS_MODE:= XJ_FunctionUtils.ACCESS_MODE.AM_WRITE_PLUS;
   OPEN_DONE: BOOL;
   OPEN_BUSY: BOOL;
   OPEN_ERR: BOOL;
   OPEN_EID: WORD;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td> 
</td><td>

```
FileOpen_0(
    Execute:= EX_OPEN,
    FileName:= IN_DIR,
    Mode:= IN_MODE,
    Done=> OPEN_DONE,
    Busy=> OPEN_BUSY,
    Error=> OPEN_ERR,
    ErrorID=> OPEN_EID,
    FileID=> FILE_ID);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
FileOpen_0	FileOpen	
EX_OPEN	BOOL	TRUE
IN_FILE	STRING(1985)	'DEMO/TESTFILE/VERSION.TXT'
IN_MODE	ACCESS_MODE	AM_WRITE_PLUS
OPEN_DONE	BOOL	TRUE
OPEN_BUSY	BOOL	FALSE
OPEN_ERR	BOOL	FALSE
OPEN_EID	WORD	0
FILE_ID	DWORD	3040906744

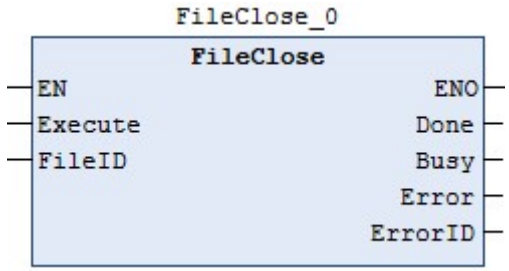
</td></tr>
</table>

2-9-4. Close the file FileClose

(1) Instruction overview

Close the specified file in the SD storage card.

Close the file [FileClose]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileClose	Close the file		FileClose_0(Execute:= , FileID:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileID	File ID	_XWORD(DWORD)	As the data type	-	The file ID of the file to be closed
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

Close the files in the SD storage card specified in "FileID".

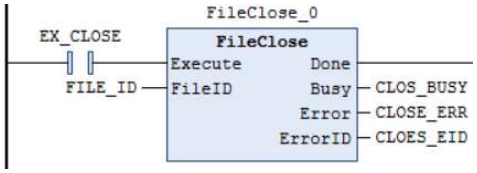
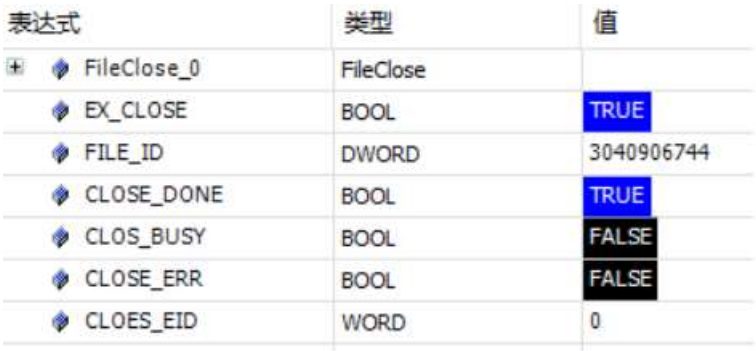
(4) Notes

- It is prohibited to execute this command multiple times for the same file, otherwise it may cause abnormal operation of the PLC and even crash.
- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- "FileID" needs to be obtained by executing the FileOpen command in advance.
- For files opened with the FileOpen command, be sure to execute this command and close the file after use, otherwise unexpected results may occur.
- When the SD storage card is unplugged while the file is open, the file will remain open. But when

reinstalling the SD memory card, it will not be possible to read or write files. To read and write the file, please reopen the file.

- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - When accessing the file specified in "FileID".
  - When the file specified in "FileID" does not exist.
  - When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

DEMO	LD	ST
Define the variables		VAR FileClose_0: FileClose; EX_CLOSE: BOOL; FILE_ID: __XWORD:= 3040906744; CLOSE_DONE: BOOL; CLOS_BUSY: BOOL; CLOSE_ERR: BOOL; CLOES_EID: WORD; END_VAR
Sample program		FileClose_0(Execute:= EX_CLOSE, FileID:= FILE_ID, Done=> CLOSE_DONE, Busy=> CLOS_BUSY, Error=> CLOSE_ERR, ErrorID=> CLOES_EID);
Operation result		

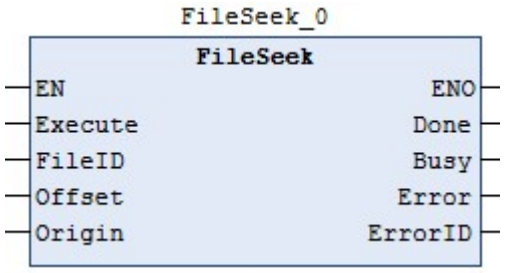
表达式	类型	值
FileClose_0	FileClose	
EX_CLOSE	BOOL	TRUE
FILE_ID	DWORD	3040906744
CLOSE_DONE	BOOL	TRUE
CLOS_BUSY	BOOL	FALSE
CLOSE_ERR	BOOL	FALSE
CLOES_EID	WORD	0

2-9-5. Find the file FileSeek

(1) Instruction overview

Set the file location indicator in the specified file on the SD storage card.

Find the file [FileSeek]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileSeek	Find the file		FileSeek_0(Execute:= , FileID:= , Offset:= , Origin:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileID	File ID	_XWORD(DWORD)	As the data type	-	The file ID of the file where the file location indicator is to be set
Offset	Offset	DINT	As the data type	-	The offset position of the file location indicator relative to "Origin"
Origin	Refrence location	_eFSEEK_ORIGIN	As the data type	_eFSEEK_ORIGIN._SEEK_SET	The reference position of the file location indicator
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

- Set the file location indicator in the SD storage card file specified in the file ID "FileID". The file location indicator refers to the location in the file where reading or writing begins when executing file data read and write instructions. For example, when reading from the beginning of a file, the FileSeek command should be used to set the file location indicator at the beginning of the file, and then the FileRead command should be executed.

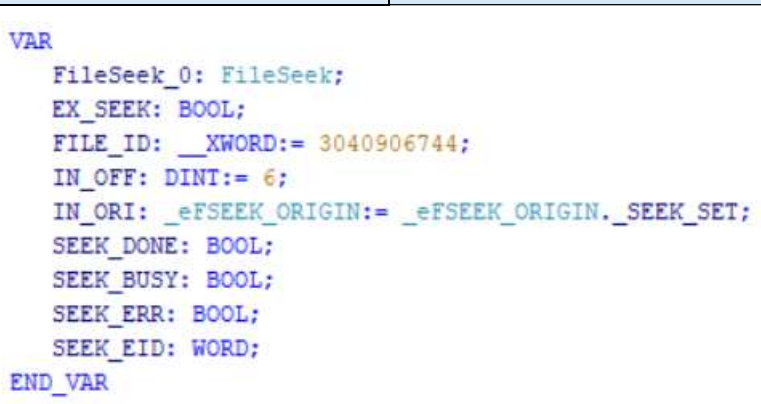
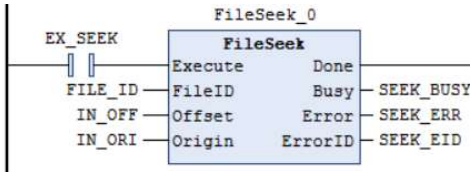
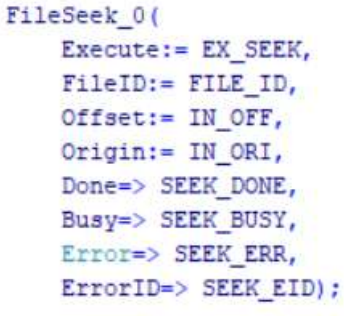
- Set the position of the file location indicator as the reference position "Origin" plus the offset "Offset".
- The data type of "Origin" is enumeration _eFSEEK_ORIGIN. The meaning of enumeration values is as follows:

Enum	Meaning
_SEEK_SET	At the beginning of the file
_SEEK_CUR	The location of the current file location indicator
_SEEK_END	At the end of the file

(4) Notes

- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- "FileID" needs to be obtained by executing the FileOpen command in advance.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - When accessing the file specified in "FileID".
  - When the file specified in "FileID" does not exist.
  - When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

DEMO	LD	ST
Define the variables		
Sample program		

```
VAR
    FileSeek_0: FileSeek;
    EX_SEEK: BOOL;
    FILE_ID: __XWORD:= 3040906744;
    IN_OFF: DINT:= 6;
    IN_ORI: _eFSEEK_ORIGIN:= _eFSEEK_ORIGIN._SEEK_SET;
    SEEK_DONE: BOOL;
    SEEK_BUSY: BOOL;
    SEEK_ERR: BOOL;
    SEEK_EID: WORD;
END_VAR
```



```
FileSeek_0(
    Execute:= EX_SEEK,
    FileID:= FILE_ID,
    Offset:= IN_OFF,
    Origin:= IN_ORI,
    Done=> SEEK_DONE,
    Busy=> SEEK_BUSY,
    Error=> SEEK_ERR,
    ErrorID=> SEEK_EID);
```

Operation result	

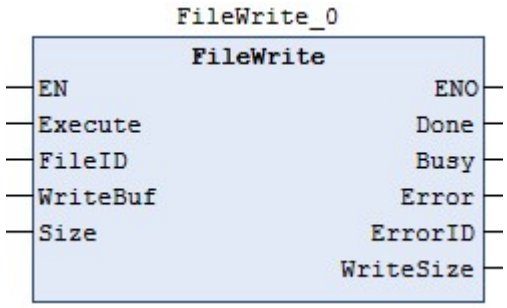
表达式	类型	值
FileSeek_0	FileSeek	
EX_SEEK	BOOL	TRUE
FILE_ID	DWORD	3040906744
IN_OFF	DINT	6
IN_ORI	_EFSEEK_ORIGIN	_SEEK_SET
SEEK_DONE	BOOL	TRUE
SEEK_BUSY	BOOL	FALSE
SEEK_ERR	BOOL	FALSE
SEEK_EID	WORD	0

2-9-6. Write in the file FileWrite

(1) Instruction overview

Write the data to the specified file on the SD storage card.

Write in the file[FileWrite]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileWrite	Write in the file		FileWrite_0(Execute:= , FileID:= , WriteBuf:= , Size:= , Done=> , Busy=> , Error=> , ErrorID=> , WriteSize=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileID	File ID	_XWORD(DWORD)	As the data type	-	The file ID to write data to
Size	Write in element numbers	UINT	As the data type	-	Number of elements to be written
WriteBuf	Write cache array	BOOL/ BYTE/WORD/DWORD/LWORD / USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/ REAL/LREAL/ TIME/DATE/TOD/DT/STRING	As the data type	-	The write object of the written data
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

WriteSize	Actual number of elements written	UINT	As the data type	-	Actual number of elements written

(3) Function

- For the SD storage card file specified in the file ID "FileID", write the data written to the cache WriteBuf[ ] to the location indicated by the file location indicator.
- The file location indicator should be set to any location in advance through the FileSeek command.
- The amount of data written is (WriteBuf [ ] data type size) × "Size".
- The actual number of elements written is saved in "WriteSize". Usually, the value of "Size" is consistent with the value of "ReadSize".

(4) Note

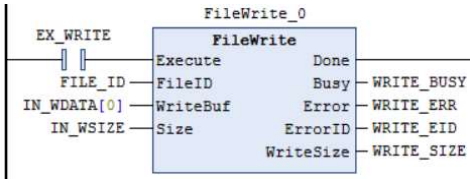
- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- When the value of "Size" exceeds the array area of WriteBuf[], it is necessary to avoid PLC exceptions.
- "FileID" needs to be obtained by executing the FileOpen command in advance.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - When accessing the file specified in "FileID".
  - When the file specified in "FileID" does not exist.
  - When accessing the SD memory card and unable to access due to an exception.
  - When the file specified in "FileID" is not opened in writable mode.
  - ReadBuf[] passes in the entire array, and the ErrorID displays error code "16#20".

(5) Sample program

DEMO	LD	ST
Define the variables		

```
VAR
   FileWrite_0: FileWrite;
   EX_WRITE: BOOL;
   FILE_ID: __XWORD:= 3040906744;
   IN_WDATA: ARRAY[0..2] OF BYTE:= [16,32,64];
   IN_WSIZE: UINT:= 3;
   WRITE_DONE: BOOL;
   WRITE_BUSY: BOOL;
   WRITE_ERR: BOOL;
   WRITE_EID: WORD;
   WRITE_SIZE: UINT;
END_VAR
```

Sample program




```
FileWrite_0(
    Execute:= EX_WRITE,
    FileID:= FILE_ID,
    WriteBuf:= IN_WDATA[0],
    Size:= IN_WSIZE,
    Done=> WRITE_DONE,
    Busy=> WRITE_BUSY,
    Error=> WRITE_ERR,
    ErrorID=> WRITE_EID,
    WriteSize=> WRITE_SIZE);
```

Operation result

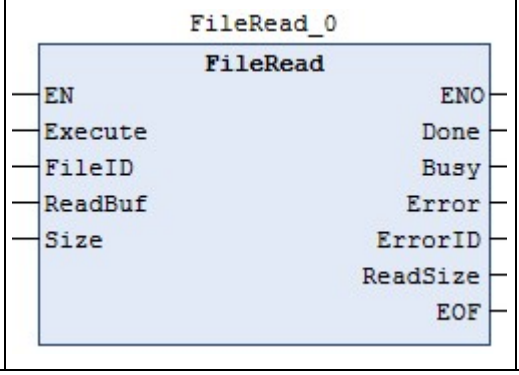
表达式	类型	值
⊞ FileWrite_0	FileWrite	
EX_WRITE	BOOL	TRUE
FILE_ID	DWORD	3040906744
⊟ IN_WDATA	ARRAY [0..2] OF BYTE	
IN_WDATA[0]	BYTE	16
IN_WDATA[1]	BYTE	32
IN_WDATA[2]	BYTE	64
IN_WSIZE	UINT	3
WRITE_DONE	BOOL	TRUE
WRITE_BUSY	BOOL	FALSE
WRITE_ERR	BOOL	FALSE
WRITE_EID	WORD	0
WRITE_SIZE	UINT	3

2-9-7. Read the file FileRead

(1) Instruction overview

Read the data of the specified file from the SD storage card.

Read the file [FileRead]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileRead	Read the file		FileRead_0(Execute:= , FileID:= , ReadBuf:= , Size:= , Done=> , Busy=> , Error=> , ErrorID=> , ReadSize=> , EOF=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileID	File ID	_XWORD(DWORD)	As the data type	-	The file ID to read data from
Size	Read number of elements	UINT	As the data type	-	Number of elements to be read
ReadBuf	Read cache array	BOOL/ BYTE/WORD/DWORD/LWORD / USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/ REAL/LREAL/ TIME/DATE/TOD/DT/STRING	As the data type	-	The write object of the read data
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

ReadSize	Actual read number of elements	UINT	As the data type	-	Actual read number of elements
EOF	File end flag	BOOL	FALSE/TRUE	FALSE	Have you reached the end of the file True: Arrived FALE: Not arrived

(3) Function

- For the SD storage card file specified in the file ID "FileID", start reading data from the location indicated by the file location indicator and save it to the read cache ReadBuf[].
- The file location indicator should be set to any location in advance through the FileSeek command.
- The amount of data read is (ReadBuf[] data type size) × "Size".
- The actual number of elements read is saved in "ReadSize". Usually, the value of "Size" is consistent with the value of "ReadSize". If the amount of data from the location of the file location indicator to the end of the file is less than "Size", no exception will occur, and the data up to the end of the file will be saved to ReadBuf[]. At this point, the value of "ReadSize" becomes smaller than the value of "Size".
- In addition, when reading to the end of the file, the value of "EOF" at the end of the file becomes true. If not, the value of "EOF" is FALSE.

(4) Note

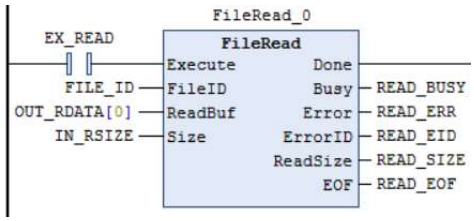
- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- When reading data up to the end of the file and the data size cannot be fully divided by the data type size of ReadBuf[], data with less than the data size of ReadBuf[] will also be processed. The file location indicator will come to the end of the file, and the value of "EOF" will become true.
- When the value of "Size" exceeds the array area of ReadBuf[], it is necessary to avoid PLC exceptions. The element after "Size" in ReadBuf[] (which will not be overwritten by reading) remains the value before executing this instruction.
- "FileID" needs to be obtained by executing the FileOpen command in advance.
- When this instruction is completed, save the value in "EOF". When the value of "Done" changes from FALSE to True.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - When accessing the file specified in "FileID".
  - When the file specified in "FileID" does not exist.
  - When accessing the SD memory card and unable to access due to an exception.
  - When the file specified in "FileID" is not opened in readable mode.
  - ReadBuf[] passes in the entire array, and the ErrorID displays error code "16#20".

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define the variables</td><td colspan="2">

```
VAR
   FileRead_0: FileRead;
   EX_READ: BOOL;
   FILE_ID: __XWORD:= 3040906744;
   OUT_RDATA: ARRAY[0..2] OF BYTE;
   IN_RSIZE: UINT:= 3;
   READ_DONE: BOOL;
   READ_BUSY: BOOL;
   READ_ERR: BOOL;
   READ_EID: WORD;
   READ_SIZE: UINT;
   READ_EOF: BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
FileRead_0(
    Execute:= EX_READ,
    FileID:= FILE_ID,
    ReadBuf:= OUT_RDATA,
    Size:= IN_RSIZE,
    Done=> READ_DONE,
    Busy=> READ_BUSY,
    Error=> READ_ERR,
    ErrorID=> READ_EID,
    ReadSize=> READ_SIZE,
    EOF=> READ_EOF);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
⊞ FileRead_0	FileRead	
EX_READ	BOOL	TRUE
FILE_ID	DWORD	3040906744
⊟ OUT_RDATA	ARRAY [0..2] OF BYTE	
OUT_RDATA[0]	BYTE	16
OUT_RDATA[1]	BYTE	32
OUT_RDATA[2]	BYTE	64
IN_RSIZE	UINT	3
READ_DONE	BOOL	TRUE
READ_BUSY	BOOL	FALSE
READ_ERR	BOOL	FALSE
READ_EID	WORD	0
READ_SIZE	UINT	3
READ_EOF	BOOL	FALSE

</td></tr>
</table>

2-9-8. Write string FilePuts

(1) Instruction overview

Write the string into the specified file on the SD storage card.

Write string [FilePuts]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FilePuts	Write string	FilePuts_0 FilePuts EN — ENO Execute — Done FileID — Busy In — Error ErrorID	FilePuts_0(Execute:= , FileID:= , In:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileID	File ID	_XWORD(DWORD)	As the data type	-	The file ID to write data to
In	The string to be written	STRING	As the data type	-	The string to be written
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

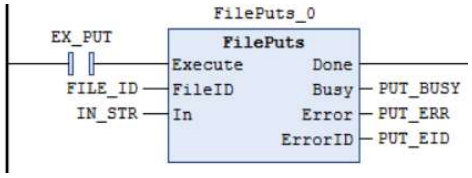
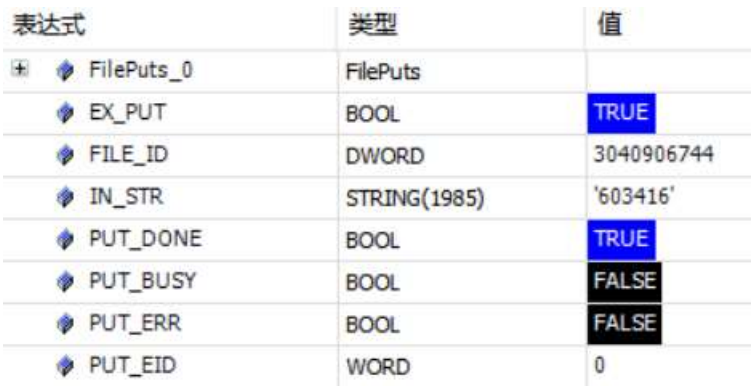
- Write the string "In" to the location indicated by the file location indicator for the SD storage card specified in the file ID "FileID".
- The file location indicator should be set to any location in advance through the FileSeek command.

(4) Note

- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- "FileID" needs to be obtained by executing the FileOpen command in advance.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.

- The SD storage card is write protected.
- When the remaining space of the SD storage card is insufficient.
- When accessing the file specified in "FileID".
- When the file specified in "FileID" does not exist.
- When accessing the SD memory card and unable to access due to an exception.
- When the file specified in "FileID" is not opened in writable mode.

(5) Sample program

DEMO	LD	ST
Define the variables	VAR FilePuts_0: FilePuts; EX_PUT: BOOL; FILE_ID: __XWORD:= 3040906744; IN_STR: STRING(1985):= '603416'; PUT_DONE: BOOL; PUT_BUSY: BOOL; PUT_ERR: BOOL; PUT_EID: WORD; END_VAR	
Sample program		FilePuts_0(Execute:= EX_PUT, FileID:= FILE_ID, In:= IN_STR, Done=> PUT_DONE, Busy=> PUT_BUSY, Error=> PUT_ERR, ErrorID=> PUT_EID);
Operation result		

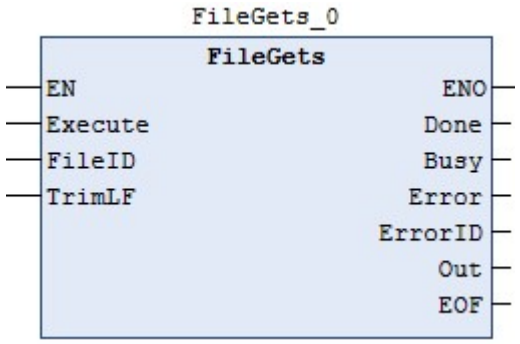
表达式	类型	值
FilePuts_0	FilePuts	
EX_PUT	BOOL	TRUE
FILE_ID	DWORD	3040906744
IN_STR	STRING(1985)	'603416'
PUT_DONE	BOOL	TRUE
PUT_BUSY	BOOL	FALSE
PUT_ERR	BOOL	FALSE
PUT_EID	WORD	0

2-9-9. Read string FileGets

(1) Instruction overview

Read 1 line of string from the specified file on the SD storage card.

Read string [FileGets]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileGets	Read string		FileGets_0(Execute:= , FileID:= , TrimLF:= , Done=> , Busy=> , Error=> , ErrorID=> , Out=> , EOF=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileID	File ID	_XWORD(DWORD)	As the data type	-	The file ID to read data from
TrimLF	Specify deletion Line break code	BOOL	FALSE/TRUE	FALSE	Specify deletion line break code TRUE: delete FALSE: not delete
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID
Out	Read string	STRING	As the data type	-	Read string
EOF	File end flag	BOOL	FALSE/TRUE	FALSE	Reached the end of the file True: Arrived FALE: Not arrived

(3) Function

- Read one line of string from the location of the file location indicator for the SD storage card file specified in the file ID "FileID".
- The file location indicator should be set to any location in advance through the FileSeek command.
- The separation between rows is recognized based on the newline code. Output the read string to the read string "Out".
- Line breaks are automatically recognized from three types of CR, LF, and CR+LF. When the value of TrimLF, which specifies the deletion of the line break code, is set to true, it will be written to "Out" after deleting the line break code in the string.
- When reading to the end of the file, the value of the "EOF" at the end of the file flag becomes true.

(4) Note

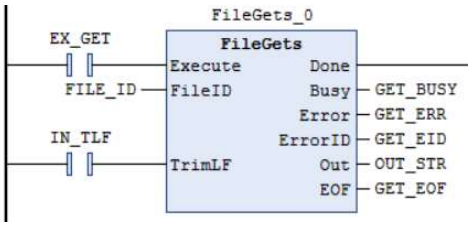
- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- When the length of a string exceeds 1985 bytes, the 1985 byte and its preceding string, as well as the last NULL character, will be saved to "Out".
- "FileID" needs to be obtained by executing the FileOpen command in advance.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - When accessing the file specified in "FileID".
  - When the file specified in "FileID" does not exist.
  - When accessing the SD memory card and unable to access due to an exception.
  - When the file specified in "FileID" is not opened in readable mode.

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define the variables</td><td colspan="2">

```
VAR
    FileGets_0: FileGets;
    EX_GET: BOOL;
    FILE_ID: __XWORD:= 3040906744;
    IN_TLF: BOOL;
    GET_DONE: BOOL;
    GET_BUSY: BOOL;
    GET_ERR: BOOL;
    GET_EID: WORD;
    OUT_STR: STRING(1985);
    GET_EOF: BOOL;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
FileGets_0(
    Execute:= EX_GET,
    FileID:= FILE_ID,
    TrimLF:= IN_TLF,
    Done=> GET_DONE,
    Busy=> GET_BUSY,
    Error=> GET_ERR,
    ErrorID=> GET_EID,
    Out=> OUT_STR,
    EOF=> GET_EOF);
```

</td></tr>
</table>

Operation result	

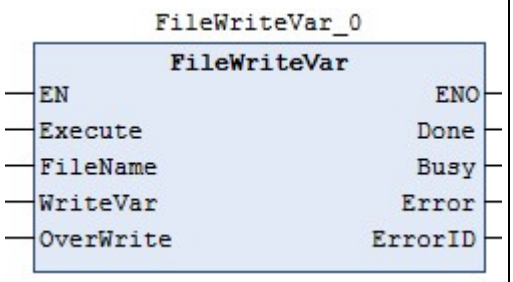
表达式	类型	值
FileGets_0	FileGets	
EX_GET	BOOL	TRUE
FILE_ID	DWORD	3040906744
IN_TLF	BOOL	FALSE
GET_DONE	BOOL	TRUE
GET_BUSY	BOOL	FALSE
GET_ERR	BOOL	FALSE
GET_EID	WORD	0
OUT_STR	STRING(1985)	'416603416603416'
GET_EOF	BOOL	TRUE

2-9-10. Write variable file FileWriteVar

(1) Instruction overview

Write the value of one variable in binary form into the specified file on the SD storage card.

Write variable file [FileWriteVar]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileWriteVar	Write variable file		FileWriteVar_0(Execute:= , FileName:= , WriteVar:= , OverWrite:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileName	Specified file name	STRING	Maximum 1985 characters	-	The file name of the file to write variable data to
WriteVar	Specify the variables to be written	BOOL/ BYTE/WORD/DWORD/LWORD/ USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/ REAL/LREAL/ TIME/DATE/TOD/DT/STRING	As the data type	-	Variable to be written
OverWrite	Overlay selection	BOOL	FALSE/TRUE	FALSE	True: Allow overwrite FALSE: Do not overwrite
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

- Write a variable "WriteVar" in binary form to the file of the SD storage card specified in the file name "FileName".
- When there is no file with the same name as "FileName" in the SD storage card, a new file will be created.
- If a file with the same name as "FileName" already exists in the SD storage card, depending on the value allowed to overwrite "OverWrite", the following steps will be taken:

“OverWrite” value	Operation
TRUE	Overwrite this file
FALSE	Not overwriting the file, an exception occurred

(4) Note

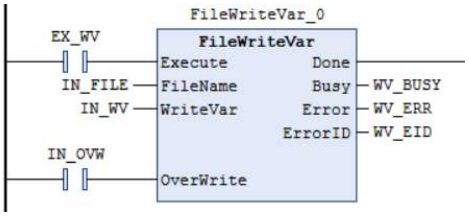
- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- "FileID" needs to be obtained by executing the FileOpen command in advance.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - The SD storage card is write protected.
  - When the remaining space of the SD storage card is insufficient.
  - When accessing the file specified in "FileID".
  - When the value of "FileName" is an illegal file name.
  - When the number of files and directories that can be created exceeds.
  - When a file with the same name as "FileName" already exists and is being accessed.
  - When a file with the same name as "FileName" already exists and the value of "OverWrite" is FALSE.
  - When a file with the same name as "FileName" already exists and is read-only.
  - When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

DEMO	LD	ST
Define the variables		

```
VAR
   FileWriteVar_0: FileWriteVar;
   EX_WV: BOOL;
   IN_FILE: STRING(1985):= 'DEMO/TESTFILE/VERSION.TXT';
   IN_WV: ARRAY[0..2] OF BYTE:= [64,32,16];
   IN_OVW: BOOL;
   WV_DONE: BOOL;
   WV_BUSY: BOOL;
   WV_ERR: BOOL;
   WV_EID: WORD;
END_VAR
```

Sample program




```
FileWriteVar_0(
    Execute:= EX_WV,
    FileName:= IN_FILE,
    WriteVar:= IN_WV,
    OverWrite:= IN_OVW,
    Done=> WV_DONE,
    Busy=> WV_BUSY,
    Error=> WV_ERR,
    ErrorID=> WV_EID);
```

Operation result

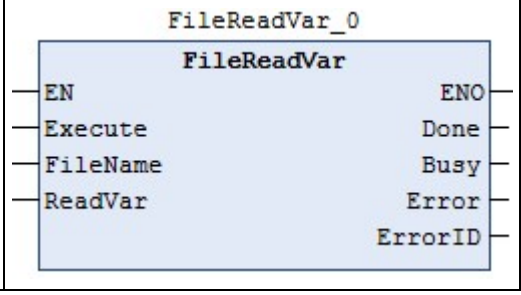
表达式	类型	值
FileWriteVar_0	FileWriteVar	
EX_WV	BOOL	TRUE
IN_FILE	STRING(1985)	'DEMO/TESTFILE/VERSION.TXT'
IN_WV	ARRAY [0..2] OF BYTE	
IN_WV[0]	BYTE	64
IN_WV[1]	BYTE	32
IN_WV[2]	BYTE	16
IN_OVW	BOOL	TRUE
WV_DONE	BOOL	TRUE
WV_BUSY	BOOL	FALSE
WV_ERR	BOOL	FALSE
WV_EID	WORD	0

2-9-11. Read variable file FileReadVar

(1) Instruction overview

Read the value of the specified file from the SD memory card in binary form and write it into a variable.

Read variable file [FileReadVar]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileReadVar	Read variable file		FileReadVar_0(Execute:= , FileName:= , ReadVar:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileName	The specified file name	STRING	Maximum 1985 characters	-	The file name of the file to read variable data from
ReadVar	Specify reading variables	BOOL/ BYTE/WORD/DWORD/LWORD / USINT/UINT/UDINT/ULINT/ SINT/INT/DINT/LINT/ REAL/LREAL/ TIME/DATE/TOD/DT/STRING	As the data type	-	Save read data variables
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

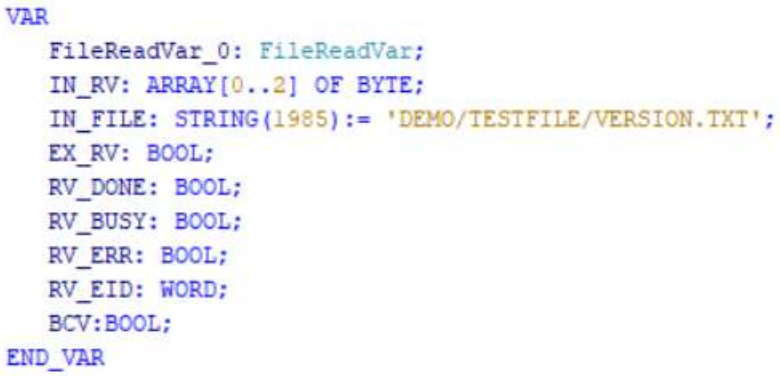
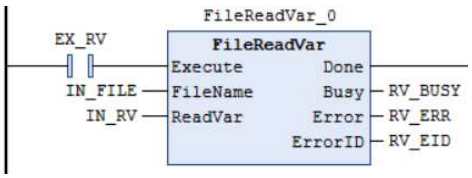
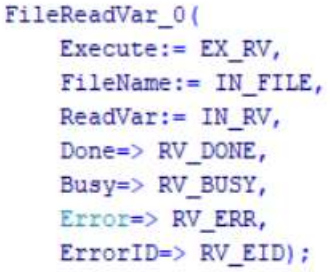
(3) Function

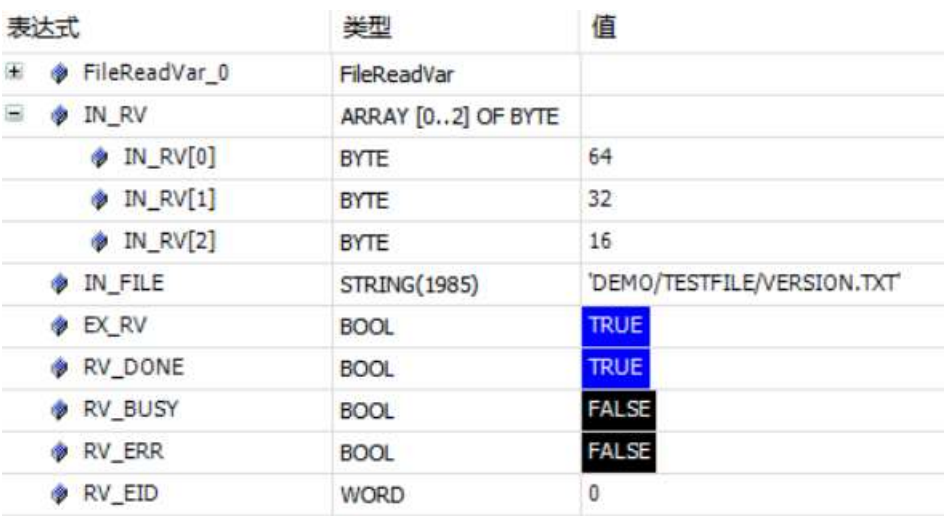
- Read variable data in binary form for the SD storage card file specified in the file name "FileName", and substitute the read value into the read location variable "ReadVar".

(4) Note

- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- When the size of the specified file is greater than the size of "ReadVar", no exception will occur, and only the data corresponding to the size of "ReadVar" will be read.
- When the size of the specified file is less than the size of "ReadVar", no exception will occur, and only the data corresponding to the size of the specified file will be read. The remaining area of "ReadVar" will maintain its value before executing this instruction.
- When "ReadVar" is the entire structure, adjustment areas may be inserted between members depending on the specific structure.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - When the value of "FileName" is an illegal file name.
  - When the file specified in "FileName" does not exist.
  - When accessing the file specified in "FileName".
  - When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

DEMO	LD	ST
Define the variables		
Sample program		

Operation result	

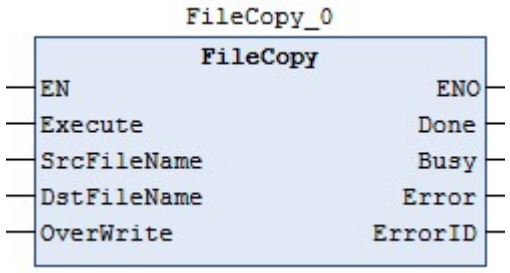
表达式	类型	值
FileReadVar_0	FileReadVar	
IN_RV	ARRAY [0..2] OF BYTE	
IN_RV[0]	BYTE	64
IN_RV[1]	BYTE	32
IN_RV[2]	BYTE	16
IN_FILE	STRING(1985)	'DEMO/TESTFILE/VERSION.TXT'
EX_RV	BOOL	TRUE
RV_DONE	BOOL	TRUE
RV_BUSY	BOOL	FALSE
RV_ERR	BOOL	FALSE
RV_EID	WORD	0

2-9-12. Copy file FileCopy

(1) Instruction overview

Copy the specified file from the SD storage card.

Copy file [FileCopy]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileCopy	Copy files		FileCopy_0(Execute:= , SrcFileName:= , DstFileName:= , OverWrite:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
StcFileName	Copy source file name	STRING	Maximum 1985 characters	-	Copy source file name
DstFileName	The target file name for pasting	STRING	Maximum 1985 characters	-	The target file name for pasting
OverWrite	Overlay selection	BOOL	FALSE/TRUE	FALSE	True: Allow overwrite FALSE: Do not overwrite
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

- Copy the file from the SD storage card specified in the source file name "SrcFileName" and paste it into the destination file name "DstFileName".
- If there is no file with the same name as "DstFileName" in the SD storage card, a new file will be created.
- If a file with the same name as "DstFileName" already exists in the SD storage card, the following steps will be taken based on the allowed value of "OverWrite" to be overwritten:

“OverWrite” value	Operation
TRUE	Overwrite this file
FALSE	Not overwriting file, exception occurred

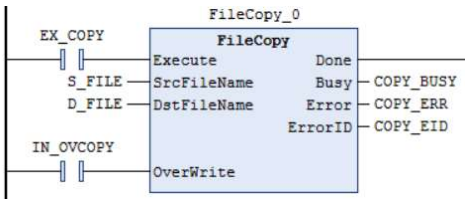
(4) Note

- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- When the SD storage card is unplugged while the file is open, the file will remain open. But when reinstalling the SD memory card, it will not be possible to read or write files. To read and write the file, please reopen the file.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - The SD storage card is write protected.
  - When the remaining space of the SD storage card is insufficient.
  - When the file specified in "SrcFileName" does not exist.
  - When the value of "SrcFileName" is an illegal file name.
  - When the value of "DstFileName" is an illegal file name.
  - When the number of files and directories that can be created exceeds.
  - When a file with the same name as "DstFileName" already exists and is being accessed.
  - When a file with the same name as "DstFileName" already exists and the value of "OverWrite" is FALSE.
  - When a file with the same name as "DstFileName" already exists and is read-only.
  - When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define the variables</td><td colspan="2">

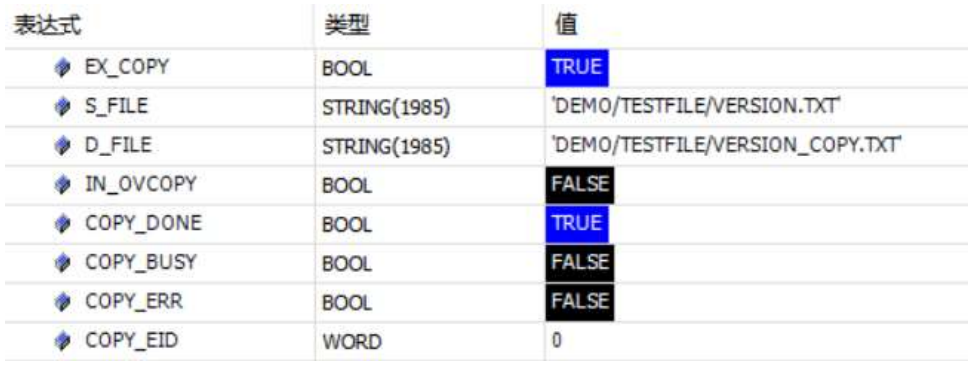
```
VAR
    FileCopy_0: FileCopy;
    EX_COPY: BOOL;
    S_FILE: STRING(1985):= 'DEMO/TESTFILE/VERSION.TXT';
    D_FILE: STRING(1985):= 'DEMO/TESTFILE/VERSION_COPY.TXT';
    IN_OVCOPY: BOOL;
    COPY_DONE: BOOL;
    COPY_BUSY: BOOL;
    COPY_ERR: BOOL;
    COPY_EID: WORD;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>


</td><td>

```
FileCopy_0(
    Execute:= EX_COPY,
    SrcFileName:= S_FILE,
    DstFileName:= D_FILE,
    OverWrite:= IN_OVCOPY,
    Done=> COPY_DONE,
    Busy=> COPY_BUSY,
    Error=> COPY_ERR,
    ErrorID=> COPY_EID);
```

</td></tr>
</table>

Operation result	

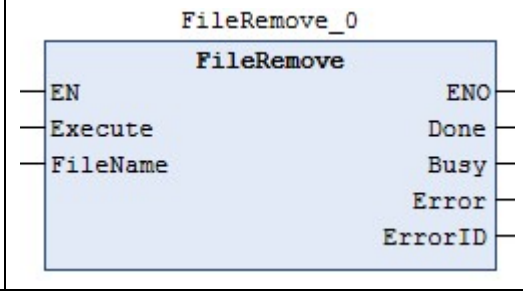
表达式	类型	值
EX_COPY	BOOL	TRUE
S_FILE	STRING(1985)	'DEMO/TESTFILE/VERSION.TXT'
D_FILE	STRING(1985)	'DEMO/TESTFILE/VERSION_COPY.TXT'
IN_OVCOPY	BOOL	FALSE
COPY_DONE	BOOL	TRUE
COPY_BUSY	BOOL	FALSE
COPY_ERR	BOOL	FALSE
COPY_EID	WORD	0

2-9-13. Delete file FileRemove

(1) Instruction overview

Delete the specified file from the SD storage card.

Delete file [FileRemove]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileRemove	Delete file		FileRemove_0(Execute:= , FileName:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileName	The specified file name	STRING	Maximum 1985 characters	-	The file name to be deleted
VAR_OUTPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

- Delete the file from the SD storage card specified in the specified file name "FileName".

(4) Note

- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- When the SD storage card is unplugged while the file is open, the file will remain open. But when reinstalling the SD memory card, it will not be possible to read or write files. To read and write the file, please reopen the file.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - The SD storage card is write protected.
  - When the file specified in "FileName" does not exist.
  - When accessing the file specified in "FileName".
  - When a file with the same name as "FileName" already exists and is read-only.

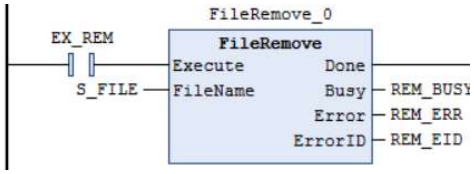
- When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define the variables</td><td colspan="2">

```
VAR
    FileRemove_0: FileRemove;
    EX_REM: BOOL;
    S_FILE: STRING(1985):= 'DEMO/TESTFILE/VERSION.TXT';
    REM_DONE: BOOL;
    REM_BUSY: BOOL;
    REM_ERR: BOOL;
    REM_EID: WORD;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
FileRemove_0(
    Execute:= EX_REM,
    FileName:= S_FILE,
    Done=> REM_DONE,
    Busy=> REM_BUSY,
    Error=> REM_ERR,
    ErrorID=> REM_EID);
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

表达式	类型	值
FileRemove_0	FileRemove	
EX_REM	BOOL	TRUE
S_FILE	STRING(1985)	'DEMO/TESTFILE/VERSION.TXT'
REM_DONE	BOOL	TRUE
REM_BUSY	BOOL	FALSE
REM_ERR	BOOL	FALSE
REM_EID	WORD	0

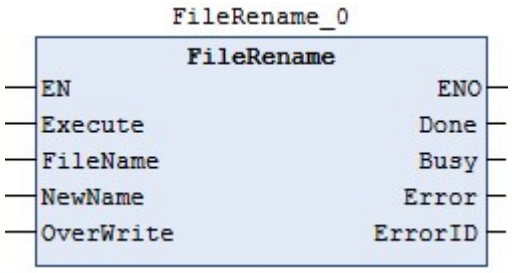
</td></tr>
</table>

2-9-14. Rename file FileRename

(1) Instruction overview

Change the name of the specified file or directory in the SD storage card.

Rename file [FileRename]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	XJ_FunctionUtils		

Instruction	Name	Graph	ST
FileRename	Rename file		FileRename_0(Execute:= , FileName:= , NewName:= , OverWrite:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Instruction enable	BOOL	FALSE/TRUE	FALSE	Input high-level rising edge to start command
FileName	Original file name	STRING	Maximum 1985 characters	-	Original file name
NewName	The changed file name	STRING	Maximum 1985 characters	-	The changed file name
OverWrite	Overlay selection	BOOL	FALSE/TRUE	FALSE	True: Allow overwrite FALSE: Do not overwrite
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed flag	BOOL	FALSE/TRUE	FALSE	Instruction completion signal
Busy	Executing flag	BOOL	FALSE/TRUE	FALSE	Instruction execution status
Error	Error flag	BOOL	FALSE/TRUE	FALSE	Instruction error status
ErrorID	Error code	WORD	As the data type	-	Instruction error ID

(3) Function

- Change the name of the file or directory specified in the original file name "FileName" to the updated file name "NewName" on the SD storage card.
- If there is no file with the same name as "NewName" in the SD storage card, a new file will be created.
- If a file with the same name as "NewName" already exists in the SD storage card, the following steps will be taken based on the allowed value of "OverWrite" to be overwritten:

“OverWrite” value	Operation
TRUE	Overwrite the file or directory
FALSE	Not overwriting the file or directory, an exception occurred

(4) Note

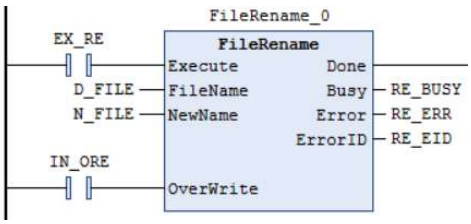
- Even if the value of "Execute" becomes FALSE or the instruction execution time exceeds the task cycle, this instruction will continue to be executed until the end. Whether the processing has ended normally can be confirmed by whether the value of "Done" has changed to true.
- If the directories of "FileName" and "NewName" are different, the file will be moved to the directory specified by "NewName".
- When the SD storage card is unplugged while the file is open, the file will remain open. But when reinstalling the SD memory card, it will not be possible to read or write files. To read and write the file, please reopen the file.
- An exception will occur in the following situations. "Error" is true.
  - When the SD memory card is not in a usable state.
  - The SD storage card is write protected.
  - When the file specified in "FileName" does not exist.
  - When the value of "FileName" or "FileName" is an illegal file name.
  - When accessing the file specified in "FileName".
  - When the number of files and directories that can be created exceeds.
  - When a file with the same name as "FileName" already exists and the value of "OverWrite" is FALSE.
  - When a file with the same name as "DstFileName" already exists and the value of "OverWrite" is true and the file is read-only.
  - When accessing the SD memory card and unable to access due to an exception.

(5) Sample program

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define the variables</td><td colspan="2">

```
VAR
    FileRename_0: FileRename;
    EX_RE: BOOL;
    D_FILE: STRING(1985):= 'DEMO/TESTFILE/VERSION_COPY.TXT';
    N_FILE: STRING(1985):= 'DEMO/TESTFILE/NEW_VERSION.TXT';
    IN_ORE: BOOL;
    RE_DONE: BOOL;
    RE_BUSY: BOOL;
    RE_ERR: BOOL;
    RE_EID: WORD;
END_VAR
```

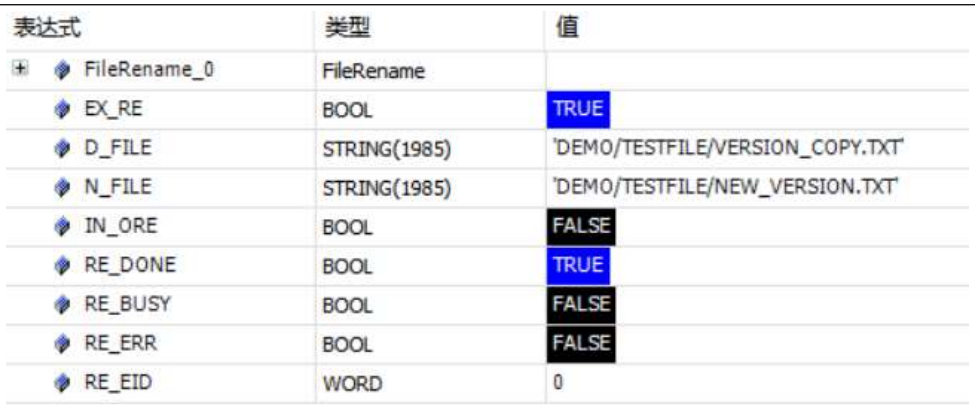
</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
FileRename_0(
    Execute:= EX_RE,
    FileName:= D_FILE,
    NewName:= N_FILE,
    OverWrite:= IN_ORE,
    Done=> RE_DONE,
    Busy=> RE_BUSY,
    Error=> RE_ERR,
    ErrorID=> RE_EID);
```

</td></tr>
</table>

Operation result	

表达式	类型	值
FileRename_0	FileRename	
EX_RE	BOOL	TRUE
D_FILE	STRING(1985)	'DEMO/TESTFILE/VERSION_COPY.TXT'
N_FILE	STRING(1985)	'DEMO/TESTFILE/NEW_VERSION.TXT'
IN_ORE	BOOL	FALSE
RE_DONE	BOOL	TRUE
RE_BUSY	BOOL	FALSE
RE_ERR	BOOL	FALSE
RE_EID	WORD	0

3. Motion instructions

3-1. Single axis

3-1-1. Single axis instruction overview

Command	Function
MC_Power	Put the axis into a runnable state
MC_Reset	Reset axis internal related errors
MC_Stop	Stop controller movement
MC_Halt	Pause the execution of functional blocks in progress
MC_Home	Homing
MC_Jog	Jog run
MC_MoveAbsolute	Implement a control axis to reach the specified absolute position
MC_MoveAdditive	Accelerate to move an additional distance at a given speed
MC_MoveRelative	Move the axis to a relative position from the current axis position
MC_MoveSuperImposed	Based on the previous motion, the superimposed velocity and acceleration run an additional distance
MC_MoveVelocity	The shaft continues to operate at a specified speed
MC_PositionProfile	Perform movements according to time-location planning
MC_VeloctyProfile	Perform movements according to time-velocity planning
MC_AccelerationProfile	Perform movements according to time-acceleration planning
MC_ReadActualPosition	Reads the current position of the current relevant axis
MC_ReadActualTorque	Reads the current torque of the current relevant axis
MC_ReadActualVlocity	Reads the current speed of the current related axis
MC_ReadAxisError	Obtain the error code
MC_ReadBoolParameter	Obtain parameter values based on parameter serial number
MC_ReadParameter	Obtain the parameter value based on the parameter ID
SMC_ReadSetPosition	Reads the set position of the current axis
SMC_ReadFBError	Read the historical error information of the function block
MC_WriteBoolParameter	Modify the parameter values of a specific Boolean variable specified by the user
MC_WriteParameter	Modify a special parameter specified by a user
SMC_ClearFBError	Clear historical error information of the function block
SMC_ErrorString	Read the error description corresponding to the error code
SMC3_ReinitDrive	Re-invoke the drive/axis
SMC3_ETC_WriteParameter_CoE	Set COE parameters for the axis
MC_TouchProbe	Probe instruction

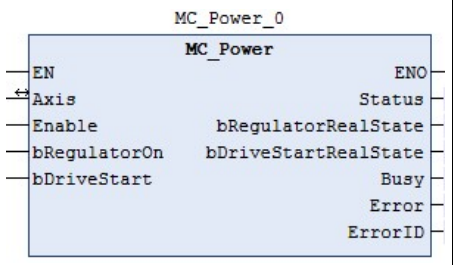
3-1-2. Single axis instructions

3-1-2-1. Axis enable [MC_Power]

(1) Instruction overview

It is used to enable a specified axis to enter or exit the runable state, also called axis enable.

Axis enable [MC_Power]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_Power	Axis enable		MC_Power_0(Axis:= , Enable:= , bRegulatorOn:= , bDriveStart:= , Status=> , bRegulatorRealState=> , bDriveStartRealState=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	Valid	BOOL	TRUE,FALSE	FALSE	Must be set to TRUE to activate processing of function blocks
bRegulatorOn	Enable	BOOL	TRUE,FALSE	FALSE	Must be set to TRUE to enable the function block
bDriveStart	Enable drive	BOOL	TRUE,FALSE	FALSE	Must be set to TRUE to turn off emergency stop processing for function blocks
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Status	Can run	BOOL	TRUE,FALSE	FALSE	TRUE if the axis is ready
bRegulatorRealState	Enable valid	BOOL	TRUE,FALSE	FALSE	Active status of the axis enablement
bDriveStartRealState	Drive can be used	BOOL	TRUE,FALSE	FALSE	The drive is not interrupted by the quick stop mechanism is TRUE
Busy	Executing	BOOL	TRUE,FALSE	FALSE	The processing of the function block did not complete is TRUE
Error	Error	BOOL	TRUE,FALSE	FALSE	TRUE if an exception occurs
ErrorID	Error code	SMC_ERROR	-	0	In normal cases, the value is 0. When an exception occurs, an error code is displayed

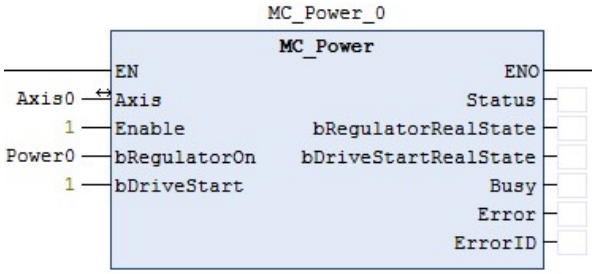
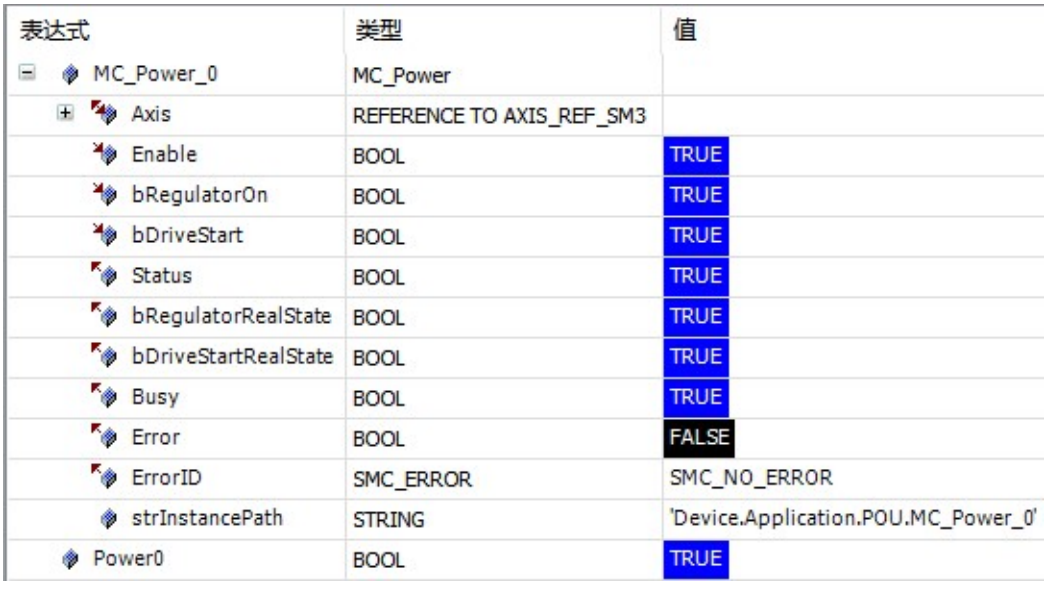
(3) Function description

- Processes the input only if Enable is TRUE.
- Call MC_Power with bRegulatoron = FALSE to set nAxisState of the reference axis to state (disabled) off. Then the axis can't move.
- If there are no errors on the disabled axis, call MC_Power with bRegulatorOn = TRUE, sets its AxisState to pause.
- If an error is detected, set the axis status to errorstop.
- If the input Enable, bRegulatoron and bDrivestart are TRUE, but the output Status remains FALSE for a long time, there may be a drive power-level hardware problem. If the power supply fails (also during operation), the nAxisState of the reference axis is set to errorstop.

(4) Notes

Write only one MC_Power per axis, otherwise a situation similar to a dual coil will occur.

(5) Program example

DEMO	LD	ST
Define variables	VAR MC_Power_0: MC_Power; Power0 :BOOL; END_VAR	
Sample program		1 MC_Power_0(2 Axis:= Axis0, 3 Enable:= 1, 4 bRegulatorOn:= Power0, 5 bDriveStart:= 1, 6 Status=> , 7 bRegulatorRealState=> , 8 bDriveStartRealState=> , 9 Busy=> , 10 Error=> , 11 ErrorID=>);
Operation result		

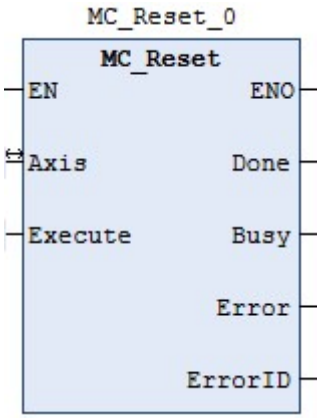
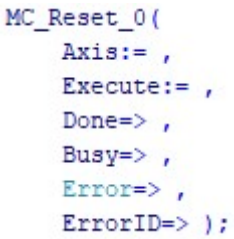
表达式	类型	值
MC_Power_0	MC_Power	
Axis	REFERENCE TO AXIS_REF_SM3	
Enable	BOOL	TRUE
bRegulatorOn	BOOL	TRUE
bDriveStart	BOOL	TRUE
Status	BOOL	TRUE
bRegulatorRealState	BOOL	TRUE
bDriveStartRealState	BOOL	TRUE
Busy	BOOL	TRUE
Error	BOOL	FALSE
ErrorID	SMC_ERROR	SMC_NO_ERROR
strInstancePath	STRING	'Device.Application.POU.MC_Power_0'
Power0	BOOL	TRUE

3-1-2-2. Axis reset [MC_Reset]

(1) Instruction overview

Clear the axis error.

Axis reset [MC_Reset]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_Reset	Axis reset		


```
MC_Reset_0(
    Axis:= ,
    Execute:= ,
    Done=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE,FALSE	FALSE	The rising edge of the input value initiates the execution of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Done	BOOL	TRUE,FALSE	FALSE	TRUE if the reset is performed
Busy	Executing	BOOL	TRUE,FALSE	FALSE	TRUE when the function block execution has not finished
Error	Error	BOOL	TRUE,FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- When the shaft communication is normal, change the axis state from errorstop to Standstill, turns the abnormal state of the axis to the normal running state.
- The function block returns the error SMC_R_NO_ERROR_TO_RESET when called in a state other than errorstop.

(4) Notes

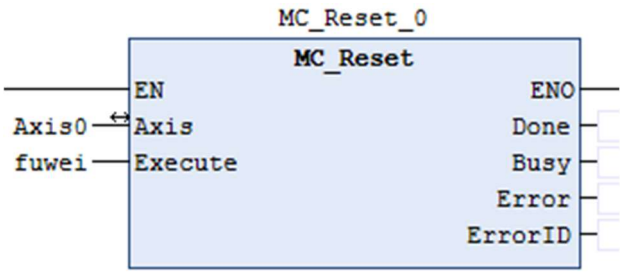
MC_RESET is unable to recover communication and will display an error message saying " SMC_R_CANNOT_RESET_COMMUNICATION _ERROR ".

(5) Program example

<table>
<tr><th>DEMO</th><th>LD</th><th>ST</th></tr>
<tr><td>Define variables</td><td colspan="2">

```
VAR
    power: BOOL;
    fuwei:BOOL;
    MC_Power_0: MC_Power;
    MC_Reset_0: MC_Reset;
    MC_Home_0: MC_Home;
END_VAR
```

</td></tr>
<tr><td>Sample program</td><td>



</td><td>

```
MC_Reset_0(
    Axis:= Axis0,
    Execute:= fuwei,
    Done=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

</td></tr>
<tr><td>Operation result</td><td colspan="2">

MC_Reset_0	MC_Reset	
Axis	REFERENCE TO AXI...	
Execute	BOOL	TRUE
Done	BOOL	TRUE
Busy	BOOL	FALSE
Error	BOOL	FALSE
ErrorID	SMC_ERROR	SMC_NO_ERROR
OldExecute	BOOL	TRUE
rbp	MC_ReadBoolParam...	
wbp	MC_WriteBoolParam...	
wState	WORD	100
wCount	WORD	0
wCount2	WORD	0
pAxisStored	POINTER TO AXIS_...	16#FFFF870876565...
eErrorIDStored	SMC_ERROR	SMC_NO_ERROR
strInstancePath	STRING	'Device.Application.P...

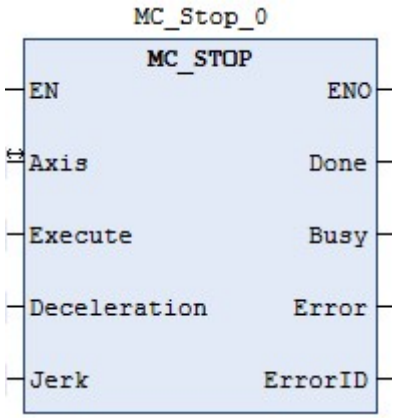
</td></tr>
</table>

3-1-2-3. Stop controller motion [MC_Stop]

(1) Instruction overview

Deceleration stop the axis.

Stop controller motion [MC_Stop]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_Stop	Forced stop		MC_Stop_0(Axis:= , Execute:= , Deceleration:= , Jerk:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Trigger the axis motion at rising edge
Deceleration	Deceleration speed	LREAL	Positive value	0	Deceleration speed. the unit is [command unit/s^2]
Jerk	Jerk	LREAL	Positive value	0	Jerk. the unit is [command unit/s^3]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Done	BOOL	TRUE/FALSE	FALSE	Becomes TRUE upon completion of execution
Busy	Executing	BOOL	TRUE/FALSE	FALSE	Becomes TRUE after receiving instruction
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

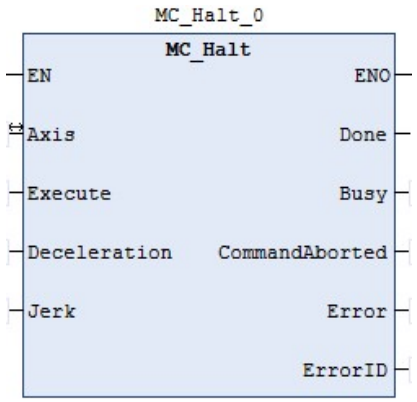
- MC_Stop Sets the axis to the stop state. Therefore, the motion of the currently running function block instance is aborted. The only exception is a running Mc_Stop instance, which will not be aborted. Instead, the MC_Stop instance that was just started returns an error.)
- As long as the axis is in the stopped state, no other instance can perform motion on it. If the axis reaches the speed value of zero, the Done output is set to TRUE. As long as the Execute input is TRUE, the axis remains stop. If Execute goes to FALSE and the Done output is TRUE, the axis goes to a standstill.

3-1-2-4. Pause the motion [MC_Halt]

(1) Instruction overview

Slow down to stop the movement being performed by the axis, and the stopped movement can resume the unfinished part of the execution.

Pause the motion [MC_Halt]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_Halt	Pause the motion		MC_Halt_0(Axis:= , Execute:= , Deceleration:= , Jerk:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Start execution at rising edge
Deceleration	Deceleration speed	LREAL	Positive value	0	Deceleration speed. the unit is [command unit/s^2]
Jerk	Jerk	LREAL	Positive value	0	Jerk. the unit is [command unit/s^3]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Done	BOOL	TRUE/FALSE	FALSE	Becomes TRUE upon completion of execution
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE when the function block execution has not finished
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	TRUE if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- This function block stops referencing axes in a controlled manner. If operations of other function blocks

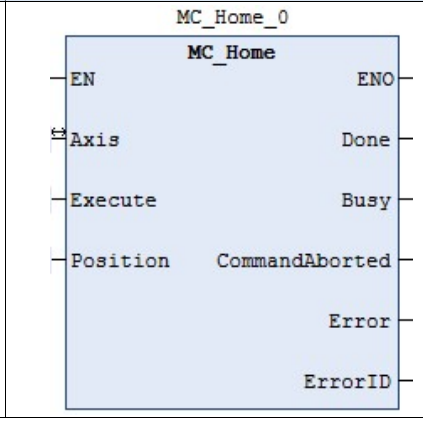
are running at this time, the operation is aborted. The axis enters discrete motion until it reaches velocity 0. If the Done output of MC_Halt is set, the axis comes to a standstill. As long as MC_Halt is active, a new motion command can be issued to interrupt the execution of MC_Halt. Unlike MC_Stop, MC_Halt can be interrupted.

3-1-2-5. Axis homing [MC_Home]

(1) Instruction overview

This command executes the pulse motor homing action, and the specific homing process is determined by the homing mode designed by the bus driver.

Axis homing [MC_Home]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_Home	axis homing		MC_Home_0(Axis:= , Execute:= , Position:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Execute at the rising edge
Position	Axis reached location	LREAL	Data range	0	Represents the homing position of the axis
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	Change to TRUE upon completion of execution
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	Command is interrupted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error

ErrorID	Error code	SMC_ERROR	-	0	Error recognition

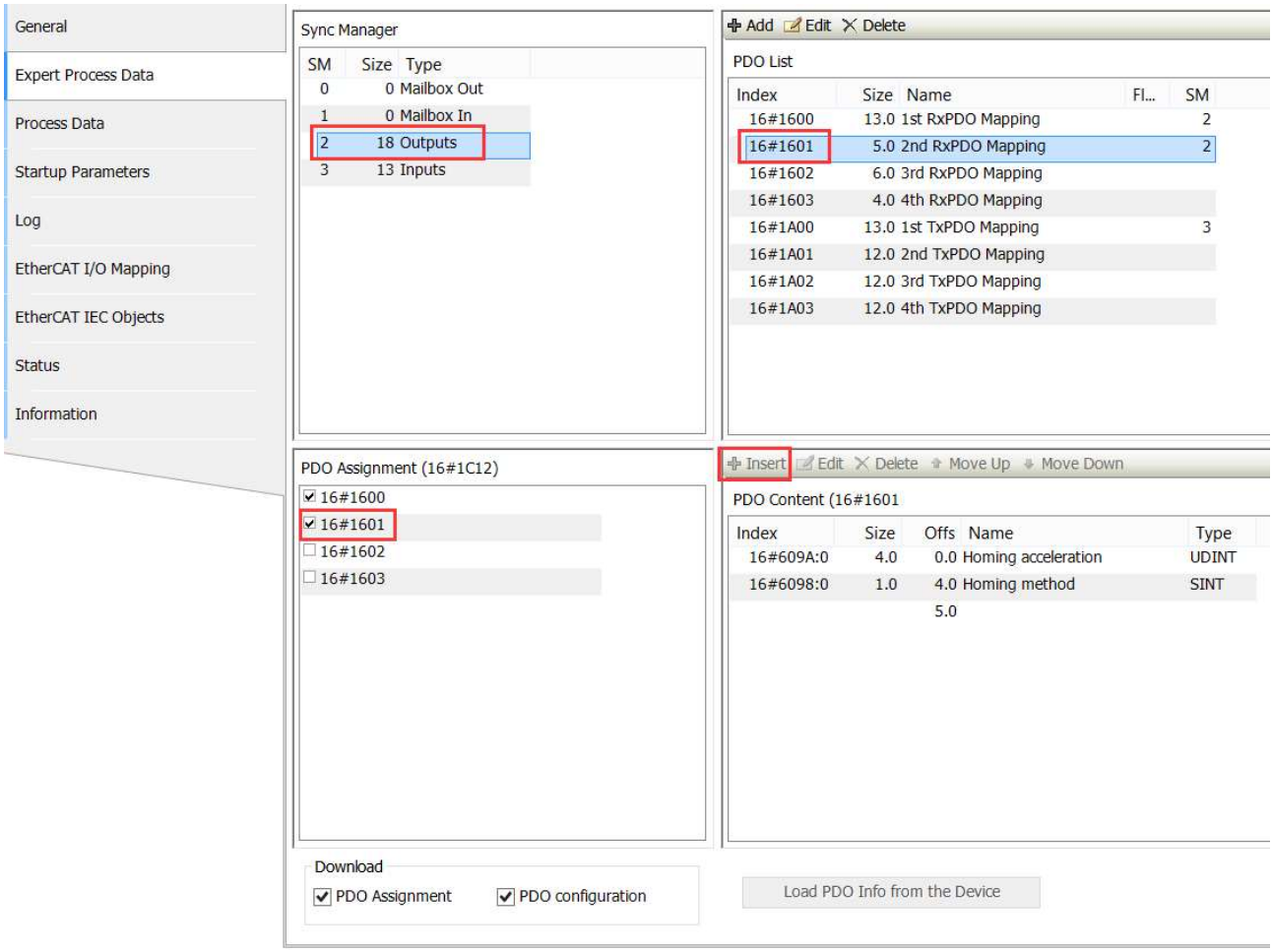
(3) Function

- This function block is for homing operation, and the position is the zero position of the axis.
- The running status of this function block is in Standstill, and the state of the instruction running is homing. Other states cannot be run.
- Start the instruction at the rising edge of Execute.
- Before executing the homing, it is necessary to configure the bus driver homing parameters, such as homing mode, speed, acceleration, etc. Please refer to the driver manual. For example, DS5C1 series servo driver user manual chapter 6.5 HM mode.
- The general bus driver needs to set the index and sub index data as shown in the table below for returning to zero:

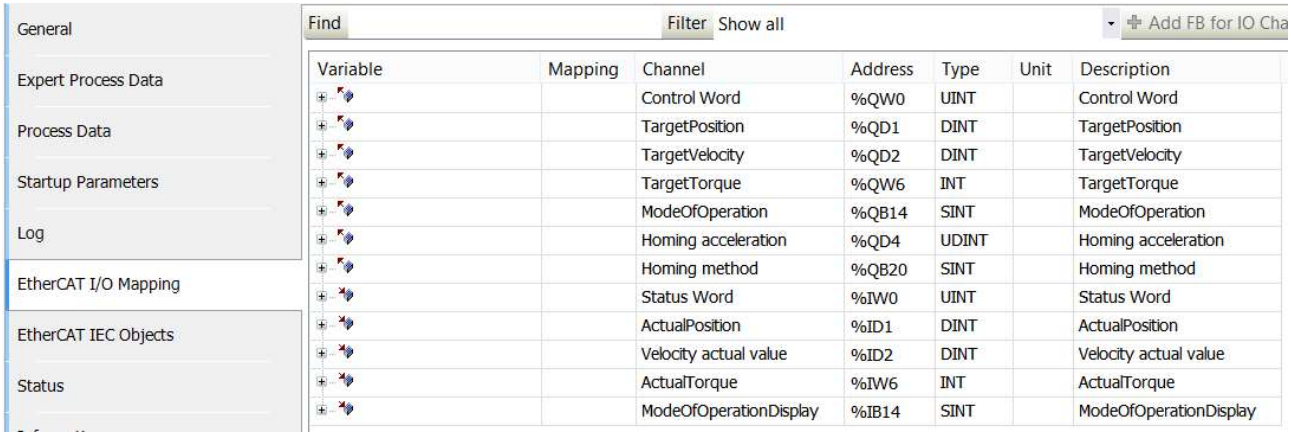
Index	Sub index	Explanation
0x6098	00h	Homing mode
0x6099	01h	The speed of the process from starting to homing to finding zero point, with a higher value to reduce the homing time
0x6099	02h	The speed of the process from finding zero point to homing completed, with lower values to improve accuracy
0x609A	00h	Homing acceleration

(4) Application example

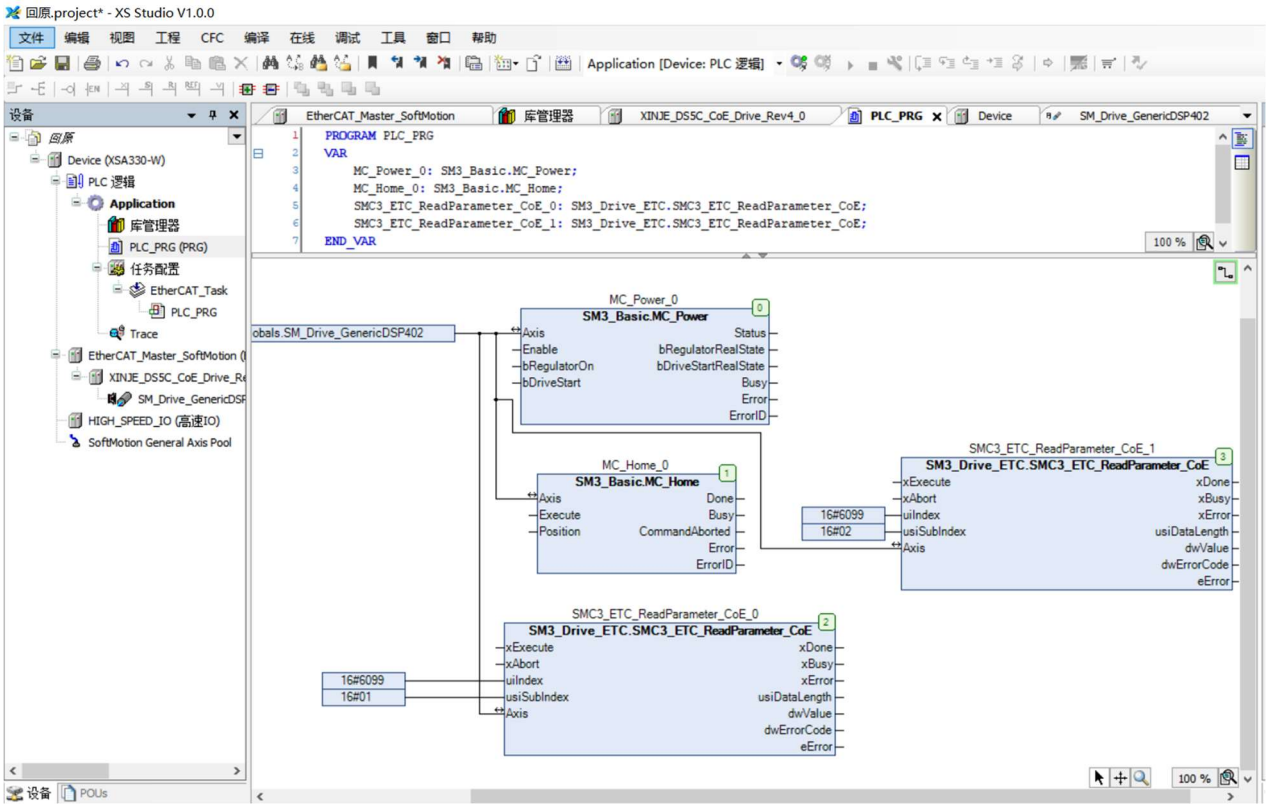
Example 1: Taking the Xinje DS5C servo as an example, the specified axis homing in homing mode 1. P5-22 is the positive limit setting address, with a default value of 1, which corresponds to the servo terminal SI1. P5-23 is the negative limit setting address, with a default value of 2, which corresponds to the servo terminal SI2. P5-27 sets the address for the origin, with a default value of 3, which corresponds to the servo terminal SI3.
Make the program:
Select [expert settings], select 1601 in [Expert process data] – [PDO assignment], add 6098h, 609Ah in 1601.



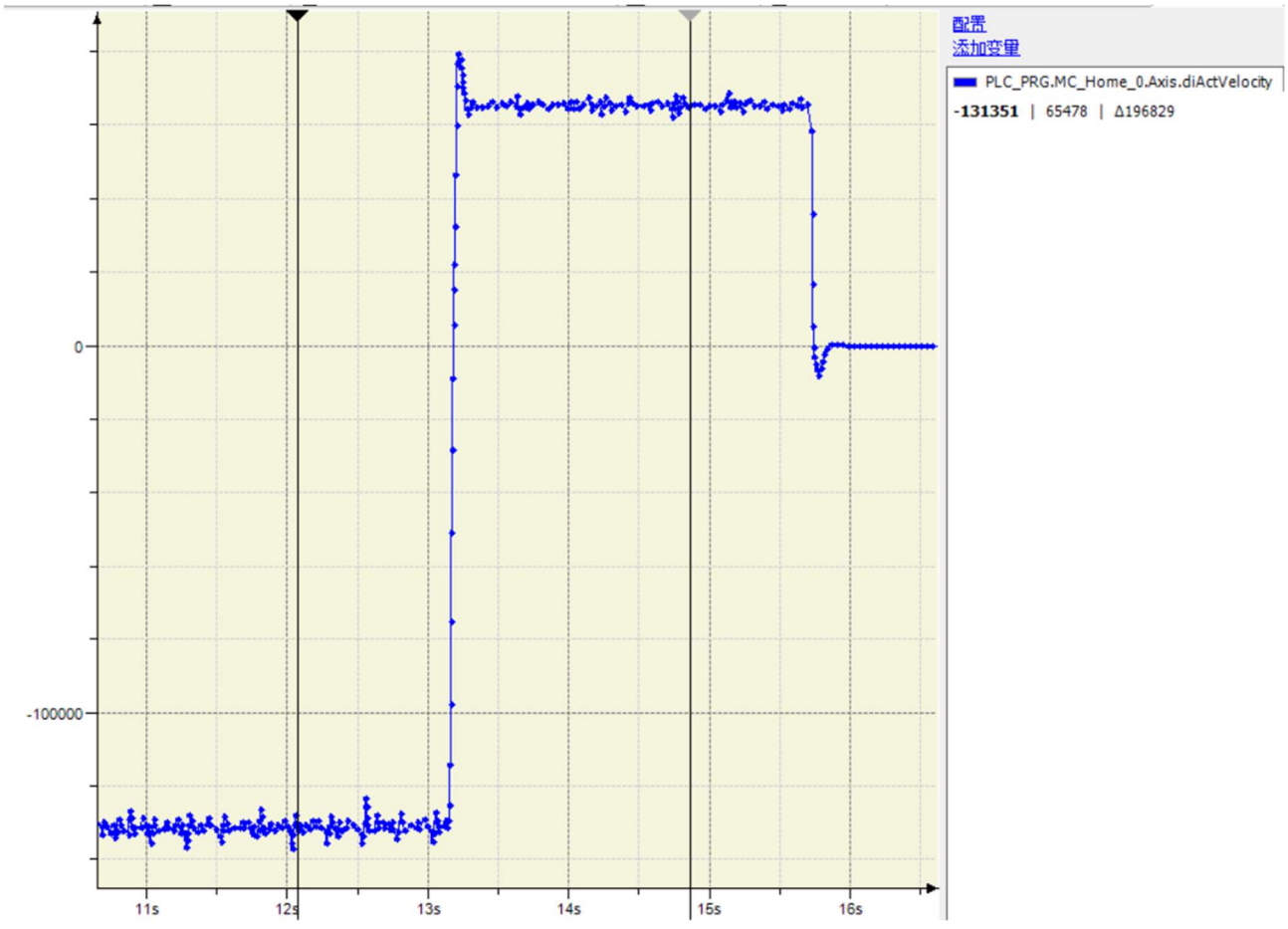
The new added parameters can be seen in [EtherCAT I/O mapping].



Read the homing speed in 6099h through SMC3_ETC_ReadParameter_CoE, set homing mode 1 in [Ethercat I/O mapping], homing acceleration set to 13107200, after enabling through MC_Power, set ON the function block MC_Home, then give SI1 signal.



Check the homing speed through Trace.

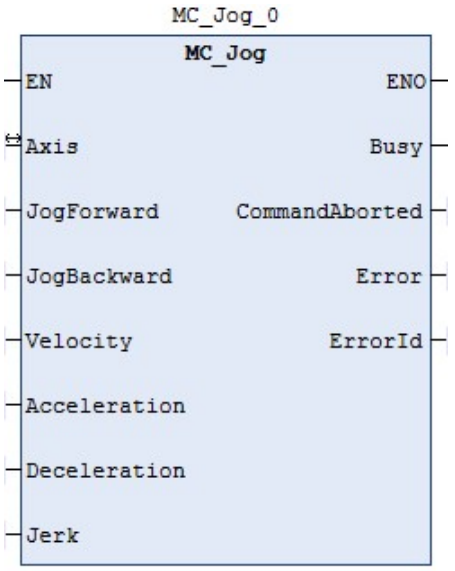
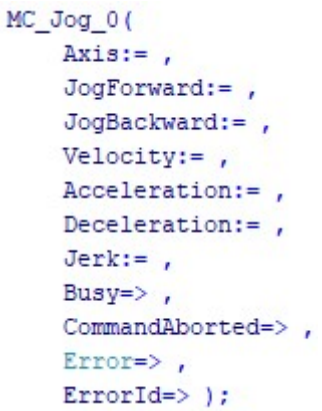


3-1-2-6. Jog run [MC_Jog]

(1) Instruction overview

This command is used to manually control the axis movement in the specified direction.

Jog run [MC_Jog]			
Execution condition	-	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_Jog	Jog run		


```
MC_Jog_0(
    Axis:= ,
    JogForward:= ,
    JogBackward:= ,
    Velocity:= ,
    Acceleration:= ,
    Deceleration:= ,
    Jerk:= ,
    Busy=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorId=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
JogForward	Forward jog run	BOOL	TRUE/FALSE	FALSE	If JogForward is TRUE, the axis will move forward with the given parameters (Velocity, Acceleration, Deceleration, and Jerk), and if JogBackward is also TRUE, the axis will not move
JogBackward	Reverse jog run	BOOL	TRUE/FALSE	FALSE	If JogBackward is TRUE, the axis will move reverse with the given parameters (Velocity, Acceleration, Deceleration, and Jerk), and if JogForward is both TRUE, the axis will not move
Velocity	Target speed	LREAL	0, positive number	0	Maximum speed [u/s]
Acceleration	Target acceleration speed	LREAL	positive number	0	Acceleration value [u/s^2]

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Deceleration	Target deceleration speed	LREAL	positive number	0	Deceleration value [u/s^2]
Jerk	Target jerk speed	LREAL	positive number	0	Jerk value [u/s^3]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error indentification

(3) Function description

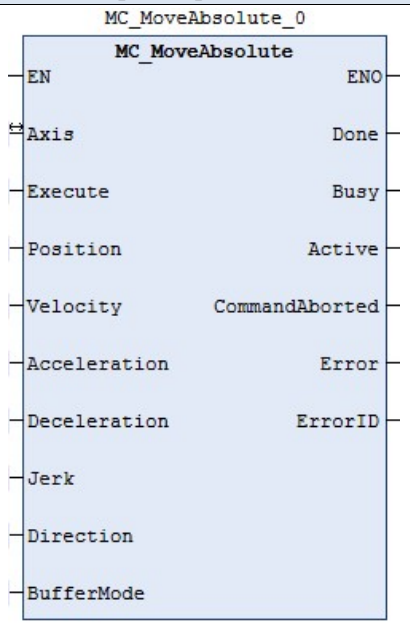
- Perform jog operation according to the specified Velocity (target speed).
- When forward operation is required, set JogForward (valid for forward operation) to TRUE; When reverse running is required, set JogBackward (valid for negative running) to TRUE.
- Setting both JogForward (positive running valid) and JogBackward (negative running valid) to TRUE will prevent any movement from occurring. If MC_Jog command speed setting exceeds the maximum jog speed in the axis parameters, it will be executed at the maximum jog speed.

3-1-2-7. Absolute position [MC_MoveAbsolute]

(1) Instruction overview

This command is used to move the control axis to the specified absolute position according to the set parameters.

Absolute position [MC_MoveAbsolute]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_MoveAbsolute	absolute position		MC_MoveAbsolute_0(Axis:= , Execute:= , Position:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Direction:= , BufferMode:= , Done=> , Busy=> , Active=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Starting execution at the rising edge
Position	Position	LREAL	Data range	0	Target position of motion
Velocity	Target speed	LREAL	0, positive number	0	Maximum speed [u/s]
Acceleration	Target acceleration speed	LREAL	positive number	0	Acceleration value [u/s^2]
Deceleration	Target deceleration speed	LREAL	positive number	0	Deceleration value [u/s^2]
Jerk	Target jerk	LREAL	positive	0	Jerk value [u/s^3]

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
	speed		number		
Direction	Direction	MC_DIRECTION	3: fastest 2: current 1: Positive 0: shortest -1: Negative	0	Fastest: Automatically select the fastest direction to move Current: Move in the current direction Positive: Forward run Shortest: Select direction based on the shortest path Negative: Reverse run
BufferMode	Buffer mode	MC_BUFFER_MODE	-	0	If the function block is Busy, only BufferMode=Aborting is allowed
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	In control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error indentification

(3) Function description

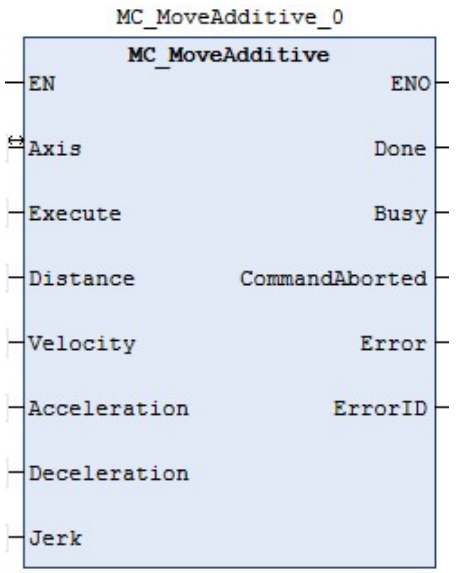
- This function block moves the shaft to an absolute position and uses the values of speed, deceleration, acceleration, and jerk. If not set, the instruction will end at speed 0 after execution. (i.e. standstill state)
- The start command start at the rising edge of Execute.

3-1-2-8. Position overlay [MC_MoveAdditive]

(1) Instruction overview

The axis is overlaid with the data specified by Distance on the original command position, which is used for online stacking position during the motion axis control process.

Position overlay [MC_MoveAdditive]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_MoveAdditive	Position overlay		MC_MoveAdditive_0(Axis:= , Execute:= , Distance:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Starting execution at the rising edge
Distance	Position	LREAL	Data range	0	This data is stacked position data
Velocity	Target speed	LREAL	0, positive number	0	Maximum speed [u/s]
Acceleration	Target acceleration speed	LREAL	positive number	0	Acceleration value [u/s^2]
Deceleration	Target deceleration speed	LREAL	positive number	0	Deceleration value [u/s^2]
Jerk	Target jerk speed	LREAL	positive number	0	Jerk value [u/s^3]

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error indentification

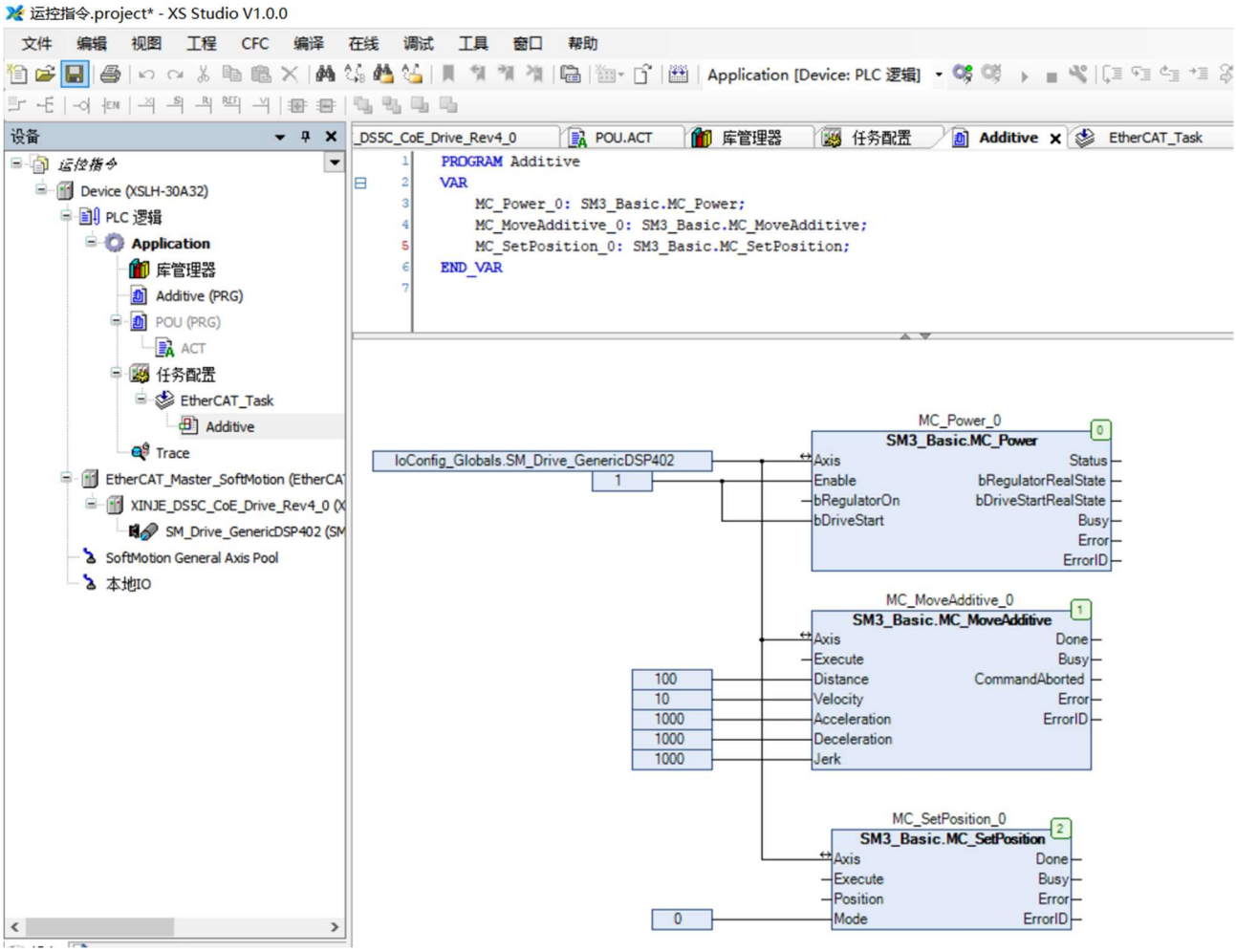
(3) Function description

- This function block is a stacking position command, and the Distance data is the stacking data of the axis;
- If the running status of this function block is Discrete Motion, the CommandAbort of other instructions will be set ON during usage;
- In the standstill state, this instruction can run independently to achieve relative positioning requirements;
- Acceleration or Deceleration is zero, and the instruction operation is in an abnormal state, but the state of the axis is Discrete Motion;
- The start command will start at the rising edge of Execute.

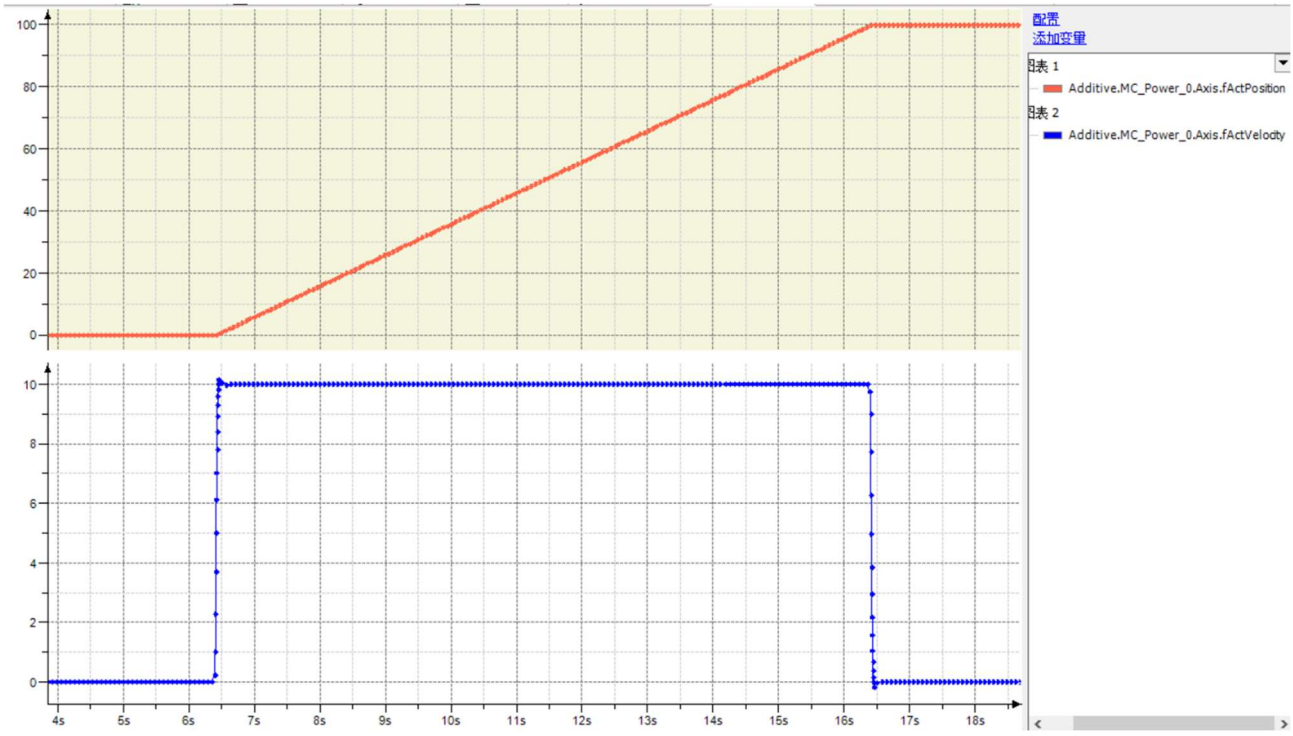
(4) Application example

Example 1: It can be directly used and run through "MC_MoveAdditive" after "MC_Power" is enabled.

① Programming: You can first set the position to 0 through "MC_SetPosition", enable it, and then set ON MC_ MoveAdditive, can be configured to run at a speed of 10 and at a position of 100.

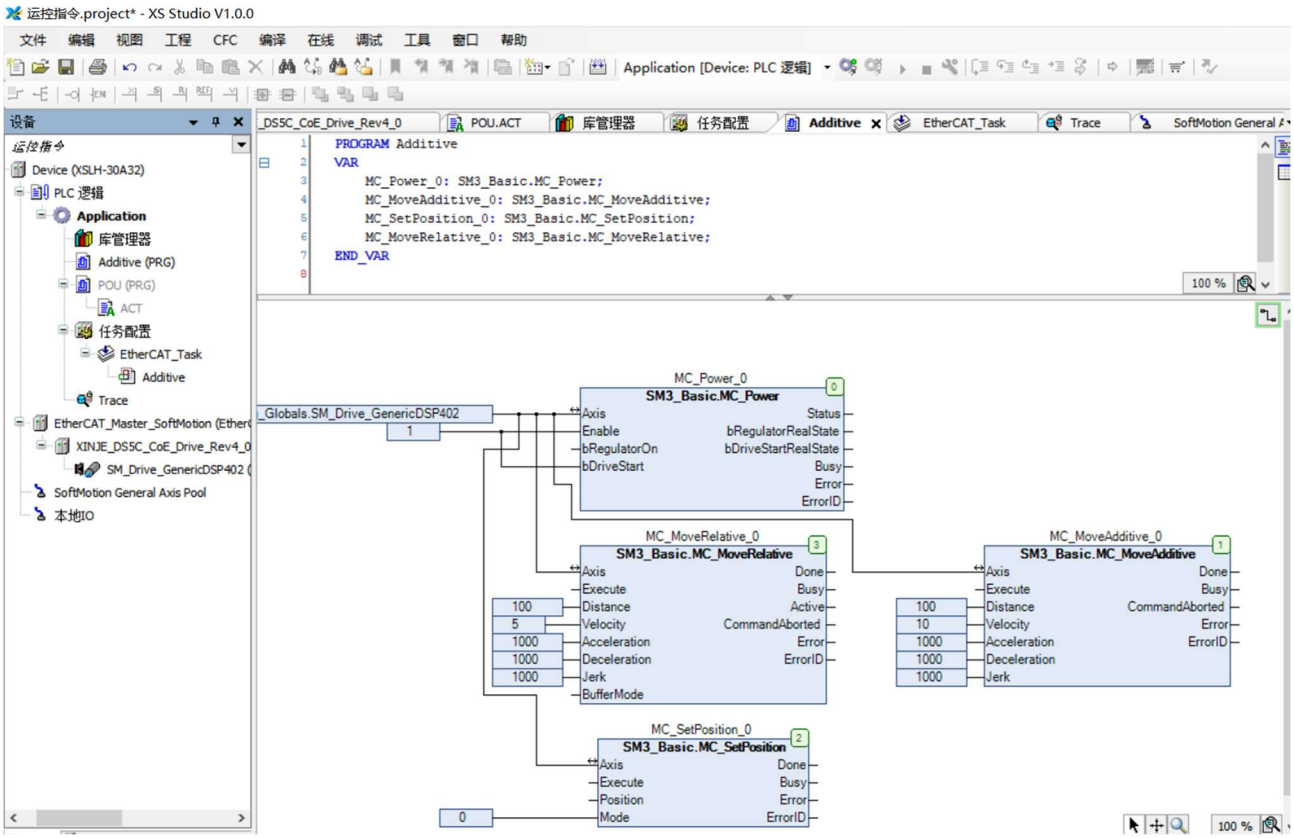


② The position and speed of operation can be seen through Trace.

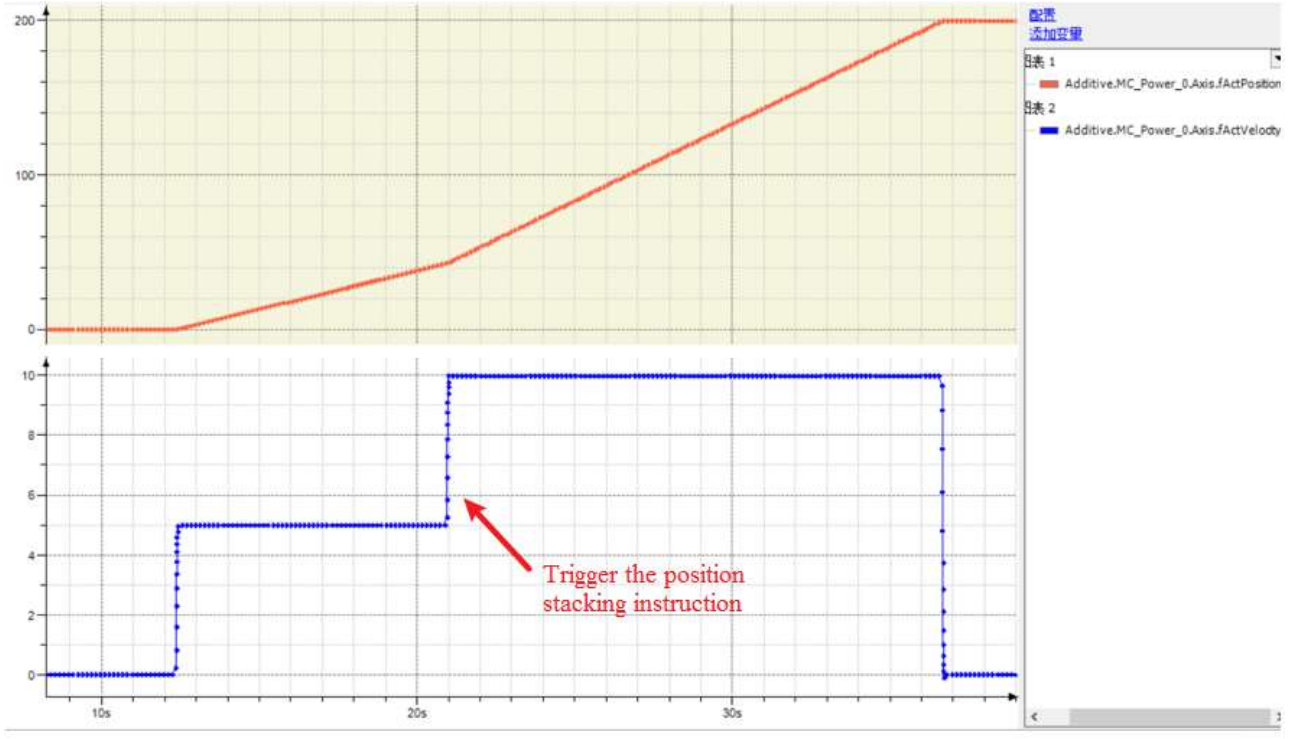


Example 2: After "MC_Power" is enabled, execute "MC_MoveRelative" first, and then "MC-MoveAdditive".

① Write a program: "MC_MoveRelative" is executed at a speed of 5 and a target position of 100. It can be turned on during operation or after waiting for the execution of "MC_MoveRelative" to complete. The speed set for "MC_MoveAdditive" is 10 and the target position is 100. The execution speed after turning on is 10 and the final position is 100. That is, the position is superimposed, and the acceleration is not superimposed, only the velocity is reset.



② The current speed and position can be monitored through trace.

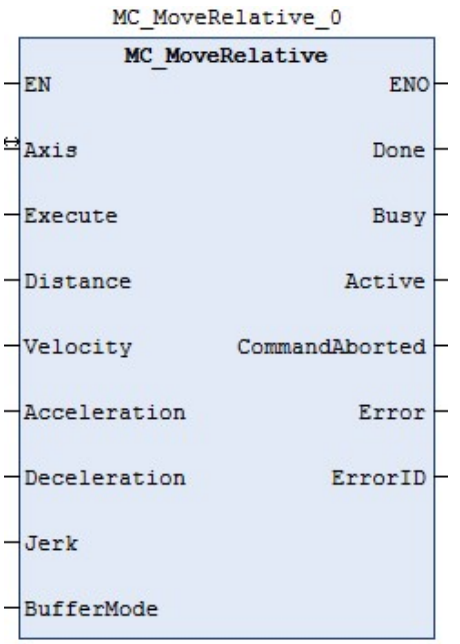


3-1-2-9. Relative position [MC_MoveRelative]

(1) Instruction overview

The axis runs in relative position, which is specified by Distance.

Relative position [MC_MoveRelative]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

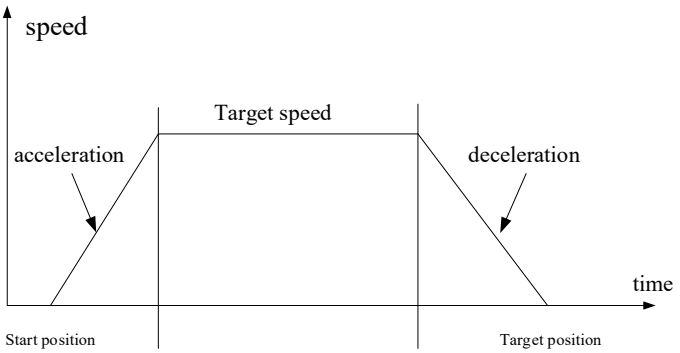
Instruction	Name	Graphic representation	ST language
MC_MoveRelative	Relative position		MC_MoveRelative_0(Axis:= , Execute:= , Distance:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , BufferMode:= , Done=> , Busy=> , Active=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

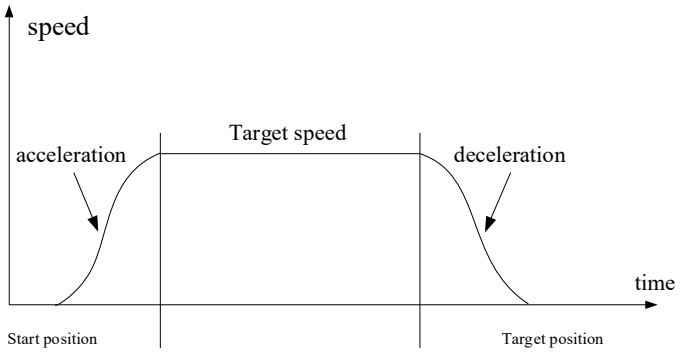
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Starting execution at the rising edge
Distance	Position	LREAL	Data range	0	The relative distance between the target and its current position
Velocity	Target speed	LREAL	0, positive number	0	Maximum speed [u/s]
Acceleration	Target acceleration speed	LREAL	positive number	0	Acceleration value [u/s^2]
Deceleration	Target deceleration	LREAL	positive number	0	Deceleration value [u/s^2]

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
	speed				
Jerk	Target jerk speed	LREAL	positive number	0	Jerk value [u/s^3]
BufferMode	Buffer mode	MC_BUFFER_MODE	-	0	If the function block is Busy, only BufferMode=Aborting is allowed
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	In control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

- The running status of this function block is in Standstill, and the state of the instruction during execution is Discrete Motion. Pay attention to the running status of this axis during instruction execution to avoid interrupting other instructions on this axis or being interrupted by other instructions.
- The start command will start at the rising edge of Execute. This command is valid for repeated rising edges in Discrete Motion, and can refresh the latest Position every time.
- Acceleration or Deceleration is zero, and the instruction operation is in an abnormal state, but the state of the axis is Discrete Motion;
- Trapezoidal acceleration and deceleration action
  Velocity, Acceleration and Deceleration have value but Jerk is 0.



- S curve acceleration and deceleration action
  Velocity, Acceleration, Deceleration and Jerk have value.



3-1-2-10. Superimposed relative motion command [MC_MoveSuperImposed]

(1) Instruction overview

The speed and position data of the axis are superimposed on the running command based on the original command speed and position.

Superimposed relative motion [MC_MoveSuperImposed]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_MoveSuperImposed	Superimp-osed relative motion command	MC_MoveSuperImposed_0 MC_MoveSuperImposed EN, Axis, Execute, Abort, Distance, VelocityDiff, Acceleration, Deceleration, Jerk ENO, Done, Busy, CommandAborted, Error, ErrorID, DistanceTravelled, SuperImposedVelocity, SuperImposedAcceleration	MC_MoveSuperImposed_0(Axis:= , Execute:= , Abort:= , Distance:= , VelocityDiff:= , Acceleration:= , Deceleration:= , Jerk:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=> , DistanceTravelled=> , SuperImposedVelocity=> , SuperImposedAcceleration=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Starting execution at the rising edge
Abort	Abort	BOOL	TRUE/FALSE	FALSE	Abort ongoing motion and reset all outputs
Distance	Position	LREAL	Data range	0	This data is stacked position data

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
VelocityDiff	Stacking velocity	LREAL	0, positive number	0	Maximum velocity of superimposed motion [u/s]
Acceleration	Target acceleration speed	LREAL	positive number	0	Acceleration value [u/s^2]
Deceleration	Target deceleration speed	LREAL	positive number	0	Deceleration value [u/s^2]
Jerk	Target jerk speed	LREAL	positive number	0	Jerk value [u/s^3]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

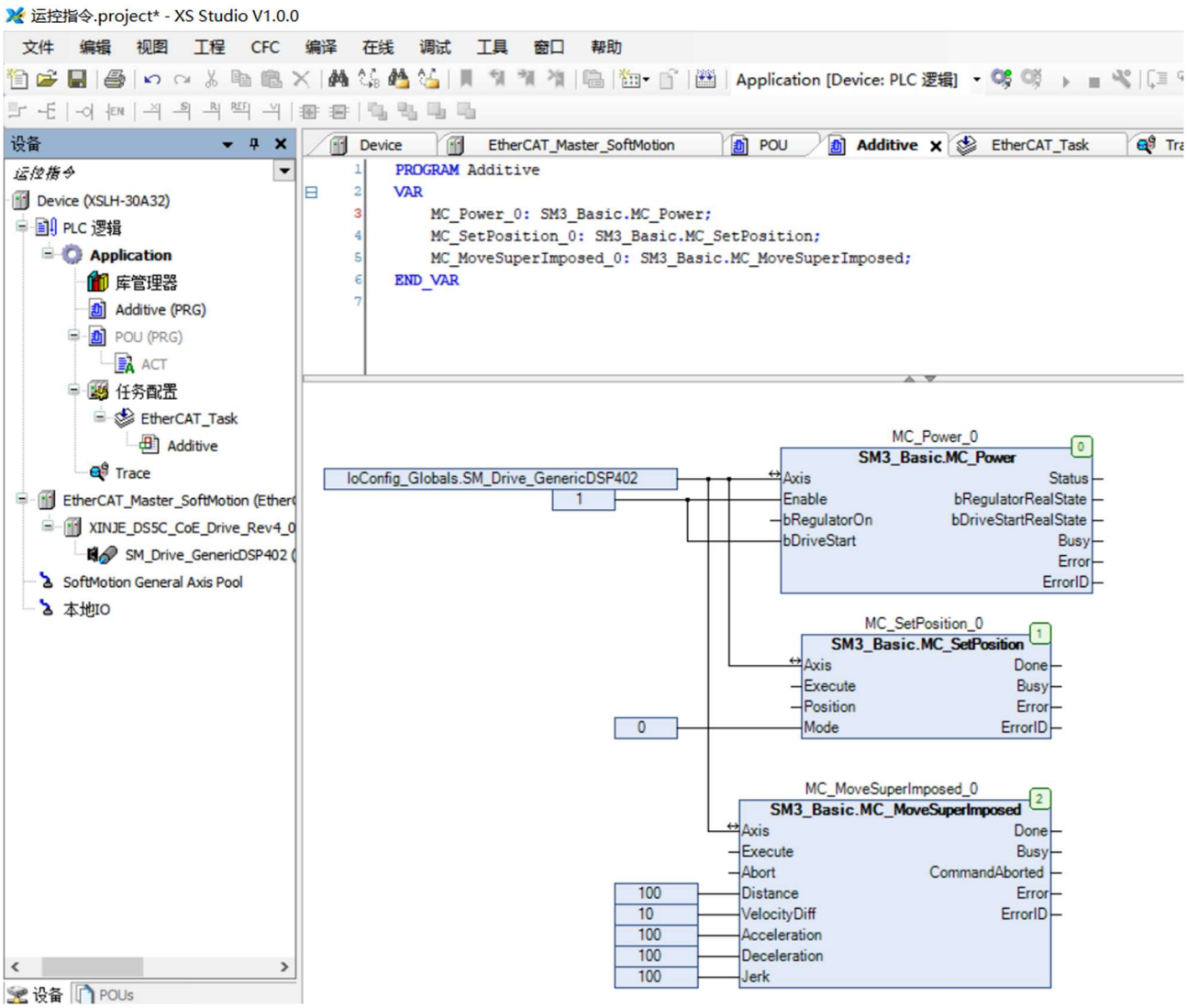
(3) Function description

- This function block is for stacking position and speed commands, while VelocityDiff and Distance represent the speed and position stacked on other commands, respectively;
- MC_MoveSuperImposed can be superimposed on any other instruction;
- MC_MoveSuperImposed can also be aborted by MC_ MoveSuperImposed;
- In the StandStill state, the function block MC_ MoveSuperimposed is similar to MC_ MoveRelative;
- The start command will start at the rising edge of Execute.

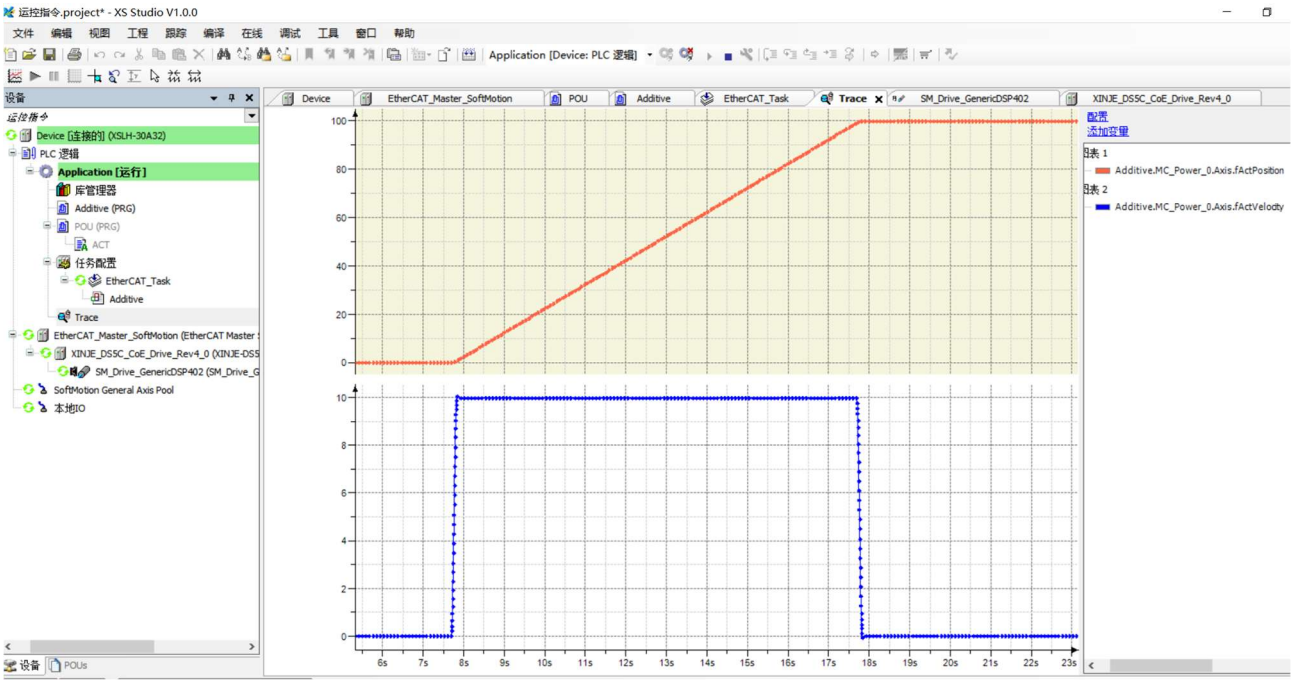
(4) Application

Example 1: It can directly run through "MC_MoveSuperImposed" after "MC_Power" is enabled.

Programming: You can first set the position to 0 through "MC_SetPosition", enable it, and then conduct MC_ MoveSuperImplied, can be configured with a running speed of 10 and a running position of 100.

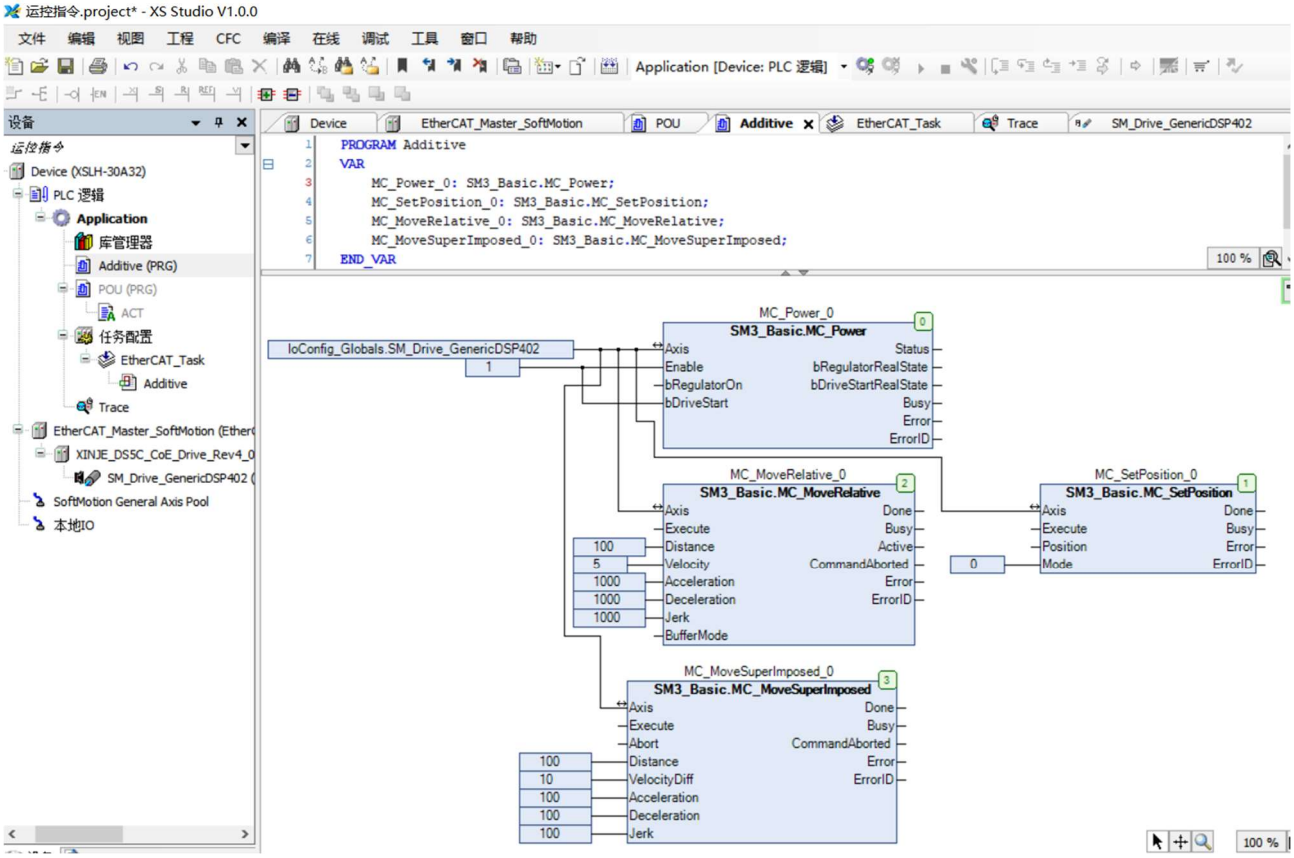


The current speed and position can be monitored through trace.

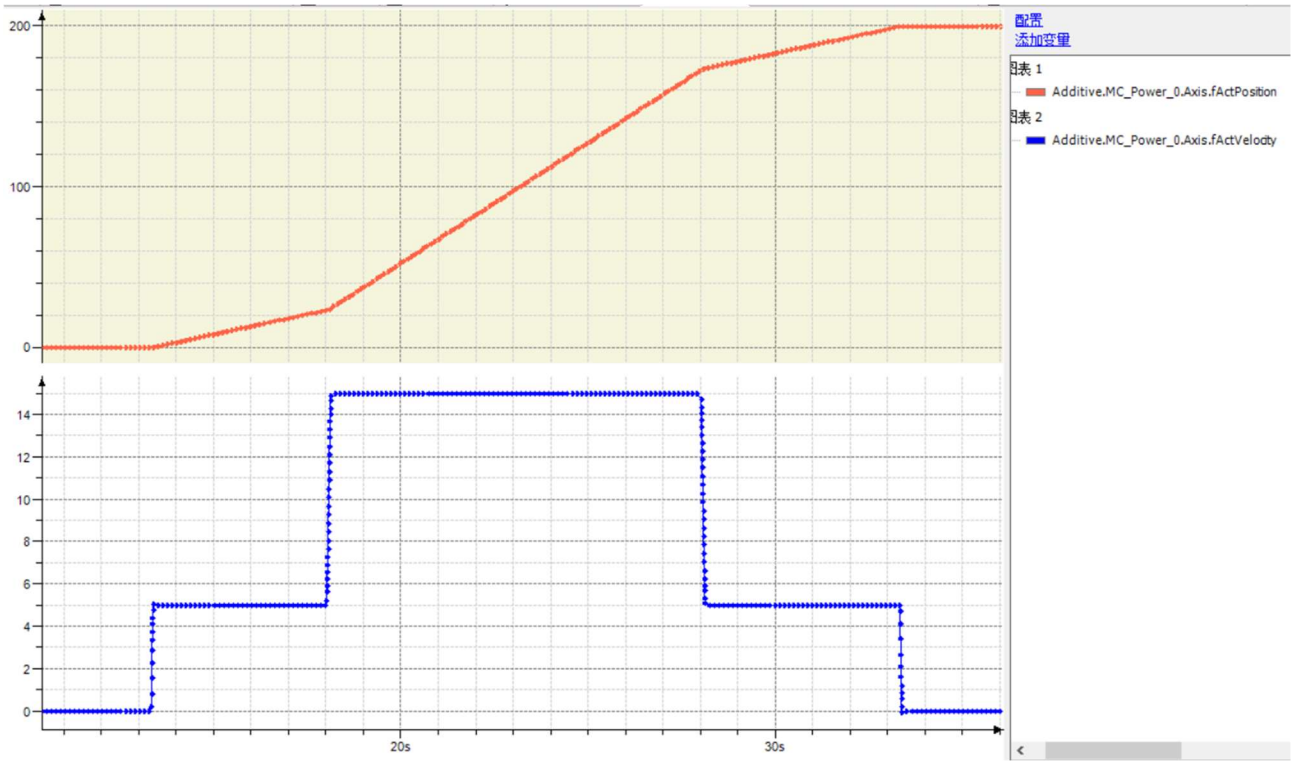


Example 2: After "MC_Power" is enabled, execute "MC_MoveRelative" first, and then "MC_MoveSuperImposed".

Programming: "MC_MoveRelative" is executed at a speed of 5u/s and a target position of 100u. MC_MoveSuperImposed can be turned on during operation or after waiting for the execution of "MC_MoveRelative" to complete. The speed set for "MC_MoveSuperImposed" is 10u/s and the target position is 100u. The position after turning on is the sum of the target positions set in the two instructions. MC_MoveSuperImposed instruction overlays both position and speed.



The current speed and position can be monitored through trace.

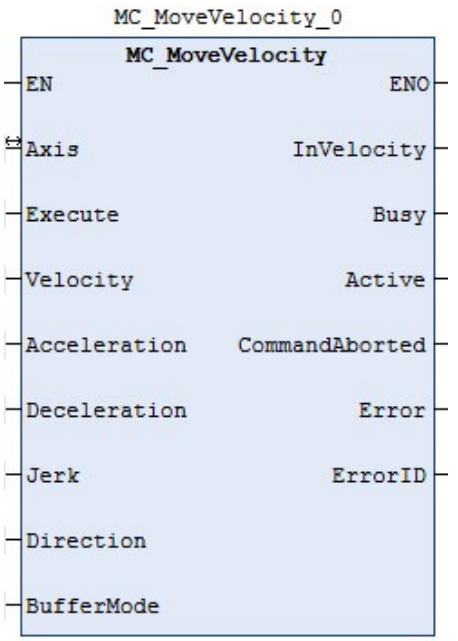


3-1-2-11. Speed control [MC_MoveVelocity]

(1) Instruction overview

This function block moves infinitely at a specified speed.

Speed control [MC_MoveVelocity]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_MoveVelocity	Speed control command		MC_MoveVelocity_0(Axis:= , Execute:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Direction:= , BufferMode:= , InVelocity=> , Busy=> , Active=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Starting execution at the rising edge
Velocity	Speed	LREAL	0, positive value	0	Speed operation value [u/s]
Acceleration	Target acceleration speed	LREAL	0, positive value	0	Acceleration value [u/s^2]
Deceleration	Target deceleration speed	LREAL	0, positive value	0	Deceleration value [u/s^2]
Jerk	Target jerk speed	LREAL	0, positive value	0	Jerk value [u/s^3]

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Direction	Direction	MC_DIRECTION	3: fastest 2: current 1: Positive 0: shortest -1: Negative	0	Fastest: Automatically select the fastest direction to move Current: Move in the current direction Positive: move forward Shortest: Select direction based on the shortest path Negative: move reverse
BufferMode	Buffer mode	MC_BUFFER_MODE	-	0	If the function block is Busy, only BufferMode=Aborting is allowed
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
InVelocity	Flag for reaching the set speed	BOOL	TRUE/FALSE	FALSE	After reaching the set speed, it is set to TRUE
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	In control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
Command Aborted	Command is aborted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

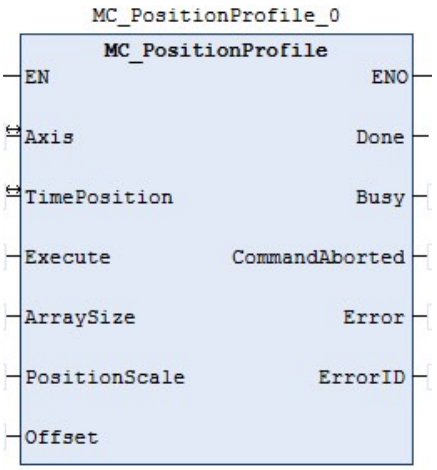
- Change the Velocity parameter to control the speed of the drive.
- The execution of the function block must have a condition of rising edge.
- The InVelocity of the function block indicates that the operating speed of the instruction has reached the set value.
- The Busy of the function block indicates that the current function block is currently executing.

3-1-2-12. Position profile [MC_PositionProfile]

(1) Instruction overview

Users can plan their own "time - position" data table, and the controller will complete the motion according to the planned data.

Position profile [MC_PositionProfile]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_PositionProfile	Position profile command		MC_PositionProfile_0(Axis:= , TimePosition:= , Execute:= , ArraySize:= , PositionScale:= , Offset:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
TimePosition	Data table	MC_TP_REF	-	-	User planned time location data table
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Starting execution at the rising edge
ArraySize	Dynamic array	INT	Positive value	0	Number of arrays used in the running profile
PositionScale	Comprehensive factor	LREAL	Data range	1	Overall Position Scale Factor
Offset	Offset	LREAL	Data range	0	Position offset
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	Command is aborted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- This function block is a profile motion model for time periods and positions, running in Discrete Motion mode according to the data set by the user in the TimePosition variable.
- The running status of this function block is in Standstills, and the state of the instruction running is Discrete Motion. Other states cannot be run.
- The command starts at the rising edge of Execute, and this command runs repeatedly in Discrete Motion.
- TimePosition is MC_ TP_ REF data type;

The specific description of MC_TP_REF is as follows:

Member	Type	Initial value	Description
Number_of_pairs	INT	0	Number of segments in the contour path
IsAbsolute	BOOL	TRUE	Absolute motion (TRUE) and relative motion selection
MC_TP_ArrayARRAY[1..N] OF SMC_TP			Array of time and location

The specific description of SMC_TP is as follows;

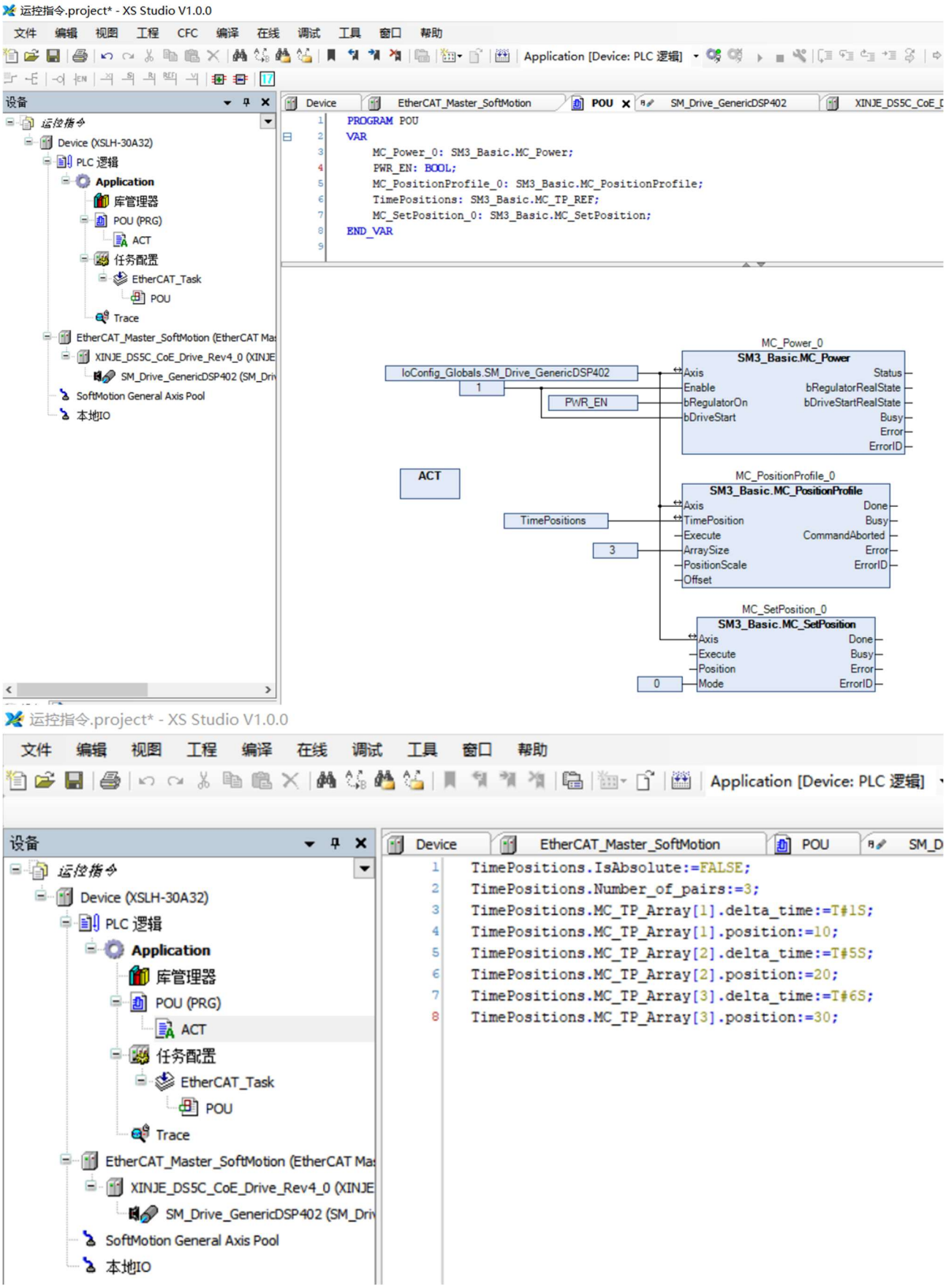
Member	Type	Initial value	Description
delta_time	TIME	TIME#0ms	Time of position segment
position	LREAL	0	Current position value

Note: When the speed corresponding to the set position data changes, relevant adjustments are made according to the S-curve.

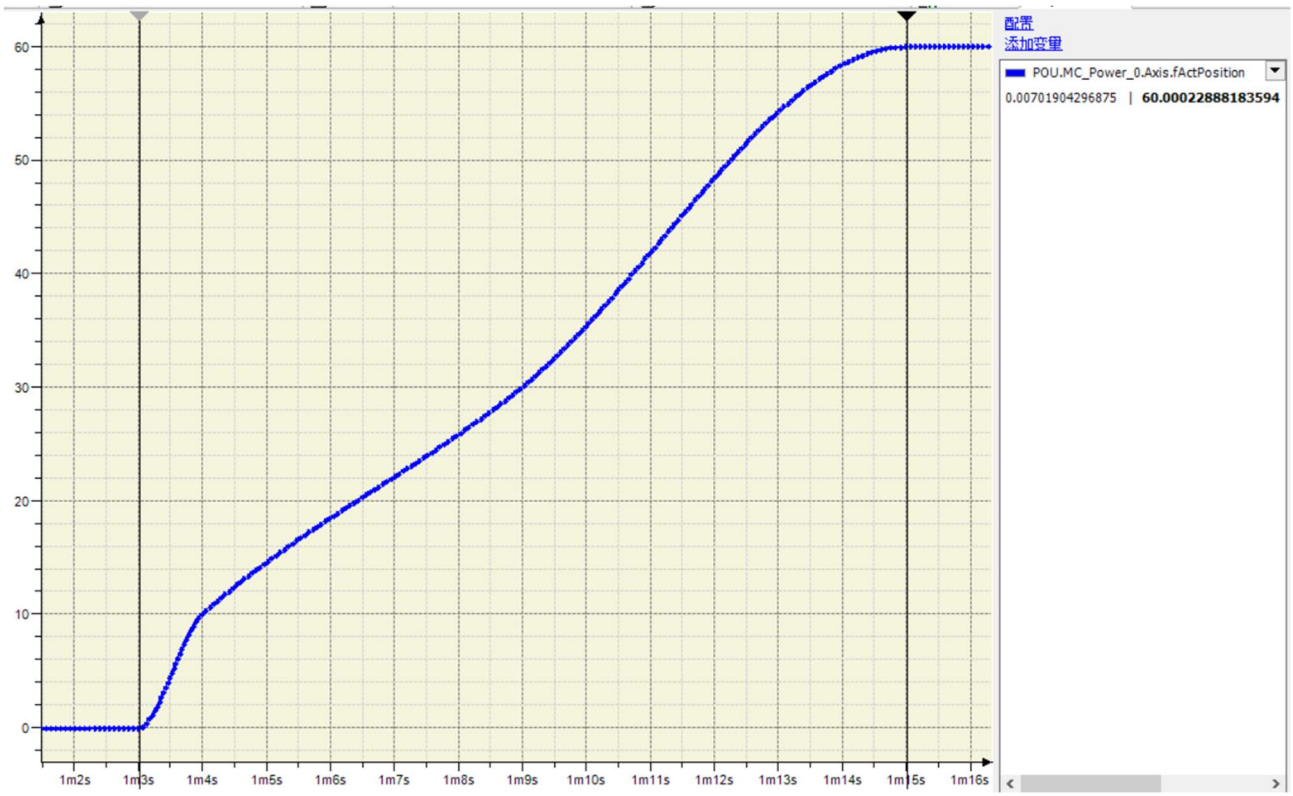
(4) Application

Example 1: Set the execution of three segments and set them as relative motion. The first segment runs at a position of 10 within 1 second, the second segment runs at a position of 20 within 5 seconds, and the third segment runs at a position of 30 within 6 seconds. The total duration of these three segments is 12 seconds, and the total distance of execution is 60.

Write a program using the "MC_Power" and "MC_PositionProfile" instructions, enable it, and then set and run the three segments as the command MC_ PositionProfile. Set the position to 0 through the command MC_SetPosition before executing. In ACT (Action Properties of POU), set the time and position.



You can set 'Trace' to view the execution time and location.

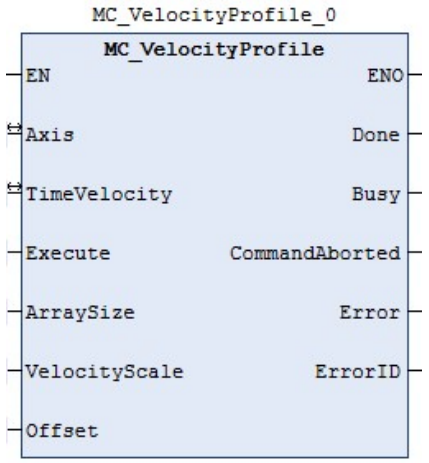

配置
添加变量
POU.MC_Power_0.Axis.fActPosition
0.00701904296875 | 60.00022888183594
60
50
40
30
20
10
0
1m2s
1m3s
1m4s
1m5s
1m6s
1m7s
1m8s
1m9s
1m10s
1m11s
1m12s
1m13s
1m14s
1m15s
1m16s

3-1-2-13. Speed profile [MC_VelocityProfile]

(1) Instruction overview

Similar to MC_PositionProfile, MC_VelocityProfile plans motion by defining "time - velocity" data.

Speed profile [MC_VelocityProfile]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_VelocityProfile	Speed profile command		MC_VelocityProfile_0(Axis:= , TimeVelocity:= , Execute:= , ArraySize:= , VelocityScale:= , Offset:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
TimePosition	Data table	MC_TP_REF	-	-	User planned time - speed data table
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Starting execution at the rising edge
ArraySize	Dynamic array	INT	Positive value	0	Number of arrays used in the running profile
VelocityScale	Speed factor	LREAL	Data range	1	Scale factor of speed
Offset	Offset	LREAL	Data range	0	Speed offset
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	Command is aborted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- This function block is a contour motion model for time periods and speeds, running in Continuous Motion mode according to the data set by the user in the TimeVelocity variable.
- The running status of this function block is in Standstills, and the state of the instruction running is Discrete Motion. Other states cannot be run.
- The command will start at the rising edge of Execute, this instruction runs repeatedly in Discrete Motion.
- TimeVelocity is MC_TV_REF data type:

The detail description of MC_TP_REF is as follows:

Member	Type	Initial value	Description
Number_of_pairs	INT	0	Number of segments in the contour path
IsAbsolute	BOOL	TRUE	Absolute motion (TRUE) and relative motion selection
MC_TP_ArrayARRAY[1..N] OF SMC_TP			Array of time and location

The detail description of SMC_TP is as follows:

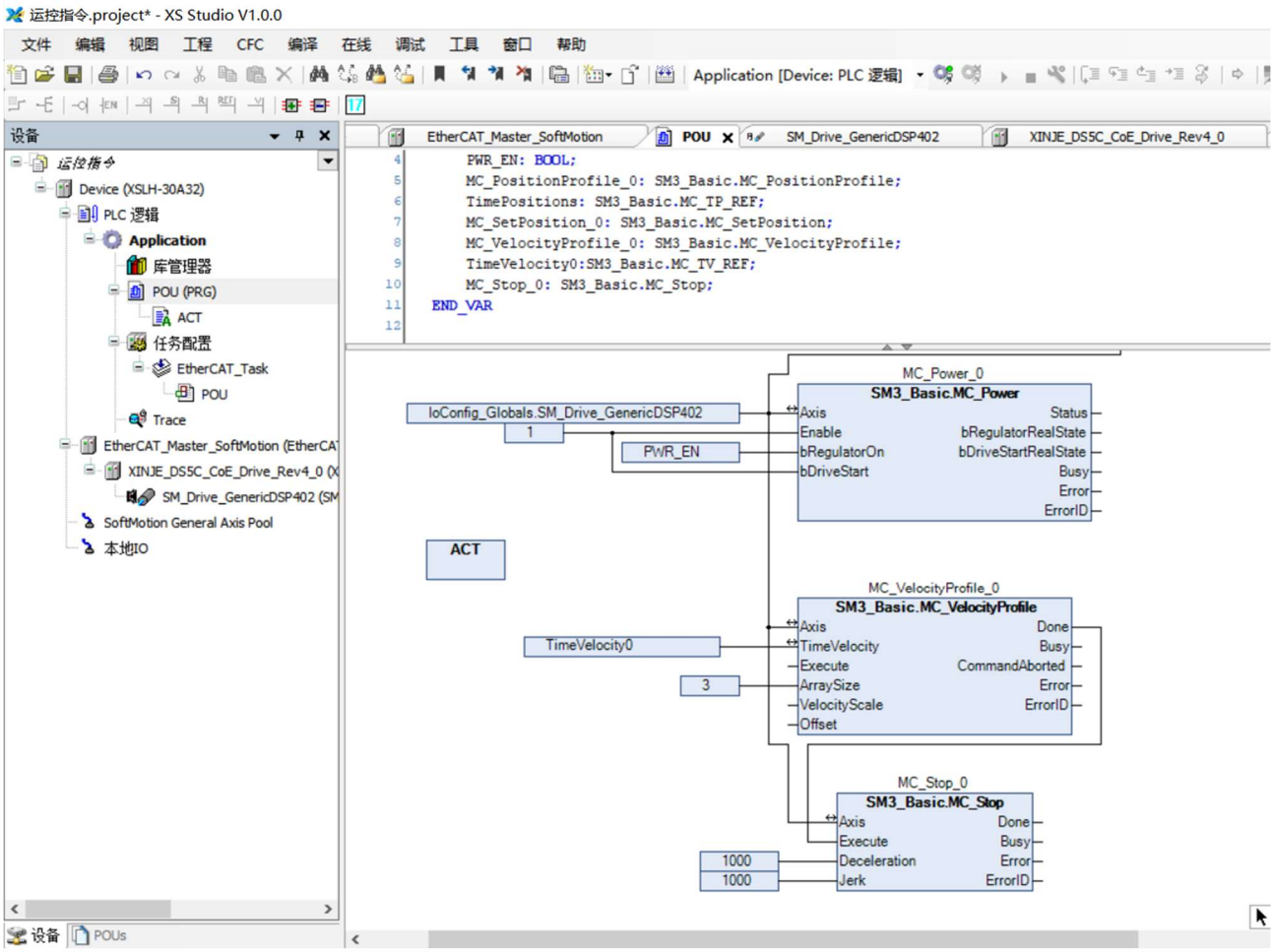
Member	Type	Initial value	Description
delta_time	TIME	TIME#0ms	Time of position segment
position	LREAL	0	Current position value

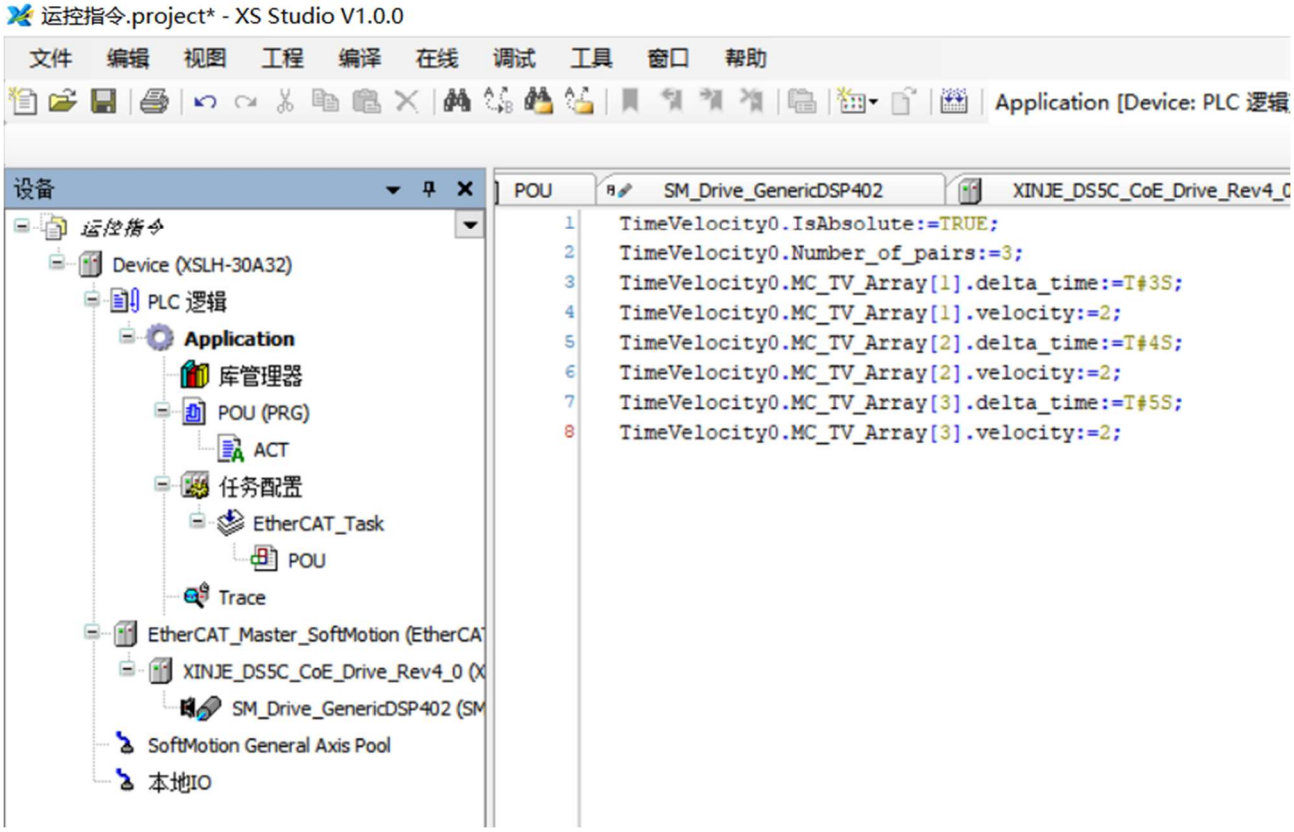
Note: The entire speed process is calculated using an S-curve acceleration and deceleration method, and the speed of each contour segment is calculated using a superposition method; When instructions are repeatedly run, the speed is also stacked to avoid exceeding the speed limit during instruction usage; Repeated operation must return the state of this axis to the Standstill state.

(4) Application

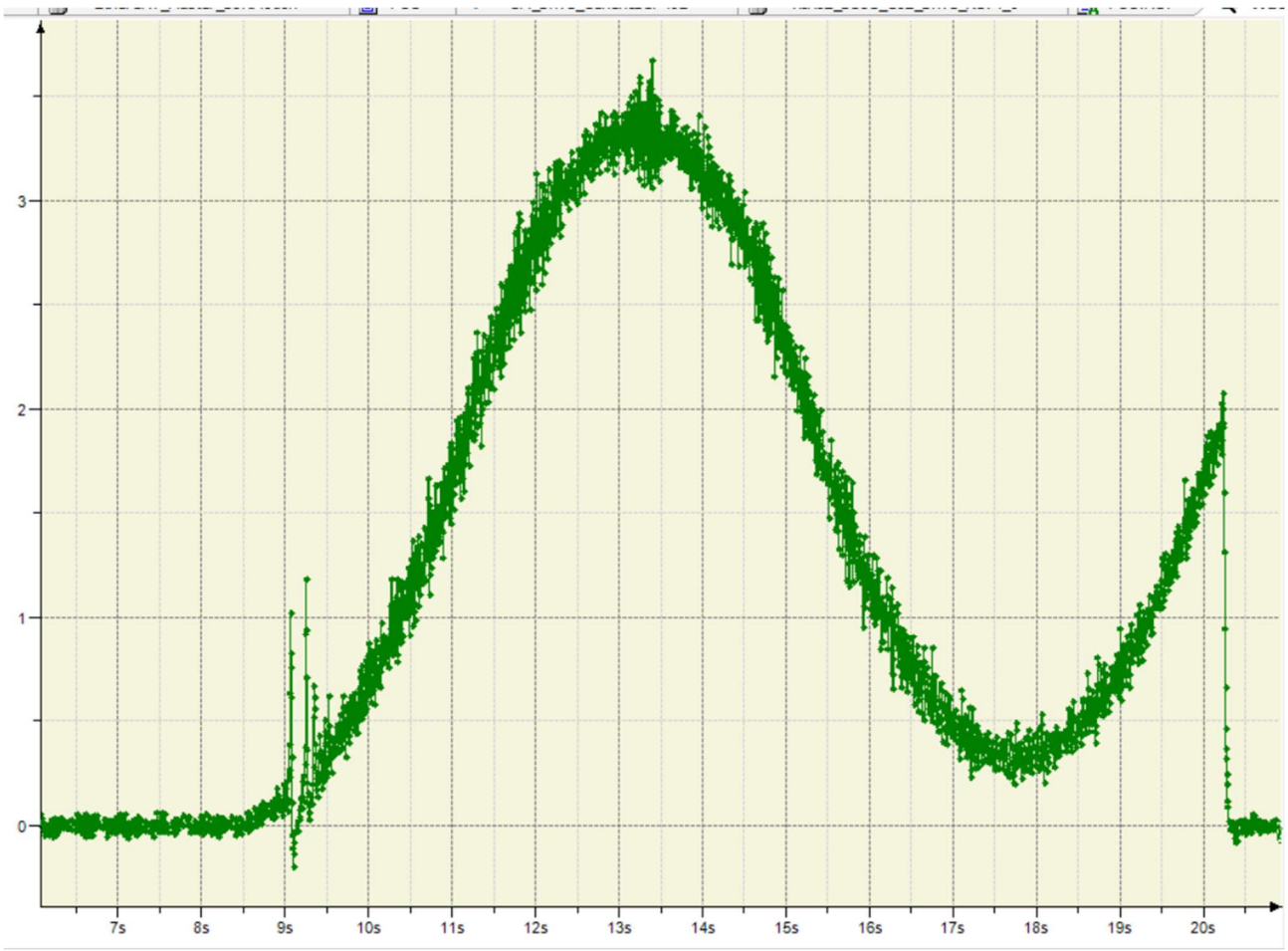
Example 1: Set the execution of three segments, with absolute motion. The first segment reaches a speed of 2 after 3 seconds, the second segment reaches a speed of 2 after 4 seconds, and the third segment reaches a speed of 2 after 5 seconds. The total duration of these three segments is 12 seconds, and the final execution speed is 2.

Make the program by using “MC_Power”, “MC_VelocityProfile”, “MC_Stop” instructions. After enabled, run the three segments as the command setting of MC_VelocityProfile. In ACT (POU action), set the time and speed.





You can view time and speed in Trace.



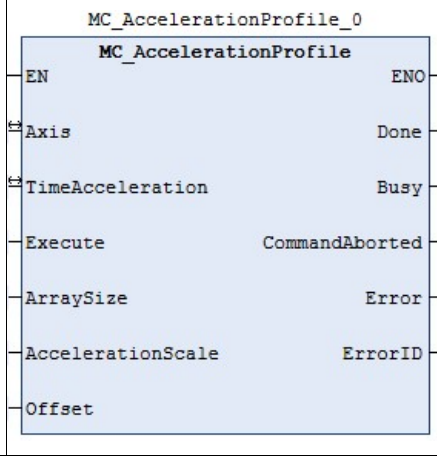
Note: when MC_VelocityProfile is Done, the speed is not 0. So you need to add STOP or other operation and control commands later.

3-1-2-14. Acceleration profile [MC_AccelerationProfile]

(1) Instruction overview

Simliar toMC_PositionProfile instruction, MC_AccelerationProfile plans motion by defining "time - acceleration" data.

Acceleration profile [MC_AccelerationProfile]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_AccelerationProfile	Acceleration profile comand		MC_AccelerationProfile_0(Axis:= , TimeAcceleration:= , Execute:= , ArraySize:= , AccelerationScale:= , Offset:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
TimeAcceleration	Data table	MC_TA_REF	-	-	Reference time/acceleration description
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	Starting execution at the rising edge
ArraySize	Dynamic array	INT	Positive value	0	Number of arrays used in the running profile
AccelerationScale	Comprehensive factor	LREAL	Data range	1	Scale factor for acceleration or deceleration
Offset	Offset	LREAL	Data range	0	Acceleration offset
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	Command is aborted	BOOL	TRUE/FALSE	FALSE	True if the command has been terminated by another command
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- This function block is a contour motion model for the time period and acceleration/deceleration. The running mode is Discrete Motion, which runs according to the data set in the user's TimeAcceleration variable.
- The running status of this function block is in Standstills, and the state of the instruction running is Discrete Motion. Other states cannot be run.
- The command will start at the rising edge of Execute, and the repeated running speed of this command in Discrete Motion is the superposition of the previous one, which can easily cause system faults.
- TimeVelocity is MC_TV_REF data type.

The specific description of MC_TP_REF is as follows:

Member	Type	Initial value	Description
Number_of_pairs	INT	0	Number of segments in the contour path
IsAbsolute	BOOL	TRUE	Absolute motion (TRUE) and relative motion selection
MC_TP_ArrayARRAY[1..N] OF SMC_TP			Array of time and location

The specific description of SMC_TP is as follows:

Member	Type	Initial value	Description
delta_time	TIME	TIME#0ms	Time of position segment
position	LREAL	0	Current position value

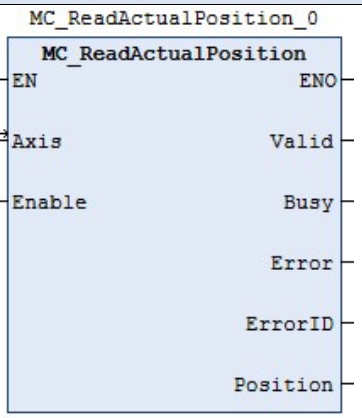
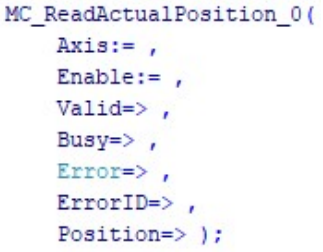
Note: The set acceleration is reflected in the change in velocity, and all acceleration changes follow the S-curve. The acceleration data from the final result change to [starting acceleration is A, ending acceleration is B] (A+B)/2 is reflected in the final velocity.

3-1-2-15. Read actual position [MC_ReadActualPosition]

(1) Instruction overview

Used to read the current actual position value of the axis.

Read actual position [MC_ReadActualPosition]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_ReadActualPosition	read the actual position		


```
MC_ReadActualPosition_0(
    Axis:= ,
    Enable:= ,
    Valid=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> ,
    Position=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	Valid	BOOL	TRUE/FALSE	FALSE	Read the current position of the servo for the true state
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	Obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
Position	Obtained axis position	LREAL	Data range	0	Axis position data read from instructions

(3) Function description

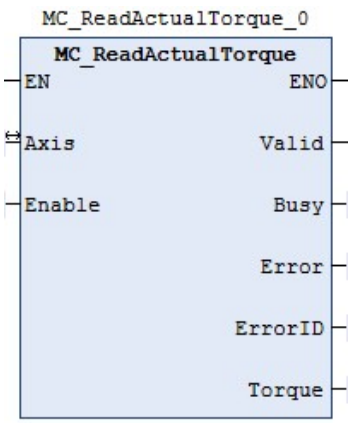
- Read the actual position command in the driver through this command, which is the Enable level enable effect. Instructions can be used multiple times without affecting each other.

3-1-2-16. Read current torque [MC_ReadActualTorque]

(1) Instruction overview

Used to read the actual torque value of the shaft.

Read current torque [MC_ReadActualTorque]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_ReadActualTorque	Read the current torque value		MC_ReadActualTorque_0(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , Torque=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	Valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	Obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
Torque	Current torque value obtained	LREAL	Data range	0	The current torque data read by the command

(3) Function description

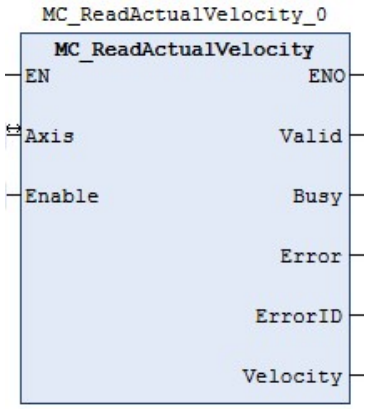
- Reads the current torque value instruction in the driver through MC_ReadActualTorque, which is the enable effect of the Enable level. Instructions can be used multiple times without affecting each other.

3-1-2-17. Read the current speed [MC_ReadActualVelocity]

(1) Instruction overview

Used to read the actual speed value of the axis.

Read current speed [MC_ReadActualVelocity]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_ReadActualVelocity	Read the current speed		MC_ReadActualVelocity_0(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , Velocity=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	Valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	Obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
Velocity	Current speed value obtained	LREAL	Data range	0	The current speed data read by the instruction

(3) Function description

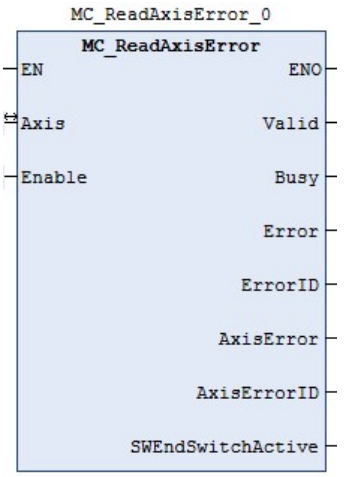
- Reads the current speed value instruction in the driver through MC_ReadActualVelocity, which is the effect of the Enable level. Instructions can be used multiple times without affecting each other.

3-1-2-18. Read axis error status [MC_ReadAxisError]

(1) Instruction overview

Read the axis error.

Read axis error status [MC_ReadAxisError]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
SMC_AxisReadSettingsualVelocity	Read the axis error status		MC_ReadAxisError_0(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , AxisError=> , AxisErrorID=> , SWEndSwitchActive=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	Valid	BOOL	TRUE /FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	Obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
AxisError	Axis error flag	BOOL	TRUE/FALSE	FALSE	Axis error flag
AxisErrorID	Axis error code	DWORD	Positive value, 0	0	Read axis error code
SWEndSwitchActive	Soft limit	BOOL	TRUE/FALSE	FALSE	If the soft limit is exceeded, it is TRUE

(3) Function description

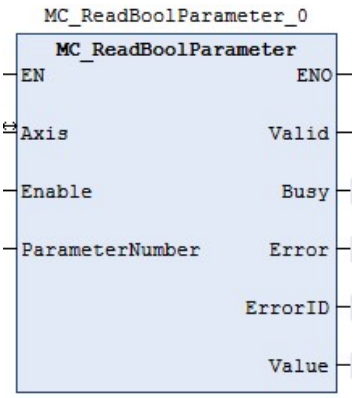
- Reads the error code in the driver through MC_ReadAxisError, and the instruction is Enable level effect. Instructions can be used multiple times without affecting each other.

3-1-2-19. Read the axis bit parameter [MC_ReadBoolParameter]

(1) Instruction overview

Read the value of the specified BOOL type variable.

Read axis bit parameter [MC_ReadBoolParameter]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_ReadBoolParameter	Read the axis bit parameters		MC_ReadBoolParameter_0(Axis:= , Enable:= , ParameterNumber:= , Valid=> , Busy=> , Error=> , ErrorID=> , Value=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Enable	Valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
ParameterNumber	Axis parameter number	DINT	Positive value, 0	0	Parameter number
VAR_OUTPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Valid	Obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
Value	Parameter	BOOL	TRUE/FALSE	FALSE	Read the value of the parameter

	value				

(3) Function description

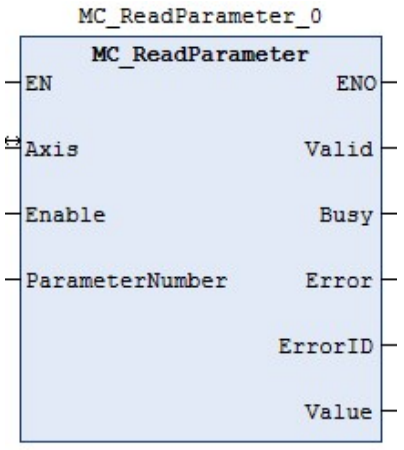
- Reads the bit data status in the driver through MC_ReadBoolParam, and the instruction is Enable level effect. Instructions can be used multiple times without affecting each other.

3-1-2-20. Read the axis parameter [MC_ReadParameter]

(1) Instruction overview

Used to read the specified parameter value.

Read axis parameter [MC_ReadParameter]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_ReadParameter	Read the axis parameter		MC_ReadParameter_0(Axis:= , Enable:= , ParameterNumber:= , Valid=> , Busy=> , Error=> , ErrorID=> , Value=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	Valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
ParameterNumber	Axis parameter number	DINT	Positive value, 0	0	Parameter number
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	Obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
					block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
Value	Parameter value	LREAL	Data range	0	Read the value of the parameter

(3) Function description

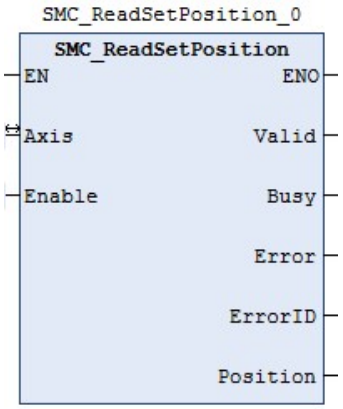
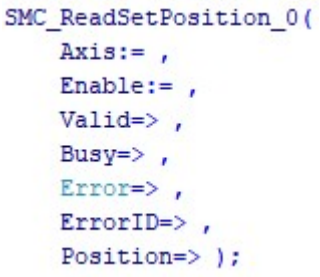
- Reads the bit data status in the driver through MC_ReadParam, and the instruction is Enable level effect. Instructions can be used multiple times without affecting each other.

3-1-2-21. Read axis instruction position [SMC_ReadSetPosition]

(1) Instruction overview

This function block can be used to read the current set position of the drive.

Read axis instruction position [MC_ReadSetPosition]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
SMC_ReadSetPosition	Read axis command position		


```
SMC_ReadSetPosition_0(
    Axis:= ,
    Enable:= ,
    Valid=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> ,
    Position=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	Valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	Obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function

					block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
Position	Position	LREAL	Data range	0	Set position value

(3) Function description

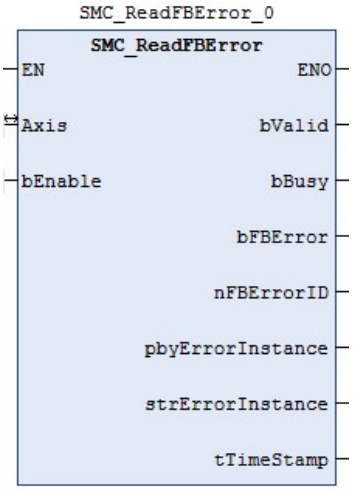
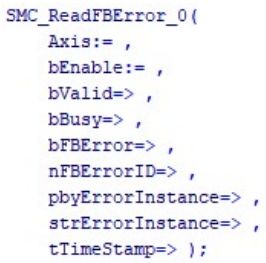
- Enable is true, valid if there are no errors, and Busy output is TURE.
- The output value of Position is the value of Axis.fSetPosition.
- If Enable is changed to FALSE, then Valid and Busy outputs as FALSE. Position remains at the value before FALSE.

3-1-2-22. Read function block error [SMC_ReadFBError]

(1) Instruction overview

Read error messages of the axis function block.

Read function block error [MC_ReadFBError]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
SMC_ReadFBError	Read function block error		


```
SMC_ReadFBError_0(
    Axis:= ,
    bEnable:= ,
    bValid=> ,
    bBusy=> ,
    bFBError=> ,
    nFBErrorID=> ,
    pbyErrorInstance=> ,
    strErrorInstance=> ,
    tTimeStamp=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
bEnable	Valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
bValid	Obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is

					TRUE
bBusy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
bFBError	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
nFBErrorID	Error code	SMC_ERROR	-	0	Error recognition
pbyErrorInstance	Error pointer	POINTER TO BYTE	-	-	Pointer pointing to error reporting function block
strErrorInstance	Error pointer	STRING	-	-	-
tTimeStamp	Time stamp	TIME	-	0	The timestamp at which the error occurred

(3) Function description

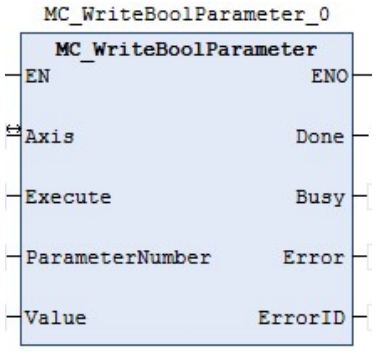
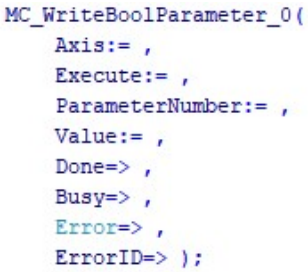
- Enable is true, valid if there are no errors, and Busy output is TURE.
- If there is a function block alarm, the output of bFBError is true.
- If Enable is changed to FALSE, then Valid and Busy outputs as FALSE.

3-1-2-23. Set axis bit parameter [MC_WriteBoolParameter]

(1) Instruction overview

Write parameter values of type BOOL.

Set axis bit parameter [MC_WriteBoolParameter]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_WriteBoolParameter	Set the axis bit parameter		


```
MC_WriteBoolParameter_0(
    Axis:= ,
    Execute:= ,
    ParameterNumber:= ,
    Value:= ,
    Done=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
ParameterNumber	Parameter number	DINT	Positive number, 0	0	Parameter ID
Value	Parameter value	BOOL	TRUE/FALSE	FALSE	-
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	True if the parameter value has been successfully written
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

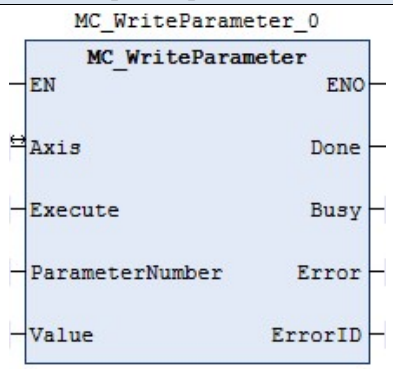
- Sets the bit parameter of the axis through MC_WriteBoolParameter, and the instruction is triggered at the rising edge of Execute. Instructions can be used multiple times without affecting each other.

3-1-2-24. Set the axis parameter [MC_WriteParameter]

(1) Instruction overview

Write the specified parameter value.

Set axis parameter [MC_WriteParameter]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_WriteParameter	Set the axis parameter		MC_WriteParameter_0(Axis:= , Execute:= , ParameterNumber:= , Value:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
ParameterNumber	Parameter number	DINT	Positive value, 0	0	Parameter ID
Value	Parameter value	LREAL	Data range	-	Write the value that needs to be set
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	True if the parameter value has been successfully written
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

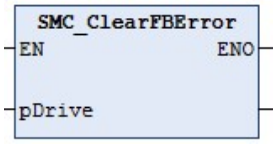
- Set the axis parameter through MC_WriteParameter, the instruction is triggered at the rising edge of Execute. Instructions can be used multiple times without affecting each other.

3-1-2-25. Clear the error [SMC_ClearFBError]

(1) Instruction overview

Clear the historical error information of the function block.

Clear the error [SMC_ClearFBError]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
SMC_ClearFBError	Clear error		SMC_ClearFBError(pDrive:=)

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
pDrive	Axis pointer	POINTR TO AXIS_REF_SM3	-	-	Mapping to the axis
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
SMC_ClearFBError	Valid	BOOL	TRUE/FALSE	FALSE	Clear error is true

(3) Function description

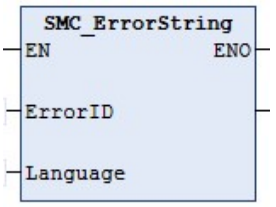
- When an error occurs on the axis, after calling the reset function block to reset the axis, it is necessary to call the function block to clear the historical error status of the axis. The rising edge of Execute will trigger the execution of this instruction.

3-1-2-26. Read the error [SMC_ErrorString]

(1) Instruction overview

Read the error description information corresponding to the error code.

Read the error [SMC_ClearFBError]			
Execution condition	-	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
SM3_Error.SMC_ErrorString	Read the error		SMC_ErrorString(ErrorID:= , Languag

(2) Related variables

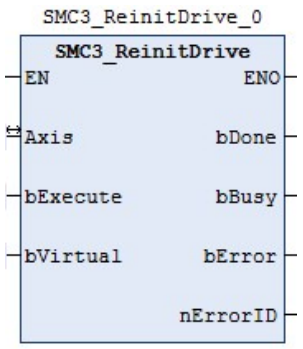
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
ErrorID	Error code	SMC_ERROR	-	-	Error recognition
Language	Language	-	-	-	Required language
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
SMC_ErrorString	Error	STRING(100)	-	-	Error description information

3-1-2-27. Recall driver/axis [SMC3_ReinitDrive]

(1) Instruction overview

This function block calls the driver/axis again. This means that the startup phase runs again and the application cannot control the driver until the function block is set to bDone:=TRUE.

Recall driver/axis [SMC3_ReinitDrive]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
SMC3_ReinitDrive	Recall driver/axis		SMC3_ReinitDrive_0(Axis:= , bExecute:= , bVirtual:= , bDone=> , bBusy=> , bError=> , nErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
bExecute	Valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
bVirtual	Language	BOOL	TRUE/FALSE	FALSE	Input determines whether an axis actually exists or is simulated
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
bDone	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
bBusy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
bError	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
nErrorID	Error code	SMC_ERROR	-	0	Error recognition

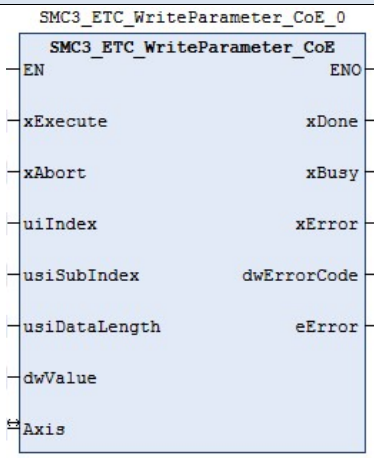
(3) Function description

- Used to reset the drive and reconnect with the controller.
- Input bvirtual to determine whether an axis truly exists or is simulated. If the input bVirtual is set to TRUE, the axis is set to virtual mode. Then, it will be replaced by simulations similar to virtual drive devices. This has no impact on the fieldbus device, it will continue to operate as usual, but will not receive messages from or send messages to the actual device.

3-1-2-28. Set the axis COE parameter [SMC3_ETC_WriteParameter_CoE]

(1) Instruction overview

Set the axis COE parameter [SMC3_ETC_WriteParameter_CoE]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Drive_ETC		

Instruction	Name	Graphic representation	ST language
SMC3_ETC_WriteParameter_CoE	Set the axis COE parameter		SMC3_ETC_WriteParameter_CoE_0(xExecute:= , xAbort:= , uiIndex:= , usiSubIndex:= , usiDataLength:= , dwValue:= , Axis:= , xDone=> , xBusy=> , xError=> , dwErrorCode=> , eError=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
xExecute	Valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
XAbort	Abort	BOOL	TRUE/FALSE	FALSE	Terminate an ongoing write request
uiIndex	Object index	UINT	-	-	Object index, eg. 16#6060
usiSubInde	Object subindex	USINT	-	-	Object subindex, eg. 0
usiDataLength	Write in data length	USINT	-	-	The length of data written in bytes (1-4)
dwValue	Write value	DWORD	-	-	Write in value DWORD
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
bDone	Completed	BOOL	TRUE/FALSE	FALSE	TRUE after execution is completed
bBusy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet

					ended
bError	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
nErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Application

Modify the value of 607D:

```
VAR
   SMC3_ETC_WriteParameter_CoE_0:   SMC3_ETC_WriteParameter_CoE;
   SMC3_ETC_WriteParameter_CoE_1:   SMC3_ETC_WriteParameter_CoE;
   WRITE: BOOL;
   SMC3_ReinitDrive: SMC3_ReinitDrive;
END_VAR
```

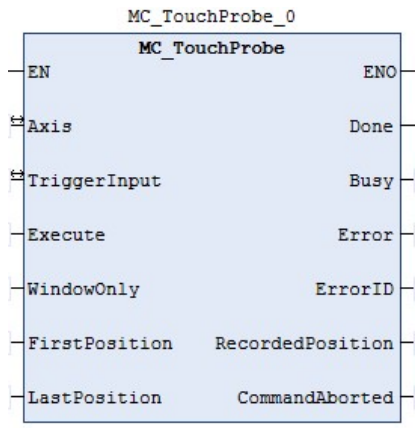
```
SMC3_ETC_WriteParameter_CoE_0(
   xExecute: = WRITE, //trigger at the rising edge
   xAbort: = ,
   uiIndex: =16#607D , //object index such as 16#6060
   usiSubIndex: =1 , //object subindex such as 0
   usiDataLength: =4 , // the length of data written in bytes (1-4)
   dwValue: = 16#8000, //write in value DWORD, can customize DWORD variables
   Axis: =SM_Drive_GenericDSP402 , // SoftMotion axis
   xDone=> ,
   xBusy=> ,
   xError=> ,
   dwErrorCode=> ,
   eError=> );
SMC3_ETC_WriteParameter_CoE_1(
   xExecute: = WRITE,
   xAbort: = ,
   uiIndex: =16#607D ,
   usiSubIndex: =2 ,
   usiDataLength: =4 ,
   dwValue: = 16#8000,
   Axis: =SM_Drive_GenericDSP402 ,
   xDone=> ,
   xBusy=> ,
   xError=> ,
   dwErrorCode=> ,
   eError=> );
```

3-1-2-29. Probe [MC_TouchProbe]

(1) Instruction overview

This function block is used to record the position of the axis when a triggering event occurs.

Probe [MC_TouchProbe]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphic representation	ST language
MC_TouchProbe	Enable probe		

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	-	-	Specified axis
TruggerInput	Trigger signal	TRIIGGER_ REF	-	-	Reference trigger signal source
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
WindowOnly	Trigger window	BOOL	TRUE/FALSE	FALSE	
FirstPosition	Parameter value	LREAL	-	0	The starting position for receiving triggering events
LastPosition	Parameter value	LREAL	-	0	The final location to receive the triggering event
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	Completed	BOOL	TRUE/FALSE	FALSE	True if the parameter value has been successfully written
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
RecordedPosition	Trigger record location	LREAL	-	0	Current position when triggered
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	The current instruction has been interrupted and set to TRUE

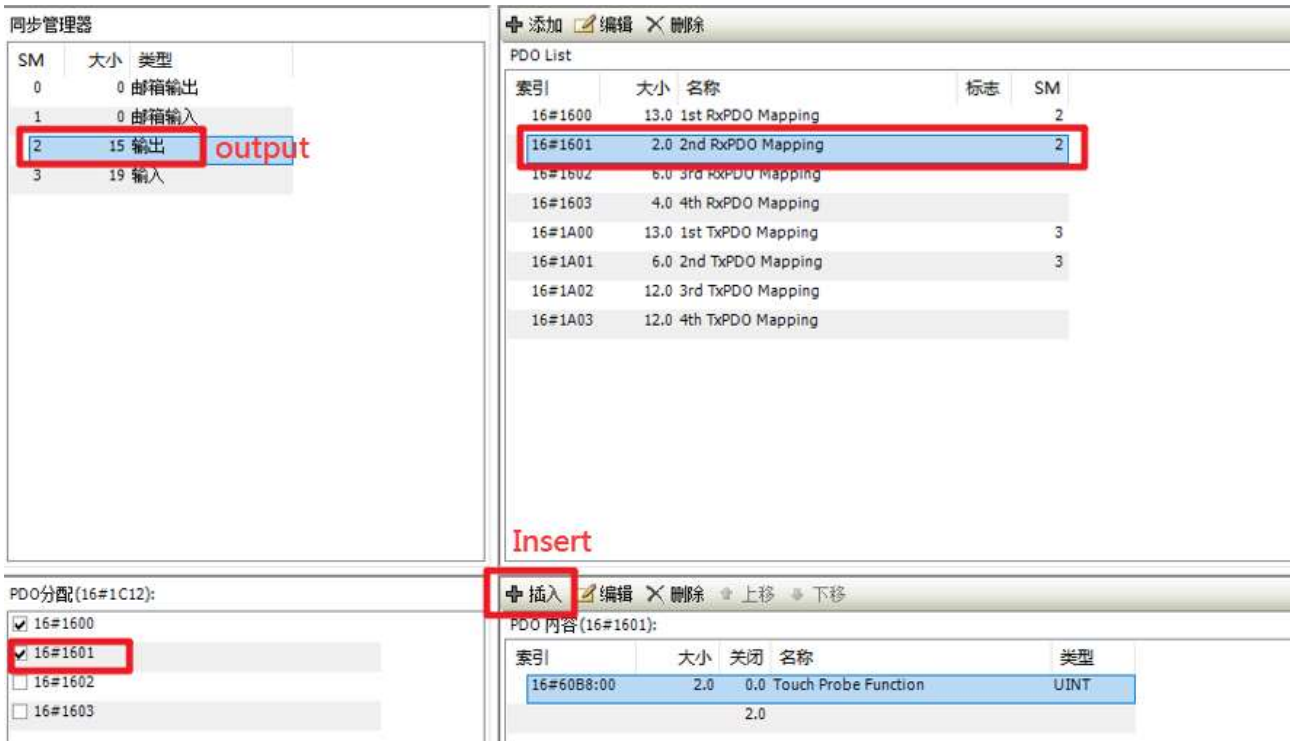
(3) Function description

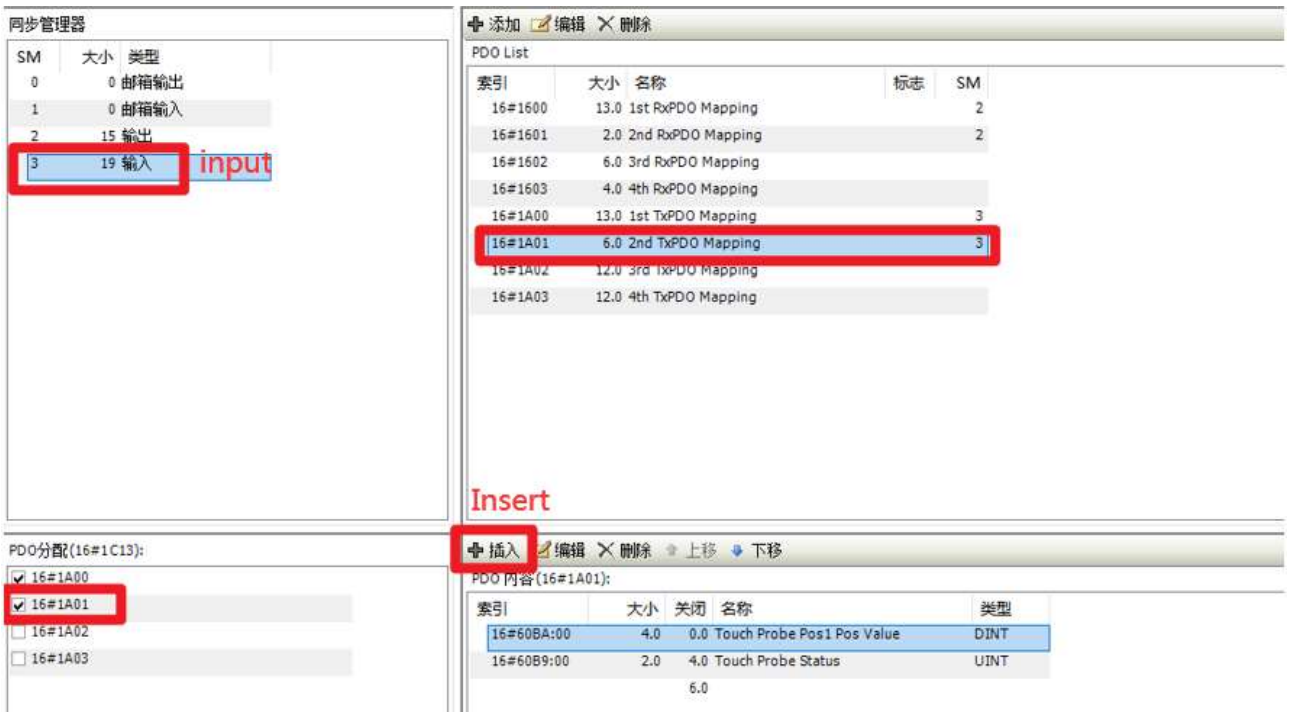
- Record the current position of the running axis when the TruggerInput signal is triggered through the function block MC_TouchProbe.
- When the driver is locked: the driver will collect the locking signal at the recording position and wait until the controller.

(4) Application

Example 1: Taking Xinje DS5C servo as an example, to achieve locking position function of probe 1 at the rising edge.

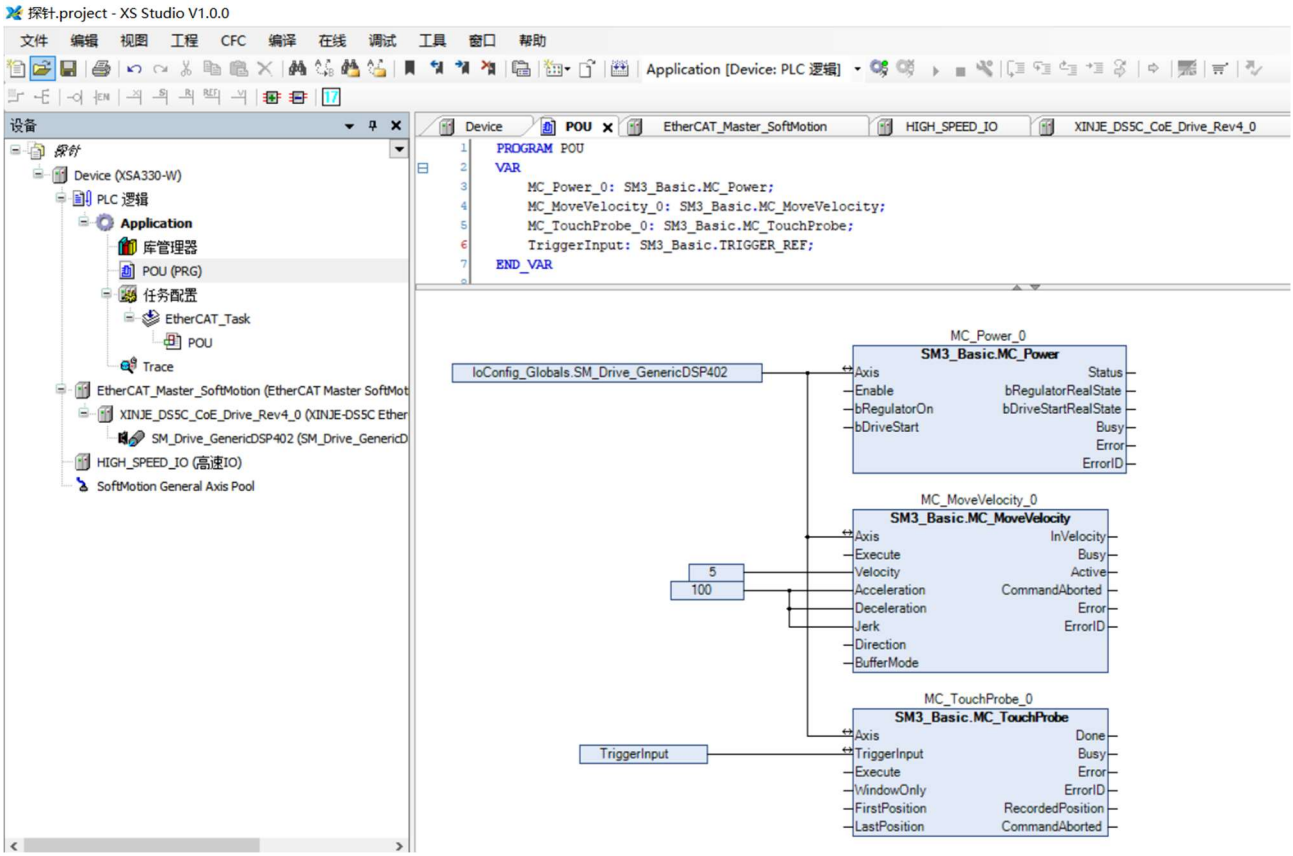
Programming:
(1) Check 'Enable Expert Settings', select 1601 and 1A01 in PDO Allocation in Expert Process Data, add 60B8h in 1601, and 60BAh and 60B9h in 1a01.



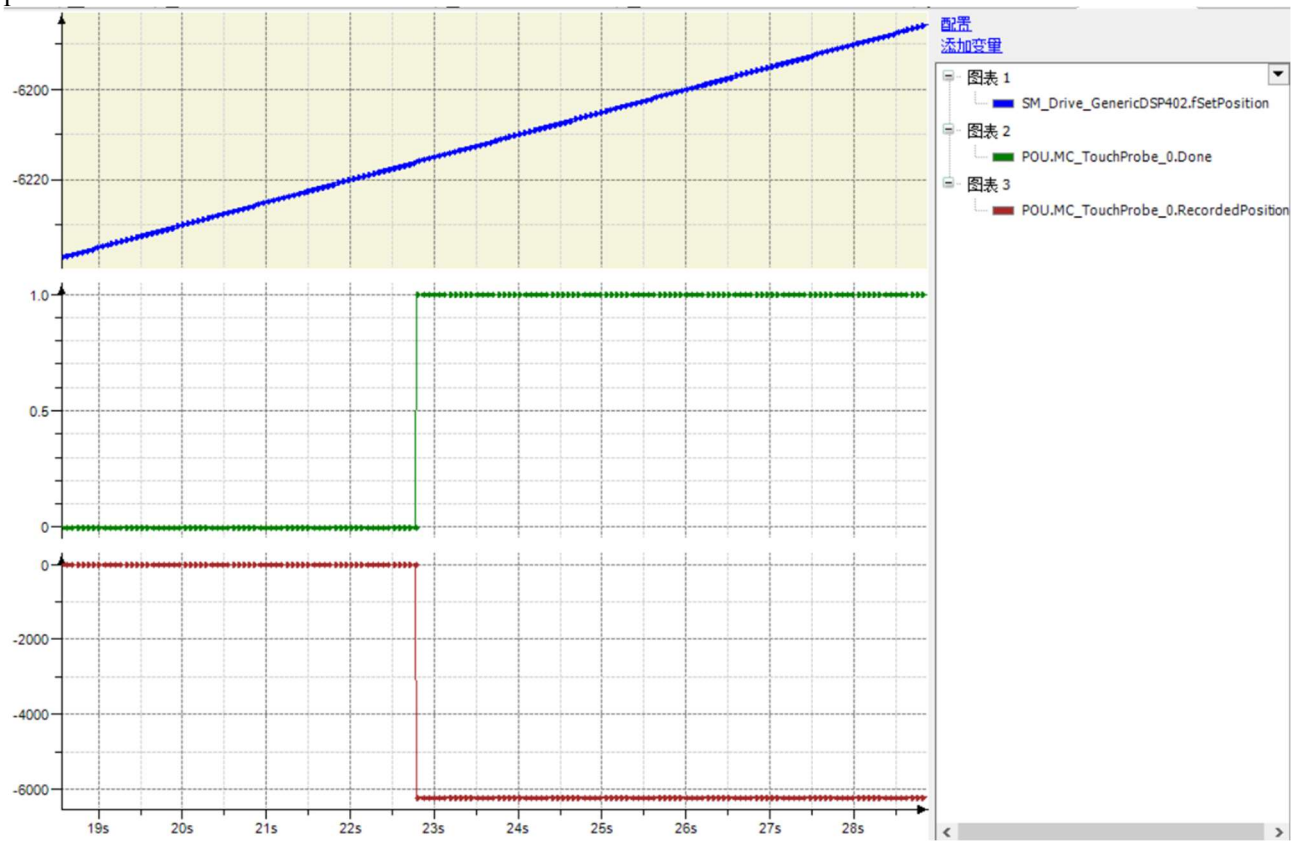


(2) You can see the newly added parameters in "EthercatI/O Mapping".

(3) Set bFastLatching to true, bInput to false, and iTriggerNumber to 0 in the input pin TriggerInput of the function block "MC_TouchProbe". After the function block is turned on, it will open the rising edge of probe 1. After conduction, it can be seen that the value in 60B8h is 17, which is the rising edge of probe 1.



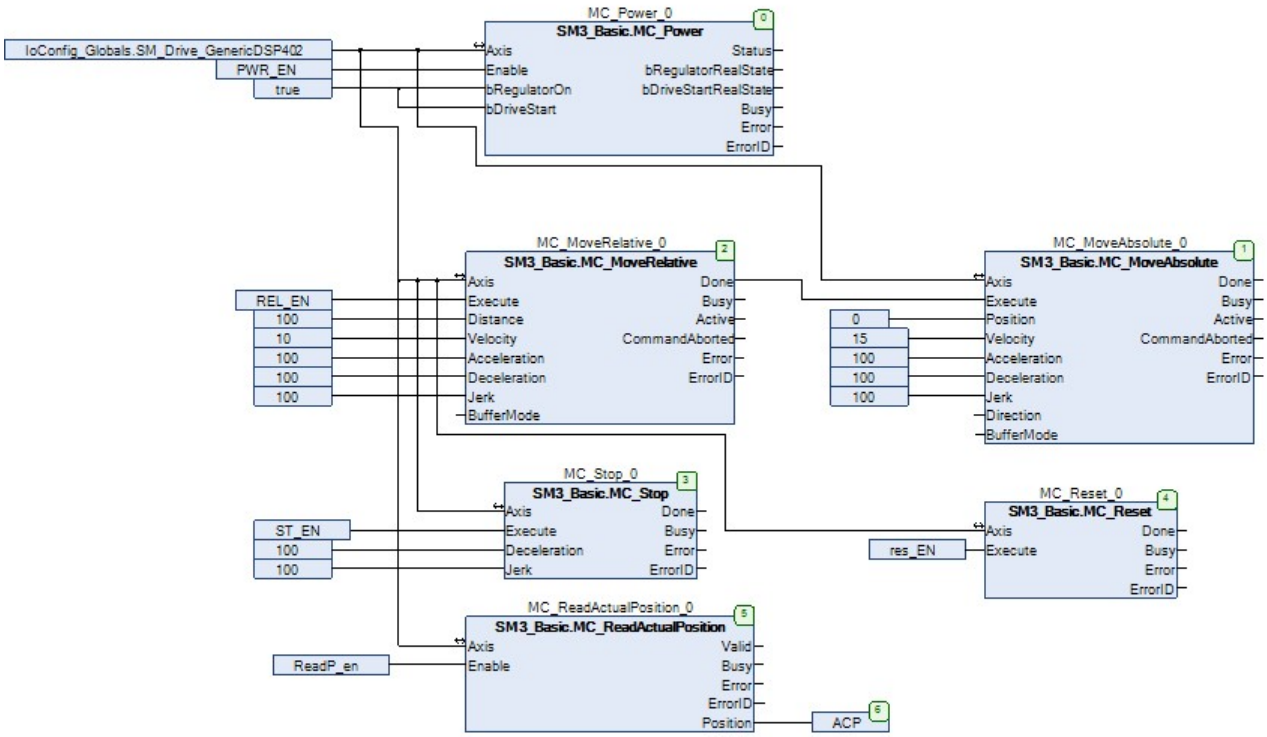
(4) Can be found in 'Trace' that after MC_TouchProbe command done, it will immediately latch the current position.



3-1-3. Single axis function application

Example 1: To achieve axis 0 movement in two segments to reach the designated position of 0 Pluse, the first segment operates at a speed of 10 Pluse/S and an acceleration of 100 Pluse/S^2 for relative motion. Then, relative to the starting position, a further 100 Pluse is run. After reaching the relative position of the target, an absolute motion is run at a speed of 15 Pluse/S and an acceleration of 100 Pluse/S^2 for absolute motion. The motion reaches the target position of 0 Pluse. During the movement process, the real-time position can be read, and the movement can also be stopped. If an error occurs, the axis can also be reset.

```
PROGRAM POU
VAR
    //功能块定义  define the function block
    MC_Power_0: SM3_Basic.MC_Power;
    MC_MoveRelative_0: SM3_Basic.MC_MoveRelative;
    MC_MoveAbsolute_0: SM3_Basic.MC_MoveAbsolute;
    MC_Stop_0: SM3_Basic.MC_Stop;
    MC_Reset_0: SM3_Basic.MC_Reset;
    MC_ReadActualPosition_0: SM3_Basic.MC_ReadActualPosition;
    PWR_EN: BOOL;//使能  enable
    ST_EN: BOOL;//停止  stop
    res_EN: BOOL;//复位  reset
    ReadP_en: BOOL;//实时位置读取  read real-time position
    ACP: LREAL;//读取的位置  read position
END_VAR
```



3-2. Axis group function

3-2-1. Axis group instruction

Instruction	Function
MC_AddAxisToGroup	add the axis to the axis group
MC_RemoveAxisFromGroup	remove the axis from the axis group
MC_UngroupAllAxes	delete all the axes in the axis group
MC_GroupEnable	enable the axis group
MC_GroupDisable	disable the axis group
MC_GroupReset	reset the axis group
MC_GroupSetPosition	set the axis group position
MC_SetCoordinateTransform	coordinate transformation
MC_SetDynCoordTransform	connect two axis groups
MC_GroupContinue	axis group continue
MC_GroupHalt	axis group halt
MC_GroupInterrupt	axis group interrupt
MC_GroupStop	axis group stop
MC_GroupSetOverride	change speed, acceleration, or active and controlled actions
MC_SetKinTransform	kinematic coordinate system conversion
MC_MoveCircularAbsolute	circular move to absolute position
MC_MoveCircularRelative	circular move to relative position
MC_MoveDirectAbsolute	move to absolute position
MC_MoveDirectRelative	move to relative position
MC_MoveLinearAbsolute	linear move to absolute position
MC_MoveLinearRelative	linear move to relative position
MC_GroupReadActualPosition	read actual position
MC_GroupReadActualVelocity	read actual speed
MC_GroupReadConfiguration	read parameters
MC_GroupReadError	read error
MC_GroupReadStatus	read status
SMC_StartupAxisGroup	startup the axis group
SMC_GroupPower	power on the axis group
SMC_GroupInterruptAt	break assignment
SMC_GroupEnableResumeAfterError	Resume motion after axis error
SMC_GroupJog	axis group jog run
SMC_GroupWait	axis group wait

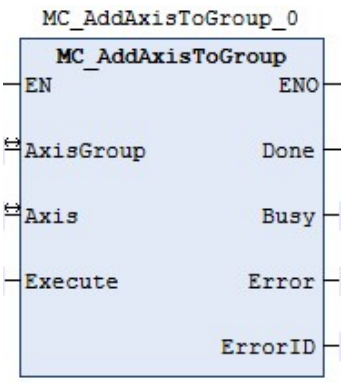
3-2-2. Axis group instructions

3-2-2-1. Add axis to axis group [MC_AddAxisToGroup]

(1) Instruction overview

Add an axis to the axis group using command mode.

Add axis to axis group [MC_AddAxisToGroup]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_AddAxisToGroup	Add axis to axis group		MC_AddAxisToGroup_0(AxisGroup:= , Axis:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
Axis	axis	AXIS_REF_SM3	-	-	specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- The instruction is used to add axis to the specified axis group and bind them together in the relationship of the axis group.
- When the Done variable of the instruction becomes TRUE, it indicates that the axis has been successfully added to the axis group. Note that setting Execute to FALSE does not remove the axis from the axis group.

If you need to remove the axis from the axis group, you need to use MC_ RemoveAxisFromGroup instruction.

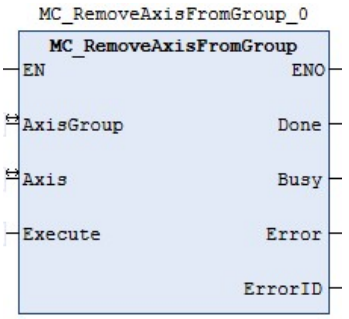
- This command can only be executed when the axis group is in the GroupDisabled state. If this command is executed after the axis group is enabled, an error will be reported.

3-2-2-2. Remove axis from the axis group [MC_RemoveAxisFromGroup]

(1) Instruction overview

Remove the axis from the axis group.

Remove axis from axis group [MC_RemoveAxisFromGroup]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_RemoveAxisFromGroup	remove axis from the axis group		MC_RemoveAxisFromGroup_0(AxisGroup:= , Axis:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
Axis	axis	AXIS_REF_SM3	-	-	specified axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

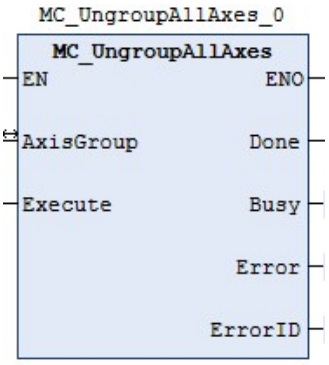
- This command can only be executed when the axis group is not enabled. If this command is executed after the axis group is enabled, an error will be reported.
- When the Done variable of the instruction becomes TRUE, it indicates that the axis has been successfully removed from the axis group.

3-2-2-3. Ungroup all the axes [MC_UngroupAllAxes]

(1) Instruction overview

Remove all the axes contained in a certain axis group and dissolve the axis group.

Ungroup axis group [MC_UngroupAllAxes]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_UngroupAllAxes	ungroup the axis group		MC_UngroupAllAxes_0(AxisGroup:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	Valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

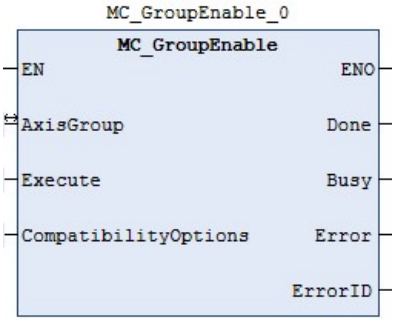
- Remove all axes from the axis group. If the status of the axis group is not GroupDisabled, an error will be generated and the axis will not be removed from the axis group.

3-2-2-4. Enable the axis group [MC_GroupEnable]

(1) Instruction overview

Enable the axis group.

Enable the axis group [MC_GroupEnable]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupEnable	Enable the axis group		MC_GroupEnable_0(AxisGroup:= , Execute:= , CompatibilityOptions:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
CompatibilityOptions	compatibility option	SMC_AXIS_GROUP_COMPATIBILITY_OPTIONS	-	-	Parameters that exist for compatibility with previous versions
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended

Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

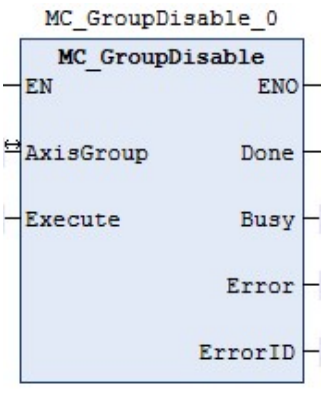
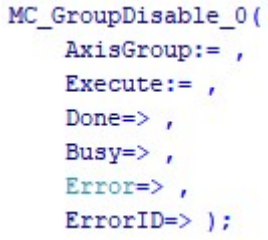
- Before calling the motion control and other commands of the axis group, SMC_GroupPower or MC_Power needs to be called to enable the axis group.
- When the Done variable of the instruction becomes TRUE, it indicates that the axis group has successfully switched to the Standby state.
- The types of axes that can be specified in the axis group can only be "servo axis" and "virtual servo axis". If other axis types are specified, an exception will occur.
- When executing this command, all axes under the axis group must be in a stop state.
- If there are axes that already belong to other axis groups and have already been enabled, MC_GroupEnable cannot be executed, the command will cause an unexpected error.
- The conditions for invalidating the axis group include: executing MC_GroupDisable command, switch to program mode to stop running, and start MC trial run.

3-2-2-5. Disable the axis group [MC_GroupDisable]

(1) Instruction overview

The axis group is switched to the Disabled state, and motion control of the axis group cannot be performed in this state.

Disable the axis group [MC_GroupDisable]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupEnable	Disable the axis group		


```
MC_GroupDisable_0(
    AxisGroup:= ,
    Execute:= ,
    Done=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial	Description

				value	
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

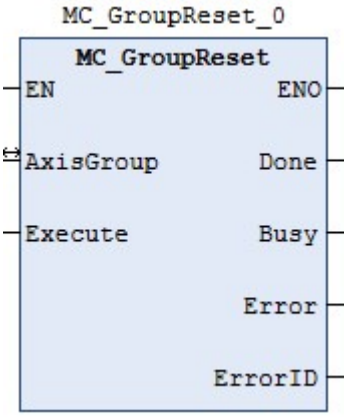
- This command switches the specified axis group to the GroupDisable state.
- When the axis group status changes to GroupDisable, the cache instructions for the specified AxesGroup will be cleared.

3-2-2-6. Axis group reset [MC_GroupReset]

(1) Instruction overview

Release the abnormal state of the axis group and axis.

Reset the axis group [MC_GroupReset]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupReset	axis group reset		


```
MC_GroupReset_0(
    AxisGroup:= ,
    Execute:= ,
    Done=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

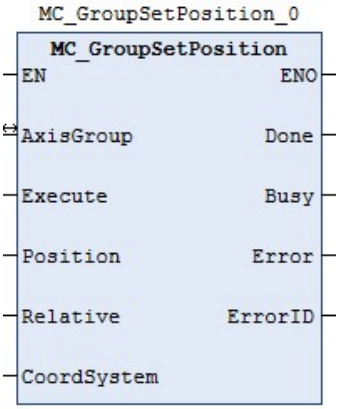
- On the rising edge of Execute, handle the exceptions of the AxesGroup specified axis group in the GroupEnable state and the exceptions of the axis to which the axis group belongs. The exceptions that can be resolved include "minor faults" in the axis and axis group, "abnormal monitoring information", and driver error reset.
- Regardless of whether the axis is in servo ON or servo OFF state, abnormal release processing can be executed.
- For axes with driver errors, the driver error reset process should be executed first, and then the exception relief process should be executed.
- The driver error reset processing can choose to clear the driver error or remain unchanged within the axis parameter [driver error reset monitoring time]. Driver error reset is performed simultaneously on all axes belonging to the axis group.
- The exception object that can be resolved is the exception that occurs when the rising edge of Execute is activated. Cannot perform exception resolution on exceptions that occur during the exception resolution process.
- If the command is executed during an incorrect deceleration stop of the axis group, it cannot be executed because abnormal release cannot be performed before the axis stops. In addition, if an abnormal error occurs on the axis itself in the axis group, it cannot be resolved through this command.

3-2-2-7. Set axis group position [MC_GroupSetPosition]

(1) Instruction overview

Used to set the command positions of each axis in the axis group.

Set axis group position [MC_GroupSetPosition]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupSetPosition	set axis group position		MC_GroupSetPosition_0(AxisGroup:= , Execute:= , Position:= , Relative:= , CoordSystem:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
Position	position	SMC_POS_REF	data range	0	axis target position
Relative	position mode	BOOL	TRUE/FALSE	FALSE	Relative position mode=True, absolute position mode=False (default)
CoordSystem	Applied coordinate system	SMC_COORD_SYSTEM	-	-	Applied coordinate system
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

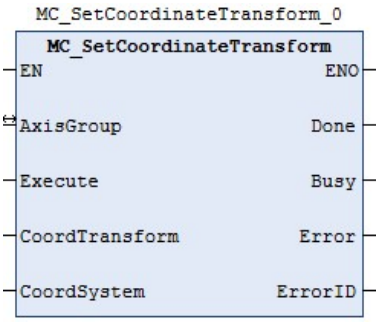
- Set the coordinate position under the specified coordinate system in the axis group.
- The instruction is executed in the GroupStandby state of the axis group, and cannot be executed in the dynamic coordinate system or executed with MC_GroupContinue instruction simultaneously.

3-2-2-8. Coordinate transform [MC_SetCoordinateTransform]

(1) Instruction overview

Used to convert different command coordinates of reference coordinate systems.

Coordinate transform [MC_SetCoordinateTransform]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_SetCoordinateTransform	Coordinate transform		MC_SetCoordinateTransform_0(AxisGroup:= , Execute:= , CoordTransform:= , CoordSystem:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
CoordTransform	coordinate transform	MC_COORD_REF	-	-	Coordinate transformation, i.e. the product coordinate system (PCS1 or PCS2) or machine coordinate system (MCS) represented by the World Coordinate System (WCS)
CoordSystem	Applied coordinate system	SMC_COORD_SYSTEM	-	-	Target coordinate system, allowing for conversion of PCS_1, PCS_2 and MCS
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction

					execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

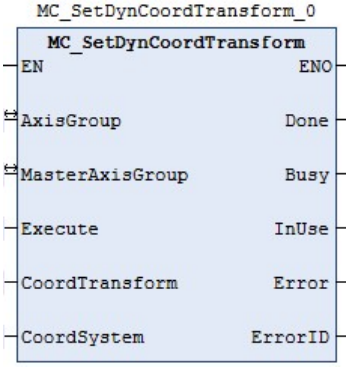
- Set the world coordinate system (WCS) and product/machine coordinate system (PCS */MCS) in the axis group for coordinate system conversion.
- When PCS is a dynamic coordinate system (PCS moves relative to WCS), MC_SetDynCoordTransform needs to be used.
- This command only performs coordinate system conversion and is independent of motion control commands.

3-2-2-9. Dynamic coordinate system conversion [MC_SetDynCoordTransform]

(1) Instruction overview

When the specified coordinate system moves relative to WCS, this command needs to be called to achieve coordinate system conversion.

Dynamic coordinate transform [MC_SetDynCoordTransform]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_SetDynCoordTransform	Dynamic coordinate system conversion		MC_SetDynCoordTransform_0(AxisGroup:= , MasterAxisGroup:= , Execute:= , CoordTransform:= , CoordSystem:= , Done=> , Busy=> , InUse=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
MasterAxisGroup	master axis group	AXIS_GROUP_REF_SM3	-	-	specified master axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the

					function block
CoordTransform	Coordinate system to be converted	MC_COORD_REF	-	-	The tool coordinate system of the spindle group is relative to the coordinates and direction of PCS
CoordSystem	Coordinate System	SMC_COORD_SYSTEM	-	-	The PCS coordinate system to be converted (PCS1 or PCS2)
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
InUse	Reference system	BOOL	-	-	Indicates the dynamic coordinate system that the axis group still needs to reference
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

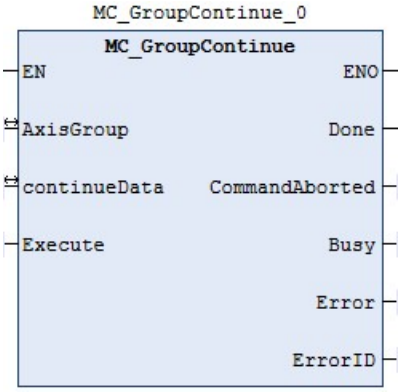
- Usually used when using dynamic PCS, such as MC_TrackConveyorBelt or MC_TrackRotaryTable is called together.
- Instruction SMC_SetDynCoordTransformEX provides a more universal interface.

3-2-2-10. Axis group continue running [MC_GroupContinue]

(1) Instruction overview

Release the interrupt status of the axis group and continue executing unfinished commands.

Axis group continue running [MC_GroupContinue]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupContinue	axis group continue running		MC_GroupContinue_0(AxisGroup:= , continueData:= , Execute:= , Done=> , CommandAborted=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
continueData	Continuing motion data	SMC_AXIS_GROUP_CONTINUE_DATA	-	-	Axis group position during motion interruption
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

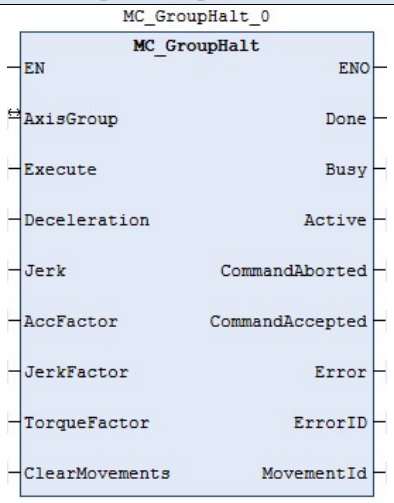
- Release the interrupt status of the axis group and continue executing unfinished commands.

3-2-2-11. Axis group halt [MC_GroupHalt]

(1) Instruction overview

Used to pause the current axis group motion.

Axis group halt [MC_GroupHalt]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupHalt	axis group halt		MC_GroupHalt_0(AxisGroup:= , Execute:= , Deceleration:= , Jerk:= , AccFactor:= , JerkFactor:= , TorqueFactor:= , ClearMovements:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
Deceleration	deceleration	LREAL	positive number	0	Maximum resultant deceleration [command unit/s^2]
Jerk	jerk speed	LREAL	positive number	0	Maximum resultant jerk speed[command unit/s^3]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk factor	LREAL	0-1	1	Jerk speed factor, multiply the maximum velocity of each axis by this jerk factor, and the value is between [0,1]
VAR_OUTPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution

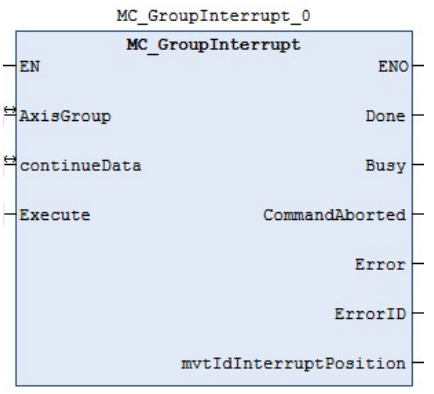
					is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
CommandAccepted	motion accepted	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	True when the motion is being executed or completed

3-2-2-12. Axis group interruption [MC_GroupInterrupt]

(1) Instruction overview

Interrupt the currently moving axis group, which can be done through MC_GroupContinue instruction continues to execute unfinished motion instructions.

Axis group interruption [MC_GroupInterrupt]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupInterrupt	axis group interruption		MC_GroupInterrupt_0(AxisGroup:= , continueData:= , Execute:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=> , mvtIdInterruptPosition=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
continueData	Continuing motion data	SMC_AXIS_GROUP_CONTINUE_DATA	-	-	Motion information when axis group motion is interrupted
VAR_INPUT	Name	Data type	Effective range	Initial value	Description

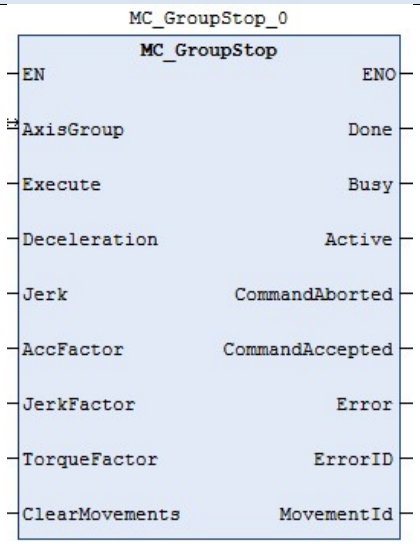
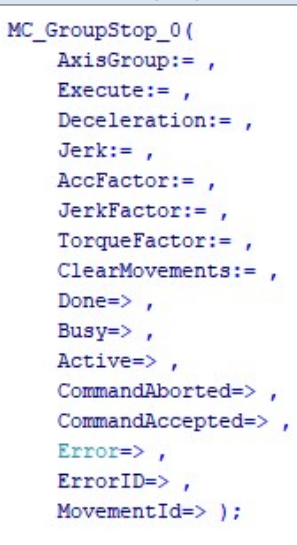
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
mvtIdInterruptPosition	interrupt position ID	SMC_Movement_Id	-	-	The Movement ID corresponding to the interrupt location

3-2-2-13. Axis group stop [MC_GroupStop]

(1) Instruction overview

Stop the movement of the axis group.

Axis group stop [MC_GroupStop]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupStop	axis group stop		


```
MC_GroupStop_0(
    AxisGroup:= ,
    Execute:= ,
    Deceleration:= ,
    Jerk:= ,
    AccFactor:= ,
    JerkFactor:= ,
    TorqueFactor:= ,
    ClearMovements:= ,
    Done=> ,
    Busy=> ,
    Active=> ,
    CommandAborted=> ,
    CommandAccepted=> ,
    Error=> ,
    ErrorID=> ,
    MovementId=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_	-	-	specified axis group

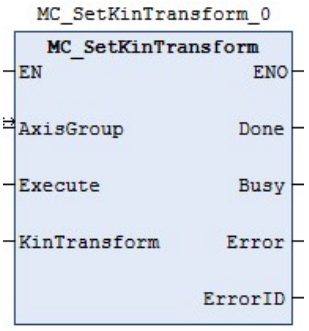
		SM3			
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
Deceleration	deceleration	LREAL	positive number	0	Maximum resultant deceleration [command unit/s^2]
Jerk	jerk speed	LREAL	positive number	0	Maximum resultant jerk speed [command unit/s^3]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk factor	LREAL	0-1	1	Jerk speed factor, multiply the maximum velocity of each axis by this jerk factor, and the value is between [0,1]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
CommandAccepted	motion accepted	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	True when the motion is being executed or completed

3-2-2-14. Kinematic coordinate transformation [MC_SetKinTransform]

(1) Instruction overview

Set the kinematic transformation of the axis group from the ACS coordinate system to the MCS coordinate system.

Kinematic coordinate transformation [MC_SetKinTransform]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_SetKinTransform	Kinematic coordinate transformation		MC_SetKinTransform_0(AxisGroup:= , Execute:= , KinTransform:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	The rising edge of the input value will initiate the execution of the function block
KinTransform	Kinematic transformation values	TRAFO.MC_KIN_REF_SM3	-	-	Kinematic transformation
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

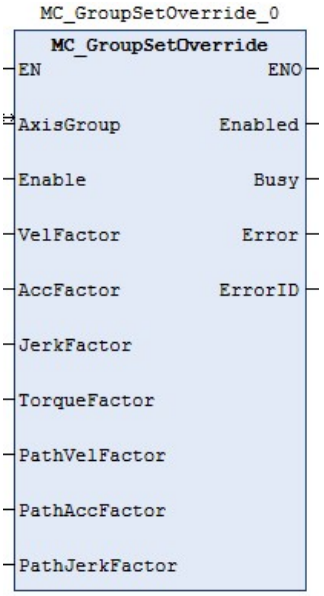
- Set up kinematic transformations between the Axis Coordinate System (ACS) and the Machine Coordinate System (MCS).
- After executing this function block, the tool offset will be reset.

3-2-2-15. Set axis group overshoot value [MC_GroupSetOverride]

(1) Instruction overview

When the axis group is in the Moving state, change the motion speed of the axis group.

Set axis group overshoot value [MC_GroupSetOverride]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupSetOverride	set the axis group overshoot value		MC_GroupSetOverride_0(AxisGroup:= , Enable:= , VelFactor:= , AccFactor:= , JerkFactor:= , TorqueFactor:= , PathVelFactor:= , PathAccFactor:= , PathJerkFactor:= , Enabled=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
VelFactor	speed factor	LREAL	0-1	1	Speed factor, the maximum speed of each axis multiplied by this speed factor, with a value between [0, 1]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk speed factor	LREAL	0-1	1	jerk velocity factor, multiply the maximum velocity of each axis by this jerk factor, and the value is between [0,1]
PathVelFactor	Resultant velocity	LREAL	0-1	1	Resultant velocity factor, the maximum

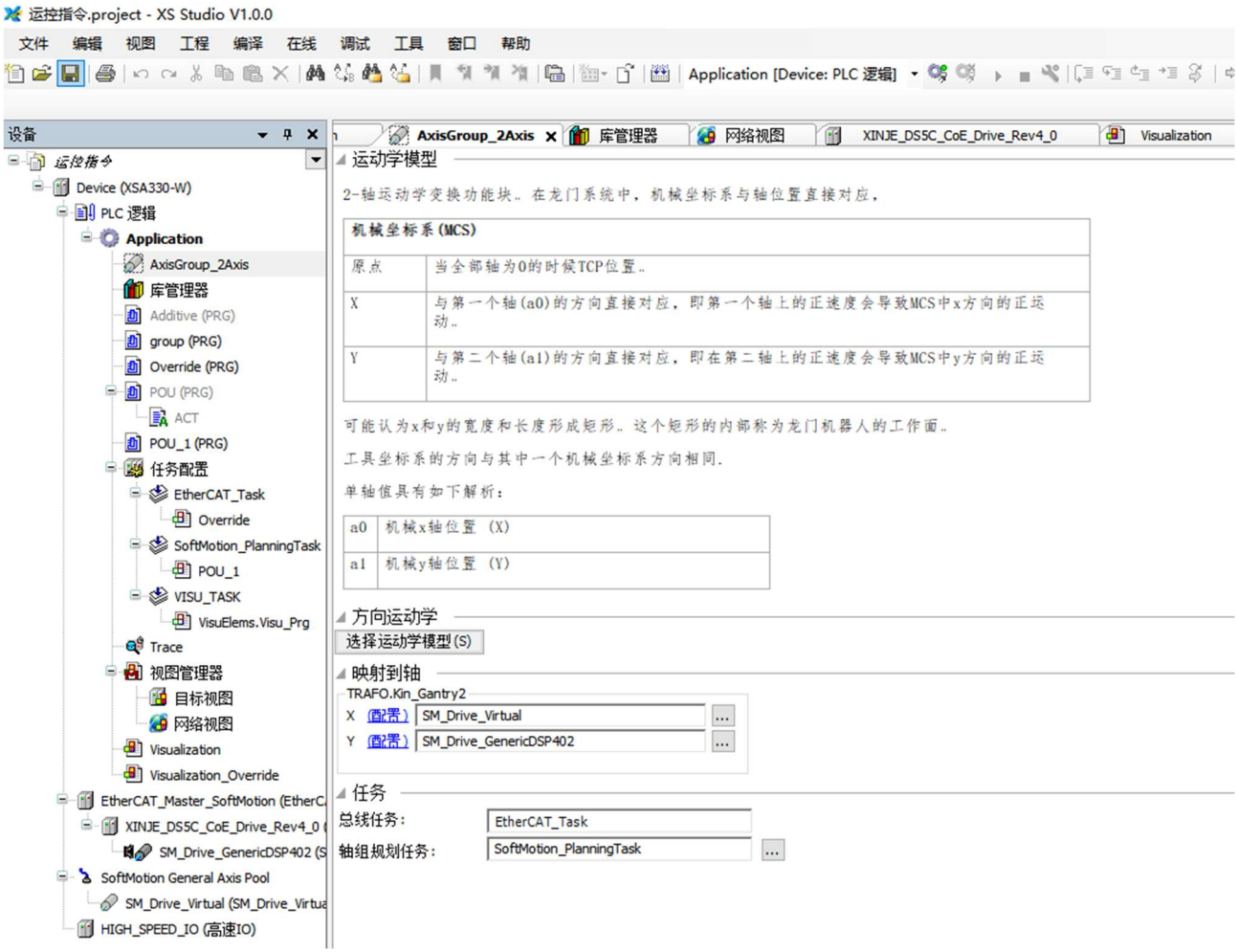
	factor				resultant velocity of the entire axis group's motion trajectory multiplied by this velocity factor, with a value between [0,1]
PathAccFactor	resultant acceleration factor	LREAL	0-1	1	Resultant acceleration factor, the maximum resultant acceleration of the entire axis group motion trajectory multiplied by this resultant acceleration factor, with a value between [0,1]
PathJerkFactor	resultant jerk speed factor	LREAL	0-1	1	The resultant jerk factor is the maximum resultant jerk speed of the entire axis group's motion trajectory multiplied by this resultant jerk factor, with a value between [0,1]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Enabled	completed	BOOL	TRUE/FALSE	FALSE	When the overshoot factor setting completed is TRUE
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

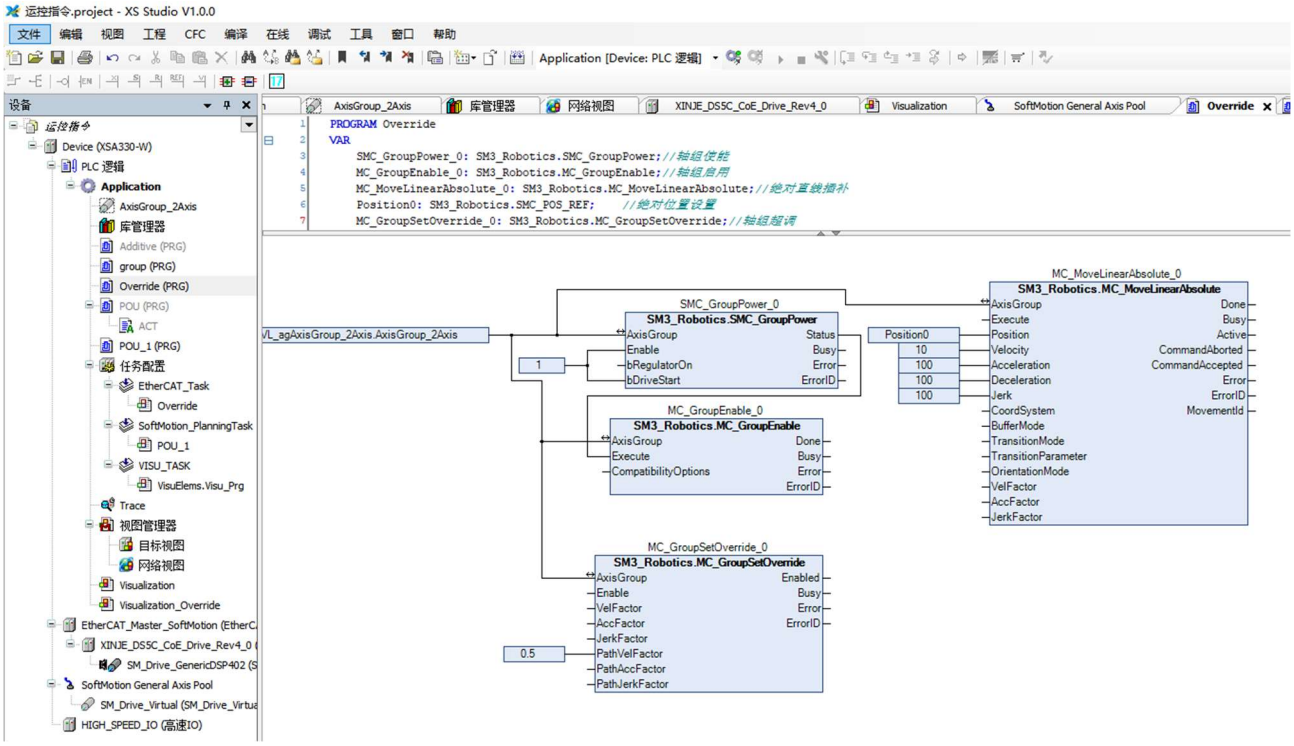
- The axis group can change the speed, acceleration, and acceleration of movement by executing this command in the Moving state.
- The given overshoot factor remains valid until a new value is set. For example, for an axis group in CP motion (even following motion), setting VelFactor or PathVelFactor to 0 will cause a sudden stop in the motion trajectory. If MC_GroupStop is currently in an Active state, an error will be returned.
- MC_GroupStop instruction is not affected by the overshoot factor.
- Re-enable the axis group, and the overshoot factor remains valid.
- The default value of the overshoot factor is 1.
- Reducing AccFactor or JerkFactor can cause position overshoot and may cause damage to the equipment.

(4) Application

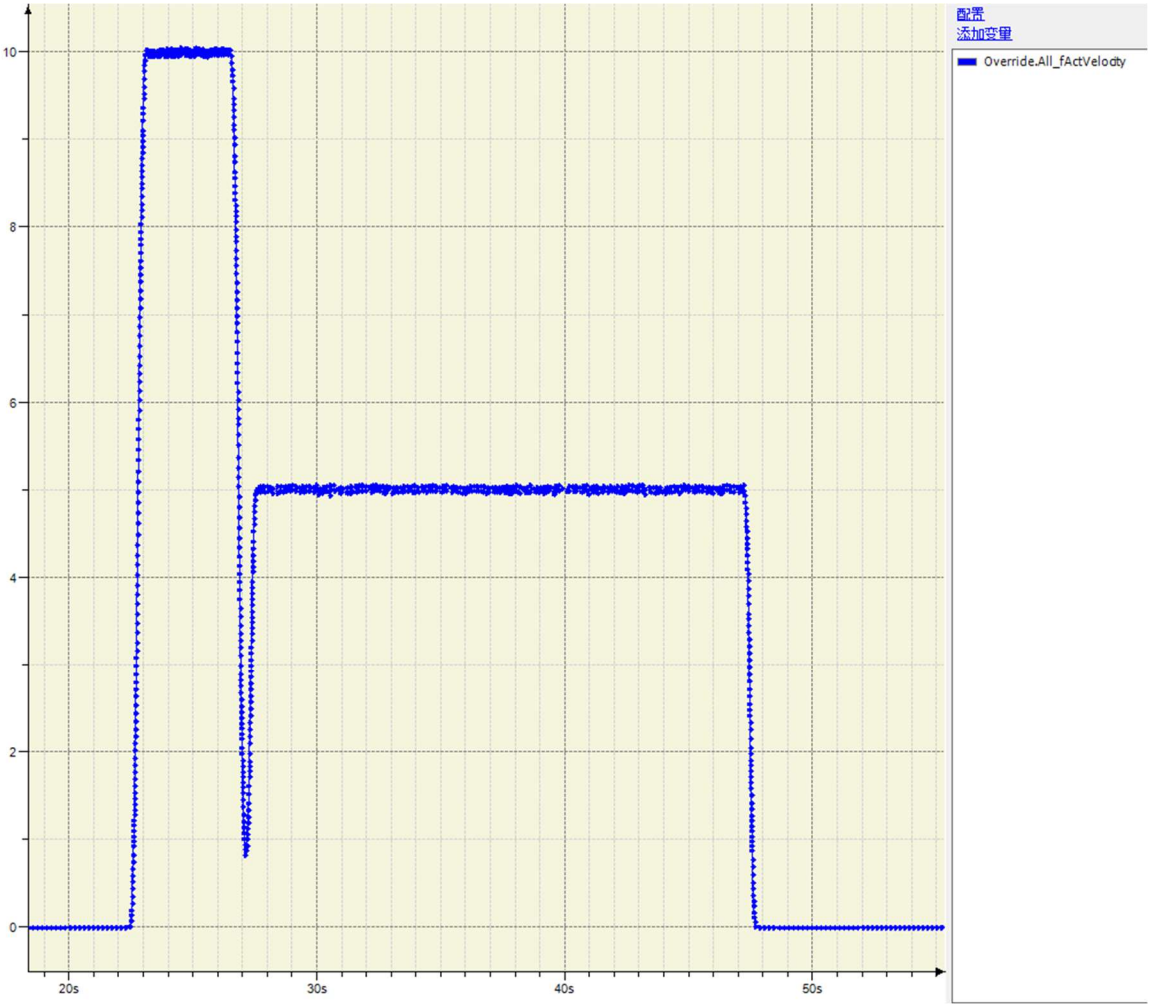
Example 1: Select the kinematic model of the two axes of the gantry, select the virtual axis for the X-axis, and select the Ethercat real axis for the Y-axis. During the execution of the "MC-MoveLinearAbsolute" absolute linear interpolation process, use the "MC_GroupSetOverride" axis group overshoot command to change the axis group speed.



Programming: Set the current positions of both axes to 0, set the target position to (100, 100), perform absolute linear interpolation motion with a starting point of (0,0), and set the axis group speed to 10u/s. Use the axis group overshoot command to change the axis group speed to 5u/s.



In Trace, the current speed of the axis group can be seen. After conducting the axis group overshoot command, the speed changes from 10u/s to 5u/s.

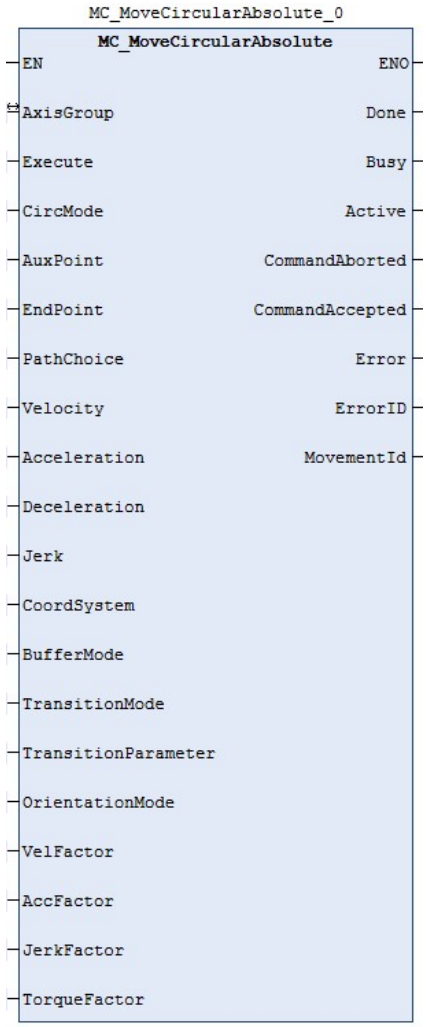


3-2-2-16. Absolute arc interpolation [MC_MoveCircularAbsolute]

(1) Instruction overview

Control the axis group to perform arc interpolation motion in absolute position mode.

Absolute arc interpolation [MC_MoveCircularAbsolute]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_MoveCircularAbsolute	absolute arc interpolation		MC_MoveCircularAbsolute_0(AxisGroup:= , Execute:= , CircMode:= , AuxPoint:= , EndPoint:= , PathChoice:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , OrientationMode:= , VelFactor:= , AccFactor:= , JerkFactor:= , TorqueFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction

CircMode	arc mode	SMC_CIRC_MODE	Border/Center/Radius	-	Specify the method of arc interpolation: Border: Three point arc Center: Center arc Radius: Radius arc
AuxPoint	Auxiliary point	SMC_POS_REF	-	-	Specify auxiliary points in the coordinate system. Refer to CircMode
EndPoint	end point	SMC_POS_REF	-	-	Specify the end position in the coordinate system
PathChoice	direction	MC_CIRC_PATHCHOICE	CLOCKWISE/COUNTER_CLOCKWISE	-	motion direction: CLOCKWISE: clockwise COUNTER_CLOCKWISE: counterclockwise
Velocity	speed	LREAL	0, positive number	0	Max resultant velocity [command unit/s]
Deceleration	deceleration	LREAL	positive number	0	Max resultant deceleration[command unit/s^2]
Jerk	jerk speed	LREAL	positive number	0	Max resultant jerk speed [command unit/s^3]

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
CoordSystem	reference coordinate system	SMC_COORD_SYSTEM	SMC_COORD_SYSTEM	-	reference coordinate system
BufferMode	buffer mode	MC_BUFFER_MODE	-	0	Action when specifying multiple start motion commands
TransitionMode	Corner transition mode	MC_TRANSITION_MODE	TMNone/TMStartVelocity/TMCornerDistance	-	TMNone: No mixing TMStartVelocity: Speed based mixing TMCornerDistance: Distance based mixing
TransitionParameter	Corner transition parameters	array [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL	0, positive number	0	Corner transition parameters
OrientationMode	interpolation positioning mode	SMC_ORIENTATION_MODE	GreatCircle/Axis	-	GreatCircle: Move along the shortest path from the starting position to the target position. In this mode, even if the starting and ending positions are within the specified area, the implemented path may still leave this area. Axis: The positioning axis moves within the specified area from the start

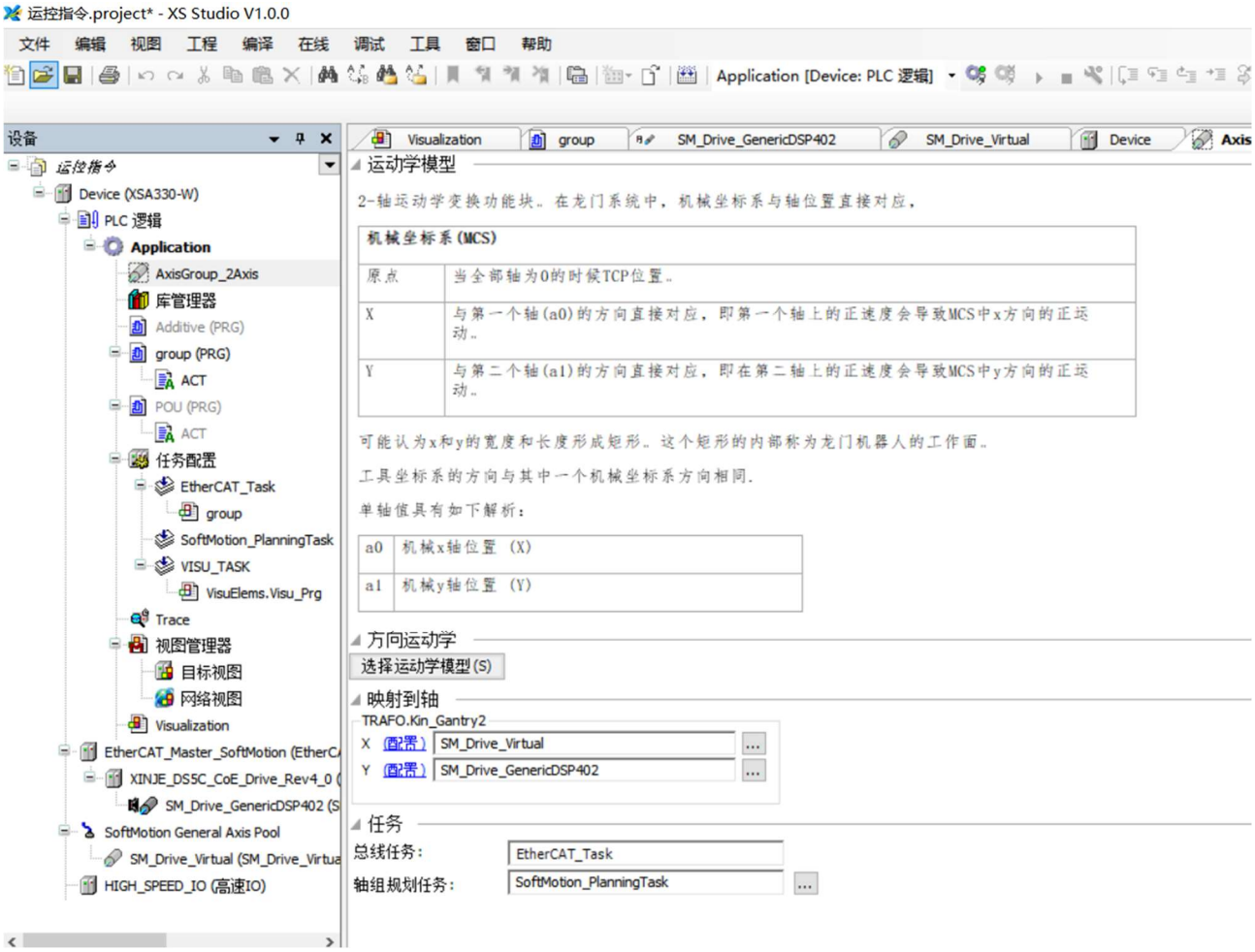
					position to the end position, and not all kinematic transformations support this mode
VelFactor	speed factor	LREAL	0-1	1	Speed factor, the maximum speed of each axis multiplied by this speed factor, with a value between [0, 1]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk speed factor	LREAL	0-1	1	Jerk factor, the maximum velocity of each axis multiplied by this jerk factor, with a value between [0,1]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
CommandAccepted	Motion reception	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	Valid when CommandAccepted or Done is TRUE

(3) Function description

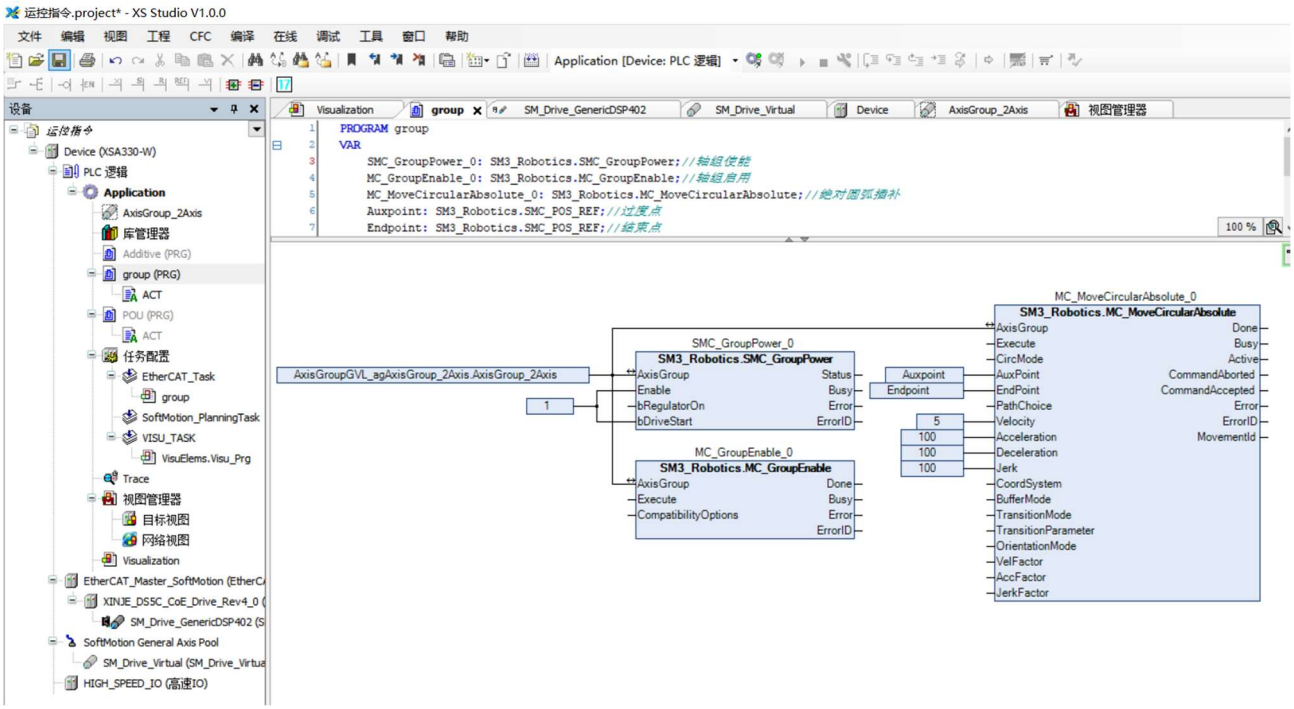
- The rising edge of the Execute triggers the command movement, while the falling edge has no effect on the command movement.
- When using the center arc mode, input the values of parameters AuxPoint [1] and AuxPoint [2] as the distance between the center of the circle and the starting point of the arc; When using the radius arc mode, AuxPoint [1] represents the radius value, and AuxPoint [2] is invalid.
- The numerical values of EndPoint [1]~EndPoint [8] represent the endpoint coordinates of each axis

(4) Application

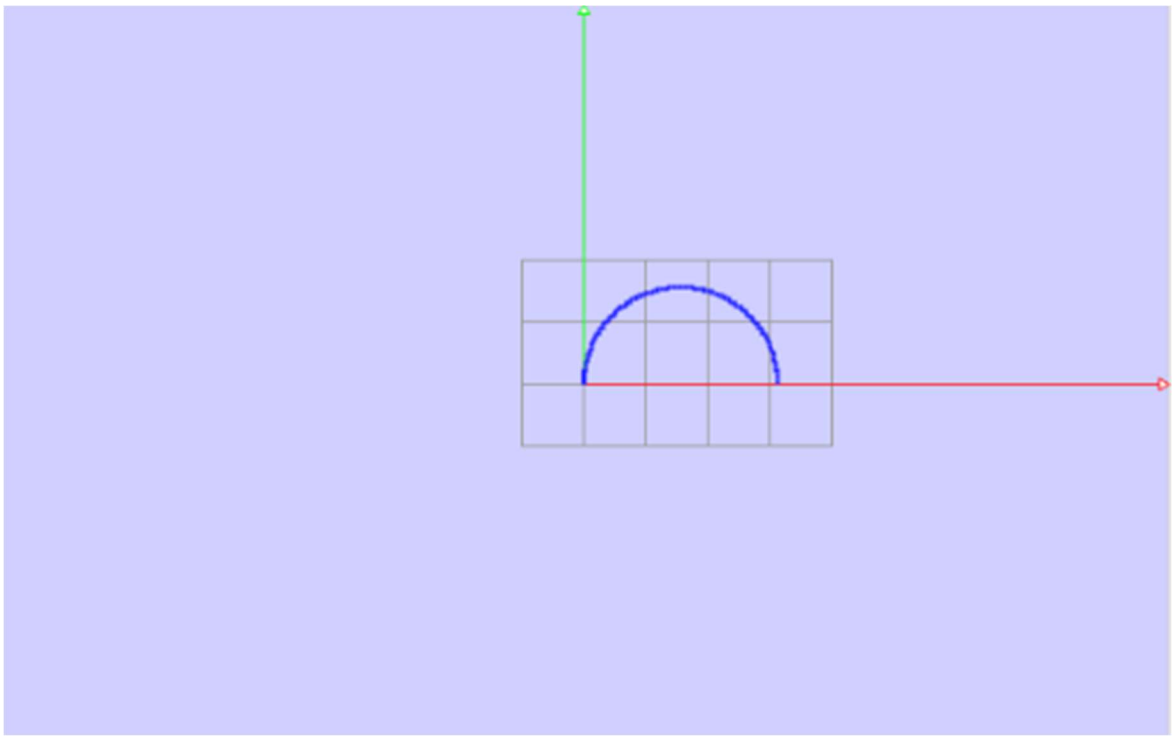
Example 1: Select the kinematic model of the two axes of the gantry, select the imaginary axis for the X-axis, and select the Ethercat real axis for the Y-axis to perform an absolute arc interpolation.



Programming: Set the current positions of both axes to 0, select the arc mode as three points to determine an arc, set AuxPoint to (50, 50), EndPoint to (100, 0), set the speed of both axes to 5, and draw an arc absolutely at the origin (0, 0).



In the view, the arc trajectory can be seen.

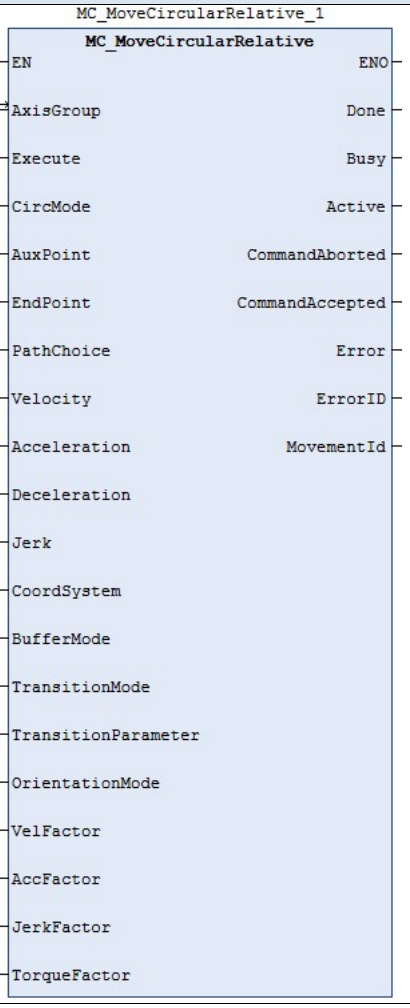


3-2-2-17. Relative arc interpolation [MC_MoveCircularRelative]

(1) Instruction overview

Control the axis group to perform circular interpolation motion in relative position mode.

Relative arc interpolation [MC_MoveCircularRelative]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_MoveCircularRelative	Relative arc interpolation		MC_MoveCircularRelative_0(AxisGroup:= , Execute:= , CircMode:= , AuxPoint:= , EndPoint:= , PathChoice:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , OrientationMode:= , VelFactor:= , AccFactor:= , JerkFactor:= , TorqueFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction

CircMode	arc mode	SMC_CIRC_MODE	Border/Center/Radius	-	Specify the method of arc interpolation: Border: Three point arc Center: Center arc Radius: Radius arc
AuxPoint	Auxiliary point	SMC_POS_REF	-	-	Specify auxiliary points in the coordinate system. Refer to CircMode
EndPoint	end point	SMC_POS_REF	-	-	Specify the end position in the coordinate system
PathChoice	direction	MC_CIRC_PATHCHOICE	CLOCKWISE/COUNTER_CLOCKWISE	-	motion direction: CLOCKWISE: clockwise COUNTER_CLOCKWISE: counterclockwise
Velocity	speed	LREAL	0, positive number	0	Max resultant speed [command unit/s]
Deceleration	deceleration	LREAL	positive number	0	Max resultant deceleration [command unit/s^2]
Jerk	jerk speed	LREAL	positive number	0	Max resultant jerk speed [command unit/s^3]
CoordSystem	reference coordinate system	SMC_COORD_SYSTEM	SMC_COORD_SYSTEM	-	reference coordinate system
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
BufferMode	buffer mode	MC_BUFFER_MODE	-	0	specify the multiple start motion commands action
TransitionMode	Corner transition mode	MC_TRANSITION_MODE	TMNone/TMStartVelocity/TMCornerDistance	-	TMNone: no mixing TMStartVelocity: mixed based on speed TMCornerDistance: mixed based on distance
TransitionParameter	Corner transition parameters	array [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL	0, positive number	0	Corner transition parameters
OrientationMode	Interpolation positioning mode	SMC_ORIENTATION_MODE	GreatCircle/Axis	-	GreatCircle: Move along the shortest path from the starting position to the target position. In this mode, even if the starting and ending positions are within the specified area, the implemented path may still leave this area. Axis: The positioning axis moves

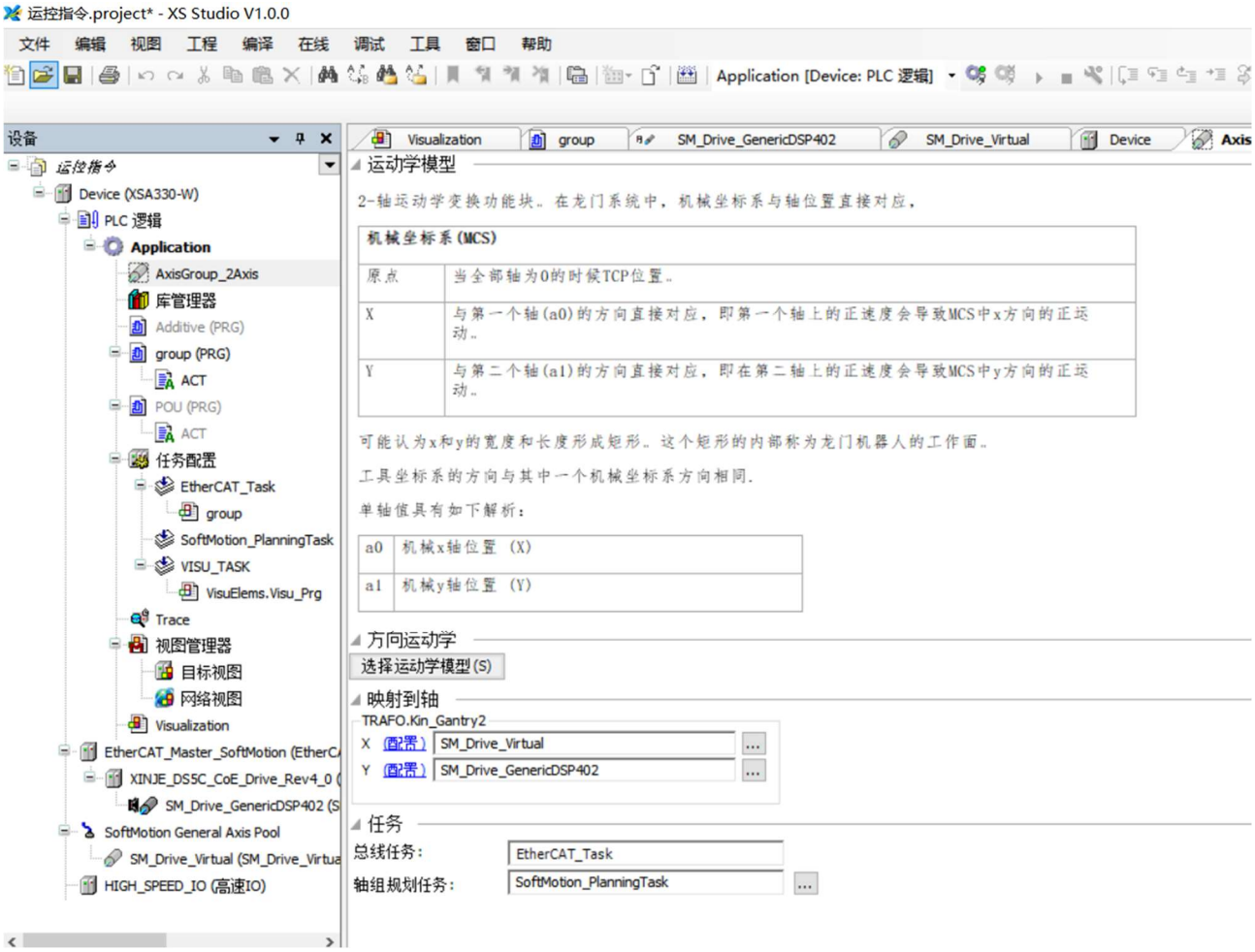
					within the specified area from the start position to the end position, and not all kinematic transformations support this mode
VelFactor	speed factor	LREAL	0-1	1	Speed factor, the maximum speed of each axis multiplied by this speed factor, with a value between [0, 1]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk factor	LREAL	0-1	1	Jerk factor, the maximum velocity of each axis multiplied by this jerk factor, with a value between [0,1]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
CommandAccepted	Motion reception	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	Valid when CommandAccepted or Done is TRUE

(3) Function description

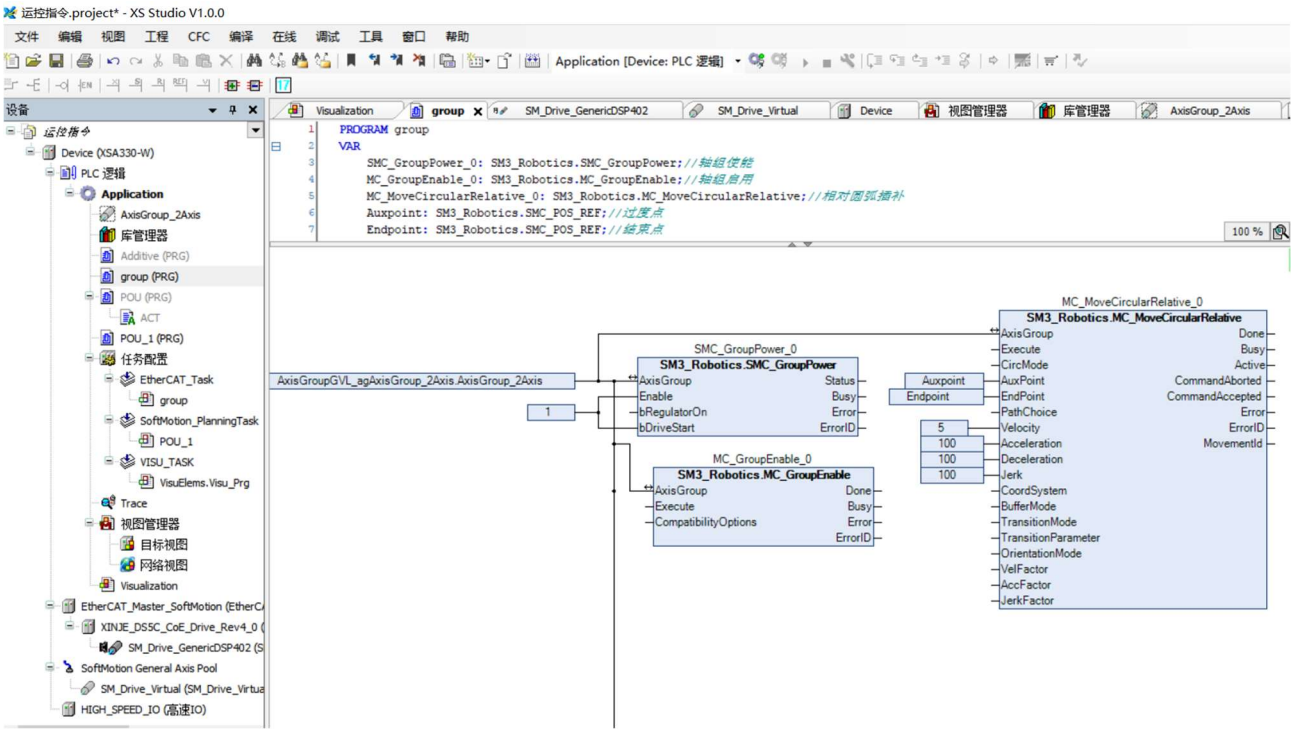
- The rising edge of the Execute triggers the command movement, while the falling edge has no effect on the command movement.
- When using the center arc mode, input the values of parameters AuxPoint [1] and AuxPoint [2] as the distance between the center of the circle and the starting point of the arc. When using the radius arc mode, AuxPoint [1] represents the radius value, and AuxPoint [2] is invalid.
- The numerical values of EndPoint [1]~EndPoint [8] represent the endpoint coordinates of each axis.

(4) Application

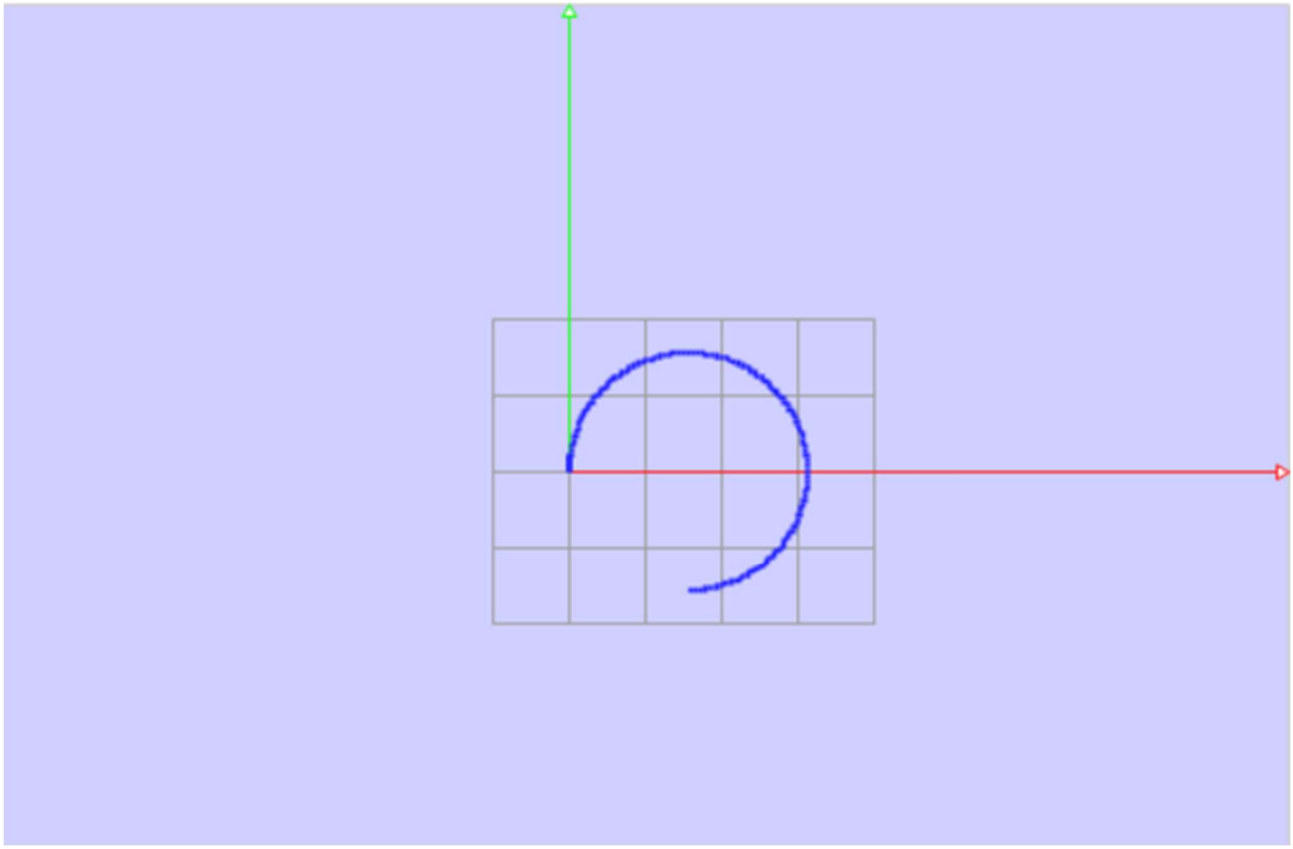
Example 1: Select the kinematic model of the two axes of the gantry, select the imaginary axis for the X-axis, and select the Ethercat real axis for the Y-axis to perform a relative arc interpolation.



Programming: Set the current positions of both axes to 0, select the arc mode as three points to determine an arc, set the auxiliary point AuxPoint to (50, 50), set the EndPoint to (50, -50), set the speed of both axes to 5, and draw an arc relative to the current point (0, 0).



In the view, the arc trajectory can be seen.

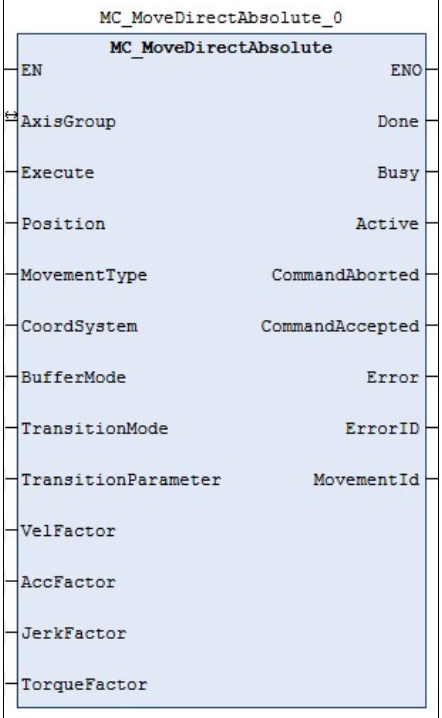


3-2-2-18. Absolute position quick positioning [MC_MoveDirectAbsolute]

(1) Instruction overview

Control all axes within the axis group to run at the specified speed to the absolute position endpoint.

Absolute position quick positioning [MC_MoveDirectAbsolute]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_MoveDirectAbsolute	Absolute position quick positioning		MC_MoveDirectAbsolute_0(AxisGroup:= , Execute:= , Position:= , MovementType:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , VelFactor:= , AccFactor:= , JerkFactor:= , TorqueFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
Position	position	SMC_POS_REF	-	-	Absolute target position in the specified reference coordinate system
MovementType	PTP motion mode	SMC_PTP_MOVEMENT_TYPE	-	-	Fast (0): PTP motion mode with time priority Path_ Invariant: PTP motion with fixed path
CoordSystem	reference coordinate	SMC_COORD_SYSTEM	SMC_COORD_SYSTEM		reference coordinate system

	system				
BufferMode	buffer mode	MC_BUFFER_MODE	-	0	specify the multiple start motion commands action
TransitionMode	Corner transition mode	MC_TRANSITION_MODE	TMNone/ TMStartVelocity /TMCornerDistance	-	TMNone: no mixing TMStartVelocity: mixed based on speed TMCornerDistance: mixed based on distance
Transition Parameter	Corner transition parameters	array [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL	0, positive number	0	Corner transition parameters
VelFactor	speed factor	LREAL	0-1	1	Speed factor, the maximum speed of each axis multiplied by this speed factor, with a value between [0, 1]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk factor	LREAL	0-1	1	Jerk factor, the maximum velocity of each axis multiplied by this jerk factor, with a value between [0,1]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
CommandAccepted	Motion reception	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	True when the motion is being executed or completed

(3) Function description

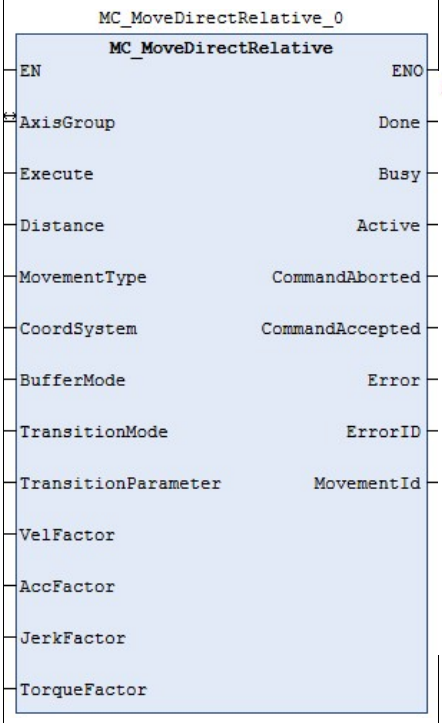
- In motion, each axis is a relatively independent absolute position motion, and its motion trajectory is uncertain.

3-2-2-19. Relative position quick positioning [MC_MoveDirectRelative]

(1) Instruction overview

Control all axes within the axis group to run at the specified speed to the end point of the relative position.

Relative position quick positioning [MC_MoveDirectRelative]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_MoveDirectRelative	relative position quick positioning		MC_MoveDirectRelative_0(AxisGroup:= , Execute:= , Distance:= , MovementType:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , VelFactor:= , AccFactor:= , JerkFactor:= , TorqueFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
Distance	relative position	SMC_POS_REF	-	-	Relative target position in the specified reference coordinate system
MovementType	PTP motion mode	SMC_PTP_MOVEMENT_TYPE	-	-	Fast (0): PTP motion mode with time priority Path_ Invariant: PTP motion with fixed path
CoordSystem	reference coordinate	SMC_COORD_SYSTEM	SMC_COORD_SYSTEM		reference coordinate system

	system				
BufferMode	buffer mode	MC_BUFFER_MODE	-	0	specify the multiple start motion commands action
TransitionMode	Corner transition mode	MC_TRANSITION_MODE	TMNone/TMStartVelocity/TMCornerDistance		TMNone: no mixing TMStartVelocity: mixed based on speed TMCornerDistance: mixed based on distance
Transition Parameter	Corner transition parameters	array [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL	0, positive number	0	Corner transition parameters
VelFactor	speed factor	LREAL	0-1	1	Speed factor, the maximum speed of each axis multiplied by this speed factor, with a value between [0, 1]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
JerkFactor	jerk factor	LREAL	0-1	1	Jerk factor, the maximum velocity of each axis multiplied by this jerk factor, with a value between [0,1]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
Command Aborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
Command Accepted	Motion reception	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	True when the motion is being executed or completed

(3) Function description

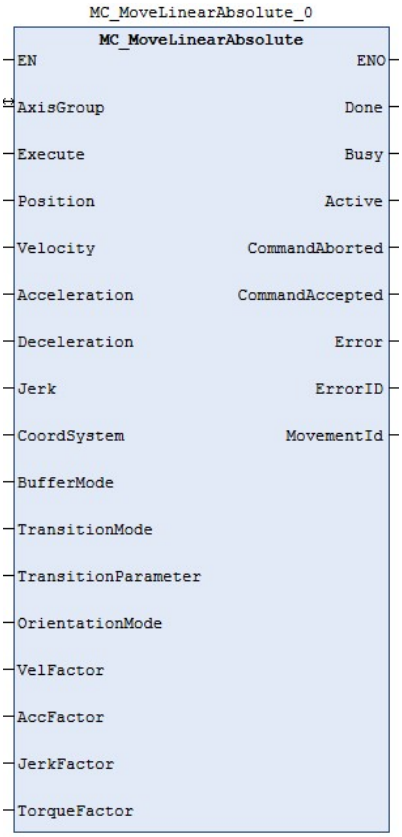
- In motion, each axis is a relatively independent absolute position motion, and its motion trajectory is uncertain.

3-2-2-20. Absolute position linear interpolation [MC_MoveLinearAbsolute]

(1) Instruction overview

The linear interpolation motion of the absolute position mode of the control axis group in the specified coordinate system.

Absolute position linear interpolation [MC_MoveLinearAbsolute]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_MoveLinearAbsolute	absolute position linear interpolation		MC_MoveLinearAbsolute_0(AxisGroup:= , Execute:= , Position:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , OrientationMode:= , VelFactor:= , AccFactor:= , JerkFactor:= , TorqueFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE /FALSE	FALSE	Execute current instruction
Position	position	SMC_POS_REF	-	-	Absolute target position in the specified reference coordinate system
Velocity	speed	LREAL	0, positive number	0	Max resultant speed [command unit/s]
Acceleration	acceleration	LREAL	positive number	0	Max resultant acceleration [command

					unit /s^2]
Deceleration	deceleration	LREAL	positive number	0	Max resultant deceleration [command unit /s^2]
Jerk	jerk speed	LREAL	positive number	0	Max resultant jerk speed [command unit /s^3]
CoordSystem	reference coordinate system	SMC_COORD_SYSTEM	SMC_COORD_SYSTEM	-	reference coordinate system
BufferMode	buffer mode	MC_BUFFER_MODE	-	0	specify the multiple start motion commands action
TransitionMode	Corner transition mode	MC_TRANSITION_MODE	TMNone/TMStartVelocity/TMCornerDistance	-	TMNone: no mixing TMStartVelocity: mixed based on speed TMCornerDistance: mixed based on distance
TransitionParameter	Corner transition parameters	array [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL	0, positive number	0	corner transition parameters

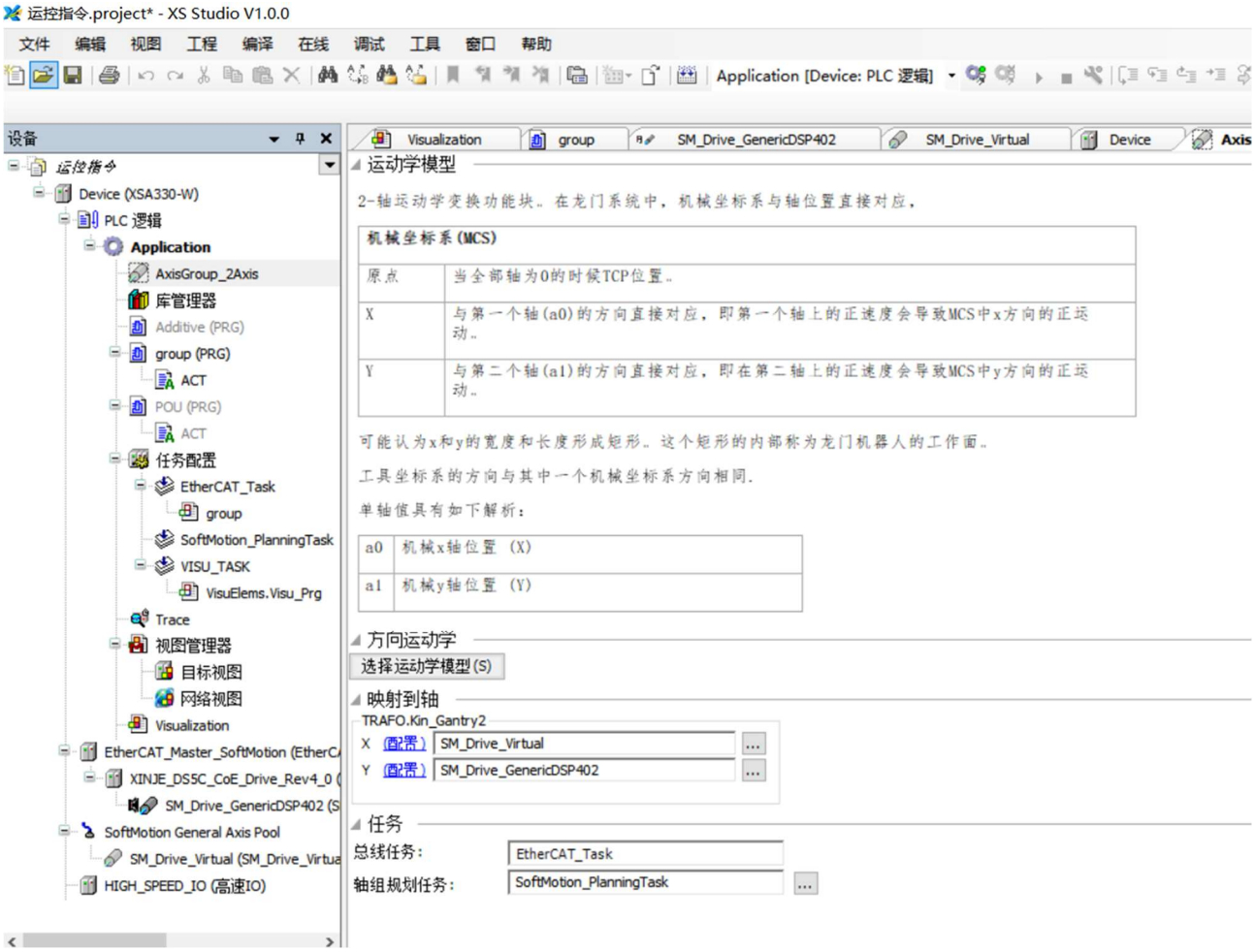
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
OrientationMode	interpolation positioning mode	SMC_ORIENTATION_MODE	GreatCircle/ Axis	-	GreatCircle: Move along the shortest path from the starting position to the target position. In this mode, even if the starting and ending positions are within the specified area, the implemented path may still leave this area. Axis: The positioning axis moves within the specified area from the start position to the end position, and not all kinematic transformations support this mode
VelFactor	speed factor	LREAL	0-1	1	Speed factor, the maximum speed of each axis multiplied by this speed factor, with a value between [0, 1]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk factor	LREAL	0-1	1	Jerk factor, the maximum velocity of each axis multiplied by this jerk factor, with a value between [0,1]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed

Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
CommandAccepted	Motion reception	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	True when the motion is being executed or completed

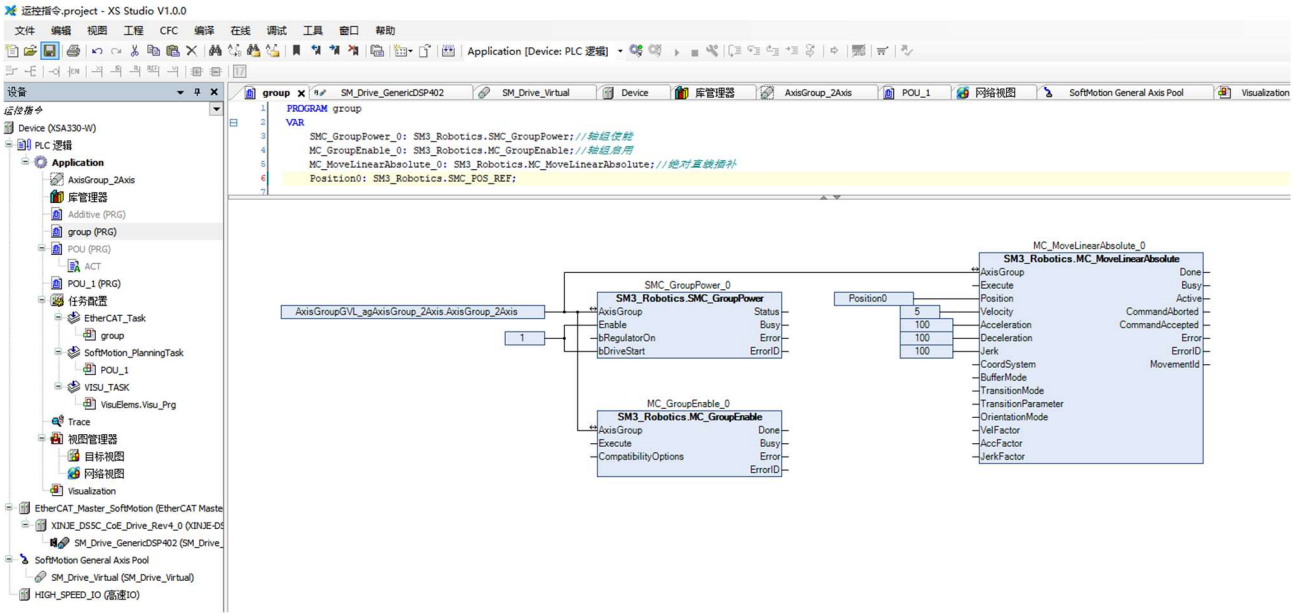
(3) Function description

- Control the linear interpolation motion of the axis group in the absolute position mode under the specified coordinate system.
- The relationship between the parameters BufferMode, TransitionMode, and TransitionParameter is explained as follows:
  (1) When BufferMode selects mcBuffered mode, TransitionMode only supports mcTMNone mode;
  (2) When BufferMode selects mcBlendingPrevious mode, TransitionMode can be selected.
- McTMConstantVelocity and mcTMCornerDistance modes.

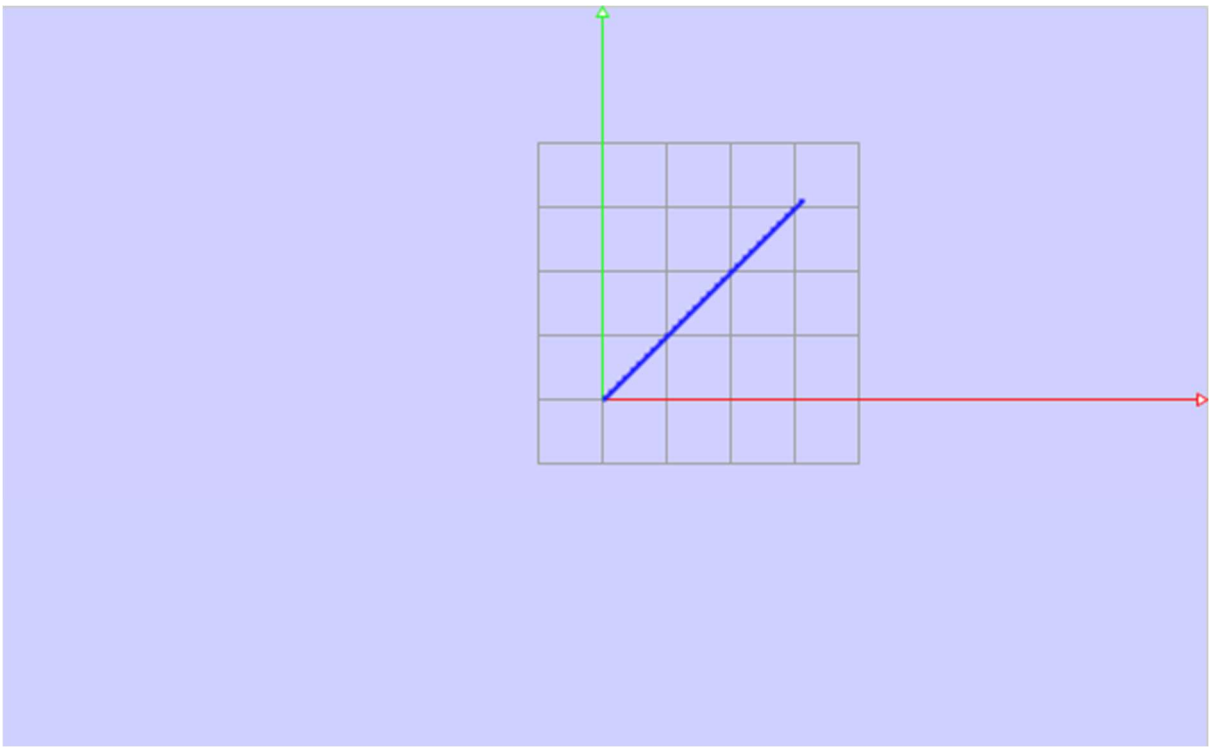
Example 1: Select the kinematic model of the two axes of the gantry, select the imaginary axis for the X-axis, and select the Ethercat real axis for the Y-axis to perform an absolute position interpolation.



Programming: Set the current positions of both axes to 0, set the position to (100, 100), and perform absolute linear interpolation motion starting from the starting point (0,0).



In the view, the straight line trajectory can be seen.

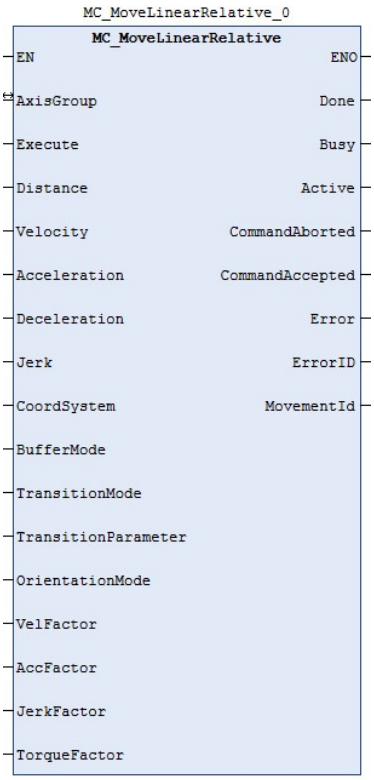


3-2-2-21. Relative position linear interpolation [MC_MoveLinearRelative]

(1) Instruction overview

The linear interpolation motion of the relative position mode of the control axis group in the specified coordinate system.

Relative position linear interpolation [MC_MoveLinearRelative]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_MoveLinearRelative	relative position linear interpolation		MC_MoveLinearRelative_0(AxisGroup:= , Execute:= , Distance:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , OrientationMode:= , VelFactor:= , AccFactor:= , JerkFactor:= , TorqueFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
Distance	relative position	SMC_POS_REF	-	-	Relative target position in the specified reference coordinate system
Velocity	speed	LREAL	0, positive number	0	Max resultant speed [command unit/s]
Acceleration	acceleration	LREAL	positive number	0	Max resultant acceleration [command unit /s^2]
Deceleration	deceleration	LREAL	positive	0	Max resultant deceleration [command

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
			number		unit /s^2]
Jerk	jerk speed	LREAL	positive number	0	Max resultant jerk speed [command unit /s^3]
CoordSystem	reference coordinate system	SMC_COORD_SYSTEM	SMC_COORD_SYSTEM	-	reference coordinate system
BufferMode	buffer mode	MC_BUFFER_MODE	-	0	specify the multiple start motion commands action
TransitionMode	Corner transition mode	MC_TRANSITION_MODE	TMNone/TMStartVelocity/TMCornerDistance	-	TMNone: no mixing TMStartVelocity: mixed based on speed TMCornerDistance: mixed based on distance

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
TransitionParameter	Corner transition parameters	array [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL	0, positive number	0	Corner transition parameters
OrientationMode	Interpolation positioning mode	SMC_ORIENTATION_MODE	GreatCircle /Axis	-	GreatCircle: Move along the shortest path from the starting position to the target position. In this mode, even if the starting and ending positions are within the specified area, the implemented path may still leave this area. Axis: The positioning axis moves within the specified area from the start position to the end position, and not all kinematic transformations support this mode
VelFactor	speed factor	LREAL	0-1	1	Speed factor, the maximum speed of each axis multiplied by this speed factor, with a value between [0, 1]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk factor	LREAL	0-1	1	Jerk factor, the maximum velocity of each axis multiplied by this jerk factor, with a value between [0,1]

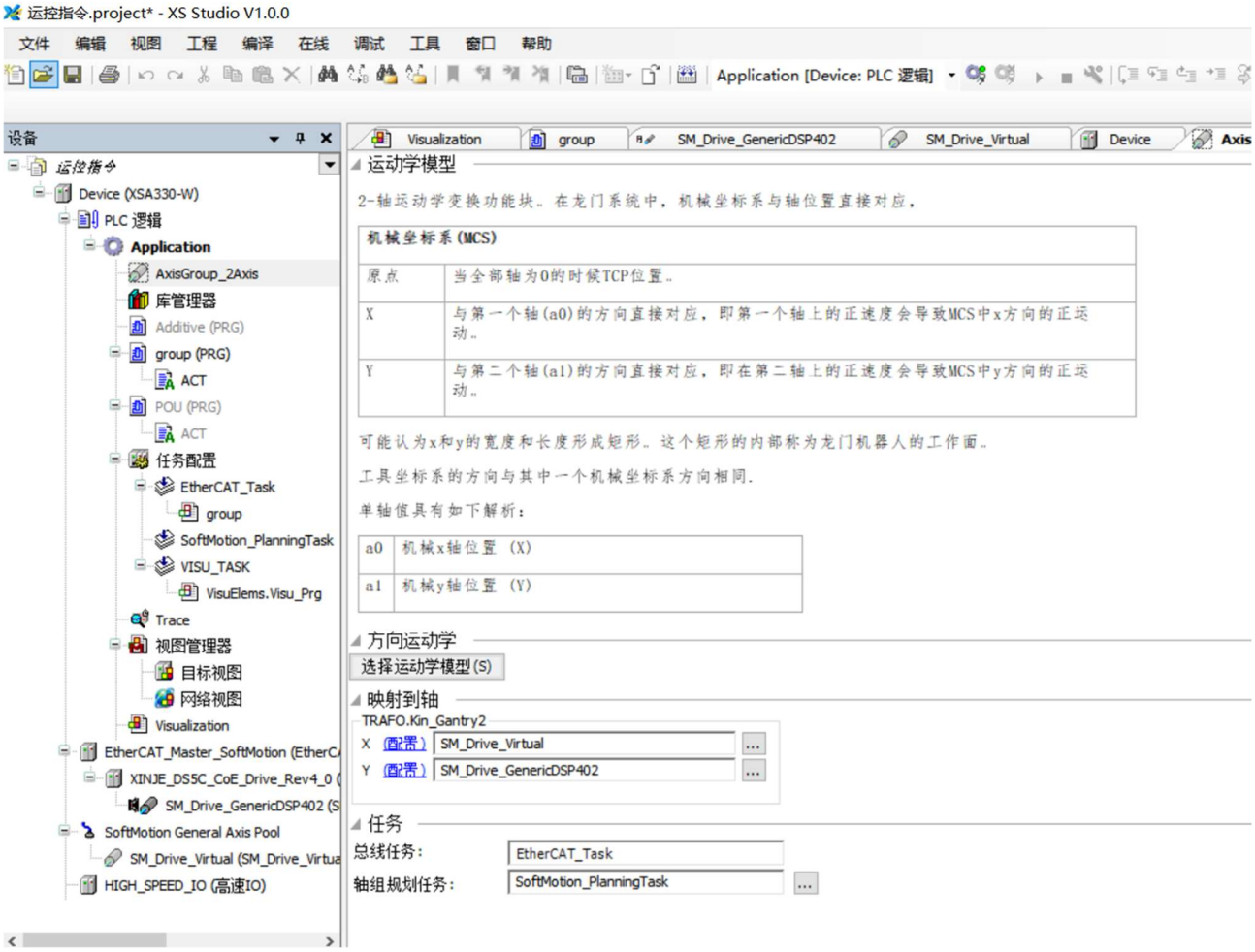
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
CommandAccepted	Motion reception	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	True when the motion is being executed or completed

(3) Function description

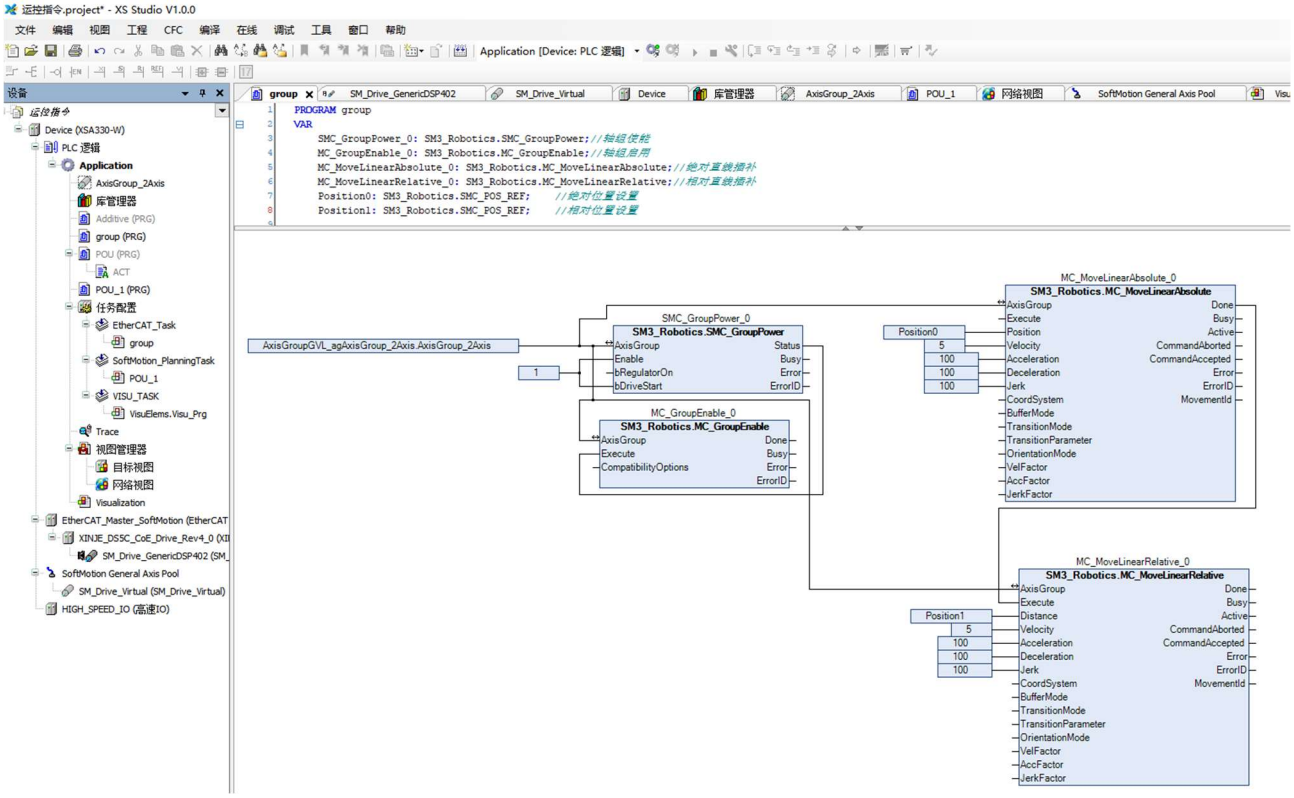
- The parameter TransitionParameter is only valid when the TransitionMode is mcTMCornerDistance.

(4) Application

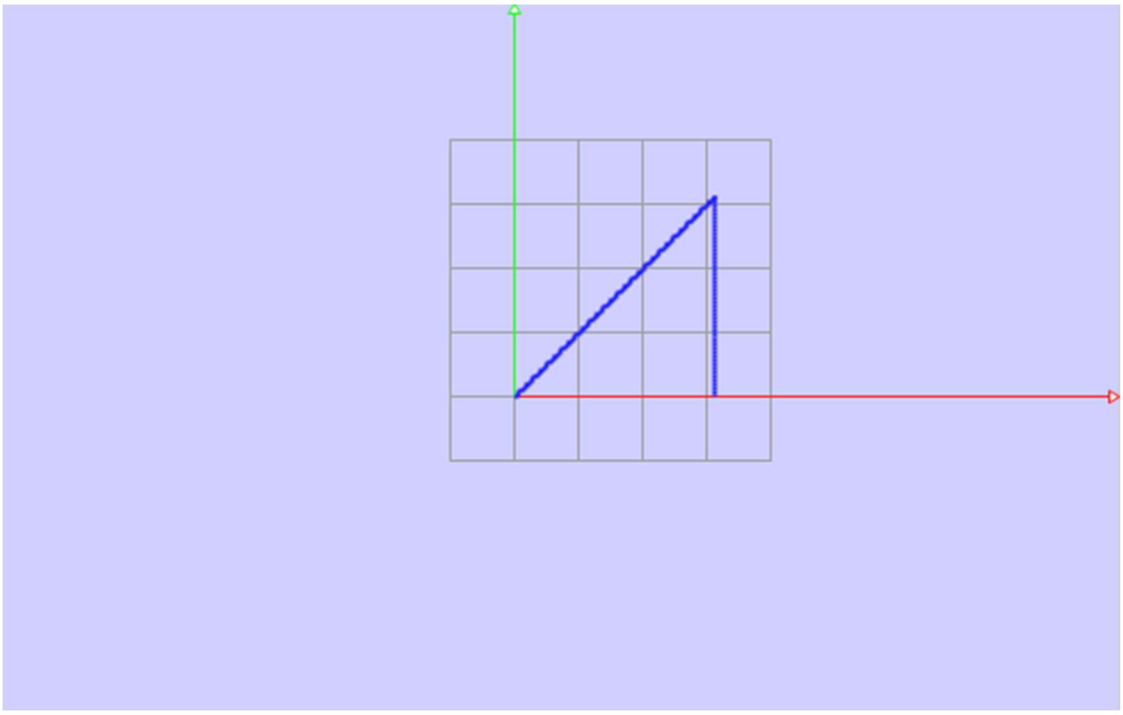
Example 1: Select the kinematic model of the two axes of the gantry, select the imaginary axis for the X-axis, and select the Ethercat real axis for the Y-axis to perform a relative position interpolation.



Programming: Set the current positions of both axes to 0, that is, the starting position is (0, 0), the absolute linear interpolation position is set to (100, 100), and then the relative linear interpolation position is set to (0, -100).



In the view, the straight line trajectory can be seen.

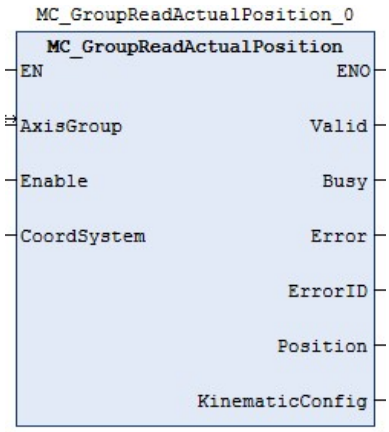


3-2-2-22. Read the feedback position of the axis group [MC_GroupReadActualPosition]

(1) Instruction overview

Read the feedback position of the axis group in the specified coordinate system.

Read the feedback position of the axis group [MC_GroupReadActualPosition]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupReadActualPosition	read axis group feedback position		MC_GroupReadActualPosition_0(AxisGroup:= , Enable:= , CoordSystem:= , Valid=> , Busy=> , Error=> , ErrorID=> , Position=> , KinematicConfig=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
CoordSystem	Apply coordinate system	SMC_COORD_SYSTEM	-	-	Apply coordinate system
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	error recognization
Position	position	SMC_POS_REF	-	-	axis group actual position
KinematicConfig	kinematic	TRAFO.CONFIGDATA	-	-	The kinematic configuration

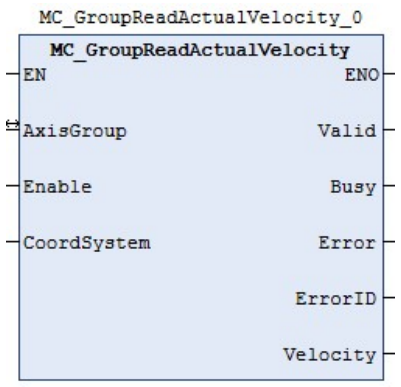
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
	configuration				of the current position. Only set when the coordinate system is Cartesian (i.e. not set in ACS)

3-2-2-23. Read the feedback speed of the axis group [MC_GroupReadActualVelocity]

(1) Instruction overview

Read the feedback speed of the axis group in the specified coordinate system.

Read the feedback speed of the axis group [MC_GroupReadActualVelocity]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupReadActualVelocity	read axis group feedback speed		MC_GroupReadActualVelocity_0(AxisGroup:= , Enable:= , CoordSystem:= , Valid=> , Busy=> , Error=> , ErrorID=> , Velocity=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
CoordSystem	Apply coordinate system	SMC_COORD_SYSTEM	-	-	Apply coordinate system
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the

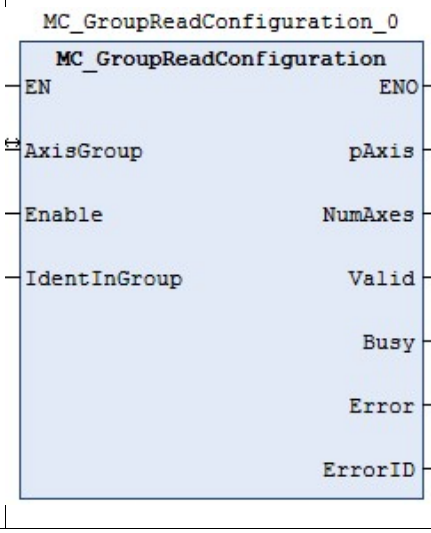
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
					function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	error recognization
Velocity	speed	SMC_POS_REF	-	-	The current feedback speed of the axis group. If the Cartesian coordinate system is selected; Velocity. c contains Cartesian velocities: (X, Y, Z) is the velocity vector, and (A, B, C) is the angular velocity around the x, y, and z axes, respectively

3-2-2-24. Read axis group configuration parameters [MC_GroupReadConfiguration]

(1) Instruction overview

Read the configuration parameters such as axis and quantity contained in the axis group.

Read axis group configuration parameters [MC_GroupReadConfiguration]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupReadConfiguration	read axis group configuration parameters		MC_GroupReadConfiguration_0(AxisGroup:= , Enable:= , IdentInGroup:= , pAxis=> , NumAxes=> , Valid=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis	AXIS_GROUP_REF_SM3	-	-	specified axis group

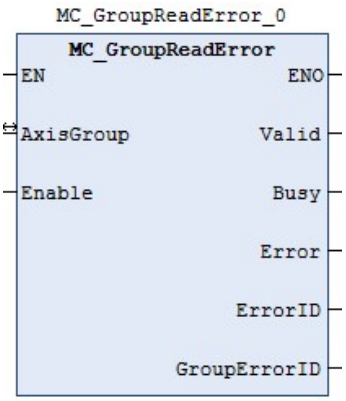
	group				
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
IdentInGroup	axis number	IDENT_IN_GROUP_REF_SM3	0, positive number	0	Enter the number of the corresponding axis in the axis group
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
pAxis	reference axis	POINTER TOAXIS_REF_SM3	-	-	selected reference axis
NumAxes	axis quantity	UDINT	0, positive number	0	axis quantity in the axis group
Valid	obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	error recognization

3-2-2-25. Read axis group error [MC_GroupReadError]

(1) Instruction overview

Obtain error information for the axis group.

Read axis group error [MC_GroupReadError]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupReadError	read axis group error		MC_GroupReadError_0(AxisGroup:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , GroupErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Valid	obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Single axis error code indication
GroupErrorID	error code	SMC_ERROR	-	0	Axis group error code indication

(3) Function description

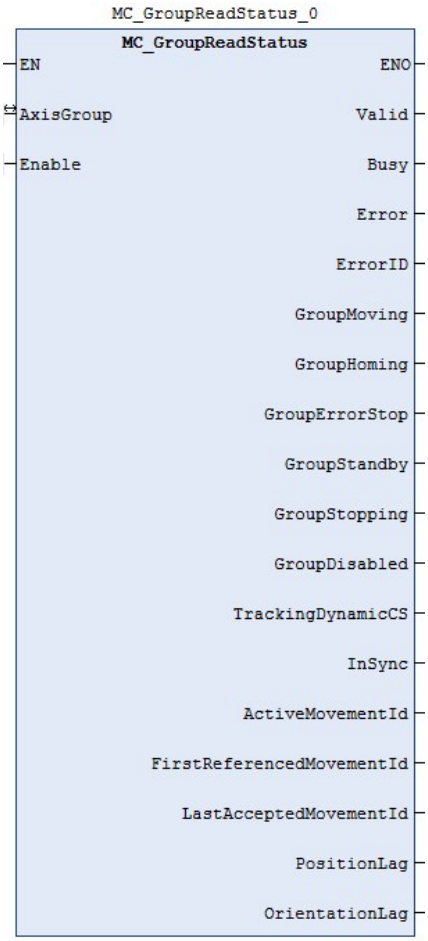
- Obtain error information for single axis and axis group, such as reading hard limit (or soft limit) or single axis error.

3-2-2-26. Read the current operating status of the axis group [MC_GroupReadStatus]

(1) Instruction overview

Used to obtain the current motion state of the axis group.

Read the current operating status of the axis group [MC_GroupReadStatus]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
MC_GroupReadStatus	read axis group current motion status		MC_GroupReadStatus_0(AxisGroup:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , GroupMoving=> , GroupHoming=> , GroupErrorStop=> , GroupStandby=> , GroupStopping=> , GroupDisabled=> , TrackingDynamicCS=> , InSync=> , ActiveMovementId=> , FirstReferencedMovementId=> , LastAcceptedMovementId=> , PositionLag=> , OrientationLag=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	valid	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to activate the processing of the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description

Valid	obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Single axis error code indication
GroupErrorID	error code	SMC_ERROR	-	0	Axis group error code indication
GroupMoving	moving	BOOL	TRUE/FALSE	FALSE	Axis group in motion is TRUE
GroupHoming	homing	BOOL	TRUE/FALSE	FALSE	Axis group homing is TRUE
GroupErrorStop	error stop	BOOL	TRUE/FALSE	FALSE	Axis group error stop is true
GroupStandby	ready to move	BOOL	TRUE/FALSE	FALSE	Axis group motion preparation status is TRUE
GroupStopping	stopping	BOOL	TRUE/FALSE	FALSE	True when the axis group movement stops
GroupDisabled	not enable axis group	BOOL	TRUE/FALSE	FALSE	Axis group invalid status is TRUE
TrackingDynamicCS	Currently in a dynamic coordinate system	BOOL	TRUE/FALSE	FALSE	True when currently using a dynamic coordinate system
InSync	On the path or already in place	BOOL	TRUE/FALSE	FALSE	True in continuous interpolation motion when the obtained position belongs to the specified path; In normal point position and interpolation motion, it is true when the current position is equal to the target position
ActiveMovementId	motion segment number	SMC_Movement_Id	0, positive number	-	The identifier of the activity movement. Id=0 indicates no active movement

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
LastAcceptedMovementId	motion segment number	SMC_Movement_Id	0, positive number	0	The identifier of the last accepted motion. Id=0 indicates that no action has been taken yet

(3) Function description

- Calling this command can continuously obtain information such as the motion status of the axis group, whether it is in place, and the motion segment number.

3-2-2-27. Startup the axis group [SMC_StartupAxisGroup]

(1) Instruction overview

Used for testing and debugging axis groups.

Startup the axis group [SMC_StartupAxisGroup]			
Execution condition	-	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
SMC_StartupAxisGroup	startup the axis group	SMC_StartupAxisGroup_0 SMC_StartupAxisGroup EN ENO AxisGroup	SMC_StartupAxisGroup_0(AxisGroup:=);

(2) Related variables

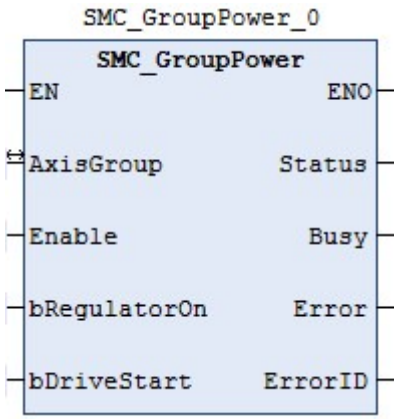
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group

3-2-2-28. Enable the axis group [SMC_GroupPower]

(1) Instruction overview

Enable all axes under the axis group, equivalent to calling MC_Power for all axes under the axis group.

Enable the axis group [SMC_GroupPower]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
SMC_GroupPower	enable the axis group		SMC_GroupPower_0(AxisGroup:= , Enable:= , bRegulatorOn:= , bDriveStart:= , Status=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

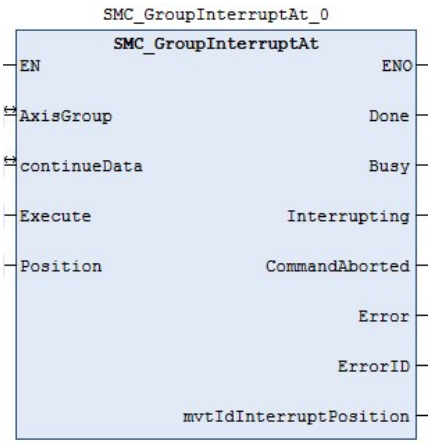
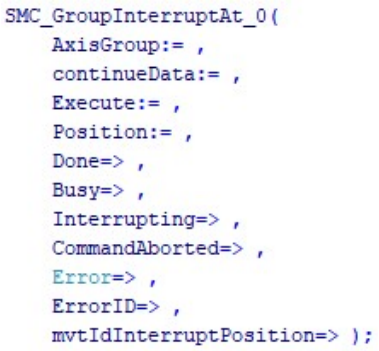
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	valid	BOOL	TRUE/FALSE	FALSE	Enter TRUE to start the module running
bRegulatorOn	enable	BOOL	TRUE/FALSE	FALSE	Set TRUE to enable the axis group
bDriveStart	enable drive	BOOL	TRUE/FALSE	FALSE	Must be set to TRUE to turn off emergency stop processing for the function block
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Status	can run	BOOL	TRUE/FALSE	FALSE	True when the axis group is ready
Valid	obtain flag	BOOL	TRUE/FALSE	FALSE	If the output value is valid, then it is TRUE
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	error code

3-2-2-29. Break Assignment [SMC_GroupInterruptAt]

(1) Instruction overview

Interrupt the currently moving axis group.

Break Assignment [SMC_GroupInterruptAt]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
SMC_GroupInterruptAt	break assignment		


```
SMC_GroupInterruptAt_0(
    AxisGroup:= ,
    continueData:= ,
    Execute:= ,
    Position:= ,
    Done=> ,
    Busy=> ,
    Interrupting=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> ,
    mvtIdInterruptPosition=> );
```

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
continueData	Continuing motion data	SMC_AXIS_GROUP_CONTINUE_DATA	-	-	Motion information when axis group motion is interrupted
VAR_INPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
Position	position	SMC_GroupInterruptPosition	-	-	The location where the ongoing movement will be interrupted
VAR_OUTPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Interrupting	interrupting	BOOL	TRUE/FALSE	FALSE	Signal indicating that the function block is currently being interrupted
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error code

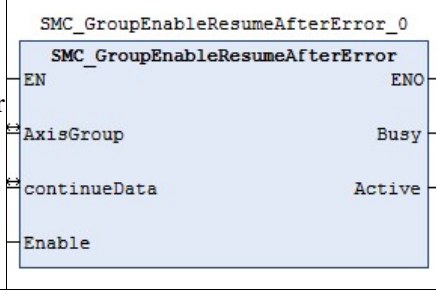
mvtIdInterrupt Position	Interrupt location ID	-	-	-	MovemenID corresponding to the interrupt location

3-2-2-30. Reboot after error reset [SMC_GroupEnableResumeAfterError]

(1) Instruction overview

Restore the axis group state that was interrupted due to an error.

Reboot after error reset [SMC_GroupEnableResumeAfterError]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
SMC_GroupEnableResume AfterError	restart after error reset		SMC_GroupEnableResumeAfterError_0(AxisGroup:= , continueData:= , Enable:= , Busy=> , Active=>);

(2) Related variables

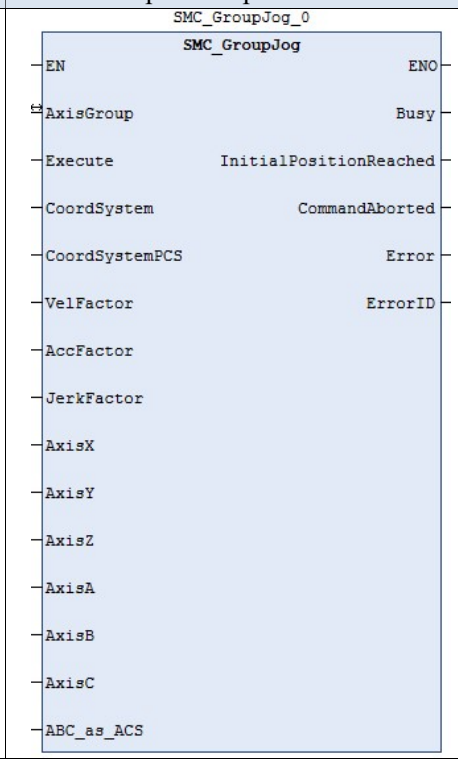
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
continueData	Continuing motion data	SMC_AXIS_ GROUP_CONTINUE_DATA	-	-	Axis group position during motion interruption
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	valid	BOOL	TRUE/FALSE	FALSE	Enable command function
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in call	BOOL	TRUE/FALSE	FALSE	Writing resume data

3-2-2-31. Axis group jog run [SMC_GroupJog]

(1) Instruction overview

Control the axis group to perform Jog motion in the specified coordinate system.

Axis group jog run [SMC_GroupJog]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
SMC_GroupJog	axis group jog run		SMC_GroupJog_0(AxisGroup:= , Execute:= , CoordSystem:= , CoordSystemPCS:= , VelFactor:= , AccFactor:= , JerkFactor:= , AxisX:= , AxisY:= , AxisZ:= , AxisA:= , AxisB:= , AxisC:= , ABC_as_ACS:= , Busy=> , InitialPositionReached=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_REF_SM3	-	-	specified axis group
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
CoordSystem	reference coordinate system	SMC_COORD_SYSTEM	-	-	reference coordinate system
CoordSystemPCS	PCS coordinate system	SMC_COORD_SYSTEM	-	-	The internal coordinate system used for jogging is PCS, and SMC_SetDynCoordTransformEx is used when needed to make changes
VelFactor	speed factor	LREAL	0-1	1	Speed factor, the maximum speed of each axis multiplied by this speed factor, with a

					value between [0, 1]
AccFactor	acceleration factor	LREAL	0-1	1	Acceleration factor, the maximum velocity of each axis multiplied by this acceleration factor, with a value between [0,1]
JerkFactor	jerk factor	LREAL	0-1	1	Jerk factor, the maximum velocity of each axis multiplied by this jerk factor, with a value between [0,1]
AxisX	Axis X	IAxisRef	-	0	The X-axis in the coordinate system is set to 0 if it is not used
AxisY	Axis Y	IAxisRef	-	0	The Y-axis in the coordinate system is set to 0 if it is not used
AxisZ	Axis Z	IAxisRef	-	0	The Z-axis in the coordinate system is set to 0 if it is not used
AxisA	Axis A	IAxisRef	-	0	Control the a-coordinate (rotation around the z-axis) or the axis of the first tool axis. If not applicable, set to 0
AxisB	Axis B	IAxisRef	-	0	Control the b coordinate (rotation around the y axis) or the axis of the second tool axis. If not applicable, set to 0
AxisC	Axis C	IAxisRef	-	0	Control the c-coordinate (rotation around the z axis) or the axis of the third tool axis. If not applicable, set to 0
ABC_as_ACS	Start coordinate conversion	BOOL	TRUE/FALSE	FALSE	If true, the positions of AxisA, AxisB, and AxisC will be interpreted as the target position of the tool's kinematic axis, otherwise represented as the ZYZ direction. If CoordSystem is set to ACS, it will be ignored. If true, the kinematic transformation of the axis group must be Kin_ Coupled type and supports SMC_ORIENTATION_MODE.Axis
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
InitialPosition Reached	Received location	BOOL	TRUE/FALSE	FALSE	-
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	error code

(3) Function description

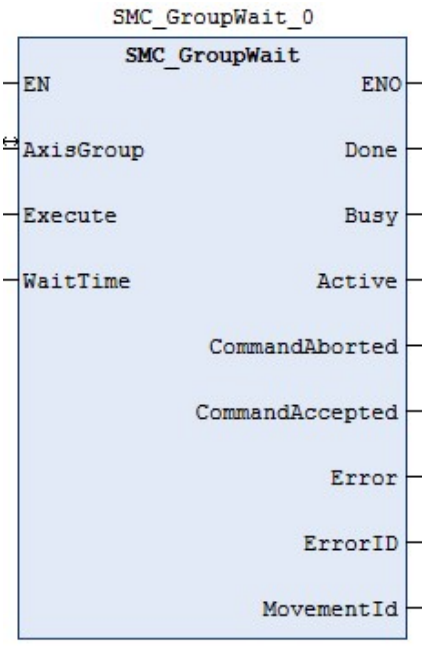
- When the axis group undergoes JOG motion (jog), it will move according to the position and direction provided by the configured axis.
- The command will convert the input coordinate system into the target coordinate system for output, such as JOG motion in a machine coordinate system using MCS.
- Set ABC_as_ACS parameters is using a hybrid transformation of the reference axis, where X/Y/Z is the position of the Cartesian coordinate system and A/B/C is the position of the tool coordinate system.
- When the product coordinate system is used for real-time changes in position, it can be achieved by configuring the input coordinate system CoordSystemPCS, and the real-time changes of the axis group in the coordinate system need to be marked.
- To restart this FB without changing CoordSystemPCS, it is necessary to add another move command, such as MC_GroupHalt. If this usage principle is not followed, the error SMC_ AXIS_ GROUP_ PCS_ STILL_ IN_USE will be returned.

3-2-2-32. Axis group wait [SMC_GroupWait]

(1) Instruction overview

Set the delay waiting for the axis group.

Axis group wait [SMC_GroupWait]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Robotics		

Instruction	Name	Graphical representation	ST language
SMC_GroupJog	axis group wait		SMC_GroupWait_0(AxisGroup:= , Execute:= , WaitTime:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
AxisGroup	axis group	AXIS_GROUP_ REF_SM3	-	-	specified axis group

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Execute current instruction
WaitTime	wait time	LREAL	0, positive number	0	The time to wait on the path, in seconds
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
CommandAccepted	motion accepted	BOOL	TRUE/FALSE	FALSE	True when the module successfully calls the axis group
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	error code
MovementId	motion flag	SMC_Movement_Id	TRUE/FALSE	FALSE	True when the motion is being executed or completed

(3) Function description

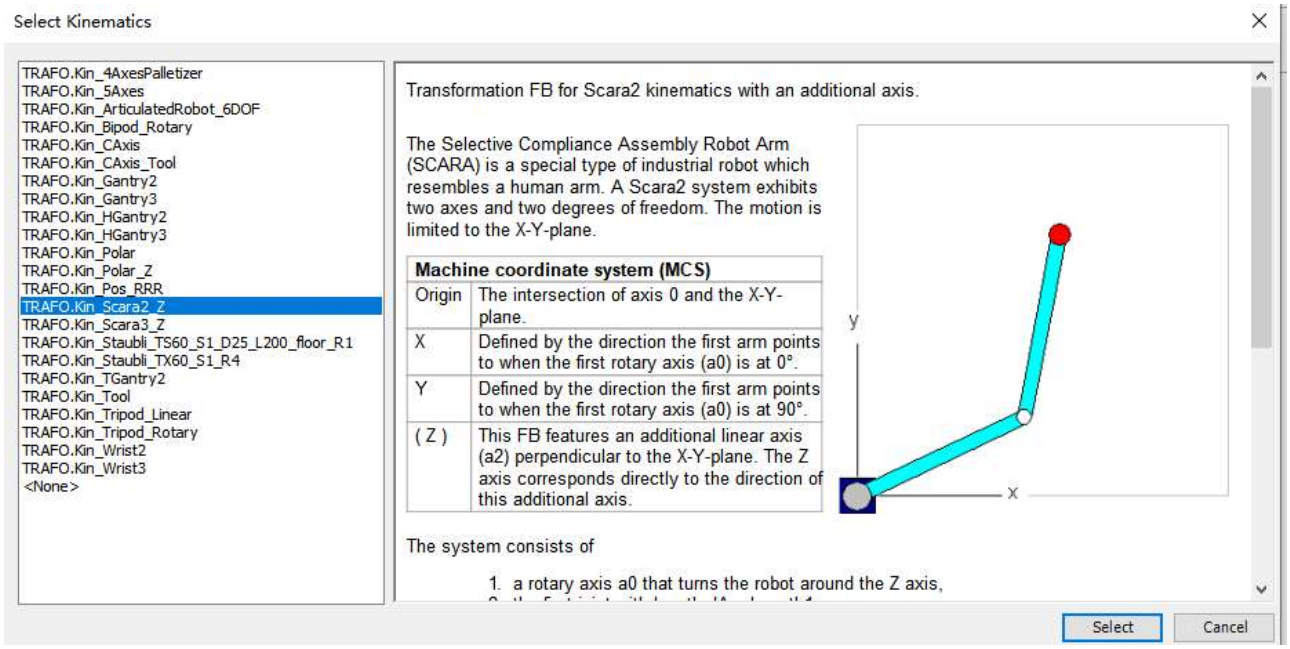
- If the waiting time between two adjacent instructions and the time configured for the task are the same, the actual waiting time between two movements through SMC_ GroupWait instruction has become fewer, for example, the execution of the next instruction usually starts from the next cycle of the task cycle, but if there is a SMC_GroupWait in the middle, then After the delay of the SMC_GroupWait instruction ends, it will immediately start the next motion instruction without waiting for the start of the next cycle of the task cycle.
- If there is a waiting command after tracking the movement, the axis group will track the endpoint of the previous movement within the specified time.
- If each axis is not in a Standstill state but is not controlled by the axis group and waits to be called, the axis group will report an error SMC_AXIS_GROUP_IDLE_WAIT_AXES_MOVING.
- Unlike updating every cycle of the task cycle, If the time of the SMC_ GroupWait instruction is different from a multiple of the bus cycle time, it can also immediately follow the next instruction after the waiting time, making subsequent movements smoother to start.
- Due to technical reasons, the waiting time can be increased by up to one cycle in the following situations: - non-tracking -> waiting -> tracking - tracking -> waiting -> non-tracking - tracking -> waiting -> PTP-tracking.

3-2-3. Axis group function application

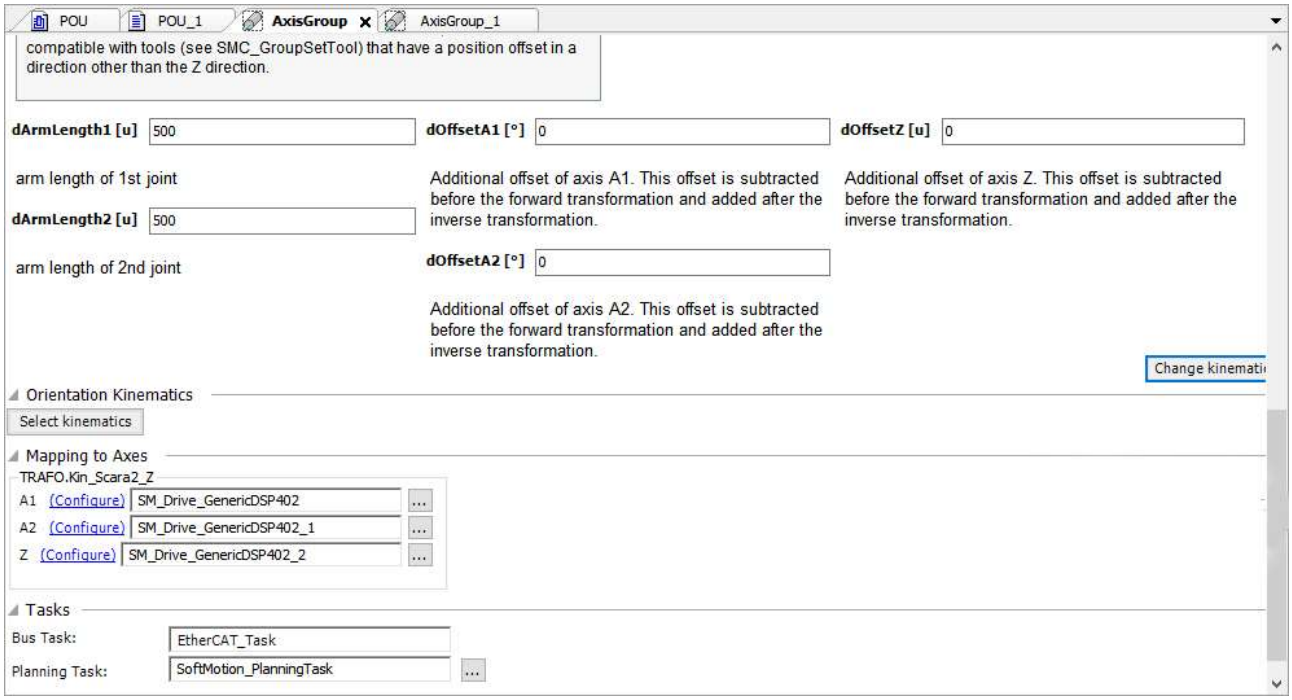
Example: This routine introduces the implementation of motion control for a SCARA system with two nodes.

Program operation:

(1) Right click on "Application" in the engineering equipment bar, select "Add Object" - "AxisGroup", name it AxisGroup, and open it. Select the SCARA kinematic model through the "Kinematic Model" interface.

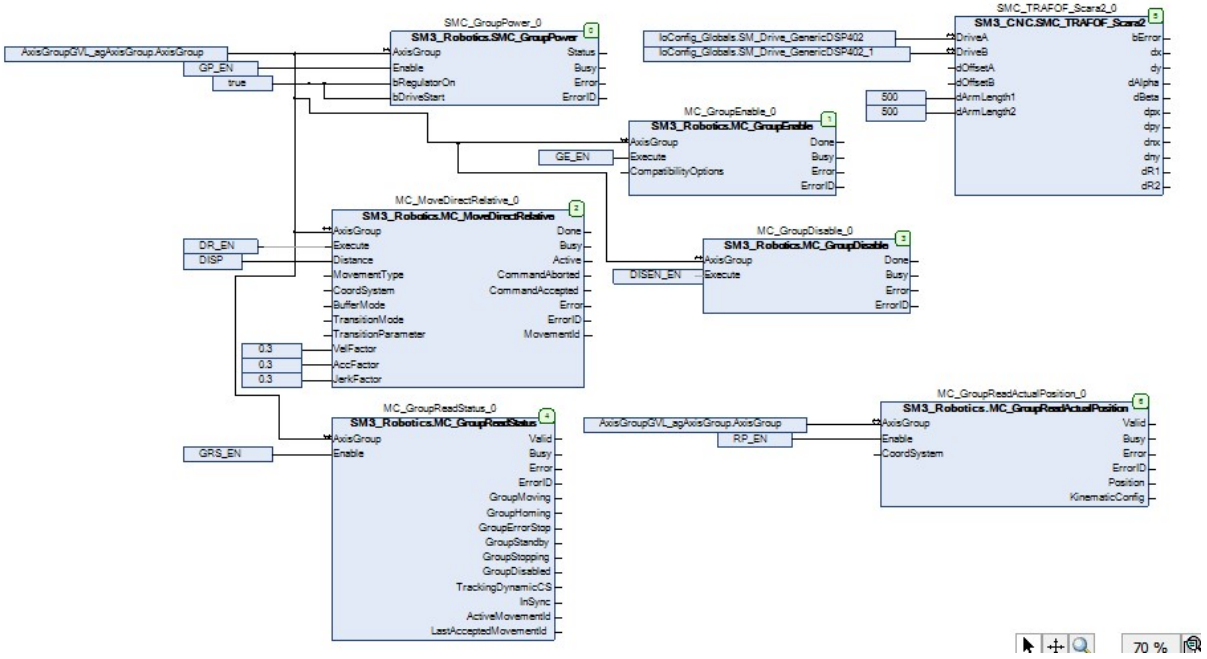


(2) Configure kinematic parameters: Set the length of the large and small arms to 500, and have zero angle offset. Axis parameter configuration, at mapping axis positions.



(3) Write kinematic instructions and configure motion parameters based on motion trajectory

```
PROGRAM POU
VAR
    SMC_GroupPower_0: SM3_Robotics.SMC_GroupPower;
    GP_EN: BOOL;//伺服使能 servo enable
    GE_EN: BOOL;//轴组启动 axis group startup
    MC_MoveDirectRelative_0: SM3_Robotics.MC_MoveDirectRelative;
    DR_EN: BOOL;//移动到相对位置 move to relative position
    DISP: SM3_Robotics.SMC_POS_REF;//相对位置设置 set relative position
    DIST: SM3_Robotics.SMC_POS_REF;
    MC_GroupEnable_0: SM3_Robotics.MC_GroupEnable;
    MC_GroupReadStatus_0: SM3_Robotics.MC_GroupReadStatus;
    GRS_EN: BOOL;//读取轴组状态 read axis group status
    Kin_HGantry2_0: TRAFO.Kin_HGantry2;
    SMC_TRAFOF_Gantry2_0: SM3_CNC.SMC_TRAFOF_Gantry2;
    SMC_TRAFOF_Scara2_0: SM3_CNC.SMC_TRAFOF_Scara2;
    MC_GroupReadActualPosition_0: SM3_Robotics.MC_GroupReadActualPosition;
    RP_EN: BOOL;//读取实时位置 read real-time position
    MC_GroupDisable_0: SM3_Robotics.MC_GroupDisable;
    DISEN_EN: BOOL;//轴组失能 axis group disable
END VAR
```



3-3. CAM function

3-3-1. CAM instruction list

Instruction	Description
MC_CamTableSelect	Connect the selected CAM table to the actual cam table
MC_CamIn	Cam binding
MC_CamOut	Cam unbinding
MC_GearIn	Set the gear ratio of the main and slave axis and activate the electronic gears
MC_GearInPos	Set the synchronization distance and gear ratio of the master and slave axis and activate the electronic gear
MC_GearOut	Disconnect the main and slave axis electronic gears
MC_Phasing	Phase offset
SMC_CAMBounds	Spindle coupling slave axis
SMC_CAMBounds_Pos	Slave axis coupling spindle
SMC_CamEditor	Display cam table in visualization
SMC_CamRegister	Only read tappet information
SMC_GetCamSlaveSetPosition	Obtain cam table slave axis position
SMC_GetTappetValue	Evaluate output tappets
SMC_ReadCAM	Read the cam table
SMC_WriteCAM	Write the cam table

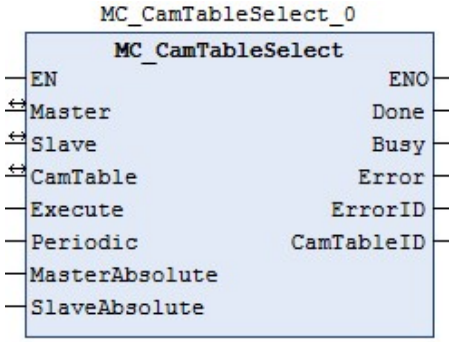
3-3-2. CAM instructions

3-3-2-1. Cam table designation [MC_CamTableSelect]

(1) Instruction overview

Used to select the cam table to be executed, which needs to be used together with MC_CamIn command.

Cam table designation [MC_CamTableSelect]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
MC_CamTableSelect	cam table designation		MC_CamTableSelect_0(Master:= , Slave:= , CamTable:= , Execute:= , Periodic:= , MasterAbsolute:= , SlaveAbsolute:= , Done=> , Busy=> , Error=> , ErrorID=> , CamTableID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Master	Master axis	AXIS_REF	-	-	map to the master axis
Slave	slave axis	AXIS_REF	-	-	map to the slave axis
CamTable	cam table	MC_CAM_REF	-	-	map to cam table description
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
Periodic	repetitive mode	BOOL	TRUE/FALSE	FALSE	TRUE: Execute the specified cam table periodically and repeatedly FALSE: Execute the cam table only once
MasterAbsolute	master axis absolute mode	BOOL	TRUE/FALSE	FALSE	TRUE: represents absolute coordinates FALSE: represents relative coordinates
SlaveAbsolute	slave axis absolute mode	BOOL	TRUE/FALSE	FALSE	TRUE: represents absolute coordinates FALSE: represents relative coordinates
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended

Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
CamTableID	effective CAMID	MC_CAM_ID	-	-	Select effective Cam_ID, used together with CamTableID in MC_CamIn

(3) Function description

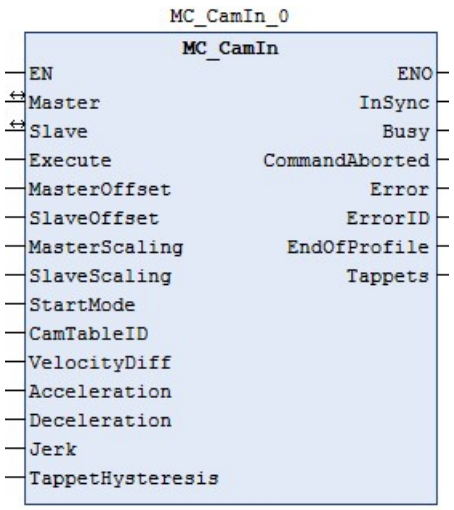
- This instruction is implemented by the "SM3_Basic" library.
- This command is used to specify the cam table required for electronic cam operation, so before using this command, the cam table must be edited (by the cam editor or online editing).
- Execute the specified cam table at the rising edge of Execute, or refresh the specified cam table after updating the cam table.
- When the Done signal output is TRUE, the output variable "CamTableID" is generated and takes effect.
- The main and slave axis cannot be specified as the same axis, otherwise there will be an error output.

3-3-2-2. CAM binding [MC_CamIn]

(1) Instruction overview

Master slave axis binding, can set the engagement mode and related speed of the slave axis.

CAM binding [MC_CamIn]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
MC_CamIn	Cam action start		MC_CamIn_0(Master:= , Slave:= , Execute:= , MasterOffset:= , SlaveOffset:= , MasterScaling:= , SlaveScaling:= , StartMode:= , CamTableID:= , VelocityDiff:= , Acceleration:= , Deceleration:= , Jerk:= , TappetHysteresis:= , InSync=> , Busy=> , CommandAborted=> , Error=> , ErrorID=> , EndOfProfile=> , Tappets=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Master	master axis	AXIS_REF	-	-	map to master axis
Slave	slave axis	AXIS_REF	-	-	map to slave axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
MasterOffset	master axis offset	LREAL	-	0	master table offset
SlaveOffset	slave axis offset	LREAL	-	0	slave table offset
MasterScaling	master axis scale	LREAL	-	1	Scale factor of the main configuration file
SlaveScaling	slave axis scale	LREAL	-	1	Scale factor for slave configuration files
StartMode	engagement method of slave axis	MC_StartMode	0-4	absolute	0: absolute position 1: relative position 2: ramp_in (ramp in) 3: ramp_in_pos (positive ramp in) 4: ramp_in_neg (negative ramp im)
CamTableID	cam table	MC_CAM_ID	-	-	Define the using of cam table, output point of MC_CamTableSelect, used together with CamTableID
VelocityDiff	speed	LREAL	Negative number, 0, positive number	0	Maximum velocity of superimposed motion [u/s]
Acceleration	target acceleration	LREAL	Negative number, 0, positive number	0	acceleration [u/s^2]
Deceleration	target deceleration	LREAL	Negative number, 0, positive number	0	deceleration [u/s^2]
Jerk	target jerk speed	LREAL	Negative number, 0, positive number	0	jerk speed [u/s^3]
TappetHysteresis	tappet hysteresis	LREAL	-	-	tappet hysteresis

VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
InSync	cam take effect	BOOL	TRUE/FALSE	FALSE	TRUE indicates that the slave axis is synchronized with the spindle based on the cam table
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition
EndOfProfile	curve completed	BOOL	TRUE/FALSE	FALSE	Pulse output: The cycle of cam contour ends
Tappets	tappet table	SMC_TappetData	-	-	for tappet signal processed by SMC_GetTappetValue

(3) Function description

- This instruction is implemented by the "SM3_Basic" library.
- This command can perform synchronous cam action as the cam table for the phase (spindle) and displacement (slave axis).
- There are two production methods for the cam table specified by this instruction:

(1) Compile using a cam editor;

(2) Build a cam table data structure through programming.

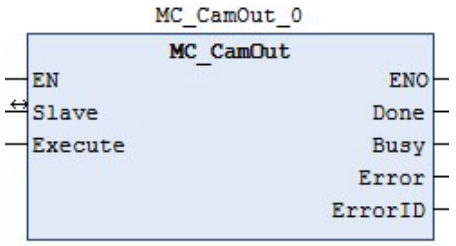
- In a cam system, to call a cam curve, first call MC_CamTableSelect command to select the corresponding cam table and then executes MC_CamIn. If you want to replace the cam curve, then call MC_CamTableSelect again to reselect the cam table.
- Need to use MC_CamOut command to release the cam coupling relationship between the master and slave axis.
- When the command is executed, the cam relationship between the slave axis and the spindle will be released when the slave axis of the command executes other motion commands, and Command-Aborted variable of MC_CamIn is output as TRUE.

3-3-2-3. Cam unbinding [MC_CamOut]

(1) Instruction overview

Release the cam coupling relationship between the specified slave axis and its corresponding spindle.

Cam unbinding [MC_CamOut]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
MC_CamOut	release the cam action		MC_CamOut_0(Slave:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Slave	slave axis	AXIS_REF	-	-	map to the axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	TRUE when instruction execution is completed
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

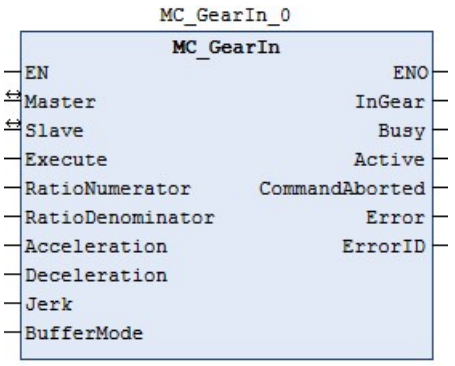
- This instruction is implemented by the "SM3_Basic" library.
- This command is used to release the cam coupling relationship between the specified slave axis and its corresponding spindle.
- Release the slave axis cam coupling relationship at the rising edge of Execute.
- After the cam relationship is disconnected, the slave axis may not necessarily stop. If the speed of the slave shaft is not 0 before executing the command, then after the command DONE signal is completed, the cam coupling relationship is disconnected, but the slave axis still operates at the speed before cutting out.
- If the command is executed without a cam coupling relationship for the slave axis, an ERROR error will occur.

3-3-2-4. CAM action start [MC_GearIn]

(1) Instruction overview

Set the gear ratio of the master and slave axes and activate the electronic gears.

CAM action start [MC_GearIn]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
MC_GearIn	cam action start		MC_GearIn_0(Master:= , Slave:= , Execute:= , RatioNumerator:= , RatioDenominator:= , Acceleration:= , Deceleration:= , Jerk:= , BufferMode:= , InGear=> , Busy=> , Active=> , CommandAborted=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Master	master axis	AXIS_REF	-	-	map to master axis
Slave	slave axis	AXIS_REF	-	-	map to slave axis
VAR_INPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
Execute	valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
RatioNumerator	numerator of gear ratio	DINT	positive number	1	offset on the master table
RatioDenominator	denominator of gear ratio	UDINT	positive number	1	offset on the slave table
Acceleration	target acceleration	LREAL	positive number	0	acceleration [u/s^2]
Deceleration	target deceleration	LREAL	positive number	0	deceleration [u/s^2]
Jerk	target jerk speed	LREAL	positive number	0	jerk speed [u/s^3]
BufferMode	buffer mode	MC_BUFFER_MODE	-	-	-

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
InGear	gear ratio reached	BOOL	TRUE/FALSE	FALSE	The slave station moves at the given master station speed ratio, and the Target speed reached by the slave axis is TRUE
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	in control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- Start the cam action at the rising edge of Execute
- To decouple after executing the electronic gear, the GearOut command must be used.
- This command is a speed electronic gear function, and the loss of synchronization distance caused during the acceleration process will not be automatically compensated.
- When the Busy signal is TRUE during command execution, repeatedly triggering the rising edge of Execute will not restart the command regardless of whether the target speed of the slave axis has been reached.
- When reaching the target speed, InGear is TRUE, and at this time, the slave axis movement quantity=the spindle movement quantity * RatioNumerator/RatioDenominator.
- If the spindle speed changes in real time, please use this command with caution.

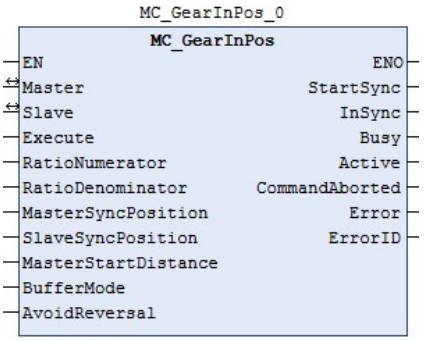
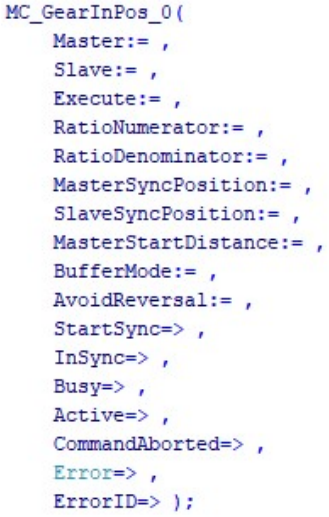
Note: Do not use MC_SetPosition during instruction execution to prevent accidents caused by rapid motor operation.

3-3-2-5. Position specified gear action [MC_GearInPos]

(1) Instruction overview

Set the electronic gear ratio between the main axis and the slave axis for electronic gear action.
Specify the spindle position and slave axis position of starting synchronization, and the starting synchronization distance of the spindle, and use this to complete the action of entering the electronic gear.

Position specified gear action [MC_GearInPos]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
MC_GearInPos	position specified gear action		

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Master	master axis	AXIS_REF	-	-	map to master axis
Slave	slave axis	AXIS_REF	-	-	map to slave axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
RatioNumerator	numerator of gear ratio	DINT	positive number	1	master axis offset
RatioDenominator	denominator of gear ratio	UDINT	positive number	1	slave axis offset
MasterSyncPosition	Spindle synchronization position	LREAL	Negative number, 0, positive number	0	Main position for axis synchronous operation
SlaveSyncPosition	slave axis synchronization position	LREAL	Negative number, 0, positive number	0	Slave position for axis synchronous operation
MasterStartDistance	Execute	LREAL	Negative number,	0	The main distance of the gears in

	synchronous spindle position		0, positive number		the program (start the slave axis to enter synchronization)
BufferMode	buffer mode	MC_BUFFER_MODE	-	-	-
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
AvoidReversal	Prohibit reverse run	BOOL	TRUE/FALSE	FALSE	FALSE: Indicates that slave axis reversal is physically possible and acceptable. TRUE: Indicates that slave axis reversal is physically impossible, Otherwise, it may cause danger. It is only applicable to modular driven axis. If reverse rotation cannot be avoided, the axis will stop due to an error
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
StartSync	Start synchronization	BOOL	TRUE/FALSE	FALSE	If the electronic gear starts processing, it is true
InSync	Arrival synchronization	BOOL	TRUE/FALSE	FALSE	Electronic gear command completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Active	In control	BOOL	TRUE/FALSE	FALSE	Change to TRUE in control
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is TRUE
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- This instruction is implemented by the "SM3_Basic" library.
- After starting the action, the slave axis target speed is the speed obtained by multiplying the spindle speed by the gear ratio, and the acceleration and deceleration actions are carried out.
- The process from the start of synchronization to the end of synchronization in this function block is essentially an electronic cam that slave axis follows the spindle within the synchronization interval. At this time, based on the spindle range (MasterSyncPosition-MasterStartDistance, MasterSyncPosition) and the slave axis range (current position, SlaveSyncPosition), the command will automatically design a cam curve based on the set gear ratio and the above three parameters, When performing synchronization, slave axis will follow the spindle to complete the cam movement.
- If the master-slave axis operates in linear mode, it is necessary to ensure that the above parameters are set reasonably, otherwise the gear action cannot be carried out correctly. Therefore, it is recommended to use this command when the master-slave axis is in modulus axis mode. For example, both the master and slave axis linear working modes move in a forward direction, if the master axis position

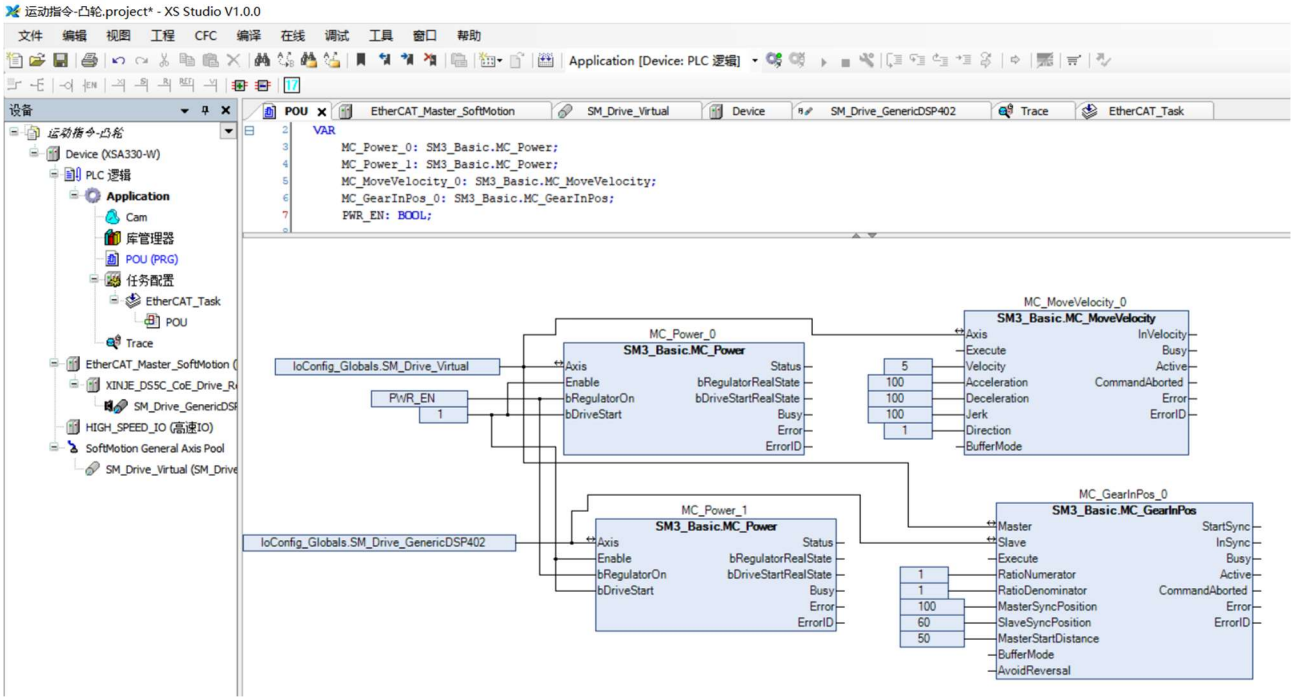
MasterSyncPosition-MasterStartDistance or the slave axis position>SlaveSyncPosition, the electronic gear action cannot be cut in.

- At the same time as synchronization is completed (InSync is true), the target speed is reached, and then the slave axis movement quantity=the spindle movement quantity RatioNumerator/RatioDenominator
- For AvoidReversal: If the slave axis is a modal axis and the spindle speed (the multiple relationship of gear ratio) is not relative to the velocity relationship of the slave axis, then MC_GearInPos will try to avoid reversing of the slave axis. It attempts to "stretch" the motion of the slave axis by adding 5 slave axis cycles. If this stretching is invalid, an error will occur and slave axis will error stop. If the slave axis speed is associated with the spindle speed (which is a multiple of the gear ratio), an error will occur and the axis will error stop. If the slave axis is a linear axis mode axis, an error will occur at the rising edge of Execute.

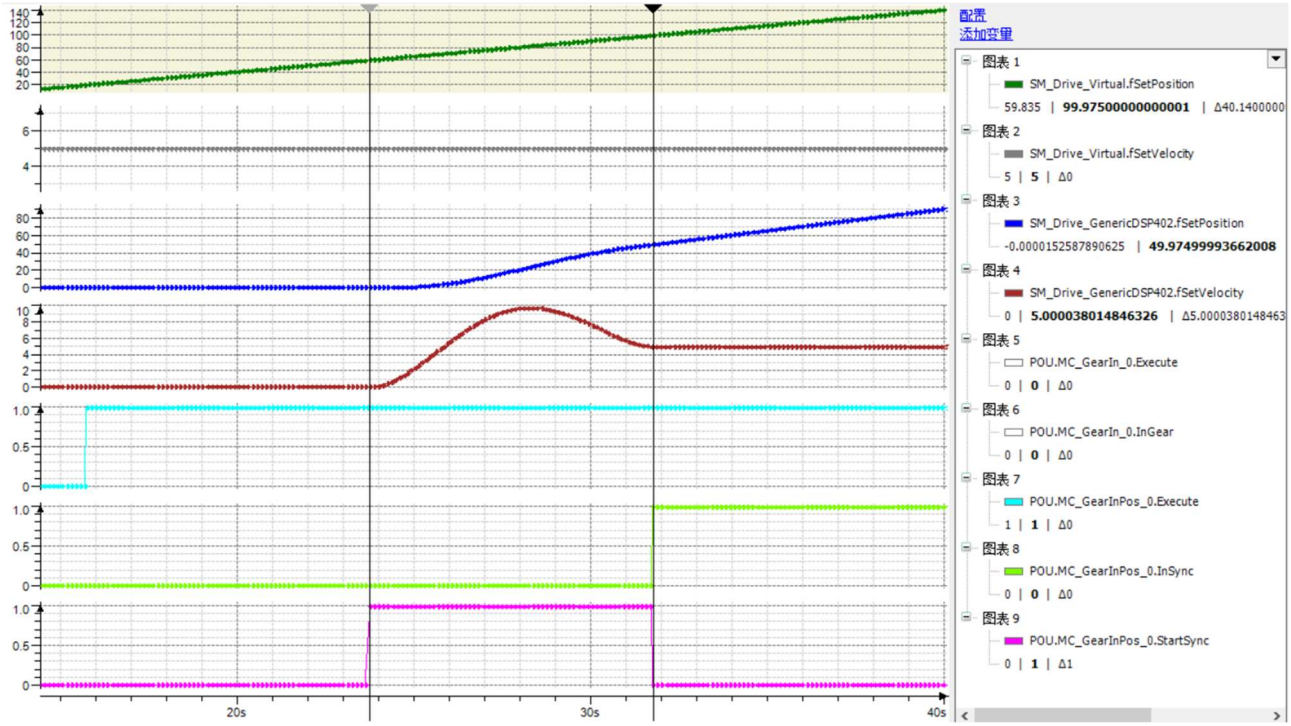
(4) Application

Example 1: Set the electronic gear ratio between the main axis and the slave axis to perform electronic gear action. Specify the spindle position and slave axis position to start synchronization, and the starting synchronization distance of the spindle, and use this to complete the action of entering the electronic gear. Select the virtual axis for the main axis, select the Ethercat real axis from the slave axis, set the main axis speed to 5u/s, and the direction is positive.

(1) Programming: The starting positions of both axes are set to 0, the spindle is set to a speed of 5 u/s using the function block "MC_MoveVelocity", and the direction is positive. The electronic gear ratio of the two axes is set to 1:1 using "MC_GearInPos", the spindle synchronization position is 100u, the slave axis synchronization position is 50u, and the spindle starting synchronization distance is 40u.



(2) You can use 'Trace' to view the current status. As can be seen from the cursor, synchronization starts at 60u for the spindle position. After synchronization is completed, the spindle position is at 100u, the slave axis position is at 50u, and the slave axis speed is at 5u/s.

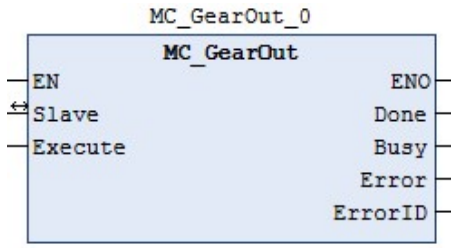

配置
添加变量
图表 1
SM_Drive_Virtual.fSetPosition
59.835 | 99.97500000000001 | Δ40.1400000
图表 2
SM_Drive_Virtual.fSetVelocity
5 | 5 | Δ0
图表 3
SM_Drive_GenericDSP402.fSetPosition
-0.0000152587890625 | 49.97499993662008
图表 4
SM_Drive_GenericDSP402.fSetVelocity
0 | 5.000038014846326 | Δ5.000038014846
图表 5
POU.MC_GearIn_0.Execute
0 | 0 | Δ0
图表 6
POU.MC_GearIn_0.InGear
0 | 0 | Δ0
图表 7
POU.MC_GearInPos_0.Execute
1 | 1 | Δ0
图表 8
POU.MC_GearInPos_0.InSync
0 | 0 | Δ0
图表 9
POU.MC_GearInPos_0.StartSync
0 | 1 | Δ1
20s
30s
40s

3-3-2-6. Gear action release [MC_GearOut]

(1) Instruction overview

Disconnect the electronic gear coupling between the slave axis and spindle.

Gear action release [MC_GearOut]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
MC_GearOut	gear action release		MC_GearOut_0(Slave:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Slave	slave axis	AXIS_REF	-	-	map to the axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
Done	completed	BOOL	TRUE/FALSE	FALSE	If the coupling between the slave axis and the spindle electronic gear is disconnected, it is true
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

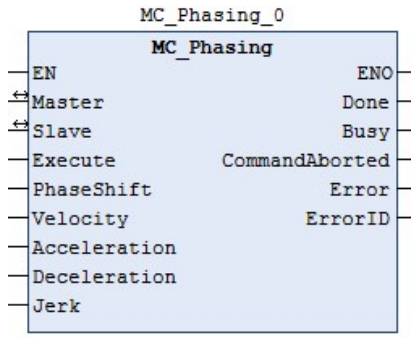
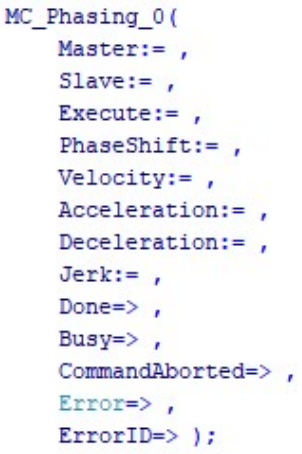
- This instruction is implemented by the "SM3_Basic" library.
- After cutting out the electronic gear, the speed from the slave axis is the speed before cutting out, so it is necessary to cooperate with MC_Stop command stops the slave axis.
- Specify the action object axis through Slave, specify Deceleration, and abort the executing command MC_GearIn (gear action start), MC_ GearInPos (position specified gear action) command.
- This instruction has no effect for the master axis action of command MC_GearIn and MC_GearInPos.

3-3-2-7. Master slave axis phase offset [MC_Phasing]

(1) Instruction overview

Specify the phase deviation between the master and slave axes.

Master slave axis phase offset [MC_Phasing]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
MC_Phasing	master slave axis phase offset		

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Master	master axis	AXIS_REF	-	-	map to master axis
Slave	slave axis	AXIS_REF	-	-	map to the axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
PhaseShift	Phase compensation	LREAL	Negative number, 0, positive number	0	Specify the phase compensation amount for the spindle
Velocity	speed	LREAL	0, positive number	0	Maximum speed value when performing phase shift [command unit/s]
Acceleration	acceleration	LREAL	0, positive number	0	Maximum acceleration during phase shift execution [command unit /s^2]
Deceleration	deceleration	LREAL	0, positive number	0	Maximum deceleration during phase shift execution [command unit /s^2]
Jerk	jerk speed	LREAL	0, positive number	0	Maximum jerk speed during phase shift execution [command unit /s^3]
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description

Done	completed	BOOL	TRUE/FALSE	FALSE	If the electronic gear coupling between the slave axis and the spindle is disconnected, it is true
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
CommandAborted	instruction interrupted	BOOL	TRUE/FALSE	FALSE	Module execution interrupted is true
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

- Perform phase shift at the rising edge of Execute, slave axis automatically calculates a smooth curve to complete the phase shift from the slave axis to the main axis. The phase difference between the main and slave axes is the PhaseShift value of the input signal, and a positive value is the lag of the slave axis to the main axis.
- After completing the offset, the Done signal output is true.
- Compensate for the phase difference between the master and slave axes based on the set PhaseShift, Velocity, Acceleration, and Deceleration.
- When the phase difference between the master and slave axes reaches PhaseShift, the Done signal is output.
- When executing the command, the spindle command position and feedback position remain unchanged, and the slave axis is adjusted. After completion, the phase difference between the slave axis and the spindle is PhaseShift. The final result of this command is the phase offset between the given values of the axis, so the actual feedback value of the real axis may not be consistent with the final offset.

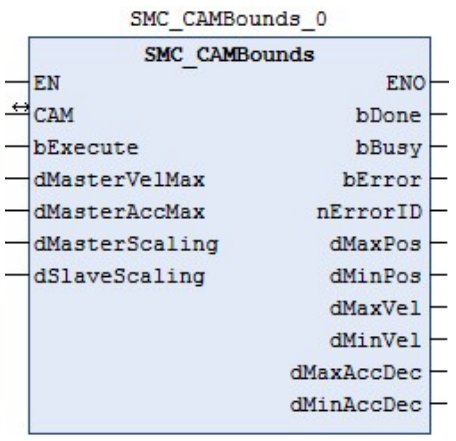
3-3-2-8. CAM range [SMC_CAMBounds]

(1) Instruction overview

When the slave axis is coupled with the spindle cam, the maximum position, speed, and acceleration of the slave axis can be calculated using this function block.

The spindle moves under maximum input speed and acceleration/deceleration restrictions. This command can check whether the curve is correct when designing the cam table, provided that the maximum acceleration, deceleration, speed, etc. of the spindle are known.

CAM range [SMC_CAMBounds]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
SMC_CAMBounds	CAM range		SMC_CAMBounds_0(CAM:= , bExecute:= , dMasterVelMax:= , dMasterAccMax:= , dMasterScaling:= , dSlaveScaling:= , bDone=> , bBusy=> , bError=> , nErrorID=> , dMaxPos=> , dMinPos=> , dMaxVel=> , dMinVel=> , dMaxAccDec=> , dMinAccDec=>);

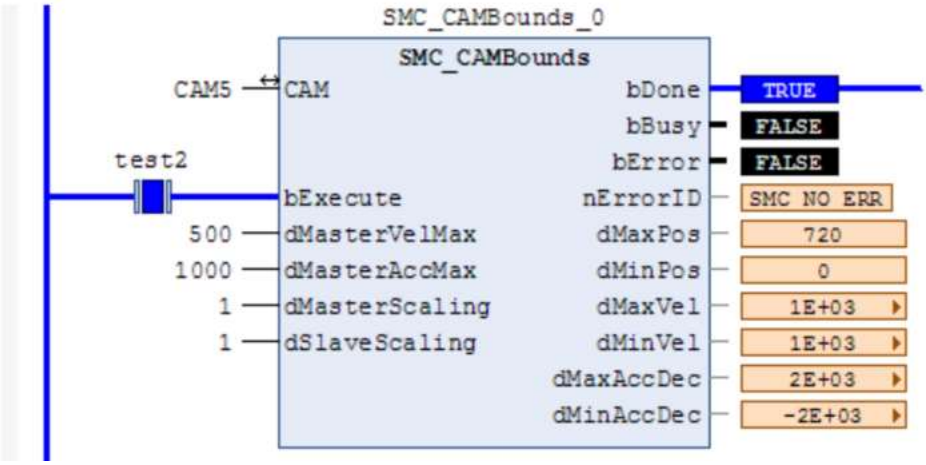
(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Cam	CAM	MC_CAM_REF	-	-	map to the cam
VAR_INPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
bExecute	valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
dMasterVelMax	max speed	LREAL	-	1	Maximum spindle speed in absolute mode
dMasterAccMax	max acceleration	LREAL	-	0	Maximum spindle acceleration in absolute mode
dMasterScaling	Scale Factor	LREAL	-	1	Scale factor for spindle cam application
dSlaveScaling	Scale Factor	LREAL	-	1	Scale factor for slave cam application
VAR_OUTPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
bDone	Completed	BOOL	TRUE/FALSE	FALSE	If the coupling between the slave axis and the spindle electronic gear is disconnected, it is true
bBusy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
bError	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
nErrorID	Error code	SMC_ERROR	-	0	Error recognition
dMaxPos	Max position	LREAL	-	0	Calculate the maximum position of the slave axis based on the cam table
dMinPos	Min position	LREAL	-	0	Calculate the minimum position of the slave axis based on the cam table
dMaxVel	Max speed	LREAL	-	0	Calculate the maximum speed of the slave axis
dMinVel	Min speed	LREAL	-	0	Calculate the minimum speed of the slave axis
dMaxAccDec	Max acceleration	LREAL	-	0	Calculate the maximum acceleration of the slave axis

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
dMinAccDec	Min acceleration	LREAL	-	0	Calculate the minimum acceleration of the slave axis

(3) Function description

- Based on the rising edge of bExecute, the "dMasterVelMax", "dMasterAccMax", "dMasterScaling", and "dSlaveScaling" values of the input variables are integrated with the cam table data to calculate the equivalent value of the "maximum position" and "minimum position" of the slave axis. For example, if the spindle cycle is 360 and the cam table is a straight line with a slope of 2, the calculated result is shown in the following figure:



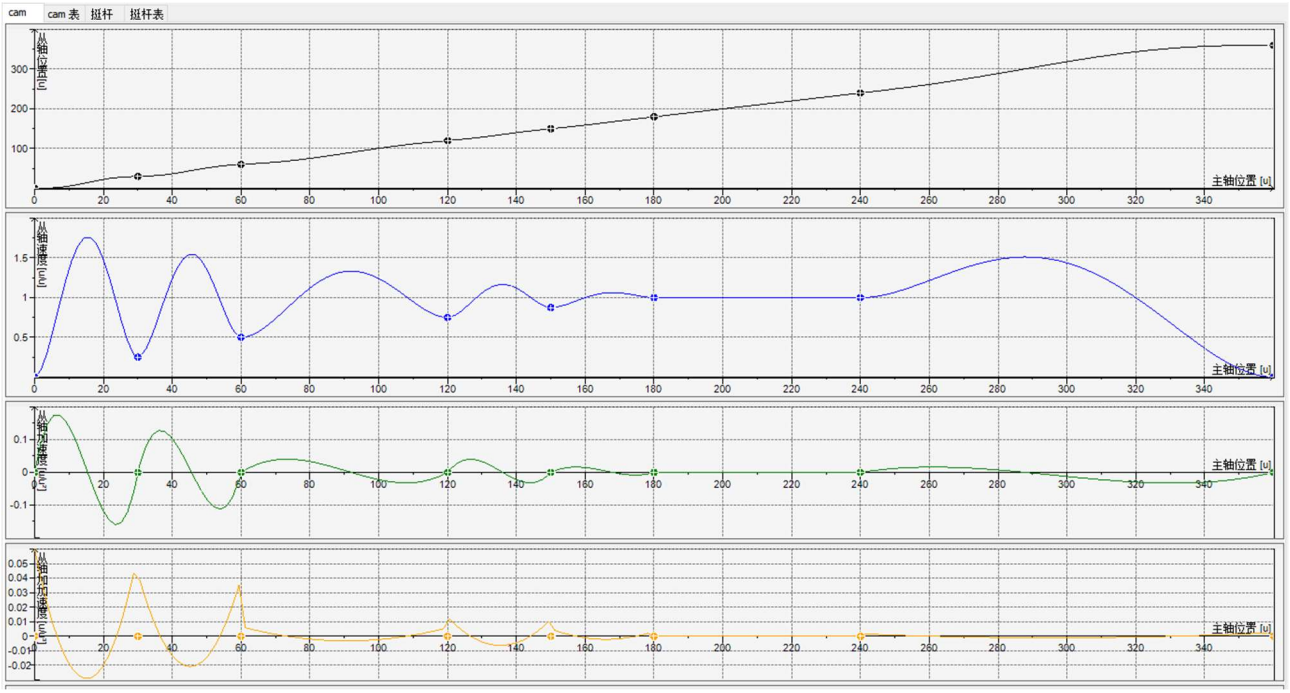
- The spindle can be calculated using this command when running in absolute mode or when the spindle is set to cycle mode, and the modulus is set to spindle cycle.
- The cam table is XYVA (valid in polynomial mode), and one-dimensional arrays, two-dimensional arrays, etc. are invalid.

(4) Application

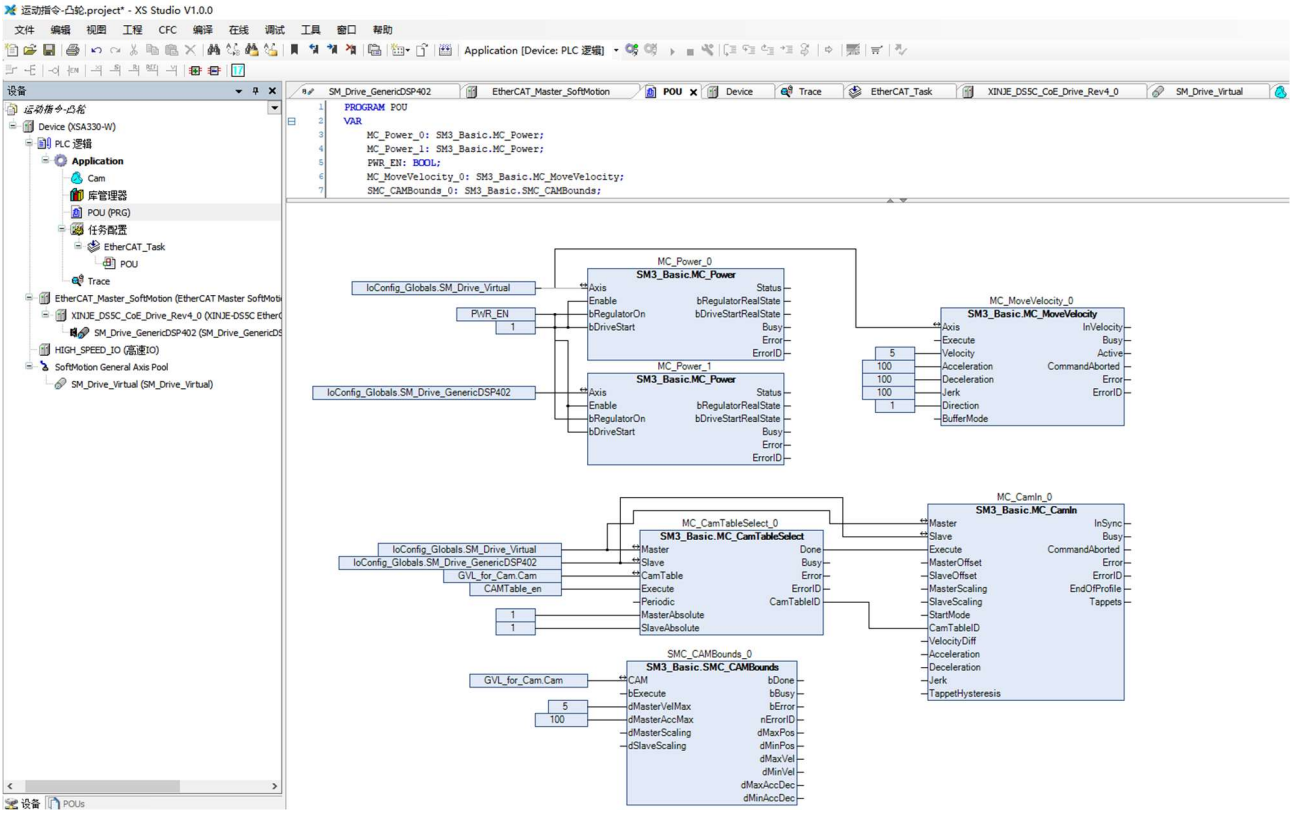
Example: Establish an electronic cam table, and calculate the maximum position, speed, acceleration/deceleration of the slave axis through "SMC_CAMBounds" after coupling the slave axis and the spindle cam. Select the virtual axis for the main axis and the Ethercat real axis for the slave axis.

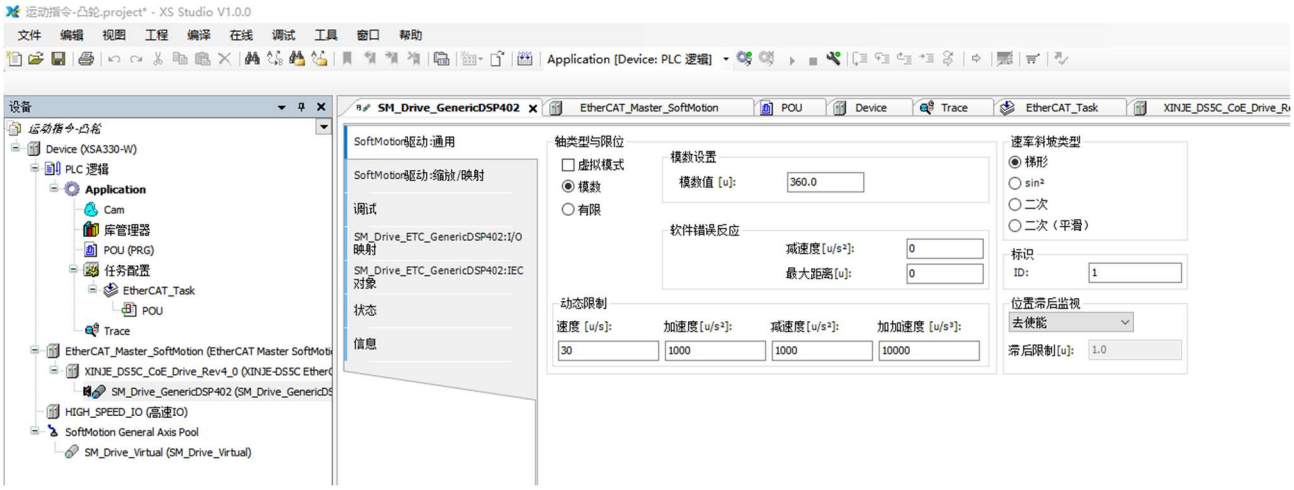
Program:

Right click on "Application" in the project device bar, select "Add Object" - "Cam Table", name it Cam, and open it to add the corresponding cam table parameters as shown in the following figure.

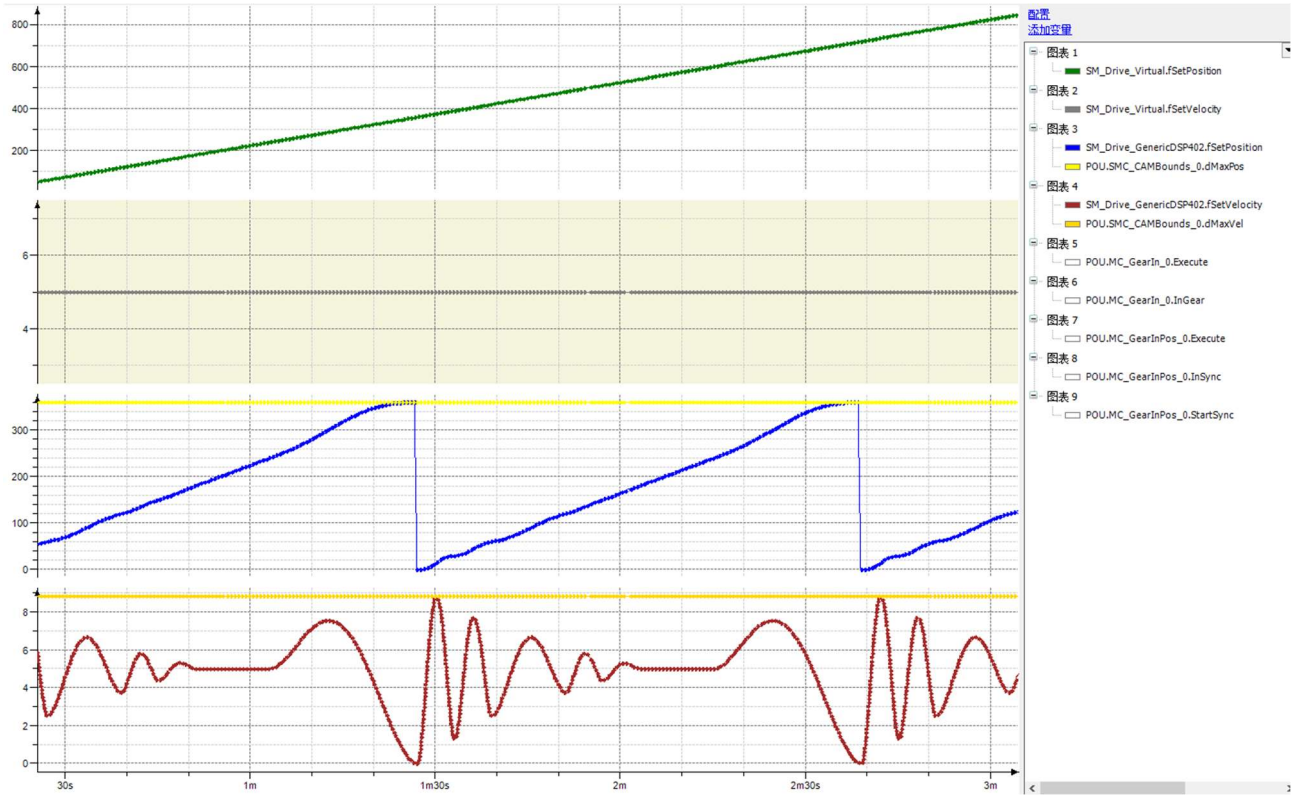


Select the cam table by instruction MC_CamTableSelect, select the coupling mode as absolute, and configure the electronic cam module MC_CamIn. Set the spindle running speed to 5 and acceleration to 100 using "MC-MoveVelocity", and calculate the maximum position, speed, and acceleration of the slave station using "SMC_CAMBounds". The master station runs synchronously with the master slave bounding, and sets the slave axis running mode to modulus - running between 0 and 360.





By monitoring the position and speed of the slave axis through "Trace", it can be seen that the maximum position and speed matched during operation.

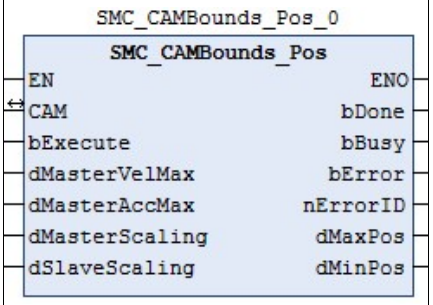


3-3-2-9. CAM position range [SMC_CAMBounds_Pos]

(1) Instruction overview

When the slave axis is coupled with the spindle cam, the maximum and minimum positions of the slave axis can be calculated using this function block. Compare to SMC_CAMBounds, this function block lacks calculations such as maximum acceleration, and all other work is consistent.

CAM position range [SMC_CAMBounds_Pos]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
SMC_CAMBounds_Pos	CAM position range		SMC_CAMBounds_Pos_0(CAM:= , bExecute:= , dMasterVelMax:= , dMasterAccMax:= , dMasterScaling:= , dSlaveScaling:= , bDone=> , bBusy=> , bError=> , nErrorID=> , dMaxPos=> , dMinPos=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Cam	CAM	MC_CAM_REF	-	-	map to the cam
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
bExecute	Valid	BOOL	TRUE/FALSE	FALSE	Rising edge signal, execute current command
dMasterVelMax	Max speed	LREAL	-	1	Maximum spindle speed in absolute mode
dMasterAccMax	Max acceleration	LREAL	-	0	Maximum spindle acceleration in absolute mode
dMasterScaling	Scale Factor	LREAL	-	1	Scale factor for spindle cam application
dSlaveScaling	Scale Factor	LREAL	-	1	Scale factor for slave axis cam application
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
bDone	Completed	BOOL	TRUE/FALSE	FALSE	If the coupling between the slave axis and the spindle electronic gear is disconnected, it is true
bBusy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
bError	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
nErrorID	Error code	SMC_ERROR	-	0	Error recognition

dMaxPos	Max position	LREAL	-	0	Calculate the maximum position of the slave axis based on the cam table
dMinPos	Min position	LREAL	-	0	Calculate the minimum position of the slave axis based on the cam table

(3) Function description

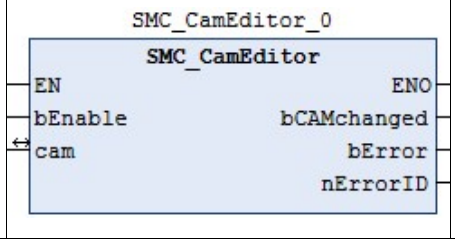
- Based on the rising edge of bExecute, the "dMasterVelMax", "dMasterAccMax", "dMasterScaling", and "dSlaveScaling" values of the input variables are combined with the cam table data to calculate the "maximum position" and "minimum position" of the slave axis.
- The spindle can be calculated using this command when running in absolute mode or when the spindle is set to cycle mode, and the modulus is set to spindle cycle.
- The cam table is XYVA (valid in polynomial mode), and one-dimensional arrays, two-dimensional arrays, etc. are invalid.

3-3-2-10. Display cam table in visualization [SMC_CamEditor]

(1) Instruction overview

This function block is used to display cam table in visualization. This feature block does not work without TargetVideo or WebVideo.

Display cam table in visualization [SMC_CamEditor]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA (-L not support visualization)
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
SMC_CamEditor	display cam table in visualization		SMC_CamEditor_0(bEnable:= , bCAMchanged=> , bError=> , nErrorID=> , cam:=);

(2) Related variables

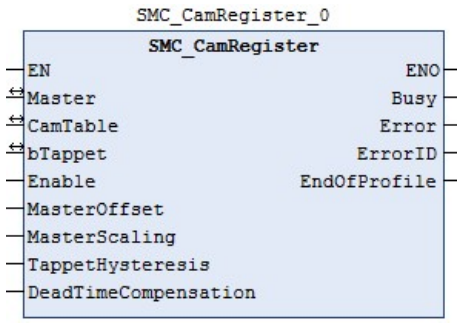
VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Cam	CAM	MC_CAM_REF	-	-	map to the cam
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
bEnable	valid	BOOL	TRUE/FALSE	FALSE	Modify the cam table when it is true
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
bCAMchanged	Completed	BOOL	TRUE/FALSE	FALSE	The cam in the visualization has changed. Then it is true
bError	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
nErrorID	Error code	SMC_ERROR	-	0	Error recognition

3-3-2-11. CAM tappet control [SMC_CamRegister]

(1) Instruction overview

Implement cam tappet control (cam switch). When editing the cam, it is possible not to edit the curves of the master and slave axes, and simply configure the tappet table to achieve tappet control through this function block.

CAM tappet control [SMC_CamRegister]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
SMC_CamRegister	CAM tappet control		SMC_CamRegister_0(Master:= , CamTable:= , bTappet:= , Enable:= , MasterOffset:= , MasterScaling:= , TappetHysteresis:= , DeadTimeCompensation:= , Busy=> , Error=> , ErrorID=> , EndOfProfile=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Master	Master axis	AXIS_REF	-	-	map to the master axis
CamTable	CAM table	MC_CAM_REF	-	-	Mapping to an electronic cam, i.e. an electronic cam instance
bTappet	Tappet output	ARRAY [1..MAX_NUM_TAPPETS]OF BOOL	-	-	tappet point output
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Enable	Executing	BOOL	TRUE/FALSE	FALSE	Execute current instruction is TRUE
MasterOffset	master axis position	LREAL	-	0	Offset on main table
MasterScaling	master axis scale	LREAL	-	1	Scale factor of the main configuration file
TappetHysteresis	tappet hysteresis	LREAL	-	-	tappet hysteresis size
DeadTimeCompensation	Dead time compensation	LREAL	-	0	Deadband time compensation in seconds. The expected spindle position will be calculated using linear extrapolation method
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Busy	Executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	Error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	Error code	SMC_ERROR	-	0	Error recognition
EndOfProfile	Curve cycle completion	BOOL	TRUE/FALSE	FALSE	True if the spindle position is greater than or equal to the set cycle

(3) Function description

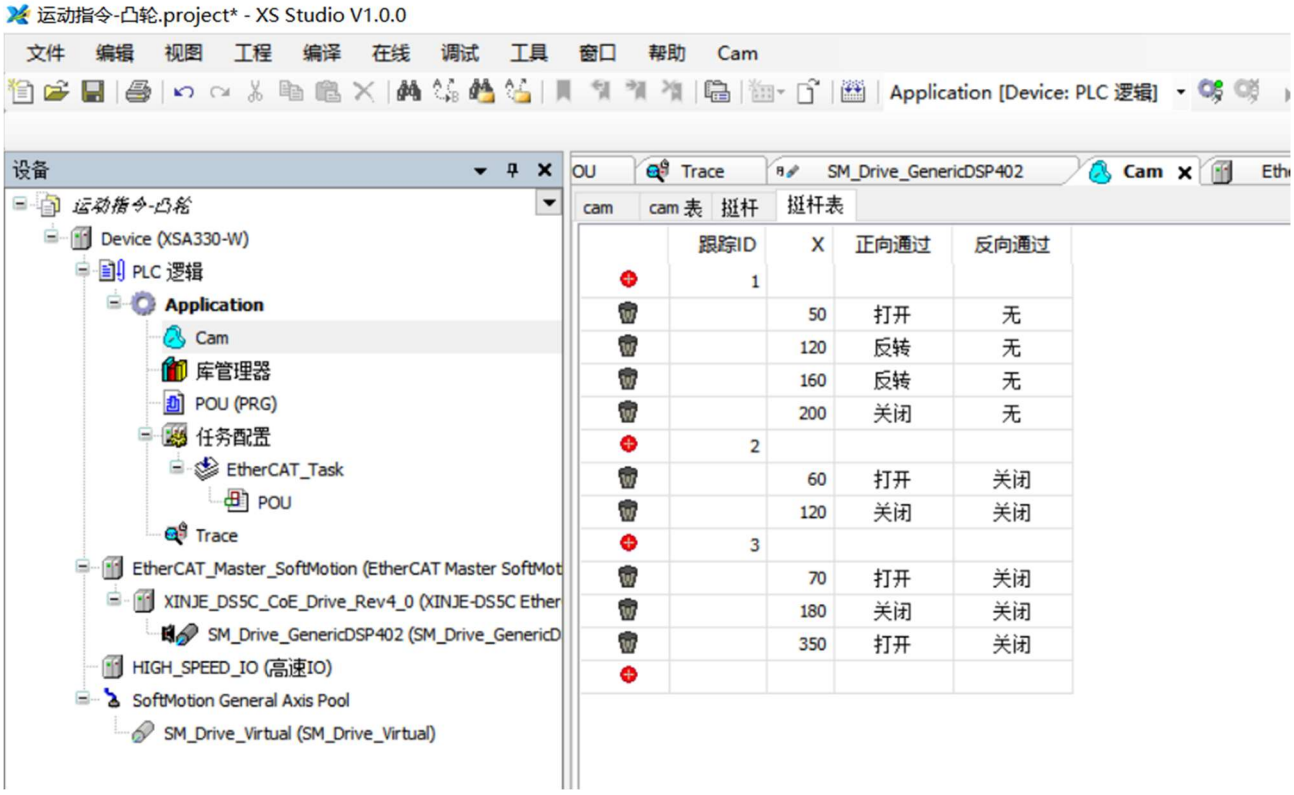
- Enable signal is TRUE, and if there is no error output, Busy output is TRUE to execute tappet control.
- The control function block is not related to the slave axis in the electronic cam, only the spindle cycle and tappet table need to be configured.
- bTappet is a one-dimensional Boolean structure (MAX_NUM_TAPPETS=512), and bTappet[i] corresponds to ith tappet output DeadTimeCompensation, the unit is second. When set to a positive value, it is ahead the tappet signal, and when set to a negative value, it is behind the tappet signal.

(4) Application

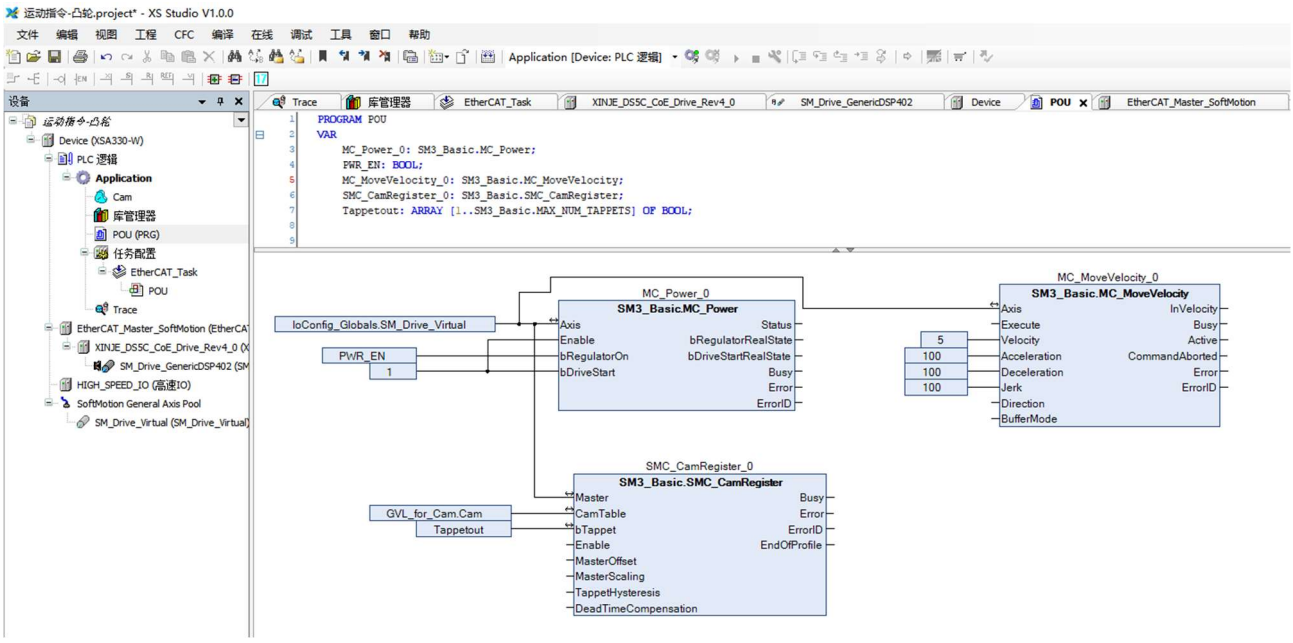
Example: Set 3 tappets in the tappet table in the cam table. Set the spindle speed to 5 and the acceleration to 100, and run the spindle through the trajectory set in the cam table. The main axis is selected as the virtual axis, and the slave axis is selected as the real axis.

Programming

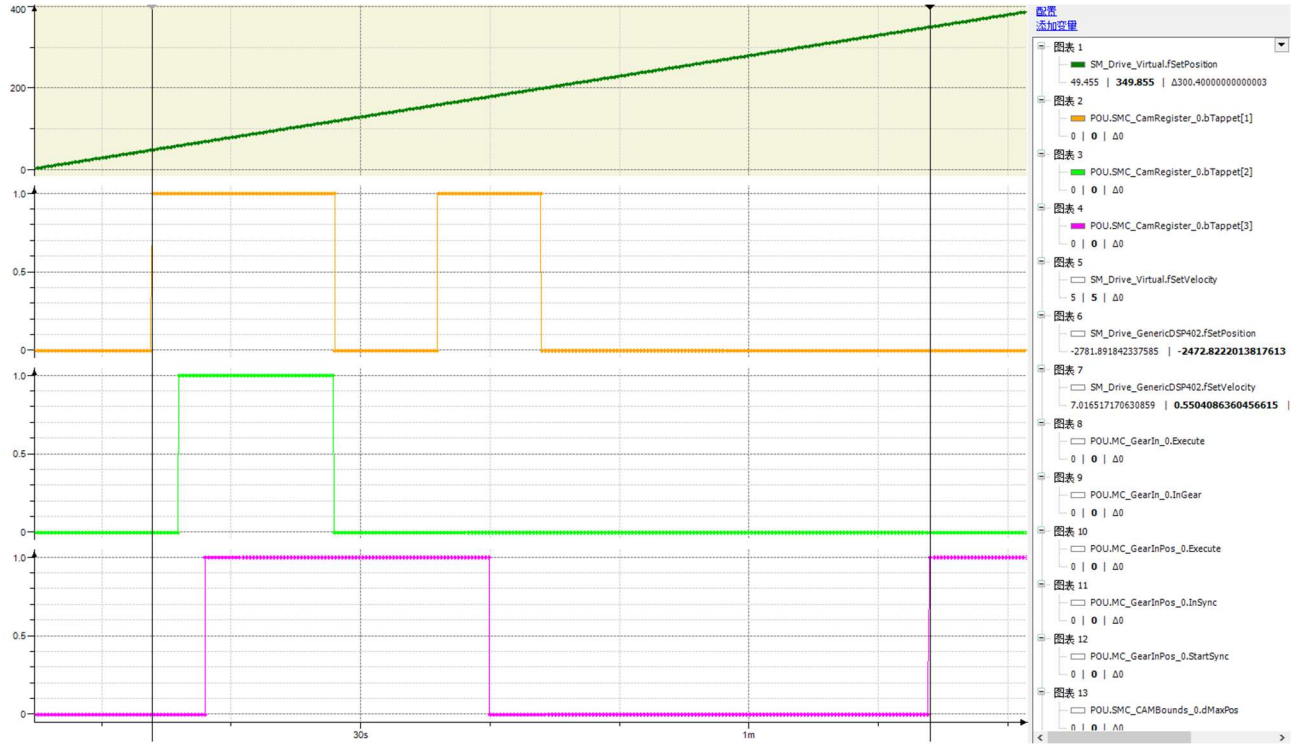
Right click on "Application" in the project device bar, select "Add Object" - "Cam Table", name it Cam, and open it to add the corresponding tappet parameters as shown in the following figure.



Add enabling module MC_ Power in the main program, spindle running speed module MC_MoveVelocity. By using the function block "SMC_CamRegister", the output status of all IDs in the tappet table can be monitored.



You can see all the current tappet output states of the spindle in 'Trace'.

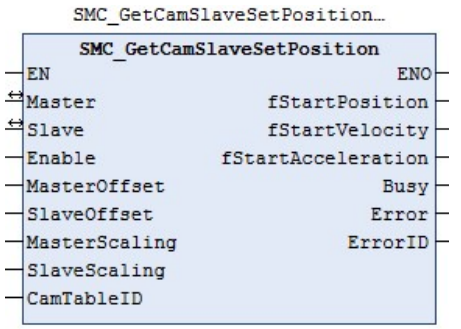


3-3-2-12. Get cam slave axis position [SMC_GetCamSlaveSetPosition]

(1) Instruction overview

Read the cam table slave axis position, speed, and acceleration information.

Get cam slave axis position [SMC_GetCamSlaveSetPosition]			
Execution condition	Normally ON/OFF triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
SMC_GetCamSlaveSetPosition	obtain the cam slave axis position		SMC_GetCamSlaveSetPosition_0(Master:= , Slave:= , Enable:= , MasterOffset:= , SlaveOffset:= , MasterScaling:= , SlaveScaling:= , CamTableID:= , fStartPosition=> , fStartVelocity=> , fStartAcceleration=> , Busy=> , Error=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Master	master axis	AXIS_REF	-	-	map to the master axis
Slave	slave axis	AXIS_REF	-	-	map to the slave axis
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
Execute	valid	BOOL	TRUE/FALSE	FALSE	execute current command at the rising edge signal
MasterOffset	master axis offset	LREAL	-	0	master table offset
SlaveOffset	slave axis offset	LREAL	-	0	slave table offset
MasterScaling	master scale	LREAL	-	1	Scale factor of the main configuration file
SlaveScaling	slave axis	LREAL	-	1	Scale factor of the slave configuration file
CamTableID	cam table	MC_CAM_ID	-	-	define the using of cam, use together with MC_CamTable Select output point CamTableI
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
fStartPosition	slave axis position	LREAL	-	0	The position of the slave axis obtained from the cam table and the current spindle information
fStartVelocity	slave axis speed	LREAL	-	0	The slave axis speed obtained from the cam table and the current spindle

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
					information
fStartAcceleration	slave acceleration	LREAL	-	0	The slave axis acceleration obtained from the cam table and the current spindle information
Busy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
Error	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
ErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

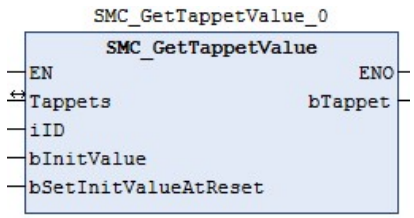
- The output value calculated by this instruction is: Y= (cam((Cam starting spindle lookup position +Masteroffset)* MasterScaling)+slaveoffset)* SlaveScaling, Cam is a function of the cam table. For example, if the starting spindle position of the cam is 0, the master-slave axis is scaled to 1, the Masteroffset is 100, and the slaveoffset is 0, the function block outputs the position of the slave axis corresponding to the cam table at 100.
- This function block only requires the successful construction of the cam table to read the position of the slave axis, and there is no requirement for the operation of the master slave axis.

3-3-2-13. Obtain tappet output value [SMC_GetTappetValue]

(1) Instruction overview

To read the current tappet status, it is necessary to use with MC_CamIn command.

Obtain tappet output value [SMC_GetTappetValue]			
Execution condition	-	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
SMC_GetTappetValue	Obtain the tappet output value		SMC_GetTappetValue_0(Tappets:= , iID:= , bInitValue:= , bSetInitValueAtReset:= , bTappet=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
Tappet	tappet	SMC_Tappet Data	-	-	map to a tappet
VAR_INPUT	Name	Data type	Effective range	Initial value	Description
iID	tappet ID	INT	positive value, 0	0	tappet ID
bInitValue	initial value	BOOL	TRUE/FALSE	FALSE	The initial value of the tappet to be allocated on the first call
bSetInitValueAtReset	tappet reset	BOOL	TRUE/FALSE	FALSE	If it is true, the output value of the tappet will be set to the initial value when the CamIn function block restarts. If it is FALSE, the value of the tappet will be maintained when the CamIn function block is restarted
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
bTappet	tappet value	BOOL	TRUE/FALSE	FALSE	tappet value

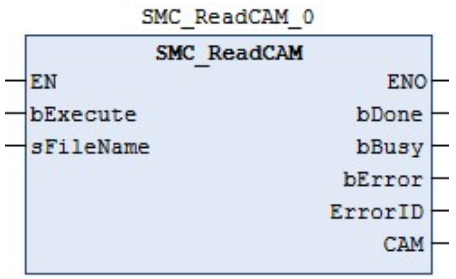
(3) Function description

- This function block needs to use together with MC_CamIn.

3-3-2-14. Read the cam table [SMC_ReadCAM]

(1) Instruction overview

Read the cam table [SMC_ReadCAM]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

Instruction	Name	Graphical representation	ST language
SMC_ReadCAM	read the cam table		SMC_ReadCAM_0(bExecute:= , sFileName:= , bDone=> , bBusy=> , bError=> , ErrorID=> , CAM=>);

(2) Related variables

VAR_INPUT	Name	Data type	Effective range	Initial value	Description
bExecute	execute	BOOL	TRUE/FALSE	FALSE	execute the function block at the rising edge
sFileName	file name	STRING(255)	-	-	The name of the file, which contains a cam description that defines ASCII format
VAR_OUTPUT	Name	Data type	Effective range	Initial value	Description
bDone	completed	BOOL	TRUE/FALSE	FALSE	True if Cam has been read in
bBusy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
bError	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
nErrorID	error code	SMC_ERROR	-	0	Error recognition
CAM	-	MC_CAM_REF	STRUCT(nElements : = 0, nTappets : = 0)	-	-

(3) Function description

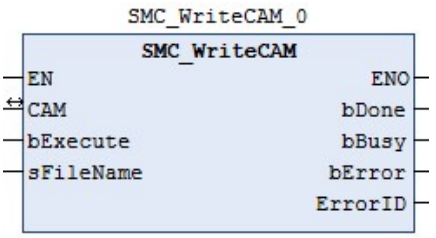
- This function block is designed for loading cams during runtime, and make it available for modules MC_CamTableSelect and MC_CamIn. The cam to be loaded must be created before the cam editor, and saved in "*.CAM" file (refer to SMC_WriteCAM).
- This function block needs to use together with MC_CamIn.
- The size of the loadable cam is limited by the global constant gc_SMC_FILE_MAXCAMEL (Number of Elements) and gc_SMC_FILE_ MAXCAMTAP (number of cam switch actions).

3-3-2-15. Modify the cam table [SMC_WriteCAM]

(1) Instruction overview

Used to store the cam table created in the cam editor in a file at runtime.

Modify the cam table [SMC_WriteCAM]			
Execution condition	Edge triggering	Suitable model	XS, XSA
Firmware	-	Software	-
Library	SM3_Basic		

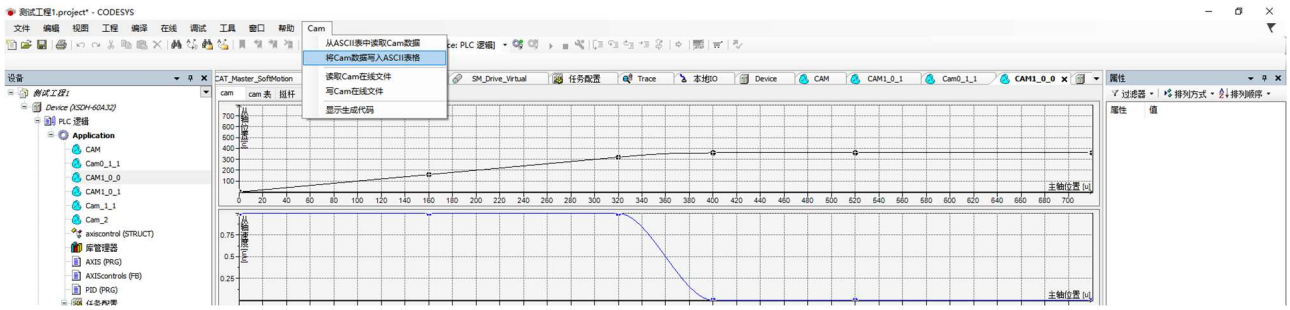
Instruction	Name	Graphical representation	ST language
SMC_WriteCAM	modify the cam table		SMC_WriteCAM_0(CAM:= , bExecute:= , sFileName:= , bDone=> , bBusy=> , bError=> , ErrorID=>);

(2) Related variables

VAR_IN_OUT	Name	Data type	Effective range	Initial value	Description
CAM	cam	MC_CAM_REF	-	-	map to the cam
VAR_INPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
bExecute	execute	BOOL	TRUE/FALSE	FALSE	execute the function block at the rising edge
sFileName	file name	STRING(255)			The name of the file, which contains the defined cam description in ASCII format
VAR_OUTPUT	**Name**	**Data type**	**Effective range**	**Initial value**	**Description**
bDone	completed	BOOL	TRUE/FALSE	FALSE	True if the cam is written into the file
bBusy	executing	BOOL	TRUE/FALSE	FALSE	True when the execution of the function block has not yet ended
bError	error	BOOL	TRUE/FALSE	FALSE	Function block execution error
nErrorID	error code	SMC_ERROR	-	0	Error recognition

(3) Function description

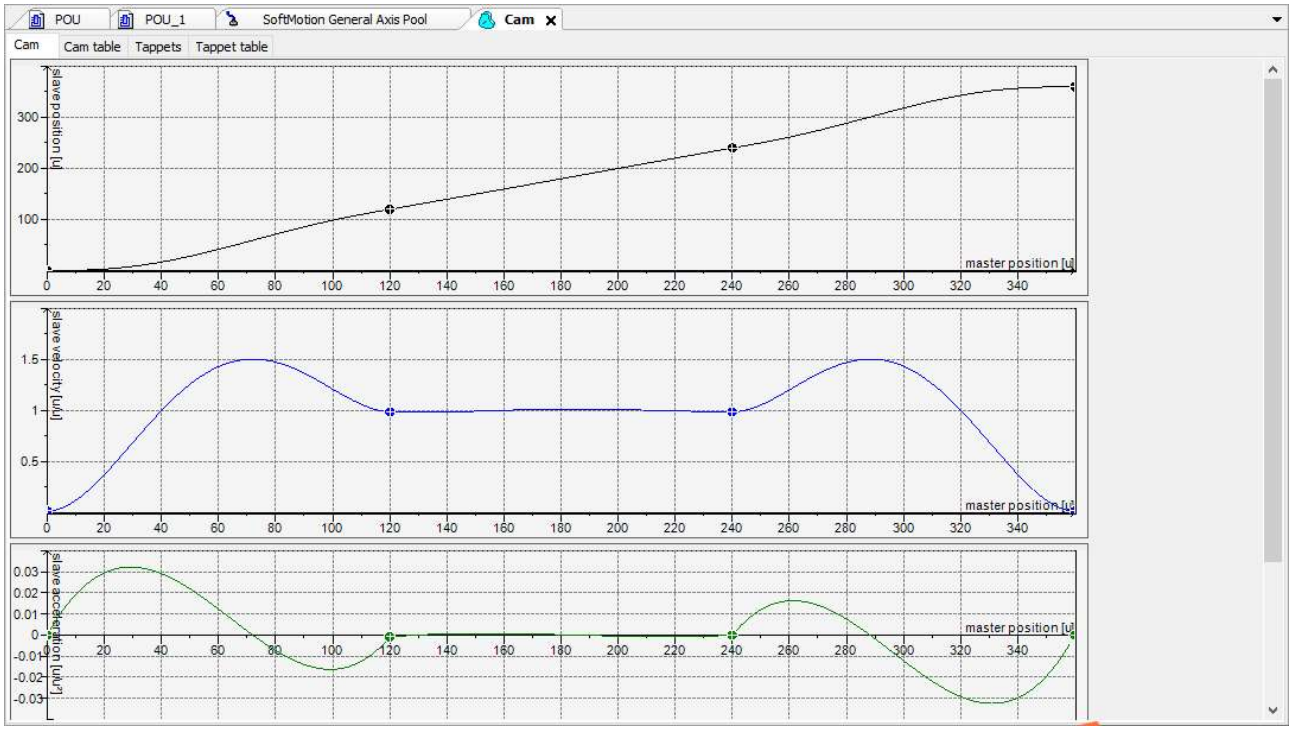
- Store the CAM information connected with “CAM” in the file “sFileName” at the rising edge of bExecut.
- Successfully stored and completed bDone signal output is true.
- The stored cam table information is limited by hardware memory.
- Attention: This function can be executed during program operation, and cam table information can also be manually stored in offline information.



3-3-3. CAM function application

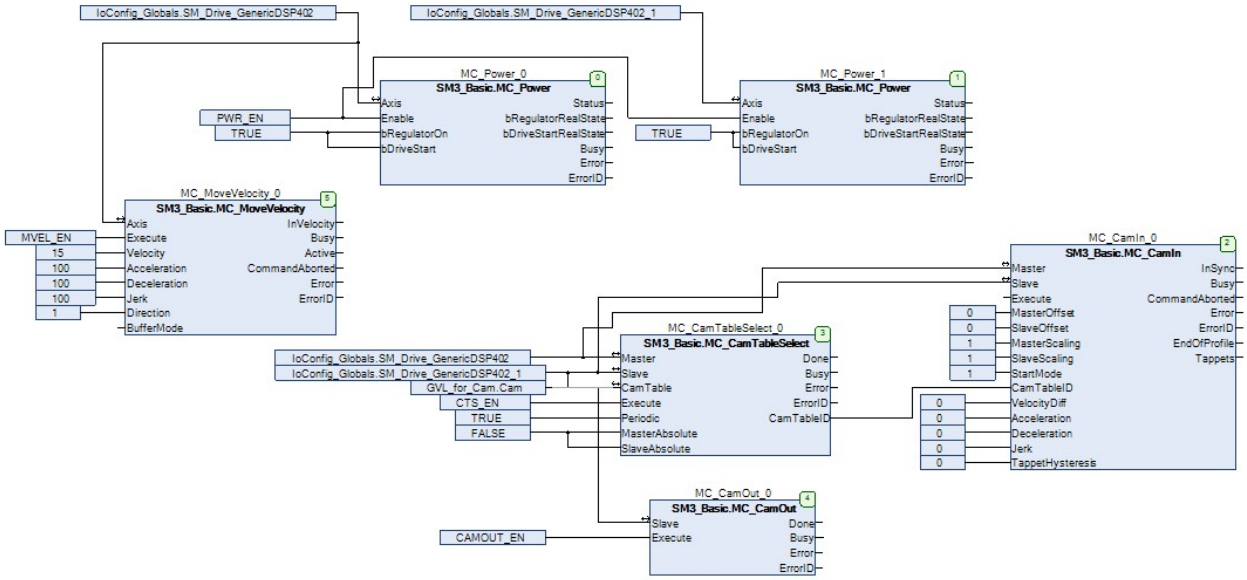
Example 1: Using an electronic cam to achieve spindle running speed mode, with a speed of 15 Plus/S, an acceleration of 100 Plus/S^2, and a positive direction. The slave station runs according to the cam table.
Program operation:

(1) Right click on "Application" in the engineering equipment bar, select "Add Object" - "Cam Table", name it Cam, and open it to add the corresponding cam table parameters as shown in the following figure.



(2) Add the enable module MC_Power, spindle running speed module MC_MoveVelocity in the main program POU_1, calling the cam table module MC_CamTableSelect, configured the electronic cam module MC_CamIn and unbinding electronic cam module MC_CamOut.

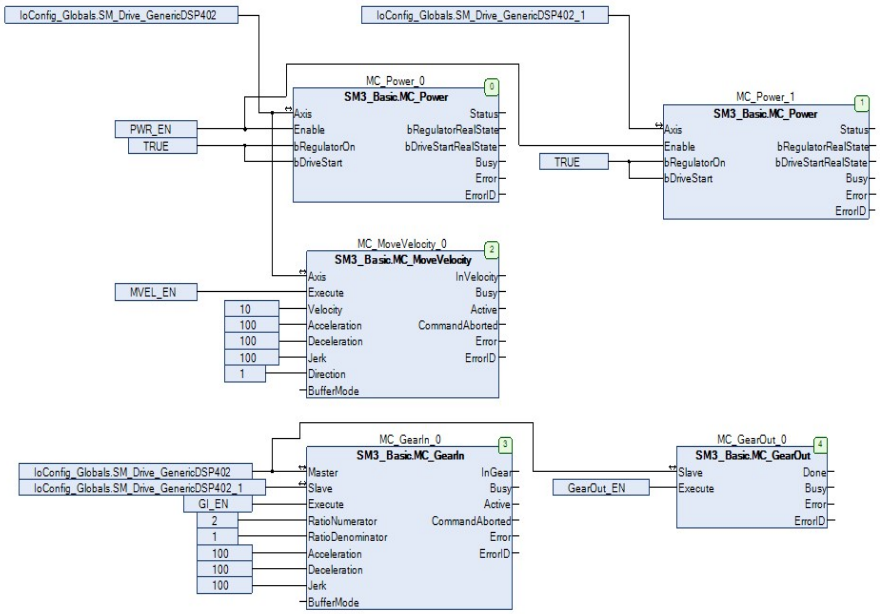
```
PROGRAM POU_1
VAR
    MC_Power_0: SM3_Basic.MC_Power;
    PWR_EN: BOOL;//主从使能 master slave enabled
    MC_Power_1: SM3_Basic.MC_Power;
    MC_CamIn_0: SM3_Basic.MC_CamIn;
    MC_CamTableSelect_0: SM3_Basic.MC_CamTableSelect;
    CTS_EN: BOOL;//凸轮表加载 load the cam table
    MC_CamOut_0: SM3_Basic.MC_CamOut;
    CAMOUT_EN: BOOL;//凸轮解绑 unbinding cam
    MC_MoveVelocity_0: SM3_Basic.MC_MoveVelocity;
    MVEL_EN: BOOL;//主轴速度模式 master axis speed mode
END_VAR
```



Note: If the slave axis is in motion during CAMOUT execution, and after the command is executed, the slave axis will continue to run at its original speed, stop the instruction through MC_Stop, MC_Halt.

Example 2: Using electronic gears to achieve spindle running speed mode, with a speed of 15 Plus/S, an acceleration of 100 Plus/S^2, and a positive direction. The slave station moves according to the electronic gear ratio of 2. So at this point, the speed of the slave station is 30 Plus/S.

```
PROGRAM POU_2
VAR
    MC_Power_0: SM3_Basic.MC_Power;
    MC_Power_1: SM3_Basic.MC_Power;
    PWR_EN: BOOL;//使能 enable
    MC_GearIn_0: SM3_Basic.MC_GearIn;
    MC_GearOut_0: SM3_Basic.MC_GearOut;
    MC_MoveVelocity_0: SM3_Basic.MC_MoveVelocity;
    MVEL_EN: BOOL;//速度模式 speed mode
    GearOut_EN: BOOL;//断开电子齿轮 disconnect the electronic cam
    GI_EN: BOOL;//绑定电子齿轮 binding the electronic cam
    SMC_GroupPower_0: SM3_Robotics.SMC_GroupPower;
END_VAR
```


IoConfig_Globals.SM_Drive_GenericDSP402
IoConfig_Globals.SM_Drive_GenericDSP402_1
MC_Power_0
SM3_Basic.MC_Power
0
Axis
Enable
bRegulatorOn
bDriveStart
Status
bRegulatorRealState
bDriveStartRealState
Busy
Error
ErrorID
PWR_EN
TRUE
MC_Power_1
SM3_Basic.MC_Power
1
Axis
Enable
bRegulatorOn
bDriveStart
Status
bRegulatorRealState
bDriveStartRealState
Busy
Error
ErrorID
TRUE
MC_MoveVelocity_0
SM3_Basic.MC_MoveVelocity
2
Axis
Execute
Velocity
Acceleration
Deceleration
Jerk
Direction
BufferMode
InVelocity
Busy
Active
CommandAborted
Error
ErrorID
MVEL_EN
10
100
100
100
1
MC_GearIn_0
SM3_Basic.MC_GearIn
3
Master
Slave
Execute
RatioNumerator
RatioDenominator
Acceleration
Deceleration
Jerk
BufferMode
InGear
Busy
Active
CommandAborted
Error
ErrorID
IoConfig_Globals.SM_Drive_GenericDSP402
IoConfig_Globals.SM_Drive_GenericDSP402_1
GI_EN
2
1
100
100
100
MC_GearOut_0
SM3_Basic.MC_GearOut
4
Slave
Execute
Done
Busy
Error
ErrorID
GearOut_EN

Appendix

Error code	Reasons	Description
0	SMC_NO_ERROR	no error
1	SMC_DI_GENERAL_COMMUNICATION_ERROR	Communication error. The fieldbus slave station is no longer operational
2	SMC_DI_AXIS_ERROR	axis error
3	SMC_DI_FIELDBUS_LOST_SYNCRONICITY	Loss of synchronization of fieldbus
10	SMC_DI_SWLIMITS_EXCEEDED	The position exceeds the SWLimit setting range
11	SMC_DI_HWLIMITS_EXCEEDED	Abnormal hardware position limit
12	SMC_DI_LINEAR_AXIS_OUTOFRANGE	Linear axis incremental position overflow
13	SMC_DI_HALT_OR_QUICKSTOP_NOT_SUPPORTED	Drive status Halt or Quickstop not supported
14	SMC_DI_VOLTAGE_DISABLED	Drive does not have power
15	SMC_DI_IRREGULAR_ACTPOSITION	This error is no longer used
16	SMC_DI_POSITIONLAGERROR	Excessive position error
17	SMC_DI_HOMING_ERROR	homing error
18	SMC_DI_LICENSING_ERROR	License issues
20	SMC_REGULATOR_OR_START_NOT_SET	Axis status cannot execute motion control commands
21	SMC_WRONG_CONTROLLER_MODE	Axis in incorrect controller mode
25	SMC_INVALID_ACTION_FOR_LOGICAL	Invalid logical axis action
30	SMC_FB_WASNT_CALLED_DURING_MOTION	The motion command is running and cannot be interrupted
31	SMC_AXIS_IS_NO_AXISS_REF	AXIS_REF variable type error
32	SMC_AXIS_REF_CHANGED_DURING_OPERATION	AXIS_REF variable has been swapped while the module is active
33	SMC_FB_ACTIVE_AXIS_DISABLED	Axis disabled during movement. MC_Power.bRegulatorOn
34	SMC_AXIS_NOT_READY_FOR_MOTION	Axis is unable to execute motion commands in its current state because Axis has not issued a signal indicating that it is following the target value
35	SMC_AXIS_ERROR_DURING_MOTION	Drive encountered an error while running
40	SMC_VD_MAX_VELOCITY_EXCEEDED	Over the max speed fMaxVelocity
41	SMC_VD_MAX_ACCERATION_EXCEEDED	Over the max acceleration fMaxAcceleration
42	SMC_VD_MAX_DEDRATION_EXCEEDED	Over the max deceleration fMaxDeceleration
50	SMC_3SH_INVALID_VELACC_VALUES	Invalid speed or acceleration value
51	SMC_3SH_MODE_NEEDS_HWLIMIT	Mode request using terminal switches for security reasons
67	SMC_MAC_TOO_MANY_TASK	There are too many tasks to generate axes using SDO
68	SMC_MAC_ATOMIC_ADD_FAILED	Atomic addition failed
69	SMC_SDO_INVALID_DATALENGTH	SDO reading resulted in invalid data length (>4)
70	SMC_SCM_NOT_SUPPORTED	not supported mode
71	SMC_SCM_AXIS_IN_Error_STATE	In the current mode, the controller mode cannot be changed
72	SMC_SCM_INTERRUPTED	SMC_SetControllerMode is interrupted by MC_Stop or errorstop
75	SMC_ST_WRONG_CONTROLLER_MODE	The axis is not in the correct controller mode. Abandoned and no longer returned by SMC_ SetTorque
80	SMC_RAG_ERROR_DURRING_STARTUP	axis group startup error
81	SMC_RAG_ERROR_AXIS_NOT_INITIALIZED	The axis has not reached the target state yet
85	SMC_PP_WRONG_AXIS_TYPE	Function block does not support virtual or

Error code	Reasons	Description
		logical axes
86	SMC_PP_NUMBER_OF_ABSOLUTE_BITS_INVALID	The absolute number of digits is invalid and must be within the range of 8~32
90	SMC_CGR_ZERO_VALUES	invalid value
91	SMC_CGR_DRIVE_POWERED	As long as the drive is in control mode, the transmission parameters must not be changed
92	SMC_CGR_INVALID_POSPERIOD	modulus cycle invalid (<=0 or >half of the bus bandwidth)
93	SMC_CGR_POSPERIOD_NOT_INTEGRAL	The modulus period expressed in increments is not an integer, but the modulus is processed by driver
110	SMC_P_FTASKCYCLE_EMPTY	Axis does not contain information about cycle time (fTaskCycle=0)
120	SMC_R_NO_ERROR_TO_RESET	Error free axis
121	SMC_R_DRIVE_DOESNT_ANSWER	Axis does not execute error reset
122	SMC_R_ERROR_NOT_RESETABLE	Unable to reset error
123	SMC_R_DRIVE_DOESNT_ANSWER_IN_TIME	Communication with axis is not working
124	SMC_R_CANNOT_RESET_COMMUNICATION_ERROR	If there is a communication error, it cannot be reset
130	SMC_RP_PARAM_UNKNOWN	Unknown parameter number
131	SMC_RP_REQUESTING_ERROR	Error transferring to drive. Please refer to the error number in the function block instance ReadDriveParameter
132	SMC_RP_DRIVE_PARAMETER_NOT_MAPPED	No available drive parameter assignments
133	SMC_RP_PARAM_CONVERSION_ERROR	Failed to convert value to drive parameter/from drive parameter to value. Unknown soft motion parameters
140	SMC_WP_PARAM_INVALID	Parameter number unknown or not allowed to be written
141	SMC_WP_SENDING_ERROR	Please refer to the error number in the function block instance WriteDriveParameter
142	SMC_WP_DRIVE_PARAMETER_NOT_MAPPED	No available drive parameter assignments
143	SMC_WP_PARAM_CONVERSION_ERROR	Failed to convert value to drive parameter/from drive parameter to value. Unknown soft motion parameters
170	SMC_H_AXIS_WASNT_STANDSTILL	The axis is not in a stationary state
171	SMC_H_AXIS_DIDNT_START_HOMING	Drive not started homing
172	SMC_H_AXIS_DIDNT_ANSWER	Drive not responding when homing is completed
173	SMC_H_ERROR_WHEN_STOPPING	Error stopping after reset. No need to set deceleration
174	SMC_H_AXIS_IN_ERRORSTOP	The drive is in an error stopped state. Unable to perform homing.
180	SMC_MS_UNKNOWN_STOPPING_ERROR	Unknown error while stopping
181	SMC_MS_INVALID_ACCDEC_VALUES	Invalid speed or acceleration value
182	SMC_MS_DIRECTION_NOT_APPLICABLE	Direction=shortest not applicable
183	SMC_MS_AXIS_IN_ERRORSTOP	The drive is in an error stopped state. Unable to execute stop
184	SMC_BLOCKING_MC_STOP_WASNT_CALLED	Not call through Execute=TRUE to block axis instance MC_Stop. Must call MC_Stop (Execute=FALSE)
185	SMC_MS_AXIS_ALREADY_STOPPING	Unable to interrupt ongoing MC_ Stop
200	SMC_UNKNOWN_TASK_INTERVAL	Unable to determine task interval for bus task
201	SMC_MA_INVALID_VELACC_VALUES	Invalid speed or acceleration value
202	SMC_MA_INVALID_DIRECTION	Direction error

Error code	Reasons	Description
226	SMC_MR_INVALID_VELACC_VALUES	Invalid speed or acceleration value
227	SMC_MR_INVALID_DIRECTION	Direction error
251	SMC_MAD_INVALID_VELACC_VALUES	Invalid speed or acceleration value
252	SMC_MAD_INVALID_DIRECTION	Direction error
276	SMC_MSI_INVALID_VELACC_VALUES	Invalid speed or acceleration value
277	SMC_MSI_INVALID_DIRECTION	Direction error
278	SMC_MSI_INVALID_EXECUTION_ORDER	Except interruption, do not call main mvtFB after SMC_MoveSupplicated
300	SMC_LOGICAL_NO_REAL_AXIS	No longer in use, for compatibility only
301	SMC_MV_INVALID_ACCDEC_VALUES	Invalid speed or acceleration value
302	SMC_MV_DIRECTION_NOT_APPLICABLE	Direction=shortest/fastest not applicable
325	SMC_PP_ARRAYSIZE	Wrong array size
326	SMC_PP_STEP0MS	Step time =t#0s
350	SMC_VP_ARRAYSIZE	Wrong array size
351	SMC_VP_STEP0MS	Step time =t#0s
375	SMC_AP_ARRAYSIZE	Wrong array size
376	SMC_AP_STEP0MS	Step time =t#0s
400	SMC_TP_TRIGGEROCCUPIED	Trigger activated
401	SMC_TP_COULDNT_SET_WINDOW	DriveInterface not support window function
402	SMC_TP_COMM_ERROR	Communication error
410	SMC_AT_TRIGGERNOTOCCUPIED	Trigger unassigned
426	SMC_MCR_INVALID_VELACC_VALUES	Invalid speed or acceleration value
427	SMC_MCR_INVALID_DIRECTION	Invalid direction
451	SMC_MCA_INVALID_VELACC_VALUES	Invalid speed or acceleration value
452	SMC_MCA_INVALID_DIRECTION	Invalid direction
453	SMC_MCA_DIRECTION_NOT_APPLICABLE	Direction=fastest, not applicable
475	SMC_SDL_INVALID_AXIS_STATE	SMC_ChangeDynamicLimits can only be called in a static state. or power_off
476	SMC_SDL_INVALID_VELACC_VALUES	Invalid speed, acceleration, deceleration, or jerk
600	SMC_CR_NO_TAPPETS_IN_CAM	Cam not set tappet point
601	SMC_CR_TOO_MANY_TAPPETS	tappets ID exceeded MAX_NUM_TAPPETS
602	SMC_CR_MORE_THAN_32_ACCESSES	a CAM_REF over 32 access times
625	SMC_CI_NO_CAM_SELECTED	not select cam
626	SMC_CI_MASTER_OUT_OF_SCALE	Spindle out of valid range
627	SMC_CI_RAMPIN_NEEDS_VELACC_VALUES	Must be ramp_ Must specify the velocity and acceleration values for ramp_in function
628	SMC_CI_SCALING_INCORRECT	Scale the variable fEditor/TableMasterMin/Max incorrect
629	SMC_CI_TOO_MANY_TAPPETS_PER_CYCLE	Too many tappets are active in one cycle
640	SMC_CB_NOT_IMPLEMENTED	Function block of given cam format is not implemented
675	SMC_GI_RATIO_DENOM	RatioDenominator = 0
676	SMC_GI_INVALID_ACC	Invalid acceleration
677	SMC_GI_INVALID_DEC	Invalid deceleration
678	SMC_GI_MASTER_REGULATOR_CHANGED	The status of the main device 'enabled/disabled' has been changed without permission
679	SMC_GI_INVALID_JECK	Jerk invalid
725	SMC_PH_INVALID_VELACCDEC	Invalid speed and acceleration/deceleration values
726	SMC_PH_ROTARYAXIS_PERIOD0	fPositionPeriod=0 modulus axis
750	SMC_NO_CAM_REF_TYPE	The given cam type is not MC_CAM_REF
751	SMC CAM TABLE DOES NOT COVER MASTER SCALE	The curve data does not include the main regions xStart and xEnd in CamTable
752	SMC_CAM_TABLE_EMPTY_MASTER_RANGE	The main range of the Cam data table is

Error code	Reasons	Description
		empty
753	SMC_CAM_TABLE_INVALID_MASTER_MINMAX	Cam data master has invalid maximum and minimum values
754	SMC_CAM_TABLE_INVALID_SLAVE_MINMAX	Cam data slave device has invalid maximum and minimum values
775	SMC_GIP_MASTER_DIRECTION_CHANGE	During the slave axis coupling, the spindle changed its direction of rotation
776	SMC_GIP_SLAVE_REVERSAL_CANNOT_BE_AVOIDED	Input AvoidReversal has been set, but it cannot be avoided from slave reversing
777	SMC_GIP_AVOID_REVERSAL_FOR_FINITE_AXIS	Input AvoidReversal cannot be limited slave axis setting
778	SMC GIP MASTERSTARTDISTANCE MUST BE ZERO BUFFERED	If BufferMode hasn't been interrupted, then MasterStartInstance cannot be positive
779	SMC_GIP_CANNOT_START_SYNC	Unable to start synchronization. This situation may occur when GearInPos is commanded to act as a buffer or hybrid motion, and the main controller is in a stationary state when activated
800	SMC_BC_BL_TOO_BIG	Gear return ratio (fBacklash) too large (>position periode/2)
825	SMC_QPROF_DIVERGES	Internal error: quadratic trajectory calculation failed
826	SMC_QPROF_INVALID_PARAMETER	Internal error: quadratic trajectory calculation failed
827	SMC_QPROF_NO_RESULT	Internal error: quadratic trajectory calculation failed
828	SMC_QPROF_INVALID_NEW_LBD	Internal error: quadratic trajectory calculation failed
829	SMC_QPROF_BAD_NEGOTIATION	Internal error: quadratic trajectory calculation failed
830	SMC_QPROF_INVALID_INTERVAL	Internal error: quadratic trajectory calculation failed
831	SMC_QPROF_NOT_ENOUGH_PHASES	Internal error: quadratic trajectory calculation failed
832	SMC_TG_INTERNAL_ERROR	Internal error: quadratic trajectory calculation failed
850	SMC_SRT_NOT_STANDSTALL_OR_POWEROFF	Allow action only when stationary or powered off
851	SMC_SRT_INVALID_RAMPTYPE	Invalid slope type
852	SMC_SMT_NOT_STANDSTALL_OR_POWEROFF	Allow action only when stationary or powered off
853	SMC_SMT_INVALID_MOVEMENTTYPE_OR_POSITIONPERIOD	Invalid motion type or position period
854	SMC_SMT_AXIS_NOT_VIRTUAL	Function block only applies to virtual axes
1000	SMC_NO_LICENSE	Target not licensed by CNC
1001	SMC_INT_VEL_ZERO	Unable to process path because set speed=0
1002	SMC_INT_NO_STOP_AT_END	The end speed of the last object in the path <>0
1003	SMC_INT_DATA_UNDERRUN	Warning: The geographic information list was processed in DataIn, but did not reach the end of the list. Reason: Unable to set EndOfList SMC_Interpolator of queue in DataIn faster than path generation function blocks
1004	SMC_INT_VEL_NONZERO_AT_STOP	speed when stop>0
1005	SMC_INT_TOO_MANY_RECURSIONS	SMC_Interpolator excessive recursion. Soft

Error code	Reasons	Description
		motion error
1006	SMC_INT_NO_CHECKVELOCITIES	SMC_CheckVelocity is not the last processed function block, it accessed data outside the queue through poqDataIn
1007	SMC_INT_PATH_EXCEEDED	Internal or numerical error
1008	SMC_INT_VEL_ACC_DEC_ZERO	Speed and acceleration/deceleration are zero or too low
1009	SMC_INT_DWIPOTIME_ZERO	Use dwIpoTime=0 to call FB
1010	SMC_INT_JERK_NONPOSITIVE	Jerk invalid, as Jerk must be positive
1011	SMC_INT_QPROF_DIVERGES	Internal error. The calculation of quadratic velocity distribution does not converge
1012	SMC_INT_INVLALID_VELOCITY_MODE	Invalid speed mode
1013	SMC_INT_TOO_MANY_AXES_INTERPOLATED	The number of inserted axes exceeds the allowed number of axes. You are using a restricted version
1014	SMC_INT_DEGENERATE_SEGMENT	This segment is numerically degenerate: it is very short and located at the end of the queue. It should be ignored
1015	SMC_HIGH_CURVATURE_SPLINE	The calculation of interpolation points failed because the curvature of the spline curve is too high. Try changing the path to avoid sharp corners
1050	SMC_INT2DIR_BUFFER_TOO_SMALL	Warning: The poqDataIn of the created output queue is too small, cannot guarantee compliance with stopping
1051	SMC_INT2DIR_PATH_FITS_NOT_IN_QUEUE	The path is not completely in the queue
1070	SMC_XINT_INVALID_DIRECTION	Direction input has invalid value
1071	SMC_XINT_NOINTERSECTION	Unable to determine the position of the given x position on the CNC path
1080	SMC_WAR_INT_OUTQUEUE_TOO_SMALL	Warning: The poqDataIn of the created output queue is too small, cannot guarantee compliance with station
1081	SMC_WAR_END_VELOCITIES_INCORRECT	Warning: Inconsistent final speed
1100	SMC_CV_ACC_DEC_VEL_NONPOSITIVE	Speed and acceleration/deceleration values: non positive
1120	SMC_CA_INVALID_ACCDEC_VALUES	fGapVelocity/fGapAcceleration/FGAPDegregation values: non positive
1130	SMC TOK COMPLETE TOKEN AT END OF INPUT	Input has been exhausted, but there are still unfinished marks
1131	SMC TOK NOT A VALID TOKEN	Input does not match any token type
1132	SMC_TOK_MUBILITY_INPUT	Ambiguous input, there may be multiple token types
1133	SMC_TOK_STRING_TOO_LONG_FOR_TOKEN	The string is too long to store in the token. (String text, variable name, or identifier too long)
1134	SMC_TOK_INVALID_NUMLIT	Invalid numeric text
1150	SMC_PRS_FUNC_DECL_TOO_MORY_ARGMENTS	Too few parameters provided for the function in G code
1151	SMC_PRS_FUNC_DECL_TOO_MANY_ARGMENTS	Too many parameters provided for the function in G code
1152	SMC_PRS_FUNC_DECL_WRONG_ARGUMENT_TYPE	Check the parameter types that match the function declaration. Return error
1153	SMC_PRS_LOCAL_VAR_NOT_FOUND	Cannot find local variable on the stack
1154	SMC_PRS_INVALID_STRING	Unable to read string from token
1155	SMC_PRS_TOO_MANY_CLOSING_BRACKETS	More closing parentheses than opening parentheses
1156	SMC_PRS_TOO_MANY_OPENING_BRACKETS	More start parentheses than end parentheses

Error code	Reasons	Description
1157	SMC_PRS_NO_SUCH_INFIX_OPERATOR	Unable to find infix operator
1158	SMC_PRS_NO_SUCH_PREFIX_OPERATOR	Unable to find prefix operator
1159	SMC_PRS_OPERATOR_INVALID_PRECEDENCE	Obtain two operators with equal priority but unequal associativity
1160	SMC_PRS_NOT_A_TERM	Expected validity period
1161	SMC_PRS_EXPRESSION_INVALID_SEQUENCE	Invalid tag sequence found in expression
1162	SMC_PRS_TOO_MANY_TERMS	Obtained more terms than expected
1163	SMC_PRS_STACK_OVERFLOW	Unable to parse expression because the stack is too small
1164	SMC_PRS_VAR_NAME_READY_USED	The name of a subroutine parameter or variable is already used for other subroutine parameters or parameters
1165	SMC_PRS_INCOMPLETE_SENTENCE_IN_TOKEN_QUEUE	The input token queue of the g-code parser does not contain a complete g-code statement
1167	SMC_PRS_TOO_MANY_SUBPROGRAMS	Cannot store subroutine declaration because symbol table capacity has been exceeded
1168	SMC_PRS_TOO_MANY_SUBPROGRAM_PARAMETERS	The maximum number of subroutine parameters has been exceeded
1169	SMC_PRS_SUBPROGRAM_LOOKUP_FAILED	Failed to find subroutine declaration. Perhaps the subroutine name is incorrect, or the search path for the subroutine is incomplete
1170	SMC_PRS_VAR_NOT_FOUND	Variable not found in symbol table
1171	SMC_PRS_TOKEN_TYPE_UNKNOWN	Unknown token type
1172	SMC_PRS_GOT_NO_TERM	No terminology after G-address
1173	SMC_PRS_INVALID_VAR_TYPE	Unknown variable type (not LREAL, BOOL, or string)
1174	SMC_PRS_UNEXPECTED_TOKEN	Different types of tokens (such as operators or identifiers) are required here
1175	SMC_PRS_ID3_EXPECTED	An identifier with a length of 3 or longer is required here
1176	SMC_PRS_ID_TOO_LONG	Identifier too long (over 80 characters)
1177	SMC_PRS_GADDRESS_EXPECTED	G address is required here (such as "G", "F", "X")
1178	SMC_PRS_NWORD_EXPECTED	An N character is required here (such as' N10 ')
1179	SMC_PRS_NWORD_INVALID_SENTURE_NUMBER	The sentence number is not within the range of [0,..., 16 # FFFFFFFF]
1180	SMC_PRS_NWORD_SENTENCE_NUMBER_NO_NUMBER_LITERAL	The sentence number must be a numerical literal. (For example, expressions are not allowed.)
1181	SMC_PRS_USE_OF_RESERVED_KEYWORD	Identifier is a reserved keyword and cannot be used here
1182	SMC_PRS_SUBPROGRAMS_SIGNATURE_MISMATCH	A different signature is used or the subroutine declaration has been read error
1183	SMC_PRS_INITIAL_VALUE_HAS_WRONG_TYPE	The initial value of this local variable has the wrong type
1184	SMC_PRS_TOO_MANY_LOCAL_VARIABLES	The maximum number of subroutine parameters has been exceeded
1185	SMC_PRS_SUBPROG_MUST_BE_FIRST_SENTENCE	The subroutine declaration must be the first sentence in the g code file
1186	SMC_PRS_ONLY_ONE_subgram_PER_FILE_ALLOWED	Only one subroutine is allowed in the G code file.
1187	SMC_PRS_LET_AFTER_REGULAR_SENTENCE	LET declaration is not allowed after regular G-code statements
1188	SMC_PRS_UNMATCHED_END_SUBPROGRAM	END_SUBPROGRAM. statement does not

Error code	Reasons	Description
		match subroutine declaration
1189	SMC_PRS_UNEXPECTED_TOKENS_AFTER_SUBPROGRAM	After END_SUBPROGRAM., no more tokens allowed
1190	SMC_PRS_MISSING_END_SUBPROGRAM	The subroutine does not terminat with END_SUBPROGRAM.
1200	SMC_DEC_ACC_TOO_LITTLE	Acceleration value not allowed
1201	SMC_DEC_RET_TOO_LITTLE	Deceleration value not allowed
1202	SMC_DEC_OUTQUEUE_RAN_EMPTY	Insufficient data. Queue read and empty
1203	SMC_DEC_JUMP_TO_UNKNOWN_LINE	Unable to perform jump to line because the line number is unknown
1204	SMC_DEC_INVALID_SYNTAX	Invalid syntax
1205	SMC_DEC_3DMODE_OBJECT_NOT_SUPPORTED	Object not supported in 3D mode
1206	SMC_DEC_NEGATIVE_PERIOD	Invalid negative value as additional axis period
1207	SMC_DEC_DIMENSIONS_EXCLUSIVE_AU	Interpolation is not performed on both axes A and U. PA and PU are mutually exclusive
1208	SMC_DEC_DIMENSIONS_EXCLUSIVE_BV	Interpolation is not performed on both axes B and V. PB and PV are mutually exclusive
1209	SMC_DEC_DIMENSIONS_EXCLUSIVE_CW	Interpolation is not performed on both axes C and W. PC and PW are mutually exclusive
1210	SMC_DEC_DCS_NOT_ALL_OF_ABC_GIVEN	For G54/G55/G56, if the directional mode is not equal to SMC_ORI_CONVENTION.ADDAXES, then all A, B, C or none must be given
1211	SMC_DEC_DCS_2D_NOT_IN_XY_PLANE	The Z-axis of the decoder CS must be equal to the Z-axis of the machine CS
1212	SMC_DEC_CIRCLE_NON_UNIFORM_SCALING	Scaling the circle or ellipse that has twist command
1213	SMC_DEC_ROTATION_AFFECTS_SCALING	The new relative rotation of DCS (G55) will affect scaling
1214	SMC_DEC_DCS_NOT_ALL_OF_IJK_GIVEN	For G54/G55/G56, all I, J, K or none must be given
1250	SMC_IPR_LOCAL_VAR_UNKNOWN_TYPE	Unable to read local variable due to invalid type
1251	SMC_IPR_LVALUE_WRONG_TYPE	The variable type that should be written is incorrect
1252	SMC_IPR_EVAL_STACK_OVERFLOW	Unable to evaluate expression because the evaluation stack is too small
1253	SMC_IPR_NOT_A_NUMBER	The numerical term becomes NaN
1254	SMC_IPR_DIVISION_BY_ZERO	Divide by zero
1255	SMC_IPR_INVALID_SCALING_FACTORS	Invalid Scale Factor
1256	SMC_IPR_LVALUE_WRONG_TYPE	When the subroutine returns, the interpreter stack is empty. internal error
1257	SMC_IPR_INTERPRETER_STACK_OVERFLOW	The interpreter stack is too small. There are too many local variables in the G code or too many active subroutines
1258	SMC_IPR_INVALID_INTERPRETER_STACK_BUFFER	The given buffer of the interpreter stack is 0 or less than 1024 bytes
1259	SMC_IPR_BUFFER_TOO_SMALL	The buffer of the output queue is too small
1280	SMC_DNCCS_NO_DATA	There is no available data at all
1281	SMC_DNCCS_TOO_MANY_CALLSTACKS	There are too many call stacks between the current state of iObjNo and the interpreter
1282	SMC_DNCCS_INVALID_PROGRAM_INDEX	Invalid program index
1283	SMC_DNCCS_TOO_MANY_PROGRAM	Too many different programs
1284	SMC_DNCCS_WRONG_TASK	FB called the wrong task
1300	SMC_GCV_BUFFER_TOO_SMALL	buffer too small
1301	SMC_GCV_BUFFER_WRONG_TYPE	Wrong type of buffer element

Error code	Reasons	Description
1302	SMC_GCV_UNKNOWN_IPO_LINE	Unable to find the current row of the interpolator
1400	SMC_CNC_INTERNAL_ERROR	Internal error in CNC
1401	SMC_PATH_MAX_HPOINTS_EXCEEDED	The path element cannot be saved beyond H switching points of MAX_IPOSWITCHES. Adjust the switching point using different "O" values, or reduce the number of H-points for each path element
1410	SMC_TRC_G75_NOT_ALLOWED	G75 is not allowed during tool radius correction
1411	SMC_TRC_QUEUE_FULL_NON_CARTESIAN	The queue is full, but there are no other Cartesian elements
1412	SMC_TRC_SPLINE3D_5_NOT_SUPPORTED	Tool radius correction does not support motion type SPLINE3D_5
1414	SMC_TRC_PLANE_CHANGE_NOT_ALLOWED	Do not allow plane changes during tool radius correction
1450	SMC_NAV_MAX_SUBSPROGRAM_NESTING_EXCEEDED	The maximum nesting level for subroutine call has been exceeded
1451	SMC_NAV_RETURN_FROM_MAIN	Do not allow main G code programs to return statements
1452	SMC_NAV_SUBPROGRAM_DECLARATION_NOT_FOUND	The subroutine declaration was not found in the subroutine CNC file
1453	SMC_NAV_NOT_ENOUGH_SPACE_FOR_COMPLETE_SENTENCE	Unable to add the next G code statement to the output statement queue because there is not enough space
1500	SMC_NO_CNC_REF_TYPE	The given CNC program is not SMC CNC REF type
1501	SMC_NO_OUTQUEUE_TYPE	The given output queue is not SMC_ OutQueue type
1502	SMC_GEOINFO_BUFFER_MISALIGNED	Not using 4-byte aligned buffer section in pbyBuffer
1600	SMC_3D_MODE_NOT_SUPPORTED	Function block only applies to 2D paths
1700	SMC_SAA_SMOOTHAREA_TOO_LARGE	Smooth range too large
1701	SMC_SAA_SP_INVALID_INPUT	Invalid input dSmoothingPart]0..1]
1800	SMC_SA_QUEUE_NOT_IN_BUFFER	SMC SegmentAnalyzer detected that the out of queue buffer is full but not complete. The function block can only run when OutQueue is fully suitable for the buffer
1801	SMC_SA_QUEUE_CHANGED_DURING_OP	When the function block operates on it, the out of queue buffer has changed
1820	SMC_OS_INVALID_PARAMETER	Invalid input value inside dSplittingParameter
1830	SMC_BSSP_IPO_NOT_ACTIVE	Unable to save location. Inserter not active
1831	SMC_BS_SAVEDPOS_NOT_REACHED	No saved location has been found so far. It may have taken a different path
1832	SMC_BS_NO_POS_STORED	The structure passed in ePos does not contain a saved location. SMC BlockSearchSavePos did not executed or executed in an incorrect manner
1900	SMC_INVALID_FEATURE_FLAG	The functional symbol must be within the range of {1,..., 31}
1901	SMC_SMB_HFUN_NOT_SUPPORTED	Function block does not support the h function
1902	SMC_SMB_ONLY_3DMODE	Function block only works in 3D mode
1903	SMC_SMB_ERROR_COMPUTING_SPLINE	Calculate the internal error of a spline curve
1910	SMC_SMM_INVALID_PARAM_NUMBER	WadAdditionalParameter too large
1950	SMC_INVALID_PARAMETER	One of the input values is invalid

Error code	Reasons	Description
1951	SMC_BLENDING_NOT_SUPPORTED_BY_PREVIOUS_MOVEMENT	The previous move does not support mixing
1952	SMC_BUFFERED_NOT_SUPPORTED_BY_PREVIOUS_MOVEMENT	The previous move does not support buffered moves
1953	SMC_INHERIT_OWNER_ACTIVE_MOVEMENT_NOT_CALLED	Cannot inherit owner because activity move has not been called in this loop. internal error
1954	SMC_MOVING_WITHOUT_ACTIVE_MOVEMENT	The axis is moving, but there is no active move function block. Losing subsequent movements in one unstoppable movement after another
1955	SMC_BUFFER_MODE_NOT_SUPPORTED	Command movement does not support configured buffering mode
1956	SMC_ERROR_IN_A_PREVIOUS_MOVEMENT	An error occurred during the previous move
1957	SMC_MORE_THAN_ONE_MOVEMENT_PER_INSTANCE	Only one busy buffer/mixed move allowed per FB instance
2000	SMC_RNCF_FILE_DOESNT_EXIST	file does not exist
2001	SMC_RNCF_NO_BUFFER	Unallocated buffer
2002	SMC_RNCF_BUFFER_TOO_SMALL	buffer too small
2003	SMC_RNCF_DATA_UNDERRUN	Insufficient data. Buffer read, it is empty
2004	SMC_RNCF_VAR_COULDNT_BE_REPLACED	Unable to replace placeholder variable
2005	SMC_RNCF_NOT_VARLIST	Input pvl does not point to SMC_ VARLIST Object
2006	SMC_RNCF_NO_STRINGBUFFER	Input pStringBuffer does not be used or pointed to SMC_StringBuffer type variables
2007	SMC_RNCF_STRINGBUFFER_OVERRUN	In CNC programs, more different strings are used than the string buffer can accommodate
2008	SMC_RNCF_SUBPROGRAM_FILE_NOT_FOUND	Unable to find file for subroutine
2050	SMC_RNCQ_FILE_DOESNT_EXIST	could not open file
2051	SMC_RNCQ_NO_BUFFER	Undefined buffer
2052	SMC_RNCQ_BUFFER_TOO_SMALL	buffer too small
2053	SMC_RNCQ_UNEXPECTED_EOF	Unexpected end of file
2100	SMC_ADL_FILE_CANNOT_BE_OPENED	could not open file
2101	SMC_ADL_BUFFER_OVERRUN	Buffer overflow. WriteToFile must be called more frequently
2200	SMC_RCAM_FILE_DOESNT_EXIST	could not open file
2201	SMC_RCAM_TOO_MUCH_DATA	The saved cam is too large
2202	SMC_RCAM_WRONG_COMPILE_TYPE	Incorrect compilation mode
2203	SMC_RCAM_WRONG_VERSION	The version of the file is incorrect
2204	SMC_RCAM_UNEXPECTED_EOF	Unexpected end of file
3001	SMC_WDPF_CHANNEL_OCCUPIED	This error is outdated and reserved for compatibility only. SMC WDPF TIMEOUT PREPARING LIST
3002	SMC_WDPF_CAN_CREATE_FILE	This error is outdated and reserved for compatibility only. could not create file
3003	SMC_WDPF_ERROR_WHEN_READING_PARAMS	This error is outdated and reserved for compatibility only. Error reading parameters
3004	SMC_WDPF_TIMEOUT_PREPARING_LIST	This error is outdated and reserved for compatibility only. Timed out when preparing parameter list
5000	SMC_ENC_DENOM_ZERO	The conversion coefficient of encoder reference dwRatioTechUnitsDenom named is 0
5001	SMC_ENC_AXISUSEDBYOTHERFB	Attempting to handle other modules moving on the encoder axis
5002	SMC_ENC_FILTER_DEPTH_INVALID	Invalid filter depth
6000	SMC_INTERNAL_ERROR_INIT_MOVEMENT	Internal error when initializing new move

Error code	Reasons	Description
6001	SMC_INVALID_AXIS_TYPE	Invalid axis type, not finite or modular (internal error)
10000	SMC_PCCQ_POINTBUFFERTOOSMALL	Buffer pBuffer too small
10001	SMC_PCCQ_INPUTBUFFERFULLBUTNOTFINALIZED	The function block must be applied to a path that is completely suitable for the buffer
11000	SMC_AXIS_GROUP_WRONG_STATE	The axis group is in an error state for the requested operation
11001	SMC_AXIS_GROUP_TOO_MANY_AXES	Axis group has been added with more than the maximum allowed number of axes
11002	SMC_AXIS_GROUP_INVALID_DYNLIMITS	Invalid dynamic limit for a single axis (fSWMaxVelocity/ acceleration/deceleration/jerk)
11004	SMC_AXIS_GROUP_INVALID_COORD_SYSTEM	The given coordinate system is invalid for the requested operation
11005	SMC_AXIS_GROUP_SINGLE_AXISS_ERROR	The axis of the axis group is in the wrong state
11006	SMC_MOVE_INVALID_BUFFER_MODE	The given buffer mode is not supported
11007	SMC_MOVE_INVALID_DYNAMIC_FACTOR	The dynamic coefficient is not within the range of [0.. 1]
11008	SMC_MOVE_INVALID_DYNAMICS	Invalid dynamic restrictions for movement
11009	SMC_AXIS_GROUP_AXIS_NOT_PART_OF_GROUP	The given axis is not part of an axis group
11010	SMC_AXIS_GROUP_NOT_SUPPORTED	The requested operation is not supported
11011	SMC_AXIS_GROUP_KINEMATICS_NOT_SET	No kinematic configurations have been configured yet
11012	SMC_AXIS_GROUP_WRONG_NUMBER_OF_AXES	The number of axes is not equal to the number of axes required for motion transformation
11013	SMC AXIS GROUP INTERRUPTED BY SINGLE AXIS	Coordinated movement interrupted by single axis movement
11014	SMC_AXIS_GROUP_FOLLOW_SETVALUES	Error tracking calculated settings
11015	SMC_AXIS_GROUP_TOO_MANY_dependency	One axis group cannot rely on multiple axis groups
11016	SMC_AXIS_GROUP_MUTUAL_DEPENDENCY	Axis group A may not depend on another axis group that depends on A
11017	SMC_AXIS_GROUP_DEPENDENCY_IN_DIFFERENT_TASK	The dependent axis group must be executed in the same task
11018	SMC_AXIS_GROUP_AXISS_IN_DIFFERENT_TASK	Axes belonging to an axis group must be executed in the same task
11019	SMC_AXIS_GROUP_PCS_STILL_IN_USE	When PCS is still being used by buffered motion commands, the function block undergoes a second activation
11020	SMC AXIS GROUP CMD ABORTED DUE TO ERROR	Error in previous motion command
11021	SMC_AXIS_GROUP_INVALID_PARAMETER	Invalid parameters for management function block
11022	SMC_AXIS_GROUP_UNSUPPORTED_RAMPTYPE	One axis in the group uses an unsupported ramp type. Only supports trapezoidal and quadratic types
11023	SMC_MOVE_INVALID_TRANSITION_PARAMETER	Invalid conversion parameter
11024	SMC_AXIS_GROUP_INTERNAL_ERROR	Internal error in the state machine of the axis group
11025	SMC_AXIS_GROUP_CPTR_CANNOT_FOLLOW	CP tracking: unable to maintain path; Attempting to reduce dynamics on the path and/or entering CP tracking
11026	SMC_AXIS_GROUP_CONTINUE_WRONG_POSITION	The current position does not allow continuation
11027	SMC_AXIS_GROUP_CONTINUE_BUFFER_TOO_SM	The buffer in ContinueData is too small;

Error code	Reasons	Description
	ALL	Using large external move queue buffer
11028	SMC_AXIS_GROUP_CONTINUE_WRONG_CHECKSUM	The parity and error in continue data, continue data is not written by MC_GroupInterrupt, but is modified later
11029	SMC_AXIS_GROUP_IDLE_WAIT_AXES_MOVING	Command SMC_GroupWait, simultaneously, the axis moves independently of the axis group (single axis movement or similar)
11030	SMC_AXIS_INVERSE_TRAFO_EXCEEDING_POSLIMITS	The inverse transformation will cause the axis value to exceed the configured limit
11031	SMC_AXIS_GROUP_CANNOT_ADD_SAME_AXIS	Prohibit adding the same axis to the axis group twice
11032	SMC_AXIS_GROUP_TRANSFORMATION_NOT_SET	Do not set conversion
11033	SMC_AXIS_GROUP_NON_RESUMABLE_ERROR	Unable to recover path after current axis group error
11034	SMC_AXIS_GROUP_CONTINUE_DATA_NOT_WRITTEN	The continuation data for MC_GroupContinue was not written correctly. (MC GroupInterrupt not called/unrecoverable axis group error)
11100	SMC_KERNEL_PTP_INTERNAL_ERROR	Internal errors in the kernel
11101	SMC_KERNEL_PTP_INVALID_TASKCYCLETIME	Task cycle time is not positive
12000	SMC_TRAFO_INVALID_PARAMETERS	The converted parameter value is invalid
12001	SMC_TRAFO_INVALID_CONSTELLATION	The requested location is outside the converted work area
12002	SMC_TRAFO_INVALID_COUNPING	Tools cannot work together with the positioning components of the machine. Alternatively, the positioning component is not implemented ISMPocationKinetics2, or this tool is unable to process directional images for positioning
12003	SMC_TRAFO_NOT_INITIALIZED	FB realized |ISMkineticWithinitialization|, but not initialized yet
12004	SMC_TRAFO_NO_TOOL_WITH_OFFSET_ALLOWED	Special errors in the positioning section Kin_4Axis_Palletzer, it realized ISMPositionKinetics Offset2. But in reality, it's not possible to handle offsets <>0
12005	SMC AXIS GROUP TOOL OFFSET INCOMPATIBLE_WITH_KINEMATICS	The configured tool offset is not compatible with the used kinematics
12100	SMC_CP_CACHE_FULL	CP planner's cache size is too small
12101	SMC_CP_EVAL_ERROR	Calculation of position on path element failed (internal error)
12102	SMC_CP_NON_CONTINUABLE_STATE	Reached non continuous state (internal error)
12103	SMC_CP_MAX_LENGTH_EXCEEDED	Exceeded maximum trajectory length (internal error)
12104	SMC_CP_ACCELERATION_TOO_HIGH	Path acceleration too high (invalid state, internal error)
12105	SMC_CP_MAX_ITERATIONS_EXCEEDEDSMC_CP_MAX_ITERATIONS_EXCEEDED	Exceeded maximum number of iterations (internal error)
12106	SMC_CP_NO_TRAJECTORY	Unable to calculate trajectory (internal error)
12107	SMC_CP_OUT_QUEUE_FULL	Output queue full (internal error)
12108	SMC_CP_QUEUE_UNDERRUN	Insufficient queue operation: there are no remaining elements in the queue
12109	SMC_CP_INVALID_QUEUE	Invalid queue (invalid size or pointer)
12110	SMC_CP_Blend_INTERNAL_ERROR	Internal error while mixing two CP movements
12111	SMC_CP_CIRCLE_COLLINEAR_POINTS	The three points defined a circle are collinear
12112	SMC_CP_CIRCLE_CENTER_NOT_ON_BISECTOR	The center point is not on the vertical

Error code	Reasons	Description
		bisector of the starting and ending points
12113	SMC_CP_CIRCLE_RADIUS_ZERO	Zero radius
12114	SMC_CP_CIRCLE_DISTANCE_LARGER_THAN_DIAMETER	The distance between the start and end points is greater than the diameter (when trying to create a circle using SMC_Circ_Mode.radius)
12115	SMC_CP_CIRCLE_START_AND_ENDPOINT_EQUAL	Equal starting and ending points (SMC_Circ_Mode.Radius)
12116	SMC_CP_BLENDING_DEGENERATE_SPLINE	Mixed spline degradation (too short or irregular)
12117	SMC_CP_ELEMENT_TOO_SHORT	Unable to create path element because it is too short
12118	SMC_CP_CAN_NOT_CUT_PATH	Unable to cut path to abort movement (internal error)
12119	SMC_CP_INVALID_ANGULAR_VEL_ACC	The given angular velocity or acceleration is invalid (quaternions with non zero real parts, internal error)
12120	SMC_CP_INVALID_ORIENTATION	Invalid given direction (non orthogonal matrix or non identity quaternion, internal error)
12121	SMC_CP_TIME_BUDGET_EXCEEDED	Exceeded the calculated given time budget (internal error)
12122	SMC_AXIS_GROUP_AXIS_LIMIT_VIOLATED	Axis restrictions have been violated. If the CP moves too close to the singularity or the path has very high curvature, this situation may occur
12125	SMC_CP_CONFIGS_DIFFER	The kinematic configuration of the starting position is different from the ending position. CP motion will pass through singularities
12126	SMC_CP_BUS_TASK_NOT_CALLED	This error is outdated and reserved for compatibility only. If the command CPhalt/-stop is used, some information must be transferred to the bus task. We expect this to occur before the next cycle of the slow task
12127	SMC_CP_NO_ROOT_IN_INTERVAL_FOUND	Unable to find the stop trajectory in the interval. (Internal error)
12128	SMC_CP_KIN_DOES_NOT_SUPPORT_AXIS_ORIENTATION_IPO	Kinematics does not support continuous path movement direction mode 'axis'
12129	SMC_CP_AXIS_ORIENTATION_IPO_NOT_SUPPORTED_FOR_CPTR	CP movement with dynamic PCS (tracking) does not support axis direction interpolation mode
12130	SMC_CP_INVALID_PATH_ELEM	Invalid path element created (internal error)
12131	SMC_CP_TRANSITION_NOT_SMOOTH	The conversion between two path elements is not G2 continuous and did not stop (internal error)
12132	SMC CP AXIS ORIENTATION IPO CONFIG DIFFERS	The axis direction interpolation mode was used, but there is singularity in positional kinematics between the start and end positions
12133	SMC_CP_AXIS_ORIENTATION_IPO_OFFSET2_NOT_IMPLEMENTED	Position kinematics has configurations but does not implement interfacesIsmPositionKinetics Offset2. In this case, axis direction interpolation is not possible
12134	SMC_CP_ROTARY_AXIS_PERIOD_MISMATCH	The target position of the rotation axis will not be reached within the command cycle. (For example, the instruction target position

Error code	Reasons	Description
		is -170°, but it will reach the target at position 190°.) This means that the selected target position interpolation is not compatible with the instruction position
12135	SMC_CP_ROTARY_AXIS_RANGE_VIOLATION	During CP movement, the rotation axis violated the allowed axis range. CP movement is not within the work area
12136	SMC_CP_COMPUTE_TARGET_DISCONTINUITY	Due to the discontinuity in the end position (internal error), the target trajectory cannot be calculated
12137	SMC_CP_TRACTION_NOT_SMOOTH	The trajectory is not smooth, and the phase end state is not equal to the current state (internal error)
12138	SMC CP ERROR CREATING PARAM TRANSFORM	Unable to create parameter conversion (internal error)
12139	SMC_CP_PTP_DATA_NUMBER_OF_AXES	Error generating path invariant PTP element (internal error)
12140	SMC_CP_NEGATIVE_PATH_VELOCITY	Error calculating new trajectory: Path velocity after stage 1 (positive acceleration of length dTau1) is negative (internal error)
12141	SMC_CP_TRANSITIONING_FROM_SINGLE_AXIS_MOVEMENT_NOT_SUPPORTED	Transition from single axis movement to continuous path movement is not supported
12142	SMC_CP_PLANNER_NO_PROGRESS	CP-planner has not made any progress. This is an internal error that can trigger very sharp corners (such as mixing short elements or small angular distances)
12143	SMC_CP_INTERNAL_EVAL_CACHE_ERROR	There is an internal error in the evaluation cache of the CP planner

WUXI XINJE ELECTRIC CO., LTD.

Address: 816 Jianshe West Road, Binhu District, Wuxi City, Jiangsu Province, China

Tel: 0510-85134136 Fax: 0510-85111290

Website: www.xinje.com Email: sales@xinje.com