INOVANCE

FORWARD, ALWAYS PROGRESSING

H5U and Easy Series Programmable Logic Controllers

Programming and Application Guide

Industrial Automation

Intelligent Elevator

New Energy Vehicle

Industrial Robot

Rail Transit

Data code 19012249 A11

Preface

Introduction

The H5U series Programmable Logic Controller (PLC), a new generation of small-sized PLC developed by Inovance, supports EtherCAT bus communication and features powerful motion control and distributed I/O control functions. It allows process encapsulation and reuse using the FB/FC function, as well as multi-layer network communication through the RS485, CAN, Ethernet, and EtherCAT interfaces.

Easy series small- and medium-sized PLCs are available in eight models, covering the demands of automation equipment requiring small footprint, multi-axis motion control, accurate temperature control, and easy networking.

This guide describes the basic knowledge, quick start guide, communication, motion control, and high-speed counter of H5U and Easy series PLCs.

Related Manuals

Data Code	Doc Name	Description
19011419	H5U Series Programmable Logic Controller User Guide	Describes installation and wiring of H5U series programmable logic controller.
PS00006444	Easy Series Programmable Logic Controller User Guide	Describes installation and wiring of Easy series programmable logic controller.
19012250	H5U&Easy Series Programmable Logic Controller Instruction Guide	Describes basic instructions and examples of H5U&Easy series programmable logic controller.

Revision

Date	Version	Revision
May 2023	A11	Added the BYTE variable and section 3.3.7 "Defining Specific Unions." Added section 3.11.5 "Encrypting Function Blocks (FB) or Functions (FC)". Added section 3.12 "Folder" for program blocks, function blocks (FBs), and functions (FCs). Added local extension modules GL20-2SCOM and GL20-2S485 (see section 5.2.2.3.7 "Communication Modules"). Added section 5.4 "GR10-EC-6SW Branch Module" and section 5.5 "GS20-ECT-8L Module." Added the H5U series PLC PROFINET communication function (see section 11.1 "Overview"). Corrected details of the earlier version.
January 2023	A10	Added local modules. Corrected details of the earlier version.

Date	Version	Revision
November 2022	A09	Added the LiteST programming language. Added section 10.3.4 "EtherNet/IP Slave Application Example" and section 10.3.1.4 "Exporting EDS Files". Added section 6.3.2 "Free Protocol Cancellation (SerialSR Instruction)." Added section 6.7 "Modifying Serial Port Parameters." Added models of extension modules supported by Easy series host and running cases of all modules. Added extension modules and use methods of GL20-RTU-ECT coupler. Corrected details of the earlier version.
September 2022	A08	Added the EIP function. Added use methods of IP variables. Added Easy series functions. Corrected details of the earlier version.
August 2021	A07	Added the method of setting the FB default. Added graphic block functions of quickly increasing or decreasing label numbers and implementing incremental paste. Added the function of packing and decompressing project archives. Added the function of uploading and downloading Updown files. Added the function of setting axis settings parameters by instructions. Added the function of dragging the motion control axis. Added the function of disabling the EtherCAT slave station by instructions. Corrected details of the earlier version.
May 2021	A06	Kept the material versions consistent.
March 2021	A03	Added chapter 16 "Electronic Cam." Added chapter 15 "Bus Encoder Axes." Added chapter 17 "Offline Commissioning." Added chapter 18 "Memory Management." Corrected details of the earlier version.
August 2020	A02	Optimized the function of binding structure variables to soft elements. Added the function of setting IP addresses using system variables. Added the function of binding array variables to soft elements. Corrected details of the earlier version.

Date	Version	Revision
May 2020	A01	Added hard limit processing and the function of automatically starting the EtherCAT slave station, and updated the software UI. Added the interpolation function and axis group configuration. Added the Down file download and login functions. Updated motion control axis fault codes.
February 2020	A00	First release

Note

This guide is applicable to AutoShop V4.4.0.0 and PCB software V5.0.0.0 and later version.

Document Acquisition

This guide is not delivered with the product, but an electronic PDF version is available. To obtain it, visit

- *www.inovance.com*, click Downloads, search for the keyword, and download the guide.
- Scan the QR code on the controller using your mobile phone to obtain a complete set of product manuals.

Warranty

For faults or damage that occur during normal use within the warranty period, Inovance provides free repair service. (For details about the warranty period of the equipment, see the order list.) When warranty expires, maintenance fee will be charged by Inovance.

Within the warranty period, maintenance fee will be charged by Inovance for the following damage:

- Damage caused by improper operation
- Damage caused by fire, flood, or abnormal voltage
- Damage caused by use beyond the intended scope of application
- Damage caused by use under unintended working conditions
- Secondary damage caused by force majeure events (such as natural disaster, earthquake, and lightning stroke)

The maintenance fee is charged according to the latest Price List of Inovance if not otherwise agreed upon.

For details, see the Product Warranty Card.

Table of Contents

1 Overview

1.1 Introduction

1.1.1 Product Introduction

The H5U series PLC, a new generation of small-sized PLC developed by Inovance, supports EtherCAT bus communication and features powerful motion control and distributed I/O control functions. It allows process encapsulation and reuse using the FB/FC function, and multi-layer network communication through the RS485, CAN, Ethernet, and EtherCAT interfaces.

Easy series small- and medium-sized PLCs are available in eight models, covering the demands of automation equipment requiring small footprint, multi-axis motion control, accurate temperature control, and easy networking.

1.1.2 Software Introduction

AutoShop is programming configuration software provided by Inovance for small-sized PLCs. It provides a friendly programming and commissioning environment and supports various and powerful communication and control functions. AutoShop supports various programming languages, such as the ladder diagram (LD), sequential function chart (SFC), and structured text (LiteST).

AutoShop has the following features:

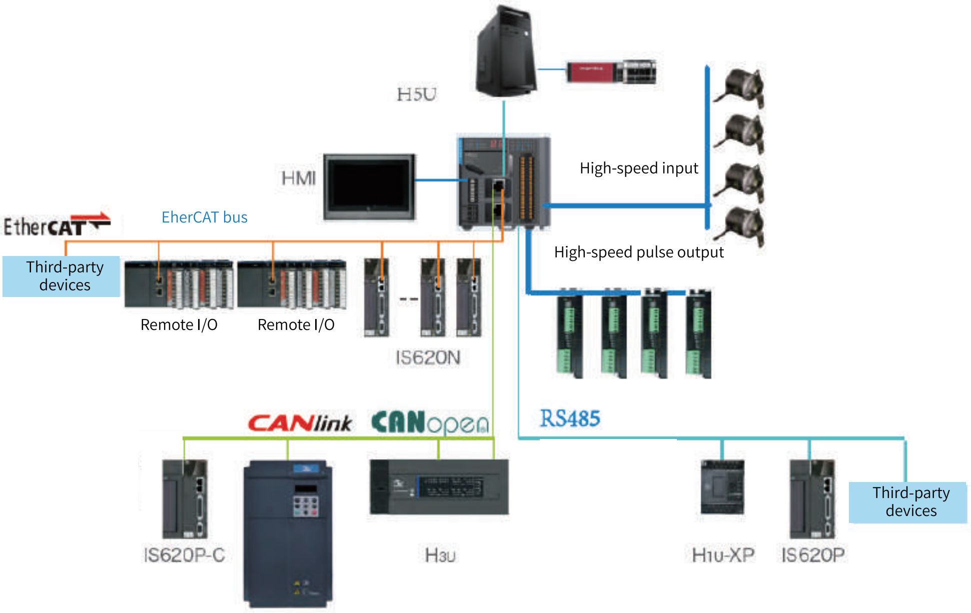
- Flexible communication mode: This feature allows AutoShop to communicate with PLC through COM, USB, and Ethernet. It enables the remote operation and remote collaborative commissioning functions for users, greatly facilitating users.
- Powerful network support: This feature enables the Modbus standard communication function based on configuration and supports CANopen communication configuration and Inovance CANlink communication configuration, greatly reducing difficulties and improving the working efficiency.
- Powerful motion control function: AutoShop has abundant motion control instructions and supports many functions such as the G codes, axis positioning, electronic cam, and flying shear/ chasing shear.
- Convenient and diverse commissioning methods: AutoShop supports various functions such as motion profile graphics, monitoring, online modification, oscilloscope, and fault diagnosis, facilitating user commissioning and localization.
- Powerful intellectual property protection function: Functions such as upload password, download password, identifier, and upload forbidding help to effectively protect the intellectual property.

1.1.3 Networking Schemes

H5U series networking scheme

H5U provides the EtherCAT, CAN, Ethernet, and RS485 interfaces to implement multi-level network communication and meet requirements of multiple scenarios. It is equipped with four high-speed inputs, four medium-speed inputs, and four high-speed outputs, which can realize 4-axis pulse output and 4-axis encoder counting.

The following figure shows a typical application topology.



Easy series networking scheme

Two Easy series models are used as an example to introduce the typical application topology, as shown in the following figures.

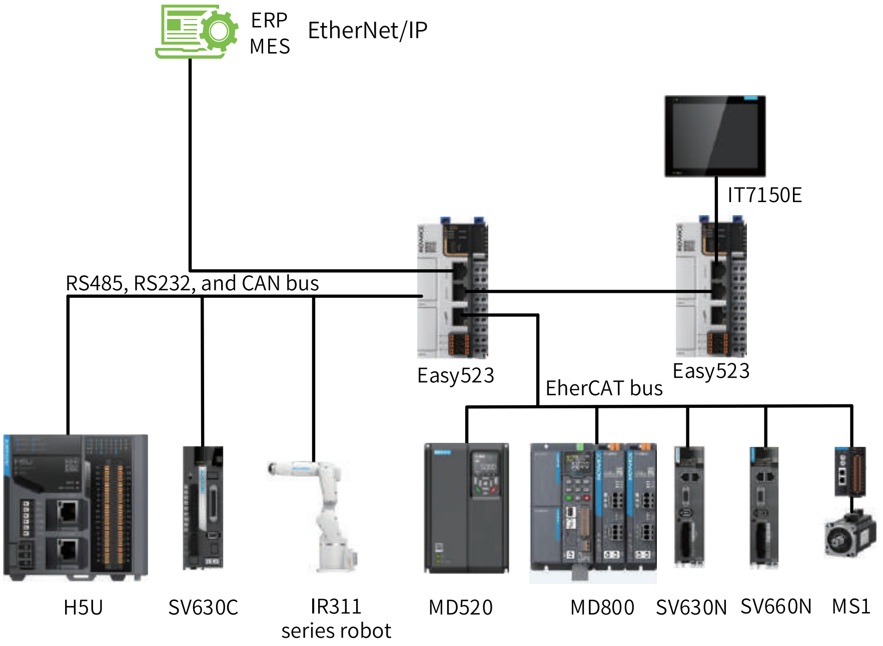


Figure 1-1 Easy523 series application topology

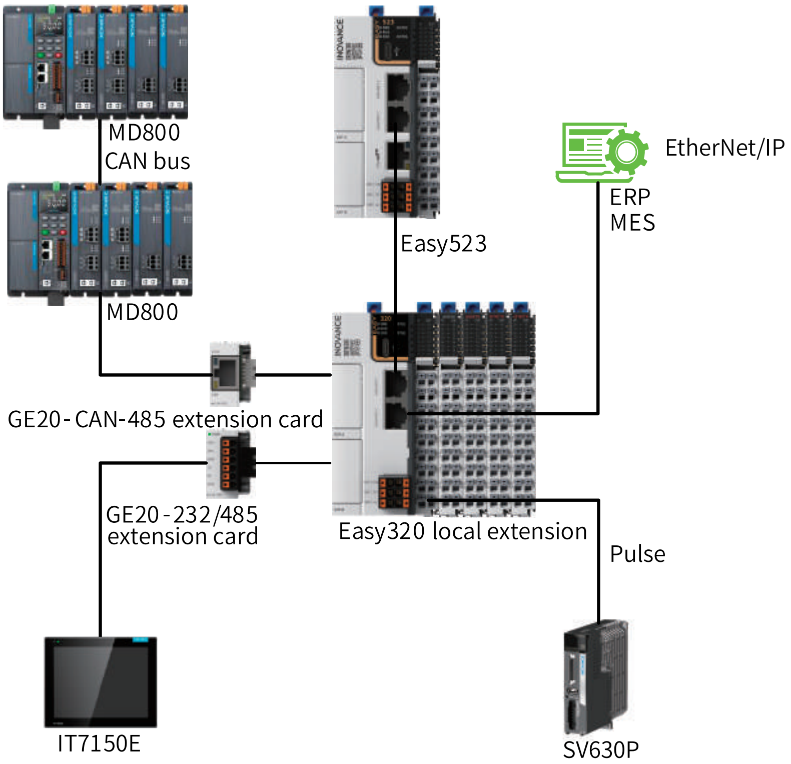


Figure 1-2 Easy320 series application topology

1.2 Obtaining and Installing the Software

1.2.1 How to Obtain

AutoShop programming software is provided for free. To obtain it, visit *www.inovance.com*, search for the keyword, and download the guide.

Note

As Inovance constantly improves its products and documentation, it is recommended that you update your software versions and consult the latest reference materials when needed to facilitate your application design.

1.2.2 Installation Environment Requirements

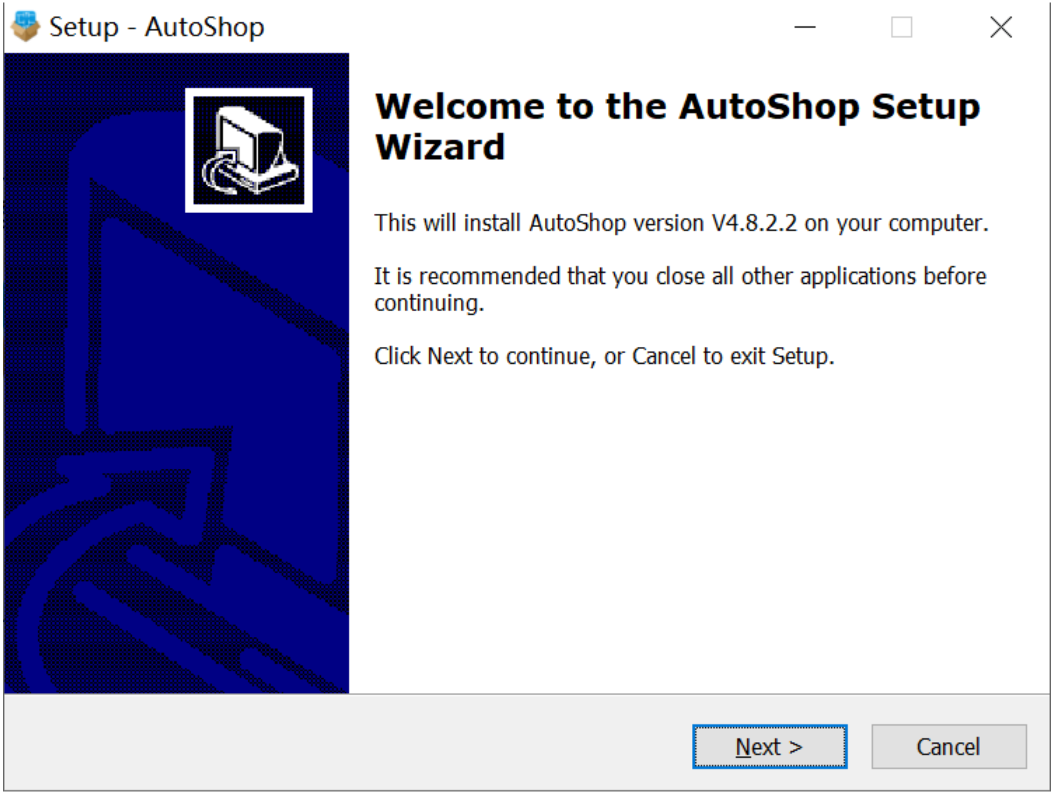
The following table lists the items required for a PC where AutoShop programming software is installed.

Item	Requirement
Operating system	Windows 7 or 10, 64-bit is recommended
Primary frequency of CPU	4 GHz or above
Memory	4 GB or above
Hard drive	More than 5 GB

1.2.3 Installing the Software

To install AutoShop, you can download the software installation package of the latest version at www.inovance.com, and the installation UI of the latest version shall prevail. AutoShop V4.8.1.0 is installed on Windows 10 as an example.

1. Decompress the "AutoShop V4.8.1.0 Setup.zip" package.
2. Double-click "AutoShop V4.8.1.0 Setup.exe." In the "Select language" dialog box, select a language and click "OK".
3. Click "Next".



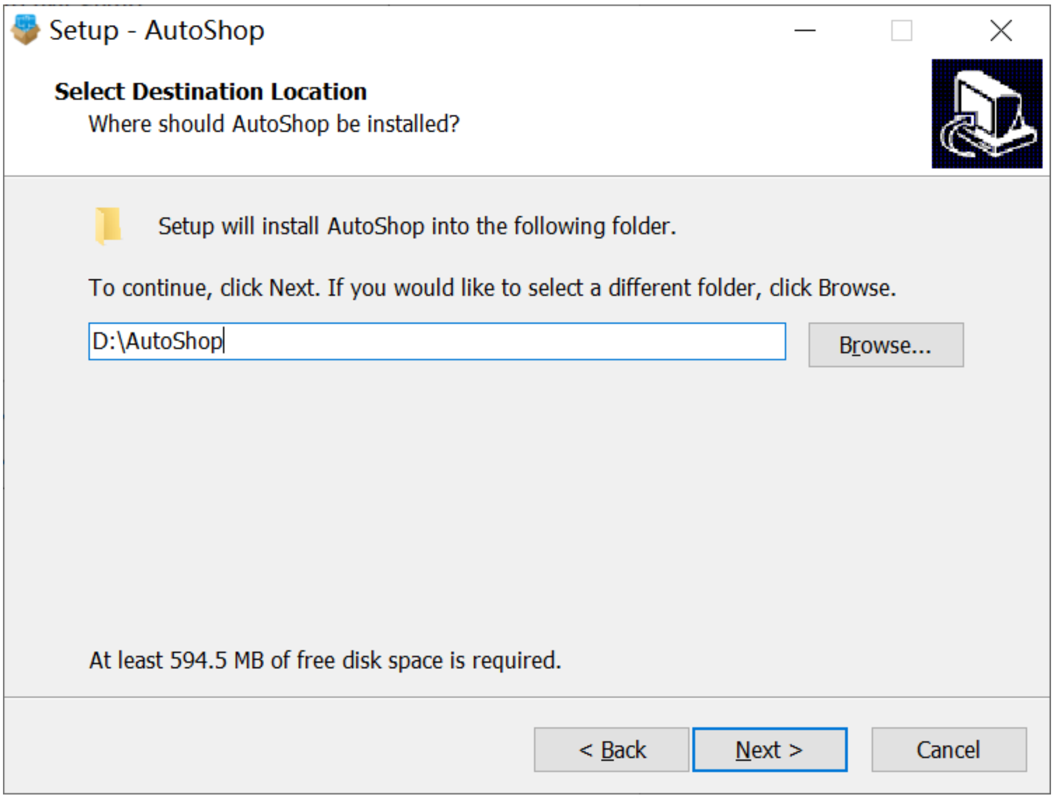
Welcome to the AutoShop Setup Wizard

This will install AutoShop version V4.8.2.2 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

4. Click "Browse". In the dialog box, select an installation path and click "Next". In general, the default path is used.



Setup - AutoShop

Select Destination Location

Where should AutoShop be installed?

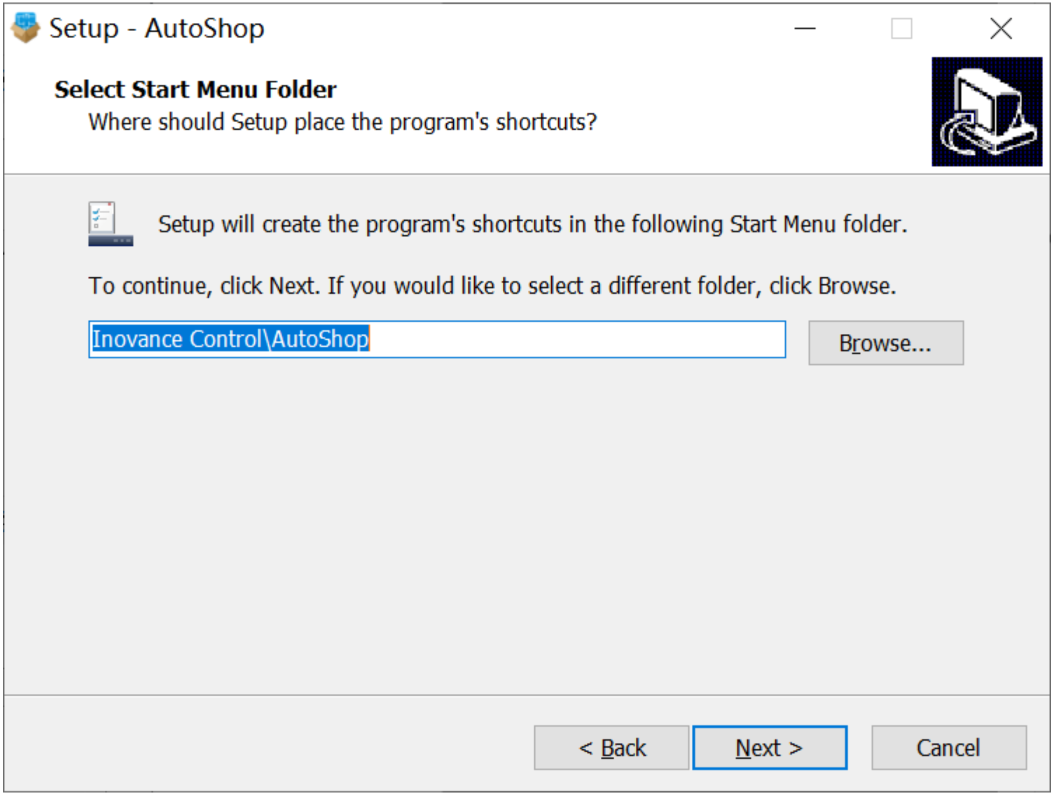
Setup will install AutoShop into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

D:\AutoShop

At least 594.5 MB of free disk space is required.

5. Click "Browse". In the dialog box, select an installation path for the program shortcut and click "Next". In general, the default path is used.



Setup - AutoShop

Select Start Menu Folder

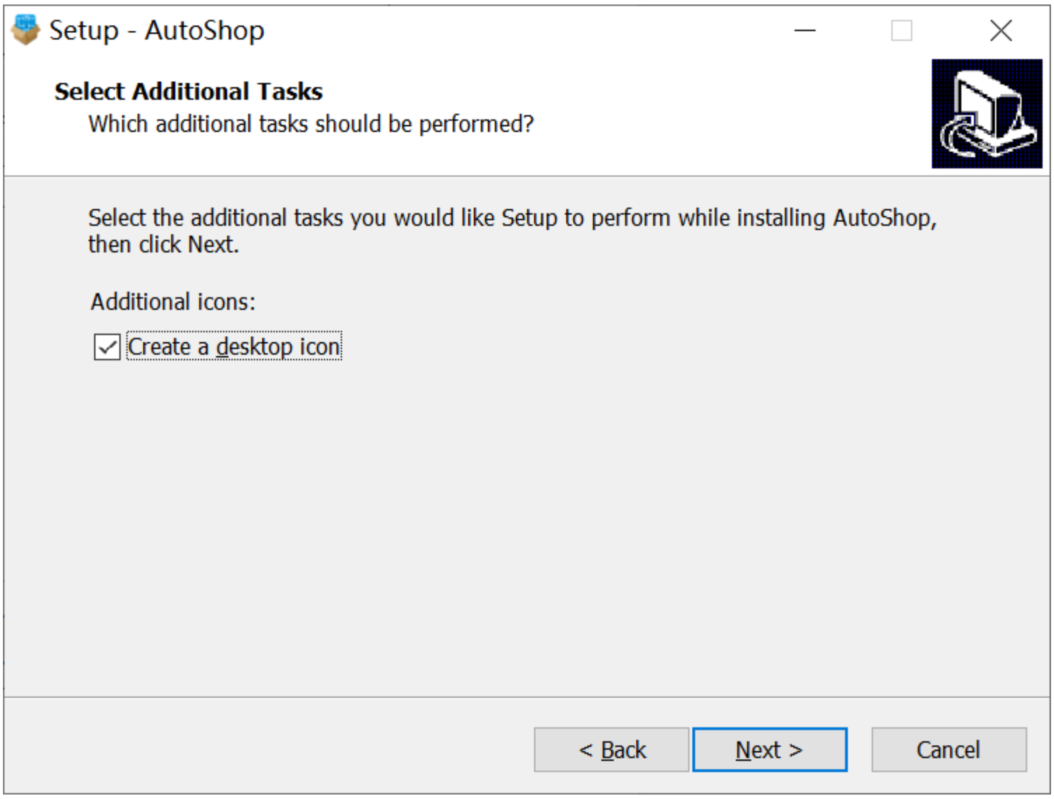
Where should Setup place the program's shortcuts?

Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

Inovance Control\AutoShop

6. Click "Next".



Setup - AutoShop

Select Additional Tasks

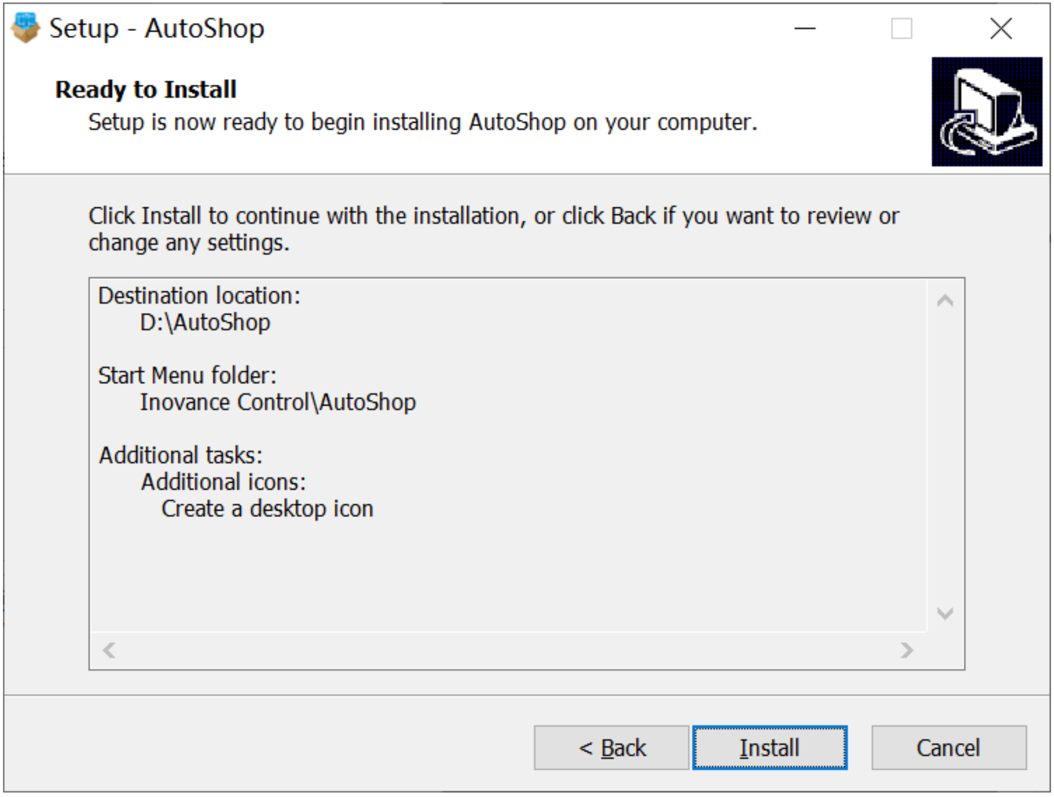
Which additional tasks should be performed?

Select the additional tasks you would like Setup to perform while installing AutoShop, then click Next.

Additional icons:

☑ Create a desktop icon

7. Click "Install".



Setup - AutoShop

Ready to Install

Setup is now ready to begin installing AutoShop on your computer.

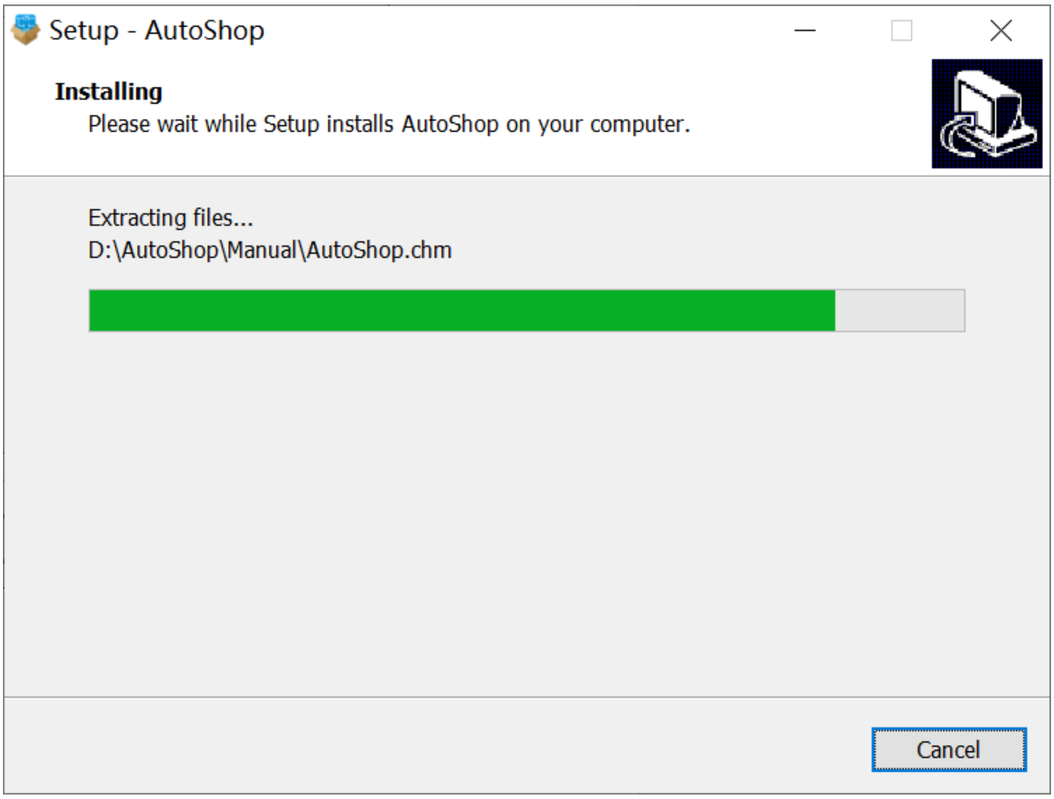
Click Install to continue with the installation, or click Back if you want to review or change any settings.

Destination location:
    D:\AutoShop

Start Menu folder:
    Inovance Control\AutoShop

Additional tasks:
    Additional icons:
      Create a desktop icon

The following figure shows the installation progress.



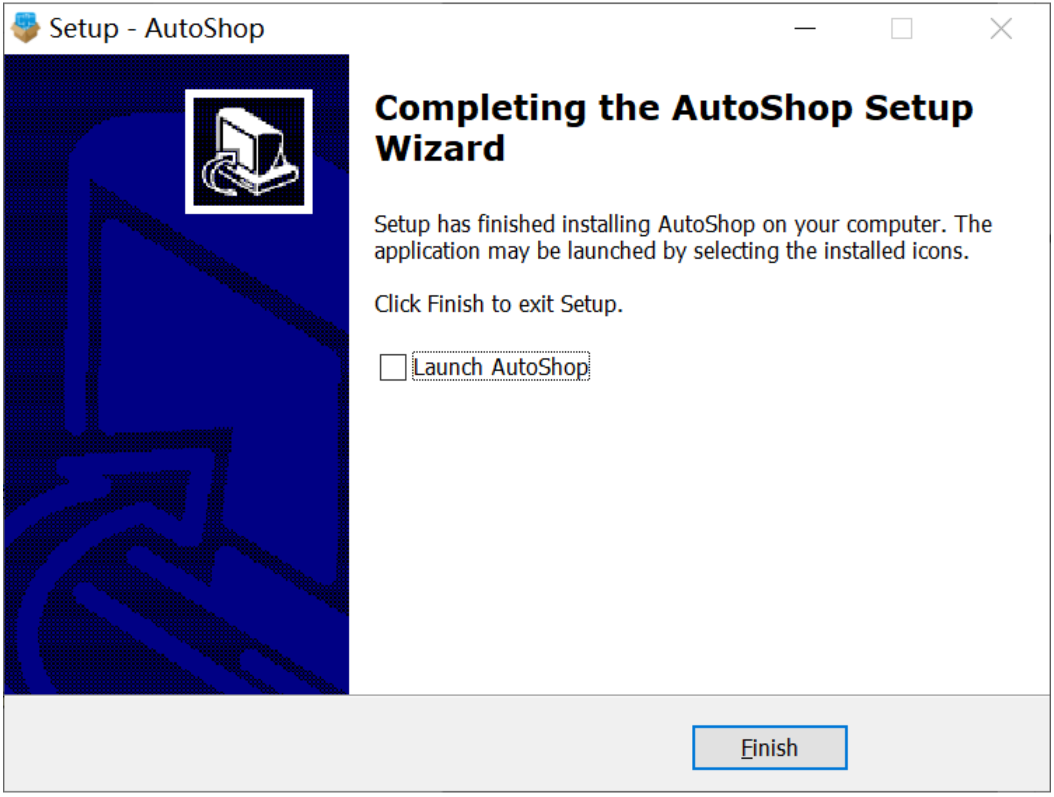
Setup - AutoShop

Installing

Please wait while Setup installs AutoShop on your computer.

Extracting files...
D:\AutoShop\Manual\AutoShop.chm

8. Click "Finish."



Setup - AutoShop

Completing the AutoShop Setup Wizard

Setup has finished installing AutoShop on your computer. The application may be launched by selecting the installed icons.

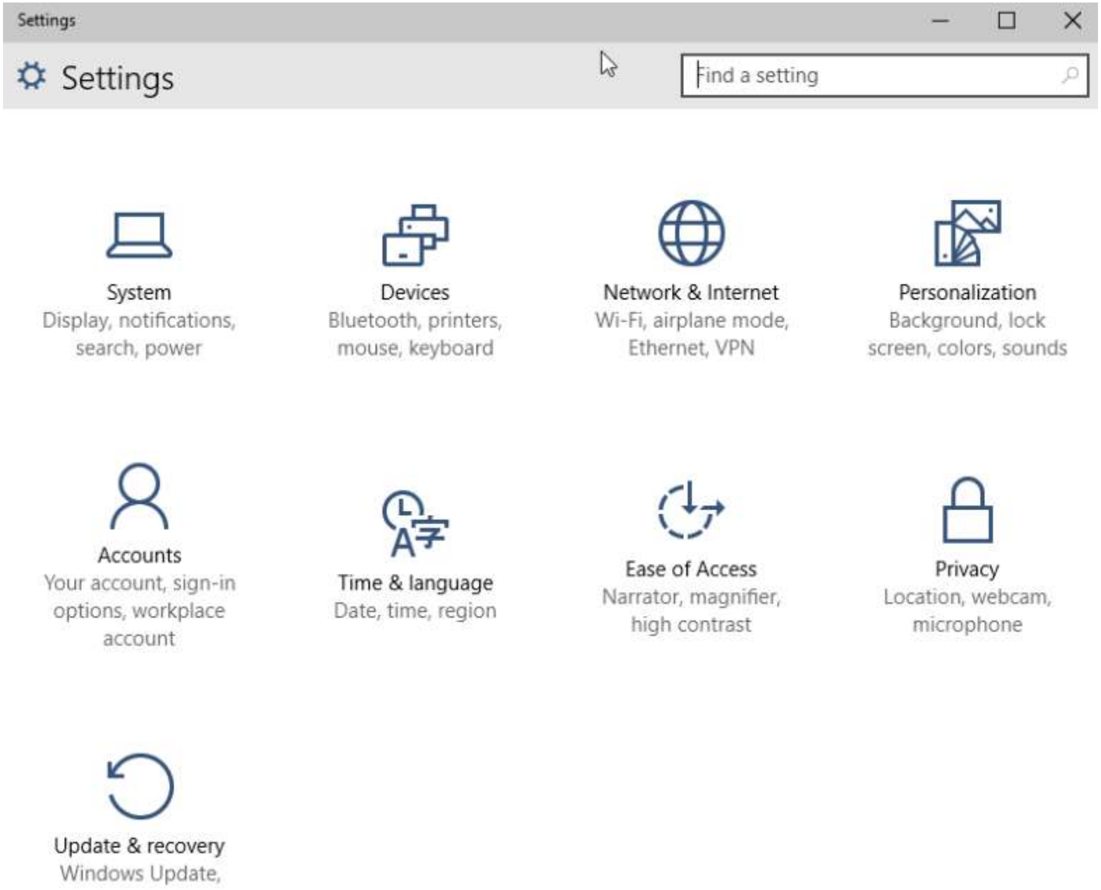
Click Finish to exit Setup.

□ Launch AutoShop

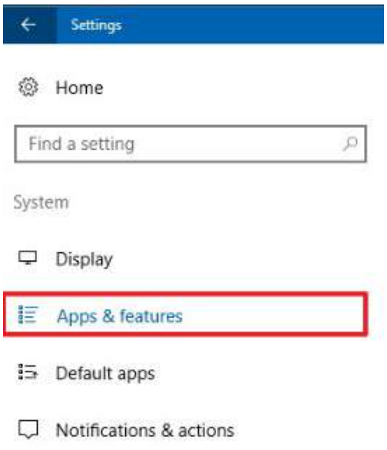
1.2.4 Uninstalling the Software

AutoShop V4.6.5.0 is uninstalled on Windows 10 as an example.

1. Click on the desktop. On the start menu that is displayed, click to access the "Windows setting" page.


Settings
Settings
Find a setting
System
Display, notifications, search, power
Devices
Bluetooth, printers, mouse, keyboard
Network & Internet
Wi-Fi, airplane mode, Ethernet, VPN
Personalization
Background, lock screen, colors, sounds
Accounts
Your account, sign-in options, workplace account
Time & language
Date, time, region
Ease of Access
Narrator, magnifier, high contrast
Privacy
Location, webcam, microphone
Update & recovery
Windows Update,

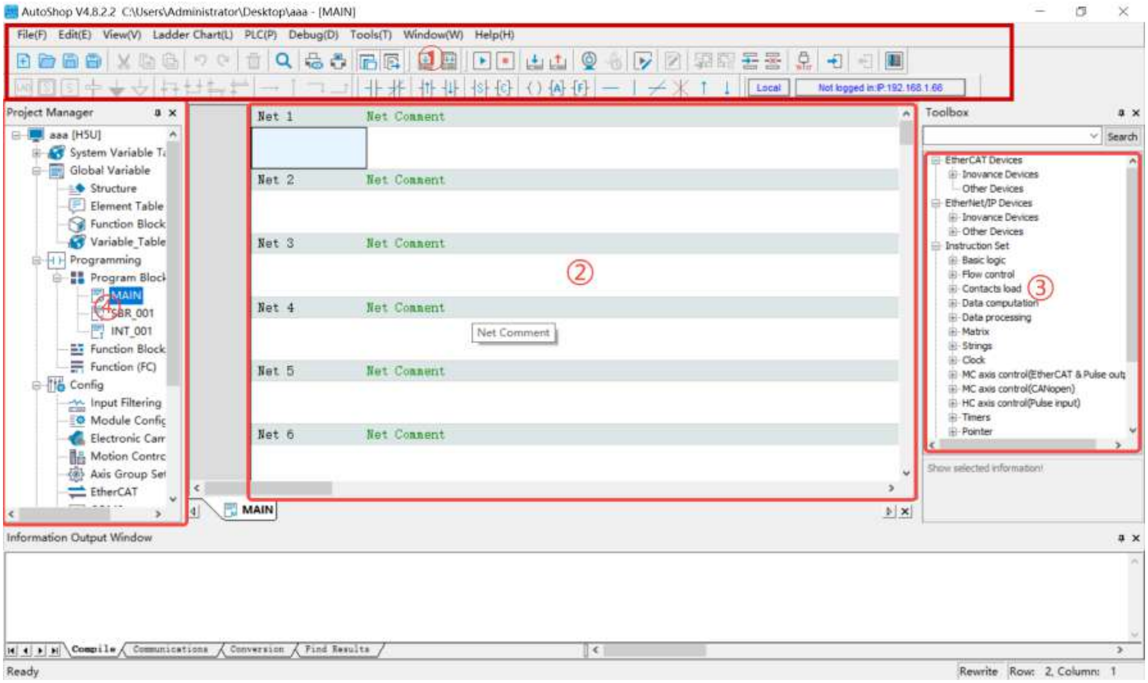
2. Click "Apps & Features" as shown below.



3. On the "Apps & Features" page, click "AutoShop Vx.x.x.x" and then "Uninstall".
4. Click "Uninstall". The "AutoShop uninstall wizard" dialog box is displayed.
5. Click "Yes". After the software is uninstalled, click OK.

1.3 Software Interface

The main screen of AutoShop programming software consists of the menu bar, toolbar, engineering management section, program editing section, and toolbox, as shown in the following figure.



No.	UI	Description
①	Menu bar and toolbar	Programming software operation menus, containing settings of programming, commissioning, and communication, and shortcut modes of file management and programming commissioning tools.
②	Program editing section	Compiling application programs for users
③	Tool kit	Set of instructions supported by the slave station and select PLC in a project. Tool kits are classified into ladder diagram ones and ST ones and can be switched to each other. The two types only differ in instructions supported in the instruction sets.
④	Engineering management	Including parameter management, variable management, program management, and configuration management of the PLC project.

2 Quick Start

2.1 Overview

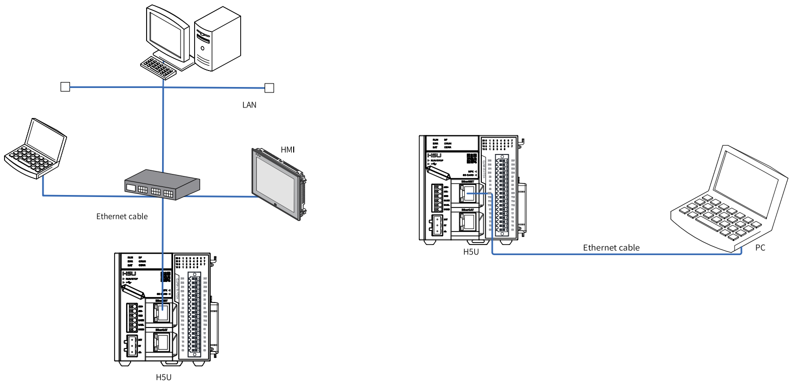
This section provides a simple programming example and general functions used in the programming commissioning process for you to quickly master programming and commissioning. It is quite helpful to beginners.

2.2 Communication Connection

2.2.1 Overview

An PLC communicates with the PC where AutoShop is located through USB or Ethernet connection to upload, download, monitor, and commission the program.

For example, H5U communicates with the PC through Ethernet, achieving multi-point connection or point-to-point connection in hub-based, switch-based, or direction connection mode, as shown in the following figure.



2.2.2 Ethernet Connection

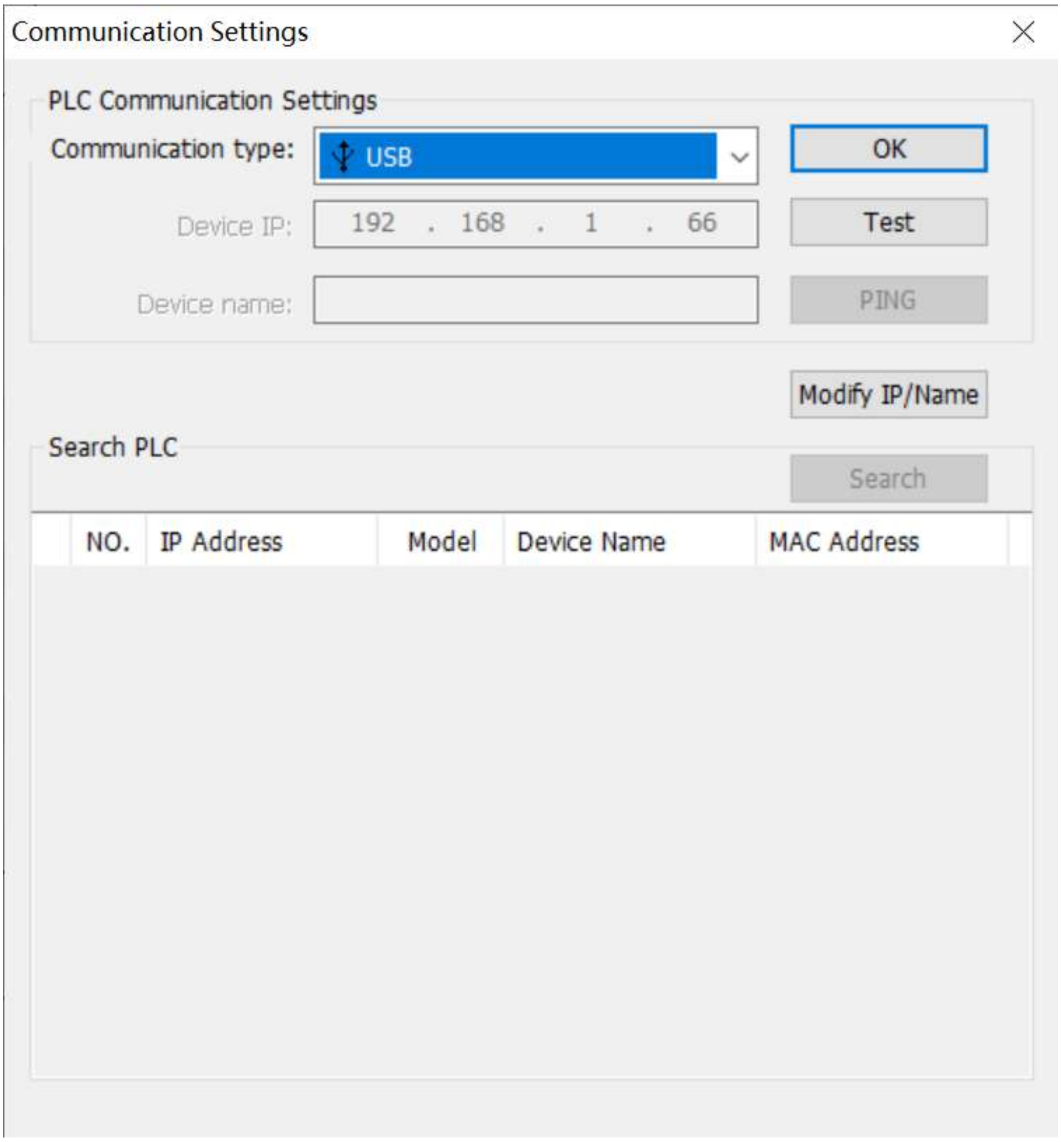
When connecting a PC to an PLC through Ethernet connection, you may need to connect to the target PLC and change the PLC IP address and device name.

Connecting to the target PLC

To connect a PC to an PLC through Ethernet connection, you need to select the PLC with the specified IP address.

- If the IP address of the PLC is provided, configure the communication type and IP address of the PLC and then connect to the target PLC.

  1. Connect the PC to the PLC using a network cable.
  2. Double-click on the desktop of the PC to start the AutoShop programming software.
  3. Choose "Tools" > "Communication Settings" in the menu bar or click in the toolbar. The "Communication Settings" dialog box is displayed.



Communication Settings

PLC Communication Settings

Communication type: USB

Device IP: 192 . 168 . 1 . 66

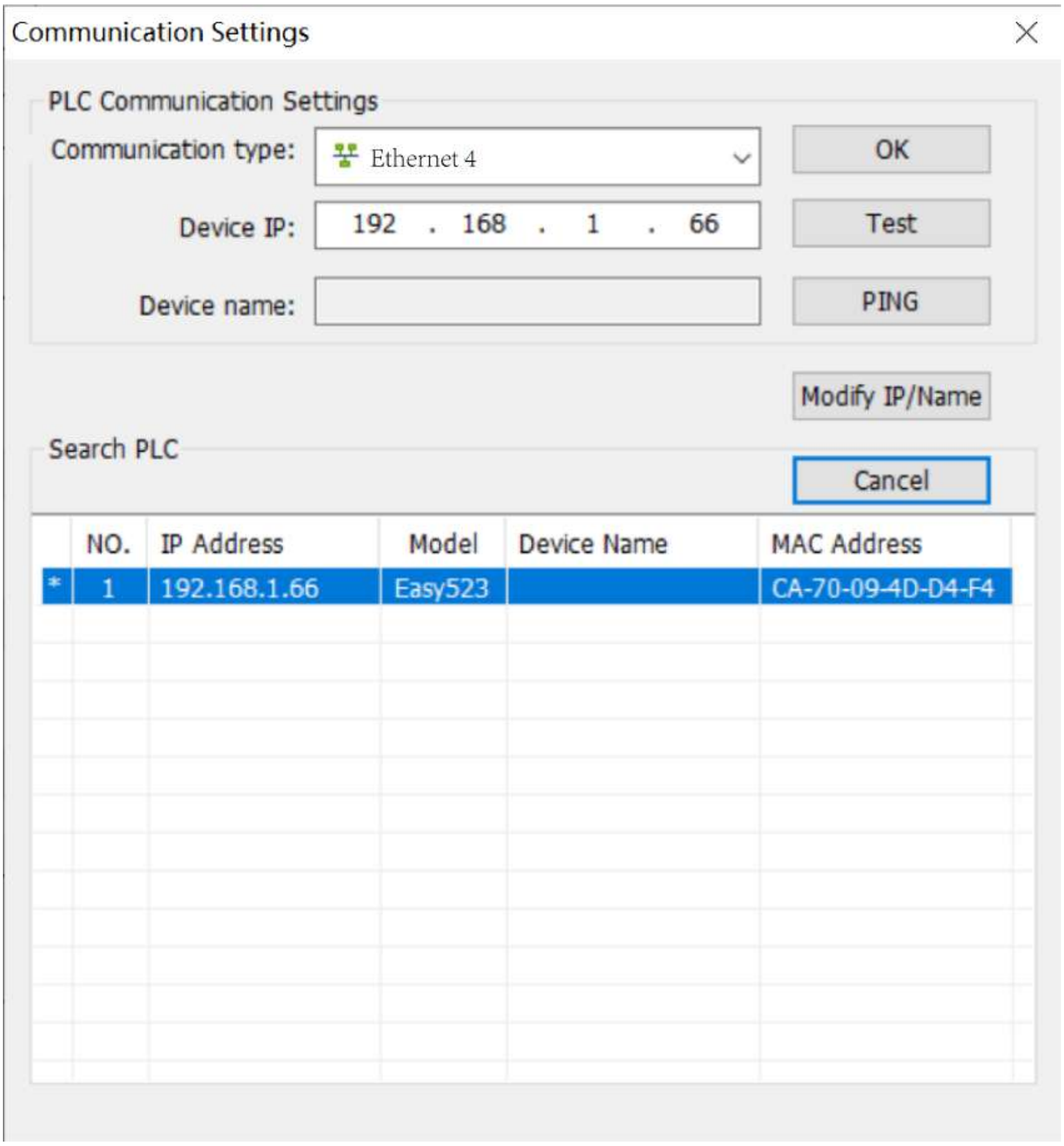
Device name:

Search PLC

NO.	IP Address	Model	Device Name	MAC Address

4. Set parameters in the "Communication Settings" dialog box.

- Set "Communication type" to "Ethernet".
- Set "Device IP" to the actual IP address of the PLC.



Communication Settings

PLC Communication Settings

Communication type: Ethernet 4

Device IP: 192 . 168 . 1 . 66

Device name:

Search PLC

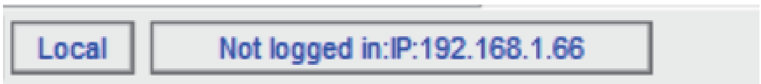
	NO.	IP Address	Model	Device Name	MAC Address
*	1	192.168.1.66	Easy523		CA-70-09-4D-D4-F4

Note

To test the network connection between the PC and the PLC, click "PING".

5. Click "Test" to check whether the target PLC is connected successfully.
   - H5U: When LEDs of the connected H5U display 0 alternatively, the H5U is connected successfully.
   - Easy: When the RUN indicator of the connected Easy flashes, the Easy is connected successfully.
6. Click "OK" to connect to the target PLC.

   Then, the IP address of the PLC is displayed in the toolbar and you can download, upload, monitor, or modify the PLC program online.

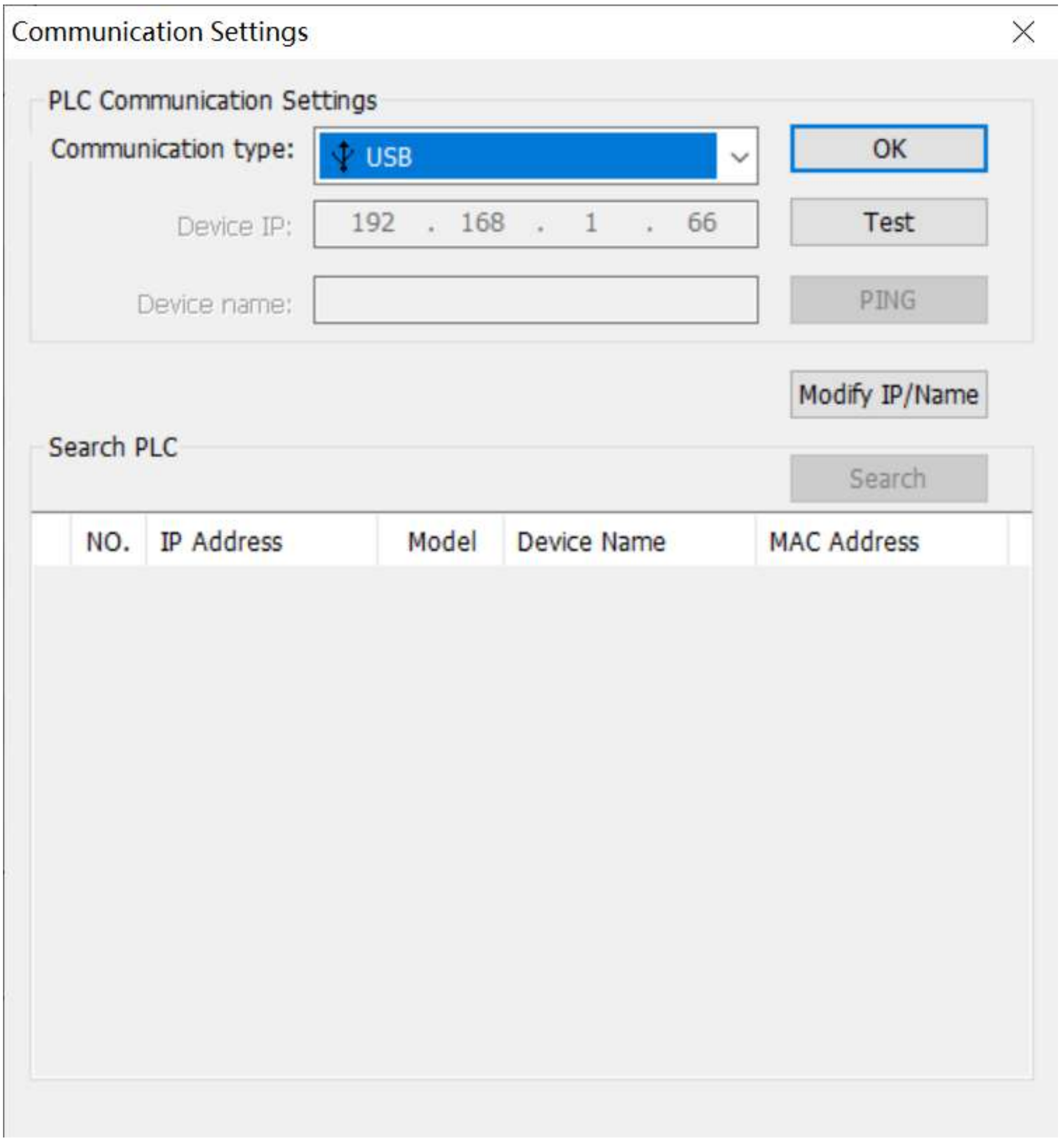


Local

Not logged in:IP:192.168.1.66

- If the IP address of the PLC is not provided, you can search for the target PLC.

1. Connect the PC to the PLC using a network cable.
2. Double-click on the desktop of the PC to start the AutoShop programming software.
3. Choose "Tools" > "Communication Settings" in the menu bar or click in the toolbar. The "Communication Settings" dialog box is displayed.



Communication Settings

PLC Communication Settings

Communication type: USB

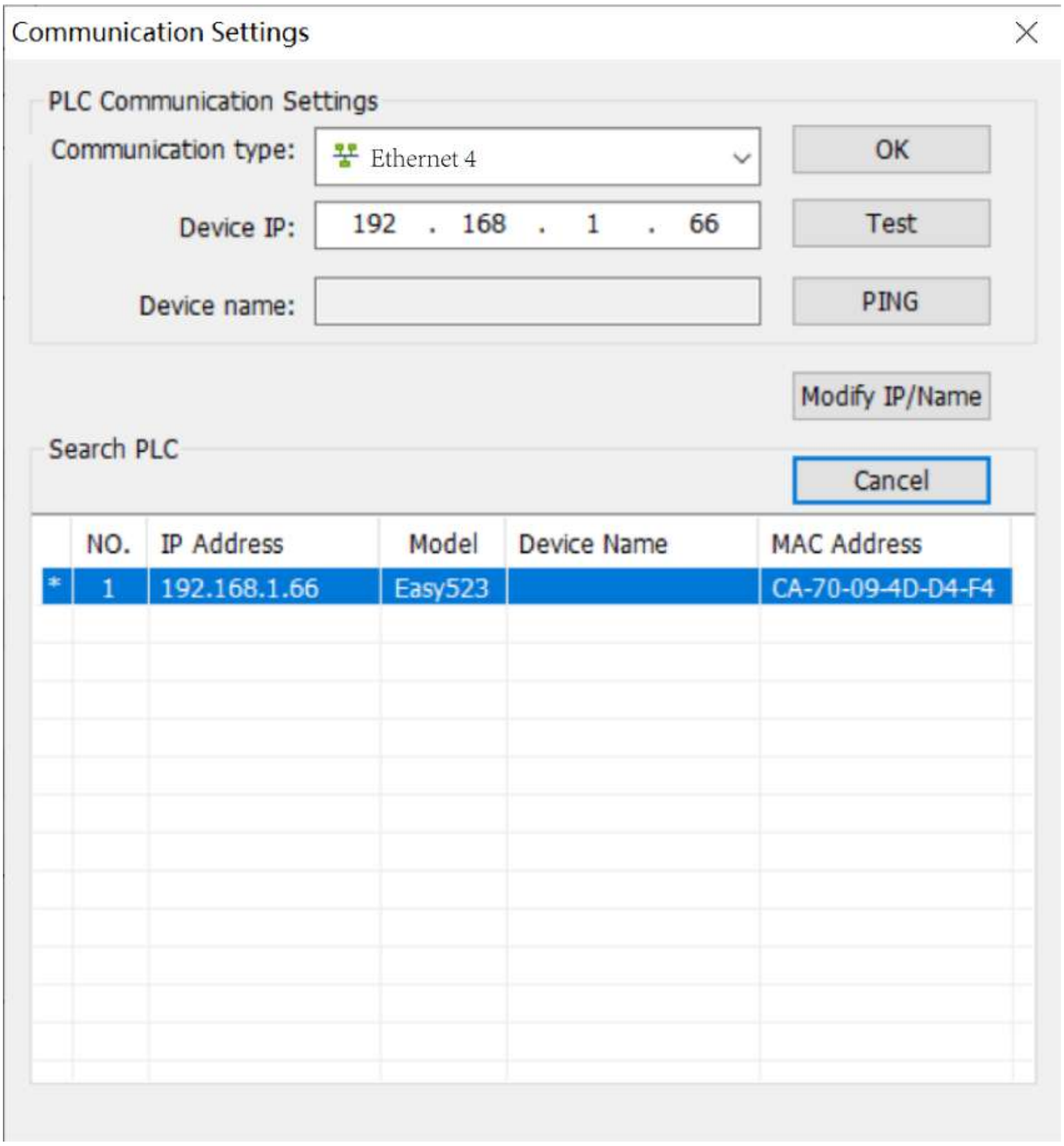
Device IP: 192 . 168 . 1 . 66

Device name:

Search PLC

NO.	IP Address	Model	Device Name	MAC Address

4. Set "Communication type" to "Ethernet", and click "Search" to search for the PLC connected in the LAN.
   - In switch-based connection mode, you can only search for the PLC in the same segment of the PC.
   - In direct connection mode, you can search for the PLC in the same segment or a different segment of the PC.



Communication Settings

PLC Communication Settings

Communication type: Ethernet 4

Device IP: 192 . 168 . 1 . 66

Device name:

Search PLC

	NO.	IP Address	Model	Device Name	MAC Address
*	1	192.168.1.66	Easy523		CA-70-09-4D-D4-F4

5. Select the IP address row in the search result and click "Test" to check whether the PLC is connected successfully.

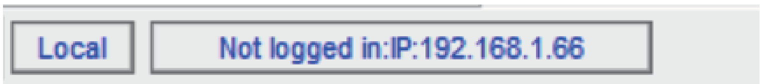
   - H5U: When LEDs of the connected H5U display 0 alternatively, the H5U is connected successfully.
   - Easy: When the RUN indicator of the connected Easy flashes, the Easy is connected successfully.

Note

To test the network connection between the PC and the PLC, click "PING".

6. Click "OK" to connect to the target PLC.

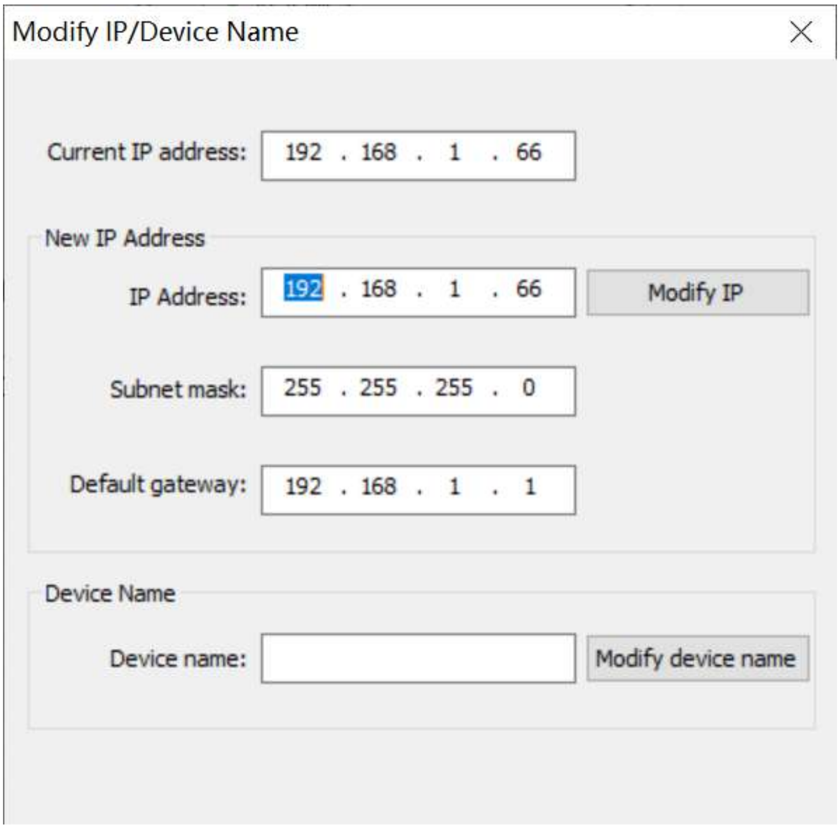
   Then, the IP address of the PLC is displayed in the toolbar and you can download, upload, monitor, or modify the PLC program online.



Changing the PLC IP address and device name

You can change the IP address or the PLC as needed or change the device name of the PLC for differentiation.

1. After connecting to the target PLC, click "Modify IP/Name" in the "Communication Settings" dialog box. The "Modify IP/Name" dialog box is displayed.



Modify IP/Device Name

Current IP address: 192 . 168 . 1 . 66

New IP Address

IP Address: 192 . 168 . 1 . 66

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 192 . 168 . 1 . 1

Device Name

Device name:

2. Set "IP Address", "Subnet mask", and "Default gateway" in the "New IP Address" section, and click "Modify IP". In the dialog box that is displayed, click "OK". The new IP address takes effect after the PLC is powered on again.

Note

When the new PCB software (V3.0.0.0) is used with an old AutoShop background (V4.0.0.0 or earlier version), the following situation will occur:

When you modify the IP in the communication settings, even if the software prompts success, the IP may not be successfully modified.

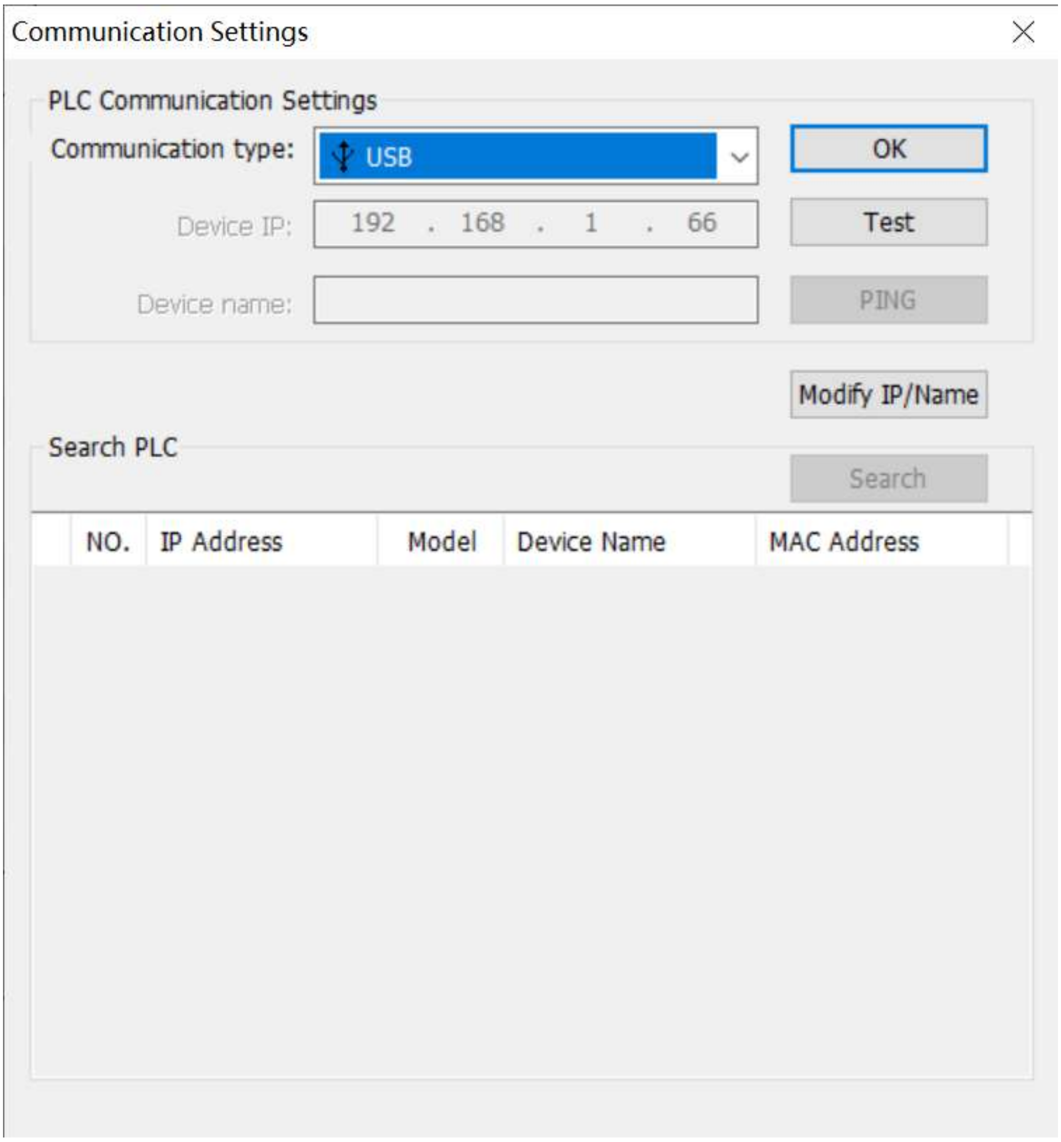
3. In the "Device Name" section, set "Device name" as needed and click "Modify device name". In the dialog box that is displayed, click "OK".

2.2.3 USB Connection

When connecting a PC to an PLC through USB connection, you may need to connect to the target PLC and change the PLC IP address and device name.

Connecting to the target PLC

1. Connect the PC to the PLC using a USB cable.
2. Double-click on the desktop of the PC to start the AutoShop programming software.
3. Choose "Tools" > "Communication Settings" in the menu bar or click in the toolbar. The "Communication Settings" dialog box is displayed.



Communication Settings

PLC Communication Settings

Communication type: USB

Device IP: 192 . 168 . 1 . 66

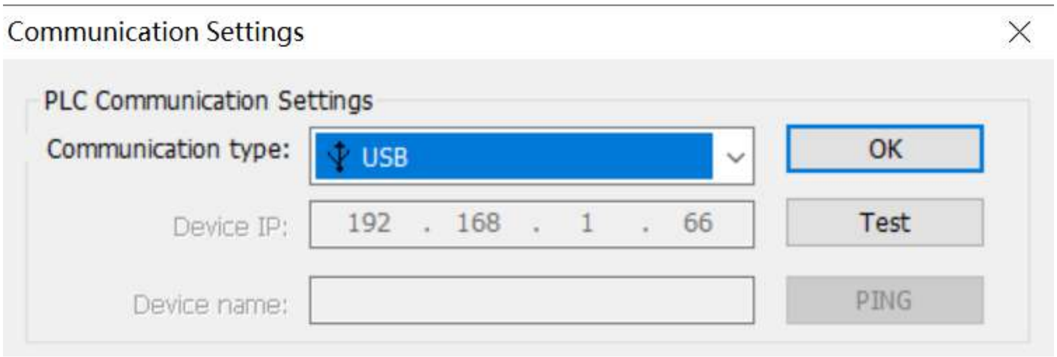
Device name:

Search PLC

NO.	IP Address	Model	Device Name	MAC Address

4. Set "Communication type" to "USB" and click "Test" to check whether the target PLC is connected successfully.

- H5U: When LEDs of the connected H5U display 0 alternatively, the H5U is connected successfully.
- Easy: When the RUN indicator of the connected Easy flashes, the Easy is connected successfully.



Communication Settings

PLC Communication Settings

Communication type: USB

Device IP: 192 . 168 . 1 . 66

Device name:

Caution

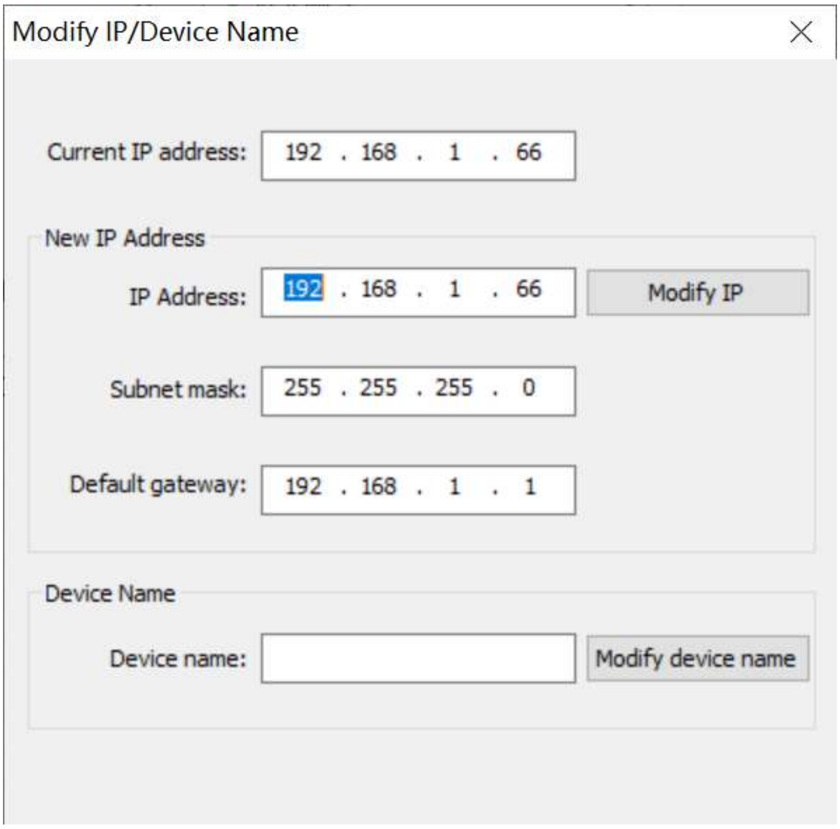
If AutoShop V4.0.0.0 is used with an old PCB software (V3.0.0.0 or earlier versions), clicking "Test" will not initiate connection to the H5U device. In this case, upgrade the PLC software.

5. Click "OK" to connect to the target PLC.
   Then, you can download, upload, monitor, or modify the PLC program online.

Changing the PLC IP address and device name

You can change the IP address or the PLC as needed or change the device name of the PLC for differentiation.

1. After connecting to the target PLC, click "Modify IP/Name" in the "Communication Settings" dialog box. The "Modify IP/Name" dialog box is displayed.



Modify IP/Device Name

Current IP address: 192 . 168 . 1 . 66

New IP Address

IP Address: 192 . 168 . 1 . 66

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 192 . 168 . 1 . 1

Device Name

Device name:

2. Set "IP Address", "Subnet mask", and "Default gateway" in the "New IP Address" section, and click "Modify IP". In the dialog box that is displayed, click "OK". The new IP address takes effect after the PLC is powered on again.

Note

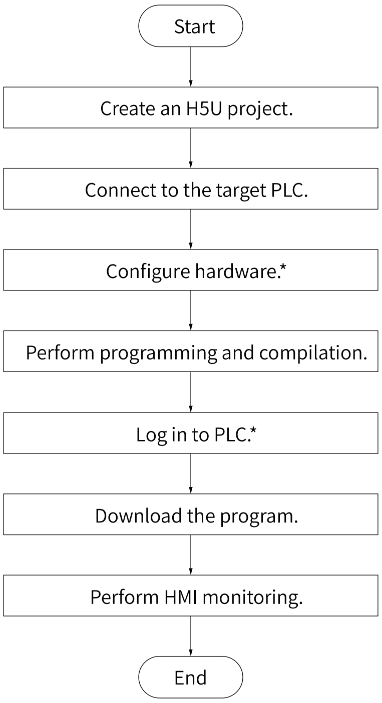
When the new PCB software (V3.0.0.0) is used with an old AutoShop background (V4.0.0.0 or earlier version), the following situation will occur:

When you modify the IP in the communication settings, even if the software prompts success, the IP may not be successfully modified.

3. In the "Device Name" section, set "Device name" as needed and click "Modify device name". In the dialog box that is displayed, click "OK".

2.3 Programming Process

A typical user program flowchart of the H5U series PLC is compiled and commissioned as an example, as shown in the following figure.



Note

* If only the H5U host is used, skip the "Hardware configuration" step. If connected to a PLC without a login password, skip the "Login PLC" step.

2.4 Programming Example

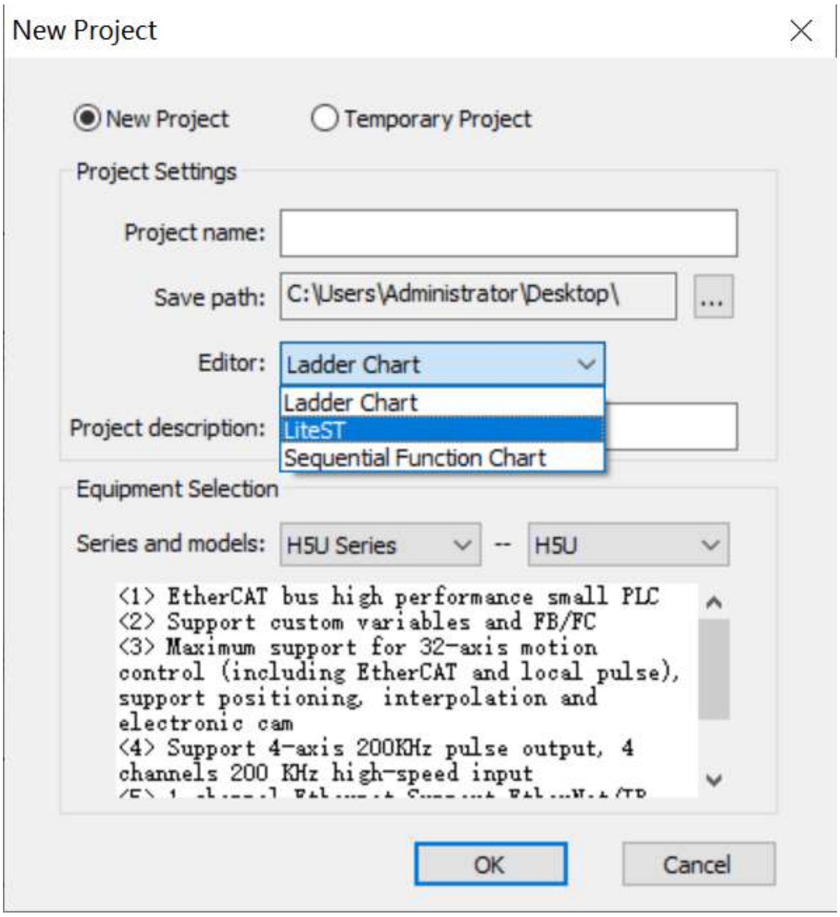
2.4.1 Example Requirements

Compile and commission the next marquee program based on the following requirements:

Using a 16-output module, PLC program shifts one bit rightwards from Y20 every 1s. When the output reaches Y27, the PLC program returns the output to Y20 for a new cycle, which is monitored through HMI.

2.4.2 Creating a Project

1. Double-click the AutoShop shortcut icon on the desktop.
2. Choose "File" > "New Project" in the menu bar or click in the toolbar. In the dialog box that is displayed, choose an editor type from the "Editor" drop-down list, and set "Series and models" to "H5U".



New Project

● New Project ○ Temporary Project

Project Settings

Project name:

Save path: C:\Users\Administrator\Desktop\

Editor: Ladder Chart

Ladder Chart
LiteST
Sequential Function Chart

Project description:

Equipment Selection

Series and models: H5U Series -- H5U

<1> EtherCAT bus high performance small PLC
<2> Support custom variables and FB/FC
<3> Maximum support for 32-axis motion control (including EtherCAT and local pulse), support positioning, interpolation and electronic cam
<4> Support 4-axis 200KHz pulse output, 4 channels 200 KHz high-speed input

3. Set "Project name" and "Save path", and click "OK" to create a project. Then, the main screen of the project is displayed.

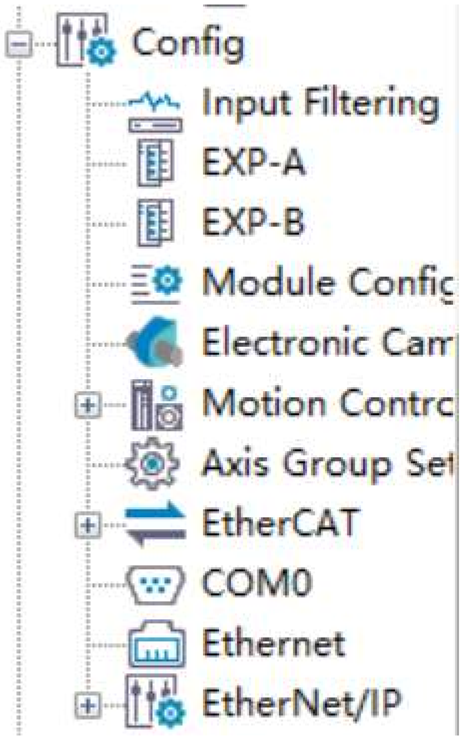
2.4.3 Connecting to Target PLC

This section takes the Ethernet connection between the PC and PLC as an example. For details about how to connect to the target PLC, see "Connecting to the target PLC" in *"2.2.2 Ethernet Connection" on page 29*.

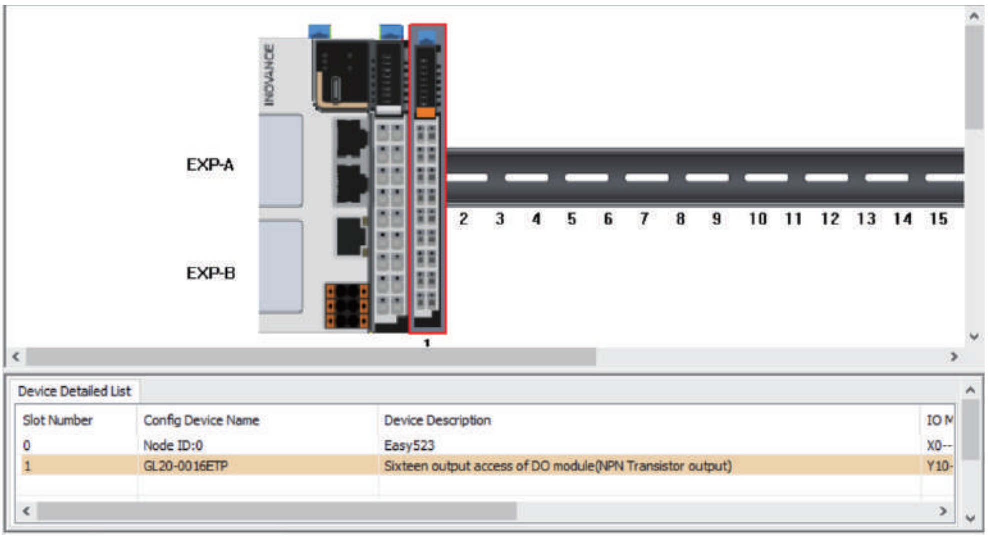
2.4.4 (Optional) Configuring Hardware

If the local extension module is installed for the H5U host, perform the steps in this section. Otherwise, skip this section.

1. In the "Project Manager" tree, unfold "Config" and double-click "Module Config". The "Module Config" page is displayed.

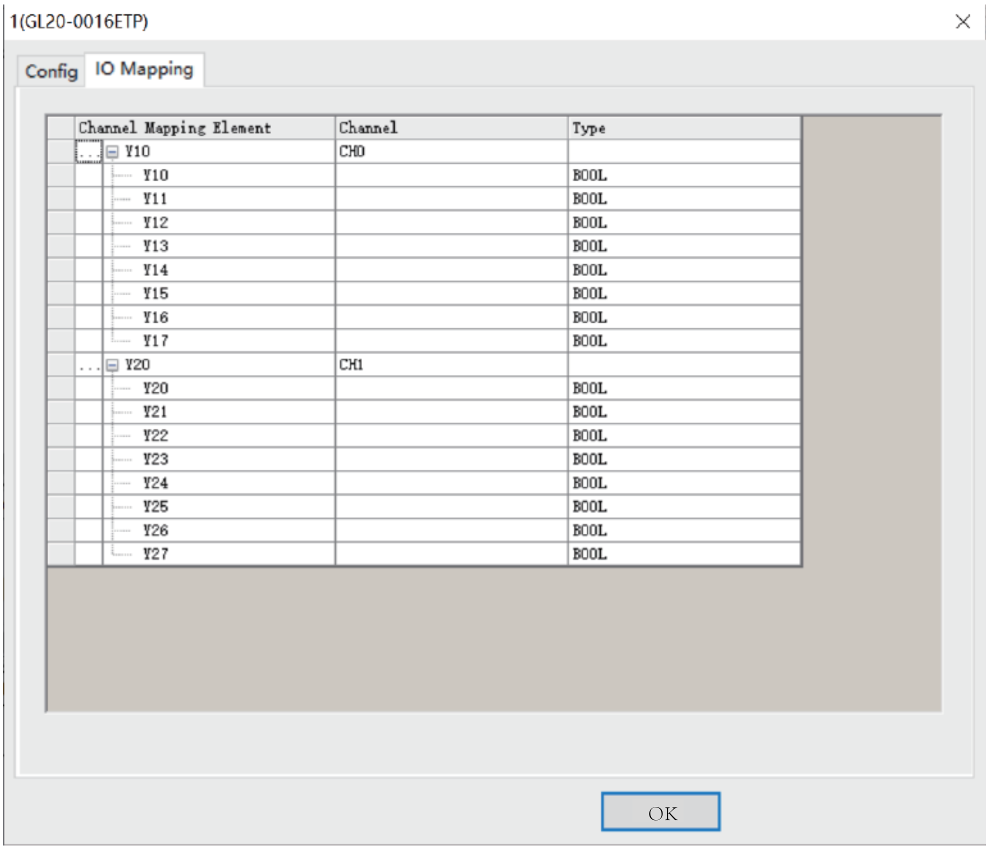


2. Double-click the modules under "Module" on the right of the page based on the installation sequence to add the corresponding modules.



Slot Number	Config Device Name	Device Description	IO M
0	Node ID:0	Easy523	X0--
1	GL20-0016ETP	Sixteen output access of DO module(NPN Transistor output)	Y10-

3. Double-click an added module and configure the channel mapping element corresponding to the module channel.

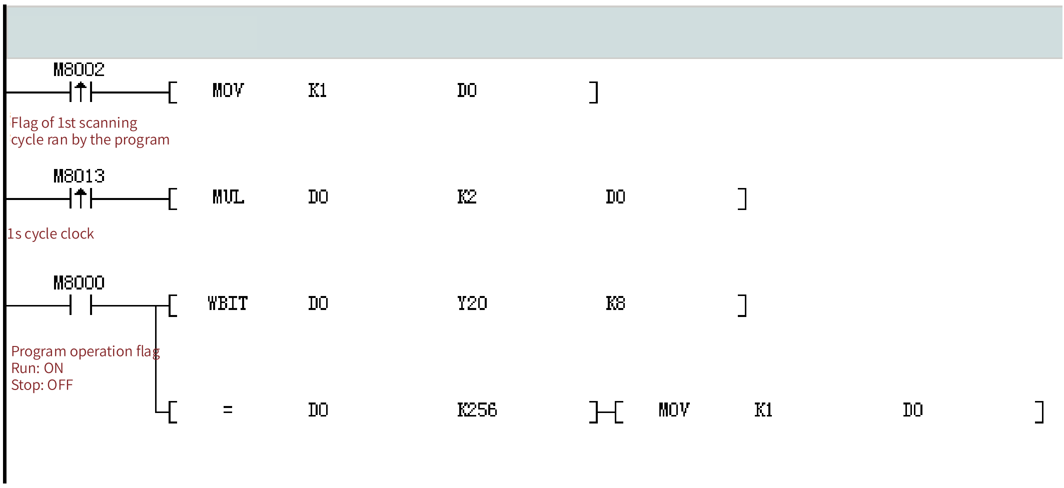


Channel Mapping Element	Channel	Type
Y10	CH0	
Y10		BOOL
Y11		BOOL
Y12		BOOL
Y13		BOOL
Y14		BOOL
Y15		BOOL
Y16		BOOL
Y17		BOOL
Y20	CH1	
Y20		BOOL
Y21		BOOL
Y22		BOOL
Y23		BOOL
Y24		BOOL
Y25		BOOL
Y26		BOOL
Y27		BOOL

4. Click "OK".

2.4.5 Programming and Compiling

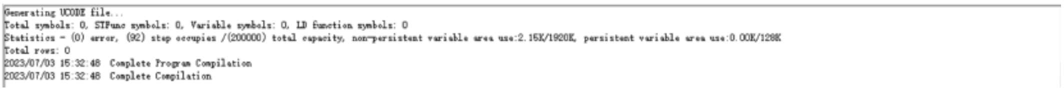
1. In the "Project Manager" tree, unfold "Programming" and double-click "MAIN" to compile the marquee program.



2. In the toolbar, click 📥 or 📥 for compiling.

- 📥: Compile the opened program.
- 📥: Compile the overall project.

After compiling is completed, the compilation information is displayed at the bottom of the main screen.

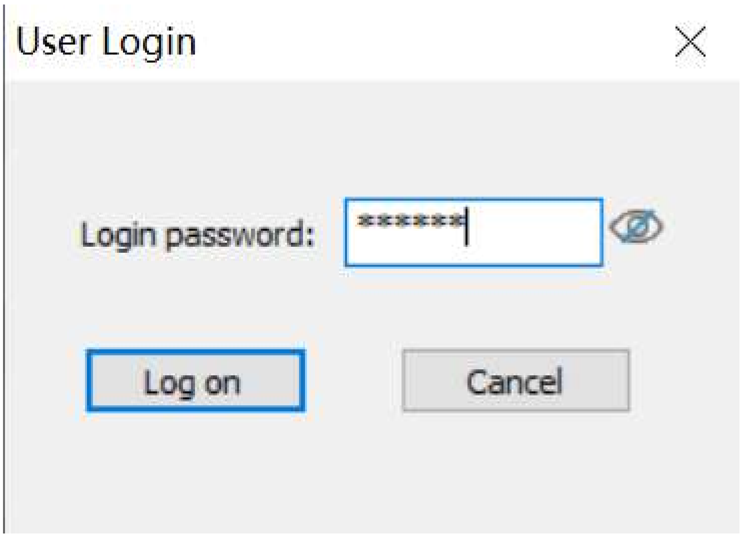


```
Generating UCODE file...
Total symbols: 0, STFunc symbols: 0, Variable symbols: 0, LD function symbols: 0
Statistics - (0) error, (92) step occupies /(200000) total capacity, non-persistent variable area use:2.15K/1920K, persistent variable area use:0.00K/128K
Total rows: 0
2023/07/03 15:32:48  Complete Program Compilation
2023/07/03 15:32:48  Complete Compilation
```

2.4.6 (Optional) Logging In to PLC

In case of Ethernet connection between a PC and a PLC with the login password set, to modify, upload, download, or verify the program online, perform the steps in this section. Otherwise, skip this section.

1. In the toolbar, click ➜. The "User Login" page is displayed.



User Login

Login password: ******

2. Enter the PLC login password in the "Login password" text box and click "Log on".

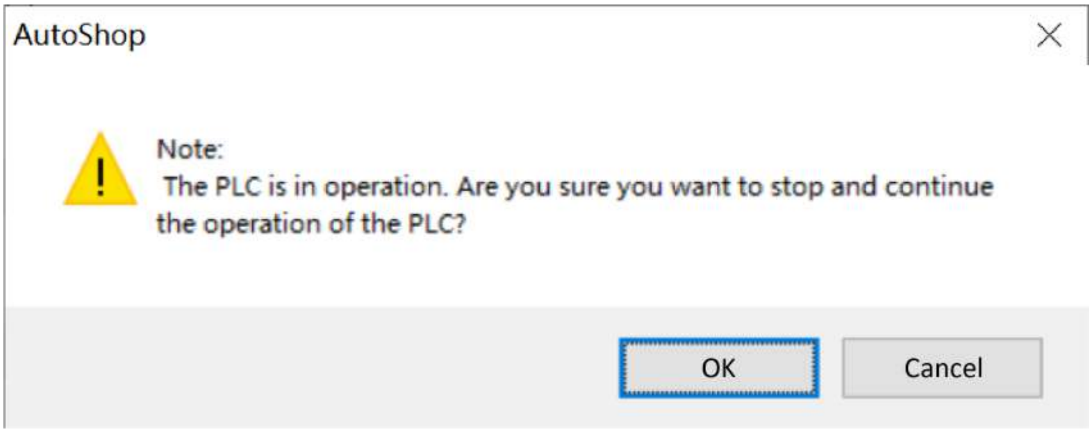
Note

For more details about logging in to the PLC, see section "2.10 Logging in to PLC".

2.4.7 Downloading Program

1. Choose "PLC" > "Download" in the menu bar or click ⤓ in the toolbar to download the program.

- If the PLC is in the running state, perform operations to access the prompt dialog box and click "OK", as shown in the following figure: Proceed to Step 2.



AutoShop

Note:
The PLC is in operation. Are you sure you want to stop and continue the operation of the PLC?

- If the PLC is in the stopped state: Proceed to Step 3.

2. Upon download completion, click "OK" in the dialog box to switch the PLC to the running state.

3. Upon download completion, click ▶ in the toolbar to switch the PLC to the running state.

Note

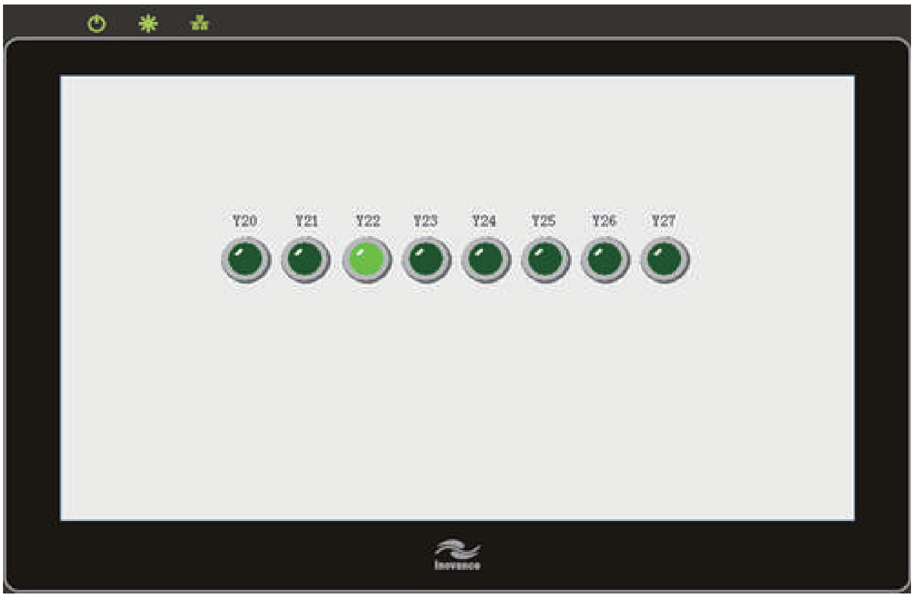
When the new PCB software (V3.0.0.0) is used with an old AutoShop background (V4.0.0.0 or earlier version), the following situation will occur:

If the PLC is not configured with a login password, projects downloaded through the new background cannot be uploaded through the old background.

2.4.8 HMI Monitoring

You can monitor the execution of the marquee program through the HMI.

1. Create a project, connection, or variable and view the configuration screen on the HMI side. For details, see *XX Series HMI Quick Start*.
2. When running the PLC program, you can monitor the marquee status change on the HMI configuration screen.



2.5 Switching PLC Working Modes

An PLC can work in RUN or STOP mode.

- RUN: The PLC detects input at the X point, scans and calculates the user program, refreshes elements, and enables output and communication at the Y point.
- STOP: The PLC stops scanning of the program and output at the Y point, and disables the communication function.

PLC working modes can be switched in either of the following two ways:

- Toggle the "RUN/STOP" switch on the PLC.
- In the toolbar, click ▶ to run the PLC or click ■ to stop the PLC.

2.6 Modifying Program Online

After the mode is enabled, online editing of the program does not affect the running PLC. Therefore, the program can be edited without the need of stopping the PLC, facilitating program commissioning.

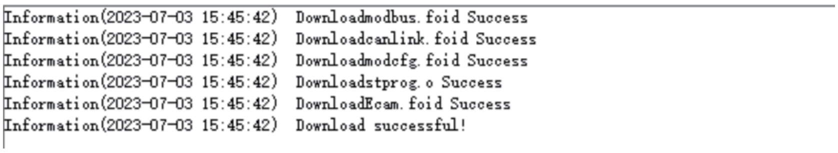
⚠ Caution

- Program modification does not support:
  - Adding or deleting program files
  - Renaming program files
  - Modifying program file properties
  - Encrypting or decrypting subprograms
- Modifying configuration is not supported.
- Global variables:
  - Adding or deleting variables is supported.
  - Modifying names or comments of variables is supported, but modifying other features is not supported.
- FB/FC local variables:
  - Adding, modifying, or deleting IN/INOUT/OUT type variables is not supported.
  - Adding or deleting Var type variables is supported.
  - Var variables added before the current online modification cycle: Modifying names or comments of Var type variables is supported, but modifying other features is not supported.
  - Var variables added during the current online modification cycle: Modifying any features of Var variables is supported.
- Do not power off the system within 5s to 10s after successful online modification and download of the program. Otherwise, program errors may occur.

1. Choose "PLC" > "Online Edit Mode" in the menu bar or click in the toolbar to enable the online modification mode. At the same time, the software enters the monitoring mode.
   In online modification mode, if the running program is different from the program in the PLC, the "Failed to enter online modification mode!" prompt dialog box is displayed. In this case, you need to check whether the program is started correctly.

2. Click to compile the program online as required.

3. Click to download the modified program to the running PLC. The PLC does not stop running during the operation.
   Information shown in the following figure is displayed in the lower part of the main screen. Then, the PLC runs with the modified program.



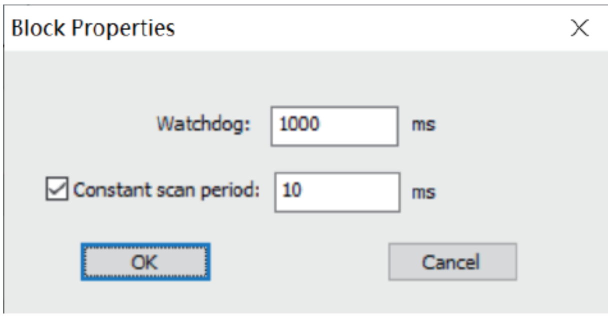
```
Information(2023-07-03 15:45:42)  Downloadmodbus.foid Success
Information(2023-07-03 15:45:42)  Downloadcanlink.foid Success
Information(2023-07-03 15:45:42)  Downloadmodcfg.foid Success
Information(2023-07-03 15:45:42)  Downloadstprog.o Success
Information(2023-07-03 15:45:42)  DownloadEcam.foid Success
Information(2023-07-03 15:45:42)  Download successful!
```

2.7 Setting Program Scan Cycles

You can select the constant scan cycle or the non-constant scan cycle.

- When the constant scan cycle is selected, the program runs by the specified scan cycle. If the actual running time is less than or equal to the set scan cycle, the PLC scans the program based on the set value. If the actual running time is greater than the set scan cycle, the PLC scans the program based on the actual running time.
- When the non-constant scan cycle is selected, the PLC automatically adjusts the scan cycle based on the running time of the program.

1. In the "Project Manager" tree, right-click "Function block" and select "Block Properties". The "Block Properties" dialog box is displayed.



Block Properties

Watchdog: 1000 ms

☑ Constant scan period: 10 ms

2. Set "Watchdog" and "Constant scan period".

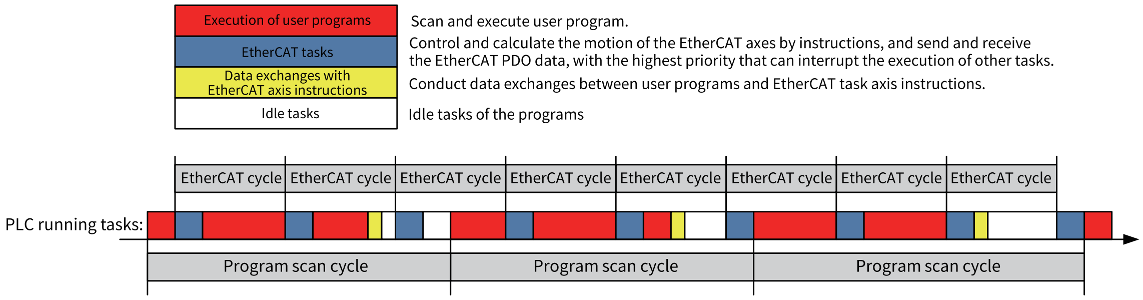
- ① Watchdog: indicates the maximum scan cycle of the program. When the running time of the program is greater than the value, the PLC reports an error and stops running.
- ② Constant scan period: When the check box is selected, the constant scan cycle mode is selected and the scan cycle equals to the value. Otherwise, the scan cycle equals to the actual running time.

3. Click "OK" and download the program to the PLC. The settings take effect immediately.

2.8 Setting EtherCAT Task Cycles

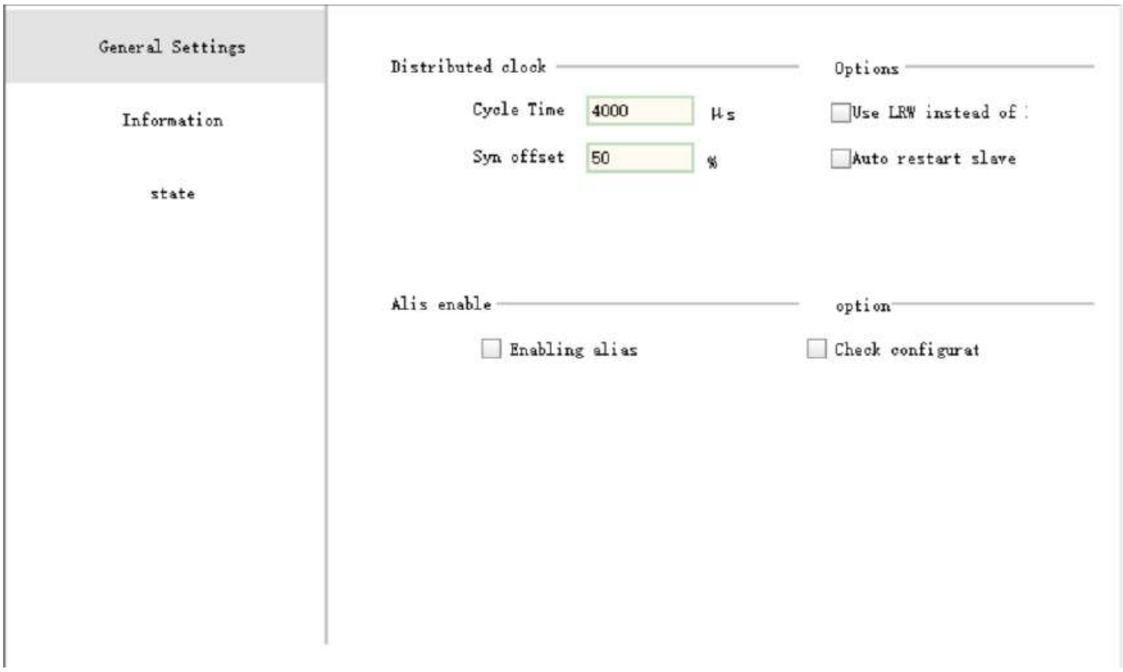
You can set EtherCAT task cycles as needed.
One program scan cycle contains the program execution, EtherCAT task, and EtherCAT axis instruction data exchange. When program scanning is idle, the scan cycle also contains the idle task. The priority of the EtherCAT task is the highest so that the EtherCAT task can interrupt execution of other tasks.

The following figure shows the relationship between the EtherCAT task cycle and the program scan cycle.



🟥 Execution of user programs	Scan and execute user program.
🟦 EtherCAT tasks	Control and calculate the motion of the EtherCAT axes by instructions, and send and receive the EtherCAT PDO data, with the highest priority that can interrupt the execution of other tasks.
🟨 Data exchanges with EtherCAT axis instructions	Conduct data exchanges between user programs and EtherCAT task axis instructions.
□ Idle tasks	Idle tasks of the programs

1. In the "Project Manager" tree, unfold "Config" and double-click "EtherCAT". The "General Settings" page is displayed.



Distributed clock

Cycle Time 4000 μs

Syn offset 50 %

Options

□Use LRW instead of

□Auto restart slave

Alis enable

□ Enabling alias

option

□ Check configurat

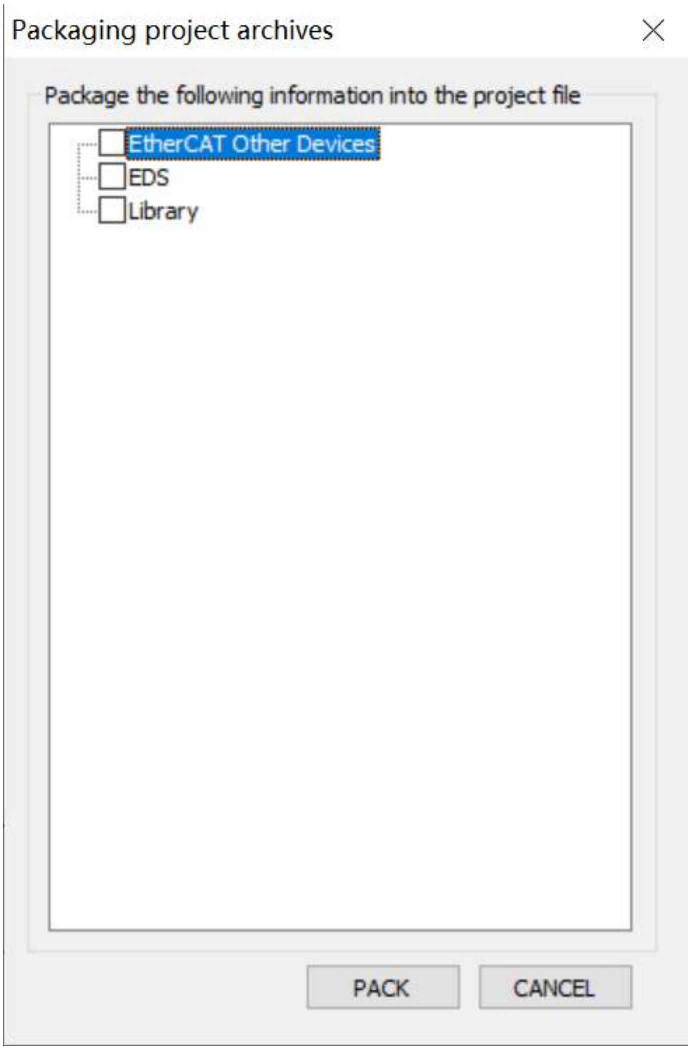
2. Set the cycle time of the EtherCAT task to a value ranging from 1000 μs to 9000 μs.
3. (Optional) In monitoring mode, click the "Status" tab. On the "EtherCAT task monitoring" page that is displayed, obtain communication information.
4. Click "OK" and download the program to the PLC. The settings take effect immediately.

2.9 Packing and Decompressing Project Archives

A project archive is an .hclib file consisting of the current project, EDS file, library file, and third-party EtherCAT device XML file. The current project cannot be a temporary project, and the library file used in the program is packed by default. To provide your project for other users, you only need to pack the project archive into an .hclib file and send it to the target user. The user then decompresses the file and directly compiles and downloads the project archive, without the need of sending the EDS file, library file, or third-party XML file.

Packing a project archive

1. In the menu bar, choose "File" > "Pack Project Archives". The "Packaging project archives" dialog box is displayed.



Packaging project archives

Package the following information into the project file

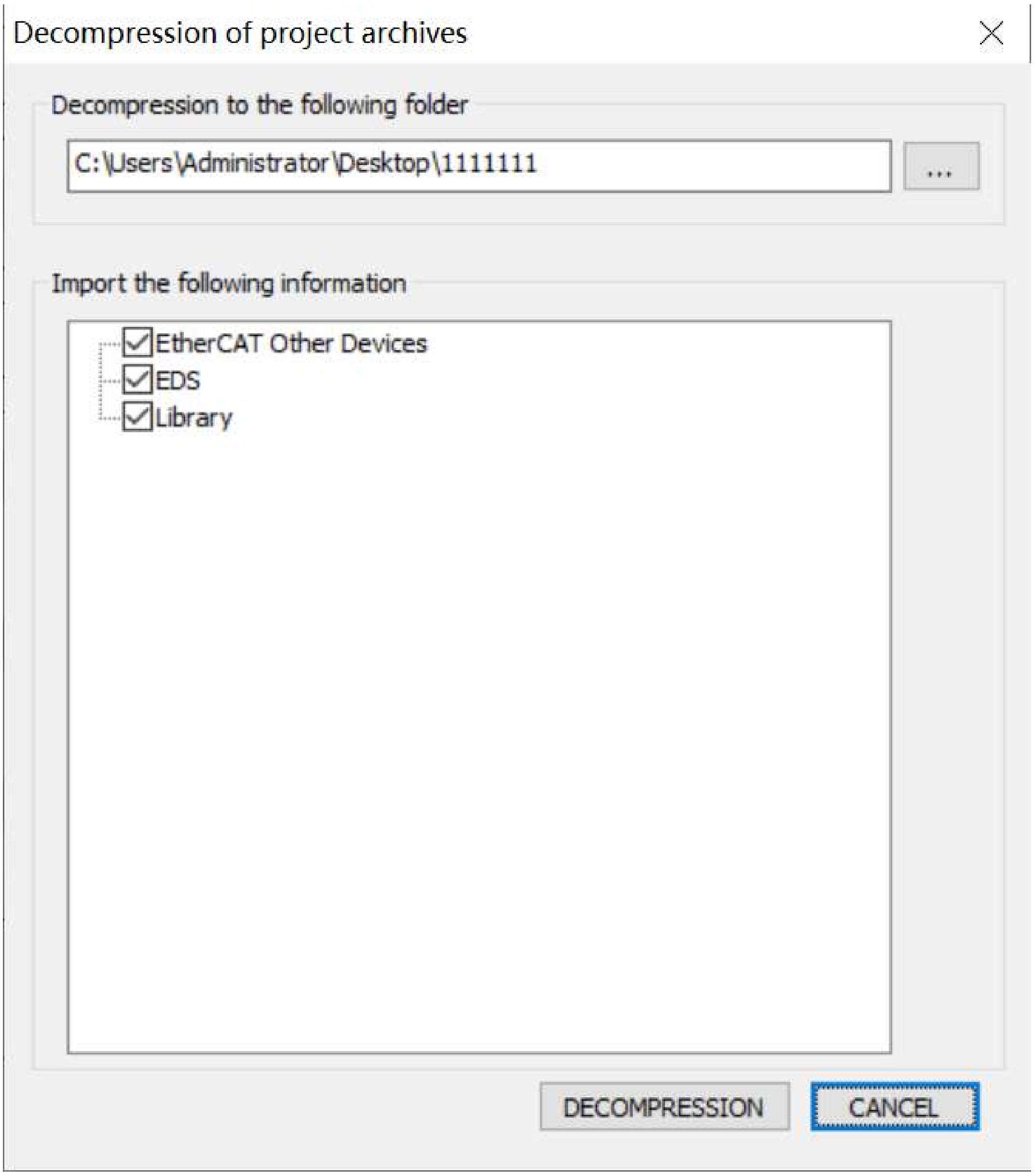
- □ EtherCAT Other Devices
- □ EDS
- □ Library

2. Select the files to be packed (library files used in the program are automatically selected), and click "PACK" to save the .hclib file.

Decompressing a project archive

Prerequisites: An .hclib file is prepared.

1. In the menu bar, choose "File" > "Decompress Project Archives". The "Open" dialog box is displayed.
2. Select and open the .hclib file to be decompressed. The "Decompression of project archives" dialog box is displayed.


Decompression of project archives

Decompression to the following folder

C:\Users\Administrator\Desktop\1111111

Import the following information

☑EtherCAT Other Devices

☑EDS

☑Library

3. Set "Decompression to the following folder".
   By default, the project archive is decompressed into the folder with the project archive name in the folder where the .hclib file is located. If the storage folder does not exist, it is automatically created. You can also click "..." to select a target folder for decompression.

4. Select required files (all selected y default), click "DECOMPRESSION", and import the third-party EtherCAT device XML file, EDS file, and library file.

2.10 Logging in to PLC

2.10.1 Overview

⚠ Caution

Only PLCs with a login password set need to use this function.

The login function protects the intellectual property of customers and prevents the PLC program from unauthorized modification.

The following table lists the number of login users, login password, and login permissions.

Item		Description
Number of login users		Only one user can log in to the PLC at a time. Other users can log in only after the user logs out. Otherwise, a login failure is prompted.
Login password		A login password contains up to eight characters in ANSI C mode.
Login permissions	Upload and download[1]	A permission password is required.
	Modifying the program online	A permission password is required.
	RTC	A permission password is required.
	Changing the IP address	A permission password is required.
	Clearing the PLC program storage space	A permission password is required.
	Verifying the program	A permission password is required.

Note

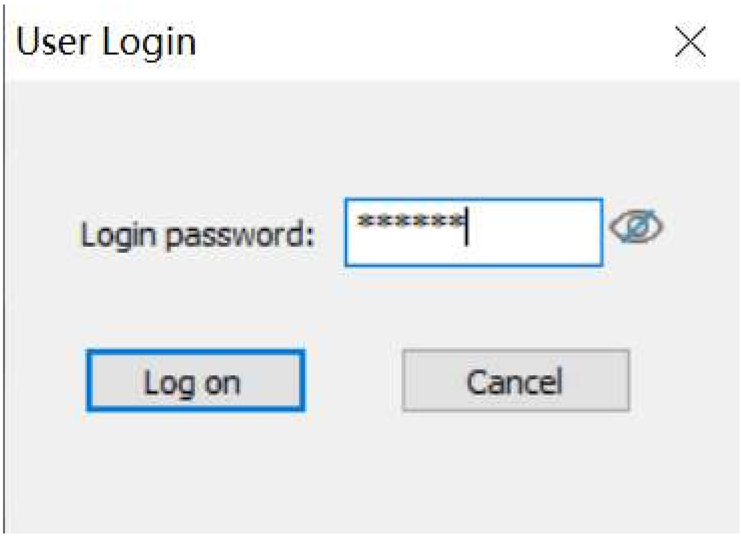
[1]: Upload and download refer to the upload and download of user programs and configuration data through the AutoShop programming software.

2.10.2 Logging In to and Logging Out of PLC

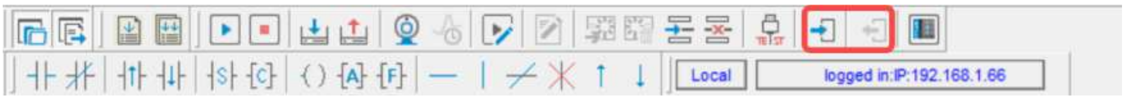
Logging in to PLC

Prerequisites: The PC is properly connected to the PLC, and the PLC is online.

1. In the toolbar, click ➡. The "User Login" page is displayed.



2. Enter the PLC login password in the "Login password" text box and click "Log on" to log in to the PLC.
   After successful login, the login button turns gray and the logout button turns red.



Note

After a user logs in to the PLC, other users attempting to log in to the PLC will receive a pop-up prompt saying that the PLC is being used and rejects the login request. Only after the current user logs out can other users log in to the PLC.

Logging out of PLC

Prerequisites: The PC is properly connected to the PLC, and the PLC runs properly.

In the toolbar, click ⇦ to log out of the PLC.

After successful logout, the login button turns blue and the logout button turns gray.

2.10.3 Managing Login Password

The password used to log in to the PLC can be set, changed, or deleted.

Note

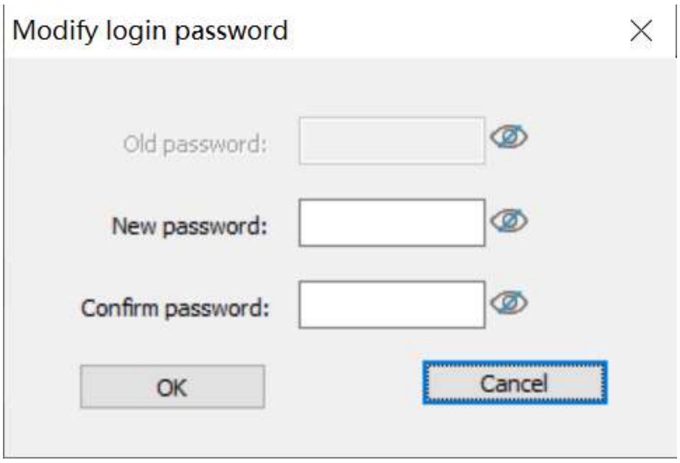
1. The password is saved in memory. When you re-open the project and re-connect to the PLC, you need to log in again.
2. When generating a Down file, you can opt for the login password. If no password is set, you do not need to enter a password.
3. A password can only contain some characters in the ANSI character set, including uppercase and lowercase letters, numbers, and some special characters such as parenthesis (()), comma (,), exclamatory mark (!), and atmark (@).
4. When you opt for the login password, some functions, such as IP address modification, PLC scan, and login password modification, will exit the login status. After opting for the login password, you need to log in again.

Setting PLC login password

When no login password is set for the PLC, you can set a PLC login password.

Prerequisites: The PC is properly connected to the PLC, and the PLC is online.

1. In the menu bar, choose "PLC" > "Set/Modify Login PLC Password". The "Modify login password" dialog box is displayed.



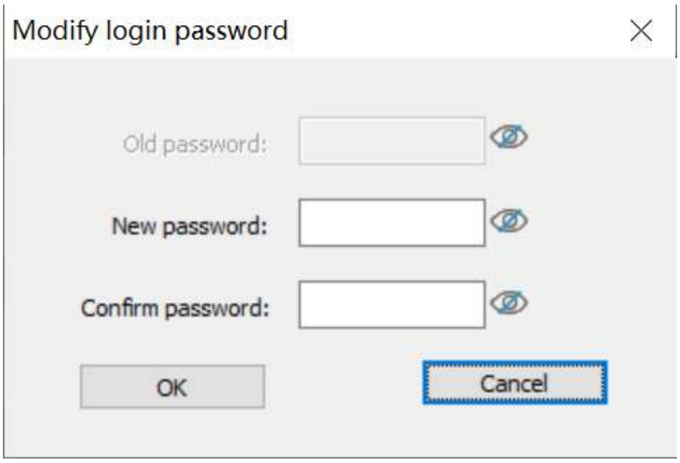
2. Enter the new password in the "New password" and "Confirm password" text boxes, and click "OK" to set the PLC login password.

Changing PLC login password

When a login password is set for the PLC, you can change the PLC login password.

Prerequisites: The PC is properly connected to the PLC, and the PLC is online.

1. In the menu bar, choose "PLC" > "Set/Modify Login PLC Password". The "Modify login password" dialog box is displayed.



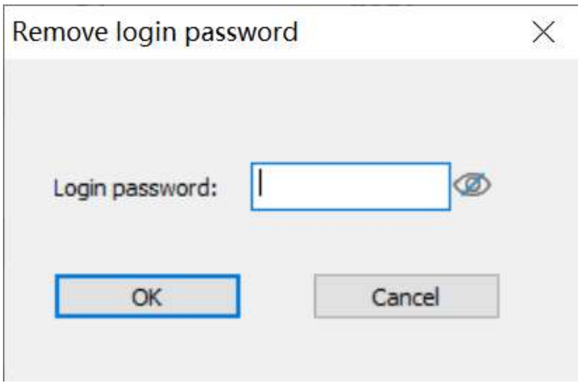
2. Enter the old password in the "Old password" text box and the new password in the "New password" and "Confirm password" text boxes, and click "OK" to change the PLC login password.

Deleting PLC login password

When a login password is set for the PLC, you can delete the PLC login password.

Prerequisites: The PC is properly connected to the PLC, and the PLC is online.

1. In the menu bar, choose "PLC" > "Delete Login PLC Password". The "Remove login password" dialog box is displayed.



2. Enter the PLC login password in the "Login password" text box and click "OK" to delete the PLC login password.

2.11 Trace Monitor Variables

2.11.1 Overview

Like a digital sampling oscilloscope, the Trace function can record historical values of variables. When the Trace function is enabled, AutoShop starts to save the data records containing time, and you can continuously monitor variable changes on the Trace page to facilitate program commissioning.

2.11.2 Adding Trace Monitor Variables

This section introduces how to add Trace monitor variables by capturing changes of the D100, D200, and TEST variables.

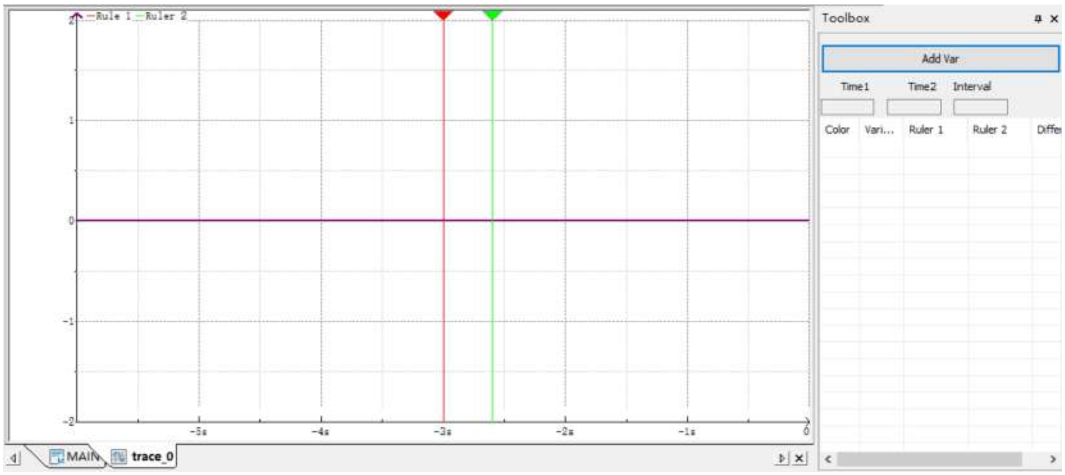
Procedure

1. In the "Project Manager" tree, right-click "Trace" and select "New" to create a Trace monitor view.

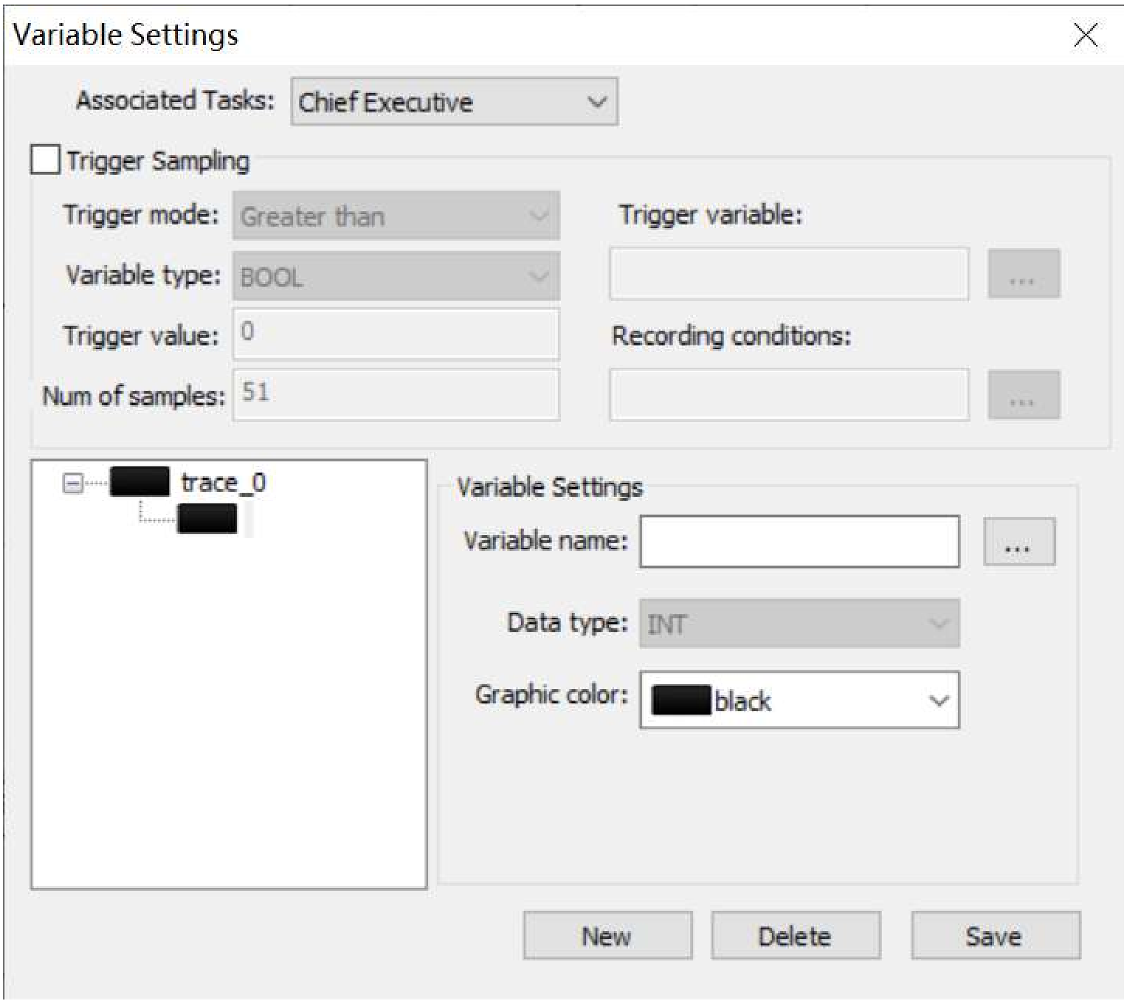
Note

Right-click the newly created monitor chart and select "Delete" to delete the monitor chart, or select "Rename" to rename the monitor chart.

2. Double-click the new Trace monitor view to access the Trace page.



3. Click "Add Var". The "Add Var" dialog box is displayed.



Variable Settings

Associated Tasks: Chief Executive

□ Trigger Sampling

Trigger mode: Greater than

Variable type: BOOL

Trigger value: 0

Num of samples: 51

Trigger variable: ...

Recording conditions: ...

[redacted] trace_0

[redacted]

Variable Settings

Variable name: ...

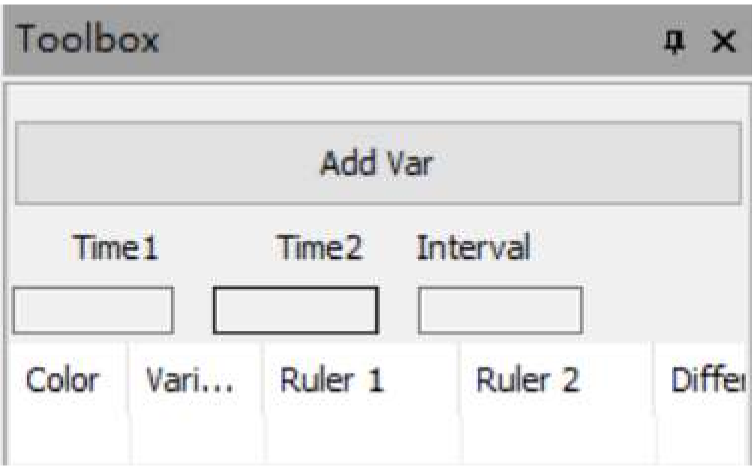
Data type: INT

Graphic color: ■ black

Parameter Name	Description
Associated Tasks	Indicates a variable used to select a monitor for sampling in main tasks or EtherCAT tasks. Select "Chief Executive" or "Ethercat Task" from the drop-down list.
Trigger Sampling Enables	Indicates the function switch to trigger sampling enabling, which is valid upon checked.

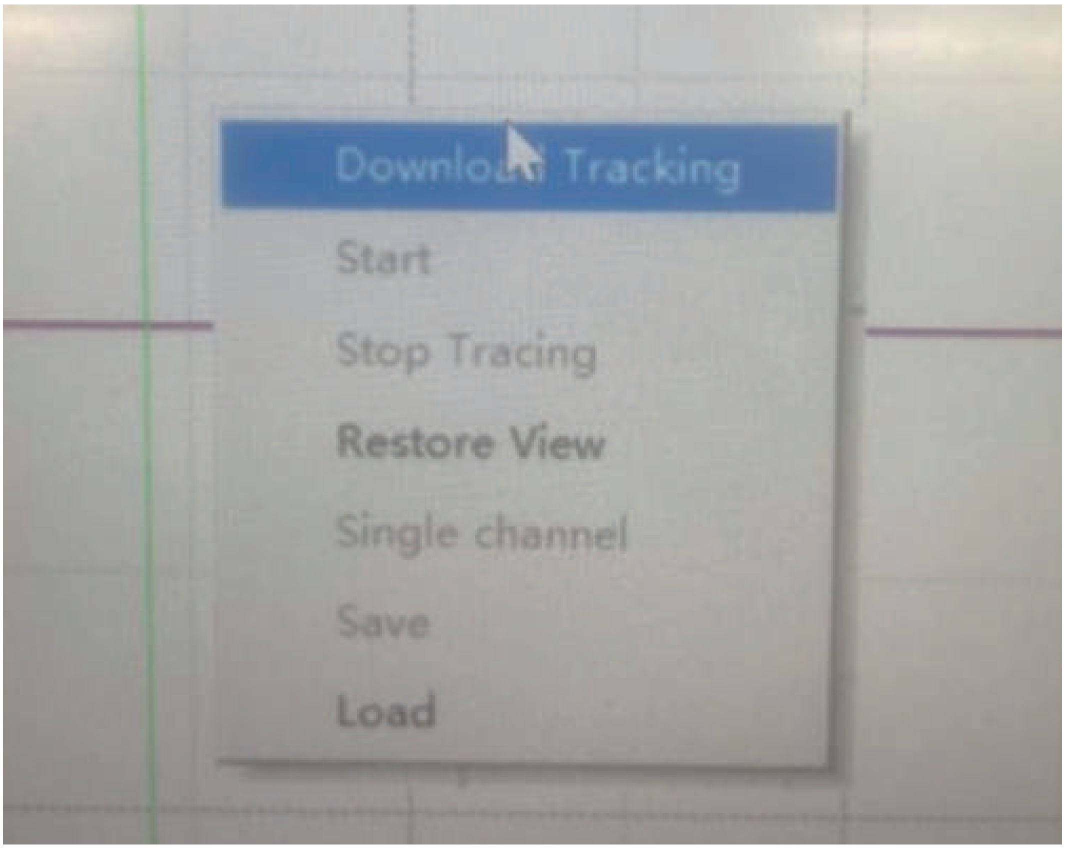
Parameter Name	Description
Trigger mode	Indicates the mode to trigger sampling. Select "Greater than", "Less than", "Not equal to", "Equal to", "Greater than or equal to", or "Less than or equal to" from the drop-down list.
Trigger value	Indicates that triggering stops when the trigger variable meets the trigger mode and trigger value conditions.
Variable type	Indicates the data type of a trigger variable. Select "BOOL", "INT", "DINT", or "REAL" from the drop-down list.
Number of samples	Indicates the number of sampling points before trigger stops when trigger conditions are met.
Trigger variable	Indicates a trigger object.
Recording conditions	Indicates a BOOL variable of recording conditions to start tracking. This parameter can be left empty.
Variable Settings	Indicates the variable name of the monitor, corresponding variable data type, and graphic color on the Trace monitor view. **Note:** Up to six variables are supported.

4. Set "Associated Tasks" to "Chief Executive" and set the D100, D200, and TEST variables to be captured as an example. Set other parameters according to the preceding table, and click "Save" to save the variable settings.
   View variable information in the toolbox on the right.



Parameter Name	Description
Color	Indicates the legend color of a variable on the Trace view.
Variable Name	Indicates the name of a variable.
Time1	Indicates the value and time of the point corresponding to ruler 1.
Time2	Indicates the value and time of the point corresponding to ruler 2.
Difference	Indicates the value and time difference between the points corresponding to the two rulers.

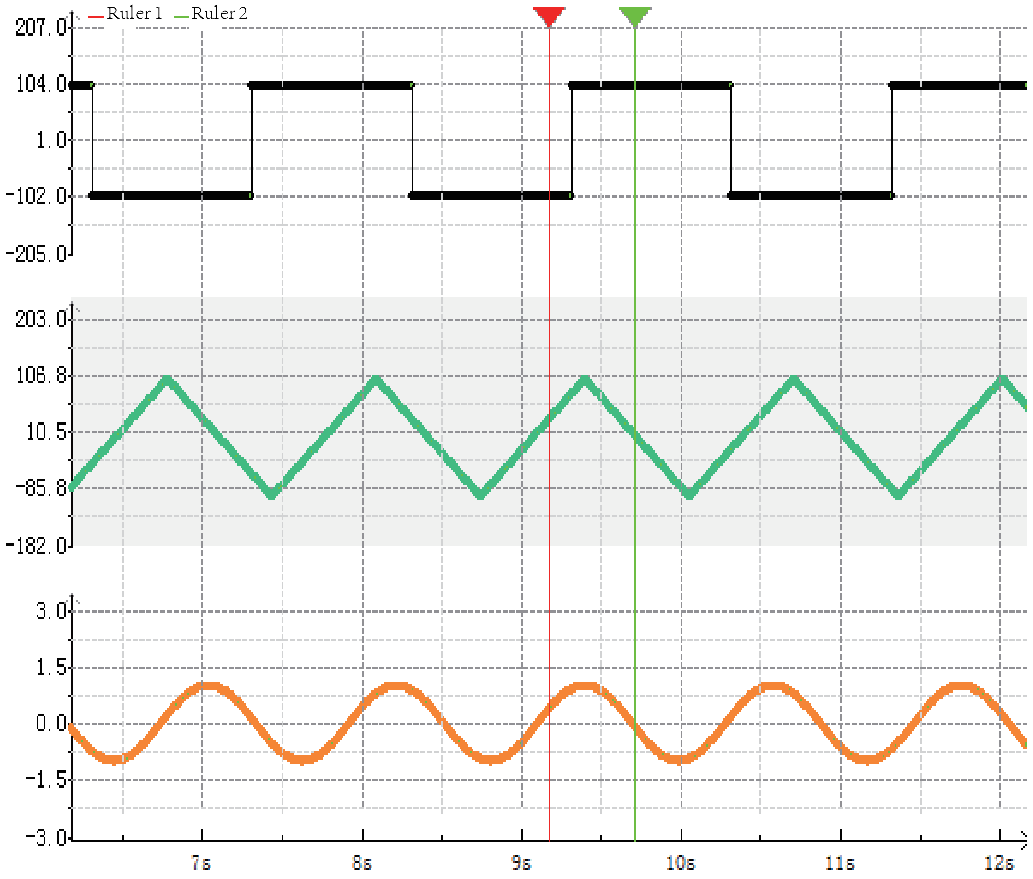
5. After configuring the variables, right-click on the Trace view and select "Download Tracking".



The following table lists the shortcut menus.

Parameter Name	Description
Download Tracking	Indicates to download the set tracking variables and start tracking.
Start	Indicates to start the original variables to continue to track data acquisition after tracking variables are stopped.
Stop Tracing	Indicates to stop data acquisition after tracking starts.
Restore View	Indicates to restore the initial acquisition speed and data interval after you use the mouse roller to zoom in or out and translate the view.
Single channel/ Multichannel	Indicates variable collection for a single coordinate system or multiple coordinate systems.
Save	Indicates to save all collected data in the tracking stopped state.
Load	Indicates to load the saved file of collected data.

After tracking is downloaded, you can start to monitor variables.



Note

- In Trace mode, the horizontal and vertical coordinates of the chart automatically adapt to variable values. You can adjust the coordinates by scrolling the mouse wheel. Specifically, directly scroll the mouse wheel to zoom in or out the horizontal coordinate, and scroll the mouse wheel while holding down the Ctrl key to zoom in or out the vertical coordinate.
- In Monitor mode, you can drag the rulers to view the data of specific points, or drag the curve back and forth to view the data of key points. During this process, the Trace monitor chart stops scrolling, but data is still being collected. To restore to automatic scrolling mode, right-click the Trace monitor chart and select "Restore View".
- For other operations to the Trace monitor chart, see the preceding table of shortcut menus.

Trace monitor example

When M0 is set to TRUE, Trace sampling starts. When D0 is greater than value 2 of "Trigger value", sampling stops after 51 data entries are sampled. D0 and D1 are tracking variables.

2.11.3 Importing or Exporting Trace Data

Exporting Trace data

Prerequisite: Data acquisition is stopped, that is, tracking is stopped.

Right-click on the Trace monitor view and select "Save" to save data to the corresponding folder.

Importing Trace data

Prerequisite:

- A Trace data file is prepared.
- Data acquisition is stopped, that is, tracking is stopped.

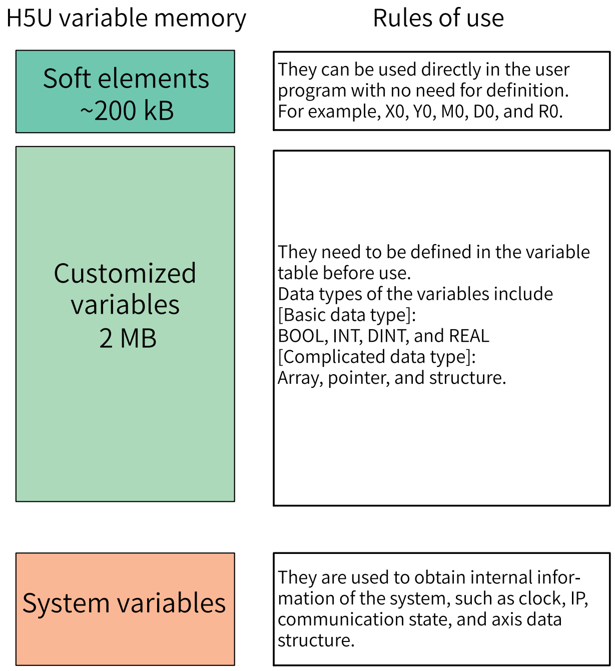
Right-click on the Trace monitor view and select "Load". In the dialog box that is displayed, select a Trace data file to load data to the Trace monitor view.

3 Programming Basics

3.1 Overview

The variable memory structure for programming contains soft elements, customized variables, and system variables.

The following figure shows the corresponding rules of use.



They can be used directly in the user program with no need for definition. For example, X0, Y0, M0, D0, and R0.

They need to be defined in the variable table before use.
Data types of the variables include
[Basic data type]:
BOOL, INT, DINT, and REAL
[Complicated data type]:
Array, pointer, and structure.

They are used to obtain internal information of the system, such as clock, IP, communication state, and axis data structure.

3.2 Elements

3.2.1 Bit Elements

The PLC supports bit elements. The following table describes the specific type, range, number of points, and description of bit elements.

Type	Range	Number of Points	Data Type	Description
X	X0 to X1777	1024 points, octal	BOOL	Input
Y	Y0 to Y1777	1024 points, octal	BOOL	Output
M	M0 to M7999	8000 points	BOOL	M0 to M999 not retained upon power failure, M1000 to M7999 retained upon power failure

Type	Range	Number of Points	Data Type	Description
S	S0 to S4095	4096 points	BOOL	S0 to S999 not retained upon power failure, S1000 to S4095 retained upon power failure
B	B0 to B32767	32768 points	BOOL	B0 to B999 not retained upon power failure, B1000 to B32767 retained upon power failure

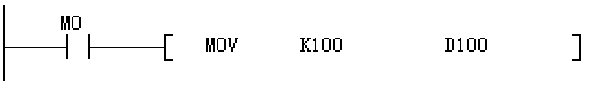
3.2.2 Word Elements

The PLC supports word elements. The following table describes the specific type, range, number of points, and description of word elements.

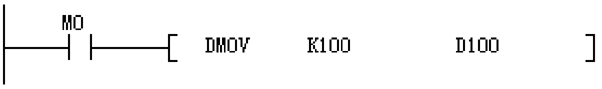
Type	Range	Number of Points	Data Type	Description
D	D0 to D7999	8000 points	BOOL/INT/DINT/REAL	D0 to D999 not retained upon power failure, D1000 to D7999 retained upon power failure
R	R0 to R32767	32768 points	BOOL/INT/DINT/REAL	R0 to R999 not retained upon power failure, R1000 to R32767 retained upon power failure
W	W0 to W32767	32768 points	BOOL/INT/DINT/REAL	W0 to W999 not retained upon power failure, W1000 to W32767 retained upon power failure

Example

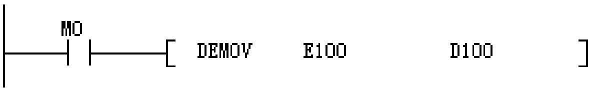
1. Word element used as a 16-bit integer
   Use the 16-bit assignment instruction to assign the value 100 to the word element D100, which occupies D100.



2. Word element used as a 32-bit integer
   Use the 32-bit assignment instruction to assign the value 100 to the word element D100, which occupies occupy D100 (low-order) and D101 (high-order).



3. Word element used as a floating-point number
   Use the floating-point instruction to assign the value 100 to the word element D100, which occupies D100 and D101.

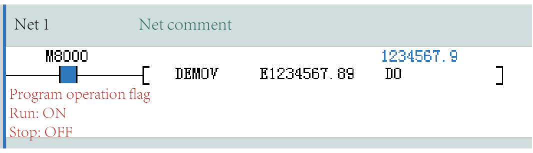


3.2.3 Special Elements

The PLC supports special elements. The following table describes the specific type, range, and description of special elements.

Type	Function	Range	Number of Points	Description
SBR	Subprogram label	SBR0 to SBR1023	1024	Used by the CALL instruction. Subprograms can be set as common subprograms or encrypted subprograms, which share the capacity of the system program area.
L	Jump label	L0 to L1023	1024 points	Used in combination with the CJ and LBL instructions
I	External interrupt	-	4	Interrupt label, X port rising edge, falling edge, rising and falling edge
	Timer interrupt	-	4	Timing duration (ms)
	Compare interrupt	-	16	Limited by the number of internal encoder axes (high-speed counters)
K	Decimal	K-32,768 to K32,767 (16-bit), K-2,147,483,648 to K2,147,483,647 (32-bit)	-	-
H	Hexadecimal	H0000 to HFFFF (16-bit), H00000000 to HFFFFFFFF (32-bit)	-	-
E	Floating-point number, real number	−3.402823e^{+38} to −1.175495e^{-38}, 0, +1.175495e^{-38} to +3.402823e^{+38}	-	Up to 7 decimal significant digits for a single-precision floating-point number (the excess will be automatically rounded off)
Character	Character, string	-	-	Used as instruction parameters

A single-precision floating-point number has a maximum of 7 significant decimal digits. If the 9-bit binary floating-point number 1234567.89 is transferred to the destination location D0, the actual value of D0 is 1234567.9. The precision is reduced.



Net 1 Net comment

M8000 [DEMOV E1234567.89 D0]

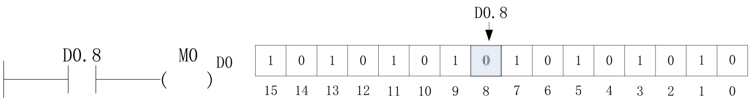
1234567.9

Program operation flag
Run: ON
Stop: OFF

3.2.4 Bit-based Operation on Word Elements

Bit-based operations on word elements can be implemented by using a dot (.). For example, writing D0.8 during programming indicates an operation on the 8th bit of the D0 word element.

Example:



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

The bits of the word element are counted from the 0th bit. When the 8th bit of D0 is 0, the output M0 is OFF; when the 8th bit of D0 is 1, the output M0 is ON.

3.3 Variables

3.3.1 Custom Variables

In a PLC programming system, in addition to using direct addresses, such as the X, Y, M, D, R and other elements, for programming, you can also use variables without specific storage addresses for programming to implement the required control logic, or the complete control process of the application object, so as to facilitate code compiling and improve the readability of the code.

Table 3–1 Supported custom variables

Type	Capacity	Data Type	Description
Pointer	4096 points (32-bit)	BOOL/INT/DINT/REAL	Pointer Variable Not retained upon power failure
BOOL	2 MB (8-bit)	BOOL/INT/DINT/REAL/IP/STRING/BYTE variable BOOL/INT/DINT/REAL/IP/STRING/BYTE array BOOL/INT/DINT/REAL/IP/STRING/BYTE compound structure	256 KB data retained upon power failure Other data not retained power failure
INT			
DINT			
REAL			
IP			
STRING			
BYTE			

3.3.2 Defining Variables

The PLC supports custom variables. You can define a global variable and directly use the variable name during programming. Abide by the following rules when naming a global variable:

- It contains only letters, digits, Chinese characters, and underscores (_) and does not start with a digit or underscore (_).
- It is not the same as the name of an element, constant, standard data type, instruction, subprogram, or interrupt subprogram.
- It cannot be keywords such as ARRAY, TRUE, FALSE, ON, OFF, and NULL.

Variable Data Types

Structures and arrays are supported. The following table lists the supported data types.

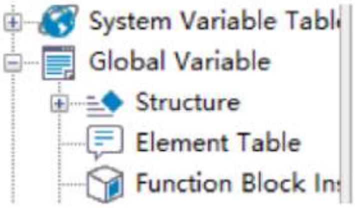
Table 3–2 Variable data types

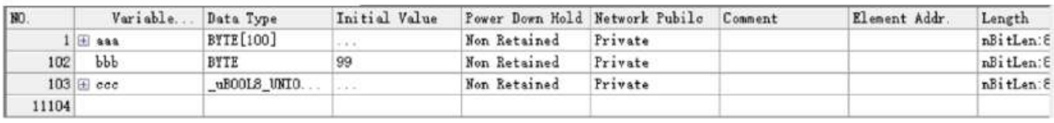
Data Type	Description
BOOL	Boolean
INT	Single-word integer

Data Type	Description
DINT	Double word integer
REAL	Real number
STRING	String type
IP	IP
BYTE	Byte

Defining Global Variables

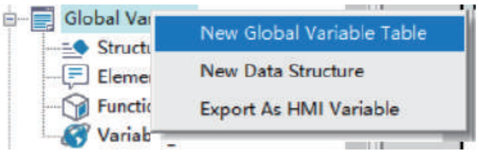
"Global Variable" in the project management window is used for variable management, allowing you to add, delete, and edit variables.





NO.	Variable...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	⊞ aaa	BYTE[100]	...	Non Retained	Private			nBitLen:8
102	bbb	BYTE	99	Non Retained	Private			nBitLen:8
103	⊞ ccc	_uBOOL8_UNIO...	...	Non Retained	Private			nBitLen:8
11104								

1. Add a variable table and variables. Right-click "Global Variable" and choose "New Global Variable Table" to create a global variable table.



2. Double-click the variable table to go to the variable editing interface.

- Edit a variable: Double-click the text box to edit or click the drop-down box to select.
- Add a variable: Right-click and choose "Insert Row(&I)".
- Delete a variable: Right-click the row to be deleted and choose "Delete Row(&L)".

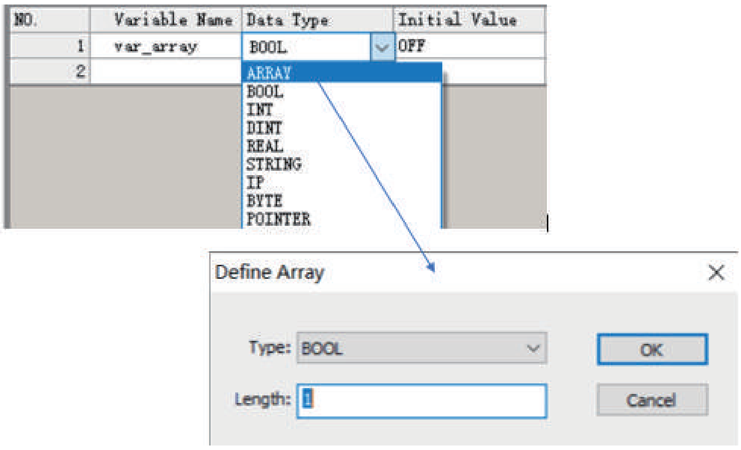
Parameter Name	Description
Variable Name	Custom variable name. You can directly use the variable name for programming.
Type	The data types include BOOL, INT, DINT, REAL, IP, STRING, and BYTE variables, BOOL, INT, DINT, REAL, IP, STRING, and BYTE arrays, and BOOL, INT, DINT, REAL, IP, STRING, and BYTE structures. If the data type is an array, you can set the type and length of the array variable in the displayed dialog box. If the data type is a pre-defined structure, you can define a structure variable.
Initial Value	You can assign an initial value to a variable. For arrays and structures, the initial value of each element can be specified individually.

Parameter Name	Description
Power Down Hold	"Power Down Hold" can be set to "Non Retained" or "Retained". The specified initial value is valid only when this parameter is set to "Non Retained".
Network Public	This parameter can be set to "Private", "Public", or "In/Out". For structure, specific union, structure array, and specific union array variables, this parameter must be set to "Private". When this parameter is set to "Public", a label configuration file named "LabelConfig.xml" will be generated in the "InteractiveFile" folder under the project directory after project compiling. Importing this configuration file into third-party software enables label communication.

3.3.3 Defining Arrays

During user programming, if the data type is set to "ARRAY", an array can be defined.

1. Select the type and length of the array variable in the displayed dialog box and click "OK" to define an array.



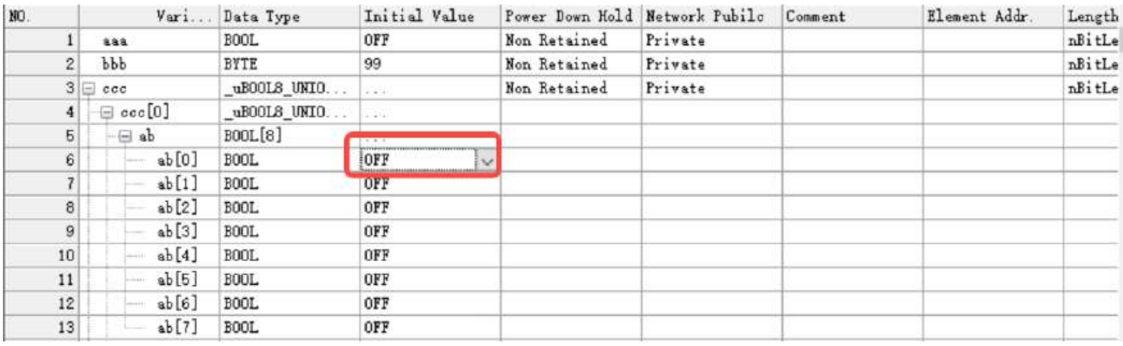
NO.	Variable Name	Data Type	Initial Value
1	var_array	BOOL	OFF
2			

Define Array

Type: BOOL

Length: 1

2. Click "+" next to the array variable to edit the initial values and comments of member variables.

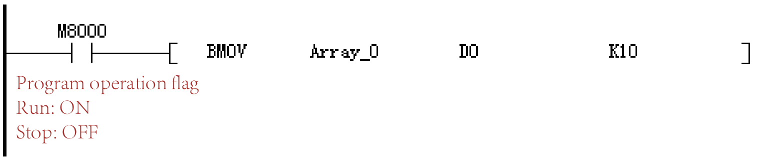


NO.	Vari...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	aaa	BOOL	OFF	Non Retained	Private			nBitLe
2	bbb	BYTE	99	Non Retained	Private			nBitLe
3	ccc	_uBOOL8_UNIO...	...	Non Retained	Private			nBitLe
4	ccc[0]	_uBOOL8_UNIO...	...					
5	ab	BOOL[8]	...					
6	ab[0]	BOOL	OFF					
7	ab[1]	BOOL	OFF					
8	ab[2]	BOOL	OFF					
9	ab[3]	BOOL	OFF					
10	ab[4]	BOOL	OFF					
11	ab[5]	BOOL	OFF					
12	ab[6]	BOOL	OFF					
13	ab[7]	BOOL	OFF					

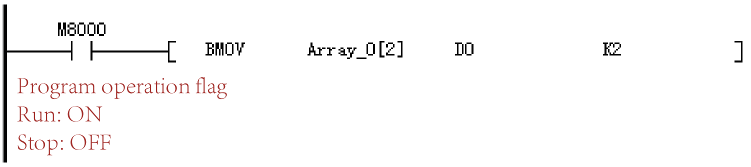
When an array is used in an instruction, if the array subscript is not specified, the access starts from the first element of the array. If the array subscript is specified, the access starts from the element specified by the subscript.

The following are two examples.

- Assign Array_0[0]–Array_0[9] to D0–D9.

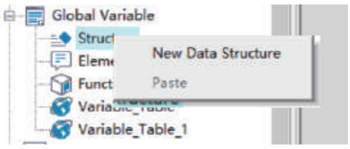


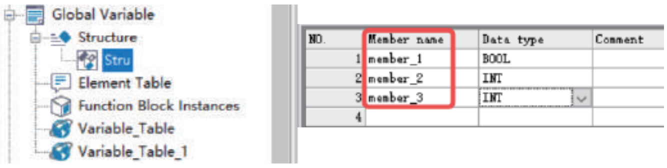
- Assign Array_0[2]–Array_0[3] to D0–D1.



3.3.4 Defining Structures

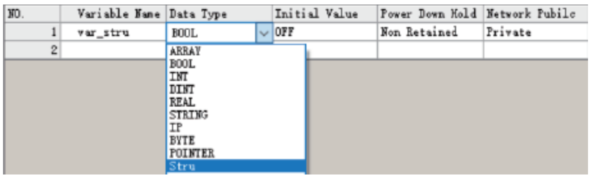
To define a structure variable, you need to define the data structure of the structure in advance. Right-click "Structure" under "Global Variable", choose "New Data Structure", and enter a structure name. The structure is defined. When defining a variable in the variable table, you can select this structure as the data type of the variable to define the variable as a structure variable.





NO.	Member name	Data type	Comment
1	member_1	BOOL	
2	member_2	INT	
3	member_3	INT	
4			

After the structure and member variables are created, you can select "Stru" in the "Data Type" column to define a structure variable.



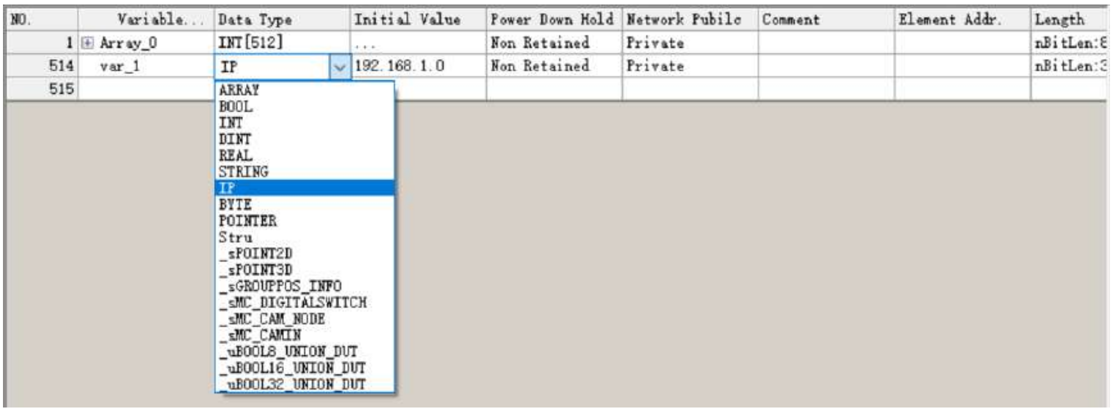
NO.	Variable Name	Data Type	Initial Value	Power Down Hold	Network Pubilc
1	var_stru	BOOL	OFF	Non Retained	Private
2					

Click the "Initial Value" column of the structure variable to set the initial values of structure variable members.

3.3.5 Defining IP Variables

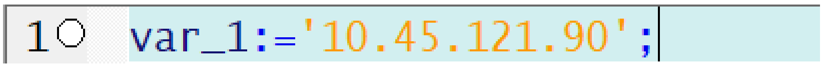
You can define IP variables in the variable table or program. An IP variable occupies 32 bits, and the default value is "192.168.1.0".

- Select "IP" from the "Type" drop-down list.



NO.	Variable...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	⊞ Array_0	INT[512]	...	Non Retained	Private			nBitLen:8
514	var_1	IP	192.168.1.0	Non Retained	Private			nBitLen:3
515								

- Use an IP variable in the ST program, and assign a value to the IP variable by using single quotation marks.

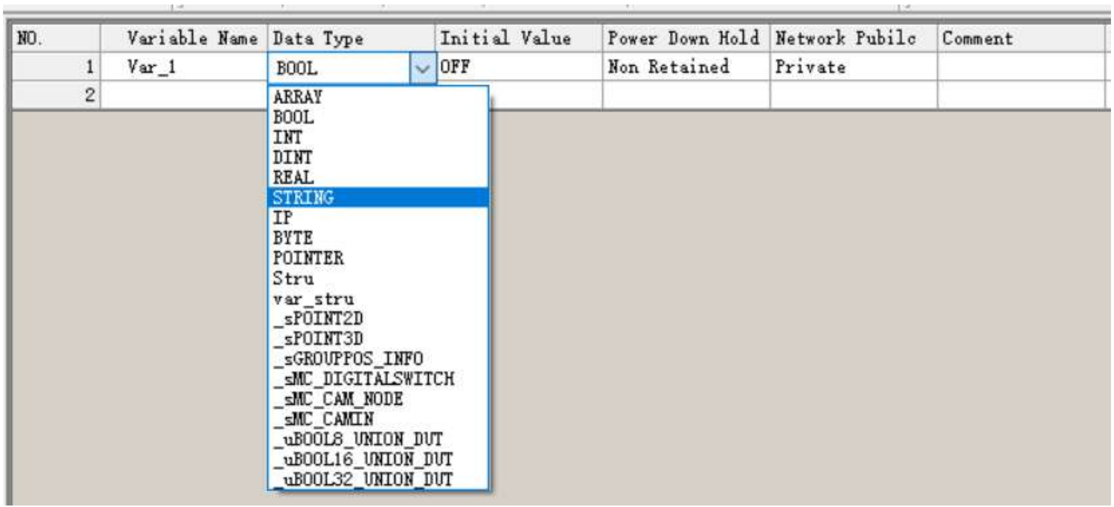


```
var_1:='10.45.121.90';
```

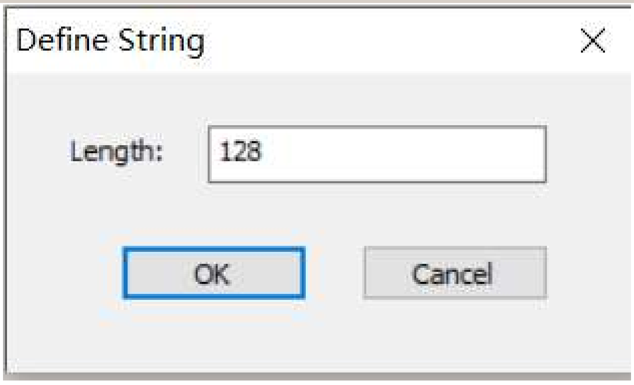
3.3.6 Defining Strings

You can define string variables in the variable table or program.

- Select "STRING" from the "Type" drop-down list of the variable table, and set the length of the string in the displayed dialog box.
  The default length is 128 bytes and the maximum length is 256 bytes. The last byte is the terminator by default.



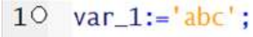
NO.	Variable Name	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment
1	Var_1	BOOL	OFF	Non Retained	Private	
2						



Define String

Length: 128

- Use a string variable in the ST program, and assign a value to the string variable by using single quotation marks.



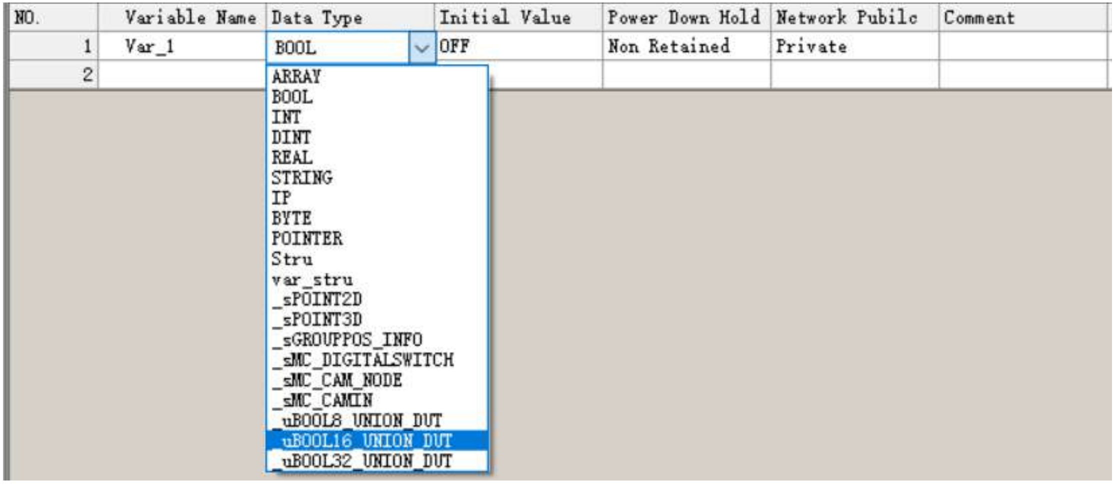
```
var_1:='abc';
```

3.3.7 Defining Specific Unions

A specific union is similar to a structure in that they both are collections of different types of elements. The difference lies in the fact that each member of a structure has its own independent storage space, while the members of a specific union share the same memory space (which is why a specific union is called a union). This will inevitably cause the members to overwrite each other, resulting in data loss. Therefore, the ideal application scenario for a specific union is when its members are not used simultaneously, but rather one after another.

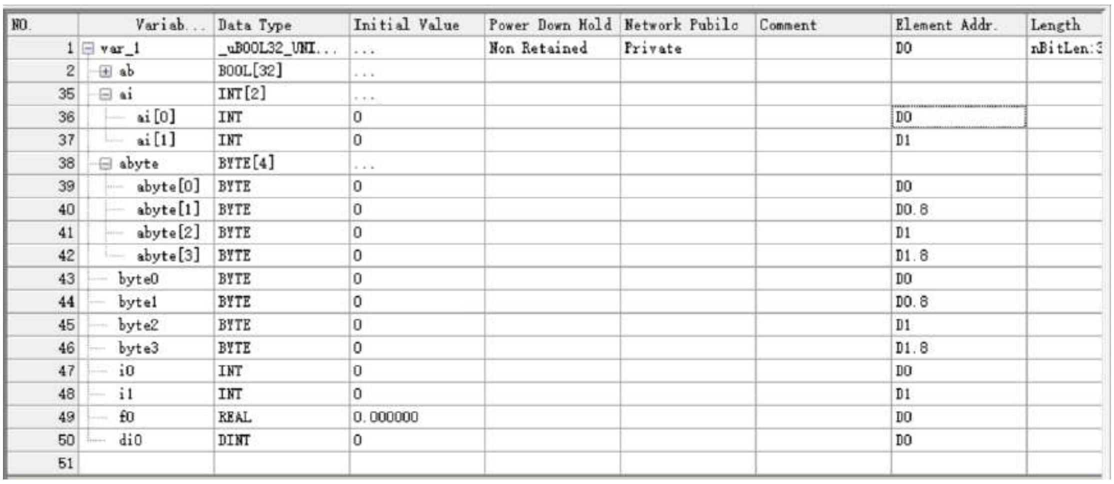
You can define specific union variables in the variable table or program. There are three types of specific union variables: _uBOOL8_UNION_DUT, _uBOOL16_UNION_DUT, and _uBOOL32_UNION_DUT, corresponding to lengths of 1 byte, 2 bytes, and 4 bytes, respectively.

- Select the required specific union variable type from the "Type" drop-down list of the variable table.



NO.	Variable Name	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment
1	Var_1	BOOL	OFF	Non Retained	Private	
2						

Take _uBOOL32_UNION_DUT as an example. Create a variable in the variable table, and select "_uBOOL32_UNION_DUT" from the "Type" drop-down list.



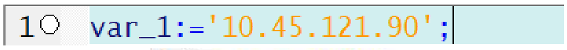
NO.	Variab...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	⊟ var_1	_uBOOL32_UNI...	...	Non Retained	Private		D0	nBitLen:3
2	⊞ ab	BOOL[32]	...					
35	⊟ ai	INT[2]	...					
36	ai[0]	INT	0				D0	
37	ai[1]	INT	0				D1	
38	⊟ abyte	BYTE[4]	...					
39	abyte[0]	BYTE	0				D0	
40	abyte[1]	BYTE	0				D0.8	
41	abyte[2]	BYTE	0				D1	
42	abyte[3]	BYTE	0				D1.8	
43	byte0	BYTE	0				D0	
44	byte1	BYTE	0				D0.8	
45	byte2	BYTE	0				D1	
46	byte3	BYTE	0				D1.8	
47	i0	INT	0				D0	
48	i1	INT	0				D1	
49	f0	REAL	0.000000				D0	
50	di0	DINT	0				D0	
51								

- In a program, you can access different members of a specific union variable by using the dot operator ("."). This allows you to parse variables in different scenarios.

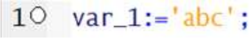
3.3.8 Using Variables

After a variable is defined, you can directly use the variable name for programming without assigning elements.

- When a common variable is used, directly use the variable name during programming.
- When an array variable is used, use "[Number]" to indicate an array element during programming. The number starts from 0.
- When a structure variable is used, use "Structure variable name.Member variable" to indicate a structure member during ST programming.
- When an IP or string variable is used, use a value enclosed in a pair of single quotation marks ('Value') to indicate the value of the variable.



```
var_1:='10.45.121.90';
```



```
var_1:='abc';
```

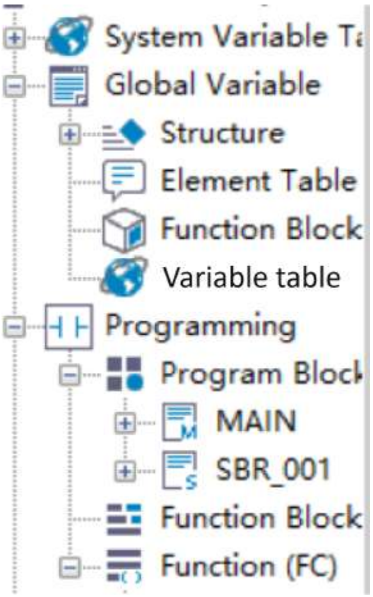
For BYTE, INT, and DINT variables and arrays, you can perform bit operations using the syntax "variable_name.bit_number" in programming. For details, see *"3.2.4 Bit-based Operation on Word Elements" on page 60*.

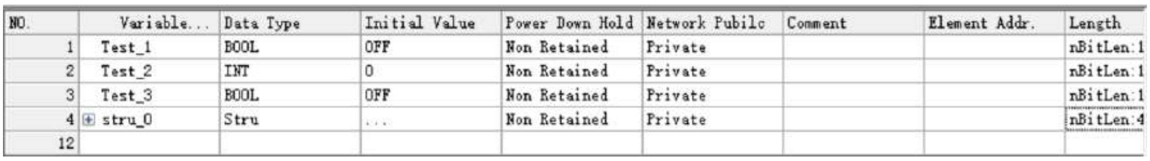
3.4 Binding Variables to Addresses

3.4.1 Overview

Customized variables can be bound to soft element addresses so that customized variable addresses are associated with soft element addresses.

You only need to enter the target address in the address column in the variable table and compile the project. Then the software will automatically generate an address for the customized variable.





NO.	Variable...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	Test_1	BOOL	OFF	Non Retained	Private			nBitLen:1
2	Test_2	INT	0	Non Retained	Private			nBitLen:1
3	Test_3	BOOL	OFF	Non Retained	Private			nBitLen:1
4	⊞ stru_0	Stru	...	Non Retained	Private			nBitLen:4
12								

3.4.2 Variable Property

A customized variable turns retentive or non-retentive at power failure according to the bound soft element.
As shown in the following figure, M1 is in the area of retention at power failure, so Test_1 is retentive at power failure after being bound to it; D100 is in the area of non-retention at power failure, so Test_2 is non-retentive at power failure after being bound to it.

The variable automatically turns retentive or non-retentive at power failure after being bound to an element.

3.4.3 Binding Basic Variables to Soft Elements

- A BOOL variable consists of one bit and can be bound only to a bit element. An INT variable consists of 16 bits and can be bound to one word element. A DINT or REAL variable consists of 32 bits and can be bound to two consecutive word elements.
- A BYTE variable consists of eight bits, and a word element consists of 16 bits. Therefore, a BYTE variable occupies only the low-order 8 bits of a word element upon binding.
- An IP variable consists of 32 bits, occupying two consecutive word elements.
- A STRING variable consists of a customized number of bytes, such as two bytes. For example, if the variable consists of five bytes, it occupies six bytes. After it is bound to soft element D0, it occupies D0, D1, and D2.

3.4.4 Binding Array Variables to Soft Elements

To bind an array variable to a soft element, enter the address to be mapped in the address column in the variable table.

- A word variable occupies a specified number of word elements based on the variable type. One INT variable occupies one 16-bit element, while a REAL or DINT variable or an IP variable occupies two 16-bit elements.
- Each member of a BYTE array variable occupies half a word element. For example, when a BYTE[5] variable is bound to D0, the first member is bound to the low-order 8 bits of D0 and the second member is bound to the high-order 8 bits of D0. Other elements are bound like this in sequence.
- Each member of a STRING array variable contains two bytes and occupies word elements in sequence.
- A BOOL variable occupies a specified number of bit elements.
- An array variable can be only bound to soft elements of corresponding types. That is, word variables can be only bound to word elements and bit variables can be only bound to bit elements.

For example, if a BOOL array variable is defined as Array_0 to be bound to the M0 element and the length is 10, the variable occupies elements M0 to M9. If an INT array variable is defined as Array_1 to be bound to the D0 element and the length is 10, the variable occupies elements D0 to D9.

3.4.5 Binding Structure Variables to Soft Elements

AutoShop earlier than V4.0.0.0 does not allow to bind the structure to soft elements. This section describes how to bind a project customized by AutoShop earlier than V4.0.0.0 to a variable after AutoShop is upgraded to V4.0.0.0, and how to bind a project customized by AutoShop 4.0.0.0 or later to a soft element.

When binding a structure variable to a soft element, enter a word element address to be mapped in the address column in the variable table, and click "Compile". AutoShop will automatically generate the address of the structure member. The address assignment rules are as follows:

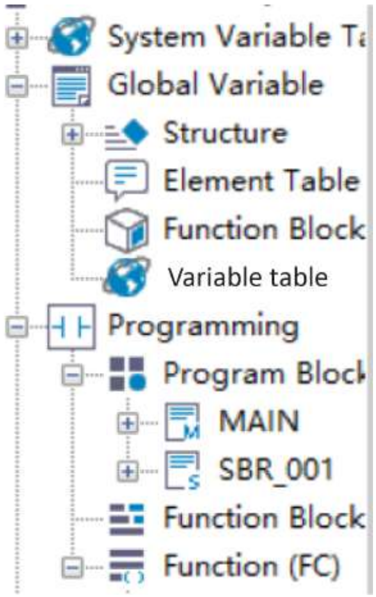
1. An INT variable occupies one 16-bit element, while a REAL or DINT variable occupies two 16-bit elements.
2. Multiple consecutive BOOL variables are considered to occupy one 16-bit element as a whole, and are assigned with an address in sequence from bit 0 of the 16-bit element. Multiple non-consecutive BOOL variables are considered to occupy one 16-bit element separately.
3. Arrays and structure variables occupy one 16-bit element as a whole.

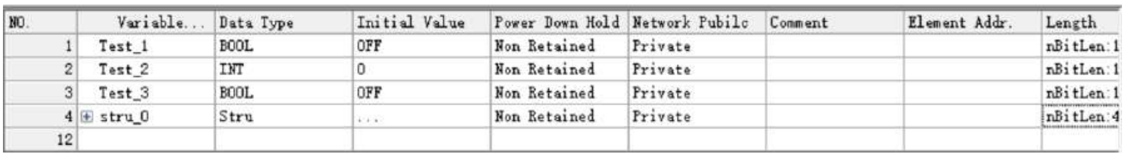
For example, the Stru_0 variable of the Stru type is defined and the D1000 element is bound.

No.	Member Variable Name	Data Type
1	member_1	BOOL
2	member_2	INT[2]
3	member_3	BOOL[2]
4	member_4	DINT
5	member_5	REAL
6	member_6	INT
7	member_7	BOOL

- Stru_0.member_1//The type is BOOL, so D1000.0 is bound.
- Stru_0.member_2//The type is INT array, so D1001 and D1002 are bound.
- Stru_0.member_3//The type is BOOL array, so D1003.0 and D1003.1 are bound.
- Stru_0.member_4//The type is DINT, so D1004 is bound.
- Stru_0.member_5//The type is REAL, so D1006 is bound.
- Stru_0.member_6//The type is INT, so D1008 is bound.
- Stru_0.member_7//The type is BOOL, so D1009.0 is bound.

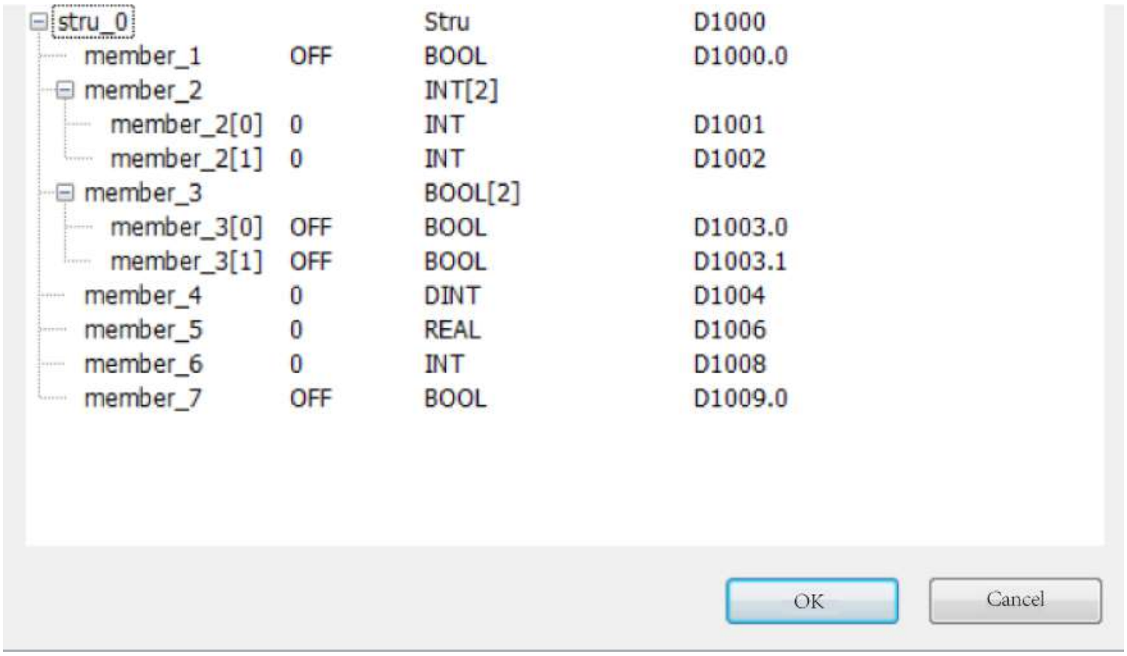
1. To bind a project customized by AutoShop 4.0.0.0 and later, enter the address to be mapped in the address column in the variable table.





NO.	Variable...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	Test_1	BOOL	OFF	Non Retained	Private			nBitLen:1
2	Test_2	INT	0	Non Retained	Private			nBitLen:1
3	Test_3	BOOL	OFF	Non Retained	Private			nBitLen:1
4	⊞ stru_0	Stru	...	Non Retained	Private			nBitLen:4
12								

Compile the project. An address will be automatically generated. Then, double-click the initial value of the corresponding structure variable in the variable table to view the mapping address of each member in the structure and set the variable input value.



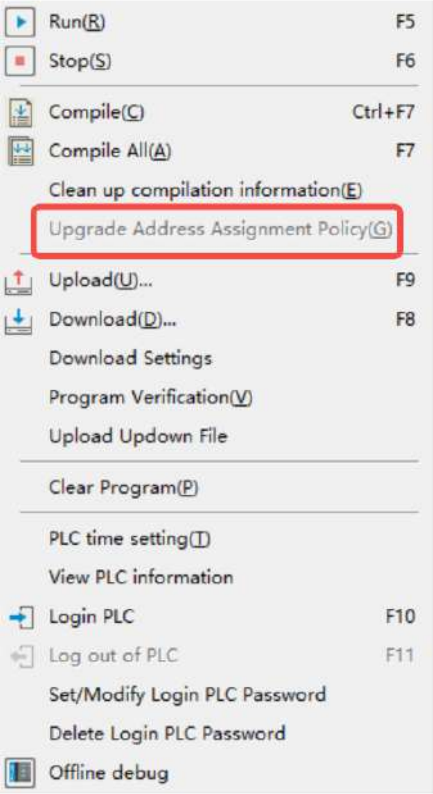
stru_0		Stru	D1000
member_1	OFF	BOOL	D1000.0
member_2		INT[2]	
member_2[0]	0	INT	D1001
member_2[1]	0	INT	D1002
member_3		BOOL[2]	
member_3[0]	OFF	BOOL	D1003.0
member_3[1]	OFF	BOOL	D1003.1
member_4	0	DINT	D1004
member_5	0	REAL	D1006
member_6	0	INT	D1008
member_7	OFF	BOOL	D1009.0

2. The address assignment policy for projects customized by AutoShop earlier than V4.0.0.0 is different. To bind a structure variable for such a project, upgrade the address assignment policy.

   The procedure is as follows: Open the project, switch to the variable table page, click "PLC" in the menu bar, and select "Upgrade Address Assignment Policy".

Note

The upgrade can result in change to the original addresses that are automatically assigned. If the monitoring variable table is used for communication with the HMI, the information of the monitoring variable table needs to be updated to the HMI. If the element binding method is used, the addresses will not be affected.

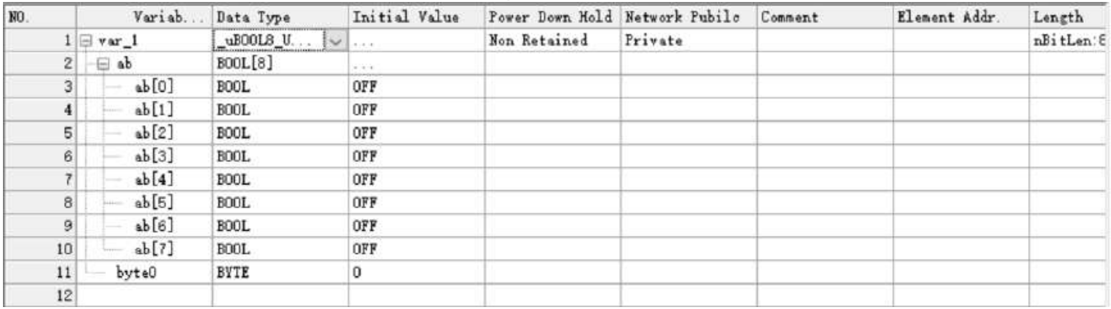


After the upgrade, the method of binding a structure variable to a soft element is the same as that later than AutoShop 4.0.0.0. For details, see the preceding procedure.

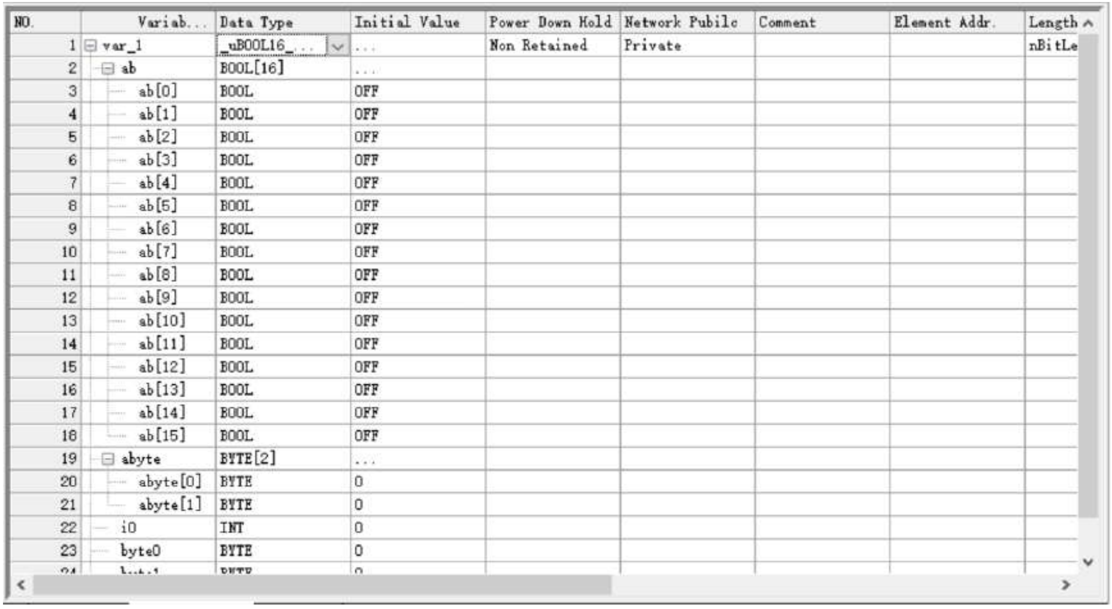
3.4.6 Binding Specific Union Variables to Soft Elements

Specific union variables are classified into three types, and specific union members obtain addresses at a fixed offset.

- A uBOOL8_UNION_DUT variable consists of eight bits and occupies the low-order 8 bits of a word element. Members of the variable are bound to soft elements in sequence based on the offset.
- A uBOOL16_UNION_DUT variable consists of 16 bits and occupies one word element. Members of the variable are bound to soft elements in sequence based on the offset.



NO.	Variab...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	⊟ var_1	_uBOOL8_U...	...	Non Retained	Private			nBitLen:8
2	⊟ ab	BOOL[8]	...					
3	ab[0]	BOOL	OFF					
4	ab[1]	BOOL	OFF					
5	ab[2]	BOOL	OFF					
6	ab[3]	BOOL	OFF					
7	ab[4]	BOOL	OFF					
8	ab[5]	BOOL	OFF					
9	ab[6]	BOOL	OFF					
10	ab[7]	BOOL	OFF					
11	byte0	BYTE	0					
12								



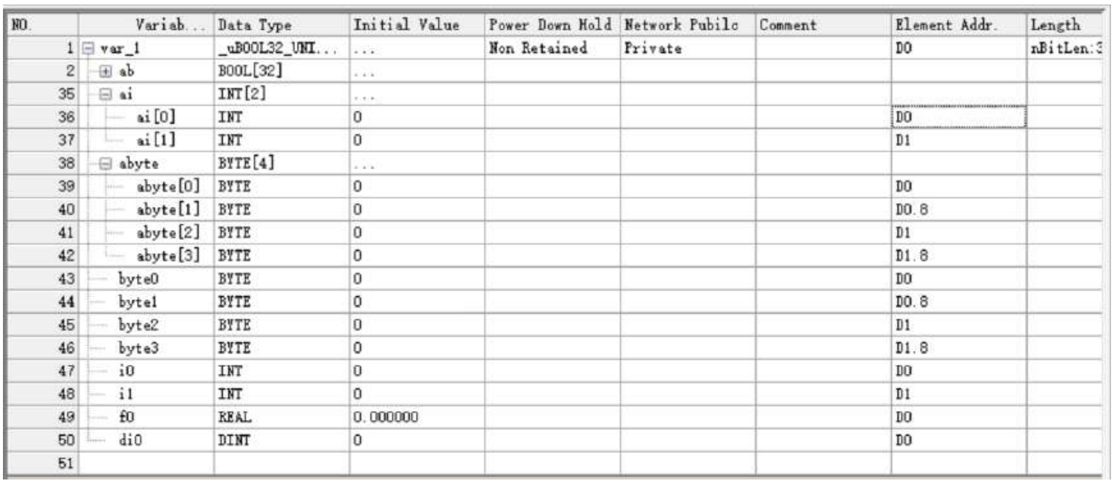
NO.	Variab...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	⊟ var_1	_uBOOL16_...	...	Non Retained	Private			nBitLe
2	⊟ ab	BOOL[16]	...					
3	ab[0]	BOOL	OFF					
4	ab[1]	BOOL	OFF					
5	ab[2]	BOOL	OFF					
6	ab[3]	BOOL	OFF					
7	ab[4]	BOOL	OFF					
8	ab[5]	BOOL	OFF					
9	ab[6]	BOOL	OFF					
10	ab[7]	BOOL	OFF					
11	ab[8]	BOOL	OFF					
12	ab[9]	BOOL	OFF					
13	ab[10]	BOOL	OFF					
14	ab[11]	BOOL	OFF					
15	ab[12]	BOOL	OFF					
16	ab[13]	BOOL	OFF					
17	ab[14]	BOOL	OFF					
18	ab[15]	BOOL	OFF					
19	⊟ abyte	BYTE[2]	...					
20	abyte[0]	BYTE	0					
21	abyte[1]	BYTE	0					
22	i0	INT	0					
23	byte0	BYTE	0					
24	byte1	BYTE	0					

- A uBOOL32_UNION_DUT variable consists of 32 bits and occupies two consecutive word elements. Members of the variable are bound to soft elements in sequence based on the offset. For example, the member var_1 is bound to soft element D0. The address offset of other BOOL array members is 0, so the initial address ab[0] is bound to D0.0, ab[15] to D0.15, and ab[31] to D1.15. The address offset of INT array members is 0, so the initial address ai[0] is bound to D0 and ai[1] to D1. The address offset of BYTE array members is 0, so the initial address abyte[0] is bound to D0, abyte[1] to D0.8, and so on.

  The offset is 0 for byte0, 8 bits for byte1, 16 bites for byte2, and 24 bits for byte 3. Therefore, the members are bound to D0, D0.8, D1, and D1.8 in sequence.

  The offset is 0 for i0 and 16 bits for i1. Therefore, the members are bound to D0 and D1 in sequence.

The offset is 0 for the floating-point number variable f0. Therefore, the variable is bound to the D0 and D1 elements. The offset is 0 for the DINT variable. Therefore, the variable is bound to the D0 and D1 elements.



NO.	Variab...	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	⊟ var_1	_uBOOL32_UNI...	...	Non Retained	Private		D0	nBitLen:3
2	⊞ ab	BOOL[32]	...					
35	⊟ ai	INT[2]	...					
36	ai[0]	INT	0				D0	
37	ai[1]	INT	0				D1	
38	⊟ abyte	BYTE[4]	...					
39	abyte[0]	BYTE	0				D0	
40	abyte[1]	BYTE	0				D0.8	
41	abyte[2]	BYTE	0				D1	
42	abyte[3]	BYTE	0				D1.8	
43	byte0	BYTE	0				D0	
44	byte1	BYTE	0				D0.8	
45	byte2	BYTE	0				D1	
46	byte3	BYTE	0				D1.8	
47	i0	INT	0				D0	
48	i1	INT	0				D1	
49	f0	REAL	0.000000				D0	
50	di0	DINT	0				D0	
51								

3.5 Using Variables as Array Subscripts

3.5.1 Rules of Use

Only one variable can be used as the subscript in the variable group.

The format is defined as array[index] or stru[index].var. In the format, array indicates an array or a structure array, index, var, and i are variables, and stru indicates the structure.

Basic combination types

- Array variables only support bit variable arrays, word variable arrays, doubleword variable arrays, and float point variable arrays, but not pointer variables.
- Index variables, as array subscript variables, support single-word INT variable (16-bit) and doubleword DINT variable (32-bit), but not soft elements or other variables such as bit variables, float point variables, or pointer variables. A specified element of an array or a specified member of the structure can be used as an index variable, such as array[index[5]] and array[stru.index]. An array element with a variable sub-index or an array member of the structure cannot be used as index variables, such as array[index[i]] and array[stru[i].index].

Complex combination types

- Array elements can be used as operands of instructions, with the index variable at the end, such as array[index], stru. Array[index], stru1[3].stru2. Array[index], and stru1.stru2.stru3. array[index].
- Members of structure arrays can be used as operands of instructions, with the index variable in the middle, such as stru[index].var, stru1[index].stru2.var, and stru1.stru2[5].stru3[index].array[3].
- Structure arrays containing two or more variables are not supported, such as stru[index1].array [index2].
- Two-dimension or multi-dimension arrays are not supported, such as array[index1][index2].

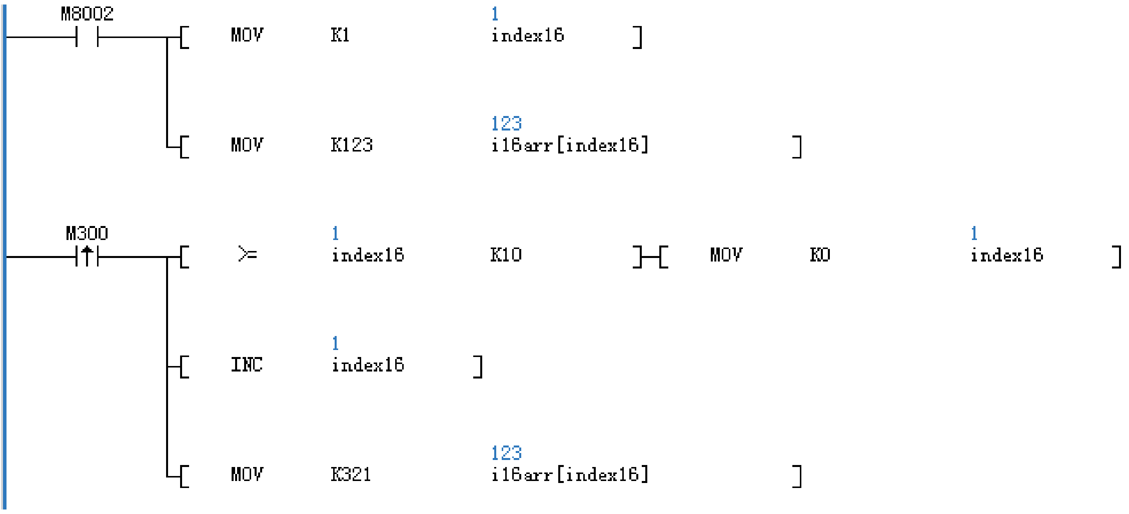
Note

- Operands of the ZSET/ZRST instructions do not support arrays that use variables as subscripts.
- Operands of the PTxxx instructions do not support arrays that use variables as subscripts.
- Operands of the SFC instructions do not support arrays that use variables as subscripts.
- For operands (array type operands) of instructions (such as the BMOV instruction for batch assignment) that use multiple consecutive variables, variables in the arr[index] format can be used, but elements of a structure array in the stru[index].var format must not be used (because they are not consecutive). If jump assignment is required, a loop instruction must be used to achieve batch jump assignment.
- This function is mainly used in operands of single-cycle instructions and is not recommended for operands of multi-cycle instructions. If it is necessary to use this function in operands of multi-cycle instructions, logic and timing must be strictly controlled. In case of poor timing control, abnormal execution or conflict may occur upon value switchover (such as the pulse output instruction axis).

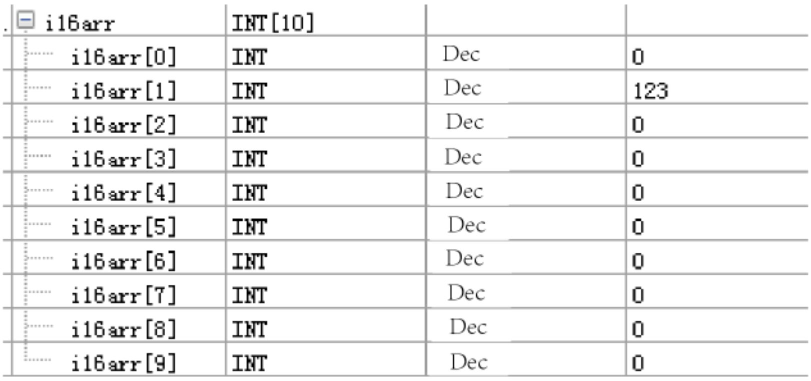
3.5.2 Programming Example

Example 1

The program for assigning a value for an element in an array is shown in the following figure.

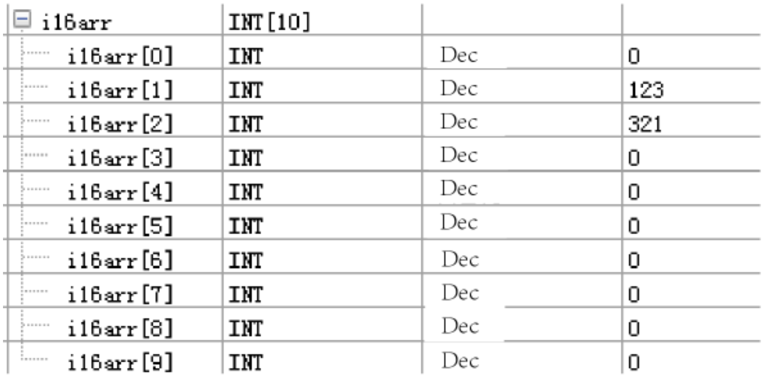


After startup, value 123 is assigned for i16arr[1]. Then, M300 assigns value 321 for the next array element upon each trigger. The following figure shows the result upon startup.



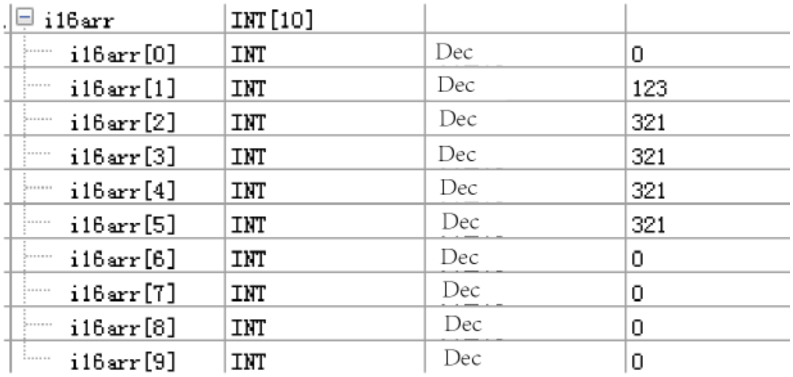
i16arr	INT[10]		
i16arr[0]	INT	Dec	0
i16arr[1]	INT	Dec	123
i16arr[2]	INT	Dec	0
i16arr[3]	INT	Dec	0
i16arr[4]	INT	Dec	0
i16arr[5]	INT	Dec	0
i16arr[6]	INT	Dec	0
i16arr[7]	INT	Dec	0
i16arr[8]	INT	Dec	0
i16arr[9]	INT	Dec	0

Result upon the first trigger by M300



⊟ i16arr	INT[10]		
i16arr[0]	INT	Dec	0
i16arr[1]	INT	Dec	123
i16arr[2]	INT	Dec	321
i16arr[3]	INT	Dec	0
i16arr[4]	INT	Dec	0
i16arr[5]	INT	Dec	0
i16arr[6]	INT	Dec	0
i16arr[7]	INT	Dec	0
i16arr[8]	INT	Dec	0
i16arr[9]	INT	Dec	0

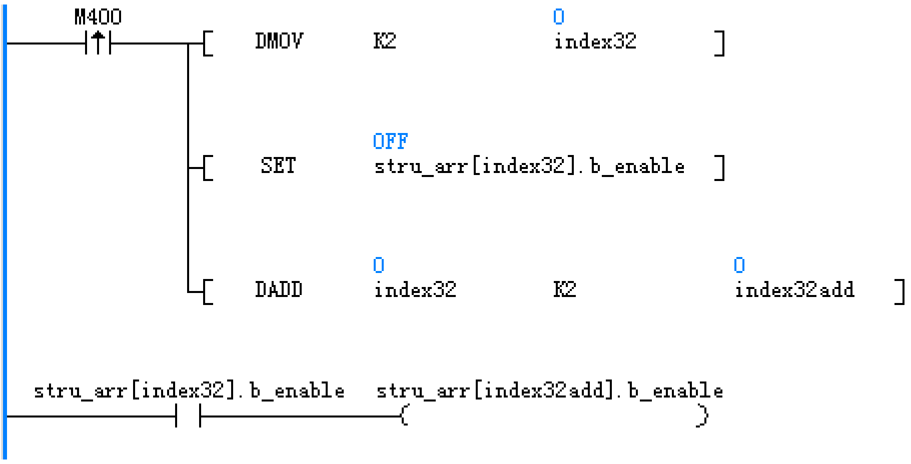
Result after multiple triggers by M300



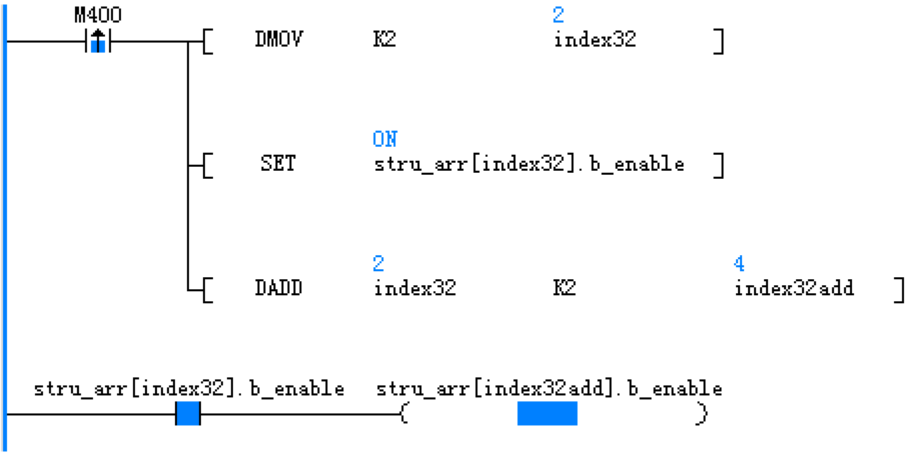
⊟ i16arr	INT[10]		
i16arr[0]	INT	Dec	0
i16arr[1]	INT	Dec	123
i16arr[2]	INT	Dec	321
i16arr[3]	INT	Dec	321
i16arr[4]	INT	Dec	321
i16arr[5]	INT	Dec	321
i16arr[6]	INT	Dec	0
i16arr[7]	INT	Dec	0
i16arr[8]	INT	Dec	0
i16arr[9]	INT	Dec	0

Example 2

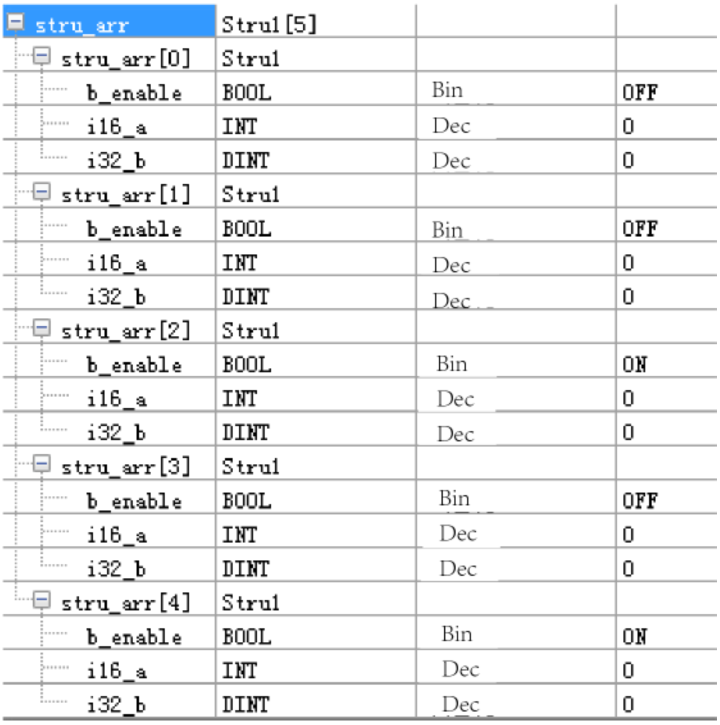
The program for operating a member variable of a structure array is shown in the following figure.


M400
DMOV
K2
0
index32
SET
OFF
stru_arr[index32].b_enable
DADD
0
index32
K2
0
index32add
stru_arr[index32].b_enable
stru_arr[index32add].b_enable

After a trigger by M400, stru_arr[2].b_enable is set and stru_arr[4].b_enable is controlled based on the status of stru_arr[2].b_enable.

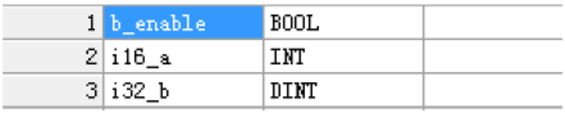


```
M400
-|↑|--+--[ DMOV   K2                          2
      |                                      index32             ]
      |
      |                   ON
      +--[ SET    stru_arr[index32].b_enable  ]
      |
      |           2                           4
      +--[ DADD   index32            K2       index32add         ]

stru_arr[index32].b_enable   stru_arr[index32add].b_enable
----| |-----------------------(                          )
```



stru_arr	Stru1[5]		
stru_arr[0]	Stru1		
b_enable	BOOL	Bin	OFF
i16_a	INT	Dec	0
i32_b	DINT	Dec	0
stru_arr[1]	Stru1		
b_enable	BOOL	Bin	OFF
i16_a	INT	Dec	0
i32_b	DINT	Dec	0
stru_arr[2]	Stru1		
b_enable	BOOL	Bin	ON
i16_a	INT	Dec	0
i32_b	DINT	Dec	0
stru_arr[3]	Stru1		
b_enable	BOOL	Bin	OFF
i16_a	INT	Dec	0
i32_b	DINT	Dec	0
stru_arr[4]	Stru1		
b_enable	BOOL	Bin	ON
i16_a	INT	Dec	0
i32_b	DINT	Dec	0

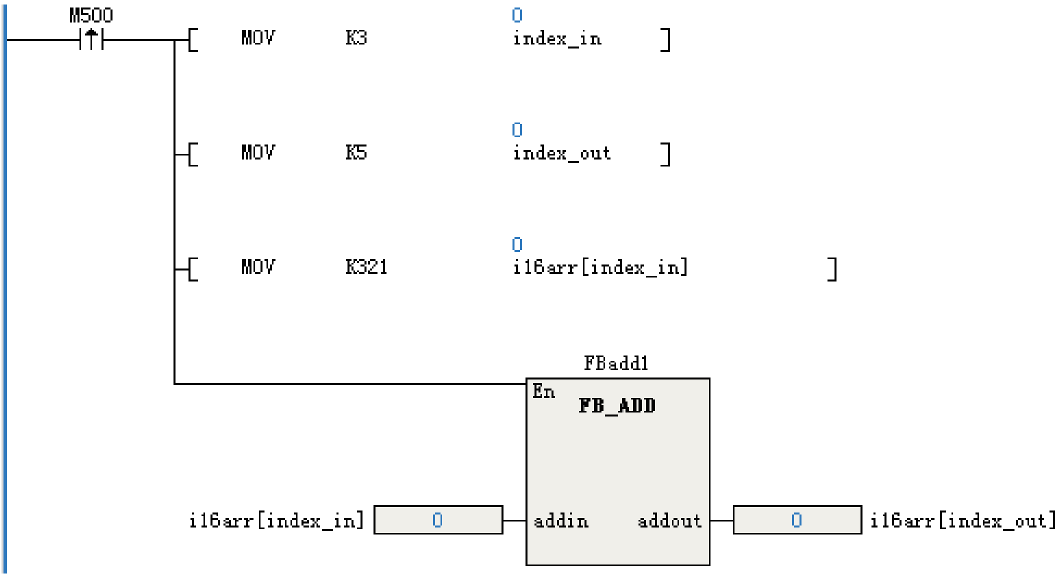
The following figure defines the structure.



1	b_enable	BOOL	
2	i16_a	INT	
3	i32_b	DINT	

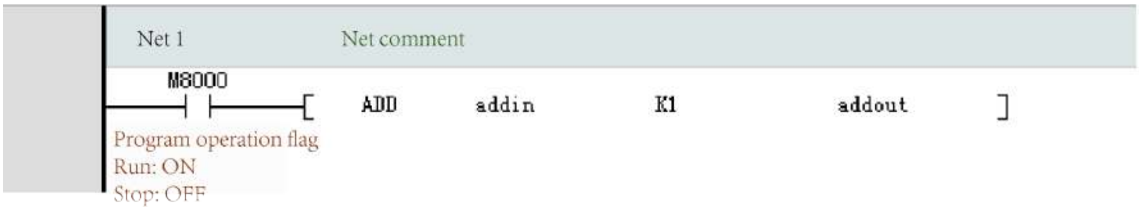
Example 3

The program for using a variable as the array subscript for FB parameters is shown in the following figure.



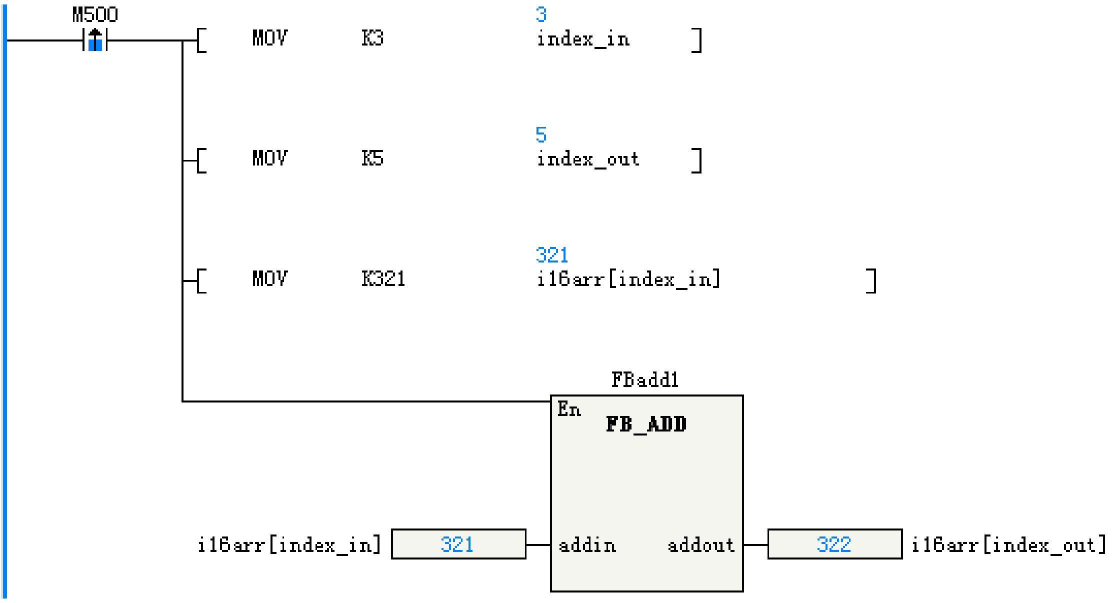
```
M500
-|↑|-- [ MOV  K3      0
                      index_in  ]
       [ MOV  K5      0
                      index_out ]
       [ MOV  K321    0
                      i16arr[index_in] ]
                      FBadd1
       ---------------En  FB_ADD
i16arr[index_in] 0 -- addin    addout -- 0 i16arr[index_out]
```

FB program

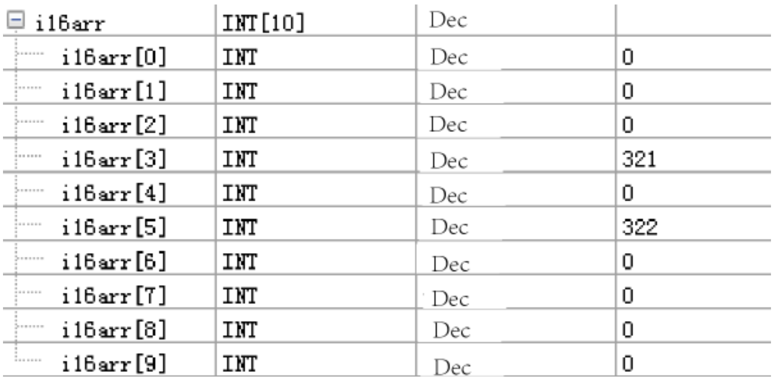


```
Net 1        Net comment
M8000
--| |--[ ADD  addin  K1  addout ]
Program operation flag
Run: ON
Stop: OFF
```

After a trigger by M500, value 321 is assigned for i16arr[3] and then for i16arr[5] after FB calculation. The FB program adds 1 to the input parameter value and then assigns the new value for the output.



```
M500
-|↑|-- [ MOV  K3      3
                      index_in  ]
       [ MOV  K5      5
                      index_out ]
       [ MOV  K321    321
                      i16arr[index_in] ]
                      FBadd1
       ---------------En  FB_ADD
i16arr[index_in] 321 -- addin    addout -- 322 i16arr[index_out]
```



⊟ i16arr	INT[10]	Dec	
i16arr[0]	INT	Dec	0
i16arr[1]	INT	Dec	0
i16arr[2]	INT	Dec	0
i16arr[3]	INT	Dec	321
i16arr[4]	INT	Dec	0
i16arr[5]	INT	Dec	322
i16arr[6]	INT	Dec	0
i16arr[7]	INT	Dec	0
i16arr[8]	INT	Dec	0
i16arr[9]	INT	Dec	0

3.6 Pointer Type Variables

3.6.1 Definition of Pointer Type Variables

Pointer type variables can be used as addresses of pointer storage soft elements or array variables. During programming, pointer type variables can be used for indirect addressing or indexed addressing. H5U and Easy series PLCs do not support the V and Z soft element functions.
In the variable table, after the variable name is defined and POINTER is selected as the data type, a pointer type variable is defined. The initial value of the pointer variable is NULL, indicating a null pointer, and the pointer variable is non-retentive upon a power failure.

Pointer type variables allow address operations and indirect addressing operations. Instruction of pointer address operations specify the address operations of pointers. The following table lists the instruction of pointer address operations. These instructions can be run to obtain addresses, realize offsets for pointer addresses, and compare pointer addresses.

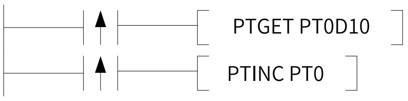
Table 3–3 Instructions of pointer address operations

Instruction	Description
PTGET	Obtaining pointer addresses
PTINC	Pointer variable address incremented by 1
PTDEC	Pointer variable address decremented by 1
PTADD	Adding offsets for pointer variable addresses
PTSUB	Deducting offsets for pointer variable addresses
PT>, PT>=, PT<, PT<=, PT=, and PT<>	PT variable contact comparison
PTMOV	Pointer variable mutual assignment

For other instructions, when pointer type variables are used, indirect addressing operations are performed on pointer type variables, indicating the operations on the values of soft elements or array variables directed by the pointer type variables. Indirect addressing operations on pointer type variables are indicated by "*Pointer type variable" during programming.

Example

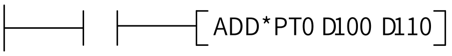
- Address operations on pointer type variables



The PTO points to the D10 address.

The PTO points to the next address.

- Indirect addressing operations on pointer type variables



Add the soft element pointed to by the PTO address to the D100.

Note

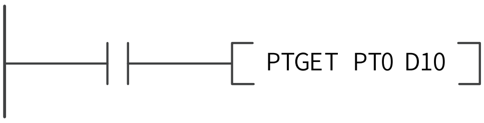
In programming, in addition to the instructions for pointer address operations in the preceding table, if other instructions use a pointer type variable, the programming software automatically adds an asterisk (*) before the variable. Alternatively, users can enter asterisk (*) manually.

3.6.2 Obtaining Directing Addresses of Pointer Type Variables

Directing addresses of pointer type variables can be obtained by the pointer variable assignment instruction (PTGET).

Example

When the instruction energy flow is effective, the pointer type variable PT0 directs to D10. That is, PT0 obtains the address of the D10 soft element.



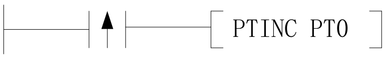
Point the PTO's address to D10

Pointer type variables can direct to bit elements (X, Y, M, S, and B), word elements (D, R, and W), and customized array variables.

3.6.3 Operations on PT Pointer Addresses

After pointer addresses of pointer type variables are obtained, they can be added or deducted to specify the element offset of pointer type variables.

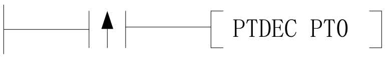
Example 1



When the instruction energy flow is effective, one soft element offset is added for the PT0 pointer address of the pointer type variable. For example, PT0 originally directed to D10 and then directs to D11 after the PTINC instruction is executed. After PTINC execution, the system automatically adds one offset based on the element or array variable type directed by the pointer type variable.

Current PT0 Pointer	PT0 Pointer After PTINC Execution
D10	D11
M200	M201
diVal[3]	diVal[4]

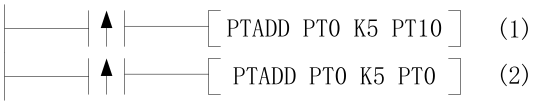
Example 2



When the instruction energy flow is effective, one soft element offset is deducted for the PT0 pointer address of the pointer type variable. For example, PT0 originally directed to D10 and then directs to D9 after the PTDEC instruction is executed. After PTDEC execution, the system automatically deducts one offset based on the element or array variable type directed by the pointer type variable.

Current PT0 Pointer	PT0 Pointer After PTDEC Execution
D10	D9
M200	M199
diVal[3]	diVal[2]

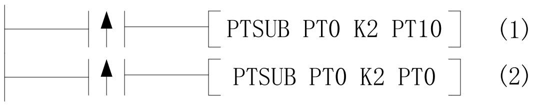
Example 3



When the energy flow of instruction (1) is effective, five soft element offsets are added for the PT0 pointer address of the pointer type variable and assigned to PT10. For example, PT0 originally directed to D10 and then PT10 directs to D15 after execution of instruction (1). When the energy flow of instruction (2) is effective, five soft element offsets are added for the PT0 pointer address and assigned to PT0. For example, PT0 originally directed to D10 and then directs to D15 after execution of instruction (2).

Current PT0 Pointer	Pointer After Execution of Instruction (1)		PT0 Pointer After Execution of Instruction (2)
	PT10 Pointer	PT0 Pointer	
D10	D15	D10	D15
M200	M205	M200	M205
diVal[3]	diVal[8]	diVal[3]	diVal[8]

Example 4



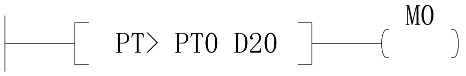
When the energy flow of instruction (1) is effective, two soft element offsets are deducted for the PT0 pointer address of the pointer type variable and assigned to PT10. For example, PT0 originally directed to D10 and then PT10 directs to D8 after execution of instruction (1). When the energy flow of instruction (2) is effective, two soft element offsets are deducted for the PT0 pointer address and assigned to PT0. For example, PT0 originally directed to D10 and then directs to D8 after execution of instruction (2).

Current PT0 Pointer	Pointer After Execution of Instruction (1)		PT0 Pointer After Execution of Instruction (2)
	PT10 Pointer	PT0 Pointer	
D10	D8	D10	D8
M200	M198	M200	M198
diVal[3]	diVal[1]	diVal[3]	diVal[1]

Note

In all the preceding examples of pointer address operations, the PTGET instruction must be used to obtain the pointer address first. When PT points to an array variable, pay attention to boundary checking when executing address operation instructions.

Example 5

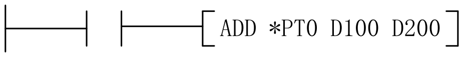


The PT> instruction determines whether the PT0 pointer address of the pointer type variable is greater than D20. If the PT0 pointer directs to D21, the output M0 is ON. Similar inspections such as PT>=, PT<, PT<=, PT=, and PT<> can determine the directions of PT pointers.

3.6.4 Indirect Addressing Operations on Pointer Type Variables

After you obtain an address for a pointer type variable by running address operation instructions, the address can be used in instructions, indicating the indirect addressing operation on the soft element or array variable directed by the pointer variable.

Example



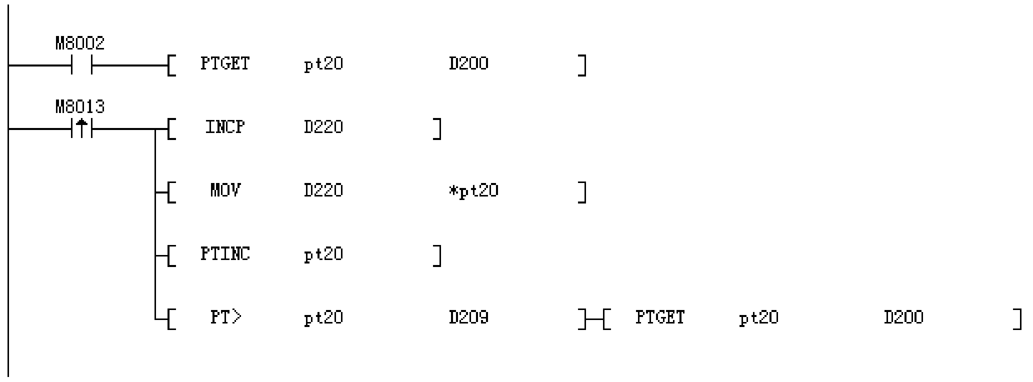
When the instruction energy flow is effective, the soft element directed by the PT0 pointer of the pointer type variable is added to D100. For example, if PT0 directs to D10, the instruction execution result is D200 = D10 + D100.

Note

To indirectly represent a specified soft element using a pointer type variable, an effective pointer address must be obtained by using a pointer address operation instruction first.

3.6.5 Use Example

This section uses a pointer type soft component as an example. The component cycles the value of D220 to the first 10 elements starting from D200 every 1 second.



Note

After a value is assigned to the address pointed to by the pointer, the pointer address is incremented by 1. Therefore, the value for the pointer address monitored by AutoShop in this program is inconsistent with the displayed data.

3.7 System Variables

3.7.1 Overview

This section describes the PLC operation status using system variables, such as the device model, version number, serial port information, and Ethernet and CAN communication status.

3.7.2 System Variable Categories

SysVar (System Variable Category)	Description
_SYS_CAN	CAN communication information, such as the station No., baud rate, and online state of the slave station
_SYS_COM	Serial communication information, such as the station No., baud rate, and online state of the slave station
_SYS_ECAT_Master	EtherCAT master station status
_SYS_ECAT_SLAVE	EtherCAT slave station status
_EthIPScanner	EtherNet/IP system variable information
_SYS_ENCODER_AXIS	Data structure of the external encoder axis
_SYS_ETHERNET	Ethernet communication information, such as the IP address, MAC address, online state, and error diagnosis.
_SYS_INFO	PLC system information, such as the firmware version, real-time clock (RTC), module diagnosis, and system logs.
_SYS_MC_AXIS	Data structure of the motion control axis
_sGROUPAXIS_INFO	Status of a coordinate axis in the axis group
_sMCGROUP_INFO	Axis group status
_sGROUPPOS_INFO	Target position of a coordinate axis in the axis group

3.7.3 _SYS_CAN for CAN Interface Running Information

Table 3–4 _CAN interface information

Name	Data Type	Description	R/W	Comparison with H3U
_CAN.BaudRate	INT	Baud rate (kbps)	R	D8285
_CAN.LoadRate	INT	Load rate (%)	R	D8240
_CAN.RxPerSec	INT	Received frames per second (FPS)	R	D8290
_CAN.TxPerSec	INT	Sent FPS	R	D8291-D8290
_CAN.RxErrCnt	INT	Receive error counter	R	High-order 8 bits of D8989

Name	Data Type	Description	R/W	Comparison with H3U
_CAN.TxErrCnt	INT	Send error counter	R	Low-order 8 bits of D8989
_CAN.Protocol	INT	Communication protocol. 0: CANlink; 1: CANopen	R	D8280

Table 3–5 _CANLink interface information

Name	Data Type	Description	R/W	Comparison with H3U
_CANLink.Address	INT	Station No. or address	R	D8284
_CANLink.Heartbeat	INT	Heartbeat time (ms)	R	D8282
_CANLink.NetworkStart	BOOL	Network startup	R	M8290
_CANLink.SyncTrigger	BOOL	Sync triggering	R	M8291
_CANLink.SyncWrErr	INT	Synchronous write error code	R	D8307
_CANLink.ConfigErr	INT	Configuration error code	R	D8308
_CANLink.NodeState[0]	INT	Local station status (=2: online; ≠2: offline)	R	D7800
_CANLink.NodeState[1]	INT	1# station status (=2: online; ≠2: offline)	R	D7801
_CANLink.Online[0]	DINT	Station status on the CANlink configuration monitoring page of AutoShop, with one bit indicating a station	R	D8241
_CANLink.Online[1]	DINT	Station status on the CANlink configuration monitoring page of AutoShop, with one bit indicating a station	R	D8242

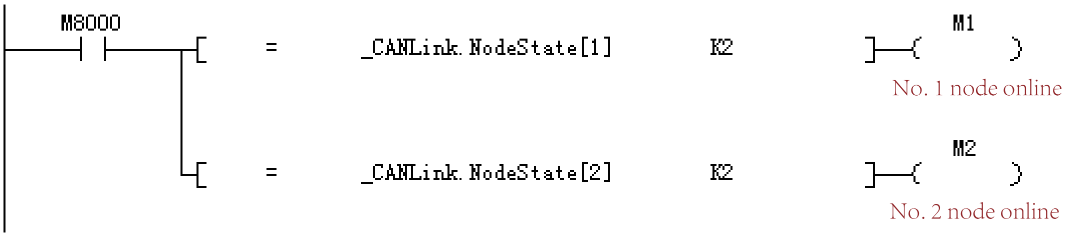
Table 3–6 _CANOpen interface information

Name	Data Type	Description	R/W	Comparison with H3U
_CANOpen.NodeID	INT	Node ID	R	D8284
_CANOpen.NodeState [0]	INT	Local station status (=5: online; ≠5: offline)	R	D7800
_CANOpen.NodeState [1]	INT	1# station status (=5: online; ≠5: offline)	R	D7801
_CANOpen.EMCY. NodeID	INT	Emergency event node ID	R	
_CANOpen.EMCY. ErrorCode	INT	Emergency event error code	R	
_CANOpen.Debug	_sCOP_DEBUG	Commissioning information	R	
_CANOpen.Debug. NodeID	INT	Commissioning information	R	
_CANOpen.Debug.State	INT	Commissioning information	R	
_CANOpen.Debug.Index	INT	Commissioning information	R	
_CANOpen.Debug. SubIndexAndSize	INT	Commissioning information	R	
_CANOpen.Debug.Data	INT[4]	Commissioning information	R	
_CANOpen.Debug.Data [0]	INT	Commissioning information	R	

Name	Data Type	Description	R/W	Comparison with H3U
_CANOpen.Debug.Data [1]	INT	Commissioning information	R	
_CANOpen.Debug.Data [2]	INT	Commissioning information	R	
_CANOpen.Debug.Data [3]	INT	Commissioning information	R	
_CANOpen.ConfigError. NodeID	INT	Configuration error node ID	R	D8287
_CANOpen.ConfigError. ConfigIndex	INT	Configuration No.	R	D8288
_CANOpen.ConfigError. ErrorCode	DINT	Fault code	R	D8254 and D8255

Program example

Determination on the online status of slave stations



3.7.4 _SYS_COM for Serial Port Running Information

Table 3–7 _COM serial port information

Name	Data Type	Description	R/W	Comparison with H3U
_COM.BaudRate	DINT	Baud rate (bps)	R	Bit4 to bit7 of D8120
_COM.DataBits	INT	Data bit	R	Bit0 of D8120
_COM.Parity	INT	Parity bit	R	Bit1 and bit2 of D8120
_COM.StopBits	INT	Stop bit	R	Bit3 of D8120
_COM.Interface	INT	Physical interface	R	-

The preceding table lists configuration information of the COM port. Each serial port corresponds to a separate system variable. _COM corresponds to COM0, and _COM1 to _COM15 correspond to COM1 to COM15 respectively.

Table 3–8 Modbus-based _MbMst master station (serial port) information

Name	Data Type	Description	R/W	Comparison with H3U
_MbMst.Port	INT	Serial port number	R	-
_MbMst.Timeout	INT	Timeout interval (ms)	R	D8129*10
_MbMst.Enable	BOOL	Enabled	R	-
_MbMst.Activate	BOOL	Activated	R	-
_MbMst.Busy	BOOL	Busy	R	-
_MbMst.Error	BOOL	Error	R	M8129
_MbMst.ResponseTime	INT	Response time (ms)	R	-

The preceding table lists Modbus-based information about serial ports of the master station. Each serial port corresponds to a separate system variable. _MbMst corresponds to COM0, and _MbMst1 to _MbMst15 correspond to COM1 to COM15 respectively.

Table 3–9 Modbus-based _MbMstEx master station (serial port) extension information

Name	Data Type	Description	R/W	Comparison with H3U
_MbMstEx. SlvDisableSetFlag	DINT	Whether to disable slave stations	R/W	-
_MbMstEx.SlvDisable	BOOL[256]	Slave station disabled	R/W	-
_MbMstEx.SlvDisable[0]	BOOL	-	R/W	-
_MbMstEx.SlvDisable[1]	BOOL	-	R/W	-
_MbMstEx.SlvDisable[...]	BOOL	-	R/W	-
_MbMstEx.SlvDisable[255]	BOOL	-	R/W	-
_MbMstEx.RetryTimes	INT	Number of retries (Modbus instructions)	R/W	-

The preceding table lists Modbus-based extension information about serial ports of the master station. Each serial port corresponds to a separate system variable. _MbMstEx corresponds to COM0, and _MbMstEx1 to _MbMstEx15 correspond to COM1 to COM15 respectively.

Table 3–10 Modbus-based _MbSlv slave station (serial port) information

Name	Data Type	Description	R/W	Comparison with H3U
_MbSlv.Port	INT	Serial port number	R	-
_MbSlv.SlaveAddress	INT	Slave address	R	D8121
_MbSlv.Connected	BOOL	Connected state	R	-

The preceding table lists Modbus-based information about serial ports of the slave station. Each serial port corresponds to a separate system variable. _MbSlv corresponds to COM0, and _MbSlv1 to _MbSlv15 correspond to COM1 to COM15 respectively.

Table 3–11 Free protocol-based _SerialSR serial port information

Name	Data Type	Description	R/W	Comparison with H3U
_SerialSR.port	INT	Serial port/Port number	R	-
_SerialSR.states	INT	Operation status	R	-
_SerialSR.sent	INT	Number of sent bytes	R	-
_SerialSR.received	INT	Number of received bytes	R	-
_SerialSR.mutexF	DINT	Interlock flag	R	-
_SerialSR.trigger	DINT	Trigger flag	R	-
_SerialSR.errorid	DINT	Error information	R	-
_SerialSR.timeout	DINT	Timeout interval (ms)	R	D8129*10

Table 3–12 Sendbuf to send buffer data

Name	Data Type	Description	R/W	Comparison with H3U
_SerialSR.sendlen	DINT	Number of sent bytes	R	-
_SerialSR.sendbuf	INT[256]	Transmission buffer	R	-
_SerialSR.sendbuf[0]	INT	-	R	-
_SerialSR.sendbuf[1]	INT	-	R	-

Name	Data Type	Description	R/W	Comparison with H3U
_SerialSR.sendbuf[2]	INT	-	R	-
_SerialSR.sendbuf[...]	INT	-	R	-
_SerialSR.sendbuf[254]	INT	-	R	-
_SerialSR.sendbuf[255]	INT	-	R	-

Table 3–13 Recvbuf to receive buffer data

Name	Data Type	Description	R/W	Comparison with H3U
_SerialSR.recvlen	DINT	Number of received bytes	R	-
_SerialSR.recvbuf	INT[256]	Reception buffer	R	-
_SerialSR.recvbuf[0]	INT	-	R	-
_SerialSR.recvbuf[1]	INT	-	R	-
_SerialSR.recvbuf[2]	INT	-	R	-
_SerialSR.recvbuf[...]	INT	-	R	-
_SerialSR.recvbuf[254]	INT	-	R	-
_SerialSR.recvbuf[255]	INT	-	R	-

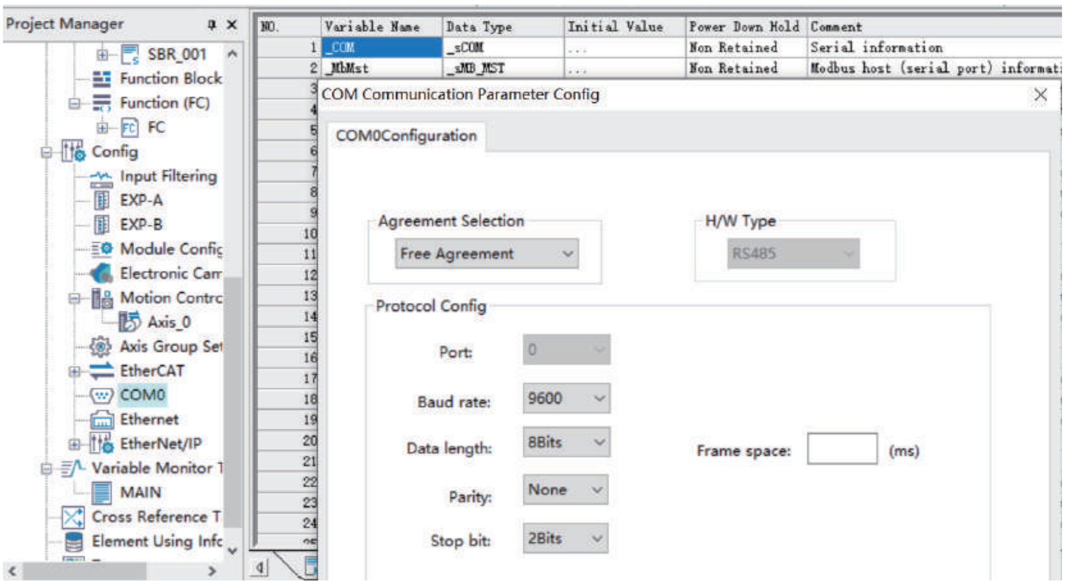
Table 3–14 Free protocol-based instruction configuration for a serial port

Name	Data Type	Description	R/W	Comparison with H3U
_SerialSR.abort	INT	Serial port free protocol canceled	R/W	-
_SerialSR.startchar_en	INT	Reception start character enable (0 to 4)	R/W	-
_SerialSR.startchar	BYTE[4]	Reception start character	R/W	-
_SerialSR.startchar[0]	BYTE	-	R/W	-
_SerialSR.startchar[1]	BYTE	-	R/W	-
_SerialSR.startchar[2]	BYTE	-	R/W	-
_SerialSR.startchar[3]	BYTE	-	R/W	-
_SerialSR.endchar_en	INT	Reception end character enable (0 to 4)	R/W	-
_SerialSR.endchar	BYTE[4]	Reception end character	R/W	-
_SerialSR.endchar[0]	BYTE	-	R/W	-
_SerialSR.endchar[1]	BYTE	-	R/W	-
_SerialSR.endchar[2]	BYTE	-	R/W	-
_SerialSR.endchar[3]	BYTE	-	R/W	-
_SerialSR.Bytetimeout_en	BOOL	Idle interrupted frame reception enable	R/W	
_SerialSR.Bytetimeout	INT	Duration for judging idle interrupted frame reception (ms)	R/W	

The preceding table lists free protocol-based instruction configuration for a serial port. Each serial port corresponds to a separate system variable. _SerialSR corresponds to COM0, and _SerialSR1 to _SerialSR15 correspond to COM1 to COM15 respectively.

Program example

1. COM configuration information



NO.	Variable Name	Data Type	Initial Value	Power Down Hold	Comment
1	_COM	_sCOM	...	Non Retained	Serial information
2	_MbMst	_sMB_MST	...	Non Retained	Modbus host (serial port) informati

COM Communication Parameter Config

COM0Configuration

Agreement Selection: Free Agreement

H/W Type: RS485

Protocol Config

Port: 0

Baud rate: 9600

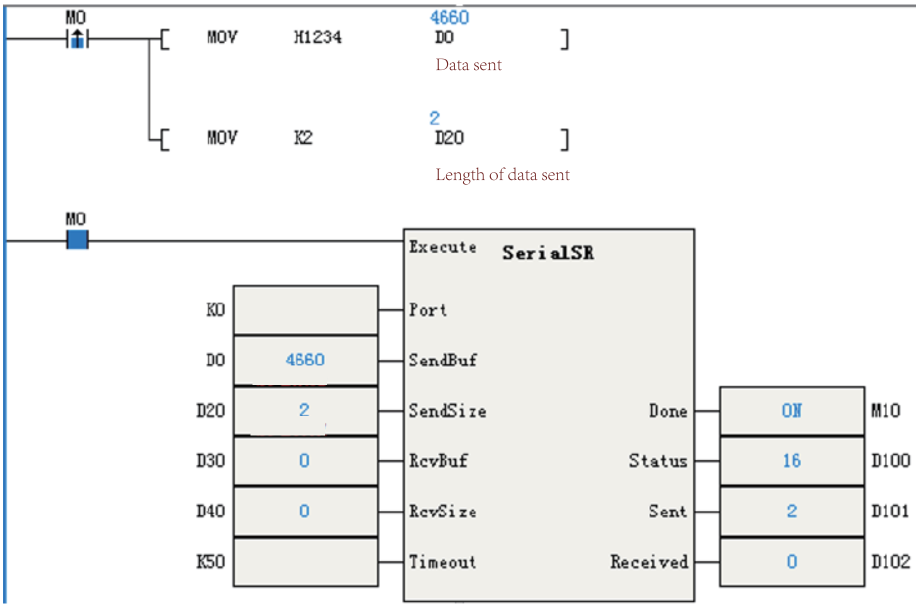
Data length: 8Bits

Parity: None

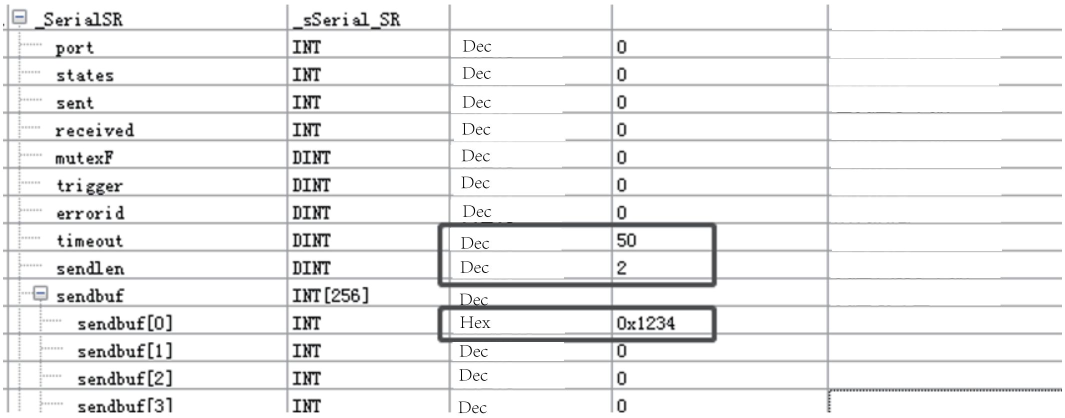
Stop bit: 2Bits

Frame space: (ms)

2. SerialSR instruction sending



```
M0
─┤↑├─┬─[ MOV   H1234   D0   ]   4660  Data sent
     └─[ MOV   K2      D20  ]   2     Length of data sent

M0
─┤ ├──── Execute   SerialSR
     K0        Port
     D0  4660  SendBuf
     D20 2     SendSize        Done      ON  M10
     D30 0     RcvBuf          Status    16  D100
     D40 0     RcvSize         Sent      2   D101
     K50       Timeout         Received  0   D102
```



_SerialSR	_sSerial_SR		
port	INT	Dec	0
states	INT	Dec	0
sent	INT	Dec	0
received	INT	Dec	0
mutexF	DINT	Dec	0
trigger	DINT	Dec	0
errorid	DINT	Dec	0
timeout	DINT	Dec	50
sendlen	DINT	Dec	2
sendbuf	INT[256]	Dec	
sendbuf[0]	INT	Hex	0x1234
sendbuf[1]	INT	Dec	0
sendbuf[2]	INT	Dec	0
sendbuf[3]	INT	Dec	0

3.7.5 _SYS_COM_SAVE for Serial Port Parameter Settings

Table 3–15 _COMSet serial port parameter settings

Name	Data Type	Description	R/W	Comparison with H3U
_COMSet.SetFlag	DINT	Parameter setting enable flag	R/W	-
_COMSet.BaudRate	DINT	Baud rate (bps)	R/W	-
_COMSet.DataBits	INT	Data bit	R/W	-
_COMSet.Parity	INT	Parity bit	R/W	-
_COMSet.StopBits	INT	Stop bit	R/W	-
_COMSet.Interface	INT	Physical interface	R/W	-
_COMSet.Protocol	INT	Communication protocol	R/W	-

The preceding table lists parameter settings of the COM port. Each serial port corresponds to a separate system variable. _COMSet corresponds to COM0, and _COM1Set to _COM15Set correspond to COM1 to COM15 respectively.

Table 3–16 _COMProtocolSet serial port parameter settings

Name	Data Type	Description	R/W	Comparison with H3U
_COMProtocolSet.port	INT	Serial port number	R	-
_COMProtocolSet. AddressSetFlag	INT	Whether to set the slave station number or address	R/W	-
_COMProtocolSet. Address	INT	Slave station number or address	R/W	-

The preceding table lists protocol-based parameter settings of the COM port. Each serial port corresponds to a separate system variable. _COMProtocolSet corresponds to COM0, and _COM1ProtocolSet to _COM15ProtocolSet correspond to COM1 to COM15 respectively.

3.7.6 _SYS_ECAT_Master for Operation Status

Table 3–17 EtherCAT master station information

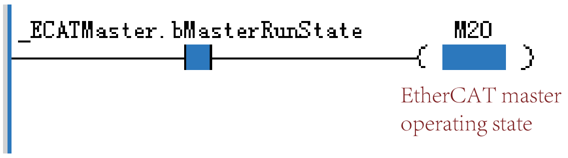
Name	Data Type	Description	R/W
_ECATMaster.bMasterRunState	BOOL	Operation status of the master station (ON: running; OFF: stopped)	R
_ECATMaster.bLinkState	BOOL	Connection status of the master station (ON: normal; OFF: LAN cable disconnected)	R
_ECATMaster.bHeartBeat	BOOL	EtherCAT task heartbeat	R
_ECATMaster.dMaxCycleTime	DINT	Maximum cycle time (µs)	R
_ECATMaster.dMinCycleTime	DINT	Minimum cycle time (µs)	R
_ECATMaster.dCycleTime	DINT	Cycle time (µs)	R
_ECATMaster.dMaxExeTime	DINT	Maximum execution time (µs)	R
_ECATMaster.dMinExeTime	DINT	Minimum execution time (µs)	R
_ECATMaster.dExeTime	DINT	Execution time (µs)	R
_ECATMaster.dtx_frames	DINT	Total number of sent frames	R
_ECATMaster.drx_frames	DINT	Total number of received frames	R
_ECATMaster.dtx_frame_rates	DINT	Frame sending rate (frame/s)	R

Name	Data Type	Description	R/W
_ECATMaster.drx_frame_rates	DINT	Frame receiving rate (frame/s)	R
_ECATMaster.dtx_bytes_rate	DINT	Byte sending rate (byte/s)	R
_ECATMaster.drx_bytes_rate	DINT	Byte receiving rate (byte/s)	R
_ECATMaster.dloss_frames	DINT	Number of lost EtherCAT frames	R
_ECATMaster.bResetTime	BOOL	Execution time and cycle time for resetting	R/W
_ECATMaster.bStartMaster	BOOL	Master station startup (When it is set to ON, the EtherCAT master station is restarted and then the value automatically turns to OFF.)	R/W
_ECATMaster.bStopMaster	BOOL	Master station stop (When it is set to ON, the EtherCAT master station is stopped and then the value automatically turns to OFF.)	R/W
_ECATMaster.bClearFrameCounter	BOOL	Frame sending and receiving counter upon resetting	R/W
_ECATMaster.iSlavesState	INT	Online status of all slave stations (1: All slave stations are online; 0: Some slave stations are offline.)	R
_ECATMaster.dLibVersion	DINT	EtherCAT library version	R
_ECATMaster.dMstVersion	DINT	EtherCAT master station version	R
_ECATMaster.dDriveVersion	DINT	Version of the EtherCAT network adapter driver	R
_ECATMaster.dtx_error_cnt	DINT	Number of EtherCAT sending errors	R
_ECATMaster.drx_timeout_cnt	DINT	Number of frame receiving timeout events by EtherCAT	R
_ECATMaster.drx_corrupt_cnt	DINT	Number of invalid frame receiving events by EtherCAT	R
_ECATMaster.drx_unmach_cnt	DINT	Number of mismatched frame receiving events by EtherCAT	R
_ECATMaster.dRxPDOLength	DINT	Total number of PDOs received by EtherCAT	R
_ECATMaster.dTxPDOLength	DINT	Total number of PDOs sent by EtherCAT	R
_ECATMaster.dConfigureState	DINT	EtherCAT configuration status	R
_ECATMaster.dDelay	DINT	EtherCAT synchronization regulator	R

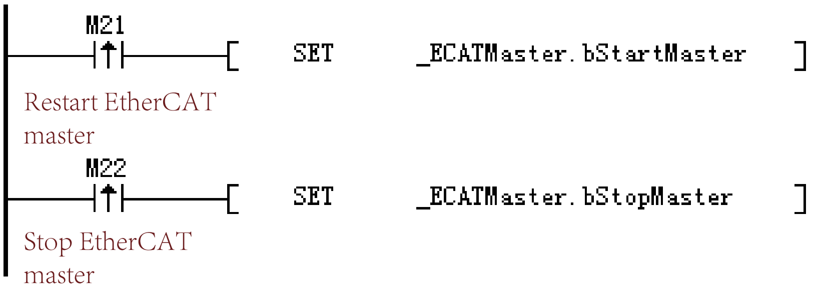
The preceding table lists information about the EtherCAT master station, such as the master station operation status and maximum cycle time.

Program example

1. Monitoring on the EtherCAT master station status



2. Restartup of the EtherCAT master station



Note

The instruction enable is edge-triggered.

3.7.7 _SYS_ECAT_Slave for Operation Status

Table 3–18 EtherCAT slave station information

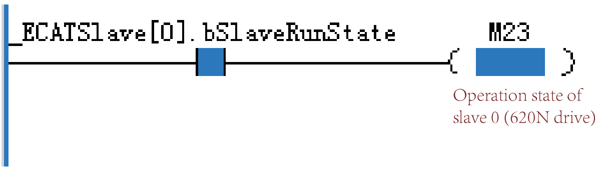
Name	Data Type	Description	R/W
_ECATSlave	_sECTSLV_INFO [125]	Operation status of the EtherCAT slave station	-
_ECATSlave[0]	_sECTSLV_INFO	-	-
_ECATSlave[0].bSlaveRunState	BOOL	Operation status of the slave station (ON: running; OFF: stopped)	R
_ECATSlave[0].bSetAliasState	BOOL	Status of slave station alias writing (ON: busy)	R
_ECATSlave[0].bSetAliasError	BOOL	Failure of slave station alias writing	R
_ECATSlave[0].bSetAlias	BOOL	Setting of the slave station alias, valid on the rising edge (used for commissioning on the AutoShop configuration page)	R/W
_ECATSlave[0].wALState	INT	Status of the EtherCAT state machine (1/2/4/8)	R
_ECATSlave[0].wAlCode	INT	Fault Codes	R
_ECATSlave[0].wActAlias	INT	Actual station alias	R
_ECATSlave[0].wTarAlias	INT	Station alias to be written (used for commissioning on the AutoShop configuration page)	R/W
_ECATSlave[0].wStationAddress	INT	Actual station name	R

Note

_ECATSlave is an array. For example, [0] represents configuration address 0.

Program example

Monitoring on the EtherCAT slave station status


_ECATSlave[0].bSlaveRunState
M23
Operation state of slave 0 (620N drive)

3.7.8 _SYS_EncAxis for Encoder Axis Information

Table 3–19 EncAxis encoder axis information

Name	Data Type	Description	R/W
_EncAxis[0]	_sENC_AXIS	-	R
_EncAxis[0].Axis	INT	Axis No.	R
_EncAxis[0].Reserced0	INT	Reserved	R
_EncAxis[0].Unit	REAL	Unit	R
_EncAxis[0].UpperPosition	REAL	Upper limit position	R
_EncAxis[0].LowerPosition	REAL	Lower limit position	R
_EncAxis[0].Position	REAL	Position	R
_EncAxis[0].Velocity	REAL	Speed	R
_EncAxis[0].Reserced1	DINT[2]	Reserved	R
_EncAxis[0].Reserced1[0]	DINT	-	R
_EncAxis[0].Reserced1[1]	DINT	-	R

Table 3–20 EncAxis[0].Counter counter information

Name	Data Type	Description	R/W
_EncAxis[0].Counter	_sENC_CNT	Counter	R
_EncAxis[0].Counter.ID	INT	Counter ID	R
_EncAxis[0].Counter.DecodeMode	INT	Decoding mode	R
_EncAxis[0].Counter.Source	INT	Signal source	R
_EncAxis[0].Counter.CountMode	INT	Count mode	R
_EncAxis[0].Counter.ControlWord	INT	Control word	R
_EncAxis[0].Counter.StatusWord	INT	Status word	R
_EncAxis[0].Counter.CountValue	DINT	Count value	R
_EncAxis[0].Counter.Frequency	REAL	Frequency	R

Table 3–21 EncAxis[0].Reset reset information

Name	Data Type	Description	R/W
_EncAxis[0].Reset	_sENC_RST	Reset	R
_EncAxis[0].Reset.Pin	INT	Reset pin	R
_EncAxis[0].Reset.Edge	INT	Reset edge	R

Table 3–22 EncAxis[0].Preset preset information

Name	Data Type	Description	R/W
_EncAxis[0].Preset	_sENC_PRESET	Resetting	R
_EncAxis[0].Preset.Pin	INT	Preset pin	R
_EncAxis[0].Preset.Edge	INT	Preset edge	R
_EncAxis[0].Preset.Value	DINT	Preset value	R
_EncAxis[0].Preset.Position	REAL	Preset position	R

Table 3–23 EncAxis[0].Probe probe information

Name	Data Type	Description	R/W
_EncAxis[0].Probe	_sENC_PROBE[2]	Probe	R
_EncAxis[0].Probe[0]	_sENC_PROBE		R
_EncAxis[0].Probe[0].Pin	INT	Pin	R
_EncAxis[0].Probe[0].Edge	INT	Probe edge	R
_EncAxis[0].Probe[0].PositiveValue	DINT	Probe rising edge	R
_EncAxis[0].Probe[0].NegativeValue	DINT	Probe falling edge	R
_EncAxis[0].Probe[0].PositivePosition	REAL	Probe rising edge position	R
_EncAxis[0].Probe[0].NegativePosition	REAL	Probe falling edge position	R
_EncAxis[0].Probe[1]	_sENC_PROBE		R
_EncAxis[0].Probe[1].Pin	INT	Pin	R
_EncAxis[0].Probe[1].Edge	INT	Probe edge	R
_EncAxis[0].Probe[1].PositiveValue	DINT	Probe rising edge	R
_EncAxis[0].Probe[1].NegativeValue	DINT	Probe falling edge	R
_EncAxis[0].Probe[1].PositivePosition	REAL	Probe rising edge position	R
_EncAxis[0].Probe[1].NegativePosition	REAL	Probe falling edge position	R

Table 3–24 EncAxis[0].Match comparison interruption

Name	Data Type	Description	R/W
_EncAxis[0].Match	_sENC_MATCH[2]	Compare	R
_EncAxis[0].Match[0]	_sENC_MATCH		R
_EncAxis[0].Match[0].Value	DINT	Value	R
_EncAxis[0].Match[0].Position	REAL	Position	R
_EncAxis[0].Match[0].Enable	BOOL	Enable	R
_EncAxis[0].Match[0].InterruptEnable	BOOL	Interruption enable	R
_EncAxis[0].Match[0].OutputEnable	BOOL	Output enable	R
_EncAxis[0].Match[0].InterruptMap	INT	Interruption association	R
_EncAxis[0].Match[0].OutputPin	INT	Output pin	R
_EncAxis[0].Match[0].OutputMode	INT	Output mode	R
_EncAxis[0].Match[0].OutputWidth	REAL	Output width	R
_EncAxis[0].Match[1]	_sENC_MATCH		R
_EncAxis[0].Match[1].Value	DINT	Value	R
_EncAxis[0].Match[1].Position	REAL	Position	R
_EncAxis[0].Match[1].Enable	BOOL	Enable	R
_EncAxis[0].Match[1].InterruptEnable	BOOL	Interruption enable	R
_EncAxis[0].Match[1].OutputEnable	BOOL	Output enable	R
_EncAxis[0].Match[1].InterruptMap	INT	Interruption association	R
_EncAxis[0].Match[1].OutputPin	INT	Output pin	R
_EncAxis[0].Match[1].OutputMode	INT	Output mode	R
_EncAxis[0].Match[1].OutputWidth	REAL	Output width	R

3.7.9 _SYS_Ethernet for Ethernet Information

Table 3–25 Ethernet network port information

Name	Data Type	Description	R/W
_Ethernet.MACAddress	INT[3]	Physical address	R
_Ethernet.MACAddress[0]	INT	-	R
_Ethernet.MACAddress[1]	INT	-	R
_Ethernet.MACAddress[2]	INT	-	R
_Ethernet.IPAddress	DINT	Local IP address	R/W
_Ethernet.Mask	DINT	Subnet mask	R/W
_Ethernet.Gateway	DINT	Gateway	R/W
IPCommand	INT	IP command	R/W

The preceding variable table lists local information such as the IP address and MAC addresses.

Table 3–26 Modbus-TCP-based MbTcpMst master station information

Name	Data Type	Description	R/W
_MbTcpMst[0]	_sMB_TCP_MST	-	R
_MbTcpMst[0].IPAddress	DINT	IP address of a slave station	R
_MbTcpMst[0].Port	INT	Port number of a slave station	R
_MbTcpMst[0].Timeout	INT	Timeout interval (ms)	R
_MbTcpMst[0].Number	INT	Configuration number	R
_MbTcpMst[0].Enable	BOOL	Enabled	R
_MbTcpMst[0].Connected	BOOL	Connected state	R
_MbTcpMst[0].Busy	BOOL	Busy	R
_MbTcpMst[0].Error	BOOL	Error	R
_MbTcpMst[0].ResponseTime	INT	Response time (ms)	R
_MbTcpMst[...]	_sMB_TCP_MST	-	R

The preceding table lists Modbus-TCP-based information about Ethernet of the master station.

Table 3–27 Modbus-TCP-based MbTcpMst slave station information

Name	Data Type	Description	R/W
_MbTcpSlv.Port	INT	Port number	R
_MbTcpSlv.SlaveAddress	INT	Slave address	R
_MbTcpSlv.Connected	BOOL	Connected state	R
_MbTcpSlv.Connections	INT	Number of connections	R
_MbTcpSlv.IPAddress	DINT[32]	List of client IP addresses	R
_MbTcpSlv.IPAddress[0]	DINT	-	R
_MbTcpSlv.IPAddress[1]	DINT	-	R
_MbTcpSlv.IPAddress[2]	DINT	-	R
_MbTcpSlv.IPAddress[...]	DINT	-	R
_MbTcpSlv.IPAddress[30]	DINT	-	R
_MbTcpSlv.IPAddress[31]	DINT	-	R

The preceding table lists ModbusTCP-based information about the client linked to the PLC slave station.

3.7.10 _EthIPScanner for Status Information

Table 3–28 EIP system variable information

Name	Data Type	Description	R/W
EthIPScanner[0].Instance	DINT	Label instance ID	R
EthIPScanner[0].Connected	DINT	Connecting status 1: Initializing 2: Invalid network path 3: No response 4: Response error 5: Timeout 6: Connection closed	R
EthIPScanner[0].GeneralStatus	INT	General error state	R
EthIPScanner[0].ExtendedStatus	INT	Extended error state	R
EthIPScanner [0].Reserved	DINT[6]	Reserved field R	R
...	...	...	...
EthIPScanner[255].Instance	DINT	Label instance ID	R
EthIPScanner[255].Connected	DINT	Connecting status 1: Initializing 2: Invalid network path 3: No response 4: Response error 5: Timeout 6: Connection closed	R
EthIPScanner[255].GeneralStatus	INT	General error state	R
EthIPScanner[255].ExtendedStatus	INT	Extended error state	R
EthIPScanner [255].Reserved	DINT[6]	Reserved field R	R

Note

- EthIPScanner[n-1] represents the tag data with instance value n, and EthIPScanner[n-1].Instance has a value of n. The array subscript corresponds one-to-one with Instance, with a difference of 1.
- To determine that "the tag with instance value n has normal communication", the system variable EthIPScanner [n-1] must meet the following conditions:
  - CLASS1 type tag: Connected==1 && GeneralStatus==0 && ExtendedStatus==0
  - CLASS3 type tag: Connected==104 && GeneralStatus==0 && ExtendedStatus==0
  - UCMM type tag: Connected==100 && GeneralStatus==0 && ExtendedStatus==0

3.7.11 _SYS_INFO PLC for Operation Information

Table 3–29 Device information (DevInfo)

Name	Data Type	Description	R/W	Comparison with H3U
_DevInfo.Device	INT	Device model ID	R	-
_DevInfo.Vender	INT	Manufacturer ID	R	-
_DevInfo.HWVersion	DINT	Hardware version	R	-
_DevInfo.SWVersion	DINT	Software version	R	D8100 and D8101

Name	Data Type	Description	R/W	Comparison with H3U
_DevInfo.FPGAVersion	DINT	FPGA version	R	D8104 and D8105
_DevInfo.NSTDVersion	DINT	Customized version	R	-

The preceding table lists the PLC device information.

Table 3–30 OSM system monitor

Name	Data Type	Description	R/W
_OSM.CPU	INT	CPU usage	R
_OSM.Memory	INT	Memory usage	R

The preceding table lists the CPU and memory usage for CPU performance diagnosis.

Table 3–31 User program information

Name	Data Type	Description	R/W	Comparison with H3U
_Program.TotalSize	DINT	Total program capacity	R	
_Program.UsedSize	DINT	Program capacity used	R	
_Program.Interval	DINT	Program task cycle (μs)	R	
_Program.CurPeriod	DINT	Current program task cycle (μs)	R	D8010
_Program.MinPeriod	DINT	Minimum program task cycle (μs)	R	D8011
_Program.MaxPeriod	DINT	Maximum program task cycle (μs)	R	D8012
_Program.CurRunTime	DINT	Current program running time (μs)	R	
_Program.MinRunTime	DINT	Minimum program running time (μs)	R	
_Program.MaxRunTime	DINT	Maximum program running time (μs)	R	
_Program.AveRunTime	DINT	Average program running time (μs)	R	
_Program.Reset	BOOL	Reset cycle time	R/W	

The preceding table lists the execution cycle of the program and task, which can be used to judge the program execution logic complexity.

Table 3–32 List of current errors (CurErrLst)

Name	Data Type	Description	R/W
_CurErrLst.Quantity	INT	Number of current errors	R
_CurErrLst.ErrorInfo	_sERR_INFO[32]	List of current errors	R
_CurErrLst.ErrorInfo[0]	_sERR_INFO		R
_CurErrLst.ErrorInfo[0].ErrorCode	INT	Fault code	R
_CurErrLst.ErrorInfo[0].ComponentID	INT	Component ID	R
_CurErrLst.ErrorInfo[0].Location	DINT	Error position	R
_CurErrLst.ErrorInfo[0].Timestamp	DINT	Timestamp, indicating the number of seconds from 00:00:00 on January 1, 1970 to the error generation time	R

Name	Data Type	Description	R/W
_CurErrLst.ErrorInfo[1]	_sERR_INFO		R
_CurErrLst.ErrorInfo[1].ErrorCode	INT	Fault code	R
_CurErrLst.ErrorInfo[1].ComponentID	INT	Component ID	R
_CurErrLst.ErrorInfo[1].Location	DINT	Error position	R
_CurErrLst.ErrorInfo[1].Timestamp	DINT	Timestamp, indicating the number of seconds from 00:00:00 on January 1, 1970 to the error generation time	R
CurErrLst.ErrorInfo[...]			
_CurErrLst.ErrorInfo[30]	_sERR_INFO		R
_CurErrLst.ErrorInfo[30].ErrorCode	INT	Fault code	R
_CurErrLst.ErrorInfo[30].ComponentID	INT	Component ID	R
_CurErrLst.ErrorInfo[30].Location	DINT	Error position	R
_CurErrLst.ErrorInfo[30].Timestamp	DINT	Timestamp, indicating the number of seconds from 00:00:00 on January 1, 1970 to the error generation time	R
_CurErrLst.ErrorInfo[31]	_sERR_INFO		R
_CurErrLst.ErrorInfo[31].ErrorCode	INT	Fault code	R
_CurErrLst.ErrorInfo[31].ComponentID	INT	Component ID	R
_CurErrLst.ErrorInfo[31].Location	DINT	Error position	R
_CurErrLst.ErrorInfo[31].Timestamp	DINT	Timestamp, indicating the number of seconds from 00:00:00 on January 1, 1970 to the error generation time	R

The preceding table lists PLC error logs, up to 32 records. You can check details in the AutoShop fault record. For details about error codes, see *"19.3 Fault Codes" on page 552*.

Table 3–33 DateTime RTC

Name	Data Type	Description	R/W	Comparison with H3U
_DateTime.Second	INT	Second, ranging from 0 to 60, in which 60 is the leap second	R	D8013
_DateTime.Minute	INT	Minute, ranging from 0 to 59	R	D8014
_DateTime.Hour	INT	Hour, ranging from 0 to 23	R	D8015
_DateTime.Day	INT	Day in a month, ranging from 1 to 31	R	D8016
_DateTime.Month	INT	Month, ranging from 1 to 12	R	D8017
_DateTime.Year	INT	Year	R	D8018
_DateTime.WeekDay	INT	Day of a week, ranging from 0 to 6, in which 0 indicates Sunday and 1 indicates Monday	R	D8019
_DateTime.YearDay	INT	Number of the day counted from January 1 of each year, ranging from 0 to 365, in which 0 indicates January 1	R	
_DateTime.Timestamp	DINT	Total number of seconds from 00:00:00 on January 1, 1970 to the current time	R	

The preceding table lists RTC information.

Table 3–34 ExtSlt local extension module diagnosis information

Name	Data Type	Description	R/W
_ExtSlt[0]	_sEXT_SLT		R
_ExtSlt[0].ConfigModule	INT	Module type of the AutoShop configuration	R
_ExtSlt[0].MountedModule	INT	Type of the installed electrical module	R
_ExtSlt[0].LogicVersion	DINT	Version of the logic device (module version)	R
_ExtSlt[0].SWVersion	DINT	Software version (module version)	R
_ExtSlt[0].Error	BOOL	Error status (ON: failed; OFF: normal)	R
_ExtSlt[1]	_sEXT_SLT		R
_ExtSlt[1].ConfigModule	INT	Configuration module type	R
_ExtSlt[1].MountedModule	INT	Installed module type	R
_ExtSlt[1].LogicVersion	DINT	Version of the logic device	R
_ExtSlt[1].SWVersion	DINT	Software version	R
_ExtSlt[1].Error	BOOL	Error	R
_ExtSlt[...]	_sEXT_SLT		
_ExtSlt[15]	_sEXT_SLT		R
_ExtSlt[15].ConfigModule	INT	Configuration module type	R
_ExtSlt[15].MountedModule	INT	Installed module type	R
_ExtSlt[15].LogicVersion	DINT	Version of the logic device	R
_ExtSlt[15].SWVersion	DINT	Software version	R
_ExtSlt[15].Error	BOOL	Error	R

Table 3–35 M8000/D8000 element

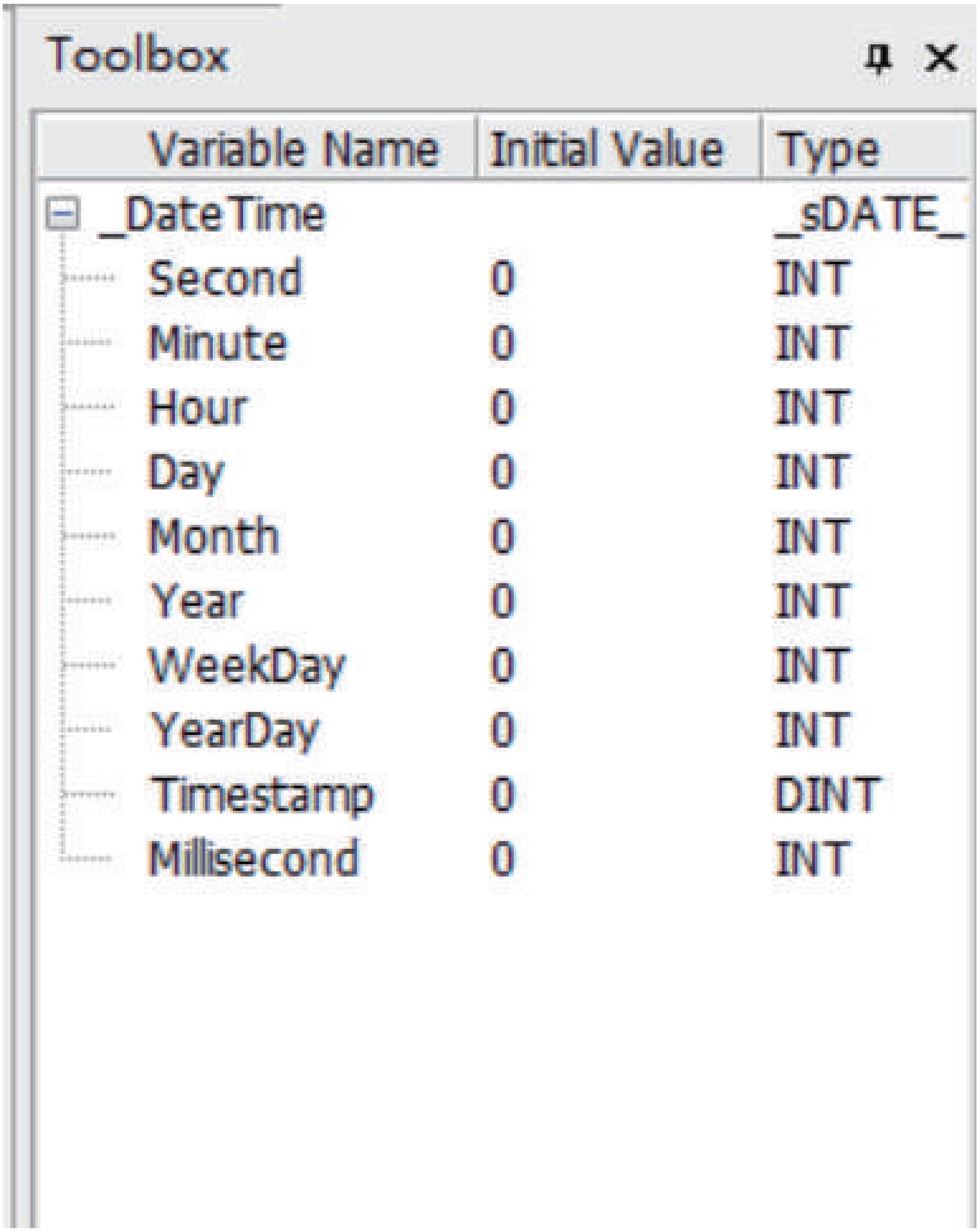
Name	Data Type	Description	R/W
M8000	BOOL	ON during running of the user program	R
M8001	BOOL	Negated M8000 state	R
M8002	BOOL	ON in the first operation cycle of the user program	R
M8003	BOOL	Negated M8002 state	R
M8011	BOOL	Oscillating clock with a cycle of 10 ms	R
M8012	BOOL	Oscillating clock with a cycle of 100 ms	R
M8013	BOOL	Oscillating clock with a cycle of 1s	R
M8014	BOOL	Oscillating clock with a cycle of 1 min	R
M8020	BOOL	Zero flag	R
M8021	BOOL	Borrow flag	R
M8022	BOOL	Carry flag	R
M8029	BOOL	Multi-cycle instruction execution completion flag, applicable to the RAMP, SORT, and SORT2 instructions	R
M8040	BOOL	SFC STL status transition disable	R
M8161	BOOL	OFF: 16-bit mode; ON: 8-bit mode; Bit processing mode for ASCII/HEX/CCD/LRC/CRC/RS	R
M8163	BOOL	Switchover flag of BINDA instruction output characters (retained or switched to 0000h)	R
M8165	BOOL	SORT2 instruction descending sort enable flag	R
M8168	BOOL	SMOV instruction data format, including OFF-BCD and ON-HEX	R
M8333	BOOL	Flag indicating all BKCMP instruction matrix comparison results are 1	R

Name	Data Type	Description	R/W
D8066	INT	Critical errors in user programs and instructions (triggered, not reset)	R
D8067	INT	Minor errors in user programs and instructions (triggered, not reset)	R

The preceding table lists information about M8000/D8000 components. A small number of such components are reserved.

Program example

RTC monitoring table



Variable Name	Initial Value	Type
_DateTime		_sDATE_
Second	0	INT
Minute	0	INT
Hour	0	INT
Day	0	INT
Month	0	INT
Year	0	INT
WeekDay	0	INT
YearDay	0	INT
Timestamp	0	DINT
Millisecond	0	INT

3.7.12 _SYS_MC_Axis for Motion Control Axis Information

Table 3–36 McAxis axis operation status

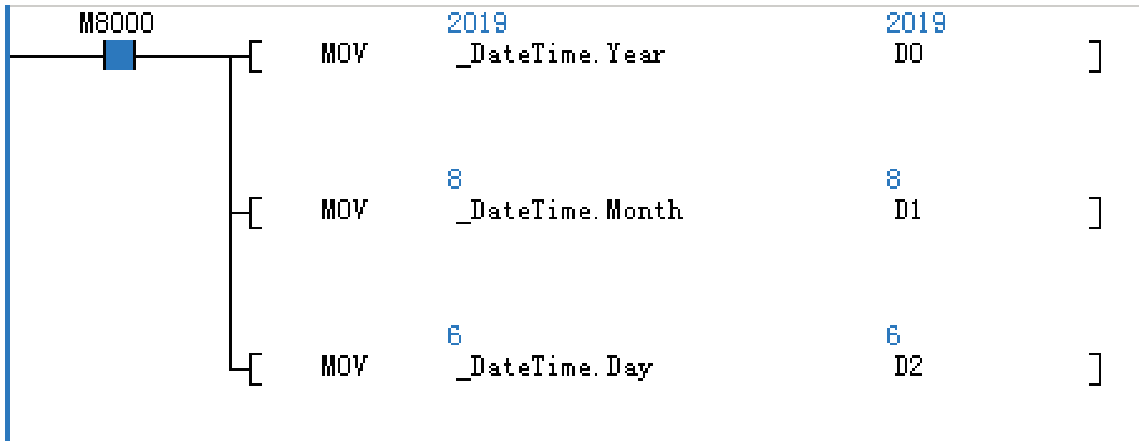
Name	Data Type	Description	R/W
_McAxis[0]	_sMCAXIS_INFO	-	-
_McAxis[0].bPowerState	BOOL	Axis enable state	R
_McAxis[0].bDebugState	BOOL	Axis commissioning state	R
_McAxis[0].fSetPosition	REAL	Position setting	R
_McAxis[0].fSetVelocity	REAL	Speed reference	R
_McAxis[0].fSet_Acc_Dec	REAL	Acceleration/Deceleration rate reference	R
_McAxis[0].fSetTorque	REAL	Torque reference	R
_McAxis[0].fActPosition	REAL	Current position	R
_McAxis[0].fActVelocity	REAL	Current speed	R
_McAxis[0].fAct_Acc_Dec	REAL	Current acceleration/deceleration rate	R
_McAxis[0].fActTorque	REAL	Current torque	R
_McAxis[0].wPLCOpenState	INT	PLCOpen state machine 0: PowerOff 1: ErrorStop 2: Stopping 3: StandStill 4: DiscreteMotion 5: ContinuousMotion 7: Homing 8: SynchronizedMotion	R
_McAxis[0].wConfigState	INT	Configuration status 0: Init (axis in the initialization state) 1: Configure finish (configuration reading completed) 2: Sync finish (synchronized with EtherCAT tasks) 3: Wait Communication (communication with the servo drive established) 4: Slave ready (initialization completed for the servo drive controlled by axes) 5: Axis ready (communication established)	R
_McAxis[0].wAxisError	INT	Axis fault[Note]	R
_McAxis[0].wServoError	INT	Drive fault[Note]	R
_McAxis[0].bEnterDebug	BOOL	Monitoring mode (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bPowerOn	BOOL	Enabled (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bStop	BOOL	Stopped (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bReset	BOOL	Reset (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bJogP	BOOL	Jog+ (online commissioning mode of AutoShop axes)	R/W

Name	Data Type	Description	R/W
_McAxis[0].bJogN	BOOL	Jog- (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bHome	BOOL	Homing (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bSetPos	BOOL	Current position setting (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bAbsPos	BOOL	Absolute positioning (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bRevPos	BOOL	Reciprocating (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bRelPos	BOOL	Relative positioning (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bVelocity	BOOL	Continuous motion (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].bTorque	BOOL	Torque mode (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].wDebugMotionType	INT	Commissioning motion type (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fJogVelocity	REAL	Jog speed (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fPositionOffser	REAL	Homing offset (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fPresetPosition	REAL	Reset position (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarPosition1	REAL	Target position 1 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarVelocity1	REAL	Target speed 1 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarAcceleration1	REAL	Target acceleration rate 1 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarDecelaration1	REAL	Target deceleration rate 1 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].wCurveType1	INT	Curve type 1 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarPosition2	REAL	Target position 2 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarVelocity2	REAL	Target speed 2 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarAcceleration2	REAL	Target acceleration rate 2 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarDecelaration2	REAL	Target deceleration rate 2 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].wCurveType2	INT	Curve type 2 (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].dUnused	DINT	Reserved	R
_McAxis[0].fTarTorque	REAL	Target torque (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].fTarTorqueSlop	REAL	Torque slope (online commissioning mode of AutoShop axes)	R/W
_McAxis[0].wControlWord	INT	Control word	R

Name	Data Type	Description	R/W
_McAxis[0].wStatusword	INT	Status word	R
_McAxis[0].dSetPosition	DINT	Target position	R
_McAxis[0].dActPosition	DINT	Current position	R
_McAxis[0].dSetVelocity	DINT	Speed reference	R
_McAxis[0].dActVelocity	DINT	Current speed	R
_McAxis[0].dSetTorque	INT	Torque reference	R
_McAxis[0].dActTorque	INT	Current torque	R
_McAxis[0].dDO	DINT	Digital output	R
_McAxis[0].dDI	DINT	Digital input	R
_McAxis[0].wModesOfOperation	INT	Control mode 6: Homing mode 8: Synchronous position mode 10: Synchronous torque mode	R
_McAxis[0]. wModesOfOperationDisplay	INT	Current control mode 6: Homing mode 8: Synchronous position mode 10: Synchronous torque mode	R
_McAxis[0].wTouchFunction	INT	Touch probe function	R
_McAxis[0].wTouchStatus	INT	Touch probe status	R
_McAxis[0].dTouch1PPos	DINT	Touch probe 1 positive edge	R
_McAxis[0].dTouch2PPos	DINT	Touch probe 2 positive edge	R
_McAxis[0].dTouch1NPos	DINT	Touch probe 1 falling edge	R
_McAxis[0].dTouch2NPos	DINT	Touch probe 2 negative edge	R
_McAxis[0].wErrorCode	INT	Fault type[Note]	R
_McAxis[0].wAxisRingPos	INT	Axis configuration position	R
_McAxis[0].wAxisID	INT	Axis ID	R
_McAxis[0].fUnits	REAL	Axis gear ratio	R
_McAxis[0].bMotionState	BOOL	Motion status, indicating whether an axis is in motion	R
_McAxis[0].bphlimit	BOOL	Positive limit input status of hardware	R
_McAxis[0].bnhlimit	BOOL	Negative limit input status of hardware	R
_McAxis[0].bhomestate	BOOL	Home switch input status of hardware	R
_McAxis[0].bpslimit	BOOL	Software positive limit reached or not	R
_McAxis[0].bnslimit	BOOL	Software negative limit reached or not	R
_McAxis[0].dLocialAxisSetPos	DINT	Local pulse axis position setting	R
_McAxis[...]	_sMCAXIS_INFO	-	R

[Note]: For details about axis faults and drive faults, see *"12.8 Fault Categories" on page 426*.

Program example

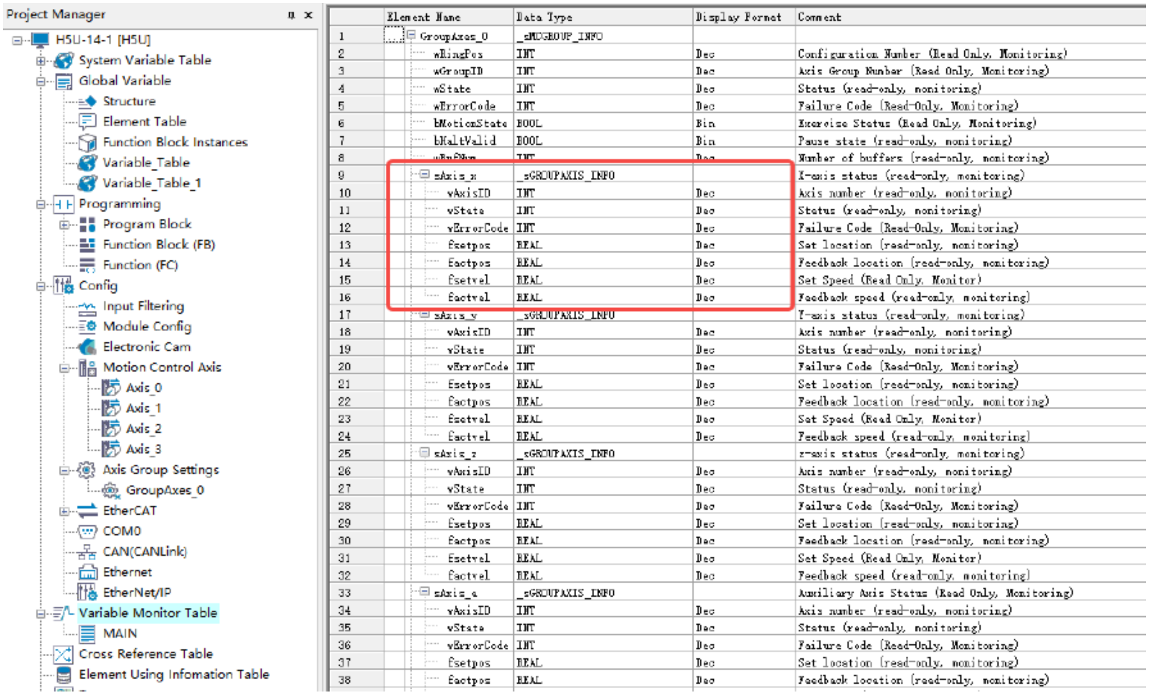


```
M8000
--| |--+--[ MOV   2019              2019  ]
       |         _DateTime.Year     D0
       |
       +--[ MOV   8                 8     ]
       |         _DateTime.Month    D1
       |
       +--[ MOV   6                 6     ]
                 _DateTime.Day      D2
```

3.7.13 _sGROUPAXIS_INFO for Status of Coordinate Axes within Axis Group

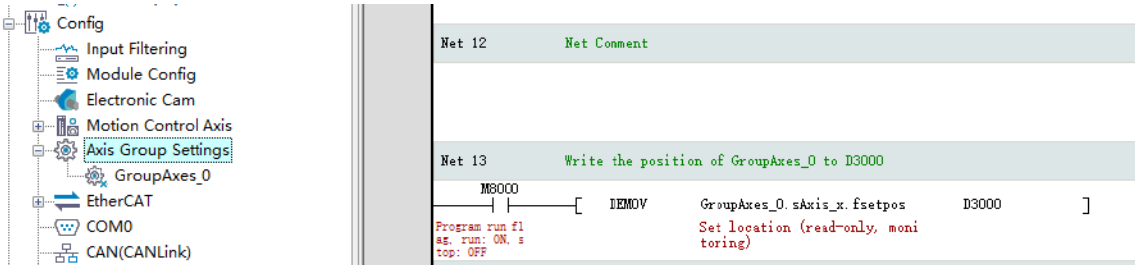
Name	Type	Description
wAxisID	INT16	Axis ID
wState	INT16	The status of an axis' PLCOpen state machine 0: PowerOff 1: ErrorStop 2: Stopping 3: StandStill 4: DiscreteMotion 5: ContinuousMotion 7: Homing 8: SynchronizedMotion
wErrorCode	INT16	The fault code of an axis
fsetpos	REAL	Position reference
factpos	REAL	Feedback position
fsetvel	REAL	Velocity reference
factvel	REAL	Feedback velocity

This system variable exists in the axis group _sMCGROUP_INFO and is used to represent the state of individual axes within the axis group.



	Element Name	Data Type	Display Format	Comment
1	GroupAxes_0	_sMCGROUP_INFO		
2	wRingPos	INT	Dec	Configuration Number (Read Only, Monitoring)
3	wGroupID	INT	Dec	Axis Group Number (Read Only, Monitoring)
4	wState	INT	Dec	Status (read-only, monitoring)
5	wErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
6	bMotionState	BOOL	Bin	Exercise Status (Read Only, Monitoring)
7	bHaltValid	BOOL	Bin	Pause state (read-only, monitoring)
8	wBufNum	INT	Dec	Number of buffers (read-only, monitoring)
9	sAxis_x	_sGROUPAXIS_INFO		X-axis status (read-only, monitoring)
10	vAxisID	INT	Dec	Axis number (read-only, monitoring)
11	vState	INT	Dec	Status (read-only, monitoring)
12	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
13	fsetpos	REAL	Dec	Set location (read-only, monitoring)
14	factpos	REAL	Dec	Feedback location (read-only, monitoring)
15	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
16	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
17	sAxis_y	_sGROUPAXIS_INFO		Y-axis status (read-only, monitoring)
18	vAxisID	INT	Dec	Axis number (read-only, monitoring)
19	vState	INT	Dec	Status (read-only, monitoring)
20	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
21	fsetpos	REAL	Dec	Set location (read-only, monitoring)
22	factpos	REAL	Dec	Feedback location (read-only, monitoring)
23	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
24	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
25	sAxis_z	_sGROUPAXIS_INFO		z-axis status (read-only, monitoring)
26	vAxisID	INT	Dec	Axis number (read-only, monitoring)
27	vState	INT	Dec	Status (read-only, monitoring)
28	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
29	fsetpos	REAL	Dec	Set location (read-only, monitoring)
30	factpos	REAL	Dec	Feedback location (read-only, monitoring)
31	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
32	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
33	sAxis_a	_sGROUPAXIS_INFO		Auxiliary Axis Status (Read Only, Monitoring)
34	vAxisID	INT	Dec	Axis number (read-only, monitoring)
35	vState	INT	Dec	Status (read-only, monitoring)
36	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
37	fsetpos	REAL	Dec	Set location (read-only, monitoring)
38	factpos	REAL	Dec	Feedback location (read-only, monitoring)

For example, write the position reference of the X-axis into the D3000 in the PLC:



```
Net 12          Net Comment

Net 13          Write the position of GroupAxes_0 to D3000
   M8000
───┤ ├────────[   DEMOV    GroupAxes_0.sAxis_x.fsetpos     D3000      ]
Program run fl             Set location (read-only, moni
ag. run: ON, s             toring)
top: OFF
```

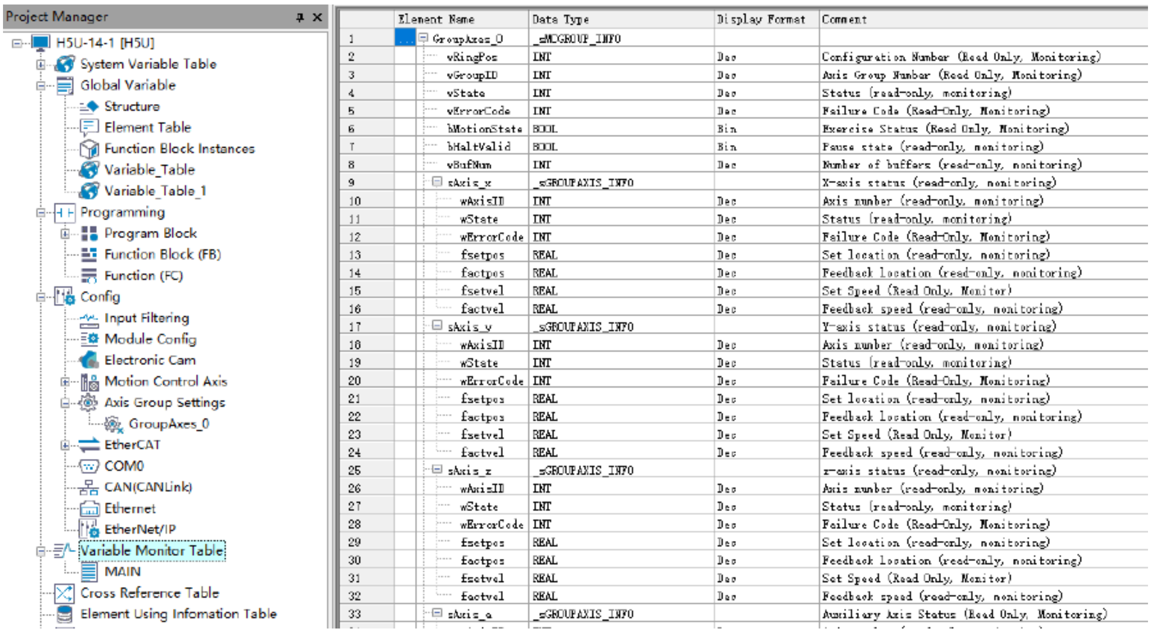
3.7.14 _sMCGROUP_INFO for Axis Group Status

Name	Type	Description
wRingPos	INT16	Axis group number
wGroupID	INT16	Axis number

Name	Type	Description
wState	INT16	Axis group status 0: Init The axis configuration in the axis group is not completed. 1: Disabled Not all axes in the axis group are enabled. 2: Single Stop An axis in the axis group calls the instruction MC_Gtop. 3: Single Homing An axis in the axis group calls the instruction MC_Home. 4: Single motion An axis in the axis group calls single-axis motion instructions such as MC_MoveAbsolute. 5: ErrorStop An axis in the axis group is in a fault state. 6: StandStill All axes in the axis group are in the StandStill state. 7: Stopping The instruction MC_GroupStop is called. 8: Synchronous Motion A linear interpolation or circular interpolation instruction is called.
wErrorCode	INT16	Fault code
bMotionState	BOOL	Motion status FALSE: Not in motion TRUE: In motion
bHaltValid	BOOL	Halt status FALSE: Halt not applied TRUE: Halt applied
wBufNum	INT16	The number of buffered curves
sAxis_x	_sGROUPAXIS_INFO	The status of the X-axis
sAxis_y	_sGROUPAXIS_INFO	The status of the Y-axis
sAxis_z	_sGROUPAXIS_INFO	The status of the Z-axis
sAxis_a	_sGROUPAXIS_INFO	The status of the auxiliary axis
fSetvel	REAL	Velocity reference In linear interpolation mode, it indicates the interpolation velocity of a space straight line. In circular interpolation mode, it indicates the linear velocity of a circular arc.
fSetacc_dec	REAL	Acceleration/deceleration reference Indicates the change rate of setvel.
fSetvel_buf	REAL	The velocity reference of a buffered curve In linear interpolation mode, it indicates the interpolation velocity of a space straight line. In circular interpolation mode, it indicates the linear velocity of a circular arc.

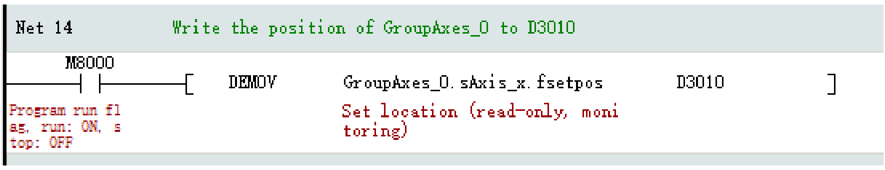
Name	Type	Description
fSetacc_dec_buf	REAL	The acceleration/deceleration reference of a buffered curve Indicates the change rate of fSetvel_buf
fSetdis	REAL	Distance reference In linear interpolation mode, it indicates the distance at which a space straight line moves after the instruction is executed. In circular interpolation mode, it indicates the length of a circular arc in which the circular arc moves after the instruction is executed.
fLeftdis	REAL	Left distance In linear interpolation mode, it indicates the left distance for this section of a space straight line after the instruction is executed. In circular interpolation mode, it indicates the length of a space circular arc left after the instruction is executed.
fCenter_x	REAL	The coordinates of point X at the center of a circular arc during circular interpolation
fCenter_y	REAL	The coordinates of point Y at the center of a circular arc during circular interpolation
fCenter_z	REAL	The coordinates of point Z at the center of a circular arc during circular interpolation
fRadius	REAL	The radius of a circular arc during circular interpolation
fStartAng	REAL	The start angle during circular interpolation
fSetAng	REAL	The motion angle during circular interpolation

This system variable is used to indicate the status of the entire axis group:



	Element Name	Data Type	Display Format	Comment
1	GroupAxes_0	_sMCGROUP_INFO		
2	vRingPos	INT	Dec	Configuration Number (Read Only, Monitoring)
3	vGroupID	INT	Dec	Axis Group Number (Read Only, Monitoring)
4	vState	INT	Dec	Status (read-only, monitoring)
5	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
6	bMotionState	BOOL	Bin	Exercise Status (Read Only, Monitoring)
7	bHaltValid	BOOL	Bin	Pause state (read-only, monitoring)
8	vBufNum	INT	Dec	Number of buffers (read-only, monitoring)
9	sAxis_x	_sGROUPAXIS_INFO		X-axis status (read-only, monitoring)
10	wAxisID	INT	Dec	Axis number (read-only, monitoring)
11	wState	INT	Dec	Status (read-only, monitoring)
12	wErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
13	fsetpos	REAL	Dec	Set location (read-only, monitoring)
14	factpos	REAL	Dec	Feedback location (read-only, monitoring)
15	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
16	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
17	sAxis_y	_sGROUPAXIS_INFO		Y-axis status (read-only, monitoring)
18	wAxisID	INT	Dec	Axis number (read-only, monitoring)
19	wState	INT	Dec	Status (read-only, monitoring)
20	wErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
21	fsetpos	REAL	Dec	Set location (read-only, monitoring)
22	factpos	REAL	Dec	Feedback location (read-only, monitoring)
23	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
24	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
25	sAxis_z	_sGROUPAXIS_INFO		z-axis status (read-only, monitoring)
26	wAxisID	INT	Dec	Axis number (read-only, monitoring)
27	wState	INT	Dec	Status (read-only, monitoring)
28	wErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
29	fsetpos	REAL	Dec	Set location (read-only, monitoring)
30	factpos	REAL	Dec	Feedback location (read-only, monitoring)
31	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
32	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
33	sAxis_a	_sGROUPAXIS_INFO		Auxiliary Axis Status (Read Only, Monitoring)

For example, write the X-axis coordinates of the center of an axis group to D3010:



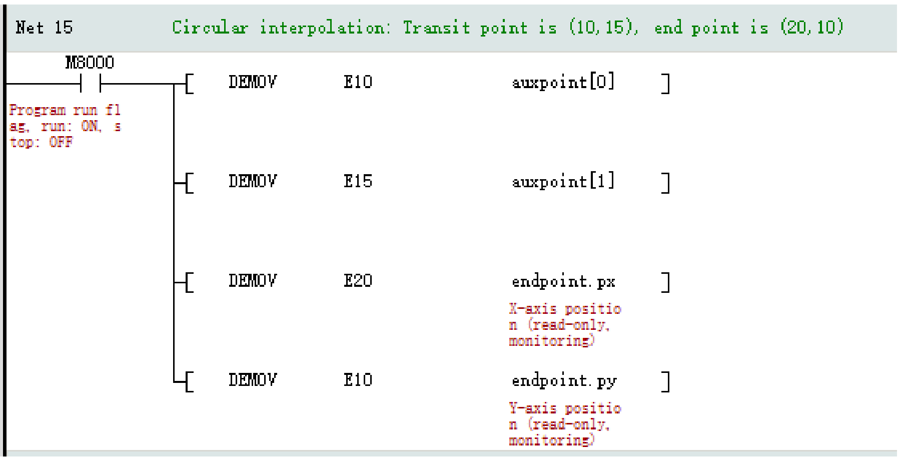
```
Net 14          Write the position of GroupAxes_0 to D3010

      M8000
  ----| |--------[  DEMOV      GroupAxes_0.sAxis_x.fsetpos      D3010      ]
Program run fl                 Set location (read-only, moni
ag, run: ON, s                 toring)
top: OFF
```

3.7.15 _sGROUPPOS_INFO for Target Positions of Coordinate Axes within Axis Group

Name	Type	Description
px	REAL	The position of the X-axis
py	REAL	The position of the Y-axis
pz	REAL	The position of the Z-axis
pa	REAL	The position of the auxiliary axis

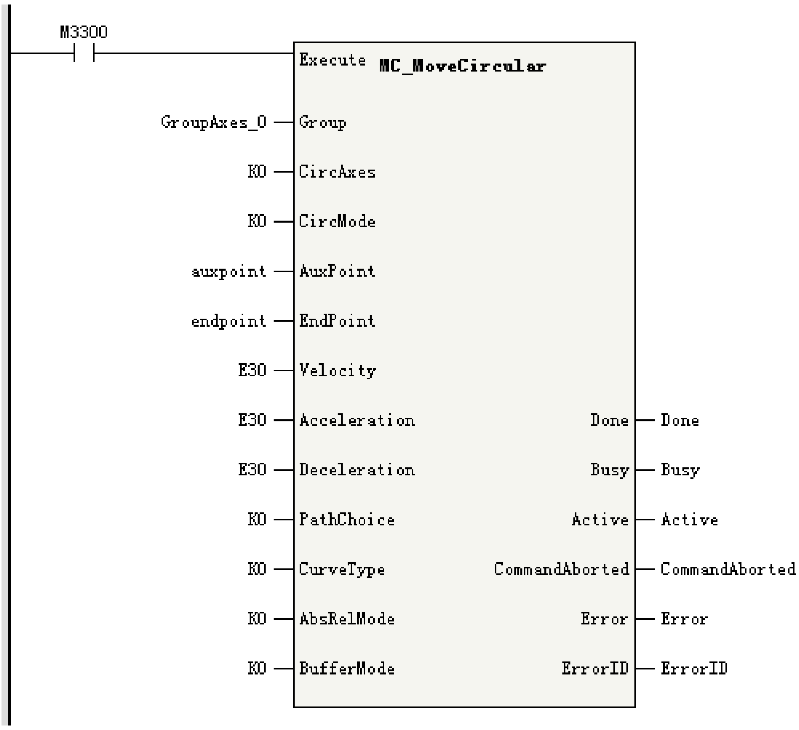
This structure sets the target position of a circular arc as an input parameter to the MC_MoveCircular.

1. Create a global variable
2. Assign values to the global variable



```
Net 15          Circular interpolation: Transit point is (10,15), end point is (20,10)
   M8000
───┤ ├───┬─[ DEMOV   E10     auxpoint[0]   ]
Program run fl
ag, run: ON, s
top: OFF
         ├─[ DEMOV   E15     auxpoint[1]   ]

         ├─[ DEMOV   E20     endpoint.px   ]
                             X-axis positio
                             n (read-only,
                             monitoring)
         └─[ DEMOV   E10     endpoint.py   ]
                             Y-axis positio
                             n (read-only,
                             monitoring)
```

3. Call the instruction MC_MoveCircular



```
   M3300
───┤ ├──────────── Execute   MC_MoveCircular
      GroupAxes_0 ─ Group
               K0 ─ CircAxes
               K0 ─ CircMode
         auxpoint ─ AuxPoint
         endpoint ─ EndPoint
              E30 ─ Velocity
              E30 ─ Acceleration                Done ─ Done
              E30 ─ Deceleration                Busy ─ Busy
               K0 ─ PathChoice                Active ─ Active
               K0 ─ CurveType       CommandAborted ─ CommandAborted
               K0 ─ AbsRelMode                 Error ─ Error
               K0 ─ BufferMode               ErrorID ─ ErrorID
```

3.8 Timer

3.8.1 Overview

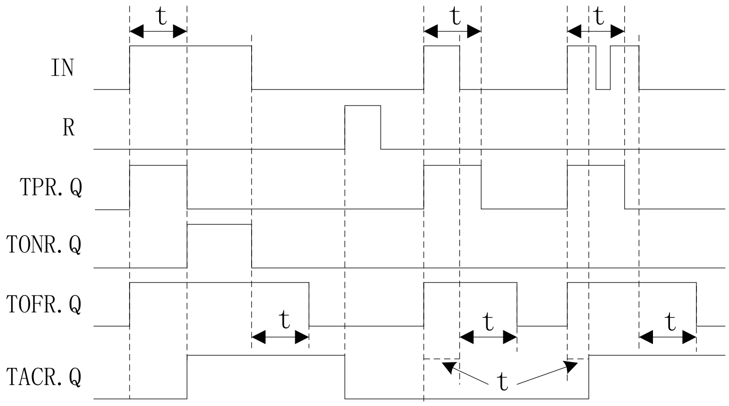
H5U supports four types of timers with the reset function, including the pulse timer (TPR), connection delay timer (TONR), off delay timer (TOFR), and accumulation timer. For details, see the IEC61131-3 standard.

The time base of the timers is 1 ms, and the timer count value and state are updated when the timer instruction is executed. The program supports a maximum of 4096 timer instructions. The instruction parameters of these four types of timers are the same, which are listed as follows:

Table 3–37 Timer instruction parameters

Name	Definition	Data Type	Description
IN	Instruction execution input	/	Start input
PT	Input variable	DINT	Delay time
R	Input variable	BOOL	Reset input
Q	Output variable	BOOL	Timer output
ET	Output variable	DINT	Current timing time

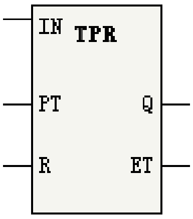
Timer timing



3.8.2 Pulse Timer - TPR

When the IN input of the timer instruction changes from OFF to ON, the timer starts timing and the output Q turns ON. At this time, no matter how the IN input flow changes, Q remains ON for the time period specified by PT. When the timing duration reaches the time period specified by PT, Q changes to OFF.

During timing of the timer, ET outputs the current timing duration. After the timing duration reaches the value specified by PT, if the IN input flow is ON, the ET value is retained; if the IN input flow is OFF, the ET value becomes 0.



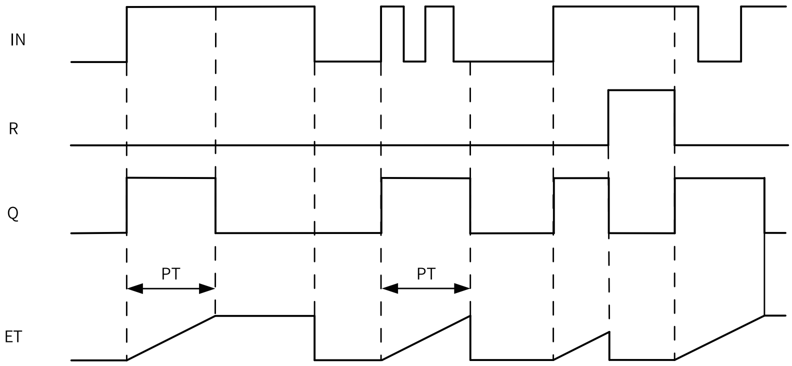
During timing, if the reset input R changes from OFF to ON, the timing duration of the TPR timer is reset to 0, and the output Q turns OFF. After the reset input R turns OFF, if the IN input flow is active, the timer resumes timing.

Parameter description:

PT ranges from 0 to 2147483647 ms (about 24 days). If the value of PT is less than or equal to 0, it is considered 0.

Timing diagram

The following figure shows the timing diagram of the parameters IN, R, Q, and ET.



Note

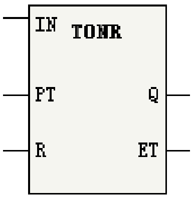
The output parameters "ET" and "Q" are updated when this instruction is executed. Therefore, the change in the state of "Q" is not at the time when the elapsed time after the timer starts equals "PT", but at the time when the instruction is executed for the first time after the elapsed time after the timer starts reaches "PT". That is, the delay of the output parameters can be up to one cycle.

3.8.3 Connection Delay Timer - TONR

When the IN input of the timer instruction changes from OFF to ON, the timer starts timing and the output Q turns ON. During the period when the IN input flow remains ON, the running time of the timer is the time specified by PT. After the timing duration reaches the time period specified by PT, Q turns ON. During the timing process or after timing is completed, when the IN input flow changes to OFF, timing ends and Q turns OFF.

When the IN input flow is ON, ET outputs the current timing duration during timing of the timer, and the ET value is retained after the timing duration reaches the value specified by PT. When the IN input flow is OFF, the ET value becomes 0.

During timing, if the reset input R changes from OFF to ON, the timing duration of the TONR timer is reset to 0, and the output Q turns OFF. After the reset input R turns OFF, to resume timer timing, you need to set the IN input flow to ON again.



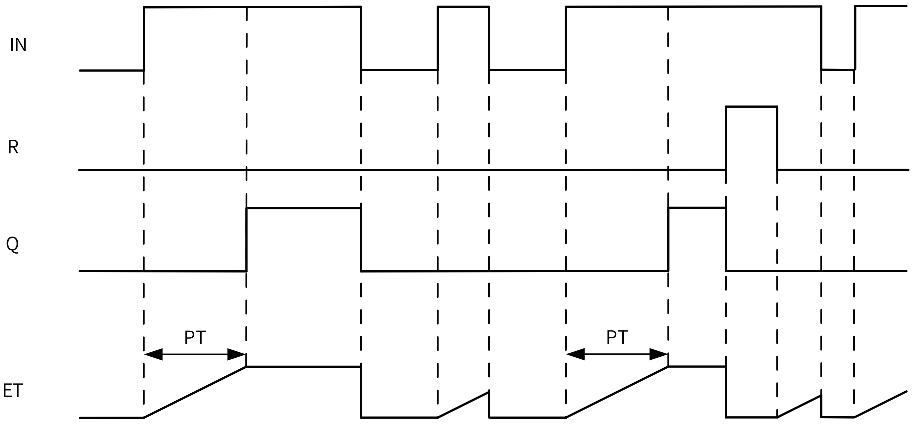
During timing, if the reset input R changes from OFF to ON, the timing duration of the TONR timer is reset to 0, and the output Q turns OFF. After the reset input R turns OFF, if the IN input flow is active, the timer resumes timing.

Parameter description:

PT ranges from 0 to 2147483647 ms (about 24 days). If the value of PT is less than or equal to 0, it is considered 0.

Timing diagram

The following figure shows the timing diagram of the parameters IN, R, Q, and ET.



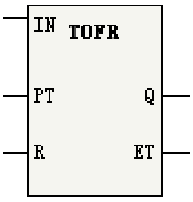
Note

The output parameters "ET" and "Q" are updated when this instruction is executed. Therefore, the change in the state of "Q" is not at the time when the elapsed time after the timer starts equals "PT", but at the time when the instruction is executed for the first time after the elapsed time after the timer starts reaches "PT". That is, the delay of the output parameters can be up to one cycle.

3.8.4 Off Delay Timer - TOFR

When the IN input of the timer instruction changes from OFF to ON, the timer starts timing and the output Q turns ON. When the IN input flow changes from ON to OFF, during the period when the IN input flow remains ON, the running time of the timer is the time specified by PT. After the timing duration reaches the time period specified by PT, Q turns OFF.

When the IN input flow is ON, the ET output is 0. When the IN input changes from ON to OFF, ET outputs the current timing duration during timing of the timer, and the ET value is retained after the timing duration reaches the value specified by PT.



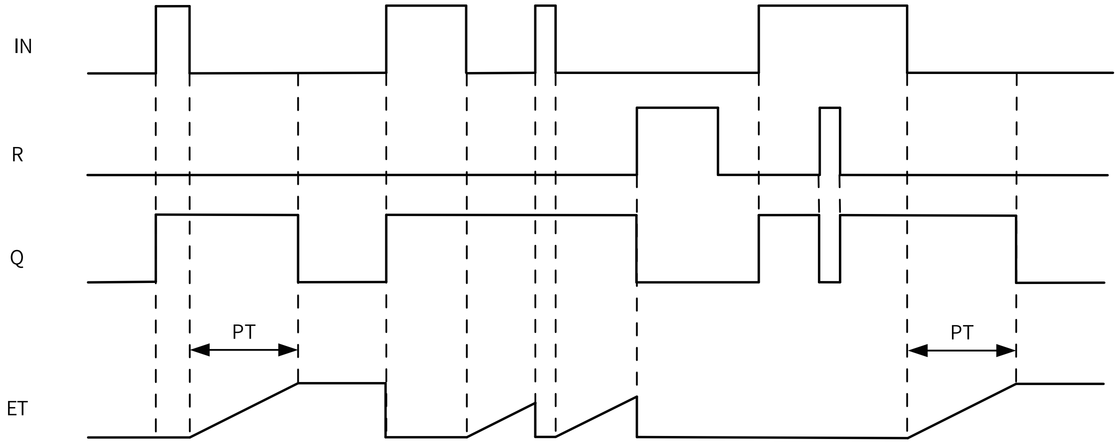
When the IN input flow is ON, if the reset input R changes from OFF to ON, the output Q turns OFF; if R resumes OFF, the output Q resumes ON. When the IN input flow changes from ON to OFF, if the reset input R changes from OFF to ON during the timing process or after timing is completed, the output Q turns OFF, and ET is reset to 0. After the reset input R turns OFF, to resume timer timing, you need to set the IN input flow to OFF again.

Parameter description:

PT ranges from 0 to 2147483647 ms (about 24 days). If the value of PT is less than or equal to 0, it is considered 0.

Timing diagram

The following figure shows the timing diagram of the parameters IN, R, Q, and ET.



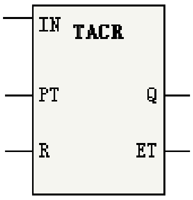
Note

The output parameters "ET" and "Q" are updated when this instruction is executed. Therefore, the change in the state of "Q" is not at the time when the elapsed time after the timer starts equals "PT", but at the time when the instruction is executed for the first time after the elapsed time after the timer starts reaches "PT". That is, the delay of the output parameters can be up to one cycle.

3.8.5 Accumulation Timer - TACR

When the IN input flow of the timer instruction is ON, if the timer value has not reached the time period specified by PT, the timer continues to count, and the output Q is OFF; when the timing duration reaches the time period specified by PT, Q turns ON. During the timing process, if IN changes from ON to OFF, the timing duration is retained. When IN turns ON again, the timer starts counting from the current retained value. After the time specified by PT is reached, Q becomes ON.

When the IN input flow is ON, ET outputs the current timing value. After the timing duration reaches the time period specified by PT, the ET value is retained. When the IN input flow turns OFF, ET remains unchanged.



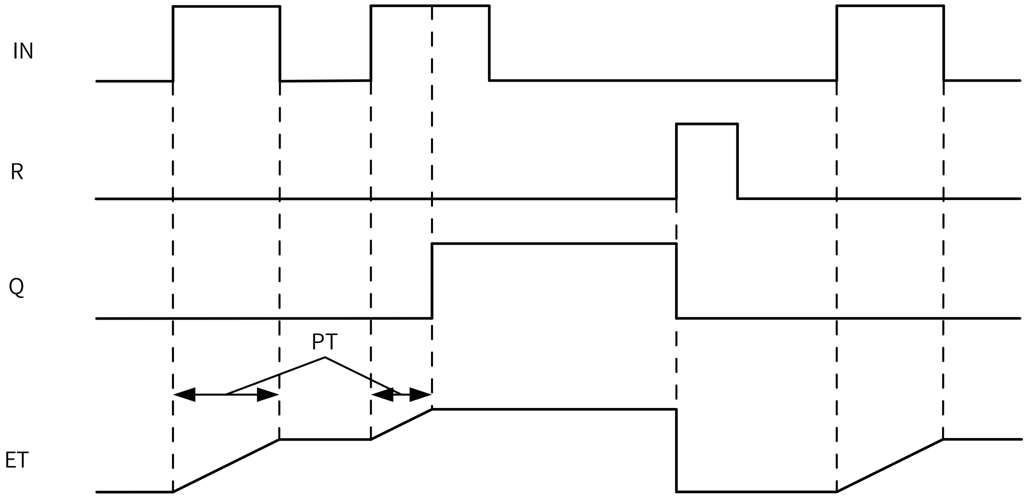
During the timing process or after timing is completed, if the reset input R changes from OFF to ON, the output Q turns OFF, and ET is reset to 0. After the reset input R turns OFF, if the IN input flow is active, the timer resumes timing.

Parameter description:

PT ranges from 0 to 2147483647 ms (about 24 days). If the value of PT is less than or equal to 0, it is considered 0.

Timing diagram

The following figure shows the timing diagram of the parameters IN, R, Q, and ET.



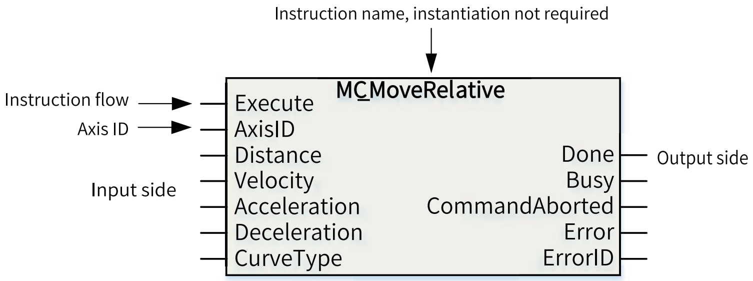
Note

The output parameters "ET" and "Q" are updated when this instruction is executed. Therefore, the change in the state of "Q" is not at the time when the elapsed time after the timer starts equals "PT", but at the time when the instruction is executed for the first time after the elapsed time after the timer starts reaches "PT". That is, the delay of the output parameters can be up to one cycle.

3.9 Graphical Block Instructions

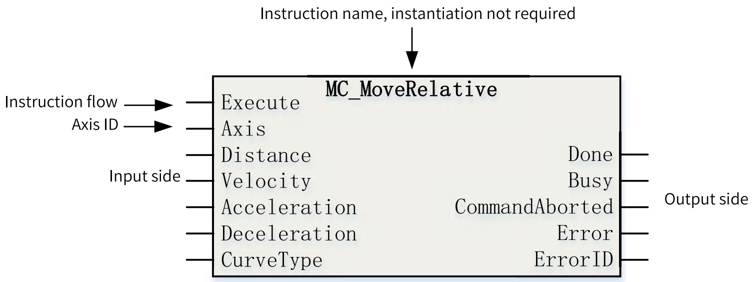
3.9.1 Instruction Composition

Some instructions support graphical block programming. An graphical block instruction is composed of the instruction name, flow signal, input side, and output side. The following figure shows the composition of a graphical block instruction of a motion control axis.



The floating-point numbers such as the target position and target velocity in the instructions are single-precision floating-point data. Therefore, the values in the instructions must meet the requirements of the range and precision of single-precision floating-point data when being processed in the PLC program. That is, a value should fall between –3.4E38 and +3.4E38, with a maximum of 7 significant digits. If a value has more than 7 significant digits, the excess part will be automatically rounded.

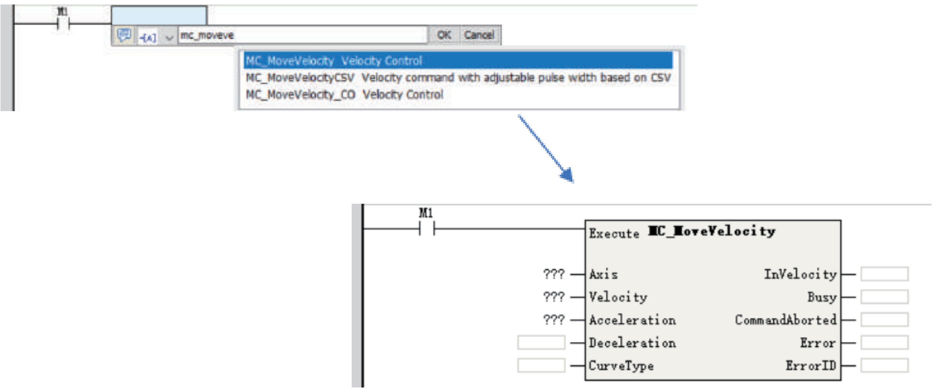
Since AutoShop 4.0.0.0 with PCB software 3.0.0.0, the motion control axis control instructions (EtherCAT/pulse output, pulse input) of graphical blocks support access by axis name. "AxisID" is changed to "Axis", and access by axis ID is still supported.



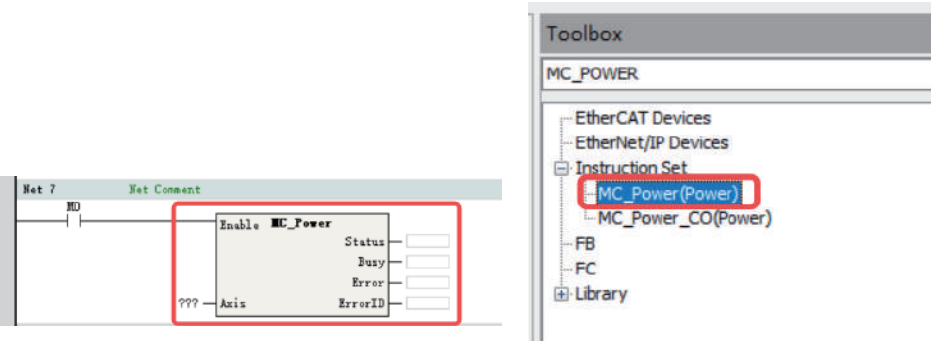
3.9.2 Programming

During programming, you only need to enter the name of a graphical block instruction and simply press the "Enter" key to add the graphical block instruction to the program network. You can also directly edit the instruction parameters.

- When editing a ladder diagram, enter an instruction name or select an instruction name according to the instruction prompt and click "OK". The graphical block instruction is added to the ladder diagram network.


M1
mc_moveve
OK
Cancel
MC_MoveVelocity Velocity Control
MC_MoveVelocityCSV Velocity command with adjustable pulse width based on CSV
MC_MoveVelocity_CO Velocity Control
M1
Execute MC_MoveVelocity
??? Axis
??? Velocity
??? Acceleration
Deceleration
CurveType
InVelocity
Busy
CommandAborted
Error
ErrorID

- Enter parameters in the graphical block instruction to complete editing of the graphical block instruction.
  In the instruction, parameters (with "???") next to ① are mandatory, and parameters next to ② are optional. If a parameter is not used, the default parameter value is used automatically in the instruction input, and the state cannot be obtained in the instruction output in the program or during monitoring and debugging.
- All instructions under "Instruction Set" in the "Toolbox" pane are in graphical block mode. During programming, you can directly double-click an instruction under "Instruction Set" to add the instruction to the current focus position of the ladder diagram.



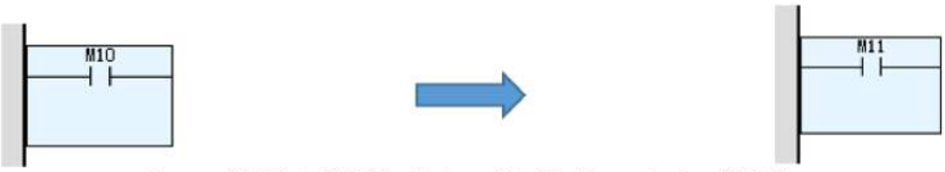
①: Double-click an instruction to add it to the ladder diagram. ②: The instruction is added successfully.

3.9.3 Labeling Function

Graphical blocks can be used to quickly increase or decrease label numbers and implement incremental paste.

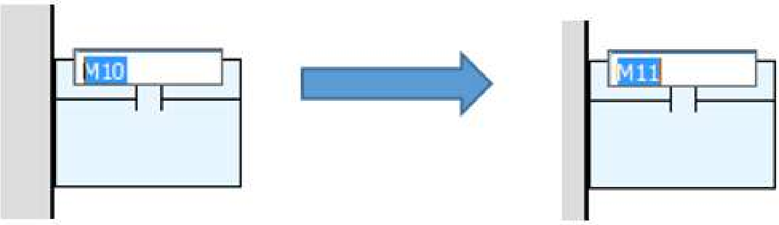
Quickly Increasing/Decreasing Label Numbers

When editing the ladder diagram, you can press "Alt"+"UP"/"DOWN" to quickly increase or decrease the label number of an element or array subscript.



Change “M10” to “M11” with keys Alt+UP after selecting “M10”.

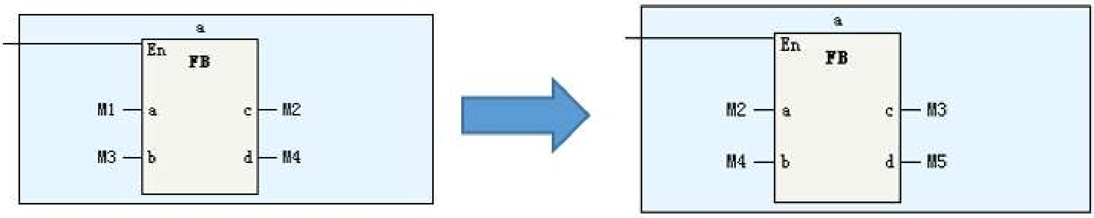
- This function can be used during command editing.



- For complex array variables, you can select the array subscript that needs to be increased or decreased.

Operate on the digit “7” through selecting the subscript of “A”.

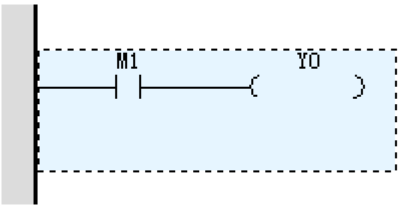
- When a function block is selected, the operation will be performed on all pins.



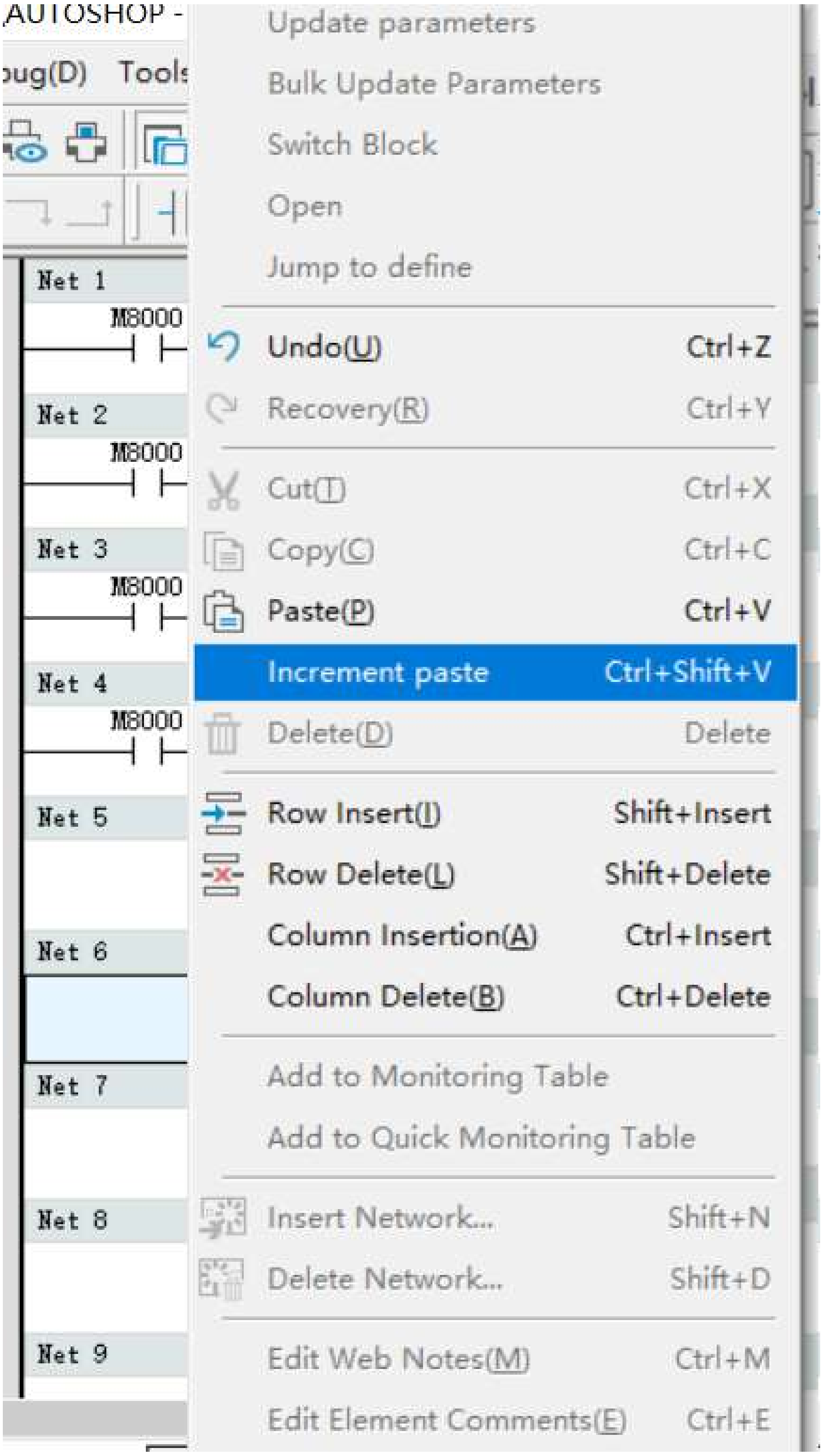
Incremental Paste

When editing the ladder diagram, you can use the incremental paste function to continuously paste the copied elements for multiple times. At the same time, the element number or array subscript can be specified during the process.

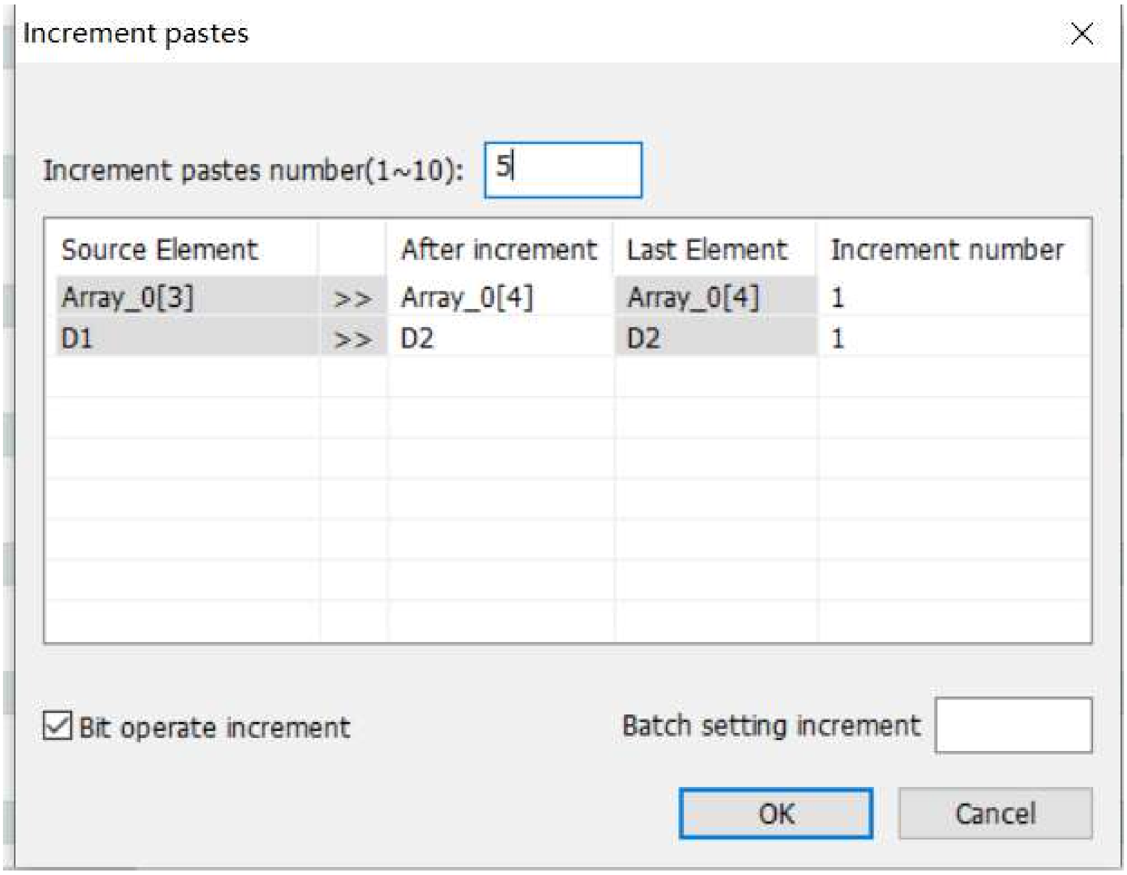
1. Select an element in the ladder diagram and press "Ctrl"+"C", or right-click the element and choose "Copy".



2. Right-click the destination position and choose "Increment paste" from the shortcut menu (or press "Ctrl"+"Shift"+"V").



3. Specify the increment value and paste times in the displayed configuration window.



Increment pastes number(1~10): 5

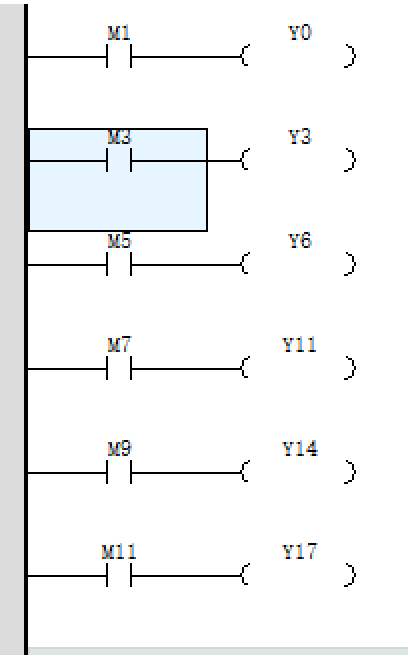
Source Element		After increment	Last Element	Increment number
Array_0[3]	>>	Array_0[4]	Array_0[4]	1
D1	>>	D2	D2	1

☑ Bit operate increment

Batch setting increment

- "Incremental pastes number (1–10)": You can set the paste times.
- "After increment": You can enter the expected value after increment, and "Increment number" is automatically calculated based on this value.
- "Increment number": You can set the increment in the target element each time a paste operation is performed.
- "Bit operate increment": During bit operation of an element, if this option is selected, the increment applies to the bit operation of the target element.
- "Batch setting increment": You can set the increments in batches.

4. Click "OK". The paste operation is performed based on the configuration.



3.10 Subprograms

3.10.1 Overview

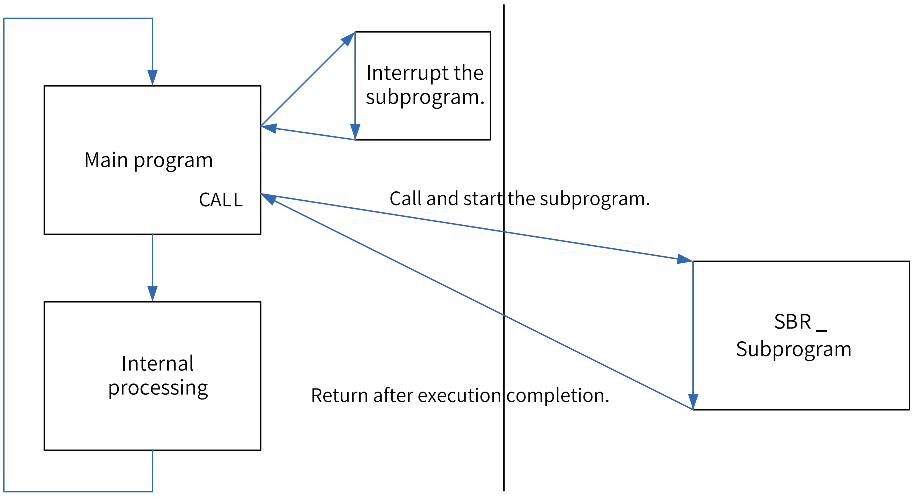
3.10.1.1 Subprogram Overview

The following table lists the subprogram categories and corresponding description.

Code	Name	Description
SBR	Subprogram	Up to 1024 subprograms are supported. Subprograms can be set as common subprograms or encrypted subprograms. Common subprograms and encrypted subprograms have infinite capacity and share the system capacity of 200,000 steps.
INT	Interrupt subprogram	External interruption: X000 to X003 input interrupt, including the rising edge, falling edge, and rising and falling edges
		Timed interruption: 4 points (time base = 1 ms)
		Comparison interruption: 16 points ranging from 1 to 16

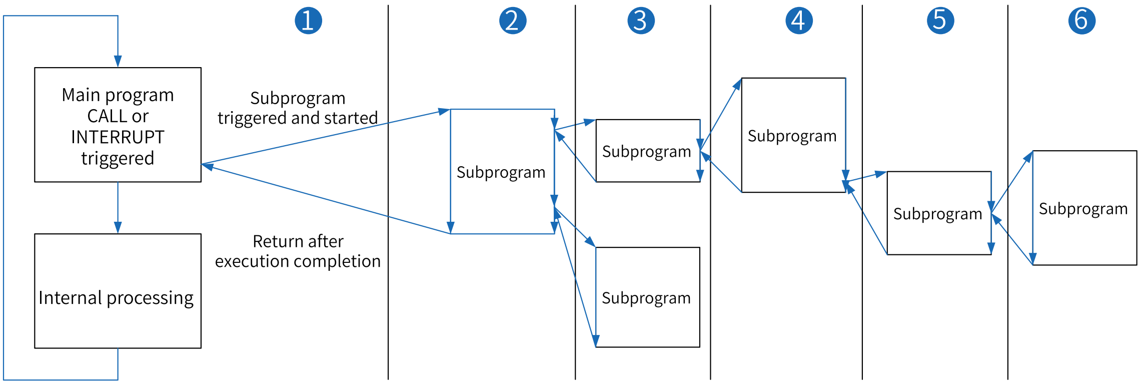
3.10.1.2 Subprogram Execution Mechanism

The following figure shows the execution logic and circular scanning methods of the main program and subprograms.


Main program
CALL
Interrupt the subprogram.
Call and start the subprogram.
SBR _ Subprogram
Internal processing
Return after execution completion.

Subprogram nesting levels

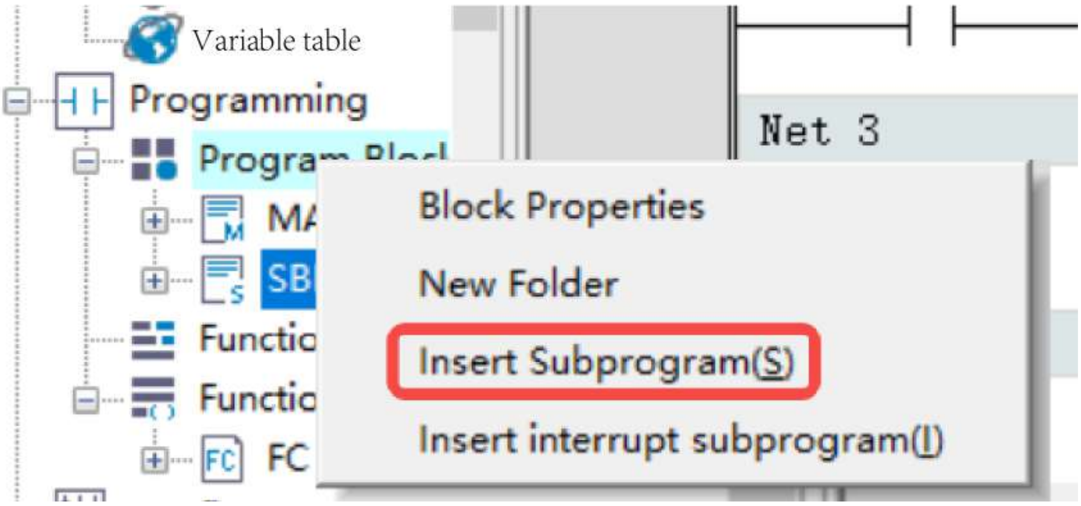
A subprogram supports up to six nesting levels. The main program calls the subprogram as level 1. The nesting level increases by 1 upon each call. If the nesting is returned, the nesting level does not increase. The following figure shows the details.


1
2
3
4
5
6
Main program CALL or INTERRUPT triggered
Subprogram triggered and started
Return after execution completion
Internal processing
Subprogram
Subprogram
Subprogram
Subprogram
Subprogram
Subprogram

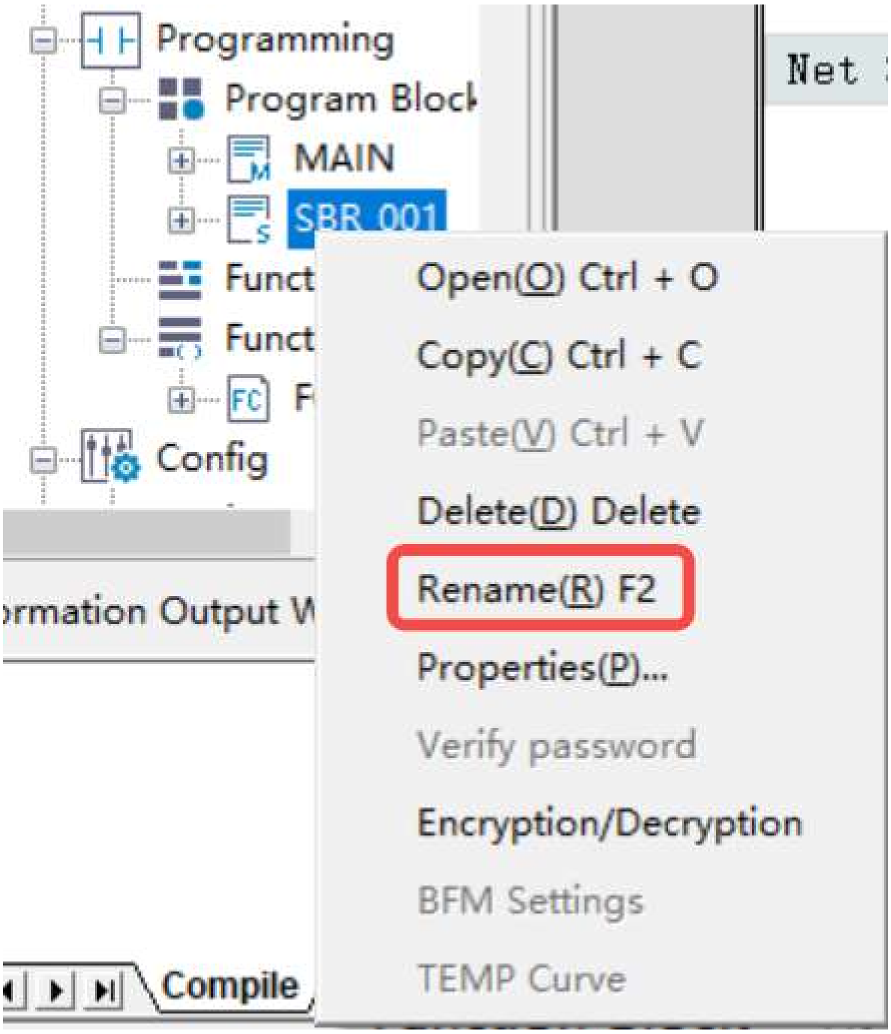
3.10.2 General Subprogram Application

3.10.2.1 Creating a General Subprogram

In the "Project Manager" tree, unfold "Programming", right-click "Function block" or a folder under "Function block", and select "Insert Subprogram". The new subprogram is displayed under "Function block".

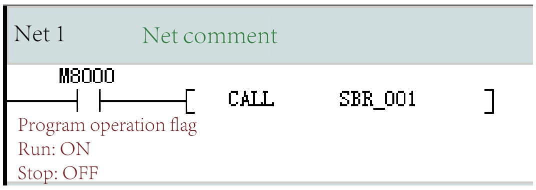


Subprogram naming rule: SBR_SN, in which the subprogram SN can be changed during renaming or property modification.



3.10.2.2 Calling a General Subprogram

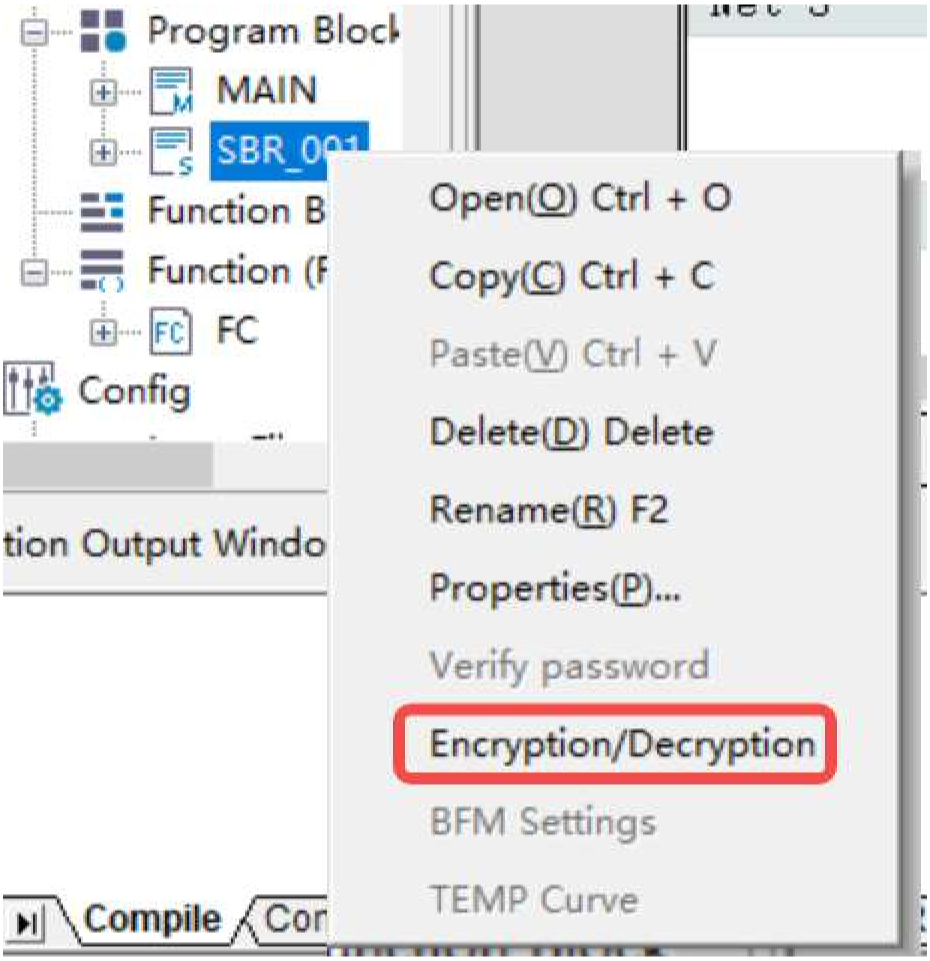
The following figure shows how to call a general subprogram.



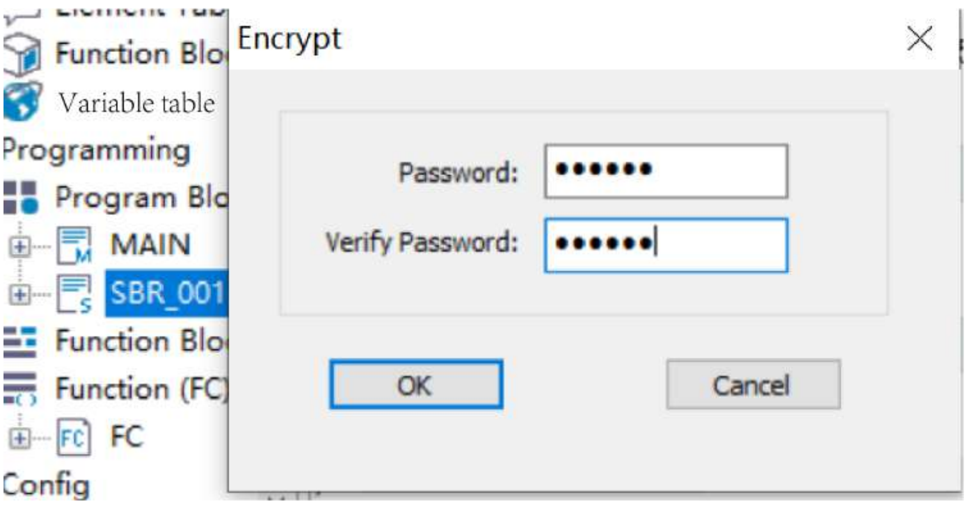
3.10.3 Encrypted Subprogram Application

3.10.3.1 Encrypting a General Subprogram

1. Encrypt the SBR_001 general subprogram as an example. Right-click SBR_001 and select "Encryption/Decryption".



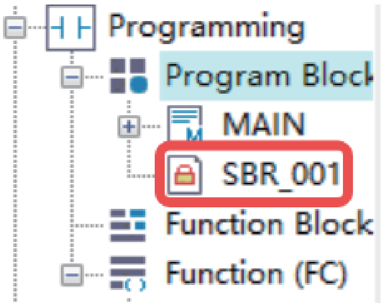
2. In the "Encrypt" dialog box that is displayed, set "Password" and "Verify Password".



Password: ••••••

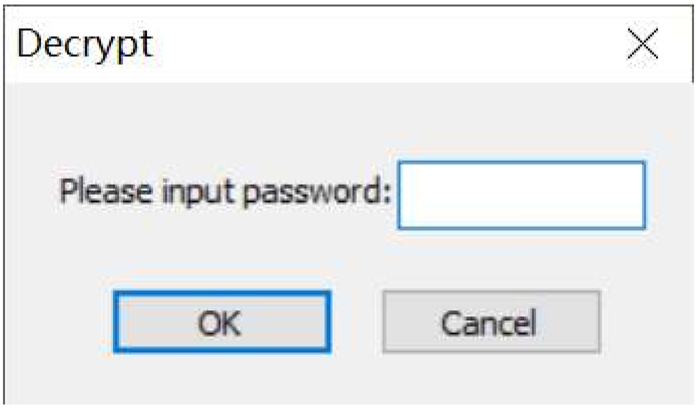
Verify Password: ••••••

After encryption, the SBR_001 general subprogram is shown in the following figure.



If you repeat the preceding operations on the encrypted general subprogram, the subprogram will be decrypted.

To access an encrypted general subprogram, you can double-click it or right-click it and select "Verify password". In the pop-up dialog box, input the correct password.



Please input password:

3.10.3.2 Calling an Encrypted Subprogram

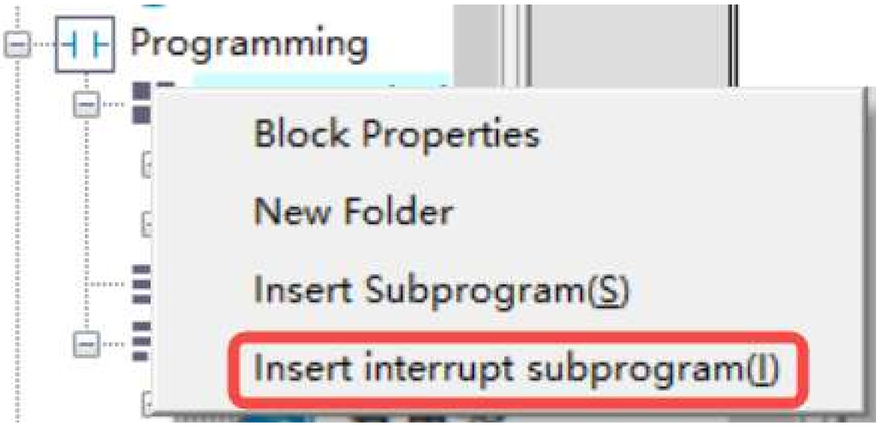
The method of calling an encrypted subprogram is the same as that of calling a general subprogram.

3.10.4 Interrupt Subprogram Application

3.10.4.1 External Interrupt Subprogram

External interrupt subprograms must be immediately executed to respond to external input signals. External interrupt subprograms are executed regardless of scan cycles.

1. In the "Project Manager" tree, unfold "Programming", right-click "POU" or a folder under "POU", and select "Insert interrupt subprogram".

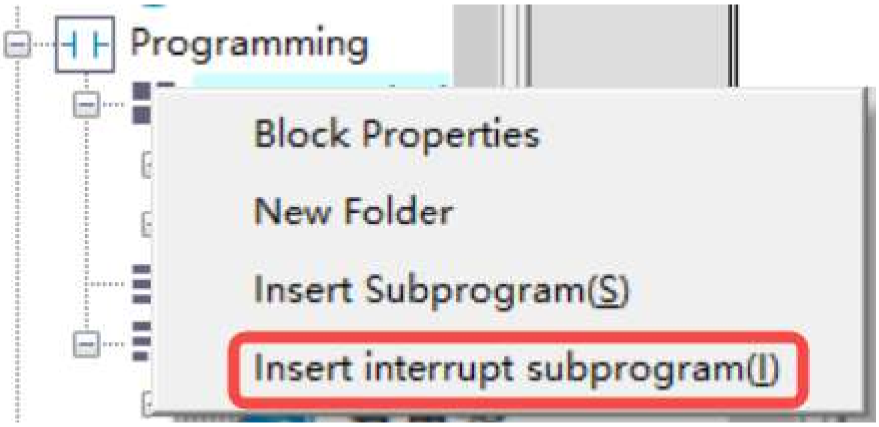


2. Right-click the inserted interrupt subprogram (such as INT_001 in the figure above) and select "Properties" to open the interrupt subprogram settings page as shown in the following figure.
3. Click ... next to the "Interrupt Event" field to open the interrupt selection page.
4. Select an external interrupt, such as X0 input interrupt, and then select the corresponding property, such as "Rising Edge", "Falling Edge", and "Rising Edge And Falling Edge".
5. Write interrupt subprograms in INT_001.
6. Enable EI in the main program. When the external interrupt conditions are met, the corresponding interrupt subprogram will be executed.

3.10.4.2 Timed Interrupt Subprogram

Timed interrupt subprograms apply to scheduled execution of set program blocks. Timed interrupt subprograms are executed regardless of scan cycles.

1. In the "Project Manager" tree, unfold "Programming", right-click "POU" or a folder under "POU", and select "Insert interrupt subprogram".



2. Right-click the inserted interrupt subprogram (such as INT_001 in the figure above) and select "Properties" to open the interrupt subprogram settings page as shown in the following figure.
3. Click ... next to the "Interrupt Event" field to open the interrupt selection page.
4. Select a timed interrupt and set Timing (ms) to a value ranging from 1 ms to 1000 ms.
5. Write interrupt subprograms in INT_001.
6. Enable EI in the main program. When the timed interrupt conditions are met, the corresponding interrupt subprogram will be executed.

3.10.4.3 Comparison Interrupt Subprogram

Comparison interrupt subprograms must be immediately executed to respond to the setpoint of the counter axis. Comparison interrupt subprograms are executed regardless of scan cycles.
For details about the procedure, see *“13.6.7 Comparison Interruption” on page 446*.

3.11 Function Blocks and Functions (FB/FC)

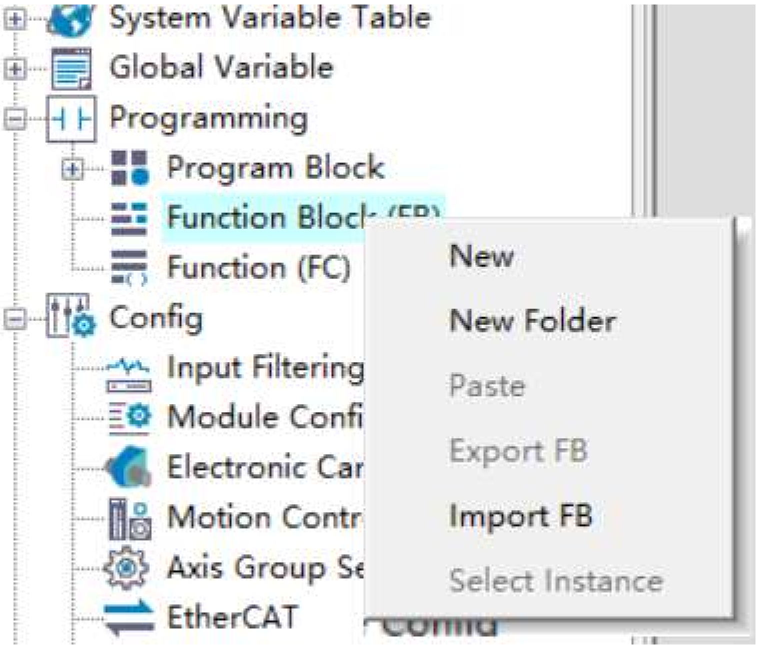
3.11.1 Function Blocks (FB)

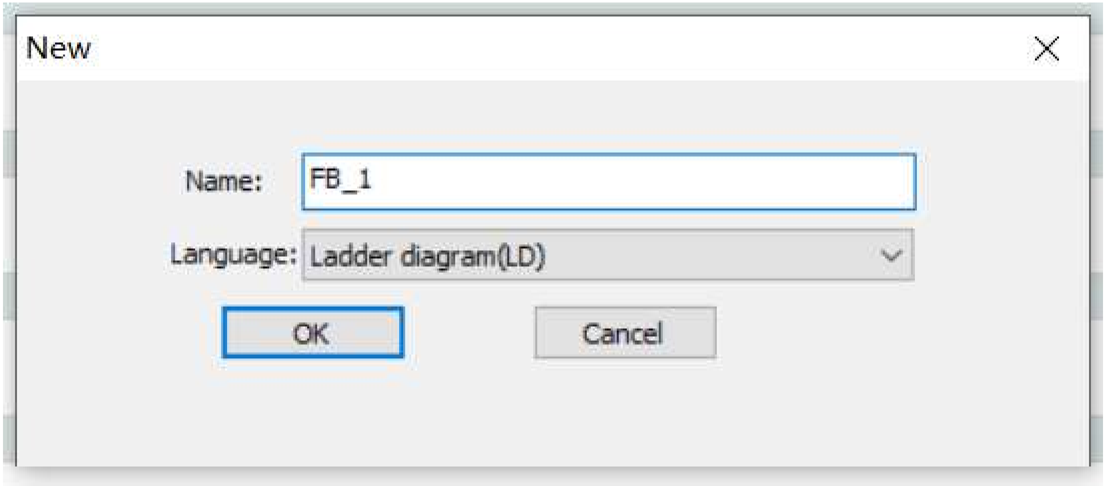
A function block (FB) abstractly encapsulates the part used repeatedly in a program into a general program block that can be called repeatedly within the program. Using encapsulated function blocks in programming can improve program development efficiency, reduce programming errors, and improve program quality.
Different instances can be created based on the same function block. These instances can output one or more values during execution. The system allocates memory for internal variables of each instance, and these variables describe the running state of the function block. With the same input parameters, different instances provide different calculation results.

The basic steps of using a function block are as follows: Create a function block -> Program the function block -> Instantiate the function block -> Run the function block -> Encapsulate the function block -> Import the function block.

Creating a Function Block

Expand the "Programming" node in the project management window, right-click "Function Block (FB)", or right-click a folder under "Function Block (FB)", choose "New", enter a name in the displayed dialog box, and click "OK". A function block is created successfully.



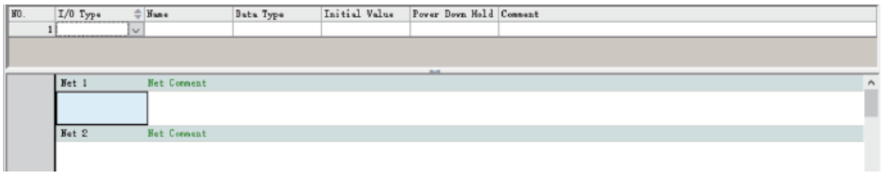


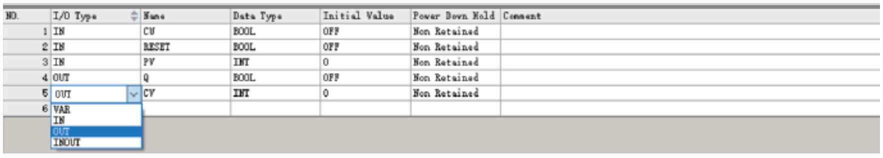
Name: FB_1

Language: Ladder diagram(LD)

Programming the Function Block

Function blocks can be programmed in the ladder diagram or structured text. Double-click the created function block under "Function Block (FB)" to go to the function block program editing interface. Compared with ordinary program editing, the function block program editing interface has an additional input/output and local variable definition window.





NO.	I/O Type	Name	Data Type	Initial Value	Power Down Hold	Comment
1	IN	CU	BOOL	OFF	Non Retained	
2	IN	RESET	BOOL	OFF	Non Retained	
3	IN	PV	INT	0	Non Retained	
4	OUT	Q	BOOL	OFF	Non Retained	
5	OUT	CV	INT	0	Non Retained	
6						

①: Input/output and local variable definition window

1. "I/O Type": attribute of the function block variable

Variable Type	Type Description	Description
IN	Input variable	The parameter is provided by the logic block that calls the variable, and the input is transferred to the instruction of the logic block.
OUT	Output variable	The parameter is provided to the logic block that calls the variable, that is, structure data is output from the logic block.
INOUT	Input/Output variable	An input/output variable can not only be transferred to the called logic block, but also can be modified inside the called logic block.
VAR	Local variable	A local variable is only valid in the current logic block and cannot be accessed externally.

2. "Name": name of the variable
3. "Data Type"

   The supported data types include BOOL, INT, DINT, REAL, BYTE, IP, and STRING. You can also define array variables and structure variables. To use structure variables, you need to create structure members in the structure of global variables.

4. "Initial Value"

   You can set the initial value of a variable when execution starts.

5. "Power Down Hold"

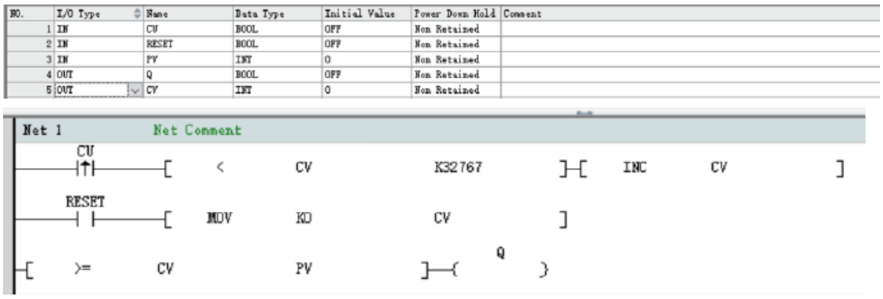
   This attribute allows you to choose whether to retain the value of a variable upon power failure.

   - "Non Retained": The variable resumes the specified initial value after power-on.
   - "Retained": If you select "Re-initialize retentive variables when downloading", the variable resumes the specified initial value during program downloading; otherwise, it retains the previous value.

The function block program adopts ladder diagram programming. It can call functions (FC) or function blocks (FB) and supports up to 8 levels of nested calls.

In addition to variables, the function block program can also use supported elements, such as M8000, as global variables.

Example: Counting Up with FB Encapsulation

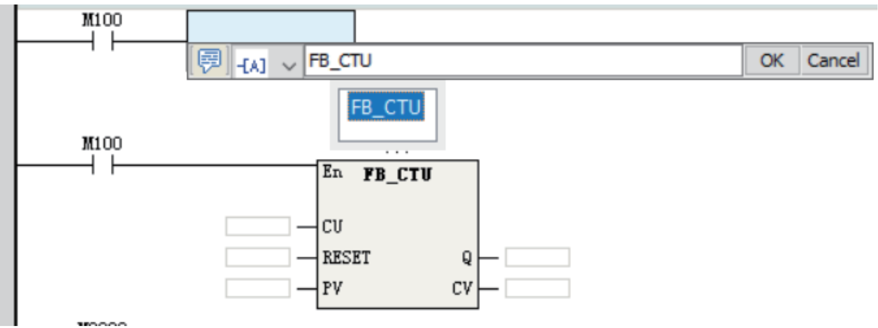


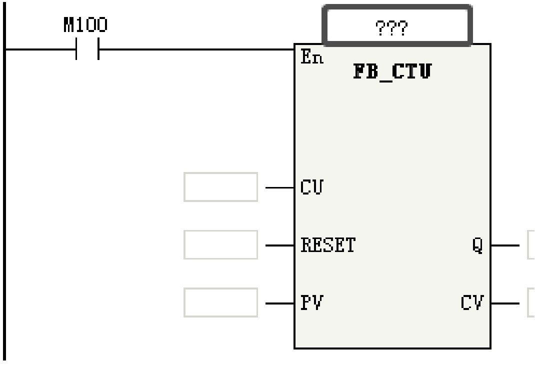
NO.	I/O Type	Name	Data Type	Initial Value	Power Down Hold	Comment
1	IN	CU	BOOL	OFF	Non Retained	
2	IN	RESET	BOOL	OFF	Non Retained	
3	IN	PV	INT	0	Non Retained	
4	OUT	Q	BOOL	OFF	Non Retained	
5	OUT	CV	INT	0	Non Retained	

Instantiating and Calling the Function Block

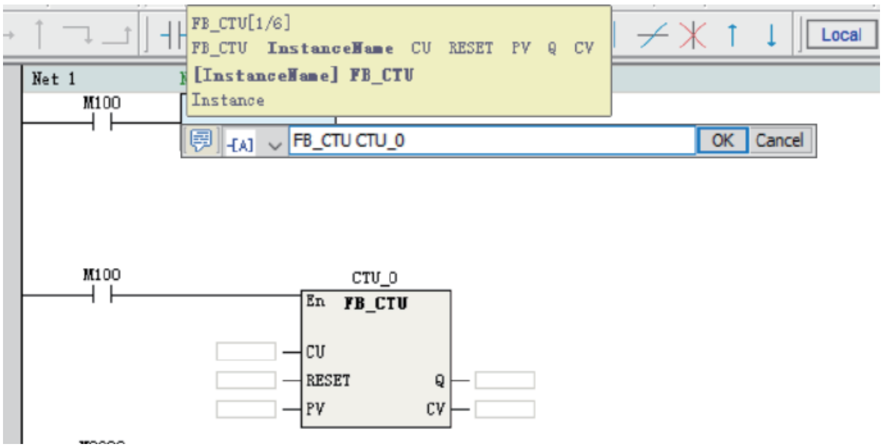
After the FB program is compiled, the function block needs to be instantiated.

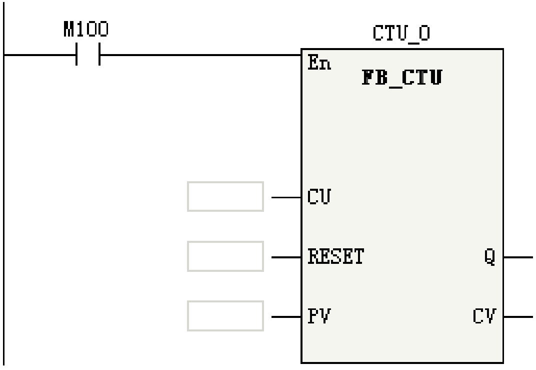
- Method 1: Directly enter the FB name in the ladder diagram application, and then enter the instance name in "???" at the top of the function block instruction to instantiate the function block.





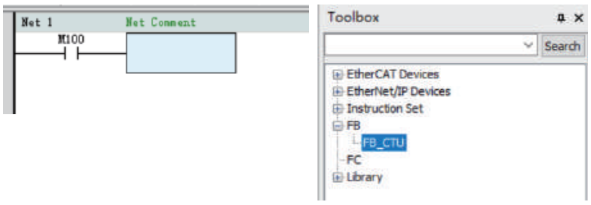
- Method 2: Directly enter the FB name+Instance name in the ladder diagram application and click "OK" to instantiate the function block.





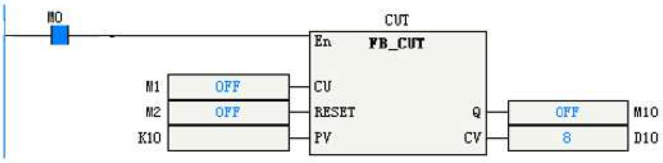
After instantiation is completed, edit the instruction parameters in the FB instruction as required by the program to call the instantiated function block.

- Method 3: Double-click the FB instruction under "FB" of the "Toolbox" pane to add the FB instruction to the selected position in the ladder diagram. Then enter the instance name in the graphic block instruction to complete the instantiation definition.

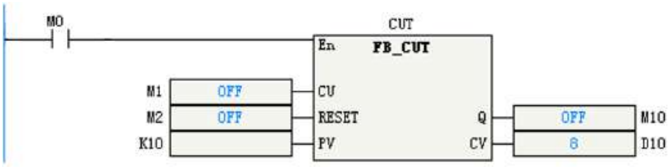


Running the Function Block

After the function block is instantiated, the En of the function block is connected to the ladder network. When the En network flow is ON, the function block program is executed, and the output of the function block changes with the input state and internal variable state. When the En network flow is OFF, the function block program is not executed, and output of the function block is not refreshed.



When the counter function block CUT flow is ON, the function block is executed. The output CV increases by 1 when the input condition CU changes on the rising edge.

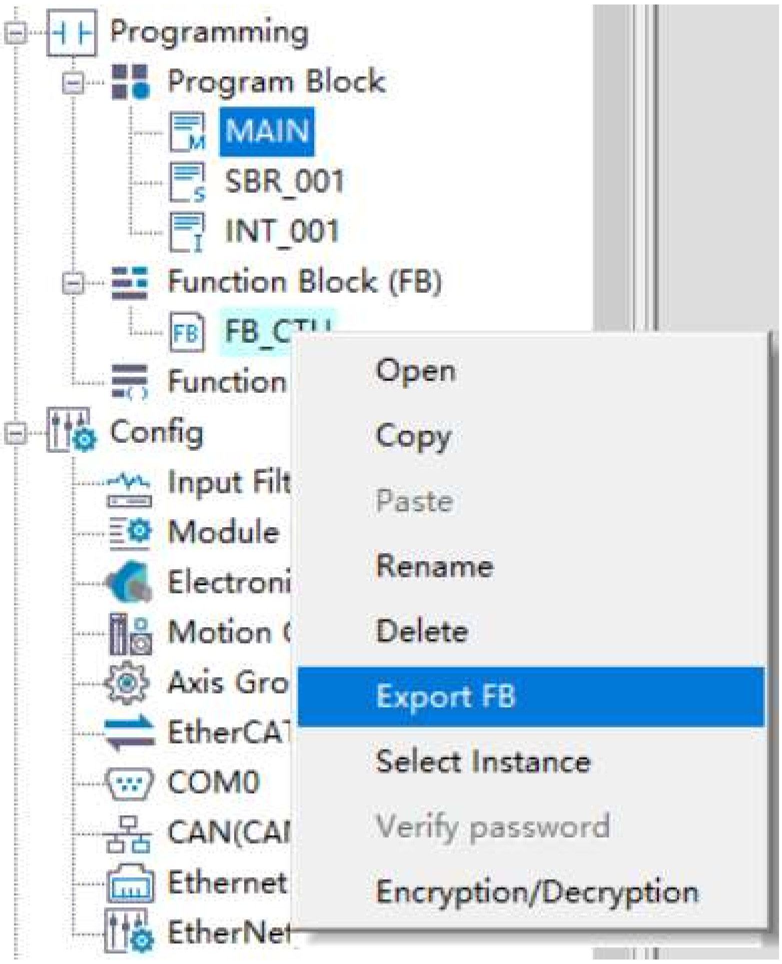


When the counter function block CUT flow is OFF, the function block is not executed. The output CV is not refreshed when the input condition CU changes on the rising edge.

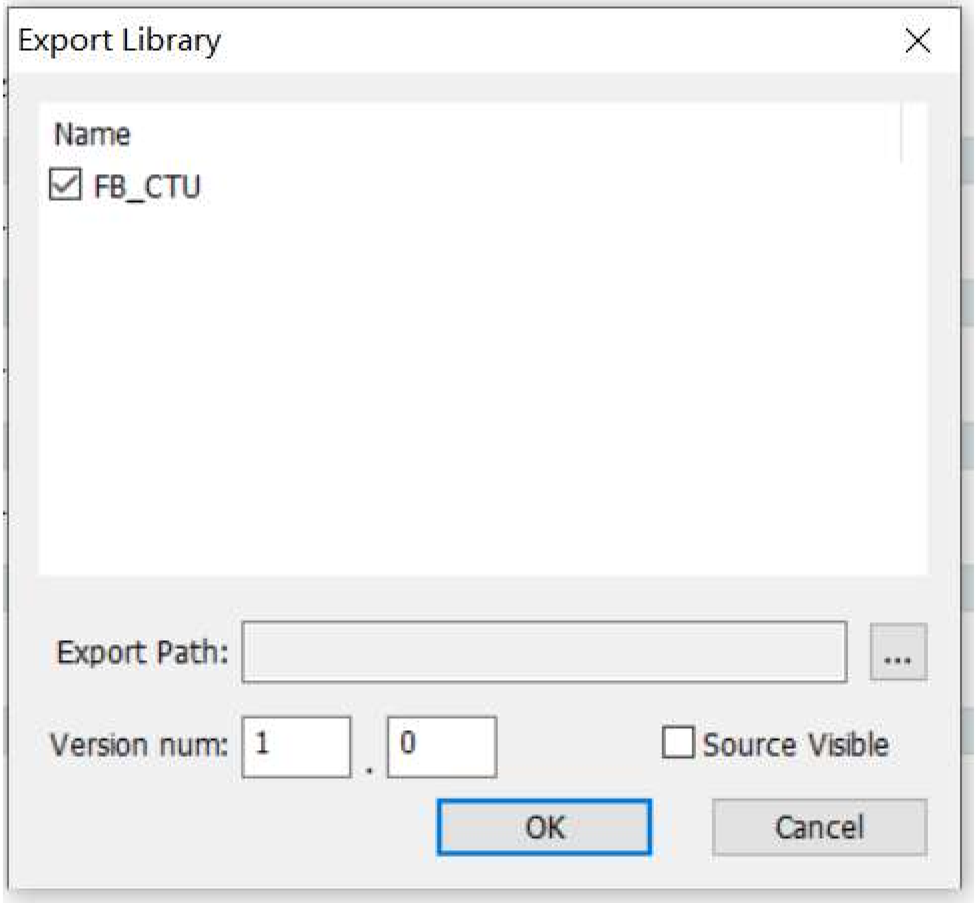
Encapsulating the Function Block

The function block can be encapsulated into a library after editing and debugging. The function block encapsulated into a library can be multiplexed in different programs through library management of AutoShop.

1. Right-click "Function Block (FB)" under "Programming" and choose "Export FB".



2. Select the function block to be encapsulated and set the version in the displayed "Export Library" window. Select "Source Visible" as required. If the source code is visible, after importing the library in the project, you can debug or modify the function block program. If the source code is invisible, after the library is imported, the function block program can only be called but not viewed or modified in the project.



Export Library

Name

☑ FB_CTU

Export Path:

Version num: 1 . 0

□ Source Visible

3. Specify "Export Path" and click "OK". The FB is exported to the specified location, and a function block in .fe format is generated.

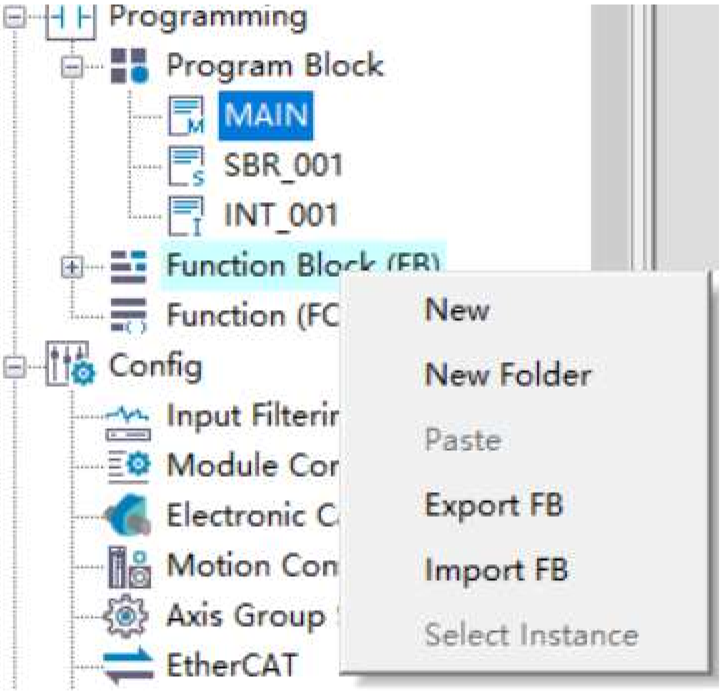
Caution

Encrypted function blocks, function blocks that call encrypted function blocks, and function blocks that call encrypted functions cannot be selected for export.

Importing the Function Block

After the function block is exported as a library, it can be called in other programs after being imported. You can import the function block library in either of the following two ways.

- Method 1: Right-click "Function Block (FB)" under "Programming" in the project management window and choose "Import FB" to import the library.



This method can only be used to import function blocks of which the source code is visible. After importing, you can double-click to open the function block program and edit and debug it. The function block library imported using this method is managed in the project. If you want to call the function block in a new project, you need to re-import the library.

- Method 2: Right-click "Library" in the "Toolbox" pane and choose "Import FB" to import the library. This method can be used to import function blocks of which the source code is visible or invisible. The libraries imported this way are managed as custom libraries, and the function blocks in the libraries can be used directly when a new project is created. You can double-click the function block library imported in the toolbox to directly add it to the ladder diagram program as an instruction. If you need to view or modify a function block program of which the source code is visible, you need to import it in the project management window.

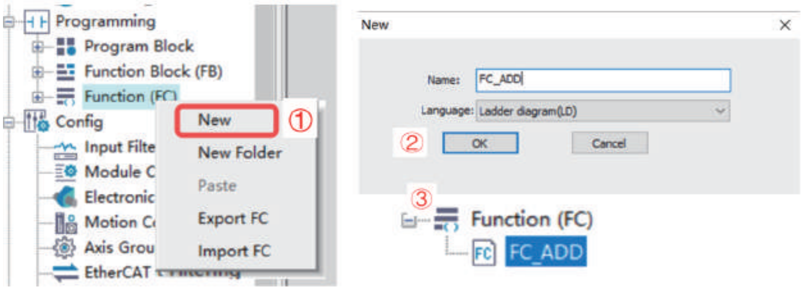
3.11.2 Functions (FC)

A function (FC) is an independently encapsulated program block. The program block can define input/output parameters and non-static internal variables. That is, when a function is called with the same input parameters, the output results are the same. An important feature of a function is that its internal variables are static, and there is no internal state storage. You will obtain the same output with the same input parameters. This is the main difference between a function and a function block.
FC, as a basic arithmetic unit, is often used in various mathematical operations. For example, sin(x) and sqrt(x) are typical functions.

The basic steps of using a function are as follows: Create a function -> Program the function -> Call the function -> Run the function -> Encapsulate the function.

Creating a Function

Expand the "Programming" node in the project management window, right-click "Function (FC)", or right-click a folder under "Function (FC)", choose "New", enter a name in the displayed dialog box, and click "OK". A function is created successfully.

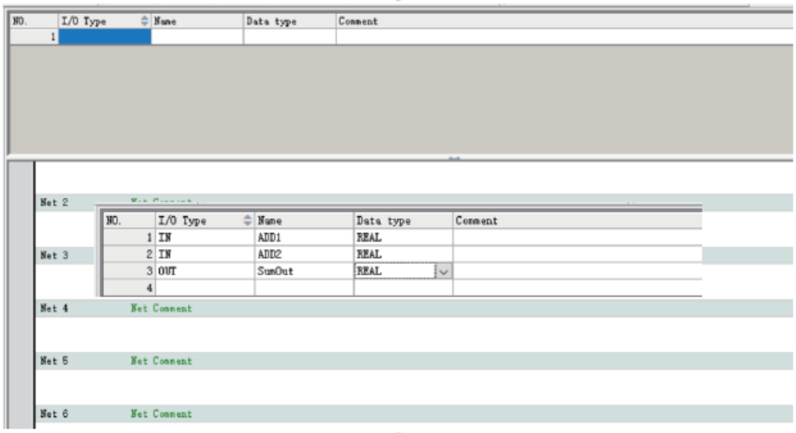


Name: FC_ADD

Language: Ladder diagram(LD)

Programming the Function

Functions can be programmed only in the ladder diagram. Double-click the created function under "Function (FC)" to go to the function program editing interface. The editing interface of the function program is similar to that of the function block. Compared with ordinary program editing, the function program editing interface has an additional input/output and local variable definition window.



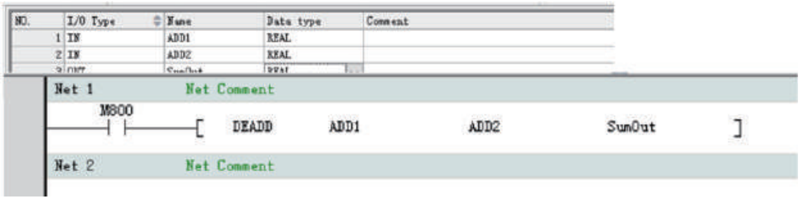
NO.	I/O Type	Name	Data type	Comment
1				

NO.	I/O Type	Name	Data type	Comment
1	IN	ADD1	REAL	
2	IN	ADD2	REAL	
3	OUT	SumOut	REAL	
4				

In the input/output and local variable definition window, you can define the input (IN), output (OUT), input/output (INOUT), and local variable (VAR) of a function block. The supported data types include BOOL, INT, DINT, REAL, BYTE, IP, and STRING. You can also define array variables and structure variables. To use structure variables, you need to create structure members in the structure of global variables.

- Compared with variables of function blocks, variables of functions do not support configuration of initial values, and all local variables are non-retentive.
- The function program adopts ladder diagram programming. It can call functions. A function can be called by other functions, function blocks, and programs.
- In addition to variables, the function program can also use M8000 as an always ON variable.
- In a function program, instructions related to states or executed for multiple cycles, such as LDP and MC_Power, cannot be used.

Example: Encapsulating the Addition Function

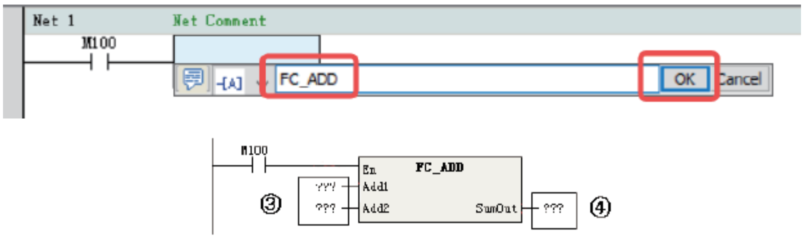


NO.	I/O Type	Name	Data type	Comment
1	IN	ADD1	REAL	
2	IN	ADD2	REAL	
3	OUT	SunOut	REAL	

Calling the Function

The function program can be called directly or used in an application after it is compiled.

- Method 1: Directly enter the function name in the ladder diagram application, press "Enter", and then edit the input/output parameters in the graphic block instruction.

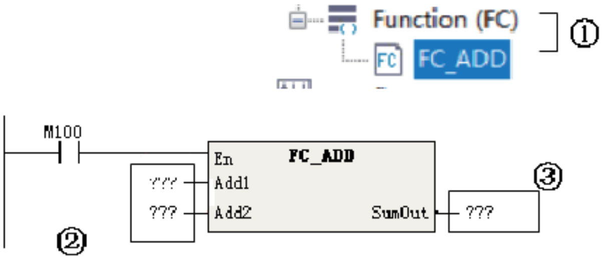


①: Enter the function name.

②: Click "OK".

③/④: Add input/output variables.

- Method 2: After a function program is created, the corresponding instruction is generated under "FC" in the "Toolbox" pane. Double-click the FC instruction under "FC" to add the FC instruction to the selected position in the ladder diagram.



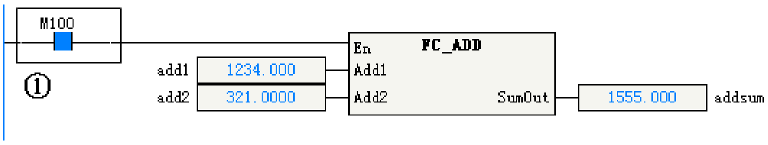
①: Double-click the FC instruction to add it.

②: Add input parameters.

③: Add output parameters.

Running the Function

After the function is called, the En of the function is connected to the ladder network. When the En network flow is ON, the function program is executed, and the output of the function is refreshed according to the input state operation. When the En network flow is OFF, the function program is not executed, and output of the function is not refreshed.



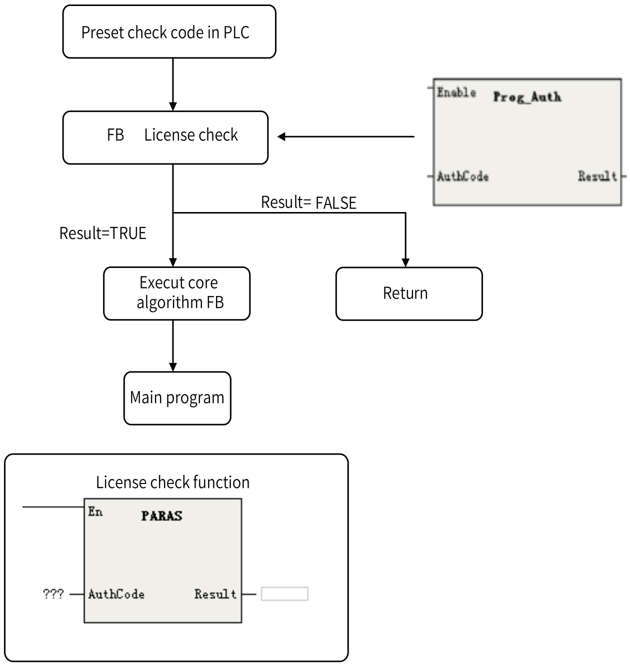
①: The function is executed when the En network flow is ON.

Encapsulating the Function

The encapsulation procedure of functions is similar to that of function blocks. For details, see the description of "Encapsulating the Function Block".

3.11.3 Authorization Function Block

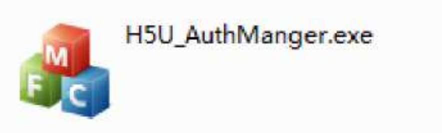
By using the Prog_Auth function, the core algorithm function block is controlled and compiled into a library file. Only authorized PLCs that pass the verification can use this library file, thus protecting the intellectual property of the equipment manufacturer.



1 - Check code
2 - Results returned (BOOL type); ON: Succeed; OFF: Failed

Setting Authorization Code

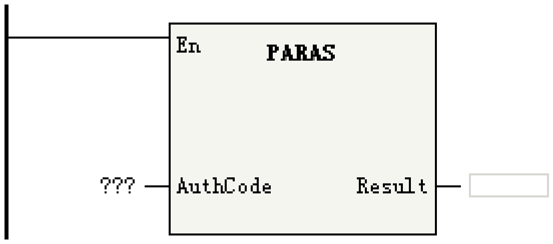
1. Run "H5U_AuthManger.exe" in the software installation directory.



2. Enter the IP address of the PLC, enter the 8-digit authorization code, and click "Set Authorization Code".
3. Click "Generate Verification Code". A string of characters is generated in the "Instruction Authorization Verification Code" text box.
4. You can also verify or clear the authorization code (only after you enter the authorization code) in the software.

Adding a Program Block

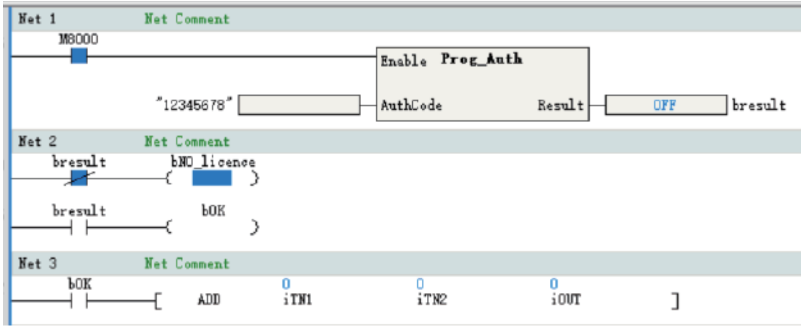
1. Open the function block to be authorized, and add the PARAS function block.



2. Enter the instruction authorization verification code generated by the software in "AuthCode".
3. The function block is authorized. If the authorization code of the PLC is inconsistent with that in the function block, the program in the function block cannot run.

Example

Since the verification code obtained by using Prog_Auth is inconsistent with the preset verification code in the PLC, the return value is "OFF", and the ADD instruction of the program is not executed.



3.11.4 FB Initial Values

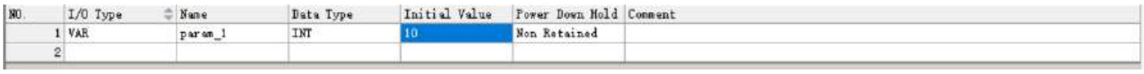
The initial values of FB settings can be modified based on the FB type or FB instance.

- Modifying the initial values based on the FB type is equivalent to modifying the initial values of the type.
- Modifying the initial values based on an FB instance is equivalent to modifying the initial values of the instance.
- If the initial values of an instance are modified, the member variables of the FB instance display the values after modification, and the background color of the cells is yellow.
- If the initial values of an instance are not modified, the member variables of the FB instance display the default values, and the background color of the cells is white.

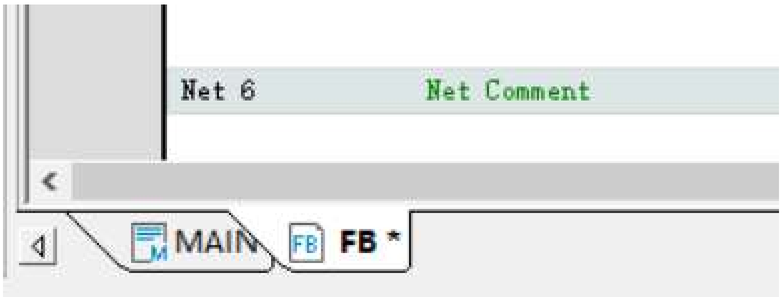
The initial values of the FB type are the default values of the instance. When the initial values of an instance are modified back to the default values, the background color of the cells changes from yellow to white.

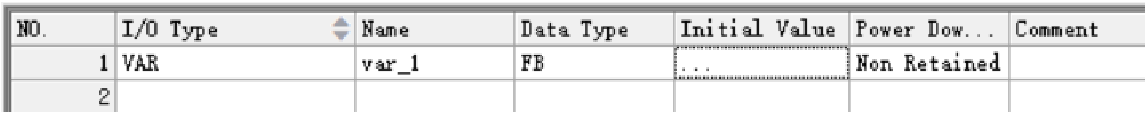
Modifying Initial Values When the FB Is Not Nested

Modify the initial value of the FB type from 0 to 10. Use the default value as the initial value of the FB instance, that is, the initial value 10 of the FB type.

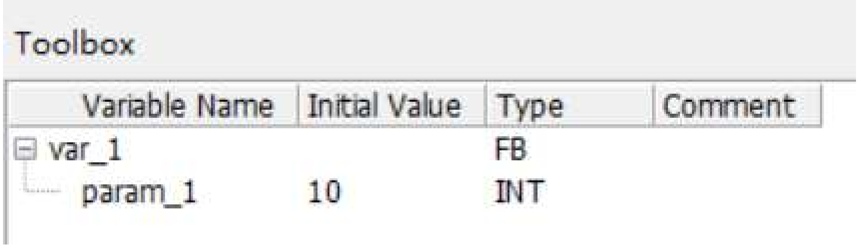


NO.	I/O Type	Name	Data Type	Initial Value	Power Down Hold	Comment
1	VAR	param_1	INT	10	Non Retained	
2						





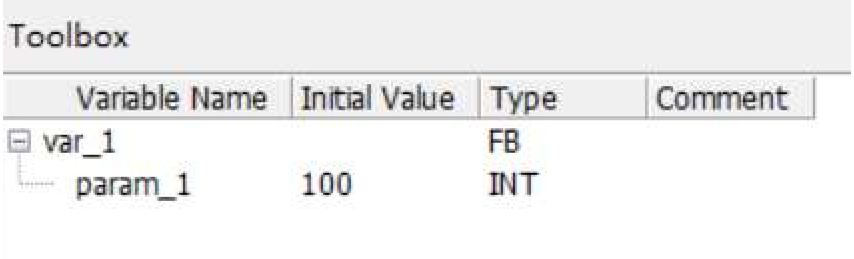
NO.	I/O Type	Name	Data Type	Initial Value	Power Dow...	Comment
1	VAR	var_1	FB	...	Non Retained	
2						



Toolbox

Variable Name	Initial Value	Type	Comment
⊟ var_1		FB	
param_1	10	INT	

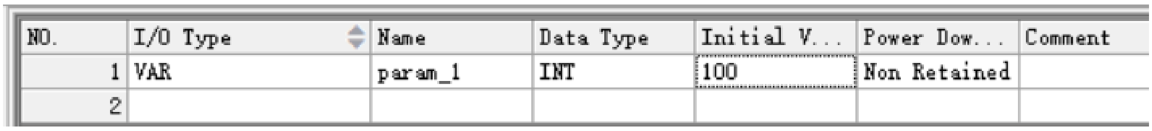
Modify the initial value of the FB instance from 10 to 100. The initial value of the FB instance is 100. At this time, if you attempt to modify the initial value of the FB type to 11, you will find that the initial value of the FB instance remains unchanged (still 100).



Toolbox

Variable Name	Initial Value	Type	Comment
⊟ var_1		FB	
param_1	100	INT	

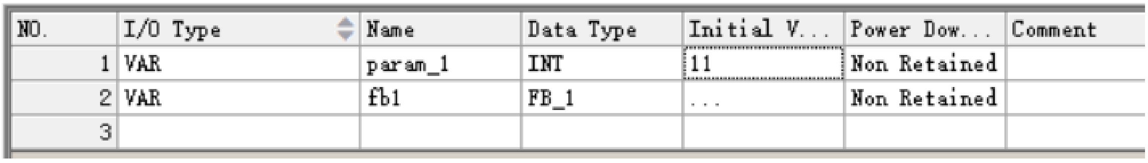
In the ladder diagram, double-click "FB" to display the FB instance. At this time, the initial value of the FB instance is displayed in the FB view instead of the initial value of the FB type. If the initial value of the variable is modified to be inconsistent with the FB, the background color will be yellow. Modifying the initial value on this interface is the same as modifying the initial value of the instance in the function block instance table.



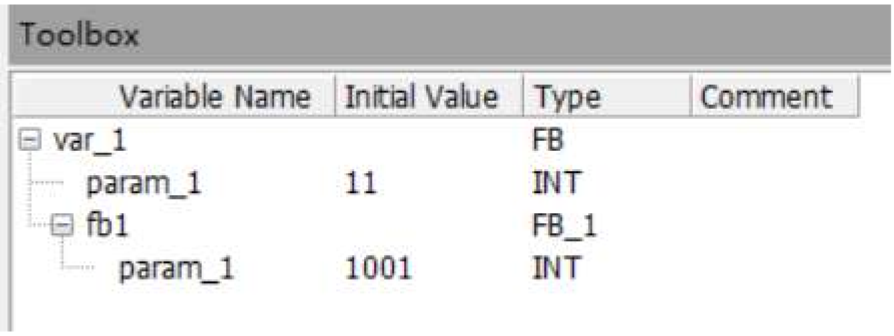
NO.	I/O Type	Name	Data Type	Initial V...	Power Dow...	Comment
1	VAR	param_1	INT	100	Non Retained	
2						

Modifying Initial Values When the FB Is Nested

Add a variable fb1 in the FB type, and set the data type to "FB_1". Modify the initial value of FB_1 from 1000 to 1001. The member variable fb1 of the FB type automatically takes the default value 1001 as the initial value, and the member variable fb1 of the FB instance also automatically takes the default value 1001 as the initial value.

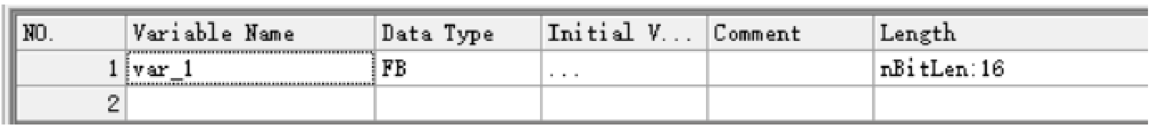


NO.	I/O Type	Name	Data Type	Initial V...	Power Dow...	Comment
1	VAR	param_1	INT	11	Non Retained	
2	VAR	fb1	FB_1	...	Non Retained	
3						

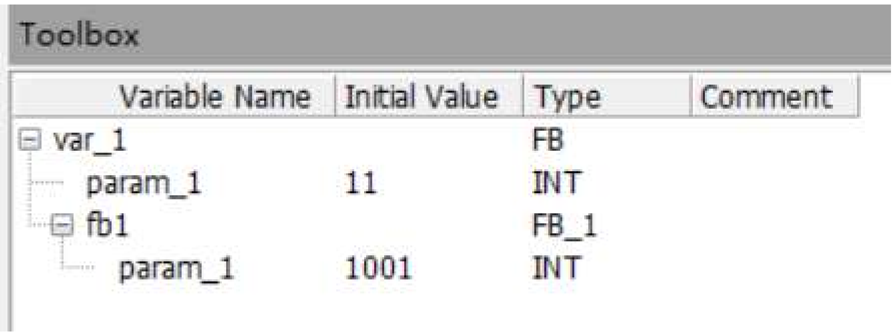


Toolbox

Variable Name	Initial Value	Type	Comment
⊟ var_1		FB	
param_1	11	INT	
⊟ fb1		FB_1	
param_1	1001	INT	



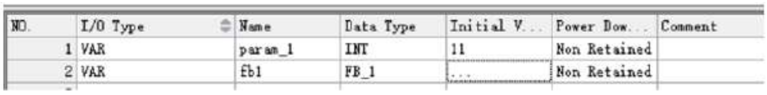
NO.	Variable Name	Data Type	Initial V...	Comment	Length
1	var_1	FB	...		nBitLen:16
2					



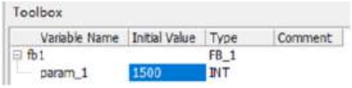
Toolbox

Variable Name	Initial Value	Type	Comment
⊟ var_1		FB	
param_1	11	INT	
⊟ fb1		FB_1	
param_1	1001	INT	

FB is the middle layer between the instance and FB_1. Modify the initial value of FB_1 to 1500 on the FB type interface. Then the initial value of the FB type changes to 1500, and the background color changes to yellow. At this time, the initial value of FB_1 of the FB instance is also 1500, but the background color is white, indicating that the default value is used.



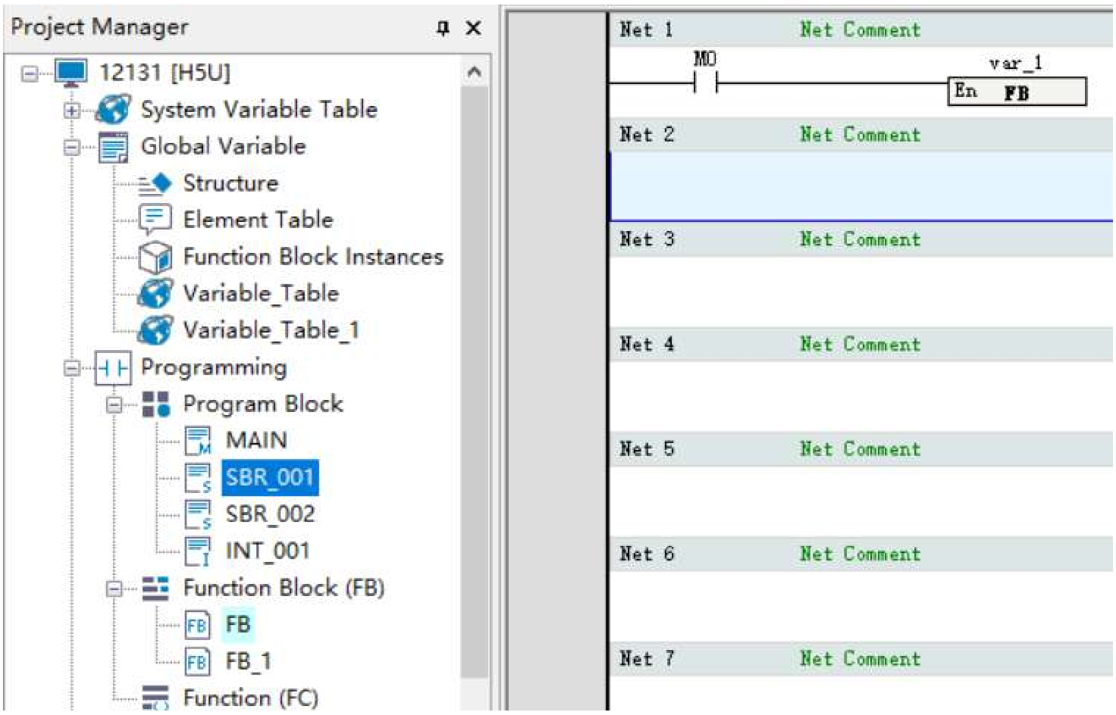
NO.	I/O Type	Name	Data Type	Initial V...	Power Dow...	Comment
1	VAR	param_1	INT	11	Non Retained	
2	VAR	fb1	FB_1	...	Non Retained	

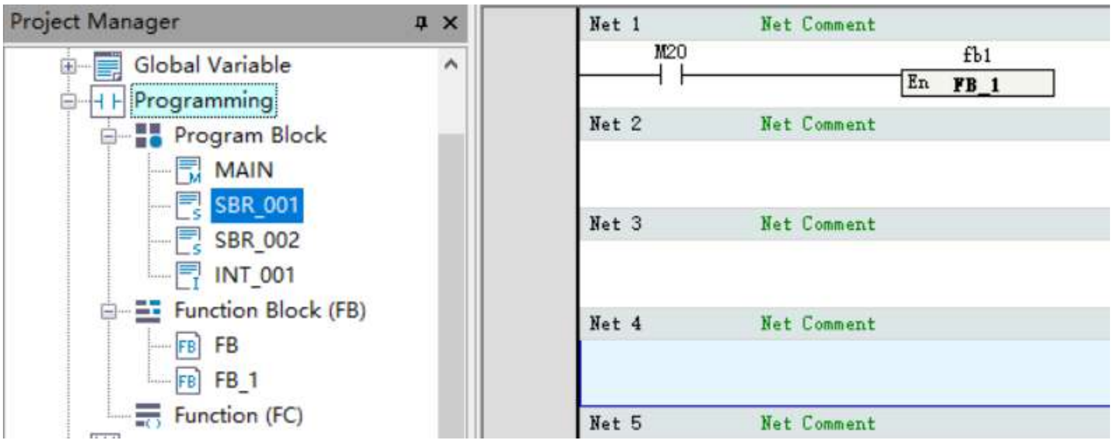


Toolbox

Variable Name	Initial Value	Type	Comment
⊟ fb1		FB_1	
param_1	1500	INT	

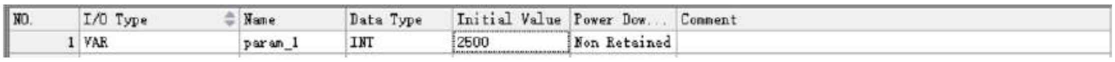
- Enter the instance interface from the main program. The initial value of the FB instance is displayed. Double-click "FB_1" to enter the FB_1 instance interface, and modify the initial value to 2000. Open the FB_1 type, and the initial value is still 1001. Open the FB instance FB_1, and the initial value is 2000.





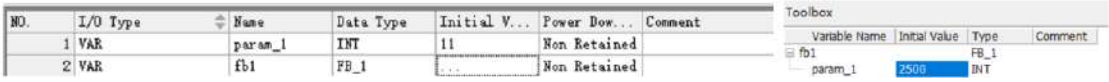
At this time, the tab name is "FB_1(var_1.fb1)".

- Double-click "FB" in the "Project Manager" navigation tree. You can see that the initial value of FB_ 1 on the FB type interface is 1500. Double-click "FB_1" in the ladder diagram of the FB type interface to enter the FB_1 instance interface. You can see that the initial value of the FB_1 instance is 1500. Modify it to 2500. Then return to the FB type interface to check the initial value of FB_1. You will find that it also changes to 2500.



NO.	I/O Type	Name	Data Type	Initial Value	Power Dow...	Comment
1	VAR	param_1	INT	2500	Non Retained	

At this time, the tab name is "FB_1(fb1)".



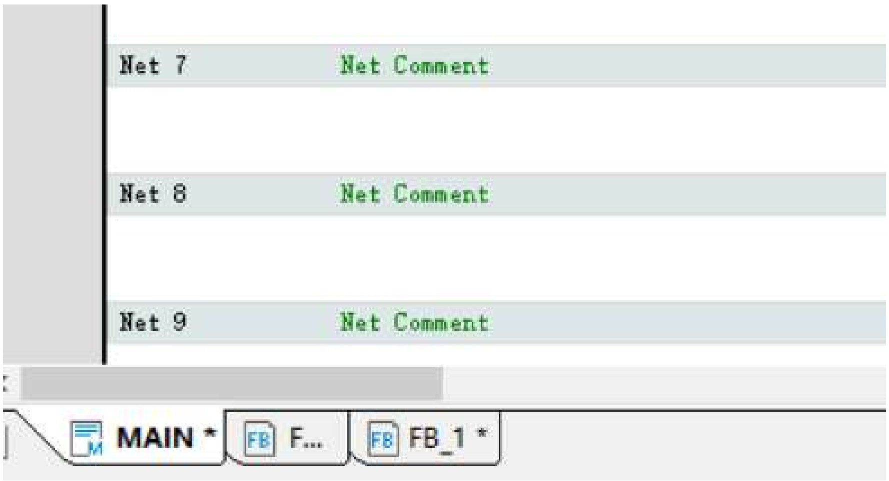
NO.	I/O Type	Name	Data Type	Initial V...	Power Dow...	Comment
1	VAR	param_1	INT	11	Non Retained	
2	VAR	fb1	FB_1	...	Non Retained	

Toolbox

Variable Name	Initial Value	Type	Comment
fb1		FB_1	
param_1	2500	INT	

Tab at the Bottom of the FB View

The tab displayed at the bottom of the FB view contains the following information from left to right: node name, instance name, and unsaved flag. The node name is the name of the project tree node, and the instance name refers to the instance name in parentheses. The following figures show the details.



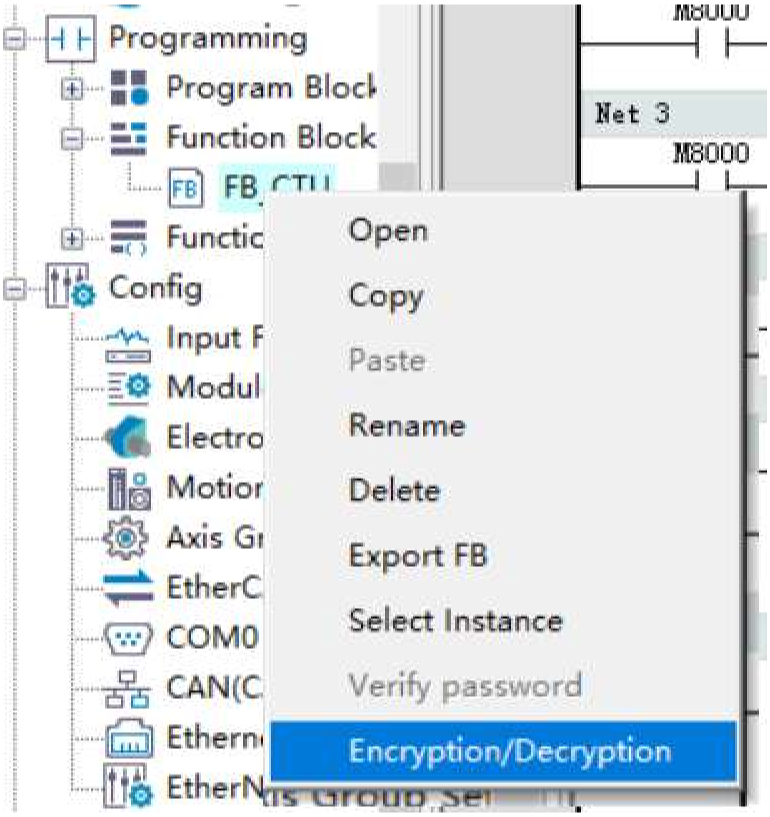
As shown in the preceding figure, "FB" is the node name, "var_1.fb1" is the instance name, and "*" indicates unsaved.

Since the tab needs to be parsed, characters including the period (.), asterisk (*), and parentheses (()) are not allowed when FBs and structures are renamed.

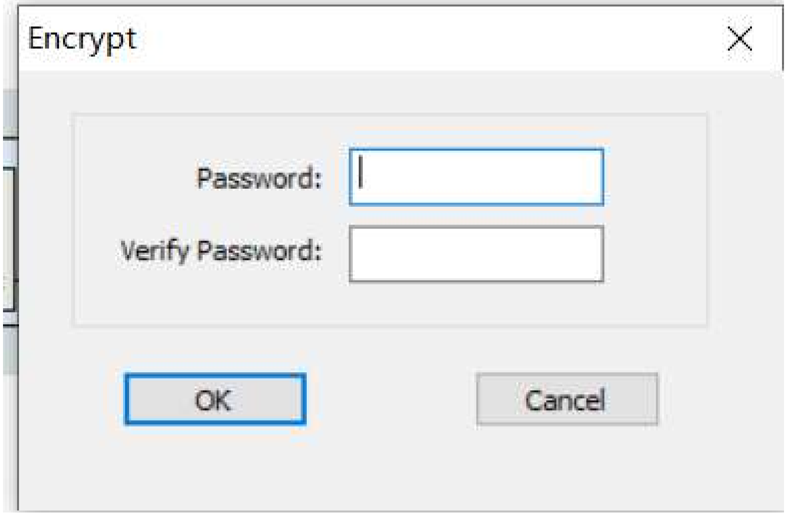
3.11.5 Encrypting FB or FC

This section takes encryption of function blocks as an example. The process is similar for encrypting functions. After encryption, the method of calling the function blocks or functions remains unchanged.

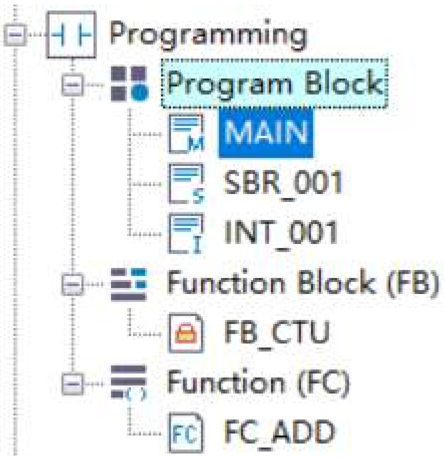
1. Choose "Programming" > "Function Block (FB)" in the project management window, right-click "FB", and choose "Encryption/Decryption".



2. Enter and confirm the password in the displayed "Encrypt" dialog box.

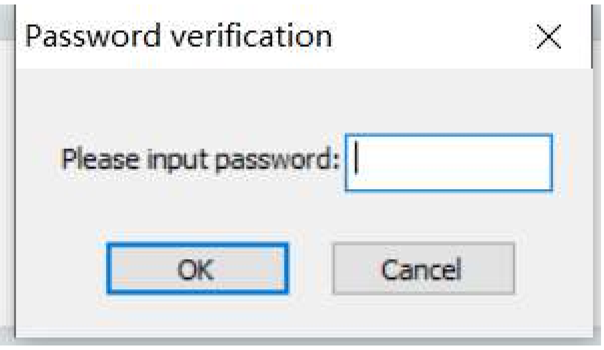


The following figure shows a function block after encryption.



Performing the preceding steps on an encrypted function block will decrypt it and restore it to its original unencrypted state.

To access an encrypted function block, you can double-click the encrypted node, or right-click the encrypted node and choose "Password verification" from the shortcut menu, and enter the correct password in the displayed dialog box.

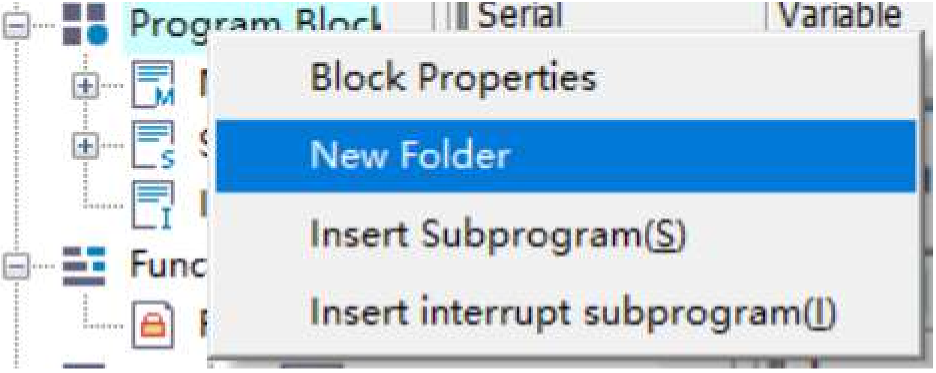


3.12 Folder

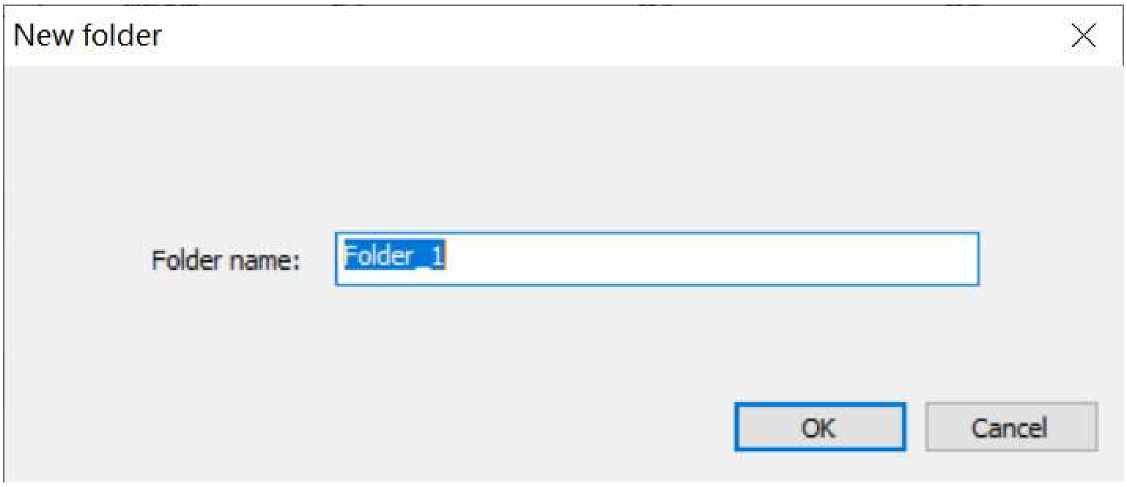
You can use folders to classify and batch operate program blocks, function blocks, and functions.

Creating a folder

1. Right-click "POU", "Function block", or "Function", and select "New folder". For example, right-click "POU", as shown in the following figure.



2. In the "New folder" dialog box that is displayed, set "Folder name" and click "OK". A folder is created.



New folder

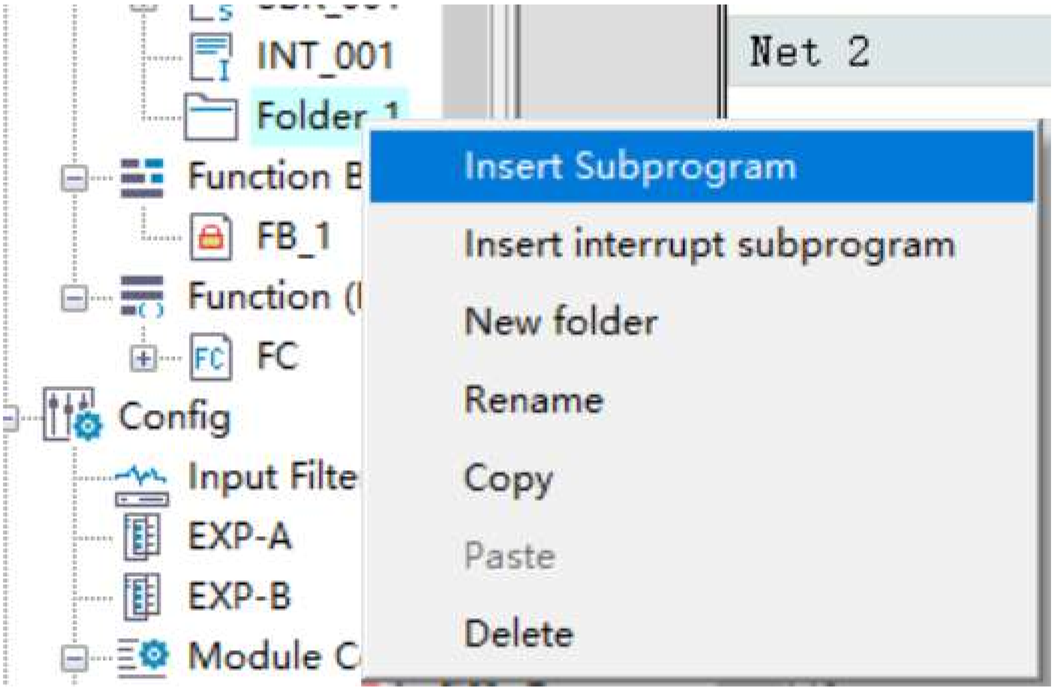
Folder name: Folder_1

Note

Nested folders can contain up to four levels. Each folder name must not:

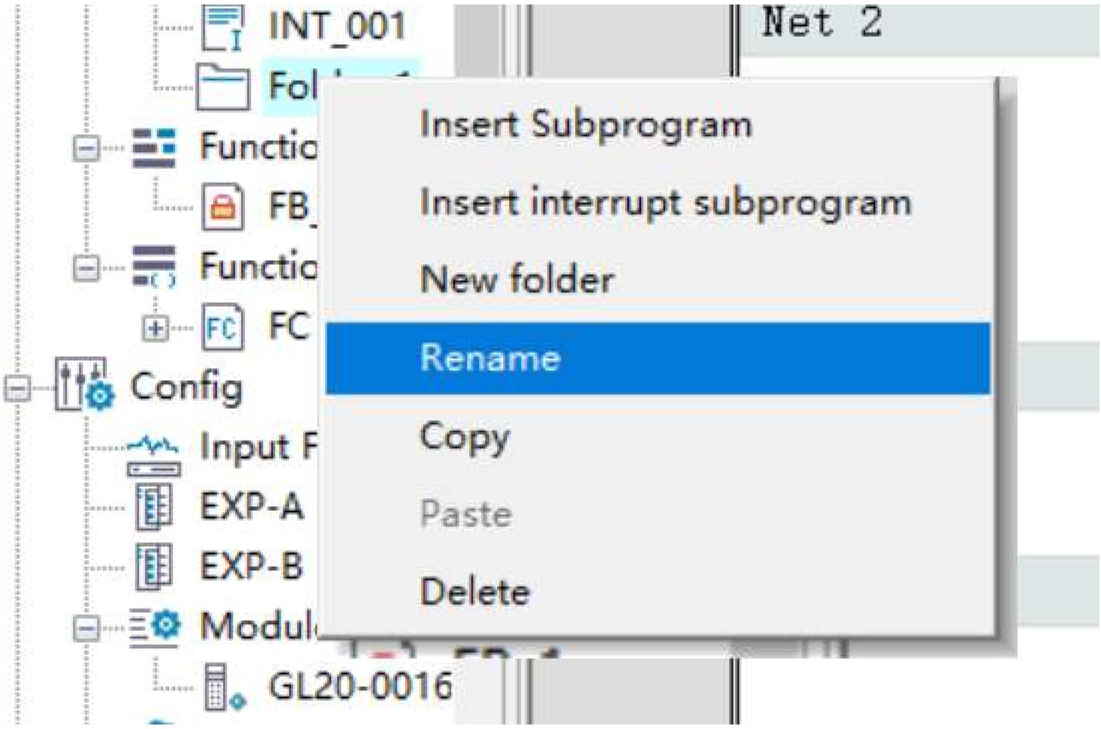
- Be empty.
- Contain special characters such as space, asterisk (*), pipe (|), backslash (\), less-than sign (<), comma (,), period (.), forward slash (/), left parenthesis ((), right parenthesis ()), and question mark (?), or start with underscore (_), numbers, SYS, or _SYS_.
- Be the same as the name of any soft element forms, standard data types, instructions, or constants.
- Be any keywords such as ARRAY, TRUE, FALSE, ON, OFF, and NULL.
- Be the same as the name of any files or folders at the same level.

3. Right-click the new folder and select "Insert Subprogram" to add the new subprogram file.
   You can drag program files from other folders to a specified folder.

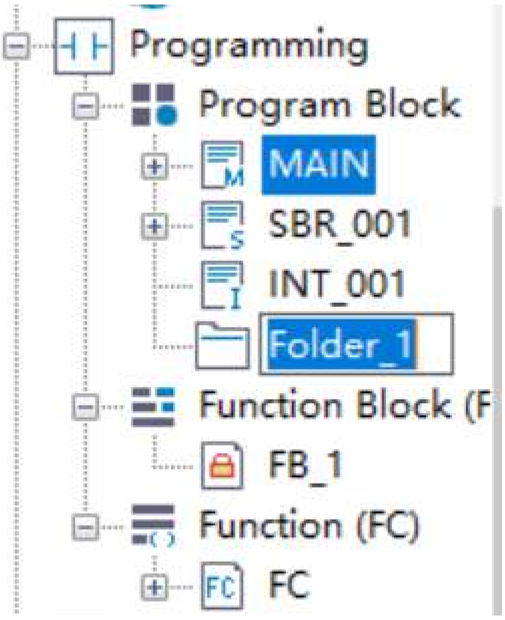


Renaming a folder

1. Right-click a folder to be renamed and select "Rename", as shown in the following figure.

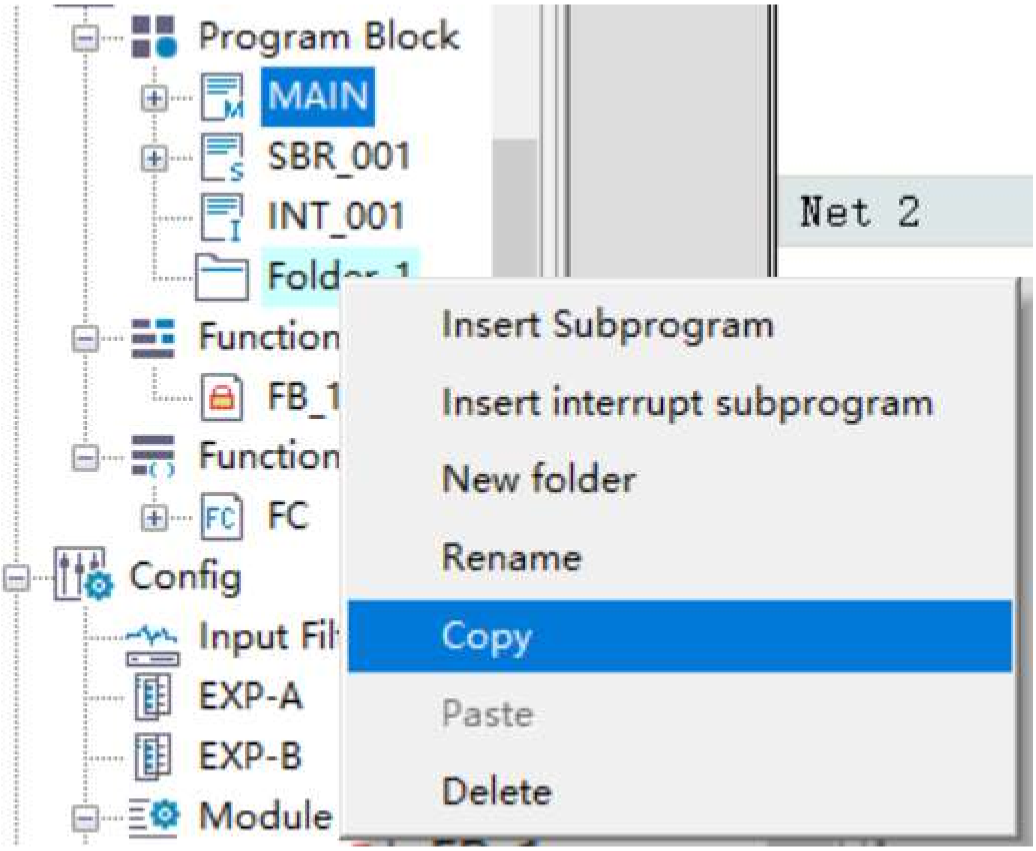


2. Enter the new folder name in the folder name text box and press "Enter", as shown in the following figure.



Copying and pasting a folder

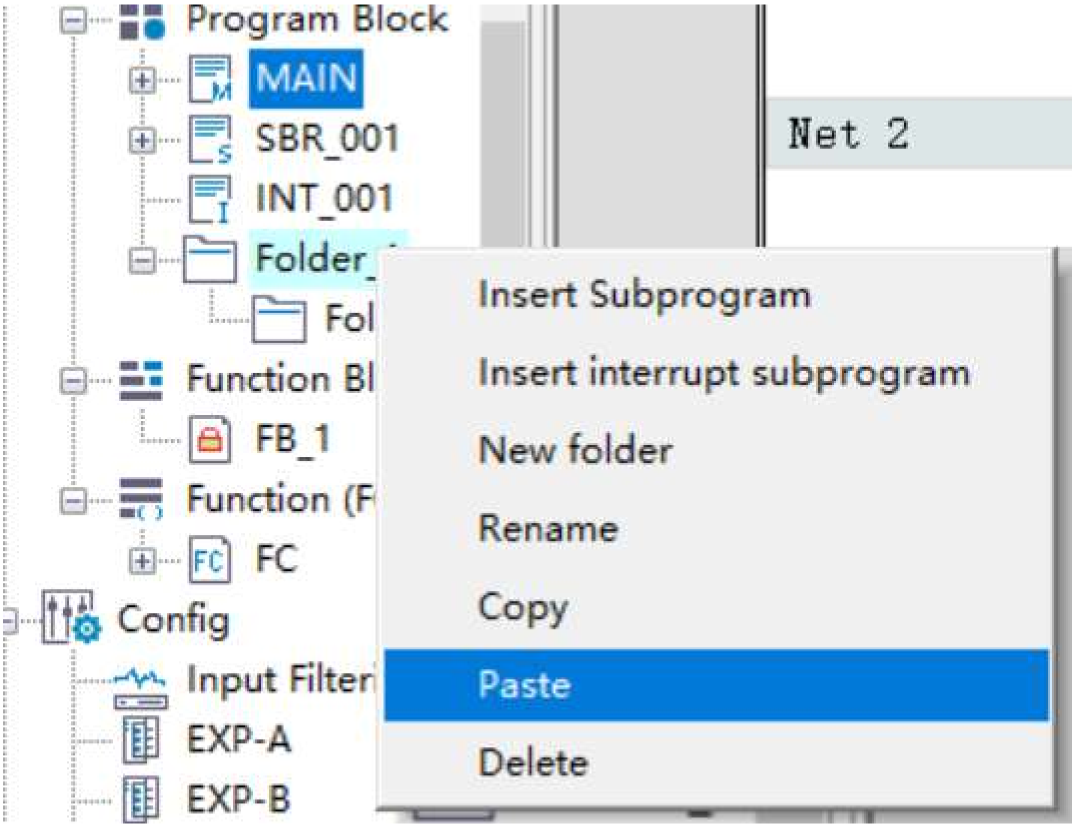
1. Right-click a folder to be copied and select "Copy" to copy the folder and the sub folders and files in the folder to the clipboard, as shown in the following figure.



Caution

Folders containing an encrypted program cannot be copied.

2. When the clipboard contains valid content, right-click the target node, and select "Paste" to paste the content in the clipboard to the target node, as shown in the following figure.

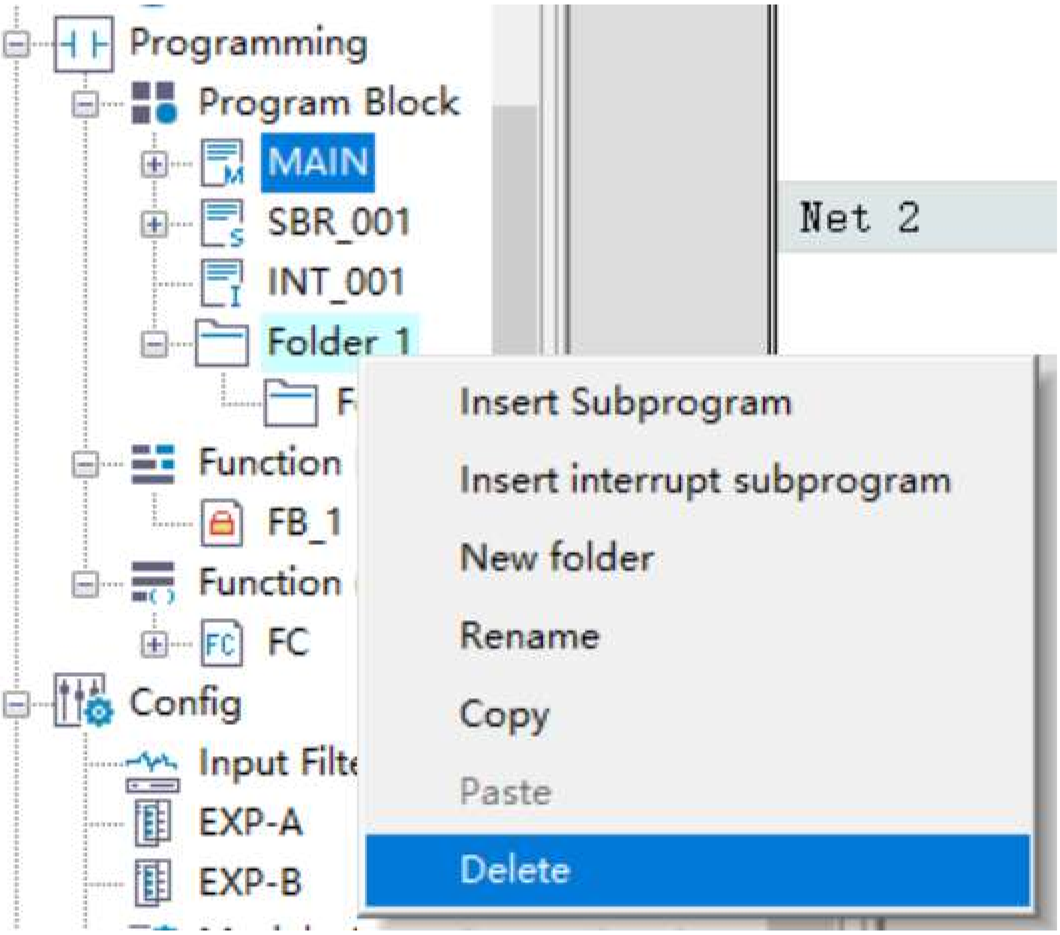


Caution

- Pasting fails when the target node contains a file or folder with the same name as the folder in the clipboard.
- Pasting fails if the target node and the folder in the clipboard will produce nested folders of more than four levels.

Deleting a folder

1. Right-click a folder to be deleted and select "Delete", as shown in the following figure.



⚠ Caution

Folders containing an encrypted program cannot be deleted.

2. In the dialog box that is displayed, click "OK" to delete the selected folder and sub folders and files in the folder.

4 Programming Languages

4.1 Programming Language (LiteST)

4.1.1 Overview

LiteST is an high-level text-based programming language for automation systems. Its syntax structure is similar to that of PASCAL. It provides a simple standard structure to make programming fast and efficient. LiteST uses many traditional characteristics of high-level languages, including variables, operators, and control statements. LiteST provides a freer text-based programming mode than IL because extra placeholders are added to ensure a hierarchical structure of the program frame for easy reading and understanding. LiteST also provides easier migration and repeatability than graphical programming modes such as LD.

Example:

```
IF A>0 THEN
    X:=10;
ELSE
    X:=0;
END_IF;
```

Note

This function requires a firmware version of V5.14.0.0 or later for the H5U series, or a firmware version of V5.67.0.0 or later for the Easy series, and an AutoShop software version of V4.8.1.0 or later.

4.1.2 Expressions

Block diagrams of different functions are basic elements in the LD programming environment. Similar to LD, expressions are basic elements for LiteST. An expression consists of operators and operands. An operand can be a constant, a variable, a function call, or other expressions.

- Constant, such as 20, 1.43, and 16#10
- Variable, such as iVar and D0:E
- Function call, whose value is the return value of a call, such as Fun1(1,2,4)
- Other expressions: such as 10+3, var1 OR var2, (x+y)/z, and iVar1:=iVar2+22

In an expression, operands are evaluated using operators in sequence defined by a particular operator priority. Operators with top priority must be first used for evaluation. Other operators with lower priority are used by priority in descending order. Operators with the same priority must be used in order from left to right in the expression.

For example, if A, B, C, and D are INT variables and are set to 1, 2, 3, and 4 respectively, A+B-C*ABS(D) must be –9 and (A+B-C)*ABS(D) must be 0.

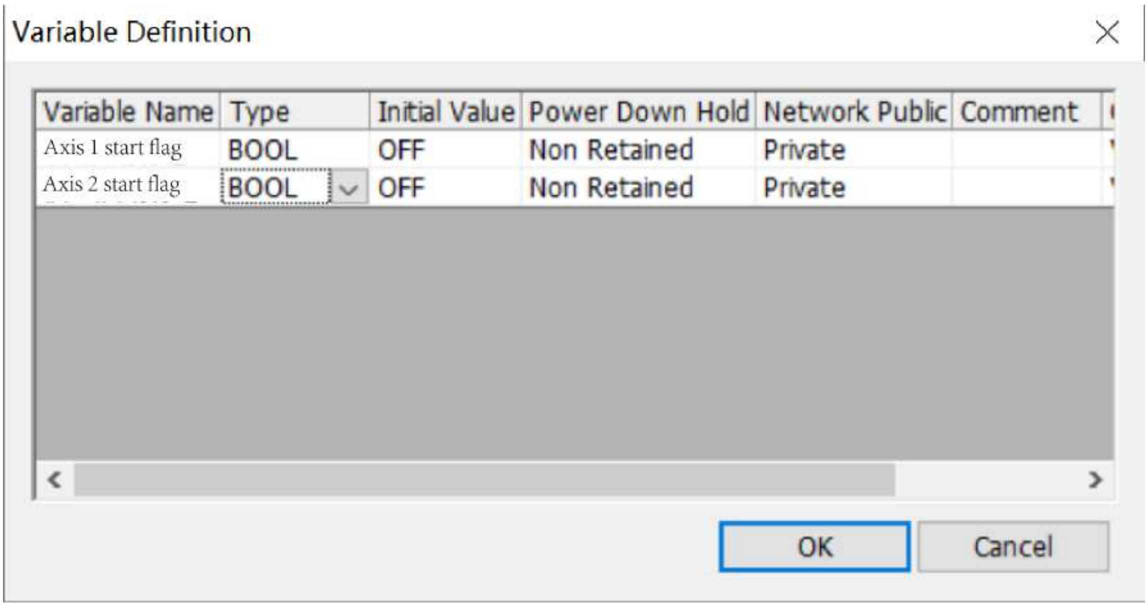
When an operator has two operands, the leftmost operand must be evaluated first. For example, in SIN (X)*COS(Y), SIN(X) must be evaluated first, then COS(Y), and finally the product of the overall expression.

Table 4–1 LiteST operators

Operation Type	Sign	Example	Priority
Bracket	(Expression)	(A+B/C), (A+B)/C, A/(B+C)	9 (highest)
Function call	Function name (separated by commas (,) in the parameter list)	LN(A), MAX(X,Y)	8
Opposite	-	-A	7
Unary plus (+)	+	+B	7
Negate	NOT	NOT C	7
Multiply	*	A*B	6
Divide	/	A/B	6
Modulo	MOD	A MOD B	6
Plus	+	A+B	5
Minus	-	A-B	5
Compare	<, >, <=, >=	A<B	4
Equal	=	A=B	4
Not equal	<>	A<>B	4
Logic AND	AND	A AND B	3
Logic XOR	XOR	A XOR B	2
Logic OR	OR	A OR B	1 (lowest)

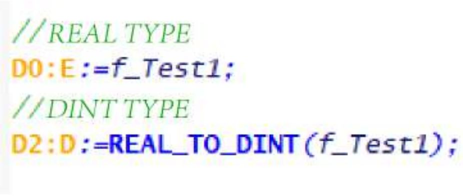
4.1.3 Variables

You can compile variables during LiteST program editing and press Enter or click in the area outside the program basic block to display the variable definition box. The default variable type is INT. During a function call or an instruction call, the data type can be automatically identified.



Variable Name	Type	Initial Value	Power Down Hold	Network Public	Comment
Axis 1 start flag	BOOL	OFF	Non Retained	Private	
Axis 2 start flag	BOOL	OFF	Non Retained	Private	

LiteST supports various data types such as D, R, and W. For example, D0 s a 16-bit integer, DO:D is a 32-bit integer, and DO:E is a 32-bit floating-point number. The following figure shows the details.



```
//REAL TYPE
D0:E:=f_Test1;
//DINT TYPE
D2:D:=REAL_TO_DINT(f_Test1);
```

4.1.4 Constants

Constants can be expressed in may ways:

1. A constant is a decimal number by default, for example, a:=100.
2. A constant can contain underlines (_), for example, a:=10#100_10, a:=16#FF_AE_12, and a:=2#1100_1111_11_10, as shown in the following figure.
3. LiteST also allows LD expression as constants. That is, K100 indicates constant 100, H indicates a 16-bit number, and E indicates a floating-point number.

4.1.5 FB, FC, Subprogram, and Interrupt

FB: In terms of input parameters, only axis parameters are variables requiring pin input and other pins do not need to be input. In addition, output pins do not need to be input.

FC: Input parameter pins must be input. Otherwise, a compilation error is reported. Output parameters can be left empty.

Subprogram: A subprogram is called in non-parametric function format, for example, SBR_001().

Interrupt: Interrupts do not need to be manually called. EI() must be called to enable interrupts and DI () must be called to disable interrupts.

Precautions

- Up to eight hierarchies can be called for FB and FC nesting.
- Up to six hierarchies can be called for SBR nesting.

4.1.6 Intelligent Input and Prompts

4.1.6.1 Quick Input

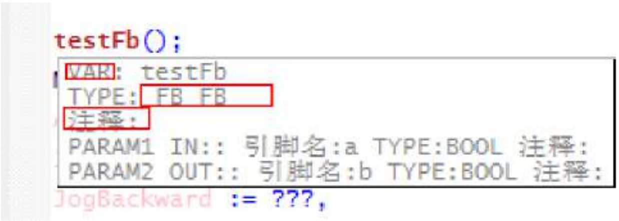
After entering the instruction name, press the tab key to complement the instruction pin. If the default parameter next to the pin is "???", the parameter must be input. Otherwise, the parameter can be input or not as required.

```
MC_Jog(Enable := ???,
    Axis := ???,
    JogForward := ???,
    JogBackward := ???,
    Velocity := ???,
    Acceleration := ???,
    Deceleration:= ,
    CurveType:= ,
    Busy => ,
    CommandAborted => ,
    Error => ,
    ErrorID => );;
```

4.1.6.2 Mouse Hover Prompt

When you hover the mouse over a variable, the variable name, type, and comment are displayed.

When you hover the mouse over an FB, FC, or instruction, the function name, function type, function comment, input and output parameters, pin name, type, and comment are displayed.



```
testFb();
VAR: testFb
TYPE: FB FB
注释:
PARAM1 IN:: 引脚名:a TYPE:BOOL 注释:
PARAM2 OUT:: 引脚名:b TYPE:BOOL 注释:
JogBackward := ???,
```

4.1.7 Syntax Instructions

4.1.7.1 Overview

The overall LiteST program consists of instructions separated by semicolons (;).

Table 4–2 LiteST syntax instructions

Instruction	Function	Example
:=	Assignment	A := B
Function block call	Function block call and output	TONR(IN := b0,PT := dVar,R := b0,Q => ,ET =>);

IF	Selection	IF A>0 THEN
		X:=10;
		ELSE
		X:=0;
		END_IF;

Instruction	Function	Example

CASE	Multiway branch	CASE A OF
		1: X:=1;
		2: X:=2;
		3: X:=3;
		ELSE
		X:=0;
		END_CASE;

WHILE	WHILE loop	A := 0;
		WHILE A <= 1000 DO
		A := A+7;
		END_WHILE;

REPEAT	REPEAT loop	A := 1;
		TOTAL := 0;
		REPEAT
		TOTAL := TOTAL + A;
		A := A+1;
		UNTIL A>10
		END_REPEAT;

FOR	FOR loop	FOR i:=0 TO 100 DO
		X[i]:=0;
		END_FOR;

EXIT	EXIT loop	EXIT;
CONTINUE	Interrupting the current loop	CONTINUE;
RETURN	Return	RETURN;

(*Text*)	Comment	(*Comment out multiple lines
		IF A=3 THEN
		A:=5;
		END_IF;
		*)

//Text	Single-line comment	//A:=5;
;	Empty statement	;

4.1.7.2 Assignment Instructions

In an assignment statement, the evaluation result of the expression is used to replace the current values of one or multiple element variables. An assignment statement must contain a variable reference on the left, followed by the assignment operator ":=" and then the evaluation expression.

Example:

A:=B*10

After execution, the value of A is 10 times the value of B.

4.1.7.3 Function Block Calls

Syntax: FB instance name (FB input variable := value, FB output variable => value,... More FB input and output variables);

Example: After you call an instance of the function block with the maximum value evaluated (MAXFB), load input parameters D0 and D1 and the output parameter D2, and execute the function, the result is assigned to the variable maxVar.

MYFB(VAR1 := D0,VAR2 := D1,RESULT => D2);

maxVar := MYFB.RESULT;

Note

myFB is the functional block instance of MAXFB.

4.1.7.4 IF

IF instructions are used to execute relevant statements according to the calculation result of condition expressions.

In IF select statements, only the statements where the condition expression value is boolean 1(TRUE) can be executed. If the condition is set to 0(FALSE), no statement is executed or the statements in the ELSE (or ELSIF) keyword conditions are executed.

1. ELSIF is not required if there is only one condition expression. In addition, ELSE is not required if the Boolean expression indicating no condition is met is not processed. For example:

```
IF condition expression THEN
        statement;
END_IF
```

2. One IF select statement can contain multiple ELSIF statements. For example:

```
IF condition expression 1 THEN
        statement;
ELSIF condition expression 2 THEN
        statement;
ELSIF condition expression 3 THEN
        statement;
ELSIF condition expression 4 THEN
        statement;
ELSE
        statement;
```

```
END_IF
```

3. IF select statements can be nested. Statement 11 is executed when the condition expressions 1 and 11 are TRUE.

```
IF condition expression 1 THEN
IF condition expression 11 THEN
        statement 11;
ELSIF condition expression 12 THEN
        statement 12;
ELSE
        statement 13;
END_IF
ELSIF condition expression 2 THEN
        statement 2;
ELSE
        statement 3;
END_IF
```

Example

```
IF score <60 THEN
        bPass := FALSE;
ELSE
        bPass := TRUE;
END_IF
```

In the example, if the score is less than 60, the test fails. Otherwise, the test passes.

Precautions

- An IF select statement consists of one IF, one THEN, and one END_IF at least.
- The keyword must be ELSIF rather then ELSEIF.
- In a condition expression, the keyword THEN is used to determine whether the expression ends.

4.1.7.5 CASE

Functions and instructions

Statements to be executed are selected from multiple statements based on the value of the specified integral expression. An integral expression can be the return value of an INT variable, a DINT variable, an expression, or a function.

```
CASE integral expression OF
```

```
  Value 1: statement 1;

  Value 2: statement 2;

  Value 3, value 4: statement 3;

  Value 5, value 6: statement 4;

      :

  ELSE

statement 5;

END_CASE;
```

Like the program expressed by LiteST, CASE instructions can be processed as follows:

- If the value of an integral expression is value 1, statement 1 is executed;
- If the value of an integral expression is value 2, statement 2 is executed;
- If the value of an integral expression is value 3 or value 4, statement 3 is executed;
- If the value of an integral expression is value 5 or value 6, statement 4 is executed;
- Otherwise, statement 5 is executed.

CASE select statements can be nested. Statement 12 is executed when the value of integral expression 1 is value 1 and the value of integral expression 11 is value 2.

```
CASE integral expression 1 OF

    1 :

   CASE integral expression 11 OF

          1 : statement 11;

          2 : statement 12;

          ELSE

          statement 1m;

   END_CASE;

    2 : statement 2;

    3 : statement 3;

    ELSE

   statement n;

END_CASE;
```

Example

```
CASE ERROR_CODE OF

    1:ERR_MSG := 'function lacking the right bracket';

    2:ERR_MSG := 'failure to process variables';

    3:ERR_MSG := 'invalid variable initial value';
    ...
    255:ERR_MSG := 'function lacking the right bracket';
    ELSE ERR_MSG := 'unknown error';
END_CASE
```

Precautions

- In CASE statements, values of expressions must be integers.
- ELSE options are optional, and some programs can contain only the CASE...OF...END_CASE structure.

4.1.7.6 WHILE

Functions and instructions

When the calculation result of a specified condition expression is TRUE, a statement is repeatedly executed.

In a sense, the WHILE loop and REPEAT loop functions are more powerful than the FOR loop function because cycle times do not need to be counted before loop execution. Therefore, only the WHILE loop and REPEAT loop are required in some cases. However, if the cycle times is clear, the FOR loop is better.

Example

```
WHILE Counter<>0 DO
   Var1:= Var1*2;
   Counter := Counter-1;
END_WHILE
```

Precautions

WHILE must be used with END_WHILE in pair.

4.1.7.7 REPEAT

Functions and instructions

After a statement is executed once, repeat it before the value of the specified condition expression changes to TRUE. A REPEAT instruction requires statement running before condition expression evaluation. Therefore, the statement must be executed.

Example

```
A := 1;
TOTAL := 0;
```

```
REPEAT
    TOTAL := TOTAL + A;
    A := A+ 1;
UNTIL A>10
END_REPEAT;
```

Numbers 1 to 10 are added together, and the result is used for the variable TOTAL.

Precautions

REPEAT, UNTIL, and END_REPEAT are necessary.

4.1.7.8 FOR

Functions and instructions

The FOR loop can be used to compile the iterative processing logic.

```
FOR control variable := Initial value TO Final value{BY incremental value} DO
      statement;
END_FOR;
```

In the preceding program,

Information in the braces is optional.

The control variable is the counter. The statement will be executed only if the value on the counter is not greater than the final value. Before statement execution, check this condition. If the initial value is greater than the final value, the statement will not be executed.

After the statement is executed the last time, the counter automatically increases the incremental value. An incremental value can be any integer. If the parameter is not set, the default value is 1. When the value on the counter is greater than the final value, the loop stops.

Example

```
sumResult := 0;
factorial := 1;
FOR i :=1 TO 10 BY 1 DO
       sumResult := sumResult + i;
       factorial := factorial * i;
END_FOR;
```

In the preceding example, sumResult (result of adding 1 to 10) and the factorial result are calculated.

Precautions

FOR must be used with END_FOR in pair.

4.1.7.9 EXIT

Functions and instructions

An EXIT instruction is used to exit the FOR, WHILE, or REPEAT loop. The instruction interrupts iterative processing of the internal FOR, WHILE, or REPEAT instruction, and executes the next step of the iterative processing.

Example

```
IF A THEN
    DATA[3] :=98;
    FOR n := 1 TO 50 BY 1 DO
        DATA[n] := DATA[n] + n;
        IF DATA[n] > 100 THEN EXIT;
        END_IF;
    END_FOR;
A :=FALSE;
END_IF;
```

In the preceding example, the variable value starts from n=1 and increases to 50 by 1 repeatedly to add n to the specified sorting variable DATA[n]. However, after the value of DATA[n] exceeds 100, the operation ends.

Precautions

- This instruction must be used between FOR and END_FOR, WHILE and END_WHILE, or REPEAT and END_REPEAT.
- To interrupt all hierarchical iterative processing (nesting), the number of the EXIT instructions must be the same as that of hierarchies.

4.1.7.10 CONTINUE

Functions and instructions

A CONTINUE instruction is used to end a FOR, WHILE, or REPEAT loop in advance and start the next loop. It is different to interrupt a loop and exit a loop. When you interrupt a loop, the loop is ignored and the next loop is executed.

Example

```
FOR Counter:=1 TO 5 BY DO
   INT1:=INT1/2;
   IF INT1=0 THEN
       CONTINUE;
```

```
    END_IF
Var:=Var1/UBT1L
END_FOR;
```

Precautions

This instruction must be used between FOR and END_FOR, WHILE and END_WHILE, or REPEAT and END_REPEAT.

4.1.7.11 RETURN

Functions and instructions

This instruction forcibly ends main programs, subprograms, FBs, or FCs.

Example

```
IF b=TRUE THEN
    RETURN;
END_IF;
a:=a+1;
```

If b is TRUE, the statement "a:=a+1;" will not be executed, and POU will be immediately returned.

Precautions

If this instruction is frequently used, the process will be complex.

4.1.7.12 Comments

Structured text can be commented in two ways:

- Single-line comment: Start with "//", for example, "// This is a comment.".
- Multi-line comment: Start with "(*" and end with "*)", for example, "(*This is a comment.*)".

Comments can be added to the LiteST editor declaration or any part in implementation.

A comment can be nested in other comments.

Example

```
(*
a:=inst.out; (*to be checked*)
b:=b+1;
*)
```

4.1.8 PLC Instructions Supported by LiteST

4.1.8.1 Basic Axis Control Instructions

Instruction	Description
MC_Power	Enable
MC_Reset	Reset
MC_ReadStatus	Axis reading status
MC_ReadAxisError	Axis reading fault
MC_ReadDigitalInput	DI reading status
MC_ReadActualPosition	Reading the current position
MC_ReadActualVelocity	Reading the actual speed
MC_ReadActualTorque	Reading the actual torque
MC_SetPosition	Setting the current position
MC_TouchProbe	Probe
MC_MoveRelative	Relative positioning
MC_MoveAbsolute	Absolute positioning
MC_MoveVelocity	Speed instruction
MC_Jog	Jog
MC_TorqueControl	Torque control instruction
MC_Home	Homing instruction
MC_Stop	Stop instruction
MC_Halt	Pause (not recoverable)
MC_SetOverRide	Overshoot value reference
MC_MoveFeed	Interrupt positioning
MC_ImmediateStop	Emergency stop
MC_MoveVelocityCSV	CSV-based speed instruction with adjustable pulse width
MC_SyncMoveVelocity	CSV-based synchronous speed instruction supporting PWM
MC_FollowVelocity	CSP-based synchronous speed instruction
MC_MoveBuffer	Multi-position instruction
MC_MoveSuperImposed	Motion superimposition
MC_SyncTorqueControl	Sync torque control instruction
MC_SetAxisConfigPara	Setting axis parameters
MC_MoveLinear	Linear interpolation
MC_GroupStop	Stopping axis group operation
MC_MoveCircular	Circular interpolation
MC_GroupPause	Pausing axis group operation

4.1.8.2 Cam and Gear Instructions

Instruction	Description
MC_CamIn	Starting cam operation
MC_CamOut	Canceling cam operation
MC_Phasing	Master axis phase shift
MC_GenerateCamTable	Updating cam table
MC_SaveCamTable	Saving cam table

Instruction	Description
MC_GearIn	Starting gear operation
MC_GearOut	Canceling gear operation
MC_GetCamTablePhase	Obtaining cam table phase
MC_GetCamTableDistance	Obtaining cam table offset
MC_DigitalCamSwitch	Controlling electronic cam tappet

4.1.8.3 Encoder Instructions

Instruction	Description
HC_Counter	High-speed counter enable
HC_Preset	High-speed counter preset value
HC_TouchProbe	High-speed counter probe
HC_Compare	High-speed counter comparison
HC_ArrayCompare	High-speed counter array comparison
HC_StepCompare	High-speed counter equidistance comparison
ENC_SetUnit	Encoder axis setting gear ratio (valid only for the local encoder axis)
ENC_SetLineRotationMode	Encoder axis setting linearity rotation mode (valid only for the local encoder axis)
ENC_Counter	Encoder axis enable
ENC_Reset	Encoder axis fault reset (only for the bus encoder axis)
ENC_ResetCompare	Encoder axis reset comparison output (only for the bus encoder axis)
ENC_Preset	Encoder axis preset value
ENC_TouchProbe	Encoder axis probe
ENC_Compare	Encoder axis comparison output (only for the local encoder axis)
ENC_ArrayCompare	Encoder axis array comparison
ENC_StepCompare	Encoder axis step comparison
ENC_GroupArrayCompare	Encoder axis group array comparison (only for the bus encoder axis)
ENC_ReadStatus	Encoder axis read status (only for the bus encoder axis)
ENC_DigitalOutput	Encoder axis digital output control (only for the bus encoder axis)

4.1.8.4 Communication Instructions

Instruction	Description
ETC_ReadParameter_CoE	Read the SDO parameter of the ETC slave station
ETC_WriteParameter_CoE	Write the SDO parameter of the ETC slave station
ETC_RestartMaster	Restart the ETC master station

4.1.8.5 Timer Instructions

Instruction	Description
TPR	Pulse timer
TONR	Connection delay timer

Instruction	Description
TOFR	Off delay timer
TACR	Accumulation timer

4.1.8.6 Interrupt Instructions

Instruction	Description
EI	Enable interrupt
DI	Disable interrupt

4.1.8.7 Operation Instructions

Instruction Type	Instruction Description	Instruction Name
Mathematical operation instruction	Modulo	MOD
Shift instruction	Bitwise left shift	SHL
	Bitwise right shift	SHR
Select instruction	Either one	SEL
	Maximum value	MAX
	Minimum value	MIN
Arithmetic operation instruction	Absolute value	ABS
	Square root	SQRT
	Natural logarithm	LN
	Common logarithm	LOG
	Power exponent	EXPT
	Sine function	SIN
	Cosine function	COS
	Tangent function	TAN
	Arcsine function	ASIN
	Arccosine function	ACOS
	Arctan function	ATAN
Word logic	And	AND
	Or	OR
	Not	NOT
	Exclusive OR	XOR

Instruction Type	Instruction Description	Instruction Name
Data type conversion	BOOL to INT	BOOL_TO_INT
	BOOL to DINT	BOOL_TO_DINT
	BOOL to REAL	BOOL_TO_REAL
	INT to REAL	INT_TO_REAL
	INT to DINT	INT_TO_DINT
	INT to BOOL	INT_TO_BOOL
	DINT to REAL	DINT_TO_REAL
	DINT to INT	DINT_TO_INT
	DINT to BOOL	DINT_TO_BOOL
	REAL to DINT	REAL_TO_DINT
	REAL to INT	REAL_TO_INT
	REAL to BOOL	REAL_TO_BOOL
	To BOOL	TO_BOOL
	To INT	TO_INT
	To DINT	TO_DINT
	To floating-point number	TO_REAL

4.1.8.8 Other Instructions

Instruction	Description
ZSET	Batch setting
ZRST	Batch reset
BITW	16-Bit variable conversion to word variable
WBIT	Word variable conversion to 16-bit variable
MSET	Memory setting
MCPY	Member reset
R_TRIG	Rising edge inspection trigger
F_TRIG	Falling edge inspection trigger

4.1.8.9 Instruction Examples

Use examples of axis control instructions

MC_Power use example:

```
MC_Power(Enable := b0, //
    Axis := Axis_0, //Axis name/ID
    Status => b4, //Axis enable flag
    Busy => b6, //Busy flag
    Error => b8, //Instruction fault flag
    ErrorID => d0); //Fault code
```

MC_Jog use example:

```
MC_Jog(Enable := b0, //
    Axis := Axis_0, //Axis name/ID
    JogForward := b1, //Forward motion, with the valid level for rising edge trigger
    JogBackward := b2, //Reserve motion, with the valid level for rising edge trigger
    Velocity := d0:E, //Target speed
    Acceleration := d2:E, //Acceleration rate
    Deceleration := d4:E, //Deceleration rate
    CurveType := d6, //Curve type. 0: T-type speed curve; 1: Five-segment running curve
    Busy => b3, //Busy flag
    CommandAborted => b4, //Execution termination
    Error => b5, //Instruction fault flag
    ErrorID => d8); //Fault code
```

Use examples of mathematical operation functions

Max use example

```
c:=MAX(a,b);
```

Use examples of other instructions

R_TRIG (rising edge) use example

(*If X0 changes from FALSE to TRUE, the rising edge trigger.Q is TRUE. Execute the M0 assignment statement in the if statement. Otherwise, do not execute the statement.*)

```
Rising edge trigger(CLK := X0,Q => );
if rising edge trigger.Q THEN
    M0:=TRUE;
END_IF;
```

F_TRIG (falling edge) use example

(*If B0 changes from TRUE to FALSE, the falling edge trigger.Q is TRUE. Execute the M0 assignment statement in the if statement. Otherwise, do not execute the statement.*)

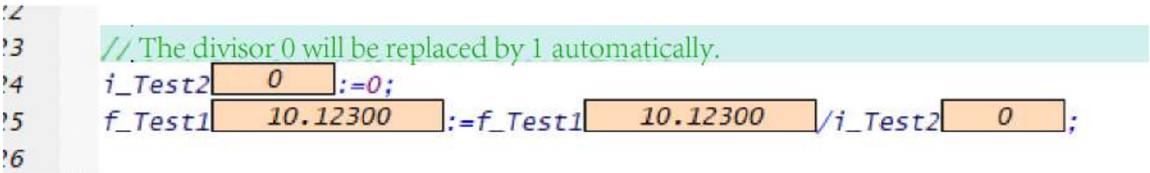
```
Falling edge trigger(CLK := B0,Q => );
if falling edge trigger.Q THEN
    M0:=TRUE;
END_IF;
```

4.1.9 Exception Protection and Handling

4.1.9.1 Division-by-zero Protection

If the divisor is 0, it automatically changes to 1.

The following figure shows the details.



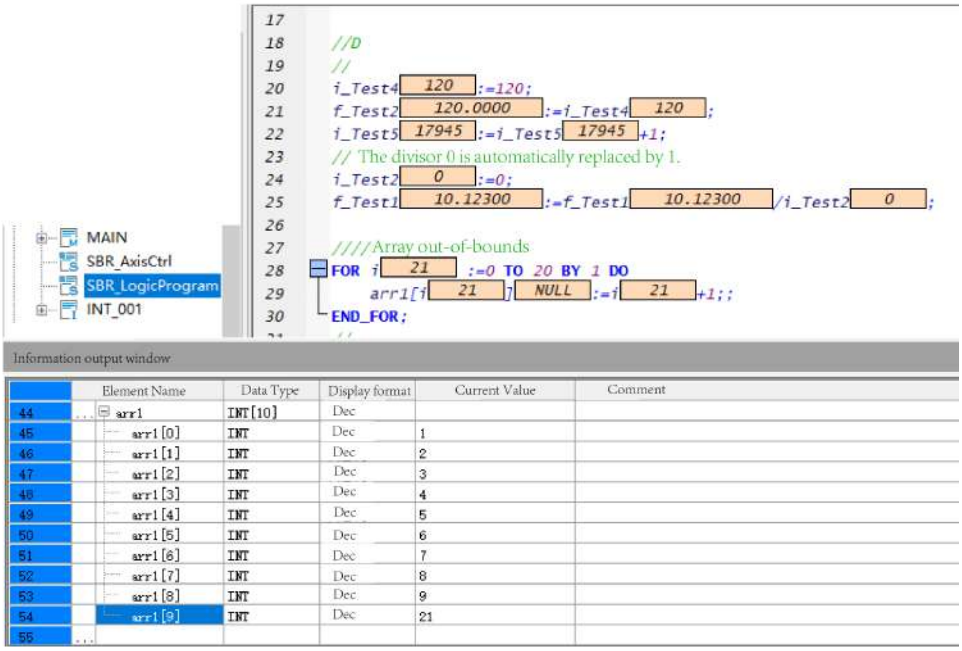
```
// The divisor 0 will be replaced by 1 automatically.
i_Test2 0 :=0;
f_Test1 10.12300 :=f_Test1 10.12300 /i_Test2 0 ;
```

The PLC LED alternatively flashes Er50 and Er81.

4.1.9.2 Array Out-of-bounds

Array out-of-bounds is automatically checked during compilation for constants and during running for variables. Values over the upper limit are stored in the element with the maximum subscript in the array, and values below the lower limit are stored in the element with the subscript 0 in the array. BOOL arrays are not checked for out-of-bounds currently.

As shown in the following figure, an INT array is defined with up to 10 elements. In case of out-of-bounds, the software reports an error and stores the values out of the range to the element with the subscript (9) in the array.



```
//D
//
i_Test4 120 :=120;
f_Test2 120.0000 :=i_Test4 120 ;
i_Test5 17945 :=i_Test5 17945 +1;
// The divisor 0 is automatically replaced by 1.
i_Test2 0 :=0;
f_Test1 10.12300 :=f_Test1 10.12300 /i_Test2 0 ;

////Array out-of-bounds
FOR i 21 :=0 TO 20 BY 1 DO
    arr1[i 21 ] NULL :=i 21 +1;;
END_FOR;
```

Information output window

	Element Name	Data Type	Display format	Current Value	Comment
44	arr1	INT[10]	Dec		
45	arr1[0]	INT	Dec	1	
46	arr1[1]	INT	Dec	2	
47	arr1[2]	INT	Dec	3	
48	arr1[3]	INT	Dec	4	
49	arr1[4]	INT	Dec	5	
50	arr1[5]	INT	Dec	6	
51	arr1[6]	INT	Dec	7	
52	arr1[7]	INT	Dec	8	
53	arr1[8]	INT	Dec	9	
54	arr1[9]	INT	Dec	21	
55	...				

The PLC LED alternatively flashes Er50 and Er80.

4.1.9.3 Infinite Loop

The program is automatically checked for an infinite loop. In case of an infinite loop, an error is reported, the infinite loop is automatically displayed, and the program stop running. The fault diagnosis page shows the error information and the error is located.

The PLC LED alternatively flashes Er1500 and Er5082.

In the error information, Er1500 indicates watchdog timeout and Er5082 indicates an infinite loop alarm.

4.1.9.4 Array Subscript Considerations

- The subscript of a constant array cannot exceed the array size.
- The subscript of an array cannot be a soft element.
- The subscript of an array in an FB or FC cannot be a global variable.
- The subscript of an array cannot be an expression.
- A variable with a complex structure can contain up to only one variable array sub-index.

4.2 Programming Language (LD)

LD is a graphical programming language. Its structure is similar to that of the circuit diagram. LD contains a series of networks (also called nodes), and each network starts from the vertical line on the left (the power rail and power flow line). A network consists of points of contact, coils, arithmetic blocks (functions, function blocks, programs, execution blocks, actions, and methods), jump instructions, labels, and connecting wires.

LD mainly includes points of contact, coils, arithmetic blocks, branches, and comments. These elements are inserted, dragged, scribed, and copied and pasted to networks to form the LD execution logic.

LD provides online commissioning functions such as monitoring, written values, force values, and breakpoints.

For details about the LD programming language, see the "AutoShop.chm". To obtain the manual, you can choose "Help" > "Help Manual" in the AutoShop menu bar and double-click "AutoShop.chm" in the "Manual" folder.

4.3 Programming Language (SFC)

SFC is a novel graphic programming language for programming according to the process flow. The icons or menu items for all SFC elements can be found in the SFC toolbar and SFC menu. You can click a specified icon to enter the required element, and set properties of the specified SFC element in the SFC input dialog box. The SFC toolbar and menu also provide shortcuts to add connecting wires. You can establish connections between SFC elements as required.

For details about the SFC programming language, see the "AutoShop.chm". To obtain the manual, you can choose "Help" > "Help Manual" in the AutoShop menu bar and double-click "AutoShop.chm" in the "Manual" folder.

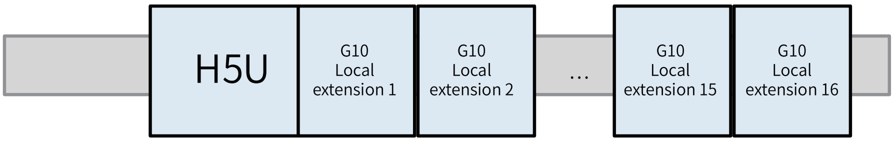
5 Extension Modules

5.1 H5U Local Extension Modules

5.1.1 Overview

H5U can carry up to 16 local extension modules, and can access local extension based on module configuration.

The following figure shows the hardware configuration for H5U to connect to local extension modules.



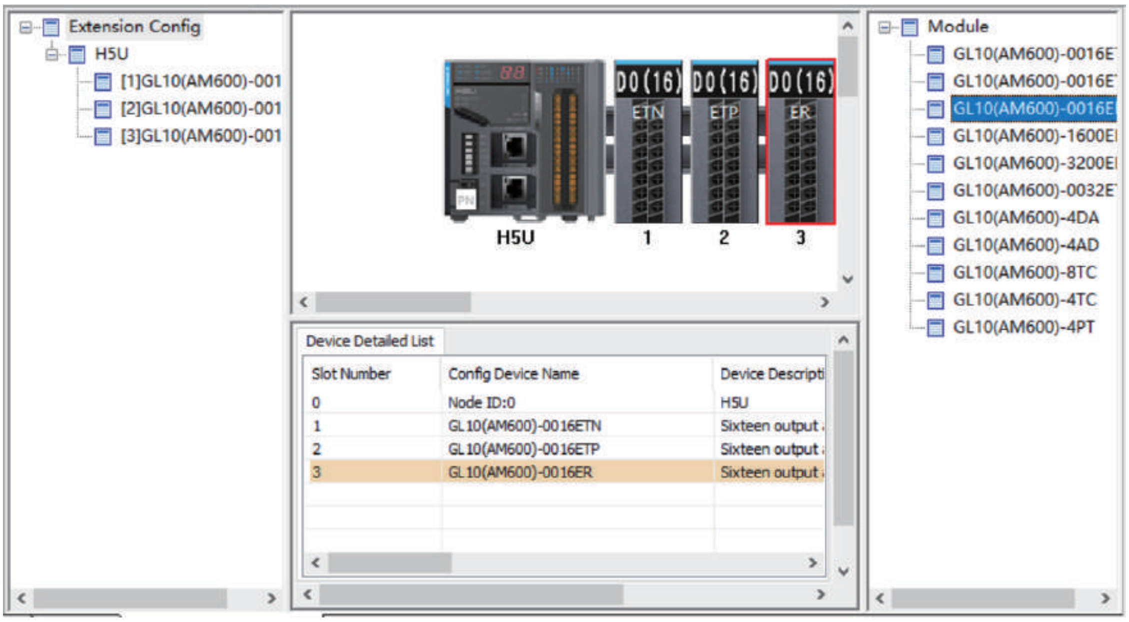
The following table lists the supported models of local extension modules.

Product	Description
GL10-0016ETP	16 digital output (DO) transistor module - PNP
GL10-0016ETN	16 DO transistor module - NPN
GL10-0016ER	16 DO relay module
GL10-1600END	16 digital input (DI) module
GL10-3200END	32 digital input (DI) module
GL10-0032ETN	32 DO module
GL10-4AD	4 analog input (AI) module
GL10-4DA	4 analog output (AO) module
GL10-4PT	4-in resistance temperature detector (RTD) module
GL10-4TC	4-in thermocouple temperature detection (TC TEMP MEAS) module
GL10-8TC	8-in TC TEMP MEAS module

5.1.2 Configuring Hardware

Local extension modules are implemented by hardware configuration. To configure hardware, perform the following steps:

1. In the "Project Manager" tree, unfold "Config" and double-click "Module Config". The "Extension Config" page is displayed.



Extension Config
- H5U
  - [1]GL10(AM600)-001
  - [2]GL10(AM600)-001
  - [3]GL10(AM600)-001

Device Detailed List

Slot Number	Config Device Name	Device Descripti
0	Node ID:0	H5U
1	GL10(AM600)-0016ETN	Sixteen output
2	GL10(AM600)-0016ETP	Sixteen output
3	GL10(AM600)-0016ER	Sixteen output

Module
- GL10(AM600)-0016E
- GL10(AM600)-0016E
- GL10(AM600)-0016E
- GL10(AM600)-1600E
- GL10(AM600)-3200E
- GL10(AM600)-0032E
- GL10(AM600)-4DA
- GL10(AM600)-4AD
- GL10(AM600)-8TC
- GL10(AM600)-4TC
- GL10(AM600)-4PT

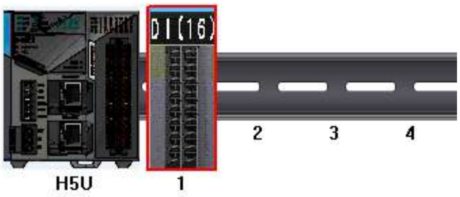
2. Click the position number of an extension module on the guide rail, double-click the module on the right or drag and move the module to the guide rail, and configure the extension module.

5.1.3 Configuring Extension Modules

5.1.3.1 DI Modules

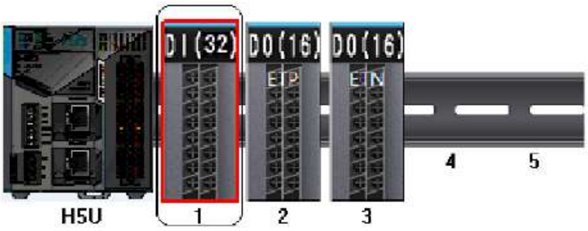
DI modules included GL10-1600END and GL10-3200END. The method of using a DI module as a local extension module is as follows:

1. In the module list, select a module to be added, and double-click the module for automatic extension on the extension rack, or drag the module to the extension rack.

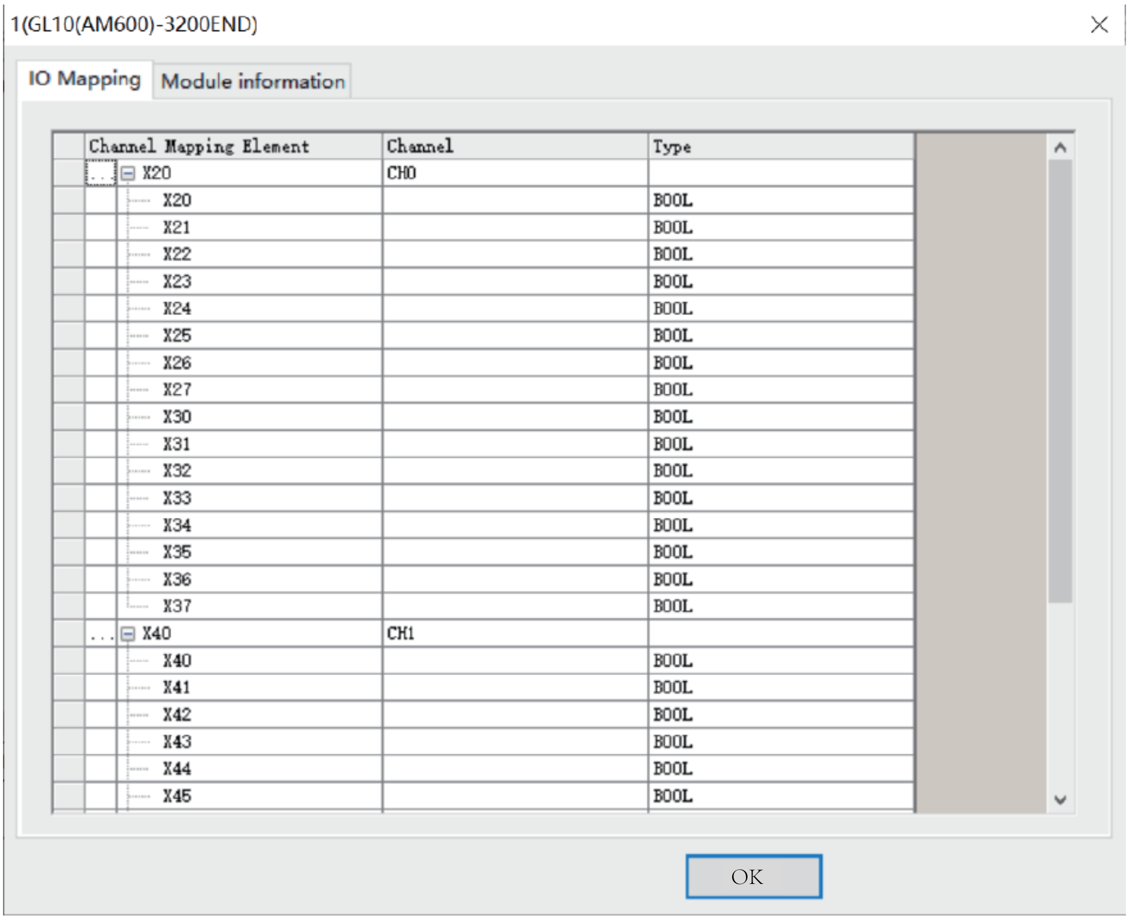


2. After local DI extension modules are connected to the master module without port configuration, numbers of input X ports on the extension modules follow the number of the input X port on the master module in sequence.
   For example, if the master module is a general H5U model, and number of the last X port on the master module is X37 after GL10-1600END connection, numbers of 16 input X ports on the extension module range from X40 to X47 and X50 to X57 during programming. This also applies to other DI extension modules by analogy.

   H5U series PLCs allow manual configuration of port numbers. You can double-click a module on the module configuration page to access the port configuration page.

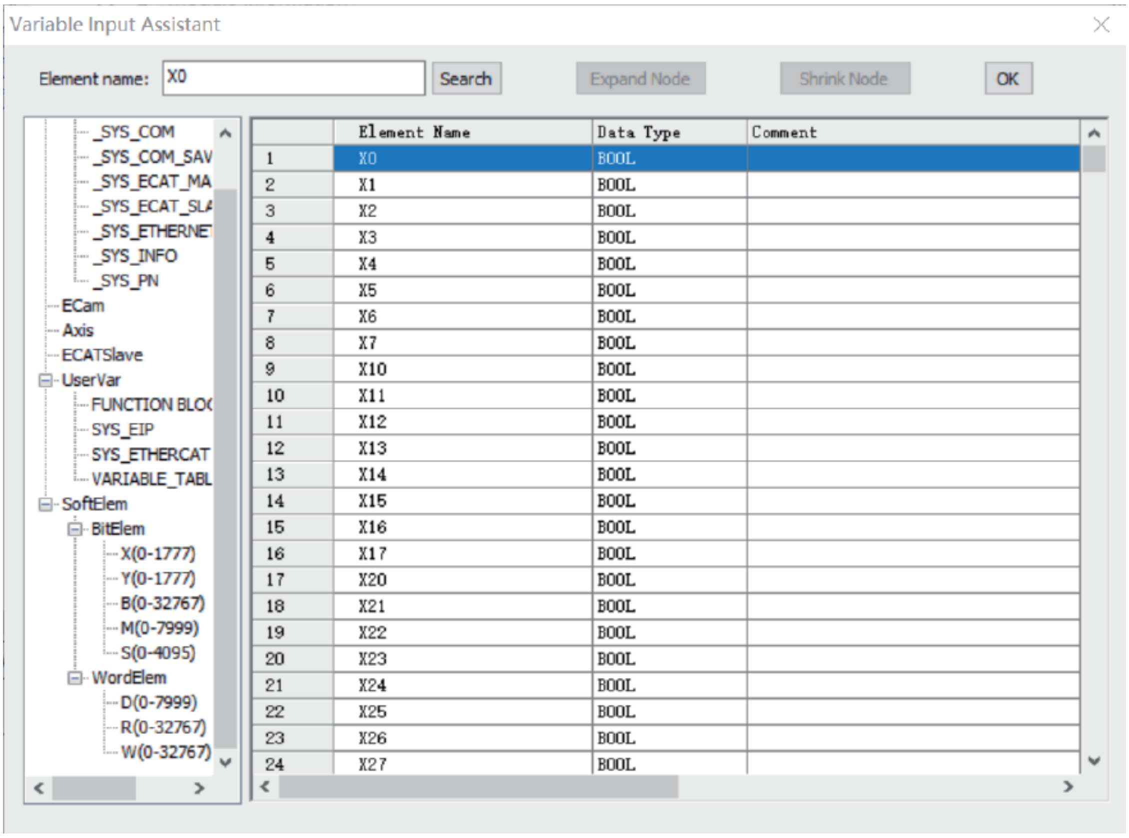


Use module 1 as an example. Double-click module 1 to access the module configuration page.



Channel Mapping Element	Channel	Type
... X20	CH0	
X20		BOOL
X21		BOOL
X22		BOOL
X23		BOOL
X24		BOOL
X25		BOOL
X26		BOOL
X27		BOOL
X30		BOOL
X31		BOOL
X32		BOOL
X33		BOOL
X34		BOOL
X35		BOOL
X36		BOOL
X37		BOOL
... X40	CH1	
X40		BOOL
X41		BOOL
X42		BOOL
X43		BOOL
X44		BOOL
X45		BOOL

Click "..." to configure the port number as required.



	Element Name	Data Type	Comment
1	X0	BOOL	
2	X1	BOOL	
3	X2	BOOL	
4	X3	BOOL	
5	X4	BOOL	
6	X5	BOOL	
7	X6	BOOL	
8	X7	BOOL	
9	X10	BOOL	
10	X11	BOOL	
11	X12	BOOL	
12	X13	BOOL	
13	X14	BOOL	
14	X15	BOOL	
15	X16	BOOL	
16	X17	BOOL	
17	X20	BOOL	
18	X21	BOOL	
19	X22	BOOL	
20	X23	BOOL	
21	X24	BOOL	
22	X25	BOOL	
23	X26	BOOL	
24	X27	BOOL	

Note

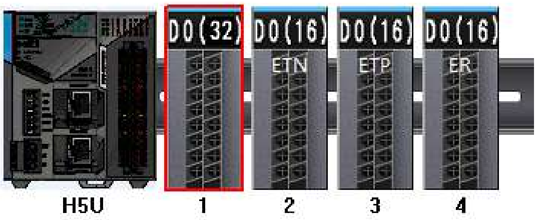
The port mapping of the H5U module is determined by the configuration. If there is no special configuration, the software arranges the port mapping in order. Even if the front-end module is deleted after the configuration is completed, the port mapping of the subsequent modules will not change.

The relay output extension module can be connected to the relay or transistor main module. Similarly, the transistor input extension module can be connected to the transistor or relay main module.

5.1.3.2 DO Modules

DO modules include GL10-0016ETP, GL10-0016ETN, GL10-0016ER, and GL10-0032ETN. The method of using a DO module as a local extension module is as follows:

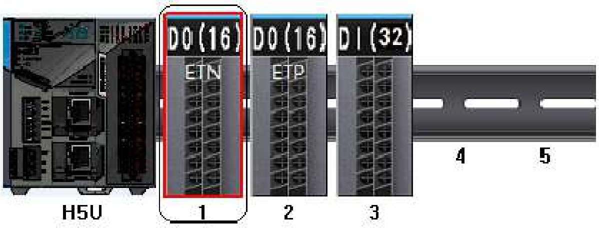
1. In the module list, select a module to be added, and double-click the module for automatic extension on the extension rack, or drag the module to the extension rack.



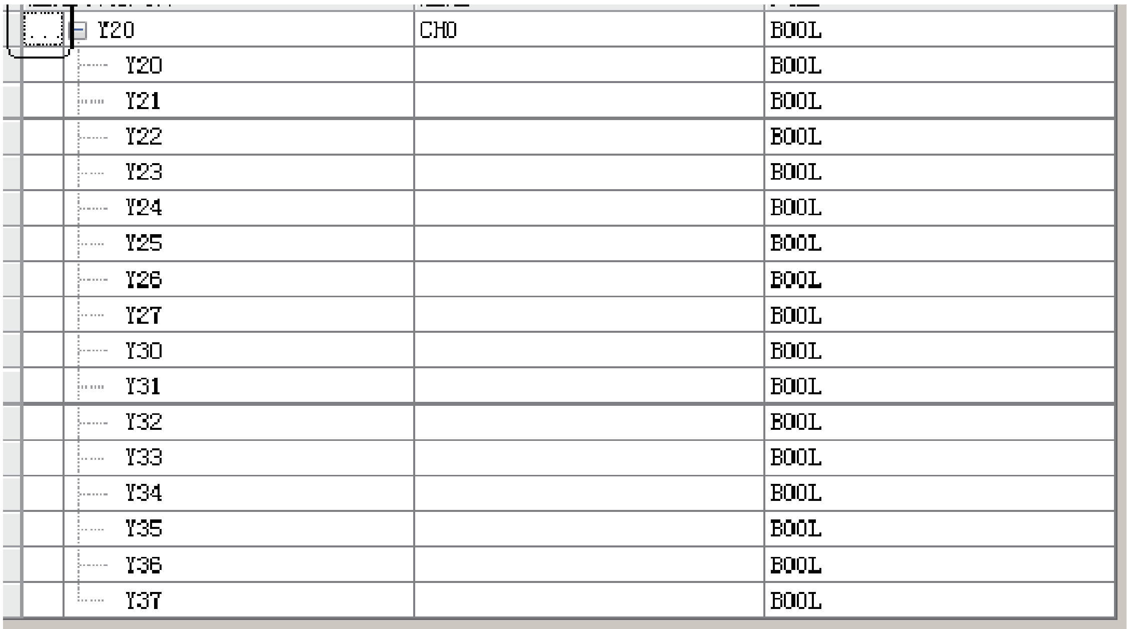
2. After local DO extension modules are connected to the master module, numbers of output Y ports on the extension modules follow the number of the Y port on the master module in sequence.

For example, if the master module is a general H5U model, and number of the last Y port on the master module is Y37 after GL10-0016END connection, numbers of 16 output Y ports on the extension module range from Y40 to Y47 and Y50 to Y57 during programming. This also applies to other DO extension modules by analogy.

H5U series PLCs allow manual configuration of port numbers. You can double-click a module on the module configuration page to access the port configuration page.

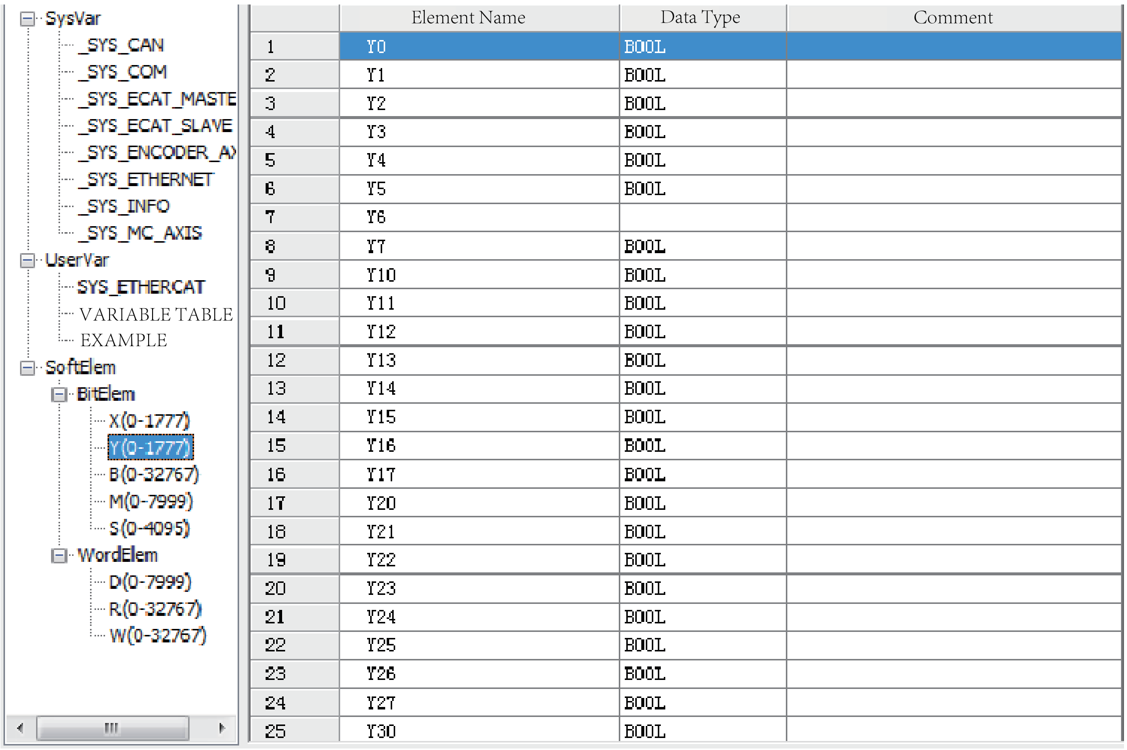


Use module 1 as an example. Double-click module 1 to access the module configuration page.



...	⊟ Y20	CH0	BOOL
	Y20		BOOL
	Y21		BOOL
	Y22		BOOL
	Y23		BOOL
	Y24		BOOL
	Y25		BOOL
	Y26		BOOL
	Y27		BOOL
	Y30		BOOL
	Y31		BOOL
	Y32		BOOL
	Y33		BOOL
	Y34		BOOL
	Y35		BOOL
	Y36		BOOL
	Y37		BOOL

Click "..." to configure the port number as required.



- SysVar
  - _SYS_CAN
  - _SYS_COM
  - _SYS_ECAT_MASTE
  - _SYS_ECAT_SLAVE
  - _SYS_ENCODER_AX
  - _SYS_ETHERNET
  - _SYS_INFO
  - _SYS_MC_AXIS
- UserVar
  - SYS_ETHERCAT
  - VARIABLE TABLE
  - EXAMPLE
- SoftElem
  - BitElem
    - X(0-1777)
    - Y(0-1777)
    - B(0-32767)
    - M(0-7999)
    - S(0-4095)
  - WordElem
    - D(0-7999)
    - R(0-32767)
    - W(0-32767)

	Element Name	Data Type	Comment
1	Y0	BOOL	
2	Y1	BOOL	
3	Y2	BOOL	
4	Y3	BOOL	
5	Y4	BOOL	
6	Y5	BOOL	
7	Y6		
8	Y7	BOOL	
9	Y10	BOOL	
10	Y11	BOOL	
11	Y12	BOOL	
12	Y13	BOOL	
13	Y14	BOOL	
14	Y15	BOOL	
15	Y16	BOOL	
16	Y17	BOOL	
17	Y20	BOOL	
18	Y21	BOOL	
19	Y22	BOOL	
20	Y23	BOOL	
21	Y24	BOOL	
22	Y25	BOOL	
23	Y26	BOOL	
24	Y27	BOOL	
25	Y30	BOOL	

Note

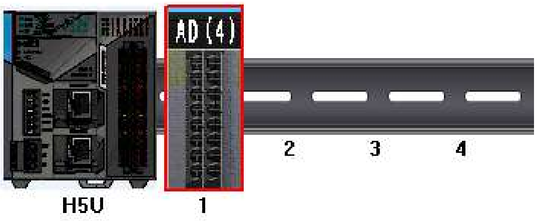
The port mapping of the H5U module is determined by the configuration. If there is no special configuration, the software arranges the port mapping in order. Even if the front-end module is deleted after the configuration is completed, the port mapping of the subsequent modules will not change.

The relay output extension module can be connected to the relay or transistor main module. Similarly, the transistor input extension module can be connected to the transistor or relay main module.

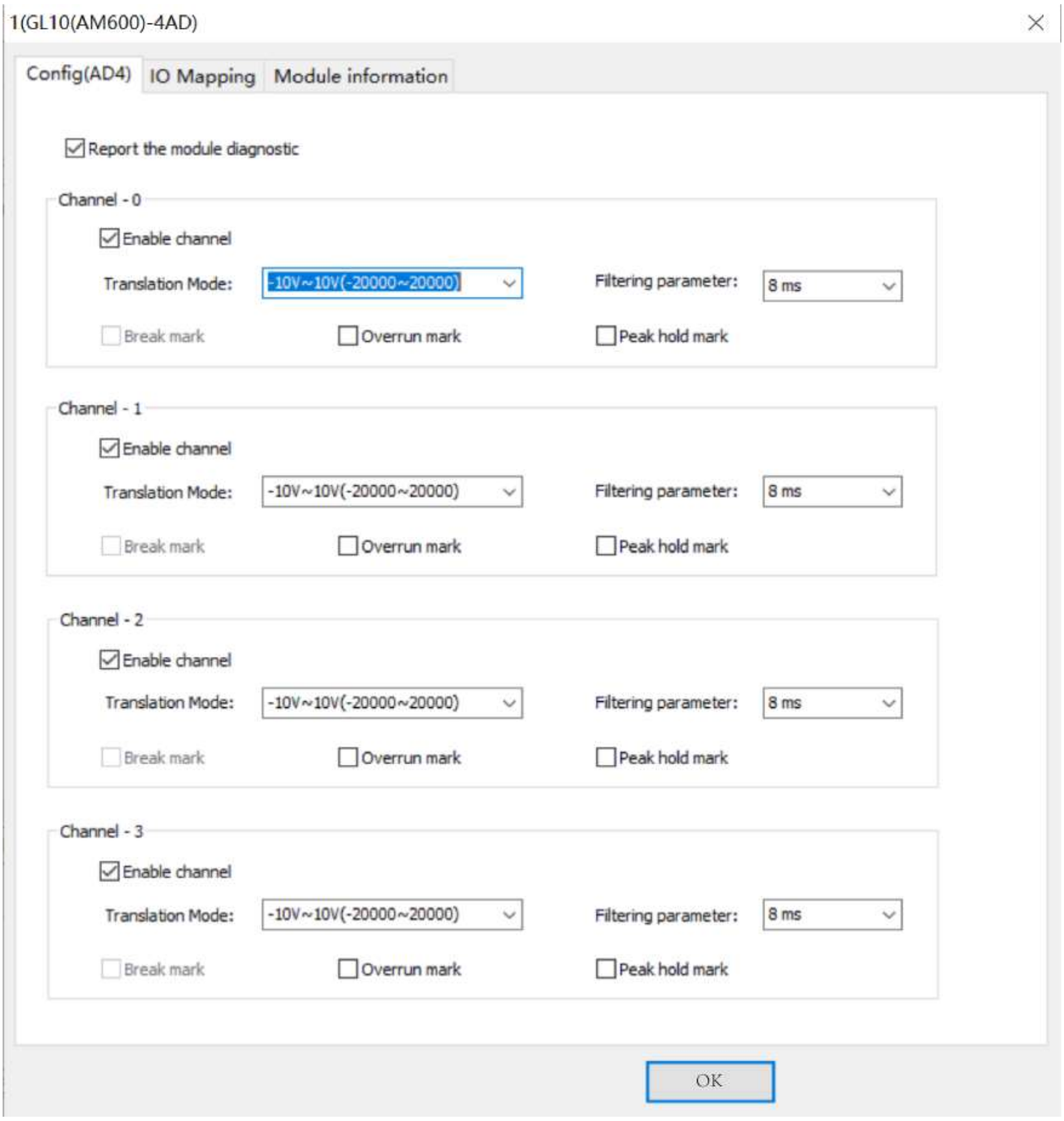
5.1.3.3 AI Modules

The method of using the GL10-4AD AI module as a local extension module is as follows:

1. In the module list, select a module to be added, and double-click the module for automatic extension on the extension rack, or drag the module to the extension rack.



2. Double-click the GL10-4AD module on the rack. The page shown in the following figure is displayed.



1(GL10(AM600)-4AD)

Config(AD4) | IO Mapping | Module information

☑ Report the module diagnostic

Channel - 0
☑ Enable channel
Translation Mode: -10V~10V(-20000~20000)
Filtering parameter: 8 ms
□ Break mark □ Overrun mark □ Peak hold mark

Channel - 1
☑ Enable channel
Translation Mode: -10V~10V(-20000~20000)
Filtering parameter: 8 ms
□ Break mark □ Overrun mark □ Peak hold mark

Channel - 2
☑ Enable channel
Translation Mode: -10V~10V(-20000~20000)
Filtering parameter: 8 ms
□ Break mark □ Overrun mark □ Peak hold mark

Channel - 3
☑ Enable channel
Translation Mode: -10V~10V(-20000~20000)
Filtering parameter: 8 ms
□ Break mark □ Overrun mark □ Peak hold mark

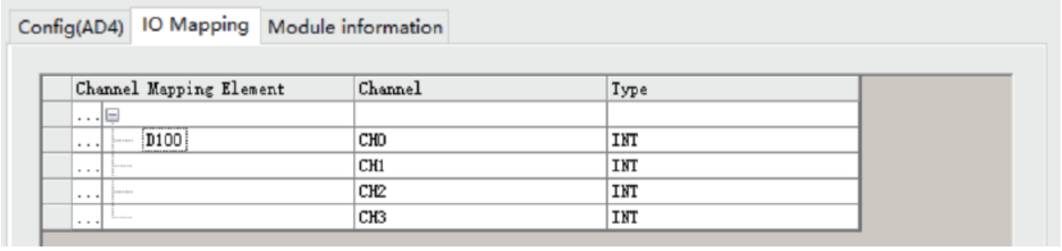
① Determine whether to select "Enable channel". If not, deselect it to save the scanning time.

② Select the corresponding span and resolution.

③ Set "Filtering parameter" to a value in the range of 1 ms to 255 ms.

④ Leave the auxiliary function items empty.

3. On the "IO Mapping" tab page, map CH0 of the 4AD module to the D element D100. In H5U, you can also map the module to a customized variable.



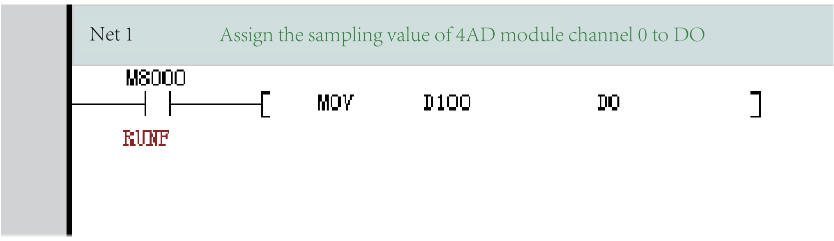
Config(AD4) | IO Mapping | Module information

Channel Mapping Element	Channel	Type
D100	CH0	INT
	CH1	INT
	CH2	INT
	CH3	INT

The following table lists the relationships between the mapped variables and actual input analog values.

Input Type	Rated Input Range	Rated Digital Value	Input Limit Range	Digital Value Limit
Analog voltage input	–10 V to +10 V	–20000 to +20000	–11 V to +11 V	–22000 to +22000
	0 V to 10 V	0 to 20000	–0.5 V to +10.5 V	–1000 to +21000
	–5 V to +5 V	–20000 to +20000	–5.5 V to +5.5 V	–22000 to +22000
	0 V to 5 V	0 to 20000	–0.25 V to +5.25 V	–1000 to +21000
	1 V to 5 V	0 to 20000	0.8 V to 5.2 V	–1000 to +21000
Analog current input	–20 mA to +20 mA	–20000 to +20000	–22 mA to +22 mA	–22000 to +22000
	0 mA to 20 mA	0 to 20000	–1 mA to +21 mA	–1000 to +21000
	4 mA to 20 mA	0 to 20000	3.2 mA to 20.8 mA	–1000 to +21000

4. Use the LD programming language to program AD sampling, and assign the voltage sampling value of CH0 from D100 to D0.



Net 1 Assign the sampling value of 4AD module channel 0 to DO

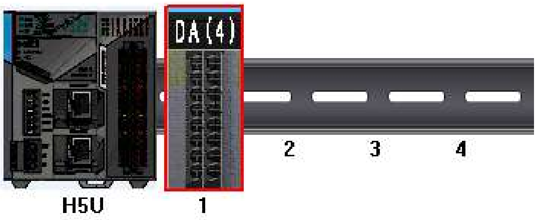
```
M8000
--| |------[ MOV  D100  D0 ]
RUNF
```

5. After compilation succeeds, download and run the project.

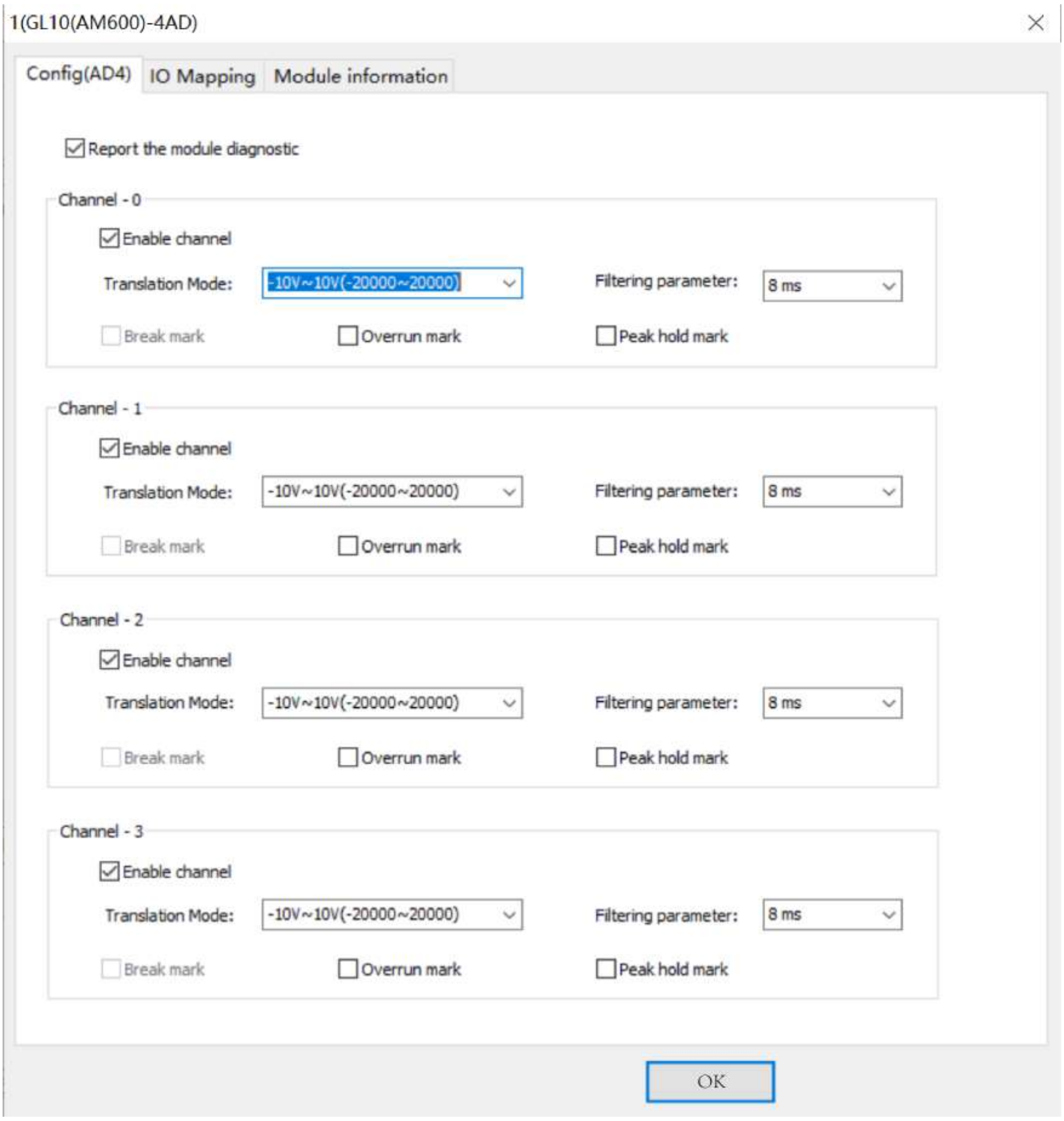
5.1.3.4 AO Modules

The method of using the GL10-4DA AO module as a local extension module is as follows:

1. In the module list, select a module to be added, and double-click the module for automatic extension on the extension rack, or drag the module to the extension rack.



2. Double-click the GL10-4DA module on the rack. The "Config(DA4)" tab page is displayed, as shown in the following figure.



1(GL10(AM600)-4AD)

☑ Report the module diagnostic

Channel - 0

☑ Enable channel

Translation Mode: -10V~10V(-20000~20000)

Filtering parameter: 8 ms

□ Break mark □ Overrun mark □ Peak hold mark

Channel - 1

☑ Enable channel

Translation Mode: -10V~10V(-20000~20000)

Filtering parameter: 8 ms

□ Break mark □ Overrun mark □ Peak hold mark

Channel - 2

☑ Enable channel

Translation Mode: -10V~10V(-20000~20000)

Filtering parameter: 8 ms

□ Break mark □ Overrun mark □ Peak hold mark

Channel - 3

☑ Enable channel

Translation Mode: -10V~10V(-20000~20000)

Filtering parameter: 8 ms

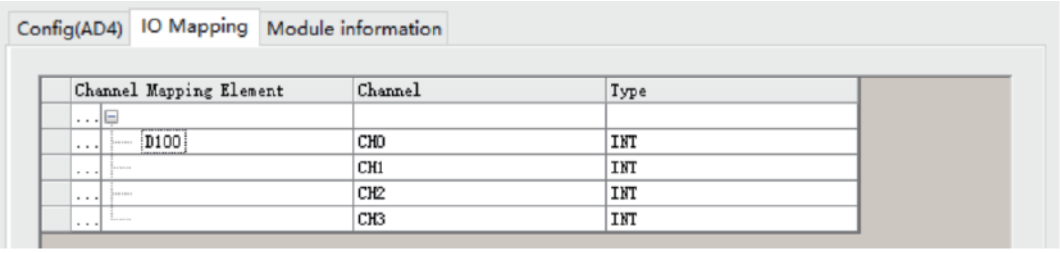
□ Break mark □ Overrun mark □ Peak hold mark

① Determine whether to select "Enable channel". If not, deselect it to save the scanning time.

② Set "Translation Mode" to select the output type and range.

③ Set "Output state after Stopping" to "Output zero", "Output Holding", or "Output preset" when the PLC is in the Stop state.

3. On the IO mapping page, map CH0 of the 4DA module to the D element D100. In H5U, you can also map the module to a customized variable.

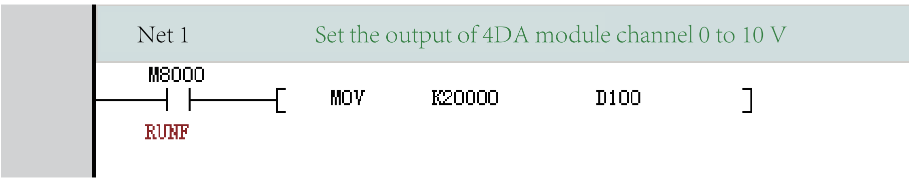


Channel Mapping Element	Channel	Type
D100	CH0	INT
	CH1	INT
	CH2	INT
	CH3	INT

The following table lists the relationships between the mapped variables and actual output analog values.

Output type	Rated Output Range	Rated Digital Value	Output Limit Range	Digital Value Limit
Analog voltage output	–10 V to +10 V	–20000 to +20000	–11 V to +11 V	–22000 to +22000
	0 V to 10 V	0 to 20000	–0.5 V to +10.5 V	–1000 to +21000
	–5 V to +5 V	–20000 to +20000	–5.5 V to +5.5 V	–22000 to +22000
	0 V to 5 V	0 to 20000	–0.25 V to +5.25 V	–1000 to +21000
	1 V to 5 V	0 to 20000	0.8 V to 5.2 V	–1000 to +21000
Analog current output	0 mA to 20 mA	0 to 20000	0 mA to 21 mA	0 to 21000
	4 mA to 20 mA	0 to 20000	3.2 mA to 20.8 mA	–1000 to +21000

4. Use the LD programming language to program the DA output. The digital values corresponding to –10 V to +10 V are –20000 to +20000, so the value 20000 is assigned for D0 and the output voltage of CH0 is +10 V.



Net 1 Set the output of 4DA module channel 0 to 10 V

```
M8000
──┤ ├────[  MOV    K20000    D100    ]
RUNF
```

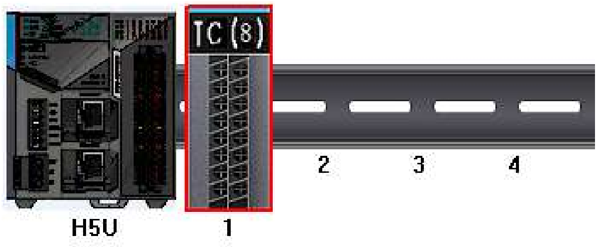
5. After compilation succeeds, download and run the project.

5.1.3.5 Temperature Detection Modules

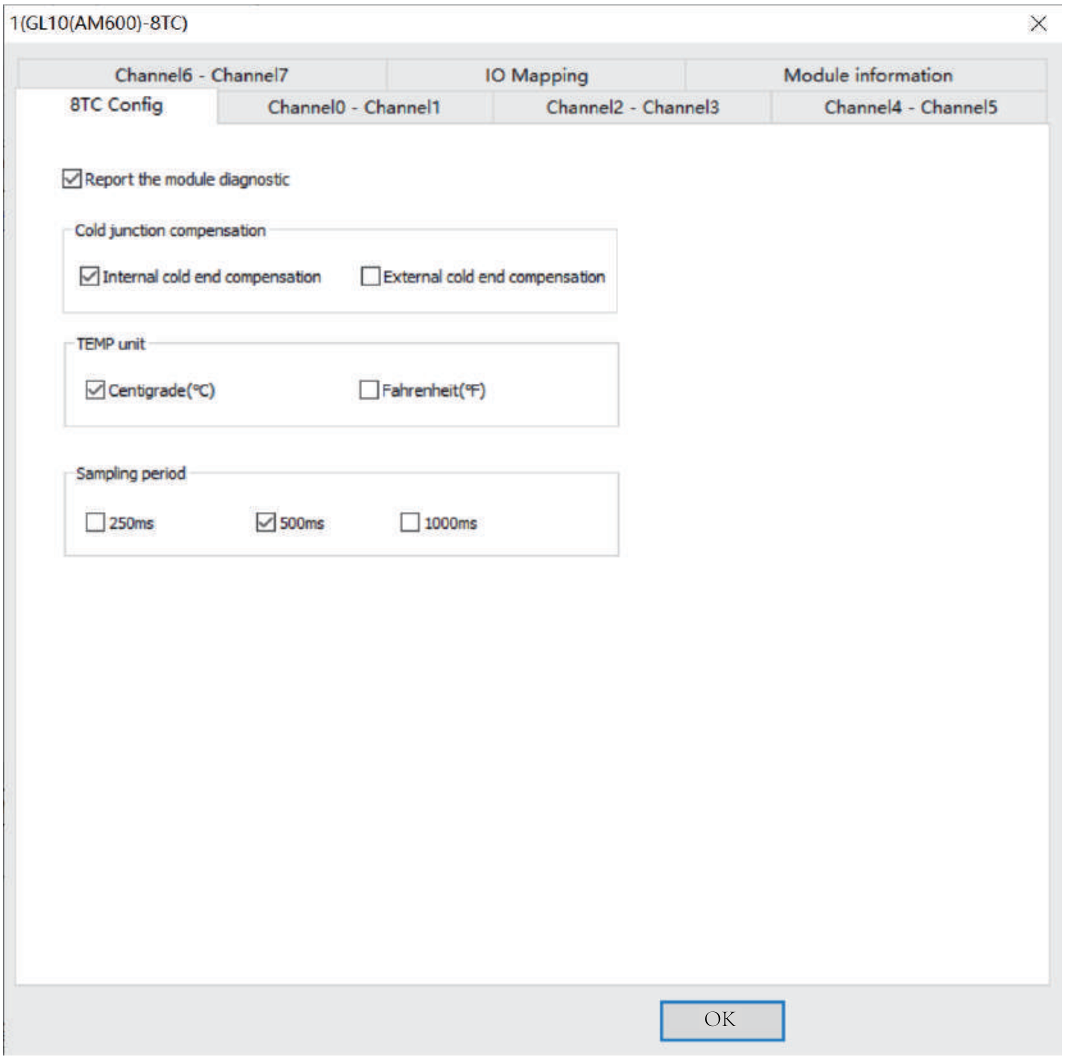
Temperature detection modules include GL10-8TC, GL10-4TC, and GL10-4PT.

Use H5U as the control master module, sample the temperature of the K thermocouple through CH0 of the GL10-8TC module, and assign the sampling value for the corresponding variable.

1. In the module list, select a module to be added, and double-click the module for automatic extension on the extension rack, or drag the module to the extension rack.



2. Double-click the GL10-8TC module on the rack. The "8TC Config" tab page is displayed, as shown in the following figure.



1(GL10(AM600)-8TC)

☑ Report the module diagnostic

Cold junction compensation

☑ Internal cold end compensation □ External cold end compensation

TEMP unit

☑ Centigrade(℃) □ Fahrenheit(℉)

Sampling period

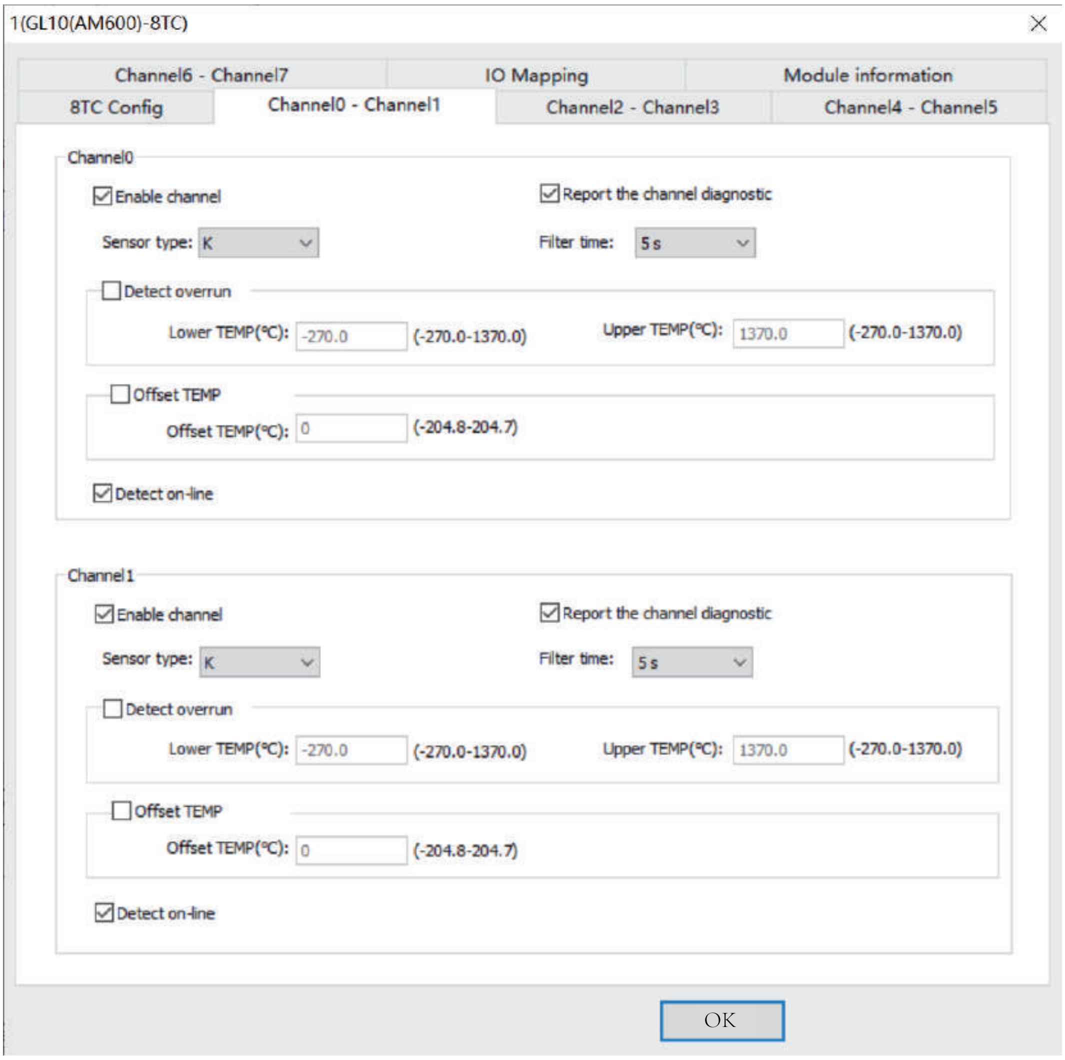
□ 250ms ☑ 500ms □ 1000ms

① To use external cold junction compensation, select "External cold end compensation". When external cold junction compensation is used, CH7 of the 8TC module is used for input of the external cold junction compensation sensor (PT100) but cannot be used to measure temperature for the thermocouple.

② Set "TEMP unit" to the corresponding temperature unit.

③ Set "Sampling period".

3. On the "CH0 –CH1" tab page, select "Enable channel' for CH0 and set "Sensor type" to "K".



Channel0

☑Enable channel ☑Report the channel diagnostic

Sensor type: K Filter time: 5 s

□Detect overrun

Lower TEMP(℃): -270.0 (-270.0-1370.0) Upper TEMP(℃): 1370.0 (-270.0-1370.0)

□Offset TEMP

Offset TEMP(℃): 0 (-204.8-204.7)

☑Detect on-line

Channel1

☑Enable channel ☑Report the channel diagnostic

Sensor type: K Filter time: 5 s

□Detect overrun

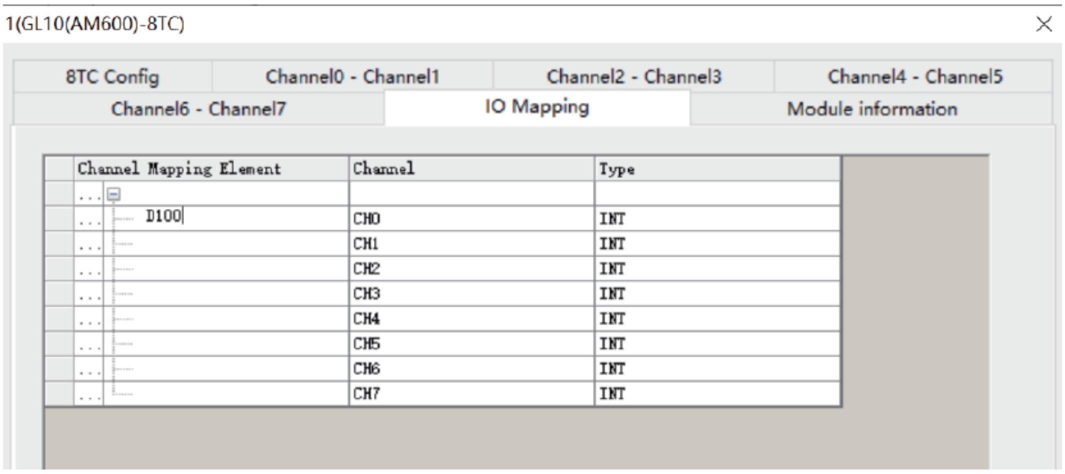
Lower TEMP(℃): -270.0 (-270.0-1370.0) Upper TEMP(℃): 1370.0 (-270.0-1370.0)

□Offset TEMP

Offset TEMP(℃): 0 (-204.8-204.7)

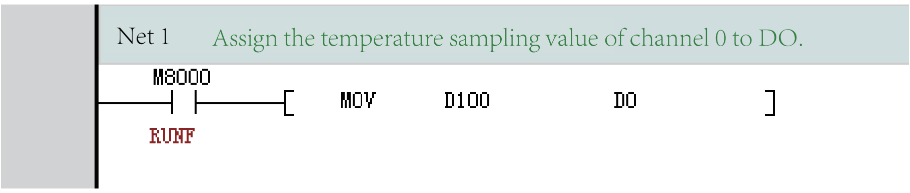
☑Detect on-line

4. On the IO mapping page, map CH0 of the 8TC module to the D element D100. In H5U, you can also map the module to a customized variable.



Channel Mapping Element	Channel	Type
D100	CH0	INT
	CH1	INT
	CH2	INT
	CH3	INT
	CH4	INT
	CH5	INT
	CH6	INT
	CH7	INT

5. Use the LD programming language to program 8TC sampling, and assign the temperature sampling value of CH0 from D100 to D0.



Net 1 Assign the temperature sampling value of channel 0 to DO.

```
M8000
--| |--[ MOV   D100   D0 ]
RUNF
```

6. After compilation succeeds, download and run the project.

5.2 Easy Local Extension Modules and Extension Cards

5.2.1 System Variables

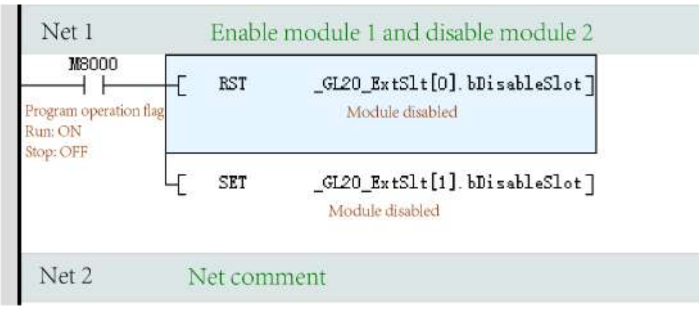
5.2.1.1 System Variables of Extension Modules

Information about the GL20 local module of _GL20_ExtSlt

Name	Data Type	Description	R/W
_GL20_ExtSlt[0].ConfigModule	DINT	Type of the configured module	R
_GL20_ExtSlt[0].MountedModule	DINT	Type of the installed module	R
_GL20_ExtSlt[0].LogicVersion	DINT	Version of the logic device	R
_GL20_ExtSlt[0].SWVersion	DINT	Software version	R
_GL20_ExtSlt[0].Error	BOOL	Error state	R
_GL20_ExtSlt[0].bDisableSlot	BOOL	Module disabled	R/W

Program example

The program enable module 1 and disable module 2 make program compilation and download take effect.



Net 1 Enable module 1 and disable module 2

```
M8000
--| |--+--[ RST   _GL20_ExtSlt[0].bDisableSlot ]
       |          Module disabled
       +--[ SET   _GL20_ExtSlt[1].bDisableSlot ]
                  Module disabled
```

Net 2 Net comment

5.2.1.2 System Variables of Extension Cards

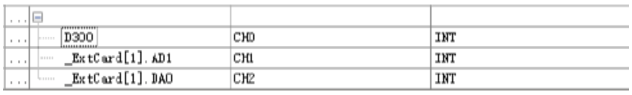
Information about the GL20 local module of _GL20_ExtSlt

Name	Data Type	Description	R/W
_ExtCard[0].ConfigModule	INT	Type of configured extension card	R
_ExtCard[0].MountedModule	INT	Type of installed extension card	R
_ExtCard[0].LogicVersion	INT	Version of the logic device	R
_ExtCard[0].SWVersion	INT	Software version	R
_ExtCard[0].Error	BOOL	Error state	R

Name	Data Type	Description	R/W
_ExtCard[0].DI0	BOOL	DI0 bit of 4 DI extension card	R/W
_ExtCard[0].DI1	BOOL	DI1 bit of 4 DI extension card	R/W
_ExtCard[0].DI2	BOOL	DI2 bit of 4 DI extension card	R/W
_ExtCard[0].DI3	BOOL	DI3 bit of 4 DI extension card	R/W
_ExtCard[0].DO0	BOOL	DO0 bit of 4 DO extension card	R/W
_ExtCard[0].DO1	BOOL	DO1 bit of 4 DO extension card	R/W
_ExtCard[0].DO2	BOOL	DO2 bit of 4 DO extension card	R/W
_ExtCard[0].DO3	BOOL	DO3 bit of 4 DO extension card	R/W
_ExtCard[0].AD0	INT	AD0 channel of analog extension card	R/W
_ExtCard[0].AD1	INT	AD1 channel of analog extension card	R/W
_ExtCard[0].DA0	INT	DA0 channel of analog extension card	R/W
_ExtCard[0].ConfigData	INT[16]	Extension card configuration data	R

4 DI is the default mapping address of the input extension card. 4 DO is the default mapping address of the output extension card. AD and DA are default mapping addresses of analog extension cards in the same program control way as modules.

Configure I/O mapping for GE20-2AD1DA-I, as shown in the following figure.

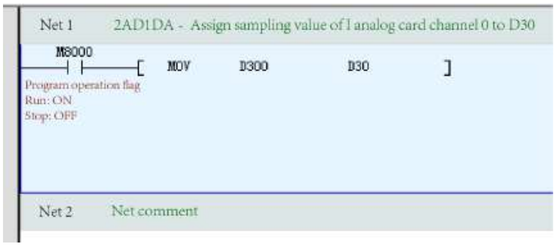


...			
...	D300	CH0	INT
...	_ExtCard[1].AD1	CH1	INT
...	_ExtCard[1].DA0	CH2	INT

Note

AutoShop earlier than version V4.8.1.0 supports modification of the GE20-2AD1DA-I/GE20-2AD1DA-V channel mapping elements. AutoShop V4.8.1.0 and later versions do not support the modification.

Use the LD programming language to program GE20-2AD1DA-I sampling, and assign the voltage sampling value of CH0 from D300 to D30, which can be compiled and downloaded for running, as shown in the following figure.



Net 1 2AD1DA - Assign sampling value of I analog card channel 0 to D30

M8000 [MOV D300 D30]

Program operation flag
Run: ON
Stop: OFF

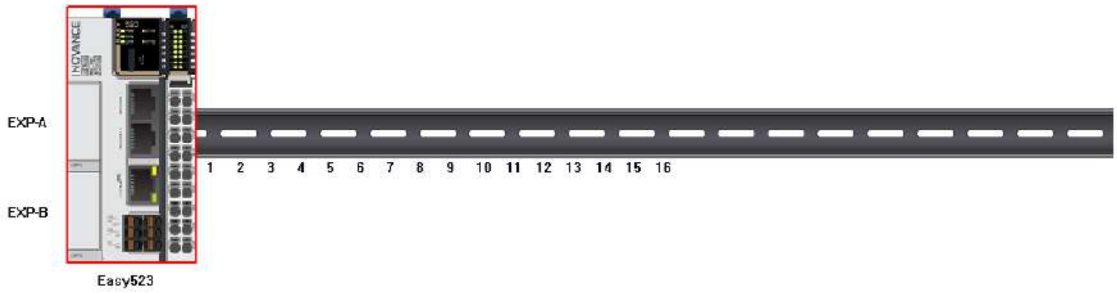
Net 2 Net comment

5.2.2 Local Extension Modules

5.2.2.1 Overview

An Easy series host can carry up to 16 modules to access local modules based on module configuration.

The following figure shows how to connect an Easy series host to a local module.



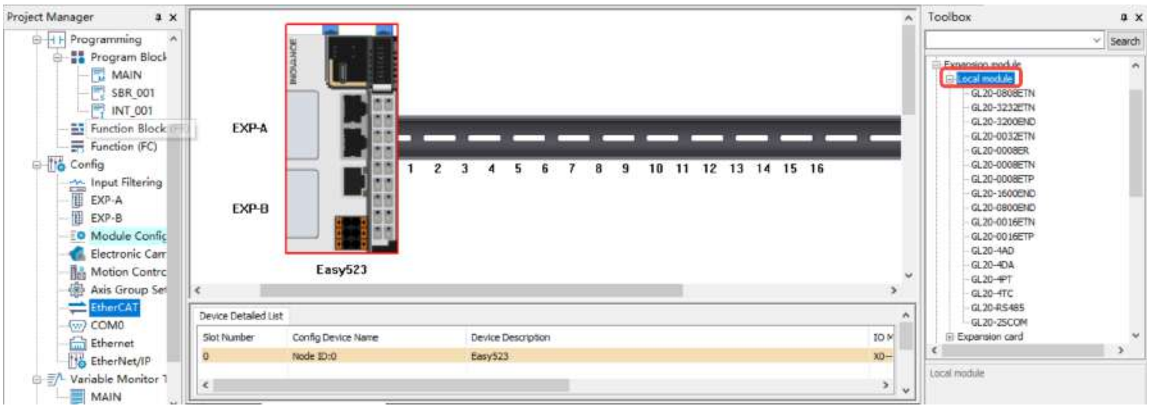
The following table lists the supported modules of local modules.

Model	Description
GL20-1600END	16 DI module
GL20-0800END	8 DI module
GL20-3200END	32 DI module
GL20-0016ETN	16 DO module
GL20-0016ETP	16 DO module
GL20-0008ETN	8 DO module
GL20-0008ETP	8 DO module
GL20-0008ER	8 DO module
GL20-0032ETN	32 DO module
GL20-0808ETN	8 DI module and 8 DO module
GL20-3232ETN	32 DI module and 32 DO module
GL20-4AD	4 AI module
GL20-4DA	4 AO module
GL20-4PT	4-in RTD module
GL20-4TC	4-in TC TEMP MEAS module
GL20-2SCOM	2 serial communication module (configurable for RS232, RS422, and RS485)
GL20-2S485	2 RS485 serial communication module

5.2.2.2 Configuring Hardware

Local extension modules are implemented by hardware configuration. To configure hardware, perform the following steps:

1. Double-click "Module Config" in AutoShop to access the configuration page, or unfold the extension module or local module node in the toolbox.
2. Double-click the corresponding local module. On the configuration page that is displayed, add the corresponding module.



Device Detailed List

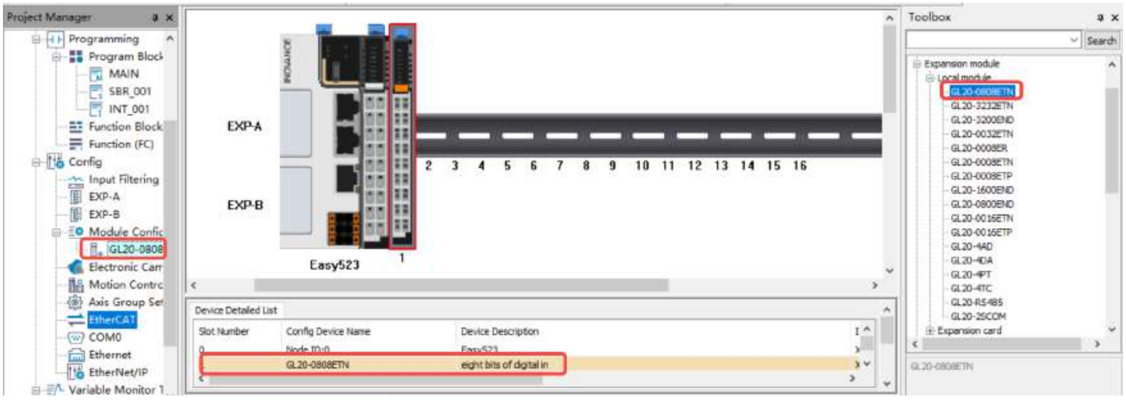
Slot Number	Config Device Name	Device Description	IO M
0	Node ID:0	Easy523	X0–

5.2.2.3 Configuring Extension Modules

5.2.2.3.1 DI Modules

DI modules include GL20-1600END, GL20-0800END, and GL20–3200END. The method of using the a DI module as a local extension module is as follows:

1. In the toolbox, select a module and double-click it. The module is automatically added to the corresponding configuration position.



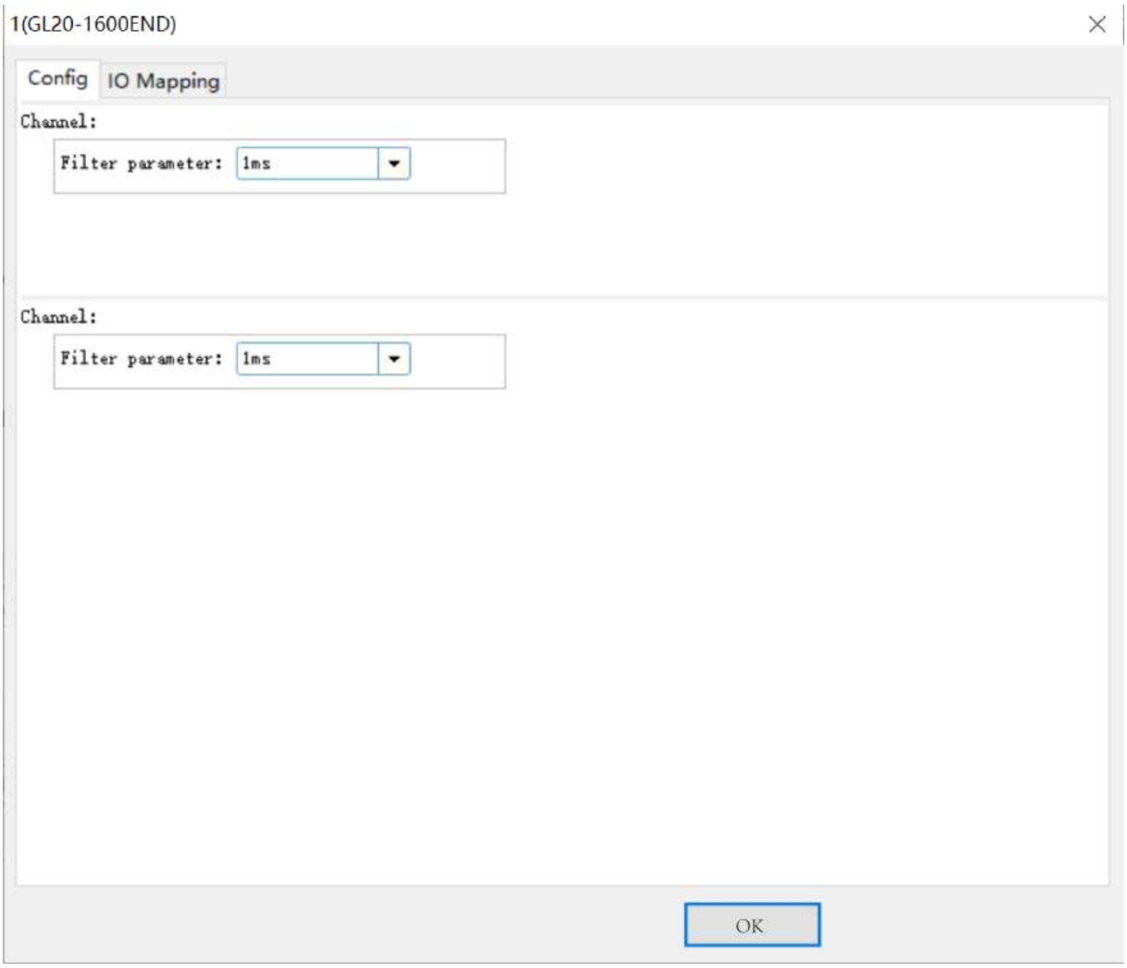
Device Detailed List

Slot Number	Config Device Name	Device Description	I
0	Node ID:0	Easy523	X
	GL20-0808ETN	eight bits of digital in	X

2. After local DI extension modules are connected to the master module without port configuration, numbers of input X ports on the extension modules follow the number of the input X port on the master module in sequence.
   For example, if the master module is an Easy series module, and number of the last X port on the master module is X37 after GL20-1600END connection, numbers of 16 input X ports on the extension module range from X40 to X47 and X50 to X57 during programming. This also applies to other DI extension modules by analogy. Taking GL20-1600END of module 1 as an example, access the configuration page in any of the following ways: ① Double-click the subnode under "Module Config". ② Double-click the module configuration on the configuration page. ③ Double-click an item in "Device Detailed List".

   The dialog box that is displayed contains the "Configure" tab page and "IO Mapping" tab page.

   The GL20-1600END series 16 DI module is used as an example. Two channels are provided on the "IO Mapping" tab page, each of which maps eight consecutive I/O elements. Two channels are also provided on the "Configure" tab page, each of which has a filter parameter. Set "Filtering parameter" as required, as shown in the following figure.



1(GL20-1600END)

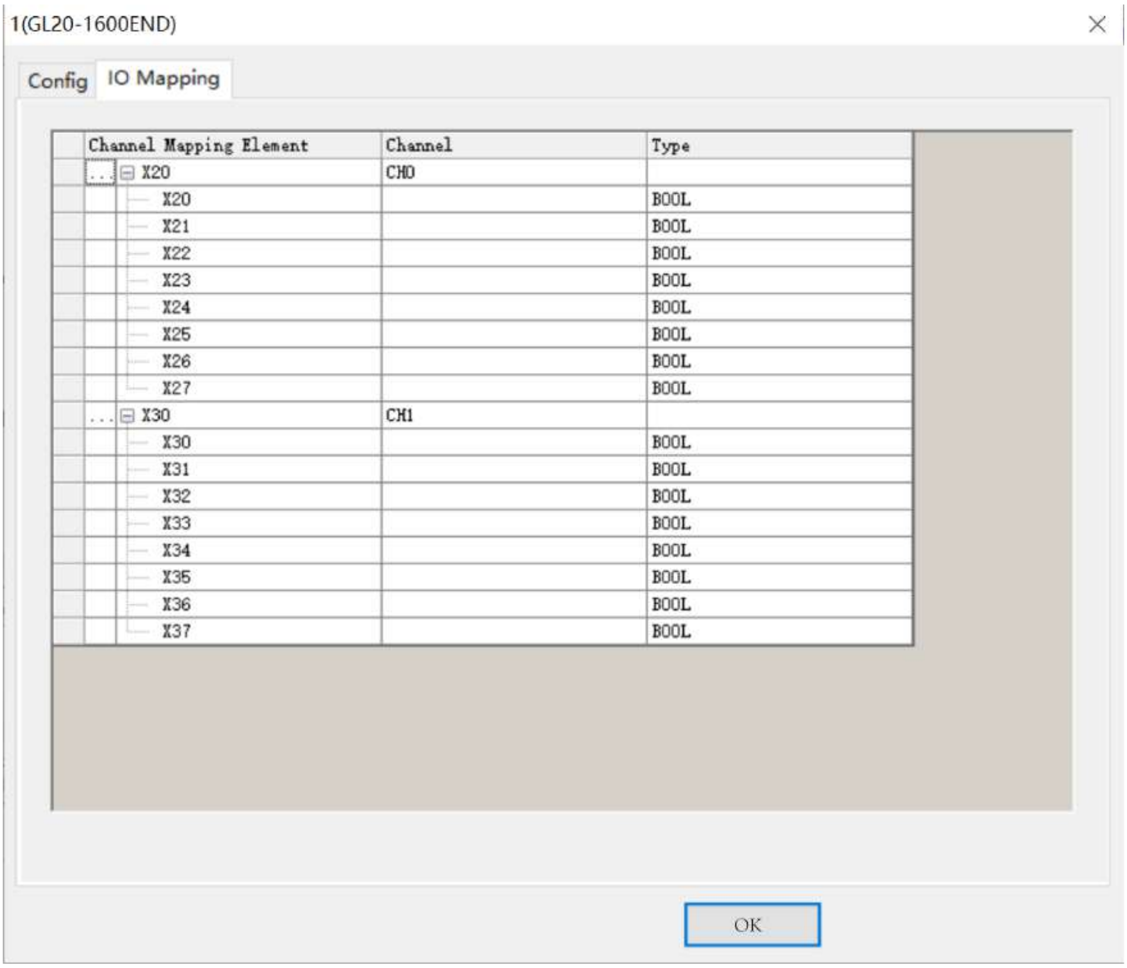
Channel:

Filter parameter: 1ms

Channel:

Filter parameter: 1ms

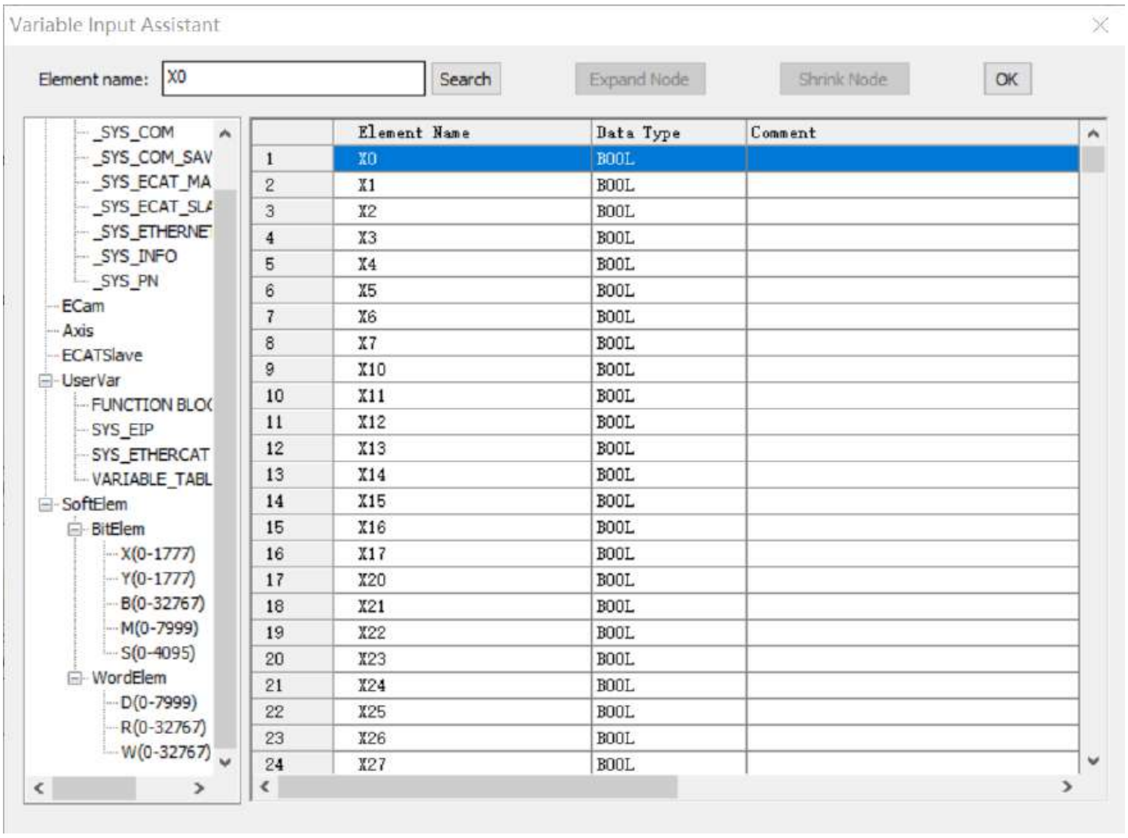
On the "IO Mapping" tab page, click "..." to pop up the variable assistant, or double-click the first mapped node such as X10 and X20 and enter the modified mapping element to modify the corresponding mapping, as shown in the following figure.



1(GL20-1600END)

Channel Mapping Element	Channel	Type
... X20	CH0	
X20		BOOL
X21		BOOL
X22		BOOL
X23		BOOL
X24		BOOL
X25		BOOL
X26		BOOL
X27		BOOL
... X30	CH1	
X30		BOOL
X31		BOOL
X32		BOOL
X33		BOOL
X34		BOOL
X35		BOOL
X36		BOOL
X37		BOOL

Click "..." to pop up the variable assistant and locate the corresponding element for mapping, as shown in the following figure.



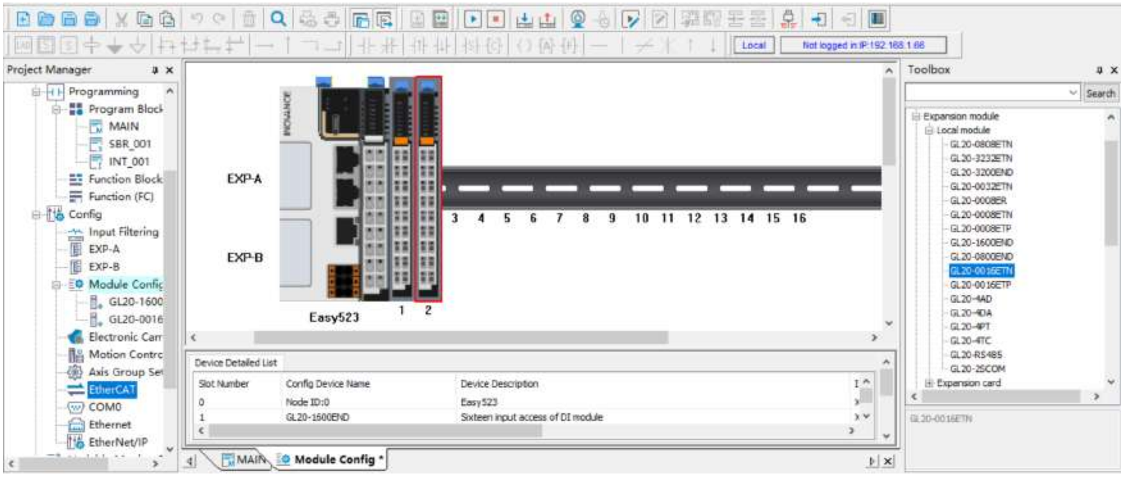
Element name: X0

	Element Name	Data Type	Comment
1	X0	BOOL	
2	X1	BOOL	
3	X2	BOOL	
4	X3	BOOL	
5	X4	BOOL	
6	X5	BOOL	
7	X6	BOOL	
8	X7	BOOL	
9	X10	BOOL	
10	X11	BOOL	
11	X12	BOOL	
12	X13	BOOL	
13	X14	BOOL	
14	X15	BOOL	
15	X16	BOOL	
16	X17	BOOL	
17	X20	BOOL	
18	X21	BOOL	
19	X22	BOOL	
20	X23	BOOL	
21	X24	BOOL	
22	X25	BOOL	
23	X26	BOOL	
24	X27	BOOL	

5.2.2.3.2 DO Modules

DO modules include GL20-0016ETN, GL20-0016ETP, GL20-0008ETN, GL20-0008ETP, GL20-0008ER, and GL20-0032ETN. The method of using a DO module as a local extension module is as follows:

1. In the toolbox, select a module and double-click it. The module is automatically added to the corresponding configuration position.



Device Detailed List

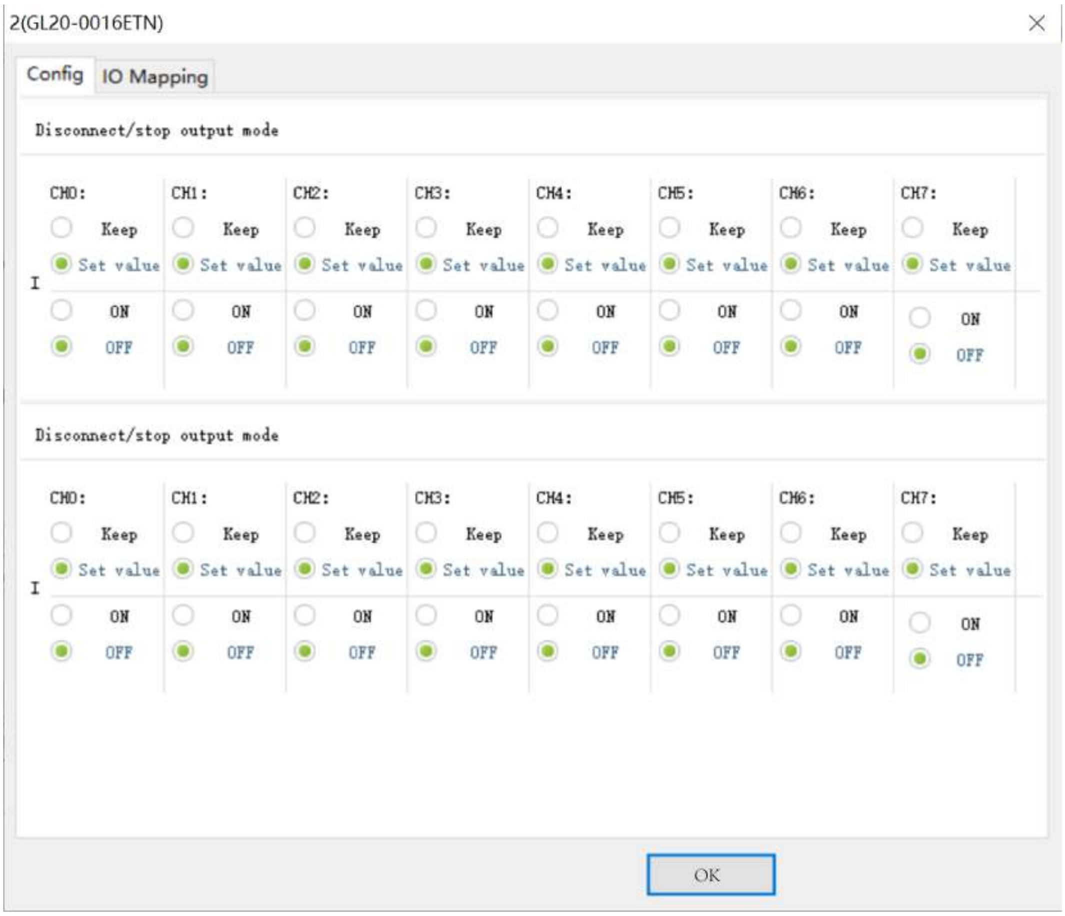
Slot Number	Config Device Name	Device Description
0	Node ID:0	Easy523
1	GL20-1600END	Sixteen input access of DI module

2. After local DO extension modules are connected to the master module without port configuration, numbers of input Y ports on the extension modules follow the number of the input Y port on the master module in sequence.
For example, if the master module is an Easy series module, and number of the last Y port on the master module is Y37 after GL20-0016ETN connection, numbers of 16 input Y ports on the extension module range from Y40 to Y47 and Y50 to Y57 during programming. This also applies to other DO

extension modules by analogy. Taking GL20-0016ETN of module 2 as an example, access the configuration page in any of the following ways: ① Double-click the subnode under "Module Config". ② Double-click the module configuration on the configuration page. ③ Double-click an item in "Device Detailed List".

The dialog box that is displayed contains the "Configure" tab page and "IO Mapping" tab page.

The GL20-0016ETN series 16 DO module is used as an example. Two channels are provided on the "IO Mapping" tab page, each of which maps eight consecutive I/O elements. Two channels are also provided on the "Configure" tab page, each of which is configured with the disconnection and output stop modes for eight I/O elements. You can set or retain the output values of I/O ports when the PLC is disconnected or stopped, as shown in the following figure.



2(GL20-0016ETN)

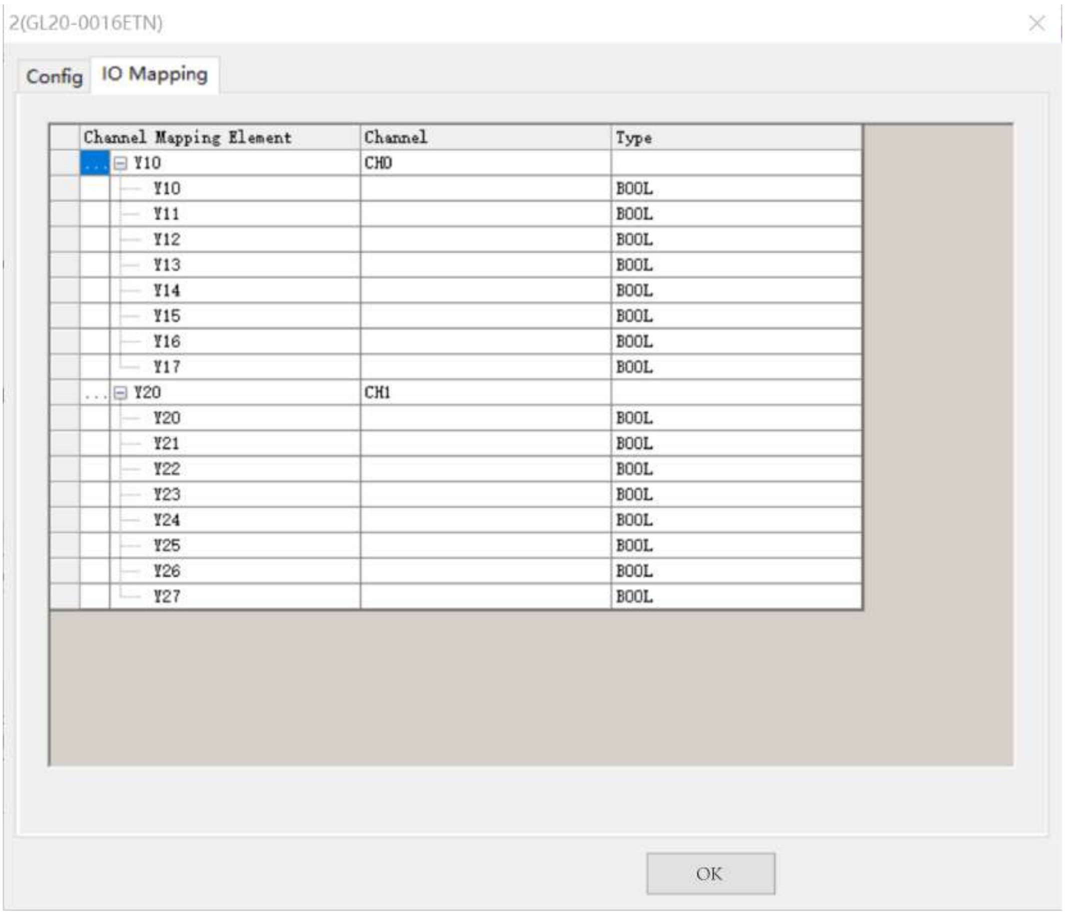
Disconnect/stop output mode

CH0:	CH1:	CH2:	CH3:	CH4:	CH5:	CH6:	CH7:
○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep
● Set value	● Set value	● Set value	● Set value	● Set value	● Set value	● Set value	● Set value
○ ON	○ ON	○ ON	○ ON	○ ON	○ ON	○ ON	○ ON
● OFF	● OFF	● OFF	● OFF	● OFF	● OFF	● OFF	● OFF

Disconnect/stop output mode

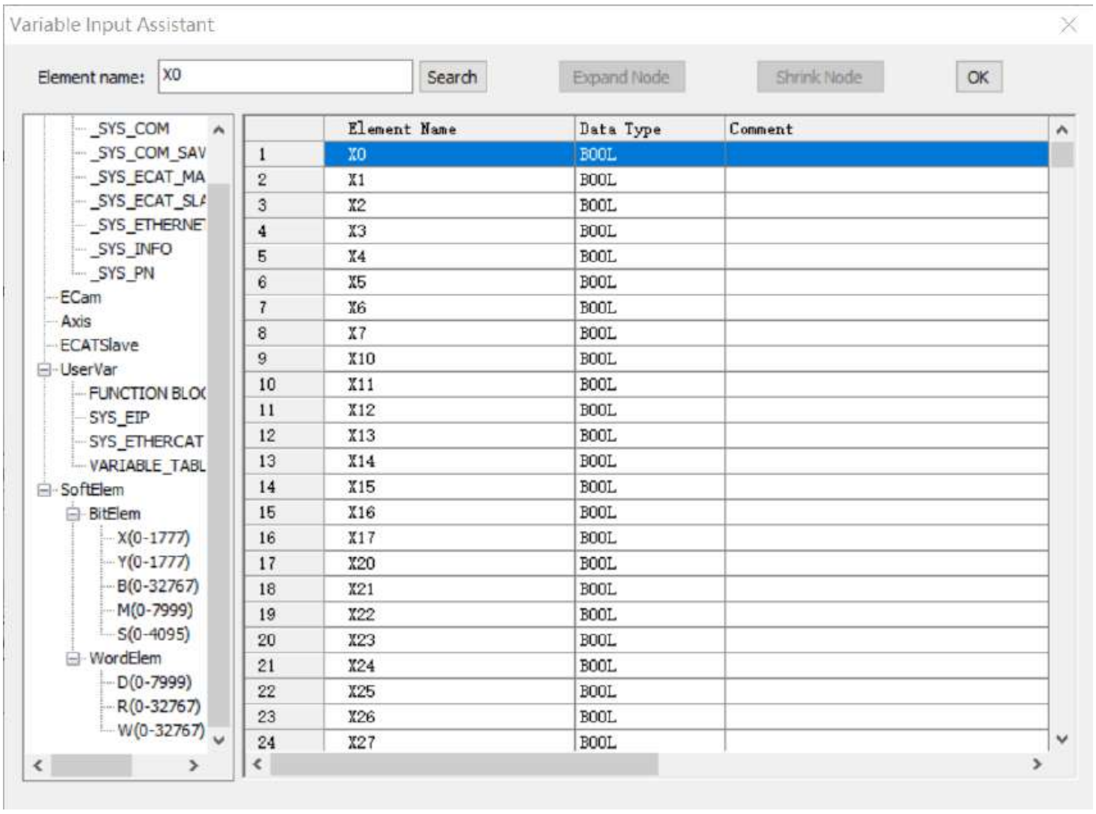
CH0:	CH1:	CH2:	CH3:	CH4:	CH5:	CH6:	CH7:
○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep
● Set value	● Set value	● Set value	● Set value	● Set value	● Set value	● Set value	● Set value
○ ON	○ ON	○ ON	○ ON	○ ON	○ ON	○ ON	○ ON
● OFF	● OFF	● OFF	● OFF	● OFF	● OFF	● OFF	● OFF

On the "IO Mapping" tab page, click "..." to pop up the variable assistant, or double-click the first mapped node such as Y10 and Y20 and enter the modified mapping element to modify the corresponding mapping, as shown in the following figure.



Channel Mapping Element	Channel	Type
... ⊟ Y10	CH0	
Y10		BOOL
Y11		BOOL
Y12		BOOL
Y13		BOOL
Y14		BOOL
Y15		BOOL
Y16		BOOL
Y17		BOOL
... ⊟ Y20	CH1	
Y20		BOOL
Y21		BOOL
Y22		BOOL
Y23		BOOL
Y24		BOOL
Y25		BOOL
Y26		BOOL
Y27		BOOL

Click "..." to pop up the variable assistant and locate the corresponding element for mapping, as shown in the following figure.



Element name: X0

	Element Name	Data Type	Comment
1	X0	BOOL	
2	X1	BOOL	
3	X2	BOOL	
4	X3	BOOL	
5	X4	BOOL	
6	X5	BOOL	
7	X6	BOOL	
8	X7	BOOL	
9	X10	BOOL	
10	X11	BOOL	
11	X12	BOOL	
12	X13	BOOL	
13	X14	BOOL	
14	X15	BOOL	
15	X16	BOOL	
16	X17	BOOL	
17	X20	BOOL	
18	X21	BOOL	
19	X22	BOOL	
20	X23	BOOL	
21	X24	BOOL	
22	X25	BOOL	
23	X26	BOOL	
24	X27	BOOL	

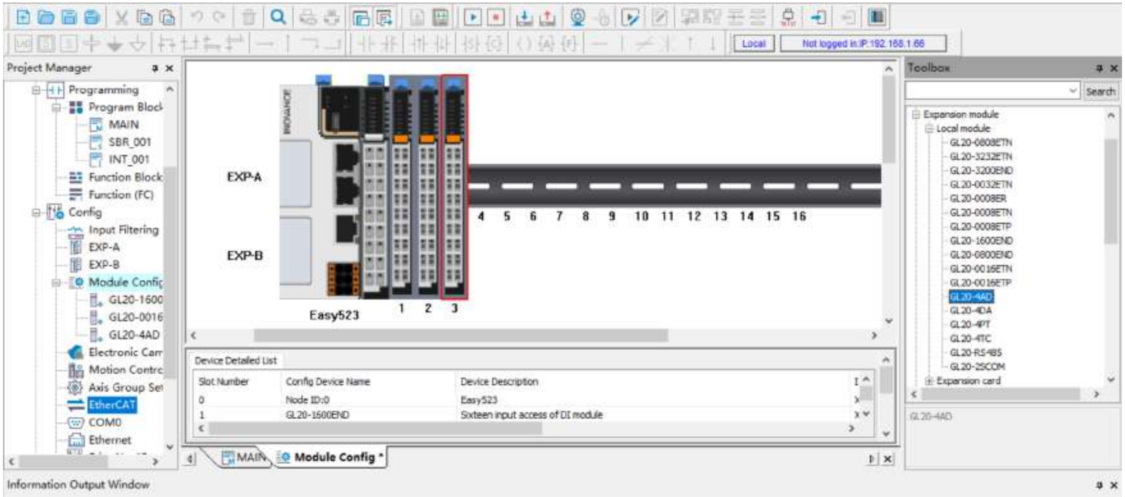
5.2.2.3.3 DI or DO Modules

DI and DO modules include GL20-0808ETN and GL20-3232ETN. GL20-0808ETN provides two channels. One channel maps eight consecutive input I/O elements, and the other channel maps eight consecutive output I/O elements, which can be considered as combination of the DI and DO modules. This also applies to GL20-3232ETN.
For details, see the use cases of DI and DO modules.

5.2.2.3.4 AI Modules

The method of using the GL20-4AD AI module as a local extension module is as follows:

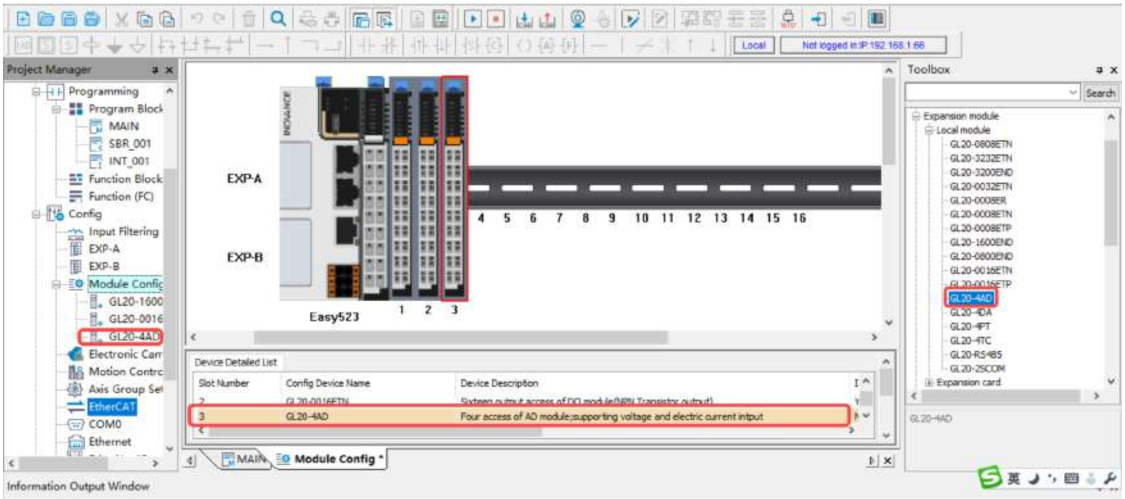
1. In the toolbox, select a module and double-click it. The module is automatically added to the corresponding configuration position.



Device Detailed List

Slot Number	Config Device Name	Device Description
0	Node ID:0	Easy523
1	GL20-1600END	Sixteen input access of DI module

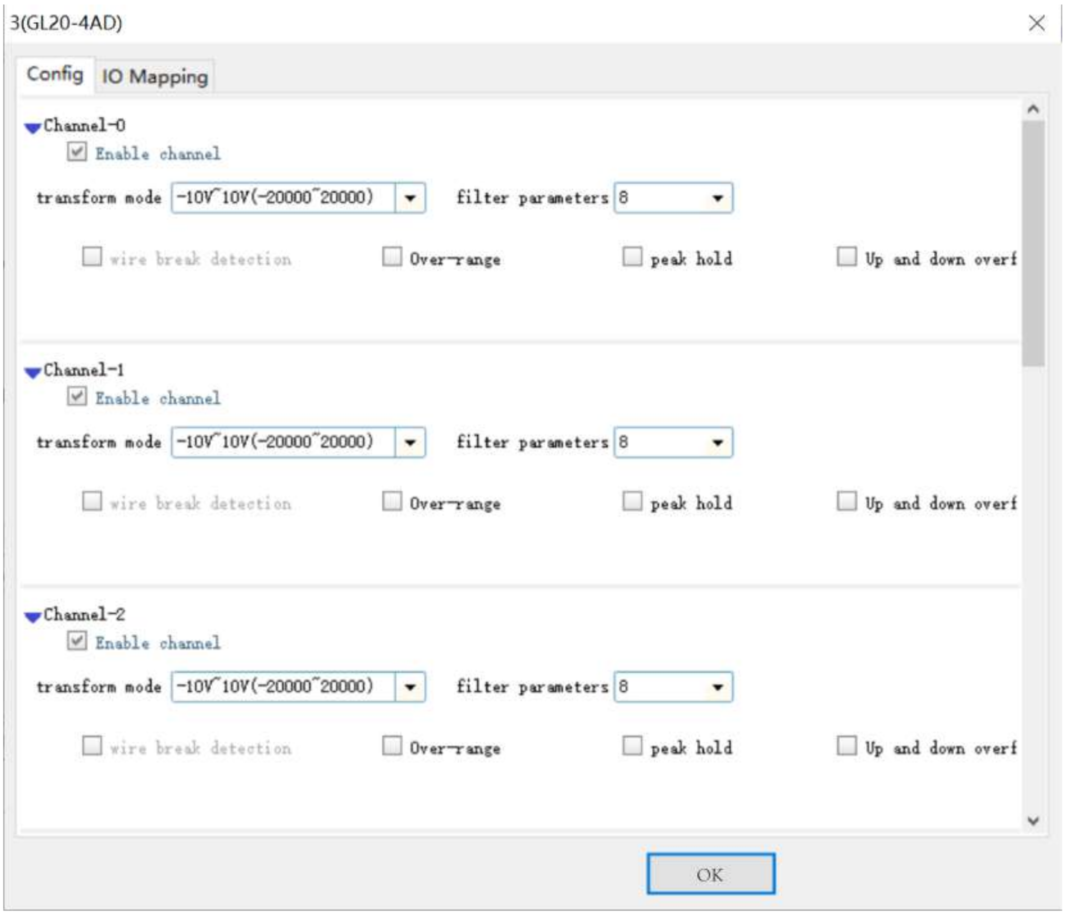
For example, if the master module is an Easy series module, connect to the GL20-4AD extension module and then access the configuration page in any of the following ways: ① Double-click the subnode under "Module Config". ② Double-click the module configuration on the configuration page. ③ Double-click an item in "Device Detailed List".



Device Detailed List

Slot Number	Config Device Name	Device Description
2	GL20-0016ETN	Sixteen output access of DO module(NPN Transistor output)
3	GL20-4AD	Four access of AD module;supporting voltage and electric current intput

2. Double-click GL20-4AD to access the parameter setting page of GL20-4AD.



3(GL20-4AD)

Config | IO Mapping

Channel-0
☑ Enable channel
transform mode -10V~10V(-20000~20000)   filter parameters 8
□ wire break detection   □ Over-range   □ peak hold   □ Up and down overf

Channel-1
☑ Enable channel
transform mode -10V~10V(-20000~20000)   filter parameters 8
□ wire break detection   □ Over-range   □ peak hold   □ Up and down overf

Channel-2
☑ Enable channel
transform mode -10V~10V(-20000~20000)   filter parameters 8
□ wire break detection   □ Over-range   □ peak hold   □ Up and down overf

GL20-4AD provides four channels, and the parameters for the four channels are set consistently. The parameters are defined as follows:

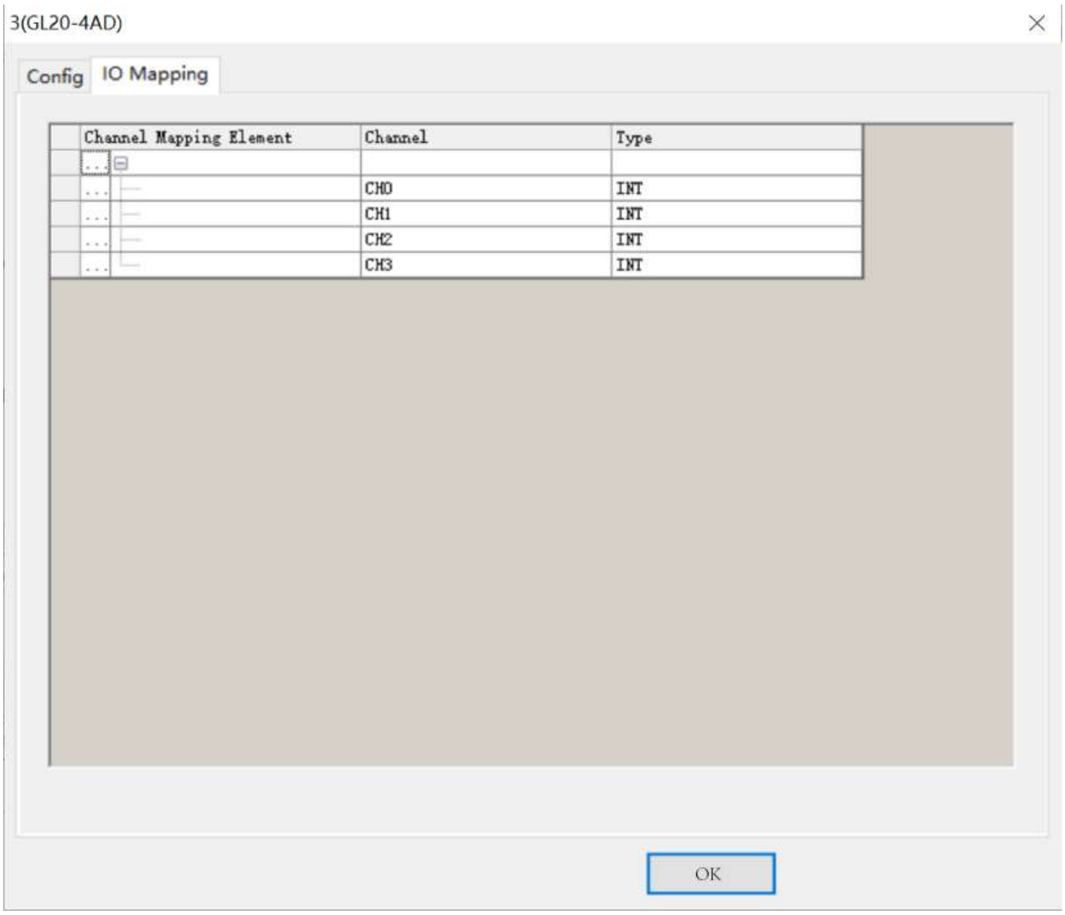
① Determine whether to select "Enable channel". If not, deselect it to save the scanning time.

② Select the corresponding span and resolution.

③ Set "Filtering parameter" to a value in the range of 1 ms to 255 ms.

④ Leave the auxiliary function items empty.

3. GL20-4AD supports consecutive I/O mapping and separate I/O mapping. Click "..." next to the parent node for consecutive mapping or double-click the node of each channel for separate mapping.

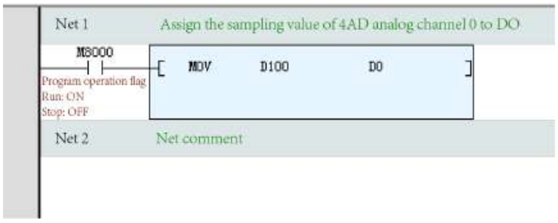


Channel Mapping Element	Channel	Type
...		
...	CH0	INT
...	CH1	INT
...	CH2	INT
...	CH3	INT

4. On the "IO Mapping" tab page, map CH0 of the 4AD module to the D element D100. In an Easy series host, you can map it to a customized variable. The following table lists the relationships between the mapped variables and actual input analog values. The EtherCAT bus coupler supports conversion of three digital spans, including –20000 to +20000, –32000 to +32000, and –27648 to +27648, while the local bus only supports conversion of the span –20000 to +20000.

Input type	Rated Input Range	Rated Digital Value	Input Limit Range	Digital Value Limit
Analog voltage input	–10 V to +10 V	–20000 to +20000	–11 V to +11 V	–22000 to +22000
	0 V to 10 V	0 to 20000	–0.5 V to +10.5 V	–1000 to +21000
	–5 V to +5 V	–20000 to +20000	–5.5 V to +5.5 V	–22000 to +22000
	0 V to 5 V	0 to 20000	–0.25 V to +5.25 V	–1000 to +21000
	1 V to 5 V	0 to 20000	0.8 V to 5.2 V	–1000 to +21000
Analog current input	–20 mA to +20 mA	–20000 to +20000	–22 mA to +22 mA	–22000 to +22000
	0 mA to 20 mA	0 to 20000	–1 mA to +21 mA	–1000 to +21000
	4 mA to 20 mA	0 to 20000	3.2 mA to 20.8 mA	–1000 to +21000

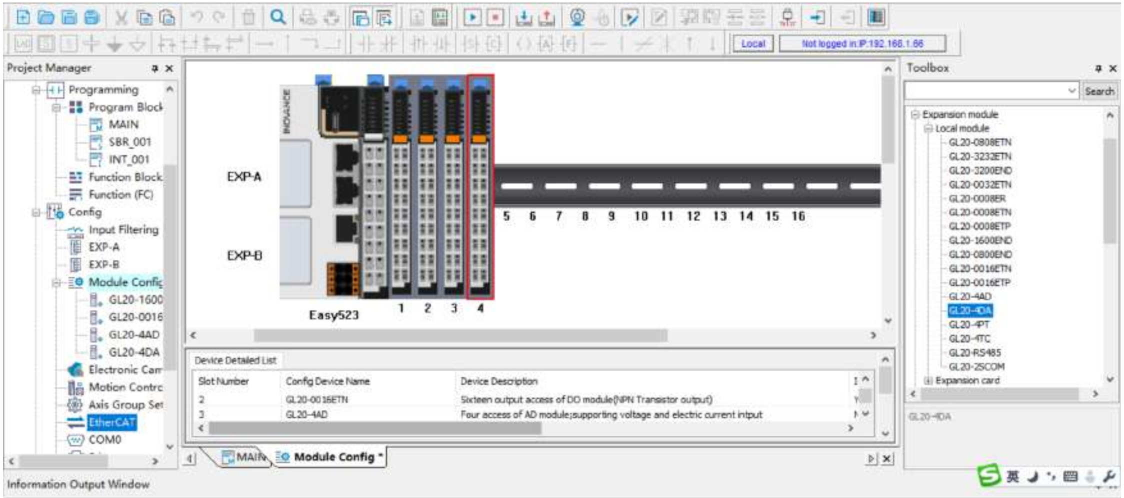
5. Use the LD programming language to program AD sampling, and assign the voltage sampling value of CH0 from D100 to D0, which can be compiled and downloaded for running.



5.2.2.3.5 AO Modules

The method of using the GL20-4DA AO module as a local extension module is as follows:

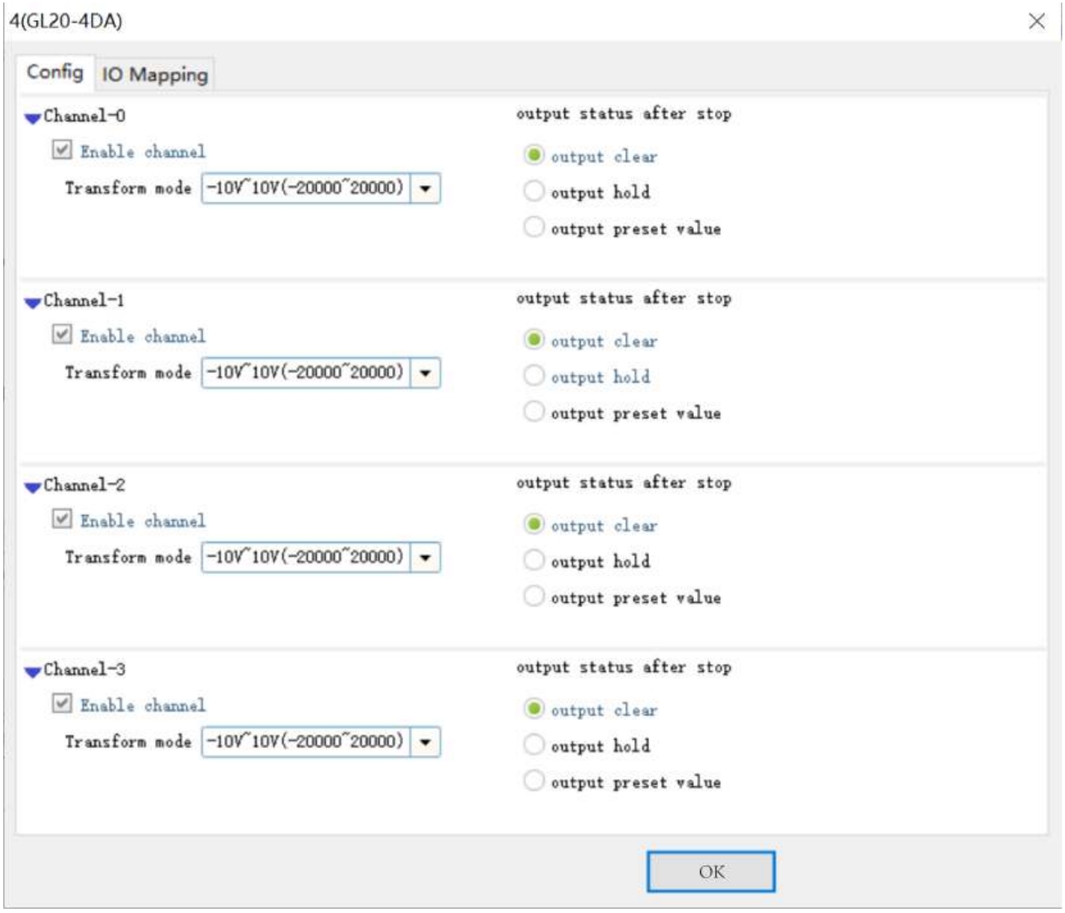
1. In the toolbox, select a module and double-click it. The module is automatically added to the corresponding configuration position.



Device Detailed List

Slot Number	Config Device Name	Device Description
2	GL20-0016ETN	Sixteen output access of DO module(NPN Transistor output)
3	GL20-4AD	Four access of AD module;supporting voltage and electric current intput

2. Access the parameter setting page of GL20-4DA.



4(GL20-4DA)

Config | IO Mapping

Channel-0
☑ Enable channel
Transform mode: -10V~10V(-20000~20000)
output status after stop: ● output clear ○ output hold ○ output preset value

Channel-1
☑ Enable channel
Transform mode: -10V~10V(-20000~20000)
output status after stop: ● output clear ○ output hold ○ output preset value

Channel-2
☑ Enable channel
Transform mode: -10V~10V(-20000~20000)
output status after stop: ● output clear ○ output hold ○ output preset value

Channel-3
☑ Enable channel
Transform mode: -10V~10V(-20000~20000)
output status after stop: ● output clear ○ output hold ○ output preset value

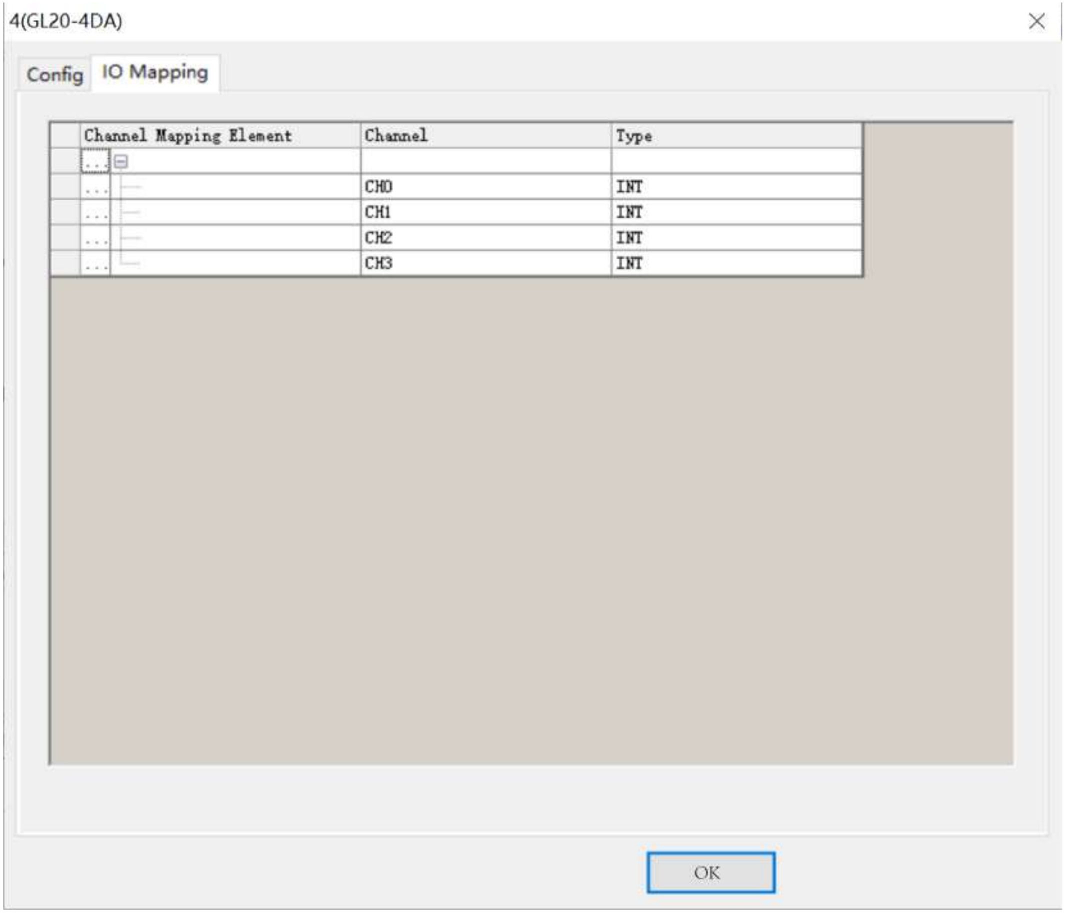
GL20-4DA provides four channels, and the parameters for the four channels are set consistently. The parameters are defined as follows:

① Determine whether to select "Enable channel". If not, deselect it to save the scanning time.

② Set "Translation Mode" to select the output type and range.

③ Set "Output state after Stopping" to "Output zero", "Output Holding", or "Output preset" for analog and digital values when the PLC is in the Stop state.

3. GL20-4DA also supports consecutive I/O mapping and separate I/O mapping. Click "..." next to the parent node for consecutive mapping or double-click the node of each channel for separate mapping.



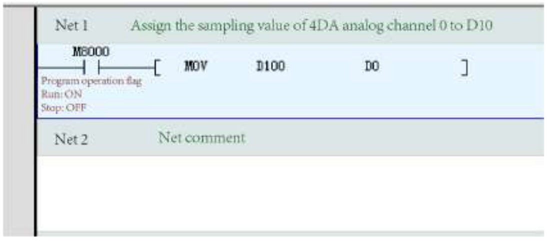
4(GL20-4DA)

Channel Mapping Element	Channel	Type
... ⊟		
...	CH0	INT
...	CH1	INT
...	CH2	INT
...	CH3	INT

4. On the "IO Mapping" tab page, map CH0 of the 4DA module to the D element D200. In an Easy model, you can also map it to a customized variable. The following table lists the relationships between the mapped variables and actual output analog values. The EtherCAT bus coupler supports conversion of three digital spans, including –20000 to +20000, –32000 to +32000, and –27648 to +27648, while the local bus only supports conversion of the span –20000 to +20000.

Output type	Rated Output Range	Rated Digital Value	Output Limit Range	Digital Value Limit
Analog voltage output	–10 V to +10 V	–20000 to +20000	–11 V to +11 V	–22000 to +22000
	0 V to 10 V	0 to 20000	–0.5 V to +10.5 V	–1000 to +21000
	–5 V to +5 V	–20000 to +20000	–5.5 V to +5.5 V	–22000 to +22000
	0 V to 5 V	0 to 20000	–0.25 V to +5.25 V	–1000 to +21000
	1 V to 5 V	0 to 20000	0.8 V to 5.2 V	–1000 to +21000
Analog current output	0 mA to 20 mA	0 to 20000	0 mA to 21 mA	0 to 21000
	4 mA to 20 mA	0 to 20000	3.2 mA to 20.8 mA	–1000 to +21000

5. Use the LD programming language to program DA sampling, and assign the voltage sampling value of CH0 from D200 to D10, which can be compiled and downloaded for running.

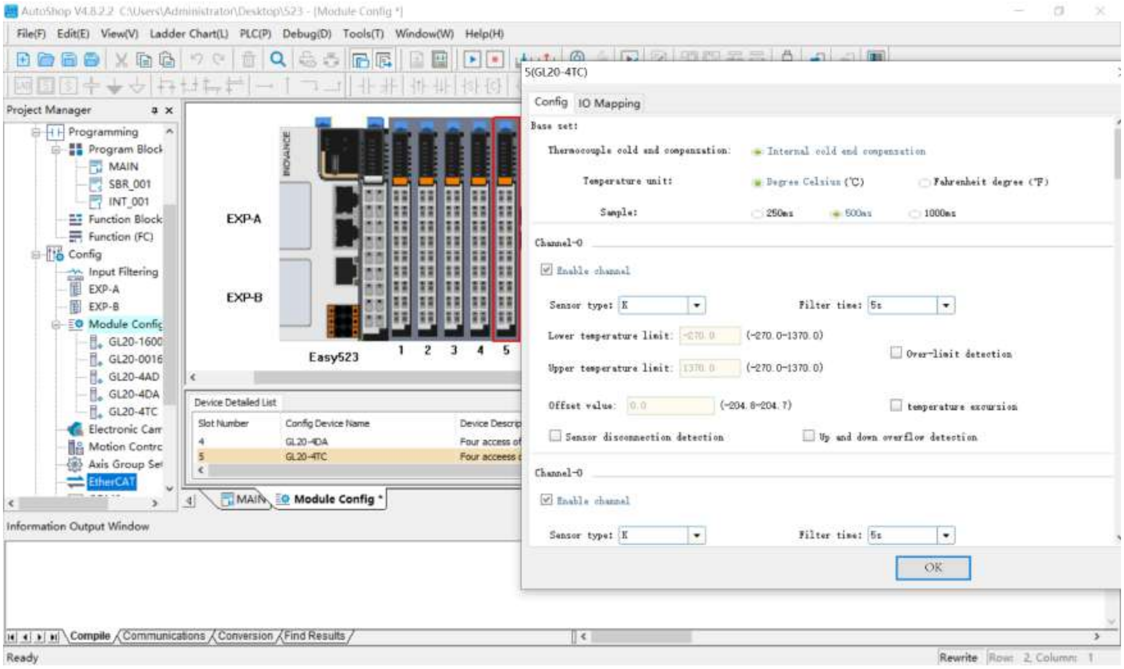


```
Net 1    Assign the sampling value of 4DA analog channel 0 to D10
  M8000
──┤ ├────[ MOV    D100    D0    ]
Program operation flag
Run: ON
Stop: OFF
Net 2    Net comment
```

5.2.2.3.6 Temperature Detection Modules

The method of using GL20-4PT 4-in RTD module and GL20-4TC 4-in TC TEMP MEAS module as local extension modules is as follows:

1. The add, delete, and parameter check operations on the page are similar to those of other local modules.
2. Each channel contains basic parameter settings. In the parameters, "Enable channel" indicates whether the current channel parameters are enabled; "Upper TEMP" and "Lower TEMP" can be set only when "Detect overrun" is selected; "Offset value" can be set only when "Offset TEMP" is selected.



Device Detailed List

Slot Number	Config Device Name	Device Descrip
4	GL20-4DA	Four access of
5	GL20-4TC	Four access

5(GL20-4TC)

Config | IO Mapping

Base set:

Thermocouple cold end compensation: ● Internal cold end compensation

Temperature unit: ● Degree Celsius (℃) ○ Fahrenheit degree (℉)

Sample: ○ 250ms ● 500ms ○ 1000ms

Channel-0

☑ Enable channel

Sensor type: K   Filter time: 5s

Lower temperature limit: -270.0 (-270.0~1370.0)

Upper temperature limit: 1370.0 (-270.0~1370.0)

□ Over-limit detection

Offset value: 0.0 (-204.8~204.7)

□ temperature excursion

□ Sensor disconnection detection   □ Up and down overflow detection

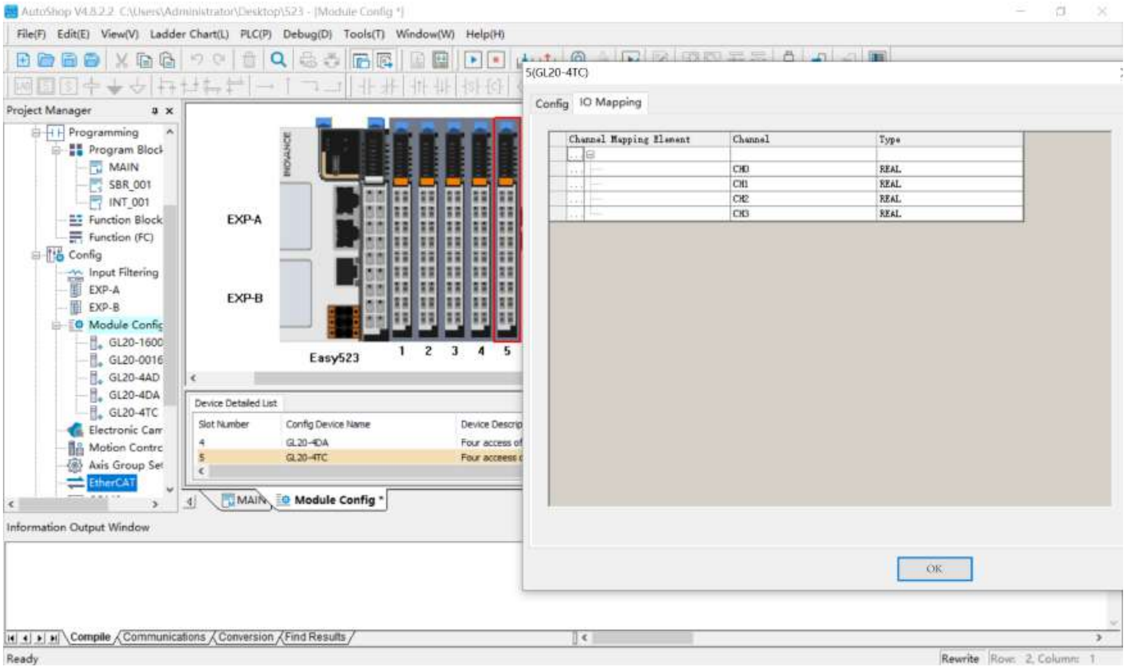
Channel-0

☑ Enable channel

Sensor type: K   Filter time: 5s

OK

3. GL20-4PT or GL20–4TC provides four channels, each of which can be used to map a soft element or REAL variable.

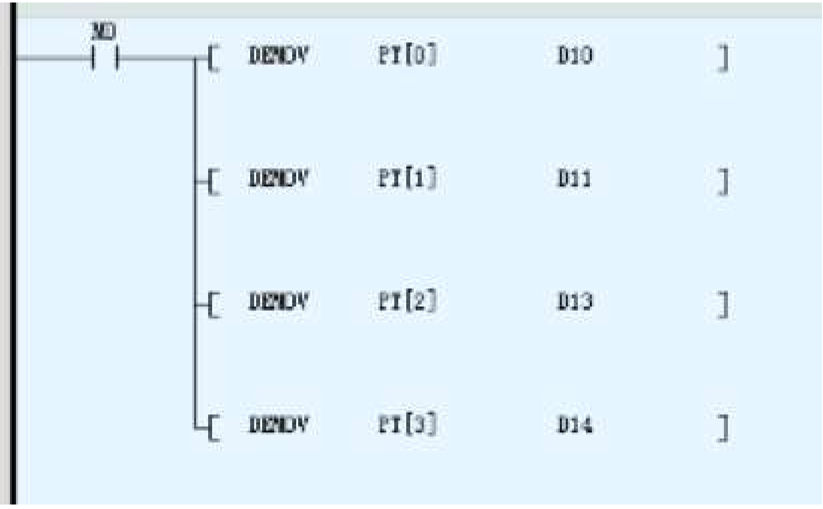


Device Detailed List

Slot Number	Config Device Name	Device Descrip
4	GL20-4DA	Four access of
5	GL20-4TC	Four acceess c

Channel Mapping Element	Channel	Type
	CH0	REAL
	CH1	REAL
	CH2	REAL
	CH3	REAL

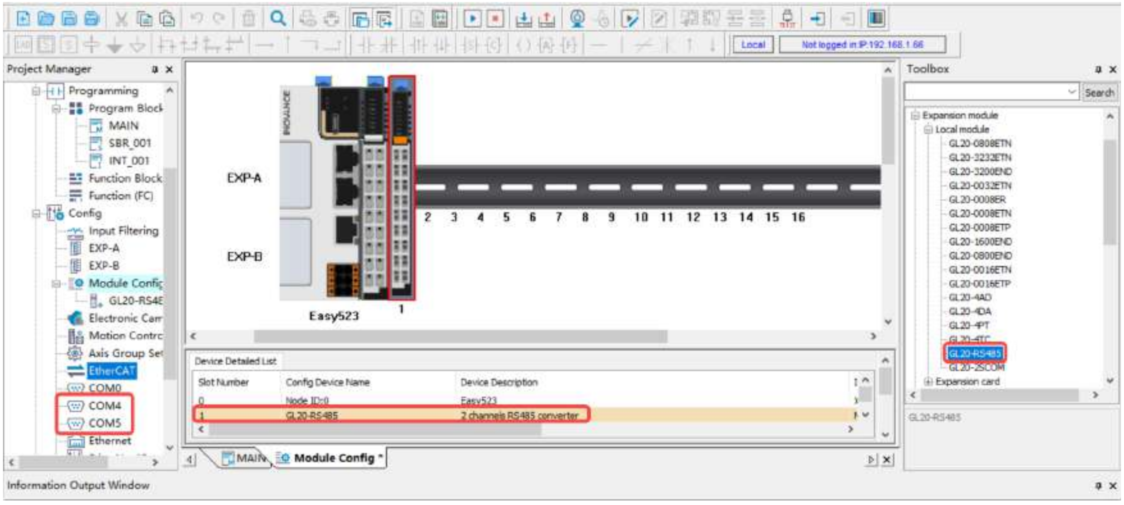
4. Use the LD programming language for sampling programming on the temperature of the GL20-4PT and GL20-4TC modules. Map CH0 to CH4 to REAL array PTs in sequence, use D10 to D14 to collect the temperature, and compile, download, and run the program, as shown in the following figure.



5.2.2.3.7 Communication Modules

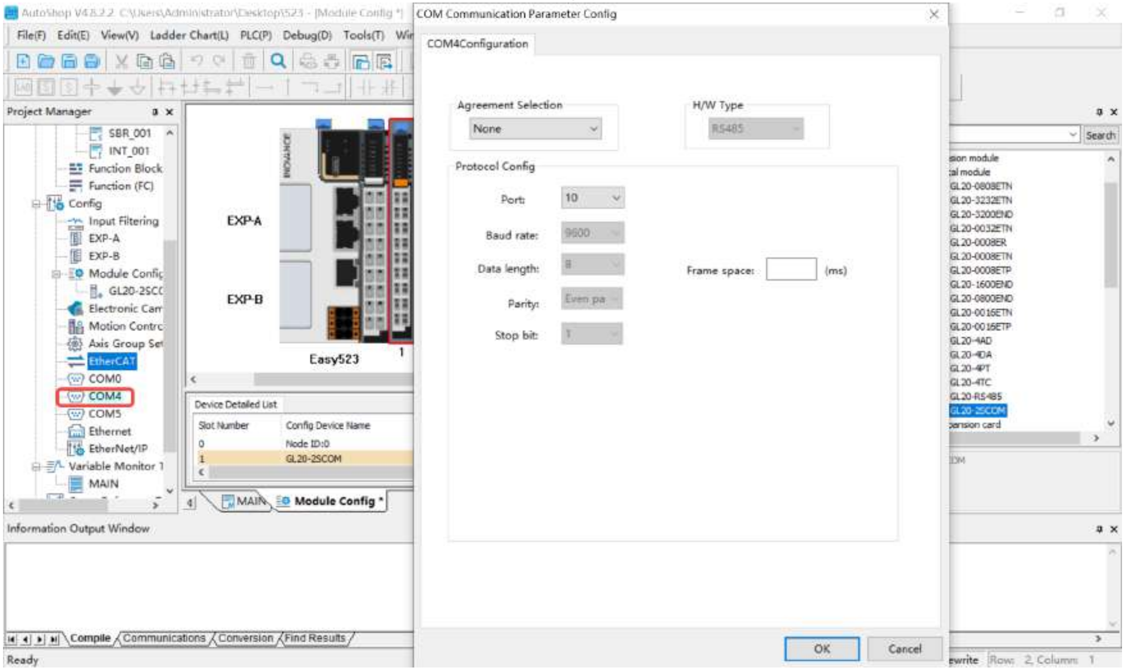
The method of using GL20-2SCOM 2-in RS232/RS485 module, GL20-2SCOM 1-in RS422 encoder module, and GL20-2S485 2 RS485 serial communication module as local extension modules is as follows:

1. The add, delete, and parameter check operations on the page are similar to those of other local modules.
2. Each module provides two channels, each of which contains basic serial port parameter settings and basic data transmission settings. In the parameters, "Enable channel" indicates whether the current channel parameters are enabled; Some parameters that are unavailable, such as "Serial port NO." and "Data bit", indicate that these parameters cannot be set on the page.



Slot Number	Config Device Name	Device Description
0	Node ID:0	Easy523
1	GL20-RS485	2 channels RS485 converter

3. After the GL20-2S485 and GL20-2SCOM modules are added, two COM ports are generated in the "Project Manager" tree, ranging from COM4 to COM15. As shown in the following figure, COM4 and COM5 are generated, corresponding to communication data of CH0 and CH1 of GL20-2S485. Double-click COM4 or COM5 to check COM communication parameters.



COM Communication Parameter Config

COM4Configuration

Agreement Selection: None

H/W Type: RS485

Protocol Config

Port: 10

Baud rate: 9600

Data length: 8

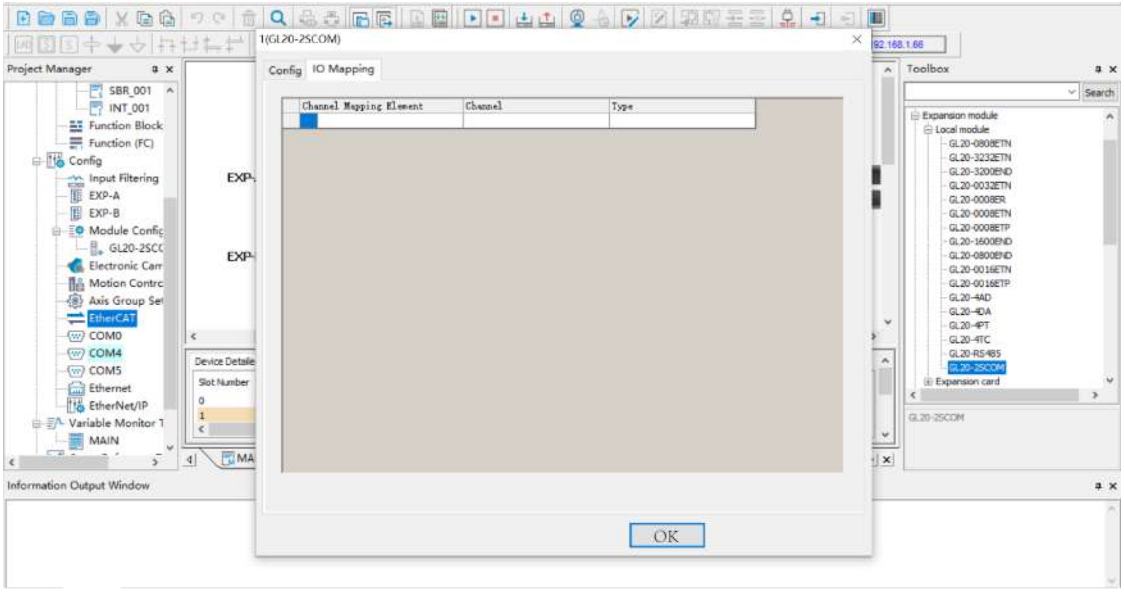
Parity: Even pa

Stop bit: 1

Frame space: (ms)

Slot Number	Config Device Name
0	Node ID:0
1	GL20-2SCOM

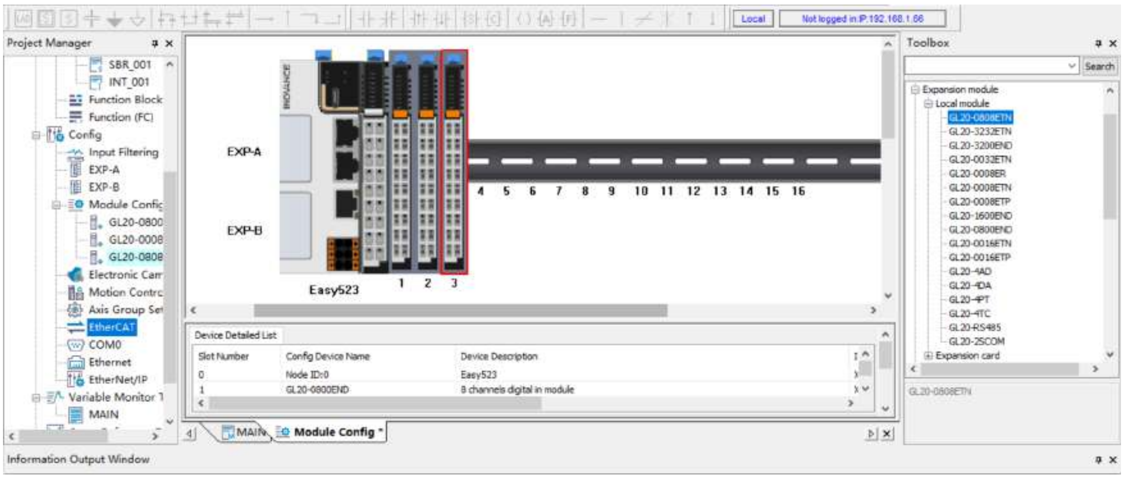
4. Select a COM port, change the COM port number, and then map corresponding data to the modified COM port. The modified parameters such as "Baud rate", "Data length", "Parity", and "Stop bit" will be synchronized to the module page shown in the preceding figure, to ensure data synchronization. After setting the corresponding COM port as the Modbus-RTU master station, right-click the port to add the Modbus configuration for Modbus communication in the same way as COM0.
   If a communication module has no I/O mapping data, the module page is empty.



Channel Mapping Element	Channel	Type

5.2.2.3.8 Application Examples

1. Connect the GL20-0800END, GL20-0008ETN, and GL20-0808ETN modules to the actual host as an example. In the "Project Manager" tree, right-click "Module Config" and select "Auto Scan" to start scanning. The scan result is displayed as shown in the following figure. The three modules can be manually added.
2. Click "Update Config" to add the mounted modules to the configuration. The default configuration is added for the modules by default.



Device Detailed List

Slot Number	Config Device Name	Device Description
0	Node ID:0	Easy523
1	GL20-0800END	8 channels digital in module

3. Download the configuration and run the modules. Nodes of the modules in the "Project Manager" tree turn green and "Ok" is displayed on the configuration page, indicating that the modules are running properly.

5.2.3 Extension Cards

5.2.3.1 Overview

An Easy series host can be equipped with up to two extension cards, and nine types of extension cards are supported. Extension card slot 1 and extension card slot 2 support different types of extension cards, as listed in the following table.

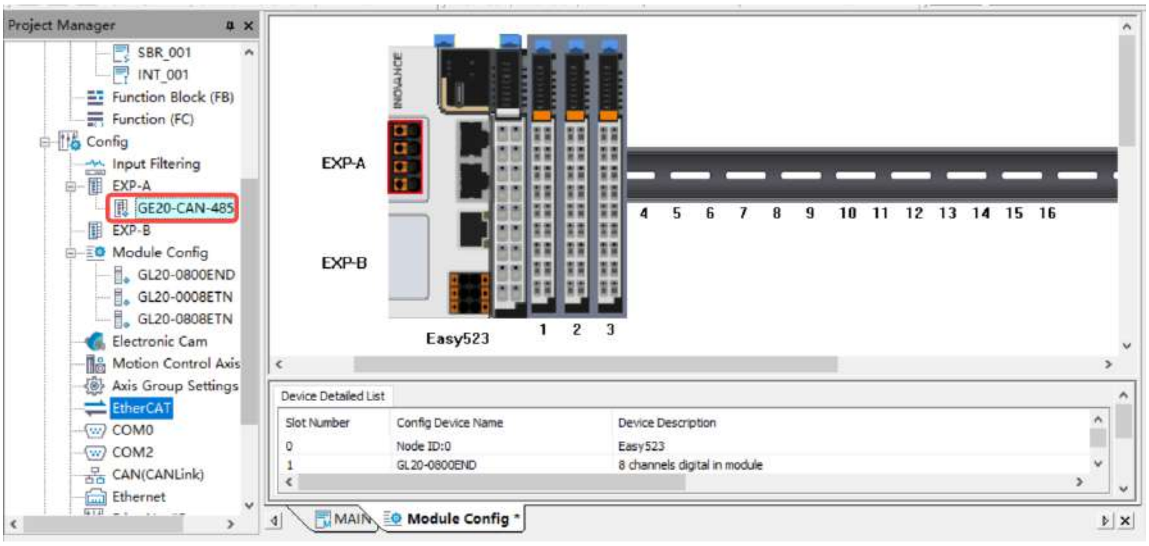
No.	Name (Model)	Function	Extension Card Slot 1	Extension Card Slot 2
1	GE20-232/485-RTC	Providing either the RS232 or RS485 extension interface RTC extension	Not supported	Supported
2	GE20-232/485	Providing either the RS232 or RS485 extension interface	Supported	Supported
3	GE20-CAN-485	Providing the CAN extension interface Providing the RS485 extension interface	Supported	Not supported
4	GE20-2AD1DA-I	Providing 2-in AD and 1-out DA extension (current)	Supported	Supported
5	GE20-2AD1DA-V	Providing 2-in AD and 1-out DA extension (voltage)	Supported	Supported
6	GE20-4DI	Providing 4-in I/O extension	Supported	Supported
7	GE20-4DO-TN	Providing 4-out I/O extension	Supported	Supported
8	GE20-RTC	Providing RTC extension	Not supported	Supported
9	GE20-TF	Providing the SD extension card interface	Not supported	Supported

5.2.3.2 Configuring Extension Cards

5.2.3.2.1 Adding Extension Cards

Extension cards can be added in two ways:

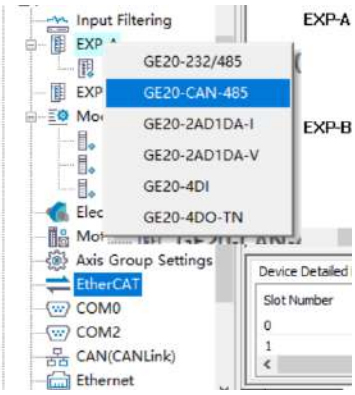
- Unfold the extension module or extension card node on the toolbox, and double-click to add corresponding extension cards.
- In the "Project Manager" tree, right-click EXP-A or EXP-B, and select the extension card type. On the configuration page that is displayed, add corresponding extension cards.



Device Detailed List

Slot Number	Config Device Name	Device Description
0	Node ID:0	Easy523
1	GL20-0800END	8 channels digital in module

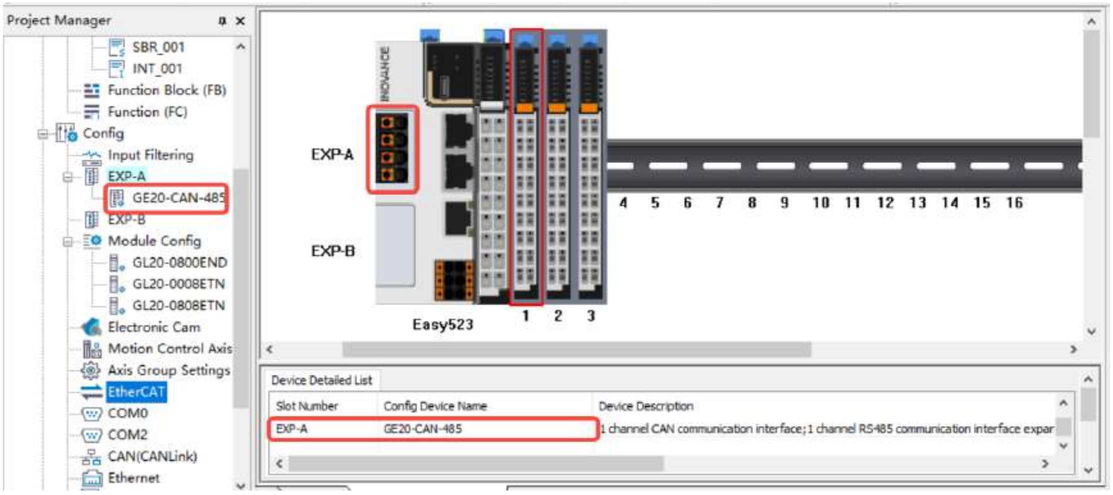
The following figure shows how to add an extension card using the shortcut menu.



5.2.3.2.2 Configuring Extension Card Parameters

You can access the extension card configuration page in the following ways:

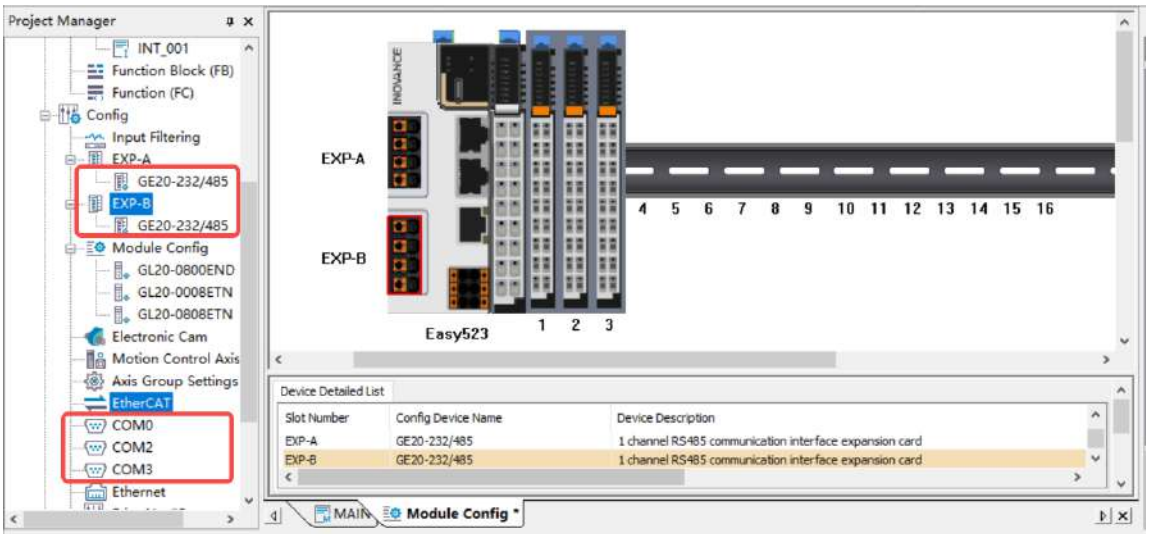
- Double-click a node under EXP-A or EXP-B.
- Double-click the EXP-A or EXP-B configuration on the configuration page.
- Double-click an item in "Device Detailed List", as shown in the following figure.



Device Detailed List

Slot Number	Config Device Name	Device Description
EXP-A	GE20-CAN-485	1 channel CAN communication interface;1 channel RS485 communication interface expar

Configurable COM nodes will be generated for the cards with COM ports, such as GE20-CAN-485, GE20-232/485-RTC, and GE20-232/485. EXP-A corresponds to COM2 and EXP-B to COM3. If GE20-CAN-485 is added for EXP-A, the CAN and COM2 nodes will be generated. If GE20-232/485 is added for EXP-B, the COM3 node will be generated. You can double-click a node for parameter configuration, as shown in the following figure.

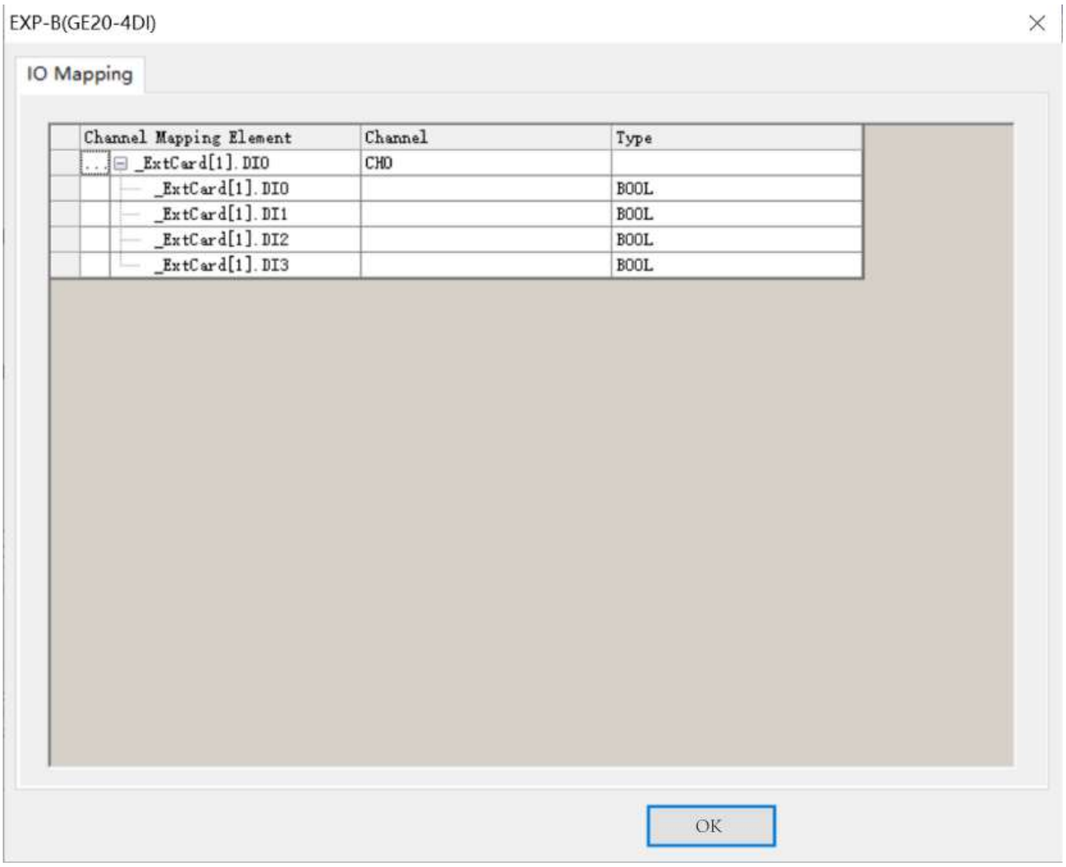


Device Detailed List

Slot Number	Config Device Name	Device Description
EXP-A	GE20-232/485	1 channel RS485 communication interface expansion card
EXP-B	GE20-232/485	1 channel RS485 communication interface expansion card

GE20-2AD1DA-I and GE20-2AD1DA-V are current-voltage hybrid extension cards. If they are added for EXP-A, the COM2 node will be generated by default. If they are added for EXP-B, the COM3 node and three Modbus instructions will be generated by default. You cannot exit COM port parameters and Modbus parameters.

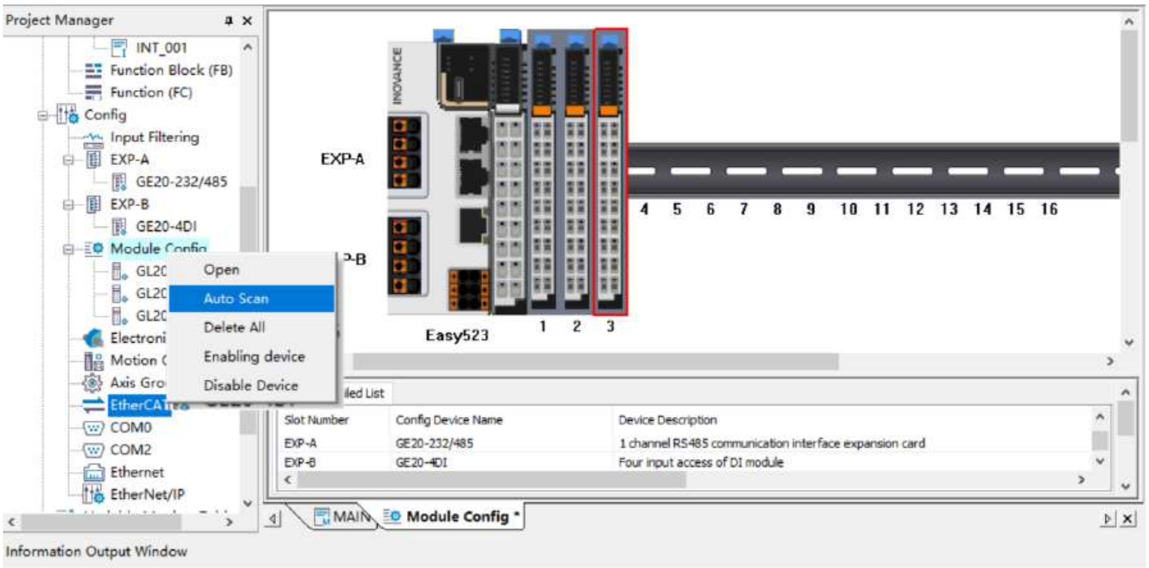
GE20-4DI and GE20-4DO extension cards are used in the same way as DI and DO modules. The extension cards provide mappings to system variables by default, which can be automatically modified.



Channel Mapping Element	Channel	Type
_ExtCard[1].DI0	CH0	
_ExtCard[1].DI0		BOOL
_ExtCard[1].DI1		BOOL
_ExtCard[1].DI2		BOOL
_ExtCard[1].DI3		BOOL

5.2.4 Application Examples

After installing the host and module, right-click "Module Config" in the "Project Manager" tree of the software, select "Auto Scan", and use the configuration scanning function to scan modules and extension cards, as shown in the following figure.



Slot Number	Config Device Name	Device Description
EXP-A	GE20-232/485	1 channel RS485 communication interface expansion card
EXP-B	GE20-4DI	Four input access of DI module

On the "Auto Scan" page, click "Update Config" to add the scanned configuration to default configurations, compile and download it, and check the operation status.

5.3 GL20-RTU-ECT Local Extension Module

5.3.1 Overview

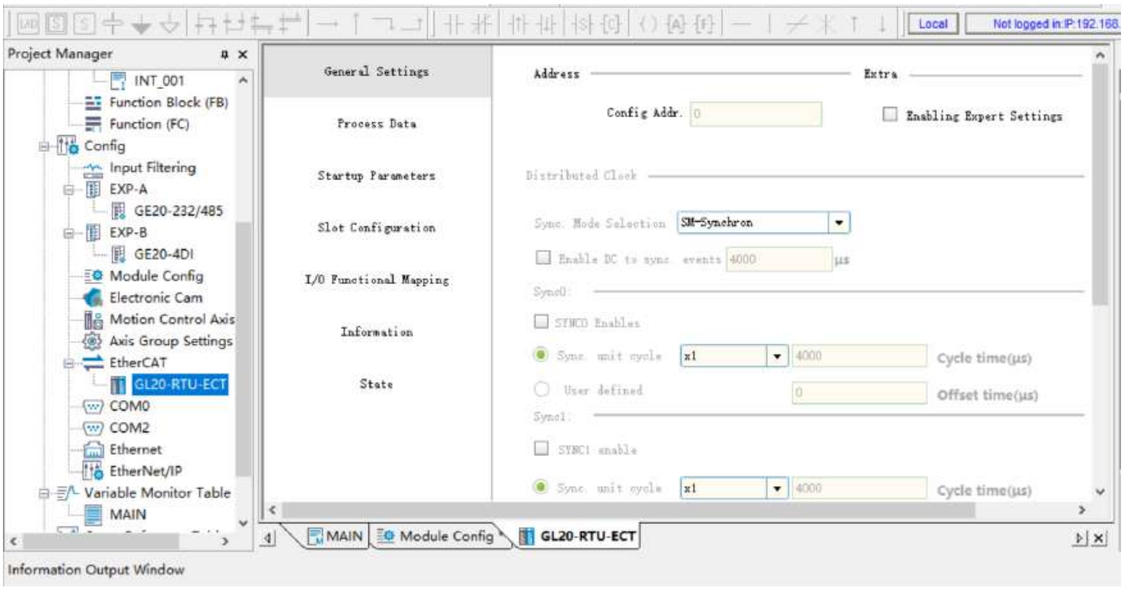
The GL20-RTU-ECT communication port module also supports the following six models.

Model	Description
GL20-0808ETN	8 DI module and 8 DO module
GL20-0008ETN	8 DO module
GL20-0008ETP	8 DO module
GL20-0008ER	8 DO module
GL20-0800END	8 DI module
GL20-4PT	4-in RTD module

5.3.2 Configuring Extension Modules

The GL20-RTU-ECT coupler is mainly mounted by using the EtherCAT bus to add corresponding modules. The procedure is as follows:

1. In the toolbox, choose "EtherCat Devices" > "Inovance Devices" > "IO coupler" and then double-click "GL20-RTU-ECT_1.2.8.0". The "GL20-RTU-ECT" slave station node is generated under "EtherCat" in the "Project Manager" tree. Double-click it to display the configuration page.



Address

Config Addr. 0

Extra

□ Enabling Expert Settings

Distributed Clock

Sync. Mode Selection SM-Synchron

□ Enable DC to sync. events 4000 µs

Sync0:

□ SYNC0 Enables

● Sync. unit cycle x1 4000 Cycle time(µs)

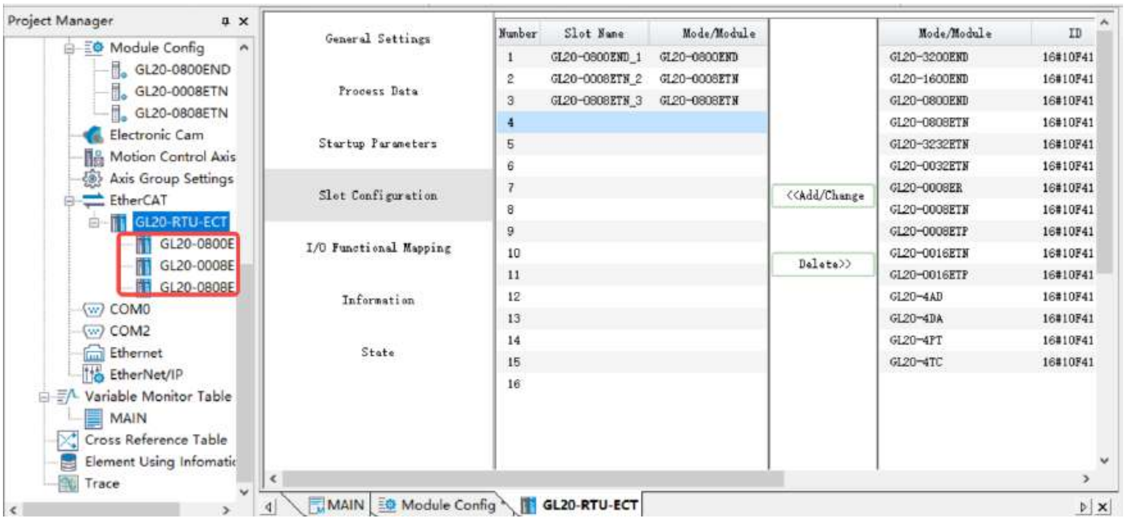
○ User defined 0 Offset time(µs)

Sync1:

□ SYNC1 enable

● Sync. unit cycle x1 4000 Cycle time(µs)

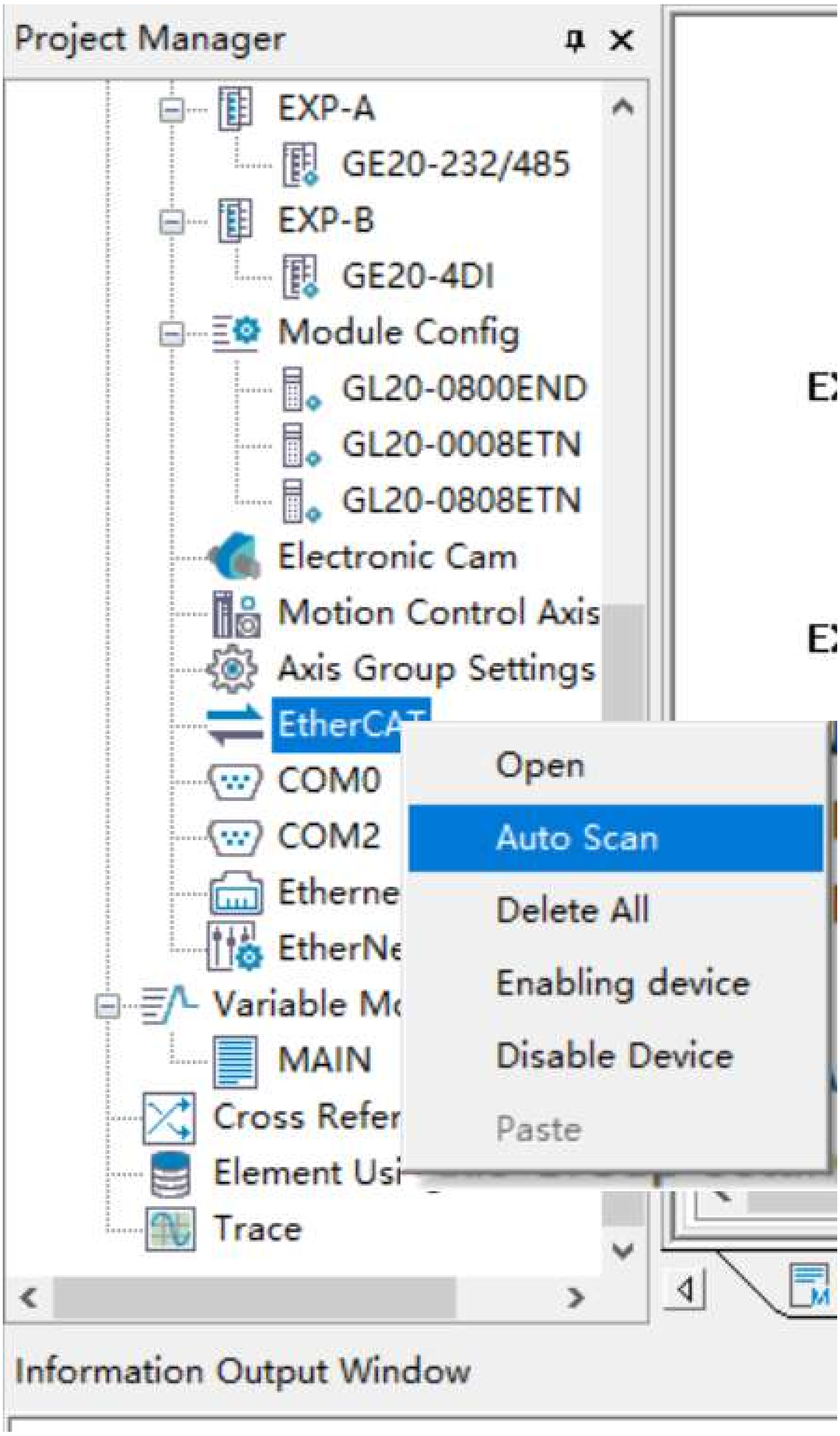
2. Click "Slot configuration" to access the slot configuration page. Select modules on the right, and click "Add/Change" or "Delete". Use the mounted modules GL20-0800END, GL20-0008ETN, and GL20-0808ETN as an example. Select the modules on the right and click "Add/Change" to add them to the left part.



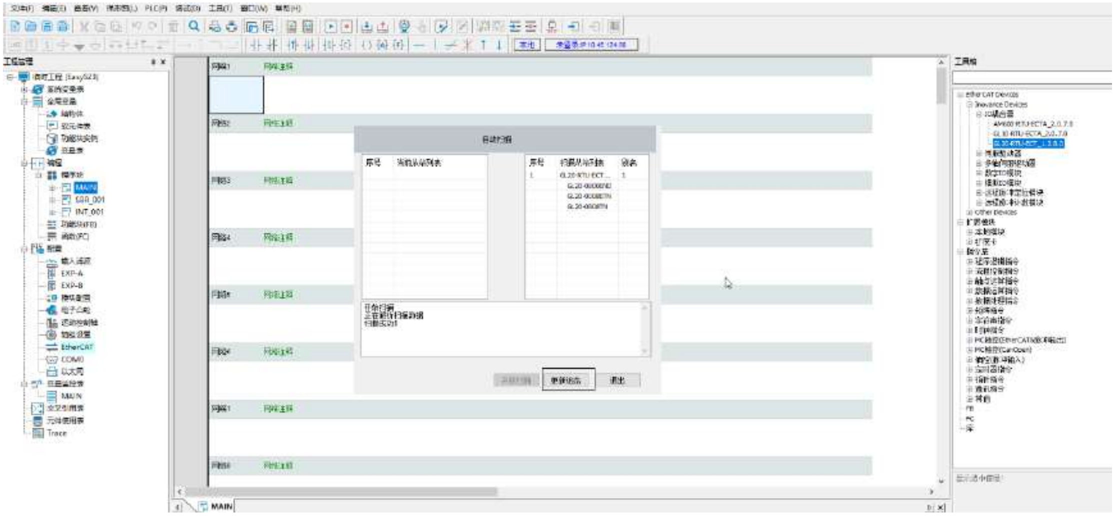
Number	Slot Name	Mode/Module
1	GL20-0800END_1	GL20-0800END
2	GL20-0008ETN_2	GL20-0008ETN
3	GL20-0808ETN_3	GL20-0808ETN
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		

Mode/Module	ID
GL20-3200END	16#10F41
GL20-1600END	16#10F41
GL20-0800END	16#10F41
GL20-0808ETN	16#10F41
GL20-3232ETN	16#10F41
GL20-0032ETN	16#10F41
GL20-0008ER	16#10F41
GL20-0008ETN	16#10F41
GL20-0008ETP	16#10F41
GL20-0016ETN	16#10F41
GL20-0016ETP	16#10F41
GL20-4AD	16#10F41
GL20-4DA	16#10F41
GL20-4PT	16#10F41
GL20-4TC	16#10F41

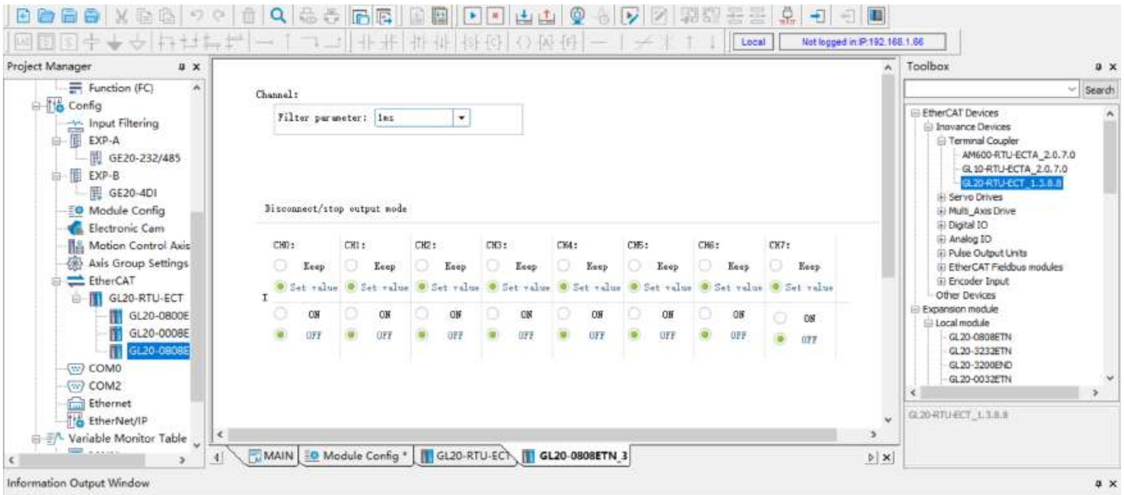
Alternatively, you can wire the modules, power on the PLC, right-click EtherCAT, and select "Auto Scan".



3. Click "Start Scan".



4. Click "Update Config" and select whether to retain current configurations. If so, configurations of slave stations scanned for one or more times will be added next to current configurations. If not, current configurations will be deleted and then scanned default configurations need to be added, with the same result as manual adding.
5. As shown in the following figure, three nodes are generated under GL20-RTU-ECT, indicating three modules. Double-click each node to access the corresponding module configuration page, and configure module parameters.



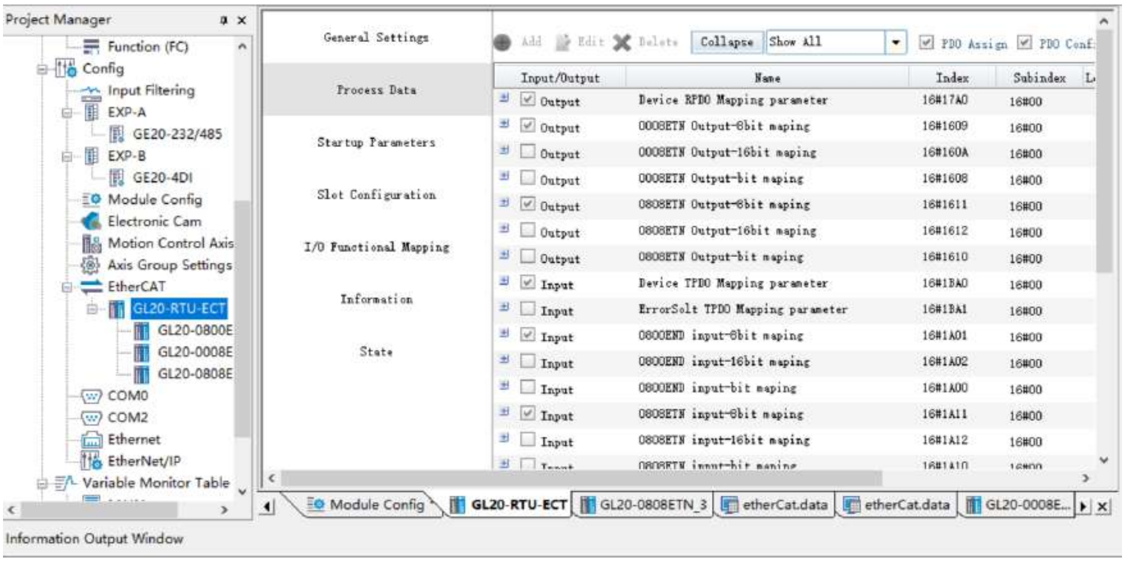
Channel:

Filter parameter: 1ms

Disconnect/stop output mode

CH0:	CH1:	CH2:	CH3:	CH4:	CH5:	CH6:	CH7:
○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep	○ Keep
● Set value	● Set value	● Set value	● Set value	● Set value	● Set value	● Set value	● Set value
○ ON	○ ON	○ ON	○ ON	○ ON	○ ON	○ ON	○ ON
● OFF	● OFF	● OFF	● OFF	● OFF	● OFF	● OFF	● OFF

6. Double-click GL20-RTU_ECT and switch to the "Process data" page and the "Start parameter" page to check the generated process data and startup parameters respectively.



Input/Output	Name	Index	Subindex
☑ Output	Device RPDO Mapping parameter	16#17A0	16#00
☑ Output	0008ETN Output-8bit maping	16#1609	16#00
□ Output	0008ETN Output-16bit maping	16#160A	16#00
□ Output	0008ETN Output-bit maping	16#1608	16#00
☑ Output	0808ETN Output-8bit maping	16#1611	16#00
□ Output	0808ETN Output-16bit maping	16#1612	16#00
□ Output	0808ETN Output-bit maping	16#1610	16#00
☑ Input	Device TPDO Mapping parameter	16#1BA0	16#00
□ Input	ErrorSolt TPDO Mapping parameter	16#1BA1	16#00
☑ Input	0800END input-8bit maping	16#1A01	16#00
□ Input	0800END input-16bit maping	16#1A02	16#00
□ Input	0800END input-bit maping	16#1A00	16#00
☑ Input	0808ETN input-8bit maping	16#1A11	16#00
□ Input	0808ETN input-16bit maping	16#1A12	16#00
□ Input	0808ETN input-bit maping	16#1A10	16#00

7. Download the generated data and run the PLC. Then, the EtherCAT and GL20-RTC-ECT icons turn green, and no PLC error is reported, indicating that the PLC is running properly.

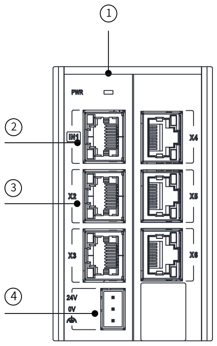
5.4 GR10-EC-6SW Branch Module

5.4.1 Overview

Caution

The GR10-EC-6SW branch module only supports use with H5U devices.

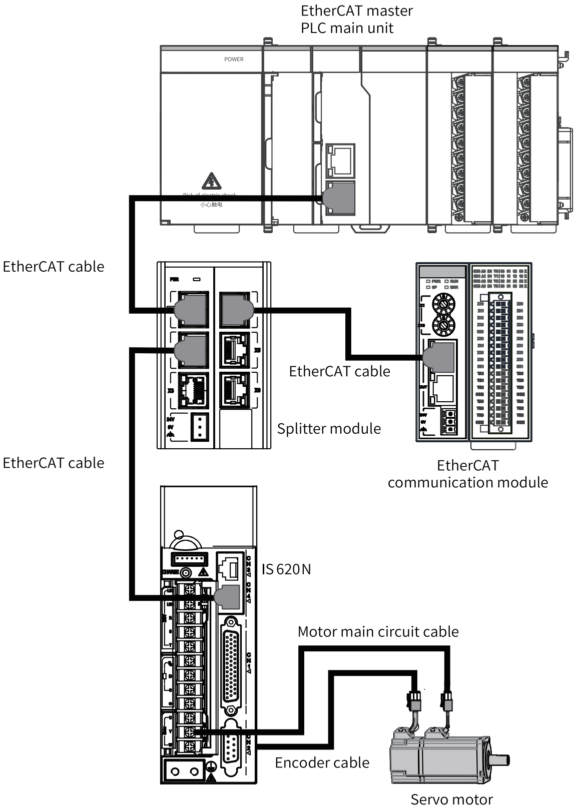
The GR10-EC-6SW branch module is used to extend Extension ports, as shown in the following figure.



No.	Terminal Name	Definition		
①	Power indicator	PWR	Green	Turned on upon power-on
②	EtherCAT input port	IN	Port1 and EtherCAT input port, connecting to the front EtherCAT master station	

No.	Terminal Name	Definition	
③	EtherCAT output port	X2	Port2 and EtherCAT output port, connecting to the rear EtherCAT slave station
		X3	Port3 and EtherCAT output port, connecting to the rear EtherCAT slave station
		X4	Port4 and EtherCAT output port, connecting to the rear EtherCAT slave station
		X5	Port5 and EtherCAT output port, connecting to the rear EtherCAT slave station
		X6	Port6 and EtherCAT output port, connecting to the rear EtherCAT slave station
④	24 V power input terminal	Power input for a module	

The GR10-EC-6SW branch module can be connected to multiple EtherCAT slave devices. System wiring is shown in the following figure.



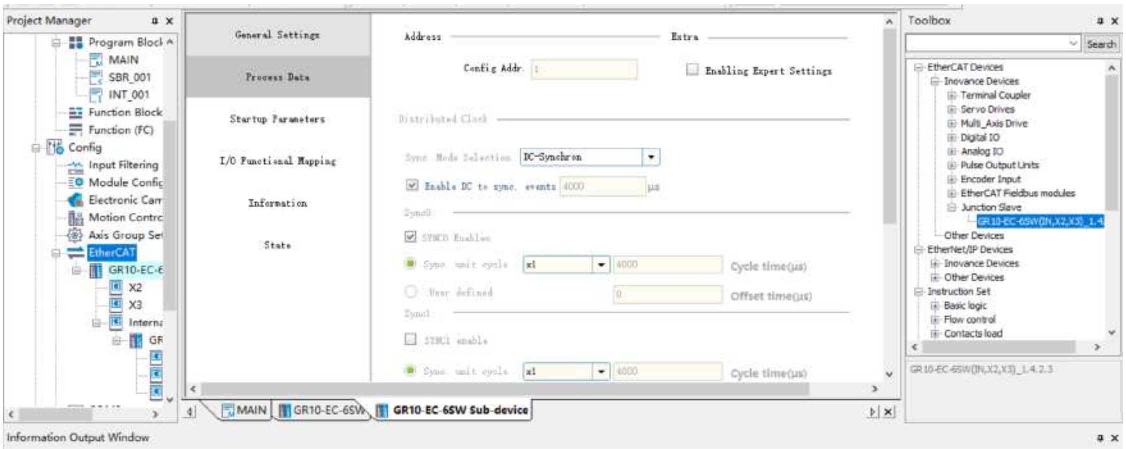
For more information about the GR10-EC-6SW branch module, see the *GR10-EC-6SW 6-EtherCAT Branch Module User Guide*.

5.4.2 Adding the Branch Module and Its Slave

Branch modules and slave devices can be added manually or by automatic scanning.

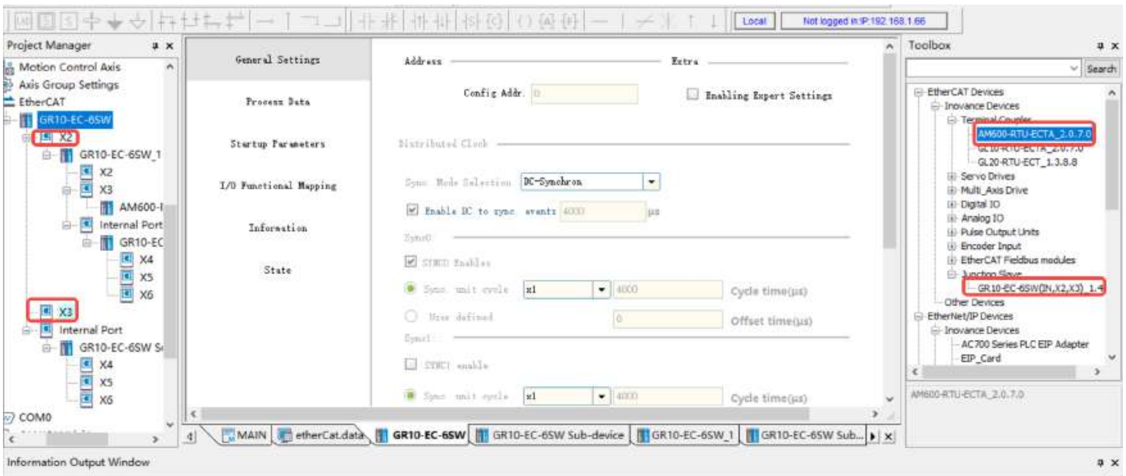
Manual adding

1. In the toolbox, choose "EtherCAT Devices" > "Inovance Devices" > "Branch module", and double-click GR10-EC-6SW.



2. Select the EtherCAT output port with the same physical configuration and add a branch module or slave device.

   For example, to use the X2 EtherCAT output port to add a branch module, select "X2" and double-click "GR10-EC-6SW" in the toolbox; to use the X3 EtherCAT output port to add the slave device AM600-RTU-ECTA, select "X3", choose "Inovance Devices" > "IO coupler" in the toolbox, and double-click "AM600-RTU-ECTA".



Note

- Up to three levels of branch modules can be added.
- Multiple slaves or branch modules can be added under the EtherCAT output port of a branch module.
- If Slave-Disable is turned on for the selected node, all branch nodes under the slave are disabled. Branch nodes also support Slave-Disable. If an EtherCAT output port is disabled, all slaves under the port are disabled.
- Slaves of branch modules do not support copy and paste.

3. For details about the EtherCAT slave station configuration, see section 9.3 "Slave Configuration" in chapter 9 "EtherCAT Communication."

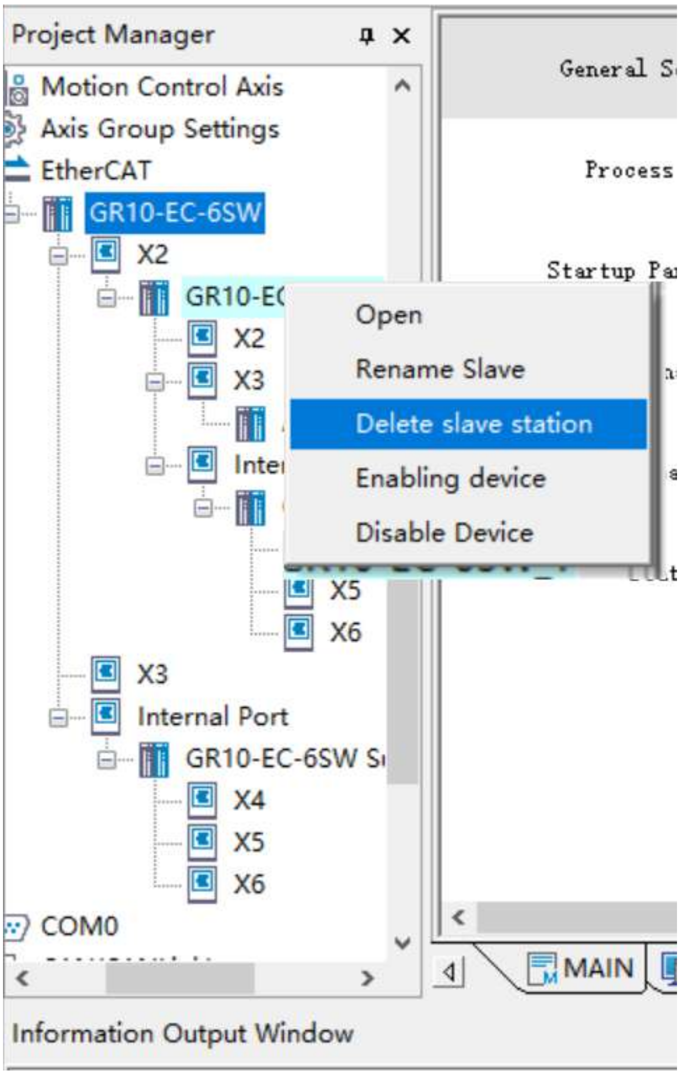
Automatic scanning

1. In the "Project Manager" tree, right-click "EtherCAT" and select "Auto Scan". Click "Update Config", select not to retain current configurations, and add the GR10-EC-6SW branch module and its slave device.

2. For details about the EtherCAT slave station configuration, see section 9.3 "Slave Configuration" in chapter 9 "EtherCAT Communication."

5.4.3 Deleting the Branch Module and Its Slave

In the "Project Manager" tree on the left, right-click the branch module or slave device under "EtherCAT", and select "Delete slave station" to delete the branch module or slave device. After the slave station is deleted, data of the slave station and all its subnodes is deleted.



Caution

The branch modules as the X2 to X6 nodes, Internal Port, and GR10-EC-6SW Sub-device node must not be deleted.

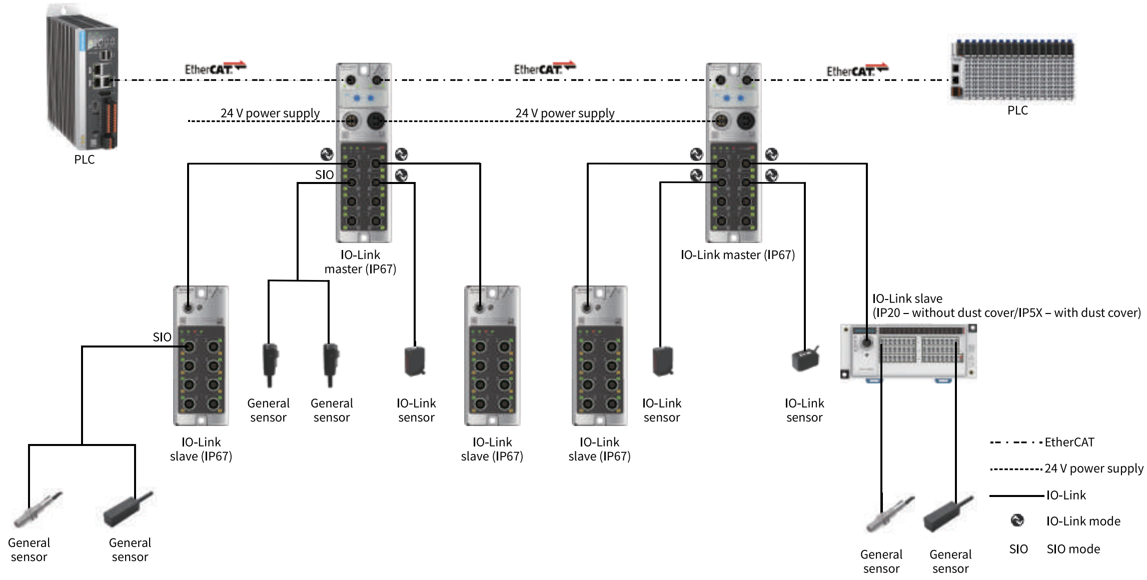
5.5 GS20-ECT-8L Module

5.5.1 Overview

The GS20-ECT-8L module is connected to the sensor or activator as an EtherCAT slave station using the IO-Link protocol.

IO-Link is a serial digital communication protocol used to integrate a smart sensor and an activator into an automation system. According to IEC 61131-9, it is the first worldwide standard I/O technology for periodic data exchange between sensors/activators and PLCs. IO-Link is an open point-to-point communication protocol but not a fieldbus protocol. The protocol features ease of use, stabilization, reliability, and plug and play and is used more and more widely with the development of industry 4.0.

The IO-Link topology consists of PLCs, IO-Link master stations (GS20-ECT-8L), IO-Link slave stations (GR20-16EMNL, GR20-16EMPL, GS20-16EMNL, or GS20-16EMPL), sensor, activator, and IO-Link cable, as shown in the following figure. IO-Link master stations are connected to each other and PLCs through EtherCAT communication. IO-Link master stations are connected to IO-Link slave stations and sensors/activators and IO-Link slave stations are connected to sensors/activators through IO-Link communication. IO-Link master stations and slave stations can supply power for sensors/activators directly.



Note

The IO-Link ports support two working modes: IO-Link mode and Standard I/O (SIO) mode. The working mode can be set separately on any port.

5.5.2 Configuring the GS20-ECT-8L Module

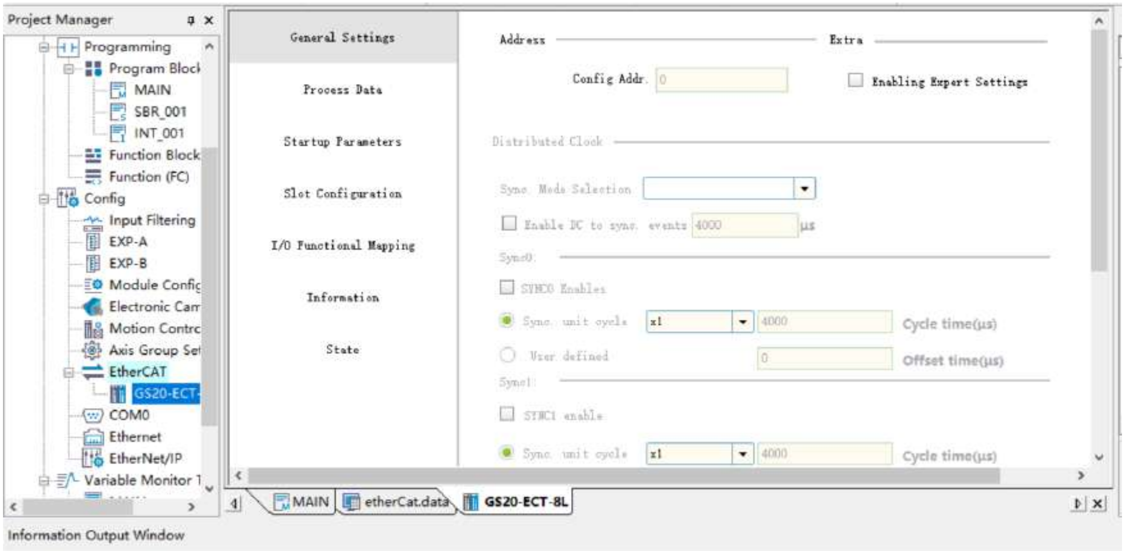
Prerequisites: The PLC is connected to the GS20-ECT-8L module through cables and is powered on.

1. Add the GS20-ECT-8L module.

In the toolbox, choose "EtherCAT Devices" > "Inovance Devices" > "EtherCAT Fieldbus modules", and double-click "GS20-ECT-8L" to add the GS20-ECT-8L module. Alternatively, in the "Project Manager" tree, right-click "EtherCAT" and select "Auto Scan". After successful scanning, click "Update Config" to update the device configuration and add the GS20-ECT-8L module.

Note

If you choose to discard the current configuration, delete all slave configurations first, and then add the scanned devices. If you choose to retain the current configuration, add the scanned slaves one by one after the existing slaves.



Address

Config Addr. 0

Extra

□ Enabling Expert Settings

Distributed Clock

Sync. Mode Selection

□ Enable DC to sync. events 4000 µs

Sync0:

□ SYNC0 Enables

● Sync. unit cycle x1 4000 Cycle time(µs)

○ User defined 0 Offset time(µs)

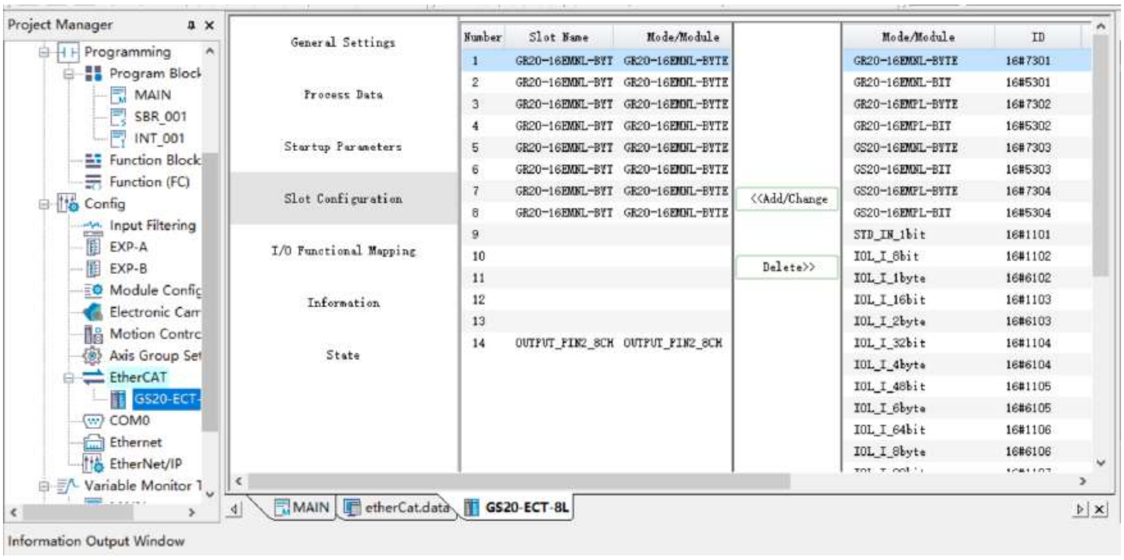
Sync1:

□ SYNC1 enable

● Sync. unit cycle x1 4000 Cycle time(µs)

2. Configure slots.

Click "Slot configuration". On the page that is displayed, select corresponding slots and click "Delete" to delete devices from the slots. When the slot area is empty, select target devices on the right and click "Add/Change" to add the selected devices to the slots. The following table lists the slot numbers, names, and description.



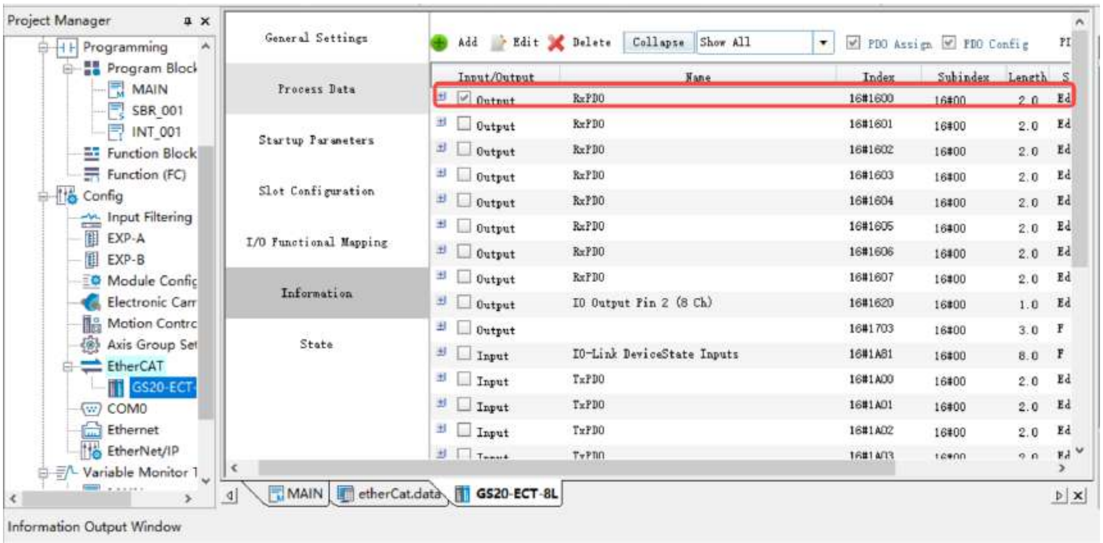
Number	Slot Name	Mode/Module
1	GR20-16EMNL-BYT	GR20-16EMNL-BYTE
2	GR20-16EMNL-BYT	GR20-16EMNL-BYTE
3	GR20-16EMNL-BYT	GR20-16EMNL-BYTE
4	GR20-16EMNL-BYT	GR20-16EMNL-BYTE
5	GR20-16EMNL-BYT	GR20-16EMNL-BYTE
6	GR20-16EMNL-BYT	GR20-16EMNL-BYTE
7	GR20-16EMNL-BYT	GR20-16EMNL-BYTE
8	GR20-16EMNL-BYT	GR20-16EMNL-BYTE
9		
10		
11		
12		
13		
14	OUTPUT_PIN2_8CH	OUTPUT_PIN2_8CH

Mode/Module	ID
GR20-16EMNL-BYTE	16#7301
GR20-16EMNL-BIT	16#5301
GR20-16EMPL-BYTE	16#7302
GR20-16EMPL-BIT	16#5302
GS20-16EMNL-BYTE	16#7303
GS20-16EMNL-BIT	16#5303
GS20-16EMPL-BYTE	16#7304
GS20-16EMPL-BIT	16#5304
STD_IN_1bit	16#1101
IOL_I_8bit	16#1102
IOL_I_1byte	16#6102
IOL_I_16bit	16#1103
IOL_I_2byte	16#6103
IOL_I_32bit	16#1104
IOL_I_4byte	16#6104
IOL_I_48bit	16#1105
IOL_I_6byte	16#6105
IOL_I_64bit	16#1106
IOL_I_8byte	16#6106

No.	Slot Name	Description
1 to 8 (corresponding to IO-Link physical ports 0 to 7 of the GS20-ECT-8L module)	GR20-16EMNL-BYTE, GR20-16EMNL-BIT, GR20-16EMPL-BYTE, GR20-16EMPL-BIT, GS20-16EMNL-BYTE, GS20-16EMNL-BIT, GS20-16EMPL-BYTE, GS20-16EMPL-BIT	Names of Inovance IO-Link slave station modules, where N indicates active low and P indicates active high
	STD_IN_1bit	Standard input
	STD_OUT_1bit	Standard output
	IOL_I_*X*byte/IOL_I_*X*bit	IO_Link input, such as IOL_I_1byte, indicating 1-byte IO-Link input
	IOL_O_*X*byte/IOL_O_*X*bit	IO_Link output, such as IOL_O_1byte, indicating 1-byte IO-Link output
	IOL_I/O_*X*/_*y*byte/IOL_I/O_*X*/_*y*bit	IO_Link I/O, such as IOL_I/O_1/_1byte, indicating 1-byte IO-Link input and 1-byte IO-Link output
9	INPUT_PIN2_8CH	PIN2 input
10	ACTOR_SHORTCIRCUIT_PIN2_8CH	PIN2 short circuit monitoring
11	ACTOR_SHORTCIRCUIT_PIN4_8CH	PIN4 short circuit monitoring
12	SENSOR_SUPPLY_SHORTCIRCUIT_8CH	PIN1 short circuit monitoring
13	SYSTEM_HARDWARE_MONITOR	Hardware status monitoring
14	OUTPUT_PIN2_8CH	PIN2 output

3. Configure process data and I/O function mapping.

   a. Click "Process data" and select corresponding I/O PDOs as required. The corresponding I/O mapping data is generated on the "I/O function mapping" page. The PDO index 16#1600 is output as an example.

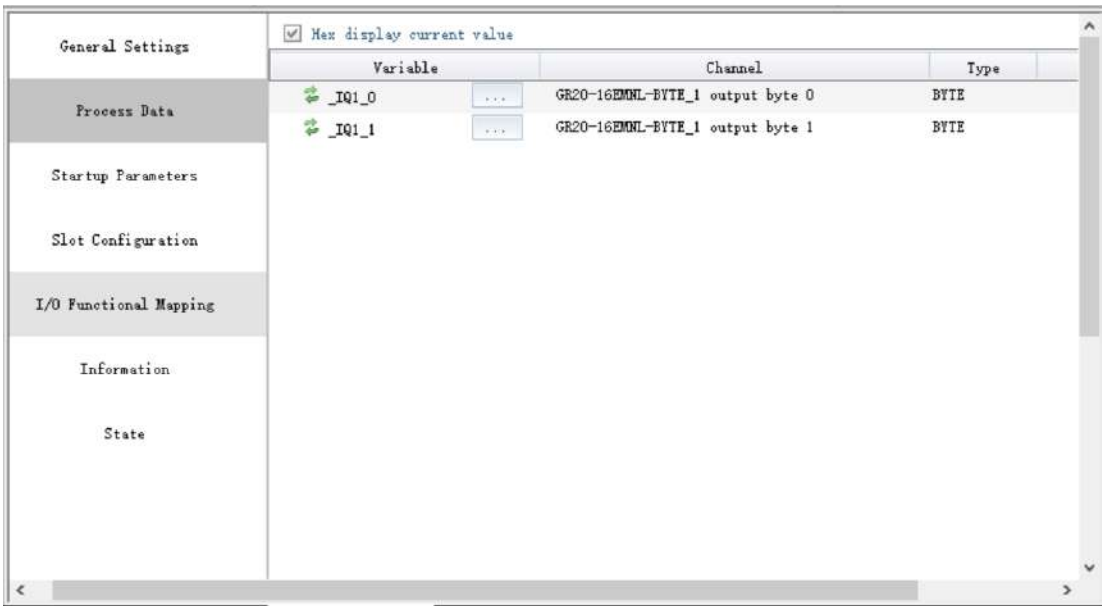


	Input/Output	Name	Index	Subindex	Length	S
☑	Output	RxPDO	16#1600	16#00	2.0	Ed
□	Output	RxPDO	16#1601	16#00	2.0	Ed
□	Output	RxPDO	16#1602	16#00	2.0	Ed
□	Output	RxPDO	16#1603	16#00	2.0	Ed
□	Output	RxPDO	16#1604	16#00	2.0	Ed
□	Output	RxPDO	16#1605	16#00	2.0	Ed
□	Output	RxPDO	16#1606	16#00	2.0	Ed
□	Output	RxPDO	16#1607	16#00	2.0	Ed
□	Output	IO Output Pin 2 (8 Ch)	16#1620	16#00	1.0	Ed
□	Output		16#1703	16#00	3.0	F
□	Input	IO-Link DeviceState Inputs	16#1A81	16#00	8.0	F
□	Input	TxPDO	16#1A00	16#00	2.0	Ed
□	Input	TxPDO	16#1A01	16#00	2.0	Ed
□	Input	TxPDO	16#1A02	16#00	2.0	Ed
□	Input	TxPDO	16#1A03	16#00	2.0	Ed

Note

- The background DINT is of the signed 32-bit type, so an error occurs when UDINT data is read. If there is no signed number with more than 32 bits, test whether the value read is correct. If decimal display is incomplete, use hexadecimal display. In hexadecimal display, unsigned 32-bit numbers are displayed. Device IDs and other UDINT type data can be viewed in hexadecimal display.
- The BITARR8 type of PDO data is mapped to the BYTE type by default, and can be operated bitwise according to the BYTE type, or mapped to a BOOL array for reading and writing.

Then, channel mappings for "GR20-16EMNL-BYTE_1 output byte 0" and "GR20-16EMNL-BYTE_1 output byte 1" are generated by default.



☑ Hex display current value

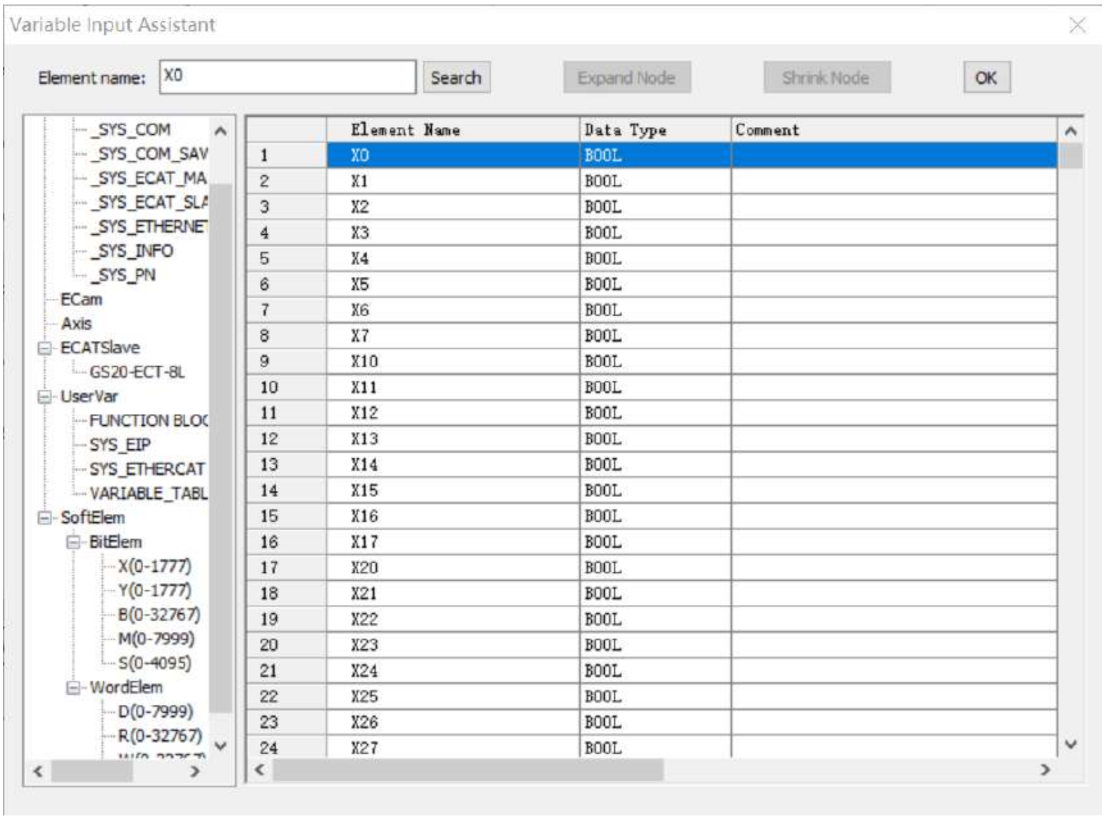
Variable		Channel	Type
_IQ1_0	...	GR20-16EMNL-BYTE_1 output byte 0	BYTE
_IQ1_1	...	GR20-16EMNL-BYTE_1 output byte 1	BYTE

b. (Optional) Set a customized variable.

Click ⇄ or ... . In the dialog box that is displayed, select a customized variable and click "OK".

Note

To use the default mapping variables, skip this step.



Element name: X0

	Element Name	Data Type	Comment
1	X0	BOOL	
2	X1	BOOL	
3	X2	BOOL	
4	X3	BOOL	
5	X4	BOOL	
6	X5	BOOL	
7	X6	BOOL	
8	X7	BOOL	
9	X10	BOOL	
10	X11	BOOL	
11	X12	BOOL	
12	X13	BOOL	
13	X14	BOOL	
14	X15	BOOL	
15	X16	BOOL	
16	X17	BOOL	
17	X20	BOOL	
18	X21	BOOL	
19	X22	BOOL	
20	X23	BOOL	
21	X24	BOOL	
22	X25	BOOL	
23	X26	BOOL	
24	X27	BOOL	

c. For example, configure the GR20_16EMPL byte operation through CH0 and the GR20_16EMPL bit operation through CH1 to map the corresponding input and output variables respectively. Set Output_pin2_chn of the corresponding port if the slave station is configured with output data.

d. Download and run the program.

The RUN indicator of the IO-Link master station flashes green and then turns steady green. The "0" indicators of the ports configured to the IO-Link mode flash green.

e. Connect the IO-Link slave station of the GR20-16EMPL model to port 0 and port 1.

The "0" indicators on port 0 and port 1 turn steady green. The "1" indicators on port 0 and port 1 are steady yellow, for example, Output_pin2_chn. At the slave station, the US indicator is steady green and the COM indicator flashes green.

4. Configure parameters of the IO-Link slave station.

Basic parameters of the IO-Link slave station can be configured by index 0x40n0 (n = 0 to 7). For details about indexes, see *"5.5.4.4 IO-Link Slave Configuration Data" on page 229*. I/O port information of the IO-Link slave station is configured as an example as follows:

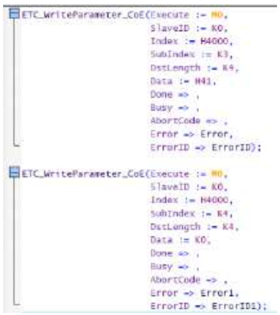
a. Run the ETC_ReadParameter and ETC_Write_Parameter instructions to set corresponding pin parameters to read and write corresponding indexes. This section describes operations of the ETC_Write_Parameter instruction, which are the same of the ETC_ReadParameter instruction.

Note

- The string type does not support reading or writing.
- The background DINT is of the signed 32-bit type, so an error occurs when UDINT data is read. If there is no signed number with more than 32 bits, test whether the value read is correct. If decimal display is incomplete, use hexadecimal display. In hexadecimal display, unsigned 32-bit numbers are displayed. Device IDs and other UDINT type data can be viewed in hexadecimal display.
- The ETC_ReadParameter and ETC_Write_Parameter instructions support reading and writing DINT type data, and do not support reading and writing 32-bit BYTE arrays (BYTE[32]).

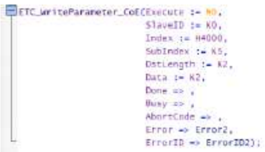
b. (Optional) Read the I/O port configuration information of the IO-Link slave station.

1). Set "Index" to "0X41" (hexadecimal) or "65" (decimal). The default value of "Subindex" of Inovance IO-Link slave station is 0, which does not need to be changed.



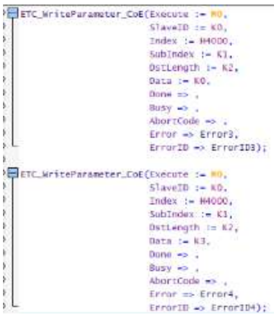
```
ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K3,
                       DstLength := K4,
                       Data := H41,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error,
                       ErrorID => ErrorID);

ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K4,
                       DstLength := K4,
                       Data := K0,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error1,
                       ErrorID => ErrorID1);
```

2). Set "Length" based on the index length of the IO-Link slave station in bytes, for example, 2 bytes. For details, see the "Object List" section in the IO-Link Slave Station User Guide.



```
ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K5,
                       DstLength := K2,
                       Data := K2,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error2,
                       ErrorID => ErrorID2);
```

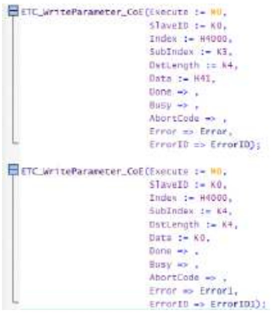
3). Set "Control" to 0 and then to 3 to read the I/O port configuration information of the IO-Link slave station.



```
ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K1,
                       DstLength := K2,
                       Data := K0,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error3,
                       ErrorID => ErrorID3);

ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K1,
                       DstLength := K2,
                       Data := K3,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error4,
                       ErrorID => ErrorID4);
```

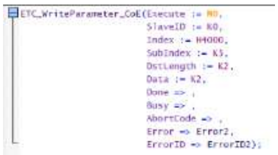
c. Configure the I/O port information of the IO-Link slave station.

1). Set "Index" to "0X41" (hexadecimal) or "65" (decimal). The default value of "Subindex" of Inovance IO-Link slave station is 0, which does not need to be changed.



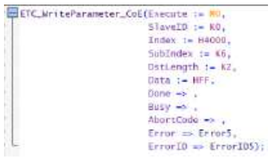
```
ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K3,
                       DstLength := K4,
                       Data := H41,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error,
                       ErrorID => ErrorID);

ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K4,
                       DstLength := K4,
                       Data := K0,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error1,
                       ErrorID => ErrorID1);
```

2). Set "Length" based on the index length of the IO-Link slave station in bytes. For details, see the "Object List" section in the IO-Link Slave Station User Guide.



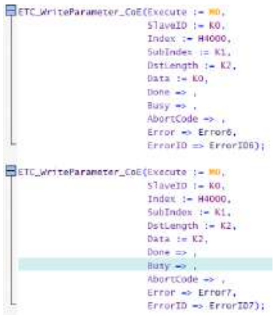
```
ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K5,
                       DstLength := K2,
                       Data := K2,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error2,
                       ErrorID => ErrorID2);
```

3). Set "Data". Value 1 indicates output and value 0 indicates input. For example, configure the low-order 8 bits of the IO-Link slave station as output and the high-order 8 bits as input, and set "Data[0]" to 255 and "Data[1]" to 0.



```
ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K6,
                       DstLength := K2,
                       Data := HFF,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error5,
                       ErrorID => ErrorID5);
```

4). Set "Control" to 0 and then to 2 to configure the I/O port information of the IO-Link slave station.



```
ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K1,
                       DstLength := K2,
                       Data := K0,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error6,
                       ErrorID => ErrorID6);

ETC_WriteParameter_CoE(Execute := M0,
                       SlaveID := K0,
                       Index := H4000,
                       SubIndex := K1,
                       DstLength := K2,
                       Data := K2,
                       Done => ,
                       Busy => ,
                       AbortCode => ,
                       Error => Error7,
                       ErrorID => ErrorID7);
```

5. Verify the configuration correctness.

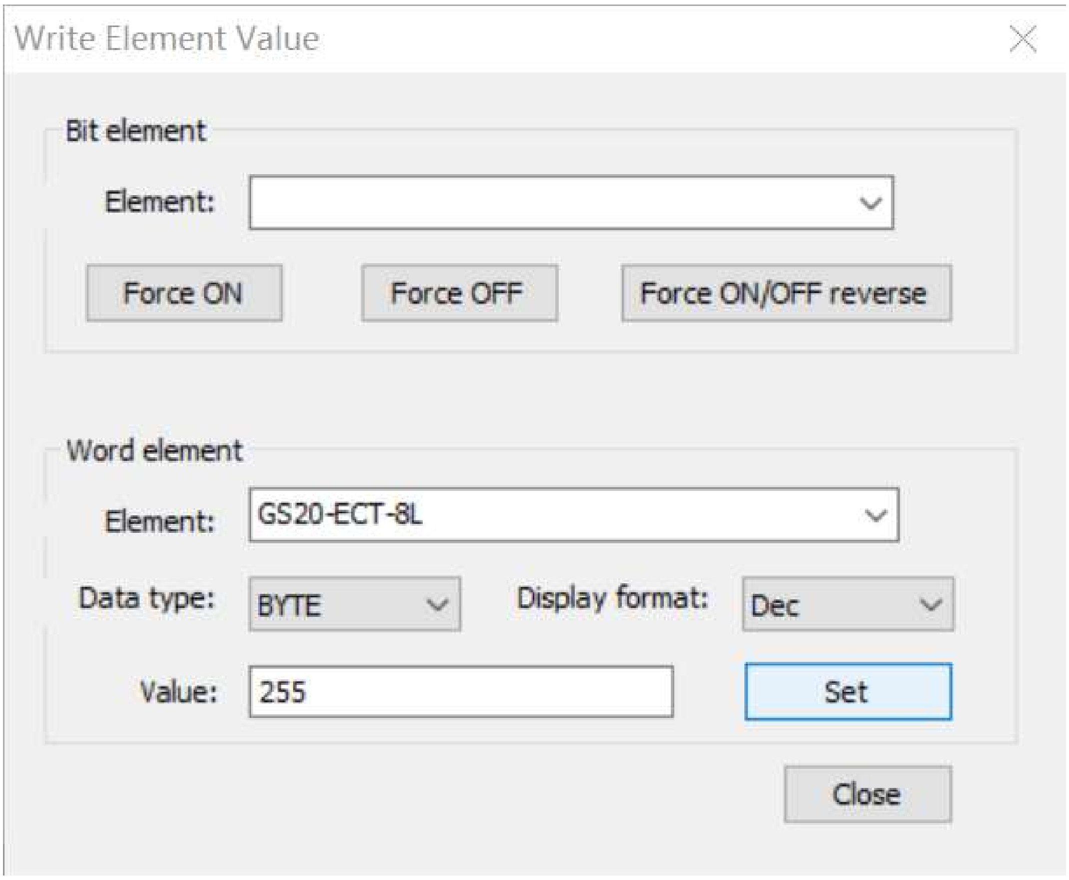
   After the configuration is completed, verify whether the configuration is correct. This section verifies whether port 0 and port 1 of GS20-ECT-8L are correctly configured based on the short connection between the low-order 8 bits (configured to output) and high-order 8 bits (configured to input) of GR20-16EMPL of the IO-Link slave station.

   a. Use an I/O cable for short connection between the low-order 8 bits and high-order 8 bits of GR20-16EMPL.

   b. Verify whether port 0 of GS20-ECT-8L is correctly configured.

      1). Use an IO-Link communication cable to connect the IO-Link port of GR20-16EMPL to port 0 of GS20-ECT-8L.

      2). Make the system enter the monitoring mode, and double-click "GS20-ECT-8L". On the page that is displayed, click "I/O function mapping", and double-click the low-order 8 bit output data of port 0 of GS20-ECT-8L. In the dialog box that is displayed, set "Value" to 255 to output signal 1 for the eight input ports of the IO-Link slave station of GR20-16EMPL.



Write Element Value

Bit element

Element:

Word element

Element: GS20-ECT-8L

Data type: BYTE

Display format: Dec

Value: 255

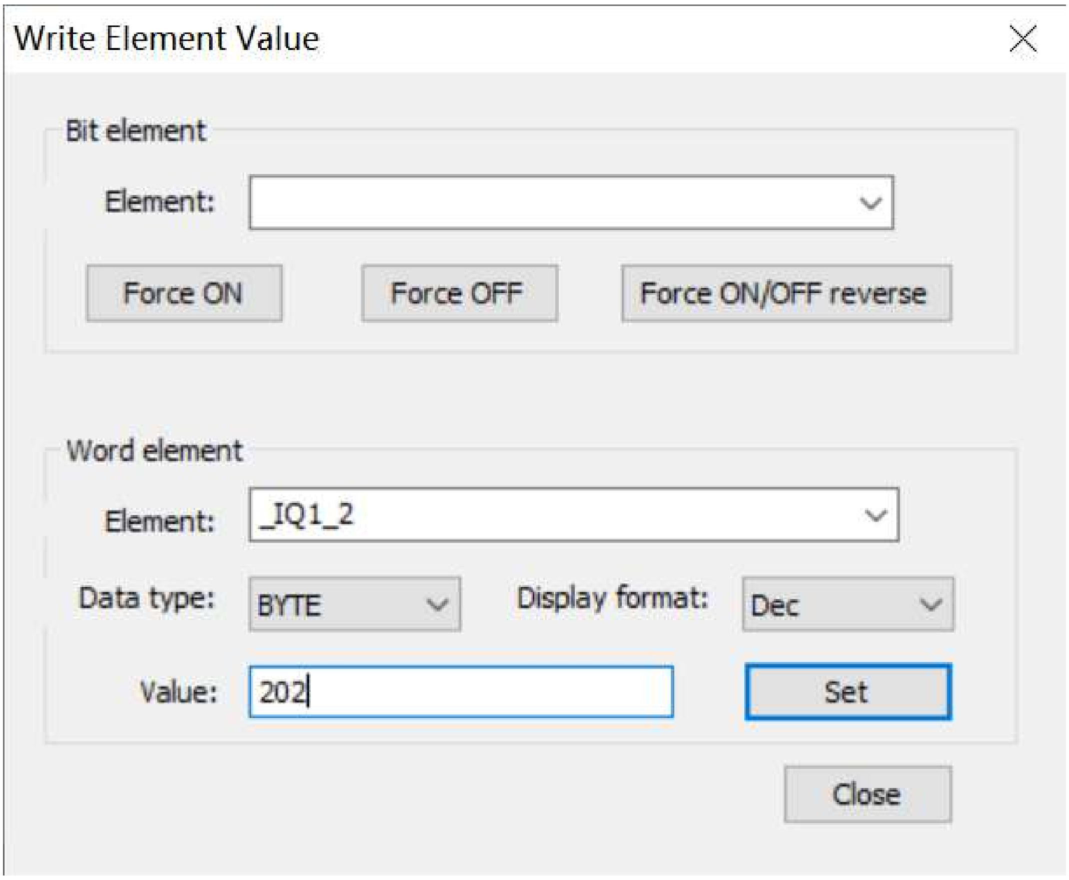
      3). Check the high-order 8 bit input data of port 0 of GS20-ECT-8L and the status of I/O indicators on the IO-Link slave station of GR20-16EMPL.

      If the high-order 8 bit input data of port 0 of GS20-ECT-8L is 255 and all the I/O indicators on the IO-Link slave station of GR20-16EMPL are turned on, the configuration is correct. Otherwise, the configuration is incorrect. Contact our technical support personnel for help.

   c. Verify whether port 1 of GS20-ECT-8L is correctly configured.

      1). Use an IO-Link communication cable to connect the IO-Link port of GR20-16EMPL to port 1 of GS20-ECT-8L.

2). Double-click "GS20-ECT-8L". On the page that is displayed, click "I/O function mapping", and double-click the low-order 8 bit output data of port 1 of GS20-ECT-8L. In the dialog box that is displayed, set "Value" to 202 as an example to output the "TRUE" or "FALSE" signal for the eight input ports of the IO-Link slave station of GR20-16EMPL. By default, BYTE variables are mapped, and the binary code of 202 is 11001010, indicating that the first, third, sixth, and seventh bits are set to TRUE and other bits are set to FALSE.



Write Element Value

Bit element

Element:

Word element

Element: _IQ1_2

Data type: BYTE

Display format: Dec

Value: 202

3). Check the high-order 8 bit input data of port 1 of GS20-ECT-8L and the status of I/O indicators on the IO-Link slave station of GR20-16EMPL.

If the high-order 8 bit input data of port 1 of GS20-ECT-8L is the same as the low-order 8 bit output data, and the I/O indicators on the IO-Link slave station of GR20-16EMPL are displayed accordingly, the configuration is correct. Otherwise, the configuration is incorrect. Contact our technical support personnel for help.

5.5.3 Fault Diagnosis

5.5.3.1 EtherCAT Diagnosis

LED Indicator		Description	Possible Cause	Solution
RUN	Off	The EtherCAT slave station is in the initialization state	The EtherCAT master station is not connected to the EtherCAT slave station	• Check that the configuration and parameter distribution are correct • Check that the communication address configuration is correct • Check that the network cable specifications (M12 interface, with a shielded Cat5e network cable) and length (within 100 m) meet the requirements
	Flashing green	The EtherCAT slave station is in the pre-operational state	The EtherCAT slave station is in a non-OP state	• Check that the EtherCAT slave station configuration is correct • Check that the EtherCAT slave station is faulty • Check that all EtherCAT slave stations are configured
	Flashing green only	The EtherCAT slave station is in the safe-operational state		
ERR	Flashing red	The EtherCAT communication network is abnormal	• The EtherCAT master station does not exchange data with the EtherCAT slave station • EtherCAT receives status conversion instructions that cannot be executed • EtherCAT synchronization fails • A watchdog error occurs in EtherCAT communication	• Check that the M12 network cable plus is correctly inserted • Check that the network cable is not damaged • Check that the PDO configuration is correct • Restart the power supply

5.5.3.2 IO-Link Diagnosis

The following table lists the error codes and their definitions.

Type	Fault Code	Description
System	0x0002	US supply overvoltage
	0x0003	US supply undervoltage
	0x0004	UA supply overvoltage
	0x0005	UA supply undervoltage
	0x0006	MCU temperature over 80°C
	0x0008	Port communication disconnection
	0x000c	Pin2 short circuit
IO-Link master station	0x1800	Port connection to no device
	0x1801	Parameter loading failure
	0x1802	Invalid vendor ID
	0x1803	Invalid device ID
	0x1804	Pin4 short circuit
	0x1805	PHY chip overtemperature
	0x1806	Pin1 short circuit
	0x1807	Pin1 overcurrent
	0x1808	Slave device event overflow
	0x1811	Pin4 short circuit (DO mode)
	0x1813	Pin4 overcurrent (DO mode)
	0x6000	Invalid cycle time
	0x6001	Incorrect version for slave station

5.5.4 Object List

5.5.4.1 Process Data

The input data of the IO-Link master station indicates TxPDO of the IO-Link slave station. If the IO-Link slave station with TxPDO is connected, the port of the IO-Link master station must be configured with input data. The following table defines TxPDO data of the IO-Link slave station. In the table, the entry names and slave implementation may be different, and n indicates the port number of the IO-Link master station.

Index 0xF100 CHn IO-Link Communication Status (for 0 ⩽ n ⩽ 7)				
Index	**Name**	**Description**	**Data Type**	**Default**
0xF100:00	Sub-index 00	Highest sub-index supported	USINT	0x08 (8dec)

0xF100:01	Sub-index 01	Status of communication between the master station and the slave station **Bit 0 to bit 3: IO-Link status** 0: Port inactive 1: Input mode 2: Output mode 3: Communication OP 4: Communication failure **Bit 4 to bit 7: error code** 00: No error 1: Watchdog error 2: Buffer overflow 3: Invalid device ID 4: Invalid vendor ID 5: Invalid version 6: Invalid frame function 7: Invalid cycle time 8: Invalid length for input process data 9: Invalid length for output data 10: Connection to no device 11: None	USINT	0x00 (0dec)
...	...		...	...
0xF100:08	Sub-index 08		USINT	0x00 (0dec)
Index 0x20n0 Ch.n Pin2 status monitoring input process data (for 0 ⩽ n ⩽ 7)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x20n0:00	Sub-index 00	Highest sub-index supported	USINT	0x02 (2dec)
0x20n0:01	Sub-index 01	Pin2 input process data • 0: Disable • 1: Enable	USINT	0x00 (0dec)
0x20n0:02	Sub-index 02	Short circuit status upon Pin2 configuration as output • 0: No short circuit information • 1: Short circuit	USINT	0x00 (0dec)
Index 0x20n1 Ch.n Pin4 and Pin1 short circuit status monitoring (for 0 ⩽ n ⩽ 7)				
Index	**Name**	**Description**	**Data Type**	**Default**

0x20n1:00	Sub-index 00	Highest sub-index supported	USINT	0x02 (2dec)
0x20n1:01	Sub-index 01	Short circuit status upon Pin4 configuration as output • 0: No short circuit information • 1: Short circuit	USINT	0x00 (0dec)
0x20n1:02	Sub-index 02	Pin1 short circuit status • 0: No short circuit information • 1: Short circuit	USINT	0x00 (0dec)
Index 0x2A02 hardware status monitoring				
Index	**Name**	**Description**	**Data Type**	**Default**
0x2A02:00	Sub-index 00	Highest sub-index supported	USINT	0x03(3dec)
0x2A02:01	Sub-index 01	Activator power voltage detection • 00: 18 < UA < 30.2 • 01: 11 < UA < 18 • 10: UA > 30.2 • 11: UA < 11	USINT	0x00 (0dec)
0x2A02:02	Sub-index 02	System power voltage detection • 00: 18 < US < 30.2 • 01: 11 < US < 18 • 10: US > 30.2 • 11: US < 11	USINT	0x00 (0dec)
0x2A02:03	Sub-index 03	MCU internal temperature detection • 00: 0 < internal temperature < 85 • 01: internal temperature < 0 • 10: internal temperature > 85 • 11: Reserved	USINT	0x00 (0dec)
Index 0x30n8 Ch.n Pin2 output data in DO mode (for 0 ⩽ n ⩽ 7)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x30n8:00	Sub-index 00	Highest sub-index supported	USINT	0x01 (1dec)
0x30n8:01	Sub-index 01	Pin2 output data • 0: Disable • 1: Enable	BIT	0
Index 0x60n0 Ch.n IO-Link input data (for 0 ⩽ n ⩽ 7)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x60n0:00	Sub-index 00	Highest sub-index supported	USINT	0x00 (0dec)

0x60n0:01	TxPDO 01	IO-Link input process data [00]	UDINT	0x00 (0dec)
...	...	...	...	...
0x60n0:20	TxPDO 32	IO-Link input process data [32]	UDINT	0x00 (0dec)
Index 0x70n0 Ch.n output data (for 0 ⩽ n ⩽ 7)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x70n0:00	Sub-index 00	Highest sub-index supported	USINT	0x00 (0dec)
0x70n0:01	RxPDO 01	IO-Link output process data [00]	UDINT	0x00 (0dec)
...	...	...	...	...
0x70n0:20	RxPDO 32	IO-Link output process data [32]	UDINT	0x00 (0dec)

The following table lists configuration data of the IO-Link port.

Index 0x20n3 Ch.n Pin4 parameters in DO mode (for 0 ⩽ n ⩽ 7)					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
0x20n3:00	IO Settings Ch.1- 8	Highest sub-index supported	USINT	RW	0x1(1dec)
0x20n3:01	Pin4 safe state	Safe state preset value upon Pin4 configuration as output • 0x00: Output 0 upon a communication error • 0x01: Output 1 upon a communication error • 0x02: Last output value upon a communication error	UDINT	RW	0x00 (0dec)
Index 0x20n2 Ch.n Pin2 parameters in DO mode (for 0 ⩽ n ⩽ 7)					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
0x20n2:00	IO Settings Ch.1- 8	Highest sub-index supported	USINT	RW	0x1(1dec)
0x20n2:01	Pin2 safe state	Safe state preset value upon Pin2 configuration as output • 0x00: Output 0 upon a communication error • 0x01: Output 1 upon a communication error • 0x02: Last output value upon a communication error	UDINT	RW	0x00 (0dec)

5.5.4.2 EtherCAT Object Dictionary Data (CoE Object)

The object dictionary of the EtherCAT IO-Link master station contains SDO-based addressing objects and the standard objects and manufacturer objects supported by the IO-Link master station. The ETG.1000.6: Application Layer protocol specification describes standard objects and ETG.5001-6220 describes the modular equipment profile objects. In addition, the manufacturer objects can be addressed by combination of indexes and sub-indexes. Sub-index 0 indicates the number of sub-indexes or highest-level sub-indexes.

Index 0x1000 EtherCAT slave station device type					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**

1000:00	Device type	Device type of the EtherCAT slave station	UDINT	RO	0x1389 (5001dec)
Index 0x1001 error register					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
1001:00	Error register	Error register	USINT	RO	0x00
Index 0x1001 EtherCAT slave station device name					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
1008:00	Device name	Device name	STRING(11)	RO	GS20-ECT-8L
Index 0x1008 EtherCAT slave station hardware version information					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
1009:00	Hardware version	Hardware version of the EtherCAT slave station	STRING(16)	RO	A00.01
Index 0x100A EtherCAT slave station protocol stack software version information					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
100A:00	Slave version	Software version of the EtherCAT slave station protocol stack	STRING(4)	RO	5.13
Index 0x100B IO-LINK master station product-related version information					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
100B:00	Software version	Highest sub-index supported	USINT	RO	0x04
100B:01	App version	Application software version	UDINT	RO	0x10100000
100B:02	FPGA version	FPGA software version	UDINT	RO	0x10100000
100B:03	IOLM version	Software version of the IO-Link master station	UDINT	RO	0x0305
Index 0x1018 EtherCAT slave station identity information					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
1018:00	Identity	Highest sub-index supported	USINT	RO	0x04 (4dec)
1018:01	Vendor ID	Vendor ID of the EtherCAT slave station	UDINT	RO	0x00100000
1018:02	Product code	Product code of the EtherCAT slave station	UDINT	RO	0x10F42EE1
1018:03	Revision	Firmware version of the product application	STRING(11)	RO	1.1.0.0
1018:04	Serial number	Production serial number	UDINT	RO	0x15FA66
Index 0x10F3 diagnosis history					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
10F3:00	Diagnosis history	Highest sub-index supported	USINT	RO	0x16

10F3:01	Maximum message	Maximum number of diagnosis messages	USINT	RO	0x14 (20dec)
10F3:02	Newest message	Sub-index of the latest diagnosis message	USINT	RO	0x00000000 (0dec)
10F3:03	Newest acknowledged message	Latest acknowledged message	USINT	RW	0x00000000 (0dec)
10F3:04	New message available	New diagnosis information	BOOL	RO	0
10F3:05	Flags	Setting for send and store diagnosis messages	UINT	RW	0x00000000 (0dec)
10F3:06	Diagnosis message 01	Diagnosis information 1	ARRAY [0..27] OF BYTE	RO	0x00000000 (0dec)
...	...	...	...	...	...
10F3:40	Diagnosis message 64	Diagnosis information 64	ARRAY [0..27] OF BYTE	RO	0x00000000 (0dec)
Index 0x3010 ESC port 0 error counter					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
3010:00	Port0 error counter	Highest sub-index supported	USINT	RO	0x04 (4dec)
3010:01	Port0 invalid frame counter	Number of invalid frames for ESC port 0	USINT	RO	0x00 (0dec)
3010:02	Port0 Rx error counter	Number of error frames for ESC port 0	USINT	RO	0x00 (0dec)
3010:03	Port0 forwarded Rx error counter	Number of error loopback frames for ESC port 0	USINT	RO	0x00 (0dec)
3010:04	Port0 lost link counter	Number of lost frames for ESC port 0	USINT	RO	0x00 (0dec)
Index 0x3011 ESC port 0 error counter					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
3011:00	Port1 error counter	Highest sub-index supported	USINT	RO	0x04 (4dec)
3011:01	Port1 invalid frame counter	Number of invalid frames for ESC port 1	USINT	RO	0x00 (0dec)
3011:02	Port1 Rx error counter	Number of error frames for ESC port 1	USINT	RO	0x00 (0dec)
3011:03	Port1 forwarded Rx error counter	Number of error loopback frames for ESC port 1	USINT	RO	0x00 (0dec)

3011:04	Port1 lost link counter	Number of lost frames for ESC port 1	USINT	RO	0x00 (0dec)
Index 0x3012 ESC error counter					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
3012:00	ESC error counter	Highest sub-index supported	USINT	RO	0x04 (4dec)
3012:01	ECAT processing unit error counter	Error counter for the ESC processing unit	USINT	RO	0x00 (0dec)
3012:02	PDI error counter	PDI error counter	USINT	RO	0x00 (0dec)
3012:03	Watchdog counter process data	Number of watchdogs for process data	USINT	RO	0x00 (0dec)
3012:04	Watchdog counter PDI	Number of PDI watchdogs	USINT	RO	0x00 (0dec)
Index 0x3016 station address					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
3016:00	Station address	Highest sub-index supported	USINT	RO	0x04 (4dec)
3016:01	Rotary switch value	DIP switch value	USINT	RO	0x00 (0dec)
3016:02	Configured station address	Configured station address	USINT	RO	0x00 (0dec)
3016:03	Configured station alias	Configured station alias	USINT	RO	0x00 (0dec)
3016:04	Alias in EEPROM	Alias in EEPROM	USINT	RO	0x00 (0dec)

The following table lists configuration data of the IO-Link port.

Index 0x80n0 Ch.n IO-Link port configuration data (for 0 ⩽ n ⩽ 7)					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
0x80n0:00	IO Settings Ch.1- 8	Highest sub-index supported	USINT	RW	0x28 (40dec)
0x80n0:04	Device ID	ID of the IO-Link device	UDINT	RW	0x00000000 (0dec)
0x80n0:05	VendorID	Vendor ID of the IO-Link device	UDINT	RW	0x00000000 (0dec)
0x80n0:06	Product ID	Product ID of the IO-Link device	USINT	RW	0x00000000 (0dec)
0x80n0:08	Serial number	Serial number of the IO-Link device	USINT	RW	0x00000000 (0dec)
0x80n0:20	IO-Link revision	Version of the specification for IO-Link device communication • Bit 0 to bit 3: Minor version • Bit 4 to bit 7: Major version	USINT	RW	0x00 (0dec)
0x80n0:21	Frame capability	Reserved	USINT	RW	0x00 (0dec)

0x80n0:22	Min cycle time	Cycle time① between the IO-Link master station and slave station, which is transmitted by IO-Link data frames of the minimum cycle • Bit 6 to bit 7: Time base • Bit 0 to bit 5: Ratio • 0x00: The IO-Link master station automatically uses the update time of the IO-Link device	USINT	RW	0x00 (0dec)
0x80n0:23	Offset time	Reserved	USINT	RW	0x00 (0dec)
0x80n0:24	Process data in length	Number of bits in the input process data transmitted by IO-Link data frames, which is recorded as 255 for 256 bits	USINT	RW	0x00 (0dec)
0x80n0:25	Process data out length	Number of bits in the output process data transmitted by IO-Link data frames, which is recorded as 255 for 256 bits	USINT	RW	0x00 (0dec)
0x80n0:26	Compatible ID	Reserved	UINT	RW	0x0000 (0dec)
0x80n0:27	Reserved	Reserved	UINT	RW	0x0000 (0dec)
0x80n0:28	Master control	**Bit 0 to bit 3:** 0: Port not used 1: Port configured as DI mode 2: Port configured as DO mode 3: Port configured as the IO-Link automatic mode 4: Port configured as the IO-Link verification mode **Bit 4 to bit 7:** 0: Invalid process data upon network disconnection 1: Output 0 upon network disconnection 2: Output of the last cycle upon network disconnection **Bit 8 to bit 15:** 0: Not verifying device parameter information 1: Verifying vendor ID and device ID in V1.0 2: Verifying vendor ID and device ID in V1.1 3: Enabling backup and restoration of slave station configuration data 4: Only enabling restoration of slave station configuration data * V1.0 and V1.1 indicates the versions of the IO-Link slave station protocol stack. * If the modes specified by values 3 and 4 are used, the port must be configured to the verification mode.	UINT	RW	0x0000 (0dec)
Index 0x90n0 Ch.n IO-Link port configuration data (for 0 ⩽ n ⩽ 7)					

Index	Name	Description	Data Type	Code	Default
0x90n0:00	IO Settings Ch.1- 8	Highest sub-index supported	USINT	R	0x28 (40dec)
0x90n0:04	Device ID	Obtained device ID of the IO-Link slave device	UDINT	R	0x00000000 (0dec)
0x90n0:05	VendorID	Obtained vendor ID of the IO-Link slave device	UDINT	R	0x00000000 (0dec)
0x90n0:06	Product ID	Obtained product ID of the IO-Link slave device (not supported)	USINT	R	0x00000000 (0dec)
0x90n0:08	Serial number	Obtained serial number of the IO-Link slave device (not supported)	USINT	R	0x00000000 (0dec)
0x90n0:20	IO-Link revision	Version of the specification for IO-Link device communication • Bit 0 to bit 3: Minor version • Bit 4 to bit 7: Major version	USINT	R	0x00 (0dec)
0x90n0:21	Frame capability	Reserved	USINT	R	0x00 (0dec)
0x90n0:22	Min cycle time	Cycle time① between the IO-Link master station and slave station, which is transmitted by IO-Link data frames of the minimum cycle • Bit 6 to bit 7: Time base • Bit 0 to bit 5: Ratio	USINT	R	0x00 (0dec)
0x90n0:23	Offset time	Reserved	USINT	R	0x00 (0dec)
0x90n0:24	Process data in length	Obtained number of bits in the input process data transmitted by IO-Link slave station data frames, which is displayed as 255 for 256 bits	USINT	R	0x00 (0dec)
0x90n0:25	Process data out length	Obtained number of bits in the output process data transmitted by IO-Link slave station data frames, which is displayed as 255 for 256 bits	USINT	R	0x00 (0dec)
0x90n0:26	Compatible ID	Reserved	UINT	R	0x0000 (0dec)
0x90n0:27	Reserved	Reserved	UINT	R	0x0000 (0dec)

① For details about the cycle time, see the following table.

Time Baseline Code	Time Baseline Value	Calculation Method	Cycle Time
00	0.1 ms	Multiplier x Time baseline	0.4 ms to 6.3 ms
01	0.4 ms	6.4 ms + Multiplier x Time baseline	6.4 ms to 31.6 ms
10	1.6 ms	32.0 ms + Multiplier x Time baseline	32.0 ms to 132.8 ms
11	Reserved	Reserved	Reserved

5.5.4.3 Configuration Data for Process Data Communication

EtherCAT PDO communication is managed by PDO assignment, PDO mapping, and process data object dictionaries. PDO assignment and PDO mapping are described and sampled as follows:

PDO assignment

PDO assignment, which is divided into two object dictionaries to receive PDO assignment and transmit PDO assignment, is used to configure PDO mapping. Indexes for the two dictionaries are respectively 0x1C12 and 0x1C13.

Index 0x1C12 RxPDO assignment				
Index	**Name**	**Description**	**Data Type**	**Default**
0x1C12:00	Sub-index 00	Assignment of output process data	USINT	0x08 (8dec)
0x1C12:01	Sub-index 01	Assignment of output process data of index 1	DT1C12ARR	0x1600 (5632dec)
0x1C12:02	Sub-index 02	Assignment of output process data of index 2	DT1C12ARR	0x1601 (5633dec)
0x1C12:03	Sub-index 03	Assignment of output process data of index 3	DT1C12ARR	0x1602 (5634dec)
0x1C12:04	Sub-index 04	Assignment of output process data of index 4	DT1C12ARR	0x1603 (5633dec)
0x1C12:05	Sub-index 05	Assignment of output process data of index 5	DT1C12ARR	0x1604 (5634dec)
0x1C12:06	Sub-index 06	Assignment of output process data of index 6	DT1C12ARR	0x1605 (5635dec)
0x1C12:07	Sub-index 07	Assignment of output process data of index 7	DT1C12ARR	0x1606 (5634dec)
0x1C12:08	Sub-index 08	Assignment of output process data of index 8	DT1C12ARR	0x1607 (5635dec)
Index 0x1C13 TxPDO assignment				
Index	**Name**	**Description**	**Data Type**	**Default**
0x1C13:00	Sub-index 00	Assignment of input process data	USINT	0x08 (8dec)
0x1C13:01	Sub-index 01	Assignment of input process data of index 1	DT1C13ARR	0x1A00 (6656dec)
0x1C13:02	Sub-index 02	Assignment of input process data of index 2	DT1C13ARR	0x1A01 (6657dec)
0x1C13:03	Sub-index 03	Assignment of input process data of index 3	DT1C13ARR	0x1A02 (6657dec)
0x1C13:04	Sub-index 04	Assignment of input process data of index 4	DT1C13ARR	0x1A03 (6658dec)
0x1C13:05	Sub-index 05	Assignment of input process data of index 5	DT1C13ARR	0x1A04 (6658dec)

0x1C13:06	Sub-index 06	Assignment of input process data of index 6	DT1C13ARR	0x1A05 (6659dec)
0x1C13:07	Sub-index 07	Assignment of input process data of index 7	DT1C13ARR	0x1A06 (6659dec)
0x1C13:08	Sub-index 08	Assignment of input process data of index 8	DT1C13ARR	0x1A07 (6660dec)

PDO mapping

PDO mapping is used to map the process data object dictionary that requires communication. PDO mapping is divided into RxPDO mapping and TxPDO mapping, and the index ranges for them are respectively 0x1600 to 0x17FF and 0x1A00 to 0x1BFF.

The object dictionary for PDO mapping contains the values of indexes, sub-indexes, and length of process data in the object dictionary for PDO communication.

Bit	31	...	16	15	...	8	7	...	0
Descrip-tion	Index			Sub-index			Object length		

Index 0x1A0n Ch.n input process data mapping (for 0 ⩽ n ⩽ 7)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x1A0n:00	Sub-index 00	Input process data mapping	USINT	0x00 (0dec)
0x1A0n:01	Sub-index 01	1. Input process data mapping	UDINT	0x70n0:01,08
...	...	...	...	...
0x1A0n:40	Sub-index 64	64. Input process data mapping	UDINT	0x70n0:40,08
Index 0x1A10 Pin2 input process data mapping (8 Ch)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x1A10:00	Sub-index 00	Input process data mapping for Pin2	USINT	0x08 (8dec)
0x1A10:01	Sub-index 01	Input process data mapping for Pin2 through CH0	BIT	0x2000:01,01
0x1A10:02	Sub-index 02	Input process data mapping for Pin2 through CH1	BIT	0x2010:01,01
0x1A10:03	Sub-index 03	Input process data mapping for Pin2 through CH2	BIT	0x2020:01,01
0x1A10:04	Sub-index 04	Input process data mapping for Pin2 through CH3	BIT	0x2030:01,01
0x1A10:05	Sub-index 05	Input process data mapping for Pin2 through CH4	BIT	0x2040:01,01

0x1A10:06	Sub-index 06	Input process data mapping for Pin2 through CH5	BIT	0x2050:01,01
0x1A10:07	Sub-index 07	Input process data mapping for Pin2 through CH6	BIT	0x2060:01,01
0x1A10:08	Sub-index 08	Input process data mapping for Pin2 through CH7	BIT	0x2070:01,01
Index 0x1A13 Pin1 short circuit process data mapping (8 Ch)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x1A13:00	Sub-index 00	Short circuit process data mapping for Pin1	USINT	0x08 (8dec)
0x1A13:01	Sub-index 01	Short circuit process data mapping for Pin1 through CH0	BIT	0x2001:02,01
0x1A13:02	Sub-index 02	Short circuit process data mapping for Pin1 through CH1	BIT	0x2011:02,01
0x1A13:03	Sub-index 03	Short circuit process data mapping for Pin1 through CH2	BIT	0x2021:02,01
0x1A13:04	Sub-index 04	Short circuit process data mapping for Pin1 through CH3	BIT	0x2031:02,01
0x1A13:05	Sub-index 05	Short circuit process data mapping for Pin1 through CH4	BIT	0x2041:02,01
0x1A13:06	Sub-index 06	Short circuit process data mapping for Pin1 through CH5	BIT	0x2051:02,01
0x1A13:07	Sub-index 07	Short circuit process data mapping for Pin1 through CH6	BIT	0x2061:02,01
0x1A13:08	Sub-index 08	Short circuit process data mapping for Pin1 through CH7	BIT	0x2071:02,01
Index 0x1A14 system hardware status monitoring process data mapping				
Index	**Name**	**Description**	**Data Type**	**Default**
0x1A14:00	Sub-index 00	Process data mapping for status monitoring of system hardware	USINT	0x03(3dec)
0x1A14:01	Sub-index 01	Process data mapping for status monitoring of system power voltage	UDINT	0x2A02:01,08
0x1A14:02	Sub-index 02	Process data mapping for status monitoring of activator power voltage	UDINT	0x2A02:02,08

0x1A14:03	Sub-index 03	Process data mapping for status monitoring of MCU internal temperature	UDINT	0x2A02:03,08
Index 0x1A80 master and slave station communication status monitoring process data mapping				
Index	**Name**	**Description**	**Data Type**	**Default**
0x1A81:00	Sub-index 00	Process data mapping for status monitoring of master and slave station communication	USINT	0x08 (8dec)
0x1A81:01	Sub-index 01	Process data mapping for status monitoring of master and slave station communication through CH0	UDINT	0xF100:01,08
0x1A81:02	Sub-index 02	Process data mapping for status monitoring of master and slave station communication through CH1	UDINT	0xF100:02,08
0x1A81:03	Sub-index 03	Process data mapping for status monitoring of master and slave station communication through CH2	UDINT	0xF100:03,08
0x1A81:04	Sub-index 04	Process data mapping for status monitoring of master and slave station communication through CH3	UDINT	0xF100:04,08
0x1A81:05	Sub-index 05	Process data mapping for status monitoring of master and slave station communication through CH4	UDINT	0xF100:05,08
0x1A81:06	Sub-index 06	Process data mapping for status monitoring of master and slave station communication through CH5	UDINT	0xF100:06,08
0x1A81:07	Sub-index 07	Process data mapping for status monitoring of master and slave station communication through CH6	UDINT	0xF100:07,08

0x1A81:08	Sub-index 08	Process data mapping for status monitoring of master and slave station communication through CH7	UDINT	0xF100:08,08
Index 0x160n Ch.n output process data mapping (for 0 ⩽ n ⩽ 7)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x160n:00	Sub-index 00	Output process data mapping	USINT	0x00 (0dec)
0x160n:01	Sub-index 01	1. Output process data mapping	UDINT	0x60n0:01,08
...	...	...	...	...
0x160n:20	Sub-index 32	32. Output process data mapping	UDINT	0x60n0:40,08
Index 0x1620 Pin2 output process data mapping (8 Ch)				
Index	**Name**	**Description**	**Data Type**	**Default**
0x1620:00	Sub-index 00	Output process data mapping for Pin2	USINT	0x08 (8dec)
0x1620:01	Sub-index 01	Output process data mapping for Pin2 through CH0	BIT	0x3008:01,01
0x1620:02	Sub-index 02	Output process data mapping for Pin2 through CH1	BIT	0x3018:01,01
0x1620:03	Sub-index 03	Output process data mapping for Pin2 through CH2	BIT	0x3028:01,01
0x1620:04	Sub-index 04	Output process data mapping for Pin2 through CH3	BIT	0x3038:01,01
0x1620:05	Sub-index 05	Output process data mapping for Pin2 through CH4	BIT	0x3048:01,01
0x1620:06	Sub-index 06	Output process data mapping for Pin2 through CH5	BIT	0x3058:01,01
0x1620:07	Sub-index 07	Output process data mapping for Pin2 through CH6	BIT	0x3068:01,01
0x1620:08	Sub-index 08	Output process data mapping for Pin2 through CH7	BIT	0x3078:01,01

5.5.4.4 IO-Link Slave Configuration Data

Index 0x40n0 Ch.n IO-Link port slave configuration parameter read/write (for 0 ⩽ n ⩽ 7)					
Index	**Name**	**Description**	**Data Type**	**Code**	**Default**
0x40n0:00	subindex0	Highest sub-index supported	USINT	RO	0x07 (7dec)

0x40n0:01	Control	• 0: Disabled • 0→2: Write parameter • 0→3: Read parameter	UDINT	RW	0x00 (0dec)
0x40n0:02	Status	• 0x00: No error • 0x02: Read success • 0x40: Error	UDINT	RW	0x00 (0dec)
0x40n0:03	Index	Index number of the slave station	USINT	RW	0x00 (0dec)
0x40n0:04	Subindex	Sub-index number of the slave station	USINT	RW	0x00 (0dec)
0x40n0:05	Length	Data length, in byte	UINT	RW	0x00 (0dec)
0x40n0:06	Data	Data	ARRAY [0..31] OF BYTE	RW	0x00 (0dec)
0x40n0:07	Fault Code	• 0x1: Operation not supported • 0x3: Device access failure • 0x4: Unauthorized operation • 0x5: Slave device in non-OP state • 0x34: Length error • 0x36: Invalid operation due to busy master station • 0x39: Port disabled	UINT	RO	0x00 (0dec)

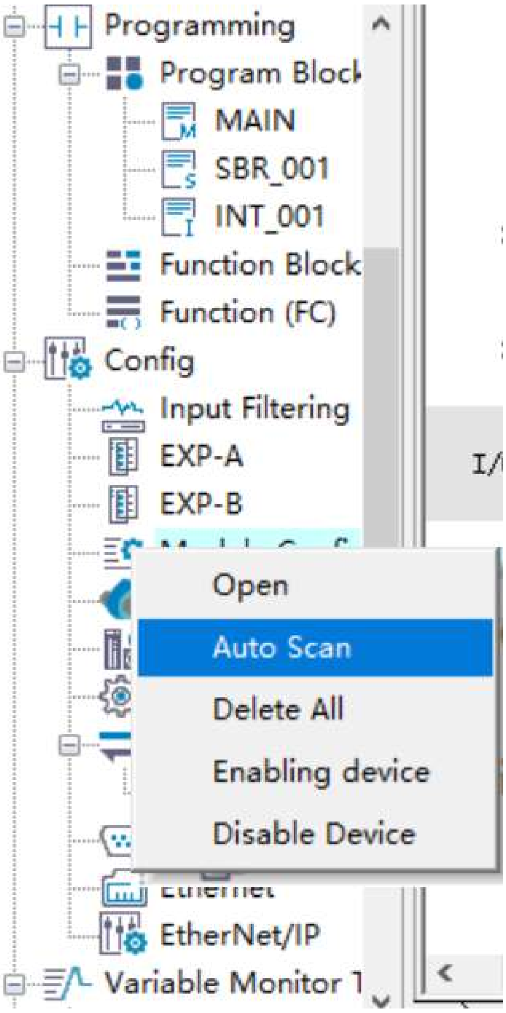
5.6 Basic Operations of Local Modules

5.6.1 Scanning Local Modules Automatically (Easy)

1. Right-click "Module Config" and select "Auto Scan".

Note

Both modules and extension cards are scanned and the scanning follows the same specifications.



2. In the dialog box that is displayed, click "Start Scan". If the PLC is running, click "Yes" in the popup window to switch to the stopped state.
3. Click "Start Scan". After the scanning is completed, the list of scanned modules is displayed. If the mounted modules are inconsistent with the modules configured in the background, they are marked red.
4. Click "Update Config".
   The "Save current axises." window is described as follows:

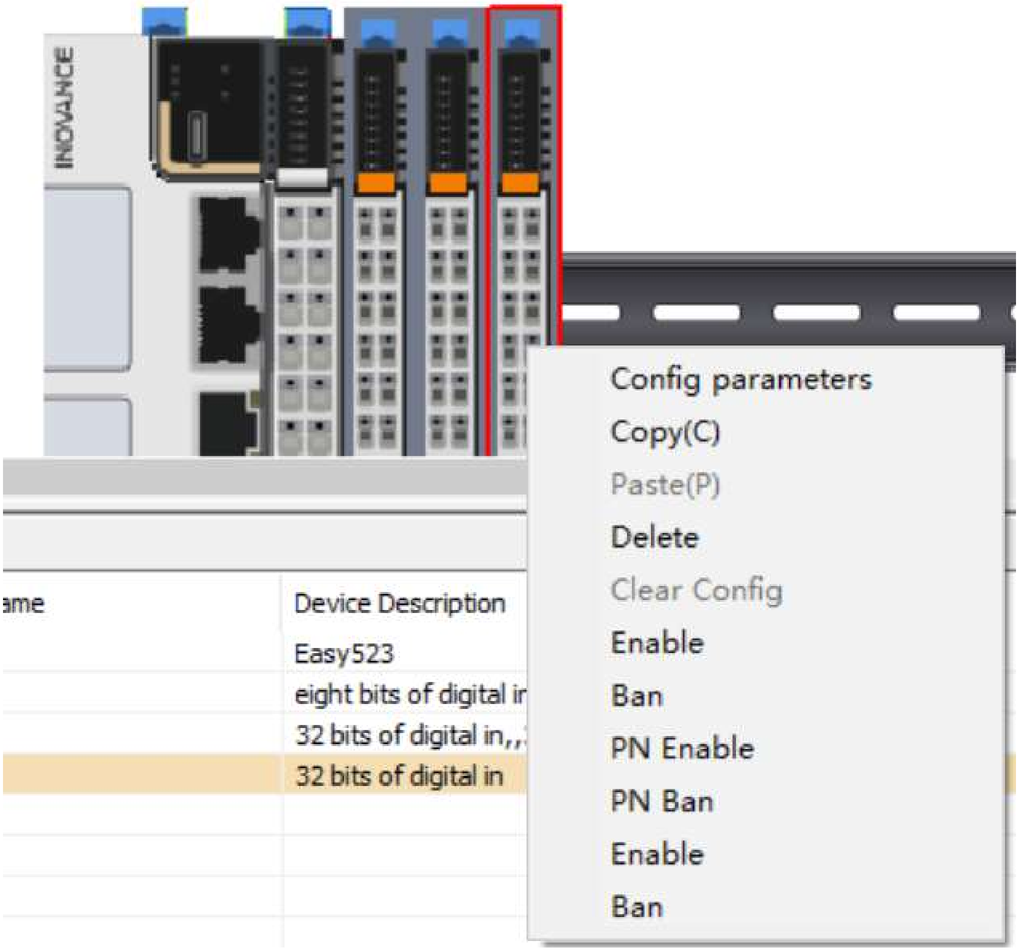
- Deselected: All the background module configurations are deleted. The default configurations of scanned modules are added according to the scan result.
- Selected: Compare the mounted modules with the modules configured in the background. If they are consistent, the mounted modules are not replaced or updated. If they are inconsistent, the background modules of the corresponding slots are deleted, the default configured modules are added, and the modules are compared in sequence by slot.

Click "OK" to add the scan result.

5.6.2 Disabling Local Modules

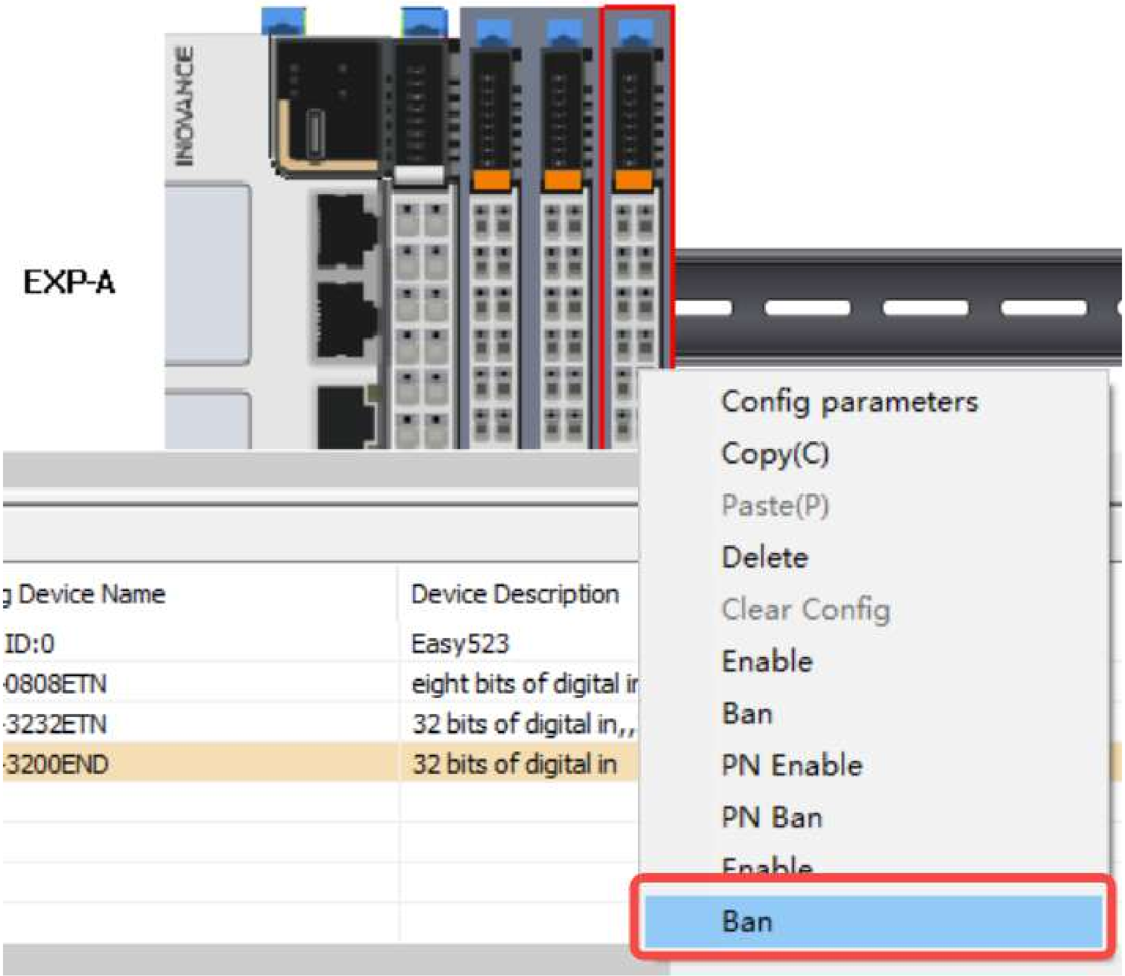
To use the function of disabling local extension modules, ensure that the physical extension modules removed are the same as the modules to be disabled in the module list of the software. The procedure is as follows:

1. In the module list, select a module to be disabled and right-click the module.



ame	Device Description
	Easy523
	eight bits of digital ir
	32 bits of digital in,,:
	32 bits of digital in

2. Choose "Ban" from the drop-down list, and download the program to the PLC device.



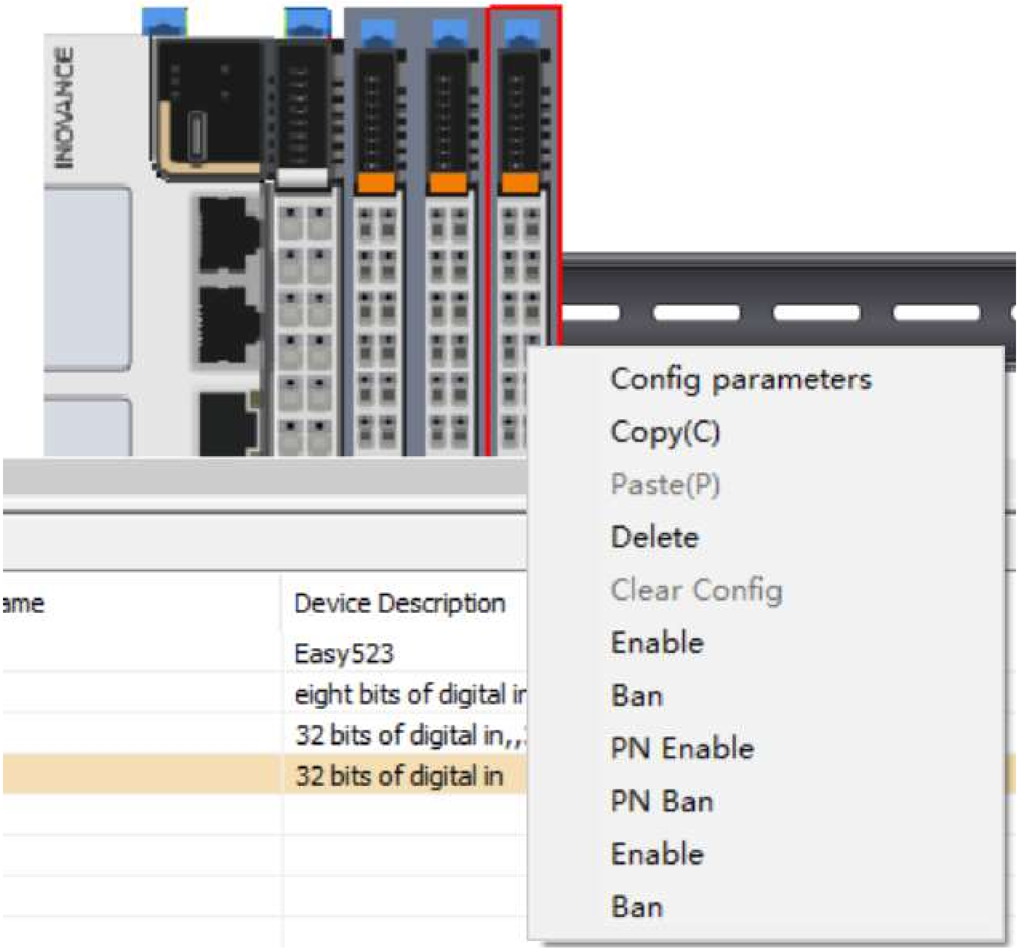
g Device Name	Device Description
ID:0	Easy523
0808ETN	eight bits of digital ir
3232ETN	32 bits of digital in,,
3200END	32 bits of digital in

3. After downloading the program, restart the PLC device.

5.6.3 Enabling Local Modules

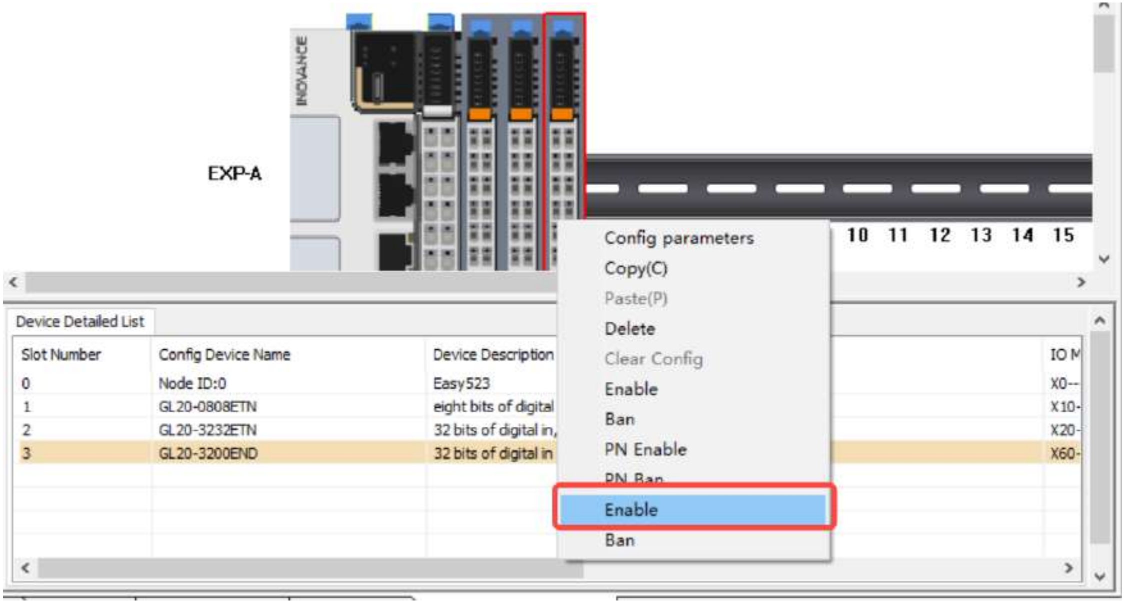
To use the function of enabling local extension modules, ensure that the physical extension modules installed are the same as the modules to be enabled in the module list of the software. The procedure is as follows:

1. In the module list, select a module to be enabled and right-click the module.



ame	Device Description
	Easy523
	eight bits of digital ir
	32 bits of digital in,,:
	32 bits of digital in

2. Choose "Enable" from the drop-down list, and download the program to the PLC device.



Device Detailed List

Slot Number	Config Device Name	Device Description	IO M
0	Node ID:0	Easy523	X0--
1	GL20-0808ETN	eight bits of digital	X10-
2	GL20-3232ETN	32 bits of digital in,	X20-
3	GL20-3200END	32 bits of digital in	X60-

3. After downloading the program, restart the PLC device.

6 Serial Communication

6.1 Overview

H5U provides a serial communication port that supports baud rates of 9600 bps, 19200 bps, 38400 bps, 57600 bps, and 115200 bps.

Serial ports supported by the Easy series are listed in the following table.

	Easy300 Series			Easy500 Series				
	Easy301	Easy302	Easy320	Easy501	Easy502	Easy521	Easy522	Easy523
Serial communication	One RS232 port and one RS485 port, supporting free protocol for serial ports	One RS232 port and one RS485 port, scalable up to two RS485/ RS232 ports, supporting free protocol for serial ports	One RS485 port, scalable up to two RS485/RS232 ports, supporting free protocol for serial ports	One RS485 port, scalable up to two RS485/RS232 ports, supporting free protocol for serial ports		One RS485 port, scalable up to two RS485/RS232 ports, supporting free protocol for serial ports		

Table 6–1 Communication protocol

Communication Protocol	Description
Free protocol	Freely sends/receives data with the SerialRS instruction
Modbus-RTU master station	A standard Modbus-RTU master station that reads/writes data from/to a slave station through Modbus configuration
Modbus-RTU slave station	A standard Modbus-RTU slave station
Modbus-ASCII master station	A standard Modbus-ASCII master station that reads/writes data from/to a slave station through Modbus configuration
Modbus-ASCII slave station	A standard Modbus-ASCII slave station

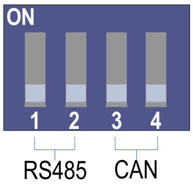
6.2 Serial Communication Network

You are recommended to use shielded twisted pairs for the RS485 bus and use twisted pairs to connect RS485+ and RS485–. A 120 Ω termination resistor is connected at both ends of the bus to prevent signal reflection. The reference grounds (GND) of RS485 signals on all nodes are connected together. A maximum of 31 nodes are supported and the distance between each node and the bus must be less than 3 m.

H5U series

Communication termination resistor DIP switch

The communication termination resistor DIP switch is located in the battery bay. ON means the termination resistor is connected (factory default: OFF). The switch schematic diagram is as follows, in which 1 and 2 are used for RS485 communication, and 3 and 4 are used for CAN communication.



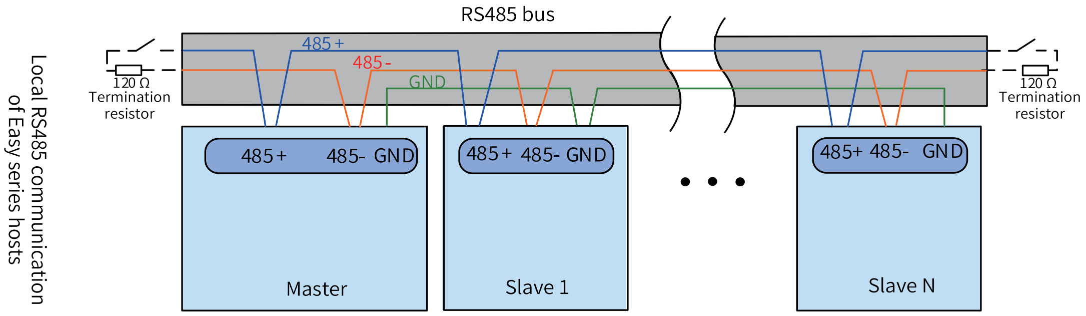
Networking of RS485 serial communication

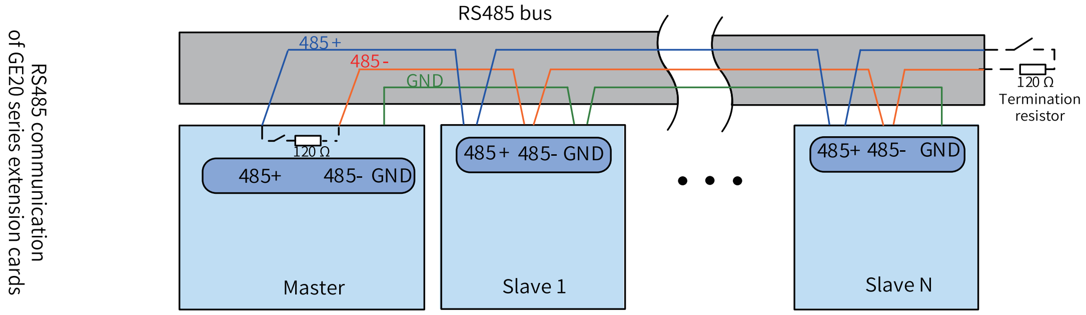
The following figure shows the RS485 bus topology.

The RS485 port of H5U has a 120 Ω termination resistor which can be turned on or off by setting the DIP switch.

Easy series

The Easy series host has no termination resistor. An external termination resistor can be connected if necessary. The RS485 communication extension card of the GE20 series has a termination resistor. The termination resistor can be turned on or off by setting the DIP switch and the default setting is OFF. The following figure shows the RS485 bus topology.

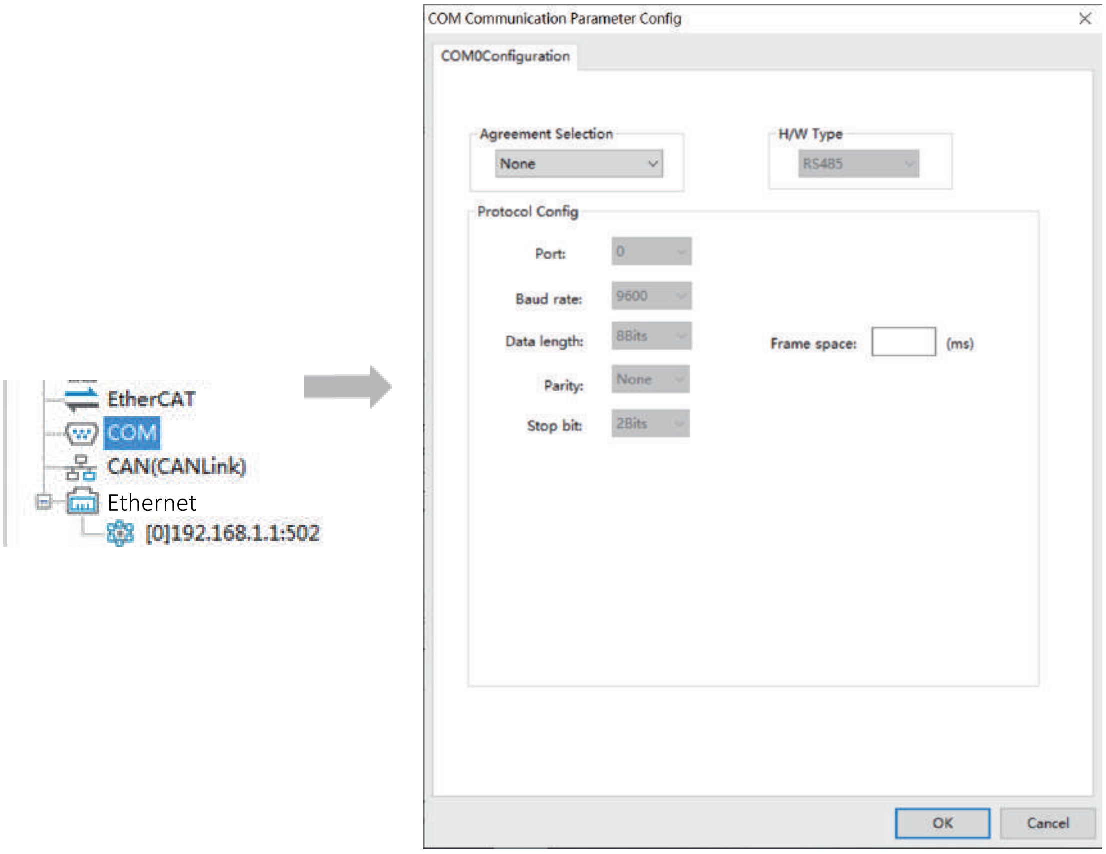




6.3 Free Protocol Configuration

6.3.1 Free Protocol Configuration

Double-click "COM". In the dialog box "COM Communication Parameter Config" displayed, select "Free Agreement", set serial port parameters, and then click "OK". Then, you can use the SerialRS instruction to send and receive data in the user program.



Agreement Selection: None

H/W Type: RS485

Protocol Config

Port: 0

Baud rate: 9600

Data length: 8Bits

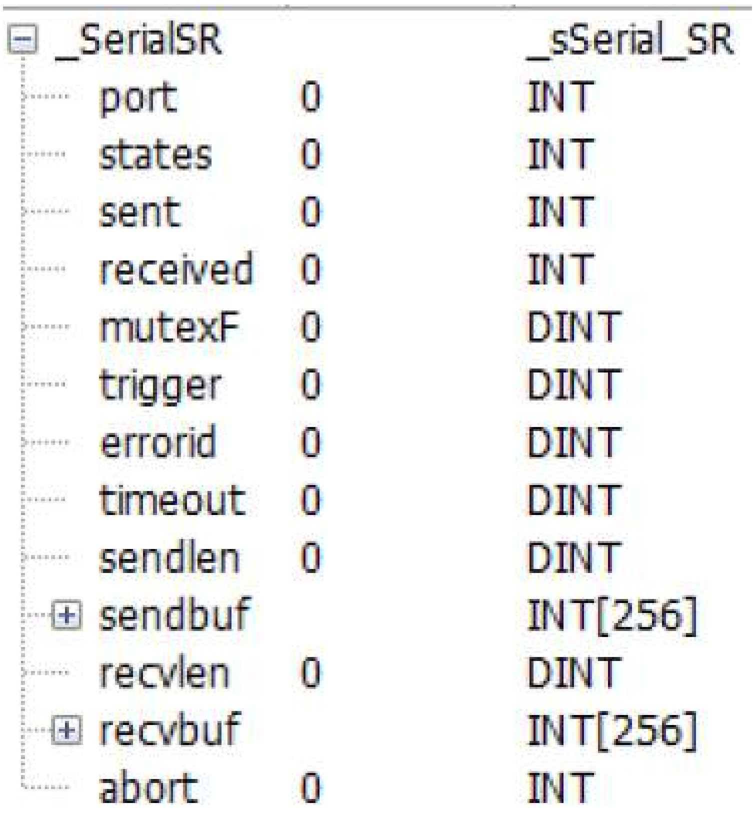
Parity: None

Stop bit: 2Bits

Frame space: (ms)

6.3.2 Free Protocol Cancellation (SerialSR Instruction)

When the free protocol is set for the COM port, you can use the SerialSR instruction to send and receive data over the free protocol, and set the system variable _SerialSR.abort to abort a free protocol send/ receive process. The modification takes effect immediately, as shown in the following figure.



⊟ _SerialSR		_sSerial_SR
port	0	INT
states	0	INT
sent	0	INT
received	0	INT
mutexF	0	DINT
trigger	0	DINT
errorid	0	DINT
timeout	0	DINT
sendlen	0	DINT
⊞ sendbuf		INT[256]
recvlen	0	DINT
⊞ recvbuf		INT[256]
abort	0	INT

Figure 6-1 _SerialSR structure

Usage:

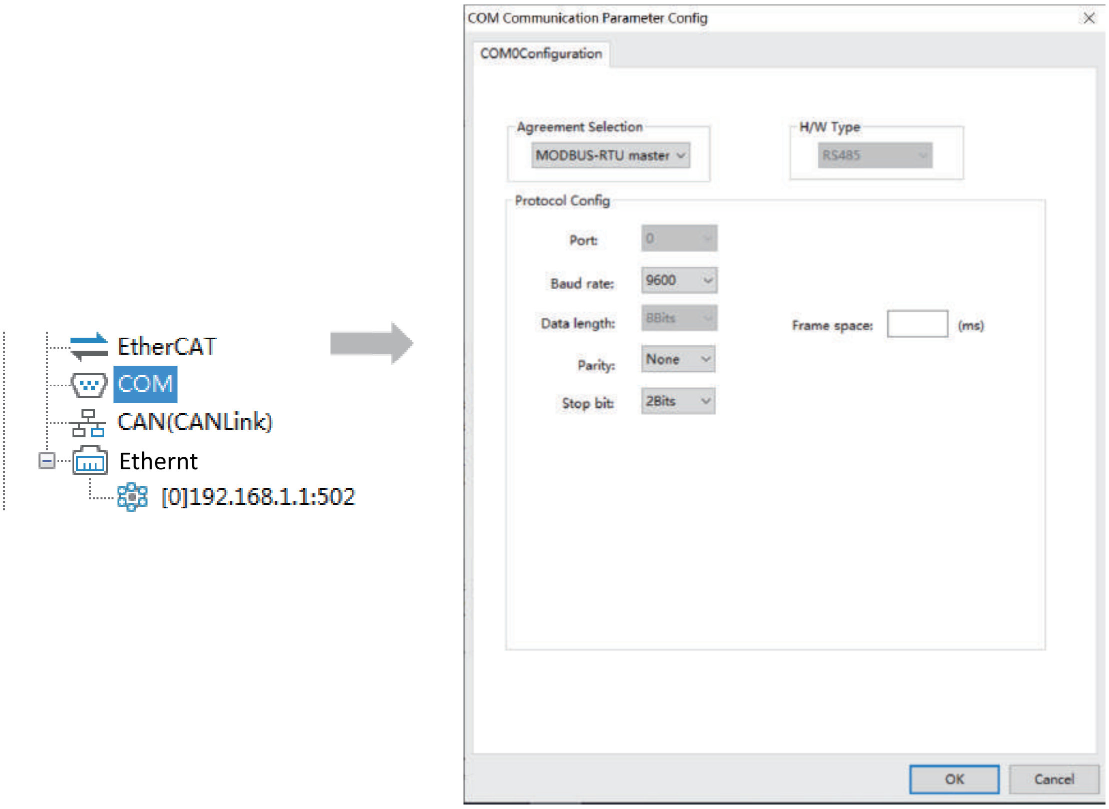
When _SerialSR.abort is set to a non-zero value, the send and receive processes can be aborted within the specified timeout period. After the processes are aborted, their states change to 16-Completed, and the DONE signal is ON.

6.4 Master Configuration

6.4.1 Modbus-RTU or Modbus-ASCII Master

Setting the serial port

Double-click "COM". In the dialog box "COM Communication Parameter Config" displayed, select "MODBUS-RTU master" or "MODBUS-ASC master", and set serial port parameters.



COM Communication Parameter Config

COM0Configuration

Agreement Selection: MODBUS-RTU master

H/W Type: RS485

Protocol Config

Port: 0

Baud rate: 9600

Data length: 8Bits

Parity: None

Stop bit: 2Bits

Frame space: (ms)

Adding the Modbus configuration

- Timeout: Sets the period for the master station to wait for the slave station to answer, in the unit of ms.
- Enabling control element: Enables or disables the connection. Customized variables are supported. If this option is not selected, the master station is enabled by default.

Accessing detailed configuration

Double-click "COM0 Modbus Config" to access the "Modbus Config" window. For detailed configuration, see *“6.4.2 Modbus Master Configuration Table” on page 239*

6.4.2 Modbus Master Configuration Table

Configure the Modbus master station.
The following describes relevant configuration items:

- Name

  The name that labels this condition configuration.

- Slave No.

  The number of the slave station you want to access. Up to 255 slave stations can be specified.

- Trigger mode and condition

  The communication modes are "Cycle" and "Trigger".

When "Cycle" is selected, the trigger condition is used to set the cycle time in ms. Then the configurations are executed according to the specified cycle.

Note

When the set cycle is smaller than the time required for communication, the configuration is executed according to the time required for communication. For example, if the set cycle is 10 ms and the actual slave response requires 20 ms, the actual execution cycle is 20 ms.

When "Trigger" is selected, the trigger condition is used to set the trigger condition variable/element. In this mode you can set the trigger condition to trigger a communication. If the slave station responds to the request, the trigger condition is automatically reset; otherwise, the trigger condition remains unchanged. If one trigger variable/element is used to trigger multiple configurations, the trigger condition will be automatically reset after all the triggered configurations are executed and the triggered configurations are not executed again.

- Function code

Function Code	Definition
0x01 (01)	Reads coils
0x02 (02)	Reads discrete inputs
0x03 (03)	Reads registers
0x04 (04)	Reads input registers
0x05 (05)	Single coil
0x06 (06)	Single register
0x0f (15)	Writes multiple coils
0x10 (16)	Writes multiple registers

- Slave register address
  The slave register address to be accessed.

  You can set the slave register address format to hexadecimal or decimal.

- Quantity
  The number of coils, discrete quantities, or registers to be accessed.

Function Code	Name	Max.
0x01 (01)	Reads coils	2000
0x02 (02)	Reads discrete inputs	2000
0x03 (03)	Reads registers	125
0x04 (04)	Reads input registers	125
0x05 (05)	Single coil	1
0x06 (06)	Single register	1
0x0f (15)	Writes multiple coils	1968
0x10 (16)	Writes multiple registers	123

- Mapped address
  The mapped address of the slave coil, discrete quantity, or register in the master station. Customized variables are supported.

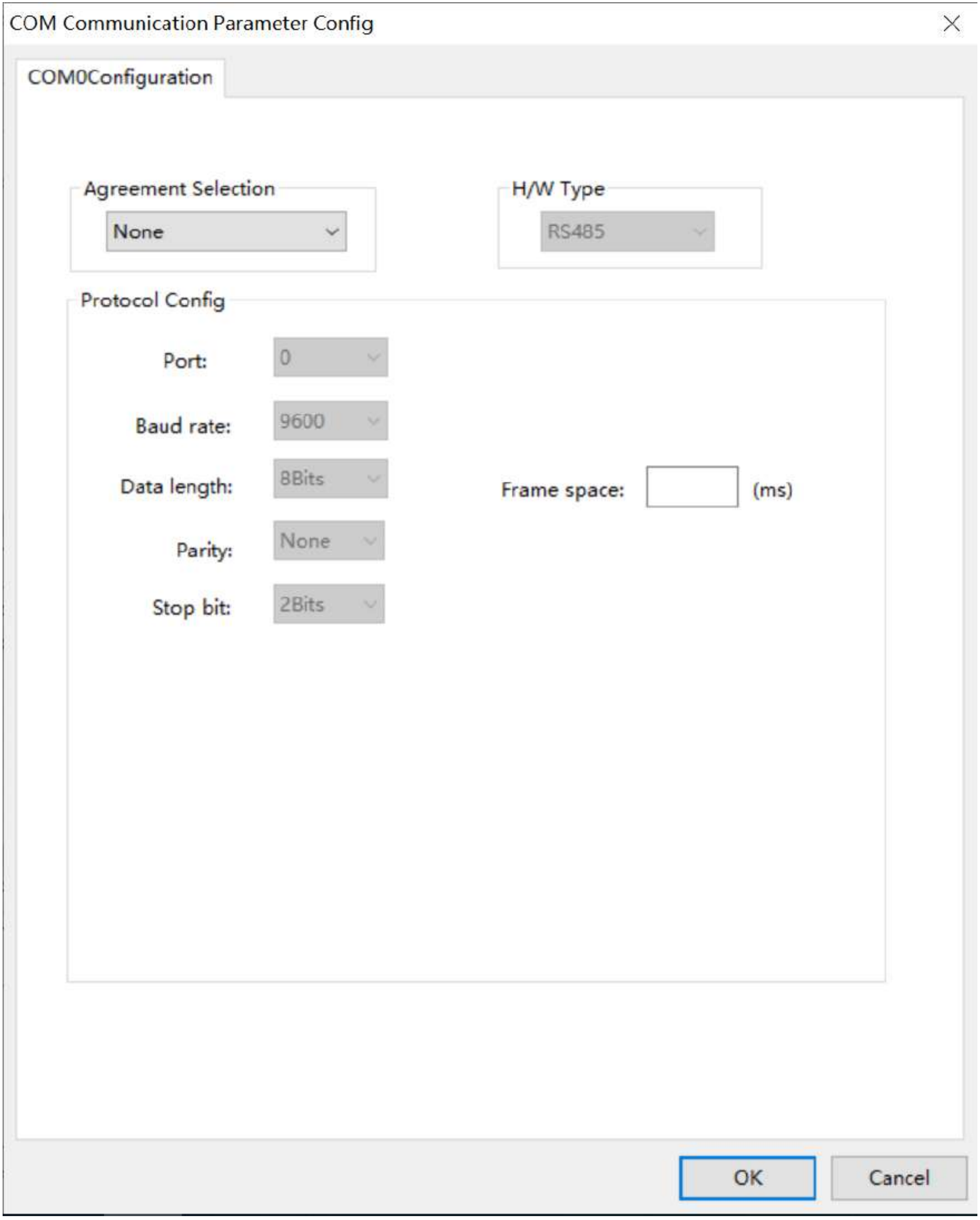
- Repeat number
  The number of retries after the slave station response times out.

6.4.3 Modbus-RTU Slave Disable

When the PLC serves as the Modbus-RTU master station, you can use system variables to disable a slave station.

Configuration

1. Set the PLC as the Modbus-RTU master station.



COM Communication Parameter Config

COM0Configuration

Agreement Selection: None

H/W Type: RS485

Protocol Config

Port: 0

Baud rate: 9600

Data length: 8Bits

Parity: None

Stop bit: 2Bits

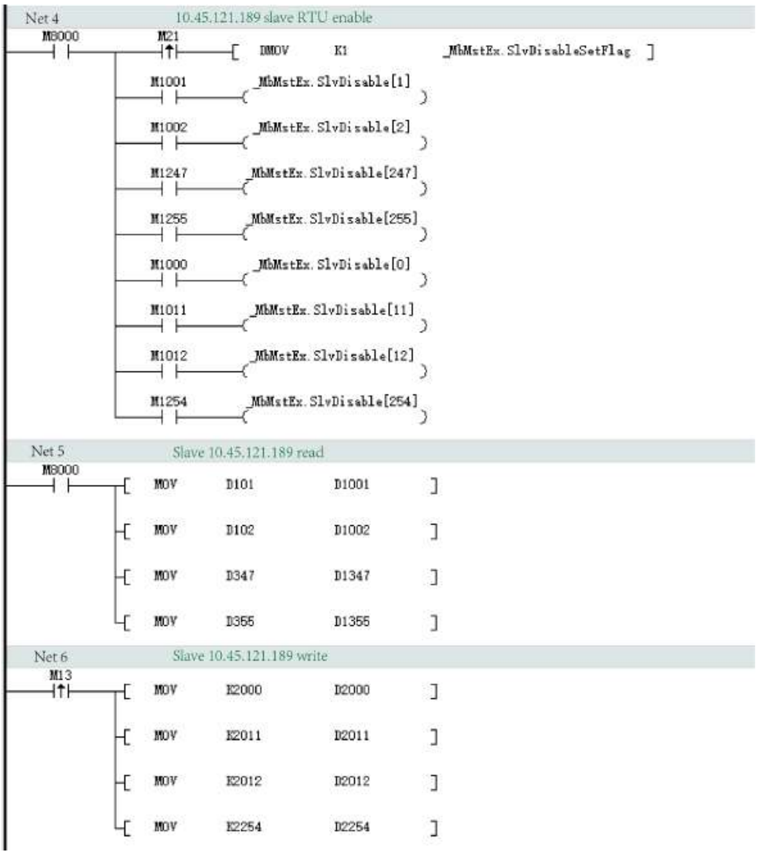
Frame space: (ms)

2. Add the configuration table.
3. Locate the slave station you want to disable based on the slave No. or slot No..
4. Configure the _MbMstEx structure array to disable slave stations under the corresponding host.

Note

- The configuration corresponding to COM[N] in the _MbMstEx [N] structure array is not retentive at power failure.
- _MbMstEx.SlvDisableSetFlag is used to enable or disable the Slave Disable function. When it is non-zero, the Slave Disable function is enabled.
- In _MbMstEx.SlvDisable[M], M is the Slave Disable flag corresponding to the slave station number mentioned in step 3. It is only valid when _MbMstEx.SlvDisableSetFlag is enabled.

Program example



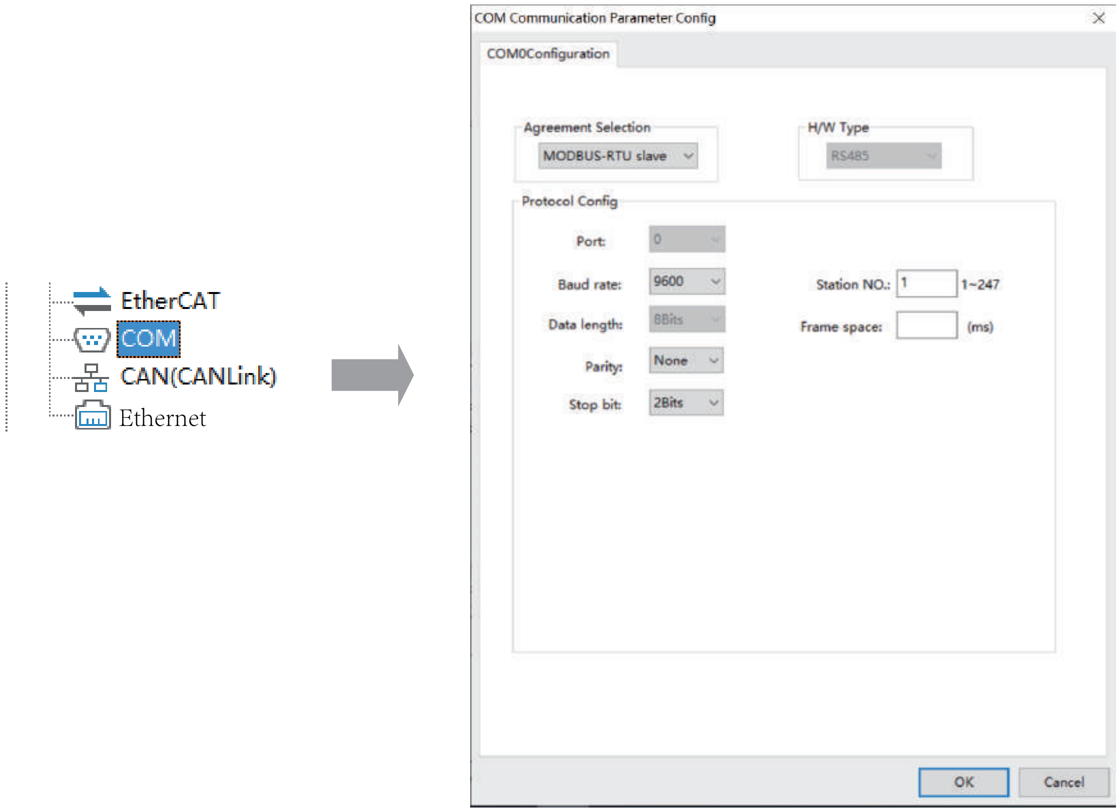
```
Net 4        10.45.121.189 slave RTU enable
M8000   M21
        ↑     [ DMOV   K1       _MbMstEx.SlvDisableSetFlag ]
        M1001 ( _MbMstEx.SlvDisable[1] )
        M1002 ( _MbMstEx.SlvDisable[2] )
        M1247 ( _MbMstEx.SlvDisable[247] )
        M1255 ( _MbMstEx.SlvDisable[255] )
        M1000 ( _MbMstEx.SlvDisable[0] )
        M1011 ( _MbMstEx.SlvDisable[11] )
        M1012 ( _MbMstEx.SlvDisable[12] )
        M1254 ( _MbMstEx.SlvDisable[254] )

Net 5        Slave 10.45.121.189 read
M8000   [ MOV   D101    D1001 ]
        [ MOV   D102    D1002 ]
        [ MOV   D347    D1347 ]
        [ MOV   D355    D1355 ]

Net 6        Slave 10.45.121.189 write
M13 ↑   [ MOV   K2000   D2000 ]
        [ MOV   K2011   D2011 ]
        [ MOV   K2012   D2012 ]
        [ MOV   K2254   D2254 ]
```

Note: When M1001, M1002, …, and M1254 are used to disable a single slave station, the read or write operation on this slave station is invalid.

6.5 Slave Configuration

6.5.1 Modbus-RTU or Modbus-ASCII Slave

Double-click "COM". In the dialog box "COM Communication Parameter Config" displayed, select "MODBUS-RTU slave" or "MODBUS-ASC slave", set serial port parameters and the slave No., and then click "OK". Download the project to H5U. For function codes and addresses supported when H5U is used as the Modbus slave station, see *"6.5.2 Parameters and Addresses" on page 243*.



COM Communication Parameter Config

COM0Configuration

Agreement Selection: MODBUS-RTU slave

H/W Type: RS485

Protocol Config

Port: 0

Baud rate: 9600

Data length: 8Bits

Parity: None

Stop bit: 2Bits

Station NO.: 1 1~247

Frame space: (ms)

6.5.2 Parameters and Addresses

- When H5U is used as the slave station, the following function codes are supported:

Function Code	Definition
0x01	Reads coils
0x02	Reads discrete quantities (same as 0x01).
0x03	Reads registers
0x04	Reads input registers (same as 0x03).
0x05	Writes a single coil.
0x06	Writes a single register.
0x0f	Writes multiple coils
0x10	Writes multiple registers
0x80 to 0xFF	Standard Modbus fault code

- When H5U is used as the slave station, addresses of coils that can be accessed by Modbus are listed in the following table:

Variable	Quantity	Address Range
M0-M7999	8000	0x0000 to 0x1F3F (0 to 7999)
B0-B32767	32768	0x3000 to 0xAFFF (12288 to 45055)
S0-S4095	4096	0xE000 to 0xEFFF (57344 to 61439)
X0-X1777 (octal)	1024	0xF800 to 0xFBFF (63488 to 64511)
Y0-Y1777 (octal)	1024	0xFC00-0xFFFF (64512 to 65535)

- When H5U is used as the slave station, addresses of registers that can be accessed by Modbus are listed in the following table:

Variable	Quantity	Start Address
D0-D7999	8000	0x0000 to 0x1F3F (0 to 7999)
R0-R32767	32768	0x3000 to 0xAFFF (12288 to 45055)

Note

W elements and Pointer variables are not supported.

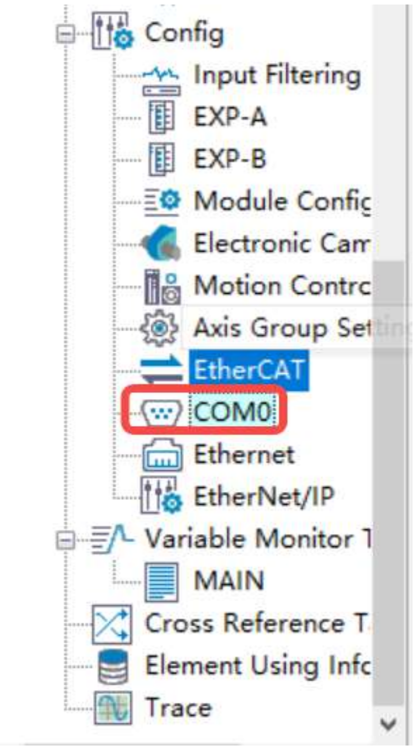
6.6 Example of Modbus-RTU Communication Application

Program requirements

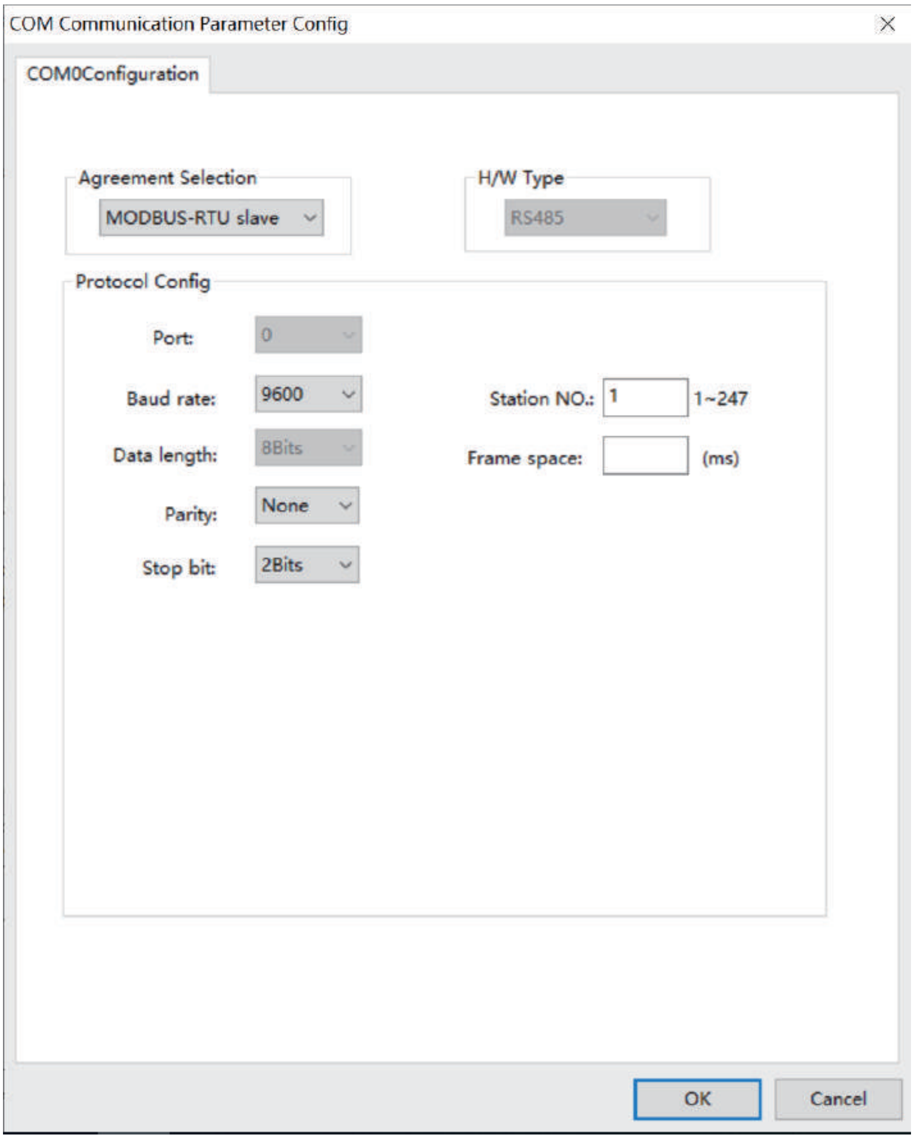
In this example, two H5Us are connected through a serial port and communicate with each other through the Modbus-RTU protocol. The master PLC reads the value in the D100 register of the slave PLC every 10 ms, and the value in D100 of the slave station is added by one every second.

Slave configuration

Double-click the COM icon to access the serial port configuration page.



In the window displayed, set the serial communication protocol and communication parameters. In this example, the protocol is Modbus-RTU and the communication parameter is 9600-8N2. Then, click "OK" to save the settings.



COM Communication Parameter Config

COM0Configuration

Agreement Selection: MODBUS-RTU slave

H/W Type: RS485

Protocol Config

Port: 0

Baud rate: 9600

Data length: 8Bits

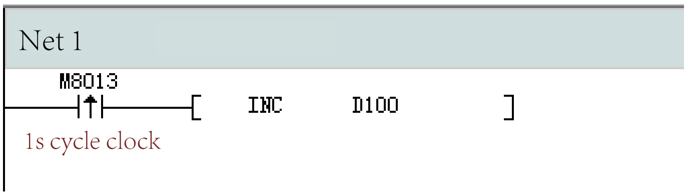
Parity: None

Stop bit: 2Bits

Station NO.: 1 1~247

Frame space: (ms)

Edit the program so that the value of D100 of the slave station is added by one every second.

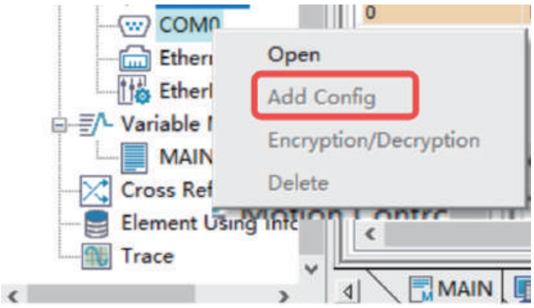


```
Net 1
M8013
--|↑|----[ INC    D100     ]
1s cycle clock
```

Then, click "Download" to download the program to the PLC.

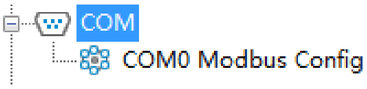
Master configuration

Double-click the COM icon. In the window displayed, set the communication protocol to Modbus-RTU master and set the communication parameter to that of the slave station. Right-click the COM icon. In the dialog box displayed, select "Add MODBUS Config".



You can set "Timeout" and "Enabling control element" in the dialog box displayed. In this example, the default settings are used: "Timeout" is set to 500 ms, and "Enabling control element" is deselected.

Click "OK". The master configuration is generated.

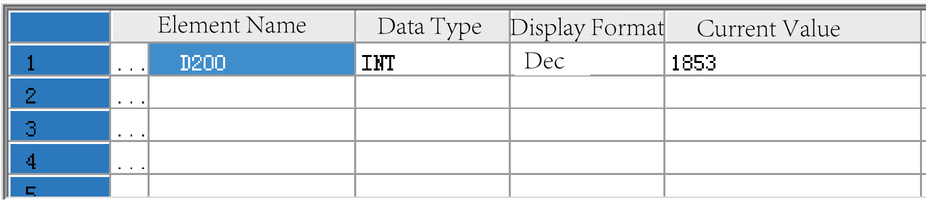


Double-click "COM0 Modbus Config". On the configuration table page displayed, click "Add" to add the configuration. In this example, the value of D100 of the slave station is stored in D200 of the master station.

Then, click "Download" to download the program to the PLC.

Effect

The value of the D100 register of the slave station can be read from the D200 register of the master station.

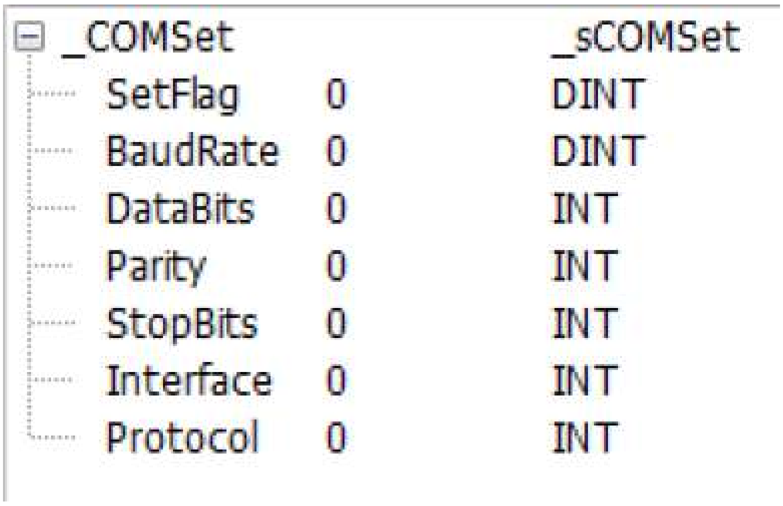


		Element Name	Data Type	Display Format	Current Value
1	...	D200	INT	Dec	1853
2	...				
3	...				
4	...				
5					

6.7 Modifying Serial Port Parameters

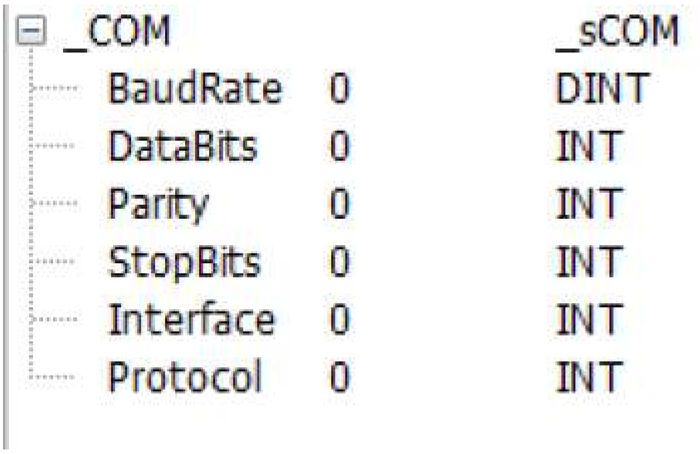
6.7.1 Modifying COM Port Parameters

When the free protocol and Modbus-RTU/Modbus-ASC master-slave are configured for the COM port, you can set the system variable _COMSet to modify the COM port parameters. The effective _COMSet parameter is synchronized to the _COM parameter, as shown in the following figure.



⊟ _COMSet		_sCOMSet
SetFlag	0	DINT
BaudRate	0	DINT
DataBits	0	INT
Parity	0	INT
StopBits	0	INT
Interface	0	INT
Protocol	0	INT

Figure 6-2 COMSet structure



⊟ _COM		_sCOM
BaudRate	0	DINT
DataBits	0	INT
Parity	0	INT
StopBits	0	INT
Interface	0	INT
Protocol	0	INT

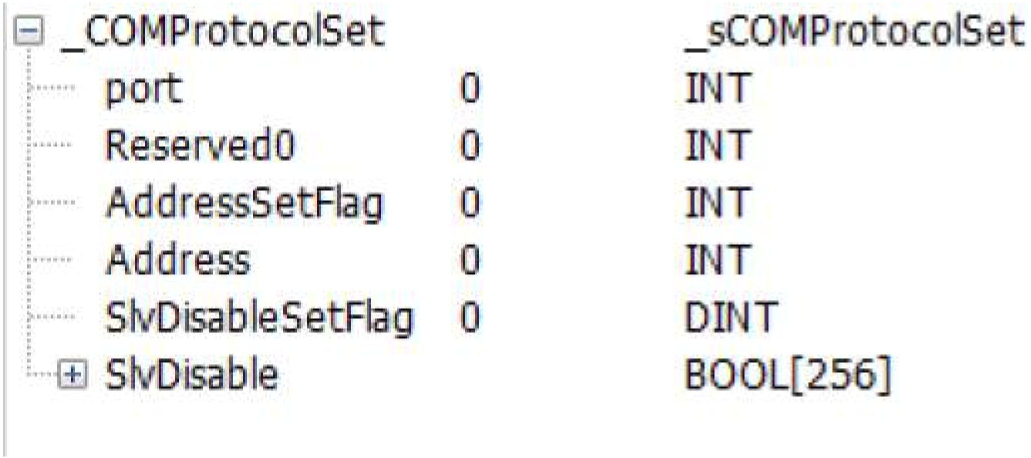
Figure 6-3 _COM structure

Usage:

1. When _COMSet.SetFlag is set to a non-zero value, parameters of the COM port can be modified online.
2. Only the baud rate, data bit, parity bit, and stop bit can be modified. The physical port and communication protocol parameters are read-only and cannot be modified.
3. Modification to these parameters takes effect only after STOP–RUN is executed on the PLC. The effective parameters can be viewed through the system variable _COM.
4. Note that when the baud rate, data bit, parity bit, or stop bit is set to an invalid value, the parameter will be reset by the system to the system background configuration parameter.
5. When the PLC is used as the Modbus-RTU or Modbus-ASC master station, the Modbus configuration table must be added; otherwise, modification to the parameter _COMSet is not synchronized to _COM.
6. After the COM parameters of the PLC are modified, parameters of the communication device connected to the port will be automatically synchronized.

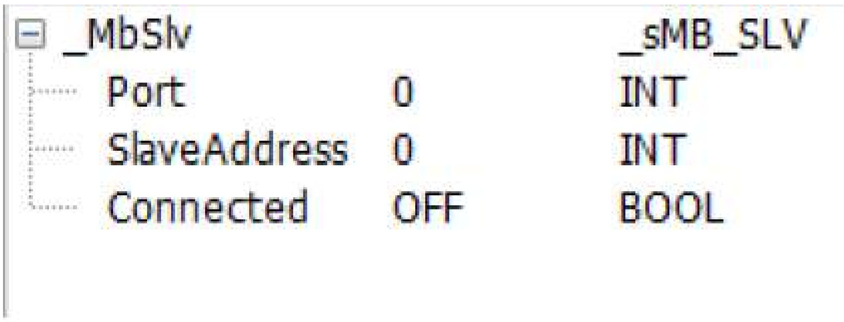
6.7.2 Modifying Slave Address Parameters

When Modbus-RTU or Modbus-ASC slave is set for the COM port, you can use the system variable _COMProtocolSet to modify the slave address parameter SlaveAddress of the COM port. After the modification takes effect, the parameter Address is synchronized to the SlaveAddress in the parameter _MbSlv, as shown in the following figure.



⊟ _COMProtocolSet		_sCOMProtocolSet
port	0	INT
Reserved0	0	INT
AddressSetFlag	0	INT
Address	0	INT
SlvDisableSetFlag	0	DINT
⊞ SlvDisable		BOOL[256]

Figure 6-4 _COMProtocolSet structure



⊟ _MbSlv		_sMB_SLV
Port	0	INT
SlaveAddress	0	INT
Connected	OFF	BOOL

Figure 6-5 _MbSlv structure

Usage:

1. When _COMProtocolSet.AddressSetFlag is set to a non-zero value, the parameter Address can be modified online.
2. _COMProtocolSet can be used to modify the parameter Address only.
3. Modification to these parameters takes effect only after STOP–RUN is executed on the PLC. The effective parameters can be viewed through the system variable _COM.

7 Ethernet Communication

7.1 Overview

H5U provides an Ethernet port. The Easy320 and Easy52X series provide two Ethernet ports. You can use AutoShop to monitor and commission the PLC and download and upload parameters from and to the PLC through Ethernet quickly and conveniently. You can also exchange data with other devices in the network through Ethernet.

The small PLC supports the Modbus-TCP protocol and includes a server and a client. It can communicate and exchange data with devices supporting Modbus-TCP. It provides socket instructions for devices not supporting Modbus-TCP to implement application protocols based on TCP/UDP. For details, see the section "Socket Instructions" in the instruction manual.

7.2 Hardware Ports

H5U provides a standard Ethernet port (one RJ45 port) and supports the Modbus-TCP Ethernet communication protocol.

The PLC hosts of the Easy320 and Easy52X series support Ethernet and the Modbus-TCP Ethernet communication protocol, as shown in the table below.

Series	Easy320	Easy521	Easy522	Easy523
Ethernet port	Supports two Ethernet ports, Modbus-TCP master-slave (when it is used as the client, a maximum of 31 servers are supported), TCP/IP, UDP, and EtherNet/IP master-slave. Supports a maximum of 32 slave stations and a minimum communication cycle of 5 ms.	Supports two Ethernet ports, Modbus-TCP master-slave (when it is used as the client, a maximum of 31 servers are supported), TCP/IP, UDP, and EtherNet/IP master-slave. Supports a maximum of 32 slave stations and a minimum communication cycle of 5 ms.		

RJ45 port specifications

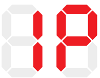
Item	Ethernet Port
Transmission rate	10 Mbps: 10BASE-T 100 Mbps: 100BASE-TX 10 Mbps/100 Mbps self-adaptive
Modulation	Baseband
Topology	Star
Medium	Cat5 twisted pairs or shielded twisted pairs with aluminum foil and braided mesh
Transmission distance	Distance between nodes: 100 m or less
Number of connections	31

7.3 IP Address Settings

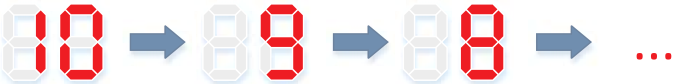
Restoring the default IP address for the H5U series PLC

The default IP address of H5U is 192.168.1.88. You can press the MFK key on the operating panel to restore the IP address to the default value as follows.

Switch the status of H5U to Stop, press and hold the MFK key until "IP" is displayed on the LED, and then press the MFK key for no more than 2s.



Then, a countdown is displayed on the LED. When the countdown reaches 0, the IP address is restored to the default value. You can press the MFK key during countdown to cancel the restoration operation.



Note: This IP address setting method is not supported by the Easy series.

Setting the IP address through a USB flash drive

For how to set the IP address through a USB flash drive, see the description in the section *"2.2.3 USB Connection" on page 35Direct connection through USB*.

Setting the IP address through Ethernet

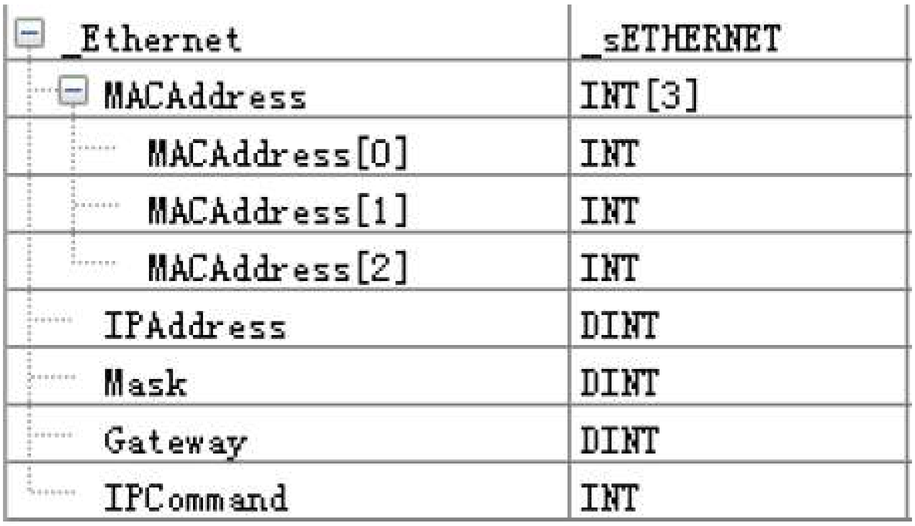
For how to set the IP address through Ethernet, see the description in the section *"2.2.2 Ethernet Connection" on page 29Connection through Ethernet*.

Setting the IP address through system variables

You can use the system variable to modify the IP address in the running state. For example, to set the variable Ethernet, perform the following operations.

1. Modify the value of the variable _Ethernet.IPCommand to 1.
2. Modify the variable _Ethernet.IPAddress to the target IP address. If the value of this variable is hexadecimal, such as 192.168.1.88, input C0A80158.
3. After inputting the IP address, modify the value of the variable _Ethernet.IPCommand to 2. Then, the value of this variable is automatically changed to 0 (display mode).

NoteThis variable can only modify the IP address and cannot modify other variables.

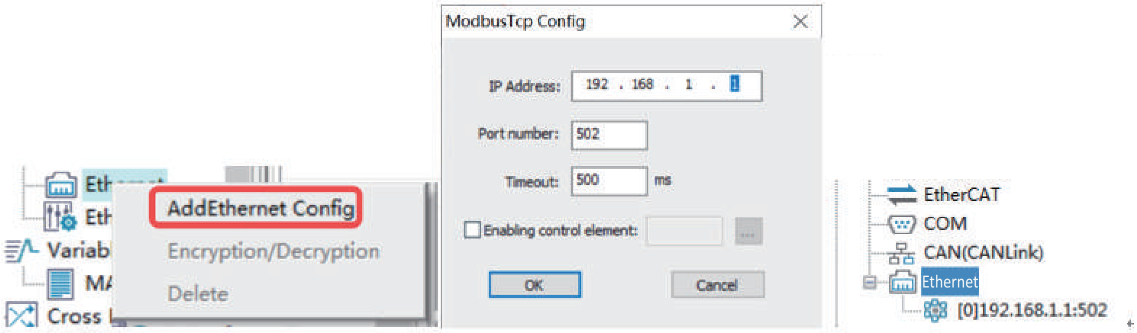


⊟ _Ethernet	_sETHERNET
⊟ MACAddress	INT[3]
MACAddress[0]	INT
MACAddress[1]	INT
MACAddress[2]	INT
IPAddress	DINT
Mask	DINT
Gateway	DINT
IPCommand	INT

7.4 Master Configuration

7.4.1 Modbus-TCP Master

The Modbus-TCP master station is the Modbus-TCP client, which can be configured through Modbus-TCP. It can communicate with a maximum of 31 Modbus-TCP servers (slave stations) at a time. Configure the Modbus-TCP master station as follows.



ModbusTcp Config

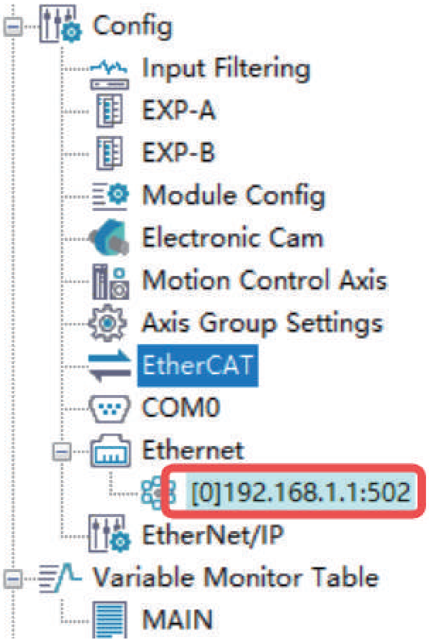
IP Address: 192 . 168 . 1 . 1

Port number: 502

Timeout: 500 ms

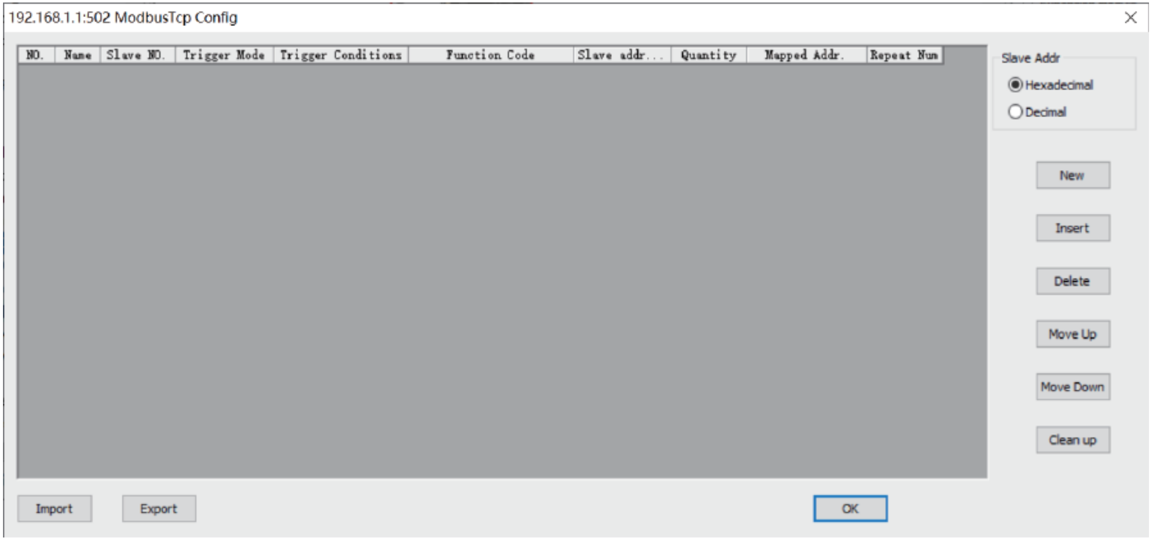
□ Enabling control element:

1. Set the IP address of the PLC. For details, see *"2.2.2 Ethernet Connection" on page 29Connection through Ethernet*.
2. Add a Modbus-TCP connection.
   - Timeout: Sets the period for the master station to wait for the slave station to answer, in the unit of ms.
   - Enabling control element: Enables or disables the connection. Customized variables are supported. If this option is not selected, the master station is enabled by default.
3. Access detailed configuration.
   Double-click the connected station to access the "Modbus Config" window. For details, see *"6.4.2 Modbus Master Configuration Table" on page 239Modbus Master Configuration Table*.



7.4.2 Modbus Master Configuration Table

Configure the Modbus master station on the following page.


192.168.1.1:502 ModbusTcp Config

NO.	Name	Slave NO.	Trigger Mode	Trigger Conditions	Function Code	Slave addr...	Quantity	Mapped Addr.	Repeat Num

Slave Addr
● Hexadecimal
○ Decimal

The following describes relevant configuration items:

- Name
  The name that labels this condition configuration.

- Slave No.
  The number of the slave station you want to access. Up to 255 slave stations can be specified.

  Modbus-TCP communication identifies slave stations by IP address. The slave No. is not checked. You can use the default slave No..

- Trigger mode and condition
  The communication modes are "Cycle" and "Trigger".

When "Cycle" is selected, the trigger condition is used to set the cycle time in ms. Then the configurations are executed according to the specified cycle.

Note

When the set cycle is smaller than the time required for communication, the configuration is executed according to the time required for communication. For example, if the set cycle is 10 ms and the actual slave response requires 20 ms, the actual execution cycle is 20 ms.

When "Trigger" is selected, the trigger condition is used to set the trigger condition variable/element. In this mode you can set the trigger condition to trigger a communication. If the slave station responds to the request, the trigger condition is automatically reset; otherwise, the trigger condition remains unchanged. If one trigger variable/element is used to trigger multiple configurations, the trigger condition will be automatically reset after all the triggered configurations are executed and the triggered configurations are not executed again.

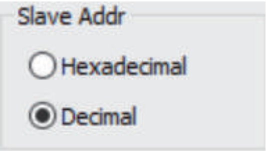
- Function code

Function Code	Definition
0x01 (01)	Reads coils
0x02 (02)	Reads discrete inputs
0x03 (03)	Reads registers
0x04 (04)	Reads input registers
0x05 (05)	Single coil
0x06 (06)	Single register
0x0f (15)	Writes multiple coils
0x10 (16)	Writes multiple registers

- Slave register address

  The slave register address to be accessed.

  You can set the slave register address format to hexadecimal or decimal.



Slave Addr

○ Hexadecimal

● Decimal

- Quantity

  The number of coils, discrete quantities, or registers to be accessed.

Function Code	Name	Max.
0x01 (01)	Reads coils	2000
0x02 (02)	Reads discrete inputs	2000
0x03 (03)	Reads registers	125
0x04 (04)	Reads input registers	125
0x05 (05)	Single coil	1
0x06 (06)	Single register	1
0x0f (15)	Writes multiple coils	1968
0x10 (16)	Writes multiple registers	123

- Mapped address

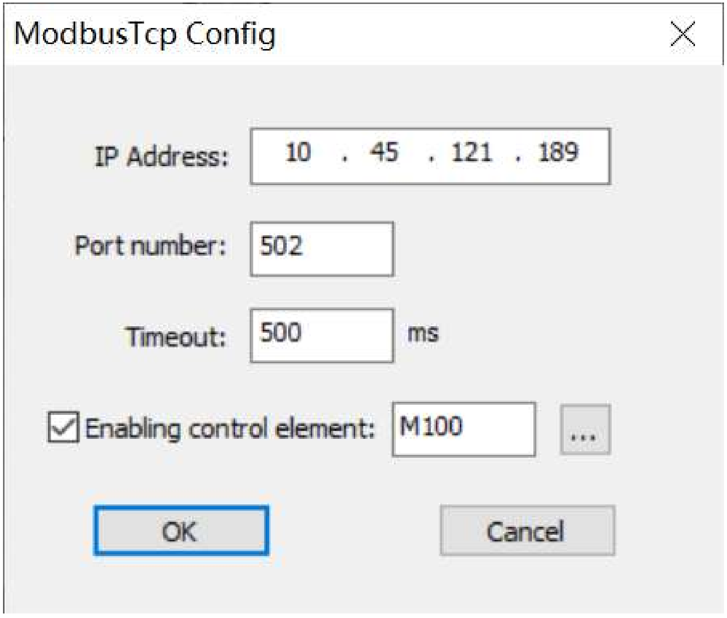
The mapped address of the slave coil, discrete quantity, or register in the master station. Customized variables are supported.

- Repeat number
  The number of retries after the slave station response times out.

7.4.3 Modbus-TCP Slave Disable

Configuration

1. Configure the PLC slave connection. Create three server connections numbered 0, 1, and 2, as shown in the following figure.



ModbusTcp Config

IP Address: 10 . 45 . 121 . 189

Port number: 502

Timeout: 500 ms

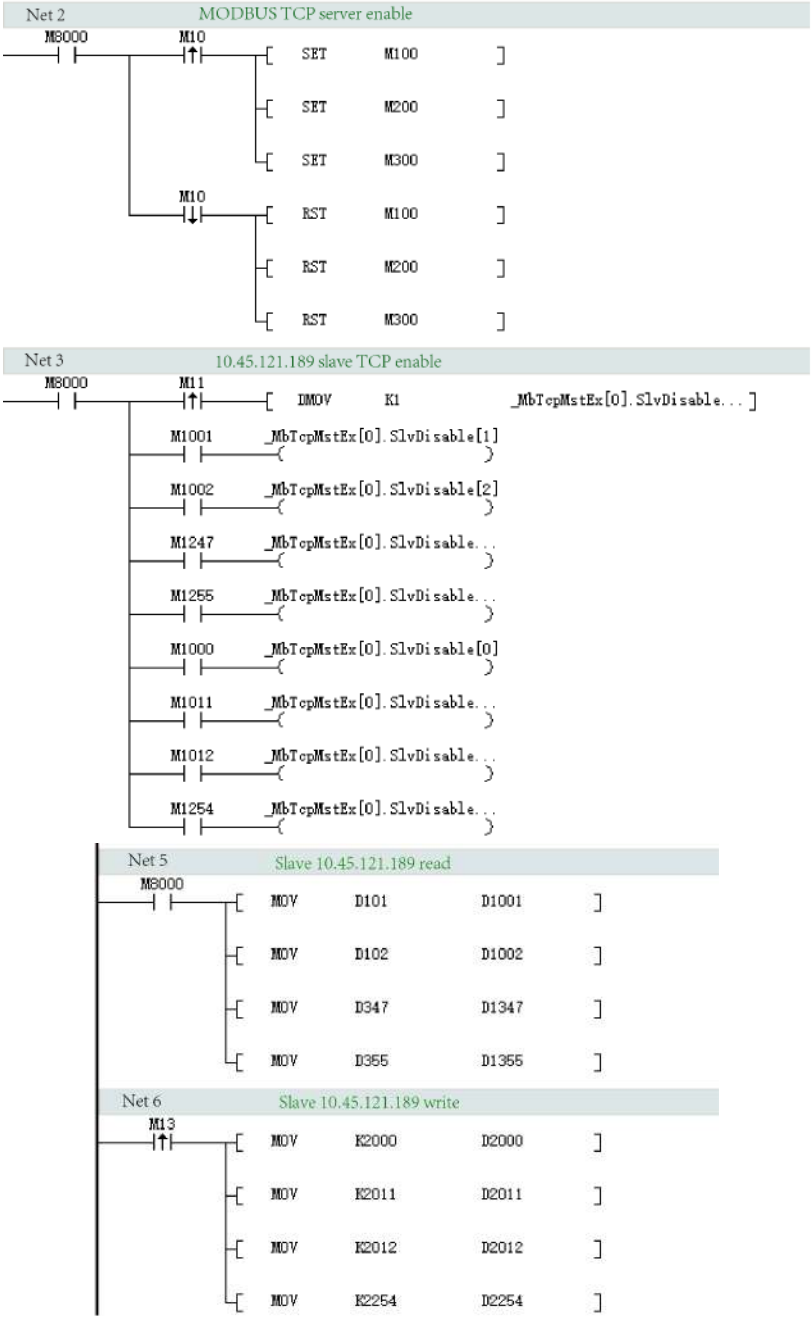
☑ Enabling control element: M100

2. Add the configuration table.
3. Locate the slave station you want to disable based on the slave No. or slot No..
4. Configure the _MbTcpMstEx structure array to disable slave stations under the corresponding host.

Note

- In the _MbTcpMstEx[N] structure array, N is the server number. In this example, N is 0.
- _MbTcpMstEx[N].SlvDisableSetFlag is used to enable or disable the Slave Disable function. When it is non-zero, the Slave Disable function is enabled.
- In _MbTcpMstEx[N].SlvDisable[M], M is the Slave Disable flag corresponding to the slave station number mentioned in step 3. It is only valid when _MbTcpMstEx[N].SlvDisableSetFlag is enabled.

Program example


Net 2
MODBUS TCP server enable
M8000
M10
SET M100
SET M200
SET M300
M10
RST M100
RST M200
RST M300
Net 3
10.45.121.189 slave TCP enable
M8000
M11
DMOV K1 _MbTcpMstEx[0].SlvDisable...
M1001 _MbTcpMstEx[0].SlvDisable[1]
M1002 _MbTcpMstEx[0].SlvDisable[2]
M1247 _MbTcpMstEx[0].SlvDisable...
M1255 _MbTcpMstEx[0].SlvDisable...
M1000 _MbTcpMstEx[0].SlvDisable[0]
M1011 _MbTcpMstEx[0].SlvDisable...
M1012 _MbTcpMstEx[0].SlvDisable...
M1254 _MbTcpMstEx[0].SlvDisable...
Net 5
Slave 10.45.121.189 read
M8000
MOV D101 D1001
MOV D102 D1002
MOV D347 D1347
MOV D355 D1355
Net 6
Slave 10.45.121.189 write
M13
MOV K2000 D2000
MOV K2011 D2011
MOV K2012 D2012
MOV K2254 D2254

Note: When M1001, M1002, …, and M1254 are used to disable a single slave station, the read or write operation on this slave station is invalid.

7.5 Slave Configuration References

7.5.1 Modbus-TCP Slave

A Modbus-TCP slave station is a Modbus-TCP server enabled with Modbus-TCP and port 502 by default.

One H5U can connect to a maximum of 16 Modbus-TCP clients (master stations) at a time. When serving as the client, the Easy320 and Easy52X series can connect to a maximum of 31 Modbus-TCP servers. Configure the slave station as follows:

1. Set the IP address. Then, the Modbus-TCP slave function is enabled and you do not need to set the communication protocol.
2. Configure the Modbus-TCP master station and create a connection. Then, you can use the IP address of the PLC to communicate with H5U through the port 502.

7.5.2 Parameters and Addresses

- When H5U is used as the slave station, the following function codes are supported:

Function Code	Definition
0x01	Reads coils
0x02	Reads discrete quantities (same as 0x01).
0x03	Reads registers
0x04	Reads input registers (same as 0x03).
0x05	Writes a single coil.
0x06	Writes a single register.
0x0f	Writes multiple coils
0x10	Writes multiple registers
0x80 to 0xFF	Standard Modbus fault code

- When H5U is used as the slave station, addresses of coils that can be accessed by Modbus are listed in the following table:

Variable	Quantity	Address Range
M0-M7999	8000	0x0000 to 0x1F3F (0 to 7999)
B0-B32767	32768	0x3000 to 0xAFFF (12288 to 45055)
S0-S4095	4096	0xE000 to 0xEFFF (57344 to 61439)
X0-X1777 (octal)	1024	0xF800 to 0xFBFF (63488 to 64511)
Y0-Y1777 (octal)	1024	0xFC00-0xFFFF (64512 to 65535)

- When H5U is used as the slave station, addresses of registers that can be accessed by Modbus are listed in the following table:

Variable	Quantity	Start Address
D0-D7999	8000	0x0000 to 0x1F3F (0 to 7999)
R0-R32767	32768	0x3000 to 0xAFFF (12288 to 45055)

Note

W elements and Pointer variables are not supported.

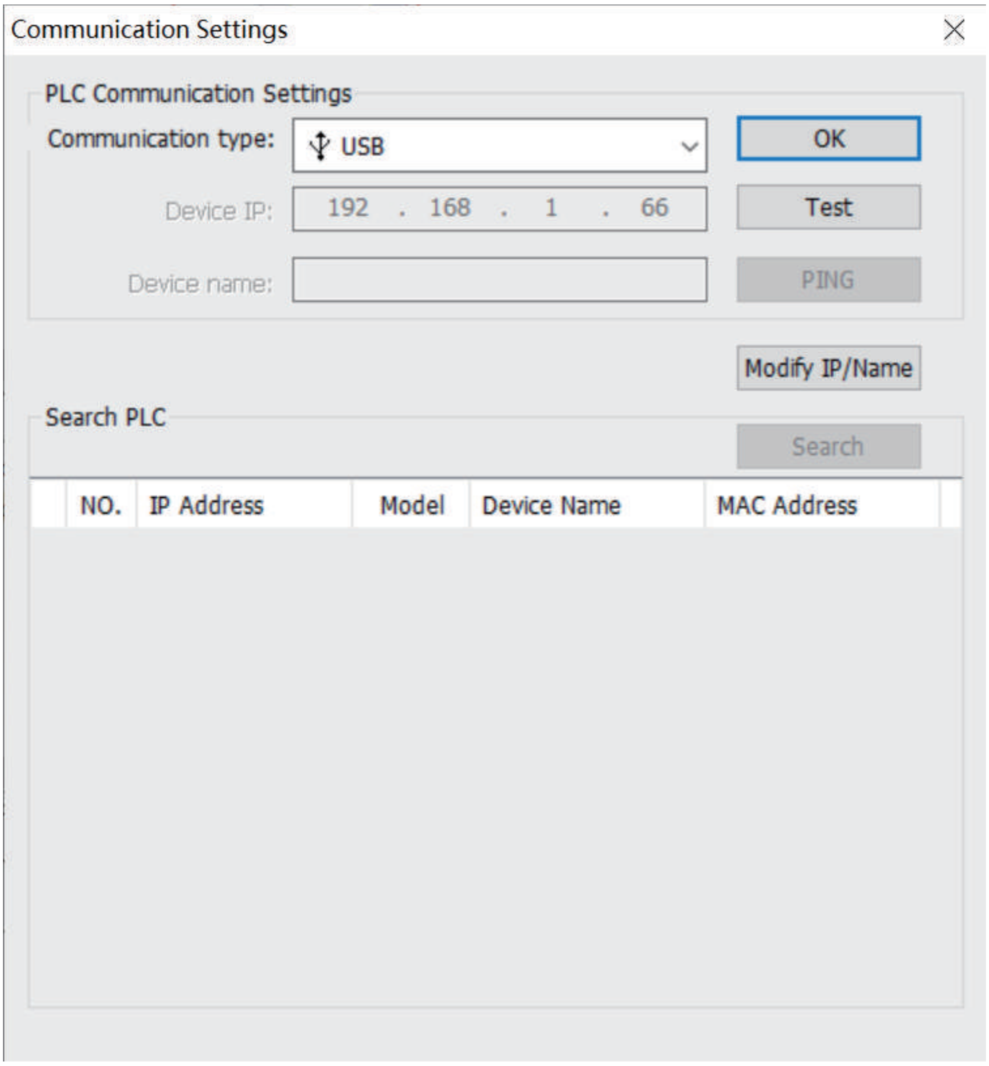
7.6 Example of Modbus-TCP Communication Application

Program requirements

In this example, two H5Us are connected through an Ethernet port and communicate with each other through the Modbus-TCP protocol. The master PLC reads the value in the D100 register of the slave PLC every 10 ms, and the value in D100 of the slave station is added by one every second.

1. Slave configuration

Click the test communication status button TEST. On the "Communication Settings" page displayed, click "Modify IP/Name".



Communication Settings

PLC Communication Settings

Communication type: USB

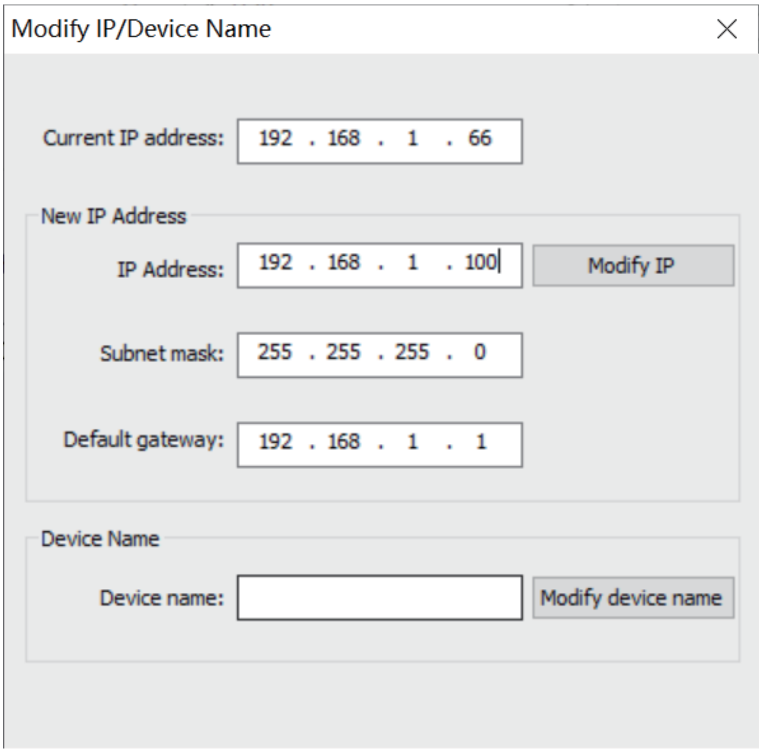
Device IP: 192 . 168 . 1 . 66

Device name:

Search PLC

NO.	IP Address	Model	Device Name	MAC Address

In the window displayed, set the slave IP address, subnet mask, and gateway, and then click "Modify IP" to modify the IP address. In this example, the IP address, subnet mask, and default gateway are set to 192.168.1.100, 255.255.255.0, and 192.168.1.1, respectively.


Modify IP/Device Name

Current IP address: 192 . 168 . 1 . 66

New IP Address

IP Address: 192 . 168 . 1 . 100

Subnet mask: 255 . 255 . 255 . 0

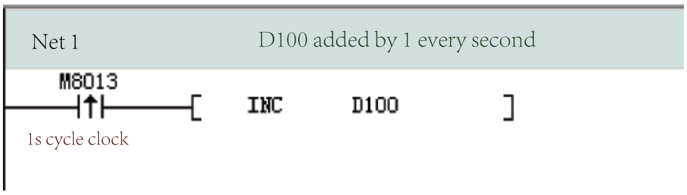
Default gateway: 192 . 168 . 1 . 1

Device Name

Device name:

When the operation is correct, the system prompts "IP modification successful!". Click "OK" to close the dialog box.

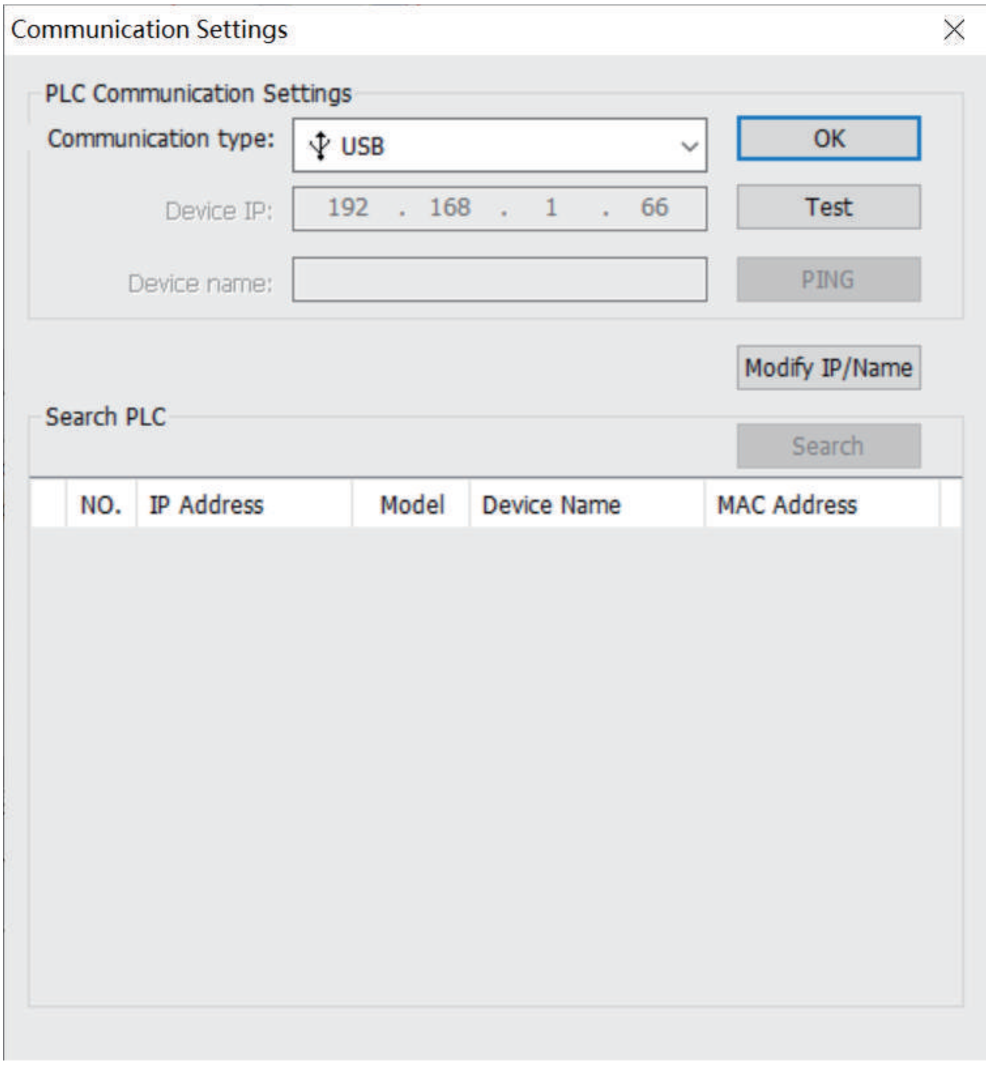
Edit the program so that the value of D100 of the slave station is added by one every second.



Then, click ⤓ to download the program to the PLC.

2. Master configuration

Click the test communication status button TEST. On the "Communication Settings" page displayed, click "Modify IP/Name".



Communication Settings

PLC Communication Settings

Communication type: USB

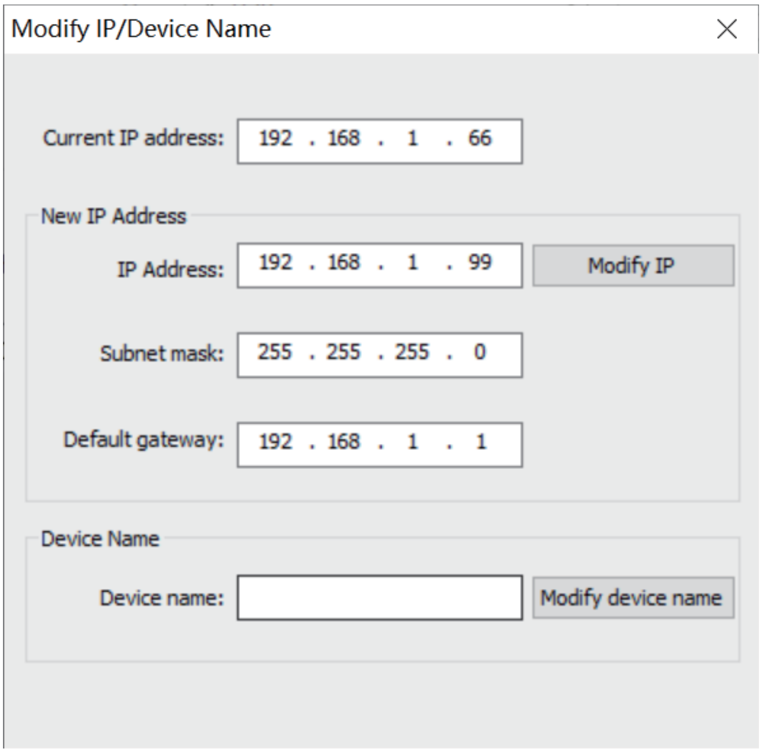
Device IP: 192 . 168 . 1 . 66

Device name:

Search PLC

NO.	IP Address	Model	Device Name	MAC Address

In the window displayed, set the slave IP address, subnet mask, and gateway, and then click "Modify IP" to modify the IP address. In this example, the IP address, subnet mask, and default gateway are set to 192.168.1.99, 255.255.255.0, and 192.168.1.1, respectively.



Modify IP/Device Name

Current IP address: 192 . 168 . 1 . 66

New IP Address

IP Address: 192 . 168 . 1 . 99

Subnet mask: 255 . 255 . 255 . 0

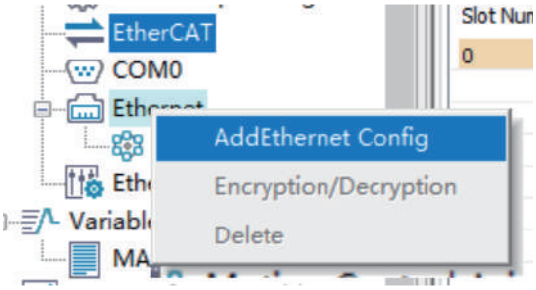
Default gateway: 192 . 168 . 1 . 1

Device Name

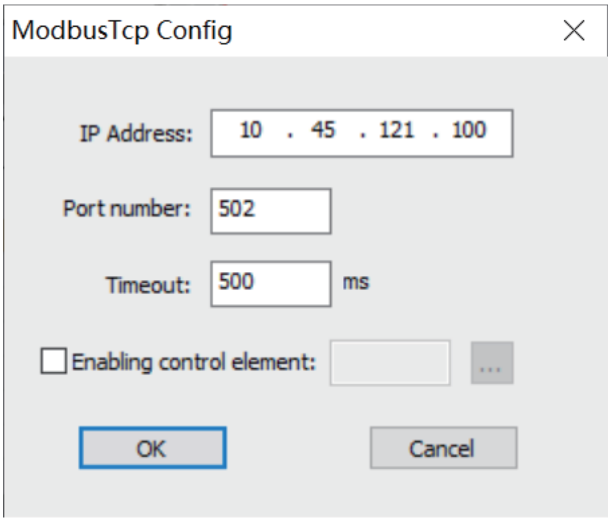
Device name:

When the operation is correct, the system prompts "IP modification successful!". Click "OK" to close the dialog box.

Right-click the Ethernet icon and select "Add Ethernet configuration".



In the dialog box displayed, set "IP Address" to 192.168.1.100 and "Timeout" to 500 ms, deselect "Enabling control element", and use default values for other configurations.



ModbusTcp Config

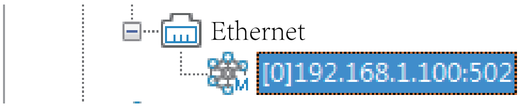
IP Address: 10 . 45 . 121 . 100

Port number: 502

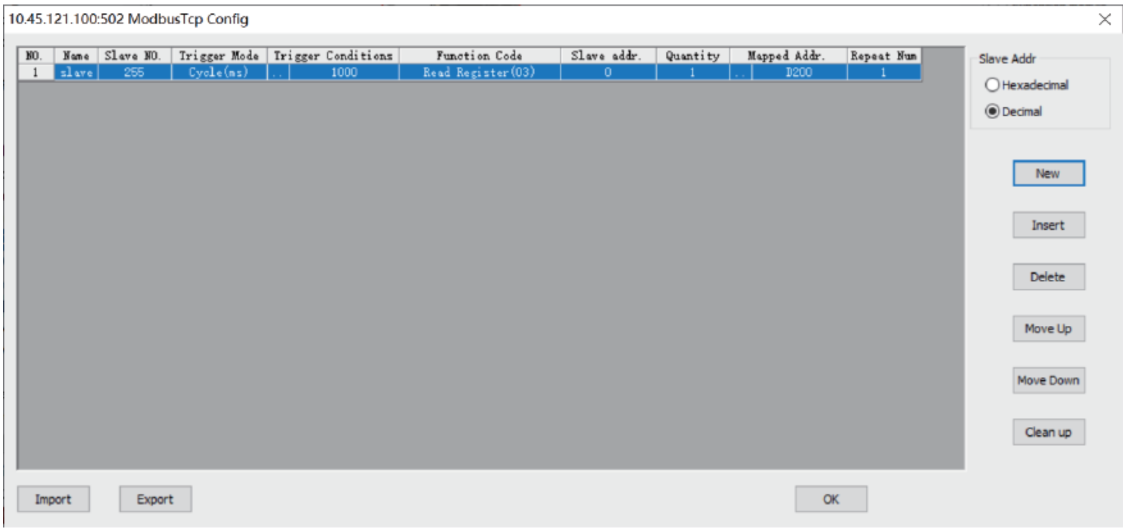
Timeout: 500 ms

□ Enabling control element:

Click "OK". The master configuration is generated.



Double-click the master configuration. On the configuration table page displayed, click "Add" to add the configuration. In this example, the value of D100 of the slave station is stored in D200 of the master station.


10.45.121.100:502 ModbusTcp Config

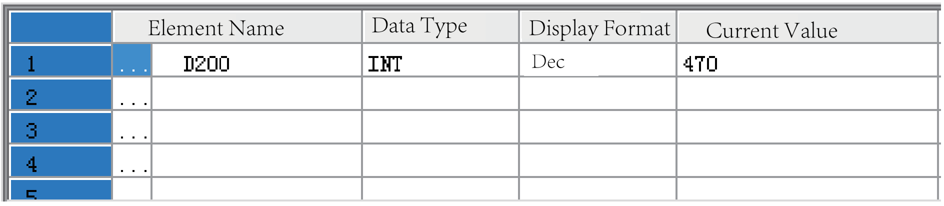
NO.	Name	Slave NO.	Trigger Mode	Trigger Conditions	Function Code	Slave addr.	Quantity	Mapped Addr.	Repeat Num
1	slave	255	Cycle(ms)	1000	Read Register(03)	0	1	D200	1

Slave Addr
○ Hexadecimal
● Decimal

Then, click to download the program to the PLC.

3. Effect

The value of the D100 register of the slave station can be read from the D200 register of the master station.



		Element Name	Data Type	Display Format	Current Value
1	...	D200	INT	Dec	470
2	...				
3	...				
4	...				
5					

8 CAN Communication

8.1 Overview

- H5U provides a CAN communication port that supports the CANlink and CANopen protocols and can be scaled up to 63 slave stations.
- The Easy series (Easy302/Easy320/Easy501/Easy502/Easy521/Easy522/Easy523) can support one master station and up to 63 slave stations by using the extension cards. They support the CANlink and CANopen protocols.

Note

This function requires a firmware version of V5.65.2.0 or later and a software version of V4.6.5.0 or later for the Easy302, Easy320, Easy501, and Easy502 series. It requires a firmware version of V5.66.0.0 or later and a software version of V4.8.0.0 or later for the Easy521, Easy522, and Easy523 series.

8.2 Hardware Ports

The CAN communication port and RS485 port of H5U are integrated to a 6-pin port. CAN extension cards of the Easy series use the RJ45 network port. For details, see the GE20-CAN-485 Communication Extension Card User Guide.

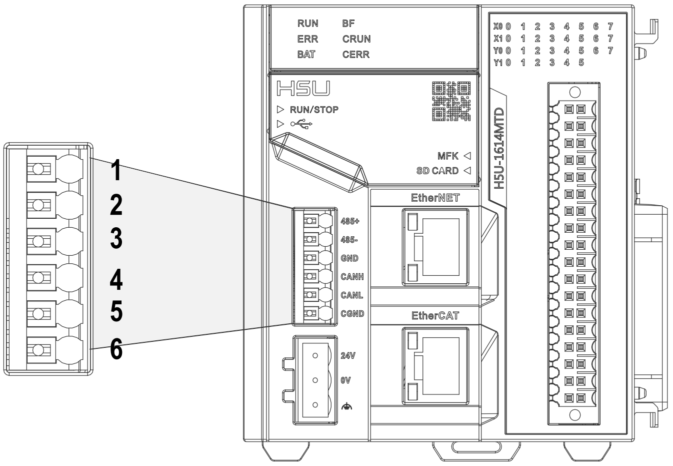


Table 8–1 Port pins

Pin	Signal Definition	Description
1	485+	Positive signal of the RS485 differential pair for COM0
2	485–	Negative signal of the RS485 differential pair for COM0
3	GND	Power ground of COM0
4	CANH	CAN communication data receiving terminal
5	CANL	CAN communication data sending terminal
6	CGND	CAN communication ground

8.3 CAN Network

8.3.1 CAN Communication Networking

The three wires of each device must be interconnected to form a CAN. 120 Ω termination resistors must be provided at both sides of the CAN bus (both the H5U and CAN extension card have a built-in resistor, which can be connected by setting the DIP switch. The default value is ON).

The CAN bus wiring diagram of H5U is as follows.

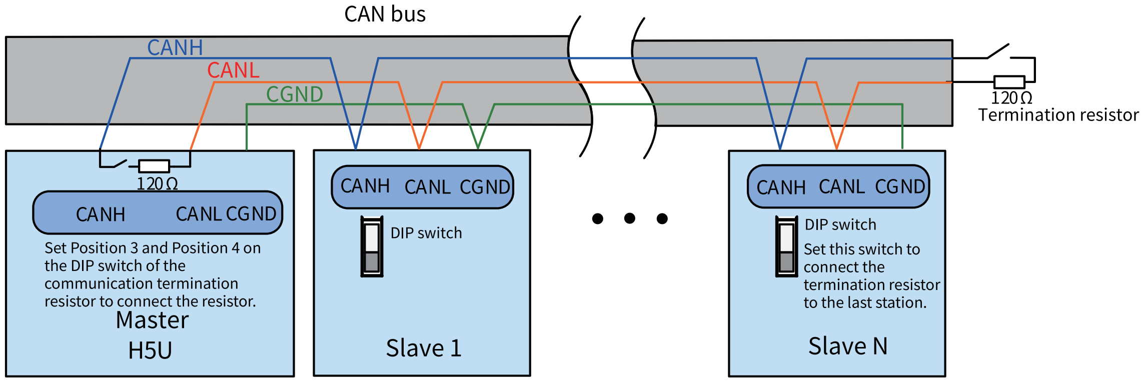

CAN bus
CANH
CANL
CGND
120 Ω
Termination resistor
120 Ω
CANH CANL CGND
Set Position 3 and Position 4 on the DIP switch of the communication termination resistor to connect the resistor.
Master
H5U
CANH CANL CGND
DIP switch
Slave 1
CANH CANL CGND
DIP switch
Set this switch to connect the termination resistor to the last station.
Slave N

Figure 8-1 Wiring diagram of a CAN network formed by multiple devices

Note

The CGND terminals of all the devices must be connected together.

8.3.2 Relationship Between CAN Communication Distance and Baud Rate

The following table lists relationship between CAN communication distances and baud rates

Baud Rate (kbps)	Distance (m)	Min. Cross-Sectional Area (mm^2)	Max. Number of Access Points
1000	20	0.3	18
500	80	0.3	31
250	150	0.3	31
125	300	0.5	31
100	500	0.5	31
50	1000	0.7	31

8.3.3 CAN Port System Variables

H5U provides a system variable named "_CAN" to view or monitor the status of the CAN port. "_CAN" is a structure variable whose data type is "_sCAN". The following table lists its members and definitions.

Member	Data Type	Description
BaudRate	INT	The baud rate, in the unit of kbps
LoadRate	INT	The network load rate, in the unit of %
RxPexSec	INT	The number of messages received per second, in the unit of FPS
TxPexSec	INT	The number of messages sent per second, in the unit of FPS

Member	Data Type	Description
RxErrCnt	INT	The count of errors received by the CAN controller
TxErrCnt	INT	The count of errors sent by the CAN controller
Protocol	INT	The communication protocol. 0: CANlink; 1: CANopen

8.4 CANlink Communication

8.4.1 CANlink3.0 Communication Principles

CANlink3.0 communication is implemented through CAN network configuration rather than CAN communication instructions. When downloading user programs, you need to download CAN network configurations to the PLC.

Understanding the principle of CANlink3.0 network configuration can help you complete the CAN configuration table.

One CANlink3.0 network can have only one master station but can have one or more slave stations.

Master and slave stations on a CANlink3.0 network communicate with each other by automatically sending and writing data rather than in query-response mode.

Example:

- To send data to slave stations, the master station "writes" register data in slave registers based on CANlink communication configurations when trigger conditions are met.
- Slave stations automatically send data to the master station and "write" the data in the receiving unit of the master station based on CANlink communication configurations.
- Slave stations automatically send data to each other and "write" the data in receiving units of slave stations based on CANlink communication configurations.
- To send data to multiple stations, a station automatically sends the "write operation" data to itself (equivalent to broadcasts), while the other stations selectively receive the data and automatically store it in their receiving units.
- For efficient data exchange during network communication, master and slave stations save "heard" broadcast data sent by other stations. You need to click "Receive Config" to set receiving slave station numbers and addresses. In this way, the stations configured as receiving stations will ignore the broadcast data from stations not configured as sending stations.

You do not need to configure CANlink3.0 slave stations because CANlink configurations can be transmitted to slave stations through a master PLC. Therefore, CANlink3.0 communication configuration items for slave stations are forwarded by the CANlink master station through configuration frames.

Upon startup, the master station sends configuration frames to CANlink slave stations and assigns the list of communication tasks. Slave stations automatically send data based on the list.

CANlink3.0 configuration items include the address of the sent register, address of the target receiving slave station, number of data entries, address of the received register, interval for sending, and trigger condition, which are required by common communication instructions. Different from common communication operations, "communicate-write" operations do not need responses.

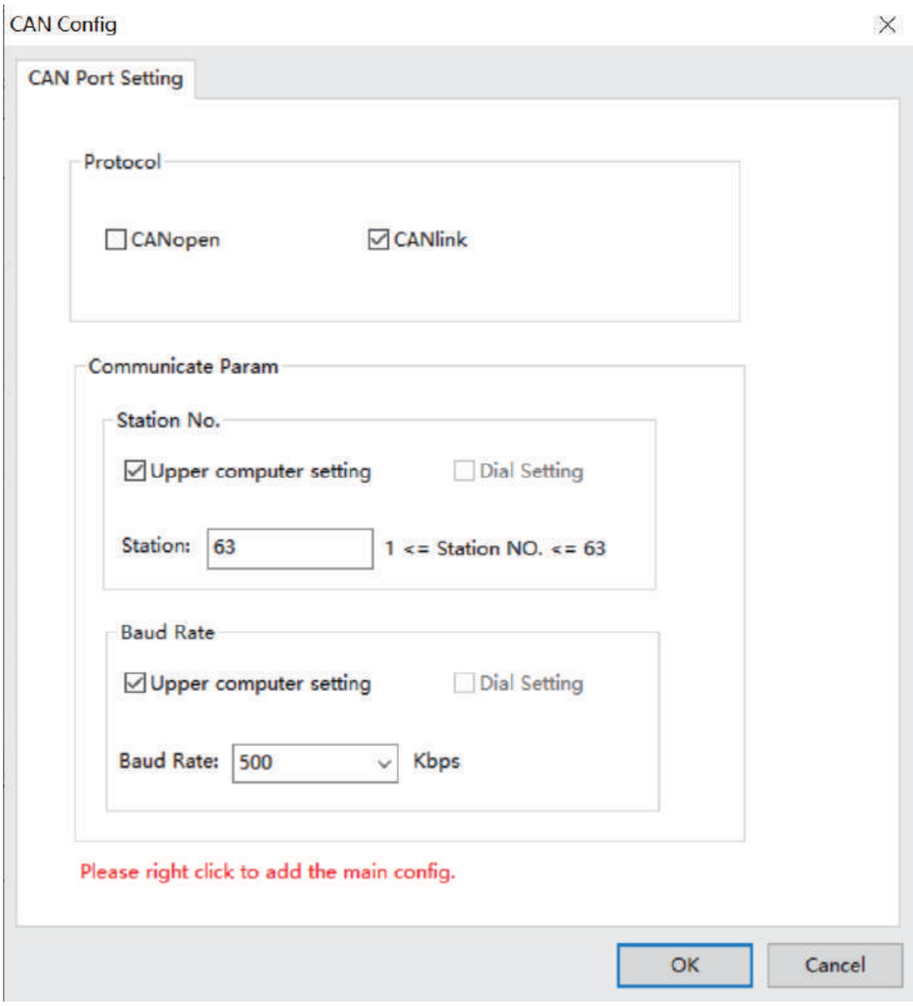
In communication scenarios where multiple slave stations must synchronously act and respond (for example, servo-driven synchronous multi-axes control and position-controlled high-speed movement), you need to set Synchronous Write for the master station. The master station writes data to slave stations and then sends broadcast command frames to make slave stations run simultaneously.

8.4.2 CANlink Configuration

Take the following steps to configure a CANlink network.

1. Configure the CANlink network through AutoShop and define the data to be exchanged.
2. Download configurations to the H5U series PLC.

After creating a project, choose "Project Manager" > "Config", and then double-click "CAN". In the window displayed, select "CANlink" as the communication protocol. Then, the system automatically determines whether the current PLC is the CANlink master station or the CANlink slave station based on the presence of the CANlink configuration.



☐ CANopen ☑ CANlink

Station No.: ☑ Upper computer setting ☐ Dial Setting

Station: 63 1 <= Station NO. <= 63

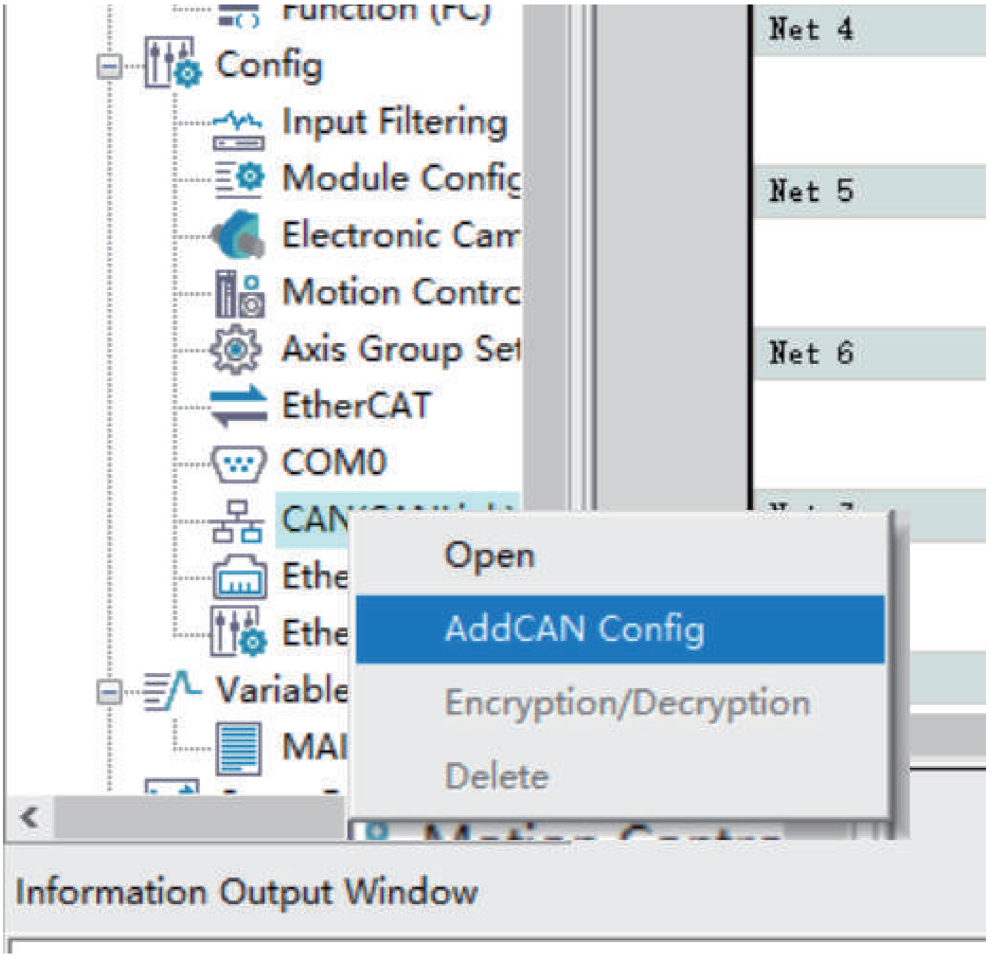
Baud Rate: ☑ Upper computer setting ☐ Dial Setting

Baud Rate: 500 Kbps

Please right click to add the main config.

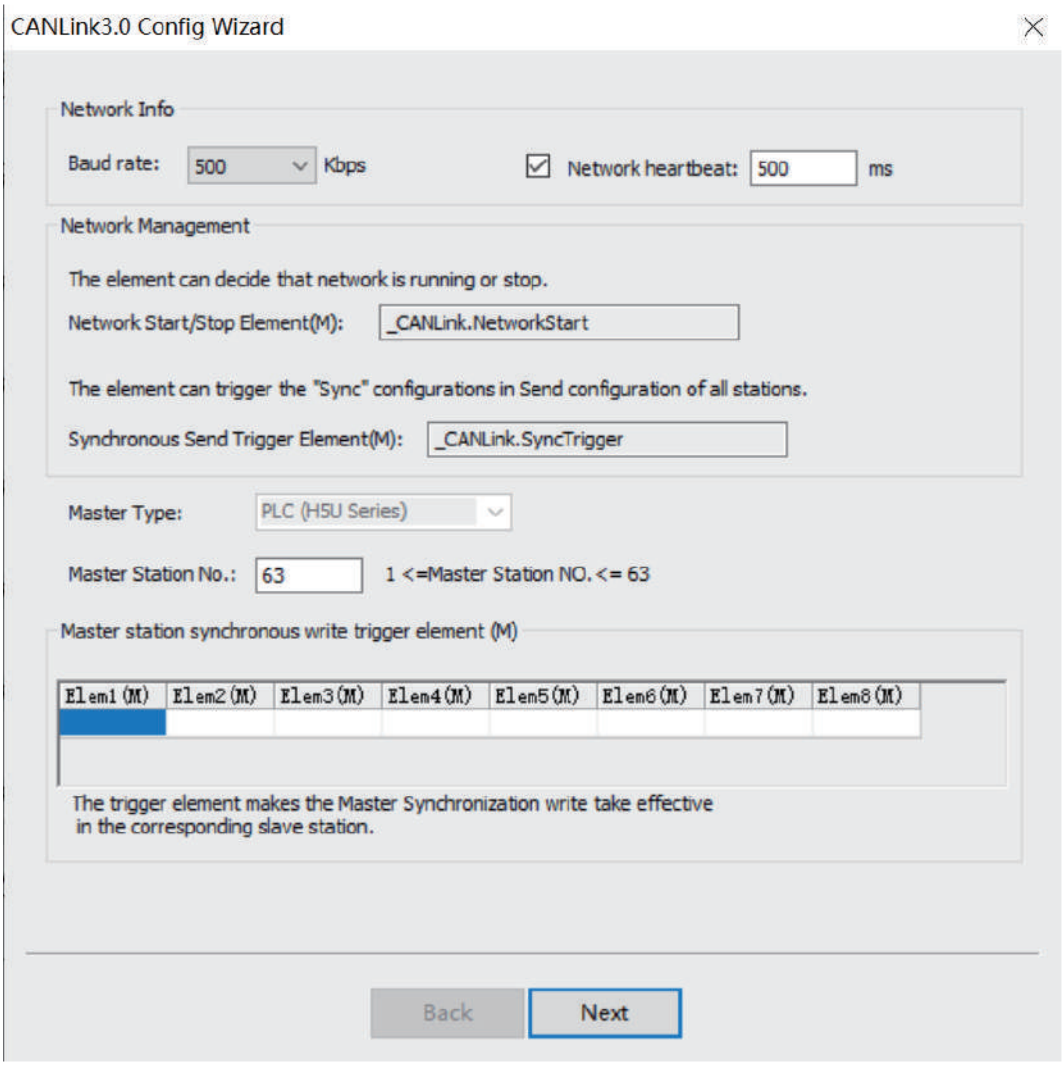
Select "CANlink", set the slave No. and baud rate, and then click "OK". When "CANlink" is selected, the system automatically determines whether the current PLC is the CANlink master station or the CANlink slave station based on the presence of the CANlink configuration.

In this example, CAN is configured as the CANlink slave station. Choose "Project Manager" > "Config". Right-click "CAN". In the window displayed, select "Add CAN configuration" to configure it as the CANlink master station, as shown in the following figure.



CANlink3.0 Config Wizard page

Double-click "CANlink Config'. The "CANlink3.0 configuration wizard" page is displayed, as shown in the following figure.



CANLink3.0 Config Wizard

Network Info

Baud rate: 500 Kbps

☑ Network heartbeat: 500 ms

Network Management

The element can decide that network is running or stop.

Network Start/Stop Element(M): _CANLink.NetworkStart

The element can trigger the "Sync" configurations in Send configuration of all stations.

Synchronous Send Trigger Element(M): _CANLink.SyncTrigger

Master Type: PLC (H5U Series)

Master Station No.: 63 1 <=Master Station NO. <= 63

Master station synchronous write trigger element (M)

Elem1(M)	Elem2(M)	Elem3(M)	Elem4(M)	Elem5(M)	Elem6(M)	Elem7(M)	Elem8(M)

The trigger element makes the Master Synchronization write take effective
in the corresponding slave station.

- Baud rate (required)
  Eight options are available for different scenarios: 20 kbps, 50 kbps, 100 kbps, 125 kbps, 250 kbps, 500 kbps, 800 kbps, and 1 Mbps. You can select the desired option from the drop-down list, and then download the configuration to the master station (this parameter is valid for the master station only, and needs to be manually modified on a slave station). You can select the baud rate based on the bus load and communication distance.

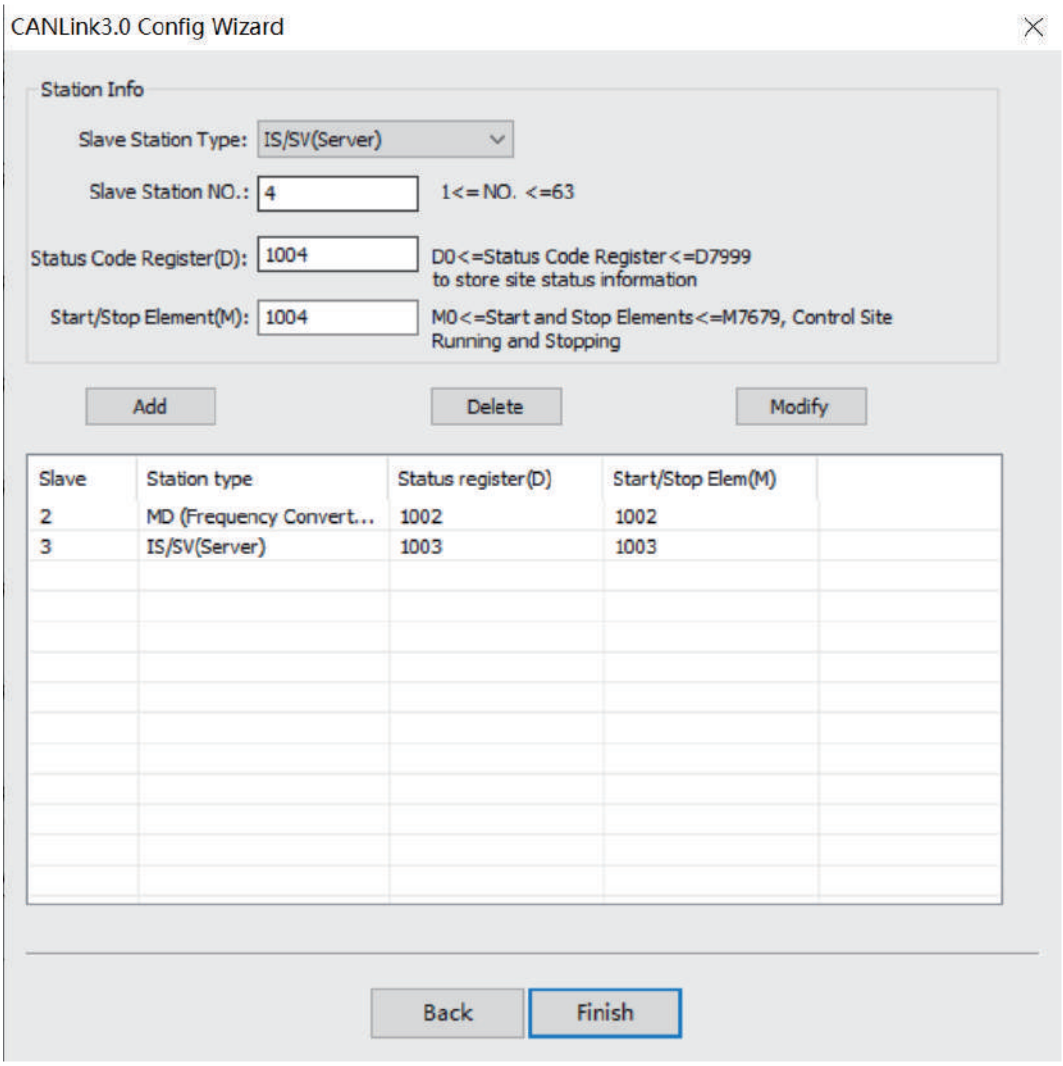
- Network heartbeat (optional)
  All slave stations send heartbeats to the master station at a specified interval. The master station monitors the state (online or offline) of each slave station through the heartbeat mechanism. Slave stations monitor the status of the master station through its heartbeats. (An interval of more than 200 ms is recommended.) If you deselect this parameter, the heartbeat function is disabled and the system cannot monitor the network.

- Master Station No. (required)
  In this example, the master station No. is the number of the PLC that serves as a master station. The number cannot be changed. If the number entered is inconsistent with the actual number, the PLC will determine that the downloaded configuration is invalid.

  For example, if you enter 7, the configuration is valid only when downloaded to station 7. Station 7 then assigns the configuration to other stations. The CANlink network configuration is downloaded to the master station and then assigned to slave stations. In this way, the system can monitor and manage the entire network through the master station in the background.

- Master station synchronous write trigger element (optional)
  It is an element triggering Synchronous Write for the master station. When a trigger element (M) is set, the corresponding configuration takes effect. The element is automatically reset after data is sent.

Click "Next". The window for adding a slave station is displayed as follows.



Slave Station Type: IS/SV(Server)

Slave Station NO.: 4 1<=NO. <=63

Status Code Register(D): 1004 D0<=Status Code Register<=D7999 to store site status information

Start/Stop Element(M): 1004 M0<=Start and Stop Elements<=M7679, Control Site Running and Stopping

Slave	Station type	Status register(D)	Start/Stop Elem(M)
2	MD (Frequency Convert...	1002	1002
3	IS/SV(Server)	1003	1003

Station information page

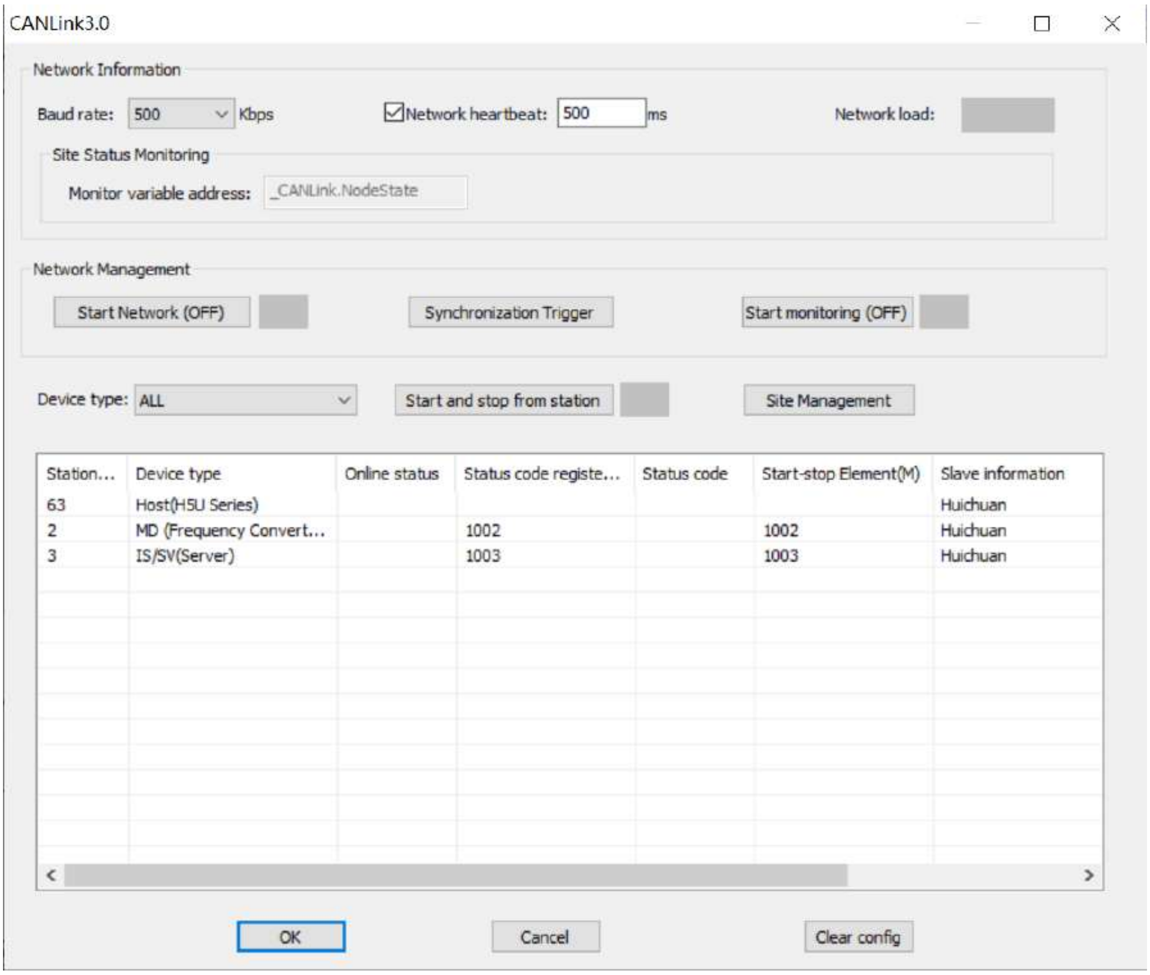
- Add
  After configuring a slave station, click "Add". The station is added to the list.
- Delete
  Select a station and click "Delete". In the confirmation dialog box "Delete it?", click "OK" (you can delete multiple stations at a time).
- Modify
  Select a slave station, modify parameters on the "Station Info" page, and then click "Modify". Do not modify the station type.

- Slave No.
  Set the CANlink slave station number.
- Status Code Register (D)
  It is used to save the status of a slave station fed back through heartbeat frames of the slave station.
- Start/Stop Element (M)
  It is an M element used to start or stop communication. When M is ON, communication is started. When M is OFF, communication is stopped.

Note

In the configuration wizard, click "OK" to save the modification made in the wizard and exit; click the "X" button in the upper right corner of the wizard to cancel the modification and exit.

Then, click "Finish". The following window is displayed.



CANLink3.0

Network Information

Baud rate: 500 Kbps

☑ Network heartbeat: 500 ms

Network load:

Site Status Monitoring

Monitor variable address: _CANLink.NodeState

Network Management

Device type: ALL

Station...	Device type	Online status	Status code registe...	Status code	Start-stop Element(M)	Slave information
63	Host(H5U Series)					Huichuan
2	MD (Frequency Convert...		1002		1002	Huichuan
3	IS/SV(Server)		1003		1003	Huichuan

- Network Information
  Baud rate: Indicates the baud rate of the master station.

  Network heartbeat: Indicates that the heartbeat function is enabled after this parameter is selected.

  Network load: Calculates the real-time load of the network (this parameter is displayed only when the network load is monitored during running of devices).

  - Network load ⩽ 50: Green (good)
  - 50 < Network load ⩽ 75: Yellow (warning)
  - 75 < Network load ⩽ 90: Red (major warning)
  - Network load > 90: ERR, red background (error)

- Site Status Monitoring

  The online status of the station will be updated to the system variable _CANLink.NodeState[64], in which _CANLink.NodeState[0] is the status of the local station, while _CANLink.NodeState[station No.] is the status of the slave station.

Status Value	Definition
1	Configurations of the slave station are available.
2	The slave station is running.
5	The slave station is disconnected.

Note

If the heartbeat function is disabled, the station monitoring function is meaningless.

- Network Management

  Start/Stop Network (OFF) (enabled when monitoring is enabled): Starts and stops network communication.

  Synchronous Send: Synchronization will be triggered. You can enable the function in the user program by setting _CANLink.SyncTrigger. After synchronous data frames are sent, _CANLink. SyncTrigger will be automatically reset.

  Start Monitor (OFF): Starts and stops network monitoring.

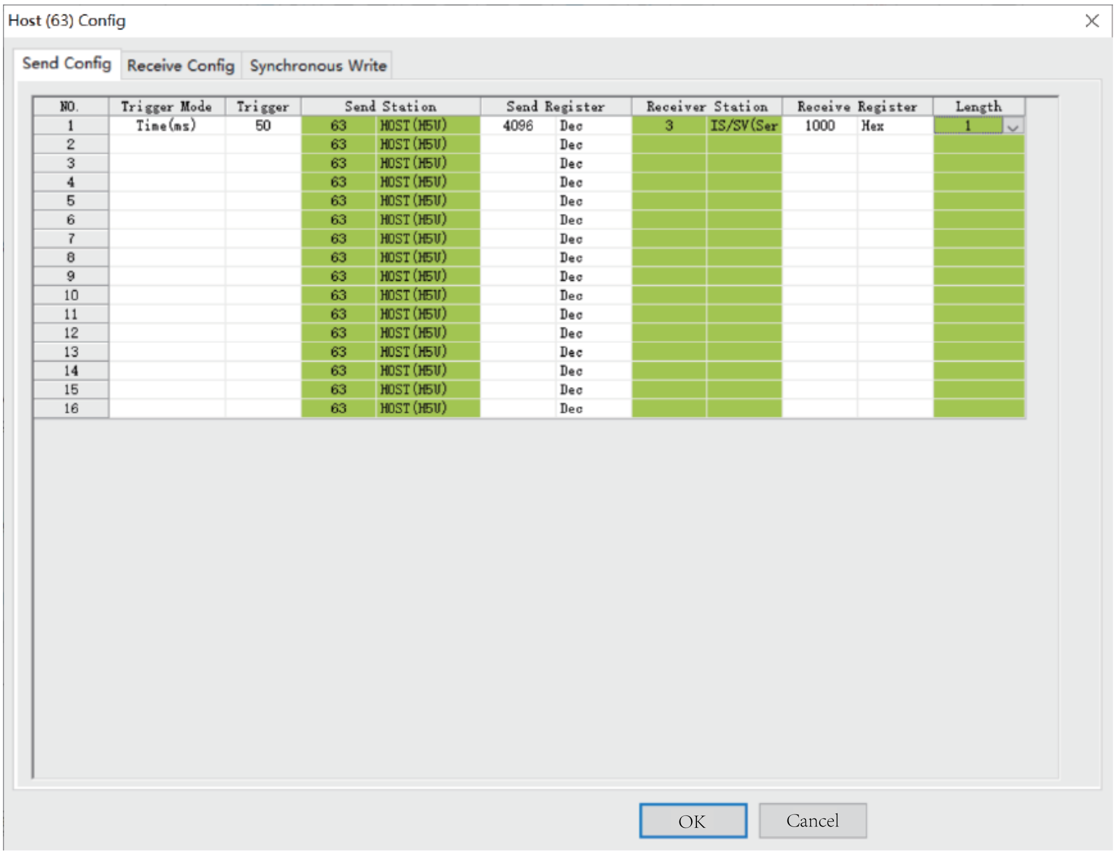
  Device type: Filters displayed stations.

  Slave start/stop: Select a slave station and control its start/stop of communication.

  Station management: Click "Station Management". The initialization wizard page is displayed. You can modify parameters of the master or slave station.

  Station configuration: On the main screen, double-click a station. The communication configuration window is displayed. Communication configuration includes sending configuration, receiving configuration, and synchronization configuration (for the master station only).

  Sending configuration:



Host (63) Config

NO.	Trigger Mode	Trigger	Send Station		Send Register		Receiver Station		Receive Register		Length
1	Time(ms)	50	63	HOST(H5U)	4096	Dec	3	IS/SV(Ser	1000	Hex	1
2			63	HOST(H5U)		Dec					
3			63	HOST(H5U)		Dec					
4			63	HOST(H5U)		Dec					
5			63	HOST(H5U)		Dec					
6			63	HOST(H5U)		Dec					
7			63	HOST(H5U)		Dec					
8			63	HOST(H5U)		Dec					
9			63	HOST(H5U)		Dec					
10			63	HOST(H5U)		Dec					
11			63	HOST(H5U)		Dec					
12			63	HOST(H5U)		Dec					
13			63	HOST(H5U)		Dec					
14			63	HOST(H5U)		Dec					
15			63	HOST(H5U)		Dec					
16			63	HOST(H5U)		Dec					

- Trigger mode

Time (ms): It is applicable to all devices. The station applies the configuration at a fixed interval. The value ranges from 1 ms to 30000 ms.

Event (M): It is applicable to the host and PLC. The station applies the configuration when the trigger condition (M element) is set. Multiple configurations can be triggered by the same M element. The element is automatically reset after data is sent. Edge trigger instructions must be used to operate M elements; otherwise, the network load will be excessive.

Synchronization: It is applicable to all devices. The master station applies the configuration when the system variable _CANLink.SyncTrigger is set. The element is automatically reset after data is sent.

Event (ms): It is applicable to IS, MD, and remote extension modules (TCM/NTCM). The station applies the configuration when it detects the changed value of the sent register and the trigger condition (disabling time) is met.

The disabling time indicates the minimum interval for sending the same configuration.

Maximum number of configuration items for one station: 256 for the host (master station), 16 for one slave station, and 256 for all slave stations.

If you select a configuration item and press "Insert", an empty configuration line will be added following the item. If you select a configuration item and press "Delete", the item will be deleted. In addition, you can press shortcut keys or right-click an item to copy-paste or delete it, and insert or delete a line.

- Register

Host and PLC register values correspond to D elements. IS and MD register values correspond to function codes. TCM/NTCM corresponds to BFM.

- Number of registers
  It is the number of sent or received consecutive D elements or function codes.
- Point-to-multi-point configuration
  When a sending station is also a receiving station, the station applies the point-to-multi-point configuration, in which no receiving station is specified. If you enter the sending station number into the receiving configuration table, the configured station can receive data sent by the sending station. The received register is the D element or function code corresponding to the receiving station.
- Received data
  The entries in the gray background indicate data received from other stations, including point-to-point and point-to-multi-point data. You can see which element or function code of which station will affect the configured station.
- Receiving configuration
  Receiving configuration applies to receiving point-to-multi-point data from other stations. Each station can receive point-to-multi-point data from eight stations.

Note

The point-to-multipoint configuration enables simultaneous application of data. This is equivalent to master synchronous configuration, but does not limit the data sending capability to the master alone. Each station can receive point-to-multipoint data from up to eight different stations, but the number of stations each station can send point-to-multipoint data to is not limited. In other words, all nodes in the network except for the sending station can receive such point-to-multipoint data. However, to receive point-to-multipoint data from a station, the receiver must be configured to allow receiving such data from the station.

Synchronous write configuration for the master station

When the trigger condition (M) is set, the Synchronous Send configuration for the master station takes effect. You can select different trigger conditions (M) to display, add, modify, or delete synchronization configurations. Synchronization configuration is applicable to scenarios in which an operation needs to be initiated synchronously.

As shown in the figure, when M1 is 1, the master station sends the three configuration items successively. Upon receipt of the items, slave stations store them in the buffer. After the last data entry is sent, the master station automatically sends a configuration application command. Upon receipt of the command, all slave stations automatically write the data in the buffer in corresponding elements or function codes. As shown in the figure, PLC 10 writes the D10 value in D10, servo 20 writes the D20 value in H200, and AC drive 30 writes the D30 value in HF003. All these values are synchronously written when slave stations receive the configuration application command. After the command is sent, the master station automatically resets the trigger element M1. Edge trigger instructions must be used to operate M elements; otherwise, the network load will be excessive.

Precautions for the trigger condition (M):

- Each trigger condition associates a maximum of 16 configuration items. It determines whether the associated synchronization configuration is valid. A maximum of 8 trigger conditions (M) are allowed.

- You can select a trigger condition from the drop-down list.
- During synchronization configuration of a 32-bit servo register, data must correspond to high-order 16 bits and low-order 16 bits respectively for the same trigger element. That is, two data entries must be written for one trigger element, one corresponding to high address bits of the 32-bit function code, and the other corresponding to low address bits. If only one entry is written or two entries are written for two trigger elements respectively, the servo will return an error, and the configuration cannot continue.

Example of 32-bit servo register synchronization configuration:

As shown in the following figure, H1112 is a 32-bit function code of the servo. During configuration of the function code, two data entries must be written, corresponding to high and low address bits respectively. When M3 is set, the master station writes D201 and D202 values in H1112. When all of the five data entries of M3 are sent, the master station sends a command to enable the slave stations and apply the configurations. Then M3 is automatically reset.

If only one address is processed for one trigger element, the servo will return an error so that synchronization cannot continue. The error will be recorded in D8307 of the master station. Fault codes are listed in *“8.5.6.2 Fault Code List” on page 300Fault Code List*.

- Master station fault codes and processing
  The following table lists configuration errors and causes. You can use the system variable _CANLink. ConfigErr to view the details.

Table 8–2 Configuration errors

Fault Code※	Cause	Solution
XX00	Reserved	None
XX01	Incorrect code	Check whether the internal definition is correct.
XX02	Incorrect index	Check whether the device type is correct.
XX03	Incorrect information	Check whether the address is valid and check the read-write property.
XX04	Reserved	Reserved
XX05	Incorrect data length	Check whether the data length exceeds the limit.
XX06	Configuration frames failing to respond within a specified time	Check whether the connection is normal.

The following table lists abnormality codes and causes. You can use the system variable _CANLink. SyncWrErr to view the details.

Table 8–3 Fault codes

Fault Code※	Cause	Solution
XX00	Reserved	Reserved
XX01	Invalid command code	Check whether the internal definition is correct.
XX02	Abnormal address	Check whether the address is normal or whether the address can be accessed.
XX03	Abnormal data	Check whether the data is within a specified range.
XX04	Invalid operation	Check whether the operation is authorized.
XX05	Invalid length	Check whether the data length exceeds the limit.
XX06	Responding timeout	Check whether the connection is normal.

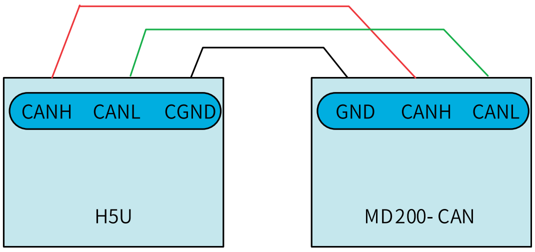
Note

- The fault codes are in decimal, where XX indicates the station number. Specifically, a fault code indicates that an error occurred when configuring XX station or sending commands to XX station.
- Fault codes of PLC slaves are similar to those of the master, except that the fault codes of PLC slaves do not contain a station number.

8.4.3 AC Drive Communication Example

Use one H5U and one MD200-CAN AC drive to control the start/stop and write the frequency of the AC drive through the CANlink bus.

1. Connect the CANlink bus.



2. Configure function codes for the slave MD200 AC drive.

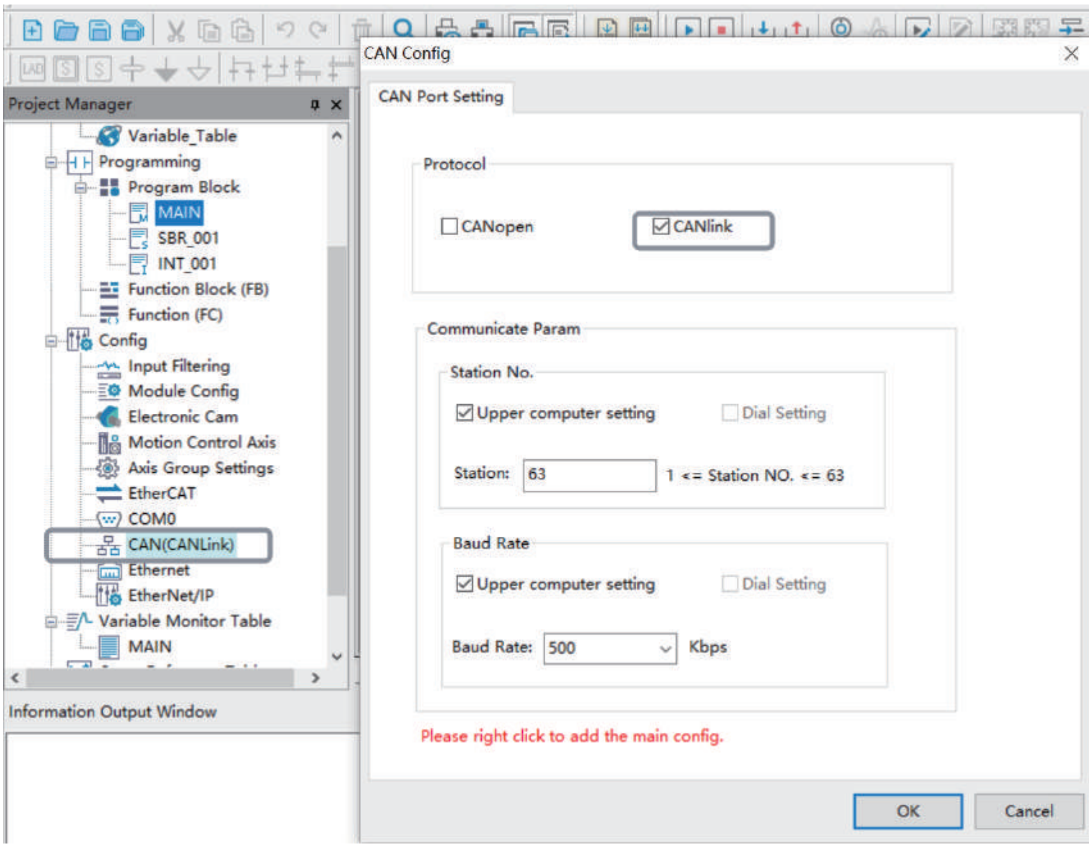
Function Code	Name	Value	Description
FD-00	Baud rate	5005	CANlink baud rate: 500 kbps
FD-02	Local address	1	Local station No. is 1.
F0-02	Command source selection	2	Communication setting
F0-03	Main frequency reference source	9	Communication setting

Control parameter addresses when MD200-CAN is used for communication with the host controller PLC:

Address	Name	Description
H1000	Communication frequency reference	–10000 to +10000 (decimal)
H1001	Feedback running frequency	-
H2000	Control commands	0001: Forward running 0002: Reverse running 0003: Forward jog 0004: Reverse jog 0005: Coast to stop 0006: Decelerate to stop 0007: Fault reset

3. Configure the H5U master station

- In AutoShop, right-click CANlink, set the station No. of the H5U master station to 63, and set the baud rate to 500 kbps.



Protocol

□ CANopen ☑ CANlink

Communicate Param

Station No.

☑ Upper computer setting □ Dial Setting

Station: 63 1 <= Station NO. <= 63

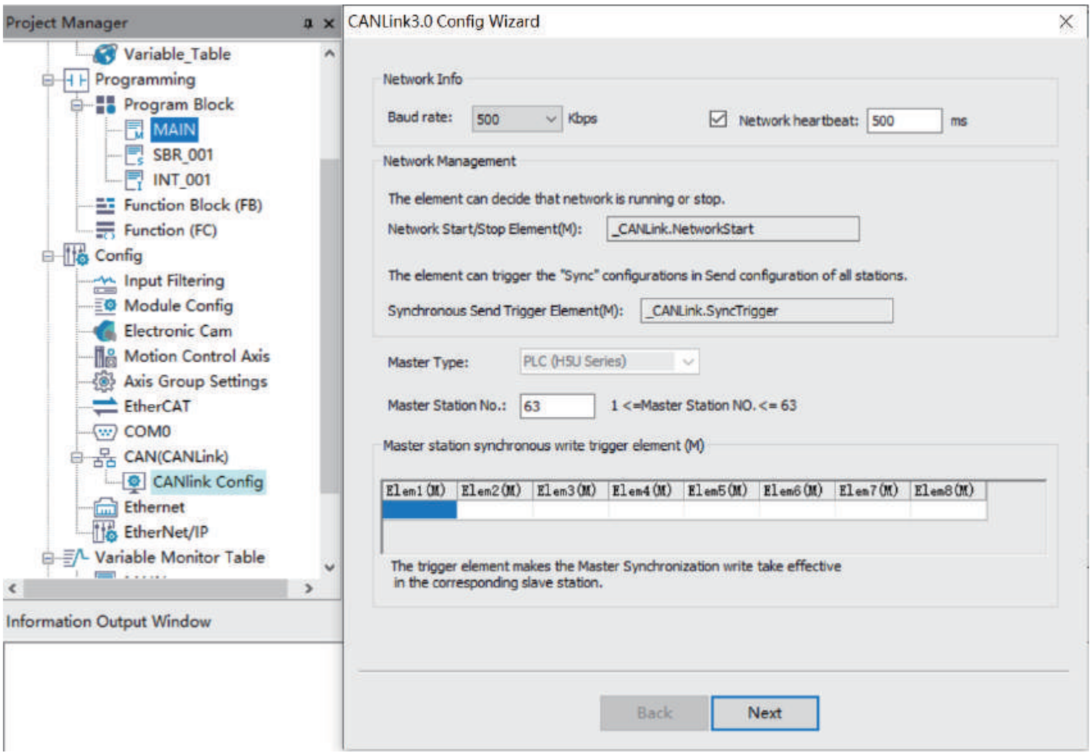
Baud Rate

☑ Upper computer setting □ Dial Setting

Baud Rate: 500 Kbps

Please right click to add the main config.

- Add an MD AC drive.



Network Info

Baud rate: 500 Kbps ☑ Network heartbeat: 500 ms

Network Management

The element can decide that network is running or stop.

Network Start/Stop Element(M): _CANLink.NetworkStart

The element can trigger the "Sync" configurations in Send configuration of all stations.

Synchronous Send Trigger Element(M): _CANLink.SyncTrigger

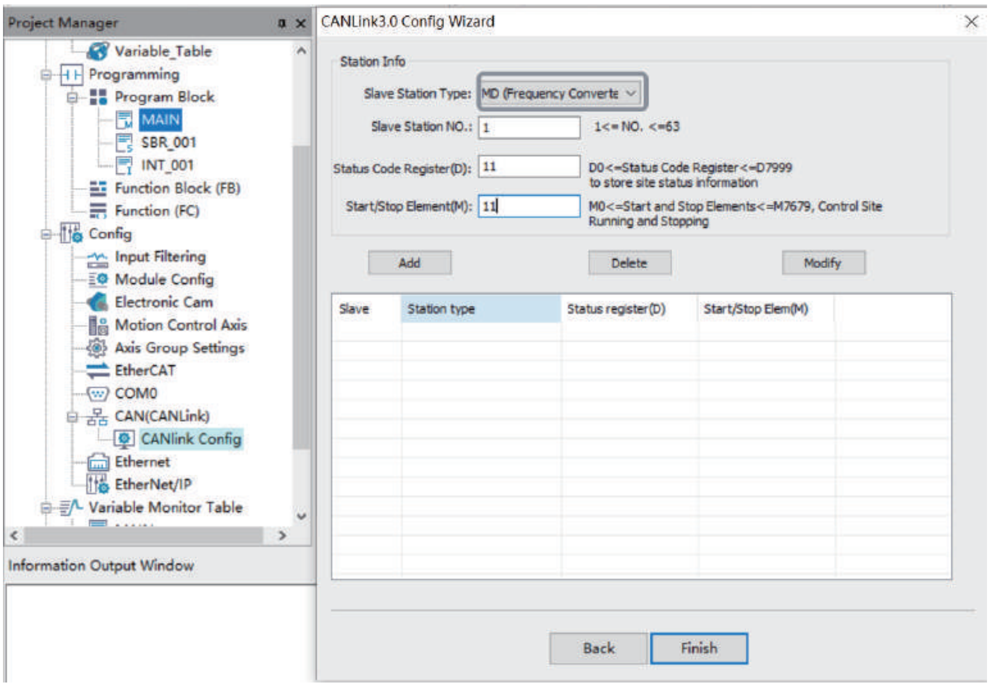
Master Type: PLC (H5U Series)

Master Station No.: 63 1 <=Master Station NO. <= 63

Master station synchronous write trigger element (M)

Elem1(M)	Elem2(M)	Elem3(M)	Elem4(M)	Elem5(M)	Elem6(M)	Elem7(M)	Elem8(M)

The trigger element makes the Master Synchronization write take effective in the corresponding slave station.



Slave Station Type: MD (Frequency Converte

Slave Station NO.: 1 1<= NO. <=63

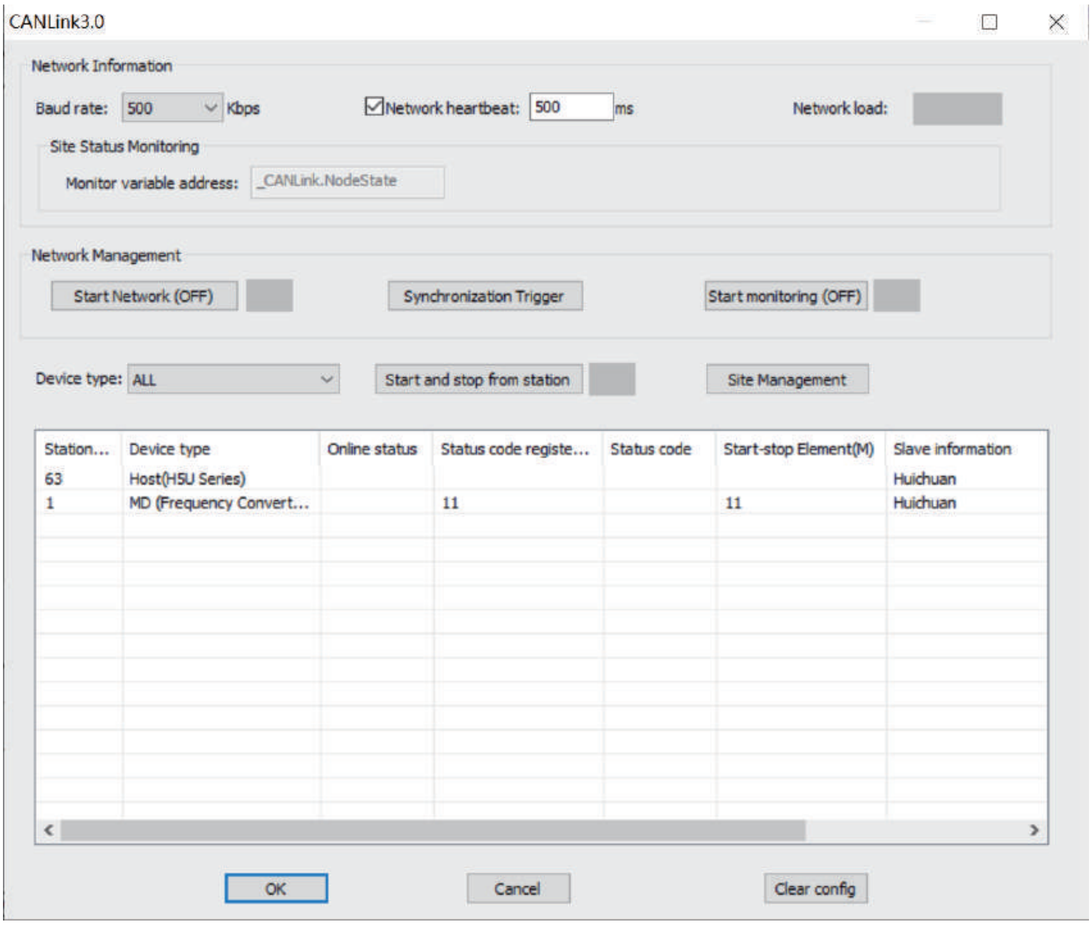
Status Code Register(D): 11 D0<=Status Code Register<=D7999 to store site status information

Start/Stop Element(M): 11 M0<=Start and Stop Elements<=M7679, Control Site Running and Stopping

Slave	Station type	Status register(D)	Start/Stop Elem(M)

Set the slave station No. according to that defined by FD-02 of the AC drive.

- Configure the master station to send data.

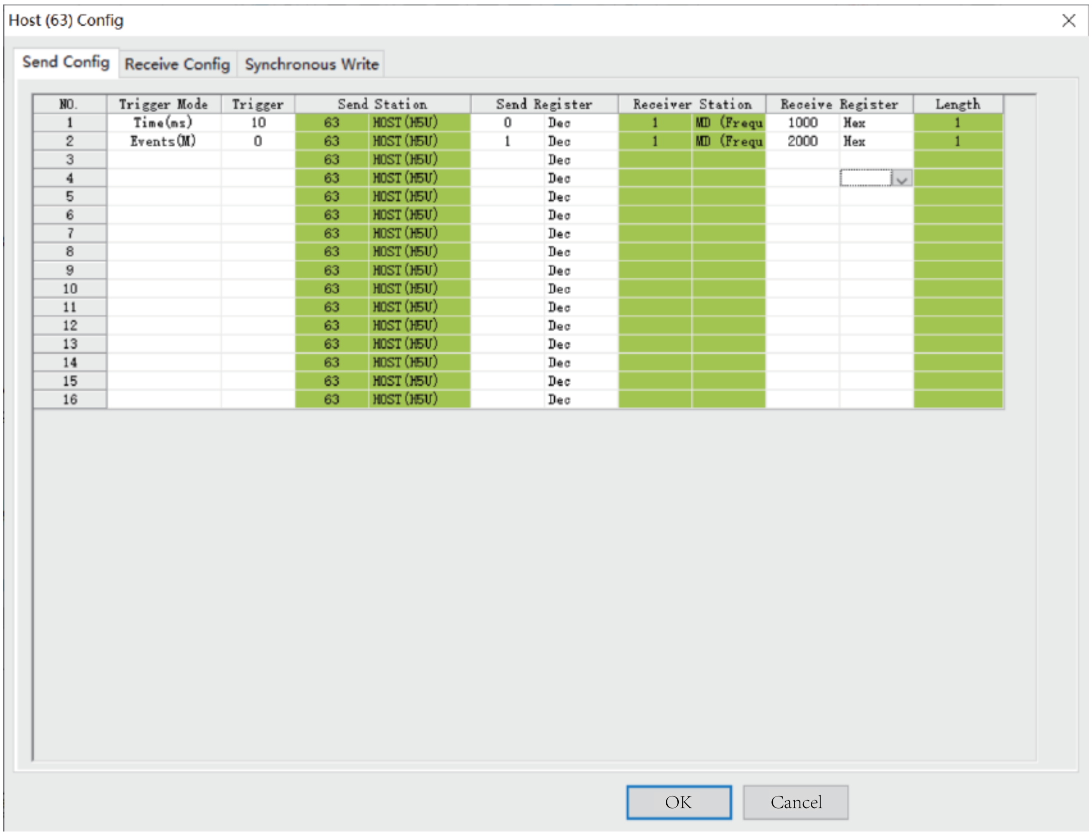


Baud rate: 500 Kbps ☑Network heartbeat: 500 ms Network load:

Monitor variable address: _CANLink.NodeState

Device type: ALL

Station...	Device type	Online status	Status code registe...	Status code	Start-stop Element(M)	Slave information
63	Host(H5U Series)					Huichuan
1	MD (Frequency Convert...		11		11	Huichuan



Host (63) Config

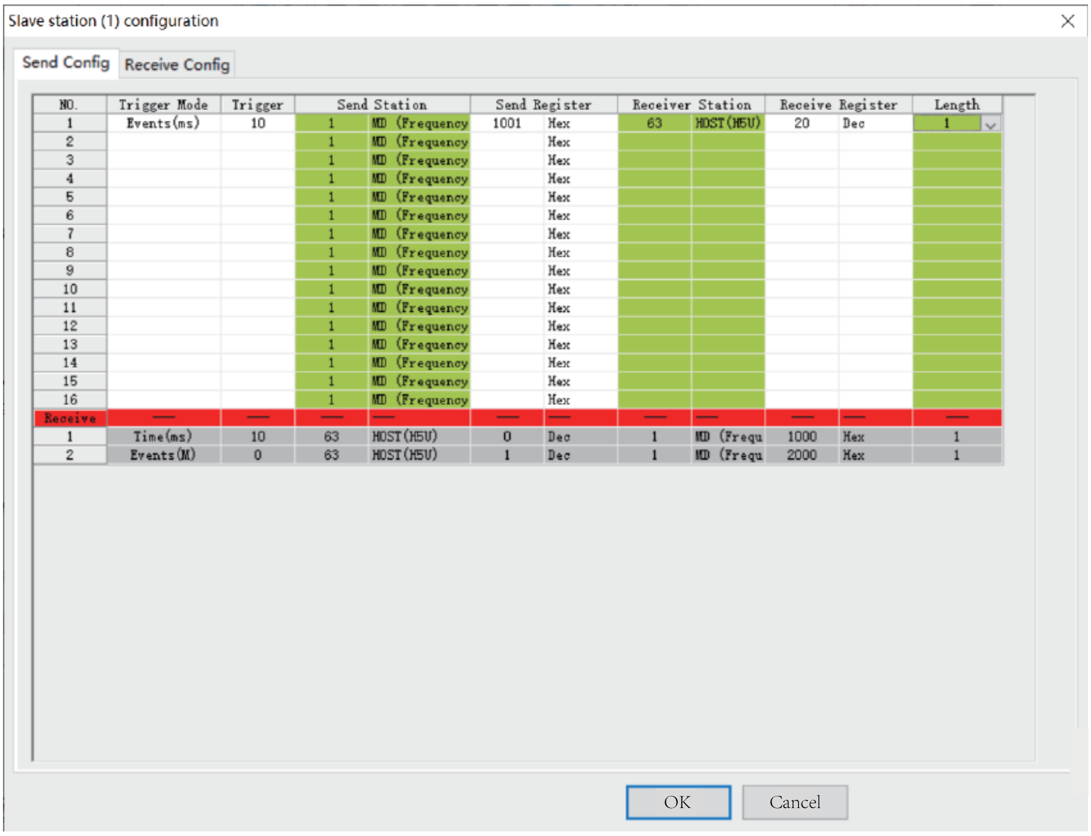
NO.	Trigger Mode	Trigger	Send Station		Send Register		Receiver Station		Receive Register		Length
1	Time(ms)	10	63	HOST(H5U)	0	Dec	1	MD (Frequ	1000	Hex	1
2	Events(M)	0	63	HOST(H5U)	1	Dec	1	MD (Frequ	2000	Hex	1
3			63	HOST(H5U)		Dec					
4			63	HOST(H5U)		Dec					
5			63	HOST(H5U)		Dec					
6			63	HOST(H5U)		Dec					
7			63	HOST(H5U)		Dec					
8			63	HOST(H5U)		Dec					
9			63	HOST(H5U)		Dec					
10			63	HOST(H5U)		Dec					
11			63	HOST(H5U)		Dec					
12			63	HOST(H5U)		Dec					
13			63	HOST(H5U)		Dec					
14			63	HOST(H5U)		Dec					
15			63	HOST(H5U)		Dec					
16			63	HOST(H5U)		Dec					

Then:

The H5U master station writes the value of the D0 element to the 1000 (frequency reference) address register of the slave station 1# at an interval of 10 ms.

When the M0 element is ON, the H5U master station writes the value of the D1 element to the 2000 (AC drive control word) address register of the slave station 1#.

- Configure the slave station to send data.

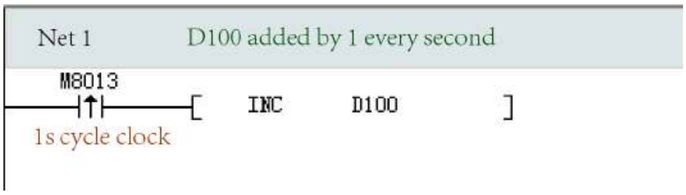


NO.	Trigger Mode	Trigger	Send Station		Send Register		Receiver Station		Receive Register		Length
1	Events(ms)	10	1	MD (Frequency	1001	Hex	63	HOST(H5U)	20	Dec	1
2			1	MD (Frequency		Hex					
3			1	MD (Frequency		Hex					
4			1	MD (Frequency		Hex					
5			1	MD (Frequency		Hex					
6			1	MD (Frequency		Hex					
7			1	MD (Frequency		Hex					
8			1	MD (Frequency		Hex					
9			1	MD (Frequency		Hex					
10			1	MD (Frequency		Hex					
11			1	MD (Frequency		Hex					
12			1	MD (Frequency		Hex					
13			1	MD (Frequency		Hex					
14			1	MD (Frequency		Hex					
15			1	MD (Frequency		Hex					
16			1	MD (Frequency		Hex					
Receive	—	—	—	—	—	—	—	—	—	—	—
1	Time(ms)	10	63	HOST(H5U)	0	Dec	1	MD (Frequ	1000	Hex	1
2	Events(M)	0	63	HOST(H5U)	1	Dec	1	MD (Frequ	2000	Hex	1

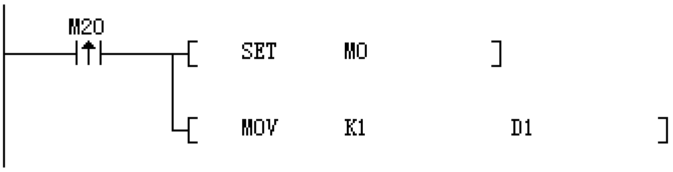
The slave AC drive 1# converts the value in the 1001 (running frequency) address register and then writes the value to the D20 element of the master station 63# at an interval of 10 ms.

- PLC programming

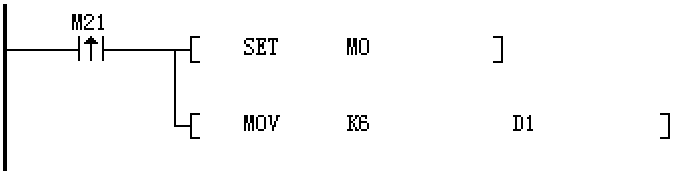
Set the AC drive running frequency to 20 Hz.



Set M20 to ON to start the AC drive and make it rotate in the forward direction.

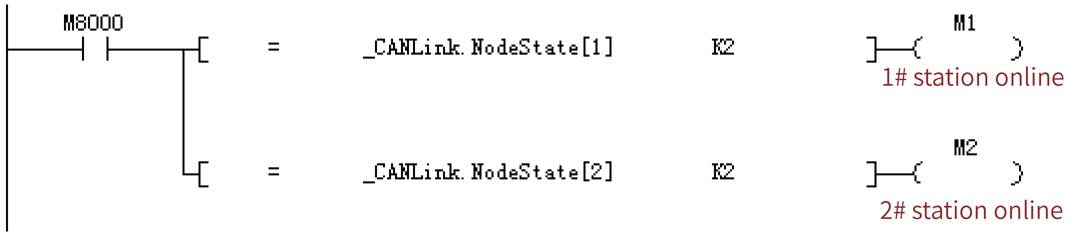


Set M21 to OFF to stop the AC drive.



Judge the online status of the CANlink slave station.

The online status of the slave station will be updated to the system variable _CANLink.NodeState [64], in which _CANLink.NodeState[1] is the state of the station 1.



8.4.4 CANlink Indicator

You can judge the CANlink communication state based on the CANlink indicator.

Table 8–4 States of CANlink indicator

Indicator	State	Description
Communication (green)	Off	CANlink bus not connected or disconnected
	On	CANlink bus connected (remote frames received on the node)
	Flashing (⩽ 3 Hz)	During CANlink communication, one flashing per frame of bus data sent or received
	Flashing (5 Hz)	Flag monitor
Fault (red)	Off	No fault
	On	Monitor timeout (node), no node (monitor)
	Flashing (0.5 Hz)	CANlink configuration error (for the configurator)
	Flashing (1 Hz)	Node lost or crash (for the monitor)
	Flashing (5 Hz)	CANlink address conflict

8.4.5 CANlink Communication Troubleshooting

Check the following items when a CANlink communication error occurs.

- Check the termination resistor
  Power off all devices. Use a multimeter to measure the resistance between CANH and CANL. The resistance should be about 60 Ω. If the resistance is too small, there are termination resistors incorrectly connected at other locations. In this case, disconnect these termination resistors. If only one termination resistor is available, the resistance is about 120 Ω, and the network connection is bad. If no termination resistor is available, communication fails. Provide termination resistors between the stations at both ends of the network.

- Check the baud rate
  Check whether the baud rate is normal. Baud rates of all stations in the network must be the same; otherwise, communication fails. Power off and then on the device or switch it from STOP to RUN so that the baud rate can take effect.

- Others
  In case of strong interference, reduce the baud rate.

8.5 CANopen Communication

8.5.1 CANopen Communication Protocol

The H5U supports the CANopen communication standard protocol DS301.

Table 8–5 CANopen communication protocol standard

Software Function Module	Slave	Master
Supported protocol	DS301 V4.02	DS301 V4.02
Maximum number of TPDOs	8	64
Maximum number of RPDOs	8	64
Number of slave station nodes	/	30
Baud rate and communication distance	1 Mbps/25 m 800 kbps/50 m 500 kbps/100 m 250 kbps/250 m 125 kbps/500 m 50 kbps/1000 m 20 kbps/2500 m 100 kbps 10 kbps	1 Mbps/25 m 800 kbps/50 m 500 kbps/100 m 250 kbps/250 m 125 kbps/500 m 50 kbps/1000 m 20 kbps/2500 m 100 kbps 10 kbps
Soft element for data exchange	W300 to W363	D0 to D7999 (configurable)

8.5.2 CANopen Axis Control Instruction List

The following table lists CANopen axis control instructions supported by H5U. See H5U Series Programmable Logic Controller Instructions Guide for detailed usage of related instructions.

Table 8–6 Instruction list

Name	Function
MC_Power_CO	Instruction for enabling the communication control servo axis
MC_Reset_CO	Instruction for resetting faults of the communication control servo axis
MC_ReadActualPosition_CO	Instruction for reading the current position of the communication control axis
MC_ReadActualVelocity_CO	Instruction for reading the current velocity of the communication control axis
MC_Halt_CO	Instruction for halting the motion of the communication control servo axis
MC_Stop_CO	Instruction for stopping the communication control servo axis
MC_MoveAbsolute_CO	Instruction for obtaining the absolute position of the communication control axis
MC_MoveRelative_CO	Instruction for obtaining the relative position of the communication control axis
MC_MoveVelocity_CO	Instruction for selecting the velocity operation mode of the communication control axis

Name	Function
MC_Jog_CO	Instruction for communication control axis jogging
MC_Home_CO	Instruction for communication control axis homing
MC_WriteParameter_CO	Instruction for writing parameters of the communication control axis
MC_ReadParameter_CO	Instruction for reading parameters of the communication control axis

8.5.3 CANopen Terminology

- NMT: Network Management
  Network management includes management of application layers, network states, and node ID allocation. It is implemented in master-slave communication mode. That is, on a CAN network, only one NMT master station exists with one or more slave stations. The service is used to control the slave station state.
- SDO: Service Data Object
  An SDO can access the data in the slave station object dictionary through indexes and sub-indexes. SDOs are used for slave station configuration. Each frame of an SDO request must be answered.
- PDO: Process Data Object
  PDOs are used to transmit real-time data. The data length ranges from one to eight bytes. Data can be transmitted in synchronous and asynchronous modes. PDO frames are primary data exchange frames after slave stations are started.
- SYNC: Synchronous
  Synchronization is implemented in master-slave communication mode. The master SYNC node regularly sends SYNC objects, and the SYNC slave node synchronously executes tasks upon receipt of the objects. SYNC frames are used for synchronous transmission through PDOs.
- COB-ID: Communication Object Identifier
  Each CANopen frame starts with a COB-ID. A COB-ID is not the slave station number. However, it is associated with the slave station number by default.

8.5.4 CANopen Indicator

You can judge the CANopen communication state based on the CANopen indicator.

Table 8–7 States of the CANopen indicator

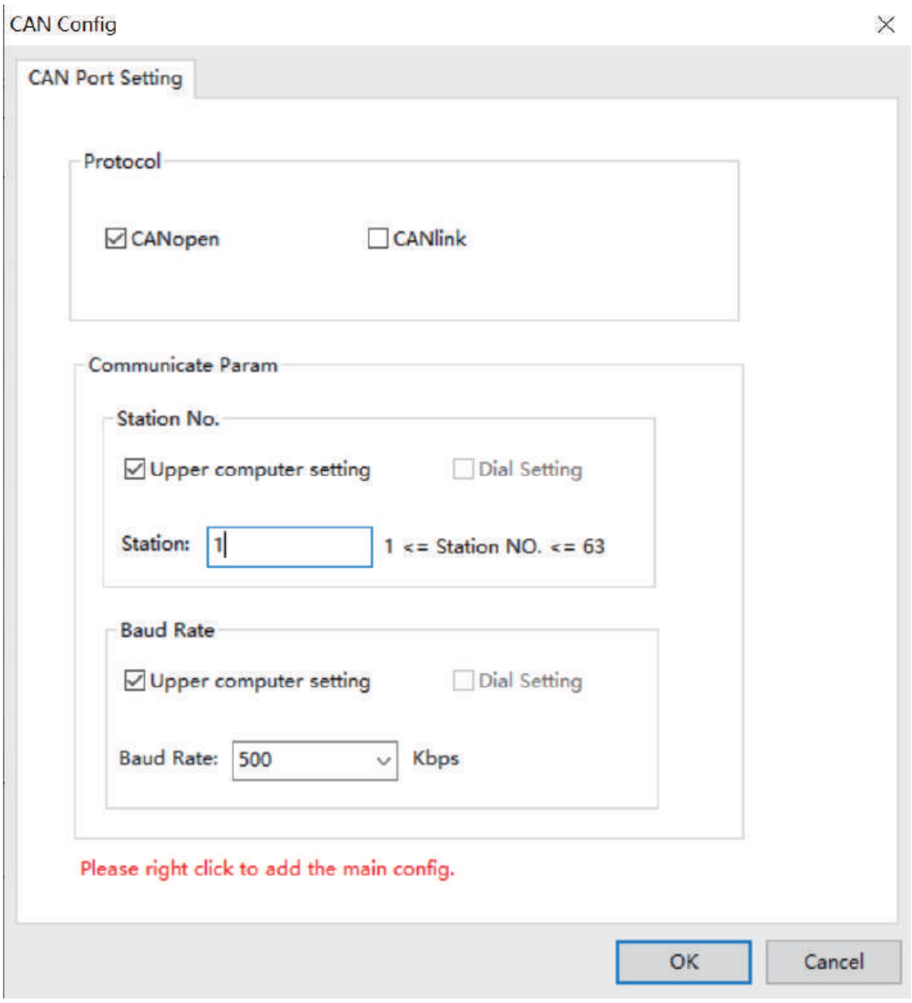
LED Indicator	CAN RUN (Green)	CAN ERR (Red)
Off	None	No error
On	Operational	Bus disconnected
Flashing slowly (at an interval of 0.8s)	Pre-operational	Pre-operational
Flashing slowly (at an interval of 1.2s)	Stopped	At least one error counter of the CAN controller hitting or exceeding the threshold (too many error frames)
Flashing twice slowly (at an interval of 1.6s)	None	Incorrect control event (node protection or heartbeat timeout)

8.5.5 CANopen Configuration

8.5.5.1 Master Configuration

When "CANopen" is selected, the system automatically determines whether the current PLC is the CANopen master station or the CANopen slave station based on the presence of the CANopen configuration.

1. After creating a project, choose "Project Manager" > "Config", and then double-click "CAN". The following window is displayed.



☑ CANopen □ CANlink

☑ Upper computer setting □ Dial Setting

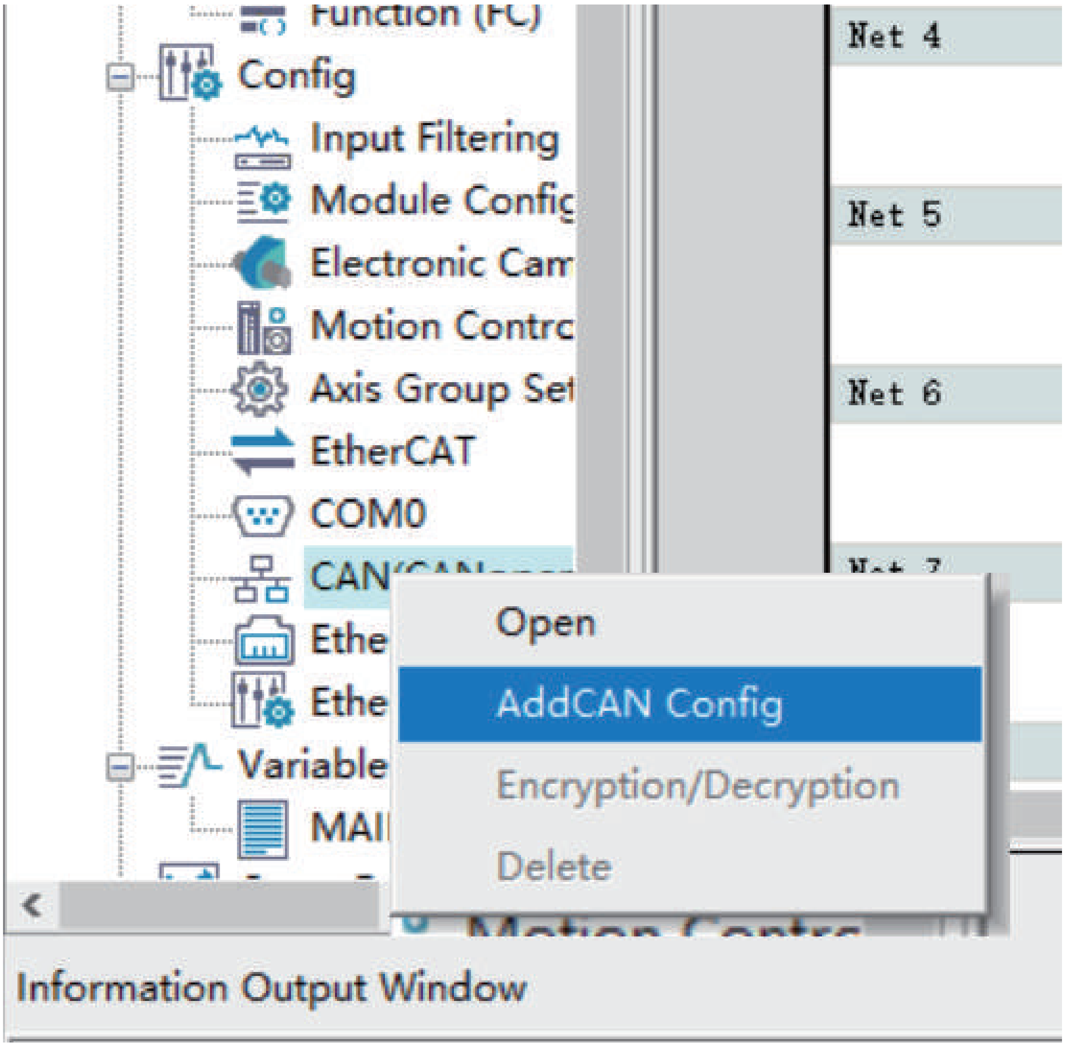
Station: 1 1 <= Station NO. <= 63

☑ Upper computer setting □ Dial Setting

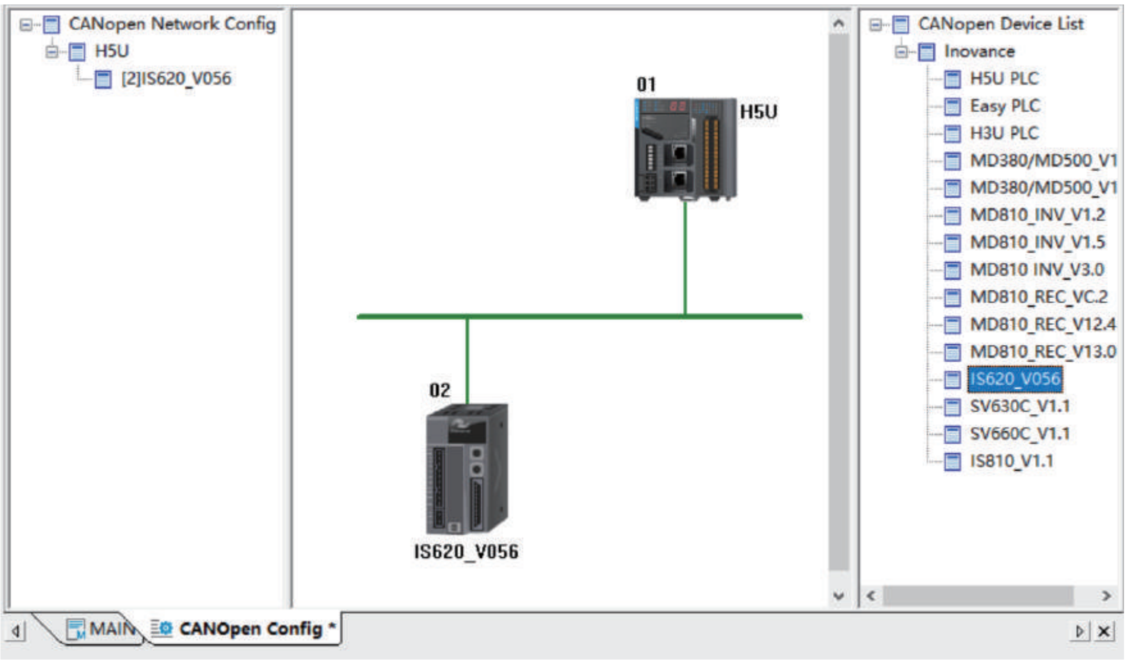
Baud Rate: 500 Kbps

Please right click to add the main config.

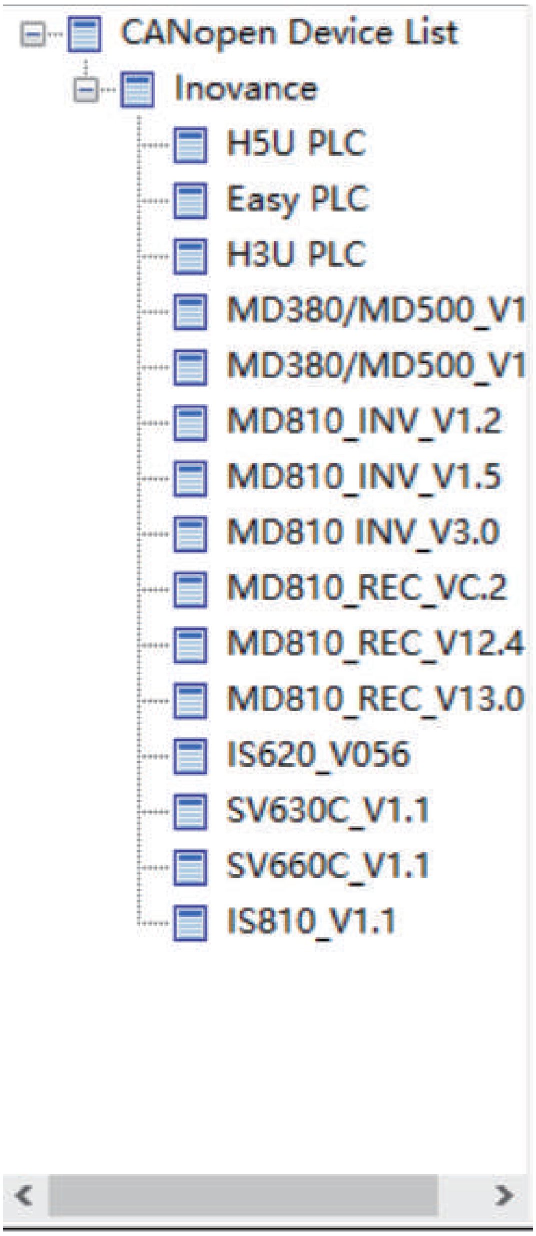
2. Select "CANopen", set the station number and baud rate, and click "OK".
   In this example, CAN is configured as the CANopen slave station. Choose "Project Manager" > "Config". Right-click "CAN". In the window displayed, select "Add CAN configuration" to configure it as the CANopen master station, as shown in the following figure.



3. Double-click "CANOpen Config". The following CANopen configuration page is displayed:

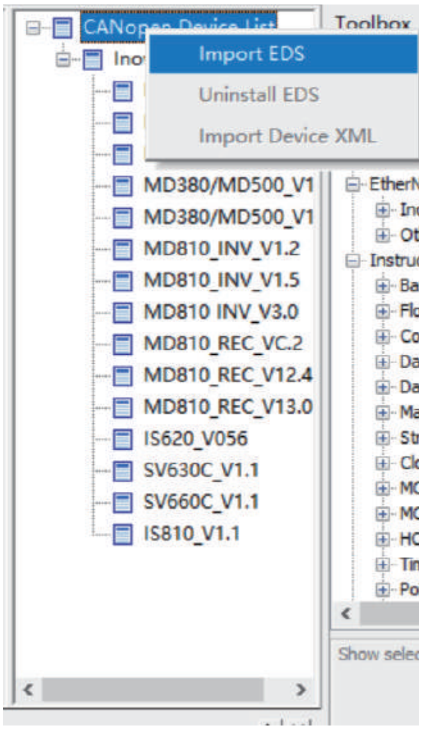


4. Double-click or drag the CANopen slave station you want to add in the device list.



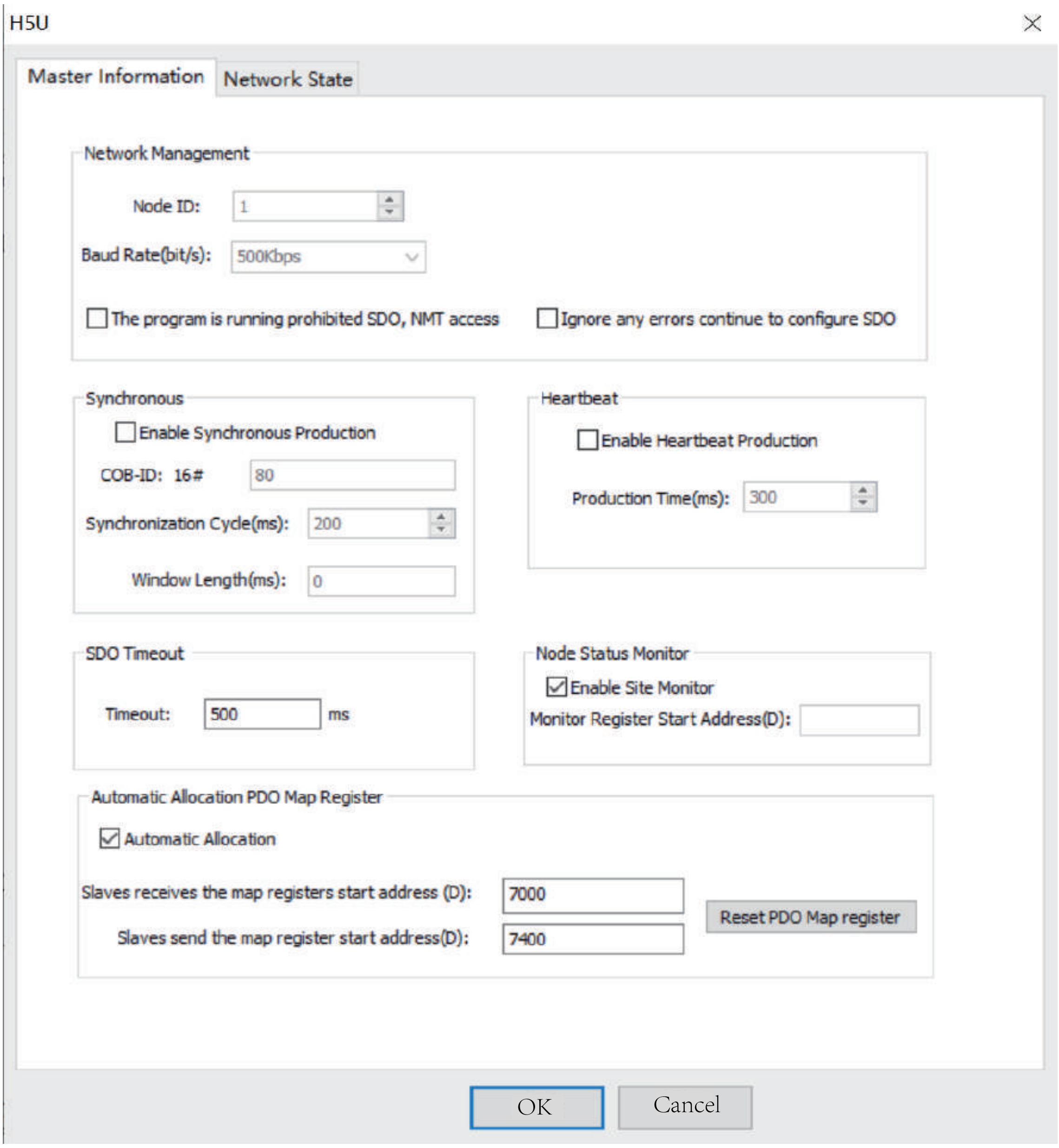
CANopen Device List
Inovance
H5U PLC
Easy PLC
H3U PLC
MD380/MD500_V1
MD380/MD500_V1
MD810_INV_V1.2
MD810_INV_V1.5
MD810 INV_V3.0
MD810_REC_VC.2
MD810_REC_V12.4
MD810_REC_V13.0
IS620_V056
SV630C_V1.1
SV660C_V1.1
IS810_V1.1

5. If the target slave station is not in the list, right-click the CANopen device list, and click "Import EDS" to import the EDS file, which can be obtained from the device supplier.



Master information page

Set the master parameters. Double-click the H5U master station in the network. The following window is displayed.



H5U

Master Information | Network State

Network Management

Node ID: 1

Baud Rate(bit/s): 500Kbps

□ The program is running prohibited SDO, NMT access

□ Ignore any errors continue to configure SDO

Synchronous

□ Enable Synchronous Production

COB-ID: 16# 80

Synchronization Cycle(ms): 200

Window Length(ms): 0

Heartbeat

□ Enable Heartbeat Production

Production Time(ms): 300

SDO Timeout

Timeout: 500 ms

Node Status Monitor

☑ Enable Site Monitor

Monitor Register Start Address(D):

Automatic Allocation PDO Map Register

☑ Automatic Allocation

Slaves receives the map registers start address (D): 7000

Slaves send the map register start address(D): 7400

- Network management

  Node ID: Indicates the master station number. If the station number is identical to the PLC number, the PLC will be initialized as the CANopen master station. If the station number is different from the PLC number, the PLC will be initialized as a CANopen slave station.

  Baud Rate: Indicates the communication baud rate valid for the master station.

  The program is running prohibited SDO, NMT access: If this option is selected, online commissioning is disabled during running of the program. The function only applies to background software.

  Ignore any errors continue to configure SDO: After this option is selected, if SDO configuration errors occur, configuration will continue. The function is valid for all slave stations. If the option is not selected, when SDO errors occur, the master station will reset slave stations through broadcasts.

- Synchronous

Enable Synchronous Production: If this option is selected, the configured station will send a sync frame repeatedly in the set synchronization cycle.

COB-ID: Indicates the ID for sync frame sending. The default value is 0x80. The parameter cannot be configured.

Synchronization Cycle (ms): Indicates the cycle for sync frame sending. The default value is 200, in the unit of ms.

Window Length (ms): The value is 0 by default. The parameter cannot be configured.

Note

Only one synchronous frame transmission can exist in one network.

- Heartbeat
  Enable Heartbeat Production: If this option is selected, the configured station will send heartbeat frames repeatedly in the set cycle.

  Production Time (ms): Indicates the cycle for heartbeat sending. The default value is 300, in the unit of ms.

Note

The default heartbeat monitoring consumption time of the master is 2.5 times the heartbeat production time. (The timeout threshold for heartbeat monitoring is 2.5 times the heartbeat production time.)

- SDO Timeout
  Timeout: Indicates the SDO waiting time. The default value is 500, in the unit of ms. SDO frames are used for network configuration. If the SDO fails to receive return frames after the third try, the master station determines that configuration times out. The waiting time for each frame is called SDO timeout.
- Node Status Monitor
  The online status of the station will be updated to the system variable _CANOpen.NodeState[64], in which _CANOpen.NodeState[0] is the status of the local station, while _CANOpen.NodeState[station No.] is the status of the slave station.

Value	State
0	Initializing
4	Stopped
5	Operational
127	Pre-operational
255	Offline

Note

If the corresponding slave does not exist, the corresponding register will not be updated. For example, if station 3 does not exist, the data of _CANOpen.NodeState[3] will not be updated.

This function works only when the heartbeat or node protection function is set on the slave, because the relevant status is fed back by the heartbeat or node protection frame of the slave.

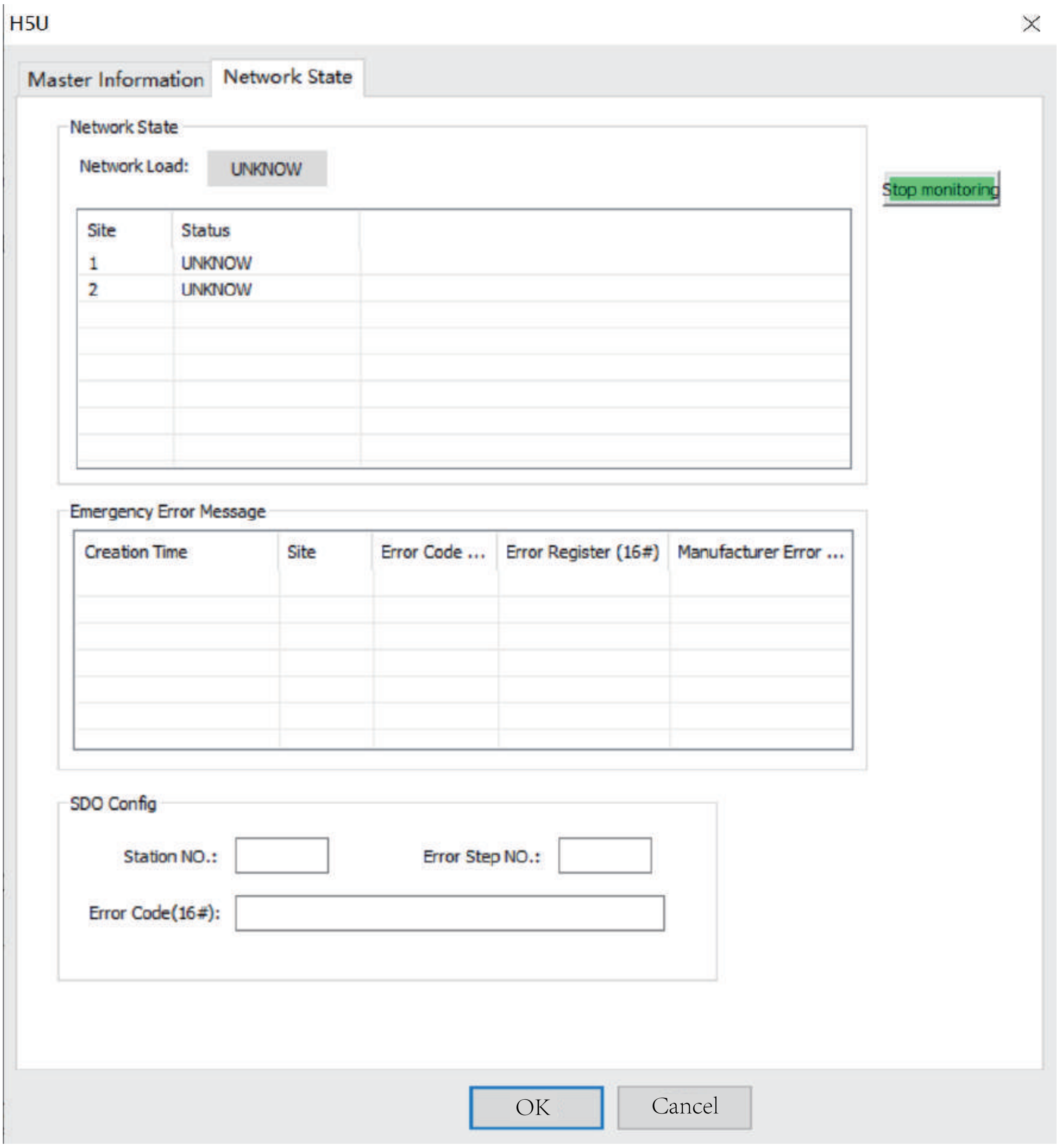
- Automatic Allocation PDO Map Register

  Automatic Allocation: If this option is selected, the system will automatically assign the address of the register for master-slave data exchange. If this option is not selected, you need to configure the start address for data exchange (by configuring the start address of each PDO). This option is selected by default.

  Slaves receive the map registers start address: Indicates the automatically assigned start address of data sent by the master station ("Automatic Allocation" must be selected).

  Slaves send the map register start address: Indicates the automatically assigned start address of data received by the master station ("Automatic Allocation" must be selected).

Network State



Network Load: UNKNOW

Site	Status
1	UNKNOW
2	UNKNOW

Creation Time	Site	Error Code ...	Error Register (16#)	Manufacturer Error ...

Station NO.:

Error Step NO.:

Error Code(16#):

Start Monitor/Stop Monitor: Information monitoring is enabled by clicking this option. Monitoring is disabled by clicking the option again.

Network Load: Monitors the network load in real time.

Network state table: Displays the station state. The table is applicable only to the master station. The state value is from the node state monitoring register.

- Emergency Error Message
  The table lists emergency error messages on the network. It is applicable only to the master station. The master PLC only caches the latest error message. If background programs are not shut down, a maximum of five messages will be cached in the background.
- SDO Config
  Station NO.: Indicates the number of the station with SDO configuration errors.

  Error Step NO.: Indicates the SDO error number. To check numbers of slave stations with parameter errors, click the SDO tab.

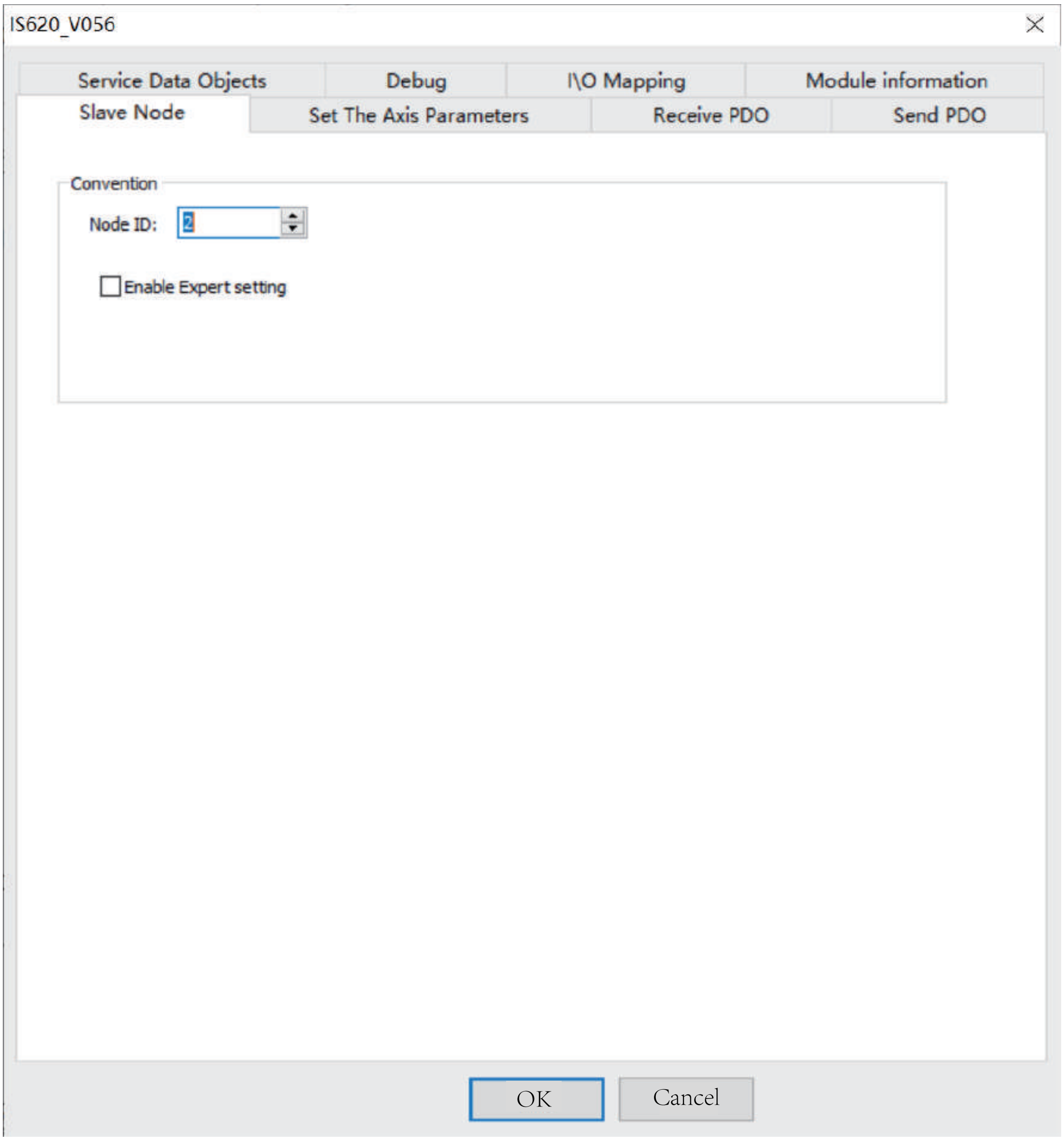
  Fault Code: Indicates the SDO fault code (standard CANopen fault code).

8.5.5.2 Slave Configuration

This section takes the IS620 slave station as an example to describe how to configure the CANopen slave station and its parameters.

General settings

Double-click a slave station in the network. The following window is displayed.

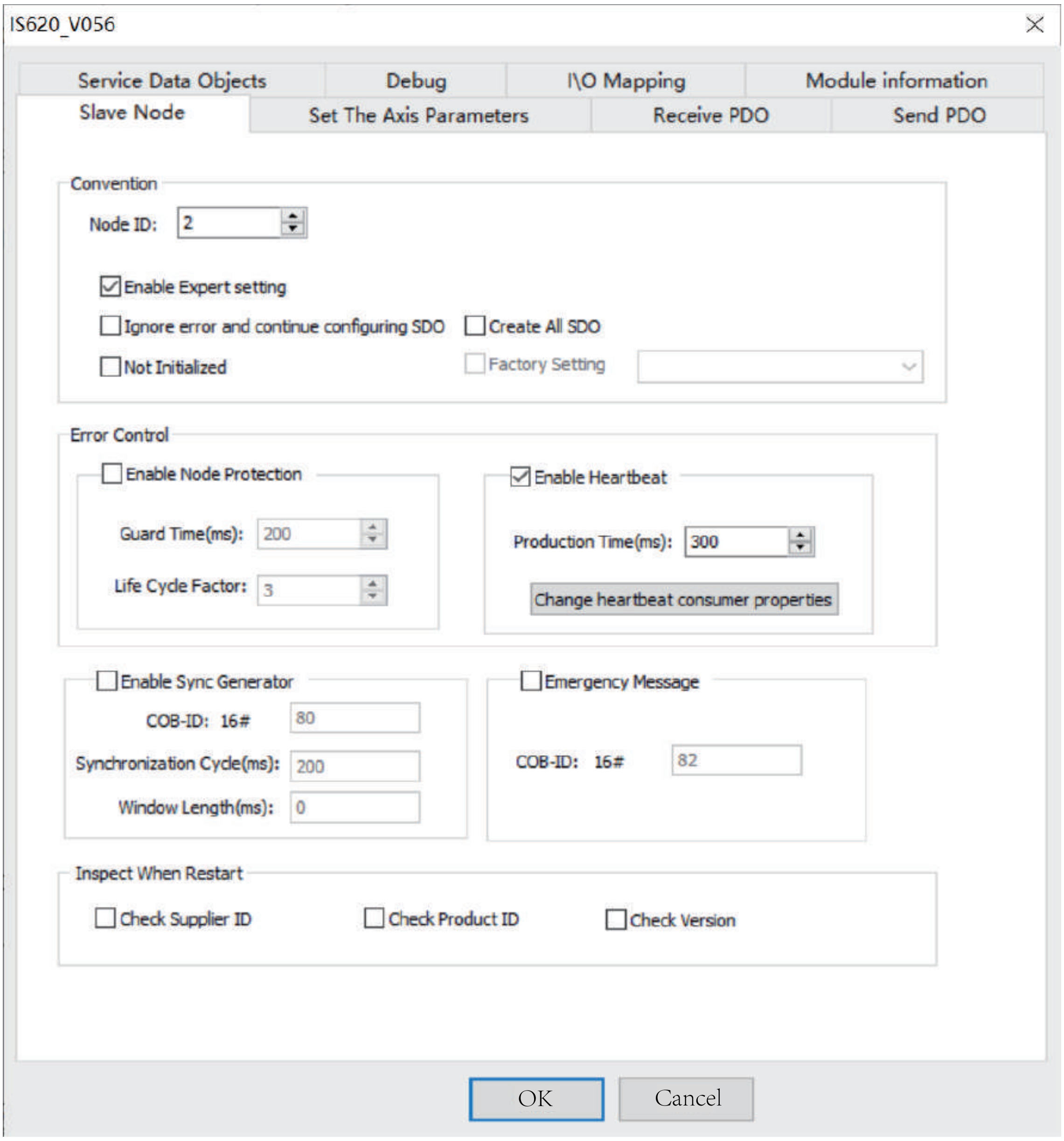


Convention

Node ID: 2

□ Enable Expert setting

Select "Enable Expert setting". The following window is displayed. (By default, this option is not selected.)



Convention

Node ID: 2

☑ Enable Expert setting

□ Ignore error and continue configuring SDO □ Create All SDO

□ Not Initialized □ Factory Setting

Error Control

□ Enable Node Protection

Guard Time(ms): 200

Life Cycle Factor: 3

☑ Enable Heartbeat

Production Time(ms): 300

Change heartbeat consumer properties

□ Enable Sync Generator

COB-ID: 16# 80

Synchronization Cycle(ms): 200

Window Length(ms): 0

□ Emergency Message

COB-ID: 16# 82

Inspect When Restart

□ Check Supplier ID □ Check Product ID □ Check Version

- Convention
  Node ID: Indicates the ID of a slave station node.

  Enable Expert setting: When this option is selected, detailed configurations of the slave station are displayed. By default, this option is not selected.

- Ignore error and continue configuring SDO
  Valid: When a configuration error (other than a check type error) occurs, configuration continues.

  Invalid: When a configuration error occurs, configuration cannot continue, and the entire network is disconnected. By default, this option is not selected.

- Create All SDO
  If this option is selected, all writable object dictionaries in the EDS will be added and initialized. By default, this option is not selected.

- Not Initialized

If this option is selected, the slave station will not be initialized (this option can be selected only when the station applies the default configuration). By default, this option is not selected.

- Factory Setting
  If this option is selected, you can select options from the drop-down list. By default, the option is not selected.

Error Control

- Node protection properties
  Enable Node Protection: If this option is selected, node protection will be enabled. By default, the option is not selected.

  Node protection timeout = Guard time x Life cycle factor

  Node protection provides a network evaluation platform on which master station and slave station monitor each other with return frames. Either the heartbeat or node protection function can be selected.

  Guard Time (ms): Indicates the node protection time, which is 200 ms by default.

  Life Cycle Factor: Indicates the node protection factor, which is 3 by default.

- Heartbeat properties
  Enable Heartbeat: If this option is selected, heartbeats will be generated. By default, this option is selected. When this option is selected, the master station will monitor the heartbeat state by default.

  Production Time (ms): Indicates the cycle for heartbeat sending.

  Change heartbeat consumer properties: It is used to set heartbeats of other stations to be monitored by the configured station. This function is disabled by default. The function can be enabled only when the slave station supports heartbeat monitoring.

- Synchronous (if supported)
  Enable Synchronous Production: If this option is selected, the configured station will send a sync frame repeatedly in the set synchronization cycle.

  COB-ID: Indicates the ID for sync frame sending. The default value is 0x80. The parameter cannot be configured.

  Synchronization Cycle (ms): Indicates the cycle for sync frame sending. The default value is 200, in the unit of ms.

  Window Length (ms): The value is 0 by default. The parameter cannot be configured.

Note

Only one synchronous frame transmission can exist in one network.

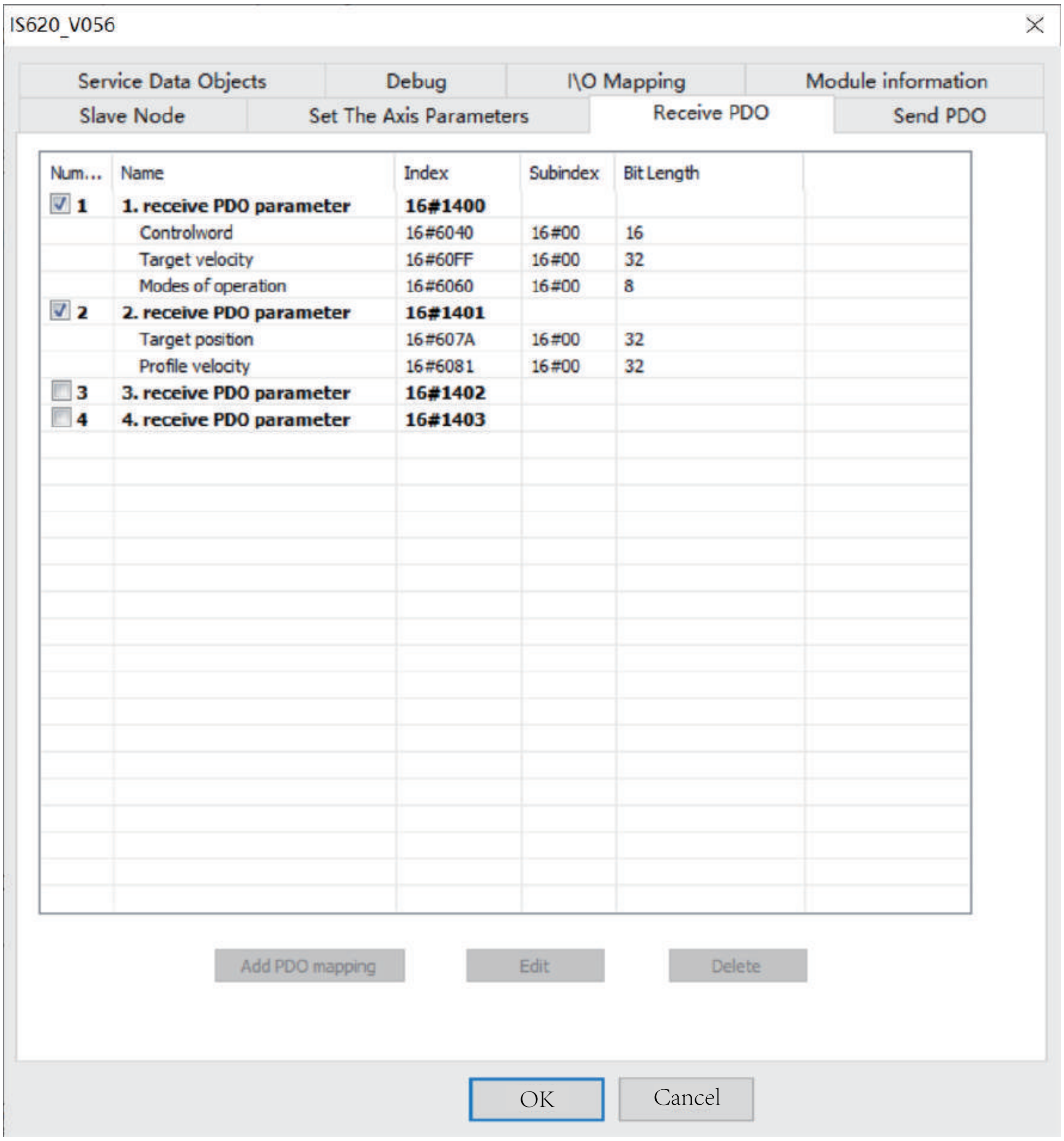
- Emergency Message
  Emergency Message: If this option is selected, you can set the COB-ID of an emergency message. By default, this option is not selected.

- Inspect When Restart

If "Check Supplier ID", "Check Product ID", or "Check Version" is selected, corresponding data will be checked before configuration of the slave station. If the check fails, the network cannot be connected.

Receive PDO/Send PDO

Click "Receive PDO" or "Send PDO". The following page is displayed.



Num...	Name	Index	Subindex	Bit Length
☑ 1	**1. receive PDO parameter**	**16#1400**		
	Controlword	16#6040	16#00	16
	Target velocity	16#60FF	16#00	32
	Modes of operation	16#6060	16#00	8
☑ 2	**2. receive PDO parameter**	**16#1401**		
	Target position	16#607A	16#00	32
	Profile velocity	16#6081	16#00	32
☐ 3	**3. receive PDO parameter**	**16#1402**		
☐ 4	**4. receive PDO parameter**	**16#1403**		

Receive PDO: Indicates the data sent by the master station to a slave station.

Send PDO: Indicates the data sent by a slave station to the master station.

8.5.5.3 PDO Enable

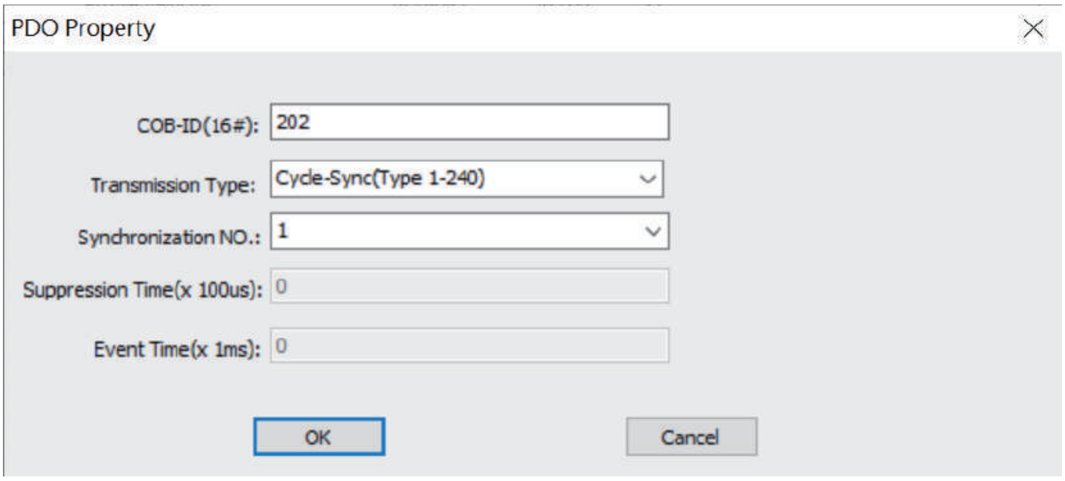
You can check the box in front of the number to enable a PDO. The PDOs in the EDS file that take effect are checked by default.

8.5.5.4 PDO Mapping Edit

You can click "Add PDO mapping", "Edit", or "Delete" to edit PDO mapping.

8.5.5.5 PDO Property Settings

The "PDO Property" page is as follows.



PDO Property

COB-ID(16#): 202

Transmission Type: Cycle-Sync(Type 1-240)

Synchronization NO.: 1

Suppression Time(x 100us): 0

Event Time(x 1ms): 0

- COB-ID
  Indicates the ID for sending a PDO parameter. Based on the CANopen DS301 protocol, default COB-IDs are available for the first four PDO parameters. COB-IDs must be different from each other, ranging from 0x180 to 0x57F.

- Transmission Type

Type	Condition for Data Sending	Condition for Valid Data
Loop-synchronization (Type 0)	Data is changed, and a sync frame is received.	Data does not take effect immediately but takes effect after a sync frame is received.
Loop-synchronization (Types 1 to 240)	Data is sent after the corresponding "number of synchronizations" frame is received.	Data does not take effect immediately but takes effect after a sync frame is received.
Asynchronization-only RTR (Type 252)	Not supported	Not supported
Asynchronization-only RTR (Type 253)	Not supported	Not supported
Asynchronization-specified by manufacturers (Type 254)	Manufacturer-defined	Manufacturer-defined
Asynchronization-specified by the configuration file (Type 255)	Data is changed or the event time is correct, and the change cycle is shorter than the suppression time.	Immediately

Note

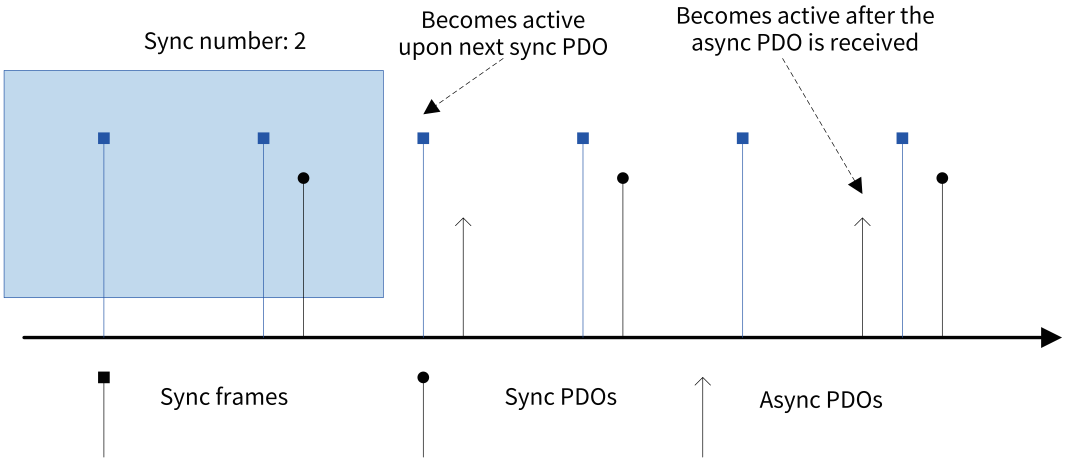
To use the synchronous type, it is necessary to enable synchronous production on a station, usually the master.

- Synchronization NO.

The number of synchronizations takes effect after "loop-synchronization (types 1 to 240)" is selected.

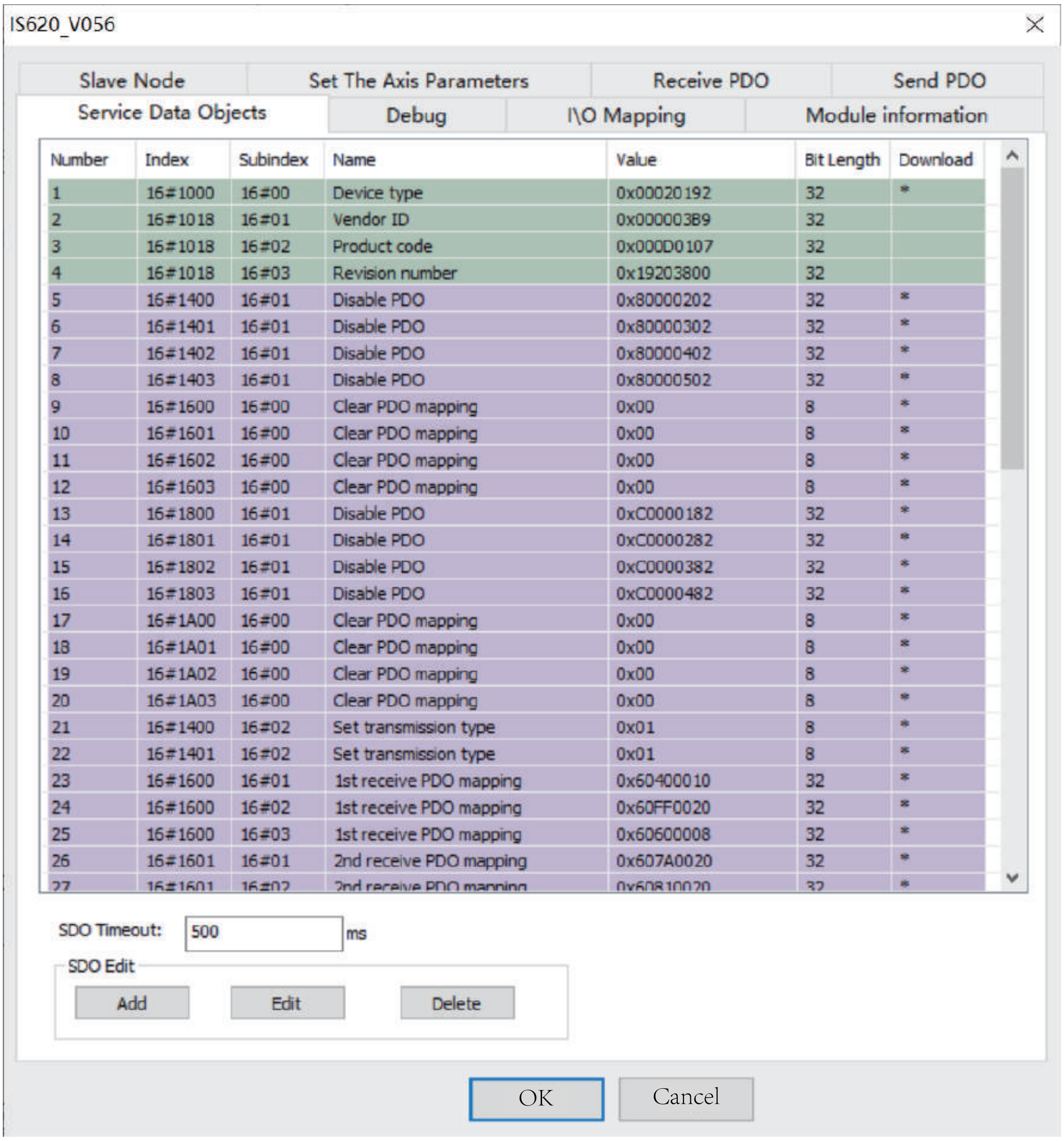
- Suppression Time
  The suppression time can be set after "asynchronization-specified by the configuration file (Type 255)" is selected. If the value is 0, the function is disabled. If the value is not 0, the suppression time is the minimum interval for frame sending.
- Event Time
  The event time can be set after "asynchronization-specified by the configuration file (Type 255)" is selected. If the value is 0, the function is disabled. If the value is not 0, the event time is the cycle for data sending. (Data sending is limited by the suppression time.)

The following figure shows the example of loop-synchronization (Type 2).


Sync number: 2
Becomes active upon next sync PDO
Becomes active after the async PDO is received
Sync frames
Sync PDOs
Async PDOs

8.5.5.6 Service Data Object (SDO)

Click the "Service Data Objects" tab. The following page is displayed.



Number	Index	Subindex	Name	Value	Bit Length	Download
1	16#1000	16#00	Device type	0x00020192	32	*
2	16#1018	16#01	Vendor ID	0x000003B9	32	
3	16#1018	16#02	Product code	0x000D0107	32	
4	16#1018	16#03	Revision number	0x19203800	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	0xC0000182	32	*
14	16#1801	16#01	Disable PDO	0xC0000282	32	*
15	16#1802	16#01	Disable PDO	0xC0000382	32	*
16	16#1803	16#01	Disable PDO	0xC0000482	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	Set transmission type	0x01	8	*
22	16#1401	16#02	Set transmission type	0x01	8	*
23	16#1600	16#01	1st receive PDO mapping	0x60400010	32	*
24	16#1600	16#02	1st receive PDO mapping	0x60FF0020	32	*
25	16#1600	16#03	1st receive PDO mapping	0x60600008	32	*
26	16#1601	16#01	2nd receive PDO mapping	0x607A0020	32	*
27	16#1601	16#02	2nd receive PDO mapping	0x60810020	32	*

SDO Timeout: 500 ms

The table lists SDO configurations automatically generated based on user settings.

SDO Edit

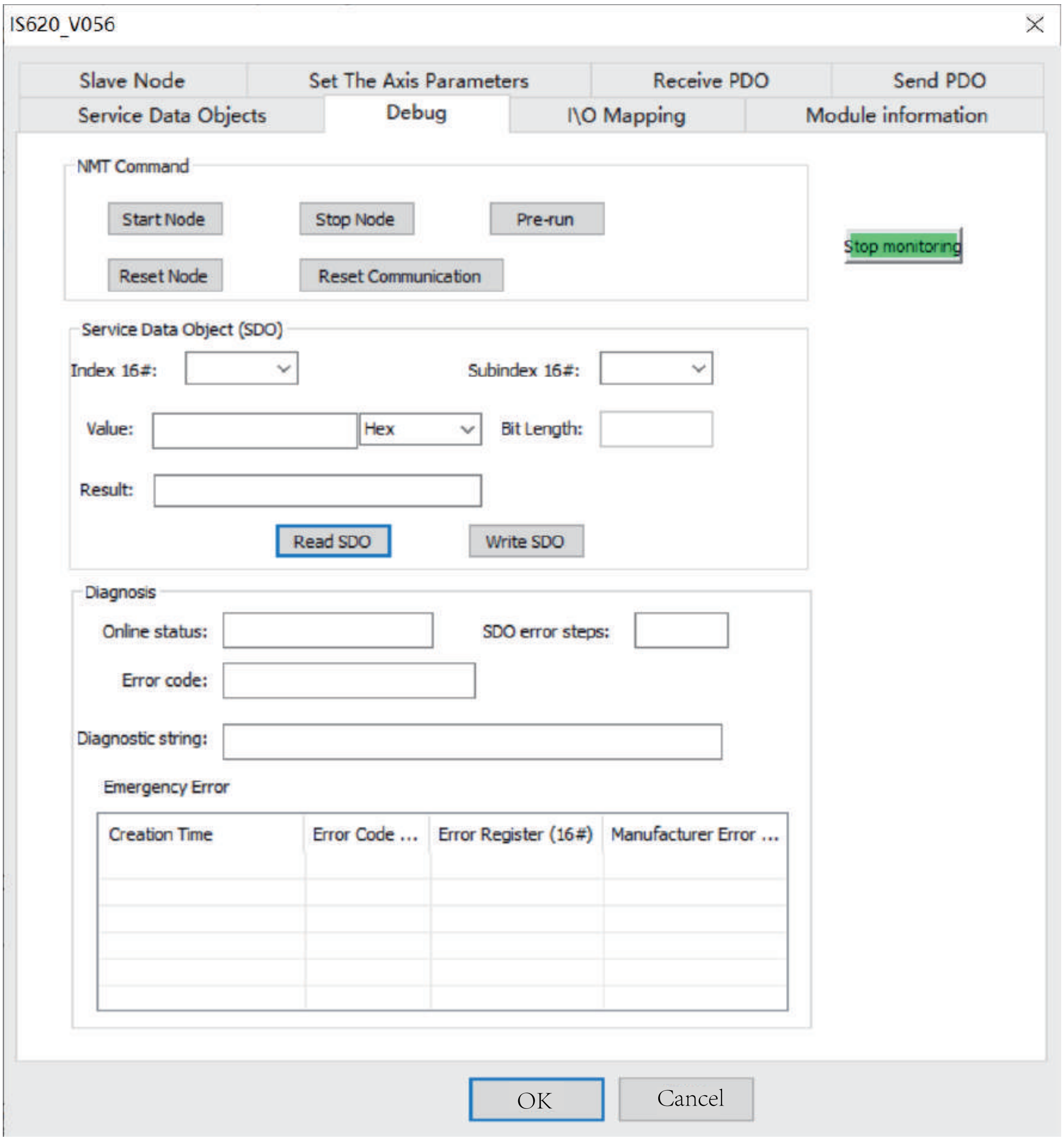
Add: Adds configurations. It is used to assign initial values to object dictionaries of a slave station.

Edit: Edits configurations.

Delete: Deletes configurations.

8.5.5.7 Online Commissioning

Click the "Debug" tab. The following page is displayed.



Creation Time	Error Code ...	Error Register (16#)	Manufacturer Error ...

Note

This function cannot be used if “The program is running, prohibited SDO, NMT access” is selected in the master.

- NMT Command

  Start Node: Sends a command to the slave station to start a node.

  Stop Node: Sends a command to the slave station to stop a node.

  Pre-run: Sends a command to the node to pre-run it.

  Reset Node: Sends a command to the node to reset it.

  Reset Communication: Sends a command to the node to reset communication.

- Service Data Object

  Index and sub-index: You can only select object dictionaries in the EDS as indexes or sub-indexes.

Value: Indicates sent or returned data.

Bit Length: It is automatically generated based on an object dictionary in the EDS. It must not be modified.

Result: Indicates abnormality information.

Read SDO and Write SDO: Reads and writes object dictionaries.

- Diagnosis
  Online status: Indicates the status of the slave station (fed back based on heartbeat or node protection).

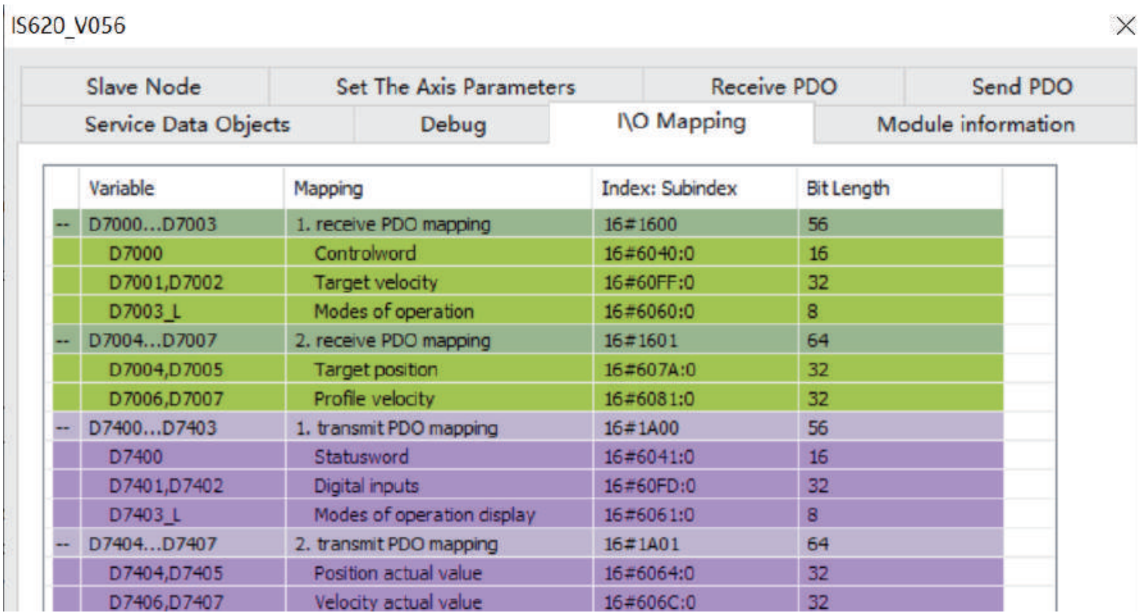
  SDO error steps: Indicates the SDO error number. This number corresponds to the "Service Data Objects" tab.

  Diagnostic string: Indicates the error message (SDO error).

  Emergency Error Information: Indicates an emergency error frame (the system monitors real-time errors and caches five error messages in the background; the PLC only caches the latest error message) (emergency error).

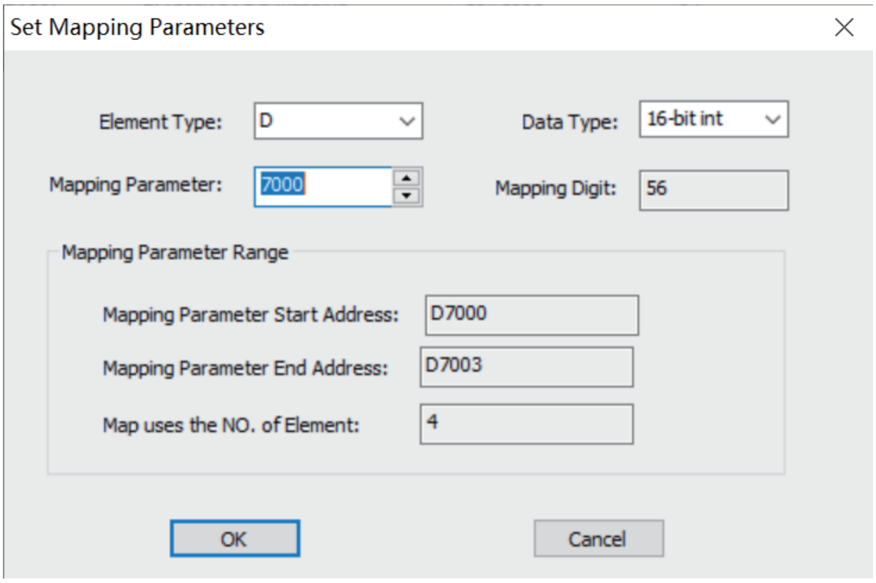
8.5.5.8 I/O Mapping

The "IO Mapping" tab page is as follows.



	Variable	Mapping	Index: Subindex	Bit Length
--	D7000...D7003	1. receive PDO mapping	16#1600	56
	D7000	Controlword	16#6040:0	16
	D7001,D7002	Target velocity	16#60FF:0	32
	D7003_L	Modes of operation	16#6060:0	8
--	D7004...D7007	2. receive PDO mapping	16#1601	64
	D7004,D7005	Target position	16#607A:0	32
	D7006,D7007	Profile velocity	16#6081:0	32
--	D7400...D7403	1. transmit PDO mapping	16#1A00	56
	D7400	Statusword	16#6041:0	16
	D7401,D7402	Digital inputs	16#60FD:0	32
	D7403_L	Modes of operation display	16#6061:0	8
--	D7404...D7407	2. transmit PDO mapping	16#1A01	64
	D7404,D7405	Position actual value	16#6064:0	32
	D7406,D7407	Velocity actual value	16#606C:0	32

This tab is used to set the communication relationship between master and slave PDOs. If "Automatic Allocation" is not selected, when you double-click an item, the following page is displayed.



Set Mapping Parameters

Element Type: D

Data Type: 16-bit int

Mapping Parameter: 7000

Mapping Digit: 56

Mapping Parameter Range

Mapping Parameter Start Address: D7000

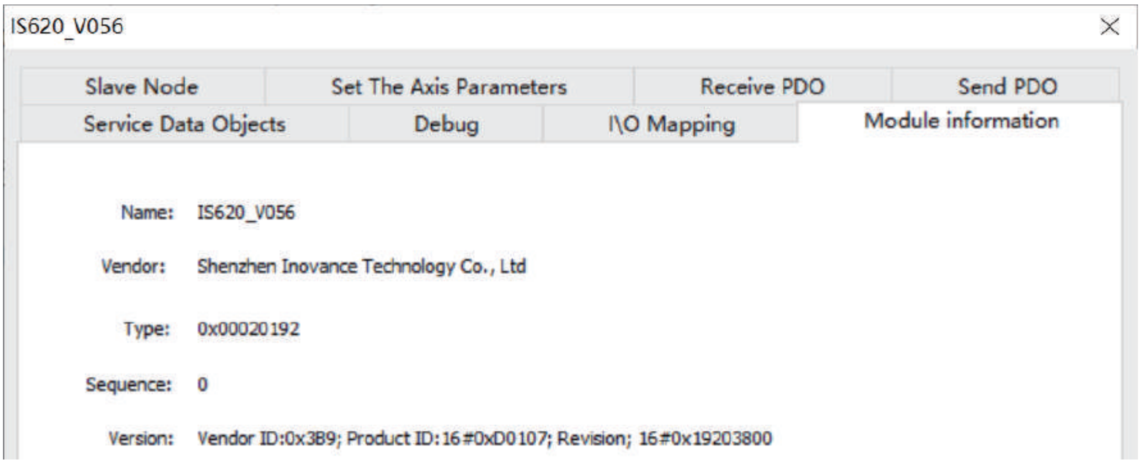
Mapping Parameter End Address: D7003

Map uses the NO. of Element: 4

You can configure the start register address for the master station corresponding to a slave PDO.

8.5.5.9 Device Information

The "Module information" tab page is as follows.



IS620_V056

Name: IS620_V056

Vendor: Shenzhen Inovance Technology Co., Ltd

Type: 0x00020192

Sequence: 0

Version: Vendor ID:0x3B9; Product ID:16#0xD0107; Revision; 16#0x19203800

Device information can be obtained from the EDS file.

8.5.6 CANopen Communication Troubleshooting

8.5.6.1 General Troubleshooting Steps

- Check the termination resistor
  Power off all devices. Use a multimeter to measure the resistance between CANH and CANL. The resistance should be about 60 Ω. If the resistance is too small, there are termination resistors incorrectly connected at other locations. In this case, disconnect these termination resistors. If only one termination resistor is available, the resistance is about 120 Ω, and the network connection is bad. If no termination resistor is available, communication fails. Provide termination resistors between the stations at both ends of the network.
- Check the baud rate

Check whether the baud rate is normal. Baud rates of all stations in the network must the same; otherwise, communication fails. Power off and then on the device or switch it from STOP to RUN so that the baud rate can take effect.

For the relationship between communication distance and baud rate, see *"8.3.2 Relationship Between CAN Communication Distance and Baud Rate" on page 263Relationship Between CAN Communication Distance and Baud Rate*.

- Check cable connections
  Interconnect CGND pins of all CAN devices to ensure that all devices share one power supply CGND port of CAN communication.

  Check whether the communication cable, shielded cable, and power supply are short-circuited.

- Others
  In case of strong interference, reduce the baud rate.

8.5.6.2 Fault Code List

SDO fault codes

Abort Code	Description	Abort Code	Description
0503 0000	Trigger bit not alternated	0601 0002	Attempt to write a read-only object
0504 0000	SDO protocol timed out	0602 0000	Object not exist in the object dictionary
0504 0001	Invalid or unknown Client/Server command word	0604 0041	Object cannot be mapped to the PDO
0504 0002	Invalid block size (for the Block Transfer mode only)	0604 0042	The number and length of the objects to be mapped exceed the PDO length
0504 0003	Invalid serial number (for the Block Transfer mode only)	0604 0043	General parameter incompatibility
0503 0004	CRC error (for the Block Transfer mode only)	0604 0047	General internal incompatibility in the device
0503 0005	Memory overflow	0606 0000	Access to an object failed due to a hardware error
0601 0000	Access to an object unsupported	0606 0010	Data type does not match. Length of service parameters does not match
0601 0001	Attempt to read a write-only object	0606 0012	Data type does not match. Length of service parameters too high
0601 0002	Attempt to write a read-only object	0606 0013	Data type does not match. Length of service parameters too short
0602 0000	Object not exist in the object dictionary	0609 0011	Sub-index does not exist
0604 0041	Object cannot be mapped to the PDO	0609 0030	Beyond the value range (for write access)
0503 0000	Trigger bit not alternated	0609 0031	Value of parameter written too large
0504 0000	SDO protocol timed out	0609 0032	Value of parameter written too small
0504 0001	Invalid or unknown Client/Server command word	0609 0036	Maximum value is less than minimum value
0504 0002	Invalid block size (for the Block Transfer mode only)	0800 0000	General error
0504 0003	Invalid serial number (for the Block Transfer mode only)	0800 0020	Data cannot be transmitted or stored to the application

Abort Code	Description	Abort Code	Description
0503 0004	CRC error (for the Block Transfer mode only)	0800 0021	Data cannot be transmitted or stored to the application due to local control
0503 0005	Memory overflow	0800 0022	Data cannot be transmitted or stored to the application due to current device state
0601 0000	Access to an object unsupported	0800 0023	Object dictionary dynamic generation fails or no object dictionary is available (for example, an object dictionary is generated through a file, but an error occurs because the file is corrupted)
0601 0001	Attempt to read a write-only object		

Emergency fault codes

Table 8–8 Main table 1 (hexadecimal)

Emergency Fault Code	Description	Emergency Fault Code	Description
00xx	No error	50xx	Device hardware
10xx	General error	60xx	Device software
20xx	Current	61xx	Internal software
21xx	Current at input end	62xx	User software
22xx	Internal current	63xx	Data setting
23xx	Current at output end	70xx	Extra module
30xx	Voltage	80xx	Monitoring
31xx	Power voltage	81xx	Communication
32xx	Internal voltage	82xx	Protocol error
33xx	Output voltage	90**	External error
40xx	Temperature	F0**	Extra function
41xx	Ambient temperature	FF**	Special device
42xx	Device temperature		

Table 8–9 Table 2 (hexadecimal)

Emergency Fault Code	Description	Emergency Fault Code	Description
0000	Incorrect reset or no error	6300	Data setting
1000	General error	7000	Extra module error
2000	Current error	8000	Monitoring error
2100	Input current	8100	General communication error
2200	Internal current	8110	CAN communication overload
2300	Output current	8120	Incorrect CAN passive method
3000	Voltage error	8130	Node protection or heartbeat error
3100	Power voltage	8140	Bus disconnection
3200	Internal voltage	8150	CAN-ID impulse
3300	Output voltage	8200	Protocol error
4000	Temperature error	8210	PDO length error
4100	Ambient temperature	8220	Excessive PDO length
4200	Device temperature	8240	Unidentifiable SYNC data length
5000	Device hardware error	8250	RPDO timeout

Emergency Fault Code	Description	Emergency Fault Code	Description
6000	Device software error	9000	External error
6100	Internal software	F000	Extra function error
6200	User software	FF00	Special device error

9 EtherCAT Communication

9.1 Overview

EtherCAT is an open industrial field technology based on Ethernet. It features short communication refresh cycles, low synchronization jitter, and low hardware costs. For details about EtherCAT principles and related technologies, see the book "Industrial Ethernet Fieldbus EtherCAT Driver Design and Applications" or visit the official website of the EtherCAT Technical Group at **https://www.EtherCAT.org.cn**.

The H5U and Easy500 series support standard EtherCAT ports (one RJ45 port). In the linear topology, they support a maximum of 72 EtherCAT slave stations and a minimum EtherCAT bus cycle of 1 ms.

Table 9–1 EtherCAT port specifications

Item	Specifications
Transmission rate	100 Mbps: 100BASE-TX
Modulation	Baseband
Topology	Linear and daisy chain
Medium	Cat5 twisted pairs or shielded twisted pairs with aluminum foil and braided mesh
Transmission distance	Distance between nodes: 100 m or less
Number of connections	72

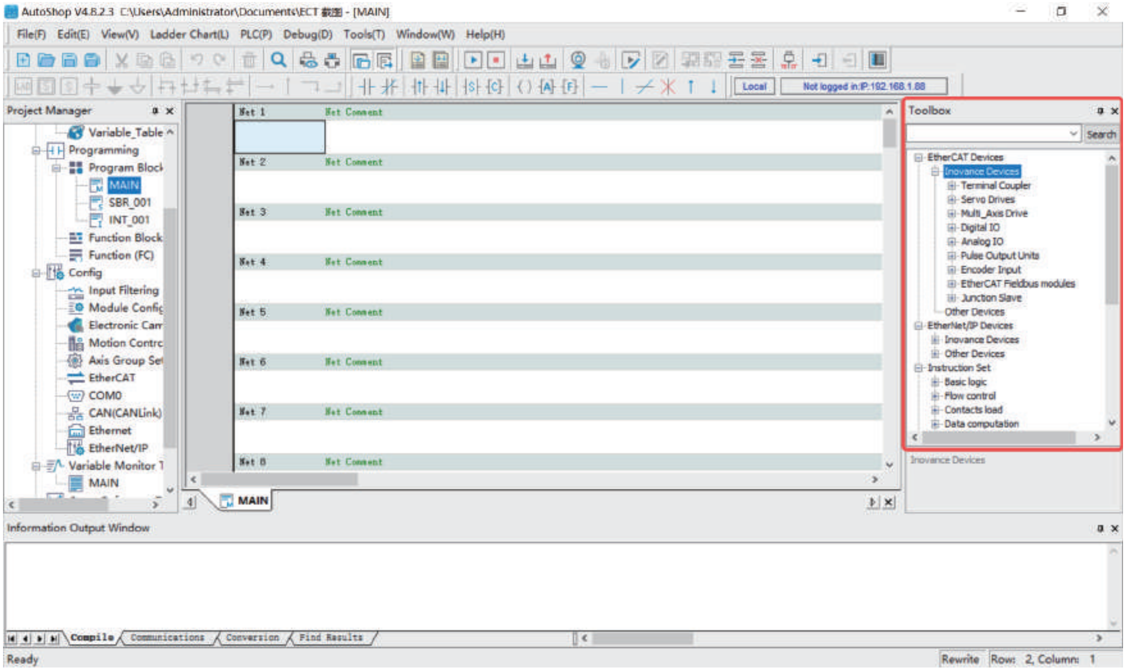
9.2 Master Configuration

9.2.1 Importing Device Description (XML)

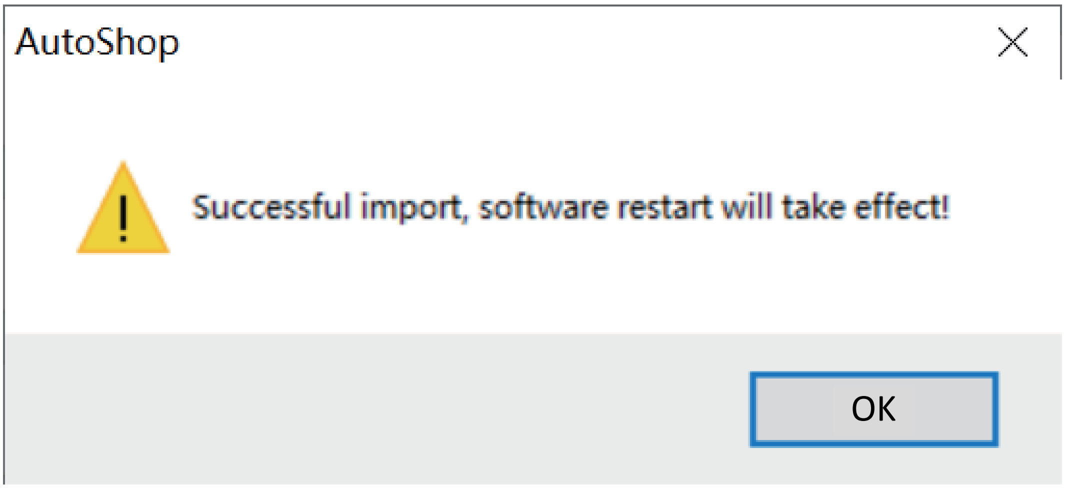
Importing device XML means importing the device description file with the suffix ".XML" that meets standards of the EtherCAT Technical Group (ETG) into the programming software AutoShop, in which, the file is parsed into EtherCAT configuration devices that can be added or deleted by users.

AutoShop provides built-in EtherCAT slave stations of Inovance, and therefore the device description files do not need to be installed. To use third-party EtherCAT devices, their description files must be installed. The following section takes importing the description file of Inovance bus motor drive SV520N as an example.

1. Create a project, open the toolbox, and locate "EtherCAT Devices".



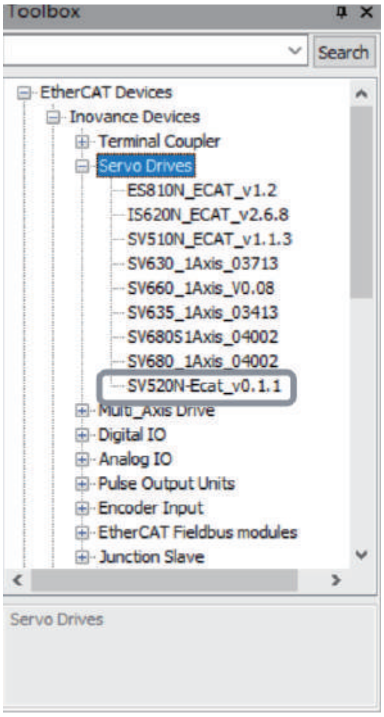
2. Right-click "EtherCAT Devices". In the dialog box displayed, select and import the target XML file.
3. Restart the software to make the imported XML file take effect. To import the XML files of multiple devices, repeat Step 2 and then restart the device.



Successful import, software restart will take effect!

You must manually restart the software before the added devices take effect.

4. The added devices are added to the list after the software is reaccessed.



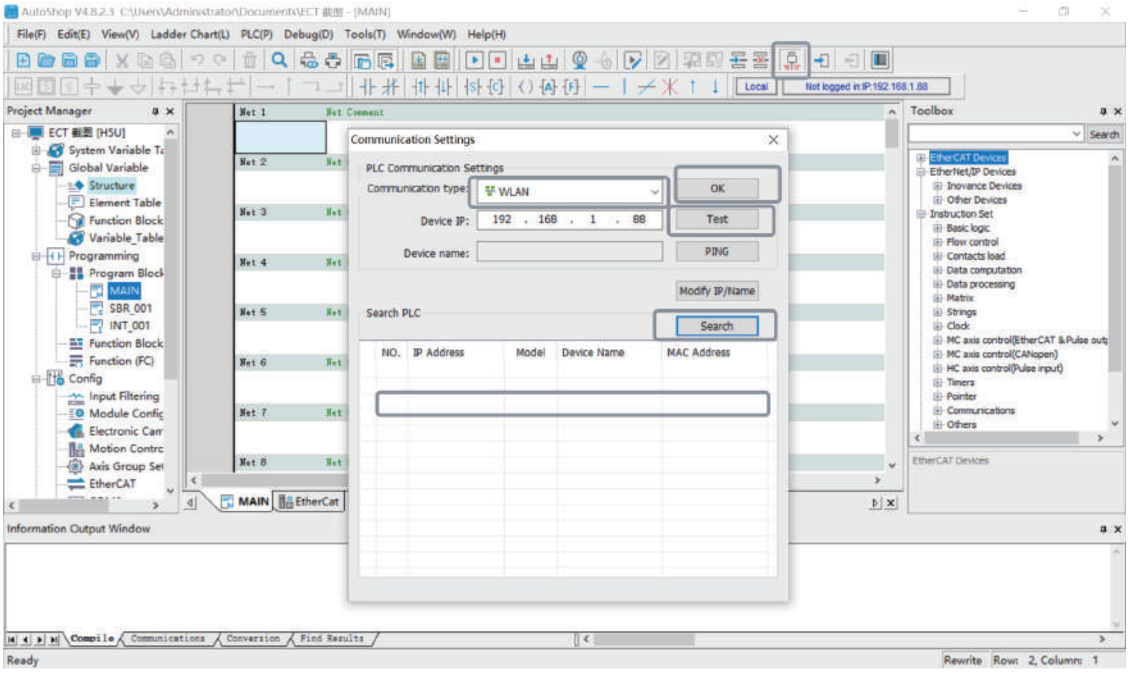
9.2.2 Scanning Devices

The following section describes how to scan a device when the communication mode is Ethernet.

Note

EtherCAT slaves can only be scanned when the PLC is in the stopped state.

1. Select the target host.


Communication Settings

PLC Communication Settings

Communication type: WLAN

Device IP: 192 . 168 . 1 . 88

Device name:

① Click "Test communication status".

② On the page displayed, click "Search".

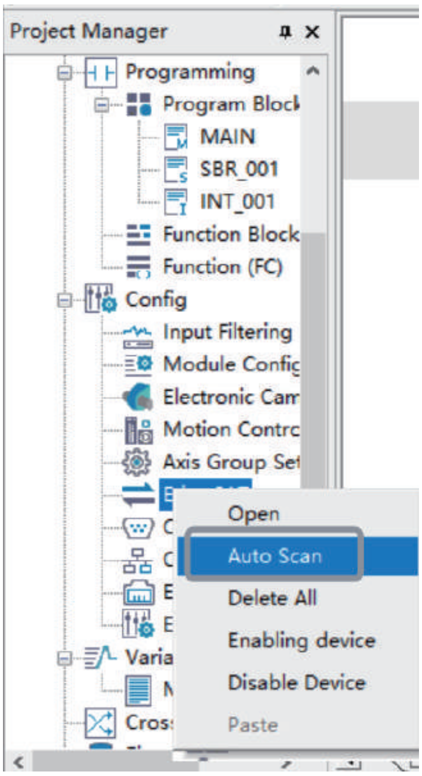
③ Select the target host.

④ Click "Test".

⑤ After confirming that the host is connected, click "OK".

2. Determine whether to automatically associate the motion control axis.
   If "Automatically create axes and associate slaves when creating new slaves" is selected, each time an EtherCAT slave station of the drive type (such as IS620N) is added, a motion control axis is automatically added. In this case, this option is not selected.

3. Right-click EtherCAT and then select "Auto Scan".



4. In the dialog box displayed, click "Start Scan". If the PLC is running, click "Yes" to stop the PLC first.

5. When scanning is completed, the found slave stations are displayed. Click "Update Config" to add the found devices to the configuration list, or click "Exit" to not add the found devices to the configuration list.
   After the configuration list is updated, the following figure is displayed and IS620N is automatically associated with the motion control axis.

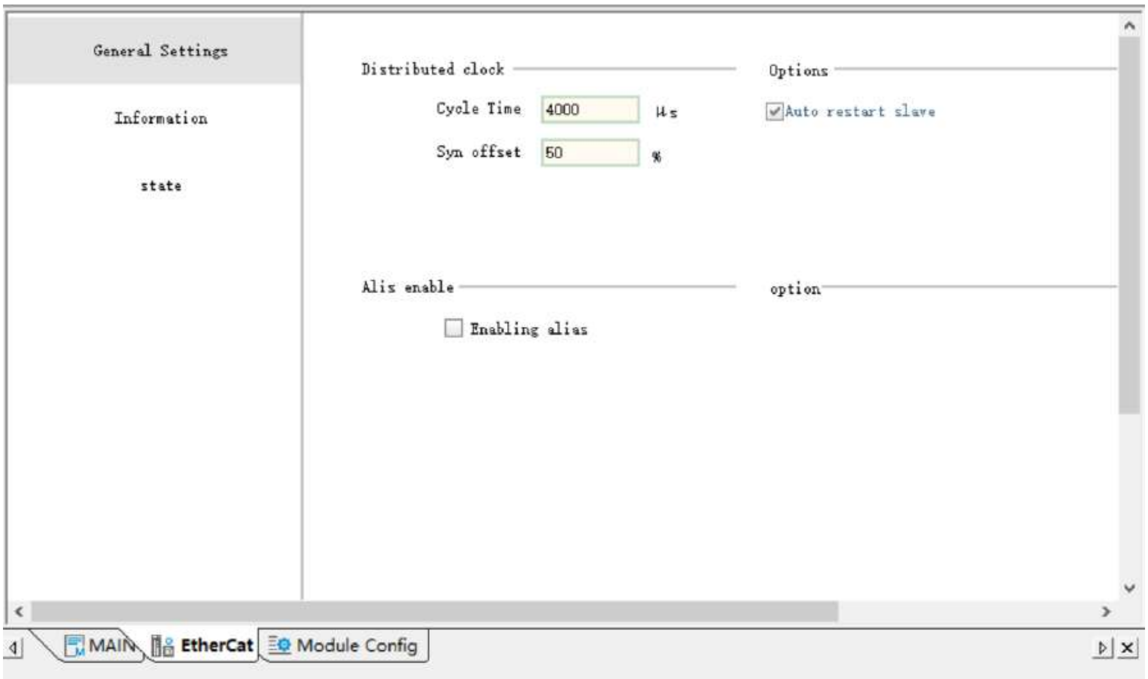
   When the IN/OUT pins of the slave station are reversed, the message "IN/OUT port connection error" is displayed in the "Message" column. In this case, the "Update Config" button is grayed out, and you need to manually check the connection of the physical link and perform scanning again.

Note

The IN/OUT reversal detection is available to H5U with the firmware version of V6.0 or later. This function is unavailable to the Easy series.

9.2.3 Master Configuration

On the "Normal setting" page, set parameters shown in the following figure.



Distributed clock

Cycle Time 4000 μs

Syn offset 50 %

Options

☑ Auto restart slave

Alis enable

☐ Enabling alias

option

Parameter Name	Description
Cycle time	Indicates the EtherCAT data frame sending interval and the EtherCAT task cycle time.
Synchronization offset	Indicates the relative offset (in percentage) of the EtherCAT task with respect to the Sync0 interrupt of the slave station.
Automatic restart of slave	When this option is selected, the EtherCAT slave station is automatically restarted when it attempts to connect to the network after disconnection. **Note: This function is supported by AutoShop 4.0.0.0 matching the PCB software 3.0.0.0.**
Alias enable	After this option is selected, the aliases of all EtherCAT slave stations are enabled. In this case, even "Alias enable" is not selected for some slave stations, such slave stations are stilled started with their aliases. When the network contains branch modules, you are recommended to select this option. If "Alias enable" is not selected for the master station, when you select "Alias enable" for a single slave station, the alias mode is applied only to this slave station. Then, aliases are mixed with formal names, causing a communication error.

9.2.4 Start/Stop, Disable, and Enable

Start/Stop control

You can start or stop an EtherCAT bus but cannot start or stop a single slave station. The procedure is as follows.

When the PLC state changes from STOP to RUN, the EtherCAT master station starts to run automatically.

When the PLC state changes from RUN to STOP, the EtherCAT master station stops automatically.

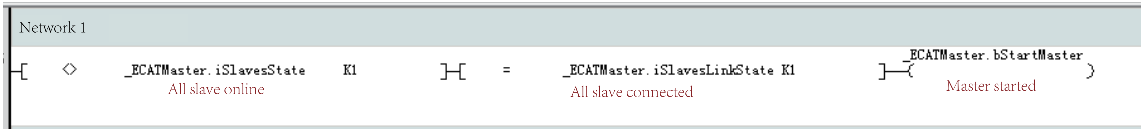
When the PLC is running, you can use the system variables to start or stop the EtherCAT master station.

Note Only the startup and stop of EtherCAT bus are supported, while the startup and stop of individual slaves are not supported.

System Variable	Data Type	Function
bStopMaster	BOOL	Stop of the EtherCAT master station The EtherCAT master station stops at the rising edge of the variable input and then the variable is automatically reset.
bStartMaster	BOOL	Startup of the EtherCAT master station When the EtherCAT bus fails or stops, the EtherCAT master station is restarted at the rising edge of the variable input and then the variable is automatically reset.

Auto Restart

For AutoShop 4.0.0.0 matching PCB software of a version earlier than 3.0.0.0, you can run the PLC program together with system variables iSlavesState, iSlavesLinkState, and bStartMaster to automatically restart the EtherCAT bus.

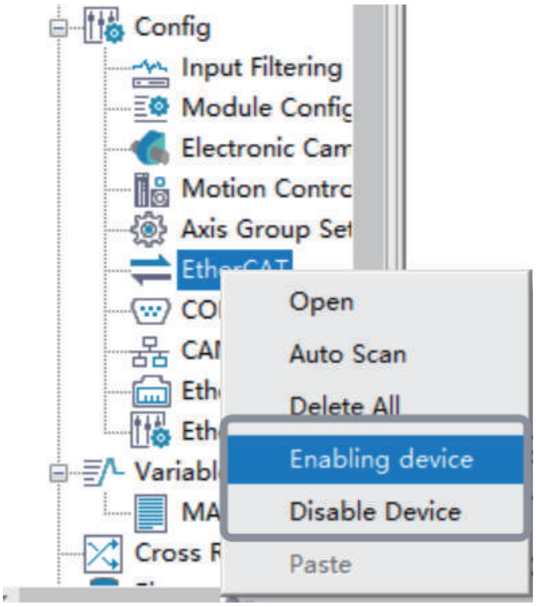


For AutoShop 4.0.0.0 matching PCB software of version 3.0.0.0 or later, you can select "Auto restart of slave" on the "Normal setting" tab page to automatically restart the slave station.

Disable and Enable

When "Motion control axis-Virtual axis mode" is selected, even if you disable the EtherCAT master station (all the master and slave stations under this bus are disabled), you can still control the motion control axis through the program. For details, see description of the virtual axis mode of the motion control axis. Note that after the EtherCAT master station is disabled, all EtherCAT slave stations are disabled and bus servo axes associated with the slave stations cannot move.

Right-click the EtherCAT master station, and then select "Enabling device" or "Disable Device".

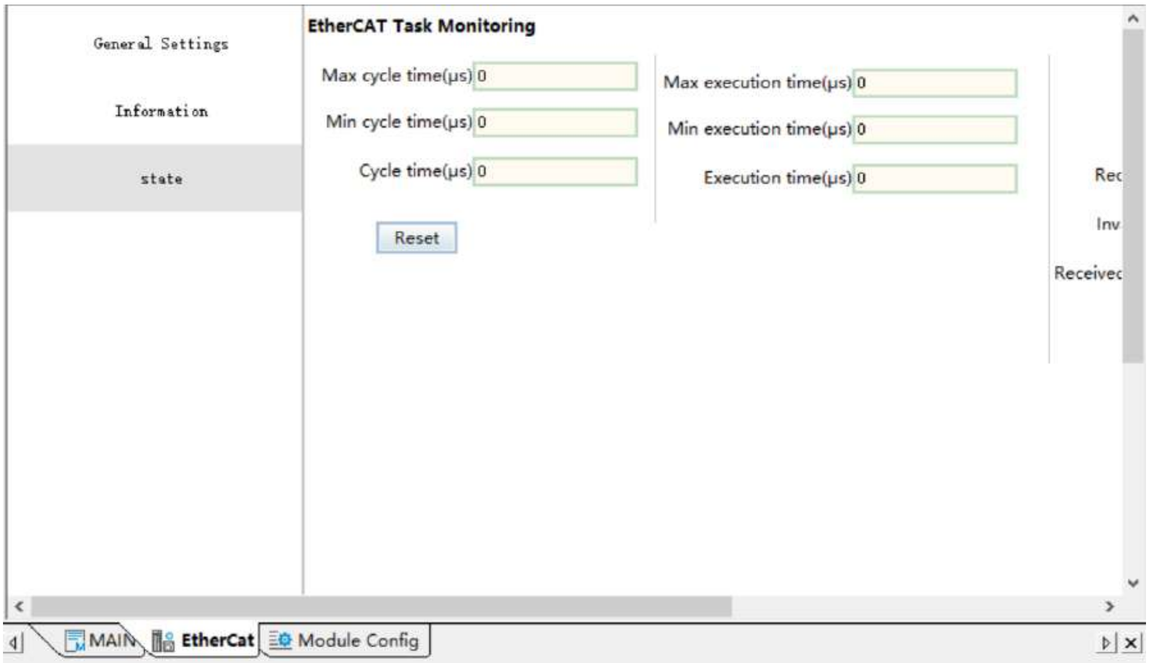


Caution

A disabled slave must be removed from the actual physical link. Otherwise, the slave will affect the startup of all subsequent slaves.

9.2.5 Master Status Monitoring

On the "Status" page, you can view the running information of the EtherCAT bus, as shown in the following figure.



Max cycle time(μs) 0
Min cycle time(μs) 0
Cycle time(μs) 0
Max execution time(μs) 0
Min execution time(μs) 0
Execution time(μs) 0

- The left column displays the execution information of the EtherCAT task. The following table lists functions and corresponding system variables.

System Variable	Data Type	Function
dMaxCycleTime	DINT	Maximum cycle time of the EtherCAT task
dMinCycleTime	DINT	Minimum cycle time of the EtherCAT task
dCycleTime	DINT	Cycle time of the EtherCAT task in the previous period
dMaxExeTime	DINT	Maximum execution time of the EtherCAT task
dMinExeTime	DINT	Minimum execution time of the EtherCAT task
dExeTime	DINT	Execution time of the EtherCAT task in the previous period
bResetTime	BOOL	Execution time and cycle time for resetting

- The right column displays the data sending and receiving status of the EtherCAT bus.

System Variable	Data Type	Function
dtx_error_cnt	DINT	Number of EtherCAT data frame sending errors
drx_timeout_cnt	DINT	Number of EtherCAT data frame receiving timeout times
drx_corrupt_cnt	DINT	Number of invalid frame receiving events by EtherCAT
drx_unmatch_cnt	DINT	Number of mismatched frame receiving events by EtherCAT
dLoss_frames	DINT	Number of lost data frames by EtherCAT
bClearFrameCounter	BOOL	Resetting of the EtherCAT data frame counter register

- You can also use system variables to monitor the running, stop, or connection status of the master station.

System Variable	Data Type	Function
bMasterRunState	BOOL	Running status of the EtherCAT master station After the EtherCAT master station receives the RUN command and all slave stations are started, this variable becomes TRUE. Note: If some slave stations are disconnected during EtherCAT running, this variable is still TRUE.
bLinkState	BOOL	Connection status of the master station The variable is ON as long as one slave station is physically connected to the master station. The variable is OFF if no slave station is physically connected to the master station.
iSlavesState	INT	Online status of all slave stations When all the configured slave stations are running, the value is 1. When any slave station is not running, the value is 0.
iFirstErrorSlave	INT	When a configured slave station fails (the state machine switches to a non-OP state or is offline), this variable displays the configuration location of the first disconnected slave station.
iSlavesLinkState	INT	Physical connection status of all slave stations When the physical connections of all configured slave stations are normal, the value is 1; if the physical connection of any slave station is abnormal, the value is 0.

9.2.6 Summary of System Variables

Table 9–2 System variables for EtherCAT communication

System Variable	Data Type	Function
bMasterRunState	BOOL	Running status of the EtherCAT master station After the EtherCAT master station receives the RUN command and all slave stations are started, this variable becomes TRUE. Note: If some slave stations are disconnected during EtherCAT running, this variable is still TRUE.
bLinkState	BOOL	Connection status of the master station The variable is ON as long as one slave station is physically connected to the master station. The variable is OFF if no slave station is physically connected to the master station.
bHeartBeat	BOOL	EtherCAT real-time task heartbeat Flip once per EtherCAT real-time task cycle
bBolckHeartBeat	BOOL	EtherCAT non-real-time task heartbeat Flip once per EtherCAT non-real-time task cycle
dMaxCycleTime	DINT	Maximum cycle time of the EtherCAT task
dMinCycleTime	DINT	Minimum cycle time of the EtherCAT task
dCycleTime	DINT	Cycle time of the EtherCAT task in the previous period
dMaxExeTime	DINT	Maximum execution time of the EtherCAT task
dMinExeTime	DINT	Minimum execution time of the EtherCAT task
dExeTime	DINT	Execution time of the EtherCAT task in the previous period
dtx_frames	DINT	Total number of sent frames
drx_frames	DINT	Total number of received frames
dtx_frames_rates	DINT	Frame sending rate (frames/s)
drx_frames_rates	DINT	Frame receiving rate (frames/s)
dtx_bytes_rate	DINT	Frame sending rate (bytes/s)
drx_bytes_rate	DINT	Frame receiving rate (bytes/s)
dloss_frames	DINT	Number of lost data frames by EtherCAT
bResetTime	BOOL	Execution time and cycle time for resetting
bStopMaster	BOOL	Stop of the EtherCAT master station The EtherCAT master station stops at the rising edge of the variable input and then the variable is automatically reset.
bStartMaster	BOOL	Startup of the EtherCAT master station When the EtherCAT bus fails or stops, the EtherCAT master station is restarted at the rising edge of the variable input and then the variable is automatically reset.
bClearFrameCounter	BOOL	Resetting of the EtherCAT data frame counter register
iSlavesState	INT	Online status of all slave stations When all the configured slave stations are running, the value is 1. When any slave station is not running, the value is 0.
iFirstErrorSlave	INT	When a configured slave station fails (the state machine switches to a non-OP state or is offline), this variable displays the configuration location of the first disconnected slave station.
dLibVersion	DINT	Version of EtherCAT system library software
dMstVersion	DINT	Version of EtherCAT master station software
dDriveVersion	DINT	Version of EtherCAT network adapter drive board software

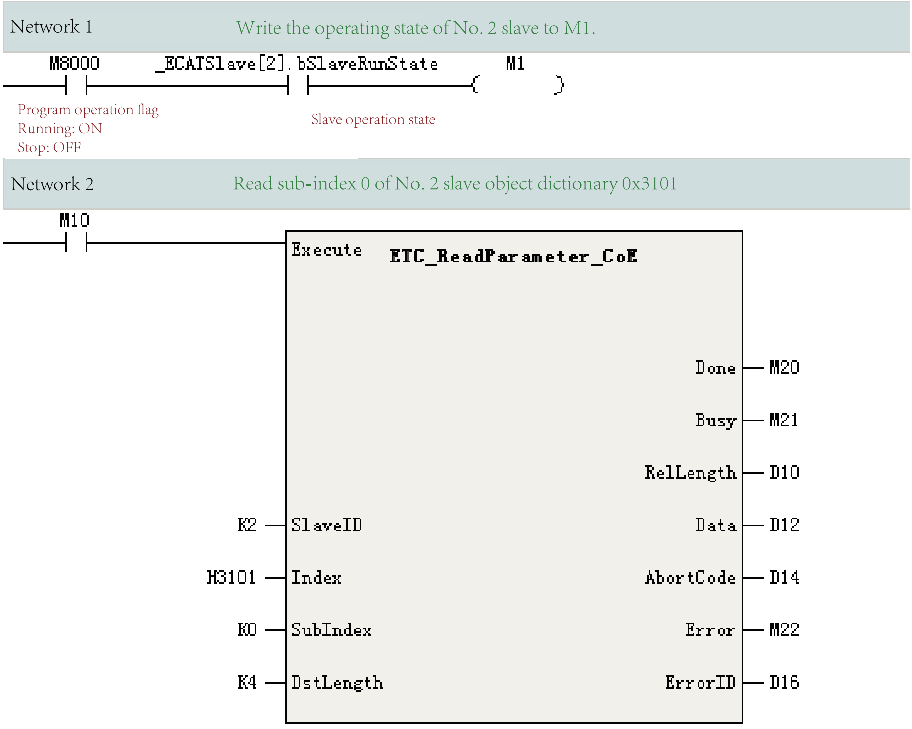
System Variable	Data Type	Function
dtx_error_cnt	DINT	Number of EtherCAT data frame sending errors
drx_timeout_cnt	DINT	Number of EtherCAT data frame receiving timeout times
drx_corrupt_cnt	DINT	Number of invalid frame receiving events by EtherCAT
drx_unmatch_cnt	DINT	Number of mismatched frame receiving events by EtherCAT
dRxPDOLength	DINT	Total length of received PDOs in the configuration (bytes)
dTxPDOLength	DINT	Total length of send PDOs in the configuration (bytes)
dConfigureState	DINT	For internal use of configuration status
dDelay	DINT	Adjustment value of EtherCAT master station synchronization regulator
iSlavesLinkState	INT	Physical connection status of all slave stations When the physical connections of all configured slave stations are normal, the value is 1; if the physical connection of any slave station is abnormal, the value is 0.

9.3 Slave Configuration

9.3.1 General Settings

Configuration address

Configuration addresses of slave stations are sequential addresses in the AutoShop device tree, which increase from 0. The configuration address can be used as a subscript of the slave system variable array or as the slave address for reading and writing SDO instructions.



Network 1 Write the operating state of No. 2 slave to M1.

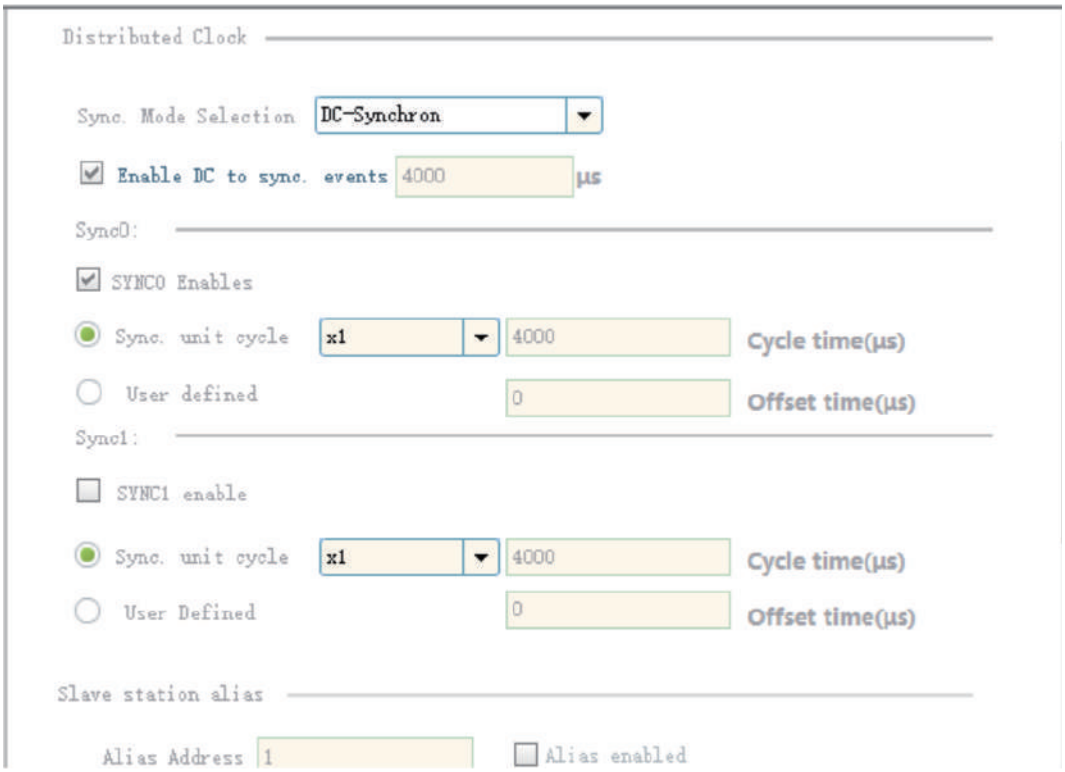
Program operation flag
Running: ON
Stop: OFF

Slave operation state

Network 2 Read sub-index 0 of No. 2 slave object dictionary 0x3101

Distributed clock (DC)

You can set the synchronization running mode for a slave station on the following page.



Distributed Clock

Sync. Mode Selection: DC-Synchron

☑ Enable DC to sync. events 4000 µs

Sync0:

☑ SYNC0 Enables

● Sync. unit cycle x1 4000 Cycle time(µs)

○ User defined 0 Offset time(µs)

Sync1:

☐ SYNC1 enable

● Sync. unit cycle x1 4000 Cycle time(µs)

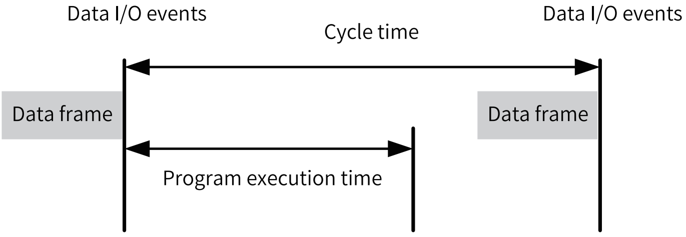
○ User Defined 0 Offset time(µs)

Slave station alias

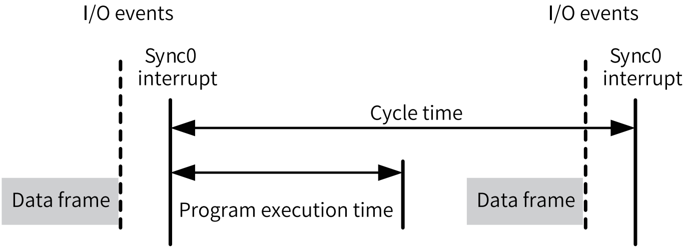
Alias Address 1 ☐ Alias enabled

Synchronization mode selection: For an EtherCAT slave station, options are "FreeRun", "SM-Synchron", and "DC-Synchron". The available options vary with the selected slave station.

For example, the GL10-RTU-ECTA module only supports SM-Synchron, and does not support SYNC0 and SYNC1. There is only one interrupt for data input and output events inside the clock slave station, and the internal processing logic of the slave station is shown in the following figure.



Take the GR10-4PME module that supports only DC-Synchron as an example. In this mode, the Sync interrupt of the slave station can be configured. The DC Sync event is enabled by default, the SYNC0 interrupt is enabled, the period of the SYNC0 interrupt is the same as the cycle time of the EtherCAT master station, and the SYNC1 interrupt is not enabled.



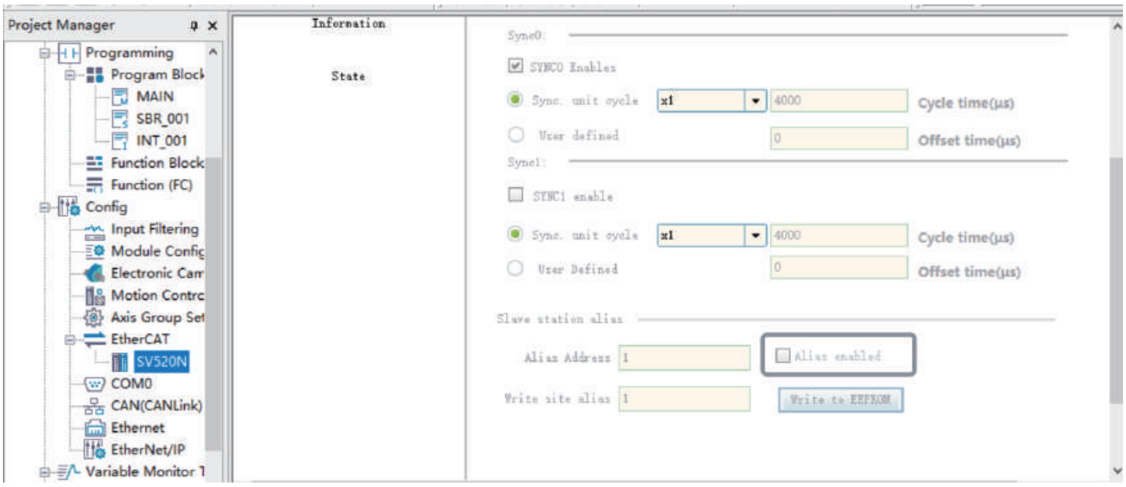
Note

Users not familiar with the EtherCAT communication principles shall not modify the default configuration in DC-Synchron mode.

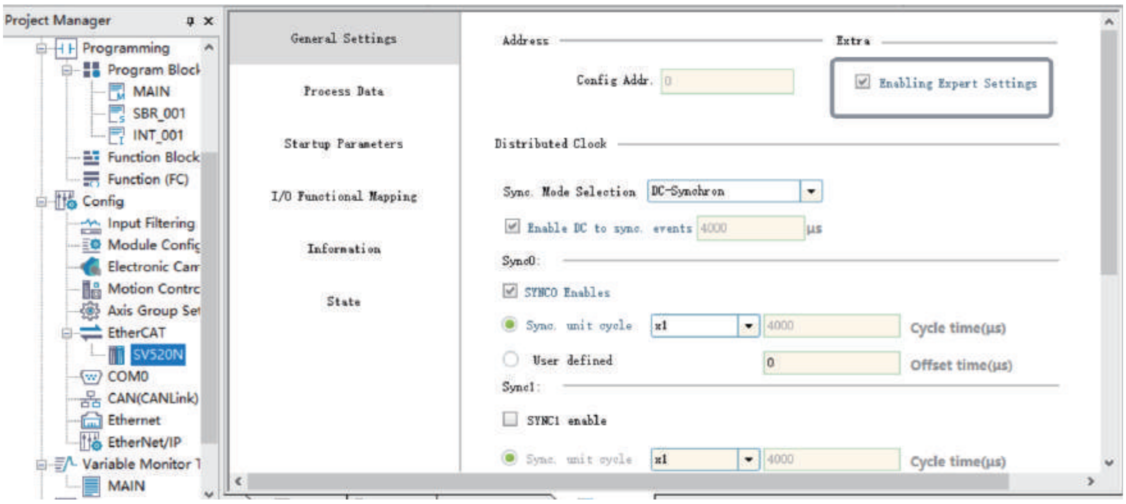
Slave alias setting

An alias can be set for a slave station only when the expert mode is enabled. To enable the station alias function, you must set an alias for the slave station first. You can set parameters to set an alias for Inovance servo, use the DIP switch to set an alias for the GR10-0808ETNE module, and set the EtherCAT master station to set an alias for GL10-RTU-ECTA. Set an alias for an ECTA module as follows.

1. Create a configuration, do not select "Alias enable", download the program, and wait until the slave station is started.



2. Select "Enable Expert setting". In the "Write site alias" text box, input an alias, enter the monitoring status, and then click "Write EEPROM". Wait until the write operation is completed. The following takes writing 1 as an example.



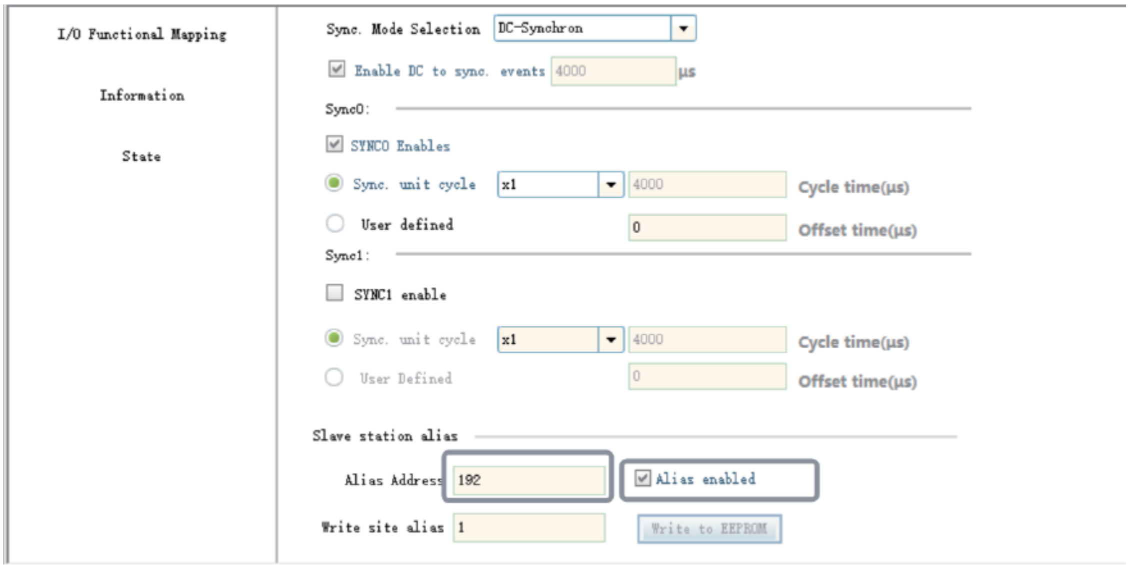
① Select "Enable Expert setting".

② Write an alias for the target station.

③ Click "Write EEPROM".

④ Click "OK" and restart the slave station.

3. Power on the slave station again. You can perform auto scanning to check whether the alias is written.
4. In the configuration, select "Alias enable". In the "Alias address" text box, input the actual alias of the current slave station, and then download the program.



Sync. Mode Selection: DC-Synchron

☑ Enable DC to sync. events 4000 μs

Sync0:

☑ SYNC0 Enables

● Sync. unit cycle x1 4000 Cycle time(μs)

○ User defined 0 Offset time(μs)

Sync1:

□ SYNC1 enable

● Sync. unit cycle x1 4000 Cycle time(μs)

○ User Defined 0 Offset time(μs)

Slave station alias

Alias Address 192 ☑ Alias enabled

Write site alias 1

① Select "Enable Expert setting".

② Select "Alias enable".

③ Write the alias address.

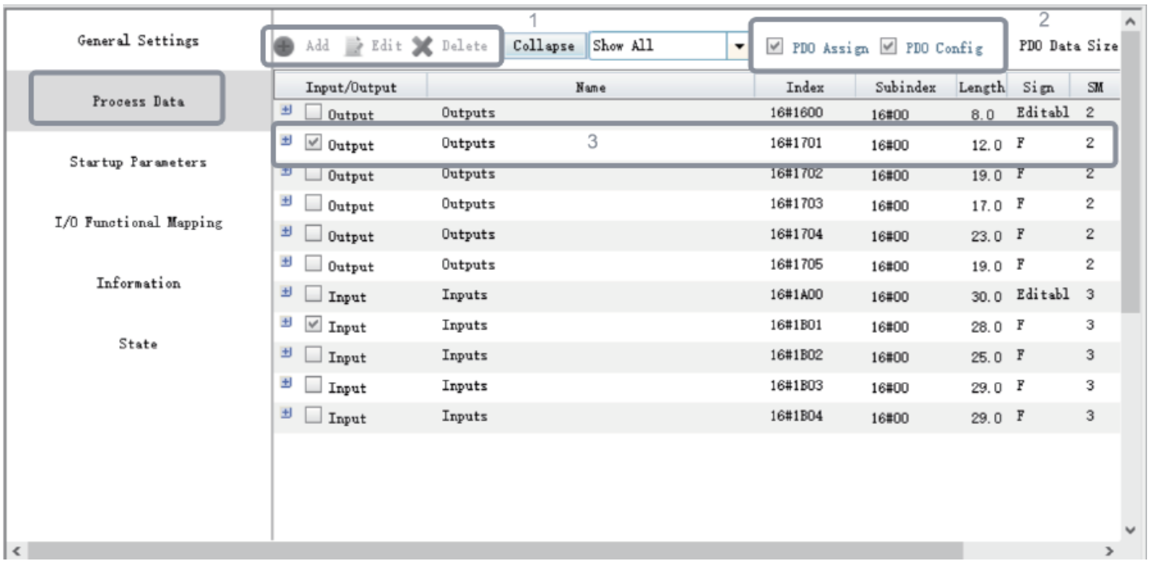
Note

- Avoid mixing the use of station alias and station names. Avoid alias conflicts when using aliases.
- It is recommended to enable the alias function for all slaves when setting up branch module networking.
- After the alias function is enabled for slaves, function blocks for reading and writing SDOs and system variables for accessing the slaves still use the configuration addresses.

9.3.2 Process Data

You can edit PDOs on the "Process data" page.

The "Process data" page is as follows.



	Input/Output	Name	Index	Subindex	Length	Sign	SM
□	Output	Outputs	16#1600	16#00	8.0	Editabl	2
☑	Output	Outputs	16#1701	16#00	12.0	F	2
□	Output	Outputs	16#1702	16#00	19.0	F	2
□	Output	Outputs	16#1703	16#00	17.0	F	2
□	Output	Outputs	16#1704	16#00	23.0	F	2
□	Output	Outputs	16#1705	16#00	19.0	F	2
□	Input	Inputs	16#1A00	16#00	30.0	Editabl	3
☑	Input	Inputs	16#1B01	16#00	28.0	F	3
□	Input	Inputs	16#1B02	16#00	25.0	F	3
□	Input	Inputs	16#1B03	16#00	29.0	F	3
□	Input	Inputs	16#1B04	16#00	29.0	F	3

① PDO editing button

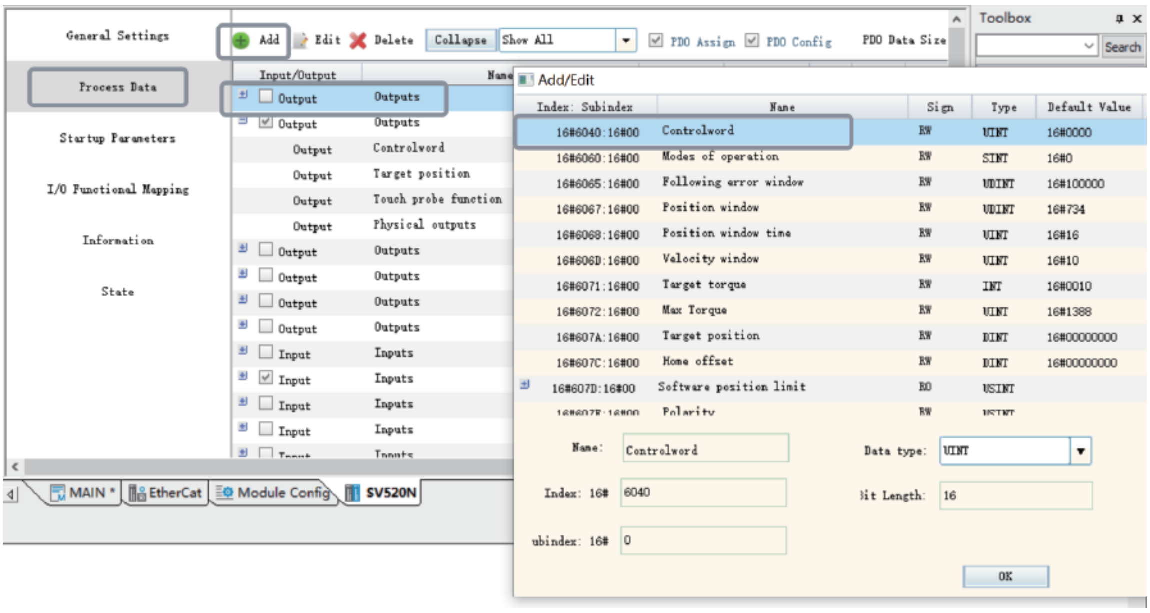
② PDO configuration downloading selection area

③ PDO display area

PDOs are divided into output PDOs and input PDOs by data flow direction. Output PDOs indicate process data sent from the EtherCAT master station to an EtherCAT slave station, such as the control word 0x6040. Input PDOs indicate process data sent from an EtherCAT slave station to the master station.

Each slave station may have one or more groups of PDOs. As shown in the preceding figure, the first group of input PDOs and the first group of output PDOs can be added, edited, or deleted.

The following describes how to add a PDO.



Index: Subindex	Name	Sign	Type	Default Value
16#6040:16#00	Controlword	RW	UINT	16#0000
16#6060:16#00	Modes of operation	RW	SINT	16#0
16#6065:16#00	Following error window	RW	UDINT	16#100000
16#6067:16#00	Position window	RW	UDINT	16#734
16#6068:16#00	Position window time	RW	UINT	16#16
16#606D:16#00	Velocity window	RW	UINT	16#10
16#6071:16#00	Target torque	RW	INT	16#0010
16#6072:16#00	Max Torque	RW	UINT	16#1388
16#607A:16#00	Target position	RW	DINT	16#00000000
16#607C:16#00	Home offset	RW	DINT	16#00000000
16#607D:16#00	Software position limit	RO	USINT	
16#607E:16#00	Polarity	RW	USINT	

Name: Controlword

Data type: UINT

Index: 16# 6040

Bit Length: 16

ubindex: 16# 0

OK

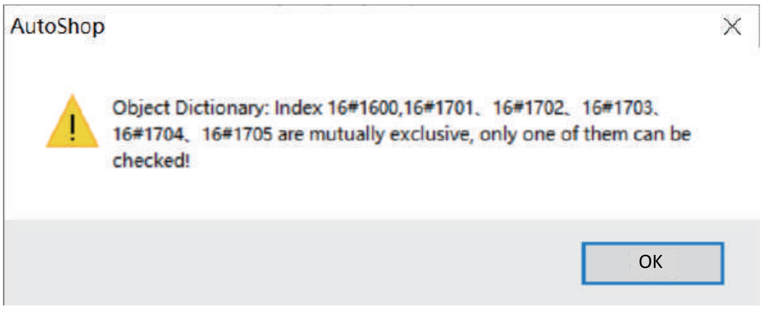
① Select a PDO in the first group.

② Click "Add".

③ Select 6060.

④ Click "OK".

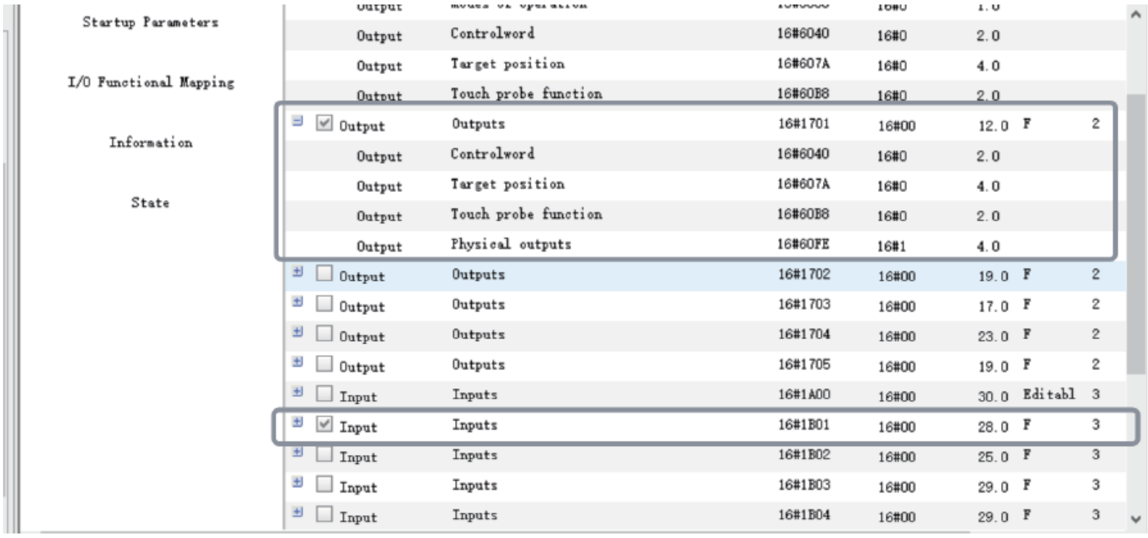
When a slave station has multiple groups of PDOs, such PDO groups are exclusive, such as IS660N. You can select one group each time.



AutoShop

Object Dictionary: Index 16#1600,16#1701、16#1702、16#1703、16#1704、16#1705 are mutually exclusive, only one of them can be checked!

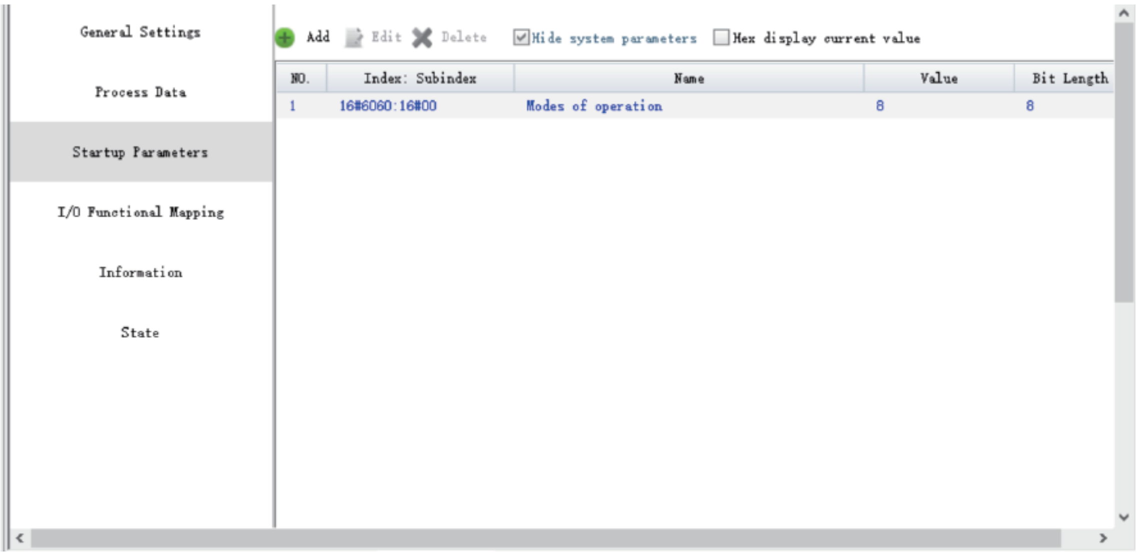
Such mutually exclusive relationship varies with the slave station. For example, you can select multiple PDO groups for GL10-RTU-ECTA.



	Output	Modes of operation	16#6060	16#0	1.0		
	Output	Controlword	16#6040	16#0	2.0		
	Output	Target position	16#607A	16#0	4.0		
	Output	Touch probe function	16#60B8	16#0	2.0		
⊟ ☑	Output	Outputs	16#1701	16#00	12.0	F	2
	Output	Controlword	16#6040	16#0	2.0		
	Output	Target position	16#607A	16#0	4.0		
	Output	Touch probe function	16#60B8	16#0	2.0		
	Output	Physical outputs	16#60FE	16#1	4.0		
⊞ □	Output	Outputs	16#1702	16#00	19.0	F	2
⊞ □	Output	Outputs	16#1703	16#00	17.0	F	2
⊞ □	Output	Outputs	16#1704	16#00	23.0	F	2
⊞ □	Output	Outputs	16#1705	16#00	19.0	F	2
⊞ □	Input	Inputs	16#1A00	16#00	30.0	Editabl	3
⊞ ☑	Input	Inputs	16#1B01	16#00	28.0	F	3
⊞ □	Input	Inputs	16#1B02	16#00	25.0	F	3
⊞ □	Input	Inputs	16#1B03	16#00	29.0	F	3
⊞ □	Input	Inputs	16#1B04	16#00	29.0	F	3

The master station downloads the PDO configuration relationships to EtherCAT slave stations in the form of startup parameters through PDO allocation and PDO mapping.

"PDO allocation" is used to download the selected PDO group number to the slave station, while "PDO allocation" is used to download editable PDOs of a group to the slave station. When PDOs are modified but "PDO allocation" and "PDO configuration" are not selected, the slave station may not be started. The configurations can be viewed in the startup parameter list.

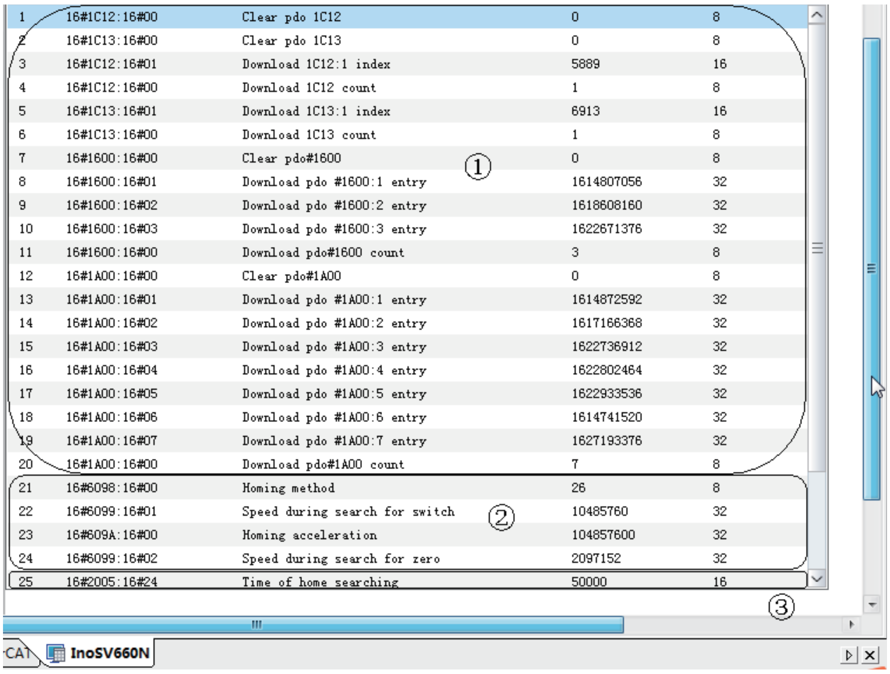


☑ Hide system parameters □ Hex display current value

NO.	Index: Subindex	Name	Value	Bit Length
1	16#6060:16#00	Modes of operation	8	8

9.3.3 Startup Parameters

Startup parameters are used to write slave station PDO configuration, factory settings, and parameters specified by some protocols (such as the 402 protocol) to the slave station through writing SDO when the slave station is in the PreOP state.
Take IS620N as an example:



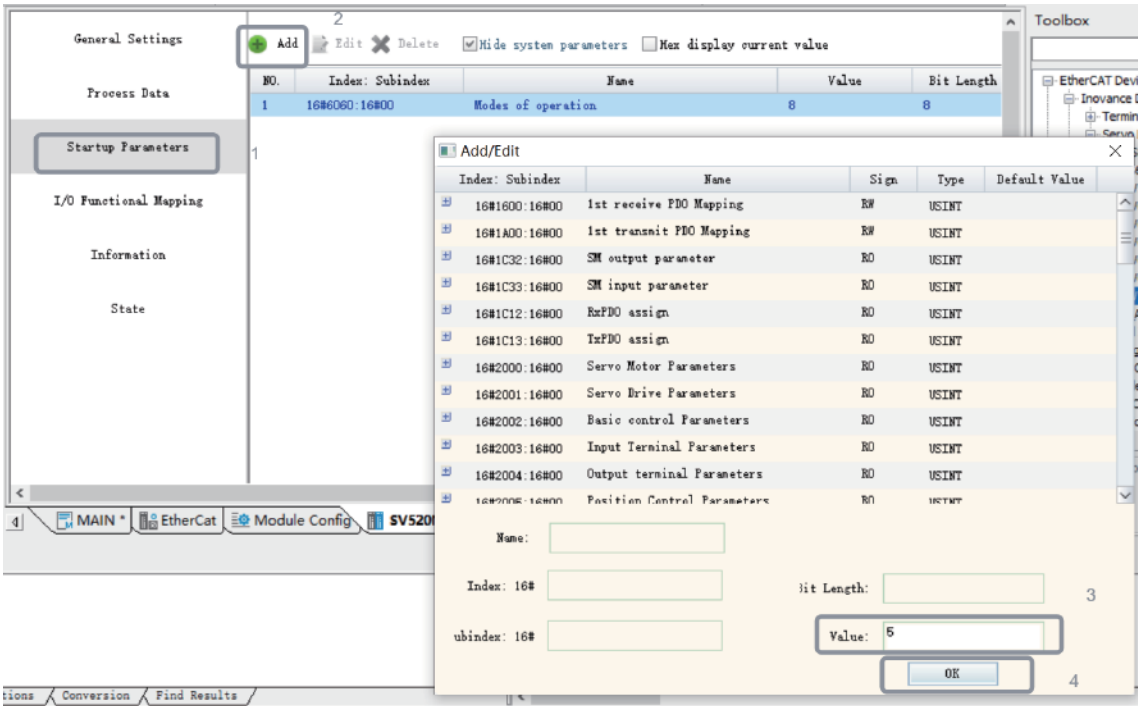
1	16#1C12:16#00	Clear pdo 1C12	0	8
2	16#1C13:16#00	Clear pdo 1C13	0	8
3	16#1C12:16#01	Download 1C12:1 index	5889	16
4	16#1C12:16#00	Download 1C12 count	1	8
5	16#1C13:16#01	Download 1C13:1 index	6913	16
6	16#1C13:16#00	Download 1C13 count	1	8
7	16#1600:16#00	Clear pdo#1600	0	8
8	16#1600:16#01	Download pdo #1600:1 entry	1614807056	32
9	16#1600:16#02	Download pdo #1600:2 entry	1618608160	32
10	16#1600:16#03	Download pdo #1600:3 entry	1622671376	32
11	16#1600:16#00	Download pdo#1600 count	3	8
12	16#1A00:16#00	Clear pdo#1A00	0	8
13	16#1A00:16#01	Download pdo #1A00:1 entry	1614872592	32
14	16#1A00:16#02	Download pdo #1A00:2 entry	1617166368	32
15	16#1A00:16#03	Download pdo #1A00:3 entry	1622736912	32
16	16#1A00:16#04	Download pdo #1A00:4 entry	1622802464	32
17	16#1A00:16#05	Download pdo #1A00:5 entry	1622933536	32
18	16#1A00:16#06	Download pdo #1A00:6 entry	1614741520	32
19	16#1A00:16#07	Download pdo #1A00:7 entry	1627193376	32
20	16#1A00:16#00	Download pdo#1A00 count	7	8
21	16#6098:16#00	Homing method	26	8
22	16#6099:16#01	Speed during search for switch	10485760	32
23	16#609A:16#00	Homing acceleration	104857600	32
24	16#6099:16#02	Speed during search for zero	2097152	32
25	16#2005:16#24	Time of home searching	50000	16

① PDO configuration parameters

② 402 protocol parameters

③ Factory parameters

On this page, you can add startup parameters as required. For example, you can add the object dictionary 0x605a and modify its value to 5 as follows.



NO.	Index: Subindex	Name	Value	Bit Length
1	16#6060:16#00	Modes of operation	8	8

Add/Edit

Index: Subindex	Name	Sign	Type	Default Value
16#1600:16#00	1st receive PDO Mapping	RW	USINT	
16#1A00:16#00	1st transmit PDO Mapping	RW	USINT	
16#1C32:16#00	SM output parameter	RO	USINT	
16#1C33:16#00	SM input parameter	RO	USINT	
16#1C12:16#00	RxPDO assign	RO	USINT	
16#1C13:16#00	TxPDO assign	RO	USINT	
16#2000:16#00	Servo Motor Parameters	RO	USINT	
16#2001:16#00	Servo Drive Parameters	RO	USINT	
16#2002:16#00	Basic control Parameters	RO	USINT	
16#2003:16#00	Input Terminal Parameters	RO	USINT	
16#2004:16#00	Output terminal Parameters	RO	USINT	
16#2005:16#00	Position Control Parameters	RO	USINT	

Name:

Index: 16#

ubindex: 16#

3it Length:

Value: 5

① Click "Add".

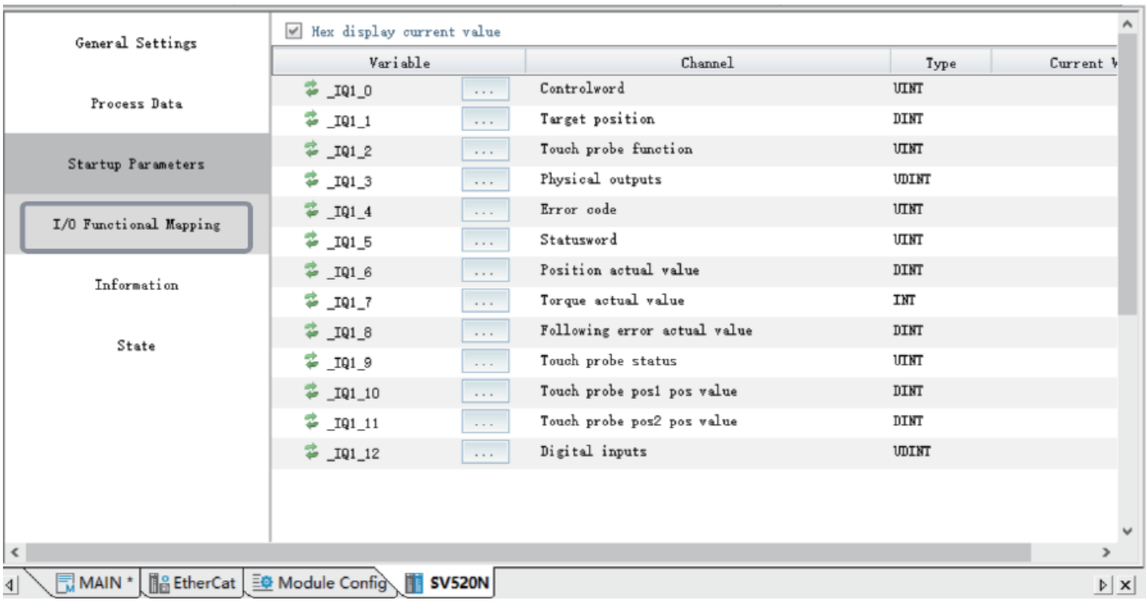
② Select 605A.

③ Modify its value to 5.

④ Click "OK".

9.3.4 I/O Function Mapping

You can control an EtherCAT slave station module by controlling the operation variables only after the PDO data is connected to the PLC.
The "IO Functional Mapping" page is as follows.



☑ Hex display current value

Variable		Channel	Type	Current V
_IQ1_0	...	Controlword	UINT	
_IQ1_1	...	Target position	DINT	
_IQ1_2	...	Touch probe function	UINT	
_IQ1_3	...	Physical outputs	UDINT	
_IQ1_4	...	Error code	UINT	
_IQ1_5	...	Statusword	UINT	
_IQ1_6	...	Position actual value	DINT	
_IQ1_7	...	Torque actual value	INT	
_IQ1_8	...	Following error actual value	DINT	
_IQ1_9	...	Touch probe status	UINT	
_IQ1_10	...	Touch probe pos1 pos value	DINT	
_IQ1_11	...	Touch probe pos2 pos value	DINT	
_IQ1_12	...	Digital inputs	UDINT	

Each time a slave station is added, a group of internal variables are automatically created and connected to the PDO of the slave station, such as IQ2_0.

Note

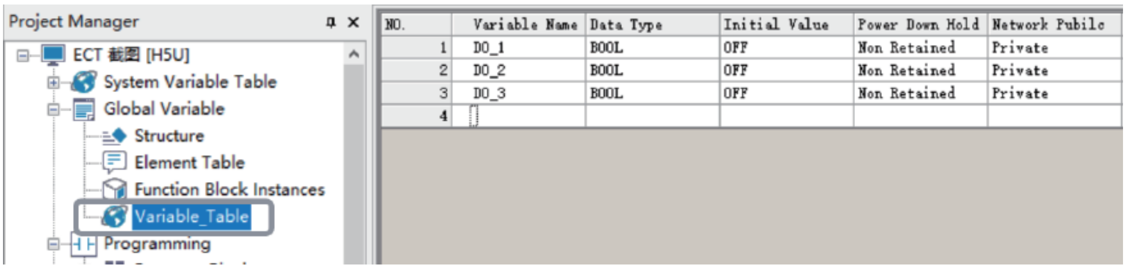
(1) The automatically generated variables change when the module position is changed or any PDO is added, deleted, or modified.

(2) If a slave is associated with a motion control axis, such as SV660N, these variables can only be controlled through axis instructions.

Associated variables

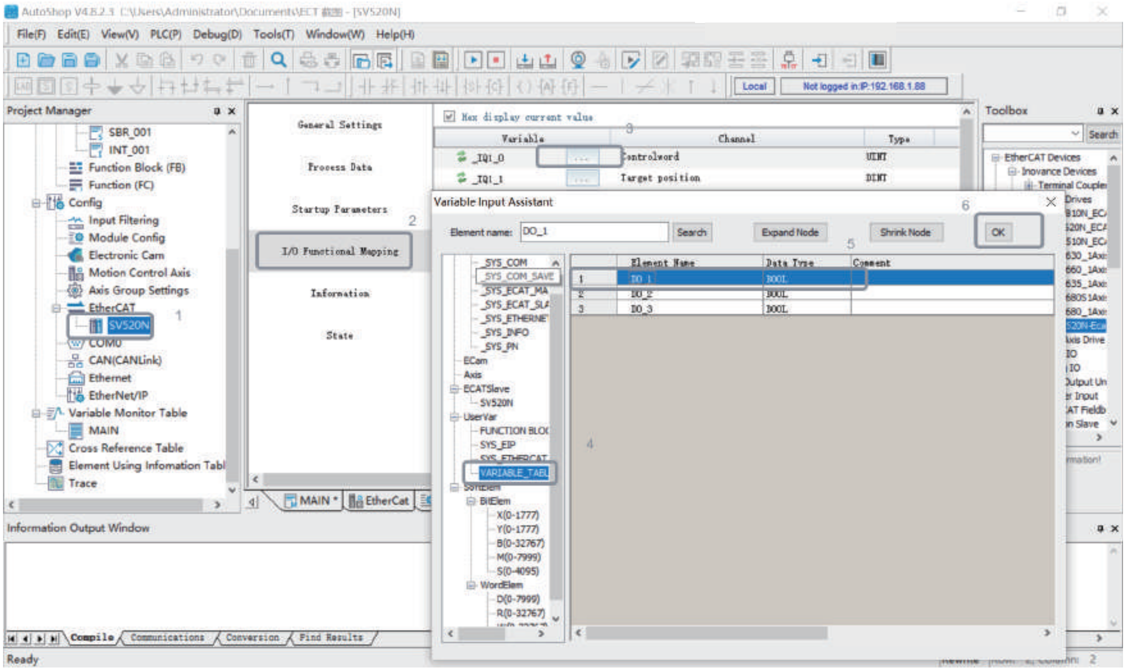
To modify an associated variable, perform the following operations (taking the GR10-1616ETNE module as an example).

1. Open the variable table and add a variable.



NO.	Variable Name	Data Type	Initial Value	Power Down Hold	Network Pubilc
1	DO_1	BOOL	OFF	Non Retained	Private
2	DO_2	BOOL	OFF	Non Retained	Private
3	DO_3	BOOL	OFF	Non Retained	Private
4					

2. Associate the variable on the "IO Functional Mapping" page.



① Open the slave station.

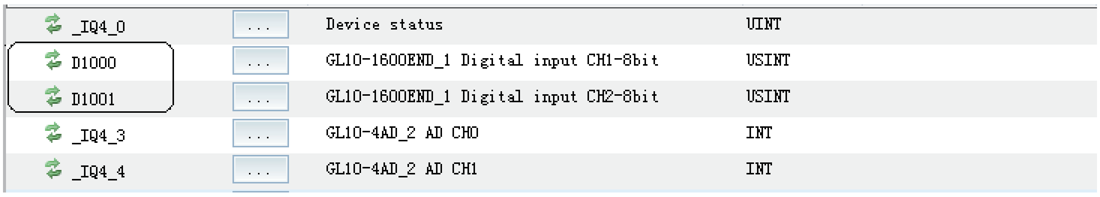
② Select "IO Functional Mapping".

③ Click the icon "...".

④ Select "Variable Table".

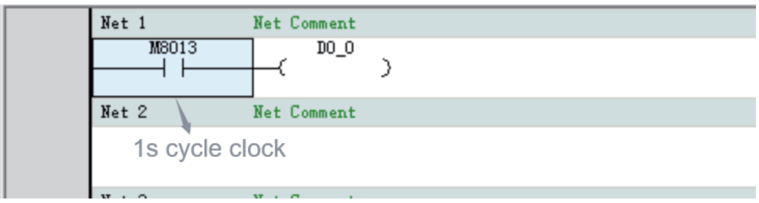
⑤ Select the variable and click "OK".

3. The following page is displayed.



_IQ4_0	...	Device status	UINT
D1000	...	GL10-1600END_1 Digital input CH1-8bit	USINT
D1001	...	GL10-1600END_1 Digital input CH2-8bit	USINT
_IQ4_3	...	GL10-4AD_2 AD CH0	INT
_IQ4_4	...	GL10-4AD_2 AD CH1	INT

4. The PLC program controls DO_0 to flash once every second.

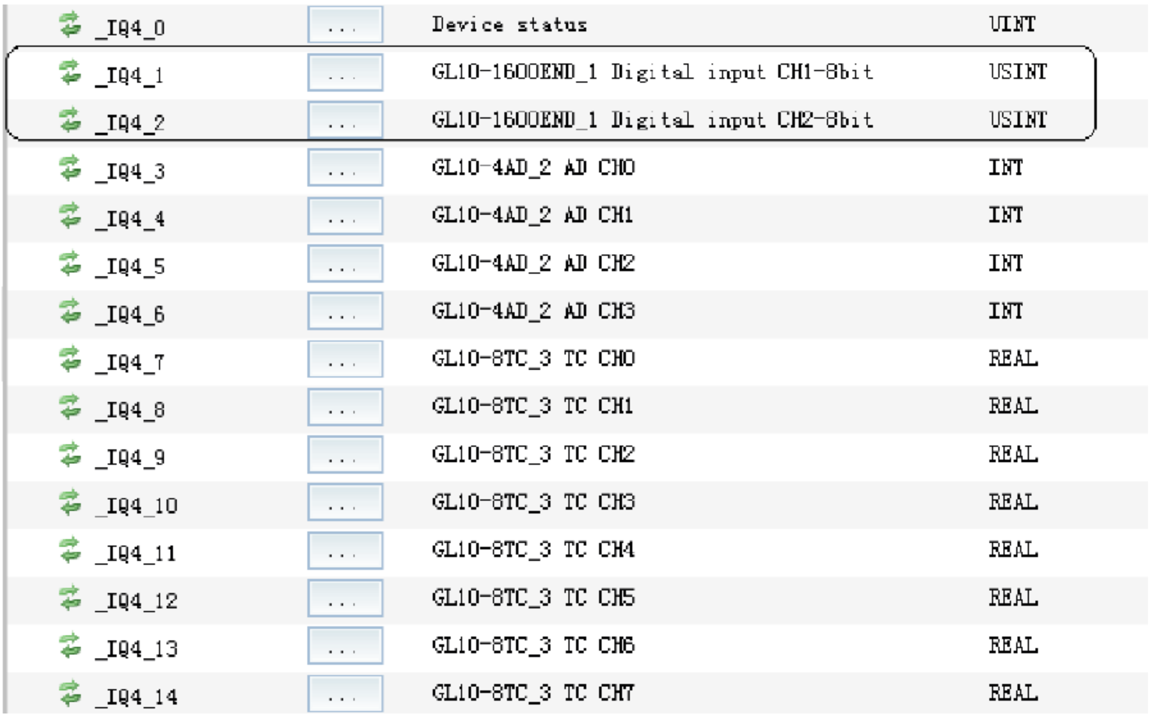


Mapping rules

Data types supported by customized variables are BOOL, BYTE, INT, DINT, and REAL. PDO variables of EtherCAT slave stations support more data types. The following table lists the mapping rules available on the "IO Functional Mapping" page.

EtherCAT Slave Station Type	Bit Length	Mapping Rules
BOOL	1	BOOL
BYTE	8	INT: Low-order 8 bits are valid. High-order 8 bits are reserved. BOOL[8]: An 8-bit BOOL-type array is used. BYTE: 8-bit BYTE-type mapping is used.
SINT	8	INT: Low-order 8 bits are valid. High-order 8 bits are reserved. BOOL[8]: An 8-bit BOOL-type array is used. BYTE: 8-bit BYTE-type mapping is used.
USINT	8	INT: Low-order 8 bits are valid. High-order 8 bits are reserved. BOOL[8]: An 8-bit BOOL-type array is used. BYTE: 8-bit BYTE-type mapping is used.
BITARR8	8	INT: Low-order 8 bits are valid. High-order 8 bits are reserved. BOOL[8]: An 8-bit BOOL-type array is used. BYTE: 8-bit BYTE-type mapping is used.
BIT8	8	INT: Low-order 8 bits are valid. High-order 8 bits are reserved. BOOL[8]: An 8-bit BOOL-type array is used. BYTE: 8-bit BYTE-type mapping is used.
INT	16	INTBOOL[16]: A 16-bit BOOL-type array is used.
UINT	16	INTBOOL[16]: A 16-bit BOOL-type array is used.
WORD	16	INT BOOL[16]: A 16-bit BOOL-type array is used.
BITARR16	16	INT BOOL[16]: A 16-bit BOOL-type array is used.
DINT	32	DINTBOOL[32]: A 32-bit BOOL-type array is used.
UDINT	32	DINTBOOL[32]: A 32-bit BOOL-type array is used.
DWORD	32	DINT BOOL[32]: A 32-bit BOOL-type array is used.
BITARR32	32	DINT BOOL[32]: A 32-bit BOOL-type array is used.
REAL	32	REAL

The following figure shows an example with the data type of USINT.

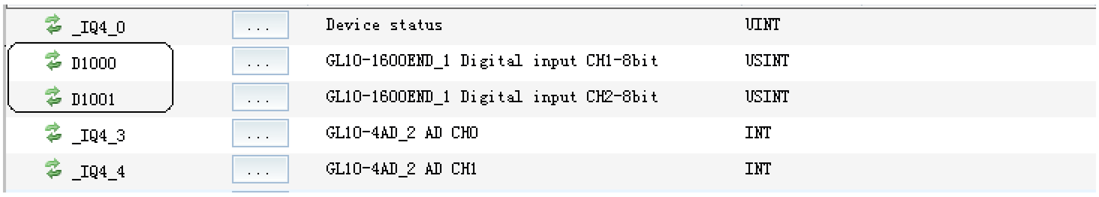


_IQ4_0	...	Device status	UINT
_IQ4_1	...	GL10-1600END_1 Digital input CH1-8bit	USINT
_IQ4_2	...	GL10-1600END_1 Digital input CH2-8bit	USINT
_IQ4_3	...	GL10-4AD_2 AD CH0	INT
_IQ4_4	...	GL10-4AD_2 AD CH1	INT
_IQ4_5	...	GL10-4AD_2 AD CH2	INT
_IQ4_6	...	GL10-4AD_2 AD CH3	INT
_IQ4_7	...	GL10-8TC_3 TC CH0	REAL
_IQ4_8	...	GL10-8TC_3 TC CH1	REAL
_IQ4_9	...	GL10-8TC_3 TC CH2	REAL
_IQ4_10	...	GL10-8TC_3 TC CH3	REAL
_IQ4_11	...	GL10-8TC_3 TC CH4	REAL
_IQ4_12	...	GL10-8TC_3 TC CH5	REAL
_IQ4_13	...	GL10-8TC_3 TC CH6	REAL
_IQ4_14	...	GL10-8TC_3 TC CH7	REAL

To detect the DI channel of the GL10-1600END module, you must allocate a variable to both GL10-1600END_1 Digital input CH1-8bit and GL10-1600END_1 Digital input CH2-8bit (you can also use the default variables IQ4_1 and IQ4_2, but they are not easy to expand and maintain). The configuration can be displayed in the following three ways.

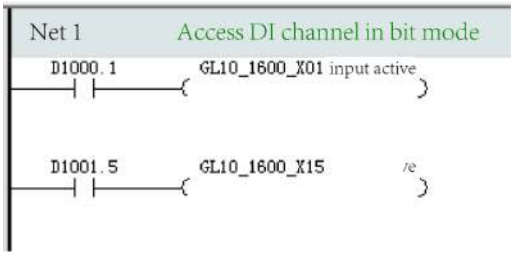
Solution 1: Associate the D element

1. Create a mapping relationship



_IQ4_0	...	Device status	UINT
D1000	...	GL10-1600END_1 Digital input CH1-8bit	USINT
D1001	...	GL10-1600END_1 Digital input CH2-8bit	USINT
_IQ4_3	...	GL10-4AD_2 AD CH0	INT
_IQ4_4	...	GL10-4AD_2 AD CH1	INT

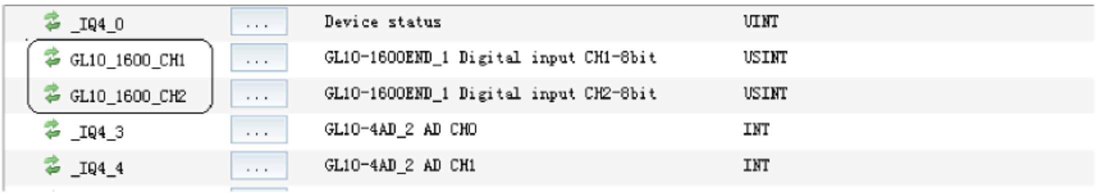
2. Call it in the program



Solution 2: Associate customized variable of the INT type

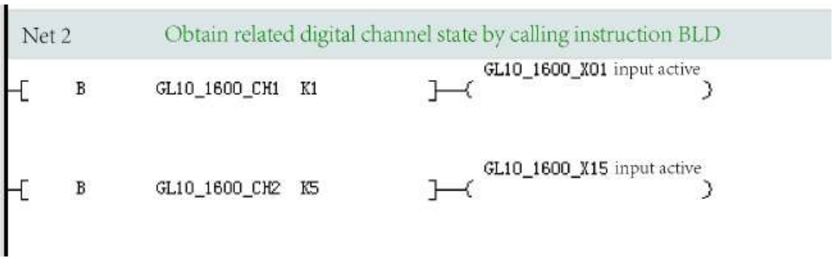
1. Create a global variable

2. Create a mapping relationship



_IQ4_0	...	Device status	UINT
GL10_1600_CH1	...	GL10-1600END_1 Digital input CH1-8bit	USINT
GL10_1600_CH2	...	GL10-1600END_1 Digital input CH2-8bit	USINT
_IQ4_3	...	GL10-4AD_2 AD CH0	INT
_IQ4_4	...	GL10-4AD_2 AD CH1	INT

3. Call it in the program

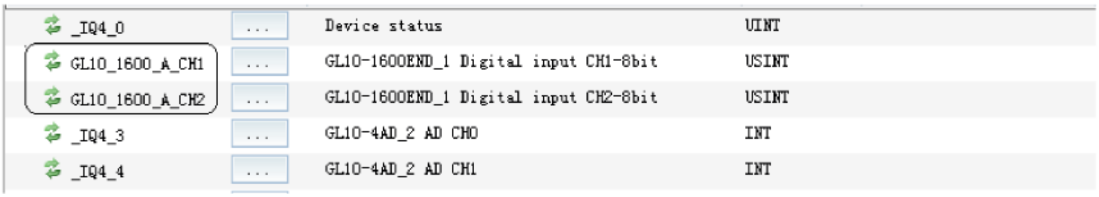


Net 2 Obtain related digital channel state by calling instruction BLD

```
[ B    GL10_1600_CH1  K1    ]—( GL10_1600_X01 input active )
[ B    GL10_1600_CH2  K5    ]—( GL10_1600_X15 input active )
```

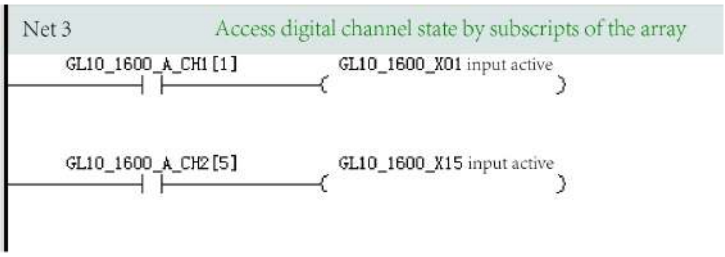
Solution 3: Associate an array of the BOOL type

1. Create a global variable array
2. Create a mapping relationship



_IQ4_0	...	Device status	UINT
GL10_1600_A_CH1	...	GL10-1600END_1 Digital input CH1-8bit	USINT
GL10_1600_A_CH2	...	GL10-1600END_1 Digital input CH2-8bit	USINT
_IQ4_3	...	GL10-4AD_2 AD CH0	INT
_IQ4_4	...	GL10-4AD_2 AD CH1	INT

3. Call it in the program



Net 3 Access digital channel state by subscripts of the array

```
GL10_1600_A_CH1[1]        GL10_1600_X01 input active
——| |——————————————————————( )
GL10_1600_A_CH2[5]        GL10_1600_X15 input active
——| |——————————————————————( )
```

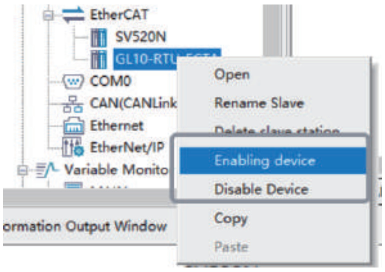
9.3.5 Start/Stop, Disable, and Enable

For AutoShop 4.0.0.0 matching PCB software of a version earlier than 3.0.0.0, you cannot start or stop a single slave station, but can only start or stop the entire EtherCAT bus.

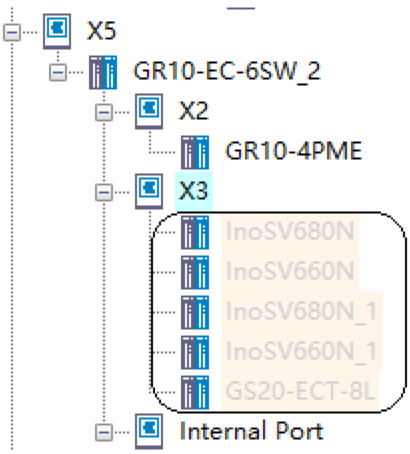
For AutoShop 4.0.0.0 matching PCB software of version 3.0.0.0 or later, you can select "Auto restart of slave" on the EtherCAT master station configuration page to automatically restart a slave station.

When the number of configured slave stations is greater than the number of connected ones, you can disable unavailable slave stations in the configuration.

To enable or disable a slave station, right-click the target slave station, and then select "Disable Device" or "Enabling device".



When the network contains branch modules, if a branch port is disabled, all slave stations mounted to this port are disabled.



9.3.6 Disabling Slaves Using Instructions

You can disable EtherCAT slave stations during programming by using the ETC_RestartMaster instruction and relevant system variables.

System variables

- The following table lists system variables used to enable or disable specified EtherCAT slave stations.

Name	Unit	Description	Retentive upon Power Failure
bDisableEnable	BOOL	Disable/Enable OFF: Enabled ON: Disabled	No
wDisableState	INT	Configuration status 0: Reserved 1: Enabled 2: Disabled	No

- The following table lists system variables of EtherCAT slave stations used to enable or disable the entire EtherCAT bus.

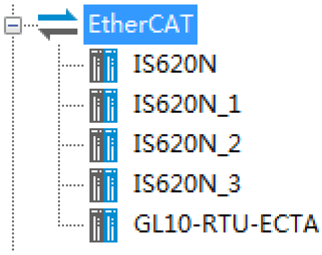
Name	Unit	Description	Retentive upon Power Failure
bDisableMaster	BOOL	Disable/Enable OFF: Enabled ON: Disabled	No
iDisableState	INT	Configuration status 0: Reserved 1: Enabled 2: Disabled	No

Usage

1. After power-on, the PLC initializes the bDisableEnable variable based on the background configuration, updates the configuration list based on the value of the bDisableEnable variable, starts the master station, and then writes the disabling status of the slave station to the wDisableState variable.
2. Wait until the PLC parses the program configuration.
3. Use the PLC program to set the value of the bDisableEnable variable.
4. Use the ETC_RestartMaster instruction to restart the EtherCAT master station.
5. After the master station is restarted, use the bDisableEnable variable to update the configuration list and write the disabling status of the slave station to wDisableState.

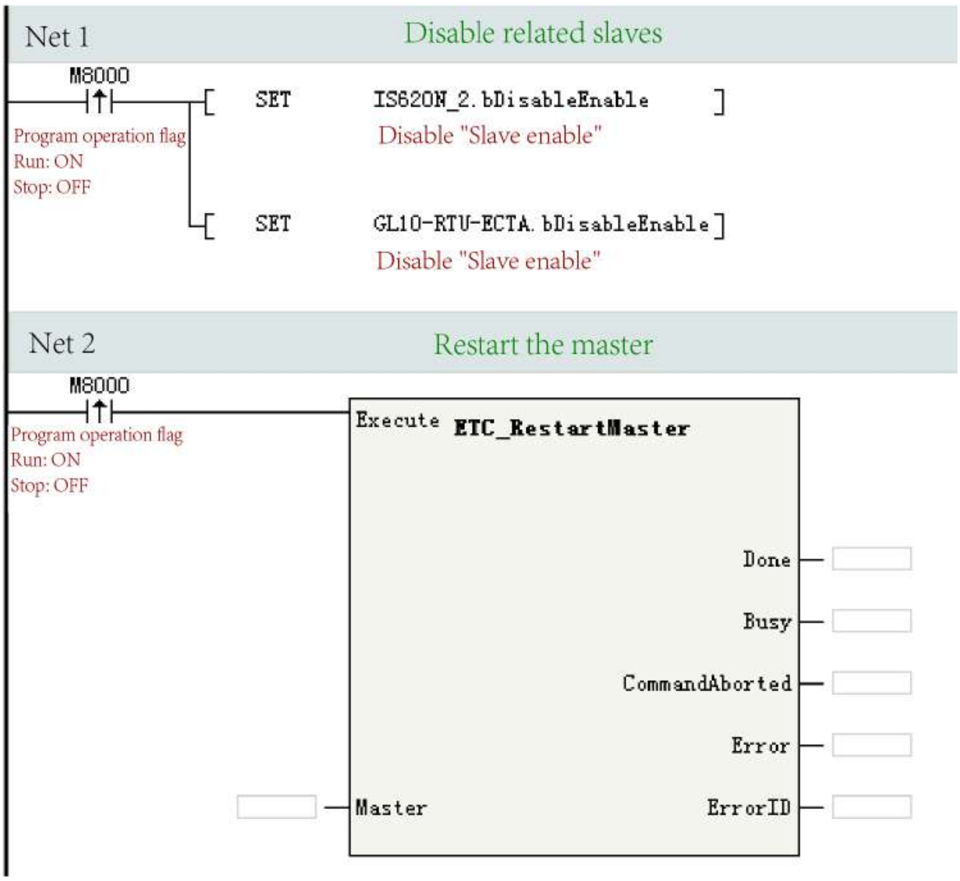
Example

1. Create a configuration by using the background software, enable the EtherCAT communication, and configure four IS620N slave stations and one AM600-RTU-ECTA slave station.

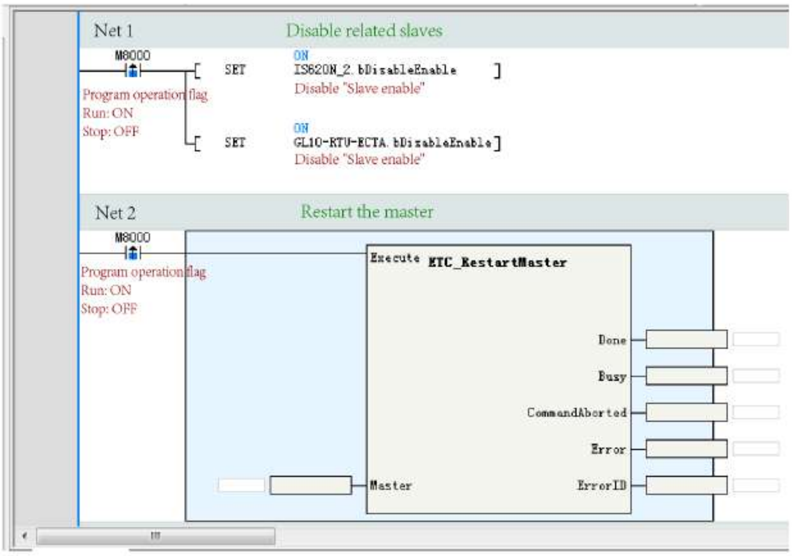


EtherCAT
IS620N
IS620N_1
IS620N_2
IS620N_3
GL10-RTU-ECTA

2. Set the system variables _DisableEnable and bDisableEnable to disable the slave stations IS620N_2 and AM600-RTU-ECTA, and then use the ETC_RestartMaster instruction to restart the master station.



3. Download the project to the controller. The slave stations IS620N_2 and AM600-RTU-ECTA are disabled after start.



9.3.7 System Variables

The following table lists system variables of EtherCAT slave stations.

System Variable	Data Type	Function
bDisableEnable	BOOL	Disable/Enable OFF: Enabled ON: Disabled
wDisableState	INT	Configuration status 0: Reserved 1: Enabled 2: Disabled
bSlaveRunState	BOOL	Running status of slave station The value is TRUE when the slave station is in the OP state; otherwise, the value is FALSE.
bSetAliasState	BOOL	Set the alias status (for the use of the background only) TRUE: Busy FALSE: Idle or setting completed
bSetAliasError	BOOL	Failed to set the alias status (for the use of the background only) TRUE: Failed to set the alias status FALSE: No fault
bSetAlias	BOOL	Set the station alias (for the use of the background only) The value of wTarAlias is written to the slave station at the rising edge of the variable.
wALState	INT	Status of the EtherCAT slave station state machine 1: INIT 2: PreOP 4: SafeOP 8: OP
wAlCode	INT	Code of failure to convert the slave station state machine. For details, see the slave station guide.
wActAlias	INT	Actual alias of the slave station. Initialization is performed once upon power-on, and the modification does not take effect.
wTarAlias	INT	Station alias to be written (for the use of the background only)
wStationAddress	INT	Sequential address of the slave station. Initialization is performed once upon power-on, and the modification does not take effect.

9.4 Faults and Diagnosis

9.4.1 Learning Faults

The BF indicator indicates the fault status of the EtherCAT bus. The following table lists the statuses and solutions.

LED Indicator	Definition	Solution
Off	No fault	/
Flashing	The EtherCAT bus is abnormal.	Troubleshoot the problem based on the fault code displayed. For details, see *"9.4.2 Fault Codes" on page 329.*
On	Failed to request for the master station.	Troubleshoot the problem based on the fault code displayed. For details, see *"9.4.2 Fault Codes" on page 329.*

You can obtain the code of an EtherCAT instruction fault based on the ErrorID in the instruction.

You can view the EtherCAT bus faults on the "Fault Diagnosis" page.

9.4.2 Fault Codes

Fault Code	Cause	Solution
8001	Failed to request for the master station.	1. Check whether the PCB software version matches the background version. 2. Restart the PLC.
8002	Failed to obtain the slave station configuration parameters.	Check whether the PCB software version matches the background version.
8003	Master station startup timed out.	Check the network connection.
8004	Failed to request for the master station.	1 Restart the PLC. 2 An error occurred while loading ECAT. Upgrade the firmware to the correct version.
...	-	-
8200	Failed to write startup parameters.	1. Check whether the startup parameter list contains any object dictionary that is not supported by the slave station. 2. Check whether the value of the object dictionary exceeds the range.
8201	The slave station is lost during running.	1. Check whether the slave station is disconnected from the network. 2. Check whether the slave station is powered off.
8202	The slave station enters a non-OP state during running.	1. Check whether the slave station is disconnected from the network. 2. Check whether the slave station is powered off.
8203	Reserved	-
8204	The slave station type does not match.	1. Check whether the network cable is inversely connected. 2. Check whether the configured device matches the connected device.
8205	The PDO address is incorrect.	1. Check whether the memory is used up. 2. Check whether the background version matches the PCB software version. 3. Power off and then on the device.
8206	The PDO length is incorrect.	Check whether the background version matches the PCB software version.
8301	Failed to switch to the INIT status.	Check whether the slave station state machine supports status conversion.
8302	Failed to switch to the PerOP status.	Check whether the slave station supports the CoE protocol.
8304	Failed to switch to the SafeOP status.	Check whether the PDO communication configuration is correct.

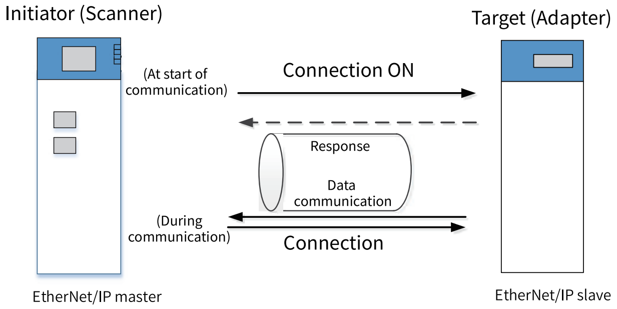
Fault Code	Cause	Solution
8308	Failed to switch to the OP status.	1. Check the network communication quality. 2. Check whether the EtherCAT task cycle is set appropriately.
8310	Configuration of the FMMU unit is incorrect.	Check whether the slave station supports the FMMU unit.
8311	The email configuration is incorrect.	Check whether the slave station supports the SM unit.
8400	The ECTA configuration is incorrect.	Check whether the configured extension module is the connected extension module.
8401	An ECTA hardware error occurred.	1. Check whether the connection between ECTA and the extension module is loose. 2. Replace ECTA.
8402	The extension module mounted to ECTA is incorrect.	Check the fault type of the extension module based on the ECTA guide.
1280	No master station.	Check whether EtherCAT bus communication is enabled.
1281	No slave station.	Check whether this slave station is configured.
1282	The SDO length to be read or written is greater than 4 or equal to 0.	Check whether the read or written SDO length is correct.
1283	No master station.	Check whether the configuration parameters of the master station are correct.
1284	Read or write operation failed. • SDO read/write timed out. • SDO does not exist. • SDO read/write is not supported due to the status of the slave station. • The SDO length to be read or written is incorrect.	Check whether the SDO operation is supported by the slave station state machine. Check whether the SDO to be read or written exists. Check whether the SDO length to be read or written is correct.
1285	Memory request failed.	1. Check whether the PLC memory is used up. 2. Contact Inovance.
1286	The master station stops.	This instruction cannot be called when the master station stops.

When an EtherCAT bus fault occurs, the fault status indicator in the lower-right corner of AutoShop turns to yellow. If you double-click this area, the "Fault Diagnosis" dialog box is displayed, in which you can view the configuration of the failed slave station, the fault code, and the fault details.

10 EtherNet/IP Communication

10.1 Overview

The open connected end is the EtherNet/IP master station, and the opened end is the EtherNet/IP slave station, as shown in the following figure.



The small PLC background AutoShop V4.4.0.5 and later versions support the EtherNet/IP function. The communication specifications are as follows.

- H5U and Easy320/Easy52X series PLCs support one EtherNet/IP master station.
- The minimum cycle communication period (RPI) is 5 ms.
- The maximum data volume of single connection communication is 1400 bytes for Easy, and 500 bytes for H5U.
- A maximum of 32 connections (consumer tags + server tags + slave connections) are supported.
- A maximum of 32 tags (producer tags + server tags) can be created.

Note

- When EtherCAT and EtherNet/IP networks both exist in a networking project, the real-time communication performance of EtherNet/IP network is reduced because EtherCAT has the highest priority by default.
- For protocol details, see the official standard documents EIP-CIP-V1-1.0 and EIP-CIP-V2-1.0.
- The firmware version must be V5.66.0.0 or later, and the software version must be AutoShop V4.8.0.0 or later.

10.2 Technical Specifications

10.2.1 EtherNet/IP Transmission Specifications

Table 10–1 Transmission specifications

Item	Technical Specifications	
	10BASE-T	100BASE-TX
Transmission rate	10 Mbps	100 Mbps
Transmission media	STP or UTP above Cat3[1]	STP or UTP above Cat5[1]
Max. cable length[2]	100 m	100 m

Meet the standard IEEE802.3.

- [1]: STP: shielded twisted pair. UTP: unshielded twisted pair.
- [2]: The maximum cable length is the distance between an EtherNet/IP unit and a network device.

10.2.2 EtherNet/IP Communication Specifications

Item				H5U Series	Easy32x Series	Easy52x Series
CIP service	Implicit message communication	Number of I/O connections at the originator[1]		32	32	32
		Number of I/O connections at the target end[2]		32	32	32
		RPI (communication cycle)		5 ms to 50000 ms		
		Bandwidth allowable for implicit (I/O) message communication	(@4 bytes)	12800 pps[3]	12800 pps[3]	12800 pps[3]
			(@250 bytes)	6400 pps[3]	12800 pps[3]	12800 pps[3]
			(@500 bytes)	3200 pps[3]	12800 pps[3]	12800 pps[3]
			(@1400 bytes)	-	8000 pps[3]	12800 pps[3]
		Maximum data size per connection[4]		500 bytes	1400 bytes	1400 bytes
		Multicast filter[5]		Supported (IGMP client function)		
	Explicit message communication	Number of Class3 tags at the originator[6]		32	32	32
		Number of Class3 tags at the target end[6]		32	32	32
		Number of UCMM tags at the initiator[6]		Number of simultaneous executions: 32	Number of simultaneous executions: 32	Number of simultaneous executions: 32
		Number of UCMM tags at the target end[6]		Number of simultaneous executions: 32	Number of simultaneous executions: 32	Number of simultaneous executions: 32

[1] I/O connections at the originator include:

- Consumer tags: Connections requested by the originator with the name of the producer tag as the connection path.
- Originator generic I/O connections: Connections requested by the originator with the instance ID of the generic I/O connections of the target end as the connection path.

[2] I/O connections at the target end include:

- Consumer tags: Responses from the target end to the connection requests with the name of the producer tag as the connection path.
- Target end generic I/O connections: Responses from the target end to connection requests with the instance ID of the generic I/O connections of the target end as the connection path.

[3] pps refers to Packet Per Second. pps is the unit of network throughput rate. Here, it means the sum of the number of grouping packets sent and received that can be processed in one second. It is calculated according to the following formula: Communication bandwidth pps = 1000 ms/RPI x Number of connections x 2.

[4] Data simultaneity within a connection is guaranteed. The device used supports Large Forward Open (CIP option specification) when the data size is greater than 509 bytes.

[5] The EtherNet/IP unit supports the IGMP client function, so the use of an Ethernet switch that supports IGMP Snooping allows filtering out unattended multicast packets.

[6] The number of tags is as follows:

- Initiator tags include the consumer tags, initiator generic I/O connections, Class3 tags at the originator, and UCMM tags at the initiator. The maximum number is 32.
- Target end tags include the producer tags, target end generic I/O connections, Class3 tags at the target end, and UCMM tags at the target end. The maximum number is 32. The maximum number of target end generic I/O connections is 16.

10.2.3 Quick Reference Table of EtherNet/IP Solutions

The quick reference table of EtherNet/IP solutions lists typical values generated based on the bandwidth allowable for implicit (I/O) message communication provided in *“10.2.2 EtherNet/IP Communication Specifications” on page 332*. It strictly adheres to the relevant communication bandwidth regulations and is designed to help customers to quickly find a solution. When user requirements do not match the solution quick reference table, the bandwidth allowable for implicit (I/O) message communication provided in *“10.2.2 EtherNet/IP Communication Specifications” on page 332* shall be used as a design constraint.

Table 10–2 Solution quick reference table for the H5U series

Specification No.	Data Size	Number of Connections	RPI	Communication Bandwidth
SPEC.001	(0,4] bytes	32	5 ms	12800 pps
SPEC.002	(4,250] bytes	32	10 ms	6400 pps
SPEC.003	(4,250] bytes	16	5 ms	6400 pps
SPEC.004	(250,500] bytes	32	20 ms	3200 pps
SPEC.005	(250,500] bytes	16	10 ms	3200 pps
SPEC.006	(250,500] bytes	8	5 ms	3200 pps

1. You are recommended to set the user program scan cycle to RPI x 2.

2. "(a,b] bytes" means an integer value greater than a but equal to or less than b.

3. When user requirements match the data in the table, but the connection number and scan cycle do not match, balance them to meet the communication bandwidth requirements.

For example, when the data size of user requirement REQ.X does not match the data size of SPEC.Y in the table, REQ.X must meet the following condition:

REQ.X RPI/REQ.X connection number = SPEC.Y RPI/SPEC.Y connection number

Table 10–3 Solution quick reference table for Easy32X

Specification No.	Data Size	Number of Connections	RPI	Communication Bandwidth
SPEC.001	(0,4] bytes	32	5 ms	128000 pps
SPEC.002	(4,250] bytes	32	5 ms	128000 pps
SPEC.003	(250,500] bytes	32	5 ms	128000 pps
SPEC.004	(500,1400] bytes	32	8 ms	8000 pps

1. You are recommended to set the user program scan cycle to RPI x 2.

2. "(a,b] bytes" means an integer value greater than a but equal to or less than b.

3. When user requirements match the data in the table, but the connection number and scan cycle do not match, balance them to meet the communication bandwidth requirements.

For example, when the data size of user requirement REQ.X does not match the data size of SPEC.Y in the table, REQ.X must meet the following condition:

REQ.X RPI/REQ.X connection number = SPEC.Y RPI/SPEC.Y connection number

Table 10–4 Solution quick reference table for Easy52X

Specification No.	Data Size	Number of Connections	RPI	Communication Bandwidth
SPEC.001	(0,4] bytes	32	5 ms	128000 pps
SPEC.002	(4,250] bytes	32	5 ms	128000 pps
SPEC.003	(250,500] bytes	32	5 ms	128000 pps
SPEC.004	(500,1400] bytes	32	5 ms	128000 pps

1. You are recommended to set the user program scan cycle to RPI x 2.

2. "(a,b] bytes" means an integer value greater than a but equal to or less than b.

3. When user requirements match the data in the table, but the connection number and scan cycle do not match, balance them to meet the communication bandwidth requirements.

For example, when the data size of user requirement REQ.X does not match the data size of SPEC.Y in the table, REQ.X must meet the following condition:

REQ.X RPI/REQ.X connection number = SPEC.Y RPI/SPEC.Y connection number

10.2.4 EtherNet/IP Solution Selection Example

Taking REQ.A as an example to describe how to use the solution quick reference table

User requirement REQ.A

One H5U connects to 32 EtherNet/IP devices.

The data size of each connection is 500 bytes for both input data and output data.

RPI is 25 ms.

Solution

1. Use the data size and connection number of REQ.A as indexes and search for the specification in the "Solution quick reference table for the H5U series" of the section *“10.2.3 Quick Reference Table of EtherNet/IP Solutions” on page 333*. SPEC.004 matches.
2. The scan cycle of REQ.A is 25 ms, greater than 20 ms of SPEC.004. In this case, you do not need to reduce the connection number and the recommended user program scan cycle is 50 ms.
3. Options:
   REQ.A communication bandwidth: 1000 ms/RPI x Connection number x 2 = 1000 ms/25 ms x 32 x 2 = 2560 pps.

The communication bandwidth provided in the "Solution quick reference table for the H5U series" of the section *“10.2.3 Quick Reference Table of EtherNet/IP Solutions” on page 333* is 3200 pps, greater than 2560 pps. Therefore, the solution is appropriate.

Taking REQ.B as an example to describe how to use the solution quick reference table

User requirement REQ.B

One H5U connects to 32 EtherNet/IP devices.

The data size of each connection is 300 bytes for both input data and output data.

RPI is 10 ms.

Solution 1

1. Use the data size and connection number of REQ.B as indexes and search for the specification in the "Solution quick reference table for the H5U series" of the section *"10.2.3 Quick Reference Table of EtherNet/IP Solutions" on page 333*. SPEC.004 matches.
2. RPI of REQ.B is 10 ms, less than 20 ms of SPEC.004.
3. Solution: Increase the RPI

RPI of REQ.B must be equal to or greater than 20 ms of SPEC.004. Therefore, you are recommended to set RPI to 20 ms.

Solution 2

1. Use the data size and RPI of REQ.B as indexes and search for the specification in the "Solution quick reference table for the H5U series" of the section *"10.2.3 Quick Reference Table of EtherNet/IP Solutions" on page 333*. SPEC.005 matches.
2. The connection number of REQ.B is 32, greater than 16 of SPEC.005.
3. Solution: Reduce the connection number

The connection number of REQ.B must be equal to or less than 16 of SPEC.005. Therefore, you are recommended to set the connection number to 16.

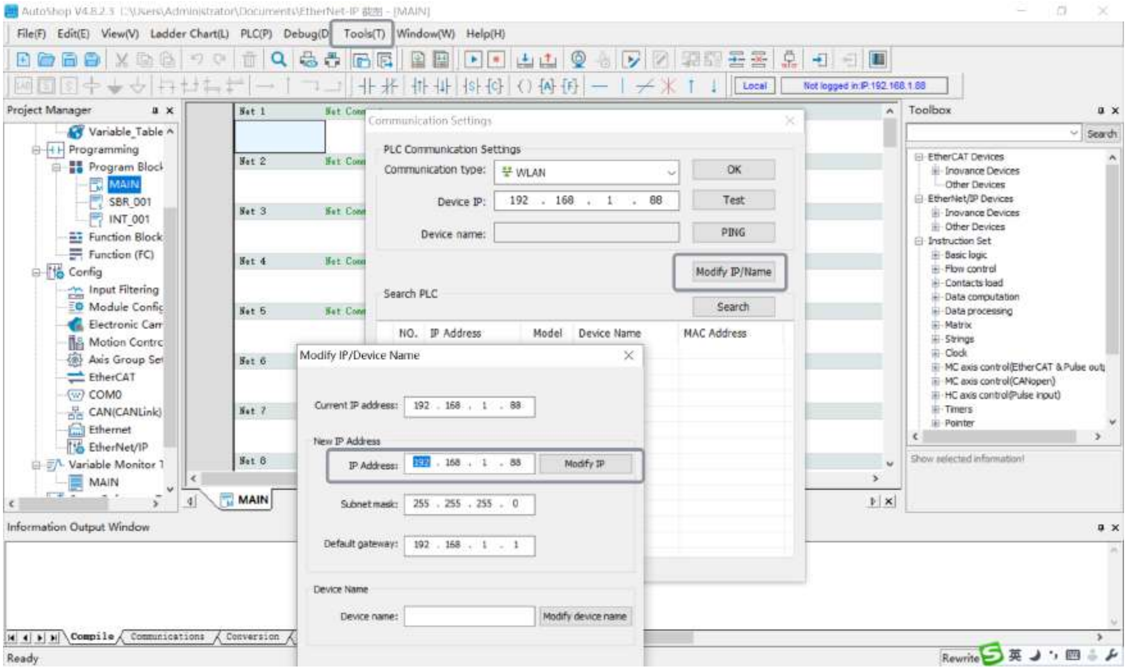
In general, both solution 1 and solution 2 are appropriate for REQ.B after adjustment.

10.3 Class 1 Communication

10.3.1 Master Configuration

10.3.1.1 EtherNet General Settings

1. Log in through the background, connect to the PLC, and configure the PLC gateway, including the IP address, mask, and gateway.


AutoShop V4.8.2.3 C:\Users\Administrator\Documents\EtherNet-IP 截图 - [MAIN]
File(F) Edit(E) View(V) Ladder Chart(L) PLC(P) Debug(D) Tools(T) Window(W) Help(H)
Local
Not logged in:IP 192.168.1.88
Project Manager
Toolbox
Information Output Window
Compile
Communications
Conversion
Ready

Communication Settings

PLC Communication Settings

Communication type: WLAN

Device IP: 192 . 168 . 1 . 88

Device name:

OK | Test | PING | Modify IP/Name

Search PLC

Search

NO.	IP Address	Model	Device Name	MAC Address

Modify IP/Device Name

Current IP address: 192 . 168 . 1 . 88

New IP Address

IP Address: 192 . 168 . 1 . 88 | Modify IP

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 192 . 168 . 1 . 1

Device Name

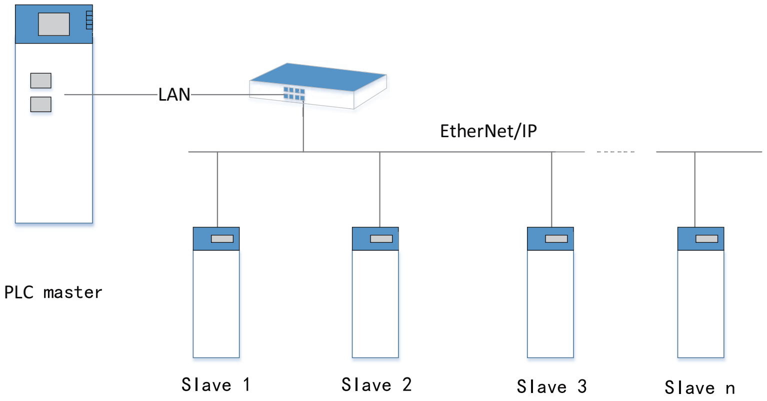
Device name: | Modify device name

2. In the main menu, choose "Tools" > "Communication Settings".
3. In the "Communication Settings" dialog box displayed, click "Search" to search for PLCs in the current network, select the target PLC, and click "Test" to check whether the PLC can be connected.

10.3.1.2 EtherNet/IP Device IP Settings

EtherNet/IP supports bus topology, star topology, and bus-star topology. In a star topology, all nodes are connected to the network hub, and nodes are easily added, deleted, and maintained. Such topology is often used due to its cost-effectiveness, easy connection, and availability of required devices.

IP addresses of all devices must be unique and in the same EtherNet/IP network segment. The following figure shows a star topology.



IP address of the EtherNet/IP master station: 192.168.1.100

IP address of the EtherNet/IP slave station 1: 192.168.1.101

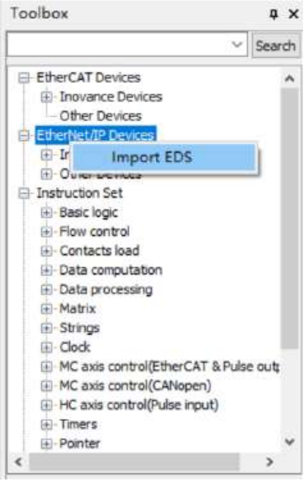
IP address of the EtherNet/IP slave station 2: 192.168.1.102

...

IP address of the EtherNet/IP slave station n: 192.168.1.XXX

10.3.1.3 Adding EtherNet/IP Slaves

1. Import the EDS description file of the slave station.
   Create a master station project. Right-click "EtherNet/IP Devices" under "Toolbox". In the menu displayed, click "Import EDS". In the dialog box displayed, select the EDS file and then click "Open". The following figure takes importing the EDS file of KEYENCE N-L20 EtherNet/IP as an example.



Note

To synchronize with the latest EDS file of a slave when there is any change to the EDS file, expand the "EtherNet/IP" menu, right-click the slave, and select "Update EDS" in the shortcut menu. In this process, select the option that the EDS file exported from the slave overwrites the original file in the directory. See *"10.3.1.4 Exporting EDS Files" on page 338* for details.

2. Create the configuration. You can create a configuration in AutoShop in two ways.
   ①: Locate the device file you want to import, and double-click the file. The device is added to the network configuration. You can add multiple EtherNet/IP slave stations one by one.

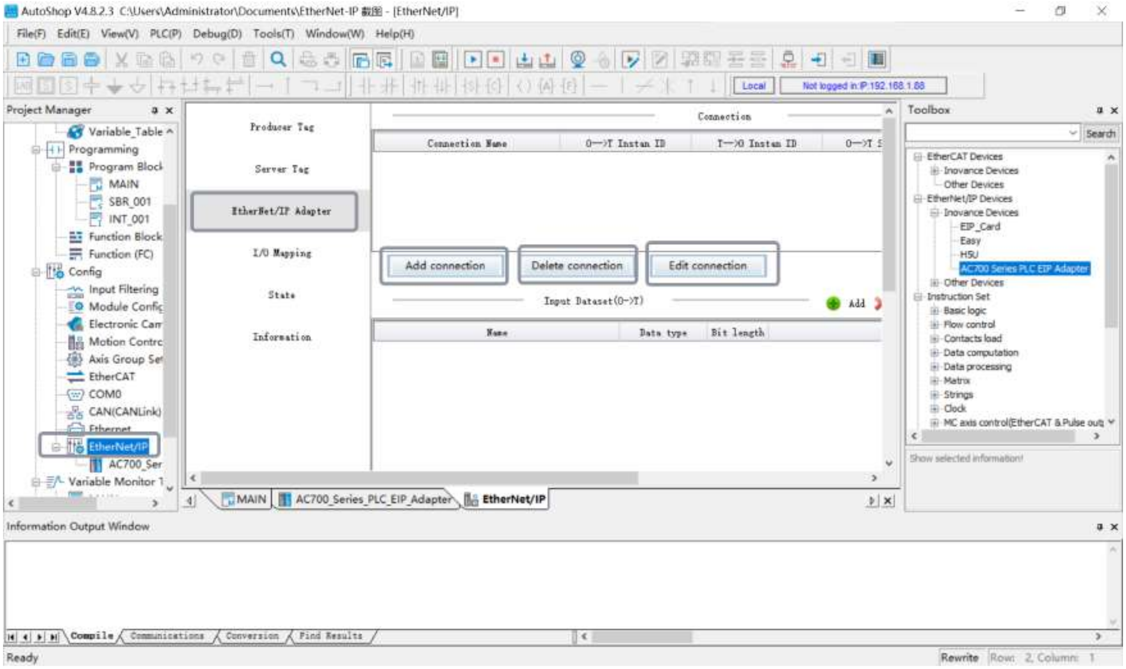
   ②: Scan a device to add it to the configuration. When AutoShop and PLC are connected, you can click "Auto Scan" to scan EtherNet/IP devices connected to the PLC. The procedure is as follows.

   a. Right-click "EtherNet/IP" in the project tree. In the menu displayed, click "Auto Scan".
   b. In the "Auto Scan" dialog box displayed, slave stations configured for the current project are displayed on the left. The names of the slave stations scanned out by the PLC after you click "Auto Scan" are displayed on the right. The slave stations in red are devices for which no matching EDS file is found in AutoShop, while configurations of slave stations in black can be updated.
   c. Click "Update Config". The system asks you to confirm whether to save the current configuration. After you select "Yes", slave stations whose configurations can be updated are added to the current configuration. If you select "No", all EtherNet/IP configurations are deleted and the slave stations whose configurations can be updated are added.

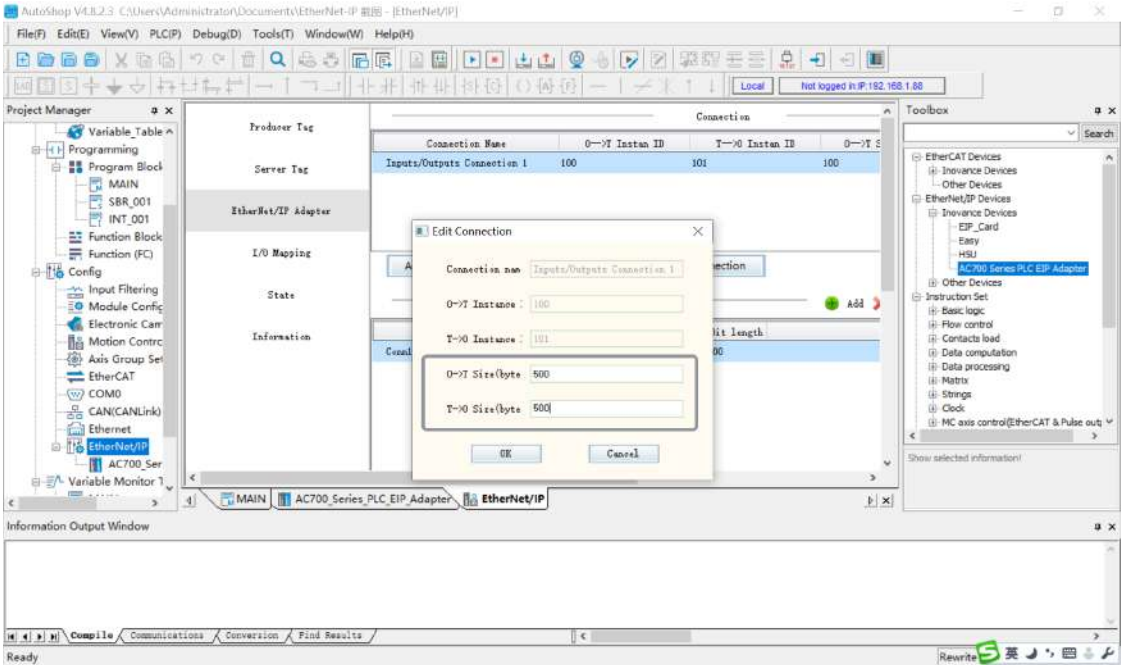
10.3.1.4 Exporting EDS Files

The way to export EDS files is basically the same for Easy and H5U. The main difference is that Easy supports a maximum of 1400 bytes while H5U supports a maximum of 500 bytes.

1. Double-click "EtherNet/IP", and then select "EIP Adapter". On the EIP slave station page displayed, you can add or delete connections (up to 16 connections are supported), as shown in the following figure.



2. Click "Edit connection" to edit the O->T and T->O sizes for connections. The default value is 500 bytes.



Connection Name	O->T Instan ID	T->O Instan ID	O->T S
Inputs/Outputs Connection 1	100	101	100

Edit Connection

Connection nam: Inputs/Outputs Connection 1

O->T Instance: 100

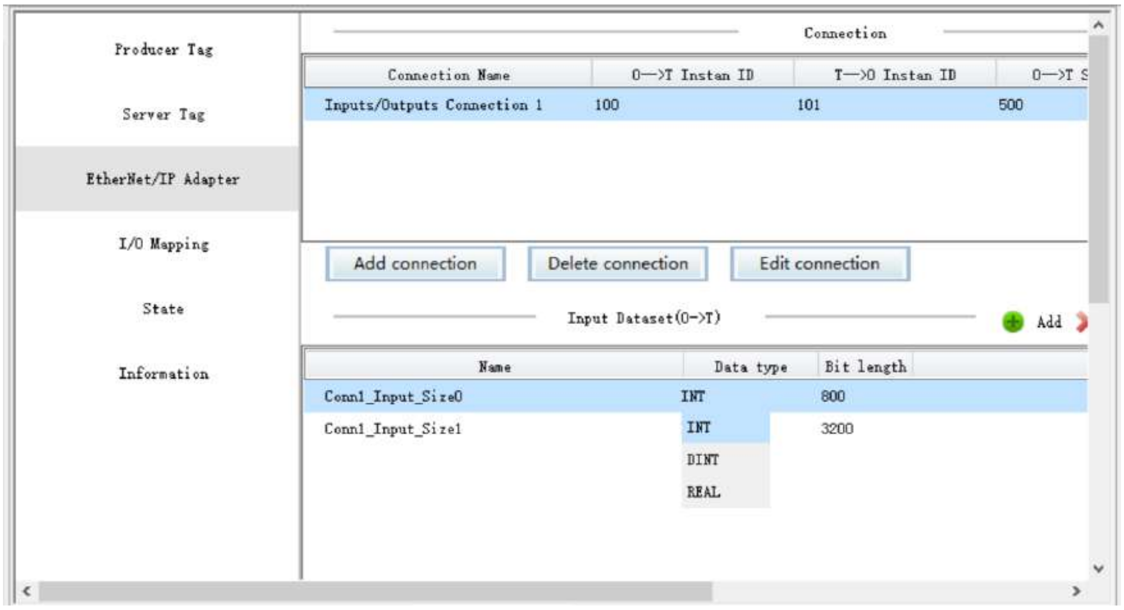
T->O Instance: 101

O->T Size(byte: 500

T->O Size(byte: 500

AutoShop V4.8.2.3 C:\Users\Administrator\Documents\EtherNet-IP 截图 - [EtherNet/IP]
File(F) Edit(E) View(V) PLC(P) Debug(D) Tools(T) Window(W) Help(H)
Project Manager
Producer Tag
Server Tag
EtherNet/IP Adapter
I/O Mapping
State
Information
Connection
OK
Cancel
Add
Toolbox
AC700 Series PLC EIP Adapter
MAIN
AC700_Series_PLC_EIP_Adapter
EtherNet/IP
Information Output Window
Compile
Communications
Conversion
Find Results
Ready
Rewrite

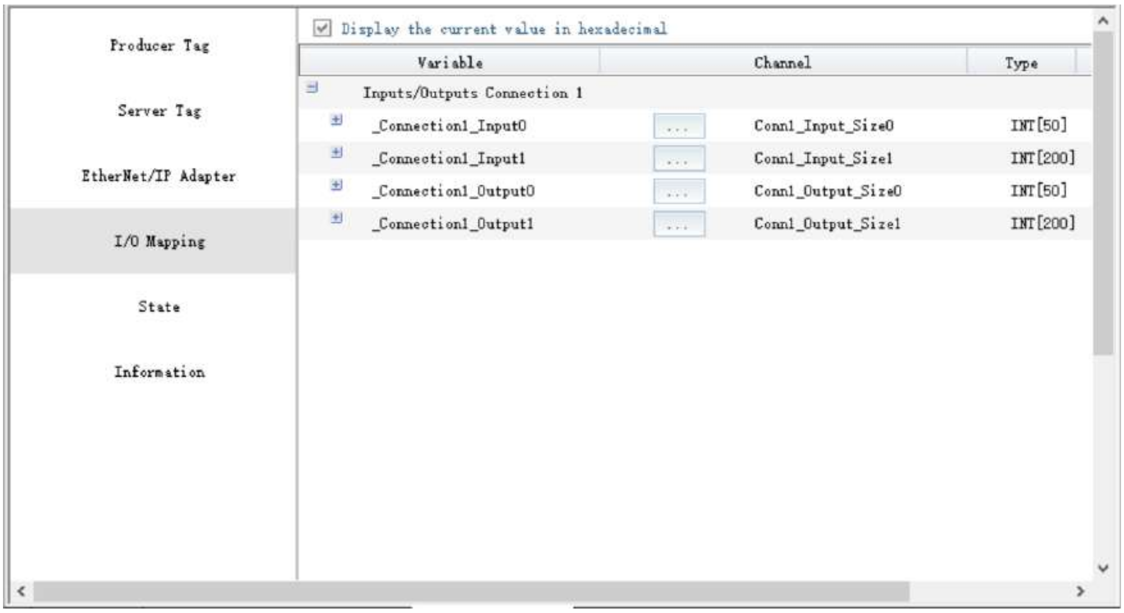
3. You can also split the size of bytes according to the data set of O->T and T->O, and split the total bytes into multiple variables of the required byte size for mapping. You can add or delete a data set to adjust the byte size, or move a data set up or down.



Connection Name	O—>T Instan ID	T—>O Instan ID	O—>T S
Inputs/Outputs Connection 1	100	101	500

Name	Data type	Bit length
Conn1_Input_Size0	INT	800
Conn1_Input_Size1	INT DINT REAL	3200

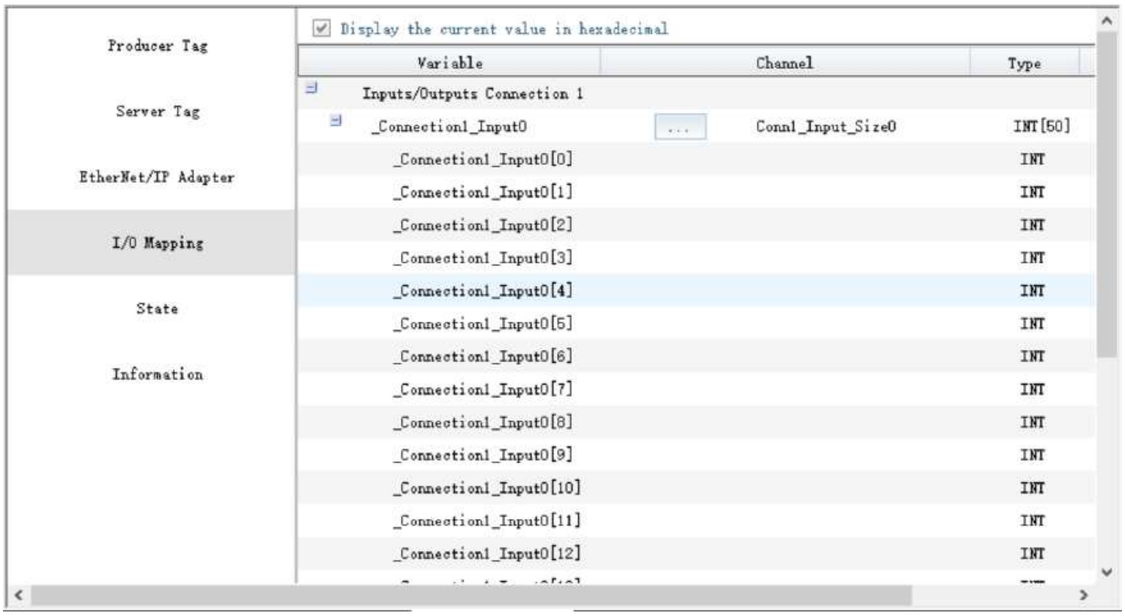
4. Split bytes will generate multiple corresponding variable which can be viewed on the "IO Mapping" page. The corresponding bytes will generate the corresponding variable numbers, and the byte size will be that of the connection.



☑ Display the current value in hexadecimal

Variable		Channel	Type
Inputs/Outputs Connection 1			
_Connection1_Input0	...	Conn1_Input_Size0	INT[50]
_Connection1_Input1	...	Conn1_Input_Size1	INT[200]
_Connection1_Output0	...	Conn1_Output_Size0	INT[50]
_Connection1_Output1	...	Conn1_Output_Size1	INT[200]

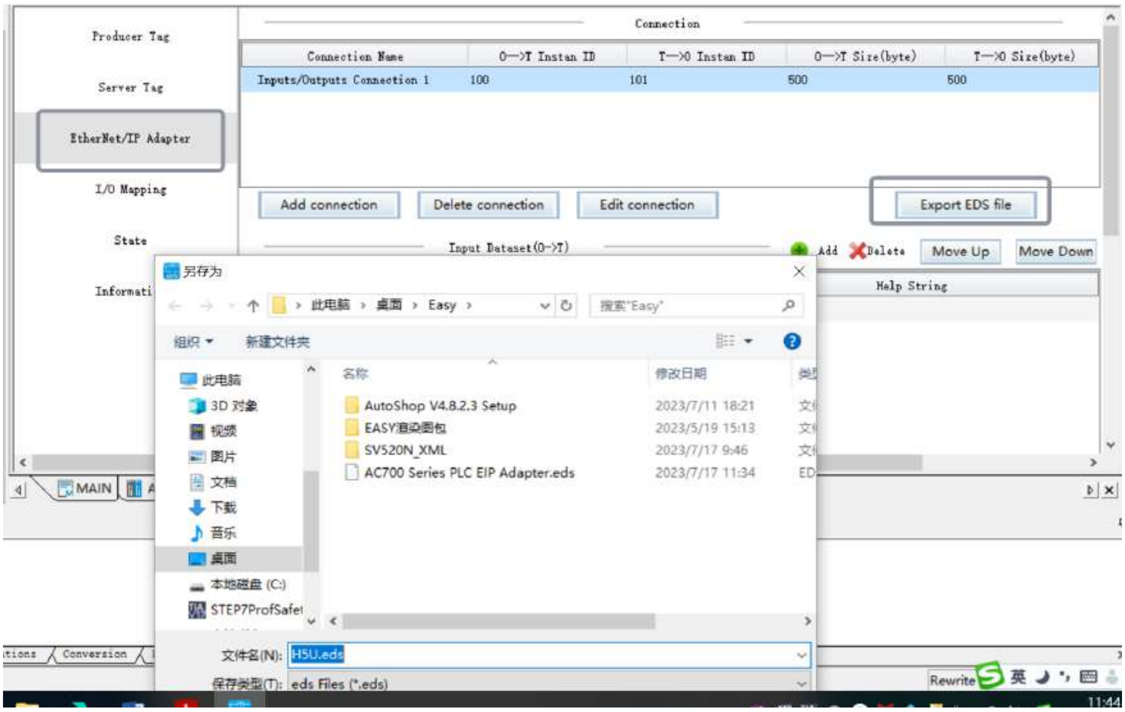
5. You can expand and view an array.



☑ Display the current value in hexadecimal

Variable		Channel	Type
Inputs/Outputs Connection 1			
_Connection1_Input0	...	Conn1_Input_Size0	INT[50]
_Connection1_Input0[0]			INT
_Connection1_Input0[1]			INT
_Connection1_Input0[2]			INT
_Connection1_Input0[3]			INT
_Connection1_Input0[4]			INT
_Connection1_Input0[5]			INT
_Connection1_Input0[6]			INT
_Connection1_Input0[7]			INT
_Connection1_Input0[8]			INT
_Connection1_Input0[9]			INT
_Connection1_Input0[10]			INT
_Connection1_Input0[11]			INT
_Connection1_Input0[12]			INT

6. Select parameters, switch to the "EtherNet/IP" page, and click "Export EDS file" to export the EDS file to any folder.



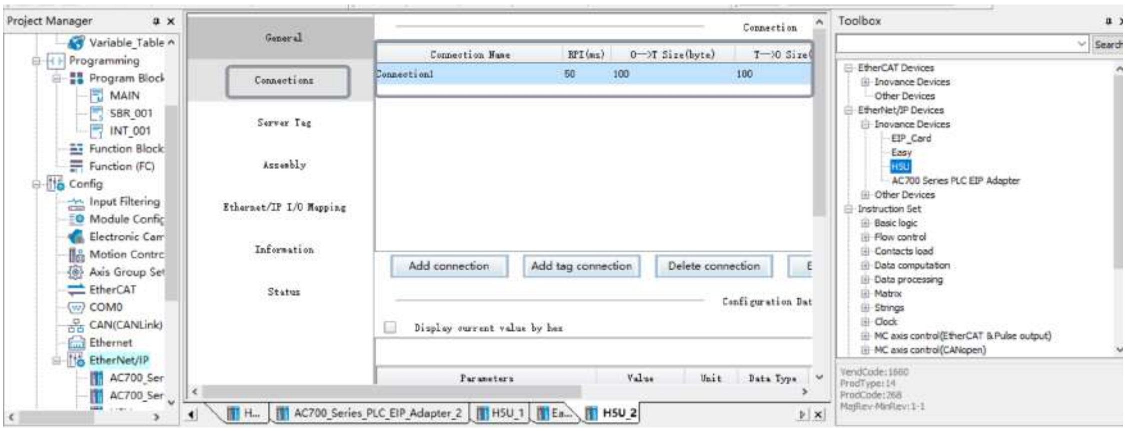
Connection Name	O—>T Instan ID	T—>O Instan ID	O—>T Size(byte)	T—>O Size(byte)
Inputs/Outputs Connection 1	100	101	500	500

名称	修改日期	类型
AutoShop V4.8.2.3 Setup	2023/7/11 18:21	文
EASY渲染图包	2023/5/19 15:13	文
SV520N_XML	2023/7/17 9:46	文
AC700 Series PLC EIP Adapter.eds	2023/7/17 11:34	ED

文件名(N): H5U.eds

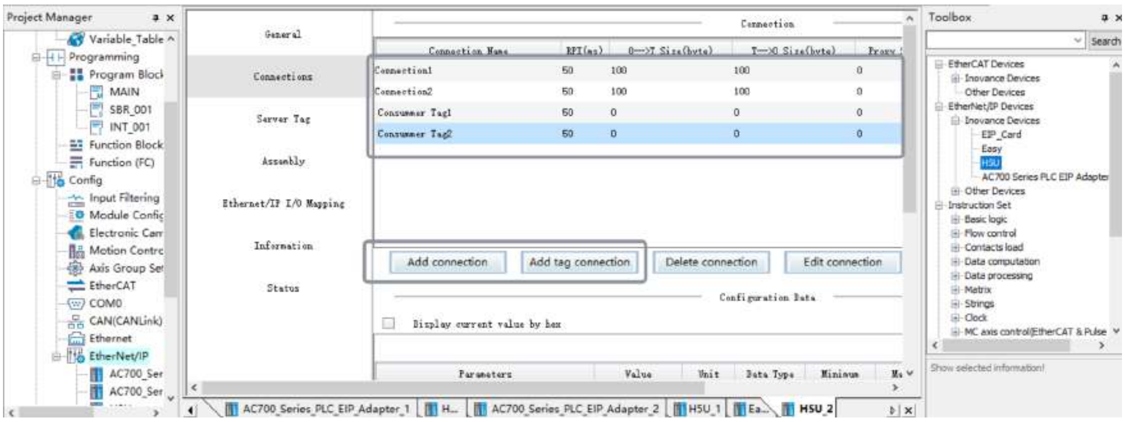
保存类型(T): eds Files (*.eds)

7. To import the exported EDS file to the background, choose "EtherNet/IP Devices" under "Toolbox", and double-click the H5U.eds file. On the following page displayed, select the I/O connection from the drop-down list of "Connection name"。



Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(
Connection1	50	100	100

8. You can click "Add connection" or "Add label connection" to add the required connection. "Add connection" is used to add an I/O connection while "Add label connection" is used to add a consumer connection.



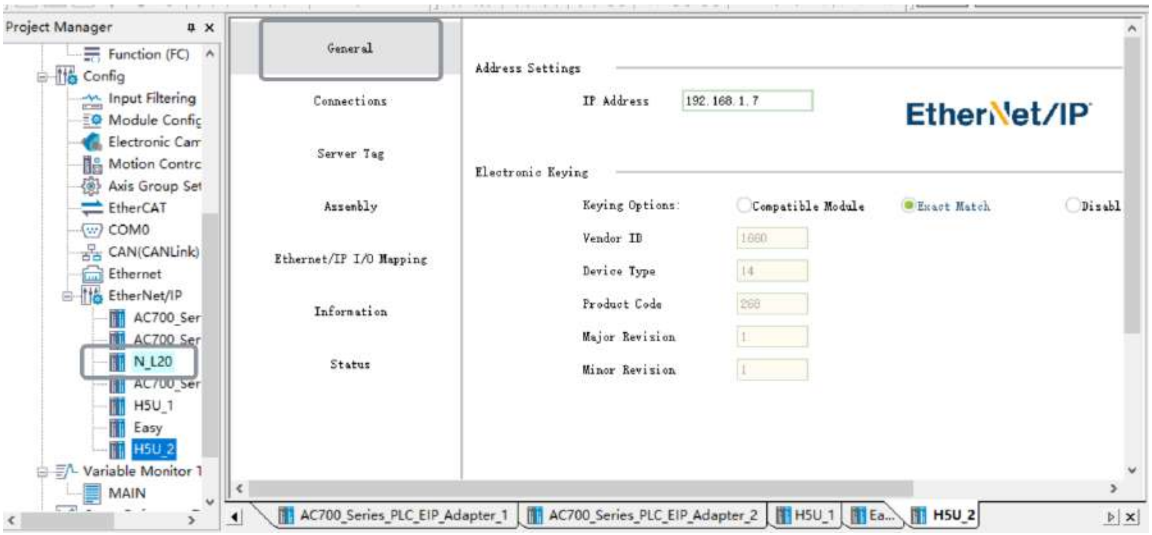
Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(byte)	Proxy
Connection1	50	100	100	0
Connection2	50	100	100	0
Consumer Tag1	50	0	0	0
Consumer Tag2	50	0	0	0

10.3.2 Slave Configuration

10.3.2.1 General Settings

Modifying the configuration of a single slave station

Double-click the EtherNet/IP slave station "N-L20" you want to set. On the page displayed, set the IP address and matching options for the slave station.



Address Settings

IP Address: 192.168.1.7

Electronic Keying

Keying Options: ○ Compatible Module ● Exact Match ○ Disabl

Vendor ID: 1660

Device Type: 14

Product Code: 268

Major Revision: 1

Minor Revision: 1

Electronic match: The EtherNet/IP master station checks whether the following fields of the EtherNet/IP slave station match those in the EDS file: supplier code, device type, product code, major version, and minor version. The following table lists the matching options.

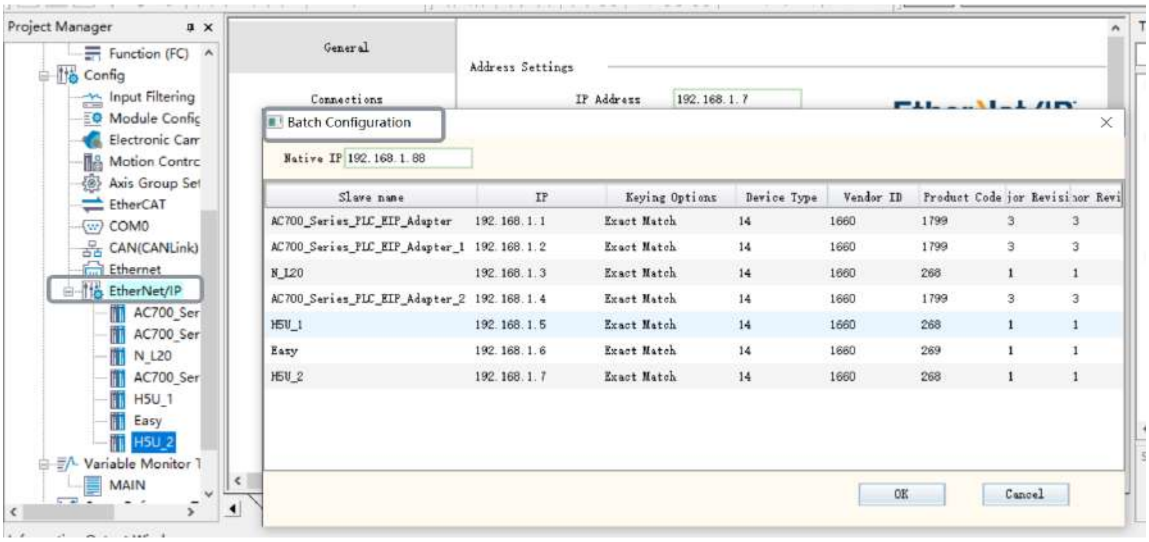
Parameter Name	Description
Compatible Module	Conditional matching. The communication is established only when the supplier code, device type, and product code are identical and the major version and minor version of the slave station are equal to or later than those in the EDS file (the version of the slave station is equal to or later than the version in the EDS file, and the slave station is compatible with EDS files of earlier versions).
Exact Match	Communication can be established between EtherNet/IP master and slave stations only when all of such fields are exactly matched.
Stop check[1]	Communication is directly established between EtherNet/IP master and slave stations without checking whether such fields are matched.

⚠ Warning

[1]: Exercise special caution when selecting "Stop check". Improper use of this option may cause physical injury, property damage, or economic loss. It is strongly advised against using the "Stop check" option.

Modifying configurations of multiple slave stations

When multiple slave stations are configured, right-click "EtherNet/IP" in the "Project Manager" section. In the menu displayed, select "Batch Configuration". In the "Batch Configuration" dialog box, modify the IP addresses and matching options of slave stations in a batch.



IP Address 192.168.1.7

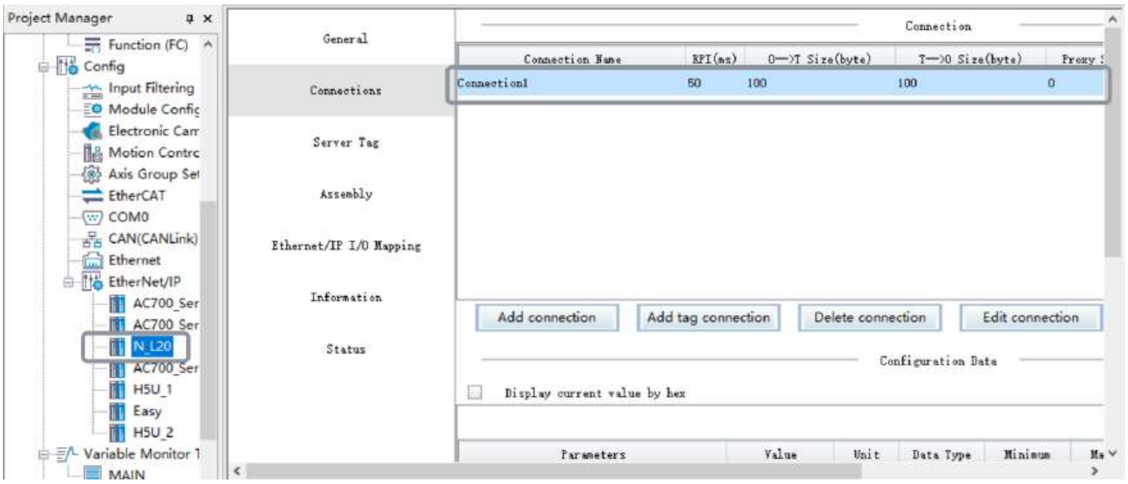
Native IP 192.168.1.88

Slave name	IP	Keying Options	Device Type	Vendor ID	Product Code	jor Revisi	nor Revi
AC700_Series_PLC_EIP_Adapter	192.168.1.1	Exact Match	14	1660	1799	3	3
AC700_Series_PLC_EIP_Adapter_1	192.168.1.2	Exact Match	14	1660	1799	3	3
N_L20	192.168.1.3	Exact Match	14	1660	268	1	1
AC700_Series_PLC_EIP_Adapter_2	192.168.1.4	Exact Match	14	1660	1799	3	3
H5U_1	192.168.1.5	Exact Match	14	1660	268	1	1
Easy	192.168.1.6	Exact Match	14	1660	269	1	1
H5U_2	192.168.1.7	Exact Match	14	1660	268	1	1

10.3.2.2 Connection Settings

1. Add a connection.

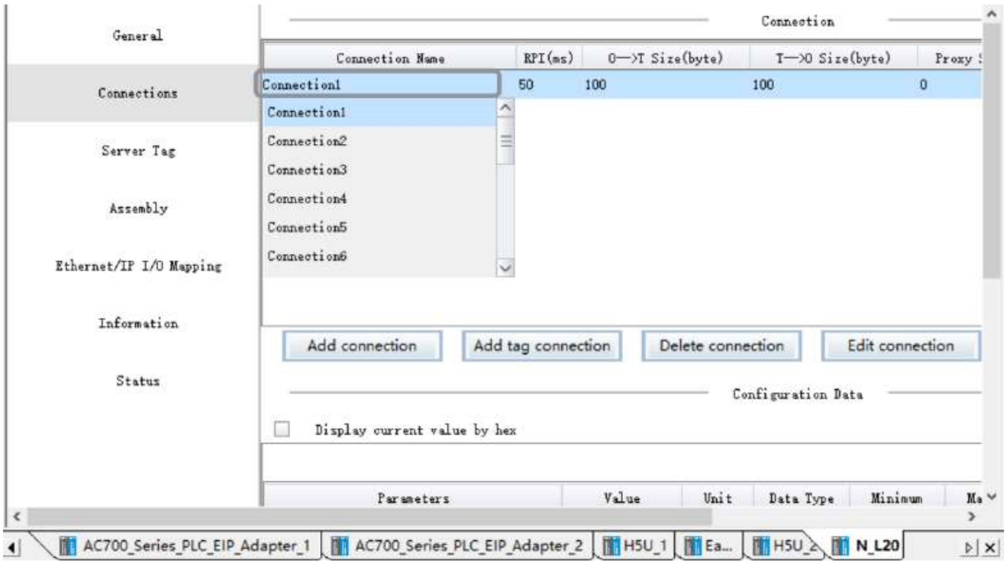
The EDS description file of an EtherNet/IP slave station contains a default connection path. After the EtherNet/IP network configuration is added, the background connection page loads the default connection path, as shown in the following figure.



Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(byte)	Proxy S
Connection1	50	100	100	0

□ Display current value by hex

You can also click the connection name and set the connection path pre-defined in the EDS file, as shown in the following figure.

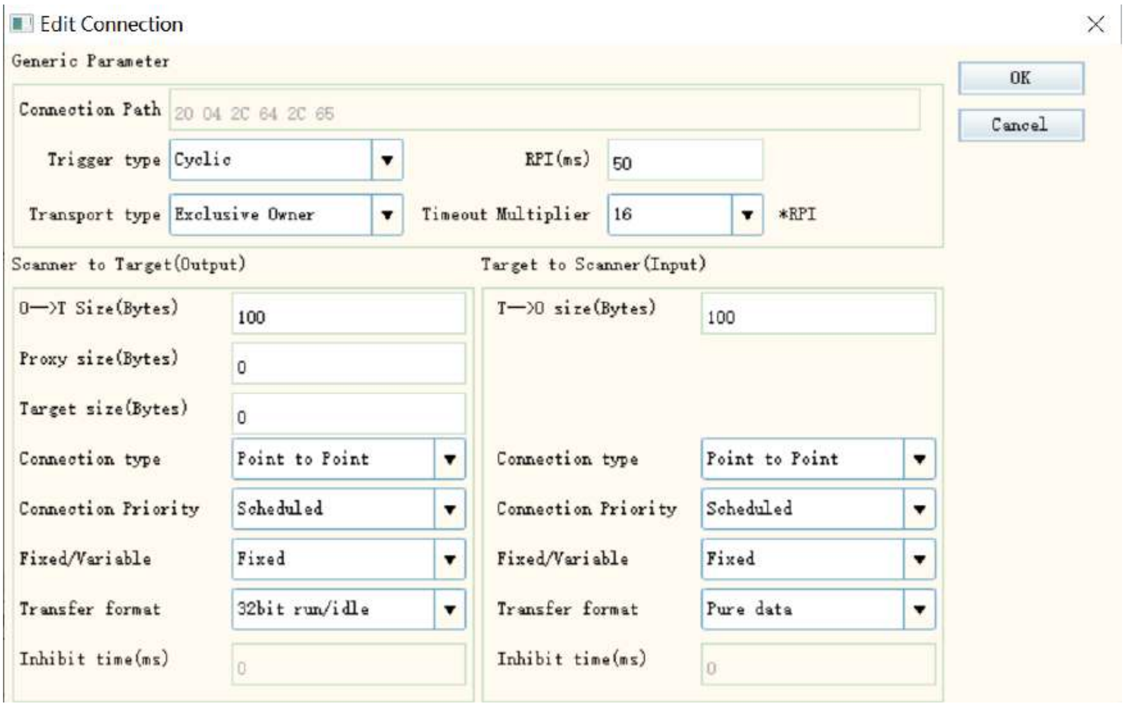


Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(byte)	Proxy S
Connection1	50	100	100	0

□ Display current value by hex

Parameters	Value	Unit	Data Type	Minimum	Ma

On the preceding page, click "Edit connection" to enter the connection setting page. Generally, all parameters other than RPI (communication cycle) of the connection must use the default values.



Generic Parameter

Connection Path: 20 04 2C 64 2C 65

Trigger type: Cyclic   RPI(ms): 50

Transport type: Exclusive Owner   Timeout Multiplier: 16 *RPI

Scanner to Target(Output)		Target to Scanner(Input)	
O—>T Size(Bytes)	100	T—>O size(Bytes)	100
Proxy size(Bytes)	0		
Target size(Bytes)	0		
Connection type	Point to Point	Connection type	Point to Point
Connection Priority	Scheduled	Connection Priority	Scheduled
Fixed/Variable	Fixed	Fixed/Variable	Fixed
Transfer format	32bit run/idle	Transfer format	Pure data
Inhibit time(ms)	0	Inhibit time(ms)	0

2. Configure the general parameters.

- Connection path: Specifies the format and connection instance of a byte stream frame.
  For example, 20 04 2C C6 2C 68 (for details, see EIP-CIP-V1-1.0, Appendix C: Data Management).

  20: Logical Segment, ClassID, 8-bit logical address

  04: Assembly Object (04H)

  2C: Logical Segment, Connection Point, 8-bit logical address

  C6: ID-C6H of the Assembly Object instance

  2C: Logical Segment, Connection Point, 8-bit logical address

  68: ID-68H of the Assembly Object instance

Note: The connection path must be configured based on the guide of the specific slave station. The connection path varies with the manufacturer.

- RPI (MS): Requested Packet Interval. It indicates the communication transmission interval in ms. The RPI of each node can be set individually without affecting each other.

3. Scan the target.

- Transmission byte size
  1) O->T Size (Bytes): Indicates the amount of data transferred from the producer (scanner) to the consumer (target device), in bytes.

  2) T->O Size (Bytes): Indicates the amount of data transferred from the consumer (target device) to the producer (scanner), in bytes.

- Transport Type
  Exclusive Owner: Allows users to set both data sending from the initiator to the target device and data receiving from the target device to the initiator.

  Redundant Owner: Allows multiple initiators to create independent and identical connections to the same target device.

  Input Only: This connection can only be used to set data receiving from the target device to the initiator.

  Listen Only: EtherNet/IP devices apply this type of connections to listen to multicast data without providing configuration or scheduling information.

- Trigger Type
  Cyclic: Periodically triggers data transmission.

  Change-Of-State: Transmits data when a change in the state of the application object is detected.

  Application Object: Transmits data when the application object is triggered.

- Connection Type
  Multicast: Multiple scanners receive data from one target device at the same time.

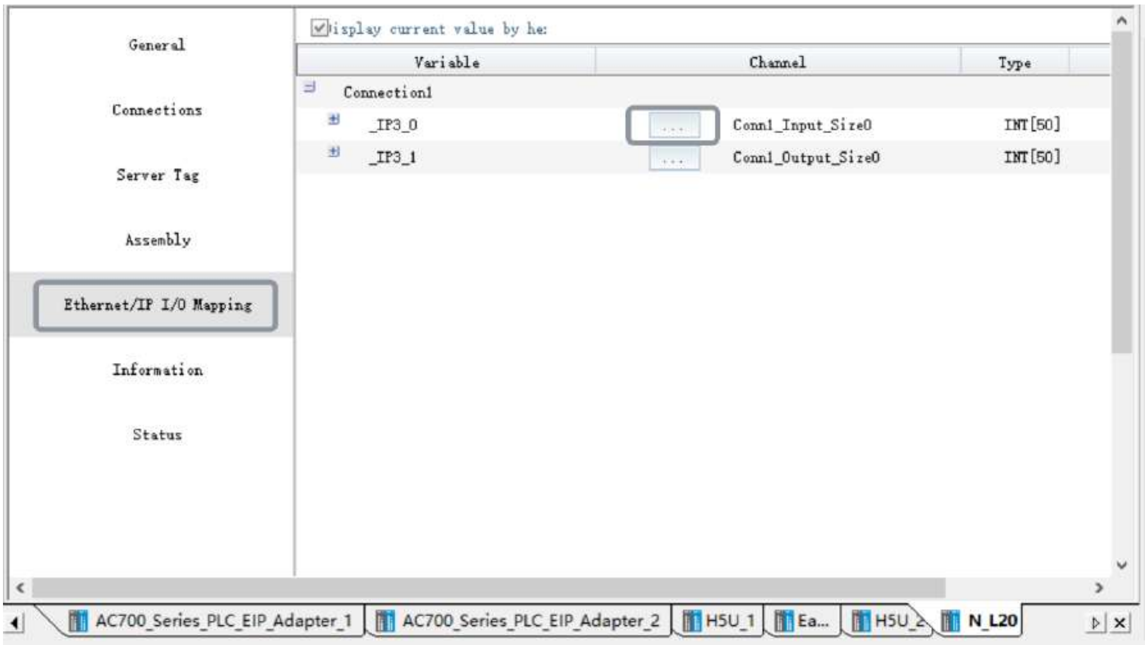
  Point-to-Point: One scanner can receive data from only one target device.

Note

Click "Add connection" to open the connection setting interface and select "Universal connection" to customize a connection as needed. This requires knowledge of the CIP protocol. For users without such knowledge, it is recommended to use the default connection configuration.

10.3.2.3 Configuring I/O Variable Mapping

You can expand an array when modifying a variable mapping. For example, to create a connection, of which the O->T size of the consumer tag is 10 bytes, the number of server tags is 5, and _IP2_0 and _IP2_1 are variables of the default mapping, click "..." and input the variables at the variable name position to modify the mapping.



☑Display current value by he:

Variable	Channel	Type
Connection1		
_IP3_0	Conn1_Input_Size0	INT[50]
_IP3_1	Conn1_Output_Size0	INT[50]

Only three data types are displayed: INT, DINT, and RAL. If you have modified the data type of the consumer tag through the data set, the data type is converted to any of these three types. The correspondence is as follows.

INT: BYTE, INT, UINT, SINT, and USINT are all displayed as INT.

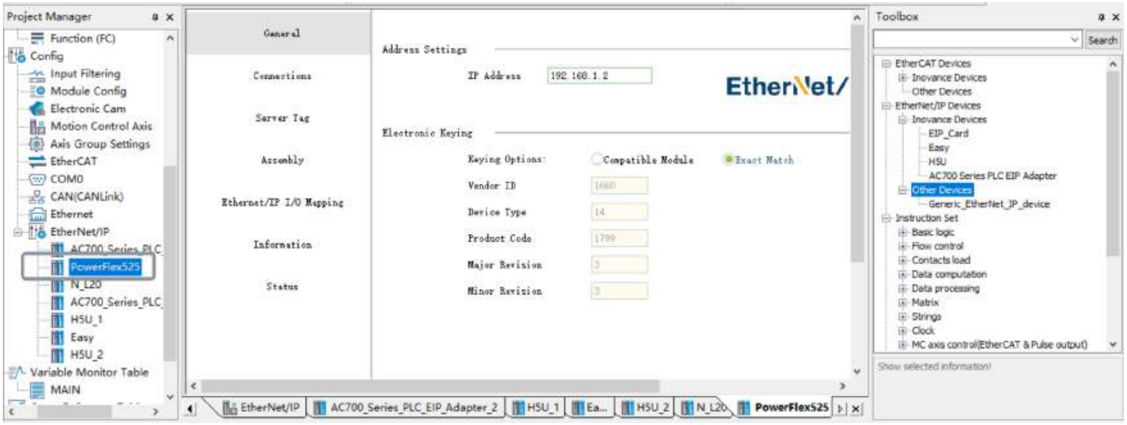
DINT: DINT, UDINT, LINT, ULINT, LWORD, DWORD, and WORD are all displayed as DINT.

REAL: REAL and LREAL are displayed as REAL.

10.3.3 EtherNet/IP Master Application Example

This project describes how to create an EtherNet/IP network in which H5U serves as the master station. In this example, the slave station is the Tockwell PowerFlex 525 AC drive.

1. Under "EtherNet/IP Devices" of "Toolbox", select the corresponding slave station for I/O communication.



Address Settings

IP Address: 192.168.1.2

Electronic Keying

Keying Options: ○Compatible Module ●Exact Match

Vendor ID: 1660

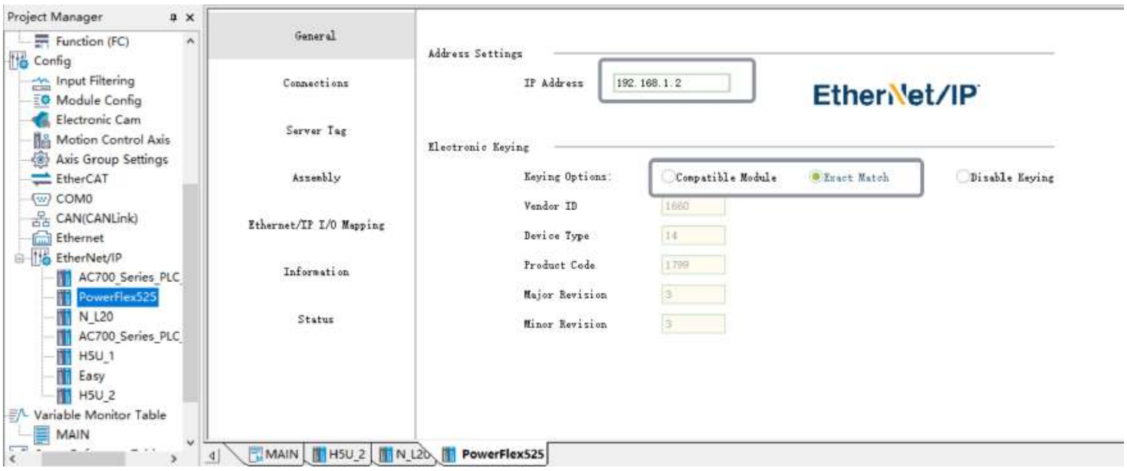
Device Type: 14

Product Code: 1799

Major Revision: 3

Minor Revision: 3

2. Set the IP address of the slave station you want to connect.



Address Settings

IP Address: 192. 168. 1. 2

Electronic Keying

Keying Options: ○ Compatible Module ● Exact Match ○ Disable Keying

Vendor ID: 1660

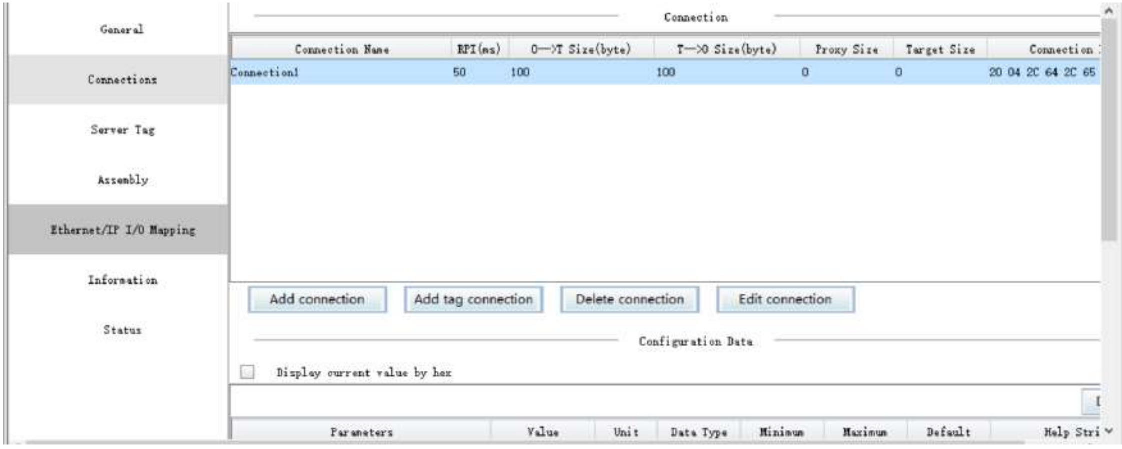
Device Type: 14

Product Code: 1799

Major Revision: 3

Minor Revision: 3

3. Set the connection parameters. The PowerFlex 525 AC drive supports only the default connection and therefore only one connection is listed under "Connection name". You can directly change the RPI. In this case, RPI is 50 ms and cannot be modified.



Connection

Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(byte)	Proxy Size	Target Size	Connection :
Connection1	50	100	100	0	0	20 04 2C 64 2C 65

Configuration Data

□ Display current value by hex

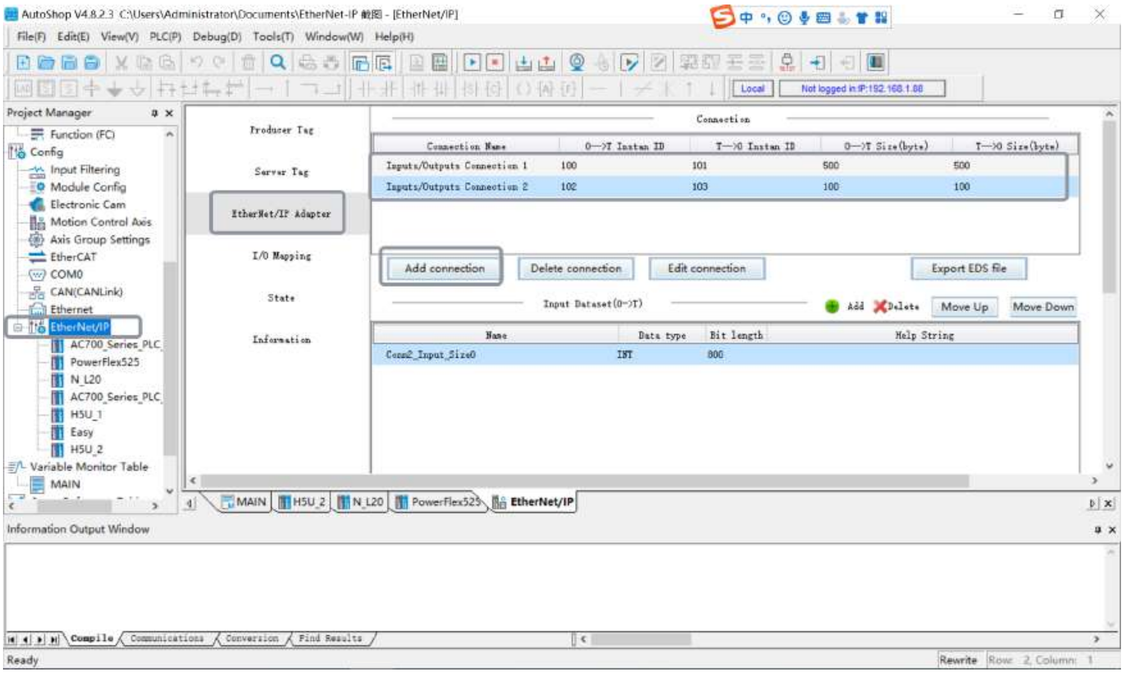
Parameters	Value	Unit	Data Type	Minimum	Maximum	Default	Help Stri

4. After making the configuration, download the program, run the PLC, and monitor the communication status.

The communication details can be viewed on the status page.

10.3.4 EtherNet/IP Slave Application Example

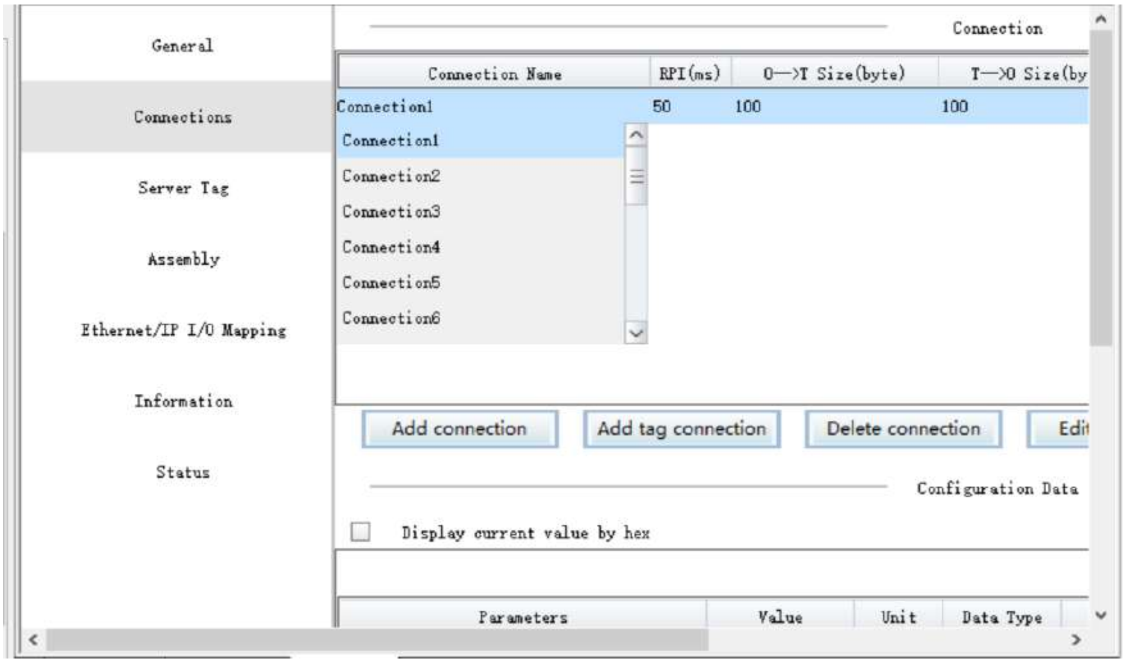
1. Add two connections, and modify their O->T size to 100 bytes and T->O size to 100 bytes. Export the EDS file, set the IP address of the connected PLC to 10.45.124.150, compile the project, and then download and run the project.



Connection Name	O->T Instan ID	T->O Instan ID	O->T Size(byte)	T->O Size(byte)
Inputs/Outputs Connection 1	100	101	500	500
Inputs/Outputs Connection 2	102	103	100	100

Name	Data type	Bit length	Help String
Conn2_Input_Size0	INT	800	

2. Import the exported EDS file. Double-click "H5U" under "EtherNet/IP" in the "Toolbox" section, add a slave station, and set the IP address of the connected PLC to 10.45.124.77.

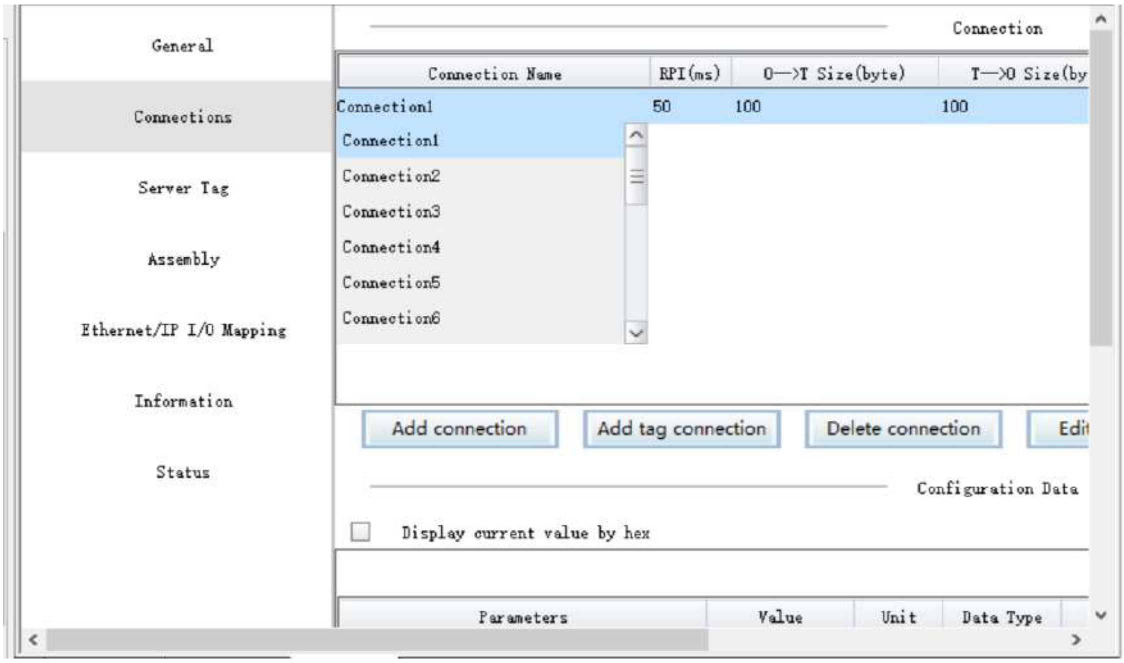


Connection Name	RPI(ms)	O->T Size(byte)	T->O Size(by
Connection1	50	100	100

□ Display current value by hex

Parameters	Value	Unit	Data Type

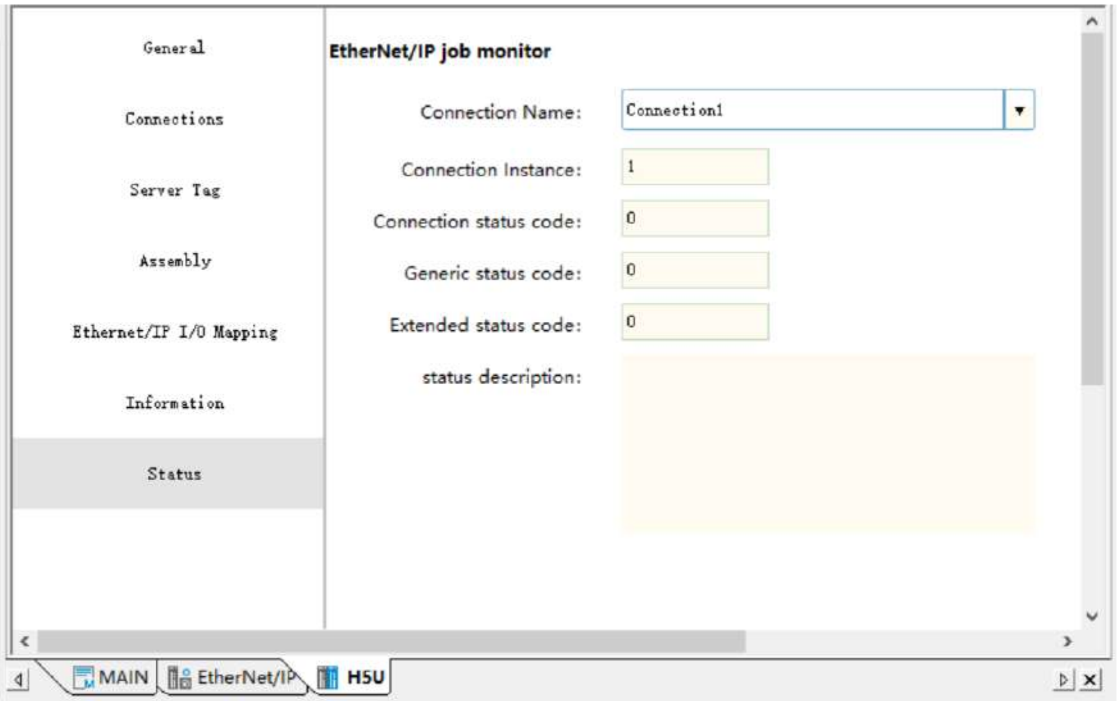
3. After the file is imported, "Connection1" is selected by default, whose O->T and T->O sizes are those you set when creating the connection.
You can click "Add connection" to add a custom I/O connection or click "Add label connection" to add a consumer tag connection. You can click "Connection name" to unfold the drop-down list which displays connections defined in the EDS file. You can switch, edit, and delete connections in the drop-down list.



Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(by
Connection1	50	100	100

Connection1
Connection2
Connection3
Connection4
Connection5
Connection6

4. On the "General" tab page, set the IP address to 10.45.124.150, and then export and run the project.



EtherNet/IP job monitor

Connection Name: Connection1

Connection Instance: 1

Connection status code: 0

Generic status code: 0

Extended status code: 0

status description:

5. If a green icon is displayed for the H5U node in the project tree, the node is connected. On the status page on which the connection status of "Connection1" is successful.

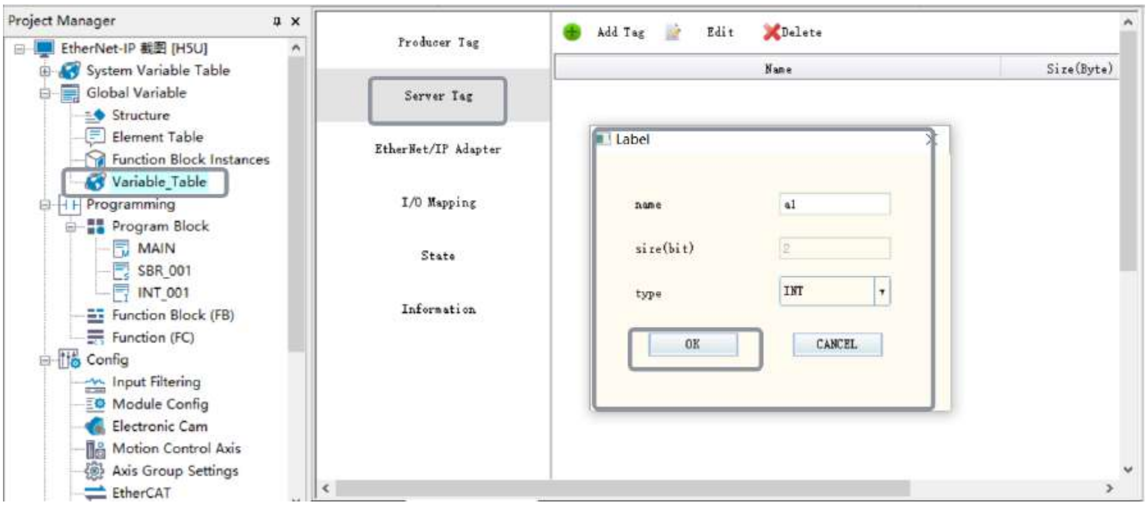
10.3.5 Tag Communication

10.3.5.1 Configuring Producer Tag Data

Configuring the producer tag data on the "Producer label" page

1. In the "Project Manager" section, unfold "Config", and double-click "EtherNet/IP" to open the "Producer label" page. Supported producer tag types are INT, DINT, REAL, and array.
2. To create a producer tag, create a variable in the variable table. On the "Producer label" page, click "Add label", input the created variable, and then click "OK".

For data of the type INT, DINT, or REAL, you can input a name to create a tag on the "Producer label" page. Then, AutoShop automatically displays the variable creation dialog box, on which you can create the variable, as shown in the following figure.



name: a1
size(bit): 2
type: INT

Directly configuring the producer tag data in the variable table

In the variable table, modify the "Network Public" property of the variable to "IN/OUT". Then, the variable is configured as a producer tag.

10.3.5.2 EtherNet/IP Consumer Tag Connection

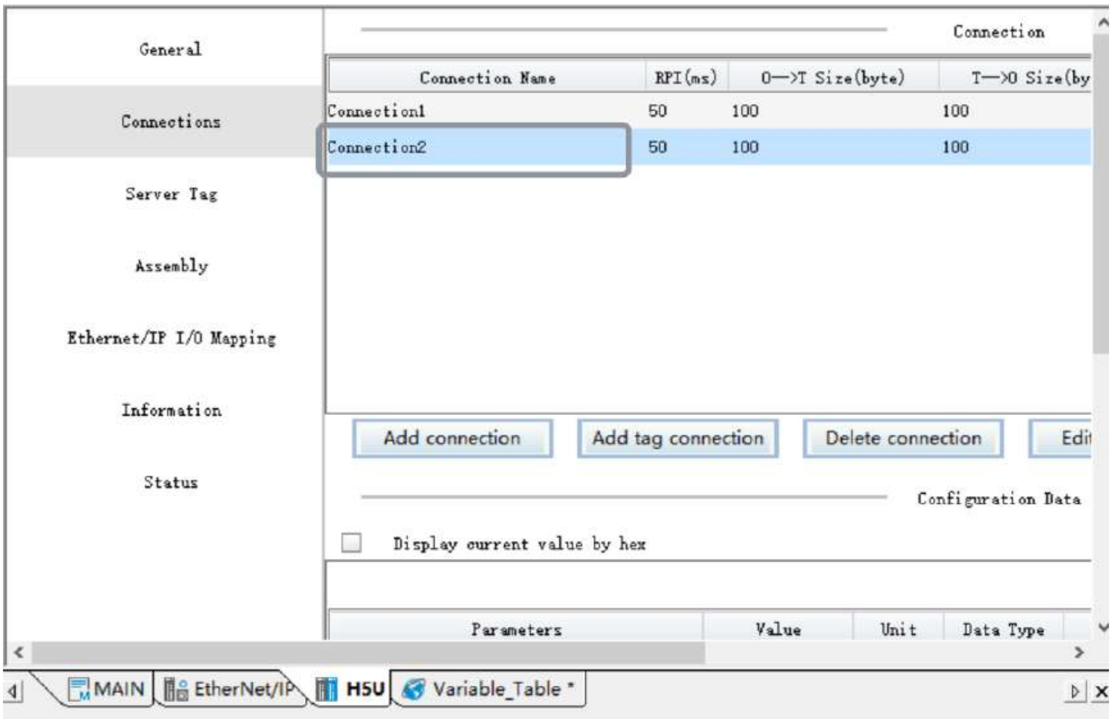
Configuring an EtherNet/IP consumer tag connection

Note

Before configuring a consumer or producer, add a slave and select a tag from the slave.

Take the EDS file of the Inovance slave station as an example. In the toolbox, import the AM600_400 Series PLC EIP Adapter.eds file, add the configuration, modify the IP address, and then access the "Connection" page, on which a connection is loaded from the EDS file by default.

If the connection is a consumer tag connection, the "Add connection" button is activated and can be used to add a custom consumer tag connection. You can also click "Edit connection" to modify the RPI, T->O size, and connection path, wherein the connection path is the consumer tag name.



Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(by
Connection1	50	100	100
Connection2	50	100	100

□ Display current value by hex

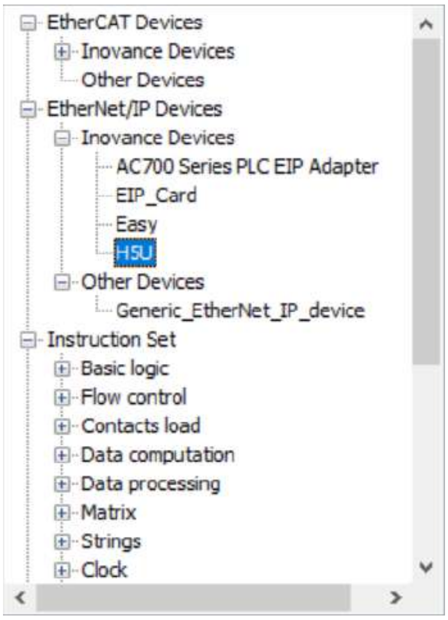
Parameters	Value	Unit	Data Type

Methods for adding a slave station

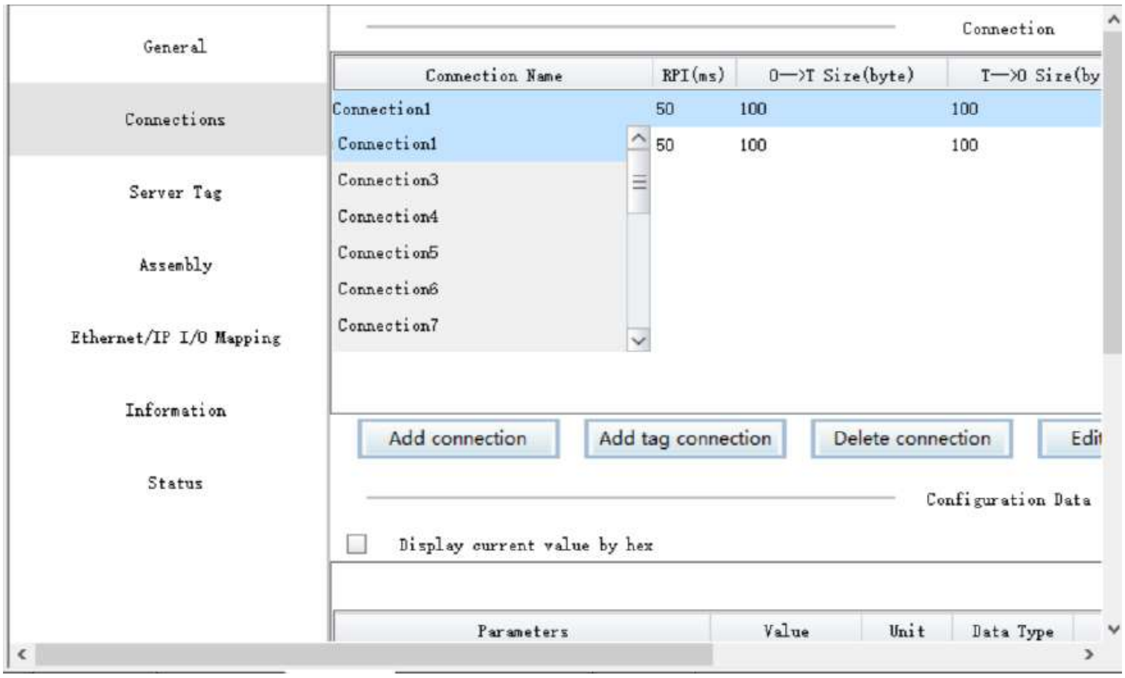
1. Import the EDS file

The following describes how to add a slave station by taking the EDS file of the Inovance slave station as an example.

1. Right-click "EtherNet/IP Devices" in the "Toolbox" section, click "Import EDS file", and select the corresponding EDS file.
2. After the EDS file is imported, double-click the EDS file, and add the slave station corresponding to EIP.



3. Select "Connection" and click the corresponding connection.



Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(by
Connection1	50	100	100
Connection1	50	100	100

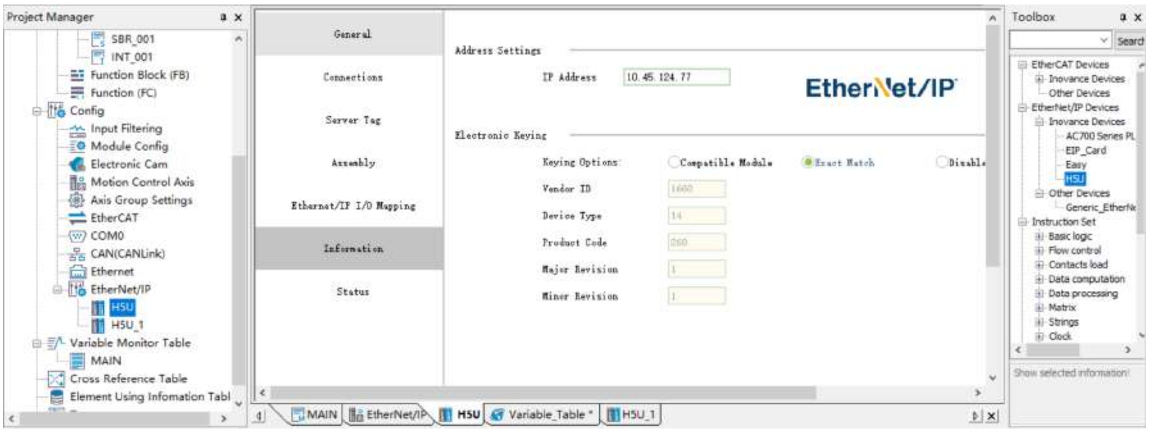
Connection3
Connection4
Connection5
Connection6
Connection7

□ Display current value by hex

Parameters	Value	Unit	Data Type

2. Use the generic template Generic_EtherNet_IP_device

When no EDS file is provided for third-party device tag communication or the product ID of the current device does not match that in the EDS file, you can add this device and manually configure the device. On the "General setting" page, set all items of electronic matching to 0 and select "Stop check", as shown in the following figure.



Address Settings

IP Address: 10.45.124.77

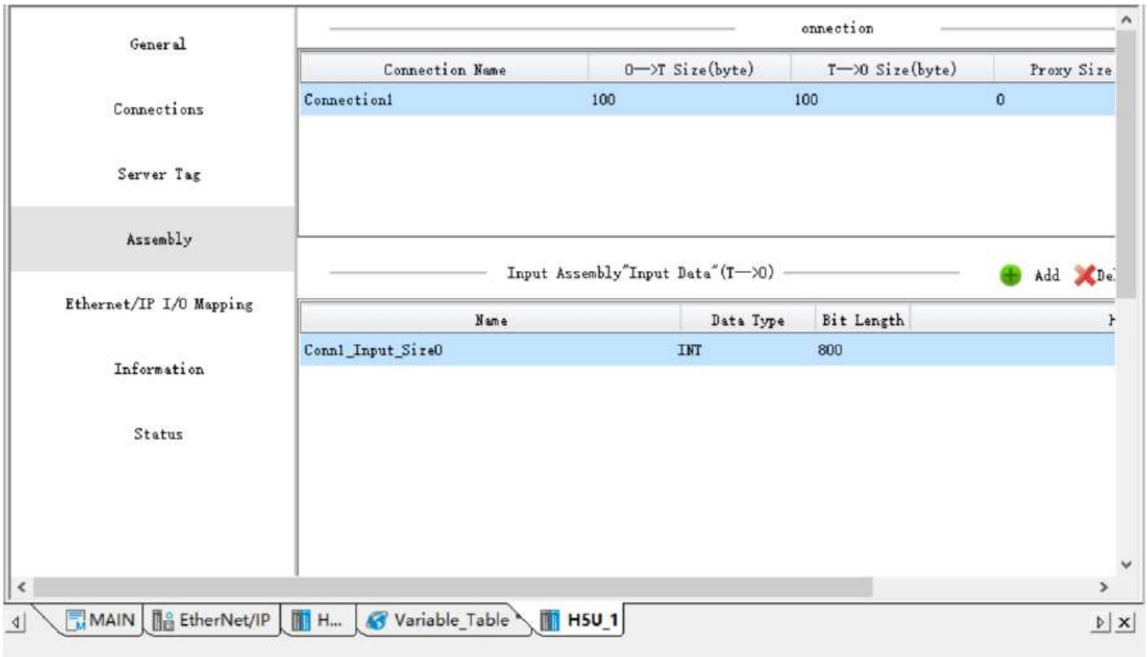
Electronic Keying

Keying Options: ○ Compatible Module ● Exact Match ○ Disable

Vendor ID: 1660
Device Type: 14
Product Code: 260
Major Revision: 1
Minor Revision: 1

10.3.5.3 Setting Tag Data Set

In the data area of tag communication, the parameter data types can be classified based on the T->O size. You can combine structure variable members to generate the corresponding I/O mapping.



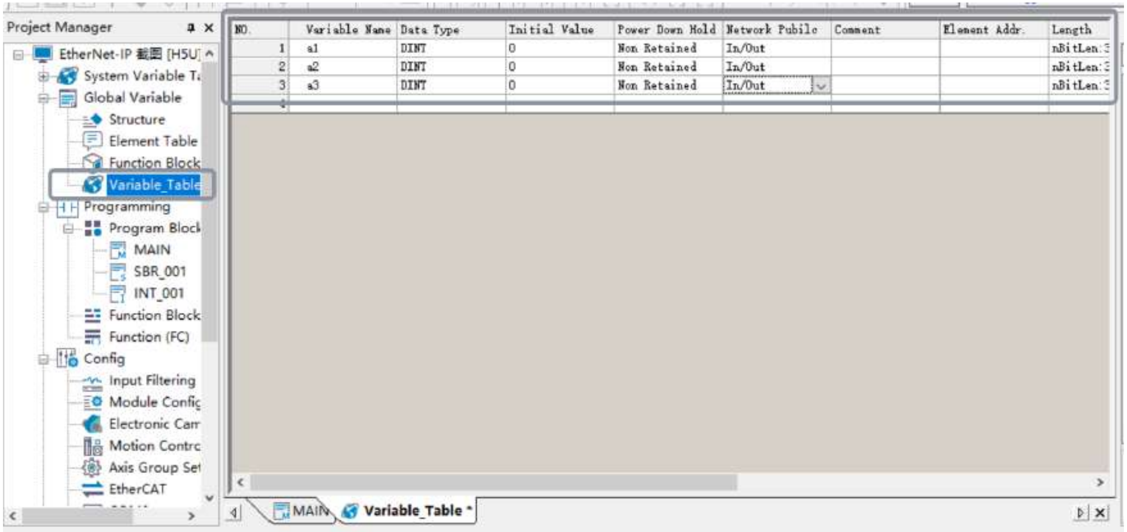
Connection Name	O—>T Size(byte)	T—>O Size(byte)	Proxy Size
Connection1	100	100	0

Name	Data Type	Bit Length
Conn1_Input_Size0	INT	800

10.3.6 Tag Communication Example

The following takes the example of one H5U consuming tags a1, a2, and a3 created by another H5U to describe how to make the configuration.

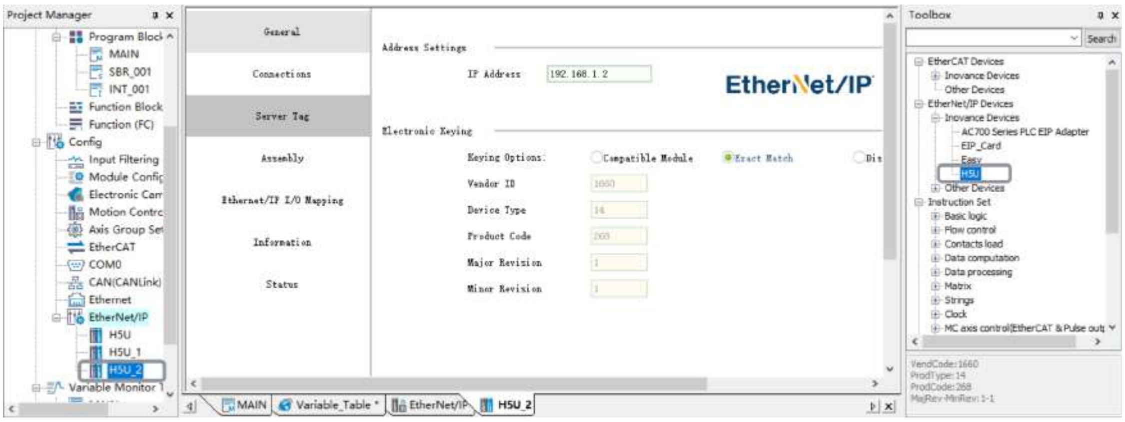
1. In the variable table, create the variable (producer). The configuration page is as follows.



NO.	Variable Name	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	a1	DINT	0	Non Retained	In/Out			nBitLen:3
2	a2	DINT	0	Non Retained	In/Out			nBitLen:3
3	a3	DINT	0	Non Retained	In/Out			nBitLen:3
4								

2. Associate the variable with the tag (producer).

3. Access the configuration page (consumer).



Address Settings

IP Address: 192.168.1.2

Electronic Keying

Keying Options: ○ Compatible Module ● Exact Match ○ Dis

Vendor ID: 1660

Device Type: 14

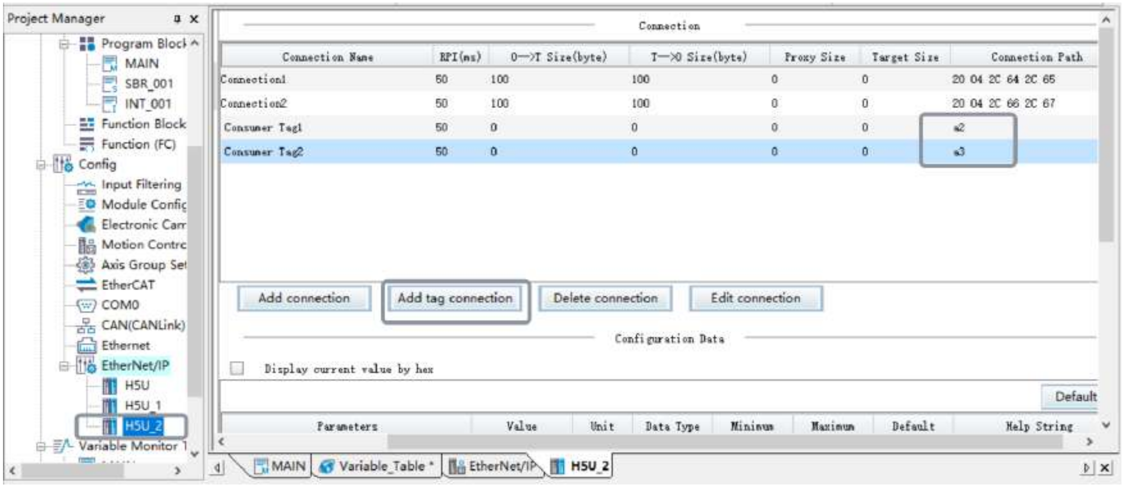
Product Code: 268

Major Revision: 1

Minor Revision: 1

VendCode:1660
ProdType:14
ProdCode:268
MajRev-MinRev:1-1

4. Add connections to consumer tags corresponding to the tags a1, a2, and a3 of H5U.



Connection

Connection Name	RPI(ms)	O—>T Size(byte)	T—>O Size(byte)	Proxy Size	Target Size	Connection Path
Connection1	50	100	100	0	0	20 04 2C 64 2C 65
Connection2	50	100	100	0	0	20 04 2C 66 2C 67
Consumer Tag1	50	0	0	0	0	a2
Consumer Tag2	50	0	0	0	0	a3

Configuration Data

□ Display current value by hex

Parameters	Value	Unit	Data Type	Minimum	Maximum	Default	Help String

After configuration, download the projects to the PLC to monitor their communication states.

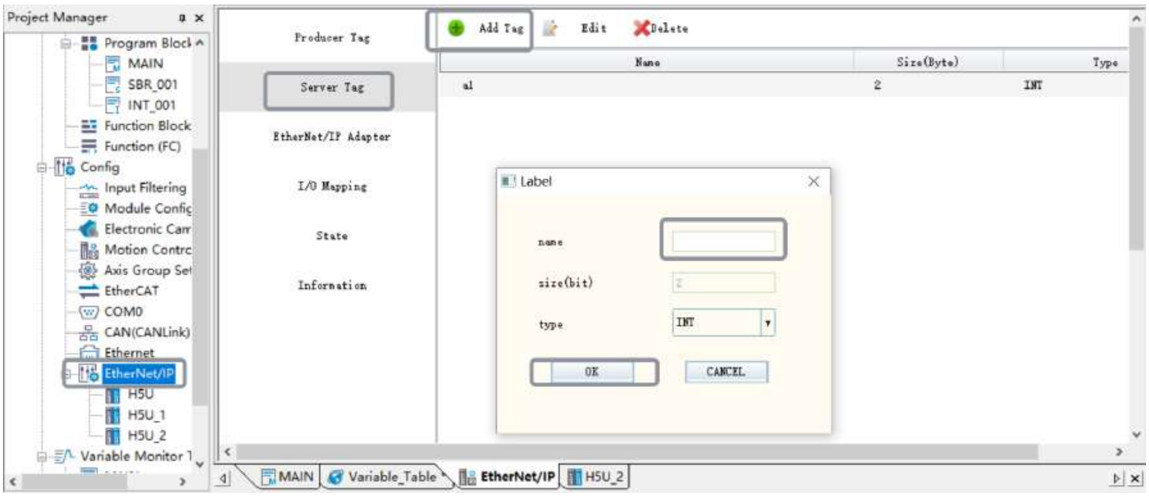
10.4 Service Message Tag Communication

10.4.1 Configuring Service Message Tags on Server

Configuration on the "Service message label" page

1. In the "Project Manager" section, unfold "Config", and double-click "EtherNet/IP" to open the "Service message label" page. Data types supported by service message tags are INT, DINT, REAL, and array.
2. To create a service message tag, you can create a variable in the variable table. On the "Service message label" page, click "Add label", input the created variable, and then click "OK".

You can also click "Add label" and input a name to create a tag on the "Service message label" page. Then, AutoShop automatically displays the variable creation dialog box, as shown in the following figure. (*If you modify the variable of a created server tag in the variable table, an error is reported when you compile the variable directly. In this case, you can click "Edit" to refresh the data type of the variable associated with the tag.*)



Name	Size(Byte)	Type
a1	2	INT

Label

name

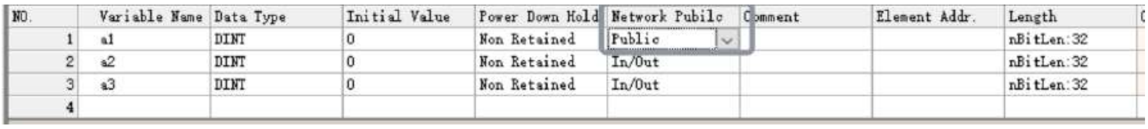
size(bit) 2

type INT

OK CANCEL

Directly configuring the tag in the variable table

In the variable table, modify the "Network Public" property of the variable to "Public". Then, the variable is configured as a service message tag.

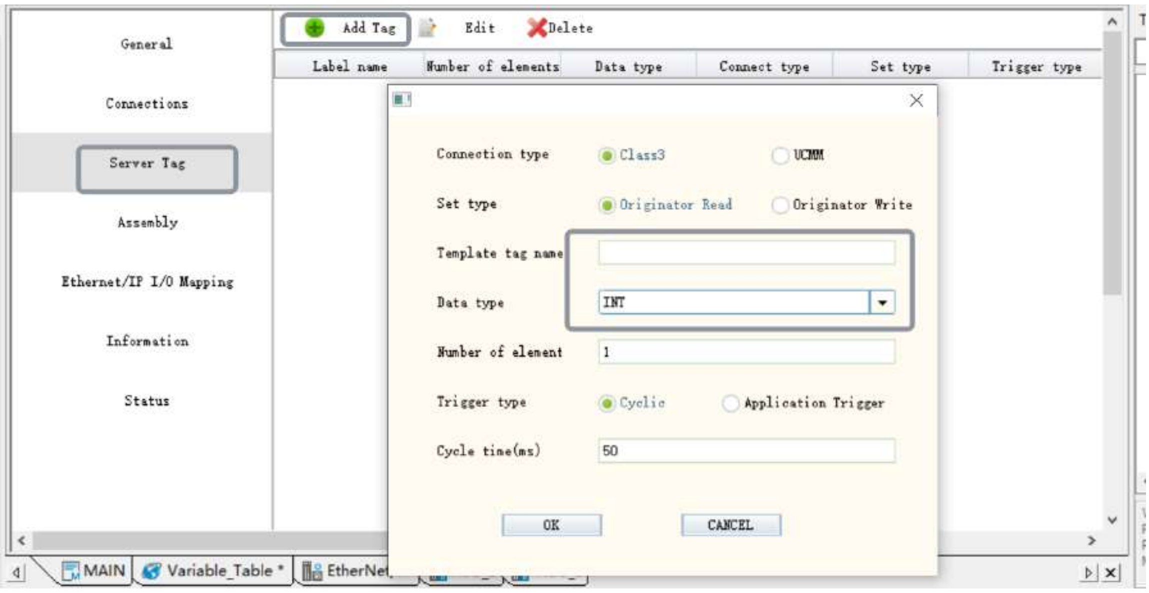


NO.	Variable Name	Data Type	Initial Value	Power Down Hold	Network Public	Comment	Element Addr.	Length
1	a1	DINT	0	Non Retained	Public			nBitLen:32
2	a2	DINT	0	Non Retained	In/Out			nBitLen:32
3	a3	DINT	0	Non Retained	In/Out			nBitLen:32
4								

10.4.2 Configuring Service Message Tags on Client

The following takes service message tag communication between Inovance H5Us as an example to describe how to configure service message tags.

1. In the "Toolbox" section, double-click the EDS file of H5U, add the configuration, modify the IP address, and access the "Service message label" page.
2. Click "Add label", and specify the name and data size of the service message tag you want to request, that is, the response service message tag created by the peer device to be connected.



Connection type ● Class3 ○ UCMM

Set type ● Originator Read ○ Originator Write

Template tag name

Data type INT

Number of element 1

Trigger type ● Cyclic ○ Application Trigger

Cycle time(ms) 50

OK CANCEL

You can click "Edit" to modify an added service message tag or double-click a tag in the list to edit the tag.

You can click "Delete" to delete an added service message tag.

The following section describes terms on the page of adding service message tag configuration.

- **Connection type**
  ① Class3 means a CIP connection is required for communication, that is, a connection-type tag communication.

  ② UCMM means a CIP connection is not required for communication, that is, connection-free tag communication.

- **Setting type**
  ① Originator Read means reading the data of the target tag.

  ② Originator Write means writing data to the target tag.

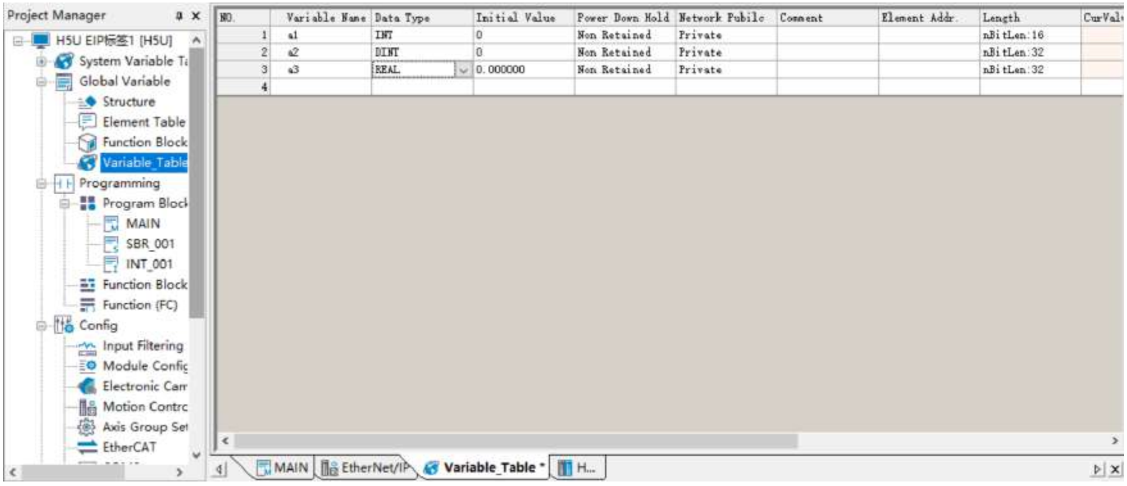
- **Target label name**: Indicates the name of the service message tag created by the requested device. The sum of the tag name size and the data size cannot exceed 487 bytes.
- **Data type**: Indicates the data type of the requested service message tag. Supported data types are INT, DINT, and REAL.
- **Element number**: Indicates the number of arrays of the requested service message tag. When the number of elements is N (N is greater than 1), an array whose size is N and type is basic data is created.
- **Trigger type**
  ① Cyclic means data is requested periodically. The valid range of "**Cycle time**" is 5 ms to 1000 ms, and the default value is 50 ms.

  ② Application Trigger indicates the status trigger and corresponds to **Input variable**. The request is triggered by the BOOL-type variables.

10.4.3 Application Example

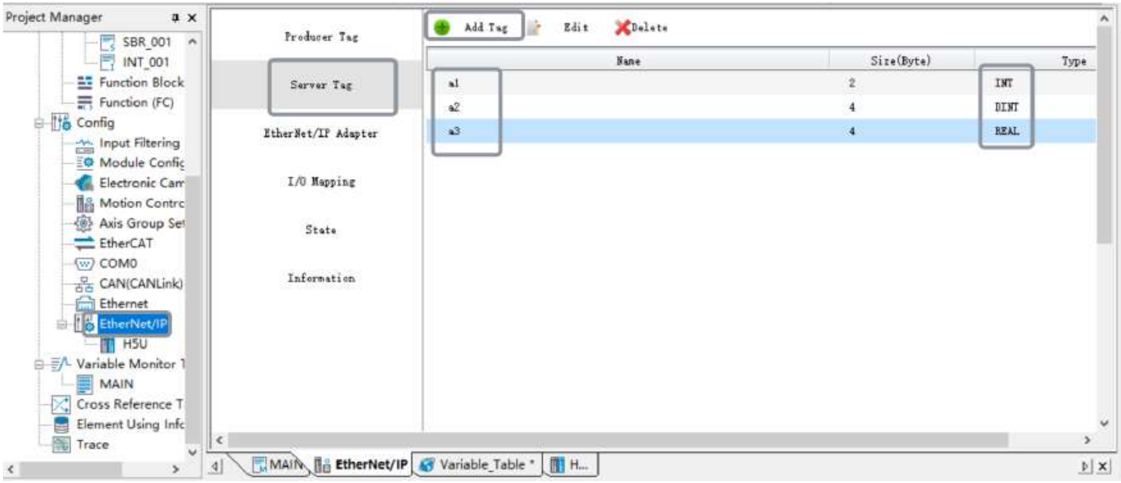
The following takes the example of H5U-1 consuming tags a1, a2, and a3 created by H5U-2.

1. Response service message tag: In the variable table, create the variable (producer). The H5U-2 configuration page is as follows.



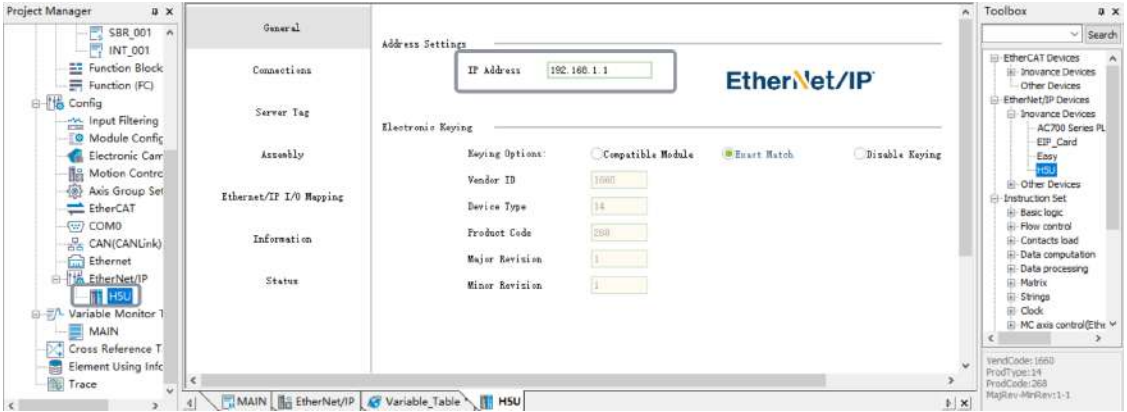
NO.	Variable Name	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length	CurVal
1	a1	INT	0	Non Retained	Private			nBitLen:16	
2	a2	DINT	0	Non Retained	Private			nBitLen:32	
3	a3	REAL	0.000000	Non Retained	Private			nBitLen:32	
4									

2. Response service message tag: In the H5U-2 project, associate the variable with the tag.



Name	Size(Byte)	Type
a1	2	INT
a2	4	DINT
a3	4	REAL

3. Request service message tag: In the H5U-1 project, configure the service message tag.



Address Settings

IP Address 192.168.1.1

Electronic Keying

Keying Options: ○ Compatible Module ● Exact Match ○ Disable Keying

Vendor ID 1660

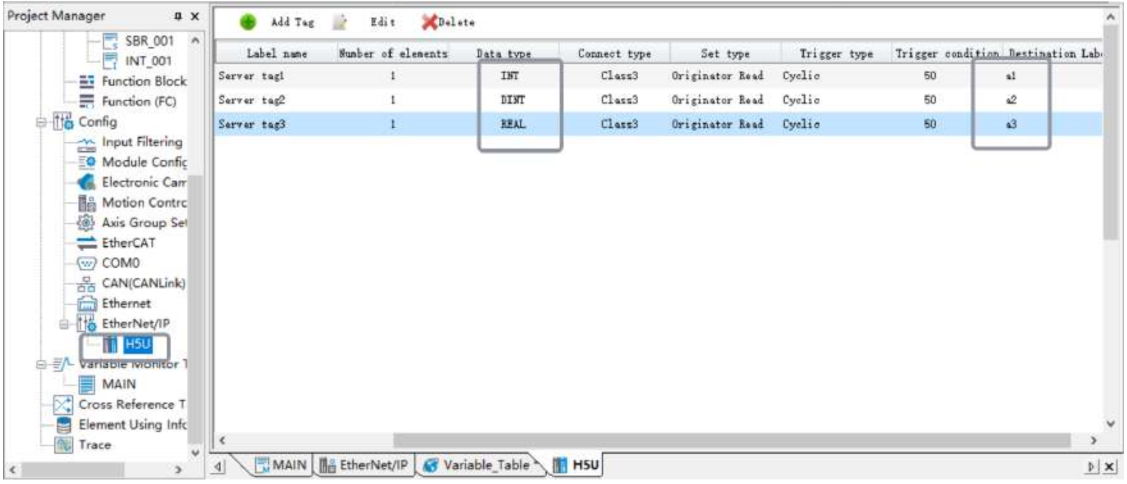
Device Type 14

Product Code 268

Major Revision 1

Minor Revision 1

VendCode:1660
ProdType:14
ProdCode:268
MajRev-MinRev:1-1



Label name	Number of elements	Data type	Connect type	Set type	Trigger type	Trigger condition	Destination Lab
Server tag1	1	INT	Class3	Originator Read	Cyclic	50	a1
Server tag2	1	DINT	Class3	Originator Read	Cyclic	50	a2
Server tag3	1	REAL	Class3	Originator Read	Cyclic	50	a3

4. After configuration, download the projects to the PLC to monitor their communication states, and monitor the request data in the H5U-1 project.

11 PROFINET Communication

11.1 Overview

About PROFINET

PROFINET, launched by PROFIBUS International (PI), is a new generation of automation bus standards based on the industrial Ethernet technology.

PROFINET provides a complete network solution for automation communication, encompassing hot topics in automation such as real-time Ethernet, motion control, distributed automation, fail-safe and network security. As a cross-supplier technology, it is fully compatible with industrial Ethernet and existing fieldbus (such as PROFIBUS) technologies, protecting existing investments.

Extension modules supported by H5U as PROFINET slave station

H5U supports connection to Siemens PLCs through the PROFINET protocol. Extension modules of H5U are operated through the TIA Portal software and data is mapped to H5U. The following table lists supported extension table.

Module Name	Input (Bytes)	Output (Bytes)	Description
0016ETN	-	2	A module with 16 DO terminals (transistor NPN output)
0016ETP	-	2	A module with 16 DO terminals (transistor PNP output)
0016ER	-	2	A module with 16 DO terminals (relay output)
1600END	2	-	A module with 16 DI terminals
3200END	4	-	An input module with 32 channels
0032ETN	-	4	An input module with 32 channels
4DA	-	8	A module with 4 DA terminals, supporting voltage and current output
4AD	8	-	A module with 4 AD terminals, supporting voltage and current input
8TC	16	-	A temperature module with 8 channels
4TC	8	-	A temperature module with 4 channels
4PT	8	-	A temperature module with 4 channels
Share IN2 BOOL	2	-	Remote module input data area
Share IN32 DINT	32	-	Remote module input data area
Share IN32 INT	32	-	Remote module input data area
Share IN32 REAL	32	-	Remote module input data area
Share IN64 DINT	64	-	Remote module input data area
Share IN64 INT	64	-	Remote module input data area
Share IN64 REAL	64	-	Remote module input data area
Share OUT2 BOOL	-	2	Remote module output data area
Share OUT32 DINT	-	32	Remote module output data area
Share OUT32 INT	-	32	Remote module output data area
Share OUT32 REAL	-	32	Remote module output data area

Module Name	Input (Bytes)	Output (Bytes)	Description
Share OUT64 DINT	-	64	Remote module output data area
Share OUT64 INT	-	64	Remote module output data area
Share OUT64 REAL	-	64	Remote module output data area

⚠ Caution

H5U devices can only be used as PROFINET slaves, and only support the reading and writing of process data of extension modules.

Data mapping relationship

- TIA Portal page
  The directory tree "Module" on the right shows all currently supported extension modules, as well as shared modules Share INXX and Share OUTXX. All modules can be added to the slots under H5U in the "Device overview" area in any order. A maximum of 16 modules can be added to the 16 slots (except Share INXX/Share OUTXX, a maximum of 15 Share IN64XX/Share OUT64XX modules or a maximum of 20 other modules can be added). In the TIA Portal software, you can operate the I or Q address of each module, map the final data to H5U, and view the data of the module in AutoShop.
- AutoShop page
  AutoShop uses the system variable "_SYS_PN" to obtain mapped data in the TIA Portal software.

  In the TIA Portal software, data areas of all H5U modules are mapped to corresponding variables to the variable table "_SYS_PN" in AutoShop. The following table lists the mapping relationship.

  - Mapping relationship of H5U extension modules

TIA Portal Module Name	AutoShop Variable Name
0016ETN	MOD1600ETN
0016ETP	MOD3200ETP
0016ER	MOD0016ER
1600END	MOD1600END
3200END	MOD3200END
0032ETN	MOD0032ETN
4DA	MOD4DA
4AD	MOD4AD
8TC	MOD8TC
4TC	MOD4TC
4PT	MOD4PT

When there are multiple identical modules, these modules will be mapped to the data areas of the corresponding indexes in the order they are added in the TIA Portal software. For example, three 3200END modules are added in the Portal software, and they are mapped to "MOD3200 [0]", "MOD3200[1]", and "MOD3200[2]", respectively.

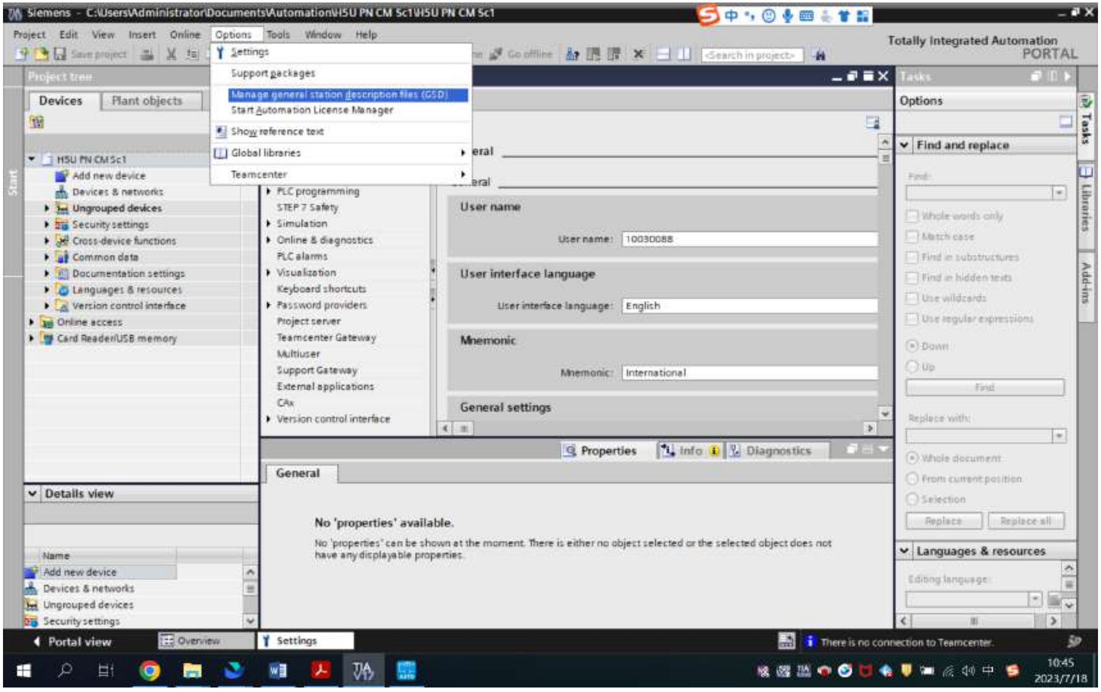
  - "Share IN*XX*" and "Share OUT*XX*" modules are mapped to variables "_share_in*XX*" and "_share_out*XX*", respectively. These modules can be used as data mapping variables for remote modules (Share IN*XX* mapped to _share_in*XX*, and Share OUT*XX* mapped to _share_out*XX*).

11.2 Configuration Process

11.2.1 TIA Portal Configuration

1. Import and install the GSD file.

a. In the menu bar, choose "Option > Manage general station description file (GSD)", as shown in the following figure.



User name

User name: 10030088

User interface language

User interface language: English

Mnemonic

Mnemonic: International

General settings

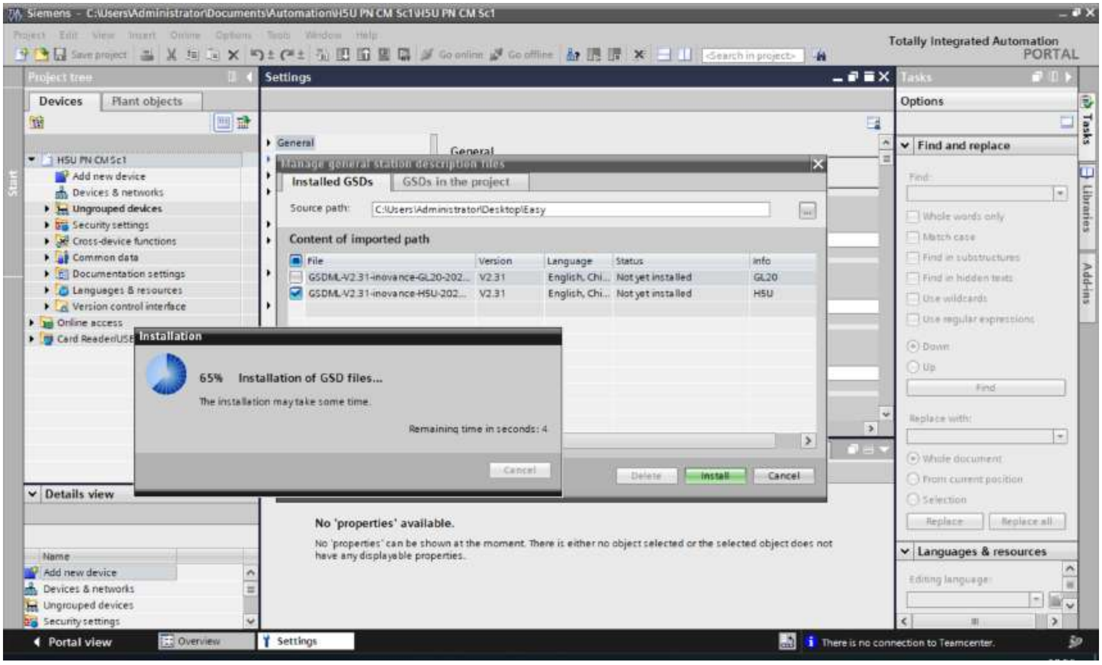
No 'properties' available.

No 'properties' can be shown at the moment. There is either no object selected or the selected object does not have any displayable properties.

There is no connection to Teamcenter.

10:45
2023/7/18

b. In the dialog box displayed, click [...], locate the GSD file (the software automatically indexes the file), select the GSD file, and then click "Install".



Source path: C:\Users\Administrator\Desktop\Easy

Content of imported path

☐ File	Version	Language	Status	Info
☐ GSDML-V2.31-inovance-GL20-202...	V2.31	English, Chi...	Not yet installed	GL20
☑ GSDML-V2.31-inovance-H5U-202...	V2.31	English, Chi...	Not yet installed	H5U

65% Installation of GSD files...

The installation may take some time.

Remaining time in seconds: 4

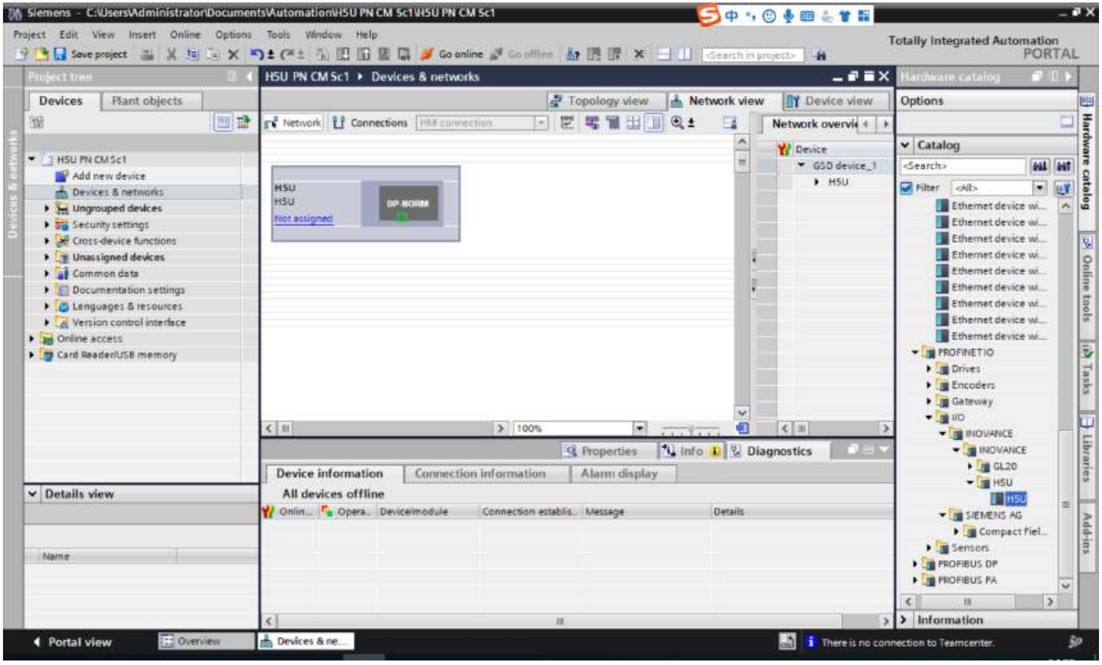
No 'properties' available.

No 'properties' can be shown at the moment. There is either no object selected or the selected object does not have any displayable properties.

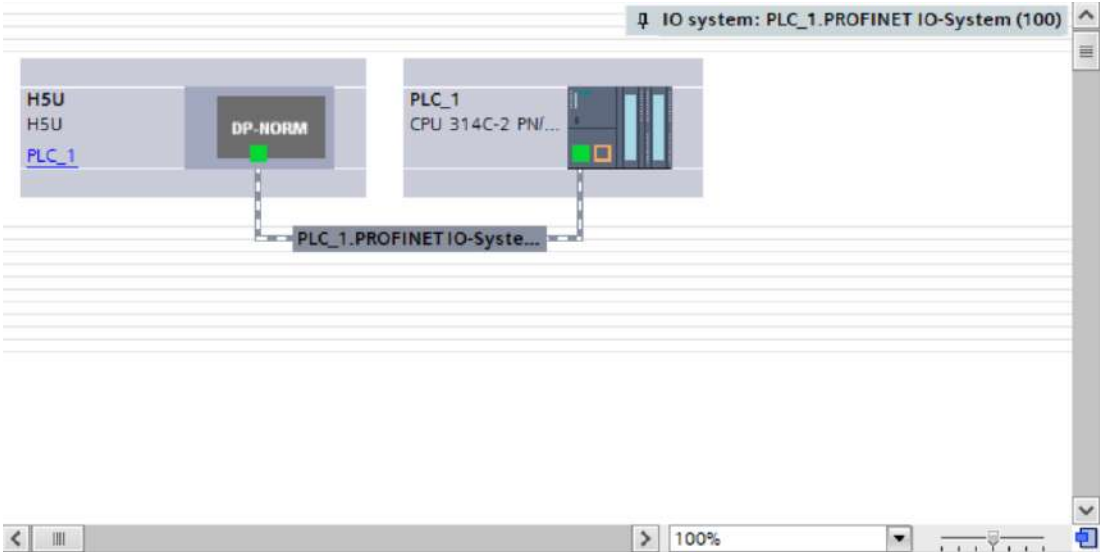
There is no connection to Teamcenter.

2. Configure H5U.

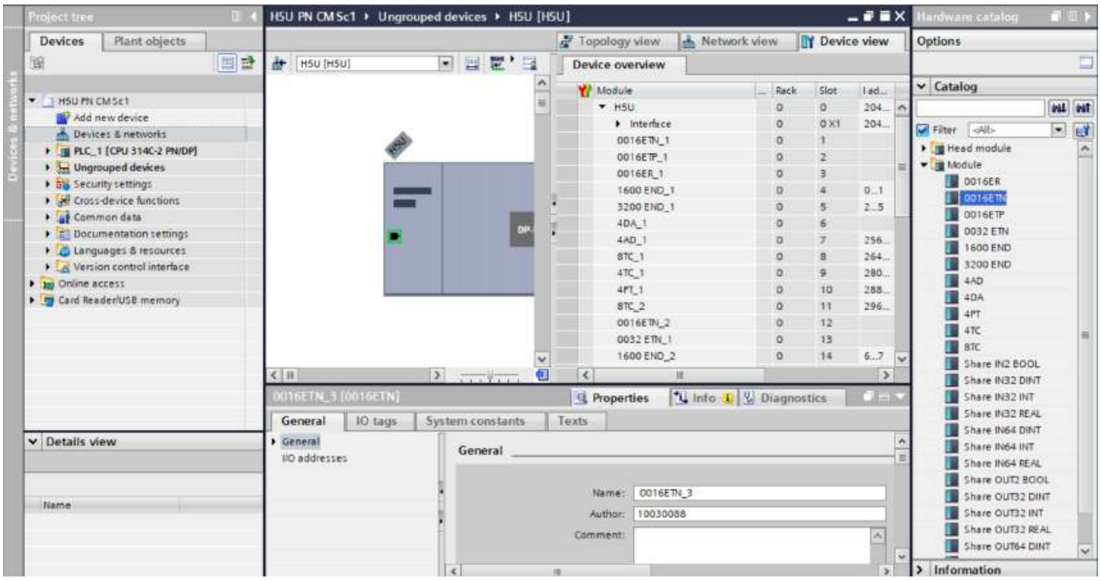
a. Double-click "H5U" in the hardware directory or drag "H5U" to the "Network view" section, as shown in the following figure.



b. Hold down the left mouse button and drag the PLC green box to the H5U green box to connect the PLC and H5U, as shown in the following figure.


IO system: PLC_1.PROFINET IO-System (100)
H5U
H5U
PLC_1
DP-NORM
PLC_1
CPU 314C-2 PN/...
PLC_1.PROFINET IO-Syste...
100%

c. Double-click a module in the hardware directory tree on the right to add the module and assign the address, as shown in the following figure.



H5U PN CM Sc1 ▸ Ungrouped devices ▸ H5U [H5U]

Device overview

Module	...	Rack	Slot	I ad...
H5U		0	0	204...
Interface		0	0 X1	204...
0016ETN_1		0	1	
0016ETP_1		0	2	
0016ER_1		0	3	
1600 END_1		0	4	0...1
3200 END_1		0	5	2...5
4DA_1		0	6	
4AD_1		0	7	256...
8TC_1		0	8	264...
4TC_1		0	9	280...
4PT_1		0	10	288...
8TC_2		0	11	296...
0016ETN_2		0	12	
0032 ETN_1		0	13	
1600 END_2		0	14	6...7

0016ETN_3 [0016ETN]

General

Name: 0016ETN_3

Author: 10030088

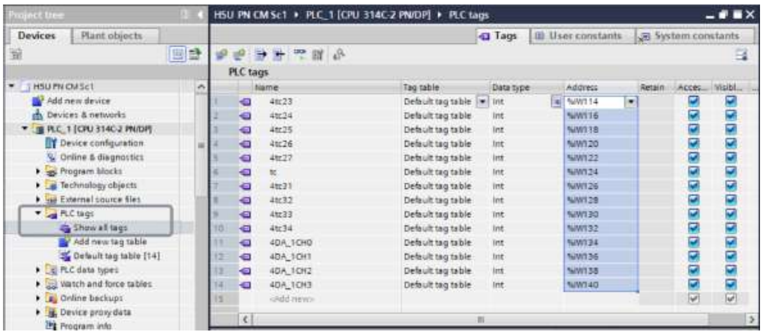
Comment:

Note

It is recommended to keep the order of added modules and slots consistent with the AutoShop interface.

---

d. Click "Display all variables". On the "PLC variable" page displayed, define the specific address variable of the module, as shown in the following figure.



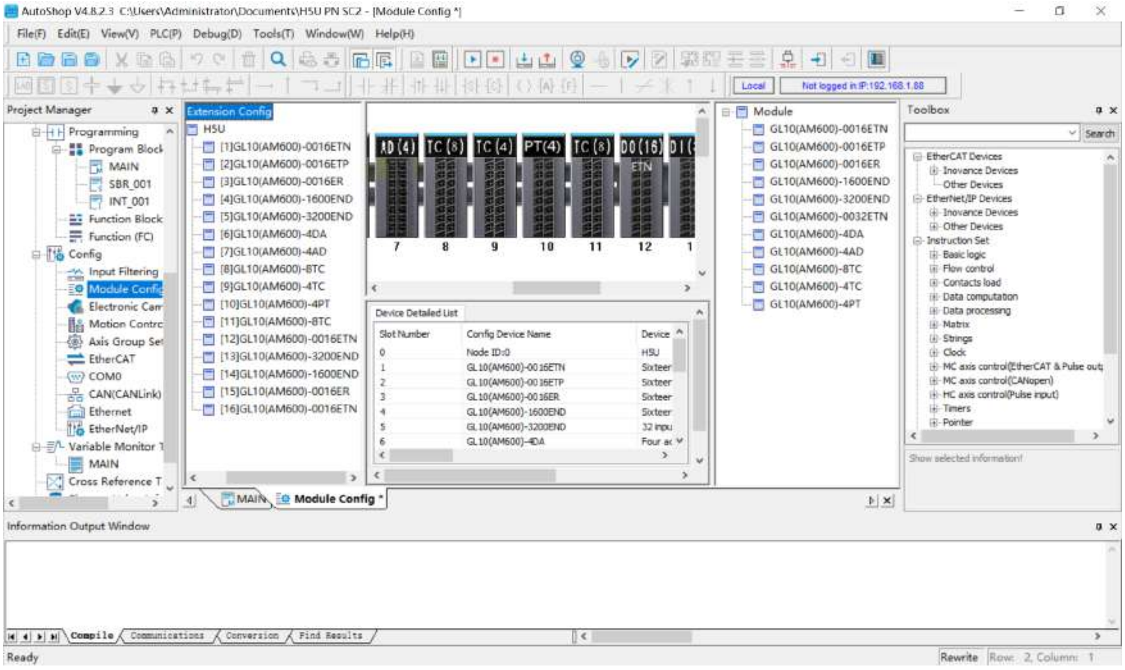
H5U PN CM Sc1 ▸ PLC_1 [CPU 314C-2 PN/DP] ▸ PLC tags

PLC tags

	Name	Tag table	Data type	Address	Retain	Acces...	Visibl...
1	4tc23	Default tag table	Int	%IW114		☑	☑
2	4tc24	Default tag table	Int	%IW116		☑	☑
3	4tc25	Default tag table	Int	%IW118		☑	☑
4	4tc26	Default tag table	Int	%IW120		☑	☑
5	4tc27	Default tag table	Int	%IW122		☑	☑
6	tc	Default tag table	Int	%IW124		☑	☑
7	4tc31	Default tag table	Int	%IW126		☑	☑
8	4tc32	Default tag table	Int	%IW128		☑	☑
9	4tc33	Default tag table	Int	%IW130		☑	☑
10	4tc34	Default tag table	Int	%IW132		☑	☑
11	4DA_1CH0	Default tag table	Int	%IW134		☑	☑
12	4DA_1CH1	Default tag table	Int	%IW136		☑	☑
13	4DA_1CH2	Default tag table	Int	%IW138		☑	☑
14	4DA_1CH3	Default tag table	Int	%IW140		☑	☑
15	<Add new>					☑	☑

11.2.2 AutoShop Configuration

1. Under "Config" in the "Project Manager" section, double-click "Module Config". In the "Module" section, double-click the module you want to add, as shown in the following figure.



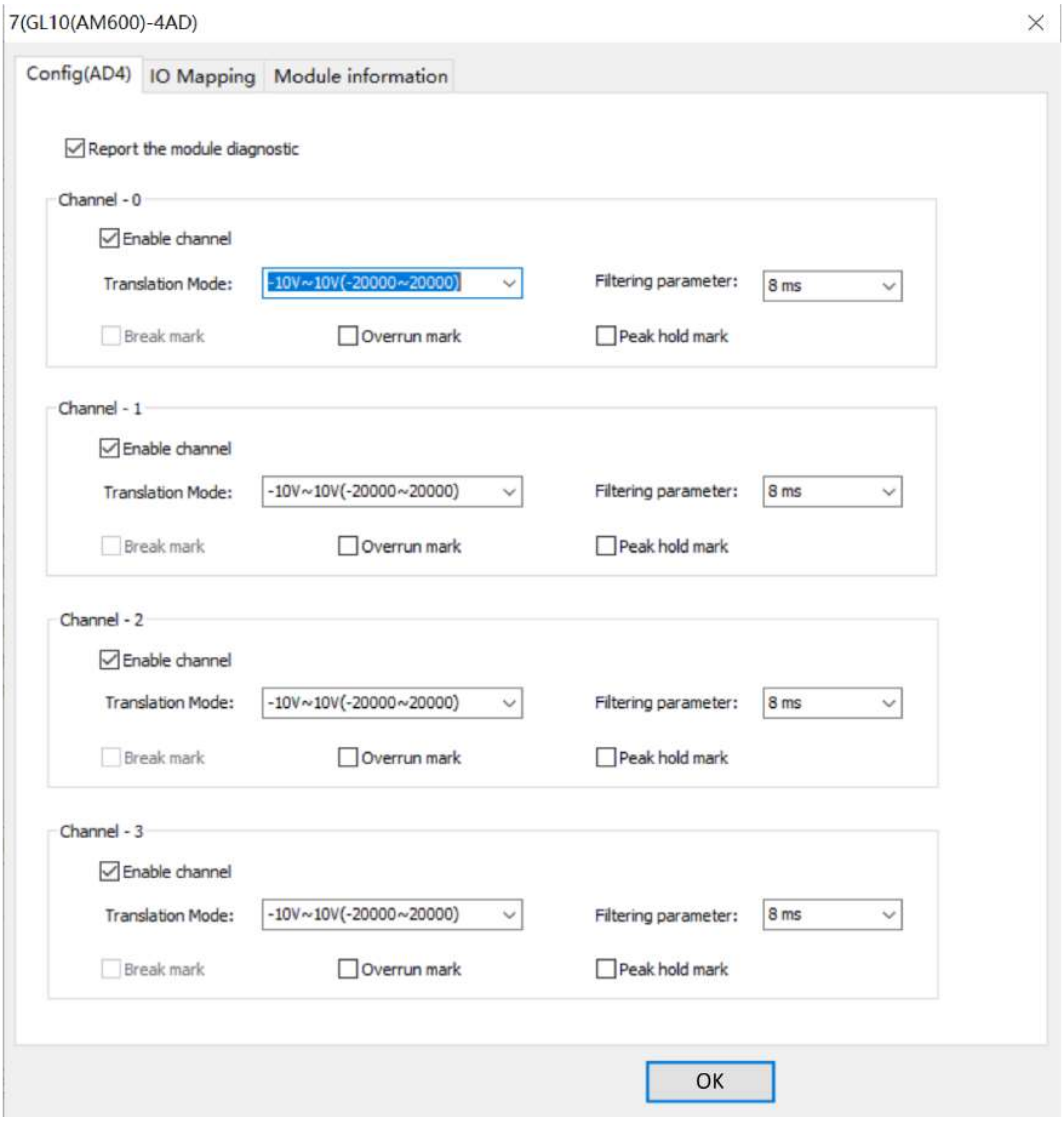
Device Detailed List

Slot Number	Config Device Name	Device
0	Node ID:0	H5U
1	GL10(AM600)-0016ETN	Sixteer
2	GL10(AM600)-0016ETP	Sixteer
3	GL10(AM600)-0016ER	Sixteer
4	GL10(AM600)-1600END	Sixteer
5	GL10(AM600)-3200END	32 inpu
6	GL10(AM600)-4DA	Four ac

Note

It is recommended to keep the order of added modules and slot numbers consistent with the TIA Portal interface.

2. In the "Device Detailed List" section, double-click the module (taking GL10-4AD as an example) for which you want to configure parameters, configure the module parameters, and then click "OK", as shown in the following figure.



7(GL10(AM600)-4AD)

Config(AD4) | IO Mapping | Module information

☑ Report the module diagnostic

Channel - 0

☑ Enable channel

Translation Mode: -10V~10V(-20000~20000) Filtering parameter: 8 ms

□ Break mark □ Overrun mark □ Peak hold mark

Channel - 1

☑ Enable channel

Translation Mode: -10V~10V(-20000~20000) Filtering parameter: 8 ms

□ Break mark □ Overrun mark □ Peak hold mark

Channel - 2

☑ Enable channel

Translation Mode: -10V~10V(-20000~20000) Filtering parameter: 8 ms

□ Break mark □ Overrun mark □ Peak hold mark

Channel - 3

☑ Enable channel

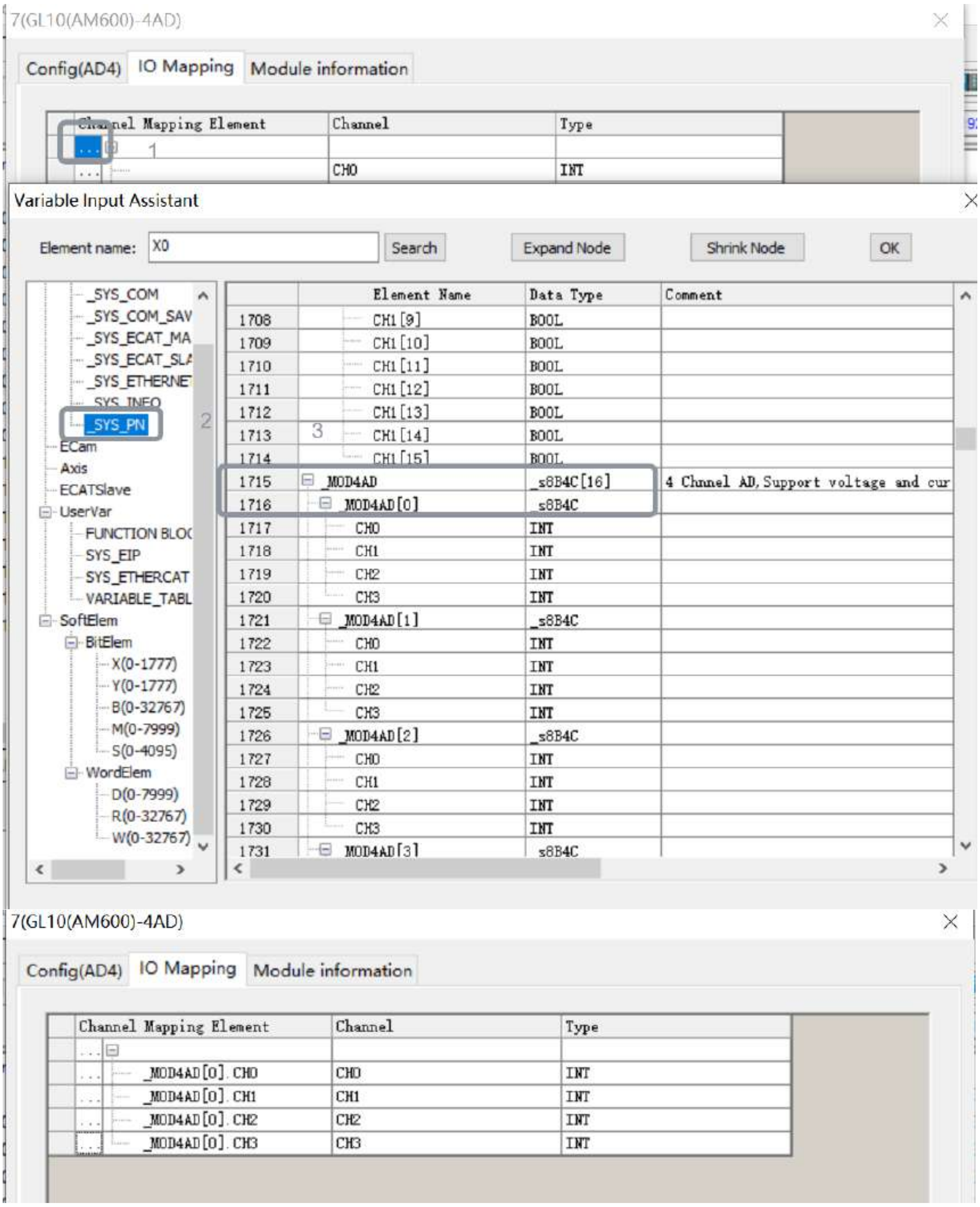
Translation Mode: -10V~10V(-20000~20000) Filtering parameter: 8 ms

□ Break mark □ Overrun mark □ Peak hold mark

3. Map the module I/O to the corresponding variable in the system variable "_SYS_PN".

   a. In the "Device Detailed List" section, double-click the module (taking GL10-4AD as an example) for which you want to perform I/O mapping, as shown in the preceding figure.

   b. Click "IO Mapping", and map the module to the variable "MOD4AD[0]", as shown in the following figure.



Channel Mapping Element	Channel	Type
	CH0	INT

Element name: X0

	Element Name	Data Type	Comment
1708	CH1[9]	BOOL	
1709	CH1[10]	BOOL	
1710	CH1[11]	BOOL	
1711	CH1[12]	BOOL	
1712	CH1[13]	BOOL	
1713	CH1[14]	BOOL	
1714	CH1[15]	BOOL	
1715	_MOD4AD	_s8B4C[16]	4 Chnnel AD, Support voltage and cur
1716	_MOD4AD[0]	_s8B4C	
1717	CH0	INT	
1718	CH1	INT	
1719	CH2	INT	
1720	CH3	INT	
1721	_MOD4AD[1]	_s8B4C	
1722	CH0	INT	
1723	CH1	INT	
1724	CH2	INT	
1725	CH3	INT	
1726	_MOD4AD[2]	_s8B4C	
1727	CH0	INT	
1728	CH1	INT	
1729	CH2	INT	
1730	CH3	INT	
1731	MOD4AD[3]	s8B4C	

Channel Mapping Element	Channel	Type
_MOD4AD[0].CH0	CH0	INT
_MOD4AD[0].CH1	CH1	INT
_MOD4AD[0].CH2	CH2	INT
_MOD4AD[0].CH3	CH3	INT

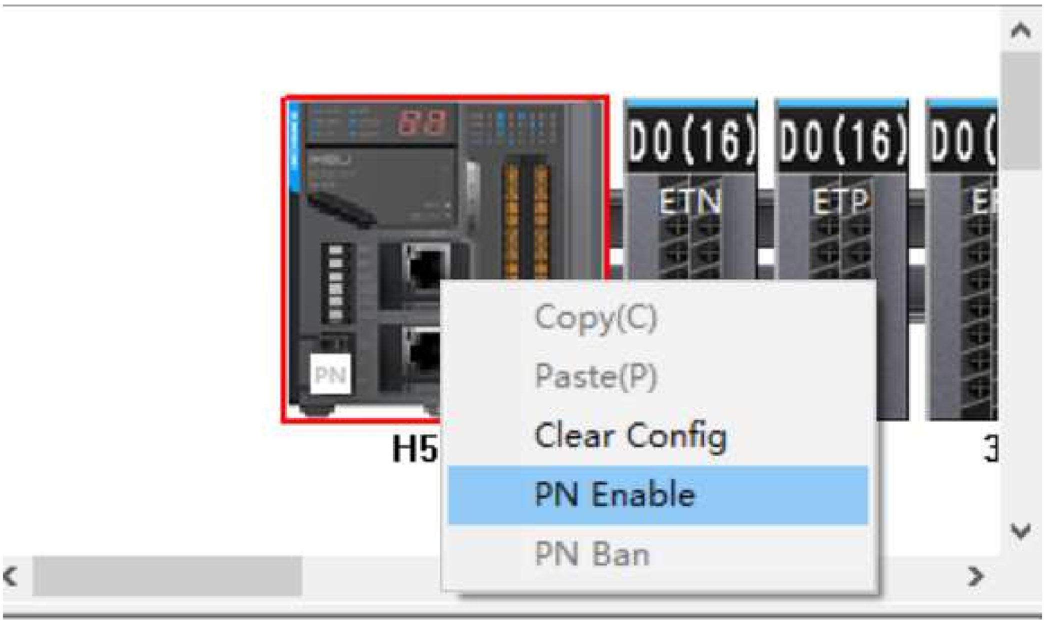
c. Click "OK", and then "OK".

11.3 Enable and Disable

PROFINET is disabled for H5U by default. To enable this function, perform the following operations.

Enable

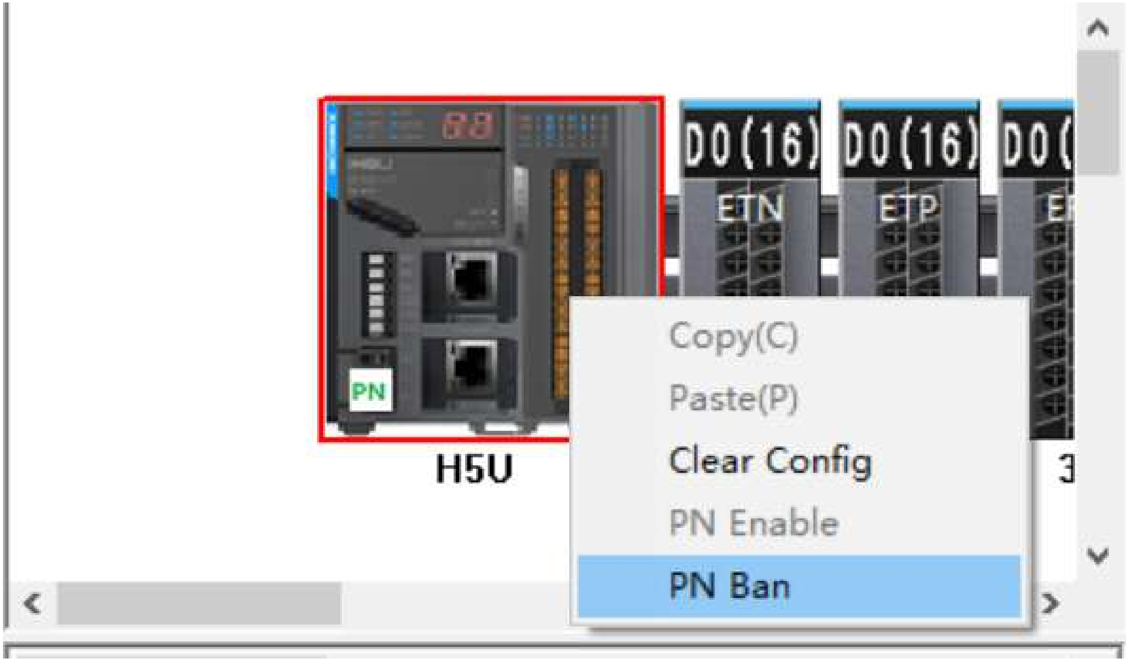
1. On the "Module Config" page, right-click "H5U", and select "PN Enable".



2. Download the program to H5U.
3. Restart H5U. Then, the function is enabled.

Disable

1. On the "Module Config" page, right-click "H5U", and select "PN Ban".



2. Download the program to H5U.
3. Restart H5U. Then, the function is disabled.

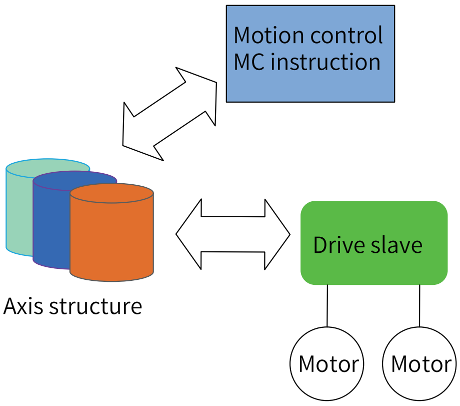
12 Motion Control

12.1 Introduction to Motion Control Axes

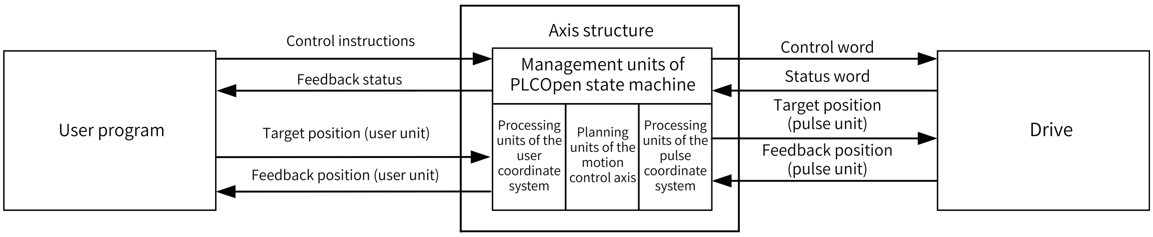
12.1.1 Overview

Basic composition and control logic

In a motion control system, the objects of motion control are called axes. The axes are the bridge between a drive and the PLC instructions. The motion control axes are used to control EtherCAT bus drive that is compliant with the CiA 402 protocol, as well as local high-speed pulse outputs and high-speed pulse inputs.



The following figure shows the basic composition and processing logic of an axis in the PLC.


User program
Control instructions
Feedback status
Target position (user unit)
Feedback position (user unit)
Axis structure
Management units of PLCOpen state machine
Processing units of the user coordinate system
Planning units of the motion control axis
Processing units of the pulse coordinate system
Control word
Status word
Target position (pulse unit)
Feedback position (pulse unit)
Drive

Scheduling mechanism of motion control instructions

Main programs, subprograms, and interrupt subprograms are provided for users to write programs. However, motion control instructions can only be called in the main programs or subprograms, not in the interrupt subprograms.

The EtherCAT tasks are hidden tasks that are not open to users, so programming of the EtherCAT tasks is not supported.

As shown in the following figure, in a main program, the PLC scans all the motion control instructions written in the program in turn, and stores the final result in the motion control parameter buffer according to the interrupt rules of the program. The PLC updates the motion control instruction when a 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 program updates the instruction status 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, the target position of the second instruction is 200, and both instructions are triggered by the soft element M1000 at the same time. When the PLC scans the program, the PLC first scans the first absolute positioning instruction, obtaining the target position 100, and then scans the second instruction, updating the target position to 200. At the end of the main task, the PLC finally writes the target position 200 to the motion control buffer and implements the instruction according to the second absolute positioning parameter, interrupting the first instruction. After obtaining the target position 200, the EtherCAT task starts to execute the absolute positioning algorithm, and sets the completion flag when the positioning is completed. After the second absolute positioning instruction in the main program obtains the completion flag, the Done signal is activated.

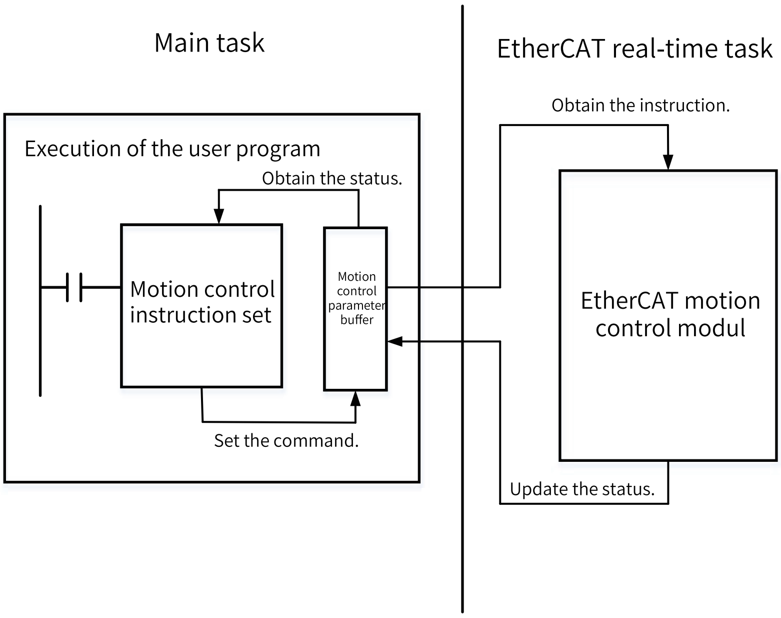

Main task
EtherCAT real-time task
Obtain the instruction.
Execution of the user program
Obtain the status.
Motion control instruction set
Motion control parameter buffer
EtherCAT motion control modul
Set the command.
Update the status.

Figure 12-1 EtherCAT instruction execution

Axis types

Supported axis types include bus servo axis, local pulse axis, bus encoder axis, and local encoder axis.

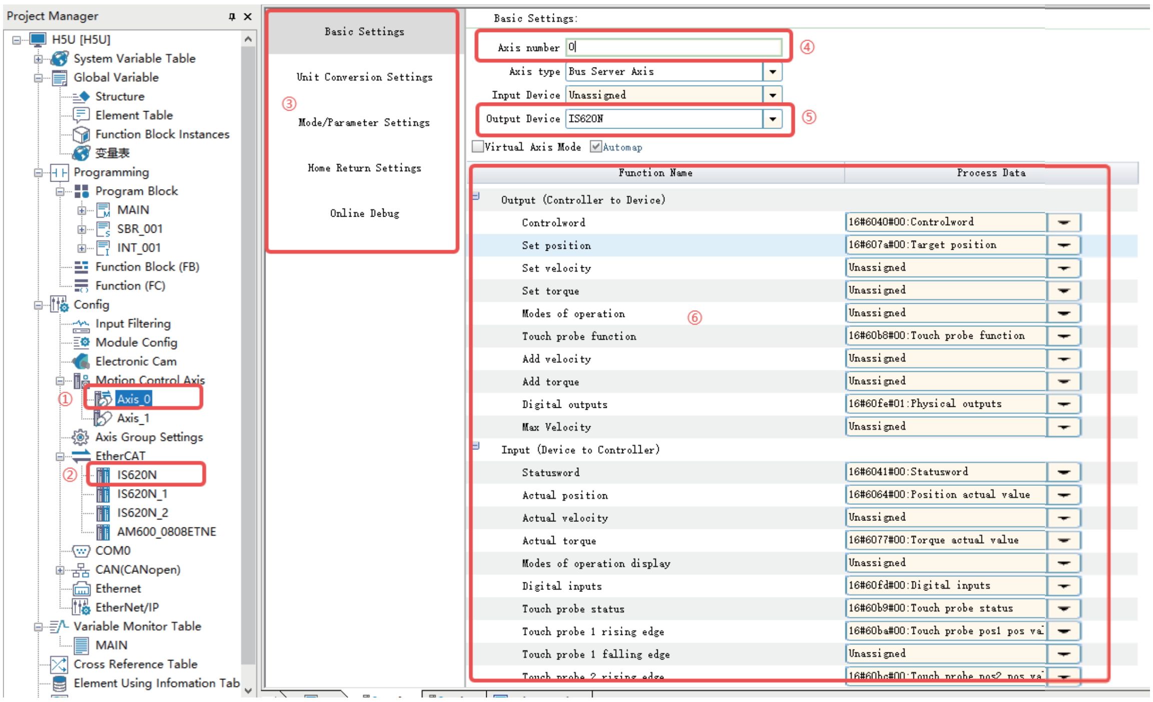
Axis Type	Content
Bus servo axis	Bus servo axis is controlled using EtherCAT slave servo drives. When virtual axis mode is disabled, the bus servo axis is assigned to the actual servo drive for use. The bus servo axis supports the control of several basic modes such as torque, point, velocity, and homing modes.
Local pulse axis	Local pulse axis is controlled using a pulse drive controlled by local high-speed I/O. Four local pulse axes can be set: Y0/Y1, Y2/Y3, Y4/Y5, and Y6/Y7. Each pulse output channel can be set to pulse+direction or CW/CCW. Up to two probe terminals can be set per pulse output channel. The local pulse axis supports control in several basic modes such as point, velocity, and homing modes. Torque mode is not supported.
Bus encoder axis	Reserved
Local encoder axis	See *“13.1 Introduction to High-speed Counter Axes” on page 427*.

To fully describe the attributes of an axis, monitor the axis status, and control the axis motion, each axis is divided into three parts.

Axis Structure	Function
Axis configuration parameter	Configures parameters of an axis, such as gear ratio, homing type, and encoder mode.
Axis system variable	Monitors the operating status and abnormal information of an axis, such as the current position and axis error code.
Axis control instruction	In a user program, axis motion control is performed using MC motion control instructions. Axis control instructions are divided into management (such as MC_Power), motion (such as MC_Jog), and status (MC_ReadStatus).

Configuration interface

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



Basic Settings:

Axis number: 0

Axis type: Bus Server Axis

Input Device: Unassigned

Output Device: IS620N

□ Virtual Axis Mode ☑ Automap

Function Name	Process Data
Output (Controller to Device)	
Controlword	16#6040#00:Controlword
Set position	16#607a#00:Target position
Set velocity	Unassigned
Set torque	Unassigned
Modes of operation	Unassigned
Touch probe function	16#60b8#00:Touch probe function
Add velocity	Unassigned
Add torque	Unassigned
Digital outputs	16#60fe#01:Physical outputs
Max Velocity	Unassigned
Input (Device to Controller)	
Statusword	16#6041#00:Statusword
Actual position	16#6064#00:Position actual value
Actual velocity	Unassigned
Actual torque	16#6077#00:Torque actual value
Modes of operation display	Unassigned
Digital inputs	16#60fd#00:Digital inputs
Touch probe status	16#60b9#00:Touch probe status
Touch probe 1 rising edge	16#60ba#00:Touch probe pos1 pos va
Touch probe 1 falling edge	Unassigned
Touch probe 2 rising edge	16#60bc#00:Touch probe pos2 pos va

① Motion control axis

② EtherCAT bus drive

③ List of axis configuration and monitoring options

④ Axis number (unique access ID for an axis)

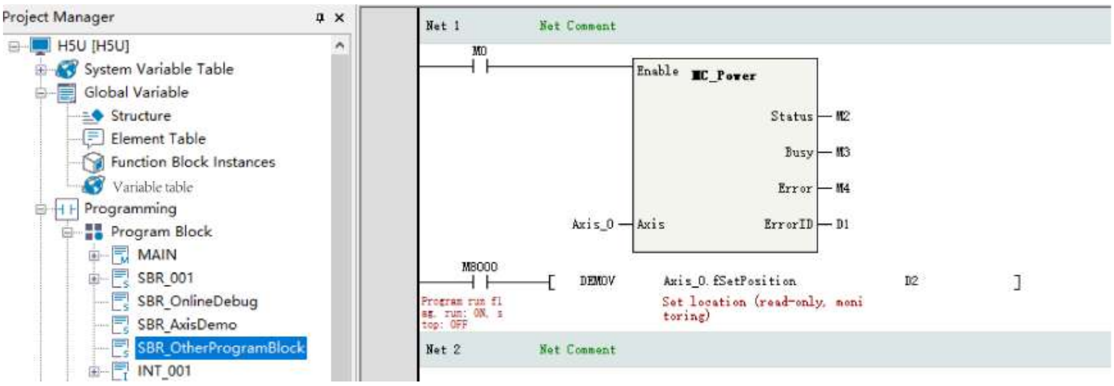
⑤ Associated physical drive

⑥ Detailed parameter settings

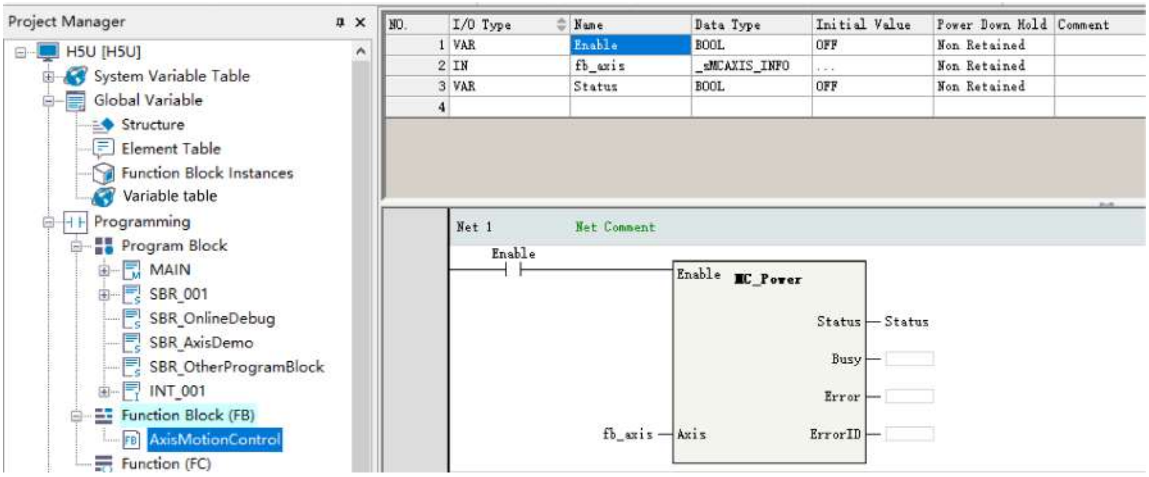
Motion control axis access modes

In the PLC program, a motion control axis can be accessed in two ways: motion control instructions and system variables.

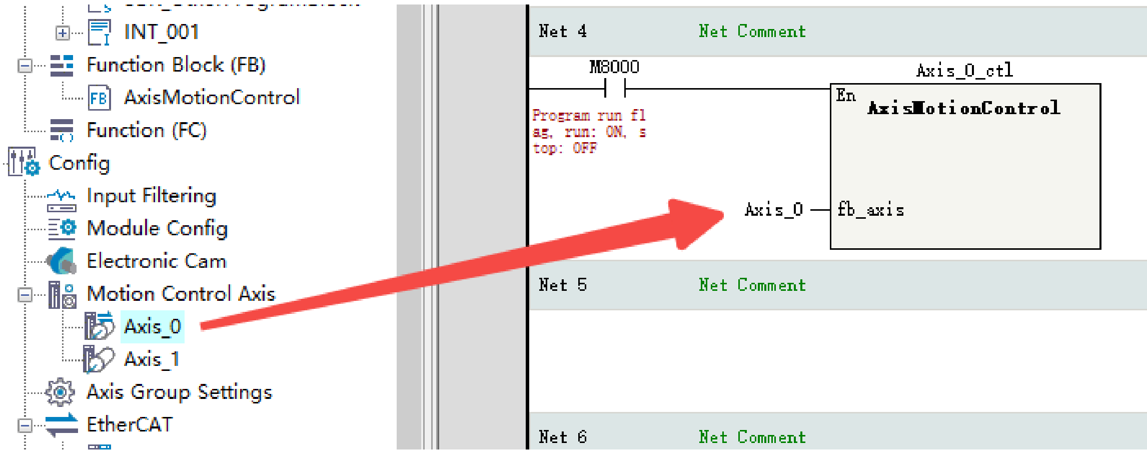
- In AutoShop of version V4.0.0.0 or later versions in combination with PCB software of version V3.0.0.0 or later versions, axis instructions and system variables can be accessed by using axis names. Axis names can also be introduced into FB as parameters, for example:



Axes can also be introduced into FB as parameters.



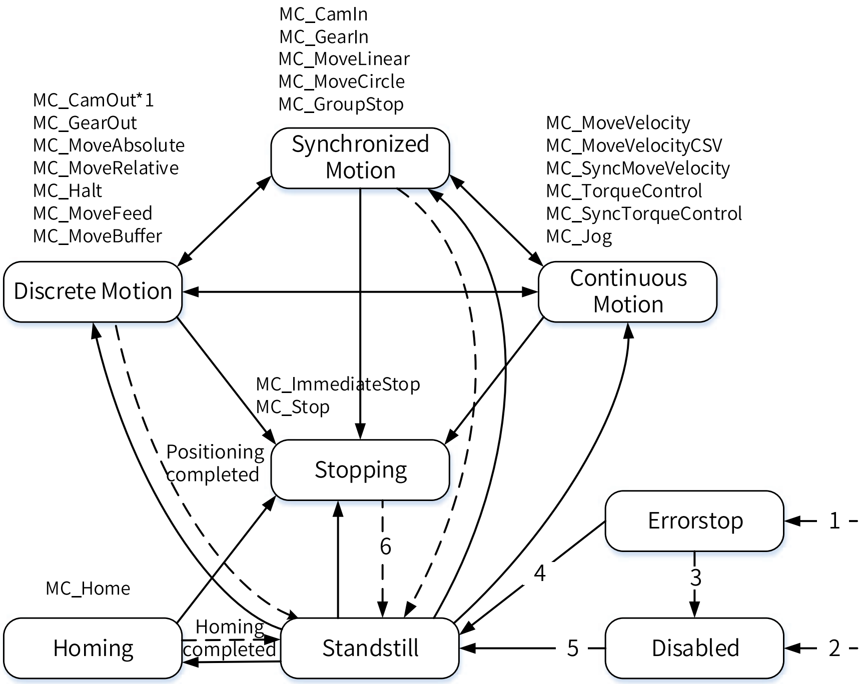
NO.	I/O Type	Name	Data Type	Initial Value	Power Down Hold	Comment
1	VAR	Enable	BOOL	OFF	Non Retained	
2	IN	fb_axis	_sMCAXIS_INFO	...	Non Retained	
3	VAR	Status	BOOL	OFF	Non Retained	
4						



12.1.2 PLCOpen State Machine

The PLCOpen state machine allows you to manage axis status and motion and complete different functions in different states.

The status transition diagram is as follows.



The detailed description is as follows.

Status Value	Status	Description
0	Disabled	Disabled
1	ErrorStop	Stopped due to a fault
2	Stopping	Stopping
3	Standstill	Enabled
4	Discrete Motion	Discrete motion
5	Continuous Motion	Continuous motion
7	Homing	Homing
8	Synchronized Motion	Synchronized motion

The following table summarizes the status transition conditions.

Transition	Transition Conditions
1	The fault detection logic of the axis detects a fault. In this case, the system immediately transits to this state.
2	The axis is free of faults and MC_Power.Enable=FALSE
3	MC_Reset is called to reset the axis fault and MC_Power.Status=FASLE.
4	MC_Reset is called to reset the axis fault and MC_Power.Status=TRUE.
5	MC_Power.Enable=TRUE and MC_Power.Status=TRUE.
6	MC_Stop(MC_ImmediateStop).Done=TRUE and MC_Stop(MC_ImmediateStop).Execute=FALSE.

12.1.3 Axis Units

Two units are used in the axis structure: user unit and pulse unit.

User unit

It refers to the measurement units used in instructions, such as millimeters, centimeters, and angles, which are called user units and usually represented by Unit.

The user coordinate systems are divided into linear coordinate system and rotary coordinate system according to working conditions.

A linear coordinate system typically includes a zero point. An axis is in forward motion when the target position increases, or in reverse motion when the target position decreases. The linear coordinate system can set positive and negative software limits.

The rotary coordinate system includes a zero point and a rotation cycle in which CW motion occurs when the target position increases and CCW motion occurs when the target position decreases.

Pulse unit

It refers to the units measured in pulses and used on the drive, which are usually represented by pulse.

The drive usually contains two parameters: the pulse zero point and the number of pulses of an encoder per revolution of the motor.

12.1.4 Axis Configuration Parameters

Attributes of motion control axes can be set as needed. The following table summarizes the axis configuration parameters.

Category	Description	Bus Servo Axis	Local Pulse Axis
Basic settings	Axis number	√	√
	Axis type	√	√
	Input device	x	x
	Output device	√	√
	Automatic mapping	√	x
	Virtual axis mode	√	√
	PDO	√	x
Unit conversion settings	Reverse	√	√
	The number of instruction pulses per revolution of the motor/encoder	√	√
	The distance per revolution of the worktable in the background	√	√
	Gear ratio numerator	√	√
	Gear ratio denominator	√	√
Mode/parameter settings	Encoder mode	√	x
	Linear/rotary mode settings	√	√
	Software limit	√	√
	Software error response	√	√
	Following error	√	√
	Axis velocity settings	√	√
	Torque limit	√	x
	Probe settings	x	√
	Output settings	x	√
	Hardware limit logic	√	x
	Not entering ErrorStop state upon a limit activation	√	√

Category	Description	Bus Servo Axis	Local Pulse Axis
Homing settings	Home signal	√	√
	Positive limit	√	√
	Negative limit	√	√
	Z signal	√	x
	Homing direction	√	√
	Home input detection direction	√	√
	Homing list	√	√
	Homing velocity	√	√
	Homing closing velocity	√	√
	Homing acceleration	√	√
	Homing timeout time	√ [Note]	√
	Negative limit terminal settings	x	√
	Positive limit terminal settings	x	√
	Home signal settings	x	√
Online commissioning	Monitoring list	√	√
	Motion commissioning	√	√

[Note]: This function is only available to Inovance servo drives.

12.1.5 Axis System Variables

In the program, you can monitor the current status of an axis through its system variables. The system variables of a bus servo axis/local pulse axis are shown in the following table.

Variable	Data Type	Function
bPowerState	BOOL	Monitoring parameter, enabled or disabled status of the axis, read-only
bDebugState	BOOL	Monitoring parameter, commissioning status of the axis, read-only
fSetPosition	REAL	Monitoring parameter, position reference of the axis, user unit, read-only
fSetVelocity	REAL	Monitoring parameter, velocity reference of the axis (that is, the change rate of the position reference), user unit, read-only
fSet_Acc_Dec	REAL	Monitoring parameter, acceleration reference of the axis (that is, the change rate of the velocity reference), user unit, read-only
fSetTorque	REAL	Monitoring parameter, torque reference of the axis, user unit, read-only
fActPosition	REAL	Monitoring parameter, feedback position of the axis, user unit, read-only
fActVelocity	REAL	Monitoring parameter, the current velocity of the axis (that is, the change rate of the feedback position), user unit, read-only
fActAcc_Dec	REAL	Monitoring parameter, the current acceleration of the axis (that is, the change rate of the feedback velocity), user unit, read-only
fActTorque	REAL	Monitoring parameter, feedback torque of the axis, user unit, read-only

Variable	Data Type	Function
wPLCOpenState	INT	Monitoring parameter, PLCOpen state machine for the axis, read-only 0: PowerOff 1: ErrorStop 2: Stopping 3: StandStill 4: DiscreteMotion 5: ContinuousMotion 7: Homing 8: SynchronizedMotion
wConfigState	INT	Monitoring parameter, configuration status of the axis, read-only 0: Init (axis in the initialization state) 1: Configure finish (configuration reading completed) 2: Sync finish (synchronized with EtherCAT tasks) 3: Wait communication (communication with the servo drive established) 4: Slave ready (initialization completed for the servo drive controlled by axes) 5: Axis ready (communication established)
wAxisError	INT	Monitoring parameter, error code for the axis in the ErrorStop state, read-only
wServoError	INT	Monitoring parameter, error code for a drive or local axis, displaying 0x603f values, read-only
bEnterDebug	BOOL	Commissioning parameter, entering the commissioning mode when the variable is valid
bPowerOn	BOOL	Commissioning parameter, axis enable instruction
bStop	BOOL	Commissioning parameter, axis stop instruction
bReset	BOOL	Commissioning parameter, axis reset instruction
bJogP	BOOL	Commissioning parameter, forward jog instruction
bJogN	BOOL	Commissioning parameter, reverse jog instruction
bHome	BOOL	Commissioning parameter, homing instruction
bSetPos	BOOL	Commissioning parameter, current position setting instruction
bAbsPos	BOOL	Commissioning parameter, absolute positioning instruction
bRevPos	BOOL	Commissioning parameter, relative positioning instruction
bRelPos	BOOL	Commissioning parameter, reciprocating motion instruction
bVelocity	BOOL	Commissioning parameter, continuous motion instruction
bTorque	BOOL	Commissioning parameter, torque instruction
wDebugMotionType	INT	Commissioning parameter, commissioning mode 0: Idle 1: Relative positioning control 2: Absolute positioning control 3: Continuous motion control 5: Reciprocating motion control 6: Torque control
fJogVelocity	REAL	Commissioning parameter, jogging velocity
fPositionOffser	REAL	Commissioning parameter, homing offset

Variable	Data Type	Function
fPresetPosition	REAL	Commissioning parameter, preset positions
fTarPosition1	REAL	Commissioning parameter, target position
fTarVelocity1	REAL	Commissioning parameter, target velocity
fTarAcceleration1	REAL	Commissioning parameter, target acceleration
fTarDeceleration1	REAL	Commissioning parameter, target deceleration
wCurveType1	REAL	Commissioning parameter, curve type
fTarPosition2	REAL	Commissioning parameter, target position 2
fTarVelocity2	REAL	Commissioning parameter, target velocity 2
fTarAcceleration2	REAL	Commissioning parameter, target acceleration 2
fTarDeceleration2	REAL	Commissioning parameter, target deceleration 2
wCurveType2	INT	Commissioning parameter, curve type 2
dUnused		Commissioning parameter, reserved
fTarTorque	REAL	Commissioning parameter, target torque
fTarTorqueSlop	REAL	Commissioning parameter, torque slope
fLimitVelocity	REAL	Commissioning parameter, velocity limit in torque mode
wControlWord	INT	Loop variable, control word 0x6060, read-only
WStatusWord	INT	Loop variable, status word 0x6061, read-only
dSetPosition	DINT	Loop variable, target position 0x607a, read-only
dActPosition	DINT	Loop variable, feedback position 0x6064, read-only
dSetVelocity	DINT	Loop variable, velocity reference 0x60ff, read-only
dActVelocity	DINT	Loop variable, feedback velocity 0x606c, read-only
dSetTorque	DINT	Loop variable, torque reference 0x6071, read-only
dActTorque	DINT	Loop variable, feedback torque 0x6077, read-only
dDO	DINT	Loop variable, DO 0x60fe:1, read-only
dDI	DINT	Loop variable, DI 0x60fd, read-only
wModesOfOperation	INT	Loop variable, control mode 0x6060, read-only
wModesOfOperationDisplay	INT	Loop variable, current control mode 0x6061, read-only
wTouchFunction	INT	Loop variable, probe function settings 0x60b8, read-only
wTouchStatus	INT	Loop variable, probe status 0x60b9, read-only
dTouch1Ppos	DINT	Loop variable, probe 1 position on the rising edge 0x60ba, read-only
dTouch2Ppos	DINT	Loop variable, probe 2 position on the rising edge 0x60bb, read-only
dTouch1Npos	DINT	Loop variable, probe 1 position on the falling edge 0x60bc, read-only
dTouch2Npos	DINT	Loop variable, probe 2 position on the falling edge 0x60bd, read-only
dMaxVelocity	DINT	Loop variable, maximum velocity 0x607f, maximum velocity
wErrorCode	INT	Loop variable, drive error code 0x603f, read-only
wAxisRingPos	INT	Configuration parameter, axis configuration position, power-on initialization
WAxisID	INT	Configuration parameter, axis ID, power-on initialization
fUnits	REAL	Configuration parameter, axis gear ratio, power-on initialization
fFilter	REAL[3]	Setting the filter coefficient of the master axis, initialized to 1.0,.0.0,0.0 upon power-on, and modifiable
bMotionState	BOOL	Monitoring parameter, motion status, indicating whether the axis is in motion, read-only
bphlimit	BOOL	Monitoring parameter, hardware positive limit input status, read-only
bnhlimit	BOOL	Monitoring parameter, hardware negative limit input status, read-only

Variable	Data Type	Function
bhomestate	BOOL	Monitoring parameter, hardware home switch input status, read-only
bpslimit	BOOL	Monitoring parameter, indicating whether the software positive limit is reached, read-only
bnslimit	BOOL	Monitoring parameter, indicating whether the software negative limit is reached, read-only
dLocialAxisSetPos	DINT	Monitoring parameter, position reference of the local pulse axis, read-only
fFollowPos	REAL	Following error, read-only

12.1.6 List of Axis Control Instructions

The following table lists single-axis control instructions. For details about how to use these instructions, see the *H5U and Easy Series Programmable Logic Controllers Instructions Guide*.

Instruction	Name
MC_Power	Enable control
MC_Reset	Fault reset
MC_ReadStatus	Axis status reading
MC_ReadAxisError	Axis error reading
MC_ReadDigitalInput	Digital input reading
MC_ReadActualPosition	Actual position reading
MC_ReadActualVelocity	Actual velocity reading
MC_ReadActualTorque	Actual torque reading
MC_SetPosition	Position setting
MC_TouchProbe	Probe
MC_MoveRelative	Relative positioning
MC_MoveAbsolute	Absolute positioning
MC_MoveVelocity	Velocity
MC_Jog	Jogging
MC_TorqueControl	Torque control
MC_Home	Homing
MC_Stop	Stop
MC_Halt	Pause
MC_ImmediateStop	Emergency stop
MC_MoveFeed	Interrupt positioning
MC_MoveBuffer	Multi-position positioning
MC_MoveSuperImposed	Motion superimposition
MC_MoveVelocityCSV	CSV-based velocity control with adjustable pulse width
MC_SyncMoveVelocity	CSV-based synchronous velocity control with adjustable pulse width
MC_SyncTorqueControl	Synchronous torque control

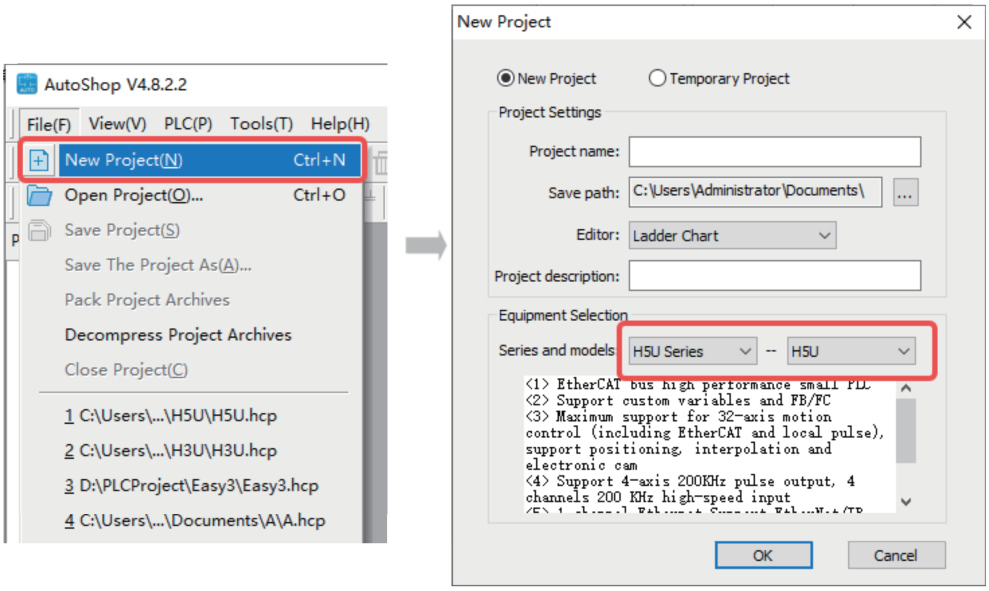
12.2 Setting Motion Control Axes

12.2.1 Creating a Project

Follow these steps to accurately control an axis: First, create a configuration based on the needs of your project. Then, set relevant parameters based on the working conditions, download the project, and carry out simple operations through online commissioning to determine whether the parameter settings are reasonable and whether the hardware connection is reliable. Lastly, write a PLC program to complete the overall control logic. Here is an example.

This routine creates a new bus servo axis and a local pulse axis, and implements simple motion through two ways: online commissioning interface and instructions.

1. Open AutoShop, click "New Project", and set the PLC type to H5U.


AutoShop V4.8.2.2
File(F)
View(V)
PLC(P)
Tools(T)
Help(H)
New Project(N) Ctrl+N
Open Project(O)... Ctrl+O
Save Project(S)
Save The Project As(A)...
Pack Project Archives
Decompress Project Archives
Close Project(C)
1 C:\Users\...\H5U\H5U.hcp
2 C:\Users\...\H3U\H3U.hcp
3 D:\PLCProject\Easy3\Easy3.hcp
4 C:\Users\...\Documents\A\A.hcp
New Project
OK
Cancel

● New Project ○ Temporary Project

Project Settings

Project name:

Save path: C:\Users\Administrator\Documents\

Editor: Ladder Chart

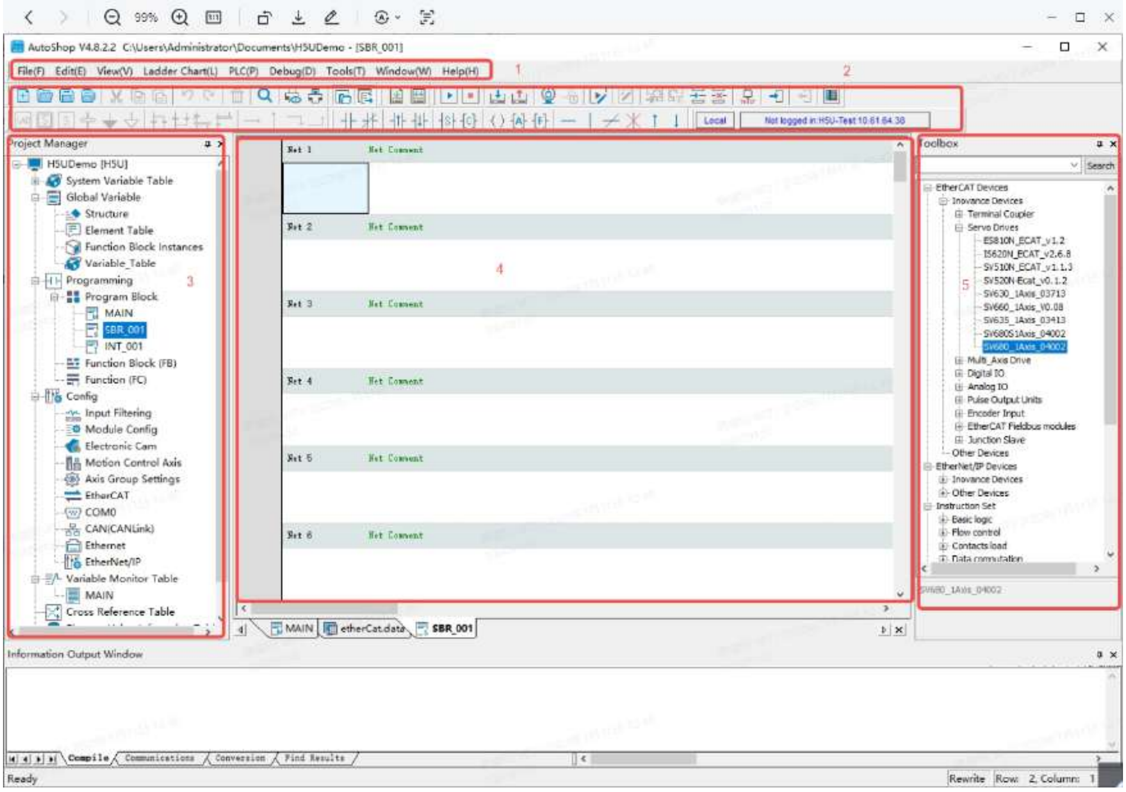
Project description:

Equipment Selection

Series and models: H5U Series -- H5U

```
<1> EtherCAT bus high performance small PLC
<2> Support custom variables and FB/FC
<3> Maximum support for 32-axis motion
control (including EtherCAT and local pulse),
support positioning, interpolation and
electronic cam
<4> Support 4-axis 200KHz pulse output, 4
channels 200 KHz high-speed input
```

2. After the project is successfully created, enter the main interface.



Zone 1: Menu bar

Zone 2: Toolbar

Zone 3: Project management area

Zone 4: Program editing section

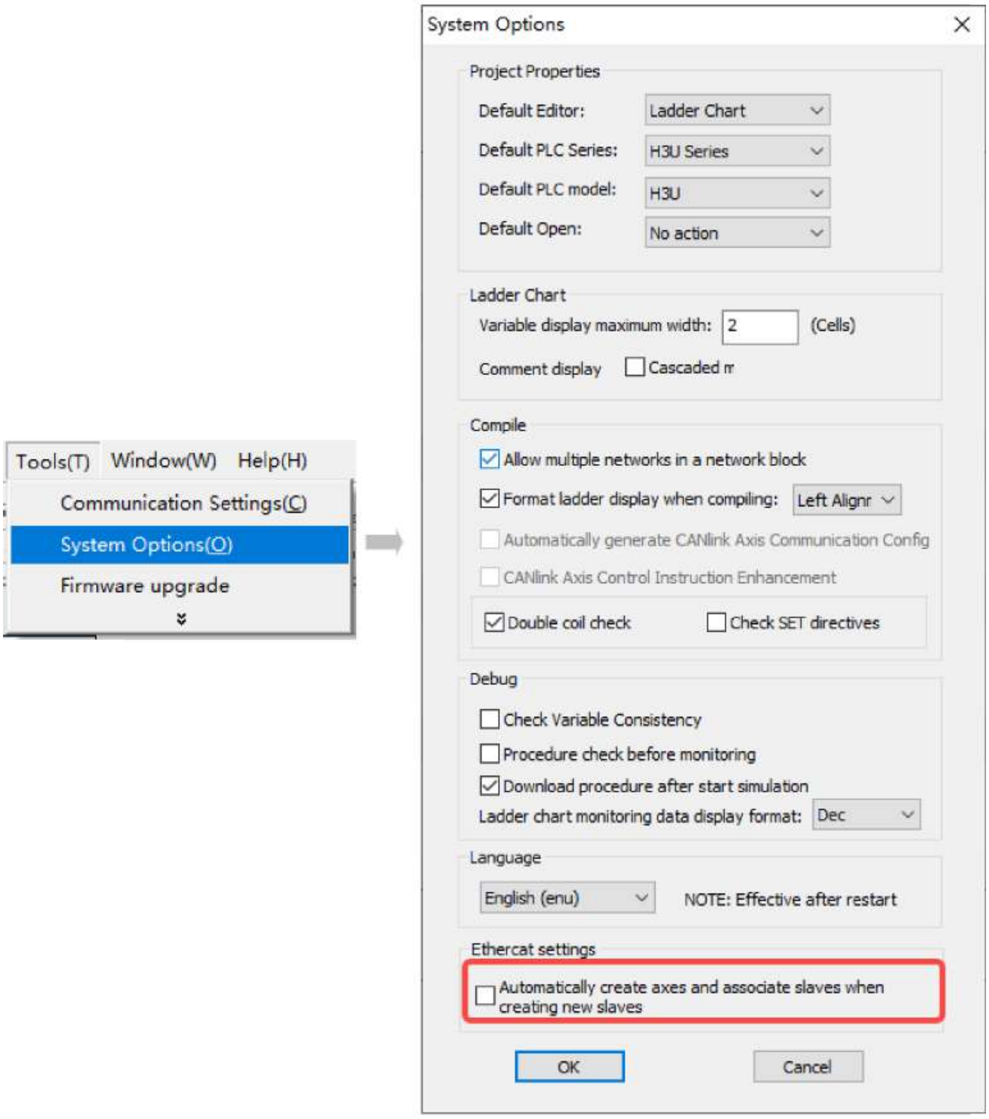
Zone 5: Toolbox

12.2.2 Creating Project Configuration

To control the IS620N motion, you can configure a servo drive and a bus servo axis and link them together in two modes: automatic scan and manual adding.
In the automatic scan mode, bus servo axes are automatically added, while local pulse axes need to be added manually. The operations of the two modes are explained below.

Automatic scan

1. In "System Options", check whether the option "Automatically create axes and associate slaves when creating new slaves" is ticked. If not, tick it.



System Options

Project Properties

Default Editor: Ladder Chart

Default PLC Series: H3U Series

Default PLC model: H3U

Default Open: No action

Ladder Chart

Variable display maximum width: 2 (Cells)

Comment display □ Cascaded m

Compile

☑ Allow multiple networks in a network block

☑ Format ladder display when compiling: Left Alignr

□ Automatically generate CANlink Axis Communication Config

□ CANlink Axis Control Instruction Enhancement

☑ Double coil check □ Check SET directives

Debug

□ Check Variable Consistency

□ Procedure check before monitoring

☑ Download procedure after start simulation

Ladder chart monitoring data display format: Dec

Language

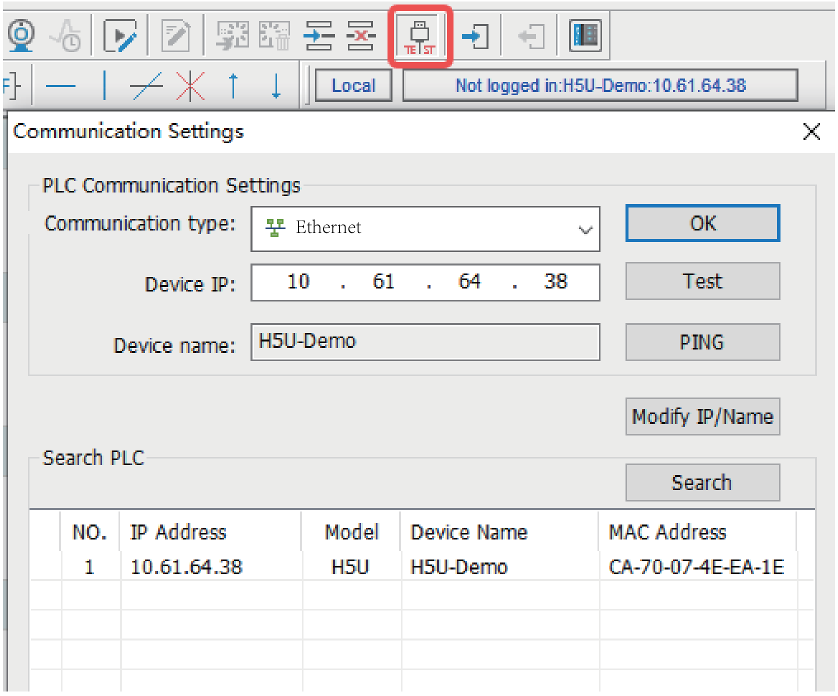
English (enu) NOTE: Effective after restart

Ethercat settings

□ Automatically create axes and associate slaves when creating new slaves

2. Check whether the computer host is normally connected to a PLC and whether the EtherCAT network port of the PLC is normally connected to a servo drive.

   The way to test the connection of PLC to the computer host is as follows.



Communication Settings

PLC Communication Settings

Communication type: Ethernet

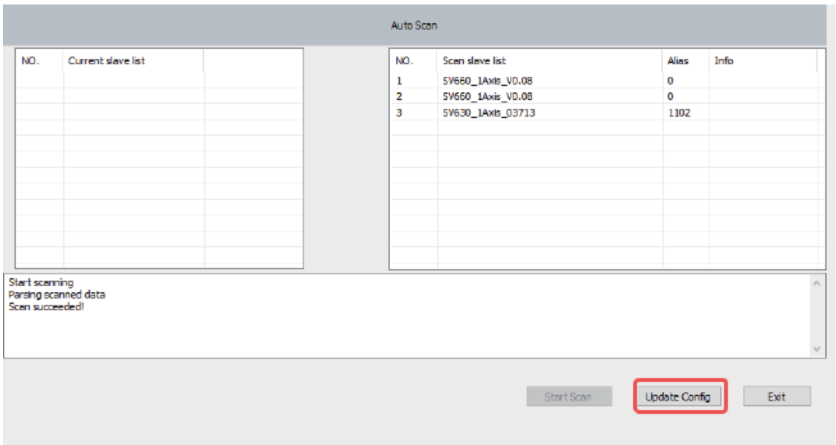
Device IP: 10 . 61 . 64 . 38

Device name: H5U-Demo

Search PLC

NO.	IP Address	Model	Device Name	MAC Address
1	10.61.64.38	H5U	H5U-Demo	CA-70-07-4E-EA-1E

3. In the toolbox, check whether the EtherCAT device list includes the IS620N. If not, add the corresponding XML file.
4. Select the master station, right-click, select "Auto Scan", and the "Auto Scan" dialog box will pop up.
5. Click "Start Scan". After the scan is completed, click "Update Config" to complete the creation of a bus servo axis.



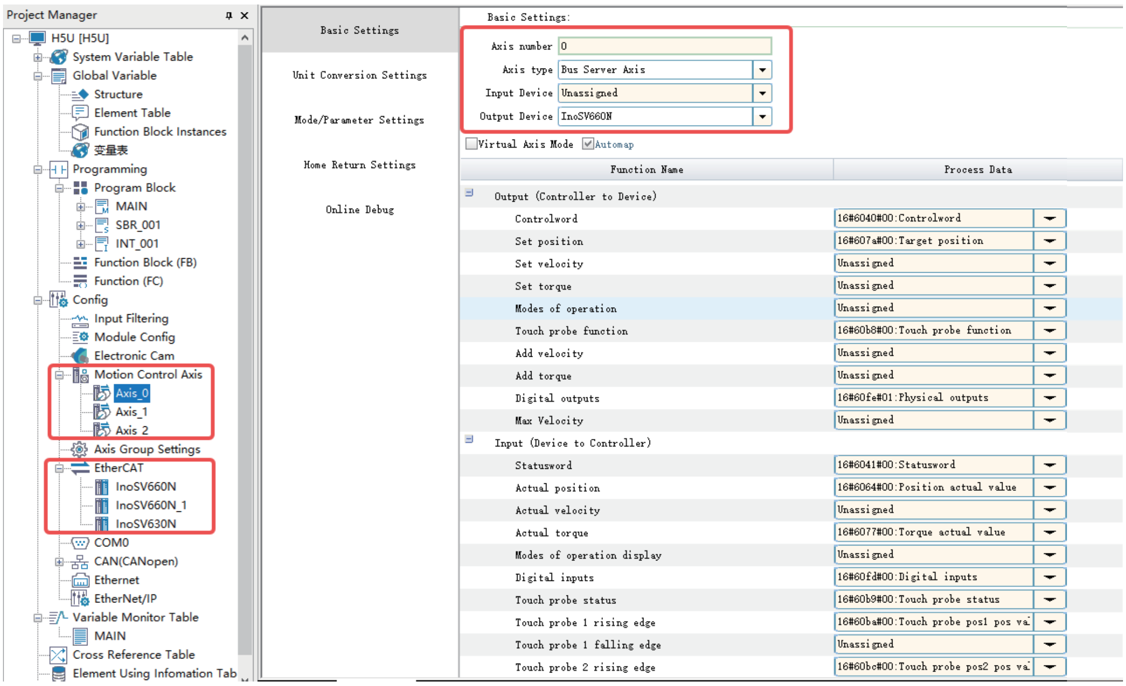
Auto Scan

NO.	Current slave list

NO.	Scan slave list	Alias	Info
1	SV660_1Axis_V0.08	0	
2	SV660_1Axis_V0.08	0	
3	SV630_1Axis_03713	1102	

```
Start scanning
Parsing scanned data
Scan succeeded!
```

6. After the scan is completed, you can see the servo drive and bus servo axis in the device tree.



Basic Settings:

Axis number 0

Axis type Bus Server Axis

Input Device Unassigned

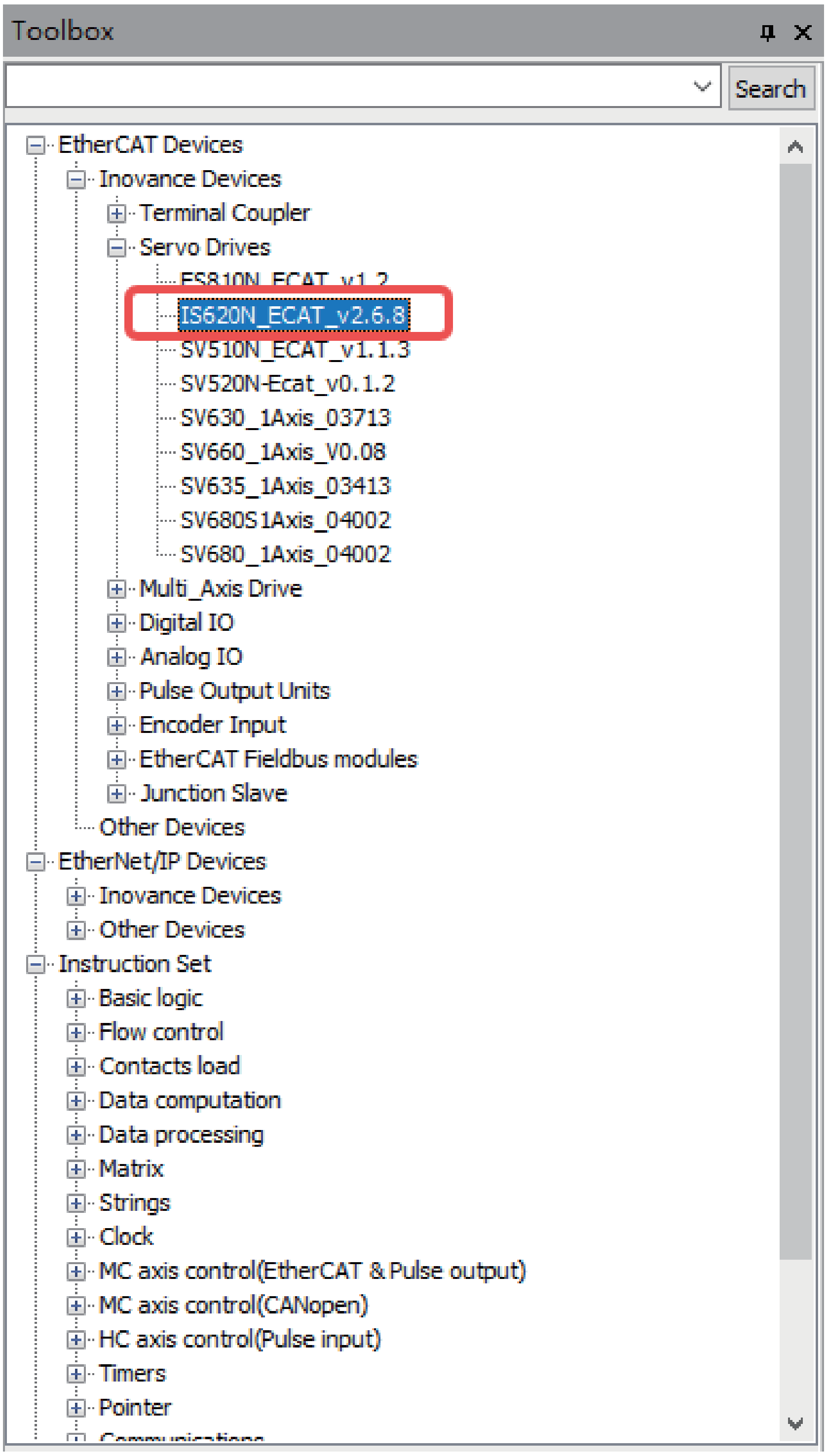
Output Device InoSV660N

□ Virtual Axis Mode ☑ Automap

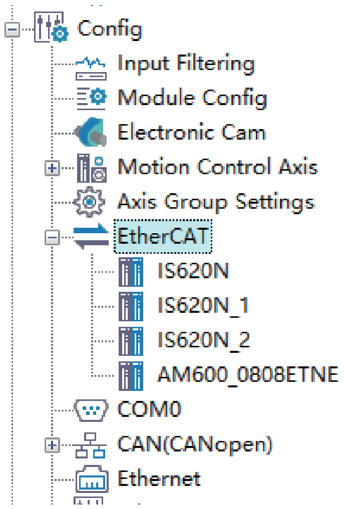
Function Name	Process Data
Output (Controller to Device)	
Controlword	16#6040#00:Controlword
Set position	16#607a#00:Target position
Set velocity	Unassigned
Set torque	Unassigned
Modes of operation	Unassigned
Touch probe function	16#60b8#00:Touch probe function
Add velocity	Unassigned
Add torque	Unassigned
Digital outputs	16#60fe#01:Physical outputs
Max Velocity	Unassigned
Input (Device to Controller)	
Statusword	16#6041#00:Statusword
Actual position	16#6064#00:Position actual value
Actual velocity	Unassigned
Actual torque	16#6077#00:Torque actual value
Modes of operation display	Unassigned
Digital inputs	16#60fd#00:Digital inputs
Touch probe status	16#60b9#00:Touch probe status
Touch probe 1 rising edge	16#60ba#00:Touch probe pos1 pos va
Touch probe 1 falling edge	Unassigned
Touch probe 2 rising edge	16#60bc#00:Touch probe pos2 pos va

Manual adding

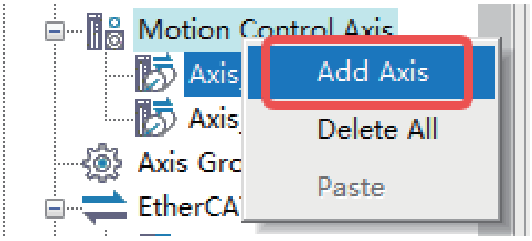
1. Open the toolbox and locate the IS620N. If it is unavailable, you can add the device profile ESI for IS620N to the toolbox by importing the file.



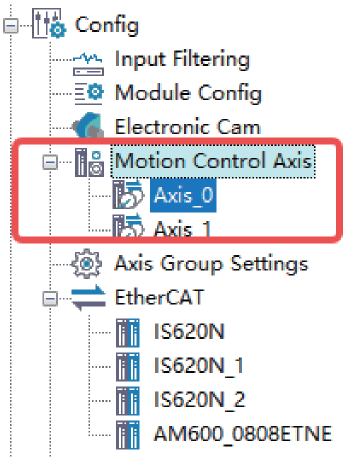
2. Double-click the IS620N in the toolbox to add an IS620N to the device tree EtherCAT configuration. If you have ticked "Automatically create axes and associate slaves when creating new slaves" in the "System Options", a bus servo axis will be added when you add the IS620N. Otherwise, the bus servo axis will not be added automatically. The assumption in this routine is that "Automatically create axes and associate slaves when creating new slaves" is not ticked.



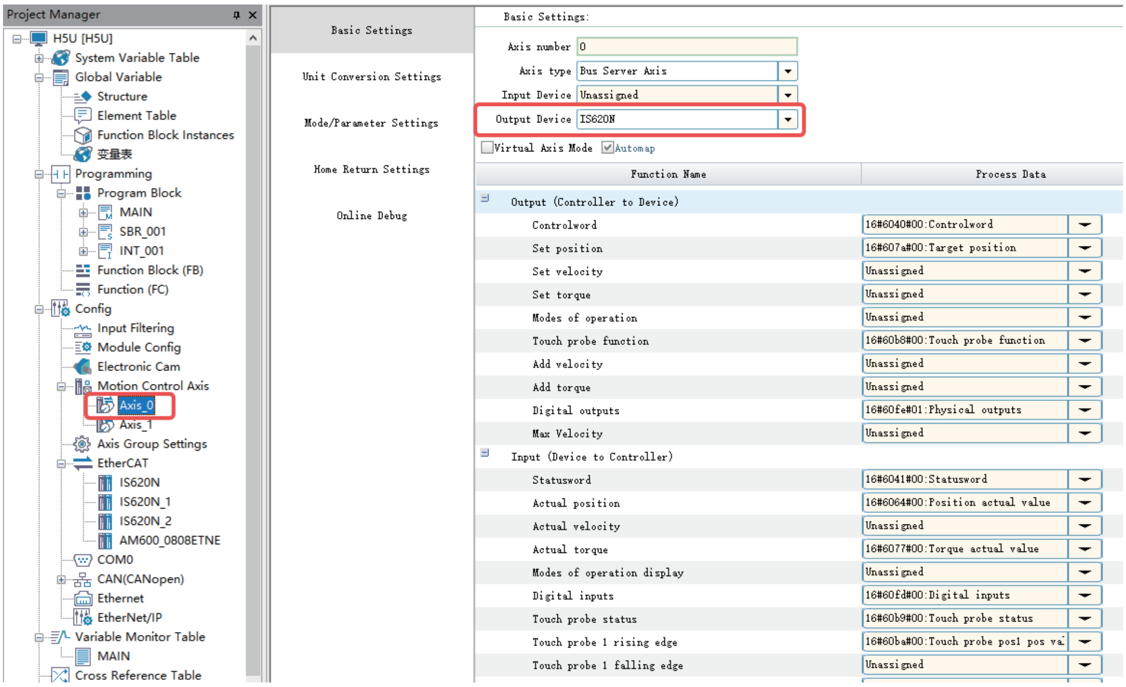
3. You can add a motion control axis by choosing "Motion Control Axis" in the device tree, and right-clicking the "Add Axis". You can establish two axes by repeating this operation twice.



Configuration after adding two axes:



4. Set the first axis as the bus servo axis and associate it with IS620N. Set the second axis to be the local pulse axis and associate it with the Y0/Y1 channel.



Basic Settings:

Axis number: 0

Axis type: Bus Server Axis

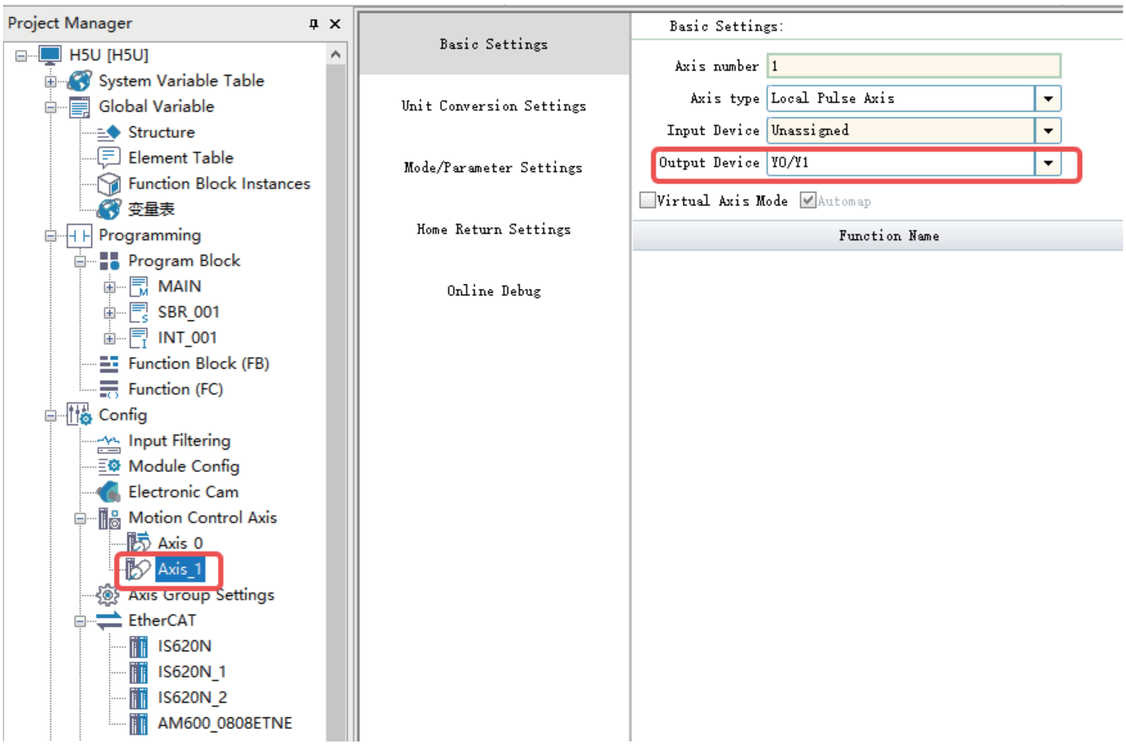
Input Device: Unassigned

Output Device: IS620N

□ Virtual Axis Mode ☑ Automap

Function Name	Process Data
Output (Controller to Device)	
Controlword	16#6040#00:Controlword
Set position	16#607a#00:Target position
Set velocity	Unassigned
Set torque	Unassigned
Modes of operation	Unassigned
Touch probe function	16#60b8#00:Touch probe function
Add velocity	Unassigned
Add torque	Unassigned
Digital outputs	16#60fe#01:Physical outputs
Max Velocity	Unassigned
Input (Device to Controller)	
Statusword	16#6041#00:Statusword
Actual position	16#6064#00:Position actual value
Actual velocity	Unassigned
Actual torque	16#6077#00:Torque actual value
Modes of operation display	Unassigned
Digital inputs	16#60fd#00:Digital inputs
Touch probe status	16#60b9#00:Touch probe status
Touch probe 1 rising edge	16#60ba#00:Touch probe pos1 pos va
Touch probe 1 falling edge	Unassigned

Figure 12-2 Adding the bus servo drive



Basic Settings:

Axis number: 1

Axis type: Local Pulse Axis

Input Device: Unassigned

Output Device: Y0/Y1

□ Virtual Axis Mode ☑ Automap

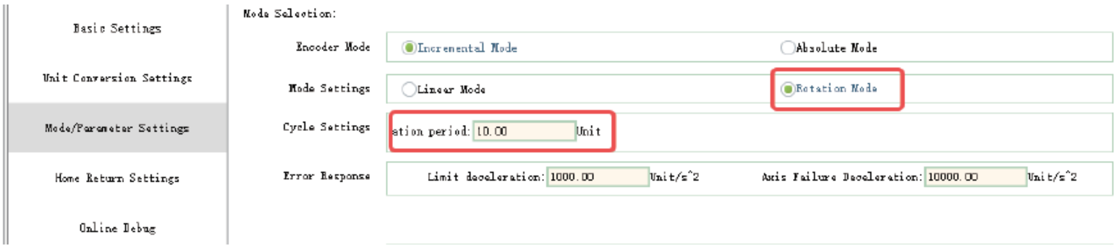
Figure 12-3 Local output axis

12.2.3 Setting Axis Parameters

12.2.3.1 Bus Servo Axis

You can set the relevant parameters of an axis based on actual working conditions and requirements. The settings in this routine are as follows.

1. Set the mode to rotary and the rotation cycle to 10.



Mode Selection:

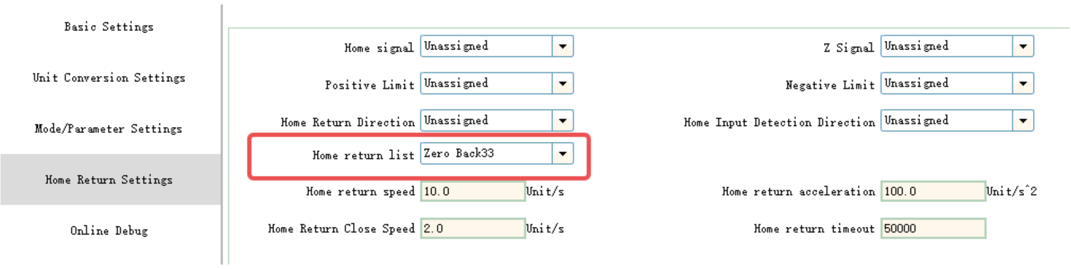
Encoder Mode ● Incremental Mode ○ Absolute Mode

Mode Settings ○ Linear Mode ● Rotation Mode

Cycle Settings ation period: 10.00 Unit

Error Response Limit deceleration: 1000.00 Unit/s^2 Axis Failure Deceleration: 10000.00 Unit/s^2

2. Set the homing mode to 33.



Home signal Unassigned Z Signal Unassigned

Positive Limit Unassigned Negative Limit Unassigned

Home Return Direction Unassigned Home Input Detection Direction Unassigned

Home return list Zero Back33

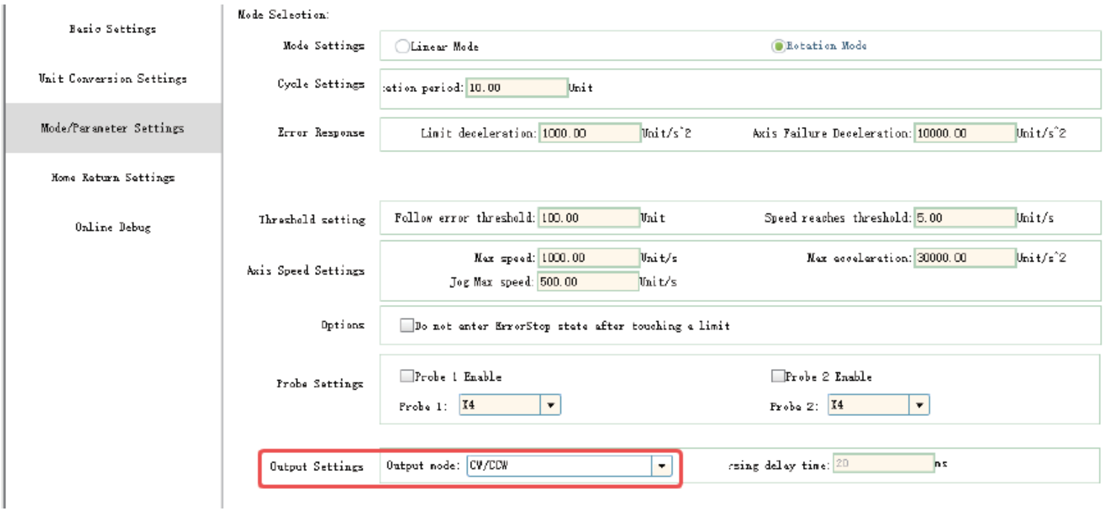
Home return speed 10.0 Unit/s Home return acceleration 100.0 Unit/s^2

Home Return Close Speed 2.0 Unit/s Home return timeout 50000

12.2.3.2 Local Pulse Axis

You can set the relevant parameters of an axis based on actual working conditions and requirements. The settings in this routine are as follows.

1. Set the pulse output mode to CW/CCW.



Mode Selection:

Mode Settings ○ Linear Mode ● Rotation Mode

Cycle Settings :ation period: 10.00 Unit

Error Response Limit deceleration: 1000.00 Unit/s^2 Axis Failure Deceleration: 10000.00 Unit/s^2

Threshold setting Follow error threshold: 100.00 Unit Speed reaches threshold: 5.00 Unit/s

Axis Speed Settings Max speed: 1000.00 Unit/s Max acceleration: 30000.00 Unit/s^2

Jog Max speed: 500.00 Unit/s

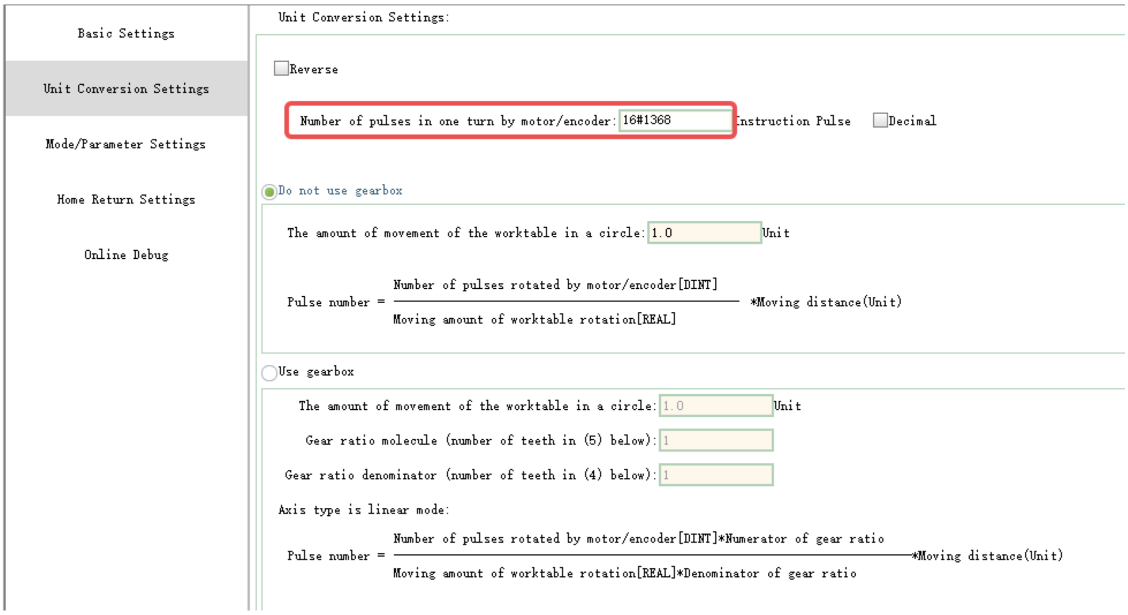
Options □ Do not enter ErrorStop state after touching a limit

Probe Setting □ Probe 1 Enable □ Probe 2 Enable

Probe 1: X4 Probe 2: X4

Output Settings Output mode: CW/CCW :rsing delay time: 20 ns

2. Set the "Pulses per motor/encoder revolution" to 5000 (16#1388).



Unit Conversion Settings:

□ Reverse

Number of pulses in one turn by motor/encoder: 16#1368 Instruction Pulse □ Decimal

● Do not use gearbox

The amount of movement of the worktable in a circle: 1.0 Unit

$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]}}{\text{Moving amount of worktable rotation[REAL]}} \text{*Moving distance(Unit)}$$

○ Use gearbox

The amount of movement of the worktable in a circle: 1.0 Unit

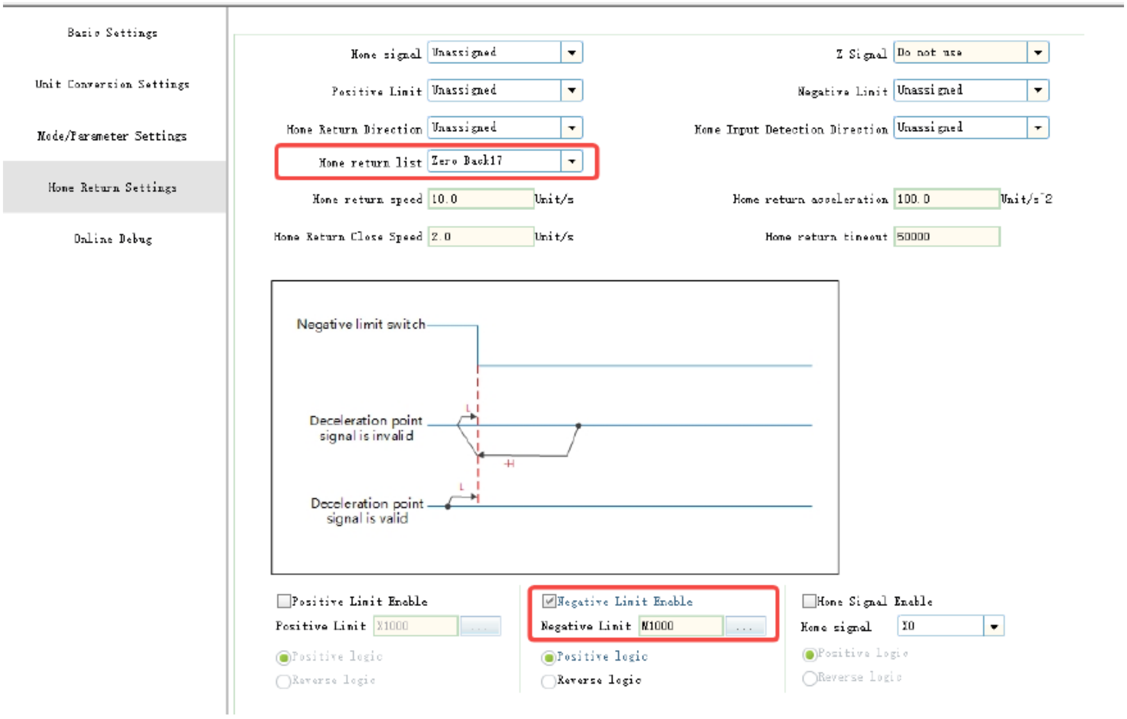
Gear ratio molecule (number of teeth in (5) below): 1

Gear ratio denominator (number of teeth in (4) below): 1

Axis type is linear mode:

$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]*Numerator of gear ratio}}{\text{Moving amount of worktable rotation[REAL]*Denominator of gear ratio}} \text{*Moving distance(Unit)}$$

3. Set the homing mode to 17 and the negative limit to M1000.



Home signal: Unassigned

Z Signal: Do not use

Positive Limit: Unassigned

Negative Limit: Unassigned

Home Return Direction: Unassigned

Home Input Detection Direction: Unassigned

Home return list: Zero Back17

Home return speed: 10.0 Unit/s

Home return acceleration: 100.0 Unit/s^2

Home Return Close Speed: 2.0 Unit/s

Home return timeout: 50000

□ Positive Limit Enable

Positive Limit: X1000

● Positive logic

○ Reverse logic

☑ Negative Limit Enable

Negative Limit: M1000

● Positive logic

○ Reverse logic

□ Home Signal Enable

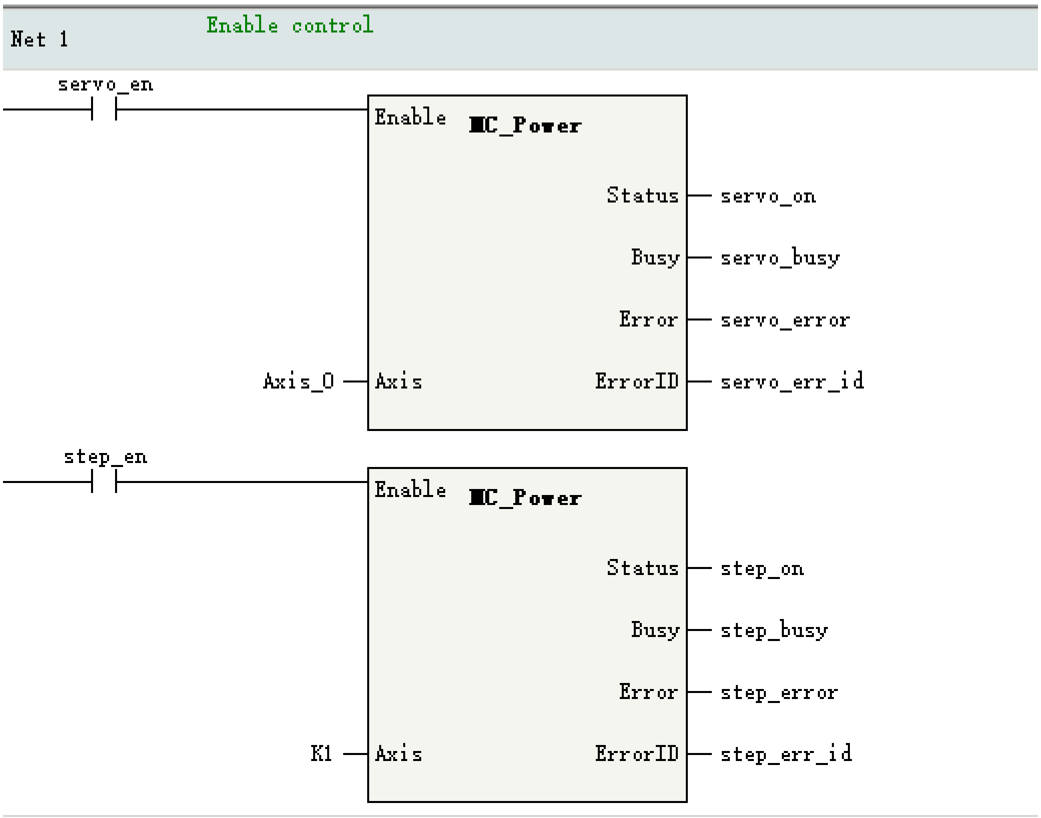
Home signal: X0

● Positive logic

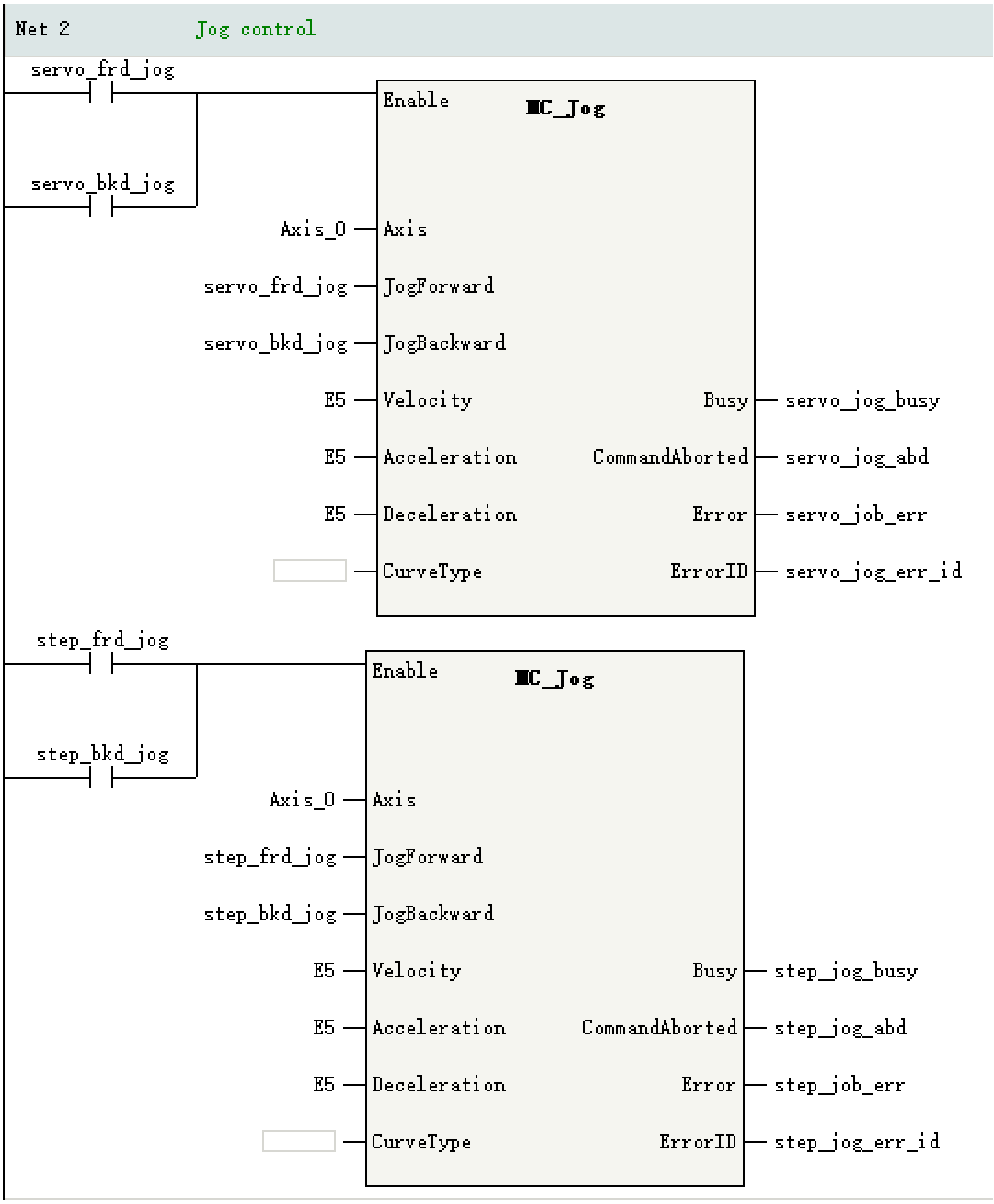
○ Reverse logic

12.2.4 Writing a Program

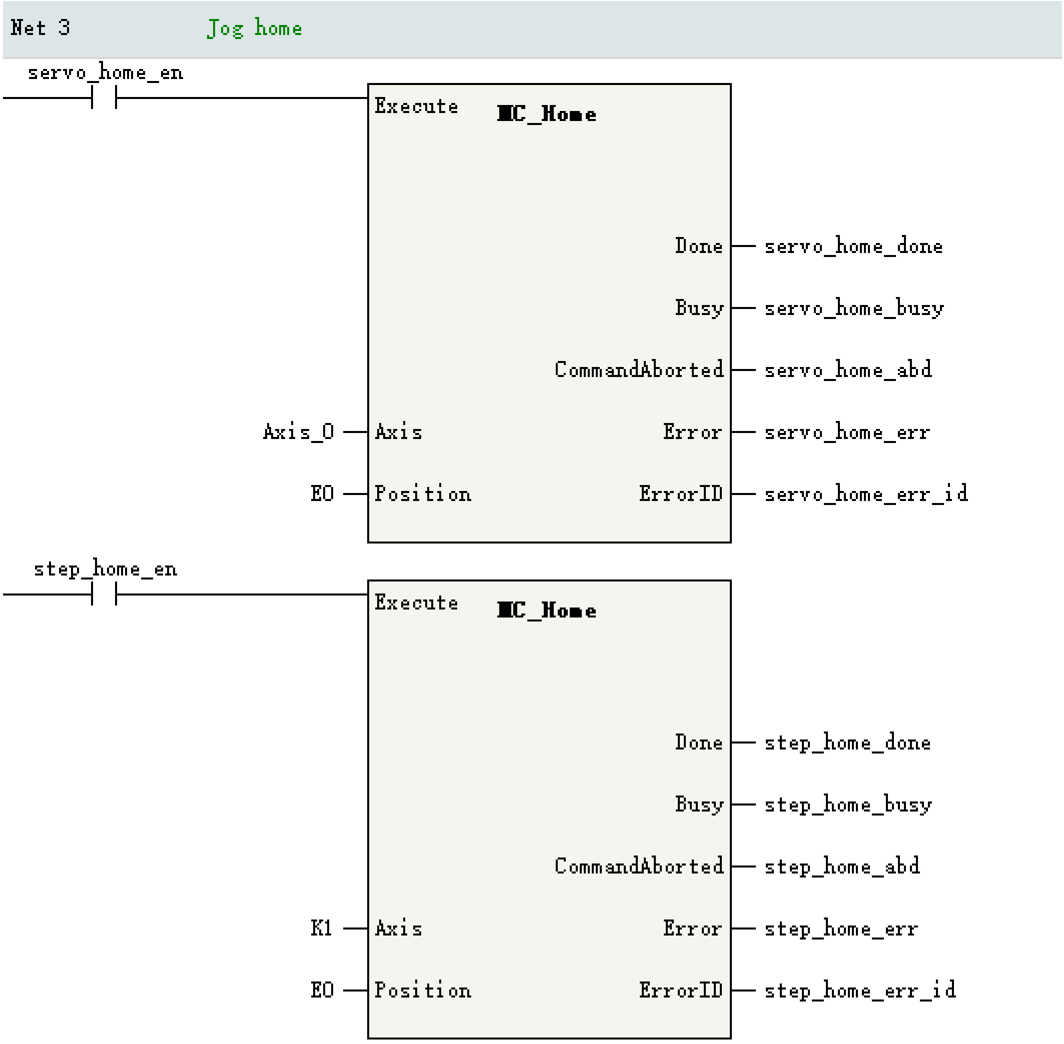
1. The function block MC_Power is used to control the enabling of an axis. MC_Power is an instruction and therefore does not need to be instantiated. This also applies to the following instructions.



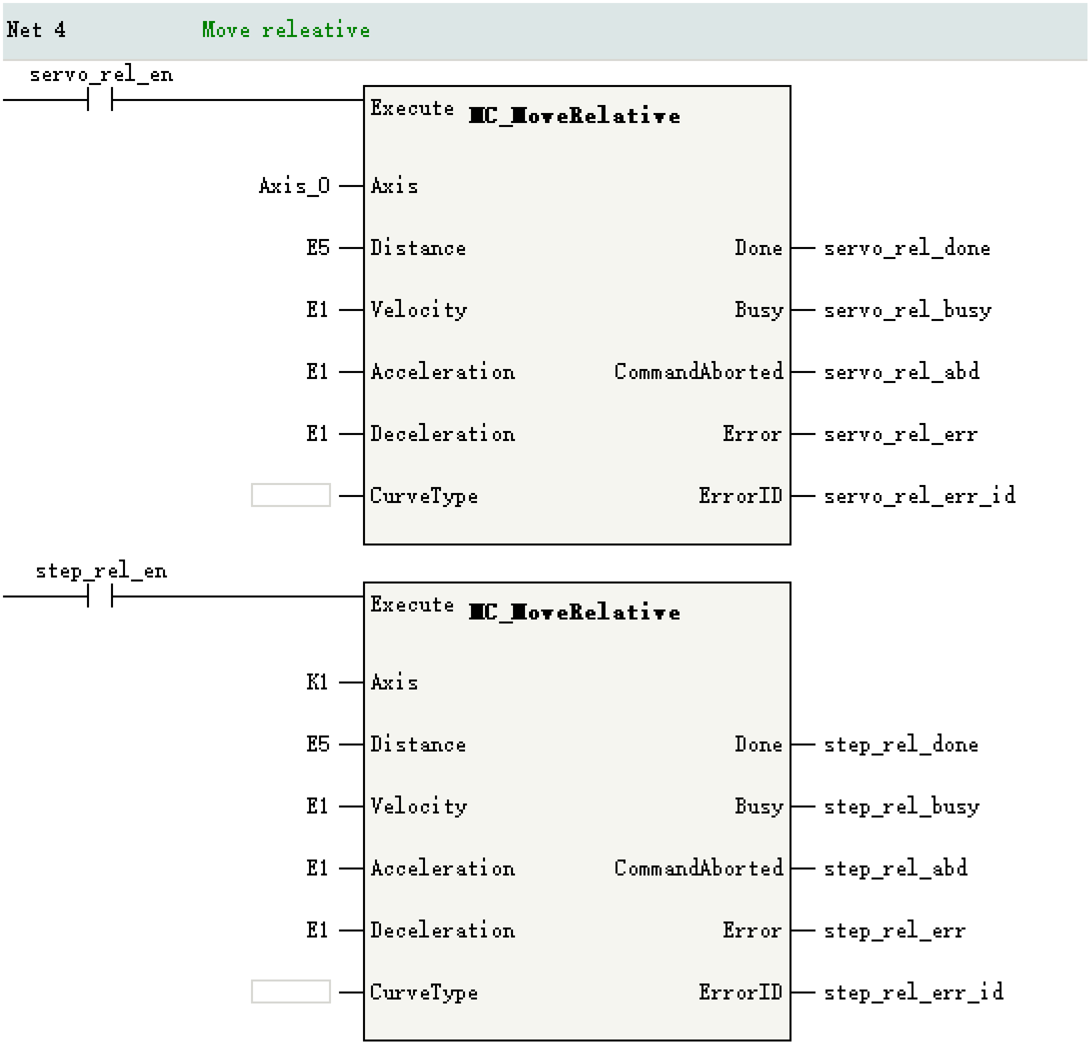
2. Call instruction MC_Jog for a test.



3. Call instruction MC_Home for a test.



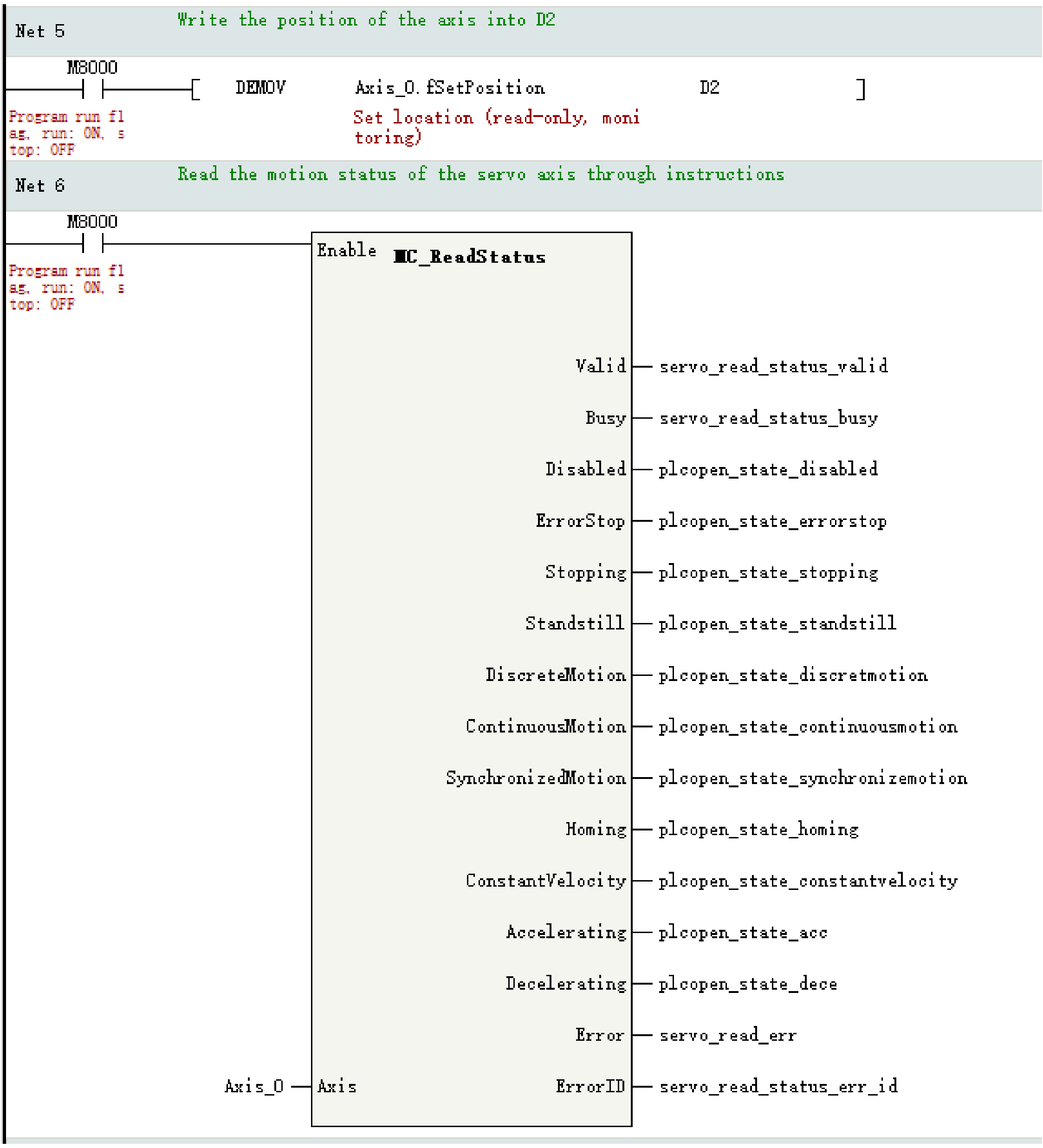
4. Call instruction MC_MoveRelative to test discrete motion.



```
Net 4          Move releative
 servo_rel_en
--| |------------------- Execute  MC_MoveRelative
              Axis_0 --- Axis
                  E5 --- Distance                Done --- servo_rel_done
                  E1 --- Velocity                Busy --- servo_rel_busy
                  E1 --- Acceleration  CommandAborted --- servo_rel_abd
                  E1 --- Deceleration           Error --- servo_rel_err
                     --- CurveType            ErrorID --- servo_rel_err_id
 step_rel_en
--| |------------------- Execute  MC_MoveRelative
                  K1 --- Axis
                  E5 --- Distance                Done --- step_rel_done
                  E1 --- Velocity                Busy --- step_rel_busy
                  E1 --- Acceleration  CommandAborted --- step_rel_abd
                  E1 --- Deceleration           Error --- step_rel_err
                     --- CurveType            ErrorID --- step_rel_err_id
```

5. Monitoring axis status

In the PLC program, the status of the axes can be monitored through function blocks or axis system variables.

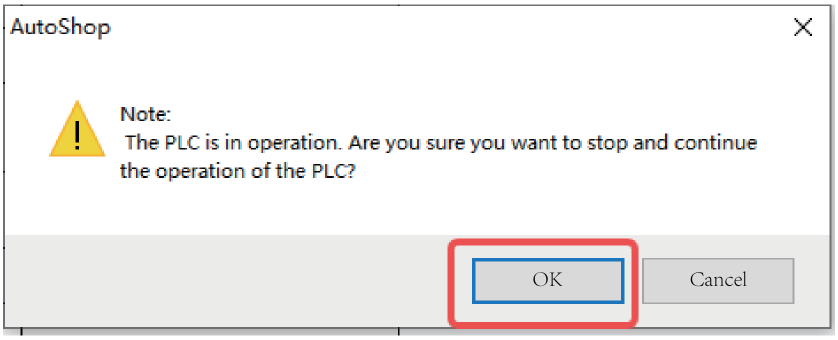


```
Net 5        Write the position of the axis into D2
    M8000
----| |--------[  DEMOV      Axis_0.fSetPosition          D2          ]
Program run fl               Set location (read-only, moni
ag, run: ON, s               toring)
top: OFF

Net 6        Read the motion status of the servo axis through instructions
    M8000
----| |----------------------- Enable  MC_ReadStatus
Program run fl
ag, run: ON, s
top: OFF
                                             Valid — servo_read_status_valid
                                              Busy — servo_read_status_busy
                                          Disabled — plcopen_state_disabled
                                         ErrorStop — plcopen_state_errorstop
                                          Stopping — plcopen_state_stopping
                                        Standstill — plcopen_state_standstill
                                    DiscreteMotion — plcopen_state_discretmotion
                                  ContinuousMotion — plcopen_state_continuousmotion
                                SynchronizedMotion — plcopen_state_synchronizemotion
                                            Homing — plcopen_state_homing
                                  ConstantVelocity — plcopen_state_constantvelocity
                                      Accelerating — plcopen_state_acc
                                      Decelerating — plcopen_state_dece
                                             Error — servo_read_err
                 Axis_0 — Axis             ErrorID — servo_read_status_err_id
```

12.2.5 Downloading a Project

After completing the programming and project setup, perform the download operation as follows.

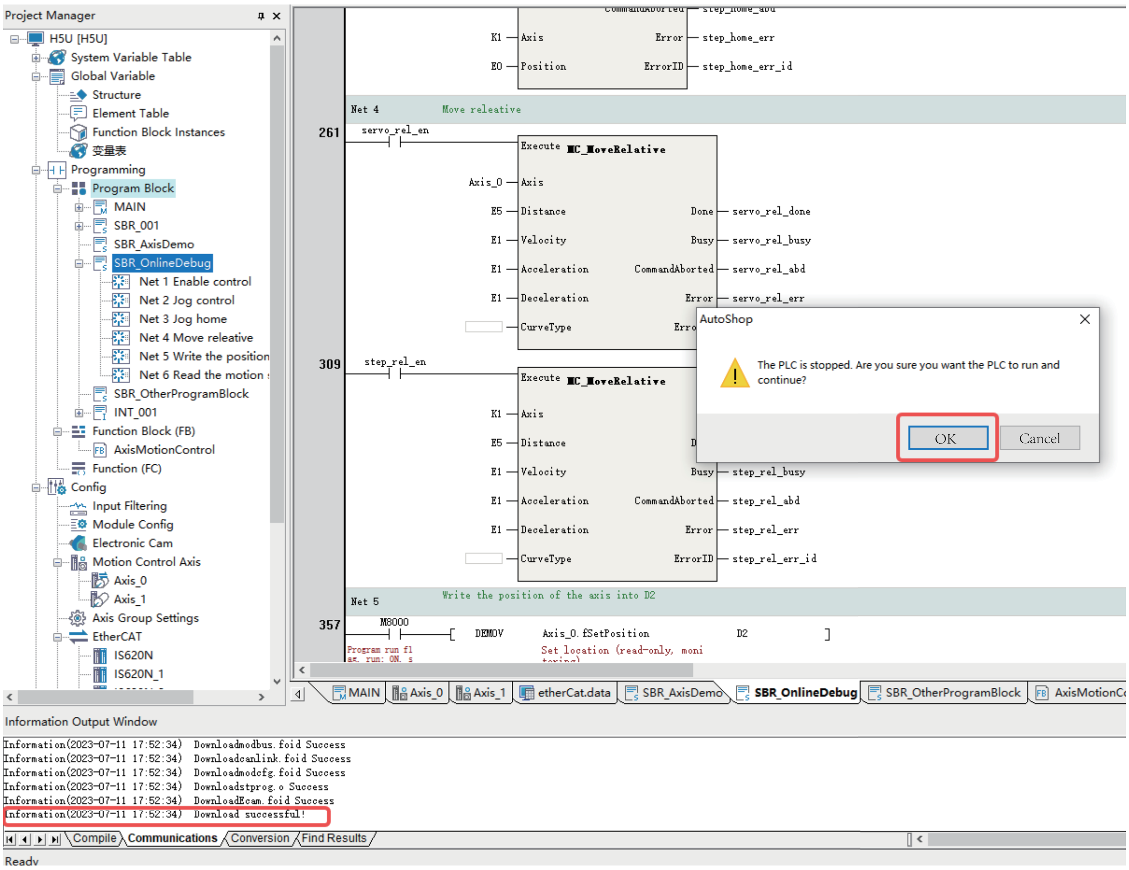
1. Click the download button , in which case the compilation operation is performed first.
2. After the compilation is completed, if the PLC is in the running state, the following dialog box will pop up. Select "OK" to go to step 3. If the PLC is in the stopped state, download the project directly and go to step 4.



AutoShop

Note:
The PLC is in operation. Are you sure you want to stop and continue the operation of the PLC?

3. If the PLC is in the running state before you download the program, you can see the download completion prompt after the download is completed. In the dialog box that pops up, select "OK" to switch the PLC to the running state.

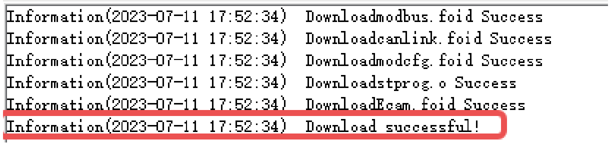


AutoShop

The PLC is stopped. Are you sure you want the PLC to run and continue?

```
Information(2023-07-11 17:52:34)  Downloadmodbus.foid Success
Information(2023-07-11 17:52:34)  Downloadcanlink.foid Success
Information(2023-07-11 17:52:34)  Downloadmodcfg.foid Success
Information(2023-07-11 17:52:34)  Downloadstprog.o Success
Information(2023-07-11 17:52:34)  DownloadEcam.foid Success
Information(2023-07-11 17:52:34)  Download successful!
```

4. If the PLC is in a stopped state before you download the program, the information output window will pop up the download completion prompt after the download is completed. Manually click the start button ▶ to switch the PLC to the running state.



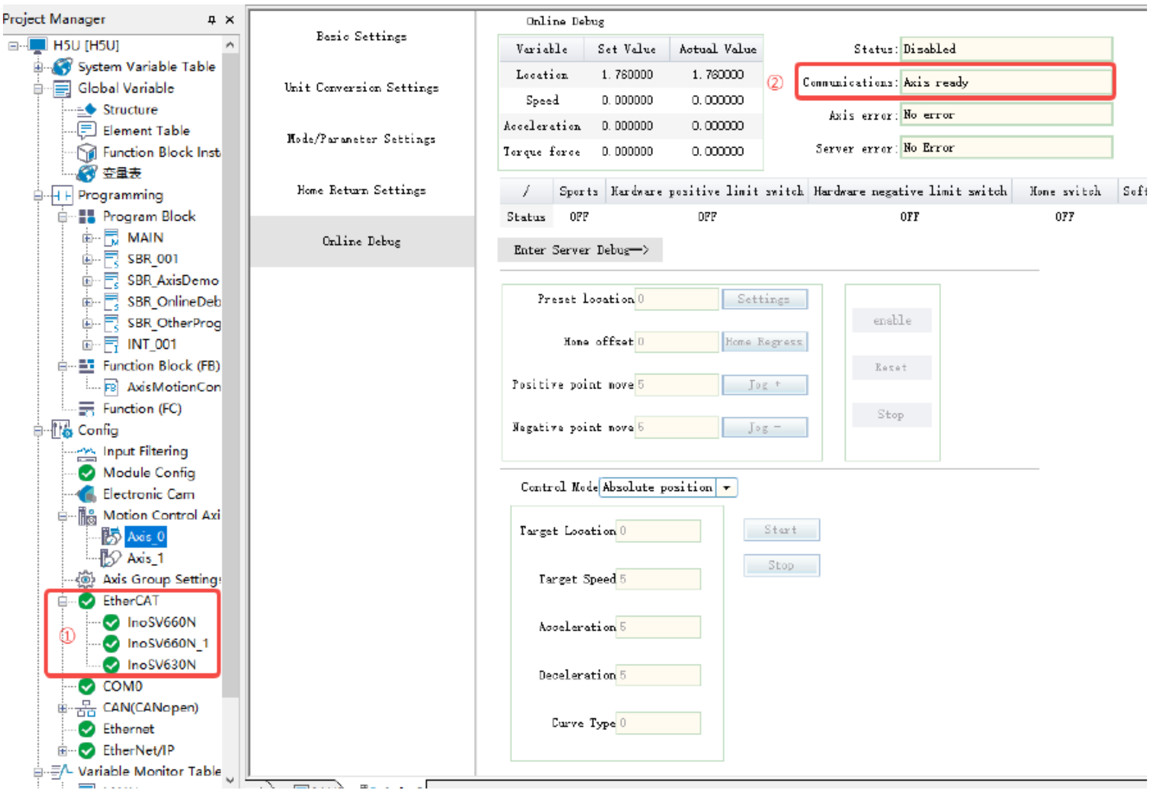
```
Information(2023-07-11 17:52:34)  Downloadmodbus.foid Success
Information(2023-07-11 17:52:34)  Downloadcanlink.foid Success
Information(2023-07-11 17:52:34)  Downloadmodcfg.foid Success
Information(2023-07-11 17:52:34)  Downloadstprog.o Success
Information(2023-07-11 17:52:34)  DownloadEcam.foid Success
Information(2023-07-11 17:52:34)  Download successful!
```

12.2.6 Basic Motions

12.2.6.1 Pre-conditions

To complete the basic action in this routine, it is recommended to first enter the monitoring mode, and click the monitoring button to enter the monitoring mode.

After entering the monitoring mode, you can see that the EtherCAT bus startup is completed and the servo axis initialization is completed.



Variable	Set Value	Actual Value
Location	1. 780000	1. 780000
Speed	0. 000000	0. 000000
Acceleration	0. 000000	0. 000000
Torque force	0. 000000	0. 000000

Status: Disabled

Communications: Axis ready

Axis error: No error

Server error: No Error

/	Sports	Hardware positive limit switch	Hardware negative limit switch	Home switch	Soft
Status	OFF	OFF	OFF	OFF	

Preset Location 0

Home offset 0

Positive point move 5

Negative point move 5

Control Mode Absolute position

Target Location 0

Target Speed 5

Acceleration 5

Deceleration 5

Curve Type 0

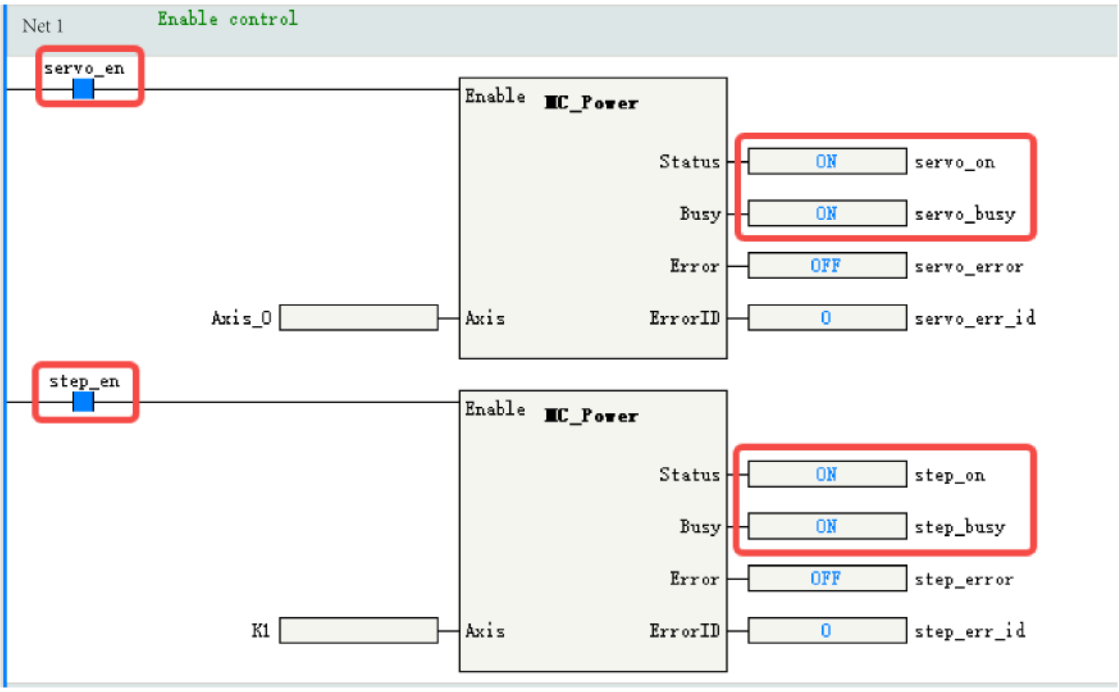
① EtherCAT initialization completed

② Axis initialization completed

The following describes the ways to control the servo axis motion through PLC program and online commissioning.

12.2.6.2 PLC Program Control

1. Enable control: Set the two BOOL variables of servo_en and step_en to TRUE, and the output of servo_on and step_on will be valid after the bus servo axis and local pulse axis are enabled.



Net 1 Enable control

servo_en

Enable MC_Power

Status ON servo_on

Busy ON servo_busy

Error OFF servo_error

Axis_0 Axis ErrorID 0 servo_err_id

step_en

Enable MC_Power

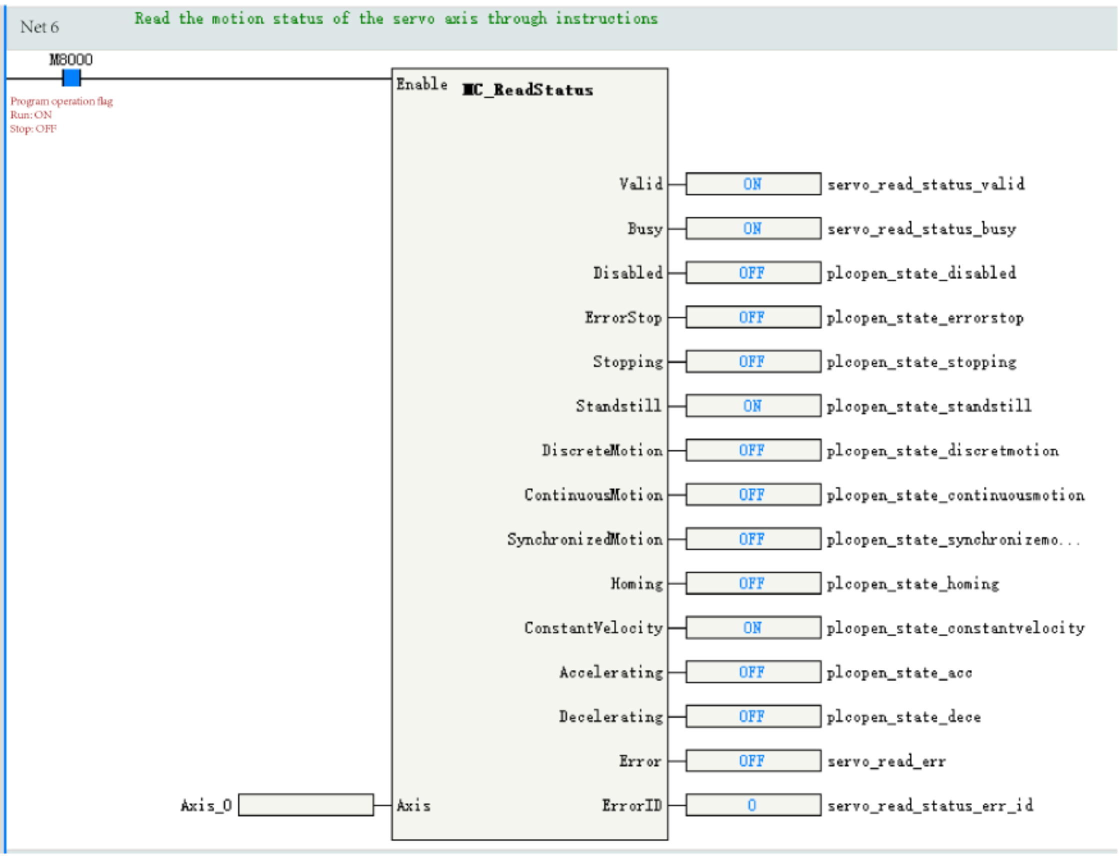
Status ON step_on

Busy ON step_busy

Error OFF step_error

K1 Axis ErrorID 0 step_err_id

Call the MC_ReadStatus instruction to view the status of the bus servo axis.



Net 6 Read the motion status of the servo axis through instructions

M8000

Program operation flag
Run: ON
Stop: OFF

Enable MC_ReadStatus

Valid ON servo_read_status_valid

Busy ON servo_read_status_busy

Disabled OFF plcopen_state_disabled

ErrorStop OFF plcopen_state_errorstop

Stopping OFF plcopen_state_stopping

Standstill ON plcopen_state_standstill

DiscreteMotion OFF plcopen_state_discretmotion

ContinuousMotion OFF plcopen_state_continuousmotion

SynchronizedMotion OFF plcopen_state_synchronizemo...

Homing OFF plcopen_state_homing

ConstantVelocity ON plcopen_state_constantvelocity

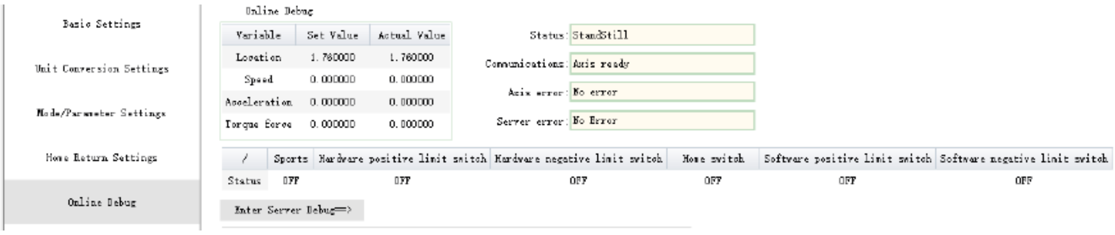
Accelerating OFF plcopen_state_acc

Decelerating OFF plcopen_state_dece

Error OFF servo_read_err

Axis_0 Axis ErrorID 0 servo_read_status_err_id

Alternatively, enter the online commissioning interface to view the axis status, which is standstill in this example.



Online Debug

Variable	Set Value	Actual Value
Location	1.780000	1.760000
Speed	0.000000	0.000000
Acceleration	0.000000	0.000000
Torque force	0.000000	0.000000

Status: StandStill

Communications: Axis ready

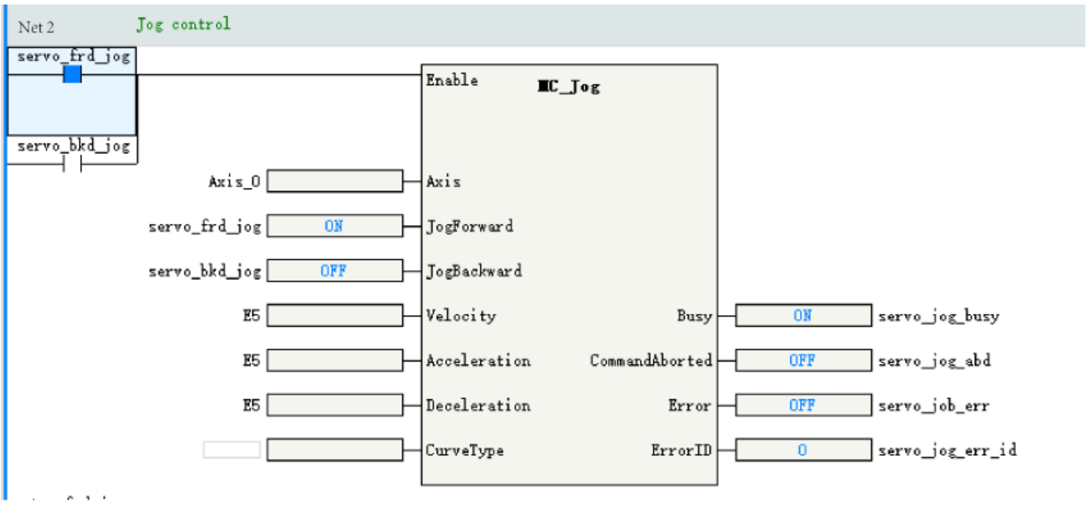
Axis error: No error

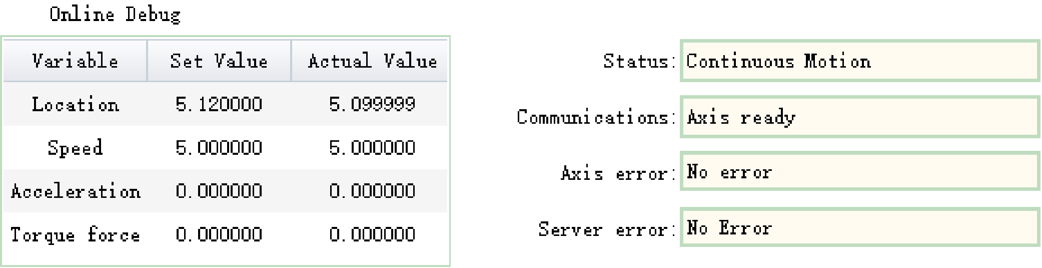
Server error: No Error

/	Sports	Hardware positive limit switch	Hardware negative limit switch	Home switch	Software positive limit switch	Software negative limit switch
Status	OFF	OFF	OFF	OFF	OFF	OFF

2. Jogging operation of the bus servo axis

- When the variable servo_frd_jog is set to TRUE, the bus servo axis starts to run forward at the velocity reference, and the actual axis driven by the servo drive runs forward at 5 rpm/s.





Online Debug

Variable	Set Value	Actual Value
Location	5.120000	5.099999
Speed	5.000000	5.000000
Acceleration	0.000000	0.000000
Torque force	0.000000	0.000000

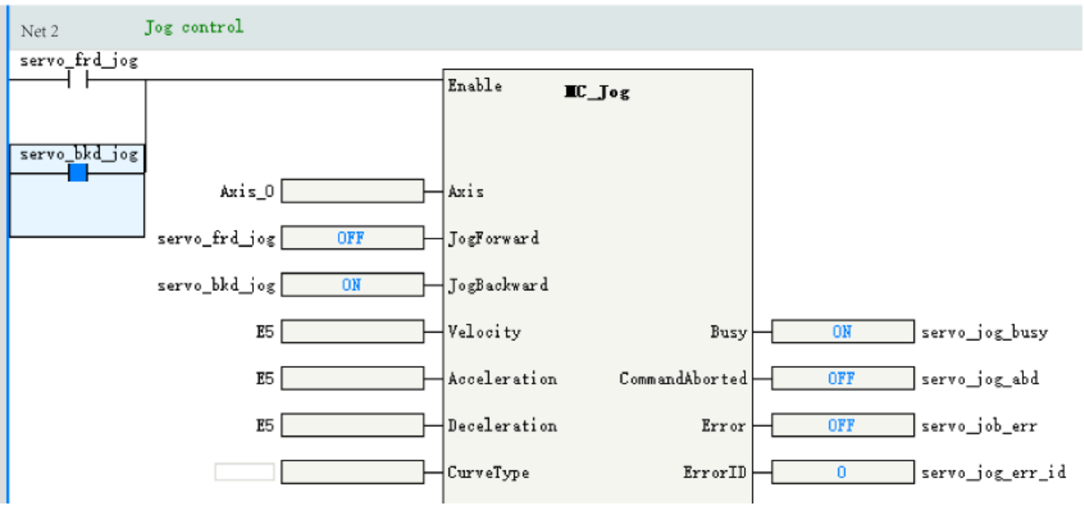
Status: Continuous Motion

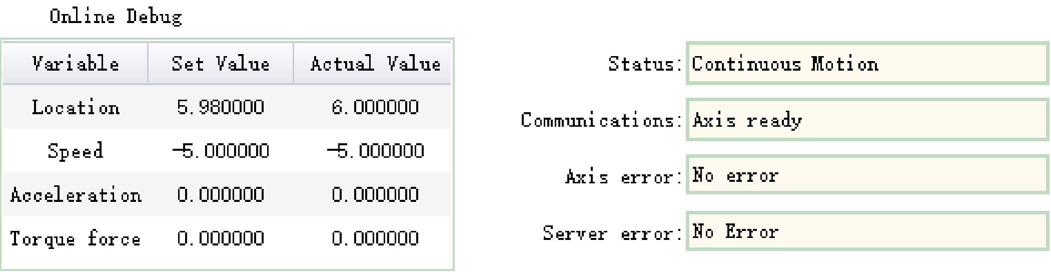
Communications: Axis ready

Axis error: No error

Server error: No Error

- When the variable servo_bkd_jog is set to TRUE, the bus servo axis starts to run in reverse at the velocity reference, and the actual axis driven by the servo drive runs in reverse at 5 rpm/s.





Online Debug

Variable	Set Value	Actual Value
Location	5.980000	6.000000
Speed	-5.000000	-5.000000
Acceleration	0.000000	0.000000
Torque force	0.000000	0.000000

Status: Continuous Motion

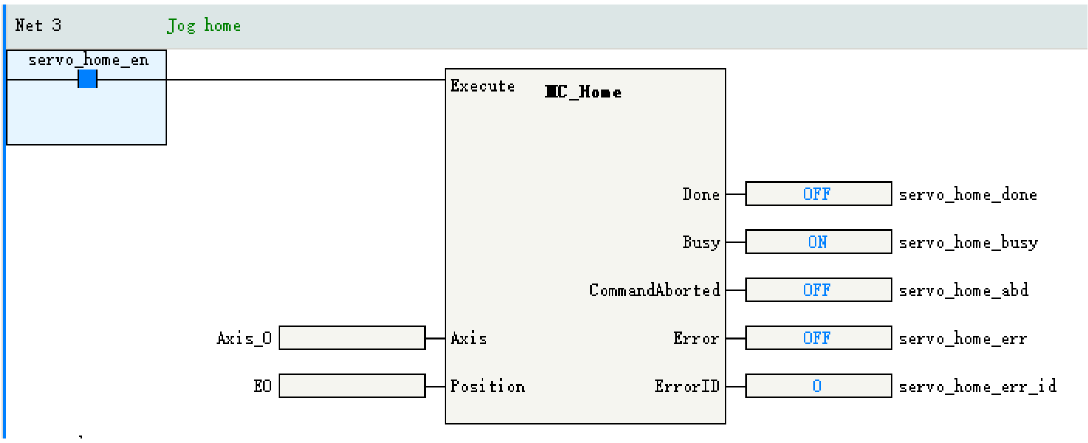
Communications: Axis ready

Axis error: No error

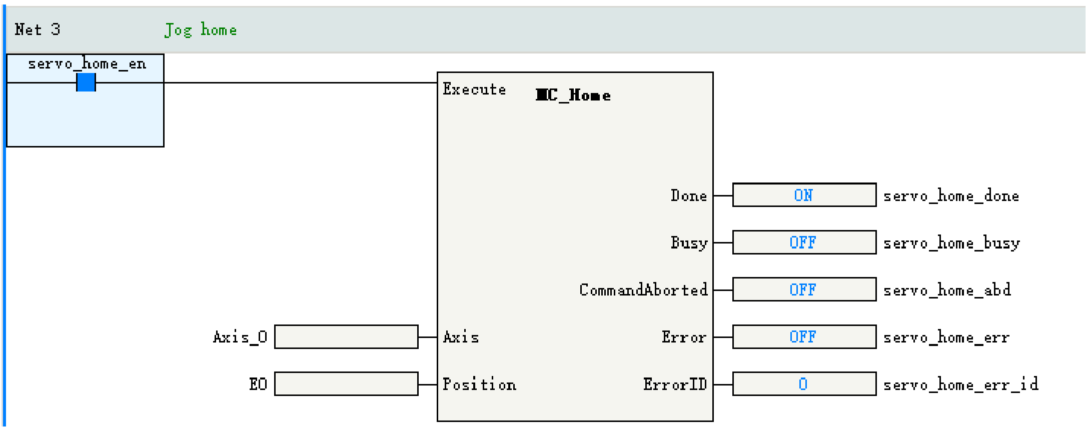
Server error: No Error

3. Homing test of the bus servo axis

- When the servo_home_en is set to ON, the servo_home_busy will automatically set to ON.

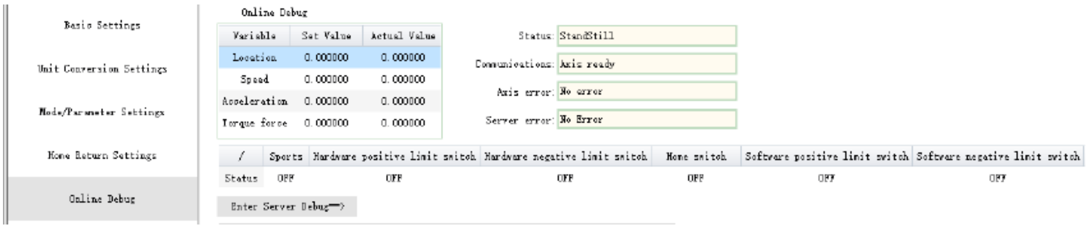


- When the servo motor encounters the Z signal, the homing is automatically completed, in which case the servo_home_done is set to ON, and the servo_home_busy is automatically set to OFF.



4. Relative positioning test of the bus servo axis

- The current position of the bus servo axis is as follows:



Online Debug

Variable	Set Value	Actual Value
Location	0.000000	0.000000
Speed	0.000000	0.000000
Acceleration	0.000000	0.000000
Torque force	0.000000	0.000000

Status: StandStill

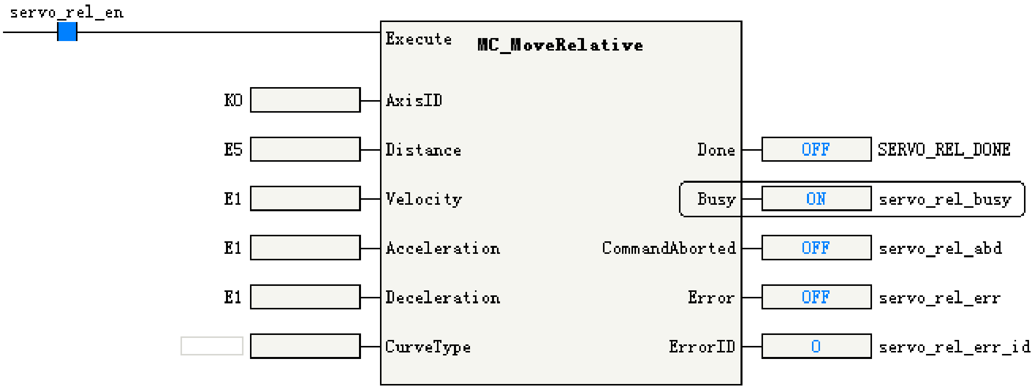
Communications: Axis ready

Axis error: No error

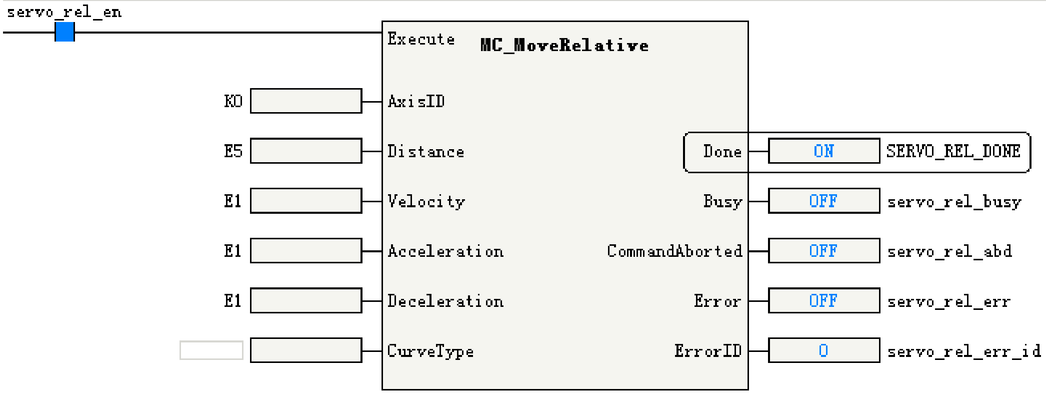
Server error: No Error

/	Sports	Hardware positive limit switch	Hardware negative limit switch	Home switch	Software positive limit switch	Software negative limit switch
Status	OFF	OFF	OFF	OFF	OFF	OFF

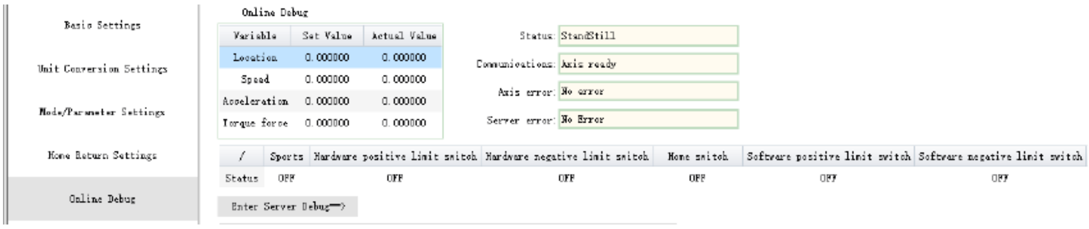
- When the variable servo_rel_en is set to TRUE, the output of variable servo_rel_busy by the function block is TRUE, in which case the bus servo axis starts to run.



- After the positioning is completed, the output of variable servo_rel_done is TRUE.



- Through the online commissioning interface, you can see that the set position of the servo at this time has increased by 5 user units compared with the original.



Online Debug

Variable	Set Value	Actual Value
Location	0.000000	0.000000
Speed	0.000000	0.000000
Acceleration	0.000000	0.000000
Torque force	0.000000	0.000000

Status: StandStill

Communications: Axis ready

Axis error: No error

Server error: No Error

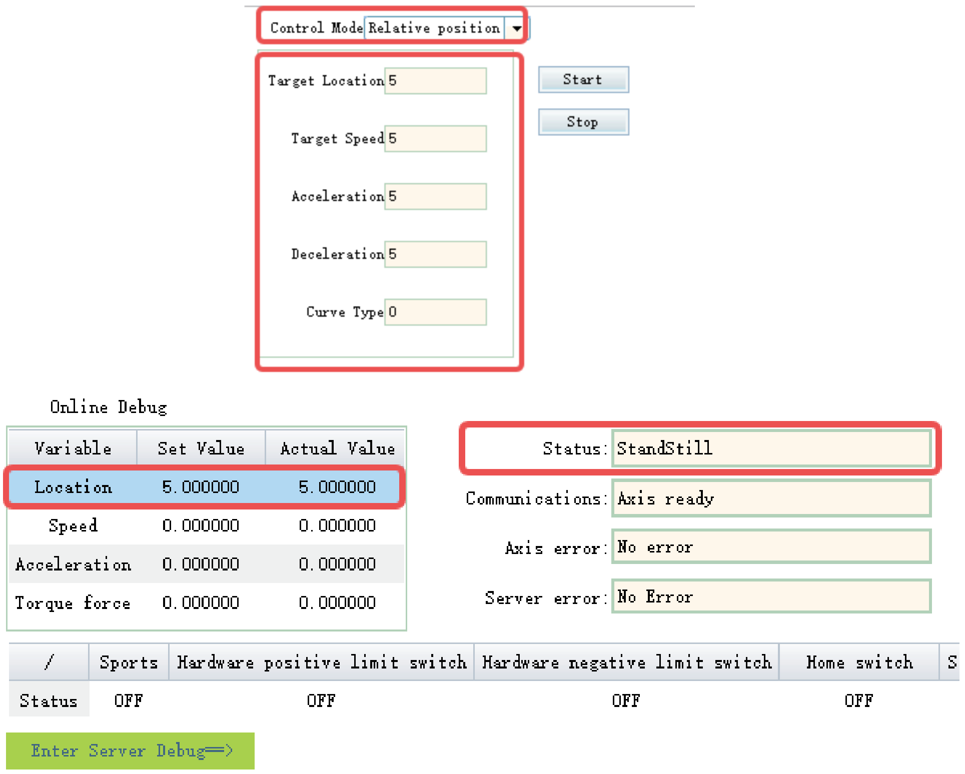
/	Sports	Hardware positive limit switch	Hardware negative limit switch	Home switch	Software positive limit switch	Software negative limit switch
Status	OFF	OFF	OFF	OFF	OFF	OFF

12.2.6.3 Online Commissioning

1. Open the online commissioning interface of the local pulse axis. Click "Enter servo debug" to enter the commissioning mode.
2. Click "Enable" to enable the servo drive.
3. Click "Homing" to start homing. Correctly operate the negative limit variable M1000 to complete the homing action.

   - When M1000 is set to FALSE, homing begins and the axis returns to home through reverse motion.
   - When M1000 is set to TRUE, the axis decelerates to stop after triggering the negative limit, and then moves forward at a low speed.
   - Set M1000 to FALSE and complete homing.

4. Choose "Relative locate mode", set the parameters as follows, and click "Start" to complete the relative positioning test.



Control Mode: Relative position

Target Location: 5

Target Speed: 5

Acceleration: 5

Deceleration: 5

Curve Type: 0

Online Debug

Variable	Set Value	Actual Value
Location	5.000000	5.000000
Speed	0.000000	0.000000
Acceleration	0.000000	0.000000
Torque force	0.000000	0.000000

Status: StandStill

Communications: Axis ready

Axis error: No error

Server error: No Error

/	Sports	Hardware positive limit switch	Hardware negative limit switch	Home switch	S
Status	OFF	OFF	OFF	OFF	

12.3 Configuring Motion Control Axes

12.3.1 Bus Servo Axis versus Local Pulse Axis

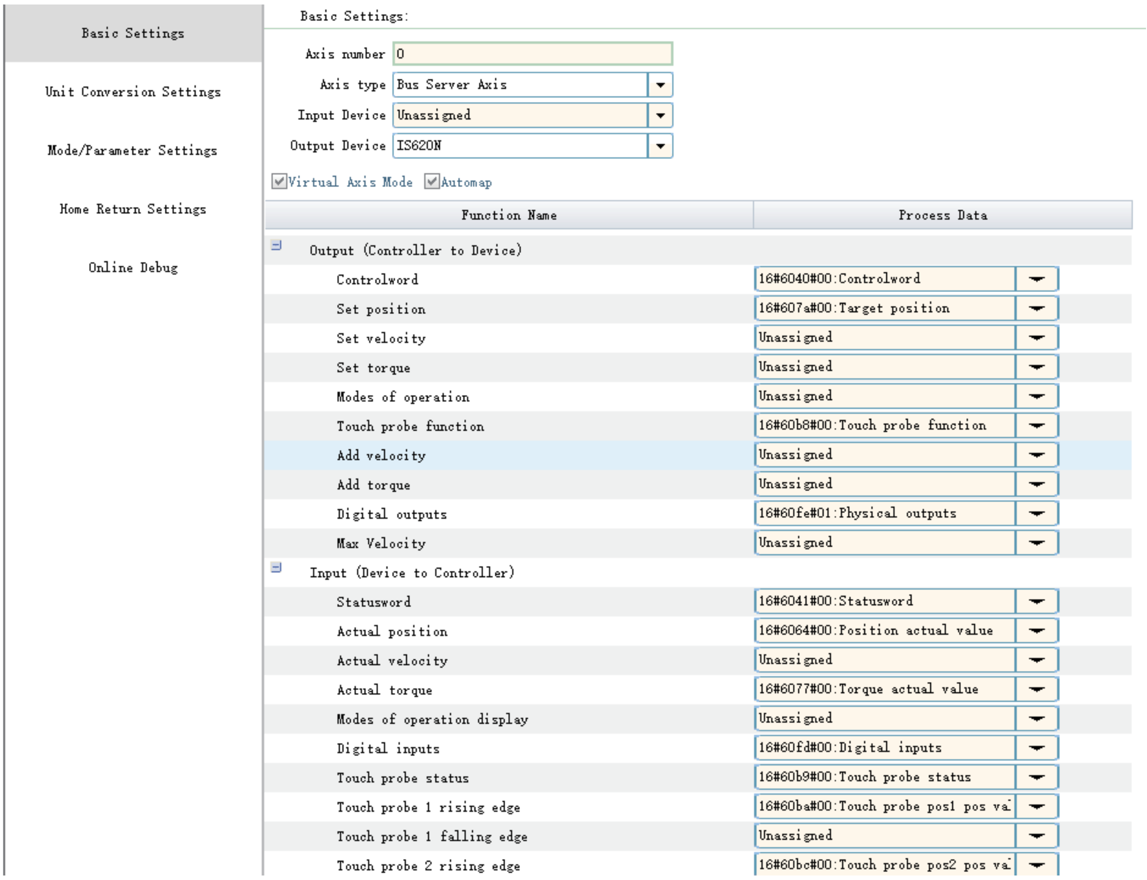
The local pulse output and the EtherCAT drive are controlled with the same set of instructions, and share the same axis structure in design. The main differences between them are listed below.

Item	Local Pulse Output	EtherCAT Bus Drive
Different axis types	A local pulse axis needs to be selected.	A bus servo axis needs to be selected.
Different output devices	Local IO terminals need to be set up, with every two of them forming a group, that is, Y0/Y1, Y2/Y3, Y4/Y5, and Y6/Y7 groups.	The PDO needs to be configured and mapped into the loop variable of the axis.
Pulse output forms	The two supported pulse forms are pulse+direction and CW/CCW, which can be selected in the "Mode/Parameter setting" -> "Output setting".	No setup is required.

Item	Local Pulse Output	EtherCAT Bus Drive
Probe function	Two probes are supported, and each probe terminal can be selected from X0 to X7. This function needs to be selected in the "Mode/Parameter setting" -> "Probe setting".	The probe terminals need to be configured according to the application guide for the EtherCAT drive.
Homing settings	The supported homing modes (except for the Z signal) are specified in the CiA 402 protocol. The limit signal and home signal of the local pulse output axis can be selected through the interface "Set the homing".	The homing modes No 1 to No 35 specified in the CiA 402 protocol are supported in setting, while the limits and home signals need to be set on the drive side.

12.3.2 Basic Settings

The basic settings interface of an axis is used to set the type of the axis, and select a physical drive. The basic settings interface is shown in the following figure.



Basic Settings:

Axis number 0

Axis type Bus Server Axis

Input Device Unassigned

Output Device IS620N

☑ Virtual Axis Mode ☑ Automap

Function Name	Process Data
Output (Controller to Device)	
Controlword	16#6040#00:Controlword
Set position	16#607a#00:Target position
Set velocity	Unassigned
Set torque	Unassigned
Modes of operation	Unassigned
Touch probe function	16#60b8#00:Touch probe function
Add velocity	Unassigned
Add torque	Unassigned
Digital outputs	16#60fe#01:Physical outputs
Max Velocity	Unassigned
Input (Device to Controller)	
Statusword	16#6041#00:Statusword
Actual position	16#6064#00:Position actual value
Actual velocity	Unassigned
Actual torque	16#6077#00:Torque actual value
Modes of operation display	Unassigned
Digital inputs	16#60fd#00:Digital inputs
Touch probe status	16#60b9#00:Touch probe status
Touch probe 1 rising edge	16#60ba#00:Touch probe pos1 pos va
Touch probe 1 falling edge	Unassigned
Touch probe 2 rising edge	16#60bc#00:Touch probe pos2 pos va

- **Axis No**: Each axis is assigned a separate number in the range of 0 to 36, which cannot be modified manually. An axis number can be used as a unique input parameter of the MC instruction to access the axis.
- **Axis type**: Optional axis types are bus servo axis, local pulse axis, bus encoder axis (not supported before version 4.2.0.0), and local encoder axis.
- **Input device**: It is used only for bus encoder axes and local encoder axes.

- **Output device**: It is active only in bus servo axis and local pulse axis modes. In case of a bus servo axis, it is used to select EtherCAT servo drives; in case of a local pulse axis, it is used to select the local high-speed output terminals. Four groups of high-speed output terminals including Y0/Y1, Y2/Y3, Y4/Y5, and Y6/Y7 are available for selection.
- **Virtual axis mode**: It is active only in bus servo axis and local pulse axis mode. When the virtual axis mode is ticked, the axis will no longer control the drive selected by the output device (high-speed output terminal), but will execute motion control instructions internally on a virtual servo axis.
- **Loop variables**: It is active only in bus encoder axis and bus servo axis modes. The EtherCAT slave communicates periodically based on the PDO and an axis is connected to the object dictionary of the EtherCAT slave through a loop variable. When automatic mapping is selected, the mapping process is assigned automatically and cannot be configured manually.

Loop variables of the bus servo axis

The list of variables is as follows. For the detailed meaning of the object dictionary, see the standard CiA 402 protocol.

Table 12–1 List of Variables

Loop Variable	Object Dictionary	Function
Controlword	0x6040	Control word
Set position	0x607a	Corresponding to the target position in servo drive CSP mode
Set velocity	0x60ff	Reserved
Set torque	0x6071	Corresponding to the target torque in CST mode of the servo drive.
Modes of operation	0x6060	Control mode, with a range of setting as follows. 6: Homing mode 8: Cyclic synchronous position (CSP) mode 9: Cyclic synchronous velocity (CSV) mode 10: Cyclic synchronous torque (CST) mode
Touch probe function	0x60b8	Probe control word
Add velocity	Reserved	Reserved
Add torque	Reserved	Reserved
Digital outputs	0x60fe:1	Digital output (DO)
Max Velocity	0x60FF	Maximum velocity
Statusword	0x6041	Status word
Actual position	0x6064	Feedback position
Actual velocity	0x606c	Feedback velocity
Actual torque	0x6077	Feedback torque
Modes of operation display	0x6061	Current control mode
Digital inputs	0x60fd	DI terminal status, with functions as follows: Bit2: Home switch Bit1: Positive limit switch Bit0: Negative limit switch
Touch probe status	0x60b9	Touch probe status
Touch probe 1 rising edge	0x60ba	Probe 1 position on the rising edge
Touch probe 1 falling edge	0x60bb	Probe 1 position on the falling edge

Loop Variable	Object Dictionary	Function
Touch probe 2 rising edge	0x60bc	Touch probe 2 positive edge
Touch probe 2 negative edge	0x60bd	Probe 2 position on the falling edge
Errorcode	0x603f	Drive error code

Parameters that need to be set for unit conversion

Table 12–2 Related Parameters

Parameter	Function
The number of pulses per revolution of the motor/encoder	Set the number of pulses required for the motor to rotate one turn according to the encoder resolution.
Gear change mechanisms in use or not	Specify whether gear change mechanisms are in use or not.
Distance per revolution of the motor/encoder	The workpiece-moving distance per revolution of the motor when no gear change mechanism is in use
Distance per revolution of the worktable	The distance per revolution on the workpiece side when gear change mechanisms are in use
Gear ratio on the workpiece side	Set a gear ratio on the workpiece side.
Gear ratio on the motor side	Set a gear ratio on the motor side.

The bus drives (local pulse axes) use pulse units when controlling a motor, and use common measurement units for motion control instructions, such as millimeters, degrees, and inches, which are called user units (Unit). According to the configuration parameters, the two units are converted to each other within an axis. The conversion between the two units is divided into the following modes.

- With gear change mechanisms

  When the gear change mechanisms are not in use, the conversion equation from user unit to pulse unit is as follows.

$$\text{Number of pulses (unit: pulse)} = \frac{\text{Number of pulses per revolution of the motor/encoder [DINT]}}{\text{Number of pulses per revolution of the motor/encoder [REAL]}} \text{ x Distance (unit: Unit)}$$

  Take the Inovance 20-bit encoder as an example. The set parameters are as follows.

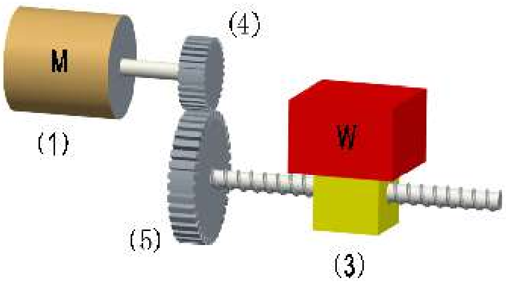
  Number of pulses per revolution of the motor/encoder = 1048576

  Distance per revolution of the motor/encoder = 1

  Then, when the target displacement given by the relative positioning instruction is 10, the actual number of pulses sent by the motion control axis is 10485760, and the motor rotates by 10 revolutions.

- With gear change mechanisms

  Typical working condition in linear mode is shown in the following figure.

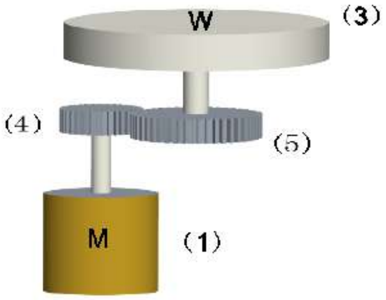


In the figure, (1) is the servo motor, (3) is the workpiece, (4) is the gear ratio denominator, and (5) is the gear ratio numerator.

The calculation equation from user unit to pulse unit is as follows.

$$\text{Number of pulses (unit: pulse)} = \frac{\text{Number of pulses per revolution of the motor/encoder [DINT] x Gear ratio numerator [DINT]}}{\text{Distance per revolution of the workbench [REAL] x Gear ratio denominator [DINT]}} \text{ x Distance (unit: Unit)}$$

Typical working condition in ring mode is shown in the following figure.



In the figure, (1) is the servo motor, (3) is the workpiece, (4) is the gear ratio numerator, and (5) is the gear ratio denominator.

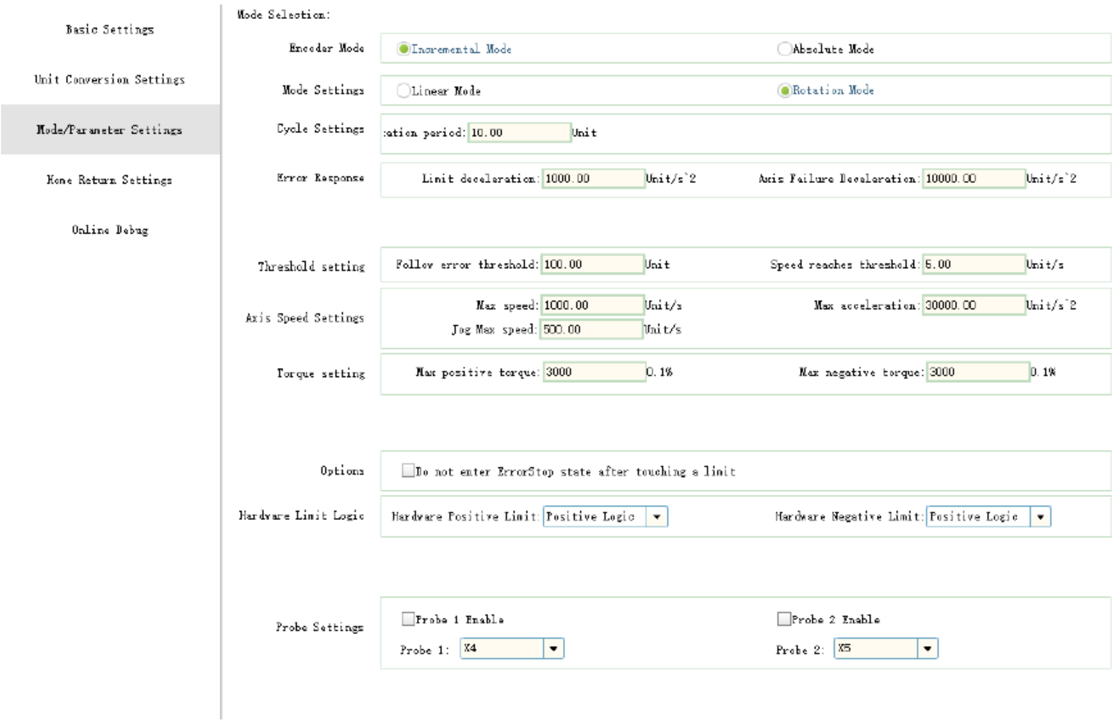
The calculation equation from user unit to pulse unit is as follows.

$$\text{Number of pulses (unit: pulse)} = \frac{\text{Number of pulses per revolution of the motor/encoder [DINT] x Gear ratio numerator [DINT]}}{\text{Distance per revolution of the workbench [REAL] x Gear ratio denominator [DINT]}} \text{ x Distance (unit: Unit)}$$

12.3.3 Mode/Parameter Settings

12.3.3.1 Configuration Interface

The following figure shows the "Basic setting" page. The parameter list varies with the selected axis type.



Mode Selection:

Encoder Mode: ● Incremental Mode ○ Absolute Mode

Mode Settings: ○ Linear Mode ● Rotation Mode

Cycle Settings: :ation period: 10.00 Unit

Error Response: Limit deceleration: 1000.00 Unit/s^2 Axis Failure Deceleration: 10000.00 Unit/s^2

Threshold setting: Follow error threshold: 100.00 Unit Speed reaches threshold: 5.00 Unit/s

Axis Speed Settings: Max speed: 1000.00 Unit/s Max acceleration: 30000.00 Unit/s^2 Jog Max speed: 500.00 Unit/s

Torque setting: Max positive torque: 3000 0.1% Max negative torque: 3000 0.1%

Options: □ Do not enter ErrorStop state after touching a limit

Hardware Limit Logic: Hardware Positive Limit: Positive Logic Hardware Negative Limit: Positive Logic

Probe Settings: □ Probe 1 Enable □ Probe 2 Enable Probe 1: X4 Probe 2: X5

12.3.3.2 Encoder Mode

The encoder mode is only valid in the bus servo axis mode and is used with incremental encoder servo drives and absolute encoder servos. Choose the mode according to the type of servo drive actually used. The processing on the PLC side is as follows.

Incremental mode

The increase in the number of revolutions caused by the overflow of the 32-bit counter of servo drive encoder is not taken into account on the PLC side. The PLC does not save the current position of the encoder when the power is turned off and on again. After the second power-on, the current position of the axis is calculated only according to the position of a single revolution given by the servo drive.

Absolute mode

The increase in the number of revolutions caused by the overflow of the 32-bit counter of servo drive encoder is taken into account on the PLC side. The PLC saves the current position given the encoder when the power is turned off and on again. After the second power-on, the encoder position of the axis saved inside the PLC and the position given the servo drive are read during the initialization to calculate the current absolute position of the axis.

Note

If an Inovance servo is used and process parameters 200B.3Bh (low-order 32 bits of mechanical absolute position) and 200B.3Dh (high-order 32 bits of mechanical absolute position) are configured, the axis calculates its current absolute position during initialization. The calculation is based on the 64-bit mechanical absolute position feedback from the servo drive and the position offset saved in the PLC. This can effectively prevent data overflow in the 32-bit counter.

12.3.3.3 Mode Setting

Set the motion control axis to linear mode or ring mode based on actual working conditions.

Linear mode

Linear mode is usually used for devices that move within the range of mechanical actions in the X-Y linear coordinate system.

Linear mode typically involves a zero point.

An increase in the feedback position during motion indicates a forward motion, while a decrease indicates a reverse motion.

Forward and reverse software limits can be set. After the software limit is enabled, an axis can only move within the limit range.

Absolute positioning mode: When the target position is greater than the start position, the axis moves forward over a distance equal to the result of the target position minus the start position. When the target position is less than the start position, the axis moves reversely over a distance equal to the result of the start position minus the target position.

Relative positioning mode: When the target displacement is greater than 0, the axis moves forward over a distance equal to the target displacement. When the target displacement is less than 0, the axis moves reversely over a distance equal to the target displacement.

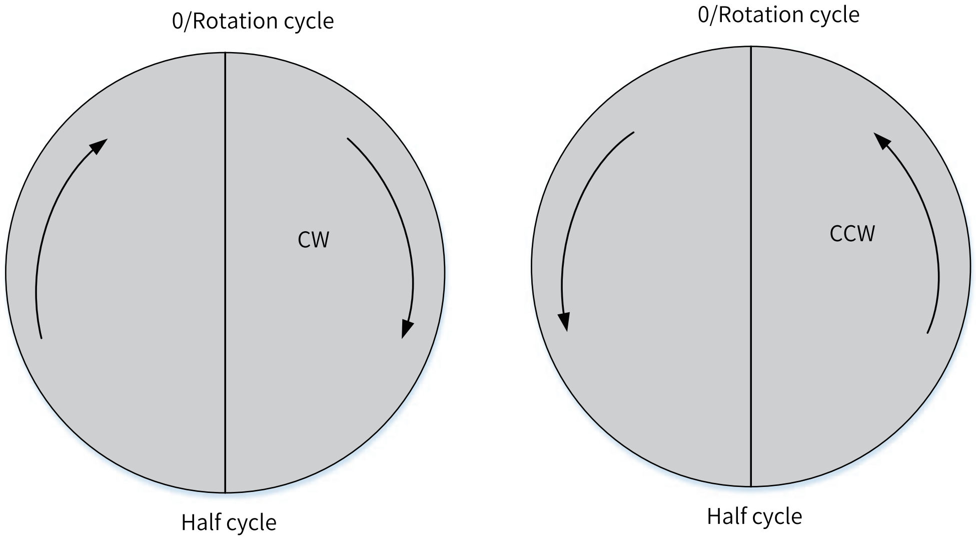
Processing mode of velocity instructions in linear mode: If the target velocity is greater than 0, a forward motion is performed. If the target velocity is less than 0, a reverse motion is performed.

Ring mode

The ring mode is a mode in the form of a ring counter that is capable of infinitely repeated counting within a set range. It is usually used in revolving stages or rolls.

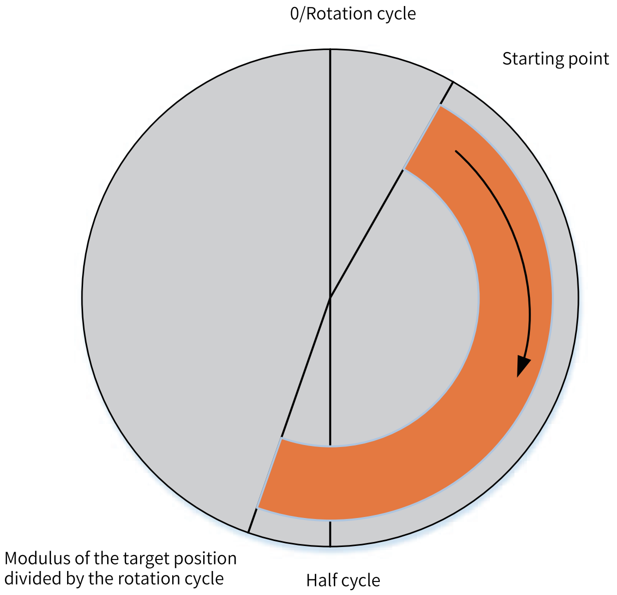
The ring mode usually involves a zero point and a rotation cycle. The feedback position of a ring counter is greater than or equal to 0 and less than one rotation cycle.

In ring mode, an increase in feedback position indicates a CW motion, and a decrease in feedback position indicates a CCW motion.

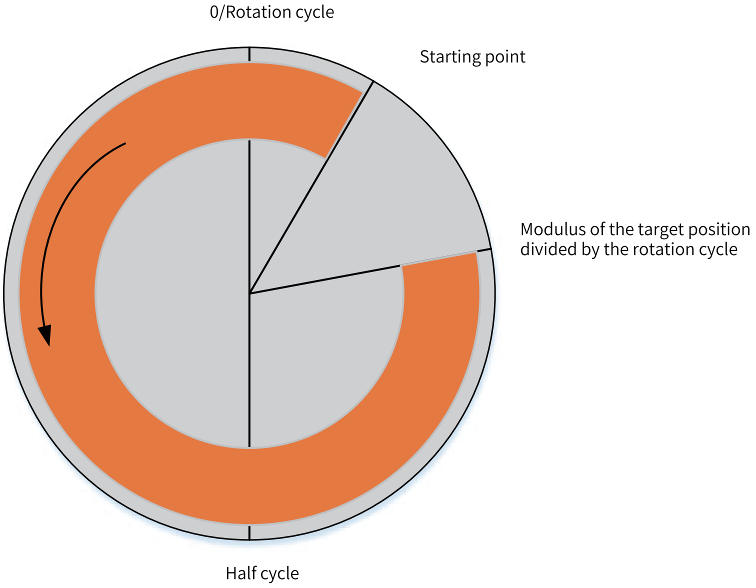


Software limits are not applicable in ring mode.

- Processing mode of relative positioning: When the target displacement is greater than 0, the axis moves clockwise over a distance equal to the target displacement. When the target displacement is less than 0, the axis moves counter-clockwise over a distance equal to the target displacement.
- Processing mode of absolute positioning:
  Forward: First, take the modulus of the target position divided by the rotation cycle. Then, move the axis in a CW manner from the start position to the target position.


0/Rotation cycle
Starting point
Modulus of the target position divided by the rotation cycle
Half cycle

Reverse: First, take the modulus of the target position divided by the rotation cycle. Then, move the axis from the start position to the target position in a CCW manner.


0/Rotation cycle
Starting point
Modulus of the target position divided by the rotation cycle
Half cycle

The shortest distance: First, take the modulus of the target position divided by the rotation cycle to obtain the target position. Then, calculate the displacement from the start point in a CW manner to the target position. If the displacement is less than or equal to half cycle, a CW motion is performed; otherwise, a CCW motion is performed to the target position.

Current direction: A motion is performed to the target position in the latest axis motion direction, or a forward motion is performed to the target position in case of initial power-on.

Processing mode of velocity instructions in ring mode: If the target velocity is greater than 0, a CW motion is performed; if the target velocity is less than 0, a CCW motion is performed.

12.3.3.4 Software Limits

Software limits can be set in linear mode.
When software limits are valid, the system constantly detects the absolute position of an axis that decelerates from the current velocity to 0 at the set limit deceleration rate according to T-type deceleration mode. If the absolute position of the axis is beyond the limit range, the axis will execute the software limit deceleration algorithm and the positioning or velocity instruction being executed will be aborted.

Software limits are invalid in homing or torque mode.

12.3.3.5 Deceleration upon Axis Fault

During the operation of an axis, if the axis must switch to the ErrorStop state due to a logic failure of the motion instruction itself, the axis will do T-type deceleration according to the setting of deceleration upon axis fault. The axis will not enter the ErrorStop state until it decelerates to 0.

12.3.3.6 Following Error

During the execution of positioning and velocity instructions, the servo drive actually works in cyclic synchronous position (CSP) mode, and the planning of the position curve is done on the PLC. The PLC sends the target position to the servo drive through 0x607A, the servo drive drives the servo motor to move, and the position of the motor encoder is sent back to the PLC through 0x6064. Due to the inherent features of the servo drive and the motor, a difference between 0x607A and 0x6064 is generated. This difference is called the following error when converted to user units. You can set a following error limit. If the absolute value of the following error of an axis exceeds the limit, the axis reports an excessive following error and enters the ErrorStop state.

12.3.3.7 Axis Speed Setting

You can set three parameters: maximum velocity, maximum acceleration, and maximum jogging velocity. When a parameter, such as the target velocity, acceleration, and deceleration, in the positioning instruction or velocity instruction exceeds the velocity limit, the relevant instruction reports an error and the axis enters the ErrorStop state.

For the bus servo axis, the maximum velocity is converted into pulse units and written into the object dictionary 0x607f of the servo drive in the form of startup parameters.

12.3.3.8 Torque Setting

The torque setting is only applicable to the bus servo axis.

If the target torque exceeds the maximum torque in the torque instruction, the instruction reports an error and the axis enters the ErrorStop state.

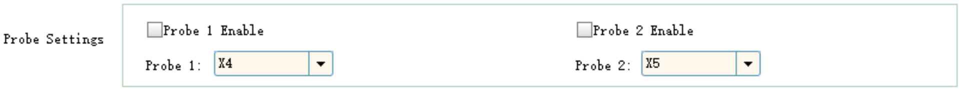
The forward torque limit will be written into the object dictionary 0x60e0 of the servo drive in the form of startup parameters, and the reverse torque limit will be written into the object dictionary 0x60e1 of the servo drive in the form of startup parameters.

12.3.3.9 Probe Setting

Probe terminals can be enabled for local pulse axes through probe setting.

Each local pulse axis can be configured with up to two probe terminals. The probe terminals can be selected from X0 to X7.

When a probe terminal is enabled, the local pulse axis can use probe instructions and interrupt positioning instructions.



Probe Settings

□Probe 1 Enable

Probe 1: X4

□Probe 2 Enable

Probe 2: X5

12.3.3.10 Output Setting

You can select Y0/Y1, Y2/Y3, Y4/Y5, and Y6/Y7 as four local pulse axes.

The output of local pulse axes can be set as pulses in the format of pulse+direction or CW/CCW.

For channels that are set as pulse axes, when pulse+direction is selected, Y0, Y2, Y4, and Y6 are the pulse terminals, and Y1, Y3, Y5, and Y7 are the 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.

12.3.3.11 Not Entering ErrorStop State upon a Limit Activation

This function is only available for AutoShop 4.0.0.0 matching PCB software of version 3.0.0.0 or later.

- When this option is selected, the motion mode is synchronized, and only the axis control instruction reports an error when the motion control axis reaches the limit, but the motion control axis does not enter the ErrorStop state.
- When this option is not selected, if the motion control axis reaches the limit after a motion control instruction other than MC_Jog is called, the motion control axis enters the ErrorStop state, and the instruction reports an error.

12.3.3.12 Hardware Limit Logic

This function is only available for AutoShop 4.0.0.0 matching PCB software of version 3.0.0.0 or later.

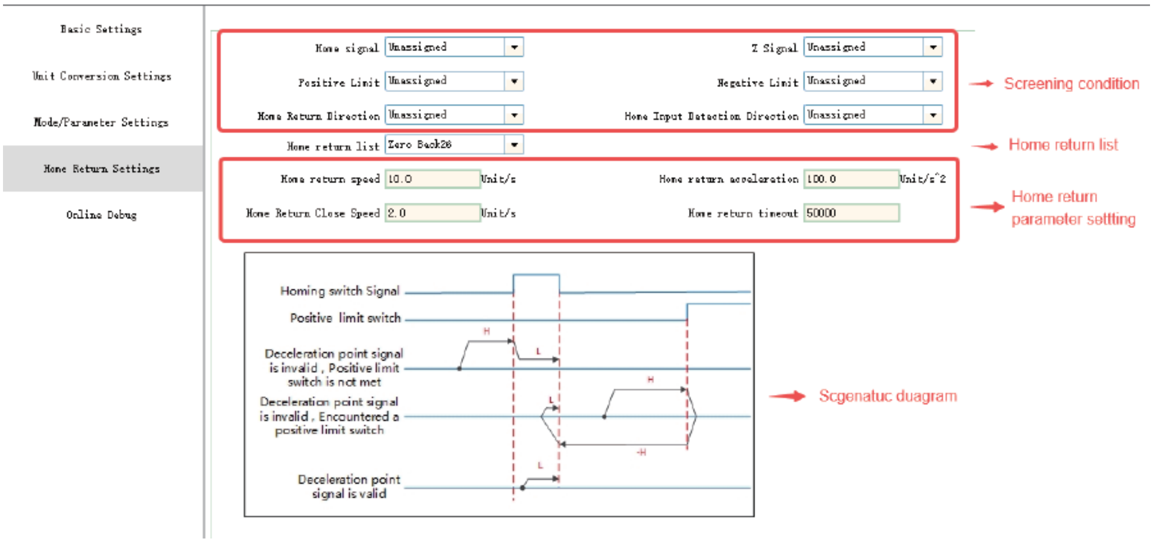
Among axis system variables, bphlimit and bnhlimit indicate the hardware positive limit and hardware negative limit, respectively. When the hardware limit logic is positive, they correspond to the values of bit1 and bit0 in the object dictionary 0x60fd, respectively. When the hardware limit logic is negative, they are inverted values of bit1 and bit0 in the object dictionary 0x60fd, respectively.

The hardware limit logic setting is only reflected in the above variables and has no impact on the limit stop processing when the servo reaches the limit.

12.3.4 Homing

The homing modes No. 1 to No. 35 specified in the CiA 402 protocol are supported.

The homing settings interface is as follows.


Basic Settings
Unit Conversion Settings
Mode/Parameter Settings
Home Return Settings
Online Debug
Screening condition
Home return list
Home return parameter settting
Homing switch Signal
Positive limit switch
Deceleration point signal is invalid , Positive limit switch is not met
Deceleration point signal is invalid , Encountered a positive limit switch
Deceleration point signal is valid
H
L
-H
Scgenatuc duagram

Home signal: Unassigned

Z Signal: Unassigned

Positive Limit: Unassigned

Negative Limit: Unassigned

Home Return Direction: Unassigned

Home Input Detection Direction: Unassigned

Home return list: Zero Back26

Home return speed: 10.0 Unit/s

Home return acceleration: 100.0 Unit/s^2

Home Return Close Speed: 2.0 Unit/s

Home return timeout: 50000

The parameters in the configuration interface are as follows.

Parameter	Description
Home signal	It is used to choose whether to use a home signal. When "Unassigned" is selected, it is not used as a mandatory filter. When "Do not use" is selected, the homing modes that must use a home signal are removed. When "Use" is selected, the homing modes that do not support a home signal are removed.
Negative limit	It is used to select whether to use a hardware left limit signal. When "Unassigned" is selected, it is not used as a mandatory filter. When "Do not use" is selected, the homing modes that must use a negative limit signal are removed. When "Use" is selected, the homing modes that do not support a negative limit signal are removed.
Positive limit	It is used to choose whether to use a hardware right limit signal. When "Unassigned" is selected, it is not used as a mandatory filter. When "Do not use" is selected, the homing modes that must use a positive limit signal are removed. When "Use" is selected, the homing modes that do not support a negative limit signal are removed.
Z signal	It is used to choose whether to use a motor Z signal. When "Unassigned" is selected, it is not used as a mandatory filter. When "Do not use" is selected, the homing modes that must use a z signal are removed. When "Use" is selected, the homing modes that do not support a Z signal are removed.

Parameter	Description
Homing direction	It is used to set the direction of motion at the beginning of homing. Forward: The direction of motion is forward when the limit (home) signal input is inactive, otherwise it is the opposite. Reverse: The direction of motion is reverse when the limit (home) signal input is inactive, otherwise it is the opposite.
Homing detection direction	The direction of motion when the home signal is reached Forward: stopping at the edge of the home signal during forward motion. Reverse: stopping at the edge of the home signal during the reverse motion.
Homing list	The homing modes, with a setting range of 1 to 35, are written into the object dictionary 0x6098 in the form of startup parameters.
Homing mode	It is used to set the relative mode or absolute mode in homing mode No. 35, which are written into the object dictionary 0x60e6 in the form of startup parameters.
Homing velocity	The homing velocity is written into the sub-index No. 1 of the object dictionary 0x6099 in the form of startup parameters after the user units are converted into pulse units.
Homing closing velocity	The homing closing velocity is written into the sub-index No. 2 of the object dictionary 0x6099 in the form of startup parameters after the user units are converted into pulse units.
Homing acceleration	The homing acceleration is written into the object dictionary 0x609A in the form of startup parameters after the user units are converted into pulse units.
Homing timeout time	This parameter is exclusive to Inovance drives.

In actual use, the homing mode is defined by several parameters, such as home signal, positive limit, negative limit, Z signal, homing direction, and homing detection direction, and then selected from the options of the homing list as needed.

It is worth noting that multiple homing modes are still left after the conditions for homing are filtered, in which case you can choose an appropriate homing mode from the homing list. For example, if the parameters are set as in the following table, you can filter out two homing modes.

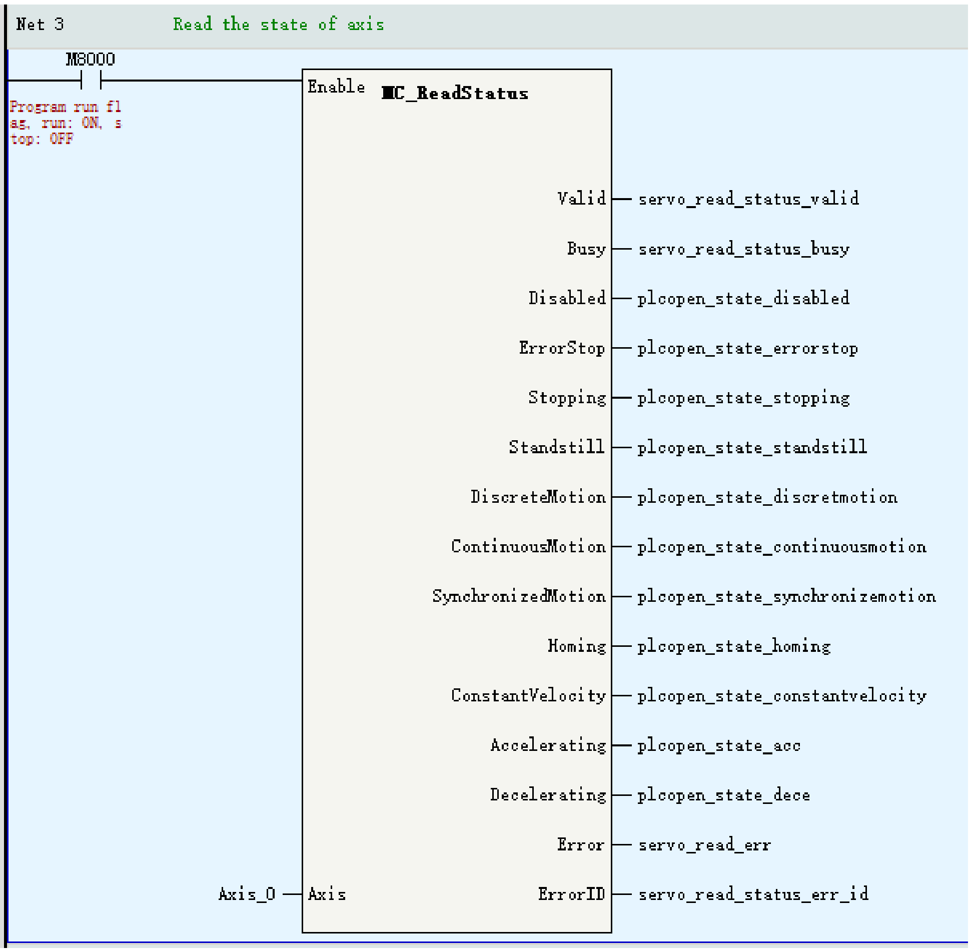
Signal	Value
Home signal	"Use"
Negative limit	"Do not use"
Positive limit	"Use"
Z signal	"Use"
Homing direction	"Forward"
Homing detection direction	"Forward"

According to the above settings, the two homing modes No. 8 and No. 10 can be filtered out, and the difference between the two is whether the home input signal is active when the homing is completed.

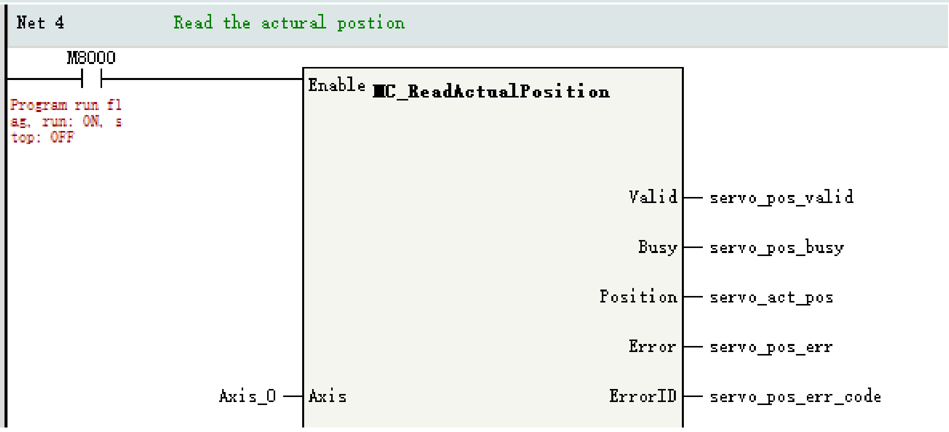
12.4 Online Monitoring

You can obtain the status of an axis through PLC instructions, system variables of the axis, and the online commissioning interface in the background.

1. Obtaining the axis status through instructions
   You can obtain the status of an axis through status instructions such as MC_ReadStatus and MC_ReadPosition.



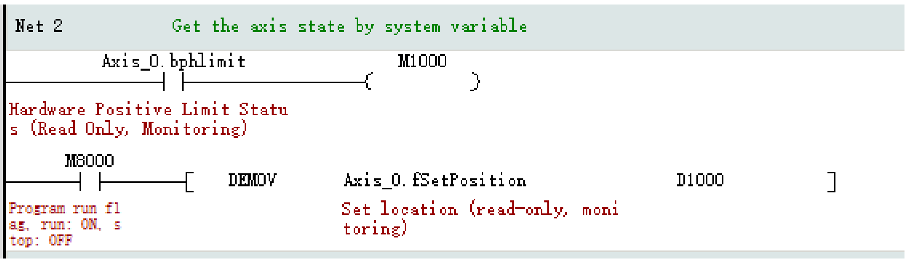
```
Net 3          Read the state of axis
M8000
Program run flag, run: ON, stop: OFF

Enable  MC_ReadStatus
                      Valid — servo_read_status_valid
                       Busy — servo_read_status_busy
                   Disabled — plcopen_state_disabled
                  ErrorStop — plcopen_state_errorstop
                   Stopping — plcopen_state_stopping
                 Standstill — plcopen_state_standstill
             DiscreteMotion — plcopen_state_discretmotion
           ContinuousMotion — plcopen_state_continuousmotion
         SynchronizedMotion — plcopen_state_synchronizemotion
                     Homing — plcopen_state_homing
           ConstantVelocity — plcopen_state_constantvelocity
               Accelerating — plcopen_state_acc
               Decelerating — plcopen_state_dece
                      Error — servo_read_err
Axis_0 — Axis       ErrorID — servo_read_status_err_id
```



```
Net 4          Read the actural postion
M8000
Program run flag, run: ON, stop: OFF

Enable  MC_ReadActualPosition
                      Valid — servo_pos_valid
                       Busy — servo_pos_busy
                   Position — servo_act_pos
                      Error — servo_pos_err
Axis_0 — Axis       ErrorID — servo_pos_err_code
```

2. Obtaining the status of an axis through system variables

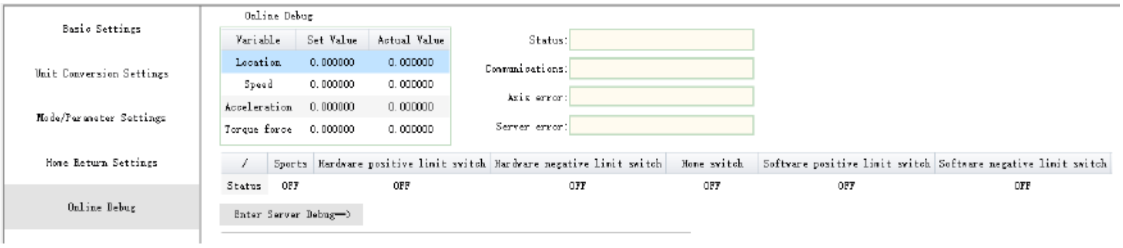
With system variables, you can view the motion status of each axis in real time.



```
Net 2          Get the axis state by system variable
Axis_0.bphlimit                    M1000
Hardware Positive Limit Status (Read Only, Monitoring)

M8000
          [ DEMOV   Axis_0.fSetPosition            D1000          ]
Program run flag, run: ON, stop: OFF
                    Set location (read-only, monitoring)
```

3. Obtaining the axis status through the online commissioning interface

Through the online commissioning interface, you can view the motion status of each axis in real time. The online monitoring interface of an axis is shown in the following figure.



Online Debug

Variable	Set Value	Actual Value
Location	0.000000	0.000000
Speed	0.000000	0.000000
Acceleration	0.000000	0.000000
Torque force	0.000000	0.000000

Status:

Communications:

Axis error:

Server error:

/	Sports	Hardware positive limit switch	Hardware negative limit switch	Home switch	Software positive limit switch	Software negative limit switch
Status	OFF	OFF	OFF	OFF	OFF	OFF

The following table shows the data that can be monitored.

Object	System Variable	Function
Position reference	fSetPosition	Target position (user unit) when PLC performs path planning
Velocity reference	fSetVelocity	Target velocity (user unit) when PLC performs path planning
Acceleration reference	fSet_Acc_Dec	Target acceleration (user unit) when PLC performs path planning
Torque reference	fSetTorque	Target torque when PLC performs torque planning (%)
Actual position	fActPosition	Current position of drive feedback (user unit)
Actual velocity	fActVelocity	Velocity calculated from actual position (user unit)
Actual acceleration	fAct_Acc_Dec	Acceleration calculated from actual velocity (user unit)
Actual torque	fActTorque	Actual torque of drive feedback (%)
Status	wPLCOpenState	The status of the PLCOpen state machine: 0: PowerOff 1: ErrorStop 2: Stopping 3: StandStill 4: DiscreteMotion 5: ContinuousMotion 7: Homing 8: SynchronizedMotion
Communication	wConfigState	The status of data communication between the motion control axis and the drive 0: Init (axis in the initialization state) 1: Configure finish (configuration reading completed) 2: Sync finish (synchronized with EtherCAT tasks) 3: Wait communication (communication with the servo drive established) 4: Slave ready (initialization completed for the servo drive controlled by axes) 5: Axis ready (communication established)
Axis error	wAxisError	The internal error of a motion control axis
Servo error	wServoError	For a local pulse axis, it is the error code of the axis; for a bus servo axis, it is the corresponding 0x603F value. See the guide for the drive used.
Motion	bMotionState	Indicates whether a motion control axis is currently in motion.

Object	System Variable	Function
Hardware positive limit	bphlimit	Indicates whether the hardware positive limit input for a motion control axis is valid.
Hardware negative limit	bnhlimit	Indicates whether the hardware negative limit input for a motion control axis is valid.
Home switch	bhomestate	Indicates whether the home switch input of a motion control axis is valid.
Software positive limit	bpslimit	Indicates whether a motion control axis has reached the software positive limit.
Software negative limit	bnslimit	Indicates whether a motion control axis has reached the software negative limit.

12.5 Axis Control Functions

12.5.1 Overview

Basic servo control can be realized through the online commissioning function, such as enable, stop, jogging, and point-to-point control. After the basic actions are confirmed to be normal, complex logic control can be realized through motion control instructions. Online commissioning and PLC instruction control cannot be used at the same time. The restrictions are as follows:

- When the MC_Stop instruction is called to make an axis enter the Stopping state, the axis cannot enter the online commissioning mode through the background.
- The MC_Power instruction and the enable function in online commissioning are in a OR relationship, that is, when either of them is valid, the axis can be enabled.
- Motion instructions, such as MC_MoveAbsolute, that have a lower priority than the online commissioning are invalid when an axis is in the online commissioning mode. In this case, the instructions will report an error if called, but the axis will not enter the fault state.

12.5.2 Online Commissioning

The functions that can be achieved by online commissioning are as follows:

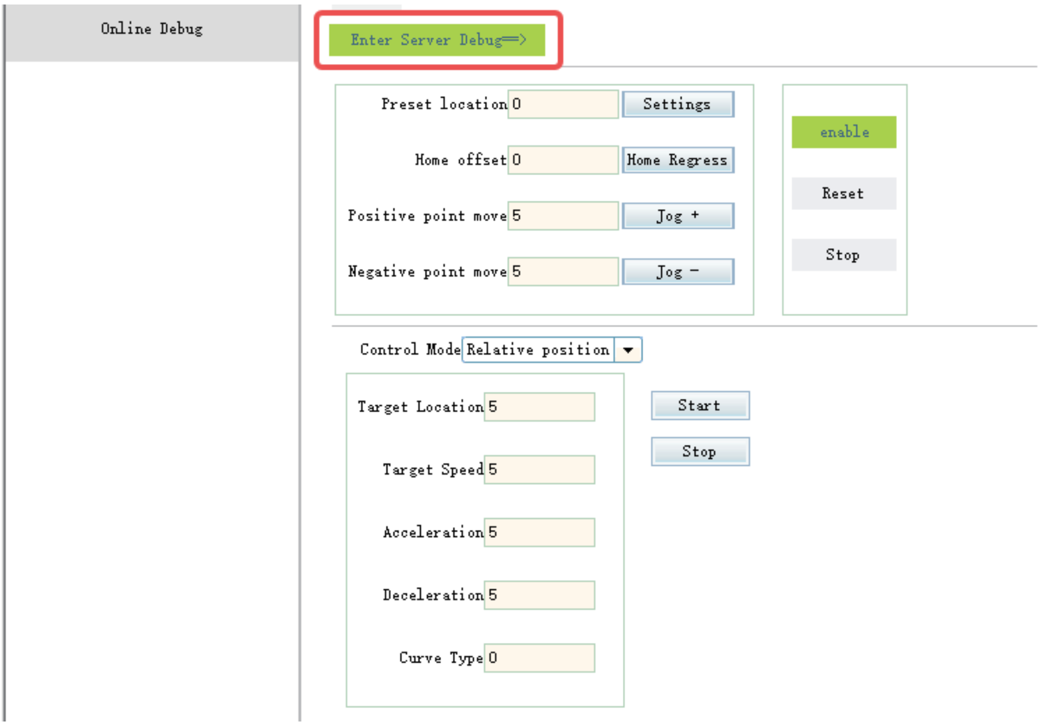
Function	System Variable	Description
Entering the online commissioning mode	bEnterDebug	Makes an axis enter the online commissioning mode, after which the PLC will not continue to execute motion control instructions.
Enable	bPowerOn	Similar to the way to call the MC_Power instruction to put an axis in an enabled or disabled state.
Reset	bReset	Resets a fault that occurs on an axis. This instruction is equivalent to the MC_Reset instruction.
Stop	bStop	Stops the motion of an axis. This instruction is equivalent to the MC_Stop instruction.
Homing	bHome	Performs a homing action. This instruction is equivalent to the MC_Home instruction.
Setting the current position	bSetPos	Sets the current position of an axis. This instruction is equivalent to the MC_SetPosition instruction.
Jog	bJogP/bJogN	Implements the jog function. This instruction is equivalent to the MC_Jog instruction.

Function	System Variable	Description
Motion Type	bDebugMotionType	Selects a motion mode from the following options: 1: Absolute positioning 2: Relative positioning 3: Continuous motion 5: Reciprocating motion 6: Torque mode
Relative positioning	bAbsPos	Implements the relative positioning. This instruction is equivalent to the MC_MoveRelative instruction.
Absolute positioning	bRelPos	Implements the absolute positioning. This instruction is equivalent to the MC_MoveAbsolute instruction.
Continuous motion	bVelocity	Implements continuous motion at a certain velocity. This instruction is equivalent to the MC_MoveVelocity instruction.
Reciprocating motion	bRevPos	Implements the reciprocating motion between two absolute positions. This instruction is equivalent to two MC_MoveAbsolute instructions in a loop.
Torque control	bTorque	Implements the torque control. This instruction is equivalent to the MC_ MoveTorque instruction.

Operation steps for online commissioning

1. Entering the online commissioning mode

   As shown in the following figure, click "Enter online debug" to put an axis in the online commissioning mode.



Preset location 0

Home offset 0

Positive point move 5

Negative point move 5

Control Mode Relative position

Target Location 5

Target Speed 5

Acceleration 5

Deceleration 5

Curve Type 0

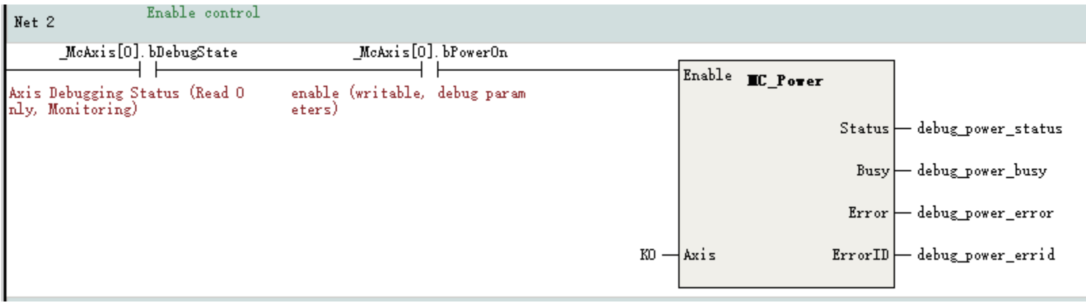
After receiving the instruction to enter the online commissioning mode, the background will do the following checks:

- If the current axis is in the Stopping state, the axis cannot enter online commissioning mode.

- If the current axis is already in motion, the user will be asked whether to enter the online commissioning mode. If the axis is forced to enter the online commissioning mode, the original motion state will be interrupted and the axis motion will stop.
- If the axis is already enabled when entering the online commissioning mode, the axis remains enabled after entering the mode.

2. Basic operations

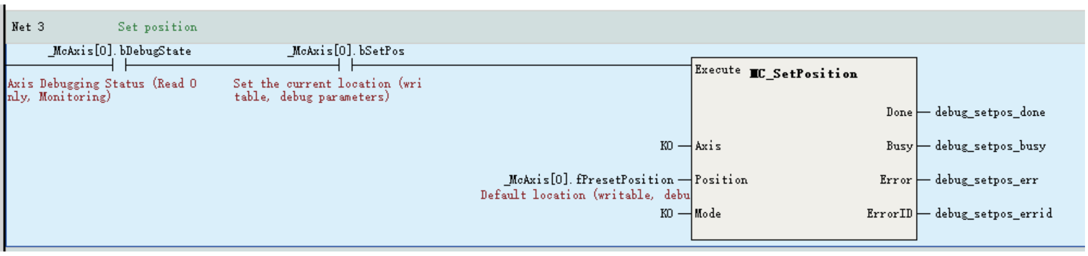
- Enable: After an axis enters the online commissioning mode, click "Enable" to make the axis enter the enabled state, and the execution effect is equivalent to the following instructions.


Net 2
Enable control
_McAxis[0].bDebugState
Axis Debugging Status (Read Only, Monitoring)
_McAxis[0].bPowerOn
enable (writable, debug parameters)
Enable
MC_Power
Status — debug_power_status
Busy — debug_power_busy
Error — debug_power_error
K0 — Axis
ErrorID — debug_power_errid

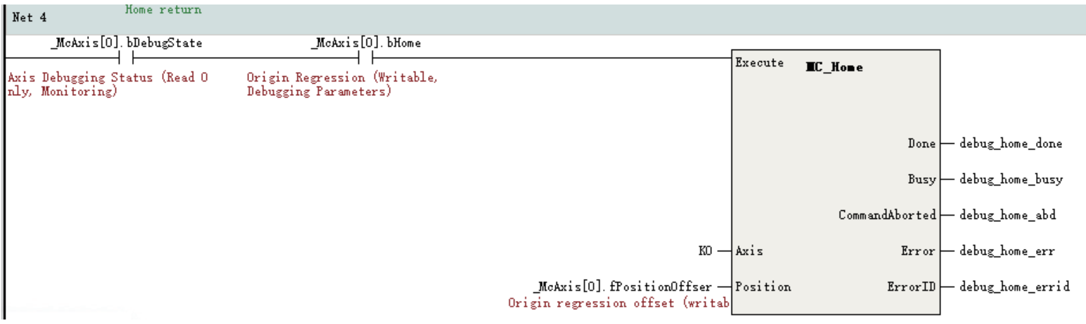
Note

The enable-control in online commissioning and the MC_Power instruction called in the PLC program jointly control the enable-state of the axis. When either of the inputs is active, the axis is enabled.

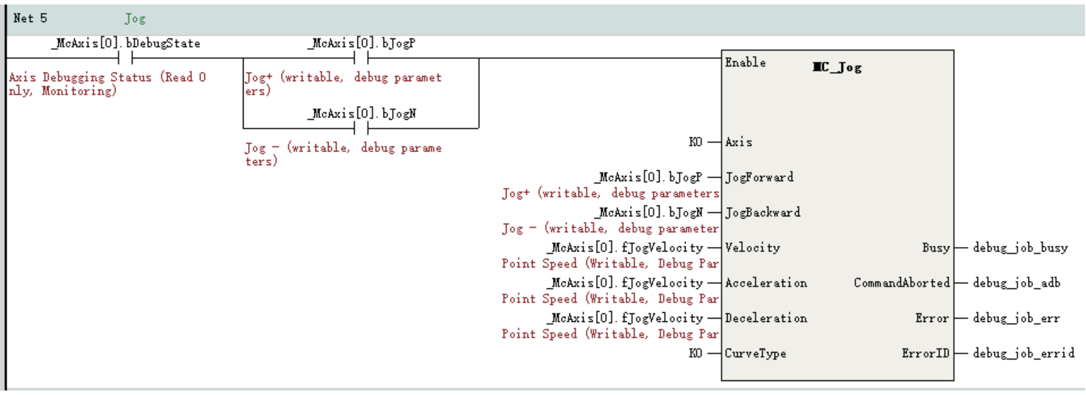
- Preset position: When an axis is in non-motion mode, click "Settings" to write the value in the preset position field to the axis. The execution logic is as follows:


Net 3
Set position
_McAxis[0].bDebugState
Axis Debugging Status (Read Only, Monitoring)
_McAxis[0].bSetPos
Set the current location (writable, debug parameters)
Execute
MC_SetPosition
Done — debug_setpos_done
K0 — Axis
Busy — debug_setpos_busy
_McAxis[0].fPresetPosition — Position
Default location (writable, debu
Error — debug_setpos_err
K0 — Mode
ErrorID — debug_setpos_errid

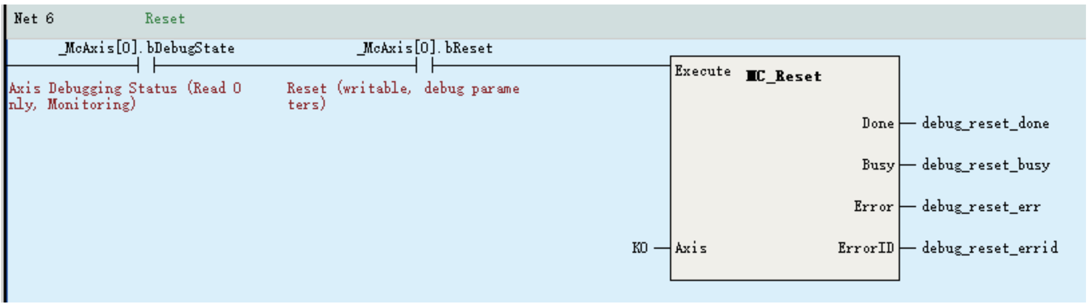
- Homing control: When an axis is in the Standstill state, click "Homing", the axis will control the servo drive to perform the homing operation. When the homing is completed, the value set in the "Home offset" field will be written into the servo drive. The execution logic is as follows:


Net 4
Home return
_McAxis[0].bDebugState
Axis Debugging Status (Read Only, Monitoring)
_McAxis[0].bHome
Origin Regression (Writable, Debugging Parameters)
Execute
MC_Home
Done — debug_home_done
Busy — debug_home_busy
CommandAborted — debug_home_abd
K0 — Axis
Error — debug_home_err
_McAxis[0].fPositionOffser — Position
Origin regression offset (writab
ErrorID — debug_home_errid

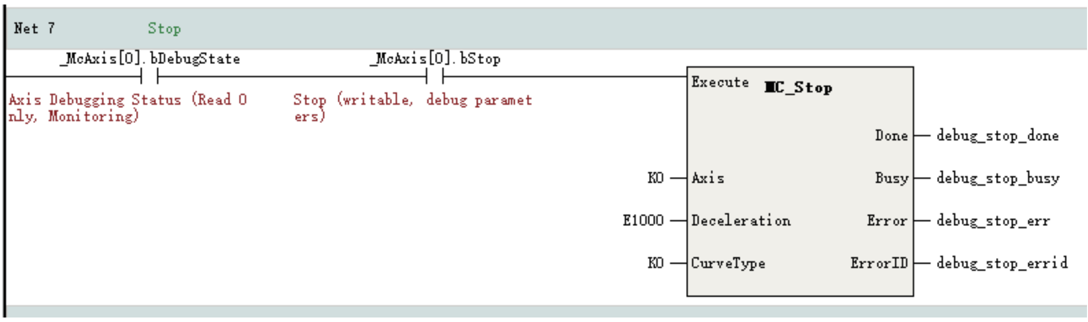
- Jog: When an axis is in the Standstill state, click "Jog+" to move the axis in the forward direction according to the target velocity set in the "Forward jog" field. Click "Jog–" to move the axis in the opposite direction according to the target velocity set in the "Reverse jog" field. The execution logic is as follows:



- Reset: When an axis is in the ErrorStop state, click "Reset" in attempt to reset the axis faults. If the servo drive fault cannot be reset, the reset may fail. The execution logic is as follows:



- Stop: There are two stop buttons in the interface, both with the same function, and both for stopping the motion of an axis. The deceleration used is the limit deceleration in the mode/ parameter settings of an axis. The execution logic is as follows:

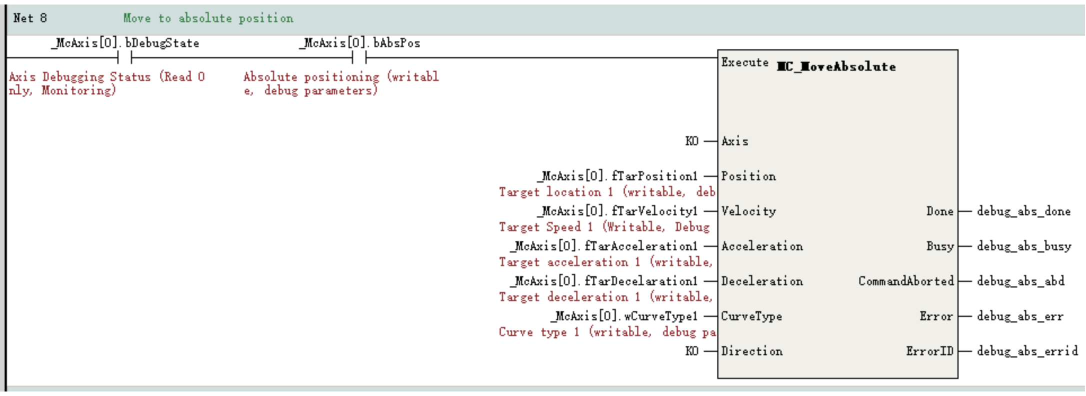


3. Motion modes

A motion mode can only be set when an axis is enabled.

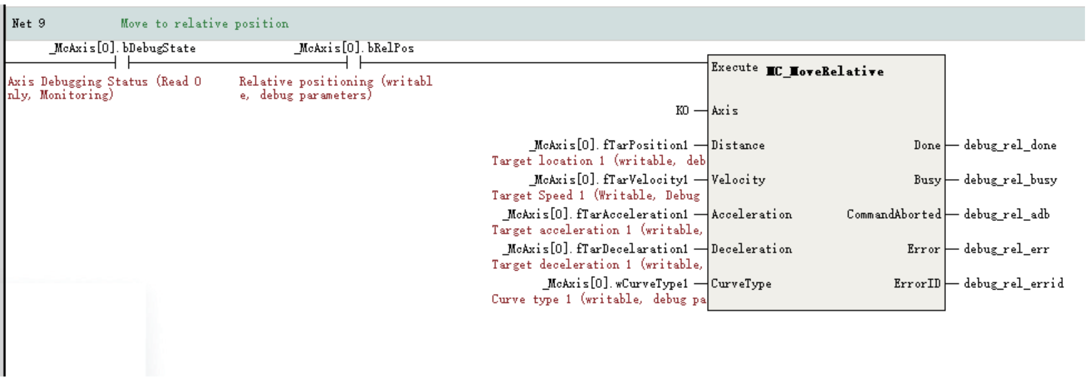
- Absolute positioning

  After the control mode is set to absolute positioning, you can set five parameters: target position, target velocity, acceleration, deceleration, and curve type. Click the start button to start absolute positioning according to the above parameters. The execution logic is as follows:


Net 8
Move to absolute position
_McAxis[0].bDebugState
Axis Debugging Status (Read Only, Monitoring)
_McAxis[0].bAbsPos
Absolute positioning (writable, debug parameters)
Execute
MC_MoveAbsolute
K0
Axis
_McAxis[0].fTarPosition1
Target location 1 (writable, deb
Position
_McAxis[0].fTarVelocity1
Target Speed 1 (Writable, Debug
Velocity
Done
debug_abs_done
_McAxis[0].fTarAcceleration1
Target acceleration 1 (writable,
Acceleration
Busy
debug_abs_busy
_McAxis[0].fTarDeceleration1
Target deceleration 1 (writable,
Deceleration
CommandAborted
debug_abs_abd
_McAxis[0].wCurveType1
Curve type 1 (writable, debug pa
CurveType
Error
debug_abs_err
K0
Direction
ErrorID
debug_abs_errid

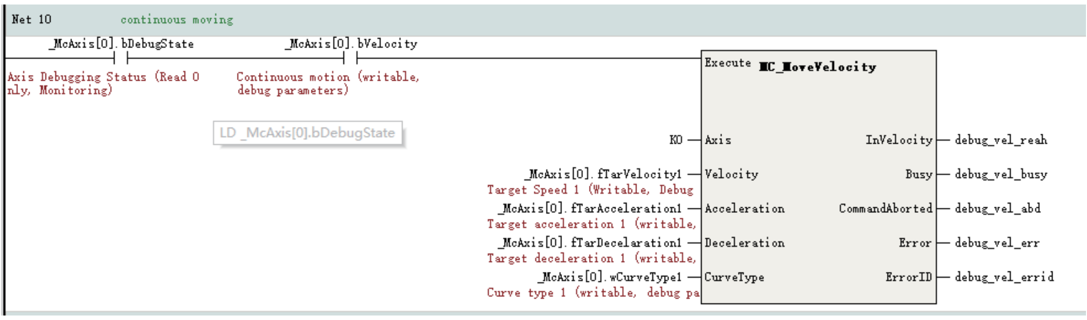
- Relative positioning

  After the control mode is set to relative positioning, you can set five parameters: target position, target velocity, acceleration, deceleration, and curve type. Click the start button to start relative positioning according to the above parameters. The execution logic is as follows:


Net 9
Move to relative position
_McAxis[0].bDebugState
Axis Debugging Status (Read Only, Monitoring)
_McAxis[0].bRelPos
Relative positioning (writable, debug parameters)
Execute
MC_MoveRelative
K0
Axis
_McAxis[0].fTarPosition1
Target location 1 (writable, deb
Distance
Done
debug_rel_done
_McAxis[0].fTarVelocity1
Target Speed 1 (Writable, Debug
Velocity
Busy
debug_rel_busy
_McAxis[0].fTarAcceleration1
Target acceleration 1 (writable,
Acceleration
CommandAborted
debug_rel_adb
_McAxis[0].fTarDeceleration1
Target deceleration 1 (writable,
Deceleration
Error
debug_rel_err
_McAxis[0].wCurveType1
Curve type 1 (writable, debug pa
CurveType
ErrorID
debug_rel_errid

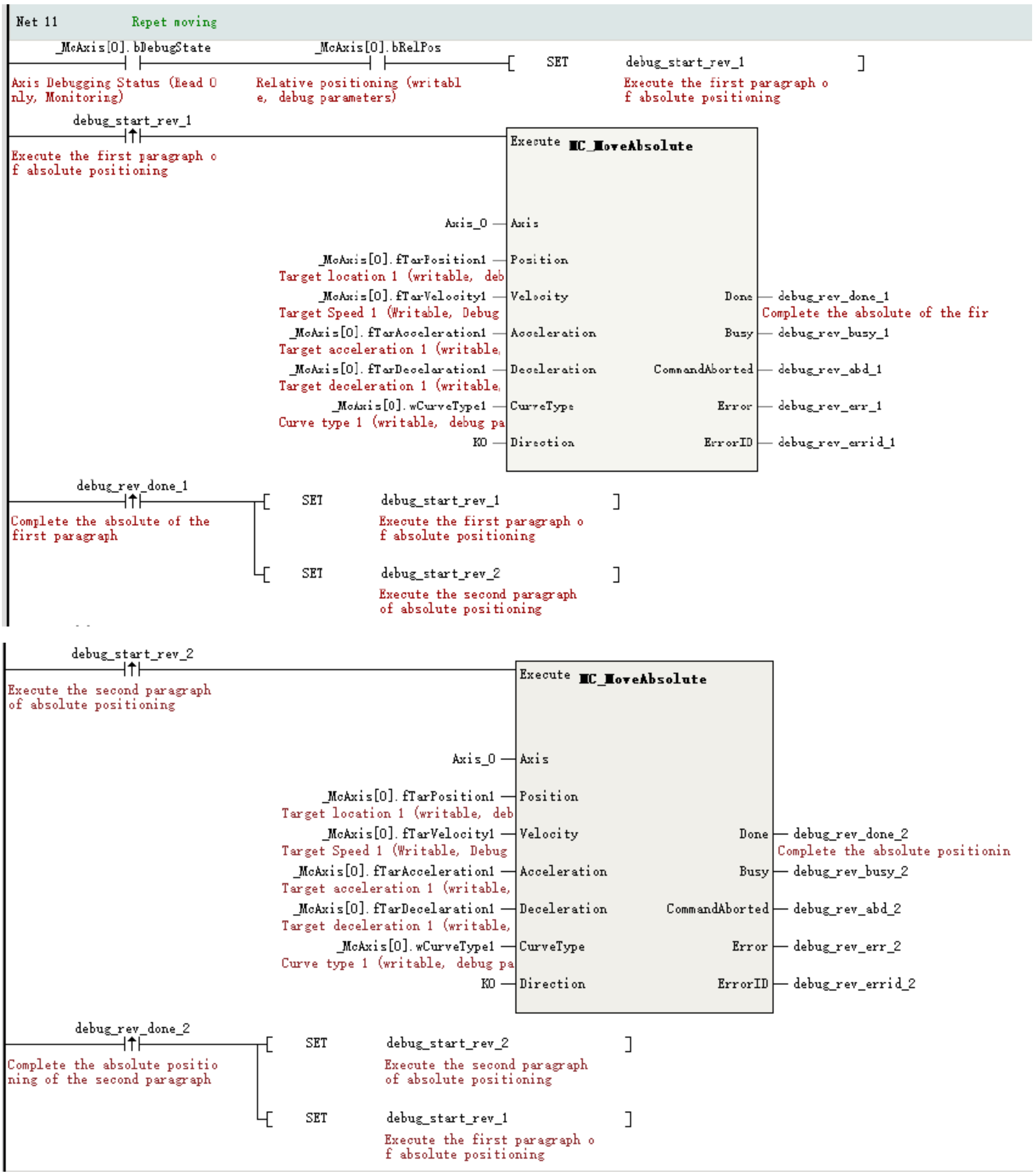
- Continuous motion

  After the control mode is set to continuous motion, you can set four parameters: target velocity, acceleration, deceleration, and curve type. Click the start button to start absolute positioning according to the above parameters. The execution logic is as follows:

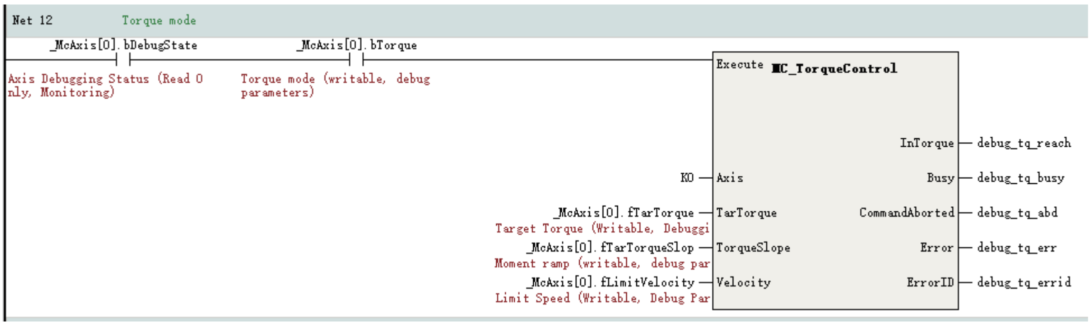

Net 10
continuous moving
_McAxis[0].bDebugState
Axis Debugging Status (Read Only, Monitoring)
_McAxis[0].bVelocity
Continuous motion (writable, debug parameters)
LD _McAxis[0].bDebugState
Execute
MC_MoveVelocity
K0
Axis
InVelocity
debug_vel_reah
_McAxis[0].fTarVelocity1
Target Speed 1 (Writable, Debug
Velocity
Busy
debug_vel_busy
_McAxis[0].fTarAcceleration1
Target acceleration 1 (writable,
Acceleration
CommandAborted
debug_vel_abd
_McAxis[0].fTarDeceleration1
Target deceleration 1 (writable,
Deceleration
Error
debug_vel_err
_McAxis[0].wCurveType1
Curve type 1 (writable, debug pa
CurveType
ErrorID
debug_vel_errid

- Reciprocating motion

  After the control mode is set to the reciprocating motion, you can set two sets of target positions, target velocities, acceleration rates, deceleration rates, and curve types. Click the start button, and the axis will first locate the target position 1 according to the set parameters, and then locate the target position 2, followed by reciprocating. The execution logic is as follows:


Net 11
Repet moving
_McAxis[0].bDebugState
Axis Debugging Status (Read Only, Monitoring)
_McAxis[0].bRelPos
Relative positioning (writable, debug parameters)
SET
debug_start_rev_1
Execute the first paragraph of absolute positioning
debug_start_rev_1
Execute the first paragraph of absolute positioning
Execute
MC_MoveAbsolute
Axis_0
Axis
_McAxis[0].fTarPosition1
Target location 1 (writable, deb
Position
_McAxis[0].fTarVelocity1
Target Speed 1 (Writable, Debug
Velocity
Done
debug_rev_done_1
Complete the absolute of the fir
_McAxis[0].fTarAcceleration1
Target acceleration 1 (writable,
Acceleration
Busy
debug_rev_busy_1
_McAxis[0].fTarDeceleration1
Target deceleration 1 (writable,
Deceleration
CommandAborted
debug_rev_abd_1
_McAxis[0].wCurveType1
Curve type 1 (writable, debug pa
CurveType
Error
debug_rev_err_1
K0
Direction
ErrorID
debug_rev_errid_1
debug_rev_done_1
Complete the absolute of the first paragraph
SET
debug_start_rev_1
Execute the first paragraph of absolute positioning
SET
debug_start_rev_2
Execute the second paragraph of absolute positioning
debug_start_rev_2
Execute the second paragraph of absolute positioning
Execute
MC_MoveAbsolute
Axis_0
Axis
_McAxis[0].fTarPosition1
Target location 1 (writable, deb
Position
_McAxis[0].fTarVelocity1
Target Speed 1 (Writable, Debug
Velocity
Done
debug_rev_done_2
Complete the absolute positionin
_McAxis[0].fTarAcceleration1
Target acceleration 1 (writable,
Acceleration
Busy
debug_rev_busy_2
_McAxis[0].fTarDeceleration1
Target deceleration 1 (writable,
Deceleration
CommandAborted
debug_rev_abd_2
_McAxis[0].wCurveType1
Curve type 1 (writable, debug pa
CurveType
Error
debug_rev_err_2
K0
Direction
ErrorID
debug_rev_errid_2
debug_rev_done_2
Complete the absolute positioning of the second paragraph
SET
debug_start_rev_2
Execute the second paragraph of absolute positioning
SET
debug_start_rev_1
Execute the first paragraph of absolute positioning

- Torque mode
  After the control mode is set to the torque mode, you can set three parameters: target torque, torque slope, and limit velocity. Click the start button to start absolute positioning according to the above parameters. The execution logic is as follows:



12.5.3 Instruction Control Rules

Motion of axes can be controlled in the PLC through instructions. The rules for calling instructions are as follows.

- Instructions do not need to be instantiated.
- The axis number in the instruction is the unique identifier to access the axis.
- The priority of motion instructions is generally lower than the priority of online commissioning mode.
- Floating-point parameters in the instructions must meet the precision range of floating-point numbers, which generally contain 7 valid digits and can be set to 9999999 at the maximum.

12.5.4 Limit Handling

Two limit detection methods are supported: software limit detection and hardware limit detection.

- To ensure correct processing of hardware limit signals, 0x60Fd must be configured in the process data.
- Software limit processing is only valid for calling position and velocity instructions in linear mode. It is invalid for calling homing and torque instructions.
- If the absolute target position of the called position instruction is within the software limit, the instruction can be executed. If the absolute target position exceeds the software limit, the execution of the positioning instruction is interrupted and the axis stops at the software limit.
- When a velocity instruction is called, if the current velocity of the axis exceeds the software limit, the execution of the velocity instruction is interrupted and the axis stops at the software limit.
- If the axis position has exceeded the positive (negative) limit, the axis can only run in the negative (positive) direction.

The following table lists system variables of limits.

Variable	Function
bphlimit	Positive limit input status of hardware
bnhlimit	Negative limit input status of hardware
bpslimit	Software positive limit status
bnslimit	Software negative limit status

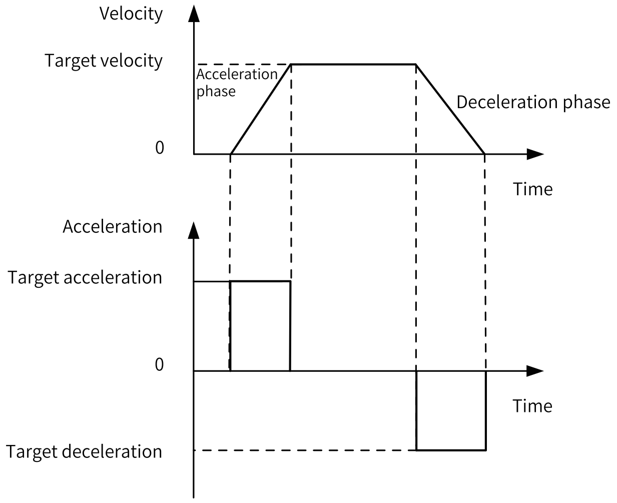
12.5.5 Positioning Curve

Two types of velocity curves are supported: T-shape acceleration/deceleration curve, and 5-segment S-shape acceleration/deceleration curve. The curve type is determined by the CuriveType parameter in the instructions.

When the axis reaches the limit or enters the ErrorStop state and decelerates to stop upon a fault, the axis decelerates to stop according to the T-shape curve.

T-shape velocity curve

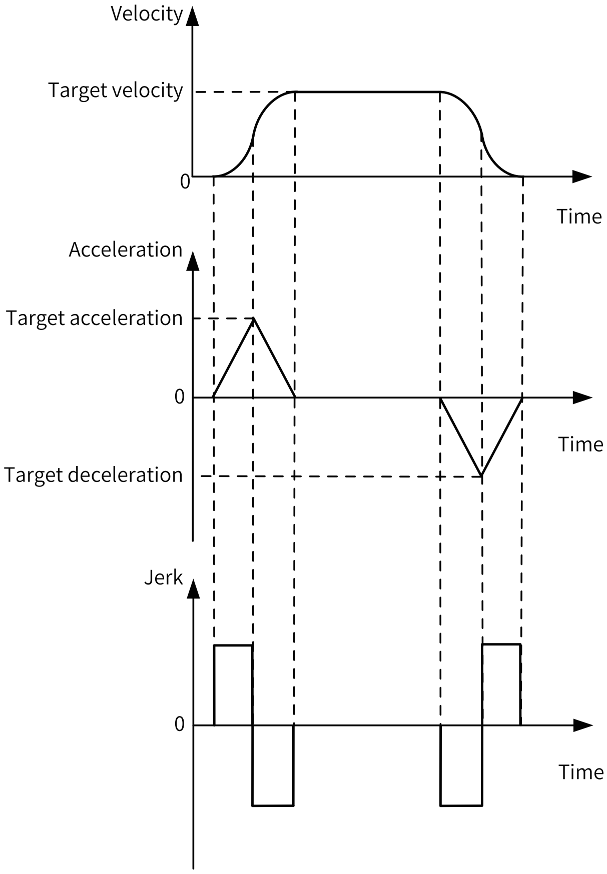
When CuriveType in the instruction is 0, the axis accelerates or decelerates according to the T-shape curve. In the T-shape velocity curve, the axis plans the curve according to the target position, target velocity, target acceleration rate, and target deceleration rate. The actual acceleration/deceleration rate is fixed during acceleration/deceleration. The following figure shows a positioning curve.



- Target position: Indicates the final position of the axis in the absolute positioning instruction, in the unit of Unit (user unit)
- Target velocity: Indicates the maximum velocity of the axis, in the unit of Unit/s (user unit/second).
- Target acceleration rate: Indicates the amount of change in velocity per second when the axis accelerates, in the unit of Unit/t^2.
- Target deceleration rate: Indicates the amount of change in velocity per second when the axis decelerates, in the unit of Unit/t^2.
- Assume that the axis initial velocity is Vs, target velocity is Vt, and target acceleration rate is Acc, then the acceleration time is:
  Tacc = (Vt – Vs)/Acc
- Assume that the axis initial velocity is Vs, target velocity is Ve, and target deceleration rate is Dec, then the deceleration time is:
  Tdec = (Vs – Ve)/Dec

S-shape velocity curve

When CuriveType in the instruction is 1, the axis accelerates or decelerates according to the S-shape curve. In the 5-segment S-shape curve, the axis plans the curve according to the target position, target velocity, target acceleration rate, and target deceleration rate, of which the target acceleration rate and target deceleration rate refer to the maximum accelerate rate and deceleration rate the axis can reach. The following figure shows a positioning curve.



The 5-segment S-shape velocity curve is divided into 5 stages based on the acceleration rate state: increasing acceleration, decreasing acceleration, constant speed, increasing deceleration, and decreasing deceleration. The constant acceleration and constant deceleration stages do not exist. During variable acceleration stages such as increasing acceleration or increasing deceleration, the actual Jerk is calculated by the PLC and cannot be set by users.

- Target position: Indicates the final position of the axis in the absolute positioning instruction, in the unit of Unit (user unit)
- Target velocity: Indicates the maximum velocity of the axis, in the unit of Unit/s (user unit/second).
- Target acceleration rate: Indicates the amount of change in velocity per second when the axis accelerates at variable velocities, in the unit of Unit/t^2. In the velocity curve, the acceleration rate at the moment (t2) when the stage changes from increasing acceleration to decreasing acceleration is certainly the target acceleration rate.
- Target deceleration rate: Indicates the amount of change in velocity per second when the axis decelerates at variable velocities, in the unit of Unit/t^2. In the velocity curve, the deceleration rate at the moment (t5) when the stage changes from decreasing acceleration to decreasing deceleration is certainly the target deceleration rate.
- Assume that the axis initial velocity is Vs, target velocity is Vt, and target acceleration rate is Acc, then the acceleration time is:
  Tacc = 2 x (Vt – Vs)/Acc
-

  Assume that the axis initial velocity is Vs, target velocity is Ve, and target deceleration rate is Dec, then the deceleration time is:

  Tdec = 2 x (Vs – Ve)/Dec

12.6 Dragging Motion Control Axes

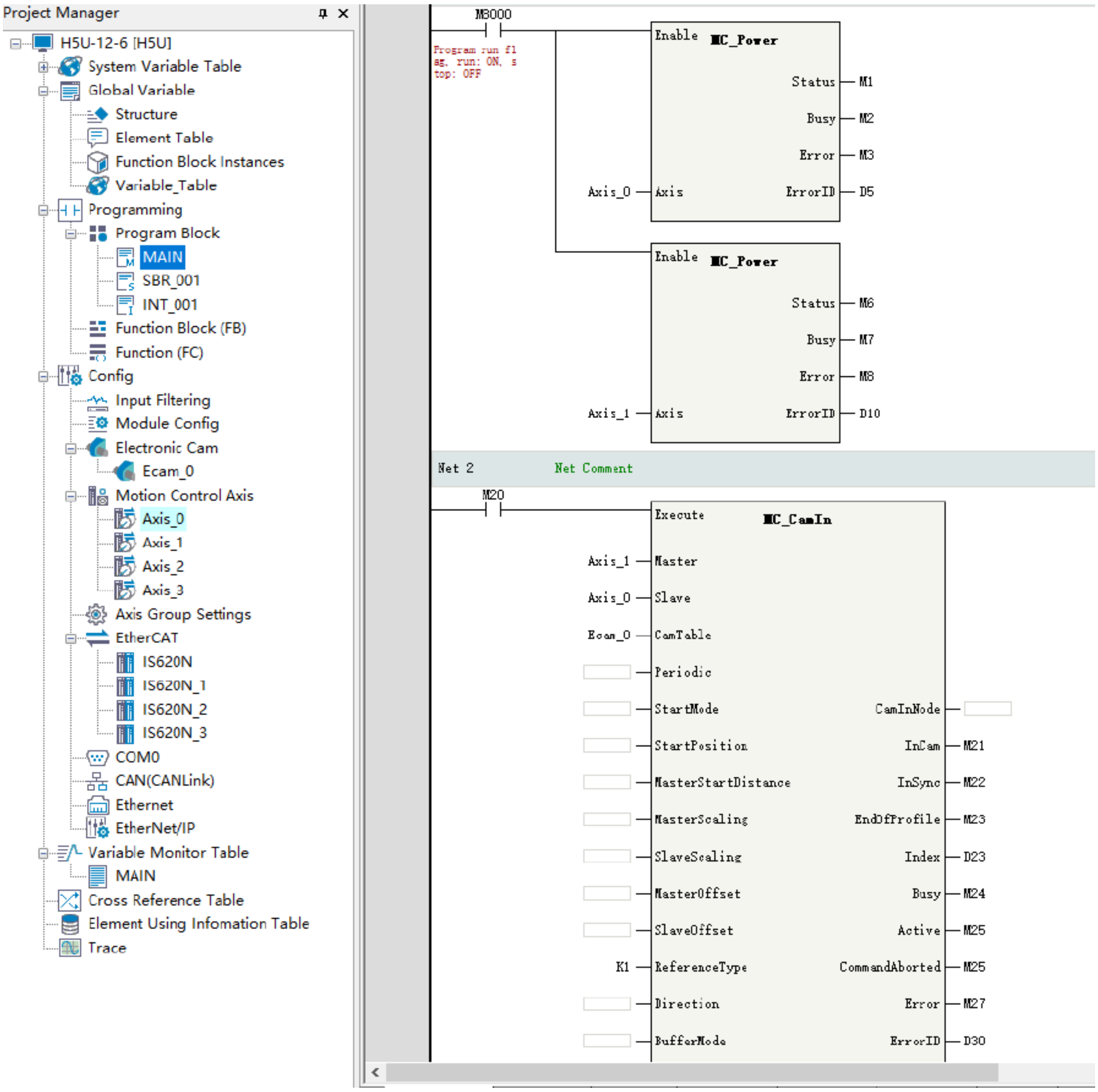
You can directly drag motion control axes to adjust their configuration sequences.

Note

This function is available in AutoShop V4.4.0.0 and PCB software V5.0.0.0 or later versions.

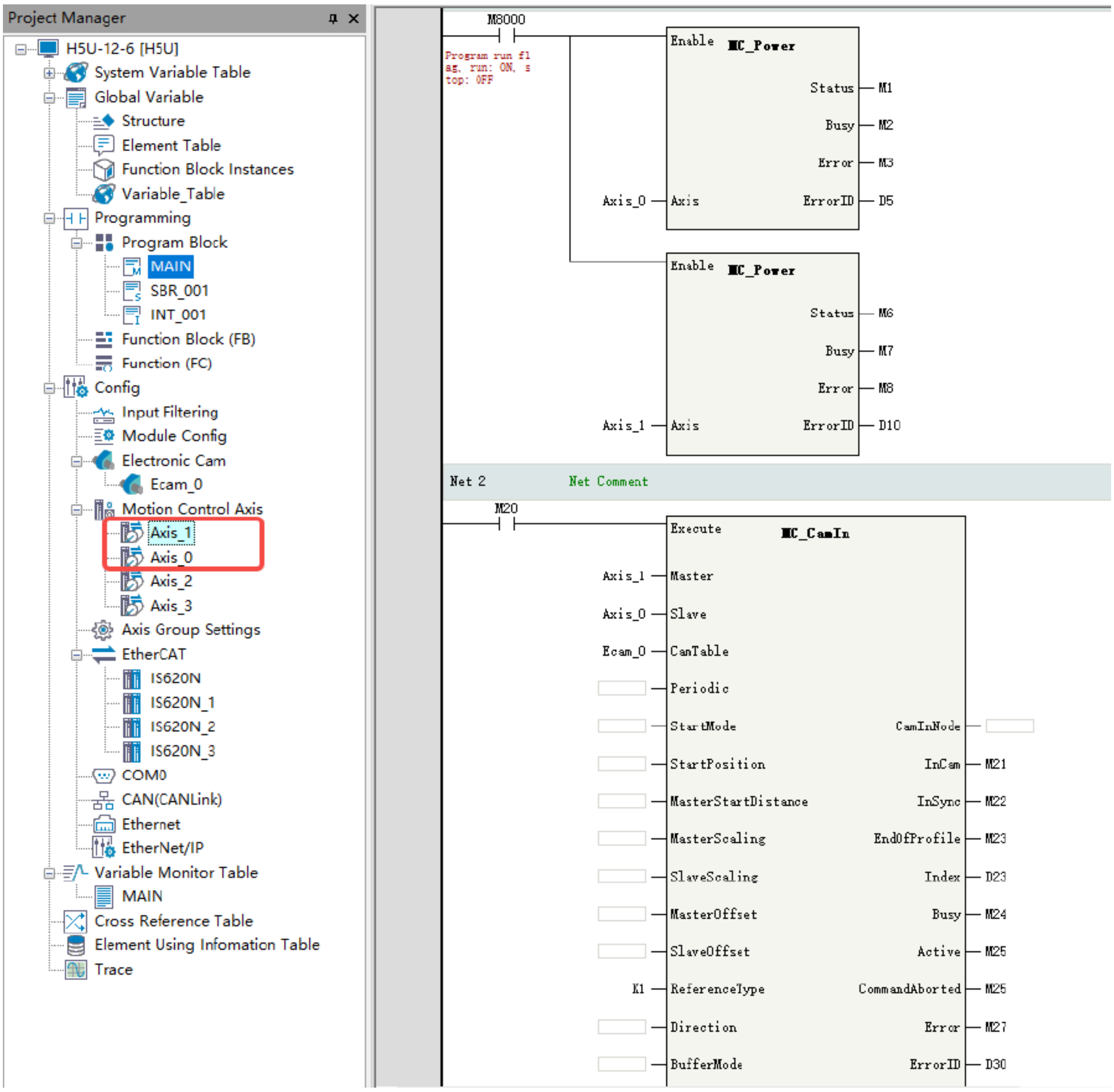
Example

Create a project containing four bus servo axes with Axis_1 as the cam master axis and Axis_0 as the cam slave axis. When the cam position type is 1 (Position set in the current period), an error is reported after the MC_CamIn instruction is called. The fault code is 9251, indicating that the configured master axis address is equal to or greater than the slave axis address. In this case, you can drag motion control axes to adjust their sequences to eliminate the error as follows.



Hover the pointer over Axis_0, hold down the left mouse button, drag Axis_0 to cover Axis_1, and then release the mouse. The configuration positions of Axis_0 and Axis_1 are changed. The servos bound with Axis_0 and Axis_1 remain unchanged.

Download the adjusted configuration to the PLC and call the MC_CamIn instruction again. Then, no error will be reported.



12.7 Modifying Axis Configuration Parameters Using Instructions

You can modify configuration parameters of an axis in the PLC program to meet different application requirements, such as software limit and rotation cycle in ring mode.

Function

- To modify configuration parameters of an axis in the PLC program, you need to use the system structure variable _sCfgAxis of the axis configuration to set parameters. The setting of this structure is not retentive upon power failure.
- After the PLC is powered on, this structure variable is initialized based on the axis configuration in the software background.
- After modifying the value of the initialized variable based on application requirements, call the MC_SetAxisConfigPara instruction. Then, the setting takes effect.

Note

The axis configuration parameters are variables that are not retentive at power failure. After the PLC is restarted, the previously set values are lost. Therefore, set the values every time the PLC is started.

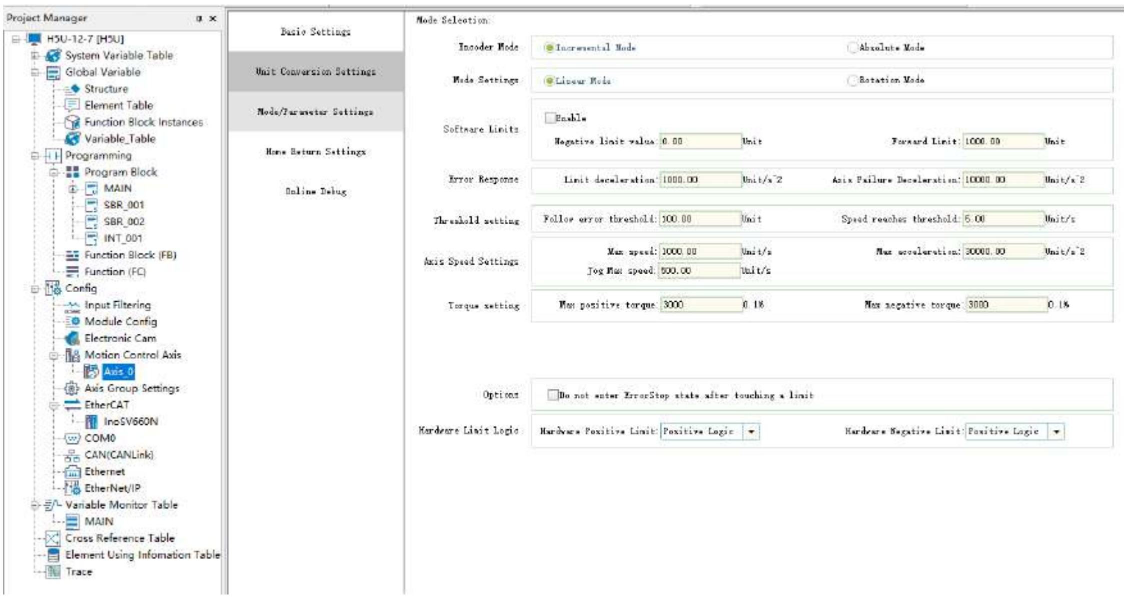
Table 12–3 Definitions of _sCfgAxis structure variables

Variable	Data Type	Description
dPlusePreCycle	DINT	Number of pulses per revolution of the motor/encoder
fDistancePreCycle	REAL	Distance per revolution of the worktable
dNumerator	DINT	Gear ratio numerator
dDenominator	DINT	Gear ratio denominator
bDirection	BOOL	Direction OFF: Forward ON: Reverse
bVirtualMode	BOOL	Virtual axis mode OFF: Invalid ON: Valid
bSoftLimitEnable	BOOL	Software limit enable control OFF: Disabled ON: Enabled
bEnterErrorStop	BOOL	Not entering ErrorStop state upon an axis fault OFF: Disabled ON: Enabled
bPLimitTerminalPolarity	BOOL	Forward limit polarity selection OFF: Positive logic ON: Negative logic
bNLimitTerminalPolarity	BOOL	Negative limit polarity selection OFF: Positive logic ON: Negative logic
bHomeTerminaPolarity	BOOL	Home signal polarity selection OFF: Positive logic ON: Negative logic
iEncoderMode	INT	Encoder mode (valid for bus servo axis) 0: Absolute 1: Incremental
iLineRotateMode	INT	Selection of linear or rotary mode 0: Linear 1: Rotary
fPLimit	REAL	Positive limit in linear mode
fNLimit	REAL	Negative limit in linear mode
fRotation	REAL	Rotation period in rotary mode
fLimitDeceleraion	REAL	Limit deceleration rate
fErrorStopDeceleration	REAL	Deceleration rate at axis fault
fFollowErrorWindow	REAL	Following error threshold
fInVelocityWindow	REAL	Velocity reach threshold

Variable	Data Type	Description
fMaxVelocity	REAL	Max. velocity
fMaxJogVelocity	REAL	Max. jog velocity
fMaxAcc	REAL	Max. acceleration rate
fMaxPTorque	REAL	Max. positive torque (bus servo axis) Only used for instruction parameter inspection, not delivered to servo
fMaxNTorque	REAL	Max. negative torque (bus servo axis) Only used for instruction parameter inspection, not delivered to servo
iHomeMethod	INT	Homing mode, local pulse axis Options are 17 to 30 and 35.
fHomeVelocity	REAL	Homing velocity, valid for local pulse axis
fHomeApproachVelocity	REAL	Homing approaching velocity, valid for local pulse axis
fHomeAcceleration	REAL	Homing acceleration rate, valid for local pulse axis
dHomeTimeOut	DINT	Homing timeout time, valid for local pulse axis
iHomePositionMode	INT	Homing position mode selection, local pulse axis 0: Absolute homing 1: Relative homing
dTouchProbeID1	DINT	ID of the probe terminal 1 (Modbus address) –2: Disabled –1: Reserved
dTouchProbeID2	DINT	ID of the probe terminal 2 (Modbus address) –2: Disabled –1: Reserved
iPluseMethod	INT	Pulse output mode (valid for local pulse axis) 3: Phase A/B 4: Pulse + direction 5: CW/CCW
dPLimitTerminalID	DINT	ID of positive limit signal (Modbus address) –2: Disabled –1: Reserved
dNLimitTerminalID	DINT	ID of negative limit signal (Modbus address) –2: Disabled –1: Reserved
dHomeTerminalID	DINT	ID of home signal (Modbus address) Only X0 to X7 are supported. –2: Disabled –1: Reserved

Example

1. Create a network consisting of H5Us and IS620Ns, enable EtherCAT communication, and add a motion control axis Axis_0 with default parameters.



Mode Selection:

Incoder Mode: ● Incremental Mode ○ Absolute Mode

Mode Settings: ● Linear Mode ○ Rotation Mode

Software Limits: □ Enable
Negative limit value: 0.00 Unit   Forward Limit: 1000.00 Unit

Error Response: Limit deceleration: 1000.00 Unit/s^2   Axis Failure Deceleration: 10000.00 Unit/s^2

Threshold setting: Follow error threshold: 100.00 Unit   Speed reaches threshold: 5.00 Unit/s

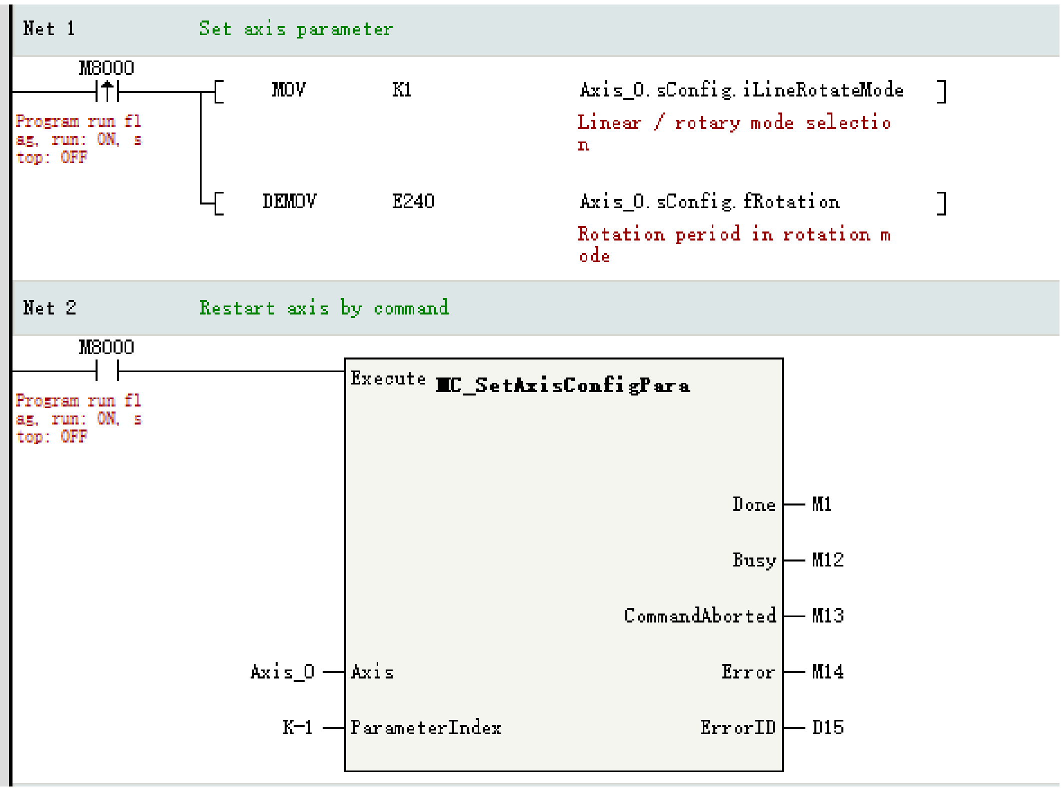
Axis Speed Settings: Max speed: 1000.00 Unit/s   Max acceleration: 30000.00 Unit/s^2
Jog Max speed: 500.00 Unit/s

Torque setting: Max positive torque: 3000 0.1%   Max negative torque: 3000 0.1%

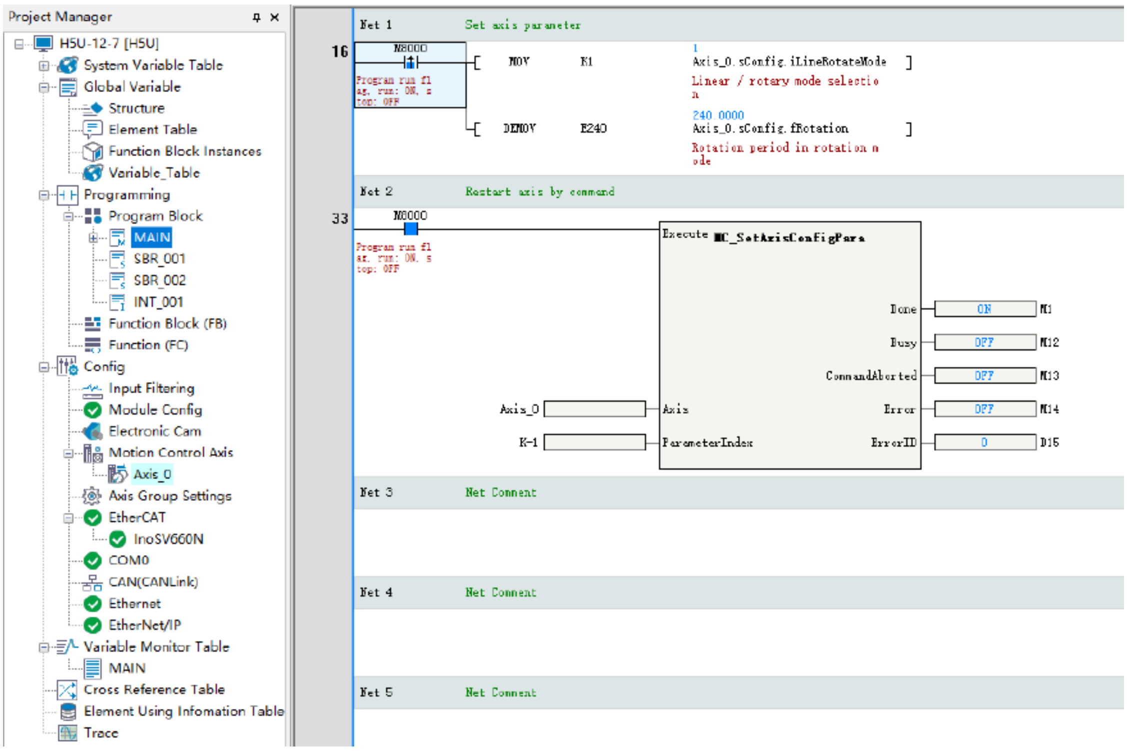
Options: □ Do not enter ErrorStop state after touching a limit

Hardware Limit Logic: Hardware Positive Limit: Positive Logic ▾   Hardware Negative Limit: Positive Logic ▾

2. Compile the PLC program, modify the mode of Axis_0 to rotary, set the rotation period to 240, and then call the MC_SetAxisConfigPara instruction to make the settings take effect.


```
Net 1          Set axis parameter
    M8000
----|↑|------+--[ MOV     K1      Axis_0.sConfig.iLineRotateMode  ]
Program run fl|                   Linear / rotary mode selectio
ag, run: ON, s|                   n
top: OFF      |
              +--[ DEMOV   E240    Axis_0.sConfig.fRotation        ]
                                  Rotation period in rotation m
                                  ode

Net 2          Restart axis by command
    M8000
----| |--------------------------- Execute  MC_SetAxisConfigPara
Program run fl
ag, run: ON, s
top: OFF
                                                         Done --- M1
                                                         Busy --- M12
                                               CommandAborted --- M13
                         Axis_0 --- Axis                Error --- M14
                            K-1 --- ParameterIndex    ErrorID --- D15
```

3. Connect one IS620N to the EtherCAT network port of the controller, and compile and download the program. The following figure shows the running effect.



```
Net 1         Set axis parameter
16   M8000
     Program run flag, run: ON, stop: OFF
        MOV     K1      1
                        Axis_0.sConfig.iLineRotateMode   ]
                        Linear / rotary mode selection
        DEMOV   E240    240.0000
                        Axis_0.sConfig.fRotation         ]
                        Rotation period in rotation mode
Net 2         Restart axis by command
33   M8000
     Program run flag, run: ON, stop: OFF
                  Execute MC_SetAxisConfigPara
                                          Done  ON   M1
                                          Busy  OFF  M12
                              CommandAborted    OFF  M13
     Axis_0    Axis                       Error OFF  M14
     K-1       ParameterIndex           ErrorID 0    D15
Net 3         Net Comment
Net 4         Net Comment
Net 5         Net Comment
```

12.8 Fault Categories

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

- Instruction faults are faults of MC axis control instructions due to reasons such as inappropriate instruction parameter settings or change to the PLCOpen state machine during axis running. You can obtain the fault codes based on ErrorID in the failed instructions.
- Axis faults are faults of axes, such as excessive following error. You can view axis fault codes in four ways. Method 1: On the "Online debug" page of the background, view the fault codes in the "Axis fault" column. Method 2: Obtain fault codes based on AxisErrorID in the MC_ReadAxisError instruction. Method 3: Obtain fault codes based on wAxisError in the axis system variables. Method 4: View fault codes on the "Fault Diagnosis" page.
- Drive faults are faults of the EtherCAT bus drive or local pulse output axis. To view the fault codes of the EtherCAT bus drive, you must configure 0x603F in the PDO mapping and associate it with the axis. You can view drive faults in three ways. Method 1: On the "Online debug" page of the background, view the fault codes in the "Servo error" column. Method 2: Obtain fault codes based on ServoErrorID in the MC_ReadAxisError instruction. Method 3: Obtain fault codes based on wServoError in the axis system variables.

13 High-speed Counter

13.1 Introduction to High-speed Counter Axes

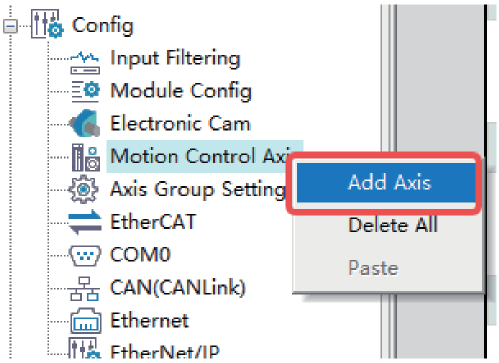
In AutoShop software and engineering applications, a counter is used as an encoder axis to enable management, and the counter associated with an axis is known as a counter axis.

AutoShop supports 4-axis 32-bit high-speed counters, which can realize phase AB frequency multiplication by 1/2/4, CW/CCW, pulse+direction, and single-phase counting. Counting signal sources are external pulse input or internal 1 ms/1 μs clock counting, which can be used to preset and latch the counter in combination with other input signals.

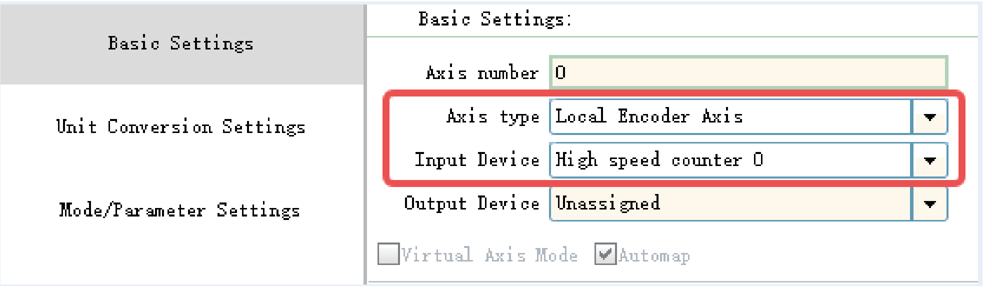
13.2 Creating Counter Axes

Before using a counter in AutoShop, you must associate the counter with an axis.

1. In the column "Project Manager", right-click a motion control axis under "Config" and choose "Add Axis" to create a motion control axis.



2. Double-click the added axis (Axis_0 is selected in the following figure). On the "Basic setting" interface displayed, select "Local encoder axis" as the axis type, and select "High Speed Counter" as the input device to associate the axis and counter. The axis number is used in the program as an axis identifier to control the corresponding counter axis.



Basic Settings:

Axis number 0

Axis type Local Encoder Axis

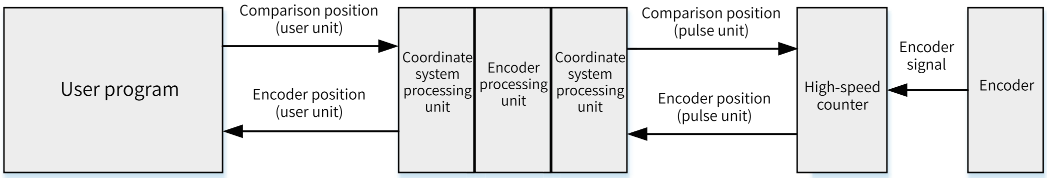
Input Device High speed counter 0

Output Device Unassigned

□ Virtual Axis Mode ☑ Automap

13.3 Counter Axis User Unit and Conversion

High-speed counters use pulse units when decoding encoder signals, and use common measurement units for counter instructions such as millimeters, degrees, and inches, which are called user units (Unit). Through unit conversion, the number of pulses can be converted into user units (Unit), which can be defined as equipment-related units (such as millimeters and revolutions) according to actual applications.



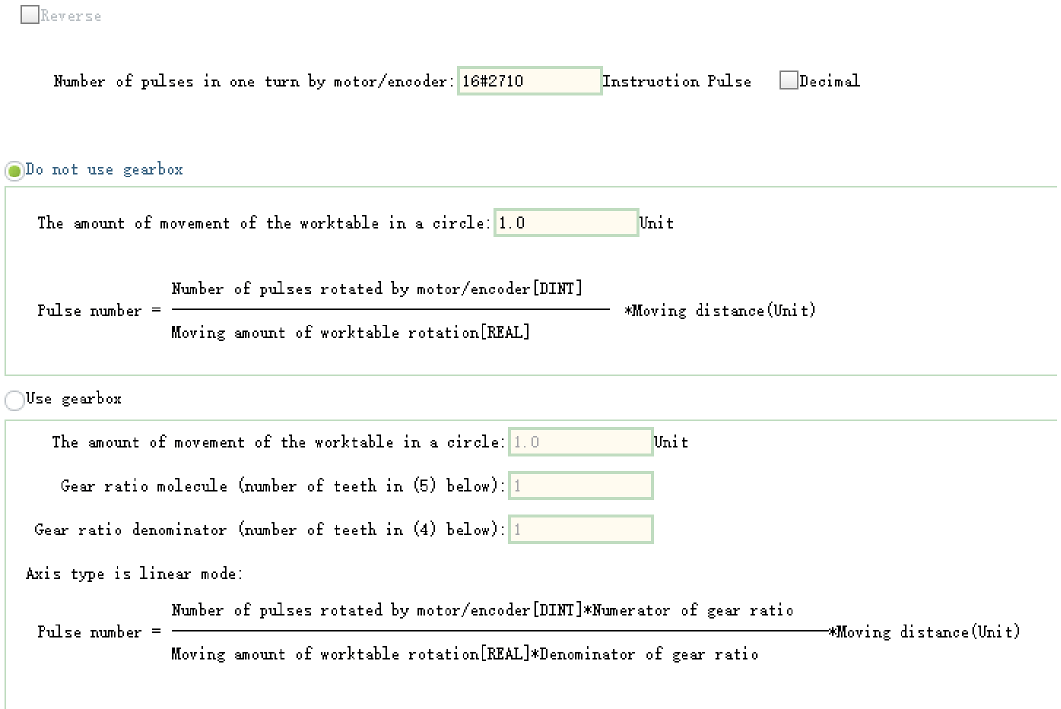
The following table lists parameters that need to be set for unit conversion.

Parameter	Function
Number of pulses in one turn by motor/encoder	Set the number of pulses required for the motor to rotate one turn according to the encoder resolution.
Gear change mechanisms in use or not	Specify whether gear change mechanisms are in use or not.
Amount of movement in one turn by motor/encoder	The workpiece movement amount per turn of the motor when no gear change mechanism is in use
Amount of movement of the worktable in a circle	The workpiece movement amount per turn of the worktable when gear change mechanisms are in use
Gear ratio numerator	Set a gear ratio on the workpiece side.
Gear ratio denominator	Set a gear ratio on the motor side.

When a servo motor is connected to a screw rod through a reducer to drive the worktable and gives feedback about the worktable position to an encoder counter through the PLC, if the counter counts the encoder in pulses, the number of pulses shall be the unit, and if the counter axis is used to represent the worktable position, then millimeter shall be the unit.

Therefore, in the program, the Unit is used uniformly as the user unit of the counter axis.

The conversion rule between the user unit (Unit) and the pulse is as follows:



□Reverse

Number of pulses in one turn by motor/encoder: 16#2710 Instruction Pulse □Decimal

● Do not use gearbox

The amount of movement of the worktable in a circle: 1.0 Unit

$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]}}{\text{Moving amount of worktable rotation[REAL]}} \text{*Moving distance(Unit)}$$

○ Use gearbox

The amount of movement of the worktable in a circle: 1.0 Unit

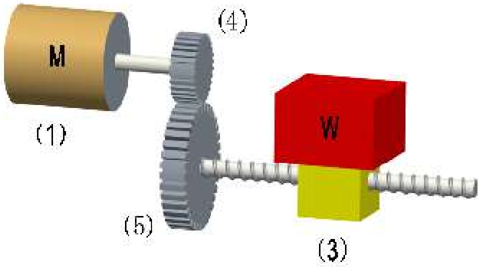
Gear ratio molecule (number of teeth in (5) below): 1

Gear ratio denominator (number of teeth in (4) below): 1

Axis type is linear mode:

$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]*Numerator of gear ratio}}{\text{Moving amount of worktable rotation[REAL]*Denominator of gear ratio}} \text{*Moving distance(Unit)}$$

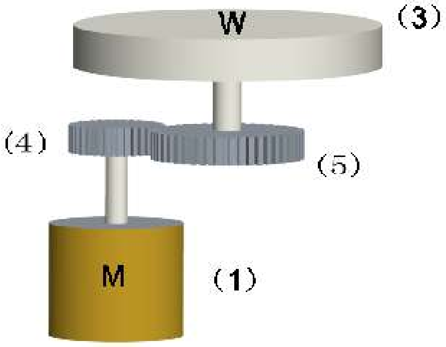
M: Motor/Encoder, W: Workbench



Axis type is rotation mode:

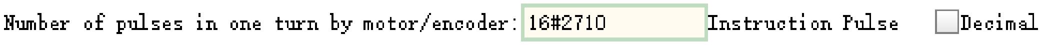
$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]*Numerator of gear ratio}}{\text{Moving amount of worktable rotation[REAL]*Denominator of gear ratio}}\text{*Moving distance(Unit)}$$

M: Motor/Encoder, W: Workbench



In the unit conversion setting, set the relevant parameters according to the actual device.

1. Number of pulses in one turn by motor/encoder: "16#" in the input box indicates that hexadecimal numbers are used.
   If the number of pulses in one turn by the encoder is 10000, whose hexadecimal equivalent is 2710, input 16#2710.



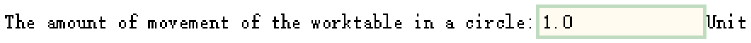
Number of pulses in one turn by motor/encoder: 16#2710 Instruction Pulse □Decimal

2. Working stroke setting: The working stroke can be set with or without gear change mechanisms.

- Without gear change mechanisms
  When gear change mechanisms are not in use, the conversion equation from user unit to pulse unit is as follows.

$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]*Numerator of gear ratio}}{\text{Moving amount of worktable rotation[REAL]*Denominator of gear ratio}}\text{*Moving distance(Unit)}$$

If one revolution of the encoder corresponds to one revolution of the working axis and the user unit (Unit) is revolution, then the working stroke of the motor/encoder per revolution shall be set to 1.



The amount of movement of the worktable in a circle: 1.0 Unit

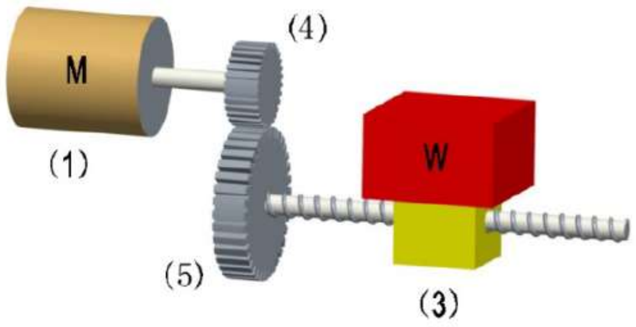
Take the Inovance 20-bit encoder as an example. The set parameters are as follows.

Number of pulses in one turn by motor/encoder = 1048576

Amount of movement in one turn by motor/encoder = 1

Then, when the target displacement given by the relative positioning instruction is 10, the actual number of pulses sent by the motion control axis is 10485760, then the motor rotates by 10 revolutions.

- With gear change mechanisms
  Typical working condition in linear mode is shown in the following figure.

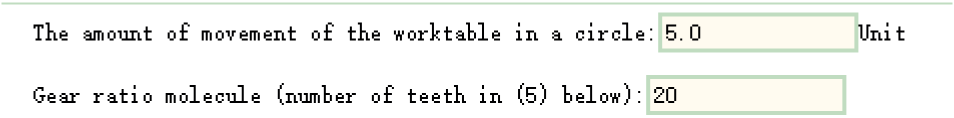


Where, (1) is the servo motor, (3) is the workpiece, (4) is the gear ratio numerator, and (5) is the gear ratio denominator.

The calculation equation from user unit to pulse unit is as follows.

$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]*Numerator of gear ratio}}{\text{Moving amount of worktable rotation[REAL]*Denominator of gear ratio}} \text{*Moving distance(Unit)}$$

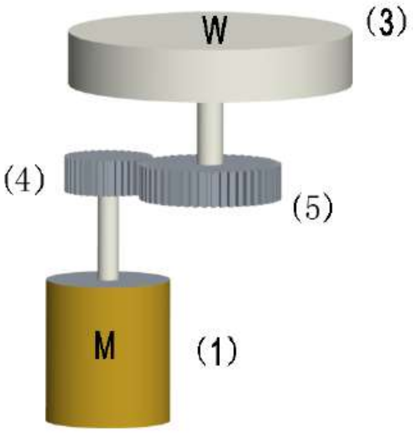
If a servo motor is connected to a screw rod through a reducer to drive the worktable, the working stroke per revolution of the screw rod is 5 mm, and the reduction ratio is 20:10. The setting is as follows.



The amount of movement of the worktable in a circle: 5.0 Unit

Gear ratio molecule (number of teeth in (5) below): 20

Typical working condition in ring mode is shown in the following figure.



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

The calculation equation from user unit to pulse unit is as follows.

$$\text{Number of pulses (unit: pulse)} = \frac{\text{Number of pulses per revolution of the motor/encoder [DINT] x Gear ratio numerator [DINT]}}{\text{Distance per revolution of the workbench [REAL] x Gear ratio denominator [DINT]}} \text{ x Distance (unit: Unit)}$$

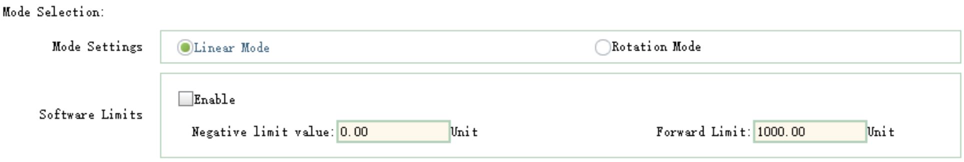
13.4 Setting Working Modes

13.4.1 Linear Mode

The counter axis moves between the negative and positive limits. After the counter axis reaches the limits, pulses in the same direction are still input; the counter axis reports an overflow while the counter axis position remains unchanged. After the counter axis reports an overflow, input the reverse pulses. The counter axis counts in reverse, and the overflow error is removed.

In linear mode, you can set the negative and positive position limits of the counter axis in the interface, with user unit (Unit) as the position unit. The negative limit must be less than or equal to 0, and the positive limit must be greater than or equal to 0.

Since the high-speed counter is a 32-bit counter, the negative and positive limits must be within the 32-bit integer range [–2147483648, +2147483647] after being converted to pulse units.



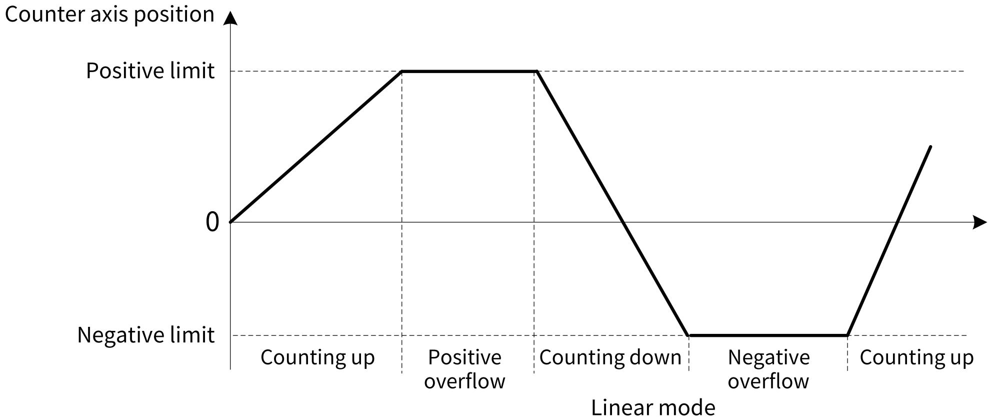
Mode Selection:

Mode Settings ● Linear Mode ○ Rotation Mode

Software Limits □ Enable

Negative limit value: 0.00 Unit Forward Limit: 1000.00 Unit

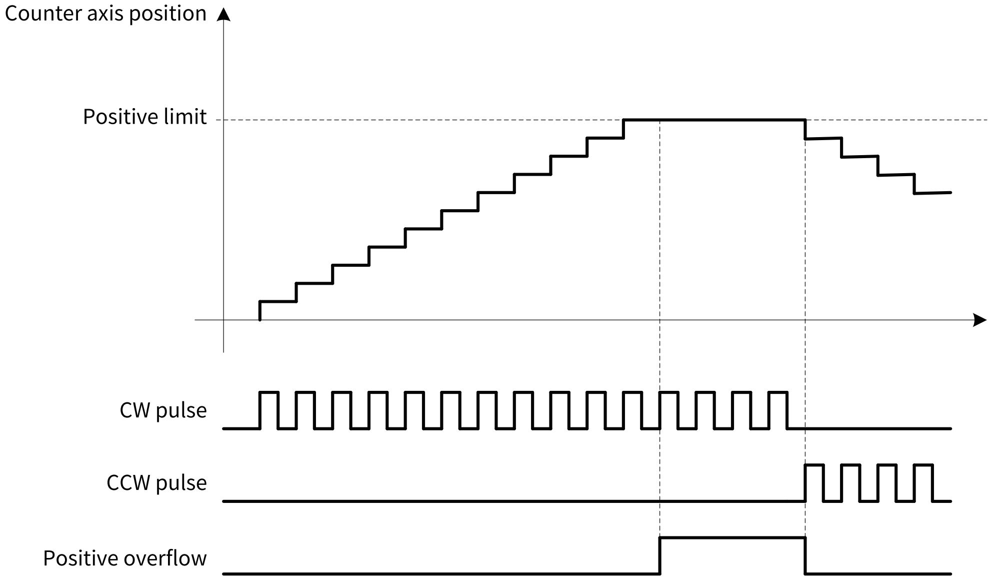
In linear mode, the high-speed counter operates in the closed interval of [negative limit, positive limit]. When the direction is negative, the count value decreases in the negative direction. After the negative limit is reached, the count value no longer decreases. When the direction is positive, the count value increases in the positive direction. After the positive limit is reached, the count value no longer increases.



Linear mode

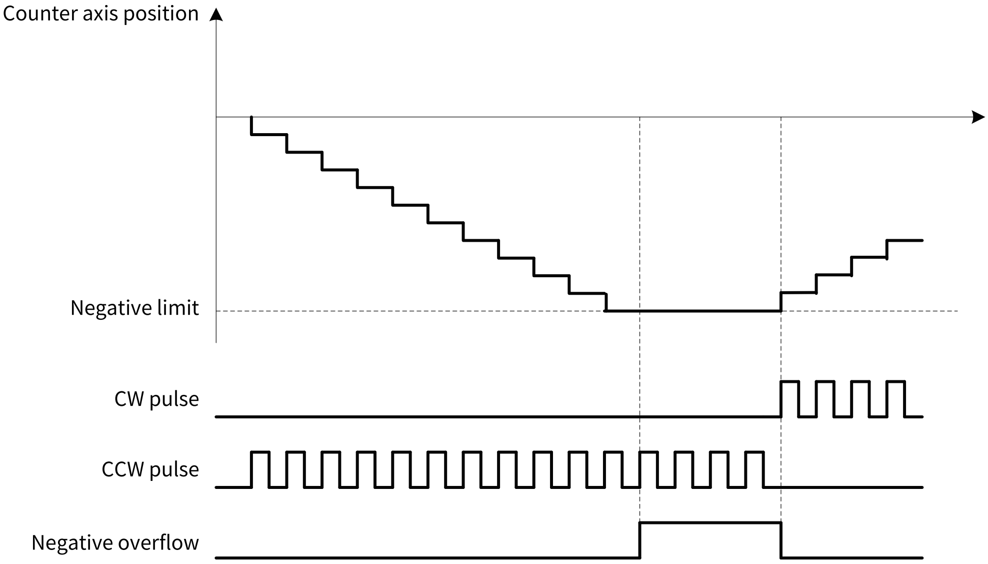
CW pulse counting

In linear mode, input CW pulses. After the incremental count of the counter axis position reaches the limit, keep inputting CW pulses. The counter axis reports a positive overflow error, and the counter axis position value remains unchanged. Input CCW pulses. The counter axis position counts down, and positive overflow error is removed.



CCW pulse counting

In linear mode, input CCW pulses. After the decremental count of the counter axis position reaches the limit, keep inputting CCW pulses. The counter axis reports a negative overflow error, and the counter axis position value remains unchanged. Input CW pulses. The counter axis position counts up, and the negative overflow error is removed.

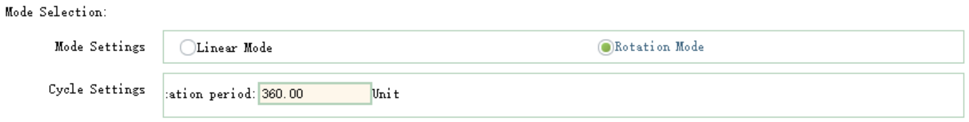


13.4.2 Rotary Mode

The position of the counter axis changes cyclically during the rotation cycle. In case of incremental count, the counter axis position turns 0 after reaching the maximum value in the rotation cycle; in case of decremental count, the counter axis position decreases from the maximum value in the rotation cycle after turning 0.

In rotary mode, you can set the rotation cycle of the counter axis in the interface, with user unit (Unit) as the cycle unit.

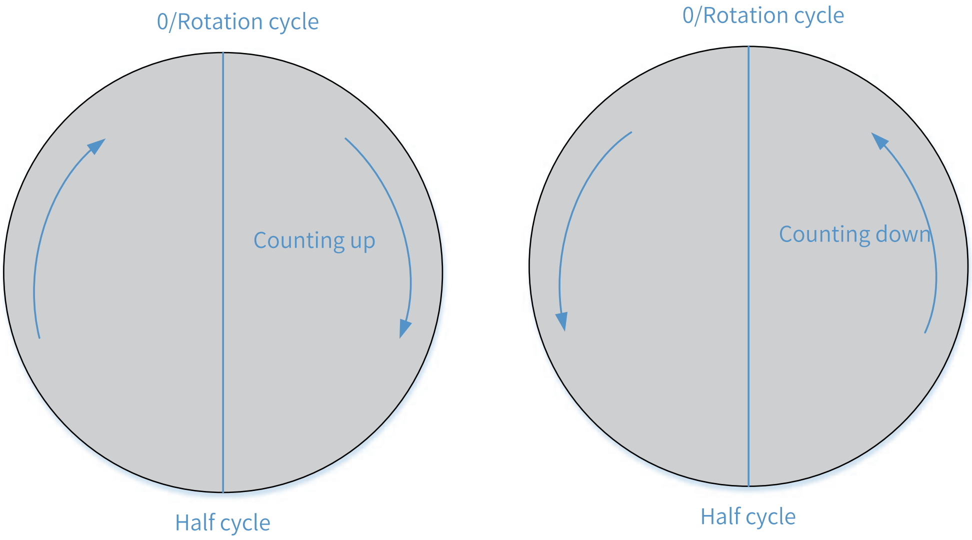
Since the high-speed counter is a 32-bit counter, the rotation cycle must be within the 32-bit integer range [–2147483648, +2147483647] after being converted to pulse units.



Mode Selection:

Mode Settings ○Linear Mode ●Rotation Mode

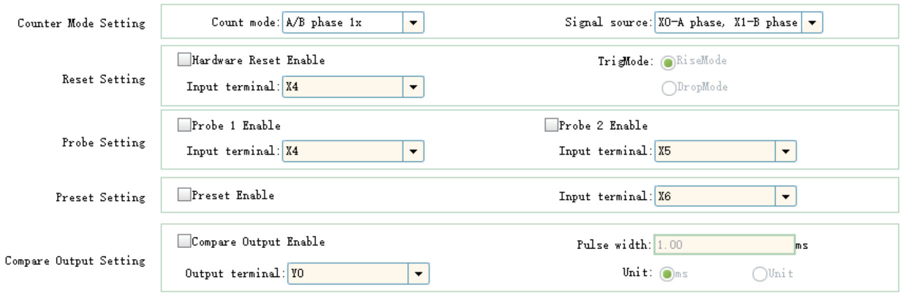
Cycle Settings :ation period: 360.00 Unit



13.5 Setting Counter Parameters

13.5.1 Overview

Parameter settings mainly involve the count mode, probes, presetting, and position output comparison function.



Counter Mode Setting
Count mode: A/B phase 1x
Signal source: X0-A phase, X1-B phase

Reset Setting
□Hardware Reset Enable
Input terminal: X4
TrigMode: ● RiseMode ○ DropMode

Probe Setting
□Probe 1 Enable
Input terminal: X4
□Probe 2 Enable
Input terminal: X5

Preset Setting
□Preset Enable
Input terminal: X6

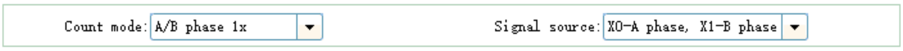
Compare Output Setting
□Compare Output Enable
Output terminal: Y0
Pulse width: 1.00 ms
Unit: ● ms ○ Unit

13.5.2 Count Modes

13.5.2.1 Overview

Local encoder axes support multiple signal counting modes, including phase A/B (frequency multiplication by 1/2/4), CW/CCW, pulse+direction, and single-phase counting.

Signal sources: Different signal sources can be selected depending on the counting mode.



Count mode: A/B phase 1x
Signal source: X0-A phase, X1-B phase

The supported input ports of signal sources in various counting modes are shown in the following table. One input port of signal source can be selected for different local encoder axes.

Port Mode	X0	X1	X2	X3	Internal 1 ms	Internal 1 μs
Phase A/B	Phase A	Phase B	Phase A	Phase B	x	x
CW/CCW	CW	CCW	CW	CCW	x	x
Pulse+direction	Pulse	Direction	Pulse	Direction	x	x
Single-phase counting	Pulse	Pulse	Pulse	Pulse	Pulse	Pulse

Note

When two input signals are required in the selected working mode, X0 and X1 make up one group of input signals, while X2 and X3 make up the other group.

The above counting modes and signal sources can be arbitrarily selected for the four counters, and repeatedly used for different counters.

13.5.2.2 Phase A/B Mode

In phase A/B mode, the encoder generates two orthogonal phase pulse signals with a phase difference of 90°, that is, phase A signal and phase B signal. When the phase A signal leads the phase B signal, the counter counts up; when the phase B signal leads the phase A signal, the counter counts down.

Phase A/B encoder wiring diagram

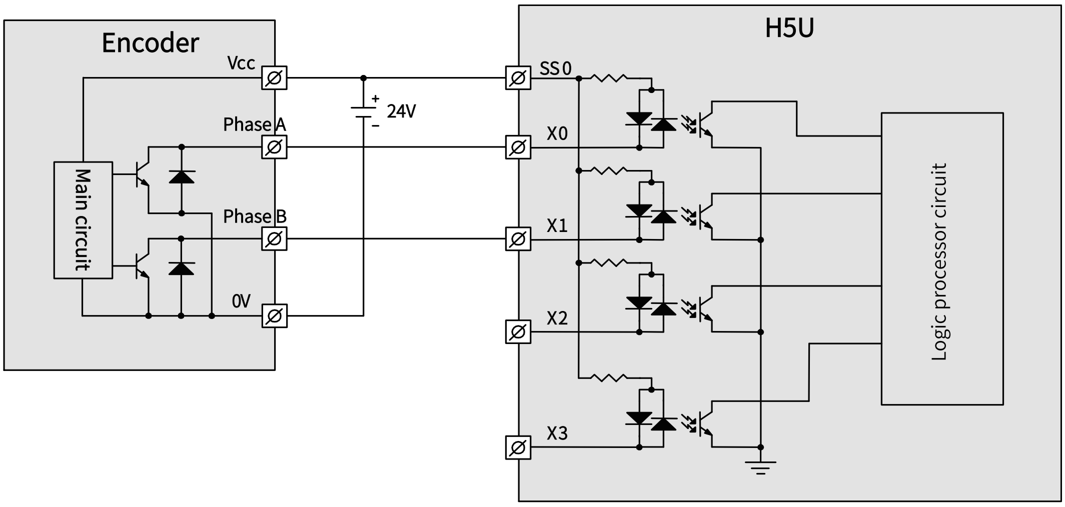


Figure 13-1 Sink Input Wiring

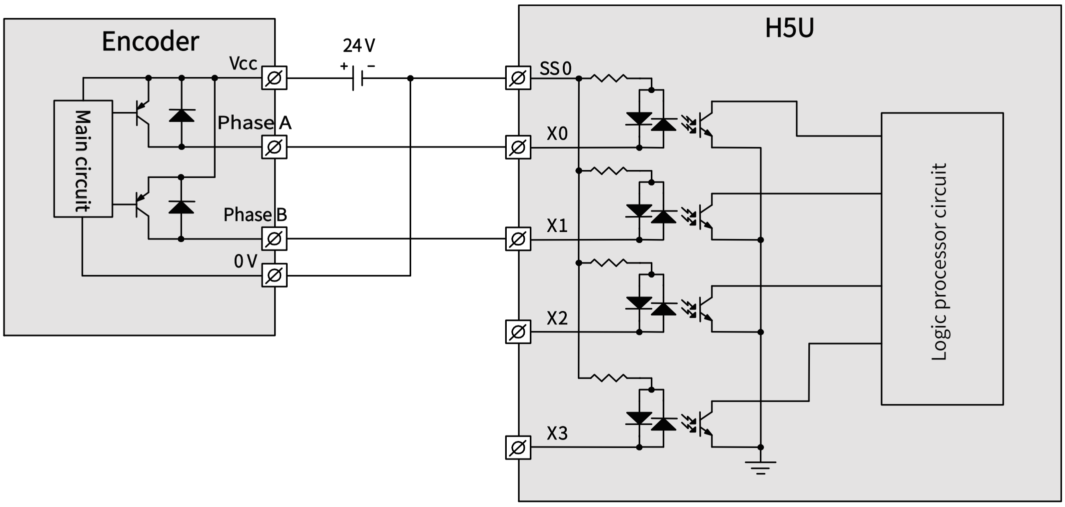
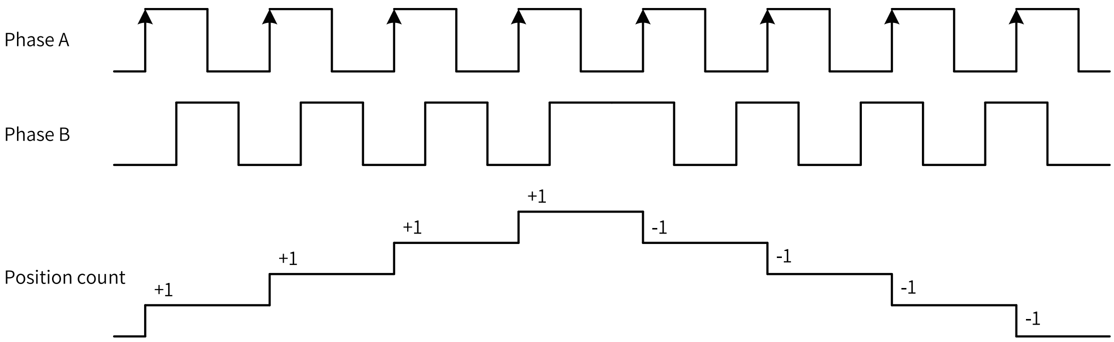


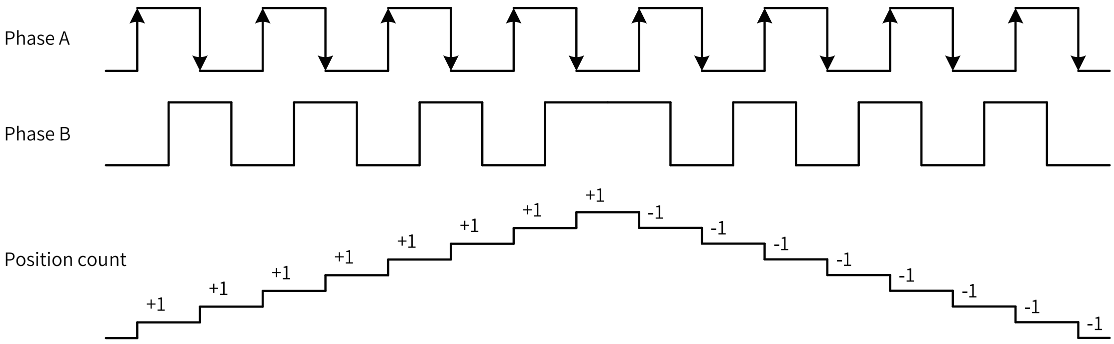
Figure 13-2 Source Input Wiring

The phase A/B pulse can be set to operate in frequency multiplication by 1, 2, or 4.

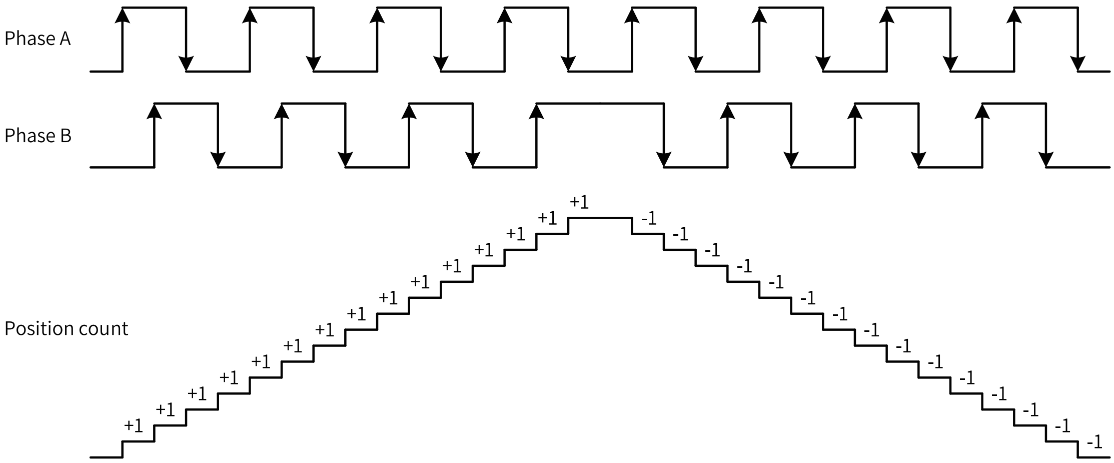
- In the mode of phase A/B frequency multiplication by 1, only the rising edge of phase A pulse is counted as shown in the following figure.



- In the mode of phase A/B frequency multiplication by 2, the rising and falling edges of phase A pulse are counted as shown in the following figure.

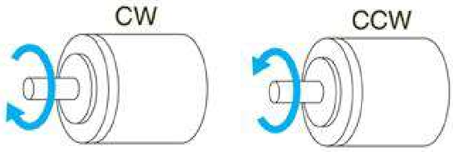


- In the mode of phase A/B frequency multiplication by 4, the rising and falling edges of the phase A pulse and the phase B pulse are counted as shown in the following figure.

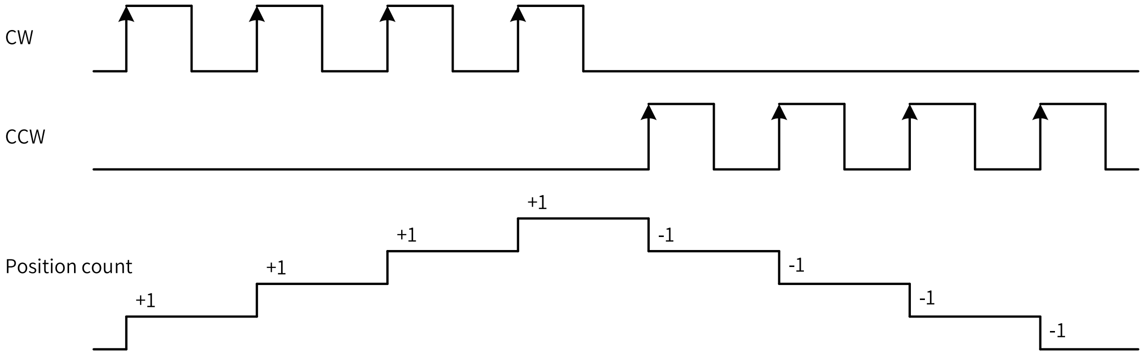


13.5.2.3 CW or CCW Mode

Clock Wise (CW) is the forward pulse signal, and Counter Clock Wise (CCW) is the reverse pulse signal. When the encoder is forward running, CW pulse signals are output; when the encoder is reverse running, CCW pulse signals are output.

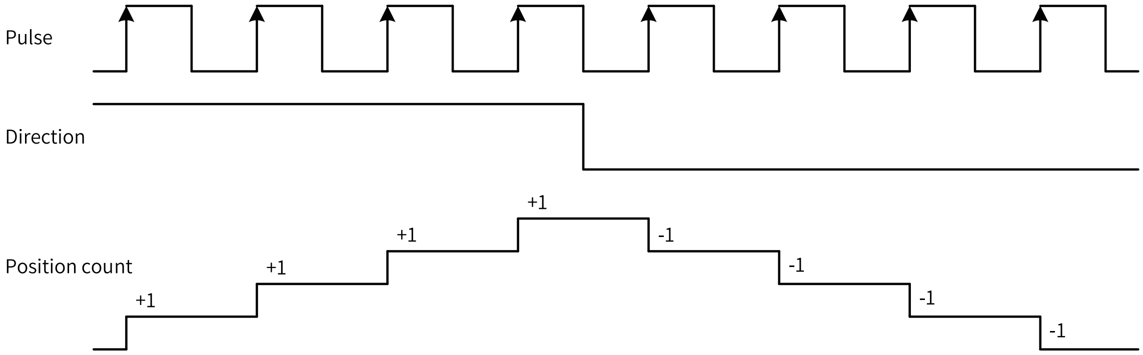


When the local encoder axis operates in this count mode, the high-speed counter counts up the CW signals and counts down the CCW signals, as shown in the following figure.



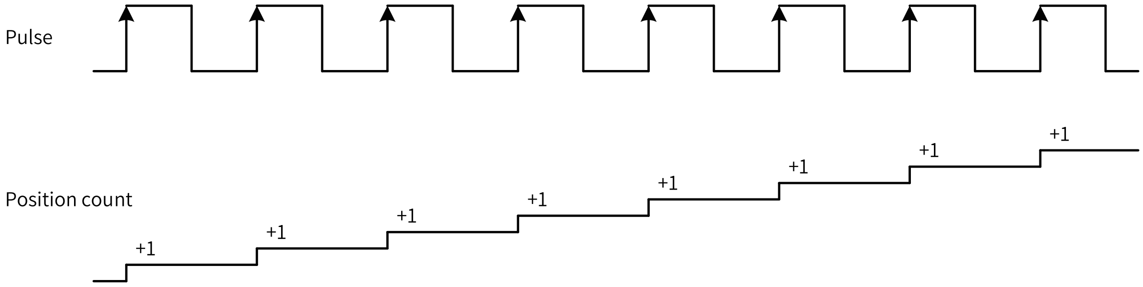
13.5.2.4 Pulse+Direction Mode

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



13.5.2.5 Single-phase Count

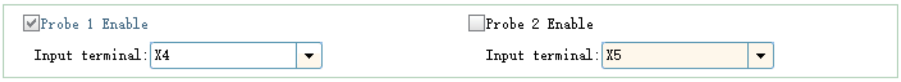
In this mode, the high-speed counter counts up the pulse signals. The position count is added with 1 when the rising edge of a pulse is input.



13.5.3 Probe Terminal Settings

Each counter supports two external inputs to latch the current value of the counter to realize the probe function. Enable the counter axis position latch of the external inputs by ticking the "Probe enable". The input terminal can be any of X0 to X7 inputs.

When the probe is enabled, read the probe position of the counter axis through the function block instruction HC_TouchProbe.



☑Probe 1 Enable

Input terminal: X4

□Probe 2 Enable

Input terminal: X5

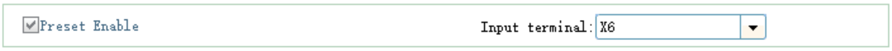
Note: For the probe functions supported by the Easy series models, see the specific model for more information.

Series	Local Encoder Axis	Local Pulse Axis
Easy301 series	Each axis has only two probes.	Dual-axis mode: 2 probes Single-axis mode: Y0, Y1, Y2, and Y3 support two probes.
Other models of the Easy series	Each axis has only one probe.	Dual-axis mode: 2 probes Single-axis mode: Probes are not supported.

13.5.4 Preset Terminal Settings

Enable the preset counter value of external inputs by ticking "Preset enable". The input terminal can be any of X0 to X7 inputs, with rising edge or falling edge as the trigger condition.

When the preset function is enabled, the encoder axis position is preset by external inputs through the function block instruction HC_Preset.



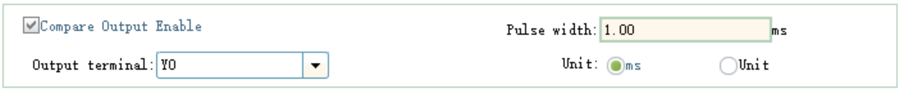
☑Preset Enable

Input terminal: X6

13.5.5 Comparison Output Terminal Settings

After "Comparison output enable" is selected, the hardware output can be realized when the positions are equal in comparison without software processing, featuring high real-time performance and microsecond-level output responses.

- After the comparison output function is enabled, in combination with function block instructions, the output controlled by the hardware circuit when the positions are equal in comparison is ON. The output terminal can be any of Y0 to Y3, and the pulse width when the output is ON can be measured in time units or user units (Unit).
- Each local encoder axis is equipped with one comparison output function, and the input terminal and output pulse width can be configured as needed.
- After configuration, the axis position comparison output is realized through the function block instructions HC_Compare, HC_ArrayCompare, and HC_StepCompare.
- When "ms" is selected as the unit, the time range for setting is 0.1 ms to 6553.5 ms. When "Unit" is selected as the unit, make sure that the set value is within the range of 1 to 65535 after being converted to pulse units.



☑Compare Output Enable

Pulse width: 1.00 ms

Output terminal: Y0

Unit: ● ms ○ Unit

The comparison output is directly output through the hardware control port, instead of software processing. Therefore, status of comparison output is not available in the Y element in the program.

The Y soft element and the comparison output control the output port in an OR relationship. If the Y element is continuously controlled to ON, the actual port output remains ON.

13.6 Counter Axis Instruction Application (H5U)

13.6.1 Overview

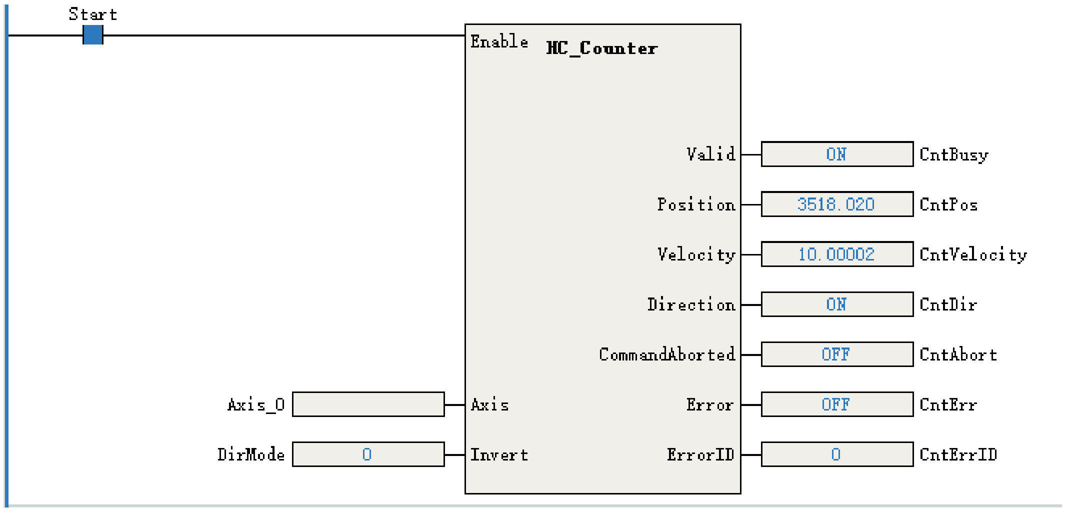
After a counter axis is set in AutoShop, the counter axis can be used in combination with function block instructions to implement functions such as axis position counting/velocity measurement, axis position presetting, and axis position latching and comparison.

13.6.2 Axis Position Count and Speed Measurement Instructions

The HC_Counter instruction can count the position and measure the velocity of the counter axis.

The counter axis position value (unit: Unit) is set according to the counter axis mode and changes within the range of the mode.

The counter axis velocity is the current real-time velocity (unit: Unit/s). The minimum velocity that can be measured by the counter axis is the velocity corresponding to 1 pulse of the counter within 1s. If 1 pulse of the counter corresponds to 0.01 Unit, the minimum velocity that can be measured is 0.01 Unit/ s.

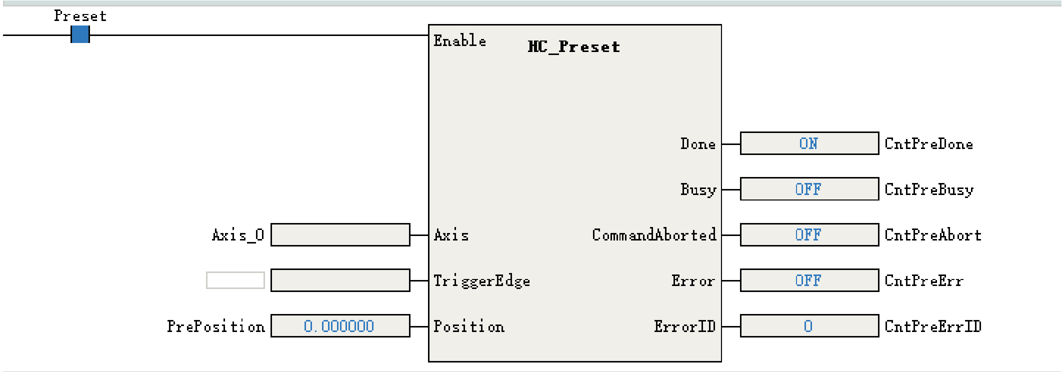


The parameter Invert in the instruction can be set to change the count direction. The modification on Invert takes effect only after this function block instruction is enabled again. The relationship between the Invert setting and the count direction is as follows.

Invert	Phase A/B	Pulse+direction	CW/CCW	Single-phase counting
0	Incremental count if phase A leads phase B Decremental count if Phase B leads Phase A	Decremental count for a low-level direction signal Incremental count for a high-level direction signal	Incremental count for phase A Decremental count for phase B	Incremental count
1	Decremental count if phase A leads phase B Incremental count if phase B leads phase A	Incremental count for a low-level direction signal Decremental count for a high-level direction signal	Decremental count for phase A Incremental count for phase B	Decremental count

13.6.3 Axis Position Preset Instructions

The instruction HC_Preset assigns the counter axis position according to the preset conditions.



The preset condition TriggerType can be set to the trigger by the rising edge of the instruction or by external X input.

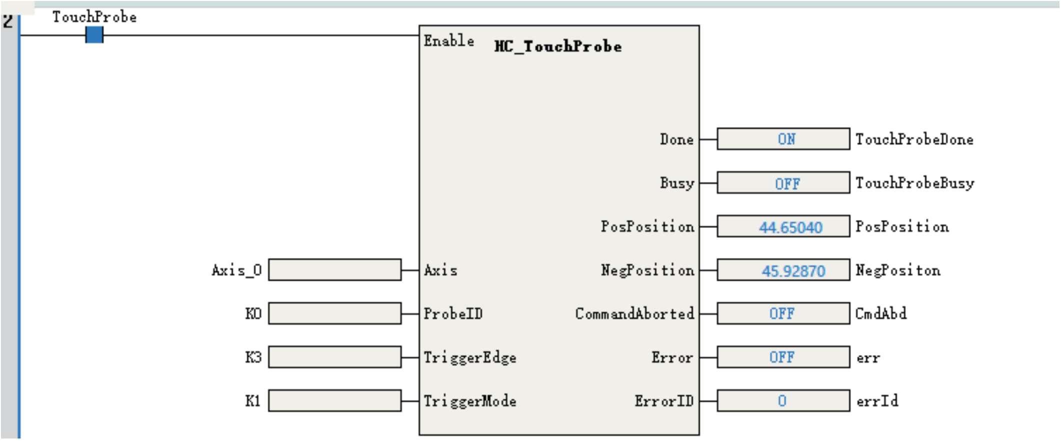
TriggerType	Definition
0	Trigger by the rising edge of the instruction flow
1	Trigger by the rising edge of the external X
2	Trigger by the falling edge of the external X
3	Trigger by the rising or falling edge of the external X

When the preset condition is set to the trigger by external X input, you need to tick "Preset function" in counter parameter settings and select the "Input terminal" and "Trigger Condition". The input terminal can be any of X0 to X7, with rising edge or falling edge as the optional trigger condition.

13.6.4 Probe Instructions

The function block instruction HC_TouchProbe can latch the counter axis position value when the external input trigger condition is valid.

Each counter axis supports two probes. During use, you need to tick the corresponding probe function in counter parameter settings and select "Input terminal" and "Trigger Condition". The input terminal can be any of X0 to X7.



The parameter ProbeID specifies the number of the probe used by the counter.

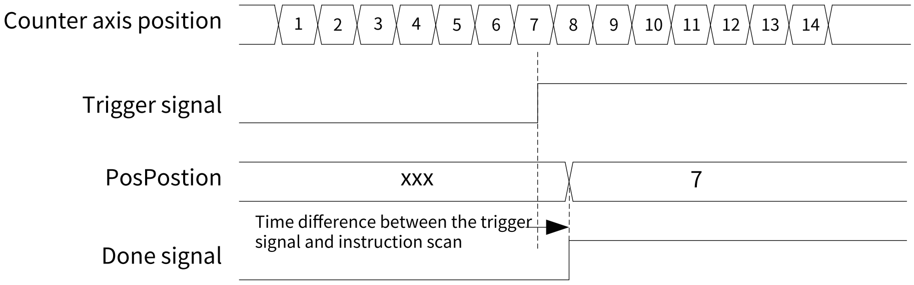
ProbeID	Definition
0	Indicates that probe 1 is used.
1	Indicates that probe 2 is used.

The parameter TriggerEdge specifies the probe trigger edge. The rising edge trigger position is latched in the output parameter PosPosition, and the falling edge trigger position is latched in the output parameter NegPosition.

TriggerEdge	Definition
1	Trigger by the rising edge of the external X
2	Trigger by the falling edge of the external X
3	Trigger by the rising or falling edge of the external X

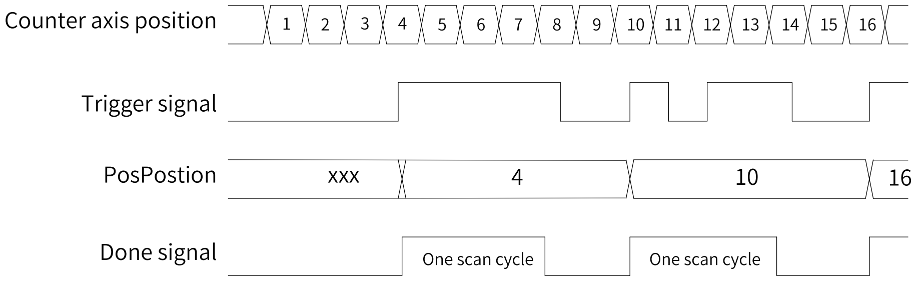
TriggerMode in the instruction can be set to the single trigger or continuous trigger.

- If the single trigger mode is used, when the function block instruction flow and the external input trigger condition are valid, the counter axis position is latched once, and the Done signal is output. The counter axis position is latched in real time based on the trigger edge, which is not affected by program execution. During instruction execution, affected by the scan cycle, when the program scans and runs to the latched instruction, it updates the latched position to the output parameter of the instruction.



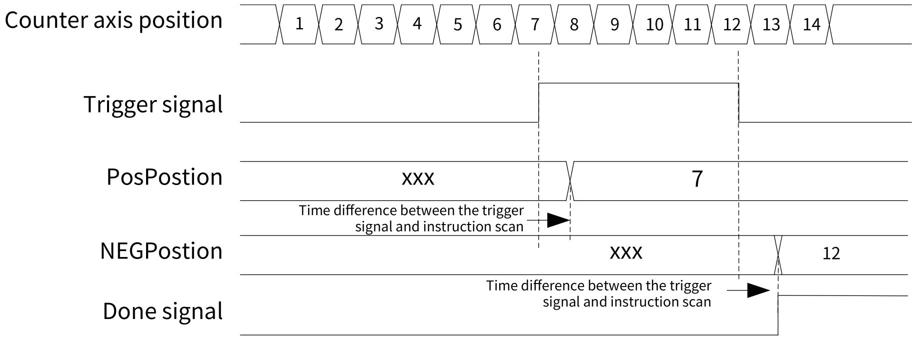
Single triggering on the rising edge

- If the continuous trigger mode is used, when the function block instruction flow and the external input trigger condition are valid, the counter axis position is latched, and the Done signal that is active for one scan cycle is output. When the Done signal becomes OFF and the external input trigger condition is valid, the counter axis position continues to be latched and the Done signal that is active for one scan cycle is output. During the scan cycle in which the Done signal is active, if the external input trigger condition is valid, the counter axis position is not latched at this time.

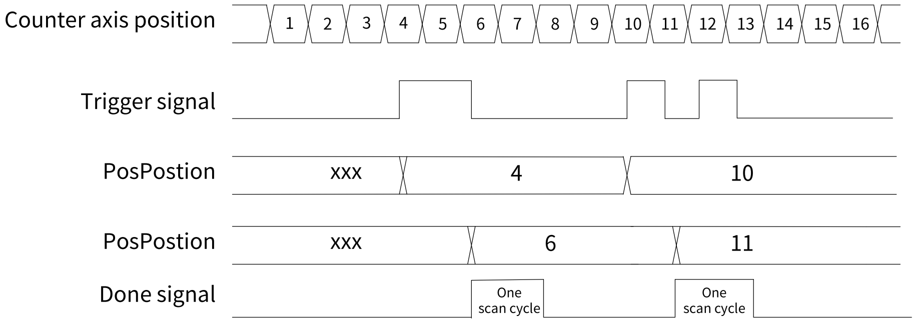


Continuous triggering on the rising edge

- When the dual-edge trigger mode is used, the Done signal is output after the instruction is triggered on both the rising and falling edges to complete the latch. In single trigger mode, the Done signal remains active until the instruction execution is completed; in continuous trigger mode, the Done signal is active for one scan cycle, and the latch signal is not responded within the scan cycle when the Done signal is active.


Counter axis position
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Trigger signal
PosPostion
xxx
7
Time difference between the trigger signal and instruction scan
NEGPostion
xxx
12
Time difference between the trigger signal and instruction scan
Done signal

Single triggering on both the rising and falling edges



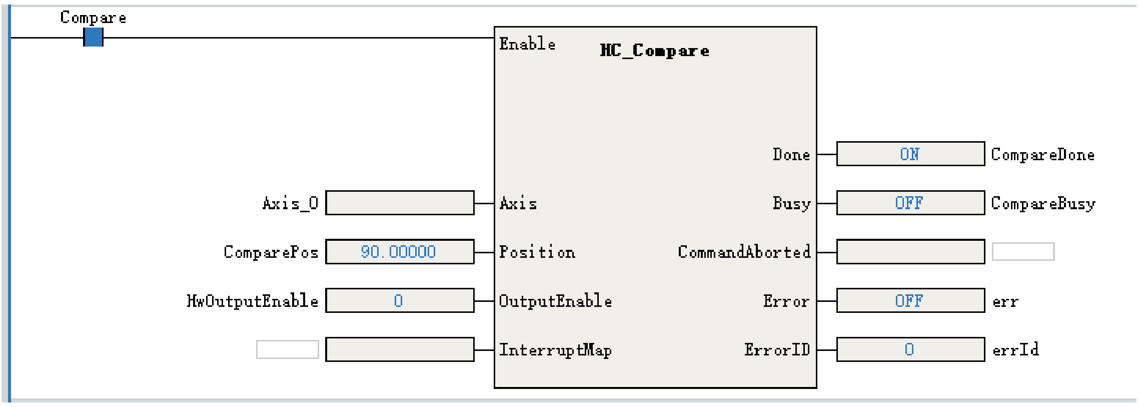
Continuous triggering on both the rising and falling edges

13.6.5 Comparison Instructions

The instructions HC_Compare, HC_StepCompare, and HC_ArrayCompare can compare the counter axis position with a single position, equally-spaced positions continuously, or multiple positions continuously.

HC_Compare

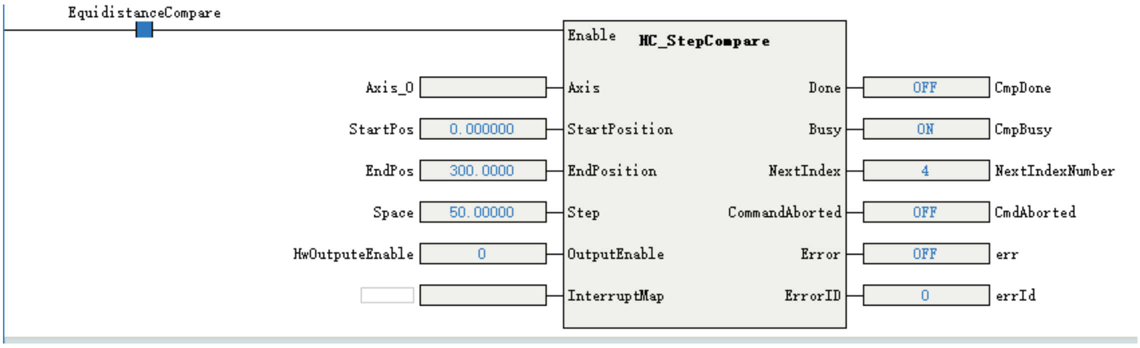
The instruction compares the counter axis position with a single position. When the instruction flow is active, the Done signal is output after the counter axis position reaches the comparison position.

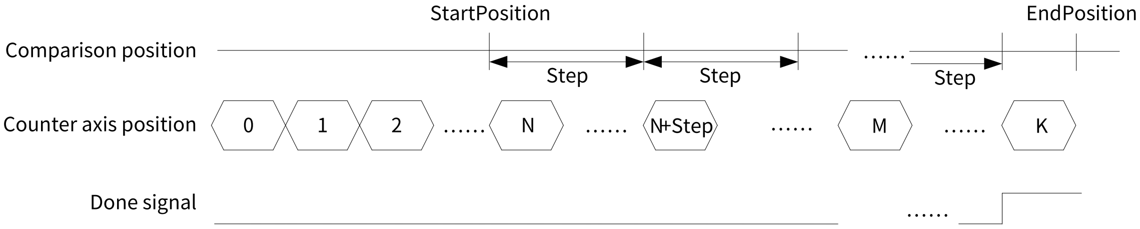


HC_StepCompare

This instruction compares the counter axis position with equally-spaced positions continuously. When the instruction flow is active, the counter axis position is compared with the position specified by StartPosition. When they are equal, the comparison position increases or decreases by a value specified by Step and then is compared with the counter axis position. The Done signal of one cycle is not output after they are equal in each comparison, but after the last comparison position is compared.

- If the StartPosition is less than the EndPostion, when the positions are equal in each comparison, the comparison position increases by a value specified by Step. When the current comparison position added with the value specified by Step is greater than the EndPosition, the current comparison position is the last one.
- If the StartPosition is greater than the EndPostion, when the positions are equal in each comparison, the comparison position decreases by a value specified by Step. When the current comparison position minus the value specified by Step is less than the EndPosition, the current comparison position is the last one.
- The output parameter NextIndex indicates the index of the next comparison point, that is, the number of completed comparison points.

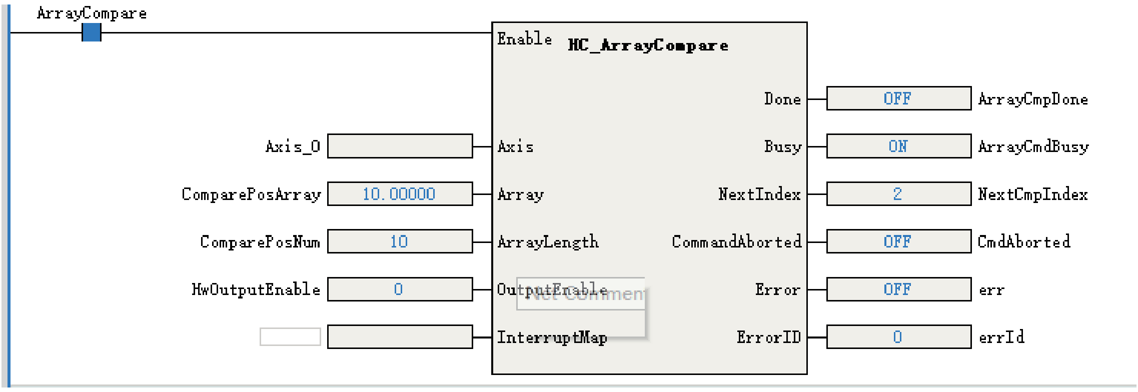




HC_ArrayCompare

This instruction compares the counter axis position with multiple positions in an array continuously. When the instruction flow is active, the counter axis position is compared with the first position in the array. If they are equal, the counter axis position is compared with the next position value in the array. After the last comparison position is compared, the Done signal is output.

- ArrayLength in the instruction specifies the array length. After all the positions in the array are compared, the Done signal is continuously output and the comparison with multiple positions is completed.
- The output parameter NextIndex indicates the index of the next comparison point, that is, the number of completed comparison points.



ArrayCompare

HC_ArrayCompare

Enable

Axis_0 — Axis

ComparePosArray: 10.00000 — Array

ComparePosNum: 10 — ArrayLength

HwOutputEnable: 0 — OutputEnable

InterruptMap

Done: OFF — ArrayCmpDone

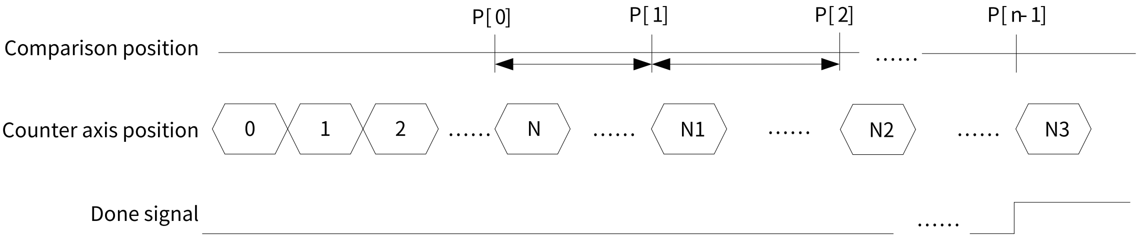
Busy: ON — ArrayCmdBusy

NextIndex: 2 — NextCmpIndex

CommandAborted: OFF — CmdAborted

Error: OFF — err

ErrorID: 0 — errId



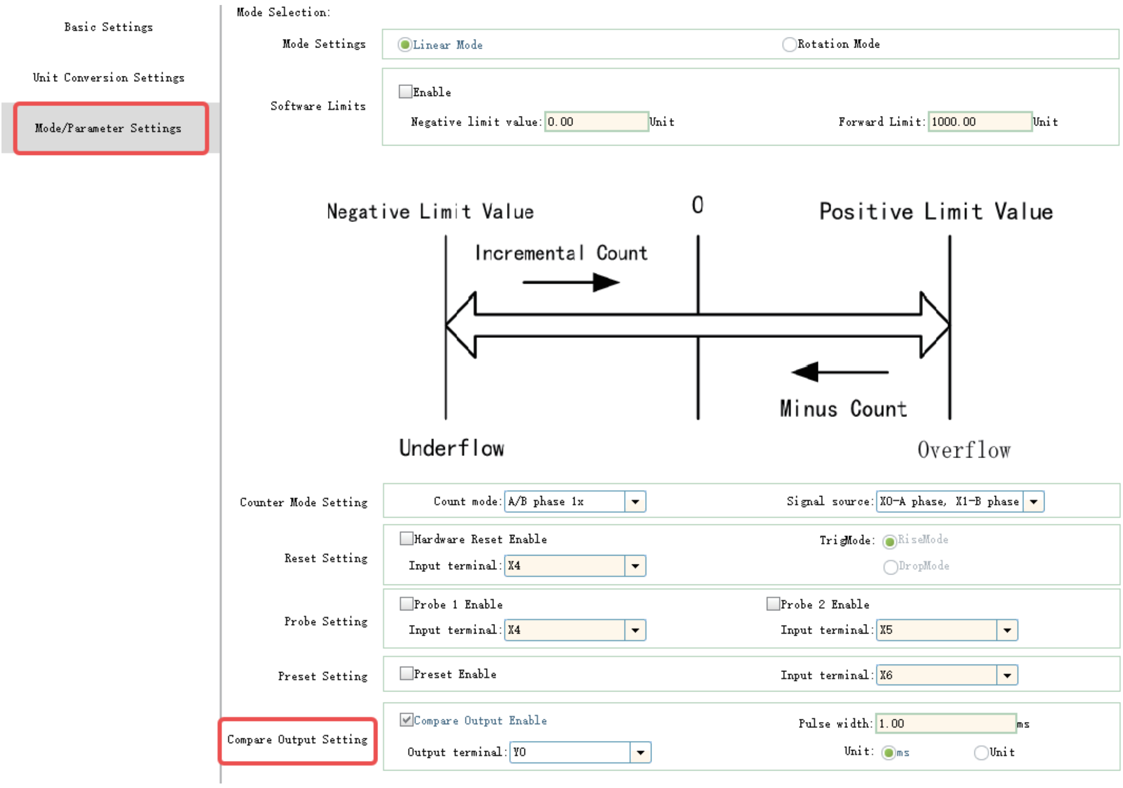
13.6.6 High-speed Hardware Comparison Output

The counter axis can realize the position comparison hardware output. When the counter axis and the comparison position are equal, the output directly controlled by the hardware circuit is ON, and the output delay is less than 1 μs.

Setting the comparison output function of the counter axis

In the parameter setting interface of the counter axis, tick "Comparison output enable".

The output terminal can be any of Y0 to Y3. The output width is set as the pulse width when the output is ON, for which the unit can be time unit or user unit (Unit).



Basic Settings

Unit Conversion Settings

Mode/Parameter Settings

Mode Selection:

Mode Settings: ● Linear Mode ○ Rotation Mode

Software Limits: □ Enable
Negative limit value: 0.00 Unit   Forward Limit: 1000.00 Unit

Counter Mode Setting: Count mode: A/B phase 1x   Signal source: X0-A phase, X1-B phase

Reset Setting: □ Hardware Reset Enable   TrigMode: ● RiseMode ○ DropMode
Input terminal: X4

Probe Setting: □ Probe 1 Enable   □ Probe 2 Enable
Input terminal: X4   Input terminal: X5

Preset Setting: □ Preset Enable   Input terminal: X6

Compare Output Setting: ☑ Compare Output Enable   Pulse width: 1.00 ms
Output terminal: Y0   Unit: ● ms ○ Unit

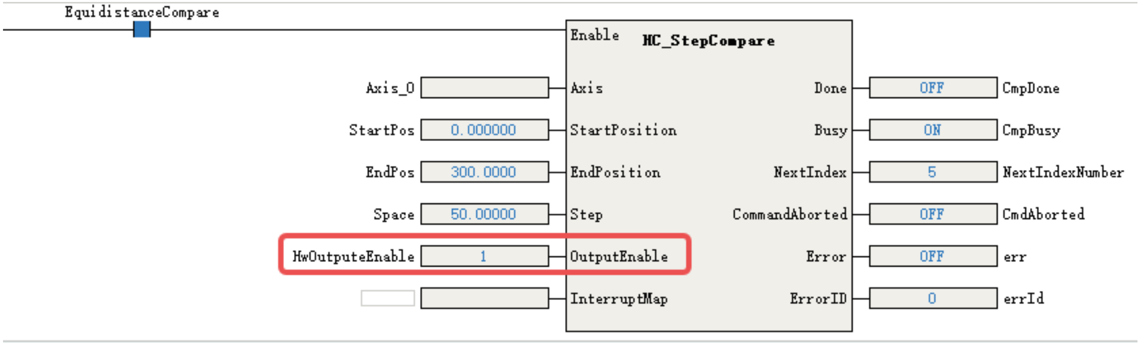
Note

When the unit of pulse output width is set to time, the time accuracy of pulse output width is 100 µs, and the maximum output width is 6500 ms. When the unit of pulse output width is set to user unit (Unit), the maximum output width is equivalent to 65535 pulses.

Enabling the OutputEnable parameter in the comparison instructions

Use MC_Compare, MC_StepCompare, and MC_ArrayCompare comparison instructions to set OutputEnable to 1, that is, associate hardware outputs when the instruction executes the comparison for equality.

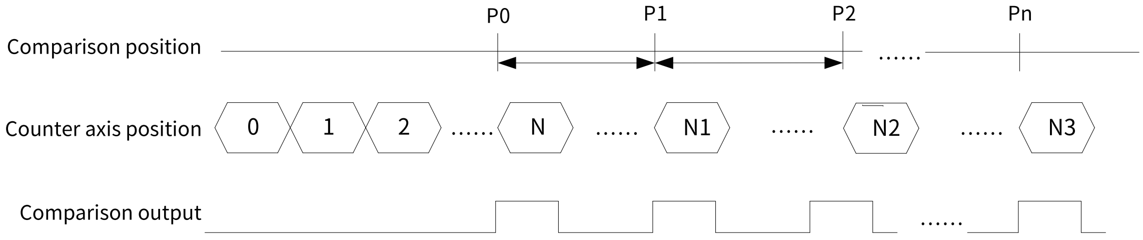
When the instruction executes the comparison for equality, the output terminal set directly through the control of hardware circuit is ON, and the output turns OFF after the continuous width output.



EquidistanceCompare

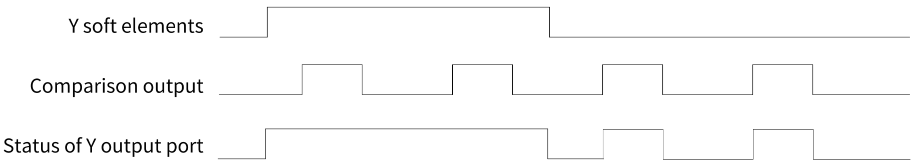
MC_StepCompare

	Enable		
Axis_0	Axis	Done	OFF CmpDone
StartPos 0.000000	StartPosition	Busy	ON CmpBusy
EndPos 300.0000	EndPosition	NextIndex	5 NextIndexNumber
Space 50.00000	Step	CommandAborted	OFF CmdAborted
HwOutputeEnable 1	OutputEnable	Error	OFF err
	InterruptMap	ErrorID	0 errId



Note

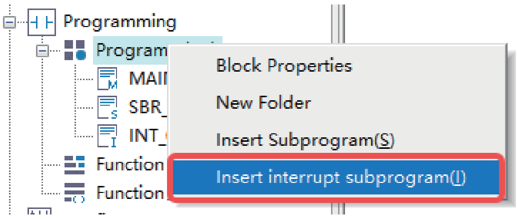
High-speed comparison hardware output is directly output through the hardware control port. Therefore, status of comparison output is not available in the Y element in the program. The Y element and the comparison output control the output port in an OR relationship. If the Y element is continuously controlled as ON, the actual port output remains ON.



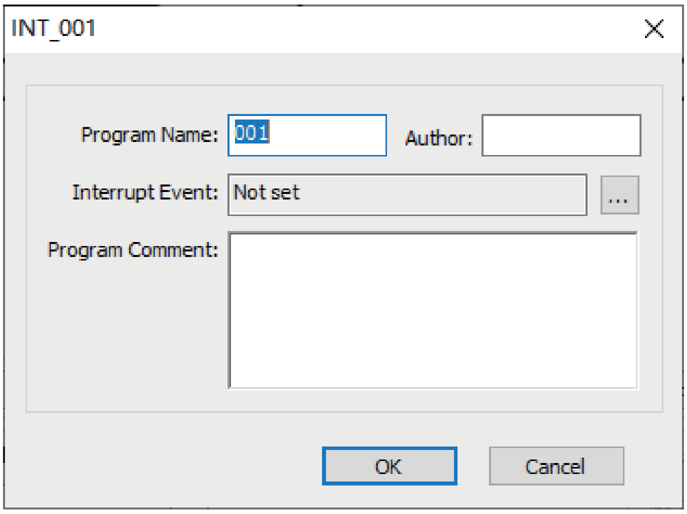
13.6.7 Comparison Interruption

When the counter axes are compared for equality, comparison interruption can be associated and the interrupt subprogram can be executed. The operation steps are as follows.

1. Under the item "Programming" in "Project Manager", right-click the "POU", and select "Insert interrupt subprogram".



2. Right-click the inserted interrupt subprogram (such as INT_001 in the figure above) and select "Properties" to open the interrupt subprogram settings page as shown in the following figure.


INT_001

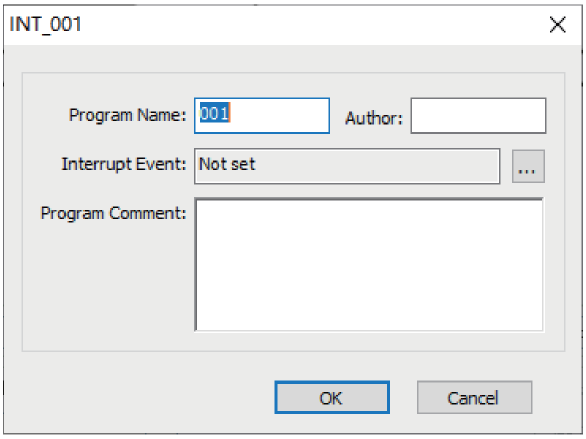
Program Name: 001

Author:

Interrupt Event: Not set

Program Comment:

3. Click the ... icon after the "Interrupt Event" field, select "Comparison interrupt", and then write interrupt subprograms in INT_001.

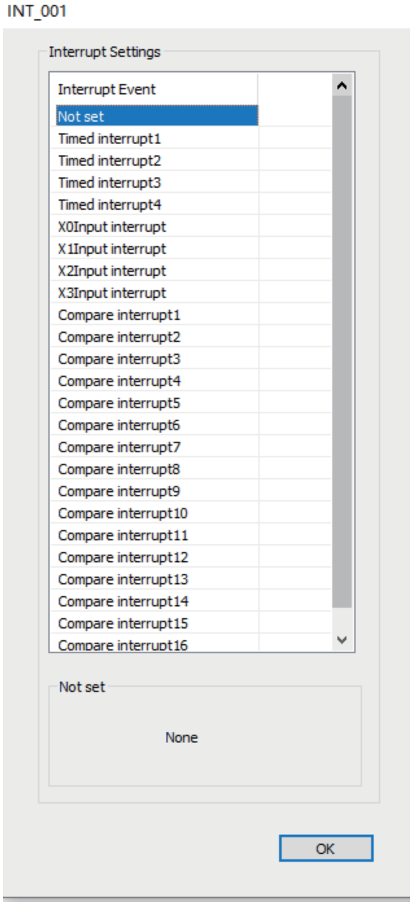


INT_001

Program Name: 001 Author:

Interrupt Event: Not set

Program Comment:

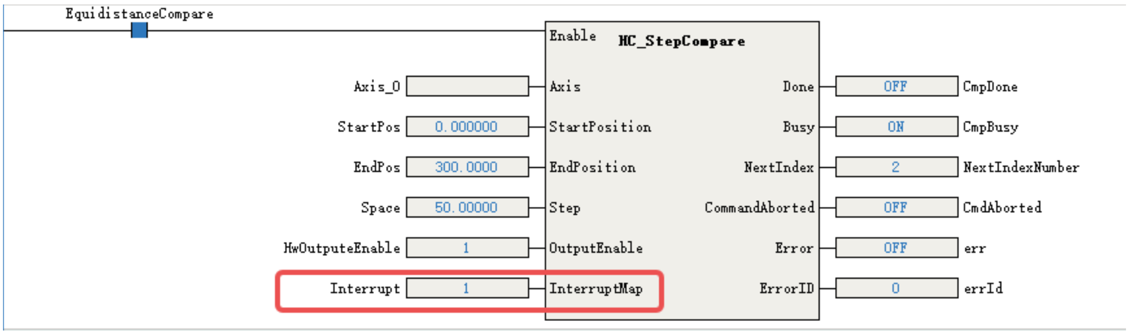


INT_001

Interrupt Settings

Interrupt Event: Not set

4. Call MC_Compare, MC_StepCompare, and MC_ArrayCompare instructions in the main program or subprogram to associate the parameter InterruptMap with the comparison interruption number, that is, setting the parameter InterruptMap as the comparison interruption number. EI is enabled in the program, and when the instruction executes the comparison for equality, the comparison of corresponding interrupt subprograms will be triggered.



EquidistanceCompare

MC_StepCompare

		Enable				
Axis_0		Axis	Done	OFF	CmpDone
StartPos	0.000000	StartPosition	Busy	ON	CmpBusy
EndPos	300.0000	EndPosition	NextIndex	2	NextIndexNumber
Space	50.00000	Step	CommandAborted	OFF	CmdAborted
HwOutputeEnable	1	OutputEnable	Error	OFF	err
Interrupt	1	InterruptMap	ErrorID	0	errId

13.7 Counter Axis Instruction Application (Easy)

13.7.1 Overview

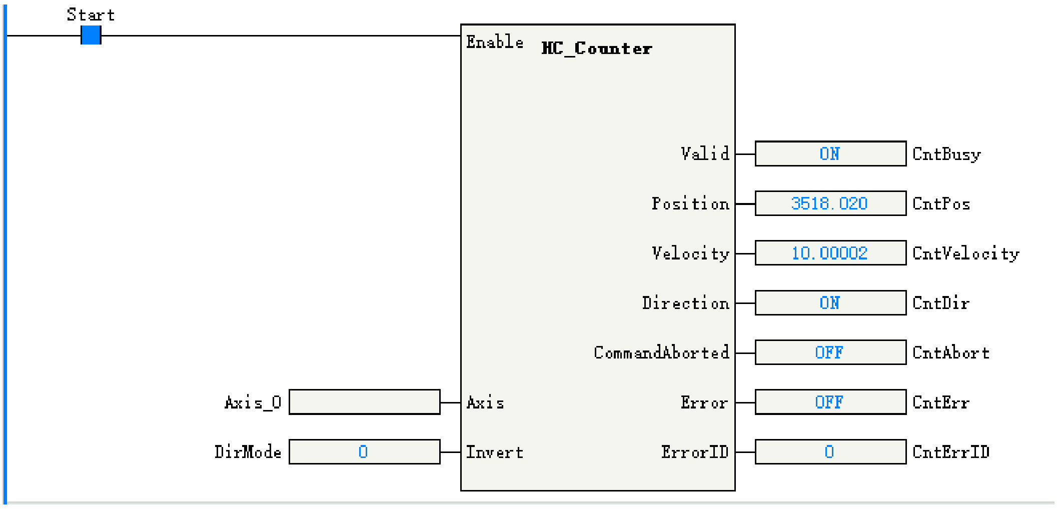
After a counter axis is set in AutoShop, the counter axis can be used in combination with function block instructions to implement functions such as axis position counting/velocity measurement, axis position presetting, and axis position latching and comparison.

13.7.2 Axis Position Count and Speed Measurement Instructions

The instruction ENC_Counter implements position counting and velocity measurement of the counter axis.

The counter axis position value (unit: Unit) is set according to the counter axis mode and changes within the range of the mode.

The counter axis velocity is the current real-time velocity (unit: Unit/s). The minimum velocity that can be measured by the counter axis is the velocity corresponding to 1 pulse of the counter within 1s. If 1 pulse of the counter corresponds to 0.01 Unit, the minimum velocity that can be measured is 0.01 Unit/s.



Start

Enable HC_Counter

Valid ON CntBusy

Position 3518.020 CntPos

Velocity 10.00002 CntVelocity

Direction ON CntDir

CommandAborted OFF CntAbort

Axis_0 Axis Error OFF CntErr

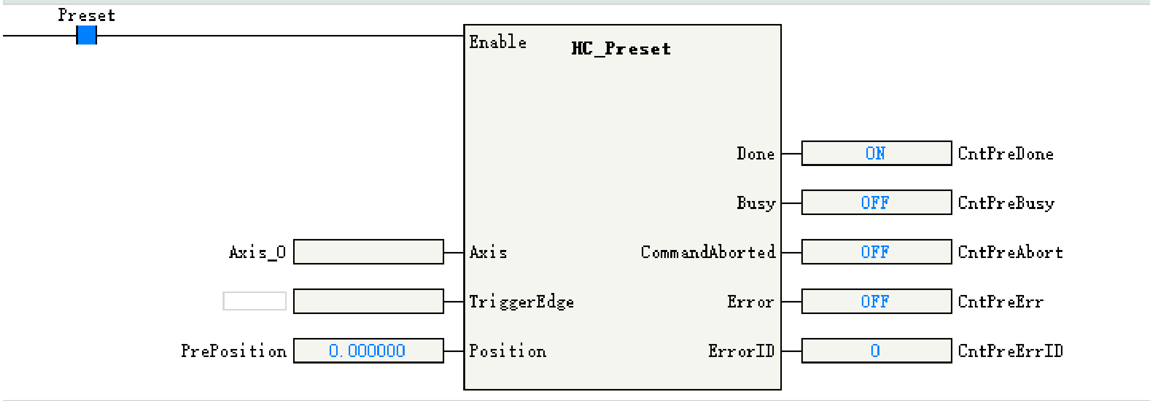
DirMode 0 Invert ErrorID 0 CntErrID

The parameter Direction in the instruction can be set to change the count direction. The modification on Direction takes effect only after this function block instruction is enabled again. The relationship between the Direction setting and the count direction is as follows.

Direction	Phase A/B	Pulse+direction	CW/CCW	Single-phase counting
0	Incremental count if phase A leads phase B Decremental count if phase B leads phase A	Decremental count for a low-level direction signal Incremental count for a high-level direction signal	Incremental count for phase A Decremental count for phase B	Incremental count
1	Decremental count if phase A leads phase B Incremental count if phase B leads phase A	Incremental count for a low-level direction signal Decremental count for a high-level direction signal	Decremental count for phase A Incremental count for phase B	Decremental count

13.7.3 Axis Position Preset Instructions

The instruction ENC_Preset assigns the counter axis position according to the preset conditions.



The preset condition TrigerMode can be set to the trigger by the rising edge of the instruction or by external X input.

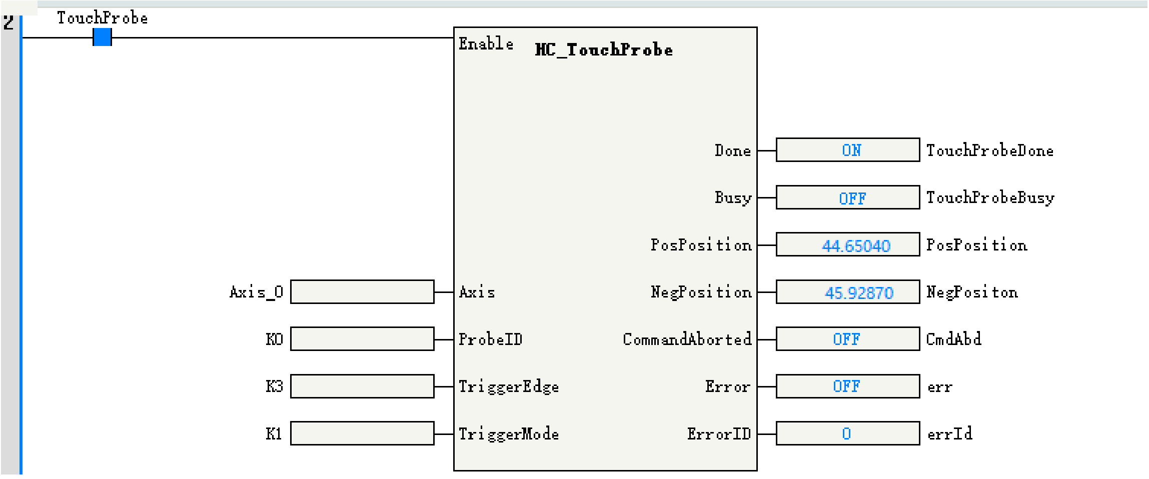
TrigerMode	Definition
0	Trigger by the rising edge of the instruction flow
1	Trigger by the rising edge of the external X
2	Trigger by the falling edge of the external X
3	Trigger by the rising or falling edge of the external X

When the preset condition is set to the trigger by external X input, you need to tick "Preset function" in counter parameter settings and select the "Input terminal" and "Trigger Condition". The input terminal can be any of X0 to X7, with rising edge or falling edge as the optional trigger condition.

13.7.4 Probe Instructions

The function block instruction ENC_TouchProbe can latch the counter axis position value when the external input trigger condition is valid.

Each counter axis supports two probes. During use, you need to tick the corresponding probe function in counter parameter settings and select "Input terminal" and "Trigger Condition". The input terminal can be any of X0 to X7.



The parameter ProbeID specifies the number of the probe used by the counter.

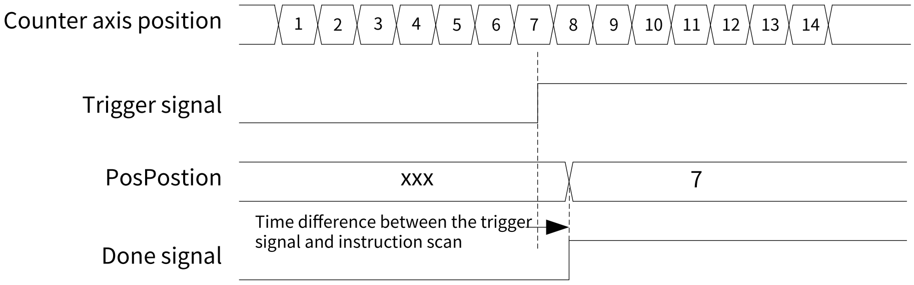
ProbeID	Definition
0	Indicates that probe 1 is used.

The parameter TriggerEdge specifies the probe trigger edge. The rising edge trigger position is latched in the output parameter PosPosition, and the falling edge trigger position is latched in the output parameter NegPosition.

TriggerEdge	Definition
0	Trigger by the rising edge of the external X
1	Trigger by the falling edge of the external X
2	Trigger by the rising or falling edge of the external X

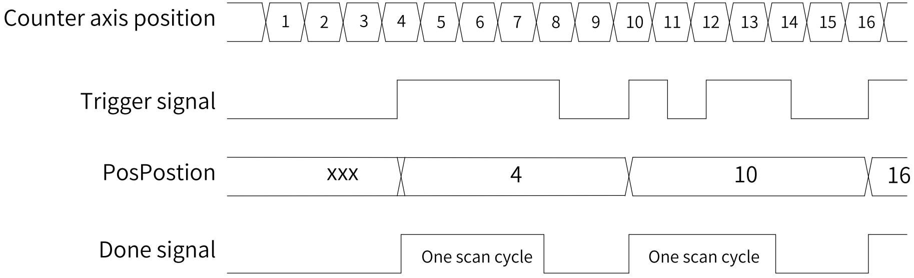
TriggerMode in the instruction can be set to the single trigger or continuous trigger.

- If the single trigger mode is used, when the function block instruction flow and the external input trigger condition are valid, the counter axis position is latched once, and the Done signal is output. The counter axis position is latched in real time based on the trigger edge, which is not affected by program execution. During instruction execution, affected by the scan cycle, when the program scans and runs to the latched instruction, it updates the latched position to the output parameter of the instruction.



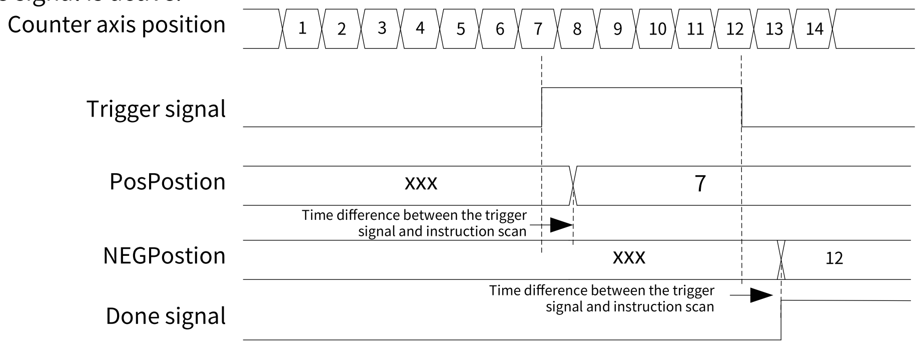
Single triggering on the rising edge

- If the continuous trigger mode is used, when the function block instruction flow and the external input trigger condition are valid, the counter axis position is latched, and the Done signal that is active for one scan cycle is output. When the Done signal becomes OFF and the external input trigger condition is valid, the counter axis position continues to be latched and the Done signal that is active for one scan cycle is output. During the scan cycle in which the Done signal is active, if the external input trigger condition is valid, the counter axis position is not latched at this time.

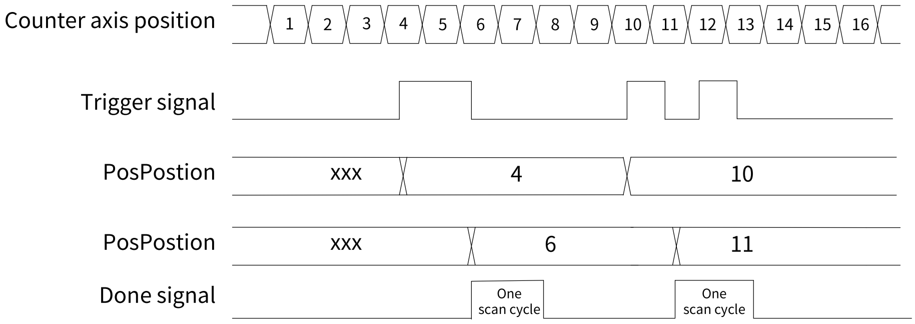


Continuous triggering on the rising edge

- When the dual-edge trigger mode is used, the Done signal is output after the instruction is triggered on both the rising and falling edges to complete the latch. In single trigger mode, the Done signal remains active until the instruction execution is completed; in continuous trigger mode, the Done signal is active for one scan cycle, and the latch signal is not responded within the scan cycle when the Done signal is active.


Counter axis position
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Trigger signal
PosPostion
XXX
7
Time difference between the trigger signal and instruction scan
NEGPostion
XXX
12
Time difference between the trigger signal and instruction scan
Done signal

Single triggering on both the rising and falling edges



Continuous triggering on both the rising and falling edges

When WindowOnly is set to ON, the window setting is enabled and the probe is only active if the encoder axis is in the window confined by FirstPosition and LastPosition.

13.7.5 Comparison Instructions

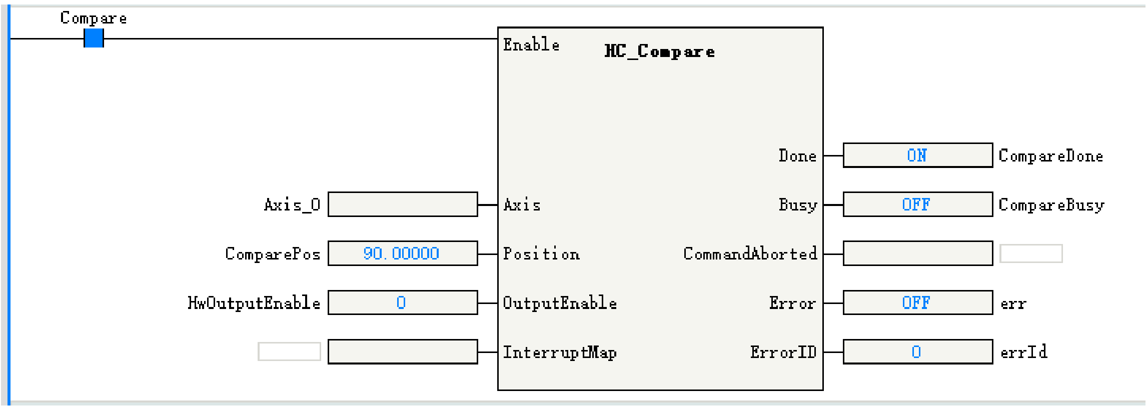
The instructions ENC_Compare, ENC_StepCompare, and ENC_ArrayCompare can compare the counter axis position with a single position, equally-spaced positions continuously, or multiple positions continuously.

ENC_Compare

The instruction compares the counter axis position with a single position. When the instruction flow is active, the Done signal is output after the counter axis position reaches the comparison position.

ENC_StepCompare

This instruction compares the counter axis position with equally-spaced positions continuously. When the instruction flow is active, the counter axis position is compared with the position specified by StartPosition. When they are equal, the comparison position increases or decreases by a value specified by Step and then is compared with the counter axis position. The Done signal of one cycle is not output after they are equal in each comparison, but after the last comparison position is compared.


Compare

HC_Compare

Enable

Axis_0 — Axis

ComparePos 90.00000 — Position

HwOutputEnable 0 — OutputEnable

InterruptMap

Done — ON CompareDone

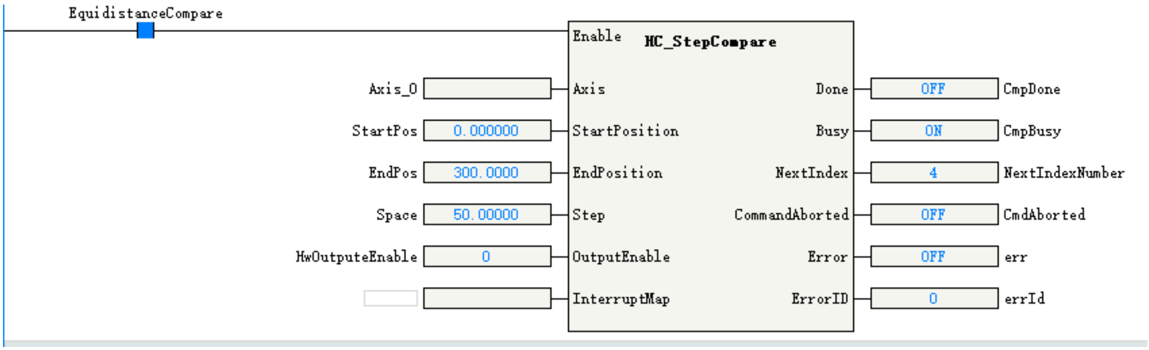
Busy — OFF CompareBusy

CommandAborted

Error — OFF err

ErrorID — 0 errId

- If the StartPosition is less than the EndPostion, when the positions are equal in each comparison, the comparison position increases by a value specified by Step. When the current comparison position added with the value specified by Step is greater than the EndPosition, the current comparison position is the last one.
- If the StartPosition is greater than the EndPostion, when the positions are equal in each comparison, the comparison position decreases by a value specified by Step. When the current comparison position minus the value specified by Step is less than the EndPosition, the current comparison position is the last one.


EquidistanceCompare

HC_StepCompare

Enable

Axis_0 — Axis

StartPos 0.000000 — StartPosition

EndPos 300.0000 — EndPosition

Space 50.00000 — Step

HwOutputeEnable 0 — OutputEnable

InterruptMap

Done — OFF CmpDone

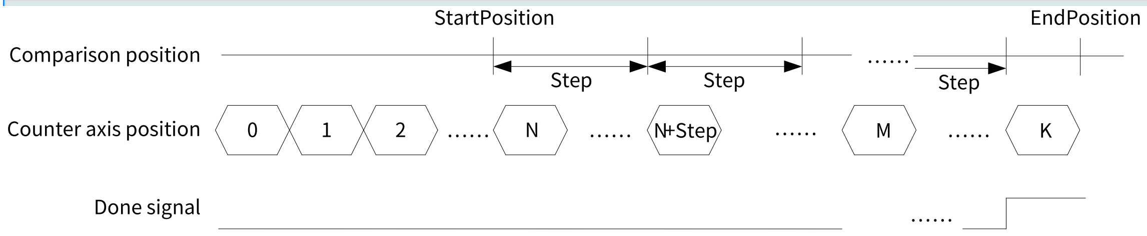
Busy — ON CmpBusy

NextIndex — 4 NextIndexNumber

CommandAborted — OFF CmdAborted

Error — OFF err

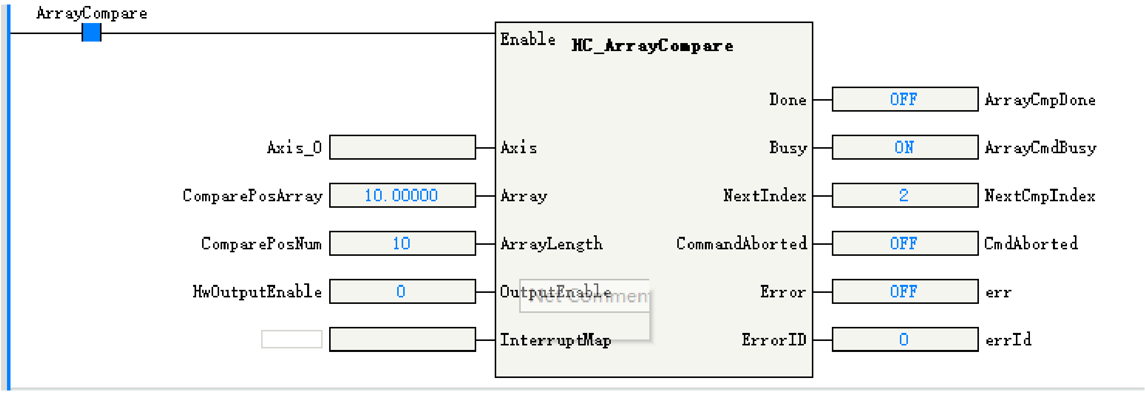
ErrorID — 0 errId



ENC_ArrayCompare

This instruction compares the counter axis position with multiple positions in an array continuously. When the instruction flow is active, the counter axis position is compared with the first position in the array. If they are equal, the counter axis position is compared with the next position value in the array. After the last comparison position is compared, the Done signal is output.

- Size in the instruction specifies the array length. After all the positions in the array are compared, the Done signal is continuously output and the comparison with multiple positions is completed.
- The output parameter Index indicates the index of the next comparison point, that is, the number of completed comparison points.



ArrayCompare

HC_ArrayCompare

Enable

Axis_0 — Axis

ComparePosArray — 10.00000 — Array

ComparePosNum — 10 — ArrayLength

HwOutputEnable — 0 — OutputEnable

InterruptMap

Done — OFF — ArrayCmpDone

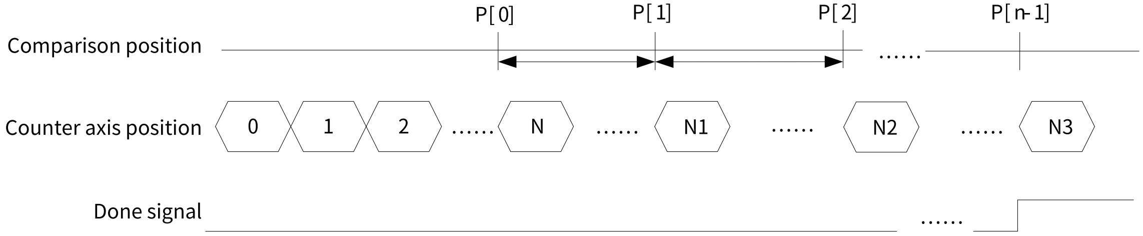
Busy — ON — ArrayCmdBusy

NextIndex — 2 — NextCmpIndex

CommandAborted — OFF — CmdAborted

Error — OFF — err

ErrorID — 0 — errId



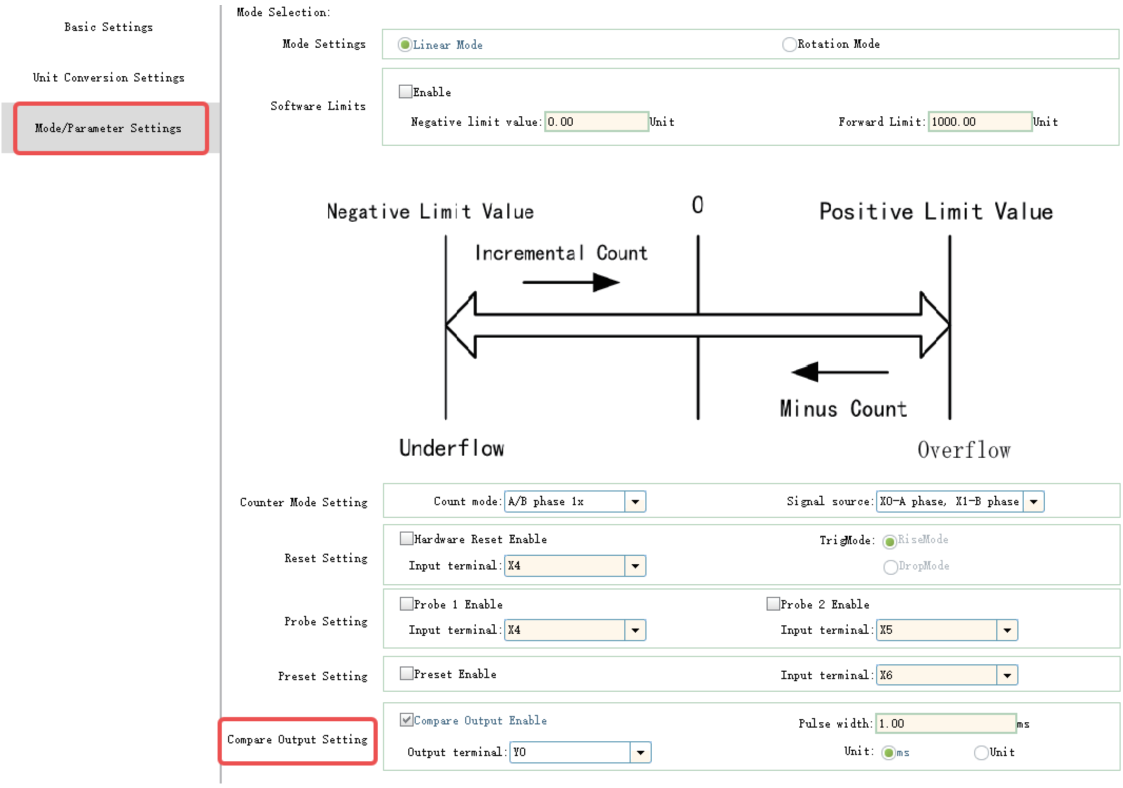
13.7.6 High-speed Hardware Comparison Output

The counter axis can realize the position comparison hardware output. When the counter axis and the comparison position are equal, the output directly controlled by the hardware circuit is ON, and the output delay is less than 1 μs.

Setting the comparison output function of the counter axis

In the parameter setting interface of the counter axis, tick "Comparison output enable".

The output terminal can be any of Y0 to Y3. The output width is set as the pulse width when the output is ON, for which the unit can be time unit or user unit (Unit).



Basic Settings

Unit Conversion Settings

Mode/Parameter Settings

Mode Selection:

Mode Settings: ● Linear Mode ○ Rotation Mode

Software Limits: □ Enable

Negative limit value: 0.00 Unit Forward Limit: 1000.00 Unit

Counter Mode Setting: Count mode: A/B phase 1x ▾ Signal source: X0-A phase, X1-B phase ▾

Reset Setting: □ Hardware Reset Enable Input terminal: X4 ▾ TrigMode: ● RiseMode ○ DropMode

Probe Setting: □ Probe 1 Enable Input terminal: X4 ▾ □ Probe 2 Enable Input terminal: X5 ▾

Preset Setting: □ Preset Enable Input terminal: X6 ▾

Compare Output Setting: ☑ Compare Output Enable Output terminal: Y0 ▾ Pulse width: 1.00 ms Unit: ● ms ○ Unit

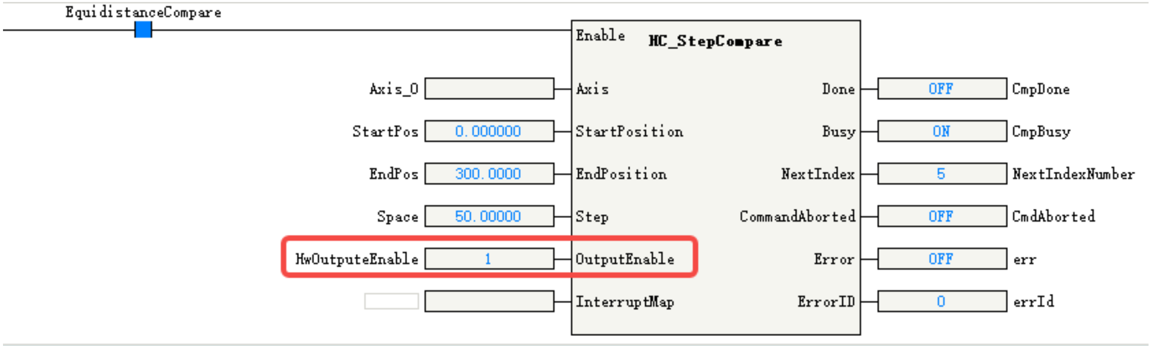
Note

When the unit of pulse output width is set to time, the time accuracy of pulse output width is 100 µs, and the maximum output width is 6500 ms. When the unit of pulse output width is set to user unit (Unit), the maximum output width is equivalent to 65535 pulses.

Enabling the OutputEnable parameter in the comparison instructions

Use ENC_Compare, ENC_StepCompare, and ENC_ArrayCompare comparison instructions to set OutputEnable to 1, that is, associate hardware outputs when the instruction executes the comparison for equality.

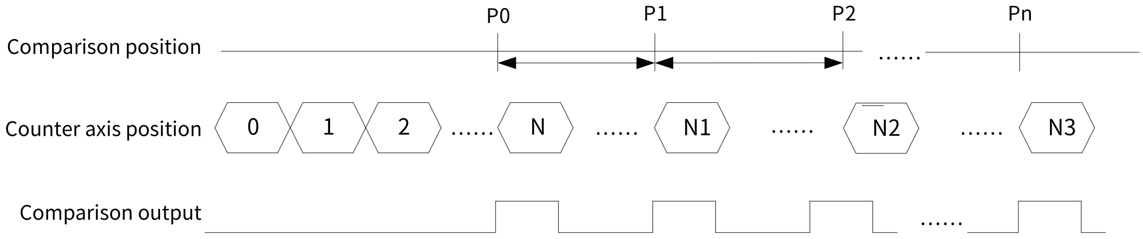
When the instruction executes the comparison for equality, the output terminal set directly through the control of hardware circuit is ON, and the output turns OFF after the continuous width output.



EquidistanceCompare

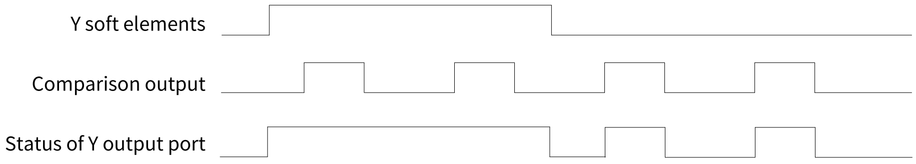
HC_StepCompare

Variable	Value	Input	Output	Value	Variable
		Enable			
Axis_0		Axis	Done	OFF	CmpDone
StartPos	0.000000	StartPosition	Busy	ON	CmpBusy
EndPos	300.0000	EndPosition	NextIndex	5	NextIndexNumber
Space	50.00000	Step	CommandAborted	OFF	CmdAborted
HwOutputeEnable	1	OutputEnable	Error	OFF	err
		InterruptMap	ErrorID	0	errId



Note

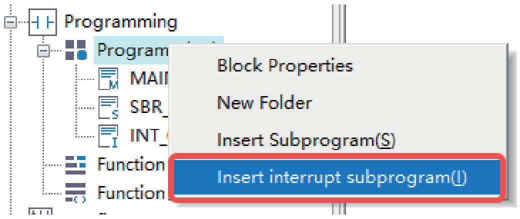
High-speed comparison hardware output is directly output through the hardware control port. Therefore, status of comparison output is not available in the Y element in the program. The Y element and the comparison output control the output port in an OR relationship. If the Y element is continuously controlled as ON, the actual port output remains ON.



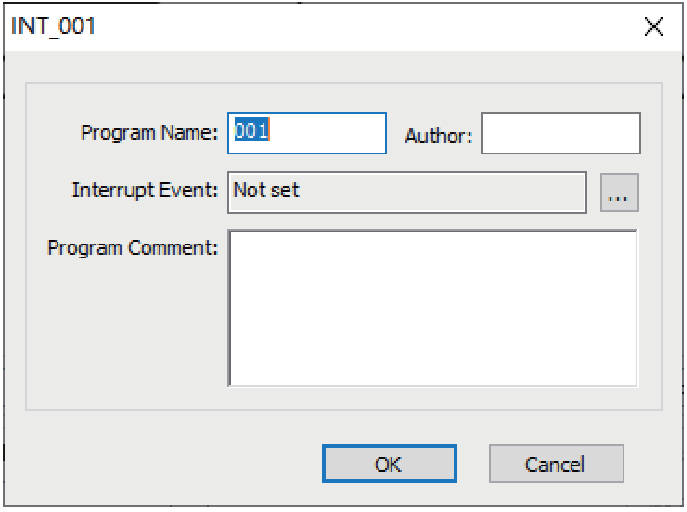
13.7.7 Comparison Interruption

When the counter axes are compared for equality, comparison interruption can be associated and the interrupt subprogram can be executed. The operation steps are as follows.

1. Under the item "Programming" in "Project Manager", right-click the "POU", and select "Insert interrupt subprogram".



2. Right-click the inserted interrupt subprogram (such as INT_001 in the figure above) and select "Properties" to open the interrupt subprogram settings page as shown in the following figure.



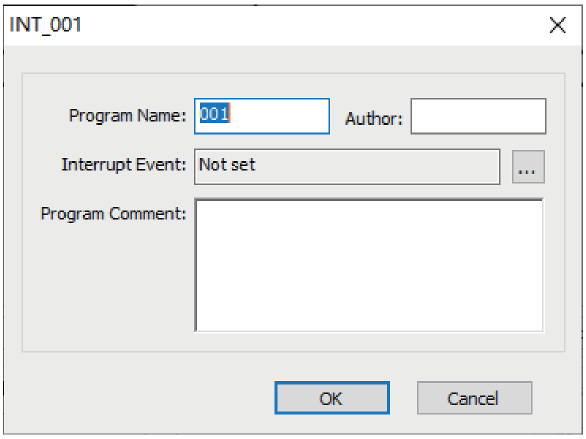
Program Name: 001

Author:

Interrupt Event: Not set

Program Comment:

3. Click the ... icon after the "Interrupt Event" field, select "Comparison interrupt", and then write interrupt subprograms in INT_001.



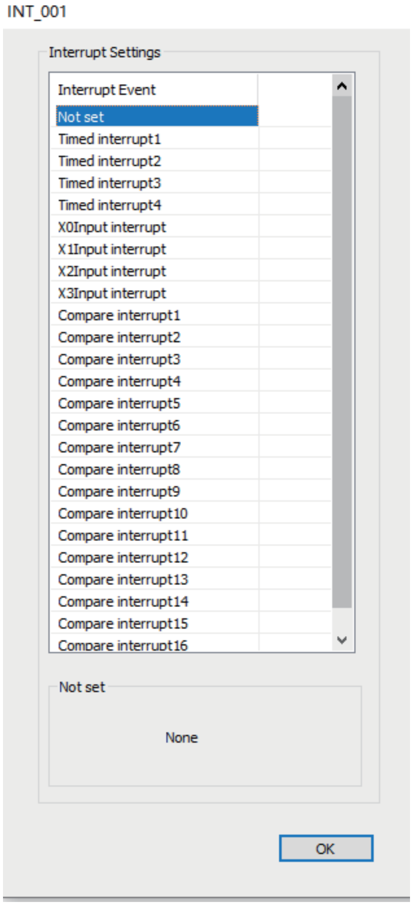
INT_001

Program Name: 001

Author:

Interrupt Event: Not set

Program Comment:



INT_001

Interrupt Settings

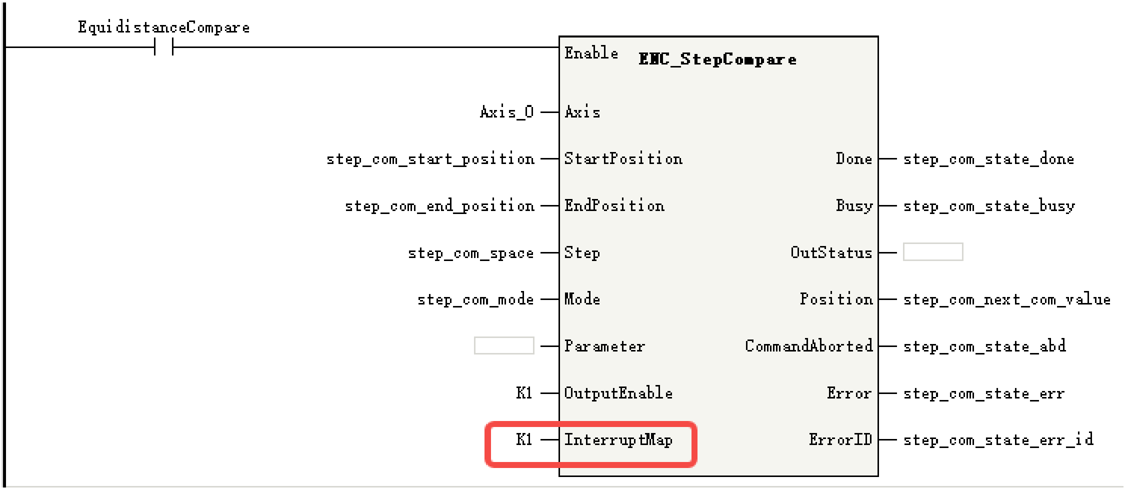
Interrupt Event

- Not set
- Timed interrupt1
- Timed interrupt2
- Timed interrupt3
- Timed interrupt4
- X0Input interrupt
- X1Input interrupt
- X2Input interrupt
- X3Input interrupt
- Compare interrupt1
- Compare interrupt2
- Compare interrupt3
- Compare interrupt4
- Compare interrupt5
- Compare interrupt6
- Compare interrupt7
- Compare interrupt8
- Compare interrupt9
- Compare interrupt10
- Compare interrupt11
- Compare interrupt12
- Compare interrupt13
- Compare interrupt14
- Compare interrupt15
- Compare interrupt16

Not set

None

4. Call ENC_Compare, ENC_StepCompare, and ENC_ArrayCompare instructions in the main program or subprogram to associate the parameter InterruptMap with the comparison interruption number, that is, setting the parameter InterruptMap as the comparison interruption number. EI is enabled in the program, and when the instruction executes the comparison for equality, the comparison of corresponding interrupt subprograms will be triggered.



EquidistanceCompare

ENC_StepCompare

Enable

Axis_0 — Axis

step_com_start_position — StartPosition

step_com_end_position — EndPosition

step_com_space — Step

step_com_mode — Mode

Parameter

K1 — OutputEnable

K1 — InterruptMap

Done — step_com_state_done

Busy — step_com_state_busy

OutStatus

Position — step_com_next_com_value

CommandAborted — step_com_state_abd

Error — step_com_state_err

ErrorID — step_com_state_err_id

13.7.8 Setting the Gear Ratio of the Axis

The PLC reconfigures the gear ratio of the local encoder axis before the local encoder axis enables count after power-on, program download, or a RUN/STOP operation.

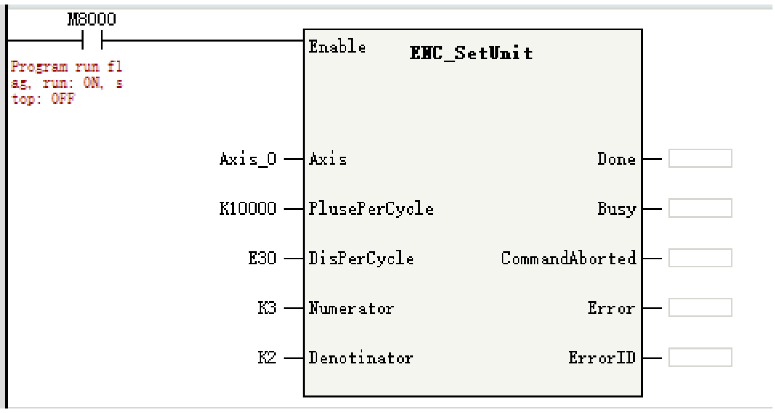
The equation for calculating the gear ratio of the local encoder axis is:

$$\text{Gear ratio} = \frac{PlusePreCycle \times Numerator}{DisPerCycle \times Denotinator}$$

For example, to realize that parameters of the local encoder axis are automatically modified to the following after the PLC is powered on:

Parameter	Value
Number of pulses in one turn by encoder	10000
Amount of movement of the worktable in a circle	30
Gear ratio numerator	3
Gear ratio denominator	2

You are recommended to add the following program to the PLC to trigger this instruction with M8000.



Note

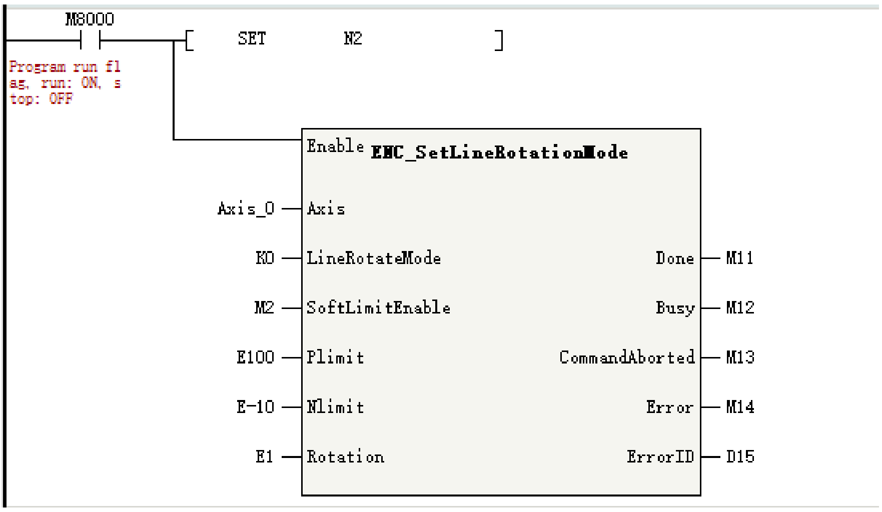
When the program is running, calling this instruction re-initializes the local encoder axis and causes an abrupt change to the feedback position of the local encoder axis.

13.7.9 Setting the Linear/Rotary Mode of the Axis

The PLC reconfigures the linear/rotary mode of the local encoder axis before the local encoder axis enables count after power-on, program download, or a RUN/STOP operation.

- When LineRotateMode is set to 0, the local encoder axis is in linear mode.
  In linear mode, when SoftLimitEnable is set to OFF, the limit is disabled. When SoftLimitEnable is set to ON means the limit is enabled, in which case PLimit represents the positive limit, and NLimit represents the negative limit.
- When LineRotateMode is set to 1, the local encoder axis is in rotary mode. In this case, Rotation represents the rotation cycle.

For example, to realize that the local encoder axis automatically switches to the linear mode after the PLC is powered on, and the limit is enabled with a positive limit of +100 and a negative limit of –10, the program is as follows.



14 Interpolation Function

14.1 Introduction to the Interpolation Function

14.1.1 Overview

The space rectangular coordinate system is adopted for interpolation, which supports linear interpolation and circular interpolation, and is performed in the form of axis groups.

- Each axis group can control up to four motion control axes (bus servo axes or local pulse axes), including three coordinate axes X, Y, and Z, and one auxiliary axis.
- The H5U supports up to eight axis groups, each of which can contain two axes (X and Y), three axes (X, Y, and Z), or four axes (X, Y, Z, and auxiliary).
- The Easy series models support different number of axes. See the specific model for the axis group supported.
- Linear and circular interpolation support BufferMode. Each axis group allows up to eight curves to be buffered, and the transition mode between curves can be set separately (see *"14.3.2 Interrupt +No Transition" on page 478* for more information on buffering and transition).

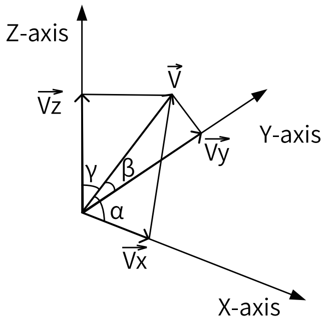


Figure 14-1 Space rectangular coordinate system

In the preceding figure, Vx, Vy, and Vz represent the velocities of the three coordinate axes, respectively, which are also the actual running velocities of the servo axes. V represents the real-time velocity of the interpolation curve. α, β, and γ represent the angles between the velocity of the interpolation curve and the coordinate axis, respectively.

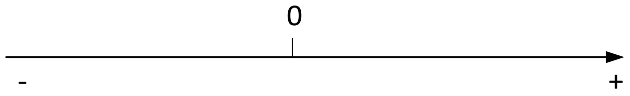


Figure 14-2 Rectangular coordinate system of the auxiliary axis

During linear interpolation, the motion control axes representing the three coordinate axes of X, Y, and Z move along the coordinate axes, while the auxiliary axis moves along a straight line from the start point to the end point.

During circular interpolation, you can choose one of the X-Y, Y-Z, and X-Z axis planes for circular interpolation, in which case if the axis group contains other axes, they will move along a straight line from the start point to the end point.

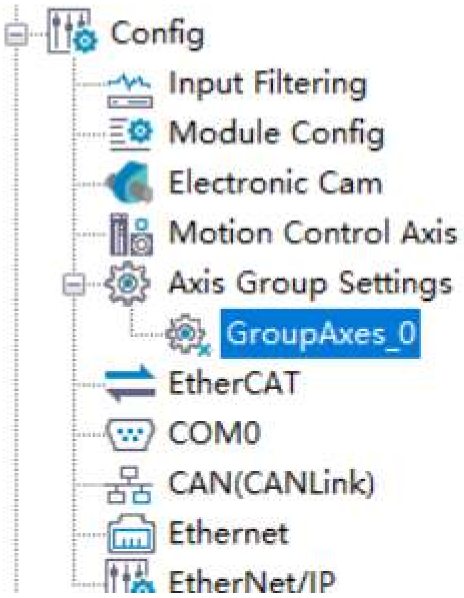
14.1.2 List of Axis Group Control Instructions

The following table lists axis group control instructions. See *H5U and Easy Series Programmable Logic Controllers Instructions Guide* for detailed usage of related instructions.

Instruction	Name
MC_MoveLinear	Linear interpolation
MC_MoveCircular	Circular interpolation
MC_MoveEllipse	Ellipse interpolation
MC_GroupStop	Stop the axis group operation.
MC_GroupPause	Pause the axis group operation.

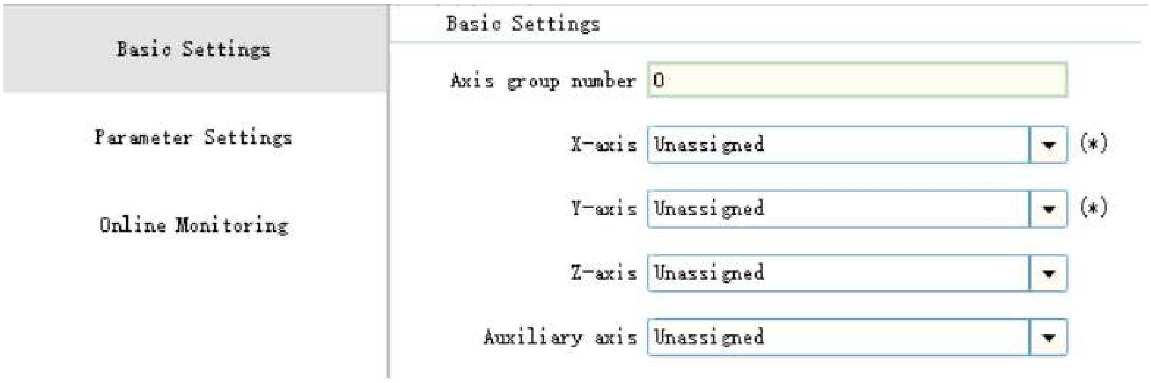
14.1.3 Configuration Interface

The menu "Axis Group Settings" is located under the node "Config". After creating an axis group, double-click the axis group to open the configuration interface of the axis group.



The interface of "Axis Group Settings" comprises three parts: "Basic setting", "Settings", and "Online monitoring".

Basic setting



Basic Settings

Axis group number 0

X-axis Unassigned (*)

Y-axis Unassigned (*)

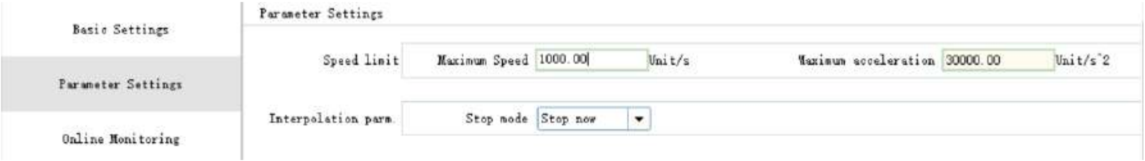
Z-axis Unassigned

Auxiliary axis Unassigned

- "Axis group number": Specifies the number of an axis group.

- Selection of coordinate axes: You can select coordinate axes from the corresponding drop-down lists, in which the X-axis and Y-axis are required, and the Z-axis and auxiliary axis are optional. One axis can exist in different axis groups.

Settings



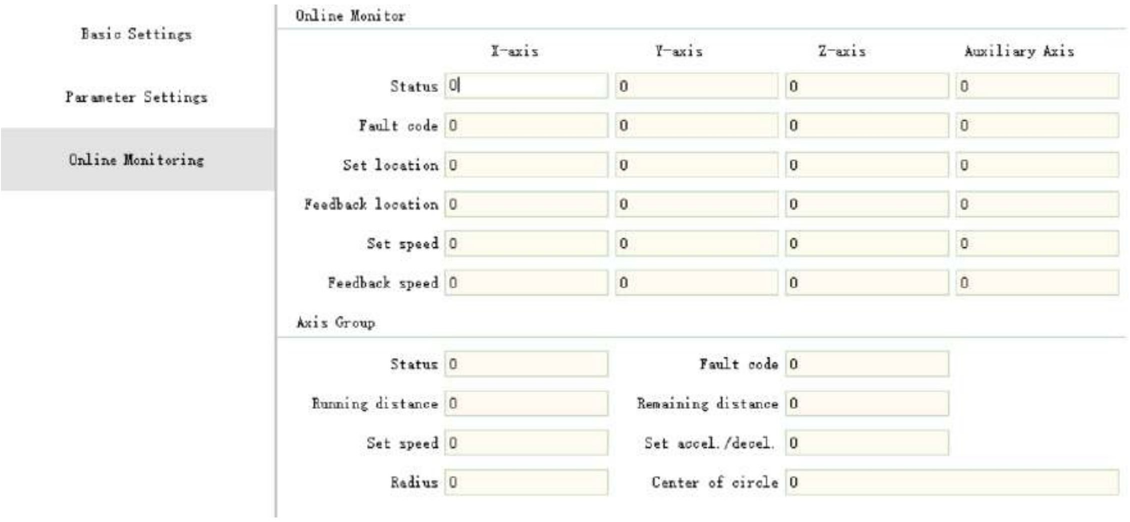
Parameter Settings

Speed limit   Maximum Speed 1000.00 Unit/s   Maximum acceleration 30000.00 Unit/s^2

Interpolation parm.   Stop mode Stop now

- "MaxVel": Specifies the maximum interpolation velocity of a space straight line in the linear interpolation mode, or the maximum linear velocity of a circular arc in the circular interpolation mode.
- "Max. acceleration": Specifies the maximum interpolation acceleration rate of a space straight line in the linear interpolation mode, or the maximum linear acceleration rate of a circular arc in the circular interpolation mode.
- "Stop mode": Specifies the stop mode when the axis group encounters an error.

Online monitoring



Online Monitor

	X-axis	Y-axis	Z-axis	Auxiliary Axis
Status	0	0	0	0
Fault code	0	0	0	0
Set location	0	0	0	0
Feedback location	0	0	0	0
Set speed	0	0	0	0
Feedback speed	0	0	0	0

Axis Group

Status	0	Fault code	0
Running distance	0	Remaining distance	0
Set speed	0	Set accel./decel.	0
Radius	0	Center of circle	0

- "MaxVel": Specifies the maximum interpolation linear velocity of a space straight line in the linear interpolation mode, or the maximum linear velocity of a circular arc in the circular interpolation mode.
- "Max. acceleration": Specifies the maximum interpolation acceleration rate of a space straight line in the linear interpolation mode, or the maximum acceleration rate of a circular arc in the circular interpolation mode.
- "Stop mode": Specifies the stop mode when the axis group encounters an error.

Table 14–1 Online monitoring parameters

Parameter	Description
Status	The status of a single-axis PLCOpen state machine 0: PowerOff 1: ErrorStop 2: Stopping 3: StandStill 4: DiscreteMotion 5: ContinuousMotion 7: Homing 8: SynchronizedMotion
Fault code	The fault code when a single axis is in the ErrorStop state
Setting position	Real-time target position for a single axis
Feedback position	Real-time feedback position for a single axis
Setting speed	Real-time velocity reference for a single axis
Feedback speed	Real-time feedback velocity for a single axis

Table 14–2 Axis group monitoring parameters

Name	Description
Status	The status of an axis group 0: Init The axis configuration in the axis group is not completed. 1: Disabled Not all axes in the axis group are enabled. 2: Single Stop An axis in the axis group calls the instruction MC_Gtop. 3: Single Homing An axis in the axis group calls the instruction MC_Home. 4: Single motion An axis in the axis group calls single-axis motion instructions such as MC_MoveAbsolute. 5: ErrorStop An axis in the axis group is in a fault state. 6: StandStill All axes in the axis group are in the StandStill state. 7: Stopping The instruction MC_GroupStop is called. 8: Synchronous Motion A linear interpolation or circular interpolation instruction is called.
Fault code	The fault code when the axis group fails due to error of a single axis
Operation distance	In linear interpolation mode, it indicates the distance at which a space straight line moves after the instruction is executed. In circular interpolation mode, it indicates the length of a circular arc in which the circular arc moves after the instruction is executed.

Name	Description
Remaining distance	In linear interpolation mode, it indicates the left distance for this section of a space straight line after the instruction is executed. In circular interpolation mode, it indicates the length of a space circular arc left after the instruction is executed.
Setting speed	In linear interpolation mode, it indicates the interpolation velocity of a space straight line. In circular interpolation mode, it indicates the linear velocity of a circular arc.
Setting acceleration/deceleration	The change rate of the velocity reference
Radius	The radius of a circular arc during circular interpolation
Center	The center of a circular arc during circular interpolation

14.2 Interpolation Operations

14.2.1 Overview

To properly execute an interpolation instruction, you need to first create an axis group and enable axes in the axis group. The following figure shows the basic process.

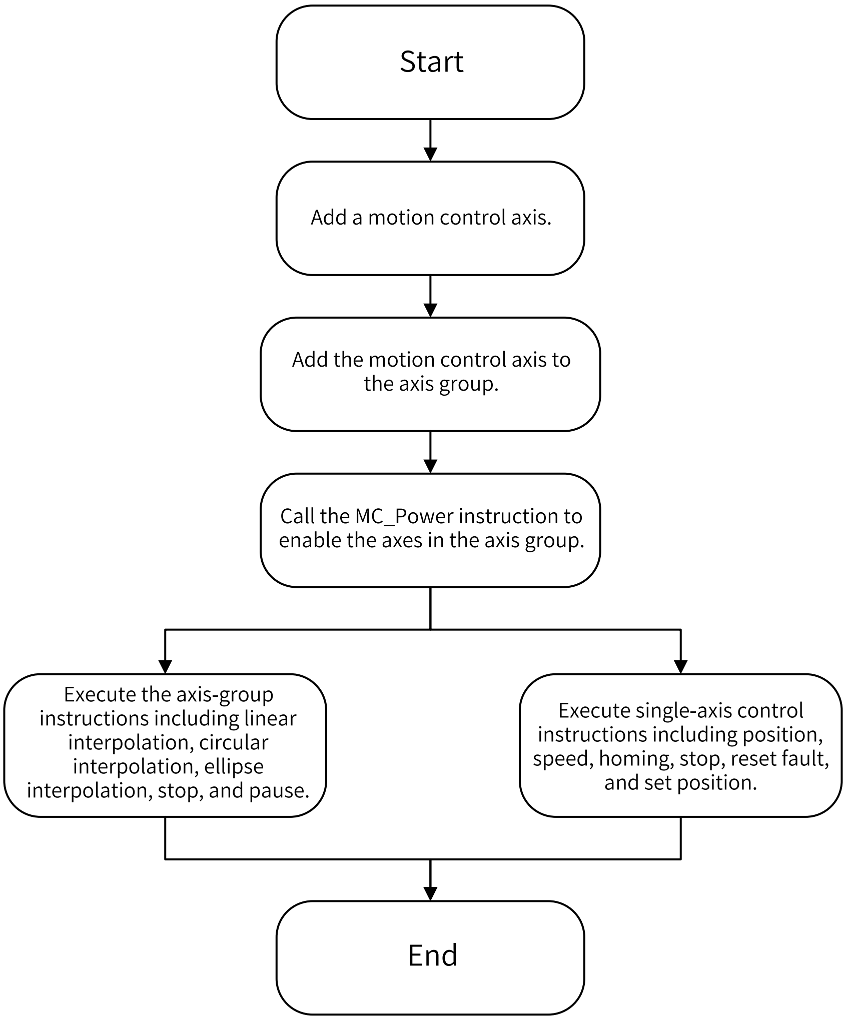

Start
Add a motion control axis.
Add the motion control axis to the axis group.
Call the MC_Power instruction to enable the axes in the axis group.
Execute the axis-group instructions including linear interpolation, circular interpolation, ellipse interpolation, stop, and pause.
Execute single-axis control instructions including position, speed, homing, stop, reset fault, and set position.
End

Figure 14-3 Flow chart of interpolation operation

Note

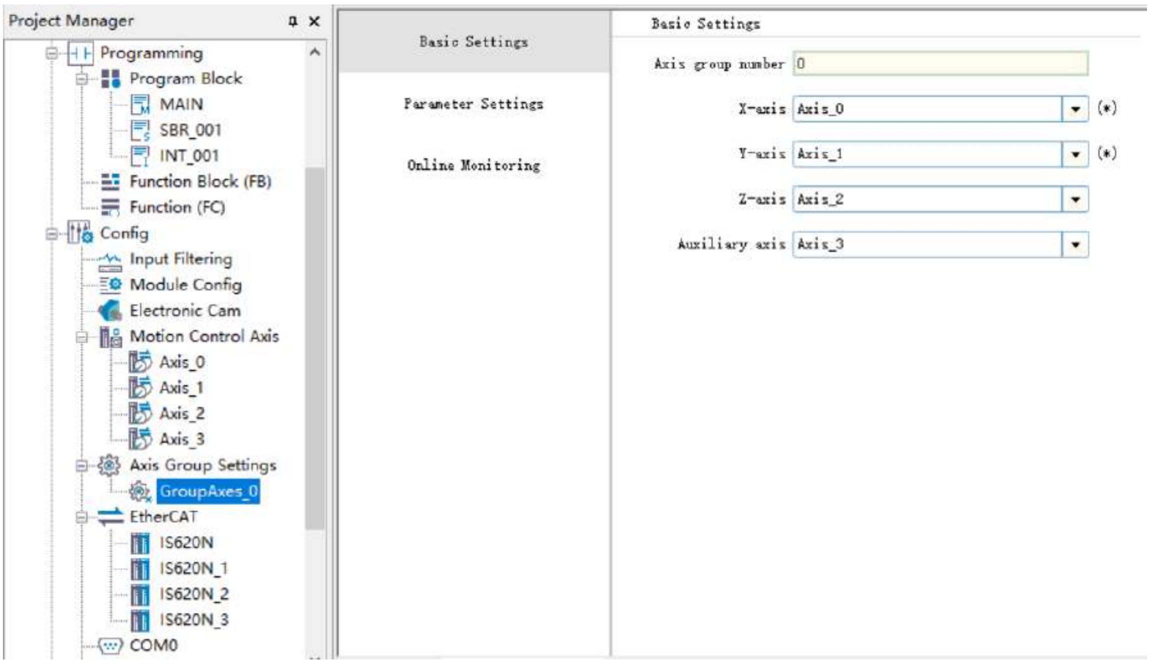
Even after an axis group is created, axes in the axis group can still execute single-axis motion and control instructions. However, motion instructions for a single axis and interpolation instructions for an axis group are mutually exclusive. These instructions cannot be activated at the same time or interrupt each other.

This section describes the basic interpolation procedures based on a routine that combines Axis_0, Axis_1, Axis_2, and Axis_3 into an axis group to perform related actions.

Detailed information on the configuration of motion control axes can be found in the section "Motion Control".

14.2.2 Creating an Axis Group

Right-click "Axis Group Settings", and select "Add Axis Group". After creating an axis group, you can choose the coordinate axes and auxiliary axis and set relevant parameters.



Basic Settings

Axis group number 0

X-axis Axis_0 (*)

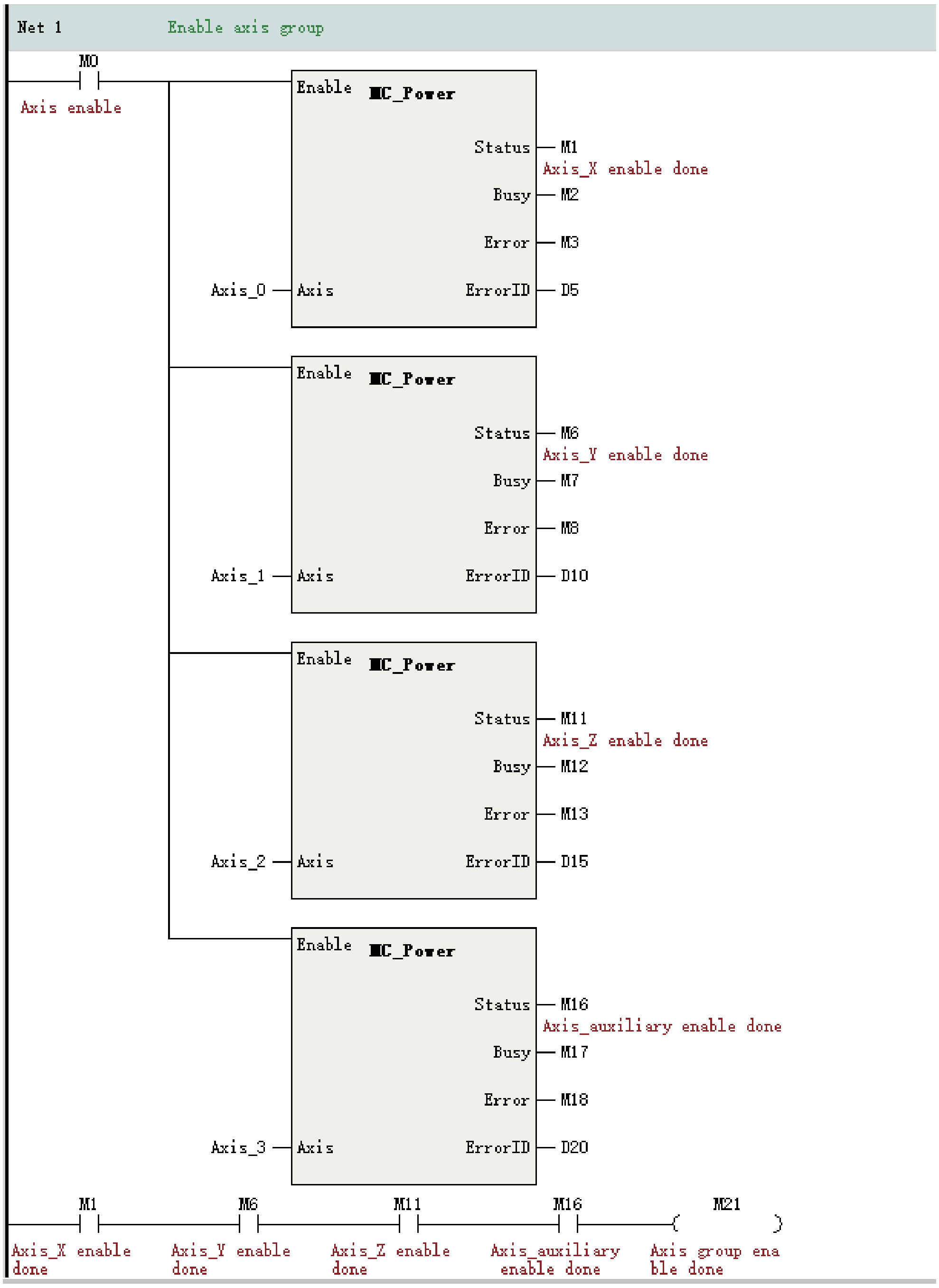
Y-axis Axis_1 (*)

Z-axis Axis_2

Auxiliary axis Axis_3

14.2.3 Enabling an Axis Group

Each single axis in the axis group is enabled and disabled through the instruction MC_Power. The axis group control instruction can only be executed if all axes in the axis group are enabled.



14.2.4 Linear Interpolation

The linear interpolation of an axis group is implemented by the instruction MC_MoveLinear.
When all axes in the axis group are in the StandStill state, the Execute is triggered, the axis group starts to implement linear interpolation, and all axes in the axis group switch to the Synchronized Motion state. In this case, the single-axis motion instructions such as MC_MoveAbsolute and MC_Stop must not be executed.

After the linear interpolation is completed, all axes in the axis group return to the StandStill state, in which case single-axis motion instructions such as MC_MoveAbsolute and MC_Stop can be executed again.

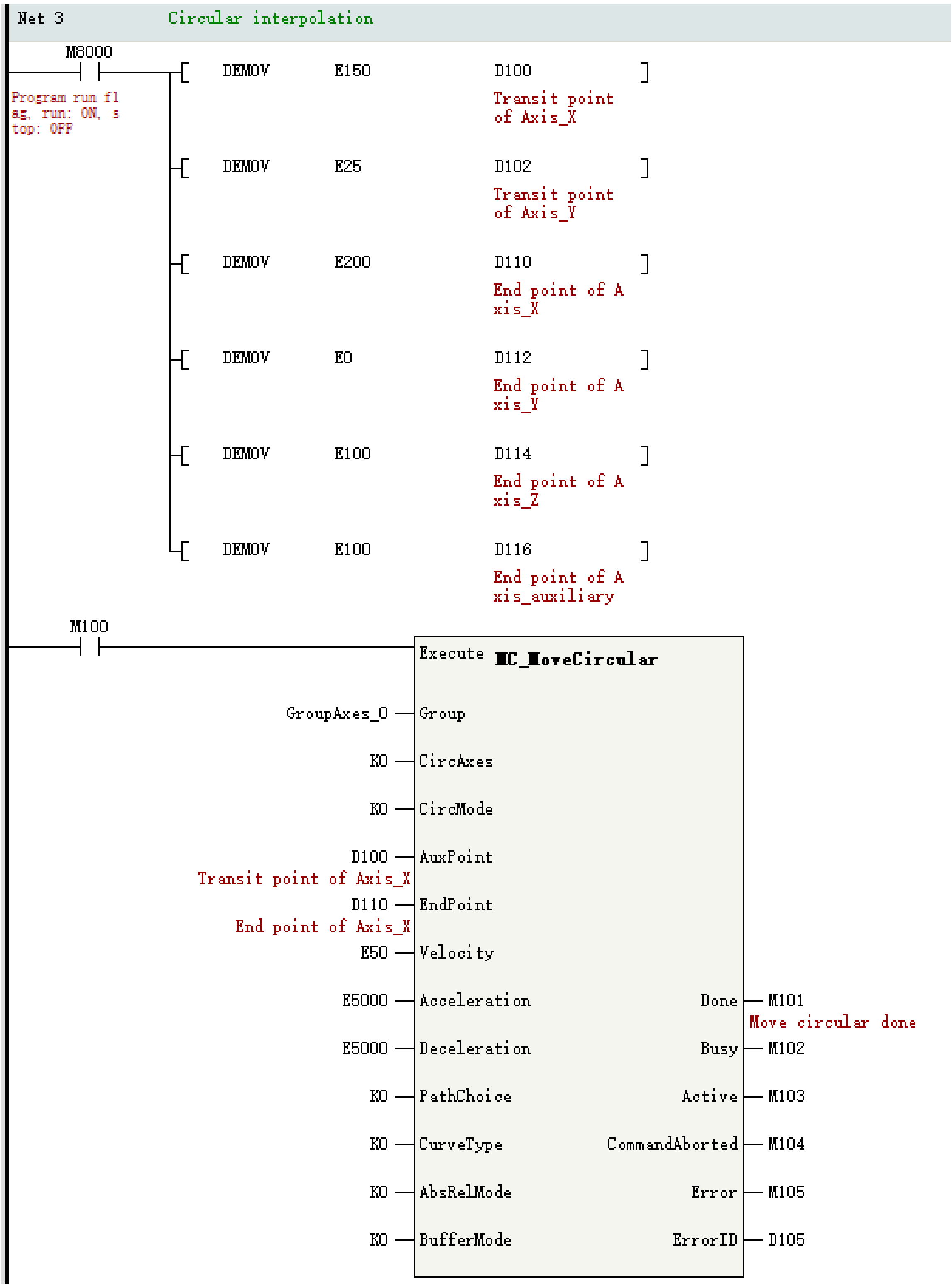
Example

This routine uses absolute positioning to position the X-axis, Y-axis, and Z-axis to the position (100,100,100), and the auxiliary axis to the position 50.

14.2.5 Circular Interpolation

Circular interpolation of an axis group is implemented by the instruction MC_MoveCircular. The conversion rules for PLCOpen state machines are the same as those for linear interpolation.
This routine implements circular interpolation of the X-Y axis plane while the Z-axis and auxiliary axis making synchronous linear motion. The circular interpolation is implemented in the border point mode, that is, with absolute positioning, first passing through the border point (150,25) and then reaching the position (200,0). The Z-axis and auxiliary axis reach the position 100.

For specific parameters related to the circular arc instructions, see *H5U and Easy Series Programmable Logic Controllers Instructions Guide*.



```
Net 3            Circular interpolation
   M8000
───┤ ├────┬─[  DEMOV     E150          D100           ]
Program run fl│                           Transit point
ag, run: ON, s│                           of Axis_X
top: OFF      ├─[  DEMOV     E25           D102           ]
              │                           Transit point
              │                           of Axis_Y
              ├─[  DEMOV     E200          D110           ]
              │                           End point of A
              │                           xis_X
              ├─[  DEMOV     E0            D112           ]
              │                           End point of A
              │                           xis_Y
              ├─[  DEMOV     E100          D114           ]
              │                           End point of A
              │                           xis_Z
              └─[  DEMOV     E100          D116           ]
                                          End point of A
                                          xis_auxiliary
   M100
───┤ ├──────────────────────── Execute  MC_MoveCircular
             GroupAxes_0 ── Group
                      K0 ── CircAxes
                      K0 ── CircMode
                    D100 ── AuxPoint
    Transit point of Axis_X
                    D110 ── EndPoint
        End point of Axis_X
                     E50 ── Velocity
                   E5000 ── Acceleration                Done ── M101
                                                                Move circular done
                   E5000 ── Deceleration                Busy ── M102
                      K0 ── PathChoice                Active ── M103
                      K0 ── CurveType       CommandAborted ── M104
                      K0 ── AbsRelMode                 Error ── M105
                      K0 ── BufferMode               ErrorID ── D105
```

14.2.6 Axis Group Stop

Stop the execution of the interpolation curve by the instruction MC_GroupStop.
The execution of the interpolation instruction is interrupted on the rising edge of Execute, and the CommandAborted output of the interpolation instruction is valid.

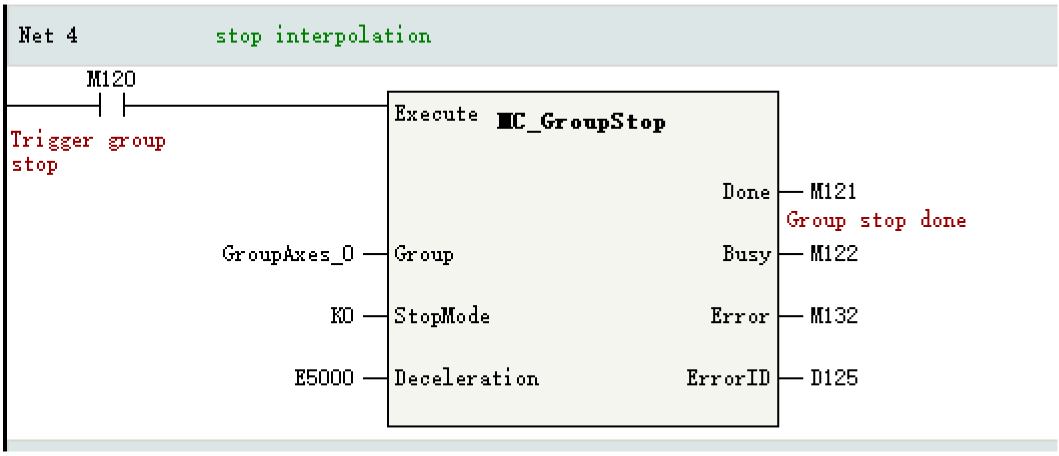
Interpolation instructions cannot be triggered when Execute is TRUE. Execute must be set to False to re-execute a new interpolation instruction.

This instruction can only be called when all axes in the axis group are in the StandStill or Synchronized Motion state. Axes are in the Synchronized Motion state while the Execute of the instruction is true.

MC_GroupStop can only stop the operation of an interpolation curve, rather than that of single-axis motion instructions such as MC_MoveAbsolute.

Example

In this routine, the following instruction is called during linear interpolation or circular interpolation to stop the operation of an interpolation curve. With the decelerate-to-stop mode in use, the deceleration to stop is 5000.



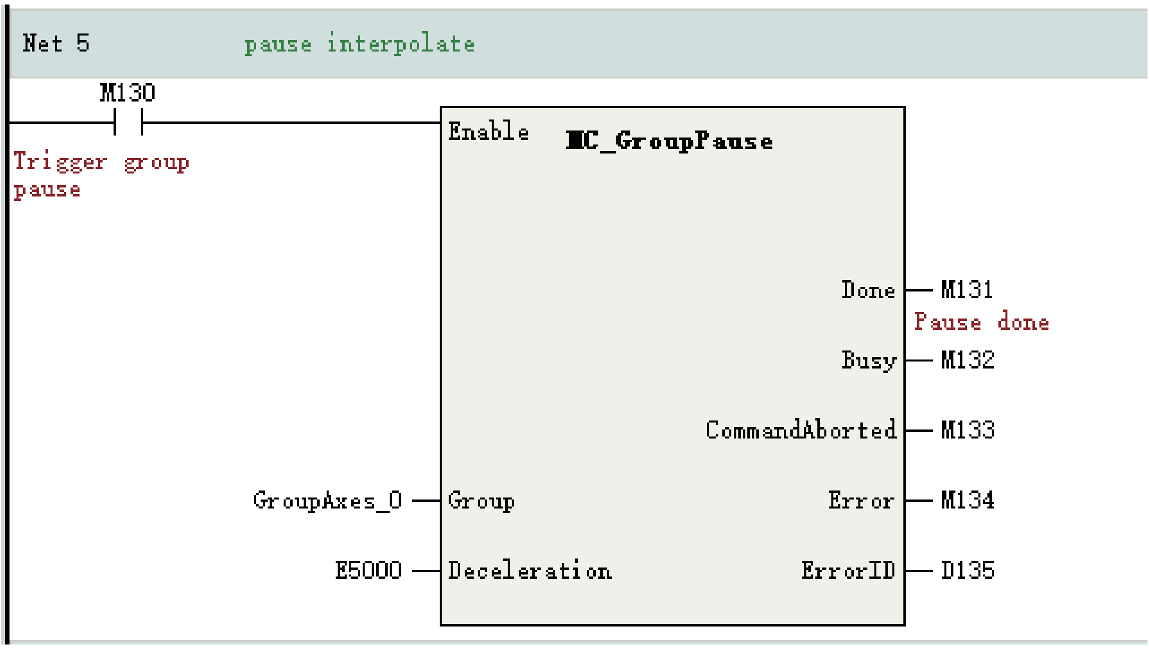
14.2.7 Axis Group Pause

The interpolation curve is paused by the instruction MC_GroupPause.
Pause the interpolation curve when Enable is TRUE and resume the execution when Enable is False.

MC_GroupPause can only pause the operation of an interpolation curve, rather than that of single-axis motion instructions such as MC_MoveAbsolute.

Example

In this routine, the following instruction is called during line or circular interpolation to pause the operation of the interpolation curve, and the deceleration to pause is 5000.

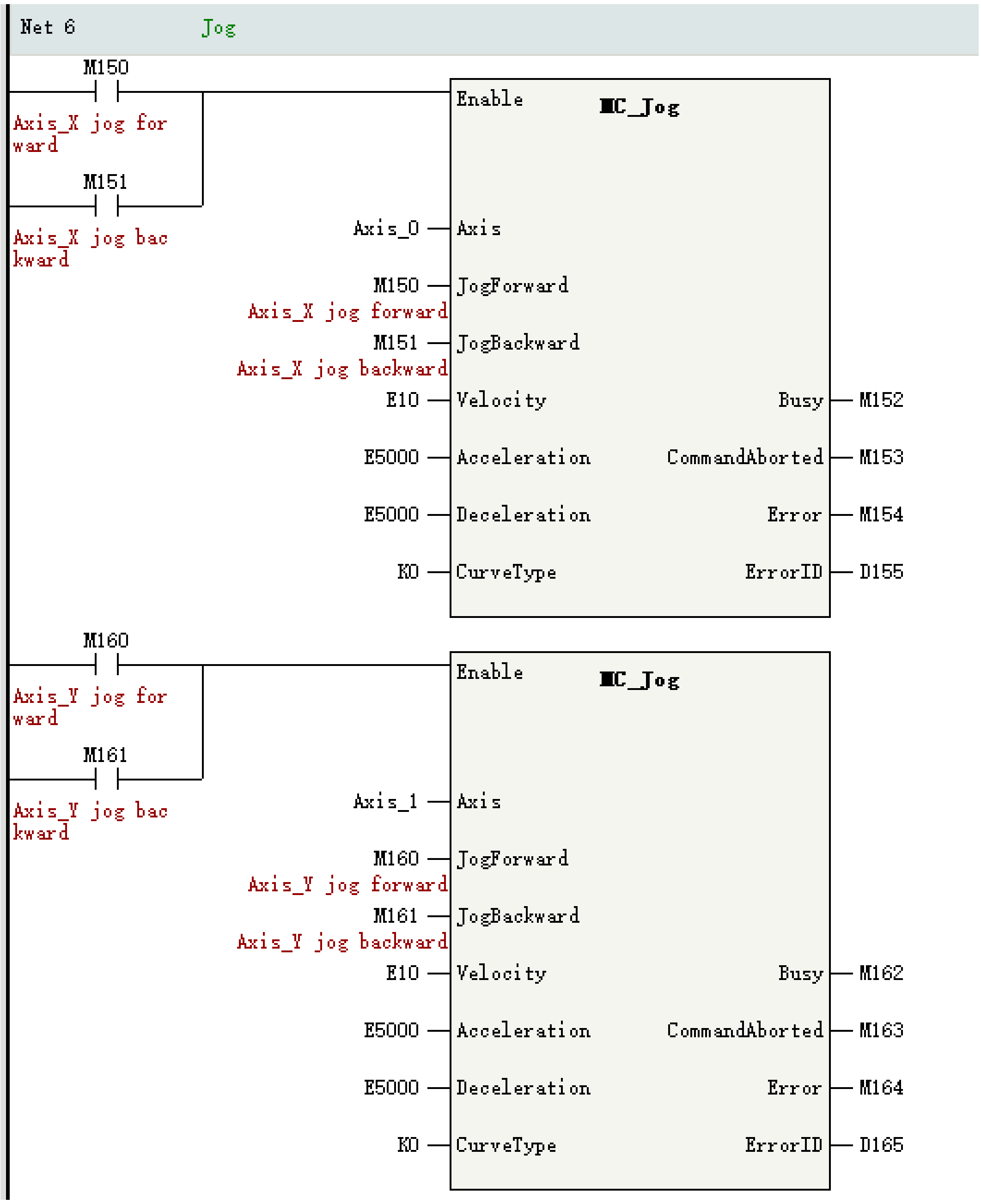


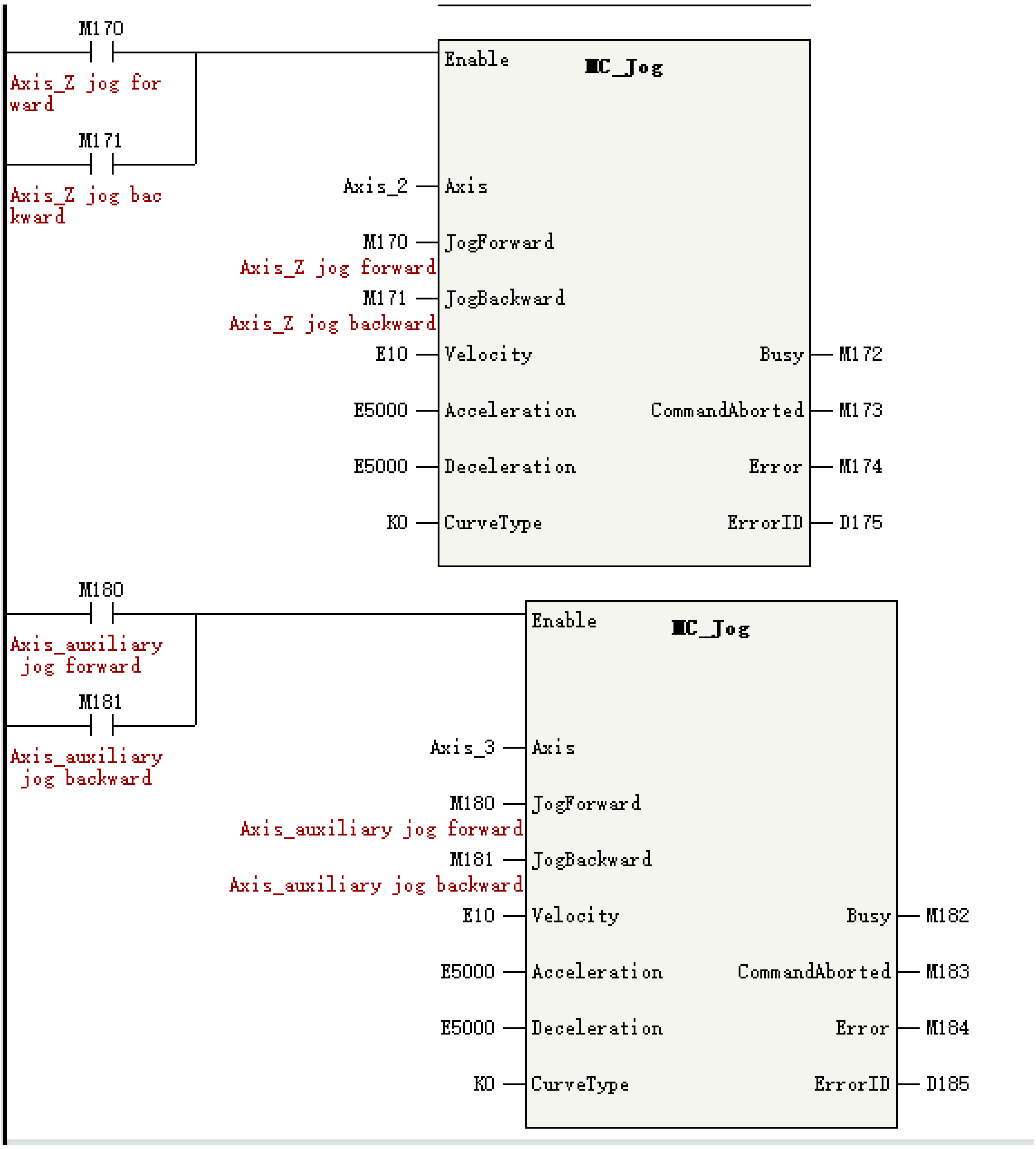
14.2.8 Single-axis Motion

When the single-axis PLCOpen state machine is in the Synchronized Motion state due to the execution of the interpolation action, the single-axis instructions such as MC_MoveAbsolute and MC_Stop must not be executed. When the axis is in the StandStill state, the single-axis motion instructions, such as MC_MoveAbsolute, MC_MoveRelative, and MC_Jog, can be called to control the single-axis operation.

Example

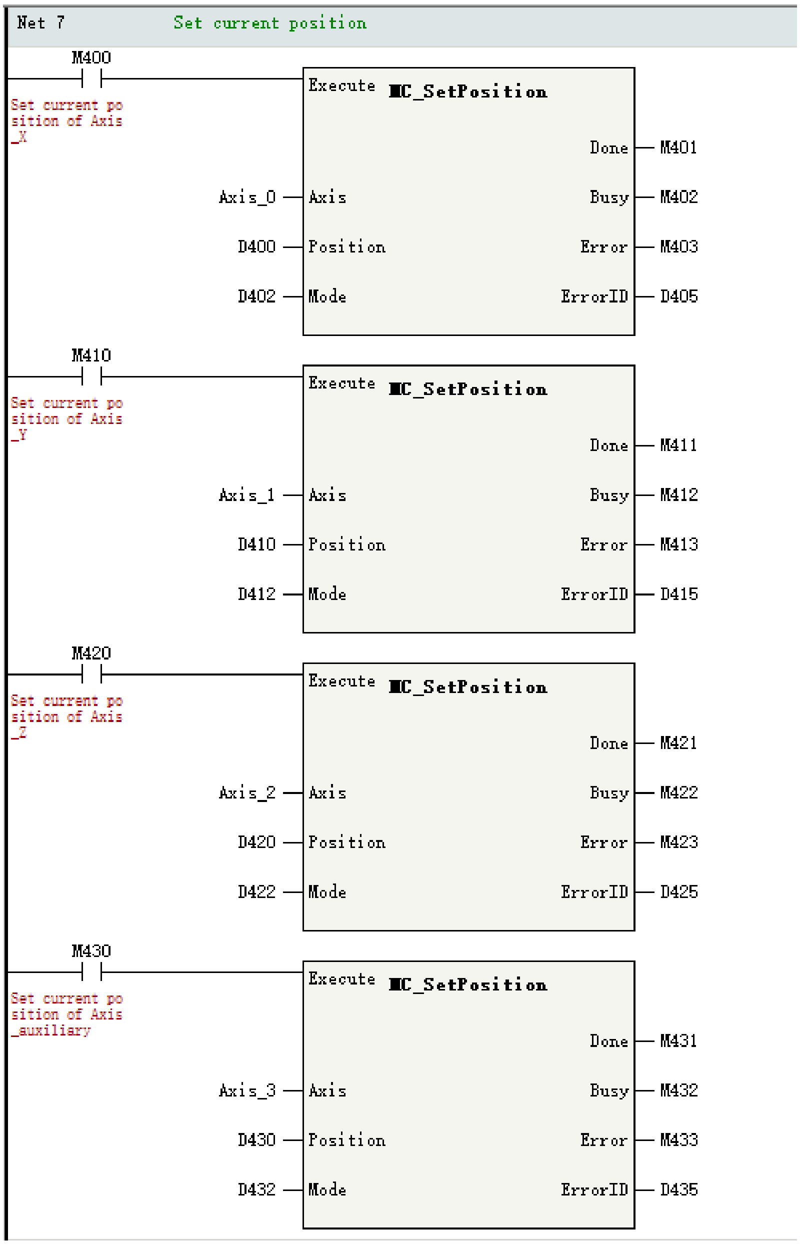
In this routine, when no interpolation instruction is executed for the axis group, the jogging of the single axis is realized through the instruction MC_Jog.


Net 6
Jog
M150
Axis_X jog forward
M151
Axis_X jog backward
MC_Jog
Enable
Axis_0 — Axis
M150 — JogForward
Axis_X jog forward
M151 — JogBackward
Axis_X jog backward
E10 — Velocity
Busy — M152
E5000 — Acceleration
CommandAborted — M153
E5000 — Deceleration
Error — M154
K0 — CurveType
ErrorID — D155
M160
Axis_Y jog forward
M161
Axis_Y jog backward
MC_Jog
Enable
Axis_1 — Axis
M160 — JogForward
Axis_Y jog forward
M161 — JogBackward
Axis_Y jog backward
E10 — Velocity
Busy — M162
E5000 — Acceleration
CommandAborted — M163
E5000 — Deceleration
Error — M164
K0 — CurveType
ErrorID — D165



14.2.9 Setting the Current Position

Set the current position through the instruction MC_SetPosition.



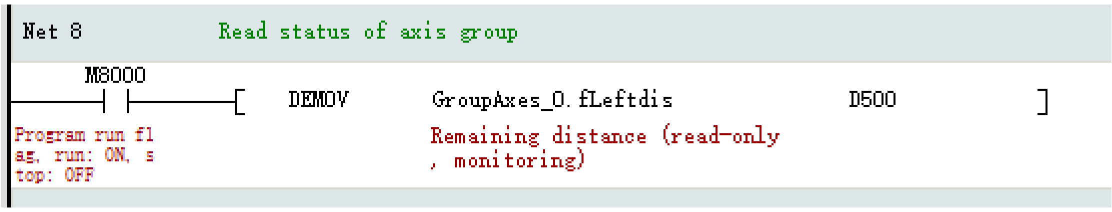
14.2.10 Reading the Current Status

Single-axis status

The status, feedback position, feedback velocity, and feedback torque of the PLCOpen state machine of a single axis are obtained through instructions MC_ReadStatus, MC_ReadActPosition, MC_ReadActVelocity, and MC_ReadActTorque. The status of a single axis can also be accessed through the system variables of the axis. See *"12.4 Online Monitoring" on page 409* for details.

The status of an axis group

Access the state of an axis group through its system variables.



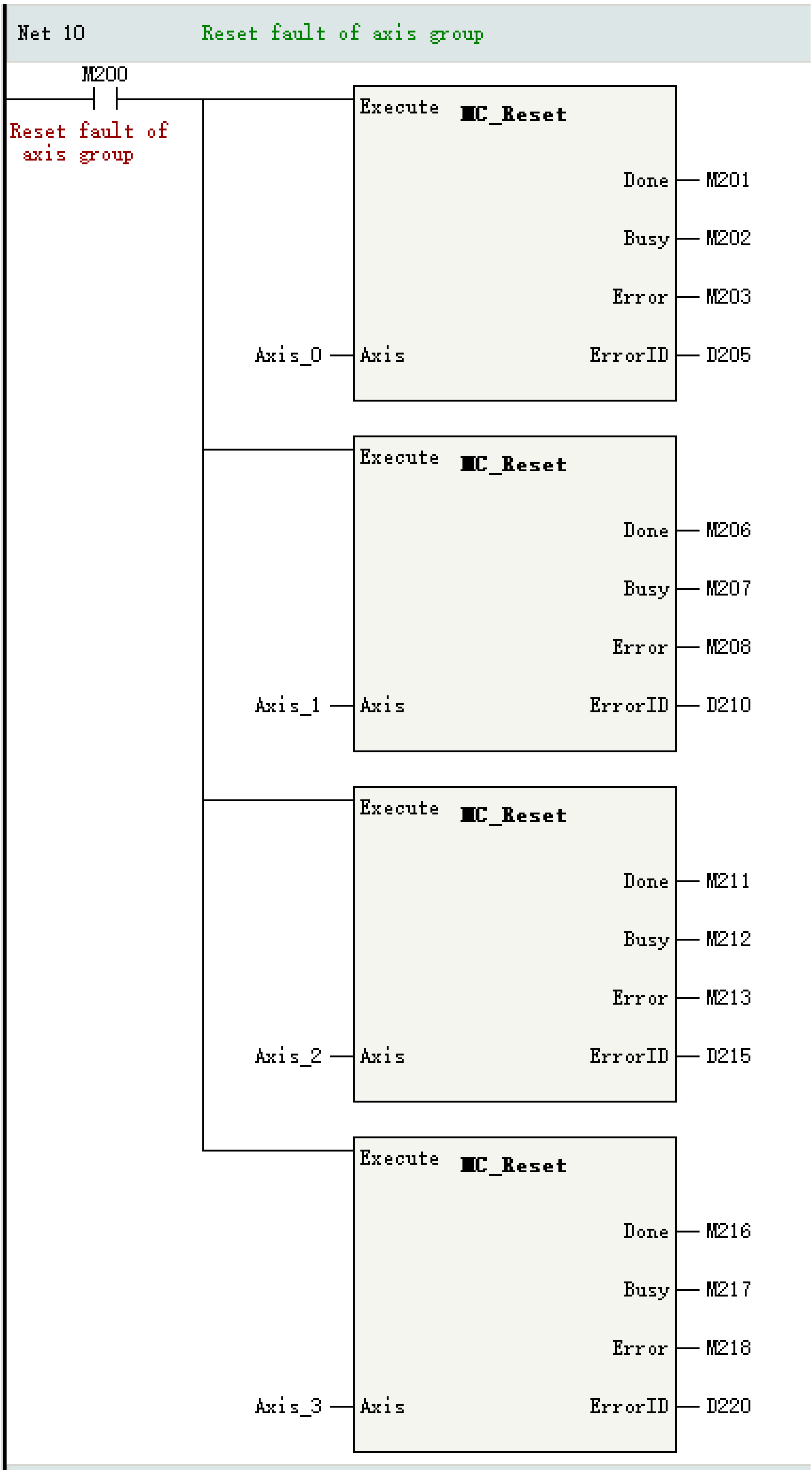
```
Net 8          Read status of axis group
   M8000
───┤ ├───[  DEMOV    GroupAxes_0.fLeftdis          D500      ]
Program run fl           Remaining distance (read-only
ag, run: ON, s           , monitoring)
top: OFF
```

14.2.11 Resetting Axis Group Faults

When a single axis enters the fault state, the fault of the axis can be reset through the instruction MC_ Reset, and only by resetting the single-axis fault can the fault of the entire axis group be removed.

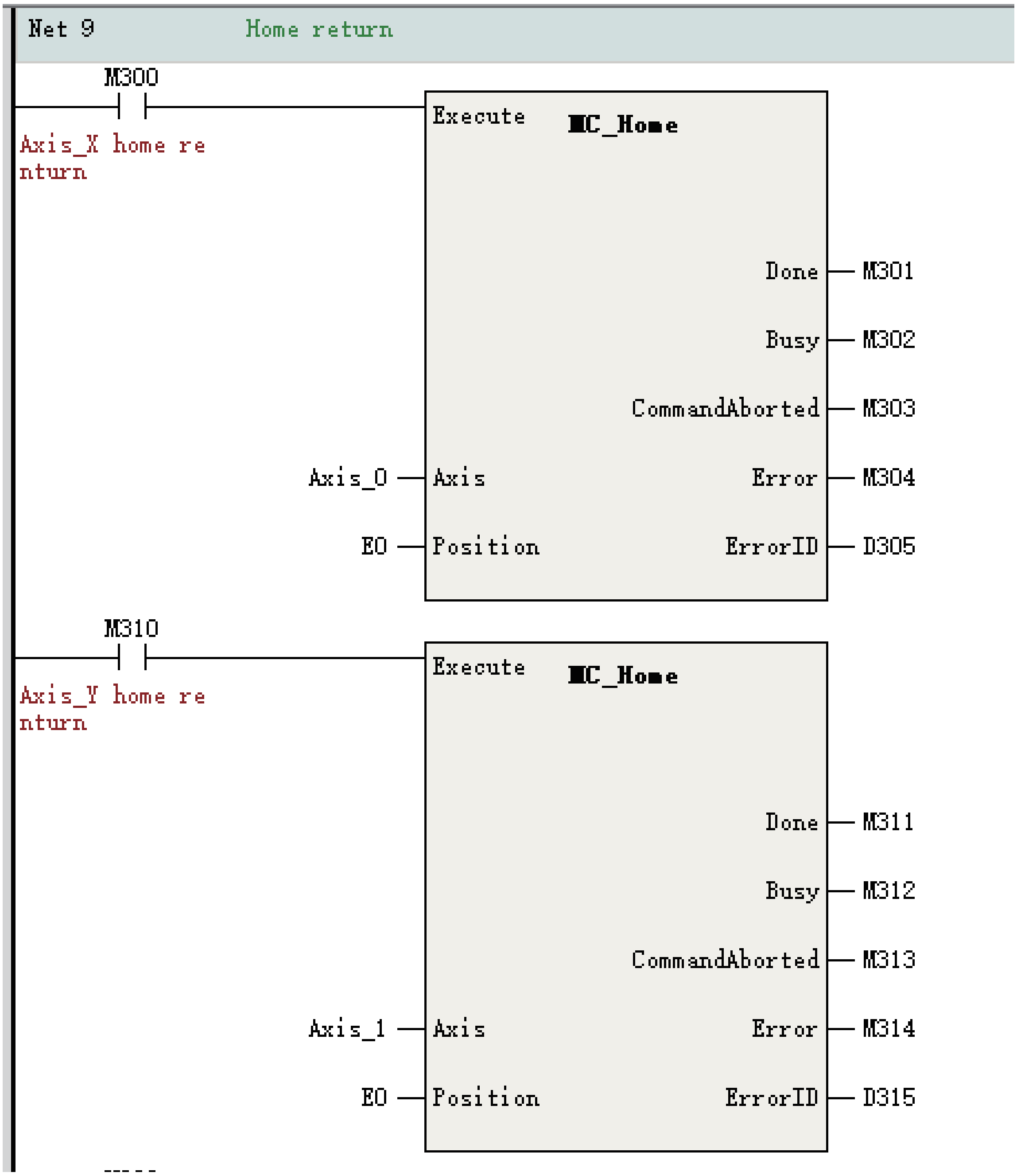
Example

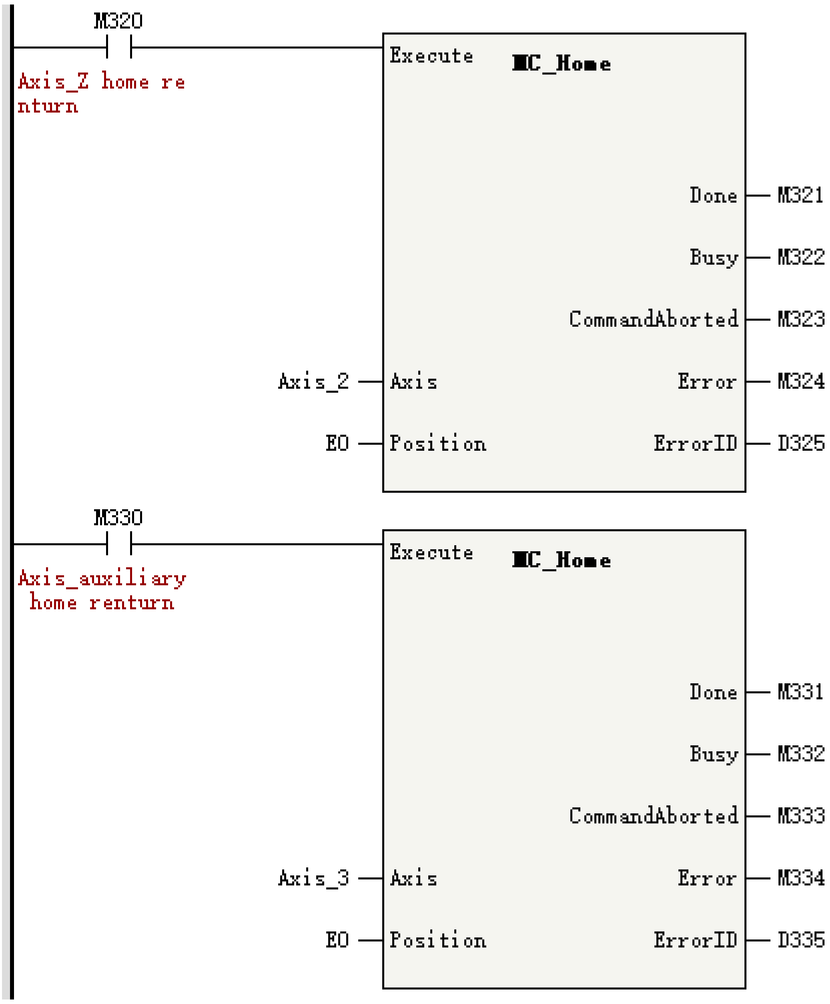
In this routine, M200 is triggered to reset the fault of the axis group.



14.2.12 Homing

The homing for an axis group can be realized through single homing. The following is the single homing implemented by the instruction MC_Home.


Net 9
Home return
M300
Axis_X home re
nturn
Execute
MC_Home
Done
M301
Busy
M302
CommandAborted
M303
Axis_0
Axis
Error
M304
E0
Position
ErrorID
D305
M310
Axis_Y home re
nturn
Execute
MC_Home
Done
M311
Busy
M312
CommandAborted
M313
Axis_1
Axis
Error
M314
E0
Position
ErrorID
D315



14.3 Buffer and Transition

14.3.1 Overview

The buffer mode refers to the process of executing instructions when multiple interpolation instructions are started at the same time.

The transition mode refers to the way when multiple curves switch between each other.

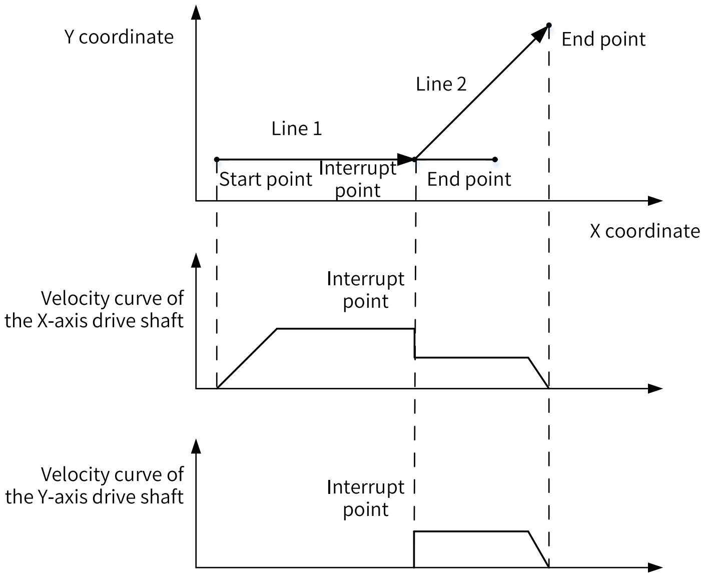
The following four combined buffer and transition modes are supported.

No.	Buffer Mode	Description
0	Interrupt+No transition	Immediately switch to the next function block. There is no transition curve.
1	Buffer+No transition	Execute the buffered function block after the first segment of deceleration is completed. There is no transition curve.
2	Previous Velocity+No transition	Move to the end of the first segment at the current velocity and start the second segment at the rate of the first segment.
3	Additional angle transition	Add acceleration of the second segment when deceleration starts in the first segment. There is a transition curve.

14.3.2 Interrupt+No Transition

The first interpolation instruction is executed first, and the second interpolation instruction is triggered before the first straight line is completed. If the BufferMode of the second interpolation instruction is set to "Interrupt+No Transition", the second interpolation instruction immediately interrupts the first interpolation instruction and starts implementing a new interpolation curve.

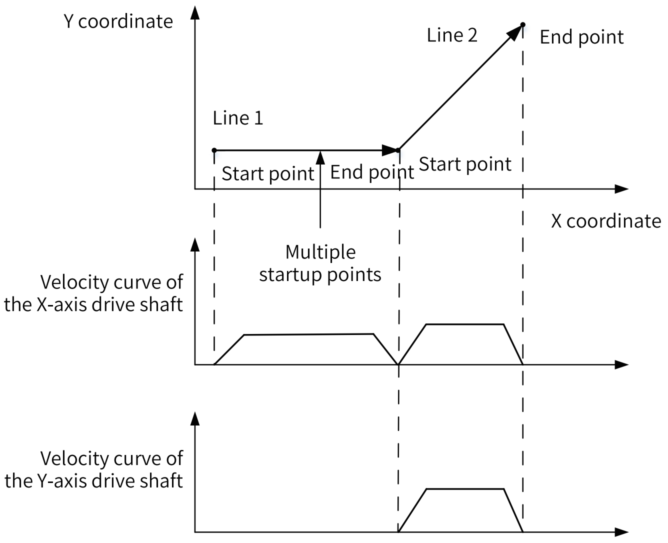
At the interrupt point, the new curve remains at the same velocity rate, and the velocities of the X-axis, Y-axis are re-decomposed, as shown in the following figure.



14.3.3 Buffer+No Transition

The first interpolation instruction is executed first, and the second interpolation instruction is triggered before the first straight line is completed. If the BufferMode of the second interpolation instruction is set to "Buffer+No Transition", the interpolator will continue to execute the first interpolation instruction.

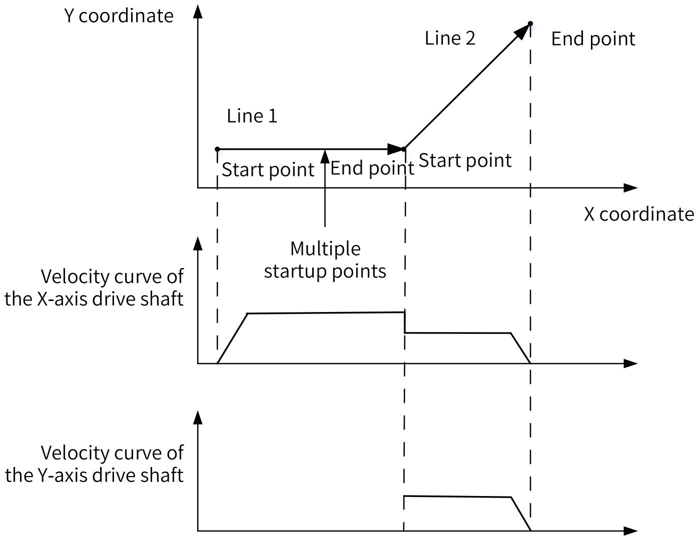
After the first interpolation instruction is executed and an active Done signal is output, the execution of the second interpolation instruction will begin, as shown in the following figure.



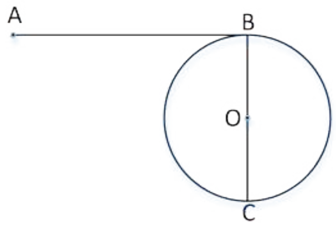
14.3.4 Previous Velocity+No Transition

The first interpolation instruction is executed first, and the second interpolation instruction is triggered before the first straight line is completed. If the BufferMode of the second interpolation instruction is set to "Previous Velocity+No Transition", the interpolator will attempt to maintain the target velocity of the first instruction to implement a full straight line.

After the first interpolation instruction is executed and an active Done signal is output, the execution of the second interpolation instruction will begin. The velocity rate will remain unchanged at the switching point, and the velocities of the coordinate axes will be redistributed. The following figure shows the details.

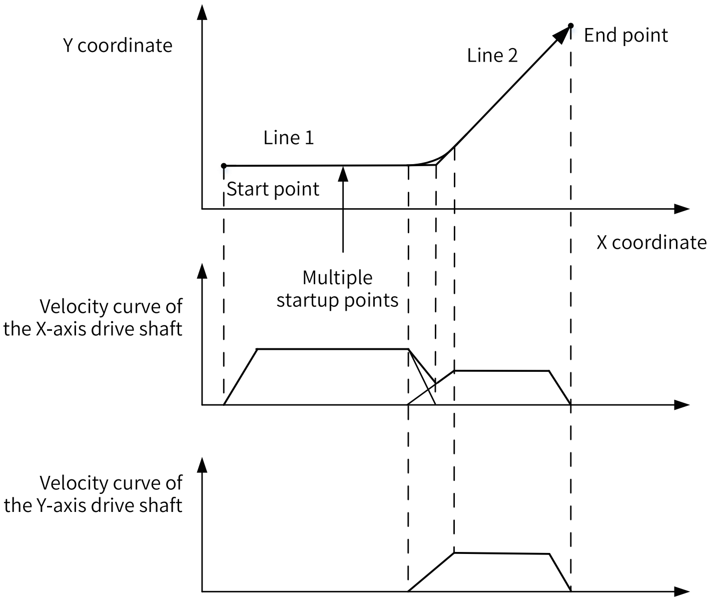


This mode is particularly suitable for switching between straight lines and circular arcs with the straight lines on the tangent line of the arcs, and can be used to maintain the constant velocity of an interpolation curve.



14.3.5 Additional Angle Transition

The first interpolation instruction is executed first, and the second interpolation instruction is triggered before the first straight line is completed. If the BufferMode of the second interpolation instruction is set to the additional angle mode, the interpolator will initiate the second interpolation instruction when it detects that the first straight line has begun to perform deceleration, and the final velocity of each coordinate axis is the sum of the velocities of the two instructions, as shown in the following figure.



14.4 Methods of Handling Single-Axis Configuration Parameters in Interpolation

Some of the configuration parameters in the interpolation are different from the single-axis configuration parameters, as shown in the following table.

Single-axis configuration parameters	Handling Methods in Interpolation
Gear ratio setting	The gear ratio of a single axis in an axis group is set on the interface "Unit conversion setting" of the single axis.
Encoder mode selection	The encoder mode of the drive can be set to incremental or absolute. Set this parameter in the interface "Mode/Parameter setting" of a single axis.

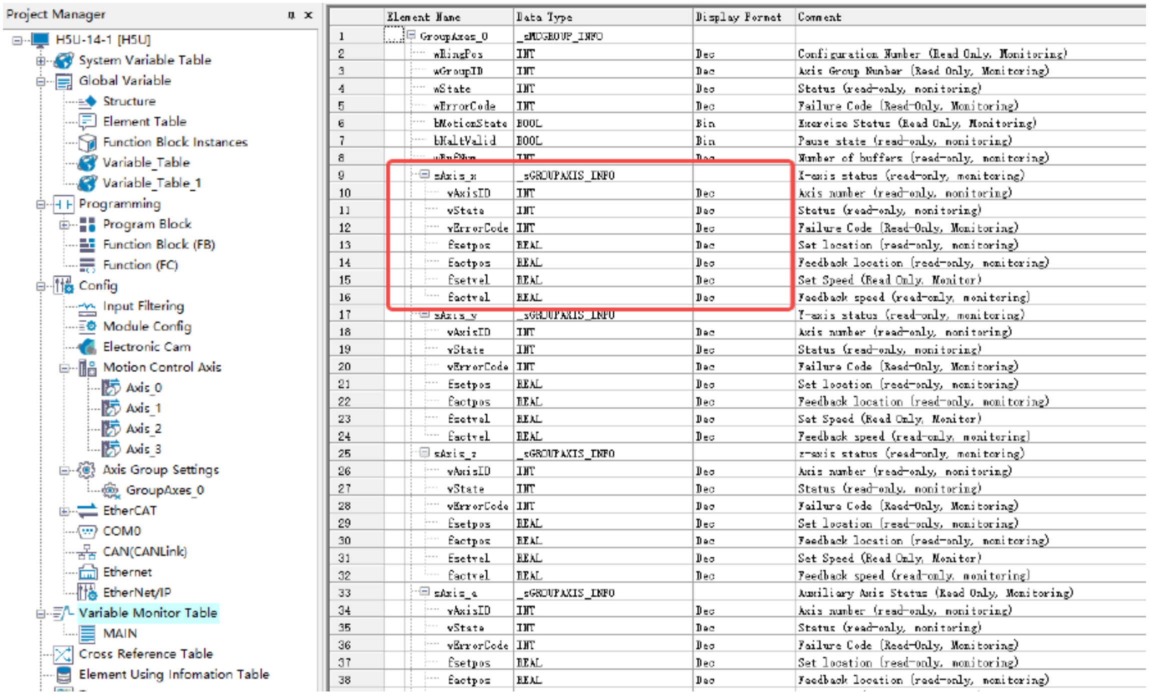
Single-axis configuration parameters	Handling Methods in Interpolation
Mode setting	Modes of axes in an axis group are divided into linear mode and ring mode according to the working conditions, and the interpolation instructions only support the linear mode.
Limit handling	Axes in the interpolation instructions support both software limits and hardware limits.
Following error	Axes in interpolation instructions support following errors.
Velocity limit	The velocity in an interpolation instruction is limited by the maximum velocity of a single axis, not by the maximum acceleration rate.
Torque limit	Not involved.

14.5 System Variables

14.5.1 _sGROUPAXIS_INFO for Status of Coordinate Axes within Axis Group

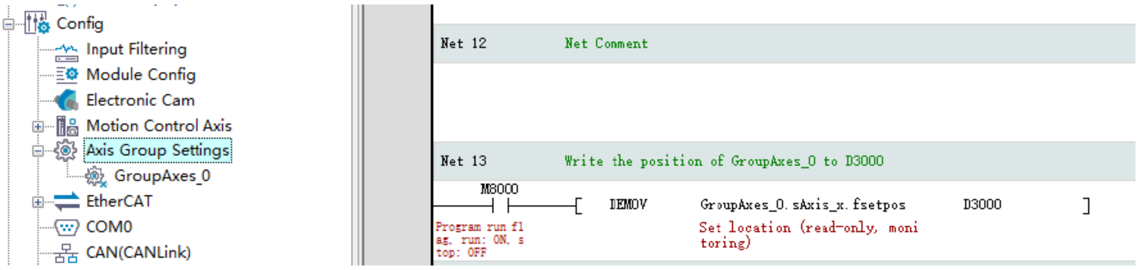
Name	Type	Description
wAxisID	INT16	Axis ID
wState	INT16	The status of an axis' PLCOpen state machine 0: PowerOff 1: ErrorStop 2: Stopping 3: StandStill 4: DiscreteMotion 5: ContinuousMotion 7: Homing 8: SynchronizedMotion
wErrorCode	INT16	The fault code of an axis
fsetpos	REAL	Position reference
factpos	REAL	Feedback position
fsetvel	REAL	Velocity reference
factvel	REAL	Feedback velocity

This system variable exists in the axis group _sMCGROUP_INFO and is used to represent the state of individual axes within the axis group.



	Element Name	Data Type	Display Format	Comment
1	GroupAxes_0	_sMCGROUP_INFO		
2	wRingPos	INT	Dec	Configuration Number (Read Only, Monitoring)
3	wGroupID	INT	Dec	Axis Group Number (Read Only, Monitoring)
4	wState	INT	Dec	Status (read-only, monitoring)
5	wErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
6	bMotionState	BOOL	Bin	Exercise Status (Read Only, Monitoring)
7	bHaltValid	BOOL	Bin	Pause state (read-only, monitoring)
8	wBufNum	INT	Dec	Number of buffers (read-only, monitoring)
9	sAxis_x	_sGROUPAXIS_INFO		X-axis status (read-only, monitoring)
10	vAxisID	INT	Dec	Axis number (read-only, monitoring)
11	vState	INT	Dec	Status (read-only, monitoring)
12	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
13	fsetpos	REAL	Dec	Set location (read-only, monitoring)
14	factpos	REAL	Dec	Feedback location (read-only, monitoring)
15	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
16	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
17	sAxis_y	_sGROUPAXIS_INFO		Y-axis status (read-only, monitoring)
18	vAxisID	INT	Dec	Axis number (read-only, monitoring)
19	vState	INT	Dec	Status (read-only, monitoring)
20	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
21	fsetpos	REAL	Dec	Set location (read-only, monitoring)
22	factpos	REAL	Dec	Feedback location (read-only, monitoring)
23	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
24	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
25	sAxis_z	_sGROUPAXIS_INFO		z-axis status (read-only, monitoring)
26	vAxisID	INT	Dec	Axis number (read-only, monitoring)
27	vState	INT	Dec	Status (read-only, monitoring)
28	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
29	fsetpos	REAL	Dec	Set location (read-only, monitoring)
30	factpos	REAL	Dec	Feedback location (read-only, monitoring)
31	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
32	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
33	sAxis_a	_sGROUPAXIS_INFO		Auxiliary Axis Status (Read Only, Monitoring)
34	vAxisID	INT	Dec	Axis number (read-only, monitoring)
35	vState	INT	Dec	Status (read-only, monitoring)
36	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
37	fsetpos	REAL	Dec	Set location (read-only, monitoring)
38	factpos	REAL	Dec	Feedback location (read-only, monitoring)

For example, write the position reference of the X-axis into the D3000 in the PLC:



```
Net 12          Net Comment

Net 13          Write the position of GroupAxes_0 to D3000
   M8000
---| |---------[  DEMOV    GroupAxes_0.sAxis_x.fsetpos      D3000      ]
Program run fl             Set location (read-only, moni
ag. run: ON, s             toring)
top: OFF
```

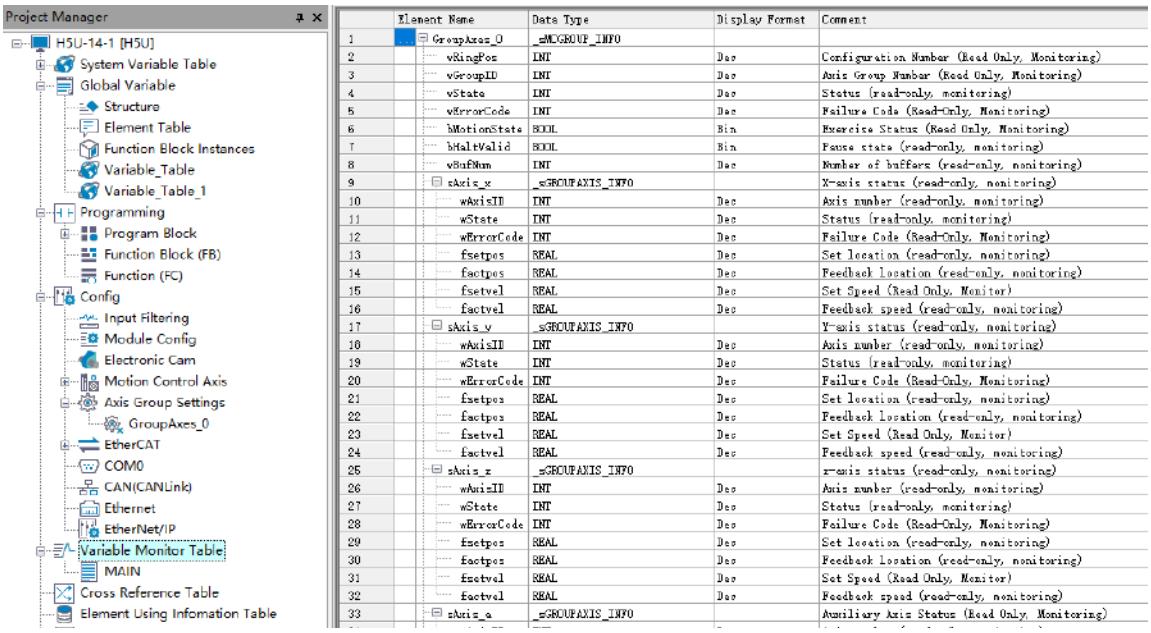
14.5.2 _sMCGROUP_INFO for Axis Group Status

Name	Type	Description
wRingPos	INT16	Axis group number
wGroupID	INT16	Axis number

Name	Type	Description
wState	INT16	Axis group status 0: Init The axis configuration in the axis group is not completed. 1: Disabled Not all axes in the axis group are enabled. 2: Single Stop An axis in the axis group calls the instruction MC_Gtop. 3: Single Homing An axis in the axis group calls the instruction MC_Home. 4: Single motion An axis in the axis group calls single-axis motion instructions such as MC_MoveAbsolute. 5: ErrorStop An axis in the axis group is in a fault state. 6: StandStill All axes in the axis group are in the StandStill state. 7: Stopping The instruction MC_GroupStop is called. 8: Synchronous Motion A linear interpolation or circular interpolation instruction is called.
wErrorCode	INT16	Fault code
bMotionState	BOOL	Motion status FALSE: Not in motion TRUE: In motion
bHaltValid	BOOL	Halt status FALSE: Halt not applied TRUE: Halt applied
wBufNum	INT16	The number of buffered curves
sAxis_x	_sGROUPAXIS_INFO	The status of the X-axis
sAxis_y	_sGROUPAXIS_INFO	The status of the Y-axis
sAxis_z	_sGROUPAXIS_INFO	The status of the Z-axis
sAxis_a	_sGROUPAXIS_INFO	The status of the auxiliary axis
fSetvel	REAL	Velocity reference In linear interpolation mode, it indicates the interpolation velocity of a space straight line. In circular interpolation mode, it indicates the linear velocity of a circular arc.
fSetacc_dec	REAL	Acceleration/deceleration reference Indicates the change rate of setvel.
fSetvel_buf	REAL	The velocity reference of a buffered curve In linear interpolation mode, it indicates the interpolation velocity of a space straight line. In circular interpolation mode, it indicates the linear velocity of a circular arc.

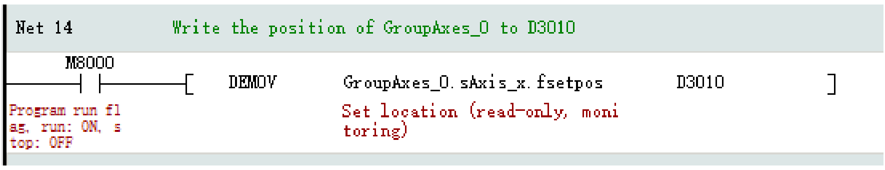
Name	Type	Description
fSetacc_dec_buf	REAL	The acceleration/deceleration reference of a buffered curve Indicates the change rate of fSetvel_buf
fSetdis	REAL	Distance reference In linear interpolation mode, it indicates the distance at which a space straight line moves after the instruction is executed. In circular interpolation mode, it indicates the length of a circular arc in which the circular arc moves after the instruction is executed.
fLeftdis	REAL	Left distance In linear interpolation mode, it indicates the left distance for this section of a space straight line after the instruction is executed. In circular interpolation mode, it indicates the length of a space circular arc left after the instruction is executed.
fCenter_x	REAL	The coordinates of point X at the center of a circular arc during circular interpolation
fCenter_y	REAL	The coordinates of point Y at the center of a circular arc during circular interpolation
fCenter_z	REAL	The coordinates of point Z at the center of a circular arc during circular interpolation
fRadius	REAL	The radius of a circular arc during circular interpolation
fStartAng	REAL	The start angle during circular interpolation
fSetAng	REAL	The motion angle during circular interpolation

This system variable is used to indicate the status of the entire axis group:



	Element Name	Data Type	Display Format	Comment
1	GroupAxes_0	_sMCGROUP_INFO		
2	vRingPos	INT	Dec	Configuration Number (Read Only, Monitoring)
3	vGroupID	INT	Dec	Axis Group Number (Read Only, Monitoring)
4	vState	INT	Dec	Status (read-only, monitoring)
5	vErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
6	bMotionState	BOOL	Bin	Exercise Status (Read Only, Monitoring)
7	bHaltValid	BOOL	Bin	Pause state (read-only, monitoring)
8	vBufNum	INT	Dec	Number of buffers (read-only, monitoring)
9	sAxis_x	_sGROUPAXIS_INFO		X-axis status (read-only, monitoring)
10	wAxisID	INT	Dec	Axis number (read-only, monitoring)
11	wState	INT	Dec	Status (read-only, monitoring)
12	wErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
13	fsetpos	REAL	Dec	Set location (read-only, monitoring)
14	factpos	REAL	Dec	Feedback location (read-only, monitoring)
15	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
16	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
17	sAxis_y	_sGROUPAXIS_INFO		Y-axis status (read-only, monitoring)
18	wAxisID	INT	Dec	Axis number (read-only, monitoring)
19	wState	INT	Dec	Status (read-only, monitoring)
20	wErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
21	fsetpos	REAL	Dec	Set location (read-only, monitoring)
22	factpos	REAL	Dec	Feedback location (read-only, monitoring)
23	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
24	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
25	sAxis_z	_sGROUPAXIS_INFO		z-axis status (read-only, monitoring)
26	wAxisID	INT	Dec	Axis number (read-only, monitoring)
27	wState	INT	Dec	Status (read-only, monitoring)
28	wErrorCode	INT	Dec	Failure Code (Read-Only, Monitoring)
29	fsetpos	REAL	Dec	Set location (read-only, monitoring)
30	factpos	REAL	Dec	Feedback location (read-only, monitoring)
31	fsetvel	REAL	Dec	Set Speed (Read Only, Monitor)
32	factvel	REAL	Dec	Feedback speed (read-only, monitoring)
33	sAxis_a	_sGROUPAXIS_INFO		Auxiliary Axis Status (Read Only, Monitoring)

For example, write the X-axis coordinates of the center of an axis group to D3010:

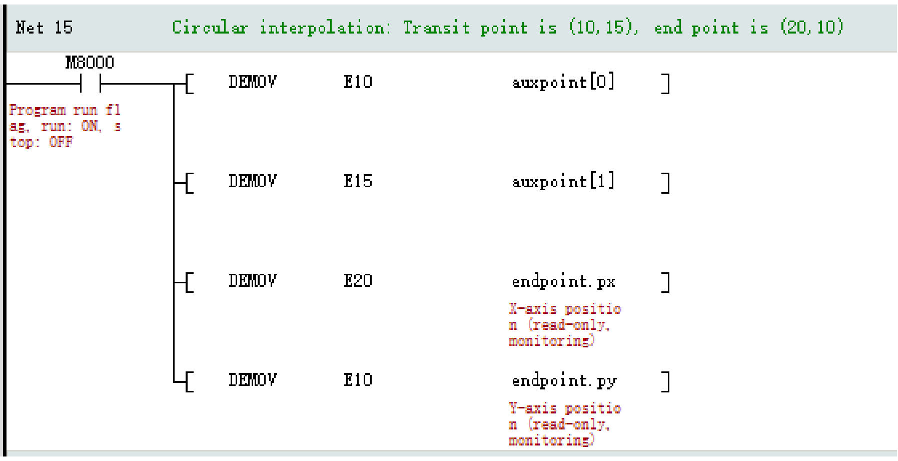

Net 14
Write the position of GroupAxes_0 to D3010
M8000
DEMOV
GroupAxes_0.sAxis_x.fsetpos
D3010
Program run flag, run: ON, stop: OFF
Set location (read-only, monitoring)

14.5.3 _sGROUPPOS_INFO for Target Positions of Coordinate Axes within Axis Group

Name	Type	Description
px	REAL	The position of the X-axis
py	REAL	The position of the Y-axis
pz	REAL	The position of the Z-axis
pa	REAL	The position of the auxiliary axis

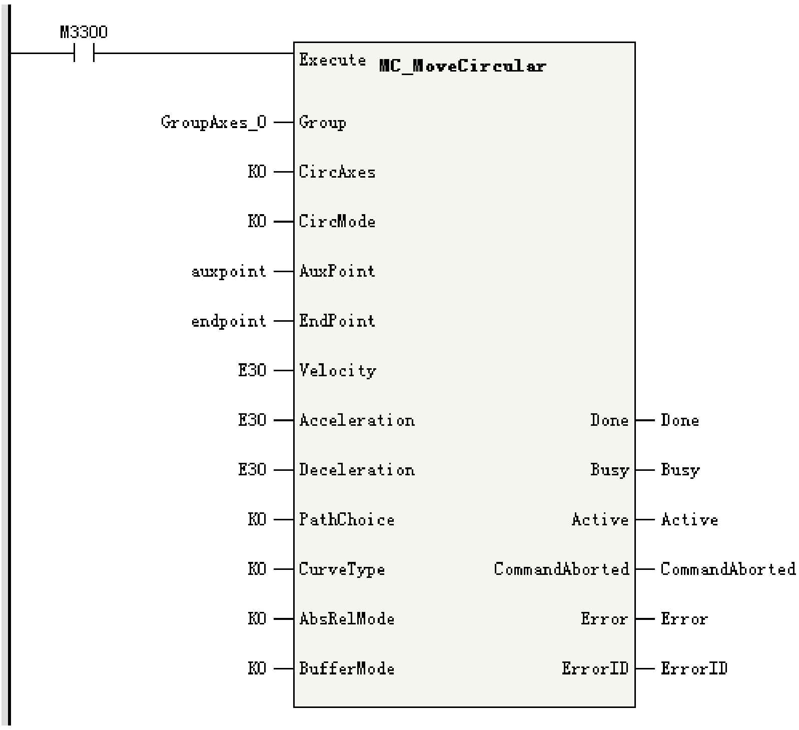
This structure sets the target position of a circular arc as an input parameter to the MC_MoveCircular.

1. Create a global variable
2. Assign values to the global variable



```
Net 15          Circular interpolation: Transit point is (10,15), end point is (20,10)
      M8000
  ----| |------+-[  DEMOV    E10          auxpoint[0]     ]
Program run fl |
ag, run: ON, s |
top: OFF       |
               +-[  DEMOV    E15          auxpoint[1]     ]
               |
               +-[  DEMOV    E20          endpoint.px     ]
               |                          X-axis positio
               |                          n (read-only,
               |                          monitoring)
               +-[  DEMOV    E10          endpoint.py     ]
                                          Y-axis positio
                                          n (read-only,
                                          monitoring)
```

3. Call the instruction MC_MoveCircular



```
      M3300
  ----| |-----------------Execute  MC_MoveCircular

          GroupAxes_0 --- Group
                   K0 --- CircAxes
                   K0 --- CircMode
             auxpoint --- AuxPoint
             endpoint --- EndPoint
                  E30 --- Velocity
                  E30 --- Acceleration                  Done --- Done
                  E30 --- Deceleration                  Busy --- Busy
                   K0 --- PathChoice                  Active --- Active
                   K0 --- CurveType           CommandAborted --- CommandAborted
                   K0 --- AbsRelMode                   Error --- Error
                   K0 --- BufferMode                 ErrorID --- ErrorID
```

14.6 Fault Codes

When a fault occurs during use of the interpolation functions, see the fault codes listed in the following table for troubleshooting.

Fault Code	Description	Solution
9400	The number of axis groups exceeds the maximum value.	Check whether the number of axis groups is greater than 8.
9401	An axis in the axis group is in a fault state.	Check whether an axis in the axis group has entered the ErrorStop state. Troubleshoot the fault based on the fault code of each axis.
9402	The number of buffered interpolation instructions is greater than 8.	Check whether the number of buffered interpolation instructions is greater than 8.
9403	The axis is reused.	Locate the reused axis and replace it with an unused axis.
9404	Failed to create the axis group.	The X-axis and Y-axis cannot be empty. Check whether the X-axis or Y-axis does not exist or is not specified.
9405	The specified Z-axis does not exist.	Check whether the axis specified by AxisID_z exists.
9406	The specified auxiliary axis does not exist.	Check whether the axis specified by AxisID_a exists.
9407	The axis group ID is duplicate.	Check whether GroupID is duplicate.
9408	Axis configuration failed.	Check whether any axis in the axis group fails to be configured. If yes, check whether the PCB software and the background match.
9409	The axis ID is less than 0.	Check whether the ID of an axis in the axis group is less than 0.
9410	The axis group is not released because the MC_SetAxesGroup instruction is triggered repeatedly in a short time period.	Do not re-trigger the MC_SetAxesGroup instruction while its Busy signal output is still active.
9411	Instruction MC_GroupStop was interrupted.	Check whether an instruction with higher priority is called while the MC_GroupStop instruction is still active.
9412	The circular interpolation instruction CircAxes is out of range.	Check whether the value of CircAxes of the circular interpolation instruction is out of range.
9413	The circular interpolation instruction CircMode is out of range.	Check whether the value of CircMode of the circular interpolation instruction is out of range.
9414	The circular interpolation instruction PathChoice is out of range.	Check whether the value of PathChoice of the circular interpolation instruction is out of range.
9415	The stop instruction StopMode is out of range.	Check whether the value of StopMode of the stop instruction is out of range.
9416	The X-axis is set to ring mode.	Do not set the motion control axis to the ring mode in an interpolation instruction.
9417	The Y-axis is set to ring mode.	Do not set the motion control axis to the ring mode in an interpolation instruction.
9418	The Z-axis is set to ring mode.	Do not set the motion control axis to the ring mode in an interpolation instruction.
9419	The auxiliary axis is set to ring mode.	Do not set the motion control axis to the ring mode in an interpolation instruction.
9420	The circular interpolation instruction is triggered repeatedly.	Do not re-trigger the same circular interpolation instruction while its Busy signal output is still active.
9421	The linear interpolation instruction is triggered repeatedly.	Do not re-trigger the same linear interpolation instruction while its Busy signal output is still active.
9422	Failed to obtain the axis group.	Check whether the axis group specified by GroupID has been created by calling MC_SetAxesGroup.

Fault Code	Description	Solution
9423	Axis configuration failed.	Check whether an instruction is triggered when axis configuration is not completed. Check whether the communication state of all axes in the axis group is "Axis ready".
9424	An axis is disabled.	Do not call the interpolation instruction when any axis is in Disabled state.
9425	An axis is executing single-axis motion instructions.	Do not call the interpolation instruction when any axis is executing single-axis motion instructions and not in StandStill state.
9426	An axis is in Stopping state.	Do not call the interpolation instruction when any axis is in Stopping state after the MC_Stop instruction is executed.
9427	The axis group is in a stopped state.	Do not call the interpolation instruction while the MC_GroupStop instruction is still active.
9428	An axis is in Homing state.	Do not call the interpolation instruction when any axis is in Homing state after the MC_Home instruction is executed.
9429	An axis is executing the position setting instruction.	Do not call the interpolation instruction when any axis is setting the current position by executing the MC_SetPosition instruction.
9430	An axis is in commissioning state.	Do not call the interpolation instruction when any axis is in commissioning state.
9431	An axis enters the commissioning state during interpolation, which interrupts the instruction execution of other axes.	Check whether any axis enters the commissioning state during interpolation.
9432	Failed to request the memory.	Check whether the memory runs out. Contact the manufacturer.
9433	The target velocity is 0 or less.	Ensure that the target velocity of the instruction is greater than 0.
9434	The target acceleration rate is 0 or less.	Ensure that the target acceleration rate of the instruction is greater than 0.
9435	The target deceleration rate is 0 or less.	Ensure that the target deceleration rate of the instruction is greater than 0.
9436	The curve type is set beyond the range.	Check whether the curve type is set to a value other than the T-shaped curve for the interpolation instruction.
9437	AbsRelMode is set incorrectly.	Check whether the parameter is set to a value other than the absolute positioning and relative positioning modes.
9438	BufferMode is set incorrectly.	Check whether the value of BufferMode is out of range.
9439	InsertMode is set incorrectly.	Check whether the value of InsertMode is proper.
9440	An axis stops due to a fault.	Locate the faulty axis and rectify the fault based on the fault code.
9441	Instruction MC_GroupStop is called repeatedly.	Do not re-trigger an MC_GroupStop instruction or call other MC_GroupStop instructions while an MC_GroupStop instruction is still active.
9442	The data buffer is not empty.	Contact Inovance for technical support.
9443	No circle can be drawn.	-
9444	The start point, end point, and border point in the circular interpolation instruction are the same point, and no circle can be drawn.	Check the input parameters of the circular interpolation instruction and ensure that the start point, end point, and border point can form a circle.
9445	The instruction buffer is full.	Contact Inovance for technical support.
9446	The velocity of the X-axis exceeds the maximum allowable velocity.	Ensure that the target velocity of the X-axis is not greater than the maximum allowable velocity.
9447	The Y-axis exceeds the maximum velocity.	Ensure that the target velocity of the Y-axis is not greater than the maximum allowable velocity.
9448	The Z-axis exceeds the maximum velocity.	The velocity of the Z-axis exceeds the maximum allowable velocity.

Fault Code	Description	Solution
9449	The auxiliary axis exceeds the maximum velocity.	Ensure that the target velocity of the auxiliary axis is not greater than the maximum allowable velocity.
9450	Failed to obtain the number of axis groups.	Update the background software to the latest version.
9451	Internal fault	Contact the manufacturer.
9452	The instruction is called when the axis is in StandStill state.	Do not call this instruction when the axis is in StandStill state.
9453	The maximum allowable velocity is exceeded.	Ensure that the target velocity of the instruction is not greater than the maximum velocity specified on the axis group configuration interface.
9454	The maximum allowable acceleration/ deceleration rate is exceeded.	Ensure that the target acceleration (deceleration) rate of the instruction is not greater than the maximum acceleration (deceleration) rate specified on the axis group configuration interface.
9455	Axis group becomes faulty due to an error reported by the linear interpolation instruction.	Identify the first linear interpolation instruction that reports the error and troubleshoot the fault based on the fault code.
9456	The axis group becomes faulty due to an error reported by the circular interpolation instruction.	Identify the first circular interpolation instruction that reports the error and troubleshoot the fault based on the fault code.
9457	The axis group becomes faulty due to an error reported by the axis group stop instruction.	Identify the first axis group stop instruction that reports the error and troubleshoot the fault based on the fault code.
9458	The axis group becomes faulty due to an error reported by the axis group pause instruction.	Identify the first axis group pause instruction that reports the error and troubleshoot the fault based on the fault code.

15 Bus Encoder Axes

15.1 Introduction to Bus Encoder Axes

Bus encoder axes support a maximum input pulse frequency of 200 kHz is supported, use the GR10-2HCE module as the driver, and can count three forms of pulses: phase A/B, pulse+direction, and CW/CCW.
Up to eight bus encoder axes can be configured. One GR10-2HCE module has two channels, each of which can be assigned for one axis.

Bus encoder axes can also work with the DI and DO terminals of the GR10-2HCE module to implement position presetting, probe, gating, and comparison output.

The feedback position of the GR10-2HCE module in pulses is sent to the PLC through PDOs, and the feedback position within the PLC is finally provided to users in the form of REAL data type through the conversion of gear ratios, which can also be used as the master axis of a cam or gear.

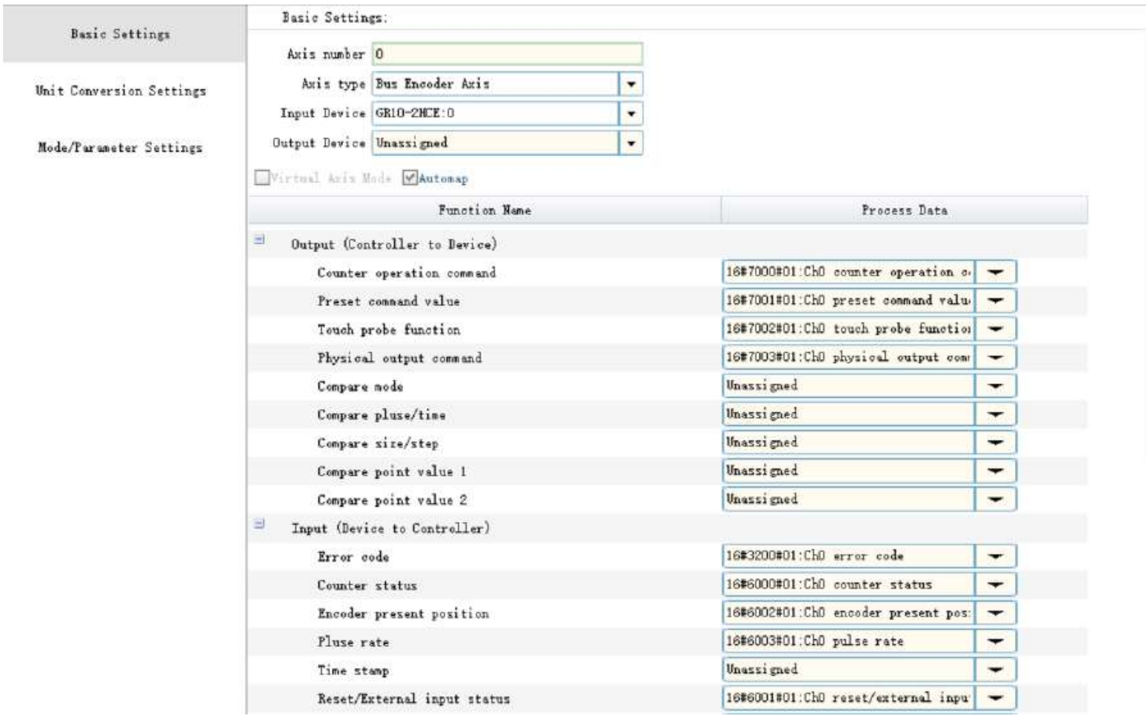
Note

The software of the GR10-2HCE module must be version V2.2.0.0 or later versions.

15.2 Software Configuration

15.2.1 Basic Settings

Bus encoder axes represent a category of motion control axes that require the user to enable EtherCAT communication and add the GR10–2HCE module before enabling a bus encoder axis.
The configuration interface of a bus encoder axis is shown in the following figure.



Basic Settings:

Axis number 0

Axis type Bus Encoder Axis

Input Device GR10-2HCE:0

Output Device Unassigned

□ Virtual Axis Mode ☑ Automap

Function Name	Process Data
Output (Controller to Device)	
Counter operation command	16#7000#01:Ch0 counter operation c
Preset command value	16#7001#01:Ch0 preset command valu
Touch probe function	16#7002#01:Ch0 touch probe functio
Physical output command	16#7003#01:Ch0 physical output com
Compare mode	Unassigned
Compare pluse/time	Unassigned
Compare size/step	Unassigned
Compare point value 1	Unassigned
Compare point value 2	Unassigned
Input (Device to Controller)	
Error code	16#3200#01:Ch0 error code
Counter status	16#6000#01:Ch0 counter status
Encoder present position	16#6002#01:Ch0 encoder present pos:
Pluse rate	16#6003#01:Ch0 pulse rate
Time stamp	Unassigned
Reset/External input status	16#6001#01:Ch0 reset/external inpu

Operating panel description

- Axis No: Each bus encoder axis has an axis number, which is the unique identifier of the encoder axis and is automatically assigned when the configuration is established.
- Axis type: Bus encoder axis.
- Input device: With H5U series PLC as an example, a bus encoder axis needs to be used with the GR10-2HCE module. Each GR10-2HCE module supports two counting channels. You can select a channel as required during device selection.
- Output device: Not supported.
- Automatic mapping: After "Automatic mapping" is ticked, the I/O variable of an axis and the PDO of the GR10-2HCE module will be automatically associated. If this option is not ticked, you can manually configure the mapping.

Variable mapping

The correspondence between the I/O variables of a bus encoder axis and the PDO of the GR10-2HCE module is shown in the following table.

Table 15–1 Output variable mapping

Variable	Channel 0 Object Dictionary	Channel 1 Object Dictionary	Description
Counter operation command	7000h:1	7000h:2	Encoder control word
Preset command value	7001h:1	7001h:2	Preset position
Touch probe function	7002h:1	7002h:2	Probe control word
Physical output command	7003h:1	7003h:2	DO terminal control word
Compare mode	7003h:3	7003h:11	Comparison output mode
Compare pulse/time	7003h:5	7003h:13	Comparison output pulses/time
Compare size/step	7003h:6	7003h:14	Comparison output array length/step

Variable	Channel 0 Object Dictionary	Channel 1 Object Dictionary	Description
Compare point value 1	7008h:1	7009h:1	Comparison output value 1
Compare point value 2	7008h:2	7009h:2	Comparison output value 2

Table 15–2 Input variable mapping

Variable	Channel 0 Object Dictionary	Channel 1 Object Dictionary	Description
Error code	3200h:1	3200h:2	Fault code
Counter status	6000h:1	6000h:2	Counter status
Encoder present position	6002h:1	6002h:2	Encoder feedback position
Pulse rate	6003h:1	6003h:2	Pulse frequency
Time stamp	6009h:1	6009h:2	Time stamp
Physical input status	6001h:1	6001h:2	DI terminal status
Touch probe status	6004h:1	6004h:2	Probe status
Touch probe pos 1 pos value	6005h:1	6005h:2	Probe 1 position on the rising edge
Touch probe pos 1 neg value	6006h:1	6006h:2	Probe 1 position on the falling edge
Touch probe pos 2 pos value	6007h:1	6007h:2	Touch probe 2 positive edge
Touch probe pos 2 neg value	6008h:1	6008h:2	Probe 2 position on the falling edge
Physical output status	600Eh:1	600Eh:2	DO terminal status
Compare error code	6003h:3	6003h:5	Comparison output fault code
Current compare number/ position	6003h:7	6003h:9	Comparison output current group subscript/position

15.2.2 Unit Conversion

The following table lists parameters that need to be set for unit conversion.

Parameter	Function
The number of pulses in one turn by motor/encode	Set the number of pulses required for the motor to rotate one turn according to the encoder resolution.
With gear change mechanisms	Specify whether gear change mechanisms are in use or not.
Amount of movement in one turn by motor/encoder	The workpiece movement amount per turn of the motor when no gear change mechanism is in use
Amount of movement of the worktable in a circle	The workpiece movement amount per turn of the worktable when gear change mechanisms are in use
Gear ratio on the motor/encoder side	Set a gear ratio on the motor/encoder side.
Gear ratio on the worktable side	Set a gear ratio on the worktable side.

The module GR10-2HCE counts in pulses, while the encoder axis instructions use common measurement units in operation, such as millimeters, degrees, and inches, which are known as user units (Unit). The conversion between the two units is divided into the following modes:

1. Without gear change mechanisms
   When gear change mechanisms are not in use, the conversion equation from user unit to pulse unit is as follows.

$$\text{Number of pulses (unit: pulse)} = \frac{\text{Number of pulses per revolution of the motor/encoder [DINT]}}{\text{Distance per revolution of the workbench [REAL]}} \text{ x Distance (unit: Unit)}$$

Take the Inovance 23-bit encoder as an example. The set parameters are as follows.

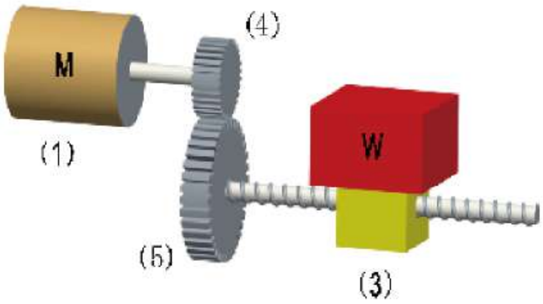
Number of pulses in one turn by motor/encode = 8388608

Workpiece movement amount per turn of the motor/encoder = 1

When the motor rotates by 10 revolutions, the number of pulses counted by the 2HCE module is 83886080, and the encoder axis instruction counts to the position increment of 10 Unit.

2. With gear change mechanisms

- Typical working condition in linear mode is shown in the following figure.

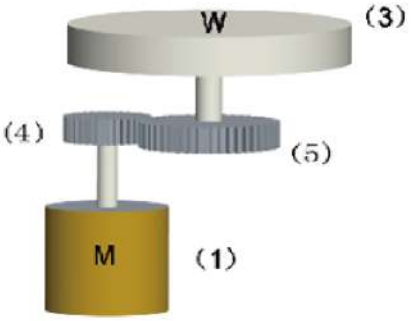


Where, (1) is the motor/encoder; (3) is the worktable; (4) is the gear ratio denominator; (5) is the gear ratio numerator.

The calculation equation from user unit to pulse unit is as follows.

$$\text{Number of pulses (unit: pulse)} = \frac{\text{Number of pulses per revolution of the motor/encoder [DINT] x Gear ratio numerator [DINT]}}{\text{Distance per revolution of the workbench [REAL] x Gear ratio denominator [DINT]}} \text{ x Distance (unit: Unit)}$$

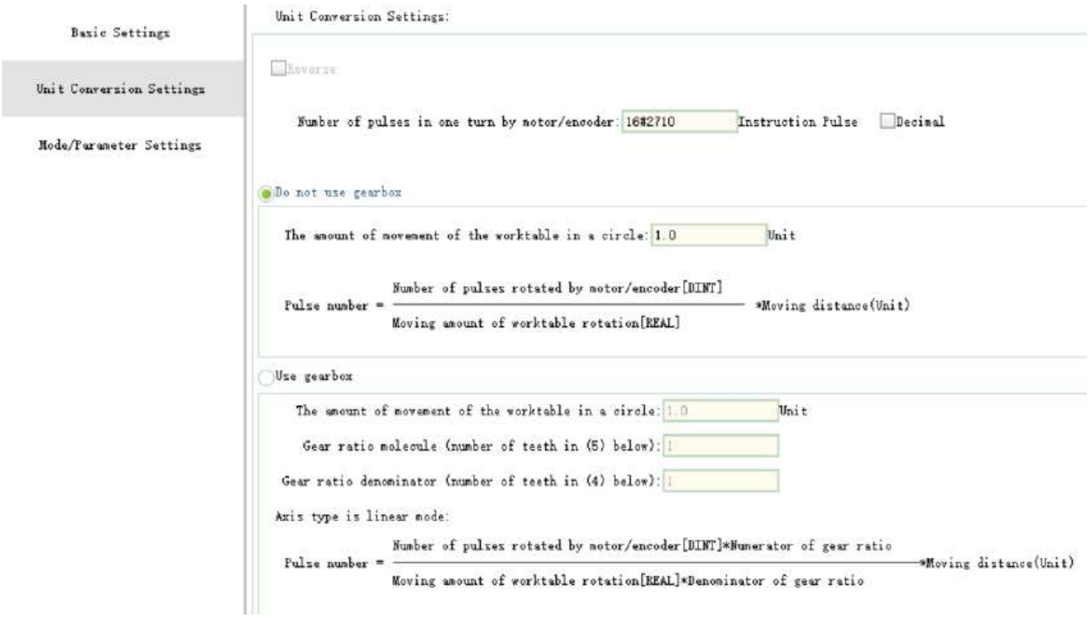
- Typical working condition in rotary mode is shown in the following figure.



Where, (1) is the motor/encoder; (3) is the worktable; (4) is the gear ratio denominator; (5) is the gear ratio numerator.

The calculation equation from user unit to pulse unit is as follows.

$$\text{Number of pulses (unit: pulse)} = \frac{\text{Number of pulses per revolution of the motor/encoder [DINT] x Gear ratio numerator [DINT]}}{\text{Distance per revolution of the workbench [REAL] x Gear ratio denominator [DINT]}} \text{ x Distance (unit: Unit)}$$



Unit Conversion Settings:

□ Reverse

Number of pulses in one turn by motor/encoder: 16#2710 Instruction Pulse □ Decimal

● Do not use gearbox

The amount of movement of the worktable in a circle: 1.0 Unit

$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]}}{\text{Moving amount of worktable rotation[REAL]}} \text{*Moving distance(Unit)}$$

○ Use gearbox

The amount of movement of the worktable in a circle: 1.0 Unit

Gear ratio molecule (number of teeth in (5) below): 1

Gear ratio denominator (number of teeth in (4) below): 1

Axis type is linear mode:

$$\text{Pulse number} = \frac{\text{Number of pulses rotated by motor/encoder[DINT]*Numerator of gear ratio}}{\text{Moving amount of worktable rotation[REAL]*Denominator of gear ratio}} \text{*Moving distance(Unit)}$$

15.2.3 Mode/Parameter Settings

15.2.3.1 Configuration Interface

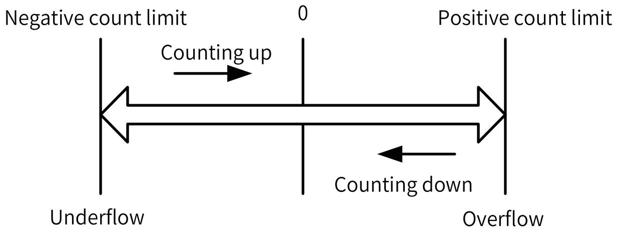
The following figure shows the mode and parameter configuration interface.

15.2.3.2 Selection of Linear or Rotary Mode

Linear mode

You can enable or disable software limits.

When the software limit is enabled, positive and negative limits can be set, and the counter will stop counting when reaching the limits, and display an out-of-limit mark. When the positive limit is reached, the output of the encoder axis system variable bPLimit is valid. When the negative limit is reached, the output of the encoder axis system variable bNLimit is valid.



If the software limit is not enabled, the count value of GR10-2HCE ranges between –2147483648 and +2147483647, jumps to –2147483648 when the positive count reaches +2147483647, and jumps to +2147483647 when the negative count reaches –2147483648.

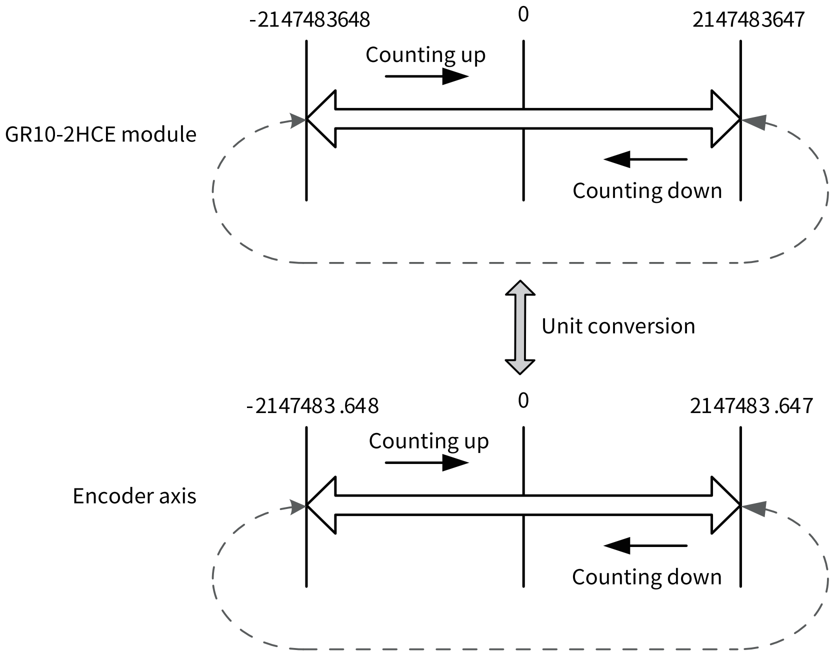
In this case, the range that the bus encoder axis can count needs to be calculated according to the scheme described in *“15.2.2 Unit Conversion” on page 491*.

Example

Example

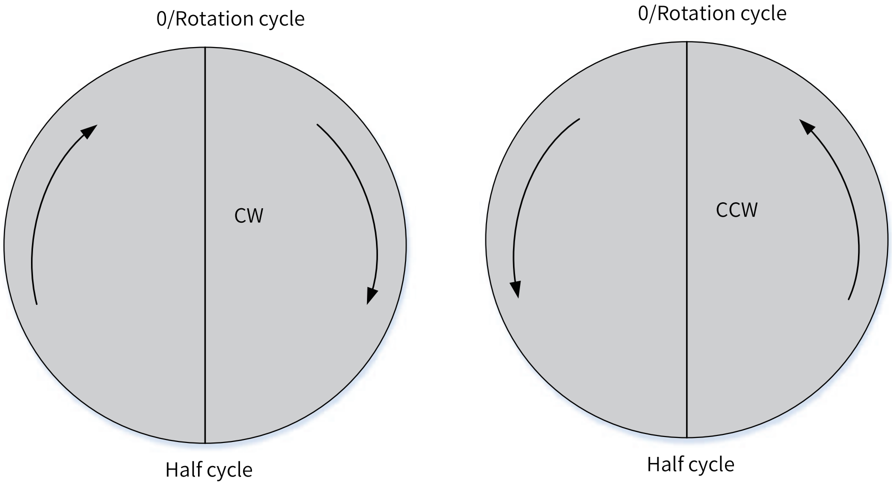
Number of pulses in one turn by motor/encoder = 1000

Amount of movement in one turn by motor/encoder = 1



Rotary mode

In rotary mode, you can set the ring period. The counter counts in a reciprocating cycle between 0 and the ring period.

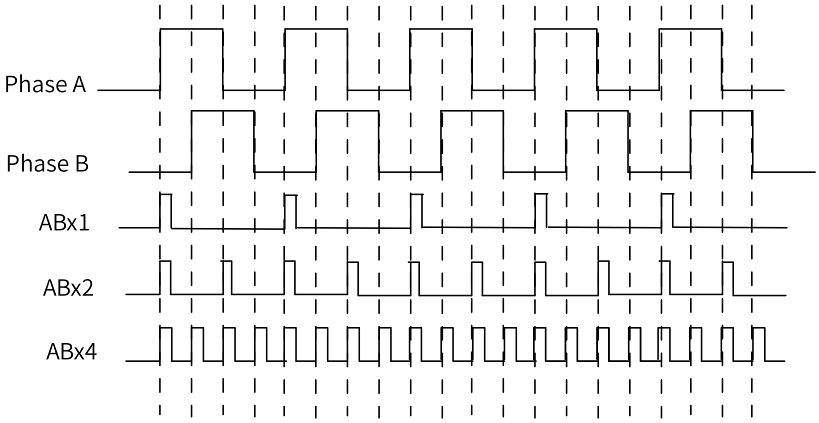


15.2.3.3 Counter Mode Selection

Count modes

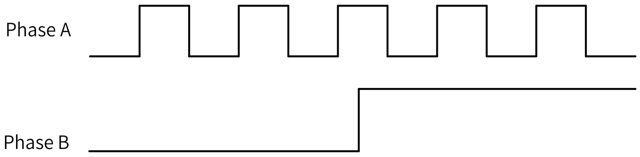
Five count modes are supported: phase A/B 1-frequency multiplication, phase A/B 2-frequency multiplication, phase A/B 4-frequency multiplication, pulse+direction, and CW/CCW.

1. Phase A/B pulse



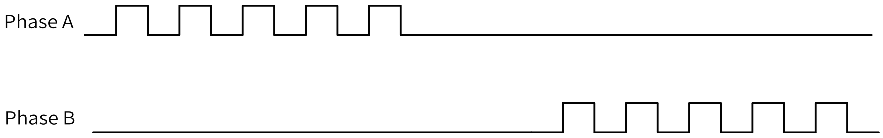
2. Pulse+direction

Phase A is used as the counter's pulse input, and phase B is used as the counter's count direction control input.



3. CW/CCW mode

For CW/CCW mode, under positive logic, the counter counts up on the rising edge that inputs pulse A and counts down on the rising edge that inputs pulse B.



Count logic

Count logic is used to set the logic of the count direction. The logic is as follows.

Pulse Mode	Positive Logic	Negative Logic
Phase A/B	Incremental count if phase A leads phase B Decremental count if phase B leads phase A	Decremental count if phase A leads phase B Incremental count if phase B leads phase A
Pulse+direction	Incremental count if phase B is at a high level Decremental count if phase B is at a low level	Decremental count if phase B is at a high level Incremental count if phase B is at a low level
CW/CCW	Incremental count if it is phase A pulse input Decremental count if it is phase B pulse input	Decremental count if it is phase A pulse input Incremental count if it is phase B pulse input

15.2.3.4 Frequency Sampling Period

It is used to set the calculation period of the pulse frequency.

15.2.3.5 Input Filter Time

It is used to set the filter time of DI terminals and ABZ input signals.

15.2.3.6 Input Terminal Function Selection

Each counting channel can be independently configured with four DI terminals. The functions that can be allocated for each terminal are shown in the following table.

Terminal	Optional Function	Default
Xn0	0: Common input 1: Probe function 1 3: Reset 4: Preset 5: Gating	Probe function 1
Xn1	0: Common input 2: Probe function 2 3: Reset 4: Preset 5: Gating	Probe function 2
Xn2	0: Common input 3: Reset 4: Preset 5: Gating	Common input
Xn3	0: Common input 3: Reset 4: Preset 5: Gating	Common input

- Common input

When a terminal is allocated with the common input function, its status can be obtained through the system variable iDIStatus or the instruction ENC_ReadStatus.

- Probe function
  When a terminal is allocated with the probe function, you need to call the instruction ENC_TouchProbe to implement the probe function. For details, see the explanation of this instruction.
- Preset function
  When a terminal is allocated with the preset function, you need to call the instruction ENC_Preset to implement the counter preset function. When the input signal of the DI terminal is active, the value of the counter will be set to the preset value of parameter Position of the instruction ENC_Preset.

Note

If multiple terminals are configured with a preset function, the preset function is activated when one of the input terminals receives an active signal.

- Gating function
  If the DI terminal signal is set as a gating signal, the gating function is enabled, in which case the counter is enabled to start counting only after the parameter Enable of ENC_Counter is ON and the gating input signal is active. If the DI terminal signal is not set as a gating signal, and the gating function is not enabled, then the counter starts counting after the parameter Enable of ENC_Counter is ON.

Note

If multiple terminal input signals are used as gate signals, these gate signals must be active at the same time for the counter to start counting.

15.2.3.7 Output Terminal Function Selection

Each counting channel can be independently configured with three DO terminals. The functions that can be allocated for each terminal are shown in the following table.

Terminal	Optional Function	Default
Yn0	0: Common output 3: One-dimensional comparison output 4: Two-dimensional comparison output (only supported by channel 0)	3: One-dimensional comparison output
Yn1	0: Common output	0: Common output
Yn2	0: Common output	0: Common output

- Common output
  When a DO terminal is set to a common output terminal, its status can be controlled by the instruction ENC_DigitalOutput. See description of the instruction ENC_DigitalOutput for detailed usage.
- One-dimensional comparison output mode

When a terminal is allocated with the one-dimensional comparison output function, the step comparison output or array comparison output functions can be performed. See instructions ENC_StepCompare and ENC_ArrayCompare for detailed guidance.

- Two-dimensional comparison output mode
When a bus encoder axis is bound to channel 0 of the GR10-2HCE module, the output terminal Y00 can be allocated with the two-dimensional comparison output function. See the instruction ENC_GroupArrayCompare for detailed usage.

15.3 System Variables

Table 15–3 _sPoint2D: Structure of coordinate points in a two-dimensional coordinate system

Variable	Type	Function
px	REAL	Coordinate points on X-axis
py	REAL	Coordinate points on Y-axis

Table 15–4 _sENC_PDO: PDOs of the bus encoder axis

Variable	Type	Function
iControlWord	INT	Control word
iStatusWord	INT	Status word
dPresetPosition	DINT	Preset position
dActualPositionValue	DINT	Feedback position
dActualVelocityValue	DINT	Feedback velocity
dTimeStamp	DINT	Time stamp
iDOControlWord	INT	DO terminal control word
iDOStatusWord	INT	DO terminal status word
iCompareMode	INT	Comparison output mode
dCompare_Pluse_Time	DINT	Comparison output pulses/time
dCompare_Size_Step	DINT	Comparison output array length/step
dCompareValue_1	DINT	Comparison output comparison value 1
dCompareValue_2	DINT	Comparison output comparison value 2
dCompareAct_Num_Pos	DINT	Next position/comparison value of comparison output
iCompareErrorCode	INT	Fault code of comparison output
iDIStatus	INT	DI terminal status
iTouchProbeFunc	INT	Probe control word
iTouchProbeStatus	INT	Probe status word
dPos1PosValue	DINT	Probe 1 position latched on the rising edge
dPos2PosValue	DINT	Probe 2 position latched on the rising edge
dPos1NegValue	DINT	Probe 1 position latched on the falling edge
dPos2NegValue	DINT	Probe 2 position latched on the falling edge
iErrorCode	INT	Fault code

Table 15–5 _sENC_CONFIG: General configuration parameters for encoder axes

Variable	Type	Function
bLimitEnable	BOOL	Limit enable
fUnits	REAL	Gear ratio

Variable	Type	Function
iRingMode	INT	Linear/ring mode 0: Linear mode 1: Ring mode
fNegLimitPos	REAL	Negative limit in linear mode
fForLimitPos	REAL	Positive limit in linear mode
fRotationPeriod	REAL	Rotation cycle in rotary mode

Table 15–6 _sENC_EXT_AXIS: System variables of the bus encoder axis

Variable	Type	Function
bEnable	BOOL	Count state of the encoder OFF: Stop counting ON: Enable counting
bActDir	BOOL	Count direction OFF: Forward (incremental count) ON: Reverse (decremental count)
bPLimit	BOOL	Arrival at the positive limit OFF: Disabled ON: Enabled
bNlimit	BOOL	Arrival at the negative limit OFF: Disabled ON: Enabled
iConfigAddress	INT	Configuration address
iAxisID	INT	Axis No.
fPosition	REAL	Feedback position (user unit)
fVelocity	REAL	Feedback velocity (user unit)
dPosition	DINT	Feedback position (pulse unit)
dFrequency	DINT	Pulse frequency (pulse unit)
iAxisState	INT	Statuses of the axis 0: Stop counting 1: Fault state of the axis 5: Enable counting
iConfigState	INT	Configuration statuses of the axis 1: Init (axis in the initialization state) 2: Configure finish (reading of configuration data completed) 3: Sync finish (synchronized with EtherCAT tasks) 4: Wait communication (communication with the drive established) 5: Slave ready (initialization completed for the servo drive controlled by axes) 6: Axis ready (communication established)
iAxisError	INT	Axis fault code
iSlaveAxisError	INT	Drive fault code
sCounter	_sENC_CNT	Reservation for compatibility with the local encoder axis

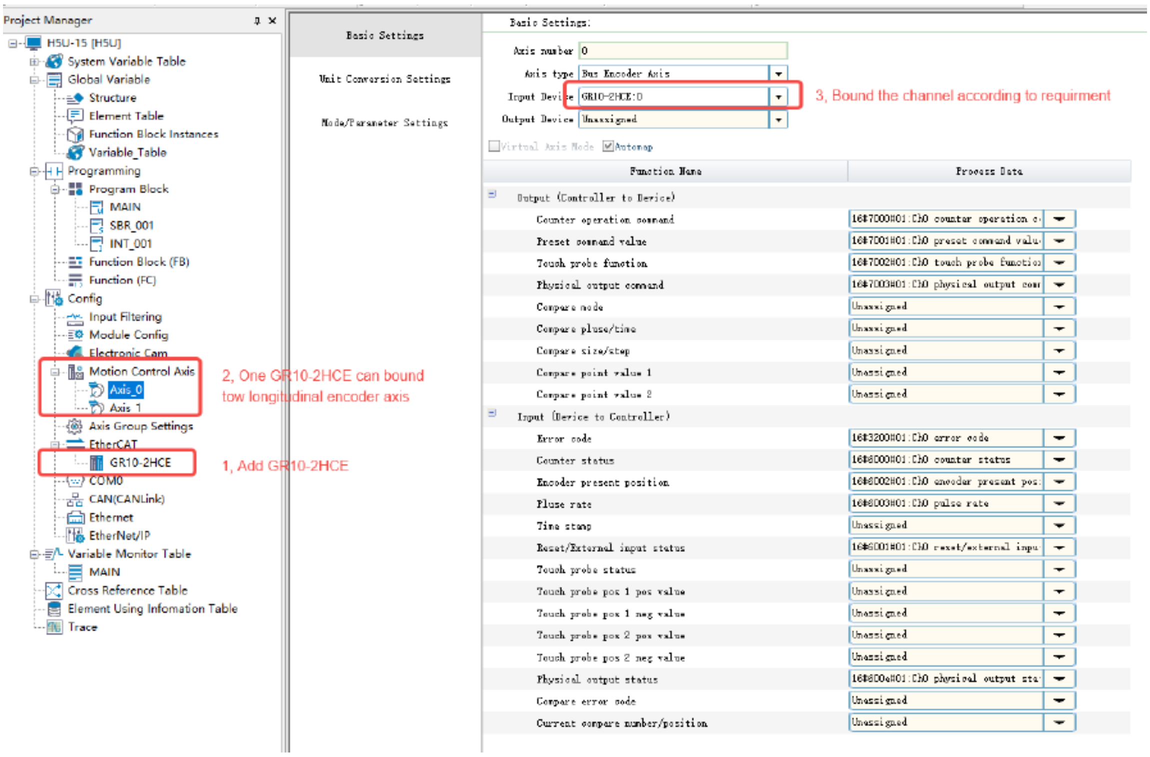
Variable	Type	Function
sReset	_sENC_RST	Reservation for compatibility with the local encoder axis
sPreset	_sENC_PRESET	Reservation for compatibility with the local encoder axis
sProbe	_sENC_PROBE[2]	Reservation for compatibility with the local encoder axis
sMatch	_sENC_MATCH[2]	Reservation for compatibility with the local encoder axis
sPDO	_sENC_PDO	Parameter value area of PDOs
sConfigure	_sENC_CONFIG	Parameter value area of configuration

15.4 Function Demonstration

15.4.1 Establishing the Configuration

In this example, the hardware required is as follows: one H5U and one GR10-2HCE module with its channels CH0 and CH1 connected to the external phase A/B pulse of fixed frequency.

Create a project and add the GR10-2HCE module to the EtherCAT configuration. The two bus encoder axes are automatically added. Among them, Axis_0 is automatically bound to channel CH0 of the GR10-2HCE module, while Axis_1 is automatically bound to channel CH1 of the GR10-2HCE module.



2, One GR10-2HCE can bound tow longitudinal encoder axis

1, Add GR10-2HCE

Basic Settings:

Axis number: 0

Axis type: Bus Encoder Axis

Input Device: GR10-2HCE:0

3, Bound the channel according to requirment

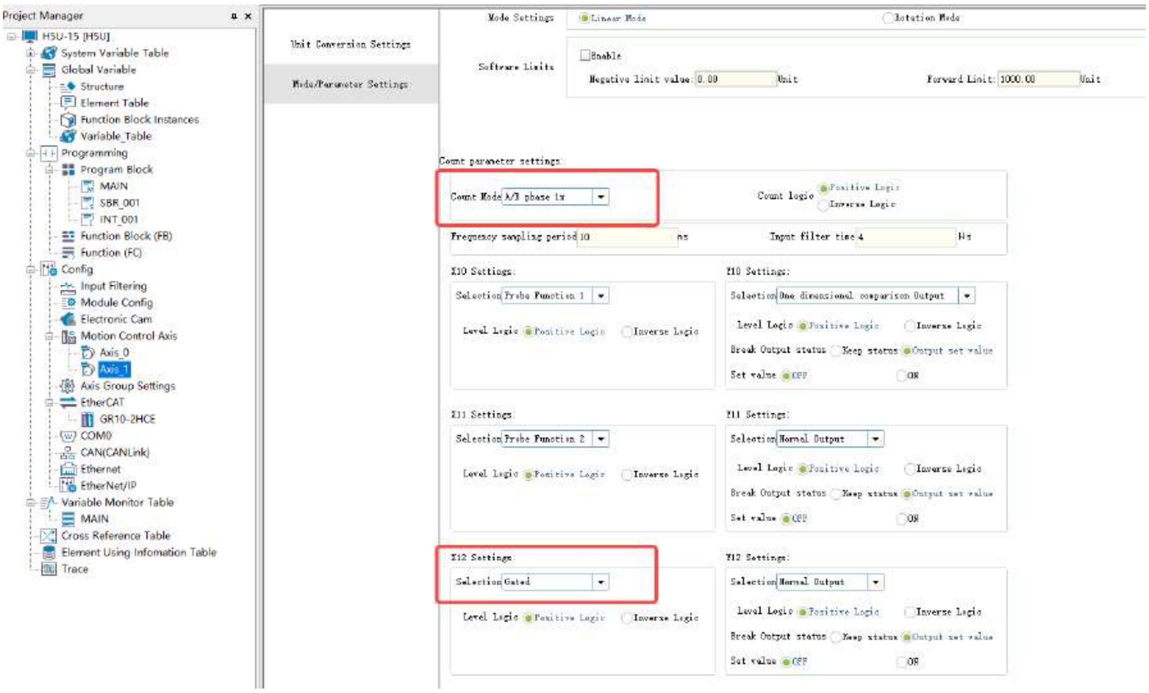
Output Device: Unassigned

□ Virtual Axis Mode ☑ Automap

Function Name	Process Data
Output (Controller to Device)	
Counter operation command	16#7000#01:Ch0 counter operation c
Preset command value	16#7001#01:Ch0 preset command valu
Touch probe function	16#7002#01:Ch0 touch probe functio
Physical output command	16#7003#01:Ch0 physical output com
Compare mode	Unassigned
Compare pluse/time	Unassigned
Compare size/step	Unassigned
Compare point value 1	Unassigned
Compare point value 2	Unassigned
Input (Device to Controller)	
Error code	16#3200#01:Ch0 error code
Counter status	16#6000#01:Ch0 counter status
Encoder present position	16#6002#01:Ch0 encoder present pos
Pluse rate	16#6003#01:Ch0 pulse rate
Time stamp	Unassigned
Reset/External input status	16#6001#01:Ch0 reset/external inpu
Touch probe status	Unassigned
Touch probe pos 1 pos value	Unassigned
Touch probe pos 1 neg value	Unassigned
Touch probe pos 2 pos value	Unassigned
Touch probe pos 2 neg value	Unassigned
Physical output status	16#600e#01:Ch0 physical output sta
Compare error code	Unassigned
Current compare number/position	Unassigned

15.4.2 Counter Enabling

Set the "Counting mode" to "A/B phase frequency" for both Axis_0 and Axis_1, and select "Door control" in "X12 setting" for Axis_1.



Mode Settings ● Linear Mode ○ Rotation Mode

Software Limits □ Enable

Negative limit value: 0.00 Unit Forward Limit: 1000.00 Unit

Count parameter settings:

Count Mode A/B phase 1x

Count logic ● Positive Logic ○ Inverse Logic

Frequency sampling period 10 ms Input filter time 4 Hz

X10 Settings:

Selection Probe Function 1

Level Logic ● Positive Logic ○ Inverse Logic

Y10 Settings:

Selection One dimensional comparison Output

Level Logic ● Positive Logic ○ Inverse Logic

Break Output status ○ Keep status ● Output set value

Set value ● OFF ○ ON

X11 Settings:

Selection Probe Function 2

Level Logic ● Positive Logic ○ Inverse Logic

Y11 Settings:

Selection Normal Output

Level Logic ● Positive Logic ○ Inverse Logic

Break Output status ○ Keep status ● Output set value

Set value ● OFF ○ ON

X12 Settings:

Selection Gated

Level Logic ● Positive Logic ○ Inverse Logic

Y12 Settings:

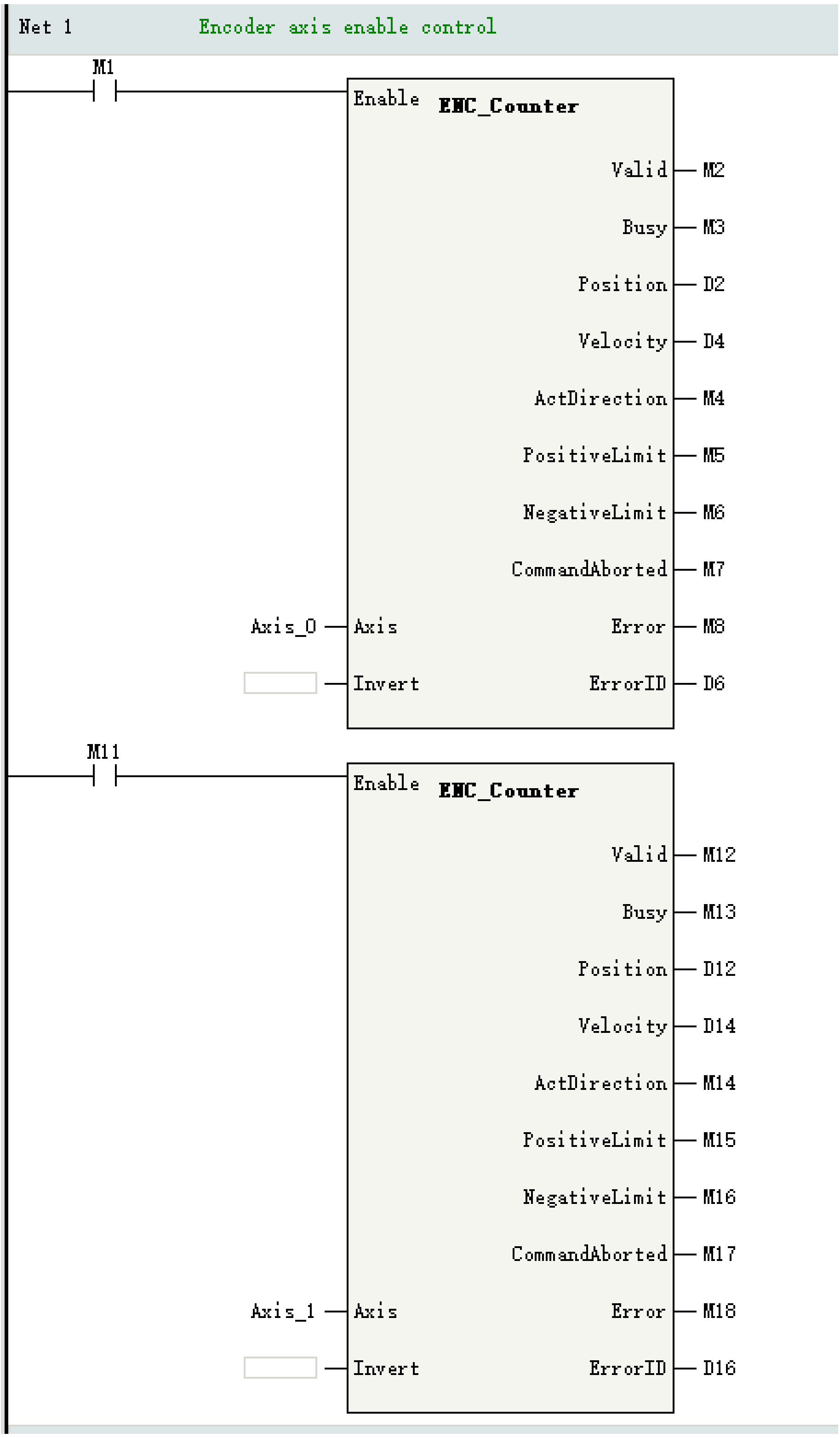
Selection Normal Output

Level Logic ● Positive Logic ○ Inverse Logic

Break Output status ○ Keep status ● Output set value

Set value ● OFF ○ ON

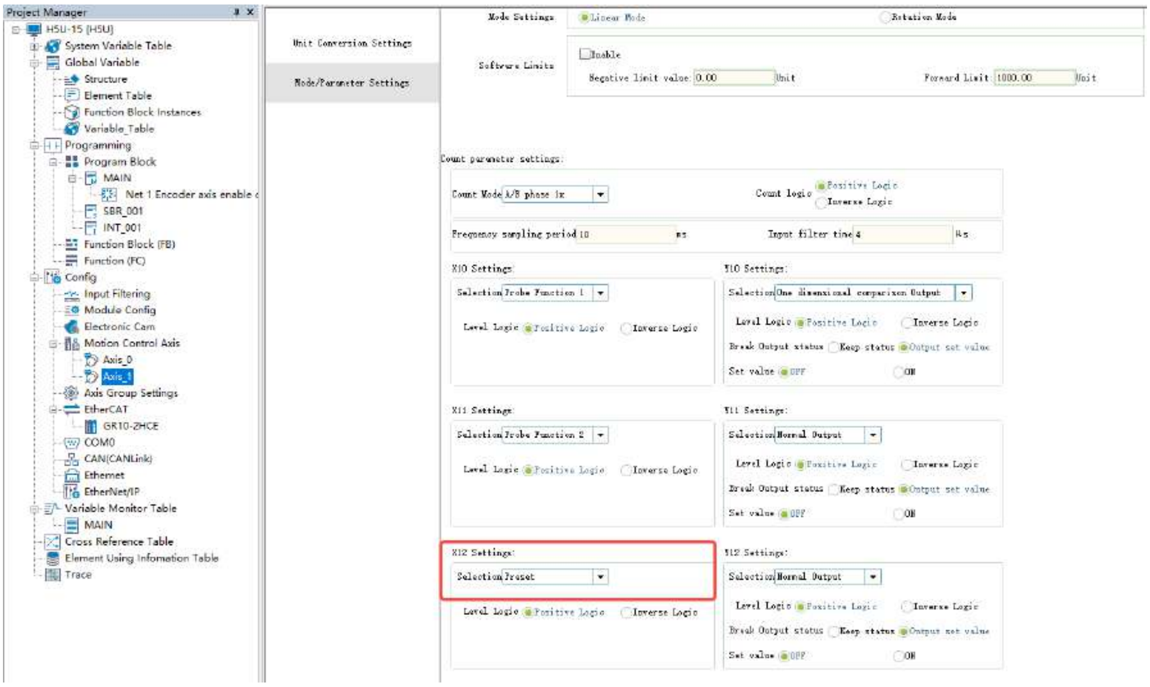
Call the instruction ENC_Counter to enable both bus encoders.



- After M1 is set to ON, M2 and M3 output is ON, D2 displays the current position, and the bus encoder axis Axis_0 starts counting.
- After M11 is set to ON, if the X12 input of the gating terminal is OFF, the M13 output of the instruction is ON, the M12 output is OFF, and D12 shows the current position, then the encoder axis Axis_1 does not count. If the X12 input of the gating terminal is ON and the M12 output of the instruction is ON, then the encoder axis Axis_1 starts counting.

15.4.3 Counter Presetting

Allocate X13 with the preset function and combine it with the instruction ENC_Preset to realize the function of presetting the count value for the encoder Axis_1 of the terminal.



Mode Settings ● Linear Mode ○ Rotation Mode

Software Limits □ Enable

Negative limit value: 0.00 Unit Forward Limit: 1000.00 Unit

Count parameter settings:

Count Mode A/B phase 1x Count logic ● Positive Logic ○ Inverse Logic

Frequency sampling period 10 ms Input filter time 4 μs

X10 Settings:

Selection Probe Function 1

Level Logic ● Positive Logic ○ Inverse Logic

Y10 Settings:

Selection One dimensional comparison Output

Level Logic ● Positive Logic ○ Inverse Logic

Break Output status ○ Keep status ● Output set value

Set value ● OFF ○ ON

X11 Settings:

Selection Probe Function 2

Level Logic ● Positive Logic ○ Inverse Logic

Y11 Settings:

Selection Normal Output

Level Logic ● Positive Logic ○ Inverse Logic

Break Output status ○ Keep status ● Output set value

Set value ● OFF ○ ON

X12 Settings:

Selection Preset

Level Logic ● Positive Logic ○ Inverse Logic

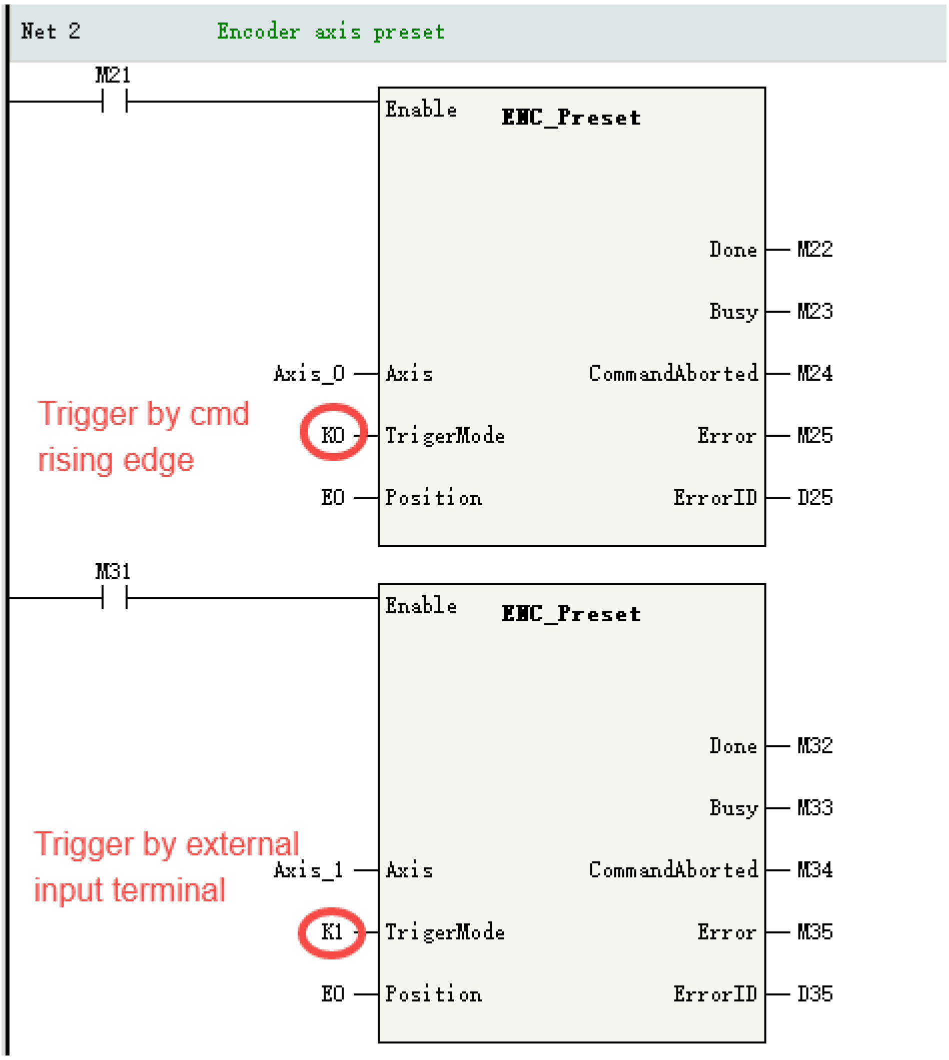
Y12 Settings:

Selection Normal Output

Level Logic ● Positive Logic ○ Inverse Logic

Break Output status ○ Keep status ● Output set value

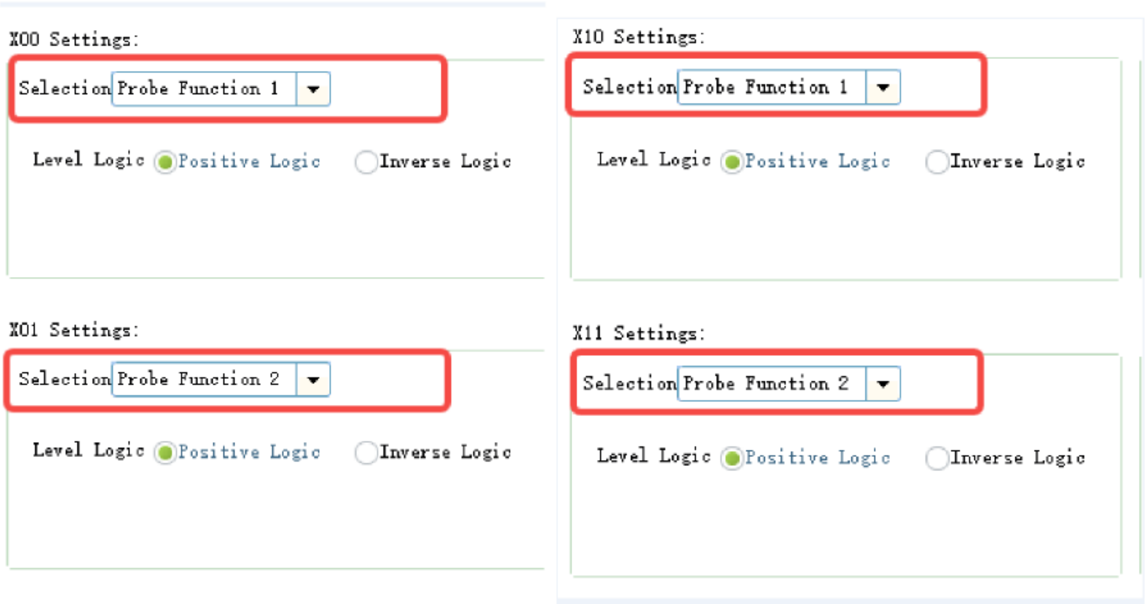
Set value ● OFF ○ ON



- Set M21 to ON. Then the current count value of Axis_0 will be set to 0, and M22 output will be ON after the setting is completed.
- Set M31 to ON and M33 output to ON. When the terminal X13 of the GR10-2HCE module switches its state from OFF to ON, the current count value of Axis_1 will be set to 0, and the M32 output will be ON after the setting is completed.

15.4.4 Probe Function

Allocate the probe function 1 and probe function 2 for X00 and X10 in the encoder axis Axis_0 and for X10 and X11 in the encoder axis Axis_1.



X00 Settings:

Selection Probe Function 1

Level Logic ● Positive Logic ○ Inverse Logic

X10 Settings:

Selection Probe Function 1

Level Logic ● Positive Logic ○ Inverse Logic

X01 Settings:

Selection Probe Function 2

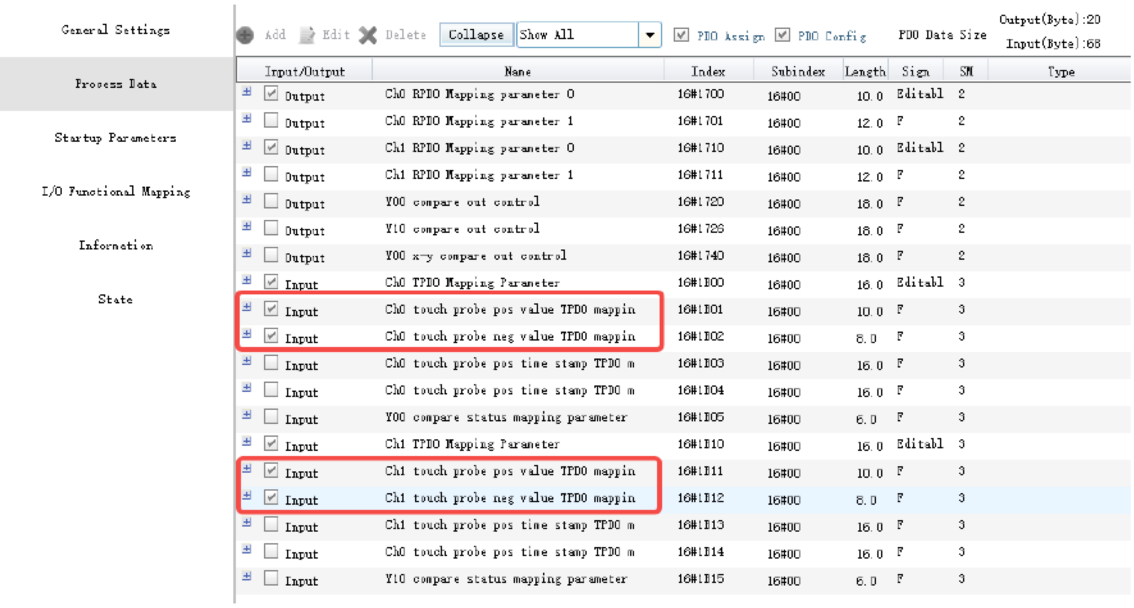
Level Logic ● Positive Logic ○ Inverse Logic

X11 Settings:

Selection Probe Function 2

Level Logic ● Positive Logic ○ Inverse Logic

Tick the following PDO data on the interface "过程数据" of GR10-2HCE.

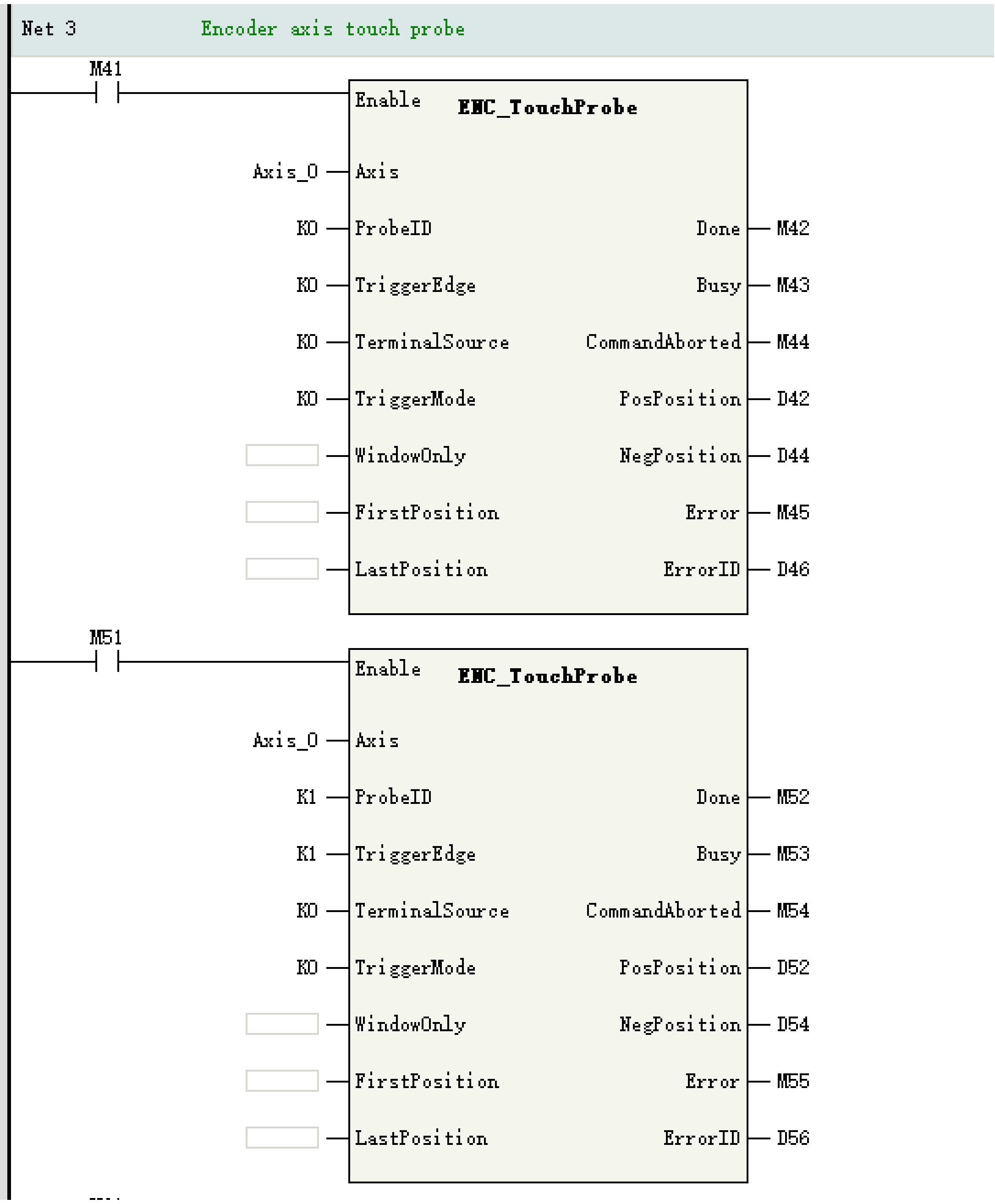


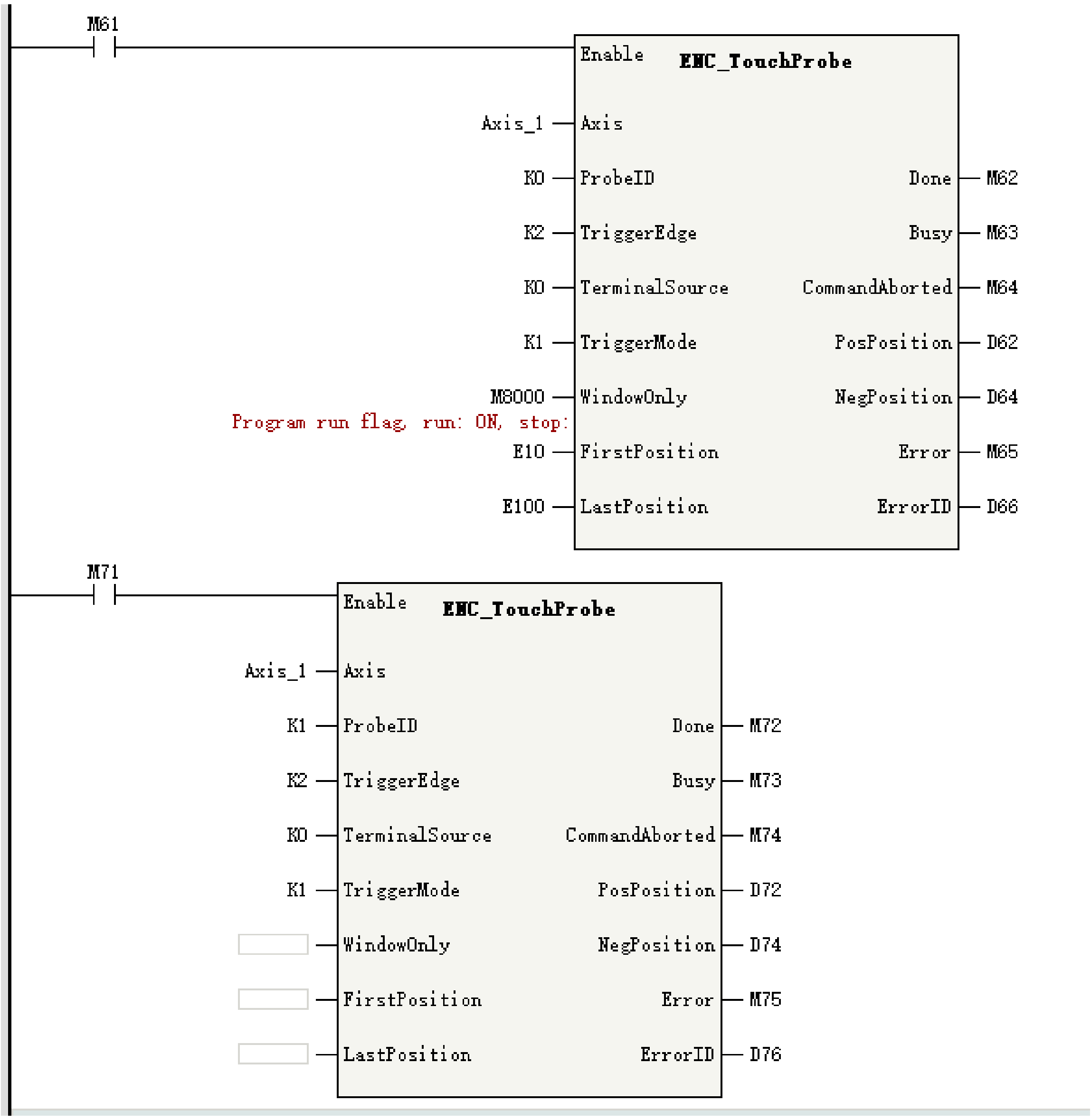
☑ PDO Assign ☑ PDO Config PDO Data Size Output(Byte):20 Input(Byte):68

	Input/Output	Name	Index	Subindex	Length	Sign	SM	Type
☑	Output	Ch0 RPDO Mapping parameter 0	16#1700	16#00	10.0	Editabl	2	
□	Output	Ch0 RPDO Mapping parameter 1	16#1701	16#00	12.0	F	2	
☑	Output	Ch1 RPDO Mapping parameter 0	16#1710	16#00	10.0	Editabl	2	
□	Output	Ch1 RPDO Mapping parameter 1	16#1711	16#00	12.0	F	2	
□	Output	Y00 compare out control	16#1720	16#00	18.0	F	2	
□	Output	Y10 compare out control	16#1726	16#00	18.0	F	2	
□	Output	Y00 x-y compare out control	16#1740	16#00	18.0	F	2	
☑	Input	Ch0 TPDO Mapping Parameter	16#1B00	16#00	16.0	Editabl	3	
☑	Input	Ch0 touch probe pos value TPDO mappin	16#1B01	16#00	10.0	F	3	
☑	Input	Ch0 touch probe neg value TPDO mappin	16#1B02	16#00	8.0	F	3	
□	Input	Ch0 touch probe pos time stamp TPDO m	16#1B03	16#00	16.0	F	3	
□	Input	Ch0 touch probe pos time stamp TPDO m	16#1B04	16#00	16.0	F	3	
□	Input	Y00 compare status mapping parameter	16#1B05	16#00	6.0	F	3	
☑	Input	Ch1 TPDO Mapping Parameter	16#1B10	16#00	16.0	Editabl	3	
☑	Input	Ch1 touch probe pos value TPDO mappin	16#1B11	16#00	10.0	F	3	
☑	Input	Ch1 touch probe neg value TPDO mappin	16#1B12	16#00	8.0	F	3	
□	Input	Ch1 touch probe pos time stamp TPDO m	16#1B13	16#00	16.0	F	3	
□	Input	Ch0 touch probe pos time stamp TPDO m	16#1B14	16#00	16.0	F	3	
□	Input	Y10 compare status mapping parameter	16#1B15	16#00	6.0	F	3	

Call the instruction ENC_TouchProbe to control the probe function of the encoder axis. The functions of the four probes are set as follows.

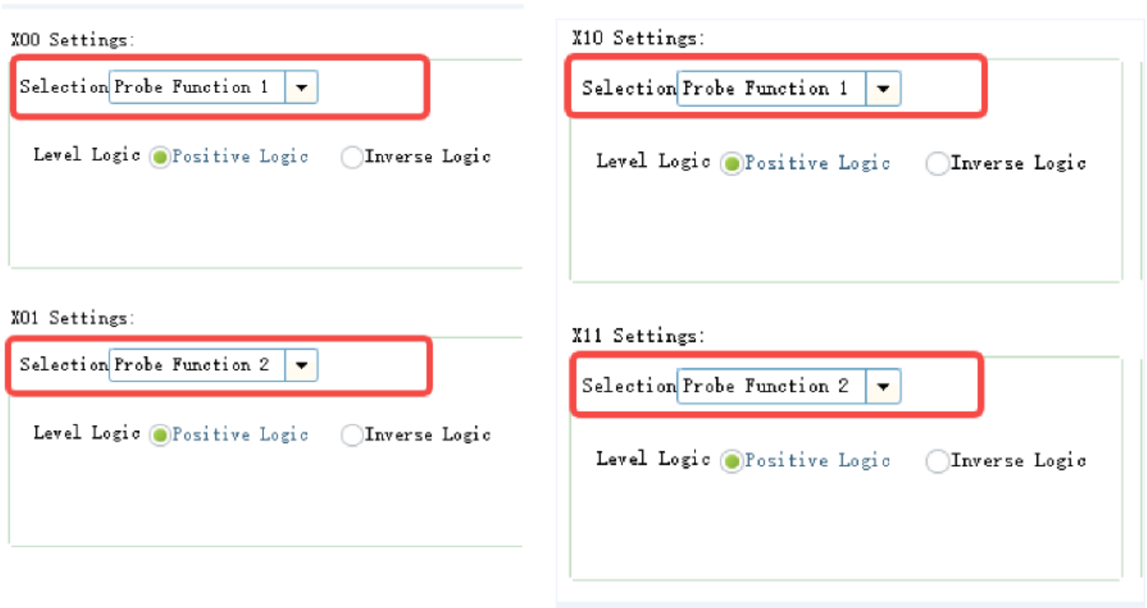
Para.	Axis_0 probe 1	Axis_0 probe 2	Axis_1 probe 1	Axis_1 probe 2
Probe ID	Probe 1	Probe 2	Probe 1	Probe 2
Trigger edge	Trigger by the rising edge only	Trigger by the falling edge only	Trigger by both rising and falling edges	Trigger by both rising and falling edges
Terminal	DI terminal	DI terminal	DI terminal	DI terminal
Trigger mode	Single trigger	Single trigger	Continuous trigger	Continuous trigger
Window limit	Disabled	Disabled	Enabled	Disabled
Start position	-	-	10	-
End position	-	-	100	-


Net 3
Encoder axis touch probe
M41
Enable
EMC_TouchProbe
Axis_0
Axis
K0
ProbeID
Done
M42
K0
TriggerEdge
Busy
M43
K0
TerminalSource
CommandAborted
M44
K0
TriggerMode
PosPosition
D42
WindowOnly
NegPosition
D44
FirstPosition
Error
M45
LastPosition
ErrorID
D46
M51
Enable
EMC_TouchProbe
Axis_0
Axis
K1
ProbeID
Done
M52
K1
TriggerEdge
Busy
M53
K0
TerminalSource
CommandAborted
M54
K0
TriggerMode
PosPosition
D52
WindowOnly
NegPosition
D54
FirstPosition
Error
M55
LastPosition
ErrorID
D56



15.4.5 One-dimensional Comparison Output

Allocate the one-dimensional comparison output function for Y00 in the encoder axis Axis_0 and Y10 in the encoder axis Axis_1.



X00 Settings:

Selection Probe Function 1

Level Logic ● Positive Logic ○ Inverse Logic

X01 Settings:

Selection Probe Function 2

Level Logic ● Positive Logic ○ Inverse Logic

X10 Settings:

Selection Probe Function 1

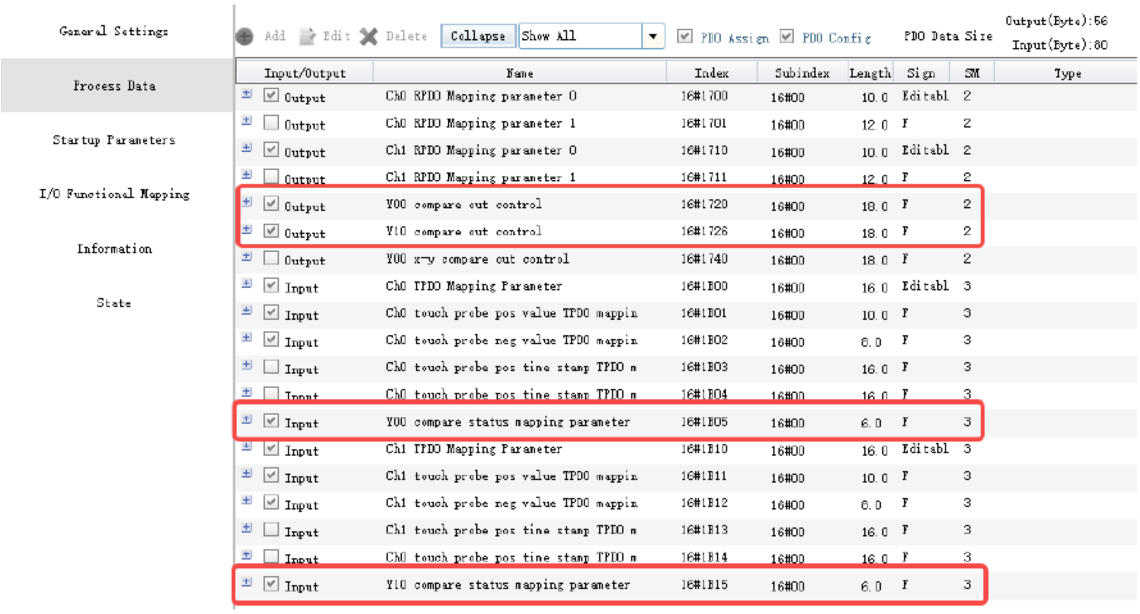
Level Logic ● Positive Logic ○ Inverse Logic

X11 Settings:

Selection Probe Function 2

Level Logic ● Positive Logic ○ Inverse Logic

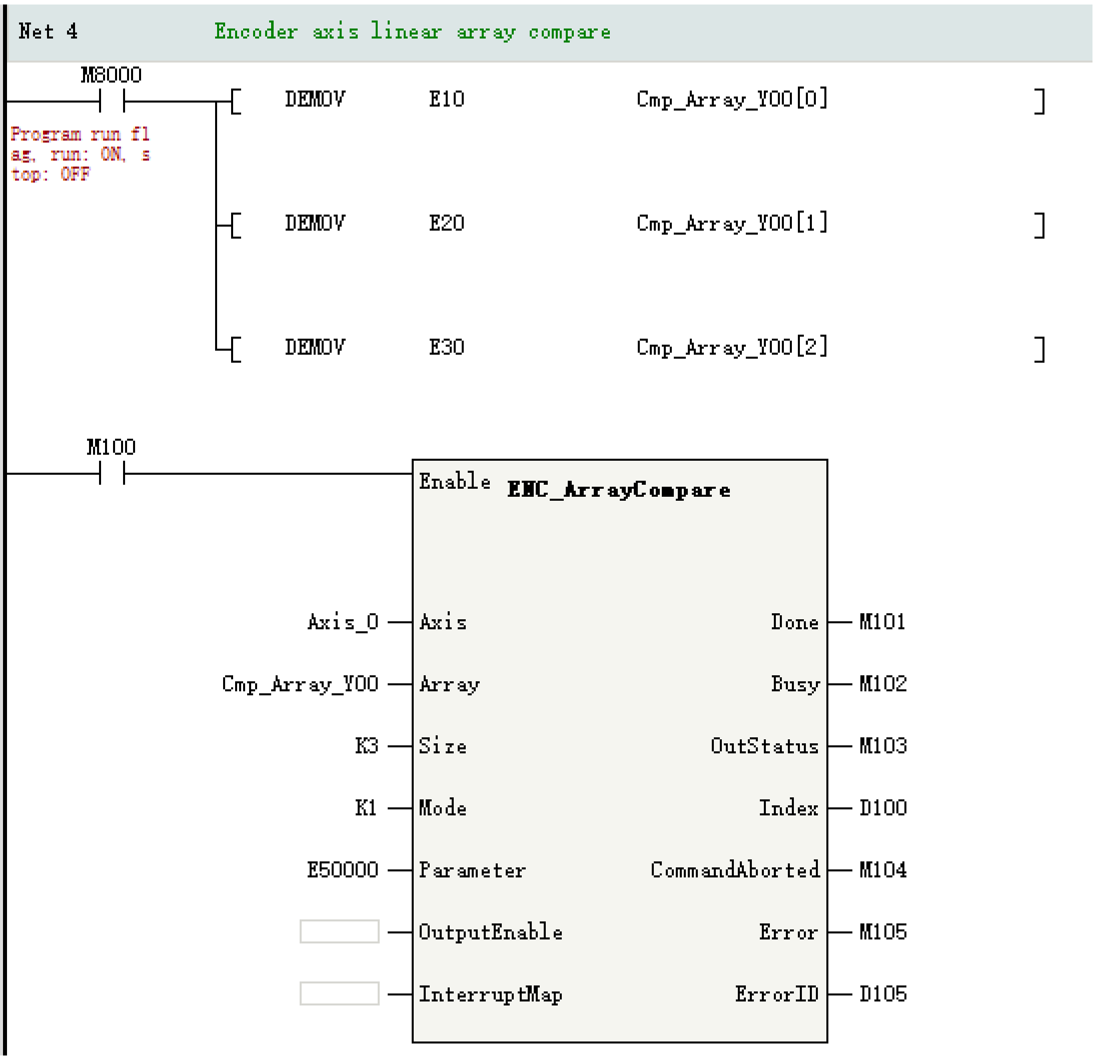
Tick the following PDO data on the interface "Process data" of GR10-2HCE.



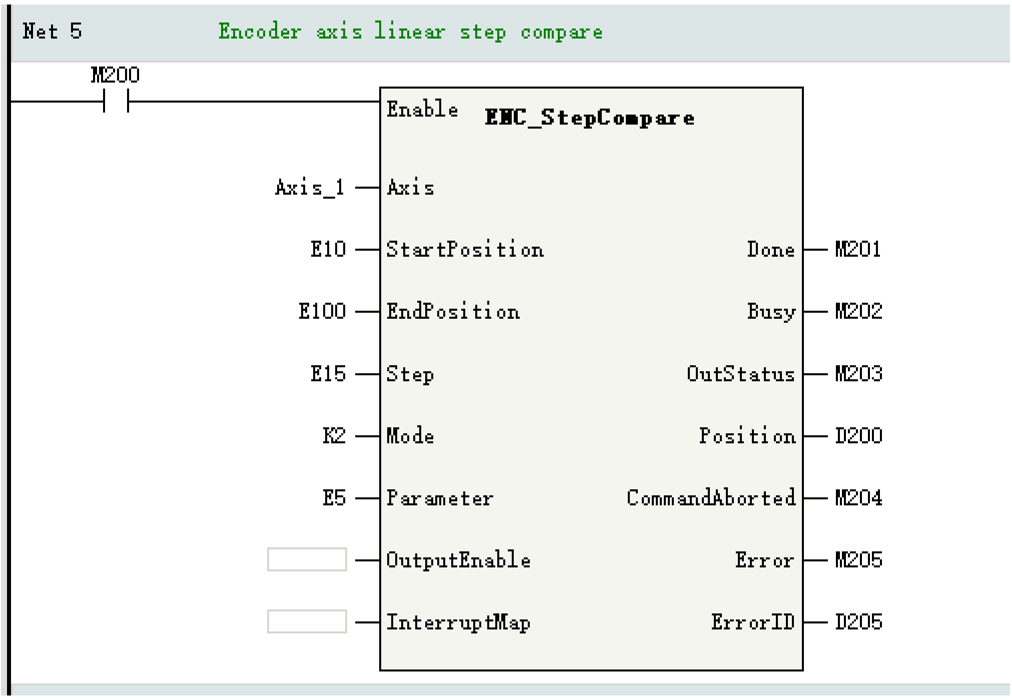
☑ PDO Assign ☑ PDO Config PDO Data Size Output(Byte):56 Input(Byte):80

Input/Output	Name	Index	Subindex	Length	Sign	SM	Type
☑ Output	Ch0 RPDO Mapping parameter 0	16#1700	16#00	10.0	Editabl	2	
□ Output	Ch0 RPDO Mapping parameter 1	16#1701	16#00	12.0	F	2	
☑ Output	Ch1 RPDO Mapping parameter 0	16#1710	16#00	10.0	Editabl	2	
□ Output	Ch1 RPDO Mapping parameter 1	16#1711	16#00	12.0	F	2	
☑ Output	Y00 compare out control	16#1720	16#00	18.0	F	2	
☑ Output	Y10 compare out control	16#1726	16#00	18.0	F	2	
□ Output	Y00 x-y compare out control	16#1740	16#00	18.0	F	2	
☑ Input	Ch0 TPDO Mapping Parameter	16#1B00	16#00	16.0	Editabl	3	
☑ Input	Ch0 touch probe pos value TPDO mappin	16#1B01	16#00	10.0	F	3	
☑ Input	Ch0 touch probe neg value TPDO mappin	16#1B02	16#00	6.0	F	3	
□ Input	Ch0 touch probe pos time stamp TPDO m	16#1B03	16#00	16.0	F	3	
□ Input	Ch0 touch probe pos time stamp TPDO m	16#1B04	16#00	16.0	F	3	
☑ Input	Y00 compare status mapping parameter	16#1B05	16#00	6.0	F	3	
☑ Input	Ch1 TPDO Mapping Parameter	16#1B10	16#00	16.0	Editabl	3	
☑ Input	Ch1 touch probe pos value TPDO mappin	16#1B11	16#00	10.0	F	3	
☑ Input	Ch1 touch probe neg value TPDO mappin	16#1B12	16#00	6.0	F	3	
□ Input	Ch1 touch probe pos time stamp TPDO m	16#1B13	16#00	16.0	F	3	
□ Input	Ch0 touch probe pos time stamp TPDO m	16#1B14	16#00	16.0	F	3	
☑ Input	Y10 compare status mapping parameter	16#1B15	16#00	6.0	F	3	

Call the instruction ENC_ArrayCompare to implement the array comparison of terminal Y00, and output 50 ms at 10, 20, and 30 points, respectively.



Call the instruction ENC_StepCompare to implement the step comparison of terminal Y10, with the start point at 10, end point at 100, and step size as 15. The comparison output is performed in level control mode with a level length of 5 Unit.



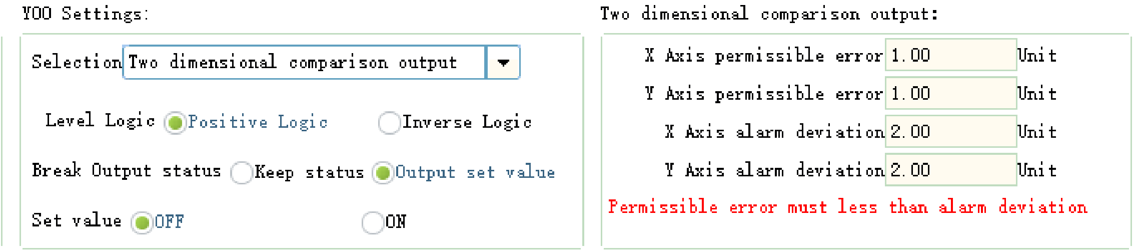
Net 5 Encoder axis linear step compare

M200 — Enable EMC_StepCompare

Input	Pin	Pin	Output
Axis_1	Axis		
E10	StartPosition	Done	M201
E100	EndPosition	Busy	M202
E15	Step	OutStatus	M203
K2	Mode	Position	D200
E5	Parameter	CommandAborted	M204
	OutputEnable	Error	M205
	InterruptMap	ErrorID	D205

15.4.6 Two-dimensional Comparison Output

The GR10-2HCE module supports the two-dimensional comparison output function only at terminal Y00, so only the terminal Y00 of the encoder axis Axis_0 bound to channel CH0 can be allocated with the two-dimensional comparison output function.



Y00 Settings:

Selection: Two dimensional comparison output

Level Logic ● Positive Logic ○ Inverse Logic

Break Output status ○ Keep status ● Output set value

Set value ● OFF ○ ON

Two dimensional comparison output:

X Axis permissible error 1.00 Unit

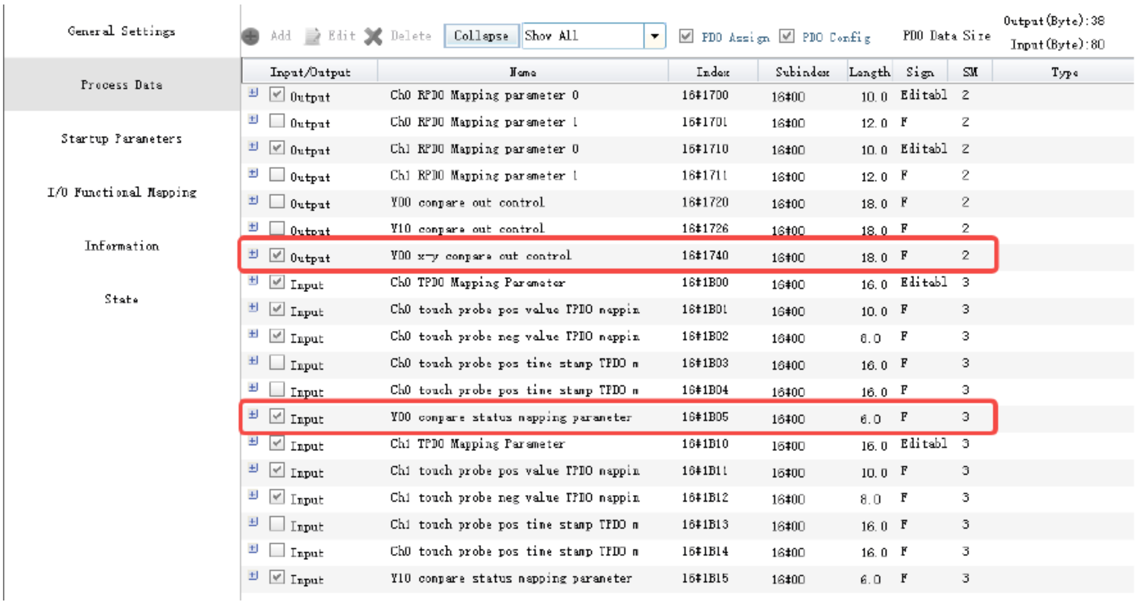
Y Axis permissible error 1.00 Unit

X Axis alarm deviation 2.00 Unit

Y Axis alarm deviation 2.00 Unit

Permissible error must less than alarm deviation

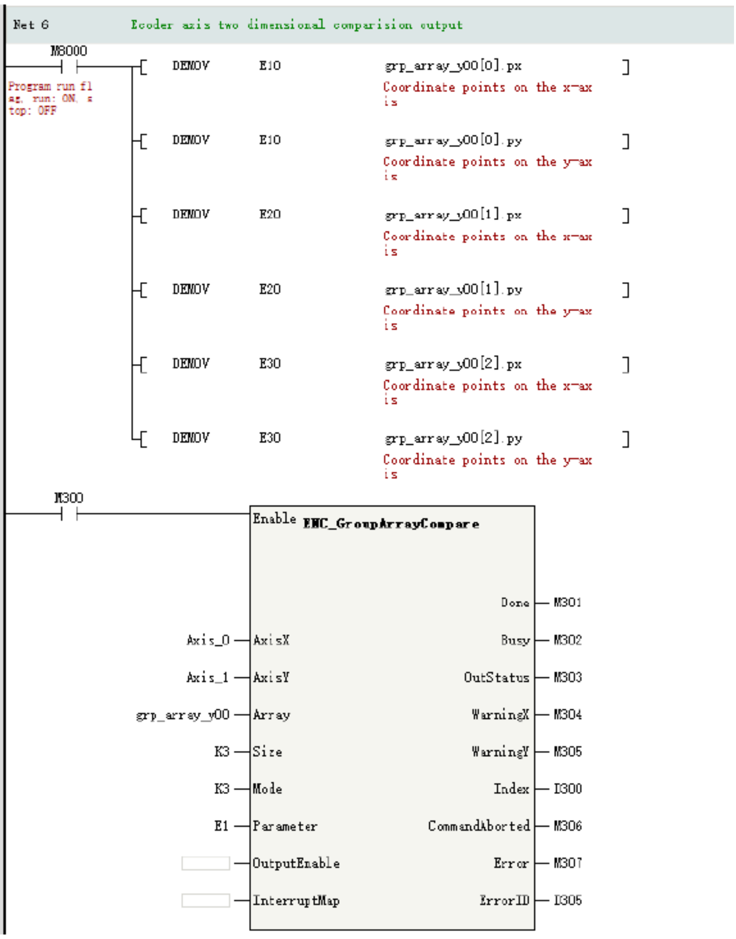
Tick the following PDO data on the interface "Process data" of GR10-2HCE.



PDO Data Size Output(Byte):38 Input(Byte):80

Input/Output	Name	Index	Subindex	Length	Sign	SM	Type
☑ Output	Ch0 RPDO Mapping parameter 0	16#1700	16#00	10.0	Editabl	2	
□ Output	Ch0 RPDO Mapping parameter 1	16#1701	16#00	12.0	F	2	
☑ Output	Ch1 RPDO Mapping parameter 0	16#1710	16#00	10.0	Editabl	2	
□ Output	Ch1 RPDO Mapping parameter 1	16#1711	16#00	12.0	F	2	
□ Output	Y00 compare out control	16#1720	16#00	18.0	F	2	
□ Output	Y10 compare out control	16#1726	16#00	18.0	F	2	
☑ Output	Y00 x-y compare out control	16#1740	16#00	18.0	F	2	
☑ Input	Ch0 TPDO Mapping Parameter	16#1B00	16#00	16.0	Editabl	3	
☑ Input	Ch0 touch probe pos value TPDO mappin	16#1B01	16#00	10.0	F	3	
☑ Input	Ch0 touch probe neg value TPDO mappin	16#1B02	16#00	8.0	F	3	
□ Input	Ch0 touch probe pos time stamp TPDO m	16#1B03	16#00	16.0	F	3	
□ Input	Ch0 touch probe pos time stamp TPDO m	16#1B04	16#00	16.0	F	3	
☑ Input	Y00 compare status mapping parameter	16#1B05	16#00	6.0	F	3	
☑ Input	Ch1 TPDO Mapping Parameter	16#1B10	16#00	16.0	Editabl	3	
☑ Input	Ch1 touch probe pos value TPDO mappin	16#1B11	16#00	10.0	F	3	
☑ Input	Ch1 touch probe neg value TPDO mappin	16#1B12	16#00	8.0	F	3	
□ Input	Ch1 touch probe pos time stamp TPDO m	16#1B13	16#00	16.0	F	3	
□ Input	Ch0 touch probe pos time stamp TPDO m	16#1B14	16#00	16.0	F	3	
☑ Input	Y10 compare status mapping parameter	16#1B15	16#00	6.0	F	3	

Call the instruction ENC_GroupArrayCompare to realize the comparison output function, and set three comparison points (10,10), (20,20), and (30,30). The comparison output is performed in level control mode with a high initial level.

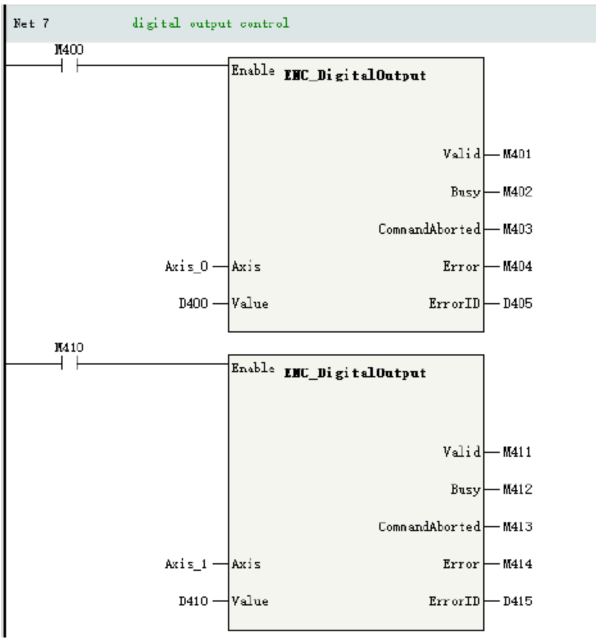


```
Net 6          Ecoder axis two dimensional comparision output

 M8000
--| |---+--[ DEMOV   E10   grp_array_y00[0].px   ]
Program |                  Coordinate points on the x-ax
run fl  |                  is
ag, run:+--[ DEMOV   E10   grp_array_y00[0].py   ]
ON, s   |                  Coordinate points on the y-ax
top: OFF|                  is
        +--[ DEMOV   E20   grp_array_y00[1].px   ]
        |                  Coordinate points on the x-ax
        |                  is
        +--[ DEMOV   E20   grp_array_y00[1].py   ]
        |                  Coordinate points on the y-ax
        |                  is
        +--[ DEMOV   E30   grp_array_y00[2].px   ]
        |                  Coordinate points on the x-ax
        |                  is
        +--[ DEMOV   E30   grp_array_y00[2].py   ]
                           Coordinate points on the y-ax
                           is

 M300
--| |---------------- Enable ENC_GroupArrayCompare
                                                  Done -- M301
               Axis_0 -- AxisX                    Busy -- M302
               Axis_1 -- AxisY               OutStatus -- M303
        grp_array_y00 -- Array                WarningX -- M304
                   K3 -- Size                 WarningY -- M305
                   K3 -- Mode                    Index -- D300
                   E1 -- Parameter    CommandAborted -- M306
                      -- OutputEnable            Error -- M307
                      -- InterruptMap          ErrorID -- D305
```

15.4.7 DO Terminal Control

If the DO terminal of the GR10-2HCE module is used as a common DO terminal, call the instruction ENC_DigitalOutput in the program to control the terminal. The address mapping table of the DO terminals is as follows.

Address	Controlled Terminal
D400.0	Y00
D400.1	Y01
D400.2	Y02
D410.0	Y10
D410.1	Y11
D410.2	Y12

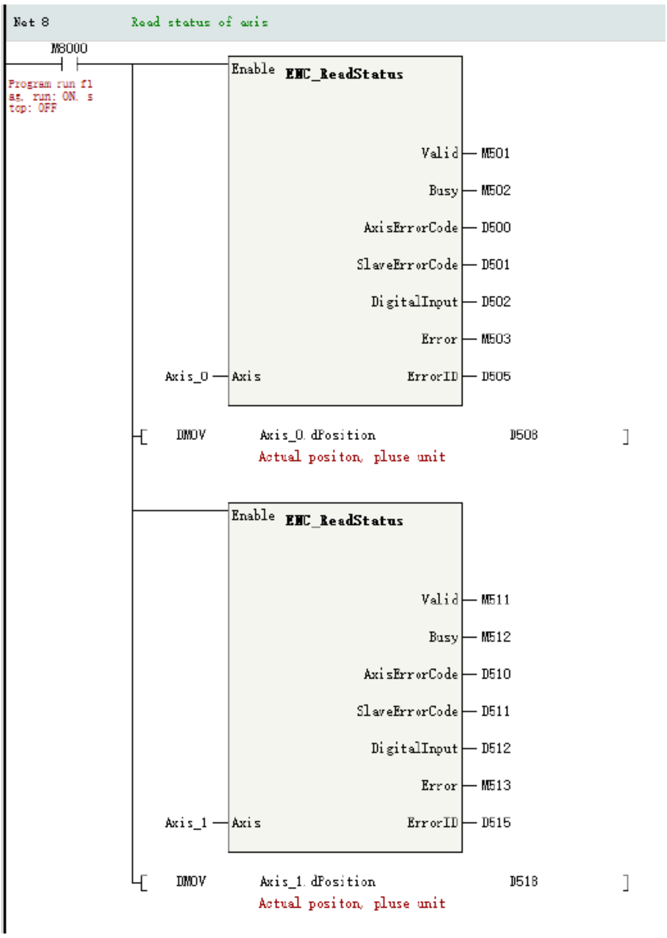


15.4.8 Obtaining Axis Status

Call the instruction ENC_ReadStatus to obtain the fault code of an axis and the status of the DI terminal. The address mapping table of the DI terminals is as follows.

Address	Mapped Terminal
D502.0	X00
D502.1	X01
D502.2	X02
D502.3	X03
D512.0	X10
D512.1	X11
D512.2	X12
D512.3	X13

Through the system variables, write the feedback position of the pulse unit of Axis_0 to D508 and the feedback frequency of the pulse unit of Axis_1 to D518.


Net 8
Read status of axis
M8000
Program run flag, run: ON, stop: OFF
Enable
EMC_ReadStatus
Valid — M501
Busy — M502
AxisErrorCode — D500
SlaveErrorCode — D501
DigitalInput — D502
Error — M503
Axis_0 — Axis
ErrorID — D505
DMOV Axis_0.dPosition D508
Actual positon, pluse unit
Enable
EMC_ReadStatus
Valid — M511
Busy — M512
AxisErrorCode — D510
SlaveErrorCode — D511
DigitalInput — D512
Error — M513
Axis_1 — Axis
ErrorID — D515
DMOV Axis_1.dPosition D518
Actual positon, pluse unit

16 Electronic Cam

16.1 Introduction to Electronic Cam

Electronic cam essentially involves the motion of the slave axis following the master axis. The motion relationship between the master axis and the slave axis can be expressed in a cam table data or electronic gear ratio approach.

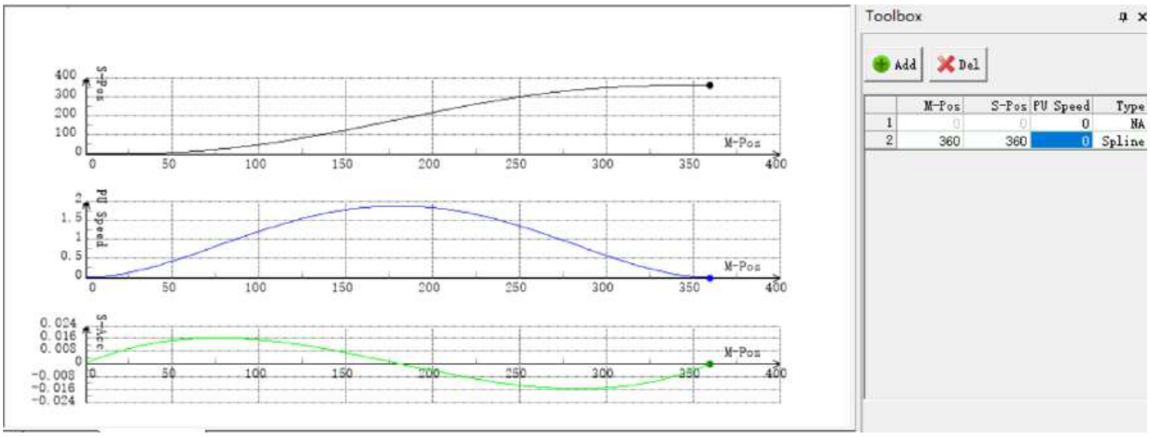
- In the cam table data approach, up to 361 key points can be created. In the electronic gear ratio approach, only one constant ratio is applied between the master axis and the slave axis.
- If electronic gear is used, just set the numerator and denominator of the electronic gear ratio and there is no need to set cam table data. If electronic cam is used, set electronic cam table data first.
- The programming software can be configured with 16 cam tables, each with up to 361 key points. Up to 8 electronic cams can be used simultaneously in the program.
- During cam execution, it is allowed to add, delete, and modify key points of a cam table, and the modified cam table takes effect in the next cam cycle.

16.2 Software Configuration

16.2.1 Overview

In "Project Manager", expand "Configure", and double-click "Electronic Cam" to open the relevant configuration page.

The cam table page contains a graphic editing area on the left and a parameter point editing area on the right.



	M-Pos	S-Pos	PU Speed	Type
1	0	0	0	NA
2	360	360	0	Spline

16.2.2 Cam Node Settings

You can set cam nodes in the parameter point editing area.

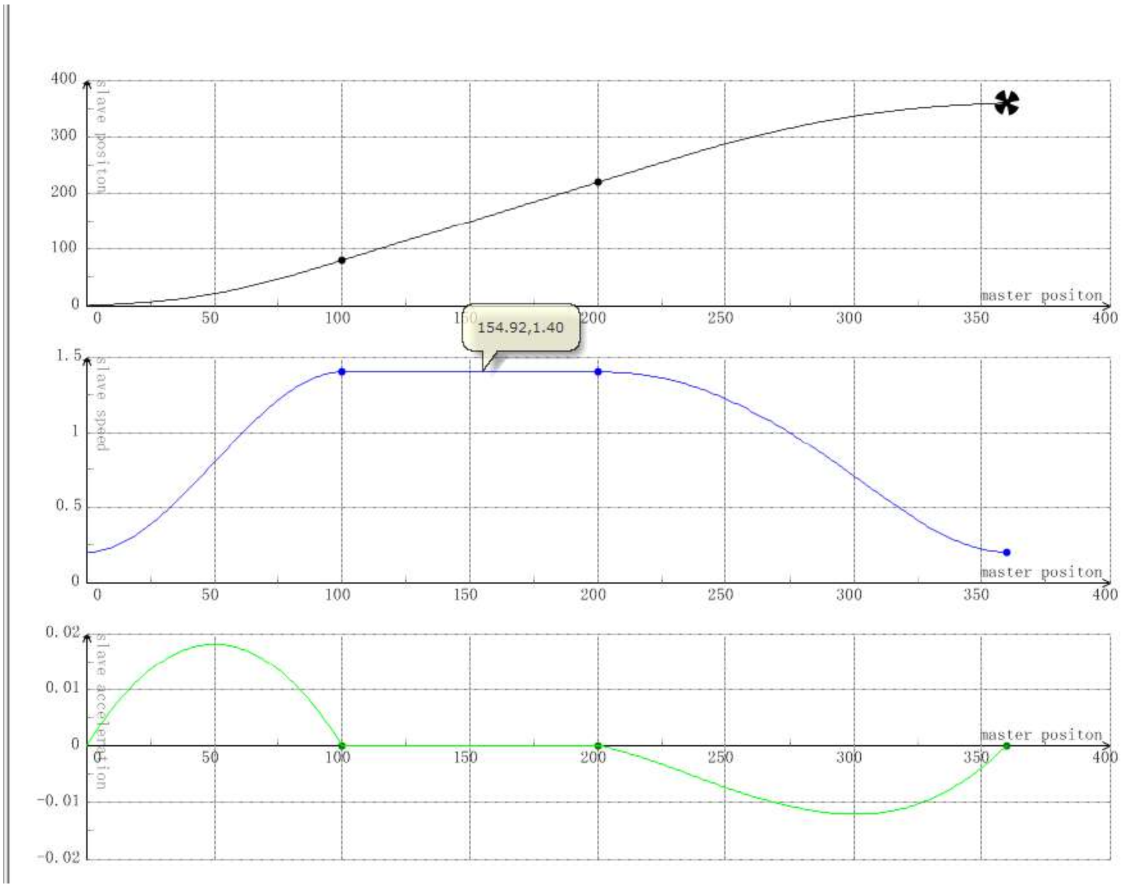
Click "Add" to add a cam node data row and edit the relevant data. To delete a node, select the corresponding node data row and click "Del".

Table 16–1 Definitions of cam node parameters

Para.	Function
M-Pos	Master axis phase Sets the phase of the master axis (relative mode)
S-Pos	Slave axis displacement Sets the offset of the slave axis (relative mode)
PU-Speed	Connection speed Automatically generated when the curve type is set to straight line, or manually set when the curve type is set to quintic curve
Type	Sets the curve type Line: Straight line Spline: Quintic curve

16.2.3 Cam Curve Settings

In the graphic editing area, you can set cam curves, including position, speed ratio, and acceleration ratio curves.



Cam curve description

1. Cam key points on the position curve can be moved up or down and left or right. The speed ratio curve can only be moved up or down. The acceleration ratio curve does not allow modification.
2. The last point can only be dragged up and down. To change the value of the last point leftwards or rightwards, manually modify the data in the toolbar on the right.

3. Hover the mouse cursor over a point in the coordinate system, and the specific coordinate information will be displayed.
4. Right-click to insert or delete a key point.
5. Click a line segment between two key points in any coordinate system, and the line segments between the two key points in all the three coordinate systems will become bolded.

16.2.4 Import and Export

You can export or import each individual cam table.
Select the electronic cam you want to export or import and right-click it to export/import the electronic cam to/from a CSV file.

16.2.5 Uploading Cam Tables

All the cam tables saved in a board can be uploaded by using the upload feature.

16.2.6 Calling System Variables and Instructions

When a cam table is created, the software backend assigns a system variable to represent the cam table. The status of the cam table can be monitored in the PLC program and can be used as a parameter for instructions such as MC_CamIn. In addition, values of key points in the cam table can be modified and updated using the MC_GenerateCamTable instruction.

16.3 System Variables

16.3.1 Cam Nodes

Each key point can be represented by a cam node variable, with the data type being _sMC_CAM_NODE. Member variables of this structure are shown in the following table.

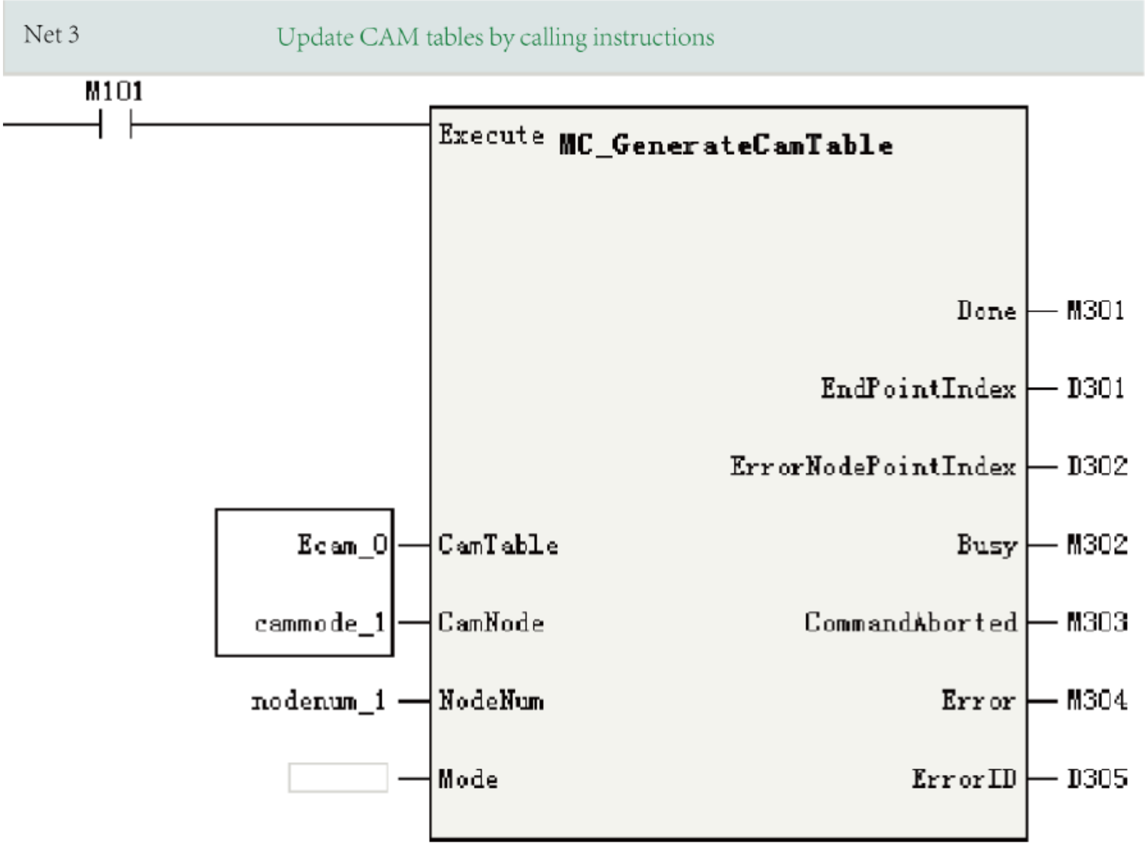
Table 16–2 _sMC_CAM_NODE structure

Variable Name	Data Type	Function Description
fPhase	REAL	Master axis phase
fDistance	REAL	Slave axis displacement
fVel	REAL	Connection speed
fAcc	REAL	Connection acceleration rate (reserved)
iCuve	INT	Curve type 0: Reserved 1: Straight line 2: Quintic curve

Each cam node structure is used as a member variable of a cam table structure to store key point data of the cam table. See *“16.3.2 Cam Tables” on page 517*for details.

In the program, you can also customize cam node arrays for updating cam tables.

The MC_GenerateCamTable instruction can be used to overwrite the existing cam node array in Ecam_ 1 with a newly-defined camnode_1 cam node array.



16.3.2 Cam Tables

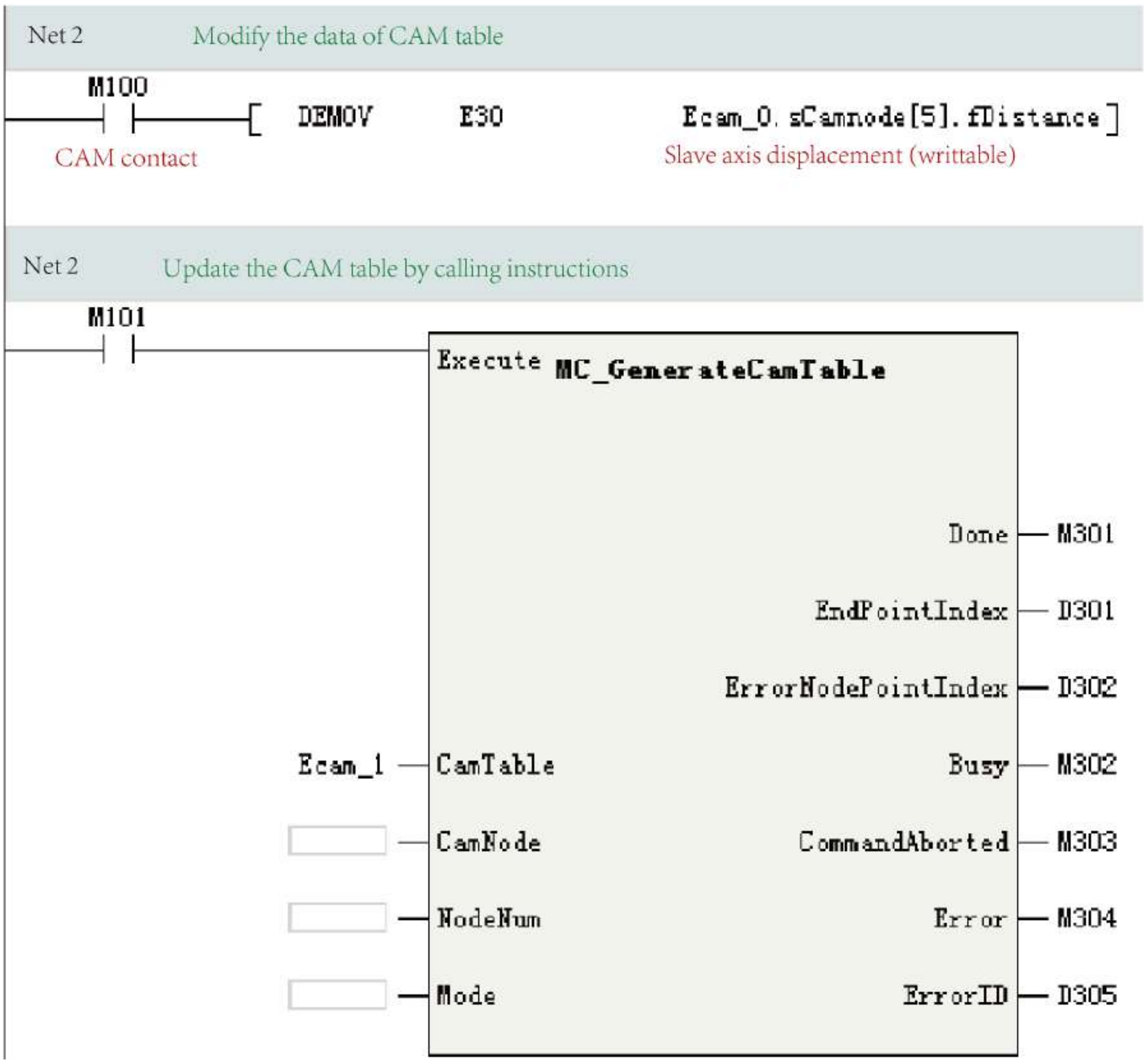
Cam tables can only be created and configured in the software tool and cannot be created in the program. However, coordinates of cam key points can be modified in the program to facilitate modification of process parameters.

Table 16–3 _sMC_CAMTABLE

Variable Name	Data Type	Read/Write Property	Function Description
iCamID	INT	RO	ID number
bSaveState	BOOL	RO	Saving a cam table ON: Saving OFF: Idle
bCheckState	BOOL	RO	Checking a cam table ON: Checking OFF: Idle
bNew	BOOL	RO	Updating a cam table ON: Updating OFF: Idle

Variable Name	Data Type	Read/Write Property	Function Description
iErrorCode	INT	RO	Error code corresponding to the cam check/save failure
iSetNodeNum	INT	RW	Total number of set key points
iActNodeNum	INT	RW	Total number of actual key points Updated after the first run and after each execution of the MC_GenerateCamTable instruction
fMaxPhase	REAL	RO	Cam cycle
sCamnode	_sMC_CAM_NODE [361]	RW	Cam key point array Needs the MC_GenerateCamTable instruction for updating after modification in the program The master axis phases must be arranged in ascending order, otherwise an error will occur.

Cam table variables are automatically generated for cam tables created in the software tool.



16.3.3 Cam Contact Nodes

The system variable for the cam contact node is _sMC_CAMIN.

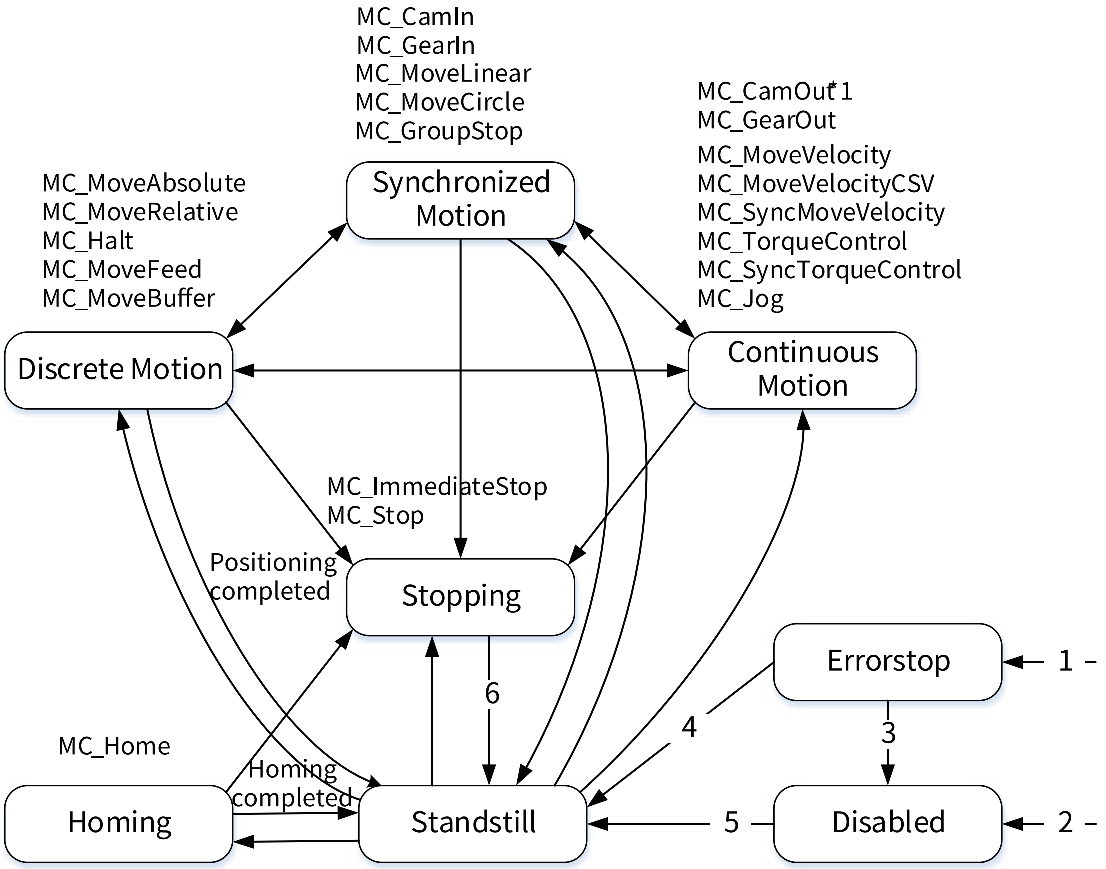
Table 16–4 _sMC_CAMIN data structure

Variable Name	Data Type	Read/Write Property	Function Description
iCamInID	INT	RO	Cam combination ID
iCAMTableID	INT	RO	The cam table being executed
iMasterID	INT	RO	Master axis ID
iSlaveID	INT	RO	Slave axis ID
iState	INT	RO	Reserved and not defined
iCamCnt	INT	RO	Number of cam cycles already executed
iNodeCnt	INT	RO	Key points waiting for execution
fMasterStartpos	REAL	RO	Start position of the master axis
fSlaveStartPos	REAL	RO	Start position of the slave axis
fphase	REAL	RO	Current phase of the master axis
fDistance	REAL	RO	Current displacement of the slave axis
fPhaseShift	REAL	RO	Super-imposed amount of phase shift
fPhaseVelocity	REAL	RO	Super-imposed velocity of phase shift
fPhaseAcc	REAL	RO	Super-imposed acceleration of phase shift

This variable can only be defined as an output variable of the MC_CamIn instruction in the program.

16.4 State Machines

State machines of the electronic cam are shown in the following figure.



Note

Calling the MC_CamOut instruction can start the Continuous Motion state only when OutMode of the instruction is 0.

Table 16–5 State definitions

Status	Function Description
Disabled	Disabled
ErrorStop	Stopped due to a fault
Standstill	Enabled
Homing	Homing
Stopping	Stopped
Discrete Motion	Discrete motion
Continuous Motion	Continuous motion
Synchronized Motion	Synchronized motion

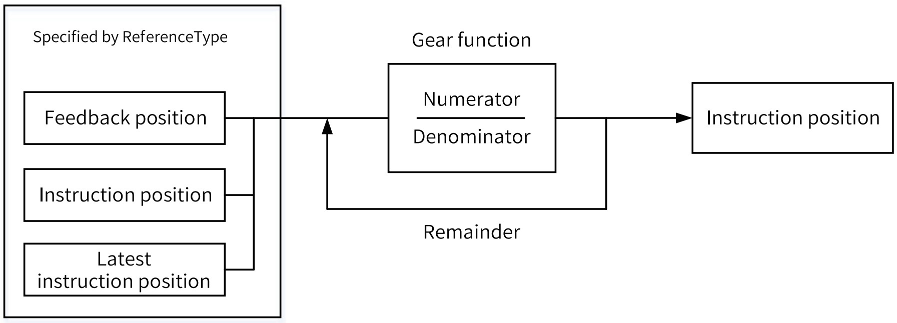
Table 16–6 State transition conditions

Transition	Transition Conditions
1	The fault detection logic of the axis detects a fault. In this case, the system immediately transits to this state.
2	The axis is free of faults and MC_Power.Enable=OFF
3	MC_Reset is called to reset the axis fault and MC_Power.Status=FASLE.
4	MC_Reset is called to reset the axis fault and MC_Power.Status=ON.
5	MC_Power.Enable=ON and MC_Power.Status=ON.
6	MC_Stop(MC_ImmediateStop).Done=ON and MC_Stop(MC_ImmediateStop).Execute=OFF.

16.5 Electronic Cam Operations

16.5.1 Gear Operation

Basic block diagram



Function description

Gear operation is applicable to the following types of master and slave axes.

- Master axis: bus servo axis, local pulse axis, local encoder axis, and bus encoder axis

- Slave axis: bus servo axis and local pulse axis

Use the MC_GearIn (starting gear operation) instruction to start gear operation. Use the MC_GearOut (ending gear operation) instruction or the MC_Stop (forced stop) instruction to end synchronized gear operation.

After gear operation starts, the slave axis accelerates or decelerates, with the target speed being the speed of master axis multiplied by the gear ratio.

Time before the slave axis reaches the target speed is called the Catching phase. Time after the slave axis reaches the target speed is called the InGear phase.

Gear operation is executed by setting the gear ratio between the master and slave axes.

When the gear ratio is positive, the slave axis moves in the same direction as the master axis. When the gear ratio is negative, the slave axis moves in the opposite direction to the master axis.

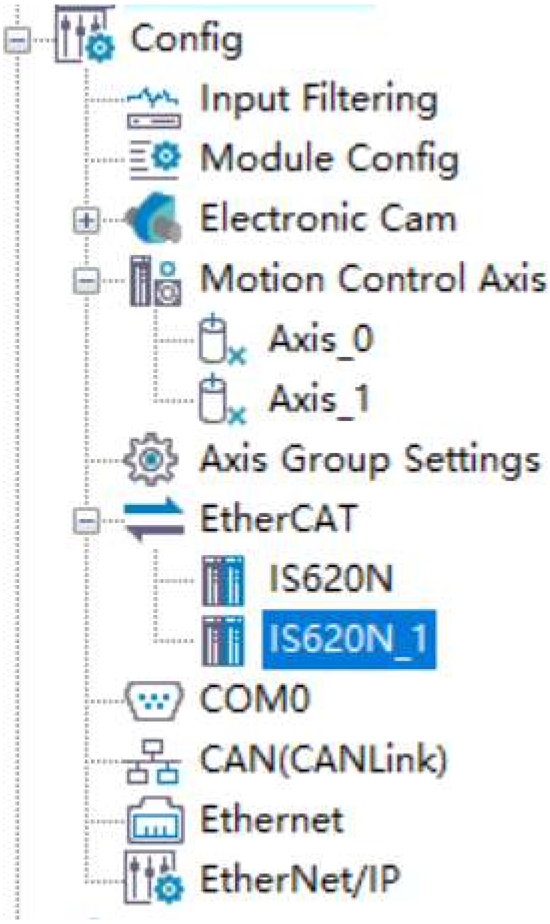
For detailed functions, see the MC_GearIn instruction in the "Electronic Cam Instructions" section of the instructions guide.

Example

Job: Create two bus servo axes and make the second axis follow the first axis at a gear ratio of 1:1 for gear operation.

Procedure:

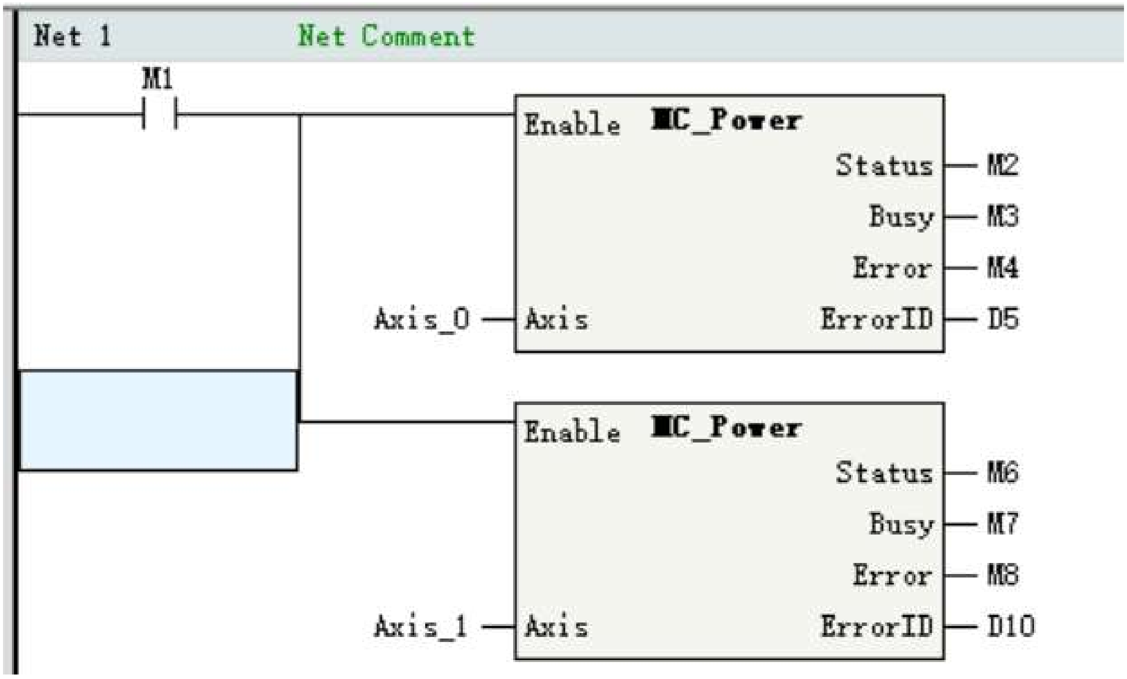
1. Create a project. Create two bus servo axes, one as a master axis and the other as a slave axis.



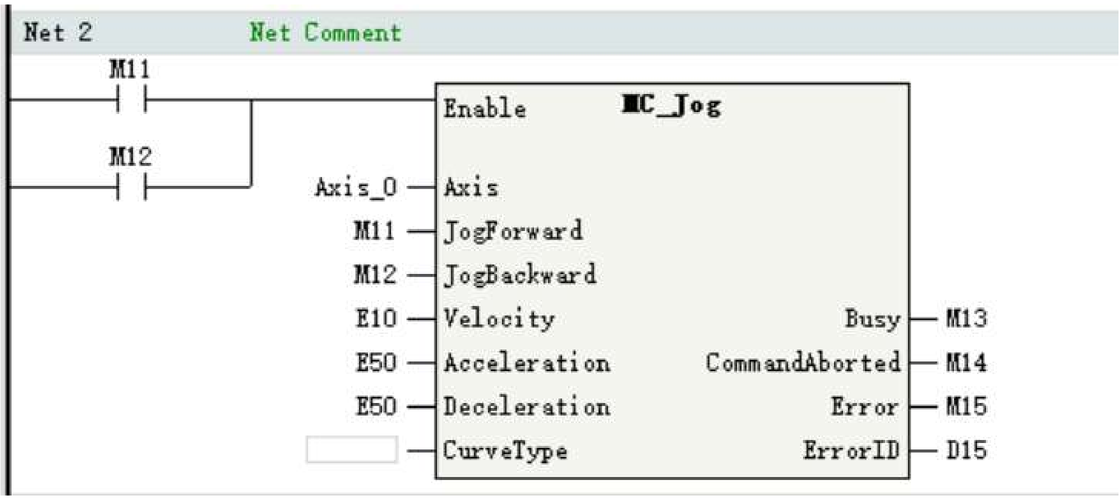
Set Axis_0 of the two bus servo axes as the master axis and Axis_1 as the slave axis.

Bind IS620N of the two servo drives to Axis_0 and IS620_1 to Axis_1.

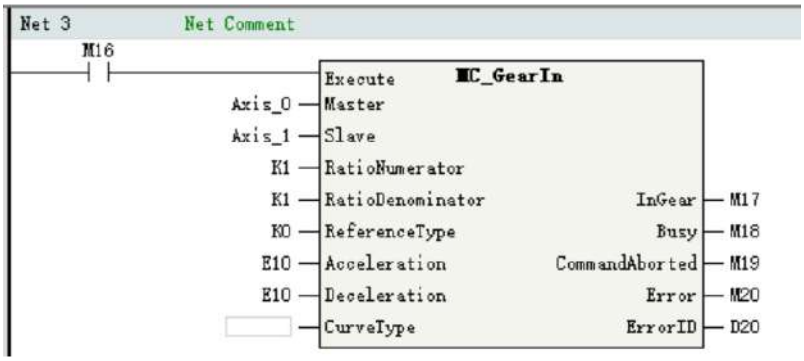
2. Call the MC_Power instruction to control the enabling of the master and slave axes.



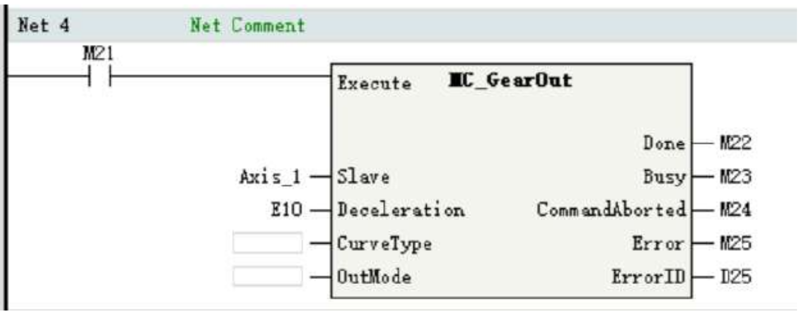
3. Call the MC_Jog instruction to control the forward and reverse motion of the master axis.



4. Call the MC_GearIn instruction to execute gear operation, with the gear ratio set to 1:1.



5. Call the MC_GearOut instruction to end gear operation.

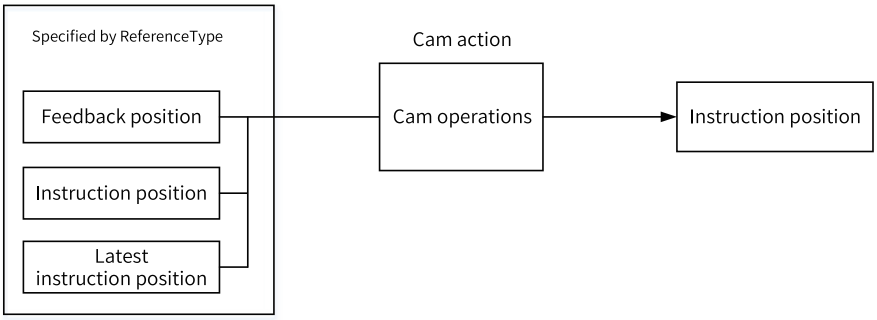


```
Net 4            Net Comment
     M21
 ----| |---------- Execute   MC_GearOut
                                           Done --- M22
            Axis_1 --- Slave               Busy --- M23
               E10 --- Deceleration  CommandAborted --- M24
                   --- CurveType              Error --- M25
                   --- OutMode              ErrorID --- D25
```

16.5.2 Cam Operation

Cam operation refers to the motion of the slave axis in sync with the position of the master axis according to a cam table.

Basic block diagram



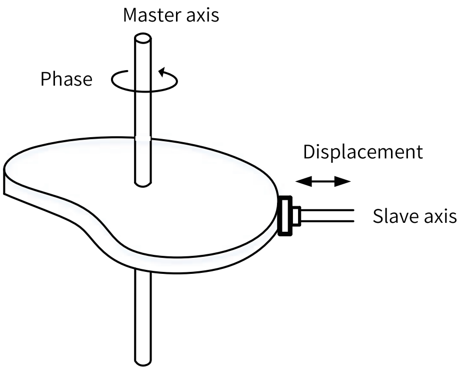
Function description

Cam operation is applicable to the following types of master and slave axes.

- Master axis: bus servo axis, local pulse axis, local encoder axis, and remote encoder axis
- Slave axis: motion control axis

Use the MC_CamIn (starting cam operation) instruction to start cam operation or change cam tables. Use the MC_CamOut (ending cam operation) instruction or the MC_Stop (forced stop) instruction to end cam operation.

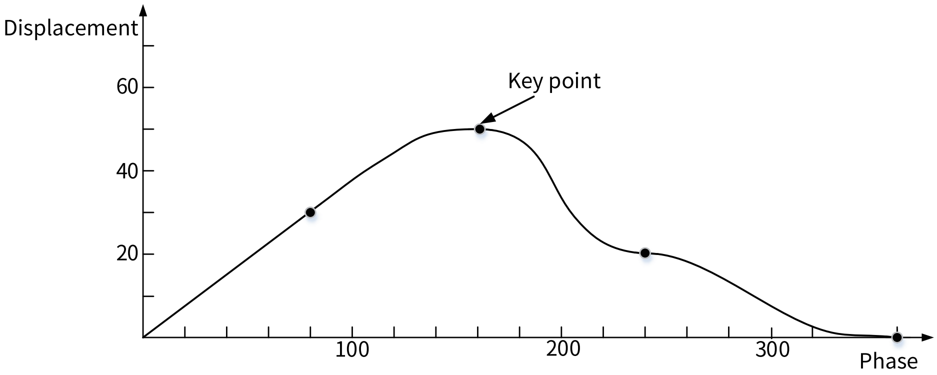
In a typical cam structure shown in the following figure, the master axis rotates periodically and the slave axis moves back and forth along a direction under the control of the master axis.



The electronic cam simulates such a structure, selecting one axis (bus servo axis, local pulse axis, local encoder axis, or remote encoder axis) as the master axis and another axis (bus servo axis or local pulse axis) as the slave axis. The master axis and the slave axis move in a synchronized way according to a set cam curve.

Cam curves

A cam curve is a 2D coordinate system, where the horizontal axis represents the phase of the master axis and the vertical axis represents the displacement of the slave axis. Set some key points in the coordinate system, and connect every two key points with a set curve (such as a straight line or a quintic curve) to form a cam curve.



Cam table

	Phase	Displacement
Start point	0	0
	80	30
	160	50
	240	20
End point	360	0

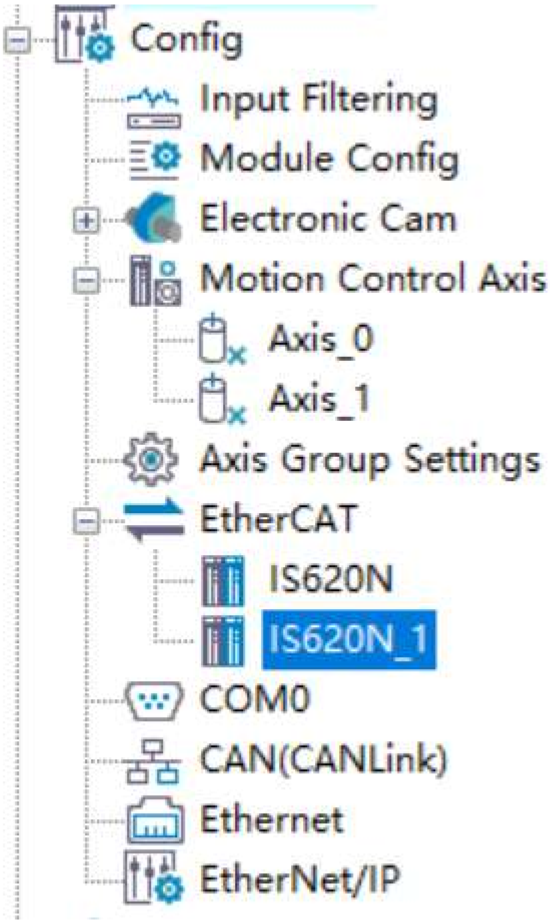
For detailed functions, see the MC_CamIn and MC_CamOut instructions in the "Electronic Cam Instructions" section of the instructions guide.

Example

Job: Create two servo axes. Set Axis_0 as the cam master axis, and Axis_1 as the cam slave axis that follows Axis_0 to execute cam operation.

Procedure:

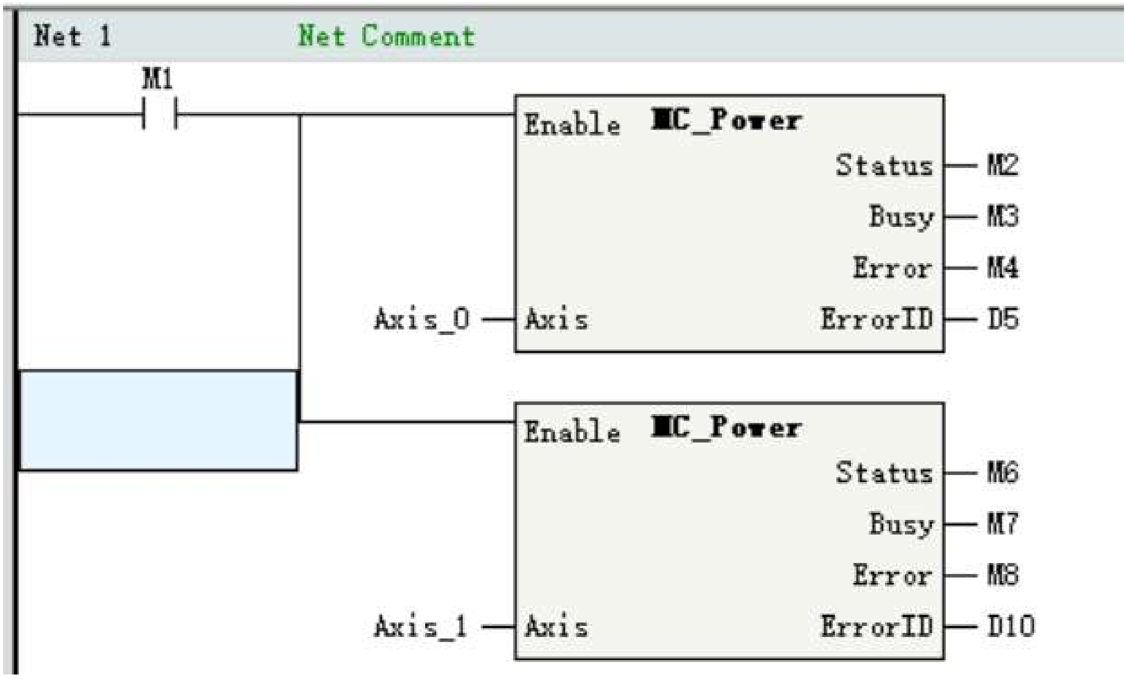
1. Create a project. Create two bus servo axes, one as a master axis and the other as a slave axis.



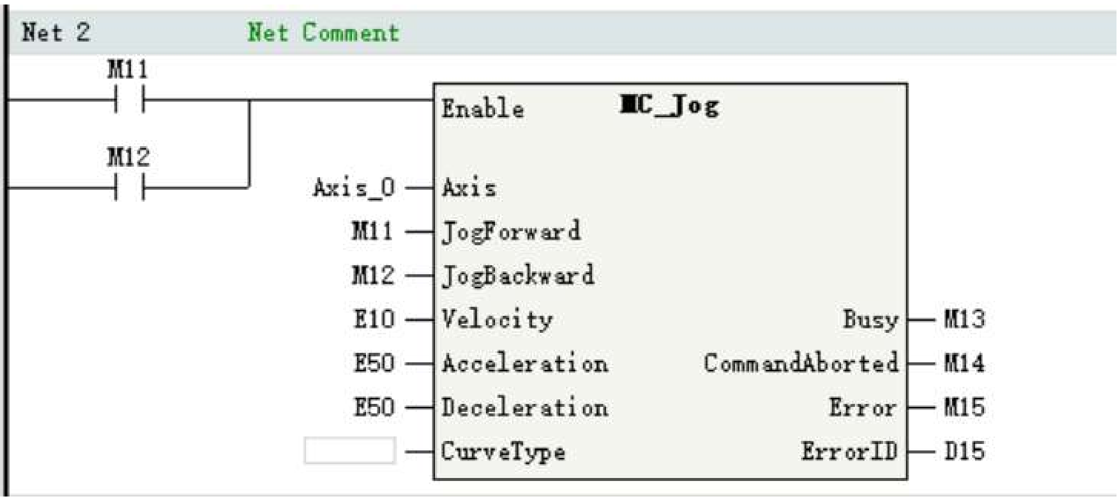
Set Axis_0 of the two bus servo axes as the master axis and Axis_1 as the slave axis.

Bind IS620N of the two servo drives to Axis_0 and IS620_1 to Axis_1.

2. Create a cam table.
3. Call the MC_Power instruction to control the enabling of the master and slave axes.

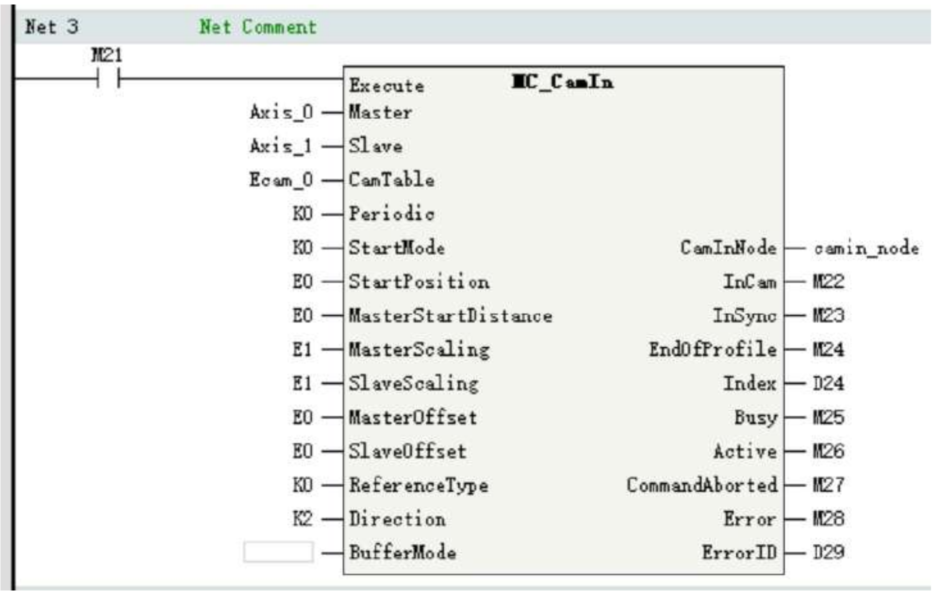


4. Call the MC_Jog instruction to control the forward and reverse motion of the master axis.



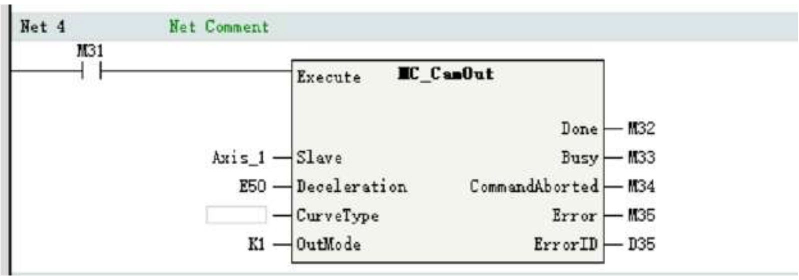
M11, M12 — Enable
Axis_0 — Axis
M11 — JogForward
M12 — JogBackward
E10 — Velocity
E50 — Acceleration
E50 — Deceleration
— CurveType
Busy — M13
CommandAborted — M14
Error — M15
ErrorID — D15

5. Call the MC_CamIn instruction to execute cam operation.



M21 — Execute
Axis_0 — Master
Axis_1 — Slave
Ecam_0 — CamTable
K0 — Periodic
K0 — StartMode
E0 — StartPosition
E0 — MasterStartDistance
E1 — MasterScaling
E1 — SlaveScaling
E0 — MasterOffset
E0 — SlaveOffset
K0 — ReferenceType
K2 — Direction
— BufferMode
CamInNode — camin_node
InCam — M22
InSync — M23
EndOfProfile — M24
Index — D24
Busy — M25
Active — M26
CommandAborted — M27
Error — M28
ErrorID — D29

6. Call the MC_CamOut instruction to end cam operation.



M31 — Execute
Axis_1 — Slave
E50 — Deceleration
— CurveType
K1 — OutMode
Done — M32
Busy — M33
CommandAborted — M34
Error — M35
ErrorID — D35

16.5.3 Cam Tables

16.5.3.1 Introduction to Cam Tables

In the cam function module, a pair of data comprised of the master axis phase and the slave axis displacement is defined as cam data, and combination of cam data is defined as a cam table.

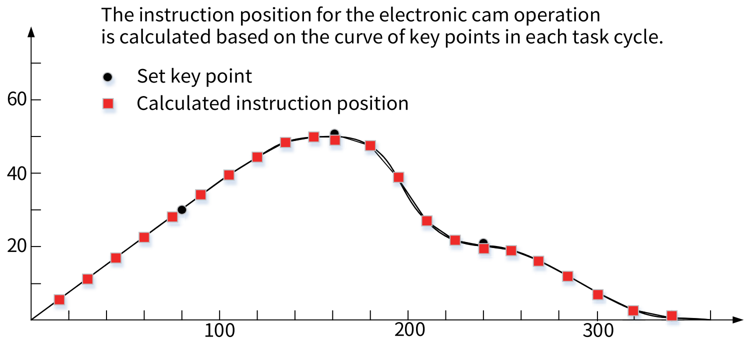
Phase and displacement values of cam data in a cam table are relative quantities expressed in relation to the start point "0.0".

In cam operation, the slave axis displacement is calculated based on the master axis phase and the set curve type, to control the operation of the slave axis.

After a cam table is created with the cam editor in AutoShop, cam data in the cam table can be modified in the user program.

Cam table

	Phase	Displacement
Start point	0	0
	80	30
	160	50
	240	20
End point	360	0



16.5.3.2 Cam Table Specifications

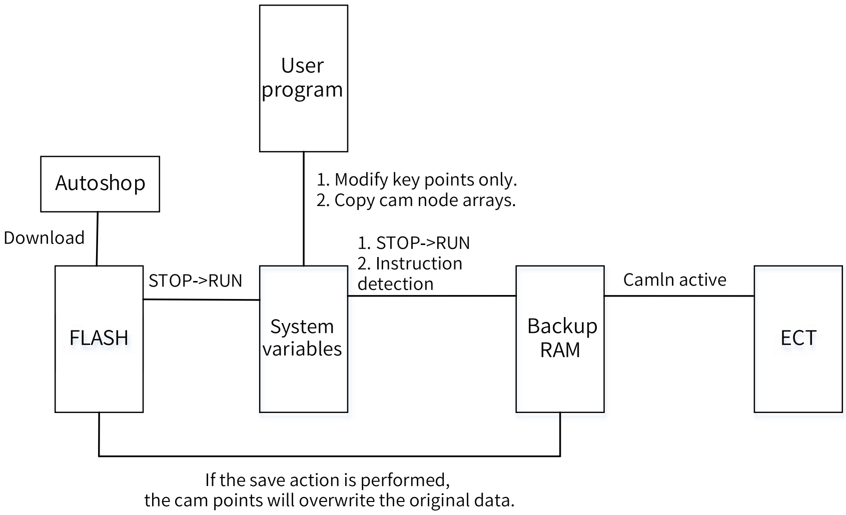
Observe the following specifications when creating cam tables.

Table 16–7 Cam table specifications

Item	Description
Total number of cam key points supported by each cam table	361
Total number of cam tables supported	16
Number of cam tables that can be executed simultaneously in the PLC	8
Rules for switching cam tables during cam operation	Switch cam tables using the MC_CamIn instruction, and the newly selected cam table takes effect in the next cam cycle.
Reading and writing cam data	View the status and key point data of each cam table by using the global variable named after the cam table. You can modify key point data in a cam table directly and make the modification take effect by using the MC_GenerateCamTable instruction. The cam will act according to the modified cam table in the next cam cycle.
Saving cam tables	Modified cam tables can be saved to the non-volatile memory of the PLC using the MC_SaveCamTable instruction.

16.5.3.3 Cam Table Data Flow

The data execution flow of cam tables is shown in the following figure.

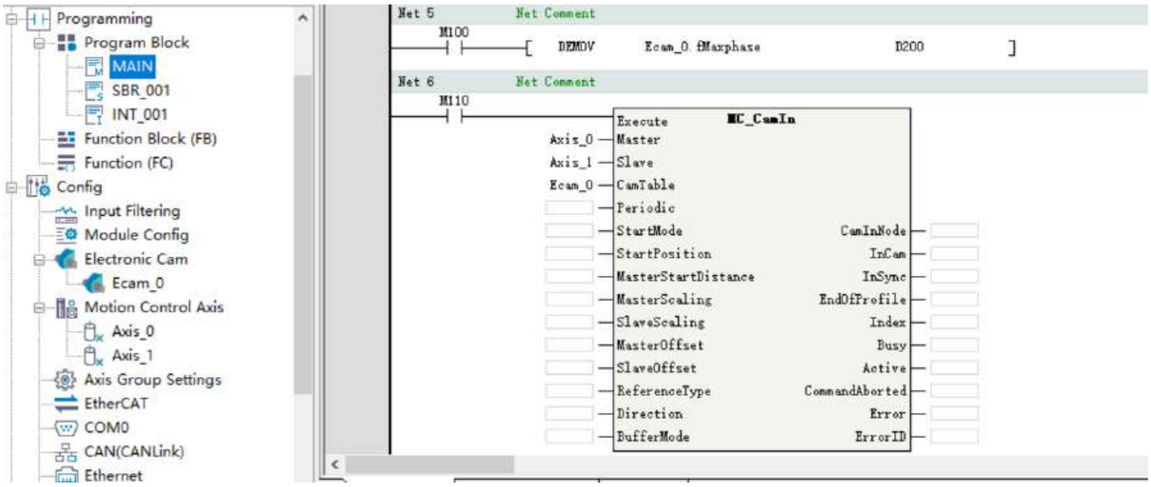


Data flow description

1. In the background, download a cam curve to the non-volatile memory.
2. In the background, upload a cam table file from the non-volatile memory.
3. When the cam table in the non-volatile memory switches from STOP to RUN after download, the cam table is loaded to the cam table system variable and initialized to the backup area.
4. The cam table in the user area is updated to the EtherCAT memory when the execution of MC_CamIn starts or after one cam motion cycle is completed. Then, EtherCAT works according to the updated cam nodes.
5. With the user program, you can modify a cam key point in the system variable or copy a new cam node array to an existing cam table, and then copy the modified cam table to the backup RAM by using the MC_GenerateCamTable instruction.
6. After modifying or creating cam key points in the user program, call the MC_GenerateCamTable instruction to check the rationality of the cam table.
7. Call the MC_SaveCamTable instruction to write the cam table in the backup area into the non-volatile memory.

16.5.3.4 Creating Cam Tables

Cam table variables can only be created in the background. Every time a cam table is added, a cam table variable is created by default. The name of the cam table variable is the name of the cam table in the configuration. You can obtain the status of the cam table by using the variable and use the status as an input parameter for cam instructions.



16.5.3.5 Switching Cam Tables

During cam execution, you can switch to a different cam table by triggering the MC_CamIn instruction. After triggering, the cam table becomes buffered, and the buffered cam table takes effect in the next cam cycle.

Only one cam table can be buffered. If multiple MC_CamIn instructions are triggered in succession, the cam table triggered earlier will be overwritten by the cam table triggered later.

Here is an example of triggering two instructions.

- Trigger M110 first. After the instruction detects that the cam parameter is set correctly, M111 (Busy) output becomes active, Axis_1 starts to move according to the cam curve set by the Ecam_0 cam table, and M112 (Active) output becomes active. If M120 is triggered before one cam cycle is completed, the Ecam_1 cam table becomes buffered and M121 (Busy) output becomes active.
- After Axis_1 completes the first cam cycle, the first cam instruction is aborted and M113 (CommandAborted) output becomes active. In this case, Axis_1 starts to move according to the cam curve set by the Ecam_1 cam table and M122 (Active) output becomes active.

16.5.3.6 Modifying Cam Table Data

Cam data can be modified through the following three methods.

1. Modify values in the cam node array of the cam table through the PLC program, and then execute the MC_GenerateCamTable instruction. The modification takes effect in the next cam cycle.

Cam node array A

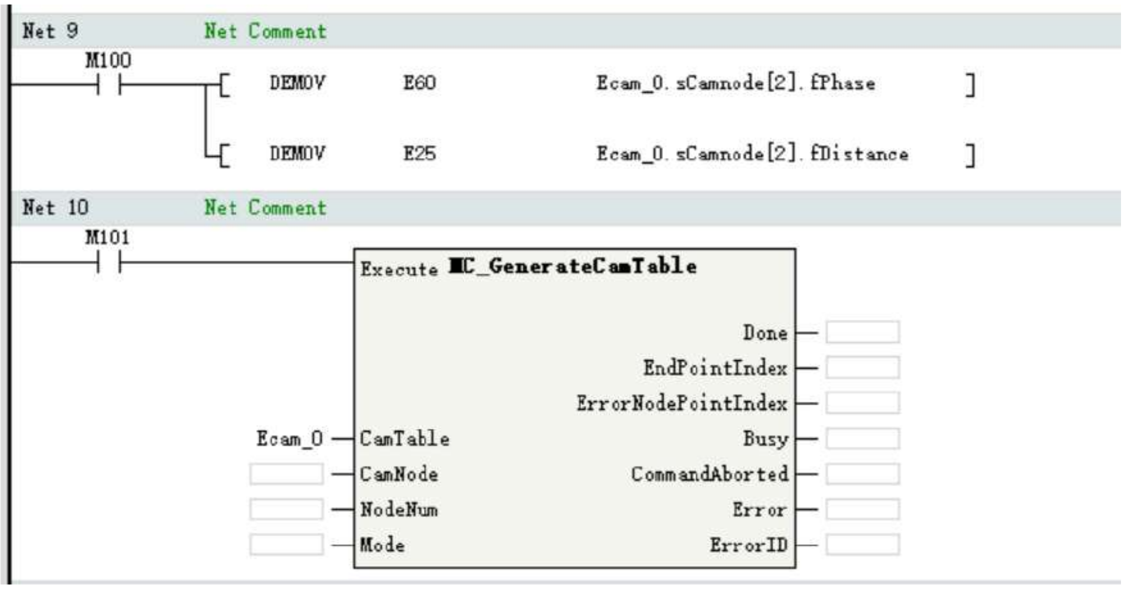
	Phase	Displacement
Start point	0	0
	40	30
	80	50
	120	20
End point	160	0
	0	0
	0	0

Modify key points only.

Cam node array A

	Phase	Displacement
Start point	0	0
	40	30
	60	25
	120	20
End point	160	0
	0	0
	0	0

Here is a program example:



```
Net 9        Net Comment
 M100
 --| |--+--[ DEMOV   E60   Ecam_0.sCamnode[2].fPhase     ]
        |
        +--[ DEMOV   E25   Ecam_0.sCamnode[2].fDistance  ]

Net 10       Net Comment
 M101
 --| |------------ Execute  MC_GenerateCamTable
                                               Done --
                                      EndPointIndex --
                                 ErrorNodePointIndex --
          Ecam_0 -- CamTable                   Busy --
                 -- CamNode          CommandAborted --
                 -- NodeNum                   Error --
                 -- Mode                    ErrorID --
```

2. Modify the number of key points in the cam table, and then execute the MC_GenerateCamTable instruction. The modification takes effect in the next cam cycle.

Cam node array A

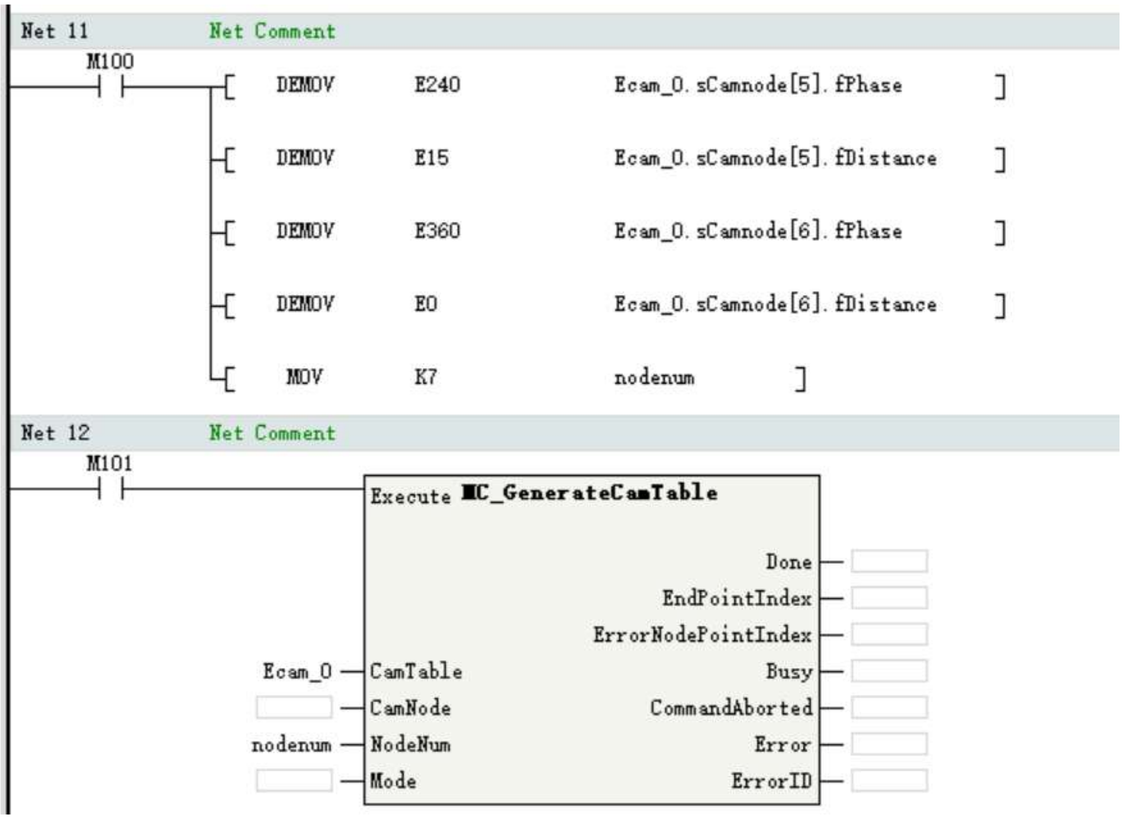
	Phase	Displacement
Start point	0	0
	40	30
	80	50
	120	20
End point	160	0
	0	0
	0	0

Add key points. →

Cam node array A

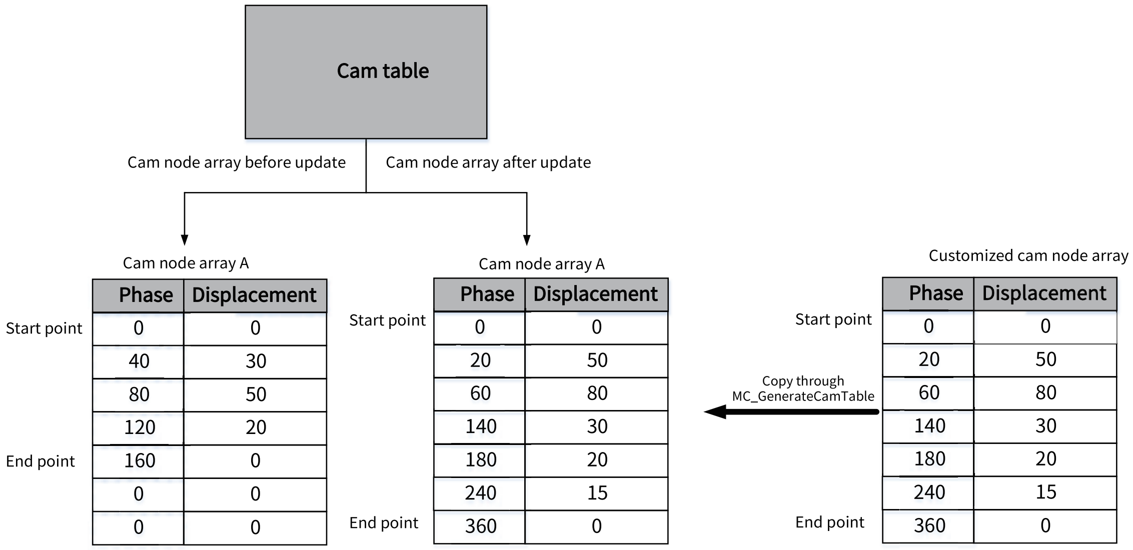
	Phase	Displacement
Start point	0	0
	40	30
	80	50
	120	20
	160	0
	240	15
End point	360	0

Here is a program example:



```
Net 11          Net Comment
M100
─┤ ├──┬─[ DEMOV   E240   Ecam_0.sCamnode[5].fPhase     ]
      ├─[ DEMOV   E15    Ecam_0.sCamnode[5].fDistance  ]
      ├─[ DEMOV   E360   Ecam_0.sCamnode[6].fPhase     ]
      ├─[ DEMOV   E0     Ecam_0.sCamnode[6].fDistance  ]
      └─[ MOV     K7     nodenum   ]
Net 12          Net Comment
M101
─┤ ├──────────── Execute  MC_GenerateCamTable
                                                  Done ─
                                         EndPointIndex ─
                                   ErrorNodePointIndex ─
           Ecam_0 ─ CamTable                      Busy ─
                  ─ CamNode             CommandAborted ─
          nodenum ─ NodeNum                      Error ─
                  ─ Mode                       ErrorID ─
```

3. Create a completely new cam node array through the PLC program, and then copy the values in the cam node array to the cam table by using the MC_GenerateCamTable instruction. The modified cam table takes effect in the next cam cycle.



Cam table
Cam node array before update
Cam node array after update
Copy through MC_GenerateCamTable

Cam node array A (before update)

	Phase	Displacement
Start point	0	0
	40	30
	80	50
	120	20
End point	160	0
	0	0
	0	0

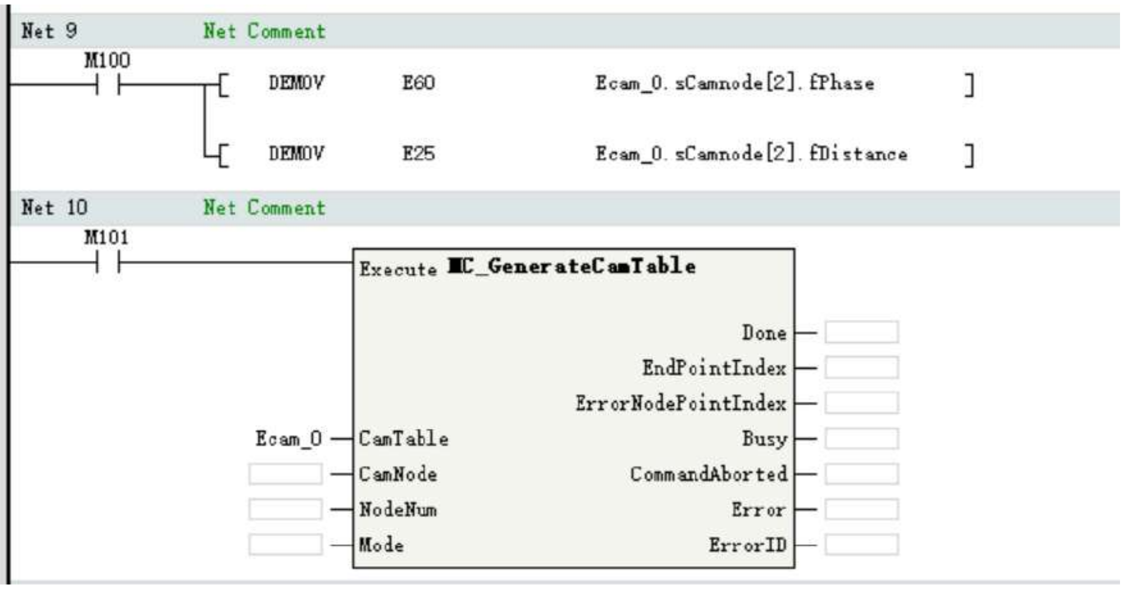
Cam node array A (after update)

	Phase	Displacement
Start point	0	0
	20	50
	60	80
	140	30
	180	20
	240	15
End point	360	0

Customized cam node array

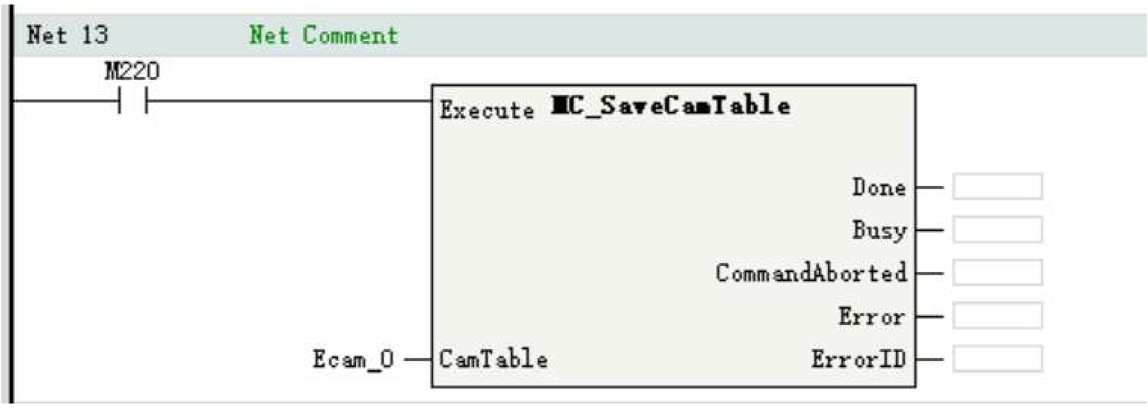
	Phase	Displacement
Start point	0	0
	20	50
	60	80
	140	30
	180	20
	240	15
End point	360	0

Here is a program example:



16.5.3.7 Saving Cam Tables

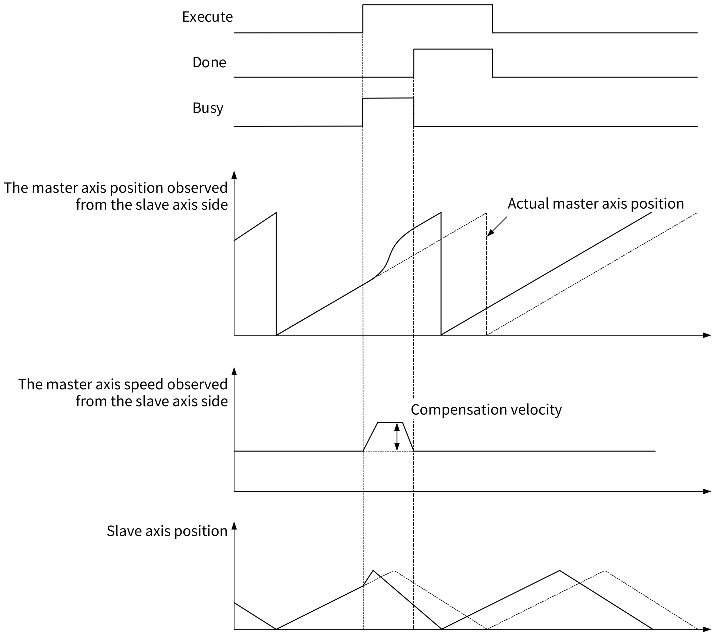
Modified cam tables can be written to the non-volatile storage space by using the MC_SaveCamTable instruction.



16.5.4 Master Axis Phase Compensation

This function allows the master axis phase to be shifted (observed from the slave axis) for instructions in operation.
Starting the MC_Phasing (master axis phase shift) instruction can compensate the phase for the synchronized control instruction.

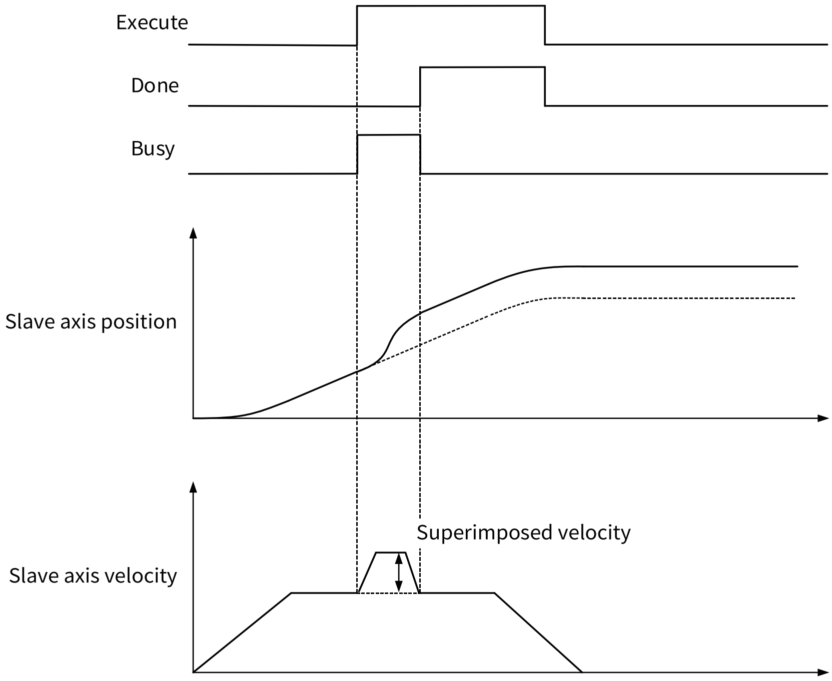
The MC_Phasing (master axis phase shift) instruction can specify parameters such as phase compensation, target speed, acceleration rate, and deceleration rate.


Execute
Done
Busy
The master axis position observed from the slave axis side
Actual master axis position
The master axis speed observed from the slave axis side
Compensation velocity
Slave axis position

16.5.5 Motion Superimposition

Call the MC_MoveSuperImposed instruction to implement motion superimposition on the motion control axis.

The MC_MoveSuperImposed (motion superimposition) instruction can specify parameters such as position compensation, target speed, acceleration rate, and deceleration rate.



16.5.6 Methods of Handling Single-Axis Configuration Parameters in Cam or Gear

Methods of handling single-axis configuration parameters in cam or gear are shown in the following table.

Table 16–8 Methods of handling single-axis configuration parameters

Para.	Method of Handling
Gear ratio setting	The master and slave axes of the cam and gear support gear ratio modification. Set gear ratios in the "Unit conversion setting" interfaces of the cam and gear, respectively.
Encoder mode selection	The master and slave axes of the cam and gear support drive encoder mode setting. The encoder can be set to incremental or absolute mode. Set this parameter in the "Mode/ Parameter setting" interfaces of the cam and gear, respectively.
Circular/linear mode setting	The master and slave axes of the cam and gear support linear and circular modes. Set this parameter in the "Mode/Parameter setting" interfaces of the cam and gear, respectively.
Limit handling	In linear mode, the slave axes of the cam and gear support software limit but do not support limit-based deceleration. The slave axes stop immediately after encountering the limit. In both linear and circular modes, the slave axes of the cam and gear support hard limit, and stop immediately after encountering the limit.
Deceleration rate at axis fault	When an abnormal instruction parameter causes a fault on a slave axis, the slave axis decelerates according to the deceleration rate specified by the axis fault deceleration parameter and then enters the ErrorStop state.
Following error	The slave axes of the cam and gear support following error during operation.
Speed limit	Instructions related to the cam and gear are not restricted by the maximum speed parameter in the configuration.
Acceleration rate limit	The deceleration rate of the MC_CamOut instruction is restricted by the maximum acceleration parameter in the slave axis configuration. The instruction reports an error and the axis enters the ErrorStop state when the deceleration rate exceeds the limit. The acceleration rate and deceleration rate of the MC_GearIn instruction are restricted by the maximum acceleration parameter in the slave axis configuration. The instruction reports an error and the axis enters the ErrorStop state when the acceleration rate or deceleration rate exceeds the limit. The deceleration rate of the MC_GearOut instruction is restricted by the maximum deceleration parameter in the slave axis configuration. The instruction reports an error and the axis enters the ErrorStop state when the deceleration rate exceeds the limit.
Torque limit	The torque limit value is written into the servo drive as a startup parameter and is controlled by the servo drive.

17 Offline Commissioning

17.1 Overview

⚠ Caution

Offline commissioning is available in AutoShop V4.2.0.0 and later versions.

Offline commissioning allows users to commission the logic, motion control, and communication functions of the program.

Offline commissioning covers the motion control axis, module configuration, communication, and online modification functions. It comes with a status display interface that facilitates real-time monitoring of the status of the PLC and module I/O channels. In addition, offline commissioning can be used with the online simulation function of the IT7000 series HMI to achieve simulation commissioning, making commissioning more convenient.

The following table lists the functions supported or not supported in the offline commissioning mode.

Supported or Not	Function	Description
Supported	Program	Supports main program, subprogram, and FB/FC.
		Supports the "timed interrupt" subprogram.
		Supports online modification.
	Motion control	Supports local pulse axis and EtherCAT bus servo axis.
		Supports local encoder axis; uses simulation of internal clock signal.
	Communication	Supports serial communication configuration; uses the COM9 port of computers; supports Modbus master/slave protocol; supports free serial protocol.
		Supports Ethernet Modbus-TCP protocol; supports Ethernet TCP/UDP communication.
		Supports online simulation with HMI. Communication is established through specified ports. A communication variable table is generated in the background and imported to the HMI. Monitoring objects include all components and customized variables.
	Directive	Supports all instructions, except the HOUR, DHOUR/TWR, TRD instructions.
	Extension	Supports extension module I/O, local DIDO, and function virtualization.
	Other functions	Supports RTC clock; uses the clock built in Windows; does not support PLC time setting.
		Keeps consistent with PLC.
Not supported	Basic	Does not support time setting.
		Does not support "Set/Modify Login PLC Password" or "Delete Login PLC Password".
		Cannot enter or exit offline commissioning during compile, download, or firmware upgrade process.
	Program	Does not support hard interruption, edge interruption, or comparison interruption.
	Motion control	Does not support bus encoder axis.
	Communication	Does not support CANlink or CANopen.
		Does not support EtherCAT communication or instructions.

Note

- Offline commissioning of the operating system may result in some difference between the timer and the actual PLC, involving the timed-interrupt subprogram and internal clock signal.
- In offline commissioning, the CPU and memory usage displayed is the CPU and memory usage of the PC.
- Functions that are not supported cannot be used normally, but they do not affect offline commissioning and therefore need no special treatment.

17.2 Starting Offline Commissioning

Prerequisite: The project to be commissioned offline is created or opened.

1. In the toolbar, click to start the offline debugger. Run the PLC program that has been created or opened.
   The interface is comprised of the following parts.

No.	Description
①	PLC status
②	Status of I/O channels of the PLC. Green indicates active I/O, whereas gray indicates inactive I/O.
③	Status of extension modules. Status of digital extension modules are consistent with the PLC, whereas the section of analog extension modules shows the values in the current mapped registers.

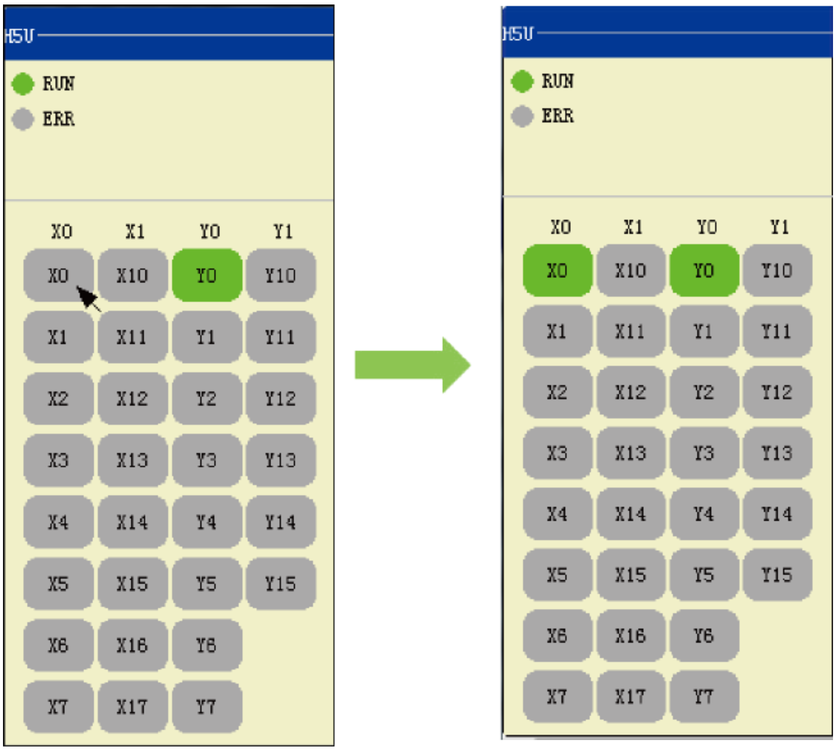
Note

After starting offline commissioning, you can perform the following operations:

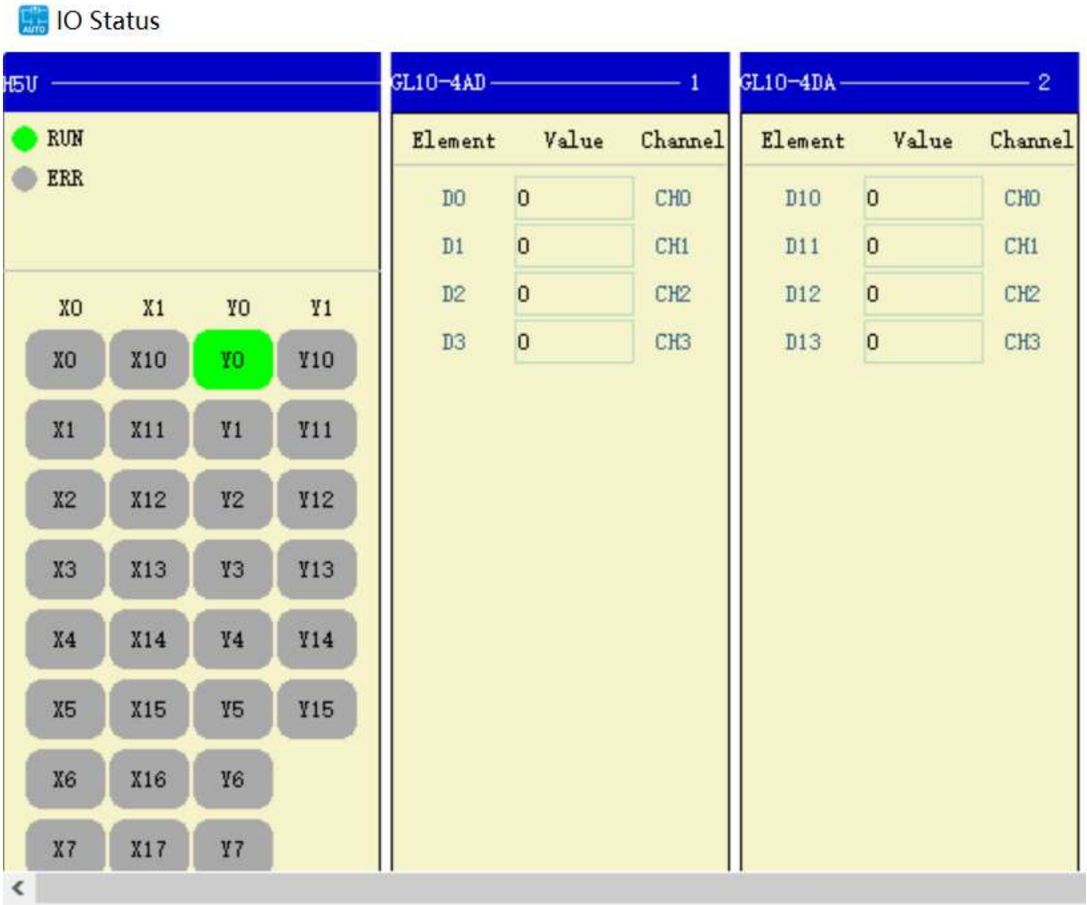
- Monitor the status of the program online, including the inputs and outputs of the PLC.
- Modify or download the PLC program.
- Modify the program online.

2. Commission digital and analog terminals.

- Commission digital terminals: For any I/O point that is not controlled by the program, directly click the I/O point to switch its ON/OFF state.
  I/O points controlled by the program run according to the logic of the program.



- Commission analog terminals: Click a value input field to enter an analog value.
  After the value is input, the parameter is written to the corresponding mapping element. Similar to digital terminals, mapping elements controlled by the program run according to the logic of the program.



IO Status

H5U

● RUN

● ERR

X0	X1	Y0	Y1
X0	X10	Y0	Y10
X1	X11	Y1	Y11
X2	X12	Y2	Y12
X3	X13	Y3	Y13
X4	X14	Y4	Y14
X5	X15	Y5	Y15
X6	X16	Y6	
X7	X17	Y7	

GL10-4AD 1

Element	Value	Channel
D0	0	CH0
D1	0	CH1
D2	0	CH2
D3	0	CH3

GL10-4DA 2

Element	Value	Channel
D10	0	CH0
D11	0	CH1
D12	0	CH2
D13	0	CH3

Note

Actual sensors cannot be connected during offline commissioning. Therefore, analog input values must be typed manually for offline commissioning of analog terminals.

17.3 Motion Control Axes in Offline Commissioning

Local pulse axis and bus servo axis

- In offline commissioning, local pulse axis and bus servo axis can be used without special modifications in the program.
- All functions, including probes and hardware limits, related to external input cannot be used in offline commissioning.
- Using the homing instruction does not need special settings. When the homing instruction is used, homing mode No. 35 is activated (that is, the current position is taken as the home).

Local encoder axis and bus encoder axis

- To use local encoder axis in offline commissioning, select the single-phase counting mode and select internal clock signal (1 ms or 1 µs) as the signal source. If other modes are used, the local encoder axis cannot start counting.
- In offline commissioning, only HC_Counter and HC_Preset instructions are supported for local encoder axis.

- Bus encoder axis is not supported in offline commissioning.

17.4 Simulation Commissioning with InoTouchPad

17.4.1 Overview

The H5U or Easy series PLC can work with the IT7000 series HMI to achieve simulation commissioning without physical objects through the commissioning function of AutoShop and InoTouchPad software.

The PLC and HMI communicate with each other through TCP monitoring protocol, supporting customized variables and read and write operations on soft elements.

17.4.2 PLC Configuration

The PLC and HMI employ internal communication for simulation commissioning. Therefore, there is no need to configure the PLC's IP address. Users only need to export the PLC's variables to the HMI monitoring variable table to complete the PLC configuration.

1. After compiling the user program, open the variable table, right-click any area of the variable table, and select "Export HMI Monitoring Variable Table (H)" in the shortcut menu.
2. Select the path to save the file to be exported, enter the file name, and click "Save" to complete the export.

17.4.3 HMI Configuration

To use simulation commissioning, add a PLC connection on the HMI and import the variable table. For H5U, specific operation steps are as follows.

1. Create an HMI connection.

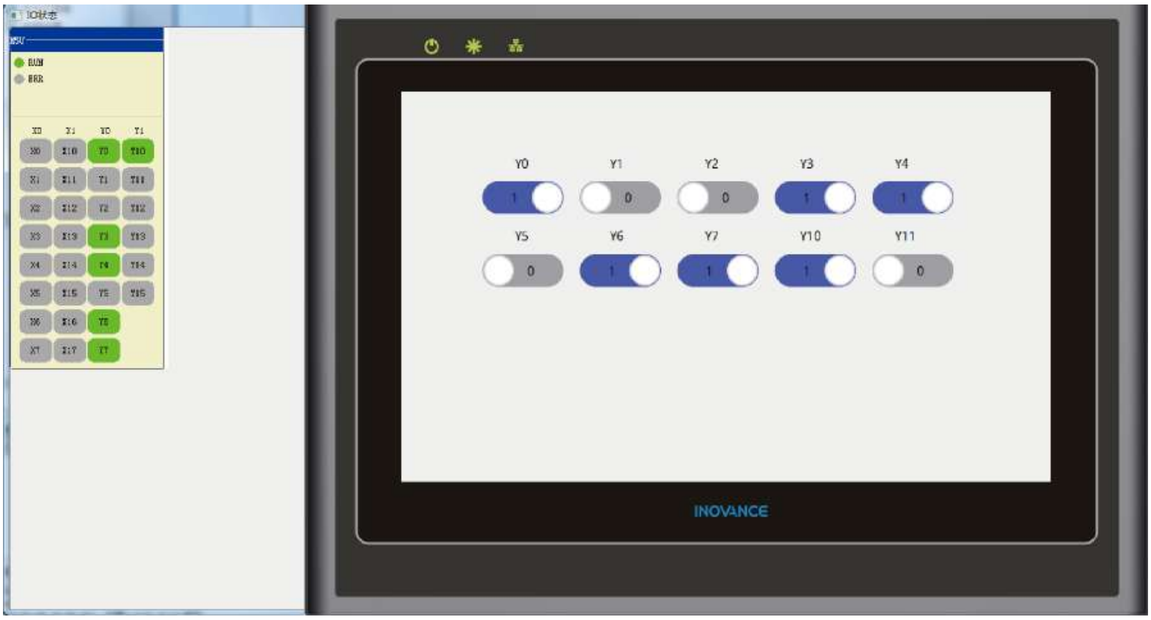
   a. Double-click "Connection" to open the connection tab.
   b. Click + to add a connection.
   c. In the "Communication protocol" column, select "H5U TCP monitoring protocol".
   d. In the "IP Address" field, enter "127.0.0.1".

2. Import the HMI monitoring variable table.

   a. Double-click "Add variable group" to add a variable group.
   b. Right-click the newly created variable group and select "Import".
   c. In the opened dialog box, select the HMI monitoring variable table exported from the PLC and click "Open".
   d. In the "Select device" field, select the HMI connection created in step 1.
   After the import is completed, variables are automatically generated in the variable group.

17.4.4 Starting Commissioning

After editing the program and importing the variable table for HMI and PLC, you can start the offline commissioning function of AutoShop and the online simulation function of InoTouchPad for commissioning.



18 Memory Management

18.1 Overview

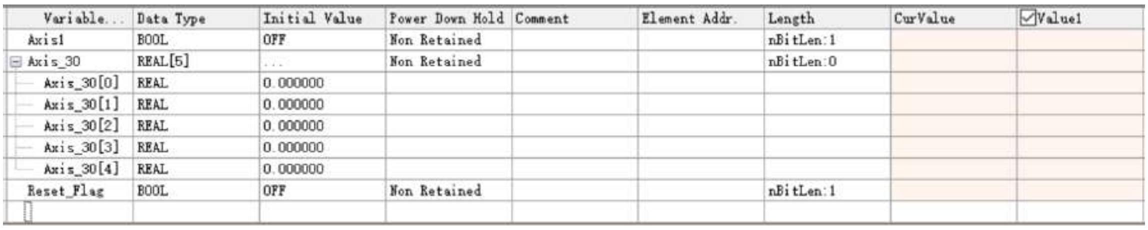
Memory management includes customized variable and soft element variable memory management. Variable memory data at a specific moment can be obtained and used as the basis for commissioning and analysis. Variable memory data at a specific moment can also be saved as recipe data for commissioning parameters of different processes or recipe parameters of multiple steps of one process. Specifically, variable memory data can be used in the following scenarios:

- When the program encounters an exception, obtain the current variable memory data for problem analysis.
- Obtain multiple sets of variable memory data parameters at a specific moment and save them as recipe parameter files for use by other machines.
- Monitor the values of all the data in the current variable table in real time.
- Synchronize the current variable memory data parameters to the initial values.
- Save data of different recipe parameters when commissioning different processes of one program.
- Save different sets of recipe parameters when commissioning different steps of one process.

18.2 Memory Management of Customized Variable Tables

18.2.1 Expanding and Collapsing Complex Type Variables

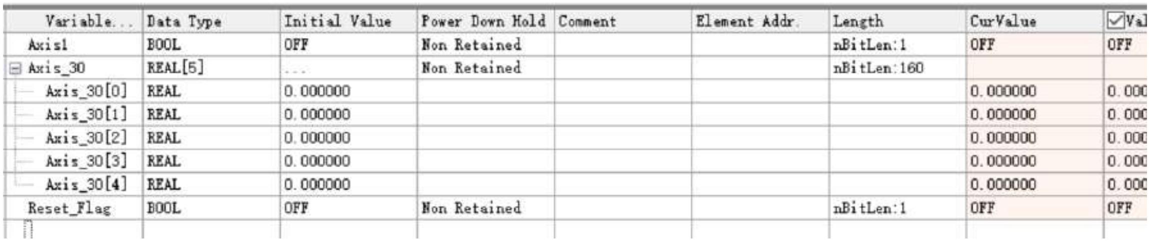
Complex type variables contained in the customized variable table are arrays and structures. The system supports expanding and collapsing sub-members of arrays and structures. For sub-members, only the initial value, comment, and data columns are editable, while values in other columns are not editable.



Variable...	Data Type	Initial Value	Power Down Hold	Comment	Element Addr.	Length	CurValue	☑Value1
Axis1	BOOL	OFF	Non Retained			nBitLen:1		
⊟ Axis_30	REAL[5]	...	Non Retained			nBitLen:0		
Axis_30[0]	REAL	0.000000						
Axis_30[1]	REAL	0.000000						
Axis_30[2]	REAL	0.000000						
Axis_30[3]	REAL	0.000000						
Axis_30[4]	REAL	0.000000						
Reset_Flag	BOOL	OFF	Non Retained			nBitLen:1		

18.2.2 Monitoring Variables

The variable table supports monitoring. Click the "Download and monitoring" button in the user project, and the system will monitor all variables displayed on the current variable table page without the need to add variables to the monitoring table separately.

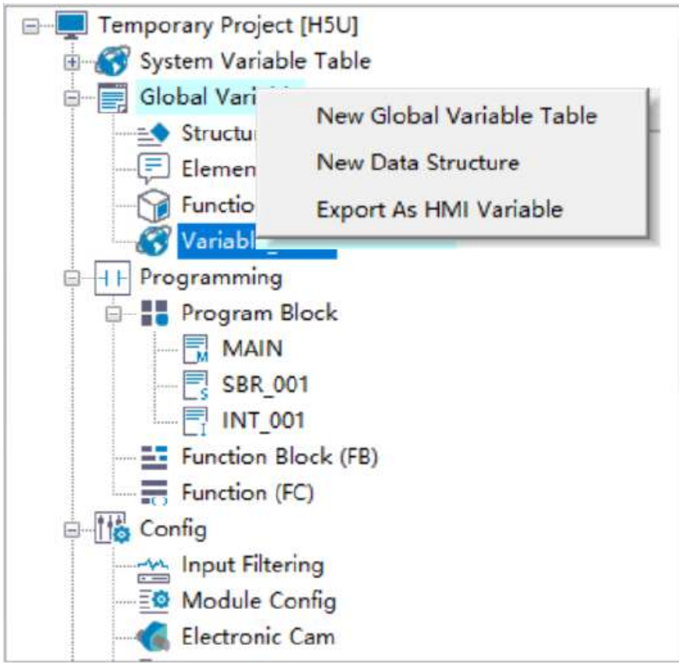


Variable...	Data Type	Initial Value	Power Down Hold	Comment	Element Addr.	Length	CurValue	☑Val
Axis1	BOOL	OFF	Non Retained			nBitLen:1	OFF	OFF
⊟ Axis_30	REAL[5]	...	Non Retained			nBitLen:160		
Axis_30[0]	REAL	0.000000					0.000000	0.000
Axis_30[1]	REAL	0.000000					0.000000	0.000
Axis_30[2]	REAL	0.000000					0.000000	0.000
Axis_30[3]	REAL	0.000000					0.000000	0.000
Axis_30[4]	REAL	0.000000					0.000000	0.000
Reset_Flag	BOOL	OFF	Non Retained			nBitLen:1	OFF	OFF

18.2.3 Reading and Writing Memory Data

In the running state, for a single variable table, select the specified array or structure, right-click it, and select "Write Memory" to write non-null data to the PLC. In the monitoring state, right-click and select "Read Memory" to read the current variable memory data from the PLC to the selected data column of the variable table.

In the running state, you can upload or download newly-added variable values through one click. Specifically, right-click "Global Variable", and select "Upload project variable value" to batch read the current variable memory data from the PLC to the selected data columns of the variable table; or select "Download project variable value" to batch write non-null data in the variable table to the PLC.



Note

- Select the "Auto stop for writing" option, and the PLC will automatically stop before memory data is written.
- If the "Auto stop for writing" option is not selected, memory data may not be written in the same scan cycle.

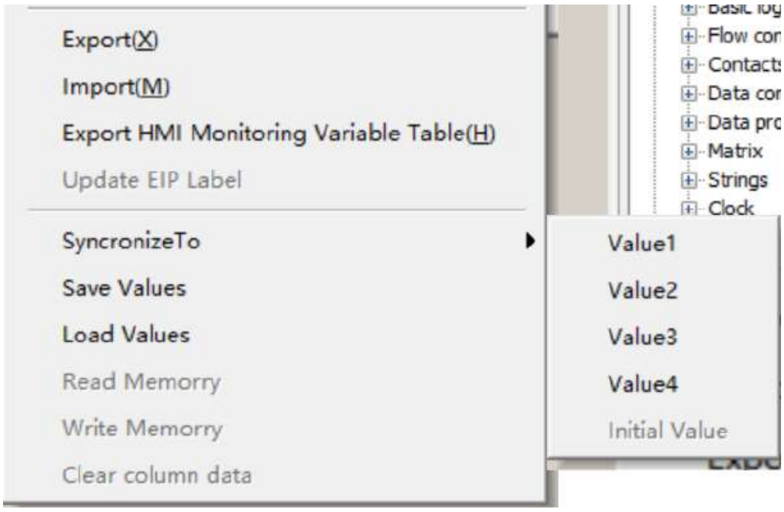
In the stop state, only retentive variables can be written.

18.2.4 Synchronizing and Clearing Data

Data values, current values, and initial values can be synchronized with each other.

Synchronizing data

For example, to synchronize initial values to current values, select the target initial value item (or select multiple initial value items), right-click and select "Synchronize To" > "Value".



Note

Current values cannot be synchronized to initial values.

Clearing data

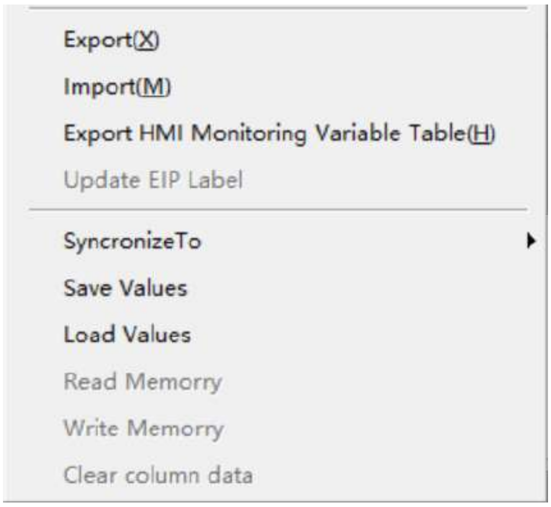
Select "Clear Column Data" to delete the currently selected data value. This operation can only be performed on data values.

For example, to clear the Data3 column, select "Data3", right-click it, and select "Clear Column Data".

18.2.5 Saving and Loading Data

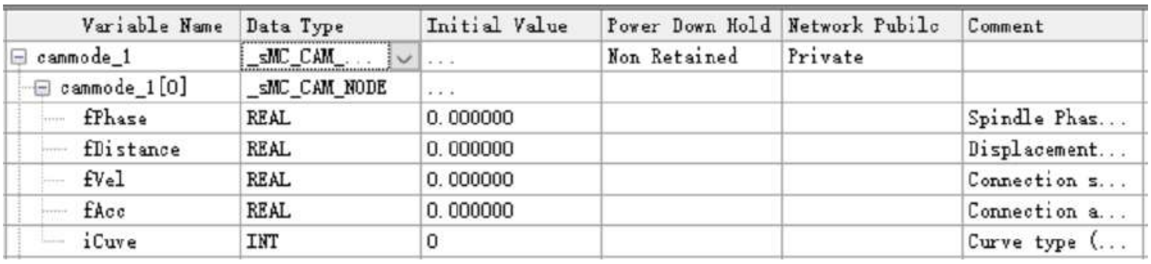
You can select "Save Values" in the shortcut menu to save specified data values in a CSV file. Then, open the CSV file using EXCEL to edit the data values.

You can also select "Load Values" in the shortcut menu to import the edited CSV recipe file to the PLC.



18.2.6 Editing Initial Values and Comments of Variables

You can directly expand the variable table to edit initial values and comments of member data, or define them in the customized variable box in the program. These two methods are equivalent to each other.



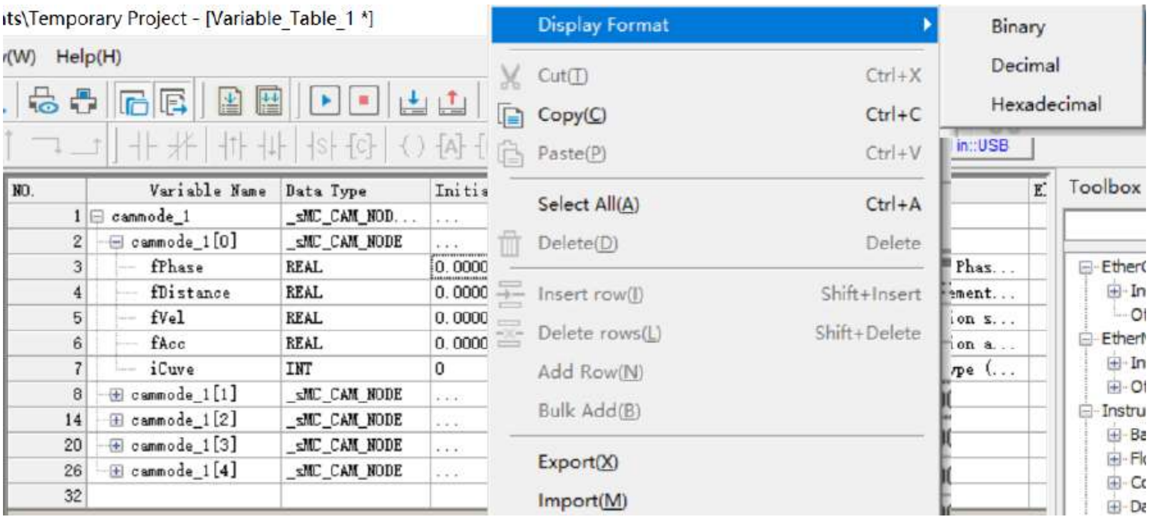
Variable Name	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment
⊟ cammode_1	_sMC_CAM_...	...	Non Retained	Private	
⊟ cammode_1[0]	_sMC_CAM_NODE	...			
fPhase	REAL	0.000000			Spindle Phas...
fDistance	REAL	0.000000			Displacement...
fVel	REAL	0.000000			Connection s...
fAcc	REAL	0.000000			Connection a...
iCuve	INT	0			Curve type (...

Note

English characters and special characters such as commas (,), brackets ([]), and parentheses (()) in comments of variable member data are automatically filtered out.

18.2.7 Switching and Displaying Number Systems

Customized variable tables allow switchover between decimal, binary, and hexadecimal displays.



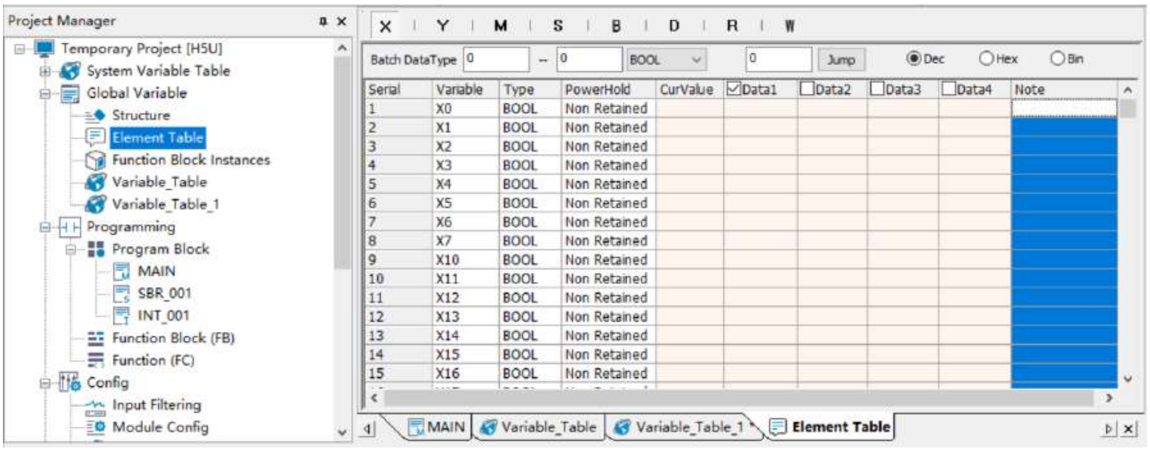
its\Temporary Project - [Variable_Table_1 *]

NO.	Variable Name	Data Type	Initia
1	⊟ cammode_1	_sMC_CAM_NOD...	...
2	⊟ cammode_1[0]	_sMC_CAM_NODE	...
3	fPhase	REAL	0.0000
4	fDistance	REAL	0.0000
5	fVel	REAL	0.0000
6	fAcc	REAL	0.0000
7	iCuve	INT	0
8	⊞ cammode_1[1]	_sMC_CAM_NODE	...
14	⊞ cammode_1[2]	_sMC_CAM_NODE	...
20	⊞ cammode_1[3]	_sMC_CAM_NODE	...
26	⊞ cammode_1[4]	_sMC_CAM_NODE	...
32			

18.3 Memory Management of Soft Elements

18.3.1 Operation Interface

The element table provides the comment and memory management functions, as shown in the following figure.



Batch DataType 0 -- 0 BOOL   0 Jump   ● Dec ○ Hex ○ Bin

Serial	Variable	Type	PowerHold	CurValue	☑Data1	□Data2	□Data3	□Data4	Note
1	X0	BOOL	Non Retained						
2	X1	BOOL	Non Retained						
3	X2	BOOL	Non Retained						
4	X3	BOOL	Non Retained						
5	X4	BOOL	Non Retained						
6	X5	BOOL	Non Retained						
7	X6	BOOL	Non Retained						
8	X7	BOOL	Non Retained						
9	X10	BOOL	Non Retained						
10	X11	BOOL	Non Retained						
11	X12	BOOL	Non Retained						
12	X13	BOOL	Non Retained						
13	X14	BOOL	Non Retained						
14	X15	BOOL	Non Retained						
15	X16	BOOL	Non Retained						

① Soft element switchover area. Switch to different soft elements to manage their memories and comments.

② This area is used to batch modify the data type of selected soft elements on the current page. To select soft elements, enter the start and end numbers of the elements in the two edit boxes.

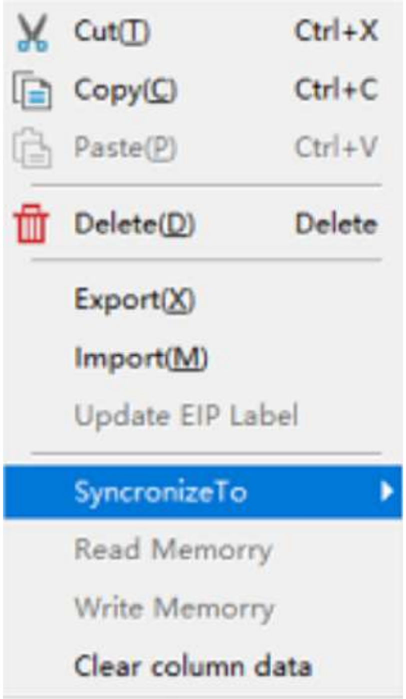
③ Click the drop-down box to select the data type to be changed to. For example, switch to the D elements, enter 50 and 100 to select the D50 to D100 elements, and then click the drop-down box to select REAL. The data type of D50, D52, D54...D100 is changed to REAL. (Note: REAL and DINT each are composed of 32 bits, occupying two soft elements.)

④ This area can redirect you to a specific element. For example, switch to the D elements, enter 1000 in this box, and then click "Jump" or press Enter. The page automatically redirects to the D1000 element. (Note: For the X and Y elements, only octal numbers of up to five digits can be entered in an input box. If the entered number exceeds the total number of elements, the page redirects to the end.)

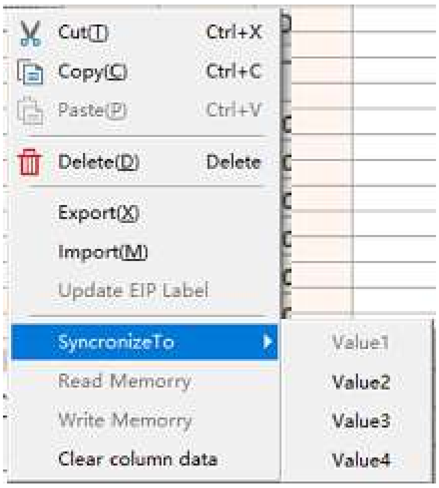
⑤ This area is used to change the display format of the data value and current value columns. For example, clicking the "Hex" option will switch all data to hexadecimal display.

18.3.2 Data Operation

The shortcut menu of soft elements provides multiple function options, as shown in the following figure.



- The Cut, Copy, Paste, and Delete options are available for the value columns and the comment column.
- The Edit option is available for only six columns, including the data type column, value columns, and comment column.
  The value columns can be edited in any mode, while other columns cannot be edited in the monitoring mode. All these columns can be edited in the online modification mode.
- Import and Export: Only rows with modified data can be exported. If there is any data error in a row during import, the system reports the error in the information output window and skips the row. When the number of error rows exceeds 100, the import stops.
- Synchronize To: synchronizes all data in a selected column to a value column selected using the menu. For example, right-click a cell in the "Value1" column and choose "Synchronize To" > "Value2" to overwrite the Value2 column with the data in the Value1 column.



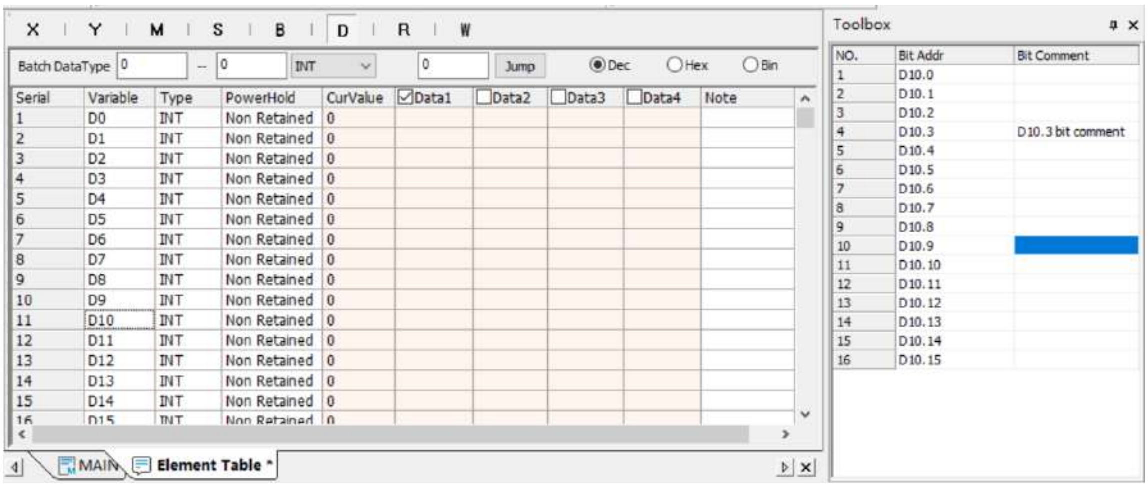
- Read Memory and Write Memory: Reading or writing soft element memories is allowed only in the online modification mode or monitoring mode. On the interface of a single element, tick the header of a value column, right-click and select "Read Memory" or "Write Memory" to read or write to the specific column you ticked. To read or write data values to the entire soft element table in batch, right-click "Global Variable", and select "Upload project variable value" to batch read the current variable memory data from the PLC to the selected data columns of the variable table; or select "Download project variable value" to batch write non-null data in the variable table to the PLC.

- Clear Column Data: clears all data in a column.

18.3.3 Bit Comments

Select word elements (D, R, or W) in the element table and then select a row. Bit comments of the row are displayed in the toolbox area. You can edit the bit comments in the toolbox area.

For example, select D10, edit the comment of D10.3, and save the modification. As a result, the bit comment is displayed in the ladder diagram.



Serial	Variable	Type	PowerHold	CurValue	☑Data1	□Data2	□Data3	□Data4	Note
1	D0	INT	Non Retained	0					
2	D1	INT	Non Retained	0					
3	D2	INT	Non Retained	0					
4	D3	INT	Non Retained	0					
5	D4	INT	Non Retained	0					
6	D5	INT	Non Retained	0					
7	D6	INT	Non Retained	0					
8	D7	INT	Non Retained	0					
9	D8	INT	Non Retained	0					
10	D9	INT	Non Retained	0					
11	D10	INT	Non Retained	0					
12	D11	INT	Non Retained	0					
13	D12	INT	Non Retained	0					
14	D13	INT	Non Retained	0					
15	D14	INT	Non Retained	0					
16	D15	INT	Non Retained	0					

NO.	Bit Addr	Bit Comment
1	D10.0	
2	D10.1	
3	D10.2	
4	D10.3	D10.3 bit comment
5	D10.4	
6	D10.5	
7	D10.6	
8	D10.7	
9	D10.8	
10	D10.9	
11	D10.10	
12	D10.11	
13	D10.12	
14	D10.13	
15	D10.14	
16	D10.15	

18.3.4 Rules of Editing Data Types

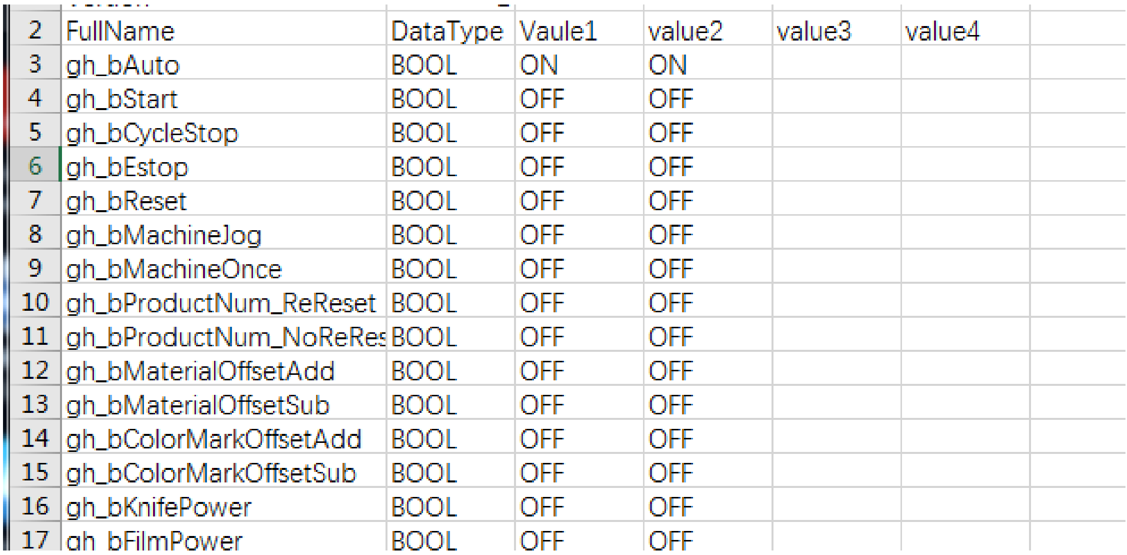
1. For bit elements (X, Y, M, S, and B), the data type can only be set to BOOL, and the values are either ON or OFF.
2. For word elements, the data type can be set to INT, DINT, or REAL. An INT element is composed of 16 bits, whereas a DINT or REAL element is composed of 32 bits.
3. A 32-bit word element occupies two element rows. Therefore, when a word element is set as the REAL or DINT type, values in the row next to the word element are automatically cleared and data type of that row cannot be edited. In addition, the data column of the row preceding the word element cannot be edited. In memory reading, the row next to a REAL or DINT element are skipped.

18.4 Function Demonstration

Save different sets of recipe parameters when commissioning different steps of one process.

1. Create a standard project, define variables and structures and write the program for the project, and then download the project to the PLC.
2. Enter project monitoring.

3. Open the variable table and tick "Value1". At a specific moment, right-click and select "Read Memory" to read the variable memory as the recipe parameters for the step of the specific moment. As a result, the variable memory values obtained at the specific moment are displayed in the "Value1" column. In batch reading of memory values, the memory values read through right-click at a specific moment will be read to the entire selected value column of the variable table, whether open or not, and will be used as the recipe parameters for that step.
4. Similarly, tick "Value2". At another moment, right-click and select "Read Memory" to obtain recipe parameters for another step and have them displayed in the "Value2" column. Batch reading of memory values is similar to the previous operation. Values obtained through batch reading are used as the recipe parameters for another step.
5. Save the recipe. Before saving the recipe, you can edit some of the obtained values (or export and edit them in an EXCEL file and then load the file). After editing, right-click and select "Save" to save the recipe data of different steps in a CSV file in the disk for backup.
6. Edit recipe values. Open the saved recipe CSV file in Excel and fine-tune the variable parameters.



2	FullName	DataType	Vaule1	value2	value3	value4
3	gh_bAuto	BOOL	ON	ON		
4	gh_bStart	BOOL	OFF	OFF		
5	gh_bCycleStop	BOOL	OFF	OFF		
6	gh_bEstop	BOOL	OFF	OFF		
7	gh_bReset	BOOL	OFF	OFF		
8	gh_bMachineJog	BOOL	OFF	OFF		
9	gh_bMachineOnce	BOOL	OFF	OFF		
10	gh_bProductNum_ReReset	BOOL	OFF	OFF		
11	gh_bProductNum_NoReRes	BOOL	OFF	OFF		
12	gh_bMaterialOffsetAdd	BOOL	OFF	OFF		
13	gh_bMaterialOffsetSub	BOOL	OFF	OFF		
14	gh_bColorMarkOffsetAdd	BOOL	OFF	OFF		
15	gh_bColorMarkOffsetSub	BOOL	OFF	OFF		
16	gh_bKnifePower	BOOL	OFF	OFF		
17	gh_bFilmPower	BOOL	OFF	OFF		

7. Load the recipe. After saving the data modified in step 6, right-click in the variable table and click "Load Values". As a result, the modified configuration parameters are displayed in the "Value1" and "Value2" columns.
8. Write the recipe values. Tick "Value2", right-click and select "Write Memory" to write the recipe parameters for the corresponding step into the memory.

19 Fault Diagnosis

19.1 Diagnosis Through the Panel

19.1.1 Indicators

States and meanings of the panel indicators of the H5U series are shown in the following table.

Indicator	Meaning
RUN	Current system status (running or stopped) ON: Running; OFF: Stopped
ERR	System fault
BAT	Battery alarm
BF	EtherCAT bus fault
CRUN	CAN running
CERR	CAN error

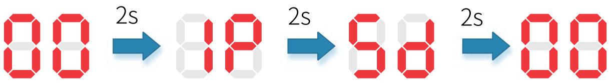
States and meanings of the panel indicators of the Easy series are shown in the following table.

Port Type	Interface Mark	Definition	Indicator Color	Description
I/O indicator	IN/OUT	I/O status	Yellow-green	• Steady ON: Input or output active • OFF: Input or output inactive
Status indicator	PWR	Power supply	Yellow-green	• Steady ON: Power supply normal • OFF: Power supply abnormal
	RUN	Normal running	Yellow-green	• Steady ON: User program running • OFF: User program stopped
	ERR	Running error	Red	• OFF: No major error • Flashing: Major error
	ETH1	EtherNET1 Link	Yellow-green	• Steady ON: Connected • Flashing: Communication in progress • OFF: Disconnected
	ETH2	EtherNET2 Link	Yellow-green	• Steady ON: Connected • Flashing: Communication in progress • OFF: Disconnected

19.1.2 MFK Key

19.1.2.1 Overview

The MFK key works with the LED display to support multi-function menu operations. Long-press the MFK key, and the LED display will toggle between different function menus at an interval of 2 seconds, as shown in the following figure.



When the function menu you want to enter is displayed on the LED display, release the MFK key and then short-press the MFK key to enter the function menu. Note: Short-press is a brief press for less than 2 seconds.

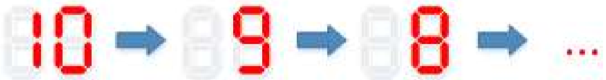
If you enter a menu that cannot be executed, the LED display shows an error.

Display Code	Name	Description
	E1	The PLC is in a non-secure state (running or downloading) and operation is prohibited.
	E2	No SD card or programming file is detected.
	E3	Multiple programming files are detected in the SD card.
	E4	Programming file data is abnormal or the device model is not compatible.
	E5	Password verification error

19.1.2.2 Restoring the Factory Default IP Address

The factory default IP address of the CPU module is 192.168.1.88. If you forget the IP address after modifying it, leading to a failure in networking and communication with another PC, you can enter the "IP" menu and reset the IP address of the CPU module to the factory default.

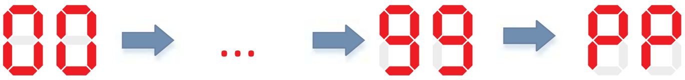
Enter the "IP" menu, and the LED display will start counting down from 10 to 0.



Before the countdown reaches 0, a short-press on the MFK key can cancel the reset. When the countdown ends, the IP is reset and the factory default IP address will be used.

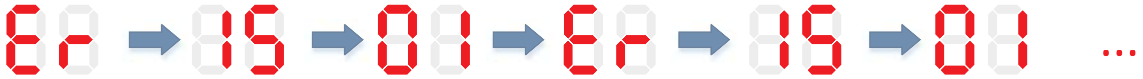
19.1.2.3 Writing User Programs Through SD Cards

Save the SD card programming file compiled using AutoShop into the "PLCProgram" directory of the SD card. Then, mount the SD card to the PLC main module. Enter the "Sd" menu to start writing the user program in the SD card into the PLC host. The LED display shows the programming progress (00 to 99). When the programming is completed, the LED display shows "PP".



19.1.2.4 LED Display of the CPU Module

When a system fault occurs, the LED display of the CPU shows the fault code, with "Er" and the code of the fault displayed alternately. For example, when the fault code is Er1501, the LED display is as follows:

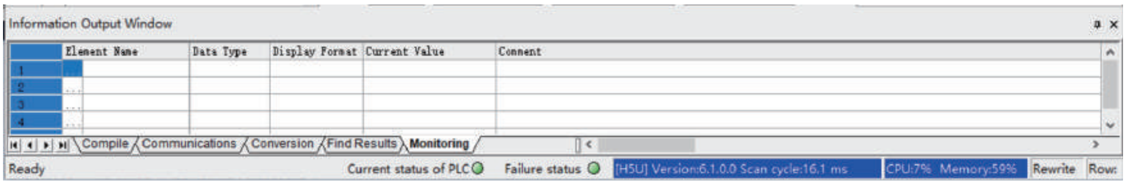


For detailed definitions of fault codes, see *“19.3 Fault Codes” on page 552*

19.2 Diagnosis Through Software

19.2.1 Obtaining Basic PLC Information

1. You can click in the toolbar to enter the monitoring mode. In the monitoring mode, the basic information of the PLC is displayed in the lower right corner of the interface. The basic information includes:


Information Output Window

	Element Name	Data Type	Display Format	Current Value	Comment
1					
2	...				
3	...				
4	...				

Ready   Current status of PLC ●   Failure status ●   [H5U] Version:6.1.0.0 Scan cycle:16.1 ms   CPU:7% Memory:59%   Rewrite   Row:

(1) PLC status indicator and fault indicator

(2) Current firmware version and program scan cycle

(3) CPU and memory utilization rates

Table 19–1 Indicator states and meanings

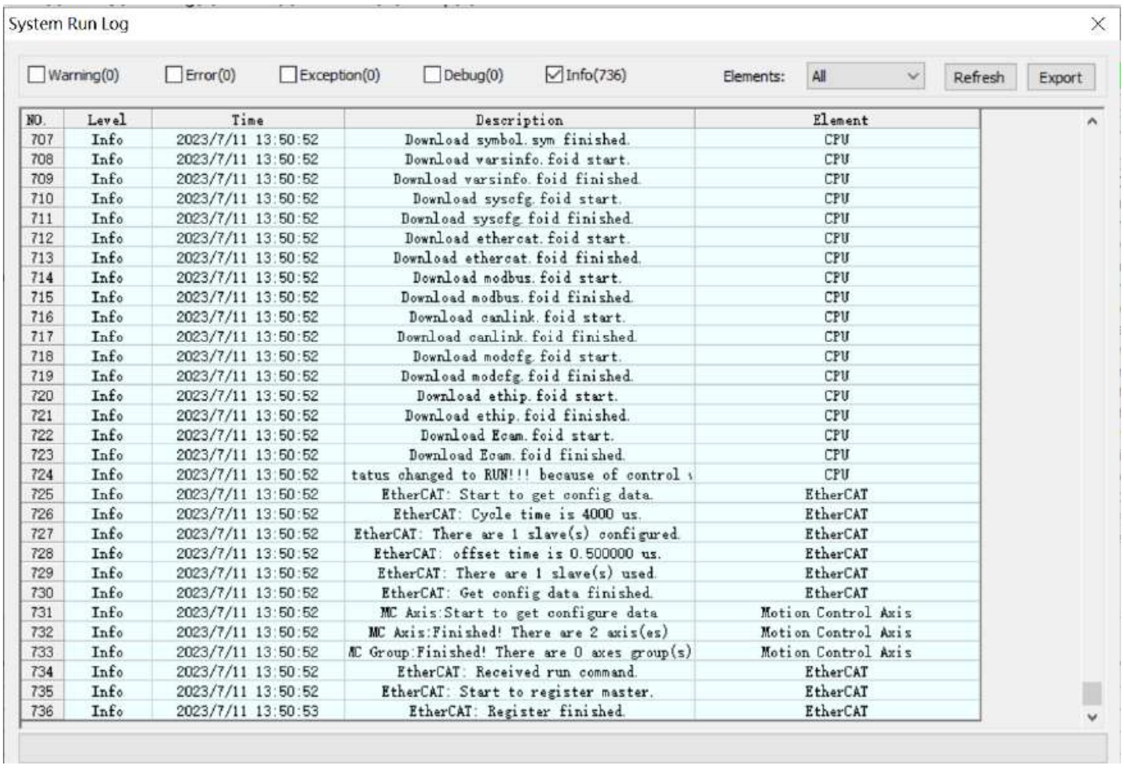
Current PLC status		Status	
Green	Running	Green	No fault
Red	Stopped	Yellow	Minor error
-	-	Red	Major error

2. Double-click the fault indicator to enter the fault diagnosis page and obtain detailed fault information.

19.2.2 Viewing Operation Logs

Operation logs include fault logs and system logs. To view operation logs, follow these steps:

1. In the toolbar, click "Debug" and then choose "System Run Log View".
2. The system operation log page is displayed.



☐ Warning(0) ☐ Error(0) ☐ Exception(0) ☐ Debug(0) ☑ Info(736) Elements: All

NO.	Level	Time	Description	Element
707	Info	2023/7/11 13:50:52	Download symbol. sym finished.	CPU
708	Info	2023/7/11 13:50:52	Download varsinfo. foid start.	CPU
709	Info	2023/7/11 13:50:52	Download varsinfo. foid finished.	CPU
710	Info	2023/7/11 13:50:52	Download syscfg. foid start.	CPU
711	Info	2023/7/11 13:50:52	Download syscfg. foid finished.	CPU
712	Info	2023/7/11 13:50:52	Download ethercat. foid start.	CPU
713	Info	2023/7/11 13:50:52	Download ethercat. foid finished.	CPU
714	Info	2023/7/11 13:50:52	Download modbus. foid start.	CPU
715	Info	2023/7/11 13:50:52	Download modbus. foid finished.	CPU
716	Info	2023/7/11 13:50:52	Download canlink. foid start.	CPU
717	Info	2023/7/11 13:50:52	Download canlink. foid finished.	CPU
718	Info	2023/7/11 13:50:52	Download modcfg. foid start.	CPU
719	Info	2023/7/11 13:50:52	Download modcfg. foid finished.	CPU
720	Info	2023/7/11 13:50:52	Download ethip. foid start.	CPU
721	Info	2023/7/11 13:50:52	Download ethip. foid finished.	CPU
722	Info	2023/7/11 13:50:52	Download Ecam. foid start.	CPU
723	Info	2023/7/11 13:50:52	Download Ecam. foid finished.	CPU
724	Info	2023/7/11 13:50:52	tatus changed to RUN!!! because of control	CPU
725	Info	2023/7/11 13:50:52	EtherCAT: Start to get config data.	EtherCAT
726	Info	2023/7/11 13:50:52	EtherCAT: Cycle time is 4000 us.	EtherCAT
727	Info	2023/7/11 13:50:52	EtherCAT: There are 1 slave(s) configured.	EtherCAT
728	Info	2023/7/11 13:50:52	EtherCAT: offset time is 0.500000 us.	EtherCAT
729	Info	2023/7/11 13:50:52	EtherCAT: There are 1 slave(s) used.	EtherCAT
730	Info	2023/7/11 13:50:52	EtherCAT: Get config data finished.	EtherCAT
731	Info	2023/7/11 13:50:52	MC Axis:Start to get configure data	Motion Control Axis
732	Info	2023/7/11 13:50:52	MC Axis:Finished! There are 2 axis(es)	Motion Control Axis
733	Info	2023/7/11 13:50:52	MC Group:Finished! There are 0 axes group(s)	Motion Control Axis
734	Info	2023/7/11 13:50:52	EtherCAT: Received run command.	EtherCAT
735	Info	2023/7/11 13:50:52	EtherCAT: Start to register master.	EtherCAT
736	Info	2023/7/11 13:50:53	EtherCAT: Register finished.	EtherCAT

- (1) Select which categories of information to display.
- (2) Select which elements to display, refresh the operation logs, or export the operation logs.

19.3 Fault Codes

The software tool prompts various categories of fault codes when faults occur in user programming. The following table lists the fault codes and corresponding solutions.

Table 19–2 Fault codes

Fault Code	Message	Description	Troubleshooting
Program			
1500	User program watchdog timed out	The user program execution time is too long and has exceeded the set program watchdog time.	Increase the watchdog time as appropriate, or check whether there is a program block with unexpectedly long execution time in the user program.
1501	Undefined instruction	The instruction is not supported.	Upgrade the PLC firmware to the version that supports the instruction.
1502	Incomplete user program, length error	The user program is incomplete, and the length is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1503	Program authorization protection identifier error. Check whether the identifier matches.	The program authorization protection identifier is incorrect. Check whether the authorization protection identifier of the device is set correctly.	Contact the equipment provider.
1504	User program empty	The user program is empty. There is no valid program.	Re-download the user program.

Fault Code	Message	Description	Troubleshooting
1505	Block POU identifier error	The block POU identifier is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1510	Subprogram identifier error	The subprogram identifier is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1511	Subprogram type error	The subprogram type is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1512	Subprogram serial number error or out of range	The subprogram serial number is incorrect or out of range.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1513	Incorrect, duplicate, or conflicting subprogram address	The subprogram address is incorrect, duplicated, or conflicting.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1514	Interrupt subprogram serial number error or out of range	The interrupt subprogram serial number is incorrect or out of range.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1515	Incorrect, duplicate, or conflicting interrupt subprogram address	The interrupt subprogram address is incorrect, duplicated, or conflicting.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1516	Interrupt subprogram edge error (not rising edge or falling edge)	The interrupt subprogram edge is incorrect (not rising edge or falling edge).	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1517	Interrupt timing duration range error in the interrupt subprogram timer	The interrupt timing duration range of the interrupt subprogram timer is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1520	OBprog program identifier error	The OBprog program identifier is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1521	OBprog program type error	The OBprog program type is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1522	OBprog program serial number error or out of range	The OBprog program serial number is incorrect or out of range.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1523	Incorrect, duplicate, or conflicting OBprog program address	The OBprog program address is incorrect, duplicated, or conflicting.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1524	OBprog program variable quantity error	The variable quantity of the OBprog program is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1525	OBprog program variable length error	The variable length of the OBprog program is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1526	OBprog program header data error	The header data of the OBprog program is incorrect.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
1530	CJ-LBL instruction LBL serial number error or out of range	The LBL serial number of the CJ-LBL instruction is incorrect or out of range.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.

Fault Code	Message	Description	Troubleshooting
1531	Incorrect, duplicate, or conflicting LBL address of CJ-LBL instruction	The LBL address of the CJ-LBL instruction is incorrect, duplicated, or conflicting.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5001	Exception in user program execution or instruction return value error, some instructions not executed	Execution of the user program is abnormal or the return value of the instruction is incorrect, and some instructions are not executed, causing program execution to end abnormally.	Check the logic of the user program for any exception in execution process or execution logic.
5010	CALL instruction subprogram serial number error or out of range	The subprogram serial number of the CALL instruction is incorrect or out of range.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5011	CALL instruction subprogram non-existent or not initialized	The subprogram of the CALL instruction does not exist or is not initialized.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5012	CALL instruction subprogram nesting levels out of range or less than or equal to 0	The number of subprogram nesting levels of the CALL instruction is out of range.	Modify the program logic to reduce the subprogram nesting levels.
5013	Relationship error returned by the subprogram of the CALL instruction	The subprogram of the CALL instruction returns a relationship error.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5014	Mismatch between subprogram call and subprogram return	Subprogram execution is abnormal. The subprogram call and subprogram return do not match.	Check whether the subprogram call and return are disordered due to the abnormal end of the user program.
5015	Interrupt subprogram undefined	The interrupt subprogram is undefined or does not exist.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5016	Interrupt queue full and interrupt lost in the interrupt subprogram timer	The interrupt queue of the interrupt subprogram timer is full and the interrupt is lost.	Modify the interrupt subprogram properties or logic, and reduce the number of interrupts as appropriate.
5020	FBFC program serial number error or out of range	The FBFC program serial number is incorrect or out of range.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5021	FBFC program non-existent or not initialized	The FBFC program does not exist or is not initialized.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5022	FBFC program variable non-existent or not initialized	The variable of the FBFC program does not exist or is not initialized.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5023	FBFC program nesting levels out of range or less than or equal to 0	The number of FBFC program nesting levels is out of range.	Modify the program logic to reduce the FBFC program nesting levels.
5024	Relationship error returned by FBFC program	The FBFC program returns a relationship error.	Check whether the FBFC special instruction is used in the wrong position, or recompile and download the user program.

Fault Code	Message	Description	Troubleshooting
5025	Mismatch between OBprog program call and program return	OBprog program execution is abnormal. The program call and program return do not match.	Check whether the program call and return are disordered due to the abnormal end of the user program.
5030	CJ-LBL instruction LBL serial number error or out of range	The LBL serial number of the CJ-LBL instruction is incorrect or out of range.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5031	CJ-LBL instruction LBL non-existent or not initialized	The LBL of the CJ-LBL instruction does not exist or is not initialized.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5032	FOR-NEXT instruction nesting levels out of range or less than or equal to 0	The number of nesting levels of the FOR-NEXT instruction is out of range.	Modify the program logic to reduce the FOR-NEXT instruction nesting levels.
5033	FOR-NEXT instruction loops out of range or less than or equal to 0	The number of FOR-NEXT instruction loops is out of range or less than or equal to 0.	Modify the program logic to change the number of FOR-NEXT instruction loops.
5034	FOR-NEXT instruction loops equal to 0	The number of FOR-NEXT instruction loops is 0.	Modify the program logic to change the number of FOR-NEXT instruction loops.
5035	FOR and NEXT not paired	The FOR and NEXT instructions are not paired.	Check whether the disorder is caused by abnormal stop of the user program.
5080	Array subscript access out of bounds	The array access subscript is greater than the maximum array subscript value, and the subscript value in use has been changed to the maximum array subscript value.	Double-click the fault code to go to the corresponding program position to modify the subscript value.
5081	Division-by-zero protection, divisor 0 replaced by 1	The division-by-zero protection is triggered and the divisor 0 is replaced by 1 automatically.	Double-click the fault code to go to the corresponding program position to modify the divisor.
5082	Long-time no response from program loop	The program loop has no response for a long time.	Double-click the fault code to go to the corresponding program position to modify the loop statement.
5083	Array subscript access out of bounds	The array access subscript is less than 0, and the subscript value in use has been changed to 0.	Double-click the fault code to go to the corresponding program position to modify the subscript value.
5084	Invalid data	The floating-point data is invalid.	Check whether the input values of functions such as LN, LOG, SQRT are legal.
5101	Instruction parameter variable address error, or variable non-existent	The address of the parameter variable of the instruction is incorrect, or the variable does not exist.	Check whether the address of the parameter variable of the instruction is normal and whether the variable exists.
5102	Instruction parameter variable size error, or variable non-existent or out of range	The size of the parameter variable of the instruction is incorrect. The variable does not exist or is out of range.	Check whether the data length of the parameter variable of the instruction is out of range.
5104	Instruction parameter sequence error or relationship error	The instruction parameter sequence or relationship is incorrect.	Check whether the parameter sequence or relationship of the instruction is correct.

Fault Code	Message	Description	Troubleshooting
5105	String data error or length error in string instruction	The character string data or length of the string instruction is incorrect.	Check whether the character string data of the string instruction is illegal.
5110	Pointer serial number error or out of range	The serial number of the Pointer is incorrect or out of range.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
5111	Pointer not initialized or not pointing to a valid data variable	The Pointer is not initialized or does not point to a valid data variable.	Check whether the Pointer is initialized and whether it points to a valid variable address.
5112	Variable pointed to by the Pointer non-existent or out of range	The variable pointed to by the Pointer does not exist or is out of range.	Check the variable address pointed to by the Pointer or initialize the Pointer again.
5113	Pointer offset out of range	The offset of the Pointer is out of range.	Check whether the offset of the Pointer is too large. If yes, reduce the offset.
5114	Variable pointed to by the Pointer execution result non-existent or out of range	The variable pointed to by the execution result of the Pointer does not exist or is out of range.	Check whether the variable address pointed to by the execution result of the Pointer exists and whether it is out of range.
5120	Counter instruction instantiation failed	Failed to instantiate the counter instruction.	Recompile and download the user program.
5121	Counter instruction comparand error or out of range	The comparand of the counter instruction is incorrect or out of range.	Check whether the comparand of the counter instruction is incorrect or out of range.
5130	Timer instruction instantiation failed	Failed to instantiate the timer instruction.	Recompile and download the user program.
5131	Timer instruction comparand error or out of range	The comparand of the timer instruction is incorrect or out of range.	Check whether the comparand of the timer instruction is incorrect or out of range.
5140	Number of SFC STL parallel branch/parallel recombination/selective branch/selective recombination lines out of range	The number of SFC STL parallel branch/parallel recombination/selective branch/selective recombination lines is out of range.	Ensure that the number of SFC STL parallel branch/ parallel recombination/selective branch/selective recombination lines is within the specified range.
5150	Function block instruction instantiation failed	Failed to instantiate the function block instruction.	Recompile and download the user program.
5160	Array subscript variable code error or non-existent	The subscript variable code of the array is incorrect or does not exist.	Recompile and download the user program.
5161	Array subscript variable data error or out of range	The subscript variable of the array is incorrect or out of range.	Modify the value of the subscript variable so that the array falls within the allowable range.
5600	SerialSR instruction instantiation failed	Failed to instantiate the SerialSR instruction.	Recompile and download the user program.
5601	SerialSR instruction port ID out of range	The port ID of the SerialSR instruction is out of range.	Modify the port ID of the SerialSR instruction.
5602	SerialSR instruction protocol error	The protocol of the SerialSR instruction is incorrect.	Set the free protocol for the serial port by using the software tool.

Fault Code	Message	Description	Troubleshooting
5603	SerialSR instruction port conflict	Multiple instructions call the SerialSR instruction at the same time, and the instruction that fails to preempt the port reports an error.	Modify the instruction scheduling timing to implement time division multiplexing.
5604	SerialSR instruction TX data length out of range or less than 0	The TX data length of the SerialSR instruction is out of range or less than 0.	Check whether the TX data length of the SerialSR instruction is out of range or less than 0.
5605	SerialSR instruction TX data buffer error	Failed to obtain the TX data buffer of the SerialSR instruction.	Enable this instruction again.
5606	SerialSR instruction RX data length out of range or less than 0	The RX data length of the SerialSR instruction is out of range or less than 0.	Check whether the RX data length of the SerialSR instruction is out of range or less than 0.
5607	SerialSR instruction RX data buffer error	Failed to obtain the RX data buffer of the SerialSR instruction.	Enable this instruction again.
6580	Invalid axis ID in the CANopen axis instruction	The axis ID specified in the CANopen axis instruction is invalid.	Modify the axis ID.
6701	Invalid memory address: element or variable non-existent	The memory address is invalid. The element or variable to access does not exist.	Modify the instruction parameter to use a valid element or variable.
6705	Invalid memory size: memory non-existent or out of range	The memory size is invalid. The number of elements or variables to access is too large or out of range.	Modify the instruction parameter to adjust the number of elements or variables.
6706	Improper data or data out of range	The instruction parameter is improper or out of the allowable range.	Refer to the instructions guide to modify the instruction parameter value.
6711	Invalid variable address: variable non-existent	The variable address is invalid. The element or variable to access does not exist.	Modify the instruction parameter to use a valid element or variable.
6712	Invalid variable size: variable out of range	The variable size is invalid. The number of elements or variables to access is too large or out of range.	Modify the instruction parameter to adjust the number of elements or variables.
6713	Invalid variable encoding	The variable encoding is invalid.	Recompile and download the user program, or recompile and download the user program after replacing the software tool.
CPU			
1011	FPGA initialization failed	FPGA initialization failed.	The device hardware is faulty. Replace the device and return the faulty device to the factory for repair.
1012	Interrupt initialization failed	Interrupt initialization failed.	The device hardware is faulty. Replace the device and return the faulty device to the factory for repair.
1013	Timer interrupt initialization failed	Failed to initialize the timer interrupt of the user program.	Restart the device and try again, or replace the device and return the faulty device to the factory for repair.

Fault Code	Message	Description	Troubleshooting
5200	Error in data retention upon power failure	An error occurs to data retention upon power failure.	Check whether the function of data retention upon power failure works properly.
5238	2038 problem imminence warning	The device will not work normally after 11:14:07 on January 19, 2038 (UTC+8).	Change the device time.
5250	Low RTC battery voltage	The battery voltage of the RTC clock is low. If the device is powered off at this time, the system time will be restored to the initial value.	Replace the battery of the RTC clock while keeping the device powered on.
5900	Network down: Ethernet IP address conflict	When the device connects to the network or starts running after stop, or when its IP address is modified, it detects whether its IP address is used by other devices in the current network. If yes, the device automatically shuts down the network to avoid conflict.	Change the device IP address.
Local I/O			
5300	Initialization failed	Initialization failed.	The device hardware is faulty. Replace the device and return the faulty device to the factory for repair.
5301	Invalid DI filter parameter configuration	The DI filter parameter configuration data is invalid.	Modify the DI filter parameter configuration data.
Extension Module			
5400	Failed to initialize extension module interface hardware	The hardware of the extension module interface is faulty, which causes the initialization to fail.	The device hardware is faulty. Replace the device and return the faulty device to the factory for repair.
5401	Failed to parse extension module configuration data	The configuration data of the extension module cannot be parsed correctly because its format does not meet requirements.	Clear the compilation information using AutoShop and recompile and download the program. If the problem persists, delete the module configurations and add modules and configurations again one by one.
5402	Failed to initialize extension module interface slot	The slot of the extension module interface is faulty, which causes the initialization to fail.	1. Check whether the extension module interface slot is short-circuited. If yes, eliminate the short circuit. 2. Check whether the installed module hardware works properly. If not, replace the module.
5403	Extension module not installed	The extension module is configured but not installed.	Install the extension module as required, or modify the configuration of the extension module.
5404	Module installed inconsistent with module configured	The module installed in the slot must be inconsistent with the configured module; otherwise, it cannot work properly.	Install the extension module as required and modify module configuration accordingly to ensure consistency.

Fault Code	Message	Description	Troubleshooting
5405	Extension module interface hardware exception	The extension module interface is abnormal.	1. Check whether the extension module interface slot is short-circuited. If yes, eliminate the short circuit. 2. Check whether the installed module hardware works properly. If not, replace the module.
5406	Extension module interface software error	The extension module interface software is abnormal.	1. Upgrade the PLC firmware. 2. If the problem persists after the firmware upgrade, replace the device and return the faulty device to the factory for repair.
5411	Module in the slot not powered	The module requires external power supply to function properly, but the external power supply is not on.	Connect the external power supply correctly according to the module specifications.
5412	Slot module hardware fault	The module has an internal fault and cannot work properly.	Replace the module and return the faulty module to the factory for repair.
5413	Slot module over-temperature	The module detected a high internal temperature that may lead to malfunction.	1. Do not install the module in an environment that does not meet the relevant temperature requirements. 2. Replace the module and return the faulty module to the factory for repair.
5419	Slot module channel input or output overflow	For the input channel, the input signal has exceeded the upper sampling threshold. Sampling cannot be performed properly, and there is a possibility that the input port may be burned. For the output channel, the output value of the corresponding channel has exceeded the set upper threshold, and signals cannot be output properly.	Input channel: Check the actual input signal value. If the signal input to this channel has exceeded the set sampling range under normal working conditions, modify the sampling range as appropriate. If the signal is abnormal, check the output device or instrument of the signal. Output channel: Check the set output value and ensure that the set output is within the set range. If the set range cannot meet requirements, modify it as appropriate.
5420	Slot module channel input or output underflow	For the input channel, the input signal has fallen below the lower sampling threshold, and sampling cannot be performed properly. For the output channel, the output value of the corresponding channel has fallen below the set lower threshold, and signals cannot be output properly.	Input channel: Check the actual input signal value. If the signal input to this channel has exceeded the set sampling range under normal working conditions, modify the sampling range as appropriate. If the signal is abnormal, check the output device or instrument of the signal. Output channel: Check the set output value and ensure that the set output is within the set range. If the set range cannot meet requirements, modify it as appropriate.

Fault Code	Message	Description	Troubleshooting
5421	Slot module channel input upper limit exceeded or current output disconnected	For the input channel, the input signal has exceeded the upper sampling threshold. At this time, the signal can be sampled normally but the accuracy cannot be guaranteed. For the current output channel, the output port is not connected to the load or the impedance of the connected load is too large, so that the current cannot be output normally.	Input channel: Check the actual input signal value. If the signal input to this channel has exceeded the set sampling range under normal working conditions, modify the sampling range as appropriate. If the signal is abnormal, check the output device or instrument of the signal. Current output channel: Ensure that the load of the output port is connected properly and reliably, and that the load impedance is within the range specified in the module specifications.
5422	Slot module channel input lower limit exceeded or voltage output short-circuited	For the input channel, the input signal has fallen below the lower sampling threshold. At this time, the signal can be sampled normally but the accuracy cannot be guaranteed. For the voltage output channel, the output port is possibly short-circuited or the impedance of the connected load is too small, so that the voltage cannot be output normally.	Input channel: Check the actual input signal value. If the signal input to this channel has exceeded the set sampling range under normal working conditions, modify the sampling range as appropriate. If the signal is abnormal, check the output device or instrument of the signal. Voltage output channel: Ensure that the load of the output port is connected properly and reliably, and that the load impedance is within the range specified in the module specifications.
5423	Slot module channel input disconnected or output hardware faulty	For the input channel, no input signal is connected to the input port or the input signal is too weak and cannot be detected or sampled. For the output channel, the channel hardware is faulty and may have burned out.	Input channel: Ensure that the signal of the input port is normal and valid and is connected properly and reliably. Output channel: Replace the module and return the faulty module to the factory for repair.
Local Encoder Axis			
6300	Input device not assigned or assigned input device invalid	The local encoder axis must be assigned with a high-speed counter, and each high-speed counter can only be assigned to one axis, otherwise the axis cannot work properly.	Assign a high-speed counter that has not been assigned yet in "Input Device" on the "Basic Settings" page of the axis.

Fault Code	Message	Description	Troubleshooting
6301	Axis unit conversion configuration invalid	After a high-speed counter is assigned to an axis, its count value (pulse unit) is converted into the equivalent in user unit (Unit) according to the unit conversion setting parameter. If the number of pulses per revolution of the encoder, the displacement of the encoder per revolution, or the gear ratio of the transmission device is set incorrectly, the axis cannot work properly.	Check the settings on the "Unit Conversion Settings" page of the axis and correct the parameter values.
6302	Axis software limit or revolution cycle configuration invalid	In linear mode, the negative limit must be less than 0, and the positive limit must be greater than 0. In rotary mode, the revolution cycle must be greater than 0. Since the high-speed counter is a 32-bit counter, the negative limit, positive limit, and revolution cycle must be 32-bit integers in the range of [–2147483648, +2147483647] after being converted into pulse units.	Linear mode: Modify the positive and negative limits to ensure that the negative limit is less than 0, the positive limit is greater than 0, and they are 32-bit integers in the range of [–2147483648, +2147483647] after being converted into pulse units. Rotary mode: Modify the revolution cycle to ensure that it is greater than 0 and is a 32-bit integer in the range of [–2147483648, +2147483647] after being converted into pulse units.
6303	Axis counting mode or signal source configuration invalid	The high-speed counter supports the following counting modes and signal sources: A/B phase frequency multiplication by 1: X0-A phase, X1-B phase, X2-A phase, X3-B phase A/B phase frequency multiplication by 2: X0-A phase, X1-B phase, X2-A phase, X3-B phase A/B phase frequency multiplication by 4: X0-A phase, X1-B phase, X2-A phase, X3-B phase CW/CCW: X0-CW, X1-CCW, X2-CW, X3-CCW Pulse +direction: X0-pulse, X1-direction, X2-pulse, X3-direction	Select a supported counting mode and signal source.
6304	Axis preset function: input terminal invalid	The preset function supports the input terminals X0, X1, X2, X3, X4, X5, X6, and X7.	Select an input terminal supported by the preset function.

Fault Code	Message	Description	Troubleshooting
6305	Axis probe 1: input terminal invalid	Probe 1 supports the input terminals X0, X1, X2, X3, X4, X5, X6, and X7.	Select an input terminal supported by probe 1.
6306	Axis probe 2: input terminal invalid	Probe 2 supports the input terminals X0, X1, X2, X3, X4, X5, X6, and X7.	Select an input terminal supported by probe 2.
6307	Axis comparison output: terminal invalid	The comparative output supports the output terminals Y0, Y1, Y2, and Y3.	Select an output terminal supported by the comparison output.
6308	Axis comparison output: pulse width invalid	When the unit is ms, the time range is 0.1 ms to 6553.5 ms. When the unit is Unit, the set value must fall between 1 and 65535 after being converted into pulse units.	Modify the pulse width to ensure that it is within the allowable range.
CANlink			
6400	Station address conflict: Station address already exists in the network.	In CANlink communication, the addresses of all stations connected to the network must be unique. Address conflict detection is performed after a device node is powered on and initialized or the station address is modified. If the address is duplicated, a fault is reported and all CANlink bus activities of the node are stopped.	Change the station address to ensure that there are no duplicate addresses in the network.
6401	Slave offline	Failed to communicate with the slave because it is offline.	Check whether the CAN network connection works properly. Ensure that the connection is reliable without short circuit or open circuit, CANH and CANL are not reversely connected, and the terminal resistance is normal.
6411	Slave configuration exception response (1) "Undefined encoding used"	During configuration of a slave, the slave returns exception response (1) "Undefined encoding used".	Check whether the type/model of the connected device is consistent with the configuration.
6412	Slave configuration exception response (2) "Configured index exceeds the maximum value supported by the node"	During configuration of a slave, the slave returns exception response (2) "Configured index exceeds the maximum value supported by the node".	Check whether the type/model of the connected device is consistent with the configuration.
6413	Slave configuration exception response (3) "Register address non-existent or inaccessible"	During configuration of a slave, the slave returns exception response (3) "Register address non-existent or inaccessible".	Check whether the type/model of the connected device is consistent with the configuration.

Fault Code	Message	Description	Troubleshooting
6415	Slave configuration exception response (5) "Register data length invalid"	During configuration of a slave, the slave returns exception response (5) "Register data length invalid".	Check whether the type/model of the connected device is consistent with the configuration.
6416	Waiting for slave configuration command response timed out	During configuration of a slave, waiting for slave response timed out.	Check whether the type/model of the connected device is consistent with the configuration.
6421	Slave synchronization exception response (1) "Illegal command code"	When a synchronization command is sent to a slave, the slave returns exception response (1) "Illegal command code".	Check whether the type/model of the connected device is consistent with the configuration.
6422	Slave synchronization exception response (2) "Register address non-existent or inaccessible"	When synchronization data is sent to a slave, the slave returns exception response (2) "Register address non-existent or inaccessible".	Check whether the type/model of the connected device is consistent with the configuration.
6423	Slave synchronization exception response (3) "Value beyond allowable range"	When synchronization data is sent to a slave, the slave returns exception response (3) "Value beyond allowable range".	1. Check whether the set value in the corresponding register address has exceeded the allowed range. 2. Check whether the type/model of the connected device is consistent with the configuration.
6424	Slave synchronization exception response (4) "Operation unreachable or not allowed in the current state"	When synchronization data is sent to a slave, the slave returns exception response (4) "Operation unreachable or not allowed in the current state".	Check whether the type/model of the connected device is consistent with the configuration.
6425	Slave synchronization exception response (5) "Data length invalid"	When synchronization data is sent to a slave, the slave returns exception response (5) "Data length invalid".	Check whether the type/model of the connected device is consistent with the configuration.
6426	Waiting for slave synchronization command response timed out	Waiting for slave response to a synchronization command timed out.	Check whether the type/model of the connected device is consistent with the configuration.
CANopen			
6401	Node offline	Failed to communicate with the node because it is offline.	Check whether the CAN network connection works properly. Ensure that the connection is reliable without short circuit or open circuit, CANH and CANL are not reversely connected, and the terminal resistance is normal.
Modbus Master			
5500	8-bit data required for Modbus-RTU serial port	The Modbus-RTU serial port only supports 8-bit data.	Use 8-bit data for Modbus-RTU serial port.

Fault Code	Message	Description	Troubleshooting
6001	Slave returned exception response (01) "Illegal function code"	The function code received in the query is not an allowable action for the server (or slave). This may be because the function code is only applicable to new devices, and is not implementable in the unit selected. It could also indicate that the server (or slave) is in the wrong state to process a request of this type, for example, because it is unconfigured and is being asked to return register values.	Check whether the server (or slave) supports the function code.
6002	Slave returned exception response (02) "Illegal data address"	The data address received in the query is not an allowable address for the server (or slave). More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, a request with the offset 96 and the length 4 will succeed, but a request with the offset 96 and the length 5 will result in exception code 02.	Check whether the corresponding function code of the server (or slave) supports all the addresses accessed by this configuration.
6003	Slave returned exception response (03) "Illegal data"	A value contained in the query data field is not an allowable value for server (or slave). This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect. It specifically does not mean that a data item submitted for storage in a register has a value outside the expectation of the application program, since the Modbus protocol is unaware of the significance of any particular value of any particular register.	Check whether the value is within the allowed range.
6004	Slave returned exception response (04) "Slave device fault"	An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.	Check whether slave is abnormal or faulty.

Fault Code	Message	Description	Troubleshooting
6128	Response station number and requested station number mismatch	After the master sends a request frame, the station number in the received response frame is inconsistent with that in the transmitted request frame.	Check whether the connected slave is a normal Modbus slave.
6129	Response function code and requested function code mismatch	After the master sends a request frame, the function code in the received response frame is inconsistent with that in the transmitted request frame.	Check whether the connected slave is a normal Modbus slave.
6130	Response data address and requested data address mismatch	After the master sends a request frame, the data address in the received response frame is inconsistent with that in the transmitted request frame.	Check whether the connected slave is a normal Modbus slave.
6131	Response data value and requested data value mismatch	After the master sends a request frame, the data value in the received response frame is inconsistent with that in the transmitted request frame.	Check whether the connected slave is a normal Modbus slave.
6240	Cache address mapping in configuration invalid	The cache address mapping in the configuration is invalid and the configuration cannot be executed correctly.	Modify the cache address mapping in the configuration to a valid variable or element address.
6255	Request timed out	After sending a request frame, if the master does not receive a response from the slave within the specified timeout period, it retries according to the set number of retries. When the retry attempts exceed the set number, the master considers the slave abnormal and reports a request timeout error.	1. Ensure that the communication network cable is connected reliably. 2. Ensure that the slave station number is consistent with the configured slave station number. 3. Modify the timeout period to ensure that the master can receive the response frame within the timeout period. 4. Check whether the connected slave is a normal Modbus slave.
Modbus-TCP Master			
6000	Configuration disconnected	The Modbus-TCP client fails to establish a TCP connection with the server.	1. Ensure that the communication network cable is connected reliably. 2. Check whether the slave IP address and port ID are consistent with the configuration. 3. If the client and server are connected through a network bridge, router, or gateway, make sure that the client and server gateways are set correctly.

Fault Code	Message	Description	Troubleshooting
6001	Slave returned exception response (01) "Illegal function code"	The function code received in the query is not an allowable action for the server (or slave). This may be because the function code is only applicable to new devices, and is not implementable in the unit selected. It could also indicate that the server (or slave) is in the wrong state to process a request of this type, for example, because it is unconfigured and is being asked to return register values.	Check whether the server (or slave) supports the function code.
6002	Slave returned exception response (02) "Illegal data address"	The data address received in the query is not an allowable address for the server (or slave). More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, a request with the offset 96 and the length 4 will succeed, but a request with the offset 96 and the length 5 will result in exception code 02.	Check whether the corresponding function code of the server (or slave) supports all the addresses accessed by this configuration.
6003	Slave returned exception response (03) "Illegal data"	A value contained in the query data field is not an allowable value for server (or slave). This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect. It specifically does not mean that a data item submitted for storage in a register has a value outside the expectation of the application program, since the Modbus protocol is unaware of the significance of any particular value of any particular register.	Check whether the value is within the allowed range.
6004	Slave returned exception response (04) "Slave device fault"	An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.	Check whether slave is abnormal or faulty.

Fault Code	Message	Description	Troubleshooting
6128	Response station number and requested station number mismatch	After the master sends a request frame, the station number in the received response frame is inconsistent with that in the transmitted request frame.	Check whether the connected slave is a normal Modbus slave.
6129	Response function code and requested function code mismatch	After the master sends a request frame, the function code in the received response frame is inconsistent with that in the transmitted request frame.	Check whether the connected slave is a normal Modbus slave.
6130	Response data address and requested data address mismatch	After the master sends a request frame, the data address in the received response frame is inconsistent with that in the transmitted request frame.	Check whether the connected slave is a normal Modbus slave.
6131	Response data value and requested data value mismatch	After the master sends a request frame, the data value in the received response frame is inconsistent with that in the transmitted request frame.	Check whether the connected slave is a normal Modbus slave.
6240	Cache address mapping in configuration invalid	The cache address mapping in the configuration is invalid and the configuration cannot be executed correctly.	Modify the cache address mapping in the configuration to a valid variable or element address.
6255	Request timed out	After sending a request frame, if the master does not receive a response from the slave within the specified timeout period, it retries according to the set number of retries. When the retry attempts exceed the set number, the master considers the slave abnormal and reports a request timeout error.	1. Ensure that the communication network cable is connected reliably. 2. Ensure that the slave station number is consistent with the configured slave station number. 3. Modify the timeout period to ensure that the master can receive the response frame within the timeout period. 4. Check whether the connected slave is a normal Modbus slave.
EtherNet/IP			
6600	EtherNet/IP connection instance: not connected	When the PLC is in the "Stopped" state, the connection instance is displayed as not connected.	Switch the PLC to the "Running" state.

Fault Code	Message	Description	Troubleshooting
6602	EtherNet/IP connection instance: connection failed, target device offline	1. The IP of the connection instance is inconsistent with the IP of the target device. 2. The target device is powered off or is in the "Stopped" state. 3. Network device failure.	1. Set the IP of the connection instance to the IP of the target device. 2. Switch the target device to the "Running" state. 3. Replace the switch or cable. In case of connection across network segments, configure a gateway.
6603	EtherNet/IP connection instance: connection failed, target device not responding	The target device is online, but the EtherNet/IP protocol of the target device is abnormal.	Power off and on the target device.
6604	EtherNet/IP connection instance: connection failed, target device returned an error response	The specified connection path does not exist, or the specified data size does not match.	Open the connection status page, check the connection status code and status description, and modify the connection path, data size, or other relevant parameters.
6605	EtherNet/IP connection instance: connection timeout	1. High CPU load (reaching 90%) of the PLC or target device can cause timeout in packet message reception. 2. Network device failure.	1. Increase the PLC scan cycle, increase the RPI cycle of the EtherNet/IP connection, and reduce the number of EtherNet/IP connections to reduce the CPU load. Use a high-performance PLC or target device. 2. Replace the switch or cable.
6607	EtherNet/IP configuration data format inconsistent	The software version does not match the board version.	Select AutoShop software and PLC firmware that are compatible with each other.
6631	EtherNet/IP display label: cannot connect to the path specified in the request	1. The IP of the parameter is inconsistent with the IP of the target device. 2. The target device is powered off or is in the "Stopped" state. 3. Network device failure.	1. Use the target device IP as the request parameter. 2. Switch the target device to the "Running" state. 3. Network troubleshooting: Replace the switch or cable. In case of connection across network segments, configure a gateway.
6632	EtherNet/IP display label: request timed out, no response received	1. High CPU load (reaching 90%) of the PLC or target device can cause timeout in packet message reception. 2. Network device failure.	1. Reduce the CPU load or use a high-performance PLC or target device. 2. Replace the switch or cable.
6633	EtherNet/IP display label: request succeeded, but error response received	1. The name of the specified label is inconsistent with that of the target label. 2. The element quantity of the specified label is inconsistent with that of the target label, and exceeds the limit.	1. Set the parameter according to the name of the target label. 2. Set the element quantity to a value less than or equal to the element quantity of the target label.
6634	EtherNet/IP display label: request succeeded, but data type inconsistent	The data type of the specified label is inconsistent with that of the target label.	Set the parameter according to the data type of the target label.

Fault Code	Message	Description	Troubleshooting
6635	EtherNet/IP display label: request succeeded, but data length inconsistent	The data packet returned by the target device is abnormal.	Power off and on the target device.
		EtherCAT	
8001	Failed to read master configuration	Failed to read the master configuration information.	Check whether the board software and software tool versions match.
8002	Failed to obtain slave configuration parameters	Failed to obtain slave configuration parameters.	Check whether the board software and software tool versions match.
8003	EtherCAT startup timed out	EtherCAT startup timed out.	1. Check whether the network is properly connected. 2. Check whether the connected slave is consistent with the configuration. 3. Check whether the slave type matches.
8004	Failed to request the master	Failed to request the master.	Restart the PLC.
8200	Failed to write slave startup parameters to SDO	Failed to write the slave startup parameters to the SDO.	1. Check whether there is an object dictionary that is not supported by the slave in the startup parameter list. 2. Check whether the value of the object dictionary is out of range.
8201	Slave lost during operation	The slave is lost during operation.	1. Check whether the network with the slave is disconnected. 2. Check whether the slave is powered off.
8202	Slave state machine switched to non-OP mode	The slave state machine is switched to non-OP mode.	1. Check whether the network with the slave is disconnected. 2. Check whether the slave is powered off.
8203	Slave state machine switching failed	Slave state machine switching failed.	-
8204	Slave type mismatch	The slave type is incorrect.	1. Check whether the network cable is reversely connected. 2. Check whether the connected device matches the configuration.
8205	PDO address error	The PDO address is incorrect.	1. Check whether the memory runs out. 2. Check whether the background and board software versions match. 3. Power off and restart the PLC.
8206	PDO length error	The PDO length is incorrect.	Check whether the background and board software versions match.
8301	Failed to switch to INIT state	Failed to switch to INIT state.	Check whether the slave station machine supports state transition.
8302	Failed to switch to PerOP state	Failed to switch to PerOP state.	Check whether the slave supports the CoE protocol.
8304	Failed to switch to SafeOP state	Failed to switch to SafeOP state.	Check whether the PDO communication configuration is correct.
8308	Failed to switch to OP state	Failed to switch to OP state.	1. Check the network communication quality. 2. Check whether the EtherCAT task cycle is appropriate.

Fault Code	Message	Description	Troubleshooting
8310	FMMU unit configuration error	An FMMU unit configuration error occurs.	Check whether the slave supports the FMMU unit.
8311	Email configuration error	An email configuration error occurs.	Check whether the slave supports the SM unit.
8400	ECTA configuration error	The ECTA configuration is incorrect.	Check whether the configured extension module is consistent with the actually connected extension module.
8401	ECTA hardware error	An ECTA hardware error occurs.	1. Check whether the connection between the ECTA and the extension module is loose. 2. Replace the ECTA.
8402	ECTA extension module error	An ECTA extension module error occurs.	1. Locate the extension module with the ERR indicator on. 2. Read the diagnosis object dictionary of the faulty module by using ETC_ReadParameter_CoE. 3. Determine the fault type based on description of the diagnosis object dictionary of the extension module in the ECTA application guide and eliminate the fault.
Motion Control Axis			
9001	Local axis emergency stop active	The emergency stop terminal input is active, and the pulse output is stopped.	Disable the emergency stop terminal input and then call the MC_Reset instruction to reset the fault.
9003	Overspeed	The pulse output frequency exceeds 200 kHz.	Check whether the pulse frequency obtained by multiplying the target velocity by the gear ratio exceeds 200 kHz.
9020	Homing error	The negative limit is not mapped.	Map the negative limit on the configuration interface.
9021	Homing error	The positive limit is not mapped.	Map the positive limit on the configuration interface.
9022	Homing error	The home signal is not mapped.	Map the home switch on the configuration interface.
9023	Homing error	1. The output frequency exceeds 200 kHz when the axis runs at the homing velocity. 2. The output frequency exceeds 200 kHz when the axis runs at the homing approach velocity.	1. Modify the unit conversion setting to ensure that the homing velocity and homing approach velocity do not exceed 200 kHz. 2. Change the homing velocity to ensure that the output frequency does not exceed 200 kHz. 3. Change the homing approach velocity to ensure that the output frequency does not exceed 200 kHz.
9024	Homing error	Homing timed out.	1. Check whether the limit signal and home signal can be connected normally. 2. Check whether the homing timeout time is too short.
9025	Homing error	The limit signal is incorrect during homing.	Check whether the limit signal that is not applicable to the current homing mode is triggered.
9030	Limiting active	The limit signal input is active during positioning.	Check whether the limit is reached during normal running.

Fault Code	Message	Description	Troubleshooting
9031	Synchronization error	The target number of transmitted pulses and the actual number of transmitted pulses do not match.	Check whether the limit is reached during normal positioning.
9101	Axis type error or non-existent	1. The type of the axis specified by AxisID is incorrect. 2. The axis specified by AxisID does not exist.	1. Check whether the instruction supports the axis specified by AxisID. 2. Check whether the axis specified by AxisID exists.
9102	Axis configuration failed	1. The axis configuration data is lost. 2. The axis configuration parameters are improper.	Check whether the parameters are correct.
9103	MC_Reset called when the axis is not faulty	The MC_Reset instruction is called when the axis is not faulty.	Check whether the MC_Reset instruction is called when the axis is not switched to ErrorStop state.
9104	Axis state unknown when MC_ReadStatus is called	The axis is in unknown state when the MC_ReadStatus instruction is called.	Check whether the current state of the axis is uncontrollable by using the online monitoring function.
9105	Current position setting not allowed	The MC_SetPosition instruction is called during running or stop.	Set the current position when the axis is in StandStill, Poweroff, or ErrorStop state.
9106	Stopping upon fault	The axis is stopping upon a fault.	Execute the instruction after stop upon fault is completed, the fault is resolved, and the reset instruction is executed.
9107	Improper parameter	The parameters are improper.	Check whether the parameters on the left of the instruction are set properly.
9108	Improper PLCOpen state machine	The PLCOpen state machine is improper.	Check whether the current PLCOpen state machine satisfies the execution conditions for this instruction. If not, call the relevant instruction to switch the axis to the required state.
9110	MC_Stop called repeatedly during stop	The MC_Stop instruction is called repeatedly during stop.	Trigger only one MC_Stop instruction at a time.
9111	Instruction linked list lost	The instruction linked list is lost.	1. Check whether the background version and board version match. 2. Contact the manufacturer.
9112	AxisID changed	The value of AxisID is changed while the instruction flow is active.	Do not change the axis number while the flow is active for Enable instructions such as MC_Power and MC_Jog.
9113	Reset by MC_Reset timed out	Reset by executing the MC_Reset instruction timed out.	1. Check whether the drive fault can be reset. 2. Check whether the axis fault type supports reset.
9114	Failed to write to 0x6060	The axis fails to write to 0x6060.	1. Check for interference in network communication. 2. Check whether the slave supports the object dictionary 0x6060.
9115	MC_Halt called when the axis is in Stopping state	The MC_Halt instruction is called when the axis is in Stopping state.	Do not call the MC_Halt instruction when the axis is in Stopping state.

Fault Code	Message	Description	Troubleshooting
9116	Axis in online commissioning mode	The current axis is in online commissioning mode.	Check whether the current axis is in online commissioning mode. PLC motion control instructions are invalid in online commissioning mode.
9118	Maximum acceleration (deceleration) exceeded	The acceleration (deceleration) of the instruction exceeds the maximum acceleration.	Check whether the acceleration (deceleration) of the instruction exceeds the maximum acceleration.
9119	MC_Jog target velocity exceeded maximum jogging velocity	The target velocity of the MC_Jog instruction exceeds the maximum jogging velocity.	Check whether the target velocity of the MC_Jog instruction exceeds the maximum jogging velocity.
9120	Target velocity exceeded maximum velocity	The target velocity exceeds the maximum velocity.	Check whether the target velocity of the instruction exceeds the maximum velocity.
9121	Jog forward and reverse motion signals both active	The forward and reverse motion signals of the jog instruction are both active.	Ensure that the forward and reverse motion signals of the jog instruction are not active at the same time.
9122	Control word not mapped to EtherCAT bus axis	The control word is not mapped to the EtherCAT bus axis.	Add the control word in the PDO and map it to the axis.
9123	Target position not mapped to EtherCAT bus axis	The target position is not mapped to the EtherCAT bus axis.	Add the target position in the PDO and map it to the axis.
9124	Target torque not mapped to EtherCAT bus axis	The target torque is not mapped to the EtherCAT bus axis.	Add the target torque in the PDO and map it to the axis.
9125	Status word not mapped to EtherCAT bus axis	The status word is not mapped to the EtherCAT bus axis.	Add the status word in the PDO and map it to the axis.
9126	Current position not mapped to EtherCAT bus axis	The current position is not mapped to the EtherCAT bus axis.	Add the feedback position in the PDO and map it to the axis.
9127	0x60fd not mapped to EtherCAT bus axis	0x60fd is not mapped to the EtherCAT bus axis.	Add 0x60fd in the PDO and map it to the axis.
9128	Current torque not mapped to EtherCAT bus axis	The current torque is not mapped to the EtherCAT bus axis.	Add the current torque in the PDO and map it to the axis.
9129	Probe control word not mapped to EtherCAT bus axis	The probe control word is not mapped to the EtherCAT bus axis.	Add the probe control word in the PDO and map it to the axis.
9130	Probe status word not mapped to EtherCAT bus axis	The probe status word is not mapped to the EtherCAT bus axis.	Add the probe status word in the PDO and map it to the axis.
9131	Probe position not mapped to EtherCAT bus axis	The probe position is not mapped to the EtherCAT bus axis.	Add the probe position in the PDO and map it to the axis.
9132	Probe channel occupied by interrupt positioning instruction	An interrupt positioning instruction is being executed and the probe channel is occupied.	The probe instruction and interrupt positioning instruction must not occupy the same probe channel at the same time. When the two instructions are called simultaneously in the program, the interrupt positioning instruction takes priority.
9133	Imaginary axis mode enabled	The imaginary axis mode is enabled.	The current instruction does not support the imaginary axis mode.

Fault Code	Message	Description	Troubleshooting
9134	Imaginary axis probe in use	The imaginary axis probe is being used.	The H5U supports two imaginary axis probes. Check whether the current probe is out of range.
9135	Interrupt signal not triggered in interrupt positioning	The interrupt signal is not triggered in the interrupt positioning instruction.	During execution of the interrupt positioning instruction, no interrupt signal is detected after positioning is completed.
9136	Probe channel occupied by another instruction during interrupt positioning	The probe channel is occupied by another instruction during the interrupt positioning process.	Ensure that the probe channel is not occupied during the interrupt positioning process.
9137	Control mode 0x6060 not mapped to bus driver	The control mode 0x6060 is not mapped to the bus driver.	Add 0x6060 in the PDO and map it to the axis.
9138	Control mode 0x6061 not mapped to bus driver	The control mode 0x6061 is not mapped to the bus driver.	Add 0x6061 in the PDO and map it to the axis.
9139	MC_Home called repeatedly during homing	The MC_Home instruction is called repeatedly during homing.	Do not call the MC_Home instruction repeatedly during homing.
9140	Target torque exceeded maximum value	The target torque of the instruction exceeds the maximum value.	Check whether the target torque of the instruction exceeds the positive and negative torque limits.
9141	Maximum velocity not mapped to bus driver	The maximum velocity is not mapped to the bus driver.	Add 0x607f in the PDO and map it to the axis.
9142	Immediate stop instruction active	The immediate stop instruction is active.	Check whether the immediate stop instruction has been called.
9143	Immediate stop instruction called repeatedly	The immediate stop instruction is called repeatedly.	Check whether the immediate stop instruction is called repeatedly.
9144	Limit reached during jogging	The limit is reached during jogging.	Check whether the limit is active.
9145	Target position exceeded 9999999	The precision is reduced if a single-precision floating-point number exceeds 9999999. Therefore, the target position must not exceed this value.	1. Check whether the target position is correct. Set the target position again. 2. Change the gear ratio to ensure that the target position is not greater than 9999999.
9146	Target velocity exceeded 9999999	The precision is reduced if a single-precision floating-point number exceeds 9999999. Therefore, the target velocity must not exceed this value.	1. Check whether the target velocity is correct. Set the target velocity again. 2. Change the gear ratio to ensure that the target velocity is not greater than 9999999.
9147	Target acceleration exceeded 9999999	The precision is reduced if a single-precision floating-point number exceeds 9999999. Therefore, the target acceleration must not exceed this value.	1. Check whether the target acceleration is correct. Set the target acceleration again. 2. Change the gear ratio to ensure that the target acceleration is not greater than 9999999.

Fault Code	Message	Description	Troubleshooting
9148	Target deceleration exceeded 9999999	The precision is reduced if a single-precision floating-point number exceeds 9999999. Therefore, the target deceleration must not exceed this value.	1. Check whether the target deceleration is correct. Set the target deceleration again. 2. Change the gear ratio to ensure that the target deceleration is not greater than 9999999.
9149	Axis in sync control mode, abortion not allowed	1. A single-axis motion instruction is called when the axis is performing interpolation in sync control mode. The single-axis motion instruction reports an error.	1. Do not call single-axis motion instructions during interpolation.
9150	MC_Halt in execution, abortion not allowed	The MC_MoveSuperImposer instruction is called while the MC_Halt instruction is still active.	Do not call the MC_MoveSuperImposer instruction while the MC_Halt instruction is still active.
9151	MC_MoveVelocityCSV PulseWidth out of range	The variable PulseWidth of the MC_MoveVelocityCSV instruction is out of range.	Ensure that the parameter value is within the allowable range.
9152	Object dictionary 60FFh not associated in I/O mapping of bus servo axis when MC_MoveVelocityCSV is called	The object dictionary 60FFh is not associated in I/O mapping of the bus servo axis when the MC_MoveVelocityCSV instruction is called.	Ensure that the object dictionary 60FFh is associated in I/O mapping of the bus servo axis when the MC_MoveVelocityCSV instruction is called.
9153	Probe terminal not configured	The probe terminal is not configured.	Check whether the software tool version supports configuration of the probe terminal ID.
9154	MC_SetAxisConfigPara ParameterIndex out of range	The value of ParameterIndex of the MC_SetAxisConfigPara instruction is out of range.	Ensure that the parameter value is within the allowable range.
9155	Instruction execution not allowed when axis configuration parameters are being modified	The configuration parameters of the axis are being modified, and execution of this instruction is not allowed before the modification is completed.	Perform the enable operation after axis initialization is completed.
9156	Multi-execution of MC_SetAxisConfigPara not allowed	MC_SetAxisConfigPara does not support multi-execution.	Note that this instruction does not support re-execution or multi-execution.
9157	Gear/cam motion instruction not supported by axis	The gear/cam motion instruction is not supported by the axis due to axis properties.	Ensure that the axis is not in single-axis mode or that the PLC supports the motion instruction.
9200	Failed to obtain cam table configuration file	Failed to obtain the cam table configuration file.	1. Check whether the board software and software tool match. 2. Re-download the cam configuration table.
9201	Failed to obtain master axis	Failed to obtain the master axis.	1. Check whether the master axis called in the program exists. 2. Check whether the master axis has reported an error.

Fault Code	Message	Description	Troubleshooting
9202	Failed to obtain slave axis	Failed to obtain the slave axis.	1. Check whether the slave axis called in the program exists. 2. Check whether the slave axis has reported an error.
9203	Failed to obtain cam table	Failed to obtain the cam table.	Check whether the cam table called exists.
9204	Number of cams executed simultaneously in the PLC program exceeded maximum value	The number of cams executed simultaneously in the PLC program exceeds the maximum allowable value.	Check whether the number of cams executed simultaneously in the program exceeds the threshold.
9205	No cam node found	The corresponding cam node is not found.	This instruction can be called only when the slave axis is in cam engagement state.
9206	Master axis changed during cam engagement	The master axis is changed during cam engagement.	Do not change the master axis during cam engagement.
9207	MC_CamIn StartMode out of range	StartMode of the MC_CamIn instruction is out of range.	Ensure that the parameter value is within the specified range.
9208	MC_CamIn StartPosition exceeded maximum value	StartPosition of the MC_CamIn instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9209	MC_CamIn MasterStartDistance exceeded maximum value	MasterStartDistance of the MC_CamIn instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9210	MC_CamIn MasterScaling exceeded maximum value	MasterScaling of the MC_CamIn instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9211	MC_CamIn SlaveScaling exceeded maximum value	SlaveScaling of the MC_CamIn instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9212	MC_CamIn MasterOffset exceeded maximum value	MasterOffset of the MC_CamIn instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9213	MC_CamIn SlaveOffset exceeded maximum value	SlaveOffset of the MC_CamIn instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9214	MC_CamIn MasterScaling not positive	MasterScaling of the MC_CamIn instruction is not a positive number.	Set this parameter to a positive number.
9215	MC_CamIn SlaveScaling not positive	SlaveScaling of the MC_CamIn instruction is not a positive number.	Set this parameter to a positive number.
9216	MC_CamIn/MC_GearIn ReferenceType out of range	ReferenceType of the MC_CamIn/MC_GearIn instruction is out of range.	Ensure that the parameter value is within the specified range.
9217	MC_CamIn Direction out of range	Direction of the MC_CamIn instruction is out of range.	Ensure that the parameter value is within the specified range.

Fault Code	Message	Description	Troubleshooting
9218	MC_CamIn BufferMode out of range	BufferMode of the MC_ CamIn instruction is out of range.	Ensure that the parameter value is within the specified range.
9219	Master axis phases in cam table node array not in ascending order	The master axis phases in the node array of the cam table are not sorted in ascending order.	Sort the master axis phases in ascending order when customizing cam table nodes.
9220	Curve type setting of cam table node array out of range	The curve type setting of the node array of the cam table is out of range.	Check whether the curve type of the cam node array is set incorrectly.
9221	MC_CamOut target deceleration exceeded maximum value	The target deceleration of the MC_CamOut instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9222	MC_CamOut target deceleration out of range	The target deceleration of the MC_CamOut instruction is out of range and causes stop upon a fault.	Ensure that the target deceleration is within the specified range.
9223	MC_Phasing target acceleration exceeded maximum value	The target acceleration of the MC_Phasing instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9224	MC_Phasing target acceleration out of range	The target acceleration of the MC_Phasing instruction is out of range.	Ensure that the target acceleration is within the specified range.
9225	MC_Phasing target velocity exceeded maximum value	The target velocity of the MC_Phasing instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9226	MC_Phasing target velocity out of range	The target velocity of the MC_Phasing instruction is out of range.	Ensure that the target deceleration is within the specified range.
9227	MC_CamOut curve type setting out of range	The curve type setting of the MC_CamOut instruction is out of range.	Ensure that the curve type setpoint in the instruction is within the specified range.
9228	MC_CamOut Mode out of range	The value of Mode of the MC_CamOut instruction is out of range.	Ensure that the value of Mode is within the specified range.
9229	Cam node array empty detected by MC_ GenerateCamTable	The MC_GenerateCamTable instruction detects that the cam node array is empty.	Contact Inovance for technical support.
9230	MC_GenerateCamTable node quantity input exceeded maximum value	The node quantity specified by the MC_ GenerateCamTable instruction exceeds the maximum allowable value.	Check whether the target node quantity specified in the instruction is beyond the specified range.
9231	MC_GenerateCamTable Mode out of range	The value of Mode of the MC_GenerateCamTable instruction is out of range.	Ensure that the parameter value is within the specified range.
9232	MC_GenerateCamTable node quantity input too small	The node quantity specified by the MC_ GenerateCamTable instruction is too small.	Ensure that the node quantity is 2 or more.

Fault Code	Message	Description	Troubleshooting
9233	MC_GearIn RatioNumerator equal to 0	The value of RatioNumerator of the MC_GearIn instruction is 0.	Set this parameter to a non-zero floating-point number.
9234	MC_GearIn RatioDenominator not greater than 0	The value of RatioDenominator of the MC_GearIn instruction is not greater than 0.	Set this parameter to a floating-point number greater than 0.
9235	MC_GenerateCamTable in execution when MC_SaveCamTable is called	The MC_GenerateCamTable instruction is being executed when the MC_SaveCamTable instruction is called.	Do not call the MC_SaveCamTable instruction before the cam table data update operation is completed.
9236	MC_SaveCamTable in execution when MC_GenerateCamTable is called	The MC_SaveCamTable instruction is being executed on the cam table when the MC_GenerateCamTable instruction is called.	Do not call the MC_GenerateCamTable instruction before the cam table is saved.
9237	Failed to open cam table during execution of MC_SaveCamTable	Failed to open the cam table file during execution of the MC_SaveCamTable instruction.	1. Check whether the PLC memory runs out. 2. Replace the PLC.
9238	Failed to write cam point quantity when saving the cam table	Failed to write the cam point quantity when the cam table is being saved.	1. Check whether the PLC memory runs out. 2. Replace the PLC.
9239	Failed to write data when saving cam table	Failed to write data when the cam table is being saved.	1. Check whether the PLC memory runs out. 2. Replace the PLC.
9240	Phase of the first point not 0	The phase of the first point is not 0.	Ensure that the phase of the first point is 0.
9241	Displacement of the first point not 0	The displacement of the first point is not 0.	Ensure that the displacement of the first point is 0.
9242	MC_GearOut Mode out of range	The value of Mode of the MC_GearOut instruction is out of range.	Ensure that the value of Mode is within the specified range.
9243	MC_Phasing target deceleration exceeded maximum value	The target deceleration of the MC_Phasing instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9244	MC_GearIn target deceleration exceeded maximum value	The target deceleration of the MC_GearIn instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9245	MC_CamIn Periodic out of range	The value of Periodic of the MC_CamIn instruction is out of range.	Ensure that the parameter value is within the specified range.
9246	Cam table phase exceeded maximum value	The phase in the cam table exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number does not exceed 9999999.
9247	Absolute value of cam table displacement exceeded maximum value	The absolute value of the displacement in the cam table exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number does not exceed 9999999.

Fault Code	Message	Description	Troubleshooting
9248	Absolute value of cam table link velocity exceeded maximum value	The absolute value of the link velocity in the cam table exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number does not exceed 9999999.
9249	Gear node empty	The gear node is empty.	Contact Inovance for technical support.
9250	Master axis same as slave axis	The master axis and slave axis are the same.	Do not use the same axis as both the master axis and slave axis of the cam gear.
9251	Master axis configuration address greater than or equal to slave axis address	The configuration address of the master axis is greater than or equal to that of the slave axis.	When ReferenceType is set to set position of the current cycle, ensure that the configuration address of the master axis is less than that of the slave axis.
9252	Master axis filter coefficient fFilter[0] corresponding to the slave axis out of range	The master axis filter coefficient fFilter[0] corresponding to the slave axis is out of range.	Ensure that the value of this variable is between 0 and 1 (0 and 1 included).
9253	Master axis filter coefficient fFilter[1] corresponding to the slave axis out of range	The master axis filter coefficient fFilter[1] corresponding to the slave axis is out of range.	Ensure that the value of this variable is between 0 and 1 (0 and 1 included).
9254	Master axis filter coefficient fFilter[2] corresponding to the slave axis out of range	The master axis filter coefficient fFilter[2] corresponding to the slave axis is out of range.	Ensure that the value of this variable is between 0 and 1 (0 and 1 included).
9255	Sum of master axis filter coefficients corresponding to the slave axis not 1	The sum of the master axis filter coefficients corresponding to the slave axis is not 1.	Ensure that the sum of the master axis filter coefficients corresponding to the slave axis is 1.
9256	Improper StartPosition and MasterStartDistance in MC_CamIn	The start position and start distance of the master axis in the MC_CamIn instruction are improper.	If the master axis works in linear mode and Direction in the instruction is set to positive, ensure that the cam synchronization point is not less than the cam engagement point.
9257	Improper StartPosition and MasterStartDistance in MC_CamIn	The start position and start distance of the master axis in the MC_CamIn instruction are improper.	If the master axis works in linear mode and Direction in the instruction is set to negative, ensure that the cam synchronization point is not greater than the cam engagement point.
9258	MC_GearOut target deceleration exceeded maximum value	The target deceleration of the MC_GearOut instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9259	MC_Phasing target deceleration out of range	The target deceleration of the MC_Phasing instruction is out of range and causes stop upon a fault.	Ensure that the target deceleration is within the specified range.
9260	MC_GearIn target deceleration out of range	The target deceleration of the MC_GearIn instruction is out of range and causes stop upon a fault.	Ensure that the target deceleration is within the specified range.
9261	MC_GearOut target deceleration out of range	The target deceleration of the MC_GearOut instruction is out of range and causes stop upon a fault.	Ensure that the target deceleration is within the specified range.

Fault Code	Message	Description	Troubleshooting
9262	MC_GearIn target acceleration exceeded maximum value	The target acceleration of the MC_GearIn instruction exceeds the maximum allowable value.	Ensure that the absolute value of the floating-point number in the motion control instruction does not exceed 9999999.
9263	MC_GearIn target acceleration out of range	The target acceleration of the MC_GearIn instruction is out of range.	Ensure that the target acceleration is within the specified range.
9264	MC_Phasing curve type setting out of range	The curve type setting of the MC_Phasing instruction is out of range.	Ensure that the curve type setpoint in the instruction is within the specified range.
9265	MC_GearIn curve type setting out of range	The curve type setting of the MC_GearIn instruction is out of range.	Ensure that the curve type setpoint in the instruction is within the specified range.
9266	MC_GearOut curve type setting out of range	The curve type setting of the MC_GearOut instruction is out of range.	Ensure that the curve type setpoint in the instruction is within the specified range.
9267	Slave axis changed during cam operation	The slave axis is modified during the cam operation.	Do not modify the slave axis during the cam operation.
9268	MC_Phasing PhasingMode out of range	The value of PhasingMode of the MC_Phasing instruction is out of range.	Ensure that the value of the parameter is within the specified range.
9269	Axis not in cam control mode when MC_CamOut is called	The current axis is not in cam control mode when the MC_CamOut instruction is called.	Ensure that the axis works in cam control mode when the MC_CamOut instruction is called.
9270	Axis not in gear control mode when MC_GearOut is called	The current axis is not in gear control mode when the MC_GearOut instruction is called.	Ensure that the axis works in gear control mode when the MC_GearOut instruction is called.
9271	Master axis position change too large within a single EtherCAT cycle during cam/gear operation	The position change of the master axis is too large within a single EtherCAT cycle during cam/gear operation.	Ensure that the position change of the master axis is not greater than half a cam cycle within a single EtherCAT cycle.
9272	MC_ GetCamTableDistance Phase out of range	The point specified by Phase in the MC_ GetCamTableDistance instruction does not fall between the start and end points.	Ensure that the point specified by Phase is within the specified curve.
9273	Slave axis changed during execution of MC_ GearIn	The slave axis is changed during execution of the MC_GearIn instruction.	Do not change the slave axis during execution of the MC_GearIn instruction.
9274	MC_DigitalCamSwitch Channel out of range	The value of Channel of the MC_DigitalCamSwitch instruction is out of range.	Ensure that the parameter value is within the specified range.
9275	No axis found	The axis is not found.	Ensure that the axis specified by Axis exists.
9276	Number of tappets allowed to be executed at the same time out of range	The number of tappets allowed to be executed at the same time is out of range.	Ensure that the number of tappets allowed to be executed at the same time is within the allowable range.

Fault Code	Message	Description	Troubleshooting
9277	MC_DigitalCamSwitch ReferenceType out of range	The value of ReferenceType of the MC_ DigitalCamSwitch instruction is out of range.	Ensure that the parameter value is within the specified range.
9278	MC_DigitalCamSwitch Number out of range	The value of Number of the MC_DigitalCamSwitch instruction is out of range.	Ensure that the parameter value is within the specified range.
9279	MC_DigitalCamSwitch Switches array empty	The Switches array of the MC_DigitalCamSwitch instruction is empty.	Check whether the length of the Switches array meets requirements.
9280	Tappet array fPosition out of range	The value of fPosition of the tappet array is out of range.	Ensure that the parameter value is within the specified range.
9281	Tappet array iMode out of range	The value of iMode of the tappet array is out of range.	Ensure that the parameter value is within the specified range.
9282	Tappet array iDirection out of range	The value of iDirection of the tappet array is out of range.	Ensure that the parameter value is within the specified range.
9283	Tappet array fParameter out of range	The value of fParameter of the tappet array is out of range.	Ensure that the parameter value is within the specified range.
9284	Time setting out of range in time mode	When the tappet comparison point is set to time mode, the time setting is out of range.	Ensure that the parameter value is within the specified range.
9285	Selected axis not under cam control when MC_ DigitalCamSwitch ReferenceType is set to 3	The selected axis is not under cam control when ReferenceType of the MC_ DigitalCamSwitch instruction is set to 3.	Call the MC_DigitalCamSwitch instruction after cam control takes effect.
9286	Axis communication interrupted during tappet execution	Axis communication is interrupted during tappet execution.	Ensure that axis communication is not interrupted during tappet execution.
9287	Duplicate comparison position start points	The comparison position start points are the same during tappet execution.	Ensure that the start points are not duplicate.
9288	Comparison position start point same as end point	The comparison position start and end point are the same during tappet execution.	Ensure that the start and end points are not duplicate.
9289	Selected tappet terminal in use	The selected tappet terminal is being used by another function.	Check whether the terminal is set as the pulse output axis.
9290	Failed to execute MC_ DigitalCamSwitch due to improper motion control axis state	The MC_DigitalCamSwitch instruction cannot be executed because the state of the motion control axis is improper.	Do not execute the MC_DigitalCamSwitch instruction in homing mode.
9291	MasterSyncPosition setting in MC_GearInPos out of range	The MasterSyncPosition setting in the MC_ GearInPos instruction is out of range.	Ensure that the parameter value is within the specified range.

Fault Code	Message	Description	Troubleshooting
9292	SlaveSyncPosition setting in MC_GearInPos out of range	The SlaveSyncPosition setting in the MC_GearInPos instruction is out of range.	Ensure that the parameter value is within the specified range.
9293	MasterStarDistance in MC_GearInPos out of range	The MasterStarDistance setting in the MC_GearInPos instruction is out of range.	Ensure that the parameter value is within the specified range.
9294	Velocity setting in MC_GearInPos over system limit	The Velocity setting in the MC_GearInPos instruction exceeds the system limit.	Ensure that the parameter value is within the specified range.
9295	Velocity setting in MC_GearInPos over setting limit	The Velocity setting in the MC_GearInPos instruction exceeds the setting limit.	Ensure that the parameter value is within the specified range.
9296	Acceleration setting in MC_GearInPos over system limit	The Acceleration setting in the MC_GearInPos instruction exceeds the system limit.	Ensure that the parameter value is within the specified range.
9297	Acceleration setting in MC_GearInPos over setting limit	The Acceleration setting in the MC_GearInPos instruction exceeds the setting limit.	Ensure that the parameter value is within the specified range.
9298	Deceleration setting in MC_GearInPos over system limit	The Deceleration setting in the MC_GearInPos instruction exceeds the system limit.	Ensure that the parameter value is within the specified range.
9299	Deceleration setting in MC_GearInPos over setting limit	The Deceleration setting in the MC_GearInPos instruction exceeds the setting limit.	Ensure that the parameter value is within the specified range.
9300	AvoidReversal in MC_GearInPos out of range	The AvoidReversal setting in the MC_GearInPos instruction is out of range.	Ensure that the parameter value is within the specified range.
9301	Zero master axis speed when MC_GearInPos instruction is started	The master axis speed is zero when the MC_GearInPos instruction is started.	Ensure that the master axis speed is not zero when starting this instruction.
9302	Zero master axis displacement in catching phase of MC_GearInPos	The master axis did not move during the catching phase of the MC_GearInPos instruction.	When MasterStarDistance is set to 0, ensure that the input MasterSyncPosition does not overlap with the current position of the master axis.
9303	Slave axis speed not zero before entering chasing phase after MC_GearInPos is started	When the MC_GearInPos instruction is started, the speed of the slave axis is not zero before entering the catching phase.	Ensure that the slave axis remains stationary before entering the catching phase.
9304	Failed to enter catching phase of MC_GearInPos	The MC_GearInPos instruction cannot enter the catching phase.	Ensure that the master axis can enter the catching phase under the current position and motion direction conditions.
9305	Slave axis over-speed during MC_GearInPos operation	During the operation of the MC_GearInPos instruction, the speed of the slave axis exceeds the limit.	Ensure that the parameter value is within the specified range.

Fault Code	Message	Description	Troubleshooting
9400	Maximum axis group quantity exceeded	The number of axis groups exceeds the maximum allowable value.	Reduce the number of axis groups in the project so that it does not exceed the maximum value.
9401	Faulty axis in axis group	An axis in the axis group is faulty.	Locate the faulty axis, view the fault codes of the axis, and rectify the fault.
9402	Number of buffered interpolation instructions exceeded 8	The number of buffered interpolation instructions is greater than 8.	Check whether the number of buffered interpolation instructions is greater than 8.
9403	Axis reused	An axis in the axis group is reused.	Each axis can be used in only one axis group. Check whether there is a reused axis in the axis group and replace it with an unused axis.
9404	Failed to create axis group	The x-axis or y-axis does not exist.	Check whether the x-axis and y-axis exist. An axis group consists of at least the x-axis and y-axis.
9405	Specified z-axis non-existent	The z-axis is specified in the instruction but does not exist in the configuration.	Check whether the z-axis specified in the instruction exists.
9406	Specified auxiliary axis non-existent	The auxiliary axis is specified in the instruction but does not exist in the configuration.	Check whether the auxiliary axis specified in the instruction exists.
9407	Axis group ID duplicated	The specified axis group ID has been used.	Change the axis group ID because the axis group ID must be unique.
9408	Axis configuration failed	Failed to configure the axis.	Check whether any axis in the axis group fails to be configured. If yes, check whether the board software and the background match.
9409	Axis ID less than 0	The axis ID is less than 0.	Check whether the ID of an axis in the axis group is less than 0.
9410	Axis group not released	The axis group is not released because the same MC_SetAxesGroup instruction is triggered repeatedly in a short time period.	Do not re-trigger the MC_SetAxesGroup instruction while its Busy signal output is still active.
9411	MC_GroupStop aborted	The MC_GroupStop instruction is aborted.	Check whether an instruction with higher priority is called while the MC_GroupStop instruction is still active.
9412	Circular interpolation instruction CircAxes out of range	The value of CircAxes of the circular interpolation instruction is out of range.	Check whether the value of CircAxes of the circular interpolation instruction is out of range.
9413	Circular interpolation instruction CircMode out of range	The value of CircMode of the circular interpolation instruction is out of range.	Check whether the value of CircMode of the circular interpolation instruction is out of range.
9414	Circular interpolation instruction PathChoice out of range	The value of PathChoice of the circular interpolation instruction is out of range.	Check whether the value of PathChoice of the circular interpolation instruction is out of range.
9415	Stop instruction StopMode out of range	The value of StopMode of the stop instruction is out of range.	Check whether the value of StopMode of the stop instruction is out of range.
9416	X-axis set to ring mode	The x-axis is set to ring mode.	Do not set the motion control axis to the ring mode in an interpolation instruction.
9417	Y-axis set to ring mode	The y-axis is set to ring mode.	Do not set the motion control axis to the ring mode in an interpolation instruction.

Fault Code	Message	Description	Troubleshooting
9418	Z-axis set to ring mode	The z-axis is set to ring mode.	Do not set the motion control axis to the ring mode in an interpolation instruction.
9419	Auxiliary axis set to ring mode	The auxiliary axis is set to ring mode.	Do not set the motion control axis to the ring mode in an interpolation instruction.
9420	Circular interpolation instruction triggered repeatedly	The circular interpolation instruction is triggered repeatedly.	Do not re-trigger the same circular interpolation instruction while its Busy signal output is still active.
9421	Linear interpolation instruction triggered repeatedly	The linear interpolation instruction is triggered repeatedly.	Do not re-trigger the same linear interpolation instruction while its Busy signal output is still active.
9422	Failed to obtain the axis group	Failed to obtain the axis group.	Check whether the axis group specified by GroupID has been created by calling MC_SetAxesGroup.
9423	Axis configuration failed	Failed to configure the axis.	Check whether an instruction is triggered when axis configuration is not completed. Check whether the communication state of all axes in the axis group is Axis ready.
9424	Axis disabled	An axis is disabled.	Do not call the interpolation instruction when any axis is in Disabled state.
9425	Axis in execution of single-axis motion instruction	The interpolation instruction is triggered when an axis is executing a single-axis motion instruction.	Do not call the interpolation instruction when any axis is executing single-axis motion instructions and not in StandStill state.
9426	Axis in Stopping state	An axis is in Stopping state.	Do not call the interpolation instruction when any axis is in Stopping state after executing the MC_Stop instruction.
9427	Axis group in Stopping state	The axis group is in Stopping state.	Do not call the interpolation instruction while the MC_GroupStop instruction is still active.
9428	Axis in Homing state	An axis is in Homing state.	Do not call the interpolation instruction when any axis is in Homing state after executing the MC_Home instruction.
9429	Axis in execution of the position setting instruction	An axis is executing the position setting instruction.	Do not call the interpolation instruction when any axis is setting the current position by executing the MC_SetPosition instruction.
9430	Axis in commissioning state	An axis is in commissioning state.	Do not call the interpolation instruction when any axis is in commissioning state.
9431	Axis in commissioning state during interpolation, aborted instruction execution of other axes	An axis enters the commissioning state during interpolation, which aborts instruction execution of other axes.	Check whether any axis enters the commissioning state during interpolation and aborts instruction execution of other axes.
9432	Failed to request memory	Failed to request the memory.	Check whether the memory runs out. Contact the manufacturer.
9433	Target velocity less than or equal to 0	The target velocity is 0 or less than 0.	Ensure that the target velocity of the instruction is greater than 0.
9434	Target acceleration less than or equal to 0	The target acceleration is 0 or less than 0.	Ensure that the target acceleration of the instruction is greater than 0.
9435	Target deceleration less than or equal to 0	The target deceleration is 0 or less than 0.	Ensure that the target deceleration of the instruction is greater than 0.
9436	Curve type setting out of range	The curve type setting is out of range.	Check whether the curve type is set to a value other than the T-shaped curve for the interpolation instruction.

Fault Code	Message	Description	Troubleshooting
9437	Improper AbsRelMode	AbsRelMode is set incorrectly.	Check whether the parameter is set to a value other than the absolute positioning and relative positioning modes.
9438	Improper BufferMode	BufferMode is set incorrectly.	Check whether the value of BufferMode is proper.
9439	Improper InsertMode	InsertMode is set incorrectly.	Check whether the value of InsertMode is proper.
9440	Axis stopped due to a fault	An axis stops due to a fault.	Locate the faulty axis and rectify the fault based on the fault code.
9441	MC_GroupStop called repeatedly	The MC_GroupStop instruction is called repeatedly.	Do not re-trigger an MC_GroupStop instruction or call other MC_GroupStop instructions while an MC_GroupStop instruction is still active.
9442	Data buffer area not empty	The data buffer area is not empty. It is an internal fault.	Contact the manufacturer.
9443	Not a circle	No circle can be drawn due to improper parameter settings.	Update the parameter settings.
9444	Not a circle	The start, end, and border points in the circular interpolation instruction are the same point, and no circle can be drawn.	Check the input parameters of the circular interpolation instruction and ensure that the start, end, and border points can form a circle.
9445	Instruction buffer area full	The instruction buffer area is full.	Contact Inovance for technical support.
9446	X-axis exceeded maximum velocity	The velocity of the x-axis exceeds the maximum allowable velocity.	Ensure that the target velocity of the x-axis is not greater than the maximum allowable velocity.
9447	Y-axis exceeded maximum velocity	The velocity of the y-axis exceeds the maximum allowable velocity.	Ensure that the target velocity of the y-axis is not greater than the maximum allowable velocity.
9448	Z-axis exceeded maximum velocity	The velocity of the z-axis exceeds the maximum allowable velocity.	Ensure that the target velocity of the z-axis is not greater than the maximum allowable velocity.
9449	Auxiliary axis exceeded maximum velocity	The velocity of the auxiliary axis exceeds the maximum allowable velocity.	Ensure that the target velocity of the auxiliary axis is not greater than the maximum allowable velocity.
9450	Failed to obtain the number of axis groups	Failed to obtain the number of axis groups.	Update the software tool to the latest version.
9451	Internal fault	An internal fault occurs.	Contact the manufacturer.
9452	Instruction called when the axis is in StandStill state	The instruction is called when the axis is in StandStill state.	Do not call this instruction when the axis is StandStill state.
9453	Maximum velocity exceeded	The maximum velocity specified on the axis group configuration interface is exceeded.	Check whether the target velocity of the instruction is greater than the maximum velocity specified on the axis group configuration interface.
9454	Maximum acceleration (deceleration) exceeded	The maximum allowable acceleration (deceleration) is exceeded.	Check whether the target acceleration (deceleration) of the instruction is greater than the maximum acceleration (deceleration) specified on the axis group configuration interface.

Fault Code	Message	Description	Troubleshooting
9455	Axis group fault due to linear interpolation instruction error	The axis group becomes faulty due to an error reported by the linear interpolation instruction.	Identify the first linear interpolation instruction that reports the error and troubleshoot the fault based on the fault code.
9456	Axis group fault due to circular interpolation instruction error	The axis group becomes faulty due to an error reported by the circular interpolation instruction.	Identify the first circular interpolation instruction that reports the error and troubleshoot the fault based on the fault code.
9457	Axis group fault due to axis group stop instruction error	The axis group becomes faulty due to an error reported by the axis group stop instruction.	Identify the first axis group stop instruction that reports the error and troubleshoot the fault based on the fault code.
9458	Axis group fault due to axis group pause instruction error	The axis group becomes faulty due to an error reported by the axis group pause instruction.	Identify the first axis group pause instruction that reports the error and troubleshoot the fault based on the fault code.
9459	X-axis performing the interpolation algorithm of another axis group	The x-axis in the axis group is performing the interpolation algorithm of another axis group.	An axis can be configured in different axis groups at the same time. However, ensure that it executes the interpolation instruction of only one axis group at the same time.
9460	Y-axis performing the interpolation algorithm of another axis group	The y-axis in the axis group is performing the interpolation algorithm of another axis group.	An axis can be configured in different axis groups at the same time. However, ensure that it executes the interpolation instruction of only one axis group at the same time.
9461	Z-axis performing the interpolation algorithm of another axis group	The z-axis in the axis group is performing the interpolation algorithm of another axis group.	An axis can be configured in different axis groups at the same time. However, ensure that it executes the interpolation instruction of only one axis group at the same time.
9462	Auxiliary axis performing the interpolation algorithm of another axis group	The auxiliary axis in the axis group is performing the interpolation algorithm of another axis group.	An axis can be configured in different axis groups at the same time. However, ensure that it executes the interpolation instruction of only one axis group at the same time.
9463	Axes in synchronous mode but not under axis group control when the MC_GroupStop instruction is called	When the MC_GroupStop instruction is called, the axes are in synchronous mode but not under axis group control, such as interpolation control or cam control.	Note that the MC_GroupStop instruction can be called only when the axes in the axis group are in synchronous mode under axis group control. Do not call the MC_GroupStop instruction when the axes enter the synchronous mode due to other instructions.
9464	Axes in synchronous mode but not under axis group control when the linear or circular interpolation instruction is called	When the linear or circular interpolation instruction is called, the axes are in synchronous mode but not under axis group control, such as interpolation control or cam control.	Note that the linear or circular interpolation instruction can be called only when the axes in the axis group are in synchronous mode under axis group control. Do not call the linear or circular interpolation instruction when the axes enter the synchronous mode due to other non-axis-group instructions.
9465	Axes in synchronous mode but not under axis group control when the MC_GroupHalt instruction is called	When the MC_GroupHalt instruction is called, the axes are in synchronous mode but not under axis group control, such as interpolation control or cam control.	Note that the MC_GroupHalt instruction can be called only when the axes in the axis group are in synchronous mode under axis group control. Do not call the MC_GroupHalt instruction when the axes enter the synchronous mode due to other instructions.

Fault Code	Message	Description	Troubleshooting
9466	Unreasonable NumOfTurns in MC_ MoveEllipse	The NumOfTurns parameter in the MC_ MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9467	Unreasonable AddLength in MC_ MoveEllipse	The AddLength parameter in the MC_MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9468	Shutdown due to MC_ MoveEllipse failure	MC_MoveEllipse instruction fails and causes shutdown.	Find the MC_MoveEllipse instruction that caused the failure and check the fault code of the instruction to further confirm the fault.
9469	Unreasonable CircAxes in MC_MoveEllipse	The CircAxes parameter in the MC_MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9470	Unreasonable CircMode in MC_MoveEllipse	The CircMode parameter in the MC_MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9471	Unreasonable PathChoice in MC_ MoveEllipse	The PathChoice parameter in the MC_MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9472	Unreasonable Velocity in MC_MoveEllipse	The Velocity parameter in the MC_MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9473	Unreasonable Acceleration in MC_ MoveEllipse	The Acceleration parameter in the MC_MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9474	Unreasonable Deceleration in MC_ MoveEllipse	The Deceleration parameter in the MC_ MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9475	Unreasonable BufferMode in MC_ MoveEllipse	The BufferMode parameter in the MC_MoveEllipse instruction is set unreasonably.	Ensure that the parameter value is within the allowable range.
9476	Cannot form ellipse due to unreasonable center point, long axis length, and short axis length	The set center point, long axis length, and short axis length are unreasonable and cannot form an ellipse.	Ensure that the parameter value is within the allowable range.
9477	Interpolation not supported by X-axis	The property of the X-axis in the axis group instruction does not support interpolation motion.	Ensure that the X-axis is not in single-axis mode.
9478	Interpolation not supported by Y-axis	The property of the Y-axis in the axis group instruction does not support interpolation motion.	Ensure that the Y-axis is not in single-axis mode.

Fault Code	Message	Description	Troubleshooting
9479	Interpolation not supported by Z-axis	The property of the Z-axis in the axis group instruction does not support interpolation motion.	Ensure that the Z-axis is not in single-axis mode.
9480	Interpolation not supported by auxiliary axis	The property of the auxiliary axis in the axis group instruction does not support interpolation motion.	Ensure that the auxiliary axis is not in single-axis mode.
9501	EtherCAT bus drive error	A drive error occurs. The fault code in the object dictionary 0x603F of the drive is 0x%x{16:16}.	1. Determine the drive fault type according to the bus drive guide and rectify the fault.
9502	Drive disabled	The drive is disabled.	1. Check whether the drive status word 0x6041 switches to the disabled state during motion. 2. Check whether communication is disconnected during motion.
9503	Limit reached	The limit is reached.	1. Check whether the software limit is configured and reached. 2. Check whether the hardware limit is reached.
9505	Failed to modify the control mode	Failed to modify the control mode.	1. Check for interference in network communication. 2. Check whether the drive supports the object dictionary 0x6060.
9508	Homing failed	Homing failed.	1. Identify the cause of the drive homing failure based on the fault code. 2. Check whether homing timed out.
9509	Axis internal calculation precision error	An axis internal calculation precision error occurs.	Check whether the floating-point data of the instruction falls beyond the single-precision floating-point number range.
9510	Following error out of range	The following error is out of range.	1. Check whether the acceleration is too large. 2. Check whether the set following error is too small.
9512	Servo drive disconnected during operation	The servo drive is disconnected during operation.	1. Check whether the drive works properly. 2. Check whether the network cable is properly connected. 3. Check for strong interference in communication.
9513	Homing failed due to a drive fault	Homing failed due to a drive fault.	Check the fault code of the drive to eliminate the fault.
9514	Homing failed because the homing offset exceeded 32 bits	Homing failed because the homing offset exceeded 32 bits.	Check whether the homing offset multiplied by the gear ratio exceeds 32 bits; if yes, change the gear ratio.
9515	Homing failed due to loss of the slave	Homing failed because the EtherCAT drive is lost.	Contact Inovance for technical support.
9516	Homing failed because the SDO failed to write to object dictionary 0x607C	Homing failed because the SDO failed to write to object dictionary 0x607C.	1. Check whether the drive supports 0x607C. 2. Check the network communication quality.
9517	Homing failed because the SDO failed to write 6 to object dictionary 0x6060	Homing failed because the SDO failed to write 6 to object dictionary 0x6060.	1. Set 0x6060 in the PDO. 2. Check the network communication quality.

Fault Code	Message	Description	Troubleshooting
9518	Homing failed because the SDO failed to read object dictionary 0x6061	Homing failed because the SDO failed to read object dictionary 0x6061.	1. Set 0x6061 in the PDO. 2. Check the network communication quality.
9519	Homing failed because the SDO failed to write 8 into object dictionary 0x6060	Homing failed because the SDO failed to write 8 into object dictionary 0x6060.	1. Set 0x6060 in the PDO. 2. Check the network communication quality.
9551	Failed to switch the control mode	Failed to switch the control mode.	Check for interference in network communication.
9552	Target velocity equal to 0	The target velocity is 0.	Check whether the target velocity of position instructions is appropriate.
9601	Axis stopped due to MC_ MoveAbsolute parameter exception	The axis stops due to parameter exception of the MC_MoveAbsolute instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9602	Axis stopped due to MC_ MoveRelative parameter exception	The axis stops due to parameter exception of the MC_MoveRelative instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9603	Axis stopped due to MC_ MoveVelocity exception	The axis stops due to exception of the MC_ MoveVelocity instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9604	Axis stopped due to MC_ Jog exception	The axis stops due to exception of the MC_Jog instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9605	Axis stopped due to MC_ MoveVelocityCSV exception	The axis stops due to exception of the MC_ MoveVelocityCSV instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9606	Axis stopped due to MC_ MoveBuffer exception	The axis stops due to exception of the MC_ MoveBuffer instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9607	Axis stopped due to MC_ MoveFeed parameter exception	The axis stops due to parameter exception of the MC_MoveFeed instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9608	Axis stopped due to MC_ Stop parameter exception	The axis stops due to parameter exception of the MC_Stop instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9609	Axis stopped due to MC_ MoveTorque parameter exception	The axis stops due to parameter exception of the MC_MoveTorque instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9610	Axis stopped due to MC_ Halt parameter exception	The axis stops due to parameter exception of the MC_Halt instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9611	Axis stopped due to MC_ MoveSuperImposed parameter exception	The axis stops due to parameter exception of the MC_MoveSuperImposed instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9612	Axis stopped due to MC_ SyncMoveVelocity error	The axis stops due to an error reported by the MC_ SyncMoveVelocity instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.

Fault Code	Message	Description	Troubleshooting
9613	Axis stopped due to MC_ SyncTorqueControl error	The axis stops due to an error reported by the MC_ SyncTorqueControl instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9614	Axis stopped due to MC_ FollowVelocity error	The axis stops due to an error reported by the MC_ FollowVelocity instruction.	Check the instruction that reports the error, and further determine the fault based on the fault code of the instruction.
9701	Failed to request memory for the encoder axis instruction	The encoder axis instruction failed to request the memory.	1. Check whether the PLC memory runs out. 2. Contact the manufacturer.
9702	1. Encoder axis type error 2. Requested encoder axis non-existent 3. Instruction not supported in offline commissioning	1. The encoder axis type is incorrect. 2. The requested encoder axis does not exist. 3. The instruction is not supported in offline commissioning.	1. This instruction does not support the set axis type. Check whether the axis type setting is incorrect. 2. The instruction is not supported in offline commissioning.
9703	Axis configuration failed	Failed to configure the axis.	Check whether the board software and the software tool match.
9704	Counter operation command not configured in I/O mapping of encoder axis	Counter operation command is not configured in I/O mapping of the encoder axis.	Configure Counter operation command in I/O mapping of the encoder axis.
9705	Counter status not configured in I/O mapping of encoder axis	Counter status is not configured in I/O mapping of the encoder axis.	Configure Counter status in I/O mapping of the encoder axis.
9706	Encoder present position not configured in I/O mapping of encoder axis	Encoder present position is not configured in I/O mapping of the encoder axis.	Configure Encoder present position in I/O mapping of the encoder axis.
9707	Pulse rate not configured in I/O mapping of encoder axis	Pulse rate is not configured in I/O mapping of the encoder axis.	Configure Pulse rate in I/O mapping of the encoder axis.
9708	Positive limit not greater than negative limit	The positive limit of the encoder axis is not greater than the negative limit.	Ensure that the positive limit of the encoder axis is greater than the negative limit.
9709	Positive limit greater than 2147483647 after being converted into the pulse unit	The positive limit of the encoder axis is greater than 2147483647 after being converted into the pulse unit.	Ensure that the positive limit of the encoder axis is less than or equal to 2147483647 after being converted into the pulse unit.
9710	Negative limit less than –2147483648 after being converted into the pulse unit	The negative limit of the encoder axis is less than –2147483648 after being converted into the pulse unit.	Ensure that the negative limit of the encoder axis is greater than or equal to –2147483648 after being converted into the pulse unit.
9711	revolution cycle in ring mode greater than 2147483647 after being converted into the pulse unit	The revolution cycle of the encoder axis in ring mode is greater than 2147483647 after being converted into the pulse unit.	Ensure that the revolution cycle of the encoder axis in ring mode is less than or equal to 2147483647 after being converted into the pulse unit.

Fault Code	Message	Description	Troubleshooting
9712	Encoder axis changed while ENC_Counter is active	The encoder axis is changed while the ENC_ Counter instruction is still active.	Do not change the encoder axis while the ENC_ Counter instruction is still active.
9713	GR10-2HCE module faulty	The GR10-2HCE module is faulty.	Check the fault code object dictionary of the GR10-2HCE module and troubleshoot the fault according to the fault code.
9714	Failed to reset the encoder axis fault	Failed to reset the encoder axis fault.	1. The current fault of the encoder axis does not support reset. 2. The encoder shaft enters the faulty state immediately after the fault is reset. Check the axis fault codes and slave fault codes to further determine the fault type.
9715	ENC_Reset called when the encoder axis is not faulty	The ENC_Reset instruction is called when the encoder axis is not faulty.	Do not call the ENC_Reset instruction when the encoder axis is not faulty.
9716	ENC_Preset TriggerMode out of range	The value of TriggerMode of the ENC_Preset instruction is out of range.	Ensure that the parameter value is within the allowable range.
9717	ENC_Preset Position greater than 9999999	The value of Position of the ENC_Preset instruction is greater than 9999999.	Set Position of the ENC_Preset instruction to a value less than or equal to 9999999.
9718	Physical output command not configured in I/O mapping of encoder axis	Physical output command is not configured in I/O mapping of the encoder axis.	Configure Physical output command in I/O mapping of the encoder axis.
9719	Preset position or comparison output position greater than positive limit	The preset position or comparison output position of the encoder axis instruction is greater than the positive limit.	Ensure that the preset position or comparison output position of the encoder axis instruction is less than or equal to the positive limit.
9720	Preset position or comparison output position less than negative limit	The preset position or comparison output position of the encoder axis instruction is less than the negative limit.	Ensure that the preset position or comparison output position of the encoder axis instruction is greater than or equal to the negative limit.
9721	Preset position or comparison output position greater than 2147483647 or less than –2147483648 after being converted into the pulse unit	The preset position or comparison output position of the encoder axis instruction is greater than 2147483647 or less than –2147483648 after being converted into the pulse unit.	Ensure that the preset position or comparison output position of the encoder axis instruction is between –2147483648 and +2147483647 after being converted into the pulse unit.
9722	Preset position or comparison output position greater than or equal to revolution cycle in ring mode	The preset position or comparison output position of the encoder axis instruction is greater than or equal to the revolution cycle in ring mode.	Ensure that the preset position or comparison output position of the encoder axis instruction is less than the revolution cycle in ring mode.
9723	ENC_TouchProbe ProbeID out of range	The value of ProbeID of the ENC_TouchProbe instruction is out of range.	Ensure that the parameter value is within the allowable range.

Fault Code	Message	Description	Troubleshooting
9724	ENC_TouchProbe TriggerEdge out of range	The value of TriggerEdge of the ENC_TouchProbe instruction is out of range.	Ensure that the parameter value is within the allowable range.
9725	ENC_TouchProbe TerminalSource out of range	The value of TerminalSource of the ENC_TouchProbe instruction is out of range.	Ensure that the parameter value is within the allowable range.
9726	ENC_TouchProbe TriggerMode out of range	The value of TriggerMode of the ENC_TouchProbe instruction is out of range.	Ensure that the parameter value is within the allowable range.
9727	Probe status word not associated in I/O mapping of the encoder axis	The probe status word is not associated in I/O mapping of the encoder axis.	Ensure that the probe status word is associated in I/O mapping of the encoder axis.
9728	Probe feedback position not associated in I/O mapping of the encoder axis	The probe feedback position is not associated in I/O mapping of the encoder axis.	Ensure that the probe feedback position is associated in I/O mapping of the encoder axis.
9729	Control word not associated in I/O mapping of the encoder axis	The control word is not associated in I/O mapping of the encoder axis.	Ensure that the control word is associated in I/O mapping of the encoder axis.
9730	Window start position not less than end position	The probe window function of the encoder axis is enabled, but the start position of the window is not less than the end position.	Ensure that the start position of the probe window is less than the end position.
9731	Xn0 not assigned with probe function	The Xn0 terminal is not assigned with the probe function.	Assign the probe function to the Xn0 terminal.
9732	Xn1 not assigned with probe function	The Xn1 terminal is not assigned with the probe function.	Assign the probe function to the Xn1 terminal.
9742	Compare mode not configured in I/O mapping of encoder axis	Compare mode is not configured in I/O mapping of the encoder axis.	Configure Compare mode in I/O mapping of the encoder axis.
9743	Compare pulse/time not configured in I/O mapping of encoder axis	Compare pulse/time is not configured in I/O mapping of the encoder axis.	Configure Compare pulse/time in I/O mapping of the encoder axis.
9744	Compare size/step not configured in I/O mapping of encoder axis	Compare size/step is not configured in I/O mapping of the encoder axis.	Configure Compare size/step in I/O mapping of the encoder axis.
9745	Compare point value 1 not configured in I/O mapping of encoder axis	Compare point value 1 is not configured in I/O mapping of the encoder axis.	Configure Compare point value 1 in I/O mapping of the encoder axis.
9746	Compare point value 2 not configured in I/O mapping of encoder axis	Compare point value 2 is not configured in I/O mapping of the encoder axis.	Configure Compare point value 2 in I/O mapping of the encoder axis.

Fault Code	Message	Description	Troubleshooting
9747	Physical output status not configured in I/O mapping of encoder axis	Physical output status is not configured in I/O mapping of the encoder axis.	Configure Physical output status in I/O mapping of the encoder axis.
9748	Compare error code not configured in I/O mapping of encoder axis	Compare error code is not configured in I/O mapping of the encoder axis.	Configure Compare error code in I/O mapping of the encoder axis.
9749	Current compare number/position not configured in I/O mapping of encoder axis	Current compare number/ position is not configured in I/O mapping of the encoder axis.	Configure Current compare number/position in I/O mapping of the encoder axis.
9750	Failed to obtain the array start address of the single-axis array comparison output instruction	Failed to obtain the start address of the array of the single-axis array comparison output instruction.	1. Check whether the PLC memory is sufficient. 2. Check whether the background and board software match. 3. Check whether the array of the instruction is out of bounds.
9751	Failed to obtain the axis group start address of the axis group array comparison output instruction	Failed to obtain the start address of the axis group of the axis group array comparison output instruction.	1. Check whether the PLC memory is sufficient. 2. Check whether the background and board software match. 3. Check whether the array of the instruction is out of bounds.
9752	Bus encoder axis not associated with slave	The bus encoder axis is not associated with any slave.	Associate the bus encoder axis with a slave.
9753	X-axis and y-axis of the axis group array comparison instruction not associated with the same slave	The x-axis and y-axis of the axis group array comparison instruction are not associated with the same slave.	Associate the x-axis and y-axis of the axis group comparison output instruction with the same slave.
9754	X-axis of the axis group array comparison instruction not associated with the first channel of the slave	The x-axis of the axis group array comparison instruction is not associated with the first channel of the slave.	Associate the x-axis of the axis group comparison output instruction with the first channel of the slave.
9755	Y-axis of the axis group array comparison instruction not associated with the second channel of the slave	The y-axis of the axis group array comparison instruction is not associated with the second channel of the slave.	Associate the y-axis of the axis group comparison output instruction with the second channel of the slave.
9756	Yn0 not assigned with the one-dimensional comparison output function	The Yn0 terminal is not assigned with the one-dimensional comparison output function.	Assign the one-dimensional comparison output function to the Yn0 output terminal corresponding to the channel.
9757	Absolute value of start value of encoder axis step comparison output instruction greater than 9999999	The absolute value of the start value of the encoder axis step comparison output instruction is greater than 9999999.	Ensure that the absolute value of the floating-point number in the motion instruction does not exceed 9999999.
9758	Absolute value of end value of encoder axis step comparison output instruction greater than 9999999	The absolute value of the end value of the encoder axis step comparison output instruction is greater than 9999999.	Ensure that the absolute value of the floating-point number in the motion instruction does not exceed 9999999.

Fault Code	Message	Description	Troubleshooting
9759	Absolute value of the step of the encoder axis step comparison output instruction greater than 9999999	The absolute value of the step of the encoder axis step comparison output instruction is greater than 9999999.	Ensure that the absolute value of the floating-point number in the motion instruction does not exceed 9999999.
9760	Absolute value of Parameter of the encoder axis step comparison output instruction greater than 9999999	The absolute value of Parameter of the encoder axis step comparison output instruction is greater than 9999999.	Ensure that the absolute value of the floating-point number in the motion instruction does not exceed 9999999.
9761	Mode of the encoder axis comparison output instruction out of range	The value of Mode of the encoder axis comparison output instruction is out of range.	Ensure that the parameter value is within the allowable range.
9762	Time for time control of the encoder axis comparison output out of range	The time for time control of the encoder axis comparison output is out of range.	Ensure that the parameter value is within the allowable range.
9763	Step of the encoder axis step comparison output instruction equal to 0	The step of the encoder axis step comparison output instruction is 0.	Set the step of the step comparison output instruction to a value other than 0.
9764	Start position of the step comparison output instruction equal to end position	The start position of the step comparison output instruction of the encoder axis is equal to the end position.	Ensure that the start position of the step comparison output instruction is not equal to the end position.
9765	Start position of the step comparison output instruction less than end position, but step negative	The start position of the step comparison output instruction of the encoder axis is less than the end position, but the step is negative.	Set the step to a positive value.
9766	Start position of the step comparison output instruction greater than end position, but step positive	The start position of the step comparison output instruction of the encoder axis is greater than the end position, but the step is positive.	Set the step to a negative value.
9767	Size of the encoder axis array comparison output instruction out of range	The value of Size of the encoder axis array comparison output instruction is out of range.	Ensure that the parameter value is within the allowable range.
9768	Absolute value of the target position of the encoder axis array comparison output instruction greater than 9999999	The absolute value of the target position of the encoder axis array comparison output instruction is greater than 9999999.	Ensure that the absolute value of the floating-point number in the motion instruction does not exceed 9999999.

Fault Code	Message	Description	Troubleshooting
9769	Axis performing one-dimensional comparison output, must not be aborted by a two-dimensional comparison output instruction	The axis is performing one-dimensional comparison output and must not be aborted by a two-dimensional comparison output instruction.	Wait for the one-dimensional comparison output to complete or stop the one-dimensional comparison output before executing the two-dimensional comparison output instruction.
9770	EtherCAT slave disconnected during operation	The EtherCAT slave is disconnected during operation.	Check whether the EtherCAT slave is disconnected during operation.
9771	Bus encoder axis in offline commissioning mode	The bus encoder axis is in offline commissioning mode.	The bus encoder axis does not support the offline commissioning mode.
9772	DI terminal not assigned with the preset position function	The DI terminal is not assigned with the preset position function.	Assign the preset position function to the DI terminal before calling the preset position instruction.
9773	Parameter in comparison instruction out of range when the pulse output mode is selected	The value of Parameter in the comparison instruction is out of range when the pulse output mode is selected.	Do not set Parameter to 0 or a negative value when the pulse output mode is selected in the comparison instruction.
9774	2HCE module failed when the comparison output instruction is called	The 2HCE module fails when the comparison output instruction is called.	1. Ensure that the input parameters are within the allowable range. 2. Check whether I/O mapping of the encoder axis is manually modified and whether it meets the I/O mapping configuration requirements of the comparison output instruction.
9775	Set position in ring mode less than 0	The set position in ring mode is less than 0.	Set the position in ring mode to a value greater than or equal to 0.
9776	Y00 not assigned with the two-dimensional comparison output function	The Y00 terminal is not assigned with the two-dimensional comparison output function.	Assign the two-dimensional comparison output function to the Y00 output terminal corresponding to the channel.
9777	Axis performing two-dimensional comparison output, cannot be aborted by a one-dimensional comparison output instruction	The axis is performing two-dimensional comparison output and cannot be aborted by a one-dimensional comparison output instruction.	Wait for the two-dimensional comparison output to complete or stop the two-dimensional comparison output before calling the one-dimensional comparison output instruction.
9800	Failed to read the number of motion control axes	Failed to read the number of motion control axes.	Change the background version.
9801	Motion control axis quantity out of range	The number of motion control axes is out of range.	Reduce the number of axes since the H5U supports at most 32 axes.
9802	Axis failed to request internal space	The axis failed to request internal storage space.	1. Check whether the memory runs out. 2. Contact the manufacturer.
9803	Failed to obtain axis parameters	Failed to obtain axis parameters.	Change the background version.
9804	Failed to obtain the slave	Failed to obtain the slave.	Change the background version.
9805	Failed to obtain the system variable	Failed to obtain the system variable.	1. Check whether the memory runs out. 2. Return the machine to the manufacturer for analysis.

Fault Code	Message	Description	Troubleshooting
9806	Improper gear ratio settings	Parameters related to the gear ratio are set improperly.	1. Ensure that the numerator and denominator of the gear ratio are greater than 0. 2. Ensure that the number of pulses per revolution of the motor/encoder is greater than 0. 3. Ensure that the displacement per revolution of the rotary table is between 0.000001 and 9999999.
9807	Improper software limiting parameters	The software limiting parameters are set improperly.	1. Ensure that the positive limit is not greater than 9999999. 2. Ensure that the negative limit is not greater than 9999999. 3. Ensure that the negative limit is not greater than the positive limit.
9808	Improper linear/rotary mode	The linear/rotary mode parameter is set improperly.	Note that only the linear mode and rotary mode are supported.
9809	Improper revolution cycle	The revolution cycle is set improperly.	Ensure that the revolution cycle is between 0.01 and 9999999.
9810	Improper encoder mode	The encoder mode is set improperly.	Ensure that the encoder mode is set properly. Note that only the incremental mode and absolute value mode are supported.
9811	Improper homing parameter setting	The homing parameter is set improperly.	1. Do not modify the homing mode of the bus servo axis. If you want to modify the homing mode of the bus servo axis, write to the SDO. 2. Check whether the homing mode is set properly. Note that only the values 17 to 30 and 35 are supported.
9812	Limit, home, or probe terminal Modbus address out of range	The Modbus address setting of the limit, home, or probe terminal is out of range.	1. Check whether the set address is out of the range of Modbus addresses. 2. Select an address among X0 to X7 for the home signal. 3. Select an address among X0 to X7 for the probe signal.
9813	Improper pulse output mode setting of the local pulse axis	The pulse output mode of the local pulse axis is set improperly.	Check whether the pulse output mode of the local pulse axis is set improperly.
9814	Improper limiting deceleration	The limiting deceleration is set improperly.	Ensure that the limiting deceleration is between 0.0001 and 9999999.
9815	Improper deceleration upon axis fault	The deceleration upon axis fault is set improperly.	Ensure that the deceleration upon axis fault is between 0.0001 and 999999.
9816	Improper maximum velocity	The maximum velocity is set improperly.	Ensure that the maximum velocity is between 0.0001 and 999999.
9817	Improper maximum positive torque	The maximum positive torque is set improperly.	Ensure that the maximum positive torque is between 1 and 65534.
9818	Improper maximum negative torque	The maximum negative torque is set improperly.	Ensure that the maximum negative torque is between 1 and 65534.
9819	Improper maximum jogging velocity	The maximum jogging velocity is set improperly.	Ensure that the maximum jogging velocity is between 0.0001 and the maximum velocity.
9820	Improper maximum acceleration	The maximum acceleration is set improperly.	Ensure that the maximum acceleration is between 0.0001 and 9999999.
9821	Improper following error threshold	The following error threshold is set improperly.	Ensure that the following error threshold is between 0.0001 and 9999999.

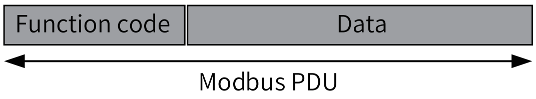
Fault Code	Message	Description	Troubleshooting
9822	Improper velocity reach threshold	The velocity reach threshold is set improperly.	Ensure that the velocity reach threshold is between 0.0001 and 9999999.
9823	Improper homing velocity	The homing velocity is set improperly.	1. Ensure that the homing velocity is between 0.0001 and 9999999. 2. Ensure that the homing velocity is not greater than the maximum velocity. 3. Ensure that the value obtained by multiplying the homing velocity by the gear ratio is between 1 and 2148483647.
9824	Improper homing approach velocity	The homing approach velocity is set improperly.	1. Ensure that the homing approach velocity is between 0.0001 and 9999999. 2. Ensure that the homing approach velocity is not greater than the maximum velocity. 3. Ensure that the value obtained by multiplying the homing approach velocity by the gear ratio is between 1 and 2148483647. 4. Ensure that the homing approach velocity is less than the homing velocity.
9825	Homing position mode setting out of range	The homing position mode setting is out of range.	Ensure that the parameter value is within the allowable range.
9826	Improper homing acceleration	The homing acceleration setting is improper.	1. Ensure that the homing acceleration is between 0.0001 and 9999999. 2. Ensure that the homing acceleration is not greater than the maximum acceleration.
9827	Homing timeout time out of range	The homing timeout time is out of range.	Ensure that the homing timeout time is greater than or equal to 1.

20 Appendix

20.1 Modbus Protocol

20.1.1 Modbus Message Description

The Modbus application protocol defines a simple protocol data unit (PDU) independent of the underlying communication layers.



The mapping of Modbus protocol on different buses or networks can introduce some additional fields on the protocol data unit. The client initiating a Modbus transaction builds a Modbus PDU and then adds an additional field to build an appropriate communication PDU.

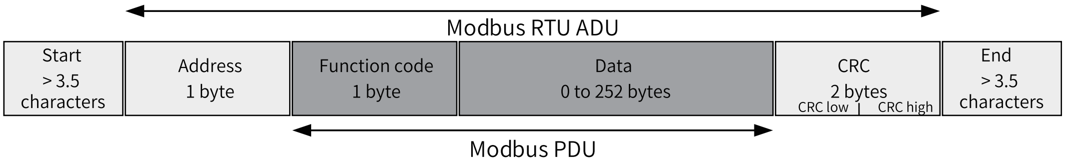
Data encoding

Modbus uses a "big-Endian" representation for addresses and data items. This means that when multiple bytes are transmitted, the most significant bit is sent first. Example:

Register Size	Value	Description
16-bit	0x1234	The first byte sent is 0x12, then 0x34.

20.1.2 Modbus-RTU Message Frame

When a device communicates over a Modbus serial link in the Remote Terminal Unit (RTU) mode, each 8-bit byte in the message consists of two 4-bit hexadecimal characters. The RTU mode features high data density, leading to higher throughput than the ASCII mode at the same baud rate. However, each message must be transmitted as a continuous character stream.



20.1.3 Modbus-ASCII Message Frame

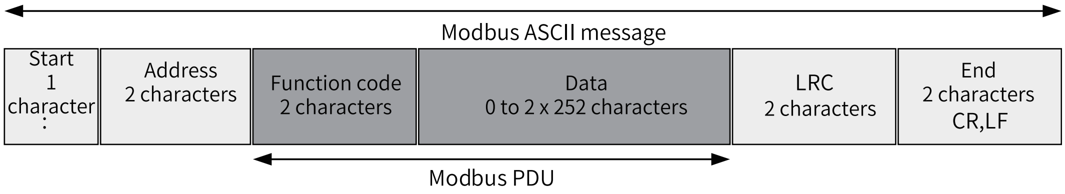
When a device on a Modbus serial link is configured to communicate in the American Standard Code for Information Interchange (ASCII) mode, each 8-bit byte in the message is sent as two ASCII characters. This mode is used when a communication link or device cannot comply with the timing management of the RTU mode.

Note

Note: This mode is less efficient than the RTU mode because one byte requires two characters.

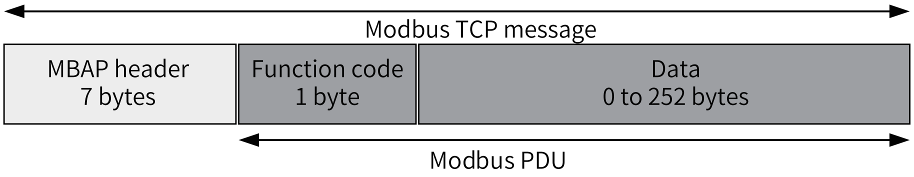
For example, the byte 0x5B is encoded as two characters: 0x35 and 0x42 (0x35 = "5" and 0x42 = "B" in ASCII encoding).

In the ASCII mode, a message is delimited by special characters at the beginning and end of the frame. A message must start with a colon (:) (ASCII hexadecimal 3A) and end with a carriage return–line feed (CR LF) pair (ASCII hexadecimal 0D and 0A).



20.1.4 Modbus-TCP Message Frame

The following figure illustrates the encapsulation of Modbus requests or responses in a Modbus-TCP/IP network.



TCP/IP uses a special message header to identify the Modbus application data unit. Such a message header is called the Modbus Protocol Application Header (MBAP).

In the MBAP header, a single-byte unit identifier is used instead of the Modbus slave address field that is commonly used on a Modbus serial link. The unit identifier is used for communication between devices (such as bridges, routers, and gateways) that support multiple independent Modbus terminal units through a single IP address.

The MBAP header includes the following fields.

Field	Length	Description	Client	Server
Transaction meta identifier	2 bytes	Identifier of the Modbus request/response transaction being processed	Started by the client	Copied by the server from the received request
Protocol identifier	2 bytes	0: Modbus protocol	Started by the client	Copied by the server from the received request
Length	2 bytes	Number of bytes of the next field	Started by the client (request)	Started by the server (response)
Unit identifier	1 byte	Identifier of the remote slave connected on a serial link or other bus	Started by the client	Copied by the server from the received request

- The header contains 7 bytes.
- Transaction processing identifier: used to pair transaction processing. In the response, the Modbus server copies the transaction identifier from the request.
- Protocol identifier: used for multiplexing within the system. The value 0 indicates the Modbus protocol.
- Length: indicates the number of bytes of the next field, including the unit identifier and data field.

- Unit identifier: used for routing within the system. This field is specifically used for communication with Modbus or Modbus+ serial link slaves through gateways between Ethernet TCP-IP networks and Modbus serial links. The Modbus client sets this field in the request, and the server must return the same value in the response.

20.1.5 Function Code Definitions

20.1.5.1 Modbus Data Model

Modbus bases its data model on a series of tables that have distinguishing characteristics. The four primary tables are:

Table	Object Type	Access Type	Comments
Discrete inputs	Single bit	Read-only	This type of data can be provided by an I/O system.
Coils	Single bit	Read-write	This type of data can be altered by an application program.
Input registers	16-bit	Read-only	This type of data can be provided by an I/O system.
Holding registers	16-bit	Read-write	This type of data can be altered by an application program.

20.1.5.2 Function Code List

Function Code	Definition
01 (0x01)	Read coils
02 (0x02)	Read discrete inputs
03 (0x03)	Read multiple registers
04 (0x04)	Read input registers
05 (0x05)	Write single coil
06 (0x06)	Write single register
15 (0x0F)	Write multiple coils
16 (0x10)	Write multiple registers

20.1.5.3 Function Code Explanation

01 (0x01): Read coils/02 (0x02): Read discrete inputs

This function code is used to read 1 to 2000 contiguous status of coils (or discrete inputs) in a remote device.

Table 20–1 Request PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x01: Read coils/0x02: Read discrete inputs
2	Coil starting address	2	Upper bits are followed by lower bits. See coil addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Number of coils	2	Upper bits are followed by lower bits (N). The maximum value of N is 2000.

Table 20–2 Response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x01: Read coils/0x02: Read discrete inputs
2	Number of bytes	1	Value: (N + 7)/8
3	Coil status	(N + 7)/8	Every 8 coils are combined into one byte. If the number of coils is not a multiple of 8, undefined bits are filled with 0. The first 8 coils are in the first byte, and the coil with the smallest address is in the least significant bit. This pattern continues for the rest of the coils.

Table 20–3 Error response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	Function code + 0x80; 0x81: Read coils/0x82: Read discrete inputs
2	Exception code	1	0x01, 0x02, 0x03, or 0x04. See the exception code list.

03 (0x03): Read multiple registers/04 (0x04): Read input registers

This function code is used to read the content of a contiguous block of holding registers (or input registers) in a remote device.

Table 20–4 Request PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x03: Read multiple registers/0x04: Read input registers
2	Register starting address	2	Upper bits are followed by lower bits. See register addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Number of registers	2	Upper bits are followed by lower bits (N). The maximum value of N is 125.

Table 20–5 Response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x03: Read multiple registers/0x04: Read input registers
2	Number of bytes	1	Value: N x 2
3	Register value	N x 2	Every two bytes represents one register value, with upper bits followed by lower bits. The register with the minimum address is in the foremost.

Table 20–6 Error response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	Function code + 0x80; 0x83: Read multiple registers/0x84: Read input registers
2	Exception code	1	0x01, 0x02, 0x03, or 0x04. See the exception code list.

05 (0x05): Write single coil

This function code is used to write a single output to either ON or OFF in a remote device.

Table 20–7 Request PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x05: Write single coil
2	Coil address	2	Upper bits are followed by lower bits. See coil addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Coil status	2	Upper bits are followed by lower bits. ON is 0xFF00, while OFF is 0x0000.

Table 20–8 Response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x05: Write single coil
2	Coil address	2	Upper bits are followed by lower bits. See coil addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Coil status	2	Upper bits are followed by lower bits. Active when the value is other than 0

Table 20–9 Error response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	Function code + 0x80; 0x85: Write single coil
2	Exception code	1	0x01, 0x02, 0x03, or 0x04. See the exception code list.

06 (0x06): Write single register

This function code is used to write a single holding register in a remote device.

Table 20–10 Request PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x06: Write single register
2	Register address	2	Upper bits are followed by lower bits. See register addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Register value	2	Upper bits are followed by lower bits.

Table 20–11 Response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x06: Write single register
2	Register address	2	Upper bits are followed by lower bits. See register addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Register value	2	Upper bits are followed by lower bits.

Table 20–12 Error response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	Function code + 0x80; 0x86: Write single register
2	Exception code	1	0x01, 0x02, 0x03, or 0x04. See the exception code list.

15 (0x0F): Write multiple coils

This function code is used to force each coil in a sequence of coils to either ON or OFF in a remote device.

Table 20–13 Request PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x0F: Write multiple coils
2	Coil starting address	2	Upper bits are followed by lower bits. See coil addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Number of coils	2	Upper bits are followed by lower bits (N). The maximum value of N is 1968.
4	Number of bytes	1	Value: (N + 7)/8
5	Coil status	(N + 7)/8	Every 8 coils are combined into one byte. If the number of coils is not a multiple of 8, undefined bits are filled with 0. The first 8 coils are in the first byte, and the coil with the smallest address is in the least significant bit. This pattern continues for the rest of the coils.

Table 20–14 Response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x0F: Write multiple coils
2	Coil starting address	2	Upper bits are followed by lower bits. See coil addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Number of coils	2	Upper bits are followed by lower bits.

Table 20–15 Error response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	Function code + 0x80; 0x8F: Write multiple coils
2	Exception code	1	0x01, 0x02, 0x03, or 0x04. See the exception code list.

16 (0x10): Write multiple registers

This function code is used to write a block of contiguous registers (1 to 123 registers) in a remote device.

Table 20–16 Request PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x10: Write multiple registers
2	Register starting address	2	Upper bits are followed by lower bits. See register addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Number of registers	2	Upper bits are followed by lower bits (N). The maximum value of N is 123.
4	Number of bytes	1	Value: N x 2
5	Register value	N x 2	Register value

Table 20–17 Response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	0x10: Write multiple registers
2	Register starting address	2	Upper bits are followed by lower bits. See register addressing. For details, see . *"7.5.2 Parameters and Addresses" on page 256*
3	Number of registers	2	Upper bits are followed by lower bits.

Table 20–18 Error response PDU

No.	Meaning of Data (Byte)	Number of Bytes	Description
1	Function code	1	Function code + 0x80; 0x90: Write multiple registers
2	Exception code	1	0x01, 0x02, 0x03, or 0x04. See the exception code list.

20.1.6 Exception Code List

Code	Name	Description
0x01	Illegal function code	The function code received in the query is not an allowable action for the server (or slave). This may be because the function code is only applicable to new devices, and is not implementable in the unit selected. It could also indicate that the server (or slave) is in the wrong state to process a request of this type, for example, because it is unconfigured and is being asked to return register values.
0x02	Illegal data address	The data address received in the query is not an allowable address for the server (or slave). More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, a request with the offset 96 and the length 4 will succeed, but a request with the offset 96 and the length 5 will result in exception code 0x02.
0x03	Illegal data value	A value contained in the query data field is not an allowable value for server (or slave). This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect. It specifically does not mean that a data item submitted for storage in a register has a value outside the expectation of the application program, since the Modbus protocol is unaware of the significance of any particular value of any particular register.
0x04	Slave device failure	An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.

20.2 Firmware Programming and Upgrade

20.2.1 Firmware Programming

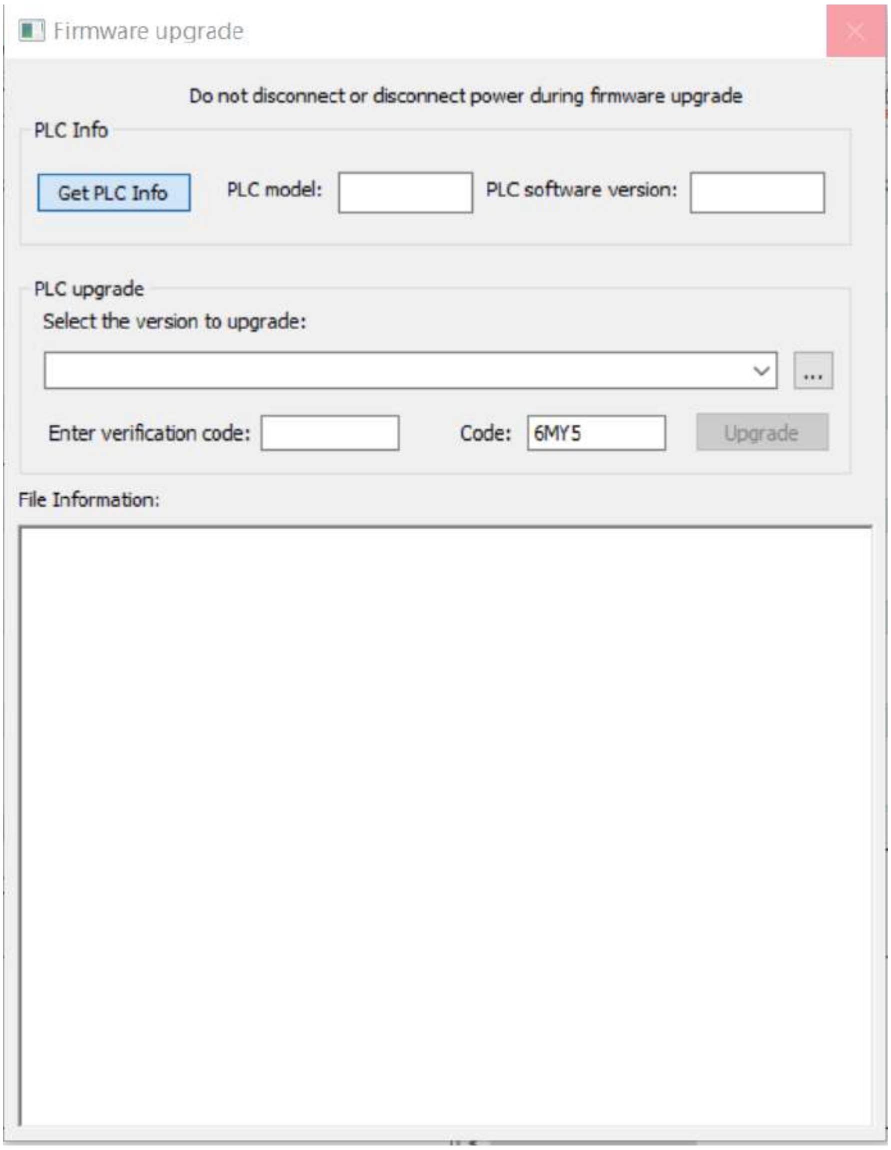
⚠ Caution

- The PLC firmware programming function of AutoShop is only available for the Easy series.
- When multiple AutoShop software applications are opened, only one of them is allowed to perform programming, while others cannot access the "Firmware burning" function.
- The Easy series models involve two programming files: one for the Easy30X, Easy32X, and Easy50X models, and the other for the Easy52X models.

Prerequisite: USB cable and firmware programming file are ready. You can log in to the official website of Inovance (*www.inovance.com*) to obtain the firmware programming file.

1. Turn off the PLC power.

2. In the menu bar, choose "Tools" > "Firmware burning". The "Firmware burning" page is displayed.



3. Click "..." and select the firmware programming file.
4. Plug the USB cable (if the USB cable is already connected, remove and re-plug it). A dialog box pops up to prompt whether to program. Click "OK" to start programming.
   After the programming succeeds, a prompt box indicating the programming is succeeded will be displayed.

20.2.2 Firmware Upgrade

20.2.2.1 Firmware Upgrade Through Ethernet

1. In the menu bar, choose "Tools" > "Firmware upgrade". The "Firmware upgrade" dialog box is displayed.
2. Select the firmware version to be upgraded, enter the verification code, and click "Upgrade".
3. Wait for the "Upgrade successful" prompt box to pop up, and the firmware upgrade is completed.

Note

During firmware upgrade, ensure that the PLC is powered normally. Powering off the PLC during upgrade may cause the PLC to fail to start or function normally. In most of such cases, firmware upgrade can still be done by using an SD card. Otherwise, the PLC needs to be returned to the factory for repair.

20.2.2.2 Firmware Upgrade Through SD Cards

⚠Warning

During upgrade using an SD card, power-off is strictly prohibited. Otherwise, the PLC may become unusable or other serious abnormalities may occur.

To upgrade the firmware using an SD card, program the SD card first according to the following steps.

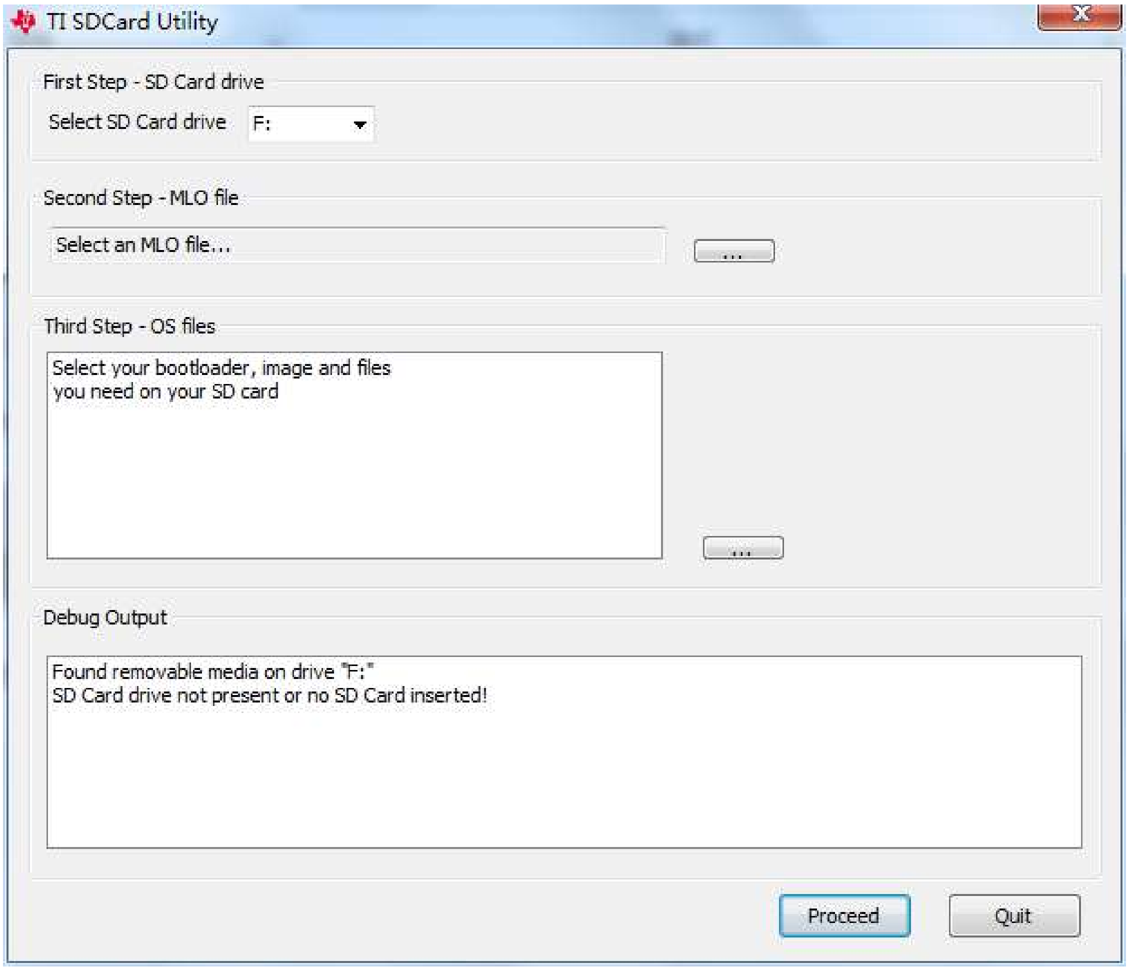
1. Prepare hardware.

   Prepare an SD (TF) card as shown in the figure, with the card storage capacity no more than 32 G.

2. Insert the SD card into a card reader and plug the card reader into the USB port of the computer.
3. Double-click the SD card programming tool to open it.

   Tool download address: http://bbs.inovance.com/t-1797.html

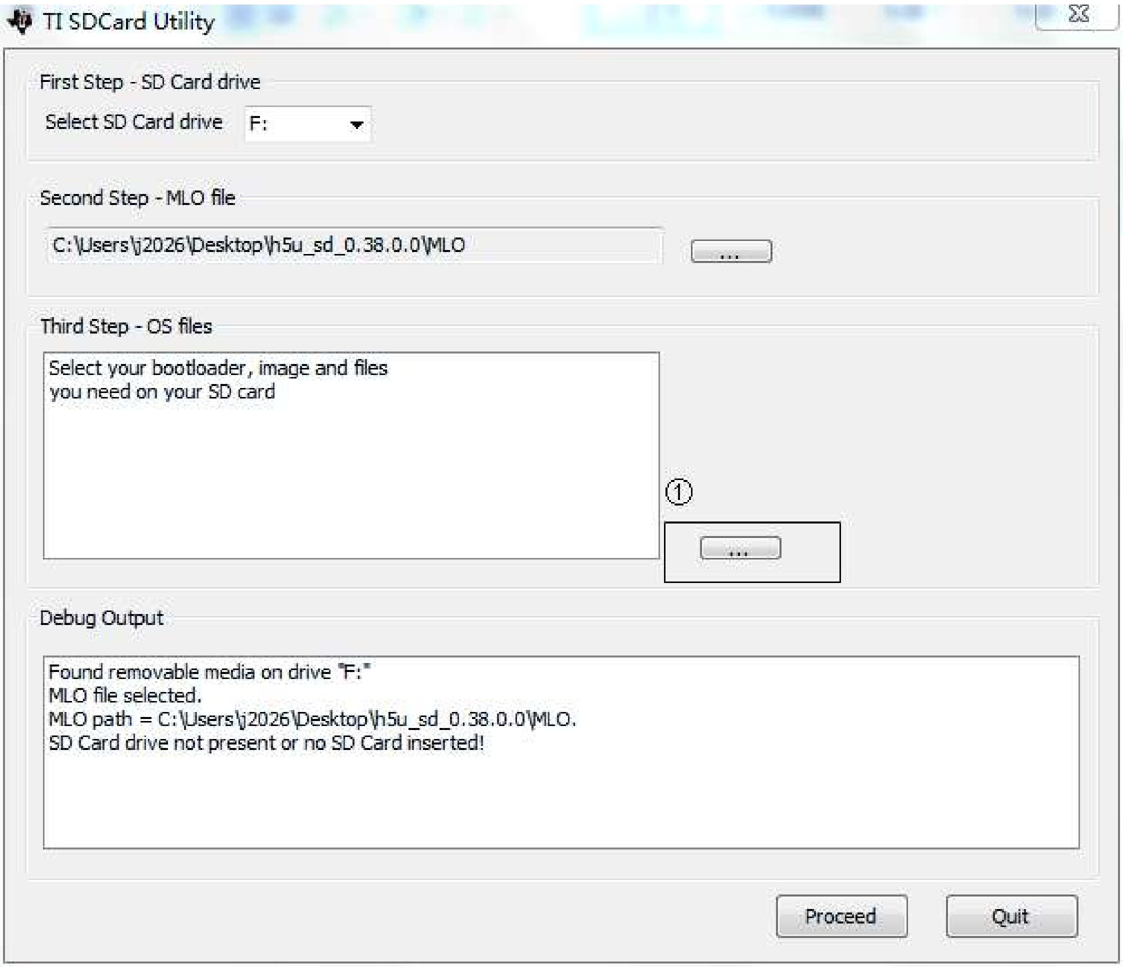
4. The following interface is displayed and shows which disk the card reader is located, as shown in the following figure.



```
Found removable media on drive "F:"
SD Card drive not present or no SD Card inserted!
```

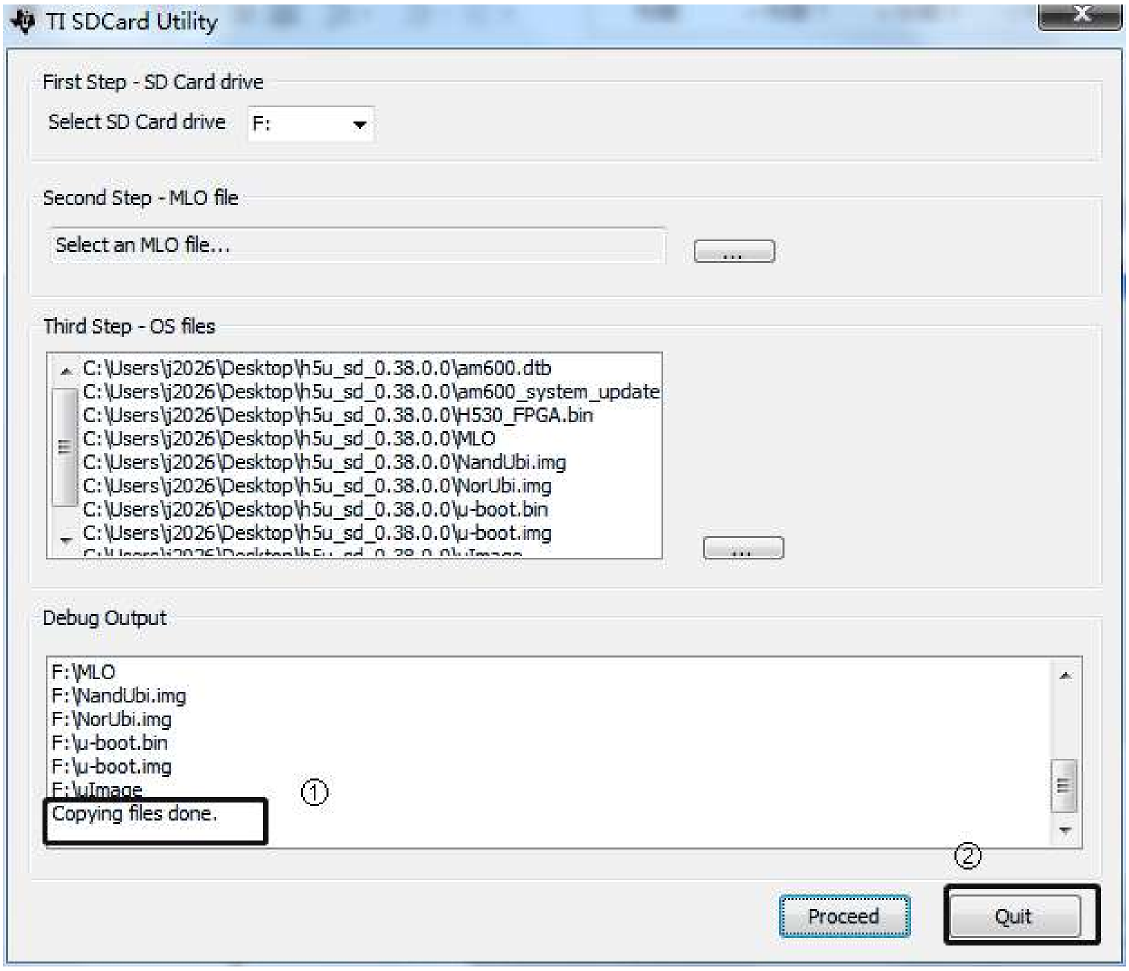
5. Select the MLO file in the upgrade package and click “Open” .

6. Click the button specified in the following figure to open the folder where the programming content is located. Press Ctrl+A to select all the files and then click "Open".



```
Found removable media on drive "F:"
MLO file selected.
MLO path = C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\MLO.
SD Card drive not present or no SD Card inserted!
```

7. Click "Proceed". After the following interface appears, click "Format" and then "Start" to format the SD card.
8. After the formatting, click "Close" to start programming. After the programming is completed, the following interface is displayed.



```
C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\am600.dtb
C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\am600_system_update
C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\H530_FPGA.bin
C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\MLO
C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\NandUbi.img
C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\NorUbi.img
C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\u-boot.bin
C:\Users\j2026\Desktop\h5u_sd_0.38.0.0\u-boot.img
```

```
F:\MLO
F:\NandUbi.img
F:\NorUbi.img
F:\u-boot.bin
F:\u-boot.img
F:\uImage
Copying files done.
```

9. Insert the SD card into the SD card slot of the controller.
10. Power off the PLC and then power it on. When the LED display shows UU, it indicates that the upgrade has started. The upgrade process may take about one minute. When the upgrade succeeds, the LED display shows 00 or CC. Remove the SD card to complete the upgrade.

Note

- During firmware upgrade, ensure that the PLC is powered normally. Powering off the PLC during upgrade may cause the PLC to fail to start or function normally. In such a case, try again to upgrade the firmware using the SD card. If the upgrade fails, send the PLC to the factory for repair.
- During firmware upgrade, if the LED display shows flashing ER, it means that the upgrade is successful but the PLC detects a program error or communication error during running.

20.3 Applying the Function of Download File Generation

20.3.1 Generating Down Files

20.3.1.1 Overview

The download file function refers to the ability of the PLC project to compile and generate a Down file, which can be downloaded without opening the original project.

- Batch update or upgrade PLC projects using an SD card.

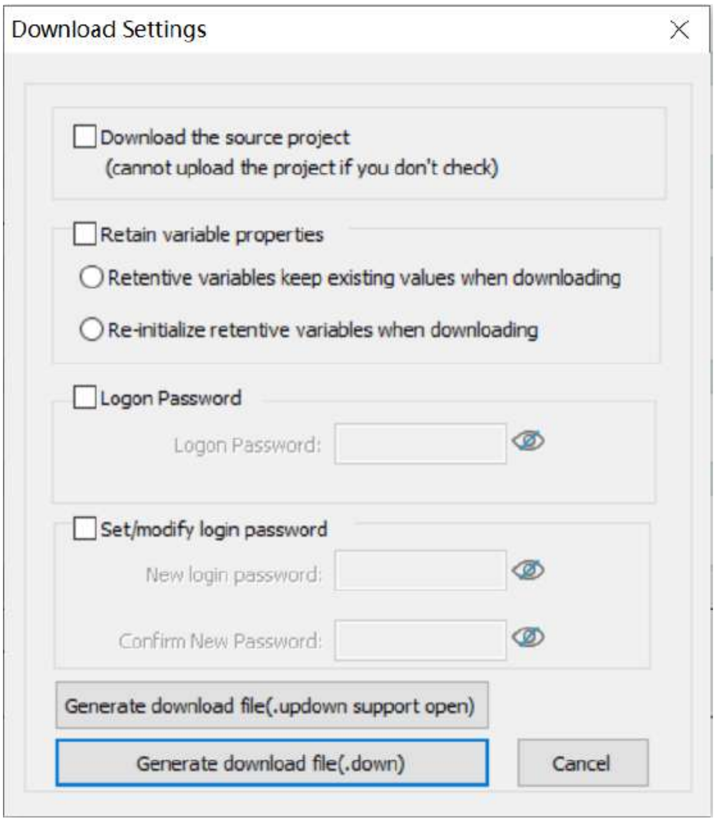
- Update a PLC project using AutoShop software tool.

20.3.1.2 Generating Down Files

Generating Down files

Before downloading a Down file, generate the Down file in the AutoShop background. The specific steps are as follows.

1. Open the PLC project and choose "File" > "Generate download file (D)".
2. In the "Download Settings" dialog box that is displayed, set the download file properties, and then click "Generate download file (.down)".


Download Settings
□ Download the source project
(cannot upload the project if you don't check)
□ Retain variable properties
○ Retentive variables keep existing values when downloading
○ Re-initialize retentive variables when downloading
□ Logon Password
Logon Password:
□ Set/modify login password
New login password:
Confirm New Password:
Generate download file(.updown support open)
Generate download file(.down)
Cancel

Interface:

- "Download the source project": Select this option to enable project uploading, or deselect this option to disable project uploading.
- "Retain variable properties"
  Retain existing values of retentive variables when downloading; or

  Re-initialize retentive variables when downloading.
- "Logon password"
  If the PLC does not have a login password, leave this option deselected.

  If the PLC has a login password, this option must be selected and the PLC login password must be entered. If this option is not selected or the password is incorrect, the upgrade will fail.
- "Set/modify login password"

To modify the current login password of the PLC, fill out the "Logon Password", "New Login Password", and "Confirm New Password" fields.

To set a login password when the PLC does not have a login password, just fill out the "New Login Password" and "Confirm New Password" fields.

- 👁: shows or hides passwords.

Note

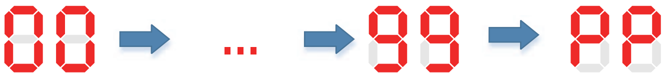
The password takes effect immediately after the Down file is upgraded.

After the password takes effect, the PLC will log out the current user.

3. Select the archive path for the Down file and click "OK".

20.3.1.3 Upgrading Down Files Through SD Cards

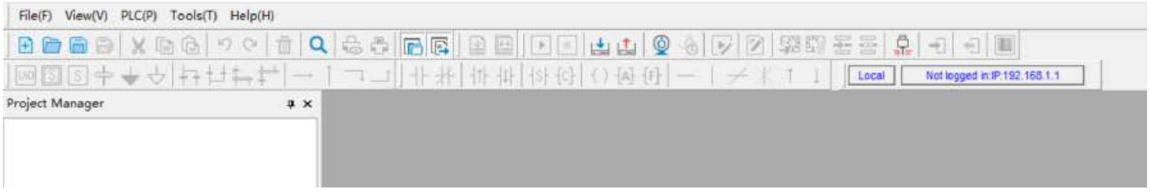
Put the Down file compiled using AutoShop in the "PLCProgram" directory of the SD card, and then insert the SD card into the PLC main module. Press and hold the MFK key on the PLC panel for three seconds to enter the "Sd" menu. Press the MFK key again to start programming the user program in the SD card into the PLC host. The LED display shows the programming progress (00 to 99), and shows "PP" after the programming is completed.



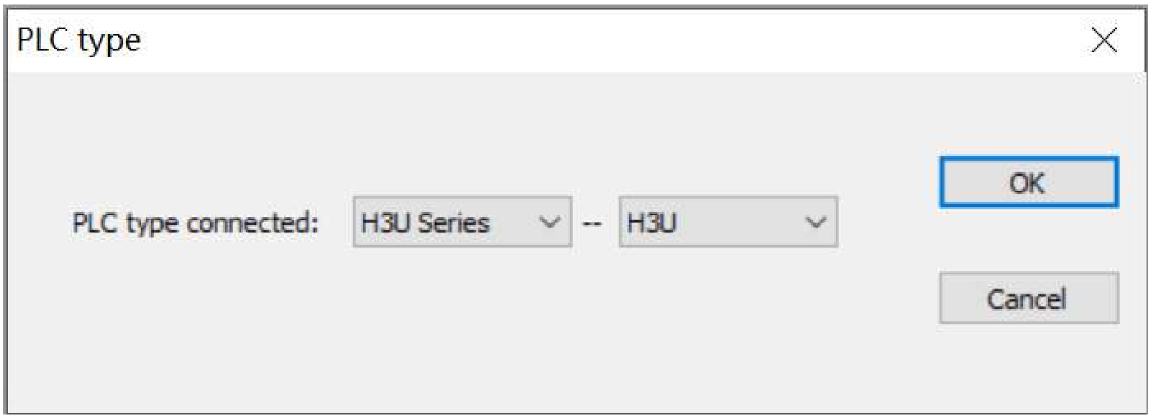
If the password is incorrect, the indicator prompts E5, indicating a password verification error.

20.3.1.4 Downloading Down Files Through AutoShop

To download the generated Down file, close the project first, as shown in the following figure.



Click the Download button and select the PLC type connected.



PLC type

PLC type connected: H3U Series -- H3U

Click "Download". If the password verification fails, the system pops up a prompt box, and you need to enter the correct password.

Note

If the PLC is in the logged-in state when the Down file is downloaded in the background, the PLC will be logged out first.

20.3.1.5 Compatibility

When a Down file generated by AutoShop V4.0.0.0 is downloaded to PCB software of V3.0.0.0 or earlier version, errors such as Down upgrade failure, file format error, and parsing failure may occur.

20.3.2 Generating Updown Files

20.3.2.1 Overview

Updown files are Down files that can be uploaded. Updown files enable upload and download of user programs.
AutoShop allows you to compile a project into an Updown file, and open the Updown file in AutoShop to further edit the project.

Updown files can be downloaded and uploaded through HMI, enabling quick copy and transfer of projects between different PLCs.

Updown files can be downloaded and uploaded using SD cards.

Updown files can also be downloaded and uploaded through AutoShop, making project management flexible.

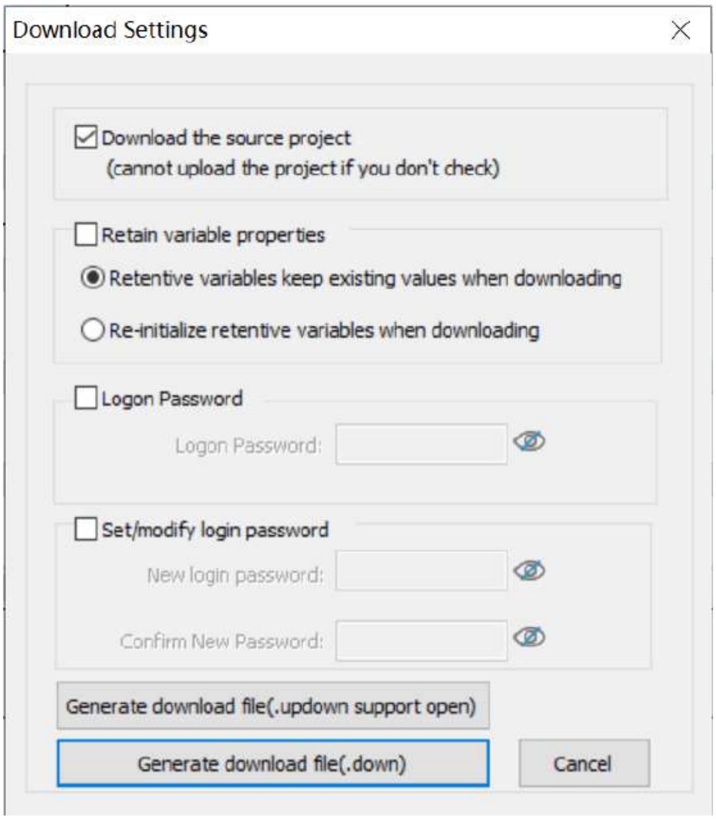
Operations involved are:

Function	Tool
Compile and generate Updown files	AutoShop
Open and edit Updown files	AutoShop
Upload and download Updown files through AutoShop	AutoShop
Upload and download Updown files through HMI	HMI (IT7000)
Upload and download Updown files through SD cards	SD card

20.3.2.2 Generating Updown Files

Follow these steps to generate an Updown file:

1. In the programmed and compiled project, choose "File" > "Generate download file". The "Download Settings" dialog box is displayed.
2. Click "Generate download file(.updown support open)" to generate an Updown file.



Download Settings

☑ Download the source project
(cannot upload the project if you don't check)

□ Retain variable properties

● Retentive variables keep existing values when downloading

○ Re-initialize retentive variables when downloading

□ Logon Password

Logon Password:

□ Set/modify login password

New login password:

Confirm New Password:

Generate download file(.updown support open)
Generate download file(.down)
Cancel

Interface:

- "Download the source project": When this option is selected, the Updown file generated includes the project source code. Only Updown files with this option selected can be opened and edited.
- "Retain variable properties"
  Retain existing values of retentive variables when downloading; or

  Re-initialize retentive variables when downloading.
- "Logon password"
  The PLC login password, which must be consistent with the login password of the target PLC to complete the download of the Updown file.

  If the target PLC does not have a login password, do not select the "Logon Password" option.
- "Set/modify login password"
  After a successful Updown file download, the target PLC's login password is updated to the "New Login Password".
- 👁: shows or hides passwords.

3. Select the archive path for the Updown file and click "OK".

20.3.2.3 Opening Updown Files

Follow these steps to open and edit an Updown file:

1. Choose "File" > "Open Project".
2. In the "Open" window that is displayed, select the "*.updown" file you want to open.

3. After selecting the file, click "Open" to open the Updown file. If the Updown file is password-protected, enter the password for verification before the file can be opened.

Note

If the Updown file has "Set/modify login password" enabled, use the "Set/modify login password" of the Updown file for verification.

If the Updown file only has "Logon Password" enabled, use the "Logon Password" of the Updown file for verification.

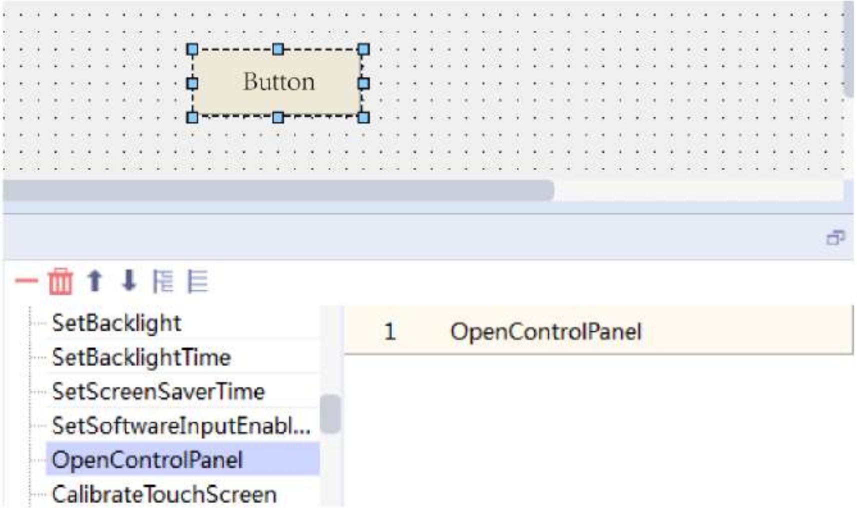
20.3.2.4 Uploading and Downloading Updown Files Through HMI

Uploading or downloading Updown files through HMI requires a firmware version of 0.8.8.27 or later.

Downloading Updown files

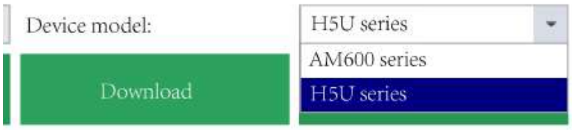
1. Enter the control panel of the IT7000 series HMI.

- In InoTouchPad, create an IT7000 project, configure a button, and configure the system function "OpenControlPanel" for the button to enter the control panel.

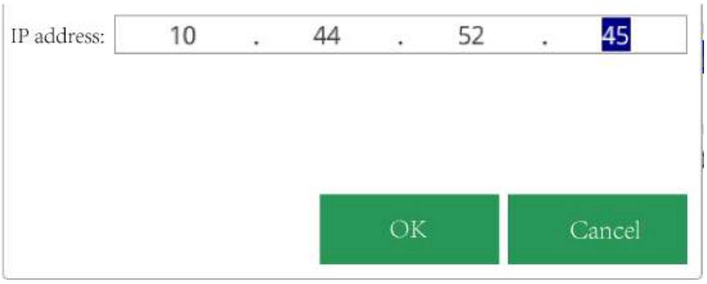


- Power on the IT7000 HMI and press and hold the screen to enter the control panel.

2. Click the "Download" menu in the control panel.
3. In the window that is displayed, select the corresponding mounted device and the Updown file to be downloaded.
4. Select the target device series and click "Download".



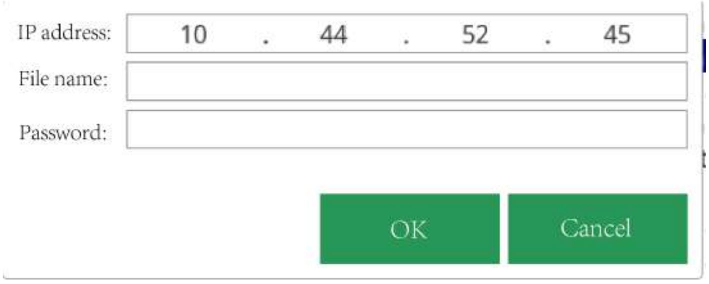
5. In the window that is displayed, enter the IP address of the PLC device and click "OK".



IP address: 10 . 44 . 52 . 45

Uploading Updown files

The steps for upload are similar to those for download. The control panel interface is as shown in the following figure.



IP address: 10 . 44 . 52 . 45

File name:

Password:

Note

To upload a file, you need to enter the device IP and rename the file to make the file name end with ".updown", for example, "test.updown".

The password for upload is the password of the uploaded Updown file.

Both Down and Updown files can be downloaded, but only Updown files can be uploaded.

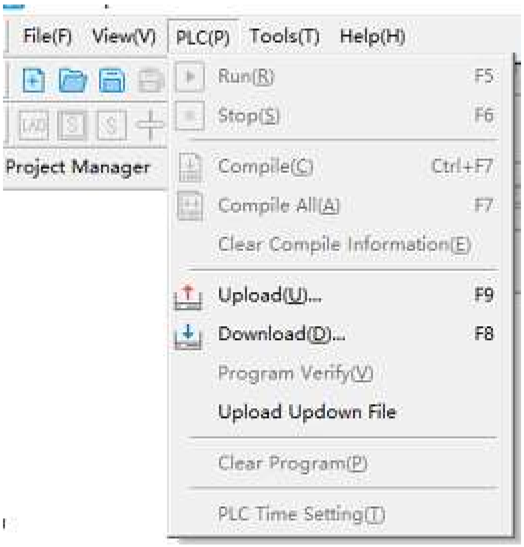
20.3.2.5 Uploading and Downloading Updown Files Through AutoShop

Downloading Updown files

Close the project. Click the Download button in the toolbar. After connecting to the PLC, select the "*. updown" file to be downloaded in the window that is displayed, as shown in the following figure.

Uploading Updown files

Close the project. Choose "PLC" > "Upload Updown File".

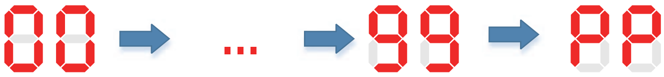


Note

If an Updown file that has been downloaded to the PLC is downloaded again in the Down format or through the background, inconsistency may occur between the Updown file uploaded and the actual project running in the PLC.

20.3.2.6 Uploading and Downloading Updown Files Through SD Cards

Put the Updown file compiled using AutoShop in the "PLCProgram" directory of the SD card, and then insert the SD card into the PLC main module. Press and hold the MFK key on the PLC panel for three seconds to enter the "Sd" menu. Press the MFK key again to start programming the user program in the SD card into the PLC host. The LED display shows the programming progress (00 to 99), and shows "PP" after the programming is completed.



If the password is incorrect, the indicator prompts E5, indicating a password verification error.

20.4 Applying Customized Variables in Communication

20.4.1 Overview

In the function of customized variables, addresses of variables are automatically allocated by the software. Therefore, the variables cannot be accessed directly using fixed addresses. Customized variables can be accessed in the following two ways:

- HMI tag communication: only available for Inovance IT7000 series touch screens
- Mapping address: applicable to all devices that support the Modbus protocol

20.4.2 Example Project Requirements

Write an H5U marquee program and have the corresponding bit elements and control word status displayed through the IT7000 HMI.

20.4.3 PLC Programming

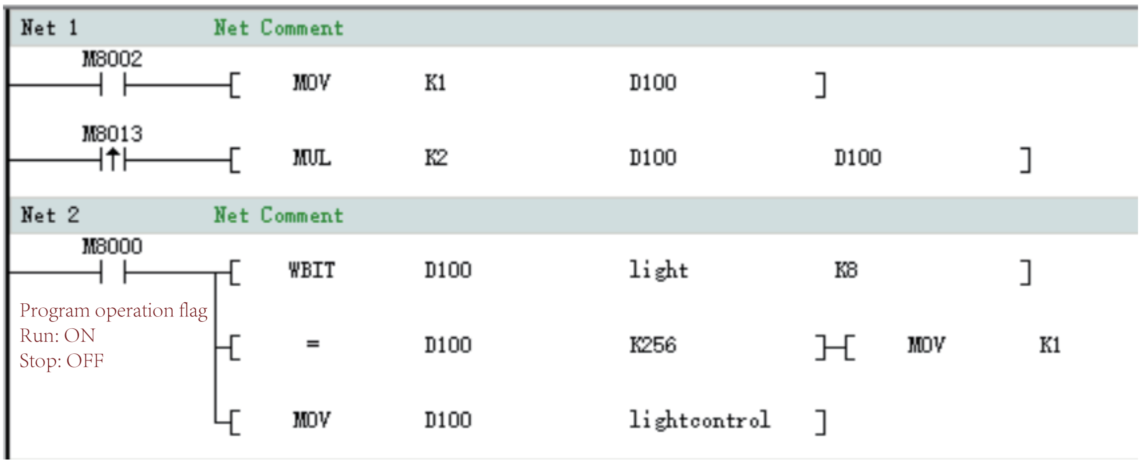
20.4.3.1 Accessing Customized Variables Through HMI Tag Communication

Creating a PLC project

Create a PLC project. For details, see *"2.4.2 Creating a Project" on page 38*.

Writing a PLC program

Write a program as shown in the following figure, where "light" is an array containing eight Bool-type variables and "light_control" is an INT-type variable used for HMI display.

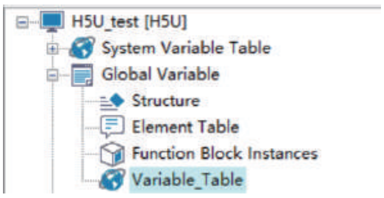


```
Net 1          Net Comment
  M8002
──┤ ├──────[ MOV   K1     D100          ]
  M8013
──┤↑├──────[ MUL   K2     D100    D100    ]
Net 2          Net Comment
  M8000
──┤ ├─────┬[ WBIT  D100   light   K8      ]
          ├[ =     D100   K256   ]─[ MOV   K1
          └[ MOV   D100   lightcontrol ]
```

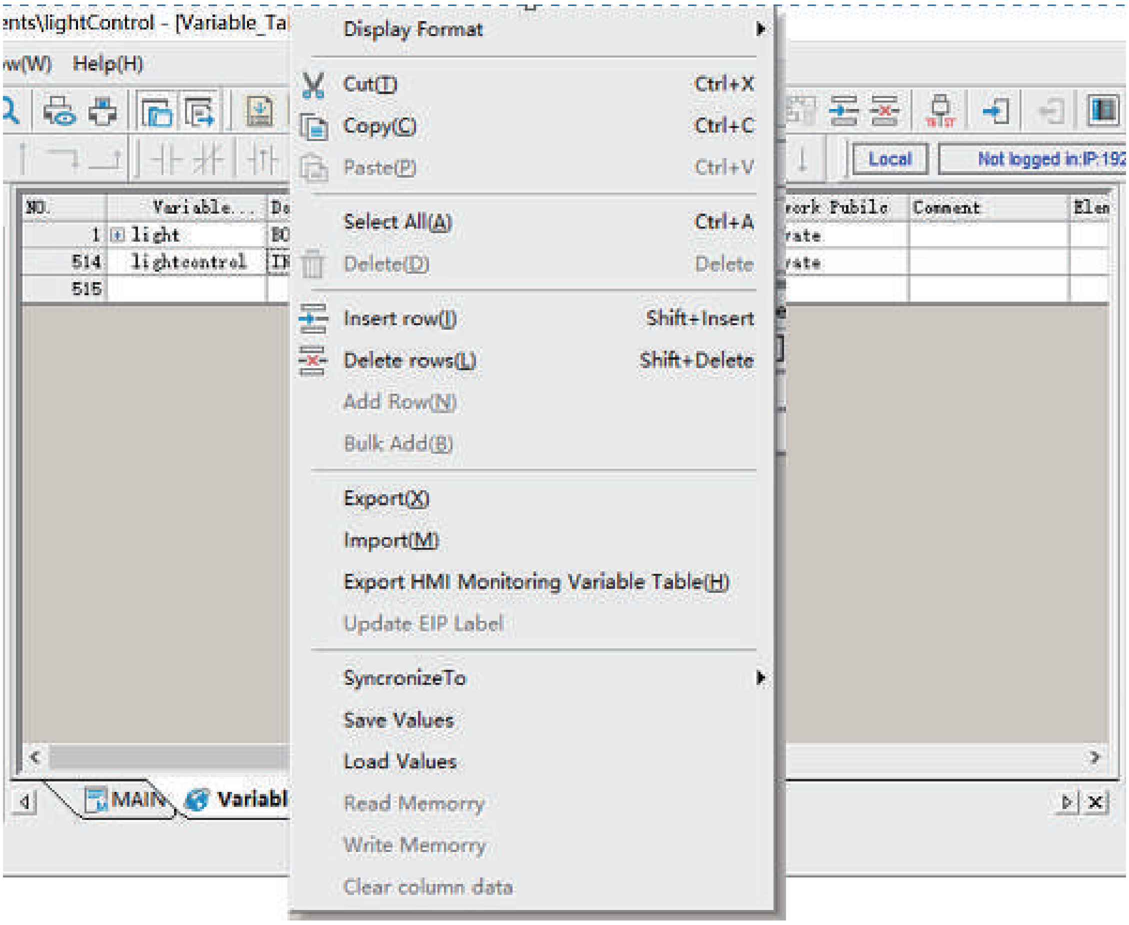
Program operation flag
Run: ON
Stop: OFF

Compiling the project and export the variable table

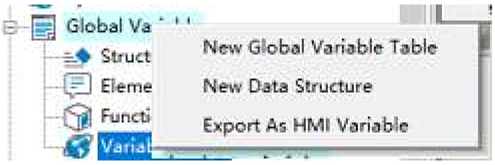
1. After writing the project, click the compile button to complete the program compilation.
2. After the compilation is completed, double-click "Variable Table" in the project management pane on the left to enter the variable table interface.



3. Right-click the variable table and click "Export HMI Monitoring Variable Table".



4. In the window that is displayed, set an archive path and file name, and then click "Save" to complete the export of the variable table.
5. (Optional) To export all customized variables, right-click "Global Variables" and select "Export As HMI Variable".



Downloading the PLC program

After exporting the variable table, click the download button to download the program to the PLC.

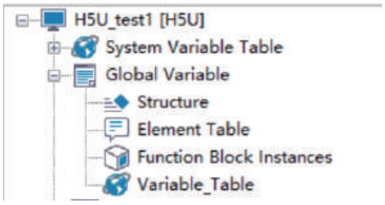
Setting the PLC IP address

For setting the PLC IP address, see "2.2 Communication Connection".

20.4.3.2 Accessing Customized Variables Through Mapping Address

1. Create a project and write a PLC program.
2. Allocate variable addresses.

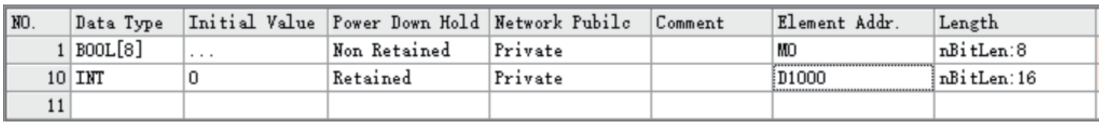
a. In the project management pane, double-click "Variable Table" to enter the variable table interface.



- H5U_test1 [H5U]
  - System Variable Table
  - Global Variable
    - Structure
    - Element Table
    - Function Block Instances
    - Variable_Table

b. Allocate soft element addresses for custom variables.

The light variable is a BOOL-type array and occupies M0 to M7, a total of eight bits, after being mapped to M0.



NO.	Data Type	Initial Value	Power Down Hold	Network Pubilc	Comment	Element Addr.	Length
1	BOOL[8]	...	Non Retained	Private		M0	nBitLen:8
10	INT	0	Retained	Private		D1000	nBitLen:16
11							

c. Compile the project to automatically generate the allocated addresses.

Click the compile button to compile the program. After the compilation is completed, the software automatically generates the allocated addresses.

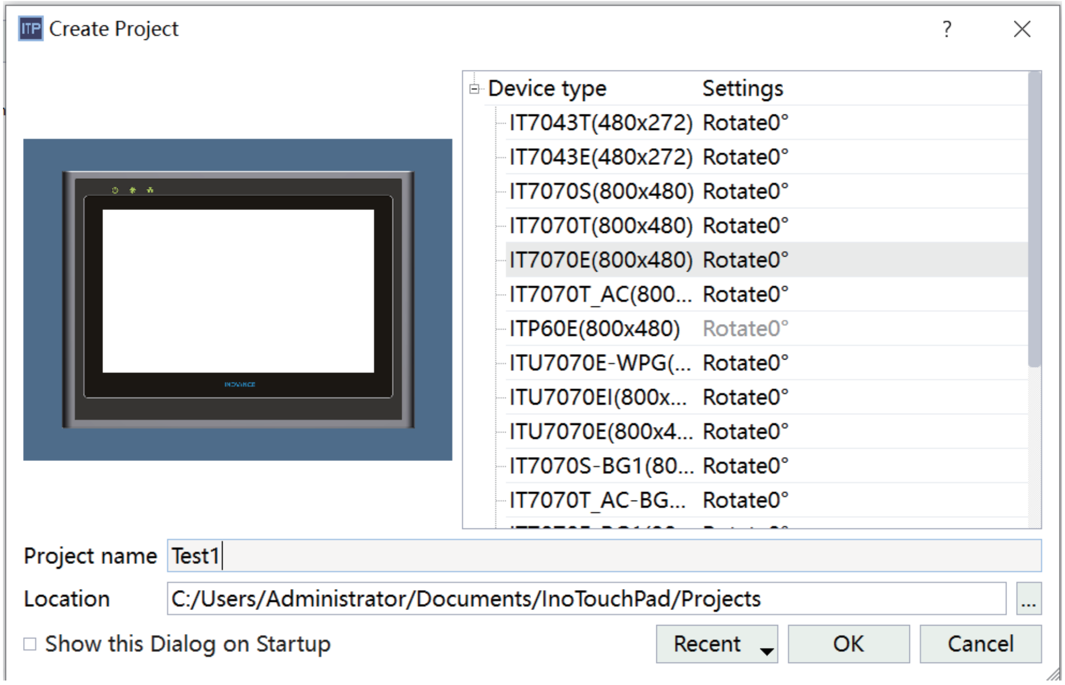
3. Download the PLC program and set the PLC IP address.

20.4.4 HMI Configuration

20.4.4.1 Accessing Customized Variables Through HMI Tag Communication

1 Creating an HMI project

Open InoTouchPad. Create a project. Set an archive path, name, and device type for the project, and then click "OK".



Create Project

Device type	Settings
IT7043T(480x272)	Rotate0°
IT7043E(480x272)	Rotate0°
IT7070S(800x480)	Rotate0°
IT7070T(800x480)	Rotate0°
IT7070E(800x480)	Rotate0°
IT7070T_AC(800...	Rotate0°
ITP60E(800x480)	Rotate0°
ITU7070E-WPG(...	Rotate0°
ITU7070EI(800x...	Rotate0°
ITU7070E(800x4...	Rotate0°
IT7070S-BG1(80...	Rotate0°
IT7070T_AC-BG...	Rotate0°

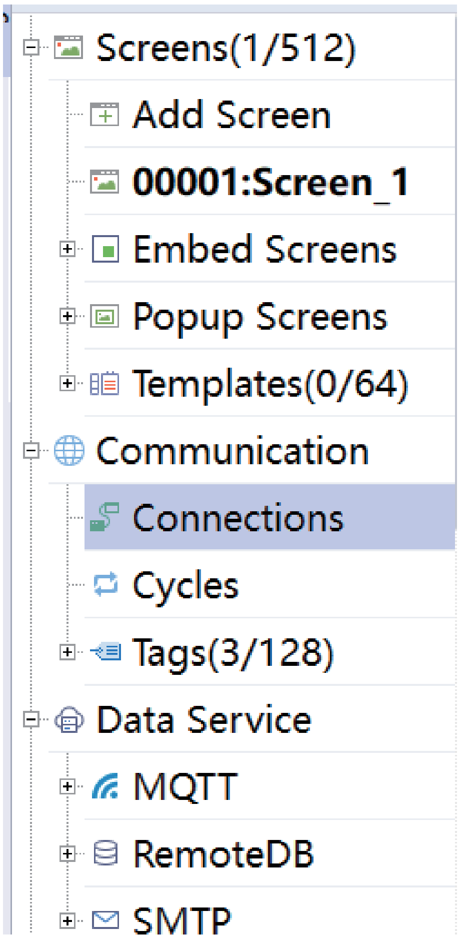
Project name: Test1

Location: C:/Users/Administrator/Documents/InoTouchPad/Projects

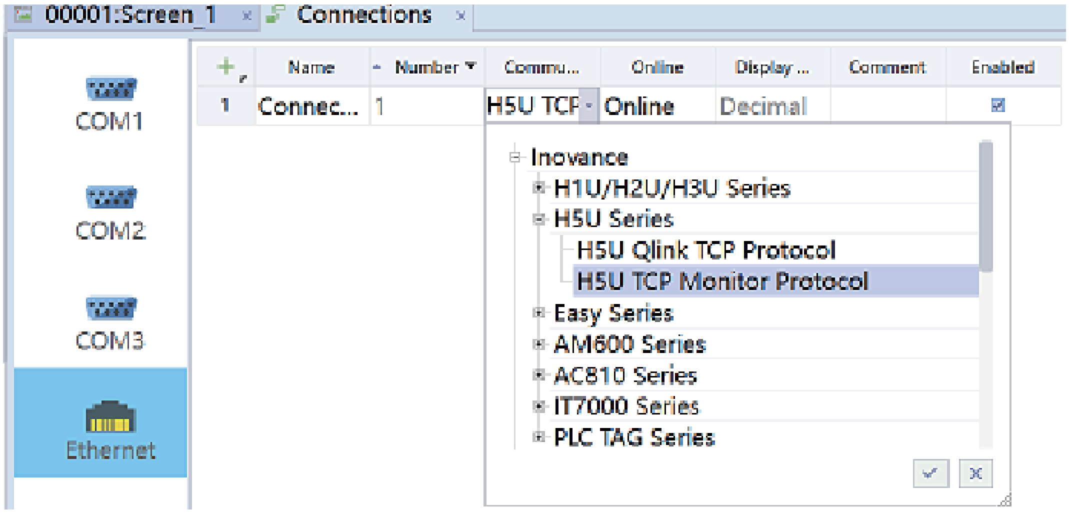
□ Show this Dialog on Startup

2 Creating a communication connection

1. In the project management pane on the left, double-click the connection tab.

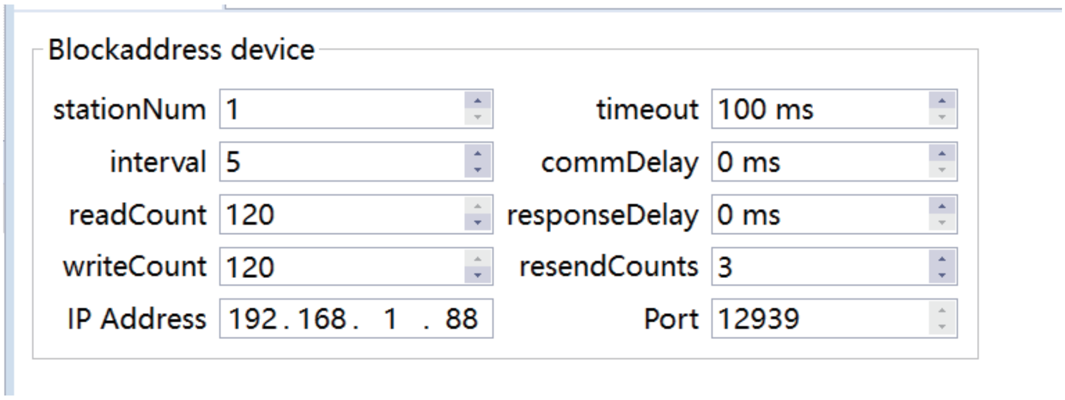


2. On the connection management page, click the "+" icon to add a connection, and select the H5U TCP monitoring protocol.



+	Name	Number	Commu...	Online	Display ...	Comment	Enabled
1	Connec...	1	H5U TCP	Online	Decimal		☑

3. Enter the IP address of the connected PLC to complete the setup.

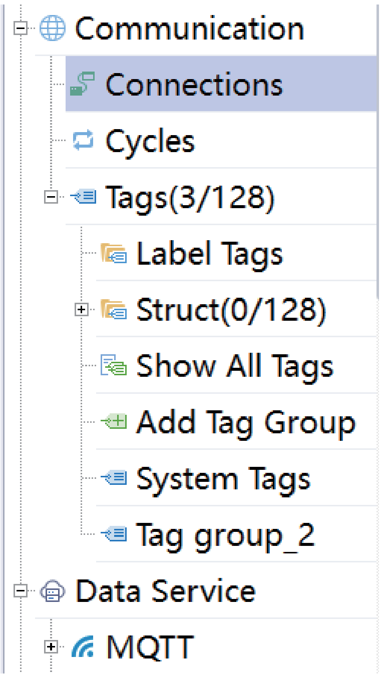


Blockaddress device

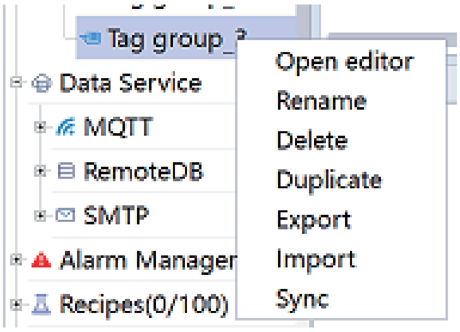
stationNum	1	timeout	100 ms
interval	5	commDelay	0 ms
readCount	120	responseDelay	0 ms
writeCount	120	resendCounts	3
IP Address	192 . 168 . 1 . 88	Port	12939

3 Adding variables

1. Click "Variable" to expand the variable menu. Click "Add variable group" to add a variable group.

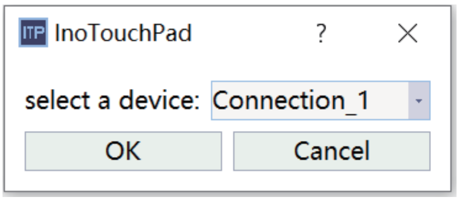


2. Right-click the newly added variable group and select "Import".



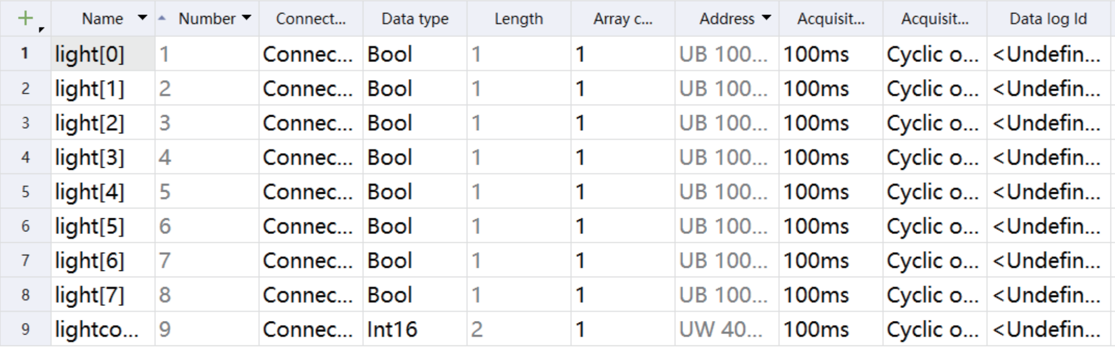
3. In the pop-up window, select the variable table exported from the PLC, and click "Open".

4. Select the created H5U connection and click "OK".



select a device: Connection_1

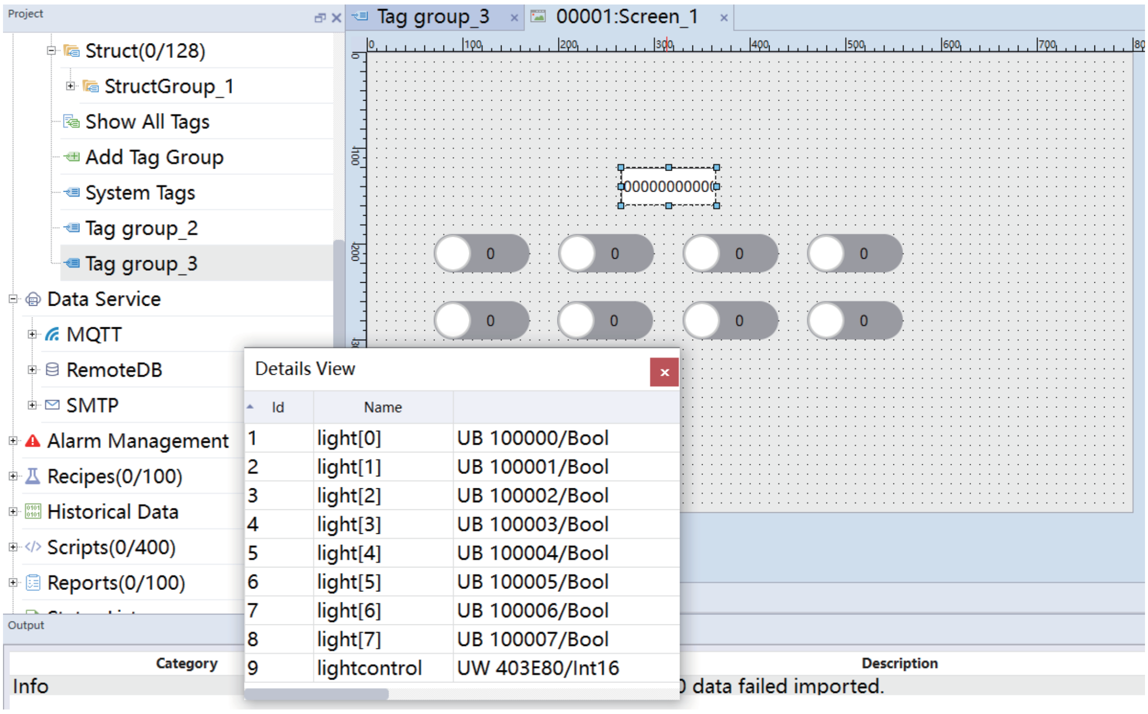
5. The variables are successfully added to the HMI variable table.



	Name	Number	Connect...	Data type	Length	Array c...	Address	Acquisit...	Acquisit...	Data log Id
1	light[0]	1	Connec...	Bool	1	1	UB 100...	100ms	Cyclic o...	<Undefin...
2	light[1]	2	Connec...	Bool	1	1	UB 100...	100ms	Cyclic o...	<Undefin...
3	light[2]	3	Connec...	Bool	1	1	UB 100...	100ms	Cyclic o...	<Undefin...
4	light[3]	4	Connec...	Bool	1	1	UB 100...	100ms	Cyclic o...	<Undefin...
5	light[4]	5	Connec...	Bool	1	1	UB 100...	100ms	Cyclic o...	<Undefin...
6	light[5]	6	Connec...	Bool	1	1	UB 100...	100ms	Cyclic o...	<Undefin...
7	light[6]	7	Connec...	Bool	1	1	UB 100...	100ms	Cyclic o...	<Undefin...
8	light[7]	8	Connec...	Bool	1	1	UB 100...	100ms	Cyclic o...	<Undefin...
9	lightco...	9	Connec...	Int16	2	1	UW 40...	100ms	Cyclic o...	<Undefin...

4 Configuring HMI

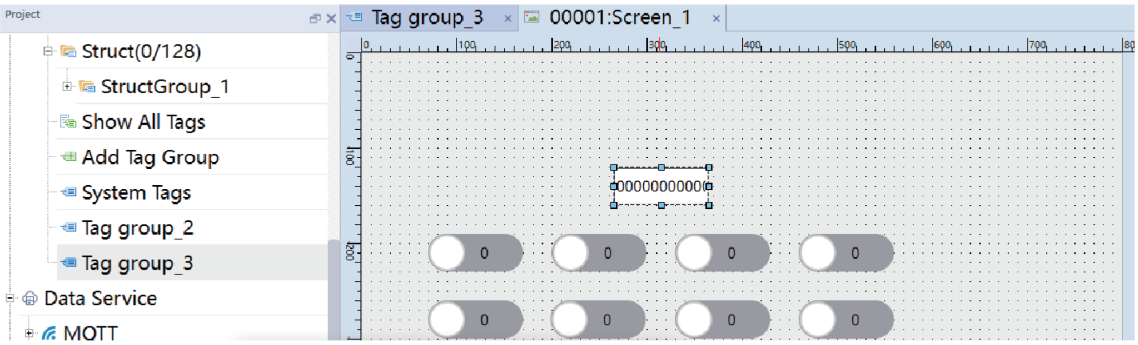
The HMI allows you to directly drag variables for programming. Drag variables from the detailed view to the programming interface one by one to complete programming.



Details View

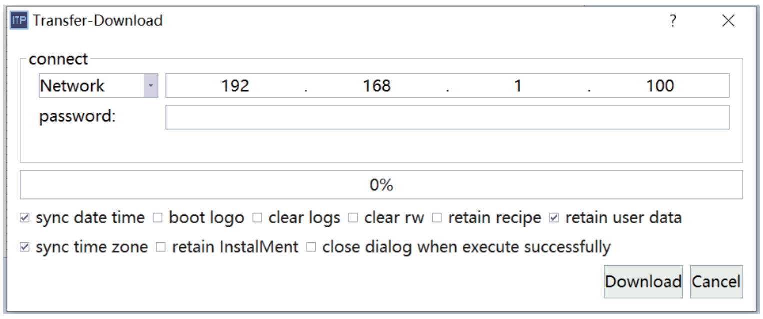
Id	Name	
1	light[0]	UB 100000/Bool
2	light[1]	UB 100001/Bool
3	light[2]	UB 100002/Bool
4	light[3]	UB 100003/Bool
5	light[4]	UB 100004/Bool
6	light[5]	UB 100005/Bool
7	light[6]	UB 100006/Bool
8	light[7]	UB 100007/Bool
9	lightcontrol	UW 403E80/Int16

Double-click the value I/O field, set the display format to binary, and set the character field length to eight bits for easy observation.



Downloading the HMI program

Click ⬇. In the window that is displayed, set the target HMI IP and click "Download". Wait for the progress bar to reach 100% to complete the HMI program download.



Transfer-Download

connect

Network 192 . 168 . 1 . 100

password:

0%

☑ sync date time □ boot logo □ clear logs □ clear rw □ retain recipe ☑ retain user data

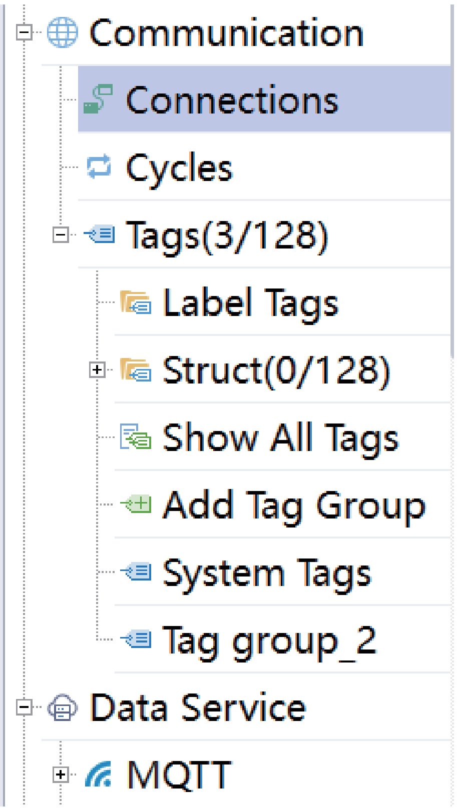
☑ sync time zone □ retain InstalMent □ close dialog when execute successfully

Download Cancel

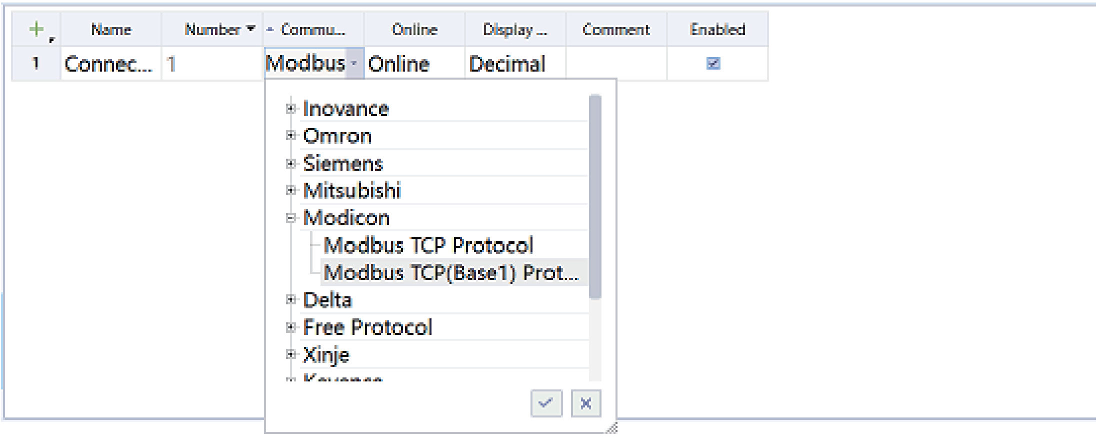
20.4.4.2 Accessing Customized Variables Through Mapping Address

1. Create an HMI project.
2. Create a communication connection.

a. In the project management pane on the left, double-click the connection tab.

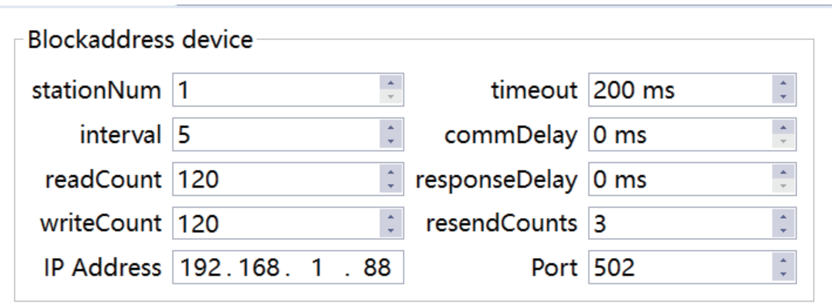


b. On the connection management page, click the "+" icon to add a connection, and select the Modbus-TCP monitoring protocol.



+	Name	Number	Commu...	Online	Display ...	Comment	Enabled
1	Connec...	1	Modbus	Online	Decimal		☑

c. Enter the IP address of the connected PLC and set the port number to 502 to complete the setup.



Blockaddress device

stationNum	1	timeout	200 ms
interval	5	commDelay	0 ms
readCount	120	responseDelay	0 ms
writeCount	120	resendCounts	3
IP Address	192 . 168 . 1 . 88	Port	502

3. Add variables.

a. Click "Variable" to expand the variable menu. Click "Add variable group" to add a variable group.

b. Double-click the newly created variable group to open it.

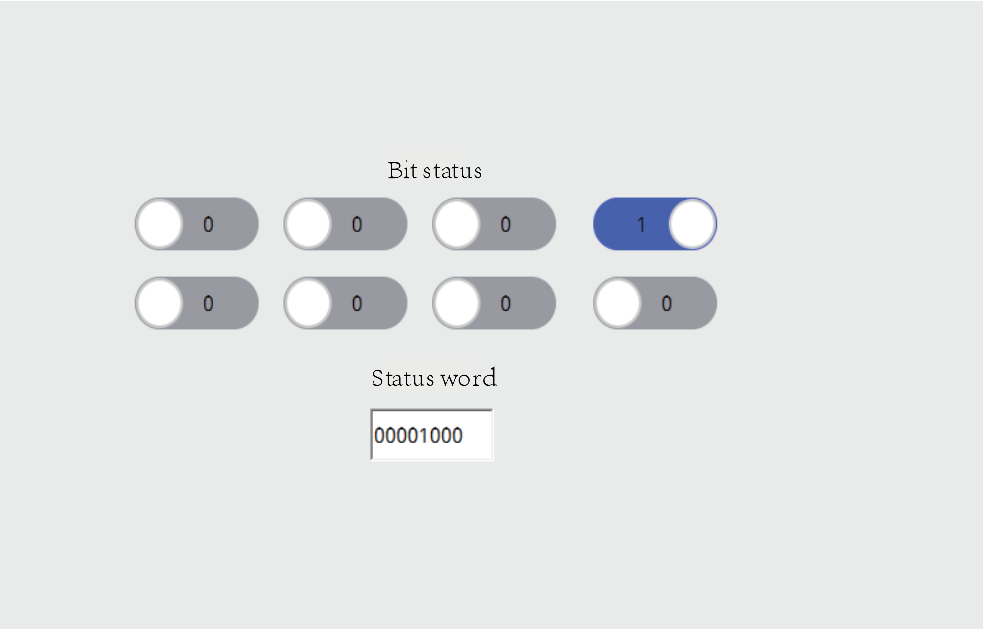
c. Add variables to be monitored.

Addresses of the added variables depend on the addresses allocated in the PLC.

4. Configure HMI and download the HMI project.

20.4.5 Example Running Results

After communication is established through the tag communication function, the bit states of customized variables and the corresponding status words can be monitored through the HMI.



00001000

19012249A11

Shenzhen Inovance Technology Co., Ltd.

www.inovance.com

Add.: Inovance Headquarters Tower, High-tech Industrial Park,
Guanlan Street, Longhua New District, Shenzhen

Tel: (0755) 2979 9595 Fax: (0755) 2961 9897

Suzhou Inovance Technology Co., Ltd.

www.inovance.com

Add.: No. 16 Youxiang Road, Yuexi Town,
Wuzhong District, Suzhou 215104, P.R. China

Tel: (0512) 6637 6666 Fax: (0512) 6285 6720