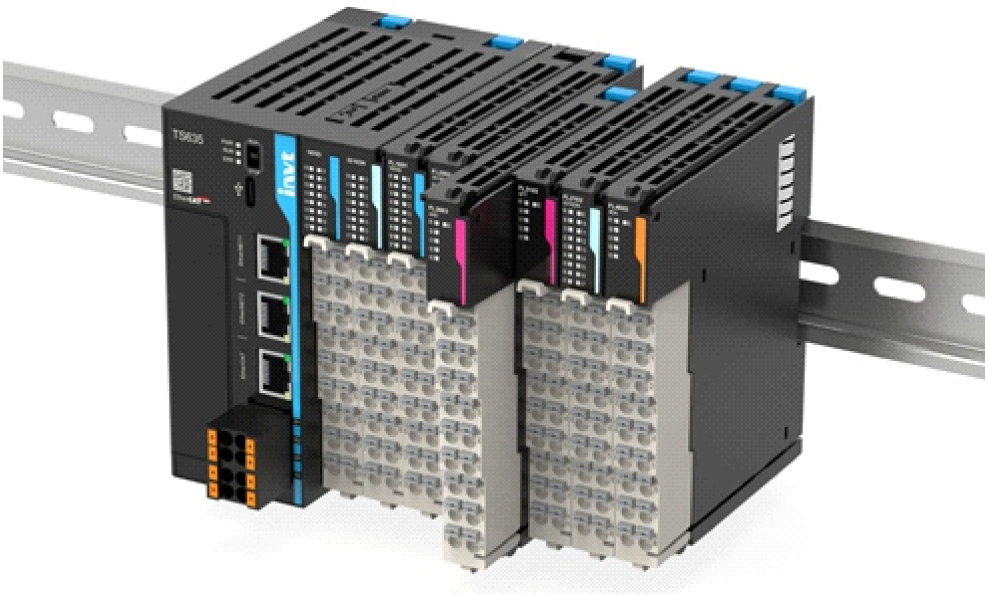
TS600 Series Programmable Logic Controller

Programming and Application Manual



SHENZHEN INVT ELECTRIC CO., LTD.

No.	Change description	Version	Change date
1	First release.	V1.0	March 2024
2	● Modified the description and diagram in section 3.4.2 Variable Attribute. ● Deleted section 3.4.4 Struct Binding to Soft Elements. ● Added section 3.7 Library Functions. ● Updated Table 3-16 _stru_IOT_CARD 4G IoT Card Information and added Table 3-17 _stru_LABEL_COM Tag Communication Information. ● Added section 3.10 Function Block and Function (FB/FC) ● Updated the titles of sections 6.3 and 6.5, and add RS232 interface definition in section 6.5. ● Updated section 6.8.3.2 Address Binding Table. ● Updated section 7.7.3.2 Address Binding Table. ● Added the description in section 8.5.1.2 Network State. ● Added the description in section 8.5.6 Service Data Object. ● Added the description in section 8.5.7 Online Debugging Function. ● Added the description in section 8.5.10 CANopen Slave Description. ● Added section 8.7 CAN Free Protocol. ● Added the description in section 9.2.2 Scanning Device. ● Added section 9.3.4.2 Associated Elements. ● Added section 15.2.2 Data Retention File Upload and Download ● Updated the model table in section 16.1.1 Summary of Expansion Module. ● Updated section 16.1.4.2 Digital Output Module. ● Added section 16.1.4.5 Hybrid Module. ● Updated chapter 18 Application of Custom Variable Communication ● Updated section 19.3 Fault Code List. ● Modified section 20.2.1 Step of Generating PLC Application Upgrade Package ● Modified the description in section 20.2.2 SD Card Upgrading Steps ● Modified the description in section 21.7 Web Page Monitoring.	V1.1	August 2024
4	● Deleted the descriptions of double word operation and floating-point operation in sections 4.3 Operating Steps and 4.4 Use Case. ● Updated the description of section 6.1 Serial Communication Resources. ● Added section 6.8.3.4 Read SN code and the Model. ● Updated section 7.1 Ethernet Communication Resources. ● Updated section 7.3.1 Factory Default IP Address. ● Added section 7.7.3.4 Read SN Code and the Model. ● Updated section 7.8.3.1 Modify Ethernet IP.	V1.2	February 2025

No.	Change description	Version	Change date
	● Updated section 9.2.2 Scanning Device. ● Updated section 9.2.3 Master Setting. ● Updated section 16.1 TS600 Series Host Local Expansion Module. ● Updated section 19.3 Fault Code List. ● Updated section 20 Firmware Burning and Upgrade.		

Preface

TS600 is a new-generation compact PLC series product independently developed by INVT. With powerful motion control and distributed I/O control functions, it supports EtherCAT fieldbus communication, can conveniently realize user process movement through custom variables, axis control configuration and graphic block instructions, and can realize multi-level network communication via RS485, CAN, EtherCAT interfaces.

TS600 series full-scene compact controllers are available in multiple models to meet various needs of users for compact automation equipment, and are suitable for compact design, multi-axis motion control, temperature control, communication networking and other scenarios.

This manual mainly introduces the basic knowledge of PLC programming, quick start, communication, motion control, the usage of high-speed counter of TS600 series, etc. The software interface in the manual may vary due to different versions.

This manual applies to the following readers: Personnel with electrical professional knowledge (such as qualified electrical engineers or personnel with equivalent knowledge).

The users using this product for the first time should read this manual carefully first. If you have any technical problems, please contact our customer service.

This manual is not delivered along with the product. To obtain an electronic version of the PDF file, you can:

Log in to INVT's official website (www.invt.com) → Service and Support → Data Download → Search for keywords and download.

List of related manuals

Name	Ordering Code
TS600 Series Programmable Logic Controller User Manual	66001-01167
TS600 Series Programmable Logic Controller Command Manual	66001-01290

Contents

Preface ..... **i**
Contents ..... **iii**
1 Overview **1**
1.1 Product Overview 1
1.2 One-stop Solutions 1
2 Quick Start **2**
2.1 Programming Software 2
2.1.1 Software Introduction 2
2.1.2 Computer Configuration Requirements 2
2.1.3 How to Get Our Software 2
2.1.4 Auto Station Pro Programming Software Installation Process 2
2.1.5 Auto Station Pro Running Interface 2
2.1.6 Programming Language 3
2.2 Communication Settings 3
2.2.1 Communication Connection Method 3
2.2.2 Ethernet Connection 3
2.3 Typical Steps of Programming for Users 4
2.4 Programming and Debugging 5
2.4.1 Sample Project Requirements 5
2.4.2 Start the Programming Environment 5
2.4.3 Establish a Project 5
2.4.4 Write a Control Program 6
2.4.5 Establish a Communication Connection 6
2.4.6 Compile and Download the Program 7
2.4.7 Start the Programmable Controller 9
2.5 Switch PLC Working Mode 9
2.6 Trace 9
2.6.1 Function Introduction 9
2.6.2 Operating Steps 9
2.7 Scan Period Setting 11
2.7.1 Time Setting 12
3 Basic Knowledge of Programming **13**
3.1 Summary 13
3.2 Soft Element 13
3.2.1 Bit Soft Element 13
3.2.2 Word Soft Element 14
3.2.3 Bit Operation of Word Elements 15
3.3 Variables 15
3.3.1 Custom Variables 15
3.3.2 Define Variables 15
3.3.3 Define Arrays 15
3.3.4 Define Struct 16
3.3.5 Bit Operation of Variables 16
3.3.6 How to Use Variables 16
3.4 Variable Binding Address 16
3.4.1 Overview 16
3.4.2 Variable Attribute 17
3.4.3 Bind Array Variables to Soft Elements 17
3.5 Use a Variable as Array Subscript 18
3.5.1 Rules of Usage 18
3.5.2 Basic Combination Types 18
3.5.3 Complex combination Type 18
3.5.4 Programming Instance 19

3.6 Graphic Block Instruction 21
3.6.1 Composition of Graphic Blocks 21
3.6.2 Graphic Block Instruction Programming 21
3.7 Library Functions 22
3.7.1 Example of Exporting an Addition Library 22
3.7.2 Example of Importing An Addition Library 24
3.7.3 Call Library Functions 25
3.8 System Variables 27
3.8.1 Overview 27
3.8.2 List of System Variables 27
3.8.3 _SYS_CAN CAN Interface Running Information 27
3.8.4 _SYS_COM Serial Port Running Information 28
3.8.5 _SYS_ECAT EtherCAT Running Status Information 30
3.8.6 _SYS_ETHERNET Ethernet Information 31
3.8.7 _SYS_INFO PLC Running Information 33
3.9 Timer 36
3.9.1 TON 36
3.9.2 TONR 37
3.9.3 TOF 37
3.9.4 TMON 38
3.10 Function Block and Function (FB/FC) 38
3.10.1 Function Block (FB) 39
3.10.2 Function (FC) 42
4 User C language 45
4.1 Overview 45
4.2 Instruction Format 45
4.3 Operating Steps 45
4.4 Use Case 48
5 Programming Language 51
5.1 Ladder Diagram (LAD) 51
5.1.1 Concept of Ladder Diagram 51
5.1.2 Basic Programming Elements of Ladder Diagram 51
5.1.3 Energy Flow 52
5.2 Instruction List (IL) 52
5.3 Sequential Function Chart (SFC) 53
6 Serial Communication 54
6.1 Serial Communication Resources 54
6.1.1 Supported Serial Communication Protocols 54
6.1.2 Applicable Baud Rate 54
6.2 Link characteristics 54
6.3 RS485 Power Terminal Wiring 54
6.4 Serial Networking Connection 55
6.5 RS232 Interface 56
6.6 Freeport Communication Protocol 57
6.6.1 Brief Introduction 57
6.6.2 Setting of Freeport Parameters 57
6.7 Modbus-RTU/ASCII Master 59
6.7.1 Brief Introduction 59
6.7.2 Serial Port Setting 59
6.7.3 Master Setting 60
6.7.4 Master Configuration Table 61
6.8 Modbus-RTU/ASCII Slave 63
6.8.1 Brief Introduction 63
6.8.2 Slave Settings 63
6.8.3 Slave Related Information 64
6.9 Modbus-RTU Communication Application Example 66
6.9.1 Brief Introduction 66
6.9.2 Enable Master 66
6.9.3 Enable Slave 67

6.9.4 Example Phenomenon 68
7 Ethernet Communication........ 69
7.1 Ethernet Communication Resources 69
7.2 Ethernet Wiring Example 70
7.3 Ethernet IP Address 70
7.3.1 Factory Default IP Address 70
7.3.2 Modify IP Address in Upper Computer Interface 70
7.3.3 Modify IP address with System Variables 72
7.4 TCP Free protocol 73
7.5 UDP Free Protocol........ 73
7.6 Modbus TCP Master 73
7.6.1 Brief Introduction 73
7.6.2 Modbus TCP Master Settings........ 73
7.6.3 Modbus TCP Master Configuration Table 74
7.7 Modbus TCP Slave 76
7.7.1 Brief Introduction 76
7.7.2 Modbus TCP Slave Settings........ 76
7.7.3 Slave Related Information........ 77
7.8 Modbus TCP Communication Application Example 79
7.8.1 Brief Introduction 79
7.8.2 Modbus TCP Master Settings........ 79
7.8.3 Modbus-TCP Slave Settings........ 80
7.8.4 Example Phenomenon 81
8 CAN Communication........ 82
8.1 Overview........ 82
8.2 Hardware Interface 82
8.3 CAN Network 83
8.3.1 CAN Communication Networking 83
8.3.2 Communication Distance Corresponding to CAN Baud Rate 83
8.3.3 CAN Interface System Variable........ 83
8.4 CANopen Communication Instruction........ 84
8.4.1 CANopen Communication Protocol........ 84
8.4.2 CANopen Axis Control Instruction List........ 84
8.4.3 Explanation of CANopen-related Terms 85
8.4.4 CANopen Indicators........ 85
8.5 CANopen Configuration........ 85
8.5.1 Master Configuration........ 87
8.5.2 Slave Configuration 90
8.5.3 PDO Enabling 92
8.5.4 PDO Mapping Editing........ 92
8.5.5 PDO Attribute Settings 92
8.5.6 Service Data Object 93
8.5.7 Online Debugging Function........ 94
8.5.8 I/O Mapping........ 95
8.5.9 Device Information 96
8.5.10 CANopen Slave Description........ 96
8.6 Troubleshooting of CANopen Communication 98
8.6.1 Common Troubleshooting Steps........ 98
8.6.2 Fault Code List 98
8.7 CAN Free Protocol........ 100
8.7.1 Function Introduction of CAN Free Format 100
8.7.2 Communication Specifications of CAN Free Format........ 100
8.7.3 CAN Free Port Configuration 100
8.7.4 CAN Free Format Data Sending and Receiving........ 101
9 EtherCAT Communication........ 102
9.1 Overview........ 102
9.2 Master Configuration 102
9.2.1 Import Device XML 102
9.2.2 Scanning Device........ 103

9.2.3 Master Setting 105
9.2.4 Start-stop & Disable & Enable 106
9.2.5 Summary of System Variables 106
9.3 Slave Configuration 108
9.3.1 General Settings........ 108
9.3.2 Process Data 109
9.3.3 Startup Parameters 111
9.3.4 I/O Function Mapping........ 112
9.3.5 Start-stop/Disable/Enable 115
9.3.6 Disable Slave by Instruction........ 115
9.3.7 System Variables........ 117
9.4 Fault and Diagnosis 118
9.4.1 Fault Acquisition 118
9.4.2 Fault Code 118
10 EtherNet/IP Communication 119
10.1 Overview........ 119
10.2 Class1 Communication........ 119
10.2.1 Slave Configuration 119
10.2.2 Use Example of Slave........ 121
11 Motion Control 125
11.1 Brief Introduction of Motion Control Axis 125
11.1.1 Overview 125
11.1.2 PLCopen State Machine........ 127
11.1.3 Unit of Axis 128
11.1.4 Axis Configuration Parameters 128
11.1.5 Axis System Variable........ 129
11.1.6 Axis Instruction List........ 132
11.2 Motion Control Axis Setting Steps........ 133
11.2.1 Create a New Project Document 133
11.2.2 Create Project Configuration 135
11.2.3 Set Axis Parameters 139
11.2.4 Write a Program 141
11.2.5 Download Project 142
11.2.6 Basic Motion........ 143
11.3 Motion Control Axis Configuration........ 146
11.3.1 Comparison of Fieldbus Servo Axis and Local Pulse Axis........ 146
11.3.2 Basic Settings........ 147
11.3.3 Mode Setting........ 149
11.3.4 Axis Speed Setting 151
11.3.5 Probe Setting 151
11.3.6 Output Settings........ 151
11.3.7 Home Setting 152
11.4 Online Monitoring 153
11.5 Axis Control Function........ 154
11.5.1 Overview 154
11.5.2 Online Debugging 154
11.5.3 Instruction Control Rule 156
11.5.4 Limit Processing........ 157
11.5.5 Positioning Acceleration and Deceleration Curve........ 157
11.6 Fault Type........ 157
12 High-speed Counter 158
12.1 Brief Introduction of High-speed Counter Axis........ 158
12.2 Create a Counter Axis........ 158
12.3 Counter Axis Unit and Conversion 159
12.4 Set the Working Mode........ 160
12.4.1 Linear Mode 160
12.4.2 Rotation mode 162
12.5 Set Counter Parameters 162
12.5.1 Overview 162

12.5.2 Counting Mode ... 163
12.5.3 Hardware Reset Settings ... 165
12.5.4 Probe Setting ... 165
12.5.5 Preset Settings ... 166
12.5.6 Comparison Output Settings ... 166
12.6 Application of Encoder Axis Instruction ... 166
12.6.1 Overview ... 166
12.6.2 Axis Position Ranging/Speed Measurement Instruction ... 166
12.6.3 Axis Position Preset Instruction ... 167
12.6.4 Probe Instruction ... 167
12.6.5 Single-step Comparison Instruction ... 169
12.6.6 Continuous Comparison Instruction ... 169
12.6.7 Array Comparison Instruction ... 170
12.6.8 Hardware Comparison Output of Encoder Axis ... 170
12.6.9 Encoder Axis Comparison Interrupt ... 171
12.6.10 Modify the Gear Ratio of Encoder Axis ... 172
12.6.11 Modify Mode/Limit Value of Encoder Axis ... 172
13 Interpolation Function ... 174
13.1 Brief Introduction of Interpolation Function ... 174
13.1.1 Basic Introduction ... 174
13.1.2 Instruction List ... 174
13.1.3 Upper Computer Axis Group Configuration Interface Configuration ... 175
13.1.4 Online Monitoring ... 176
13.2 Interpolation Operation ... 177
13.2.1 Basic Introduction ... 177
13.2.2 State Machine of Axis Group ... 177
13.2.3 Axle Group Enable and Reset ... 177
13.2.4 Linear Interpolation ... 178
13.2.5 Plane Arc Interpolation ... 179
13.2.6 Axis Group Halt ... 180
13.2.7 Axis Group Stop ... 180
13.2.8 Axis Group Immediate Stop ... 181
13.2.9 Axle Set Speed regulation ... 181
13.2.10 Axis Group Interrupt Mechanism ... 182
14 Electronic Cam Function ... 183
14.1 Brief Introduction of Electronic Cam ... 183
14.2 Software Configuration ... 183
14.2.1 Overview ... 183
14.2.2 Cam Node Setting ... 184
14.2.3 Cam Curve Setting ... 184
14.2.4 Import and Export ... 185
14.2.5 Call Instruction ... 185
14.3 State Machine ... 186
14.4 Electronic Cam Operation ... 187
14.4.1 Gear Action ... 187
14.4.2 Cam Action ... 189
14.4.3 Cam Table ... 192
14.4.4 Master Axis Phase Compensation ... 195
14.4.5 Motion Superposition ... 196
14.4.6 How to Handle Axis Configuration Parameters in Cam or Gear ... 196
15 Memory Formula Management ... 197
15.1 Overview ... 197
15.2 Power-down Variable Keeping ... 197
15.2.1 Range of Power-down Variable Keeping ... 197
15.2.2 Data Retention File Upload and Download ... 198
15.3 Memory Management of Custom Variable Table ... 199
15.3.1 Expand and Fold Complex Type Variables ... 199
15.3.2 Monitor Variables ... 199
15.3.3 Edit Variable Initializations and Comments ... 199

15.3.4 Switch and Display Binary System ... 200
15.4 Memory Management of Soft Elements ... 200
15.4.1 Operation Interface ... 200
15.4.2 Editing Rules for Data Types ... 201
16 Expansion Module ... 202
16.1 TS600 Series Host Local Expansion Module ... 202
16.1.1 Summary of Expansion Module ... 202
16.1.2 Expansion Module System Variables ... 203
16.1.3 Expansion Module Configuration ... 203
16.1.4 Expansion Module Configuration ... 204
16.2 Basic Operation of Local Expansion Module ... 215
16.2.1 Module Enable ... 215
16.2.2 Module Disable ... 215
16.2.3 Get Physical Configuration ... 216
17 Subroutine, Library ... 217
17.1 Subroutine ... 217
17.2 General Subroutine Application ... 217
17.2.1 Matters Needing Attention in the Use of General Subroutines ... 218
17.2.2 Definition of General Subroutine Variable Table ... 218
17.2.3 General Subroutine Parameter Passing ... 219
17.2.4 Examples of Using General Subroutines ... 219
17.3 Interrupt Subroutine Application ... 221
17.3.1 External Interrupt Subroutine ... 221
17.3.2 Timer Interrupt Subroutine ... 222
18 Application of Custom Variable Communication ... 224
18.1 Overview ... 224
18.2 Example Project ... 224
18.2.1 Accessing Custom Variables via Binding to Elements ... 224
18.2.2 Accessing Custom Variables via Tag Communication ... 227
19 Fault Diagnosis ... 232
19.1 Panel Diagnosis ... 232
19.1.1 Indicator ... 232
19.2 Software Diagnosis ... 233
19.2.1 Get the Basic Information of PLC ... 233
19.2.2 View Error Log ... 233
19.3 Fault Code List ... 234
19.3.1 Fault Code Classification ... 234
19.3.2 Fault Code List ... 235
20 Firmware Burning and Upgrade ... 288
20.1 Upper Computer Firmware Upgrade ... 288
20.2 SD Card Firmware Upgrade ... 289
20.2.1 Step of Generating PLC Application Upgrade Package ... 289
20.2.2 SD Card Upgrading Steps ... 290
20.3 Upgrade of Upper Computer Applications or Open the Project ... 293
20.3.1 Upgrade Applications with Upgrade Pack ... 293
20.3.2 Open a Project with Upgrade Pack ... 293
21 4G IoT Expansion Card ... 294
21.1 Overview ... 294
21.2 User Login of IoT Monitoring Platform ... 294
21.3 Add a Device Type ... 295
21.4 Add Devices to IoT Platform ... 295
21.5 Device Installation & Wiring ... 296
21.6 Establish VPN Transparent Transmission Channel ... 297
21.6.1 Connect PLC Remotely through VPN ... 298
21.6.2 FAQ ... 299
21.7 Web Page Monitoring ... 300
21.8 Module Interface Description ... 304
22 Appendix ... 306
22.1 Modbus Protocol ... 306

22.1.1 Modbus RTU 306
22.1.2 Modbus ASCII 306
22.1.3 Modbus TCP 307
22.1.4 Modbus Function Code 307
22.1.5 Example of Modbus Message Parsing 308
22.1.6 Modbus Abnormal Response 309

1 Overview

1.1 Product Overview

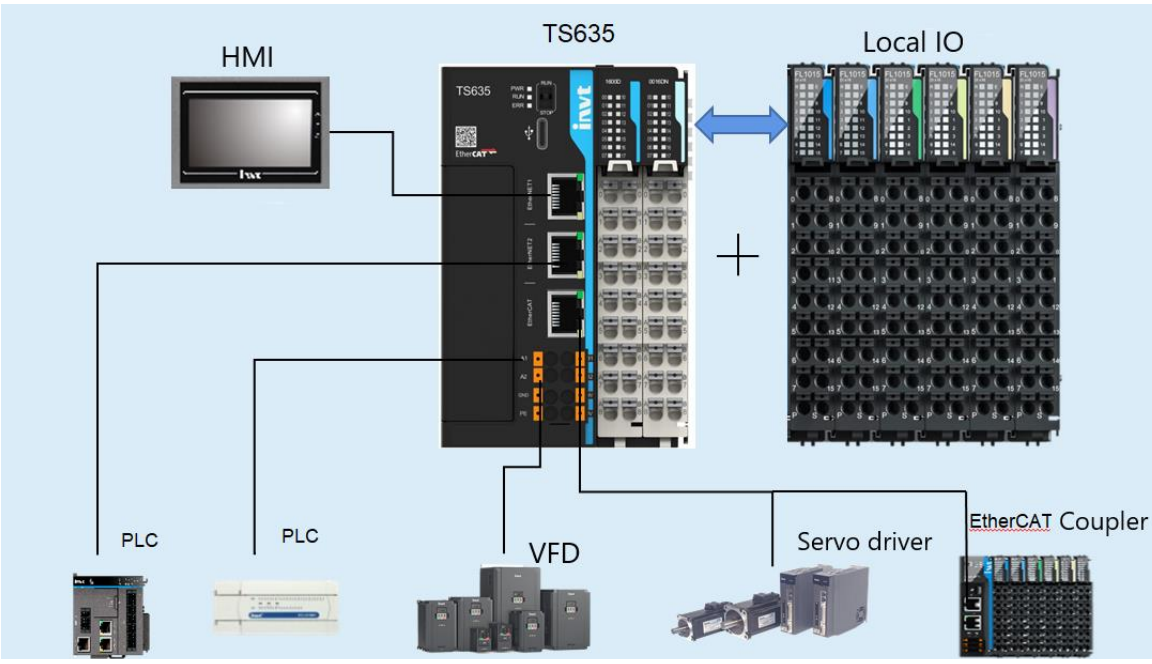
TS600 is a series of new-generation small PLC products independently developed by INVT. With powerful motion control and distributed I/O control functions, it supports EtherCAT fieldbus communication, can conveniently realize user process movement through custom variables, axis control configuration and graphic block instructions, and can realize multi-level network communication via RS485, CAN, EtherCAT interfaces.

TS600 series full-scene compact controllers are available in multiple models to meet various needs of users for compact automation equipment, and are suitable for compact design, multi-axis motion control, temperature control, communication networking and other scenarios.

1.2 One-stop Solutions

The typical application topology of TS600 series solution is illustrated by TS600 series TS635, as shown in the following figure.

Figure 1-1 Typical Application Topology of TS635



2 Quick Start

2.1 Programming Software

2.1.1 Software Introduction

Auto Station Pro is the upper computer software of INVT's TS600 Series Small PLC. The user interface of Auto Station Pro programming software is user-friendly, which is convenient for users to configure, program, debug, download and monitor PLC as required.

2.1.2 Computer Configuration Requirements

Table 2-1 Basic Configuration of Auto Station Pro Programming Environment

Item	Minimum configuration	Recommended configuration
CPU	Equivalent to Intel's Pentium D805 or above	Equivalent to Intel's Pentium Dual G4620 or above
Memory	4GB	Above 4GB
Graphics card	Functional in 1280 × 640 resolution and 256 color mode	Functional in 1280 × 640 resolution and 65535 color mode
OS	Windows 7 and later	Windows 7 and later
Communication ports	USB type-C interface, RJ45	
Other devices	Power supply	

2.1.3 How to Get Our Software

Auto Station Pro is free, and users can download the software installation package in the application software options of "Services and Support" > "Data Download" on INVT official website (www.invt.com).

Our Company is constantly improving products and materials, so the users should update the software version in time when they need it, and consult the latest released reference materials, which is beneficial to the program design of users.

2.1.4 Auto Station Pro Programming Software Installation Process

The Auto Station Pro installation package released by Shenzhen INVT Electric Co., Ltd. is a separate executable program whose installation process can be started by double-clicking.

1. Follow the installation wizard step by step. Users can choose different installation paths as required.
2. After installation, the INVT program group will appear in the Start menu; At the same time, the installer will also install the Auto Station Pro shortcut icon on the desktop.
3. Double-click the shortcut icon to run the program.

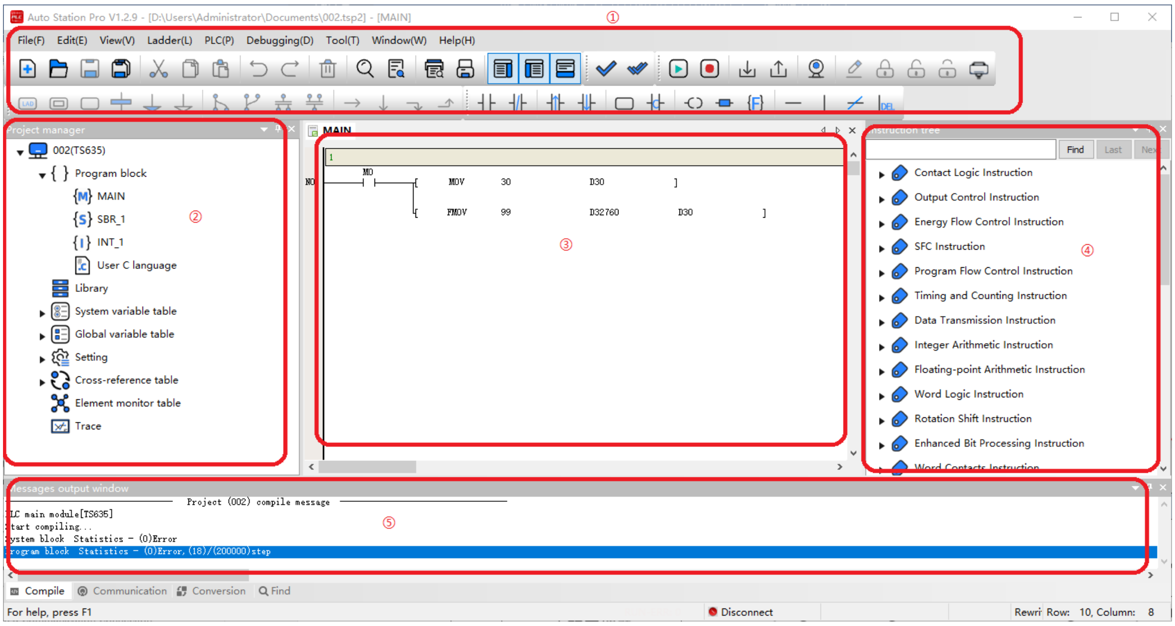
Uninstall: The software can be uninstalled through Windows Control Panel.

Note: To upgrade and install the new version of Auto Station Pro, please uninstall the old version of Auto Station Pro first.

2.1.5 Auto Station Pro Running Interface

The main interface of this program basically includes 5 parts: menu bar and toolbar, "Project manger" window, "Instruction tree" window, "Messages output" window and workspace.

Figure 2-1 Auto Station Pro Main Interface



No.	Name	Description
①	Main menu bar and shortcut toolbar	Programming software operation menu, including programming, debugging, communication and other related settings, file management and programming debugging shortcuts
②	Project manager	It includes program management, variable management, parameter management and configuration management of PLC projects.
③	Workspace	Used to write user programs
④	"Instruction tree" window	Used to load the instruction set supported by PLC in the project, and load the expansion module supported by PLC
⑤	"Messages output" window	It includes compilation information, communication information, transformation information, and search information

2.1.6 Programming Language

The programming languages supported by Auto Station Pro include:

- Ladder diagram (LAD)
- Instruction List (IL)
- Sequential Function Chart (SFC)
- User C language

2.2 Communication Settings

2.2.1 Communication Connection Method

The communication connection with Auto Station Pro can be established via USB interface or Ethernet to realize the download, monitoring and debugging of programs.

2.2.2 Ethernet Connection

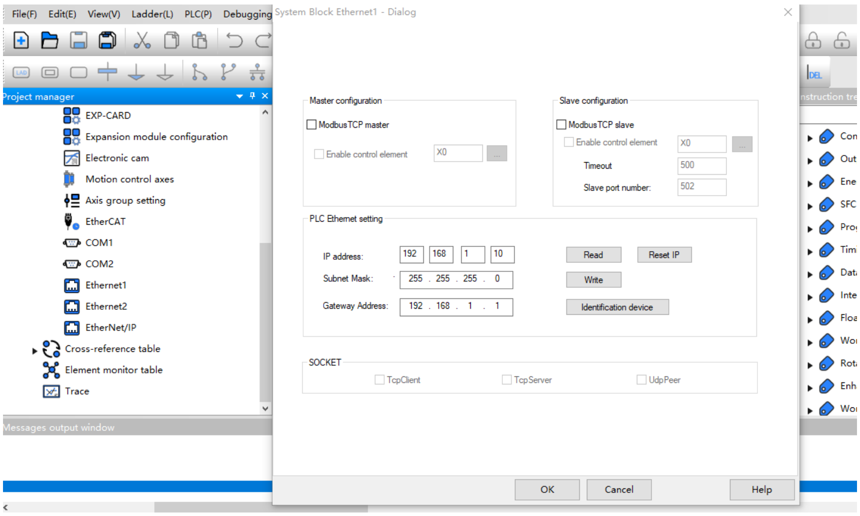
When users connect PLC via Ethernet, the operations involved may include selecting network port, reading IP address of device, modifying IP address of device and PING function. This section will focus on these operations.

Double-click Ethernet 1 in the "Project manager" bar, as shown on the left side of the figure below, and the menu of Ethernet 1 in the system block will pop up as shown on the right side of the figure below.

- PLC network port setting

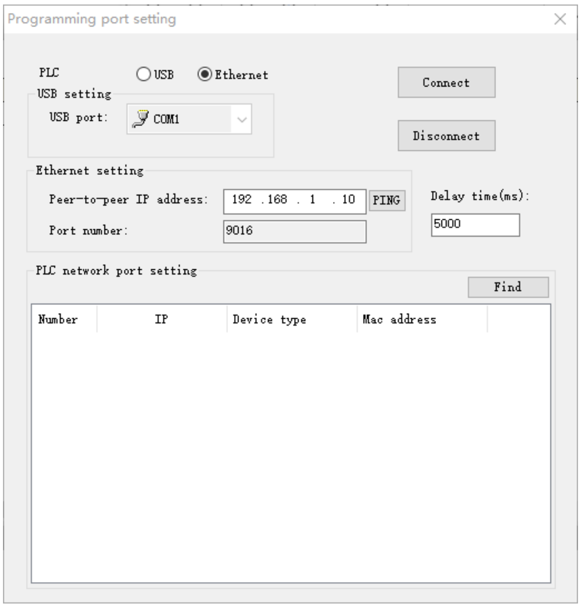
IP address, subnet mask and gateway address can be clicked to read the current configuration value. If you need to change it, you can modify it and click to write. The default IP of Ethernet 1 is 192.168.1.10. If you need to factory reset the settings, you can click to reset IP.

The high-end model of TS600 has two Ethernet ports, and you should reference Ethernet 1 for the reading and writing of IP of Ethernet 2.



- PING function

Auto station Pro comes with PING function, which can test whether the network connection between computer and target PLC device is normal, and the destination IP address should be filled in correctly during the test.



2.3 Typical Steps of Programming for Users

The user who uses INVT's PLC products for the first time needs to note that 5 steps are required to write and debug a complete user program. Take TS600 series PLC as an example.

Step 1 Establish a project

Step 2 Start hardware system configuration based on hardware connection architecture of PLC application system of TS600 series

A. If only TS600 main module is used, you can go directly to the next step.

B. If the local expansion module is used, the module needs to be configured. According to the actual selected and mounted module type and model and installation sequence, add and configure the module on the "Expansion module configuration" page of Auto Station Pro.

Step 3 Complete communication configuration and set corresponding function parameters according to the application requirements

According to the application requirements, set the corresponding functional parameters in "Setting" in the "Project manager".

Step 4 According to the application requirements, write user programs and establish related variables

Step 5 Connect PLC in Auto Station Pro programming environment, compile, download, debug and troubleshoot the user program until it runs correctly.

2.4 Programming and Debugging

2.4.1 Sample Project Requirements

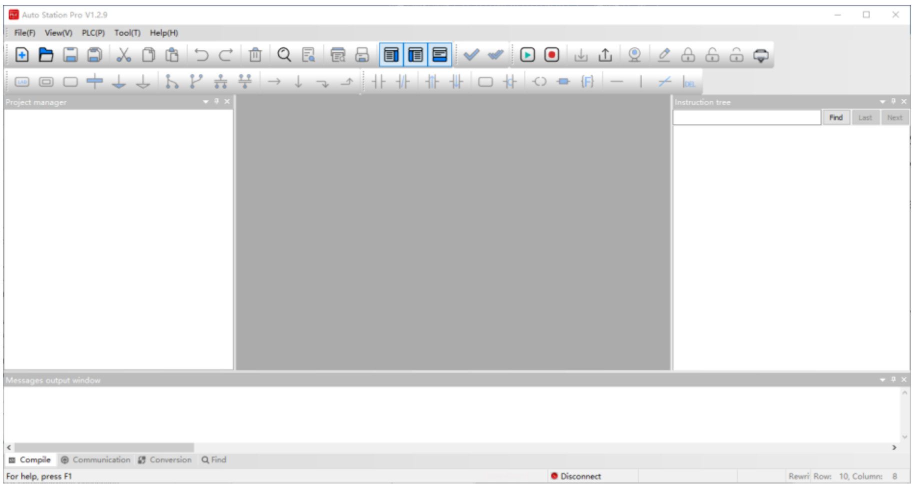
Complete a marquee programming and debugging process: Suppose the output Y0, Y1, Y2 respectively control three lights, a red one, a yellow one and a green one, and use input X0 as a start switch, the program will do the following control:

- When the switch is turned on, the red, yellow and green lights are lit alternately for 2 seconds each (using timers T0, T1 and T2 with an accuracy of 100ms)
- When the switch is turned off, the three lights go out

2.4.2 Start the Programming Environment

After the programming software is installed correctly, click the programming software icon from the Start menu bar to start the software. The main interface is as shown in Figure 2-2.

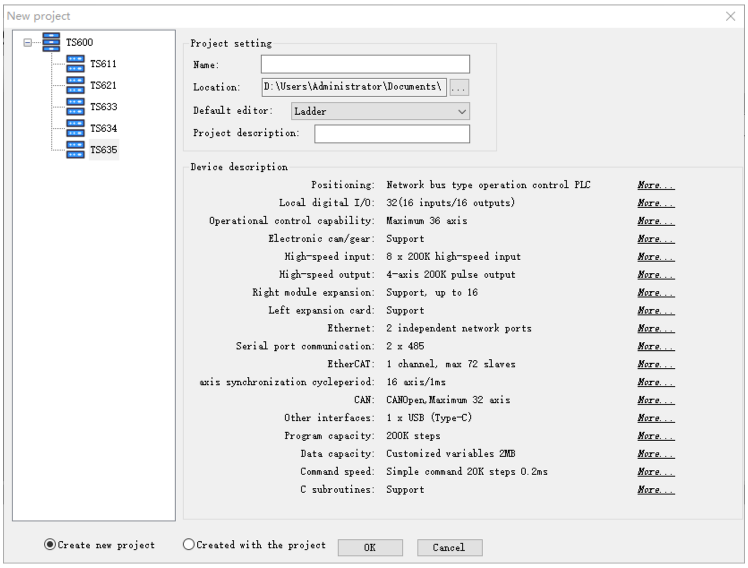
Figure 2-2 Auto Station Pro Startup Main Interface



2.4.3 Establish a Project

Create a new TS635 project by choosing "Default editor" (ladder diagram, instruction list, sequential function chart), as shown in Figure 2-3.

Figure 2-3 Auto Station Pro: New Project Interface



2.4.4 Write a Control Program

Figure 2-4 Creating related variables

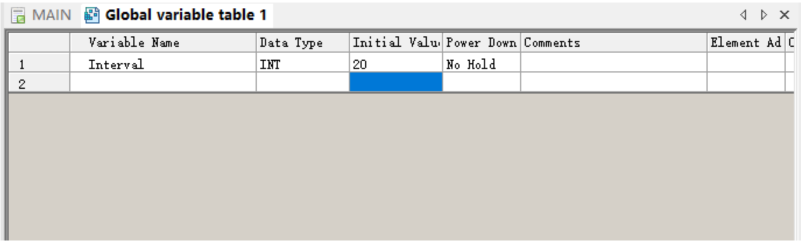
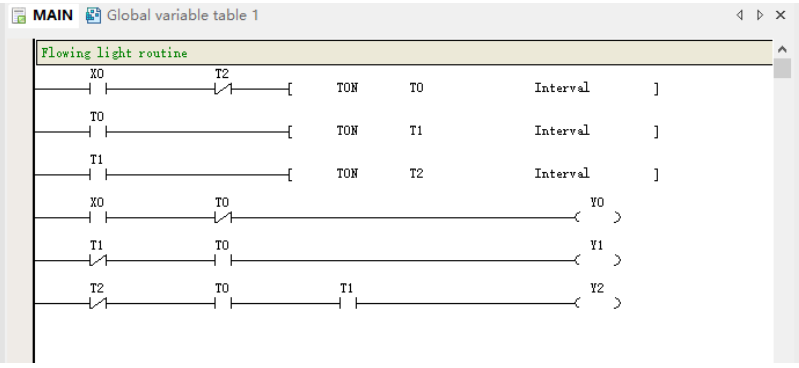


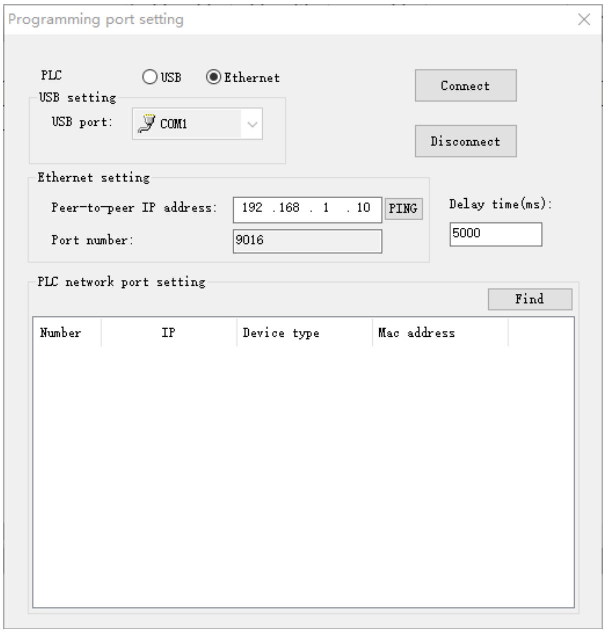
Figure 2-5 Programming user programs



2.4.5 Establish a Communication Connection

Choose "Menu bar" > "Tool" > "PLC communication". The interface is as shown in Figure 2-6.

Figure 2-6 Establishing Communication Connection



For "PLC", you can choose USB communication or Ethernet communication.

- To select USB, you need to choose "USB". After selecting the port, click "Connect", as shown in Figure 2-7.
- When choosing Ethernet connection, you need to set the "IP address" of the lower computer, then click "Connect", as shown in Figure 2-8.

Figure 2-7 Establishing USB Connection

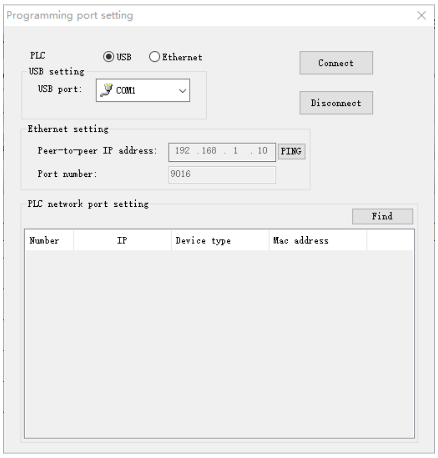
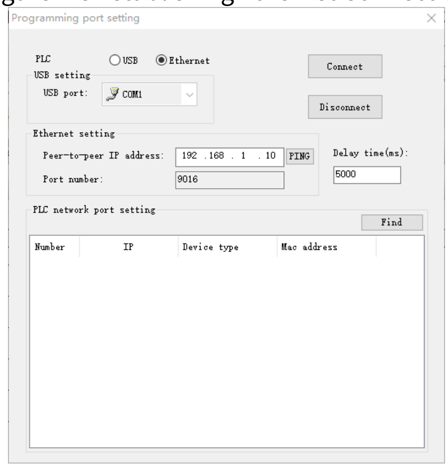


Figure 2-8 Establishing Ethernet Connection

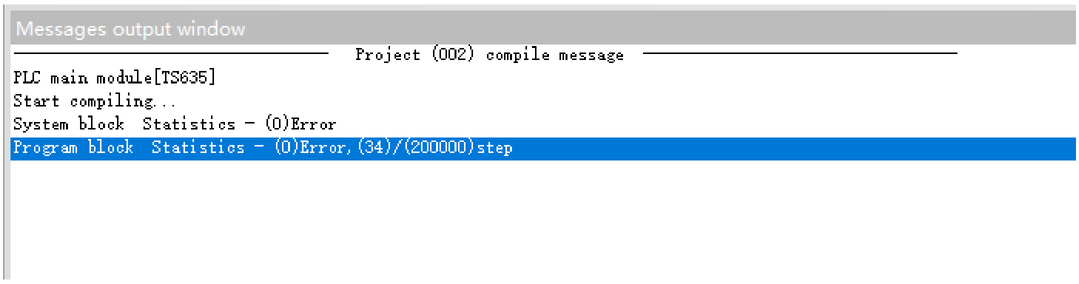


2.4.6 Compile and Download the Program

Step 1 Open the edited PLC program and click the compile button  to compile. The difference between these two compile buttons is that the left one is used to compile only the program on the currently open page, and the right one is used to compile the entire project.

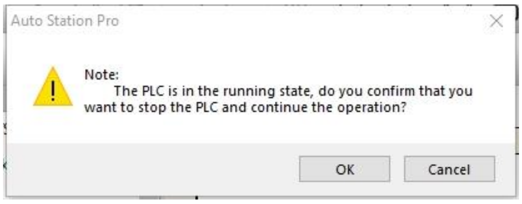
After compiling, the compilation information is output below the software.

Figure 2-9 Successfully Compiled



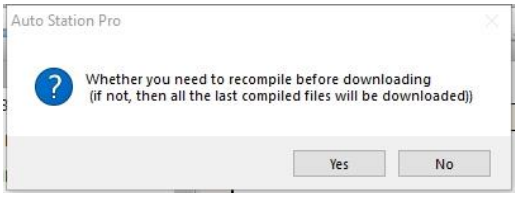
Step 2 After successful compilation, you can click the Download button to download. When downloading the program, PLC must stop. If PLC is running, the system will prompt with a dialog box, as shown in Figure 2-10.

Figure 2-10 Stop Running



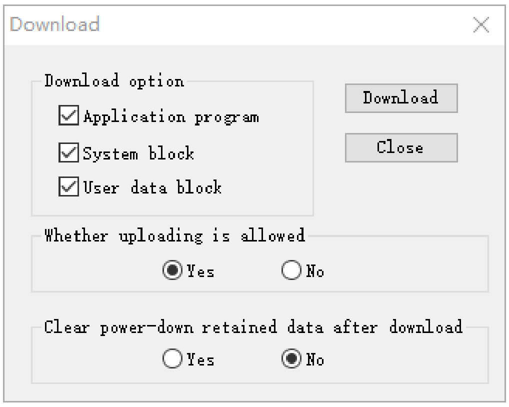
Step 3 Click the "OK" button to stop PLC, and the system will prompt with a compiling dialog box, as shown in Figure 2-11.

Figure 2-11 Compile



Step 4 If you don't want to recompile and download the last compilation result, you can select "No", otherwise, you can select "Yes" to recompile the program. After compilation, a download window will appear, as shown in Figure 2-12.

Figure 2-12 Download Options



According to your needs, select the content to be downloaded in the "Download option". In this example, only the written application program needs to be downloaded. Select the required application program, and click the "Download" button to start downloading. A progress bar will be displayed during the downloading process, and the user will be prompted of the successful download after completion.

2.4.7 Start the Programmable Controller

After the application program is downloaded, you must start the programmable controller to run. Select "PLC" > "Start", or toggle the switch of the programmable controller to RUN. After the programmable controller runs normally, close the switch connected to the input point X0, and you can see that the three lights connected to the output points Y0, Y1 and Y2 are lit in turn.

2.5 Switch PLC Working Mode

- PLC has two working modes, namely RUN and STOP:
  - ✧ RUN mode: In this mode, PLC mainly performs X-point input testing, scanning operation of user program, element refresh, Y-point output and communication.
  - ✧ STOP mode: In this mode, PLC stops the scanning of user application program and the output of Y point, and the communication also stops.

  Note: In STOP mode, if the upper computer is monitoring, PLC will still communicate with the upper computer
- If customers need to switch the working mode of PLC, there are two methods to choose:
  - ✧ Use the physical toggle switch RUN/STOP on the PLC panel to switch.
  - ✧ In Auto Station Pro, click the RUN or STOP button in the toolbar to switch between Run and Stop state.

2.6 Trace

2.6.1 Function Introduction

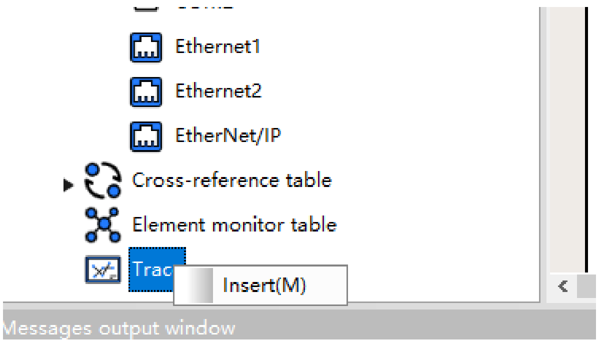
In Auto Station Pro, users can use element oscilloscope function, which is similar to digital sampling oscilloscope and can record the historical values of lower computer elements. When Trace function is enabled, Auto Station Pro will start to store the time-stamped value data records. Users can continuously monitor the changes of element values on the element oscilloscope page, which is convenient for program debugging.

2.6.2 Operating Steps

This section demonstrates how to use Trace to capture changes in D0, R0, and VAR_01, and introduces how to operate the element oscilloscope function.

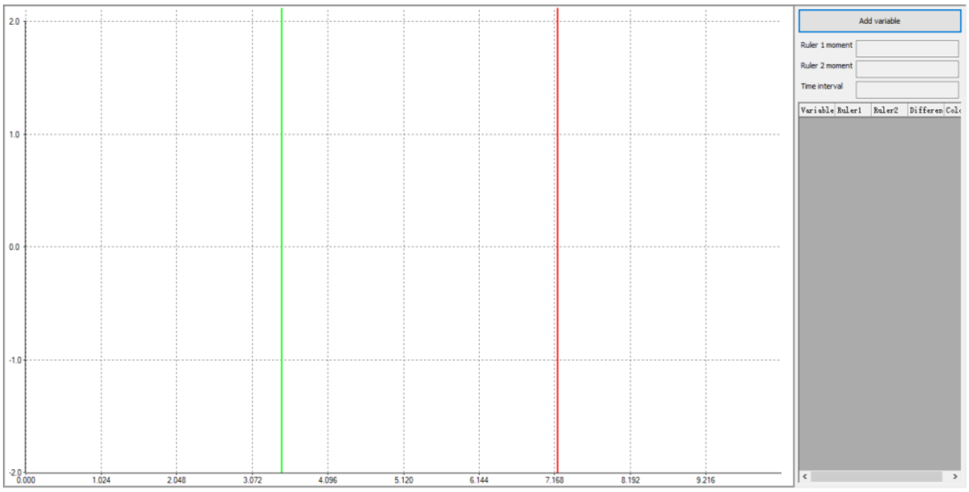
Step 1 Right-click "Trace" in Project Management and click to insert a new Trace Monitoring Table.

Figure 2-13 Open Element Oscilloscope Function



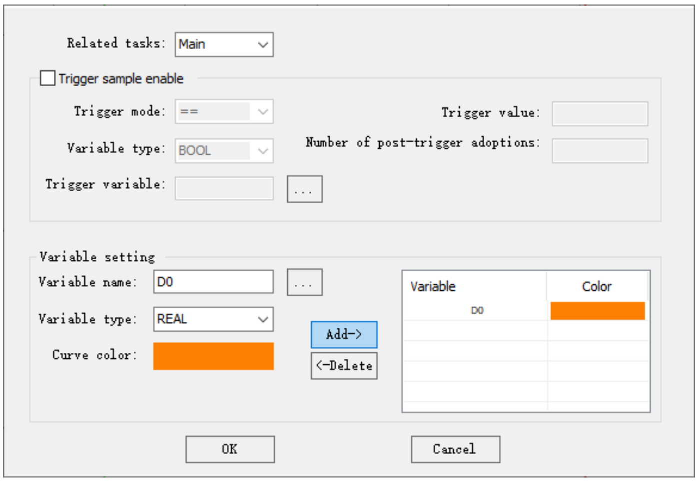
Step 2 Double-click the newly created Trace Monitoring Table to enter the Trace interface.

Figure 2-14 Main Interface of Trace



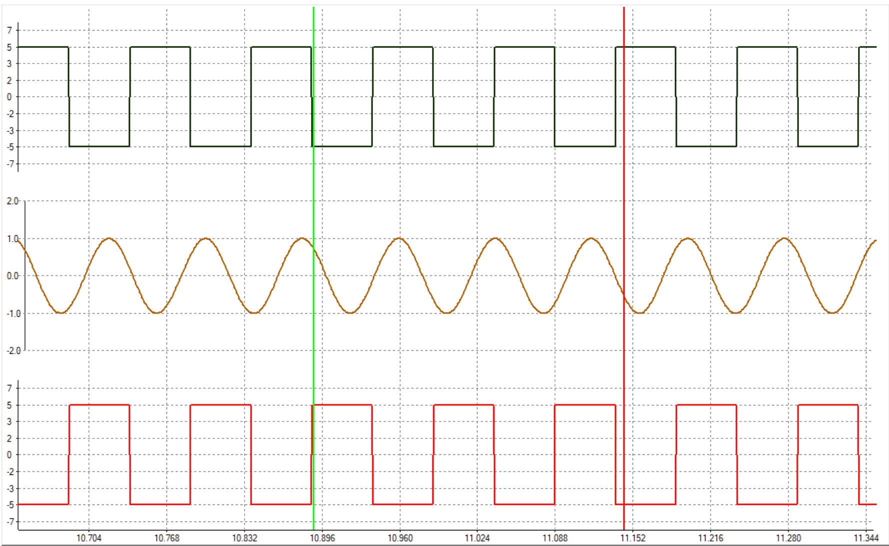
Step 3 Click "Add Variables" in Figure 2-14, and choose the main task from the associated tasks in the "Variable setting" pop-up window. Set variables D0, R0 and VAR_01 to be captured, and then click "OK" to save the variable settings.

Figure 2-15 Add Variables in Trace



Step 4 After configuring the variables, right-click on the Trace view and choose "Start Track".

Figure 2-16 Element Oscilloscope



Note: The abscissa and ordinate in the trace state will automatically adjust with the value of variables, and users can also operate with the mouse wheel.

Step 5 Right-click on the Trace view to choose single channel display or multi-channel display.

Figure 2-17 Select Single Channel

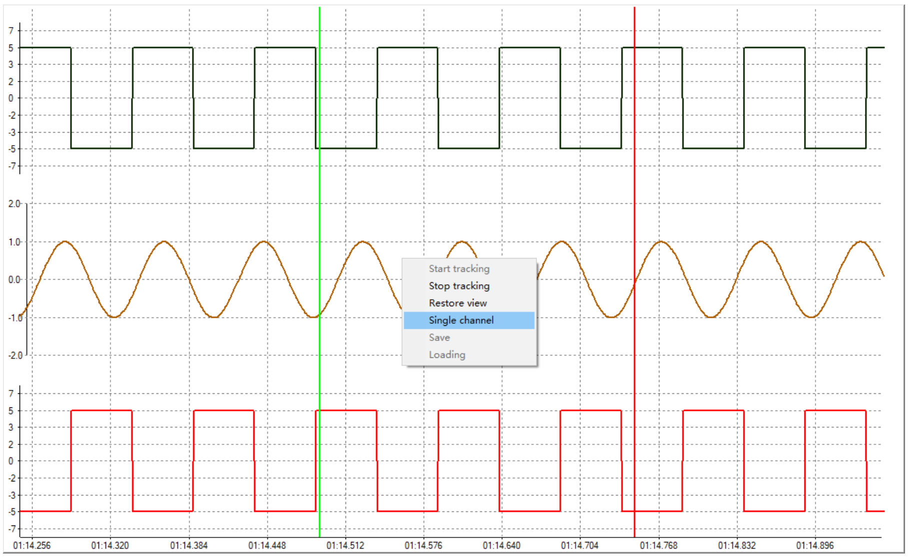
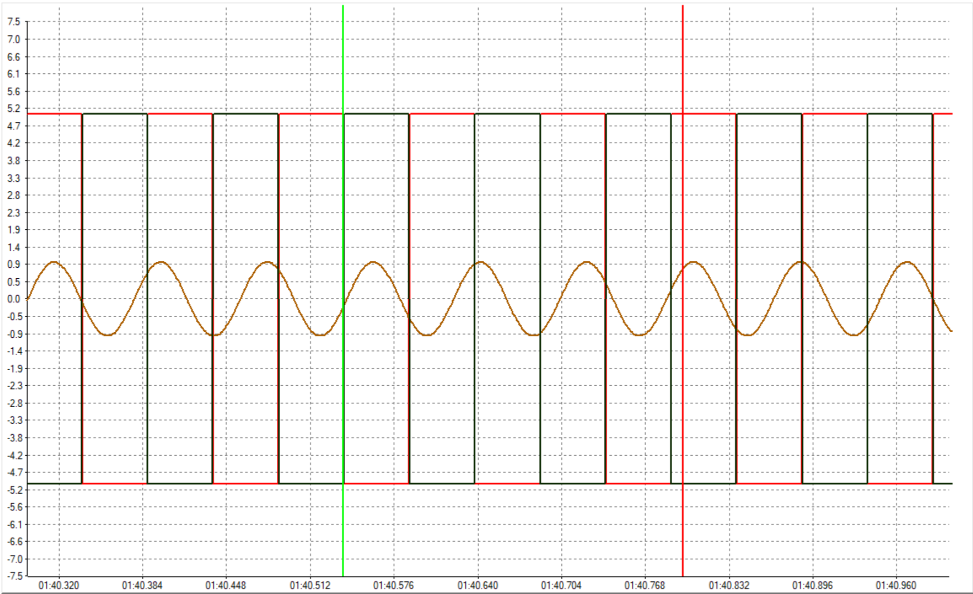


Figure 2-18 Single Channel View



2.7 Scan Period Setting

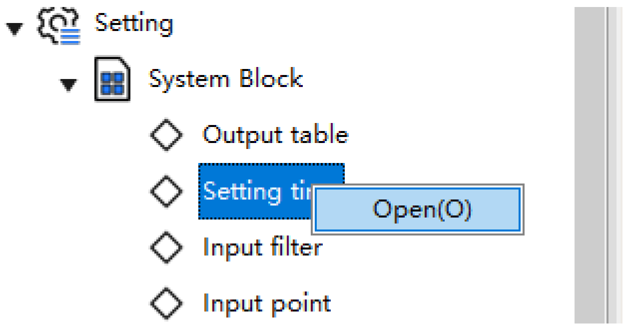
Users can choose two working modes: constant scanning period and inconstant scanning period.

- When choosing constant scanning period, the program runs according to the specified scanning period.
  - ✧ When the program running time for testing is less than the set scanning period, PLC scans the program according to the set period value.
  - ✧ When the actual program running time is longer than the set scanning period, PLC scans according to the actual running time.
- When choosing non-constant period (usually the constant scanning time is set to 0), PLC will automatically adjust the scanning period according to the program running time.

2.7.1 Time Setting

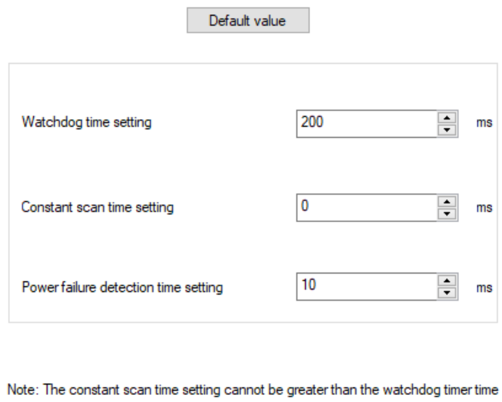
Step 1 Double-click or right-click to open "Setting time" in the system block in Project Manager.

Figure 2-19 Open Time Setting



Step 2 In the pop-up dialog box, you can set the "Scan period" of the program which is default to 0, that is, the non-constant scanning period mode. In addition, users can also set the "Watchdog time setting" and "Power failure detection time setting ".

Figure 2-20 Time Setting



3 Basic Knowledge of Programming

3.1 Summary

The variable memory structure used in programming includes soft elements, custom variables and system variables.

- Soft elements: X, Y, M, LM, S, T, C, D, R, V, Z can be directly used in user programs.
- Custom variables: VTD should be defined in the variable table before use.
- System variables: _SYS_CAN, _SYS_COM, _SYS_ECAT, _SYS_ETHERNET, _SYS_INFO, _SYS_AXSI_ENC_INFO, _SYS_AXIS_MC_INFO, _SYS_GROUP_MC_INFO, _SYS_CAM_TABLE, etc. are used to obtain system internal information.

3.2 Soft Element

3.2.1 Bit Soft Element

This PLC programming supports bit soft elements. The specific type, range, points and related description of bit soft elements are shown in the following table.

Type	Range	Points	Data Type	Description
X	X0–X1777	1,024 points⊠ Octal code	BOOL	Input
Y	Y0–Y1777	1,024 points⊠ Octal code	BOOL	Output
M	M0–M32767	32768 points	BOOL	● M0–M999: not saved in case of power-down ● From M1000: saved in case of power-down
LM	LM0–LM63	64 points	BOOL	Local auxiliary relay
S	S0–S4095	4096 points	BOOL	● S0–S999: not saved in case of power-down ● From S1000: saved in case of power-down
T	T0–T399	400 points	BOOL	● Accuracy 100ms: T0–T199, 200 points ● Accuracy 10ms: T200–T299, 100 points ● Accuracy 1ms: T300–T399, 100 points
C	C0–C255	256 points	BOOL	● 16-bit ordinary CTUD: C0 – C199, 200 points ● 32-bit ordinary CTUD: C200 – C255, 56 points

3.2.2 Word Soft Element

This PLC programming supports word soft elements. The specific type, range, points and related description of word soft elements are shown in the following table.

Type	Range	Points	Data Type	Description
D	D0–D32767	32768 points	BOOL/INT/DINT/WORD/DWORD/REAL	● D0–D999: not saved in case of power-down ● From D1000: saved in case of power-down
R	R0–R32767	32768 points	BOOL/INT/DINT/WORD/DWORD/REAL	● R0–R999: not saved in case of power-down ● From R1000: saved in case of power-down
V	V0–V63	64	INT	Temporary variable, which can only be used in subroutines
Z	Z0–Z15	16	INT	Not saved in case of power-down
T	T0–T399	400 points	INT	T0–T199: 100ms accuracy T200–T299: 10ms accuracy T300–T399: 1ms accuracy
C	C0–C235	256 points	INT/DINT	C0 – C199: 16-bit CTUD or 16-bit cyclic counter C200 – C255: 32-bit CTUD

Note: The power-down keeping range cannot be changed.

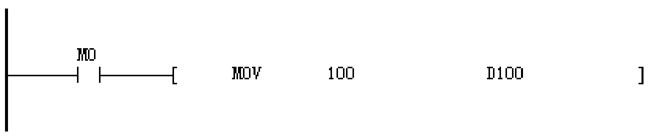
Word soft elements can be used as integers or floating-point numbers. The soft elements themselves do not have data type attribute, and the elements are interpreted as integers or floating-point numbers according to the parameter attributes of instructions.

- When used as a 16-bit integer, it occupies 1 soft element
- When used as a 32-bit integer, it occupies 2 soft element
- When used as a floating point number, it occupies 2 soft elements

Example

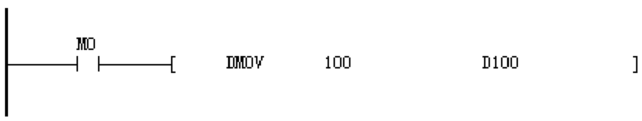
- Word soft elements used as 16-bit integers

  Use a 16-bit assignment instruction to assign a value of 100 to the word soft element D100, occupying D100.



- Word soft elements used as 32-bit integers

  Use a 32-bit assignment instruction to assign a value of 100 to the word soft element D100, occupying D100 (lower order) and D101 (higher order).



- Word soft elements used as floating-point numbers

  Use a floating point number to assign a value of 100 to the word soft element D100, occupying D100 and D101.

3.2.3 Bit Operation of Word Elements

Bit operation of word elements can be done by (.). For example, D100.8 means operation shall be done to the 8th bit on D100 word element, and the lowest bit is the 0th bit.

D100.8

1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Note: The bits of a word element start from bit 0: D100.8 can be regarded as a BOOL element, to which some bit operation instructions can be applied.

3.3 Variables

3.3.1 Custom Variables

In PLC programming system, in addition to using direct addresses, such as X, Y, M, D, R and other elements for programming, it is also possible to program with "variables" without specific addresses stored to realize the required control logic or the complete control process of application objects, thus improving the convenience and readability of code writing.

3.3.2 Define Variables

Custom variables are supported, and users can define global variables and directly use variable name for programming in the program. You need to follow the following rules when defining a global variable name:

1. It can only contain "_, letters, numbers, Chinese characters" and cannot start with "_, numbers".
2. It cannot have the same name as "soft element form, constant, standard data type, instruction".
3. It cannot be a keyword such as "ARRAY, TRUE, FALSE, ON, OFF, NULL, Struct".

Figure 3-1 Defining in Global Variable Table

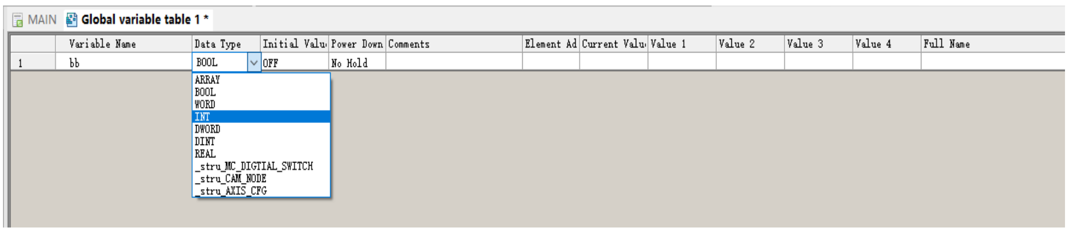
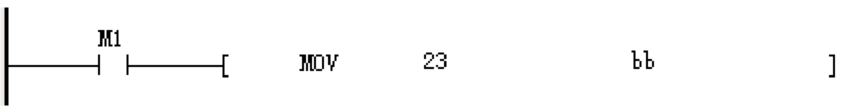


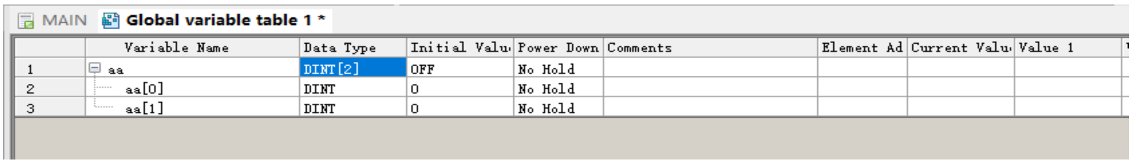
Figure 3-2 Use in Program Block



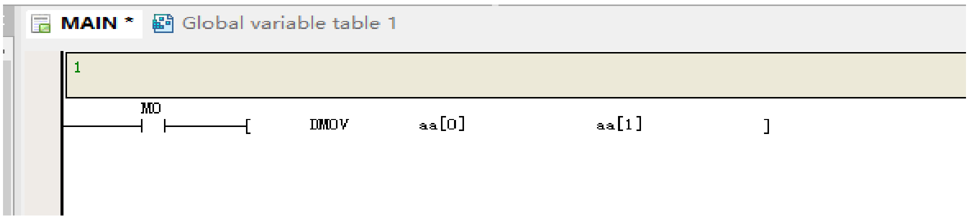
3.3.3 Define Arrays

Users can define arrays if ARRAY is chosen as the data type when programming.

1. Choose the type and length of array variables in the pop-up dialog box, and click "OK" to define the array, as shown in the following figure.

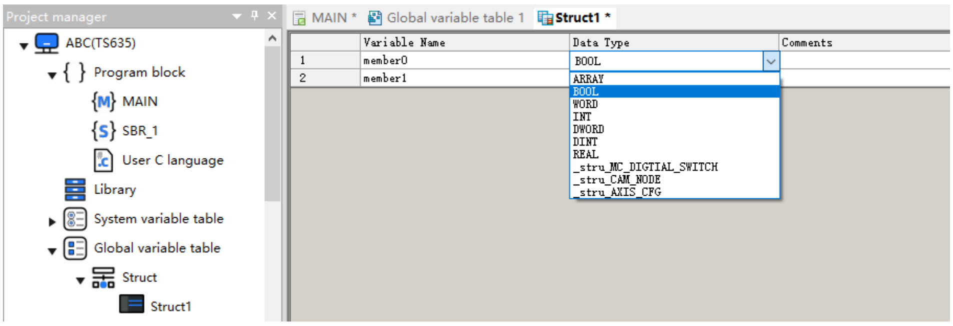


2. Use it directly in the user interface, as shown in the following figure.



3.3.4 Define Struct

Right-click in the global variable table to create a new data structure, and then right-click it to change the name of the struct variable, so as to name the struct member and select the data type of the member. When using the struct variable, the program uses "struct variable name. member variable" to represent the struct member.



3.3.5 Bit Operation of Variables

3.3.6 How to Use Variables

After the variables are defined, the variable names can be directly used for programming, and no soft elements need to be assigned.

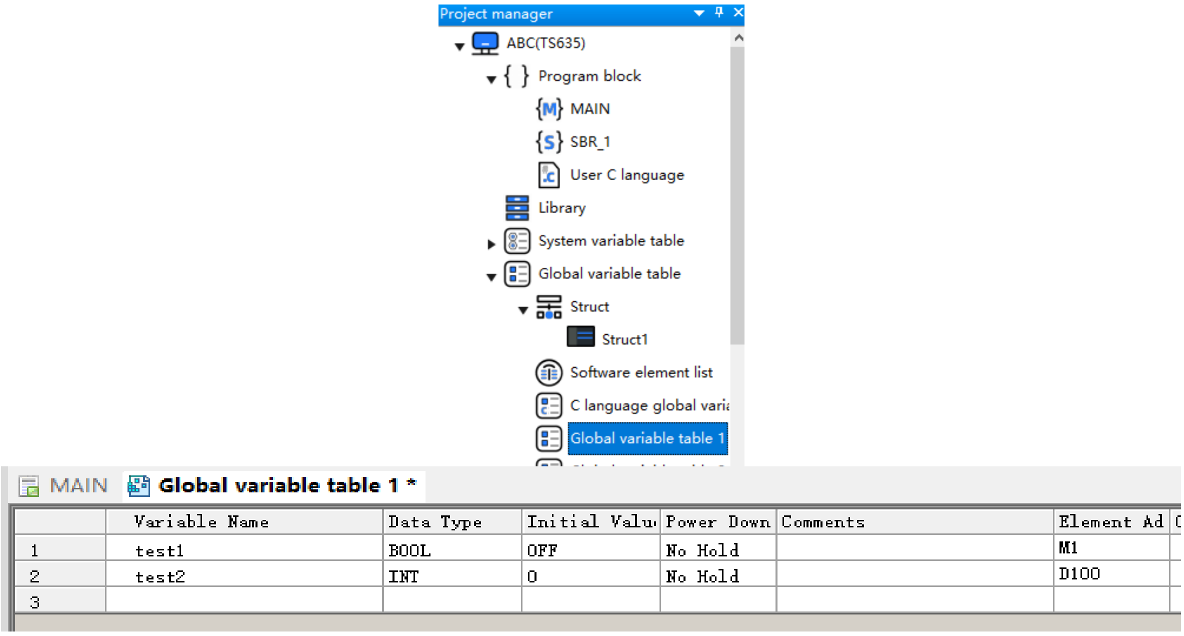
- Direct programming operation with variables
- When using array variables, the array elements are represented by "[number]" in the program, and the number starts from 0.

3.4 Variable Binding Address

3.4.1 Overview

The custom variables support binding the address of soft element. After binding, the address of custom variables is associated with the address of soft element.

To realize the function of custom variable binding software, you only need to fill the address to be associated in the address field of the variable table. Then, compile the project, and the software will automatically generate the assigned address, as shown in the following figure.

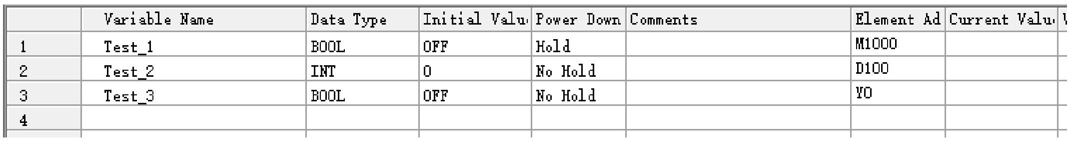


	Variable Name	Data Type	Initial Valu	Power Down	Comments	Element Ad	C
1	test1	BOOL	OFF	No Hold		M1	
2	test2	INT	0	No Hold		D100	
3							

3.4.2 Variable Attribute

After a custom variable is bound to a soft element, the power-down retention attribute changes with the bound soft element.

As shown in the following figure, M1000 is in the power-down retention area, so after Test_1 is bound to it, the power-down retention attribute correspondingly changes to retention type; While D100 is in the non-power-down retention area, after Test_2 is bound to it, the power-down retention attribute will change to non-retention type. After the element is bound, the power-down retention attribute changes automatically as the case may be, and users do not need to set it, as shown in the following figure.



	Variable Name	Data Type	Initial Valu	Power Down	Comments	Element Ad	Current Valu	\
1	Test_1	BOOL	OFF	Hold		M1000		
2	Test_2	INT	0	No Hold		D100		
3	Test_3	BOOL	OFF	No Hold		Y0		
4								

3.4.3 Bind Array Variables to Soft Elements

To bind an array variable to a soft element, you only need to fill the address to be mapped in the address field of the variable table.

- Word variables occupy the number of word elements according to variable type, one INT variable occupies a 16-bit element, and REAL and DINT variables occupy two 16-bit elements each.
- BOOL variables occupy the corresponding number of bit elements.
- Array variables can only bind soft elements of corresponding types, that is, word variables can only bind word elements, and bit variables can only bind bit elements.

For example:

Define the array variable Array_0 of BOOL type, whose length is 10, and specify binding M0 element, then M0 – M9 element will be occupied;

Define the array variable Array_1 of INT type, whose length is 10, and specify binding D0 element, then D0–D9 element will be occupied;

3.5 Use a Variable as Array Subscript

3.5.1 Rules of Usage

General rule of using a variable as array subscript: There is at most one variable which can be used as the subscript in the whole variable set.

The format is defined as array[index] or stru[index]. var, where array represents an array or struct array, index and var represent variables, stru represents a struct, and so on.

3.5.2 Basic Combination Types

- array variables, as variables of array, only support bit variable arrays, word variable arrays, double-word variable arrays and floating-point variable arrays.
- index variable, a variable as an array subscript:
  - ✧ Only single word variable INT (16 bits) and double word variable DINT (32 bits) are supported; A certain element of an array or a certain member of a struct is supported to be an index variable, such as array [index[5]], array[stru.index].
  - ✧ Soft elements are not supported, and other variables such as bit variables, floating-point variables, pointer variables, etc. are not supported; array elements with variable subscripts or struct array members are not supported to be index variables, such as array[index[i]], array[stru[i].Index];

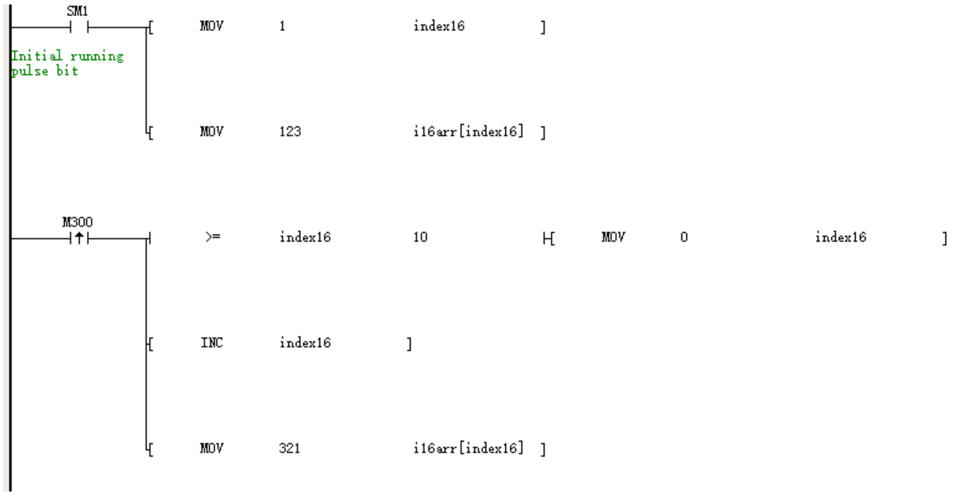
3.5.3 Complex combination Type

- The following variable types are supported:
  - ✧ You can use the element of an array as the operand of an instruction, i.e. putting an index variable at the end, such as array[index], stru.array[index], stru1[3]. stru2. array[index], stru1.stru2.stru3. array[index], and so on.
  - ✧ You can use the member of a struct array as the operand of the instruction, i.e. putting an index variable in the middle, such as stru[index].var, stru1[index].stru2.var, stru1.stru2[5].stru3[index].array[3], and so on.
- The following variable types are not supported:
  - ✧ It is not supported to use a single struct element in a struct array as the operand of an instruction, that is, to put an index variable at the end, such as stru[index], stru1.stru2.stru3[index], stru1.stru2[2].stru3.stru4[index], and so on.
  - ✧ Struct arrays with double or multiple variables, such as stru[index1]. array[index2].
  - ✧ Two-dimensional or multidimensional arrays, such as array[index1][index2].
- Instructions for use:
  - ✧ The operands of the ZSET/ZRST instruction do not support arrays with a subscript variable.
  - ✧ The operands of the PTxxx instruction do not support arrays with a subscript variable.
  - ✧ The operands of SFC instruction do not support arrays with an subscript variable.
  - ✧ For operands of instructions that use contiguous variables (array-type operands) (such as BMOV batch assignment instructions), variables in the form of arr[index] can be used in instead of a struct array element in the form of stru[index].var (because it is un-continuous); For jump assignment, you need to use loop instructions to achieve the purpose of batch jump assignment.
  - ✧ It is mainly used in the operands of single-cycle instructions, but not recommended in the operands of multi-cycle instructions. If it is used, please strictly control the logic and timing. If the timing control is not good, it may lead to abnormal execution or conflict when the value is switched (such as the axis of pulse output instruction).

3.5.4 Programming Instance

- Example 1

1. When assigning a value to an array element, and the program is as follows:



2. Start assigning 123 to i16arr[1], after which each trigger of M300 will assign 321 to each of the following array elements; After startup, the result is as shown in the following figure:

		Element Name	Data Type	Display Fo:	Current Value	New Value
1	...	i16arr	INT[16]	Decimal		
2		i16arr[0]	INT	Decimal	0	
3		i16arr[1]	INT	Decimal	123	
4		i16arr[2]	INT	Decimal	0	
5		i16arr[3]	INT	Decimal	0	
6		i16arr[4]	INT	Decimal	0	
7		i16arr[5]	INT	Decimal	0	
8		i16arr[6]	INT	Decimal	0	
9		i16arr[7]	INT	Decimal	0	
10		i16arr[8]	INT	Decimal	0	
11		i16arr[9]	INT	Decimal	0	
12		i16arr[10]	INT	Decimal	0	
13		i16arr[11]	INT	Decimal	0	
14		i16arr[12]	INT	Decimal	0	
15		i16arr[13]	INT	Decimal	0	
16		i16arr[14]	INT	Decimal	0	

3. After M300 is triggered once, the result is as follows:

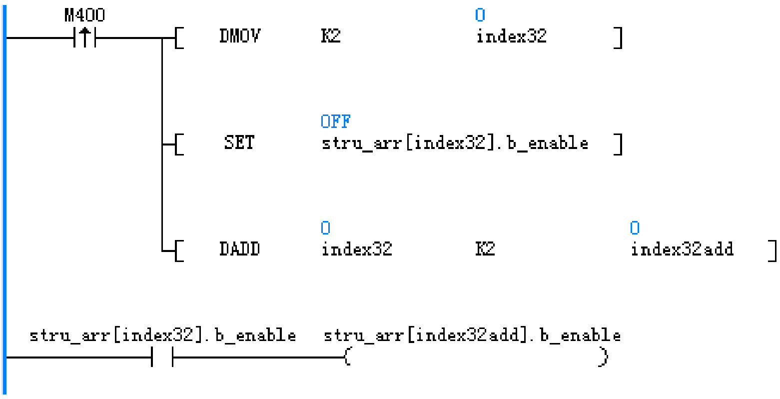
		Element Name	Data Type	Display Fo:	Current Value	New Value
1	...	i16arr	INT[16]	Decimal		
2		i16arr[0]	INT	Decimal	0	
3		i16arr[1]	INT	Decimal	123	
4		i16arr[2]	INT	Decimal	321	
5		i16arr[3]	INT	Decimal	0	
6		i16arr[4]	INT	Decimal	0	
7		i16arr[5]	INT	Decimal	0	
8		i16arr[6]	INT	Decimal	0	
9		i16arr[7]	INT	Decimal	0	
10		i16arr[8]	INT	Decimal	0	
11		i16arr[9]	INT	Decimal	0	

4. After M300 is triggered several times, the result is as follows:

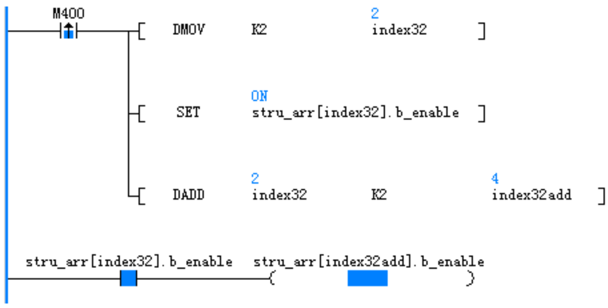
		Element Name	Data Type	Display Fo	Current Value	New Value
1	...	⊟ i16arr	INT[16]	Decimal		
2		i16arr[0]	INT	Decimal	0	
3		i16arr[1]	INT	Decimal	123	
4		i16arr[2]	INT	Decimal	321	
5		i16arr[3]	INT	Decimal	321	
6		i16arr[4]	INT	Decimal	321	
7		i16arr[5]	INT	Decimal	321	
8		i16arr[6]	INT	Decimal	0	
9		i16arr[7]	INT	Decimal	0	
10		i16arr[8]	INT	Decimal	0	
11		i16arr[9]	INT	Decimal	0	

- Example 2

1. When operating a member variable of struct array, the program is as follows:



2. After M400 is triggered, set stru_arr[2].b_enable, and control stru_arr[4].b_enable, as shown below:



Element monitoring table

EMT_1 EMT_2

		Element Name	Data Type	Display Fo	Current Value	New Value
1	...	⊟ stru_arr	Struct1[5]	Decimal		
2		⊟ stru_arr[0]	Struct1	Decimal		
3		stru_arr[0].b_e	BOOL	Decimal	OFF	
4		stru_arr[0].i16	INT	Decimal	0	
5		stru_arr[0].i32	DINT	Decimal	0	
6		⊟ stru_arr[1]	Struct1	Decimal		
7		stru_arr[1].b_e	BOOL	Decimal	OFF	
8		stru_arr[1].i16	INT	Decimal	0	
9		stru_arr[1].i32	DINT	Decimal	0	
10		⊟ stru_arr[2]	Struct1	Decimal		
11		stru_arr[2].b_e	BOOL	Decimal	ON	
12		stru_arr[2].i16	INT	Decimal	0	
13		stru_arr[2].i32	DINT	Decimal	0	
14		⊟ stru_arr[3]	Struct1	Decimal		
15		stru_arr[3].b_e	BOOL	Decimal	OFF	
16		stru_arr[3].i16	INT	Decimal	0	
17		stru_arr[3].i32	DINT	Decimal	0	
18		⊟ stru_arr[4]	Struct1	Decimal		
19		stru_arr[4].b_e	BOOL	Decimal	ON	
20		stru_arr[4].i16	INT	Decimal	0	
21		stru_arr[4].i32	DINT	Decimal	0	
22	...		WORD	Decimal		

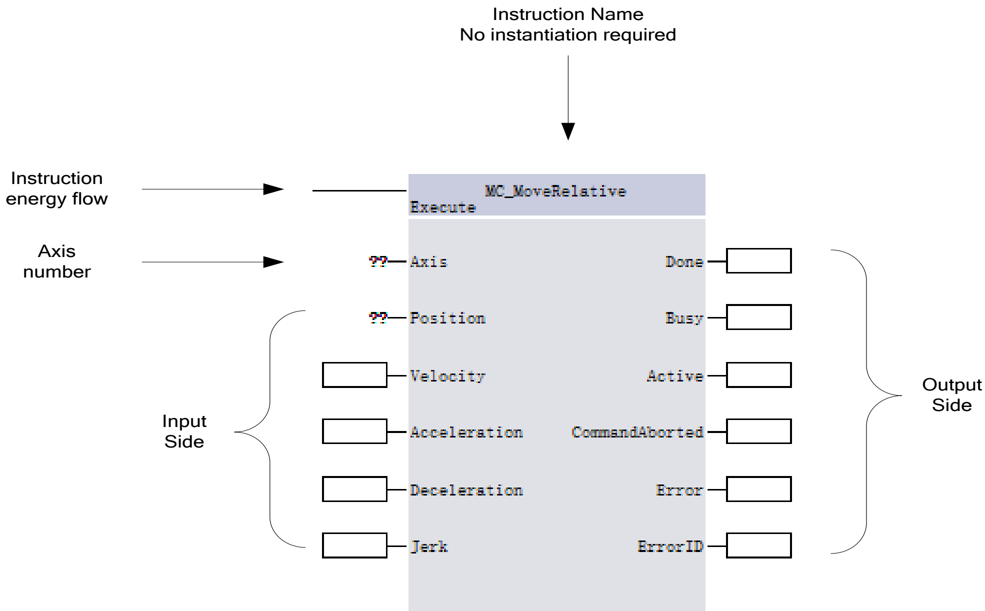
Where, the struct is defined as follows:

	Variable Name	Data Type
1	b_enable	BOOL
2	i16_a	INT
3	i32_b	DINT
4		

3.6 Graphic Block Instruction

3.6.1 Composition of Graphic Blocks

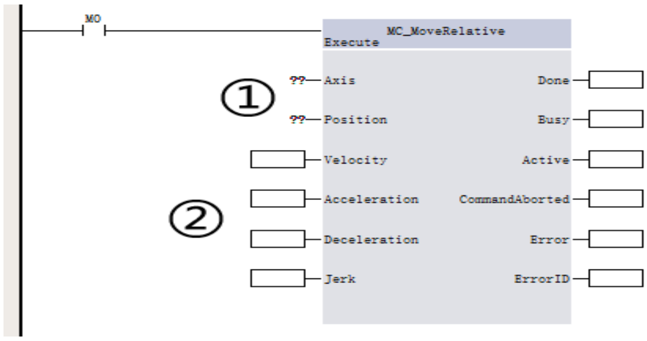
Some instructions support graphic block programming, and graphic block instructions are composed of instruction name, energy flow signal, input side and output side. Take the motion control axis graphic block instruction as an example, and its specific composition is shown in the following figure.



The floating-point numbers such as target position and target speed in the instruction are of single-precision floating-point format, so the value of the instruction should conform to the range and precision of single-precision floating-point format when processing in PLC program, that is, its numerical range is -3.4E38–3.4E38, and the precision is 7 significant digits. If the significant digits of a certain number exceed 7 digits, the extra digits will be automatically rounded.

3.6.2 Graphic Block Instruction Programming

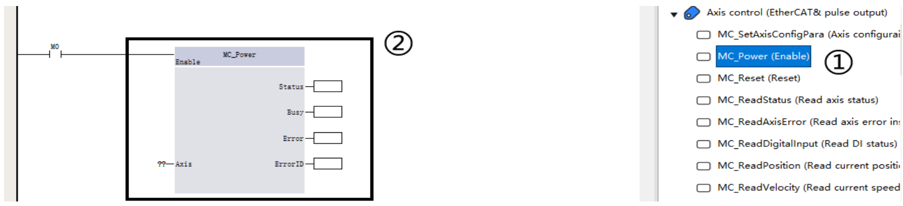
1. When programming, users need to drag the graphic block instruction from the instruction tree on the right side of the upper computer to the left, and then add the graphic block instruction to the program network. The graphic block supports direct editing of instruction parameters. As shown in the following figure:



Input parameters in the graphic block instruction to complete the editing of the graphic block instruction.

- ① "???" means parameters are required.
- ② You can choose whether to use parameters. If no parameters are used, the instruction input is default to the parameter values automatically, and the instruction state cannot be obtained when the instruction output is in the program or monitored and debugged.

2. All instructions under the toolbox instruction set node are in graphic block mode. When programming, you can directly double-click the instructions under the toolbox instruction set node and add the instructions to the current focus position of the ladder diagram, as shown in the following figure:



- ① Double-click the instruction to add it to the ladder diagram
- ② The instruction is added successfully

3.7 Library Functions

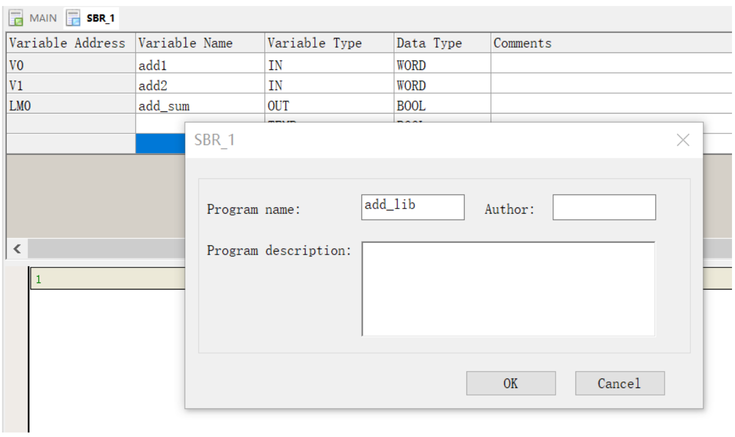
Library functions encapsulate reusable program blocks of a program into a universal library, and the program blocks in the encapsulated library can be repeatedly called in the program of different projects.

Using encapsulated libraries in programming enhances program development efficiency, reduces programming errors, and improves program quality.

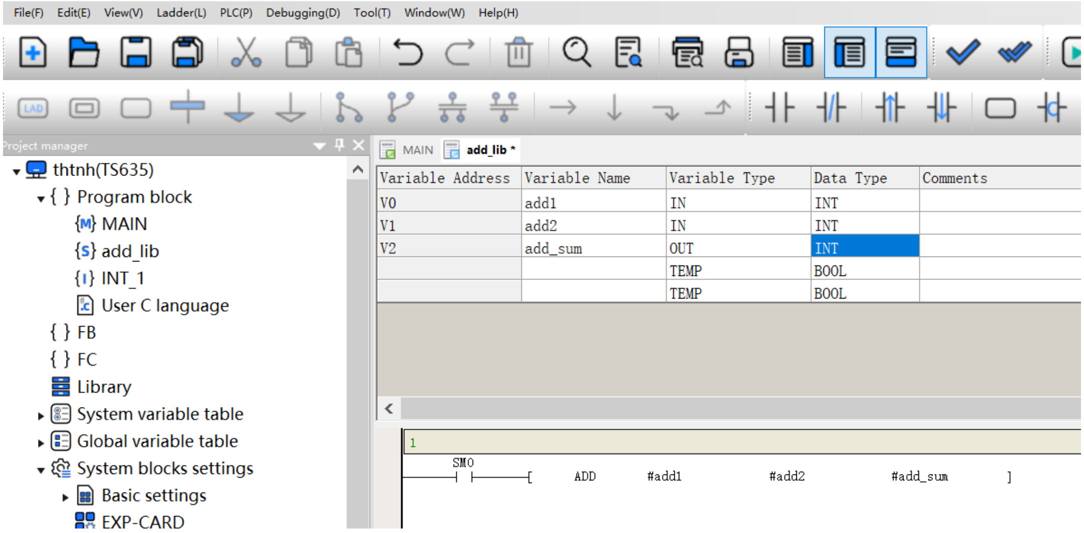
The basic step to use library functions is as follows: Choose "Create a library project to be encapsulated" → "Write the program" -> "Export a library file" → "Create current project" → "Import the library file".

3.7.1 Example of Exporting an Addition Library

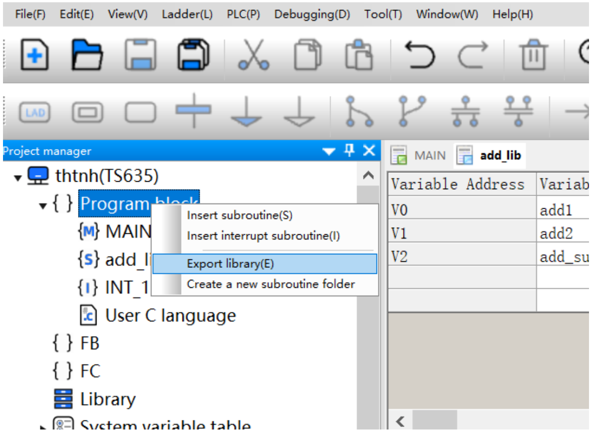
1. Create a new addition library project and then a new subroutine Add_lib.



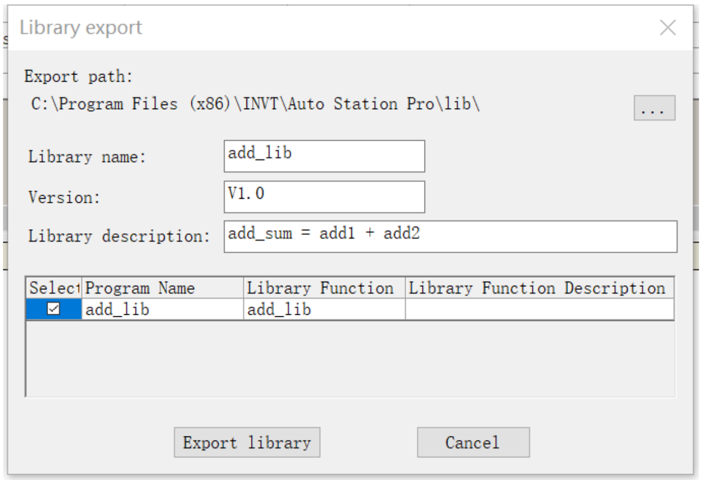
2. Write the input variables "add1" and "add2", then the output variable "add_Sum", and finally the core code.



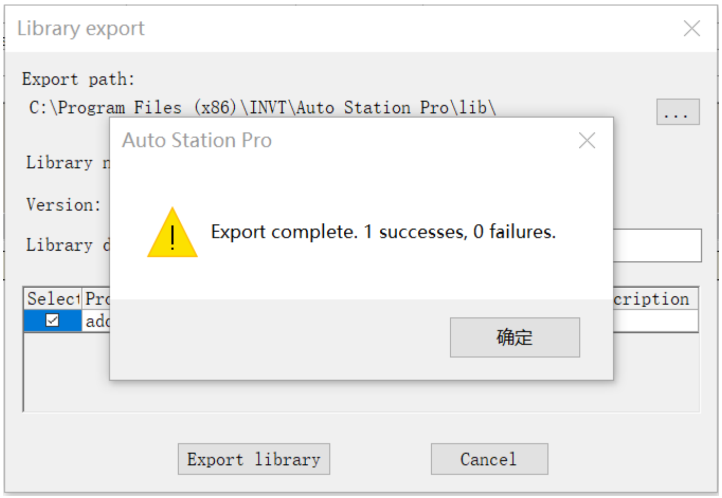
3. Right click on "Program block" in the "Project manager"interface, and select "Export library".



4. Select the save path, fill in the "Library name", "Version", and "Library description" fields, select the subroutine to save, and click"Export library".

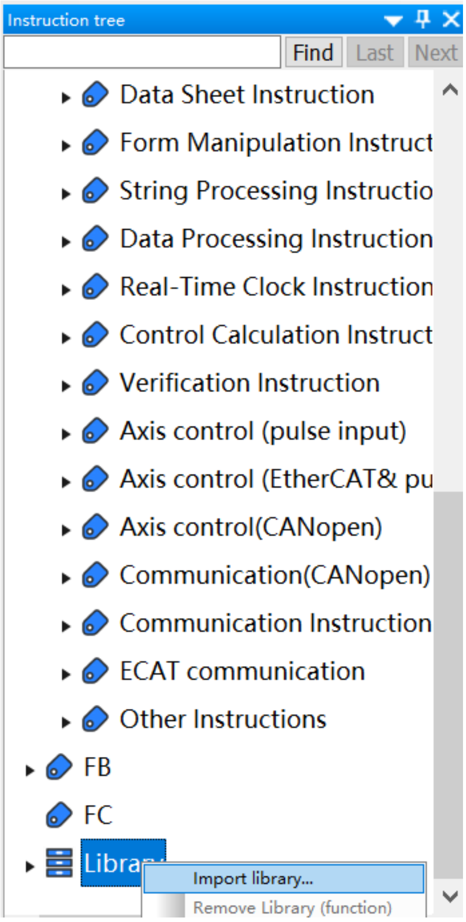


5. The interface of export success is shown as follows.

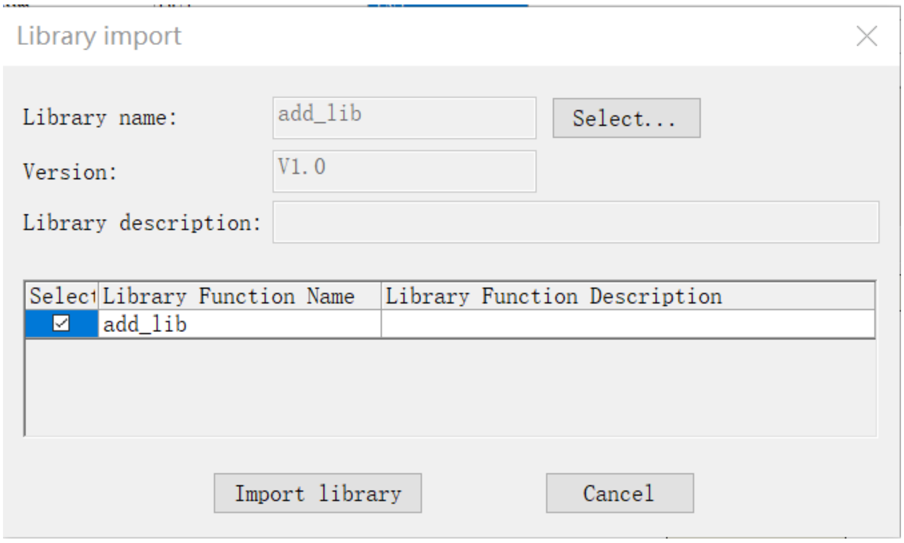


3.7.2 Example of Importing An Addition Library

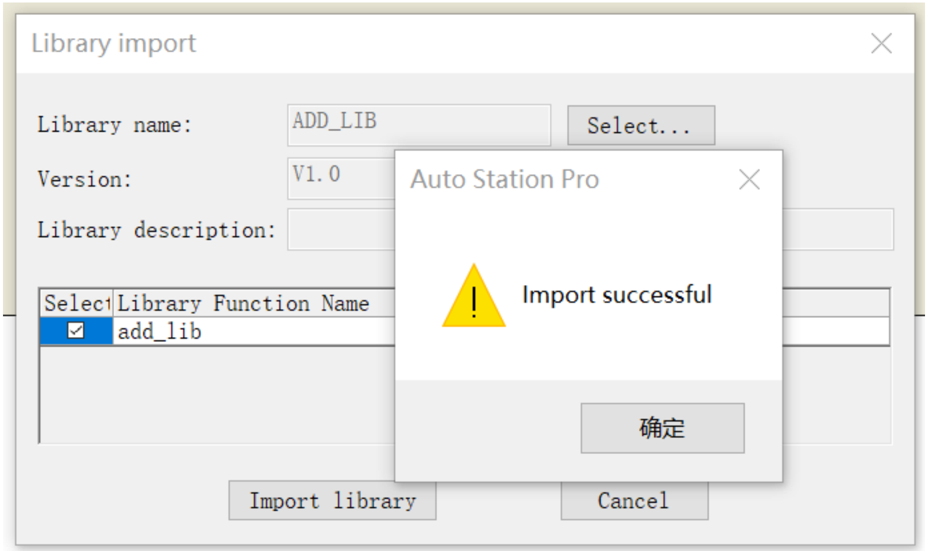
1. Open the project that requires the use of library functions, right click on "Library" at the bottom of the "Instruction tree" interface, and select "Import library".



2. Select the library save path (which is the lib file under the installation path of the upper computer software by default), then select the corresponding library, load library function, and finally click "Import library".



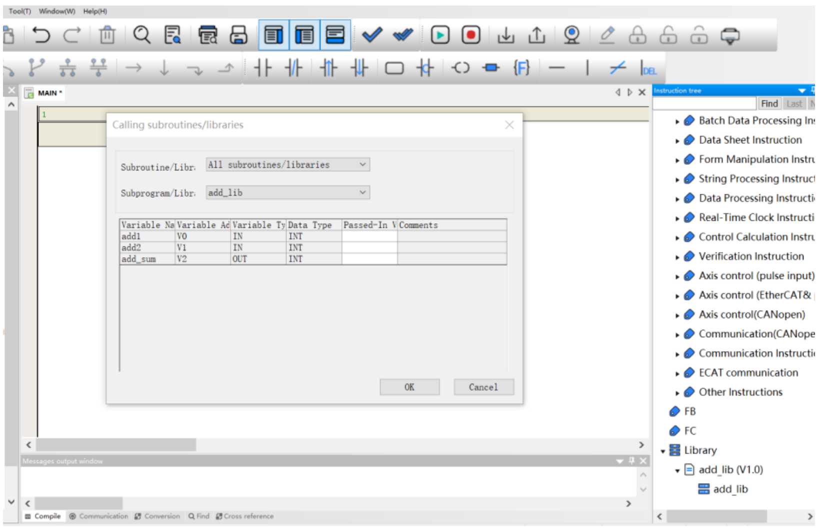
3. The interface of import success is shown as follows.



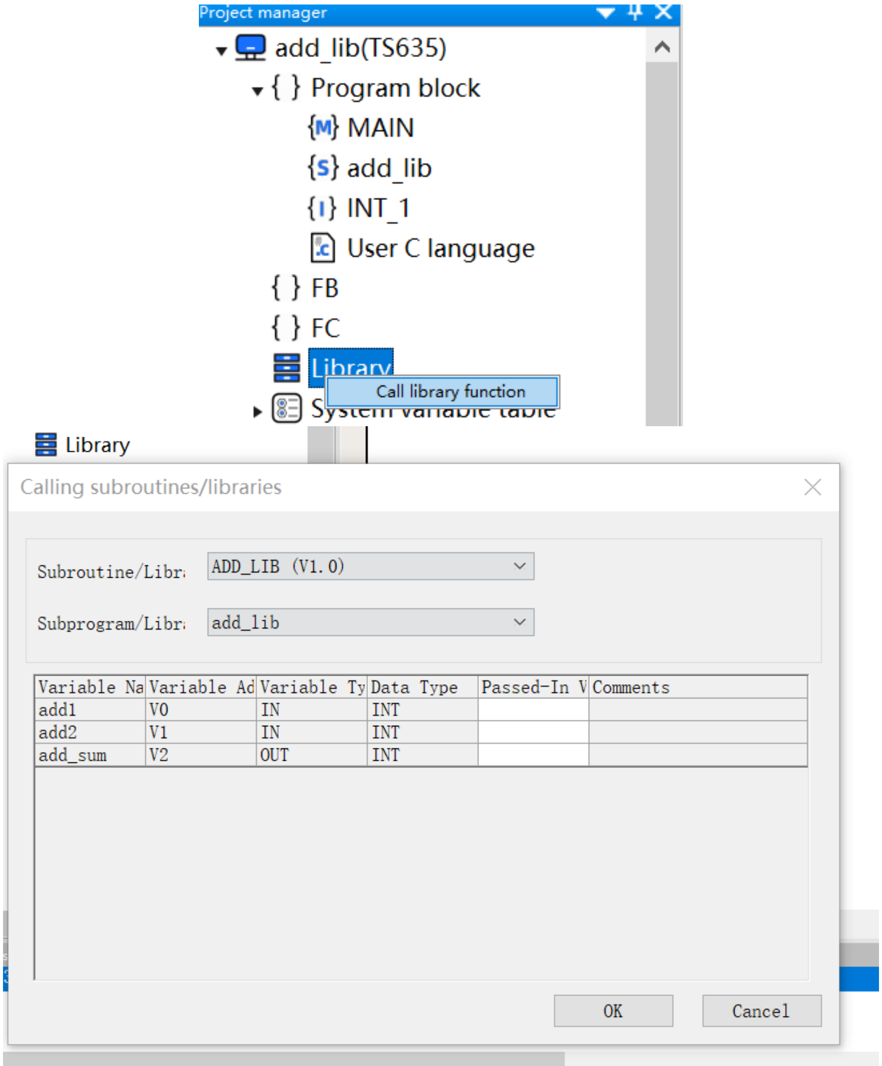
3.7.3 Call Library Functions

There are two ways to call library programs:

Method 1: In the "Instruction tree" interface, select the newly imported library program, double click on the functional program or drag it into the project, fill in the value passed in, and click "OK".

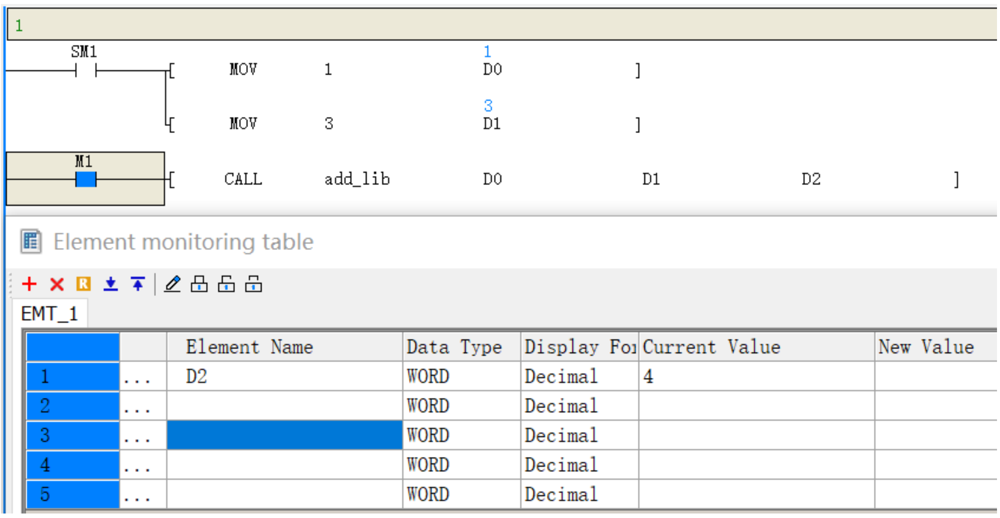


Method 2: In the "Project manager" interface, right click on "Library", select "Call library function", select "Subroutine/Libr" and "Subprogram/Libr", fill in the "Passed-In Value" field, and click "OK".



Using either of the above two methods generates the following ladder diagram code, which indicates a successful library import.

If D0=1 and D1=3 are assigned, D2=4 can be obtained, as shown in the figure below.



Note: To update the library, you need to delete original library functions, including those under "Library" in "Project manager" and "Instruction tree", and then re-import the library file.

3.8 System Variables

3.8.1 Overview

System variables are registers used to express and change PLC running status information, such as device model, version number, serial port, Ethernet and CAN communication, etc.

3.8.2 List of System Variables

Table 3-1 System Variable Information

Category of system variables	Description
_SYS_CAN	Information related to CAN communication, such as node number, baud rate, slave on-line status, etc
_SYS_COM	Serial communication related information, such as node number, baud rate, slave node online status, etc.
_SYS_ECAT	EtherCAT master and slave node status information
_SYS_ETHERNET	Ethernet communication information, such as IP, MAC, online status, error diagnosis, etc.
_SYS_INFO	PLC system information, such as SN number, firmware version, RTC clock, module diagnosis, system log, etc.
_SYS_AXSI_ENC_INFO	Encoder axis information
_SYS_AXIS_MC_INFO	Motion control axis struct
_SYS_GROUP_MC_INFO	Axis group
_SYS_CAM_TABLE	Cam table

3.8.3 _SYS_CAN CAN Interface Running Information

Table 3-2 _stru_SYS_CAN Interface Information

Name	Data Type	Description	R/W type
_sCAN.BaudRate	INT	Baud rate (kbps)	R
_sCAN.LoadRate	INT	Load rate (%)	R
_sCAN.RxPerSec	INT	Frames received per second (FPS)	R
_sCAN.TxPerSec	INT	Frames transmitted per second (FPS)	R
_sCAN.RxErrCnt	INT	Receive error counter	R
_sCAN.TxErrCnt	INT	Transmit error counter	R
_sCAN.Protocol	INT	Communication protocol	R

Table 3-3 _sCANOpen Interface Information

Name	Data Type	Description	R/W type
_sCANOpen.NodeID	INT	Node ID	R
_sCANOpen.NodeState	INT	Node status, 1 for online and 0 for offline	R
_sCANOpen.sEmcy	_stru_CANOpen_EMCY	Emergency	R

Name	Data Type	Description	R/W type
_sCANOpen.sDebug	_stru_CANOpen_DEBUG	Commissioning interface	R
_sCANOpen.sCfgErr	_stru_CANOpen_CFG_ERR	Configuration error message	R
_sCANOpen.sEmcy.NodeID	INT	Emergency Node ID	R
_sCANOpen.sEmcy.ErrorCode	INT	Emergency error code	R
_sCANOpen.sEmcy.RegAndMsErrField	INT	Error register pre-manufacturer custom error message area	R
_sCANOpen.sDebug.NodeID	INT	Debug node ID	R
_sCANOpen.sDebug.State	INT	Debug status	R
_sCANOpen.sDebug.Index	INT	Debug primary index	R
_sCANOpen.sDebug.SubIndexAndSize	INT	Debug sub-index and data size	R
_sCANOpen.sDebug.Data	INT	Debug data or error code	R
_sCANOpen.sCfgErr.NodeID	INT	Configuration error message node ID	R
_sCANOpen.sCfgErr.ConfigIndex	INT	Configuration number	R
_sCANOpen.sCfgErr.ErrorCode	DINT	Error code	R

3.8.4 _SYS_COM Serial Port Running Information

Table 3-4 sCOMx_485 Serial Port Information

Name	Data Type	Description	R/W type
_sCOMx_485.BaudRate	DINT	Baud rate	R
_sCOMx_485.DataBits	INT	Data bit	R
_sCOMx_485.Parity	INT	Check bit	R
_sCOMx_485.StopBits	INT	Stop bit	R
_sCOMx_485.Interface	INT	Physical interface	R
_sCOMx_485.Protocol	INT	Communication protocol	R
_sCOMx_485.Reserved	INT	Reserved	R

Table 3-5 _stru_COMFreePortx Serial Freeport Protocol Information

Name	Data Type	Description	R/W type
_sFreex.Sent	DINT	Number of bytes transmitted	R
_sFreex.Received	DINT	Maximum timeout (ms)	R
_sFreex.Timeout	DINT	Number of bytes transmitted	R
_sFreex.Sendlen	INT	Transmit data buffer	R
_sFreex.Sendbuf	INT	Receive data buffer	R
_sFreex.Recvbuf	INT	Number of bytes received	R
_sFreex.Recvlen	INT	Enabled state	R
_sFreex.Enable	BOOL	Activated status	R
_sFreex.Activate	BOOL	Busy state	R
_sFreex.Busy	BOOL	Completion sign	R
_sFreex.Done	BOOL	Error sign	R
_sFreex.Error	BOOL	Reserved	R

Table 3-6 _sMbMstx Serial Port Modbus RTU/ASCII Master Node Information

Name	Data Type	Description	R/W type
_sMbMstx.AddrNum	INT	Number of nodes	R
_sMbMstx.TimeOut	INT	Maximum timeout (ms)	R
_sMbMstx.ResponseTime	INT	Response time (ms)	R
_sMbMstx.Connected	BOOL	Number of connections	R
_sMbMstx.Enable	BOOL	Enabled state	R
_sMbMstx.Activate	BOOL	Activated status	R
_sMbMstx.Busy	BOOL	Busy state	R
_sMbMstx.Done	BOOL	Port Modbus communication completion flag bit	R
_sMbMstx.Error	BOOL	Port Modbus communication error flag bit	R
_sMbMstx.ErrSlID	BOOL	Slave node ID number	R

Table 3-7 _sMbMST_MSGx Serial Port Modbus RTU/ASCII Master Connection — Slave Information

Name	Data Type	Description	R/W type
_sMbMstMsgx.DisableSlv	BOOL	Slave node disabled or not	R
_sMbMstMsgx.IsSlvDisable	BOOL	Slave disability flag	R

Table 3-8 _sMbSlvx Serial Port Modbus RTU/ASCII Master Information

Name	Data Type	Description	R/W type
_sMbSlvx.SlvID	INT	Number of nodes	R
_sMbSlvx. Enable	BOOL	Enabled state	R
_sMbSlvx. Activate	BOOL	Activated status	R
_sMbSlvx. Busy	BOOL	Busy state	R
_sMbSlvx. Done	BOOL	Port Modbus communication completion flag bit	R
_sMbSlvx. Error	BOOL	Port Modbus communication error flag bit	R

Table 3-9 _stru_COMNnbusx Serial Port Modbus RTU/ASCII Slave Information

Name	Data Type	Description	R/W type
_sNNBusx.SlvId	INT	Node number	R
_sNNBusx.Delay	INT	N:N additional delay	R
_sNNBusx.RetryTimes	INT	Retry times	R
_sNNBusx.Mode	INT	N:N network refresh mode	R
_sNNBusx.Period	DINT	N:N Polling period of communication	R
_sNNBusx.Error	DINT	Communication error flag bit0–bit31 represent error flag bits for nodes numbered 0–31, respectively 1: Error, 0: No Error	R

3.8.5 _SYS_ECAT EtherCAT Running Status Information

Table 3-10 EtherCAT Master Status Information

Name	Data Type	Description	R/W type
_sECATMst.MasterRunState	BOOL	Master running status flag bit ON: Run, OFF: Stop	R
_sECATMst.LinkState	BOOL	Physical connection status of master ON: normal, OFF: network cable disconnected	R
_sECATMst.HeartBeat	BOOL	EtherCAT real-time task heartbeat	R
_sECATMst.BlockHeartBeat	BOOL	EtherCAT non-real-time task heartbeat	R
_sECATMst.MaxCycleTime	DINT	Maximum cycle time, µs	R
_sECATMst.MinCycleTime	DINT	Minimum cycle time, µs	R
_sECATMst.CycleTime	DINT	Cycle time, µs	R
_sECATMst.MaxExeTime	DINT	Maximum execution time, µs	R
_sECATMst.MinExeTime	DINT	Minimum execution time, µs	R
_sECATMst.ExeTime	DINT	Execution time, µs	R
_sECATMst.Tx_frames	DINT	Total frames sent	R
_sECATMst.Rx_frames	DINT	Total frames received	R
_sECATMst.Tx_frame_rates	DINT	Frame rate at which the data is transmitted, frames/second	R
_sECATMst.Rx_frame_rates	DINT	Frame rate at which the data is received, frames/second	R
_sECATMst.Tx_bytes_rate	DINT	The speed at which the byte is transmitted, bytes/second	R
_sECATMst.Rx_bytes_rate	DINT	The speed at which the byte is received, bytes/second	R
_sECATMst.Loss_rate	DINT	Lost EtherCAT data frame, in frames	R
_sECATMst.ResetTime	BOOL	Reset execution time and cycle time	R/W
_sECATMst.StartMaster	BOOL	Start the master	R/W
_sECATMst.StopMaster	BOOL	Stop the master	R/W
_sECATMst.ClearFrameCounter	BOOL	Reset transmit and receive data frame counter	R/W
_sECATMst.DisableMaster	BOOL	Disable Master Enable	R/W
_sECATMst.SlaveState	INT	Status of all slaves, 1: All slaves are online 0: Some slaves are not online	R
_sECATMst.FirstErrorSlave	INT	First faulty slave	R
_sECATMst.LibVersion	DINT	EtherCAT library version	R
_sECATMst.MstVersion	DINT	EtherCAT master version	R
_sECATMst.DriveVersion	DINT	EtherCAT NIC driver version	R
_sECATMst.Tx_error_cnt	DINT	EtherCAT transmit error count	R
_sECATMst.Rx_timeout_cnt	DINT	EtherCAT receive frame timeout count	R

Name	Data Type	Description	R/W type
_sECATMst.Tx_corrupt_cnt	DINT	EtherCAT receive invalid frame count	R
_sECATMst.Tx_unmach_cnt	DINT	EtherCAT receive unmatched frame count	R
_sECATMst.RxPDOLength	DINT	EtherCAT total receive PDOs	R
_sECATMst.TxPDOLength	DINT	EtherCAT total transmit PDOs	R
_sECATMst.ConfigureState	DINT	EtherCAT configuration status	R
_sECATMst.Delay	DINT	EtherCAT synchronizer	R
_sECATMst.SlvLinkState	INT	Connection status of all slave	R
_sECATMst.DisableState	INT	Master disability state	R

Table 3-11 EtherCAT Slave Status Information

Name	Data Type	Description	R/W type
_sECATSlv.Unused	BOOL	System retention	R
_sECATSlv.SlaveRunState	BOOL	Slave running status ON: Run, OFF: Stop	R
_sECATSlv.SetAliasState	BOOL	Alias written to slave: ON means busy	R
_sECATSlv.SetAliasError	BOOL	Failed to write alias to slave	R
_sECATSlv.MatchState	BOOL	Slave type mismatch	R
_sECATSlv.ConfigError	BOOL	Slave configuration error	R
_sECATSlv.SetAlias	BOOL	Set slave alias, rising edge is valid	R/W
_sECATSlv.DisableEnable	BOOL	Disable slave enable	R/W
_sECATSlv.ALState	INT	EtherCAT state machine status	R
_sECATSlv.ALCode	INT	Fault code	R
_sECATSlv.ActAlias	INT	Actual node alias	R
_sECATSlv.TarAlias	INT	Target alias to write	R/W
_sECATSlv.StationAddress	INT	Actual node name	R
_sECATSlv.SlaveRingPos	INT	Configuration address	R
_sECATSlv.SDOErrorCode	INT	Startup parameter configuration error count	R
_sECATSlv.CfgErrorCode	DINT	Configuration error code	R
_sECATSlv.DisableState	INT	Configuration state	R

3.8.6 _SYS_ETHERNET Ethernet Information

Table 3-12 _stru_ETH_NET Port Information

Name	Data Type	Description	R/W type
_sENETx.MAC	INT	Physical address	R
_sENETx.IP	DINT	Native IP address	R/W
_sENETx.NetMask	DINT	Subnet mask	R/W
_sENETx.GateWay	DINT	Gateway	R/W
_sENETx.CmdCtrl	INT	CW 0: Read 1: Edit	R/W

Name	Data Type	Description	R/W type
		2: Write 3: Error.	

Note: You can monitor the network information of this machine in the variable table, and you can also modify IP, Subnet mask, gateway and other information in the running state.

Table 3-13 _stru_MBTCP_MSTx ModbusTCP Master Information

Name	Data Type	Description	R/W type
_sMbTcpMstx.SlvIP	DINT	IP address (of slave)	R
_sMbTcpMstx. SlvPort	DINT	Port number (of slave)	R
_sMbTcpMstx.Timeout	INT	Connection timeout (ms)	R
_sMbTcpMstx.ResponseTime	INT	Response time	R
_sMbTcpMstx.Connected	BOOL	Connection flag	R
_sMbTcpMstx.Enable	BOOL	Enabled state	R
_sMbTcpMstx.Activate	BOOL	Activated status	R
_sMbTcpMstx.Busy	BOOL	Busy state	R
_sMbTcpMstx.Done	BOOL	Port Modbus communication completion flag bit	R
_sMbTcpMstx.Error	BOOL	Port Modbus communication error flag bit	R

Table 3-14 _stru_MBTCP_MST_MSGx ModbusTCP Slave Information

Name	Data Type	Description	R/W type
_sMbTcpMstMsgx.MstIP	DINT	Native IP address	R
_sMbTcpMstMsgx.MstPort	DINT	Port number	R
_sMbTcpMstMsgx.DisableSlv	BOOL	Slave node disabled or not	R
_sMbTcpMstMsgx.IsSlvDisable	BOOL	Slave disability flag	R

Table 3-15 _stru_MBTCP_SLVx ModbusTCP Master Connection — Slave Information

Name	Data Type	Description	R/W type
_sMbTcpSlvx.Connections	INT	Number of connections	R
_sMbTcpSlvx.MstIP	DINT	Master IP address table	R
_sMbTcpSlvx.MstPort	DINT	Master port number table	R
_sMbTcpSlvx.SlvIP	DINT	Slave node IP address	R
_sMbTcpSlvx.SlvPort	DINT	Slave node port number	R
_sMbTcpSlvx. SlvID	INT	Slave node ID	R
_sMbTcpSlvx.Connected	BOOL	Connection flag of corresponding node	R
_sMbTcpSlvx.Enable	BOOL	Enabled state	R
_sMbTcpSlvx.Error	BOOL	Communication error flag bit	R
_sMbTcpSlvx.ErrIP	DINT	IP address of master node with error	R
_sMbTcpSlvx.ErrPort	DINT	Port number of master node with error	R

Table 3-16 _stru_IOT_CARD 4G IoT Card Information

Name	**Data Type**	**Description**	**R/W type**
_sIotCard.ModemState	INT	Dialing status of 4G module 0: Initialization 1: No port, port read-write error. 2: No SIM card inserted. 3: SIM card has no data flow, APN error, etc. 4: Abnormal signal strength 5: Dialing activation failed 6: Dialed successfully	R
_sIotCard.MqttState	INT	MQTT server connection status 0: Not connected 1: Connected	R
_sIotCard.CSQ	DINT	Signal strength	R
_sIotCard.Reserved	DWORD	Reserved	R

Table 3-17 _stru_LABEL_COM Tag Communication Information

Name	Data Type	Description	R/W type
_sLabelCom.IP	DINT	Tag communication IP address	R
_sLabelCom.PORT	DINT	Tag communication port number	R
_sLabelCom. Reserved	DWORD	Reserved	R

3.8.7 _SYS_INFO PLC Running Information

- Get PLC production device information

Table 3-18 DevInfo Device Information

Name	**Data Type**	**Description**	**R/W type**
_sDevInfo.Device	INT	Device Model ID	R
_sDevInfo.Vender	INT	Manufacturer ID	R
_sDevInfo.HWVersion	DINT	Hardware version	R
_sDevInfo.SWVersion	DINT	Software version	R
_sDevInfo.FPGAVersion	DINT	FPGA version	R
_sDevInfo.BattVolt	DINT	Battery voltage	R

- Get CPU and memory utilization and diagnose CPU performance.

Table 3-19 OSM System Monitor

Name	**Data Type**	**Description**	**R/W type**
_sOSM.CPU	INT	cpu utilization	R
_sOSM.Memory	INT	Memory usage	R

- Obtain the execution cycle time of programs and tasks, so as to judge the complexity of program execution logic.

Table 3-20 Program User Program Information

Name	Data Type	Description	R/W type
_sProgram.TotalSize	DINT	Total program capacity	R
_sProgram.UsedSize	DINT	Used program capacity	R
_sProgram.CurRunTime	DINT	Current program runtime (μs)	R
_sProgram.MinRunTime	DINT	Minimum program runtime (μs)	R
_sProgram.MaxRunTime	DINT	Maximum program runtime (μs)	R
_sProgram.AveRunTime	DINT	Average program runtime (μs)	R
_sProgram.ConstScanTime	DINT	Constant scan time (μS)	R
_sProgram.WDT	DINT	Watchdog reset time (s)	R
_sProgram.Reset	BOOL	Reset cycle time	R/W

- The error log information of PLC is recorded.

Table 3-21 CurErrLst Error Message List

Name	Data Type	Description	R/W type
_sCurErrLst.Quantity	DINT	Current error quantity	R
_sCurErrLst.ErrInfo	_stru_ERR_INFO	Current error message list	R
_sCurErrLst.sErrInfo.SubErrorCode	INT	Sub-error code	R
_sCurErrLst.sErrInfo.MainErrorCode	INT	Main error code	R
_sCurErrLst.sErrInfo.TimStamp	DINT	Time stamp	R

- Get RTC clock

Table 3-22 RTC Clock

Name	Data Type	Description	R/W type
_sDataTime.Second	INT	Second	R
_sDataTime.Minute	INT	Minute	R
_sDataTime.Hour	INT	Hour	R
_sDataTime.Day	INT	Day	R
_sDataTime.Month	INT	Month	R
_sDataTime.Year	INT	Year	R
_sDataTime.WeekDay	INT	Week	R
_sDataTime.YearDay	INT	Days	R
_sDataTime.Timestamp	DINT	Total seconds	R

Table 3-23 UsrIntCtl Interrupt Enable Control

Name	Data Type	Description	R/W type
_sUsrIntCtl[0]	_UsrIntCtl[67]	-	-
_sUsrIntCtl[0].Enable	BOOL	Enable control bit	R
_sUsrIntCtl[0].IntID	INT	Interrupt program ID	R
_sUsrIntCtl[…]	_UsrIntCtl[67]	-	-

Table 3-24 ExtModule Expansion Module System Variable Related Information

Name	Data Type	Description	R/W type
_sExtModule.CfgNum	INT	User-configured module number	R
_sExtModule.ActNum	INT	Actually mounted module number	R
_sExtModule.Res_Align	DINT	Reserved for byte alignment	R
_sExtModule.ExtSlot	mExtSlot	–	R
_sExtModule.sExtSlot[n].CfgType	INT	nth expansion module User-configured type	R
_sExtModule.sExtSlot[n].ActType	INT	nth expansion module Type of actual mount	R
_sExtModule.sExtSlot[n].Error	BOOL	nth expansion module Error Status	R
_sExtModule.sExtSlot[n].Disable	BOOL	nth expansion module Module disabled	R
_sExtModule.sExtSlot[n].Res_Align	INT	Reserved for byte alignment	R
_sExtModule.sExtSlot[n].SWVersion	DINT	nth expansion module Software version	R
_sExtModule.sExtSlot[n].LGVersion	DINT	nth expansion module Logic device version	R

Table 3-25 ExtCard Extension Card Related Information

Name	Data Type	Description	R/W type
_sExtCard.mCfgType	INT	User-configured module type	R
_sExtCard.mActType	INT	Type of actually mounted module	R
_sExtCard.SWVersion	DINT	Software version	R
_sExtCard.LGVersion	DINT	Logic device version	R
_sExtCard.Error	BOOL	Error state	R
_sExtCard.Disable	BOOL	Module disabled	R

Table 3-26 AlarmInfo Alarm Information and Control Bits

Name	Data Type	Description	R/W type
_sAlarmInfo.Alarm_Enable	BOOL	Alarm enabled	R/W
_sAlarmInfo.Alarm_Act_Flg	BOOL	S900 – S999 alarm action flag	R
_sAlarmInfo.Alarm_Min_Num	INT	S900–S999 minimum alarm action element number	R

Table 3-27 SM System Variables

Name	Data Type	Description	R/W type
SM0	BOOL	Running monitoring bit	R
SM1	BOOL	Initial running pulse bit	R
SM2	BOOL	Power-on flag bit	R
SM3	BOOL	Error flag bit	R
SM10	BOOL	Clock oscillation with a cycle of 10ms	R

Name	Data Type	Description	R/W type
SM11	BOOL	Clock oscillation with a cycle of 100ms	R
SM12	BOOL	Clock oscillation with a cycle of 1ms	R
SM13	BOOL	Clock oscillation with a cycle of 1min	R
SM14	BOOL	Clock oscillation with a cycle of 1hour	R
SM15	BOOL	Scanning cycle oscillation bit	R
SM18	BOOL	Operation zero flag	R
SM19	BOOL	Operation borrow flag	R
SM20	BOOL	Operation carry flag	R
SM22	BOOL	Set when instruction executes incorrectly	R
SM23	BOOL	Set when instruction element number subscript overflows	R
SM24	BOOL	Set when instruction parameter is illegal	R
SM30	BOOL	Multi-cycle instruction completion flag bit	R
SM31	BOOL	BINDA instruction output character flag bit	R/W
SM32	BOOL	ATI/ITA/ASC/CCITT/CRC16/LRC/CCD instruction bit processing mode flag bit	R/W
SM33	BOOL	SORTR/SORTC instruction descending sort enabled	R/W
SM34	BOOL	SMOV instruction data format setting bit	R/W
SM35	BOOL	The comparison results of BKCMP instruction matrix are all flagged with 1	R

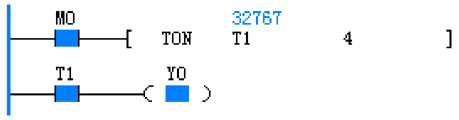
3.9 Timer

3.9.1 TON

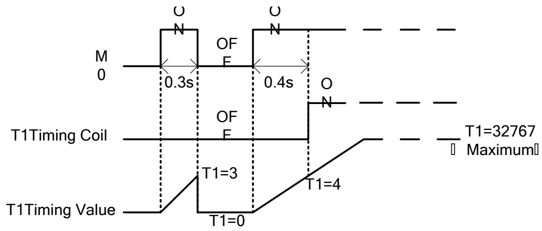
3.9.1.1 Function Description

1. When the energy flow is valid and the timing value is less than 32767, the specified T element (D) times (the timing value accumulates with the travel time). When the timing value reaches 32767, the timing value will remain unchanged at 32767.
2. When the timing value is greater than or equal to the preset value (S), the timing coil output of the specified T element is ON.
3. When the energy flow is OFF, timing stops, the timing value is reset to zero, and the timing coil output is OFF.
4. When the system executes this instruction for the first time, the timing coil value of the specified T element will be reset to OFF and the timing value will be reset to zero.

3.9.1.2 Application Example



3.9.1.3 Sample Sequence Chart

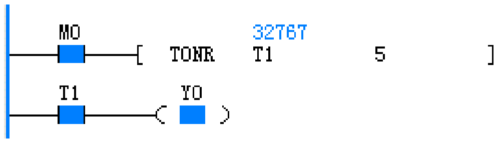


3.9.2 TONR

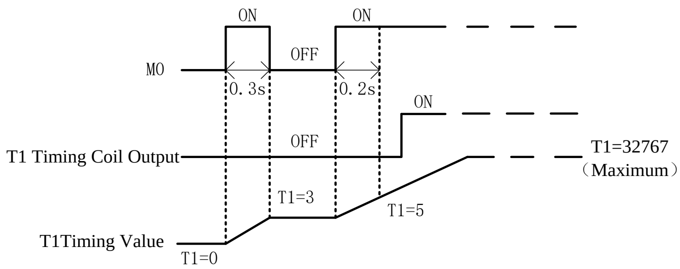
3.9.2.1 Function Description

1. When the energy flow is valid and the timing value is less than 32767, the specified T element (D) times, and the timing value increases with the travel time. When the timing value reaches 32767, the timing value will remain unchanged at 32767.
2. When the timing value is greater than or equal to the preset value (S), the timing coil output of the specified T element is ON.
3. When the energy flow is OFF, timing stops, and the timing coil and the timing value remain the current timing value.

3.9.2.2 Application Example



3.9.2.3 Sample Sequence Chart

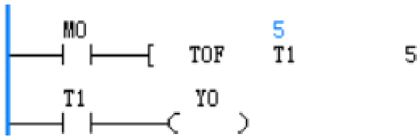


3.9.3 TOF

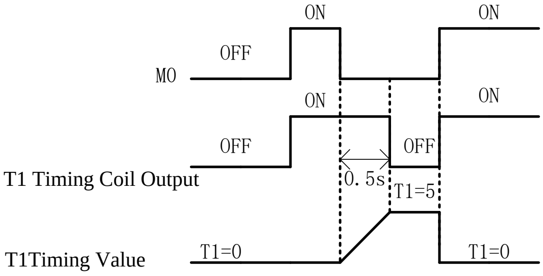
3.9.3.1 Function Description

1. When the energy flow changes from ON → OFF (falling edge), the designated timer T (D) starts timing.
2. When the energy flow is OFF, and the specified timer T has started timing, the timing is continued. Until the timing value is equal to the preset value (S), the timing coil output of the specified T element is OFF, then the timing value will remain at the preset value and will not change.
3. If the timing is not started, it will not be started even if the energy flow input is OFF.
4. When the energy flow is ON, timing stops, the timing value is reset to zero, and the timing coil output is ON.

3.9.3.2 Application Example



3.9.3.3 Sample Sequence Chart

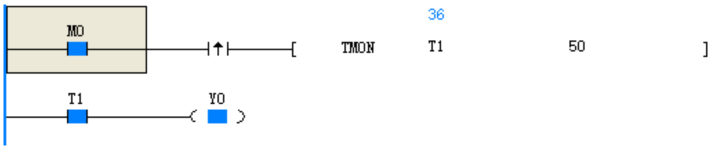


3.9.4 TMON

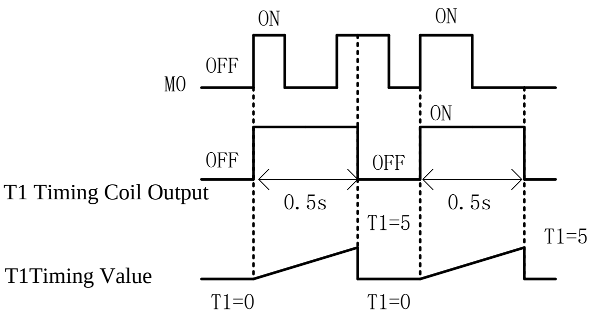
3.9.4.1 Function Description

1. When the input energy flow changes from OFF → ON (rising edge) and is in an untimed state, the specified timer T (D) starts timing (starting from the current value), the timing coil output is kept ON under the timing state (the length of the timing state is determined by S).
2. In the timing state (the timing length is determined by S), no matter how the energy flow changes, the timing is kept, and the timing coil output is kept ON.
3. When the timing value is reached, timing stops, the timing value is reset to zero, and the coil output is reset to OFF.

3.9.4.2 Application Example



3.9.4.3 Sample Sequence Chart



3.10 Function Block and Function (FB/FC)

Function block programs use ladder diagrams for programming. Within function block programs, functions (FCs) or function blocks (FBs) can be called, supporting up to six levels of nested calls.

3.10.1 Function Block (FB)

FB function blocks encapsulate reusable program blocks of a program into a universal function block, and the FB function blocks can be repeatedly called in the program of the same project.

Using encapsulated FB function blocks in programming enhances program development efficiency and improves program readability.

The FB function block can generate one or more values when executed, retains its own special internal variables, and the system allocates memory for the state variables within the FB function blocks. These internal variables constitute its own state characteristics. For input variable values with identical parameter values, different internal state variables may exist, leading to different computational results.

Before using the FB function blocks, it needs to be instantiated (by adding an independent FB instantiation table in the global variable table).

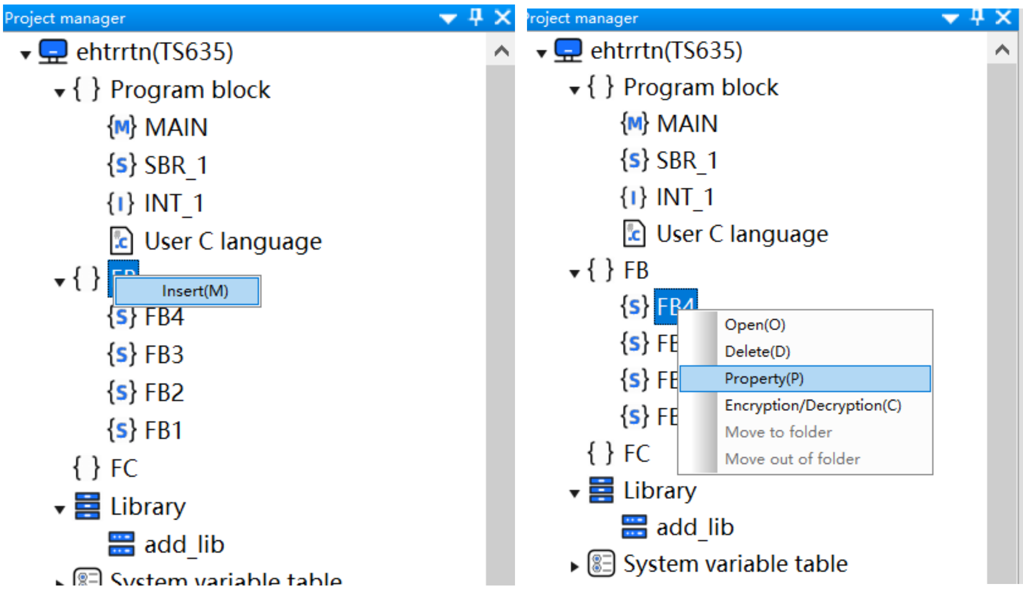
The basic steps for using the FB function block are: Create a function block > Function block programming> Function block instantiation> Run the function block.

3.10.1.1 Create a Function Block

You can create a new function block through the Auto Station Pro software.

Step 1 Right-click "{ }FB" under the corresponding project name node, select "Insert", and a new function block named FB1 is created.

Step 2 Right-click "FB1" to access relevant settings such as "Name", "Encryption", etc.



3.10.1.2 Function Block Programming

Double-click on the newly created function block under the "{ }FB" node to enter the function block program editing interface. The function block program editing interface has an additional window for input/output and local variable definitions compared to regular program editing. The parameters of an FB function block can be classified into Input (IN), Output (OUT), Input/Output (IN_OUT), and Local Variables (TEMP).

Note: In addition to using variables, the FB function block program can also use supported soft elements as global variables, such as D0.

The input/output and local variable definition window is shown below.

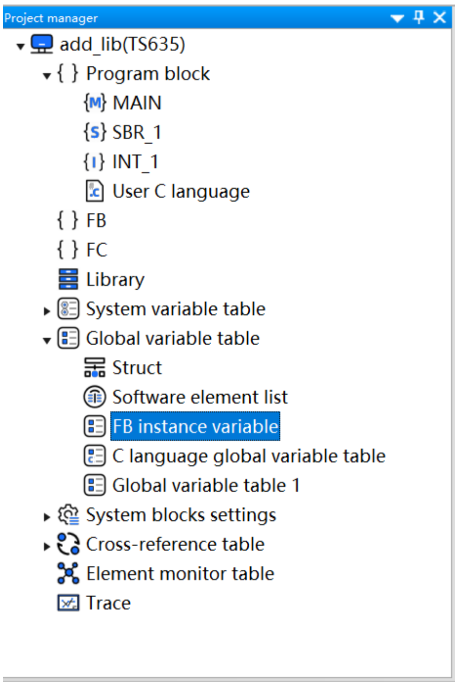
Variable Name	Variable Type	Data Type	Comments
add1	IN	INT	
add2	IN	INT	
add_sum	OUT	INT	
	TEMP	BOOL	
	TEMP	BOOL	

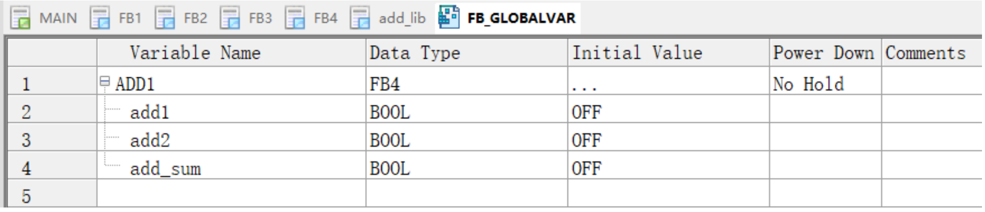
- Variable name: Customize the name of the variable.
- Variable type

Variable type	Category	Description
IN	Input variable	The parameters are provided by the logic block that calls input variables, and the input is the commands passed to the logic block, where IN type parameters will be generated on the left side of the graphic block as input parameters.
OUT	Output variable	Provide parameters to the logic block that calls output variables, indicating it outputs structured data from the logic block. OUT type parameters will be generated on the right side of the graphic block as output parameters.
INOUT	Input/Output variables	Input and output variables can not only be passed into the called logic block, but also can be modified within the called logic block. IN_OUT type parameters serve as input and output parameters, which can be generated on the left side, right side or the top of the graphic block.
TEMP	Local variables	Valid only within this logic block and cannot be accessed externally.

- Data types: Variable data types supported include BOOL, INT, DINT, WORD, DWORD, and REAL (other types will be developed iteratively with each version).
- Note: Used to annotate the meaning of variables.

After the FB function block is instantiated in the MAIN program, the variables defined in the function block will appear in the "FB instance variable" and can be set with initial values and power-down retention attributes.





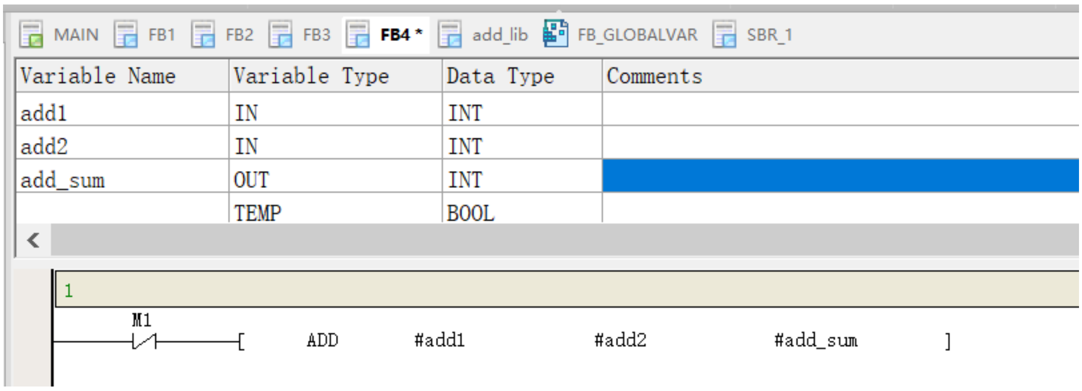
	Variable Name	Data Type	Initial Value	Power Down	Comments
1	ADD1	FB4	...	No Hold	
2	add1	BOOL	OFF		
3	add2	BOOL	OFF		
4	add_sum	BOOL	OFF		
5					

- Initial value: The initial data set when the variable is executed at the beginning.
- Power-down retention: The power-down retention attribute can be set to "Hold" or "Not hold" for variables.
  - Not hold: The variables are restored to the initial values set after repower-on.
  - Hold: Maintain the variable's previous running value after repower-on.

Note: If the option "Clear power-down retention data after download" is selected in the download option box, the variables will be restored to the initial values during program downloading.

3.10.1.3 Example of Encapsulating Integer Addition Using FB

1. **Editing the function blocks**



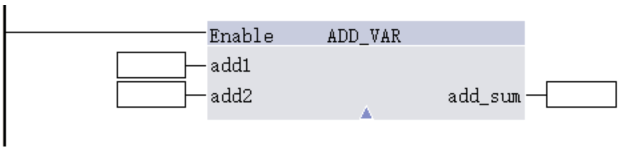
Variable Name	Variable Type	Data Type	Comments
add1	IN	INT	
add2	IN	INT	
add_sum	OUT	INT	
	TEMP	BOOL	

2. **Instantiating and calling the function blocks**

After writing the FB function block program, use it in the MAIN application program by instantiating and calling the function block.

There are three methods for instantiating function blocks:

Method 1: In the ladder diagram application, directly enter the FB name directly, then input the instance name in the "??" at the top of the function block instruction to complete the instantiation of the function block.



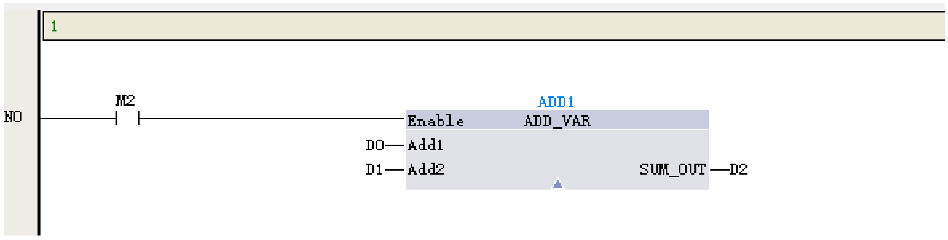
Method 2: In the ladder diagram application, directly enter the FB name + space + instance name, then click OK to complete the instantiation of the function block.

Method 3: Under the FB node in the toolbox, double-click the FB command to add the FB command to the selected position in the ladder diagram. After adding it, enter the instance name in the graphic block command to complete the instantiation.

After the function block is instantiated, edit the command parameters in the instantiated FB command according to the program requirements, and complete the instantiation call of the function block.

3. Running the function blocks

After the FB function block is instantiated, the "Enable" of the function block is connected to the power flow network of the ladder diagram. When the Enable network flow is valid (ON), the function block program is executed, and the output of the function block is refreshed based on the input state and internal variable state. When the Enable network flow is invalid (OFF), the function block program is not executed, and the output of the function block is not refreshed.



The ADD_VAR function block executes when the power flow condition is ON: D2 = D0 + D1.

4. Encapsulating function blocks (Support for future iterations)

The edited and debugged FB function blocks can be encapsulated into a library, which can be imported, exported, and managed through the library of the Auto Station Pro software, achieving reuse of the same library in different programs.

3.10.2 Function (FC)

A function (FC) is an independently encapsulated function, wherein the program block can define input/output type parameters. The main characteristic of a function is that its internal variables are static, without any internal state storage. The same input parameters will always yield the same output, which is the main difference between a function (FC) and a function block (FB). Functions (FC) are commonly used for various mathematical operations, such as sin(x) and cos(x), which are typical types of functions.

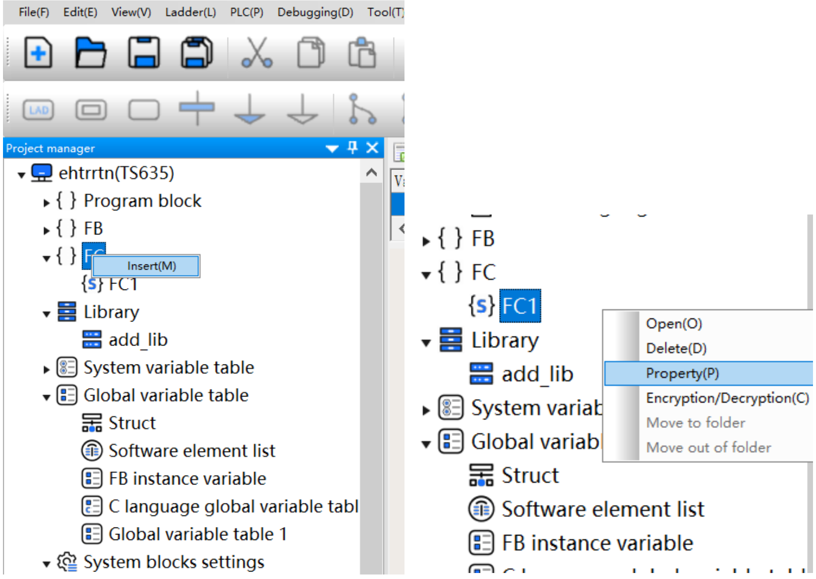
The basic steps for using a function are: create a function > program the function > call the function > run the function.

3.10.2.1 Creating a New Function

You can create a new function block through the Auto Station Pro software.

Step 1 Right-click the "{ } FC" under the project name node, select "Insert", and a new function block named "FC1" is created.

Step 2 Right-click "FC1" to access relevant settings such as "Name", "Encryption", etc.

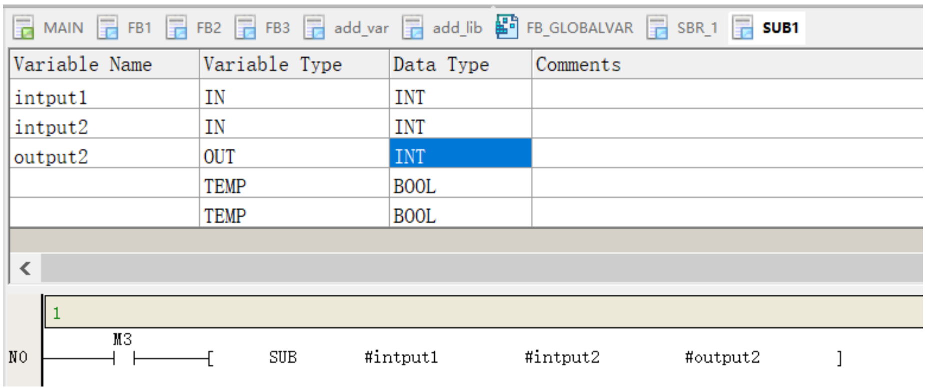


3.10.2.2 Function Programming

Double-click the newly created function under the "{ } FC" node to enter the function block program editing interface. The function block program editing interface has an additional window for input/output and local variable definitions compared to regular program editing. The parameters of an FC function block can be classified into Input (IN), Output (OUT), Input/Output (IN_OUT), and Local Variables (TEMP).

Note: In addition to using variables, the FC function block program can also use supported soft elements as global variables, such as D0.

The input/output and local variable definition window is shown below.



Variable Name	Variable Type	Data Type	Comments
intput1	IN	INT	
intput2	IN	INT	
output2	OUT	INT	
	TEMP	BOOL	
	TEMP	BOOL	

- Variable name: Customize the name of the variable.
- Variable type

Variable type	Category	Description
IN	Input variable	The parameters are provided by the logic block that calls input variables, and the input is the commands passed to the logic block, where IN type parameters will be generated on the left side of the graphic block as input parameters.
OUT	Output variable	Provide parameters to the logic block that calls output variables, indicating it outputs structured data from the logic block. OUT type parameters will be generated on the right side of the graphic block as output parameters.
INOUT	Input/Output variables	Input and output variables can not only be passed into the called logic block, but also can be modified within the called logic block. IN_OUT type parameters serve as input and

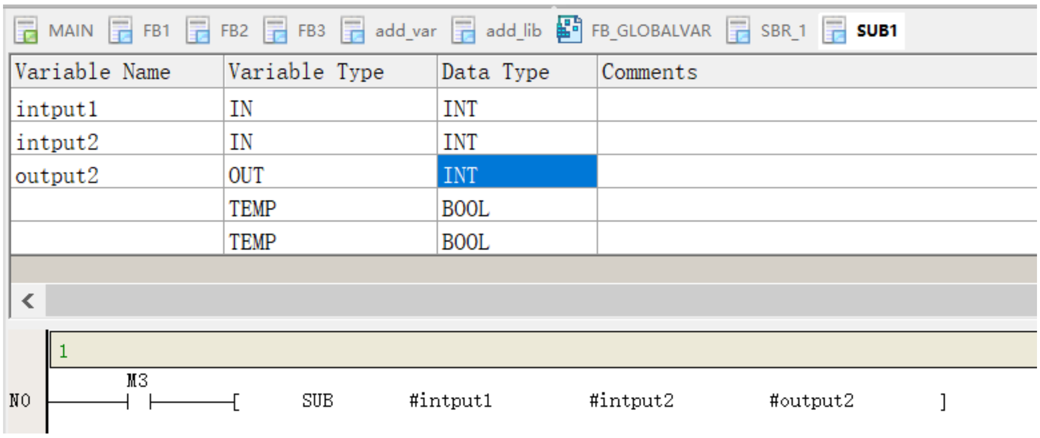
Variable type	Category	Description
		output parameters, which can be generated on the left side, right side or the top of the graphic block.
TEMP	Local variables	Valid only within this logic block and cannot be accessed externally.

- Data types: Variable data types supported include BOOL, INT, DINT, WORD, DWORD, and REAL (other types will be developed iteratively with each version).
- Note: Used to annotate the meaning of variables.

Note: Compared to FB function blocks, the FC functions cannot set initial values and power-off retention properties, and all local variables are not retained.

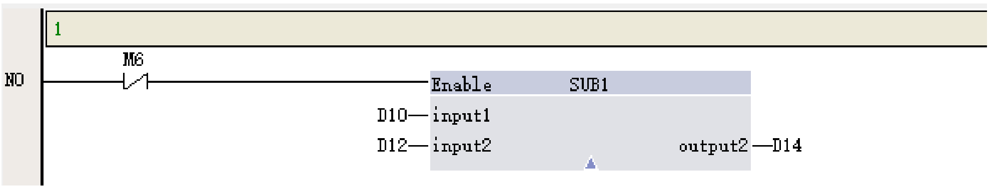
3.10.2.3 Example of Encapsulating Integer Subtraction Using FC

1. **Editing the function**



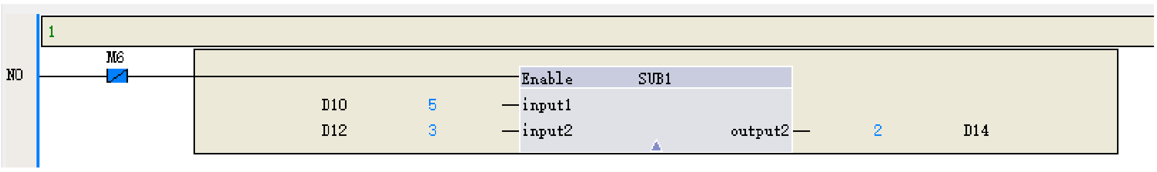
2. **Calling the FC function**

After writing the FC function block program, use it in the MAIN application program by calling the function block.



3. **Running the FC function**

The "Enable" of the FC function is connected to the power flow network of the ladder diagram. When the Enable network flow is valid (ON), the function block program is executed, and the output of the function block is refreshed based on the input state. When the Enable network flow is invalid (OFF), the function block program is not executed, and the output of the function block is not refreshed.



The SUB1 function executes when the power flow condition is ON: D14 = D10 - D12.

4. **Encapsulating the FC function (Support for future iterations)**

The edited and debugged FC functions can be encapsulated into a library, which can be imported, exported, and managed through the library of the Auto Station Pro software, achieving reuse of the same library in different programs.

4 User C language

4.1 Overview

TS Series PLC supports users to use C language to write functional blocks in programming software, call them where needed, and supports common C language attribute library. Users' C language is called in ladder diagram using CALL instruction, parameters can transfer bit element M, word element D and word element R, read and write element values. And users can use C language to replace other PLC programming languages to realize complex logic and arithmetic operations, which can effectively improve the development efficiency of programmers.

4.2 Instruction Format

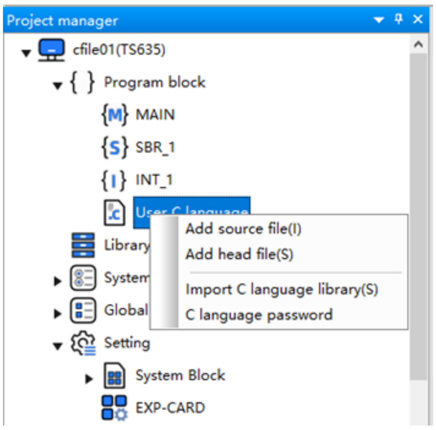
User C language function is called in ladder diagram by using CALL instruction, which includes CALL instruction, function name and parameters. The function name is the same as C language file name, which is defined when it is created, and the interface name cannot be changed in the file.

- CALL instruction
- Function name (ButtleSort)
- Parameters (M10 D10)

4.3 Operating Steps

Step 1 Create C language

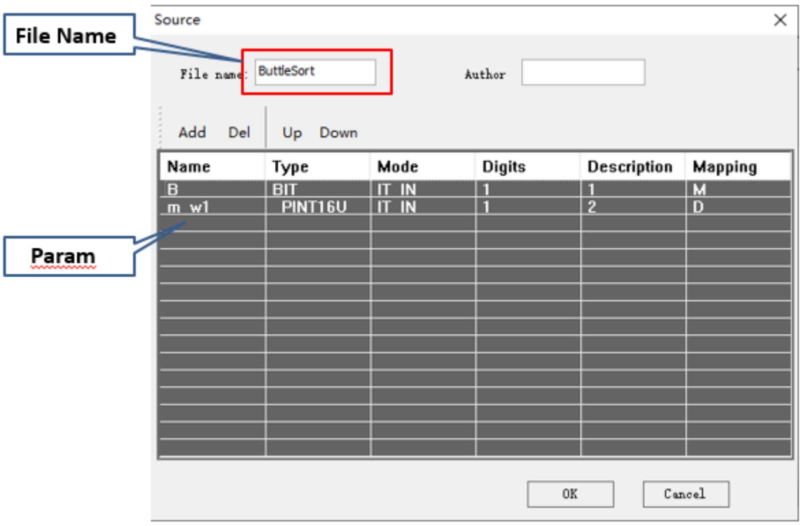
Open the PLC editing software, choose "User C language" node in the "Project manager" toolbar on the left, right-click and select "Add source file", and the software interface will pop up the user C language interface design window.



Step 2 Design C language interface

Fill in the user C language function information in the C language source file interface design dialog box

The function name is required, and it cannot be the same as the subroutines, interrupt subroutines and other C language functions, that is, avoid using function names containing SBR_ and INT_strings; Up to 16 parameters can be added, parameters can not be left blank, must have an unique name which can not be any PLC soft element name.



The supported data types are as follows:

Type	Description
BIT	Boolean quantity
_INT16U	16-bit unsigned integer
_INT16S	16-bit signed integer
_INT32U	32-bit unsigned integer
_INT32S	32-bit signed integer
_FP32	32-bit floating point
_PINT16U	16-bit unsigned int pointer
_PINT16S	16-bit signed int pointer
_PINT32U	32-bit unsigned int pointer
_PINT32S	32-bit signed int pointer
_PFP32S	32-bit float pointer

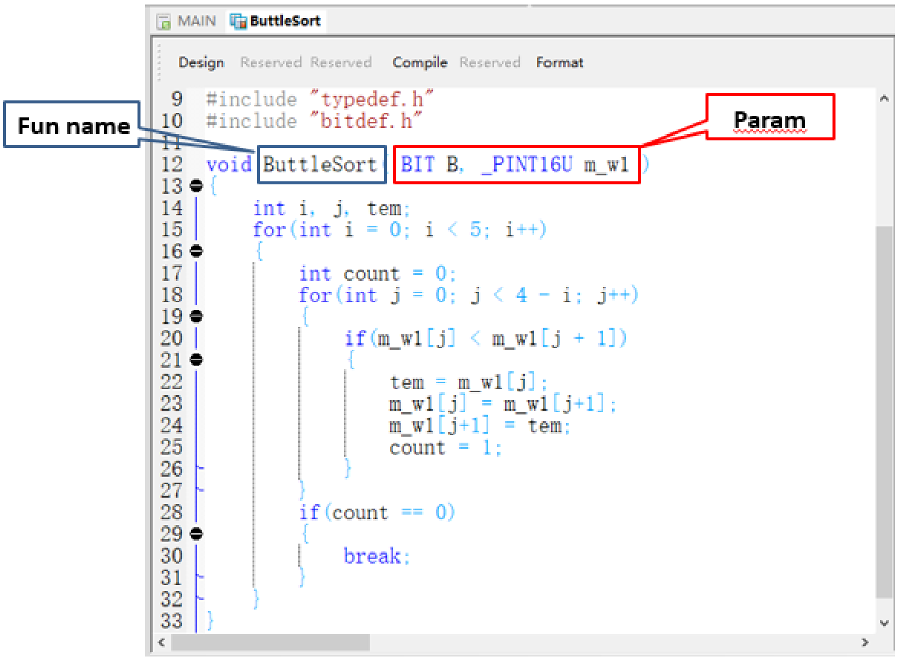
Step 3 C language editing

After creation, enter the C language editing interface where users can write the functions needed to realize.

The part generated by default includes the included header file (including three header files by default: plcstdafx.h, typedef.h, bitdef.h), C language interface function body, and users do not need to manually change or delete the default header file, function interface name, return value type and function parameters, otherwise compilation errors may happen.

Click the "Design" button to re-edit the interface design, then this part will be re-produced, and the previous part needs to be manually deleted to avoid compilation errors. In the example, the function is of C language bubble sort algorithm.

- Parameter passing mode: When ladder diagram is called, the passed-in M and D are the start addresses of B and m_w1. As shown in the following figure, when the elements in the instruction ButtleSort are M0 and D0, then B[0] is M0, B[10] is M10, m_w1[0] is D0, and m_w1[10] is D10 in C language functions; If the parameters used in the ladder diagram are M100 and D100, then B[0] is M100 and m_w1[0] is D100.

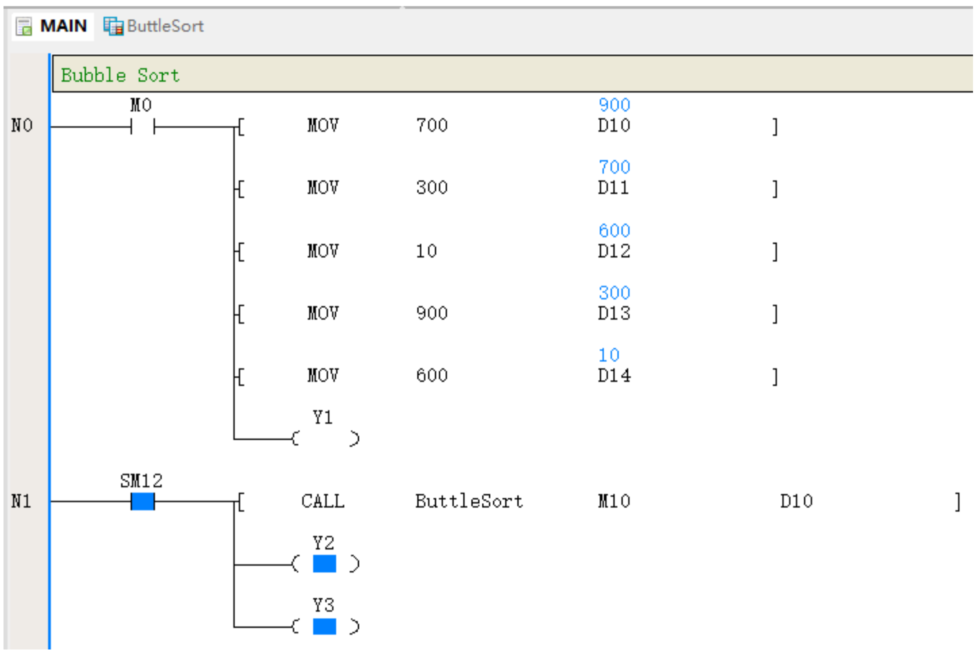


Step 4 Use of C language program:

User C language functions are called by CALL instructions. For example, for the bubble sort function mentioned above, the following shall be input in ladder diagram:

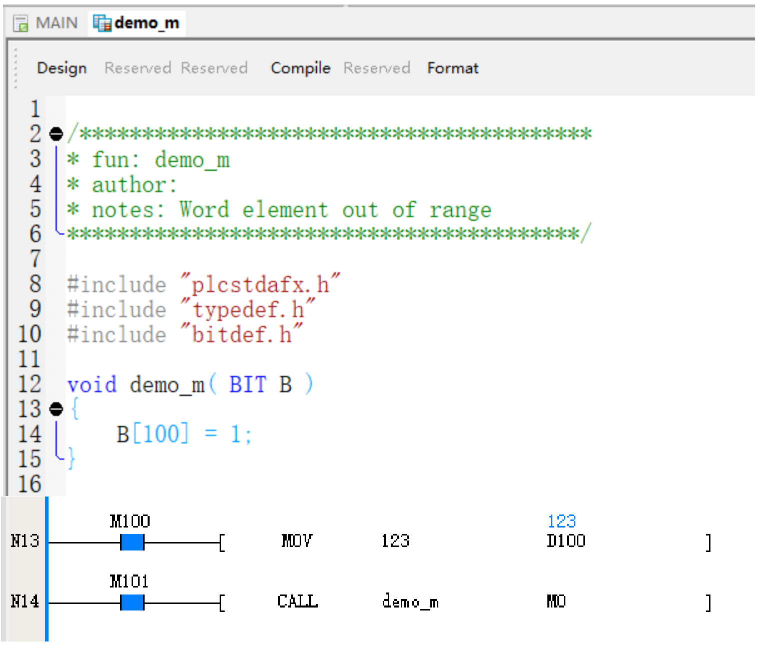
- CALL instruction
- Function name (ButtleSort)
- Parameters (M10 D10)

When compiling a ladder diagram, check the instruction block. If it is CALL instruction, check whether its function name is C language function (note the difference between ladder diagram subroutine and interrupt subroutine). If it is C language function, check parameters, match parameter type and quantity, and if the instruction block is correct, compile the C language file and generate an executable file.



4.4 Use Case

1. Bit element (M) pass parameter

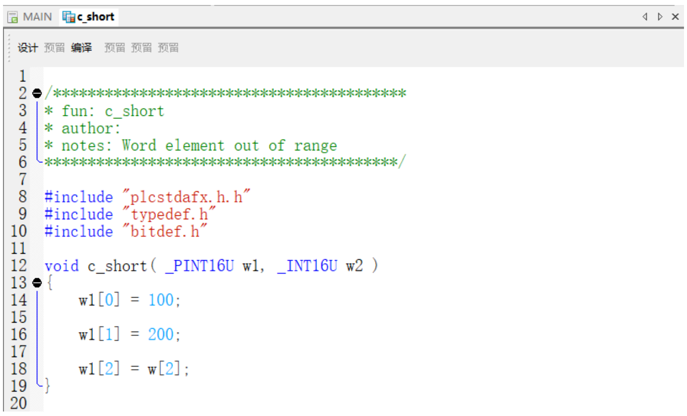


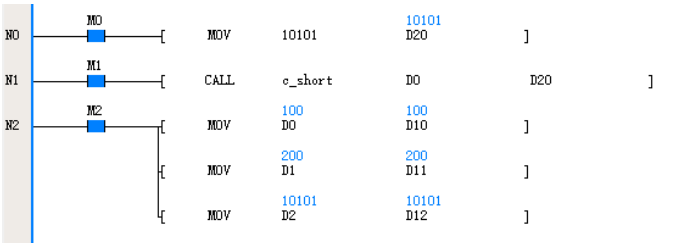
In the above case, setting M101 to 1 will trigger the call of C language function, in which M100 is set to 1, so that M100 element energy flow section can be executed.

Bit soft elements are used as per array, and support position 1 and bit clearing, such as B[0]=1; B[1]=0; And assignment, such as B[0]=B[1]. If the pass parameter is M0, then B[0] is M0 and B[100] is M100 when the function is called. If the pass parameter is M10, then B[0] is M10 and B[100] is M110 when the function is called.

C language function parameters / Ladder diagram call parameters	B[0]	B[10]
M0	B[0]<==>M0	B[10]<==>M10
M10	B[0]<==>M10	B[10]<==>M20

2. 16-bit pointer type data and 16-bit integer data pass parameters





Parameter passing method: when ladder diagram is called

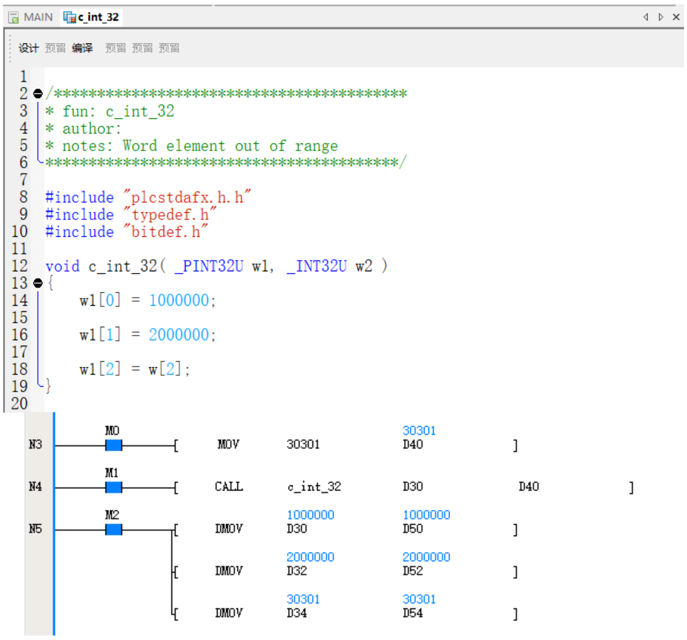
- When the function parameter is of pointer type, the passed-in element address is the start address of the C language function parameter.
- When the function parameter is of 16-bit integer type, the passed-in element is passed in the form of value.

As shown in the above figure, when c_short is called, D0 and D20 are passed in, where D0 is passed as a pointer and D20 is passed as a value. When the function is called, w1[0] is the address corresponding to element D0, w1[1] is the address corresponding to D1, and w1[2] is the address corresponding to D2. By assigning values to w1[0], w1[1] and w1[2] in C language functions, the values of elements D0, D1 and D2 will be changed correspondingly. w2 is the value passed, and the value is equal to the value of D20.

If the pass parameters are D10, D30, then w1[0] is D10, w1[1] is D11, w1[2] is D12 when the function is called, and the value of w2 is equal to the value of D30.

C language function parameters / Ladder diagram call parameters	w1[0]	w1[10]
D0	w1[0]<==>D0	w1[10]<==>D10
D10	w1[0]<==>D10	w1[10]<==>D20

3. 32-bit pointer type data and 32-bit integer data pass parameters



Parameter passing method: when ladder diagram is called

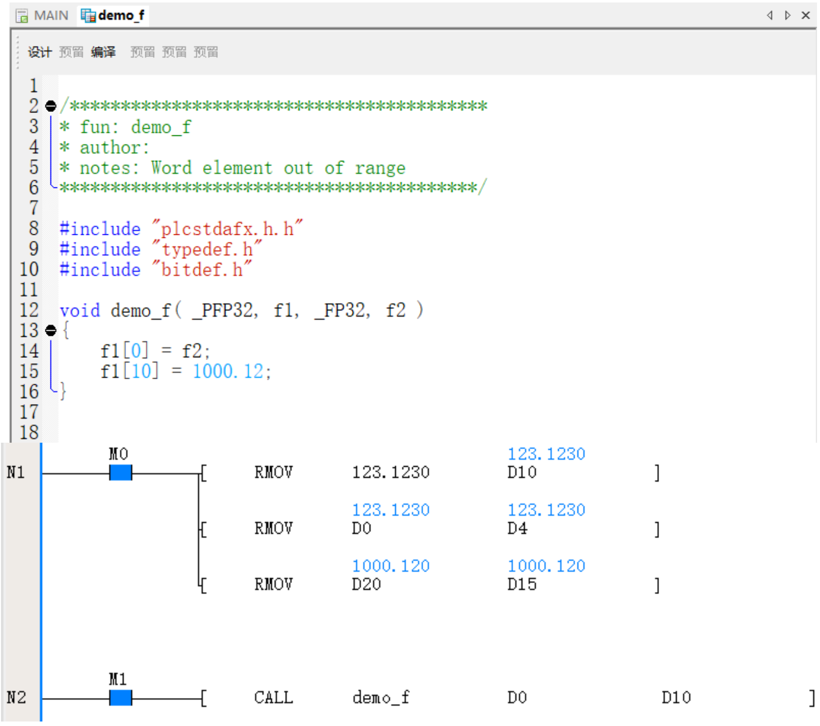
- When the function parameter is of 32-bit pointer type, the passed-in element address is the start address of the C language function parameter.
- When the function parameter is of 32-bit integer type, the passed-in element is passed in the form of value.

As shown in the above figure, when c_int_32 is called, D30 and D40 are passed in, where D30 is passed as a pointer and D40 is passed as a value.

Because it is of 32-bit pointer type, when the function is called, w1[0] points to the address corresponding to two elements with address bits D30 and D31, and w1[1] points to the address corresponding to two elements with address bits D32 and D33. By assigning values to w1[0] and w1[1] in C language functions, the values of D30, D31 and D32, D33 will be correspondingly changed. w2 is the value passed, and the value is equal to the value formed by combining the element addresses of D40 and D41.

C language function parameters / Ladder diagram call parameters	w1[0]	w1[10]
D0	w1[0]<==>D0D1	w1[10]<==>D20D21
D10	w1[0]<==>D10D11	w1[10]<==>D30D31

4. 32-bit float pointer type data and 32-bit floating point data pass parameters



Parameter passing method: when ladder diagram is called

- When the function parameter is of 32-bit float pointer type, the passed-in element address is the start address of the C language function parameter.
- When the function parameter is of 32-bit floating point type, the passed-in element is passed as a value.

As shown in the above figure, when demo_f is called, D0, D10 are passed in, where D0 is passed as a pointer and D10 is passed as a value.

Because it is of 32-bit pointer type, when the function is called, f1[0] points to the address corresponding to two elements with address bits D0 and D1, and w1[10] points to the address corresponding to two elements with address bits D20 and D21. By assigning values to f1[0] and f1[10] in C language functions, the values of D0, D1 and D20, D21 will be correspondingly changed. f2 is the value passed, and the value is equal to the value formed by combining the element addresses of D10 and D11.

C language function parameters / Ladder diagram call parameters	f1[0]	f1[10]
D0	f1[0]<==>D0D1	f1[10]<==>D20D21
D10	f1[0]<==>D10D11	f1[10]<==>D30D31

5 Programming Language

There are three programming languages: ladder diagram (LAD), instruction list (IL) and sequential function chart (SFC).

5.1 Ladder Diagram (LAD)

5.1.1 Concept of Ladder Diagram

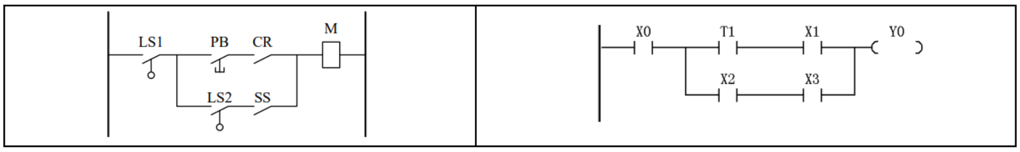
Ladder diagram is a common graphical PLC programming language similar to electrical (relay) control diagram.

Its main features include:

- With a left bus bar, while the right bus bar is omitted.
- All control output elements (coils) and function blocks (application instructions) have only one energy flow input.

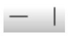
Electrical control diagram and ladder diagram are equivalent to some extent, as shown in Figure 5-1.

Figure 5-1 Equivalent Relationship between Electrical Control Diagram and Ladder Diagram



5.1.2 Basic Programming Elements of Ladder Diagram

Several basic programming elements can be abstracted from ladder diagram according to the principle of electrical (relay) control diagram:

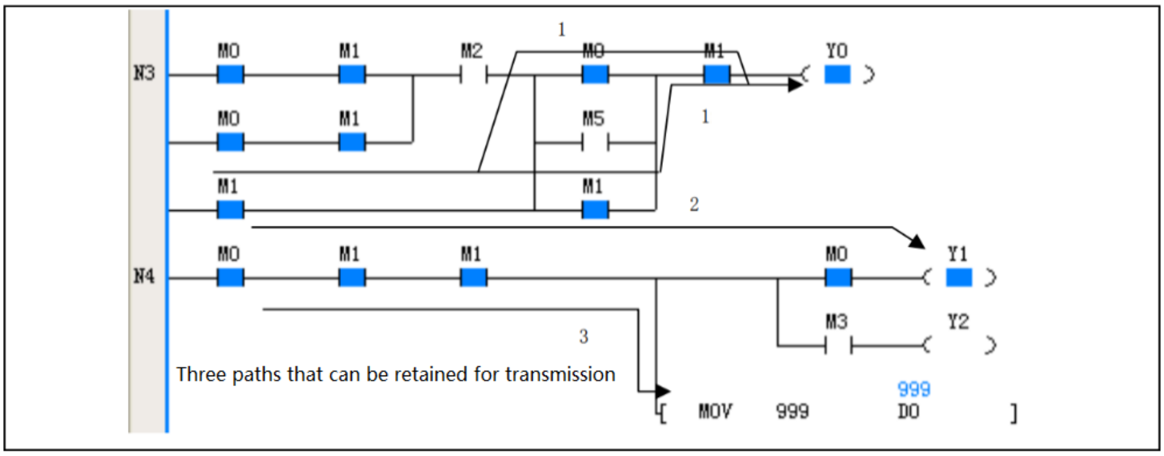
Element	Signification	Description
Left bus	-	Corresponding to the control bus in an electrical control diagram, it provides control power to the control loop.
Connecting line		It represents the electrical connection in an electrical control diagram, which is used to conduct other elements connected to each other.
Contact		It represents the input contact in an electrical control diagram, and it controls the on-off of the control current in the loop and determines the direction of the control current. The connection relationship between parallel connection and series connection of contacts essentially represents the operation relationship of input logic of control circuit and controls the transmission of energy flow.
Coil		It represents the relay output in the electrical control diagram.
Function block		Also called application instruction, it corresponds to the actuators or functional devices connected in the electrical control diagram which are used to achieve special functions, and a functional block can complete specific control functions or control calculation functions (such as data transmission, data operation, timer, counter, etc.).

5.1.3 Energy Flow

Energy flow is a very important concept in a ladder diagram program. Energy flow is used to drive coil elements and application instructions, and is similar to the output of drive coil and the control current executed by mechanism in an electrical control diagram.

In a ladder diagram, the front end of coil or application instruction must be connected with energy flow. Only when there is valid energy flow, the coil element can output and the application instruction can be effectively executed. Figure 5-2 shows the energy flow transfer in a ladder diagram and the driving effect of energy flow on coil or function block are demonstrated.

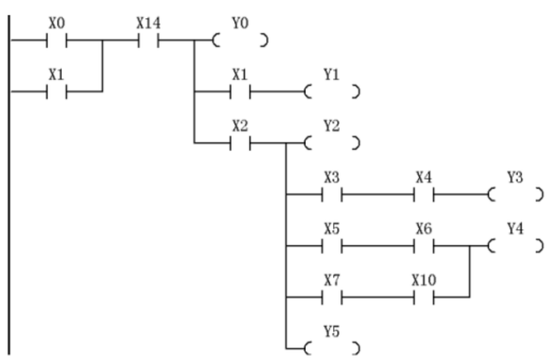
Figure 5-2 Energy Flow Transfer and Driving Effect of Energy Flow



5.2 Instruction List (IL)

Instruction list is a textual user program, which is a set of instruction sequences written by users. The user program stored in the main module of PLC for execution is actually an instruction sequence that can be recognized by the main module. The system executes each instruction in the sequence one by one to realize the control function of the user program. Figure 5-3 shows an example of converting a ladder diagram into an instruction list.

Figure 5-3 Conversion of Ladder Diagram into Instruction List

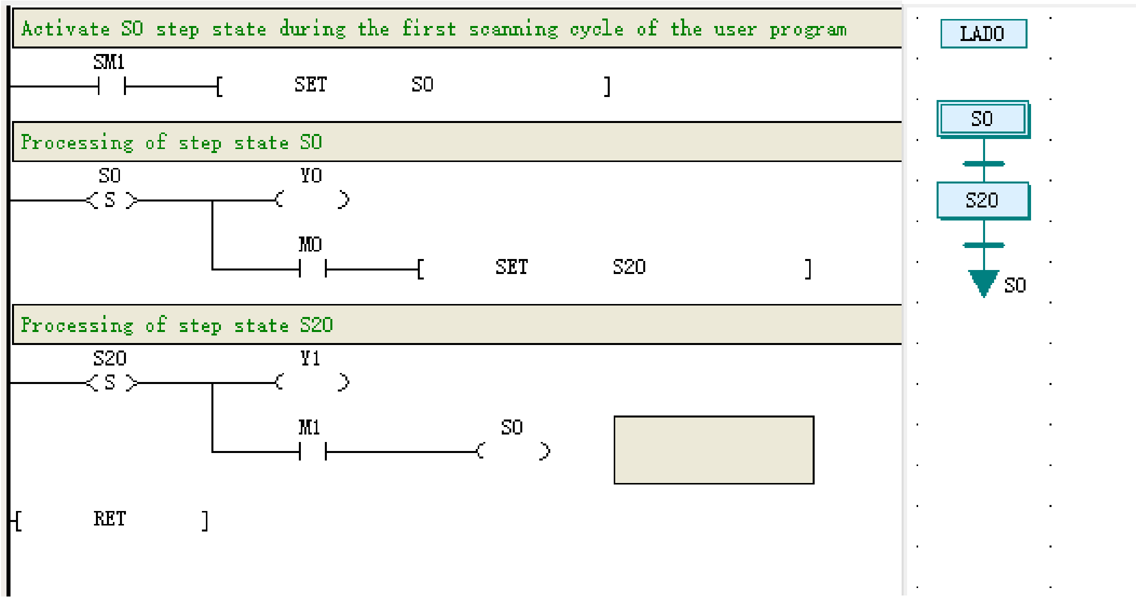
LAD	IL
	LD X0 OR X1 AND X14 MPS OUT Y0 AND X1 OUT Y1 MPP AND X2 MPS OUT Y2 AND X3 AND X4 OUT Y3 MRD LD X5 AND X6 LD X7 AND X10 ORB ANB OUT Y4 MPP OUT Y5

5.3 Sequential Function Chart (SFC)

Sequential function chart is a graphical user program framework design language, which is usually used to realize sequential control function.

Sequential control refers to a control process that can be divided into multiple procedures (processing steps) and processed according to a certain working sequence. The structure of the user program designed according to the sequential function chart is intuitive and clear, and is consistent with the actual sequential control process. Figure 5-4 shows an example of a simple sequential function chart.

Figure 5-4 Sequential Function Chart



6 Serial Communication

Serial Communication refers to a communication mode which transmits data between peripheral devices and computers by bit through data signal lines, ground lines, control lines, etc. This communication mode uses less data lines, which can save communication cost in long-distance communication.

This chapter introduces the serial communication function of TS600 Series Small PLC in detail, including communication resources and communication protocol, and illustrations with examples.

6.1 Serial Communication Resources

6.1.1 Supported Serial Communication Protocols

<table>
<tr><th>Main module</th><th>Communication ports</th><th>Communication port type</th><th>Supported protocols</th><th>Note</th></tr>
<tr><td rowspan="2">TS635</td><td>COM1</td><td>RS485</td><td rowspan="2">Modbus RTU/ASCII master communication protocol, Modbus RTU/ASCII slave communication protocol, freeport protocol, and tag communication protocol</td><td rowspan="3">TS635 supports two 485 and one expandable 232 interfaces</td></tr>
<tr><td>COM2</td><td>RS485</td></tr>
<tr><td>TS600 CAN-232 expansion card</td><td>COM3</td><td>RS232</td><td>Modbus RTU/ASCII slave communication protocol, freeport protocol, and tag communication protocol</td></tr>
</table>

6.1.2 Applicable Baud Rate

<table>
<tr><th>Communication protocol</th><th>Applicable baud rate</th></tr>
<tr><td>Freeport protocol</td><td rowspan="3">115200, 57600, 38400, 19200, 9600, 4800, 2400 and 1200bps</td></tr>
<tr><td>Modbus RTU (master, slave)</td></tr>
<tr><td>Modbus ASCII (master, slave)</td></tr>
</table>

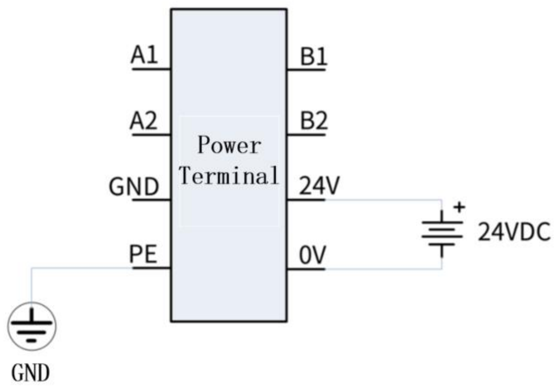
Note: Serial protocols cannot be reused. For example, COM1 can only use one protocol rather than multiple protocols at the same time.

6.2 Link characteristics

- Physical layer: RS232, RS485
- Link layer: asynchronous transmission
- Data bit: 8-bit (RTU), 7-bit (ASCII)
- Baud rates: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200bps
- Verification method: even check, odd check or no check
- Stop bit: 1 bit or 2 bits

6.3 RS485 Power Terminal Wiring

- Pin diagram

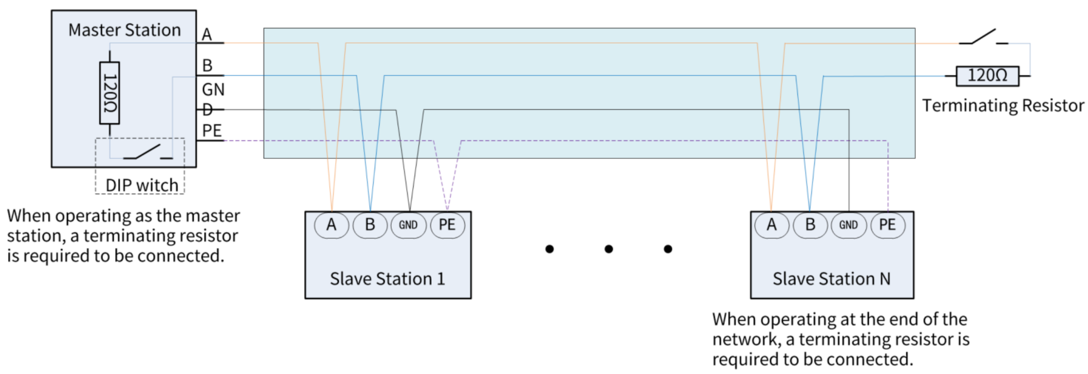


- Terminal function table

Pin	Function
A1	COM1 RS485+
B1	COM1 RS485-
A2	COM2 RS485+
B2	COM2 RS485-
GND	Power ground
24V	Power supply
PE	Protective earthing
0V	Power ground

6.4 Serial Networking Connection

The TS600 Series RS485 interface has a built-in 120 Ω terminal resistor which can be selected with a toggle switch. RS485 bus networking supports up to 31 nodes with the address range of 1-247, and supports broadcast. RS485 bus networking wiring diagram is as follows:

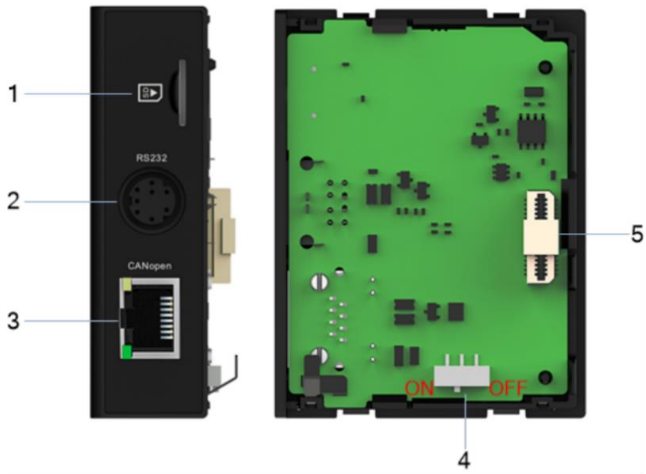


Note:

- Shielded twisted pair is recommended for RS485 bus, and A and B are connected by twisted pair.
- 120 Ω terminal matching resistors are connected at both ends of the bus to prevent signal reflection.
- The reference ground of 485 signals at all nodes is connected together.
- The distance of each node branch line should be less than 3m.

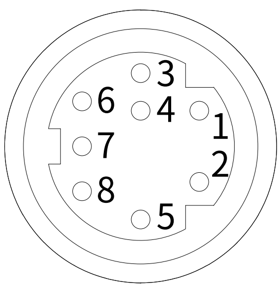
6.5 RS232 Interface

The following figure shows the CAN-232 expansion module and its interface definition.



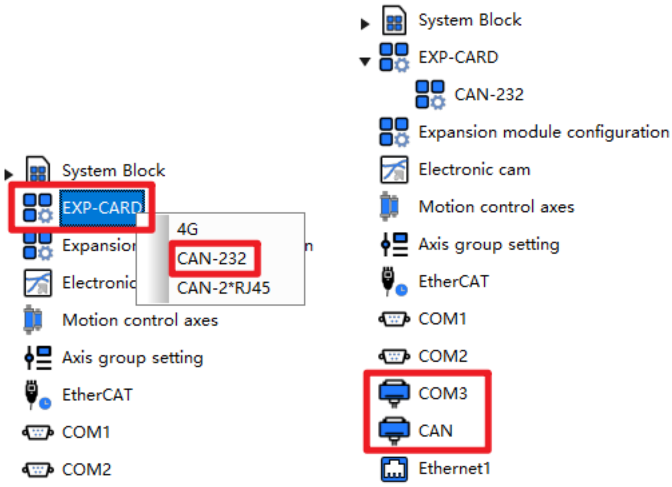
No.	Port type	Definition	Description
1	SD card slot	Insert MicroSD card here	Used for SD card firmware upgrade
2	RS232 Interface	RS232 bus interface	Used for RS232 communication
3	CANopen interface	CANopen bus interface	See the specific definition in the later sections
4	Switch	CANopen terminal resistance toggle switch	Toggled to ON: Built-in terminal resistor engaged Toggled to OFF: Built-in terminal resistor disengaged
5	Bus snap interface	Connected with the main module	Connected with PLC main module

RS232 interface definition:



PIN	Definition Description
1	NC
2	NC
3	GND
4	RS232 RX
5	RS232 TX
6	NC
7	NC
8	NC

When the CAN-232 expansion card is properly installed, the 232 and CAN communication configuration options will appear when "CAN-232" is selected for the expansion card.



6.6 Freeport Communication Protocol

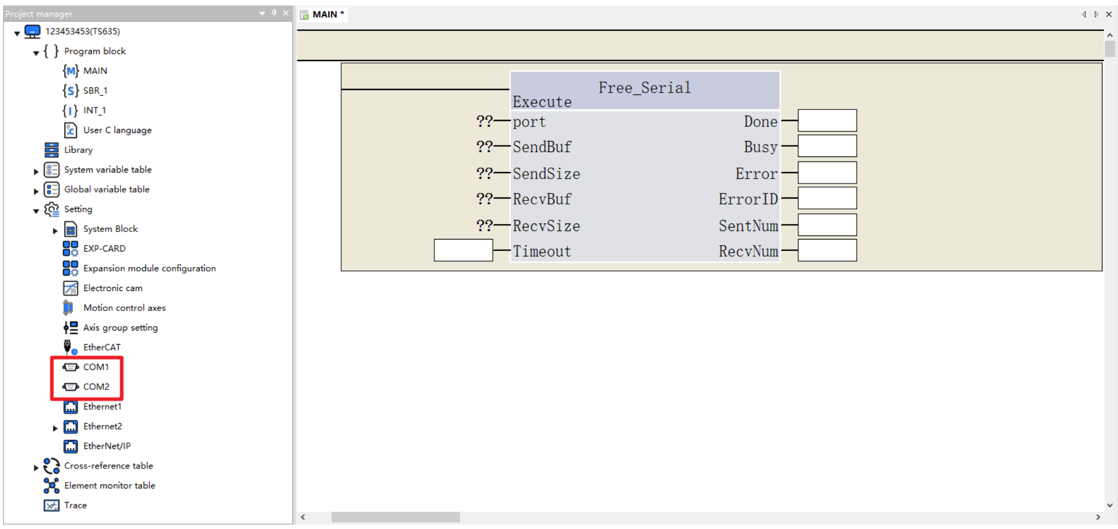
6.6.1 Brief Introduction

Freeport protocol is a communication protocol mode in a user-defined communication data format, which can transmit and receive serial data by instructions. In this mode, PLC can be used to communicate with devices in various custom formats, such as frequency converters, bar code scanners, instruments and other intelligent devices using freeport communication protocols. For a single device, PLC can communicate through RS232 or RS485. An RS485 network can also be formed with multiple devices.

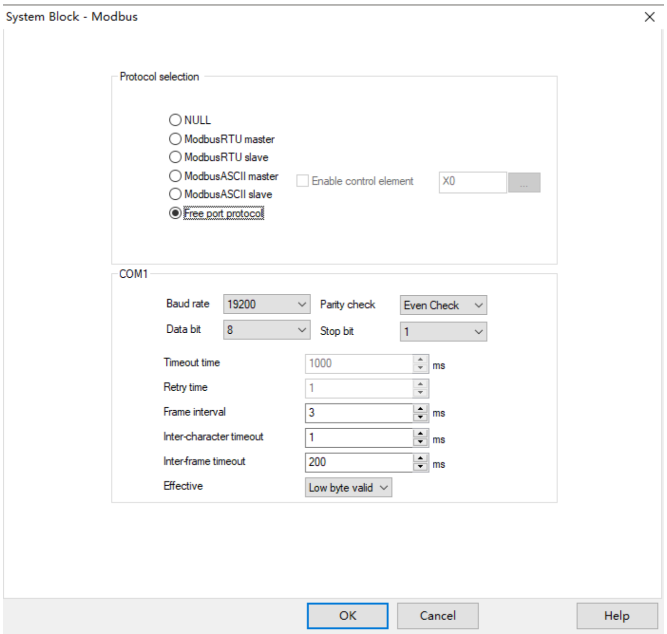
Note: The FreeSerial protocol instruction can only be used if the freeport parameters in the system block are configured.

6.6.2 Setting of Freeport Parameters

Step 1 Select the corresponding "COM" option in the system block to enter the communication port setting interface.



Step 2 Select "Free port protocol" in the protocol setting pop-up window to enter the freeport parameter setting interface, as shown in the following figure.



The configurable parameters are as follows:

Options	Setting Content	Note
Baud rate	115200, 57600, 38400, 19200, 9600, 4800 (default: 19200)	-
Data bit	Set to 7 or 8 (default to 8)	-
Parity bit	Set to no check, odd check and even check (default to even check)	-
Stop bit	Set to 1 or 2 (default to 1)	-
Frame spacing	Default to 0	Time between transmission of each frame (ms)
Inter-character timeout	Default to 0	Discarded when the time between two characters received exceeds this value.
Inter-frame timeout	Default to 200	Discarded when the time between two bytes received exceeds this value.
Effective byte	Low byte active/high and low byte active	● Low byte active: When transmitting or receiving data, manipulate the low byte of a word element, if two bytes are to be transmitted, then transmit the low byte of two word elements. ● High and low byte active: When transmitting or receiving data, manipulate the high and low byte of a word element, if two bytes are to be transmitted, then transmit the high byte and low byte of a word element.

After the configuration is completed, you can use FreeSerial instruction to transmit and receive serial freeport data. Please refer to FreeSerial instruction in the chapter of Communication Instructions of TS600 Series Programmable Logic Controller Command Manual for detailed use.

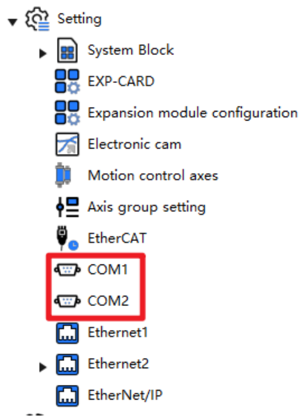
6.7 Modbus-RTU/ASCII Master

6.7.1 Brief Introduction

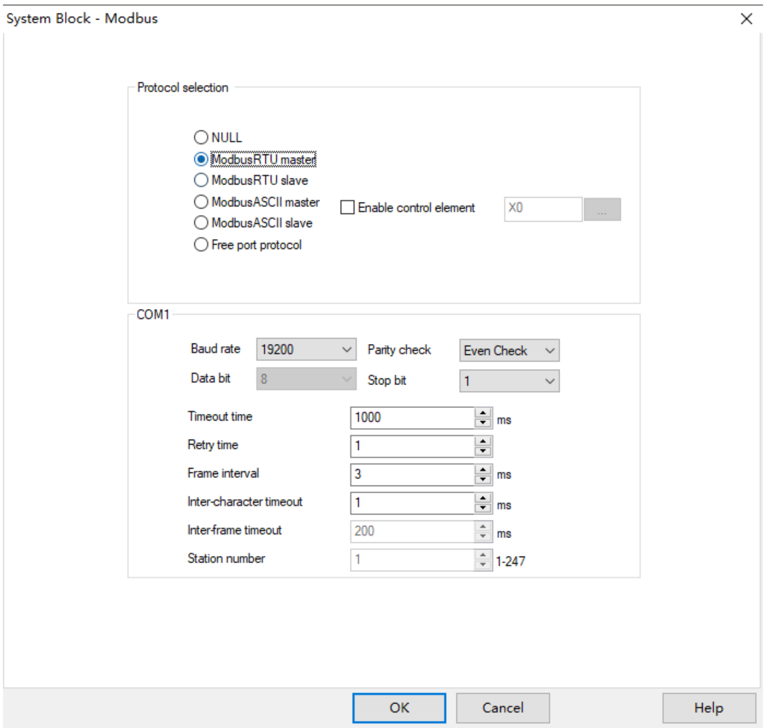
TS600 Series Small PLC supports two RS485 ModbusRTU/ASCII master or slave.

6.7.2 Serial Port Setting

Step 1 Double-click the corresponding "COM" option in the system block to enter the communication port setting interface.



Step 2 Select "ModbusRTU master" or "ModbusASCII master" as required in the pop-up window "Protocol setting".



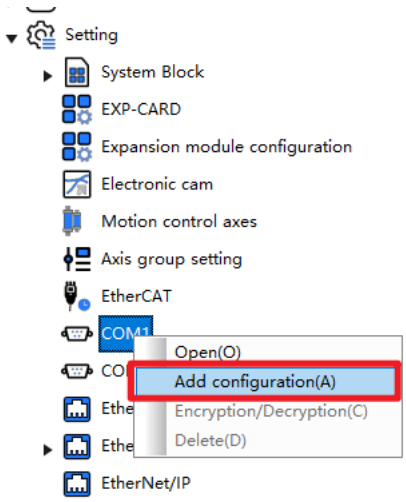
The configurable contents are as follows:

Options	Setting Content	Note
Check the enable control elements	Enable the enable control elements	This allows users to use bit elements to control the start and stop of Modbus, which is not enabled by default
Enable control elements	X, Y, M elements	Only valid when the Enable Control Element box is checked

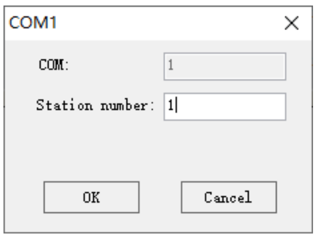
Options	Setting Content	Note
Baud rate	115200, 57600, 38400, 19200, 9600, 4800	Default to 19200
Data bit	Modbus RTU is fixed to 8 bits and Modbus ASCII is fixed to 7 bits	-
Parity bit	Set to no check, odd check or even check	Default to even check
Stop bit	Set to 1 or 2, default to 1	-
Timeout time	0–65535ms	Maximum communication timeout, default to 100ms
Retry times	0–15	Number of retransmissions when communication fails

6.7.3 Master Setting

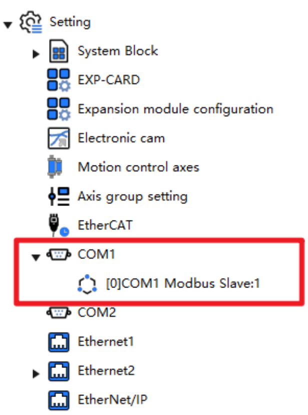
Step 1 Right-click the corresponding "COM" option in the system block and choose "Add configuration".



Step 2 Add configuration, set the slave number communicating with the master, and up to 248 slaves can be configured (0 – 247).



After configuration, the slaves are as shown in the following figure.



Step 3 Double-click "COM1 Modbus Slave" to open the Modbus Master Configuration Table. See 6.7.4 Master Configuration Table for details.

6.7.4 Master Configuration Table

This interface is used to configure the detailed configuration information of Modbus master node.

- Details of the configuration table are as follows:

Options	Setting Content	Note
Slave node ID.	0–247	The slave number 0 represents broadcasting, and up to 247 slaves are supported
Enable control element check box	Enable the enable control elements	This allows users to use bit elements to control the activation and inactivation of Modbus slaves, where each individual slave is controlled. And this option is not used by default.
Enable control elements	X, Y, M elements	Only valid when the Enable Control Element box is checked
Retry times	0–5	Number of retransmissions when communication fails
Trigger mode	"Loop" and "Trigger" modes are supported.	Loop: Cyclic loop execution. Trigger: Bit element rising edge execution.
Trigger condition	● When the trigger mode uses "Loop": "Trigger condition" is used to set the cycle time (unit: ms), and the configuration is executed at the specified cycle. ● When "Trigger" is used in trigger mode: "Trigger condition" is used to set trigger condition variables/elements	● When the trigger mode uses "Loop", the configuration is executed at the specified cycle. ● When "Trigger" is used in the trigger mode, the communication is triggered once by setting the trigger condition. When the communication is completed, the trigger condition will be automatically reset.
Function	01, 02, 03, 04, 05, 06, 15 (0x0f), 16 (0x10)	Modbus function code
Slave register	0–65535	-

Options	Setting Content	Note
address		
Qty	It is determined by each function code, as shown in the following table	Number of read/write registers or coils
Mapping address	It is determined by each function code, as shown in the following table	Address of slave coil/discrete quantity/register mapped in master

Note:

✧ If the set cycle period is less than the time required for communication, it will be executed at per the cycle of the actual communication time. For example, when the set period is 10ms but the communication actually requires an execution period of 20ms, the actual execution period is 20ms.

✧ When multiple Modbus instructions are configured to use the same trigger element, and when the element is triggered, the corresponding multiple Modbus instructions will be executed, and then the element will be automatically reset.

- The optional function/quantity and mapping address elements corresponding to the function code are as follows:

Function code	Function	Qty	Optional mapping address element
01(0x01)	Read coils	1–2000	M
02(0x02)	Read discrete input	1–2000	M
03(0x03)	Read save register	1–125	D, R
04(0x04)	Read input register	1–125	D, R
05(0x05)	Write single coil	1	M
06(0x06)	Write single register	1	D, R
15(0x0F)	Write multiple coils	1–1968	M
16(0x10)	Write multiple registers	1–123	D, R

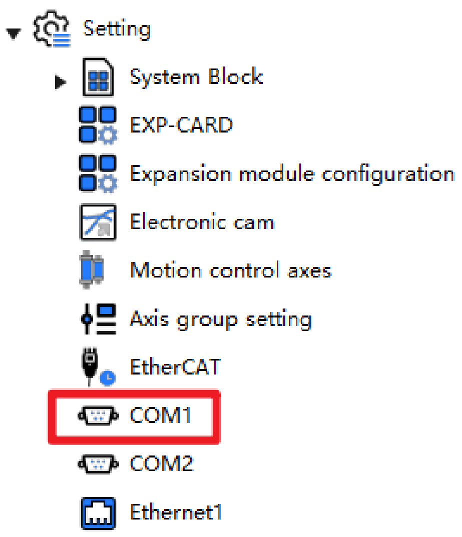
6.8 Modbus-RTU/ASCII Slave

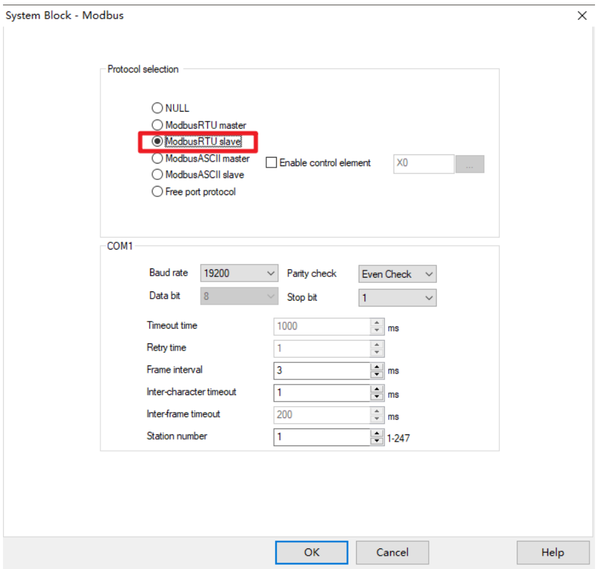
6.8.1 Brief Introduction

TS600 Series Small PLC supports two RS485 RTU/ASCII master or slave.

6.8.2 Slave Settings

Double-click the corresponding "COM" option in the system block to enter the communication port setting interface. Select "ModbusRTU slave" or "ModbusASCII slave" as required in the pop-up window "Protocol setting".





The configurable contents are as follows:

Options	Setting Content	Note
Check the enable control elements	Enable the enable control elements	This allows users to use bit elements to control the start and stop of Modbus, which is not used by default
Enable control elements	X, Y, M elements	Only valid when the Enable Control Element box is checked
Baud rate	115200, 57600, 38400, 19200, 9600, 4800	Default to 19200
Data bit	RTU is fixed to 8 bits, ASCII is fixed to 7 bits	-
Parity bit	Set to no check, odd check or even check	Default to even check
Stop bit	Set to 1 or 2, default to 1	-
Node number	1–247	Slave node ID

Note: After the configuration is completed, download it to TS600 series small PLC.

6.8.3 Slave Related Information

6.8.3.1 Function Code and Element List

When working as a slave, the supported function codes and operable elements are as follows:

Function code	Name	Type of operable elements	Note
01(0x01)	Read coils	Y, X, M, S, T, C	Read bit
02(0x02)	Read discrete input	X	Read bit
03(0x03)	Read registers	D, Z, T, C, R	Read word
05(0x05)	Write single coil	Y, M, S, T, C	Write bit
06(0x06)	Write single register	D, Z, T, C, R	Write word
15(0x15)	Write multiple coils	Y, M, S, T, C	Write bit
16(0x16)	Write multiple registers	D, Z, T, C, R	Write word

6.8.3.2 Address Binding Table

As a slave, the accessible Modbus protocol addresses are as follows:

Element	Type	Total quantity	Protocol address	Supported function codes (Decimal)	Description
M0–M32767	Bit element	32768	0x0000–0x7FFF (0–32767)	01, 05, and 15	M0 – M999: not saved in case of power-down
S0–S4095	Bit element	4096	0x8000–0x8FFF (32768–36863)	01, 05, and 15	S0–S999: not saved in case of power-down
X0–X1777	Bit element	1024	0xA000–0xA3FF (40960–41983)	01, 02	Input, octal encoded, BOOL type
Y0–Y1777	Bit element	1024	0xB000 – 0xB3FF (45056 – 46079)	01, 05, and 15	Output, octal encoded, BOOL type
T0–T399	Bit element	400	0xC000 – 0xC18F (49152 – 49551)	01, 05, and 15	● Accuracy 100ms: T0 – T199, 200 points ● Accuracy 10ms: T200–T299, 100 points ● Accuracy 1ms: T300–T399, 100 points
C0–C255	Bit element	256	0xC200 – 0xC2FF (49664 – 49919)	01, 05, and 15	● 16-bit ordinary CTUD: C0 – C199, 200 points ● 32-bit ordinary CTUD: C200 – C255, 56 points
D0–D32767	Word element	32768	0x0000–0x7FFF (0–32767)	03, 06, and 16	-
R0–R16383	Word element	16384	0x8000–0xBFFF (32768–49151)	03, 06, and 16	R16384 – R32767 are not mapped to protocol address
T0–T399	Word element	400	0xE000–0xE18F (57344–57743)	03, 06, and 16	● Accuracy 100ms: T0 – T199, 200 points ● Accuracy 10ms: T200–T299, 100 points ● Accuracy 1ms: T300–T399, 100 points

Element	Type	Total quantity	Protocol address	Supported function codes (Decimal)	Description
C0–C199	Word element	200	0xE200–0xE2C7 (57856–58055)	03, 06, and 16	● 16-bit ordinary CTUD: C0 – C199, 200 points
C200–C255	Word element	56	0xE2C8–0xE337 (58056–58167)	03, 06, and 16	● 32-bit ordinary CTUD: C200 – C255, 56 points ● Doubleword elements, every two registers corresponds to one element, and the registers can be read and written in even numbers.
Z0–Z15	Word element	16	0xE400–0xE40F (58368–58383)	03, 06, and 16	-

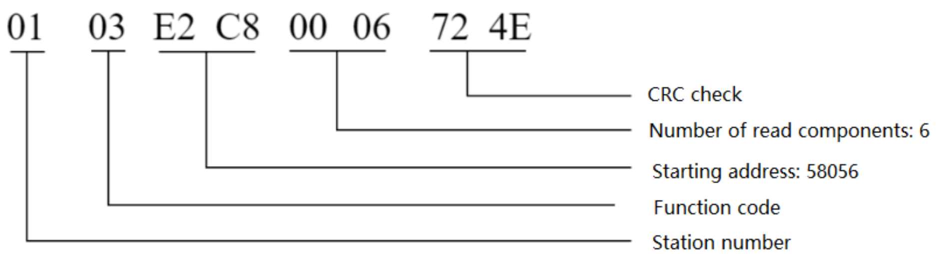
Note: T and C have word elements and bit elements.

6.8.3.3 Processing of Doubleword Elements

The current count value of C element is a word element or a doubleword element, and C200 – C255 are doubleword elements.

The reading and writing of C200 – C255 is also completed by the function codes (03, 16) of read/write registers. The address of every two registers corresponds to a C doubleword element, and the registers can be read and written in pair.

For example, when reading RTU frames of three C doubleword elements from C200 to C202:



In the returned data, the addresses 58056-58057 represent the contents of C200, with 58056 being the lower 16 bits and 58057 being the upper 16 bits.

Note: When reading doubleword elements, if the start address of reading is not an even, the abnormal code illegal address will be returned, and if the number of registers read is not an even, the abnormal code illegal data will be returned.

Examples of errors are as follows:

Master transmission: 01 03 E2 C9 00 04 A2 4F

The master transmits four word elements whose reading addresses start from E2 C9 (decimal 58057)

Slave response: 01 83 02 C0 F1

Slave reply: Illegal data address

Master transmission: 01 03 E2 C8 00 05 32 4F

The master reads five word elements whose addresses start from E2 C8 (decimal 58056)

Slave response: 01 83 03 01 31

Illegal data returned from the slave

6.8.3.4 Read SN code and the Model

When the PLC acts as a Modbus slave, the master can obtain the factory SN code and model information of the PLC through Modbus addresses "65500–65511" (12 words).

The communication messages are as follows:

Master transmission: 01 03 FF DC 00 0C B4 21

The master transmits 12 word elements whose reading addresses start from FF DC (decimal 65500)

Slave response: 01 03 18 54 30 36 31 39 42 30 30 30 32 39 30 00 00 54 53 36 33 35 00 00 00 00 00 59 45

Among them, the ASCII code "54 30 36 31 39 42 30 30 30 32 39 30" represents the 12-digit PLC SN code "T0619B000290", while "54 53 36 33 35" represents the PLC model "TS635".

6.9 Modbus-RTU Communication Application Example

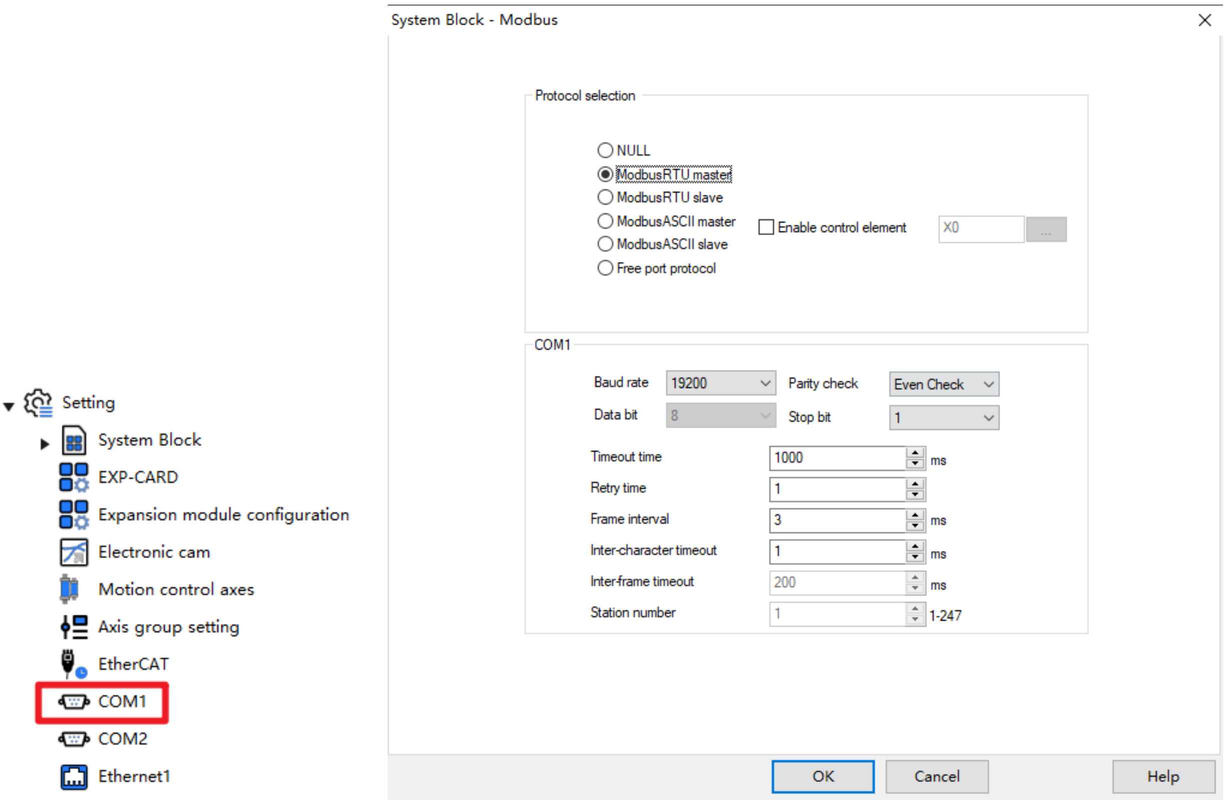
6.9.1 Brief Introduction

In this example, two sets of TS635 are configured as Modbus RTU master and slave respectively, and they communicate with each other via COM1 (physical interface RS485). For convenience of description, the following example assumes that the master number is A and the slave number is B.

6.9.2 Enable Master

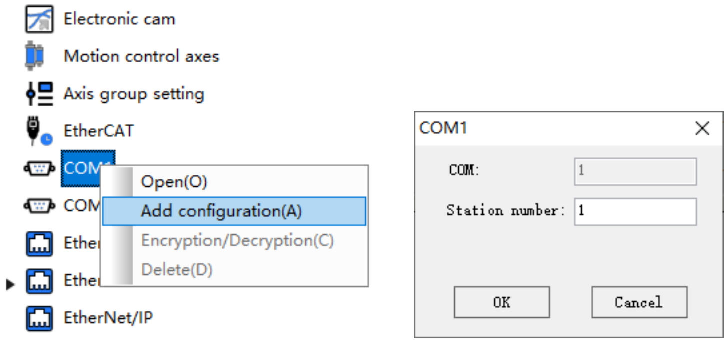
6.9.2.1 Serial Configuration

Double-click "COM1" in the system block, choose "Modbus RTU master" > "Baud rate 19200, 8 data bit, even check, 1 stop bit", and then click "OK" to save the configuration.

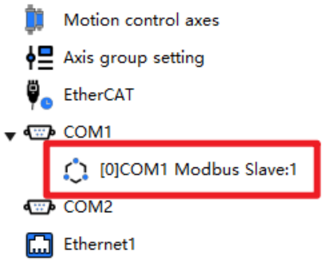


6.9.2.2 Communication Parameter Configuration

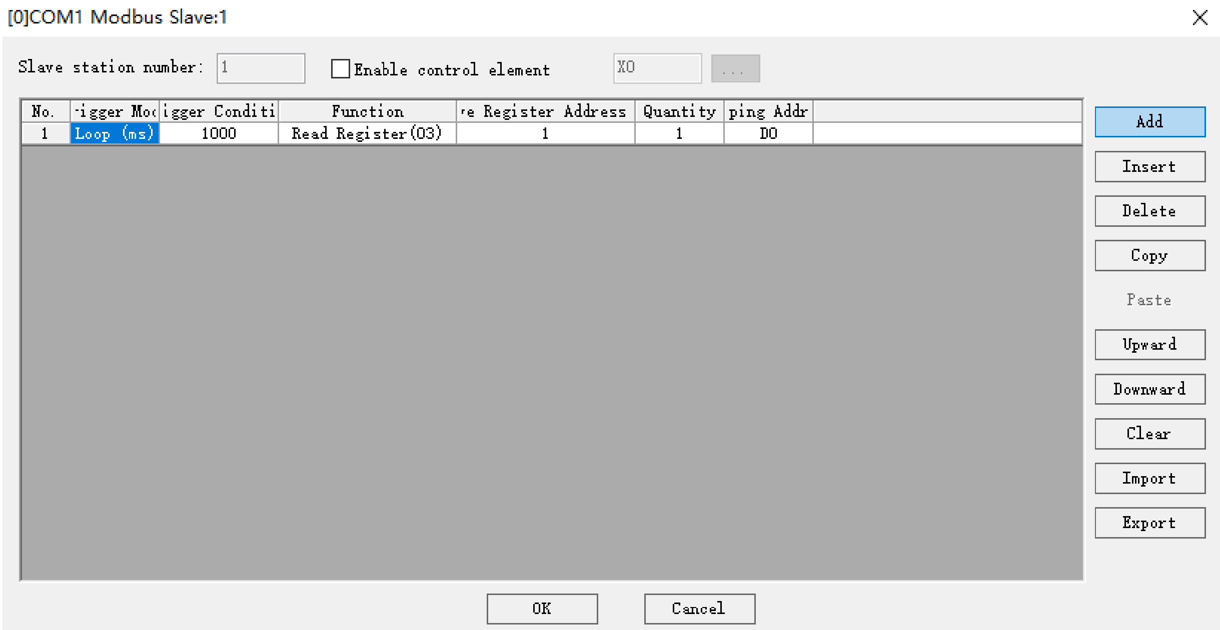
Step 1 Right-click "COM1" > "Add configuration", and configure the slave numbered to 1 in the pop-up box, and it is the target with which the communication is to be established.



Step 2 The slave is successfully added if it is as shown in the following figure.



Step 3 Double-click "COM1 Modbus Slave" to open "Modbus Master Configuration Table".

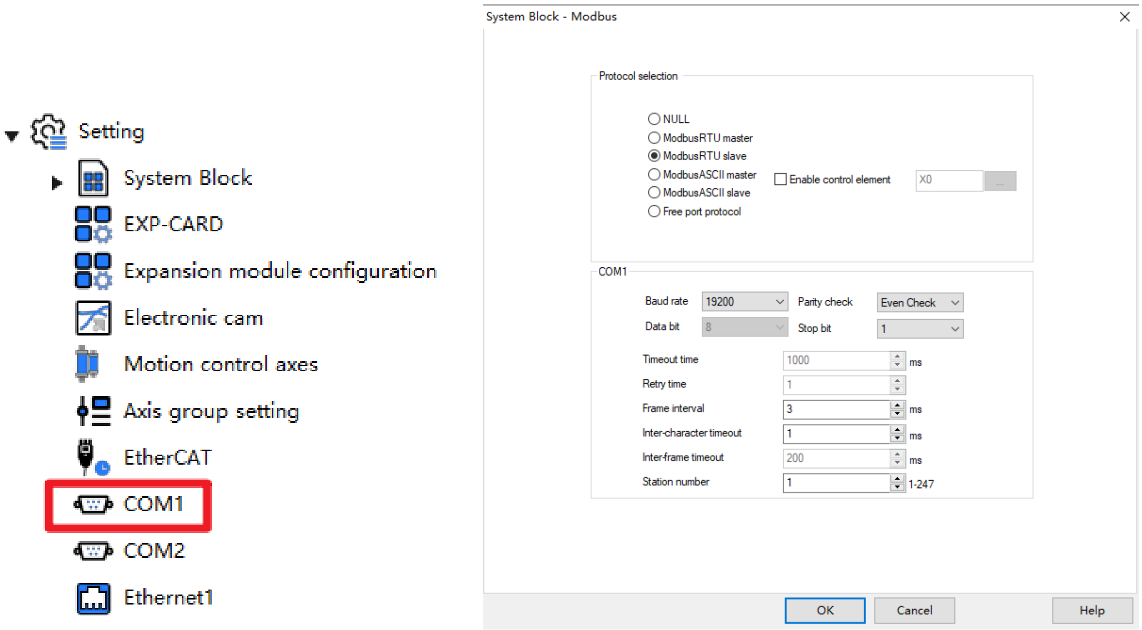


Note: If the same trigger variable can only trigger a single configuration, it is recommended to use different trigger variables.

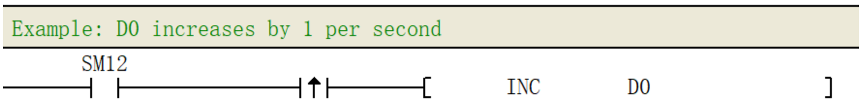
Step 4 Configure the above information and download it to "TS635 PLC A" after confirmation.

6.9.3 Enable Slave

Step 1 Double-click "COM1" in the system block, choose "ModbusRTU slave" > "Baud rate 19200，8 data bit, even check, 1 stop bit", and set the slave number to 1. Click "OK" to save the configuration.



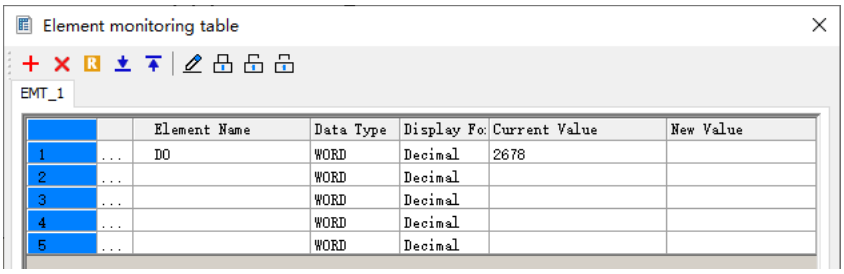
Step 2 Write a program to add 1 to D0 every second.



Step 3 Download it to "TS635 PLC B".

6.9.4 Example Phenomenon

In the element monitoring table, it can be observed that D0 in two PLCs is added by 1 every second.



		Element Name	Data Type	Display Fo:	Current Value	New Value
1	...	D0	WORD	Decimal	2678	
2	...		WORD	Decimal		
3	...		WORD	Decimal		
4	...		WORD	Decimal		
5	...		WORD	Decimal		

7 Ethernet Communication

Ethernet is the most common communication protocol standard which includes physical layer connection, electronic signal and medium access protocol. It has the advantages of low cost, high communication rate and strong anti-interference.

This chapter introduces the Ethernet communication function of TS600 Series Small PLC in detail, including communication resources and communication protocol, and illustrations with examples.

7.1 Ethernet Communication Resources

The main module of TS600 series comes with two Ethernet communication interfaces, and the data transmission rate is as high as 100M. It can be connected with Auto Station Pro upper computer for monitoring, uploading and downloading and firmware upgrading. All Ethernet ports support ModbusTCP function, which supports up to 32 connections (the connection with the same IP and port number are regarded as one) for data exchange, and a single node can be used as master and slave at the same time.

TS600 also supports TCP free protocol instruction and UDP free protocol instruction, and can easily realize TCP/UDP data interaction with other devices in the network through socket instruction. Please refer to the communication instruction chapter in the Instruction Manual for specific steps.

- The Ethernet communication protocols supported by TS600 Series Small PLC are as follows:

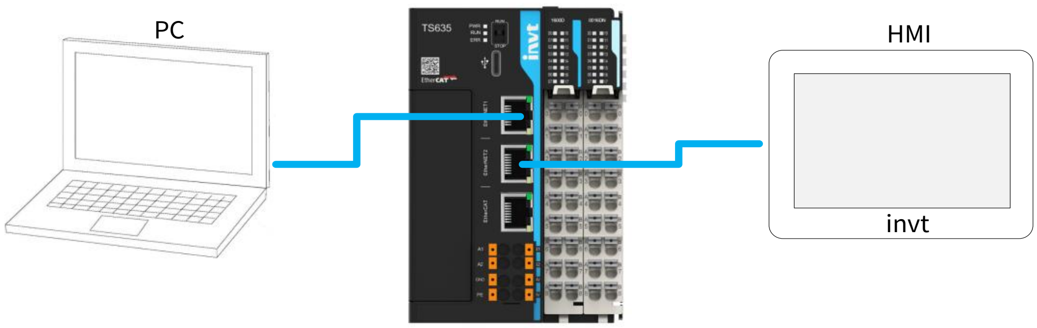
Main module	Communication ports	Communication port type	Supported protocols	Note
TS635	Ethernet 1	Standard 100 Mbps network port	ModbusTCP (master, slave), TCP free protocol, UDP free protocol, tag communication, EtherNet/IP (master, slave)	TS635 has two built-in standard Ethernet interfaces
	Ethernet 2			

Note: Ethernet protocol multiplexing is permitted. For example, Ethernet 1 can use Modbus TCP master, slave, TCP free protocol and UDP free protocol at the same time, as long as the port number is different.

- The specifications of Ethernet interfaces are as follows:

Item	Description
Communication protocol	Standard Ethernet protocol
Physical layer	100BASE-TX
Communication rate	100Mbps (100Base-TX)
Duplex mode	Full duplex
Topological structure	Linear topology structure
Transmission medium	Category-5 or higher network cables
Transmission distance	The distance between two nodes is less than 100m

7.2 Ethernet Wiring Example



In the example figure, Ethernet 1 is connected to Auto Station Pro in the upper computer to upload/download programs, and Ethernet 2 is connected to HMI for data display.

7.3 Ethernet IP Address

7.3.1 Factory Default IP Address

The default IP address of the PLC is shown as follows.

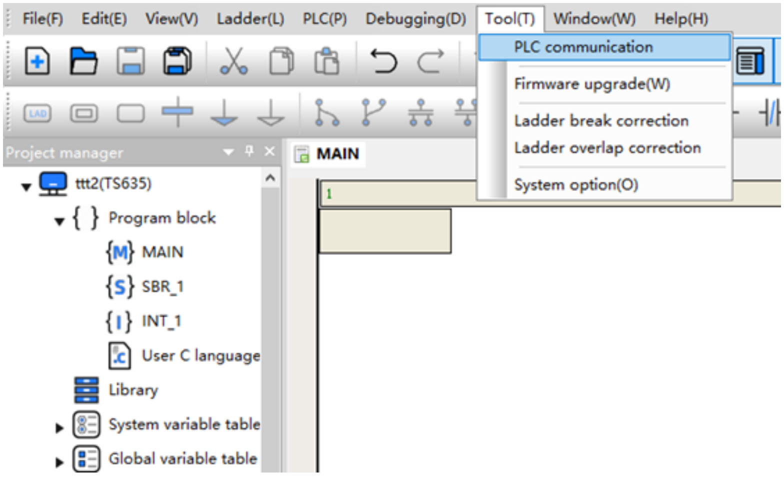
Model	Default IP address	
TS611	Ethernet is not supported	
TS620	Ethernet 1	192.168.2.10
TS630		
TS621, TS633, TS634 and TS635	Ethernet 1	192.168.1.10
	Ethernet 2	192.168.2.10

If you forget the Ethernet IP address, please use USB interface to read or modify the IP address, or reset Ethernet to the factory mode. For specific operation, refer to section 7.3.2 Modify IP Address in Upper Computer Interface or 7.3.3 Modify IP address with System Variables.

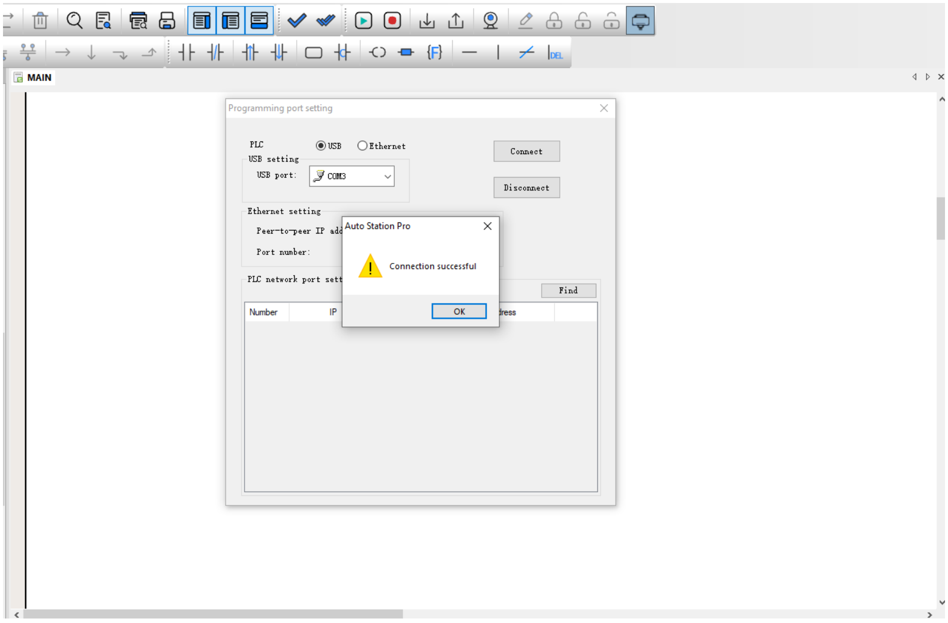
7.3.2 Modify IP Address in Upper Computer Interface

1. Modify IP address via USB

Step 1 Choose "Tool" > "PLC communication" in the toolbar.

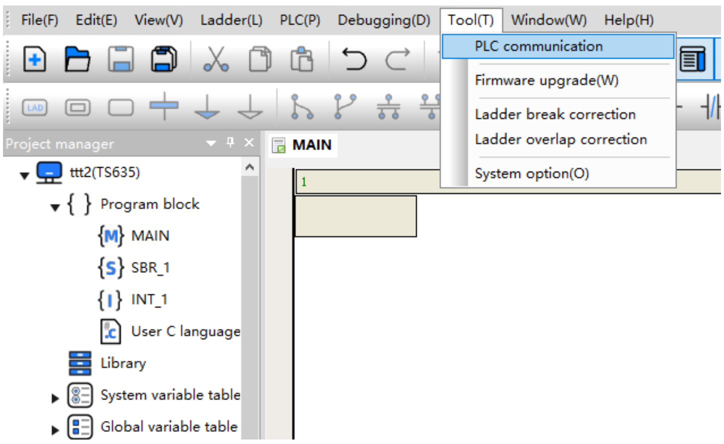


Step 2 Choose USB in "PLC", select the corresponding COM port in "USB", which is COM4 in this case, and then click the "Connect" button in the upper right corner of the interface, and the pop-up box indicating successful connection appears to indicate that USB communication has been established with PLC. Please skip step 2 and see3for subsequent operations.



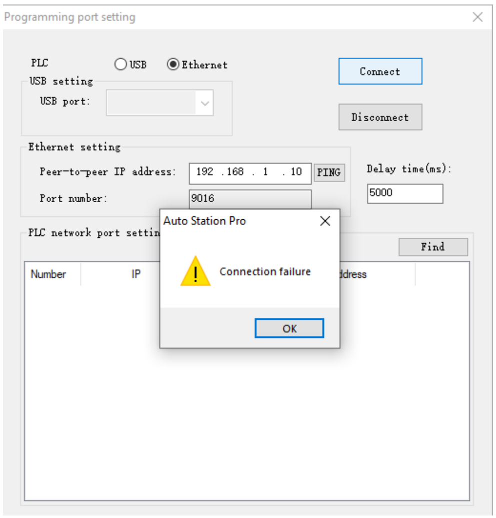
2. Modify IP address via Ethernet

Step 1 Choose "Tool" > "PLC communication" in the toolbar.



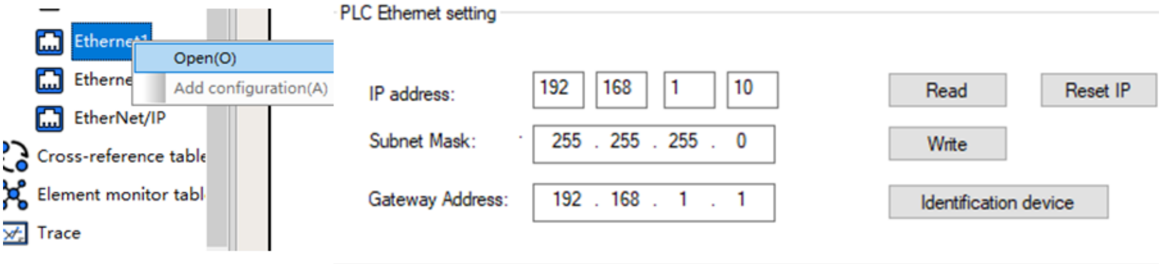
Step 2 Choose Ethernet in "PLC", and because the example physical interface is connected to Ethernet 2, so fill in the IP address of PLC network port 2, 192.168.2.10, in "Peer-to-peer IP address", and then

click the "Connect" button in the upper right corner of the interface, and a pop-up box indicating successful connection appears to indicate that Ethernet communication has been established with PLC. Please see 3 for subsequent operations.



3. Modify IP

After establishing a connection with PLC via USB or Ethernet, double-click "Ethernet" in Project Manager to view or configure Ethernet-related information.



- Reset IP: Restore all parameters of the two network ports to the factory default values.
- Read: Read the IP address, subnet mask and gateway address of port 1 or port 2.
- Write: Write IP address, subnet mask and gateway address to port 1 or port 2.
- Identify device: Reserved function, not available now.

Note: Ethernet information can be modified in running state without stopping or restarting PLC.

7.3.3 Modify IP address with System Variables

The system variable _sENETx.CmdCtrl is the control word for network configuration. Its value means:

0: Read. In this mode, PLC reads the current network information, and the network configuration cannot be written at this time.

1: Edit. In this mode, PLC can write or edit IP address, subnet mask and gateway.

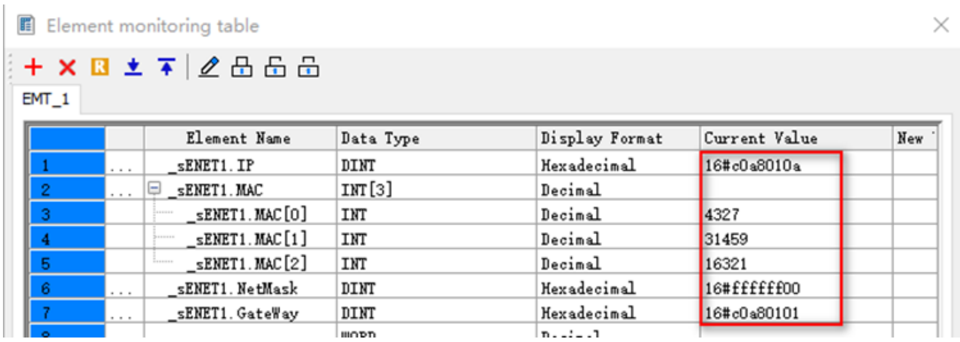
2: Write. Write IP address, subnet mask, gateway and other information of current network port system variables into PLC. After writing successfully, the control word automatically changes to 0, and when errors occur, the control word value is 3.

3: The control word is 3 when writing errors occur.

The following shows how to modify the network configuration of Ethernet 1.

Step 1 Set _sENET1.CmdCtrl to 1 to indicate that you are entering edit mode.

Step 2 Modify _sENETx.IP, _sENETx.NetMask, _sENETx.GateWay system variables. Use doublewords for IP address, subnet mask, or gateway, the variable _sENET1.IP with a value of "16#c0a8010a" in the following figure means "192（16#c0）.168（16#a8）.1（16#01）.10（16#0a）".



Step 3 Set _sENET1.CmdCtrl to 2, which indicates that the value of the current NIC system variable is written to PLC.

Step 4 _sENET1.CmdCtrl automatically changes to 0 after a successful writing and 3 when an error occurs.

7.4 TCP Free protocol

TCP protocol is a connection-oriented reliable communication protocol at transport layer based on byte stream. TS600 Series Small PLC provides the relevant TCP instructions to transmit and receive TCP protocol data. For specific usage, please refer to TCP-related instructions in the chapter of communication instructions of Instruction Manual.

7.5 UDP Free Protocol

UDP is a connectionless user datagram protocol. TS600 Series Small PLC provides the relevant UDP instructions to transmit and receive UDP protocol data. For specific usage, please refer to UDP-related instructions in the chapter of communication instructions of Instruction Manual.

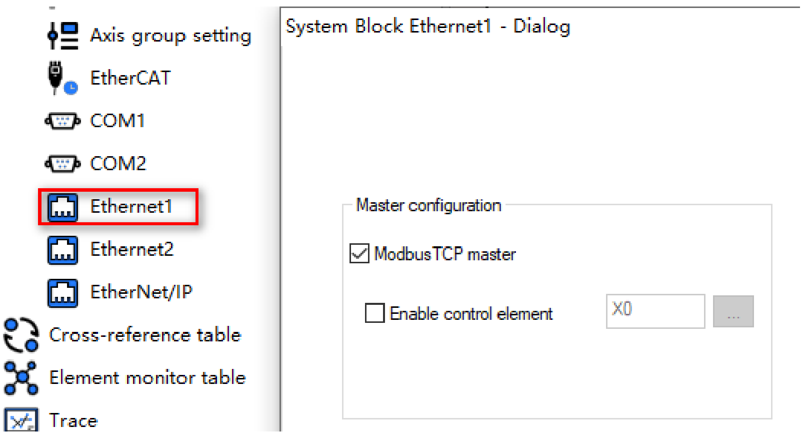
7.6 Modbus TCP Master

7.6.1 Brief Introduction

Modbus TCP slave is Modbus client, and TS635 Small PLC supports two Modbus TCP masters, and both Ethernet ports support checking Modbus TCP master and Modbus TCP slave at the same time.

7.6.2 Modbus TCP Master Settings

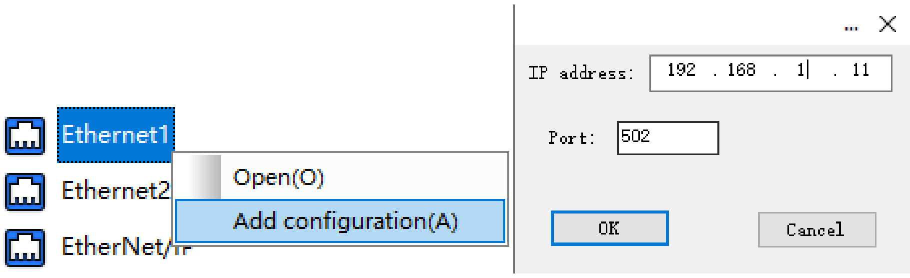
Step 1 Double-click the corresponding "Ethernet" option in the system block to enter the communication port setting interface. Check "ModbusTCP master" in the pop-up window "Protocol setting".



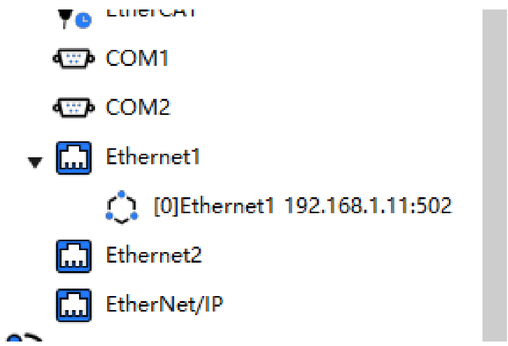
The configurable content is described as follows:

Options	Setting Content	Note
Check the enable control elements	Enable the enable control elements	This allows users to use bit elements to control the activation and inactivation of Modbus TCP slaves, where each individual slave is controlled. And this option is not used by default.
Enable control elements	X, Y, M elements	Only valid when the Enable Control Element box is checked.

Step 2 Click "Add configuration" to set the IP and port number of the slave that communicates with the master.



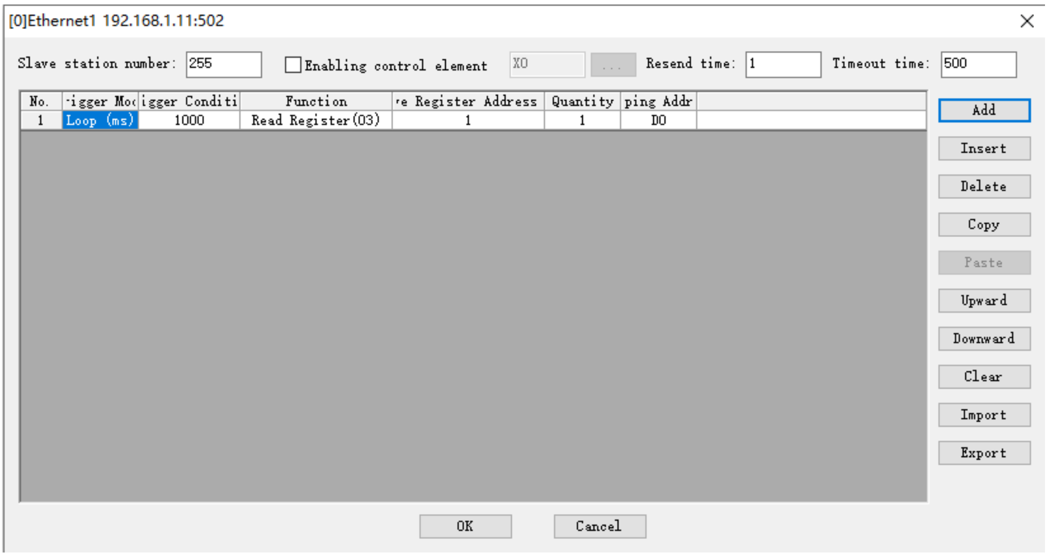
After configuration, the slaves are as shown in the following figure.



Step 3 Double-click "Ethernet1 192.168.2.11:502" to open the Modbus Master Configuration Table, as detailed in section 7.6.3 Modbus-TCP Master Configuration Table.

7.6.3 Modbus TCP Master Configuration Table

This interface is used to configure the detailed configuration information of Modbus TCP master node.



- Details of the configuration table are as follows:

Options	Setting Content	Note
Slave node ID.	0–255	-
Enable control element check box	Enable/disable enable control elements	This allows users to use bit elements to control the activation and inactivation of Modbus TCP slaves, where each individual slave is controlled. Not used by default.
Enable control elements	X, Y, M elements	Only valid when the Enable Control Element box is checked
Retry times	0–5	Number of retransmissions when communication fails
Trigger mode	"Loop" and "Trigger" modes are supported.	Loop: Cyclic loop execution. Trigger: Bit element rising edge execution.
Trigger condition	● When the trigger mode uses "Loop": "Trigger condition" is used to set the cycle time (unit: ms), and the configuration is executed at the specified cycle. ● When "Trigger" is used in trigger mode: "Trigger condition" is used to set trigger condition variables/elements	● When the trigger mode uses "Loop", the configuration is executed at the specified cycle. ● When "Trigger" is used in the trigger mode, the communication is triggered once by setting the trigger condition. When the communication is completed (the slave normally responds), the trigger condition will be automatically reset, otherwise the trigger condition will remain unchanged. If multiple configurations are triggered by the same trigger variable/element, the triggered configurations will be automatically reset after all the triggered configurations are executed after the trigger condition is set, and the triggered configurations will not be executed repeatedly.
Function	01, 02, 03, 04, 05, 06, 15 (0x0f), 16 (0x10)	Modbus function code
Slave register address	0–65535	-
Qty	It is determined by each function code, as shown in the following table.	Number of read/write registers or coils
Mapping address	It is determined by each function code, as shown in the following table.	Address of slave coil/discrete quantity/register mapped in master

Note:

✧ If the set cycle period is less than the time required for communication, it will be executed at per the cycle of the actual communication time. For example, when the set period is 10ms but the communication actually requires an execution period of 20ms, the actual execution period is 20ms.

✧ When multiple Modbus instructions are configured to use the same trigger element, and when the element is triggered, the corresponding multiple Modbus instructions will be executed, and then the element will be automatically reset.

- The optional function/quantity and mapping address elements corresponding to the function code are as follows:

Function code	Function	Qty	Optional mapping address element
01(0x01)	Read coils	1–2000	M
02(0x02)	Read discrete input	1–2000	M
03(0x03)	Read save register	1–125	D, R
04(0x04)	Read input register	1–125	D, R
05(0x05)	Write single coil	1	M
06(0x06)	Write single register	1	D, R
15(0x0F)	Write multiple coils	1–1968	M
16(0x10)	Write multiple registers	1–123	D, R

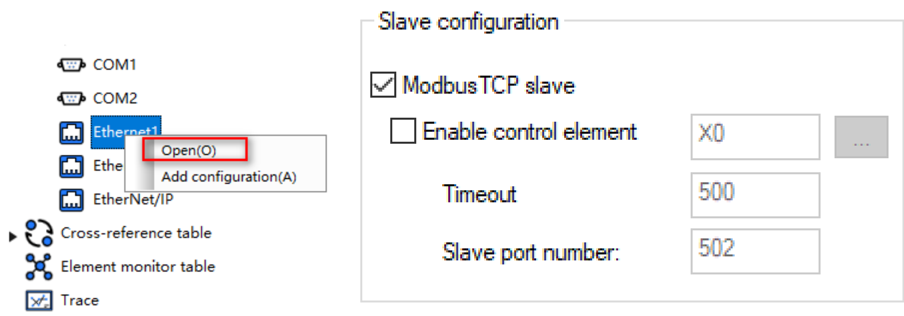
7.7 Modbus TCP Slave

7.7.1 Brief Introduction

Modbus TCP slave is Modbus server, and TS635 Small PLC supports two Modbus TCP slaves, and both Ethernet ports support checking Modbus TCP master and Modbus TCP slave at the same time.

7.7.2 Modbus TCP Slave Settings

Double-click the corresponding "Ethernet" option in the system block to enter the communication port setting interface. Check "Modbus TCP slave" in the pop-up window "Protocol setting".



The configurable content is described as follows:

Options	Setting Content	Note
Check the enable control elements	Enable the enable control elements	This allows users to use bit elements to control the activation and inactivation of Modbus TCP slaves, where each individual slave is controlled. Not used by default.
Enable control elements	X, Y, M elements	Only valid when the Enable Control Element box is checked
Port number	Fixed at 502	-

Note:

- Modbus TCP does not need to set slave number, and all slave numbers can communicate with it.
- After the configuration is completed, download it to TS600 Series Small PLC.
- Other devices can communicate with TS635 using the IP address of the corresponding Ethernet port and port number 502.

7.7.3 Slave Related Information

7.7.3.1 Function Code and Element List

When working as a slave, the supported function codes and operable elements are as follows:

Function code	Name	Type of operable elements	Note
01(0x01)	Read coils	Y, X, M, S, T, C	Read bit
02(0x02)	Read discrete input	X	Read bit
03(0x03)	Read registers	D, Z, T, C, R	Read word
05(0x05)	Write single coil	Y, M, S, T, C	Write bit
06(0x06)	Write single register	D, Z, T, C, R	Write word
15(0x15)	Write multiple coils	Y, M, S, T, C	Write bit
16(0x16)	Write multiple registers	D, Z, T, C, R	Write word

7.7.3.2 Address Binding Table

As a slave, the accessible Modbus protocol addresses are as follows:

Element	Type	Total quantity	Protocol address	Supported function codes (Decimal)	Description
M0–M32767	Bit element	32768	0x0000–0x7FFF (0–32767)	01, 05, and 15	M0 – M999: not saved in case of power-down
S0–S4095	Bit element	4096	0x8000–0x8FFF (32768–36863)	01, 05, and 15	S0–S999: not saved in case of power-down
X0–X1777	Bit element	1024	0xA000–0xA3FF (40960–41983)	01, 02	Input, octal encoded, BOOL type
Y0–Y1777	Bit element	1024	0xB000–0xB3FF (45056–46079)	01, 05, and 15	Output, octal encoded, BOOL type
T0–T399	Bit element	400	0xC000–0xC18F (49152–49551)	01, 05, and 15	● Accuracy 100ms: T0 – T199, 200 points ● Accuracy 10ms: T200–T299, 100 points ● Accuracy 1ms: T300–T399, 100 points
C0–C255	Bit element	256	0xC200–0xC2FF (49664–49919)	01, 05, and 15	● 16-bit ordinary CTUD: C0 – C199, 200 points ● 32-bit ordinary CTUD: C200 – C255, 56 points
D0–D32767	Word element	32768	0x0000–0x7FFF (0–32767)	03, 06, and 16	-
R0–R16383	Word element	16384	0x8000–0xBFFF (32768–49151)	03, 06, and 16	R16384 – R32767 are not mapped to protocol address
T0–T399	Word element	400	0xE000–0xE18F (57344–57743)	03, 06, and 16	● Accuracy 100ms: T0 – T199, 200 points ● Accuracy 10ms: T200–T299, 100 points ● Accuracy 1ms: T300–T399, 100

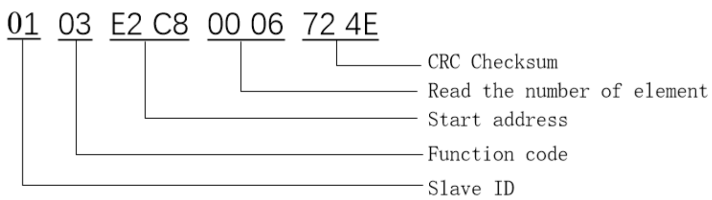
Element	Type	Total quantity	Protocol address	Supported function codes (Decimal)	Description
					points
C0–C199	Word element	200	0xE200–0xE2C7 (57856–58055)	03, 06, and 16	● 16-bit ordinary CTUD: C0 – C199, 200 points
C200–C255	Word element	56	0xE2C8–0xE337 (58056–58167)	03, 06, and 16	● 32-bit ordinary CTUD: C200 – C255, 56 points ● Doubleword elements , every two registers corresponds to one element, and the registers can be read and written in even numbers.
Z0–Z15	Word element	16	0xE400–0xE40F (58368–58383)	03, 06, and 16	-

Note: T and C have word elements and bit elements.

7.7.3.3 Processing of Doubleword Elements

The current count value of C element is a word element or a doubleword element, and C200 – C255 are doubleword elements. The reading and writing of C200 – C255 is also completed by the function codes (03, 16) of read/write registers. The address of every two registers corresponds to a C doubleword element, and the registers can be read and written in pair.

For example, when reading RTU frames of three C doubleword elements from C200 to C202:



In the returned data, the addresses 58056 and 58057 represent the contents of C200, with 58056 being the lower 16 bits and 58057 being the upper 16 bits.

Note: When reading doubleword elements, if the start address of reading is not an even, the abnormal code illegal address will be returned, and if the number of registers read is not an even, the abnormal code illegal data will be returned.

Examples of errors are as follows:

Master transmission: 01 03 E2 C9 00 04 A2 4F

The master transmits four word elements whose reading addresses start from E2 C9 (decimal 58057)

Slave response: 01 83 02 C0 F1

Slave reply: Illegal data address

Master transmission: 01 03 E2 C8 00 05 32 4F

The master reads five word elements whose addresses start from E2 C8 (decimal 58056)

Slave response: 01 83 03 01 31

Illegal data returned from the slave

7.7.3.4 Read SN Code and the Model

When the PLC acts as a Modbus slave, the master can obtain the factory SN code and model information of the PLC through Modbus addresses "65500–65511" (12 words).

The communication messages are as follows:

Master transmission: 00 00 00 00 00 06 01 03 FF DC 00 0C

The master transmits 12 word elements whose reading addresses start from FF DC (decimal 65500)

Slave response: 00 00 00 00 00 1B 01 03 18 54 30 36 31 39 42 30 30 30 32 39 30 00 00 54 53 36 33 35 00 00 00 00 00

Among them, the ASCII code "54 30 36 31 39 42 30 30 30 32 39 30" represents the 12-digit PLC SN code "T0619B000290", while "54 53 36 33 35" represents the PLC model "TS635".

7.8 Modbus TCP Communication Application Example

7.8.1 Brief Introduction

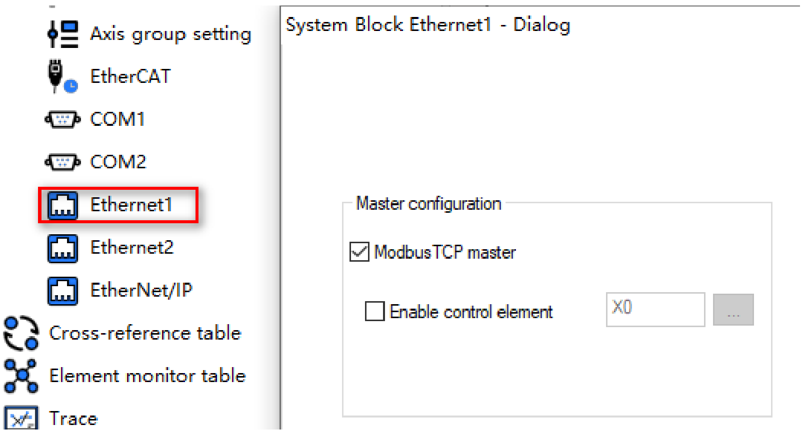
In this example, two sets of TS635 are configured to communicate as Modbus TCP master and slave respectively. The master uses Ethernet 1 and the slave uses Ethernet 2, which are connected by network cables. For convenience of description, the following example assumes that the master number is A and the slave number is B.

7.8.2 Modbus TCP Master Settings

TS635 numbered A is used as the master, and the physical interface is Ethernet 1.

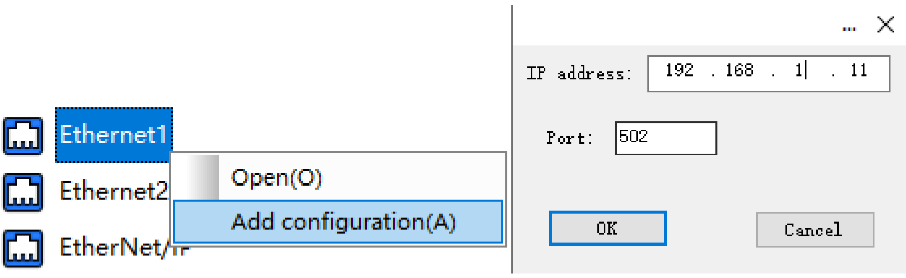
7.8.2.1 Enable TCP Master

Double-click the "Ethernet1" option in the system block to enter the Ethernet port setting interface. Check "ModbusTCP master" in the pop-up window, and then click "OK" to save the configuration.

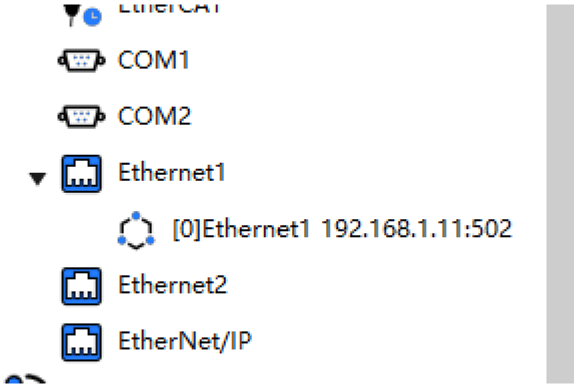


7.8.2.2 Communication parameter configuration

Step 1 Right-click "Ethernet1" > "Add configuration", fill in the slave IP address of 192.168.2.10 and port number of 502, and click "OK" to save the configuration.



The following slave appear for Ethernet 1, representing the communication with it.



Step 2 Double-click "[0]Ethernet1 192.168.2.10:502" to open the Modbus TCP Master Configuration Table.

Step 3 Configure the above communication parameter and download it to "TS635 PLC A" after confirmation.

7.8.3 Modbus-TCP Slave Settings

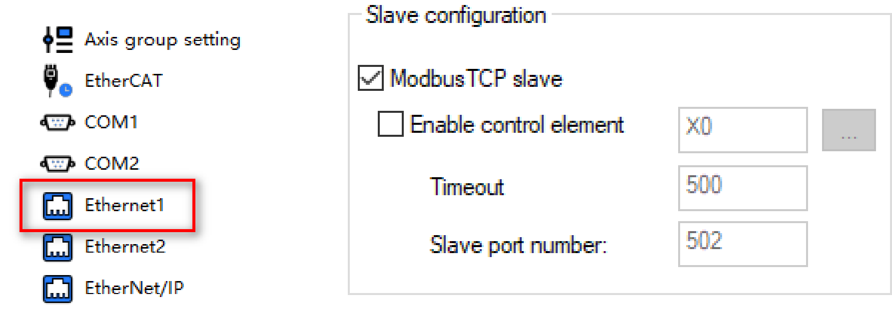
TS635 numbered B is used as the slave, and the physical interface is Ethernet 2.

7.8.3.1 Modify Ethernet IP

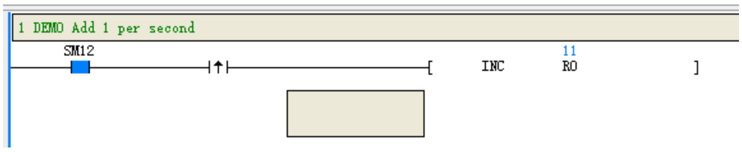
Change the IP address of Ethernet 2 to 192.168.2.10, subnet mask 255.255.255.0, and gateway address 192.168.2.1. For specific operation, refer to section 7.3.2 Modify IP Address in Upper Computer Interface or 7.3.3 Modify IP address with System Variables.

7.8.3.2 Enable TCP Slave

Step 1 Double-click the corresponding "Ethernet" option in the system block to enter the communication port setting interface. Check "ModbusTCP slave" in the pop-up window "Protocol setting".



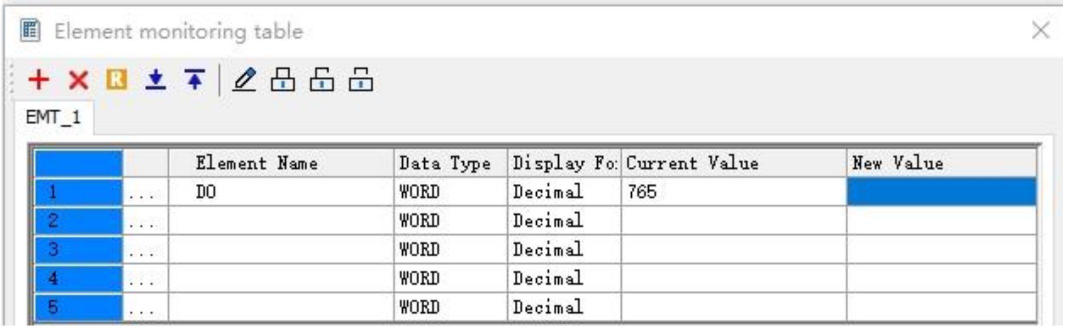
Step 2 Write a program to add 1 to D0 every second.



Step 3 Download it to "TS635 PLC B".

7.8.4 Example Phenomenon

In the element monitoring table, it can be observed that D0 in two PLCs is added by 1 every second.



		Element Name	Data Type	Display Fo:	Current Value	New Value
1	...	D0	WORD	Decimal	765	
2	...		WORD	Decimal		
3	...		WORD	Decimal		
4	...		WORD	Decimal		
5	...		WORD	Decimal		

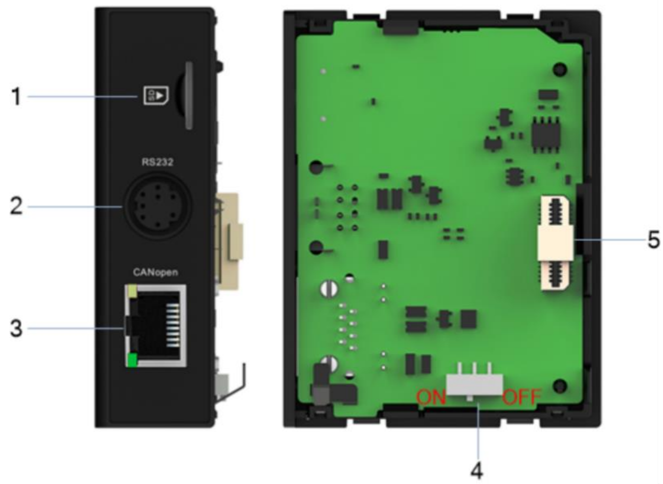
8 CAN Communication

8.1 Overview

TS600 Series PLC realizes CAN communication networking through CAN-232 or CAN-2*RJ45 expansion card, and supports CANopen communication protocol and up to 30 slaves.

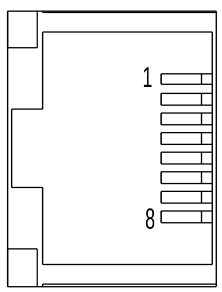
8.2 Hardware Interface

Take CAN-232 expansion card as an example to explain the hardware of CAN port.



No.	Port type	Definition	Description
1	SD card slot	Insert MicroSD card here	Used for SD card firmware upgrade
2	RS232 Interface	RS232 bus interface	Used for RS232 communication
3	CANopen interface	CANopen bus interface	See the specific definition in the later sections
4	Switch	CANopen terminal resistance toggle switch	Toggled to ON: Built-in terminal resistor engaged Toggled to OFF: Built-in terminal resistor disengaged
5	Bus snap interface	Connected with the main module	Connected with PLC main module

CANopen interface definition:

	PIN	Definition Description
	1	CANH
	2	CANL
	3	GND
	4	NC
	5	NC
	6	NC
	7	NC
	8	NC

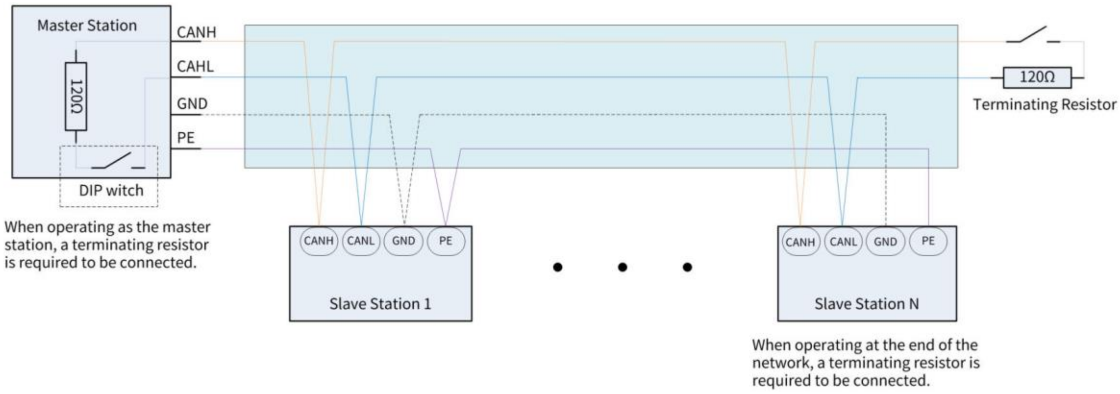
8.3 CAN Network

8.3.1 CAN Communication Networking

For CAN communication networking, all master and slave stacks require three lines (CANH, CANL, CGND) to be connected together one by one. CAN bus terminal matching resistors of 120 ohms need to be added at both ends of the bus. The terminal resistance of TS600 Series expansion card needs to be set with a toggle switch.

The schematic diagram of CAN communication networking is as follows.

Figure 8-1 Wiring Diagram of CAN Communication Networking of Multiple Devices



8.3.2 Communication Distance Corresponding to CAN Baud Rate

The distance of CAN communication is related to its set baud rate, and the comparison table is as follows.

Baud rate (kbit/s)	Distance (m)	Minimum wire diameter (mm^2)	Maximum number of access points
1000	20	0.3	18
500	70	0.3	30
250	140	0.3	30
125	280	0.5	30
100	350	0.5	30
50	700	0.7	30

8.3.3 CAN Interface System Variable

TS600 Series PLC can view or monitor CAN interface status through system variable "_sCAN". "_sCAN" is a struct variable of data type "_struc_CAN" whose member definitions are shown in the following table.

Member	Data Type	Description
BaudRate	INT	Baud rate (kbps)
LoadRate	INT	Load rate (%)
RxPerSec	INT	Frames received per second (FPS)
TxPerSec	INT	Frames transmitted per second (FPS)
RxErrCnt	INT	Receive error counter
TxErrCnt	INT	Transmit error counter
Protocol	INT	Communication protocol 0: Disabled 1: CANopen

8.4 CANopen Communication Instruction

8.4.1 CANopen Communication Protocol

TS600 Series supports CANopen communication standard protocol DS301.

Table 8-1 CANopen Communication Protocol Standard

Software function module	Slave	Master
Supported protocols	DS301 V4.02	DS301 V4.02
Maximum TPDO number	4	64
Maximum RPDO number	4	64
Number of slave nodes	-	30
Baud rate and communication distance	1Mbps/20m 800kbps/40m 500kbps/70m 250kbps/140m 125kbps/280m 100kbps/350m 50kbps/700m	1Mbps/20m 800kbps/50m 500kbps/100m 250kbps/140m 125kbps/280m 100kbps/350m 50kbps/700m
Data interaction software element	R500–R531	D0–D7999 (configurable)

8.4.2 CANopen Axis Control Instruction List

The list of CANopen axis control instructions supported by TS600 Series PLC is as follows. Please refer to TS600 Series PLC Instruction Manual for detailed usage of relevant instructions.

Table 8-2 Instruction List

Instruction Name	Function
MC_Power_CO	Communication control servo axis enabled
MC_Reset_CO	Fault reset of communication control servo axis
MC_ReadStatus_CO	Communication control reads the current state
MC_ReadActualVelocity_CO	Communication control reads the current axis speed position
MC_ReadActualPosition_CO	Communication control reads the current actual axis position
MC_ReadAcceleration_CO	Communication control reads the current axis acceleration
MC_ReadDeceleration_CO	Communication control reads the current axis deceleration
MC_ReadDIStatus_CO	Communication control reads servo DI state
MC_Halt_CO	Communication control servo axis motion halted
MC_Stop_CO	Communication control servo axis stop
MC_MoveVelocity_CO	Communication control axis velocity operation mode
MC_MoveRelative_CO	Communication control axis relative positioning
MC_MoveAbsolute_CO	Communication control axis absolute positioning
MC_Home_CO	Communication control axis homing
MC_Jog_CO	Communication control axis jogging

8.4.3 Explanation of CANopen-related Terms

- NMT: Network Management

Network management services, application layer management, network status management and node ID allocation management. The service mode is master-slave communication mode: In CAN network, there can only be one NMT master and one or more slaves. Mainly used to control the status of slaves.

- SDO: Server Data Object

Service data object, which can access the data in the slave device object dictionary through index and sub-index. This is mainly used for the slave configuration process. Every frame of SDO needs to be replied and confirmed.

- PDO: Process Data Object

Process data object, mainly used to transmit data in real time. Data transmission is limited to 1 to 8 bytes. PDO data transmission is divided into two ways: synchronous and asynchronous. PDO frame is the main data interaction frame after the slave is started.

- SYNC: SynchrONous

Synchronous service adopts master-slave communication mode, where SYNC master node transmits SYNC objects regularly, and SYNC slave node executes tasks synchronously after receiving them. This frame is mainly used for synchronous transmission of PDO.

- COB-ID: CommunicatiON Object Identifier

Each CANopen frame begins with a COB-ID, which serves as the communication object identifier of the CAN frame. COB-ID is not equal to slave number. However, it is generally initialized to be associated with the slave number by default.

8.4.4 CANopen Indicators

When CANopen communicates, the user can judge the working state according to the CANopen indicator.

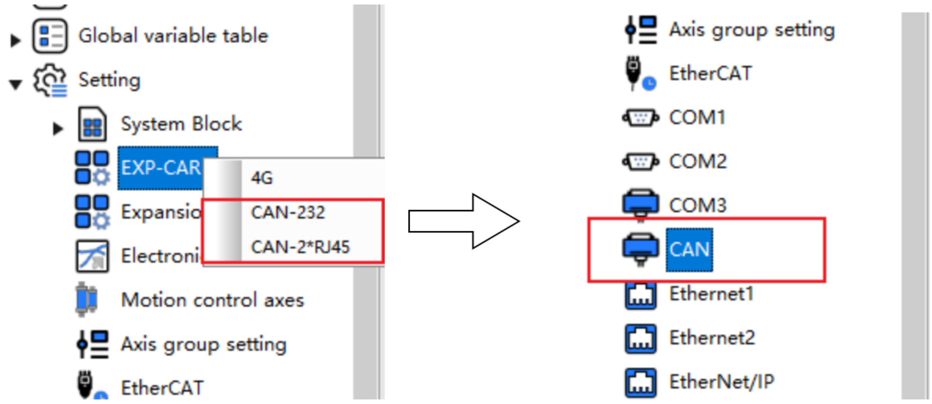
Table 8-3 Definition of CANopen Indicator Status

LED Indication	CAN Running (Green Light)	CAN Error (Yellow Light)
Off	None	No error
On	Working state	Bus off
Slow flash (cycle: 0.8 s)	Pre-running status	Pre-running status
Single flash (cycle: 1.2 s)	Stopped state	At least one error count of the CAN controller reaches or exceeds a warning value
Double flash (cycle: 1.6 s)	None	Error control event (heartbeat timeout)

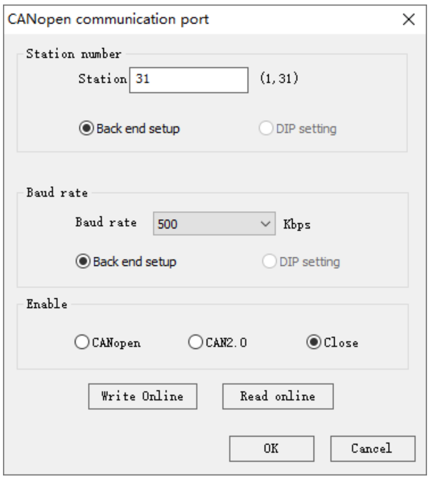
8.5 CANopen Configuration

When "CAN-232" or "CAN-2*RJ45" is selected for the expansion card, the CAN communication configuration option will appear. When the CANopen communication protocol is enabled, the system will decide whether the local computer is the CANopen master or the CANopen slave according to whether there is CANopen configuration.

Step 1 After establishing the project, right-click "Setting>EXP" and select the corresponding physical expansion module "CAN-232" or "CAN-2*RJ45".

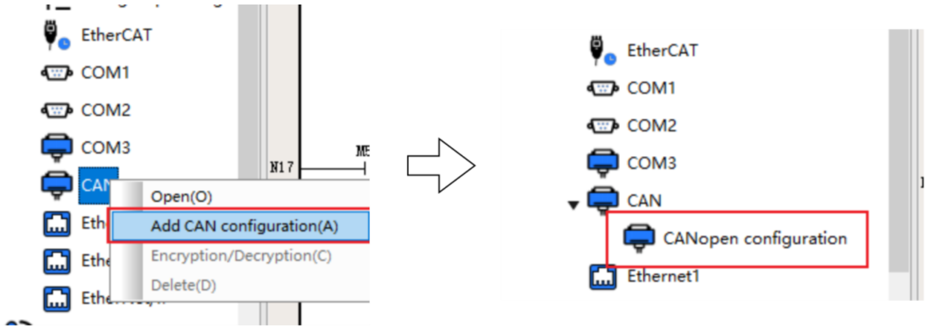


Step 2 Double-click "CAN" in "Setting" to pop up the following window.

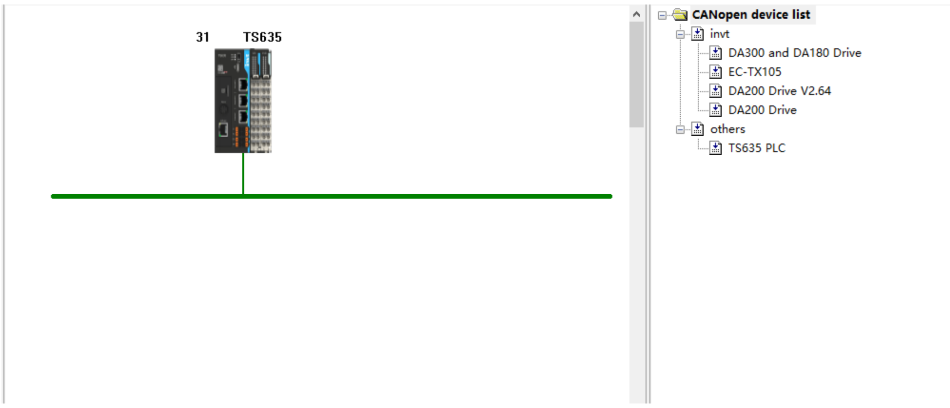


Step 3 Check to enable CANopen, and set the node number and baud rate as required, and then click "OK".

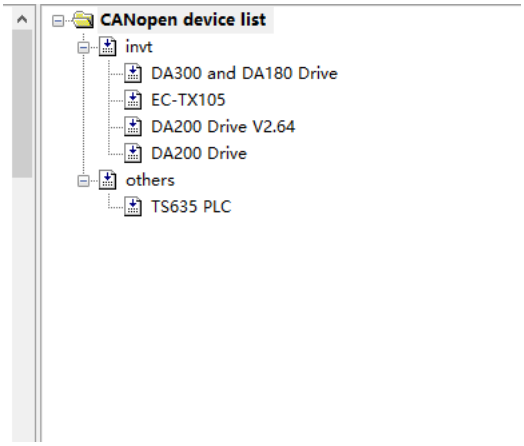
CAN, which is configured as the CANopen slave currently, can be configured as the CANopen master by right-clicking "CAN" in "Setting" and selecting "Add CAN configuration" in the pop-up menu, as shown in the following figure.



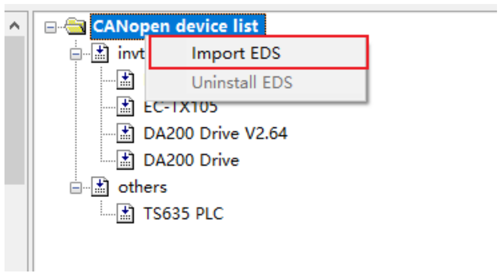
Step 4 Double-click "CANopen configuration" to open the CANopen configuration interface, as shown in the following figure.



Step 5 Double-click the device list to add the CANopen slave.



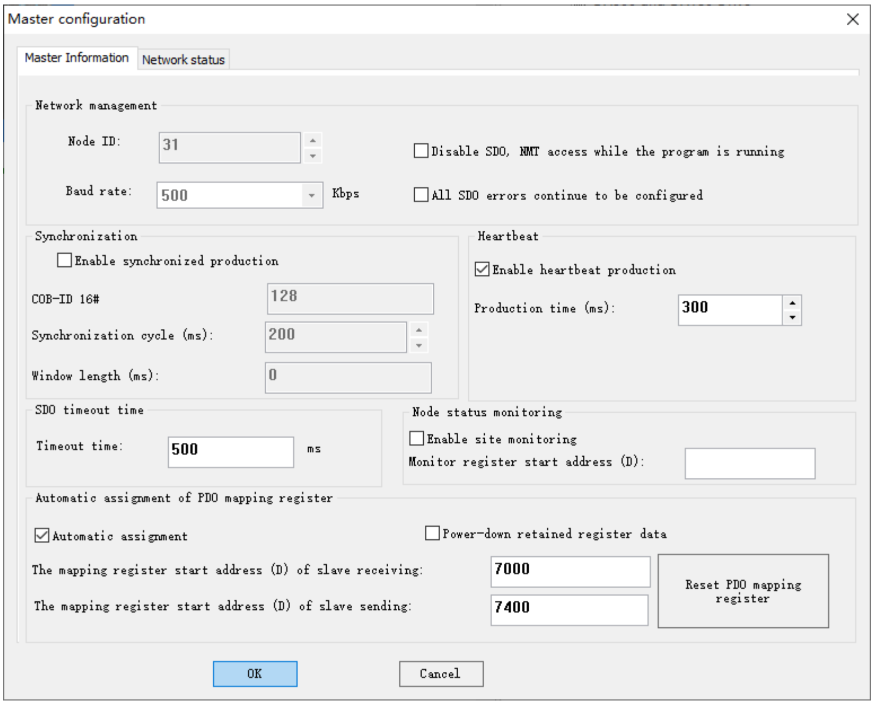
Step 6 If the slave device is not in the list, right-click on the CANopen device list and click to import EDS file (available from the device vendor).



8.5.1 Master Configuration

8.5.1.1 Master Information Interface

Set the master parameters, double-click the TS635 master in the network, and the following window appears.



1. Network management

- "Node ID": Set the network master number.
  - ✧ When the node number is the same as the node number of the PLC itself, the PLC is initialized as the CANopen master.
  - ✧ When the node number is different from the node number of the PLC itself, the PLC is initialized as a CANopen slave.
- "Baud rate": The valid communication baud rate of the master.
- "Disable SDO, NMT access while the program is running": After checking this function, online debugging function will not be used during program running.

  Note: This function only limits background software.
- "All SDO errors continue to be configured":
  - ✧ After checking this function, if SDO is configured incorrectly, it will continue to configure; This function is valid for all slaves.
  - ✧ If this function is not checked, the master will broadcast reset the slave if SDO error occurs.

2. Synchronous

- "Enable synchronized production": When checking this option, the node will transmit synchronization frames according to the time cycle set in "Synchronization cycle (ms)".
- COB-ID: Synchronization frame transmit ID, which defaults to 0x80 and is not allowed to be changed.
- "Synchronization cycle (ms)": The cycle period of sending synchronization frame, default to 200 (unit: ms).
- Window length (ms): This parameter defaults to 0, and is not allowed to be changed.

Note: Only one synchronous message can be sent at the same time in a CANopen network

3. Heartbeat

- "Enable heatbeat production": When checking this option, the node will transmit heartbeat frames according to the time cycle set in "Production time (ms)".
- "Production time (ms)": The cycle period of sending heartbeat, default to 300 (unit: ms).

  Note: The default heartbeat monitoring consumption time of the master is 2.5 times the heartbeat production time.

4. SDO timeout

"Timeout time": SDO wait time, default to 500 (unit: ms).

SDO frames are mainly used for network configuration. If SDO does not receive the return frame on time within before timeout, the master determines the configuration timeout. This time is the waiting interval time of each frame.

5. Node status monitoring

The node's online status is updated to the system variable _sCANOpen.NodeState[64], where _sCANOpen.NodeState[0] is the local state and _sCANOpen.NodeState[站号] is the corresponding slave state.

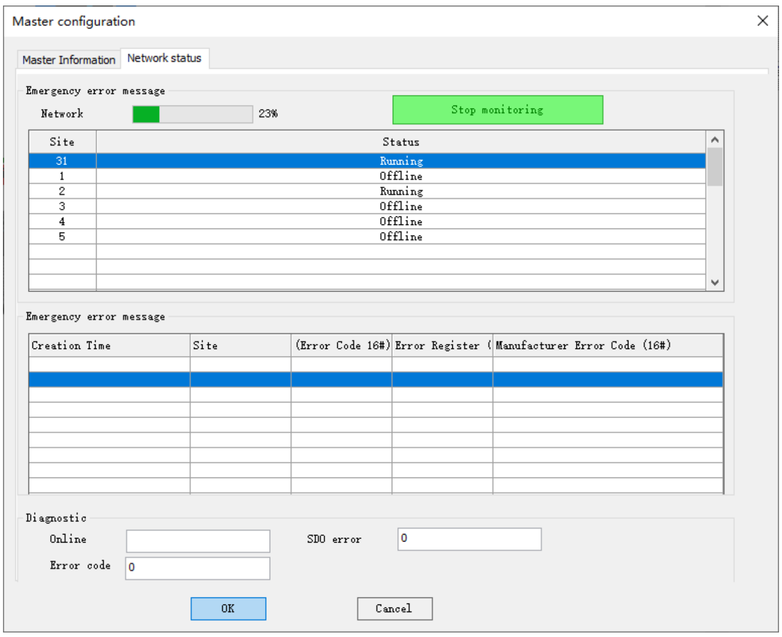
Numerical Value	State
0	Initialization state
4	Stopped state
5	Running state
127	Pre-running status
255	Offline state

Note: This function only makes sense when the slave enables the heartbeat, because this state is fed back by the slave heartbeat.

6. Automatic allocation mapping register

- "Automatic assignment": This function is checked by default.
  - ✧ If this function is checked, the register address for "master-slave" data interaction will be automatically assigned.
  - ✧ If this function is not checked, the user needs to manually set the start address of data interaction (set the start address of each PDO separately).
- "The mapping register start address (D) of slave receiving": The start address of the data sent by the master is automatically allocated (it makes sense only when automatic allocation is checked).
- "The mapping register start address (D) of slave sending": The start address of the data received by the master is automatically allocated (it makes sense only when automatic allocation is checked).

8.5.1.2 Network State



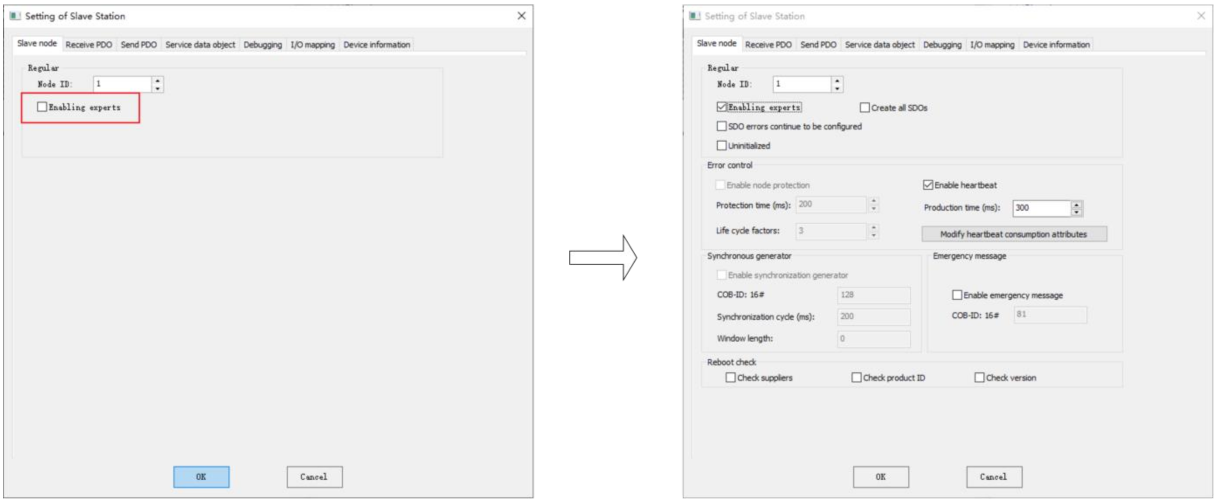
- Start/Stop monitoring: Click on this item to start state monitoring for this page. When monitoring is started, click "Stop Monitoring" to exit network monitoring.
- Network load: Monitor load status of the CAN network in real time.
- Network state table: Displays the running state of the current network nodes. This state comes from the node status system variables.
- Emergency error information
  - ✧ Display the emergency error information in the current network. The PLC master only caches the latest error message. If the page is not closed, up to 5 messages can be cached in the background.
  - ✧ For details about emergency error information, see section 8.6.2.2 Emergency Error Code.
- SDO configuration
  - ✧ Node number: The incorrect node number for SDO configuration.
  - ✧ Error step number: The number of the SDO error. Check the corresponding number information in the "Service Data Object" tab of the slave with the corresponding error parameter.
  - ✧ Error code: The SDO error code. (CANopen standard error code)

8.5.2 Slave Configuration

This section takes EC-TX105 as an example to introduce the configuration process and related parameters of CANopen slaves.

8.5.2.1 General Settings

After checking "Enabling experts", the window as shown on the right below appears.



1. General

- "Node ID": The slave node number to be configured.
- "Enabling experts": Checking this function will display detailed slave configuration. Not checked by default.
- "SDO errors continue to be configured": This option is unchecked by default.
  - ✧ "Valid": Skip to the next configuration item if configuration error occurs.
  - ✧ "Invalid": The master will not proceed with the configuration if configuration error occurs, and the node will be re-configured.
- "Create all SDOs": When this option is selected, all writable object dictionaries in the EDS will be added, initialized during configuration. Not checked by default.

2. Error control

- "Enable heartbeat": When this option is checked, the slave will generate heartbeats. After the slave heartbeat is checked, the master monitors the heartbeat status of the slave by default.
- "Production time (ms)": The cycle time when the heartbeat is sent.
- "Modify heartbeat consumption attributes": This option is not selected by default.
  - ✧ This function is used to set the heartbeats of other nodes that will be monitored by this slave.
  - ✧ This function also requires the slave to support heartbeat monitoring function.

3. Synchronization generator (if supported by slave)

- "Enable synchronization generator": When checking this option, the node will transmit synchronization frames according to the time cycle set in "Synchronization cycle (ms)".
- COB-ID: Synchronization frame transmit ID, which defaults to 0x80 and is not allowed to be changed.
- "Synchronization cycle (ms)": The cycle period of sending synchronization frame, default to 200 (unit: ms).
- "Window length": This parameter defaults to 0, and is not allowed to be changed.

Note: Only one synchronization frame can be sent in a CANopen network.

4. Emergency packets

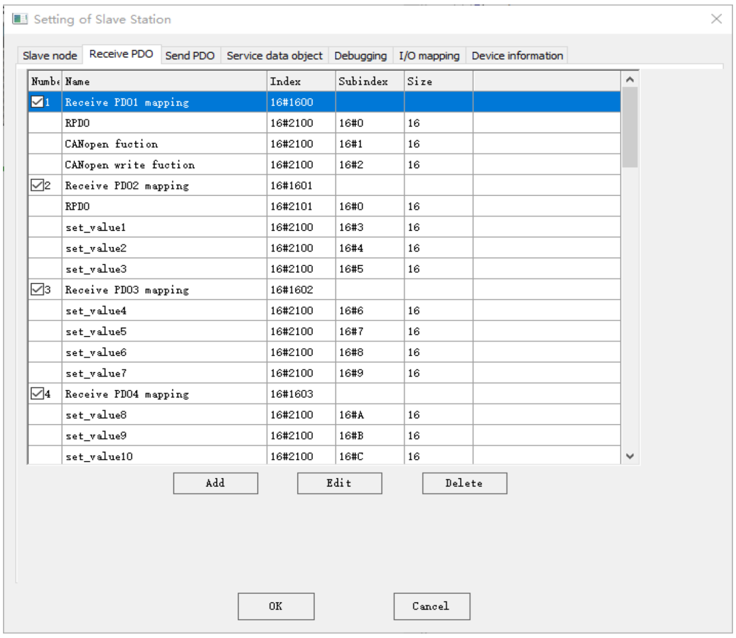
- "Enable emergency message": If this function is checked, the COB-ID of emergency packets will be set in the configuration process, but it is not checked by default.

5. Check on restart

- "Check suppliers, Check product ID, Check version": When checking the corresponding function, the relevant check will be carried out before starting slave configuration. If the check fails, the node cannot be started.

8.5.2.2 Receive PDO/Transmit PDO

Click to select receive PDO/transmit PDO, and the interface will appear:



- Receive PDO: master -> slave data
- Transmit PDO: slave -> master data

8.5.3 PDO Enabling

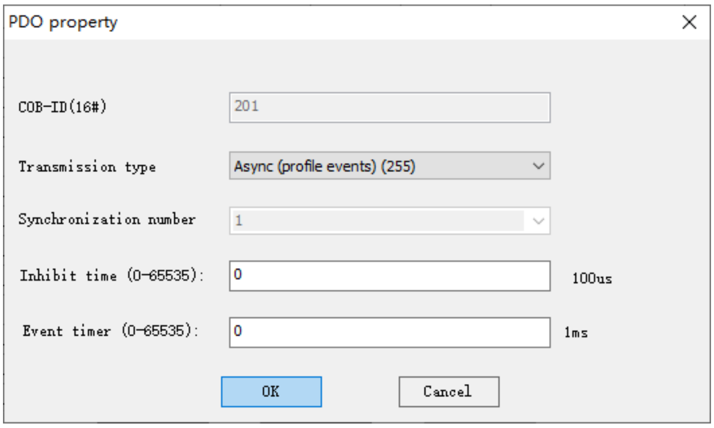
- The check box before the number is used to select whether this PDO is valid.
- If this PDO does not contain mapping objects, it cannot be checked.
- The PDO that takes effect by default in the EDS file of the slave has been checked by default.

8.5.4 PDO Mapping Editing

Used to edit the PDO mapping through the "Add PDO mapping", "Edit" and "Delete" buttons in the window.

8.5.5 PDO Attribute Settings

The PDO attribute interface appears as follows:



- COB- ID: ID number used for PDO transmission.
- Transmission type

Type	Data Transmission Conditions	Data Effective Conditions
Acyclic-synchronous (Type 0)	Data changes and a synchronization frame is received	It does not take effect immediately after receiving data, and it takes effect only after receiving a synchronization frame
Cyclic-synchronous (Type 1-240)	The data is transmitted after receiving the synchronization frame of the corresponding "synchronization number"	It does not take effect immediately after receiving data, and it takes effect only after receiving one frame synchronization
Asynchronous-Manufacturer-specific (Type 254)	Customized by each manufacturer	Customized by each manufacturer
Asynchronous-Device Profile-specific (Type 255)	The time when data changes or meets the event, and the change frequency is less than the suppression time	Effective immediately

Note: When setting up a certain node for synchronous transmission, it is necessary to enable synchronous production of the master

- Synchronization number

Used to set the synchronization number, only valid after selecting Cyclic-synchronous (Type 1-240).

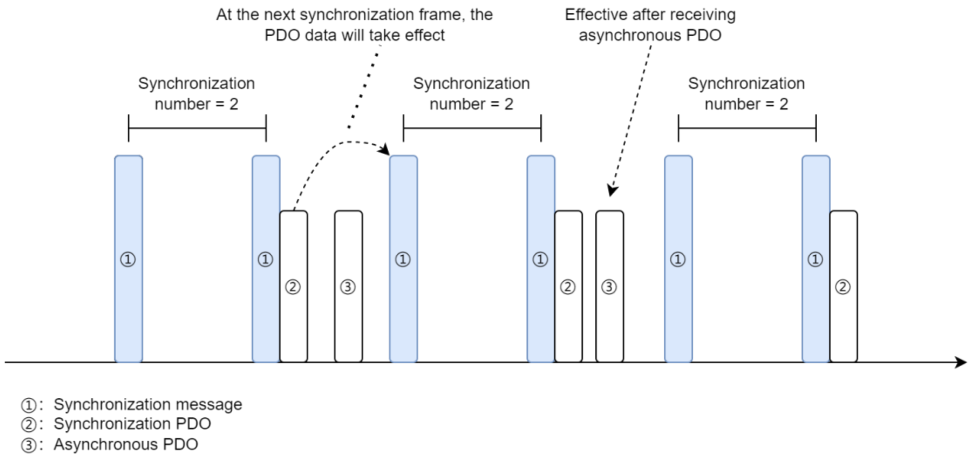
- Inhibition time

Available only after selecting Asynchronous-Device Profile-specific (Type 255). This function is invalid when set to 0; If it is not 0, it indicates the minimum interval for frame transmission.

- Event time

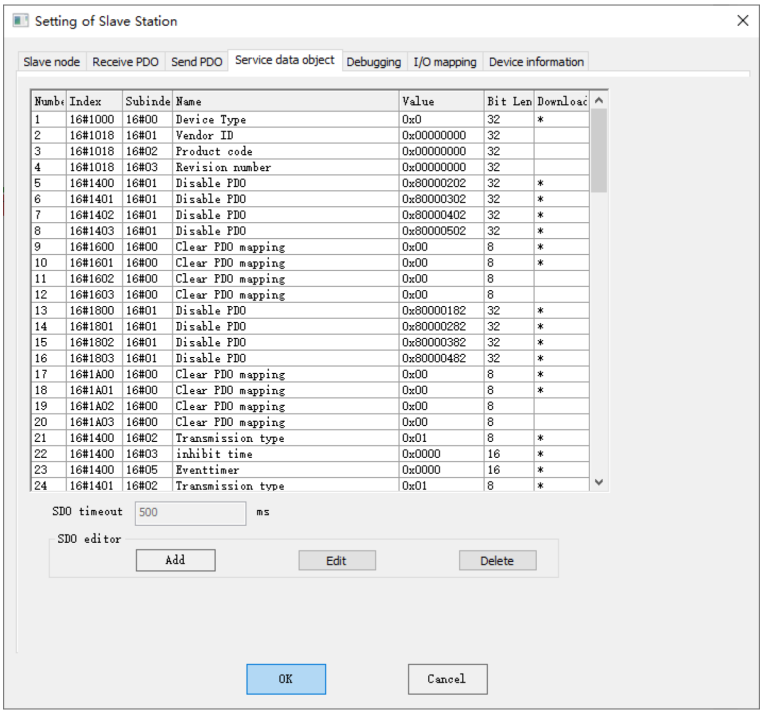
Available only after selecting Asynchronous-Device Profile-specific (Type 255). This function is invalid when set to 0. If it is not 0, it indicates the period of regular transmission (This transmission condition is also limited by the inhibition time).

The following figure takes Synchronous cycle-synchronous type = 2 as an example.



8.5.6 Service Data Object

The Service Data Object (SDO) tab interface is shown below (using DA200 as an example). The table automatically generates SDO configuration data based on the EDS file and user settings.



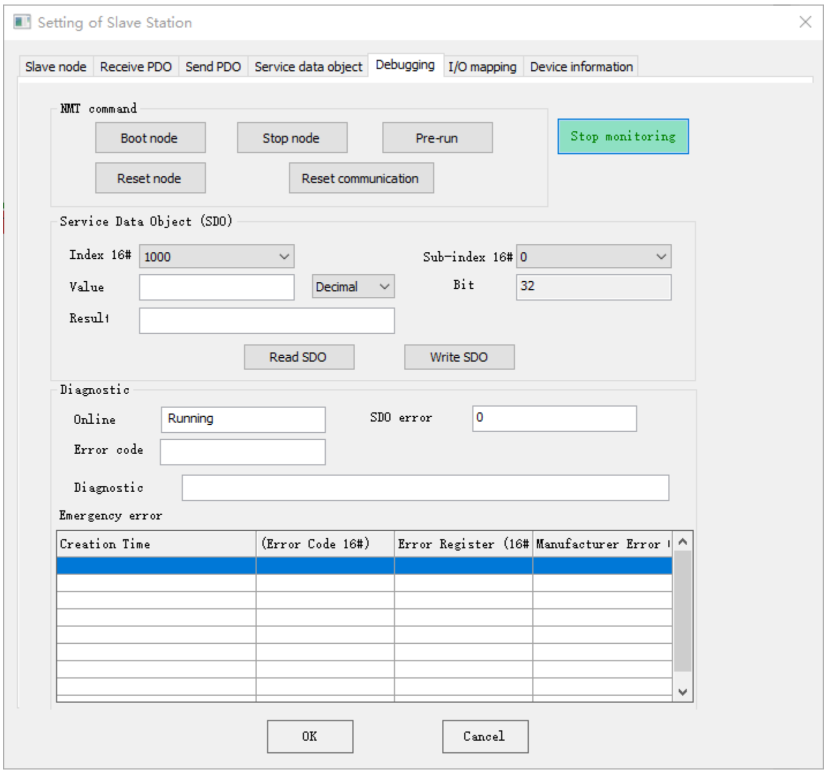
Numbe	Index	Subinde	Name	Value	Bit Len	Downloac
1	16#1000	16#00	Device Type	0x0	32	*
2	16#1018	16#01	Vendor ID	0x00000000	32	
3	16#1018	16#02	Product code	0x00000000	32	
4	16#1018	16#03	Revision number	0x00000000	32	
5	16#1400	16#01	Disable PDO	0x80000202	32	*
6	16#1401	16#01	Disable PDO	0x80000302	32	*
7	16#1402	16#01	Disable PDO	0x80000402	32	*
8	16#1403	16#01	Disable PDO	0x80000502	32	*
9	16#1600	16#00	Clear PDO mapping	0x00	8	*
10	16#1601	16#00	Clear PDO mapping	0x00	8	*
11	16#1602	16#00	Clear PDO mapping	0x00	8	
12	16#1603	16#00	Clear PDO mapping	0x00	8	
13	16#1800	16#01	Disable PDO	0x80000182	32	*
14	16#1801	16#01	Disable PDO	0x80000282	32	*
15	16#1802	16#01	Disable PDO	0x80000382	32	*
16	16#1803	16#01	Disable PDO	0x80000482	32	*
17	16#1A00	16#00	Clear PDO mapping	0x00	8	*
18	16#1A01	16#00	Clear PDO mapping	0x00	8	*
19	16#1A02	16#00	Clear PDO mapping	0x00	8	
20	16#1A03	16#00	Clear PDO mapping	0x00	8	
21	16#1400	16#02	Transmission type	0x01	8	*
22	16#1400	16#03	inhibit time	0x0000	16	*
23	16#1400	16#05	Eventtimer	0x0000	16	*
24	16#1401	16#02	Transmission type	0x01	8	*

The SDO editing options are described as follows:

- Add: Adds user configurations. The main function is to assign initial values to the object dictionary.
- Edit: Re-edits user configuration.
- Delete: Deletes user configuration.

8.5.7 Online Debugging Function

The online debugging tab interface is displayed as follows.

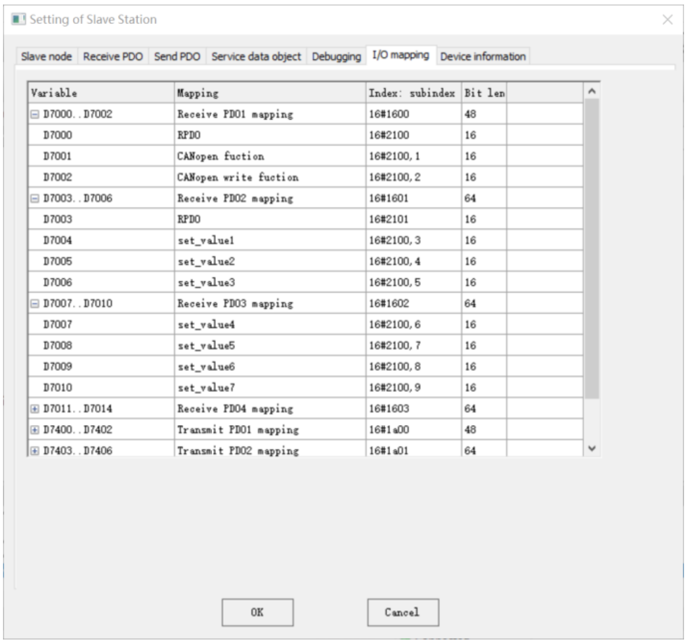


Note: If "Disable SDO, NMT access while the program is running" is selected in the master, this function cannot be used.

- NMT instructions
  - ✧ Start node: Send a start node instruction to the slave.
  - ✧ Stop node: Send a stop node instruction to the slave.
  - ✧ Pre-operation: Send a pre-operation instruction to the node.
  - ✧ Reset node: Send a reset node instruction to the node.
  - ✧ Reset communication: Send a reset communication instruction to the node.
- SDOs
  - ✧ Index and subindex: Only the object dictionary provided in the slave EDS can be selected.
  - ✧ Value: The data sent or returned.
  - ✧ Bit length: Obtained from the slave EDS object dictionary.
  - ✧ Result: Read and write state information.
  - ✧ Read SDO, write SDO: Perform the read and write operations of corresponding object dictionary.
- Diagnostics
  - ✧ Online state: The current state of the slave (based on heartbeats).
  - ✧ SDO error step number: The error number of SDO that occurred during the configuration process. This number corresponds to the number in the "Service Data Object" tab.
  - ✧ Diagnostic string: Current error message (SDO error). For details, see section 8.6.2.1 SDO Error Code.

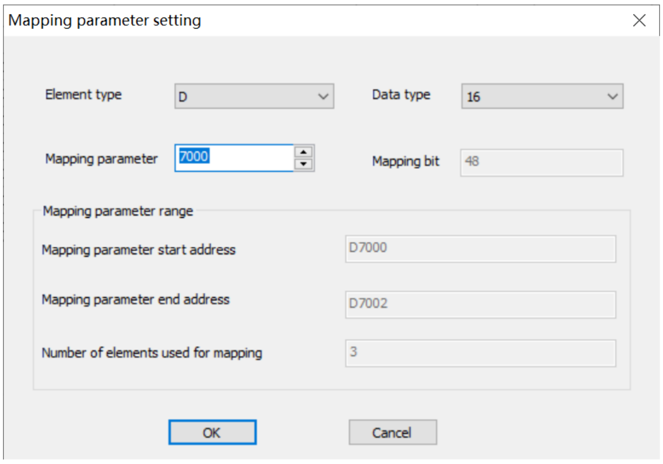
8.5.8 I/O Mapping

The I/O mapping option interface appears as follows.



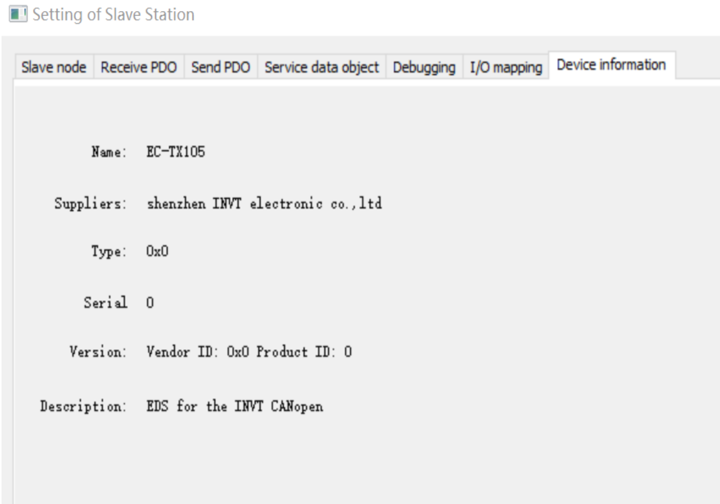
Variable	Mapping	Index: subindex	Bit len
D7000..D7002	Receive PDO1 mapping	16#1600	48
D7000	RPDO	16#2100	16
D7001	CANopen fuction	16#2100, 1	16
D7002	CANopen write fuction	16#2100, 2	16
D7003..D7006	Receive PDO2 mapping	16#1601	64
D7003	RPDO	16#2101	16
D7004	set_value1	16#2100, 3	16
D7005	set_value2	16#2100, 4	16
D7006	set_value3	16#2100, 5	16
D7007..D7010	Receive PDO3 mapping	16#1602	64
D7007	set_value4	16#2100, 6	16
D7008	set_value5	16#2100, 7	16
D7009	set_value6	16#2100, 8	16
D7010	set_value7	16#2100, 9	16
D7011..D7014	Receive PDO4 mapping	16#1603	64
D7400..D7402	Transmit PDO1 mapping	16#1a00	48
D7403..D7406	Transmit PDO2 mapping	16#1a01	64

This tab is used to set the data communication mapping relationship between master and slave PDO. If Automatic Allocation is not checked in the master settings, the user can double-click one of the mappings to set it, as shown in the following figure. The user can set the master register start address corresponding to each salve PDO by himself.



8.5.9 Device Information

The "Device information" tab has an interface as shown below:

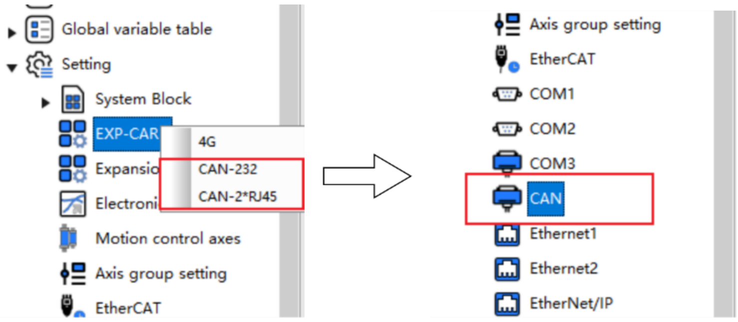


The device information of the slave is obtained from the EDS file provided by the device manufacturer.

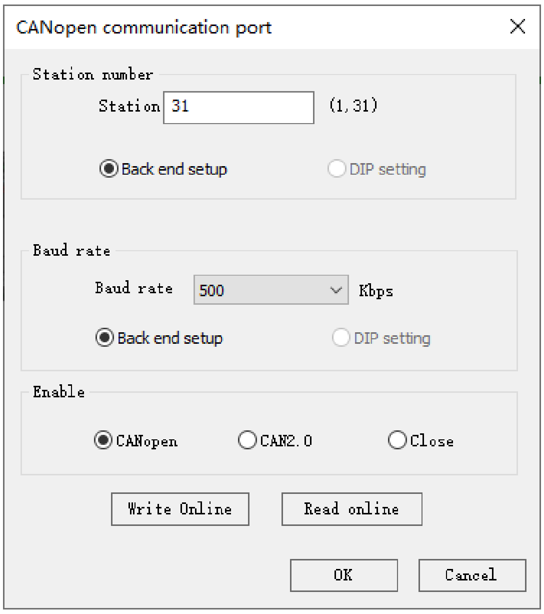
8.5.10 CANopen Slave Description

8.5.10.1 Slave Configuration Procedures

Step 1 After establishing the project, right-click "Setting>EXP" and select the corresponding physical expansion module "CAN-232" or "CAN-2*RJ45".



Step 2 Double-click "CAN" in "Setting" to pop up the following window.



Step 3 Check to enable "CANopen", and set the node number and baud rate as required, and then click "OK" to complete the slave configuration.

Note: Do not continue to add CAN configuration by right-clicking, otherwise this machine will act as the CANopen master.

8.5.10.2 Slave CANopen Data Interaction

The TS600 series programmable logic controller, when operating as a slave, only supports 4 TPDOs and 4 RPDOs with each PDO supporting a maximum of 8 bytes of data. The object dictionary of the slave is set with 16 16-bit mapped addresses for caching transmitted data and another 16 16-bit mapped addresses for caching received data. They are respectively bound to the corresponding slave R500–R531 elements, as shown below.

Table 8-4 Reference for Data Received

Index	Sub-index	Name	Access type	Data type	Slave bind elements
2109	1	1st Rx Buffer	RW	16	R500
2109	2	2st Rx Buffer	RW	16	R501
2109	3	3st Rx Buffer	RW	16	R502
2109	4	4st Rx Buffer	RW	16	R503
2109	5	5st Rx Buffer	RW	16	R504
2109	6	6st Rx Buffer	RW	16	R505
2109	7	7st Rx Buffer	RW	16	R506
2109	8	8st Rx Buffer	RW	16	R507
2109	9	9st Rx Buffer	RW	16	R508
2109	10	10st Rx Buffer	RW	16	R509
2109	11	11st Rx Buffer	RW	16	R510
2109	12	12st Rx Buffer	RW	16	R511
2109	13	13st Rx Buffer	RW	16	R512
2109	14	14st Rx Buffer	RW	16	R513
2109	15	15st Rx Buffer	RW	16	R514
2109	16	16st Rx Buffer	RW	16	R515

Table 8-5 Reference for Data Transmitted

Index	Sub-index	Name	Access type	Data type	Slave bind elements
2108	1	1st Tx Buffer	RW	16	R516
2108	2	2st Tx Buffer	RW	16	R517
2108	3	3st Tx Buffer	RW	16	R518
2108	4	4st Tx Buffer	RW	16	R519
2108	5	5st Tx Buffer	RW	16	R520
2108	6	6st Tx Buffer	RW	16	R521
2108	7	7st Tx Buffer	RW	16	R522
2108	8	8st Tx Buffer	RW	16	R523
2108	9	9st Tx Buffer	RW	16	R524
2108	10	10st Tx Buffer	RW	16	R525
2108	11	11st Tx Buffer	RW	16	R526
2108	12	12st Tx Buffer	RW	16	R527
2108	13	13st Tx Buffer	RW	16	R528
2108	14	14st Tx Buffer	RW	16	R529
2108	15	15st Tx Buffer	RW	16	R530
2108	16	16st Tx Buffer	RW	16	R531

8.6 Troubleshooting of CANopen Communication

8.6.1 Common Troubleshooting Steps

- Check the matching resistance
  - ✧ Disconnect the power supply of all devices, and measure the resistance between CANH and CANL at either end of the network with a multimeter, which should be about 60 Ω. If it is too small, it means that the matching resistors are not only connected at both ends of the network, but also wrongly connected at other positions which shall be disconnected.
  - ✧ If only one matching resistor is connected, it will be about 120 Ω, and the communication quality of the network will be very poor.
  - ✧ If no resistor is connected at all, the network cannot communicate. Make sure to connect the matching resistors at both ends of the network.
- Check baud rate
  - ✧ Check whether the baud rate is normal, and all nodes in the network can communicate normally only if the baud rate is set to the same level.
  - ✧ The baud rate of device will take effect only after power cycle or restart.

  For the relationship between communication distance and baud rate, please refer to section 8.3.2 Communication Distance Corresponding to CAN Baud Rate.
- Check the wiring
  - ✧ The CGND terminals of all CAN devices must be connected together so that all devices share the CGND terminals of CAN communication power supply.
  - ✧ Check whether there is short circuit in communication wire, shielded wire and power supply.
- Other
  - ✧ If there is severe interference on site, please try to reduce the communication baud rate when there is no way to troubleshoot.

8.6.2 Fault Code List

8.6.2.1 SDO Error Code

Interruption code	Code function description	Interruption code	Code function description
0503 0000	Triggering bit not alternated	0601 0002	Attempts to write information to a read-only object
0504 0000	SDO protocol times out	0602 0000	Object cannot be found in the object dictionary
0504 0001	Illegal or unknown Client/Server instruction word	0604 0041	Object cannot be mapped to PDO
0504 0002	Invalid block size (Block Transfer mode only)	0604 0042	Number and length of the object to be mapped exceeds the PDO length
0504 0003	Invalid serial number (Block Transfer mode)	0604 0043	Common parameter incompatibility
0503 0004	CRC Error (Block Transfer mode)	0604 0047	Common internal incompatibility of the device
0503 0005	Memory overflow	0606 0000	Object access failure caused by

Interruption code	Code function description	Interruption code	Code function description
			hardware error
0601 0000	No access to the object	0607 0010	Data type not matched; service parameter length not matched
0601 0001	Attempts to read a write-only object	0607 0012	Data type mismatch, service parameter length is too long
0601 0002	Attempts to write information to a read-only object	0607 0013	Data type mismatch, service parameter length is too short
0602 0000	Object cannot be found in the object dictionary	0609 0011	Subindex does not exist
0604 0041	Object cannot be mapped to PDO	0609 0030	Out of value range of parameter (on write access)
0800 0000	Common error	0609 0031	Written parameter value too large
0800 0020	Data failed to be transmitted or stored in the application	0609 0032	Written parameter value too small
0800 0022	Data failed to be transmitted or stored in the application due to the current state of the device	0609 0036	Max. value less than Min. value

8.6.2.2 Emergency Error Code

Table 8-6 Main Table 1 (Hexadecimal)

Emergency Error Code	Description	Emergency Error Code	Description
00xx	Error reset or no error	50xx	Hardware error
10xx	General error	60xx	Device software
20xx	Current error	61xx	Internal software
21xx	Device input	62xx	User software
22xx	Internal equipment	63xx	Data setting
23xx	Device output	70xx	Auxiliary device error
30xx	Voltage error	80xx	Monitor error
31xx	Main power supply	81xx	Communication
32xx	Internal equipment	82xx	Protocol error
33xx	Output	90xx	External error
40xx	Temperature error	F0xx	Additional function error
41xx	Environment	FFxx	Device specific error
42xx	Device	-	-

Table 8-7 Main Table 2 (Hexadecimal)

Emergency Error	Description	Emergency Error Code	Description
0000	Error reset or no error	6300	Data setting
1000	General error	7000	Additional module error
2000	Current error	8000	Monitoring error
2100	Device input current	8100	General communication error
2200	Internal device current	8110	CAN communication overload
2300	Device output current	8120	CAN passive mode error

Emergency Error	Description	Emergency Error Code	Description
3000	Voltage error	8130	Node protection or heartbeat error
3100	Power supply voltage	8140	Bus off restore
3200	Internal device voltage	8150	CAN-ID conflict
3300	Output voltage	8200	Protocol error
4000	Temperature error	8210	PDO length error
4100	Ambient temperature	8220	PDO length overrun
4200	Device temperature	8240	Unrecognized SYNC data length
5000	Device hardware error	8250	RPDO timeout
6000	Device software error	9000	External error
6100	Internal error	F000	Additional function error
6200	User software	FF00	Special device error

8.7 CAN Free Protocol

8.7.1 Function Introduction of CAN Free Format

The TS600 PLC supports one CAN free format communication with extended frames and data frames, specifically divided into standard frames (CAN2.0A) and extended frames (CAN2.0B) according to the standard protocol types.

- The frame ID length of the standard frame is 11 bits, which means the frame ID range is 000–7FF.
- The frame ID length within the extended frame is 29 bits, which means the frame ID range is 00000000–1FFFFFFF.

The TS600 can exchange free format data with other devices in the CAN network that support the CAN2.0A or CAN2.0B protocol.

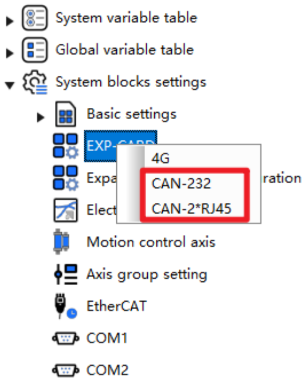
8.7.2 Communication Specifications of CAN Free Format

Item	Specification
Connector	Network port
Transmission cable	Common network cable
Supported standard protocols	CAN2.0A (11-bit identifier) CAN2.0B (29-bit identifier)
Max. number of communication codes	64
Baud rate	20, 50, 100, 125, 250, 500, 800, 1000 kbit/s
Buffer bits	16-bit

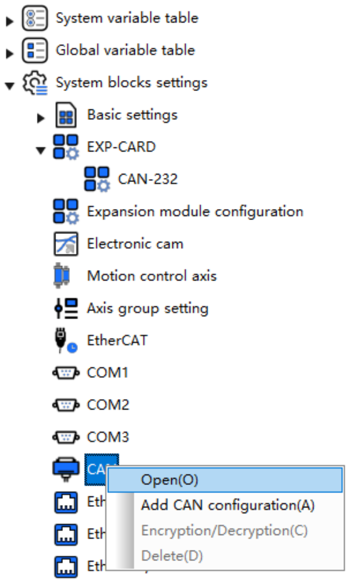
8.7.3 CAN Free Port Configuration

When using the CAN free port, it is necessary to configure the CAN port communication method. The steps are as follows:

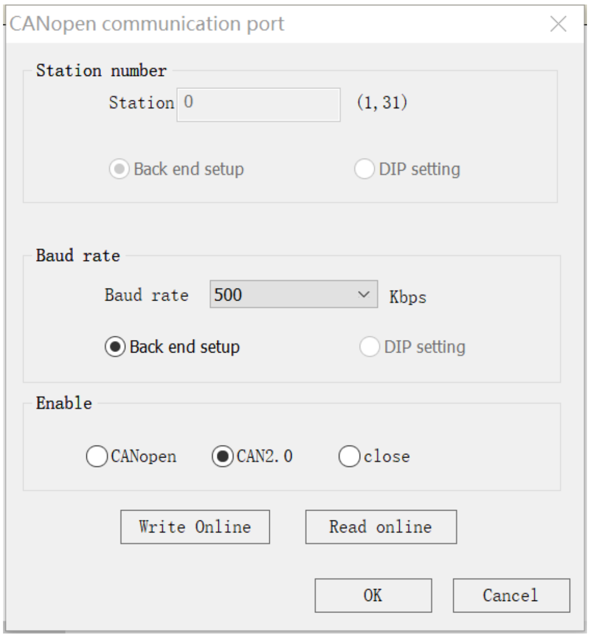
Step 1 Configure the corresponding CAN card.



Step 2 Open the CAN configuration.



Step 3 Configure the "Baud Rate" and select "CAN2.0A," then click "OK."



8.7.4 CAN Free Format Data Sending and Receiving

For details, refer to section 3.23.4 "CANfree_Recv" and section 3.23.5 "CANfree_Send" in the "TS600 Series Programmable Logic Controller Command Manual."

9 EtherCAT Communication

9.1 Overview

EtherCAT is an open industrial field technology based on Ethernet, and has the characteristics of short communication refresh period, small synchronization jitter and low hardware cost. To understand the principle and related technologies of EtherCAT, please refer to the book Driver Design and Application of Industrial Ethernet Fieldbus EtherCAT, or log in to official website of EtherCAT Technical Committee at https://www.EtherCAT.org.cn.

TS600 Series supports standard EtherCAT interface (1-way RJ45 interface), up to 72 EtherCAT slaves, adopts linear topology structure, and the minimum EtherCAT fieldbus cycle can be set to 250 µs.

Table 9-1 EtherCAT Interface Specification

Entry	Specification
Transmission speed	100Mbps: 100BASE-TX
Modulation	Baseband
Topology	Line, daisy chain
Transmission medium	Twisted pair of category 5 or above or shielded twisted pair with aluminum foil and braided shielding
Transmission distance	Distance between nodes: 100m or less
Maximum connections	72

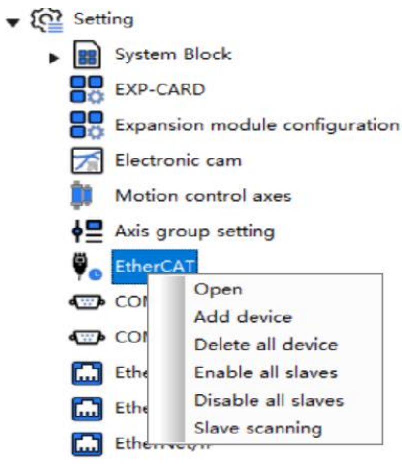
9.2 Master Configuration

9.2.1 Import Device XML

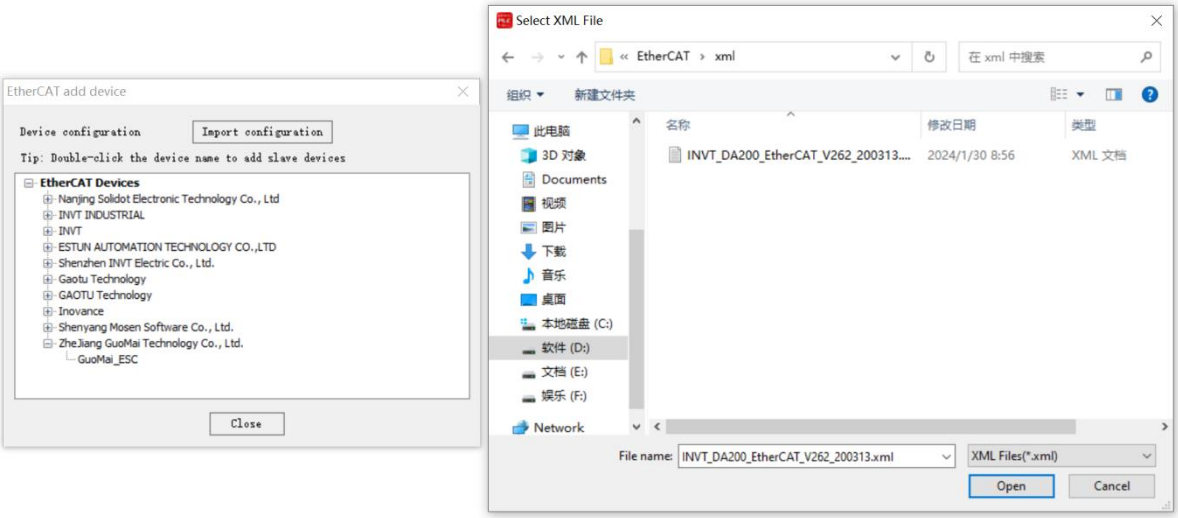
Import device XML means importing the device description file (suffixed by ". XML") conforming to ETG (EtherCAT Technical Committee) standard into the programming software AutoStation Pro, and then generating EtherCAT configuration devices for users to add, delete, etc. after the files are parsed in the software.

Auto Station Pro, the programming software, integrates INVT's common-used EtherCAT slave devices which do not need to be installed separately. If a third-party EtherCAT device is required, the device description file must be installed first. Take the import process of INVT servo driver DA260 as an example.

Step 1 Create a new project, set it up, find "EtherCAT", and right-click on on "Add device ".

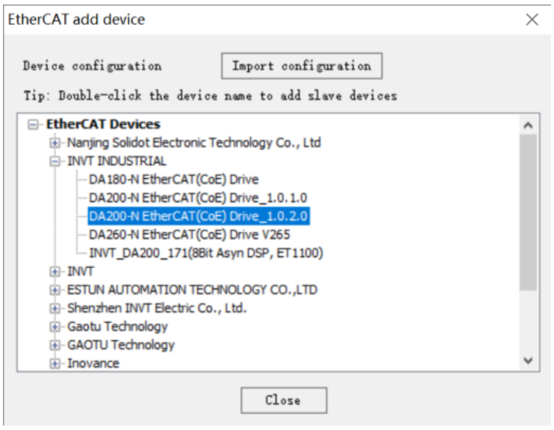


Step 2 Click "Import configuration", and in the pop-up dialog box, select the XML file to be added and import it.



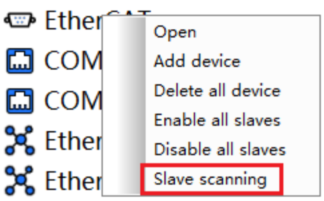
Step 3 The software must be restarted for the newly imported xml to take effect. If multiple devices are required to be imported, you can repeat step 2 to import them all before restarting the software.

Step 4 You can see the newly added device after reopening the software

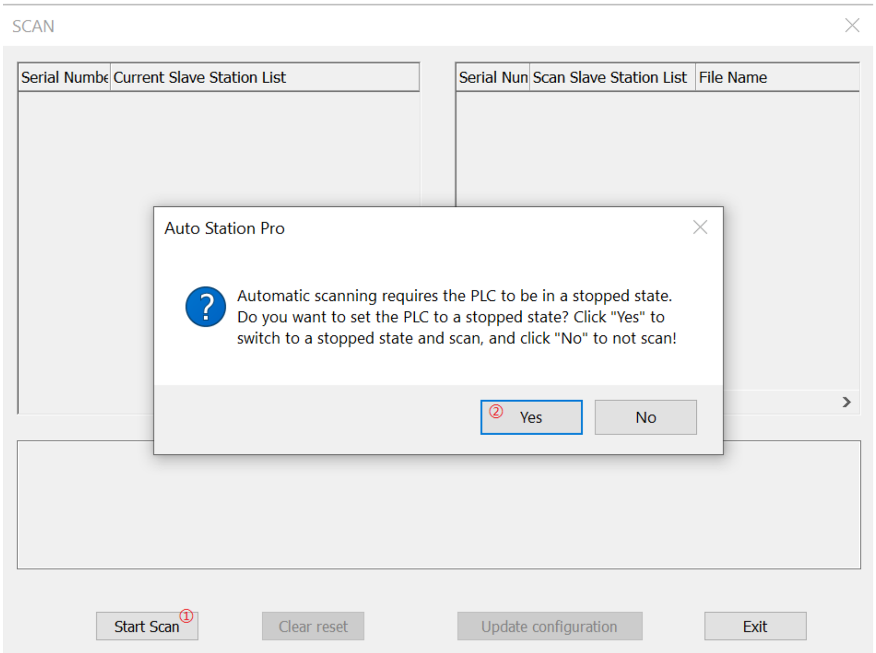


9.2.2 Scanning Device

Step 1 Right-click on "EtherCAT" > "Slave Scan".

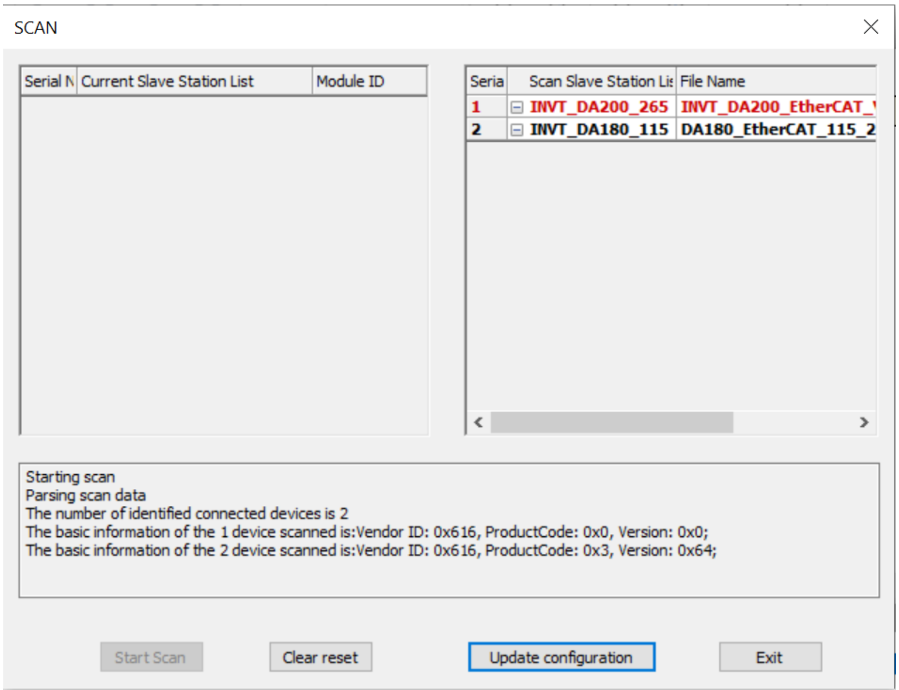


Step 2 In the pop-up interface, click "Start Scanning".

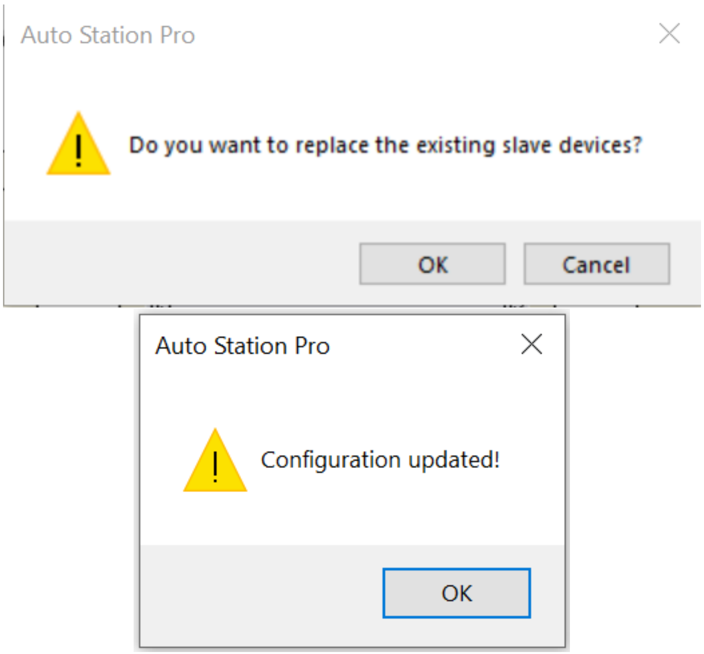


Step 3 If there are devices on the bus, the corresponding slaves will be scanned in the order of the slaves. The "Alias" can modify the device file. When you click "Update configuration," a prompt "Do you want to replace the existing slave devices?" will appear. Click "OK" to complete the configuration update while click "Cancel" to add a new device based on the existing configuration. If there are no connectable devices on the bus or the scan times out, the "Information output" window will display the message "Scan state query timedout".

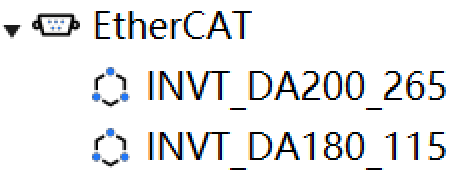
Below is an example of scanning when the master is connected to two slaves:



Step 4 After clicking "Update configuration," the following options pop up.

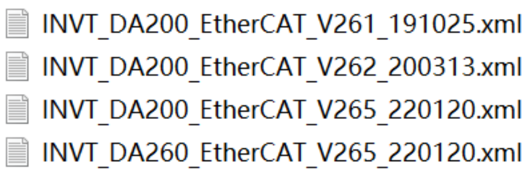


Step 5 The slave has been successfully added.



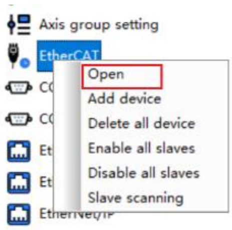
Note:

- The EtherCAT slaves can be scanned only when the PLC is in stop mode.
- For situations where the same type of device has been added but there are multiple different versions of the xml file, the scanning function program selects the first device as the configuration device by default.



9.2.3 Master Setting

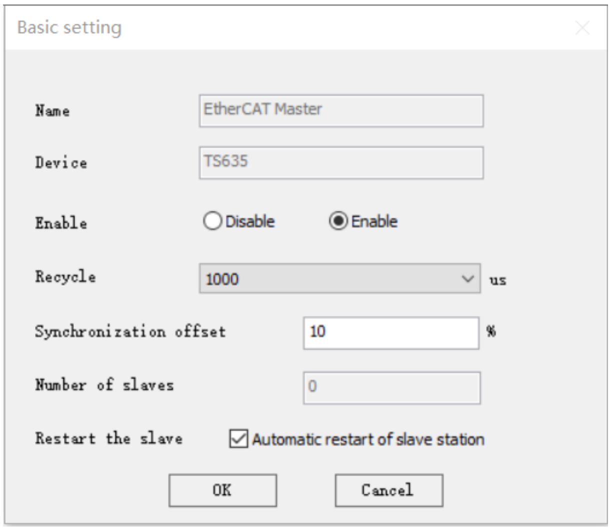
Step 1 Right-click on "EtherCAT" > "Open".



Step 2 "Basic setting" pops up. In the "Basic setting" interface, you can set the enable control, cycle time and synchronization offset of EtherCAT master.

- "Enable": When the EtherCAT master is disabled, all EtherCAT slaves are disabled, and the fieldbus servo axis associated with the slaves will no longer operate.
- "Recycle": EtherCAT data frame transmission interval and cycle time of EtherCAT task.

- "Synchronization offset": Percentage of the relative offset of the EtherCAT task relative to Sync0 interrupt of the slave.
- Restart the slave: When checked, the slave will automatically restart if it is offline. When unchecked, the slave will not restart if it is offline.



9.2.4 Start-stop & Disable & Enable

- **Start/stop Control**

It supports the start-stop of the EtherCAT fieldbus, but does not support the start-stop of a single slave. The related operations are as follows:

- ✧ When PLC switches from STOP state to RUN state, the EtherCAT master starts to run automatically.
- ✧ When PLC switches from Run state to STOP state, the EtherCAT master stops to run automatically.
- ✧ The EtherCAT master can be stopped and started by system variables during PLC operation.

Note: The start-stop of the EtherCAT fieldbus is supported, but the start-stop of a single slave is not supported.

System variables	Data type	Function
bStopMaster	BOOL	It stops the operation of EtherCAT master. The EtherCAT master is stopped on the rising edge of variable input, and the variable is automatically reset after the stop is completed.
bStartMaster	BOOL	It starts the operation of EtherCAT master. When the EtherCAT fieldbus fails or is in stop mode, it will automatically reset after the EtherCAT master is restarted on the rising edge of variable input.

- **Automatic Restart**

By default, it supports the function of automatically restarting the slave.

9.2.5 Summary of System Variables

Table 9-2 EtherCAT Communication Related System Variables

System variables	Data type	Function
MasterRunState	BOOL	Operation status of EtherCAT master When the EtherCAT master receives the run instruction and all

System variables	Data type	Function
		slaves start up, it becomes TRUE. **Note:** This variable is still TRUE if some of the slaves are offline while EtherCAT is running.
LinkState	BOOL	Master link state As long as there is a slave which has normal physical connection with the master, the variable becomes ON, otherwise it is OFF.
HeartBeat	BOOL	EtherCAT real-time task heartbeat Every EtherCAT real-time task cycle is flipped once.
BolckHeartBeat	BOOL	EtherCAT non-real-time task heartbeat Every EtherCAT non-real-time task cycle is flipped once.
MaxCycleTime	DINT	Maximum cycle period of EtherCAT task
MinCycleTime	DINT	Minimum cycle period of EtherCAT task
CycleTime	DINT	Cycle period of the last cycle of EtherCAT task
MaxExeTime	DINT	Maximum EtherCAT task execution time
MinExeTime	DINT	Minimum EtherCAT task execution time
ExeTime	DINT	Execution time of the last cycle of EtherCAT task
TxFrames	DINT	Total frames sent
RxFrames	DINT	Total frames received
TxFramesRates	DINT	Frame rate at which the data is transmitted (frames/second)
RxFramesRates	DINT	Frame rate at which the data is received (frames/second)
TxBytesRate	DINT	Frame rate at which the data is transmitted (bytes/second)
RxBytesRate	DINT	Frame rate at which the data is received (bytes/second)
LossFrames	DINT	EtherCAT lost data frames
ResetTime	BOOL	Reset execution time and cycle time
StartMaster	BOOL	It starts the operation of EtherCAT master When the EtherCAT fieldbus fails or is in stop mode, it will automatically reset after the EtherCAT master is restarted on the rising edge of variable input.
StopMaster	BOOL	It stops the operation of EtherCAT master. The EtherCAT master is stopped on the rising edge of variable input, and the variable is automatically reset after the stop is completed.
DisableMaster	BOOL	Disable Master Enable
ClearFrameCounter	BOOL	Reset EtherCAT data frame counter register
SlavesState	INT	Online status of all slaves It is 1 when all slaves in the configuration are in the running state, and 0 when only one slave is in the non-running state
FirstErrorSlave	INT	This variable is used to display the configuration location of the first offline slave when there is a slave failure in the configuration (the state machine switches to a non-OP state or goes offline)
LibVersion	DINT	EtherCAT system library software version
MstVersion	DINT	EtherCAT master software version
DriveVersion	DINT	EtherCAT network adapter driver software version
TxErrorCnt	DINT	EtherCAT data frame transmit error count
RxTimeoutCnt	DINT	EtherCAT data frame receive timeout count
RxCorruptCnt	DINT	EtherCAT receive invalid frame count
RxUnmatchCnt	DINT	EtherCAT receive unmatched frame count
RxPDOLength	DINT	Total length of PDO received in configuration (bytes)
TxPDOLength	DINT	Total length of PDO transmitted in configuration (bytes)

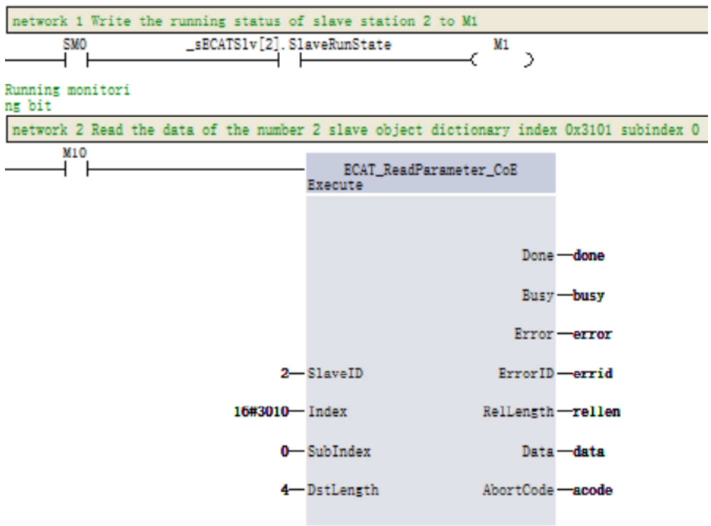
System variables	Data type	Function
ConfigureState	DINT	Configuration state internal use
Delay	DINT	Adjustment value of EtherCAT master synchronous regulator
SlavesLinkState	INT	Physical connection status of all slaves This variable is 1 when all slaves in the configuration are physically connected, otherwise it is 0
DisableState	INT	Master disability state

9.3 Slave Configuration

9.3.1 General Settings

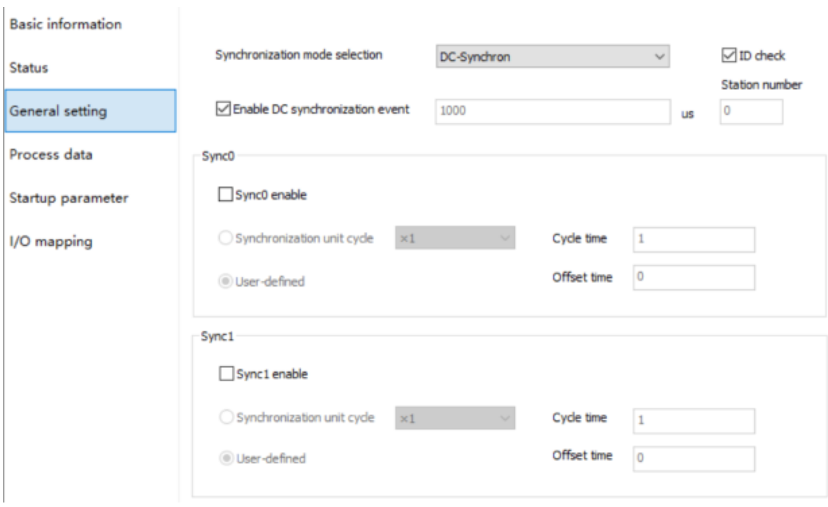
- Configuration address

The configuration address of slave refers to the sequential address in the AutoStation Pro device tree, starting from 0 and increasing sequentially. This address can be used as a subscript to a slave system variable array or as a slave address for reading and writing SDO instructions.



- Distributed clocks

This section is used to set the synchronous operation mode of the slave. The interface is as follows:

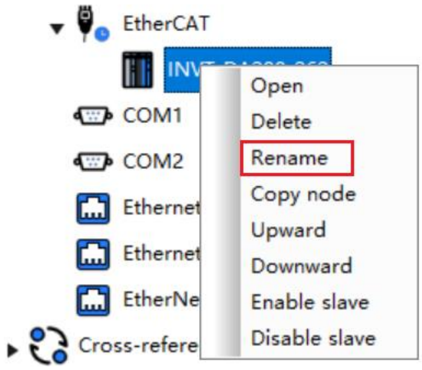


"Synchronization mode selection": Generally speaking, for EtherCAT slaves, FreeRun mode and DC-Synchronization mode which is synchronous with distributed clock can be chosen. The options supported by synchronization mode will vary depending on the slave selected.

Note: Customers are not recommended to modify the default configuration in DC-Synchronon mode unless they have a better understanding of EtherCAT communication principles.

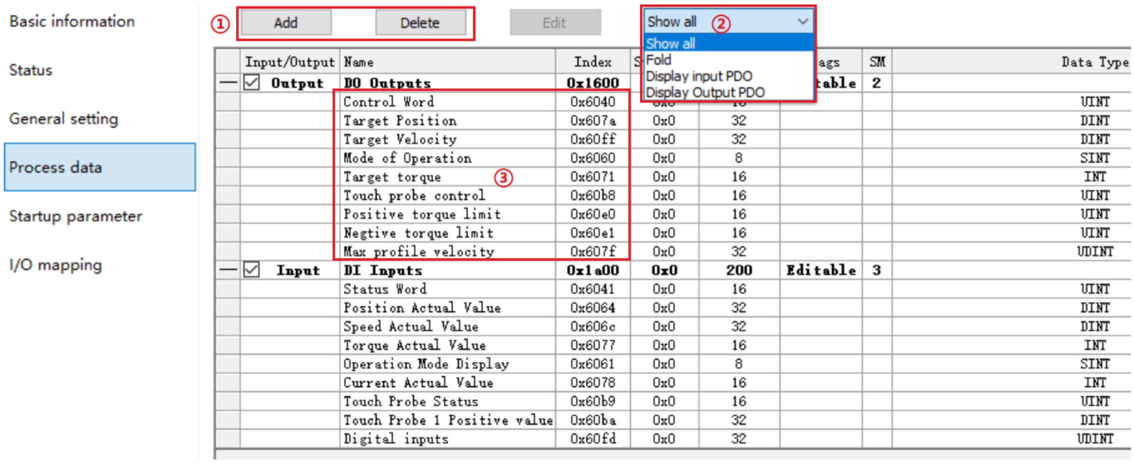
- Site alias setting

Right-click on the slave and click "Rename" to rename the slave.



9.3.2 Process Data

The process data interface is used to edit PDO, as shown in the following figure.



No.	Description
①	PDO edit button
②	PDO display selection area
③	PDO display area

PDO is divided into output group PDO and input group PDO according to data flow direction. The output group PDO represents the process data sent by the EtherCAT master to the EtherCAT slave, such as the control word 0x6040, and the input group PDO represents the process data sent by the EtherCAT slave to the master.

Each slave may have multiple groups of PDOs or a single group of PDOs, as shown in the above figure, the first group of input PDOs and the first group of output PDOs can be edited for addition, editing, deletion, etc.

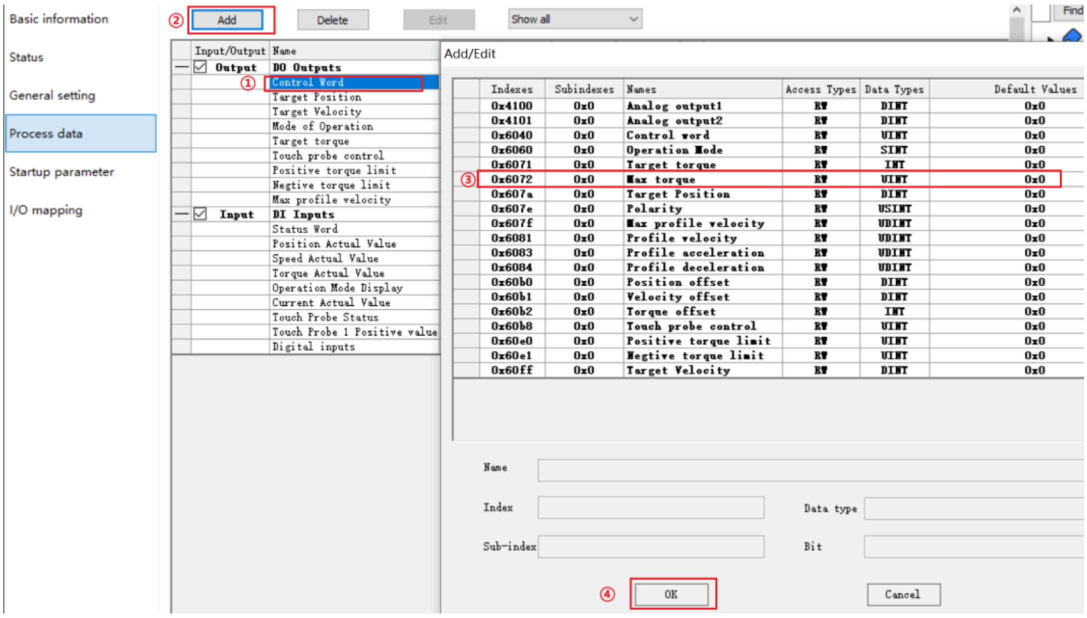
Taking adding PDO as an example, the operating steps are as follows:

Step 1 Select a PDO in the first group

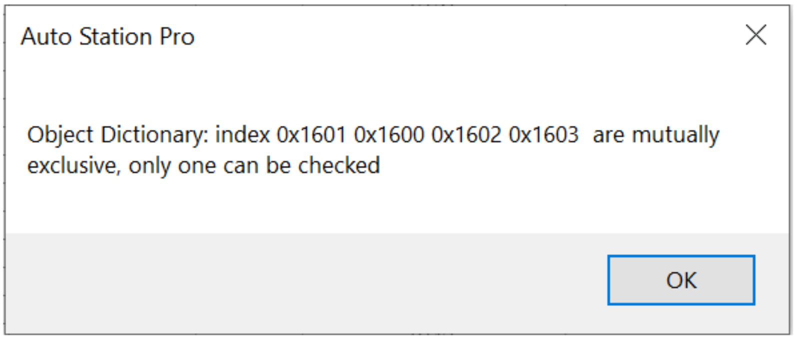
Step 2 Click "Add".

Step 3 Select 6072

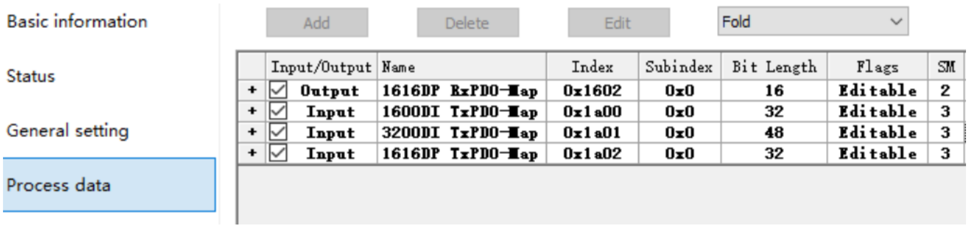
Step 4 Click "OK".



When a slave has multiple groups of PDOs, there may be mutual exclusion between these groups. For example, EC-TX508. Only one group can be selected at a time.



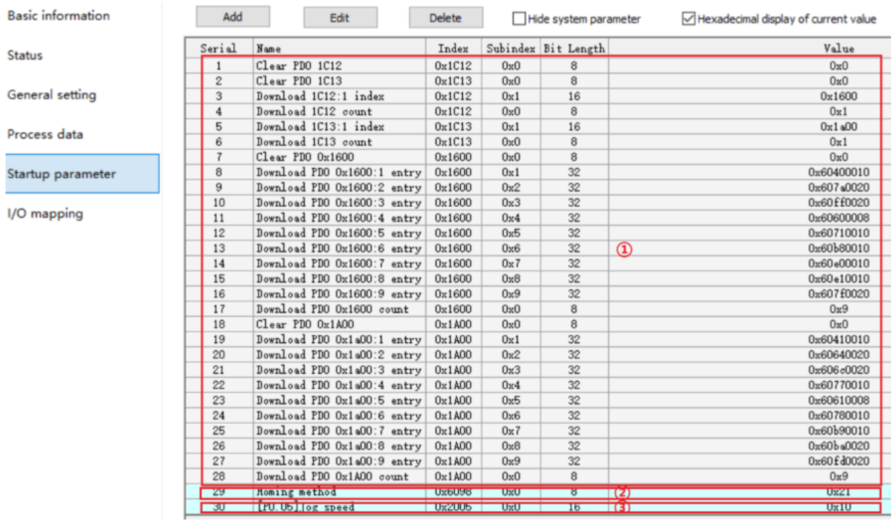
This mutual exclusion relationship is different from slave to slave. For example, FK1100-ECT-Copuler allows multiple groups to be checked.



9.3.3 Startup Parameters

The startup parameters are to write PDO configuration information of the slave, manufacturer setting parameters and parameters specified by some protocols (such as 402 protocol) to the slave by writing SDO when the slave is in PreOP state.

Take DA200 as an example:



No.	Description
①	PDO configuration parameters
②	402 protocol parameters
③	Manufacturer parameters

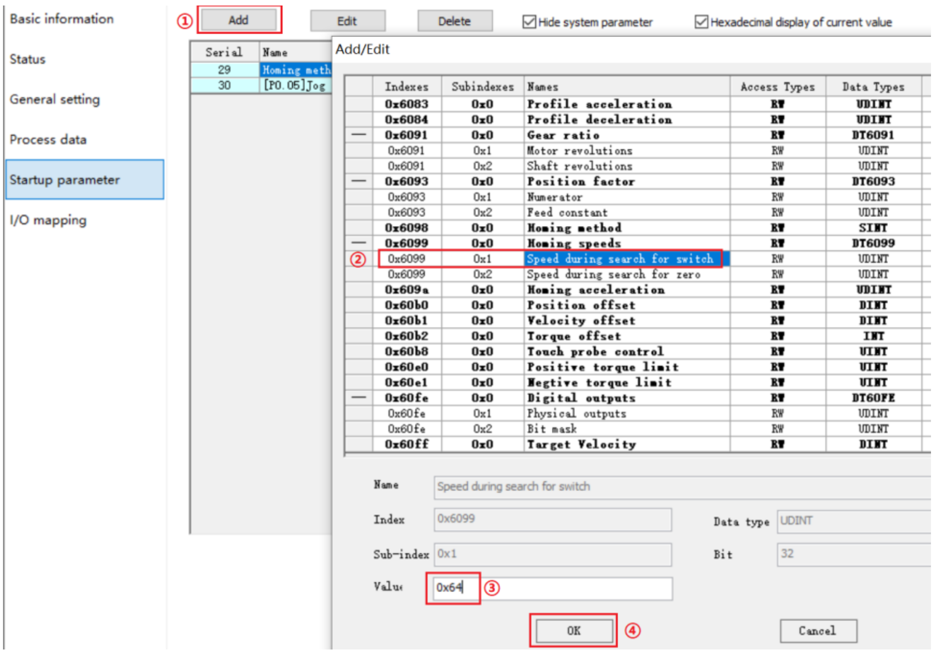
In this interface, users can add startup parameters according to their needs, such as adding the object dictionary 0x6099: 0x01 and modifying the value to 100. The steps are as follows:

Step 1 Click "Add".

Step 2 Select 6099.

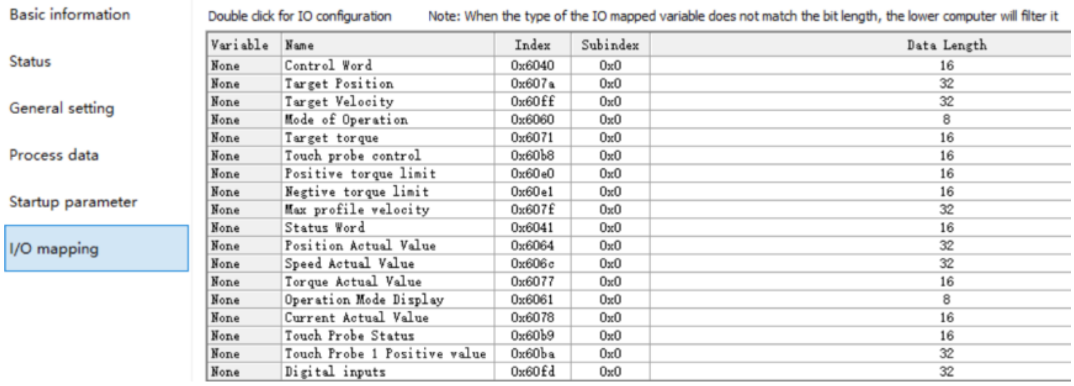
Step 3 Modify the value to 100.

Step 4 Click "OK".



9.3.4 I/O Function Mapping

Only by linking PDO data to PLC internal variables can EtherCAT slave blocks be controlled by operating variables. The I/O function mapping interface is as shown in the following figure.



Variable	Name	Index	Subindex	Data Length
None	Control Word	0x6040	0x0	16
None	Target Position	0x607a	0x0	32
None	Target Velocity	0x60ff	0x0	32
None	Mode of Operation	0x6060	0x0	8
None	Target torque	0x6071	0x0	16
None	Touch probe control	0x60b8	0x0	16
None	Positive torque limit	0x60e0	0x0	16
None	Negtive torque limit	0x60e1	0x0	16
None	Max profile velocity	0x607f	0x0	32
None	Status Word	0x6041	0x0	16
None	Position Actual Value	0x6064	0x0	32
None	Speed Actual Value	0x606c	0x0	32
None	Torque Actual Value	0x6077	0x0	16
None	Operation Mode Display	0x6061	0x0	8
None	Current Actual Value	0x6078	0x0	16
None	Touch Probe Status	0x60b9	0x0	16
None	Touch Probe 1 Positive value	0x60ba	0x0	32
None	Digital inputs	0x60fd	0x0	32

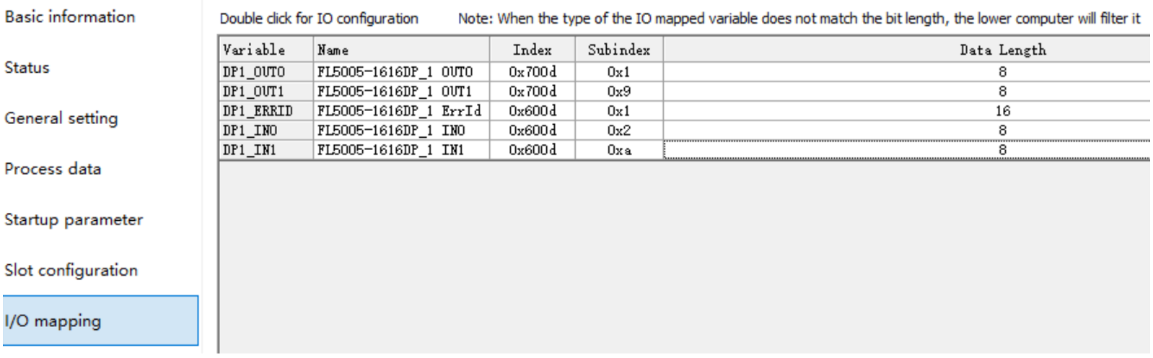
Note:

- This automatically generated set of variables will change with the module location and the addition and deletion of PDOs.
- If a slave is associated by default with a motion control axis, such as DA200, these variables can only be controlled by axis instructions.

9.3.4.1 Associated Variables

To modify the associated variables, follow these steps (take the FK1100_ECT_Coupler slave as an example).

Step 1 Open the Variable Table and add new variables.



Variable	Name	Index	Subindex	Data Length
DP1_OUT0	FL5005-1616DP_1 OUT0	0x700d	0x1	8
DP1_OUT1	FL5005-1616DP_1 OUT1	0x700d	0x9	8
DP1_ERRID	FL5005-1616DP_1 ErrId	0x600d	0x1	16
DP1_IN0	FL5005-1616DP_1 IN0	0x600d	0x2	8
DP1_IN1	FL5005-1616DP_1 IN1	0x600d	0xa	8

Step 2 Associate variables in the I/O function mapping interface.

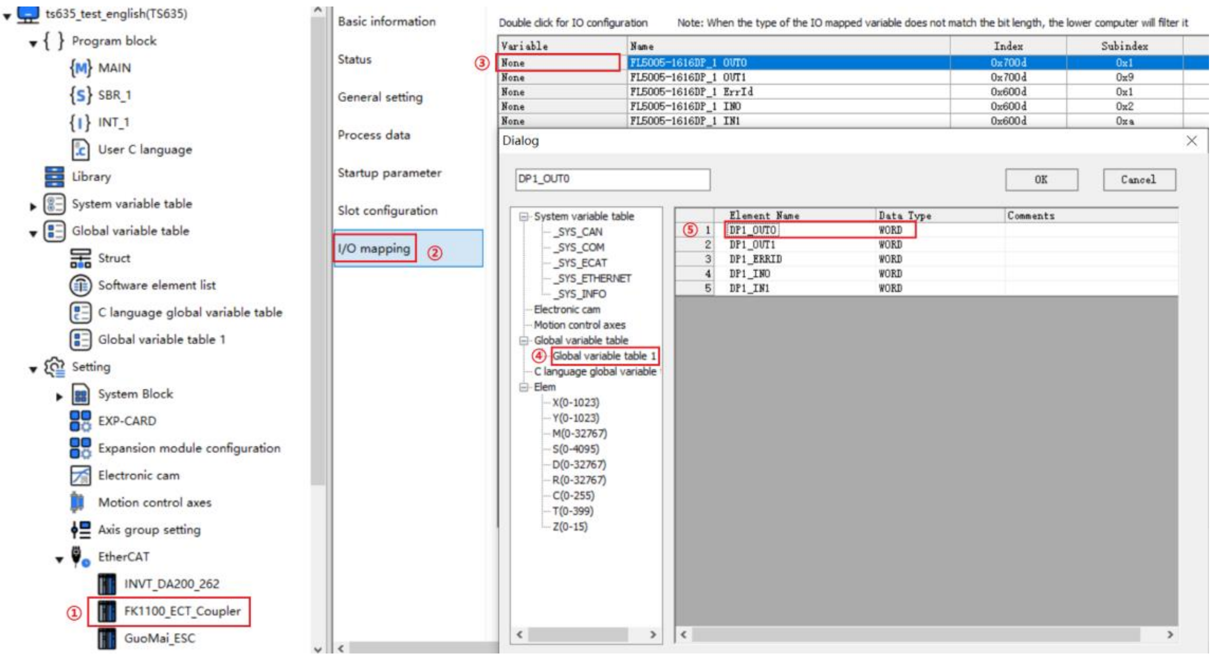
1) Open the slave device.

2) Select I/O mapping option.

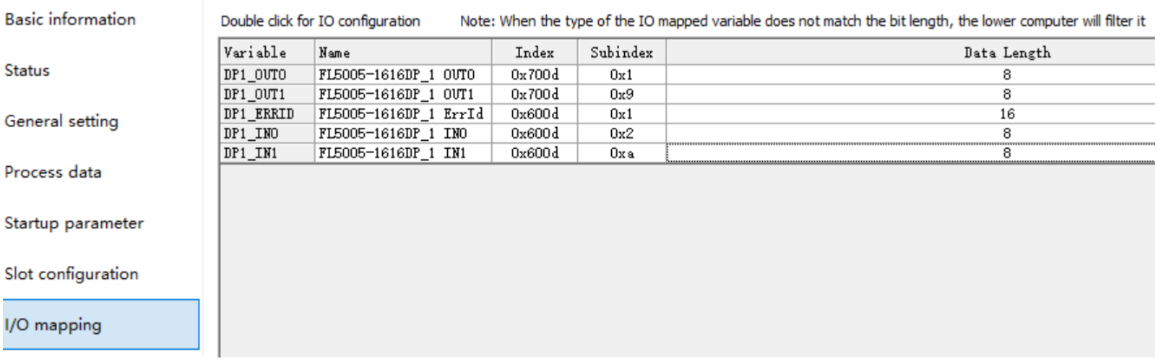
3) Select the global variable table.

4) Select the specified variable.

5) Click "OK".

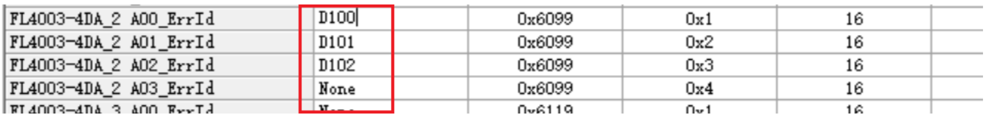


Step 3 After the association is completed, the interface is as shown in the following figure.

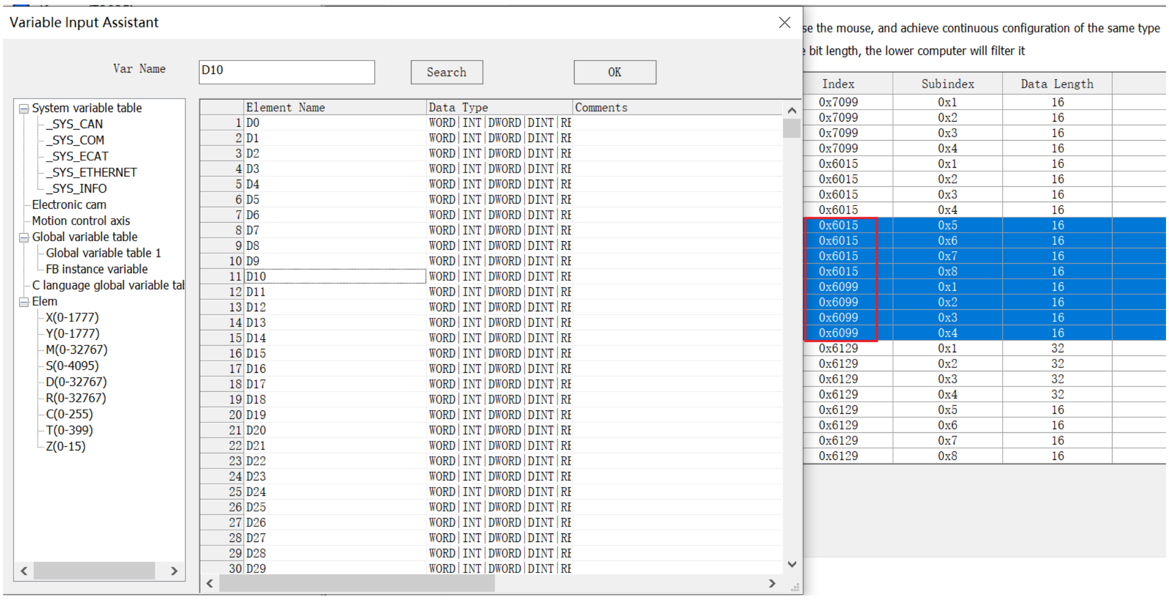


9.3.4.2 Associated Elements

Method 1: Directly enter the element in the "Variables" area.



Method 2: Add elements in batches by left-clicking on the components that need to be mapped to pop up the interface (only supports data that are of the same length and adjacent).



Method 3: Directly double-click on the variable area, and add on the small interface.

9.3.4.3 Mapping Rule

There are only six basic data types of element variables: BOOL, INT, DINT, WORD, DWORD and REAL, while there are many data types of PDO variables in EtherCAT slave. Therefore, the mapping rules in the I/O mapping interface are as follows:

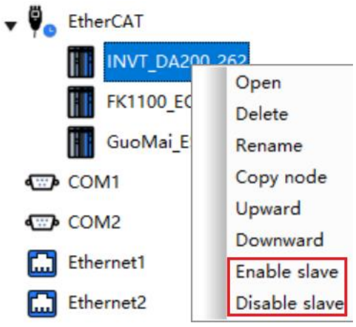
EtherCAT Slave Type	Bit Length	Mapping rule
BOOL	1	BOOL
BYTE	8	● INT: The lower eight digits are valid and the higher eight digits are reserved ● BOOL [8]: 8-bit BOOL array
SINT	8	● INT: The lower eight digits are valid and the higher eight digits are reserved ● BOOL [8]: 8-bit BOOL array
USINT	8	● INT: The lower eight digits are valid and the higher eight digits are reserved ● BOOL [8]: 8-bit BOOL array
BITARR8	8	● INT: The lower eight digits are valid and the higher eight digits are reserved ● BOOL [8]: 8-bit BOOL array
BIT8	8	● INT: The lower eight digits are valid and the higher eight digits are reserved ● BOOL [8]: 8-bit BOOL array
INT	16	● INTBOOL [16]: 16-bit BOOL array
UINT	16	● INTBOOL [16]: 16-bit BOOL array
WORD	16	● INT: The lower eight digits are valid and the higher eight digits are reserved ● BOOL [16]: 16-bit BOOL array
BITARR16	16	● INT: The lower eight digits are valid and the higher eight digits are reserved ● BOOL [16]: 16-bit BOOL array
DINT	32	● DINTBOOL [32]: 32-bit BOOL array
UDINT	32	● DINTBOOL [32]: 32-bit BOOL array

EtherCAT Slave Type	Bit Length	Mapping rule
DWORD	32	● DINT ● BOOL [32]: 32-bit BOOL array
BITARR32	32	● DINT ● BOOL [32]: 32-bit BOOL array
REAL	32	● REAL

9.3.5 Start-stop/Disable/Enable

When there are more slaves in the configuration than the actually connected devices, matching can be achieved by disabling the slaves that do not exist in the configuration.

How to disable (or enable): Select the target slave, click the right mouse button, and select Disable (Enable) function.



9.3.6 Disable Slave by Instruction

- Related system variables
  - ✧ EtherCAT slave system variable that sets whether the specified slave is enabled

System variables	Data type	Function description
DisableEnable	BOOL	Disable Enable OFF: Disable invalid ON: Disable Enable
DisableState	INT	Configuration state 0: Reserved 1: Enable 2: Disable

  - ✧ EtherCAT master system variable for disabling the entire EtherCAT fieldbus

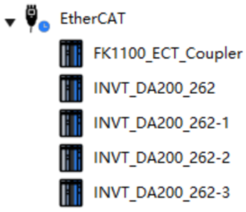
System variables	Data type	Function description
DisableMaster	BOOL	Disable Master Enable OFF: Disable invalid ON: Disable Enable
DisableState	INT	Master disability state 0: Reserved 1: Enable 2: Disable

- The method is as follows:
  - ✧ When PLC is powered on and started, it initializes the DisableEnable variable according to the configuration configured in the background, then updates the configuration list according to the DisableEnable variable disable enable state of the slave, starts the master, and writes the actual disable state of the slave into the variable DisableState.

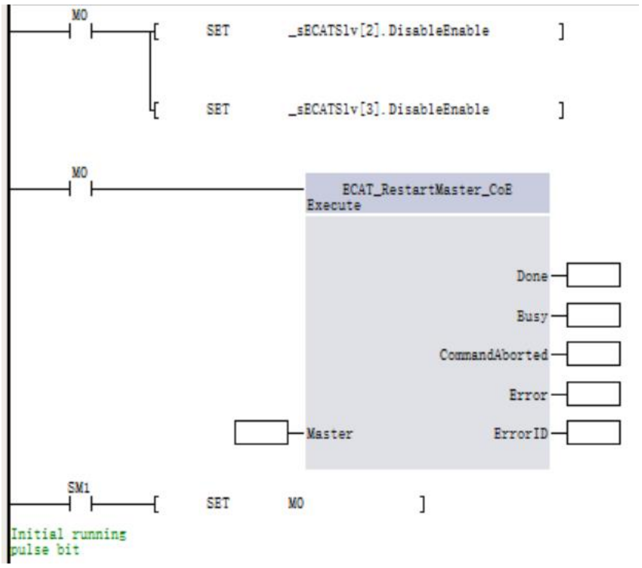
- ✧ Wait for PLC parsing program configuration.
- ✧ Reset the value of DisableEnable variable through PLC program.
- ✧ Restart the EtherCAT master through ECAT_RestartMaster_CoE instruction after setup.
- ✧ When the master is restarted, update the configuration list with the DisableEnable variable, and write the disable enable state of the slave to DisableState.

- Example

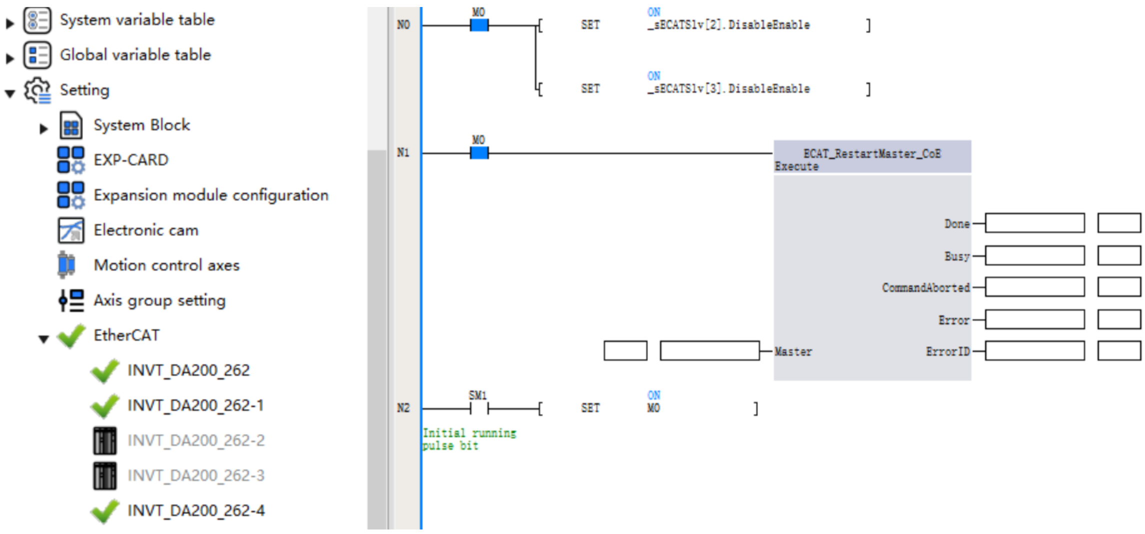
1. Set up the configuration by upper computer software, start EtherCAT communication, and configure one FK1100 slave and four DA200 slaves.



2. After setting the slave system variable DisableEnable to disable the two slaves of DA200, restart the master by instruction ECAT_RestartMaster_CoE.



3. Download the project to PLC, and the two DA200 slaves are disabled after starting.



Note: The ECAT_RestartMaster_Coe instruction is a rising edge instruction.To take effect, the state of the EtherCAT slave system variables must be changed before the first cycle when this instruction is enabled. If you modify whether to enable the system register of the slave after the instruction is enabled, the instruction must be disconnected and retriggered.

9.3.7 System Variables

The system variables associated with the EtherCAT slaves are shown in the following table:

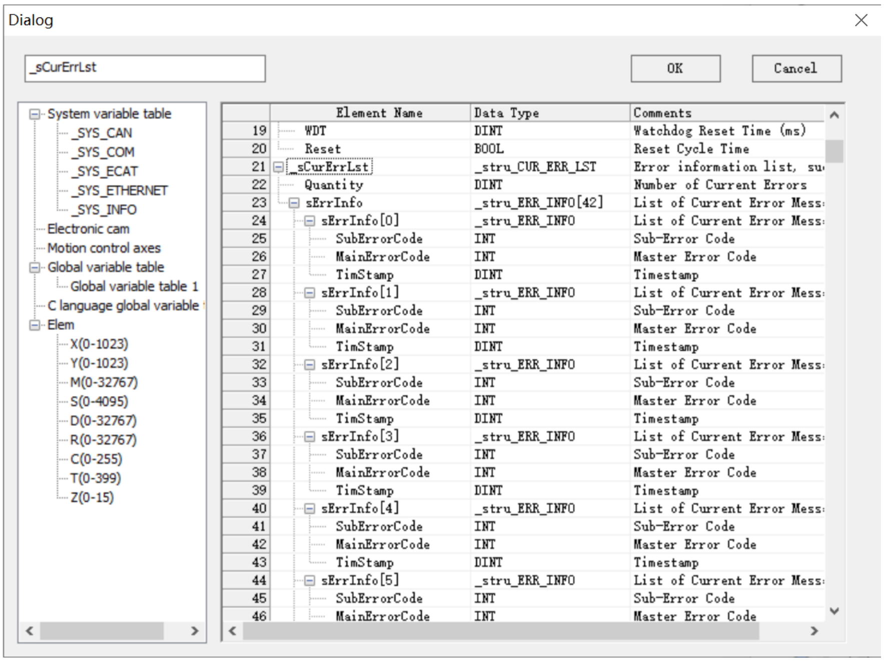
System variables	Data type	Function description
DisableEnable	BOOL	Disable Enable OFF: Disable invalid ON: Disable Enable
DisableState	INT	Configuration state 0: Reserved 1: Enable 2: Disable
bSlaveRunState	BOOL	Operation status of slave TRUE when slave is in OP mode, FALSE otherwise
SetAliasState	BOOL	Set node alias status (background specific) TRUE: Set busy FALSE: Idle or the setting is completed
SetAliasError	BOOL	Failed to set node alias (background specific) TRUE: Failed to set node alias FALSE: No fault
SetAlias	BOOL	Set the node alias (background specific) On the rising edge of the variable, write the value of wTarAlias to the slave
ALState	INT	State of slave EtherCAT state machine 1: INIT 2: PreOP 4: SafeOP 8：OP
AlCode	INT	Slave state machine transition fault code, see the manual of each slave for details
ActAlias	INT	The actual alias of the slave, which is initialized once when powered on, and its modification will not take effect
TarAlias	INT	Node alias to write (background specific)
StationAddress	INT	Sequential address of slave, which is initialized once when powered on, and its modification does not take effect

9.4 Fault and Diagnosis

9.4.1 Fault Acquisition

EtherCAT instruction failure can obtain fault codes through ErrorID of the instruction.

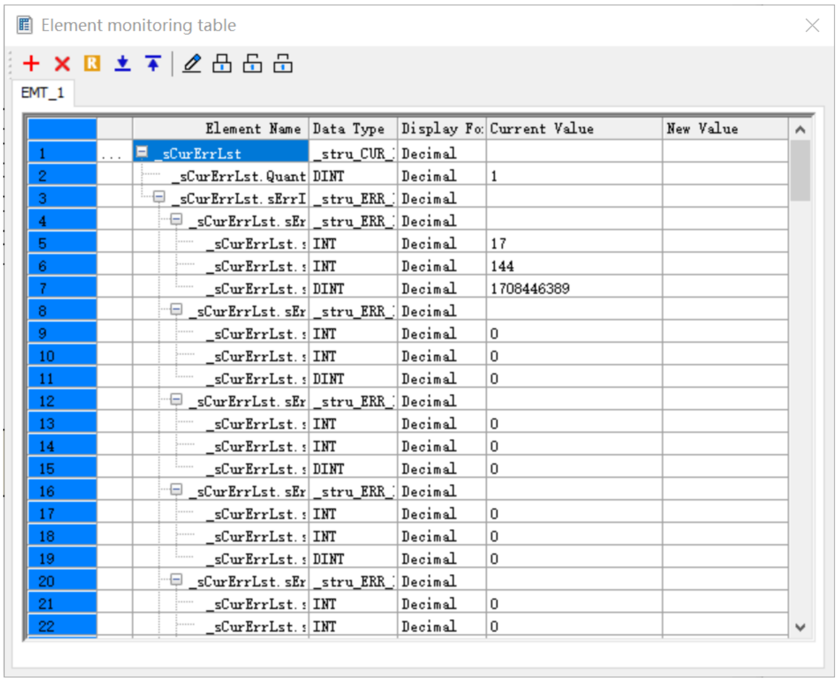
The fault of EtherCAT fieldbus can be obtained through the system variable _sCurErrLst.



9.4.2 Fault Code

For details about fault code list, see the relevant contents of main error code 0x0090 in section 19.3.2 Fault Code List.

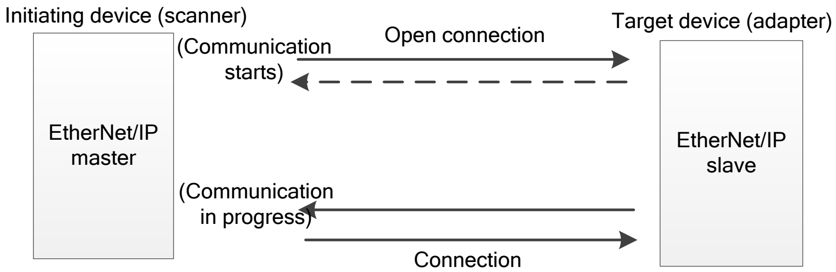
When the EtherCAT fieldbus fails, the main module's Err indicator flashes, and check the system variable _sCurErrLst.



10 EtherNet/IP Communication

10.1 Overview

The end opening the connection is the EtherNet/IP master, and the other end is the EtherNet/IP slave, as shown in the following figure.



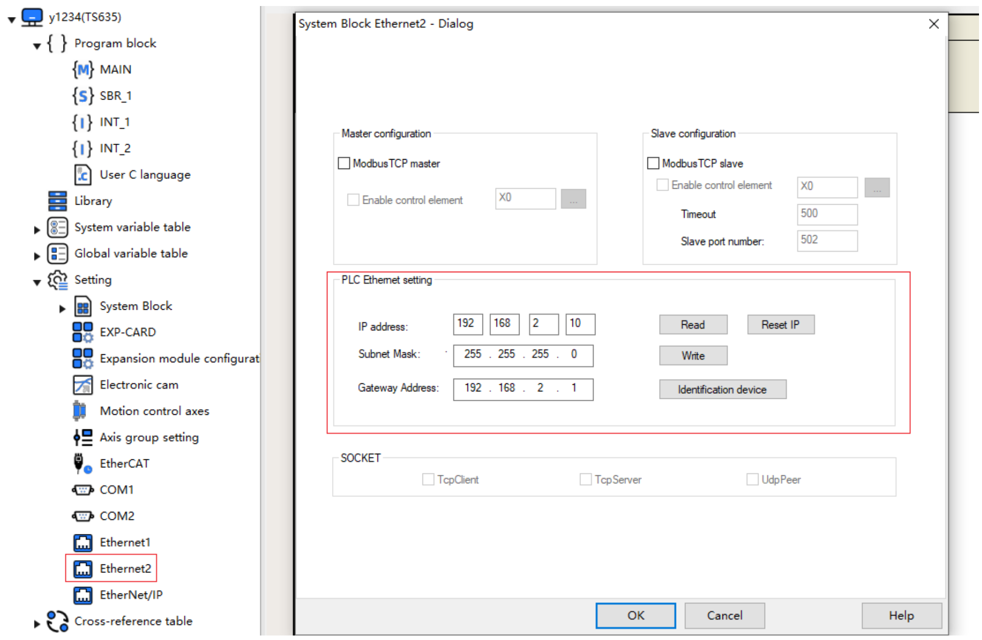
A small PLC with background Auto Station pro1.1.2 or above, firmware version 1.28 or above supports EtherNet/IP slave function, and its specifications are as follows:

1. TS635 small PLC supports one-way EtherNet/IP slave function, with fixed physical interface network port 2.
2. Minimum cyclic communication period (RPI) 5ms.
3. The maximum data volume of TS365 single connection communication is 500 bytes.
4. It supports creating up to 32 slave connections.
5. Please refer to the official standard documents of EIP-CIP-V1-1.0 and EIP-CIP-V2-1.0 for the details of protocol.

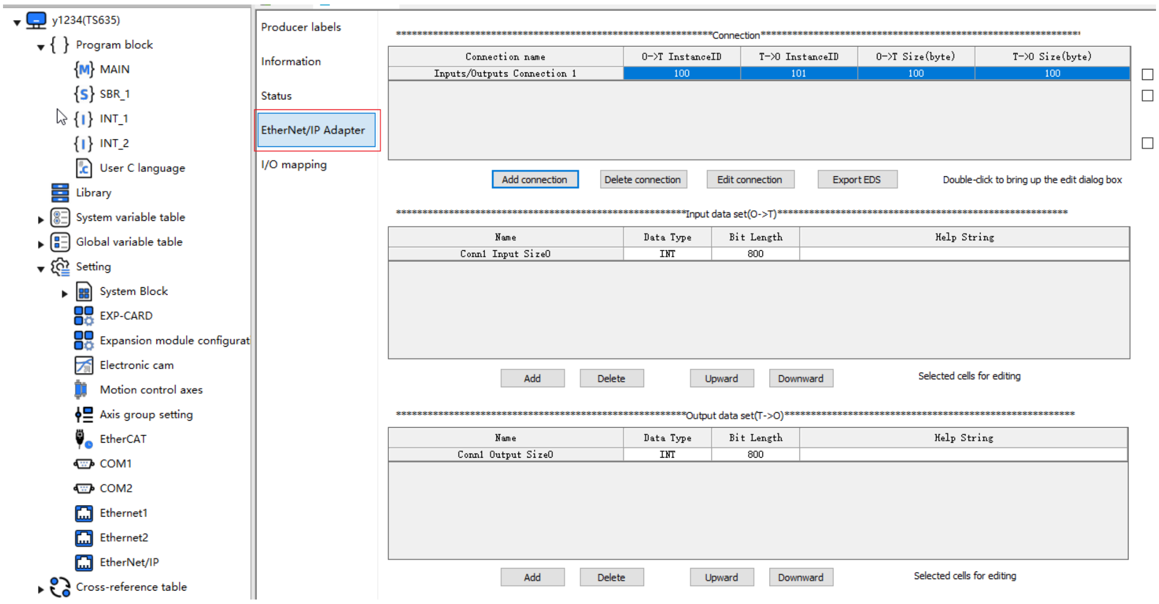
10.2 Class1 Communication

10.2.1 Slave Configuration

Step 1 Click to set Ethernet 2 and modify the network configuration.



Step 2 Configure connection parameters, click "Setting" > "Ethernet/IP" option, choose "Ethernet/IP Adapter" configuration option to enter the configuration interface, and select to add connections as needed, and up to 16 Connection can be added, 16 inputs and 16 outputs, totaling 32 connections.



- Connection parameter list definition

Parameter	Definition
Connection name	The name of the connection established
O-> T Instance ID	Actual ID from origin to target, input (slave)
T-> O Instance ID	Actual ID from target to origin, output (slave)
O-> T size (byte)	Maximum number of bytes transferred from origin to target (maximum 500 bytes)
T > O size (byte)	Maximum number of bytes transferred from target to origin (maximum 500 bytes)

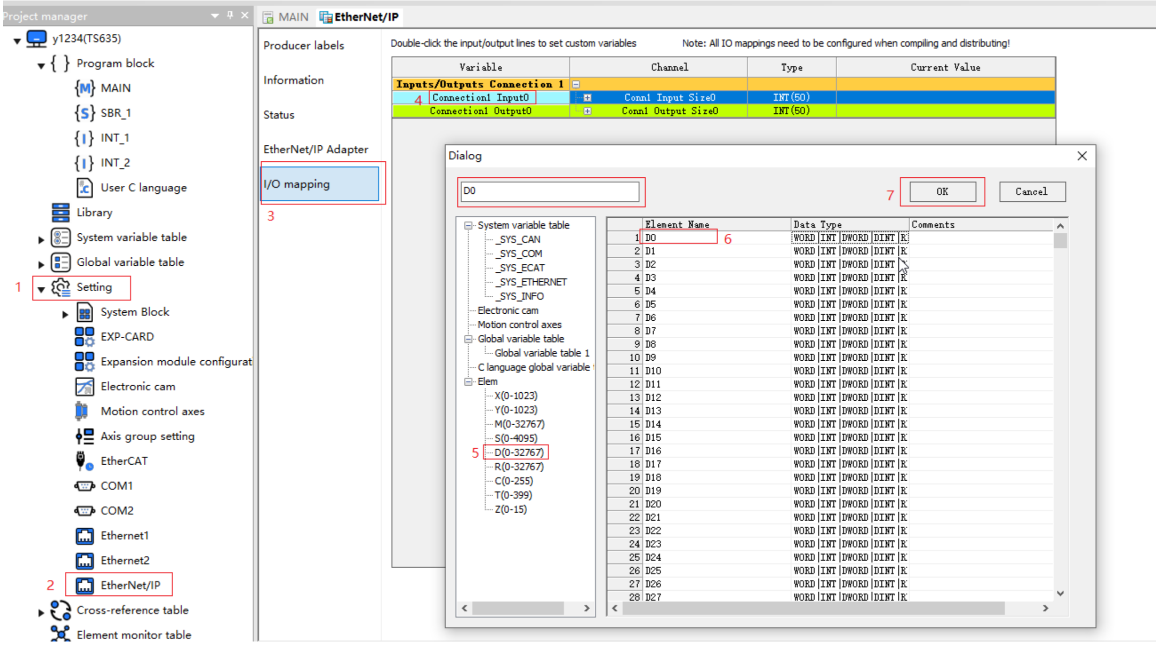
- Slave input data set (O-> T) parameter definition table

Parameter	Definition
Connection name	Dataset name
Data type	Type of data transmission
Bit Length	Total number of bits of data transferred (bytes*8)

- Slave input data set (T-> 0) parameter definition table

Parameter	Definition
Connection name	Dataset name
Data type	Type of data transmission
Bit Length	Total number of bits of data transferred (bytes*8)

Step 3 Configure IO mapping of dataset, click "Setting" > "Ethernet/IP" option, choose IO mapping configuration option, enter IO configuration interface, select data set to be configured under the corresponding connection, double-click the dataset, select the corresponding element (now D element and custom parameters of global variable table are supported), double-click to select the element, and after the element option is displayed at the upper left, click "OK" to complete IO mapping of dataset.



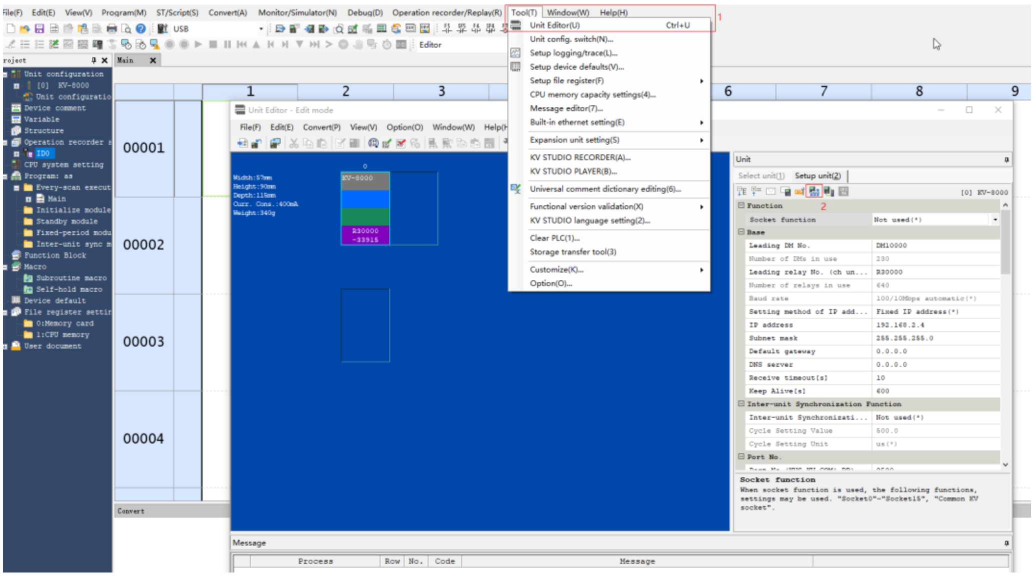
10.2.2 Use Example of Slave

10.2.2.1 Use of Connection

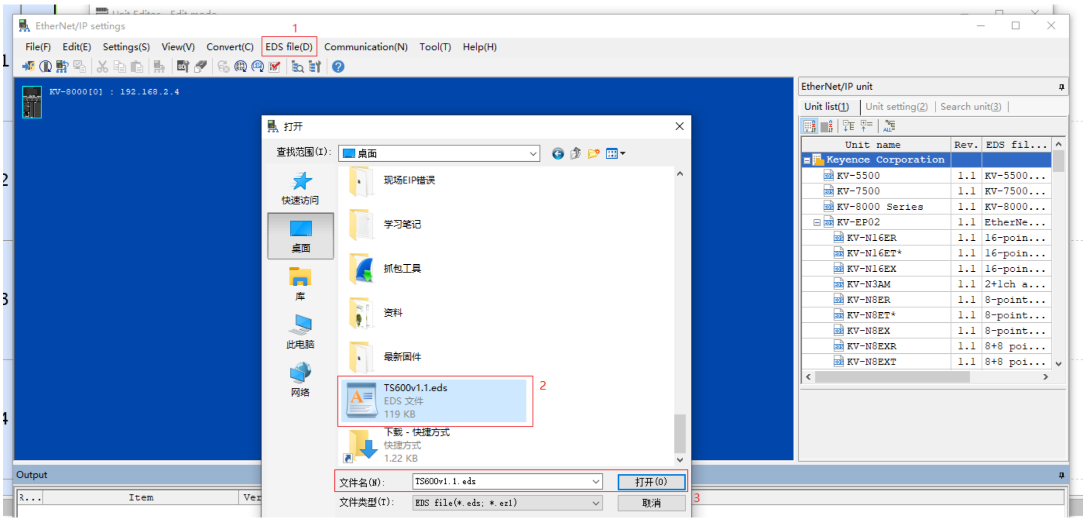
Realization function: The master uses KV-8000, and the slave uses TS635, to complete O > T transmission of 100 bytes and T > O transmission of 100 bytes.

- Operating steps of the master:

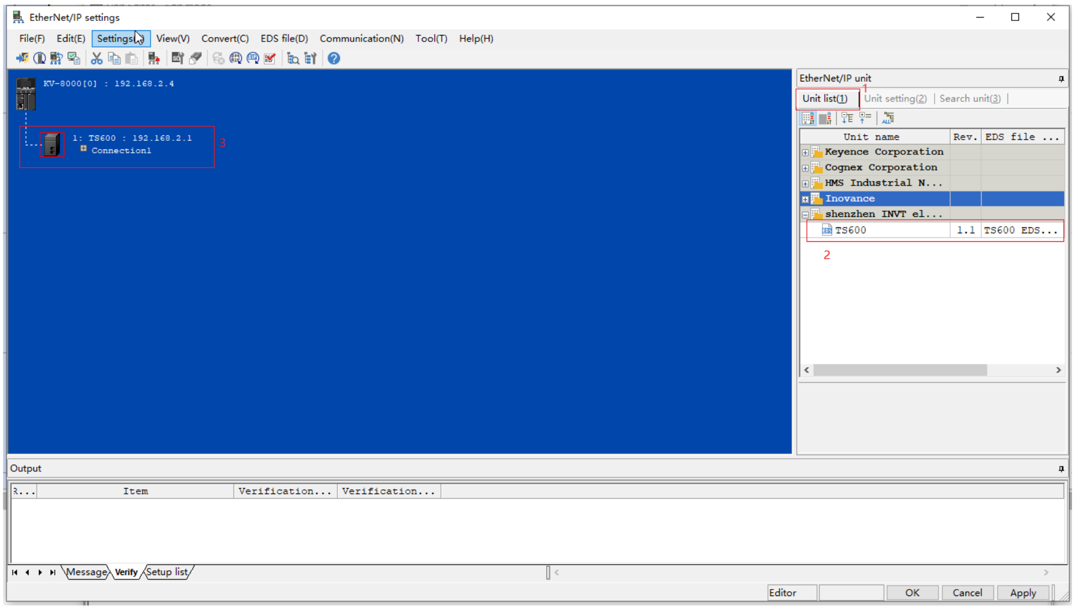
Step 1 Change the master address to 192.168.2.4, then click the cell editor in the tool and choose "EtherNet/IP setting".



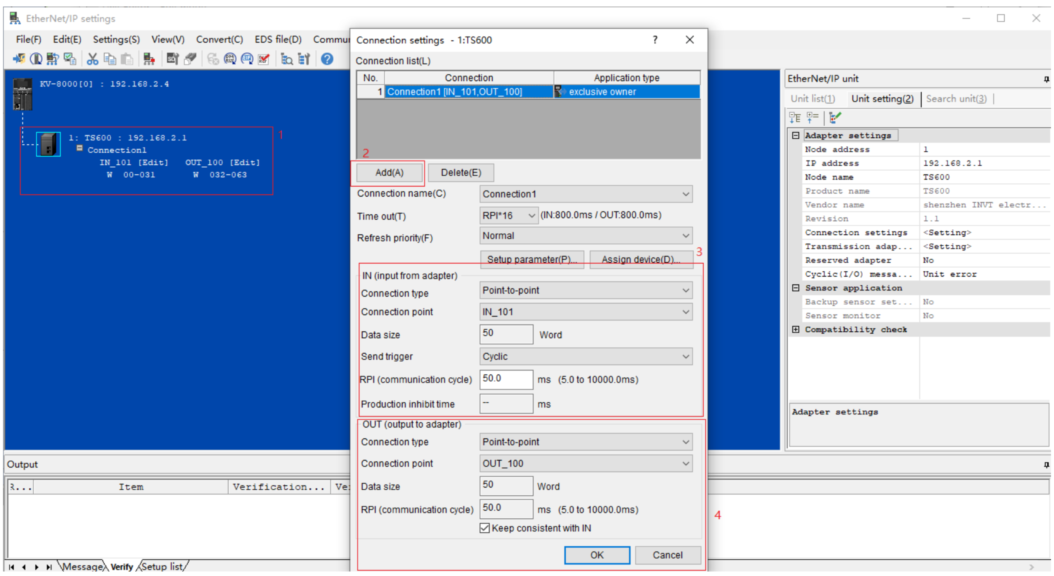
Step 2 Select the eds file in "EtherNet/IP setting", select "Login", and open the TS600.eds file.



Step 3 In "EtherNet/IP setting", click on the device list, select TS600, and double-click it to generate a TS600 slave.

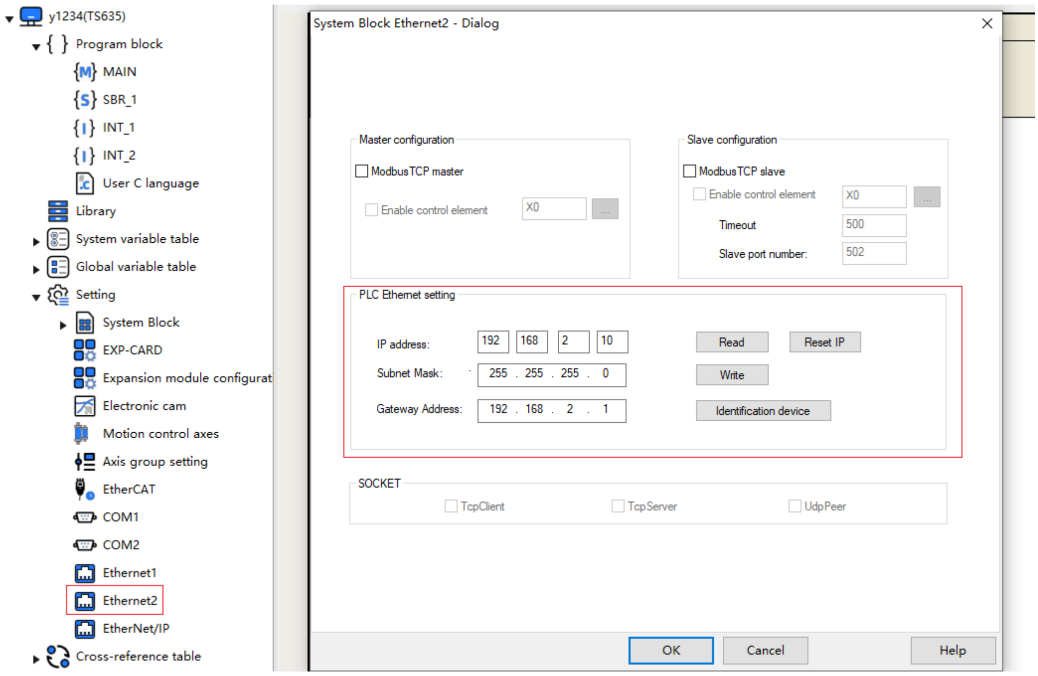


Step 4 Configure the connect parameters W0-W31 as inputs and W32-W63 as outputs in "EtherNet/IP setting" with reference to Keyence instructions.

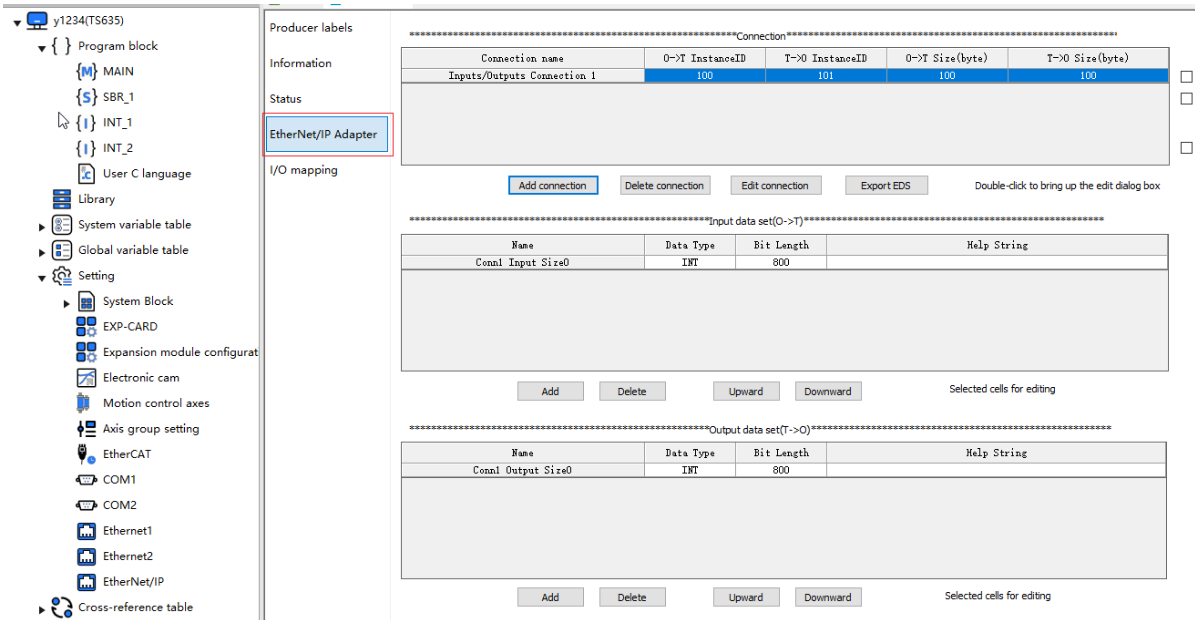


- Slave operating steps:

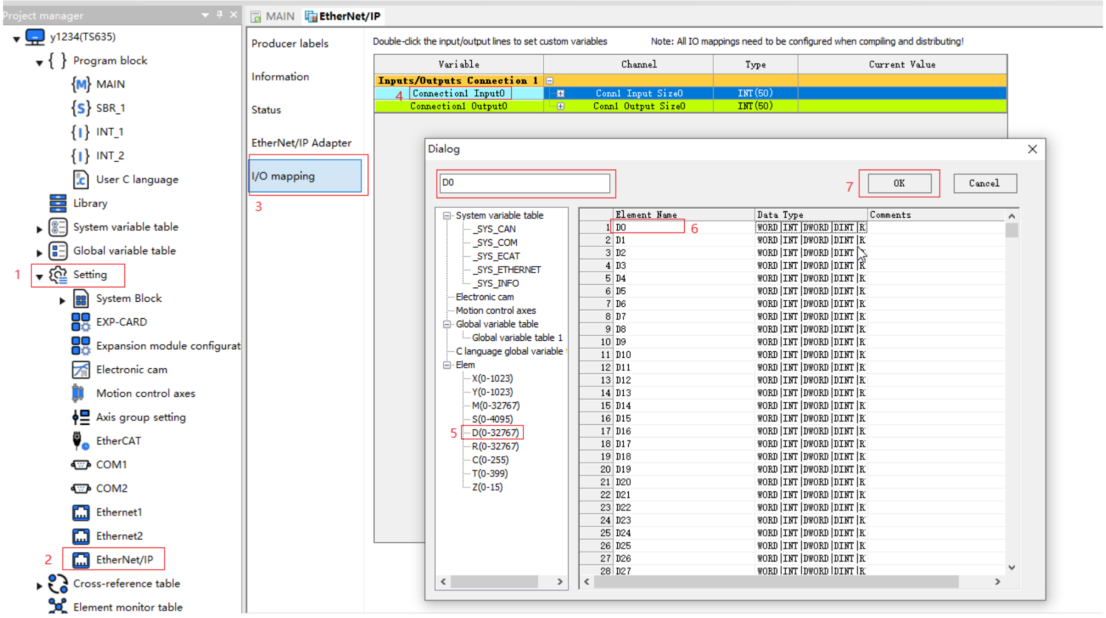
Step 1 Configure the Ethernet parameters of Network Port 2 with IP of 192.168.2.10.



Step 2 In "EtherNet/IP setting", configure connection1, 0 > T size to 100 bytes, and T > 0 size to 100 bytes. Input dataset data type as int type, input bits 800, with ID of 0 > T being automatically generated as 100, and ID of T > 0 being automatically generated as 101.

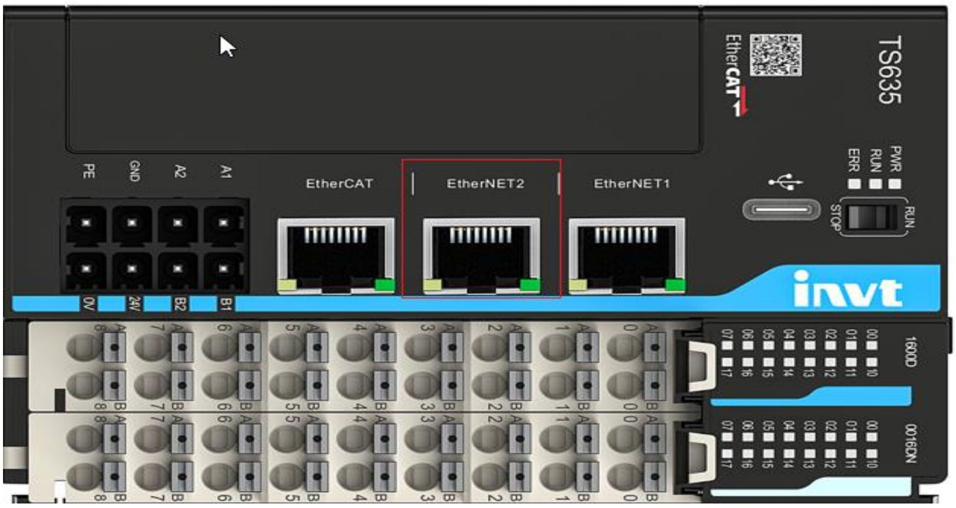


Step 3 Configure the IO dataset mapping in "EtherNet/IP setting", and configure D0-D49 as inputs and D50-D99 as outputs.



- Test steps:

Step 1 Connect Keyence network to TS635 network port 2.



Step 2 Modify the value of W32-W63 of Keyence to change with the value of D0-D49 in TS635.

Step 3 Modify the value of D50-D99 in TS635 to change with the value of W0-W31 of Keyence.

11 Motion Control

11.1 Brief Introduction of Motion Control Axis

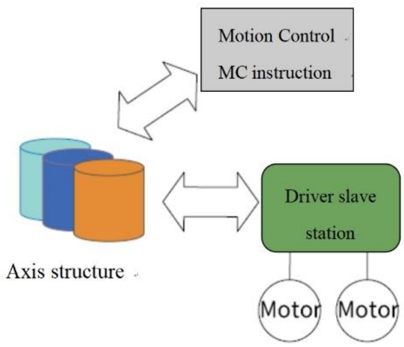
11.1.1 Overview

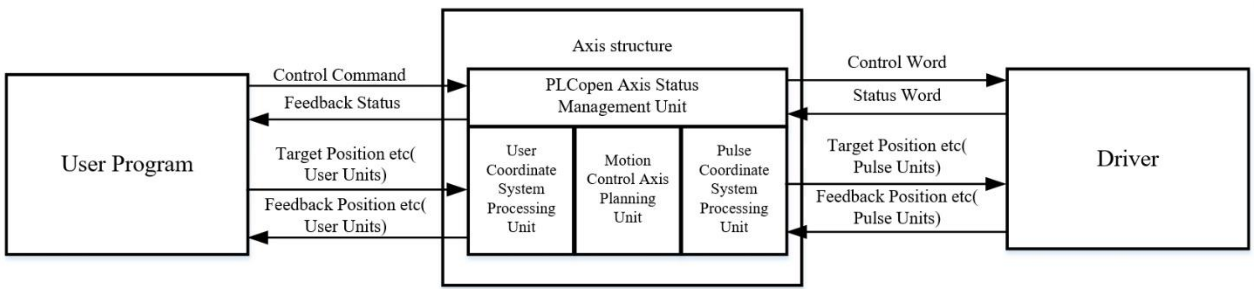
11.1.1.1 Basic Composition and Control Logic

In a motion control system, the object of motion control is called axis. An axis is the bridge between driver and PLC instructions. The motion control axis is used to control the EtherCAT fieldbus driver conforming to 402 protocol, and can also control the local high-speed pulse output and high-speed pulse input.

In PLC, the basic structure and processing logic of axis are as follows:

Main main program, subroutine and interrupt subroutine are for users to write programs, but motion control instructions can only be called in a Main main program or subroutine rather in an interrupt subroutine.



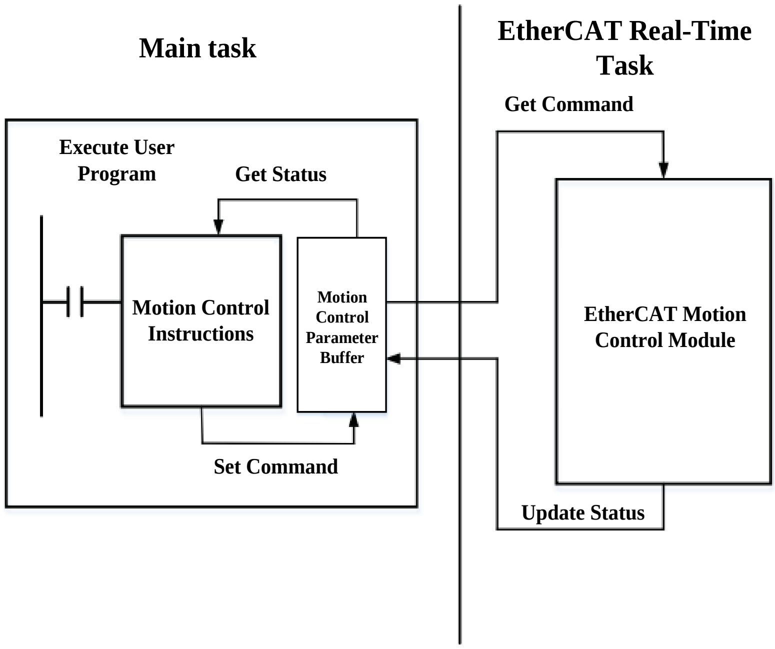


11.1.1.2 Scheduling Mechanism of Motion Control Instruction

EtherCAT tasks are hidden tasks and are not open to users, so programming in EtherCAT tasks is not supported.

In Main main program, PLC scans all motion control instructions written in the program in turn, and stores the final results in the motion control parameter buffer according to the interrupt rules of the program. The motion control instruction is updated when EtherCAT task is executed. After the execution is completed, the execution result is put into the buffer, and the motion control instruction in the Main main program updates the instruction state according to the execution result.

For example, there are two MC_MoveAbsolute instructions in the program, the target position of the first instruction is 100, and that of the second instruction is 200. The two instructions are triggered at the same time. When scanning the program, PLC first scans the first absolute positioning instruction to obtain the target position 100, and then scans the second instruction. At this time, the target position is updated to 200. At the end of Main main task, the target position of 200 is finally written into the motion control buffer, which is executed according to the second absolute positioning parameter, and the first instruction is interrupted. After the EtherCAT task obtains the target position of 200, it starts to execute the absolute positioning algorithm, and sets the completion flag after the positioning is completed. After the second absolute positioning instruction in Main main program obtains the completion flag, the Done signal is set to be active.



11.1.1.3 Axis Type

Axis type	Content
Fieldbus servo axis	● The axis controlled using EtherCAT slave servo driver. ● When the virtual axis mode is not enabled, this axis is assigned to the actual servo driver for use. ● The fieldbus servo axis supports several basic modes of control, such as jogging, speed and homing.
Local pulse axis	Axis controlled by pulse driver using local high-speed IO control. It is allowed to set four local pulse axes, namely Y0/Y1, Y2/Y3, Y4/Y5 and Y6/Y7. ● Each pulse output channel can be set as pulse + direction, positive and negative pulse or quadrature encoded pulse. ● Each pulse output channel can be provided with two probe terminals at most. ● The local pulse axis supports several basic modes of control, such as point, speed and homing, but does not support torque mode.
Local encoder axis	See section 12 High-speed Counter.

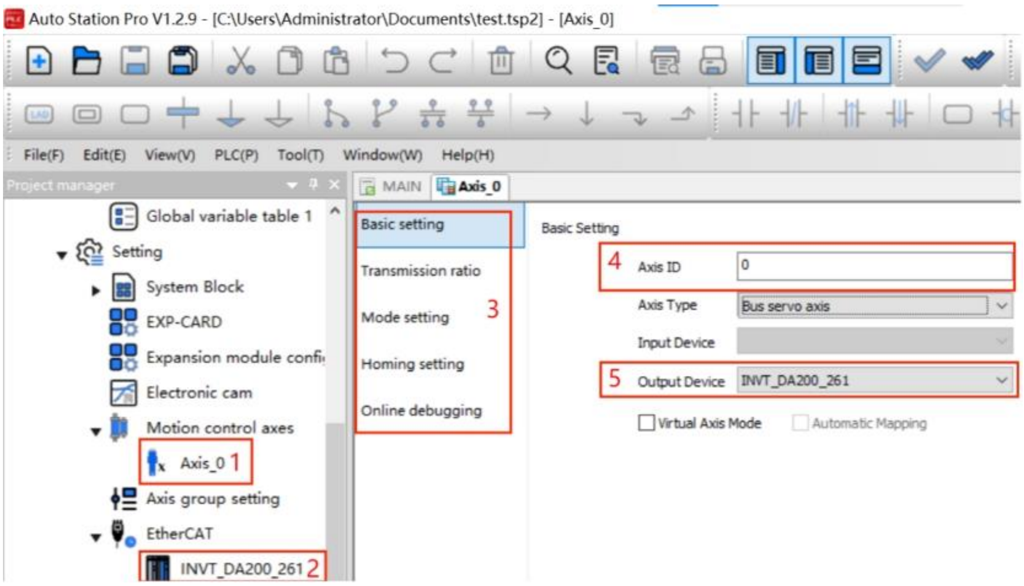
In order to comprehensively describe the axis attributes, monitor the axis state and control the axis movement, the axis is divided into three parts:

Axis Structure	Function
Axis configuration parameters	Used to configure various parameters of axis, such as gear ratio, home type, parameter limit, etc.
Axis system variable	Used to monitor the running status and abnormal information of the axis, such as the current axis position, fault code, etc.
Axis control instruction	In the user program, the axis motion control is executed using the MC motion control governance Axis control instructions are divided into management class (such as

Axis Structure	Function
	MC_Power), motion class (such as MC_Jog) and state class (such as MC_ReadStatus)

11.1.1.4 Configuration Interface

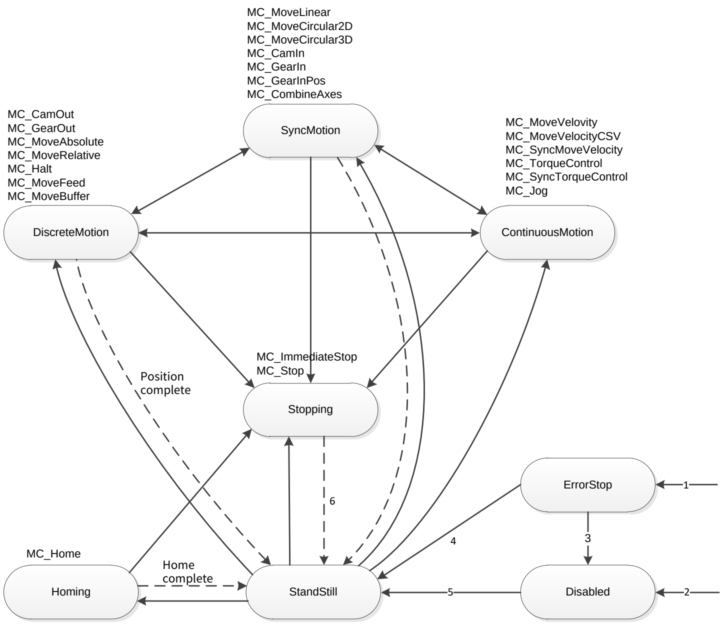
In the project, the configuration interface of axis is as follows:



No.	Description
1	Motion control axis
2	EtherCAT fieldbus driver
3	List of axis configuration and monitoring options
4	Axle number
5	Associated physical driven devices

11.1.2 PLCopen State Machine

Based on PLCopen state machine, the state and motion of the axis are managed, and different functions are completed in different state. The state transition diagram is as follows:



The detailed description is as follows.

State	Function description
Disabled	Not enabled state
ErrorStop	Fault stop state
Standstill	Enabled state
Homing	State of returning to home
Stopping	Stopped state
Discrete Motion	Discrete motion
Continuous Motion	Continuously move state
Synchronized Motion	Synchronously move state

State migration conditions:

Conversion	Conversion Condition
1	When the fault detection logic of the axis detects a fault, it immediately switches to this state.
2	When there is no fault with the axis and MC_Power.Enable is FALSE
3	When MC_Reset is called to reset axis failure and MC_Power.Status is FASLE
4	When MC_Reset is called to reset axis failure and MC_Power.Status is TRUE
5	When MC_Power.Enable is TRUE and MC_Power.Status is TRUE
6	When MC_Stop(MC_ImmediateStop).Done is TRUE and MC_Stop(MC_ImmediateStop).Execute is FALSE

11.1.3 Unit of Axis

Two units are used in the axis structure, namely Unit and pulse unit.

- Unit
  - ✧ The units of measurement such as millimeters, centimeters and angles used on the instruction side are called user units, which are usually expressed by Unit.
  - ✧ User coordinate system can be divided into linear coordinate system and rotating coordinate system according to different working conditions.
  - ✧ A linear coordinate system usually contains a zero point, and the target position increases to indicate forward motion, while the target position decreases to indicate reverse motion. Positive and negative soft limits can be set in the linear coordinate system.
  - ✧ A rotating coordinate system contains a zero point and a rotation period. In a rotation period, the target position increases to indicate clockwise movement and decreases to indicate counterclockwise movement.
- Pulse unit
  - ✧ The unit of measurement used on the driver side to measure the number of pulses, which is usually expressed as pulse.

11.1.4 Axis Configuration Parameters

In order to meet the needs of the motion control axis and the actual working conditions, the following axis configuration parameters are developed:

Classification	Content	Fieldbus servo axis	Local pulse axis
Basic settings	Axle number	√	√
	Axis type	√	√
	Input device	×	×
	Output device	√	√
	Automatic mapping	√	×
	Virtual axis mode	√	√
	PDO	√	×
Unit conversion settings	Number of instruction pulses for one revolution of motor/encoder	√	√
	In the background, the amount of movement when the worktable rotates for one revolution	√	√
	Numerator of gear ratio	√	√
	Denominator of gear ratio	√	√
Mode/parameter settings	Encoder mode	√	×
	Linear/rotation mode settings	√	√
	Software limit	√	√
	Software error response	√	√
	Axis speed setting	√	√
	Torque limit	√	×
	Probe setting	×	√
	Output settings	×	√
	Hardware limit logic	√	×
Homing setting	Home signal	√	√
	Positive limit	√	√
	Negative limit	√	√
	Z Signal	√	×
	Homing direction	√	√
	Home input detection direction	√	√
	Homing list	√	√
	Homing velocity	√	√
	Homing approach speed	√	√
	Homing acceleration	√	√
	Homing timeout	√	√
	Negative limit terminal setting	×	√
	Positive limit terminal setting	×	√
	Home signal settings	×	√
Online debugging	Monitoring list	√	√
	Motion debugging	√	√

11.1.5 Axis System Variable

Name	Data type	Function
wAxisID	WORD	Axis ID
iAxisType	INT	Axis type 0: Fieldbus servo axis 1: Fieldbus encoder axis 2: Local pulse axis

Name	Data type	Function
		3: Local encoder axis
dwPulseData	DWORD	Number of pulses required for one revolution of the current axis
fDistanceData	REAL	The worktable’s moving distance when the current axis rotates for one revolution
diGearRatioNum	DINT	Numerator of gear ratio
dwGearRatioDen	DWORD	Denominator of gear ratio
iLineRotateMode	INT	Linear/rotation mode selection
bSWLimitEnable	BOOL	Axis soft limit switch
fRotation	REAL	Number of rotation cycles of rotating axis
fMaxPLimit	REAL	Maximum positive soft limit value of linear axis mode
fMaxNLimit	REAL	Maximum negative soft limit value of linear axis mode
fAxisErrorDec	REAL	Axis error deceleration
fMaxVelocity	REAL	Maximum axis velocity limit
fMaxAcceleration	REAL	Maximum axis acceleration limit
fMaxDeceleration	REAL	Maximum axis deceleration limit
fMaxJerk	REAL	Maximum axis jerk limit
fMaxJogSpeed	REAL	Maximum speed of axis in Jog mode
fMaxPTorque	REAL	Maximum positive torque (fieldbus servo axis)
fMaxNTorque	REAL	Maximum negative torque (fieldbus servo axis)
bHWPLimitEnable	BOOL	Hardware positive limit enable signal
iHWPLimitID	INT	Hardware positive limit terminal ID
bHWNLimitEnable	BOOL	Hardware negative limit enable signal
iHWNLimitID	INT	Hardware negative limit terminal ID
bTouchProbeID1	BOOL	Probe terminal 1 enable signal
iTouchProbeID1	INT	Probe terminal 1 ID
bTouchProbeID2	BOOL	Probe terminal 2 enable signal
iTouchProbeID2	INT	Probe terminal 2 ID
bServoError	BOOL	Servo alarm enable signal
iServoErrorID	INT	Servo alarm terminal ID
bServoEnable	BOOL	Servo enable signal
iServoEnableID	INT	Servo enable terminal ID
bClearError	BOOL	Clear servo alarm enable signal
iClearErrorID	INT	Clear servo alarm terminal ID
iPulseMode	INT	Pulse axis control mode 0: Pulse + direction 1: Positive and negative pulse mode 2: Quadrature encoded pulse mode
bVirtualMode	BOOL	Virtual axis mode
iHomeMode	INT	Homing mode selection
bHomeDirection	BOOL	Homing direction
fMaxHomeSpeed	REAL	Maximum axis homing speed limit
fMaxHomeAcc	REAL	Maximum axis homing acceleration limit
fDecModuleSpeed	REAL	Maximum speed on deceleration module when axis homes
fWaitZSpeed	REAL	Maximum speed while waiting for Z signal when axis homes
bHomeSignal	BOOL	Home enable signal
iHomeSignalID	INT	Home signal terminal ID
bZSignal	BOOL	Z signal enable signal

Name	Data type	Function
iZSignalID	INT	Z signal terminal ID
bPowerState	BOOL	Axis enable state
iPLCopenState	INT	PLCOpen state
iAxisCfgState	INT	Axis communication state
iAxisError	INT	Axis error
iServoError	INT	Servo error
bMotionState	BOOL	Motion state
bHardwarePLimit	BOOL	Hardware positive limit
bHardwareNLimit	BOOL	Hardware negative limit
bSoftwarePLimit	BOOL	Software positive limit
bSoftwareNLimit	BOOL	Software negative limit
bHomeState	BOOL	Home switch
bMotionDirection	BOOL	Motion direciton
fSetPosition	REAL	Set position
fActPosition	REAL	Actual position
fSetVelocity	REAL	Set velocity
fActVelocity	REAL	Actual velocity
fSetTorque	REAL	Set torque
fActTorque	REAL	Actual torque
bPowerOn	BOOL	Enable
bReset	BOOL	Reset
bStop	BOOL	Stop
bImmediateStop	BOOL	Immediate stop
bSetPosition	BOOL	Set current position
bHome	BOOL	Home
bJogP	BOOL	Jog+
bJogN	BOOL	Jog-
bMoveAbs	BOOL	Absolute position
bMoveRel	BOOL	Relative position
bMoveVel	BOOL	Moving velocity
bTorque	BOOL	Torque control
bServoDebug	BOOL	Enter servo debugging mode
iDebugMotionType	INT	Debug mode motion type 0: Absolute motion 1: Relative motion 2: Moving velocity 3: Torque control
fPreSetPosition	REAL	Pre-set position
fPositionOffset	REAL	Home position offset
fJogVelocity	REAL	Jog velocity
fTargetPos1	REAL	Target position
fTargetVel1	REAL	Target velocity
fTargeAcc1	REAL	Target acceleration
fTargetDec1	REAL	Target deceleration
fTargetJerk1	REAL	Target jerk
fTargetTorque	REAL	Target torque
fTargetTorqueSlop	REAL	Torque slop
fLimitVelocity	REAL	Limit velocity

Name	Data type	Function
fOffsetPosition	REAL	Position offset
dwPosOffsetForResiduals	DWORD	Position offset for residuals
dwTurn	DWORD	Encoder overflow times
iControlWord	INT	Control word (read-only, PDO data, 0*6040)
iStatusWord	INT	Status word (read-only, PDO data, 0*6041)
diSetPosition	DINT	Set position (read-only, PDO data, 0*607A)
diActPosition	DINT	Actual position (read-only, PDO data, 0*6064)
diSetVelocity	DINT	Set velocity (read-only, PDO data, 0*60FF)
diActVelocity	DINT	Actual velocity (read-only, PDO data, 0*606C)
diSetTorque	DINT	Set torque (read-only, PDO data, 0*6071)
diActTorque	DINT	Actual torque (read-only, PDO data, 0*6077)
diDo	DINT	Digital quantity output (read-only, PDO data, 0*60FE)
diDI	DINT	Digital quantity input (read-only, PDO data, 0*60FD)
iModeOfOperation	INT	Mode of operation (read-only, PDO data, 0*6060)
iTouchFunction	INT	Probe function setting (read-only, PDO data, 0*60B8)
diTouch1PPos	DINT	Probe 1 rising edge position (read-only, PDO data, 0*60BA)
diTouch2PPos	DINT	Probe 2 rising edge position (read-only, PDO data, 0*60BC)
diTouch1NPos	DINT	Probe 1 falling edge position (read-only, PDO data, 0*60BB)
diTouch2NPos	DINT	Probe 2 falling edge position (read-only, PDO data, 0*60BD)
iTouchStatus	INT	Probe status (read-only, PDO data, 0*60B9)

11.1.6 Axis Instruction List

Instruction	Name
MC_Power	Axis power-on instruction
MC_Reset	Axis reset instruction
MC_ReadStatus	Axis read status instruction
MC_ReadPosition	Read actual position instruction
MC_ReadVelocity	Read actual velocity instruction
MC_SetPosition	Set position instruction
MC_MoveRelative	Relative positioning instruction
MC_MoveAbsolute	Absolute positioning instruction
MC_MoveVelocity	Velocity instruction
MC_Jog	Continuous operation instruction
MC_Home	Home instruction
MC_SetOverride	Set override instruction
MC_Stop	Stop instruction
MC_Halt	Halt instruction
MC_ImmediateStop	Immediate stop instruction
MC_MoveSuperImposed	Motion Superimposed instruction
MC_TouchProbe	Probe instruction
MC_MoveFeed	Interrupt fixed-length instruction
MC_MoveBuffer	Multi-segment positioning instruction
MC_ReadAxisError	Read axis error instruction
MC_ReadDigitalInput	Read digital input instruction
MC_MoveVelocityCSV	CSV-based velocity instruction with adjustable pulse width
MC_SyncMoveVelocity	CSV-based synchronous velocity control instruction

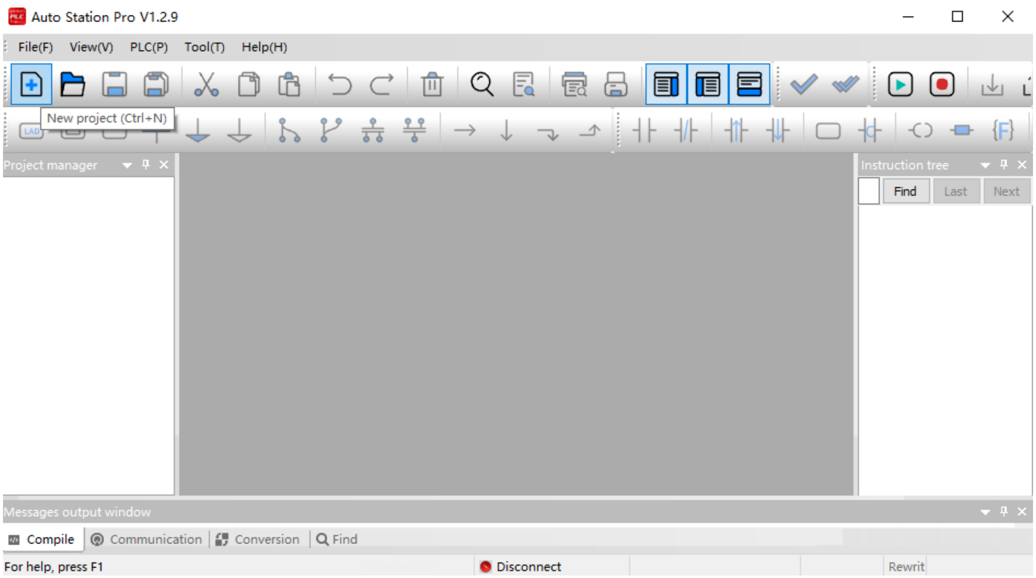
Instruction	Name
	with adjustable pulse width
MC_FollowPosition	Synchronous position instruction based on CSP mode
MC_SyncTorqueControl	Synchronous torque control instruction
MC_ReadActualTorque	Read actual torque instruction
MC_CamIn	Cam in
MC_CamOut	Cam out
MC_DigitalCamSwitch	Electronic cam tappet control
MC_GearIn	Gear in
MC_GearOut	Gear out
MC_Phasing	Principal axis phasing
MC_SaveCamTable	Save Cam Table
MC_GenerateCamTable	Update Cam Table
MC_FlyingShearIn	Flying shear in
MC_FlyingShearOut	Flying shear out
MC_TraceShearIn	Trace shear in
MC_TraceShearOut	Trace shear out
MC_MoveLiear	Linear interpolation instruction
MC_MoveCircular2D	Planar arc interpolation instruction
MC_GroupSetOverRide	Axis group speed regulation instruction
MC_GroupStop	Axis group stop instruction
MC_GroupHalt	Axis group halt instruction
MC_GroupImmediateStop	Axis group immediate stop instruction
MC_ReadGroupVelocity	Read axis group resultant velocity instruction

11.2 Motion Control Axis Setting Steps

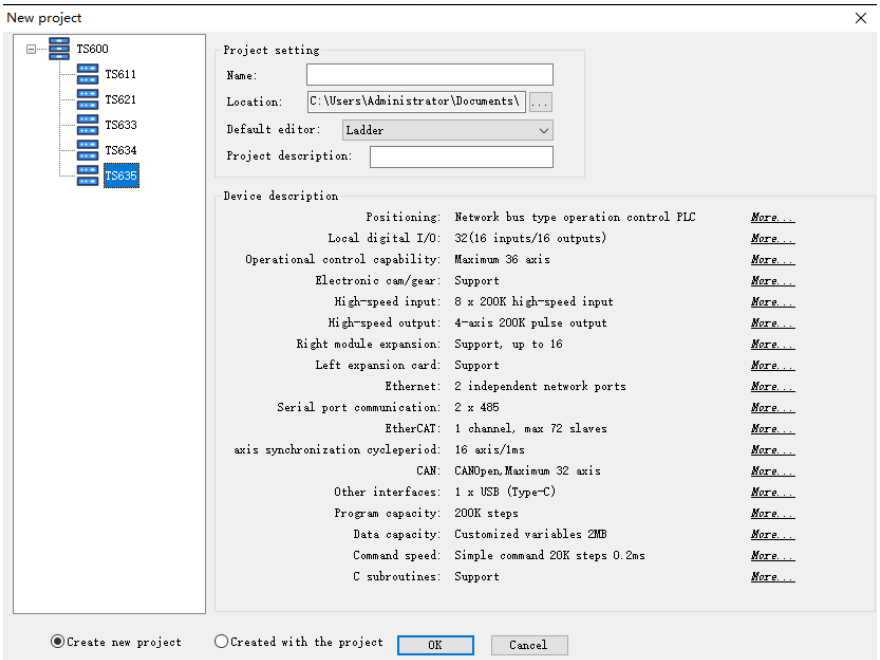
11.2.1 Create a New Project Document

This routine will create a new fieldbus servo axis and a local pulse axis, and realize simple axis control through instructions.

Step 1 Open Auto Station Pro > "New project", as shown in the following figure.

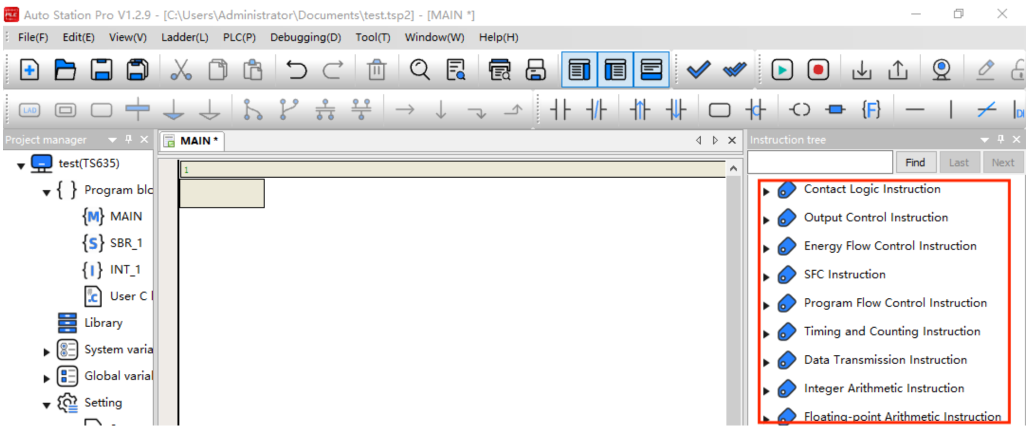


Step 2 Fill in "Name" and set the corresponding "Location". Select ladder diagram for the default editor, and the project description can be left blank.

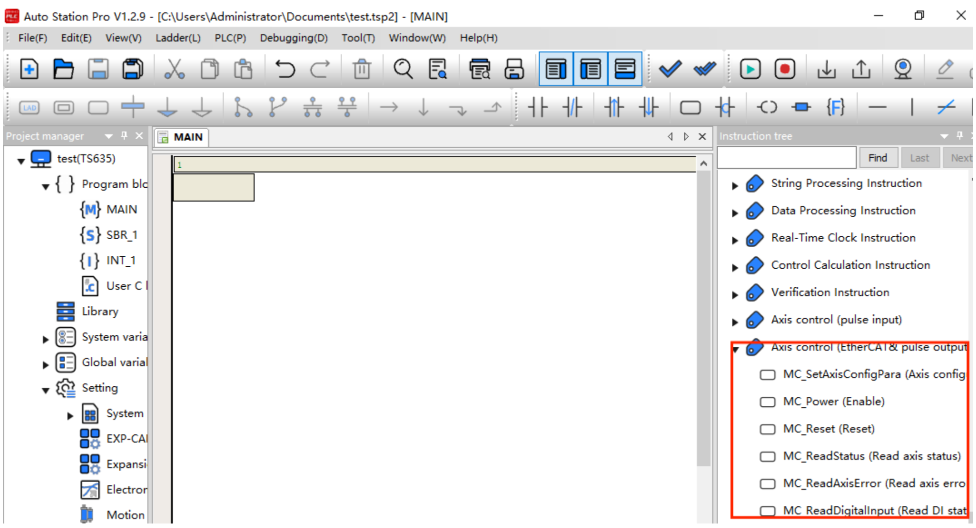


Step 3 After the project is successfully created, click "Instruction tree".

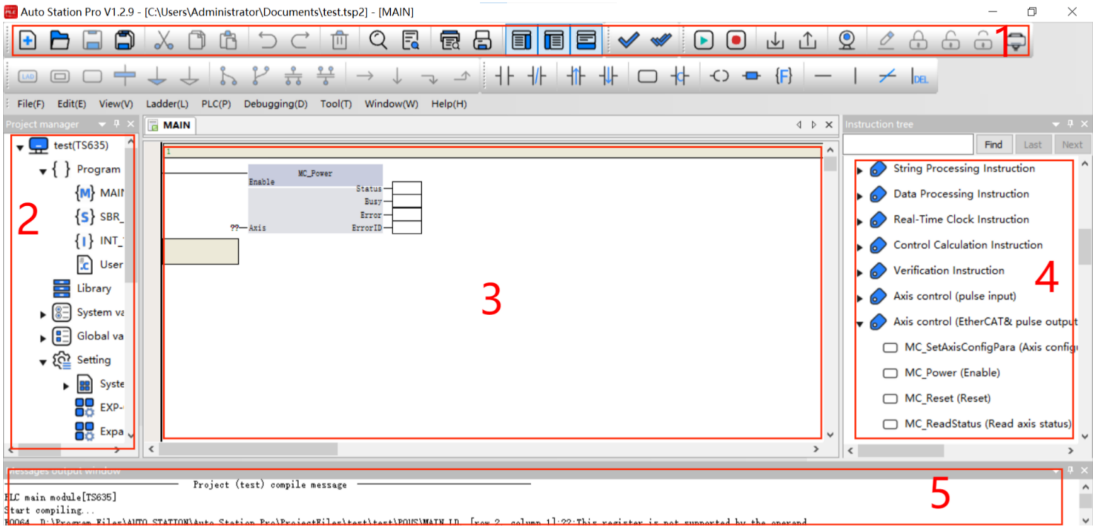
- Instruction tree instruction



- Motion control instruction of instruction tree



Step 4 Edit in the main interface after the project is successfully created



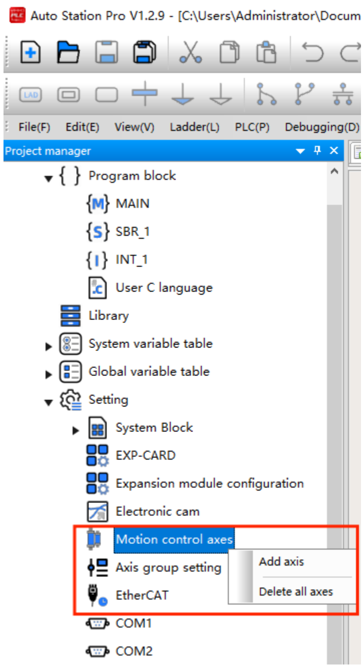
No.	Description
1	Toolbar
2	Project manager
3	Program editing area
4	Instruction tree
5	"Messages output" window

11.2.2 Create Project Configuration

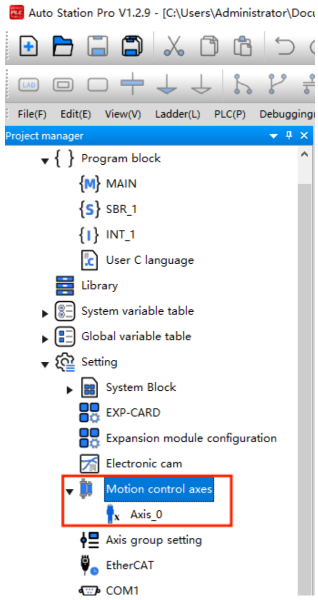
There are two ways to realize servo axis control, one is pulse control, the other is fieldbus control. The configuration methods of the two controls are explained below.

- **For fieldbus control, add a filedbus axis**

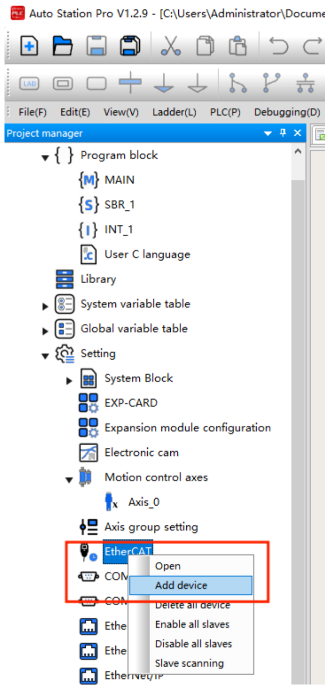
Step 1 Left-click to choose "Motion control axes" and right-click on "Add axis".



Step 2 Axis AXIS_0 is created successfully

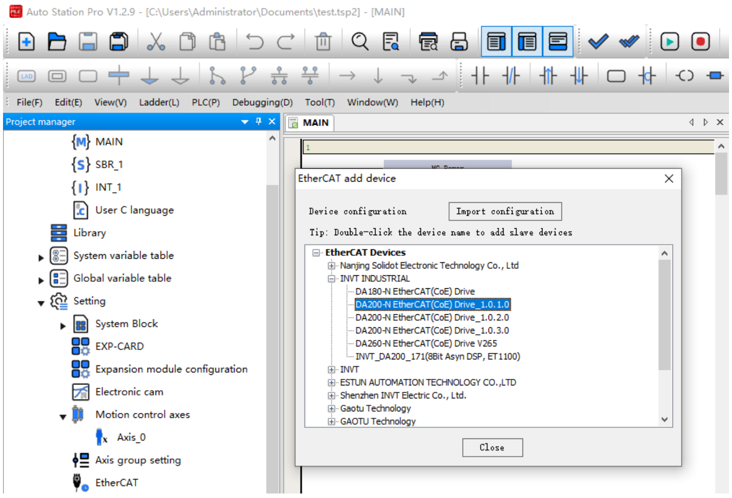


Step 3 To create an EtherCAT slave, left-click to choose "EtherCAT" and right-click on "Add device".

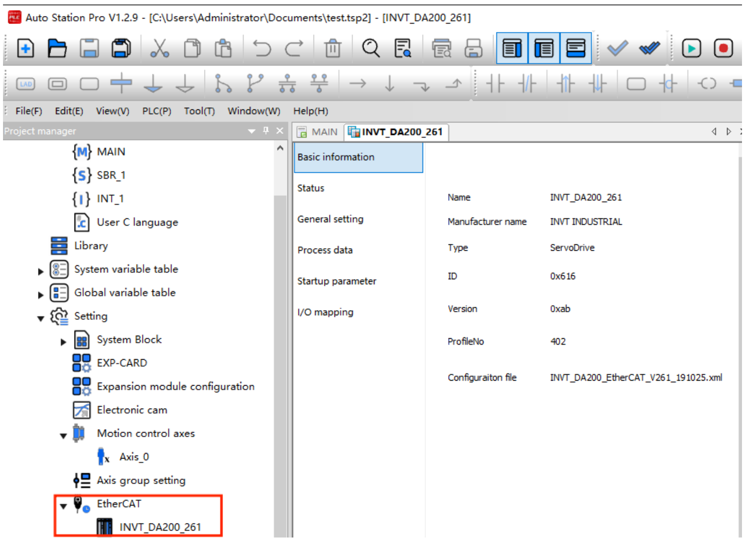


Step 4 Left double-click to select "INVT_DA200_171 device".

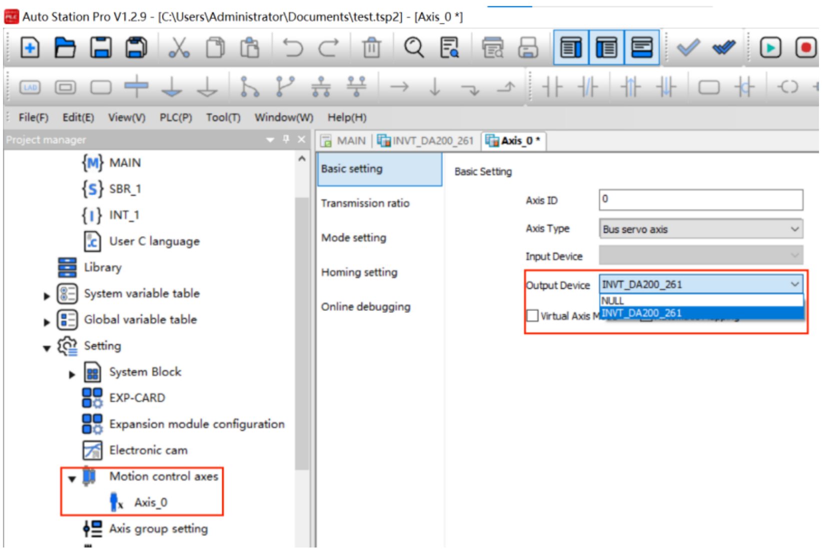
Note: If there is no option of INVT-DA200_171, select to import the XML file of DA200.



Step 5 The DA200 servo driver slave is created, and the EtherCAT add device interface is closed

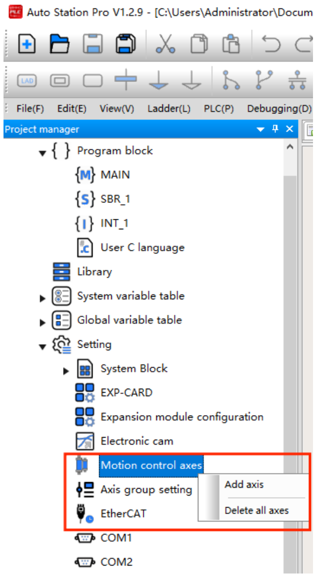


Step 6 Choose "AXIS_0" and the output device "INVT_DA200_171" to associate the axis AXIS_0 with the INVT_DA200_171 servo driver

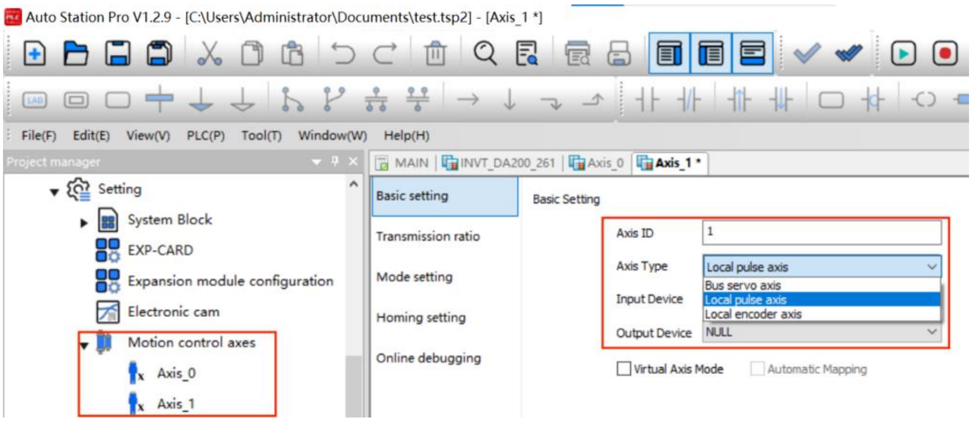


- **Pulse control**

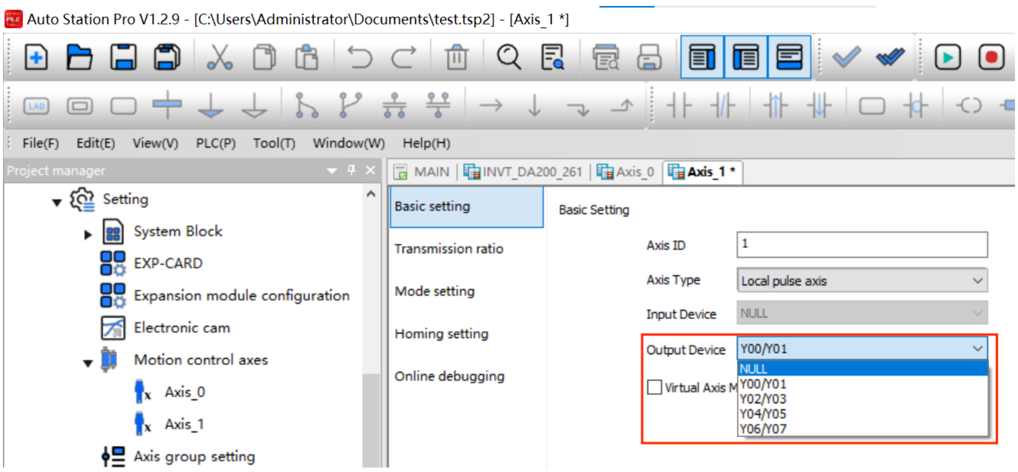
Step 1 Left-click to choose "Motion control axes" and right-click on "Add axis".



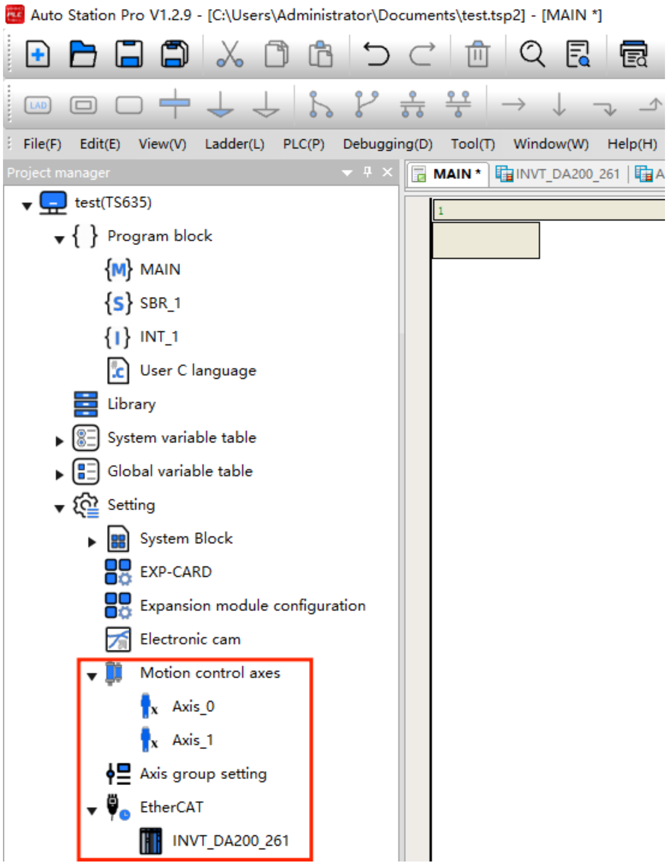
Step 2 Choose "Local pulse axis".



Step 3 Choose Y0/Y1 for output device.



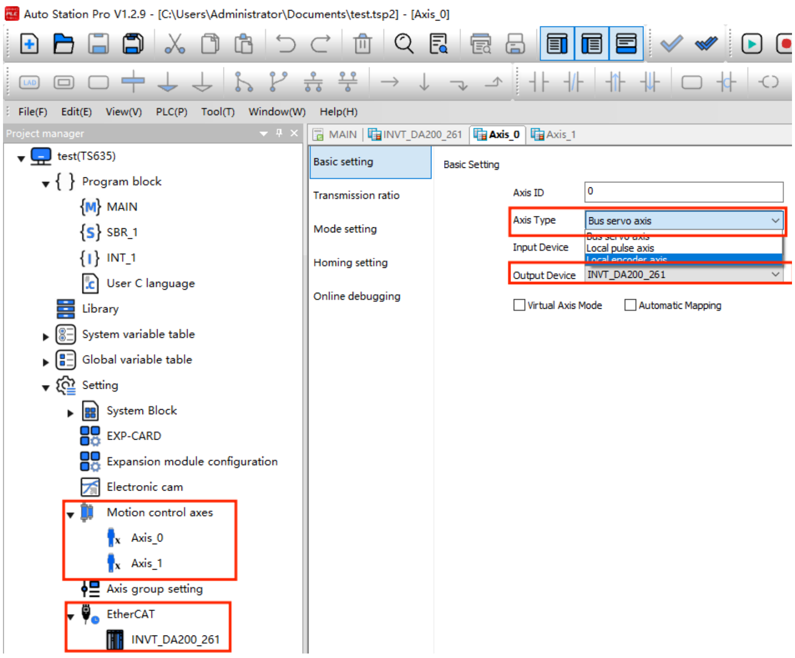
Step 4 The configuration of local pulse axis and fieldbus servo axis is completed.



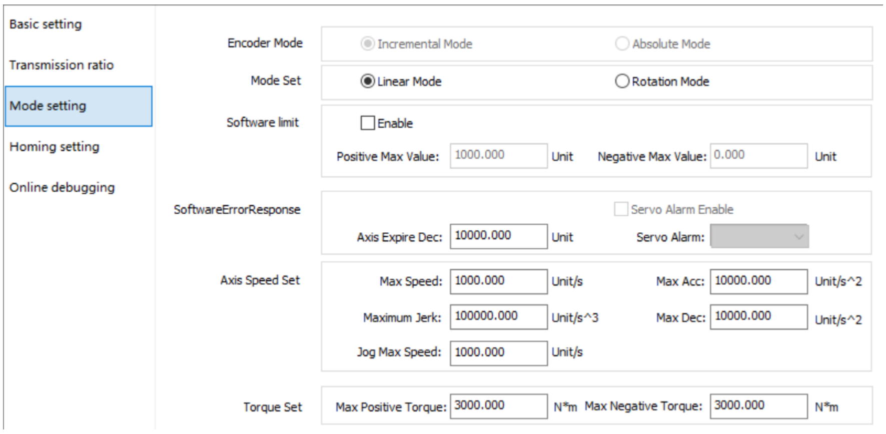
11.2.3 Set Axis Parameters

11.2.3.1 Fieldbus servo axis

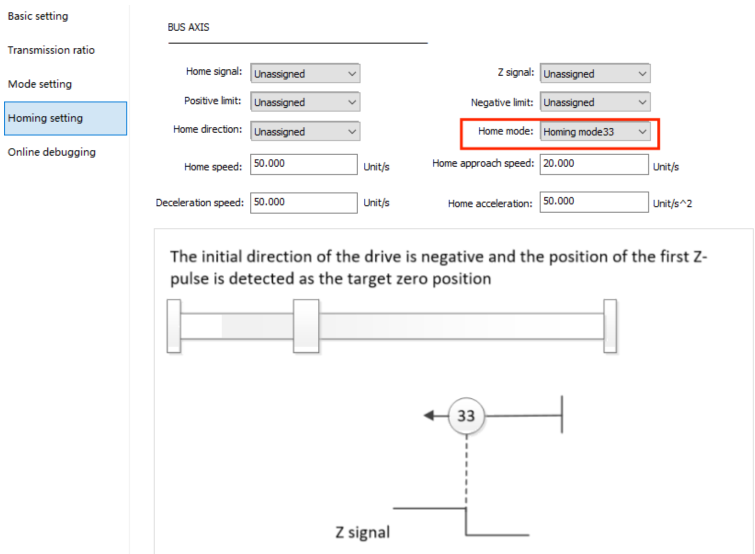
- Basic setting interface of fieldbus servo axis



- Fieldbus servo axis mode setting interface



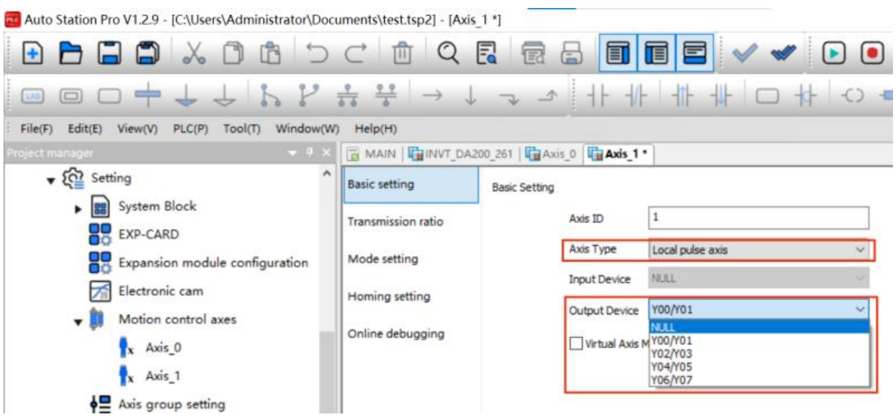
- Choose the way to home



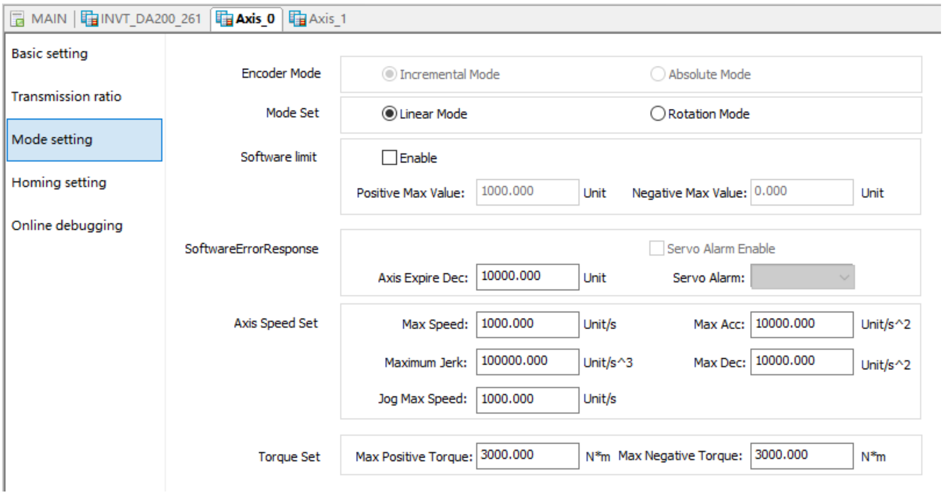
Note: The drop-down box selection of home signal, Z signal, positive limit, negative limit and homing mode is temporarily invalid.

11.2.3.2 Local pulse axis

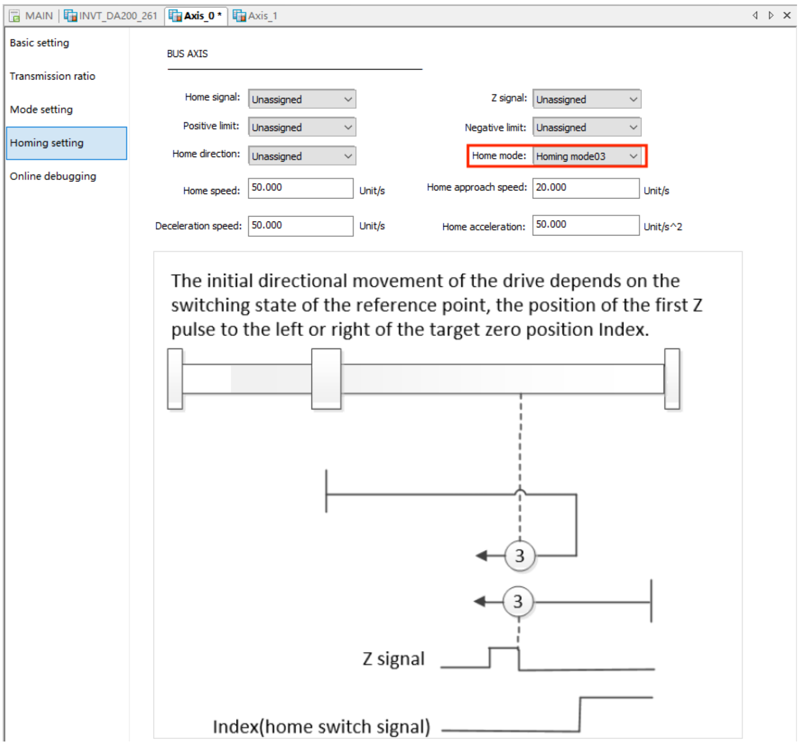
- Basic setting interface of local pulse axis



- Local pulse axis mode setting interface

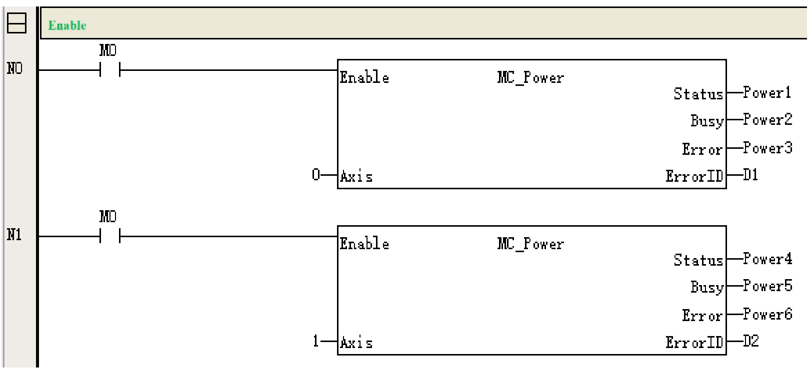


- Local pulse axis homing mode setting

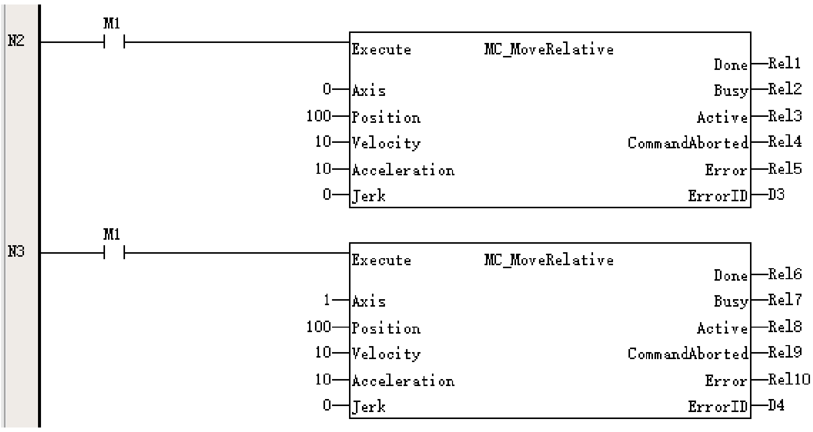


11.2.4 Write a Program

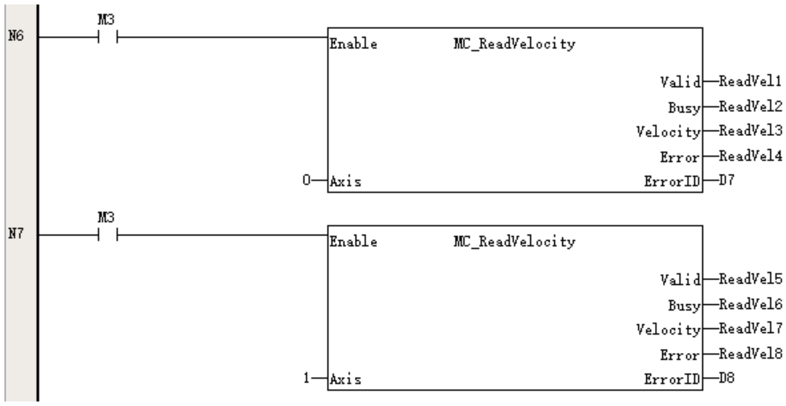
Step 1 Call the instruction MC_Power



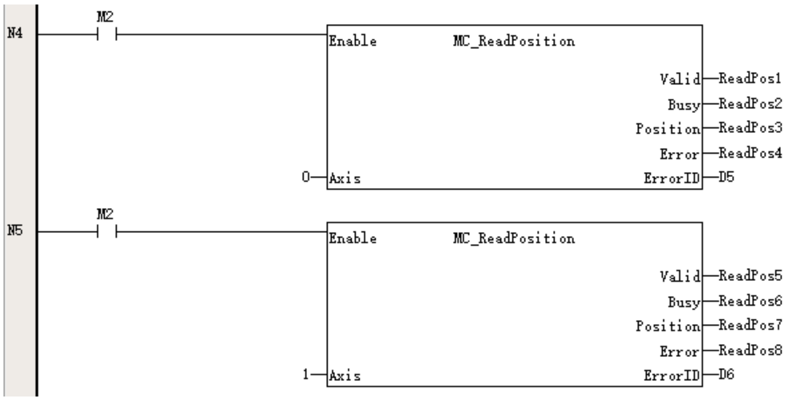
Step 2 Call the instruction MC_MoveRelative



Step 3 Call the MC_ReadVelocity function block



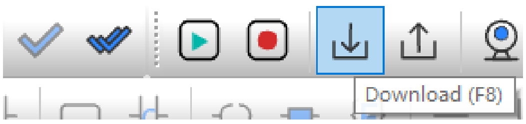
Step 4 Call the MC_ReadPosition function block



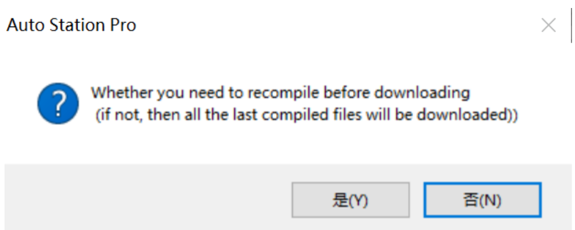
11.2.5 Download Project

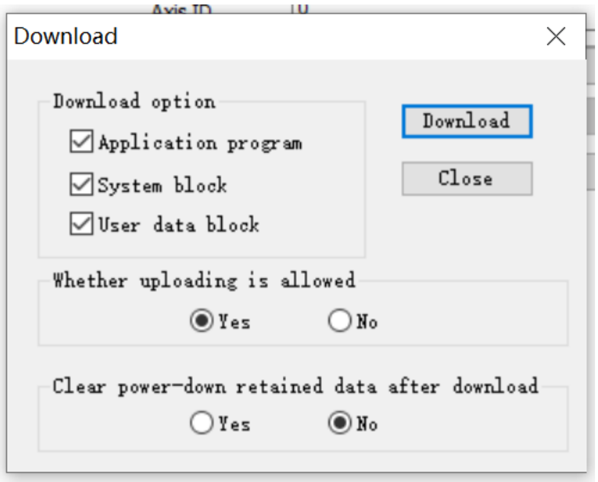
After the program is compiled, please download it according to the following steps:

Step 1 Download



Step 2 Recompile

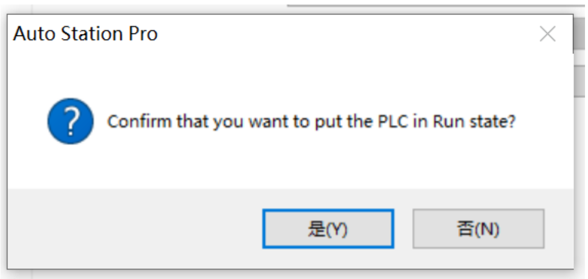




Step 3 Download

The information output window shows that the execution is correct.

Step 4 Put PLC in RUN state



11.2.6 Basic Motion

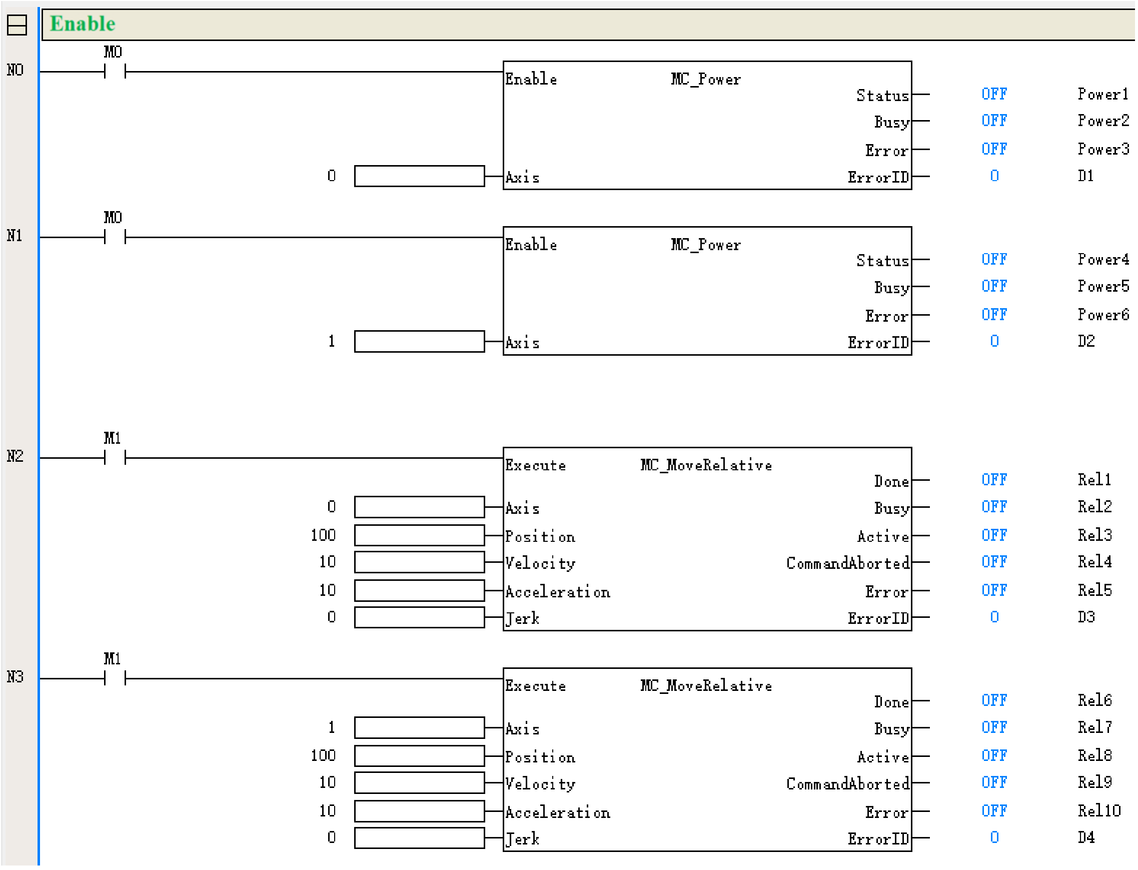
11.2.6.1 Preparation

Click monitor to enter the RUN state, as shown in the following figure.

11.2.6.2 PLC Program Control

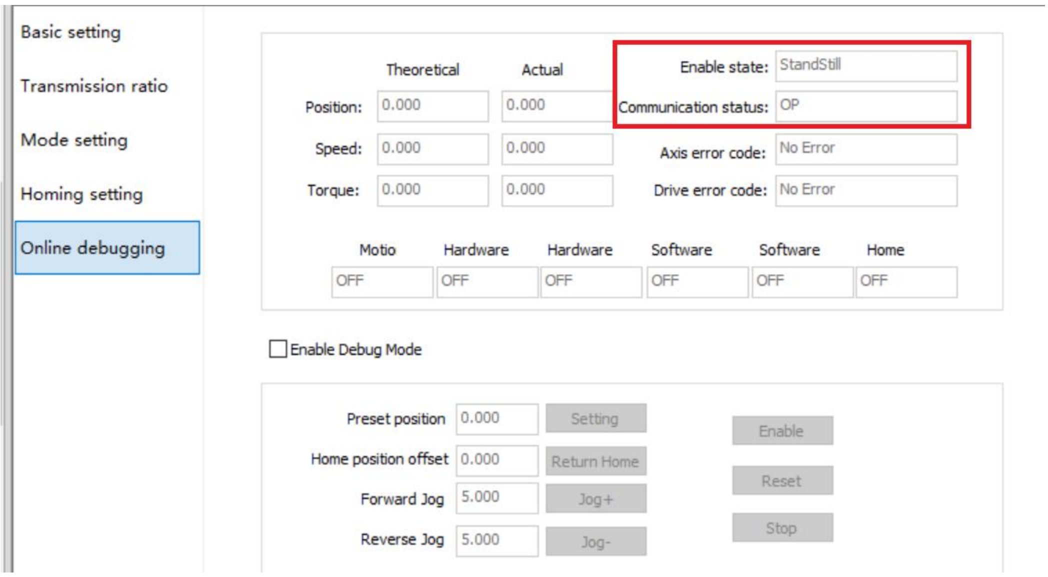
Step 1 Power-on control

Set M0 element to ON, observe whether the output Error of MC_Power function block of axis 0 and axis 1 is FALSE, if so, it means that the enable is successful; If it is TRUE, it means that the enable is not successful. Observe the corresponding ErrorID value and find the corresponding error.



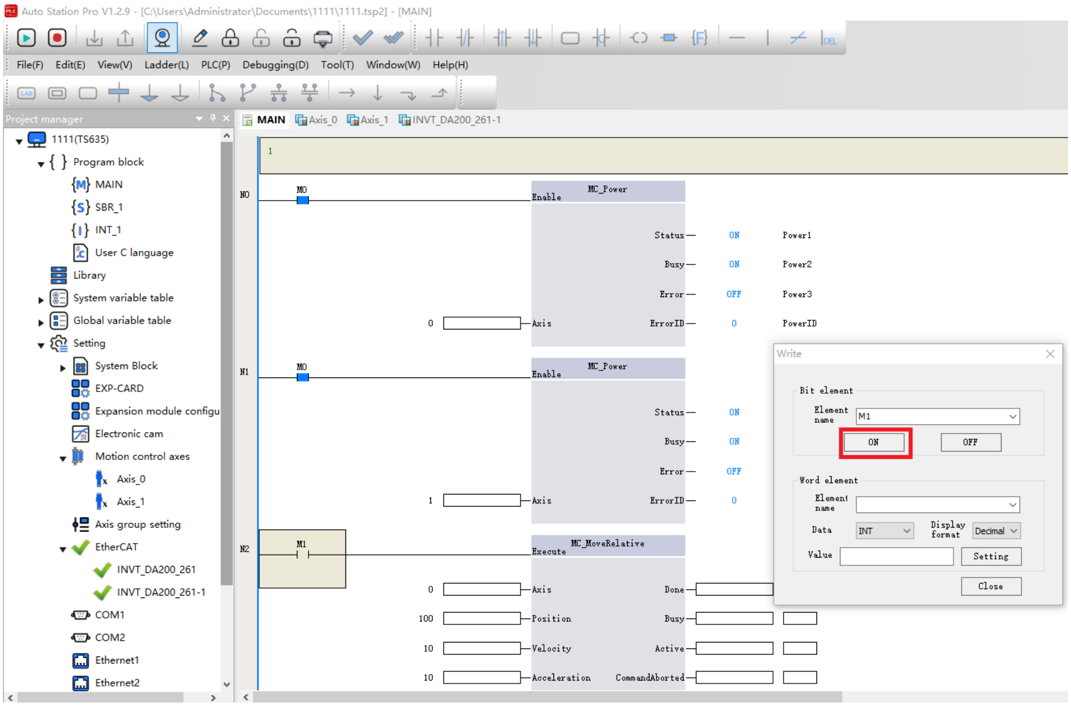
Step 2 Double-click to select Axis 0, and click online debugging to observe the enable status and communication status of the axis

As shown in the following figure, if the enable state is StandStill and the communication state is OP, it means that the communication works normally and the axis enable is successful.

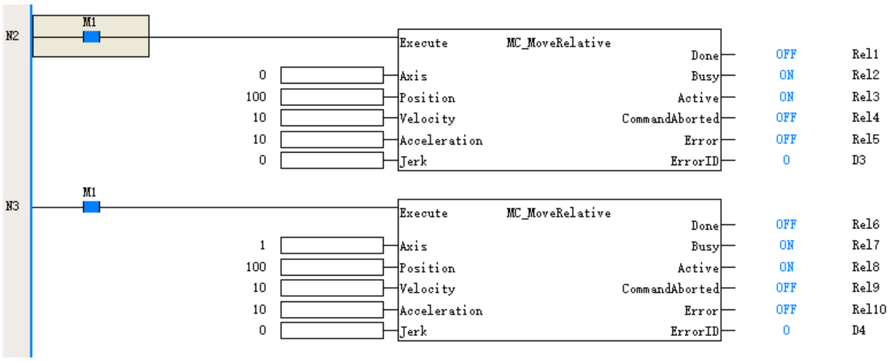


Step 3 Relative position control

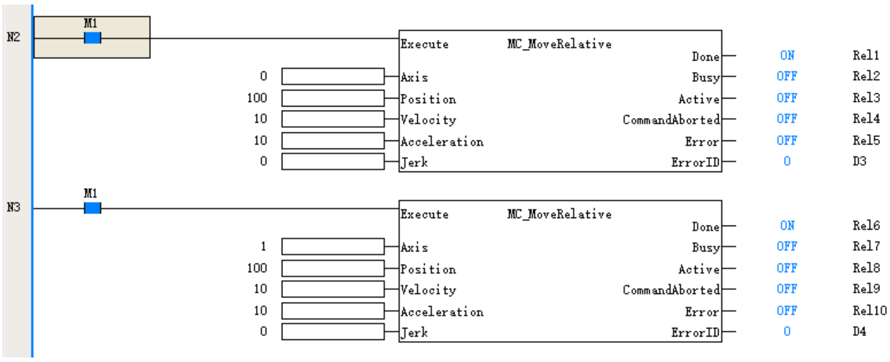
Call the MC_MoveRelative function block, set the parameters as follows, and set the M1 element to ON.



While the MC_MoveRelative function block is running:



After the MC_MoveRelative function block runs:



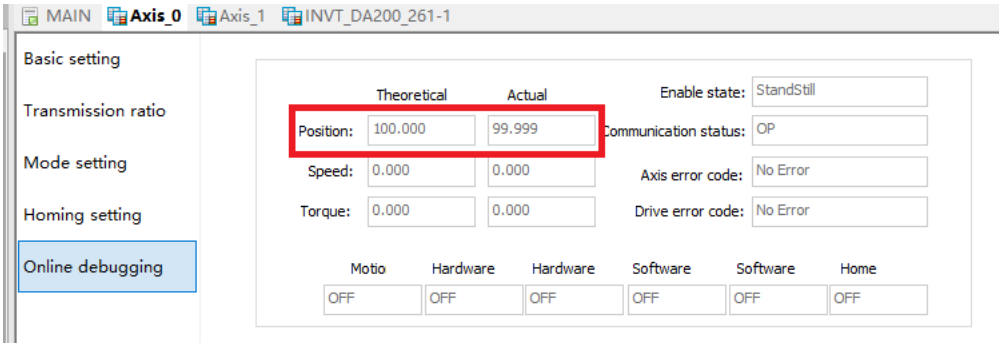
There are two ways to observe the actual running distance of Axis 0 and Axis 1:

Mode 1: Through online debugging.

Mode 2: By reading the return value of the function block.

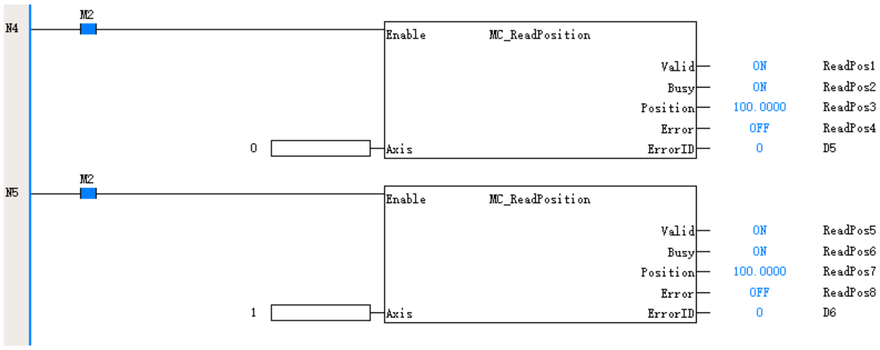
- Online debugging

  Open online debugging, and the actual running distance of Axis 0 is as follows:



- Read the return value of function block

  As shown in the following figure, the running distance between Axis 0 and Axis 1 is read to be 100



11.3 Motion Control Axis Configuration

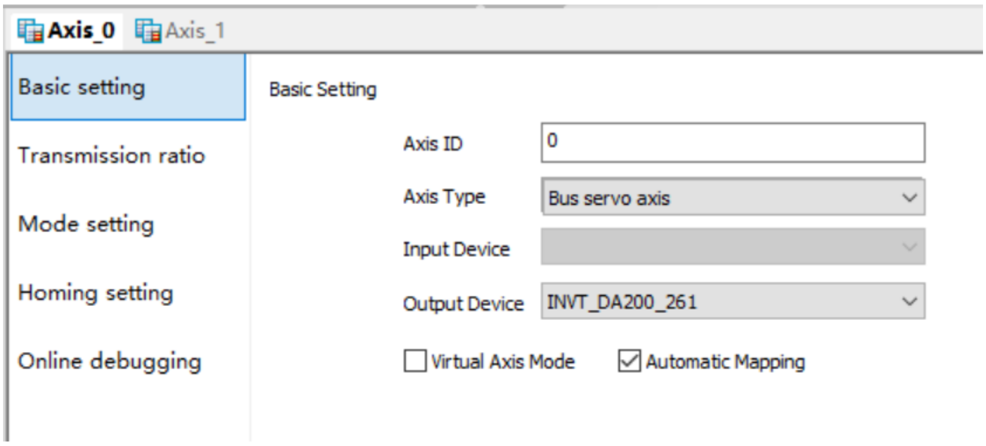
11.3.1 Comparison of Fieldbus Servo Axis and Local Pulse Axis

The local pulse output and the EtherCAT driver are controlled with the same set of instructions, and the main differences between them are listed here.

Item	Local pulse output	EtherCAT fieldbus driver
Different types of axes	Local pulse axis is required.	Fieldbus servo axis is required.
Different output devices	Local I/O terminals need to be set up in groups of Y0/Y1, Y2/Y3, Y4/Y5, and Y6/Y7, with each group containing two terminals.	The XML device description file needs to be configured and added to PDO mapping.
Pulse output form	It supports three ways of output, namely "pulse + direction", "positive and negative pulse" and "quadrature encoded pulse".	No setting is required.
Probe function	Two probes are supported, and any two points in the body X0 – X17 can be selected as probe signal input in "Mode setting".	Probes need to be configured according to the EtherCAT driver application manual.
Homing setting	In the homing setting interface, the signals needed for local pulse homing can be selected, such as home enable signal, Z signal, positive and negative hard limit signal, etc.	It supports 1-35 homing mode setting specified in 402 protocol, and limit and home signals need to be set in the servo driver.

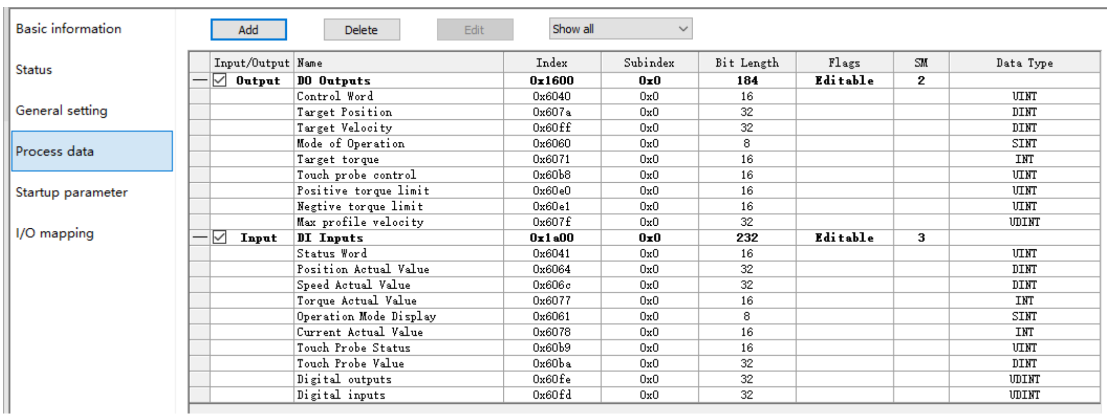
11.3.2 Basic Settings

- The basic setting interface is as shown in the following figure.



Name	Description
Axle number	● Each axis has an unique axis number, which cannot be changed manually, and is arranged in the order of adding. ● The axis number is unique, and can be used as the input parameter of MC instruction to access the axis.
Axis type	The axis type options include fieldbus servo axis, local pulse axis and local encoder axis.
Input device	Limited to local encoder axis only.
Output device	Valid only in fieldbus servo axis and local pulse axis mode. ● If it is a fieldbus servo axis, choose a EtherCAT device; ● If it is a local pulse axis, choose the combination of high-speed output terminals, including Y0/Y1, Y2/Y3, Y4/Y5 and Y6/Y7.
Virtual axis mode	Valid only in fieldbus servo axis and local pulse axis mode. After checking the virtual axis mode, the axis will no longer control the driver or high-speed output terminal selected by the output device, but generate a virtual servo axis internally to execute the motion control instruction.
Automatic mapping variables	Valid only in fieldbus servo mode. The EtherCAT master and slaves communicate with each other periodically based on PDO, and the axis is connected to the object dictionary of the EtherCAT slave through variables. When automatic mapping is selected, the mapping process is automatically allocated and cannot be configured manually.

- The fieldbus servo axis PDO cycle variables are as follows:



Input/Output	Name	Index	Subindex	Bit Length	Flags	SM	Data Type
Output	DO Outputs	0x1600	0x0	184	Editable	2	
	Control Word	0x6040	0x0	16			UINT
	Target Position	0x607a	0x0	32			DINT
	Target Velocity	0x60ff	0x0	32			DINT
	Mode of Operation	0x6060	0x0	8			SINT
	Target torque	0x6071	0x0	16			INT
	Touch probe control	0x60b8	0x0	16			UINT
	Positive torque limit	0x60e0	0x0	16			UINT
	Negtive torque limit	0x60e1	0x0	16			UINT
	Max profile velocity	0x607f	0x0	32			UDINT
Input	DI Inputs	0x1a00	0x0	232	Editable	3	
	Status Word	0x6041	0x0	16			UINT
	Position Actual Value	0x6064	0x0	32			DINT
	Speed Actual Value	0x606c	0x0	32			DINT
	Torque Actual Value	0x6077	0x0	16			INT
	Operation Mode Display	0x6061	0x0	8			SINT
	Current Actual Value	0x6078	0x0	16			INT
	Touch Probe Status	0x60b9	0x0	16			UINT
	Touch Probe Value	0x60ba	0x0	32			DINT
	Digital outputs	0x60fe	0x0	32			UDINT
	Digital inputs	0x60fd	0x0	32			UDINT

- Parameters to be set for unit conversion.

Name	Function
Number of pulses for one revolution of motor/encoder	According to the encoder resolution, set the number of pulses for one revolution of motor
Is a speed change device used	Specify whether a speed change device is used
The amount of movement when the worktable rotates for one revolution	The amount of movement on the workpiece side for 1 revolution when the speed change device is used
Gear ratio on the workpiece side	Set gear ratio on the workpiece side
Gear ratio on the motor side	Set gear ratio on the motor side

When the fieldbus driver (local pulse axis) controls the motor, pulse unit is used, and common measurement units such as mm, ° and inch are used on the motion control instruction side, which are called Unit. The two units are converted to each other internally according to the configuration parameters. The mode of conversion can be divided into the following cases:

1. Without a speed change device

When no speed change device is used, the conversion formula from Unit to pulse unit is as follows:

$$\text{Number of pulses (pluse)} = \frac{\text{The number of pulses for one revolution of the motor/encoder[DINT]}}{\text{Movement amount of one revolution of the worktable[DINT]}} \times \text{Moving distance (Unit)}$$

Take the 23-bit encoder of INVT as an example, set the parameters as follows:

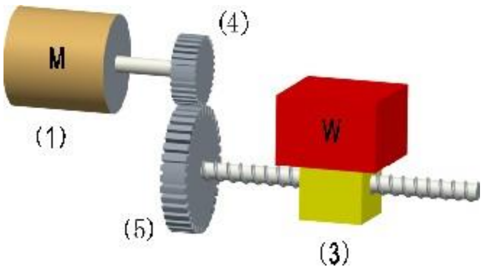
Number of pulses for one revolution of motor/encoder=8388608

The amount of movement when the worktable rotates for one revolution=1

When the target displacement given by the relative positioning instruction is 10, the actual pulse amount sent by the motion control axis is 83886080, and the motor rotates 10 revolutions at this time.

2. With a speed change device

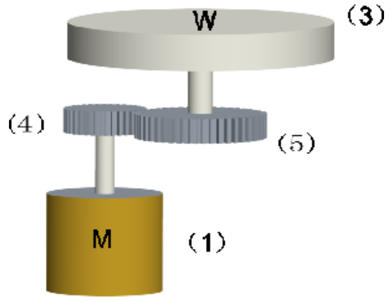
A. Typical operating conditions in linear mode are shown in the following figure.



Where, (1) is a servo motor, (3) is a workpiece, (4) is the denominator of gear ratio, and (5) is the numerator of gear ratio.

$$\text{Number of pulses (pluse)} = \frac{\text{The number of pulses for one revolution of the motor[DNIT]} \times \text{gear ratio numerator[DINT]}}{\text{Movement amount of one revolution of the worktable[DINT]} \times \text{Gear ratio denominator[DNIT]}} \times \text{Moving distance (Unit)}$$

B. Typical operating conditions in circular mode are shown in the following figure.



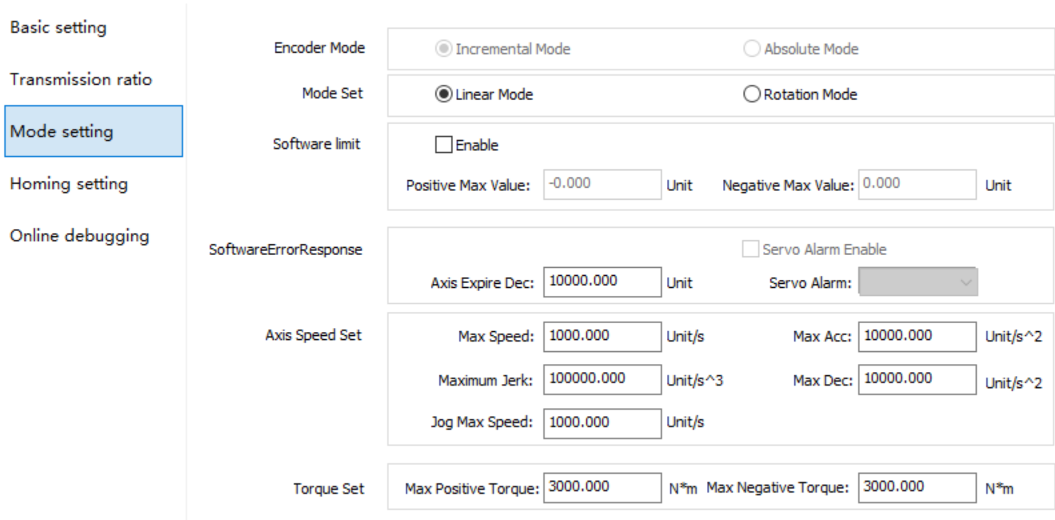
The conversion formula from Unit to pulse unit is as follows:

$$\text{Number of pulses (pluse)} = \frac{\text{The number of pulses for one revolution of the motor[DNIT]} \times \text{gear ratio numerator[DINT]}}{\text{Movement amount of one revolution of the worktable[DINT]} \times \text{Gear ratio denominator[DNIT]}} \times \text{Moving distance (Unit)}$$

11.3.3 Mode Setting

11.3.3.1 Configuration Interface

The mode setting interface is as follows:



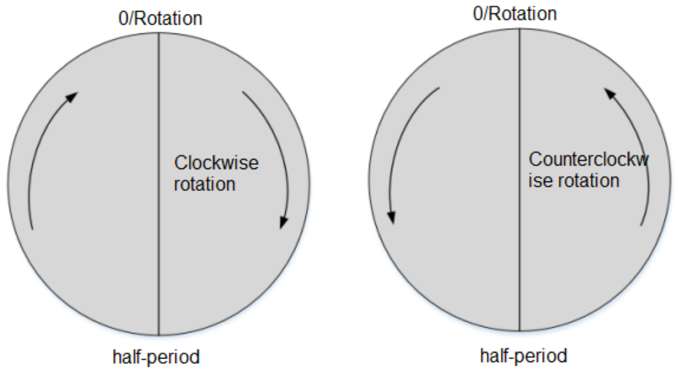
11.3.3.2 Mode Setting

According to the actual working conditions, the motion control axis can be set to linear mode and rotation mode.

- Linear mode
  - ✧ Linear mode is usually used in devices with mechanical action range in X-Y linear coordinate system. Linear mode usually contains a zero point.
  - ✧ When the feedback position increases, it indicates forward motion, otherwise it indicates reverse motion.
  - ✧ It is allowed to set positive software limit and negative software limit. When software limit is enabled, the axis can only move within the limit range.
  - ✧ Absolute positioning mode: When the target position is greater than the starting position, then move forward for the distance between the target position and the starting position; When the target position is less than the starting position, then move backward for the distance between the starting position and the target position.

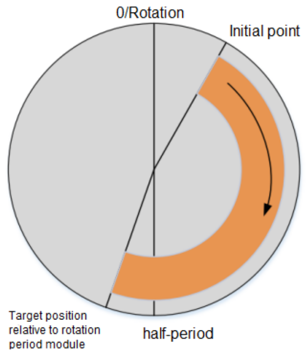
- ✧ Relative positioning mode: When the target displacement is greater than 0, move forward for the distance of target displacement, and when the target displacement is less than 0, move backward for the distance of "target displacement".
- ✧ The way to process velocity instructions in linear mode: If the target speed is greater than 0, it will move forward, otherwise it will move backward.

- Rotation mode
  - ✧ The rotation mode is a mode in which a counter repeats counting infinitely in a set orientation. It is usually used in turntables or reels.
  - ✧ The rotation mode usually includes a zero point and a rotation cycle.
  - ✧ In rotation mode, if the feedback position increases, it is considered as clockwise motion, and if the feedback position decreases, it is considered as counterclockwise motion.

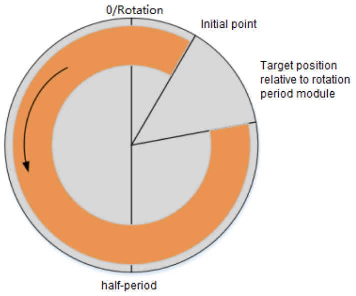


  - ✧ There is no soft limit in rotation mode.
  - ✧ Phase positioning processing: If the target displacement is greater than> 0, move the distance of target displacement clockwise, and if the target displacement is less than 0, move the distance of target displacement counterclockwise.
  - ✧ How to deal with absolute positioning:

Forward: First, the target displacement is modulated against the rotation cycle, and then the axis runs clockwise from the starting position of the axis to the target position.



Reverse: First, the target displacement is modulated against the rotation cycle, and then the axis runs counterclockwise from the starting position of the axis to the target position.



- ✧ Shortest distance: Firstly, the target displacement is modulated against the rotation cycle, and then the distance of clockwise and counterclockwise movement to the target position is compared, and the direction with shorter distance is taken as the direction to run to the target position.
- ✧ Current direction: Move to the target position according to the latest movement direction of the axis, and move forward to the target position after the first power up.

11.3.3.3 Software Limit

It is allowed to set software limit in linear mode.

If the software limit is effective, the absolute position of the axis will be detected at all times during the operation of the axis when T-type deceleration is made from the current speed to 0 according to the set limit deceleration. If the absolute position of the axis exceeds the limit range, the axis will execute the soft limit deceleration algorithm and interrupt the positioning or velocity instruction that is being executed.

Software limit is invalid in homing and moment mode.

11.3.3.4 Axis Error Deceleration

If the axis must switch to the errorstop state due to the logic failure of the motion instruction itself during the operation of the axis, the axis will trigger an emergency stop, and the axis will enter the errorstop state only after the emergency stop is completed.

11.3.4 Axis Speed Setting

It is allowed to set three parameters: maximum speed, maximum acceleration and maximum jogging speed. When the target speed, acceleration deceleration and other parameters in the positioning or velocity instructions exceed the speed limit, the axis runs with the values of maximum speed, maximum acceleration and maximum jogging speed.

In the fieldbus servo axis, the maximum speed is converted into pulse unit by unit conversion, which is written into the object dictionary 0x607f of the servo driver by starting parameters.

11.3.5 Probe Setting

The local pulse axis can enable the probe terminal through probe settings.

Each local pulse axis can be configured with up to two probe terminals. The probe terminal source can be selected from X0 to X17. After the probe terminal is enabled, the local pulse axis can use the probe instruction and interrupt fixed-length instruction.

11.3.6 Output Settings

The local pulse axis allows Y0/Y1, Y2/Y3, Y4/Y5, Y6/Y7 to be set as 4 local pulse axes. The local pulse axis allows output pulses in pulse + direction, forward and reverse direction and CW/CCW format.

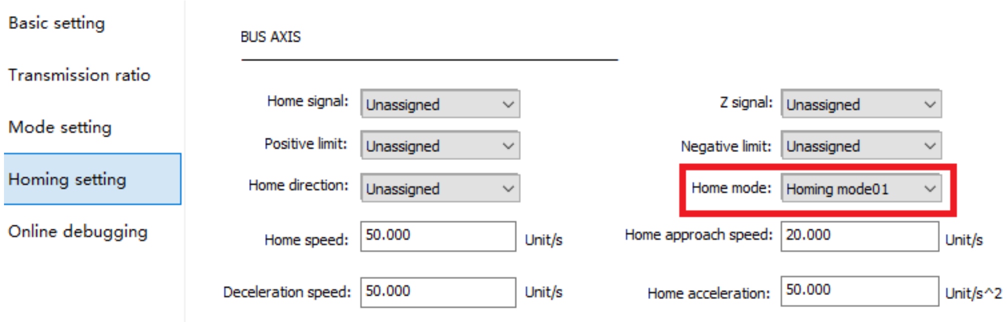
For the channel that has been set as the pulse axis, when pulse + direction is selected, Y0, Y2, Y4 and Y6 are pulse terminals, and Y1, Y3, Y5 and Y7 are direction terminals. When CW/CCW is selected, Y0, Y2, Y4 and Y6 are CW pulse terminals, and Y1, Y3, Y5 and Y7 are CCW terminals.

11.3.7 Home Setting

Homing is divided into fieldbus servo homing and pulse homing.

- Fieldbus servo homing

Fieldbus servo homing interface:



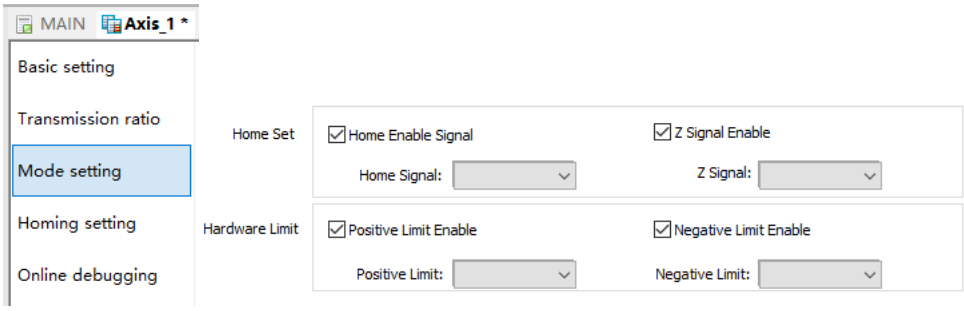
There are 11 supported homing modes, which are homing modes 01, 02, 03, 04, 17, 18, 19, 20, 33, 34 and 35.

For fieldbus homing setting, only the homing mode parameters are valid, and other interface parameters are invalid. The way to return to zero is determined by the homing mode, and the homing speed is determined by the relevant parameters of servo driver.

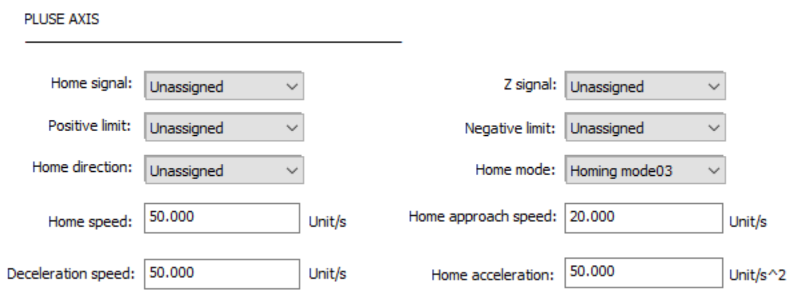
- Pulse homing

1. Home signal configuration

   Choose the mode setting option for Axis 1 and select the corresponding signal as shown below



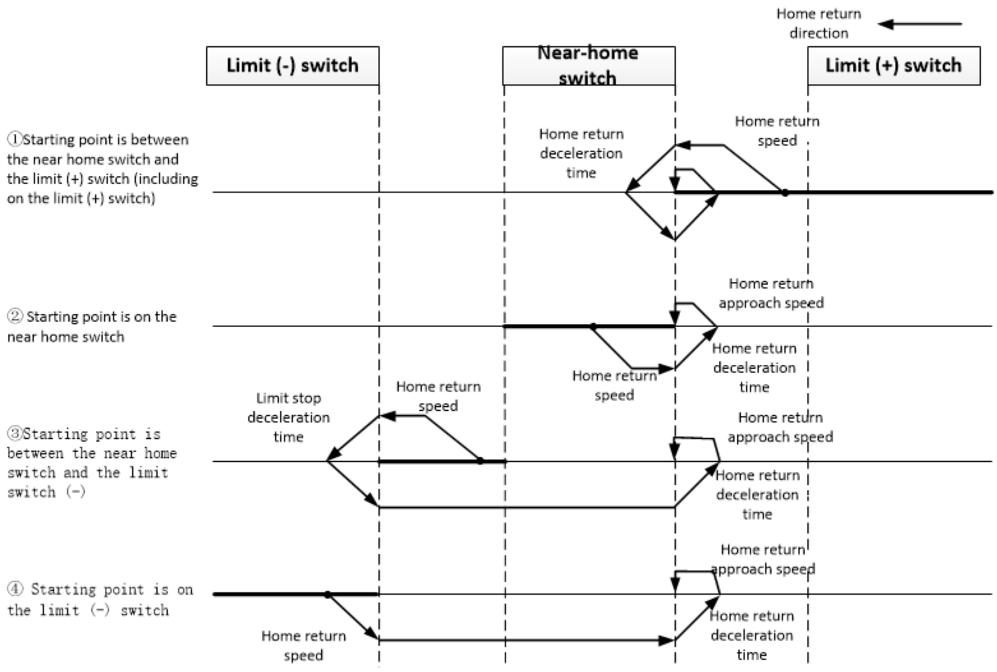
2. Pulse homing interface



The valid parameters in the current version of pulse homing interface: homing direction, homing mode, homing velocity, homing approach velocity, homing acceleration, etc. Among them, home signal, Z signal, positive limit, negative limit, etc. are used to select homing mode diagram. This function is not developed in the current version, and the decelerated running speed is invalid in the current version.

3. Schematic Diagram of Homing Mode 3

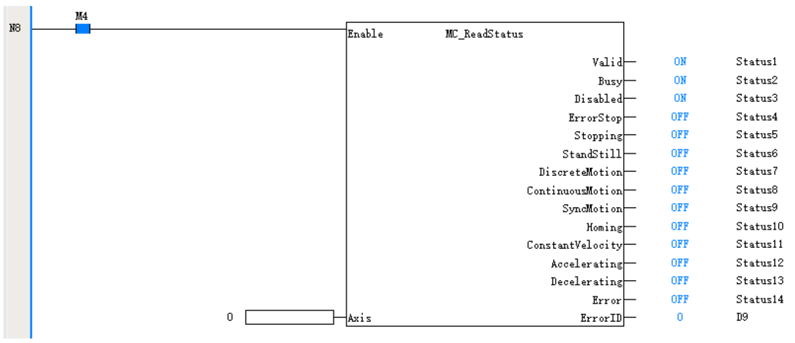
The rising edge of the near-origin switch is detected and used as the home.



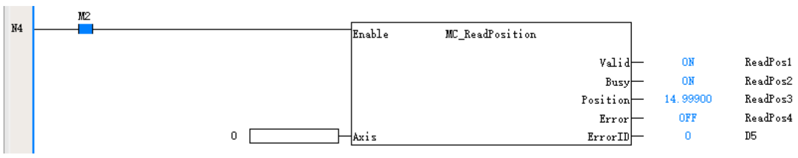
11.4 Online Monitoring

1. The axis state is obtained by means of instructions, such as reading the relevant information of the axis through state instructions like MC_ReadStatus, MC_ReadPosition and MC_ReadVelocity.

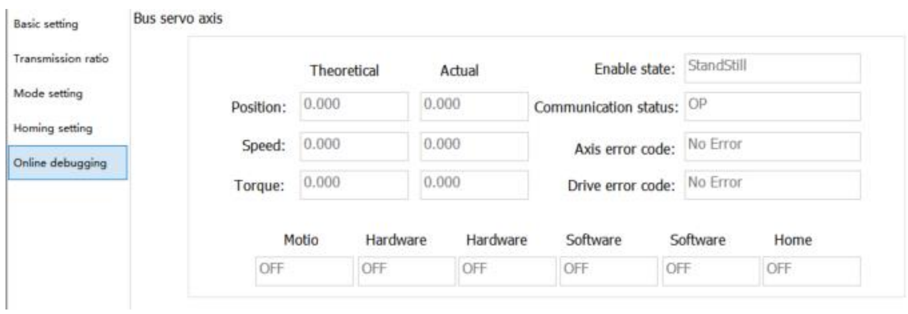
- MC_ReadStatus Read Axis Status



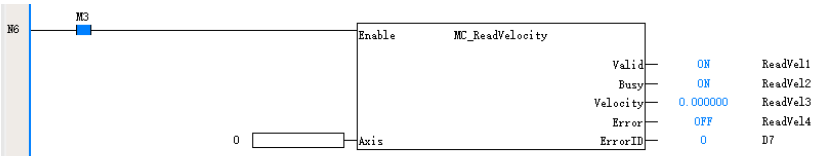
- MC_ReadPosition Read Axis Position



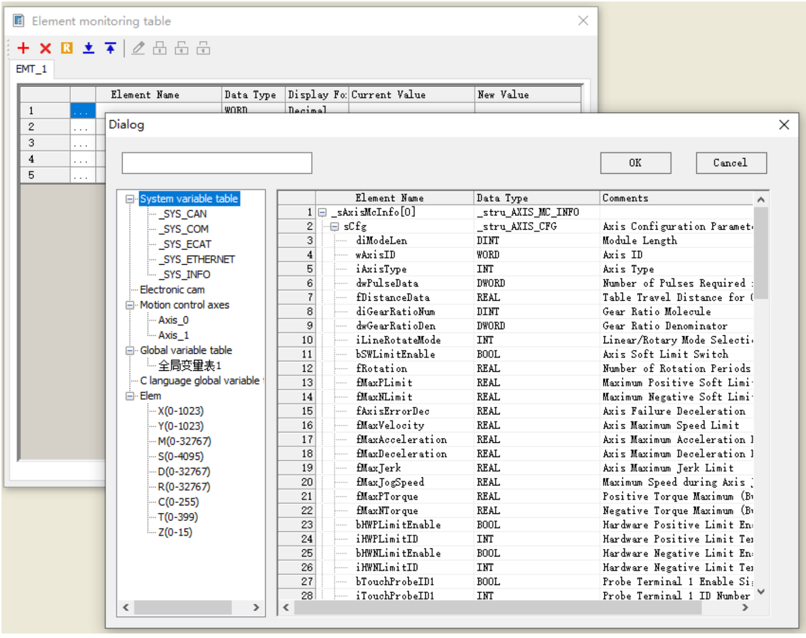
- MC_ReadVelocity Read Axis Speed



2. Acquire axis status through online debugging interface



3. Acquire axis status through axis monitoring



11.5 Axis Control Function

11.5.1 Overview

Some basic servo controls can be realized through online debugging function, such as Enable, Stop, Jog, Point Control and other functions. After confirming that the basic operation is normal, complex logic control can be realized through motion control instructions. Online debugging and PLC instruction control cannot be used at the same time, and the restrictions are as follows:

Calling MC_Stop instruction in a PLC program makes it impossible to enter online debugging mode through background when the axis is in Stop state.

The relationship between MC_Power instruction and the enable in online debugging is OR, that is, the axis can be enabled as long as one of the modes is valid.

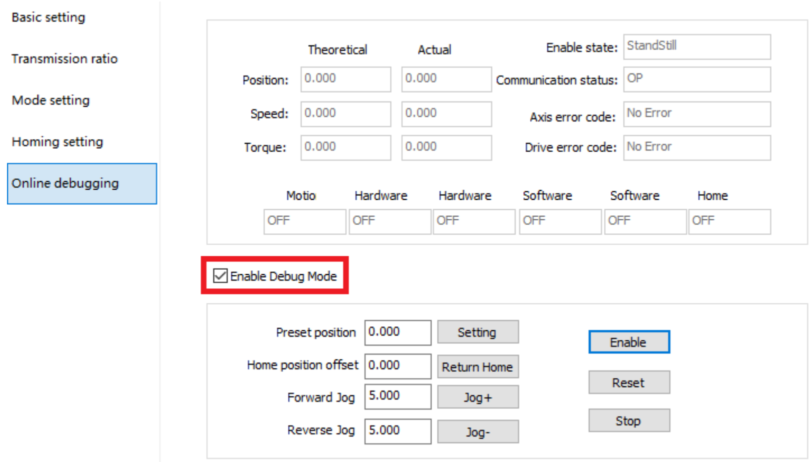
The priority of motion instructions such as MC_MoveAbsolute is lower than that of online debugging. When the axis is in online debugging mode, calling motion instructions is invalid, and the instructions report errors, but the axis will not enter an Error state.

11.5.2 Online Debugging

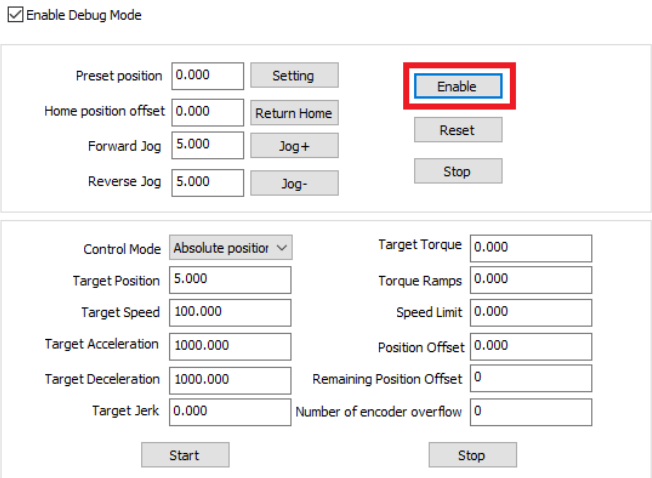
The online debugging interface is as shown below, and the supported functions include homing, JOG movement, absolute positioning, relative positioning, speed control and so on.

The online debugging operating steps are as follows:

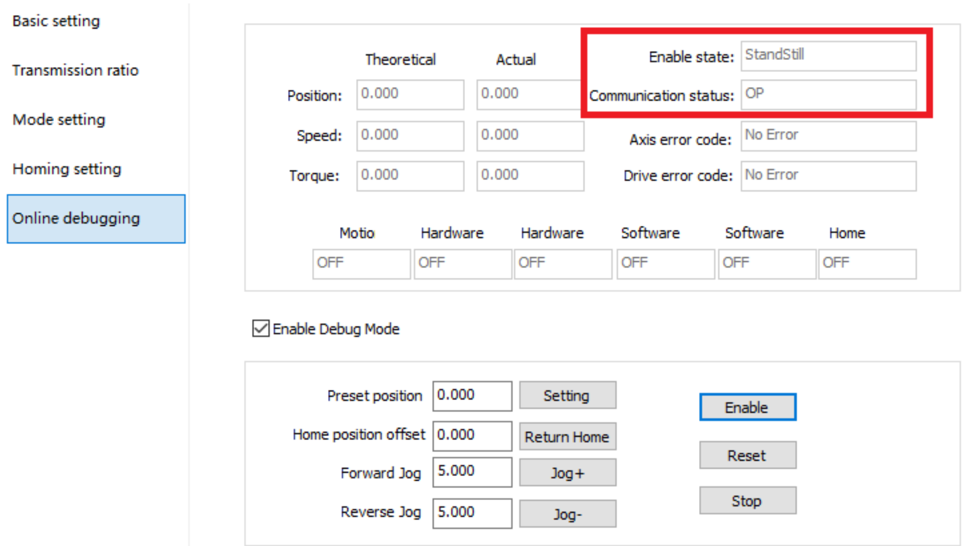
Step 1 Check "Enable Debug Mode" first.



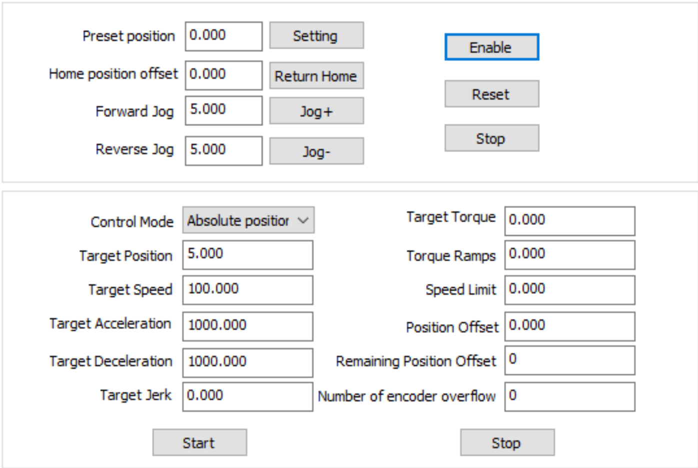
Step 2 Trigger "Enable" and observe whether the axis is in Standstill state.



Step 3 Observe axis state.



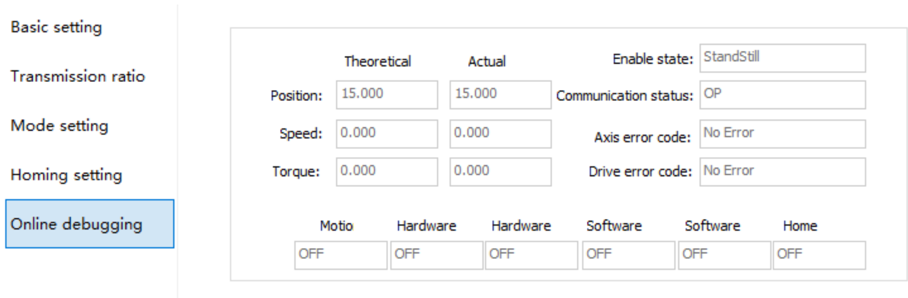
Step 4 Trigger corresponding control instructions, such as homing, JOG, absolute positioning, relative positioning, etc.



The options in the figure are described as follows:

Name	Description
Enable	It means calling MC_Power instruction for axis enable action
Reset	It means calling MC_Reset instruction for axis enable action
Stop	It means calling MC_Stop instruction for axis stop action
Settings	It means calling MC_SetPosition instruction for position setting
Homing	It means calling MC_Home instruction for homing
JOG+	JOG positive
JOG-	JOG negative
Absolute position	It means calling MC_MoveAbsolute instruction for absolute positioning action of axis
Relative position	It means calling MC_MoveRelative instruction for relative positioning action of axis
Speed mode	It means calling MC_MoveVelocity instruction to run the axis in speed mode

Step 5 Observe the change of axis position and axis speed



11.5.3 Instruction Control Rule

In PLC, the axis motion can be controlled by instructions, and the rules for calling instructions are as follows:

- Instructions do not need to be instantiated.
- In the instruction, the axis number is the unique identification to access the axis.
- The priority of motion instructions is generally lower than that of online debugging mode.
- Floating-point parameters in instructions need to meet the precision range of floating-point numbers, which are generally considered to be 7 significant digits, and can be set to 9999999 at maximum.

11.5.4 Limit Processing

It supports two kinds of limit detection: software limit detection and hardware limit detection.

- Software limit processing is only valid when calling position and velocity instructions in linear mode, but homing and torque instructions are invalid.
- Within the software limit range, call the position instructions which can be executed normally if the absolute target position does not exceed the limit; If the absolute target position exceeds the software limit, the execution of the positioning instruction is interrupted and finally stopped at the software limit.
- Within the software limit range, call the velocity instructions whose execution will be interrupted and which will stop at the software limit when the axis runs at the current speed and detects that it will exceed the soft limit.

11.5.5 Positioning Acceleration and Deceleration Curve

It supports T-type acceleration and deceleration and S-curve acceleration and deceleration, which are determined by Jerk parameter in the instruction. When Jerk parameter is 0, it indicates T-type acceleration and deceleration, and when it is greater than 0, it indicates S-curve acceleration and deceleration.

11.6 Fault Type

Axis faults are divided into instruction faults, axis faults and drive faults.

- An instruction fault is the fault caused by MC axis control instruction itself

For example, the instruction parameters are unreasonable, and the PLCOpen state machine of the axis changes during operation, which leads to instruction error. You can get the fault code by checking the ErrorID of the faulty instruction.

- An axis fault is the fault reported by the axis itself

For example, the following error is too large. It can be obtained in the background through "Axis error" in the "Online debugging" interface or through the AxisErrorID in the MC_ReadAxisError instruction.

- A driver failure is the fault of EtherCAT fieldbus driver or local pulse output axis

To get the fault of the EtherCAT fieldbus driver, 0x603F must be configured in the PDO mapping and associated with the axis. It can be obtained in the background through "Axis error" in the "Online debugging" interface or through the ServoErrorID in the MC_ReadAxisError instruction.

12 High-speed Counter

12.1 Brief Introduction of High-speed Counter Axis

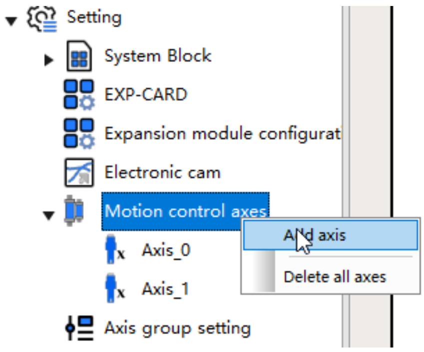
In AutoStation Pro, counters are managed in the form of encoder axis. After the counter is associated with the axis, they are collectively called counter axis or encoder axis. A total of 8-axis 32-bit high-speed counters are supported.

- Counters 0 – 3 are high speed counters, which can realize AB phase 1/2/4 frequency doubling, CW/CCW, pulse + direction and single phase counting mode.
- 4–7 are ordinary counters, which only have a single-phase counting mode.
- The 8-axis counting signal source can be chosen from external pulse input or internal 1ms/1μs clock counting; With other input signals, the preset and latch functions of the counter can be realized.

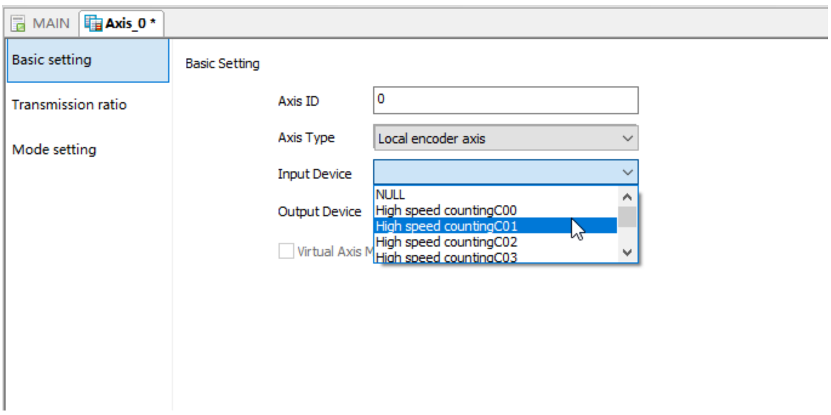
12.2 Create a Counter Axis

Before using counters in AutoStation Pro, you need to associate them with axes.

Step 1 In the "Project manger" field, right-click the motion control axis under "Setting" and choose "Add axis" to create a motion control axis.



Step 2 Double-click the newly added axis (as shown in Figure Axis_0) to open the settings page, choose "Local encoder axis" as the axis type in the "Basic Setting" interface, and choose "High speed counter" as the input device to associate the axis with the counter. The axis number is used as the axis identification in the program to realize the control of the corresponding counter axis. (High-speed counters have multiple modes, while ordinary counters only have a single-phase counting mode)



12.3 Counter Axis Unit and Conversion

High-speed counters use pulse unit when decoding encoder signals, while counter instructions use common measurement units such as mm, °, inch, etc., which are called Unit. The number of pulses can be converted into unit by conversion, which can be defined as device-specific units (cm, mm, rpm, etc.) according to the actual application.

The parameters to be set for unit conversion are as follows:

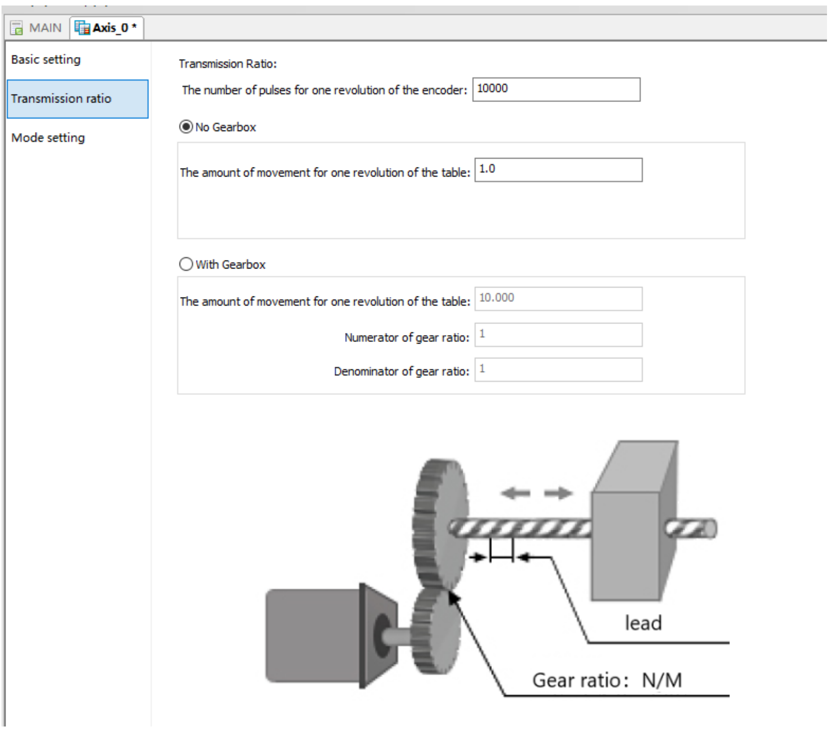
Name	Function
Number of pulses for one revolution of motor/encoder	According to the encoder resolution, set the number of pulses for one revolution of motor
Is a speed change device used	Specify whether a speed change device is used
The amount of movement when the motor/encoder rotates for one revolution	The amount of movement of workpiece when the motor rotates for 1 revolution without the speed change device
The amount of movement when the worktable rotates for one revolution	The amount of movement of workpiece when the worktable rotates for 1 revolution with the speed change device
Numerator of gear ratio	Set gear ratio on the workpiece side
Denominator of gear ratio	Set gear ratio on the motor side

For example, the servo motor drives the worktable to move by connecting screw rod of reducer, and counts encoder to feedback worktable position through the PLC controller. The counter counts encoder pulse, taking pulse as unit; The counter axis indicates the worktable position, in millimeters. Therefore, in the program, Unit is used as the unit of counter axis.

The conversion relationship between Unit and pulse is as follows:

- Without a speed change device

10,000 pulses equal to the operation of 1 Unit (cm/mm/rpm), to be decided by the user as per the actual application).



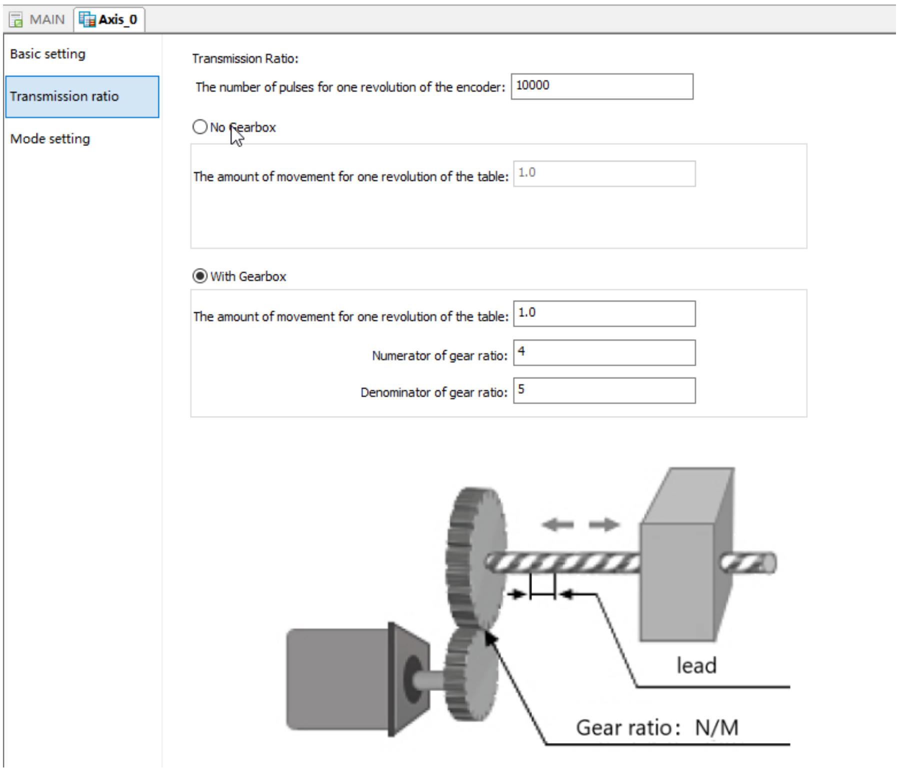
Calculation formula of pulse and actual distance:

Number of pulses (pulse)

$$= \frac{\text{The number of pulses for one revolution of } \frac{\text{motor}}{\text{encoder}} \text{ [DINT]}}{\text{Amount of movement when the worktable rotates for one revolution [REAL]}} \times \text{Moving distance (Uint)}$$

- With variable device

10,000 pulses, gear ratio 4:5 (cm/mm/r, to be decided by the user as per the actual configuration).



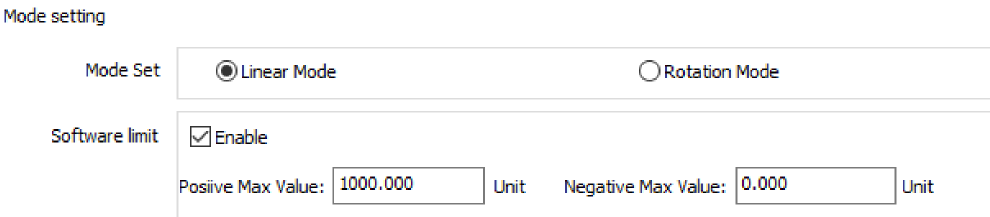
12.4 Set the Working Mode

12.4.1 Linear Mode

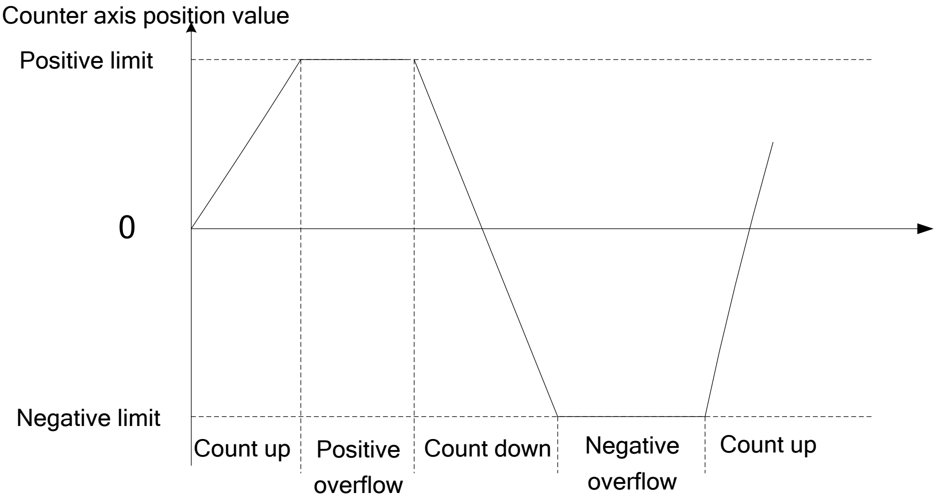
The position of the counter axis changes between the negative limit value and the positive limit value, and after the position of the counter axis reaches the limit value, the homodromous pulse is continuously input; The counter axis reports overflow while the counter axis position remains unchanged. After the counter axis reports overflow, input reverse pulse, the counter axis counts reversely, and the overflow flag is removed.

In linear mode, you can set the negative and positive position limit values of the counter axis in the interface, and Unit is used as the position unit. The negative limit value must be smaller than the positive limit value.

Since the high-speed counter is a 32-bit counter, the negative limit value and the positive limit value must be in the range of 32-bit integers after being converted into pulse unit [-2147483648, 2147483647].

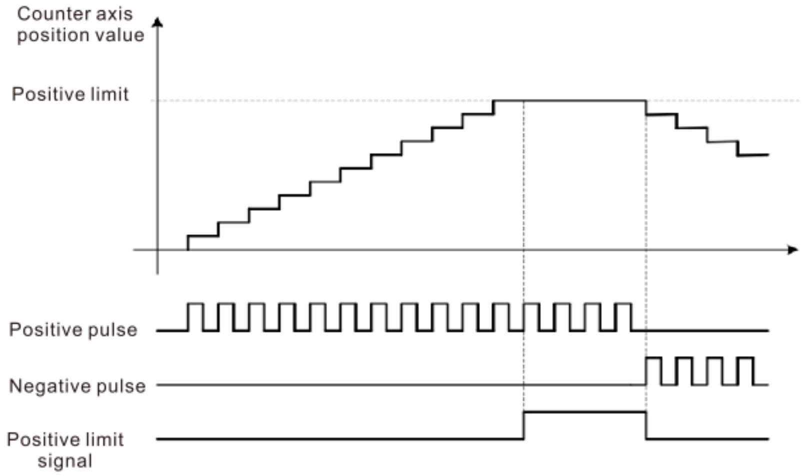


In linear mode, the high-speed counter operates in a closed interval of [negative limit value, positive limit value]. When the direction is negative, the count value decreases in the negative direction, and after reaching the negative limit value, the count value does not decrease any more; When the direction is positive, the count value increases in the positive direction, and after reaching the positive limit value, the count value does not increase.



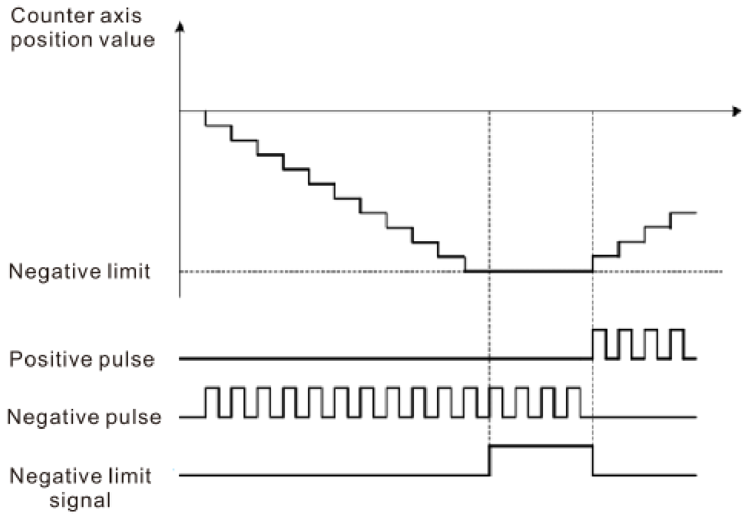
12.4.1.1 Positive Pulse Counting

In linear mode, input positive pulse, after the counter axis position counts up to the limit value, continue to input positive pulse, set the counter axis positive limit signal to 1, the counter axis position value remains unchanged. Input negative pulse, the counter axis position counts down, and set the positive limit signal to 0.



12.4.1.2 Negative Pulse Counting

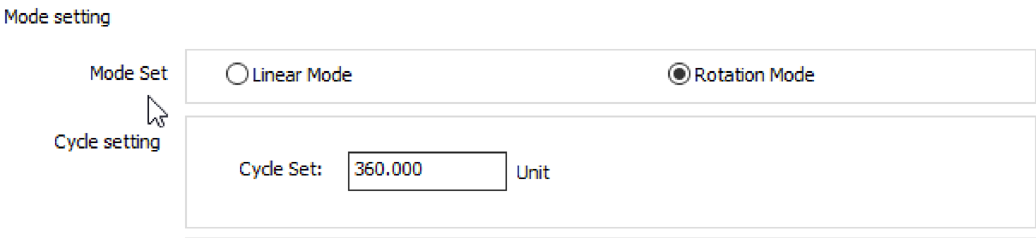
In linear mode, input negative pulse, after the counter axis position decremental counter reaches the limit value, continue to input negative pulse, set the counter axis negative limit signal to 1, the counter axis position value remains unchanged. Input positive pulse, the counter axis position counts up, and set the negative limit signal to 0.

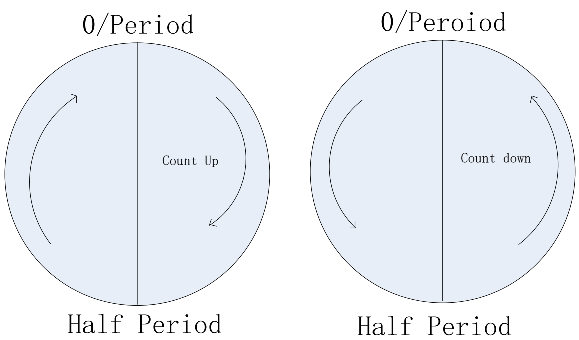


12.4.2 Rotation mode

The position of the counter axis changes cyclically in the rotation cycle. When the counter counts up, the position of the counter axis reaches the maximum value of the rotation cycle and then becomes 0. When the counter counts down, the position of the counter axis is 0 and then decreases from the maximum value of the rotation cycle.

In rotation mode, you can set the rotation cycle of the counter axis in the interface, and Unit is used as the cycle unit. Since the high-speed counter is a 32-bit counter, the rotation cycle must be in the range of 32-bit integers after being converted into pulse unit [-2147483648, 2147483647].

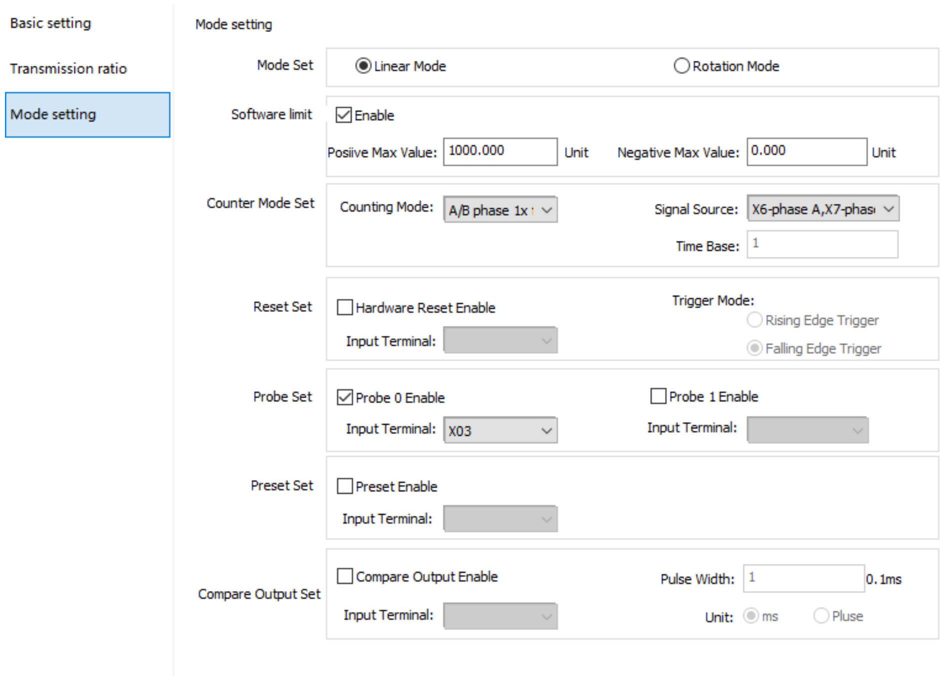




12.5 Set Counter Parameters

12.5.1 Overview

Parameter settings mainly include counting mode, reset, probe, preset and comparison output settings.



12.5.2 Counting Mode

The local encoder axis supports multiple signal counting modes, A/B phase (1/2/4 frequency doubling), CW/CCW, pulse + direction, single phase counting.

Signal source: According to different counting modes, different signal sources can be selected.

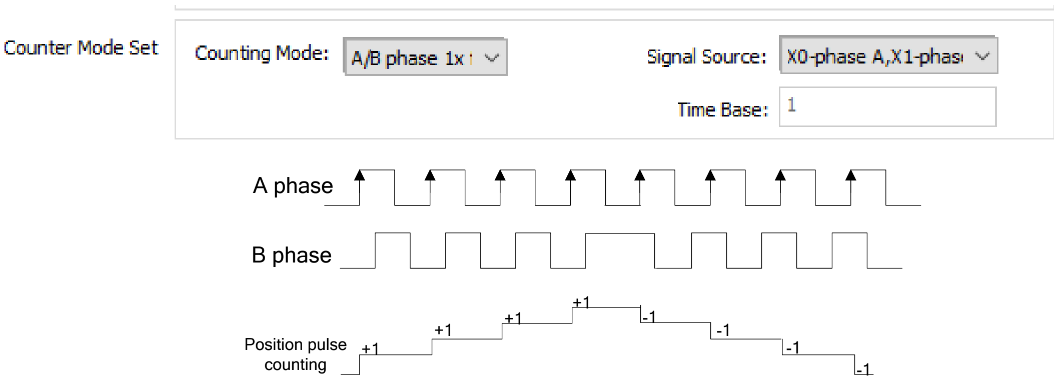
Note: High-speed counters support multiple modes, while ordinary counters only support single-phase counting mode.

12.5.2.1 A/B Phase Mode

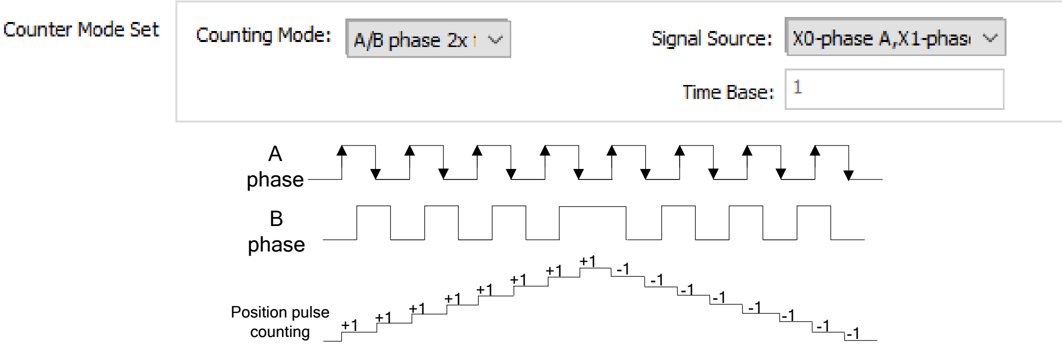
In the A/B phase mode, the encoder generates two quadrature phase pulse signals with a phase difference of 90 degrees, namely A-phase signal and B-phase signal.

- When the A-phase signal leads the B-phase signal, the counter counts up.
- When the B-phase signal leads the A-phase signal, the counter counts down.

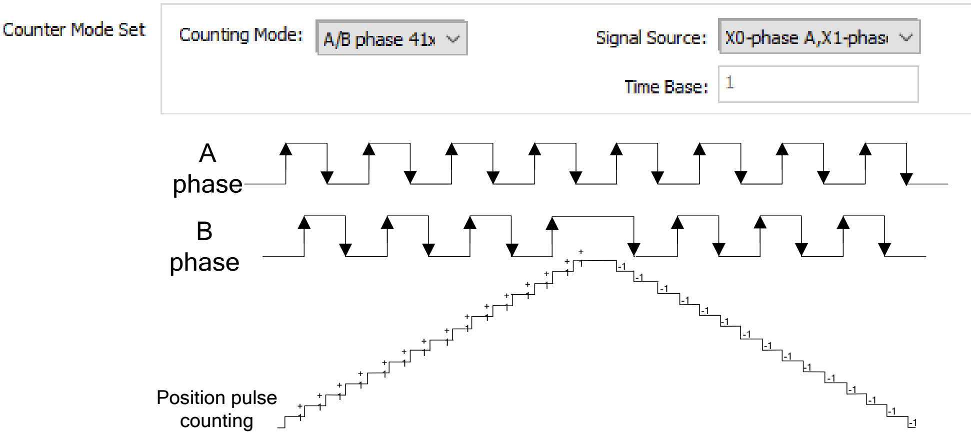
In the A/B phase 1 frequency doubling mode, only the rising edge of the A-phase pulse is counted, as shown in the following figure.



In the A/B phase 2 frequency doubling mode, only the rising/falling edge of the A-phase pulse is counted, as shown in the following figure.



In the A/B phase 4 frequency doubling mode, the rising/falling edge of the A-phase pulse and B-phase pulse is counted, as shown in the following figure.



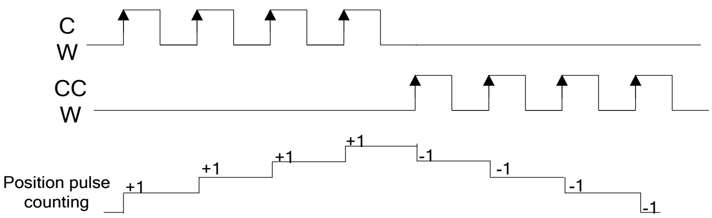
12.5.2.2 CW/CCW Mode

CW is a positive pulse signal, and CCW is a negative pulse signal.

- When the encoder rotates forward, CW outputs a pulse signal.
- When the encoder is inverted, the CCW outputs a pulse signal.

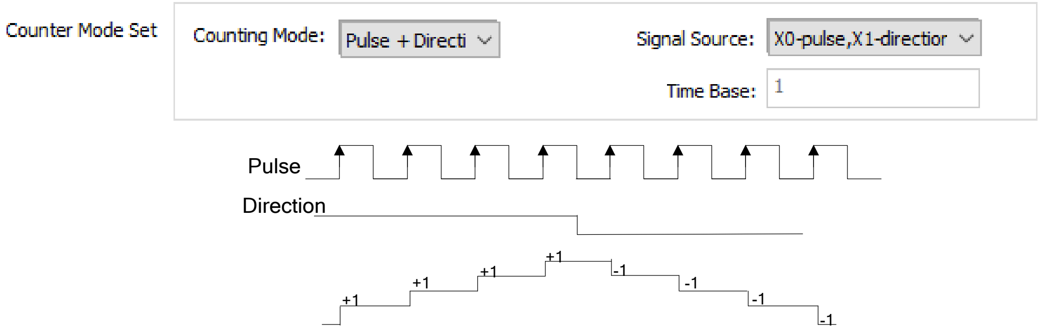
As shown in the following figure, X0 is a CW signal and X1 is a CCW signal:

In CW and CCW modes, the high-speed counter counts up the CW signal and counts down the CCW signal, as shown in the following figure.



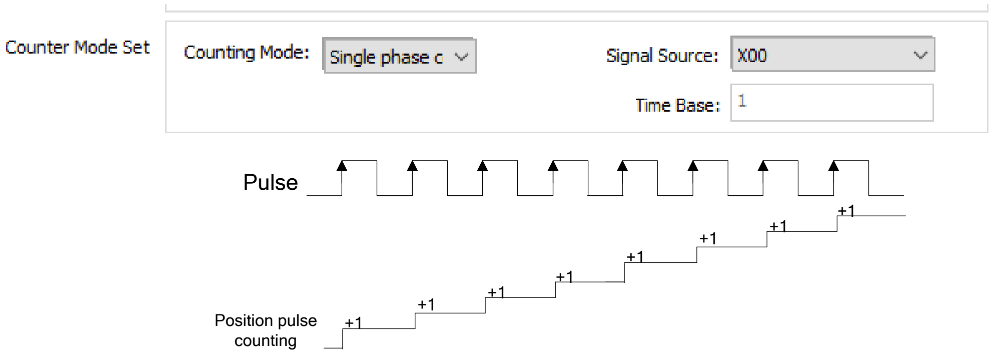
12.5.2.3 Pulse + Direction Mode

In pulse + direction mode, when the direction signal is ON, the high-speed counter counts up, and when the direction signal is OFF, the high-speed counter counts down, as shown in the following figure.



12.5.2.4 Single-phase Counting

In single-phase counting mode, the high-speed counter counts up the pulse signal, as shown in the following figure.

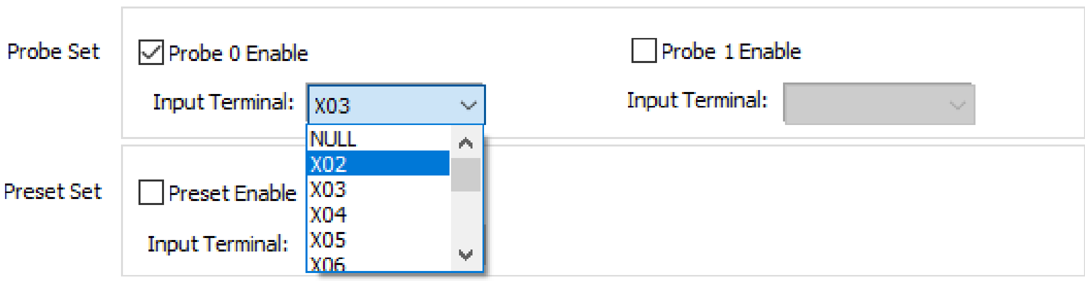


12.5.3 Hardware Reset Settings

Check hardware reset enable, configure external trigger IO (X0–X7, X10–X17), configure trigger edge mode, choose trigger by external hardware through ENC_Reset instruction, which can then reset counter count value through external IO.

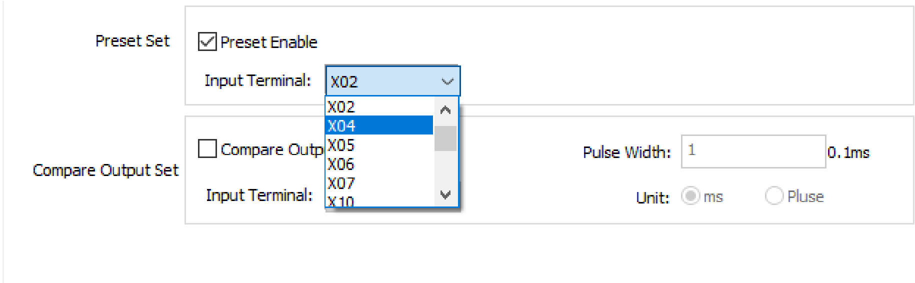
12.5.4 Probe Setting

The first four high-speed counters support 2 probes, while the last four ordinary counters do not support probe function, and the current value of the counter is latched by external hardware. Check the probe enable and choose the input terminals (X0 –X7, X10–X17). Get the corresponding latched value through the instruction ENC_TouchProbe



12.5.5 Preset Settings

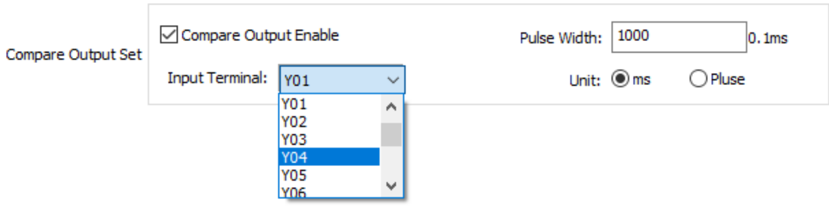
Each counter axis supports one preset function, check preset enable, configure input terminals (X0–X7, X10–X17), and choose external hardware trigger by ENC_Preset instruction, so that the preset value of the counter can be set by external hardware signal.



12.5.6 Comparison Output Settings

The comparison output function can directly output through hardware instantaneously, and after reaching the comparison value, it can respond to the output within 5 microseconds. Each counter axis supports a hardware comparison output.

Configure the output IO (Y0 – Y7, Y10 – Y17), configure the pulse hold width (16 unsigned integers), configure the hold unit 0.1 ms (hold time = hold width * 0.1 ms) or Pulse (hold time is the number of received pulses).



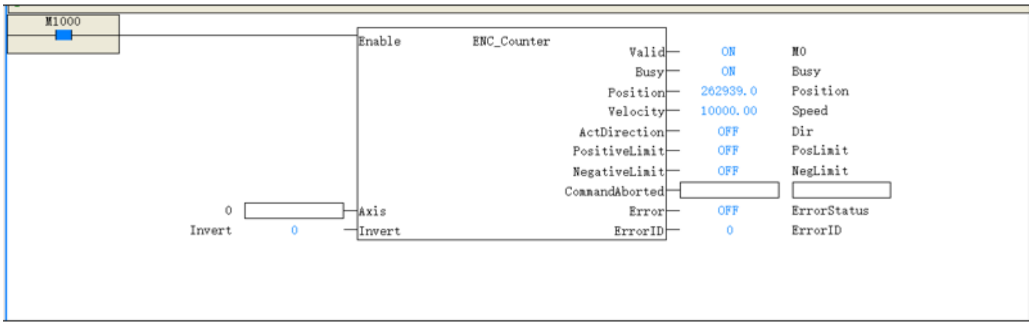
12.6 Application of Encoder Axis Instruction

12.6.1 Overview

After setting the local encoder axis in Auto Station Pro, the functions of axis position ranging, speed measurement, axis position presetting, axis position latch and axis position comparison can be realized in combination with the relevant function block applications.

12.6.2 Axis Position Ranging/Speed Measurement Instruction

1. With ENC_Counter instruction, the encoder axis position ranging and speed measurement can be carried out.
2. The position of encoder axis changes within the limited range of the encoder axis according to the mode setting, and the position unit is Unit (user-defined).
3. The encoder axis speed is the current real-time speed, and the speed unit is Unit/S. The minimum measurable speed of the encoder axis is 1Hz pulse frequency.
4. If the transmission ratio of encoder axis is set to 1:100, the real-time speed of 1Hz pulse frequency corresponds to 0.01 Unit/s.



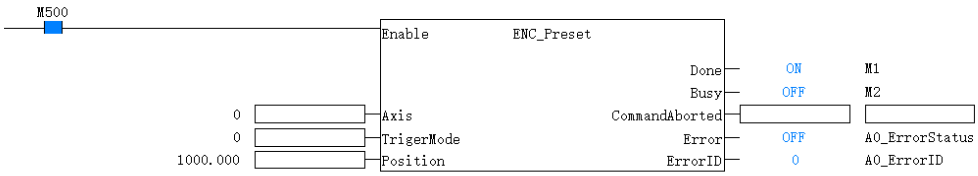
"Invert" control counting direction in the instruction, 0: count in the default direction, 1: count in the direction opposite to the actual counting direction.

The configuration parameter of the instruction is that: Enable rising edge is valid, while changing the input parameters in other hold periods is invalid. And Enable hold enables the module.

Invert	A/B Phase	Pulse + Direction	CW/CCW	Single-phase Counting
0	● A-Phase advance, B-Phase countup ● B-Phase advance, A-Phase countdown	● Direction signal low level countdown ● Direction signal high level countup	● CW countup ● CCW countdown	Countup
1	● A-phase advance, B-phase countdown ● B-Phase advance, A-Phase countup	● Direction signal low level countup ● Direction signal high level countdown	● CW countdown ● CCW countup	Countdown

12.6.3 Axis Position Preset Instruction

Enable ENC_Preset to assign the encoder axis position according to the preset mode.

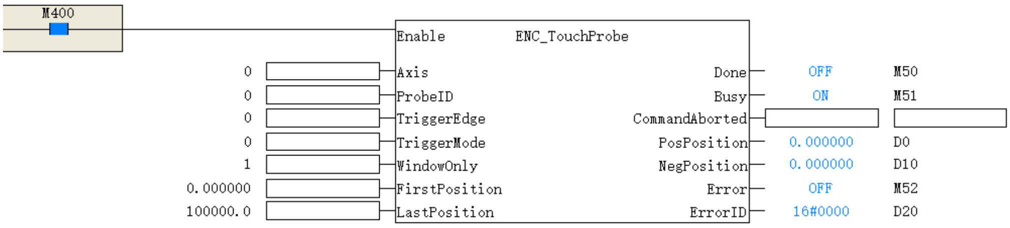


"TriggerMode" as a preset condition can choose rising edge trigger, falling edge trigger, or rising or falling edge trigger.

TriggerMode	Definition
0	Software control, Enable rising edge trigger
1	External hardware rising edge trigger (preset function need to be configured on the system side)
2	External hardware falling edge trigger (preset functions need to be configured on the system side)
3	External hardware rising or falling edge trigger (preset functions need to be configured on the system side)

12.6.4 Probe Instruction

Using the EN_TouchProbe function block instruction, the position value of the encoder axis can be latched when the corresponding external input signal reaches the trigger condition. Each encoder axis supports 2 probes, so it is necessary to check the configuration-related probe functions in the system configuration interface. (See section 12.5.4 Probe Setting for details.)



The parameter "ProbeID" indicates the probe number used by the encoder axis as follows:

ProbeID	Definition
0	It indicates the use of probe 1
1	It indicates the use of probe 2

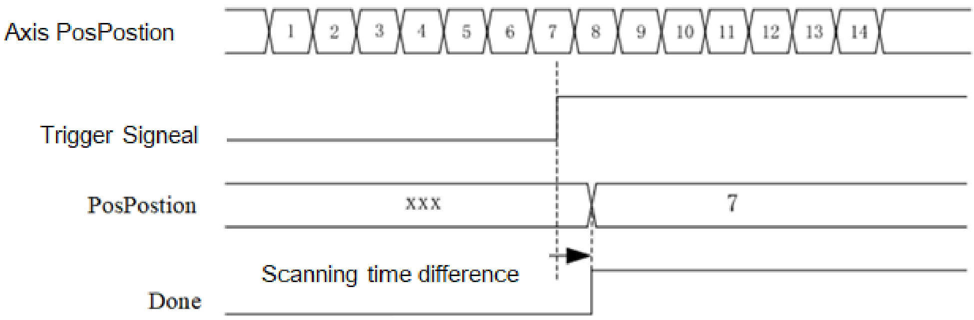
The parameter "TriggerEdge" indicates the probe trigger edge setting, the rising edge trigger position is latched in PosPosition, and the falling edge trigger position is latched in NegPosition.

TiggerEdge	Definition
0	External hardware signal rising edge trigger
1	External hardware signal falling edge trigger
2	External hardware signal rising or falling edge trigger

The parameter "TiggerMode" sets the single-trigger or continuous-trigger modes.

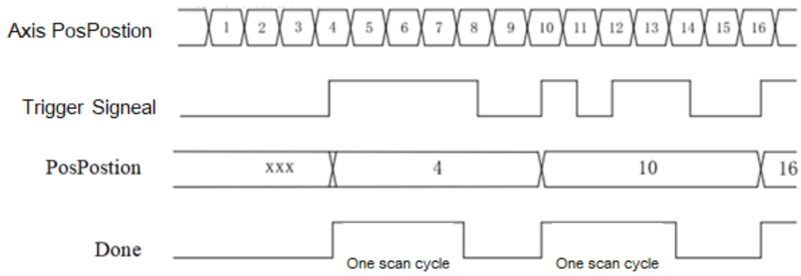
TiggerMode	Definition
0	Single trigger
1	Continuous trigger

In single trigger mode where rising edge trigger is adopted, function block instruction Enable is active, and when the external input signal reaches the trigger condition, the encoder axis position will be latched once, and the completion signal is output.



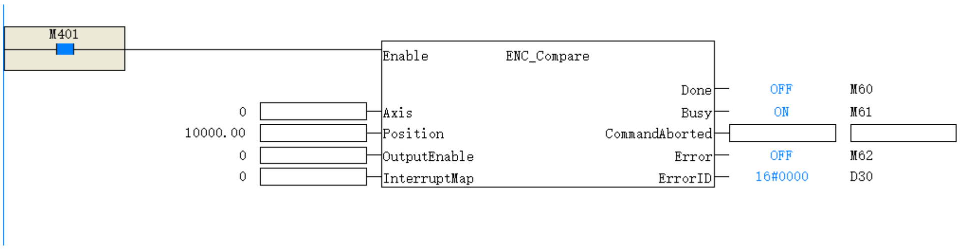
Rising edge single trigger mode

In continuous trigger mode where rising edge trigger is adopted, function block instruction Enable is active, and when the external input signal reaches the trigger condition, the encoder axis position will be latched, and the completion signal with effective time of 1 scanning period will be output. After the completion signal becomes OFF, the external input signal reaches the trigger condition again, and the counter axis position will continue to be latched, and the completion signal with effective time of 1 scanning period will be output. If the latch condition continues to be triggered within 1 scanning period when the completion signal is valid, the counter axis position will not be latched at this time.



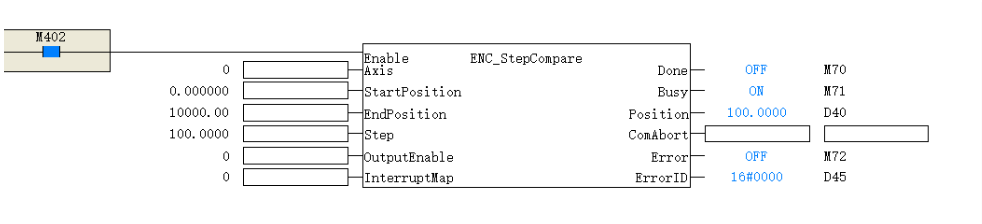
12.6.5 Single-step Comparison Instruction

Instruction ENC_Compare can compare the encoder axis with a single position, and when the encoder axis reached the comparison position, the completion signal is output.

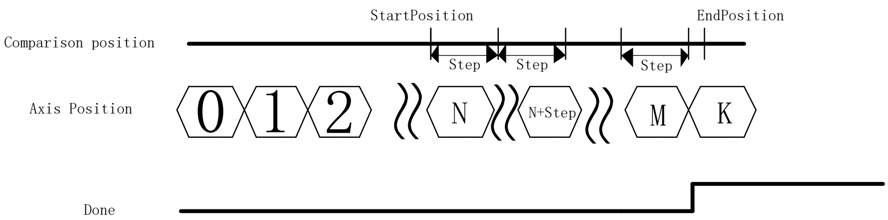


12.6.6 Continuous Comparison Instruction

Instruction ENC_StepCompare can compare the encoder axis with unit distances continuously. When the comparison is completed, the completion signal is output.

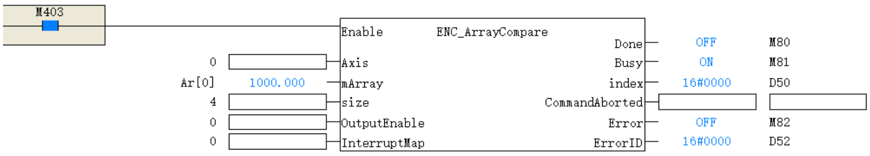


- When StartPosition <EndPosition，Step> 0, the encoder axis position is compared from StartPosition. When the comparison value is equal, the comparison position automatically increases the Step distance to start the next comparison, until the comparison position is greater than EndPosition, then the comparison is completed, and the completion signal is output. The current comparison value keeps the last comparison parameter.
- When StartPosition > EndPosition，Step < 0, the encoder axis position is compared from StartPosition. When the comparison value is equal, the comparison position automatically increases the Step distance to start the next comparison, until the comparison position is greater than EndPosition, then the comparison is completed, and the completion signal is output. The current comparison value keeps the last comparison parameter.

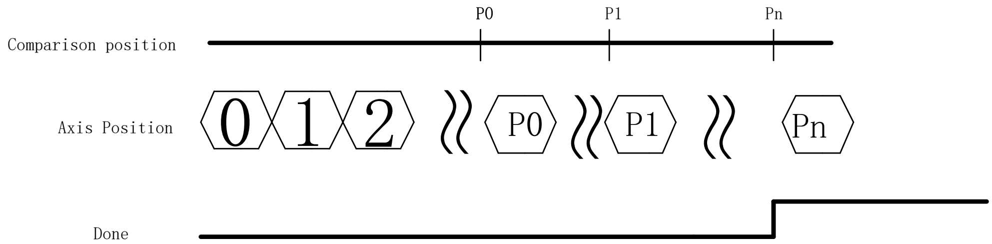


12.6.7 Array Comparison Instruction

Enc_ArrayCompare can realize continuous comparison between encoder axis and array at multiple positions. When Enable is valid, the encoder axis position starts comparison from the first position of the array, and after the comparison result is equal, the next position value of the array will be compared. Until the last comparison position is compared, a completion signal is output.



- "mArray" in the instruction sets the array to be compared. For example, to define the array Ar[10]; If the comparison starts from Array 0, fill in Ar[0], and if the comparison starts from Array 2, fill in Ar[2].
- In the instruction, "Size" sets the array length (the maximum value is 100). After the comparison of all array positions set by the array length is completed, the completion signal is continuously output and the continuous comparison of multiple positions is completed.
- The output parameter "Index" represents the number of completed comparisons.



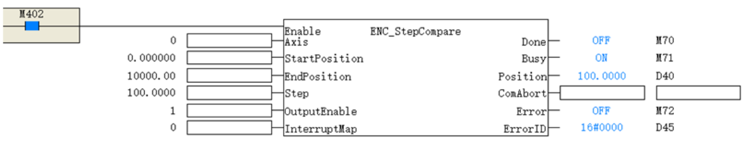
12.6.8 Hardware Comparison Output of Encoder Axis

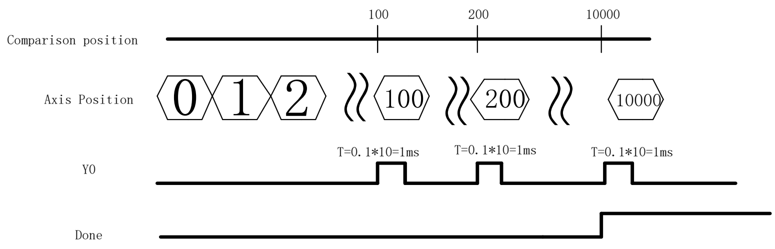
The comparison instructions Enc_Compare, Enc_StepCompare and Enc_ArrayCompare all have hardware comparison output function. Setting OutputEnable to 1 can output the corresponding hardware associated when the comparison results are equal.

Take Enc_StepCompare as an example:

Step 1 Configure the relevant configuration of the encoder axis hardware comparison output in the system configuration.

Step 2 In Enc_StepCompare, the hardware comparison output function is enabled, and OutputEnable is set to 1.



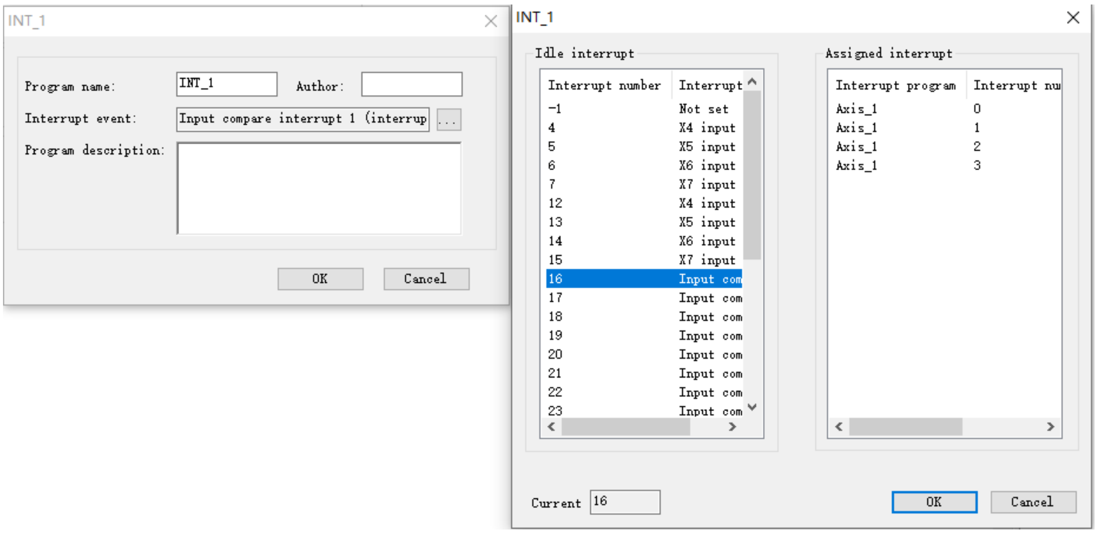


12.6.9 Encoder Axis Comparison Interrupt

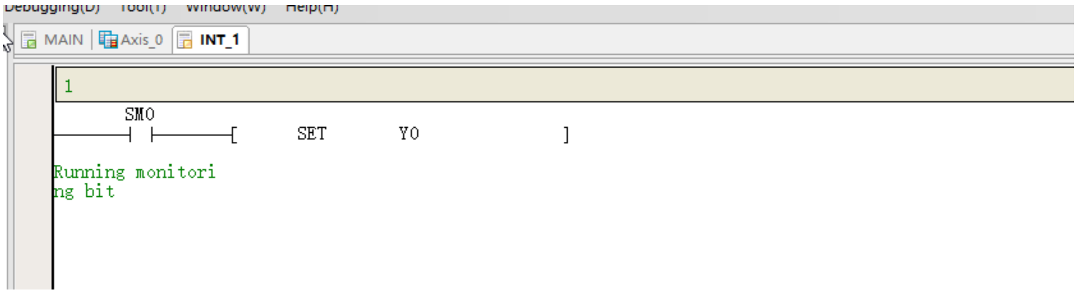
Enc_Compare, Enc_StepCompare and Enc_ArrayCompare all have interrupt service function. Set Interrupt to 1 – 16 (0: Not used), execute the corresponding interrupt service function.

Take Enc_Compare as an example:

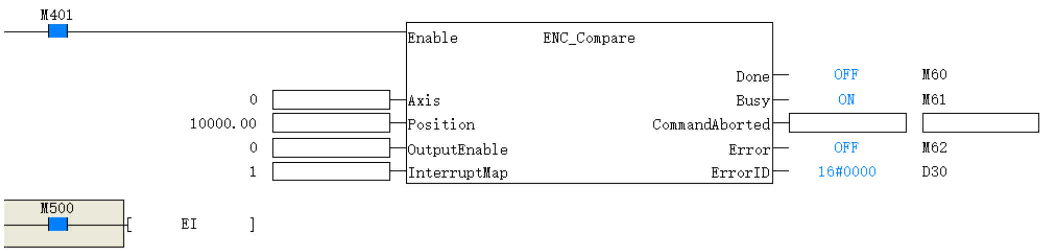
Step 1 Configure the interrupt service function (INT_1) in the system configuration.



Step 2 Write interrupt service function (INT_1).



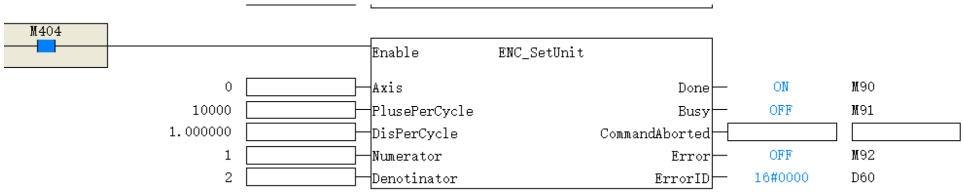
Step 3 Configure the corresponding Interrupt number for "Interrupt" in the instruction ENC_Compare and start IE.



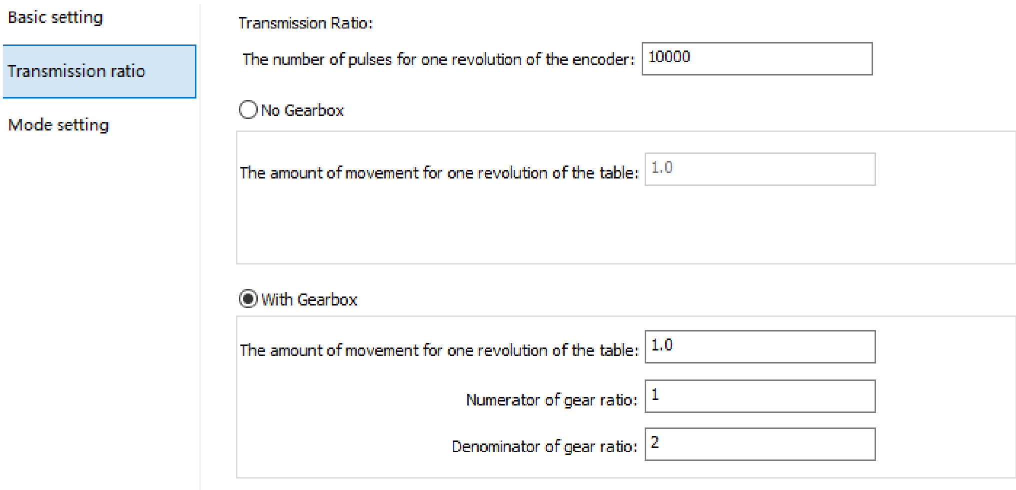
Step 4 When the encoder position value is equal to the comparison value, the interrupt service function INT_1 is triggered to light up Y0.

12.6.10 Modify the Gear Ratio of Encoder Axis

The instruction "Enc_SetUnit" can modify the electronic gear configuration of the axis in the system configuration parameters. Activate Enable, configure relevant parameters (PlusePerCycle: number of pulses per revolution of motor, DisPerCycle: actual distance per revolution, Numerator: the numerator of gear ratio, Denominator: the denominator of gear ratio).



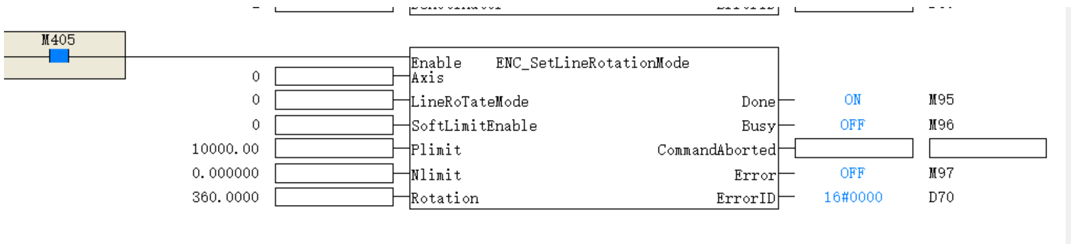
Corresponding system configuration parameters:



Note: This instruction directly modifies the system configuration parameters. When fixed prior to application, it is necessary to close other application instructions of the encoder axis. After closing, the system parameters can be modified.

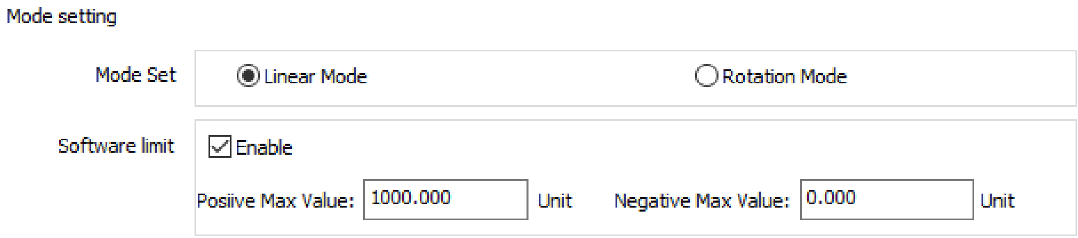
12.6.11 Modify Mode/Limit Value of Encoder Axis

The instruction "ENC_SetLineRotationMode" can modify the configuration associated with the mode in the system configuration parameters. Enter the relevant configuration, and activate Enable control terminal.



Parameter	Definition
LingRotateMode	0: Linear mode 1: Selection mode
SoftLimitEnable	0: Limit function disabled 1: Limit function enabled
Plimit	Positive limit
Nlimit	Negative limit
Rotation	Periodic value

Corresponding system configuration parameters:



Note: This instruction directly modifies the system configuration parameters. When fixed prior to application, it is necessary to close other application instructions of the encoder axis. After closing, the system parameters can be modified.

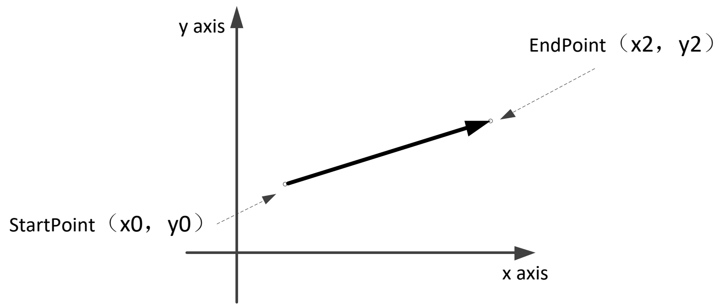
13 Interpolation Function

13.1 Brief Introduction of Interpolation Function

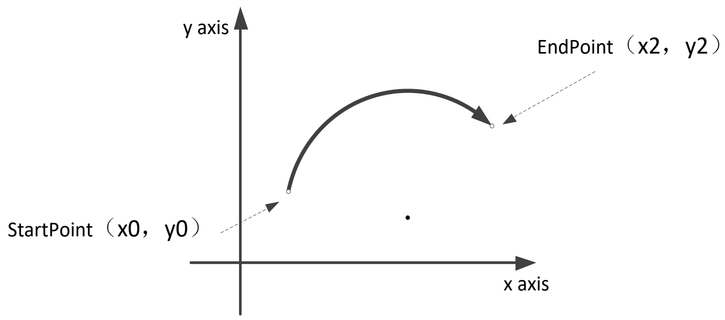
13.1.1 Basic Introduction

Interpolation adopts space rectangular coordinate system, supports linear interpolation and plane arc interpolation, and the interpolation function is realized by means of axis groups. Up to 8 axis groups can be added at the same time, and each axis group can control up to 4 motion control axes (fieldbus servo axis or local pulse axis), including X, Y and Z coordinate axes and an auxiliary axis.

When linear interpolation is adopted, the motion control axes of x, y and z coordinate axes move along the coordinate axes, and the auxiliary axis moves along the straight line from the starting point to the end point. The following schematic diagram illustrates two-axis linear interpolation:



When arc interpolation is adopted, one of XY axis plane, YZ axis plane and XZ axis plane can be selected for arc interpolation. If other axes are configured in the axis group, the other axes will not act. The following schematic diagram illustrates XY axis plane arc interpolation:



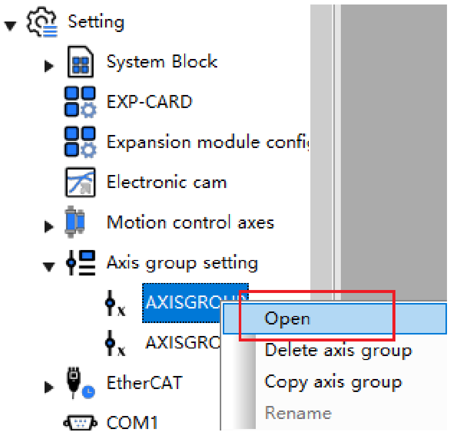
13.1.2 Instruction List

The instruction list is as follows, and the specific parameters of the instruction are detailed in the Instruction Manual.

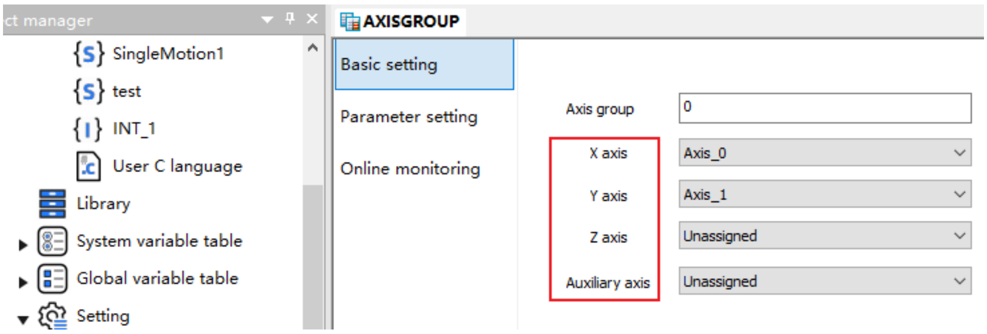
Instruction	Chinese Instruction Name
MC_MoveLinear	Linear interpolation
MC_Movecircuar2D	Plane arc interpolation
MC_GroupStop	Interpolation stop
MC_GroupImmediateStop	Interpolation immediate stop
MC_GroupHalt	Interpolation halt
MC_GroupSetOverRide	Interpolation on-line speed regulation

13.1.3 Upper Computer Axis Group Configuration Interface Configuration

Step 1 In the PLC project, choose "Setting" > "AXISGROUP", right-click on "AXISGROUP" to select to add an axis group, and then right-click on the axis group to open it, as shown in the following figure.



Step 2 On the "Basic setting" page of the axis group, configure the corresponding axis:

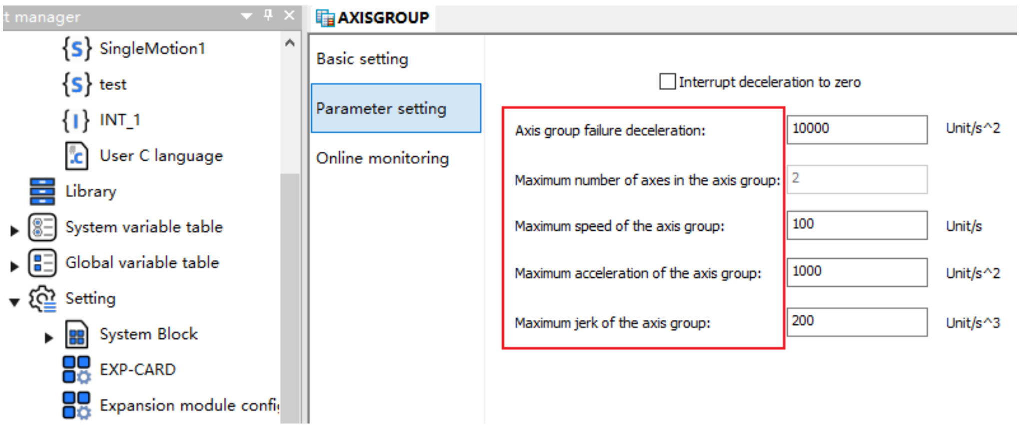


Step 3 Configure the parameters for running in the "Parameter setting" field:

- The "Interrupt deceleration to zero" option is an interpolation option when interpolation interrupts and reverse motion occurs after interruption.

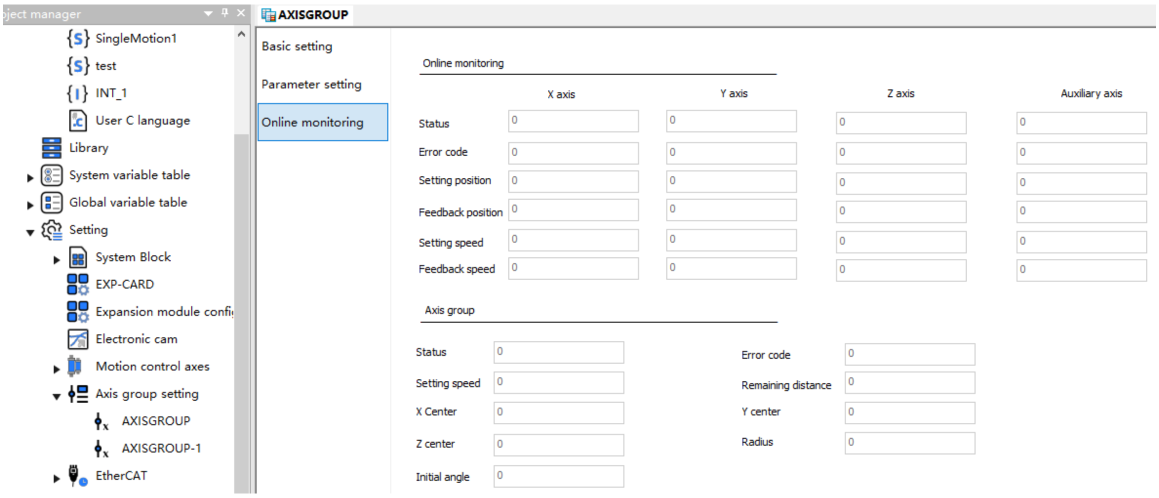
If "Interrupt deceleration to zero" is chosen, the speed will first decelerate to zero as per the current speed (the deceleration of the function block will be chosen first, and if it is too small, then the fault deceleration will be used), and then accelerate to the target speed in reverse. If "Interrupt deceleration to zero" is not chosen, the speed jumps directly to the opposite direction.

- Here, "Axis group failure deceleration" refers to the deceleration used when interpolation judgment is wrong, and the maximum speed, acceleration and jerk of axis group will all limit the interpolation.
- "Maximum number of axes in the axis group" refers to the number of axes configured in axis group (only displayed, changed as per the axis configuration in the basic configuration).



13.1.4 Online Monitoring

On the "Online monitoring" page, observe the online parameters of axis groups, which are divided into single axis monitoring in the upper part of axis groups and axis group monitoring in the lower part.



Name	Parameter	Parameter description
Single axis	State	0: The axis is not enabled 1: The axis is in fault condition 2: The axis calls stop module 3: The axis is enabled successfully 4: The axis is in a point motion state 5: The axis is in continuous motion 6: The axis is in the homing state 7: The axis is in synchronous motion or axis group motion state
	Fault code	Not enabled
	Set position	Instruction position of controller
	Feedback position	Servo feedback position
	Set velocity	Instruction speed of controller
	Feedback speed	Servo feedback speed
Axis group	State	1: There is an axis in the disable state in the axis group 2: There is an axis in the single axis stop state in the axis group 3: There is an axis in the homing state in the axis group 4: There is an axis in the single axis or master-slave axes motion state in the axis group 5: There is an axis in the error state in the axis group 6: All axes in the axis group are enabled 7: MC_GroupStop or MC_GroupImmediateStop is called 8: The function block of the axis group is pulled up and successfully enters this state
	Fault code	Axis group error code
	Set velocity	Instruction resultant velocity
	Remaining distance	Target resultant distance minus current resultant distance
	X center	Arc interpolation mode, X axis coordinates of the center of the circle
	Y center	Arc interpolation mode, Y axis coordinates of the center of the circle
	Z center	Arc interpolation mode, Z axis coordinates of the center of the circle

Name	Parameter	Parameter description
	Radius	Arc interpolation mode, arc radius
	Initial angle	The initial angle at which a circle is drawn is the angle between the starting tangent direction and the X axis, which is positive counterclockwise and negative clockwise

13.2 Interpolation Operation

13.2.1 Basic Introduction

Before creating an axis group, a single axis shall be created and configured (refer to the introduction in the single axis section), and then the relevant parameters of the axis group shall be configured referring to section "13.1.3 Upper Computer Axis Group Configuration Interface Configuration".

13.2.2 State Machine of Axis Group

The state of the axis group is related to the single-axis state, and the initial state is GroupInit. Update the axis group status in the following three situations:

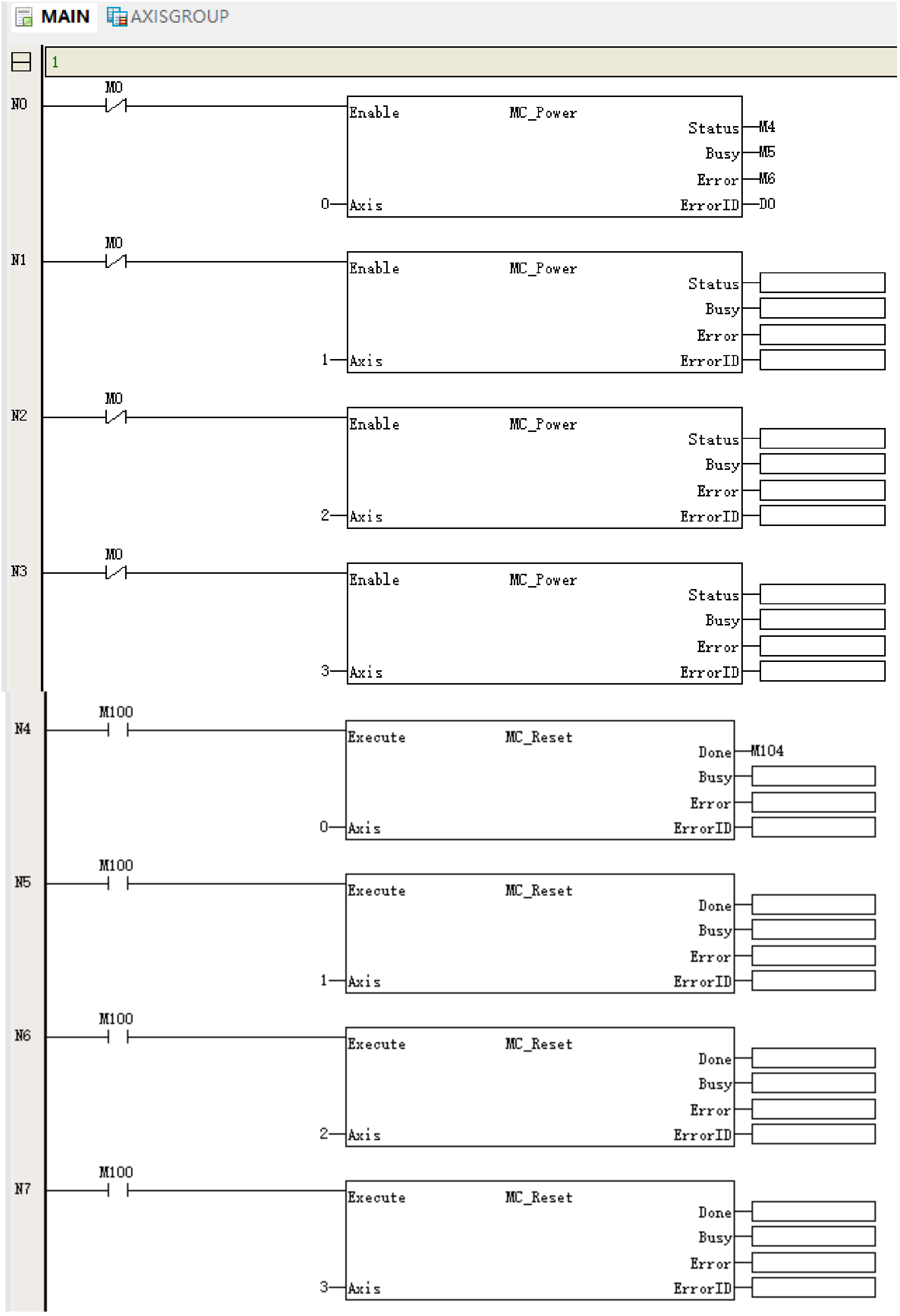
- In the initial state, check the status of the update axis group.
- If the axis group status is GroupErrorStop or SingleDisabled, check until the axis group status is updated to GroupStandStill when all the axes in the axis group are enabled.
- Each time the axis group module is pulled high, the state machine of the single axis is checked to switch the state machine of the axis group.

There are nine cases of axis group state machines:

Axis Group Status	Status Code	Status Description
GroupInit	0	Initial value
SingleDisabled	1	There is an axis in the disable state in the axis group
SingleStop	2	There is an axis in the single axis stop state in the axis group
SingleHoming	3	There is an axis in the homing state in the axis group
SingleMotion	4	There is an axis in the single axis or master-slave axes motion state in the axis group
GroupErrorStop	5	There is an axis in the error state in the axis group
GroupStandStill	6	All axes in the axis group are enabled
GroupStopping	7	Axis Group Stop or Axis Group Immediate Stop is called
GroupSynchronizeMotion	8	The function block of the axis group is pulled up and successfully enters this state

13.2.3 Axle Group Enable and Reset

The premise of axis group enable is that every single axis in the axis group is enabled, which can be completed by calling the MC_Power module of single axis; Similarly, every single axis needs to be reset before the axis group can be reset, which can be completed by calling the MC_Reset module of single axis. If every single axis in an axis group is switched to the StandStill state, the axis group state automatically switches to the GroupStandStill state. The following figure shows an example of use:

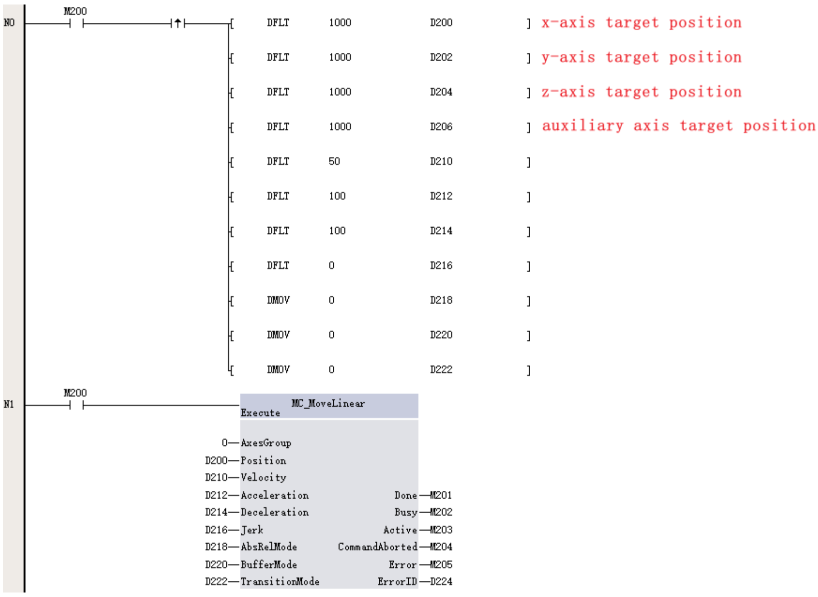


13.2.4 Linear Interpolation

Please use it with the explanation of instruction parameters in the Instruction Manual for the first time.

Linear interpolation calls the instruction MC_MoveLinear. Before running it, please make sure that the panels of "AXISGROUP-Basic setting" and "AXISGROUP-Basic setting" are configured, and all axes in the axis group are in the StandStill state.

The following figure is an example, where the rising edge will set the coordinate positions (X-axis, Y-axis, Z-axis, auxiliary axis) to (1000, 1000, 1000, 1000).



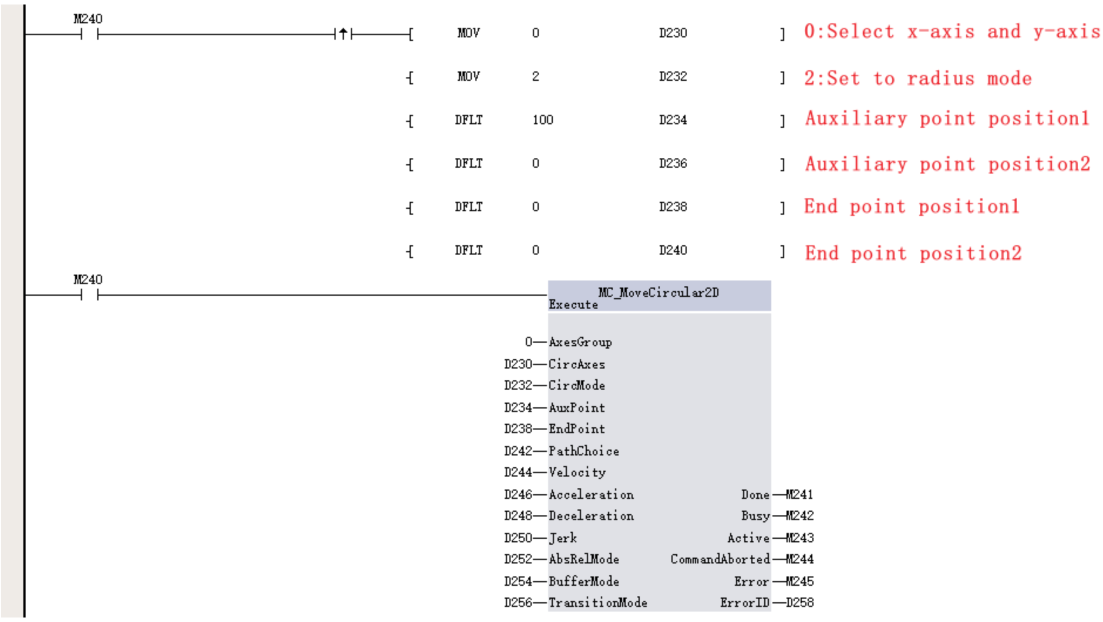
Note: The parameter "Position" of MC_MoveLinear is an array of length 4 with members of REAL type. The data length of D200 – D201 is also of REAL type, so you only need to assign the X-axis target value of 1000 to D200. Similarly, the Y-axis target value is assigned to D202, the Z-axis target value is assigned to D204, and the auxiliary axis target value is assigned to D206. At the same time, the next parameter "Velocity" should be allocated from the memory segment after D208.

13.2.5 Plane Arc Interpolation

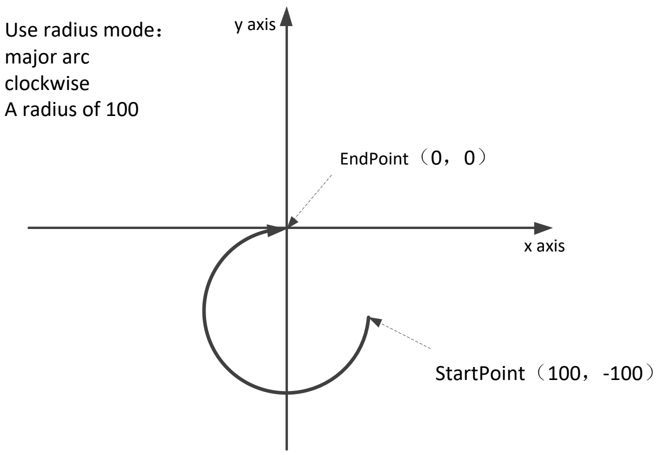
Please use it with the explanation of instruction parameters in the Instruction Manual for the first time.

Linear interpolation calls the instruction MC_MoveCircular2D. Before running it, please make sure that the panels of "AXISGROUP-Basic setting" and "AXISGROUP-Basic setting" are configured, and all axes in the axis group are in the StandStill state.

The following figure shows an example of use. The X-axis and Y-axis are selected to draw a circle in the radius mode (CircMode=2), the radius is set to 100 (AuxPoint [0] = 100), and 100 > 0, so the major arc is selected to draw a circle (the central angle is greater than 180°). PathChoice uses the default value (clockwise), and the end position is (0,0). The starting position is the current axis position (100,-100).



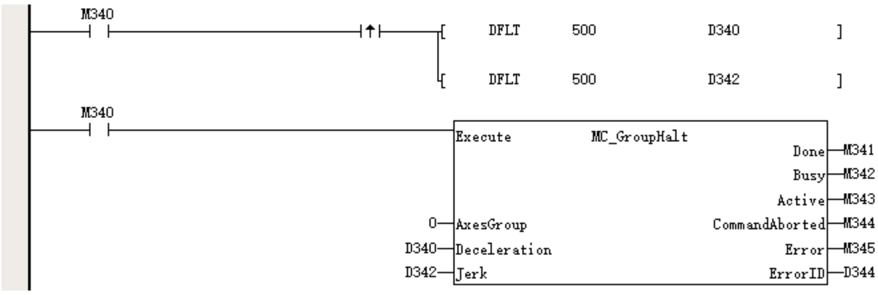
According to the given parameters, the planning trajectory is as follows:



13.2.6 Axis Group Halt

The axis group calls MC_GroupHalt for halt. Only when the current axis group state is in the GroupStandStill or GroupSynchronizeMotion state can MC_GroupHalt be enabled successfully. When finished, the axis group status is set to GroupStandStill.

The example of use is shown in the following figure.

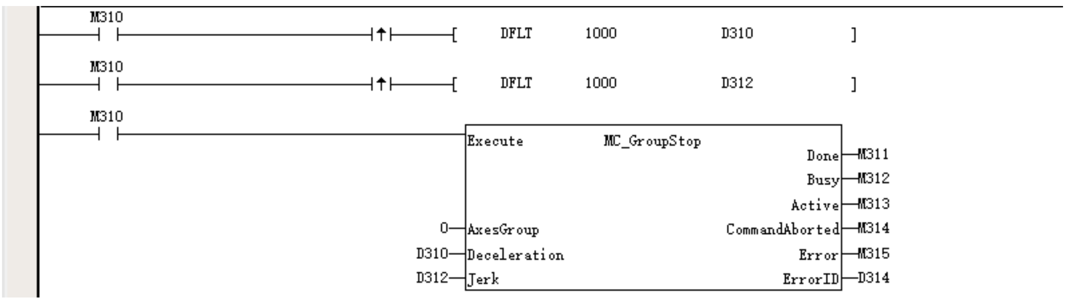


Note: MC_GroupHalt can be pulled up again to interrupt itself and will execute according to the latest module parameters. When multiple MC_GroupHalt function blocks run with the same axis, the latter function block will interrupt the previous module and the previous one will be displayed as being interrupted.

13.2.7 Axis Group Stop

The axis group calls MC_GroupStop for halt. Only when the current axis group state is in the GroupStandStill or GroupSynchronizeMotion state can MC_GroupStop be enabled successfully. When pulled up successfully, the axis group status is set to GroupStopping.

The example of use is shown in the following figure.



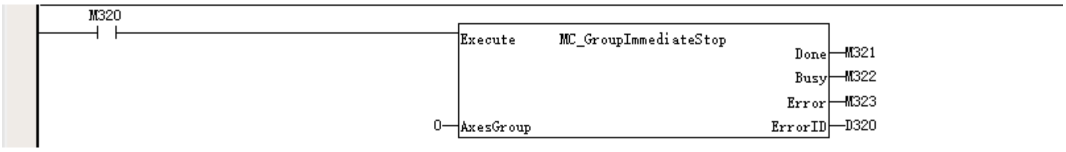
Note:

- You need to pull the function block low to clear the GroupStopping state and switch to the GroupStandStill state. GroupStop can interrupt the running GroupHalt, but it cannot interrupt GroupImmediateStop.
- MC_GroupStop can be pulled up again to interrupt itself, but when multiple MC_GroupStop run with the same axis, the first MC_GroupStop that is successfully pulled up is valid, and the latter will report an error.

13.2.8 Axis Group Immediate Stop

The axis group calls MC_GroupImmediateStop for halt. Only when the current axis group state is in the GroupStandStill or GroupSynchronizeMotion state can MC_GroupImmediateStop be enabled successfully. When pulled up successfully, the axis group status is set to GroupStopping.

The example of use is shown in the following figure.



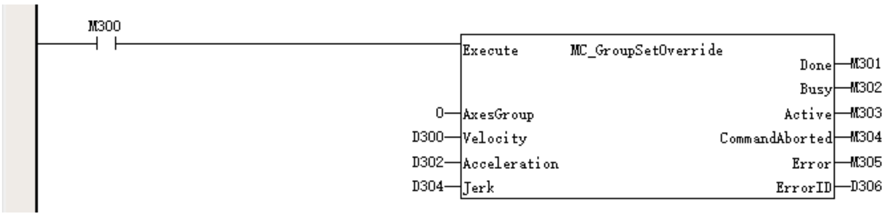
Note:

- If you need to pull the function block low to clear the GroupStopping state and switch to the GroupStandStill state. MC_GroupImmediateStop can interrupt the running GroupStop, GroupHalt.
- MC_GroupImmediateStop cannot be pulled up again to interrupt itself. When multiple blocks call the same axis group, only the first block takes effect, and the following ones will report an error.

13.2.9 Axle Set Speed regulation

The axis group calls "MC_GroupSetOverRide" for speed regulation, and if the current axis group status is GroupStandStill or GroupSynchronizeMotion, the module can complete the output of Done signal. After being pulled up successfully, the current interpolation speed will change to the target speed of speed regulation. If the parameter value of MC_GroupSetOverRide is modified online, the interpolation speed will change in real time. If MC_GroupSetOverRide is pulled down, the original interpolation speed will be restored.

The example of use is shown in the following figure.

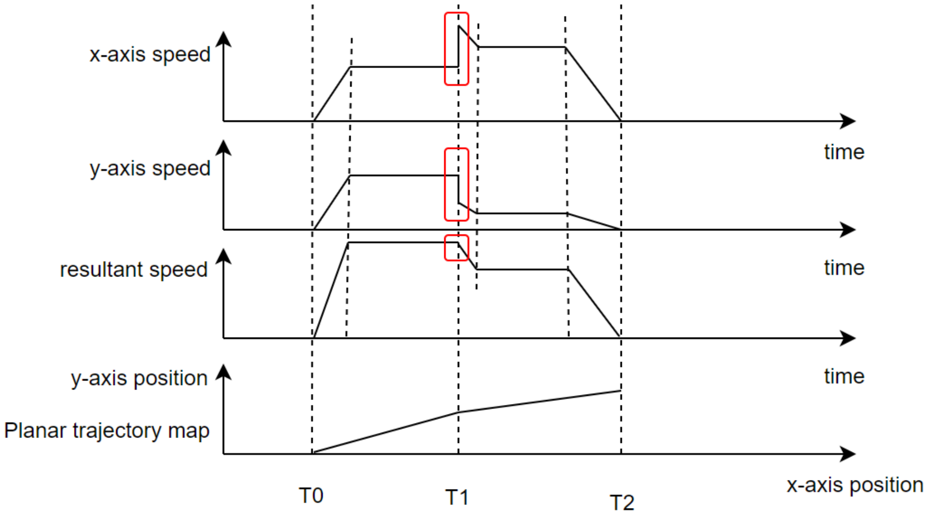


Note: At present, when multiple function blocks of MC_GroupSetOverRide call an axis group number, only the first block takes effect, and the later ones will report an error.

13.2.10 Axis Group Interrupt Mechanism

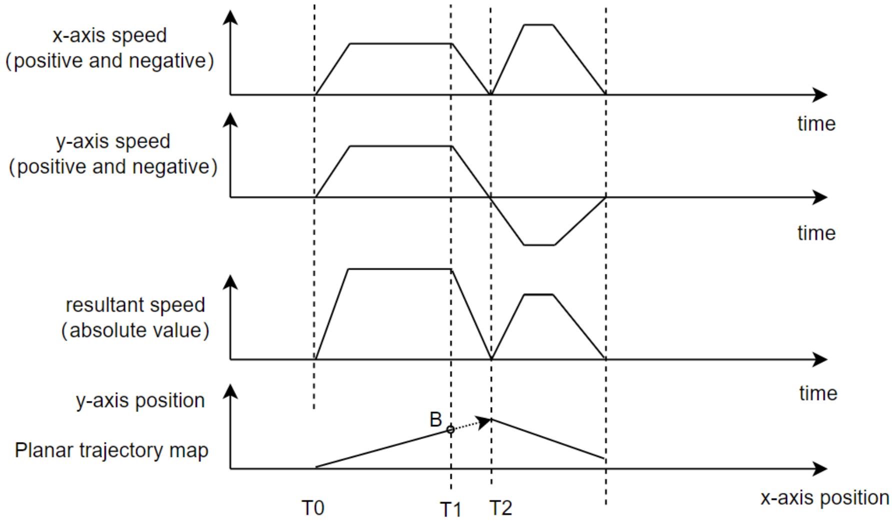
The interruption mechanism of axis group is that the resultant speed is unchanged after interrupt, while the speed of single axis may jump to a certain extent. Here, a linear interpolation is taken as an example to illustrate.

1. Default: The resultant speed is unchanged. If the resultant speed direction changes, a single-axis speed jump occurs:



Note: At time T1, the linear interpolation was interrupted, and it can be seen that there was no jump in the "resultant speed" at T1;At T1, both the "x-axis speed" and "y-axis speed" experienced a jump. If a single axis needs to undergo reverse motion,Then the speed of the axis will undergo a reverse jump from positive to negative.

2. In the case where the single axis moves in reverse after interruption, a switch "Interrupt deceleration to zero" is provided in the "AXISGROUP>Parameter setting" panel. After the user selects this option, the single axis speed will be decelerated to zero first during interruption, and then it will be accelerated to the new target speed in reverse:



Note: At time T1, the function block needs to be pulled up again to interrupt itself, which is point B, and the y-axis needs to run in the opposite direction because the "interrupt deceleration to zero" is selected. Here, the y-axis and x-axis first decelerate to zero together (as shown by the dashed arrow between T1 and T2), then accelerate from T2 to the new target position, so that there is no speed jump during the speed change process.

14 Electronic Cam Function

14.1 Brief Introduction of Electronic Cam

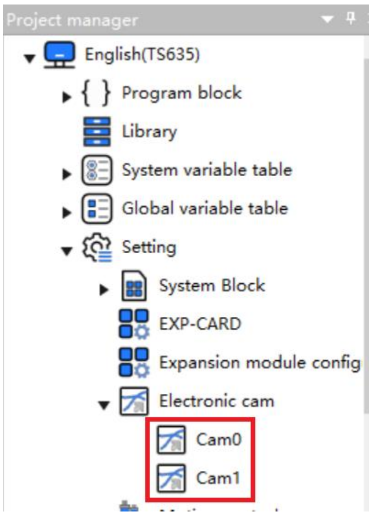
The electronic cam essentially indicates the movement of the axis following the master axis, and the movement relationship between the master and the slave axes can be expressed by the key point data of the cam table or the electronic gear ratio.

- The electronic cam table can be used to establish the data of up to 361 key points. With the electronic gear ratio, there is only a fixed proportional relationship between the master and slave axes.
- If electronic gears are used, only the numerator and denominator of electronic gear ratio need to be set instead of the cam table data. If electronic cam is used, it is necessary to set the data of electronic cam table first.
- The programming software can configure up to 16 cam tables, and 8 electronic cams can be used at the same time in the program, and each cam table has up to 361 key points.
- In the process of cam execution, the cam table is switched online by instruction trigger.

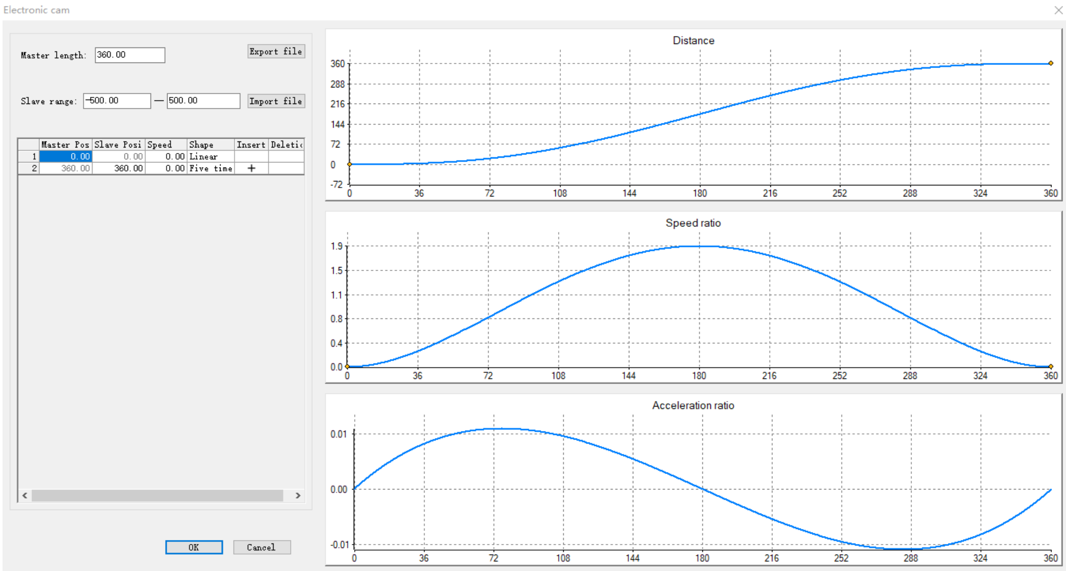
14.2 Software Configuration

14.2.1 Overview

Choose "Project manager" > "Setting", and right-click on "Electronic cam" to add the electronic cam table.



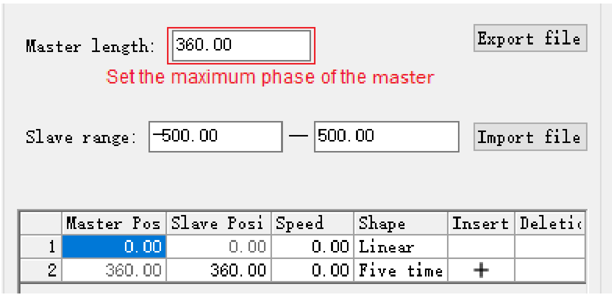
The left side of the cam table interface is a parameter point editing area, and the right side is a graphic editing area.



14.2.2 Cam Node Setting

Users can set cam nodes in the parameter point editing area according to application requirements.

By clicking "+" in the insert option you can add a cam node data and edit the related data. You can also select the specified node data and click "-" in the delete option to delete the node.

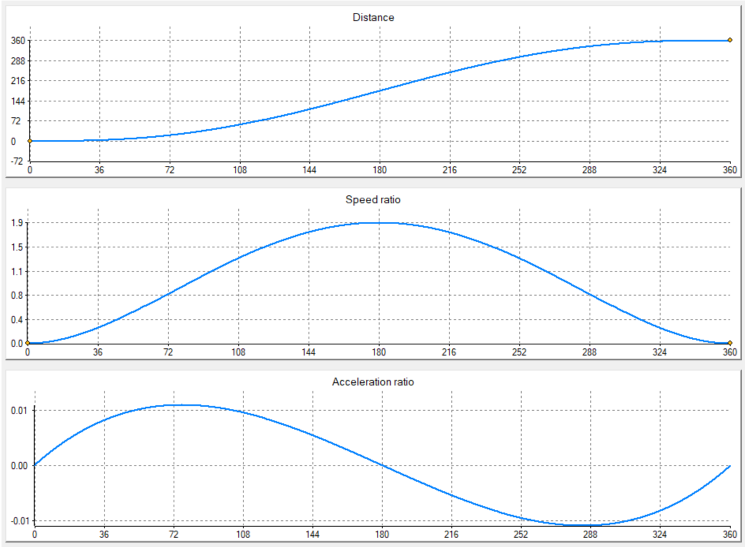


Note:

- The master axis position and slave axis position of the first point default to 0 and cannot be changed.
- The master axis phase is arranged in ascending order.
- The last point of the master axis determines the maximum period of the master axis, so there is no need to set the period separately.

14.2.3 Cam Curve Setting

Users can set cam curves in the graphic editing area according to the actual application requirements, including position, speed ratio and acceleration ratio curves.



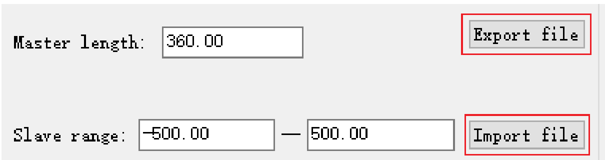
Note:

- The position curve moves the key points of the cam up and down, left and right, the speed ratio curve can only move up and down, and the acceleration ratio curve is not allowed to be changed.
- The last point is only allowed to be dragged up and down. If you need to change the size left and right, you can manually modify the master axis length.
- In the three coordinate systems, click on the line segment of two key points in any coordinate, and the line segment of two key points in the three coordinate systems is displayed in bold.

14.2.4 Import and Export

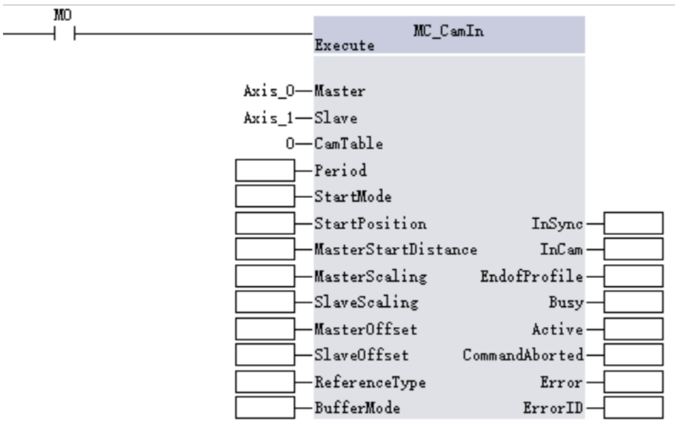
A single cam table can be exported/imported.

Choose the specified electronic cam, click export file, import file to import/export electronic cam in "txt" file format.

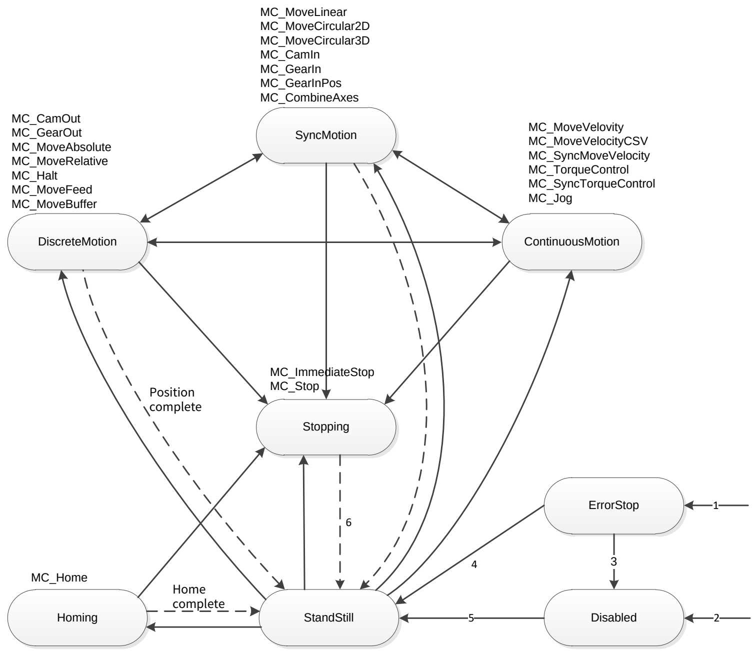


14.2.5 Call Instruction

Every time a user builds a cam table, the software background will assign a system variable to represent the cam table. The status of cam table can be monitored in PLC program and can be used as the parameter of MC_CamIn and other instructions.



14.3 State Machine



The state machine is described as follows:

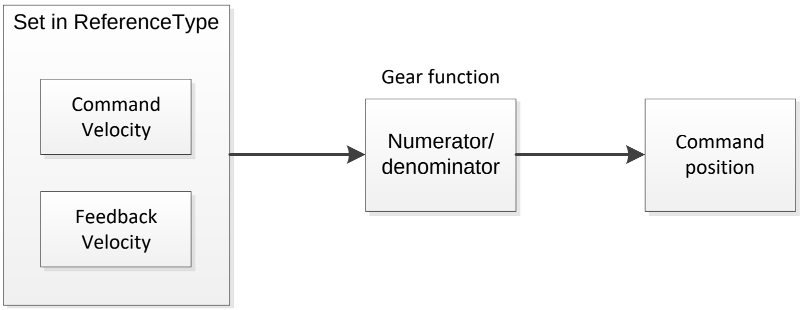
State	Function description
Disabled	Disabled state
ErrorStop	Fault stop state
Standstill	Enabled state
Homing	State of returning to home
Stopping	Stopped state
Discrete Motion	Discrete motion state
Continuous Motion	Continuous motion state
Synchronized Motion	Synchronized motion state

Conversion	Conversion Condition
1	When the fault detection logic of the axis detects a fault, it immediately switches to this state.
2	When the axis has no fault and MC_Power energy flow is OFF
3	When MC_Reset is called to reset axis failure and MC_Power energy flow is OFF
4	When MC_Reset is called to reset axis failure and MC_Power energy flow is ON
5	When MC_Power energy flow is ON and the output flag Status is ON
6	When MC_Stop(MC_ImmediateStop).Done=ON and the energy flow of the graphic block is OFF

14.4 Electronic Cam Operation

14.4.1 Gear Action

14.4.1.1 Basic Block Diagram



14.4.1.2 Function Description

The types of gear action master axis and slave axes are as follows:

- Master axis: fieldbus servo axis, local pulse axis, local encoder axis
- Slave axis: fieldbus servo axis, local pulse axis

Gear action function description:

1. Gear action is started by MC_GearIn instruction and unsynchronized by MC_GearOut instruction.
2. After the action is started, the speed obtained by multiplying the master axis speed by the gear ratio is taken as the target speed, and the acceleration and deceleration operation is carried out on the slave axis.
3. Before reaching the target speed, the phase is called Catching Phase, and after reaching it, the phase is called InGear Phase.
4. Gear action is executed by setting the gear ratio between master and slave axes.
5. When the gear ratio is positive, the slave axis runs in the same direction with the master axis; When it is negative, the slave axis runs in the opposite direction of the master axis.
6. Please refer to the instruction MC_GearIn in the Instruction Manual for details.

14.4.1.3 Example

Establish two new fieldbus servo axes, and the second axis perform gear action following the first axis at the gear ratio of 1:1.

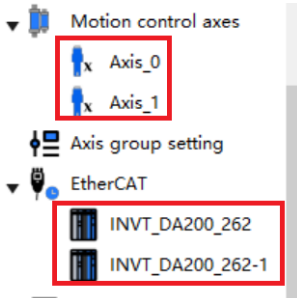
The operating steps are as follows:

The operating steps are as follows:

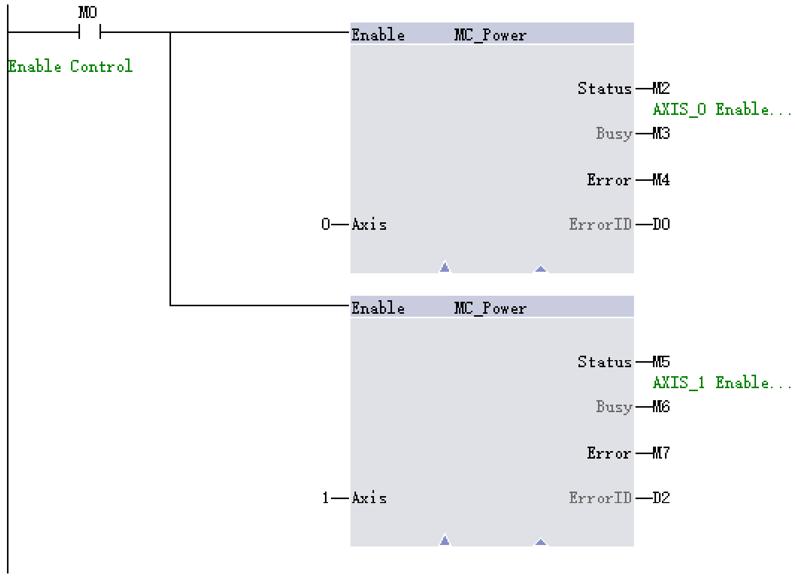
Step 1 In the new project, establish two fieldbus servo axes, one is the master axis and the other is the slave axis.

Of the two fieldbus servo axes, Axis_0 is the master axis and Axis_1 is the slave axis.

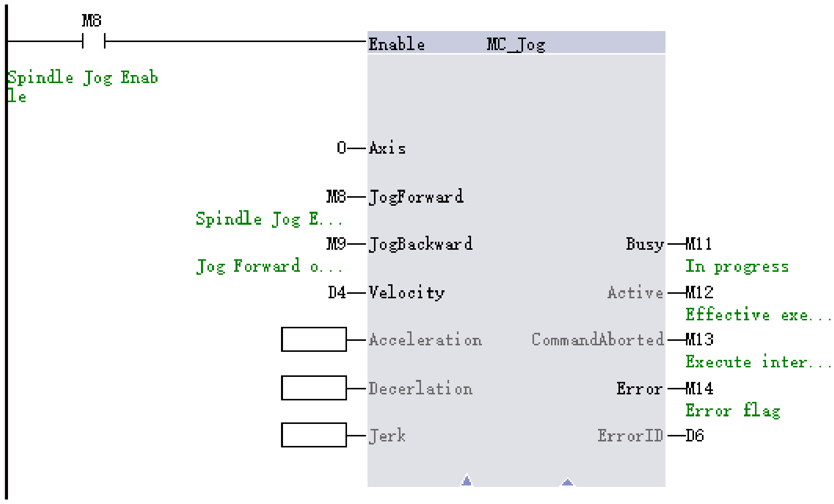
In both servo drivers, bind INVT_DA200_171 to Axis_0 and INVT_DA200_171_1 to Axis_1.



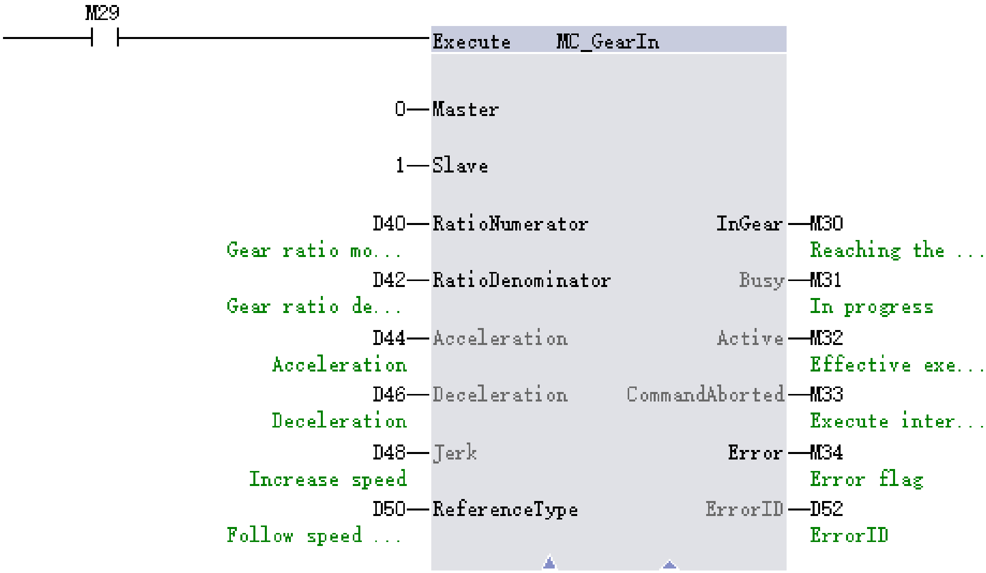
Step 2 Call MC_Power to control the enable of the master axis and slave axis.



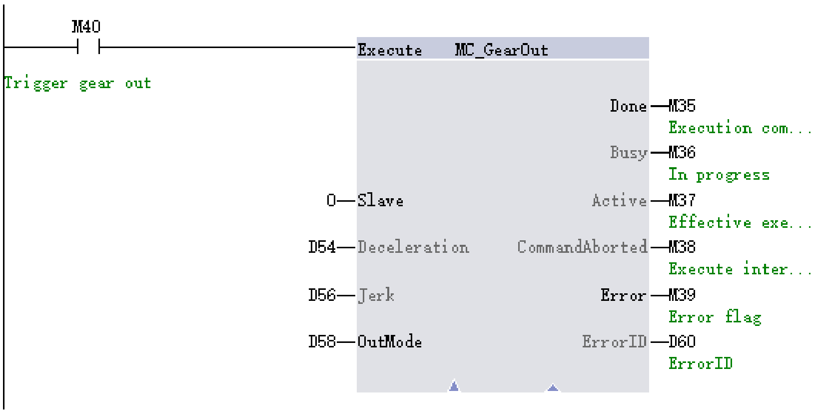
Step 3 Call MC_Jog to control the forward and reverse motion of the master axis.



Step 4 Call MC_GearIn to execute the gear following action at the gear ratio of 1:1.



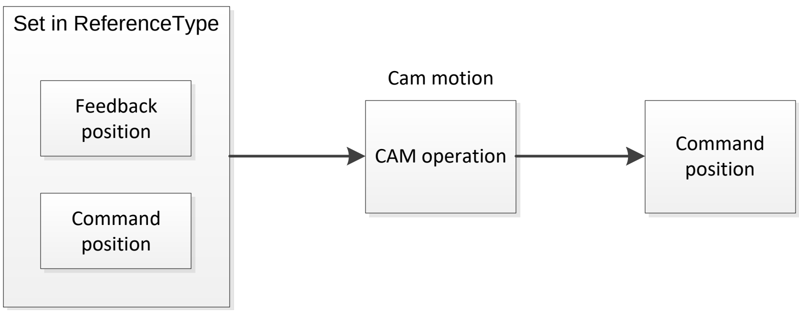
Step 5 Call MC_GearOut to touch the gear action.



14.4.2 Cam Action

Cam action means that the slave axis moves synchronously with the position of the master axis according to the cam table

14.4.2.1 Basic Block Diagram



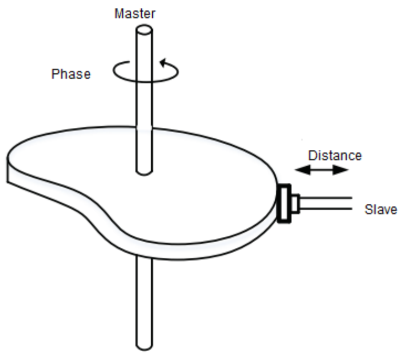
14.4.2.2 Function Description

The master axis and slave axis types applicable to cam action are as follows:

- Master axis: fieldbus servo axis, local pulse axis and local encoder axis.
- Slave axis: fieldbus servo axis, local pulse axis.

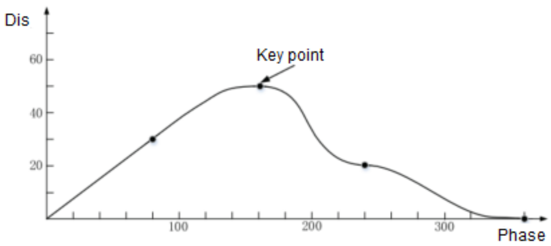
1. Start the cam action or change the cam table through MC_CamIn, and cancel cam action with MC_CamOut.

2. The typical cam structure is shown in the following figure. The master axis rotates periodically, and the slave axis reciprocates in one direction under the control of the master axis.



The electronic cam just imitates this structure, where an axis is chosen as the master axis with the other axis as the slave axis, and both of them move synchronously according to the set cam curve.

3. Cam Curve



4. Cam table

The cam table is a two-dimensional coordinate system, in which the abscissa represents the phase of the master axis and the ordinate represents the displacement of the slave axis. Some key points are set in the coordinate system, and each two key points are connected with set curves (such as straight lines and quintic curves) to form a cam curve.

Phase	Shift
0	0
80	30
160	50
240	20
360	0

Note: Please refer to the instructions MC_CamIn and MC_CamOut in the Instruction Manual for details.

14.4.2.3 Example

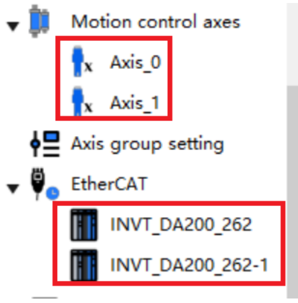
Axis_0 serves as the master axis of the cam, and Axis_1 serves as the slave axis that follows Axis_0 to perform cam action.

The operating steps are as follows:

Step 1 Create a new project, and establish two fieldbus servo axes, one as the master axis and one as the slave axis.

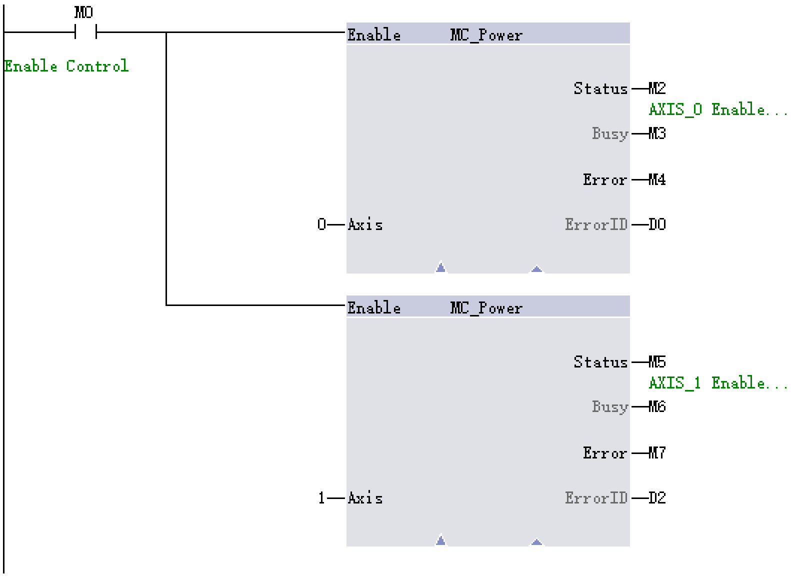
Of the two fieldbus servo axes, AXIS_0 is the master axis and AXIS_1 is the slave axis.

In both servo drivers, bind INVT_DA200_171 to Axis_0 and INVT_DA200_171_1 to Axis_1.

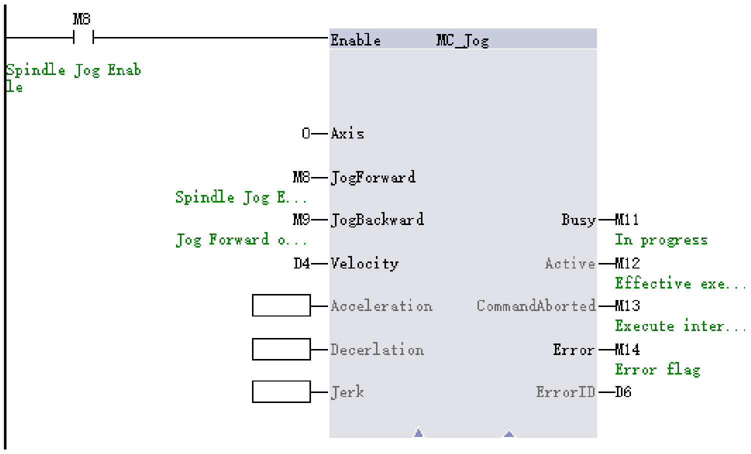


Step 2 Create a new cam table.

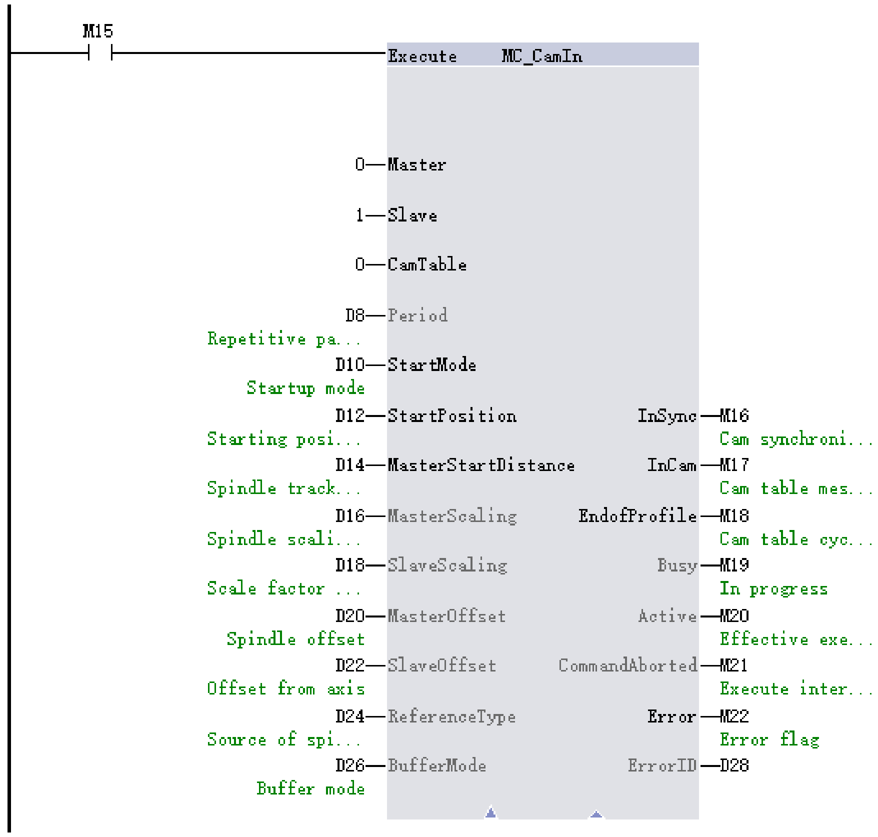
Step 3 Call MC_Power to control the master and slave axes enable.



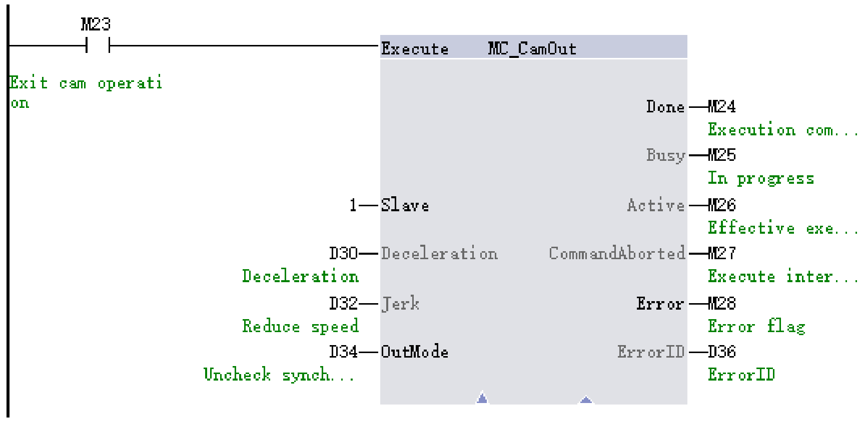
Step 4 Call MC_Jog to control the forward and reverse operation of the master axis.



Step 5 Call MC_CamIn to perform the cam action.



Step 6 Call MC_CamOut to cancel the cam action.



14.4.3 Cam Table

14.4.3.1 Brief Introduction of Cam Table

1. In the cam function module, a pair of data composed of master axis phase and slave axis displacement is defined as cam data, and the combination of multiple cam data is defined as cam table.
2. The phase and displacement values of the cam data in the cam table are expressed as relative quantities starting from the starting point "0.0".
3. In the cam action, the displacement of the slave axis is calculated according to the phase of the master axis and the set curve type, so as to control the action of the slave axis.
4. After creating the cam table through the cam editor of Auto Station Pro, the cam data in the cam table can also be changed through MC_GenerateCamTable in the user program.

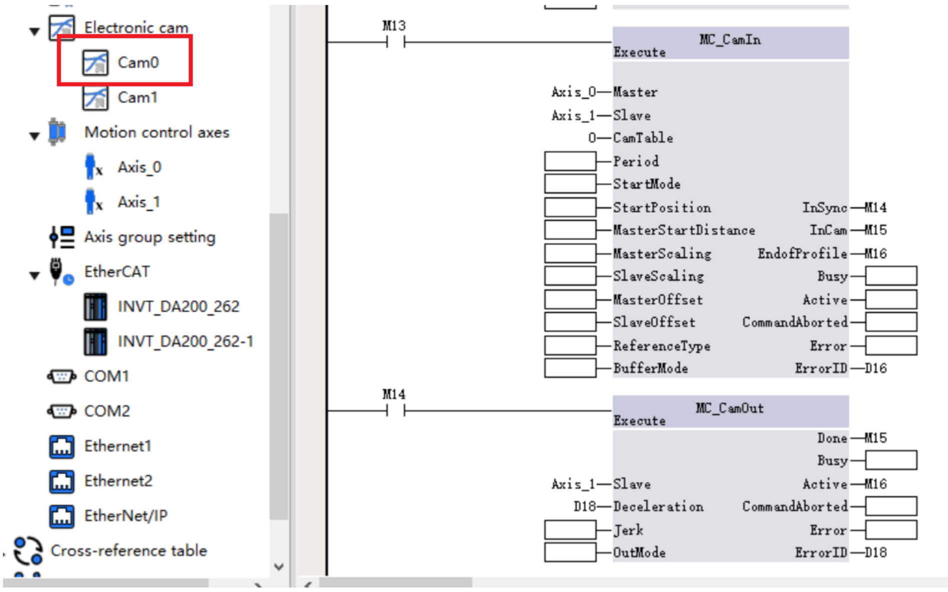
14.4.3.2 Specification of Cam Table

When creating a cam table, the user should follow the following specifications:

Item	Description
Total number of cam key points supported by each cam table	361
Total number of cam tables supported	16
Rules of switching cam table in cam action	Call MC_CamIn to switch the cam table, which will take effect on the next cam cycle
Read and write of cam data	View cam table status and key point data by system variables under the name of cam table. You can directly modify the cam key point data in the cam table, and make the changes become effective through MC_GenerateCamTable, and the cam will run according to the new cam table in the next cam cycle.

14.4.3.3 Create a Cam Table

Cam table variables can only be created in the background. Every time a cam table is added in the background, a cam table variable is created by default. And enter the ID number of the cam table for CamTable.



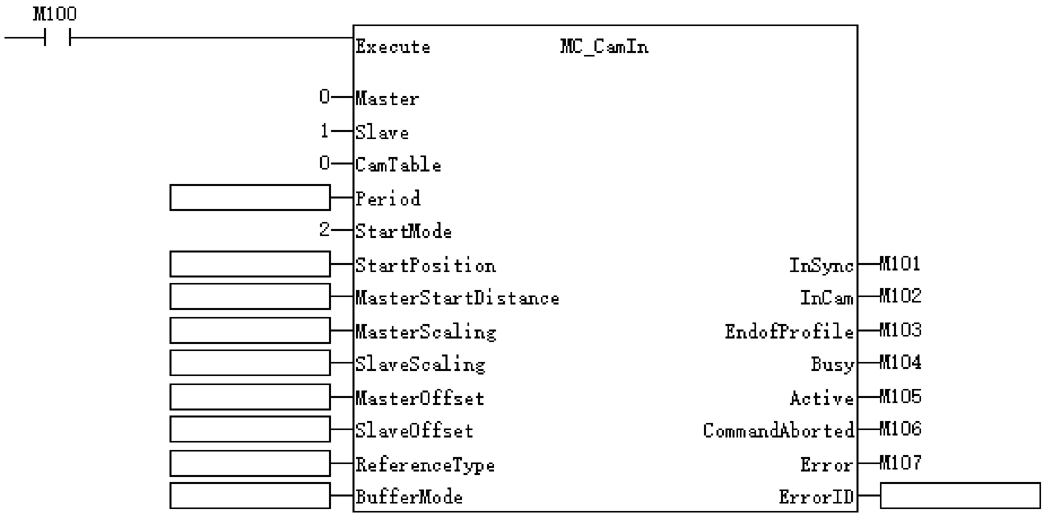
14.4.3.4 Switching Cam Table

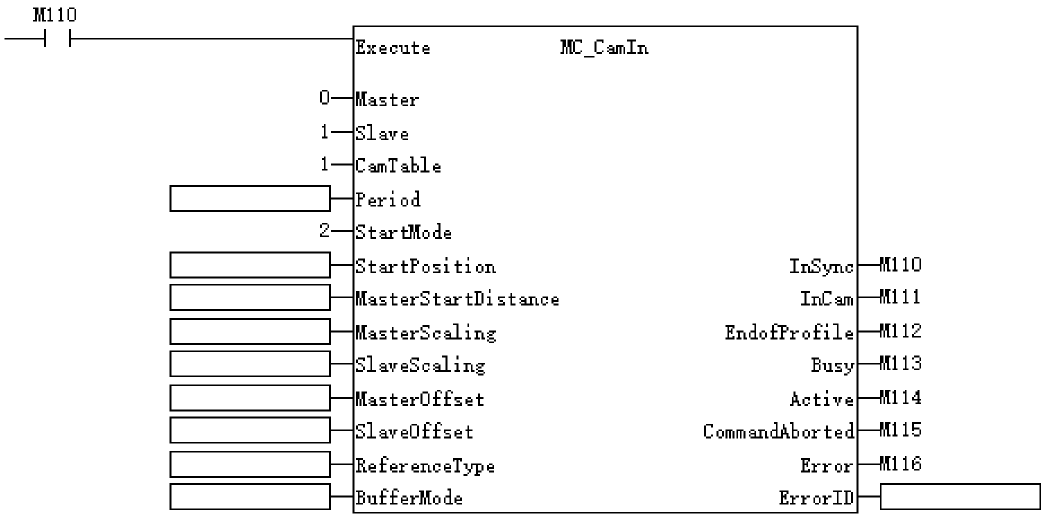
In the process of cam execution, the cam table can be replaced by triggering MC_CamIn. After that, the cam table is in the buffer state, and the buffered cam table will take effect in the next cam cycle.

Note: Only one cam table can be buffered. If multiple MC_CamIn instructions are repeated continuously, the first triggered cam table will be overwritten by the later triggered cam table. The following two instructions are shown:

Step 1 M100 is triggered first. After the instruction detects that the cam parameters are set correctly, M104(Busy) output becomes valid, Axis_1 will start moving according to the curve set by CAM0, and M105(Active) output becomes valid. When M110 is triggered before the end of a cam cycle, the cam table CAM1 is in a buffer state and the M113(Busy) output is active.

Step 2 When Axis_1 finishes the first cam cycle, the first cam instruction is interrupted, M106(CommandAborted) output is valid, Axis_1 starts to move according to the cam curve set by CAM1, and M114(Active) output is valid.





14.4.3.5 Modify Cam Table Data

The cam data can be modified temporarily with the following methods:

Create a new cam node array in PLC program, and copy the values in the cam node array to the cam table with MC_GenerateCamTable, which are executed in the next cam cycle.

Cam Table A

before replacement

Phase	Shift
0	0
50	40
90	60
130	30
200	0
0	0

Cam Table A

after replacement

Phase	Shift
0	0
30	40
70	80
100	120
240	200
360	0

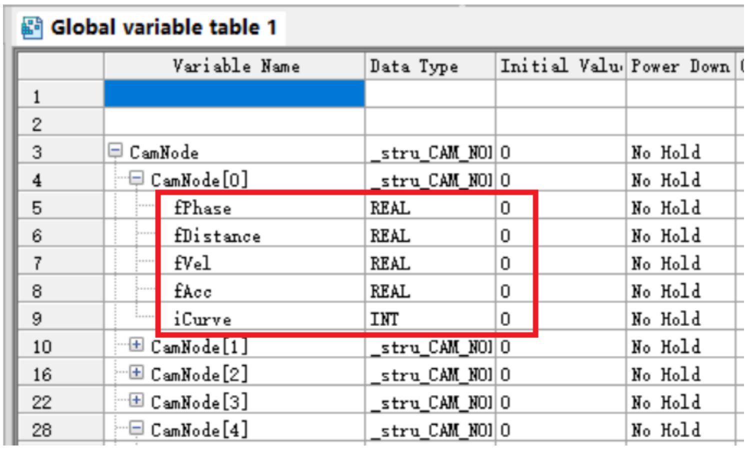
Through MC_Generate CamTable copy ←

Cam node array

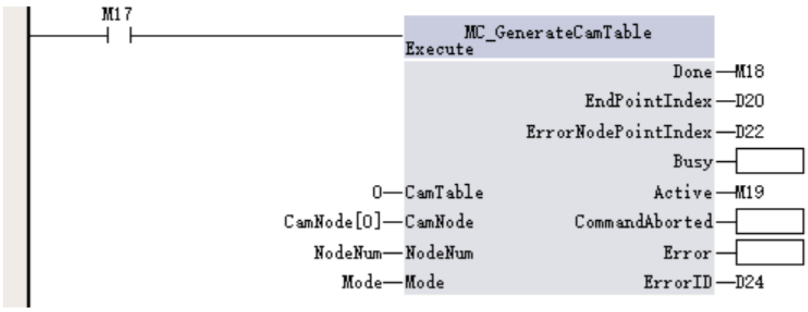
created in PLC program

Phase	Shift
0	0
30	40
70	80
100	120
240	200
360	0

Examples are as follows:


Global variable table 1

	Variable Name	Data Type	Initial Valu	Power Down
1				
2				
3	CamNode	_stru_CAM_NOI	0	No Hold
4	CamNode[0]	_stru_CAM_NOI	0	No Hold
5	fPhase	REAL	0	No Hold
6	fDistance	REAL	0	No Hold
7	fVel	REAL	0	No Hold
8	fAcc	REAL	0	No Hold
9	iCurve	INT	0	No Hold
10	CamNode[1]	_stru_CAM_NOI	0	No Hold
16	CamNode[2]	_stru_CAM_NOI	0	No Hold
22	CamNode[3]	_stru_CAM_NOI	0	No Hold
28	CamNode[4]	_stru_CAM_NOI	0	No Hold

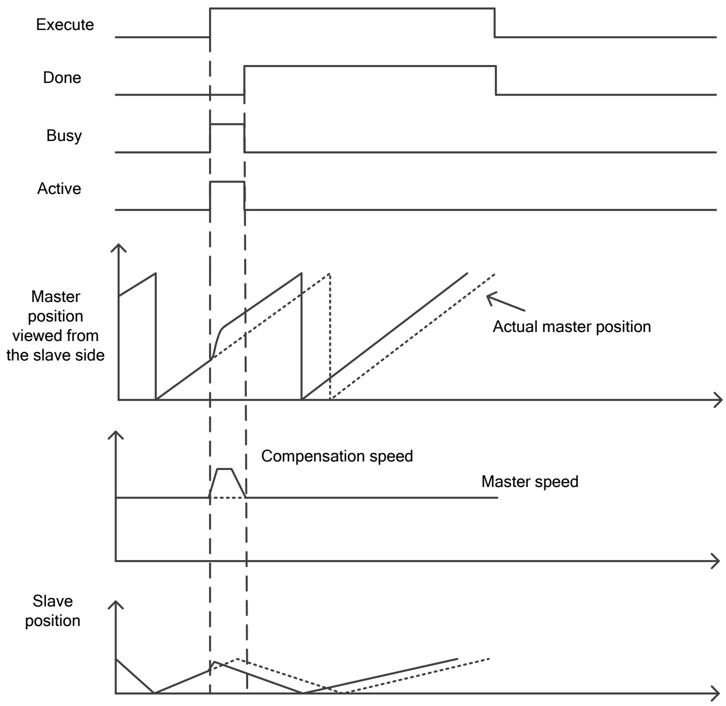


Note: The CamNodeNum above is used to specify the number of nodes in the cam table in the cam node array created in the PLC program.

14.4.4 Master Axis Phase Compensation

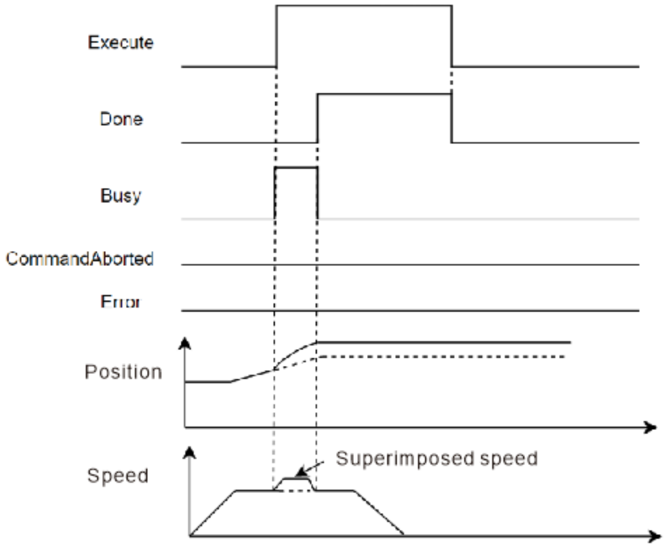
Master axis phase compensation is to perform the phase compensation function to the running master axis (viewed from the slave axis) which is executing instructions.

By starting MC_Phasing, the phase compensation of synchronization control instruction can be carried out. MC_Phasing can specify parameters such as phase compensation amount, target velocity, acceleration, jerk and so on.



14.4.5 Motion Superposition

By calling MC_MoveSuperImPosed, the motion superposition function of the motion control axis is realized, and the motion of the axis will not be interrupted during the motion superposition process.



14.4.6 How to Handle Axis Configuration Parameters in Cam or Gear

Parameter Setting	Processing Method
Gear ratio setting	The master-slave axes of cam and gear support the modification of gear ratio, which is set in the "Transmission ratio" interface.
Limit Processing	● Cam and gear support soft limit in linear mode, and slow down and stop after reaching the limit ● Cam and gear support hard limit in linear mode and circular mode, and stop immediately after reaching the hard limit
Axis error deceleration	When the instruction parameter input is abnormal, the slave axis will switch to the fault state and will decelerate as per the fault deceleration, and then enters the fault shutdown state
Speed limit	● Cam instructions are not limited by the maximum speed of axis ● The gear is limited by the maximum speed of the axis. After the maximum speed of the gear exceeds the maximum speed limit, it runs at the maximum speed.
Acceleration limit	● MC_CamIn is not limited by the maximum acceleration ● MC_GearIn, MC_CamOut and MC_GearOut are limited by the maximum acceleration, and run at the maximum acceleration (deceleration) beyond the maximum acceleration

15 Memory Formula Management

15.1 Overview

Memory management includes custom variables and soft element variable memory management. By obtaining variable memory data at a certain time as data basis for debugging and analysis, variable memory data at a certain time can also be saved as formula data for debugging parameters of different processes or formula parameters of multiple links of the same process.

Specific application scenarios are as follows:

- When the program is abnormal, the current variable memory data can be obtained to analyze the problem.
- Multiple memory data parameters of variables at a certain time are obtained and saved as formula parameter files for other machines to use (to be implemented).
- Real-time monitoring of the current value of all data in the current variable table.
- Synchronization of the variable memory data parameters at the current time to the initial value.
- When commissioning different processes for a set of programs, different formulas of parameter data can be saved (to be realized).
- When commissioning different links of the same process, different sets of formula parameters can be saved (to be realized).

15.2 Power-down Variable Keeping

15.2.1 Range of Power-down Variable Keeping

Soft elements M, S, D and R support power-down keeping of variables within a fixed range which cannot be modified, as shown in the following table:

Element Type	Name	Range of Keeping
M (0–32767)	Auxiliary relay	M1000–M32767
S (0–4095)	Status relay	S1000–S4095
D (0–32767)	Word element	D1000–D32767
R (0–32767)	Word element	R1000–R32767
Custom variable (internal private address, separated from non-power-down keeping custom variables)	Custom Variables	Maximum 128k, depending on user settings

Custom variables can be set as power-down keeping or non-power-down keeping according to users' needs. It supports users to keep up to 128k variables at power-down. The custom variables shown below

are selected as being kept or not being kept in the power-down keeping option.

	Variable Name	Data Type	Initial Valu	Power Down	Comments	Element Ad	Current Valu	Value 1
1	test1	WORD	1	Hold			0	
2	test3	REAL	2.5	Hold			0.000000	
3	test4	WORD	0	Hold			0	
4	test2	INT	3	No Hold			0	

15.2.2 Data Retention File Upload and Download

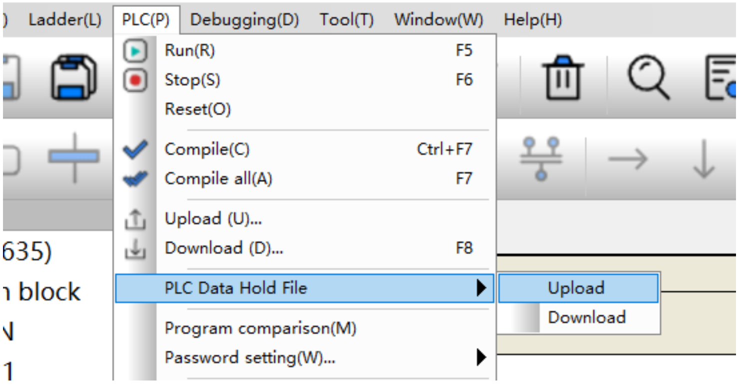
When you need to transfer the power-off retention data from one PLC to another, or when you need to save the current power-off retention data, you can use the data retention file upload and download function.

Principle: The data of the power-off retention elements and custom retention variables can be uploaded and saved as a file on the computer, and when re-downloaded, it will refresh the current values of the data retention elements.

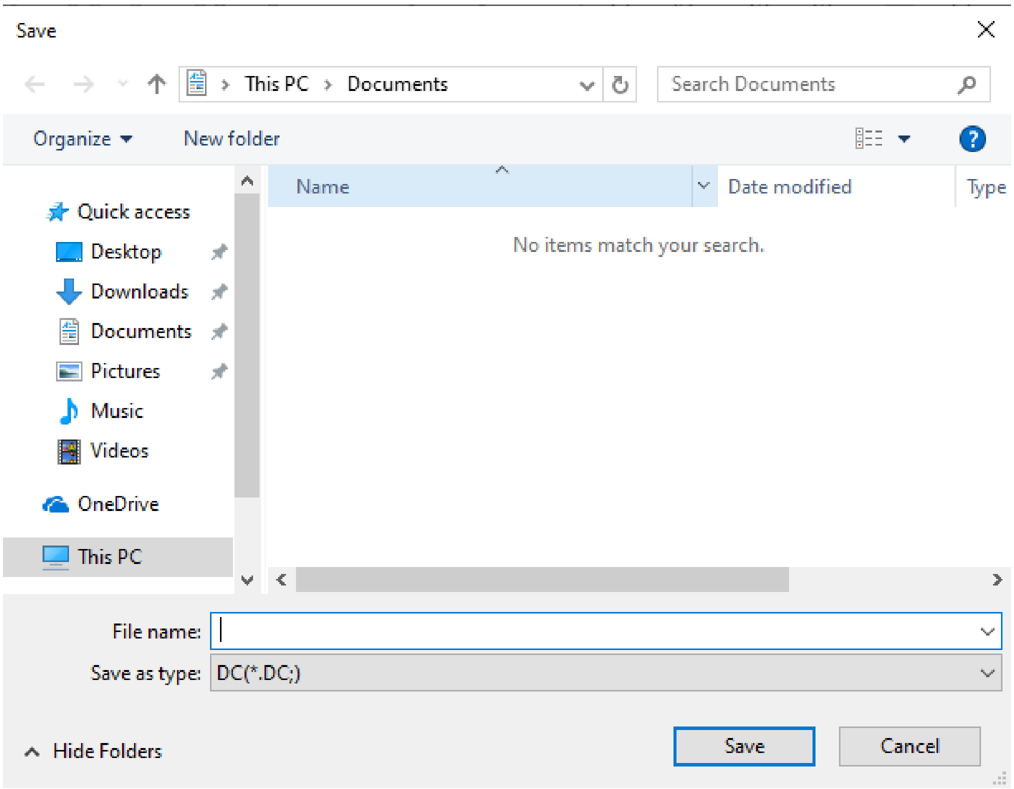
The operating steps are as follows:

Step 1 Connect to the PLC through the Auto Station Pro host controller communication.

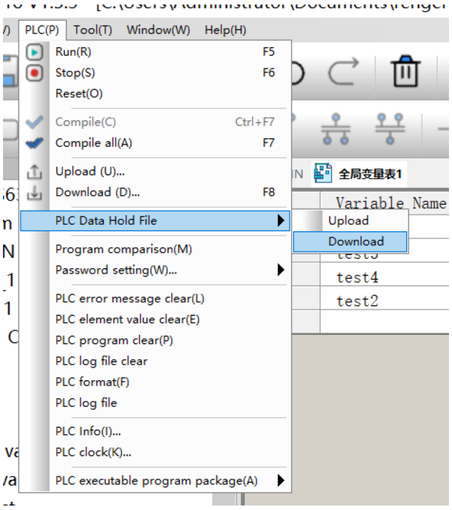
Step 2 Select "Toolbar" > "PLC" > "PLC Data Hold File" > "Upload" in order.



Step 3 Choose an appropriate location to store the data retention file.



Step 4 Use the PLC data retention file download function on another PLC.



15.3 Memory Management of Custom Variable Table

15.3.1 Expand and Fold Complex Type Variables

Custom variable table contains complex type variables, namely arrays and structs. The system supports expanding and folding the sub-members of arrays and structs. Sub-members can only be edited for initial value, comment and data value column, and the values in other columns cannot be edited.

MAIN | Global variable table 2 * | Global variable table 1 *

	Variable Name	Data Type	Initial Valu	Power Down	Comments	Element Ad	Current Valu	Value 1
1	teset3	WORD	0	Hold				
2	teset1	BOOL	ON	No Hold				
3	teset2	INT[5]	OFF	No Hold	test2			
4	teset2[0]	INT	3	No Hold	static0			
5	teset2[1]	INT	4	No Hold	static1			
6	teset2[2]	INT	5	No Hold				
7	teset2[3]	INT	6	No Hold				
8	teset2[4]	INT	7	No Hold				

15.3.2 Monitor Variables

The variable table supports the monitoring function. Click "Download and monitor" in the user project, and the system will monitor all the variables displayed on the current variable table page without adding additional variables to the monitoring table.

		Element Name	Data Type	Display Fo	Current Value	New Value
1	...	teset4	INT	Decimal	16948	
2	...	teset2	INT[5]	Decimal		
3		teset2[0]	INT	Decimal	3	
4		teset2[1]	INT	Decimal	4	
5		teset2[2]	INT	Decimal	5	
6		teset2[3]	INT	Decimal	6	
7		teset2[4]	INT	Decimal	7	

15.3.3 Edit Variable Initializations and Comments

Users can edit the initial values and comments of each member data directly after expanding the variable table, and can also define them in the program custom variable box, which is equivalent to each other.

	Variable Name	Data Type	Initial Valu	Power Down	Comments	Element Ad	Current Valu	Value 1	Value 2
1	teset5	REAL	3.5	No Hold	test5				
2	teset4	INT	0	Hold	test4				
3	teset3	WORD	0	Hold					
4	teset1	BOOL	ON	No Hold					
5	⊟ teset2	INT[5]	OFF	No Hold	test2				
6	teset2[0]	INT	3	No Hold	static0				
7	teset2[1]	INT	4	No Hold	static1				
8	teset2[2]	INT	5	No Hold					
9	teset2[3]	INT	6	No Hold					
10	teset2[4]	INT	7	No Hold					

15.3.4 Switch and Display Binary System

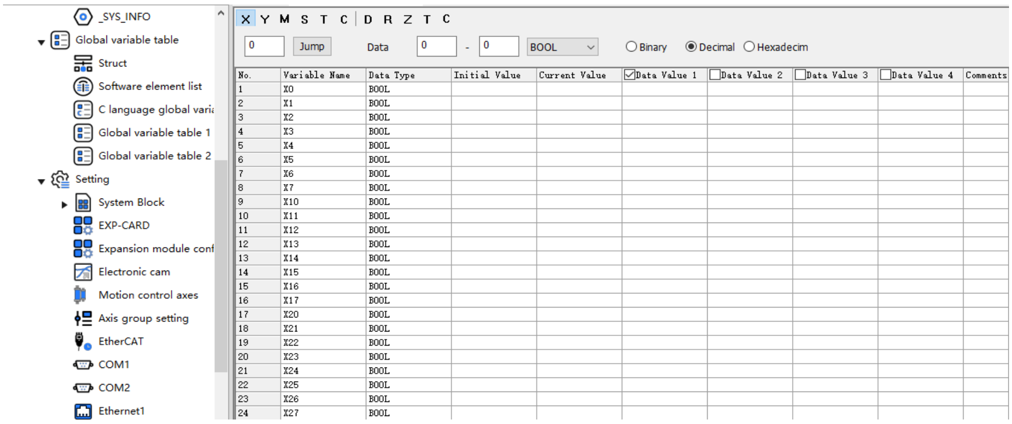
When managing the custom variable table, the user can switch to the display of decimal, binary and hexadecimal.

		Element Name	Data Type	Display Fo	Current Value	New Value
1	...	teset5	INT	Binary	2#100001	
2	...	teset4	INT	Hexadecima	16#0	
3	...	teset3	INT	Decimal	0	
4	...	teset1	INT	Decimal	13	
5	...		WORD	Decimal		

15.4 Memory Management of Soft Elements

15.4.1 Operation Interface

The soft element table contains two functions: comments and memory management, and the operation interface is as shown in the following figure.



1. Switching between soft elements can operate different soft element memories and comments.
2. Used to batch modify the data type of the current page, simply by entering the element numbers at the beginning and end to be modified through two edit boxes respectively.
3. Select the data type to change in the drop-down box. For example, to switch to D element, enter 50 and 100 to indicate the data from D50 to D100, and then click the drop-down box to select REAL, then the data type of D50, D52, D54 ⋯ D100 is modified to REAL. (Note: REAL and DINT are 32 bits, so they need 2 elements).
4. It is a jump function, for example, when switching to D element, enter 1000, click jump or press Enter, and the page will automatically jump to the position of D999.

   Note: In all input boxes, for X and Y elements you can only input octal numbers, and only a 5-digit number at maximum is allowed. If the input data exceeds the total number of elements, it will jump to the last.
5. It is to modify the display format of the data value column and the current value column. For example, clicking the hexadecimal radio box will switch all data to hexadecimal display.

15.4.2 Editing Rules for Data Types

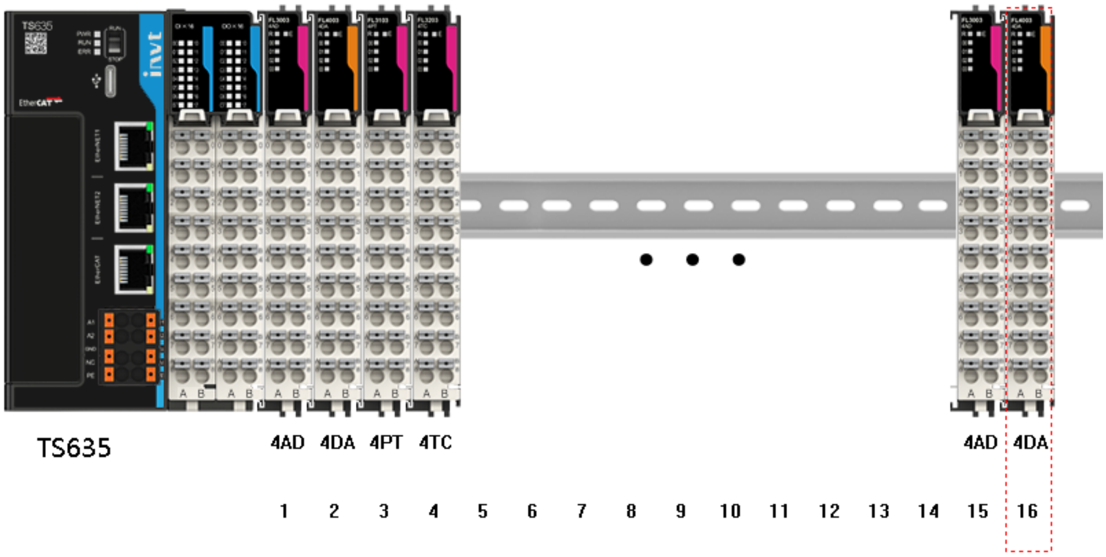
1. Bit elements (X, Y, M, S, B) are only of BOOL type, and their data values can only be selected as "ON/OFF".
2. Word element can be of three data types: INT, DINT and REAL, in which INT is 16 bits, DINT and REAL are 32 bits.
3. When the data type is 32 bits, it takes up 2 word elements, so after selecting REAL and DINT types, the data value in the next row is automatically emptied, and the data type and data value columns are not editable, and the data type columns in the previous row are not editable. When reading memory, the data types are REAL and DINT, and the next row of data values is not read.

16 Expansion Module

16.1 TS600 Series Host Local Expansion Module

16.1.1 Summary of Expansion Module

TS600 can have up to 16 local expansion modules, and access the local extension through module configuration. The hardware configuration of TS600 connected to local expansion module is shown in the following figure.



The supported local expansion module models are shown in the following table:

Product Name	Description
FL1001-1600DI	16-channel digital input module
FL1002-3200DI	32-channel digital input module
FL2201-0008DR	8-channel digital relay output module
FL2102-0016DN	16-channel digital transistor output module-NPN
FL2002-0016DP	16-channel digital transistor output module-PNP
FL3003-4AD	4-channel analog input module
FL4003-4DA	4-channel analog output module
FL3101-4PT	4-channel input thermal resistance temperature detection module
FL3201-4TC	4-channel thermocouple temperature detection module
FL2003-0032DP	32-point digital source output module
FL2103-0032DN	32-point digital sink output module
FL5005-1616DP	16-input and 16-output source module
FL5105-1616DN	16-input and 16-output sink module
FL6121-1EI-DF	One-channel incremental differential input module
FL6112-2EI	Two-channel incremental encoder input module
FL6002-2ES	Two-channel absolute encoder input module

16.1.2 Expansion Module System Variables

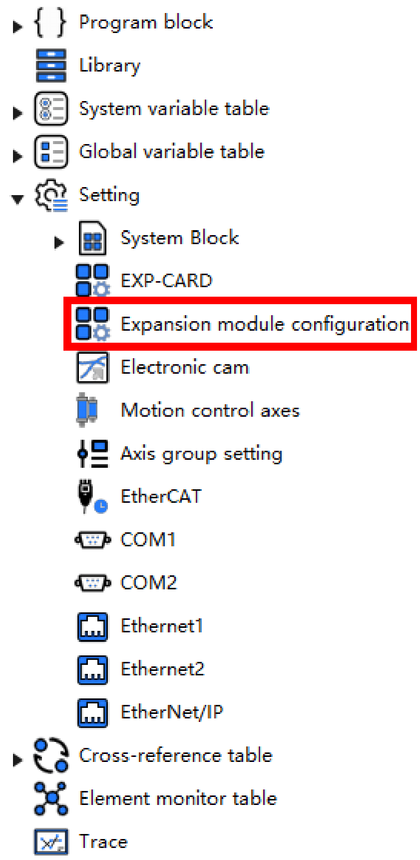
Name	Data type	Description
_sExtModule.CfgNum	INT	User-configured module number
_sExtModule.ActNum	INT	Actually mounted module number
_sExtModule.sExtSlot[n].CfgType	INT	nth expansion module User-configured type
_sExtModule.sExtSlot[n].ActType	INT	nth expansion module Type of actual mount
_sExtModule.sExtSlot[n].Error	BOOL	nth expansion module Error flag
_sExtModule.sExtSlot[n].Disable	BOOL	nth expansion module Disable flag
_sExtModule.sExtSlot[n].SWVersion	DINT	nth expansion module Software version
_sExtModule.sExtSlot[n].LGVersion	DINT	nth expansion module Logic device version

- _sExtModule.ActNum gives feedback on how many expansion modules the user actually mounts.
- _sExtModule.sExtSlot[n].ActType feeds back what type each mounted module is.
- _sExtModule.sExtSlot[n].CfgType feeds back what type of module each user configures.

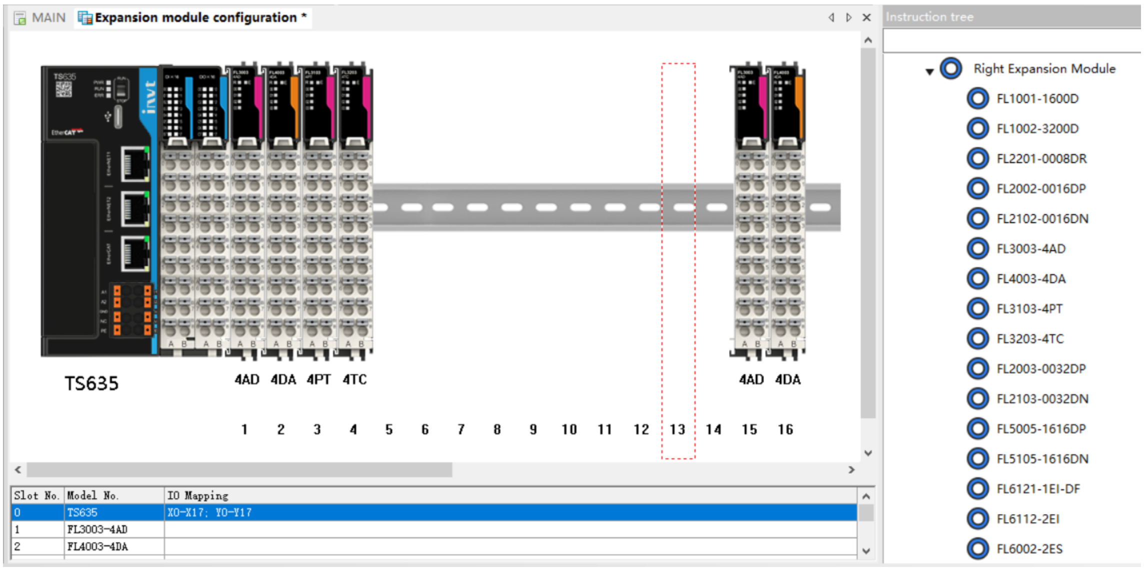
  When configuration mismatches occur, you can check whether the actual mounted and the configured ones are consistent.
- The _sExtModule.sExtSlot[n].Error variable feeds back warnings and errors in the module, and the specific error types can be fed back from the system variable _sCurErrList.

16.1.3 Expansion Module Configuration

1. Double-click "Setting" > "Expansion module configuration" in Auto Station Pro.



2. Switch to the expansion module configuration interface, double-click the module on the right side to automatically expand it on the expansion rack, or drag it to the expansion rack with the left mouse button.

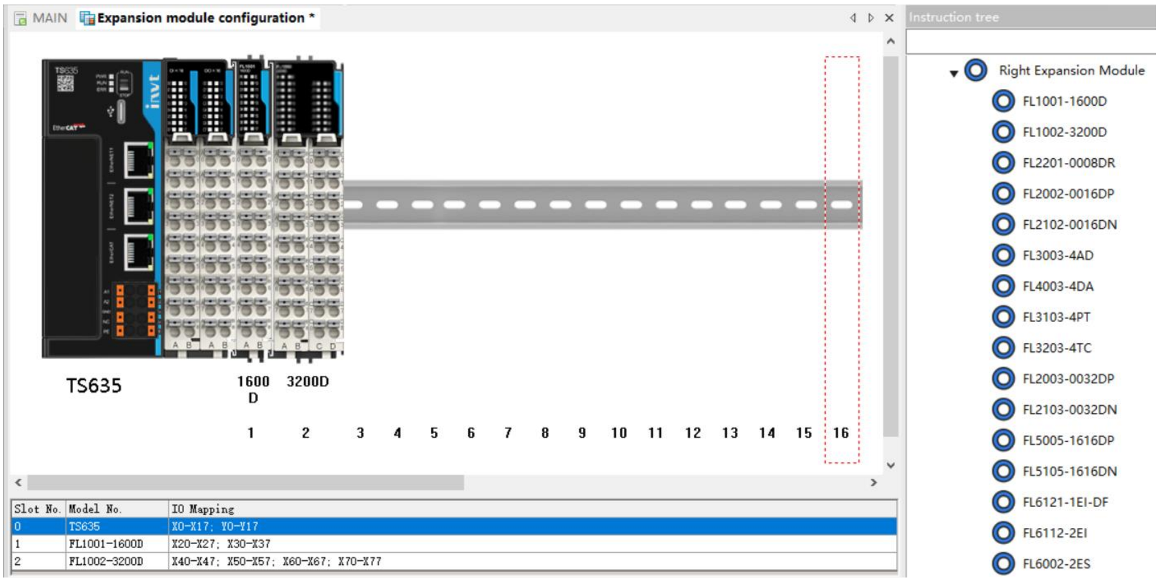


16.1.4 Expansion Module Configuration

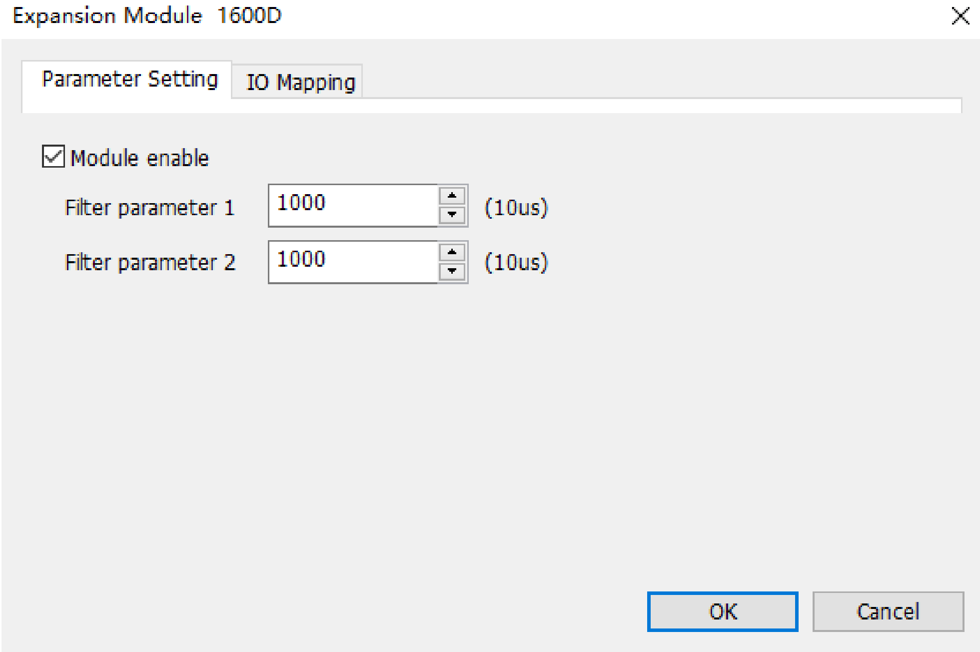
16.1.4.1 Digital Input Module

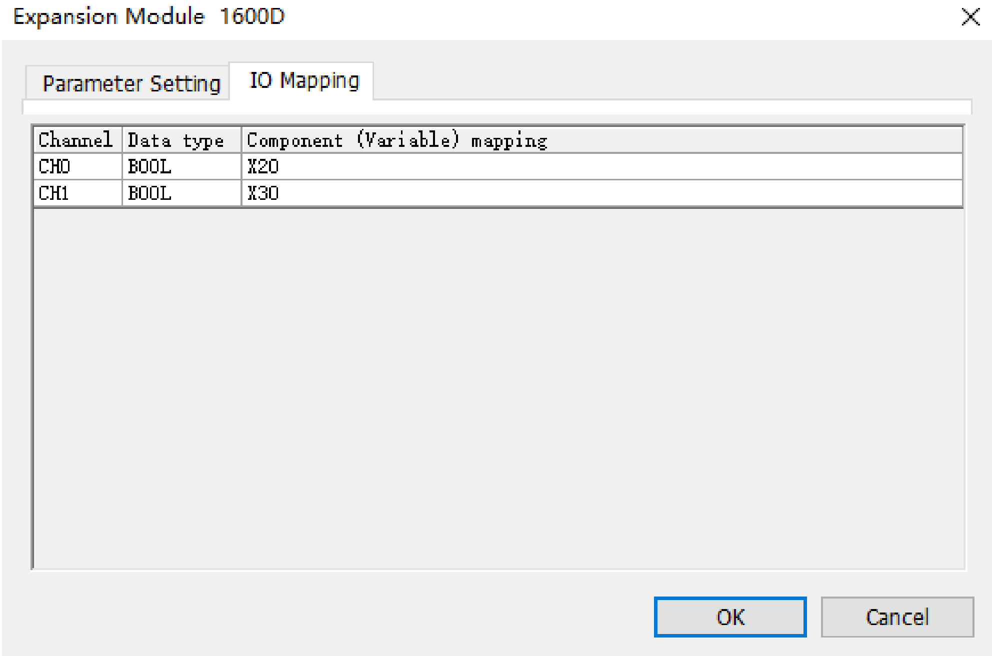
The digital input modules include FL1001-1600DI and FL1002-3200DI. As local expansion modules, the specific method to use is as follows:

Step 1 In the module list, select the module you want to add, double-click the module to automatically expand it on the expansion rack, or drag it to the expansion rack with the left mouse button.



Step 2 When the local digital input expansion module is connected to the main module, double-click the module to pop up the module configuration interface, where you can configure the filter parameters and IO mapping according to the actual demand. If the mapping variable is not manually configured by the user, the input X port number on the expansion module is followed by the X port number on the main module or the previous digital input module, and is numbered in turn.





Note:

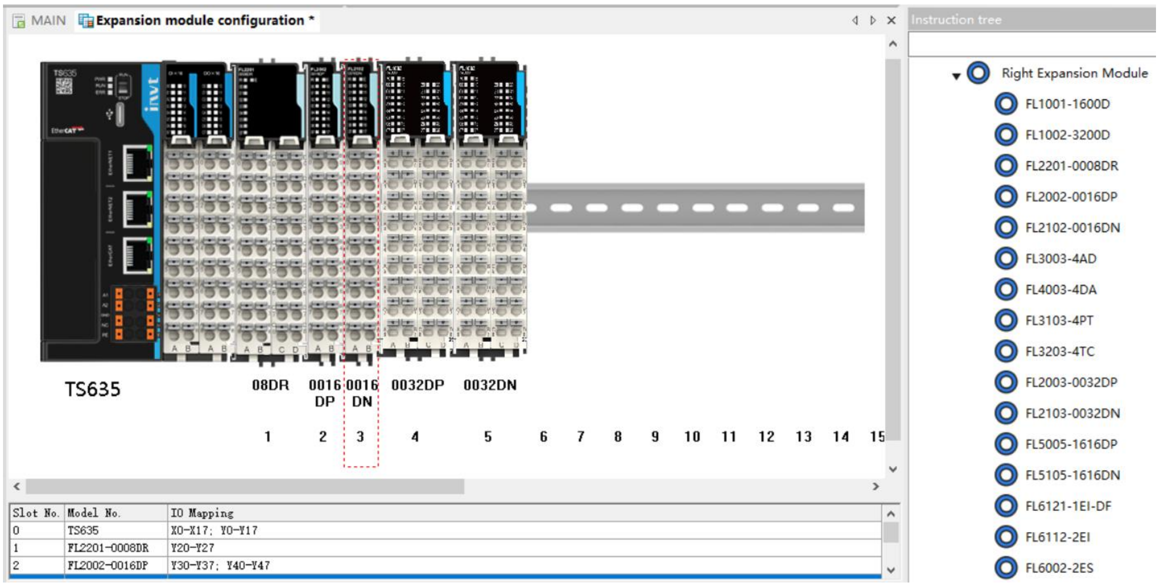
- Every 8 points in the input module share a filter parameter, the unit of this filter parameter is 10µs, and 1,000 means 10ms.
- The subscript of the IO mapping bit element needs to be a multiple of 010 (octal), for example, X20, X30, X110.

Step 3 You can program with mapped variables in your program.

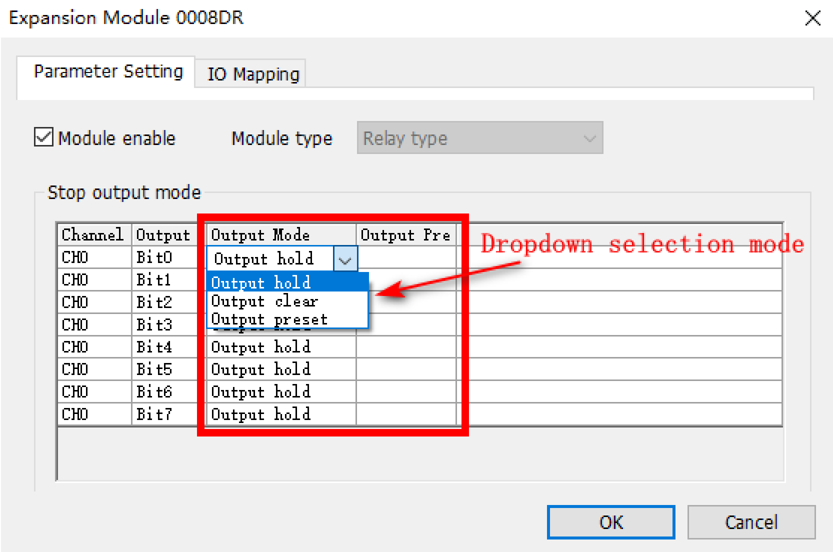
16.1.4.2 Digital Output Module

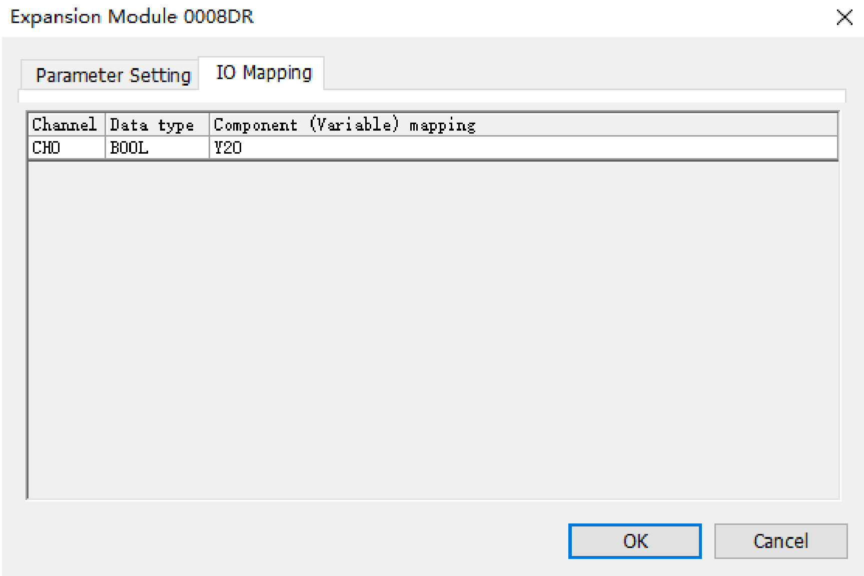
The digital output modules include FL2201-0008DR, FL2102-0016DN, FL2002-0016DP, FL2103-0032DN and FL2003-0032DP. As local expansion modules, the specific method to use is as follows:

Step 1 In the module list, select the module you want to add, double-click the module to automatically expand it on the expansion rack, or drag it to the expansion rack with the left mouse button.



Step 2 When the local digital output expansion module is connected to the main module, double-click the module to pop up the module configuration interface, where you can configure the module type, post-stop output method and IO mapping according to the actual demand. If the mapping variable is not manually configured by the user, the input Y port number on the expansion module is followed by the Y port number on the main module or the previous digital output module, and is numbered in turn.





Note:

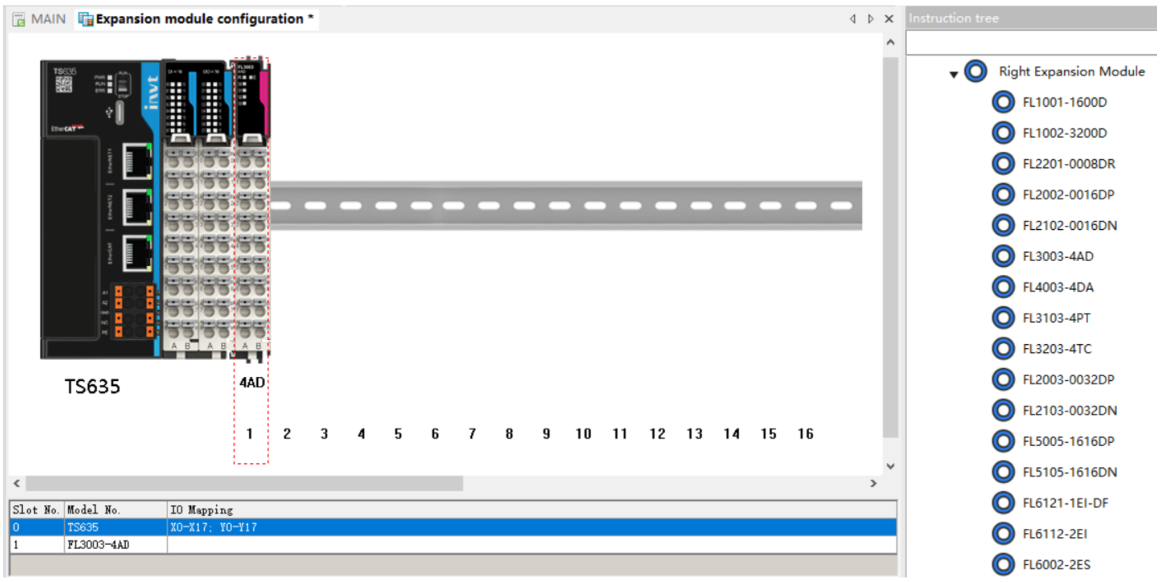
- The subscript of the IO mapping bit element needs to be a multiple of 010 (octal), for example, Y20, Y30, Y110.
- After configuring the post-stop output mode, the PLC will output according to the configured STOP output mode and STOP output preset value after entering the STOP state.

Step 3 You can program with mapped variables in your program.

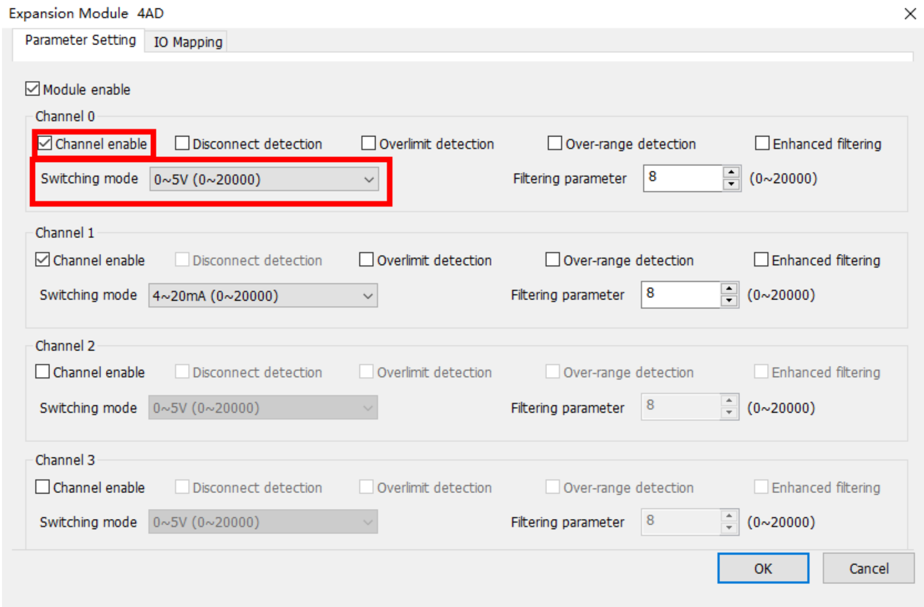
16.1.4.3 Analog Input Module

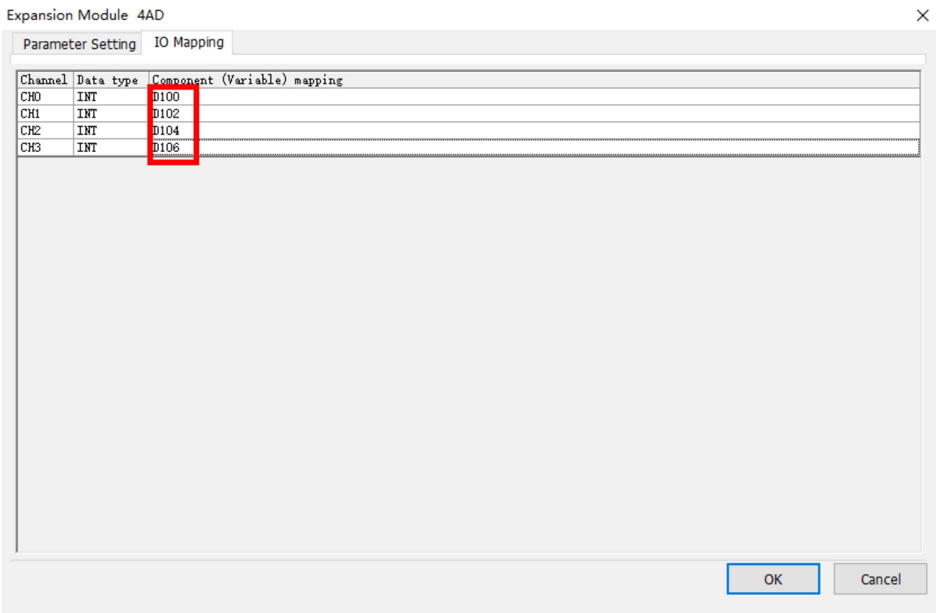
The analog input module includes FL3003-4AD. As local expansion modules, the specific method to use is as follows:

Step 1 In the module list, select the module you want to add, double-click the module to automatically expand it on the expansion rack, or drag it to the expansion rack with the left mouse button.



Step 2 When the local analog input expansion module is connected to the main module, double-click the module to pop up the module configuration interface, and configure the channel setting, conversion mode and IO mapping according to the actual demand.





Note:

- Channel Enable: Channel enable disabled for unused channels to save scanning time.
- When the channel interference is serious, the enhanced filtering is preferentially enabled, and then the filtering parameters are adjusted. The larger the filtering parameters, the better the filtering effect and the larger the corresponding lag.
- Disconnection detection is not supported when the switching mode is current mode.
- Overrun detection: When the external sampling signal exceeds ± 25V, the module outputs a warning, the Error of the module is set to 1 in the system variables, and an error code can be read in the system variable error list.
- Overrange detection: When the external sampling signal exceeds the range of the conversion mode, the module outputs a warning, the Error of the module is set to 1 in the system variables, and an error code can be read in the system variable error list.

Step 3 You can program with mapped variables in your program.

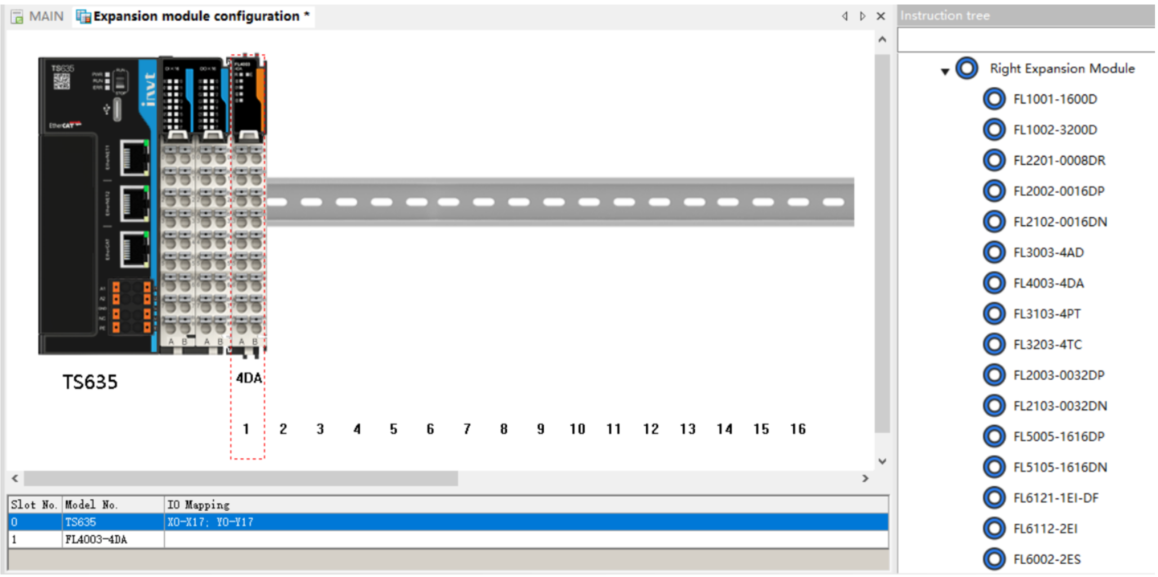
The correspondence between the mapping and the actual input analog value is as follows:

Input Type	Enter the Rated Range	Rated Corresponding Digital Quantity
Analog voltage input	-10V–10V	-20000–20000
	0V–10V	0–20000
	-5V–5V	-20000–20000
	0V–5V	0–20000
Analog current input	-20mA–20mA	-20000–20000
	0mA–20mA	0–20000
	4mA–20mA	0–20000

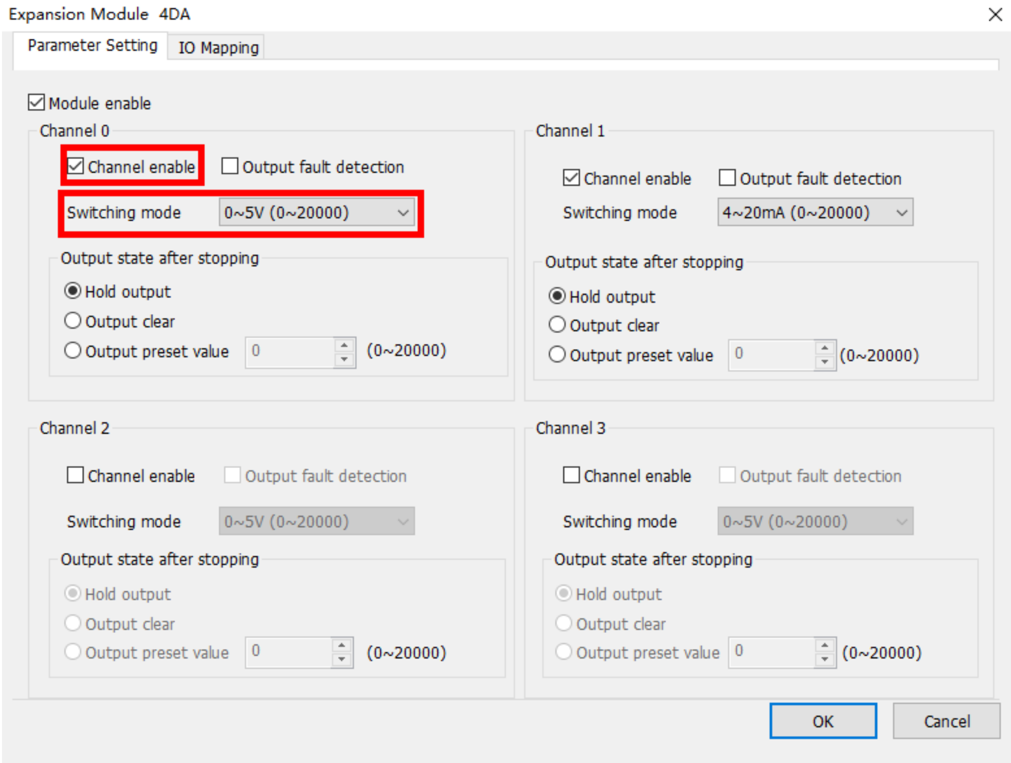
16.1.4.4 Analog Output Module

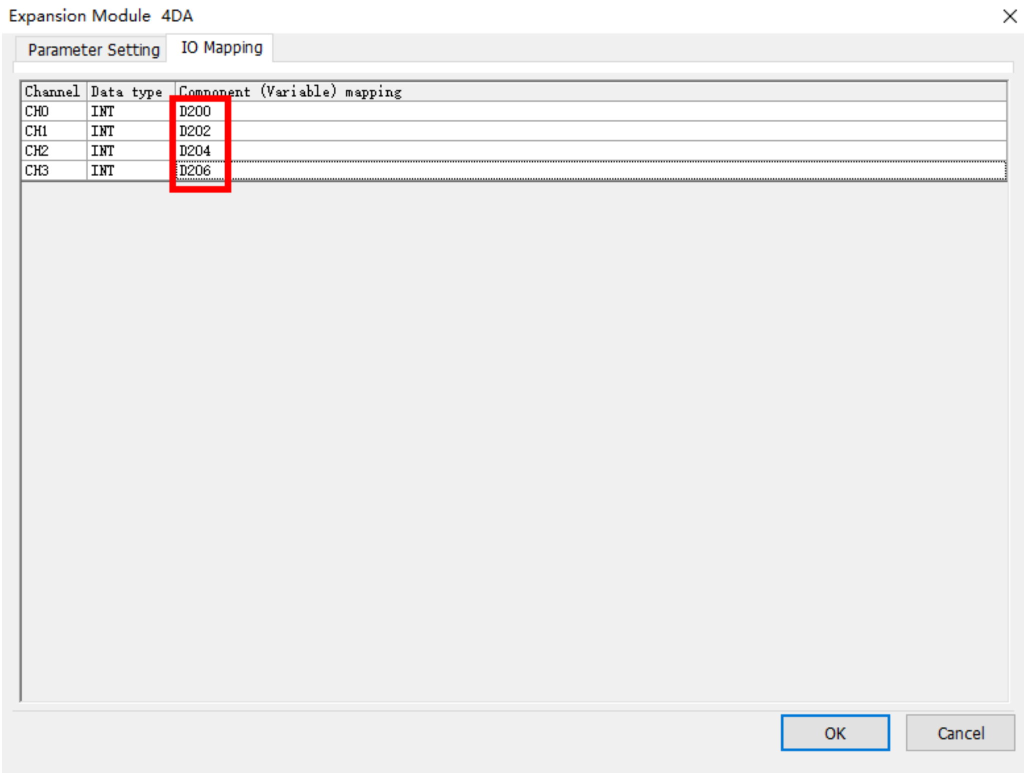
The analog output module includes FL4003-4DA. As local expansion modules, the specific method to use is as follows:

Step 1 In the module list, select the module you want to add, double-click the module to automatically expand it on the expansion rack, or drag it to the expansion rack with the left mouse button.



Step 2 When the local analog output expansion module is connected to the main module, double-click the module to pop up the module configuration interface, and configure the channel setting, conversion mode and IO mapping according to the actual demand.





Note:

- Channel Enable: Channel enable disabled for unused channels to save scanning time.
- After configuring the post-stop output mode, the PLC will output according to the configured STOP output mode and STOP output preset value after entering the STOP state.

Step 3 You can program with mapped variables in your program.

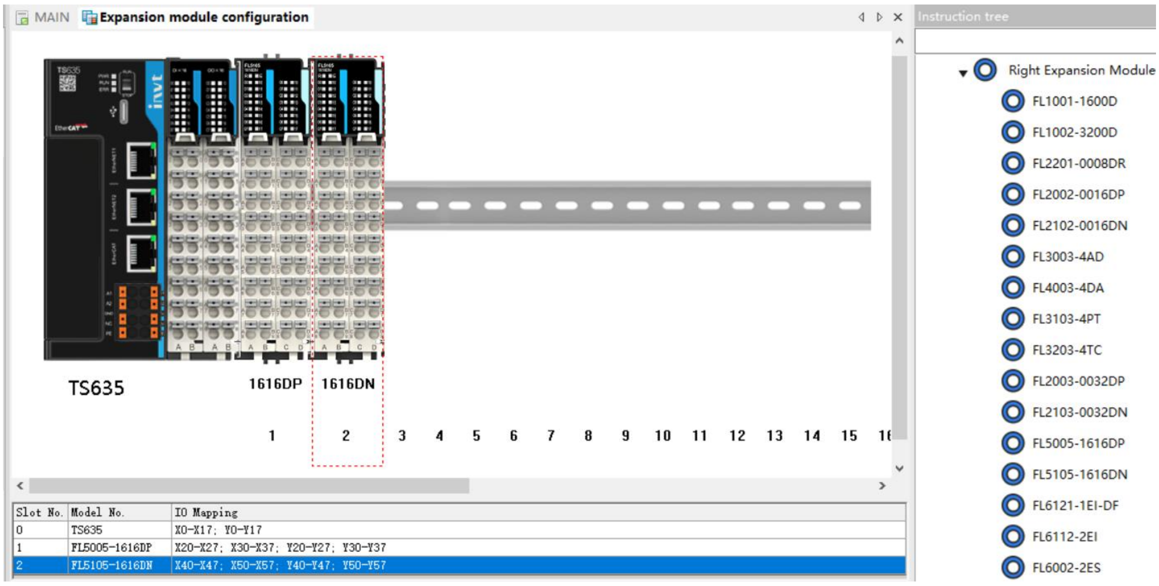
The correspondence between the mapping and the actual output analog value is as follows:

Input Type	Rated Output Range	Rated Corresponding Digital Quantity
Analog voltage output	-10V–10V	-20000–20000
	0V–10V	0–20000
	-5V–5V	-20000–20000
	0V–5V	0–20000
Analog current output	0mA–20mA	0–20000
	4mA–20mA	0–20000

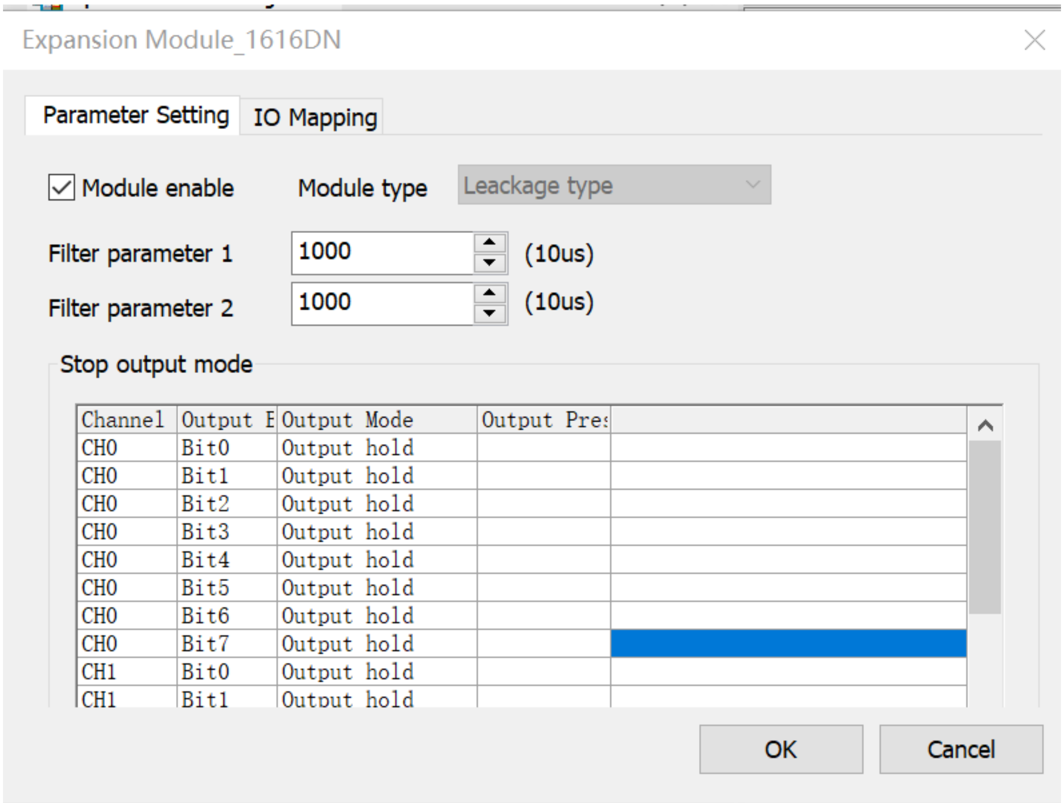
16.1.4.5 Hybrid Module

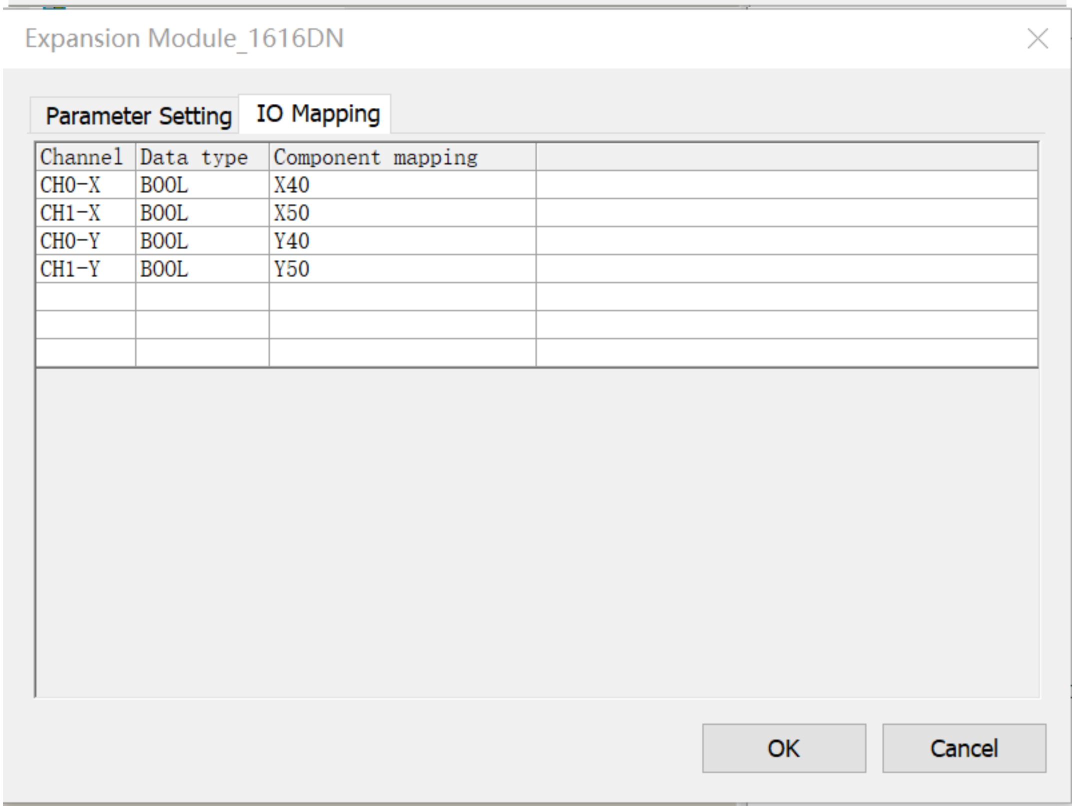
The hybrid module includes FL5005-1616DP and FL5105-1616DN. As local expansion modules, the specific method to use is as follows:

Step 1 In the module list, select the module you want to add, double-click the module to automatically expand it on the expansion rack, or drag it to the expansion rack with the left mouse button.



Step 2 When the hybrid expansion module is connected to the main module, double-click the module to pop up the module configuration interface. In the "Parameter Setting" interface, configure the "Filter parameter 1," "Filter parameter 2," and "Stop output mode" according to actual needs. In the "IO Mapping" interface, configure the "IO Mapping."

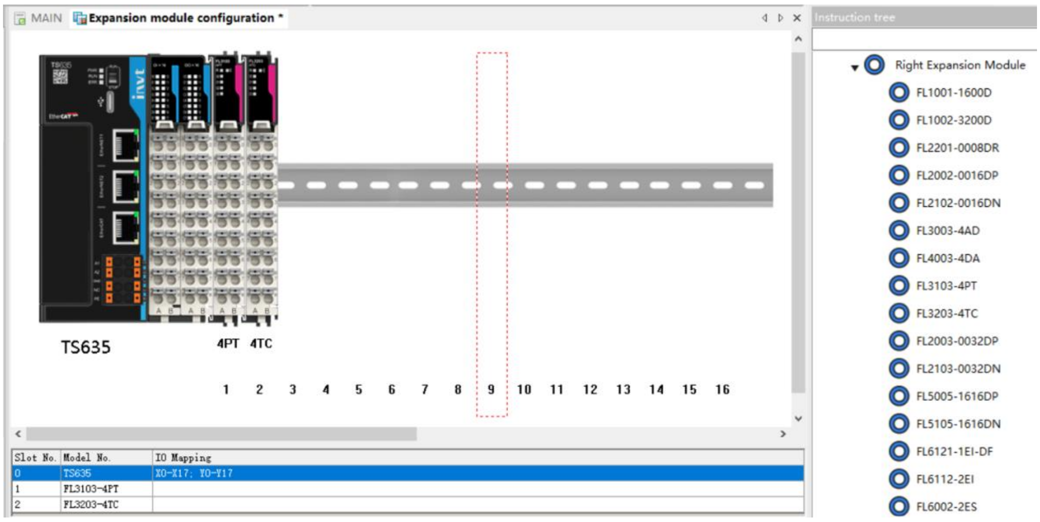




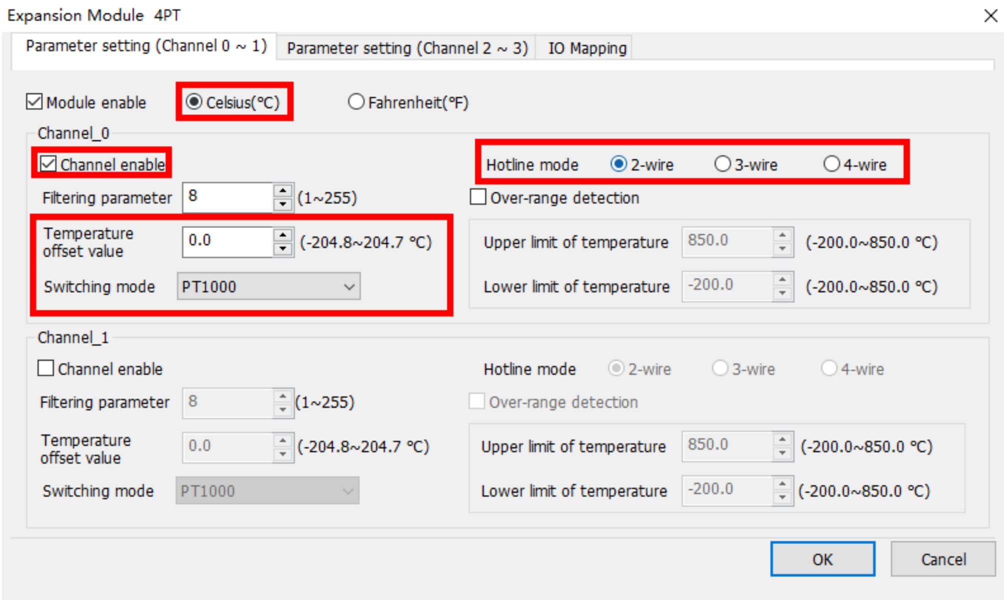
16.1.4.6 Temperature Sampling Module

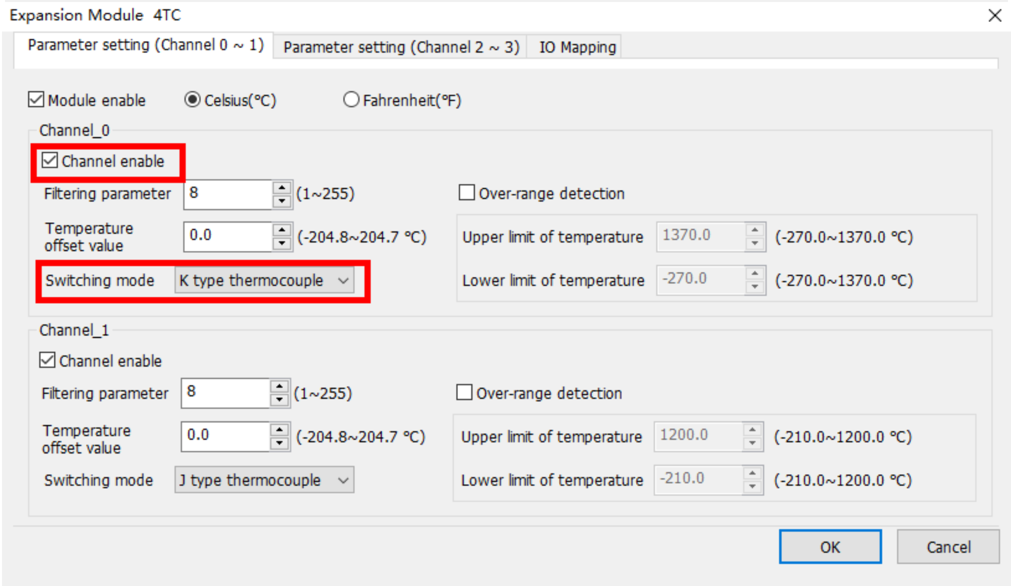
The temperature sampling modules include FL3101-4PT and FL3201-4TC. As local expansion modules, the specific method to use is as follows:

Step 1 In the module list, select the module you want to add, double-click the module to automatically expand it on the expansion rack, or drag it to the expansion rack with the left mouse button.



Step 2 When the local temperature sampling expansion modules are connected to the main module, double-click the module to pop up the module configuration interface, and configure the temperature unit, channel setting, conversion mode and IO mapping according to the actual demand.





Note:

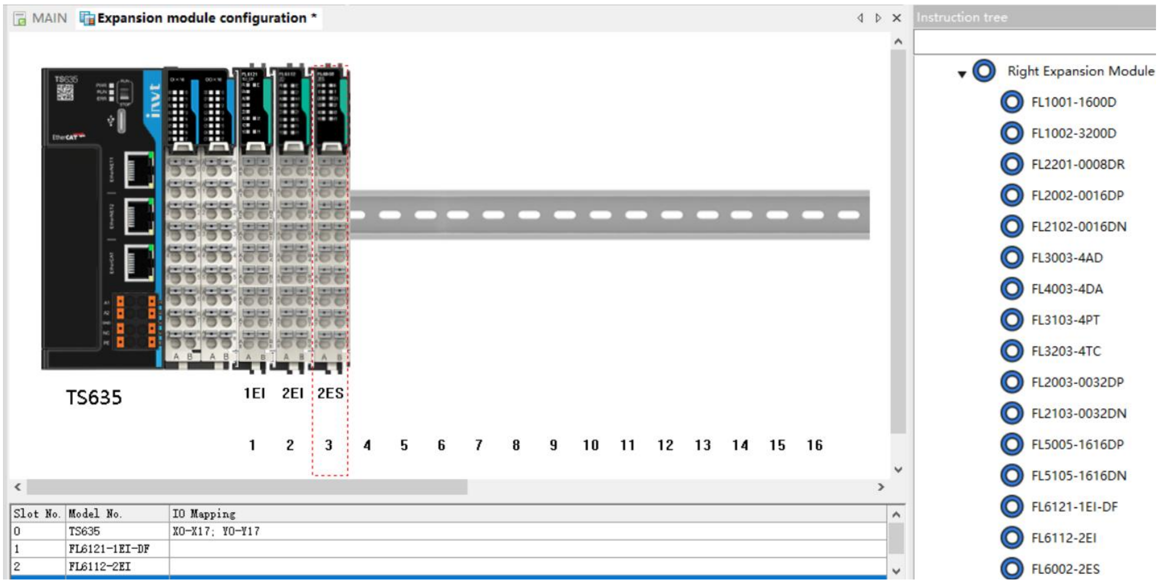
- Channel Enable: Channel enable disabled for unused channels to save scanning time.
- The upper and lower temperature limits are related to the sensor type, and the corresponding upper and lower temperature limits need to be adjusted after the conversion mode is configured.
- The temperature offset value is used to correct the measurement error caused by the internal heating of the module.
- The wiring of PT modular thermal resistance needs to be consistent with the actual wiring mode, otherwise the measurement will have errors.

Step 3 In the program, the mapped variables can be used for programming, and the values measured by the temperature measurement module are floating-point numbers.

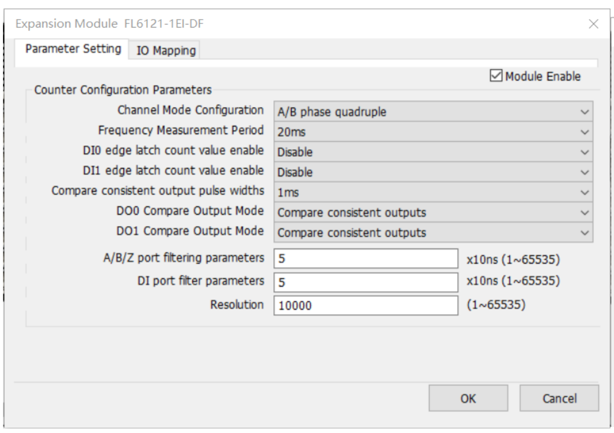
16.1.4.7 Counter module

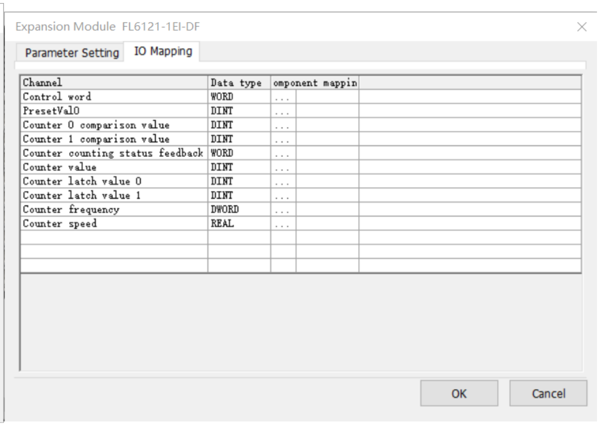
The counter modules include FL6121-1EI-DF, FL6112-2EI and FL6002-2ES. As local expansion modules, the specific method to use is as follows:

Step 1 In the module list, select the module you want to add, double-click the module to automatically expand it on the expansion rack, or drag it to the expansion rack with the left mouse button.



Step 2 When the counter expansion modules are connected to the main module, double-click the module to pop up the module configuration interface, and configure the parameters and IO mapping according to the actual demand.

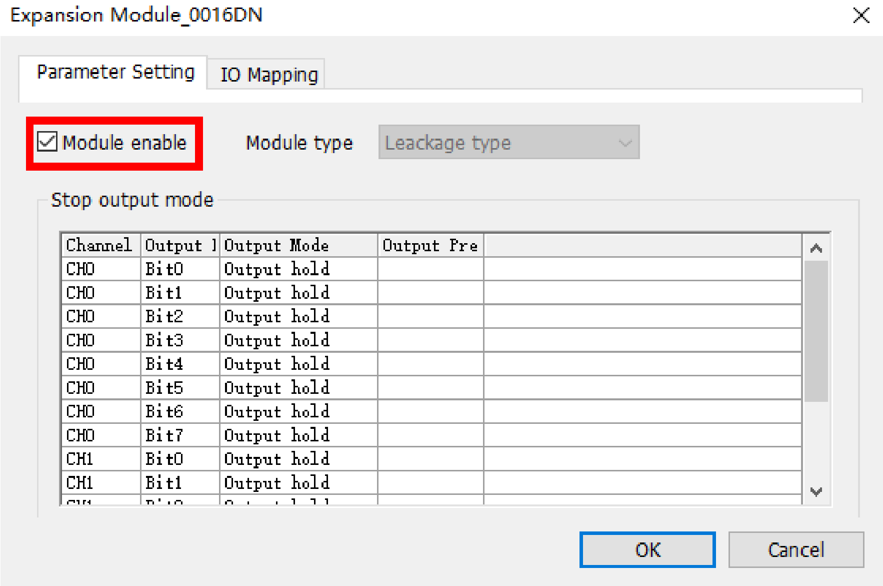




16.2 Basic Operation of Local Expansion Module

16.2.1 Module Enable

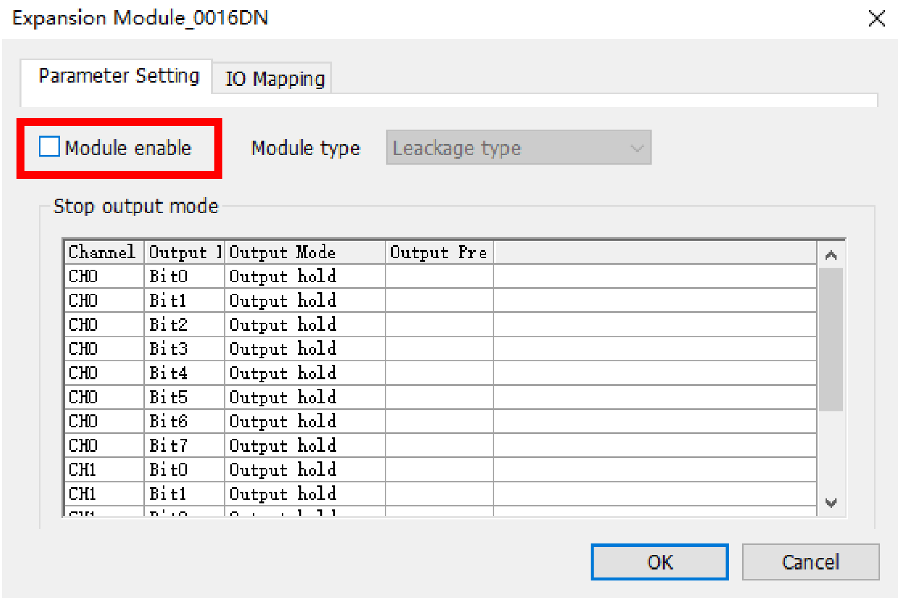
1. In the module list, select the module to be enabled, double-click it, and check "Module enable" in the pop-up configuration interface.



2. It will take effect after compilation without error and after the program is successfully downloaded.

16.2.2 Module Disable

1. In the module list, select the module to be disabled, double-click it, and uncheck "Module enable" in the pop-up configuration interface.



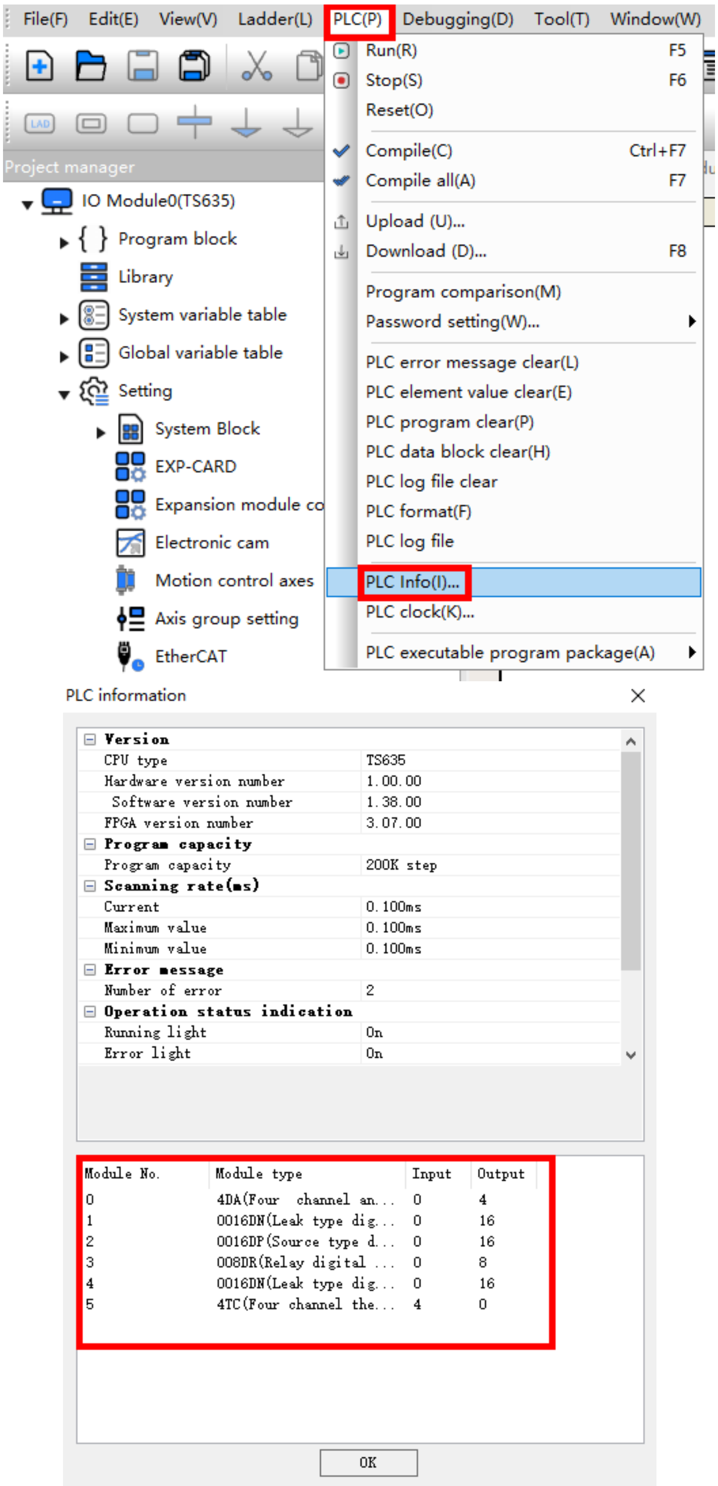
2. It will take effect after compilation without error and after the program is successfully downloaded.

Note: When the local expansion module is disabled, it is also necessary to ensure that the actual physical connection and configuration are consistent.

16.2.3 Get Physical Configuration

After TS series PLC is connected to the expansion module, the actual number of connected modules and configuration sequence can be obtained from PLC information.

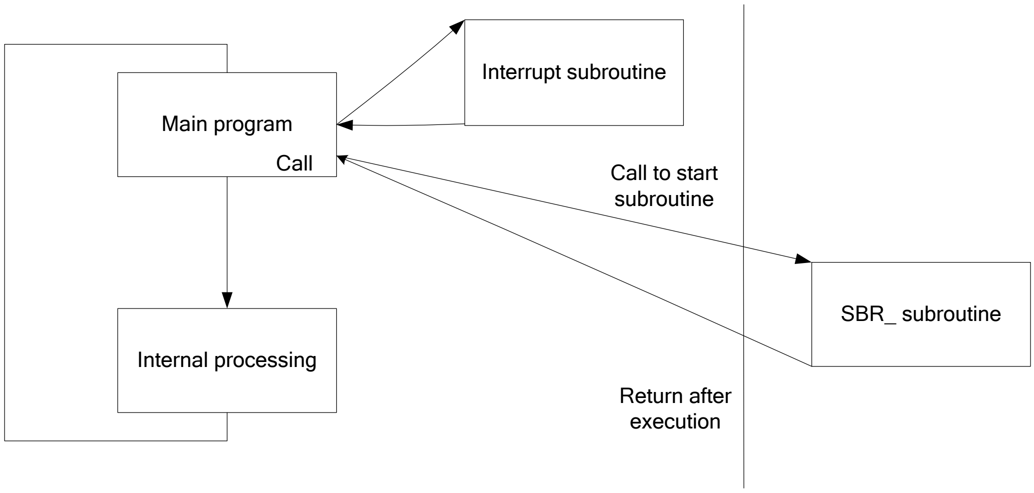
Click "PLC" > "PLC Info" on the toolbar, and check the actual physical configuration in the pop-up PLC information box.



17 Subroutine, Library

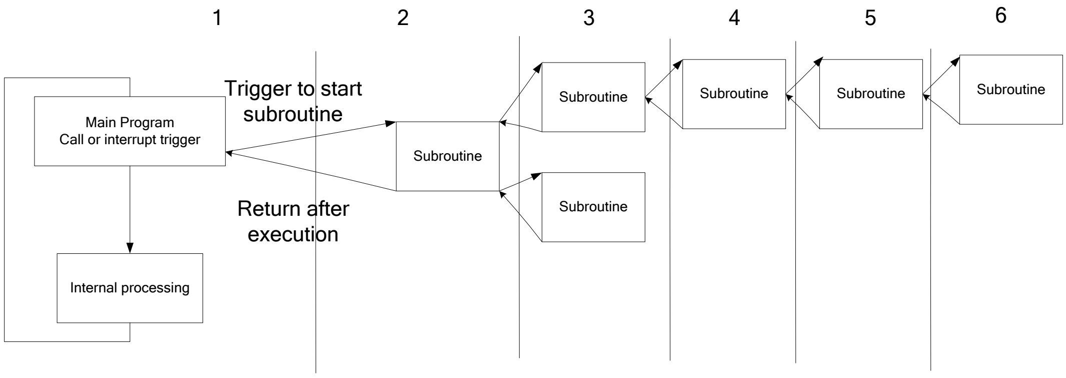
17.1 Subroutine

The execution logic and cyclic scanning mode of main program and subroutines are shown in the following figure.



Subroutine nesting layer

Subroutines support up to 6 layers of nesting, with the main program calling the subroutine as the first layer. A layer of nesting is added for each call. If the nest returns, the nesting layer is not increased. See the figure below.



17.2 General Subroutine Application

A subroutine is an independent program body, which can be called by the main program or other subroutines. Subroutines are optional components of user programs.

Writing user programs using subroutines has the following advantages:

- It can reduce the size of the user program, the repeated user program code segment with the same function, and can be written as a subroutine repeatedly called.
- It makes the structure of the program clearer, especially simplifies the structure of the main program.
- It improve the portability of user programs.

17.2.1 Matters Needing Attention in the Use of General Subroutines

When writing or calling subroutines, you should pay attention to the following matters:

- The mosaic call of subroutines is supported, and the maximum number of mosaic call layers is 6.

  The following example demonstrates a legal 6-layer mosaic call relationship:

  MAIN → SBR1 → SBR2 → SBR3 → SBR4 → SBR5 → SBR6.

  (It represents calling the corresponding subroutine using the CALL instruction)

- Recursive and circular calls to subroutines are not supported.

  The following two examples demonstrate illegal subroutine call relationships:

  ✧ MAIN → SBR0 → SBR0 (recursive call, illegal)

  ✧ MAIN → SBR0 → SBR1 → SBR0 (loop call, illegal)

- A user program can define up to 64 subroutines.
- A subroutine can define up to 16 bit-type and 16 WORD variables in the variable table.
- When calling a subroutine, it should be noted that the attributes of operands filled in the CALL instruction should match the attributes of variables defined in the variable table of the subroutine, and the compiler will check the correctness of the matching.
- No subroutine is allowed to be called in the interrupt program.

17.2.2 Definition of General Subroutine Variable Table

Attributes of Subroutine Variables

The subroutine variable table is to declare interface parameters and local variables (collectively referred to as variables) of subroutines, and specify their use attributes.

Description of Attribute Items of Subroutine Variables

The interface parameters and local variables (collectively called variables) of subroutines have the following attributes:

- Variable address

  Each subroutine interface parameter or local variable is assigned with a fixed LM element or V element address. The address is automatically assigned to subroutine interface parameters or local variables by programming software according to the data type of variables and the principle of address continuity.

- Variable

  You can name a variable (alias) of subroutine interface parameters or local variables, and you can use this variable name in programs by reference to it.

- Variable type

  Subroutine interface parameters or local variables are divided into IN type, OUT type, IN_OUT type and TEMP type:

  ✧ An IN variable is used to pass the input value of a subroutine when it is called.

  ✧ An OUT variable is used to pass the call return value for a subroutine when it returns.

  ✧ An IN_OUT variable is used to pass the input value when the subroutine is called. When the subroutine returns, it is used to pass the call return value.

  ✧ An TEMP type variable is used only as a valid local variable within the scope of subroutines.

- Variable data type

  The variable data type attribute specifies the data width and data scope of variables. The following table lists the variable data types:

Table 17-1 Variable Data Type

Variable Data Type	Data Type Description	Occupy LM/V Element Address
BOOL	Bit-type variable	It occupies 1 LM element address
INT	Signed integer variable	It occupies 1 V element address
DINT	Signed long integer variable	It occupies 2 consecutive V element addresses
WORD	Unsigned integer variable	It occupies 1 V element address
DWORD	Unsigned long integer variable	It occupies 2 consecutive V element addresses
REAL	Floating-point variable	It occupies 2 consecutive V element addresses

17.2.3 General Subroutine Parameter Passing

When calling a subroutine in the main program, if local input and output variables are defined in the subroutine, the interface parameters of the subroutine should be filled with corresponding numerical values or global/temporary variable elements.

Note: The data types of local variables and interface parameters should be consistent.

17.2.4 Examples of Using General Subroutines

Here is an example to show how to write and call subroutines

Introduction to Sample Functions

Call the subroutine SBR_1 in the main program, and allow the subroutine SBR_1 to complete the addition operation (3 +2) of two integer constants, and assign the operation result 5 to D0.

Example Operation Steps

Step 1 Create a subroutine in the project and name it SBR_1.

Step 2 Write the subroutine SBR_1.

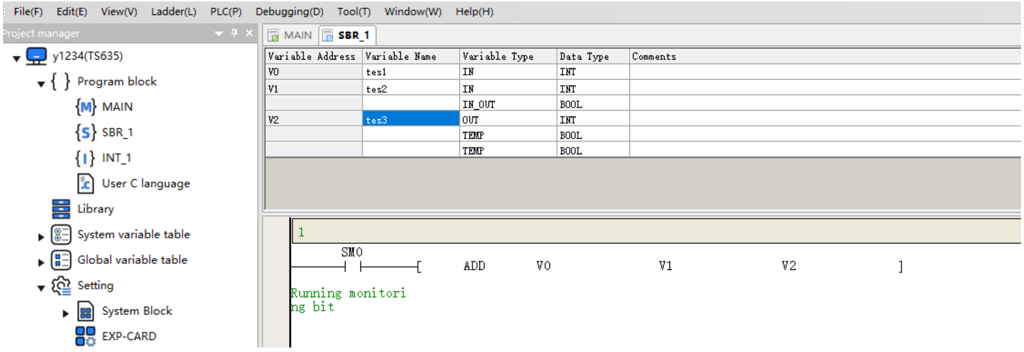
A. Establish a call operand interface of subroutine in the variable table of subroutine SBR_1.

- Define variable 1: Name the variable IN1, which is an IN-type parameter and is used as INT-type data, and it is assigned a V-element address V0 in sequence.
- Define variable 2: Name the variable IN2, which is an IN-type parameter and is used as INT-type data, and it is assigned a V-element address V1 in sequence.
- Define variable 3: Name the variable OUT1, which is an OUT-type parameter and is used as INT-type data, and it is assigned a V-element address V2 in sequence.

B. Write the implementation code of subroutine SBR_1:

```
LD SM0
ADD # IN1 # IN2 # OUT1
```

The following figure illustrates the process of writing subroutine SBR_1:



Step 3 Write the main program and call the subroutine.

In the main program, use CALL instruction to call subroutine SBR_1.

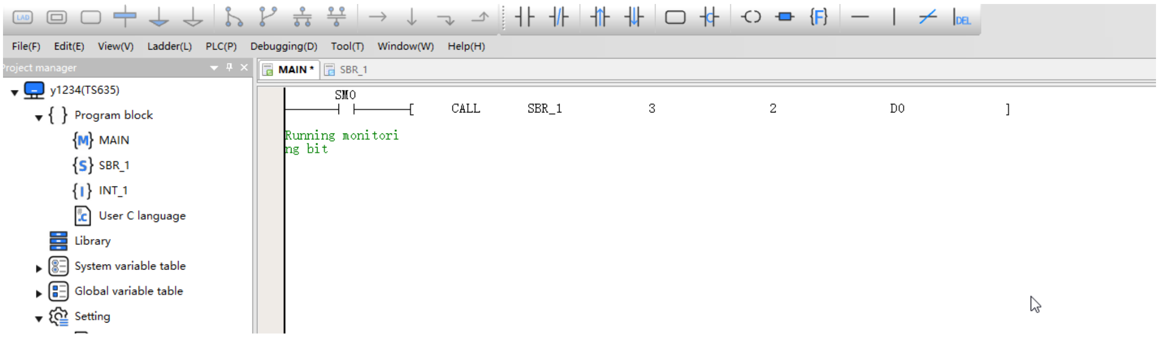
The code of the whole main program is as follows:

```
LD M0
CALL SBR_1 3 2 D0
```

You can use parameters to pass the corresponding relationship table and fill in the parameters brought in or returned when calling the subroutine.

- Bring in the parameter IN1 and pass the constant integer 3.
- Bring in the parameter IN2 and pass the constant integer 2.
- The return value OUT1 is passed to D0

See the picture below:



Step 4 Compile, download and run the user program to verify the logical correctness of the subroutine.

- Sample execution results

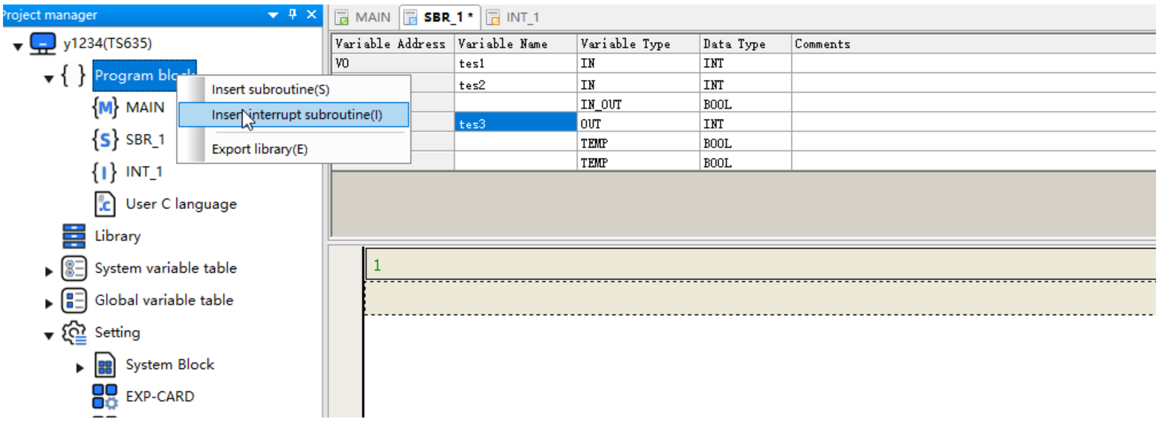
When M0=ON, the subroutine SBR_1 is called, and the operands IN1 and IN2 are passed with values 3 and 2, and then the addition operation returns a value 5, and finally D0=5.

17.3 Interrupt Subroutine Application

17.3.1 External Interrupt Subroutine

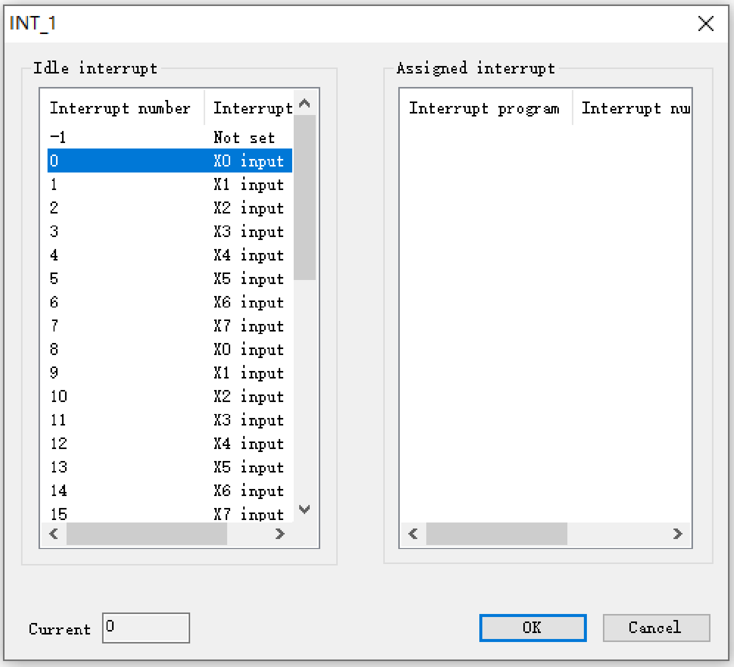
External interrupt subroutine is suitable for the scenario where the external input signal triggers and the program needs to be executed immediately to respond. External interrupt subroutines are not affected by scanning periods. The specific application steps are as follows:

Step 1 Click "Project manager", and right-click "Program block" to choose "Insert interrupt subroutine".

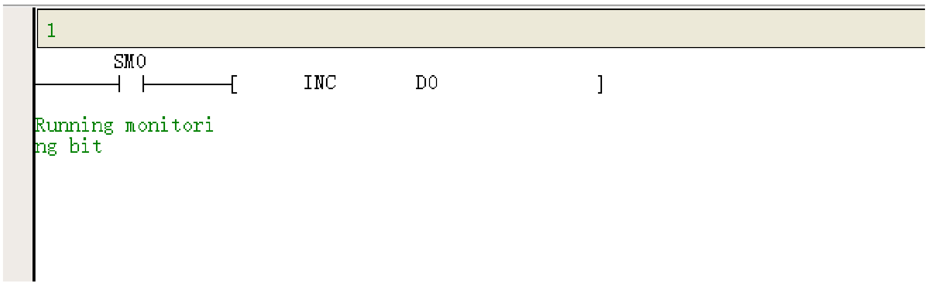


Step 2 Right-click the inserted interrupt subroutine (INT_001 in the figure) and select "Attribute" to open the interrupt subroutine setting page as shown in the following figure.

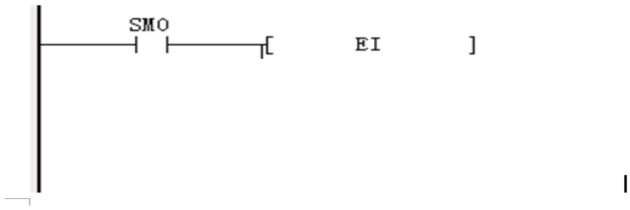
Step 3 Click the icon  after the "Interrupt event" field to enter the interrupt selection interface. Select the corresponding external interrupt. For example, after selecting X0 input interrupt, you need to select the attributes of interrupt, including rising edge and falling edge.



Step 4 Write interrupt subroutines in INT_001.



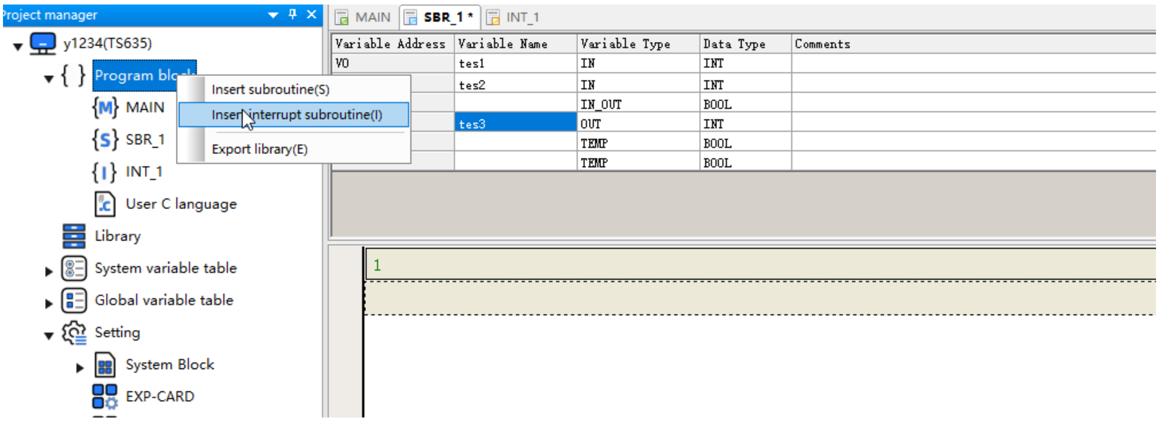
Step 5 Start EI in the main program, and when the external interrupt condition is met, the corresponding interrupt subroutine will be triggered to execute.



17.3.2 Timer Interrupt Subroutine

The specific application steps are as follows:

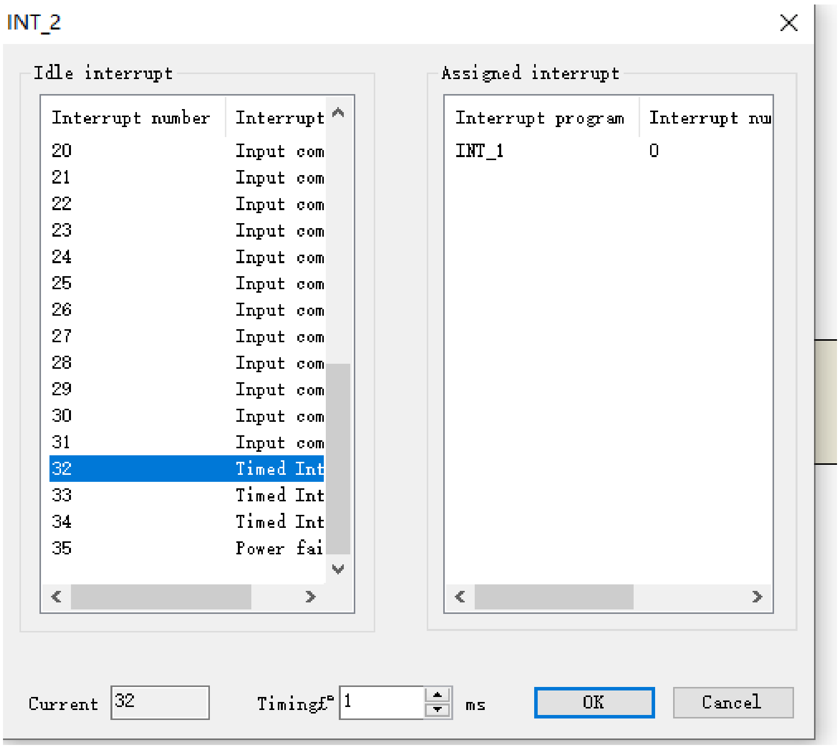
Step 1 Click "Project manager", and right-click "Program block" to choose "Insert interrupt subroutine".



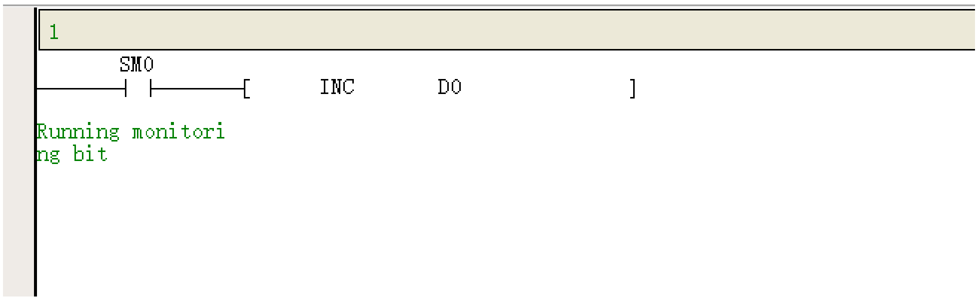
Step 2 Right-click the inserted interrupt subroutine (INT_001 in the figure) and select "Attribute" to open the interrupt subroutine setting page as shown in the following figure.

Step 3 Click the icon after the "Interrupt event" field to enter the interrupt selection interface.

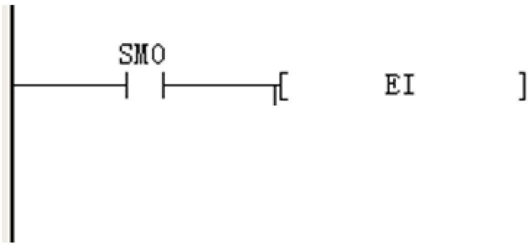
Step 4 Select the timer interrupt, and select the timing duration of the timer interrupt in ms, ranging from 1 to 1000ms.



Step 5 After that, you can write interrupt subroutines in INT_001.



Step 6 Start EI in the main program, and the corresponding interrupt subroutine will be triggered to execute when the timer interrupt condition is met.



18 Application of Custom Variable Communication

18.1 Overview

The address of the variable in the custom variable function is a dynamically changing address, so the external devices cannot access it directly through the address. Currently, custom variables can be accessed in two ways:

Method 1: Bind custom variables to soft elements. Achieving Modbus communication between external devices and TS600 series controllers through Modbus address mapping.

Method 2: Tag communication. Tag communication is a proprietary protocol for communication between our TS600 series controllers and our touch screens, allowing users to efficiently and quickly access the internal variables of TS600 series controllers through the touch screen.

18.2 Example Project

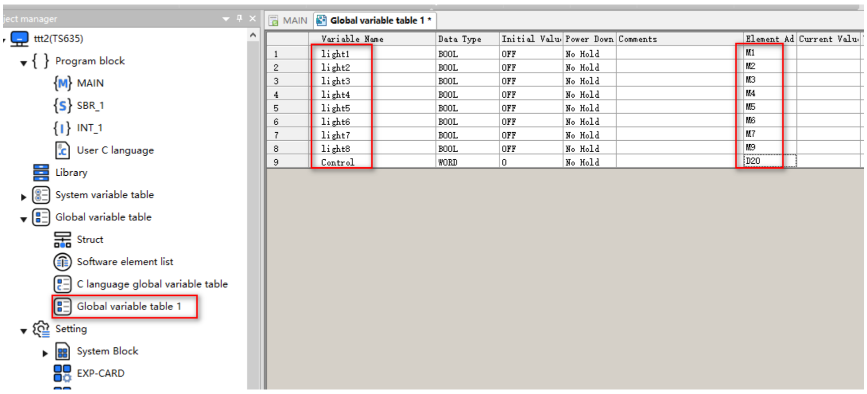
Write a TS600 series bit operation program, and display the corresponding bit elements and control word status on the INVT touch screen.

18.2.1 Accessing Custom Variables via Binding to Elements

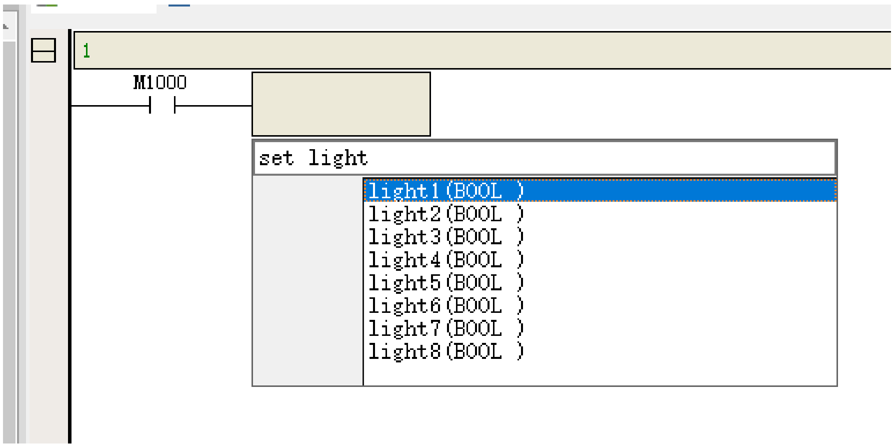
18.2.1.1 PLC Programming

Step 1 Create a new PLC project, see 2.4 Programming and Debugging for details.

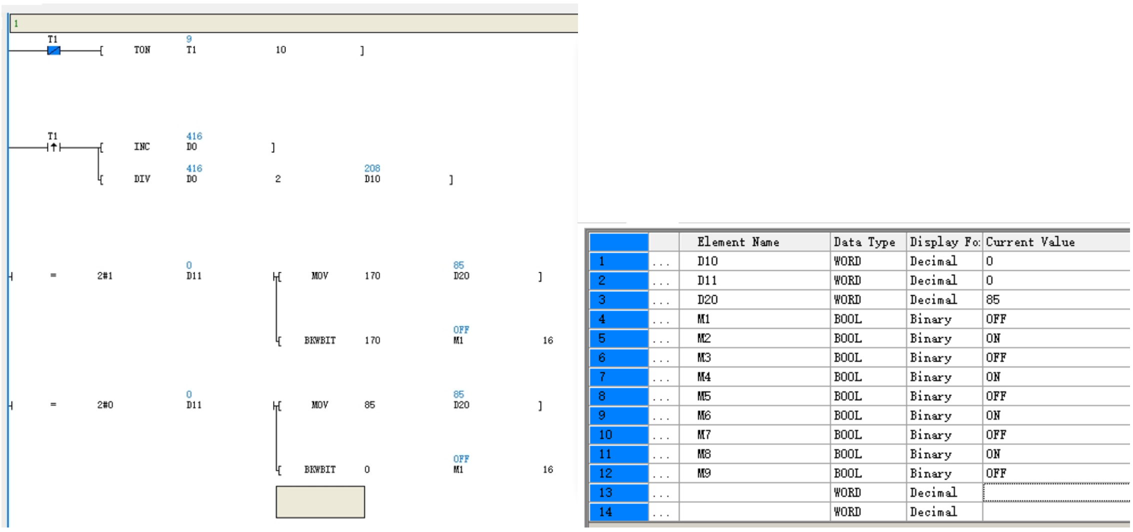
Step 2 Assign variable address, double-click "Global variable table 1" in the project management bar to enter the global variable table interface, and assign soft element address for custom variables. Variables light1 – light8 are BOOL-type arrays, and will occupy a total of 8 bit elements, namely M1 – M7 if chosen to map to M1.



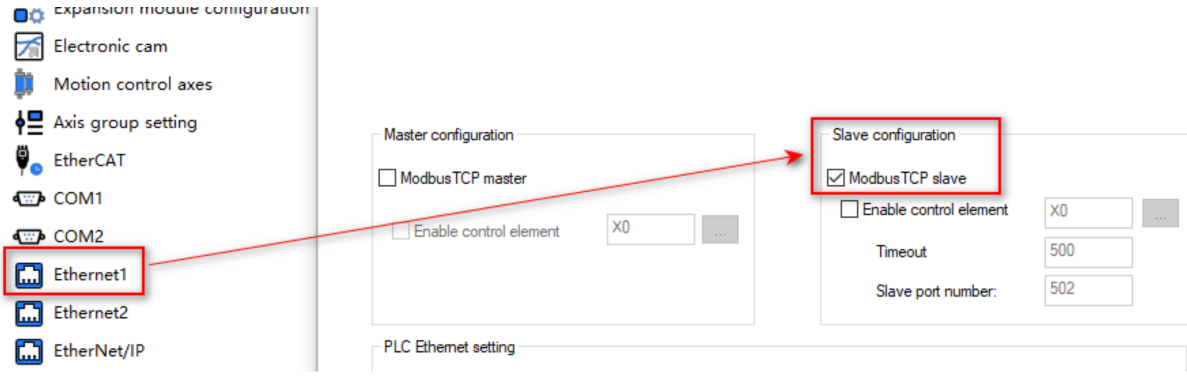
Step 3 The program is as shown in the following figure, in which light1 – light8 are the eight BOOL variables created in the previous step, and Control is the WORD variable for HMI display.



Elements are used directly for custom variables, so in actual use, the interface displays the element name.

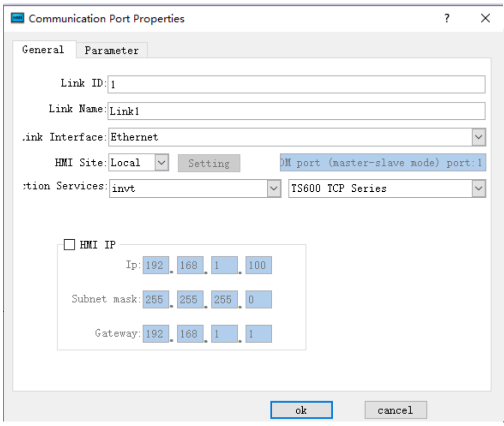


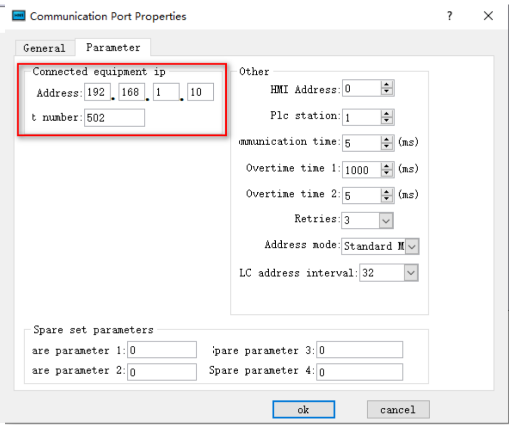
Step 4 For communication configuration, double-click "Ethernet 1" to check "ModbusTCP slave" communication in slave configuration.

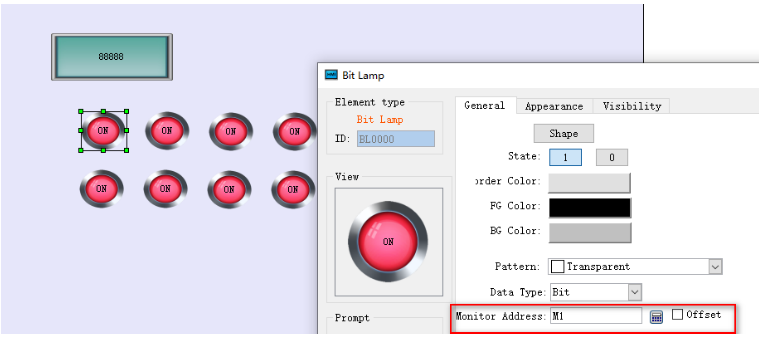


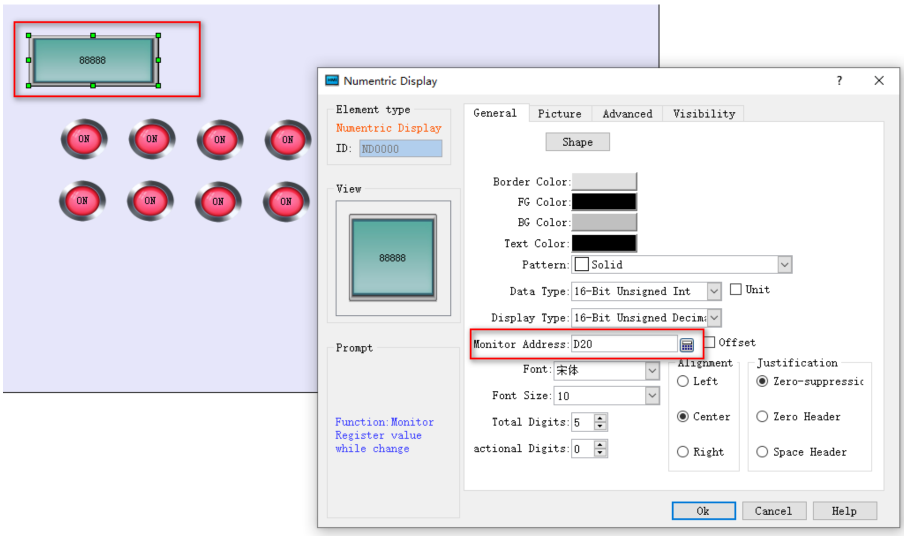
18.2.1.2 Touch Screen Programming

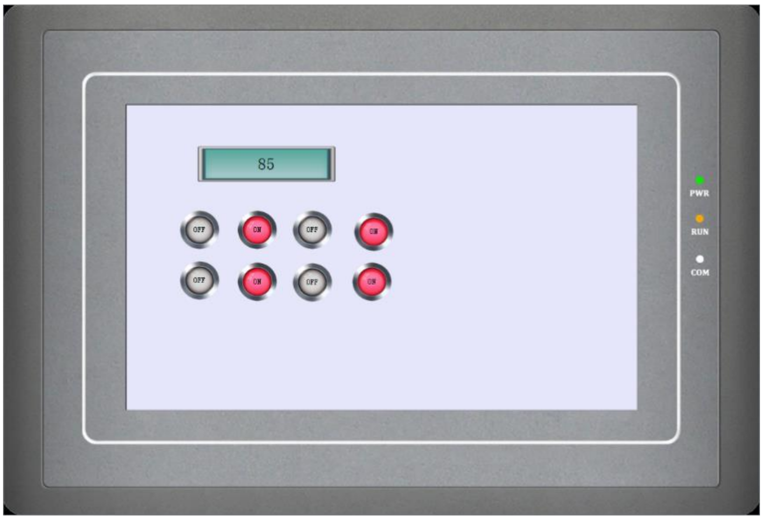
Create a touch screen project and set the corresponding mapping variables.


Communication Port Properties
General
Parameter
Link ID: 1
Link Name: Link1
Link Interface: Ethernet
HMI Site: Local
Setting
tion Services: invt
TS600 TCP Series
HMI IP
Ip: 192 168 1 100
Subnet mask: 255 255 255 0
Gateway: 192 168 1 1
ok
cancel


Communication Port Properties
General
Parameter
Connected equipment ip
Address: 192 168 1 10
t number: 502
Other
HMI Address: 0
Plc station: 1
Overtime time 1: 1000 (ms)
Overtime time 2: 5 (ms)
Retries: 3
Address mode: Standard M
LC address interval: 32
Spare set parameters
ok
cancel


88888
ON
Bit Lamp
Element type
Bit Lamp
ID: BL0000
View
Prompt
General
Appearance
Visibility
Shape
State: 1 0
FG Color:
BG Color:
Pattern: Transparent
Data Type: Bit
Monitor Address: M1
Offset


88888
ON
Numentric Display
Element type
Numentric Display
ID: ND0000
View
Prompt
Function: Monitor Register value while change
General
Picture
Advanced
Visibility
Shape
Border Color:
FG Color:
BG Color:
Text Color:
Pattern: Solid
Data Type: 16-Bit Unsigned Int
Unit
Display Type: 16-Bit Unsigned Decim
Monitor Address: D20
Offset
Font: 宋体
Font Size: 10
Total Digits: 5
Alignment
Left
Center
Right
Justification
Zero Header
Space Header
Ok
Cancel
Help


85
OFF
ON
PWR
RUN
COM

18.2.2 Accessing Custom Variables via Tag Communication

18.2.2.1 Version Restrictions

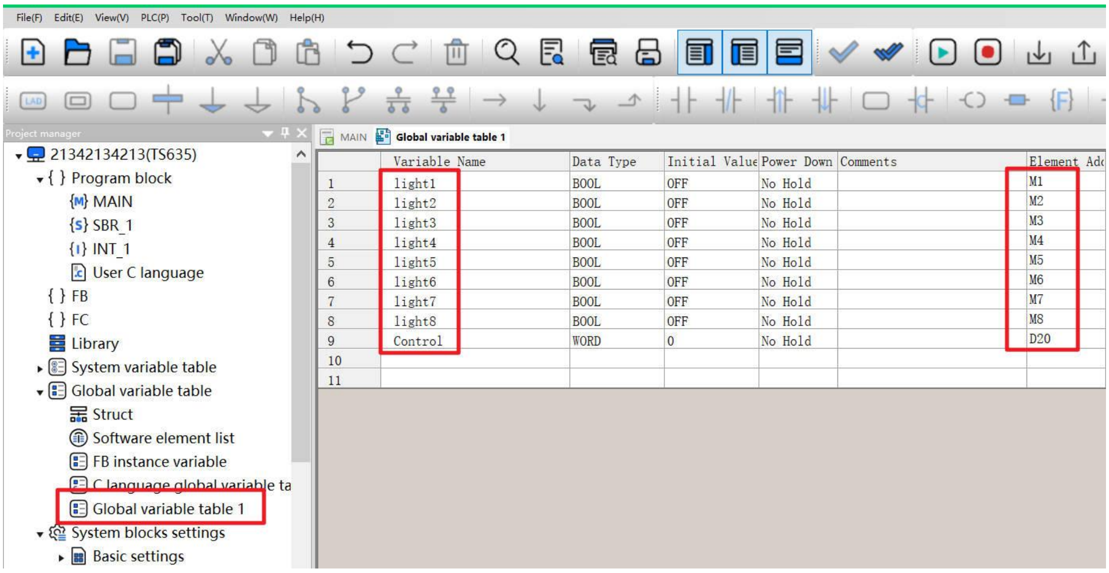
Note: The following software version requirements must be met for the tag communication function to operate properly.

Software	Minimum version number
Auto Station Pro	V1.3.2
HMITOOL	V7.1-010-build20240515
PLC firmware: TS600_MCU_XXX	V1.41.00
HMI firmware: GUI version number	7.1.5.42

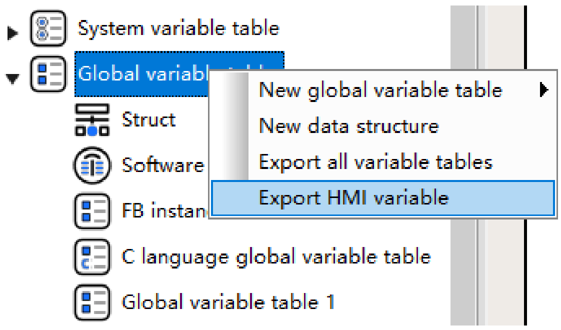
18.2.2.2 PLC Programming

Step 1 Create a new PLC project, see section 2.4 Programming and Debugging for details.

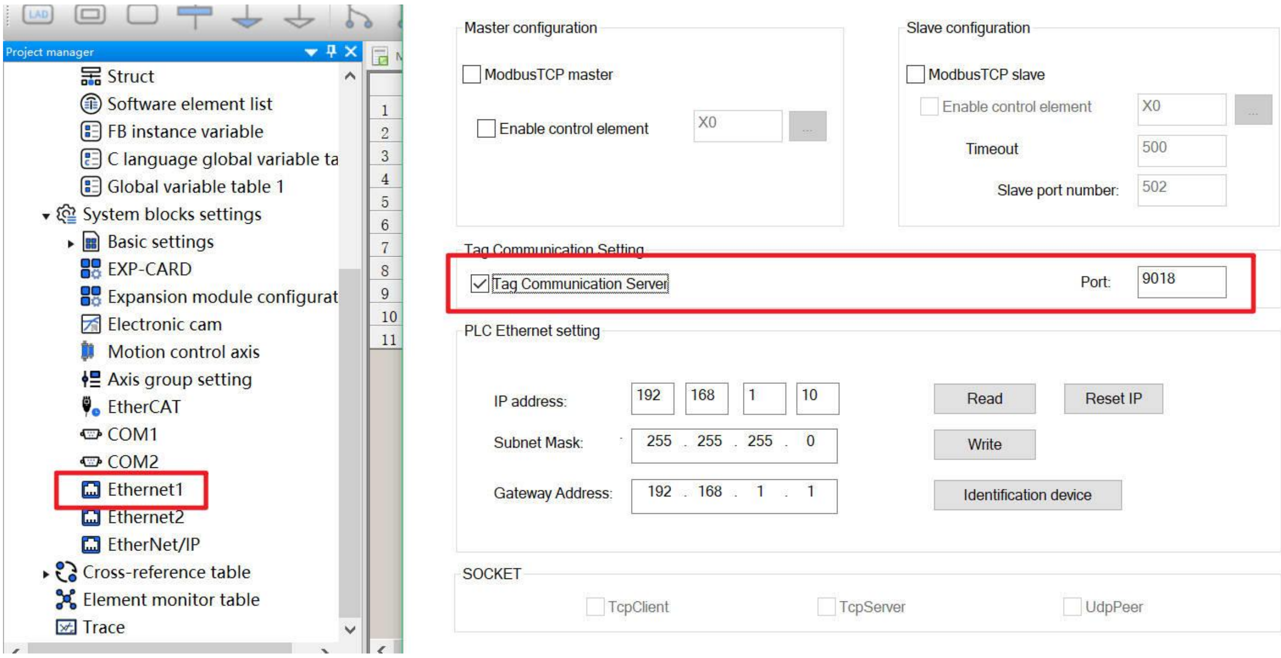
Step 2 Define custom variables.



Step 3 Compile the program and click "Export HMI variable" file, then save it in an appropriate location on your computer.



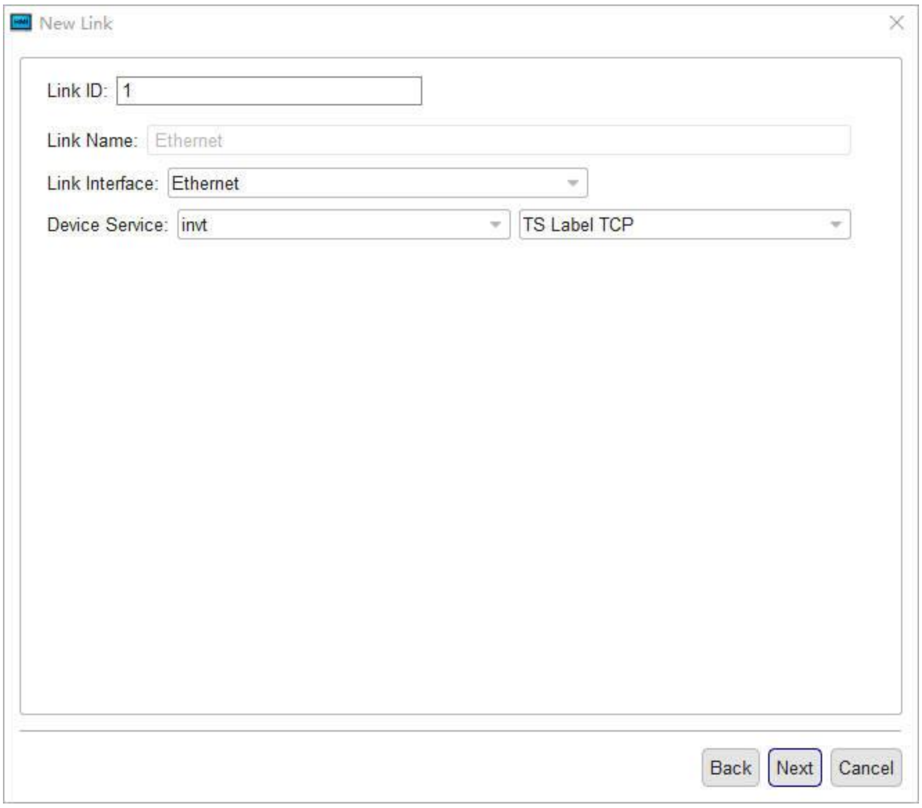
Step 4 Communication configuration; double-click "Ethernet1", select "Tag Communication Server" in the "Tag Communication Setting", and configure the port number as "9018."



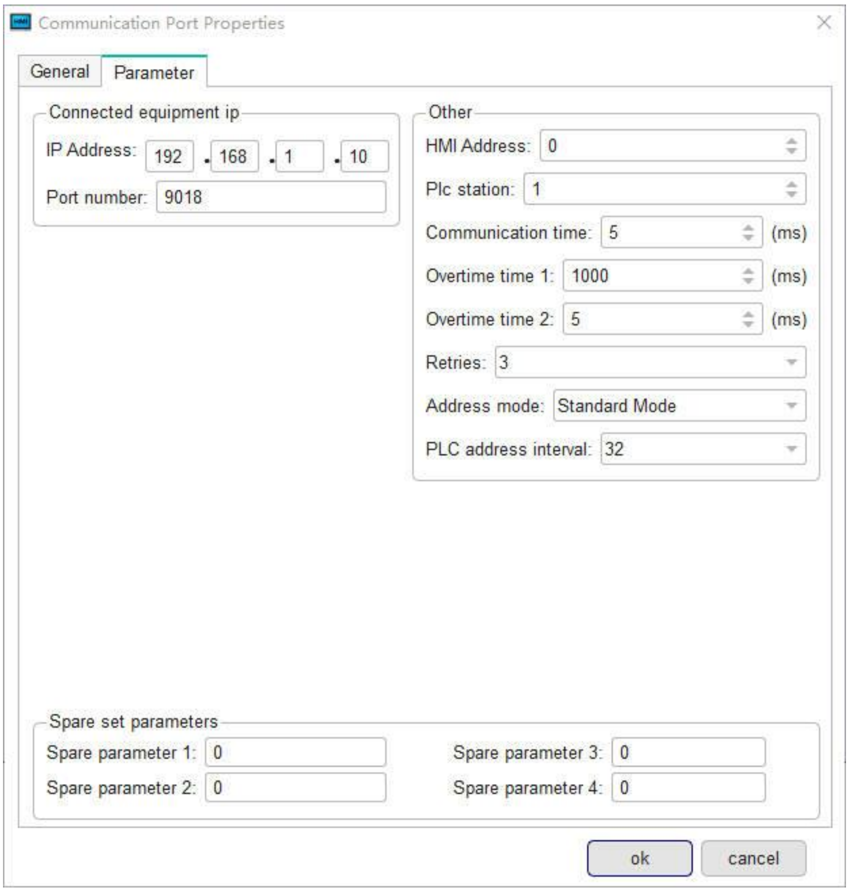
Step 5 Compile the program and download it to the controller, then run it.

18.2.2.3 Touch Screen Programming

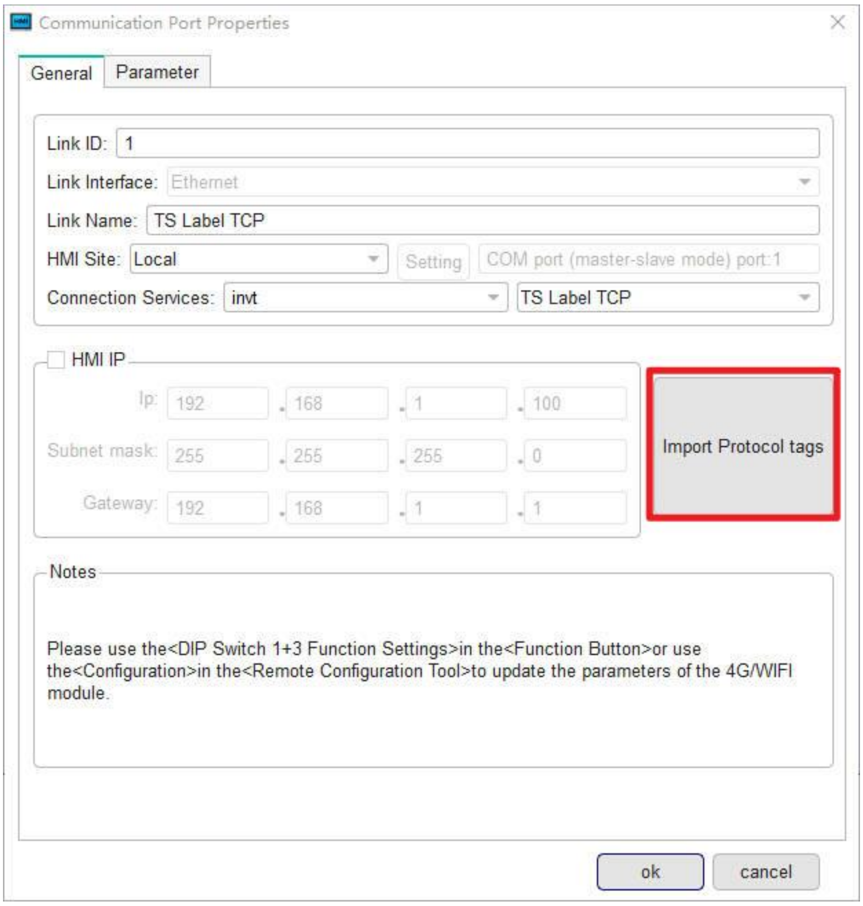
Step 1 Create a new HMI project in the HMITOOL host controller.



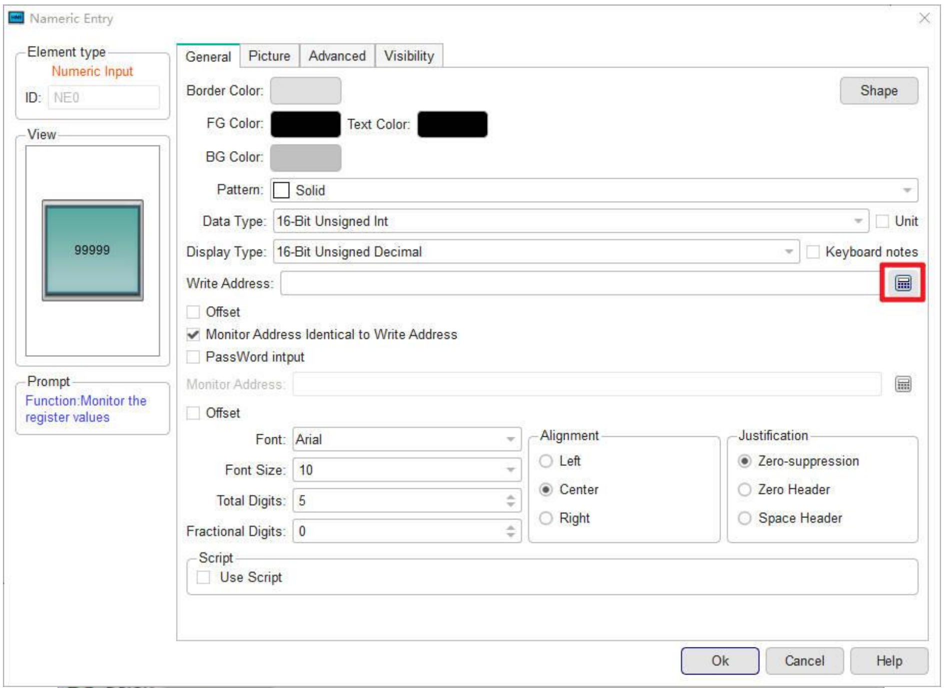
Step 2 Tag communication configuration; fill in the "IP Address" of the PLC and the "Port number" of the tag communication.

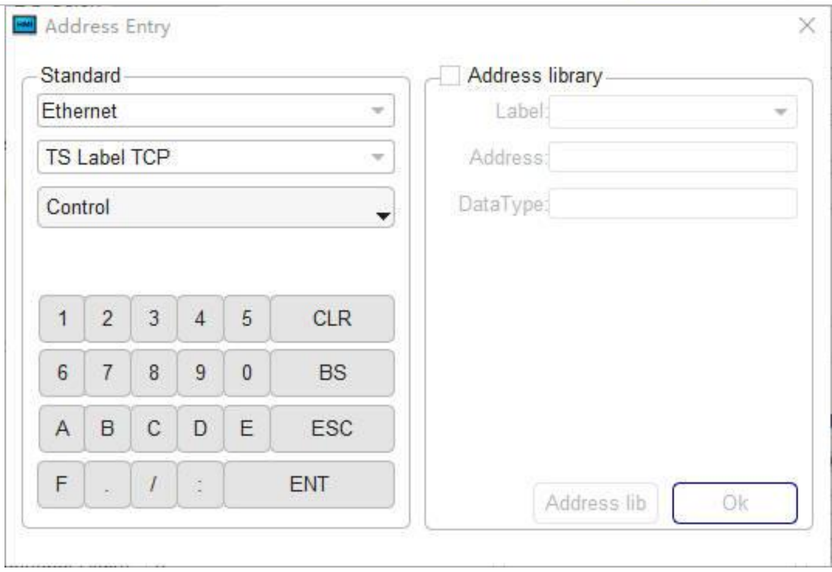


Step 3 Click "Import Protocol tags" and select the XML file exported from the Auto Station Pro host controller.

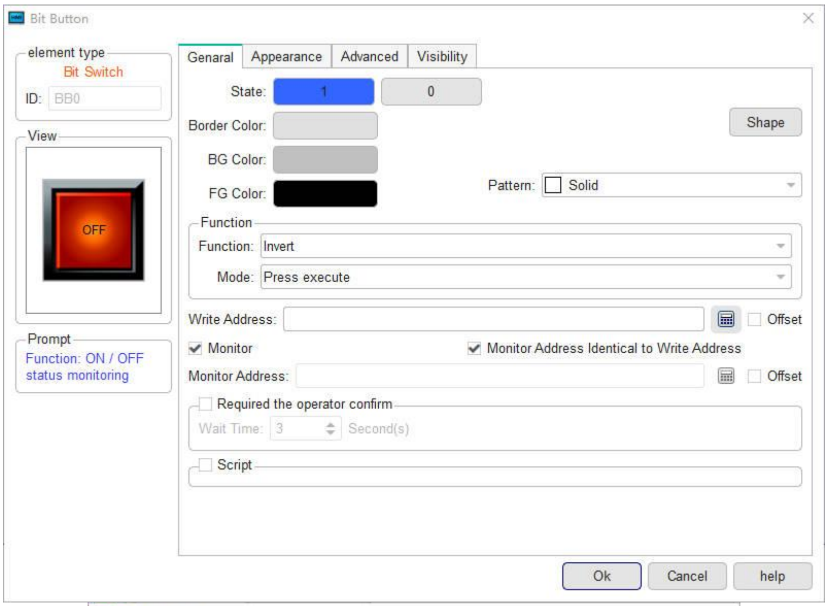


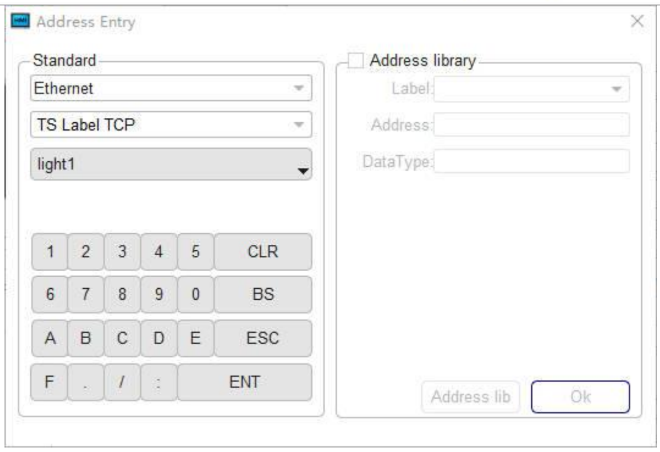
Step 4 Define the "Numeric Input" control and configure it as shown below to monitor the Control variable of the TS600 series.





Step 5 Define a "Bit Switch" control and configure it as shown below to monitor the light1 variable of the TS600 series.





Step 6 Compile and download the touch screen program.

Step 7 Properly connect the Ethernet cable between the touch screen and the PLC.

Step 8 Read and write custom variables on the PLC via the touch screen.

19 Fault Diagnosis

19.1 Panel Diagnosis

19.1.1 Indicator

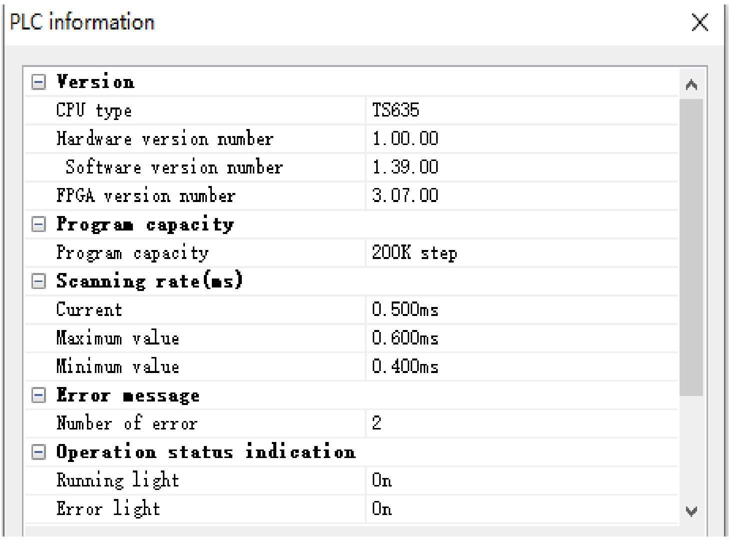
The status definition of TS600 series panel indicators is shown in the following table. When the error indicator is on, the PLC is in the STOP state, and the error level is "serious error". When the error indicator blinks, the PLC is in the RUN state, and the error level is "general error".

<table>
<tr><th>Port Type</th><th>Interface Sign</th><th>Definition</th><th>Indicator Color</th><th>Description</th></tr>
<tr><td rowspan="2">IO indicator</td><td>1600D</td><td>INPUT state display</td><td>Yellowish green</td><td>● Steady On: The input is valid or at high-speed
● Off: There is no input
● Flashing: A pulse is being input</td></tr>
<tr><td>0016DN</td><td>OUTPUT state display</td><td>Yellowish green</td><td>● Steady On: The output is valid or at high-speed
● Off: There is no output
● Flashing: A pulse is being output</td></tr>
<tr><td rowspan="6">Running state indicator</td><td>PWR</td><td>Power status indication</td><td>Yellowish green</td><td>● On: The power supply is normal.
● Off: Abnormal power supply</td></tr>
<tr><td>RUN</td><td>Running status indication</td><td>Yellowish green</td><td>● Steady On: The user program is running
● Off: The user program stops</td></tr>
<tr><td>ERR</td><td>Running error indication</td><td>Red</td><td>● Off: No error
● Steady On: Serious error
● Flash: Common error
● Slow flash: The device needs to be powered on again</td></tr>
<tr><td>EtherNET1</td><td>EtherNET1 connection</td><td>Yellow, yellowish green</td><td>● Yellowish green light is steady on: The link has been established successfully
● Yellowish green light and yellow light are flashing: Communication is in progress
● Off: The link is not established</td></tr>
<tr><td>EtherNET2</td><td>EtherNET2 connection</td><td>Yellow, yellowish green</td><td>● Yellowish green light is steady on: The link has been established successfully
● Yellowish green light and yellow light are flashing: Communication is in progress
● Off: The link is not established</td></tr>
<tr><td>EtherCAT</td><td>EtherCAT connection</td><td>Yellow, yellowish green</td><td>● Yellowish green light is steady on: The link has been established successfully
● Yellowish green light and yellow light are flashing: Communication is in progress
● Off: The link is not established</td></tr>
</table>

19.2 Software Diagnosis

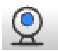
19.2.1 Get the Basic Information of PLC

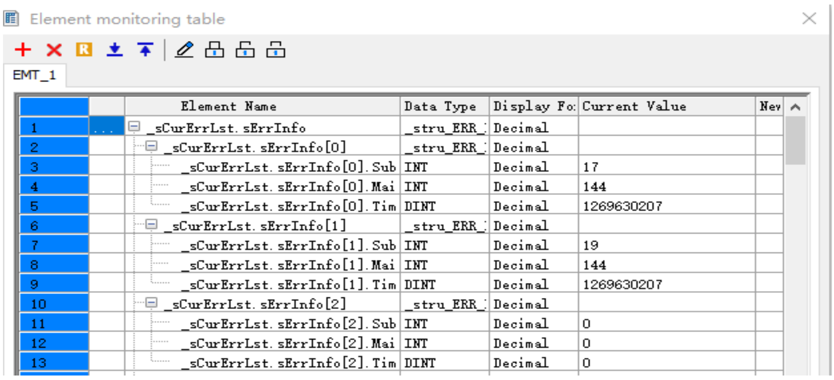
- You can click on the "PLC" on the toolbar to check PLC information, which displays the number of errors and the status of the indicators.



PLC information

Version	
CPU type	TS635
Hardware version number	1.00.00
Software version number	1.39.00
FPGA version number	3.07.00
Program capacity	
Program capacity	200K step
Scanning rate(ms)	
Current	0.500ms
Maximum value	0.600ms
Minimum value	0.400ms
Error message	
Number of error	2
Operation status indication	
Running light	On
Error light	On

- Users can click the toolbar  button to enter the monitoring mode. After entering the monitoring mode, check the error information, corresponding main error codes and subcodes through the element monitoring table.



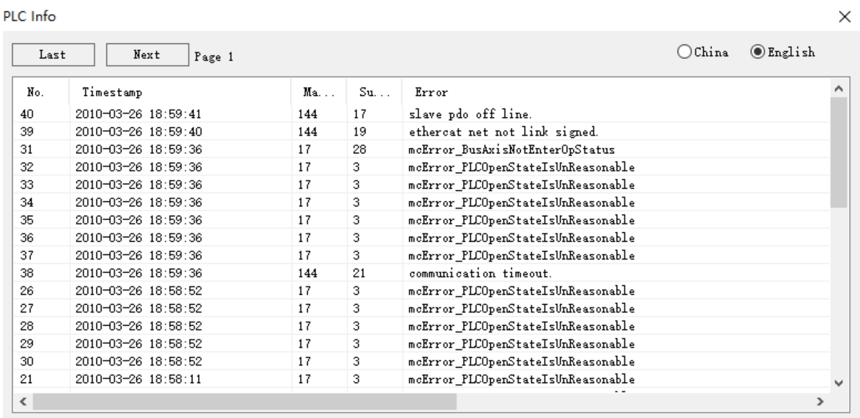
Element monitoring table

EMT_1

		Element Name	Data Type	Display Fo	Current Value	New
1	...	_sCurErrLst.sErrInfo	_stru_ERR_	Decimal		
2		_sCurErrLst.sErrInfo[0]	_stru_ERR_	Decimal		
3		_sCurErrLst.sErrInfo[0].Sub	INT	Decimal	17	
4		_sCurErrLst.sErrInfo[0].Mai	INT	Decimal	144	
5		_sCurErrLst.sErrInfo[0].Tim	DINT	Decimal	1269630207	
6		_sCurErrLst.sErrInfo[1]	_stru_ERR_	Decimal		
7		_sCurErrLst.sErrInfo[1].Sub	INT	Decimal	19	
8		_sCurErrLst.sErrInfo[1].Mai	INT	Decimal	144	
9		_sCurErrLst.sErrInfo[1].Tim	DINT	Decimal	1269630207	
10		_sCurErrLst.sErrInfo[2]	_stru_ERR_	Decimal		
11		_sCurErrLst.sErrInfo[2].Sub	INT	Decimal	0	
12		_sCurErrLst.sErrInfo[2].Mai	INT	Decimal	0	
13		_sCurErrLst.sErrInfo[2].Tim	DINT	Decimal	0	

19.2.2 View Error Log

The user can click the menu "PLC" and then select the PLC log file in the drop-down box to upload log information from PLC as follows:



PLC Info

Last Next Page 1 China English

No.	Timestamp	Ma...	Su...	Error
40	2010-03-26 18:59:41	144	17	slave pdo off line.
39	2010-03-26 18:59:40	144	19	ethercat net not link signed.
31	2010-03-26 18:59:36	17	28	mcError_BusAxisNotEnterOpStatus
32	2010-03-26 18:59:36	17	3	mcError_PLCOpenStateIsUnReasonable
33	2010-03-26 18:59:36	17	3	mcError_PLCOpenStateIsUnReasonable
34	2010-03-26 18:59:36	17	3	mcError_PLCOpenStateIsUnReasonable
35	2010-03-26 18:59:36	17	3	mcError_PLCOpenStateIsUnReasonable
36	2010-03-26 18:59:36	17	3	mcError_PLCOpenStateIsUnReasonable
37	2010-03-26 18:59:36	17	3	mcError_PLCOpenStateIsUnReasonable
38	2010-03-26 18:59:36	144	21	communication timeout.
26	2010-03-26 18:58:52	17	3	mcError_PLCOpenStateIsUnReasonable
27	2010-03-26 18:58:52	17	3	mcError_PLCOpenStateIsUnReasonable
28	2010-03-26 18:58:52	17	3	mcError_PLCOpenStateIsUnReasonable
29	2010-03-26 18:58:52	17	3	mcError_PLCOpenStateIsUnReasonable
30	2010-03-26 18:58:52	17	3	mcError_PLCOpenStateIsUnReasonable
21	2010-03-26 18:58:11	17	3	mcError_PLCOpenStateIsUnReasonable

19.3 Fault Code List

19.3.1 Fault Code Classification

Device Category	Device Type	Module Type	Main Fault Code (HEX)	Main Fault Code (DEC)
CPU	System-related	Hardware fault	0001	1
		System fault	0002	2
		Program fault	0003	3
		Reserved fault	0004–0007	4~7
	System component-related	Clock system component fault	0008	8
		IP system component fault	0009	9
		Reserved fault	000A–000F	10~15
	Functional component-related	Codesys motion control fault	0010	16
		Autonomous motion control fault	0011	17
		High speed input fault	0012	18
		CANopen axis control fault	0013	19
		Reserved fault	0014–0017	20~23
	Process library	Reserved fault	0018–002F	24~47
Backplane bus	Backplane bus-related	CPU IO failure	0030	48
		Digital quantity fault	0031	49
		Analog quantity fault	0032	50
		Fault of temperature measuring module	0033	51
		Encoder input fault	0034	52
		Reserved fault	0035–003F	53~63
Fieldbus	Serial port-related	Modbus RTU/ASCII Master 1	0040	64
		Modbus RTU/ASCII Master 2	0041	65
		Modbus RTU/ASCII Master 3	0042	66
		Modbus RTU/ASCII Slave 1	0043	67
		Modbus RTU/ASCII Master 2	0044	68
		Modbus RTU/ASCII Master 3	0045	69
		Serial freeport 1	0046	70
		Serial freeport 2	0047	71
		Serial freeport 3	0048	72
		Modbus RTU /ASCII master command 1	0049	73
		Modbus RTU /ASCII master command 2	004A	74
		Reserved fault	004B–004F	75~79
	CAN-related	CANopen	0050	80
		CANfree	0051	81
		CANnet	0052	82
		Reserved fault	0053–0057	83~87
	PROFIBUS	PROFIBUS DP	0058	88
		Reserved fault	0059–005F	89~95
	Reserved	Reserved fault	0060–006F	96~111

Device Category	Device Type	Module Type	Main Fault Code (HEX)	Main Fault Code (DEC)
Industrial Ethernet	Profinet-related	PROFINET	0070	112
		Reserved fault	0071–007F	113~127
	Ethernet/IP-related	Ethernet/IP	0080	128
		Reserved fault	0081–008F	129~143
	EtherCAT-related	EtherCAT	0090	144
		ET-Digital quantity	0091	145
		ET-Analog quantity	0092	146
		ET-Temperature measuring module	0093	147
		ET-Encoder input	0094	148
		Reserved fault	0095–009F	149~159
	Modbus TCP-related	Modbus TCP Master(Ethernet1)	00A0	160
		Modbus TCP Master(Ethernet2)	00A1	161
		Modbus TCP Slave(Ethernet1)	00A2	162
		Modbus TCP Slave(Ethernet2)	00A3	163
		Reserved fault	00A4–00AF	164~175
	TCP-related	TCP	00B0	176
		Reserved fault	00B1–00B7	177~183
	UDP-related	UDP	00B8	184
		Reserved fault	00B9–00BF	185~191
	OPC UA	OPC UA	00C0	192
	Reserved	Reserved fault	00C1–00EF	193~239
Expansion card	IoT card	4G expansion card	00F0	240
	Reserved	Reserved fault	00F1–00F3	241~243
Other	Other	Reserved fault	00F4–00FF	244~255

19.3.2 Fault Code List

Note: The main/sub fault codes are in hexadecimal format, with the corresponding decimal values shown in parentheses.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
0x0001(1)	0x0001(1)	General error	Button cell is not installed or battery voltage is too low	Check the battery.
	0x0002(2)	General error	Device supply voltage is too low (less than 19V)	Check the power supply.
0x0002(2)	0x0001(1)	Serious error	Hardware initialization error	Check whether the peripheral device works normally and whether the driver is loaded successfully
	0x0002(2)	Serious error	Failed to open GPIO	Check whether the driver and hardware work properly
	0x0003(3)	Serious error	Failed to write GPIO	Check whether the driver and hardware work properly
	0x0004(4)	Serious error	Failed to read GPIO	Check whether the driver and hardware work properly

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0005(5)	Serious error	Failed to open FPGA FMC	Check whether the driver and hardware work properly
	0x0006(6)	Serious error	SPI operation failed	Check whether the driver and hardware work properly
	0x0007(7)	Serious error	Failed to update FPGA firmware read signal	Check whether the driver and hardware work properly
	0x0008(8)	Serious error	Failed to read FPGA firmware file	Check whether the file exists or is corrupted
	0x0009(9)	Serious error	Failed to open I2C device	Check whether the driver and hardware work properly
	0x000A(10)	Serious error	Failed to write to I2C device	Check whether the driver and hardware work properly
	0x000B(11)	Serious error	Failed to read I2C device	Check whether the driver and hardware work properly
	0x000C (12)	Serious error	Failed to write FMC device	Check whether the driver or FPGA is working properly
	0x000D (13)	Serious error	Failed to read FMC device	Check whether the driver or FPGA is working properly
	0x000E(14)	Serious error	Failed to open USB device	Check whether the driver and hardware work properly
	0x000F(15)	Serious error	Failed to create USB epoll	Check whether the system is working properly
	0x0010(16)	Serious error	Programming port TCP initialization failed	Check whether the driver and hardware work properly
	0x0011(17)	Serious error	Failed to create programming port TCP epoll	Check whether the driver and hardware work properly
	0x0012(18)	Serious error	Element and variable forced setting failed	Check whether the element type and address are correct
	0x0013(19)	Serious error	Failed to open configuration file	Check whether the configuration file exists or is damaged
	0x0014(20)	General error	Power-down keeping configuration parsing failed	Check whether the profile data is correct
	0x0015(21)	General error	Failed to stop output configuration parsing	Check whether the profile data is correct
	0x0016(22)	General error	Watchdog configuration parsing failed	Check whether the profile data is correct
	0x0017(23)	General error	Constant scan time configuration parsing failed	Check whether the profile data is correct
	0x0018(24)	General error	Power-down wait time configuration parsing failed	Check whether the profile data is correct
	0x0019(25)	General error	Digital filter parameter configuration parsing failed	Check whether the profile data is correct
	0x001A(26)	General error	Advanced control configuration parsing failed	Check whether the profile data is correct
	0x001B(27)	General error	External input run configuration parsing failed	Check whether the profile data is correct
	0x001C	General	Serial port 1 configuration	Check whether the profile data is

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(28)	error	parsing failed	correct
	0x001D (29)	General error	Serial port 2 configuration parsing failed	Check whether the profile data is correct
	0x001E(30)	General error	Serial port 232 configuration parsing failed	Check whether the profile data is correct
	0x001F(31)	General error	Modbus RTU Master 1 configuration parsing failed	Check whether the profile data is correct
	0x0020(32)	General error	Modbus RTU Master 2 configuration parsing failed	Check whether the profile data is correct
	0x0021(33)	General error	Ebus serial port 1 configuration parsing failed	Check whether the profile data is correct
	0x0022(34)	General error	Ebus serial port 2 configuration parsing failed	Check whether the profile data is correct
	0x0023(35)	General error	Expansion module configuration parsing failed	Check whether the profile data is correct
	0x0024(36)	General error	Interrupt configuration code configuration parsing failed	Check whether the profile data is correct
	0x0025(37)	General error	Network port 1 configuration parsing failed	Check whether the profile data is correct
	0x0026(38)	General error	Network port 2 configuration parsing failed	Check whether the profile data is correct
	0x0027(39)	General error	Modbus TCP master 1 configuration parsing failed	Check whether the profile data is correct
	0x0028(40)	General error	Modbus TCP master 2 configuration parsing failed	Check whether the profile data is correct
	0x0029(41)	General error	CANOpen configuration parsing failed	Check whether the profile data is correct
	0x002A(42)	General error	Ethercat configuration parsing failed	Check whether the profile data is correct
	0x002B(43)	General error	Fieldbus pulse axis configuration parsing failed	Check whether the profile data is correct
	0x002C (44)	General error	Encoder axis configuration parsing failed	Check whether the profile data is correct
	0x002D (45)	General error	Fieldbus pulse axis configuration parsing failed	Check whether the profile data is correct
	0x002E(46)	General error	Encoder axis configuration parsing failed	Check whether the profile data is correct
	0x002F(47)	General error	Axis group configuration parsing failed	Check whether the profile data is correct
	0x0030(48)	General error	CAM table configuration parsing failed	Check whether the profile data is correct
	0x0031(49)	General error	Axis type configuration parsing failed	Check whether the profile data is correct
	0x0032(50)	General error	Fieldbus servo axis configuration parsing failed	Check whether the profile data is correct
	0x0033(51)	General error	Local pulse axis configuration parsing failed	Check whether the profile data is correct

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x003F(63)	Serious error	Failed to allocate profile memory	Check system free memory
	0x0040(64)	General error	Configuration parsing overrun failed	Check whether the profile data is correct
	0x0041(65)	Serious error	Failed to start application due to undervoltage	Check whether the power supply voltage is normal
	0x0042(66)	Serious error	Power failure detected	Check whether the power supply voltage is normal
	0x0043(67)	Serious error	Failed to open power-down keeping file	Check whether the power-down keeping file or file directory exists
	0x0044(68)	Serious error	Failed to get power-down keeping file size	Check whether the power-down keeping file is damaged
	0x0045(69)	Serious error	Failed to map the power-down keeping file	Check the file size and whether the system is normal
	0x0046(70)	Serious error	Failed to release the power-down keeping file mapping	Check the file size and whether the system is normal
	0x0047(71)	Serious error	Error in detecting power-down keeping file header	Check whether the power-down keeping file is damaged
	0x0048(72)	Serious error	Error in detecting power-down keeping file length	Check whether the power-down keeping file is damaged
	0x0049(73)	Serious error	Error in detecting power-down keeping file tail	Check whether the power-down keeping file is damaged
	0x004A(74)	Serious error	Error in detecting power-down keeping file CRC	Check whether the power-down keeping file is damaged
	0x0050(80)	Warning	Watchdog timeout	Check whether the user program is running correctly
	0x0051(81)	Warning	Error message mismatch	Check whether the error message is filled in correctly
	0x0052(82)	Serious error	Failed to open RND file	Check whether the system is working properly
	0x0053(83)	Serious error	Failed to create thread	Check whether the system is working properly
	0x0054(84)	Serious error	Failed to open instruction library	Check whether the system is working properly
	0x0055(85)	Serious error	Failed to open user program	Check whether the system is working properly
	0x0056(86)	Serious error	Device model does not match fieldbus axis number	Check device model and user program fieldbus number
	0x0057(87)	Serious error	PLC output type does not match the device.	Check whether the PLC output matches the device.
	0x0058(88)	General error	Configuration code 0x22 is empty.	Check whether the configuration code is set correctly.
	0x0059(89)	Serious error	The interrupt registration function is empty.	Check the interrupt registration function.
	0x005A(90)	Warning	The interrupt function is undefined.	Check the interrupt function.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x005B(91)	Warning	Exceeding interrupt source number.	Check the range of the interrupt source number.
	0x005C(92)	General error	The mapping address of the expansion module is empty.	Check the module mapping address.
0x0003(3)	0x0030(48)	General error	Illegal instruction system parameter	Please recompile and download the user program
	0x0031(49)	General error	Parameter is out of limit address range	Check whether the parameter variable data length of the instruction is out of range
	0x0032(50)	General error	Illegal instruction user parameters	Check whether the parameters of the instruction are in the wrong order or the size is set incorrectly
	0x0033(51)	General error	Wrong PID sampling time	1. Stop PID operation and check whether the parameters are set correctly 2. Check whether the setting mode is correct
	0x0034(52)	General error	Wrong PID filter constant	
	0x0035(53)	General error	Wrong PID proportional gain	
	0x0036(54)	General error	Wrong PID integration time	
	0x0037(55)	General error	Wrong PID differential gain	
	0x0038(56)	General error	Wrong PID differential time	
	0x0039(57)	General error	Wrong PID manual output PID value	
	0x003A(58)	General error	The PID setting target value exceeds the upper/lower limit of the setting value	
	0x003B(59)	General error	PID mode is not supported	
	0x003C (60)	General error	PID process value out of range error	
	0x003D (61)	General error	PID temperature sudden change error	
	0x003E(62)	General error	The setting of PID control period is unreasonable	Check whether the control period is larger than the PID sampling time
	0x003F(63)	General error	TPID mode auto-tuning failed	1. Limit cycle method: Try to lower the set temperature value and increase the sampling cycle, then rerun the program for autotuning. 2. Ascending curve method: Try to increase the sampling cycle and set the target temperature value at least 30°C higher than the process temperature value. If self-tuning is not possible for a long time, please confirm whether

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
				the control device and sensors are abnormal
	0x0050(80)	General error	Illegal ASCII code conversion value	Confirm whether the ASCII code to be converted conforms to the ASCII code specification
	0x0051(81)	General error	Clock chip read-write error	If the upper computer clock can be read normally, recompile and download the program. If the clock of the upper computer cannot be read, check whether the hardware is damaged or the battery is exhausted
	0x0052(82)	General error	Stack definition error	Check whether the stack data is normal
	0x0053(83)	General error	The divisor in the division operation is 0	Check whether the divisor in the division operation is 0
	0x0054(84)	General error	String instruction or data error	Check whether the string instruction or string data is illegal
	0x0055(85)	General error	Override between source and target operands	Check for overlap between source and target operands
	0x0080 (128)	Warning	Invalid upper and lower limit setting range	Check whether the lower limit is greater than the upper limit, and exchange the upper/lower limit operation in this case
	0x0081 (129)	Warning	-	-
	0x0082 (130)	Warning	PID deviation out of range	The calculated value of PID deviation exceeds-32768 – 32767
	0x0083 (131)	Warning	PID proportional term out of range	The calculated value of PID proportional term exceeds-32768 – 32767
	0x0084 (132)	Warning	PID integral term out of range	The calculated value of PID integral term exceeds-32768 – 32767
	0x0085 (133)	Warning	PID differential term out of range	The calculated value of PID differential term exceeds-32768 – 32767
	0x0086 (134)	Warning	PID operation result out of range	PID calculation result exceeds-32768 – 32767
	0x0087 (135)	Warning	Instruction fetches ID number incorrectly	Check the compilation ID of the upper computer
0x0008(8)	0x0001(1)	Serious error	Failed to open RTC device	Check whether the driver and hardware work properly
	0x0002(2)	Serious error	Failed to write RTC device	Check whether the driver and hardware work properly
	0x0003(3)	Serious error	Failed to read RTC device	Check whether the driver and hardware work properly
	0x0004(4)	General error	Failed to read the real time of the system	Check whether the system works normally in real time
	0x0005(5)	Warning	Failed to read RTC time of	Check whether FPGA and RTC work

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			FPGA	properly
	0x0006(6)	Warning	Failure to write RTC time of FPGA	Check whether FPGA and RTC work properly
0x0009(9)	0x0001(1)	General error	The IP segments of IP1 and IP2 repeat error	Misconfigure the network segments of IP1 and IP2.
	0x0011(17)	General error	Read: IP1 module--Open file to report error	Check whether network driver runs normally.
	0x0012(18)	General error	Read: IP1 module--Unable to get IP information	
	0x0013(19)	General error	Write: IP1 module--IP address configuration error	Check whether the data in the IP segment is valid (0–255).
	0x0014(20)	General error	Write: IP1 module--Mask configuration error	Check whether the mask data is valid (0–255).
	0x0015(21)	General error	Write: IP1 module--Gateway configuration error	Check whether the gateway data is valid (0–255).
	0x0016(22)	General error	Write: IP1 module--USB network segment repeat error	Ensure that the IP1 network segment is separated from the USB network segment (TM700--192.168.3.x).
	0x0017(23)	General error	Write: IP1 module--IP and gateway not in the same network segment error	The IP segment and gateway must be configured to the same network segment.
	0x0021(33)	General error	Read: IP2 module--Open file to report error	Check whether network driver runs normally.
	0x0022(34)	General error	Read: IP2 module--Unable to get IP information	
	0x0023(35)	General error	Write: IP2 module--IP address error	Check whether the data in the IP segment is valid (0–255).
	0x0024(36)	General error	Write: IP2 module--Mask error	Check whether the mask data is valid (0–255).
	0x0025(37)	General error	Write: IP2 module--Gateway error	Check whether the gateway data is valid (0–255).
	0x0026(38)	General error	Write: IP2 module--USB network segment repeat error	Ensure that the IP2 network segment is separated from the USB network segment (TM700--192.168.3.x).
	0x0027(39)	General error	Write: IP2 module--IP and gateway not in the same network segment error	The IP segment and gateway must be configured to the same network segment.
0x0011(17)	0x0001(1)	General error	The current axis ID is not in the valid range	Check whether the axis ID parameter setting is reasonable.
	0x0002(2)	General error	The current function block ID is not in the valid range	Check whether the ID parameter setting of the upper computer function block is reasonable.
	0x0003(3)	Warning	The PLCopen state is unreasonable and the current function block cannot be started.	Check whether the current axis state satisfies the PLCopen state machine switching process when the current instruction is triggered.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0004(4)	Warning	Axis configuration failed	Check whether the axis configuration is successful
	0x0005(5)	Warning	PDO parameter DigitalIput address is NULL	1. Check whether the parameter is mapped in the slave station IO mapping. 2. Check whether the parameter exist in the XML version of the servo slave.
	0x0006(6)	General error	Current axis error/servo error	If the axis/servo fails, you can call the MC_Reset instruction or restart the MC_Power instruction to clear the error.
	0x0007(7)	Warning	The current axis is not enabled, and the axis is in Disabled state.	Switch to the Stanstill state by calling the MC_Power instruction axis.
	0x0008(8)	General error	Axis positive hard limit triggered	Call the reset instruction to switch the axis state from ErrorStop state to Standstill state
	0x0009(9)	General error	Axis negative hard limit triggered	Call the reset instruction to switch the axis state from ErrorStop state to Standstill state
	0x000A(10)	General error	Axis positive soft limit triggered	Call the reset instruction to switch the axis state from ErrorStop state to Standstill state
	0x000B(11)	General error	Axis negative soft limit triggered	Call the reset instruction to switch the axis state from ErrorStop state to Standstill state
	0x000C (12)	Warning	Pulse axis has no output device selected	Check whether the output device is selected for the pulse axis.
	0x000D (13)	Warning	Fieldbus has no output device selected.	Check whether the output device has been selected for fieldbus.
	0x000E(14)	Warning	The current instruction does not support repeated call technology.	The current instruction does not support repeated calls to function blocks, please avoid this situation intentionally.
	0x000F(15)	Warning	Axis type setting error	Check whether the axis type matches the instruction type.
	0x0010(16)	Warning	The fieldbus base PDO address is NULL	It is not recommended to map PDO parameters in slave device description file I/O mapping
	0x0011(17)	Warning	Positive hard limit ID configuration failed	Check whether the current pulse axis input and output points are reused
	0x0012(18)	Warning	Negative hard limit ID configuration failed	Check whether the current pulse axis input and output points are reused
	0x0013(19)	Warning	Probe ID1 configuration failed	Check whether the current pulse axis input and output points are

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
				reused
	0x0014(20)	Warning	Probe ID2 configuration failed	Check whether the current pulse axis input and output points are reused
	0x0015(21)	Warning	Servo error ID configuration failed	Check whether the current pulse axis input and output points are reused
	0x0016(22)	Warning	Homing signal ID configuration failed	Check whether the current pulse axis input and output points are reused
	0x0017(23)	Warning	Z signal ID configuration failed	Check whether the current pulse axis input and output points are reused
	0x0018(24)	Warning	Axis enable ID configuration failed	Check whether the current pulse axis input and output points are reused
	0x0019(25)	Warning	Clear servo error ID configuration failed	Check whether the current pulse axis input and output points are reused
	0x001A(26)	Warning	The axis address is NULL	Check whether the axis is configured successfully
	0x001B(27)	Warning	Fieldbus axis enable failed	Fieldbus axis enable timeout, check whether EtherCAT communication and feedback status word are normal
	0x001C (28)	General error	Fieldbus does not enter Op state	Check whether EtherCAT communication is in OP state
	0x001D (29)	Warning	The current function block execution is invalid.	The current instruction function is not available yet, and it is invalid to use.
	0x001E(30)	Warning	Current axis communication timeout	Check whether EtherCAT communication is in OP state. Check whether the EtherCAT communication return value is normal.
	0x001F(31)	Warning	Under the current axis configuration, the EtherCAT synchronization period cannot be less than 1ms.	Check whether the setting of the synchronization cycle of the EtherCAT master station is less than 1ms (in case of mixed use of bus axis and pulse axis, the EtherCAT synchronization cycle cannot be less than 1 ms).
	0x0020(32)	Warning	PLC is not running.	Check whether the PLC toggle switch is set to Stop.
	0x0021(33)	Warning	Axis trigger software does not decelerate to stop.	The current axis is in the process of soft limit deceleration and stop, and the execution of the current trigger instruction is invalid.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0022(34)	Warning	The current instruction parameter address is NULL.	The current instruction parameter address is NULL, please give an input variable or contact INVT technical service.
	0x0023(35)	General error	During the pulse axis movement, the pulse frequency of the current interpolation period is greater than or equal to 200K.	The maximum operating frequency of pulse axis is not allowed to exceed 200K, so it is recommended to reduce the speed of operation.
	0x0024(36)	Warning	Pulse axis FPGA cache reaches limit value.	Prompt only
	0x0025(37)	General error	The PDO data address in EtherCAT is NULL	Check whether the EtherCAT communication is normal.
	0x0026(38)	General error	The current servo axis is not online.	Check whether EtherCAT communication is normal; Check whether the current servo axis is plugged into the network cable connection
	0x0027(39)	Warning	Current axis communication failed	During running, EtherCAT communication failed, check EtherCAT communication status
	0x0028(40)	Warning	The PDO parameter StatusWord value is 0	Check whether the EtherCAT communication is normal.
	0x0029(41)	Warning	PDO parameter ErrorCode address is NULL	1. Check whether the EtherCAT communication is normal. 2. Check whether the PDO parameter is configured.
	0x002A(42)	Warning	Torque control is not supported on the current axis.	Check axis type configuration, and torque control only supports fieldbus axis.
	0x0065 (101)	Warning	Enable instruction status exception	Enable instruction status is abnormal, please contact INVT technical service.
	0x0066 (102)	Warning	Reset instruction status exception	Reset instruction status is abnormal, please contact INVT technical service.
	0x0067 (103)	Warning	Reset timeout	Axis reset timed out, please check whether EtherCAT communication is normal.
	0x0068 (104)	Warning	Motion superposition is not supported in the current axis state.	If the current axis state does not support the superimposed motion command, refer to the specific commands for using the command.
	0x0069 (105)	Warning	Input parameter error	Instruction input parameter is not in the valid range
	0x006A (106)	Warning	MC_Stop instruction repeated call error	Please check whether the same axis is called more than once
	0x006B	Warning	MC_ImmediateStop instruction	Please check whether the same axis

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(107)		repeated call error	is called more than once
	0x006C (108)	General error	MC_Stop instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x006D (109)	General error	MC_Halt instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x006E (110)	Warning	MC_SetOverride instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x006F (111)	General error	MC_MoveVelocity instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0070 (112)	General error	MC_MoveRelative instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0071 (113)	General error	MC_MC_MoveAbsoulte instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0072 (114)	General error	MC_Jog instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0073 (115)	General error	MC_Inch instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0074 (116)	General error	MC_Home instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0075 (117)	Warning	MC_SetPosition instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0076 (118)	Warning	Invalid trigger of MC_SetOverride instruction in current axis state	The current axis is in the process of commutation, and the speed regulation is not effective.
	0x0077 (119)	Warning	The current axis is in the operation of the axis group.	Run the axis after the axis group completes running.
	0x0078 (120)	Warning	Axis state is not in StandStill state.	Please switch the axis state to StandStill state before triggering the

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
				current instruction.
	0x0079 (121)	Warning	Invalid MC_Reset instruction reset	The current axis state is not in ErrorStop state, and the reset is invalid.
	0x007A (122)	Warning	Invalid interpolation period value setting	Check EtherCAT synchronization cycle settings.
	0x007B (123)	Warning	Invalid trigger of MC_Stop instruction	Check whether the current axis state can trigger the instruction
	0x007C (124)	Warning	Invalid trigger of MC_Halt instruction	Check whether the current axis state can trigger the instruction
	0x007D (125)	Warning	Invalid trigger of MC_ImmediateStop instruction	Check whether the current axis state can trigger the instruction
	0x007E (126)	Warning	MC_TouchProbe instruction input parameter is not in the valid range	Please check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state.
	0x007F (127)	Warning	MC_MoveSuperImosed instruction input parameter is not in the valid range	Please check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state.
	0x0080 (128)	Warning	MC_Home instruction is called repeatedly	Please check whether the homing function block is called more than once on the same axis.
	0x0081 (129)	General error	MC_MoveFeed instruction input parameter is not in the valid range.	Please check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state.
	0x0082 (130)	Warning	The probe channel used is not configured	Check whether PDO data in "Process data" of the configuration interface of servo axis of upper computer has been added. (Possible mappings: 0x60B8, 0x60B9, 0x60BA, 0x60BB, 0x60BC, and 0x60BD)
	0x0083 (131)	Warning	Interrupt fixed-length function, when Mode=0 or 1, the probe signal has not arrived when the first segment is finished.	Check whether the probe signal is triggered correctly.
	0x0084 (132)	Warning	When the probe function is triggered, the probe channel used has been occupied by the interrupt fixed-length function.	Check the probe channels for conflicting.
	0x0085 (133)	Warning	Axis configuration index parameter is not in the valid range	Check whether the axis configuration index parameter is in the valid range
	0x0086 (134)	Warning	The axis parameter entered by the MC_SetAxisConfigPara instruction is not in the valid	Please check whether the axis setting parameters are within the valid range

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			range	
	0x0087 (135)	General error	MC_MoveBuffer instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0088 (136)	General error	MC_SyncMoveVelocity instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x0089 (137)	General error	MC_MoveVelocityCSV instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x008A (138)	General error	MC_SyncTorqueControl instruction input parameter is not in the valid range	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x008B (139)	Warning	PDO data used is not configured	The process data 0x6060 and 0x6061 in the servo configuration of the upper computer are not configured
	0x008C (140)	Warning	PDO data used is not configured	The process data 0x606C is not configured in the servo configuration of the upper computer
	0x008D (141)	Warning	PDO data used is not configured	The process data 0x60FF is not configured in the servo configuration of the upper computer.
	0x008E (142)	Warning	PDO data used is not configured	The process data 0x6071 and 0x607F are not configured in the servo configuration of the upper computer.
	0x008F (143)	Warning	PDO data used is not configured	The process data 0x6083 and 0x6084 are not configured in the servo configuration of the upper computer.
	0x0090 (144)	Warning	Single axis speed regulation function is not supported in the current axis state.	Check the current axis status and whether it meets the requirements of speed regulation function.
	0x0091 (145)	Warning	The probe does not support virtual axis or pulse axis.	Check whether the current axis type is configured as bus axis.
	0x0092 (146)	General error	MC_TorqueControl instruction input parameter is not in the valid range	Check whether the instruction parameters are within the valid range, and call the MC_Reset instruction to reset the axis state.
	0x0093 (147)	Warning	Address 0x6077 or 0x6087 is empty	Check whether 0x6077 or 0x6087 is mapped in the servo "process data".
	0x0094 (148)	Warning	Failed to switch to the target control mode	Check whether the bus communication is normal.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0095 (149)	Warning	Failed to write SDO parameters	Check whether the bus communication is normal
	0x0096 (150)	General error	MC_Homing instruction input parameter is not in the valid range	Check whether the instruction parameters are within the valid range, and call the MC_Reset instruction to reset the axis state.
	0x0097 (151)	Warning	MC_Homing instruction triggered repeatedly with error	Check whether the same axis has been called more than once
	0x0098 (152)	Warning	MC_Homing/MC_Home instruction timeout error	Check whether the Max. homing time setting is correct.
	0x0099 (153)	Warning	Control mode is switching	Check whether the bus communication is normal.
	0x00C9 (201)	Warning	The master and slave axes use the same axis ID	Check whether the master and slave axes are the same.
	0x00CA (202)	General error	Input parameter error of MC_GearOut function block	Check whether the input parameters of GearOut are within the constraint range of the command parameter list.
	0x00CB (203)	Warning	Invalid trigger of MC_GearOut function block	Check whether the slave axis is in gear action, and check whether the slave axis is in gear engagement action
	0x00CC (204)	General error	Input parameter error of MC_GearIn function block	Check whether the input parameters of MC_GearIn are within the constraint range of the command parameter list.
	0x00CD (205)	Warning	The current command to run the master axis does not meet the requirements.	Check whether the master axis state meets the requirements; Run the MC_Phasing instruction to check whether the current axis is in cam or gear action
	0x00CE (206)	Warning	The master axis has not reached the target velocity.	Check whether the current master axis has reached the target velocity.
	0x00CF (207)	General error	Input parameter error of MC_CamOut function block	Check whether the input parameters of MC_CamOut are within the constraint range of the command parameter list.
	0x00D0 (208)	Warning	Invalid trigger of MC_CamOut function	Check whether the slave axis is in cam action, and check whether the slave axis is in cam engagement action
	0x00D1 (209)	General error	Input parameter error of MC_CamIn function block	Check whether the input parameters of MC_CamIn are within the constraint range of the command parameter list.
	0x00D2 (210)	Warning	The current CamTable ID is not within the valid range.	Check whether the CamTable ID is within the constraint range of the command parameter list.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x00D3 (211)	Warning	Setting error of StartPosition or MasterStartDistance in MC_CamIn command	Check whether MasterStartDistance and StartPosition are in the current master axis running direction in absolute position mode
	0x00D4 (212)	Warning	The MC_CamIn instruction is in absolute position mode, with StartPosition ahead of MasterStartDistance	Check whether StartPosition is ahead of MasterStartDistance in the absolute position mode.
	0x00D5 (213)	General error	The input parameters of the MC_Phasing command are not within the valid range.	Check whether the input parameters of MC_Phasing are within the constraint range of the command parameter list.
	0x00E1 (225)	Warning	Master axis phase setting error	Check whether the master axis phases of two adjacent keypoints are less than or equal to 0.001 in the user-defined cam table of the MC_GenerateCamTable command.
	0x00E2 (226)	Warning	The start point of the cam table cannot be set as a non-zero parameter.	Check whether the positions of the master and slave axes at the start point of the cam are set to non-zero in the user-defined cam table of the MC_GenerateCamTable command.
	0x00E3 (227)	Warning	The current NodeNum parameter cannot be set to 0	Check whether the MC_NodeNum parameter is set to 0 in the current mode in the GenerateCamTable command.
	0x00E4 (228)	Warning	The current NodeNum parameter is not within the valid range.	Check whether the MC_NodeNum parameter is set within the constraint range of the command parameter list in the current mode in the GenerateCamTable command.
	0x00E5 (229)	Warning	Curve type setting error in cam table	Check whether the cam curve type settings are within the constraint range of the command parameters list. They only support 0 (which represents straight lines) and 1 (which represents quintic curves).
	0x00E6 (230)	Warning	The cam table is empty	Check whether the cam table is configured
	0x00E7 (231)	Warning	Encoder master axis enable failed	Check whether the count instruction ENC_Counter is enabled when the encoder master axis is being used.
	0x00E8 (232)	Warning	The length of the user-defined cam table is not within the valid range	Check that the length of the user-defined cam table array must be 32 in the MC_GenerateCamTable command
	0x00E9 (233)	Warning	Custom tappet switch is not in the valid range	Check that the length of the user-defined switch array must be

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
				32 in the MC_DigitalCamSwitch command.
	0x00EA (234)	Warning	The ReferenceType parameter settings are not within the valid range.	Check whether the ReferenceType parameter setting is in the valid range for the current instruction.
	0x00EB (235)	Warning	Channel parameter setting is not in the valid range	Check whether the current instruction and Channel parameter setting are within the valid range.
	0x00EC (236)	Warning	The Number parameter setting is not in the valid range.	Check whether the Number parameter setting of the current instruction is within the valid range.
	0x00ED (237)	Warning	The Switches parameter address is NULL.	Check whether the Switches parameter of the current instruction has a given variable.
	0x00EE (238)	Warning	In the tappet switch, the position is not ascending.	Check whether the Position in the Switches parameter is ascending in the current instruction, if not, please modify it
	0x00EF (239)	Warning	The current axis state does not support the use of tappet instruction.	Check whether the axis state is in the homing state.
	0x00F0 (240)	Warning	Action setting is not in the valid range in tappet switch.	Check whether the Action in the Switches parameter of the current instruction is within the specified valid range.
	0x00F1 (241)	Warning	The Channel is currently in use.	Check whether the Channel is reused.
	0x00F2 (242)	Warning	In the tappet switch, Position is set beyond the rotating axis modulus period.	Check whether the Position in the Switches parameter of the current instruction exceeds the rotation period value in the rotating axis mode.
	0x00F3 (243)	General error	The input parameters of the MC_CombineAxes instruction are not within the valid range.	Check whether the command parameters are within the valid range, and call the MC_Reset command to reset the axis state
	0x00F4 (244)	Warning	Phase of the MC_GetCamTableDistance command is not within the valid range between the start and end points.	Check whether the input parameter Phase of this command is within the valid range between the start and end points.
	0x00F5 (245)	Warning	CurveType parameter setting is not in a valid range	Check whether the CurveType parameter setting is in the valid range for the current instruction.
	0x00F6 (246)	Warning	The phases of the start and end points for the MC_GetCamTableDistance command is not arranged in	Check if the phase difference between the start and end points for this command is less than 0.001 Check whether Phase in CamTable

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			ascending order.	is in ascending order
	0x00F7 (247)	Warning	The current master axis has entered the ErrorStop state, and the function block has stopped running.	Check the reason why the master axis has entered the ErrorStop state.
	0x00F8 (248)	Warning	Multiple cam tables are used to save instructions on the same axis.	Check whether the user program uses multiple cam tables to save instructions on the same axis.
	0x00F9 (249)	Warning	The cam table instruction update is not completed and the save cam table instruction is called.	Check whether the user program has not completed the cam table update command and has called the cam table save command instead.
	0x012D (301)	General error	Function block input parameter error	In plane arc interpolation mode 2, if the distance between the start and end points is greater than twice the radius, check and correct the parameters.
	0x012E (302)	General error	Axis group ID settings exceeds the range	Check and correct the axis group ID
	0x012F (303)	General error	Two or more identical axis IDs are configured in the axis group	Check and correct the duplicated axis IDs in the axis group configuration interface
	0x0130 (304)	General error	The distance from the start end to the circle center is not equal to that from the end point to the circle center in the plane arc function block.	In plane arc interpolation mode 1, check and modify the distance from the start point to the circle center and that from the and end point to the circle center.
	0x0131 (305)	General error	The start point, circle center, and end point are on the same straight line in the plane arc function block.	In planar arc interpolation mode 0, the starting point, the auxiliary point and the ending point are in a straight line.
	0x0132 (306)	General error	The calculated circle center position is not unique in the plane arc function block.	In plane arc interpolation mode 2, ensure that the start point is equal to the end point.
	0x0133 (307)	General error	In the GroupImmediateStop module, the same axis group can only call this function block once, and the second function block starts reporting an error.	For the same axis group, the second immediate axis group stop module reports error.
	0x0134 (308)	General error	Axis group is in GroupImmediateStopping state	Pull down the MC_GroupImmediateStop module first, and then pull up the MC_GourpStop module.
	0x0135 (309)	General error	In the GroupStop module, the same axis group can only call this function block once, and the second function block starts reporting an error.	For the same axis group, the second MC_GroupStop module reports error when pulled up.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0136 (310)	General error	The configured velocity parameters are not within a reasonable range	Check the corresponding parameters
	0x0137 (311)	General error	The configured acceleration parameters are not within a reasonable range	Check the corresponding parameters
	0x0138 (312)	General error	The configured deceleration parameters are not within a reasonable range	Check the corresponding parameters
	0x0139 (313)	General error	The configured Jerk parameters are not within a reasonable range	Check the corresponding parameters
	0x013A (314)	General error	The configured AbsRelMode parameters are not within a reasonable range	Check the corresponding parameters
	0x013B (315)	General error	Interpolation is not allowed as there a single axis is in the rotation mode in the axis group	De-select the rotation mode option in the single axis configuration interface
	0x013C (316)	General error	Interpolation is not allowed as there a single axis is in the debugging mode in the axis group	De-select the debugging mode option in the single axis configuration interface
	0x013D (317)	General error	The radius parameter is not allowed to be zero	Check the corresponding parameters
	0x013E (318)	General error	The parameter CircAxes is not within the allowed range	Check the corresponding parameters
	0x013F (319)	General error	The parameter CircMode is not within the allowed range	Check the corresponding parameters
	0x0140 (320)	General error	The parameter PathChoice is not within the allowed range	Check the corresponding parameters
	0x0141 (321)	General error	The array parameters passed in by the upper computer are incorrect	Enable upper computer error protection
	0x0142 (322)	General error	It is not allowed to modify the parameter CircAxes during the operation of arc interpolation	Interrupt the arc interpolation first, and then modify the parameter CircAxes
	0x0143 (323)	General error	The current state does not allow axis group velocity regulation	The current state does not allow axis group velocity regulation, including moderate axis group deceleration
	0x0144 (324)	General error	An unconfigured axis group number has been used	Configure an axis group number for the used axis group in the "Axis Group Settings" list on the upper computer
	0x0145 (325)	General error	There is a pulse axis velocity exceeding 200kHz	There is a pulse axis velocity exceeding 200kHz
	0x0146 (326)	General error	Two axis groups use the same axis, so that when one axis group is in motion state, the	Modify the reused axis, or run two axis groups at different times.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			other axis group cannot enter the motion state.	
	0x0147 (327)	Warning	Multiple cam table save instructions are used in the same cam table.	Check the cam table save instructions in the ladder diagram.
	0x0148 (328)	Warning	The instruction update is not completed and the save instruction is called.	Check the order of calling the update instruction and save instruction.
0x0012(18)	0x0101 (257)	General error	System configuration-Code header frame configuration error	Update the corresponding version of the host controller or contact technical support.
	0x0102 (258)	General error	System configuration-Code length configuration error	
	0x0103 (259)	General error	System configuration-Code module type configuration error	
	0x0104 (260)	General error	System configuration-Code module length configuration error	
	0x0105 (261)	General error	System Configuration-Parameter module type configuration error	
	0x0106 (262)	General error	System Configuration-Parameter module length configuration error	
	0x0201 (513)	Warning	Counter reset module-Axis ID exceeds maximum value	Configure the correct axis ID.
	0x0202 (514)	Warning	Counter reset module-This axis does not belong to encoder axis	
	0x0203 (515)	Warning	Counter reset module-Axis number not configured	
	0x0301 (769)	Warning	Comparator reset module-Axis ID exceeds maximum value	
	0x0302 (770)	Warning	Counter reset module-This axis does not belong to encoder axis	
	0x0303 (771)	Warning	Counter reset module-Axis number not configured	
	0x0401 (1025)	Warning	Preset module-Axis ID exceeds maximum value	
	0x0402 (1026)	Warning	Preset module-This axis does not belong to encoder axis	
	0x0403 (1027)	Warning	Preset module-No axis number configured	
	0x0404 (1028)	Warning	Preset module-TrigerMode trigger mode parameter exception	Configure the correct TrigerMode range parameter (0–3).

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0405 (1029)	Warning	Preset module-Preset position out of range	Configure preset positions within the specified range.
	0x0406 (1030)	Warning	Preset module-No preset position configured	Configure the preset position.
	0x0501 (1281)	Warning	Counter module-Axis ID exceeds maximum value	Configure the correct axis ID.
	0x0502 (1282)	Warning	Counter module-This axis does not belong to encoder axis	
	0x0503 (1283)	Warning	Counter module-Axis number not configured	
	0x0601 (1537)	Warning	Comparator module-Axis ID exceeds maximum value	
	0x0602 (1538)	Warning	Comparator module-This axis does not belong to encoder axis	
	0x0603 (1539)	Warning	Comparator module-Axis number not configured	
	0x0604 (1540)	Warning	Comparator module-Comparison position out of limit	Configure comparison values within the specified range.
	0x0605 (1541)	Warning	Comparator module-No comparison position configured	Configure comparison values.
	0x0606 (1542)	Warning	Comparator module-System does not enable hardware comparison output for this axis	Enable hardware comparison output in the axis setting interface.
	0x0607 (1543)	Warning	Comparator module-Interrupt number out of range	Configure the interrupt number within the range of 0-16.
	0x0608 (1544)	Warning	Comparator module-Undefined interrupt function	Generate the corresponding interrupt function in the program.
	0x0701 (1793)	Warning	One-dimensional step length comparison module-Axis ID exceeds maximum value	Configure the correct axis ID.
	0x0702 (1794)	Warning	One-dimensional step length comparison module-This axis does not belong to encoder axis	
	0x0703 (1795)	Warning	One-dimensional step comparison module-Axis number not configured	
	0x0704 (1796)	Warning	One-dimensional step length comparison module-Starting position out of range	Configure the position value within the specified range.
	0x0705 (1797)	Warning	One-dimensional step length comparison module-End position out of range	
	0x0706 (1798)	Warning	One-dimensional step length comparison module-Single step position out of range	
	0x0707	Warning	One-dimensional step length	Configure the position parameters.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(1799)		comparison module-Starting position not configured	
	0x0708 (1800)	Warning	One-dimensional step comparison module-End position not configured	
	0x0709 (1801)	Warning	One-dimensional step length comparison module-Single step position not configured	
	0x070A (1802)	Warning	One-dimensional step comparison module-System does not enable hardware comparison output for this axis	Enable hardware comparison output in the axis setting interface.
	0x070B (1803)	Warning	One-dimensional step length comparison module-Interrupt number out of range	Configure the interrupt number within the range of 0–16.
	0x070C (1804)	Warning	One-dimensional step length comparison module-Undefined interrupt function	Generate the corresponding interrupt function in the program.
	0x070D (1805)	Warning	Linear mode-The starting position is less than the ending position, and the single step position is positive	The starting position is less than the ending position, and the single step position is positive.
	0x070E (1806)	Warning	Linear mode-The starting position is greater than the ending position, and the single-step position is negative	The starting position is greater than the ending position, and the single-step position is negative.
	0x0801 (2049)	Warning	Linear array comparison module-Axis ID exceeds maximum value	Configure the correct axis ID.
	0x0802 (2050)	Warning	Linear array comparison module-This axis does not belong to encoder axis	
	0x0803 (2051)	Warning	Linear array comparison module-Axis number not configured	
	0x0804 (2052)	Warning	Linear array comparison module-Array not configured	Configure array position.
	0x0805 (2053)	Warning	Linear array comparison module-Array greater than 100	The length of the array is less than 100.
	0x0806 (2054)	Warning	Linear array comparison module-Array size out of array range	The length of the array is less than array boundary.
	0x0807 (2055)	Warning	Linear array comparison module-Array size not configured	Configure array size.
	0x0808 (2056)	Warning	Linear array comparison module-System does not enable hardware comparison	Enable hardware comparison output in the axis setting interface.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			output for this axis	
	0x0809 (2057)	Warning	Linear array comparison module-Interrupt number out of range	Configure the interrupt number within the range of 0–16.
	0x080A (2058)	Warning	Linear array comparison module-Undefined interrupt function	Generate the corresponding interrupt function in the program.
	0x080B (2059)	Warning	Array size ≤ 0	Array length is greater than 0.
	0x0901 (2305)	Warning	Probe module-Axis ID exceeds maximum value	Configure the correct axis ID.
	0x0902 (2306)	Warning	Probe module-This axis does not belong to encoder axis	
	0x0903 (2307)	Warning	Probe module-No axis number configured	
	0x0904 (2308)	Warning	Probe module-Probe number parameter error	Probe number range: 0–1.
	0x0905 (2309)	Warning	Probe module-No probe number configured	Configure probe number.
	0x0906 (2310)	Warning	Probe module-Edge parameter error	Edge parameter range: 0–2
	0x0907 (2311)	Warning	Probe module-Edge parameters not configured	Configure edge parameters.
	0x0908 (2312)	Warning	Probe module-Trigger mode parameter error	Mode parameter range: 0–1
	0x0909 (2313)	Warning	Probe module-Window start parameter out of range	Configure position parameters within the specified range.
	0x090A (2314)	Warning	Probe module-Window end parameter out of range	Configure position parameters within the specified range.
	0x090B (2315)	Warning	Probe module-Probe not enabled	Enable probe in the axis setting interface.
	0x090C (2316)	Warning	In linear mode, the initial value is greater than or equal to the end value	In linear mode, the initial value is less than the end value.
	0x0A01 (2561)	Warning	Linear mode configuration-Axis ID exceeds maximum value	Configure the correct axis ID.
	0x0A02 (2562)	Warning	Linear mode configuration-This axis does not belong to encoder axis	
	0x0A03 (2563)	Warning	Linear mode configuration-No axis number configured	
	0x0A04 (2564)	Warning	Linear mode configuration-Mode selection parameter error (0–1)	Mode selection parameter 0–1
	0x0A05 (2565)	Warning	Linear mode configuration-Mode selection parameter not configured	Mode parameters are not configured.
	0x0A06	Warning	Linear mode	Configure software limit parameters.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(2566)		configuration-Software limit enable parameter not configured	
	0x0A07 (2567)	Warning	Linear mode configuration-Positive limit configuration out of range value	Configure the position value within the specified range.
	0x0A08 (2568)	Warning	Linear mode configuration-Negative limit position out of range value	
	0x0A09 (2569)	Warning	Linear mode configuration-Period value position out of range value	
	0x0A0A (2570)	Warning	Linear mode configuration-Positive limit parameter not configured	Configure positive limit parameters.
	0x0A0B (2571)	Warning	Linear mode configuration-Negative limit parameter not configured	Configure negative limit parameters.
	0x0A0C (2572)	Warning	Linear mode configuration-Period value parameter not configured	Configure cycle values.
	0x0A0D (2573)	Warning	The period value is zero	Cycle values must not be zero.
	0x0A0E (2574)	Warning	Negative limit is greater than positive limit	Positive limit is greater than negative limit.
	0x0B01 (2817)	Warning	Gear ratio mode configuration-Axis ID exceeds maximum value	Configure the correct axis ID.
	0x0B02 (2818)	Warning	Gear ratio mode configuration-This axis does not belong to encoder axis	
	0x0B03 (2819)	Warning	Gear ratio mode configuration-No axis number configured	
	0x0B04 (2820)	Warning	Gear ratio mode configuration-Parameters for one revolution of encoder not configured	Configure parameters for one revolution of encoder.
	0x0B05 (2821)	Warning	Gear ratio mode configuration-Parameters for one revolution of worktable not configured	Configure parameters for one revolution of worktable.
	0x0B06 (2822)	Warning	Gear ratio mode configuration-Maximum value of gear ratio numerator parameter not configured	Configure gear ratio numerator parameter.
	0x0B07	Warning	Gear ratio mode	Configure gear ratio denominator

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(2823)		configuration-Gear ratio denominator parameter not configured	parameter.
0x0013(19)	0x0001(1)	Warning	Axle number out of range	Check the axis number setting.
	0x0002(2)	Warning	Axis number does not exist in CANopen configuration or PDO configuration error	Check CANopen configuration setting.
	0x0003(3)	Warning	Absolute position instruction speed ≤ zero	Check the speed parameter in the instruction.
	0x0004(4)	Warning	Absolute position instruction speed ≤ zero	
	0x0005(5)	Warning	Speed mode instruction speed ≤ zero	
	0x0006(6)	Warning	Jog instruction speed ≤ zero	
	0x0007(7)	Warning	Absolute position instruction DEC speed ≤ zero	Check the DEC speed parameter in the instruction.
	0x0008(8)	Warning	Absolute position instruction DEC speed ≤ zero	Check the DEC speed parameter in the instruction.
	0x0009(9)	Warning	Speed mode instruction DEC speed ≤ zero	Check the DEC speed parameter in the instruction.
	0x000A(10)	Warning	Jog instruction DEC speed ≤ zero	Check the DEC speed parameter in the instruction.
	0x000B(11)	Warning	Homing failed	Check whether disconnection occurred.
	0x000C (12)	Warning	Homing timeout	Check CANopen configuration PDO setting.
	0x000D (13)	Warning	The axis is not enabled, and the current instruction cannot be executed	Enable CANopen axis.
	0x000E(14)	Warning	Not in "Fault Stop State", the reset axis error instruction cannot be executed	Check the axis state.
	0x000F(15)	Warning	The axis is in the "Stop" state, and the current instruction cannot be executed	
	0x0010(16)	Warning	The axis is homing, and the current instruction cannot be executed	
	0x0011(17)	Warning	The axis is moving continuously, and the current instruction cannot be executed	
	0x0012(18)	Warning	The axis is positioning, and the current instruction cannot be executed	
	0x0013(19)	Warning	The axis is in the "Fault Stop" state, and the current instruction cannot be executed	
	0x0014(20)	Warning	Axis enable timeout	Check whether CANopen

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0015(21)	Warning	CANopen is not configured	configuration setting is correct.
	0x0016(22)	Warning	Fault reset timeout	Check whether the line connection is normal
	0x0017(23)	Warning	SDO write timeout	
	0x0018(24)	Warning	SDO read timeout	
	0x0019(25)	Warning	SDO instruction error	
	0x001A(26)	Warning	Software limit reached in axis operation	Check the software limit settings.
	0x001B(27)	Warning	Axis absolute positioning failure	Check whether the line connection is normal
	0x001C (28)	Warning	Axis relative positioning failure	
	0x001D (29)	Warning	Homing speed set incorrectly	Check whether the instruction speed parameters are set correctly.
	0x001E(30)	Warning	Axis halt instruction execution timeout	Check whether the line connection is normal
	0x001F(31)	Warning	Homing approach speed is set incorrectly	Check whether the homing approach speed parameters are set correctly in the CANopen configuration.
	0x0020(32)	Warning	Homing acceleration set incorrectly	Check whether the homing ACC parameters are set correctly in the CANopen configuration.
	0x0021(33)	Warning	Speed operation instruction execution failure	Check whether the line connection is normal
	0x0022(34)	Warning	Jog instruction execution failure	
	0x0023(35)	Warning	Enable instruction is not allowed to be called on the same axis	Check whether the Power instruction is called twice for the same axis number.
	0x0024(36)	General error	The slave corresponding to the axis is offline	Check the communication state of the slave or whether the slave is enabled.
0x0030(48)	0x0001(1)	General error	Module configuration fault	Check whether the network configuration corresponds to the physical configuration of the module
	0x0002(2)	Warning	Incorrect module parameter setting	Check module parameter configuration
0x0031(49)	0x0001(1)	General error	Digital input module configuration fault	Check whether the network configuration corresponds to the physical configuration of the module
	0x0002(2)	Warning	Digital input module parameter configuration failure	Check module parameter configuration
	0x2001 (8193)	General error	Digital output module configuration failure	Check whether the network configuration corresponds to the physical configuration of the module
	0x2002 (8194)	Warning	Digital output module parameter configuration failure	Check module parameter configuration
	0x2003	Warning	Digital output module output	Check the module output port power

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(8195)		port power supply failure	supply
	0x2004 (8196)	Warning	Digital output module output failure	Check whether the module output port load exceeds the specification range
0x0032(50)	0x0001(1)	General error	Analog input module configuration failure	Check whether the network configuration corresponds to the physical configuration of the module
	0x0012(18)	Warning	Analog input channel 0 parameter configuration failure	Check channel 0 parameter configuration
	0x0015(21)	Warning	Analog input channel 0 signal source open circuit fault	Check the physical connection of Channel 0 signal source
	0x0016(22)	Warning	Analog input channel 0 sampling signal out of limit fault	Check whether the channel 0 sampling signal exceeds the chip limit
	0x0017(23)	Warning	Analog input channel 0 sampling signal exceeds upper limit fault	Check whether the channel 0 sampling signal exceeds the upper range
	0x0018(24)	Warning	Analog input channel 0 sampling signal exceeds the lower range	Check whether the channel 0 sampling signal exceeds the lower range
	0x0022(34)	Warning	Analog input channel 1 parameter configuration failure	Check channel 1 parameter configuration
	0x0025(37)	Warning	Analog input channel 1 signal source open circuit fault	Check the physical connection of Channel 1 signal source
	0x0026(38)	Warning	Analog input channel 1 sampling signal out of limit fault	Check whether the channel 1 sampling signal exceeds the chip limit
	0x0027(39)	Warning	Analog input channel 1 sampling signal exceeds upper limit fault	Check whether the channel 1 sampling signal exceeds the upper range
	0x0028(40)	Warning	Analog input channel 1 sampling signal exceeds the lower range	Check whether the channel 1 sampling signal exceeds the lower range
	0x0032(50)	Warning	Analog input channel 2 parameter configuration failure	Check channel 2 parameter configuration
	0x0035(53)	Warning	Analog input channel 2 signal source open circuit fault	Check the physical connection of Channel 2 signal source
	0x0036(54)	Warning	Analog input channel 2 sampling signal out of limit fault	Check whether the channel 2 sampling signal exceeds the chip limit
	0x0037(55)	Warning	Analog input channel 2 sampling signal exceeds upper limit fault	Check whether the channel 2 sampling signal exceeds the upper range
	0x0038(56)	Warning	Analog input channel 2 sampling signal exceeds the lower range	Check whether the channel 2 sampling signal exceeds the lower range
	0x0042(66)	Warning	Analog input channel 3 parameter configuration failure	Check channel 3 parameter configuration

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0045(69)	Warning	Analog input channel 3 signal source open circuit fault	Check the physical connection of Channel 3 signal source
	0x0046(70)	Warning	Analog input channel 3 sampling signal out of limit fault	Check whether the channel 3 sampling signal exceeds the chip limit
	0x0047(71)	Warning	Analog input channel 3 sampling signal exceeds upper limit fault	Check whether the channel 3 sampling signal exceeds the upper range
	0x0048(72)	Warning	Analog input channel 3 sampling signal exceeds the lower range	Check whether the channel 3 sampling signal exceeds the lower range
	0x2001 (8193)	General error	Analog output module configuration fault	Check whether the network configuration corresponds to the physical configuration of the module
	0x2003 (8195)	Warning	Analog output module output port power supply failure	Check the module output port power supply
	0x2012 (8210)	Warning	Analog output channel 0 parameter configuration failure	Check channel 0 parameter configuration
	0x2014 (8212)	Warning	Analog output channel 0 output fault	Check channel 0 output for short/open circuit
	0x2022 (8226)	Warning	Analog output channel 1 parameter configuration failure	Check channel 1 parameter configuration
	0x2024 (8228)	Warning	Analog output channel 1 output fault	Check channel 1 output for short/open circuit
	0x2032 (8242)	Warning	Analog output channel 2 parameter configuration failure	Check channel 2 parameter configuration
	0x2034 (8244)	Warning	Analog output channel 2 output fault	Check channel 2 output for short/open circuit
	0x2042 (8258)	Warning	Analog output channel 3 parameter configuration failure	Check channel 3 parameter configuration
	0x2044 (8260)	Warning	Analog output channel 3 output fault	Check channel 3 output for short/open circuit
0x0033(51)	0x0001(1)	General error	Temperature sampling module configuration failure	Check whether the network configuration corresponds to the physical configuration of the module
	0x0012(18)	Warning	Temperature sampling channel 0 parameter configuration failure	Check channel 0 parameter configuration
	0x0015(21)	Warning	Temperature sampling channel 0 signal source open circuit fault	Check the physical connection of Channel 0 signal source
	0x0017(23)	Warning	Temperature sampling channel 0 sampling signal exceeds the upper range	Check whether the channel 0 sampling signal exceeds the upper range
	0x0018(24)	Warning	Temperature sampling channel 0 sampling signal exceeds the lower range	Check whether the channel 0 sampling signal exceeds the lower range
	0x0022(34)	Warning	Temperature sampling channel	Check channel 1 parameter

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			1 parameter configuration failure	configuration
	0x0025(37)	Warning	Temperature sampling channel 1 signal source open circuit fault	Check the physical connection of Channel 1 signal source
	0x0027(39)	Warning	Temperature sampling channel 1 sampling signal exceeds the upper range	Check whether the channel 1 sampling signal exceeds the upper range
	0x0028(40)	Warning	Temperature sampling channel 1 sampling signal exceeds the lower range	Check whether the channel 1 sampling signal exceeds the lower range
	0x0032(50)	Warning	Temperature sampling channel 2 parameter configuration failure	Check channel 2 parameter configuration
	0x0035(53)	Warning	Temperature sampling channel 2 signal source open circuit fault	Check the physical connection of Channel 2 signal source
	0x0037(55)	Warning	Temperature sampling channel 2 sampling signal exceeds the upper range	Check whether the channel 2 sampling signal exceeds the upper range
	0x0038(56)	Warning	Temperature sampling channel 2 sampling signal exceeds the lower range	Check whether the channel 2 sampling signal exceeds the lower range
	0x0042(66)	Warning	Temperature sampling channel 3 parameter configuration failure	Check channel 3 parameter configuration
	0x0045(69)	Warning	Temperature sampling channel 3 signal source open circuit fault	Check the physical connection of Channel 3 signal source
	0x0047(71)	Warning	Temperature sampling channel 3 sampling signal exceeds the upper range	Check whether the channel 3 sampling signal exceeds the upper range
	0x0048(72)	Warning	Temperature sampling channel 3 sampling signal exceeds the lower range	Check whether the channel 3 sampling signal exceeds the lower range
0x0034(52)	0x0001(1)	General error	Module configuration does not match	Check whether the module network configuration corresponds to the physical configuration
	0x0002(2)	Warning	Module parameter configuration fault	Check the module configuration parameters
	0x0003(3)	Warning	Module output port power supply fault	Check the module output port power supply
	0x0004(4)	Warning	Module output port fault	Check the module output port
	0x0012(18)	Warning	Module channel 0 parameter configuration fault	Check the parameter configuration of module channel 0
	0x0014(20)	Warning	Module channel 0 output fault	Check the output of module channel 0
	0x0015(21)	Warning	Module channel 0 signal	Check the physical connection of

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			source open circuit fault	channel 0 signal source
	0x0016(22)	Warning	Module channel 0 sampling signal out of limit fault	Check whether the sampling signal of module channel 0 exceeds the limit
	0x0017(23)	Warning	Module channel 0 sampling signal exceeds the upper range	Check whether the sampling signal of module channel 0 exceeds the upper range
	0x0018(24)	Warning	Module channel 0 sampling signal exceeds the lower range	Check whether the sampling signal of module channel 0 exceeds the lower range
	0x0022(34)	Warning	Module channel 1 parameter configuration fault	Check the parameter configuration of module channel 1
	0x0023(35)	Warning	Module channel 1 output fault	Check the output of module channel 1
	0x0024(36)	Warning	Module channel 1 signal source open circuit fault	Check the physical connection of channel 1 signal source
	0x0025(37)	Warning	Module channel 1 sampling signal out of limit fault	Check whether the sampling signal of module channel 1 exceeds the limit
	0x0026(38)	Warning	Module channel 1 sampling signal exceeds the upper range	Check whether the sampling signal of module channel 1 exceeds the upper range
	0x0027(39)	Warning	Module channel 1 sampling signal exceeds the lower range	Check whether the sampling signal of module channel 1 exceeds the lower range
	0x0028(40)	Warning	Module output port power supply fault	Check the module output port power supply
0x0040(64)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code	Check the master connected with PLC
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program
0x0041(65)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or	Check the element address

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			sent exceeds the storage space of the element)	
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code	Check the master connected with PLC
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program
0x0042(66)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of	Check the master connected with PLC

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			elements exceeds the maximum limit specified by the function code	
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program
0x0043(67)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code	Check the master connected with PLC
	0x000C (12)	Warning	The received slave address does not match the requested	Check the slave connected with PLC

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			slave address	
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program
0x0044(68)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code	Check the master connected with PLC
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
				of the upper computer; Re-download the program
0x0045(69)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code	Check the master connected with PLC
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program
0x0046(70)	0x0001(1)	Warning	COM port parameters not configured	Choose the serial freeport in the upper computer and configure the parameters of the serial port

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0002(2)	Warning	Transmit length or receive length setting error	Check the transmit length or receive length
	0x0003(3)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the receive or transmit buffer element address and the transmit or receive length of instruction parameters
	0x0004(4)	Warning	Port setting error	Check whether the Port setting in the instruction is correct
	0x0005(5)	Warning	Transmit instruction execution failed	Retry
	0x0006(6)	Warning	Receive instruction execution failed	Retry
	0x0007(7)	Warning	Incomplete received data	Check the length of data transmitted by the sender
	0x0008(8)	Warning	Receive data timeout	Check whether the serial line connection is normal
	0x0009(9)	Warning	Instruction execution failed	Retry
0x0047(71)	0x0001(1)	Warning	COM port parameters not configured	Choose the serial freeport in the upper computer and configure the parameters of the serial port
	0x0002(2)	Warning	Transmit length or receive length setting error	Check the transmit length or receive length
	0x0003(3)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the receive or transmit buffer element address and the transmit or receive length of instruction parameters
	0x0004(4)	Warning	Port setting error	Check whether the Port setting in the instruction is correct
	0x0005(5)	Warning	Transmit instruction execution failed	Retry
	0x0006(6)	Warning	Receive instruction execution failed	Retry
	0x0007(7)	Warning	Incomplete received data	Check the length of data transmitted by the sender
	0x0008(8)	Warning	Receive data timeout	Check whether the serial line connection is normal
	0x0009(9)	Warning	Instruction execution failed	Retry
0x0048(72)	0x0001(1)	Warning	COM port parameters not configured	Choose the serial freeport in the upper computer and configure the parameters of the serial port
	0x0002(2)	Warning	Transmit length or receive length setting error	Check the transmit length or receive length
	0x0003(3)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the receive or transmit buffer element address and the transmit or receive length of instruction parameters
	0x0004(4)	Warning	Port setting error	Check whether the Port setting in the instruction is correct

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0005(5)	Warning	Transmit instruction execution failed	Retry
	0x0006(6)	Warning	Receive instruction execution failed	Retry
	0x0007(7)	Warning	Incomplete received data	Check the length of data transmitted by the sender
	0x0008(8)	Warning	Receive data timeout	Check whether the serial line connection is normal
	0x0009(9)	Warning	Instruction execution failed	Retry
0x0049(73)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device fault	Check whether the master connected with the PLC is configured correctly.
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user.	Check whether the serial connection is normal.
	0x0007(7)	Warning	The communication connection is disconnected.	Check whether the line connection is normal.
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol.	Check whether the baud rate, data bit and parity bit are configured correctly.
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly.
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element).	Check the element address.
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code.	Check the master connected with PLC.
	0x000C(12)	Warning	The received slave address does not match the requested slave address.	Check the slave connected with PLC.
	0x000D(13)	Warning	The received function code does not match the requested	Check the slave connected with PLC.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			function code.	
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the host controller. Re-download the program.
	0x000F(15)	Warning	The input port number is out of range.	Check whether the input port number of the function block corresponds to the actual port.
	0x0010(16)	Warning	The serial port is not opened.	Check whether the corresponding port for ModbusRTU master is checked in the configuration tree on the right side.
	0x0011(17)	Warning	Slave address setting error	The slave address setting is not within the range of 0 to 255.
	0x0012(18)	Warning	Function code setting error	The function code setting does not meet the Modbus requirements; check whether the function code setting is correct.
	0x0013(19)	Warning	Incorrect data count setting for the corresponding function code	The data count set by this function code does not meet the Modbus requirements.
	0x0014(20)	Warning	Data buffer cache size setting error	Check the size of the defined data buffer cache to ensure that it is not less than the required amount of data.
	0x0015(21)	Warning	Two function blocks are enabled simultaneously on the same port.	Check whether the function blocks are enabled simultaneously.
0x004A(74)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device fault	Check whether the master connected with the PLC is configured correctly.
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal.
	0x0007(7)	Warning	The communication connection is disconnected.	Check whether the line connection is normal.
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol.	Check whether the baud rate, data bit and parity bit are configured correctly.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly.
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element).	Check the element address.
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code.	Check the master connected with PLC.
	0x000C(12)	Warning	The received slave address does not match the requested slave address.	Check the slave connected with PLC.
	0x000D(13)	Warning	The received function code does not match the requested function code.	Check the slave connected with PLC.
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the host controller. Re-download the program.
	0x000F(15)	Warning	The input port number is out of range.	Check whether the input port number of the function block corresponds to the actual port.
	0x0010(16)	Warning	The serial port is not opened.	Check whether the corresponding port for ModbusRTU master is checked in the configuration tree on the right side.
	0x0011(17)	Warning	Slave address setting error	The slave address setting is not within the range of 0 to 255.
	0x0012(18)	Warning	Function code setting error	The function code setting does not meet the Modbus requirements; check whether the function code setting is correct.
	0x0013(19)	Warning	Incorrect data count setting for the corresponding function code	The data count set by this function code does not meet the Modbus requirements.
	0x0014(20)	Warning	Data buffer cache size setting error	Check the size of the defined data buffer cache to ensure that it is not less than the required amount of data.
	0x0015(21)	Warning	Two function blocks are enabled simultaneously on the same port.	Check whether the function blocks are enabled simultaneously.
0x0050(80)	0x0001(1)	General error	CANopen communication error	Please check whether the CAN network line connection is normal, ensure that there is no reverse

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
				connection, short connection or open circuit between CANH and CANL, and check whether the terminal resistance is connected correctly, and whether the baud rate of CAN communication matches.
	0x0002(2)	General error	CANopen configuration error	Check whether the upper computer configuration matches the actual situation
	0x0003(3)	Warning	CANopen load rate is too high	Detect whether too many PDOs are configured, and there are devices on the fieldbus that transmit CAN messages autonomously, such as CAN analyzers or multiple CANopen masters. This situation may lead to poor communication status, data loss and other problems.
0x0051(81)	0x0001(1)	Serious error	Internal error in CAN module	Restart the PLC or contact technical support.
	0x0002(2)	General error	CAN free port instruction parameter setting is incorrect.	Check the input parameters of the function block.
	0x0003(3)	General error	CAN free port data sending timeout	1. Check whether the CAN baud rate configuration is correct. 2. Check whether the CAN hardware connection is normal. 3. Check the terminal matching resistor.
	0x0004(4)	General error	CAN free port data receiving timeout	1. Check whether the CAN baud rate configuration is correct. 2. Check whether the CAN hardware connection is normal. 3. Check the terminal matching resistor.
	0x0005(5)	General error	CAN bus error	1. Reduce environmental interference. 2. Check the baud rate configuration. 3. Check the terminal matching resistor.
	0x0006(6)	General error	CAN free port is not configured.	Please configure CAN 2.0 communication in the CAN options.
0x0080 (128)	0x1020 (4128)	Serious error	Wrong number of module matches	Update the version of the host controller or contact technical support.
	0x1040 (4160)	Serious error	Connection 0 module length matching exception	

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x1041 (4161)	Serious error	Connection 1 module length matching exception	
	0x1042 (4162)	Serious error	Connection 2 module length matching exception	
	0x1043 (4163)	Serious error	Connection 3 module length matching exception	
	0x1044 (4164)	Serious error	Connection 4 module length matching exception	
	0x1045 (4165)	Serious error	Connection 5 module length matching exception	
	0x1046 (4166)	Serious error	Connection 6 module length matching exception	
	0x1047 (4167)	Serious error	Connection 7 module length matching exception	
	0x1048 (4168)	Serious error	Connection 8 module length matching exception	
	0x1049 (4169)	Serious error	Connection 9 module length matching exception	
	0x104A (4170)	Serious error	Connection 10 module length matching exception	
	0x104B (4171)	Serious error	Connection 11 module length matching exception	
	0x104C (4172)	Serious error	Connection 12 module length matching exception	
	0x104D (4173)	Serious error	Connection 13 module length matching exception	
	0x104E (4174)	Serious error	Connection 14 module length matching exception	
	0x104F (4175)	Serious error	Connection 15 module length matching exception	
	0x1060 (4192)	Serious error	Connection 0 module input length exception	
	0x1061 (4193)	Serious error	Connection 1 module input length exception	
	0x1062 (4194)	Serious error	Connection 2 module input length exception	
	0x1063 (4195)	Serious error	Connection 3 module input length exception	
	0x1064 (4196)	Serious error	Connection 4 module input length exception	
	0x1065 (4197)	Serious error	Connection 5 module input length exception	
	0x1066 (4198)	Serious error	Connection 6 module input length exception	
	0x1067 (4199)	Serious error	Connection 7 module input length exception	
	0x1068 (4200)	Serious error	Connection 8 module input length exception	

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x1069 (4201)	Serious error	Connection 9 module input length exception	
	0x106A (4202)	Serious error	Connection 10 module input length exception	
	0x106B (4203)	Serious error	Connection 11 module input length exception	
	0x106C (4204)	Serious error	Connection 12 module input length exception	
	0x106D (4205)	Serious error	Connection 13 module input length exception	
	0x106E (4206)	Serious error	Connection 14 module input length exception	
	0x106F (4207)	Serious error	Connection 15 module input length exception	
	0x1080 (4224)	Serious error	Connection 0 module output length exception	
	0x1081 (4225)	Serious error	Connection 1 module output length exception	
	0x1082 (4226)	Serious error	Connection 2 module output length exception	
	0x1083 (4227)	Serious error	Connection 3 module output length exception	
	0x1084 (4228)	Serious error	Connection 4 module output length exception	
	0x1085 (4229)	Serious error	Connection 5 module output length exception	
	0x1086 (4230)	Serious error	Connection 6 module output length exception	
	0x1087 (4231)	Serious error	Connection 7 module output length exception	
	0x1088 (4232)	Serious error	Connection 8 module output length exception	
	0x1089 (4233)	Serious error	Connection 9 module output length exception	
	0x108A (4234)	Serious error	Connection 10 module output length exception	
	0x108B (4235)	Serious error	Connection 11 module output length exception	
	0x108C (4236)	Serious error	Connection 12 module output length exception	
	0x108D (4237)	Serious error	Connection 13 module output length exception	
	0x108E (4238)	Serious error	Connection 14 module output length exception	
	0x108F (4239)	Serious error	Connection 15 module output length exception	
	0x10A0	Serious	Connection 0 element	

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(4256)	error	matching exception	
	0x10A1 (4257)	Serious error	Connection 1 element matching exception	
	0x10A2 (4258)	Serious error	Connection 2 element matching exception	
	0x10A3 (4259)	Serious error	Connection 3 element matching exception	
	0x10A4 (4260)	Serious error	Connection 4 element matching exception	
	0x10A5 (4261)	Serious error	Connection 5 element matching exception	
	0x10A6 (4262)	Serious error	Connection 6 element matching exception	
	0x10A7 (4263)	Serious error	Connection 7 element matching exception	
	0x10A8 (4264)	Serious error	Connection 8 element matching exception	
	0x10A9 (4265)	Serious error	Connection 9 element matching exception	
	0x10AA (4266)	Serious error	Connection 10 element matching exception	
	0x10AB (4267)	Serious error	Connection 11 element matching exception	
	0x10AC (4268)	Serious error	Connection 12 element matching exception	
	0x10AD (4269)	Serious error	Connection 13 element matching exception	
	0x10AE (4270)	Serious error	Connection 14 element matching exception	
	0x10AF (4271)	Serious error	Connection 15 element matching exception	
	0x3000 (12288)	Warning	Connection 0 connection not established	
	0x3001 (12289)	Warning	Connection 1 connection not established	
	0x3002 (12290)	Warning	Connection 2 connection not established	
	0x3003(12291)	Warning	Connection 3 connection not established	
	0x3004 (12292)	Warning	Connection 4 connection not established	
	0x3005 (12293)	Warning	Connection 5 connection not established	
	0x3006 (12294)	Warning	Connection 6 connection not established	
	0x3007 (12295)	Warning	Connection 7 connection not established	

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x3008 (12296)	Warning	Connection 8 connection not established	
	0x3009 (12297)	Warning	Connection 9 connection not established	
	0x300A (12298)	Warning	Connection 10 connection not established	
	0x300B (12299)	Warning	Connection 11 connection not established	
	0x300C (12300)	Warning	Connection 12 connection not established	
	0x300D (12301)	Warning	Connection 13 connection not established	
	0x300E (12302)	Warning	Connection 14 connection not established	
	0x300F (12303)	Warning	Connection 15 connection not established	
	0x3020 (12320)	Warning	Connection 0 path error	Check the configuration path.
	0x3021 (12321)	Warning	Connection 1 path error	
	0x3022 (12322)	Warning	Connection 2 path error	
	0x3023 (12323)	Warning	Connection 3 path error	
	0x3024 (12324)	Warning	Connection 4 path error	
	0x3025 (12325)	Warning	Connection 5 path error	
	0x3026 (12326)	Warning	Connection 6 path error	
	0x3027 (12327)	Warning	Connection 7 path error	
	0x3028 (12328)	Warning	Connection 8 path error	
	0x3029 (12329)	Warning	Connection 9 path error	
	0x302A (12330)	Warning	Connection 10 path error	
	0x302B (12331)	Warning	Connection 11 path error	
	0x302C (12332)	Warning	Connection 12 path error	
	0x302D (12333)	Warning	Connection 13 path error	
	0x302E (12334)	Warning	Connection 14 path error	
	0x302F	Warning	Connection 15 path error	

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(12335)			
	0x3040 (12352)	Warning	Connection 0 transmission data size mismatch	Check the size of the data matching between the sender and receiver.
	0x3041 (12353)	Warning	Connection 1 transmission data size mismatch	
	0x3042 (12354)	Warning	Connection 2 transmission data size mismatch	
	0x3043 (12355)	Warning	Connection 3 transmission data size mismatch	
	0x3044 (12356)	Warning	Connection 4 transmission data size mismatch	
	0x3045 (12357)	Warning	Connection 5 transmission data size mismatch	
	0x3046 (12358)	Warning	Connection 6 transmission data size mismatch	
	0x3047 (12359)	Warning	Connection 7 transmission data size mismatch	
	0x3048 (12360)	Warning	Connection 8 transmission data size mismatch	
	0x3049 (12361)	Warning	Connection 9 transmission data size mismatch	
	0x304A (12362)	Warning	Connection 10 transmission data size mismatch	
	0x304B (12363)	Warning	Connection 11 transmission data size mismatch	
	0x304C (12364)	Warning	Connection 12 transmission data size mismatch	
	0x304D (12365)	Warning	Connection 13 transmission data size mismatch	
	0x304E (12366)	Warning	Connection 14 transmission data size mismatch	
	0x304F (12367)	Warning	Connection 15 transmission data size mismatch	
	0x3060 (12384)	Warning	Connection 0 other errors	Contact technical support.
	0x3061 (12385)	Warning	Connection 1 other errors	
	0x3062 (12386)	Warning	Connection 2 other errors	
	0x3063 (12387)	Warning	Connection 3 other errors	
	0x3064 (12388)	Warning	Connection 4 other errors	
	0x3065 (12389)	Warning	Connection 5 other errors	
	0x3066 (12390)	Warning	Connection 6 other errors	

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x3067 (12391)	Warning	Connection 7 other errors	
	0x3068 (12392)	Warning	Connection 8 other errors	
	0x3069 (12393)	Warning	Connection 9 other errors	
	0x306A (12394)	Warning	Connection 10 other errors	
	0x306B (12395)	Warning	Connection 11 other errors	
	0x306C (12396)	Warning	Connection 12 other errors	
	0x306D (12397)	Warning	Connection 13 other errors	
	0x306E (12398)	Warning	Connection 14 other errors	
	0x306F (12399)	Warning	Connection 15 other errors	
	0x30A0 (12448)	Warning	Connection 0 communication timeout	Check the cables, connection ports, connection mode, and attribute ID.
	0x30A1 (12449)	Warning	Connection 1 communication timeout	
	0x30A2 (12450)	Warning	Connection 2 communication timeout	
	0x30A3 (12451)	Warning	Connection 3 communication timeout	
	0x30A4 (12452)	Warning	Connection 4 communication timeout	
	0x30A5 (12453)	Warning	Connection 5 communication timeout	
	0x30A6 (12454)	Warning	Connection 6 communication timeout	
	0x30A7 (12455)	Warning	Connection 7 communication timeout	
	0x30A8 (12456)	Warning	Connection 8 communication timeout	
	0x30A9 (12457)	Warning	Connection 9 communication timeout	
	0x30AA (12458)	Warning	Connection 10 communication timeout	
	0x30AB (12459)	Warning	Connection 11 communication timeout	
	0x30AC (12460)	Warning	Connection 12 communication timeout	
	0x30AD (12461)	Warning	Connection 13 communication timeout	
	0x30AE	Warning	Connection 14 communication	

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	(12462)		timeout	
	0x30AF (12463)	Warning	Connection 15 communication timeout	
	0x4020 (16416)	Serious error	Network configuration exception	Contact technical support.
	0x4040 (16448)	Serious error	Network initialization exception	
	0x4060 (16480)	Serious error	Thread attribute initialization failed	
	0x4080 (16512)	Serious error	Thread request stack failure	
	0x40A0 (16544)	Serious error	Thread setting scheduling policy failed	
	0x40C0 (16576)	Serious error	Thread priority setting failed	
	0x40E0 (16608)	Serious error	Failed to set parent thread inheritance policy	
	0x4100 (16640)	Serious error	Failed to create thread	
0x0090 (144)	0x0001(1)	General error	Failed to apply for master	Check whether the card software matches the background version; Restart PLC
	0x0002(2)	General error	Wrong master version	Check whether the single board software matches the background version
	0x0003(3)	General error	The number of PDO entries sent exceeds the maximum limit.	Check whether the number of PDOs sent exceeds the maximum limit.
	0x0004(4)	General error	The number of PDO configuration objects sent exceeds the maximum limit	Check whether the number of PDO configuration objects sent exceeds the maximum limit.
	0x0005(5)	General error	The number of PDO entries received exceeds the maximum limit	Check whether the number of PDOs entries received exceeds the maximum limit.
	0x0006(6)	General error	The number of PDO configuration objects received exceeds the maximum limit	Check whether the number of PDO configuration objects received exceeds the maximum limit.
	0x0007(7)	General error	The number of startup parameters exceeds the maximum limit	Check whether the number of startup parameters exceeds the maximum limit.
	0x0008(8)	General error	The number of servos exceeds the maximum limit	Check whether the number of configured servos exceeds the maximum limit.
	0x0009(9)	General error	The number of slaves exceeds the maximum limit	Check whether the number of configured slaves exceeds the maximum limit.
	0x000A(10)	General error	Wrong configuration type	Reserved

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x000B(11)	General error	Configured number does not match actual number of connections	Check whether the actual number of connected slaves is less than the configured number of slaves.
	0x000C (12)	General error	DC mode is not supported by slaves	Reserved
	0x000D (13)	General error	Wrong Slave type	Check whether the devices in the configuration match the actual connected devices
	0x000E(14)	General error	The number of mapped slaves exceeds the set value	Check whether the actual number of connected slaves is greater than the configured number of slaves.
	0x000F(15)	General error	Mapping slave transmit PDO communication exception	Reserved
	0x0010(16)	General error	Mapping slave receive PDO communication exception	Reserved
	0x0011(17)	General error	The slaves have been detected to be offline	Check whether the network among slaves is disconnected; Check whether the slaves are powered off
	0x0012(18)	General error	Failed to initialize slave parameters	Contact the manufacturer.
	0x0013(19)	General error	Network connection failure	Check whether the slaves are connected; Check whether all slaves are powered off
	0x0014(20)	General error	Unable to identify the number of slaves	Reserved
	0x0015(21)	Warning	Aperiodic communication timeout	Reserved
	0x0016(22)	Serious error	Failed to apply for master	Contact the manufacturer.
	0x0017(23)	Serious error	Illegal IO mapping	Reserved
	0x0018(24)	General error	Failed to write the homing parameters.	Check whether the values of the homing parameters are reasonable.
	0x0019(25)	General error	Failed to write the user startup parameters.	Check whether the values of the user startup parameters are reasonable.
	0x001A(26)	General error	The received frame check failed	1. Check whether the network cable is category 6 or above. 2. Investigate whether there are factors in the field that may cause communication interference.
	0x001B(27)	General error	Communication timeout for the master to read the slave state	1. Check whether the slave is disconnected. 2. Check whether the CPU load rate is too high.
	0x001C(28)	General error	Master exception reset	1. Check whether the slave state machine is abnormal.

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
				2. Check whether the slave is offline.
0x00A0 (160)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0006(6)	Warning	Modbus TCP master-slave connection timeout	Check whether the connection of network cable is normal, and whether the ip and port number are set correctly
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code	Check the master connected with PLC
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0011(17)	Warning	Slave ID setting error	Check whether the SlaveID parameter configuration of the instruction is correct.
	0x0012(18)	Warning	Function code setting error	Check whether the FunCode parameter configuration of the instruction is correct.
	0x0013(19)	Warning	Incorrect data quantity setting under the corresponding function code	Check whether the FunCode and DataCounts parameter configuration of the instruction is correct.
	0x0014(20)	Warning	The setting of data buffer size is incorrect.	Check whether the DataBuffer parameter configuration of the instruction is correct.
0x00A1 (161)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0006(6)	Warning	Modbus TCP master-slave connection timeout	Check whether the connection of network cable is normal, and whether the ip and port number are set correctly
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the	Check the master connected with PLC

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
			maximum limit specified by the function code	
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program
	0x0011(17)	Warning	Slave ID setting error	Check whether the SlaveID parameter configuration of the instruction is correct.
	0x0012(18)	Warning	Function code setting error	Check whether the FunCode parameter configuration of the instruction is correct.
	0x0013(19)	Warning	Incorrect data quantity setting under the corresponding function code	Check whether the FunCode and DataCounts parameter configuration of the instruction is correct.
	0x0014(20)	Warning	The setting of data buffer size is incorrect.	Check whether the DataBuffer parameter configuration of the instruction is correct.
0x00A2 (162)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0006(6)	Warning	Modbus TCP master-slave connection timeout	Check whether the connection of network cable is normal, and whether the ip and port number are set correctly
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code	Check the master connected with PLC
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program
0x00A3 (163)	0x0001(1)	Warning	Standard Modbus error, exception code 01, illegal function code	Check whether the configuration of function code accessed by master connected with PLC is legal
	0x0002(2)	Warning	Standard Modbus error, exception code 02, illegal register address	Check whether the address configuration accessed by the master connected with PLC is legal
	0x0003(3)	Warning	Standard Modbus error, exception code 03, illegal data value	Check whether the value included in the master's query to the server (or the slave) is not allowed.
	0x0004(4)	Warning	Standard Modbus error, exception code 04, slave device failure	Check whether the master connected with the PLC is configured correctly
	0x0005(5)	Warning	Communication timeout, the communication time exceeds the maximum communication time set by the user	Check whether the serial connection is normal
	0x0006(6)	Warning	Modbus TCP master-slave connection timeout	Check whether the connection of network cable is normal, and whether the ip and port number are set correctly
	0x0007(7)	Warning	The communication connection is disconnected	Check whether the line connection is normal
	0x0008(8)	Warning	The received data frame does not conform to the Modbus protocol	Check whether the baud rate, data bit and parity bit are configured correctly
	0x0009(9)	Warning	CRC/LRC check error	Check whether the baud rate, data

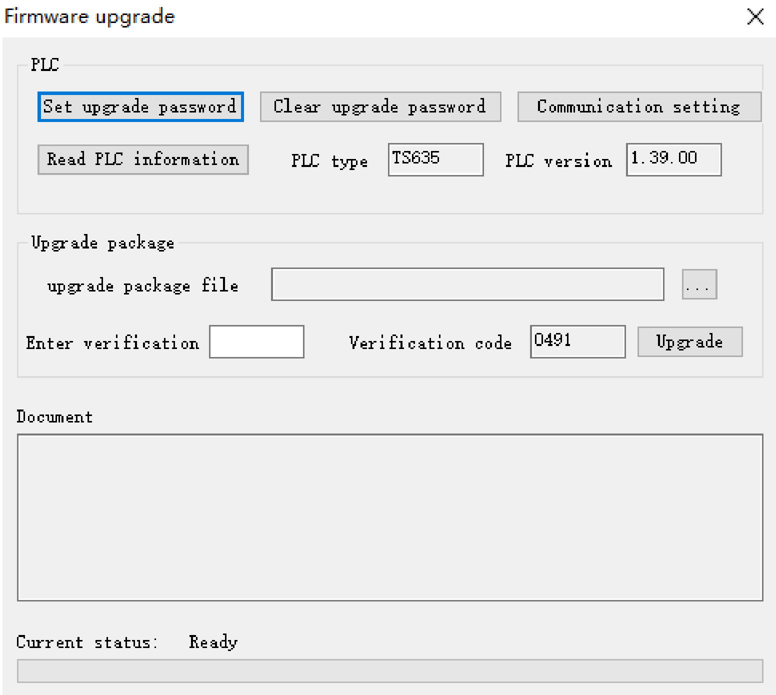
Main fault code	Sub fault code	Error level	Meaning of error	Solutions
				bit and parity bit are configured correctly
	0x000A(10)	Warning	Element address overflow (the amount of data received or sent exceeds the storage space of the element)	Check the element address
	0x000B(11)	Warning	The length of data received does not conform to the protocol or the number of elements exceeds the maximum limit specified by the function code	Check the master connected with PLC
	0x000C (12)	Warning	The received slave address does not match the requested slave address	Check the slave connected with PLC
	0x000D (13)	Warning	The received function code does not match the requested function code	Check the slave connected with PLC
	0x000E(14)	Warning	Instruction execution failed	Check the parameter configuration of the upper computer; Re-download the program
0x00B0 (176)	0x0001(1)	General error	Client connection failed	Check whether the server side is turned on; Check whether the IP address set by the client is the IP address of the server; Check whether the network cable connection is loose
	0x0002(2)	General error	Instruction parameter setting error	Check whether the data quantity setting value is less than or equal to 0
	0x0003(3)	General error	Instruction parameter element number setting error	The amount of data sent or received exceeds the capacity of the data transmitting or receiving area
	0x0004(4)	General error	Server listening failed	Server socket not created, recreate server socket
	0x0005(5)	General error	Transmit instruction execution failed	Check the network connection
	0x0006(6)	General error	Receive instruction execution failed	Check the network connection
0x00B8 (184)	0x0002(2)	General error	Instruction parameter setting error	Check whether the data quantity setting value is less than or equal to 0
	0x0003(3)	General error	Instruction parameter element number setting error	The amount of data sent or received exceeds the capacity of the data transmitting or receiving area
	0x0005(5)	General error	Transmit instruction execution failed	Check the network connection

Main fault code	Sub fault code	Error level	Meaning of error	Solutions
	0x0006(6)	General error	Receive instruction execution failed	Check the network connection
0x00F0 (240)	0x0001(1)	Serious error	The system version is too low to start the IoT card	Update system firmware version
	0x0002(2)	Serious error	Serious error in starting the IoT module	Check whether the driver and hardware work properly
	0x0003(3)	Warning	Abnormal signal strength	Check whether the driver and hardware work properly
	0x0004(4)	Warning	No port or port read-write error	Check whether the driver and hardware work properly
	0x0005(5)	Warning	Dial activation failed	Check whether the driver and hardware work properly
	0x0006(6)	Warning	No SIM card inserted.	Check whether the SIM card is installed correctly.
	0x0007(7)	Warning	SIM card has no data flow, Vpn error, etc.	Change another SIM card.

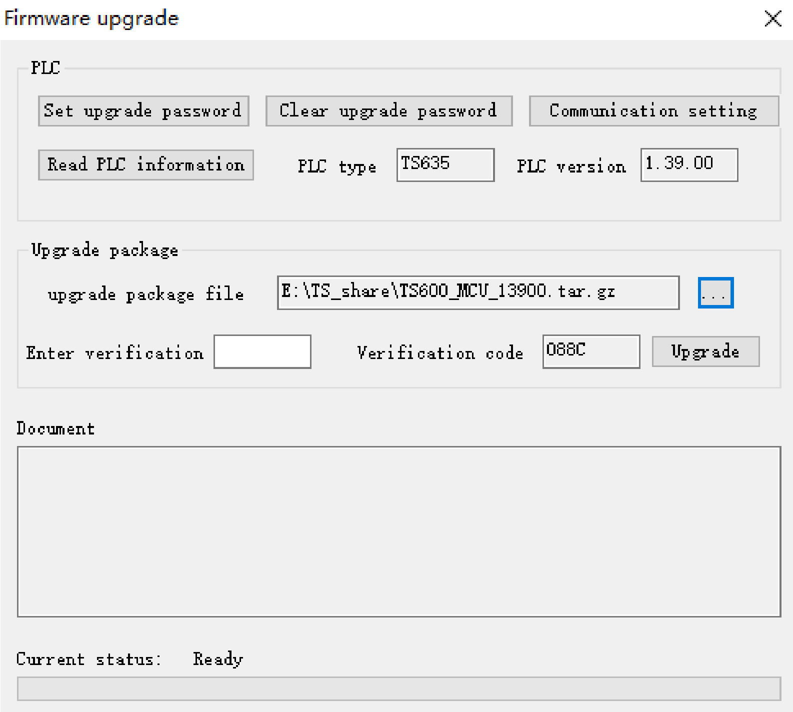
20 Firmware Burning and Upgrade

20.1 Upper Computer Firmware Upgrade

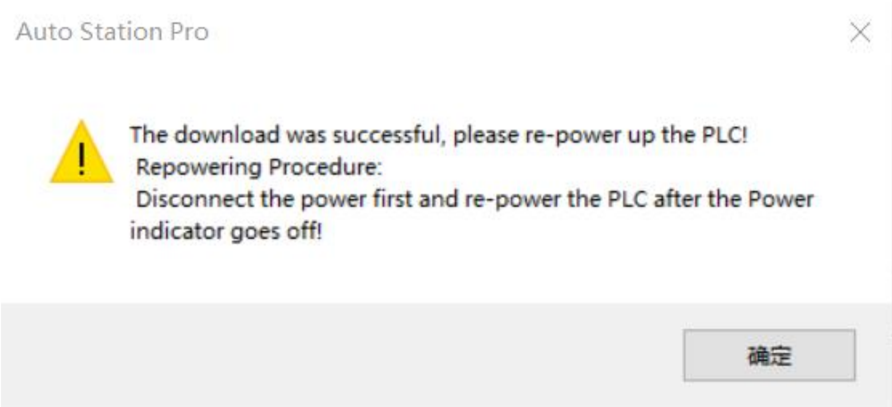
Step 1 When users use Ethernet or USB to connect PLC, they can enter the firmware upgrade interface in the toolbar "Tool" > "Firmware upgrade".



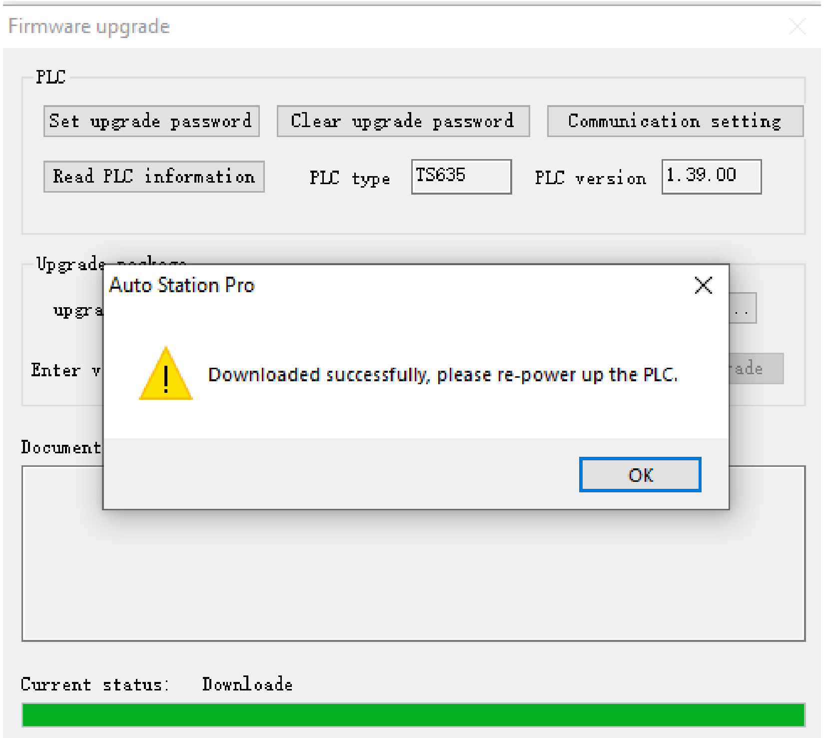
Step 2 Select the firmware version to be upgraded, enter the verification code, and click the Upgrade button.



Step 3 At this point, a prompt regarding update precautions will pop up.



Step 4 Click the "OK" button and wait for the pop-up "Download successfully" prompt box to complete the firmware upgrade.



Note:

- After the upgrade is successful, it is necessary to manually power off and restart the PLC. After manual power off and restart, you can check the version information through PLC information to verify whether the upgrade is successful.
- If the upgrade is successful and the PLC is restarted, the red indicator on the PLC flashes slowly, indicating that the project in the PLC is incompatible with the firmware. You need to set the PLC's switch to the STOP position, power off and restart the PLC, then recompile the project and download the PLC program, and finally set it to the RUN position.

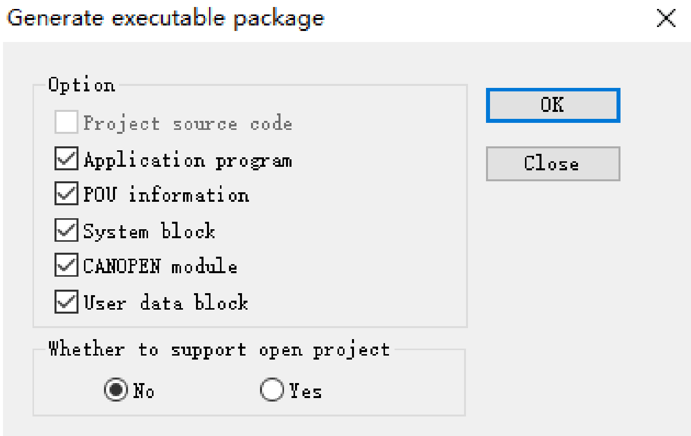
20.2 SD Card Firmware Upgrade

20.2.1 Step of Generating PLC Application Upgrade Package

Use a SD card to upgrade PLC application function, which means that PLC project can compile, generate the download project files, which is convenient for users to download without opening the original project. Use an SD card for batch updating or upgrading PLC projects, or use Auto Station Pro backend software to update PLC projects.

Before downloading the project file, you need to generate and download the project file through Auto Station Pro in the background. The specific operation steps are as follows:

Step 1 Open the PLC project and click "PLC>PLC executable package (A)>Generate" menu bar.



Step 2 Set the attributes of the downloaded file in the "Generate executable package" interface that pops up in the system, and then click "OK".

- Options: Check "Project source code", "Application program", "POU information", "System block" and "User data block".
  - ✧ "Project source code": Support to open the project, required
  - ✧ "Application program": Executable applications
  - ✧ "POU information": POU information
  - ✧ "System block": System related data configuration
  - ✧ "User data block": User related data configuration
- Whether to support opening the project, check: "No", "Yes".
  - ✧ No: The generation package file cannot open application project through AutoStation Pro, source upload is not supported, and the file format is *. cmf.
  - ✧ Yes: The generate package file can open the application project through AutoStation Pro, the source code upload is supported, and the file format *. upcmf.

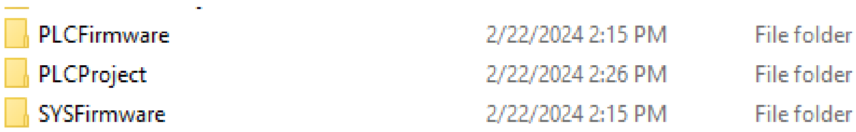
Step 3 Generate the PLC application name in the standard format of TS*_PROJECT_*. cmf or TS*_PROJECT_*. upcmf file for customer upgrade, where * is variable multiple characters, the former * is the product model, and the latter * is usually the project name.

20.2.2 SD Card Upgrading Steps

Step 1 Prepare a SD card with the storage capacity up to 32G, and it cannot be partitioned.

Step 2 Create the directory to be upgraded in SD card root directory.

- The PLC application upgrade directory is named PLCProject
- The PLC firmware upgrade directory is named PLCFirmware
- The system firmware upgrade directory is named SYSFirmware



Note: Only the directories that need to be upgraded are created, and the directories that do not need to be upgraded may not be created.

Step 3 Copy the PLC application package, or PLC firmware, or system firmware to the corresponding directory. The file is provided by the manufacturer and the name cannot be changed at will.

- The standard format of PLC application name is: TS*_PROJECT_*. cmf or TS*_PROJECT_*. upcmf, where * is variable multiple characters, the former * is the product model, and the latter * is usually the project name.

  Note: This package file is generated by the Auto Station Pro software. For details, see section 20.2.1 Step of Generating PLC Application Upgrade Package.

- The standard format of PLC firmware name is: TS600_MCU_*.tar.gz, where * is variable multiple characters, usually 5 digits to indicate the version.

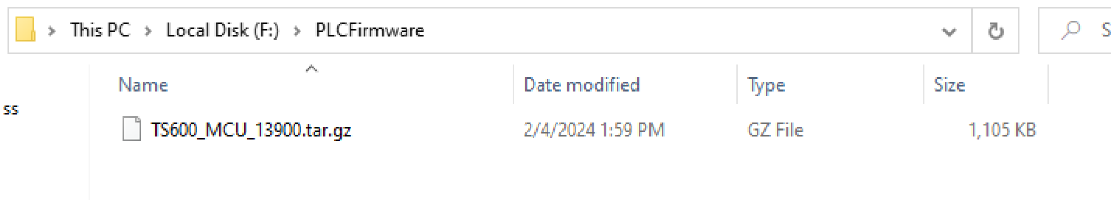
  Note: This firmware file is provided by the manufacturer.

- The standard format of system firmware name is: TS600_ARM_*.patch, where * is variable multiple characters, usually 5 digits to indicate the major version and P+3 digits for the patch version.

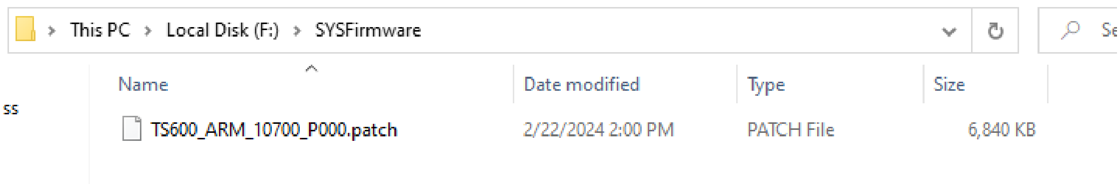
  Note: This firmware file is provided by the manufacturer.

Step 4 There is only one file in each directory that needs to be upgraded. More than one file may cause unexpected problems.

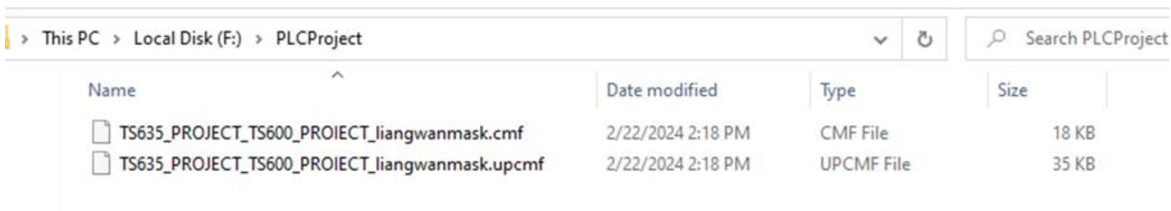
- **PLC firmware file**



- **System firmware file**



- **Application package file**



Step 5 Power off the PLC controller, and insert a SD card.

Step 6 Power up the PLC controller, and wait for the upgrade to complete.

If the upgrade is successful, the run indicator will flash for about 4 seconds, then the error indicator will flash slowly, waiting for the controller to turn off power.

Step 7 Power off the PLC controller, remove the SD card, and wait for the upgrade to complete.

You can check whether the SD card upgrade is successful through the log.

- **The successful upgrade of PLC firmware and system firmware is as shown in the figure below.**

```
33    2023-09-21 13:34:26    0    0    INFO:PLC firmware upgrade success
32    2023-09-21 13:34:24    0    0    INFO:system firmware upgrade success
31    2023-09-21 13:34:16    0    0    INFO:TS600 start run...
```

- **Upgrade failure log descriptions are as follows.**

```
65    2023-09-22 11:38:15    0    0    INFO:system firmware upgrade fail! because of: 0x101
66    2023-09-22 11:38:15    0    0    INFO:plc firmware upgrade fail! because of: 0x111
```

Note: SD card upgrade is only detected during power-on, and upgrade is no longer detected during operation. After successful upgrade, remove the SD card in time, otherwise the controller will not run normally.

Upgrade success and failure are displayed in the log file through information. If the upgrade fails, the controller runs normally and will not report errors. You need to view the reasons for the upgrade failure in the log file.

- System firmware upgrade failure error code

Upgrade Failure Error Code	Cause of Failure	Solution
0x101	The patch file could not be found	Check whether the file exists and whether the file has a standard name
0x102	Failed to get local version number	Check system files
0x103	Patch version is too low to upgrade	System version greater than or equal to version 1.05
0x104	Replication failed	Usually caused by insufficient storage space and insufficient memory
0x105	Check failure	Check the file for corruption

- PLC firmware upgrade failure error code

Upgrade Failure Error Code	Cause of Failure	Solution
0x111	PLC firmware file not found	Check whether the file exists and whether the file has a standard name
0x112	Failed to copy file	Usually caused by insufficient storage space and insufficient memory
0x113	Failed to create directory extract file	General file corruption
0x114	Script execution failed	Usually the file is corrupted, or the firmware does not meet manufacturer standard

- PLC application upgrade failure error code

Upgrade Failure Error Code	Cause of Failure	Solution
0x121	Upgrade package file not found	Check whether the file exists and whether the file has a standard name
0x122	Failed to copy file	Usually caused by insufficient storage space and insufficient memory
0x124	Failed to open the file	Usually the upgrade package file is corrupt or the file does not exist
0x125	Failed to allocate memory	Usually caused by insufficient memory
0x126	File header error	Usually the upgrade package file is corrupt or in a non-standard upgrade package format
0x127	Failed to write file	Usually caused by file unpacking failure, or

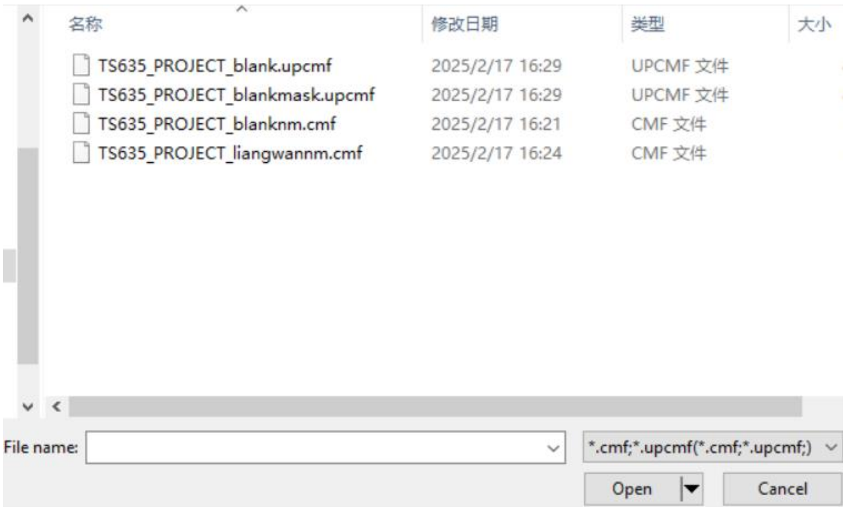
Upgrade Failure Error Code	Cause of Failure	Solution
		there is a system problem
0x128	CRC verification failed	Usually files corrupt, or in non-standard upgrade package format

20.3 Upgrade of Upper Computer Applications or Open the Project

20.3.1 Upgrade Applications with Upgrade Pack

Click on "PLC>PLC executable package (A)>Download" menu bar in the upper computer toolbar. The following dialog box pops up.

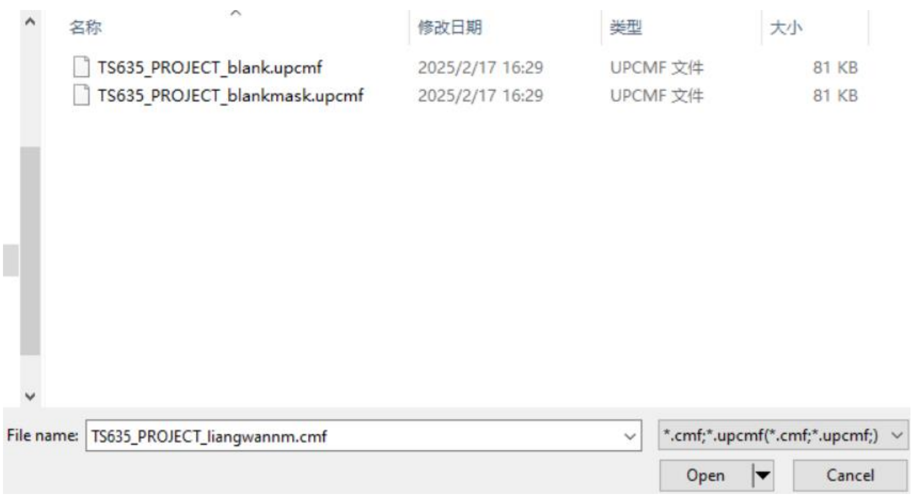
Select the file *. cmf or *. upcmf to upgrade. Where *. cmf is the source upgrade package that cannot be uploaded, and *. upcmf is the source file package that can be uploaded. After successful download, click Run to run the downloaded application.



20.3.2 Open a Project with Upgrade Pack

Click on the "File>Open project" menu bar in the upper computer toolbar. The following dialog box pops up and select the file *. upcmf to open.

Note: Only the upgrade pack in *. upcmf format supports opening the project file.



21 4G IoT Expansion Card

21.1 Overview

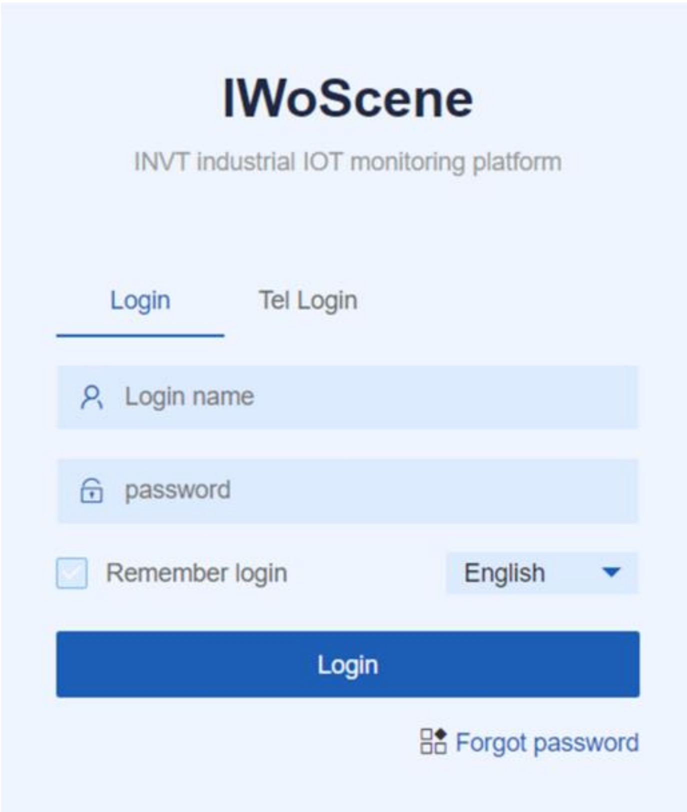
The TS-4G expansion card is a kind of functional expansion card applicable to INVT's TS600 Series PLC products. It supports 4G Internet of Things, which can help customers realize remote data collection, remote upload/download and remote debugging.

The status of this 4G expansion card can be obtained from 4G module system variables or logs.

21.2 User Login of IoT Monitoring Platform

The PLC can be monitored and operated remotely after adding devices by using the IoT monitoring platform.

Enter iot.invt.com in the address bar of Google Browser and press Enter to visit the login page of the industrial IoT application platform. As shown in the following figure, enter the account number and password to complete the login.



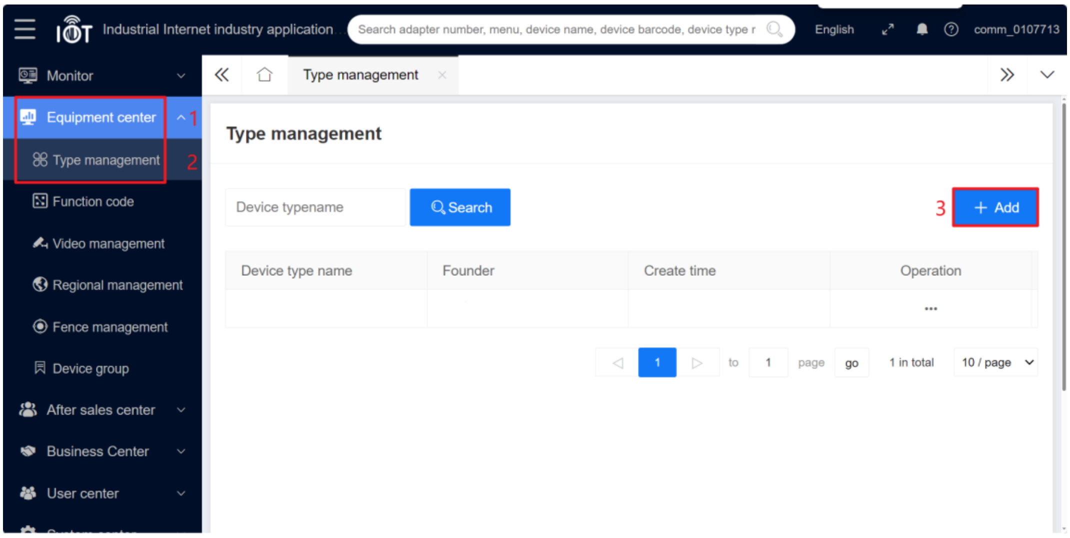
Note:

- To obtain the account number and password, please contact the industry administrator.
- The account number and password are the authentication credentials of INVT Industrial IoT Platform. After logging in, the device management function is available. Users should keep them properly and take adequate preventive measures to prevent others from stealing them. If the user name and password are stolen, great losses may be caused.
- Before using the device for remote operation, users should communicate with the site to ensure safety in advance, otherwise heavy losses may be caused.

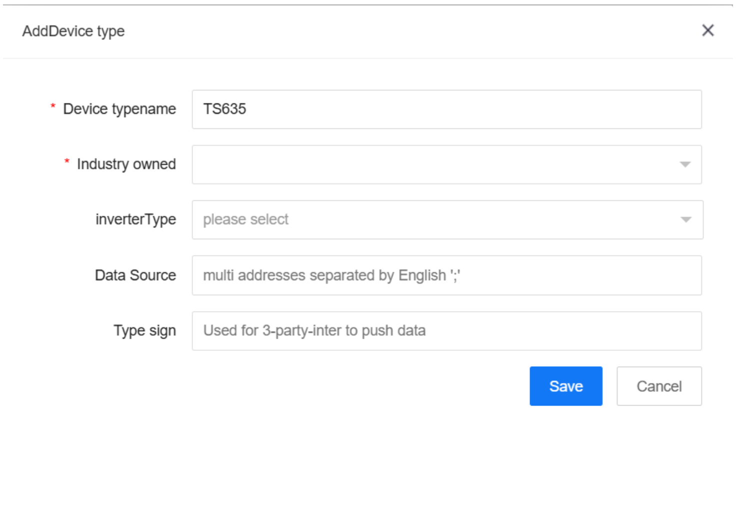
- The IoT sim card is forced to be bound to a device, which means a sim card can only be used on the device that is powered on and connected to the Internet with this card for the first time. Users should not insert the IoT sim card into other devices, otherwise the sim card will be locked.
- This product is an industrial IoT product. INVT has taken necessary technical measures to ensure data security, but there may still be network security risks beyond our control or responsibility, such as hacking. If it is not for the harm caused by the quality defects of this product itself, our company shall not be responsible for any losses.

21.3 Add a Device Type

After logging in successfully, the home page will be displayed. Click on "Equipment center" > "Type management" > "Add", and a dialog box for adding device type will pop up.

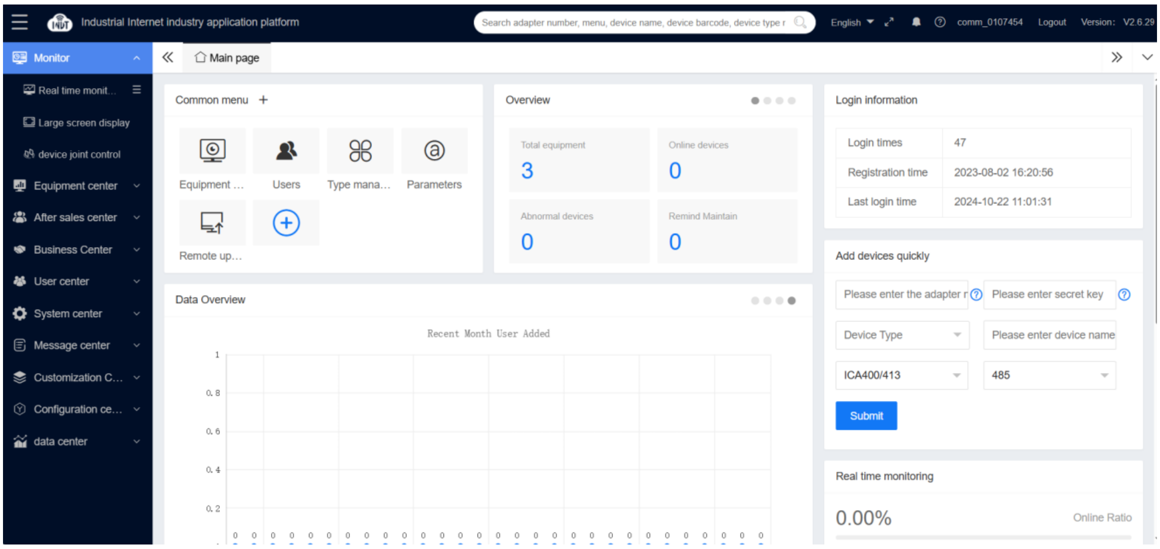


Fill in the corresponding device type name and select the corresponding industry and click "Save".



21.4 Add Devices to IoT Platform

Return to the homepage of the IoT business platform, enter the adapter code (referring to the module ID, which is the barcode of the PLC), key and device alias in the "Add devices quickly" section on the homepage, select the device type according to the monitoring type, and then click "Submit" after confirming the input is correct.



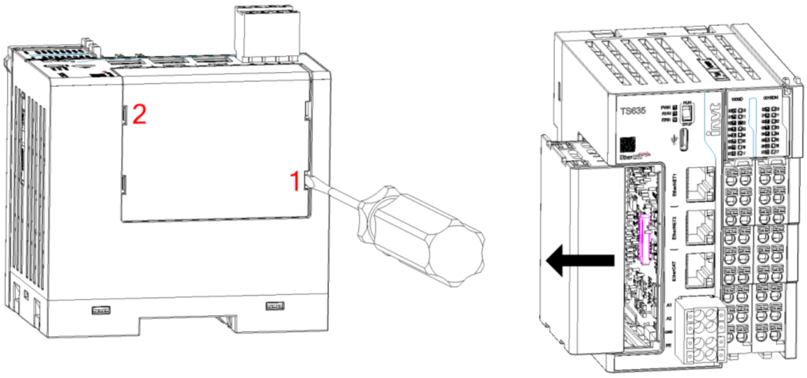
21.5 Device Installation & Wiring

Step 1 Slide out the SIM card holder of 4G module and insert the SIM card into the card holder.

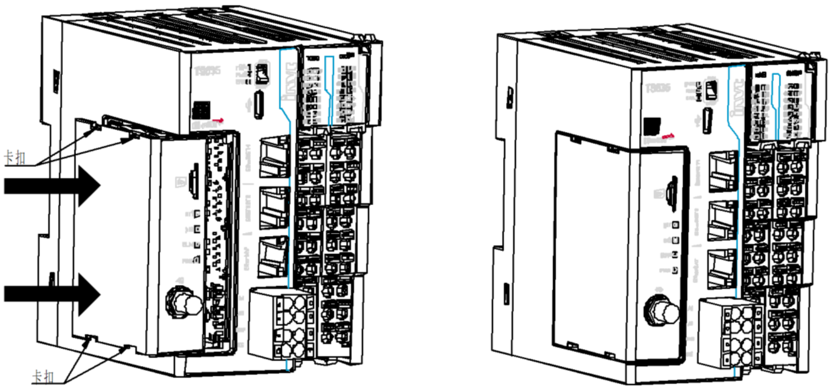
Step 2 Install the 4G antenna correctly and place the 4G antenna in a position with strong signal.

Note: It is forbidden to place the 4G antenna in the box with signal shielding effect.

Step 3 Gently pry open the cover plate snaps with a tool on the side of the product (in the order of positions 1 and 2); Slide the cover plate to the left horizontally and take it out.



Step 4 Slide the expansion card into the guide slot horizontally, and then press the snap positions on the upper and lower sides of the expansion card hard until the expansion card is snapped tightly (there is obvious snap sound after it is installed in place).



Step 5 Power on and start PLC.

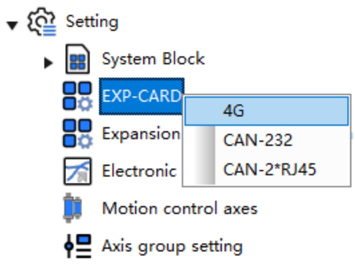
Step 6 When the NET indicator flashes at a frequency of 75ms, the network is ready and data transmission begins.

21.6 Establish VPN Transparent Transmission Channel

Note: At present, VPN transparent transmission is only used in China.

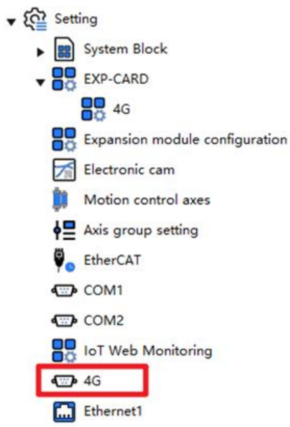
VPN transparent transmission is a method of downloading and monitoring PLC remotely through Auto Station Pro upper computer. The following tutorial describes how to use this function in Auto Station Pro.

Step 1 Right-click on "EXP-CARD", select "4G", and the 4G button will pop up.

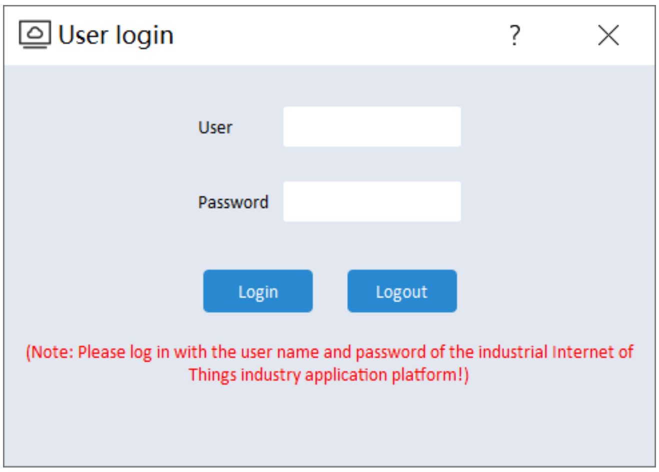


Step 2 Download the configuration to PLC and toggle the toggle switch to set it to RUN state.

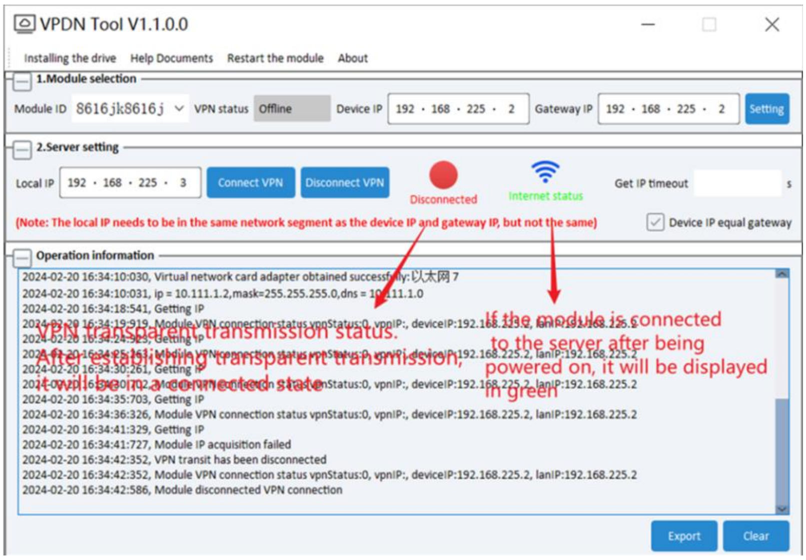
Step 3 Double-click "4G" to open the VPN Tool. Before establishing the VPN transparent transmission channel, please ensure that the network adapter on the PC side does not have a manually set IP, otherwise please set it to automatically obtain IP. If the relevant network adapter is disabled, enable it.



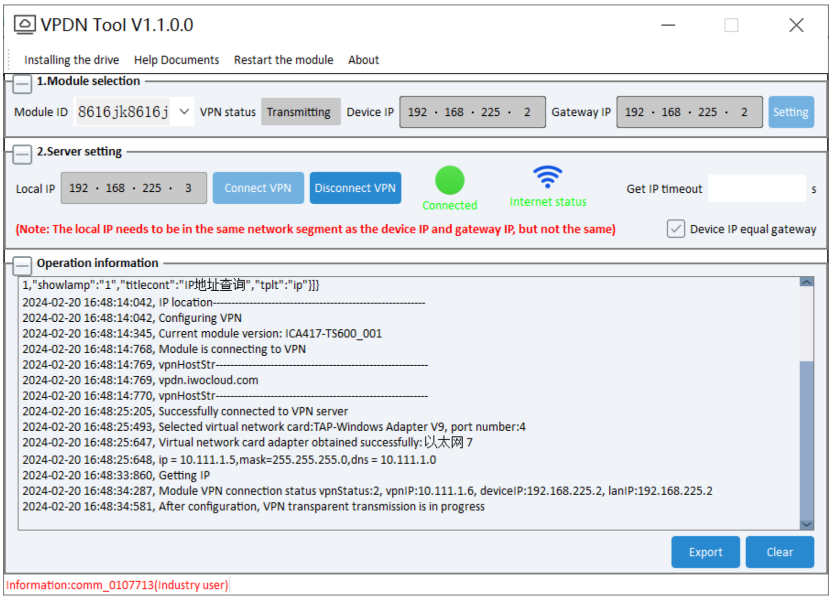
Step 4 Enter the account number and password of the IoT monitoring platform to log in (the account & password mentioned in the first step to log in to the web platform) and enter the main interface of the tool.



Step 5 Log in to enter the main interface, select the corresponding ID of the module, and check the device IP to be equal to the gateway IP. The device IP, gateway IP and local IP of the IoT module are automatically obtained, and users do not need to fill them.



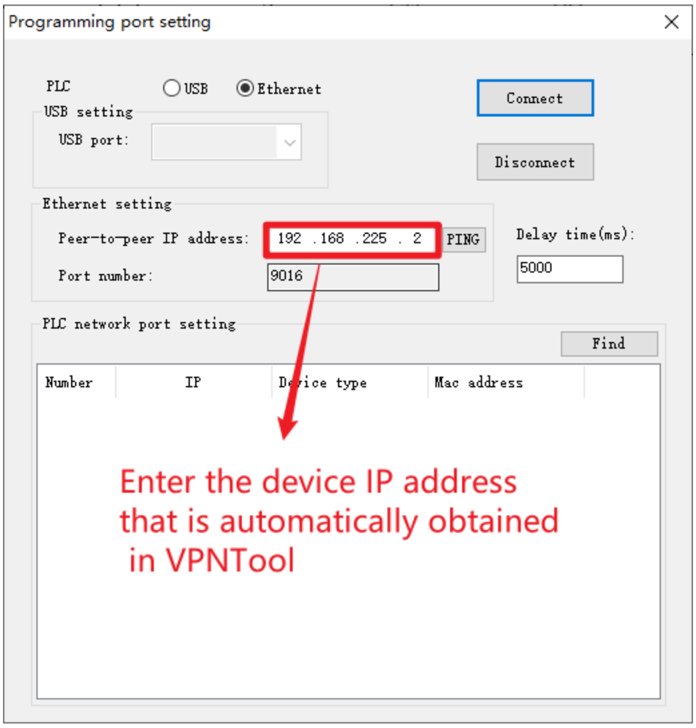
Step 6 Click the button to establish VPN transparent transmission and the connection progress will be displayed. After the progress bar reaches 100%, the transparent transmission is successfully established.



21.6.1 Connect PLC Remotely through VPN

After completing the above steps, PLC can be operated remotely.

Open the Auto Station Pro upper computer software, fill in the IP address of the device automatically obtained in VPN Tool in the remote IP address field, and then click "Connect" to connect to PLC remotely.



21.6.2 FAQ

You can view module system variables or logs to obtain current module information. The following are answers to some common questions.

1. After power-on, the POWER indicator does not flash or illuminate.

Answer: Check whether the device is installed correctly.

2. When using 4G to surf the Internet, the network status indicator (NET) always flashes slowly, and a window indicating MQTT connection abnormality pops up when establishing VPN transparent transmission.

Answer: (1) The sim card is not installed in place. Please reinstall it after power off to ensure good contact.

(2) Move the 4G antenna to a place where the signal strength is good.

(3) Inquire whether the sim card is activated and whether there is a balance.

3. Interruption happens when VPN transparent transmission tool establishes VPN transparent transmission.

Answer: (1) Check whether the module is online.

(2) Check whether the computer has installed the required network adapter driver (named TAP-Windows Adapter V9). Please click the installation driver at the top left of VPN transparent transmission tool to install it.

4. It is impossible to ping the device after VPN transparent transmission is established normally.

Answer: (1) The network cable connection between the module and the transparent transmission device is abnormal, so ensure the normal connection.

(2) The local network adapter has a manually set IP address, and set the IPV4 of the network adapter to be acquired automatically.

(3) The local IP of the transparent transmission device and the device IP of the module are not in the same network segment, modify the device IP of the module in the VPN transparent transmission tool or make modifications in the upper computer software of the transparent transmission device to put them in the same network segment.

(4) The gateway IP of the transparent transmission device is inconsistent with that of the module, so

modify the gateway IP of the module in the VPN transparent transmission tool or make modifications in the upper computer software of the transparent transmission device to make sure they use the same gateway IP.

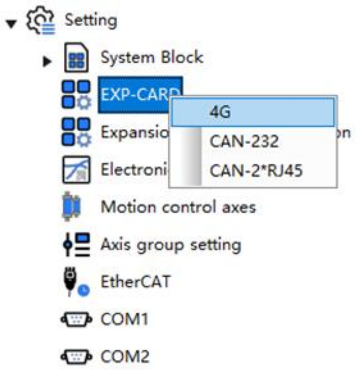
21.7 Web Page Monitoring

Note: VPN transparent transmission and web page monitoring cannot be used at the same time.

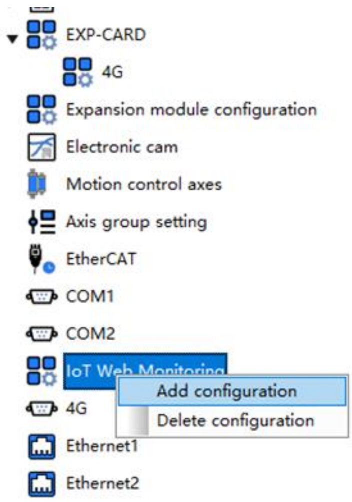
The web page monitoring function uploads PLC element values to our IoT cloud server, and users can check the historical data in the web page to know the historical running status of PLC. At present, only monitoring Modbus address is supported. The following tutorial will help you monitor PLC elements on web pages.

First, you need to configure the elements to be monitored in the Auto Station Pro upper computer.

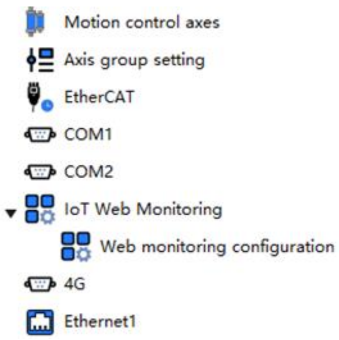
Step 1 Right-click on EXP-CARD, select "4G", and the "IoT Web Monitoring" button will pop up.



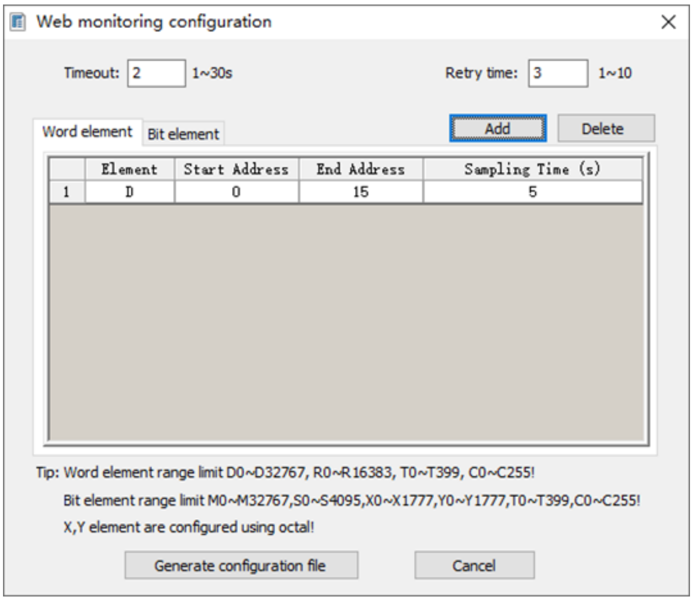
Step 2 Right-click on "IoT Web Monitoring" > "Add configuration" to add the IoT web page monitoring configuration file.



Step 3 Double-click the "Web monitoring configuration" button to pop up the web monitor configuration bar.



Step 4 Click the add button to add a configuration. In this case we select Monitoring D0–D15, and the sampling time is 30 seconds.



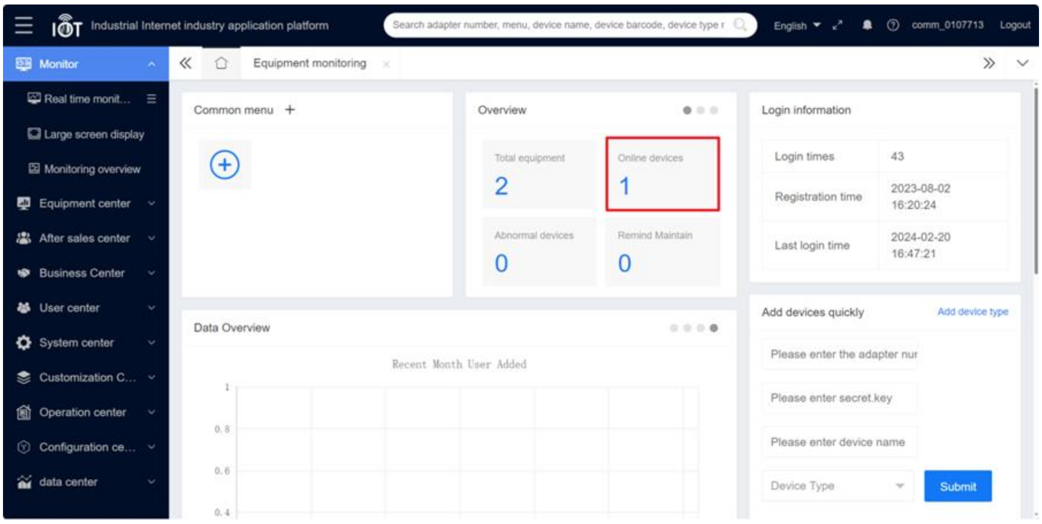
The configurable contents are as follows:

Options	Setting Content	Note
Sampling time	1–1800s	The time when the element data is uploaded to the cloud server periodically
Timeout time	1–30s	Wait timeout when the communication with cloud server fails
Retry times	1–10 times	The number of retries required when the cloud server fails to collect.

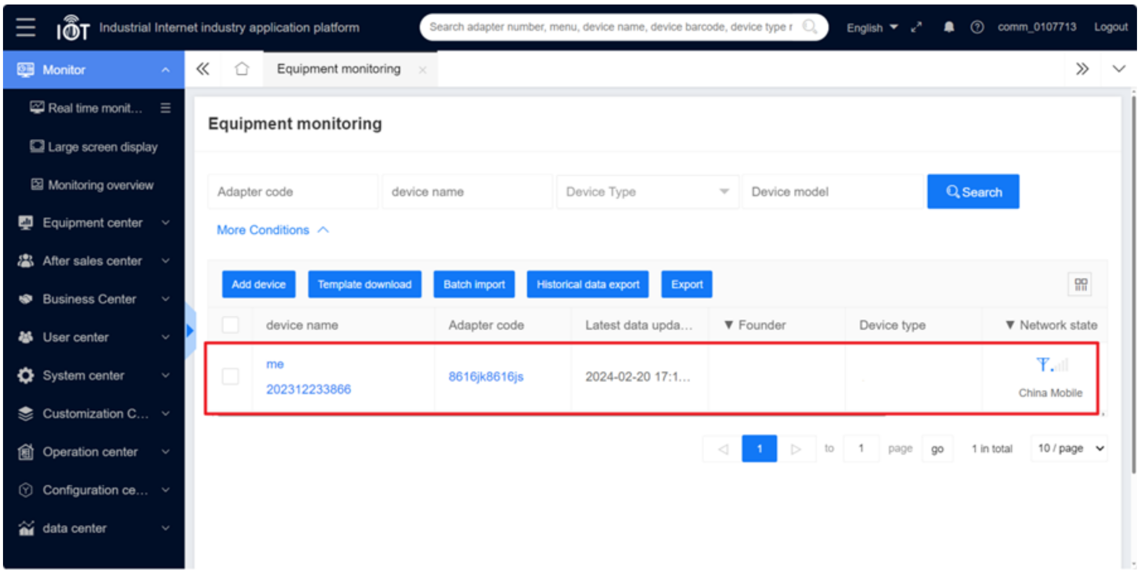
Step 5 After the configuration is completed, click "Generate configuration file", then download this project to PLC, and set PLC to RUN state.

At this point, you have configured the elements to be monitored, and you can view these elements in the industrial IoT monitoring platform.

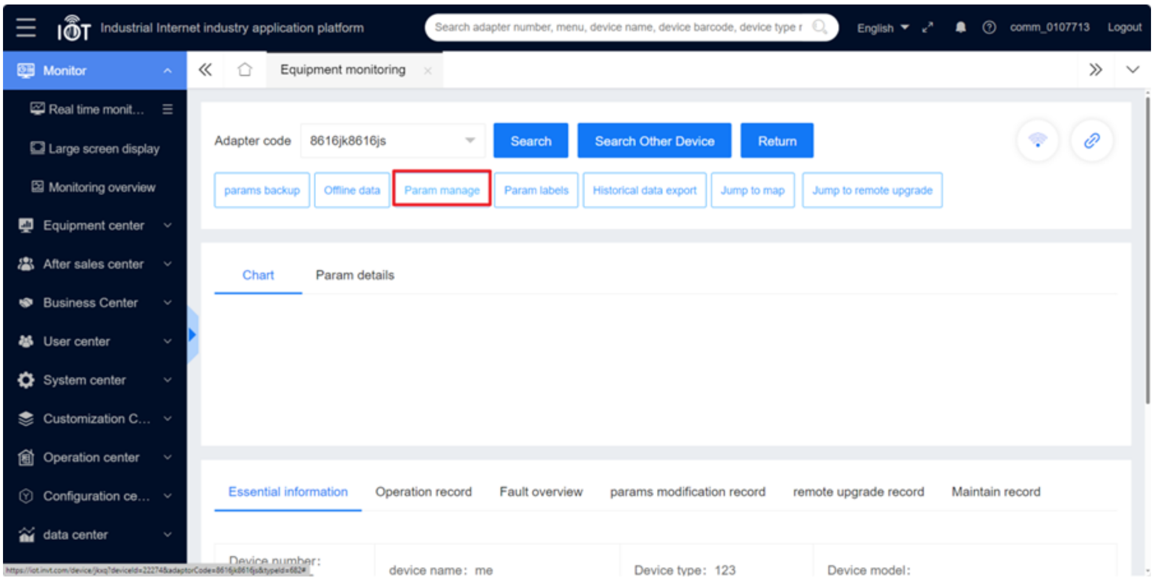
Step 1 On the homepage of the IoT monitoring platform, click online devices to enter the device monitoring page.



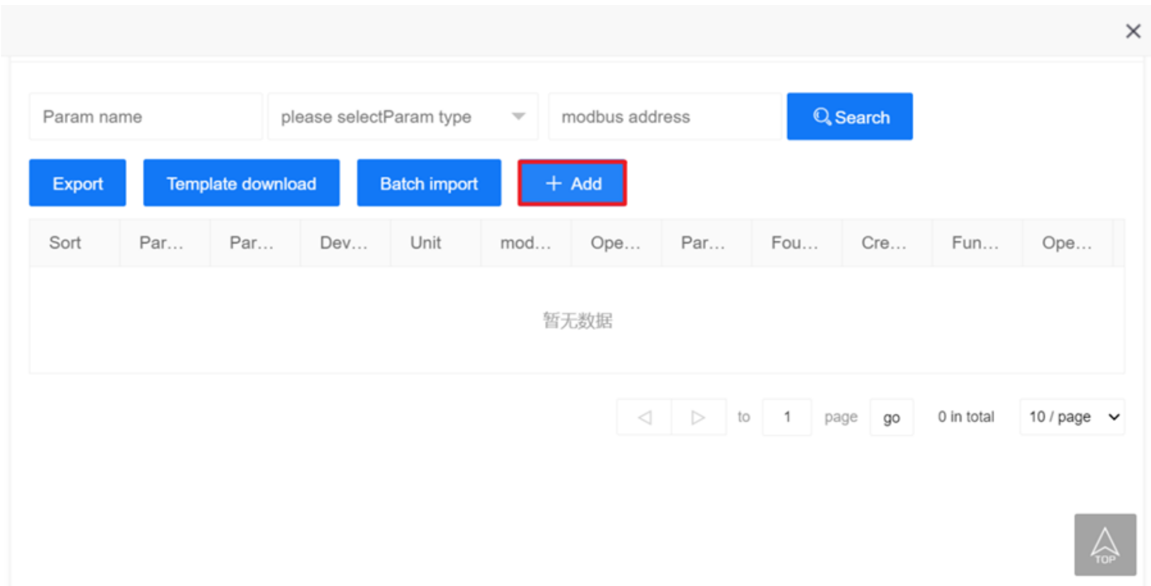
Step 2 Select the device currently in use on the device monitoring page and click its adapter code to move to the next tier of device monitoring page.



Step 3 Click "Param manage" on this page.

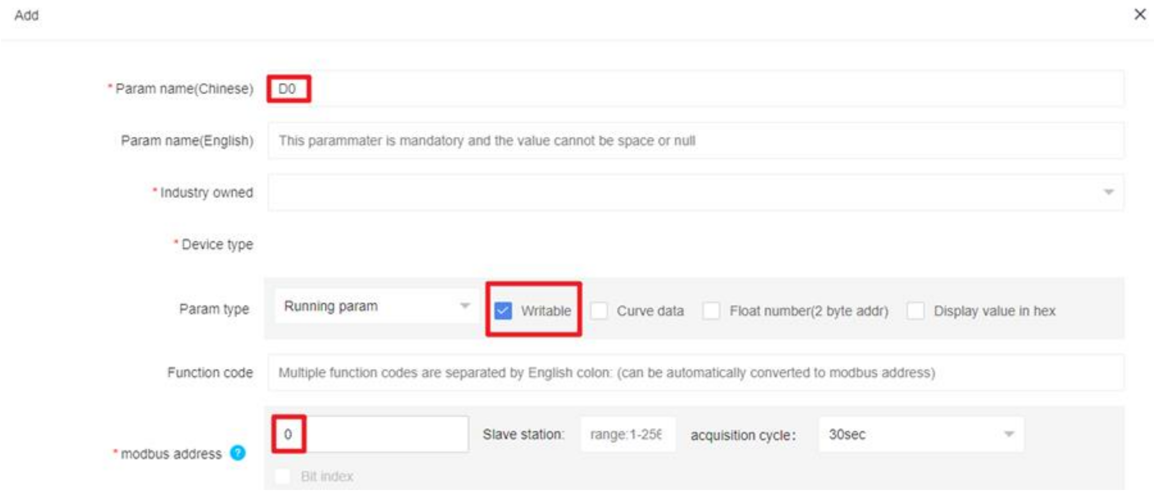


Step 4 Click "Add" to configure a monitor.



Step 5 Add parameter name, industry and device type, and check writable (element value can be written).

- When monitoring D0 element, the Modbus address is filled in "0".
- When monitoring R0 element, the Modbus address is filled in "32768".

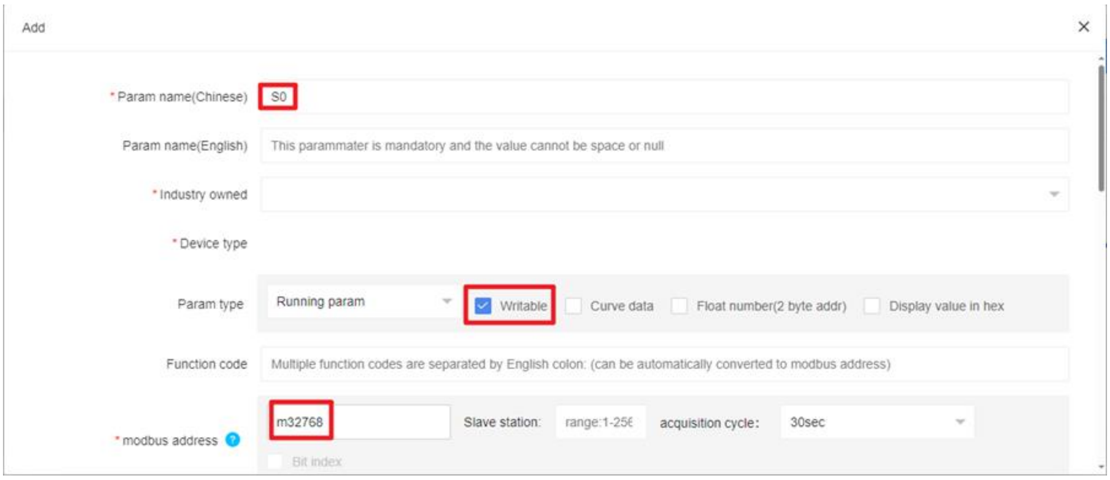


The following table shows the Modbus addresses, and you can query the elements to be monitored according to the following table.

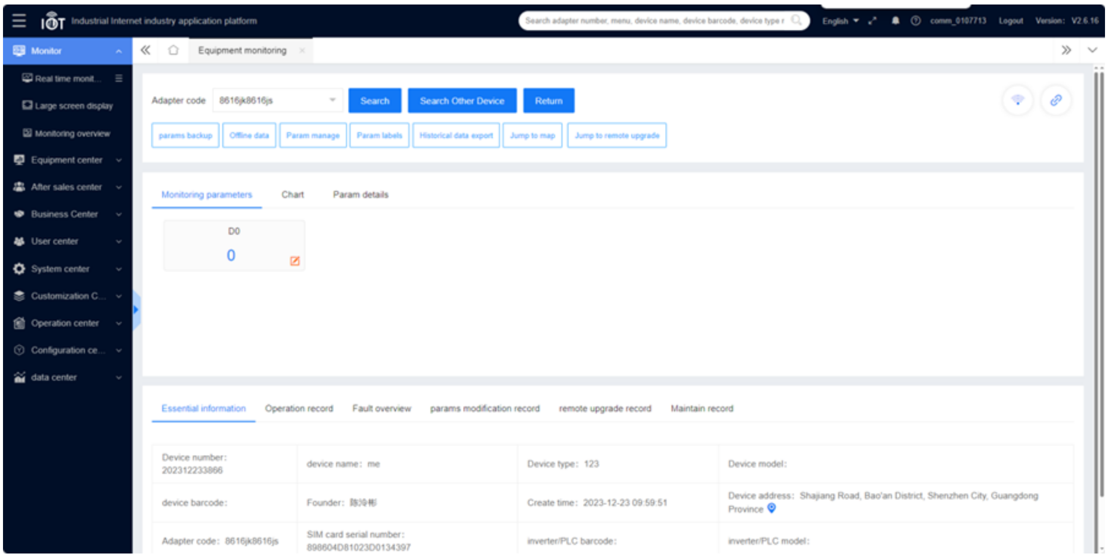
Element	Type	Total quantity	Modbus address
M0–M32767	Bit element	32768	m0–m32767
S0–S4095	Bit element	4096	m32768–m36863
X0–X1777	Bit element	1024	m40960–m41983
Y0–Y1777	Bit element	1024	m45056–m46079
T0–T399	Bit element	400	m49152–m49551
C0–C255	Bit element	256	m49664–m49919
D0–D32767	Word element	32768	0–32767
R0–R16383	Word element	32768	32768–49151
T0–T399	Word element	400	57344–57743
C0–C255	Word element	256	57856–58111
Z0–Z15	Word element	16	58368–58383

- When monitoring bit elements, the rule for filling in the Modbus address is: "m"+"Modbus address".

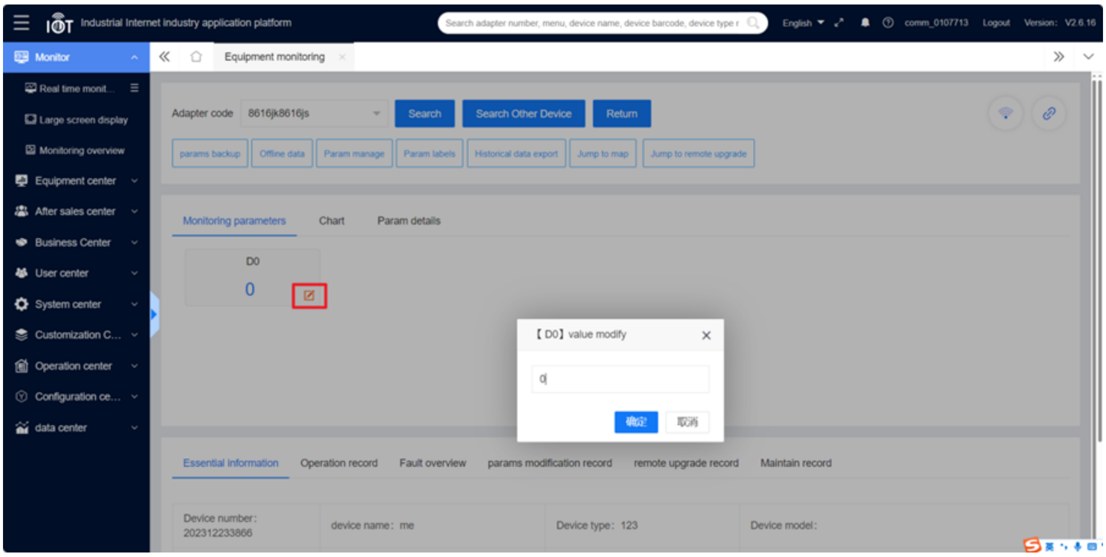
  Example: When monitoring M0 bit elements, the Modbus address is filled in "m0".
- When monitoring coil elements, such as S0, the Modbus address is filled with m32768. (m represents the read-write coil followed by the Modbus address)



Step 6 At this time, the value of D0 can be monitored in the monitoring parameters.

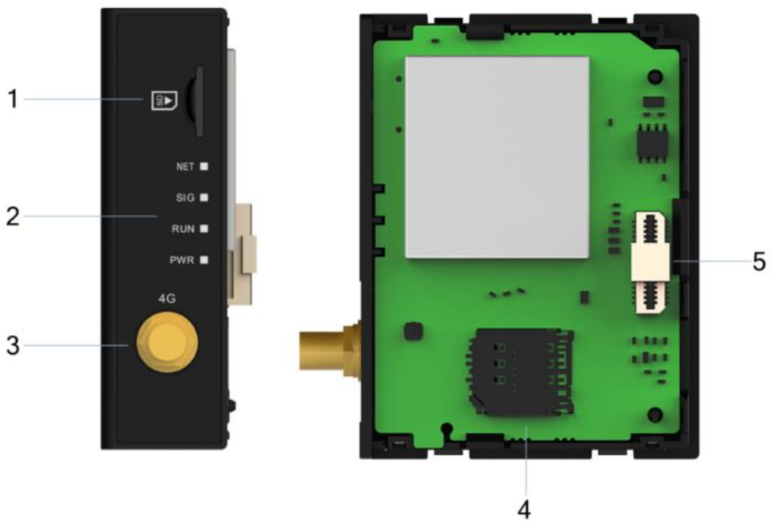


Step 7 Click the Write button in the lower right corner to write the element value.



21.8 Module Interface Description

See the following table for the meaning of 4G module interface.



No.	Port type	Interface sign	Definition	Description
1	SD card socket	SD	Micro SD	Standard definition
2	Indicator	NET	Network indicator	Slow flash: No sim card/registering in the network/network register failure Flash: Data link established
		SIG	Signal indicator	Steady On: Good signal Slow flash: The signal is average Off: Poor signal
		RUN	Operation status indicator	Steady On: MQTT login Flash at an interval of 3s: MQTT login failed Off: Abnormal operation Flash at an interval of 1s: VPN on
		PWR	Power indicator lamp	Power on: ON Power off: Off
3	4G antenna interface	4G	4G antenna interface	-
4	SIM card interface	-	Micro SIM card	-
5	Board-to-board interface	-	Communicate with the host	Only applicable to TS600 Series PLC host

22 Appendix

22.1 Modbus Protocol

22.1.1 Modbus RTU

When the device uses RTU (Remote Terminal Unit) mode to communicate on Modbus serial link, each 8-bit byte in the message contains two 4-bit hexadecimal characters. The main advantages of this mode are higher data density and higher throughput than ASCII mode at the same baud rate. Each message must be sent in a continuous character stream.

The format of each byte (11 bits) of RTU mode is:

- Coding system: 8-bit binary

  Each 8-bit byte in the message contains two 4-bit hexadecimal characters (0 – 9, A – F)

- Bits per Byte: 1 start bit

  8 data bits, the least significant bit is sent first

  1 bit as parity bit

  1 stop bit, and 2 stop bits are required when there is no check bit

The data frame format is as follows:

Start	Slave Address	Function code	Data	CRC Check	End
⩾ 3.5 characters	1 character	1 character	0 – 252 characters	2 character	⩾ 3.5 characters

22.1.2 Modbus ASCII

When devices on the Modbus serial link is configured to communicate using an ASCII (American Standard Code for Information Interchange) mode, each 8-bit byte in the message is sent with two ASCII characters. This mode is used when the communication link or device cannot comply with the timing management of RTU mode. **Note:** Because one byte requires two characters, this mode is less efficient than RTU.

Example: Byte 0X5B is encoded as two characters: 0x35 and 0x42 (ASCII code 0x35 = "5", 0x42 = "B").

The format of each byte (10 bits) in ASCII mode is:

- Coding system: Hexadecimal, ASCII characters 0 – 9, A – F

  Each ASCII character in the message contains 1 hexadecimal character

- Bits per Byte: 1 start bit

  7 data bits, the least significant bit is sent first

  1 bit as parity bit

  1 stop bit, and 2 stop bits are required when there is no check bit

The data frame format is as follows:

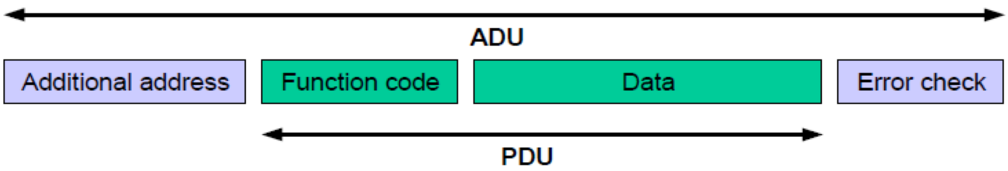
Beginning Character	Slave Address	Function code	Data	LRC Check	End
1 character":"	2 character	2 character	0–2x252 characters	2 character	2 characters: CR, LF

22.1.3 Modbus TCP

Modbus messaging service provides client/server communication between devices connected in the same Ethernet TCP/IP network.

Modbus protocol defines a simple Protocol Data Unit (PDU) independent of the underlying communication layer. Modbus protocol mapping on a particular fieldbus or network can introduce some additional domains on application data units (ADU).

Figure 22-1 General Modbus Protocol Frame



The data frame format is as follows:

MBAP Header	Function code	Data
7 bytes	2 bytes	0 – 252 bytes

Data frame format description:

Domain	Length	Master	Slave	Description
1 character":"	2 character	0–2x252 characters	2 character	2 character
Transaction meta identifier	2 bytes	Master startup	Slave replication	Modbus requests/responds to the identification code of the transaction, and the master identifies the identifier after sending the request to confirm that the identifiers of the master and slave are consistent
Protocol identifier	2 bytes	Master startup	Slave replication	Modbus protocol: 0
Length	2 bytes	Master startup	Slave startup	Number of subsequent messages (bytes)
Unit identifier	1 bytes	Master startup	Slave replication	Slave identification code/slave address
Transaction meta identifier	2 bytes	Master startup	Slave replication	Modbus requests/responds to the identification code of the transaction, and the master identifies the identifier after sending the request to confirm that the identifiers of the master and slave are consistent

22.1.4 Modbus Function Code

Function code	Name	Note
01(0x01)	Read coils	Read bit
02(0x02)	Read discrete input	Read bit
03(0x03)	Read save register	Read word
04(0x04)	Read input register	Read word

Function code	Name	Note
05(0x05)	Write single coil	Write bit
06(0x06)	Write single register	Write word
15(0x15)	Write multiple coils	Write bit
16(0x16)	Write multiple registers	Write word

22.1.5 Example of Modbus Message Parsing

The request and response messages for reading a register with the above three Modbus protocols are given below.

22.1.5.1 Modbus-RTU

- Master request message

Slave Address	Function code	Start address (high)	Start address (low)	Quantity (high)	Quantity (low)	CRC Check
0x01	0x03	0x00	0x01	0x00	0x01	0xD5 0xCA

- Slave response message

Slave Address	Function code	Number of Bytes	Register Value (High)	Register Value (Low)	CRC Check
0x01	0x03	0x02	0x00	0x00	0xB8 0x44

22.1.5.2 Modbus-ASCII

- Master request message

Start	Slave Address	Slave Address	Function code	Function code	Start address (high)	Start address (high)	Start Address (Low)	Start Address (Low)
0x3A	0x30	0x31	0x30	0x33	0x30	0x30	0x30	0x31
Quantity (high)	**Quantity (high)**	**Quantity (low)**	**Quantity (low)**	**LRC Check**	**LRC Check**	**End Character**	**End Character**	
0x30	0x30	0x30	0x31	0x46	0x41	0x0D	0x0A	

- Slave response message

Start	Slave Address	Slave Address	Function code	Function code	Number of Bytes	Number of Bytes	Register Value (High)	Register Value (High)
0x3A	0x30	0x31	0x30	0x33	0x30	0x32	0x30	0x30
Register Value (Low)	**Register Value (Low)**	**LRC Check**	**LRC Check**	**End Character**	**End Character**			
0x30	0x30	0x46	0x41	0x0D	0x0A			

22.1.5.3 Modbus-TCP

- Master request message

Transaction Identifier (High)	Transaction Identifier (Low)	Protocol identifier	Protocol identifier	Length (Height)	Length (Low)
0x00	0x00	0x00	0x00	0x00	0x06
Unit identifier	**Function code**	**Start address (high)**	**Start address (low)**	**Quantity (high)**	**Quantity (low)**
0xFF	0x03	0x00	0x01	0x00	0x01

- Slave response message

Transaction Identifier (High)	Transaction Identifier (Low)	Protocol identifier	Protocol identifier	Length (Height)	Length (Low)
0x00	0x00	0x00	0x00	0x00	0x05
Slave Address	**Function code**	**Number of Bytes**	**Register Value (High)**	**Register Value (Low)**	
0xFF	0x03	0x02	0x00	0x00	

22.1.6 Modbus Abnormal Response

The Modbus abnormal response code is as follows:

Function code	Name	Note
01	Illegal function code	The slave response function code is incorrect
02	Invalid data address	Illegal slave response data address
03	Invalid data	Illegal slave response data
04	Slave device failure	Slave device error

Your Trusted Industry Automation Solution Provider

Shenzhen INVT Electric Co., Ltd.
Address: INVT Guangming Technology Building, Songbai Road, Matian,
Guangming District, Shenzhen, China
INVT Power Electronics (Suzhou) Co., Ltd.
Address: No. 1 Kunlun Mountain Road, Science & Technology Town,
Gaoxin District, Suzhou, Jiangsu, China
Website: www.invt.com

INVT mobile website

INVT e-manual

66001-01291

202502 (V1.2)