



# 中型 PLC 编程软件 使用手册



工业自动化



智能电梯



新能源汽车



工业机器人



轨道交通



资料编码 19010334 B07

# 前言

## 资料简介

本手册面向汇川中型可编程逻辑控制器（以下简称中型PLC），涵盖AM、AC和AP等系列。主要介绍中型PLC编程软件InoProShop对控制器编程时所需的网络配置、编程环境、编程语言、程序诊断等相关知识。

## 更多资料

资料编码	资料名称	内容简介
19010539	中型PLC编程手册（运动控制篇）	介绍中型PLC运动控制系统组成、运动控制程序机制、MC指令详解、仿真与调试相关操作。
19011700	中型PLC指令手册	介绍中型PLC的基本指令。
PS00017338	中型PLC编程手册（机器人集成控制篇）	介绍中型PLC编程软件InoProShop对控制器机器人编程部分功能使用相关知识。
PS00003145	中型PLC编程软件使用手册（本手册）	介绍中型PLC软件的基本功能、快速入门、网络配置、编程基础等内容。

## 版本变更记录

修订日期	发布版本	变更内容
2024-05	B07	<ul style="list-style-type: none"><li>新增第68页“3.11 多人协同”章节。</li><li>新增第70页“3.12 工程比较”章节。</li><li>新增第76页“3.13 微分监视器”章节。</li><li>新增第80页“3.14 指令配置与调试”章节。</li><li>在第101页“4.2.4 IO 模块配置”和第198页“4.4.10 IO 模块”章节新增适配GL20S系列扩展模块。</li><li>在第179页“4.4.5 CiA402 轴”章节新增“绝对值编码器”内容。</li><li>在第310页“4.11.3 PLC 作为EtherNet/IP 主站的配置”章节新增“禁用/启用EtherNet/IP远程从站”内容。</li><li>新增第346页“FINS TCP&amp;UDP通信”章节。</li><li>新增第475页“8.4 CPU实时负载超限”章节。</li><li>...</li></ul>
2023-12	B06	<p><b>新增内容：</b></p> <ul style="list-style-type: none"><li>新增第58页“工程安全管理”章节。</li><li>新增第66页“Application添加对象”章节。</li></ul> <p><b>修改内容：</b></p> <ul style="list-style-type: none"><li>更新第514页“9.8.8 轴诊断码”章节。</li></ul>

修订日期	发布版本	变更内容
2023-12	B05	<p><b>新增内容:</b></p> <ul style="list-style-type: none"> <li>• 新增第56页“3.8 工程版本升级”章节。</li> <li>• 新增第142页“4.2.6 IO映射参数说明”章节。</li> <li>• 新增第145页“扩展卡配置”章节。</li> <li>• 在第198页“4.4.10 IO 模块”章节中新增“禁用GL20系列IO模块”内容。</li> <li>• 新增第290页“CAN自由协议”章节。</li> <li>• 在第325页“4.11.5 PLC 作为EtherNet-IP 从站的配置”章节中新增“全局变量表EIP 配置”内容。</li> <li>• 新增第382页“5.5.2 变量定义”章节。</li> <li>• 新增第385页“5.5.4 掉电保持规则”章节。</li> <li>• 在第420页“6.3.6 LD 菜单命令”章节中新增“批量更新运算块”和“更新运算块调用”内容。</li> <li>• 新增第520页“SVN功能”章节。</li> </ul> <p><b>修改内容:</b></p> <ul style="list-style-type: none"> <li>• 在第101页“4.2.4 IO 模块配置”章节增加GL20系列模块配置内容。</li> <li>• 更新第125页“4.2.5 高速IO 配置”章节内容。</li> <li>• 在第146页“4.4.2 常用功能”章节中优化“显示设备差异”和“拷贝扫描设备时别名地址是否生效”内容。</li> <li>• 更新第442页“7.3 故障诊断”章节内容。</li> <li>• 更新第512页“9.8.7 EtherCAT 诊断码”章节内容。</li> </ul>
2023-08	B04	<ul style="list-style-type: none"> <li>• 新增用户程序升级指导说明。</li> <li>• 优化部分功能描述。</li> </ul>
2022-09	B03	<ul style="list-style-type: none"> <li>• 新增在线诊断和轴诊断功能。</li> <li>• 新增轴诊断码。</li> <li>• 更新EtherCAT诊断码。</li> </ul>
2022-01	B02	<ul style="list-style-type: none"> <li>• 新增ST编程、编译等相关功能。</li> <li>• 优化EIP、变量定义等功能。</li> </ul>
2020-11	B01	<ul style="list-style-type: none"> <li>• 6.6小节新增HMC_Reset, HMC_TouchProbe。</li> <li>• 对上一版本进行细节勘误。</li> </ul>
2020-10	B00	<ul style="list-style-type: none"> <li>• 增加“3.9 EtherNET通讯”。</li> <li>• 增加“附录I 同步工程信息”。</li> <li>• 增加“附录J PLC运行异常处理方法指导”。</li> <li>• 对上一版本进行细节勘误。</li> </ul>
2018-04	A00	第一版发行

## 关于手册获取

本手册不随产品发货，如需获取电子版PDF文件，可以通过以下方式获取：

- 登录汇川技术官方网站（[www.inovance.com](http://www.inovance.com)），“服务与支持-资料下载”，搜索关键字并下载。
- 使用手机扫产品机身二维码，获取产品配套手册。

## 保修声明

正常使用情况下，产品发生故障或损坏，汇川技术提供保修期内的保修服务（产品保修期请详见订货单）。超过保修期，将收取维修费用。

保修期内，以下情况造成的产品损坏，将收取维修费用。

- 不按手册中的规定操作本产品，造成的产品损坏。
- 火灾、水灾、电压异常，造成的产品损坏。

- 将本产品用于非正常功能，造成的产品损坏。
- 超出产品规定的使用范围，造成的产品损坏。
- 不可抗力（自然灾害、地震、雷击）因素引起的产品二次损坏。

有关服务费用按照厂家统一标准计算，如有契约，以契约优先的原则处理。

详细保修说明请参见《产品保修卡》。

# 目录

前言 .....	1
1 产品简介 .....	10
1.1 概述 .....	10
1.1.1 产品简介 .....	10
1.1.2 产品配置及模块说明 .....	13
1.1.3 系统应用流程 .....	17
1.2 InoProShop 概述 .....	18
1.2.1 InoProShop 简介 .....	18
1.2.2 InoProShop 与硬件的连接 .....	19
1.2.3 软件获取与安装 .....	19
1.2.4 安装步骤 .....	20
1.2.5 卸载InoProShop .....	25
2 快速入门 .....	26
2.1 启动编程环境 .....	26
2.2 编写用户程序的典型步骤 .....	27
2.2.1 概述 .....	27
2.2.2 用户系统的配置操作 .....	28
2.2.3 用户程序的编写操作 .....	29
2.2.4 用户程序变量与端口的关联配置 .....	30
2.2.5 配置用户程序的执行方式和运行周期 .....	30
2.2.6 用户程序的编译、登录下载 .....	30
2.3 用InoProShop 编写一个跑马灯样例工程 .....	33
2.4 如何登录主模块 .....	38
2.4.1 登录主模块的必备条件与操作简介 .....	38
2.4.2 在InoProShop 中扫描中型PLC 网络设备 .....	38
2.4.3 扫描不到设备的处理对策 .....	41
3 基本功能 .....	43
3.1 界面导航 .....	43
3.2 编译命令 .....	43
3.3 资源使用表 .....	44
3.3.1 概述 .....	44
3.3.2 功能介绍 .....	44
3.4 符号配置 .....	48
3.5 交叉引用 .....	52
3.6 监控表 .....	52
3.7 转到下一层 .....	53
3.8 工程版本升级 .....	56
3.9 工程安全管理 .....	58
3.9.1 加密工程文件 .....	58
3.9.2 工程用户权限管理 .....	60
3.10 Application添加对象 .....	66
3.11 多人协同 .....	68
3.12 工程比较 .....	70
3.13 微分监视器 .....	76

---

3.14 指令配置与调试.....	80
<b>4 网络配置 .....</b>	<b>86</b>
<b>4.1 设备组态 .....</b>	<b>86</b>
4.1.1 设备组态 .....	86
4.1.2 网络组态 .....	86
4.1.3 硬件组态 .....	91
4.1.4 设备树操作.....	93
4.1.5 组态编译错误定位 .....	93
<b>4.2 CPU 配置 .....</b>	<b>94</b>
4.2.1 概述.....	94
4.2.2 CPU配置的一般过程.....	94
4.2.3 CPU 参数配置 .....	95
4.2.4 IO 模块配置 .....	101
4.2.5 高速IO 配置 .....	125
4.2.6 IO映射参数说明 .....	142
<b>4.3 扩展卡配置 .....</b>	<b>145</b>
<b>4.4 EtherCAT 配置 .....</b>	<b>145</b>
4.4.1 概述.....	145
4.4.2 常用功能 .....	146
4.4.3 EtherCAT 主站 .....	157
4.4.4 EtherCAT 从站 .....	165
4.4.5 CiA402 轴 .....	179
4.4.6 虚轴.....	191
4.4.7 GR10-4PME 定位模块.....	192
4.4.8 GR10-2HCE 计数模块.....	194
4.4.9 分支器 .....	196
4.4.10 IO 模块 .....	198
4.4.11 库 (隐含变量) .....	201
<b>4.5 Modbus 设备编辑器 .....</b>	<b>212</b>
4.5.1 串行硬件端口.....	212
4.5.2 网络组态 .....	213
4.5.3 Modbus 主站配置 .....	218
4.5.4 Modbus 主站通信配置.....	220
4.5.5 Modbus 主站广播配置.....	223
4.5.6 Modbus 从站配置 .....	224
4.5.7 Modbus设备诊断.....	225
4.5.8 Modbus常见故障.....	226
4.5.9 Modbus 变量编址 .....	226
4.5.10 Modbus通信帧格式说明.....	228
<b>4.6 串口自由协议使用 .....</b>	<b>230</b>
4.6.1 概要.....	230
4.6.2 串口配置 .....	231
4.6.3 通讯配置 .....	233
4.6.4 数据发送与接收寄存器.....	233
4.6.5 串口调试助手模拟通讯.....	235
<b>4.7 Modbus TCP 设备编辑器 .....</b>	<b>236</b>
4.7.1 概述.....	236
4.7.2 Modbus TCP主站配置 .....	236
4.7.3 Modbus TCP 主站通信配置 .....	237
4.7.4 Modbus TCP 从站配置 .....	240
4.7.5 Modbus TCP 设备诊断 .....	241
4.7.6 Modbus TCP 常见故障 .....	243

4.7.7 Modbus TCP 变量编址 .....	243
4.7.8 Modbus TCP 通信帧格式说明.....	245
4.8 CANopen 网络 .....	249
4.8.1 CANopen 通信简介.....	249
4.8.2 CANopen 主站配置.....	253
4.8.3 CANopen 从站配置.....	257
4.8.4 CANopen模块.....	269
4.8.5 CANopen参数配置 .....	269
4.8.6 编程接口 .....	272
4.9 CANlink 3.0 配置编辑器 .....	272
4.9.1 概述.....	272
4.9.2 CANlink3.0 网络组成 .....	273
4.9.3 CANlink 一般使用流程.....	274
4.9.4 CANlink 网络配置 .....	274
4.9.5 网络管理 .....	277
4.9.6 发送配置 .....	278
4.9.7 接收配置 .....	282
4.9.8 主站同步写 .....	283
4.9.9 本地从站配置.....	285
4.9.10 设备接入CANlink3.0 网络 .....	285
4.10 CAN自由协议 .....	290
4.10.1 概述 .....	290
4.10.2 网络组态 .....	290
4.10.3 配置CAN自由协议 .....	293
4.10.4 CANBus库 .....	295
4.10.4.1 CANBus库枚举类型 .....	295
4.10.4.2 CANBus库结构体类型 .....	297
4.10.4.3 CANBus功能块.....	298
4.10.4.4 CANBus功能块错误码 .....	302
4.10.4.5 CANBus功能块使用示例.....	306
4.11 EtherNet-IP 通信 .....	308
4.11.1 协议概述 .....	308
4.11.2 EtherNet-IP通信规格 .....	309
4.11.3 PLC 作为EtherNet-IP 主站的配置 .....	310
4.11.4 PLC 作为EtherNet-IP 主站的配置例程 .....	319
4.11.5 PLC 作为EtherNet-IP 从站的配置 .....	325
4.11.6 PLC 作为EtherNet-IP 从站的配置例程 .....	333
4.11.7 EtherNet-IP通讯状态诊断 .....	336
4.12 Profibus-DP 总线.....	339
4.12.1 概述 .....	339
4.12.2 Profibus-DP 使用的一般过程 .....	340
4.12.3 Profibus-DP 主站配置 .....	341
4.12.4 Profibus-DP 从站配置 .....	342
4.12.5 Profibus-DP 模块 .....	345
4.13 FINS TCP&UDP通信 .....	346
4.13.1 概述 .....	346
4.13.2 FINS TCP&UDP通信配置 .....	349
4.14 与HMI 通信配置 .....	353
4.14.1 通信配置 .....	353
4.14.2 通信示例 .....	355
4.14.3 常见故障分析 .....	357
5 编程基础 .....	360

---

5.1 概述 .....	360
5.2 直接地址 .....	360
5.2.1 定义语法 .....	360
5.2.2 PLC 直接地址存储区域 .....	361
5.3 变量 .....	361
5.3.1 概述 .....	361
5.3.2 变量定义 .....	362
5.3.3 变量类型 .....	374
5.3.4 变量导入与导出 .....	379
5.4 常量 .....	380
5.5 掉电保持变量 .....	382
5.5.1 概述 .....	382
5.5.2 变量定义 .....	382
5.5.3 掉电保持变量表 .....	384
5.5.4 掉电保持规则 .....	385
5.5.5 掉电保持模式 .....	386
5.5.6 地址分配 .....	387
5.5.7 配方操作 .....	390
5.5.8 使用说明 .....	393
6 编程语言 .....	395
6.1 InoProShop 支持的编程语言简介 .....	395
6.2 结构化文本语言(ST) .....	395
6.2.1 概述 .....	395
6.2.2 表达式 .....	395
6.2.3 ST 指令 .....	396
6.2.4 ST 编辑 .....	402
6.2.4.1 ST 工具箱 .....	402
6.2.4.2 智能输入 .....	403
6.2.4.3 折叠和缩放功能 .....	404
6.2.4.4 IEC 文本编辑器界面颜色 .....	404
6.3 梯形图(LD) .....	407
6.3.1 概述 .....	407
6.3.2 梯形图元素 .....	408
6.3.3 LD 编辑器选项 .....	411
6.3.4 元素选择 .....	414
6.3.5 标准编辑命令 .....	416
6.3.6 LD 菜单命令 .....	420
6.3.7 单键命令 .....	430
6.3.8 划线功能 .....	431
6.3.9 拖拽操作 .....	433
6.3.10 图形显示工具 .....	435
6.3.11 LD 调试 .....	436
6.3.12 梯形图数据更新 .....	439
7 诊断 .....	440
7.1 诊断简介 .....	440
7.2 组态诊断 .....	440
7.2.1 概述 .....	440
7.2.2 网络组态诊断 .....	440
7.2.3 硬件组态诊断 .....	441
7.3 故障诊断 .....	442

7.4 在线诊断 .....	446
7.4.1 概述 .....	446
7.4.2 诊断流程 .....	446
7.4.3 扫描设备 .....	447
7.4.4 登录PLC .....	447
7.5 设备自身诊断信息列表 .....	449
7.5.1 CPU 诊断 .....	449
7.5.2 EtherCAT 诊断 .....	449
7.5.3 IO 诊断 .....	450
7.5.4 Profibus-DP 诊断 .....	450
7.5.5 ModbusRTU 诊断 .....	454
7.5.6 ModbusTCP 诊断 .....	454
7.5.7 CANlink 诊断 .....	454
7.6 诊断编程接口 .....	455
7.6.1 概述 .....	455
7.6.2 诊断编程接口简介 .....	455
7.6.3 CPU 诊断编程接口 .....	456
7.6.4 CANopen 诊断编程接口 .....	458
7.6.5 Profibus-DP 诊断编程接口 .....	458
7.6.6 CANlink 诊断编程接口 .....	460
7.6.7 ModbusRTU 诊断编程接口 .....	461
7.6.8 ModbusTCP 诊断编程接口 .....	461
7.6.9 EtherCAT 诊断编程接口 .....	461
7.6.10 CPU 停止控制 .....	461
7.6.11 轴诊断 .....	462
8 FAQ .....	465
8.1 CPU 占有率过高 .....	465
8.1.1 CPU 占有率定义 .....	465
8.1.2 分析步骤 .....	465
8.1.3 常见优化方式 .....	466
8.2 PLC运行异常 .....	466
8.2.1 概述 .....	466
8.2.2 现象描述 .....	466
8.2.3 原因分析及解决方法 .....	467
8.3 获取文件夹失败 .....	474
8.4 CPU实时负载超限 .....	475
9 附录 .....	477
9.1 各通信端口的通信协议简介 .....	477
9.1.1 概述 .....	477
9.1.2 Mini-USB 端口及其内置通信协议 .....	477
9.1.3 COM通信端口及其内置协议 .....	477
9.1.4 CANopen 通信协议 .....	478
9.1.5 CANlink通信协议 .....	478
9.1.6 Ethernet 端口及通信协议 .....	478
9.1.7 EtherCAT 端口及通信协议 .....	478
9.1.8 高速IO 接口 .....	479
9.1.9 Mini-SD 卡插槽 .....	479
9.1.10 本地总线扩展接口 .....	479
9.1.11 Profibus-DP端口 .....	479
9.2 软元件概述 .....	479
9.3 基本指令速查表 .....	480

---

9.4 PLC编程软件升级 .....	484
9.4.1 版本说明 .....	484
9.4.2 升级方法 .....	484
9.4.3 常见问题 .....	487
9.5 PLC用户程序升级 .....	494
9.5.1 通过InoProShop工具升级 .....	494
9.5.2 通过SD卡升级 .....	494
9.6 AM400/600高速I/O接线指导 .....	496
9.7 高速IO兼容性 .....	500
9.7.1 新旧界面介绍 .....	500
9.7.2 高速IO诊断 .....	502
9.8 诊断码和诊断信息 .....	506
9.8.1 概述 .....	506
9.8.2 CPU 诊断码 .....	507
9.8.3 IO 模块诊断码 .....	508
9.8.4 DP 诊断码 .....	509
9.8.5 CANlink 诊断码 .....	510
9.8.6 Modbus 诊断码 .....	511
9.8.7 EtherCAT 诊断码 .....	512
9.8.8 轴诊断码 .....	514
9.9 同步工程信息 .....	519
9.9.1 概述 .....	519
9.9.2 自动下载同步工程信息 .....	519
9.9.3 手动下载同步工程信息 .....	520
9.9.4 同步工程信息的特殊说明 .....	520
9.10 SVN功能 .....	520
9.10.1 概述 .....	520
9.10.2 搭建SVN服务器和SVN库 .....	521
9.10.3 安装CodeMeter运行环境和SVN插件 .....	521
9.10.4 获取授权码和离线授权 .....	524
9.10.5 SVN操作指导 .....	529
9.10.6 卸载SVN插件 .....	533

# 1 产品简介

## 1.1 概述

### 1.1.1 产品简介

汇川中型可编程逻辑控制器（以下简称中型PLC），涵盖AM/AC/AP等系列，为用户提供智能自动化解决方案。中型PLC采用IEC61131-3编程语言体系，支持PLCopen标准编程语言。

- AM400/AM600采用机架式布局，每个机架支持本地扩展16个扩展模块，并可通过Profibus-DP、EtherCAT、CANopen等多种工业现场总线远程扩展机架。AM600本地扩展模块通过内部总线协议进行IO扩展，支持数字输入/输出模块、模拟输入/输出模块、温度模块等多种功能模块。通过EtherCAT总线可实现高性能运动控制功能；具有单轴加减速控制功能、电子齿轮功能、电子凸轮功能，还可通过高速I/O实现单轴基本定位功能，且最高频率可达200kHz；同时支持RS485、以太网、USB等通信功能。
- AM300/AM500系列是汇川技术推出的新一代标准型中型PLC，支持EtherCAT（仅AM500支持）、EtherNet/IP、OPC UA、Modbus TCP(支持两路同一网段的IP地址)、TCP/IP、RS485(最大可扩展至3路)、CAN总线等多种通信协议。
- AC系列采用书本式结构，支持EtherCAT、EtherNet/IP、OPC UA、Modbus TCP/RTU、UDP、Socket等多种通信协议，使用灵活，满足用户多样化的应用需求。其中AC700最大支持1ms-32轴运动控制，AC800最大支持4ms-256轴运动控制。AP700系列智能机械控制器，是基于Intel Atom处理器硬件平台，符合PLCopen规范的高性能多轴运动控制器，使用高速 EtherCAT总线，可以实现多轴伺服控制，同时配有15寸工业级TFT显示触摸屏，方便用户输入控制指令或运控程序，显示内部数据，适用于先进制造业的高速生产装备和大型设备的控制，尤其是类机床智能设备的控制应用。
- AM780-N是一款具有宽温特性的中型可编程逻辑控制器，具有抗振能力强，体积小，结构紧凑，拆装和接线便捷，扩展能力强，IO控制刷新速度快，抗扰能力强以及高可靠性等优点，广泛应用于风电、冶金、户外电站和矿井等行业。

中型可编程控制器具有以下功能特点：

- 多种运动控制功能：总线运动控制、脉冲运动控制。
- 支持更多的 I/O 点数：最多可达上万点。
- 更大的程序容量和数据存储区。
- 更快的指令执行速度。
- 支持更多的高端现场总线（EtherCAT、CANopen、EtherNet/IP、Profibus-DP）。
- 更易用的软件，满足用户不同应用需求。
- 支持在线侦错模式。
- 支持在线编辑模式。

下面列出了中型PLC涵盖的CPU模块对象以及对应的差异点。

表1-1 CPU模块软件功能特性

产品型号【注1】	本地扩展模块数	程序存储空间	数据存储空间	掉电数据保存大小	运动控制轴数	高速I/O功能	软元件特性	输出类型
AM401-CPU1608TP	8	10M	20M	480K	伺服轴4个，4PME 4个	16入8出高速I/O	✓	源型输出
AM402-CPU1608TP	8	10M	20M	480K	伺服轴8个，4PME 4个	16入8出高速I/O	✓	源型输出
AM401-CPU1608TN	8	10M	20M	480K	伺服轴4个，4PME 4个	16入8出高速I/O	✓	漏型输出

产品型号【注1】	本地扩展模块数	程序存储空间	数据存储空间	掉电数据保存大小	运动控制轴数	高速I/O功能	软元件特性	输出类型
AM402-CPU1608TN	8	10M	20M	480K	伺服轴8个, 4PME 4个	16入8出高速I/O	✓	漏型输出
AM403-CPU1608TN	16	10M	20M	480K	伺服轴16个, 4PME 4个	16入8出高速I/O	✓	漏型输出
AM600-CPU1608TP	16	10M	20M	480K	最大32个(推荐20个以下)	16入8出高速I/O	✓	源型输出
AM600-CPU1608TN	16	10M	20M	480K	最大32个(推荐20个以下)	16入8出高速I/O	✓	漏型输出
AM610-CPU1608TP	16	10M	20M	480K	×	16入8出高速I/O	✓	源型输出
AC801-0221-U0R0	×	128M	128M	5M(需外接UPS)	48	×	×	×
AC802-0222-U0R0	×	128M	128M	5M(需外接UPS)	128	×	×	×
AC810-0122-U0R0	×	128M	128M	5M(需外接UPS)	256	×	×	×
AC812-0322-U0R0	×	128M	128M	5M(需外接UPS)	256	×	×	×
AP702-0221-U0R0	×	128M	128M	5M(需外接UPS)	48	×	×	×
AP703-0221-U0R0	×	128M	128M	5M(需外接UPS)	48	×	×	×
AP705-LM【注2】	×	128M	128M	5M(需外接UPS)	48	×	×	×
AC703	×	128M	128M	5M, 内置掉电保存功能, 无需外接UPS	32	8入4出高速I/O	×	漏型输出
AC702	×	128M	128M	5M, 内置掉电保存功能, 无需外接UPS	16	8入4出高速I/O	×	漏型输出
AM320-0808TN	16	10M	20M	512K	×	8入8出高速I/O	×	漏型输出
AM521-0808TN	16	10M	20M	512K	8	8入8出高速I/O	×	漏型输出
AM522-0808TN	16	10M	20M	512K	16	8入8出高速I/O	×	漏型输出
AM523-0808TN	16	10M	20M	512K	32	8入8出高速I/O	×	漏型输出
AM780-N	32	128M	128M	5MB, 内置掉电保存功能, 无需外接UPS	32	×	✓	×

表1-2 CPU 模块通讯特性

产品型号【注1】	通讯					
	EtherCAT	Profinet-DP	CANopen/ CANlink	Modbus TCP	Modbus (串口)	EtherNET/IP
AM401-CPU1608TP	1路(最多128从站)	×	1路(最多63从站)	1路(最多63从站)	1路 (最多31从站)	1路(最多64从站)
AM402-CPU1608TP	1路(最多128从站)	×	1路(最多63从站)	1路(最多63从站)	1路 (最多31从站)	1路(最多64从站)
AM401-CPU1608TN	1路(最多128从站)	×	1路(最多63从站)	1路(最多63从站)	1路 最多31从站)	1路(最多64从站)
AM402-CPU1608TN	1路(最多128从站)	×	1路(最多63从站)	1路(最多63从站)	1路 (最多31从站)	1路(最多64从站)
AM403-CPU1608TN	1路(最多128从站)	×	1路(最多63从站)	1路 (最多63从站)	2路(每路最多31从站)	1路(最多64从站)
AM600-CPU1608TP	1路(最多128从站)	×	1路(最多63从站)	1路(最多63从站)	2路 (每路最多31从站)	1路(最多64从站)
AM600-CPU1608TN	1路(最多128从站)	×	1路(最多63从站)	1路(最多63从站)	2路 (每路最多31从站)	1路(最多64从站)
AM610-CPU1608TP	×	1路(最多124从站)	×	1路(最多63从站)	2路 (每路最多31从站)	1路(最多64从站)
AC801-0221-U0R0	1路(每路最多128从站)	×	×	2路(最多128从站)	2路 (每路最多31从站)	1路(最多64从站)
AC802-0222-U0R0	2路(每路最多128从站)	×	×	2路(最多128从站)	2路 (每路最多31从站)	1路(最多64从站)
AC810-0122-U0R0	2路(每路最多128从站)	×	×	2路(最多128从站)	2路 (每路最多31从站)	1路(最多64从站)
AC812-0322-U0R0	2路(每路最多128从站)	×	×	2路(最多128从站)	2路 (每路最多31从站)	1路(最多64从站)
AP702-0221-U0R0	1路(每路最多128从站)	×	×	2路(最多128从站)	2路 (每路最多31从站)	1路(最多64从站)
AP703-0221-U0R0	1路(每路最多128从站)	×	×	2路(最多128从站)	2路 (每路最多31从站)	1路(最多64从站)
AP705-LM【注2】	1路(每路最多128从站)	×	×	2路(最多128从站)	2路 (每路最多31从站)	1路(最多64从站)
AC703	1路(每路最多128从站)	×	×	2路(最多63从站)	2路 (每路最多31从站)	1路(最多64从站)
AC702	1路(每路最多128从站)	×	×	2路(最多63从站)	2路 (每路最多31从站)	1路(最多64从站)

产品型号【注1】	通讯					
	EtherCAT	Profibus-DP	CANopen/ CANlink	Modbus TCP	Modbus (串口)	EtherNET/IP
AM320-0808TN	×	×	1路(最多30从站)	2路(作为主站，最多63从站；作为从站，最多16主站)	最多可支持3路（主单元支持1路，扩展卡可扩展2路） (每路最多31从站)	2路(作为主站，最多16从站；作为从站，最多16主站)
AM521-0808TN	1路(每路最多127从站)	×	1路(最多30从站)	2路(作为主站，最多63从站；作为从站，最多16主站)	最多可支持3路（主单元支持1路，扩展卡可扩展2路） (每路最多31从站)	2路(作为主站，最多16从站；作为从站，最多16主站)
AM522-0808TN	1路(每路最多127从站)	×	1路(最多30从站)	2路(作为主站，最多63从站；作为从站，最多16主站)	最多可支持3路（主单元支持1路，扩展卡可扩展2路） (每路最多31从站)	2路(作为主站，最多16从站；作为从站，最多16主站)
AM523-0808TN	1路(每路最多127从站)	×	1路(最多30从站)	2路(作为主站，最多63从站；作为从站，最多16主站)	最多可支持3路（主单元支持1路，扩展卡可扩展2路） (每路最多31从站)	2路(作为主站，最多16从站；作为从站，最多16主站)
AM780-N	1路(每路最多128从站)	×	1路(最多8从站)	2路(作为主站，最多63从站；作为从站，最多16主站)	最多可支持9路（主单元支持1路，扩展模块可扩展8路，即4个GL20-2S485-N模块） (每路最多31从站)	2路(作为主站，最多32从站；作为从站，最多64主站)

## 说明

- 【注1】：1)上述信息不包括电源模块及尾板;2)AM610-CPU1608TP产品已经停止维护。
- 【注2】：AP705-LM为泛机床行业专机。

### 1.1.2 产品配置及模块说明

中型可编程控制器产品丰富，用户可根据需要选择适用于具体应用的产品配置。以AM600系列PLC为例，中型PLC包含AM600-CPU1608TP架构（EtherCAT+CANopen总线型）和AM610-CPU1608TP架构（Profibus-DP总线型），典型集成示意图如下图所示：

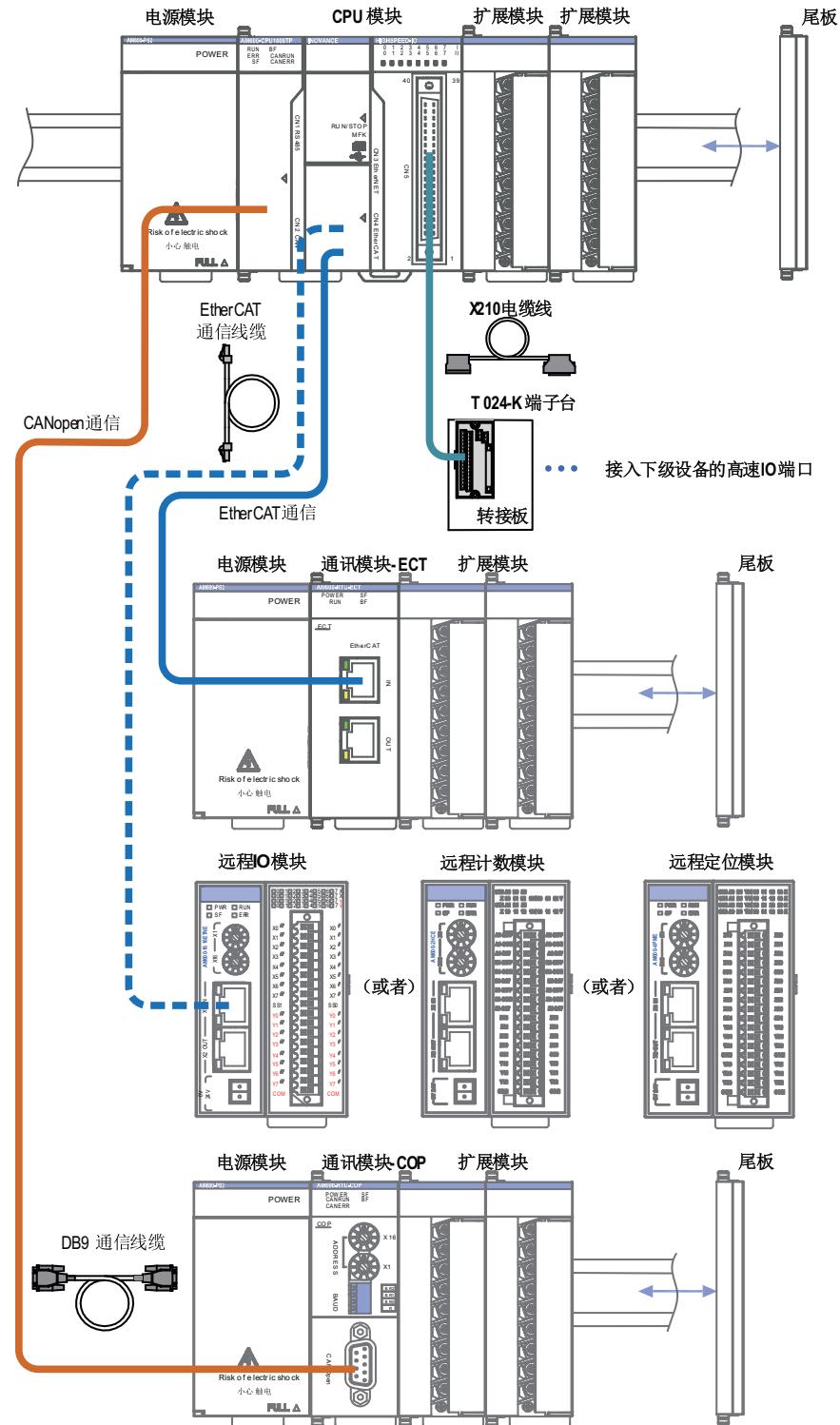


图1-1 AM600-CPU1608TP系统集成示意图

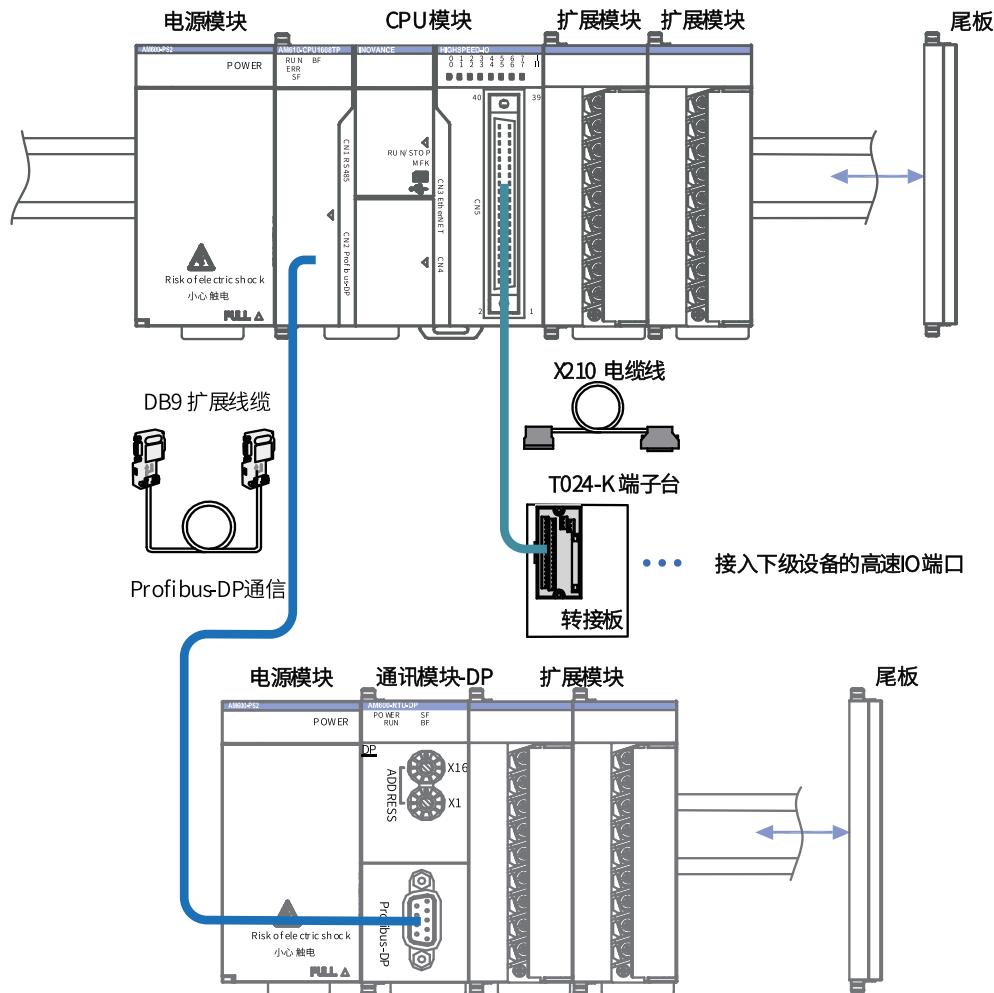


图1-2 AM610-CPU1608TP系统集成示意图

## 说明

AM610-CPU1608TP产品和AM600-RTU-DP模块产品已经停止维护。

AM600系列中型PLC按功能可分为电源模块、CPU模块、远程通信模块及扩展模块等四大类，具体包含：

订货编码	型号【注】	分类	描述
01440010	AM600-PS2	电源模块	220V电压输入，24V/2A输出
01440028	AM401-CPU1608TP	CPU模块	1路RS485，1路CANopen/CANlink，1路LAN 支持4轴运动控制，支持EtherCAT 内置16入8出高速I/O 源型输出
01440029	AM402-CPU1608TP	CPU模块	1路RS485，1路CANopen/CANlink，1路LAN 支持8轴运动控制，支持EtherCAT 内置16入8出高速I/O 源型输出

## 产品简介

订货编码	型号【注】	分类	描述
01440032	AM401-CPU1608TN	CPU模块	1路RS485, 1路CANopen/CANlink、1路LAN 支持4轴运动控制, 支持EtherCAT 内置16入8出高速I/O, 漏型输出
01440031	AM402-CPU1608TN		1路RS485, 1路CANopen/CANlink, 1路LAN 支持8轴运动控制, 支持EtherCAT 内置16入8出高速I/O 漏型输出
01440126	AM403-CPU1608TN		2路RS485, 1路CANopen/CANlink, 1路LAN 支持16轴运动控制, 支持EtherCAT 内置16入8出高速I/O 漏型输出
01440014	AM600-CPU1608TP		2路RS485, 1路CANopen/CANlink, 1路LAN 支持基本运动控制功能, 支持EtherCAT 内置16入8出高速I/O 源型输出
01440016	AM610-CPU1608TP		2路RS485, 1路LAN 支持基本运动控制功能, 支持Profibus-DP 内置16入8出高速I/O 源型输出
01440143	AC812-0322-U0R0	书本式控制器	2路USB2.0, 2路USB3.0 1路RS485/RS232, 4路LAN 支持多达256轴运动控制功能, 支持EtherCAT 多功能扩展槽, 内部Mini-PCIE扩展槽
01440038	AC810-0122-U0R0		2路USB2.0, 2路USB3.0 1路RS485/RS232, 4路LAN 支持多达128轴运动控制功能, 支持EtherCAT 多功能扩展槽, 内部Mini-PCIE扩展槽
01440101	AC802-0222-U0R0		2路USB2.0, 2路USB3.0 1路RS485/RS232, 4路LAN 支持多达48轴运动控制功能, 支持EtherCAT 内部Mini-PCIE扩展槽
01440103	AC801-0221-U0R0		2路USB2.0, 2路USB3.0 1路RS485/RS232, 支持以太网 支持多达48轴运动控制功能, 支持EtherCAT 内部Mini-PCIE扩展槽
01440066	GL10-1600END	数字输入模块	16点DI模块, 直流24V输入 (源/漏型)
01440081	GL10-0016ER	数字输出模块	16点DO模块, 继电器输出
01440069	GL10-0016ETP		16点DO模块, 晶体管输出 (源型)
01440067	GL10-0016ETN		16点DO模块, 晶体管输出 (漏型)
01440080	GL10-4AD	模拟输入模块	4通道AD模块, 支持电压/电流模拟量输入
01440071	GL10-4DA	模拟输出模块	4通道DA模块, 支持电压/电流模拟量输出
01440082	GL10-0032ETN	数字量输出模块	32点DO模块, 晶体管输出 (漏型)
01440066	GL10-3200END	数字量输入模块	32点DI模块, 直流24V输入 (源/漏型)
01440121	GL10-2PH	差分输出脉冲模块	2路高速差分输出脉冲定位模块, 输出频率1MHz, 8路普通输入
01440129	GL10-4PM	本地脉冲定位模块	4通道脉冲定位输出
01440075	GL10-4PT	温度模块	4通道热电阻温度采集, 支持多种热电阻类型
01440078	GL10-4TC	温度模块	4通道热电偶温度采集, 支持多种热电偶类型

订货编码	型号【注】	分类	描述
01440070	GL10-8TC	温度模块	8通道热电偶温度采集，支持多种热电偶类型
01440074	GL10-PS2	电源模块	220V电压输入，24V/2A输出
01440077	GR10-0808ETNE	EtherCAT从站IO模块	8点数字量输出，晶体管输出（漏型）；8点数字量输入，支持源漏型
01440255	GR10-1616ETNE	EtherCAT从站IO模块	16点数字量输出，晶体管输出（漏型）；16点数字量输入，支持源漏型
01440252	GR10-4PME	EtherCAT从站定位模块	EtherCAT从站4通道定位模块
01440279	GR10-2HCE	EtherCAT从站计数模块	EtherCAT从站2通道计数模块
01440058	GR10-4ADE	模拟量输入模块	EtherCAT从站4通道模拟量输入模块
01440060	GR10-4DAE	模拟量输出模块	EtherCAT从站4通道模拟量输出模块
01440123	GR10-4TCS-PID（已停产）	温度控制模块	4通道TC热电偶温度检测模块
01440059	GR10-8TCE	温度检测模块	8通道热电偶温度采集，支持多种热电偶类型
01440127	GR10-8PBE	EtherCAT从站探针模块	GR10系列8通道EtherCAT探针模块
01440256	GR10-2PHE	差分输出脉冲模块	支持2通道高速差分输出脉冲，可配置8个输入
01440135	GR10-EC-6SW	6路EtherCAT分支模块	1路EtherCAT输入，5路EtherCAT输出
01440177	GR10-1616ERE-BD	扩展板卡	GR10系列可编程控制器16点输入16点输出EtherCAT从站板卡
01440012	AM600-RTU-DP	DP通讯模块	Profibus-DP协议通讯接口模块
01440073	GL10-RTU-ECTA	EtherCAT通讯模块	EtherCAT协议通讯接口模块
01440013	AM600-RTU-ECTA	EtherCAT通讯模块	EtherCAT协议通讯接口模块
01440083	GL10-RTU-COP	CANopen通讯模块	CANopen协议通讯接口模块

## 说明

原AM600系列的扩展模块已升级更新为GL10/GR10系列，升级前后产品兼容；AM610-CPU1608TP产品和AM600-RTU-DP模块产品已经停止维护。

### 1.1.3 系统应用流程

中型可编程逻辑控制器的基本应用流程如下图所示，模块的安装及配线操作指导请参见《AM600系列可编程逻辑控制器硬件手册》和《AC810系列可编程逻辑控制器硬件手册》。

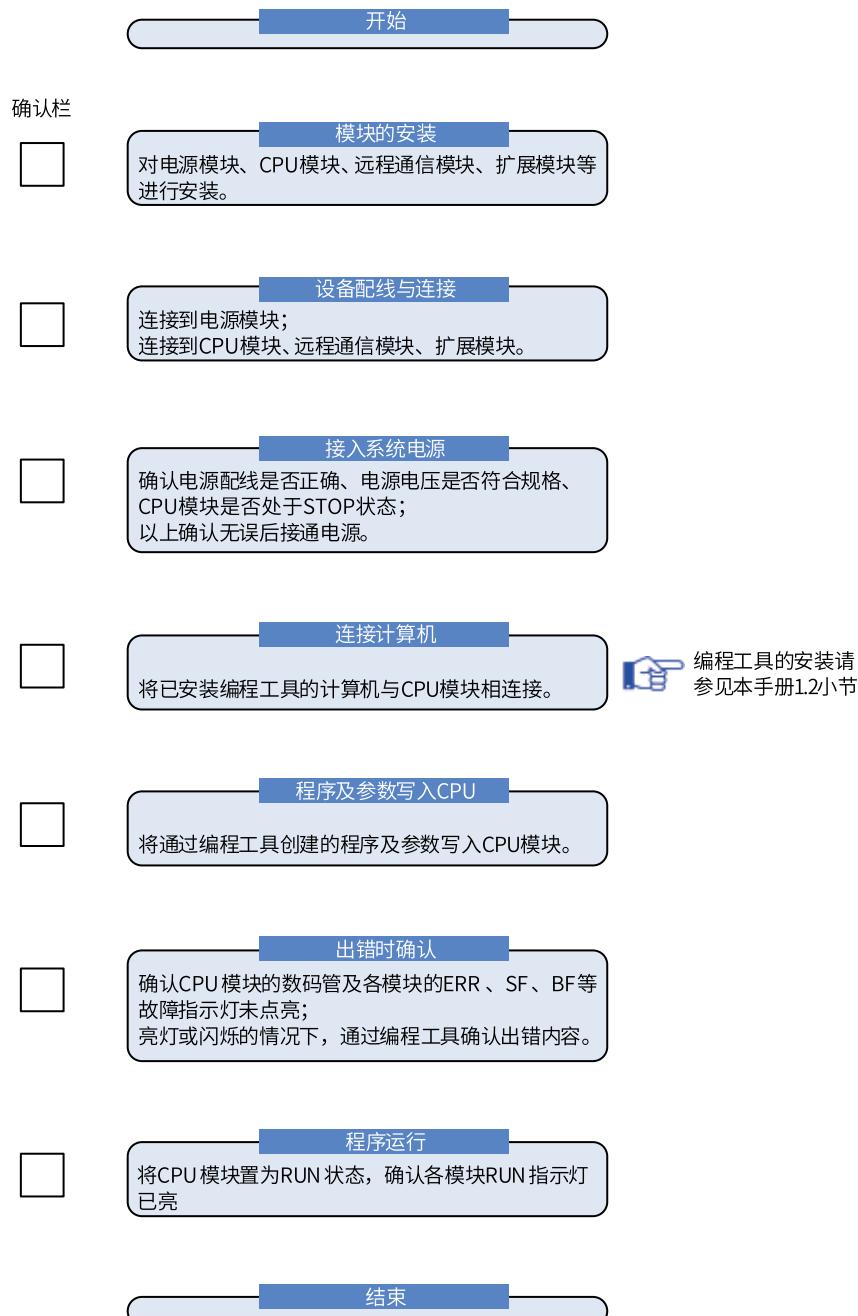


图1-3 中型可编程控制器应用基本流程图

## 1.2 InoProShop 概述

### 1.2.1 InoProShop 简介

InoProShop是面向中型可编程控制器产品的编程组态软件。InoProShop基于CoDeSys V3（简称CoDeSys）平台，为中型可编程控制器提供一套完整的配置、编程、调试和监控环境，可以灵活自由地处理功能强大的IEC语言。

通过InoProShop可完成对工程和设备的管理，为中型PLC产品提供以下配置方案：

- CPU配置；
- I/O 模块配置；
- EtherCAT总线；
- Profibus-DP总线；
- CANopen/CANlink总线；
- Modbus/ModbusTCP总线；
- EtherNet/IP总线；
- 高速I/O。

支持程序的编写、下载和调试等功能，并为编程者提供如下便利：

- 标准化编程（符合IEC 61131-3标准）
 

支持多种编程语言：结构化文本（ST）、梯形图（LD）、顺序功能图（SFC）和IEC61131-3扩展编程语言连续功能图（CFC）。
- 灵活的功能块库
 

全面的功能块库并支持用户自定义库。
- 离线仿真功能
 

不需要连接PLC硬件，完成程序调试仿真。
- 智能的调试查错功能
 

预编译及编译查错，快速定位编程错误，诊断及日志。
- 采样跟踪
 

过程变量的时序图建立。

### 1.2.2 InoProShop 与硬件的连接

编程设备可以通过以太网（可经过集线器、交换机等）或USB线缆与中型PLC相连，使用InoProShop软件编写用户程序，将程序下载到PLC后进行程序监控并控制PLC。



图1-4 InoProShop软件与硬件连接示意图

### 1.2.3 软件获取与安装

#### 软件的获取

汇川中型可编程控制器的用户编程软件InoProShop为免费软件，如需安装文件及中型PLC系列产品的参考资料等，可通过以下途径获取：

- 从汇川的各级经销商处获得软件安装光盘；

- 在汇川技术官网（[www.inovance.com](http://www.inovance.com)）的“服务与支持”“资料下载”页面免费下载软件安装包；

由于汇川公司在不断完善产品和资料，建议用户在需要时及时更新软件版本，查阅最新发布的参考资料，有利于用户的应用设计。

## 软件安装环境要求

具备以下条件的台式PC或便携式PC机：

- Windows 7/或Windows 10操作系统；推荐64位操作系统；
- 内存：4GB或更高配置；
- 空间：可用硬盘空间5GB以上。

PC与中型PLC控制器按以下方式完成连接：

连接方式	所需电缆	备注
采用LAN网络电缆连接（推荐）	需要本地网络中有1个空闲的LAN网口、1根网络电缆。	支持PC与中型PLC之间较远距离连接，如在办公室对车间里的中型PLC进行编程等应用环境，而且互通速率更快
采用USB电缆连接	需要1根USB电缆，其电缆的连接中型PLC控制器一端需为Mini USB插头。	目前支持AM400/AM600系列PLC

## 1.2.4 安装步骤

### 安装前准备

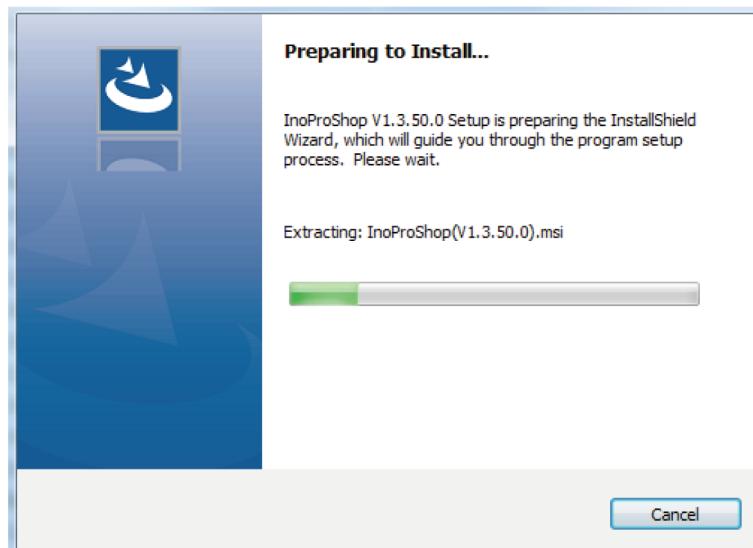
首次安装InoProShop时，请检查电脑硬盘的剩余空间情况，确认所要安装的目标盘剩余空间有5GB以上，直接安装即可。

如果是升级安装InoProShop，请先备份已有的工作文件，然后卸载旧版本InoProShop；重新启动电脑后，再开始安装新版本软件。

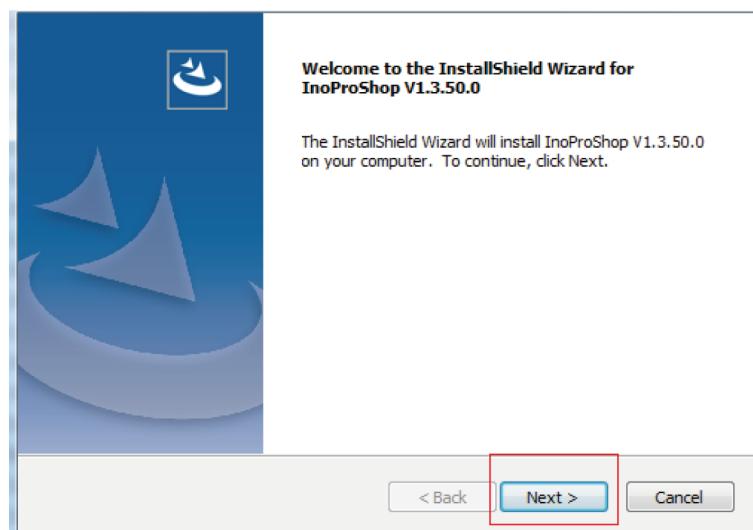
### 开始安装

通过Windows的资源管理器，在安装文件所在目录，双击打开InoProShop（V\*.\*\*.\*）.exe文件（V\*.\*\*.\*为InoProShop的软件版本，请确保您安装的版本为最新版本）。

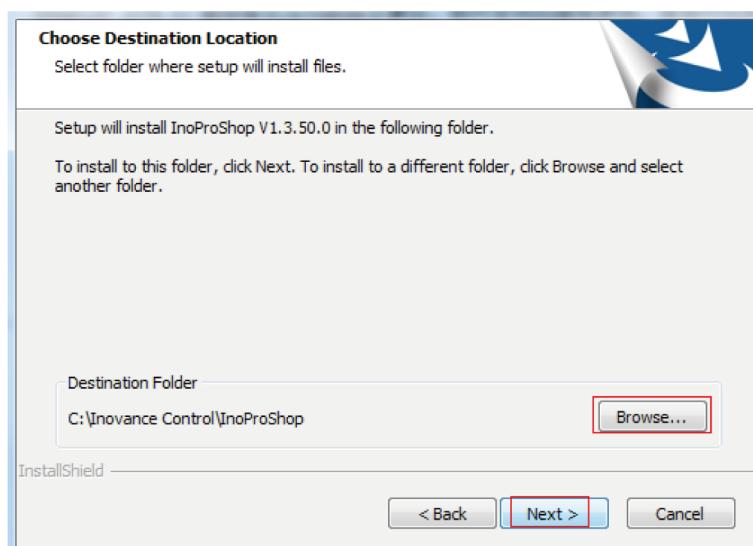
启动安装后，可以看到如下界面，表示进入安装准备阶段：



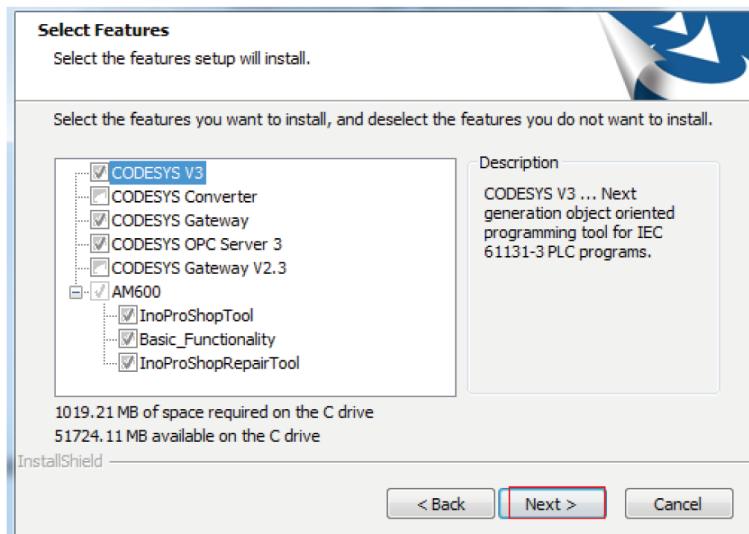
出现如下提示界面后，点击“Next”，开始安装：



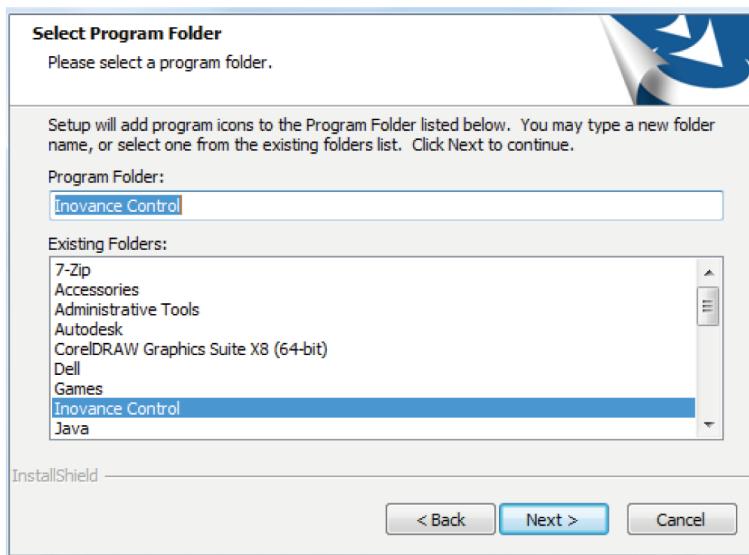
设置好软件安装路径后，点击“Next”，进入下一步：



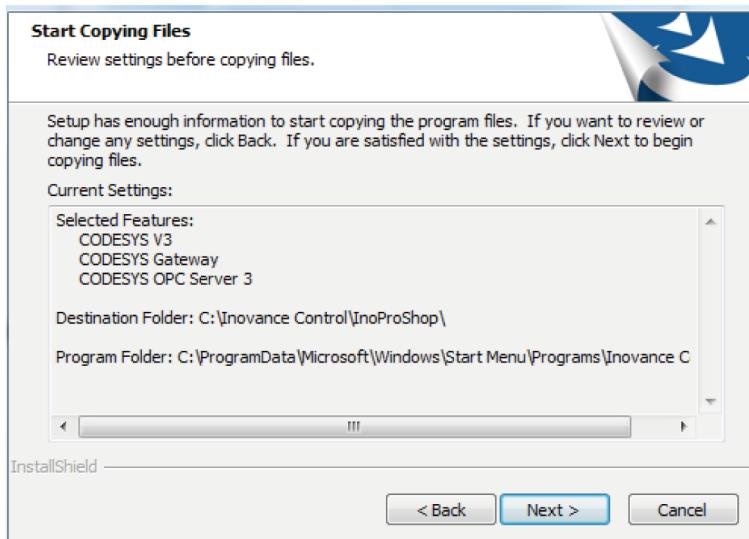
进入安装组件选择界面，可个性化进行勾选，如无特殊需求，按默认勾选即可，点击“Next”：



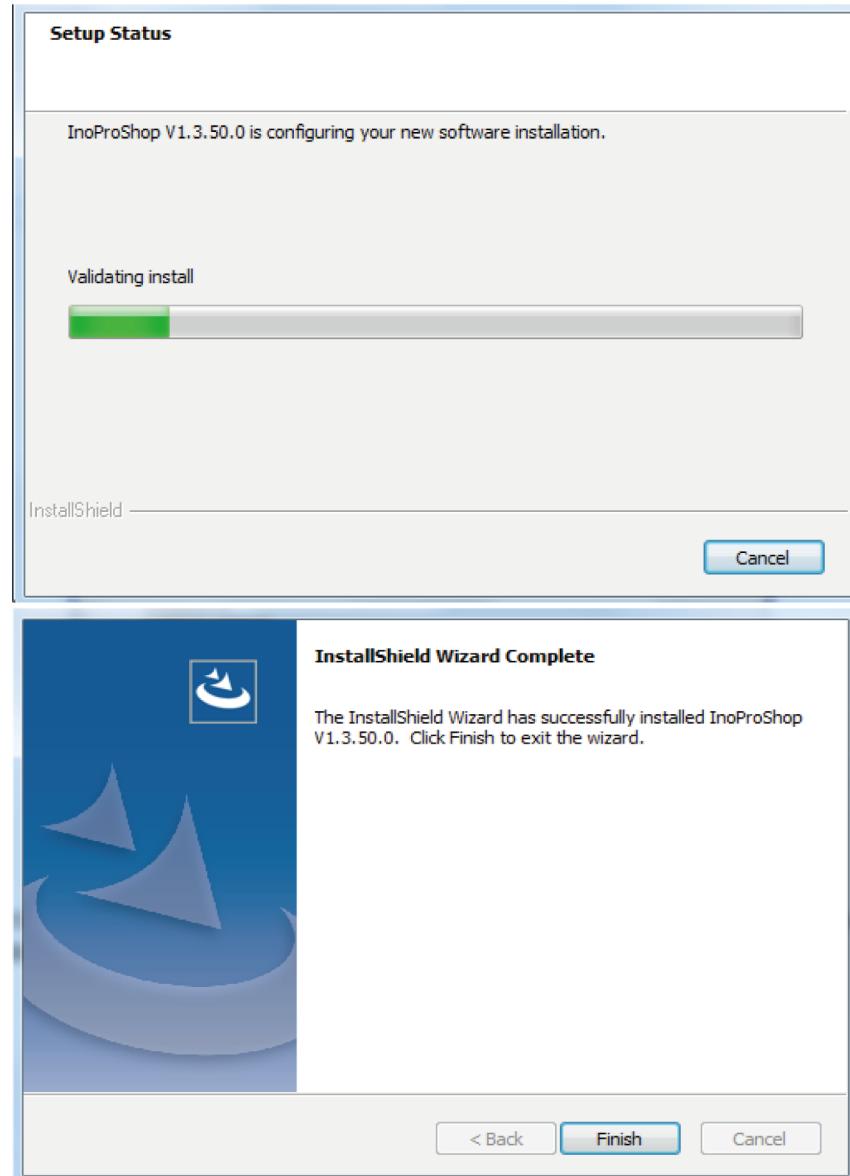
出现如下提示界面，点击“Next”：



出现如下提示界面，点击“Next”：

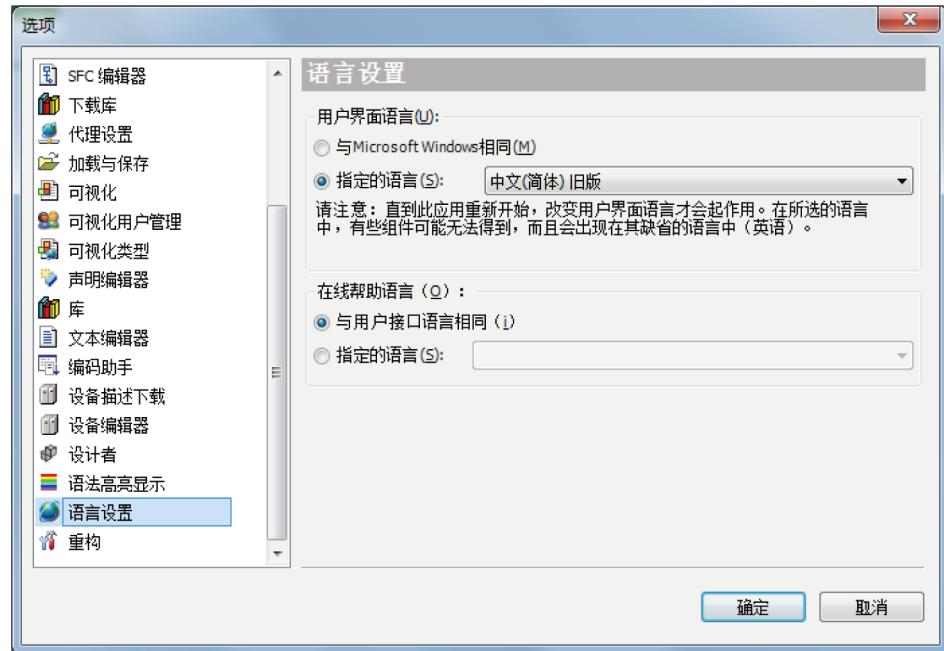


显示如下界面后，等待安装进度条，直到出现下图所示界面提示，点击“Finish”完成InoProShop的安装。



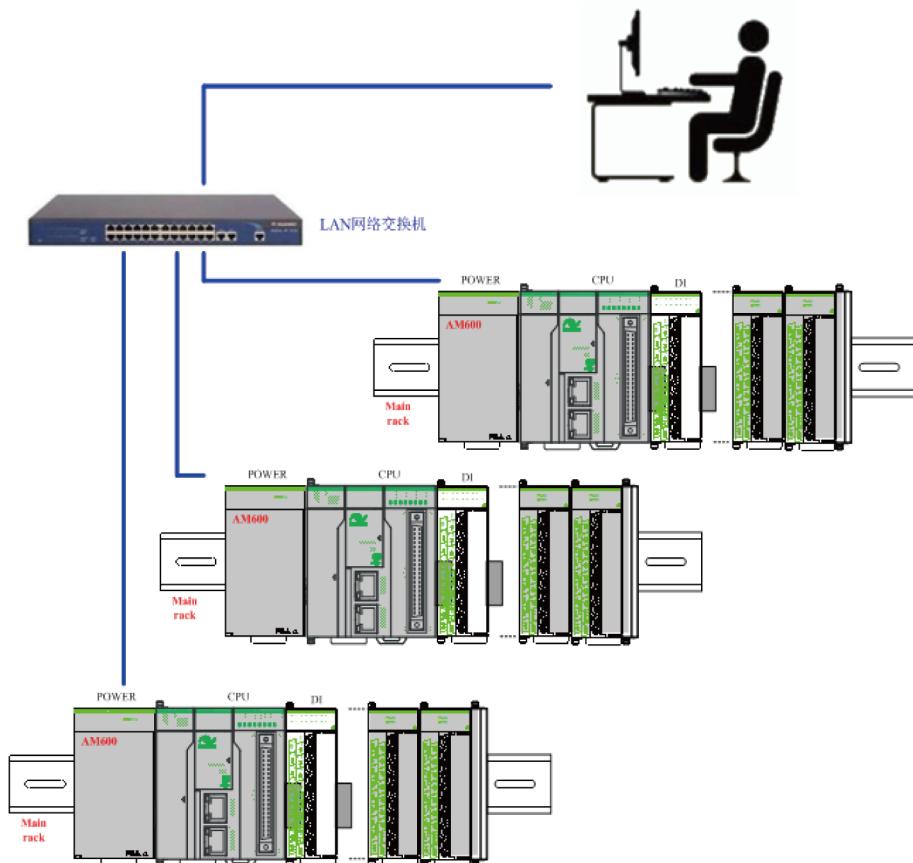
## 语言设置

完成安装后，InoProShop界面的操作语言默认为简体中文，若需要切换为其他语言，可点击软件主界面的“选项” “语言设置”，进行语言选择设置。



### 确认所选控制器是否正确

有时同一个局域网内有多个AM600，当选择登录了某台控制器后，可能需要试验确认所选控制器是否正确：



试验方法是在InoProShop的“Device”设备页面，选择“系统设置”标签，点击“识别设备”按钮。

- ① 双击Device设备。
- ② 打开“系统设置”界面。

③ 点击“识别设备”，“通讯设置”界面所选的PLC数码管会交替闪烁。

如下图：



## 1.2.5 卸载InoProShop

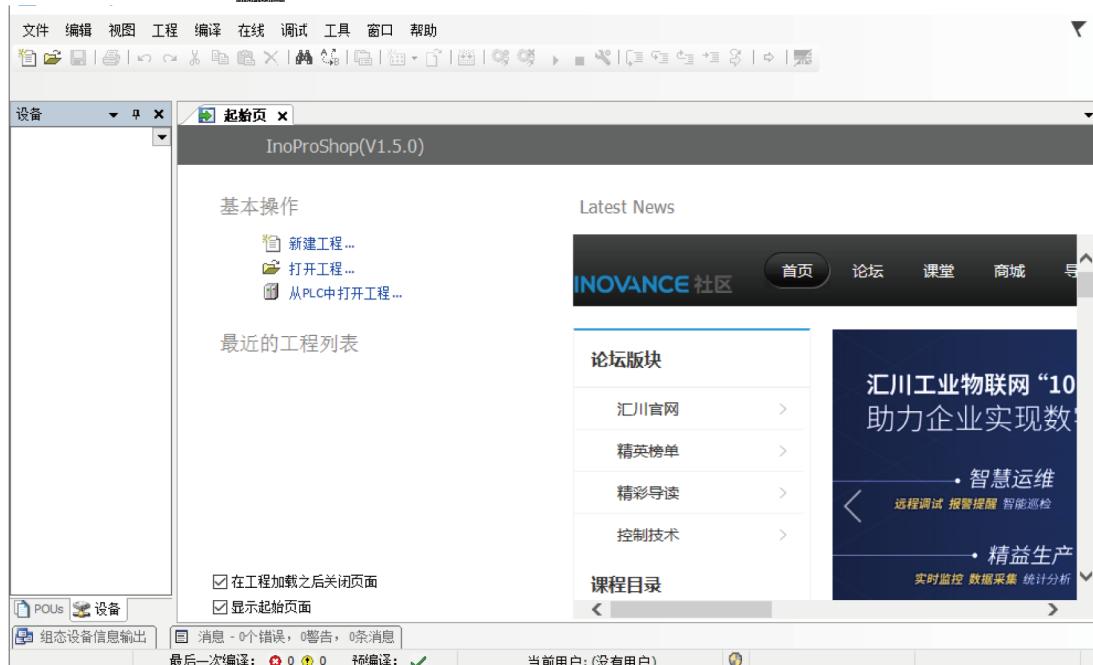
使用标准Windows系统卸载软件方法卸载InoProShop即可，具体步骤如下：

1. 退出InoProShop软件，确认Gateway已关闭。
2. 如果操作系统任务栏存在CoDeSys图标，可在该图标上点击鼠标右键，选择“退出”（Exit）关闭“Gateway”。
3. 选择“开始->设置->控制面板”（Start -> Settings -> Control Panel）。
4. 双击“添加/删除程序”（Add or Remove Programs）。
5. 选择需要卸载的软件项，找到“InoProShop”。
6. 右键单击软件，选择“删除”按钮，并确认删除。

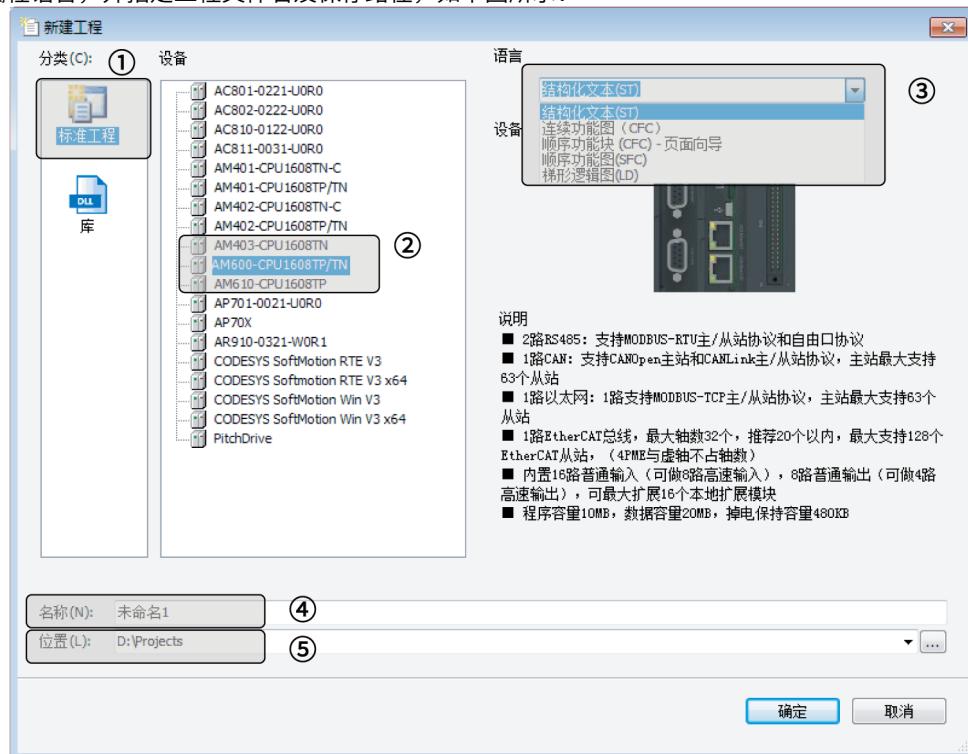
## 2 快速入门

### 2.1 启动编程环境

1. 双击桌面编程软件图标  即可启动InoProShop编程环境，起始页显示画面如下：

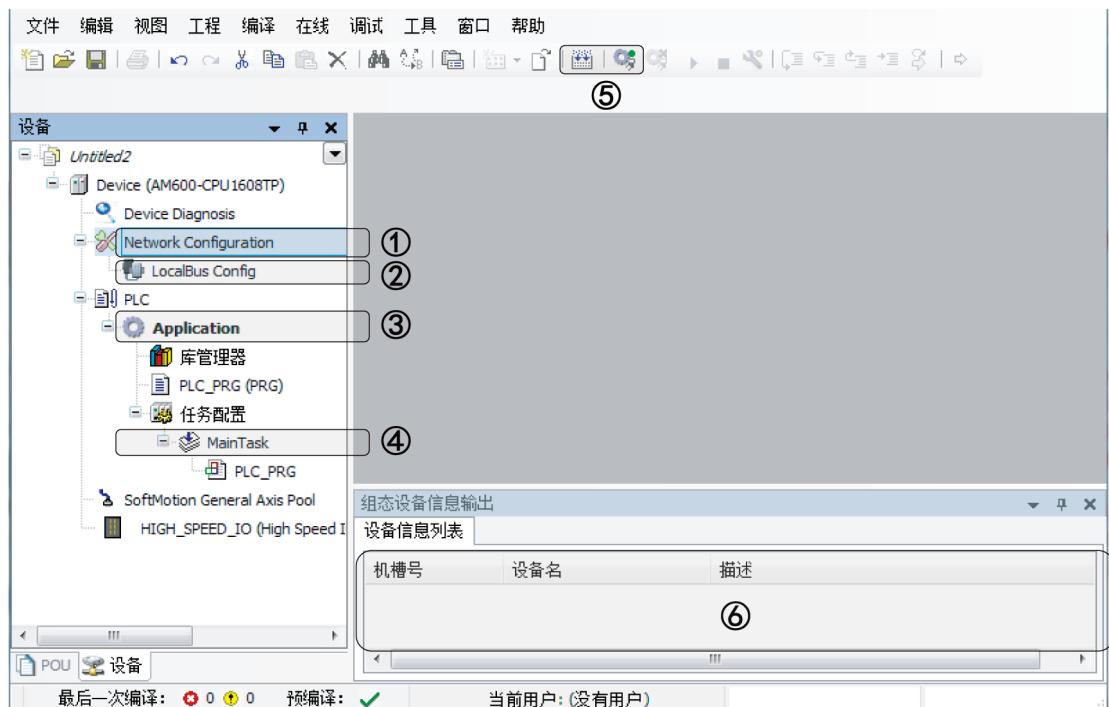


2. 点击菜单栏左上角  新建工程或者选择“文件” - “新建工程”，选择工程类型“标准工程”选择设备类型和编程语言，并指定工程文件名及保存路径，如下图所示：



序号	描述
①	选择工程类型为“标准工程”
②	选择主模块机型
③	选择熟悉的编程语言
④	填写工程名
⑤	选择存放路径

3. 点击“确定”后，进入系统组态配置与编程界面，常用的按钮与窗口分布如下图：



序号	描述
①	网络配置
②	本地总线配置
③	用户程序管理单元
④	配置任务执行方式及周期
⑤	编译、登录及调试
⑥	设备信息窗口

## 2.2 编写用户程序的典型步骤

### 2.2.1 概述

初次使用汇川中型PLC的用户需要注意，编写调试一个完整的用户程序需要5个步骤。

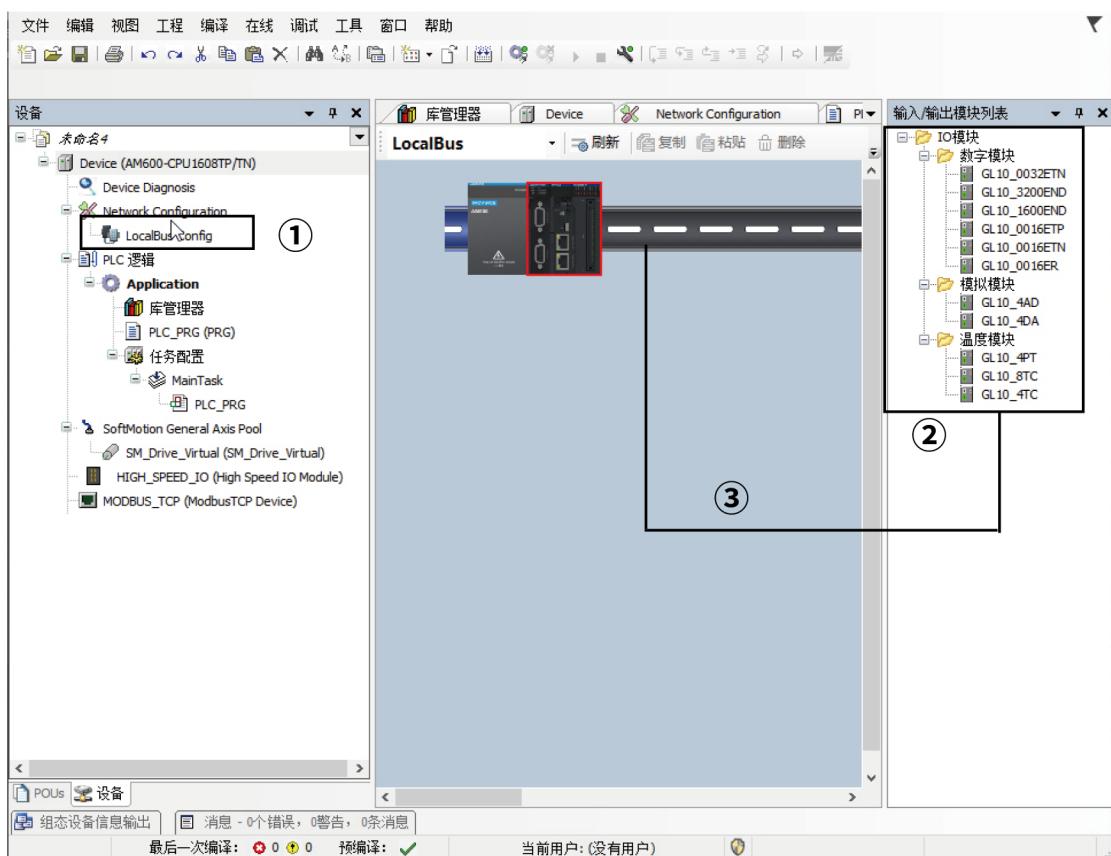
1. 基于中型PLC应用系统的硬件连接架构进行硬件系统配置。

- 若只用了CPU主模块和IO扩展模块，只需进行硬件配置：即根据实际选用的模块类型、型号、安装顺序，在InoProShop的硬件配置页面把这些“元件”放进“主机架”。

- 若用到了扩展机架，需要先配置网络总线，再根据扩展机架数量，增置对应数量的网络扩展模块，然后给每个机架放置扩展模块硬件。
2. 根据应用系统的控制工艺编写用户程序。用户程序编程基于数据的存储宽度、使用范围来自由定义变量，可以与硬件配置无关。
  3. 将系统架构中的各硬件端口对应的输入端口变量 (I) 、输出状态 (Q) 或数值 (M) 与用户程序中的变量进行关联。
  4. 配置网络通讯的同步周期（如EtherCAT总线）。根据各任务的实时性要求，配置用户程序单元的执行周期。
  5. 在InoProShop编程环境下登录中型PLC，下载用户程序，仿真调试、排错，直到正确无误地运行。

## 2.2.2 用户系统的配置操作

在InoProShop的主画面，双击左侧设备树中的“LocalBus Config”项，进入PLC主机架的硬件配置画面：



① 双击进入本地扩展模块配置界面；

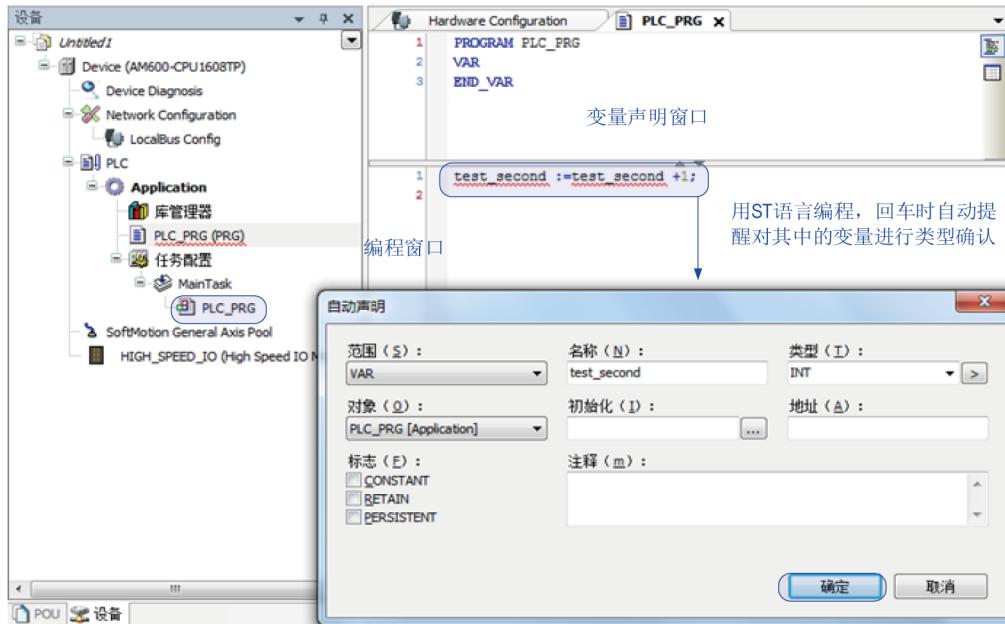
② 扩展模块元件库；

③ 安装槽上选中CPU单元右侧位置，在扩展模块元件库，双击选中所需的IO模块，依次摆放。根据实际应用系统使用的模块型号、安装顺序，从右侧的扩展模块库中依次双击选中模块，将其拖放到“安装机架”上；若要删除某个模块，选中该模块后按Del键可以删除。

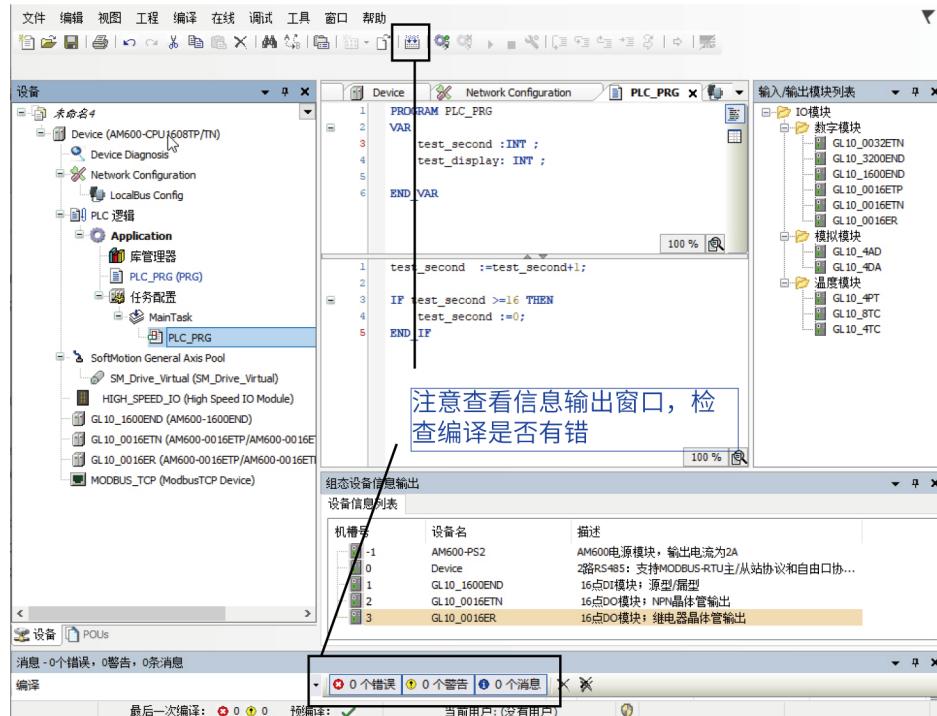
以AM600为例，主机架上最多可以接入16个扩展模块。

## 2.2.3 用户程序的编写操作

双击左侧设备树窗口中的“PLC\_PRG(PRG)”项，即可打开用户编程界面，编程语言为ST（新建工程时选择），如下图所示。与C语言编程相似，每个变量需要声明后才能使用；如果先直接编写程序语句，回车时编程环境会自动弹出声明框；经用户填写并点击“确定”后，变量声明窗口会自动增加该变量的声明语句，这样简化了编程：

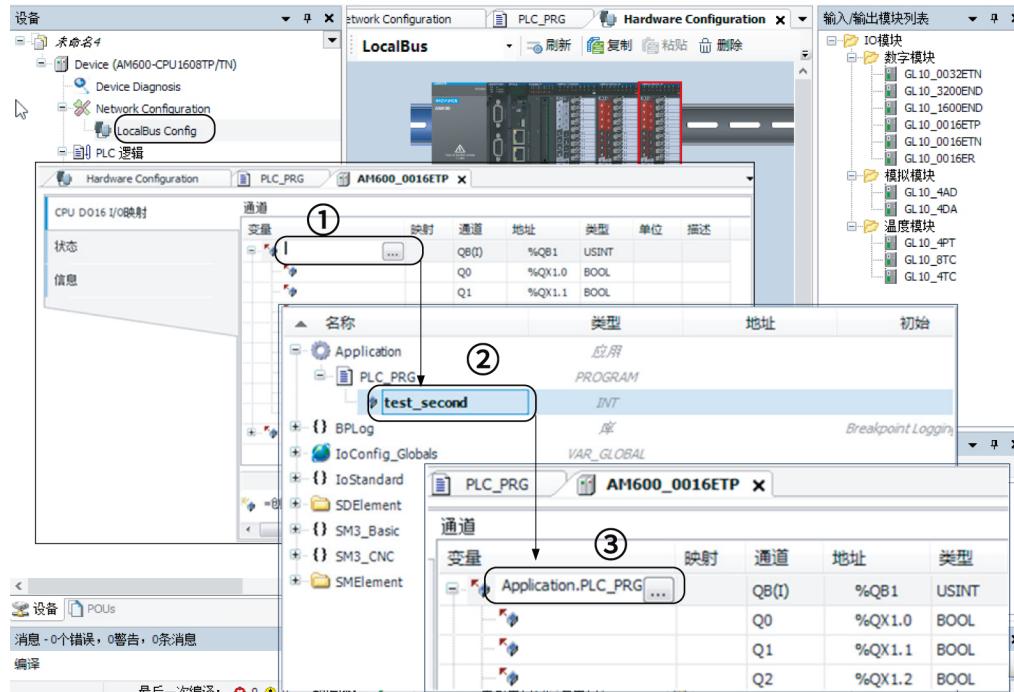


编写一个简单示例：将第二个变量赋值给第一个变量，然后递增：



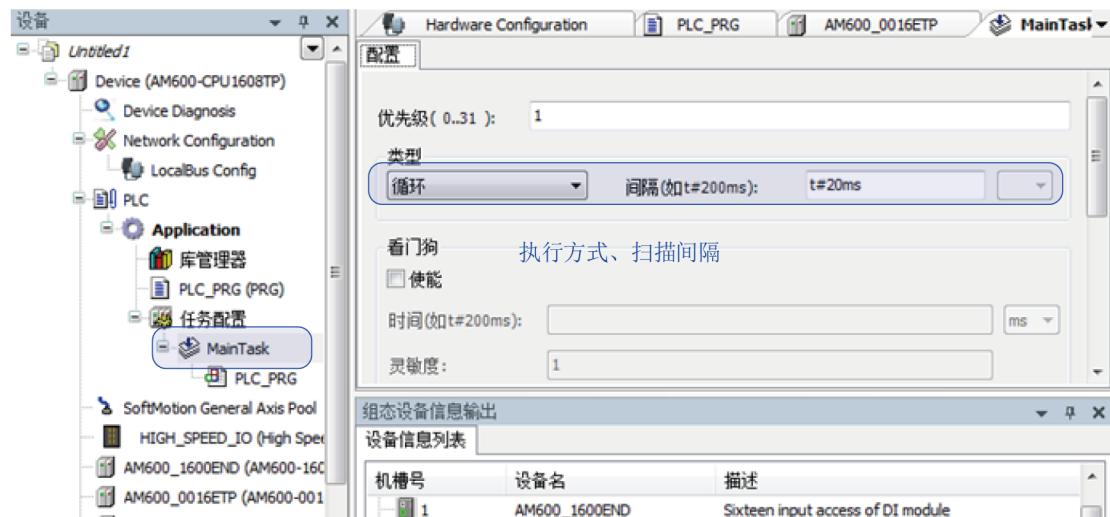
## 2.2.4 用户程序变量与端口的关联配置

在本地总线配置页面，将需要关联的硬件端口与用户程序中的变量进行关联。如下图所示，将“test\_display”的变量值，关联到第一个DO模块的输出端口，配置步骤如下：



## 2.2.5 配置用户程序的执行方式和运行周期

上文示例中编写的子程序默认为20ms执行一次，如要改为其他的执行方式，如反复执行、定时执行和执行周期等，可以按下图所示分别设置：



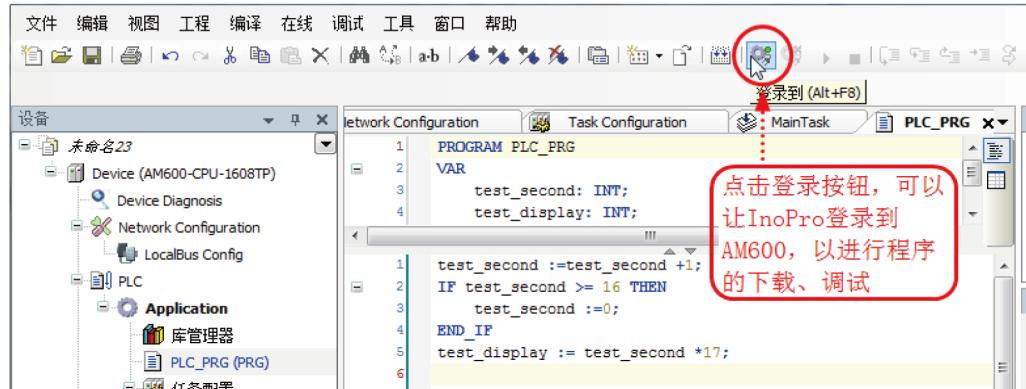
## 2.2.6 用户程序的编译、登录下载

完成上文的程序编写后编译程序，查看是否有错；若有错，点击错误信息行可定位到用户程序的报错点，方便修改，直到错误全部排除。

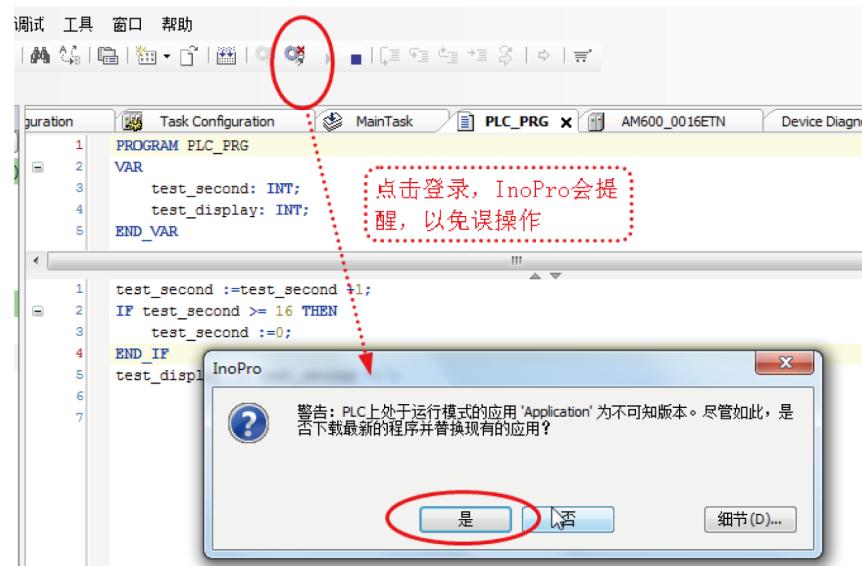
相关编译信息会显示在如下编译信息框中：



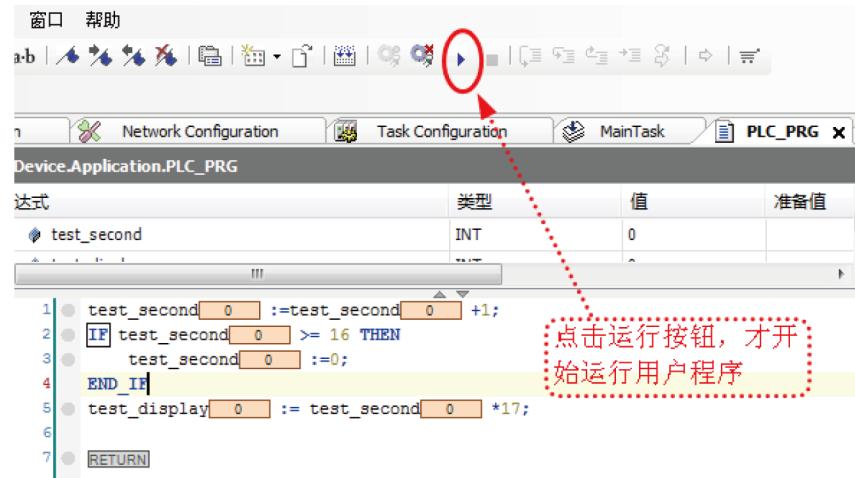
编译无误后，点击“在线” - “登录到”，如下图所示：



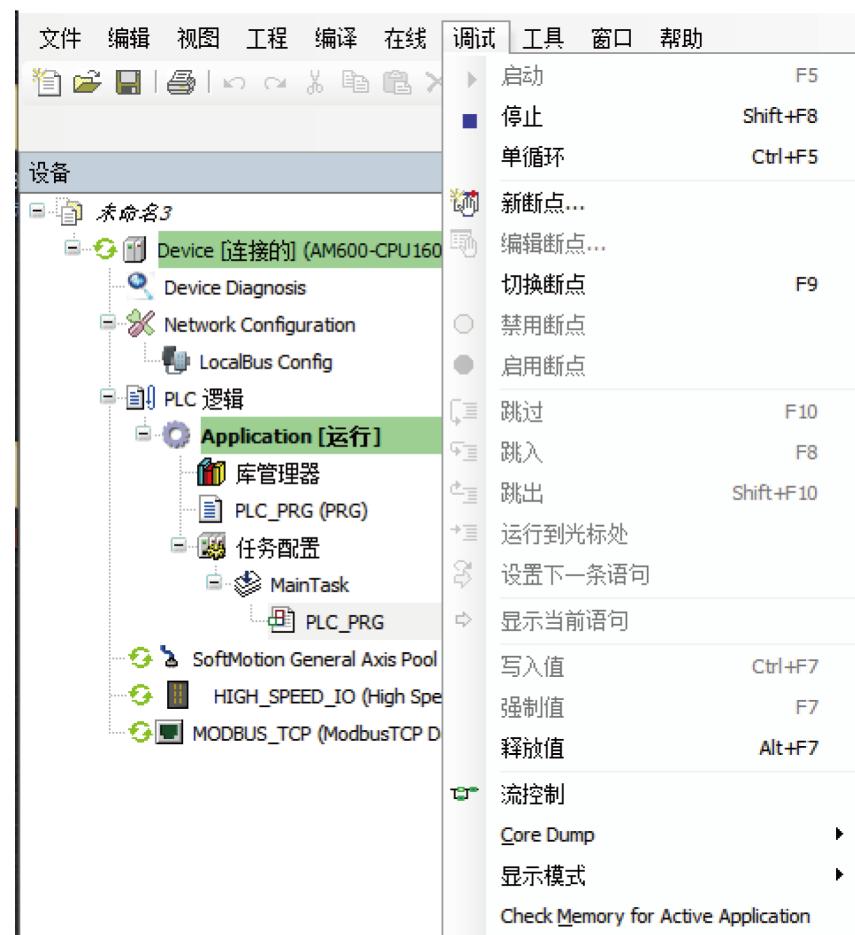
然后弹出如下对话框，选择是否创建程序并继续下载：



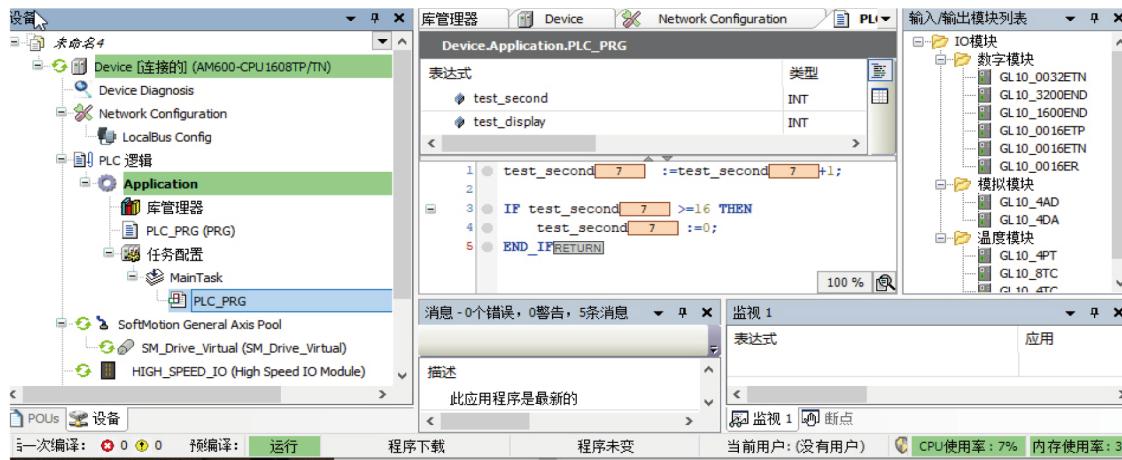
选择“是”，上位机与设备建立连接并保持，初始状态为“停止”，如下图所示：



点击“调试” - “启动”，设备进入运行状态，并开始执行用户程序。



正在运行的用户程序监控画面如下图所示：



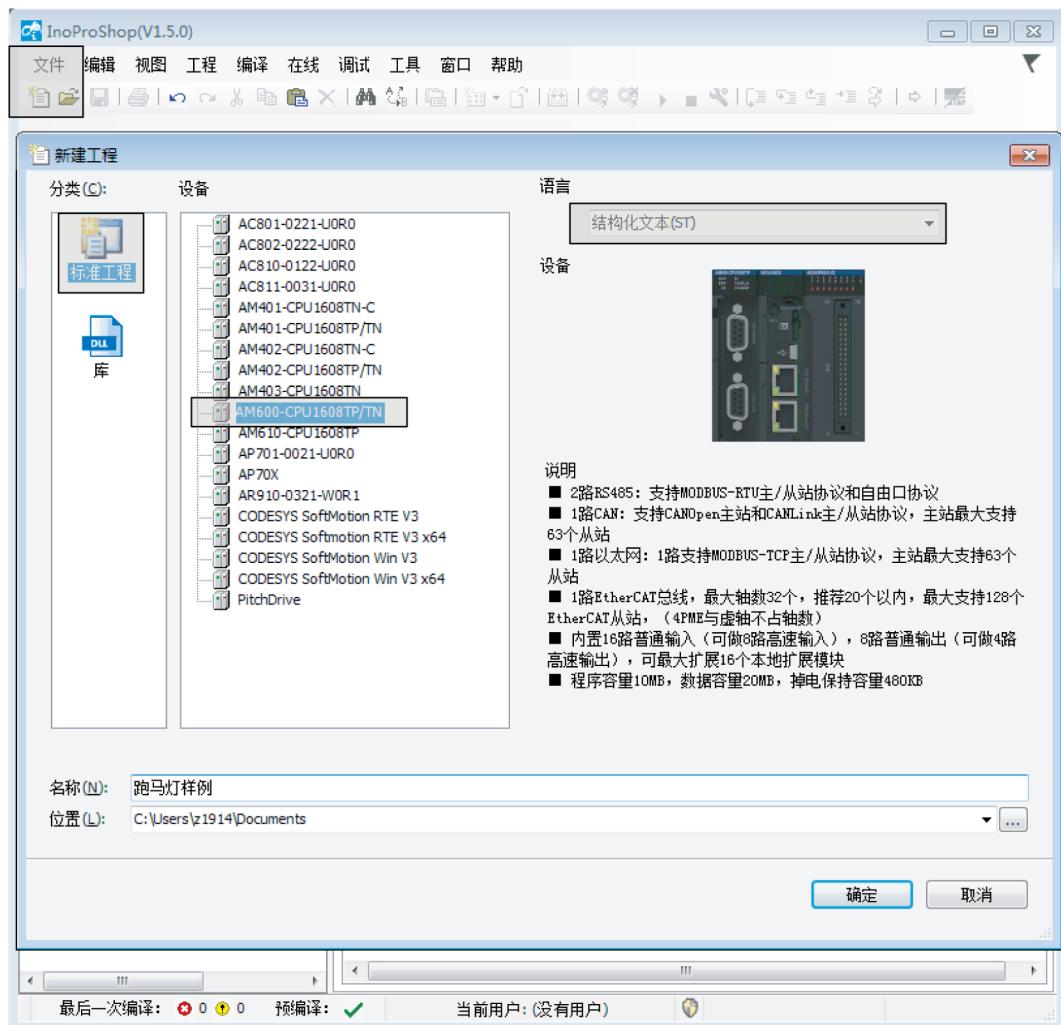
此时查看AM600后面的第一个DO模块，可以看到其输出状态指示灯以二进制计数方式循环计数。

## 2.3 用InoProShop 编写一个跑马灯样例工程

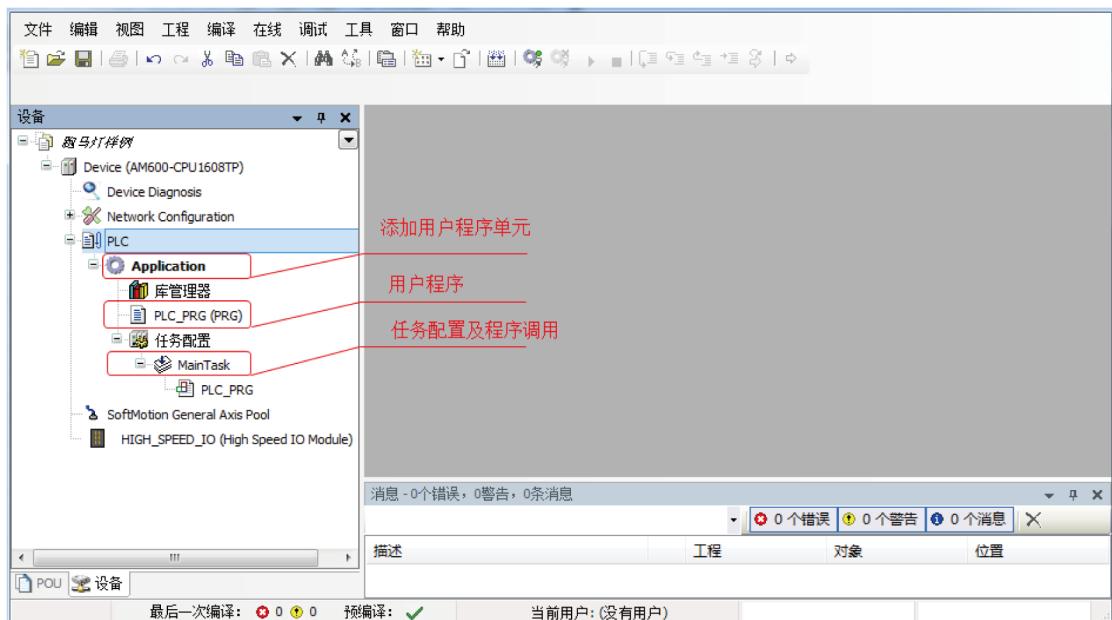
### 启动InoPro编程环境

新建工程：

点击菜单栏左上角 新建工程或者“文件” - “新建工程”，选择工程类型为“标准工程”，选择设备类型（选择主模块的机型）和编程语言，指定工程文件名及保存路径，如下图所示：

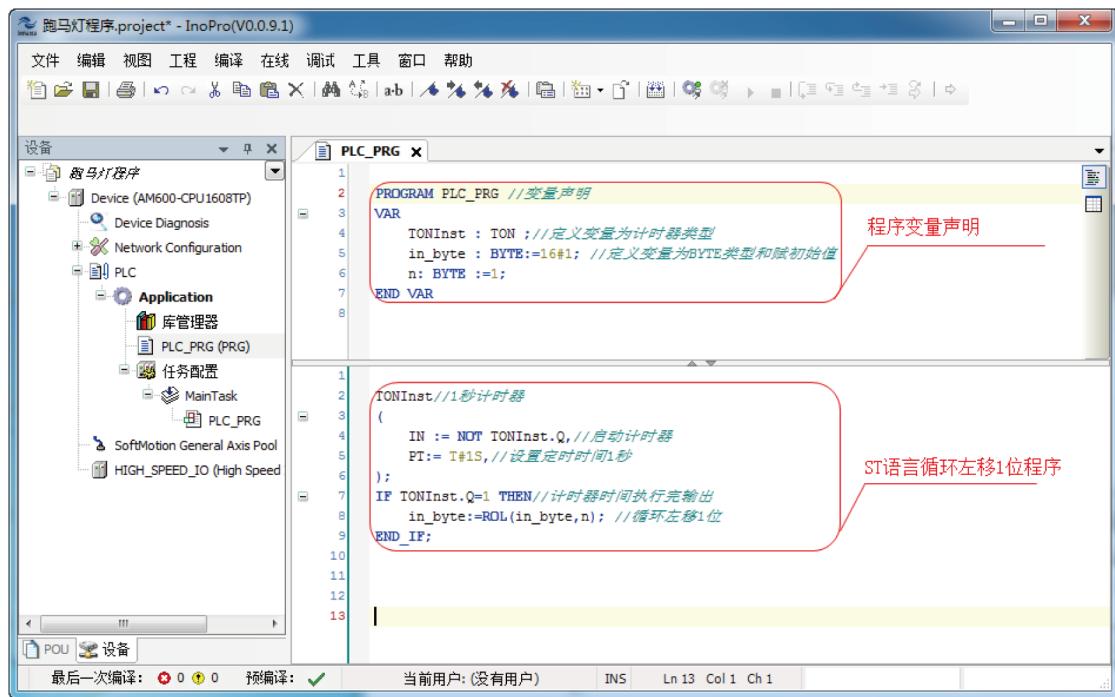


## 系统组态配置与编程界面



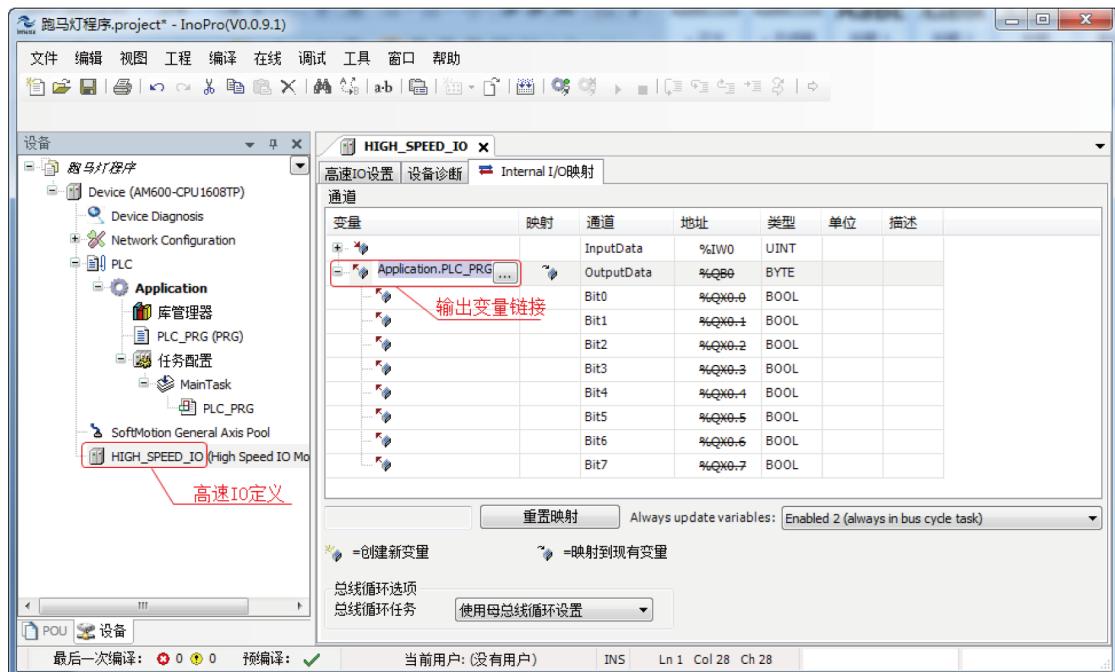
## ST语言编写“跑马灯样例程序”

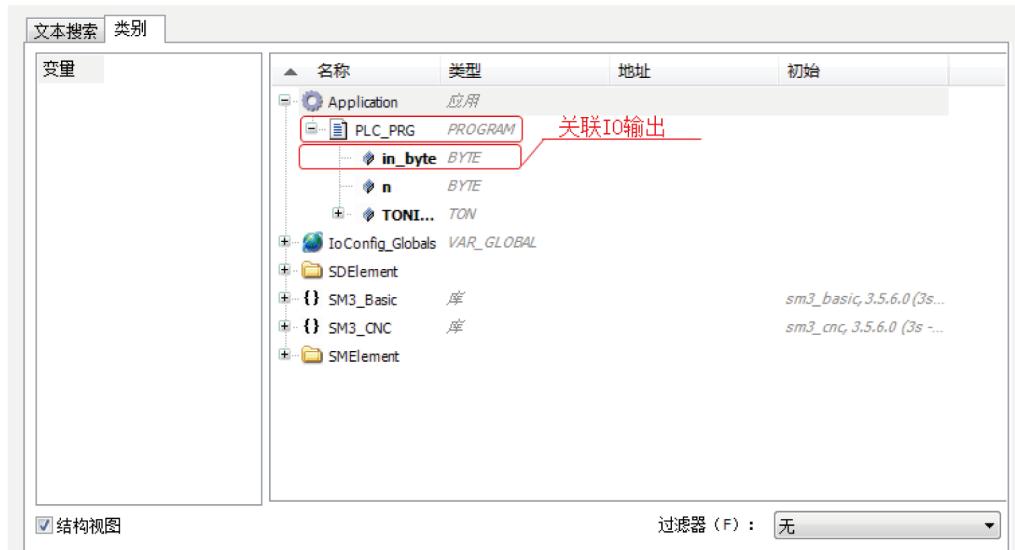
双击打开“PLC\_PRG”程序组织单元。



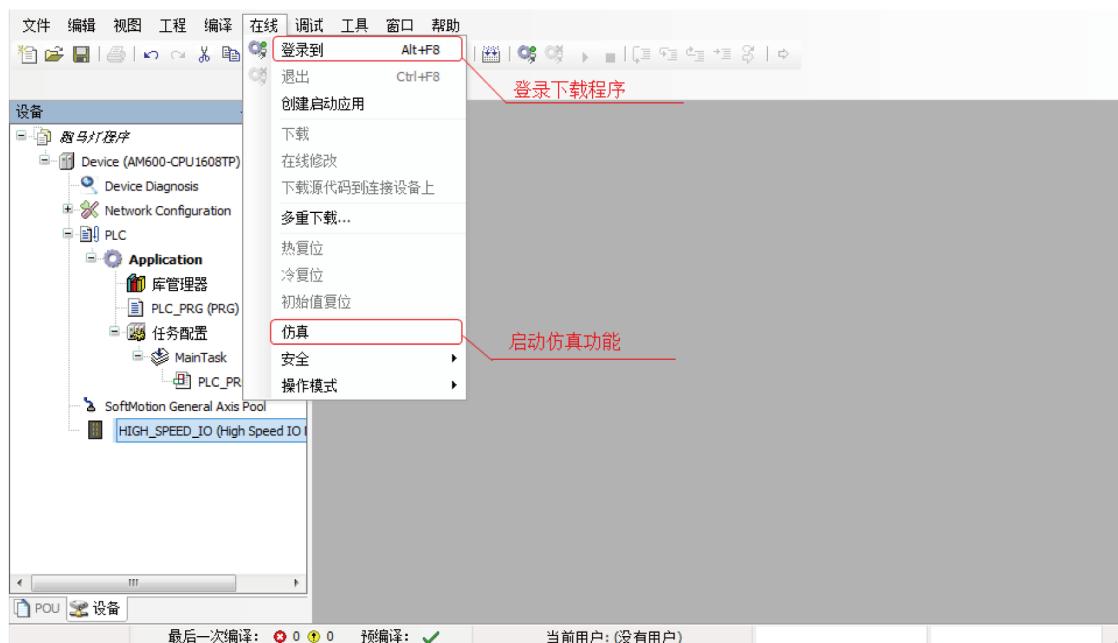
## 关联PLC输出IO

循环左移“in\_byte”变量与PLC自带的8路输出端口链接（bit0-bit7），观察输出灯的变化。





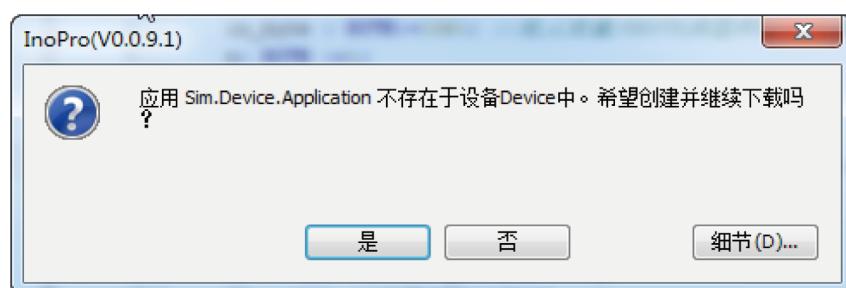
## 仿真调试



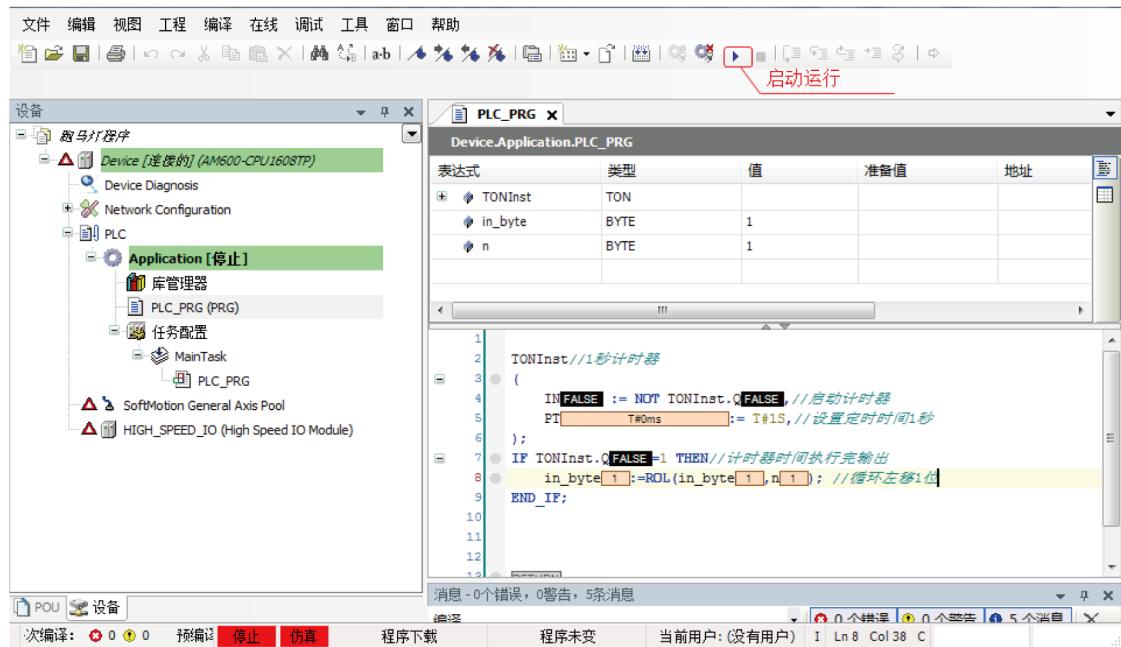
点击“仿真”进入仿真功能，这时不需要链接PLC亦可观察IO移位状态。

## 在仿真模式下下载程序

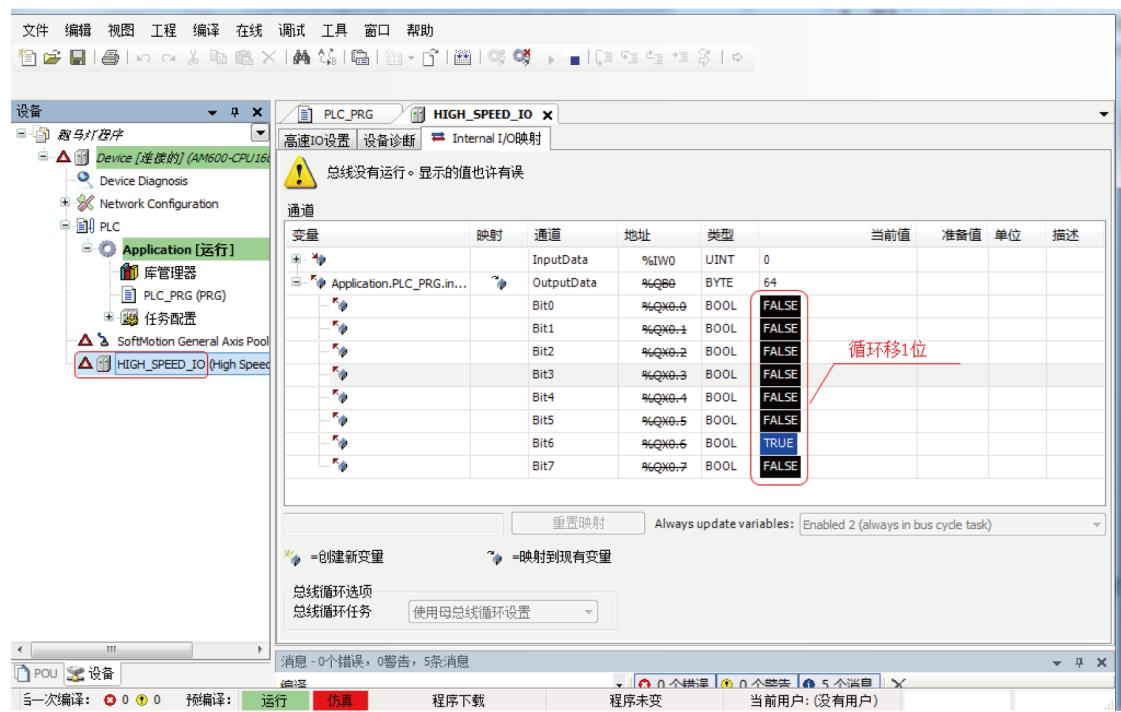
点击“登录”，在仿真模式下下载程序。



## 下载完成后，启动运行PLC



## 监控IO变化



## 2.4 如何登录主模块

### 2.4.1 登录主模块的必备条件与操作简介

“登录主模块”是指在PC上运行的InoProShop与中型PLC主模块建立通讯，从而进行用户程序的运行、下载、启停和监控，以及程序参数查看或修改操作等。

- 目前有两种方法登录中型PLC：通过LAN局域网；通过USB连接。
- PC与中型PLC之间可以通过网线进行1对1直连；也可以通过路由器、集线器无线连接PLC，这种情况下，一台PC可以与多台中型PLC联机，也可以多台PC访问同一个中型PLC。
- PC与中型PLC两者的IP地址必需在同一个网段才能登录，否则InoProShop将无法扫描到中型PLC。比如，AM600的出厂默认IP地址为192.168.1.88，若PC的IP地址为192.168.1.xxx（这里xxx范围为1~254，但不得与AM600的IP相同），那么InoProShop就可以扫描到AM600并与之交互数据，进行用户程序下载、运行监控等。若AM600的IP被人为修改过，其地址与PC不在同一IP网段，二者将无法建立通讯，此时需要将AM600的IP地址恢复为出厂地址192.168.1.88，再将PC本机的地址修改为192.168.1.xxx，二者建立1对1联机后，将AM600的地址修改为所需的IP网段地址。
- 如果通过用户想通过USB登录PLC，插上USB连接线（MiniUSB口），等待2~5s-60s就可以扫描到设备。

USB连接注意事项：

1. USB驱动在安装软件时会自动安装，如果没有自动安装，可以在安装目录中的Common文件夹下找到，如下图所示。



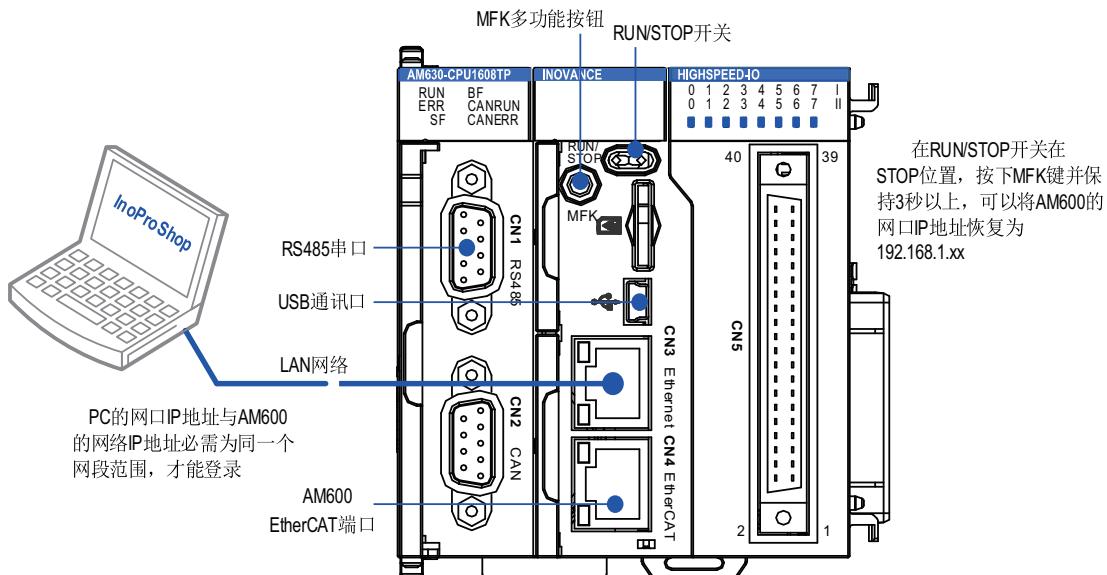
然后通过Windows设备管理器更新驱动，从安装目录中安装驱动，USB连接成功后，Windows设备管理器显示如下图所示。



2. 若USB连接和网络连接同时存在，默认使用网络连接（网络扫描速度更快）。

### 2.4.2 在InoProShop 中扫描中型PLC 网络设备

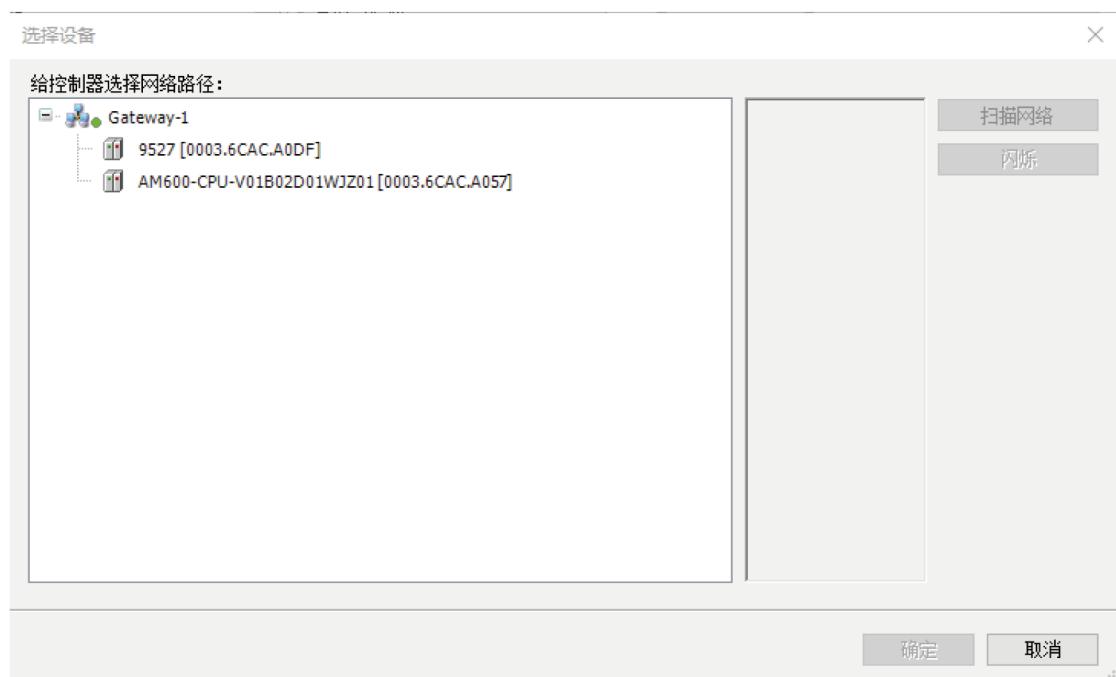
PC可通过LAN网络登录中型PLC，以AM600为例，连接方式如下：



在InProShop中，双击Device(AM600-CPU-1608TP/TN)，弹出如下界面：



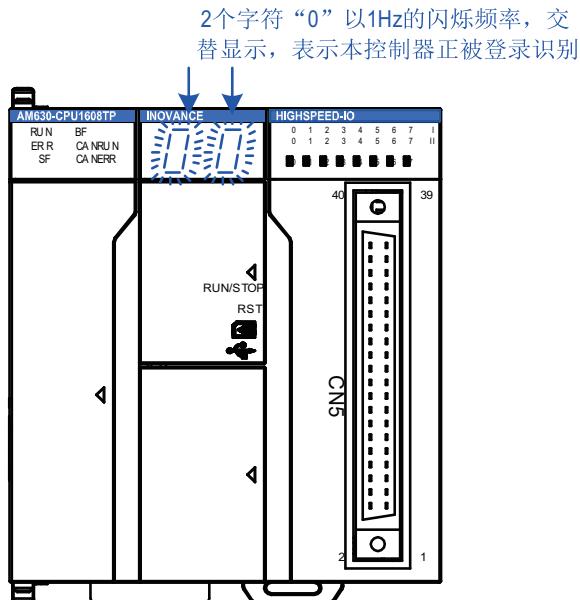
在该界面点击“扫描网络”标签，弹出如下界面，在窗口左侧点击其中的AM600-CPU控制器，即可在窗口右侧可以看到其简介信息：



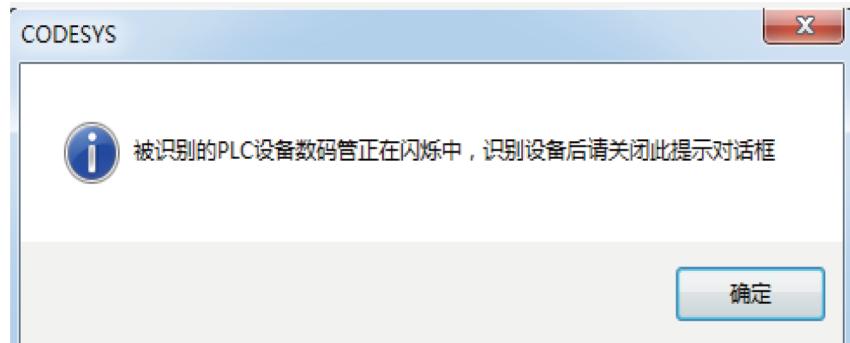
上例图中找到了2台控制器，分两行显示：

- 第1台9527[0003.6CAC.A0DF]: 网段内的设备, 名称9527, 括号中数字的最后2位“DF”为该AM600的IP地址第4个段位, 为16进制显示, 转换为十进制为223。
- 第2台AM600CPU-V01B02D01WJZ01[0003.6CAC.A057]: 网段内的另一个设备, 其设备名为AM600CPU-V01B02D01WJZ01。用户登录后可以根据需要自行修改设备名, 这样在有多台控制器的应用场合方便识别。

此时, 登录的AM600或AM610上的两位数码管, 将交替显示字符“0”, 如下图所示:

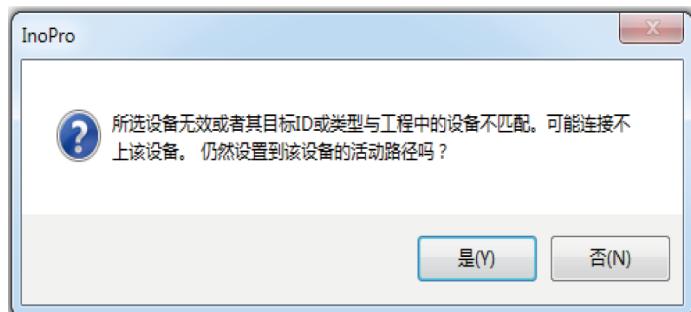


直到用户点击InoProShop中弹出窗口的确认按钮后, 才停止闪烁, 恢复原有的显示信息:



双击选中的设备, 或者选中设备后再点击“确认”即可激活上位机与当前设备的连接。

若当前的工程中记录的控制器标识号与所选择的控制器不符, 可能提示以下信息, 若要联机, 点击“是”确认。



### 2.4.3 扫描不到设备的处理对策

如果在InoProShop中扫描不到AM600设备，可能原因和对策如下：

1. CoDeSys网关没有启动。

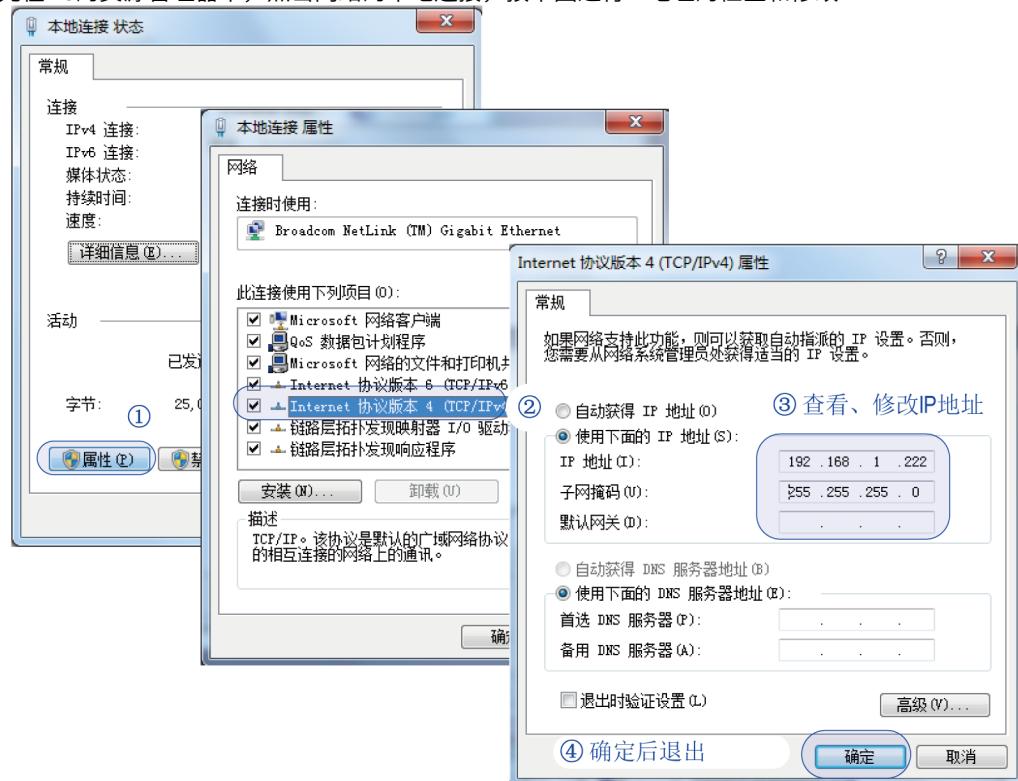
解决方法：重新启动网关后，再进行扫描。



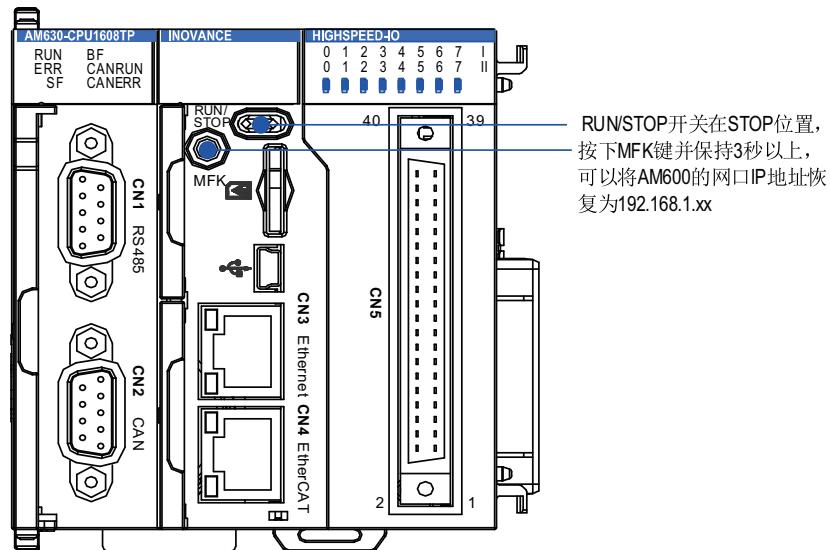
2. PC的IP地址与AM600的IP地址不在同一个网段。

解决方法：将PC的IP地址设置到AM600 IP地址的同一网段。如果忘记了AM600的IP地址，建议将其恢复为默认IP，再将PC的IP地址设置为192.168.1.xxx网段，即可正常扫描设备。

a. 首先在PC的资源管理器中，点击网络的本地连接，按下图进行IP地址的检查和修改：

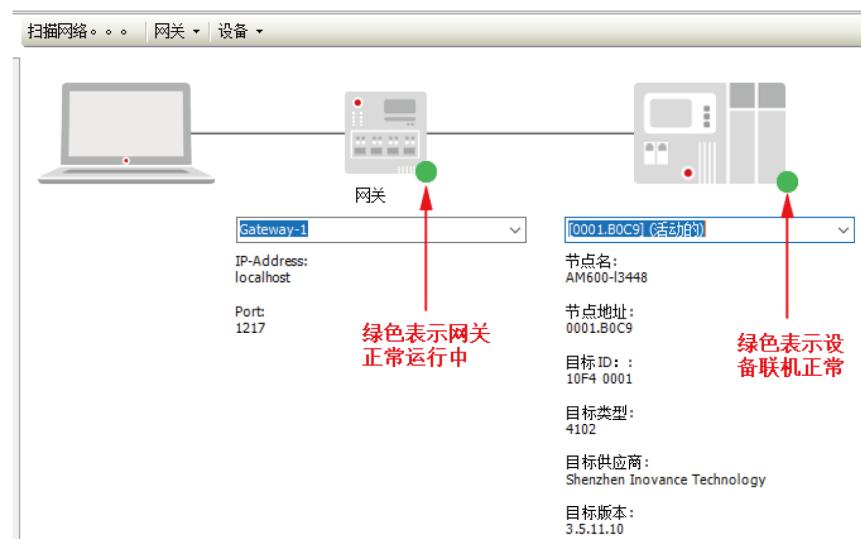


b. 其次恢复AM600的出厂默认IP地址（192.168.1.88）。AM600上电启动后，将RUN/STOP拨到STOP状态，然后按住MFK键直到数码管显示IP，松开按键；如果确认要复位IP地址，再按一下MFK键，数码管开始显示10, 9, 8...倒计时：在计数到0之前按MFK键取消复位操作，倒计时结束，IP复位完成，PLC重新上电后，将使用新的IP地址。



无论是复位AM600的默认IP，还是通过InoProShop后台软件修改AM600的IP地址，都会使修改后的IP地址立即生效。

一旦扫描联机成功，在设备扫描画面可看到如下网络状态信息：



# 3 基本功能

## 3.1 界面导航

向左和向右的按钮，分别代表返回到上一次编辑位置和恢复到下一个编辑位置。鼠标点击后，可帮助用户更快定位修改用户程序位置。



## 3.2 编译命令

在软件编译菜单中集成了编译和清除等功能，不同版本功能的对比，如下表所示。

1.5.2版本	1.7.3 SP4版本

1.5.2版本	1.7.3 SP4版本	功能说明
编译	检查程序	检查用户程序编写有无错误。
生成代码	编译	把所有代码编译为PLC可以执行的代码。 “生成代码”更新为“编译”后，对于标签通讯中符号配置所要用到的生成文件功能，也可单击“编译”来实现，或者单击快捷键  。
重新编译	重新编译	清除上次的编译信息，重新执行编译命令。
生成运行时系统文件…	生成Runtime文件…	测试时使用。
清除	清除	清除上次的编译信息和下载信息。
清除全部	清除全部	清除编译信息、下载信息和引用信息，所有的库、工程数据编译信息重新刷新。

1.5.2版本	1.7.3 SP4版本	功能说明
打包用户程序	打包用户程序	生成用户程序文件，可将该用户程序文件拷贝至SD卡，通过SD卡升级用户程序。
-	重名检测	<p>检查工程中拥有相同名称的符号或变量，以免对相同名称的符号或者变量使用造成歧义。</p> <p><b>检查类型：</b></p> <ul style="list-style-type: none"> <li>• 设备</li> <li>• 变量表</li> <li>• LD</li> <li>• CFC</li> <li>• ST</li> </ul> <p><b>检查范围：</b></p> <ul style="list-style-type: none"> <li>• 库</li> <li>• 设备</li> <li>• 全局变量表</li> <li>• 自定义类型</li> <li>• 本地变量</li> <li>• 本地POU包含Function、FunctionBlock、PRG和METH</li> </ul>

重名检测规则如下表所示。

		当前工程			引用库	
		局部变量	全局变量	类型	全局变量	类型
当前工程	局部变量	编译报错	-	-	-	-
		检测报错	-	-	-	-
	全局变量	编译通过	编译报错	-	-	-
		检测警告	检测报错	-	-	-
	类型	编译通过	编译通过	编译报错	-	-
		检测警告	检测警告	检测报错	-	-
引用库	全局变量	编译通过	编译通过	编译通过	编译报错	-
		检测警告	检测警告	检测警告	检测报错	-
	类型	编译通过	编译通过	编译通过	编译通过	编译报错
		检测警告	检测警告	检测警告	检测警告	检测报错

## 3.3 资源使用表

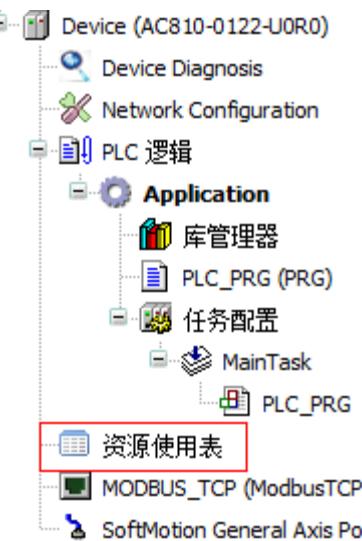
### 3.3.1 概述

资源使用表主要有以下功能：

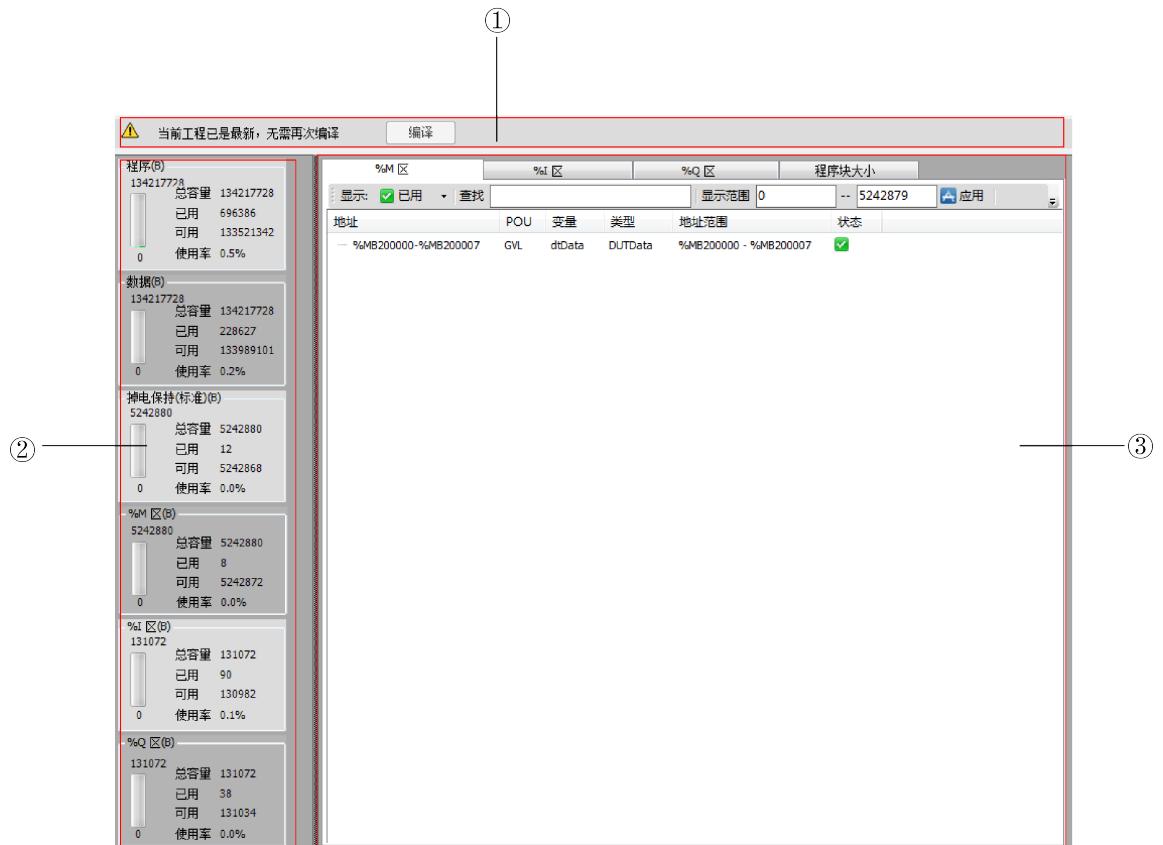
- 显示I/Q/M区地址使用情况，包含已用地址、冲突地址、空闲地址的分类显示。
- 显示程序区、数据区、掉电保持区及I/Q/M区区域的使用大小、可用大小及使用率。

### 3.3.2 功能介绍

新建工程后，用户在设备树上双击“资源使用表”，可以显示资源使用表界面。



界面如下图所示。



## 基本功能

序号	项目	描述
①	数据刷新区	点击界面上“生成代码”按钮，待成功后，界面会根据生成的数据，提取本界面需要的信息，刷新本界面。除此之外，点击“编译”->“生成代码”也会触发界面刷新。 【注】本功能只有在生成代码成功后，才会刷新界面。
②	区域使用信息区	主要显示程序区、数据区、掉电保持区、I/Q/M区的基本信息，包括：区域总容量、已用区大小、可用区大小及区域使用率等信息。 【注】掉电保持区非为传统模式下掉电区和标准模式下掉电区，传统模式下为系统原始分配掉电区，标准模式下则为M区重叠（最大为整个M区，用户可以根据实际情况设置M区范围）。
③	I/M/Q区详细使用信息及程序块信息显示区	I/M/Q区地址的使用情况，包括：地址关联的变量信息、地址冲突情况、未用地址区信息等；程序块大小信息主要包括程序中调用的功能块类型及大小信息。另外支持查找、地址范围设置、页码定位等功能。

其中信息显示区的界面如下图所示。

%M 区		%I 区		%Q 区		程序块大小	
显示：		已用	查找	显示范围		0	131071
<input checked="" type="checkbox"/> 显示： <input checked="" type="checkbox"/> 已用   查找 <input type="text"/>   显示范围 <input type="text" value="0"/> -- <input type="text" value="131071"/> <input type="button" value="应用"/>   <input type="button" value="翻页"/>							
地址	POU	变量	类型	地址范围	状态		
%IB0-%IB1	IS620N	--	UINT	%IB0 - %IB1	<input checked="" type="checkbox"/>		
%IB2-%IB3	IS620N	--	UINT	%IB2 - %IB3	<input checked="" type="checkbox"/>		
%IB4-%IB7	IS620N	--	DINT	%IB4 - %IB7	<input checked="" type="checkbox"/>		
%IB8-%IB9	IS620N	--	INT	%IB8 - %IB9	<input checked="" type="checkbox"/>		
%IB12-%IB15	IS620N	--	DINT	%IB12 - %IB15	<input checked="" type="checkbox"/>		
%IB16-%IB17	IS620N	--	UINT	%IB16 - %IB17	<input checked="" type="checkbox"/>		
%IB20-%IB23	IS620N	--	DINT	%IB20 - %IB23	<input checked="" type="checkbox"/>		
%IB24-%IB27	IS620N	--	DINT	%IB24 - %IB27	<input checked="" type="checkbox"/>		
%IB28-%IB31	IS620N	--	UDINT	%IB28 - %IB31	<input checked="" type="checkbox"/>		
%IB32-%IB33	IS810N	--	UINT	%IB32 - %IB33	<input checked="" type="checkbox"/>		
%IB34-%IB35	IS810N	--	UINT	%IB34 - %IB35	<input checked="" type="checkbox"/>		
%IB36	IS810N	--	SINT	%IB36	<input checked="" type="checkbox"/>		
%IB40-%IB43	IS810N	--	DINT	%IB40 - %IB43	<input checked="" type="checkbox"/>		
%IB44-%IB47	IS810N	--	DINT	%IB44 - %IB47	<input checked="" type="checkbox"/>		
%IB48-%IB49	IS810N	--	UINT	%IB48 - %IB49	<input checked="" type="checkbox"/>		
%IB52-%IB55	IS810N	--	DINT	%IB52 - %IB55	<input checked="" type="checkbox"/>		
%IB56-%IB59	IS810N	--	DINT	%IB56 - %IB59	<input checked="" type="checkbox"/>		
%IB60-%IB63	IS810N	--	DINT	%IB60 - %IB63	<input checked="" type="checkbox"/>		
%IB64-%IB67	IS810N	--	UDINT	%IB64 - %IB67	<input checked="" type="checkbox"/>		
%IB68-%IB69	IS810N	--	UINT	%IB68 - %IB69	<input checked="" type="checkbox"/>		
%IB70-%IB71	IS810N	--	UINT	%IB70 - %IB71	<input checked="" type="checkbox"/>		
%IB72	IS810N	--	SINT	%IB72	<input checked="" type="checkbox"/>		
%IB76-%IB79	IS810N	--	DINT	%IB76 - %IB79	<input checked="" type="checkbox"/>		
%IB80-%IB83	IS810N	--	DINT	%IB80 - %IB83	<input checked="" type="checkbox"/>		
%IB84-%IB85	IS810N	--	UINT	%IB84 - %IB85	<input checked="" type="checkbox"/>		
%IB88-%IB91	IS810N	--	DINT	%IB88 - %IB91	<input checked="" type="checkbox"/>		
%IB92-%IB95	IS810N	--	DINT	%IB92 - %IB95	<input checked="" type="checkbox"/>		
%IB96-%IB99	IS810N	--	DINT	%IB96 - %IB99	<input checked="" type="checkbox"/>		
%IB100-%IB103	IS810N	--	UDINT	%IB100 - %IB103	<input checked="" type="checkbox"/>		

该界面分为4个显示选项，分别是%M区、%I区、%Q区和程序块大小，分别显示对应区域的使用信息。

## 菜单选项

菜单选项中包含：显示选项、查找、范围设置、应用按钮和翻页选项。

序号	项目	描述
1	显示选项	在%M区、%I区、%Q区中，选项内容为：全部、已用、冲突、空闲；在程序块大小显示界面，选项内容为：全部、结构体和功能块。
2	查找选项	可对地址、POU、变量、地址范围四列选项中的名称进行字符匹配，任何一项匹配，即算满足条件。查找选项中内容改变便会触发表格刷新。
3	范围设置	用户可以根据实际需要，查看自己设定的地址范围，范围不可以超出对应区域的极限范围。范围显示需要点击应用选项，才可生效。程序块大小界面中无该项。
4	应用选项	点击该按钮时，系统会根据显示条件、查找内容、及显示范围三个条件进行数据筛选，符合要求的会显示在表格中。
5	翻页选项	为保证表格刷新性能，表格默认最多添加1000个地址段单元，当数据超过1000个地址段时，系统将采用分页显示，可通过点击上/下页或是输入指定页码项，系统便会跳转到对应页面。

## 表格

表格中主要根据筛选条件显示符合条件的地址段使用信息。

序号	项目	描述
1	地址	按照从小到大的顺序，显示地址区，最小单位为Byte，当一个变量所占地址不冲突时，将以地址区域（如“%MB0-%MB3”）的形式显示。一个地址下可能关联了一个变量，也可能关联多个变量，存在多个变量时，变量行将以子节点的形式显示在相应的地址段下方。
2	POU	显示变量所有的POU名称。
3	变量	显示该区域关联的变量名称。
4	类型	显示变量类型。
5	地址范围	显示变量的整个地址区间。
6	状态	<p>显示地址区的使用情况，已用时图标为<span style="color: green;">✓</span>，冲突为<span style="color: red;">✗</span>，空闲下为空。除此之外，<span style="color: blue;">i</span> 表示为系统使用，600和400系列M区存在系统使用区域，范围为%MB491520-%MB524287，当用户将变量关联到此范围区域时，系统不会报错，在本界面将以<span style="color: blue;">i</span> 提示。800系列%M区不存在系统使用区。</p> <p>注意：</p> <ul style="list-style-type: none"> <li>• 地址冲突检测是按照Byte位进行检测的，当不同变量使用了同一地址不同bit位，系统检测时，将其标记为地址冲突，具体冲突情况以实际bit位使用情况为准。</li> <li>• 当用户定义变量与设备分配IO地址冲突时，只表示同一个地址存在多处使用，用户根据实际功能是否使用来判断是否存在问题。</li> </ul>

## PLC直接地址存储区域

不同PLC 提供的直接存储区域不同。对于PLC 数据，%I、%Q 区地址不能掉电保存，对于%M 区可以掉电保存。AM600、AM610、AM401、AM402 编程系统提供128kB (Byte) 的输入区域 (I 区) ，128kB (Byte) 输出区域 (Q 区) 和512kB 存储区域 (M 区) ，其中存储区域中的前480kB 用户可以直接使用，后32k 为系统使用区域 (主要用作软元件) ，用户不要直接使用。编程时，用户可以直接访问地址，也可以定义变量后把变量映射到地址间接访问。存储区域定义及使用的地址范围如下表。

区域	用途	大小	地址范围
I区 (%I) 128kB	用户使用区域	64kWords	%IW0~%IW65535
Q区 (%Q) 128kB	用户使用区域	64kWords	%QW0~%QW65535
M区 (%M) 512kB	用户使用区域	240kWords	%MW0~%MW245759
	用SD元件	10000Words	%MW245760~%MW255759
	用SM元件	10000BytesWords	%MB511520~%MB521519
	保留	2768Bytes	%MB521520~%MB524287

AC800 系列编程系统提供128kB (Byte) 的输入区域 (I 区) , 128kB (Byte) 输出区域 (Q 区) 和5MB 存储区域 (M 区) 。AC800 系列不支持SD 和SM 软元件, %M 区地址可以随意使用。存储区域定义及使用的地址范围如下表。

区域	用途	大小	地址范围
I区 (%I) 128kB	用户使用区域	64kWords	%IW0~%IW65535
Q区 (%Q) 128kB	用户使用区域	64kWords	%QW0~%QW65535
M区 (%M) 5MB	用户使用区域	2.5MWords	%MW0~%MW2321439

## 3.4 符号配置

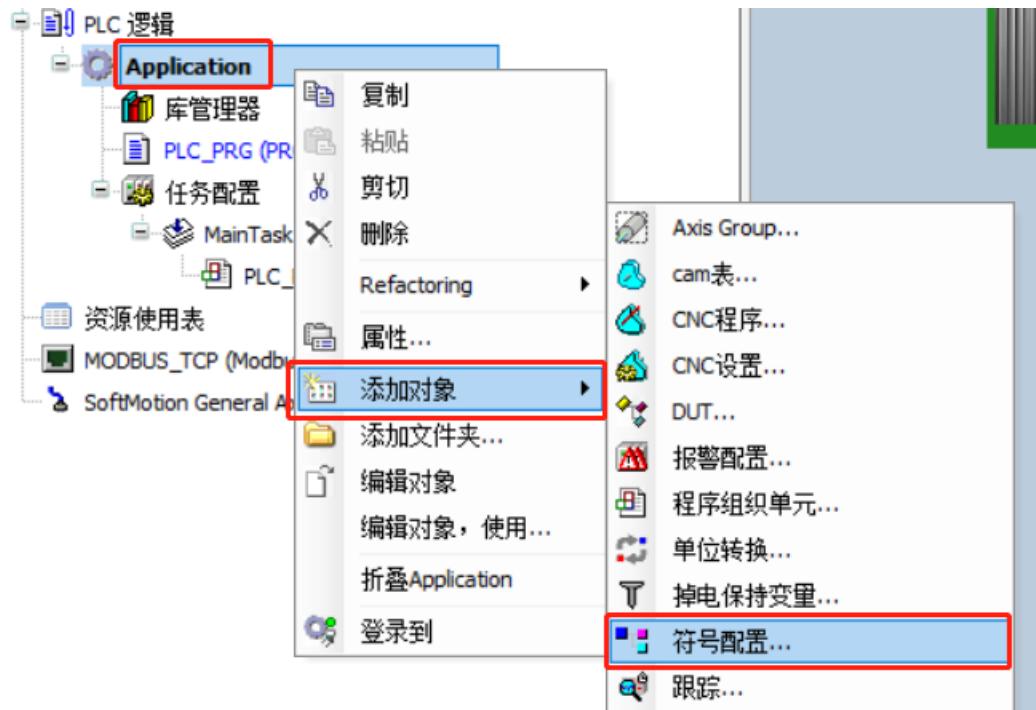
使用符号配置为项目变量准备具有特定访问权限的符号。使用这些符号，可以从外部访问变量，比如OPC服务器。生成代码时，还会生成一个符号配置文件（工程目录下后缀为\*.xml），命名方式为：<project name><设备名称><应用程序名称.xml>，其中包含符号的说明。比如可以导入HMI中进行标签通信，访问变量。

### 添加符号配置

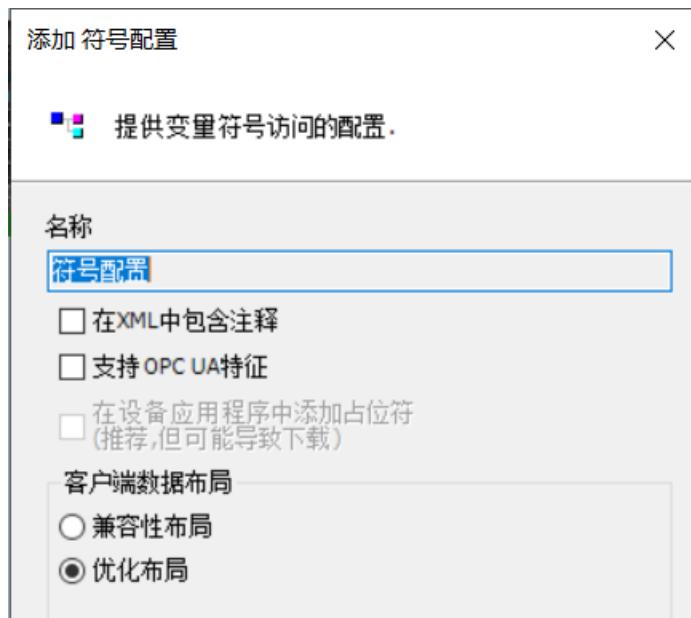
#### 说明

生成符号配置要求编译没有任何错误。

选择工程树下Application，右键选择添加对象，选择符号配置进行添加。



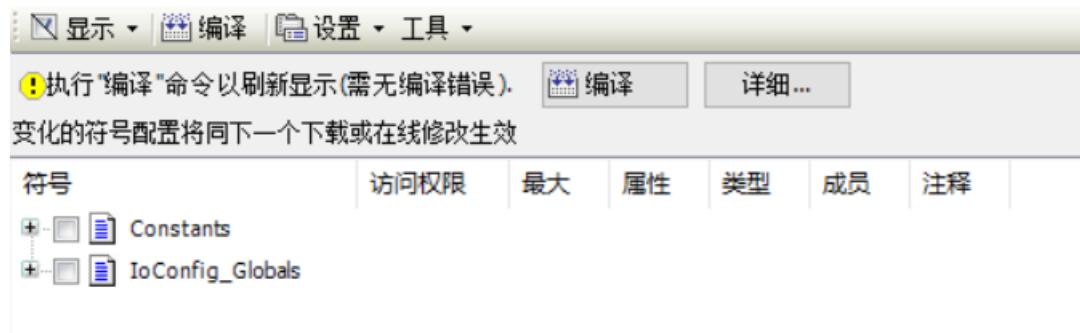
出现弹窗如下图所示：



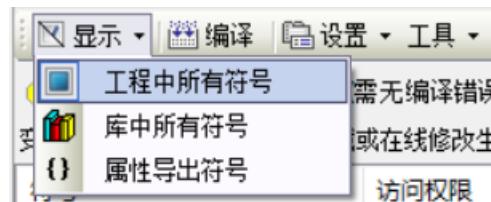
勾选	说明
在XML中包含注释	导出的XML包含变量注释信息
支持OPC UA特征	支持OPC UA访问符号变量
兼容性布局	和类型成员定义偏移大小一致，如果类型成员不完全支持符号访问，偏移大小以实际编译偏移，中间会有空白
优化布局	根据选择的类型成员计算偏移，未选择的不计算偏移

## 符号配置界面介绍

生成后符号配置界面如下图所示：



显示选项，可对下方符号标签进行显示。

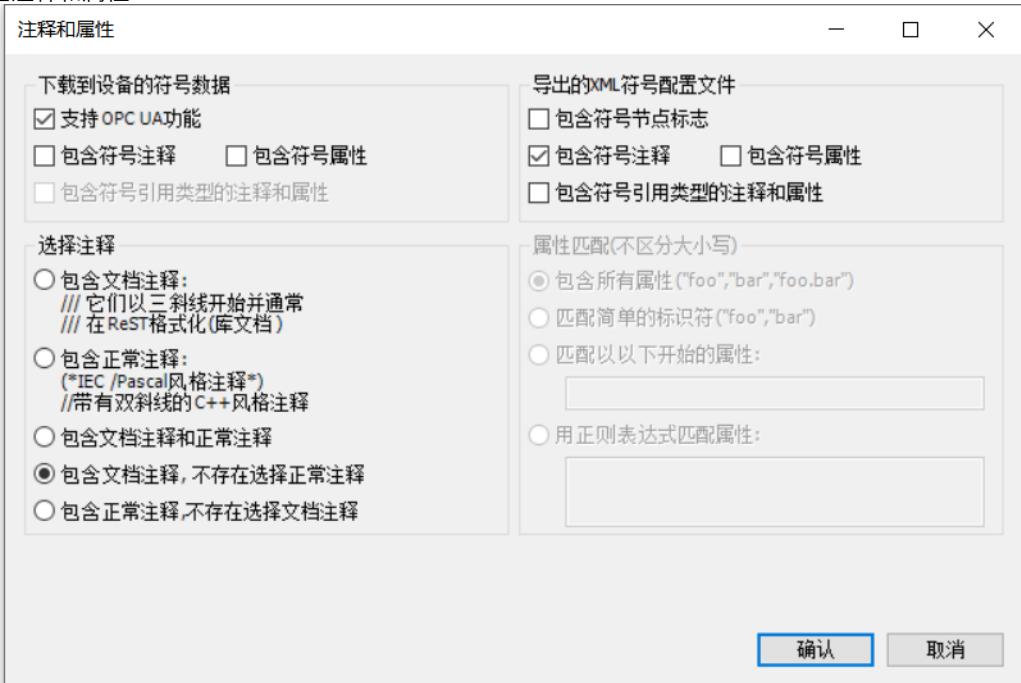


名称	说明
工程中所有符号	显示工程的所有变量
库中所有符号	显示引用库的所有变量
属性导出符号	显示属性控制导出的变量 ({attribute 'symbol' := read})

设置选项，如下图所示。



- 配置注释和属性



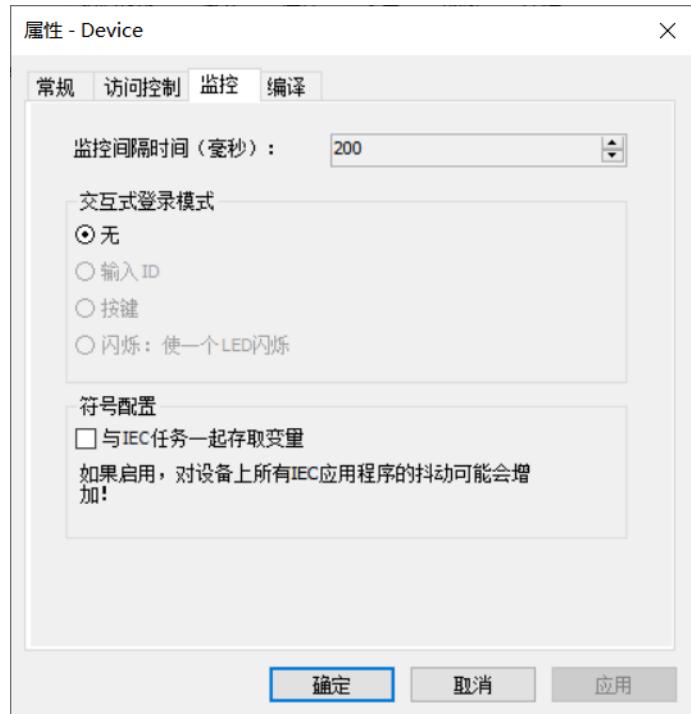
左上侧表示下载到PLC的数据，右上侧表示导出XML数据格式。

符号：一般表示变量，符号属性表示变量特性（Attribute信息）。

注释格式：表示注释下载或者显示格式。

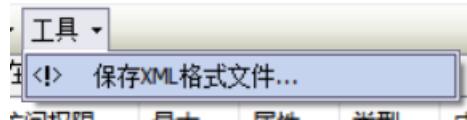
属性匹配：导出XML或者下载到PLC时，哪些属性信息是否包含。匹配规则三种为：所有属性、属性只能是标识符格式，以特定字符串开始和正则表达式。

- 配置和IEC任务同步



其他接口访问符号变量时，是否和IEC同步，不能在IEC执行期间访问，以防止变量不同步。

- 工具选项。



导出XML数据模型，如果第三方解析离线符号，可以参考此数据模型。

## 符号配置过程样例

新建全局变量A\_0、A\_1、A\_2，并且在用户程序中至少应用一个变量，如下图所示。

### 说明

如果单个全局变量表中任一变量都没有在用户程序中应用，则对应的变量表不会出现在符号配置中。

```
VAR_GLOBAL
  A_0:ARRAY[0..9] OF INT;
  A_1:ARRAY[0..19] OF DINT;
  A_2:BOOL;           A_2:=1;
END_VAR
```

在Application中添加符号配置，并且勾选“在XML中包含注释”，点击上侧工具栏界面中选择-检查程序，对应变量表及变量出现在符号配置中，对所需要配置的变量表进行勾选，并且配置好对应的访问权限（只读

、只写、可读可写）后在上侧工具栏界面中选择-编译（生成代码）。

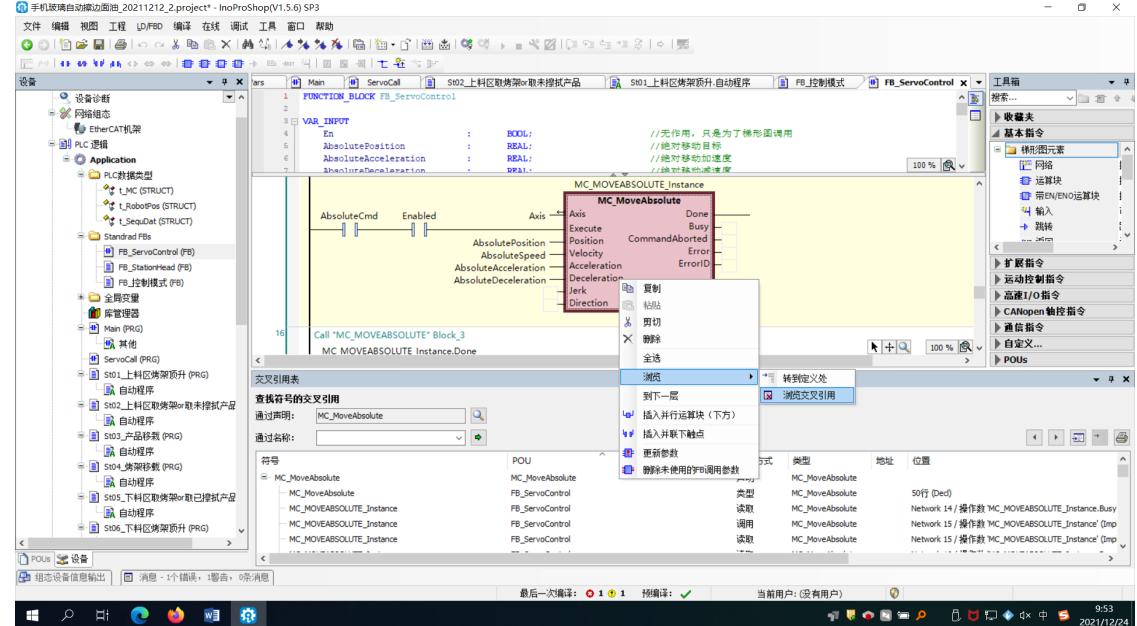


可在工程所在目录下找到生成后缀为.xml的文件，用来导入到IT7000中进行标签通信。

## 3.5 交叉引用

通过“交叉引用”功能可以快速查找“目标对象”在整个工程中的调用位置。

- 找到需要“交叉引用”的对象，鼠标右键调出“交叉引用”。



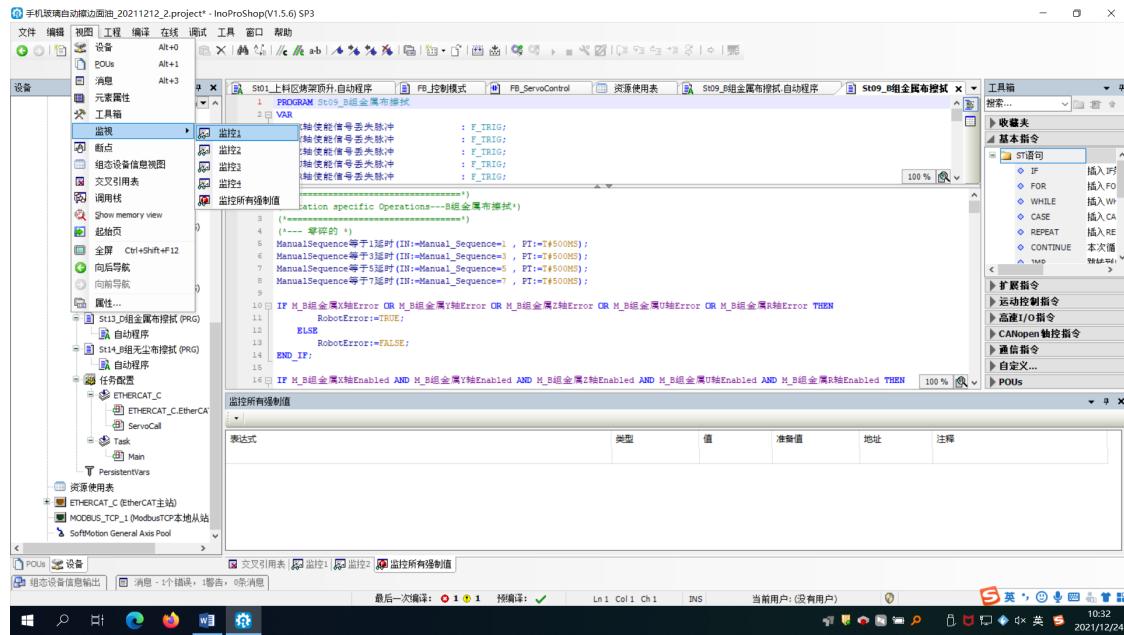
- 在工程下方“交叉引用表”查看“目标对象”在整个工程中的使用情况，双击交叉引用表中的信息可以跳转到工程中具体的使用位置。

符号	POU	类型	地址	位置	对象
= MC_MoveAbsolute	MC_MoveAbsolute	声明	MC_MoveAbsolute	50行 (Def)	MC_MoveAbsolute
- MC_MoveAbsolute	FB_ServoControl	类型	MC_MoveAbsolute	Network 14 / 操作数 'MC_MOVEABSOLUTE_Instance.Busy' (Imp)	FB_ServoControl [Device: PLC]
- MC_MOVEABSOLUTE_Instance	FB_ServoControl	读取	MC_MoveAbsolute	Network 15 / 操作数 'MC_MOVEABSOLUTE_Instance' (Imp)	FB_ServoControl [Device: PLC]
- MC_MOVEABSOLUTE_Instance	FB_ServoControl	调用	MC_MoveAbsolute	Network 15 / 操作数 'MC_MOVEABSOLUTE_Instance' (Imp)	FB_ServoControl [Device: PLC]
- MC_MOVEABSOLUTE_Instance	FB_ServoControl	读取	MC_MoveAbsolute	Network 16 / 操作数 'MC_MOVEABSOLUTE_Instance' (Imp)	FB_ServoControl [Device: PLC]
上料区_顶升伺服轴.JOB_MC_MOVEABSOLUTE_Instance	S01_上料区顶升架提升	读取	MC_MoveAbsolute	116行, 62列 (Imp)	S01_上料区顶升架提升 [Device: PLC]
上料区_顶升伺服轴.JOB_MC_MOVEABSOLUTE_Instance	S01_上料区顶升架提升.自动程序	读取	MC_MoveAbsolute	290行, 160列	上料区_顶升伺服轴.JOB_MC_MOVEABSOLUTE_Instance [Device: PLC]

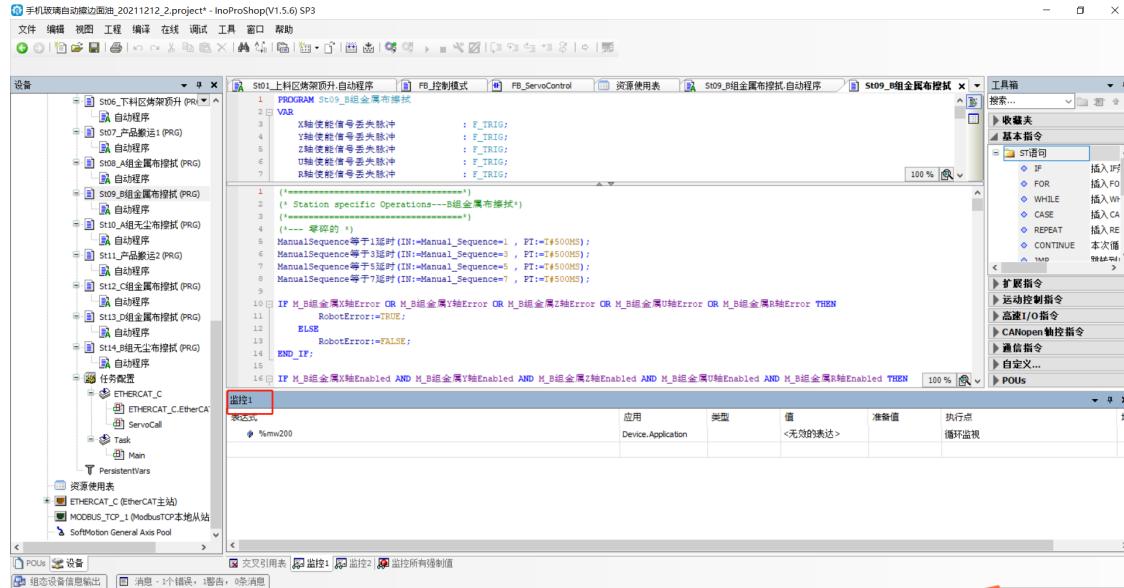
## 3.6 监控表

通过“监控表”功能可以对变量、地址进行监控，在程序运行中可以通过监控表查看监控变量的数据类型、当前值以及可以通过写入值为变量进行赋值。

- 在工具栏视图监视监控表中添加监视视图。



## 2. 在工程栏下方添加想要监控的变量或者地址。

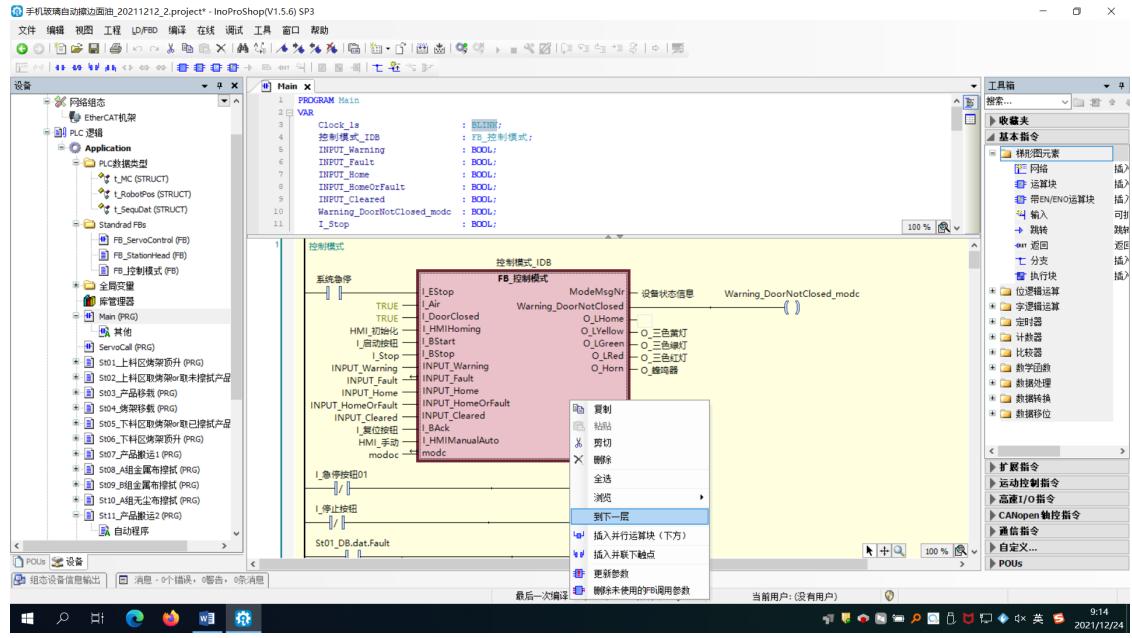


## 3.7 转到下一层

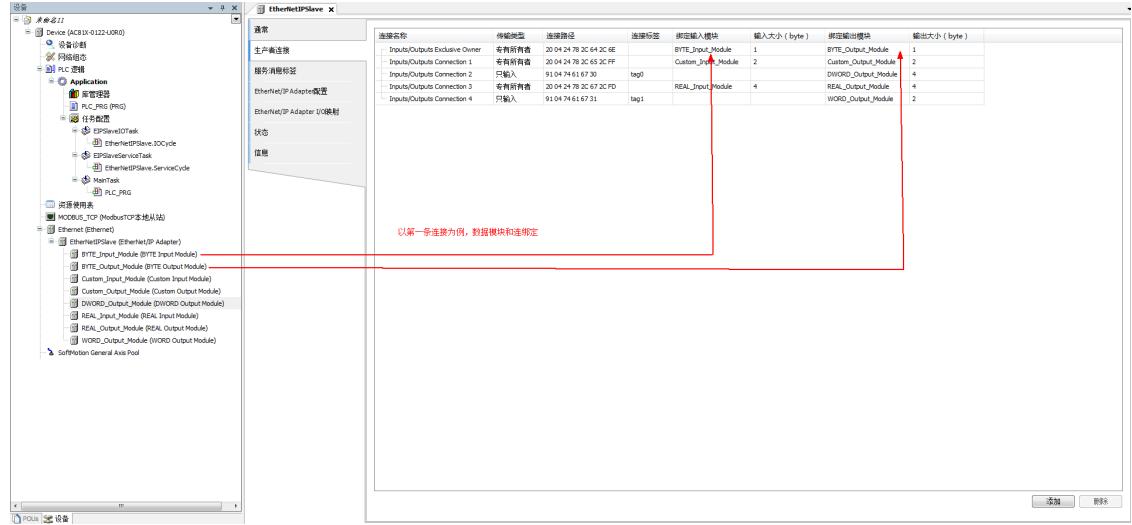
### 自定义功能块

1. 找到目标功能块，鼠标右键弹出“到下一层”，点击“到下一层”。

## 基本功能

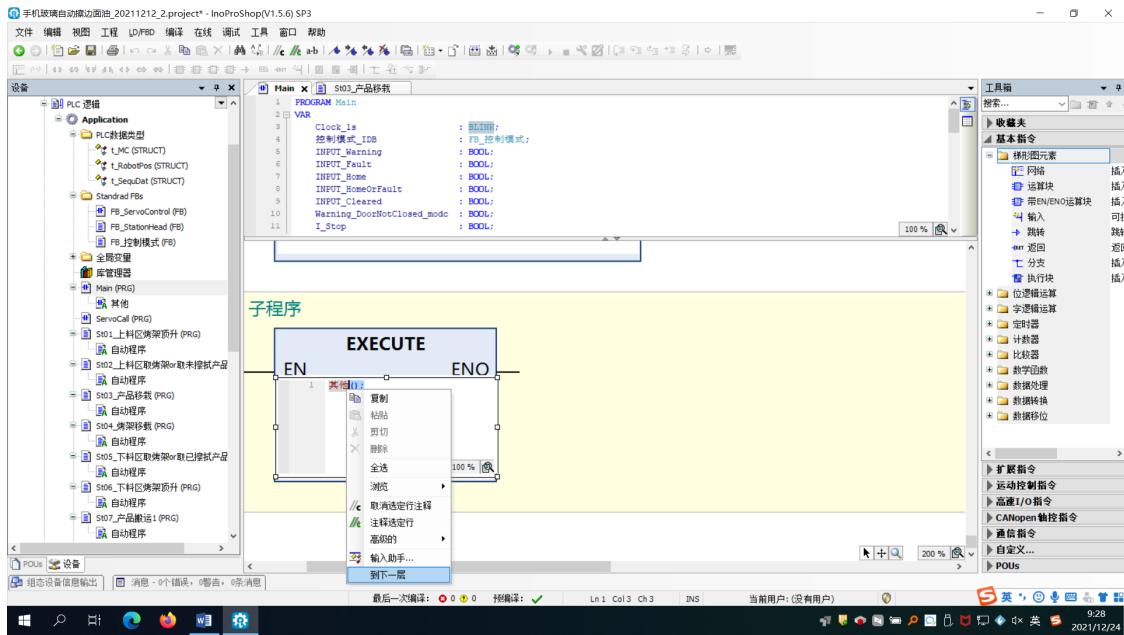


### 2. 转到功能块内部。

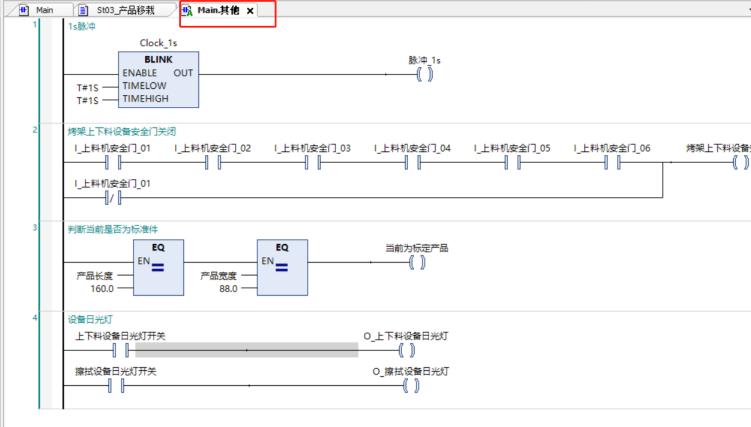


## 在程序组织单元下新建的“动作”

### 1. 找到目标“动作”，鼠标右键弹出“到下一层”，点击“到下一层”。

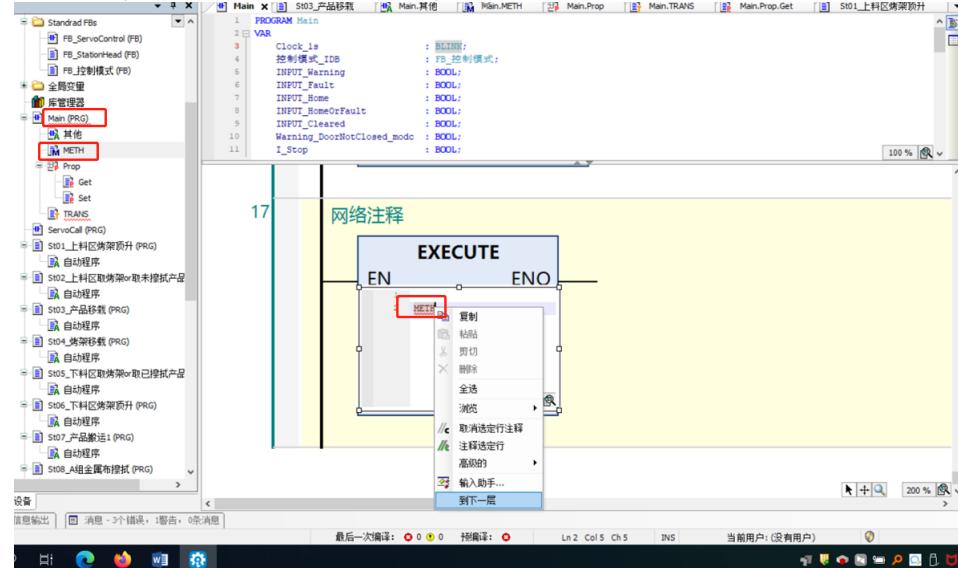


## 2. 转到“动作”内部。



## 在程序组织单元下新建的“方法”

### 1. 找到目标“方法”，鼠标右键弹出“到下一层”，点击“到下一层”。



2. 转到 “方法” 内部。

Main.METH X PersistentVars Main

```
1 METHOD METH
2 VAR_INPUT
3 END_VAR
4
5 VAR
6 上料区烤架有无到位延时: INT;
7 IN: INT;
8 PT: INT;
9 上料区烤架到位延时: INT;
10 上料区烤架有无丢失延时: INT;
11 上料区烤架到位丢失延时: INT;
12 A: TIME;
13
14 (*=====
15 (* Station specific Operations---上料区烤架顶升*)
16 (*=====
17 (*+++++伺服到达位置反馈*)
18 (*--- 伺服到达位置反馈 CurrentPosition>0.0+1.0) AND (M_上料区顶升伺服CurrentPosition<0.0+1.0) THEN
19 M_上料区顶升伺服到达Home位:=TRUE;
20 ELSE
21 M_上料区顶升伺服到达Home位:=FALSE;
22 END_IF;
23
24 IF (M_上料区顶升伺服CurrentPosition>M_上料区顶升伺服上限位-1.0) AND (M_上料区顶升伺服CurrentPosition<M_上料区顶升伺服上限位+1.0) THEN
25 M_上料区顶升伺服到达上限位:=TRUE;
26 ELSE
27 M_上料区顶升伺服到达上限位:=FALSE;
28 END_IF;
29
30 END_IF;
```

3.8 工程版本升级

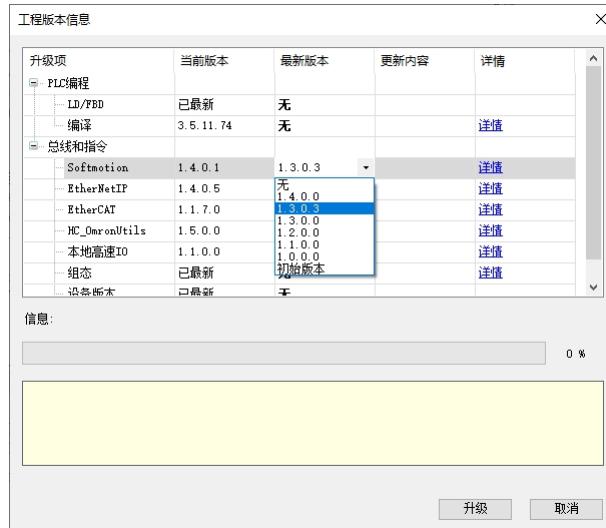


InoProShop V1.8.0.0及以上版本支持该功能。

操作步骤

## 1. 打开工程版本升级对话框。

- 在InoProShop V1.8.0.0及以上版本打开旧工程，如果旧工程版本存在可升级项，则会自动弹出“工程版本信息”对话框。
  - 在菜单栏选择“工程 > 工程版本信息”，打开“工程版本信息”对话框。
  - 在状态栏双击，打开“工程版本信息”对话框。



2. (可选) 如需指定Softmotion库或EtherCAT库升级的版本号, 在“最新版本”列单击对应Softmotion库或EtherCAT库待升级的版本号, 默认选择升级至最新版本。
3. 单击“升级”, 所有升级项将升级至最新版本 (Softmotion库或EtherCAT库如指定升级的版本号, 则Softmotion库或EtherCAT库版本除外), 同时将备份旧工程到原工程目录下。
4. (可选) 如需查看升级详情, 单击“详情”。

### 升级说明

- 编译器

编译器不升级则保持原有版本, 升级则工程编译器和欧姆龙库版本自动相匹配, 工程编译器版本与欧姆龙库版本映射关系如下表所示, 版本匹配后工程编译才能成功。

HC_OmronUtils库版本	编译器版本
1.1.0.0 (初始版本)	<ul style="list-style-type: none"> <li>• 3.5.11.10</li> <li>• 3.5.11.11</li> </ul>
1.2.0.0及以上	3.5.11.70及以上

---

### 说明

在“编译选项”界面中编译器版本的升级选项已禁用, 打开“编译选项”界面方式: 在菜单栏选择“工程 > 工程设置”, 单击“编译选项”。

---

- LD/FBD编程语言

LD/FBD编程语言不升级则保持原有版本, 升级则升级至最新版本, 在InoProShop 1.5.2及以上版本建立的工程中默认为最新版本。

- SoftMotion库

打开工程后SoftMotion库的版本状态。

原有SoftMotion版本	打开工程后SoftMotion版本
1.0.0.0~1.3.0.3	默认自动升级到最新版本
初始版本	保持初始版本
1.4.0.0及以上	保持原有版本

选择升级则升级至最新版本。

- EtherNet/IP库

打开工程后EtherNet/IP库的版本状态。

原有EtherNet/IP版本	打开工程后EtherNet/IP版本
初始版本	<ul style="list-style-type: none"> <li>• 打开InoProShop V1.7.3 SP5版本之前建立的工程, 且工程没有在InoProShop V1.7.3 SP5/SP6版本保存过: 默认自动升级到最新版本</li> <li>• 打开工程, 且工程在InoProShop V1.7.3 SP5/SP6版本保存过: 保持初始版本</li> </ul>
1.0.0.0及以上	保持原有版本

选择升级则升级至最新版本。

- EtherCAT库

EtherCAT库不升级则保持原有版本, 升级则升级至最新版本。

- 欧姆龙库 (HC\_OmronUtils)

打开工程后欧姆龙库的版本状态。

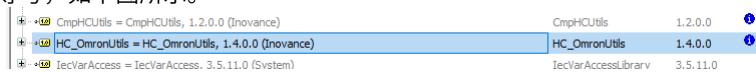
原有欧姆龙库版本	打开工程后欧姆龙库版本
初始版本	保持原有版本
1.0.2.0及以上	工程直接引用欧姆龙库和第三方库中的欧姆龙库，保持原有版本

选择升级或新建工程，则工程直接引用欧姆龙库升级到最新版本，第三方库中的欧姆龙库保持原有版本；如对第三方库中的欧姆龙库进行升级，则找到对应的库源码工程进行升级即可。

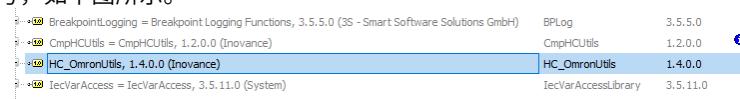
在InoProShop V1.8.0.0版本新建工程或者升级工程后，直接引用的欧姆龙库版本号需与工程版本信息中的欧姆龙版本号一致。如果通过库管理器修改欧姆龙版本号与工程版本信息中的不一致，编译将报错，不建议通过库管理器对欧姆龙库进行版本管控。

不推荐第三方库中使用欧姆龙的占位符库，建议使用嵌入库，因占位符库会随着工程中的占位符配置而改变版本号。

- 占位符库带有等号，如下图所示。



- 嵌入库带有逗号，如下图所示。



- 本地高速IO库

本地高速IO库不升级则保持原有版本，升级则升级至最新版本。

- 组态

对工程中的网络组态、硬件组态和设备诊断功能进行优化升级，在InoProShop 1.5.2及以上版本建立的工程中默认为最新版本。

- 设备版本

设备版本不升级则保持原有版本，升级则升级至最新版本。

## 3.9 工程安全管理

### 3.9.1 加密工程文件

工程文件支持通过设置密码进行保护，防止未经允许的情况下被他人使用。设置工程文件密码后，用户在打开此工程时需要输入密码进行校验，校验通过后才可正常使用该工程文件。

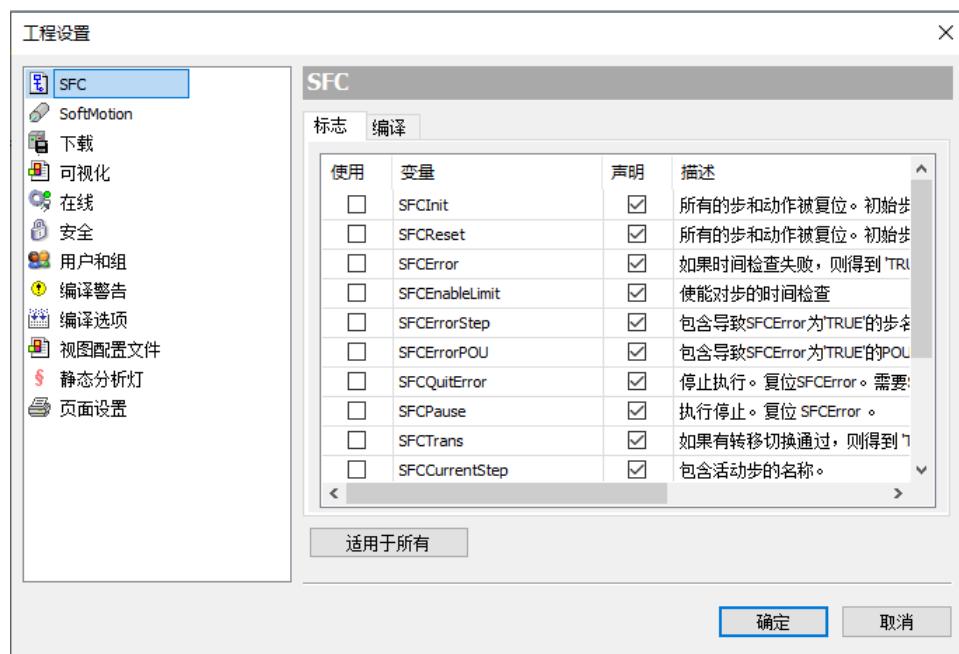


注意

加密工程文件后，务必牢记该密码，如果遗忘密码，密码将无法找回，工程文件将会永久丢失。

#### 操作步骤

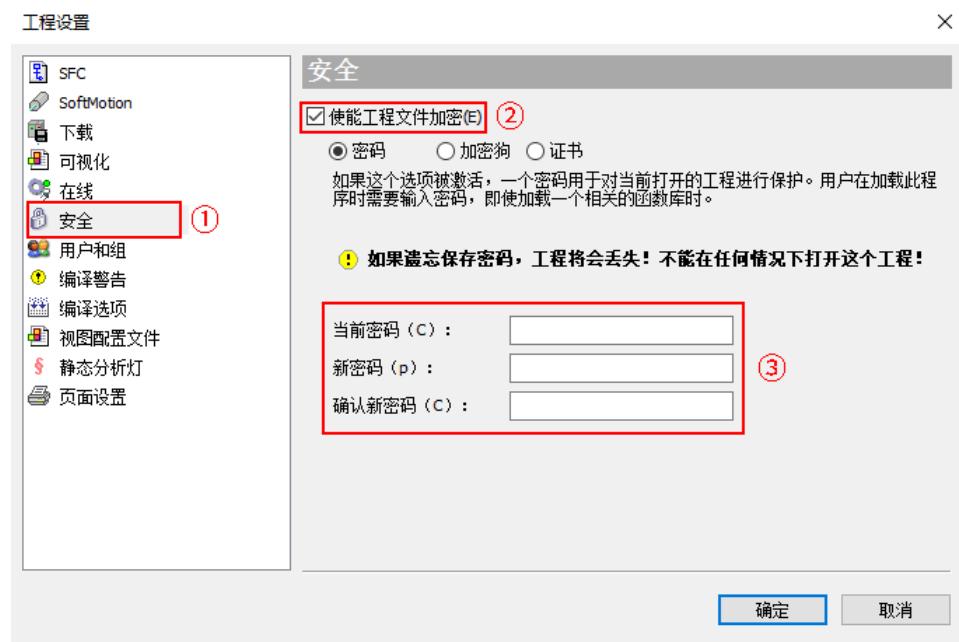
- 在菜单栏选择“工程 > 工程设置”，打开“工程设置”对话框。



2. 单击“安全”，勾选“使能工程文件加密”复选框，输入当前密码、新密码和确认新密码，单击“确定”，完成加密工程文件。

## 说明

首次设置工程文件密码时，当前密码为空，无需输入当前密码。



## 后续操作

1. 重新打开该工程，打开“密码”对话框。



2. 输入工程文件密码，单击“确定”。

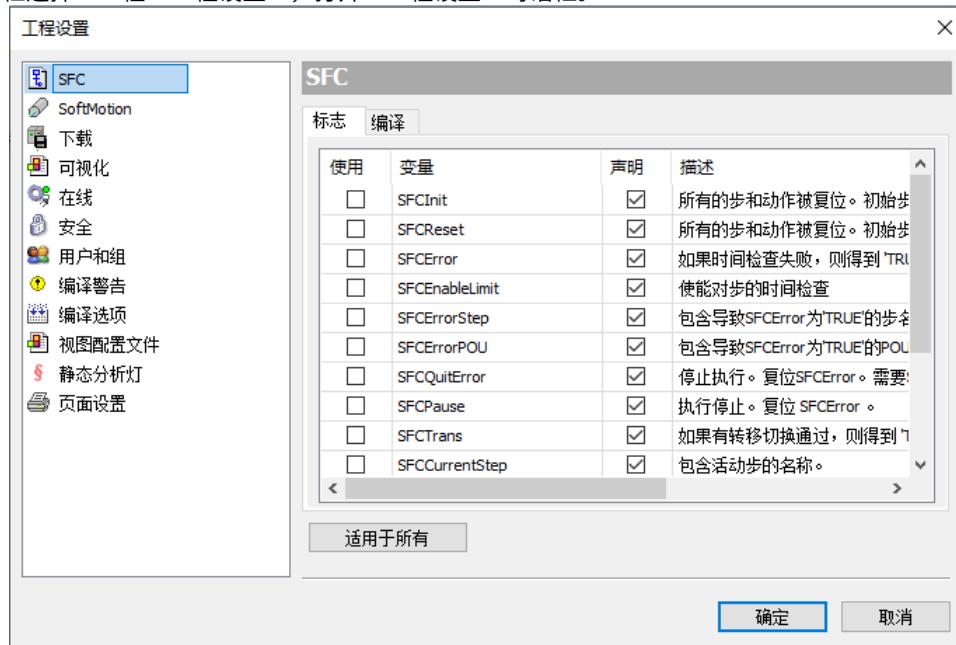
### 3.9.2 工程用户权限管理

该功能用于配置当前工程的用户权限，工程中不同功能的修改、浏览、添加或删除子项和移除操作可通过用户权限进行管理。

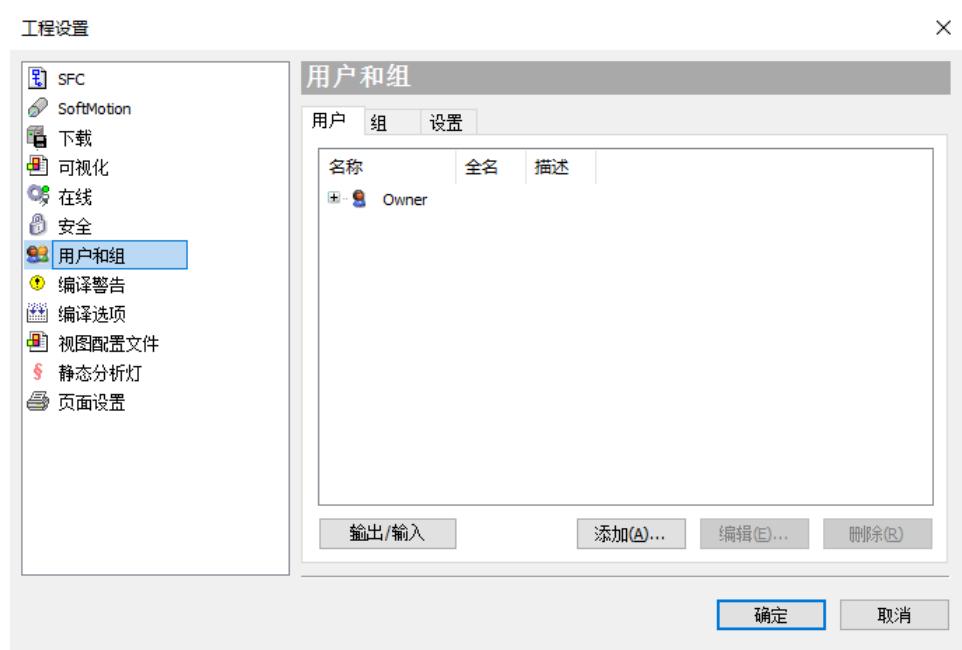
工程用户权限管理分为用户和组，系统默认两个组（Everyone和Owner）和一个用户（Owner），用户是组的成员，组也可以是另一个组的成员。组Everyone和Owner不能被删除，只能重命名，所有新添加的用户都会自动分配到Everyone组中。用户Owner不能被删除，对应初始密码为空，可以重命名和修改密码，强烈建议修改Owner的初始密码，否则任何用户都可以越过其他帐户使用Owner帐户登录，使分配的权限不起作用。

添加用户和组完成后，需要对设备树节点功能进行分配权限，权限以组为单位进行分配，默认分配Everyone组，缺省权限为授权；用户登录工程时将根据分配的权限进行管理工程。

1. 在菜单栏选择“工程 > 工程设置”，打开“工程设置”对话框。

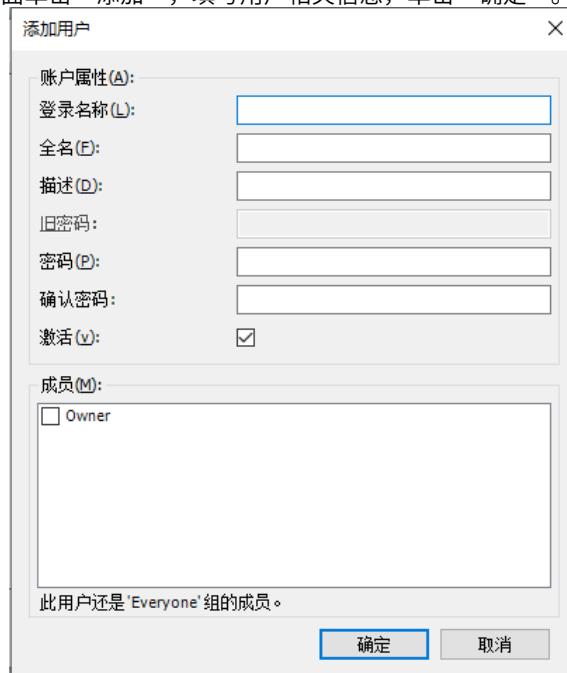


2. 单击“用户和组”，进入“用户和组”管理界面。



## 添加用户

- 在“用户和组 > 用户”界面单击“添加”，填写用户相关信息，单击“确定”。



## 说明

- 新用户默认为“Everyone”组的成员，如需加到其他组，勾选组前复选框。
- 编辑用户：在“用户和组 > 用户”界面选中用户名，单击“编辑”，在打开的对话框中修改用户信息，单击“确定”。
- 删除用户：在“用户和组 > 用户”界面选中用户名，单击“删除”。

- (可选) “用户名”输入“Owner”(初始密码为空)，单击“登录”。



### 说明

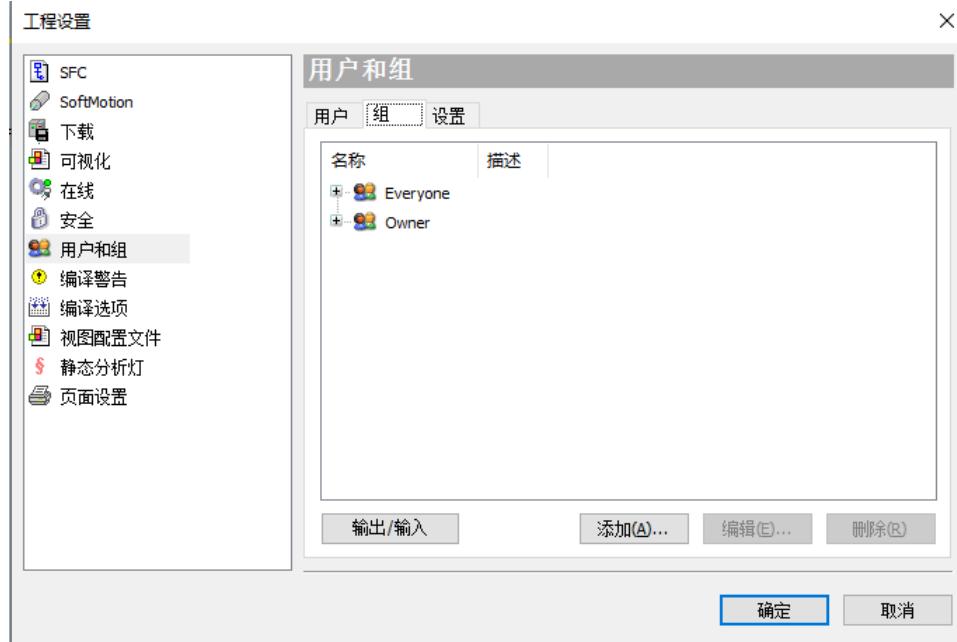
工程打开首次添加/编辑/删除用户需要由“Owner”组用户授权（系统默认创建Owner用户），授权通过后无需再进行授权。

### 导出/导入用户

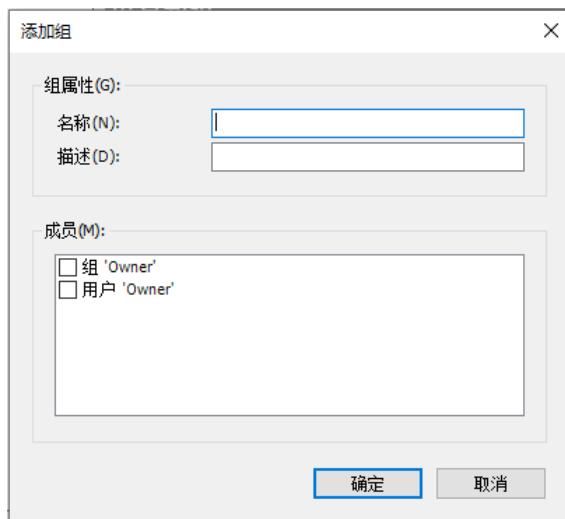
- 导出用户：在“用户和组 > 用户”界面单击“输出/输入”，选择“输出用户和群组”，在打开的界面中选择保存至本地的路径，单击“保存”。
- 导入用户：在“用户和组 > 用户”界面单击“输出/输入”，选择“输入用户以及群组”，在打开的界面中选择保存在本地路径的用户保存文件，单击“打开”。

### 添加组

- 在“用户和组”界面单击“组”，切换至组管理界面。



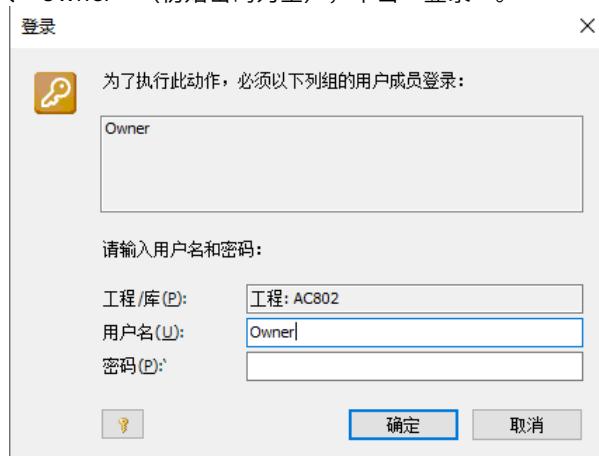
- 在“用户和组 > 组”界面单击“添加”，填写组相关信息，单击“确定”。



## 说明

- 系统默认创建“Everyone”组和“Owner”组，一个组也可以是另一个组的成员。
- 编辑组：在“用户和组 > 组”界面选中组名，单击“编辑”，在打开的对话框中修改组信息，单击“确定”。
- 删除组：在“用户和组 > 组”界面选中组名，单击“删除”。

3. (可选) “用户名”输入“Owner”(初始密码为空)，单击“登录”。



## 说明

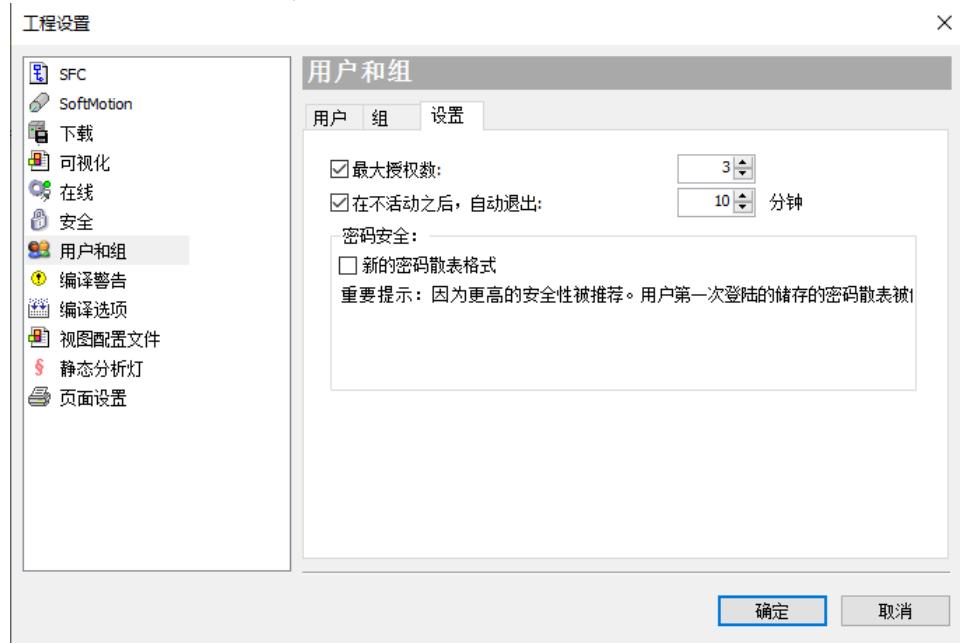
工程打开首次添加/编辑/删除组需要由“Owner”组用户授权（系统默认创建Owner用户），授权通过后无需再进行授权。

## 导出/导入组

- 导出用户：在“用户和组 > 组”界面单击“输出/输入”，选择“输出用户和群组”，在打开的界面中选择保存至本地的路径，单击“保存”。
- 导入用户：在“用户和组 > 组”界面单击“输出/输入”，选择“输入用户以及群组”，在打开的界面中选择保存在本地路径的用户保存文件，单击“打开”。

## 设置

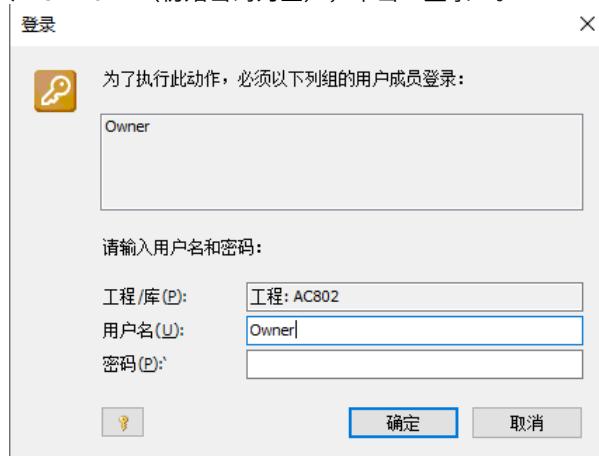
1. 在“用户和组”界面单击“设置”，切换至设置界面。



2. 填写设置相关信息，单击“确定”。

参数名称	如何理解	如何设置
最大授权数	用户登录进行授权时，尝试使用密码进行登录授权的次数。超过该次数，则该用户帐户将被停用。	根据所需设置 缺省值：3
在不活动之后，自动退出	如果在此处指定的时间段（分钟）内未通过鼠标或键盘进行任何用户操作，该用户将自动退出。	根据所需设置 缺省值：10

3. (可选) “用户名”输入“Owner”(初始密码为空)，单击“登录”。



## 说明

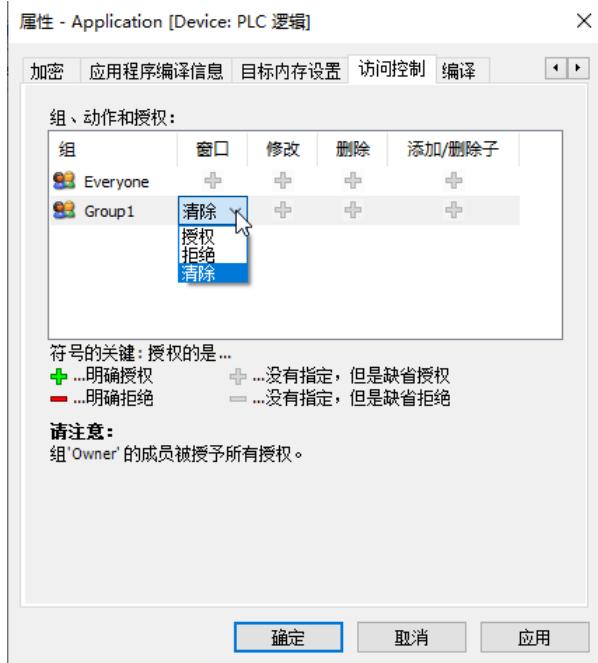
工程打开首次管理设置界面时需要由“Owner”组用户授权（系统默认创建Owner用户），授权通过后无需再进行授权。

## 分配权限

- 在左侧设备树中右键单击待授权的节点，以“Application”节点为例，选择“属性”，打开“属性”对话框。



- 单击“访问控制”页签，选择分配权限的组，在不同动作列双击“+”进行选择“授权”、“拒绝”或“清除”，单击“确定”，完成分配权限。



## 用户登录/退出工程

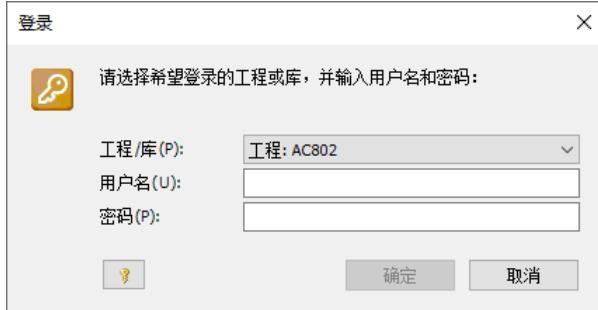
添加用户和组，以及分配权限完成后，用户登录工程才能使用相应节点功能。

用户登录/退出工程包含两种方式：

- 通过菜单栏

- 用户登录工程

1. 在菜单栏选择“工程 > 用户管理 > 用户登录”，打开“登录”对话框。



2. 输入用户名和密码，单击“确定”，登录工程。

- 用户退出工程

在菜单栏选择“工程 > 用户管理 > 用户退出”，退出工程。

- 通过状态栏

- 用户登录工程

1. 在状态栏双击“当前用户：xx”区域，打开“登录”对话框。



2. 输入用户名和密码，单击“确定”，登录工程。

- 用户退出工程

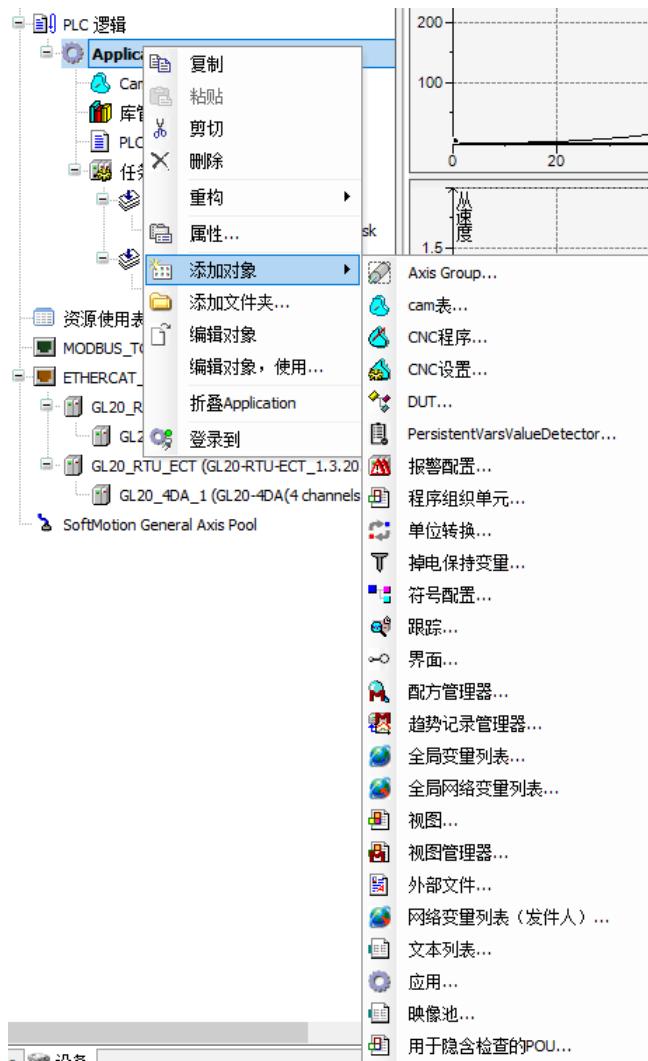
在状态栏双击“当前用户：xx”区域，单击“退出”，退出工程。

## 3.10 Application添加对象

Application支持添加对象功能，支持的功能如下，未列出的功能表示不支持。

- CAM表
- DUT
- 程序组织单元
- 掉电保持变量
- 符号配置
- 跟踪
- 界面
- 全局变量列表
- 应用
- 用于隐含检查的POU

在左侧设备树中右键单击“Application”，选择“添加对象 > XX”，例如选择“添加对象 > cam表”。



## CAM表

CAM表详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > SoftMotion > 对象编辑器 > Cam 编辑器”章节。

## DUT

DUT详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > CODESYS Development System > 参考, 用户接口 > 对象 > “DUT” 对象”章节。

## 程序组织单元

程序组织单元详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > CODESYS Development System > 参考, 用户接口 > 对象 > 对象 ‘POU’ ”章节。

## 掉电保持变量

掉电保持变量详细介绍，具体请参见[第382页 “掉电保持变量”](#)。

## 符号配置

符号配置详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > CODESYS Development System > 参考,用户接口 > 对象 > “符号配置”对象”章节。

## 跟踪

跟踪详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > CODESYS Development System > 参考,用户接口 > 对象 > “跟踪”对象”章节。

## 界面

界面详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > CODESYS Development System > 参考,用户接口 > 对象 > 对象‘POU’ > “接口”对象”章节。

## 全局变量列表

全局变量列表详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > CODESYS Development System > 参考,用户接口 > 对象 > “GVL”对象 - 全局变量列表”章节。

## 应用

应用详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > CODESYS Development System > 参考,用户接口 > 对象 > “应用”对象”章节。

## 用于隐含检查的POU

用于隐含检查的POU详细介绍，具体请参见《在线帮助》中“CoDeSys编程系统 > CODESYS Development System > 参考,用户接口 > 对象 > 对象‘隐式检查POU’”章节。

## 3.11 多人协同



注 意

InoProShop V1.8.1.0及以上版本支持该功能。

---

## 简介

多人协同是指多个用户连接到同一台PLC，协同完成组态及程序逻辑编写工作，主要包括监控、调试、在线修改、在线比较、上传和下载等操作。

支持对同源工程和非同源工程进行多人协同操作，非同源工程不允许进行比较。

---

## 说明

同源工程指原始工程是同一个，迭代版本可不同；非同源工程是指原始工程不是同一个。例如，原始工程A经过多版本迭代升级，该工程为同源工程；原始工程A和原始工程B经过多版本迭代升级，两个工程多次下载至PLC，PLC用户程序以最后下载一次的工程为准。

---

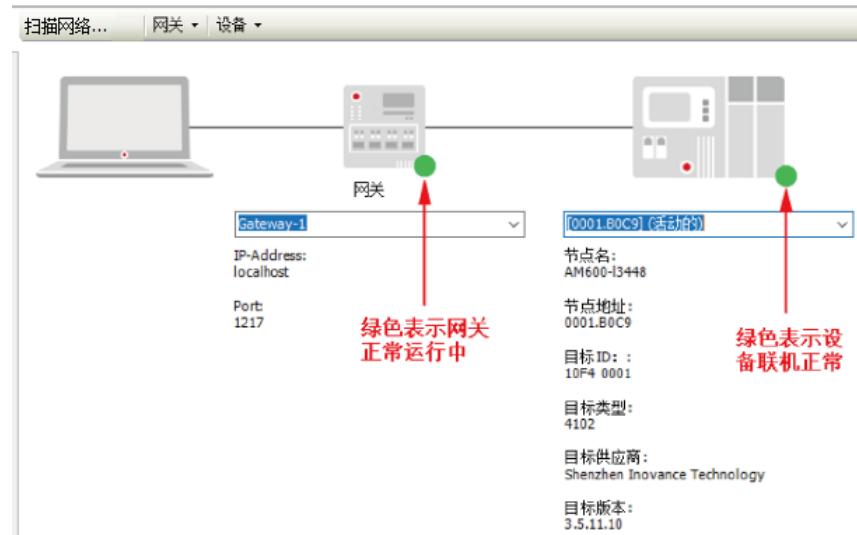
多人协同工作时，PLC作为服务端，可将不同编程人员的控制逻辑下载至PLC，其他编程人员可将PLC最新的控制逻辑上传至本地，建议团队成员之间的工作提前做好分工规划，通过团队协作有序高效完成编程工作。例如成员A编程完将程序下载至PLC，成员B将PLC程序上传至本地再进行编程工作，以此类推。

多人协同支持离线同步和在线同步：

- 离线工程同步：用户可通过菜单栏选择“工程 > 在线比较”，通过在线比较界面同步PLC中的当前工程信息。
- 在线工程同步：用户修改完本地程序后，登录PLC时无论触发在线修改还是全下装，在弹出的提示框中都将显示“在线比较”按钮，可通过该功能同步PLC中的当前工程信息。

## 操作步骤

多用户通过“扫描网络”连接登录同一台PLC，即可进行多人协同工作。



多人协同操作时可能会影响到其他用户，特别是下载的操作（在线修改、完整下载等），界面将给出提示信息。

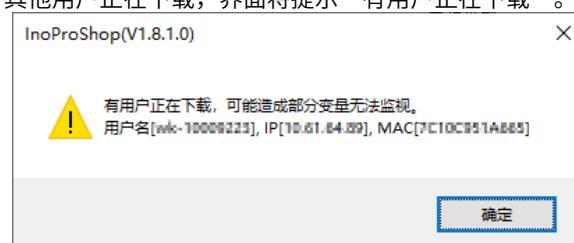
- 支持最大同时登录PLC的用户个数为5个，第6个用户登录时界面将提示“用户已达到上限”。

InoProShop(V1.8.1.0) ×

用户已达到上限

确定

- 用户登录PLC后，如果有其他用户正在下载，界面将提示“有用户正在下载”。



- 用户登录PLC后，如果用户进行下载操作，此时其他用户正在进行监控、调试、在线修改、上传和下载等操作时，界面将提示“有用户正在XX，将强制下载”。



下载程序建议分开进行，如果同时进行，可能存在程序文件损坏的风险。

## 3.12 工程比较



InoProShop V1.8.1.0及以上版本支持该功能。

### 简介

工程比较是指比较不同版本工程之间的差异，并支持合并差异部分。工程比较分为离线工程比较和在线工程比较：

- 离线工程比较：当前打开的工程与磁盘上的工程进行比较。
- 在线工程比较：当前打开的工程与在线PLC内部的程序进行比较。

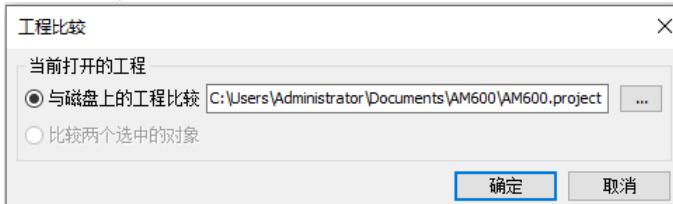
### 说明

- 对象比较时，只能比较两个同类型的对象。
- 在线工程比较时，只能比较当前打开工程的不同版本，且设备类型一致。
- 离线工程比较时，支持不同工程比较。

### 操作步骤（离线工程比较）

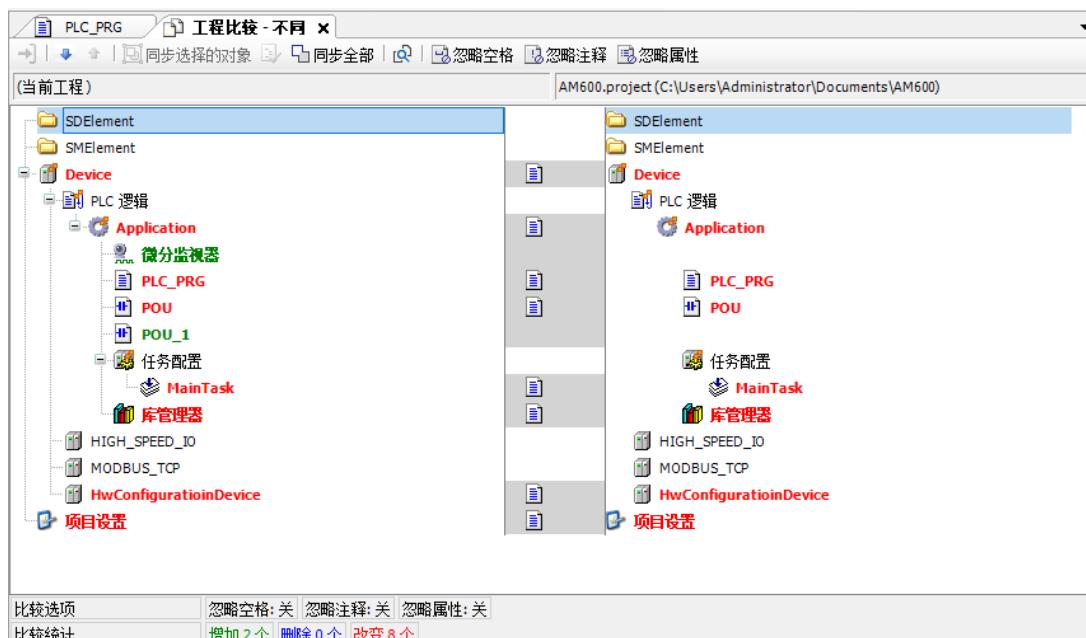
#### • 工程比较

1. 打开InoProShop编程软件并打开本地工程。
2. 在菜单栏选择“工程 > 比较”，打开“工程比较”对话框。



3. 单击“...”，选择磁盘上的工程，单击“确定”，打开“工程比较-不同”界面。

左侧显示当前工程，右侧显示磁盘上的工程；界面底部显示比较选项开启情况和比较统计结果。



工程比较界面工具栏图标功能及含义如下表所示。

图标	含义
	打开对象的详细比较视图。 使能条件：选中的对象内容不同。
	返回至工程对象的比较视图。 使能条件：当前视图是对象详细比较视图。
	选择下一个存在差异的对象/块。 使能条件：选中的对象/块下面存在差异的对象/块。
	选择上一个存在差异的对象/块。 使能条件：选中的对象/块上面存在差异的对象/块。
	同步选择的对象。（支持多选） 使能条件：选中的对象包含未同步的差异。
	恢复选择对象至同步前状态。（支持多选） 使能条件：选中的对象所有差异部分处于同步状态。
	接受对象的属性和权限访问。 使能条件：选中的对象具有不同的属性和访问权限。
	同步所有差异对象。 使能条件：对象列表里包含未同步差异的对象。
	恢复所有差异对象至同步前状态。 使能条件：对象列表里所有差异对象的差异部分处于同步状态。
	切换对象比较模式，选中后，可以选择左右两边任意两个对象做比较。 <b>说明：</b> 选中对象的图标右上角将显示一个放大镜标识。
	选中后，语义相关的空格（例如字符串文字）不忽略外，其他空格都将忽略。
	选中后，将忽略注释的差异。
	选中后，将忽略对象的属性或访问权限的差异。

图标	含义
	接受改变区域。 使能条件：当前鼠标焦点所在区域是未同步的差异区域。
	恢复改变区域。 使能条件：当前鼠标焦点所在区域是同步后的差异区域。

## 说明

同步过程即将磁盘上的工程对象差异部分同步至当前打开的工程对象中。

工程比较界面差异部分使用图标符号和字体颜色区分，具体说明如下表所示。

图标&字体颜色	含义
黑色字体	对象相同，无差异。
比较对象图标右上角显示 	对象的子对象不同，存在差异。
蓝色字体	对象仅在右侧磁盘工程中存在。
绿色字体	对象仅在左侧当前打开的工程中存在。
	对象具有不同的属性。
	对象具有不同的访问权限。
	对象的实现不同。
	将右侧磁盘工程中对象同步至当前打开工程中。
	在当前打开工程中删除对象。
	同步右侧磁盘工程中对象的属性。
	同步右侧磁盘工程中对象的访问权限。
	同步右侧磁盘工程中对象。

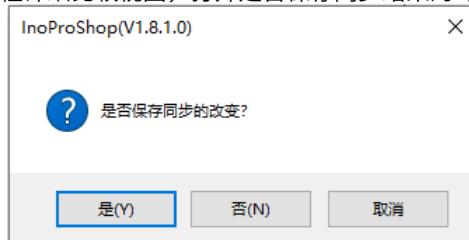
4. 根据不同使用场景执行相应操作。

使用场景	操作步骤
同步选择对象	<ul style="list-style-type: none"> <li>单选同步对象：鼠标单击选中待同步对象，或通过 、 选中，单击 。</li> <li>多选同步对象：按住【Ctrl】键，鼠标逐个单击待同步对象，单击 。</li> </ul> <p><b>说明：</b>如需恢复选择对象至同步前状态，单击 。</p>
同步全部对象	<p>单击 。</p> <p><b>说明：</b>如需恢复所有差异对象至同步前状态，单击 。</p>
同步对象的属性和权限访问	<ol style="list-style-type: none"> <li>选中具有不同属性和权限访问的对象，单击 。</li> <li>在打开的对话框中单击“确定”。</li> </ol>

使用场景	操作步骤
比较任意两个对象	<p>1 单击 ，切换对象比较模式。          2 在左侧工程选择待比较对象，在右侧工程选择待比较对象。          3 在工具栏单击“比较”，进入对象的详细比较视图。          4 选中待详细比较的内容，单击  同步块，同步该差异部分。说明：如需恢复改变区域，单击  恢复块。</p>
开启忽略空格	单击  忽略空格。  说明：如需关闭忽略空格，再次单击该图标即可。
开启忽略注释	单击  忽略注释。  说明：如需关闭忽略注释，再次单击该图标即可。
开启忽略属性	单击  忽略属性。  说明：如需关闭忽略属性，再次单击该图标即可。
进行对象详细比较	<p>1 选中待详细比较的对象，单击  或双击该对象，进入对象详细比较视图。          2 选中待详细比较的内容，单击  同步块，同步该差异部分。说明：如需恢复改变区域，单击  恢复块。</p>

### 5. 保存同步结果。

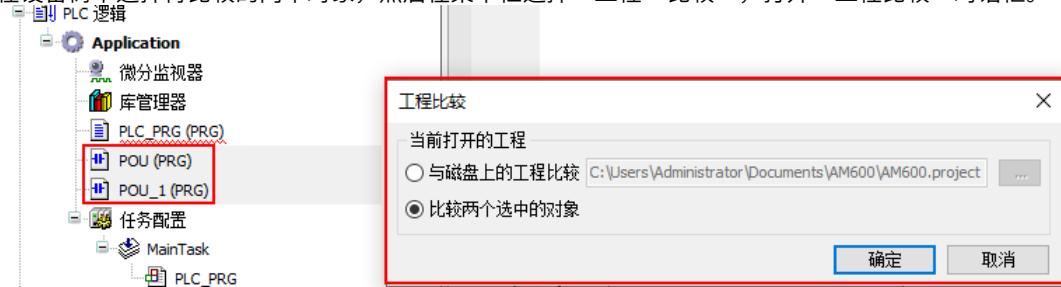
- a. 关闭工程比较视图或关闭工程详细比较视图，打开是否保存同步结果的对话框。



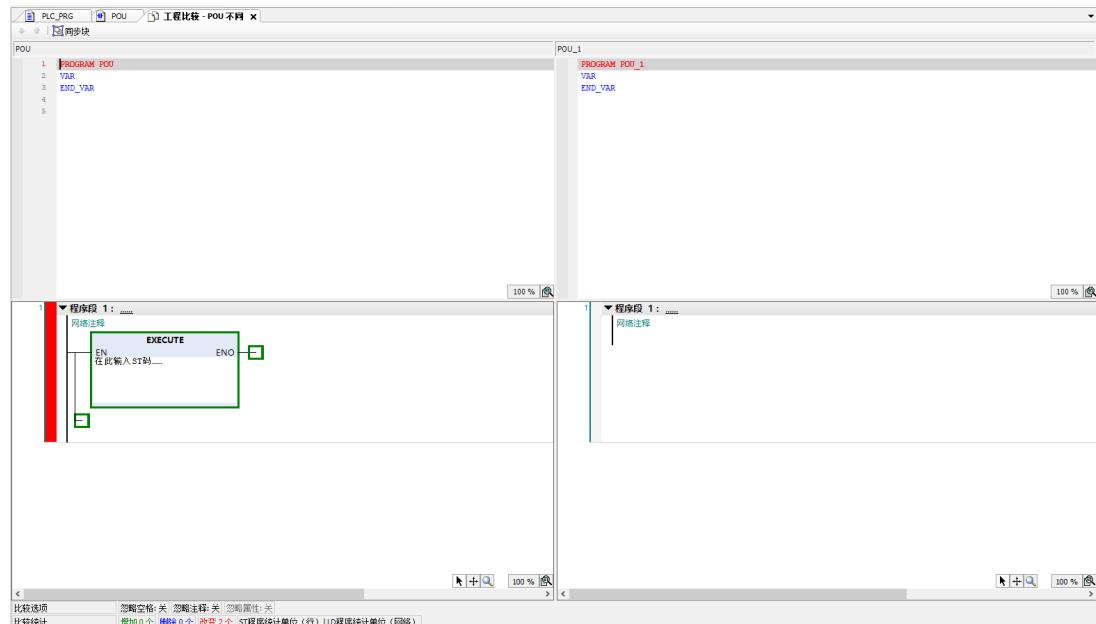
- b. 单击“是”，保存同步结果。

### ● 对象比较

1. 打开InoProShop编程软件并打开本地工程。
2. 在设备树中选择待比较的两个对象，然后在菜单栏选择“工程 > 比较”，打开“工程比较”对话框。



3. 单击“确定”，打开“工程比较”界面。

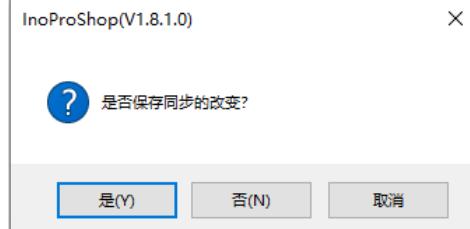


4. 选中待详细比较的内容，单击 同步块，同步该差异部分。

### 说明

如需恢复改变区域，单击 恢复块。

5. 关闭工程详细比较视图，打开是否保存同步结果的对话框。



6. 单击“是”，保存同步结果。

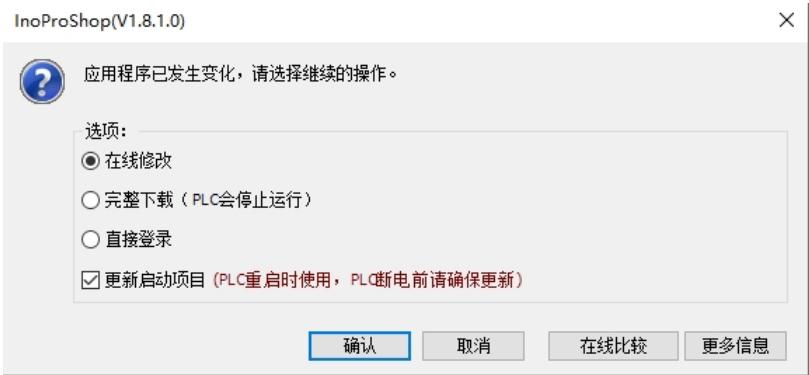
### 操作步骤（在线工程比较）

在线工程比较操作与离线工程比较操作类似，相同操作部分请参考离线工程比较操作，本节只介绍差异部分操作。

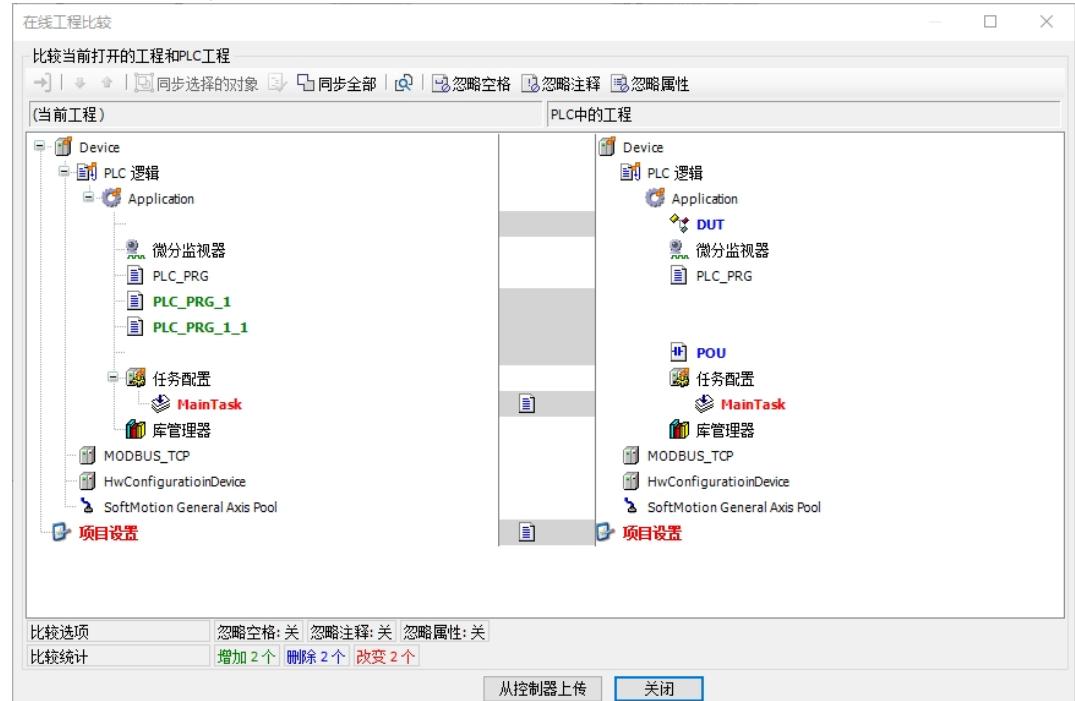
1. 打开在线工程比较界面。

- 用户程序下载时进入

- a. PLC用户程序已发生变化，下载程序时打开如下图所示对话框。



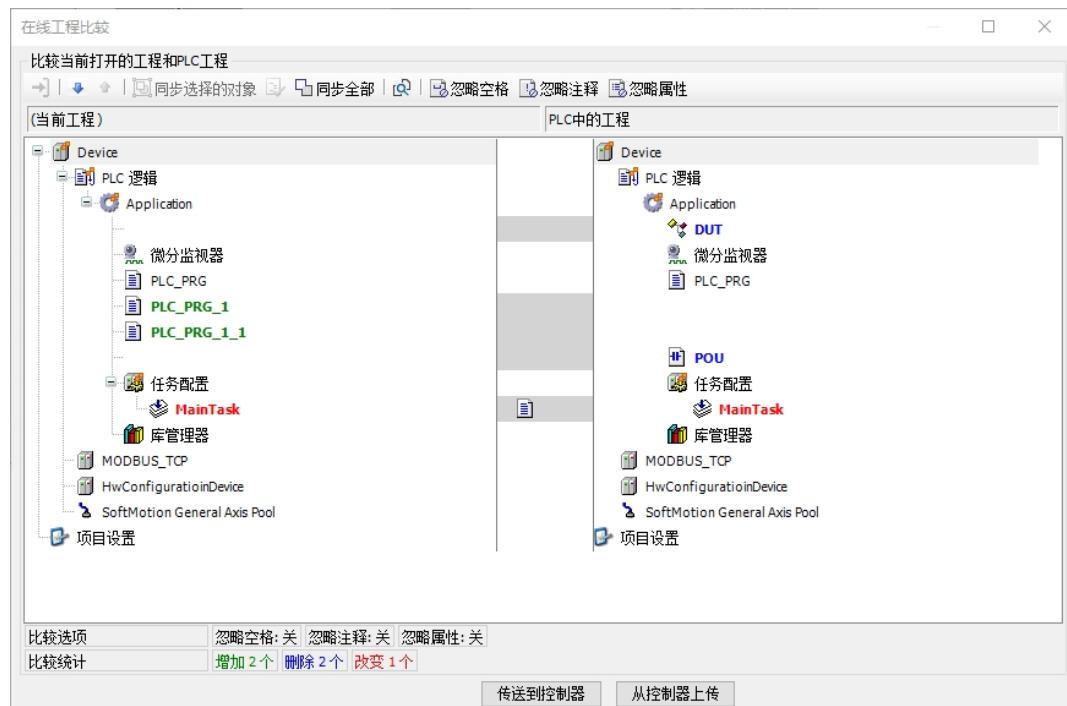
b. 单击“在线比较”，打开“在线工程比较”对话框。



c. 在线工程比较操作具体请参见离线工程比较操作部分内容。

- 菜单栏进入

a. 在菜单栏选择“工程 > 在线比较”，打开“在线工程比较”对话框。



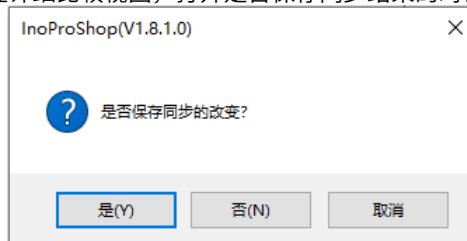
b. 在线工程比较操作具体请参见离线工程比较操作部分内容。

2. (可选) 根据不同使用场景执行相应操作。

使用场景	操作步骤
当连接的PLC工程有更新且需要比较最新的PLC工程时	单击“从控制器上传”，将PLC的工程上传至“在线工程比较”右侧。
当需要将当前工程下载至PLC时（下载规则同多重下载默认参数功能）	单击“传递到控制器”，将当前工程下载至PLC。

3. 保存同步结果。

a. 关闭工程比较视图或关闭工程详细比较视图，打开是否保存同步结果的对话框。



b. 单击“是”，保存同步结果。

### 3.13 微分监视器

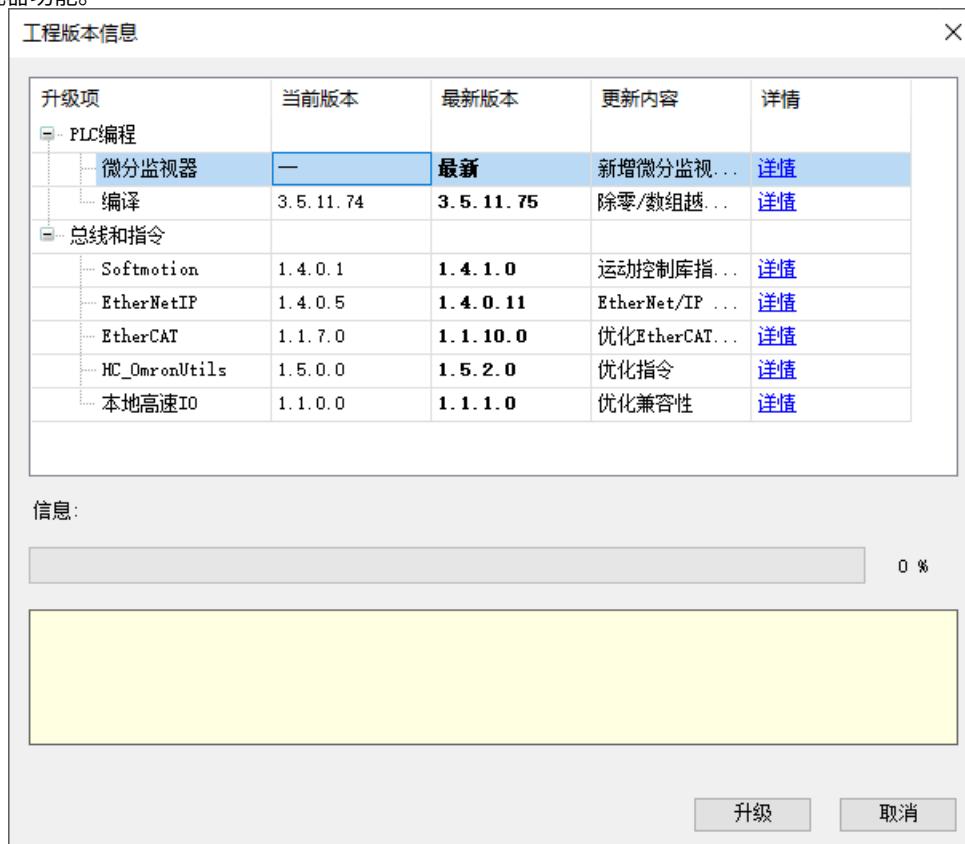


InoProShop V1.8.1.0及以上版本支持该功能。

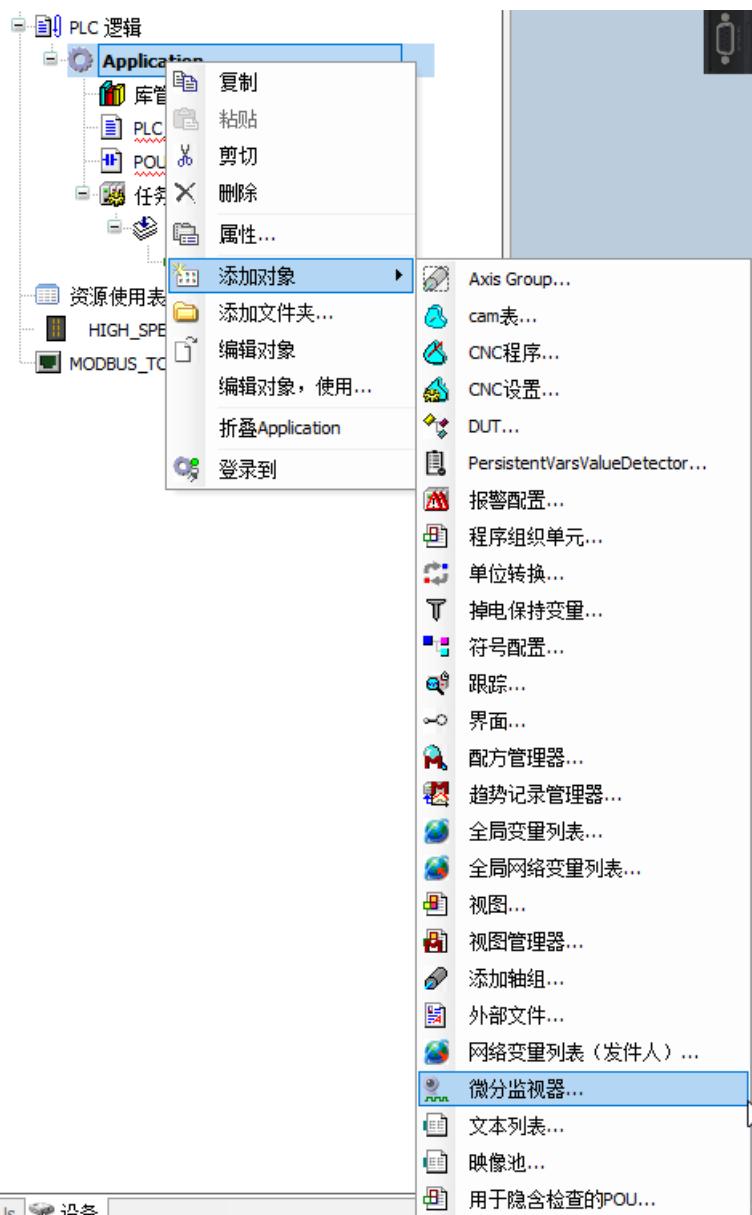
该功能用于监视设定任务周期内BOOL/BIT变量的变化次数，最多只能监控8个变量。

### 工程兼容性说明

- 仅标准工程支持微分监视器（库工程不支持）
- 旧工程对微分监视器的兼容
  - 在InoProShop V1.8.1.0及以上版本打开低于InoProShop V1.8.1.0版本创建的工程，InoProShop软件界面自动打开“工程版本信息”对话框，显示升级项“微分监视器”，单击“升级”将自动添加微分监视器功能。



- 在左侧设备树中右键单击“Application”，在右键菜单中选择“添加对象 > 微分监视器”，在打开对话框中单击“打开”，添加微分监视器功能。



### 操作步骤

#### 1. 打开微分监视器界面。

- 在左侧设备树中双击“微分监视器”。
- 在菜单栏选择“视图 > 微分监视器”。
- 在工具栏单击 。
- 按【ALT+4】组合键。



界面参数信息说明如下表所示。

参数名称	参数说明
启动/停止	启动/停止微分监视器。
应用名	微分监视器所在的应用名称，用于区分多应用（Application）运行场景下的应用名称。在PLC支持多应用运行的场景下，微分监视器需要在工程中挂载多应用中的任务，界面只会显示当前活跃应用下的微分监视器视图，切换应用下的微分监视器会保存上一个应用下微分监视器的数据和加载最新微分监视器的数据，并更新监视状态。 <b>注意：</b> 多应用运行场景下，不能跨应用监视其它应用下的变量。
变量名	监视变量的名称。 监视精准场景：监视变量只在任务A中进行读写，监视任务选择任务A，可准确监视变量的状态；其他情况下监视可能存在遗漏计数。
条件	监视变量的条件，单击图标选择监视的条件。 •  上升沿 •  下降沿 默认监视上升沿。
任务	监视变量挂载的任务，可单击选择挂载的任务。 默认为MainTask。
计数	统计触发监视变量条件的次数。 • 在监视变量所在的任务周期内触发监视变量条件，即使在周期内发生了多次翻转，也只计数一次。 • 变量计数条件是与上一个周期最后一次记录的布尔类型值进行比较，所以在启动微分监视器监视时，程序第一次运行的第一个周期中不进行计数操作。
状态	监视变量的状态指示。 • 不触发监视变量条件：空白 • 触发监视变量条件：绿灰指示图标交替闪烁，触发一次闪烁6s左右时长。
注释	监视变量的注释内容。 当触发监测变量条件后，在预编译信息中获取该变量的注释信息，最多显示100个字符串。
添加/删除	添加/删除监视变量。

## 2. 添加监视变量。

- 单击“添加”，添加监视变量行。
- 在添加行“变量名”列双击文本框，输入监视变量名或者单击“...”选择监视变量。
- 在添加行“条件”列单击 或 设置监视条件。
- 在添加行“任务”列单击任务选择待挂载的任务。

微分监视器						
启动   停止   应用名: Application						
变量名	条件	任务	计数	状态	注释	
PLC_PRG.i		MainTask	0			
PLC_PRG.bool01		MainTask	0			
SD0_SD511.SD0		MainTask	0			

## 说明

- 如设置的监视变量无效，文本框的边框颜色将变为红色。
- 如需删除监视变量，选择待删除行，单击“删除”。
- 当工程处于在线或启动微分监视器时不能删除监视变量。

3. 检查是否正常登录运行PLC。
    - 如正常登录运行PLC，请执行下一步。
    - 如未正常登录运行PLC，请确保PLC正常登录运行再执行下一步。
  4. 单击“启动”，启动微分监视器监视。
- 

### 说明

停止微分监视器监视，包含以下情况：

- 单击“停止”，在打开的提示框中单击“是”。
  - 发生全部停止故障等级的控制器异常。
  - 下载用户程序。
  - 存储器全部清除。
  - InoProShop在线连接断开。
- 

### 后续登录时监视状态说明

- 登录PLC后退出登录，再次登录PLC时监视状态保持登录前的状态（监视已开启或未开启）。
- 登录PLC前监视已开启，监视变量状态由InoProShop微分监视器的监视变量设置（变量名、条件、任务和变量排序）和PLC运行的监视变量设置是否完全一致判断：
  - 如果完全一致，则微分监视器该监视变量维持监视状态。
  - 如果不一致，则微分监视器该监视变量的变量名文本框边框颜色变为红色，需要右键单击该监视变量行选择“微分监控复位”，重新对其进行监视（重新计数）。
- 完整下载后再登录PLC，微分监视器监视恢复初始状态，等待监视状态开启。
- 多人协同时：不同工程程序触发完整下载，再次登录PLC，微分监视器监视恢复初始状态，等待监视状态开启；相同工程程序直接下载，微分监视器监视状态以启动微分监视器的后台计数为准，在线修改以最新登录的后台工程为准，其它后台连接中断。

## 3.14 指令配置与调试

---



### 注意

- InoProShop V1.8.1.0及以上版本支持该功能。
  - 目前仅支持HC\_LoopControl库中的PID\_AT和PID\_ATC指令。
- 

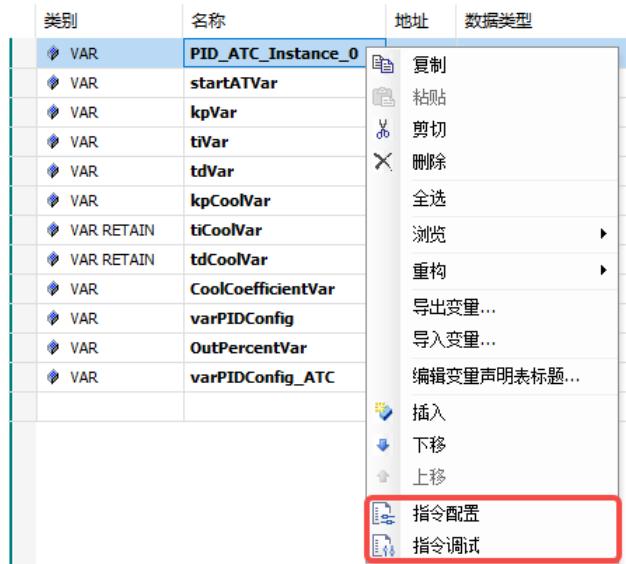
## 简介

对于某些复杂的指令（如PID相关指令）提供了指令配置和指令调试界面。用户使用此界面配置或调试这类指令时，无需参考指令手册即可知道每个参数的作用，并且轻松配置和调试指令。

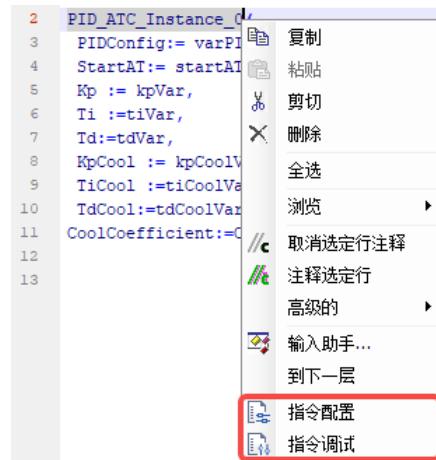
- 指令配置：提供了各参数名称、上下限范围、置灰属性、个性化展示等，使复杂的指令配置简单化。
- 指令调试：指令在线调试时，此界面显示调试该指令的一些常用参数，用户可以在此界面操作这些参数，并可直观地看到调试结果。

支持指令配置和指令调试的指令创建实例后，右键单击实例名称，在右键菜单中选择“指令配置”或“指令调试”，打开对应实例的指令配置和指令调试界面。

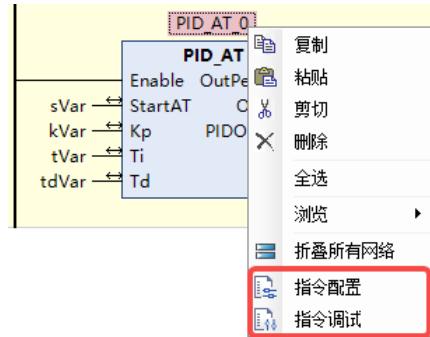
- 在变量表中打开



- 在ST语言程序中打开



- 在LD语言程序中打开



## 指令配置

指令配置界面包含功能视图和参数视图两种显示模式，在右上角单击“功能视图”或“参数视图”可进行切换。

- 功能视图

单击左侧各参数类别，右侧显示相应类别的参数。参数呈结构化显示，用户可以直观的配置各参数的值。



### ● 参数视图

单击左侧各参数类别，右侧显示相应类别的参数。参数呈表格化显示，用户可以看到更详细的信息，包含所有参数、FB中的名称、最小值、最大值和注释等。

功能视图   参数视图						
所有参数	功能视图中的名称	FB中的名称	初始值	最小值	最大值	注释
通用配置	控制对象	PIDConfig.GeneralConfig.PhysicalQuantity	流量			控制的物理对象
	单位	PIDConfig.GeneralConfig.PhysicalUnit	l/s			物理对象的单位
控制设置	控制模式	PIDConfig.GeneralConfig.ControlMode	单极性加热			PID控制模式，0：反作用(单极性加热)；1：正作用(单极性冷却)
联锁	冷却系数	CoolCoefficient	0.0	≥0	≤100	冷却系数
级联	手动模式	PIDConfig.GeneralConfig.ManualEnable	自动			手动模式使能，TRUE：手动模式使能。
延时控制	手动值	PIDConfig.GeneralConfig.ManualValue	0.0	% ≥0.0	% ≤100.0	手动模式输出百分比。单位%。
	联锁开关	InLockSW	FALSE			联锁开关
	联锁安全值	InLockVal	12	% ≥0.0	% ≤100.0	联锁安全值
输入配置	控制器为主调	PIDConfig.GeneralConfig.IsMaster	FALSE			是否级主调：0：否 1：是
输入基本设置	控制器为副调	PIDConfig.GeneralConfig.IsSlave	FALSE			是否级副调：0：否 1：是
模拟量输入标定	延时启动	PIDConfig.GeneralConfig.DelayStartUp	10	ms ≥0	ms ≤10000...	延时启动时间，单位毫秒。
设定值配置	延时关闭	PIDConfig.GeneralConfig.Delay ShutDown	11	ms ≥0	ms ≤10000...	延时关闭时间，单位毫秒。
输出配置						
输出基本设置						
输出错误响应						
PWM输出						
模拟量输出标定						
高级配置						
PID参数						
积分优化						
其他优化						
调节参数						
其他参数						

### 离线编辑

- 功能视图模式和参数视图模式都可以编辑参数值。
- 参数编辑支持撤销和恢复。
- 参数输入范围超限时会报错误提示（该错误仅在配置界面提示，编译不会报错）。

输出上限值:	<input type="text" value="100"/> %
输出下限值:	<input type="text" value="150"/> %
请输入正确范围内的值: 输出下限值 < 当前值 ≤ 100。	

- 若参数在编程区绑定了其他变量，则编辑参数值时，编辑的是绑定的变量。

```

2 PID_ATC_Instance_0(
3   PIDConfig:= varPIDConfig_ATC,
4   StartATC:= startATCVar,
5   Kp := kpVar,
6   Ti :=tiVar,
7   Td:=tdVar,
8   KpCool := kpCoolVar,
9   TiCool :=tiCoolVar,
10  TdCool:=tdCoolVar,
11  CoolCoefficient:=CoolCoefficientVar);
12
13 PID_ATC_Instance_0.PIDConfig := varPIDConfig_ATC;

```

可以识别  
不能识别



注意

只可以识别功能块内部参数引脚绑定的变量，功能块外部引脚绑定的不能识别。

## 在线监视

### ● 功能视图

控制设置

控制对象:	流量	单位:	/s
控制模式:	单极性加热	冷却系数:	0
手动模式:	自动 [手动]	手动值:	0 %

联锁

联锁开关:	FALSE	联锁安全值:	12 %
-------	-------	--------	------

级联

控制器为主调:	FALSE	控制器为副调:	FALSE
---------	-------	---------	-------

延时控制

延时启动:	10 ms
延时关闭:	11 ms

输入框显示实时监视值（十进制值），双击未置灰的输入框，打开准备值输入对话框。



输入准备值，单击“确定”，输入框内同时显示监视值和准备值。

手动模式:

### ● 参数视图

功能视图中的名称	FB中的名称	初始值	监视值	准备值	最小值	最大值	注释
控制对象	PIDConfig.GeneralConfig.PhysicalQuantity	流量	流量				控制的物理对象
单位	PIDConfig.GeneralConfig.PhysicalUnit	/s	/s				物理对象的单位
控制模式	PIDConfig.GeneralConfig.ControlMode	单极性加热	单极性加热				PID控制模式，0:
冷却系数	CoolCoefficient	0.0	0		≥0	≤100	冷却系数
手动模式	PIDConfig.GeneralConfig.ManualEnable	自动	自动	手动			手动模式使能，T
手动值	PIDConfig.GeneralConfig.ManualValue	0.0	% 0	%	≥0	≤100.0	手动模式输出百分比
联锁开关	InlockSW	FALSE	FALSE				联锁开关
联锁安全值	InlockVal	12	% 12	%	≥0	≤100.0	% 联锁安全值
控制器为主调	PIDConfig.GeneralConfig.IsMaster	FALSE	FALSE				是否串级主调:0:
控制器为副调	PIDConfig.GeneralConfig.IsSlave	FALSE	FALSE				是否串级副调:0:
延时启动	PIDConfig.GeneralConfig.DelayStartUp	10	ms 10	ms	ms ≥0	ms ≤10000... ms	延时启动时间, ms
延时关闭	PIDConfig.GeneralConfig.Delay ShutDown	11	ms 11	ms	ms ≥0	ms ≤10000... ms	延时关闭时间, ms

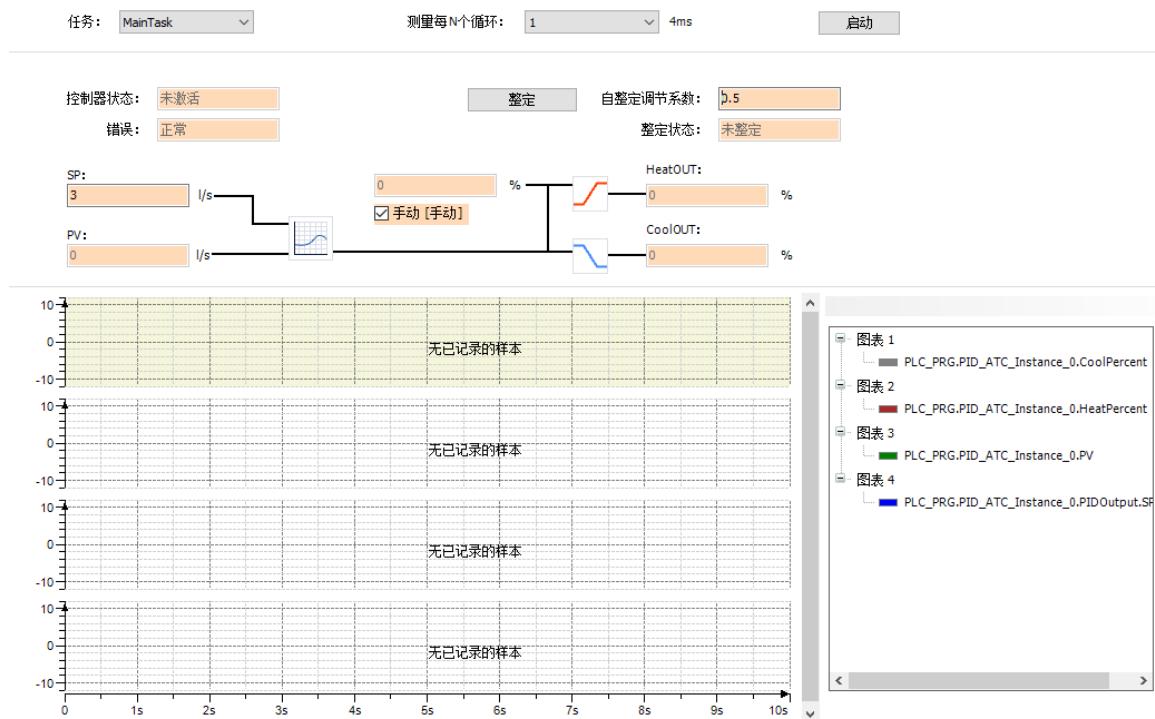
在线时，参数视图增加“监视值”和“准备值”列，可以显示实时监视值（十进制值）和输入准备值。

## 指令调试

指令调试界面显示指令调试时一些常用参数，用户通过操作这些参数进行调试，并直观地看到调试结果。

每个指令调试参数不同，对应的调试界面也不同。以“PID\_ATC”指令为例，如下图所示，参数说明如下表所示。

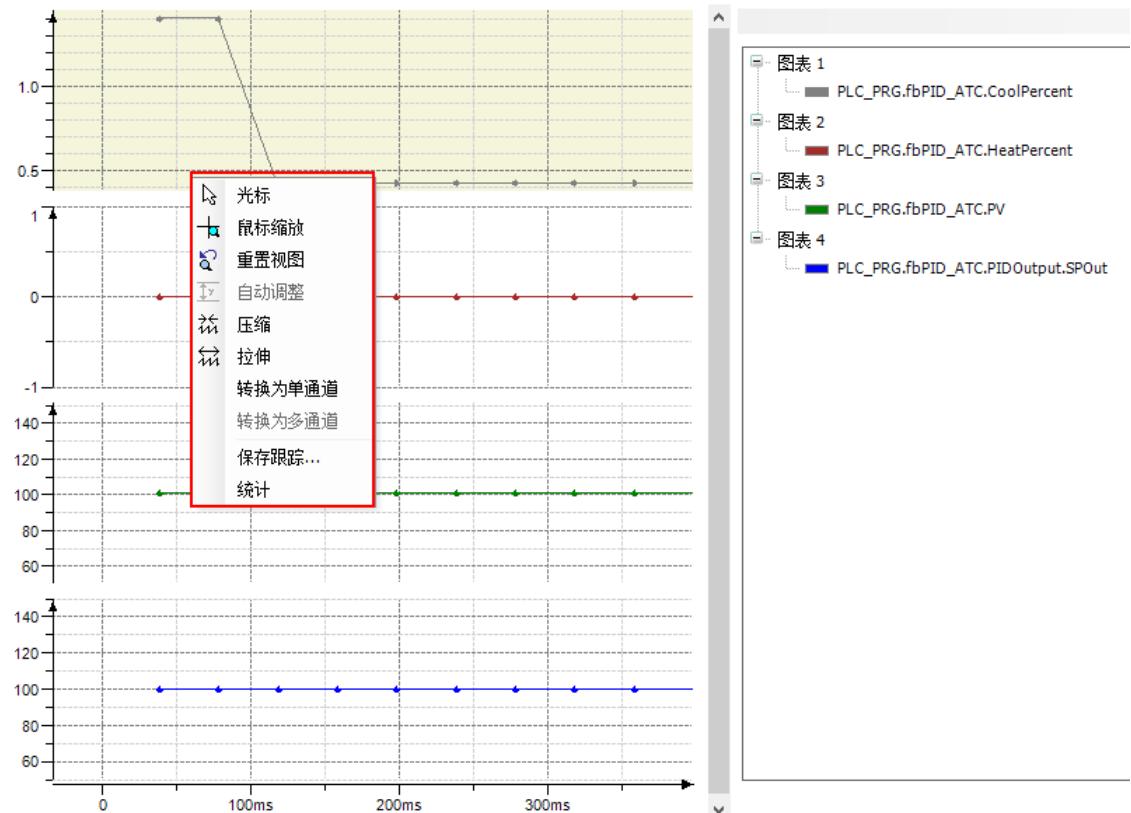
## 基本功能



参数名称	参数说明
任务	任务可选择用户创建的任意任务和EtherCAT任务（与Trace示波器保持一致）。
测量每N个循环	采集的周期（任务周期*N）。
启动/停止	<ul style="list-style-type: none"> <li>单击“启动”，示波器进行采集。</li> <li>每次单击“启动”时，将清空上次的示波器数据并重新开始。</li> <li>示波器启动状态下，转离线或关闭调试界面，都会先自动停止示波器。</li> <li>单击“停止”，示波器停止采集。</li> </ul>
控制器状态	对应变量State，仅显示。
错误	对应变量ErrorBits，仅显示。 如果有参数错误，监视值框后面显示  图标，鼠标悬停显示详细错误号和错误描述。
整定	对应变量StartAT，按下写准备值。 调用导航栏“写入值”命令，触发整定值为StartAT=TRUE，整定按钮置灰。 整定成功后会反写为FALSE，按钮可再次触发。 假如在整定过程中需要停止整定，需要在指令配置“参数视图 > 其他参数”界面中“StartAT”置为“整定完成”，然后调用导航栏“写入值”命令。
自整定调节系数	对应编辑变量PIDConfig.GeneralConfig.ATCoefficient，在线可写可强制。
整定状态	对应变量PIDOutput.ATState，仅显示。
SP	当前设定值，仅显示。
PV	当前实际PV值，仅显示。
OUT	对应变量OutPercent，仅显示。
手动	对应变量PIDConfig.GeneralConfig.ManualEnable，勾选后写准备值。 调用导航栏“写入值”命令，写入值TRUE，触发手动模式。 <ul style="list-style-type: none"> <li>手动模式下，手动值文本框可双击写准备值，操作变量为PIDConfig.GeneralConfig.ManualValue。</li> <li>自动模式下，手动值本框不可写准备值。</li> </ul>

设置“PID\_ATC”指令相关参数，参数设置完成后单击“启动”，启动示波器，观察调试结果。

在示波器上单击右键，打开右键菜单，可进行相应的操作，如下图所示。



## 4 网络配置

### 4.1 设备组态

#### 4.1.1 设备组态

设备组态是用户进行PLC编程的第一步，具体包括“网络组态”和“硬件组态”两个功能，用户可以通过这两个功能来对设备进行布局。

- 网络组态  
从总线型的网络拓扑视角设计，是设备组态的入口。
- 硬件组态  
添加中型PLC扩展IO模块。

#### 4.1.2 网络组态

新建一个InoProShop工程后，在左侧设备树中双击“Network Configuration”的节点，如下图所示。

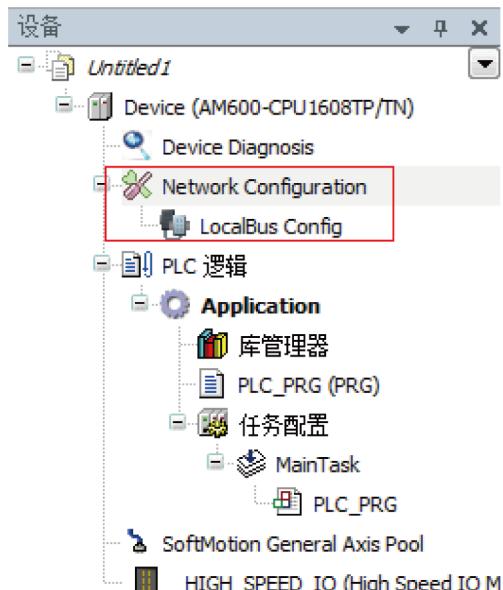


图4-1 网络组态节点

双击节点后，打开网络组态“Network Configuration”的界面以及“网络设备列表”（[第87页“设置PLC作为主站或从站设备”](#)），网络组态界面显示用户工程当前所使用的PLC设备，而“网络设备列表”显示当前PLC支持的所有设备。

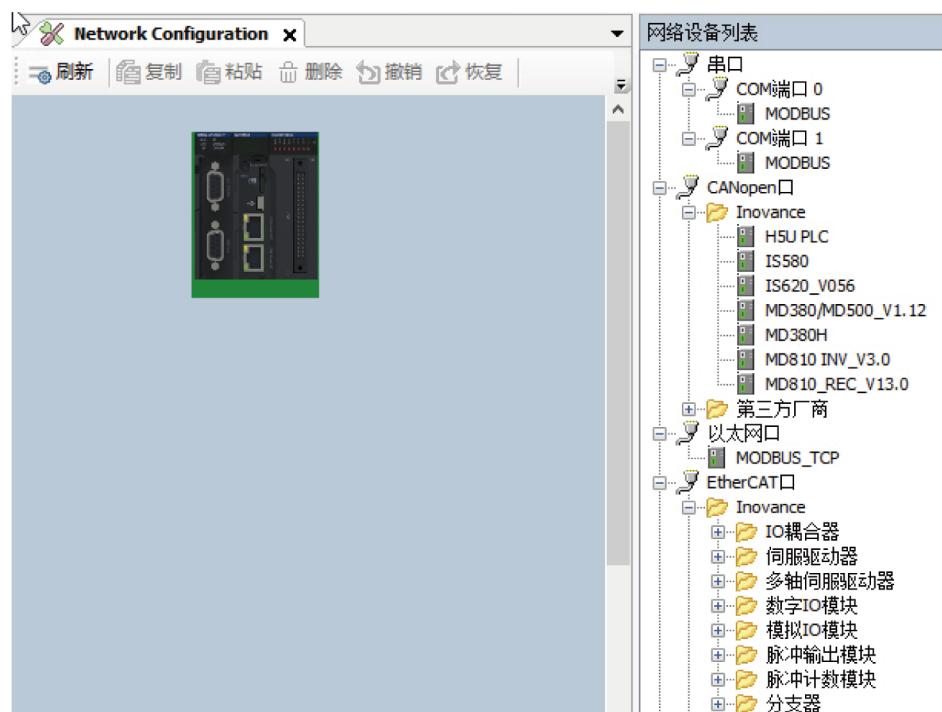
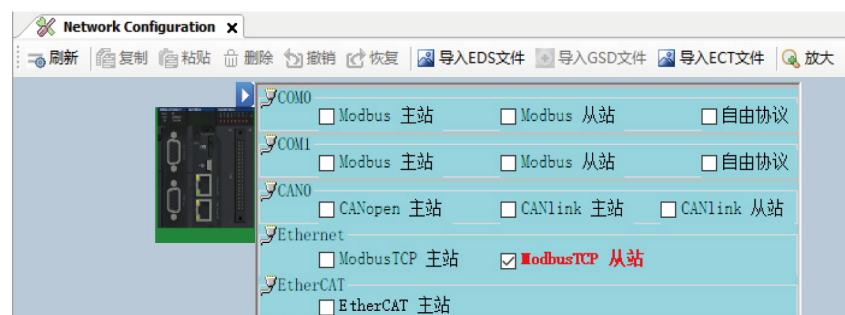


图4-2 网络组态“Network Configuration”

## 设置PLC作为主站或从站设备

单击网络组态中的PLC设备，会显示PLC支持的主/从站使能窗口，如下图所示，根据应用需要选中窗口中的复选框按钮即可使能CPU所支持的主/从站功能。



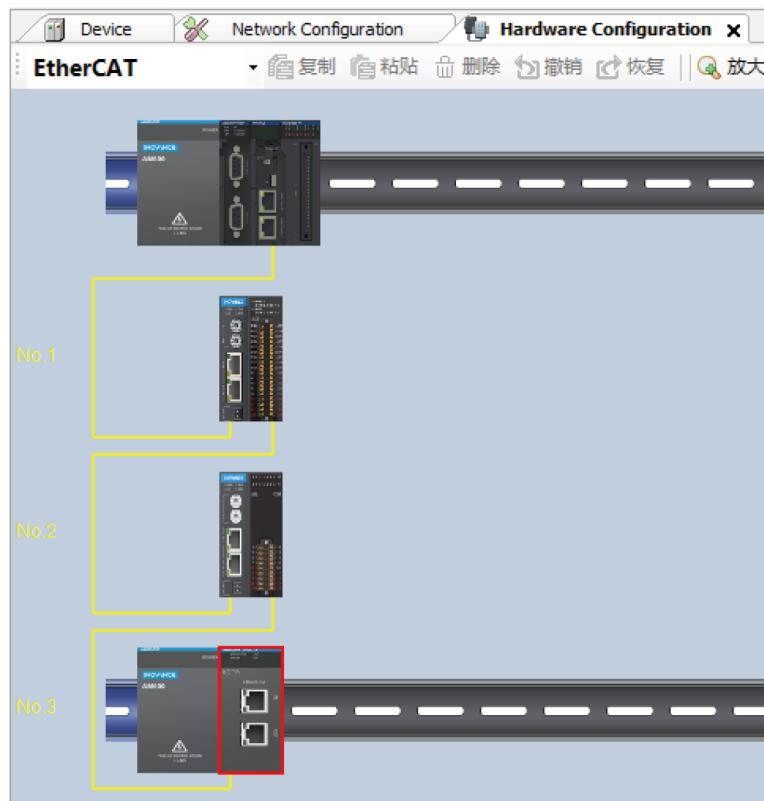
除CANlink主站外，使能CPU的其他特定主站功能时，都会显示总线型的拓扑界面。例如下图所示为使能EtherCAT主站：



- 添加从站设备

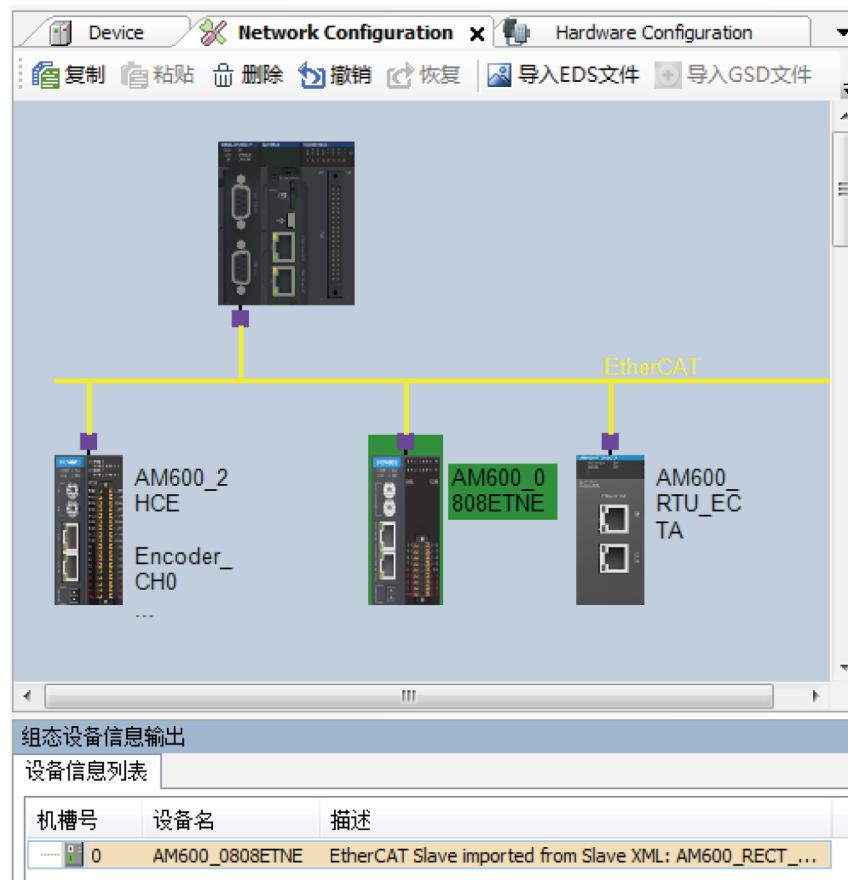
使能CPU中的某个特定的主站后，即可添加其相对应的总线下的从站设备，添加从站设备有三种方式(以EtherCAT总线为例)：

1. 先使能EtherCAT主站功能，然后从网络设备列表中的“EtherCAT口”节点下选中一个从站设备节点，按住鼠标左键不松拖拽到网络组态界面中。
2. 先使能EtherCAT主站功能，然后从[第87页“设置PLC作为主站或从站设备”](#)所示的网络设备列表中的“EtherCAT口”节点下双击一个从站设备节点即可。
3. 直接在网络设备列表中的“EtherCAT口”节点下双击一个从站设备节点添加，此种方式会默认先使能CPU内部的特定主站功能。
4. 如果添加EtherCAT分支器设备，请参见[第197页“添加分支器及其下从站设备”](#)。添加完从站，如果想配置从站后的IO模块(针对AM600系列通讯从站)，则双击设备即可进入硬件组态配置“Hardware Configuration”界面进行配置。



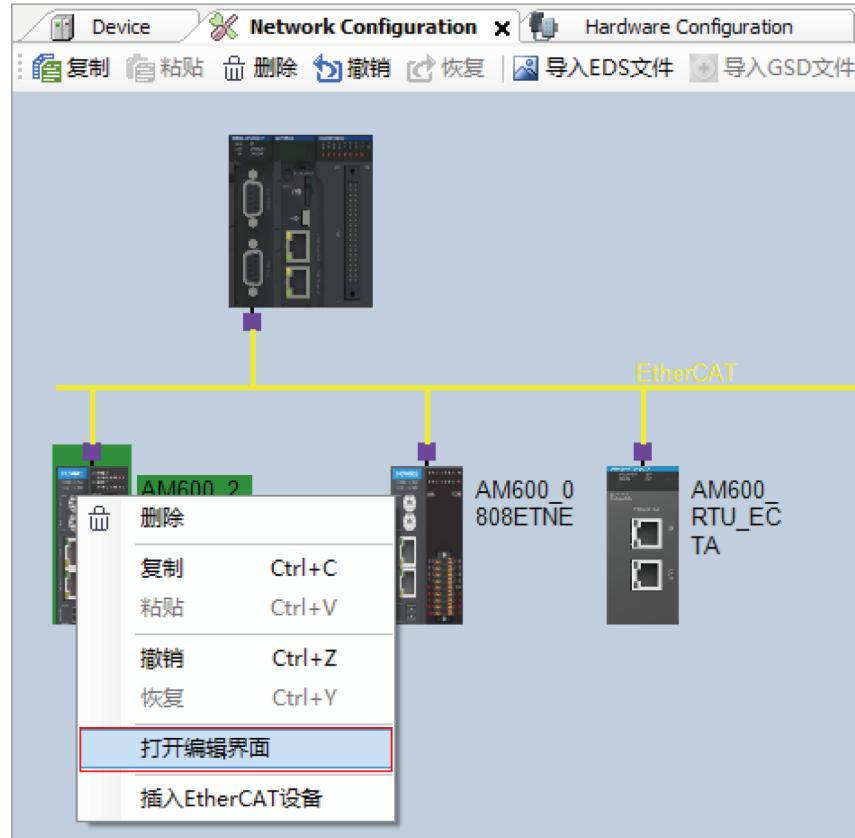
- 查看设备基本信息

选中网络组态界面中的设备后，可在“组态设备信息输出” - “设备信息列表”中找到对应的设备基本信息。



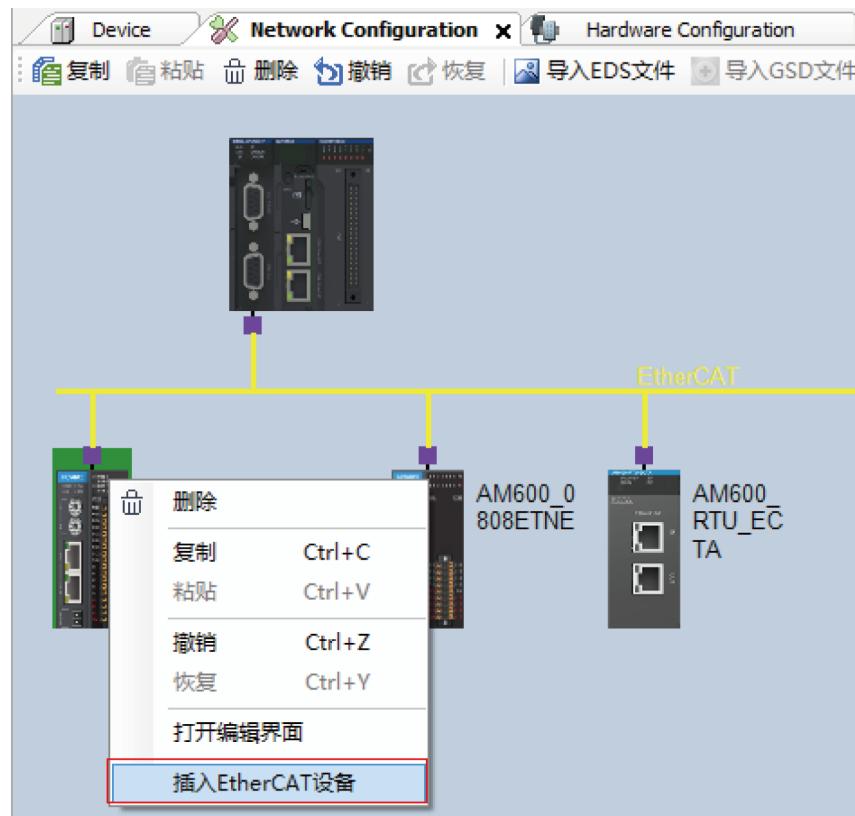
- 打开设备配置界面

右键单击网络组态中的EtherCAT从站，通过“打开编辑界面”选项进入设备的配置界面，如下图所示。



- 插入EtherCAT从站

右键单击网络组态中的EtherCAT从站，通过“插入EtherCAT设备”选项插入一个EtherCAT从站设备，如下图所示。



- 组态设备可进行复制、删除、添加等操作，详情请参见组态设备的常用操作。
- 编译组态

当网络组态中有两个或多个Modbus从站设备站号相同，或ModbusTCP从站设备的IP地址相同，编译工程时会显示在编译信息输出框中，详情请参见组态编译错误定位。

## 设备信息列表

设备信息列表通过菜单项“视图”->“组态设备信息视图”打开，用于显示组态设备的基本信息，包括机槽号、设备名称、描述。如果默认最小化在软件底部(

消息 - 0个错误, 0警告, 0条消息 组态设备信息输出)，需要手动点击打开列表。

组态设备信息输出		
设备信息列表		
机槽号	设备名	描述
-1	AM600-PS2	AM600电源模块，输出电流为2A
0	Device	2路RS485, 1路CAN, 标准以太网, EtherCAT总线, 总线运动控制
1	AM600_0016ER	16点DO模块；继电器晶体管输出
2	AM600_4DA	4通道DA模块；支持电压/电流输出

图4-3 设备信息列表

- 机槽号

对应硬件组态中的设备插槽，无论是主机架上的模块或是通讯从站后的模块，其槽号都是从1开始。-1号机槽对应AM600电源模块，0号机槽对应CPU模块。

- 设备名

与软件左侧设备视图中所示的设备名称一致

- 描述

设备的基本描述，包括设备基本工作指标和功能。

单击设备列表中的某行可以定位组态界面中的对应设备，双击某行也可打开对应的设备组态界面。

## 组态设备的常用操作

组态设备基本操作包括设备的复制、粘贴、撤销、恢复、删除、导入EDS、GSD、ECT文件、放大、缩小功能。



### 说明

- 设备复制、粘贴、删除、撤销、恢复操作在硬件组态界面只针对IO模块；在网络组态界面只针对从站。
- 如果对网络组态中的从站进行复制、粘贴、删除，则后面的模块也会被相应操作。

- 导入EDS: 网络设备列表中默认带部分CANopen设备和EtherNet/IP设备，如果要加入其他CANopen设备或EtherNet/IP设备，需要导入其对应的标准EDS文件。导入成功后，该设备会添加到网络设备列表中，如果是汇川设备，会显示在Inovance节点下，否则会显示在“第三方厂商”节点下。
- 导入GSD: 网络设备列表中默认带部分DP设备，如果要加入其他DP设备，需要导入其对应的标准GSD文件。导入成功后，设备会添加到网络设备列表中的“DP口”节点下，如果是汇川设备，会显示在Inovance节点下，否则会显示在“第三方厂商”节点下。
- 导入ECT: 网络设备列表中默认带部分EtherCAT设备，如果要加入其他EtherCAT设备，需要导入其对应的标准EtherCAT xml(\*.xml)文件。导入成功后，设备会添加到网络设备列表中的“EtherCAT口”节点下，如果是汇川设备，会显示在Inovance节点下，否则会显示在“第三方厂商”节点下。

### 4.1.3 硬件组态

硬件组态引入了实际设备组态中的机架和机槽概念，用来模拟现场设备的模块化配置。硬件组态主要面向中型PLC系列产品的IO模块。

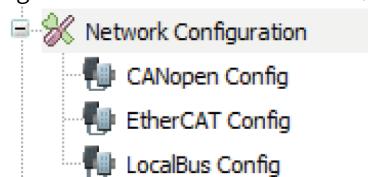
从组态流程来讲，如果是添加远程IO模块，应该在网络组态中完成通讯模块组态之后，再在硬件组态中进行IO模块配置；如果只是配置本地IO模块，则直接打开硬件组态操作即可。硬件组态同时支持多总线IO配置，具体取决于所选用的CPU型号。

#### 进入硬件组态界面

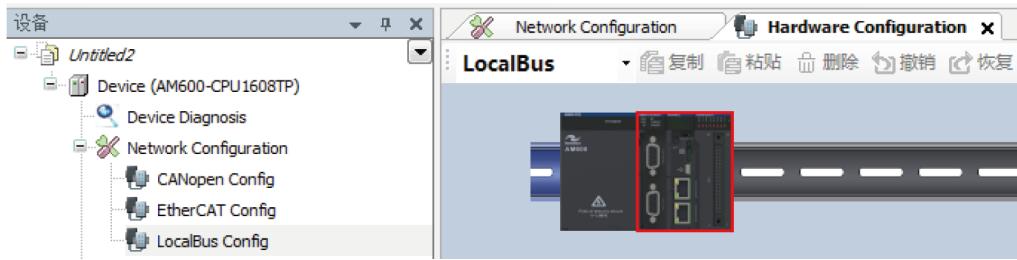
除Modbus设备和ModbusTCP设备外，其他总线类型设备都有对应的硬件组态界面。

进入硬件组态界面分为以下两种方式：

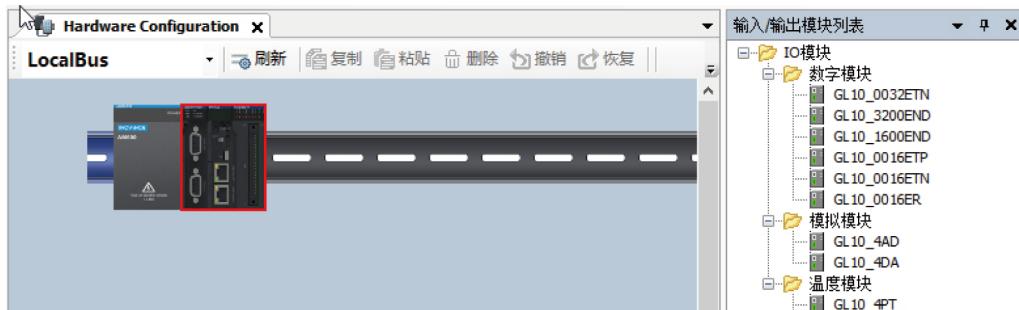
1. 在“Network Configuration”界面双击设备图标。
2. 双击软件左侧设备树“Network Configuration”节点下的某个总线节点，如下图所示。



默认有“LocalBus Config”即本地总线配置节点，双击即可进入本地模块的配置。



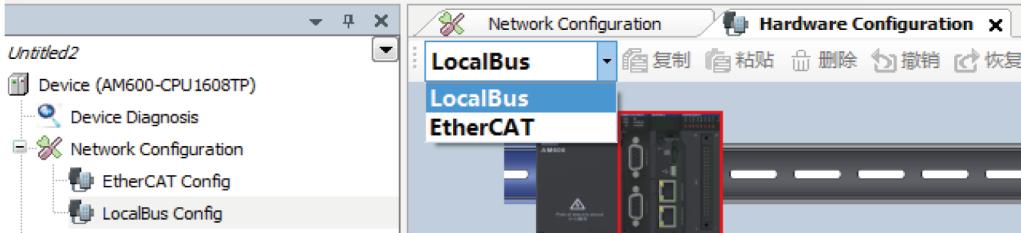
同时软件右侧会显示"输入/输出模块列表"。



## 总线切换

有两种方式可实现硬件组态总线之间的切换:

- 双击软件左侧设备树"Network Configuration"节点下的某个总线节点，进入相应的配置界面。
- 在当前硬件组态界面选中其他总线类型，如下图所示。



## 添加模块

有3种方式用来添加IO模块:

1. 双击打开机架上的一个空机槽，在弹出的模块列表中双击特定模块即可添加。
2. 在右侧视图的模块列表中，选中一个设备节点，按住鼠标左键将其拖放到空机槽上。
3. 选中一个机架(点击图中的蓝色部位)或设备，然后双击视图右侧模块列表中的某个设备，即可将设备按顺序自动添加到机架上的空槽中。而如果选中某个空槽，则会添加到该空槽中。



## 拖拽模块

通过选中一个模块按住鼠标左键进行模块的拖拽操作，拖动到目标槽位置松开鼠标即可。拖拽操作包括两个模块之间的位置交换，或将一个模块拖动到空槽中，但不支持主机架和扩展机架之间模块拖拽操作。

### 4.1.4 设备树操作

添加总线设备后，用户可以在设备树选中某一设备，通过右键菜单或快捷键对设备进行复制、粘贴、删除、剪切和拖动等相关操作。菜单项如下图所示：



各功能操作符合基本标准操作。

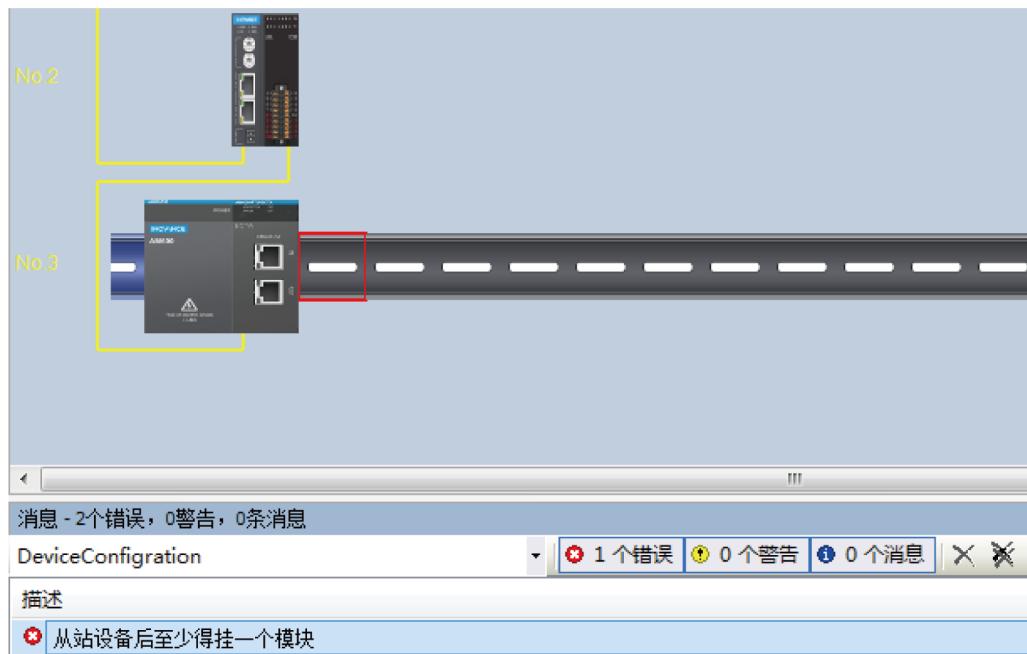
#### 说明

复制和剪贴功能仅适用于本地主站设备、本地从站设备以及单独轴设备。

### 4.1.5 组态编译错误定位

组态设备制定了一些配置规则以及错误检测机制：比如网络组态中两个MODBUS设备的站号相同，或是TCP设备的IP地址相同；硬件组态设备中扩展机架上的从站设备后没有接IO模块等都会导致组态编译报错。

在编译工程时，如果出现组态错误，InoProShop消息输出框中会显示，双击其中的错误列表可以自动定位到对应的组态界面，并会闪烁3次红色的矩形框。如图所示：



## 4.2 CPU 配置

### 4.2.1 概述

CPU模块即中型PLC主模块，CPU配置基于PLC硬件控制系统的要求，来完成对PLC及其控制系统的配置。中型PLC支持EtherCAT总线、Profibus DP、Modbus RTU、CAN总线、ModbusTCP，另外还提供高速I/O功能。因此，若要完成整个CPU的配置，则需要根据PLC硬件网络配置相应的总线参数。

以AM600 CPU模块为例，其自带高速I/O，后面可扩展本地IO模块。另外，在CPU模块配置界面还可以配置CPU系统参数、CPU固件升级等功能。

### 4.2.2 CPU配置的一般过程

1. 设计CPU整个硬件网络结构。
2. 根据设计的硬件架构在网络组态中激活相应总线，并添加总线对应的从站。目前CPU支持EtherCAT总线、DP总线、CANopen、CANlink、ModbusRTU、ModbusTCP。
3. 如果是EtherCAT AM600从站、CANopen AM600从站或者DP AM600从站，需要在硬件组态中添加I/O模块。
4. 配置总线对应的主站、从站和模块配置参数。

AM600 CPU模块默认带高速I/O功能，每个CPU最多可扩展16个本地I/O模块，另外还需按具体需求对CPU本身系统参数、PLC I/O刷新设置、PLC总线任务、PLC用户管理、日志、升级、任务配置等进行设置。

如需了解总线及对应从站的参数配置，请参见各总线对应的章节。PLC I/O刷新设置、PLC总线任务、PLC用户管理、日志、任务配置等是Codesys自带功能，详见Codesys软件帮助链接。本文中的CPU配置主要讲述中型PLC的功能：CPU参数配置、I/O模块配置和高速I/O配置。

### 4.2.3 CPU 参数配置

#### 系统设置

系统设置用于配置CPU故障停机、掉电保存位置、网络地址和系统时间，如下图所示。



图4-4 系统设置对话框

#### 错误时的运行模式

- 组态错误时停机：出现组态不一致时CPU是否停止运行，例如，CPU后挂接的组态IO和实际硬件连接IO不匹配是否停机。
- 系统错误时停机：出现系统错误时CPU是否停止运行，如中断错误、堆栈溢出等。
- Flash错误时停机：出现Flash错误时CPU是否停止运行，暂不支持。
- SD卡错误时停机：出现SD卡错误时CPU是否停止运行，如SD卡内存满，SD卡丢失等，暂时不支持。

#### 掉电保存

- 保存位置：设置掉电保存位置：本地存储器和SD扩展卡。

#### 说明

当选择SD扩展卡时，首先要确保SD扩展卡是否存在，否则掉电保存数据将丢失。

#### 网络设置

- 网口：PLC使用的EtherNET通讯网口名，AM600和AM400系列只有一个EtherNET网口，AC800系列有两个EtherNET通讯网口。不同PLC的网络名称不同，用户可以对不同网口设置不同的网络信息。
- 使用下面的IP：PLC的IP地址既可以通过手动修改，也可以自动获取，此配置用于手动修改PLC网络信息。
- 自动获取IP：PLC的IP地址由路由器或者交换机自动分配。注意：此功能当前只有AC700、AC800支持。
- IP地址：PLC的IP地址。
- 子网掩码：PLC的子网掩码。
- 网关：设置PLC的网关。注意：此功能当前只有AC700、AC800支持。
- 读取：读取PLC的IP地址和子网掩码，显示在IP地址和子网掩码编辑框中。
- 写入：将编辑框中的IP地址和子网掩码写入PLC。如果在登录状态并且使用网络连接，系统将会退出登录，此时用户需重连设备；如果使用USB连接，则不需要重连设备。

## 说明

- 中型PLC的两个网口禁止设置相同网段IP，否则影响连接。
- 读、写PLC的IP地址时，需要先在通信设置选项卡选择要读、写的PLC设备，另外，写入的IP地址和子网掩码要符合IP地址和子网掩码规范。
- 识别设备：识别要连接的PLC。当在通信设置选项卡扫描PLC时，有可能扫描到多台PLC设备，当选中其中一台PLC后，点击此按钮，此时PLC设备的显示面板上的两位数码管，将交替显示字符“0”，如图所示：

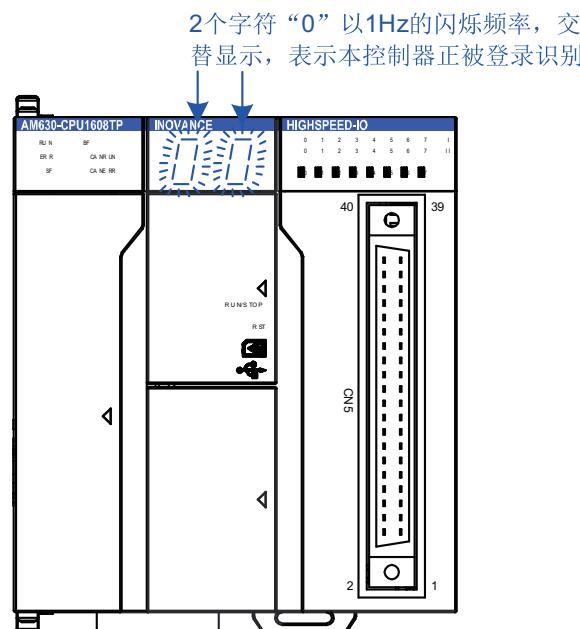


图4-5 PLC数码管处于识别状态

同时后台软件InoProShop显示下图中的正在识别对话框，当用户关闭此对话框时表示识别完成，数码管恢复正常。



图4-6 正在识别对话框

### RTC配置

- PLC时间：显示读取的当前PLC时间。
- 读取：读取PLC时间。
- 写入：写入PLC当前设置的日期和时间，当前设置的日期和时间在左侧的日期和时间编辑框内。
- 同步到本地日期/时间：写入当前电脑对应的时间到PLC。

### 说明

在写入PLC时间或者同步本地日期/时间时，有可能会对PLC造成影响，如影响总线同步功能，因此在写入PLC时间前，请先确认PLC是否处于停止状态，建议在写入时间后热复位PLC。

### 时区

- PLC时区：读取的PLC时区，如北京时区（东八区）为UTC+8。
- 读取：读取PLC对应的时区。
- 写入：写入PLC当前选择的时区。

### 升级

升级页面用于PLC的固件升级，如[第98页“4-7 升级对话框”](#)所示。PLC固件升级包提供要升级的软件数据，其中可能包含UBOOT、Device Tree、内核、系统程序中的一种或者几种，一般升级包只包含系统程序。

### 说明

此处的升级功能不再支持，使用InoProShop工具进行升级。

升级功能不再支持，请使用 InoProShop 工具进行升级（菜单中【工具】-【InoProShop 工具】）！

PLC信息

PLC型号:  固件版本:  版本详细信息

固件升级

升级前请扫描并选择设备，升级过程中请不要断电

固件升级包:  选择固件升级包

兼容设备:  固件版本:  固件详细信息

升级进度:

图4-7 升级对话框

### PLC信息

- PLC型号：当前PLC型号，如AM600、AM610。
- 固件版本：当前PLC的固件版本，如1.2.3.0。
- 版本详细信息：显示当前PLC详细的版本信息，详细版本信息可能包含UBOOT、Device Tree、内核、系统程序版本中一种或者几种，如图3-8所示。如果PLC没有固件升级过，可能不包含版本信息。
- 获取PLC信息：获取PLC型号信息和固件版本信息，如果PLC没有进行固件升级，可能获取不到PLC信息。

### 固件升级

- 固件升级包：设置固件升级包，固件升级包以.upgrade为扩展名。
- 兼容设备：显示固件升级包的兼容设备，只有和当前PLC型号兼容的设备才能升级。
- 固件版本：显示固件升级包的固件版本。
- 固件详细信息：获取固件升级包的详细信息，详细信息如下图。



图4-8 固件详细信息对话框

- 升级：开始固件升级。升级时会检查设备类型和升级固件文件版本，如果升级的固件版本比PLC的固件版本高则直接升级，版本相同不需要升级，如果升级固件版本旧于PLC设备固件版本需要确认后才能升级。

## 说明

- 升级之前必需先在“通信设置”窗口扫描设备并选择要升级的PLC；
- 升级过程中不能断电，否则可能造成系统不可恢复的故障；
- 升级大概需要2分钟，完成后会自动重启设备；
- 重启后（升级完成后）数码管会显示00或者动态变化的数字；
- 另外升级完成后，PLC设备名称可能会发生变化，需要重新扫描设备。

升级完成后，获取PLC信息及版本详细信息，核对PLC信息及版本详细信息是否与固件版本及详细信息一致。如下图所示：

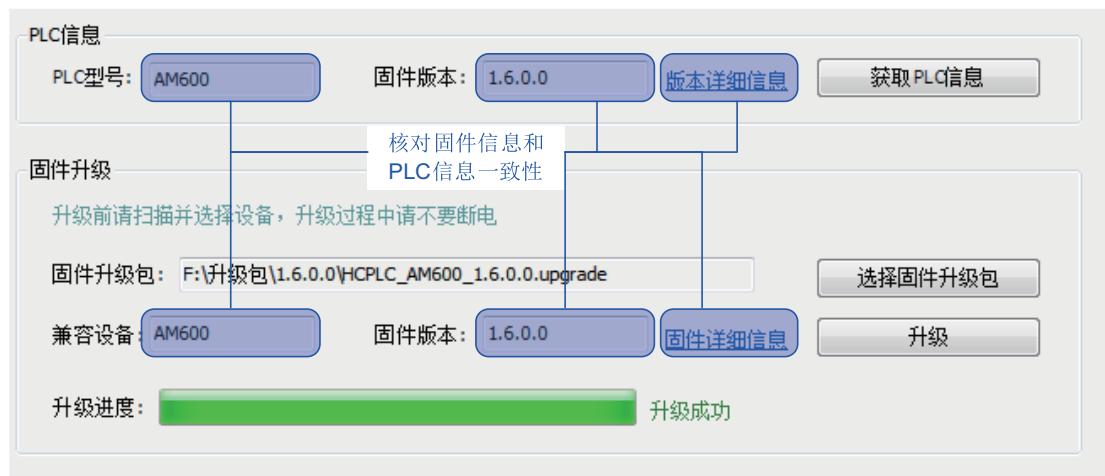


图4-9 核对升级是否成功

## 信息

显示PLC设备基本信息：名称、供应商、组、类型、序号、版本，模块序号、描述、图像，如下图所示。另外，登陆后信息页面会显示CPU的单板软件版本和逻辑软件版本信息。单板软件版本为CPU系统程序版本，逻辑软件版本为CPU内FPGA软件版本。



图4-10 CPU信息页面

## 4.2.4 IO 模块配置

### 概述

IO模块分为GL10系列和GL20(S)系列。

模块系列	模块类别	
GL10系列	数字量输入 (DI)	
	数字量输出 (DO)	继电器输出 (ER型)
		NPN输出 (ETN型)
		PNP输出 (ETP型)
	模拟量输入 (AD)	
	模拟量输出 (DA)	
	温度检测模块	4TC (4通道温度检测模块, 支持热电偶)
		8TC (8通道温度检测模块, 支持热电偶)
		4PT (4通道温度检测模块, 支持热电阻)
GL20(S)系列	数字量输入 (DI)	
	数字量输出 (DO)	继电器输出 (ER型)
		NPN输出 (ETN型)
		PNP输出 (ETP型)
	数字量输入输出 (DI DO)	
	模拟量输入 (AD)	
	模拟量输出 (DA)	
	温度检测模块	4PT (4通道温度检测模块, 支持热电阻)
		4TC (4通道温度检测模块, 支持热电偶)
	通信模块	2CAN (2路CAN通信模块)
		2S485 (2路RS485扩展模块)
		2SCOM (2路串口模块)
		1DNM (1路DeviceNet主站模块)
	工艺模块	2SSI (2路SSI通信)

### GL10系列数字量输入模块

数字量输入模块无模块参数配置，只有I/O映射、状态和信息页面，一般情况下只需要在I/O映射界面映射I/O变量以获取数字量输入值。在此以16点数字量输入模块为例。

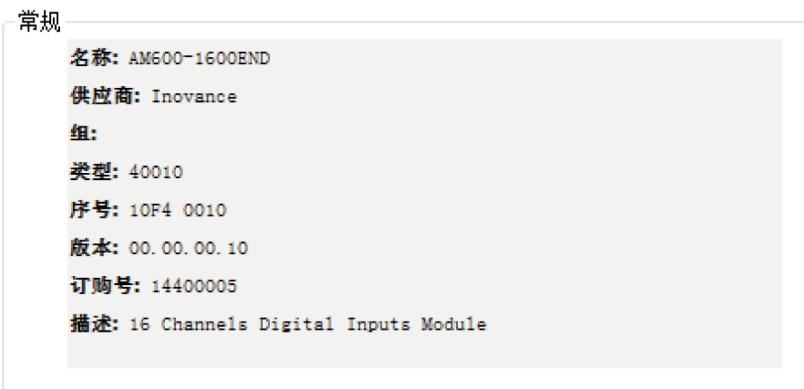
#### 1. DI16 I/O映射

DI16为16位数字输入模块，如下图所示，可通过在I/O映射界面对每位或者每8位映射到一个变量，获取输入值，具体请参照I/O映射链接。



## 2. 信息

显示DI16模块设备基本信息：名称、供应商、组、类型、序号、版本，订购号、描述、图像，如下图所示。另外，登录后信息页面会显示DI模块逻辑软件版本，逻辑软件版本为DI模块内FPGA软件版本。

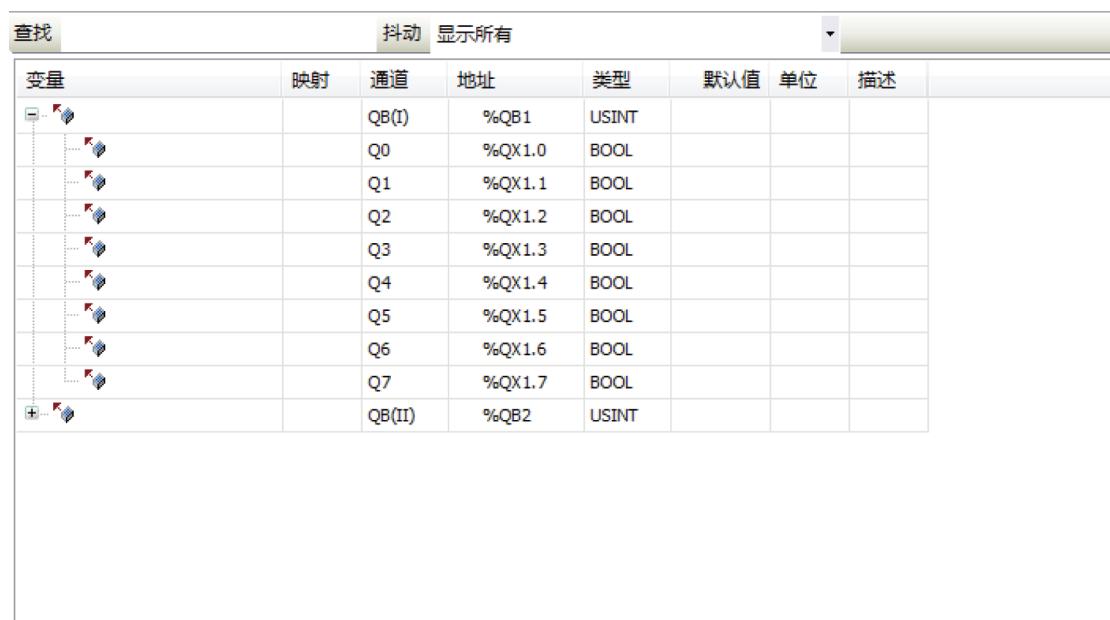


## GL10系列数字量输出模块

数字量输出模块，无模块参数配置，只有I/O映射、状态和信息页面，一般情况下只需要在I/O映射界面映射I/O变量，然后把映射变量值输出到数字量输出模块。数字量输出包含16点和32点，两者的I/O映射界面相似。在此以16点数字量输出模块为例。

### 1. DO16 I/O映射

DO16为16位输出模块，如下图所示，可通过在I/O映射界面对每位或者每8位映射到一个变量，以输出变量值，具体请参照I/O映射链接。



The screenshot shows a table for mapping variables. The columns are labeled: 变量 (Variable), 映射 (Mapping), 通道 (Channel), 地址 (Address), 类型 (Type), 默认值 (Default Value), 单位 (Unit), and 描述 (Description). The data in the table is as follows:

变量	映射	通道	地址	类型	默认值	单位	描述
		QB(I)	%QB1	USINT			
		Q0	%QX1.0	BOOL			
		Q1	%QX1.1	BOOL			
		Q2	%QX1.2	BOOL			
		Q3	%QX1.3	BOOL			
		Q4	%QX1.4	BOOL			
		Q5	%QX1.5	BOOL			
		Q6	%QX1.6	BOOL			
		Q7	%QX1.7	BOOL			
		QB(II)	%QB2	USINT			

Below the table are several buttons: '重置映射' (Reset Mapping), '总是更新变量' (Always update variable), and '启用1 (如果未在任何任务中使用, 适用总线周期)' (Enable 1 (if not used in any task, apply bus cycle)). At the bottom are two icons: a star with a plus sign and a gear, both with explanatory text: '=创建新变量' (Create new variable) and '=对现有变量进行映射' (Map existing variable).

## 2. 信息

显示DO16模块设备基本信息：名称、供应商、组、类型、序号、版本，模块号、描述、订购号、图像，如下图所示。另外，登录后信息页面会显示DO模块逻辑软件版本，逻辑软件版本为DO模块内FPGA软件版本。



## GL10系列模拟量输入模块

### 1. 一般配置

模拟量输入模块包括4个通道，每个通道都有参数配置和I/O映射寄存器（16位）设置，下面仅对每个模块的一个通道进行说明。



- 模块诊断上报：模块出现故障时，是否上报给父设备（如CPU、远程模块从站），如果上报，则若父设备设置了故障停机，则父设备会停止本设备的运行。
- 使能通道：是否激活此通道，只有激活了通道，此通道才能使用。
- 通道诊断上报：模块对应的通道出现故障时，是否上报给父设备（如CPU、远程模块从站），如果上报。若父设备设置了故障停机，则父设备会停止本设备的运行。
- 转换模式：设置模拟量输入转换类型，此设置决定了通道输入转换类型和转换值的范围，转换类型及对应的数字量值关系、范围如下表所示。

-	输入额定范围	额定对应数字量	输入极限范围	极限对应数字量
模拟电压输入	-10V~10V	-20000~20000	-11V~11V	-22000~22000
	0V~10V	0~20000	-0.5V~10.5V	-1000~21000
	-5V~5V	-20000~20000	-5.5V~5.5V	-22000~22000
	0V~5V	0~20000	-0.25V~5.25V	-1000~21000
	1V~5V	0~20000	0.8V~5.2V	-1000~21000
模拟电流输入	-20mA~20mA	-20000~20000	-22mA~22mA	-22000~22000
	0mA~20mA	0~20000	-1 mA~21mA	-1000~21000
	4mA~20mA	0~20000	3.2mA~20.8mA	-1000~21000

- 滤波参数：模拟量输入通道滤波时间，范围1ms~255ms。
- 断线标志：设置模拟量输入通道是否检测断线，因不能区分模拟量输入0值和断线，所以所有转换模式范围内，包含0值输入的，不能激活断线标志。
- 超限标志：设置模拟量输入通道是否检测超限。
- 峰值保持功能：设置模拟量输入通道是否保持峰值输入。

## 2. AI4 I/O映射

AI4为4通道模拟量输入，每个通道模拟输入对应一个16位的整数值，模拟量值和数字量值关系见模拟量输入一般配置。可以在此界面面对每个16位整数值映射一个变量，以获取一个输入通道模拟量对应的数字量值。具体请参照I/O映射链接。

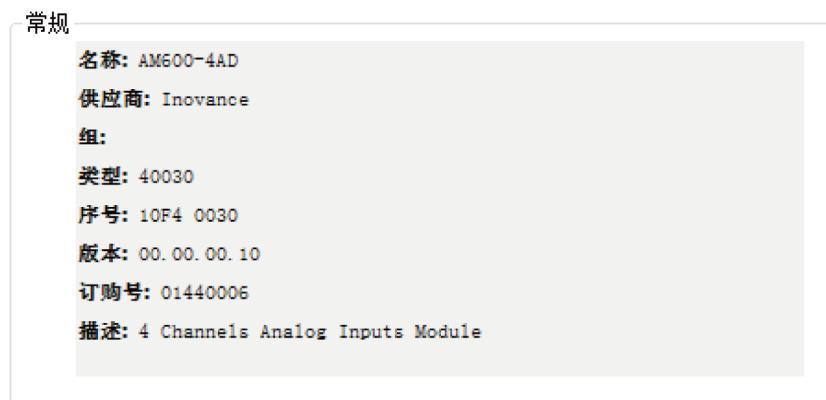
通道							
变量	映射	通道	地址	类型	单位	描述	
AI4CH		AI4CH	%IW2	ARRAY [0..3] OF INT			
		AI4CH[0]	%IW2	INT			
		AI4CH[1]	%IW3	INT			
		AI4CH[2]	%IW4	INT			
		AI4CH[3]	%IW5	INT			

=创建新变量      
 =映射到现有变量      
 Always update variables: Enabled 1 (use bus cycle task if not used in any task)

### 3. 信息

显示AI4模块设备基本信息：名称、供应商、组、类型、序号、版本，模块号、描述、订购号、图像。

另外，登录后信息页面会显示AI4模块单板软件版本和逻辑软件版本，单板软件版本为AI4嵌入式软件版本，逻辑软件版本为AI4模块内FPGA软件版本。



## GL10系列模拟量输出模块

### 1. 一般配置

模拟量输出模块包括4个通道，每个通道都包括参数配置和I/O映射寄存器设置（16位），下面仅对每个模块的一个通道进行说明。



- 模块诊断上报：模块出现故障时，是否上报给父设备（如CPU、远程模块从站）。如果上报，则若父设备设置了故障停机，则父设备会停止本设备的运行。
- 使能通道：是否激活此通道，只有激活了通道，此通道才能使用。
- 通道诊断上报：模块对应的通道出现故障时，是否上报给父设备（如CPU、远程模块从站），如果上报，则若父设备设置了故障停机，则父设备会停止本设备的运行。

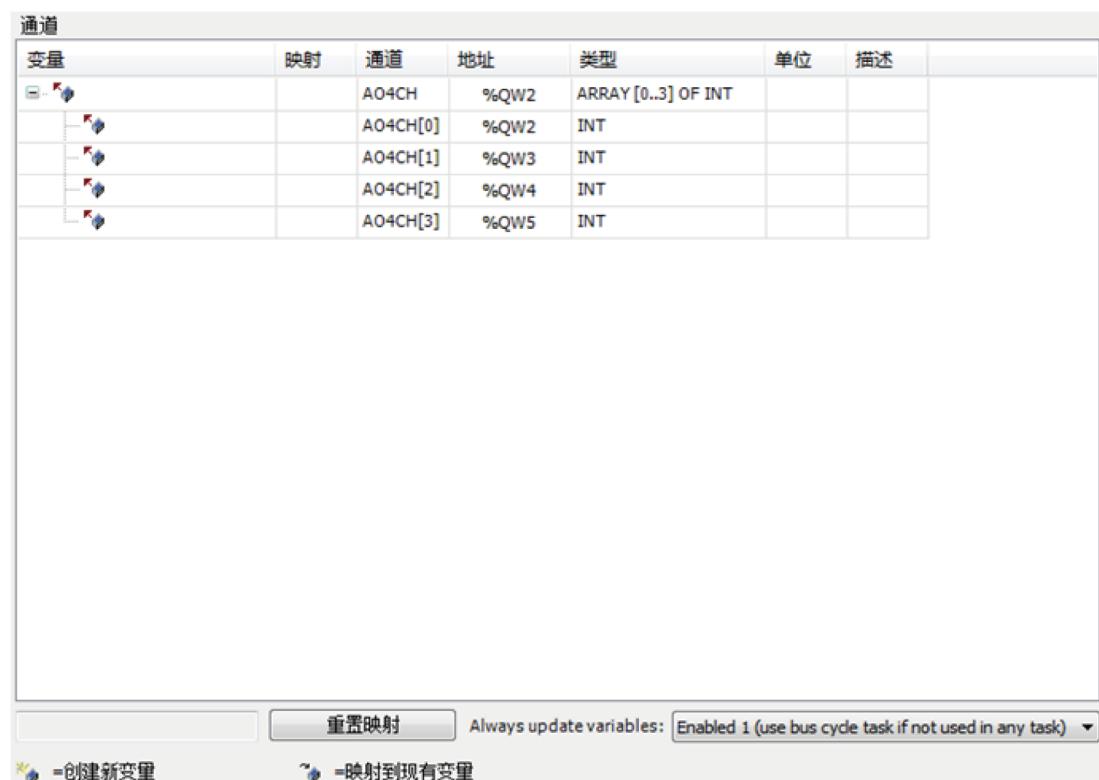
- 转换模式：设置模拟量输出转换类型，此设置决定了此通道输出转换类型和转换的值范围，转换类型及对应的数字量值关系、范围如下表：

~	输出额定范围	额定对应数字量	输出极限范围	极限对应数字量
模拟电压输出	-10V~10V	-20000~20000	-11V~11V	-22000~22000
	0V~10V	0~20000	-0.5V~10.5V	-1000~21000
	-5V~5V	-20000~20000	-5.5V~5.5V	-22000~22000
	0V~5V	0~20000	-0.25V~5.25V	-1000~21000
	1V~5V	0~20000	0.8V~5.2V	-1000~21000
模拟电流输出	0mA~20mA	0~20000	0 mA~21mA	0~21000
	4mA~20mA	0~20000	3.2mA~20.8mA	-1000~21000

- 停止后输出状态：模块运行停止后，设置输出保持值。
- 输出清零：模块运行停止后，输出一直为0。
- 输出保持：模块运行停止后，输出一直保持上次输出值。
- 输出预设值：模块运行停止后，输出一直为预定值。预设值可以设置模拟量值，也可以设置数字量值，模拟量值和数字量值一一对应，修改其中一个，另外一个自动变化。预设值范围和当前转换模式有关，具体请参见上述转换模式。

## 2. AO4 I/O映射

AO4为4通道模拟量输出，每个通道模拟输出对应一个16位的整数值，模拟量值和数字量值关系见模拟量输出一般配置，如下图所示。可以在此界面对每个16位整数值映射一个变量，以此变量值输出到当前通道，然后经模拟量输出模块转换为模拟量值输出。具体请参照I/O映射链接。



## 3. 信息

显示AO4模块设备基本信息：名称、供应商、组、类型、序号、版本、模块号、描述、订购号、图像。

另外，登录后信息页面会读取AO4模块单板软件版本和逻辑软件版本并显示出来，单板软件版本为AO4嵌入式软件版本，逻辑软件版本为AO4模块内FPGA软件版本。

**常规**

名称: AM600-4DA  
供应商: Inovance Control Technology  
组:  
类型: 40040  
序号: 10F4 0040  
版本: 00.00.00.10  
订购号: 01440007  
描述: 4 Channels Analog Outputs Module

**版本**

单板软件版本:  
逻辑软件版本:

**图像**

无图像

## GL10系列温度检测模块

温度检测模块包含4TC（4通道温度检测模块，支持热电偶）、8TC（8通道温度检测模块，支持热电偶）和4PT（4通道温度检测模块，支持热电阻），都有对应的一般配置和通道配置。

一般配置用于配置温度检测模块的单位类型和采样周期，通道配置分别配置每个通道传感器类型、滤波时间、超限、温度偏移等参数。

### 1. 一般配置

不同类型温度检测模块配置有稍微差异，4TC和8TC有冷端补偿功能，4PT不支持。另外8TC支持外部冷端补偿，而4TC不支持。下图以8TC配置界面为例。

模块诊断上报

热电偶冷端补偿

内部冷端补偿  外部冷端补偿

温度单位

摄氏度(°C)  华氏度(°F)

采样周期

250ms  500ms  1000ms

- 模块诊断上报：模块出现故障时，是否上报给父设备（如CPU、远程模块从站），如果上报，则若父设备设置了故障停机，则父设备会停止本设备的运行。
- 冷端补偿：选择冷端补偿方式。只有8TC才支持外部冷端补偿，并且8TC使用通道7（最后一个通道）来进行外部冷端补偿输入。
- 温度单位：温度检测模块输入使用的单位，支持摄氏度或者华氏度。
- 采样周期：设置温度检测模块采样时间，支持250ms、500ms和1000ms。

## 2. 通道配置

不同类型模块支持不同的通道。4TC和4PT支持4通道；8TC支持8通道；由于每个通道的配置参数基本相同，仅对一个通道进行说明。8TC一个通道配置如下图。

通道 - 0

使能通道  通道诊断上报

传感器类型:  滤波时间:

超限检测 温度下限(°C):  (-270-1370) 温度上限(°C):  (-270-1370)

温度偏移 偏移值(°C):  (-204.8-204.7)

传感器断线检测

- 使能通道：是否激活此通道，只有激活了通道，此通道才能使用。
- 通道诊断上报：模块对应的通道出现故障时，是否上报给父设备（如CPU、远程模块从站），如果上报，则若父设备设置了故障停机，则父设备会停止本设备的运行。
- 缺省值：复位此通道值为默认值。
- 传感器类型：8TC和4TC传感器类型、传感器规格如下表。默认值使用K传感器。

项目	传感器名称	摄氏温度范围 (°C)	华氏温度范围 (°F)
热电偶类型	B	250°C~1800°C	482°F~3272°F
	E	-270°C~1000°C	-454°F~1832°F
	N	-200°C~1300°C	-328°F~2372°F
	J	-210°C~1200°C	-346°F~2192°F
	K	-270°C~1372°C	-454°F~2502°F
	R	-50°C~1768°C	-58°F~3214°F
	S	-50°C~1768°C	-58°F~3214°F
	T	-270°C~400°C	-454°F~752°F

项目	传感器名称	摄氏温度范围 (°C)	华氏温度范围 (°F)
热电阻类型	Pt100	-200°C~850°C	-328°F~1562°F
	Pt500	-200°C~850°C	-328°F~1562°F
	Pt1000	-200°C~850°C	-328°F~1562°F
	Cu100	-50°C~150°C	-58°F~302°F

- 滤波参数：温度检测模块此通道使用的滤波时间，范围0s~100s。默认5s。
- 超限检测：使用此通道超限检测功能，如果不在温度上下限之间，会报超限故障，范围请参见上表。
- 温度偏移：设置温度检测模块偏移补偿值，范围-204.8~204.7。
- 传感器断线检测：使能传感器断线报警功能。

### 3. I/O映射

不同类型的温度检测模块包含不同个数的通道，相应的，也包含不同个数的I/O映射。下图为4PT的I/O映射界面，每个通道参数值为温度值。具体请参照I/O映射链接。

变量	映射	通道	地址	类型	默认值	单位	描述
		Temperature	%ID13	ARRAY [0..3] OF REAL			
		Temperature[0]	%ID13	REAL			
		Temperature[1]	%ID14	REAL			
		Temperature[2]	%ID15	REAL			
		Temperature[3]	%ID16	REAL			

### 4. 信息

显示温度检测模块设备基本信息：名称、供应商、组、类型、序号、版本，模块号、描述、订购号、图像。

另外，登录后信息页面会显示温度检测模块单板软件版本和逻辑软件版本，单板软件版本为温度检测模块嵌入式软件版本，逻辑软件版本为温度检测模块内FPGA软件版本。



## GL20(S)数字量输入模块

数字量输入模块包含通道配置、设备诊断、I/O映射、状态和信息页面，一般情况下只需要在I/O映射界面映射I/O变量以获取数字量输入值。数字量输入包含8点、16点和32点模块，三者的界面相似，以16点数字量输入模块为例。

### 1. 通道配置。

通道配置界面如下图所示，可以设置每个通道的滤波时间。



### 2. 设备诊断。

设备诊断界面如下图所示，在模块发生错误时，可以通过设备诊断界面查看模块的错误信息。



## 3. I/O映射。

I/O映射界面如下图所示，可通过在I/O映射界面对每位或者每8位映射到一个变量，获取输入值，具体请参见[第142页 “4.2.6 I/O映射参数说明”](#)。

I/O映射							
变量	映射	通道	地址	类型	默认值	单位	描述
IB(I)		IB(I)	%IB1	USINT			
IB(II)		IB(II)	%IB2	USINT			

下方工具栏包含：查找、过滤、显示所有、给I/O通道添加FB...、转到实例、连续I/O地址。下方状态栏显示：复位所有映射变量、一直更新变量、使能1(如果未在任何任务中使用则使用总线循环)。图标说明：=创建新变量、=映射到现有变量。

## 4. 信息。

显示模块设备基本信息：名称、供应商、组、类型、序号、版本，订购号、描述、图像，如下图所示。另外，登录后信息页面会显示模块逻辑软件版本，逻辑软件版本为模块内FPGA软件版本。

常规

名称: GL20 16通道DI模块  
供应商: Inovance  
组:  
类型: 41010  
序号: 10F4 1010  
版本: 00.00.00.10  
订购号:  
描述: 16通道DI模块

图像

无图像

## GL20(S)数字量输出模块

数字量输出模块包含通道配置、设备诊断、I/O映射、状态和信息页面，一般情况下只需要在I/O映射界面映射I/O变量，然后把映射变量值输出到数字量输出模块。数字量输出包含8点、16点和32点模块，三者的界面相似，以16点数字量输出模块为例。

### 1. 通道配置。

通道配置界面如下图所示，设置每个通道的停止/断网后输出状态，可设置为“全部输出保持”、“全部输出预设值”和“按位设置”。

组	0	1	2	3	4	5	6	7
I	FALSE							

组	0	1	2	3	4	5	6	7
I	FALSE							

### 2. 设备诊断。

设备诊断界面如下图所示，在模块发生错误时，通过设备诊断界面查看模块的错误信息。

诊断状态		
序号	诊断状态	诊断代码

### 3. I/O映射。

通过在I/O映射界面对每位或者每8位映射到一个变量，以输出变量值，具体请参见[第142页 “4.2.6 I/O映射参数说明”](#)，如下图所示。

变量	映射	通道	地址	类型	默认值	单位	描述
	Q8(I)		%QB1	USINT			
	Q0		%QX1.0	BOOL			
	Q1		%QX1.1	BOOL			
	Q2		%QX1.2	BOOL			
	Q3		%QX1.3	BOOL			
	Q4		%QX1.4	BOOL			
	Q5		%QX1.5	BOOL			
	Q6		%QX1.6	BOOL			
	Q7		%QX1.7	BOOL			
	QB(II)		%QB2	USINT			

#### 4. 信息。

显示DO16模块设备基本信息：名称、供应商、组、类型、序号、版本，模块号、描述、订购号、图像，如下图所示。另外，登录后信息页面会显示DO模块逻辑软件版本，逻辑软件版本为DO模块内FPGA软件版本。



## GL20数字量输入输出模块

数字量输入输出模块包含通道配置、设备诊断、I/O映射、状态和信息页面，以8点数字量输入模块为例。

#### 1. 通道配置。

通道配置界面如下图所示，设置每个通道的滤波时间和停止/断网后输出状态，停止/断网后输出状态可设置为“全部输出保持”、“全部输出预设值”和“按位设置”。

## 2. 设备诊断。

设备诊断界面如下图所示，在模块发生错误时，可通过设备诊断界面查看模块的错误信息。

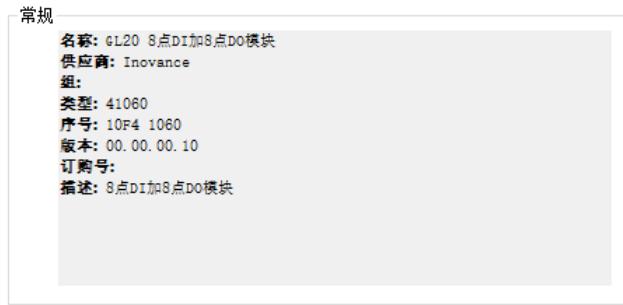
### 3. I/O映射。

I/O映射界面如下图所示，通过在I/O映射界面对每位或者每8位映射到一个变量，获取输入值，具体请参见第142页“4.2.6 IO映射参数说明”。

变量		映射	通道	地址	类型	默认值	单位	描述
			IB(I)	%IX1.0	USINT			
			I0	%IX1.0	BOOL			
			I1	%IX1.1	BOOL			
			I2	%IX1.2	BOOL			
			I3	%IX1.3	BOOL			
			I4	%IX1.4	BOOL			
			I5	%IX1.5	BOOL			
			I6	%IX1.6	BOOL			
			I7	%IX1.7	BOOL			
			QB(I)	%Q01	USINT			
			Q0	%QX1.0	BOOL			
			Q1	%QX1.1	BOOL			
			Q2	%QX1.2	BOOL			
			Q3	%QX1.3	BOOL			
			Q4	%QX1.4	BOOL			
			Q5	%QX1.5	BOOL			
			Q6	%QX1.6	BOOL			

4. 信息

显示模块设备基本信息：名称、供应商、组、类型、序号、版本，订购号、描述、图像，如下图所示。另外，登录后信息页面会显示DIDO模块逻辑软件版本，逻辑软件版本为DIDO模块内FPGA软件版本。



## GL20(S)模拟量输入模块

模拟量输入模块包含4AD（4路模拟量输入）、8ADI（8路模拟量输入，支持电流输入）和8ADV（8路模拟量输入，支持电压输入），页面包括通道配置、设备诊断、I/O映射、状态和信息页面，以4AD模拟量输入模块为例。

### 1. 通道配置。

模拟量输入模块包括4个通道，每个通道都有参数配置和I/O映射寄存器（16位）设置，下面仅对每个模块的一个通道进行说明。

**▲ 通道 - 0**

使能通道

数字输出范围  -20000~20000  -32000~32000  -27648~27648

转换模式:  滤波参数:

断线标志  超限标志  峰值保持功能  溢出检测

**▲ 通道 - 1**

使能通道

数字输出范围  -20000~20000  -32000~32000  -27648~27648

转换模式:  滤波参数:

断线标志  超限标志  峰值保持功能  溢出检测

**▲ 通道 - 2**

使能通道

数字输出范围  -20000~20000  -32000~32000  -27648~27648

转换模式:  滤波参数:

断线标志  超限标志  峰值保持功能  溢出检测

**▲ 通道 - 3**

使能通道

数字输出范围  -20000~20000  -32000~32000  -27648~27648

转换模式:  滤波参数:

断线标志  超限标志  峰值保持功能  溢出检测

- 使能通道：是否激活此通道，只有激活通道，此通道才能使用。
- 数字输出范围：设置模拟量输出量程，决定此通道输出转换的值范围。
- 转换模式：设置模拟量输入转换类型，此设置决定了通道输入转换类型和转换值的范围，转换类型及对应的数字量值关系、范围如下表所示。

-	输入额定范围	额定对应数字量	输入极限范围	极限对应数字量
模拟电压输入	-10V~10V	-20000~20000 -32000~32000 -27648~27648	-10.24V~10.24V	-20400~20400 -32640~32640 -28200~28200
	0V~10V	0~20000 0~32000 0~27648	-0.5V~10.24V	-1000~20400 -1600~32640 -1382~28200
	-5V~5V	-20000~20000 -32000~32000 -27648~27648	-5.12V~5.12V	-20400~20400 -32640~32640 -28200~28200
	0V~5V	0~20000 0~32000 0~27648	-0.25V~5.12V	-1000~20400 -1600~32640 -1382~28200
	1V~5V	0~20000 0~32000 0~27648	0.8V~5.12V	-1000~20400 -1600~32640 -1382~28200
	-20mA~20mA	-20000~20000 -32000~32000 -27648~27648	-20.56mA~20.56mA	-20400~20400 -32640~32640 -28200~28200
	0mA~20mA	0~20000 0~32000 0~27648	-1mA~20.56mA	-1000~20400 -1600~32640 -1382~28200
	4mA~20mA	0~20000 0~32000 0~27648	3.2mA~20.56mA	-1000~20400 -1600~32640 -1382~28200

- 滤波参数：模拟量输入通道滤波时间，范围1ms~255ms。
- 断线标志：设置模拟量输入通道是否检测断线，因不能区分模拟量输入0值和断线，所以所有转换模式范围内，包含0值输入的，不能激活断线标志。
- 超限标志：设置模拟量输入通道是否检测超限。
- 峰值保持功能：设置模拟量输入通道是否保持峰值输入。

## 2. 设备诊断。

设备诊断界面如下图所示，在模块发生错误时，通过设备诊断界面查看模块的错误信息。



## 3. I/O映射。

AI4为4通道模拟量输入，每个通道模拟输入对应1个16位的整数值，模拟量值和数字量值关系请参见通道配置。可以在此界面面对每个16位整数值映射1个变量，以获取1个输入通道模拟量对应的数字量值，具体请参见[第142页“4.2.6 IO映射参数说明”](#)。

The screenshot shows a software interface for configuring I/O mappings. At the top, there are tabs for '查找' (Search), '过滤' (Filter), '显示所有' (Show All), and buttons for '给I/O通道添加FB...' (Add FB to I/O channel), '转到实例' (Go to instance), and '连续I/O地址' (Continuous I/O address). Below the tabs is a table with columns: 变量 (Variable), 映射 (Mapping), 通道 (Channel), 地址 (Address), 类型 (Type), 默认值 (Default Value), 单位 (Unit), and 描述 (Description). The table contains four rows, each representing a channel mapping:

变量	映射	通道	地址	类型	默认值	单位	描述
		AI4CH	%IW1	ARRAY [0..3] OF INT			
	AI4CH[0]		%IW1	INT			
	AI4CH[1]		%IW2	INT			
	AI4CH[2]		%IW3	INT			
	AI4CH[3]		%IW4	INT			

At the bottom of the interface are several buttons: '复位所有映射变量' (Reset all mapped variables), '一直更新变量' (Update variable continuously), and a dropdown menu '使能1(如果未在任何任务中使用则使用总线循环)'. Below these buttons are two icons: a blue star icon labeled '=创建新变量' (Create new variable) and a blue star with a minus sign icon labeled '=映射到现有变量' (Map to existing variable).

#### 4. 信息。

显示模块设备基本信息：名称、供应商、组、类型、序号、版本，模块号、描述、订购号、图像。另外，登录后信息页面会显示AI4模块单板软件版本和逻辑软件版本，单板软件版本为AI4嵌入式软件版本，逻辑软件版本为AI4模块内FPGA软件版本。



## GL20(S)模拟量输出模块

模拟量输出模块包含通道配置、设备诊断、I/O映射、状态和信息页面。

#### 1. 通道配置。

模拟量输出模块包括4个通道，每个通道都包括参数配置和I/O映射寄存器设置（16位），下面仅对每个模块的一个通道进行说明。

**▲ 通道 - 0**

使能通道

转换模式:

数字输出范围  -20000~20000  -32000~32000  -27648~27648

停止后输出状态  输出清零  输出保持  输出预设值

---

**▲ 通道 - 1**

使能通道

转换模式:

数字输出范围  -20000~20000  -32000~32000  -27648~27648

停止后输出状态  输出清零  输出保持  输出预设值

---

**▲ 通道 - 2**

使能通道

转换模式:

数字输出范围  -20000~20000  -32000~32000  -27648~27648

停止后输出状态  输出清零  输出保持  输出预设值

---

**▲ 通道 - 3**

使能通道

转换模式:

数字输出范围  -20000~20000  -32000~32000  -27648~27648

停止后输出状态  输出清零  输出保持  输出预设值

- 使能通道：是否激活此通道，只有激活通道，此通道才能使用。
- 数字输出范围：设置模拟量输出量程，决定此通道输出转换的值范围。
- 转换模式：设置模拟量输出转换类型，此设置决定了此通道输出转换类型和转换的值范围，转换类型及对应的数字量值关系、范围如下表所示。

-	输出额定范围	额定对应数字量	输出极限范围	极限对应数字量
模拟电压输出	-10V~10V	-20000~20000 -32000~32000 -27648~27648	-10.24V~10.24V	-20400~20400 -32640~32640 -28200~28200
	0V~10V	0~20000 0~32000 0~27648	-0.5V~10.24V	-1000~20400 -1600~32640 -1382~28200
	-5V~5V	-20000~20000 -32000~32000 -27648~27648	-5.12V~5.12V	-20400~20400 -32640~32640 -28200~28200
	0V~5V	0~20000 0~32000 0~27648	-0.25V~5.12V	-1000~20400 -1600~32640 -1382~28200
	1V~5V	0~20000 0~32000 0~27648	0.8V~5.12V	-1000~20400 -1600~32640 -1382~28200
	0mA~20mA	0~20000 0~32000 0~27648	-1mA~20.56mA	-1000~20400 -1600~32640 -1382~28200
	4mA~20mA	0~20000 0~32000 0~27648	3.2mA~20.56mA	-1000~20400 -1600~32640 -1382~28200

- 停止后输出状态：模块运行停止后，设置输出保持值。
- 输出清零：模块运行停止后，输出一直为0。
- 输出保持：模块运行停止后，输出一直保持上次输出值。
- 输出预设值：模块运行停止后，输出一直为预定值。预设值可以设置模拟量值，也可以设置数字量值，模拟量值和数字量值一一对应，修改其中一个，另外一个自动变化。预设值范围和当前转换模式有关，具体请参见上述转换模式。

## 2. 设备诊断。

设备诊断界面如下图所示，在模块发生错误时，通过设备诊断界面查看模块的错误信息。



## 3. I/O映射。

AO4为4通道模拟量输出，每个通道模拟输出对应一个16位的整数值，模拟量值和数字量值关系请参见通道配置，如下图所示。可以在此界面对每个16位整数值映射一个变量，以把此变量值输出到当前通道，然后经模拟量输出模块转换为模拟量值输出，具体请参见[第142页 “4.2.6 I/O映射参数说明”](#)。

查找 过滤 显示所有							给IO通道添加FB...	转到实例	连续IO地址
变量	映射	通道	地址	类型	默认值	单位	描述		
AO4CH		AO4CH	%QW1	ARRAY [0..3] OF INT					
		AO4CH[0]	%QW1	INT					
		AO4CH[1]	%QW2	INT					
		AO4CH[2]	%QW3	INT					
		AO4CH[3]	%QW4	INT					

=创建新变量 =映射到现有变量
复位所有映射变量 一直更新变量: 使能1(如果未在任何任务中使用则使用总线循环)

#### 4. 信息。

显示模块设备基本信息：名称、供应商、组、类型、序号、版本、模块号、描述、订购号、图像。另外，登录后信息页面会读取AO4模块单板软件版本和逻辑软件版本并显示出来，单板软件版本为AO4嵌入式软件版本，逻辑软件版本为AO4模块内FPGA软件版本。



## GL20(S)温度检测模块

温度检测模块包含4PT（4通道温度检测模块，支持热电阻）和4TC（4通道温度检测模块，支持热电偶），有对应的一般配置和通道配置。一般配置用于配置温度检测模块的单位类型和采样周期，通道配置分别配置每个通道传感器类型、滤波时间、超限、温度偏移等参数，以4PT温度检测模块为例。

#### 1. 一般配置。

温度单位		
<input checked="" type="radio"/> 摄氏度(°C)	<input type="radio"/> 华氏度(°F)	
采样周期		
<input type="radio"/> 250ms	<input type="radio"/> 500ms	<input type="radio"/> 1000ms

- 温度单位：温度检测模块输入使用的单位，支持摄氏度或者华氏度。
- 采样周期：设置温度检测模块采样时间，支持250ms、500ms和1000ms。

## 2. 通道配置。

▲ 通道 - 0

<input checked="" type="checkbox"/> 使能通道	<input type="button" value="缺省值"/>
传感器类型: Pt100	滤波时间: 5 s
<input type="checkbox"/> 超限检测 温度下限(°C): -200 (-200-850)      温度上限(°C): 850 (-200-850)	
<input type="checkbox"/> 温度偏移 偏移值(°C): 0 (-204.8-204.7)	
<input type="checkbox"/> 传感器断线检测 <input type="checkbox"/> 溢出检测	

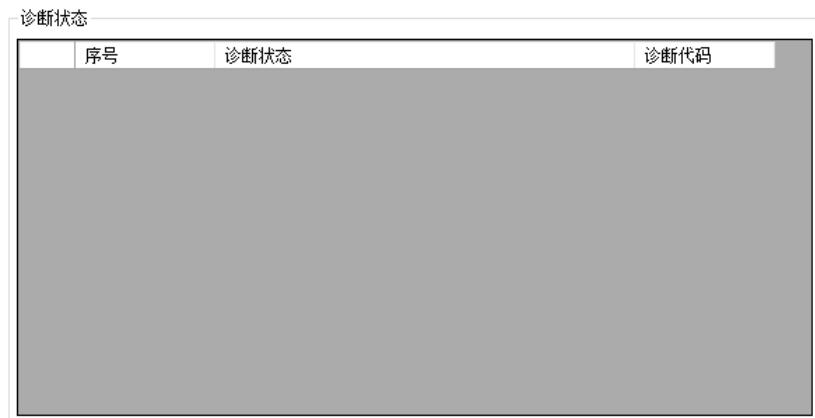
- 使能通道：是否激活此通道，只有激活通道，此通道才能使用。
- 缺省值：复位此通道值为默认值。
- 传感器类型：4PT传感器类型、传感器规格如下表所示，默认值使用K传感器。

类型	传感器名称	摄氏温度范围 (°C)	华氏温度范围 (°F)
热电阻	Pt100	-200°C~850°C	-328°F~1562°F
	Pt500	-200°C~850°C	-328°F~1562°F
	Pt1000	-200°C~850°C	-328°F~1562°F
	Cu100	-50°C~150°C	-58°F~302°F
	KTY84	-50°C~150°C	32°F~392°F
	NTC5K (B值2000)	-30°C~200°C	-22°F~392°F
	NTC5K (B值3950)	-15°C~100°C	5°F~212°F
	NTC5K (B值6000)	0°C~100°C	32°F~212°F
	NTC10K (B值2000)	-25.0°C~200°C	-13°F~392°F
	NTC10K (B值3950)	0°C~150°C	32°F~302°F
	NTC10K (B值6000)	-6°C~100°C	42.8°F~212°F

- 滤波参数：温度检测模块此通道使用的滤波时间，范围0s~100s。默认5s。
- 超限检测：使用此通道超限检测功能，如果不在温度上下限之间，会报超限故障。
- 温度偏移：设置温度检测模块偏移补偿值，范围-204.8~204.7。
- 传感器断线检测：使能传感器断线报警功能。
- 溢出检测：使能传溢出报警功能。

## 3. 设备诊断。

设备诊断界面如下图所示，在模块发生错误时，可以通过设备诊断界面查看模块的错误信息。



#### 4. I/O映射。

不同类型的温度检测模块包含不同个数的通道，相应的也包含不同个数的I/O映射。下图为4PT的I/O映射界面，每个通道参数值为温度值，具体请参见[第142页 “4.2.6 I/O映射参数说明”](#)。

变量	映射	通道	地址	类型	默认值	单位	描述
		Temperature	%ID1	ARRAY [0..3] OF REAL			
		Temperature[0]	%ID1	REAL			
		Temperature[1]	%ID2	REAL			
		Temperature[2]	%ID3	REAL			
		Temperature[3]	%ID4	REAL			

#### 5. 信息。

显示温度检测模块设备基本信息：名称、供应商、组、类型、序号、版本，模块号、描述、订购号、图像。另外，登录后信息页面会显示温度检测模块单板软件版本和逻辑软件版本，单板软件版本为温度检测模块嵌入式软件版本，逻辑软件版本为温度检测模块内FPGA软件版本。

常规

名称: CL20\_4通道PT模块  
供应商: Inovance  
组:  
类型: 41050  
序号: 10F4\_1050  
版本: 00.00.00.10  
订购号:  
描述: 4通道PT模块

图像

无图像

## GL20通信模块

GL20通信模块包括2CAN, 2S485、2SCOM和1DNM模块，具体配置操作请参见各模块的《用户手册》。

## GL20工艺模块

GL20-2SSI 2通道编码器输入模块，具体配置操作请参见《GL20-2SSI 2通道编码器输入模块用户手册》。

### 4.2.5 高速IO 配置

#### 概述



- InoProShop V1.8.0.0及以上版本、固件1.3.0.0及以上版本支持该功能。
- AM300/AM400/AM500/AM600/AC700系列支持高速IO功能。
- 使能成高速IO的端口无法作为普通端口使用，即无法通过IO映射改变IO状态。
- 本地脉冲轴不支持仿真功能。

高速IO包含以下几个功能：

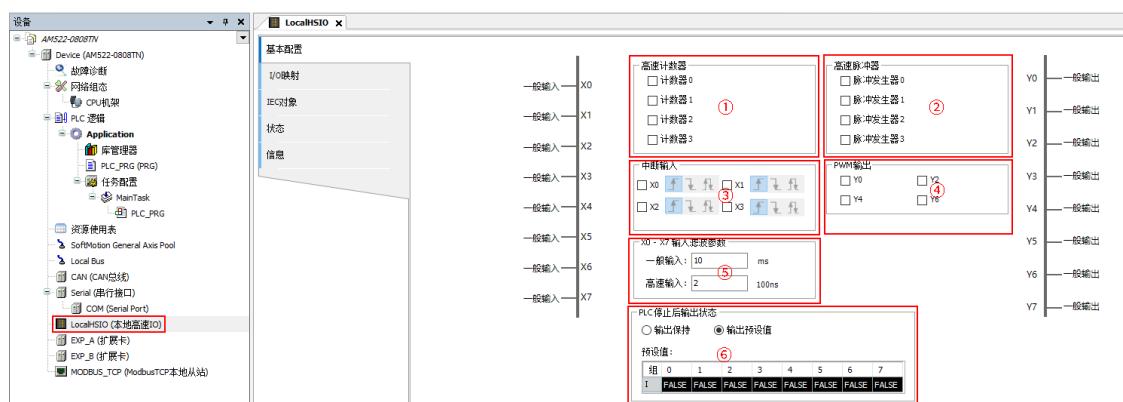
- 高速计数功能
- 高速输出功能
- PWM输出功能
- 高速输入沿中断功能

不同PLC系列的高速IO功能支持路数如下表所示。

功能项	AM400/600系列	AM300/500系列	AC700系列
高速计数功能	8路	4路	4路
高速脉冲功能	4路	4路	-
PWM输出功能	8路	4路	2路
高速输入沿中断功能	8路	4路	-

### AM300/AM500/AC700高速IO配置

1. AM300/AM500/AC700高速IO接线指导，具体请参见各系列产品对应的《硬件手册》或《用户手册》。
2. 在左侧设备树中双击“LocalHSIO(本地高速IO)”，进入“LocalHSIO”配置界面。



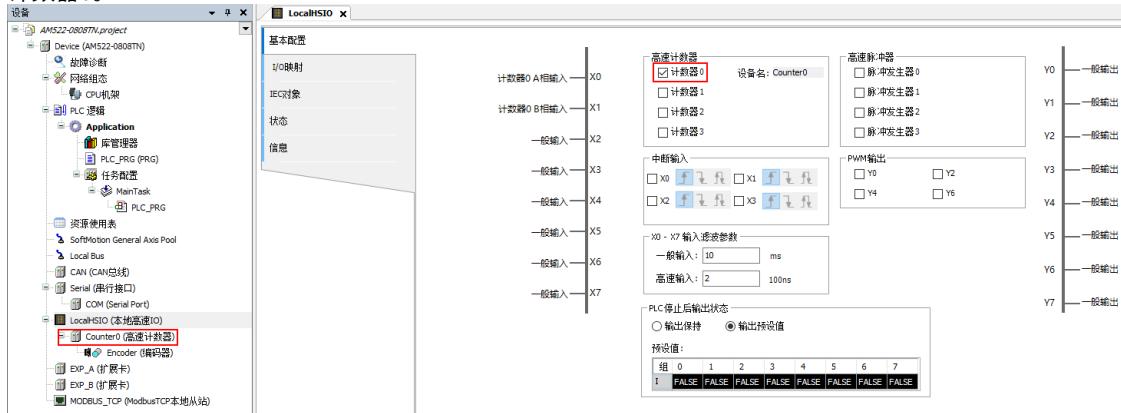
序号	功能配置
①	高速计数器功能配置
②	高速脉冲输出功能配置
③	高速输入沿中断功能配置
④	PWM输出功能配置
⑤	端子滤波参数配置
⑥	PLC停止后输出状态配置

### 配置高速计数器功能

配置高速计数器功能，包括计数模式（单相、AB相1/2/4倍频、脉冲+方向、CW/CCW等）、硬件复位、探针、预置、比较输出。

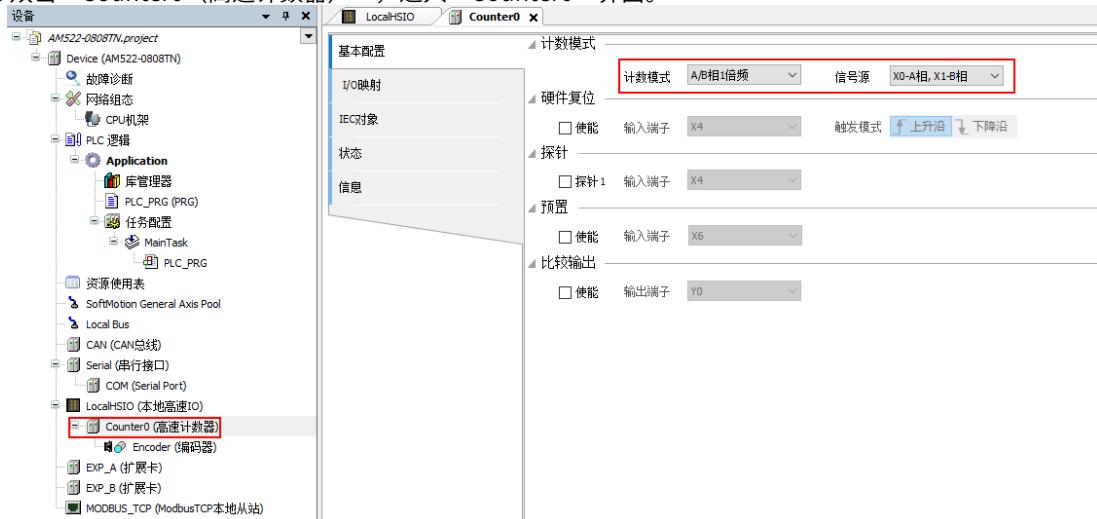
以“计数器0”为例进行介绍（支持计数器0～计数器3）。

- 在“基本配置”界面勾选“计数器0”，在左侧设备树上会自动插入“Counter0（高速计数器）”，使能计数器0。



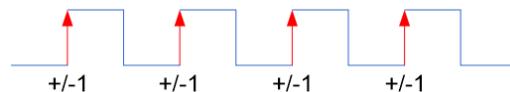
- 配置计数模式和信号源。

- 双击“Counter0（高速计数器）”，进入“Counter0”界面。

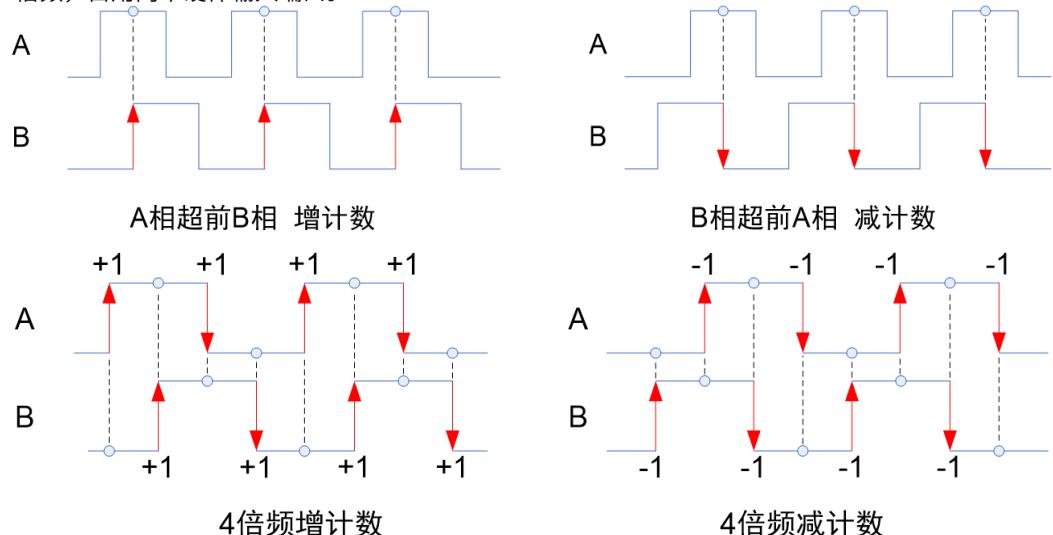


- 在“基本配置”界面“计数模式”下拉选项中选择使用的计数模式，“信号源”下拉选项中选择使用的硬件输入端子。

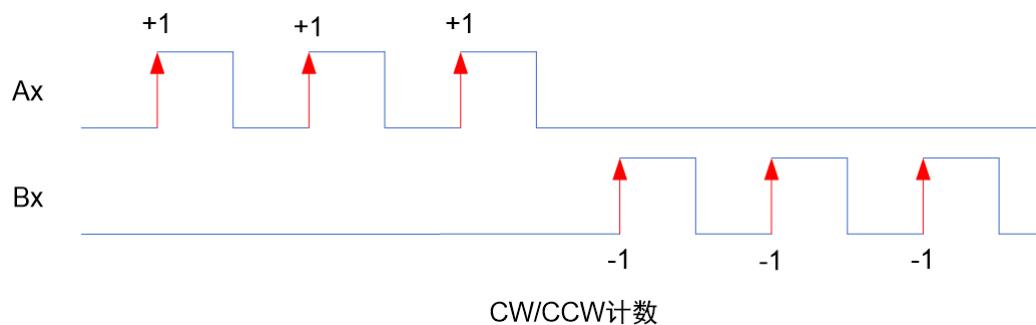
- 单相计数：“计数模式”选择“单相计数”，接收外部单相编码器脉冲信号，只占用一个硬件输入端口；或者选择软件内部定时产生脉冲信号，支持1μs/1ms，不占用硬件输入端口。



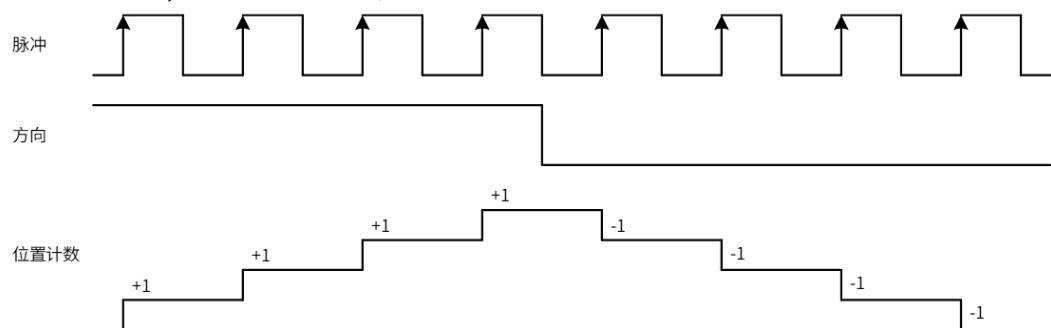
- AB相计数：“计数模式”选择“A/B相1倍频”，接收外部AB相编码器脉冲信号，AB相也可配置为2/4倍频，占用两个硬件输入端口。



- CW/CCW计数：“计数模式”选择“CW/CCW”，接收外部CW/CCW编码器脉冲信号，占用两个硬件输入端口。



- 脉冲+方向计数：“计数模式”选择“脉冲+方向”，在该模式下，方向信号为ON时，高速计数器对脉冲信号增计数；方向信号为OFF时，高速计数器对脉冲信号减计数。



3. 配置硬件复位功能，每个高速计数器配置一路硬件复位。



- 勾选“使能”复选框。
- 根据需求配置输入端子和触发模式。

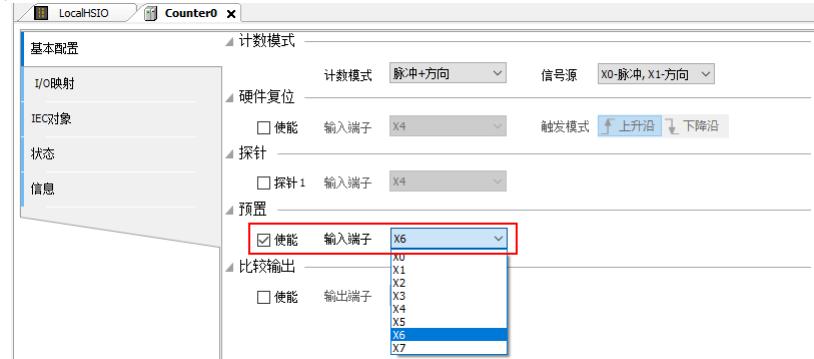
#### 4. 配置探针功能，每个高速计数器配置一路探针输入。



- 勾选“探针1”复选框。
- 根据需求配置输入端子。

配置完成后，通过HC\_TouchProbe功能块，实现计数器的位置锁存，或者通过HC\_VirtualTouchProbe功能块，实现指定轴的位置锁存。

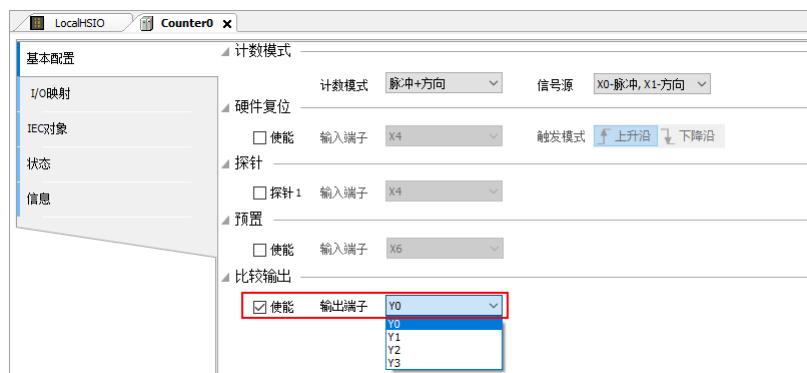
#### 5. 配置预置功能，每个高速计数器配置一路预置输入。



- 勾选“使能”复选框。
- 根据需求配置输入端子。

配置完成后，通过HC\_Preset功能块，实现计数器的位置预置。

#### 6. 配置比较输出功能，每个高速计数器配置一路比较输出功能。

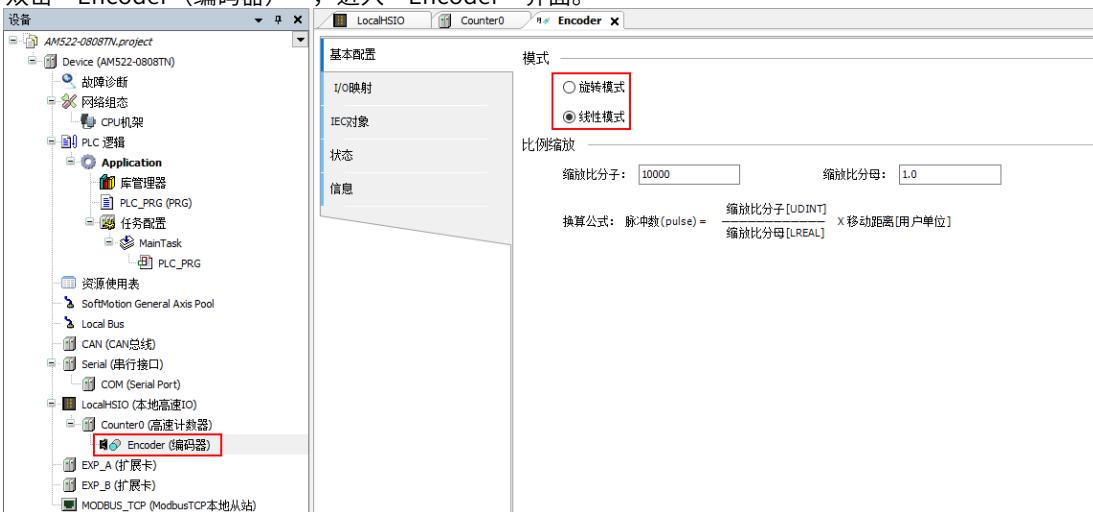


- 勾选“使能”复选框。
- 根据需求配置输出端子。

配置完成后，通过HC\_Compare、HC\_ArrayCompare、HC\_StepCompare功能块，实现计数器的位置比较输出。

## 7. 配置编码器轴模式。

- 双击“Encoder（编码器）”，进入“Encoder”界面。



- 根据实际使用情况选择合适的工作模式（旋转模式或线性模式）。

- 旋转模式：**高速计数器在[0, 旋转周期)的左闭右开区间内循环工作。由于高速计数器为32位计数器，旋转周期换算为脉冲单位后必需在32位整数范围[-2147483648, 2147483647]。
- 线性模式：**高速计数器在[负向限制值, 正向限制值]的闭区间内工作。当方向为负向时，计数值向负方向减小，到达负向限制值后，计数值不再减小；当方向为正向时，计数值向正方向增加，到达正向限制值后，计数值不再增加。由于高速计数器为32位计数器，负向限制值与正向限制值换算为脉冲单位后必需在32位整数范围[-2147483648, 2147483647]。

- 设置缩放比分子和缩放比分母。

高速计数器计数时采用脉冲单位，运动控制指令侧则使用，例如毫米、度、英寸等常见的度量单位，我们称之为用户单位（Unit）。

用户单位到脉冲单位的转换公式如下：

$$\text{脉冲数 (pulse)} = \frac{\text{缩放比分子 [UDINT]}}{\text{缩放比分母 [LREAL]}} \times \text{移动距离 [用户单位]}$$

举例：

缩放比分子= 10000

缩放比分母= 1.0

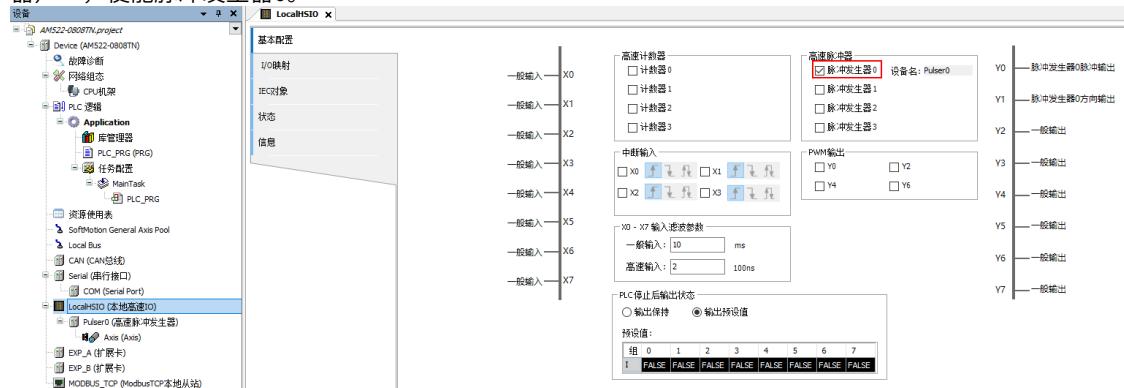
当高速计数器实际接收到的脉冲数为10000时，高速计数器的计数值增加1。

### 配置高速脉冲输出功能

配置高速脉冲轴功能，包括输出模式（仅脉冲、AB相、脉冲+方向、CW/CCW等）、探针、原点、正限位、负限位和急停。

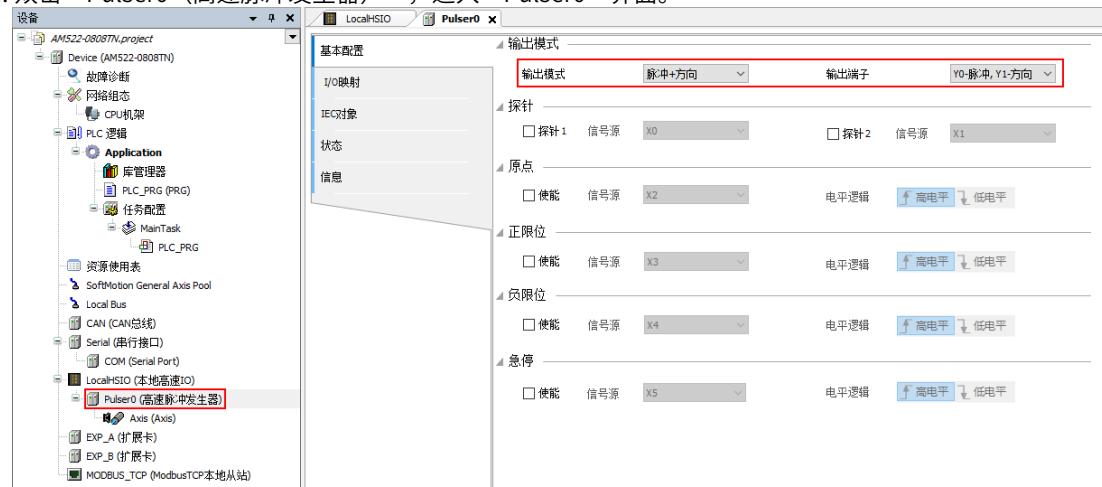
以“脉冲发生器0”为例进行介绍（支持高速脉冲发生器0~3）

- 在“基本配置”界面勾选“脉冲发生器0”，在左侧设备树上会自动插入“Pulser0（高速脉冲发生器）”，使能脉冲发生器0。



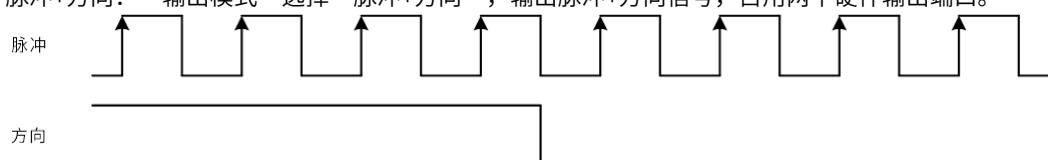
- 配置输出模式和输出端子。

- 双击“Pulser0（高速脉冲发生器）”，进入“Pulser0”界面。

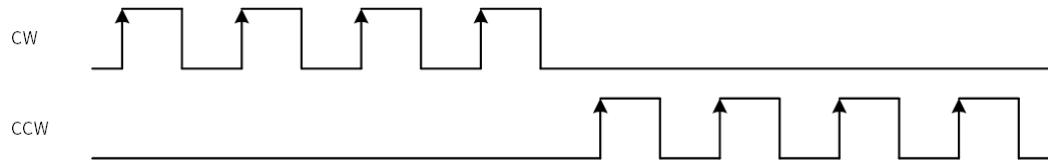


- 在“基本配置”界面“输出模式”下拉选项中选择使用的输出模式，“输出端子”下拉选项中选择使用的硬件输出端子。

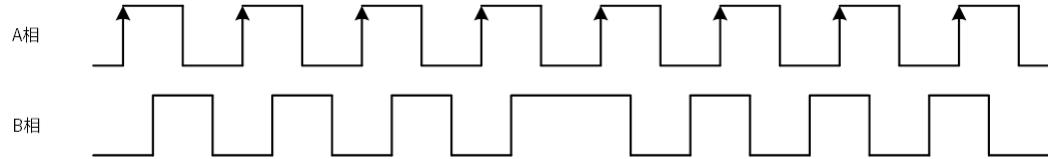
- 脉冲+方向：“输出模式”选择“脉冲+方向”，输出脉冲+方向信号，占用两个硬件输出端口。



- CW/CCW：“输出模式”选择“CW/CCW”，输出CW/CCW信号，占用两个硬件输出端口。



- A/B相：“输出模式”选择“A/B相”，输出AB相信号，占用两个硬件输出端口。



- 仅脉冲：“输出模式”选择“仅脉冲”，输出单相脉冲信号，占用一个硬件输出端口，仅支持Y0/Y1。



### 3. 配置探针功能，每个高速脉冲发生器配置两路探针输入。

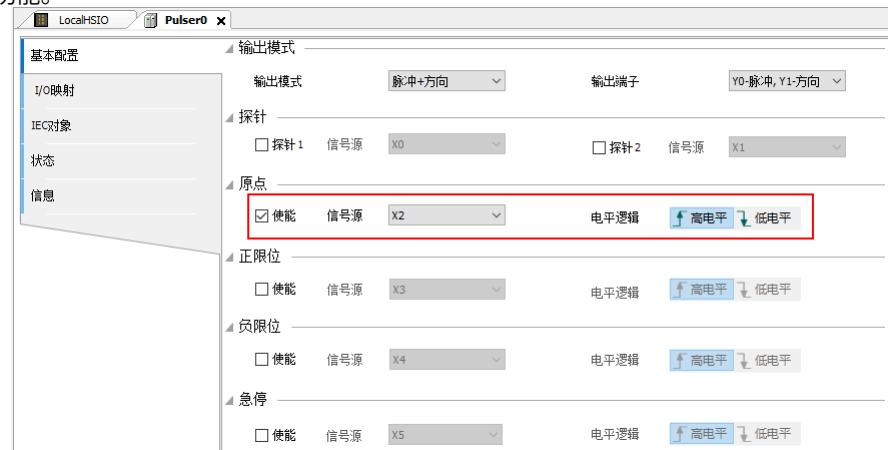


a. 勾选“探针1”复选框或“探针2”复选框。

b. 根据需求配置信号源。

配置完成后，通过MC\_TouchProbe功能块，实现脉冲轴的位置锁存。

### 4. 配置原点功能。

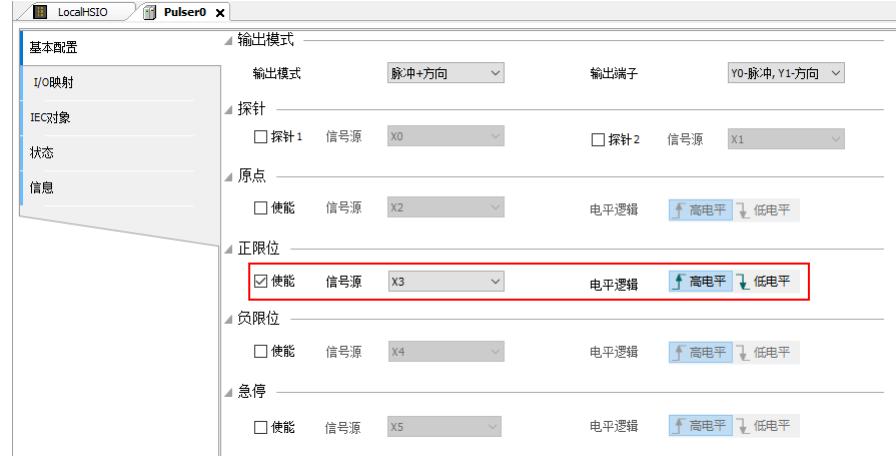


a. 勾选“使能”复选框。

b. 根据需求配置信号源和电平逻辑。

配置完成后，通过MC\_Home功能块，实现脉冲轴的回原功能。回原方式支持CiA 402协议的17-30和35号回原方式。

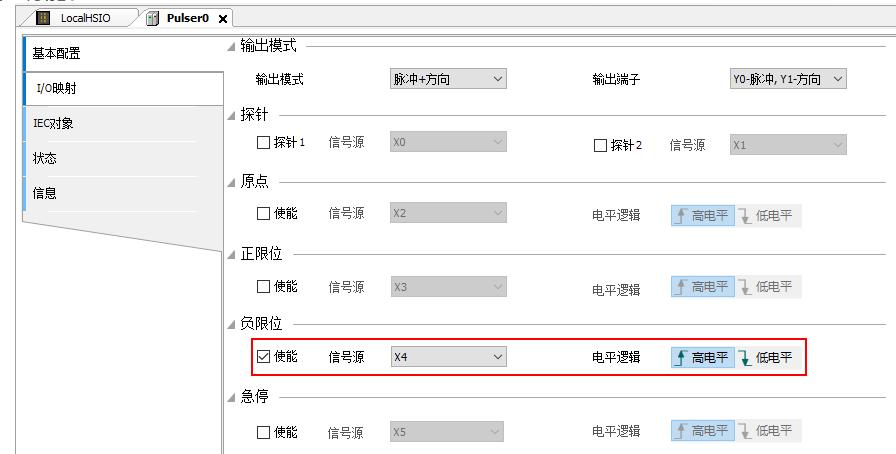
#### 5. 配置正限位功能。



- 勾选“使能”复选框。
- 根据需求配置信号源和电平逻辑。

配置完成后，通过MC\_Home功能块，实现脉冲轴的正限位和回原功能。回原方式支持CiA 402协议的17-30和35号回原方式。

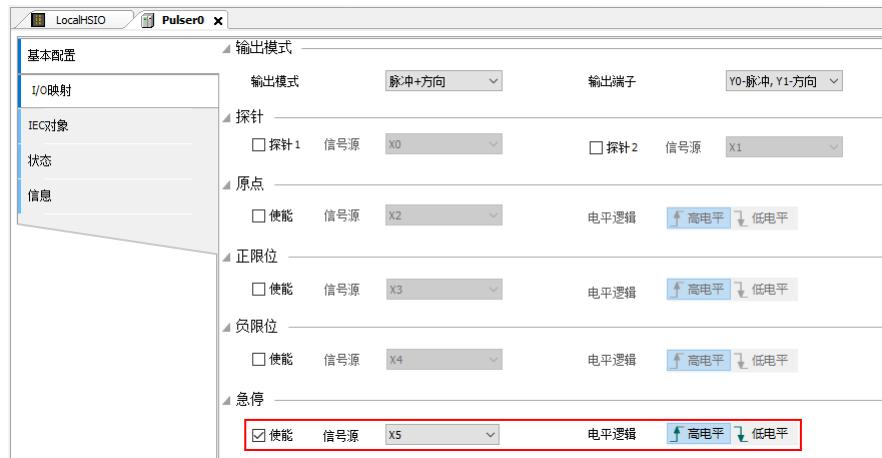
#### 6. 配置负限位功能。



- 勾选“使能”复选框。
- 根据需求配置信号源和电平逻辑。

配置完成后，通过MC\_Home功能块，实现脉冲轴的负限位和回原功能。回原方式支持CiA 402协议的17-30和35号回原方式。

#### 7. 配置急停功能。



- 勾选“使能”复选框。
- 根据需求配置信号源和电平逻辑。

配置完成后，通过外部端子的电平信号触发脉冲轴的急停功能。

#### 8. 配置脉冲轴基本设置和脉冲轴原点参数，具体请参见第179页“4.4.5 CiA402 轴”。

### 配置高速输入中断

外部中断频率不得高于1kHz，否则可能会出现中断丢失。

以X0中断为例进行介绍。

- 在“基本配置”界面勾选“X0”，使能X0中断输入，并单击X0后图标（上升沿有效、下降沿有效，上升沿+下降沿有效），配置X0沿中断。

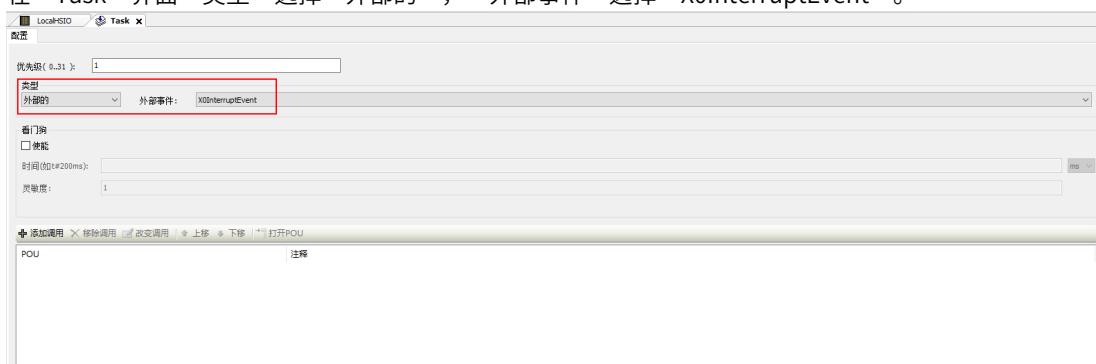


- 配置中断任务。

- 在左侧设备树中右键单击“任务配置”，在右键菜单中选择“添加对象>任务”。
- 在打开的对话框中单击“打开”。



c. 在“Task”界面“类型”选择“外部的”，“外部事件”选择“X0InterruptEvent”。

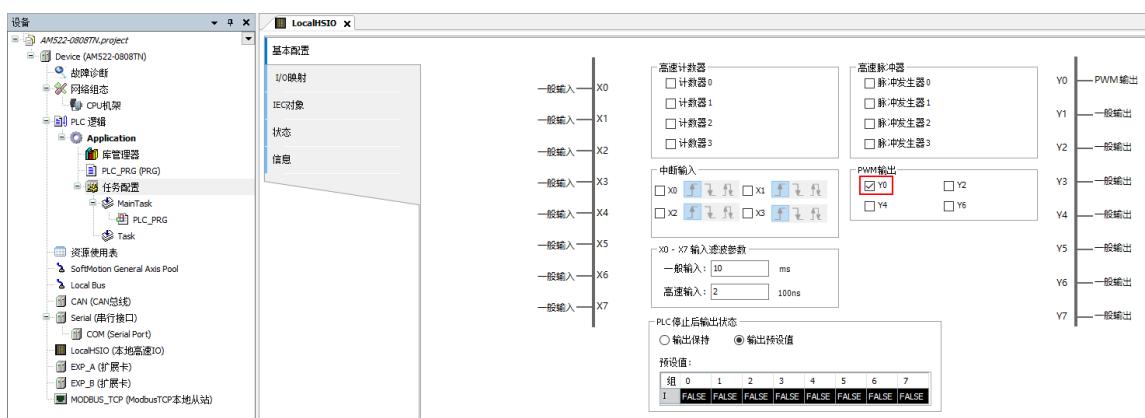


d. 主程序中调用外部中断功能块“HC\_EnableInterrupt”，触发中断任务中的程序执行，HC\_EnableInterrupt具体使用请参见《中型PLC指令手册》。

### 配置PWM输出功能

以Y0输出端子为例进行介绍。

在“基本配置”界面勾选“Y0”，使能Y0输出端子，在程序中调用HC\_PWM功能块，可以实现PWM波的输出。



AM300/AM500系列PLC和AC700系列PLC的PWM输出规格如下表所示。

类别	AM300/500系列	AC700系列
最大脉冲周期	10000000μs	3355443μs
最小脉冲周期	0.5μs	0.5μs

类别	AM300/500系列	AC700系列
最大脉宽	10000000μs	3355443μs
最小脉宽	0.2μs	0.2μs

## 说明

AM300/500系列PLC支持4路PWM输出（Y0/Y2/Y4/Y6），AC700系列PLC支持2路PWM输出（Y0/Y2）。

## 配置端子滤波参数

在“基本配置”界面根据实际所需配置一般输入和高速输入滤波参数。



对于未在高速计数器信号源中配置的端子，生效的是一般滤波参数；对于已在高速计数器信号源中配置的端子，生效的是高速滤波参数。

AM300/500系列PLC和AC700系列PLC滤波参数配置差异如下表所示。

差异项	AM300/500系列	AC700系列
一般输入值的限制值	2ms~1000ms	1ms~1000ms
高速输入值的单位	100ns	200ns
高速输入值的限制值	2~1000	1~1000

## 配置PLC停止后输出状态

配置PLC停止后，Y输出端子的电平状态，可设置为“输出保持”和“输出预设值”。

- 输出保持：PLC停止运行后，Y输出端子保持停止运行时的状态。
- 输出预设值：PLC停止运行后，Y输出端子保持用户预设的状态。

在“基本配置”界面“PLC停止后输出状态”区域根据实际所需设置为“输出保持”和“输出预设值”。





## 注意

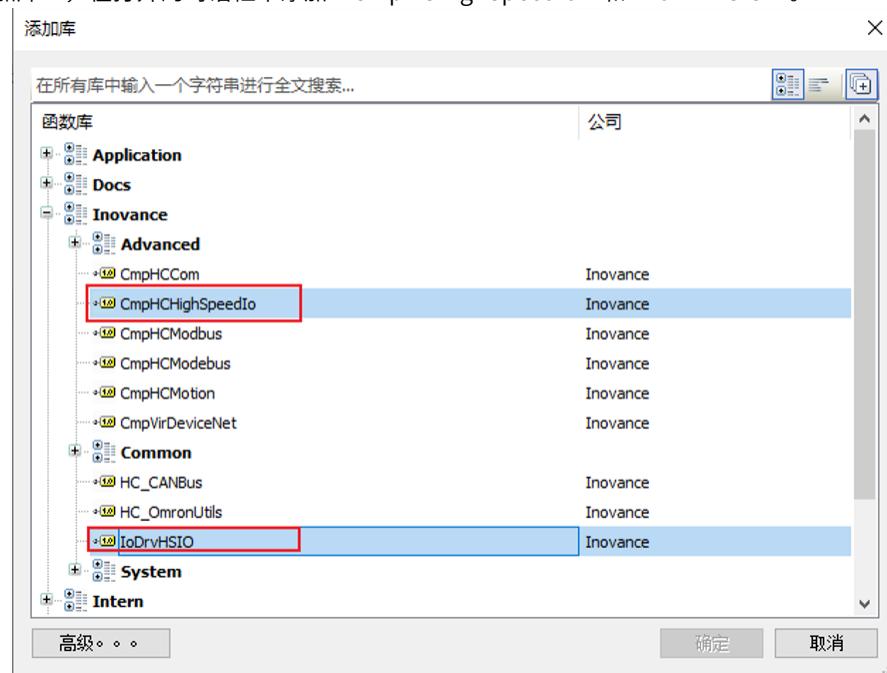
PLC停止后输出状态设置为输出保持时，需要在“PLC设置”界面中设置“停止时输出的方式”为“保持当前值”。

## 添加高速IO库

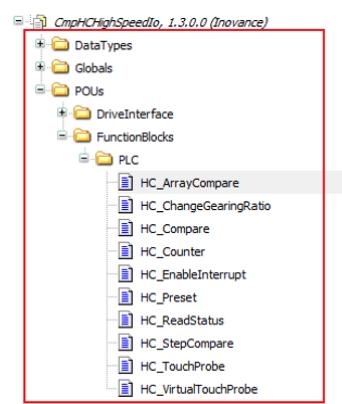
1. 在左侧设备树中双击“库管理器”，进入“库管理器”界面。



2. 单击“添加库”，在打开的对话框中添加“CmpHCHighSpeedIo”和“IoDrvHSIO”。



添加完成后，界面显示高速IO库的功能块。

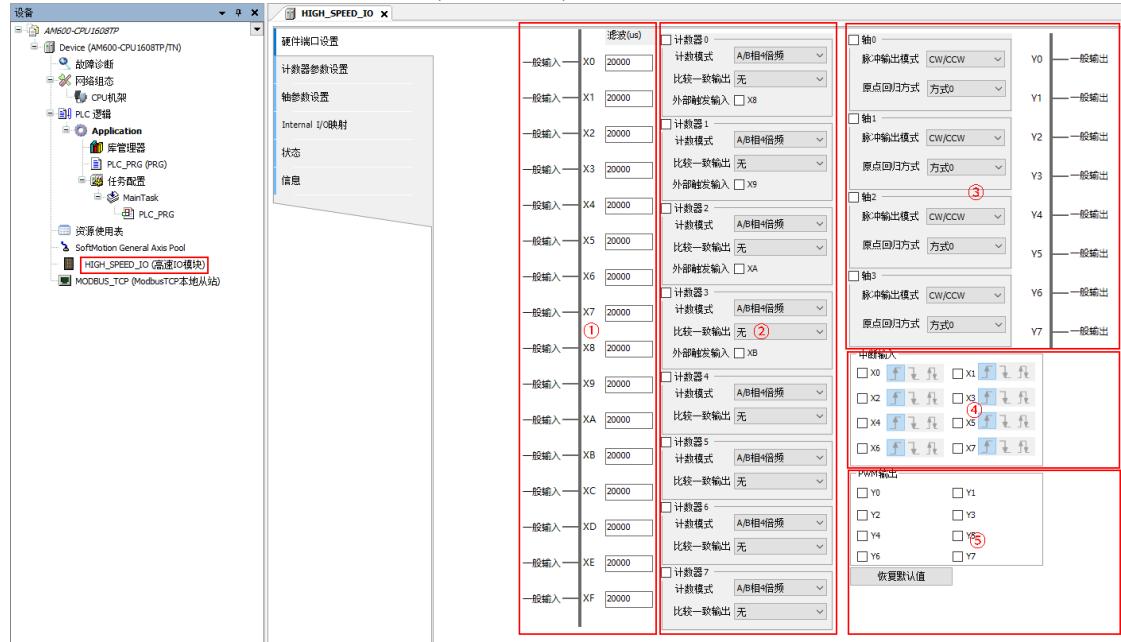




AM300/500系列PLC和AC700系列PLC是同一个高速IO库，但AC700系列PLC不支持脉冲输出部分和中断部分(HC\_EnableInterrupt)的功能块。

## AM400/AM600高速IO配置

1. AM400/AM600高速IO接线指导，具体请参见第496页“9.6 AM400/600高速I/O接线指导”。
2. 在左侧设备树中双击“HIGH\_SPEED\_IO(高速IO模块)”，进入“HIGH\_SPEED\_IO”配置界面。



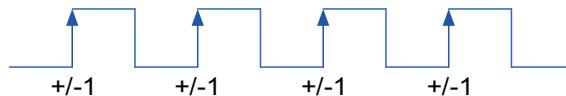
序号	功能配置
①	端子滤波参数配置
②	高速计数器功能配置
③	高速脉冲输出功能配置
④	高速输入沿中断功能配置
⑤	PWM输出功能配置

### 配置高速计数器功能

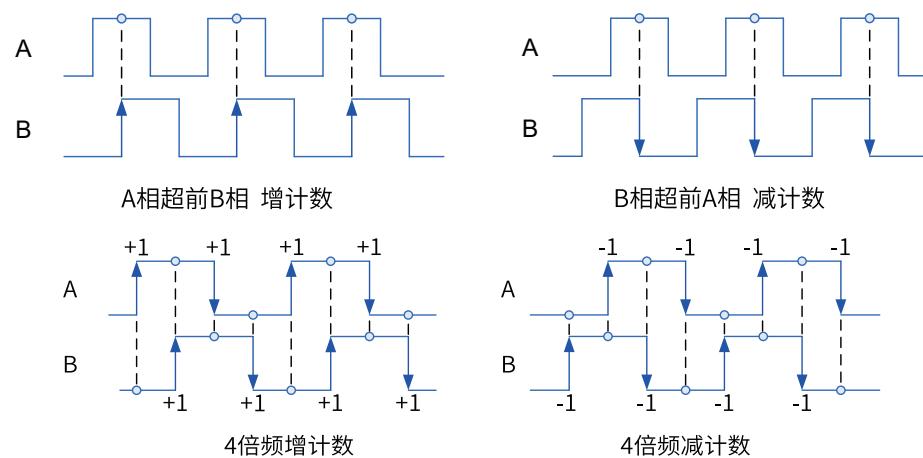


1. 高速计数功能的参数配置：包括计数模式选择、比较一致输出以及外部触发输入；其中计数模式包括单相计数，AB相计数，CW/CCW计数，内部时钟计数。以“计数器0”为例（计数器号支持0~7），配置步骤如下：

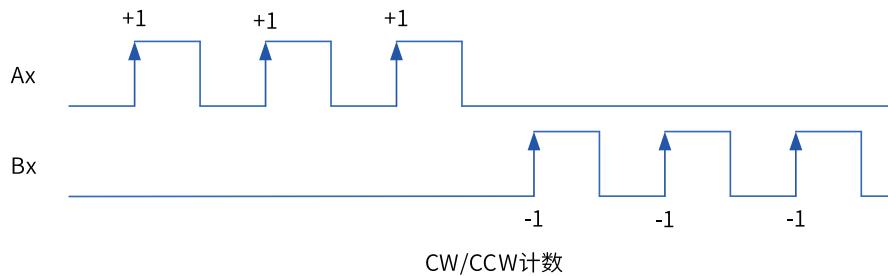
- 勾选使用的“计数器0”。
- 配置高速计数的模式：  
单相计数：接收外部单相编码器脉冲信号，只占用硬件X0端口。



AB相计数：接收外部AB相编码器脉冲信号，AB亦可配置成4倍频，占用硬件X0、X1端口。



CW/CCW计数：接收外部CW/CCW编码器脉冲信号，占用硬件X0、X1端口。



内部时钟：高速计数器0的脉冲信号来源于软件内部定时产生脉冲信号，支持1us/10us/100us/1ms。

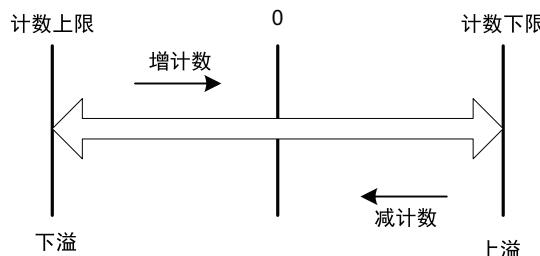
- 比较一致输出：当高速I/O的计数值达到设定值的时候，对应的硬件输出端口会输出相应的电平信号。使用该功能时候需调用中断使能功能块HC\_EnableInterrupt和计数值设定功能块HC\_SetCompare/HC\_SetCompareM。
- 外部触发输入：启用外部触发输入引脚可以实现高速计数器值的锁存和脉冲宽度测量的功能，计数锁存功能需要调用HC\_TouchProbe功能块；脉冲宽度测量功能需要调用HC\_MeasurePulseWidth功能块。
- 滤波参数：调整高速计数端口的滤波，默认2us。

## 2. 实例化高速计数器，数据类型COUNTER\_REF；以“计数器0”为例配置：

“计数器0”实例化默认名称：HS\_Counter0。

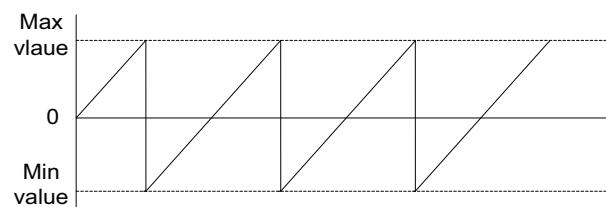
计数方式：

- 线性计数：线性计数器在最大值和最小值之间计数，当正向计数达到最大值或反向计数达到最小值时停止计数，溢出标志有效。

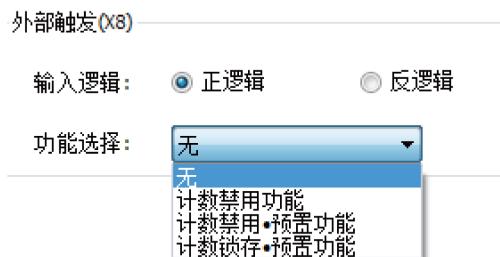


- 环形计数：需要配合“HC\_SetRing”使用。

环形计数器在最大值和最小值之间计数，当正向计数超过最大值之后跳转到最小值，当反向计数时小于最小值则跳转到最大值。

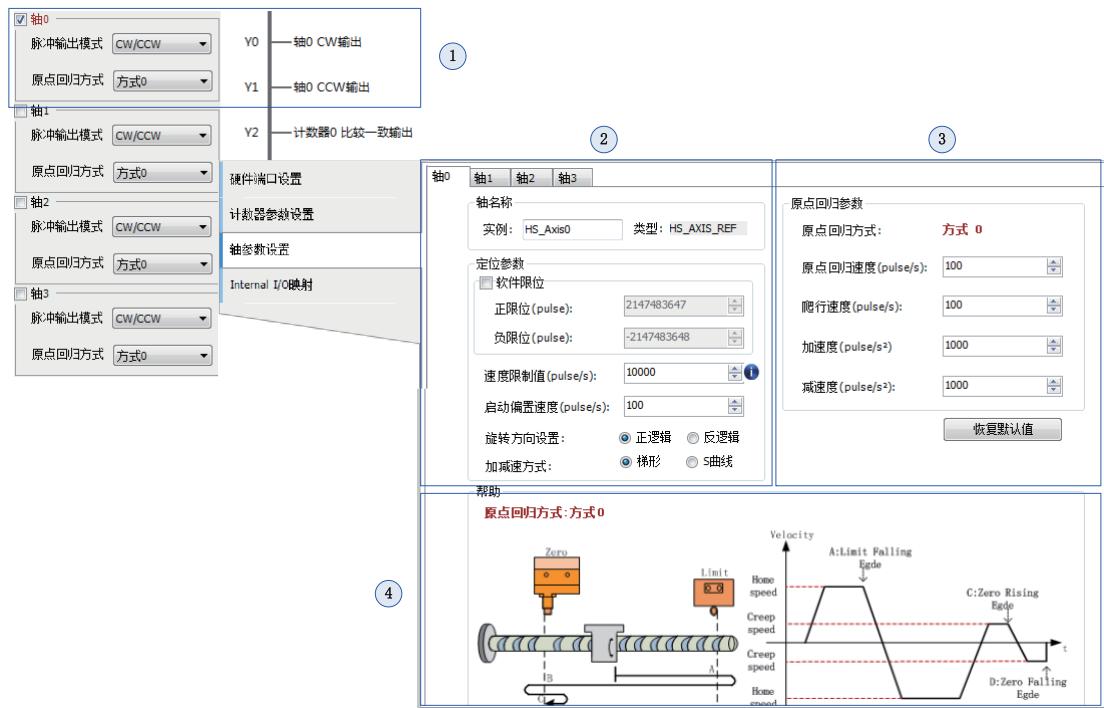


## 3. 选择外部触发功能。



当X8输入的电平有效时，可以对计数器的功能进行设定，例如X8可配置作为禁用“计数器0”计数的信号；禁用“计数器0”预制功能；计数锁存的功能。

### 配置高速脉冲输出功能

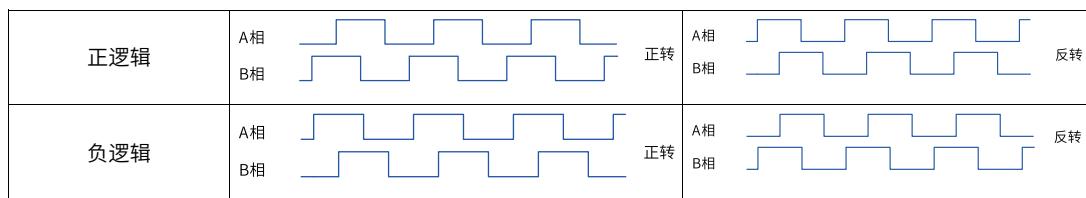


1. 配置高速脉冲输出功能，包括输出脉冲方式（脉冲+方向、CW/CCW、AB相）、原点回归方式。

以“轴0”为例配置（轴号支持0~3）：

- 勾选使用的“轴0”。
- 配置高速脉冲输出的工作方式：

脉冲指令形态	脉冲+方向	
	正转	反转
正逻辑	PULSE  SIGN 	PULSE  SIGN 
负逻辑	PULSE  SIGN 	PULSE  SIGN 
CW/CCW		
正逻辑	CW  CCW   	CW  CCW 
负逻辑	CW  CCW   	CW  CCW 
脉冲指令形态	A/B相	
	正转	反转



- 回零方式

支持0~3，详见回零示意图。

2. 轴的实例化，数据类型HS\_AXIS\_REF。以“轴0”为例配置：

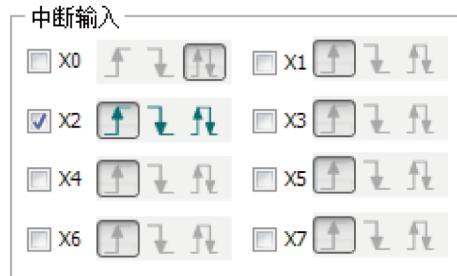
- “轴0”实例化默认名称：HS\_Axis0。
- 正负限位：软限位。
- 速度限制：最大速度限制。
- 启动偏置速度：脉冲启动时基底速度。
- 加速方式：梯形加速和S曲线加速。

3. 原点回归参数配置，包括回零速度、爬行速度等。

需要配合“MC\_Home\_P”功能块使用。

4. 不同的原点回归方式的示意图。

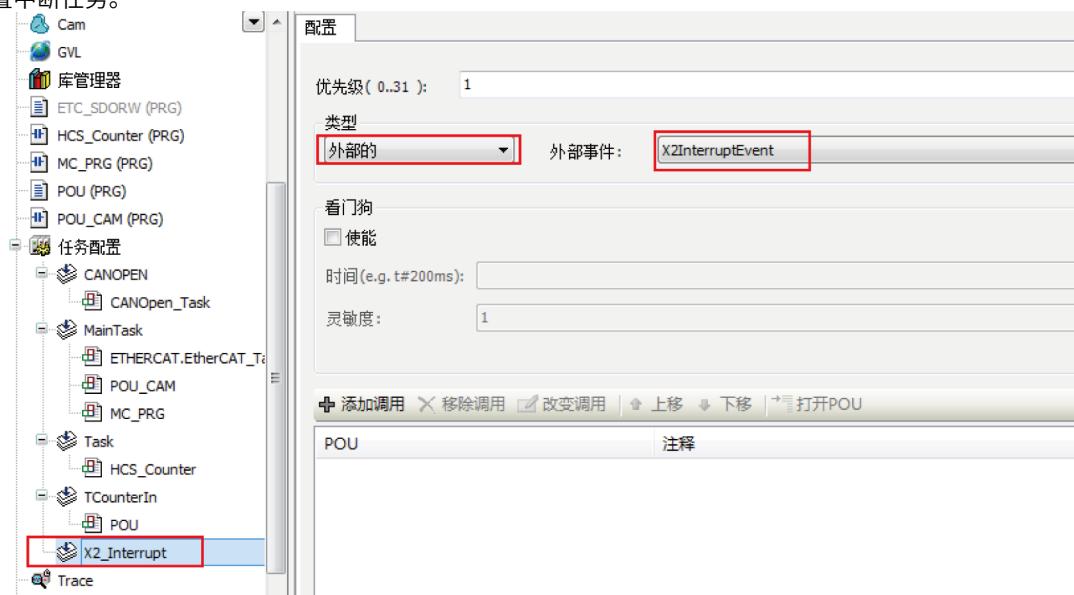
#### 配置高速输入中断



1. 勾选X2高速输入中断。

选择配置X2的沿中断：上升沿有效/下降沿有效/上升沿和下降沿都有效。

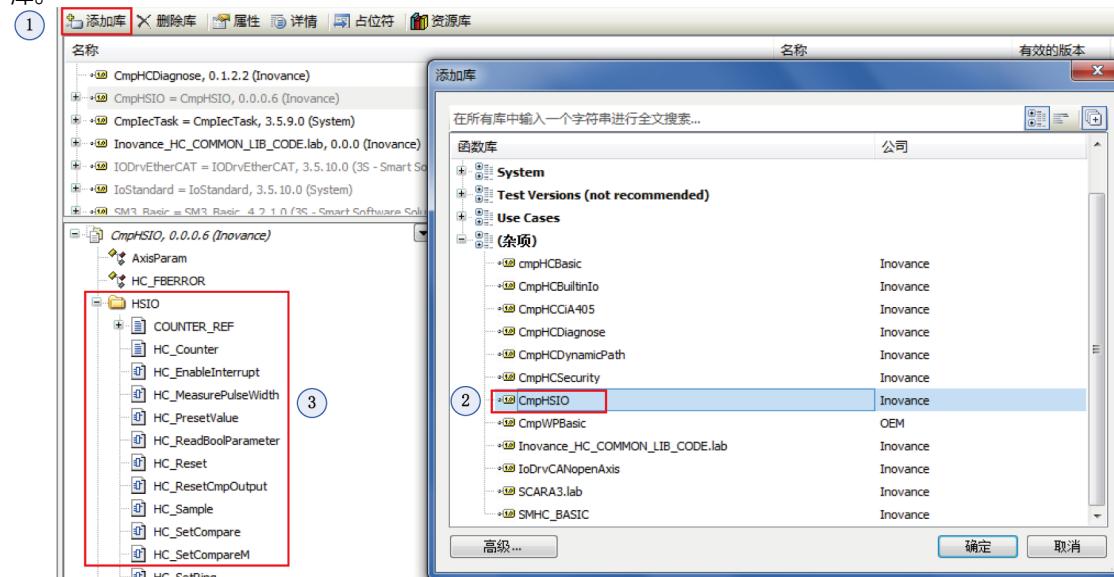
2. 配置中断任务。



打开外部中断功能块“HC\_EnableInterrupt”，增加X2的中断任务，当X2上升沿信号有效时，触发中断任务中的程序执行。

### 添加高速IO库

- 在左侧设备树中双击“库管理器”，打开“库管理器”界面。
- 单击“添加库”，在打开的对话框中展开“杂项”，选择“CmpHSIO”，单击“确定”，添加新高速IO库。



- 在库列表中单击新高速IO库，展开“HSIO”，显示新高速IO库的功能块。

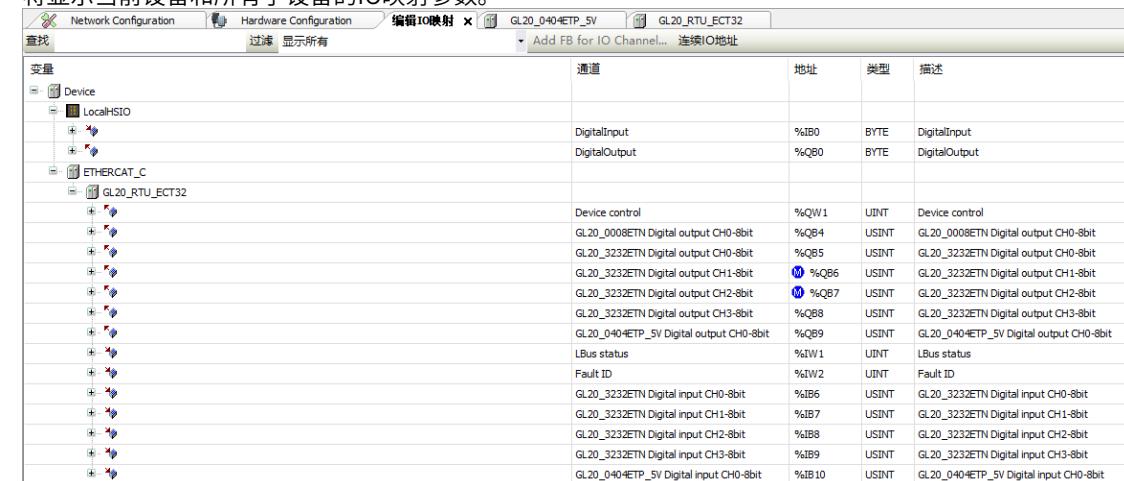
### 高速IO诊断

高速IO诊断，具体请参见第502页“9.7.2 高速IO诊断”。

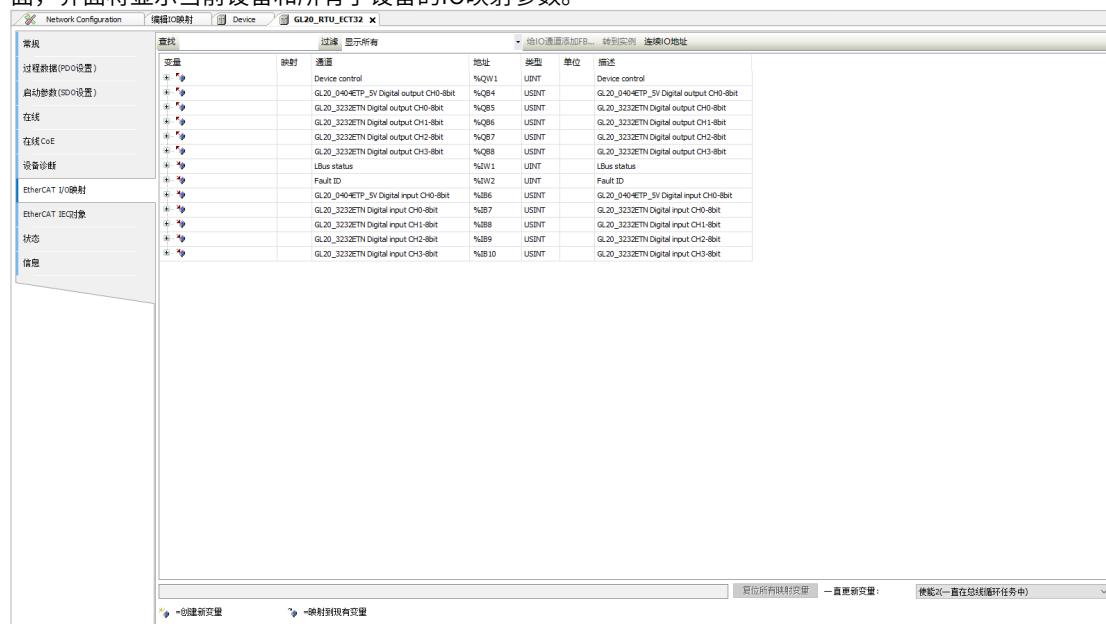
## 4.2.6 IO映射参数说明

进入IO映射配置界面分为以下两种方式：

- 在左侧设备树中右键任意一个设备，在右键菜单中选择“编辑IO映射”，打开“编辑IO映射”界面，界面将显示当前设备和所有子设备的IO映射参数。



- 在左侧设备树中双击通信接口模块，在打开的界面中单击“xx I/O映射”页签，打开“xx I/O映射”界面，界面将显示当前设备和所有子设备的I/O映射参数。



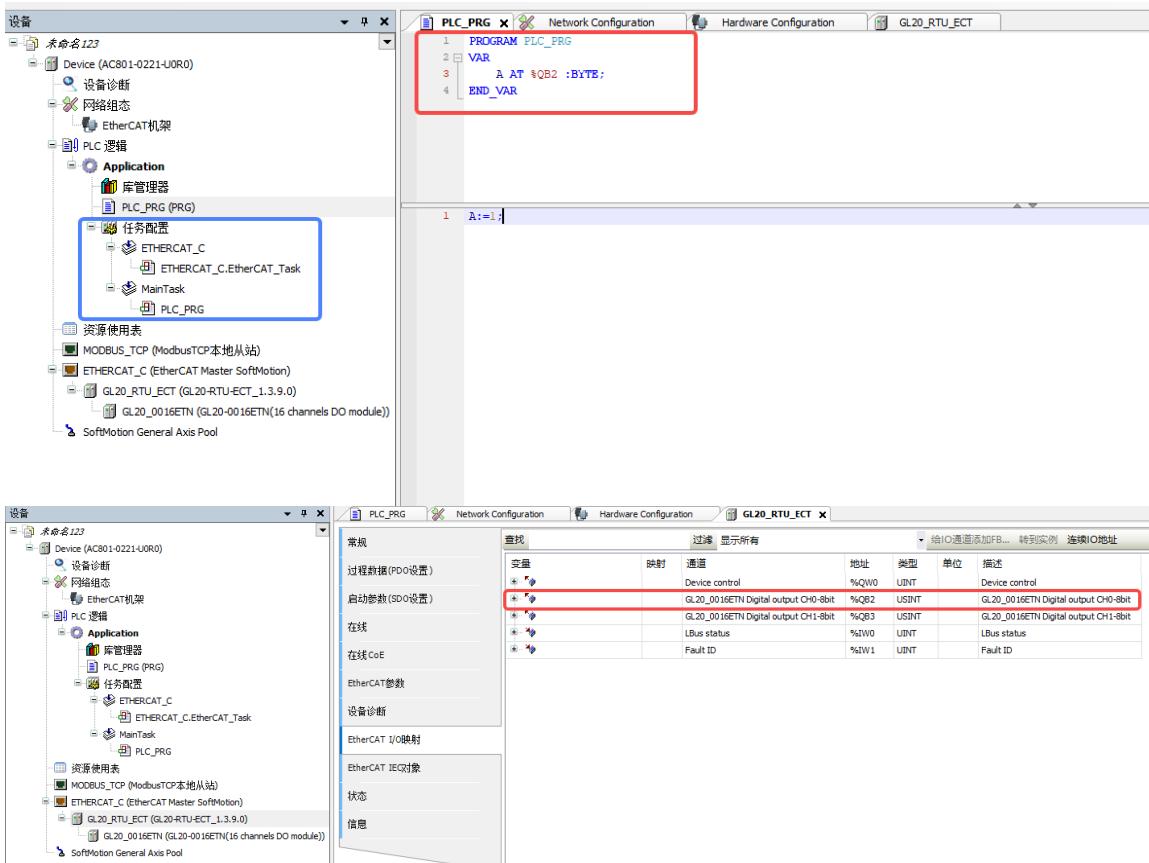
界面中各参数说明如下表所示。

参数名称	参数说明
查找	根据关键字搜索查找相应的I/O映射通道。
过滤	根据过滤规则显示相应的变量。
连续IO地址	将设备的I/O映射地址重新分配为连续地址（只在时序地址分配模式下生效）。
变量	<ul style="list-style-type: none"> <li>I/O映射通道映射到现有变量：双击“变量”列单元格，再单击“...”，在打开的“输入助手”界面中选择待映射的现有变量，单击“确定”。如果变量不存在，则在编译时会报错。</li> <li>I/O映射通道映射到新变量：双击“变量”列单元格，在文本框中输入新建变量的名称，按“Enter”键。新建变量的名称需要满足IEC编程规范，否则在编译时会报错。</li> </ul>
映射	<p>单击图标可切换I/O映射通道映射模式，分为创建新变量模式和映射到现有变量模式。</p> <ul style="list-style-type: none"> <li>图标表示创建新变量模式。</li> <li>图标表示映射到现有变量模式。</li> </ul>
通道	当前I/O映射通道的名称。
地址	<p>当前I/O映射变量被分配的地址。如果需要自定义I/O地址，可通过双击“地址”列单元格进行修改。</p> <p>地址分配方式分为：时序地址分配和顺序地址分配，默认的地址分配方式为时序地址分配。</p> <ul style="list-style-type: none"> <li>时序地址分配：按照设备添加的时间先后顺序分配I/O映射变量的地址。</li> <li>顺序地址分配：按照设备在设备树中的位置分配I/O映射变量的地址。</li> </ul> <p>设置地址分配方式：在左侧设备树中双击PLC设备[例如“Device(AC702)”]，在打开的界面中单击“PLC设置”，设置“其他设置”区域中“顺序分配I/O地址”参数。</p> <ul style="list-style-type: none"> <li>勾选：地址分配方式为顺序地址分配。</li> <li>去勾选：地址分配方式为时序地址分配。</li> </ul>
类型	当前I/O映射变量的数据类型。
单位	当前I/O映射变量的单位。
描述	当前I/O映射变量的描述信息。
默认值	当前I/O映射变量的默认值。 <b>说明：</b> 有条件显示此参数，当在“PLC设置”界面中“停止时输出的方式”设置为“将所有输出设置为默认值”时显示。
当前值	当前I/O映射变量在程序实际运行过程中的实际值。 <b>说明：</b> 当工程进入在线状态时界面才显示该参数。

参数名称	参数说明
预备值	当前IO映射变量预备要写入的值，可通过双击“预备值”列单元格进行预备值编辑，编辑完成后可以通过组合键“Ctrl+F7”或者菜单命令将预备值写入IO映射变量。 <b>说明：</b> 当工程进入在线状态时界面才显示该参数。
复位所有映射变量	将当前IO映射变量表格中的所有映射变量置为空值。 <b>说明：</b> 此操作只会复位映射变量值，不会复位映射变量的默认值和地址等信息。
一直更新变量	设置IO映射变量任务执行模式。 <sup>[1]</sup> <ul style="list-style-type: none"> <li>使用父设备设置：系统运行时不进行IO映射循环。</li> <li>使能一（如果未在任何任务中使用则使用总线循环任务）：优先选择其他已使用当前设备地址的任务，如果没有，则使用当前设备所属的总线任务。</li> <li>使能二（一直在总线循环任务中）：使用当前设备所属的总线任务。</li> </ul>

### [1]: IO映射变量任务执行模式示例

“%QB2” 地址被使用在 “PLC\_PRG” 该POU中，同时 “%QB2” 地址被分配给GL20通信接口模块下面的模块上，GL20通信接口模块为EtherCat从站，如下图所示。



在任务配置中有两个任务：ETHERCAT\_C和MainTask，ETHERCAT\_C任务执行ETHERCAT总线循环，MainTask主任务执行PLC\_PRG。

- 当通信接口模块的“一直更新变量”参数配置为“使用父设备设置”时，此IO配置将不会生成数据刷新的POU，IO数据在工程运行时将不会变化。
- 当通信接口模块的“一直更新变量”参数配置为“使能一（如果未在任何任务中使用则使用总线循环任务）”时，此IO配置会正常生成数据刷新的POU，并且此POU在MainTask任务中执行。
- 当通信接口模块的“一直更新变量”参数配置为“使能二（一直在总线循环任务中）”时，此IO配置会正常生成数据刷新的POU，并且此POU在EtherCat\_C任务中执行。

- 如果地址没有在PLC\_PRG中直接使用，则不论是配置为“使能一”还是“使能二”，此IO配置生成的数据刷新POU都会在EtherCat\_C任务中执行。
- IO配置在哪个任务中执行并不是和任务名称强相关，而是和当前设备所属的总线循环所依赖的任务有关。

#### 相关说明

- 交叉引用：IO映射通道关联了变量后也可通过右键单击变量单元格，选择“浏览 > 浏览交叉引用”查看当前变量在程序中的交叉引用信息。
- 地址分配：当地址分配给设备后，该地址资源属于当前被分配的设备，无论这个地址是否映射变量，所以尽量不要在POU中使用被分配给设备的地址。如果需要使用设备的地址，尽量通过映射变量的方式去实现功能。
- 模式切换：从顺序地址模式切换到时序地址模式时不会改变设备的地址分配，而从时序地址模式切换到顺序地址模式时会按照设备在设备树中的位置重新分配地址。
- 顺序模式：在顺序地址模式下，手动地址的优先级高于自动地址，当配置的手动地址与自动地址冲突时会优先将自动地址重新进行分配。
- 时序模式：在时序地址模式下，手动地址的优先级与自动地址相同，当配置的手动地址与自动地址冲突时，会提示有冲突并且不允许分配。
- 单个IO映射通道（Device Control）被分配到某指定地址，IO映射通道的地址可以手动分配。Bit0～Bit15是IO映射通道的子元素，子元素的地址不允许手动分配，子元素的地址由IO映射通道的地址决定。

## 4.3 扩展卡配置



注意

仅AM300/AM500系列PLC支持扩展卡配置，其他系列PLC可跳过本节。

AM300/AM500系列PLC扩展卡配置操作，具体请参见各扩展卡对应的《用户手册》。

## 4.4 EtherCAT 配置

### 4.4.1 概述

EtherCAT是一种基于以太网的开放式工业现场技术，具有通信刷新周期短、同步抖动小、硬件成本低等特点，支持线型、树型、星型以及混合网络拓扑结构。EtherCAT从站必须使用专用的通信控制芯片（ESC），EtherCAT主站可以使用标准的以太网控制器。

如需了解EtherCAT原理和相关技术，请参考书籍《工业以太网现场总线EtherCAT驱动程序设计和应用》，或者登录EtherCAT技术委员会官网，网址：<https://www.EtherCAT.org.cn>。

## 4.4.2 常用功能

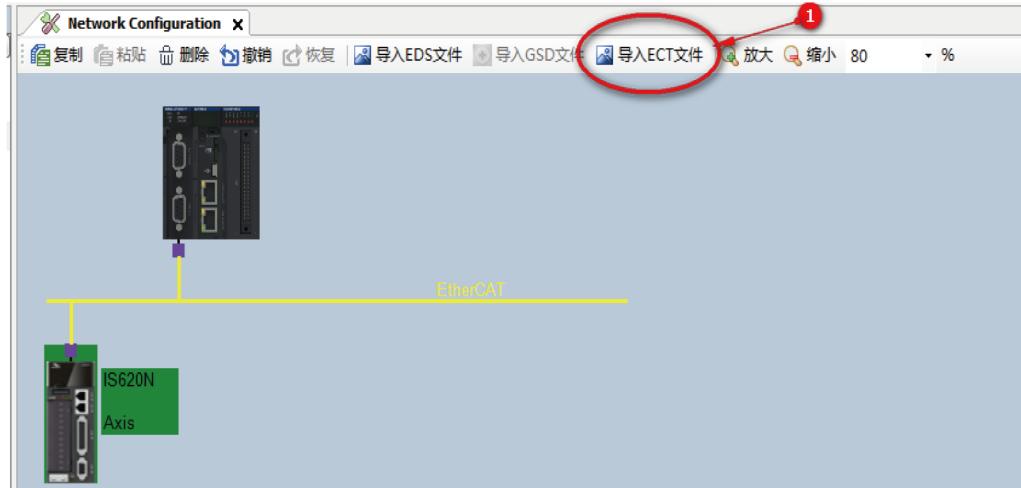
### 设备安装

EtherCAT设备安装是指将符合ETG (EtherCAT Technology Group, EtherCAT技术协会) 标准的设备描述XML文件导入至编程软件InoProShop，经过软件的文件解析处理后生成可供用户添加、删除等操作的EtherCAT组态设备。编程软件InoProShop内部集成了汇川所有EtherCAT从站设备，无需单独安装。如需使用第三方EtherCAT设备，必须先安装第三方厂商提供的设备描述文件。

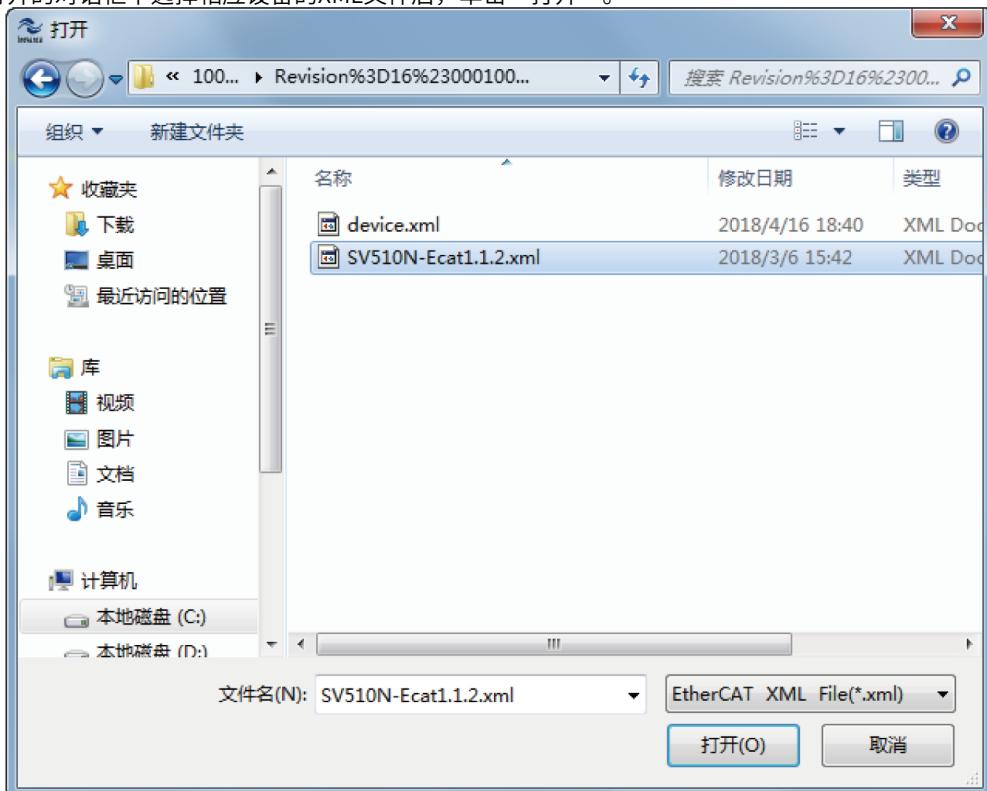
安装方法有两种，分别在网络组态界面安装和通过菜单栏安装，具体操作步骤如下。

- 网络组态界面安装

1. 在“网络组态”界面单击“导入ECT文件”。



2. 在打开的对话框中选择相应设备的XML文件后，单击“打开”。

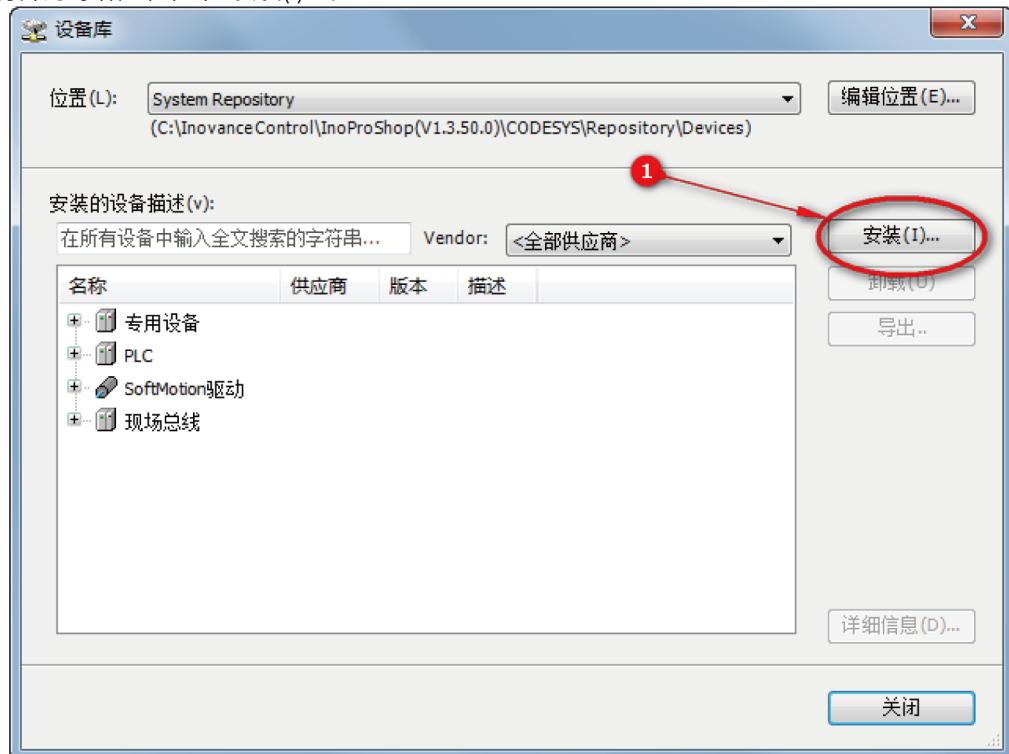


- 通过菜单栏安装

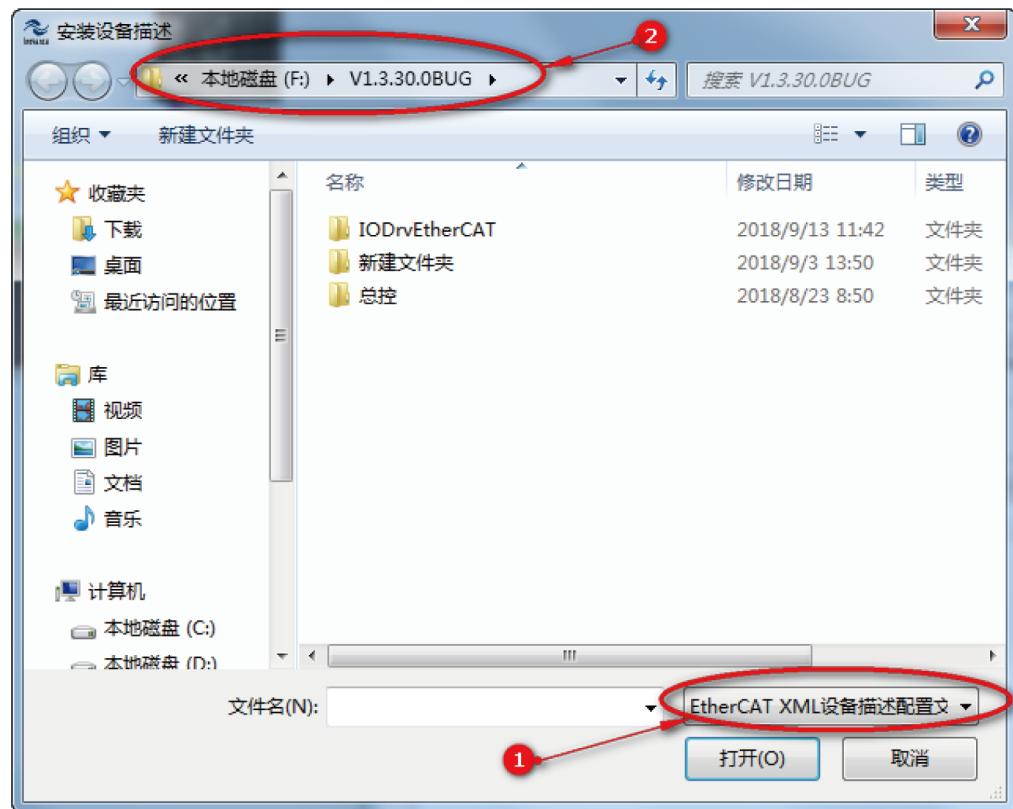
1. 在菜单栏中选择“工具 > 设备库”。



2. 在打开的对话框中单击“安装(I)”。



3. 在打开的对话框中选择“EtherCAT XML设备描述配置文件”项，再选择存放在本地PC机的从站设备描述文件，单击“打开”。

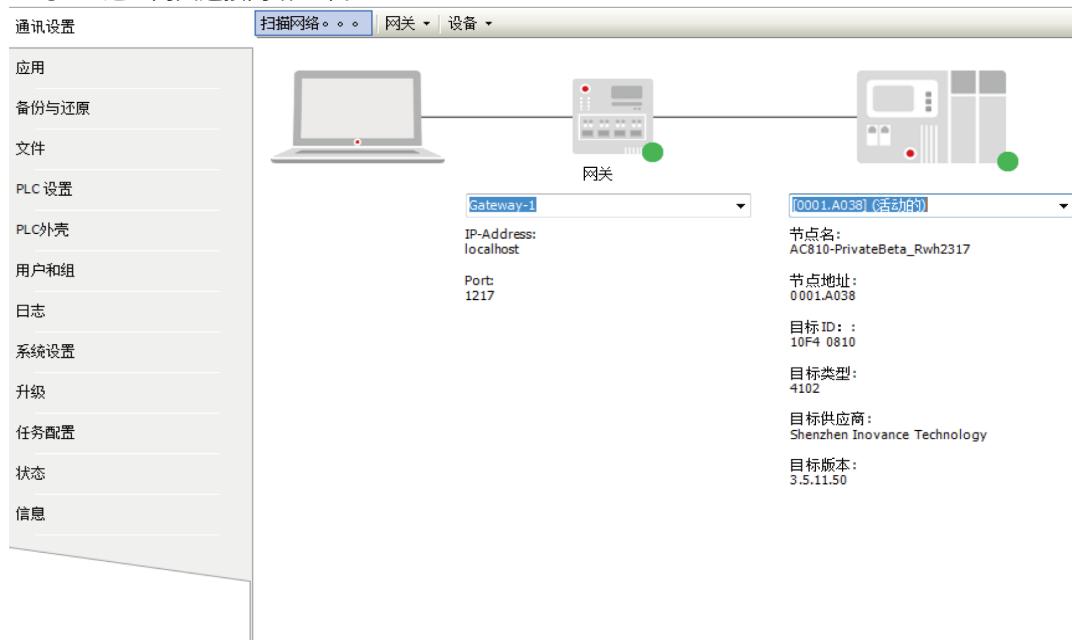


## 扫描设备

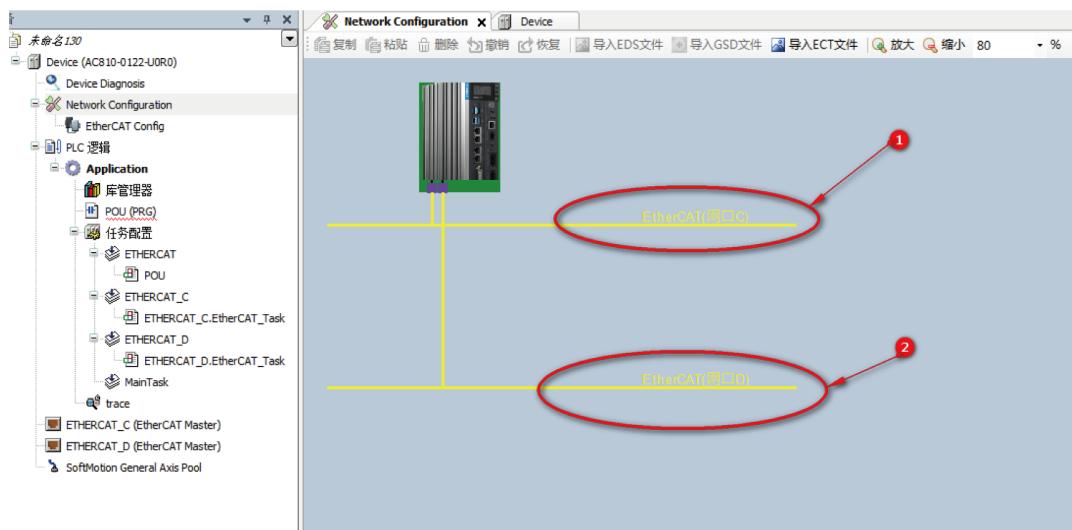
推荐使用扫描功能，按照【热复位】->【退出登录】->【扫描设备】流程操作。

### ● 前提条件

- PC与PLC通过网关连接网络正常。



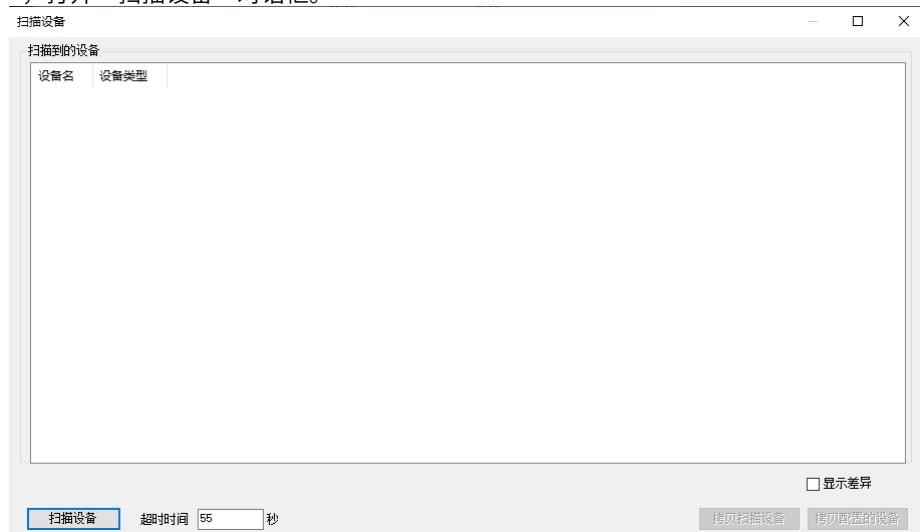
- PLC与EtherCAT从站连接网络正常。
  - 编程软件组态端口配置信息与PLC实际物理连接端口一致。
- 为保持一致，建议使用扫描操作前先下载一次端口配置信息。



- PLC处于停止状态。

#### ● 操作步骤

1. 在左侧设备树中右键单击“EtherCAT\_x (EtherCAT Master SoftMotion)”，在右键菜单中选择“扫描设备”，打开“扫描设备”对话框。



#### 说明

超时时间：表示执行一次扫描操作允许最大的超时时间，扫描不到结果时，可以适当延长扫描超时时间，最小值为20秒。

2. 单击“扫描设备”，扫描成功后设备将显示在扫描设备列表中。



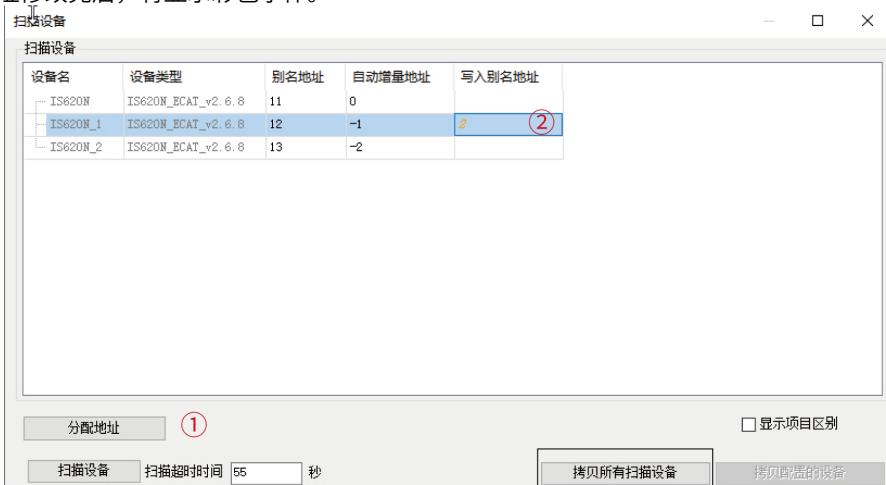
3. 单击“拷贝扫描设备”，添加设备至设备树和组态中。

#### ● 相关操作

##### ■ 分配别名地址

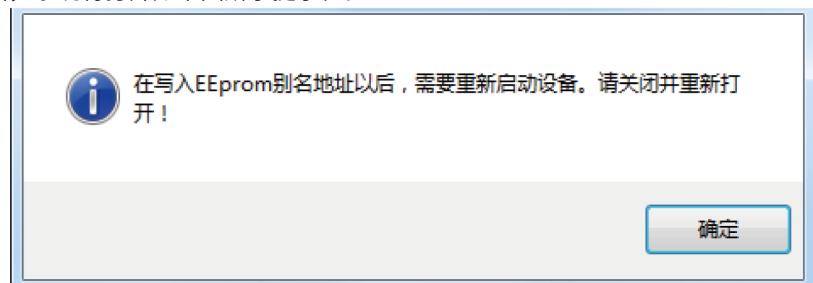
1. 在“写入别名地址”列双击待修改别名地址的文本框，输入新的别名地址。

别名地址修改完后，将显示彩色字体。



2. 单击“分配地址”，使新的别名地址生效。

分配成功后，系统将打开如下图所示提示框。



##### ■ 显示设备差异

勾选“显示差异”，显示现有组态设备与扫描设备之间的差异，如下图所示。



- 当现有组态设备无别名地址时，系统不对比别名地址，只对比设备类型。如果设备类型完全匹配，那么设备列表的字体为绿色。
- 当现有组态设备有别名地址时，系统将对比设备类型和设备别名地址。当二者完全一致时，设备才判定为完全匹配，否则判定为不匹配，不匹配时设备列表的字体为红色。

上图中每个功能名称及其操作如下表所示。

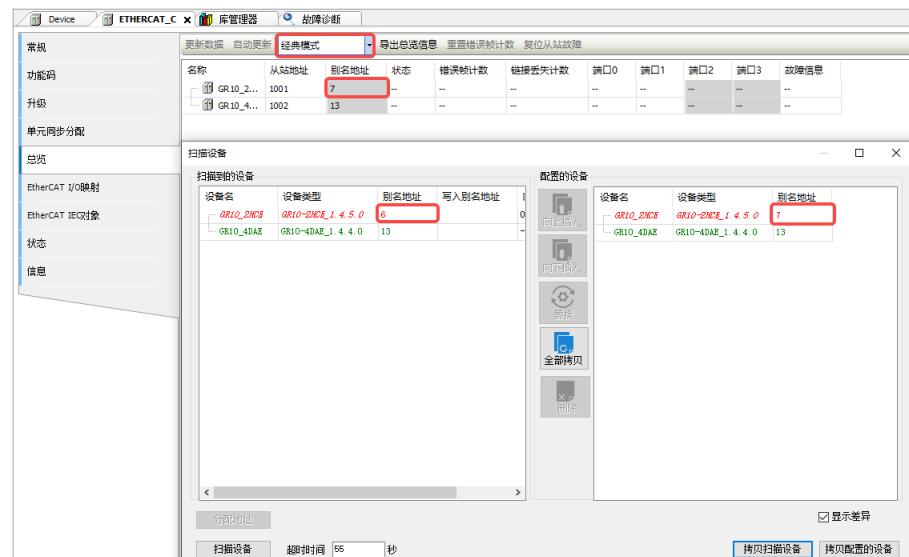
编号	功能名称	功能操作
①	向上插入	选择左侧的扫描设备和右侧现有设备后，单击 ，则扫描设备会插入到右侧选择设备之前（模块之间也可以使用此命令）
②	向下插入	选择左侧的扫描设备和右侧现有设备后，单击 ，则扫描设备会插入到右侧选择设备之后。（模块之间也可以使用此命令）
③	替换	选择左侧的扫描设备和右侧现有设备后（左侧设备必须和右侧设备为同一类设备），单击 ，则右侧设备会替换为扫描的设备。
④	全部拷贝	单击 ，右侧设备清空，左侧扫描设备全部复制到右侧。
⑤	删除	选择右侧设备，单击 ，删除单个设备。（模块也可以使用此命令）

#### ■ 拷贝扫描设备时别名地址是否生效

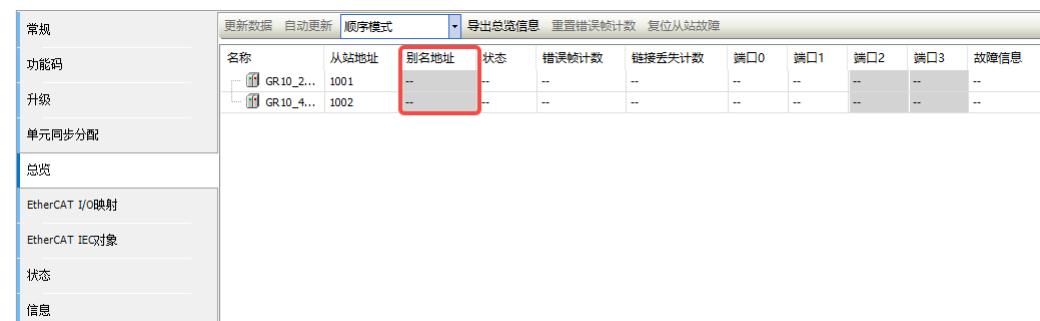
单击“拷贝扫描设备”时，系统会根据EtherCAT总览界面配置的从站通信模式设置扫描设备的别名地址信息。

- 当EtherCAT总览界面配置的从站通信模式为“经典模式”或“顺序模式”时，扫描到的设备与工程组态生成的设备默认为“顺序模式”，扫描到的设备别名地址会保存到设备数据中，但别名地址不生效。

例如，下图中EtherCAT总览界面中的从站通信模式为“经典模式”，扫描到的设备和配置的设备的设备类型一致，但别名地址不一致。



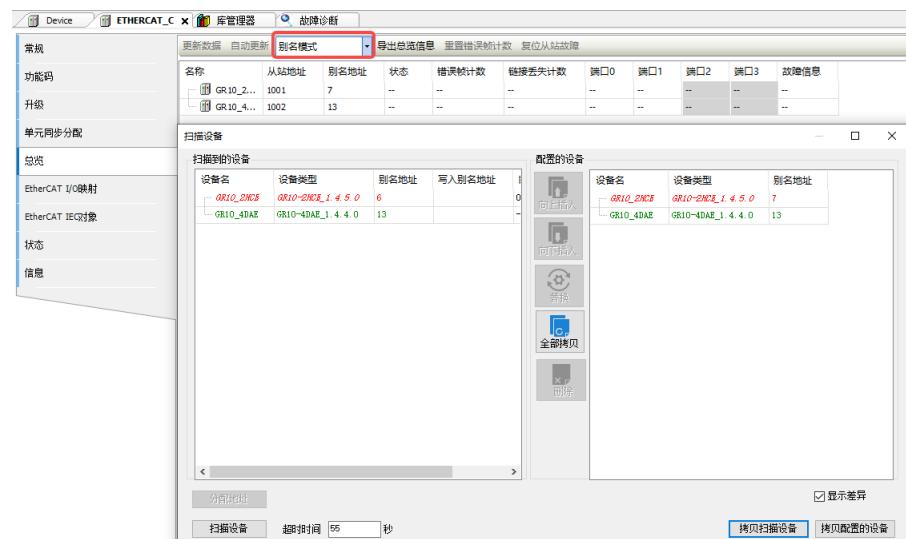
单击“拷贝扫描设备”，系统会根据扫描到的设备类型新建设备对象，同时保存别名地址，但该设备的通信模式仍然为“顺序模式”，别名地址并未真正生效，如下图所示。



该设备的通信模式切换为“别名模式”时，设备的别名地址为扫描到的别名地址，如下图所示。



- 当EtherCAT总览界面配置的从站通信模式为“别名模式”时，扫描到的设备与工程组态生成的设备默认为“别名模式”，扫描到的设备别名地址会保存到设备数据中，别名地址将立即生效。例如，下图中EtherCAT总览界面中的从站通信模式为“别名模式”，扫描到的设备和配置的设备的设备类型一致，但别名地址不一致。



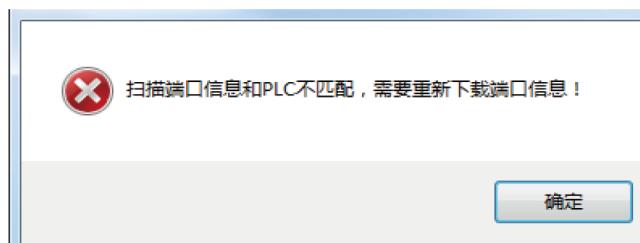
单击“拷贝扫描设备”，系统会根据扫描到的设备类型新建设备对象，该设备的通信模式为“别名模式”，生成的新设备对象的别名地址为扫描到的别名地址，如下图所示。



### ● 异常说明

编程软件组态端口信息和PLC实际物理连接的端口信息不一致，将打开如下图所示提示框。

此时需要重新下载编程软件端口信息。



例如编程软件组态端口配置为“EtherCAT\_C”，但实际MAC地址显示为“EtherCAT\_D”，可能引起扫描C口设备时，扫描结果为D口的设备。

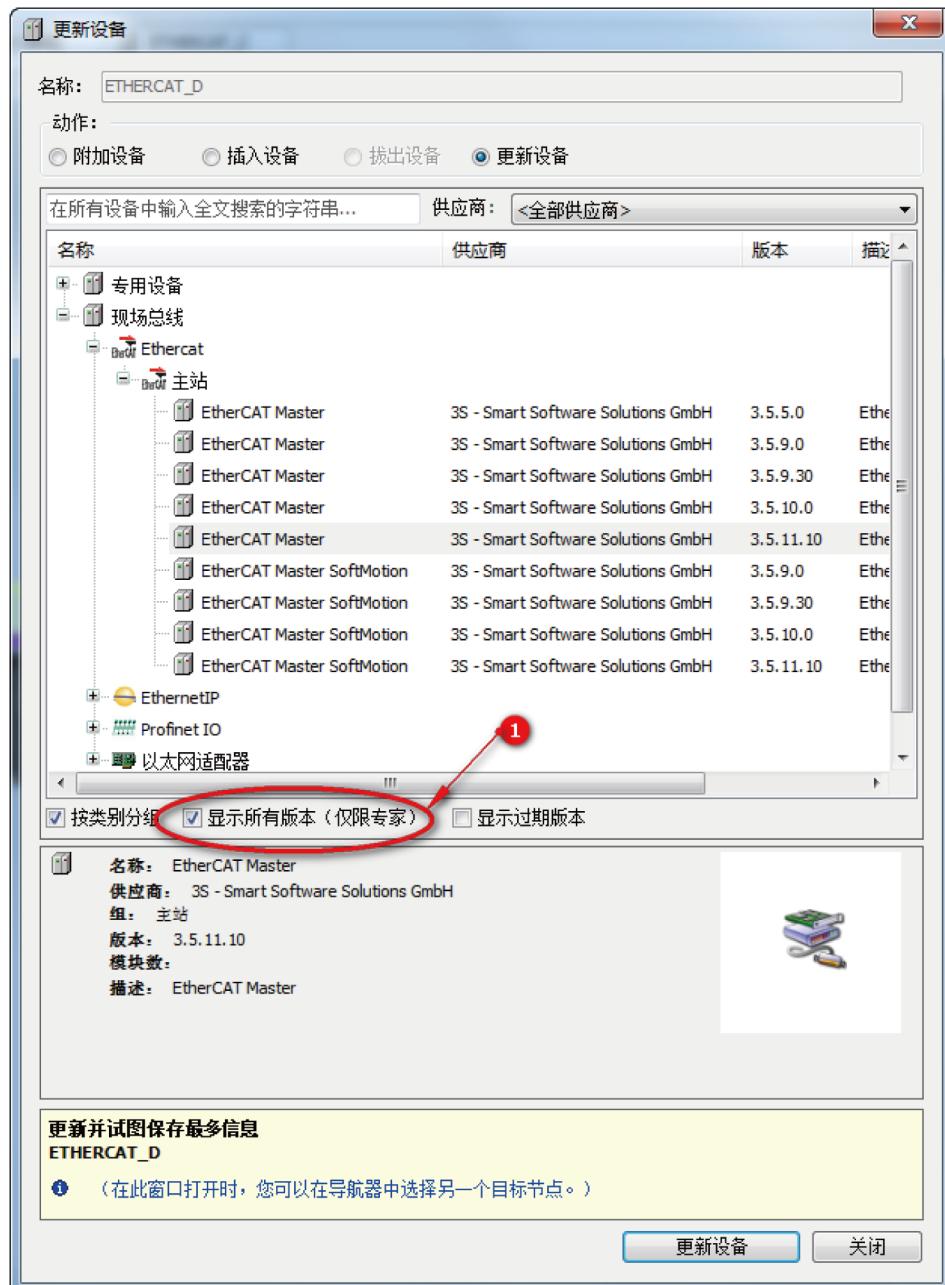


EtherCAT总线正常启动后，若用户退出登录，在线执行“扫描设备”时扫描结果可能会失败或者不准确。原因是“扫描设备”过程中以SDO通讯方式读取从站信息（例如对象字典0xF050）可能超时，导致影响扫描结果(SDO通讯属于异步通信，CPU负载、总线循环周期、用户程序执行时间都会影响SDO通讯)。

## 更新设备

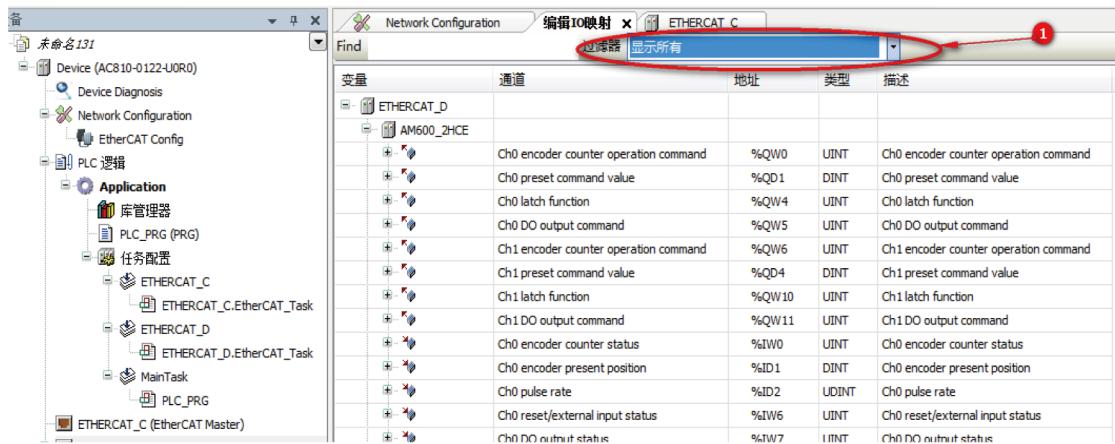
如果主站的版本不匹配或者需要升级，可以使用更新设备命令。

操作步骤：右键单击EtherCAT主站，在弹出的选项栏中选择“更新设备”。选中后会弹出如下对话框，勾选“显示所有版本（仅限专家）”可显示所有的版本，选择对应的版本后单击“更新设备”：



## 编辑IO映射

选择“编辑IO映射”命令会弹出如下界面。



在“过滤器”中选择合适的过滤器，可以过滤掉不需要的显示选择项。

## 总线任务

PLC所有IEC总线任务严格按照相同的逻辑顺序循环扫描执行，逻辑顺序可分为四个步骤：刷新输入（1）、执行IEC任务（2）、刷新输出（3）、执行总线循环（4），如下图所示：

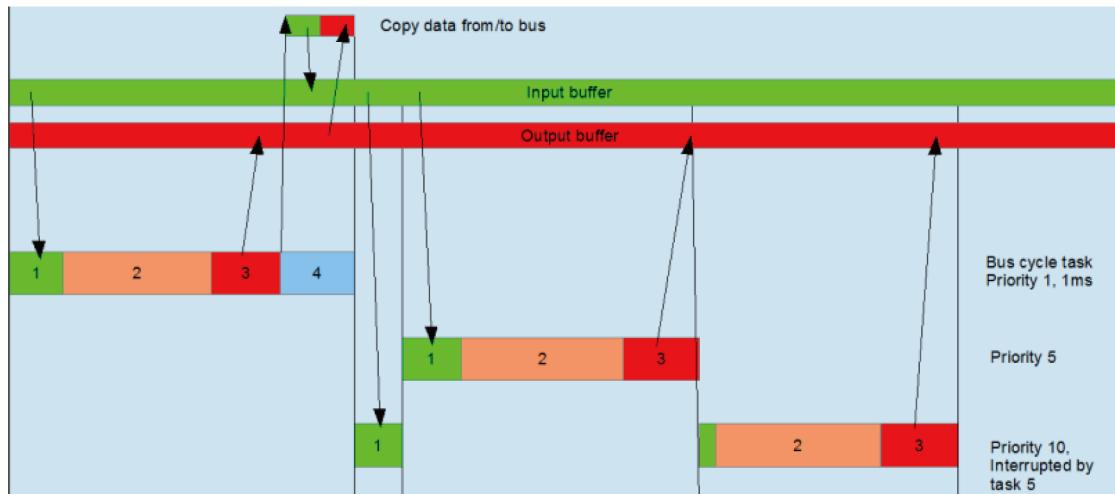


图4-11 总线循环任务

各个步骤的详细说明如下表所示：

编号	步骤	图例	说明
1	刷新输入	绿	IEC任务开始之前，从总线输入缓冲区读取数据，并将数据拷贝到与任务相关的输入变量中。
2	执行IEC任务	橙	扫描执行总线任务下的POU。
3	刷新输出	红	IEC任务结束之前，将任务中与总线相关的输出变量拷贝到总线输出数据缓冲区。
4	总线循环	蓝	执行总线通讯程序，主要由底层I/O驱动实现，包含两个功能：将总线输出缓冲区数据传输到远端从站的数据接收缓冲区，以及将远端从站发送的数据读取到总线输入缓冲区。

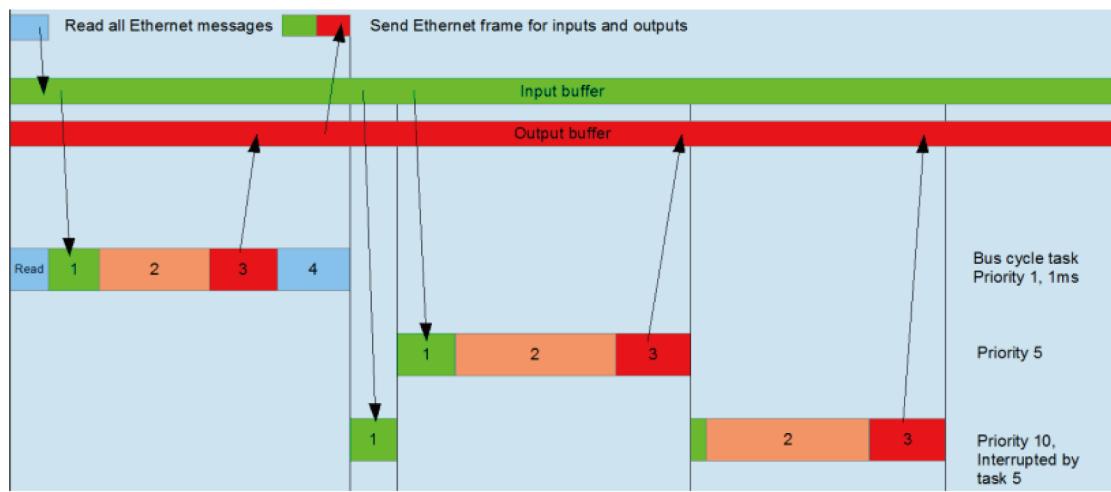


## 警 告

- 如果一个输出变量被多个任务使用，变量的值为不确定（此任务输出变量可能被其他任务修改、覆盖）；
- 当一个任务被高优先级的任务打断时，高优先级任务从输入缓冲区读取数据，并将数据同步到当前任务中的输入变量，可能会造成同一个扫描周期内输入变量值不一致的问题。可以通过任务开始的时候复制输入变量值、任务调用复制的输入变量的方式进行避免。

**EtherCAT 特殊的总线循环动作**

上一个周期的总线数据会在IEC输入之前被拷贝。



EtherCAT 主站设置 中的选项 “激活每个任务上的消息” 可以被激活。这种情况下每个任务的附加信息将会被发送到设备。这样总线通讯即可在多个任务下执行，并且这样还能够减少总线上的负载。

使用“激活每个任务消息”选项后的 EtherCAT 总线周期表格如下图所示：

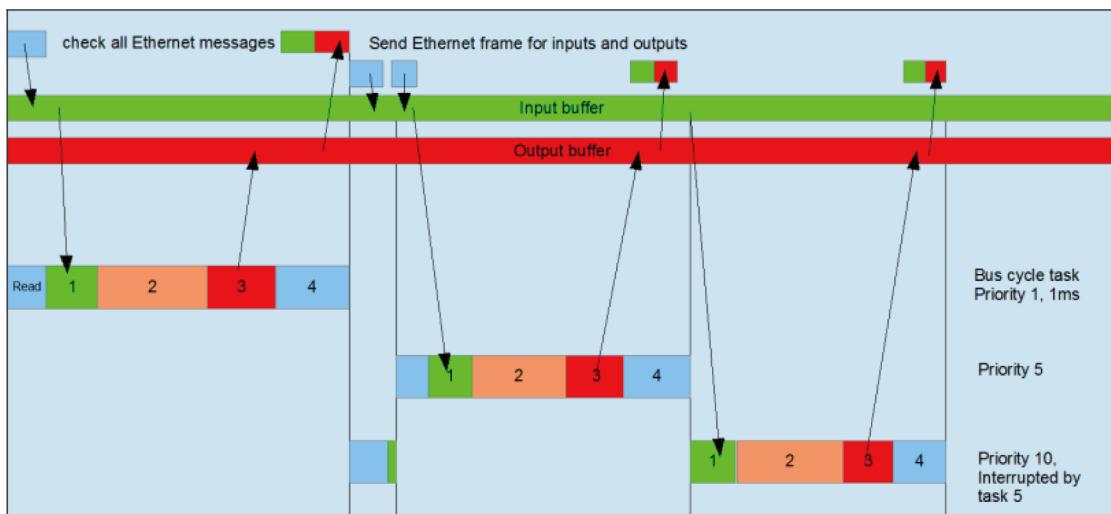
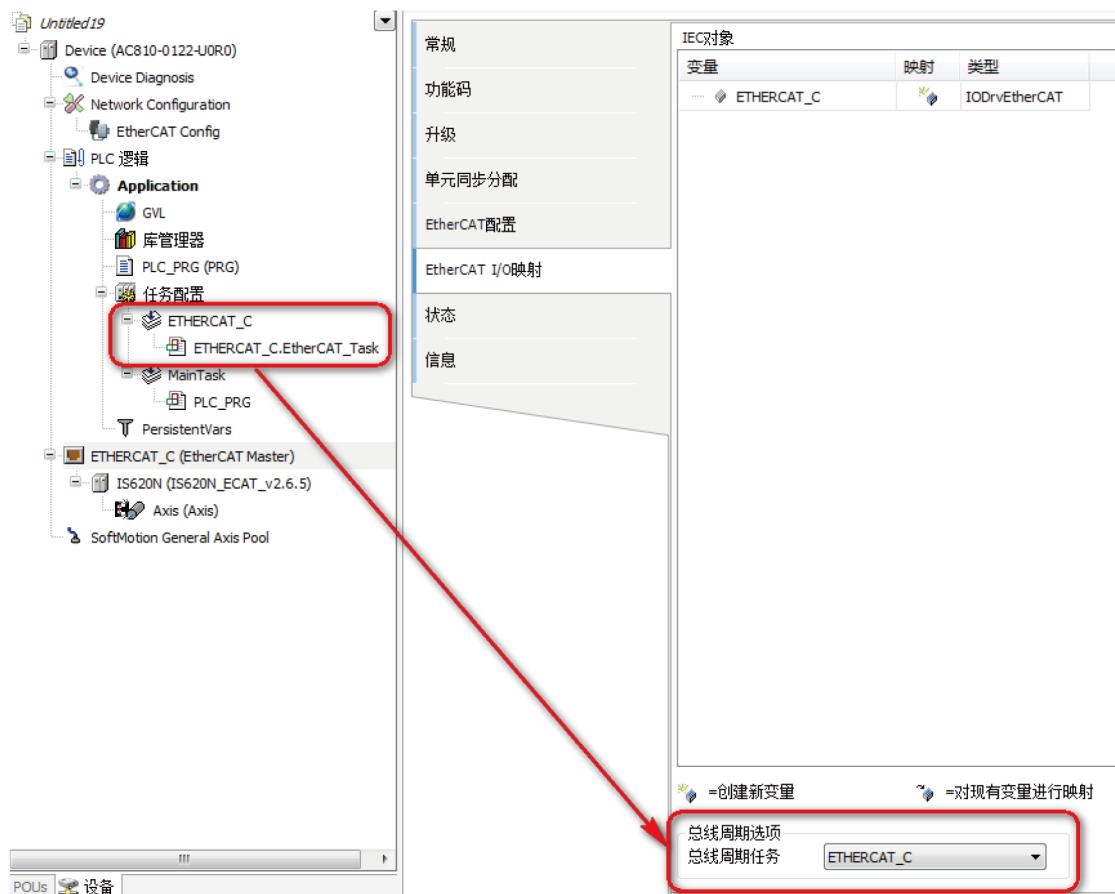


图4-12 EtherCAT 总线周期表格

## 说明

- EtherCAT主站设备自动插入后，“EtherCAT\_\*\*\*”任务也自动插入到当前应用的任务配置中；
- EtherCAT主站的总线周期任务必须与EtherCAT\_\*\*\*.EtherCAT\_Task在同一任务中执行；
- EtherCAT主站的输入输出与EtherCAT\_\*\*\*.EtherCAT\_Task在同一任务中执行；EtherCAT主站I/O映射中总线周期任务需设置对应的EtherCAT任务。因此，设备的控制程序（例如PLCOpen的轴控指令）推荐在该任务下执行。



### 4.4.3 EtherCAT 主站

#### 常规

EtherCAT主站配置对话框提供了“主站”的主要设置。



### 自动配置主站/从站

如果勾选该选项，主站和从站的主要配置将会自动完成。此时，所有从站设备编辑器将不显示FMMU/Sync设置选项卡。

### 说明

- 自动配置是默认选择，强烈推荐用于标准应用。
- 如果取消选中该选项，所有主站和从站设置都必须手动完成，这要求配置人员具备充分的专业知识。

### EtherCAT NIC 设置

- “目标地址 (MAC)”

接收报文的EtherCAT网络成员的MAC地址。如果选中“广播”选项，则使用广播地址（FF FF FF FF FF FF）。

- “网络冗余”

如果使用环型拓扑，需要冗余，则激活此选项。使用此功能，如果网络连接出现单点故障，EtherCAT 网络功能依然可以保持正常工作。激活此选项，还需要定义用于冗余功能的第二个EtherCAT NIC 设置。

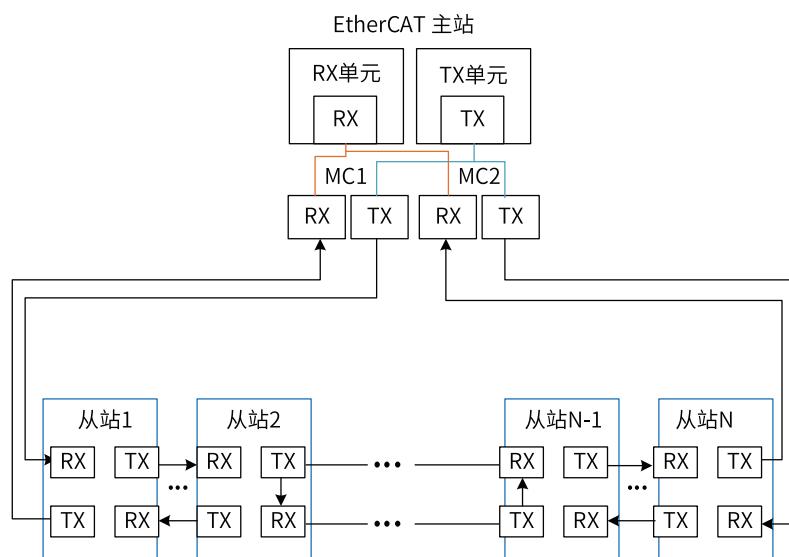


图4-13 EtherCAT环拓扑（冗余）

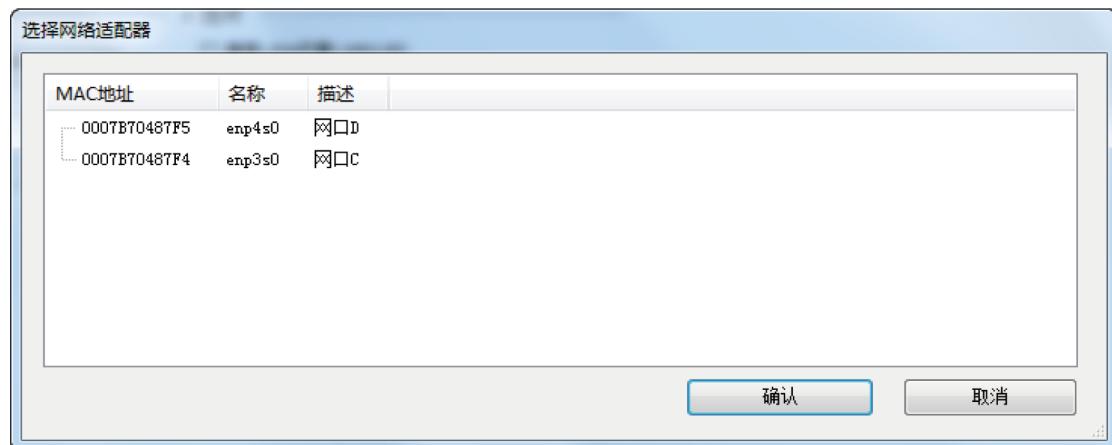
- “源地址 (MAC)”
 

PLC的MAC地址
- “网络名称”
 

网卡名称，可选择以下选项之一：
- “通过 MAC 选择网络” / “通过名称选择网络”
 

每个EtherCAT NIC有唯一的 MAC地址。因此，如果使用“通过MAC选择网络”，将不能在其他设备上使用此工程。

如果想使工程独立于设备，最好使用“通过名称选择网络”。在每个选项中，都可以使用“浏览...”选项得到当前可用的目标设备的MAC地址和名称，如下图所示：



#### ● 备份EtherCAT NIC 设置

如果“网络冗余”选项被激活，则会显示此设置。用户可根据“备份EtherCAT NIC设置”来定义相关的设置项。

### 分布式时钟

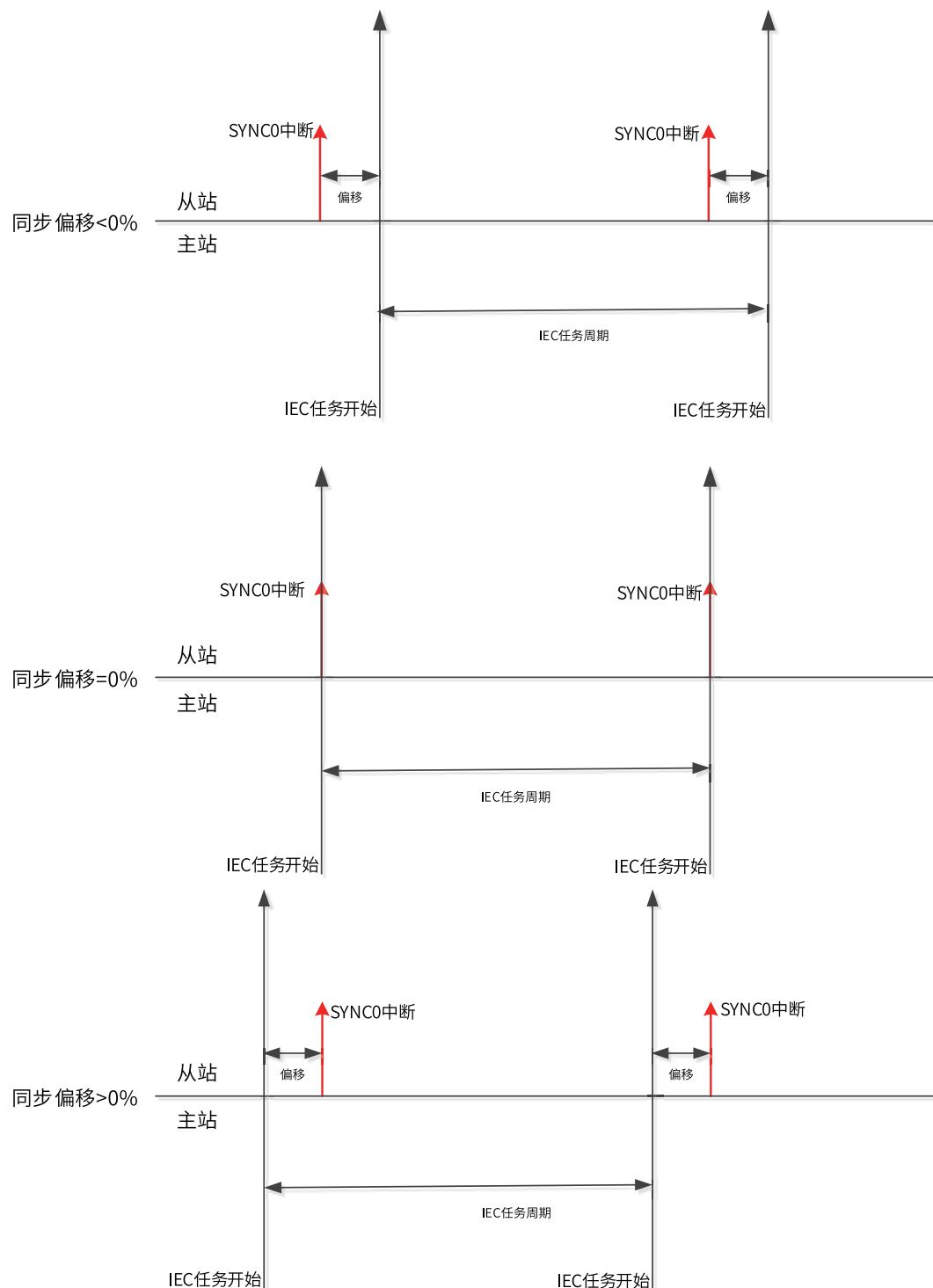
#### ● “循环时间 [μs]”

执行EtherCAT主站功能的周期时间。它必须与EtherCAT主站绑定的IEC任务的循环时间相同。如果从站“分布时钟”功能被激活，则该“循环时间”将同步刷新从站设备编辑器中“分布时钟”的设置值。

#### ● “同步偏移 [%]”

EtherCAT主站IEC任务（PLC任务）的循环时间相对于参考分布时钟（一般为SYNC0同步中断）偏移的比例，范围为-50%~+50%，默认值：30%。

同步偏移 [%] = (SYNC0中断时间 - PLC任务循环开始时间) / PLC任务循环时间



## 说明

- 默认情况下，PLC任务循环时间与从站的分布时钟的循环时间相等；
- 实际设置时，需考虑控制器的主站时钟抖动（系统实时性）、PLC任务执行时间、PLC任务循环时间的长短以及从站数等的影响。
- InoProShop V1.5.0版本开始，新建的工程EtherCAT主站类型为“EtherCAT Master SoftMotion”（与以前的所有版本“EtherCAT Master”类型有区别）。主站同步偏移默认值为30%，无特殊需求，不要更改默认值，否则，可能引入EtherCAT数据同步问题。

- “同步窗口监控”  
激活此选项会允许监控从站的同步状况。
- “同步窗口”  
同步窗口监控的时间。如果所有从站同步信息都在这个时间窗口，则变量xSyncInWindow (IoDrvEtherCAT) 被置成TRUE,否则置成FALSE。
- “诊断信息”  
在线模式下，用于显示EtherCAT主站启动、运行等相关诊断信息。

#### 选项

- “使用LRW代替LWR/LRD”  
勾选此选项将使能从站-从站的通讯。EtherCAT主站逻辑寻址将使用组合读/写命令（LRW）代替单独读（LRD）和单独写（LWR）命令。
- “使能每个任务的信息”  
勾选此选项，处理输入输出信息的读、写命令将由不同任务完成。
- “自动重启从站”  
勾选此选项，主站会在通讯异常后立即尝试重启从站。

#### 主站设置

只有在自动模式无效的情况下，才可以进行这些设置（见下文），否则这些设置会自动完成并在此对话框中隐藏。

- “输入映射地址”：用于输入数据的第一个从站的第一个逻辑地址。
- “输出映射地址”：用于输出数据的第一个从站的第一个逻辑地址。

#### 功能码

功能码指的是汇川（Inovance）伺服类产品定义的厂家参数。在主站选项卡下，可以批量读写、导入和导出多个产品的厂家参数，便于调试和维护。

全选 取消全选 读取选中参数 写入选中参数 导出功能码 导入功能码	从站	功能码编号	名称	当前值	写入值	出厂值	范围	读写权限
	IS620N	H02	基本控制参数					
		H02-00	Control mode	9	0-9	RO		
		H02-01	Absolute Encoder Mode	0	0-3	RW		
		H02-02	Rotating direction	0	0-1	RW		
		H02-03	Direction of output p...	0	0-1	RW		
		H02-05	Stop mode at servo ...	0	0-1	RW		
		H02-06	Stop mode at fault 2	1	0-2	RW		
		H02-07	Stop mode at overtr...	1	0-2	RW		
		H02-08	Stop mode at fault 1	0	0-0	RW		
		H02-09	Brake release comma...	250	0-500	RW		
		H02-10	Servo drive disable d...	150	1-1000	RW		
		H02-11	Output speed limit of...	30	0-3000	RW		
		H02-12	Waiting time from ser...	500	1-1000	RW		
		H02-15	Display of keypad wa...	0	0-1	RW		
		H02-21	Allowed minimum bra...	40	1-1000	RO		
		H02-22	Power of built-in bra...	40	1-65535	RO		
		H02-23	Resistance of built-in...	50	1-1000	RO		
		H02-24	Resistor heat dissipat...	30	10-100	RW		
		H02-25	braking resistor type	0	0-3	RW		
		H02-26	Power of external dy...	40	1-65535	RW		
		H02-27	Resistance of externa...	50	1-1000	RW		
		H02-31	Parameter initialization	0	0-2	RW		
		H02-32	Default keypad display	50	0-99	RW		
		H02-35	Display frequency of ...	0	0-20	RW		
	SV820N							

- 全选：选择所有的从站及轴，以及轴下边的伺服功能码。
- 取消全选：取消所有的选择项。
- 读取选中参数：在伺服运行状态下，读取具有RO或者RW属性的伺服功能码。
- 写入选中参数：在伺服运行状态下，写入具有WO或者RW属性的伺服功能码。
- 导出功能码：导出所有从站的伺服功能码，格式为Excel文件格式。
- 导入功能码：从外部Excel文件中，导入所有从站的功能码，如果组态与文件不一致，则报错。

## 升级

常规	全选 取消全选 下载 EtherCAT XML 文件
功能码	
升级	
单元同步分配	
总览	

从站名称	是否升级
IS620N	<input checked="" type="checkbox"/>
InoSV660N	<input checked="" type="checkbox"/>
InoSV660N_1	<input checked="" type="checkbox"/>

- 全选：选择所有的从站。
- 取消全选：取消所有选择的从站。
- 下载EtherCAT XML文件：将InoProShop中的EtherCAT从站XML文件下载到从站的E2PROM中。通过勾选多个从站可以实现批量下载。

## 单元同步分配



- 同步单元：可将从站分组，组中任一从站丢失，整个组都会丢失，但其他组不受影响。
- 增加：增加组别，之后可以在从站中选择相应的组。

## 总览

总览界面主要集中处理从站设备通信模式和显示通信状态，如下图所示。

更新数据 自动更新 顺序模式 导出总览信息 重置错误帧计数 复位从站故障										
名称	从站地址	别名地址	状态	错误帧计数	链接丢失计数	端口0	端口1	端口2	端口3	故障信息
InoSV660N	1001	--	--	--	--	--	--	--	--	--
InoSV660N_1	1002	--	--	--	--	--	--	--	--	--
InoSV660N_2	1003	--	--	--	--	--	--	--	--	--
InoSV660N_3	1004	--	--	--	--	--	--	--	--	--
InoSV660N_4	1005	--	--	--	--	--	--	--	--	--

界面中工具条各参数含义请参见下表。

参数名称	参数说明
更新数据	对从站数据刷新一次
自动更新	循环刷新监控数据
-	<ul style="list-style-type: none"> <li>• 经典模式：此时从站设备通信支持别名和顺序混合，从站窗口进行别名修改，总览窗口仅为别名显示。</li> <li>• 顺序模式：所有设备采用顺序地址通信，从站窗口和总览窗口均不可编辑别名。</li> <li>• 别名模式：所有从站采用别名通信方式，此时只有总览窗口可编辑别名地址，从站窗口仅为显示功能。</li> </ul>
导出总览信息	导出界面中表格信息。
重置错误帧计数	对错误帧计数和链接丢失计数清零。
复位故障从站	清除总览界面故障信息。

界面中表格各参数含义请参见下表。

参数名称	参数说明
名称	显示主站下所有从站的名称
从站地址	显示从站的地址信息

参数名称	参数说明
别名地址	当从站通信模式选择“别名模式”时，用于显示和从站设备别名。 • 当从站通信模式为“别名模式”时，新增从站设备时会被立即分配别名地址，通信方式为“别名模式”。 • 当从站通信模式为“经典模式”或“顺序模式”时，新增从站设备时不会被分配别名地址，通信方式为“顺序模式”。
状态	• 初始化：从站在初始化状态 • 预运行：从站在预运行状态 • 安全运行：从站在安全运行状态 • 运行：从站在运行状态 • 引导：从站在引导状态 • 无通信：无法与从站通信 • 错误：从站状态错误
错误帧计数	每一个激活的端口错误计数
链接丢失计数	每一个激活的端口链接断开计数
端口	每个端口的加载状态
故障信息	从站故障描述

## EtherCAT I/O 映射

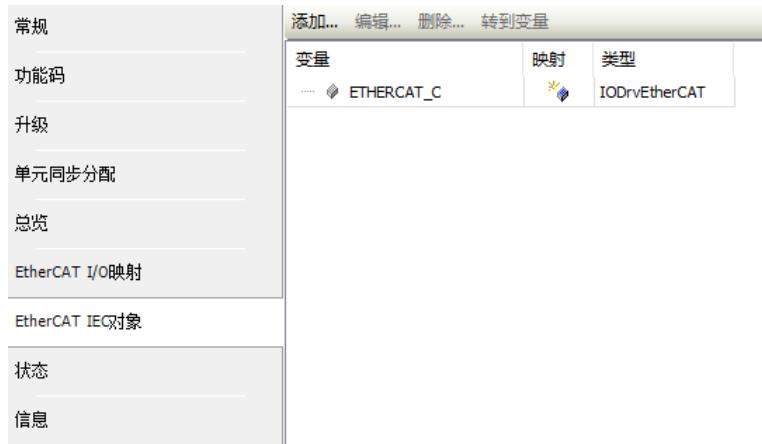
EtherCAT I/O 映射设置主站的总线循环任务。



## EtherCAT IEC对象

EtherCAT主站配置编辑器的选项卡，其中为EtherCAT I/O指定了IODrvEtherCAT类型的实例（变量），这样与EtherCAT主站连接的PLC可以由用户程序控制。有关如何映射的描述，请参见本文档章节“I/O 映射”。

自动创建的主站实例显示在对话框“IEC 对象”的底部，可以用于应用中。

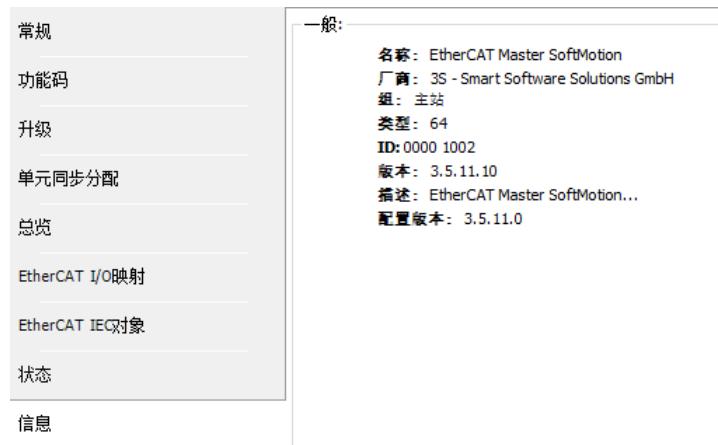


## 说明

备注：映射的变量和类型要一致。

## 信息

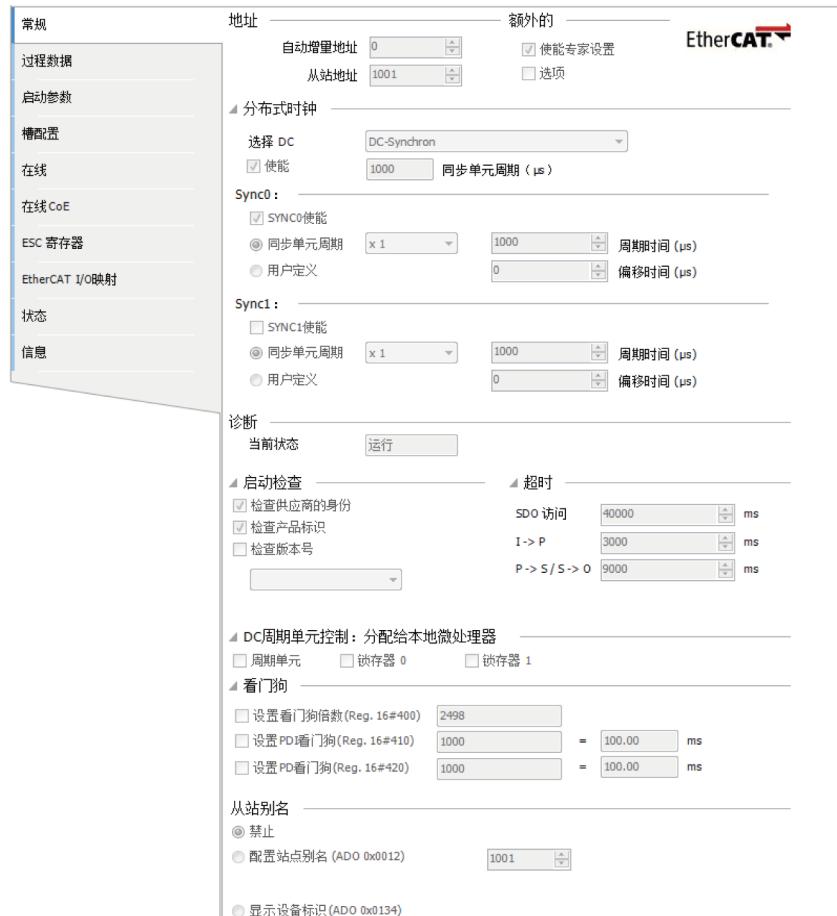
此对话框显示当前模块的下列信息：名称、供应商、组、类型、ID和版本等。



### 4.4.4 EtherCAT 从站

#### 常规

EtherCAT从站“常规”界面如下，其中提供了从站基本配置信息：



## 地址

如果主站设备编辑器中“自动配置主站/从站”没有勾选，则以下选项可设：

- “自动增量地址”：自动增量地址（16位），由网络中从站的物理拓扑位置确定。此地址只在EtherCAT主站启动时使用，通过顺序寻址的方式，将EtherCAT从站地址分配给相应物理拓扑位置的从站。

顺序寻址时，EtherCAT协议标准，从站的自动增量地址由其在物理拓扑网络的连接位置确定，用一个负数来表示。顺序寻址会发送包含的子报文数据帧，数据帧经过每个从站设备时，子报文中的自动增量地址数据将自动加1；物理从站在接收数据帧时，通过查找数据帧子报文中自动增量地址是否为0的方式，来寻址到自己的报文。这种数据帧经过从站更新报文内部地址变量的机制，被称为“顺序寻址”或者“自动增量寻址”。

- “从站地址”：从站的配置地址（正名地址），由主站在启动时分配。此地址独立于网络中的实际位置，从站的地址与其在网段内的连接顺序无关。
- “使能专家设置”：如果勾选此选项，将可以配置分布式时钟设置、启动时检查、超时设置、周期单元控制、看门狗等选项。

## 分布式时钟

- “使能”如果勾选，表示使用“分布式时钟”功能，
- 同步单元周期 (μs)：如果“分布式时钟”功能被使能，同步单元周期值将与EtherCAT主站的循环时间置相同。
- “选择 DC”：该下拉菜单提供设备描述文件中所有关于分布式时钟的设置。一般包含自由运行、SM事件同步、DC同步。各项设置的功能如下表所述：

编号	设置	功能
1	自由运行模式	在该模式下，本地控制周期由一个本地定时器中断产生，周期时间可以由主站设定，这是从站的可选功能。
2	SM事件同步	本地周期在发生数据输入或者输出事件的时候触发，主站可以将过程数据帧的发送周期写给从站，从站可以检查是否支持这个周期时间或对周期时间进行本地优化。从站可以选择支持这个功能。通常同步于数据输出事件，如果从站只有输入数据，则同步于数据输入事件。
3	DC同步	本地周期由SYNC事件触发，主站必须在SYNC事件之前完成数据帧的发送，为此要求主站时钟也要同步于参考时钟。
4	SYNC0: SYNC1:	从站同步信号0/1。从站的分布时钟控制单元（EtherCAT专用的通讯芯片的内部功能）可以产生两个同步信号，分别为SYNC0和SYNC1，这两个信号给从站的应用层程序提供中断标志，或者直接触发输出数据更新
5	SYNC 0使能： SYNC 1使能：	勾选后将启动SYNC0/SYNC1的同步信号。 “同步单元周期”：如果选择这个选项，主站周期时间乘以选择的系数即为从站的同步周期时间。“周期时间(μs)”域显示当前设置的从站周期时间。

- “用户定义”：如果选择此选项，可以在“循环时间(μs)”字段输入微秒级的用户自定义同步周期时间。

### 诊断

此部分仅在线模式下可见：

- “当前状态”：显示从站的当前通讯状态机的状态，可能值为：初始化、预运行、安全运行、运行、引导（汇川的伺服暂时不支持）。如果当前状态是‘运行’，则表示从站配置已正确完成。

### 选项

“选项”：如果一个从站设备被定义为‘可选的’，则不会创建错误消息，以防止该设备不存在于总线系统中。若要激活此选项，必须在从站设备中存储一个站地址，因此必须在E2PROM中定义并写入“站点别名”地址。且只有在勾选EtherCAT主站设置中的“自动配置主站/从站”选项并且EtherCAT从站支持此功能的前提下，此选项才有效。

### 启动检查

默认情况下，系统启动时会根据当前配置自动检查供应商ID和产品ID。如果检测到不匹配，总线会停止运行且不执行下一步操作。此设定是为了避免下载错误的配置。

### 超时

默认情况下，以下操作不定义为超时。不过，如要观察这些操作是否超过了特定时间，可以在这里设置时间：

- “SDO访问”：EtherCAT主站协议中SDO（服务数据对象）访问从站的超时时间。
- “I->P”：从站通讯状态机由‘初始化’转换到‘预运行’。
- “P->S/S->O”：从站通讯状态机由‘预运行’转换到‘安全运行’，或由‘安全运行’转换到‘运行’。
- DC周期单元控制：分配给本地微处理器来选择分布时钟功能定义的选项。该控制功能已经在EtherCAT从站的寄存器0x980中完成，可能的设置为：周期单元、锁存器单元0、锁存器单元1。

### 看门狗

- “设置倍数”：设置看门狗定时器的倍频数，以确定其计时的最小增量单位。默认值：2498，看门狗定时器的最小计时增量单位为100us。
- “设置PDI看门狗”：如果激活PDI看门狗，当EtherCAT从站的PDI(物理设备接口)通讯时间超过设置值，则会触发看门狗。

- “设置 SM 看门狗”：如果激活SM (同步管理)看门狗，当 EtherCAT从站的PD（过程数据）通讯时间超过设置值，则会触发看门狗。

### 从站别名

这些设置只有在勾选“选项”且从站设备支持别名地址（由设备描述文件定义）的情况下才有效。如果已配置从站相应的别名地址，调整其在物理拓扑网络的位置时，无需更改用户程序中的配置，从站仍可以正常运行。

注意，更改从站的别名地址后，必须重新下载用户程序才能使别名生效；另外，某些从站必须断电重启才能使别名生效，具体请参考相关从站的使用手册。

- “禁止”：如果配置此项，则表示该从站不会检测别名地址。
- “配置站点别名 (ADO 0x0012)”：在从站设备支持的情况下，且“选项”勾选的情况下，才可以在在线运行的状态下写入别名地址。
- “写入 E2PROM”：此命令只在在线模式下可见。它允许将定义的地址写入从站的E2PROM。如果从站不支持，则此命令无效且从站不能以站点别名形式工作。
- “实际地址”：此栏目只在在线模式下可见，显示从站的实际地址。它可以用来检查‘写E2PROM’的命令是否成功。
- “显示设备标识ADO(0X0134)”：保留。
- “数据字”（2字节）：保留。

## FMMU/Sync

当主站的自动配置模式配置为无效时，此对话框仅由EtherCAT从站配置编辑器的选项卡提供。它显示由设备描述文件定义的从站FMMUs（现场总线内存管理单元）和Sync（同步管理器），这些设置可以修改，例如配置从站-从站通讯。

---

### 说明

这些都是高级设置，对标准应用而言一般不需要。

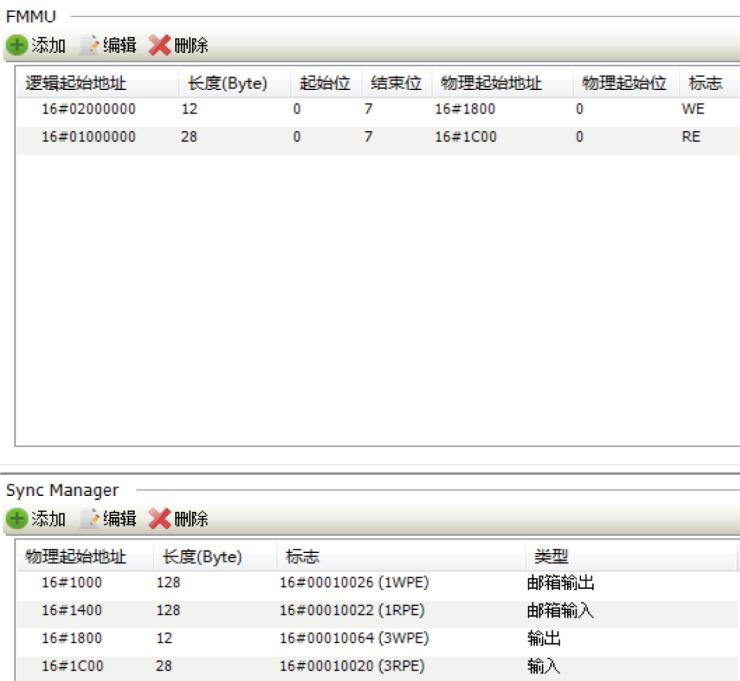
---

### FMMU

此选项卡显示了用于处理过程数据的从站现场总线内存管理单元，其中定义了每个物理地址（“Ph. GlobStartAddr”）上的逻辑地址（“Ph. Start Address”）映射。通过编辑 FMMU对话框中的“添加...”以及“编辑...”按键可以添加新单元。

### 同步管理器 (Sync Manager)

此选项卡显示了从站的同步管理器。其中可定义每个可用同步管理器的类型 (Mailbox In, Mailbox Out, Inputs, Outputs)、物理起始地址、访问类型、缓冲区以及中断需要访问的物理地址。在同步管理器编辑器中，通过“添加”或“编辑”按键可以新增或修改同步管理。



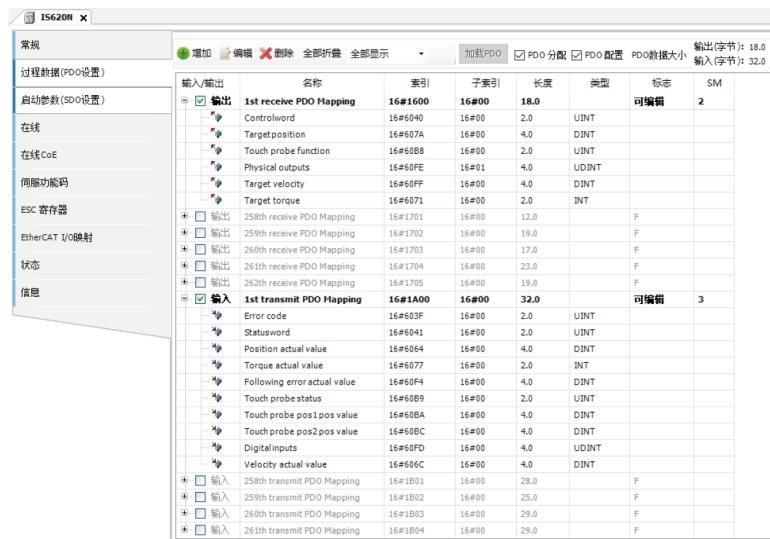
The screenshot shows two tables under the 'Sync Manager' tab:

物理起始地址	长度(Byte)	标志	类型
16#1000	128	16#00010026 (1WPE)	邮箱输出
16#1400	128	16#00010022 (1RPE)	邮箱输入
16#1800	12	16#00010064 (3WPE)	输出
16#1C00	28	16#00010020 (3RPE)	输入

## 过程数据

实际自动化控制系统中，应用程序之间通常有两种数据交换形式：时间关键（time-critical）和非时间关键（non-time-critical）。时间关键表示特定动作必须在指定时间窗口内完成，如果不能在指定时间窗口内完成通信，则有可能导致控制失效。周期性发送时间关键数据的过程称为周期性过程数据通信（PDO），非时间关键数据可以非周期性发送，在EtherCAT中采用MailBox数据通信（SDO）。

过程数据第一行为PDO编辑功能键及显示PDO信息，如下图所示：



The screenshot shows the 'IS620H' software interface with the '过程数据(PDO设置)' tab selected. It displays a table of PDO mappings:

输入/输出	名称	索引	子索引	长度	类型	标志	SM
输出	1st receive PDO Mapping	16#1600	16#00	18.0	UINT		可编辑
	Controlword	16#6040	16#00	2.0	UINT		
	Targetposition	16#607A	16#00	4.0	DINT		
	Touch probe function	16#60B8	16#00	2.0	UINT		
	Physical outputs	16#60FE	16#01	4.0	UDINT		
	Target velocity	16#60FF	16#00	4.0	DINT		
	Target torque	16#6071	16#00	2.0	INT		
输出	258th receive PDO Mapping	16#1701	16#00	12.0		F	
输出	259th receive PDO Mapping	16#1702	16#00	19.0		F	
输出	260th receive PDO Mapping	16#1703	16#00	17.0		F	
输出	261th receive PDO Mapping	16#1704	16#00	23.0		F	
输出	262th receive PDO Mapping	16#1705	16#00	19.0		F	
输入	1st transmit PDO Mapping	16#1A00	16#00	32.0		可编辑	3
	Error code	16#603E	16#00	2.0	UINT		
	Statusword	16#6041	16#00	2.0	UINT		
	Position actual value	16#6044	16#00	4.0	DINT		
	Torque actual value	16#6077	16#00	2.0	INT		
	Following error actual value	16#60F4	16#00	4.0	DINT		
	Touch probe status	16#60B9	16#00	2.0	UINT		
	Touch probe pos1pos value	16#60BA	16#00	4.0	DINT		
	Touch probe pos2pos value	16#60BC	16#00	4.0	DINT		
	Digitalinputs	16#60FD	16#00	4.0	UDINT		
	Velocity actual value	16#606C	16#00	4.0	DINT		
输入	258th transmit PDO Mapping	16#1B01	16#00	28.0		F	
输入	259th transmit PDO Mapping	16#1B02	16#00	25.0		F	
输入	260th transmit PDO Mapping	16#1B03	16#00	29.0		F	
输入	261th transmit PDO Mapping	16#1B04	16#00	29.0		F	

- “增加” 根据当前PDO组的属性（仅限可编辑属性）增加PDO选择项。用户可以选择增加一组数据或多组数据，使用“Ctrl+”，“Shift+”鼠标左键可同时增加多组数据，请注意选择增加项的对象字典索引是否正确。注： PDO增加的个数不能超过伺服手册中限制的个数。
- “编辑” 根据当前PDO组的属性（仅限可编辑属性）编辑当前的PDO选择项。
- “删除” 根据当前PDO组的属性（仅限可编辑属性）删除当前的PDO选择项。用户可以删除多个或者单个项，使用“Ctrl+”，“Shift+”鼠标左键可同时选中多项，使用快捷键“Delete”或者鼠标右键删除。

- “全部折叠”：折叠当前所有展开的PDO组。
- “过滤功能”：包含，全部显示，显示输出PDO，显示输入PDO，全部显示，显示输出和输入的PDO，显示输出PDO，只显示输出PDO，显示输入PDO，只显示输入PDO组。
- “加载 PDO”：只有在从站运行状态下，才可以将当前从站运行中的PDO组数据上传到后台。
- “PDO 分配”：勾选PDO分配，同时勾选“启动参数”界面的“显示系统参数”选择项，则启动参数组会增加当前输入和输出PDO组的分配信息，如下图所示：

行	索引 : 子索引	名称	值	位长度	是否下载	有错退出	有错跳行
1	16#1C12:16#00	clear pdo 1C12	16#00000000	8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	16#1C13:16#00	clear pdo 1C13	16#00000000	8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	16#1C12:16#01	download pdo 1C12:1 index	16#00001701	16	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	16#1C12:16#00	download pdo 1C12 count	16#00000001	8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	16#1C13:16#01	download pdo 1C13:1 index	16#00001B01	16	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	16#1C13:16#00	download pdo 1C13 count	16#00000001	8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	16#6060:16#00	Modes of operation	16#00000008	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

- “PDO 配置”：勾选PDO配置，同时勾选“启动参数”界面的“显示系统参数”选择项，则启动参数组会增加当前输入和输出PDO组的配置信息，如下图所示：

行	索引 : 子索引	名称	值	位长度	是否下载	有错退出	有错跳行
1	16#1600:16#00	clear pdo 1600	16#00000000	8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	16#1600:16#01	download pdo 1600:1 entry	16#60600008	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	16#1600:16#02	download pdo 1600:2 entry	16#60650020	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	16#1600:16#03	download pdo 1600:3 entry	16#60660010	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	16#1600:16#04	download pdo 1600:4 entry	16#60670020	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	16#1600:16#05	download pdo 1600:5 entry	16#60680010	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	16#1600:16#06	download pdo 1600:6 entry	16#606880010	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	16#1600:16#00	download pdo 1600 count	16#00000006	8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	16#1A00:16#00	clear pdo 1A00	16#00000000	8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10	16#1A00:16#01	download pdo 1A00:1 entry	16#60410010	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11	16#1A00:16#02	download pdo 1A00:2 entry	16#60640020	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
12	16#1A00:16#03	download pdo 1A00:3 entry	16#60690010	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
13	16#1A00:16#04	download pdo 1A00:4 entry	16#60BA0020	32	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

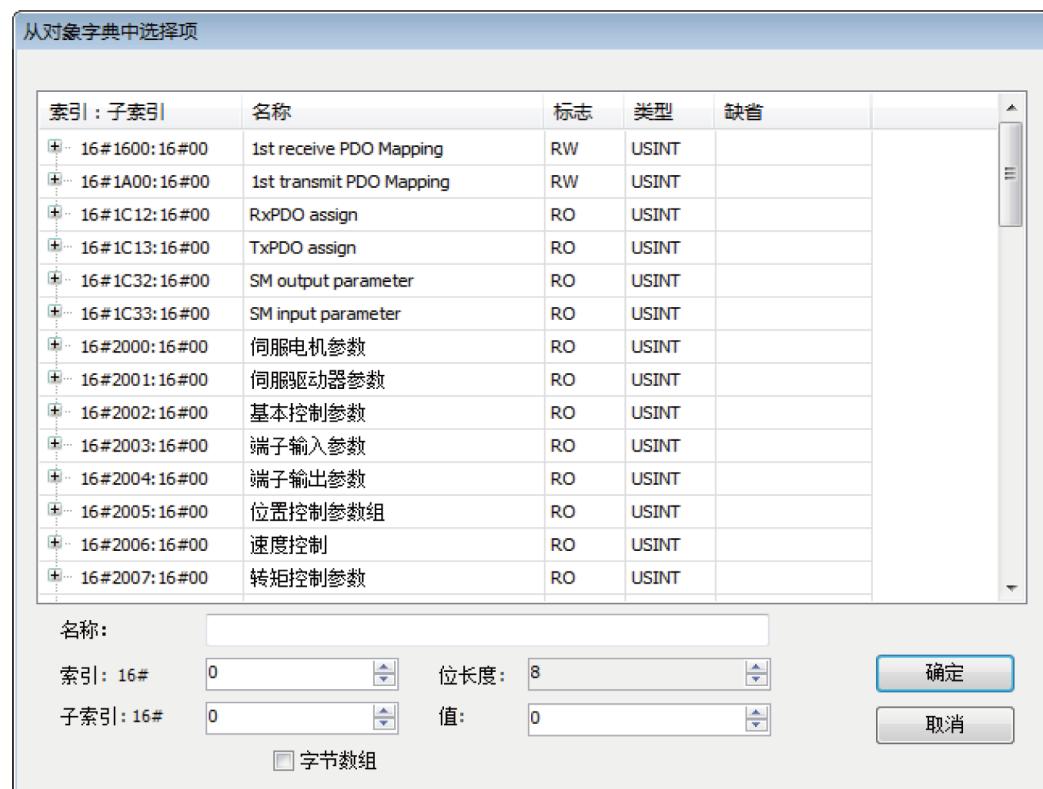
- “PDO数据大小”显示当前所有输出和输入PDO总长度。

## 启动参数

启动参数在系统启动时可由SDO（服务数据对象）传送给从站，启动参数包含了从站启动时所需的一些基本配置参数，常见界面如下：

行	索引 : 子索引	名称	值	位长度	是否下载	有错退出	有错跳行	下一行
1	16#6060:16#00	Modes of operation	8	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
2	16#6098:16#00	Homing method	26	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
3	16#6099:16#01	Speed during search for switch	10485760	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
4	16#609A:16#00	Homing acceleration	104857600	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
5	16#6099:16#02	Speed during search for zero	2097152	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
6	16#2005:16#24	Time of home searching	50000	16	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
7	16#60E0:16#00	Positive torque limit value	5000	16	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
8	16#60E1:16#00	Negative torque limit value	5000	16	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
9	16#60F7:16#00	Max profile velocity	104857600	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0

- “添加”添加一个 SDO 项目到启动参数列表中，弹出的对象字典选择框如下所示：



添加SDO之前，可以通过编辑栏下方的栏目修改其参数，定义其索引、子索引、位长度及值，从而形成一个新的启动参数。可以使用组合按键“Ctrl+”，“Shift+”鼠标左键同时添加多组启动参数。

- “编辑”可以编辑当前选择项的参数，只读参数是无法编辑的，例如系统参数。
- “删除”可以删除当前选择项，使用组合按键“Ctrl+”，“Shift+”鼠标左键同时选中多组启动参数，然后使用快捷键DEL或者删除按键删除。
- 向上移动，向下移动  
SDO列表的顺序（由上至下）代表了启动参数被传输到模块的顺序。通过“向上移动”和“向下移动”按钮可以改变其传输到模块的先后顺序。
- 全选，取消全选  
SDO下载后，可以不用重复下载。使用取消全选（系统参数无法取消）可取消下载属性，或者勾选部分需要下载的属性；使用全选可全部设置为下载属性。
- 显示系统参数  
显示PDO分配和PDO配置勾选后，SDO中增加的参数，只做显示对比使用。
- SDO部分的“值”和“注释”栏，可以通过按[ space ]或点击鼠标对其进行直接编辑。

## 说明

SDO传输过程出现错误时，可以通过如下步骤解决：

- “有错退出”：如果检测到错误，则退出SDO传输；
- “有错跳行”以及“下一行”：如果检测到错误，则SDO传输会跳转到“下一行”所指的行号。（行号显示于行栏目中）中输入的行中继续进行。

行	索引 : 子索引	名称	值	位长度	是否下载	有错退出	有错跳行	下一行
1	16#6060:16#00	Modes of operation	8	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	2
2	16#6098:16#00	Homing method	26	8	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0
3	16#6099:16#01	Speed during search for switch	10485760	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
4	16#609A:16#00	Homing acceleration	104857600	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
5	16#6099:16#02	Speed during search for zero	2097152	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
6	16#2005:16#24	Time of home searching	50000	16	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
7	16#60E0:16#00	Positive torque limit value	5000	16	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
8	16#60E1:16#00	Negative torque limit value	5000	16	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0
9	16#607F:16#00	Max profile velocity	104857600	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0

图4-14 错误处理配置示例

## 槽配置

- “槽配置”用于支持ETG5001.1协议的从站，是配置其模块或功能的选项卡。

如下图所示，当前的模式为CSP\_CSV：



用户可选以下模式：

编号	模式	定义
1	CSP	表示同步位置
2	PP	轮廓位置
3	CSV	同步速度
4	PV	轮廓速度
5	CST	同步力矩
6	PT	轮廓力矩

如果位置速度模式不满足实际需求，可切换成其他模式，如CST “同步力矩模式”。

默认的“CSP/CSV”满足大部分应用场合，如果现场驱动轴要同时使用同步位置和同步力矩功能，请选择“CSP/CST”项。

## 说明

- “变更”用于切换不同的模式；
- “删除”可以删除当前的模式；
- 如需变更新的模式，需要先删除当前槽模式。

由CSP\_CSV变更为CST模式后，过程数据和I/O映射都会有相应变化：

变更为CST前：

常规

过程数据(PDO设置)

启动参数(SDO设置)

槽配置

在线

在线CoE

伺服功能码

EtherCAT I/O映射

状态

信息

输入/输出	名称	索引	子索引	长度	类型	标志	SM
<input checked="" type="checkbox"/> 输出	Outputs	16#1600	16#00	13.0		可编辑	2
CSP_CS ControlWord	16#6040	16#00	2.0		UINT		
CSP_CS Modes of Operation	16#6060	16#00	1.0		SINT		
CSP_CS Target position	16#607A	16#00	4.0		DINT		
CSP_CS Touch probe function	16#6088	16#00	2.0		UINT		
CSP_CS Target velocity	16#60FF	16#00	4.0		DINT		
<input checked="" type="checkbox"/> 输出	Outputs	16#1610	16#00	13.0		可编辑	2
CSP_CS_1 ControlWord	16#6840	16#00	2.0		UINT		
CSP_CS_1 Modes of Operation	16#6860	16#00	1.0		SINT		
CSP_CS_1 Target position	16#687A	16#00	4.0		DINT		
CSP_CS_1 Touch probe function	16#68B8	16#00	2.0		UINT		
CSP_CS_1 Target velocity	16#68FF	16#00	4.0		DINT		
<input checked="" type="checkbox"/> 输出	Outputs	16#1620	16#00	13.0		可编辑	2
CSP_CS_2 ControlWord	16#7040	16#00	2.0		UINT		
CSP_CS_2 Modes of Operation	16#7060	16#00	1.0		SINT		
CSP_CS_2 Target position	16#707A	16#00	4.0		DINT		
CSP_CS_2 Touch probe function	16#70B8	16#00	2.0		UINT		
CSP_CS_2 Target velocity	16#70FF	16#00	4.0		DINT		
<input checked="" type="checkbox"/> 输出	Outputs	16#1630	16#00	13.0		可编辑	2
CSP_CS_3 ControlWord	16#7840	16#00	2.0		UINT		
CSP_CS_3 Modes of Operation	16#7860	16#00	1.0		SINT		
CSP_CS_3 Target position	16#787A	16#00	4.0		DINT		
CSP_CS_3 Touch probe function	16#78B8	16#00	2.0		UINT		
CSP_CS_3 Target velocity	16#78FF	16#00	4.0		DINT		
<input checked="" type="checkbox"/> 输入	Inputs	16#1A00	16#00	31.0		可编辑	3
CSP_CS Error code	16#603F	16#00	2.0		UINT		
CSP_CS StatusWord	16#6041	16#00	2.0		UINT		
CSP_CS Modes of Operation Display	16#6061	16#00	1.0		SINT		
CSP_CS Position actual value	16#6064	16#00	4.0		DINT		
CSP_CS ActualVelocity	16#606C	16#00	4.0		DINT		

变更为CST后：

常规

过程数据(PDO设置)

启动参数(SDO设置)

槽配置

在线

在线CoE

伺服功能码

EtherCAT I/O映射

状态

信息

输入/输出	名称	索引	子索引	长度	类型	标志	SM
<input checked="" type="checkbox"/> 输出	Outputs	16#1600	16#00	4.0		可编辑	2
CST ControlWord	16#6040	16#00	2.0		UINT		
CST Target torque	16#6071	16#00	2.0		INT		
<input checked="" type="checkbox"/> 输出	Outputs	16#1610	16#00	13.0		可编辑	2
CSP_CS_1 ControlWord	16#6840	16#00	2.0		UINT		
CSP_CS_1 Modes of Operation	16#6860	16#00	1.0		SINT		
CSP_CS_1 Target position	16#687A	16#00	4.0		DINT		
CSP_CS_1 Touch probe function	16#68B8	16#00	2.0		UINT		
CSP_CS_1 Target velocity	16#68FF	16#00	4.0		DINT		
<input checked="" type="checkbox"/> 输出	Outputs	16#1620	16#00	13.0		可编辑	2
CSP_CS_2 ControlWord	16#7040	16#00	2.0		UINT		
CSP_CS_2 Modes of Operation	16#7060	16#00	1.0		SINT		
CSP_CS_2 Target position	16#707A	16#00	4.0		DINT		
CSP_CS_2 Touch probe function	16#70B8	16#00	2.0		UINT		
CSP_CS_2 Target velocity	16#70FF	16#00	4.0		DINT		
<input checked="" type="checkbox"/> 输出	Outputs	16#1630	16#00	13.0		可编辑	2
CSP_CS_3 ControlWord	16#7840	16#00	2.0		UINT		
CSP_CS_3 Modes of Operation	16#7860	16#00	1.0		SINT		
CSP_CS_3 Target position	16#787A	16#00	4.0		DINT		
CSP_CS_3 Touch probe function	16#78B8	16#00	2.0		UINT		
CSP_CS_3 Target velocity	16#78FF	16#00	4.0		DINT		
<input checked="" type="checkbox"/> 输入	Inputs	16#1A00	16#00	18.0		可编辑	3
CST Error code	16#603F	16#00	2.0		UINT		
CST StatusWord	16#6041	16#00	2.0		UINT		
CST Position actual value	16#6064	16#00	4.0		DINT		
CST ActualVelocity	16#606C	16#00	4.0		DINT		
CST Torque actual value	16#6077	16#00	2.0		INT		
CST Digital inputs	16#60FD	16#00	4.0		UDINT		
<input checked="" type="checkbox"/> 输入	Inputs	16#1A10	16#00	31.0		可编辑	3
CSP_CS_1 Error code	16#683F	16#00	2.0		UINT		

- “下载槽配置”：配置完当前模块后，需要把当前配置的槽信息下载到设备中。勾选此选择项后会在启动参数中增加如下参数：

常规										
行	索引:子索引	名称	值	位长度	是否下载	有错退出	有错跳行	下一行	注释	
1	16#6060:16#00	Modes of operation	8	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	M	
2	16#6860:16#00	Modes of operation	8	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	M	
3	16#7060:16#00	Modes of operation	8	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	M	
4	16#7860:16#00	Modes of operation	8	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	M	
5	16#F030:16#00	clear slot cfg 0xf030 entries	0	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0		
6	16#F030:16#01	download slot cfg 0xf030 entry	256	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0		
7	16#F030:16#02	download slot cfg 0xf030 entry	256	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0		
8	16#F030:16#03	download slot cfg 0xf030 entry	256	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0		
9	16#F030:16#04	download slot cfg 0xf030 entry	256	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0		
10	16#F030:16#00	download slot cfg 0xf030 entry count	4	8	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0		

① - 表示需要下载的当前模块模式ID。

② - 表示需要下载的模块模式总个数。

## 在线

登陆到设备后才可使用从站在线配置编辑器。这个界面主要用于手动切换从站的状态机、读写从站的E2PROM、以及FoE协议的上传和下载和从站固件升级。

设备状态

初始化
引导状态
当前状态:

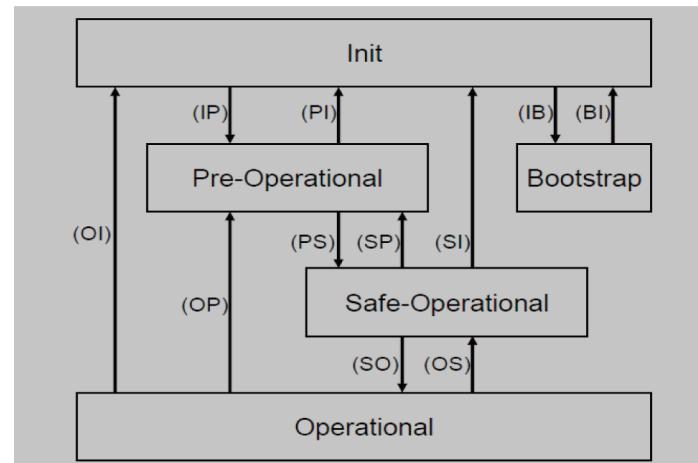
预运行
安全运行
请求状态:

运行
清除错误

FoE

E2PROM接口

EtherCAT状态机负责协调主站和从站应用程序在初始化和运行时的状态关系，如下图所示：



EtherCAT运行转换顺序：初始化->预运行->安全运行->运行。

状态过程详细描述：

状态转换	本地管理服务
IP	开始邮箱通信 (Start Mailbox Communication)
PI	停止邮箱通信 (Stop Mailbox Communication)
PS	开始输入更新 (Start Input Update)
SP	停止输入更新 (Stop Input Update)
SO	开始输出更新 (Start Output Update)
OS	停止输出更新 (Stop Output Update)
OP	停止输出更新, 停止输入更新 (Stop Output Update, Stop Input Update)
SI	停止输出更新, 停止邮箱通信 (Stop Input Update, Stop Mailbox Communication)
OI	停止输出更新, 停止输入更新, 停止邮箱通信 (Stop Output Update, Stop Input Update, Stop Mailbox Communication)
IB	开始引导模式 (Start Bootstrap Mode)
BI	重启设备 (Restart Device)

“初始化”，“预运行”，“安全运行”，“运行”以及“清除错误”按键可以用来进行调试。

### EtherCAT 文件访问

如果想要向从站传输固件文件，进行固件升级，可以点击“引导状态”按钮来将从站转换为“引导模式”。使用相应按钮可以完成对固件文件的“下载”和“上传”。此时，会打开一个保存或打开固件文件的对话框，需要使用字符串和密码来执行文件的传输。此信息由从站设备提供，并记录在从站的数据表中，升级固件过程中，不建议断电或者进行状态切换，升级完成后，再进行此类操作。

### E2PROM 访问

从站配置可以由 E2PROM 读取，也可写入 E2PROM。同固件文件传输一样，该操作也会弹出一个打开或保存文件的对话框。

用户可以利用“写 E2PROM XML”的命令直接将从站配置从XML文件写入到设备中。只有在XML文件中存在配置数据时，此命令才有效 (<配置数据>部分)

### 清除错误

当前状态有错误时，可以使用此按键，清除当前的错误状态。

## 在线COE

只在总线正常运行并且登录PLC后才可以读取在线COE的值，如下图所示：

Read this page  Auto Update  Offline from ESI file  Online from device

索引 : 子索引	名称	标志	类型	值
16#1000:16#00	Device type	RO	UDINT	---
16#1001:16#00	Error Register	RO	USINT	---
16#1008:16#00	Device name	RO	STRING(31)	
16#1009:16#00	Hardware version	RO	STRING(4)	
16#100A:16#00	Software version	RO	STRING(4)	
16#1018:16#00	Identity	RO	USINT	
16#1600:16#00	1st receive PDO Mapping	RW	USINT	
16#1701:16#00	258th receive PDO Mapping	RO	USINT	
16#1702:16#00	259th receive PDO Mapping	RO	USINT	
16#1703:16#00	260th receive PDO Mapping	RO	USINT	
16#1704:16#00	261th receive PDO Mapping	RO	USINT	
16#1705:16#00	262th receive PDO Mapping	RO	USINT	
16#1A00:16#00	1st transmit PDO Mapping	RW	USINT	
16#1B01:16#00	258th transmit PDO Mapping	RO	USINT	
16#1B02:16#00	259th transmit PDO Mapping	RO	USINT	
16#1B03:16#00	260th transmit PDO Mapping	RO	USINT	
16#1B04:16#00	261th transmit PDO Mapping	RO	USINT	
16#1C00:16#00	Sync manager type	RO	USINT	
16#1C12:16#00	RxPDO assign	RO	USINT	
16#1C13:16#00	TxDPO assign	RO	USINT	
16#1C32:16#00	SM output parameter	RO	USINT	
16#1C33:16#00	SM input parameter	RO	USINT	
16#2000:16#00	伺服电机参数	RO	USINT	
16#2001:16#00	伺服驱动器参数	RO	USINT	
16#2002:16#00	基本控制参数	RO	USINT	
16#2003:16#00	端子输入参数	RO	USINT	
16#2004:16#00	端子输出参数	RO	USINT	
16#2005:16#00	位置控制参数组	RO	USINT	
16#2006:16#00	速度控制	RO	USINT	

## EOE设置

设置

虚拟以太网端口

虚拟 MAC Id:

交换机端口  IP站

IP设置

IP地址:	<input type="text" value="0 . 0 . 0 . 0"/>
子网掩码:	<input type="text" value="0 . 0 . 0 . 0"/>
默认网关:	<input type="text" value="0 . 0 . 0 . 0"/>
DNS服务:	<input type="text" value="0 . 0 . 0 . 0"/>
DNS名称:	<input type="text" value="SV820N"/>

EtherNET over EtherCAT允许将任意EtherNET（以太网）设备通过转换端子连接到EtherCAT，同时不影响EtherCAT的实时性。类似于众所周知的互联网协议（例如TCP/IP、VPN、PPPoE(DSL)），以太网帧通过EtherCAT协议传输。该设置允许将标准网络设备通过交换机连接到终端设备，如打印机或PC机。

对于支持EtherNET over EtherCAT（EOE）的特殊从站，可以进行通讯设置，且只有在设备支持EtherNET over EtherCAT时，才会提供以下的对话框。

- “虚拟以太网端口”：此选项使能从站的EOE功能。如果激活此选项，则必须定义一个特殊的“虚拟MAC地址”。
- “交换机端口”：此设备作为IP端口，必须设置以太网通讯参数。

- “IP 站”：此设备作为IP端口，必须设置以太网通讯参数。

以太网通讯参数必须基于虚拟以太网适配器的参数设置。“IP 地址”，“子网掩码”以及“默认网关”中的条目各分配4字节以识别网络中的从站。将光标定位在相应的编辑区域后，可以对默认设置进行修改。

## 说明

IP端口必须与虚拟以太网适配器处于同一网段。例如，如果以太网适配器的地址是192.168.1.1，子网掩码是255.255.255.0，则IP端口必须在192.168.1.2 到192.168.1.254的范围之内。

- “DNS 服务器”：DNS服务器的IP地址
- “DNS 名称”：DNS服务器的名称

## 伺服功能码

功能码指的是汇川 (Inovance) 的伺服类产品定义的厂家参数。在从站的“功能码”选项卡下，针对伺服从站读写、导入和导出厂家参数，便于调试和维护。如下图所示：

功能码组						
Axis_1						
H00 Servo Motor Parameters						
H01 Servo Drive Parameters						
H02 Basic Control Parameters						
H03 Input Terminal Parameters						
H04 Output terminal Parameters						
H05 Position Control Parameters						
H06 Speed Control Parameters						
H07 Torque Control Parameters						
H08 Gain Parameters						
H09 Auto-adjusting Parameters						
H0A Fault and Protection						
H0B Display Parameters						
H0D Auxiliary Function Parameters						
H0E Communication parameters						
H3F Manufacturer Error code						
Axis_2						
Axis_3						

- “全部选择”：可以全部选择伺服功能码和“取消”全部选择项。
- “本页全选”：只针对当前页有效，全选和取消全选。
- “读取选中参数”：只有从站在运行状态时，且项目属性为读属性时，才可以读取。
- “写入选中参数”：只有从站在运行状态时，且项目属性为写属性时，才可以写入。
- “导出所选功能码”：导出选择的功能码。
- “导入功能码”：从外部导入功能码。

## ESC寄存器

“ESC寄存器”在勾选“专家模式”后才会显示，用于在高级调试中读取ESC芯片寄存器地址的值，如下图所示：

全选 取消全选 读取值 写入值 导出 导入						
选择项	地址	值	写入值	标志	长度(Byte)	描述
<input type="checkbox"/>	16#0000			R	2	Revision/Type
<input type="checkbox"/>	16#0002			R	2	Build
<input type="checkbox"/>	16#0004			R	2	SM/FMMU channels
<input type="checkbox"/>	16#0006			R	2	Ports Config/DPRAM Size
<input type="checkbox"/>	16#0008			R	2	Features
<input type="checkbox"/>	16#0010			RW	2	Configured Station Address
<input type="checkbox"/>	16#0012			R	2	Configured Station Alias
<input type="checkbox"/>	16#0020			RW	2	Register Write Protection/Enable
<input type="checkbox"/>	16#0030			RW	2	ESC Write Protection/Enable
<input type="checkbox"/>	16#0040			RW	1	ESC Reset ECAT
<input type="checkbox"/>	16#0041			R	1	ESC Reset PDI
<input type="checkbox"/>	16#0100			RW	2	ESC DL Control_L
<input type="checkbox"/>	16#0102			RW	2	ESC DL Control_H
<input type="checkbox"/>	16#0108			RW	2	Physical RW offset
<input type="checkbox"/>	16#0110			R	2	ESC DL Status
<input type="checkbox"/>	16#0120			RW	2	AL Control
<input type="checkbox"/>	16#0130			R	2	AL Status
<input type="checkbox"/>	16#0134			R	2	AL Status Code
<input type="checkbox"/>	16#0140			R	1	PDI Control
<input type="checkbox"/>	16#0141			R	1	ESC Config
<input type="checkbox"/>	16#0150			R	2	PDI Config_1
<input type="checkbox"/>	16#0152			R	2	PDI Config_2
<input type="checkbox"/>	16#0200			RW	2	ECAT IRQ Mask
<input type="checkbox"/>	16#0204			R	2	AL IRQ Mask_L
<input type="checkbox"/>	16#0206			R	2	AL IRQ Mask_H
<input type="checkbox"/>	16#0210			R	2	ECAT IRQ
<input type="checkbox"/>	16#0220			R	2	AL IRQ_L
<input type="checkbox"/>	16#0222			R	2	AL IRQ_H
<input type="checkbox"/>	16#0300			RW	2	RX Error Counter A
<input type="checkbox"/>	16#0302			RW	2	RX Error Counter B
<input type="checkbox"/>	16#0304			RW	2	RX Error Counter C
<input type="checkbox"/>	16#0306			RW	2	RX Error Counter D
<input type="checkbox"/>	16#0308			RW	2	Forwarded RX Error Counter B/A
<input type="checkbox"/>	16#030A			RW	2	Forwarded RX Error Counter D/C
<input type="checkbox"/>	16#030C			RW	1	ECAT Processing Unit Error Counter
<input type="checkbox"/>	16#030D			RW	1	PDI Error Counter
<input type="checkbox"/>	16#0310			RW	2	Lost Link Counter B/A
<input type="checkbox"/>	16#0312			RW	2	Lost Link Counter D/C

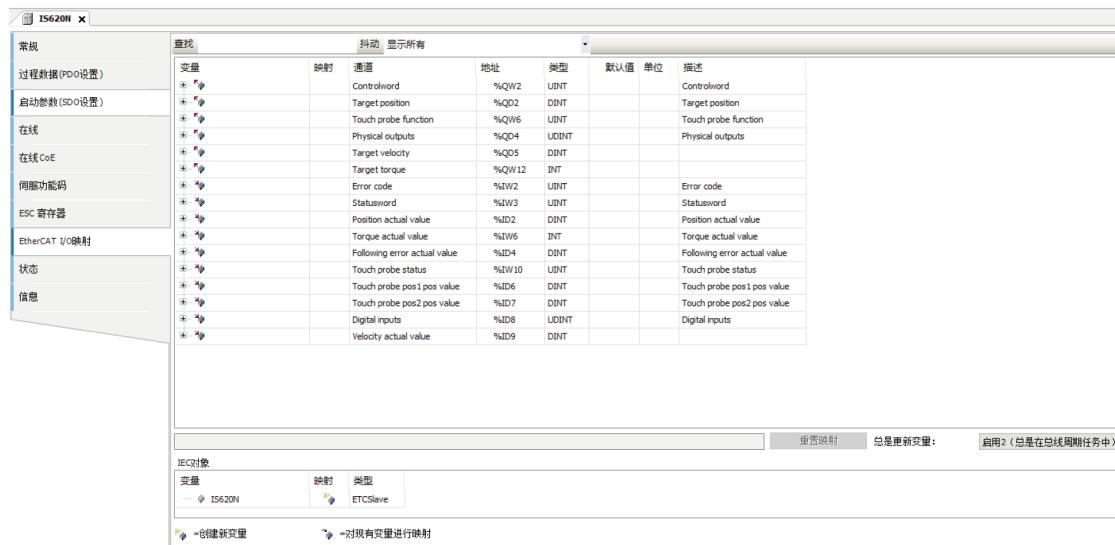
- “全选”：全部选择。
- “取消全选”取消全部的选择项。
- “读取值”：在运行状态下，可以读取选择项的值。
- “写入值”：在运行状态下，可以写入具有写入属性的值。
- “导出”：以XML格式导出选择的项。
- “导入”：导入外部符合格式要求的XML文件，只显示导出的XML项。
- 右键：可以进行十六进制、十进制和二进制切换。

## EtherCAT I/O 映射

EtherCAT从站配置编辑器中的选项卡，其中为EtherCAT I/O指定了ETCSlave类型的实例（变量）和从站定义的IO变量，因此连接到PLC的EtherCAT从站可以由用户程序控制。

有关如何映射的描述，请参见“I/O 映射”。

自动创建的从站实例显示在对话框的下部（IEC对象）主站实例，可以用于应用中



## 说明

映射的变量和类型要一致。

## 状态

这个配置编辑器表格用于EtherCAT从站配置，提供关于网卡和内部总线系统的状态信息（例如‘启动’‘停止’）以及特定设备的诊断信息。

## 信息

此对话框由 EtherCAT主站或从站配置对话框中提供。如果配置当前模块，将显示下列信息：名称、开发商、类型、版本号、分类、订单号、描述、图像等。

### 4.4.5 CiA402 轴

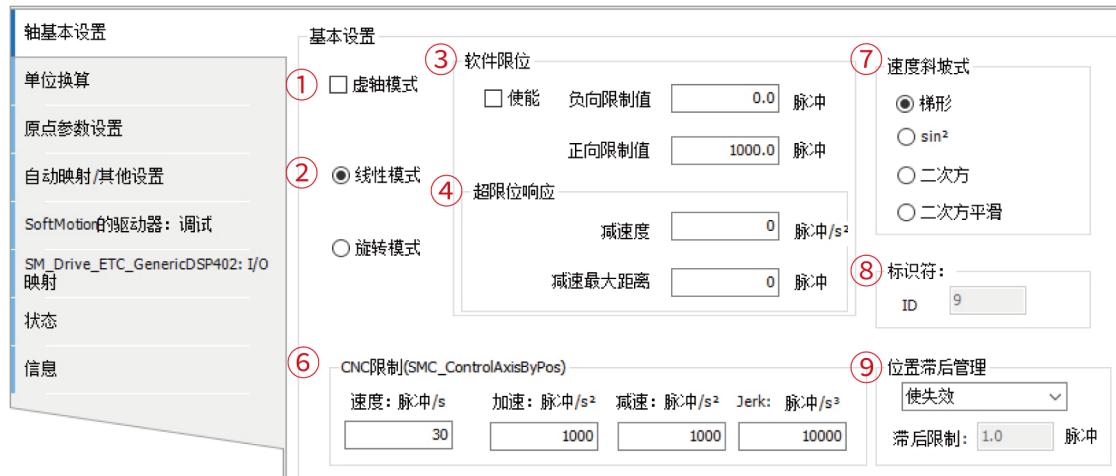
添加伺服从站后，双击“轴”会出现轴配置界面，下面按照由上至下的顺序依次对轴配置界面选项卡功能进行介绍。

#### 轴基本设置

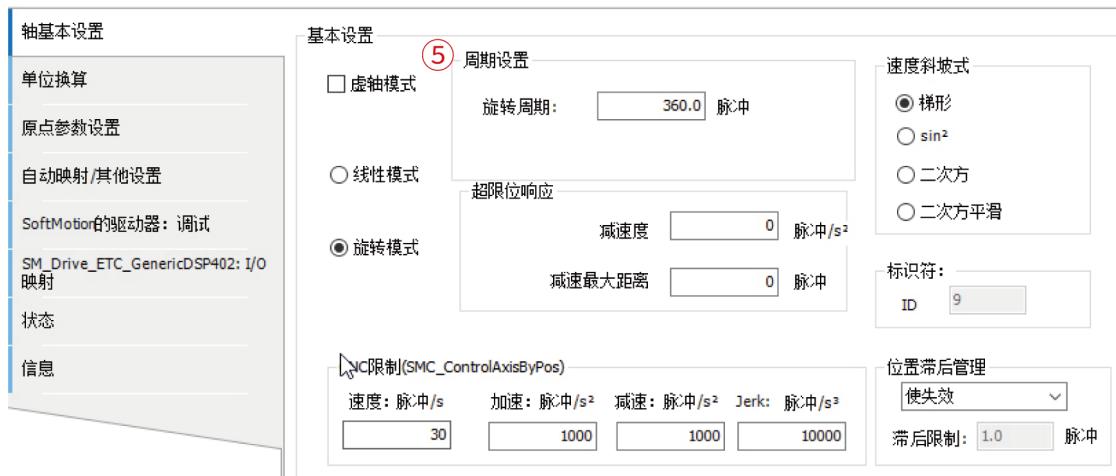
轴基本设置中包括虚轴/实轴模式两种工作模式，其定义如下表所述：

类别	功能描述
虚轴模式	虚轴模式，即可以不带物理伺服及电机运行的模式，可以模拟运行，得出自己需要的参数。不受外界环境的干扰。
实轴模式	实轴模式，必须带伺服电机运行，一些参数的获取必须在实轴模式才可以得到，如在线COE，实轴模式会有外界的干扰，如在TRACE监控过程中，会影响到显示效果。

虚轴模式和实轴模式设置



### 线性模式和旋转模式设置



图中的主要选项功能定义如下表所述：

编号	选项名称	功能说明
1	虚轴模式	勾选，则为虚轴模式；不勾选，则为实轴模式。
2	轴类型	线性模式：轴位置以线性方式增加或者减少； 旋转模式：轴位置在固定范围增加或者减少。
3	软件限位	勾选使能后，对轴负向和正向的位置限制，用于线性模式下。
4	超限位响应	与软件限位关联，使能软件限位的状态下才具有实际意义：勾选使能后，若轴位置参数超出软件限位设置，软件会对出错做出的响应，即发生错误后，驱动器在减速最大距离内停止。
5	周期设置	用于旋转模式下，对旋转周期进行限制。该参数与单位换算界面中电机旋转一圈的指令脉冲数、原点参数设置界面原点返回参数以及自动映射/其他设置界面最大速度相关联，设置时注意关联参数的设置，当关联参数值不匹配时，不匹配参数处会有警告提示该参数对应的匹配范围值。
6	CNC限制	主要用于有CNC功能轴的限制设置。
7	速度斜坡式	主要用于轴的速度变化轨迹。
8	标识符	轴对于外部ID。
9	位置滞后管理	位置滞后之后，管理轴的运行的方式。

下图为伺服启动后轴在线性模式下的界面，其中实时显示位置、速度、加速度、转矩以及状态通讯。如果有错误，则会显示错误信息。



## 说明

如果出现编码器位置信息丢失，请确认编码器电池是否接触可靠、电量充足。电池存储期间请按规定的环境温度进行存储。

## 绝对值编码器

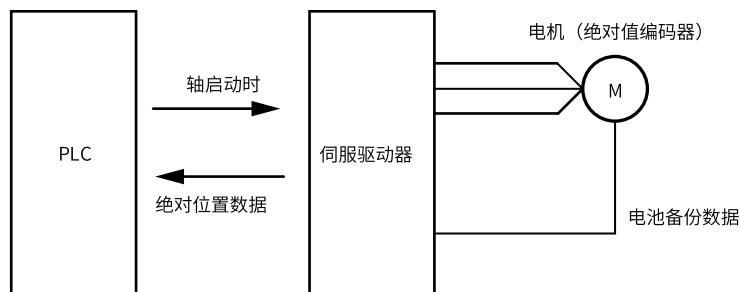


InoProShop V1.8.1.0及以上版本支持该功能

使用绝对值编码器时，无需重复进行原点复位操作，位置数据保存分为两种方式：位置数据保存在伺服端和位置数据保存在PLC端。搭配汇川伺服驱动器时，推荐使用位置数据保存在伺服端；搭配第三方伺服驱动器时，仅支持位置数据保存在PLC端。

- 位置数据保存在伺服端

切断PLC和伺服驱动器的电源后，伺服驱动器会通过内部的电池备份保存绝对位置数据。因此，再次启动时可以通过读取伺服的绝对位置数据确定编码器的位置，无需像增量编码器一样重复进行原点复位操作。

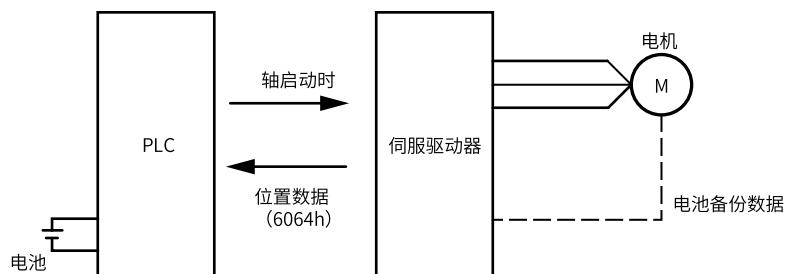


### 注意

- 请务必连接伺服驱动器的备份用电池，并切换伺服的绝对值系统选择（对象字典2002.02h）为3（绝对位置线性模式，无编码器溢出）。
- 因初次使用绝对值编码器、更换电机、伺服驱动器内部电池耗尽而丢失绝对值数据时，请务必执行“MC\_Home(原点复位)”指令以确定原点。
- 伺服断电后电机旋转圈数超出65535圈时，绝对值位置数据异常，请务必执行“MC\_Home(原点复位)”指令以确定原点。
- 通过MC\_Setposition功能块设置的轴位置偏移不做保存，需要用户自行处理。
- 触发MC\_HOME回原时设定的原点偏置超过了 $2^{32}$ 用户单位时，请务必执行“MC\_Home(原点复位)”指令以确定原点。
- 通过SMC\_ChangeGearingRatio等功能块修改轴的旋转/线性模式、旋转周期、单位换算等时，轴的位置会发生变化，请务必执行“MC\_Home(原点复位)”指令以确定原点。

- 位置数据保存在PLC端

切断PLC和伺服驱动器的电源后，位置信息会保存在PLC的掉电保持存储区。因此，再次启动时可以通过读取伺服的位置数据叠加PLC掉电保存的位置数据，最终确定编码器的位置，无需像增量编码器一样重复进行原点复位操作。



### 注意

- 请务必连接伺服驱动器的备份用电池，并切换伺服为绝对值模式，否则位置数据会发生异常。
- 因初次使用绝对值编码器、更换电机时，请务必执行“MC\_Home(原点复位)”指令以确定原点。
- 伺服断电后电机旋转超出 $2^{31}$ 用户单位时（23位电机为256圈），绝对值位置数据异常，请务必执行“MC\_HOME (原点复位)”指令以确定原点。
- 通过SMC\_ChangeGearingRatio等功能块修改轴的旋转/线性模式、旋转周期、单位换算等时，轴的位置会发生变化，请务必执行“MC\_Home(原点复位)”指令以确定原点。

可使用绝对值编码器的伺服驱动器型号如下表所示。

位置数据保存位置	伺服驱动器型号
位置数据保存在伺服端	<ul style="list-style-type: none"> <li>• ES680</li> <li>• ES810N</li> <li>• SV630</li> <li>• SV635</li> <li>• SV640</li> <li>• SV660</li> <li>• SV670</li> <li>• SV680</li> </ul>
位置数据保存在PLC端	所有伺服驱动器

### 操作步骤

- 位置数据保存在伺服端

- 连接绝对值编码器，具体请参见《XX系列伺服手册包》中绝对值编码器线连接内容部分。
- 设定绝对值系统，具体请参见《XX系列伺服手册包》中绝对值系统的设定内容部分。

以SV660N系列伺服驱动器为例，需设置电机编号2000.01h(H00.00)=14101（汇川23位绝对值编码器），设置绝对值系统选择2002.02h(H02.01)=3（绝对位置线性模式，无编码器溢出报警）。初次接通电池时会发生E731.0编码器电池故障，需设置200D.15h(H0d.20)=1复位编码器故障，再进行原点复归操作。

- 设定轴参数。

在“基本设置”界面中编码器类型选择“绝对编码器”，“位置数据保存在”选择“伺服端”。



- 执行原点复位。

程序中调用MC\_HOME执行原点复位。

- 位置数据保存在PLC端

- 连接绝对值编码器，具体请参见《XX系列伺服手册包》中绝对值编码器线连接内容部分。
- 设定绝对值系统，具体请参见《XX系列伺服手册包》中绝对值系统的设定内容部分。

以SV660N系列伺服驱动器为例，需设置电机编号2000.01h(H00.00)=14101（汇川23位绝对值编码器），设置绝对值系统选择2002.02h(H02.01)=3（绝对位置线性模式，无编码器溢出报警）。初次接通电池时会发生E731.0编码器电池故障，需设置200D.15h(H0d.20)=1复位编码器故障，再进行原点复归操作。

- 设定轴参数。

在“基本设置”界面中编码器类型选择“绝对编码器”，“位置数据保存在”选择“PLC端”。



#### 4. 执行原点复位。

程序中调用MC\_HOME执行原点复位。

#### 注意事项

满足以下任一条件时，请在轴运动前重新执行原点复位操作，否则轴可能发生意外动作。

位置数据保存位置	工况
位置数据保存在伺服端	<ul style="list-style-type: none"> <li>• 切换增量/绝对值编码器时</li> <li>• 切换位置数据保存在PLC端/伺服端方式时</li> <li>• 修改轴的旋转/线性模式、旋转周期、单位换算等时</li> <li>• 伺服不为绝对值模式或者电机不为绝对值编码器电机时</li> <li>• 修改伺服的旋转/线性模式、旋转方向、复位使能操作、电子齿轮比（6091.01h或6091.02h）时</li> <li>• 初次使用绝对值编码器、更换电机、伺服内部电池耗尽而丢失绝对值数据时</li> <li>• 伺服断电后电机旋转圈数超出65535圈时</li> <li>• 电机自机械原点的移动量超过了137438953472圈时</li> <li>• 伺服驱动器发生异常导致绝对位置信息丢失时</li> <li>• 触发MC_HOME回原时设定的原点偏置超过了<math>2^{32}</math>用户单位时</li> <li>• 线性模式下，轴的单方向运动量超出<math>32767 * 2^{32}</math>或<math>-32768 * 2^{32}</math>用户单位时</li> <li>• 发生其他异常情况时</li> </ul>
位置数据保存在PLC端	<ul style="list-style-type: none"> <li>• 切换增量/绝对值编码器时</li> <li>• 切换位置数据保存在PLC端/伺服端方式时</li> <li>• 修改轴的旋转/线性模式、旋转周期、单位换算等时</li> <li>• 伺服不为绝对值模式或者电机不为绝对值编码器电机时</li> <li>• 修改了伺服的旋转/线性模式、旋转方向、复位使能操作、电子齿轮比（6091.01h或6091.02h）时</li> <li>• 伺服断电后电机旋转超出<math>2^{31}</math>用户单位时（23位电机为256圈），绝对值位置数据异常，请务必执行“MC_HOME（原点复位）”指令以确定原点</li> <li>• 初次使用绝对值编码器、更换电机、PLC内部电池耗尽而丢失绝对值数据时</li> <li>• PLC掉电保持变量区发生异常或被清除时</li> <li>• 线性模式下，轴的单方向运动量超出<math>32767 * 2^{32}</math>或<math>-32768 * 2^{32}</math>用户单位时</li> <li>• 发生其他异常情况时</li> </ul>

## 单位换算

该界面主要功能是通过设置界面相关参数来计算脉冲数。

**用户单位**

① 脉冲  毫米  微米  纳米  度  英寸

**行程距离**

②  反向

③ 电机旋转一圈的指令脉冲数  指令脉冲/转

④ 不使用变速装置

⑤ 工作台旋转一圈的工作行程  脉冲/转

**参考：单位换算公式**

$$\text{脉冲数 (pulse)} = \frac{\text{电机旋转一圈的指令脉冲数 [DINT]}}{\text{工作台旋转一圈的工作行程 [LREAL]}} * \text{移动距离 [用户单位]}$$

⑥ 使用变速装置

⑦ 工作台旋转一圈的工作行程  脉冲/转  
(如果轴类型是旋转模式, 请参考轴基本设置界面的旋转周期值)

齿轮比分子 (下图中 (5) 的齿数)

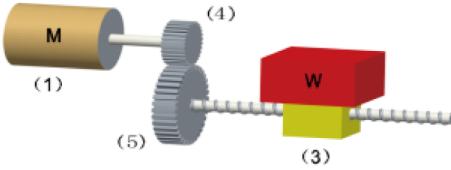
齿轮比分母 (下图中 (4) 的齿数)

**轴类型为线性模式**

**参考：单位换算公式**

$$\text{脉冲数 (pulse)} = \frac{\text{电机旋转一圈的指令脉冲数 [DINT]}}{\text{工作台旋转一圈的工作行程 [LREAL]}} * \frac{\text{齿轮比分子 [DINT]}}{\text{齿轮比分母 [DINT]}} * \text{移动距离 [用户单位]}$$

M:电机, W:工作台

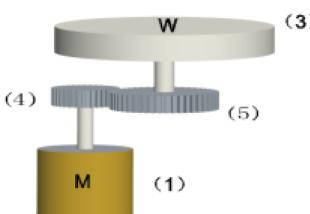


**轴类型为旋转模式**

**参考：单位换算公式**

$$\text{脉冲数 (pulse)} = \frac{\text{电机旋转一圈的指令脉冲数 [DINT]}}{\text{工作台旋转一圈的工作行程 [LREAL]}} * \frac{\text{齿轮比分母 [DINT]}}{\text{齿轮比分子 [DINT]}} * \text{移动距离 [用户单位]}$$

M:电机, W:工作台



上图中的主要选项及其功能如下表所述：

编号	选项名称	功能说明
1	用户单位	根据需要选择对应的长度单位，当单位选定后，轴基本设置、单位换算、原点参数设置以及自动映射/其他设置界面中应用的用户单位处均会作出相应改变。
2	反向	勾选后，轴以相反的方向旋转。
3	电机旋转一圈的指令脉冲数	电机的编码器分辨率，电机旋转一周需要的脉冲数量，默认值为1048576（汇川20位编码器）。
4	不使用变速装置	勾选后，可对工作台旋转一圈的工作行程根据设备情况进行设置。工作台旋转一圈的工作行程：工作台（如皮带轮、机械齿轮、减速机等）旋转一圈机械末端所移动的距离（用户单位）。可根据参考的单位换算公式计算脉冲数。
5	使用变速装置	勾选后，可对工作台旋转一圈的工作行程、齿轮比分子以及齿轮比分母根据设备情况进行设置。工作台旋转一圈的工作行程：工作台旋转一圈机械末端所移动的距离（用户单位）；齿轮比分子：工作台齿轮的齿数；齿轮比分母：电机齿轮的齿数。可根据轴基本设置界面的勾选轴类型以及对应的参考单位换算公式计算脉冲数。注：齿轮比分子和分母可进行等比例的缩放。

### 应用示例

1. 电机直接驱动丝杆移动，电机旋转一圈丝杆移动10mm，电机选择汇川IS620N增量式电机（编码器分辨率为20位），配置如下所示：



2. 电机间接驱动转盘，电机与转盘之间存在减速比为30: 1变速装置（若电机齿轮齿数为1，工作台齿轮齿数为30，即工作台齿轮旋转1圈，电机齿轮旋转30圈），转盘的行程为0-360度，电机选择汇川IS620N绝对值电机（编码器分辨率为23位），配置如下所示：

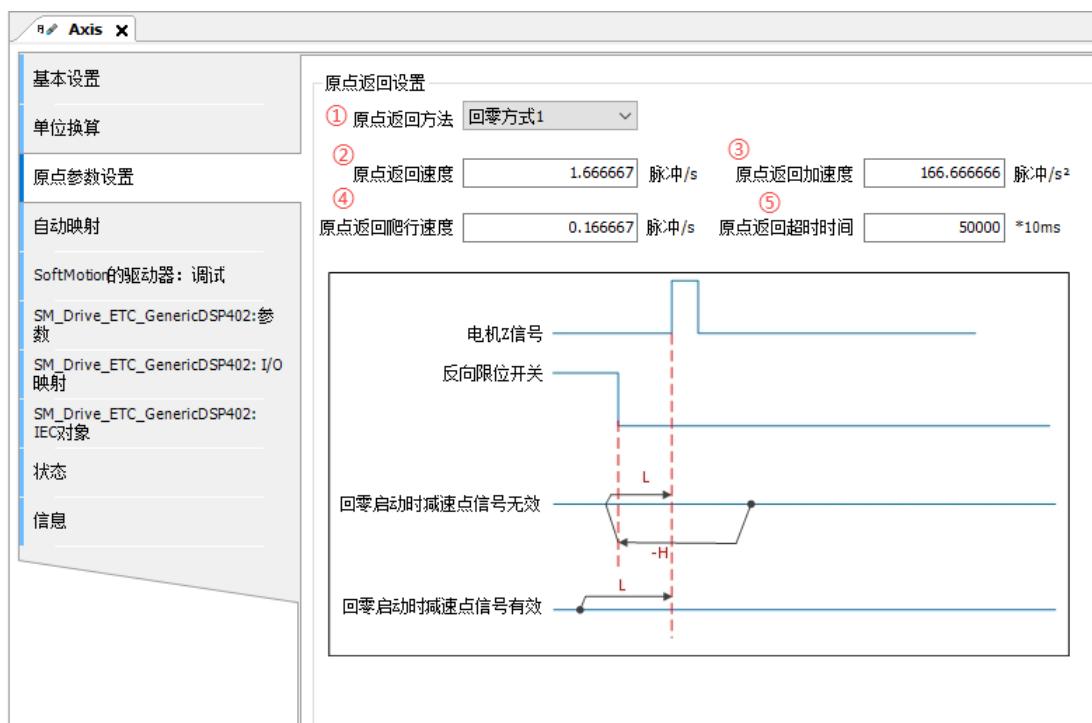


## 说明

修改齿轮比分子、齿轮比分母及工作台旋转一圈的工作行程后，会影响到轴其他界面参数，请调整其他界面参数。

## 原点参数设置

原点参数设置，主要用于轴回零图形化参数配置，如下图所示。提供了图形化配置指导，无需另外查阅伺服手册便可通过配置界面中的下拉菜单直接选择所需的回零模式，方便用户更加直观、便捷地完成参数配置过程，如下图中①所示。



图中的主要选项及其功能如下表所述：

编号	选项名称	说明
1	原点返回方法	配置驱动器回原点的方式，总共支持35种选项（实际的支持回零方式由驱动器决定）。每种不同的回零方式，下边的示例图会有所不同，根据需要选择不同的回零方式。
2	原点返回速度	回零时，驱动轴搜索减速速度点信号的高速速度值，如图中回零时的H。浮点数，最小保留6位小数，兼容逗号作为小数点的系统。
3	原点返回加速度	回零时，驱动轴搜索速度变化的加速度。浮点数，最小保留6位小数，兼容逗号作为小数点的系统。
4	原点返回爬行速度	回零时，驱动轴搜索原点时的低速速度值，如图中回零时的L。浮点数，最小保留6位小数，兼容逗号作为小数点的系统。
5	原点返回超时时间	限定原点回零总时间，超时则发生警告，驱动轴执行回零流程最大允许时间，如果回零超时，驱动轴回零失败。

## 自动映射/其他设置

The screenshot shows the 'Axis Basic Settings' software interface. On the left, there is a sidebar with the following options:

- 轴基本设置
- 单位换算
- 原点参数设置
- 自动映射/其他设置
- SoftMotion的驱动器: 调试
- SM\_Drive\_ETC\_GenericDSP402: I/O 映射
- 状态
- 信息

The main area has two tabs highlighted with red circles:

- ① 其他设置**: This tab contains settings for torque and speed limits. It includes fields for '正扭矩最大值' (Max Positive Torque) set to 5000, '负扭矩最大值' (Max Negative Torque) set to 5000, '0.1%' (likely a multiplier), '最大速度' (Max Speed) set to 100, and '脉冲/s' (Pulses per second).
- ② 映射**: This tab is for mapping I/O signals. It has two sections: '输入:' (Inputs) and '输出:' (Outputs). Both sections show tables of mapped objects, addresses, and types.

**Input Mapping (映射 - 输入):**

循环对象	对象	地址	类型
status word (in.wStatusWord)	16#6041:16#00	%IW1	UINT
actual position (diActPosition)	16#6064:16#00	%ID1	DINT
actual velocity (diActVelocity)	16#606C:16#00		
actual torque (wActTorque)	16#6077:16#00	%IW4	INT
Modes of operation display (OP)	16#6061:16#00		
digital inputs (in.dwDigitalInputs)	16#60FD:16#00	%ID7	UDINT
Touch Probe Status	16#60B9:16#00	%IW8	UINT
Touch Probe 1 rising edge	16#60BA:16#00	%ID5	DINT
Touch Probe 1 falling edge	16#60BB:16#00		
Touch Probe 2 rising edge	16#60BC:16#00	%ID6	DINT
Touch Probe 2 falling edge	16#60BD:16#00		

**Output Mapping (映射 - 输出):**

循环对象	对象	地址	类型
ControlWord (out.wControlWord)	16#6040:16#00	%QW0	UINT
set position (diSetPosition)	16#607A:16#00	%QD1	DINT
set velocity (diSetVelocity)	16#60FF:16#00		
set torque (wSetTorque)	16#6071:16#00		
Modes of operation (OP)	16#6060:16#00		
Touch Probe Function	16#60B8:16#00	%QW4	UINT
Add velocity value	16#60B1:16#00		
Add torque value	16#60B3:16#00		

图中的主要选项及其功能如下表所述：

编号	选项名称	说明
1	其他设置	<p>可对正/负扭矩最大值以及最大速度进行设置。</p> <p>正/负扭矩最大值：为保护驱动器而设定的转矩指令限制值，当驱动器转矩指令大于转矩指令限制值，则实际驱动器的转矩指令被限幅等于转矩指令限制值。</p> <p>最大速度：防止给定转矩指令过大，大于机械负载转矩，导致电机持续加速，可发生超速现象，损坏机械设备；设定速度限制后，实际转速将限制在速度限制值以内。</p> <p>浮点数，最小保留6位小数，兼容逗号作为小数点的系统。</p> <p>注：最大速度一般不可以设置为0，如果为0，轴运行可能出错。</p>
2	映射	<p>当自动映射勾选时，从站与轴之间存在关联，从站数据直接映射给轴；不勾选时，可以手动对轴映射数据中的地址进行修改。其中：</p> <p>输入格式为%I+类型对应的字母+阿拉伯数字。</p> <p>输出格式为%Q+类型对应的字母+阿拉伯数字。</p> <p>类型对应的字母（类型所占用字节）SINT-B、UINT-W、DINT-D、UDINT-D。</p> <p>当手动输入地址导致编译报错时，需要删除输入地址，重新输入正确地址。</p> <p>注：当输入地址时加单引号，编译报错但显示结果正常，此时应删除显示结果，重新输入正确格式方可解决编译报错问题。</p>

备注：需要手动对映射数据进行修改时，在勾选自动映射的情况下，可以通过鼠标右键或者键盘Delete键对目标PDO进行删除。不慎删除错误，可以通过恢复默认映射表恢复。



## 信息

该界面显示轴的基本信息，包括名称、供应商、组、类型、ID、版本、模块号、说明。



若轴参数没有改动的话， PDO和SDO界面参数保持不变。若轴参数有改动，则 PDO和SDO界面参数也对应自动变化。

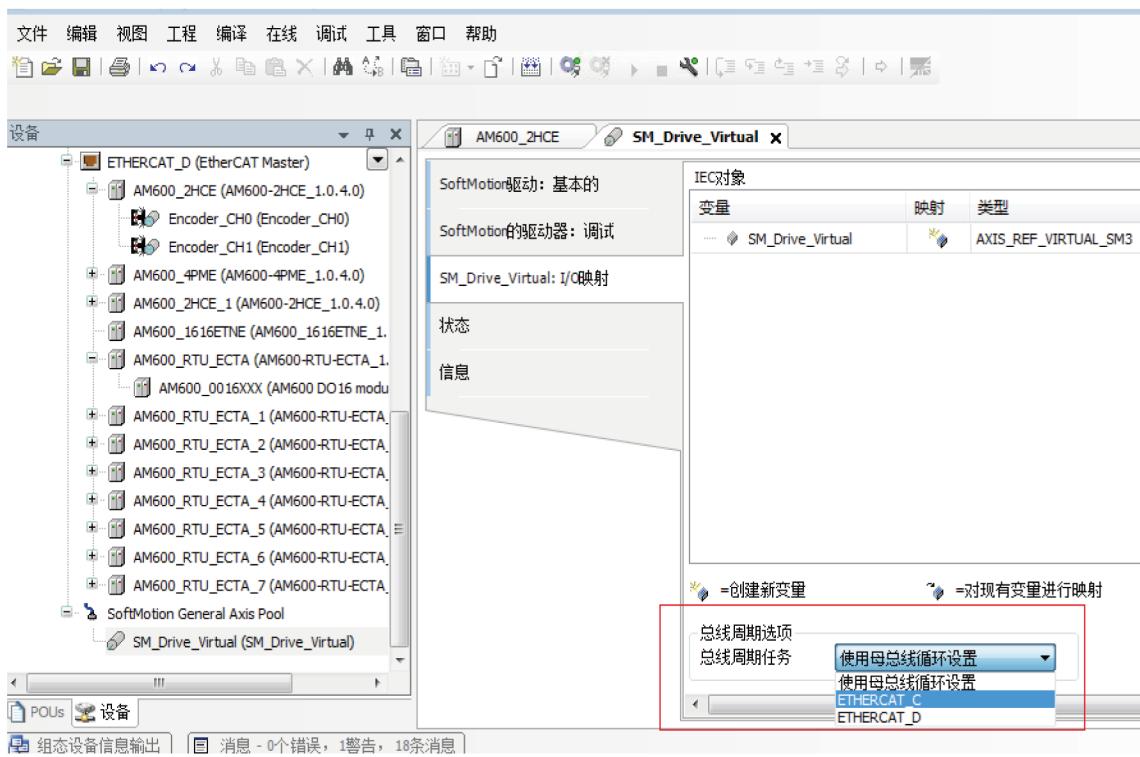
The screenshot shows two main windows of the IS620N software:

- Top Window (Network Configuration):** This window displays the "过程数据(PDO设置)" (Process Data PDO Settings) tab. It lists PDO mappings for both receive and transmit. For receive PDOs, it includes entries like "1st receive PDO Mapping" with fields such as索引 (Index) 16#1600, 子索引 (Subindex) 16#00, 长度 (Length) 18.0, and 类型 (Type) INT. For transmit PDOs, it includes "1st transmit PDO Mapping" with similar fields. The right side of the window shows summary statistics: 输出(字节): 18.0 and 输入(字节): 32.0.
- Bottom Window (System Parameters):** This window displays the "常规" (General) tab under "过程数据" (Process Data). It lists various system parameters with their values, such as "Modes of operation" (Value: 8), "Homing method" (Value: 26), "Speed during search for switch" (Value: 2796203), etc. Each row has checkboxes for "是否下载" (Download) and "有错退出" (Error Exit).

## 4.4.6 虚轴

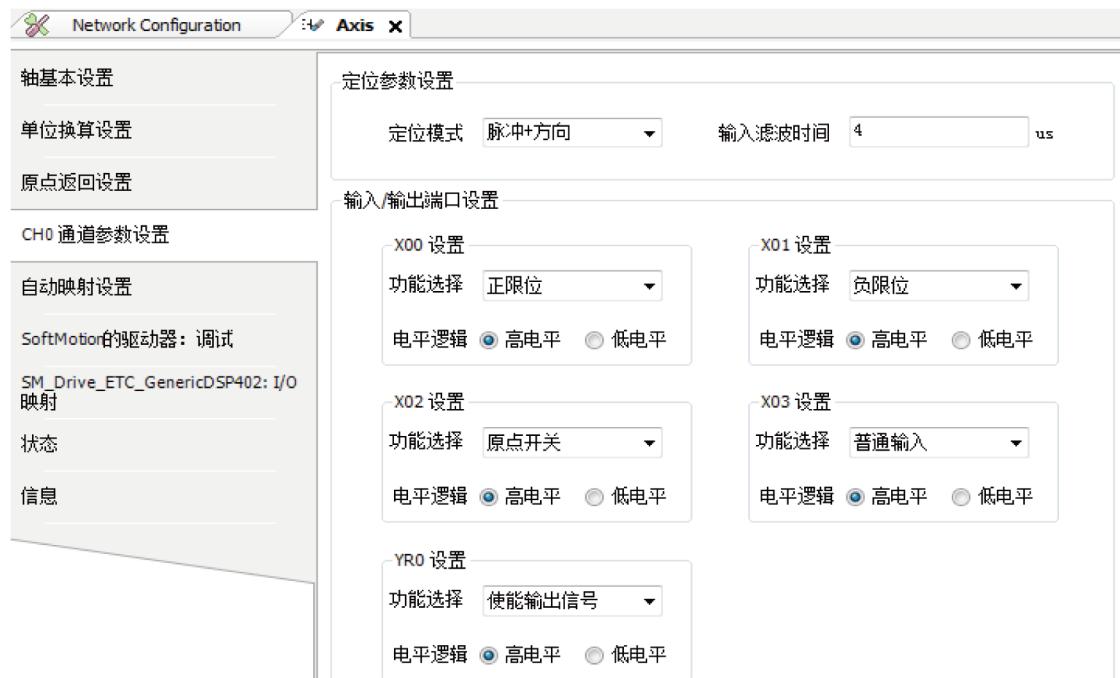
虚轴的界面与EtherCAT配置界面内容大致相同，需要注意的是，在同时启用多个EtherCAT主站并且使用虚轴时（从【轴池】 - 【虚轴】添加），虚轴的“总线周期任务”必须绑定到被调用的EtherCAT任务下（默认“使用母总线循环设置”，只适用单个主站情况），不允许同一个虚轴在多个EtherCAT任务下调用，否则会出现运行异常。

下图红色框内是虚轴的“总线周期任务”配置。



#### 4.4.7 GR10-4PME 定位模块

定位模块界面上除了增加如下界面，其他界面与CiA402轴设置界面相同。



GR10-4PME模块是一款具有4个高速输出通道的脉冲定位模块，可实现脉冲型伺服、步进等以脉冲为给定信号的驱动器的速度和位置控制。该界面用于设置GR10-4PME模块的一些配置参数，以GR10-4PME模块第一通道为例，在该界面可配置如下功能：

名称	描述	默认设置
定位模式	选择设置高速脉冲输出口输出的脉冲类型 AB相1倍频 脉冲+方向 CW/CCW	脉冲+方向
输入滤波时间	数字量输入端子的脉冲输入滤波时间	4us
X00设置	功能选择  普通输入 急停开关 正限位	X00数字量输入端子功能选择  正限位
	电平逻辑  高电平-输入高电平为有效电平 低电平-输入低电平为有效电平	X00数字量输入有效电平逻辑选择  高电平 低电平
X01设置 电平逻辑	功能选择  普通输入 急停开关 负限位	X01数字量输入端子功能选择  负限位
	X01数字量输入有效电平逻辑选择  高电平-输入高电平为有效电平 低电平-输入低电平为有效电平	高电平
X02设置 电平逻辑	功能选择  普通输入 急停开关 原点开关	X02数字量输入端子功能选择  原点开关
	X02数字量输入有效电平逻辑选择  高电平-输入高电平为有效电平 低电平-输入低电平为有效电平	高电平
X03设置 电平逻辑	功能选择  普通输入 急停开关	X03数字量输入端子功能选择  普通输入
	X03数字量输入有效电平逻辑选择  高电平-输入高电平为有效电平 低电平-输入低电平为有效电平	高电平
YR0设置 电平逻辑	功能选择  普通数字量输出端子 伺服使能 (输出信号)	数字量输出端子YR0功能设置  伺服使能 (输出信号)
	数字量输出端子YR0导通关断设置  高电平-控制给定为1时导通 低电平-控制给定为0时导通	高电平

有关定位模块的具体应用，请参见《EtherCAT远程通信应用手册》。

#### 4.4.8 GR10-2HCE 计数模块

计数模块除了增加如下界面，其他界面与CiA402轴设置界面相同。



GR10-2HCE模块是一款具有2个高速输入通道的脉冲计数模块，可实现AB相脉冲、脉冲+方向、CW/CCW形式的脉冲计数与测频。该界面用于设置GR10-2HCE模块的一些配置参数，以GR10-2HCE模块第一通道为例，该界面可配置如下功能：

名称	描述	默认设置
计数模式	通道输入脉冲输入方式选择	AB相4倍频
	AB相1倍频	
	AB相2倍频	
	AB相4倍频	
	脉冲+方向	
	CW/CCW	
频率采样周期	输入滤波频率计算采样周期	10ms
输入滤波时间	脉冲输入通道和数字量输入通道采样滤波	2us

名称		描述	默认设置
计数方向	A相超前	AB相	A相超前
		当A相超前B相时计数器增加	
		脉冲+方向	
		B相输入高电平时计数器增加	
		CW/CCW	
	B相超前	A相有计数则计数器增加	
		AB相	
		当B相超前A相时计数器增加	
		脉冲+方向	
		B相输入低电平时计数器增加	
X00设置	功能选择	CW/CCW	探针1
		B相有计数则计数器增加	
		X00数字量输入端子功能选择	
		普通输入	
		探针1	
		计数器清0	
	电平逻辑	计数器预置	高电平
		门控	
		X00数字量输入有效电平逻辑选择	
		高电平-输入高电平为有效电平	
X01设置 电平逻辑	功能选择	低电平-输入低电平为有效电平	探针2
		X01数字量输入端子功能选择	
		普通输入	
		探针2	
		计数器清0	
		计数器预置	
	X01数字量输入有效电平逻辑选择	门控	高电平
		高电平-输入高电平为有效电平	
		低电平-输入低电平为有效电平	
		X02数字量输入端子功能选择	
X02设置 电平逻辑	功能选择	普通输入	普通输入
		计数器清0	
		计数器预置	
		门控	
	X02数字量输入有效电平逻辑选择	X02数字量输入有效电平逻辑选择	
		高电平-输入高电平为有效电平	高电平
		低电平-输入低电平为有效电平	
		X02数字量输入有效电平逻辑选择	
		高电平-输入高电平为有效电平	

名称		描述	默认设置
X03设置 电平逻辑	功能选择	X03数字量输入端子功能选择	普通输入
		普通输入	
		计数器清0	
		计数器预置	
		门控	
	X03数字量输入有效电平逻辑选择	高电平	
		高电平-输入高电平为有效电平	
		低电平-输入低电平为有效电平	
Y00设置 电平逻辑	功能选择	数字量输出端子Y00功能设置	比较输出1
		普通输出	
		比较输出1	
	数字量输出端子Y00导通关断设置	高电平	
		高电平-控制给定为1时导通	
		低电平-控制给定为0时导通	
Y01设置 电平逻辑	功能选择	数字量输出端子Y01功能设置	比较输出2
		普通输出	
		比较输出2	
	数字量输出端子Y01导通关断设置	高电平	
		高电平-控制给定为1时导通	
		低电平-控制给定为0时导通	
Y02设置 电平逻辑	功能选择	数字量输出端子Y02功能设置	普通输出
		普通输出	
	数字量输出端子Y02导通关断设置	高电平	
		高电平-控制给定为1时导通	
		低电平-控制给定为0时导通	

有关定位模块的具体应用，请参见《EtherCAT 远程通信应用手册》。

#### 4.4.9 分支器

##### 分支器简介

分支器模块用于扩展EtherCAT端口，如下图所示：

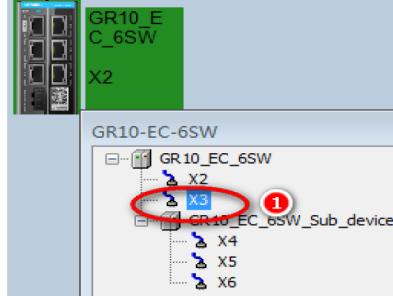


接口	选项名称	功能说明
IN1	分支器入口	分支器入口连接设备EtherCAT出口。
X2~X6	分支器出口	X2~X6 都为分支器出口，各个出口之间互不影响，每路出口可以连接一路 EtherCAT从站设备。

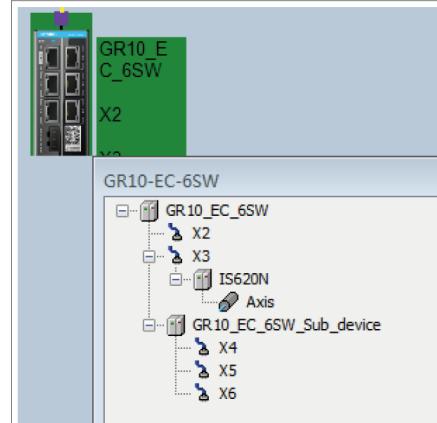
## 添加分支器及其下从站设备

分支器设备的添加，删除，复制，粘贴和普通从站设备一样（见[第87页“设置PLC作为主站或从站设备”](#)及[第91页“组态设备的常用操作”](#)），添加完分支器设备后，添加分支器下从站需按如下步骤操作：

1. 双击组态分支器设备，弹出如下图所示框：



2. 先选择与实际物理组态一致的合适的节点，再选择网络设备列表设备，即可完成如下图所示分支器从站组态：



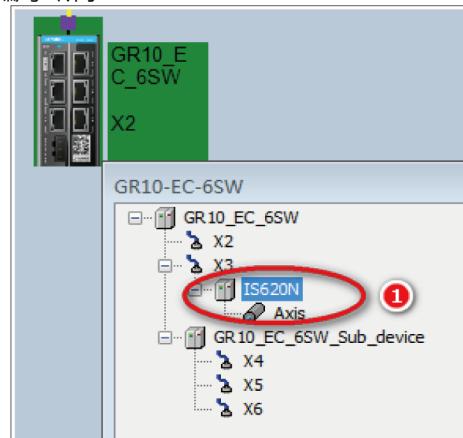
也可以使用扫描设备命令（请参见[第148页“扫描设备”](#)），扫描分支器及其下从站设备，复制到工程中，自动完成设备的组态。

## 删除分支器及其下从站设备

删除分支器设备和从站设备删除相同（请参见第91页“组态设备的常用操作”）。

删除分支器下从站设备的操作如下：

1. 打开分支器配置界面。
2. 选择需要删除的从站，如下图编号1所示：



按“删除”（Delete）按键，即可删除从站设备。

### 说明

删除后，不支持撤销和恢复操作，需要重新增加设备组态。

## 4.4.10 IO 模块

### 添加IO模块

通信接口模块包括AM600-RTU-ECTA、GL10-RTU-ECTA、GL20-RTU-ECT和GL20(S)-RTU-ECT32，添加IO模块操作方法相同，以GL20-RTU-ECT32通信接口模块操作为例。

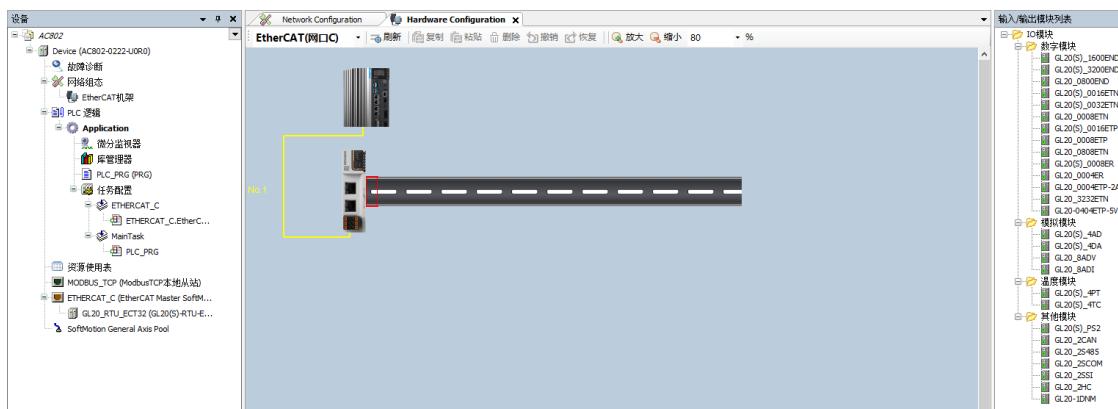
1. 使能主机EtherCAT主站并添加GL20-RTU-ECT32通信接口模块。

- a. 在左侧设备树中双击“网络组态”，勾选与GL20-RTU-ECT32通信接口模块连接的主机实际物理网口对应的“EtherCAT主站”，使能主机EtherCAT主站。
- b. 在右侧网络设备列表中，双击“GL20(S)-RTU-ECT32\_x.x.x.x”，添加GL20-RTU-ECT32通信接口模块。



2. 添加模块。

在左侧设备树中双击“EtherCAT机架”，在右侧模块列表中双击XX模块，添加XX模块。



## 说明

另一种添加模块方法：在左侧设备树“ETHERCAT（EtherCAT Master SoftMotion）”下右键单击“GL20\_RTU\_ECT32”，在右键菜单中选择“添加设备”，在打开的界面中选择“GL20-XX”，单击“添加设备”。

3. 添加完成后配置模块相应的参数，具体操作请参见各模块《用户手册》。

## 说明

GL10系列模块的配置操作，具体请参见《EtherCAT远程通信应用手册》。

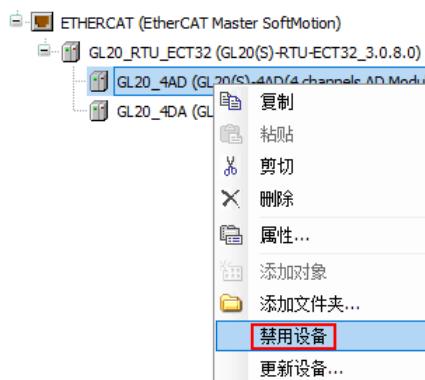
## 禁用GL20系列IO模块



注意

InoProShop V1.7.3 SP4及以上和GL20-RTU-ECT32通信接口模块的XML版本2.0.2.0及以上版本支持该功能（GL20-RTU-ECT32通信接口模块不支持）。

右键单击GL20系列IO模块，在右键菜单中选择“禁用设备”，禁用GL20系列IO模块，以GL20-4AD模块为例，如下图所示。



GL20系列IO模块禁用功能通过对象字典来控制，可通过“在线CoE”进行查看，如下图所示。

GL20_RTU_ECT					
常规	阅读这一页		自动上传	来自ESI的脱机文件	在线设备
过程数据(PDO设置)	索引: 子索引	名称	标志	类型	值
	16#1000:16#00	Device type	RO	UDINT	
	16#1001:16#00	Error Register	RO	USINT	
	16#1008:16#00	Device Name	RO	STRING(15)	
	16#100A:16#00	Software version	RO	STRING(13)	
启动参数(SDO设置)	16#1018:16#00	Identity	RO	USINT	
	16#1C00:16#00	Sync manager type	RO	USINT	
EtherCAT参数	16#1C12:16#00	RxPDO assign	RO	USINT	
	16#1C13:16#00	TxDPO assign	RO	USINT	
设备诊断	16#1C32:16#00	SM output parameter	RO	USINT	
EtherCAT I/O映射	16#1C33:16#00	SM input parameter	RO	USINT	
EtherCAT IEC对象	16#3010:16#00	Port 0 error counter	RO	USINT	
	16#3011:16#00	Port 1 error counter	RO	USINT	
状态	16#3012:16#00	ESC error counter	RO	USINT	
	16#3016:16#00	Station address	RO	USINT	
	16#3020:16#00	Fpga soft version	RO	UDINT	
信息	16#3021:16#00	Module software version	RO	USINT	
	16#5000:16#00	Disable Slot Control	RW	USINT	
	:16#01	Disable Slot Control Ch0	RW	UINT	
	:16#02	Disable Slot Control Ch1	RW	UINT	
	:16#03	Disable Slot Control Ch2	RW	UINT	
	:16#04	Disable Slot Control Ch3	RW	UINT	
	16#5001:16#00	Disable Function Control	RW	UINT	
	16#6000:16#00	4AD input	RO	USINT	
	16#6100:16#00	1600END input	RO	USINT	
	16#6140:16#00	0800END input	RO	USINT	
	16#7040:16#00	4DA output	RO	USINT	
	16#7080:16#00	0008ETP output	RO	USINT	
	16#70C0:16#00	0016ETN output	RO	USINT	

对象字典如下表所示。

对象字典	数据类型	功能描述							
16#5001:16#00	UINT	<p>用于标记是否存在禁用模块设备。</p> <ul style="list-style-type: none"> <li>• 值为0：不存在禁用模块设备。</li> <li>• 值为1：存在禁用模块设备。</li> </ul>							
16#5000:16#01	UINT	<p>用于标记1~16槽位模块禁用状态，每个bit位对应一个模块。</p> <p>Bit位与槽位对应关系为：</p> <table border="1"> <tr> <td>槽16</td><td>槽15</td><td>槽14</td><td>.....</td><td>槽3</td><td>槽2</td><td>槽1</td> </tr> </table> <ul style="list-style-type: none"> <li>• Bit位值为0：该槽位模块未禁用。</li> <li>• Bit位值为1：该槽位模块被禁用。</li> </ul>	槽16	槽15	槽14	.....	槽3	槽2	槽1
槽16	槽15	槽14	.....	槽3	槽2	槽1			
16#5000:16#02	UINT	<p>用于标记17~32槽位模块禁用状态，每个bit位对应一个模块。</p> <p>Bit位与槽位对应关系为：</p> <table border="1"> <tr> <td>槽32</td><td>槽31</td><td>槽30</td><td>.....</td><td>槽19</td><td>槽18</td><td>槽17</td> </tr> </table> <ul style="list-style-type: none"> <li>• Bit位值为0：该槽位模块未禁用。</li> <li>• Bit位值为1：该槽位模块被禁用。</li> </ul>	槽32	槽31	槽30	.....	槽19	槽18	槽17
槽32	槽31	槽30	.....	槽19	槽18	槽17			
16#5000:16#03	UINT	<p>用于标记33~48槽位模块禁用状态，每个bit位对应一个模块。</p> <p>Bit位与槽位对应关系为：</p> <table border="1"> <tr> <td>槽48</td><td>槽47</td><td>槽46</td><td>.....</td><td>槽35</td><td>槽34</td><td>槽33</td> </tr> </table> <ul style="list-style-type: none"> <li>• Bit位值为0：该槽位模块未禁用。</li> <li>• Bit位值为1：该槽位模块被禁用。</li> </ul>	槽48	槽47	槽46	.....	槽35	槽34	槽33
槽48	槽47	槽46	.....	槽35	槽34	槽33			
16#5000:16#04	UINT	<p>用于标记49~64槽位模块禁用状态，每个bit位对应一个模块。</p> <p>Bit位与槽位对应关系为：</p> <table border="1"> <tr> <td>槽64</td><td>槽63</td><td>槽62</td><td>.....</td><td>槽51</td><td>槽50</td><td>槽49</td> </tr> </table> <ul style="list-style-type: none"> <li>• Bit位值为0：该槽位模块未禁用。</li> <li>• Bit位值为1：该槽位模块被禁用。</li> </ul>	槽64	槽63	槽62	.....	槽51	槽50	槽49
槽64	槽63	槽62	.....	槽51	槽50	槽49			

#### 4.4.11 库 (隐含变量)

##### 主站的隐式实例

EtherCAT 主站只要被插入到设备列表中，将建立一个 ‘IoDrvEtherCAT’ 类型的隐式实例。实例名称与设备列表中使用的设备名称完全相同。

##### 说明

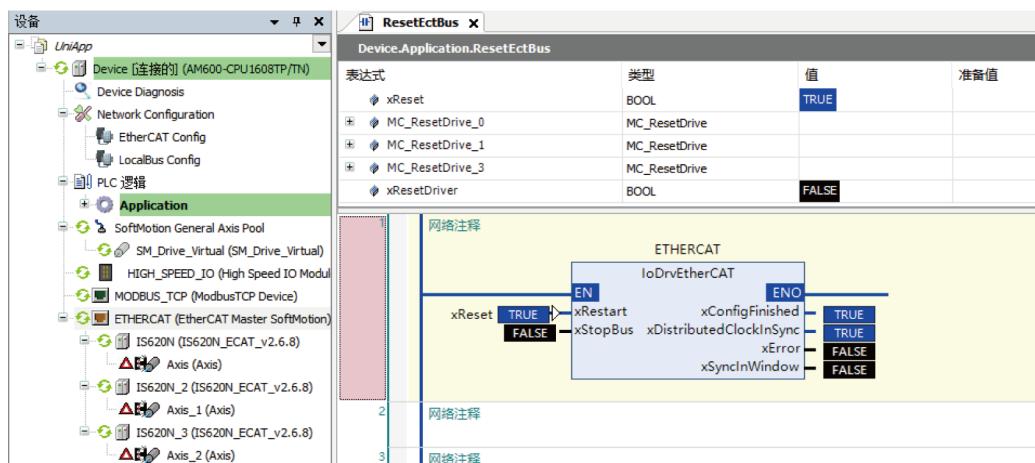
隐式实例是由系统自动生成，不可在程序中定义，否则会导致PLC运行异常。

IoDrvEtherCAT类型的隐式实例定义如下表所述：

IoDrvEtherCAT隐式实例	
输入参数	定义
xRestart	总线重启：上升沿时，当前主站会重启，所有配置参数会重新加载。
xStopBus	总线停止：电平触发，当输入值为“TRUE”时，EtherCAT总线通信将会停止，通讯进入错误状态，停止后若要继续使用，必须通过“xRestart”重启EtherCAT通信。
输出参数	定义
xConfigFinished	如果这个参数为“TRUE”，所有配置参数的传送已经正确完成。通讯正在运行。
xDistributedClockInSync	如果EtherCAT从站配置了DC(分布时钟)，总线启动时首先会配置EtherCAT从站参数，待参数配置完毕（“xConfigFinished”为“TRUE”），开始调整从站时钟以及主站的时钟。当主站和从站时钟同步成功后，输出“TRUE”；正常运行后总线出现故障丢失同步，则输出“FALSE”。DC模式下，必须等到这个参数为“TRUE”再启动运动控制功能块，否则，运动轴的位置可能会产生跳跃。
xError	如果EtherCAT主站启动时检测到错误，或者从站通讯状态机到达操作模式后通讯被中断，则该输出为TRUE，因为EtherCAT主站再收不到任何消息（比如由于连线中断）。此时，主站诊断或者日志信息定位错误原因。

##### 示例

示例程序如下：该功能块的名称为IoDrvEtherCAT，主站名ETHERCAT（AM600默认）即为其实例化名称，不需要再在程序里面实例化声明。AC800系列主站名默认为ETHERCAT\_C或者ETHERCAT\_D。



如图xRestart输入变量由FALSE变化TRUE，上升沿触发主站重启，xStopBus为True 电平触发总线通讯停止，通过输出参数状态判断总线通讯信息。

### 主站属性

属性	定义
AutoSetOperational	如果此属性被设置为TRUE，一旦通讯中止，主站总是会尝试重启从站。 默认值： FALSE
ConfigRead	若此属性返回TRUE，表明已完成配置的读取，用户可以编辑设置。例如可实现自定义SDOs的添加。
DCInSyncWindow	XDistributedClockInSync置TURE的时间窗口条件。主站的同步抖动必须在此窗口范围内，XDistributedClockInSync的值才为TRUE。 默认值： 50 微秒
DCIntegralDivider	分布时钟控制回路的积分因子。 默认值： 20
DCPropFactor	分布时钟控制回路的比例因。 默认值： 25
DCSyncToMaster	分布时钟与主站的同步性。如果设为TRUE，所有从站都会与主站同步，而不是第一个从站与PLC同步。 默认值： FALSE
DCSyncToMasterWithSysTime	与主站的分布时钟同步。如果设置为 TRUE 所有从站都将会与主站系统时间同步。通过 SysTimeRtcHighResGet 读取的时间也可以用于PLC到所有EtherCAT从站直接的同步。 默认值: FALSE
EnableTaskOutputMessage	EtherCAT消息通常都是由总线周期任务发送，额外也会由每个使用从站输出的任务发送。在总线周期任务中，会写入所有的输出，并读取所有的输入。在其他任务中，输出会多传输一次，以使他们立刻被写入各自的从站。因此要尽可能的缩短截止时间直到写的足够快。在有分布时钟协调时，这可能导致某些从站出现问题，例如伺服控制器与同步中断不同步，却对内部同步器使用写的时间点。此情况下，一个循环内可能出现多个写访问。如果 EnableTaskOutputMessage设为FALSE，则只会使用总线周期任务，额外任务不会再影响消息。 默认值： TRUE
FirstSlave	主站下第一个从站的指针。
FrameAtTaskStart	如果FrameAtTaskStart被设为TRUE，则给从站的帧内容将在任务开始被发送，以保证最小的抖动。此命令用来得到伺服驱动器的平滑运动。如果此标志位被设为TRUE，则输出缓冲帧被写入下一个循环中。 默认值： FALSE
LastInstance	关联主站列表的指针->上一个主站。
LastMessage	此属性与EtherCAT栈最新消息一起返回一个字符串，如果成功完成启动，则应该返回‘完成所有从站’。字符串的作用，同在线模式下EtherCAT主站设备编辑器中显示的诊断信息一样。
NextInstance	关联主站列表的指针->下一个主站。
NumberActiveSlaves	此属性返回实际连接从站的数量。如果StartConfigWithLessDevice设为TRUE,则可以确定实际设备的数量。
OpenTimeout	打开网管超时时间，默认为 4 秒。
StartConfigWithLessDevice	此属性可以用来影响堆栈启动行为。例如组态中配置了五个伺服控制器，但只连接了三个的情况，则堆栈会马上停止，总线配置失败，PLC报错。但是如果在第一个周期中 StartConfigWithLessDevice设为TRUE，则堆栈会尝试启动，总线正常配置运行，PLC不报错。因此，如果建立了一个10个伺服控制器的通用配置，但连接控制器的实际数量可变，并会逐个检查每一个从站的供应商ID和产品ID，如果发现一个不匹配，堆栈启动都会停止。

另外，只要设备支持，由库IloDrvBusControl.library提供的接口可以用来实现从应用外部对EtherCAT装置进行访问。

## 从站的隐式实例

对于每个 EtherCAT 从站，只要设备被插入到设备列表中，则会建立一个“ETCSlave”类型的隐式实例。实例名称与设备列表中使用的设备名称完全相同。

使用实例化对象的输入输出参数实现特殊的用法，如在应用运行时，使用从站实例获得、切换和检查从站状态。 ETCSlave类型的隐式实例定义如下表所述：

ETCSlave类型的隐式实例定义如下表所述：

ETCSlave隐式实例	
<b>ETCSlave</b> —xSetOperational <i>BOOL</i> <i>ETC_SLAVE_STATE</i> wState —	
输入参数	定义
xSetOperational	在上升沿时，尝试将从站通讯状态机设置为 ETC_SLAVE_OPERATIONAL 模式。
输出参数	定义
wState	返回从站的当前状态，可能取值为： 0: ETC_SLAVE_BOOT 1: ETC_SLAVE_INIT 2: ETC_SLAVE_PREOPERATIONAL 4: ETC_SLAVE_SAVEOPERATIONAL 8: ETC_SLAVE_OPERATIONAL

## 说明

状态ETC\_SLAVE\_OPERATIONAL表示配置已成功完成。如果配置时发生错误，有可能设备会回到之前的状态。以下以IS620N从站为示例，实现显示从站状态功能。

### ETCSlave示例

以620N为例添加620N实例名称，定义：nSlaveState: ETC\_SLAVE\_STATE;

编程	说明
IS620N(xSetOperational:= , wState=> nSlaveState );	从站IS620N隐式实例调用，将从站状态输出到nSlaveState 变量中。

## 从站属性

属性	定义
VendorID	在EtherCAT主站启动后，此属性返回设备中读取的供应商ID。
ConfigVendorID	此属性从配置中读取供应商ID。
ProductID	在EtherCAT主站启动后，该属性返回从站中读取的产品ID。
ConfigProductID	此属性从配置中读取产品ID。
SerialID	在EtherCAT堆栈启动后，该属性包含了设备序列号。
LastEmergency	如果收到消息，则将此信息存储于从站内部。基于这一属性可在应用中读取信息。另外还添加了一个日志消息。

另外，只要设备支持，由IloDrvBusControl.library库提供的接口可以实现从应用外访问EtherCAT设备。如果在高级设置中激活了供应商和产品ID，只要检测到供应商ID与配置供应商ID或者产品ID与配置产品ID不匹配，则主站的启动将被停止。

## 检查所有从站的链表

每个EtherCAT 主站和EtherCAT 从站都会隐式创建一个功能块实例，它可以用来监视各个从站的状态。为此，此实例必须在应用程序中调用，通过wState读取从站状态。为了简化编程，所有主站和从站都可以由链表找到，因此所有从站都可以由一个简单的‘WHILE’循环检查。

主站和从站分别有对应的属性NextInstance 和 LastInstance，可以返回指向下一个和最后一个从站的指针。另外，主站的FirstSlave属性有效，其提供了一个指向第一个从站的指针。

### 链表功能示例

检查所有从站状态，定义：pSlave: POINTER TO ETCSlave;

编程	说明
<pre>pSlave := EtherCAT_Master.FirstSlave; WHILE pSlave &lt;&gt; 0 DO pSlave^(); //TODO: 用户添加代码，例如可以做从站Op状态计数 pSlave := pSlave^.NextInstance; END WHILE</pre>	<p>首先通过EtherCAT_Master.FirstSlave找到主站的第一个从站。</p> <p>在‘WHILE’循环中调用各个实例，由此确定wState，然后检查状态。</p> <p>通过pSlave^.NextInstance找到指向下一个从站的指针。</p> <p>在列表结尾出指针为空，循环结束。</p>

## 用于 CoE 的 IODrvEtherCAT 函数库功能块

CoE 功能块：CANOPEN over EtherCAT 功能块

EtherCAT库IODrvEtherCAT.library使能EtherCAT配置后会被自动添加入工程中，其中包含了读写参数的功能块，因此在线模式下，能够检查与修改特殊参数。在使用CANOPEN over EtherCAT功能块时可以同时调用多个功能模块，内部请求会以队列方式处理。

CANOPEN over EtherCAT功能块包含以下功能块：

- “ETC\_CO\_SdoRead” (取出参数，长度可能超过4字节)
- “ETC\_CO\_SdoRead4” (读取参数，长度不大于4字节)
- “ETC\_CO\_SdoReadDword” (读取参数，并将值存到一个DWORD中)
- “ETC\_CO\_SdoRead\_Access” (读取所有记录)
- “ETC\_CO\_SdoRead\_Channel” (读取从站参数)
- “ETC\_CO\_SdoWrite” (写参数，长度可能超过4字节)
- “ETC\_CO\_SdoWrite4” (写参数，长度不大于4字节)
- “ETC\_CO\_SdoWriteDWord” (直接在DWORD中写参数值) E
- “ETC\_CO\_SdoWriteAccess” (写从站参数)

### ETC\_CO\_SdoRead

此功能模块由IODrvEtherCAT.library库文件提供，用于读取 EtherCAT 从站参数。与 ETC\_CO\_SdoRead4 不同，该模块支持大于 4 字节的参数。读取的参数由对象字典索引和子索引指定。

ETC_CO_SdoRead功能块	
<b>ETC_CO_SdoRead</b>	
输入参数	定义
xExecute	在这个输入的上升沿，启动读取从站参数。为了获得内部通道存储单元分配，这个实例必须通过‘xExecute:= FALSE’ 调用至少一次。
xAbsort	如果此参数为 ‘TRUE’，当前读取过程将被终止。
usiCom	EtherCAT主站个数：如果仅使用一个EtherCAT主站，usiCom为 ‘1’。使用多个主站时，第一个主站为 ‘1’，第二个为 ‘2’，依次类推。
uiDevice	从站的物理地址。 如果主站的 自动配置模式被取消，从站可以设置特有地址。这个任意选择的地址必须输入到这里。 如果激活自动配置模式，第一个从站将得到地址 ‘1001’ .当前从站地址总能在 □ 设备编辑器的从站配置对话框中的 ‘EtherCAT 地址’ 栏进行查核。
usiChannel	保留用于扩展。
wIndex	对象字典中的参数索引。
bySubIndex	对象字典中的参数子索引。
udiTimeout	在这里你可以设定以毫秒为单位的超时时间。如果在这段时间内，读取参数没有被执行，将提示一个错误信息。
pBuffer	数据缓冲区的指针，数据缓冲区即参数成功传递后数据的存储区域。
szSize	数据缓冲区（见上：pBuffer）的大小，以字节表示。
输出参数	定义
xDone	只要读取参数已经成功结束，此值输出为 ‘TRUE’。
xBusy	只要读取参数尚未结束时，此值输出为 ‘TRUE’。
xError	如果出错，此值输出为 ‘TRUE’。eError参数将显示错误原因。
eError	此输出（类型ETC_CO_ERROR）显示xError所指示的出错原因。例如 ‘ETC_CO_TIMEOUT’ 表示超时错误。
udiSdoAbort	当设备检查出错时，此输出将提供更多有关错误的信息。
szDataRead	读取字节的数目；最大的szSize（参见输入参数）。

**ENUM ETC\_CO\_ERROR**

错误	代码	说明
ETC_CO_NO_ERROR	0	没有错误
ETC_CO_FIRST_ERROR	5750	错误原因被存储于udiSdoAbort的输出
ETC_CO_OTHER_ERROR	5751	没找到主站
ETC_CO_DATA_OVERFLOW	5752	ETC_CO_Expedited 并且大小 > 4
ETC_CO_TIME_OUT	5753	超过时限
ETC_CO_FIRST_MF	5770	没有使用
ETC_CO_LAST_ERROR	5799	没有使用

**ETC\_CO\_SdoRead4**

此功能块由库 IODrvEtherCAT.library 提供，用于读取 EtherCAT 从站参数。不同于 ETC\_CO\_SdoRead 功能块只能读取不大于 4 字节的参数。读取的参数特别由对象字典里的索引和子索引指定。

ETC_CO_SdoRead4 功能块	
<b>ETC_CO_SdoRead4</b>	
输入参数	定义
xExecute	在这个输入的上升沿，启动读取从站参数。为了获得内部通道的存储单元分配，这个实例必须通过‘xExecute:= FALSE’ 调用至少一次。
xAbort	如果此输入为 ‘TRUE’，当前读取过程将被终止。
usiCom	EtherCAT 主站个数：如果仅使用一个 EtherCAT 主站，usiCom 为 ‘1’。使用多个主站时，第一个主站为 ‘1’，第二个为 ‘2’，依次类推。
uiDevice	从站的物理地址。 如果主站的自动配置模式被取消，从站可以设置特有地址。这个任意选择的地址必须输入到这里。 如果激活自动配置方式，第一个从站将得到地址“1001”。当前从站地址总能在 <input type="checkbox"/> 设备编辑器的从站配置对话框中的 ‘EtherCAT 地址’ 栏检查。
usiChannel	为将来扩展保留的，当前未使用。
wIndex	对象字典中的参数索引。
bySubIndex	对象字典中的参数子索引。
udiTimeout	在这里可以设定以毫秒为单位的超时时间。如果在此时间内，读取参数没有被执行，将生成一个错误信息。
输出参数	定义
wState	返回从站的当前状态，可能取值为： 0: ETC_SLAVE_BOOT 1: ETC_SLAVE_INIT 2: ETC_SLAVE_PREOPERATIONAL 4: ETC_SLAVE_SAVEOPERATIONAL 8: ETC_SLAVE_OPERATIONAL
xDone	只要读取参数已经成功结束，此输出为 ‘TRUE’。
xBusy	只要读取参数尚未结束，此输出为 ‘TRUE’。
xError	如果出错，此值输出为 ‘TRUE’。eError 参数将显示错误原因。
eError	此输出（类型 ETC_CO_ERROR）表示 xError 所指示的出错原因。例如 ‘ETC_CO_TIMEOUT’ 表示超时错误。
abyData	读到的参数数据会被复制到这个 4 字节的数组中。 如果第一个字节已经被读取，将保存在数组的第一个索引中。如果是 2 个或者 4 个字节数据，按照 Intel 字节顺序拷贝到这个数组中。
usiDataLength	读取字节的个数 (1,2,4)。

#### ENUM ETC\_CO\_ERROR

错误	代码	说明
ETC_CO_NO_ERROR	0	没有错误
ETC_CO_FIRST_ERROR	5750	错误原因被存储于 udiSdoAbort 的输出
ETC_CO_OTHER_ERROR	5751	没找到主站

错误	代码	说明
ETC_CO_DATA_OVERFLOW	5752	ETC_CO_Expedited 且大小 > 4
ETC_CO_TIME_OUT	5753	超过时限
ETC_CO_FIRST_MF	5770	没有使用
ETC_CO_LAST_ERROR	5799	没有使用

### ETC\_CO\_SdoReadDword

此功能块由库 IODrvEtherCAT.library 提供，如功能块 ETC\_CO\_SdoRead4一样用来读取 EtherCAT 从站参数。不过读取的数据不拷贝到数组中而是到一个DWORD中（dwData）。如果需要字节交换，则可自动完成。因此读到的数据可直接被后面的过程所用。

### ETC\_CO\_SdoRead\_Access

此功能块由库 IODrvEtherCAT.library 提供，如功能块 ETC\_CO\_SdoRead一样用来读取 EtherCAT 从站参数。不过通过附加输入xCompleteAccess (BOOL)，允许读入所有记录的完整索引。为此，xCompleteAccess 必须要设为 ‘TRUE’ 且bySubIndex必须为 ‘0’ 。

### ETC\_CO\_SdoRead\_Channel

EtherCAT 配置编辑器 > EtherCAT 编程接口 > 用于 “CAN over EtherCAT” 的 IODrvEtherCAT 函数库功能块 > ETC\_CO\_SdoRead\_Channel

此功能块由库 IODrvEtherCAT.library 提供，如功能块 ETC\_CO\_SdoRead\_Access一样用来读取所有 EtherCAT 从站参数。

但是它有一个附加输入byChannelPriority (BYTE) 用来指定CoE邮箱消息中的通道和优先权。此字节的前6位指定通道，后2位指定优先级。

### ETC\_CO\_SdoWrite

此功能块由库 IODrvEtherCAT.library 提供，用于写 EtherCAT 从站参数。不同于 ETC\_CO\_SdoWrite4 功能块可以用于读取大于 4 字节的参数。写入的参数特别由对象字典里的索引和子索引指定。

ETC_CO_SdoWrite 功能块	
<b>ETC_CO_SdoWrite</b>	
xExecute	BOOL
xAbort	BOOL
usiCom	USINT
uiDevice	UINT
usiChannel	USINT
wIndex	WORD
bySubindex	BYTE
udiTimeOut	UDINT
pBuffer	CAA_PVOID
szSize	CAA_SIZE
eMode	ETC_CO_MODE
输入参数	定义
xExecute	在这个输入的上升沿，启动写入从站参数。为了获得内部通道的存储单元分配，这个实例必须通过“xExecute:= FALSE” 调用至少一次。
xAbort	如果此输入参数为 “TRUE”，当前写入过程将被终止。
usiCom	EtherCAT主站个数：如果仅使用一个EtherCAT主站，usiCom为 ‘1’ 。使用多个主站时，第一个主站为 ‘1’ ，第二个为 ‘2’ ，依次类推。

ETC_CO_SdoWrite功能块	
uiDevice	从站的物理地址。 如果取消主站 自动配置模式，从站可以设置特有地址。这个任意选择的地址必须输入到这里。 如果激活自动配置方式，第一个从站将得到地址 ‘1001’ .当前从站地址总能在 从站配置对话框中的 ‘EtherCAT地址’ 栏检查。
usiChannel	为将来扩展保留的，当前未使用。
wIndex	对象字典中的参数索引。
bySubIndex	对象字典中的参数子索引。
udiTimeout	在这里可以设定以毫秒为单位的超时时间。如果在此时间内, 写入参数没有被执行, 将生成一个错误信息。
pBuffer	数据缓冲区的指针, 数据缓冲区即参数成功传递后数据的存储区域。
szSize	数据缓冲区 (见上: pBuffer) 的大小, 以字节表示。
eMode	这个输入 (枚举ETC_CO_MODE) 定义了写入字节的个数: 可能的值包括: ETC_CO_AUTO (自动) ,ETC_CO_EXPEDITED (加速) 或ETC_CO_SEGMENTED (分段) 。通常使用ETC_CO_AUTO 模式, 因为在这个模式下, 自动匹配数据长度。
输出参数	定义
xDone	只要写入参数已经成功结束, 此输出为 “TRUE” 。
xBusy	只要写入参数尚未结束, 此输出为 “TRUE” 。
xError	如果出错, 此输出为 “TRUE” 。eError参数将显示错误原因。
eError	此输出 (类型ETC_CO_ERROR) 表示xError所指示的出错原因。例如” ETC_CO_TIMEOUT” 表示超时错误。
udiSdoAbort	当设备出错时, 此输出将提供更多的有关错误的信息。
szDataWritten	写入的字节个数; 成功写入后, 将设置到szSize。

**ENUM ETC\_CO\_MODE**

模式	编号	说明
AUTO	0	客户选择自动模式
EXPEDITED	1	客户使用加速协议
SEGMENTED	2	客户使用分段传输协议

**ETC\_CO\_SdoWrite4**

此功能块由库 IODrvEtherCAT.library提供, 用于写 EtherCAT 从站参数。不同于 ETC\_CO\_SdoWrite功能块此功能块只支持写小于 4 字节的参数。写入的参数特别由对象字典里的索引和子索引指定。

ETC_CO_SdoWrite4功能块	
<b>ETC_CO_SdoWrite4</b>	
输入参数	定义
xExecute	启动写入从站参数。为了获得内部通道的存储单元分配, 这个实例必须通过 ‘xExecute:= FALSE’ 调用至少一次。
xAbsort	如果此输入参数为 ‘TRUE’ , 当前写入过程将被终止。

ETC_CO_SdoWrite4功能块	
usiCom	EtherCAT主站个数：如果仅使用一个EtherCAT主站，usiCom为‘1’。使用多个主站时，第一个主站为‘1’，第二个为‘2’，依次类推。
uiDevice	从站的物理地址。 如果取消主站的自动配置模式，从站可以设置特有地址。这个任意选择的地址必须输入到这里。 如果激活自动配置方式，第一个从站将得到地址‘1001’。当前从站地址总能在从站配置对话框中的“EtherCAT地址”检查。
usiChannel	为将来扩展保留的，当前未使用。
wIndex	对象字典中的参数索引。
bySubIndex	对象字典中的参数子索引。
udiTimeout	在这里可以设定以毫秒为单位的超时时间。如果在此时间内,写入参数没有被执行，将生成一个错误信息。
abyData	这个数组包含了4个写入的数据。数据必须按照Intel字节顺序存储。
usiDataLength	写入字节的个数(1, 2, 4)
输出参数	定义
xDone	一旦写参数完成这个输出将会被置位 TRUE。
xBusy	如果写没有完成此输出将会为 TRUE。
xError	如果一个错误产生此输出将会被置为 TRUE，eError 将显示错误原因。
eError	这个输出(类型 ETC_CO_ERROR)显示当前错误的原因，标识为 xError。例如“ETC_CO_TIMEOUT”表示超时。
udiSdoAbort	如果设备中发生一个错误，这个输出将提供更多的错误信息。

#### ENUM ETC\_CO\_MODE

模式	编号	说明
AUTO	0	客户选择自动模式
EXPEDITED	1	客户使用加速协议
SEGMENTED	2	客户使用分段传输协议

#### ETC\_CO\_SdoWriteDWord

此功能块由库 IODrvEtherCAT.library 提供，如功能块 ETC\_CO\_SdoWrite4一样用来写 EtherCAT 从站数据。但是要写入的数据不以数组而以一个DWORD (dwData) 的形式给出。若需要字节交换，则可以自动完成。因此要被写入的值可直接被指定。

#### ETC\_CO\_SdoWriteAccess

此功能块由库 IODrvEtherCAT.library 提供，如功能块 ETC\_CO\_SdoWrite一样用来写 EtherCAT 从站参数。

但是通过附加的输入xCompleteAccess (BOOL)，所有记录的完整索引都可被写入。为此，xCompleteAccess 必须要设为‘TRUE’且 bySubIndex 必须为‘0’。另一个输入byChannelPriority (BYTE) 可被用来指定CoE邮箱消息中的各个数据元素（通道和优先级）。

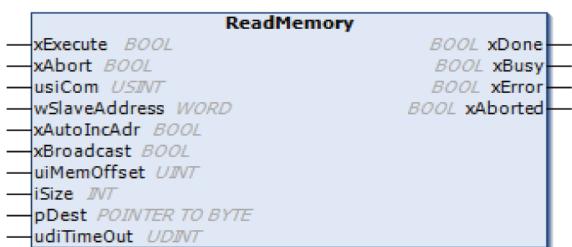
#### 直接访问 EtherCAT 从站内存

EtherCAT 配置编辑器 > EtherCAT 编程接口 > 直接访问 EtherCAT 从站内存

直接访问 EtherCAT 从站内存：ReadMemory 与 WriteMemory。

- ReadMemory

这个功能块位于函数库 IODrvEtherCAT.library 用于读取 EtherCAT 从站的内存数据。

ReadMemory功能块		
		
输入参数	类型	定义
xExecute	BOOL	上升沿：动作启动 下降沿：复位输出  如果在功能块完成动作之前有一个下降沿，输出操作会以通常方式完成并且在动作完成或者中断 (xAbort)时会被复位。在这种情况下，相关的输出值(xDone, xError, iError) 正好位于一个周期内输出。
xAbort	BOOL	如果这个输入为 TRUE，动作会立即停止并且所有输出被复位为初始值。
usiCom	USINT	主站索引 1: 第一个 EtherCAT 主站
wSlaveAddress	WORD	自动创建的地址或者设备的物理地址
xAutoIncAdr	BOOL	确定要使用地址方式的标志
xBroadcast	BOOL	如果要使用板卡的标志，如果为 true 那么wSlaveAddress 和 bAutoIncAdr 不会被使用
uiMemOffset	UINT	内存地址偏移
iSize	INT	读取字节
pDest	POINTER TO BYTE	存取数据缓冲区
udiTimeOut	UDINT	操作超时时间 (ms)
输出参数	类型	定义
xDone	BOOL	动作成功完成
xBusy	BOOL	功能块激活
xError	BOOL	TRUE: 产生错误，功能块中止动作FALSE: 没有错误
xAbsorted	BOOL	用于中止动作

示例：寄存器 0x130 读取(当前状态)

PROGRAM PLC\_PRG

VAR

etcreadmemory : ReadMemory;

wStatus : WORD;

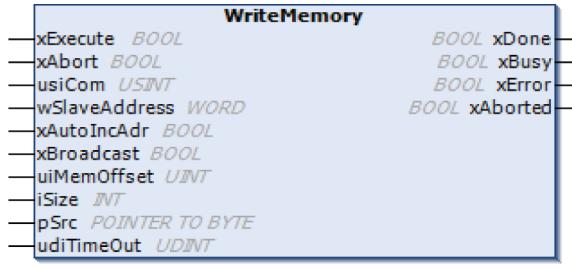
xRead : BOOL;

END\_VAR

```
etcreadmemory(xExecute := xRead, usiCom:=1, wSlaveAddress := 1002,
              xAutoIncAdr := FALSE, xBroadcast := FALSE, uiMemOffset := 16#130,
              iSize := 2, pDest := ADR(wStatus), udiTimeout := 500);
```

- WriteMemory

这个功能块位于函数库 IODrvEtherCAT.library 用于往 EtherCAT 从站内存中写数据。

WriteMemory功能块		
		
输入参数	类型	定义
xExecute	BOOL	上升沿：动作启动 下降沿：复位输出  如果在功能块完成动作之前有一个下降沿，输出操作会以通常方式完成并且在动作完成或者中断(xAbort)时会被复位。在这种情况下，相关的输出值(xDone, xError, iError)正好位于一个周期内输出。
xAbort	BOOL	如果这个输入为 TRUE，动作会立即停止并且所有输出被复位为初始值。
usiCom	USINT	主站索引 1：第一个 EtherCAT 主站
wSlaveAddress	WORD	自动创建的地址或者设备的物理地址
xAutoIncAdr	BOOL	确定要使用地址方式的标志
xBroadcast	BOOL	如果要使用板卡的标志，如果为 true 那么wSlaveAddress 和 bAutoIncAdr 不会被使用
uiMemOffset	UINT	内存地址偏移
iSize	INT	写入字节
pDest	POINTER TO BYTE	存取写入数据缓冲区
udiTimeOut	UDINT	操作超时时间 (ms)
输出参数	类型	定义
xDone	BOOL	动作成功完成
xBusy	BOOL	功能块激活
xError	BOOL	TRUE：产生错误，功能块中止动作FALSE：没有错误
xAborted	BOOL	用于中止动作

示例：写入寄存器 0x120 (控制寄存器)

PLC\_PRG

VAR

```

etcwritememory : WriteMemory;
xWrite : BOOL;
wCommand : WORD := 4; // set to safe operational
END_VAR

etcwritememory(xExecute := xWrite, usiCom:=1, wSlaveAddress := 1002,
xAutoIncAdr := FALSE, xBroadcast := FALSE, uiMemOffset := 16#120,
iSize := 2, pSrc := ADR(wCommand), udiTimeout := 500);

```

## 4.5 Modbus 设备编辑器

### 4.5.1 串行硬件端口

AM600 支持两路485 串口通信，分别是串口0 和串口1，均支持自由协议。

端口	通道	引脚	定义
	COM0 (RS485)	1	RS485-
		2	RS485+
		5	GND
	COM1 (RS485)	6	RS485-
		9	RS485+
		3	GND

AC700系列PLC支持一路485通信和一路232通信，通过I/O通信接口接线。其中，485通信引脚为1/3/5，232通信引脚为2/4/6，具体引脚定义如下表所示。

类型	功能	端子	编号	IO/通信接口	编号	类型	功能	端子
RS485	RS485+	485+	1		2	232R	RS232接收	RS232
	RS485-	485-	3		4	232T	RS232发送	
	串口地	GND	5		6	GND	串口地	
DI	开机信号		7		8		RUN/STOP	DI
DI	高速输入0	X0	9		10	X1	高速输入1	DI
DI	高速输入2	X2	11		12	X3	高速输入3	DI
DI	高速输入4	X4	13		14	X5	高速输入5	DI
DI	高速输入6	X6	15		16	X7	高速输入7	DI
DI	输入公共端	S/S	17		18	COM	输出公共端	DO
DO	高速输出0	Y0	19		20	Y1	高速输出1	DO
DO	高速输出2	Y2	21		22	Y3	高速输出3	DO

AC800系列PLC支持一路485通信和一路232通信，通过I/O通信接口接线。其中，485通信引脚为8/9/11，232通信引脚为8/10/12，具体引脚定义如下表所示。

描述	功能	信号名	编号	IO/通信接口	编号	信号名	功能	描述
输入高电平宽度为500ms的脉冲时，模拟外部开机按键按一次，启动PLC	开机信号（配合UPS使用）		1		2	P_STATUS	上电点灯信号	控制器上电启动后输出
ON-OFF变化时启动掉电保存功能	掉电检测信号	P_OK	3		4	P_STATUS	运行状态信号	控制器上电启动后输出
OFF时RUN；ON时STOP	RUN/STOP	RUN	5		6	0V	输出公共端	-
-	输入公共端	0V	7		8	GND	通信参考地	-
-	RS485+	485+	9		10	232R	RS232接收	-
-	RS485-	485-	11		12	232T	RS232发送	-

AM300/AM500系列PLC最多可支持3路RS485通信（主单元支持1路，扩展卡可扩展2路），本体引脚定义如下表所示。



信号说明	左侧端子	右侧端子	信号说明
RS485差分对正信号	485+	+24V	直流24V电源正
RS485差分对负信号	485-	0V	直流24V电源负
RS485的通信地端	GND		PE

## 说明

AC700/AC800系列，在软件界面中COM端口0对应RS232通信接口，COM端口1对应RS485通信接口，在配置使用时须区分。

## 4.5.2 网络组态

### AM400/AM600/AC700/A800网络组态

- 在左侧设备树中双击“网络组态”，进入“Network Configuration”界面。
- 单击PLC设备图片，显示PLC内部所支持的主/从站的使能信息。



### 3. 使能PLC为 Modbus主站或者Modbus从站。

- 使能PLC为Modbus主站

勾选“Modbus 主站”复选框，然后在右侧“网络设备列表”中双击“MODBUS”，使能PLC为Modbus主站并将Modbus从站添加至网络中。



- 使能PLC为Modbus从站

勾选“Modbus 从站”复选框，使能PLC为Modbus从站。

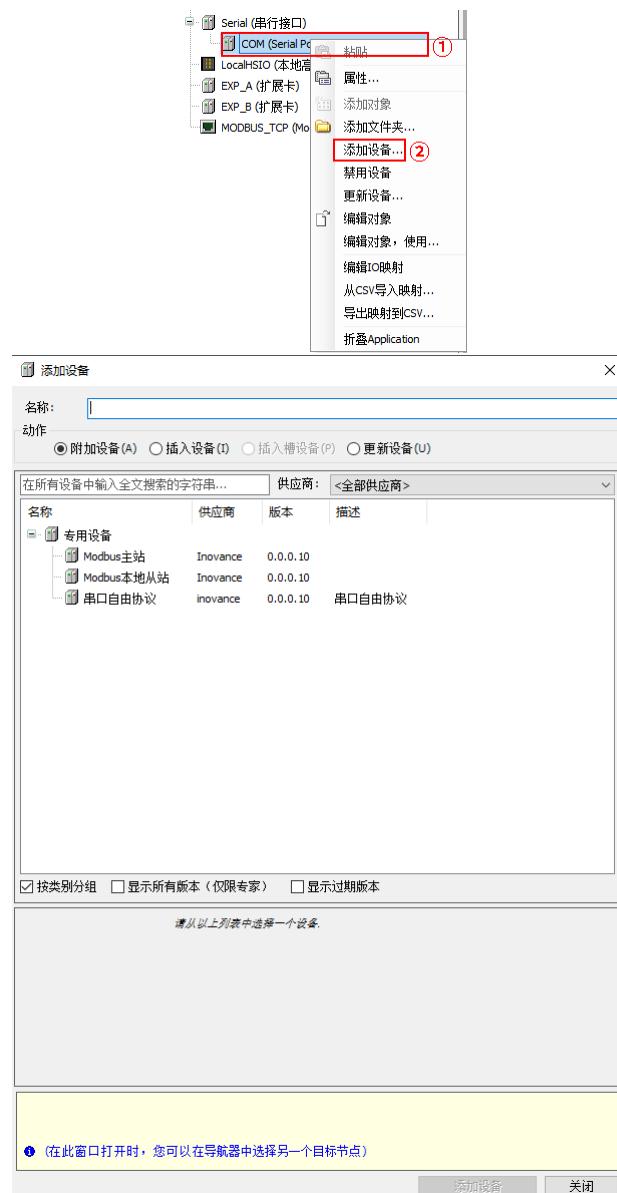


PLC做Modbus从站设备时，可以被主站设备访问的地址范围定义如下：

- 位变量操作：功能码0x01、0x05、0x0F可读写%QX0.0-%QX8191.7，共65536个位变量；功能码0x02可读写IX0.0-IX8191.7，共65536个位变量。
- 寄存器变量操作：功能码0x03、0x04、0x06、0x10可读写MW0-MW65535，共65536个寄存器变量。
- 汇川HMI可以访问系统变量SM0-SM7999，寄存器变量SD0-SD7999。

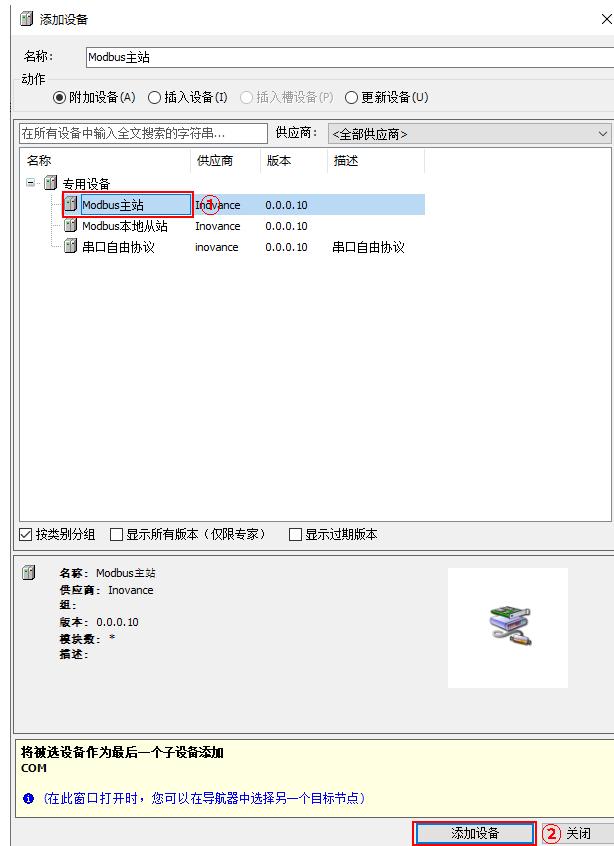
## AM300/AM500网络组态

### 1. 在左侧设备树中右键单击“COM(Serial Port)”，选择“添加设备”，打开“添加设备”对话框。

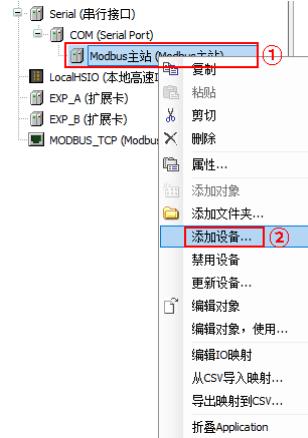


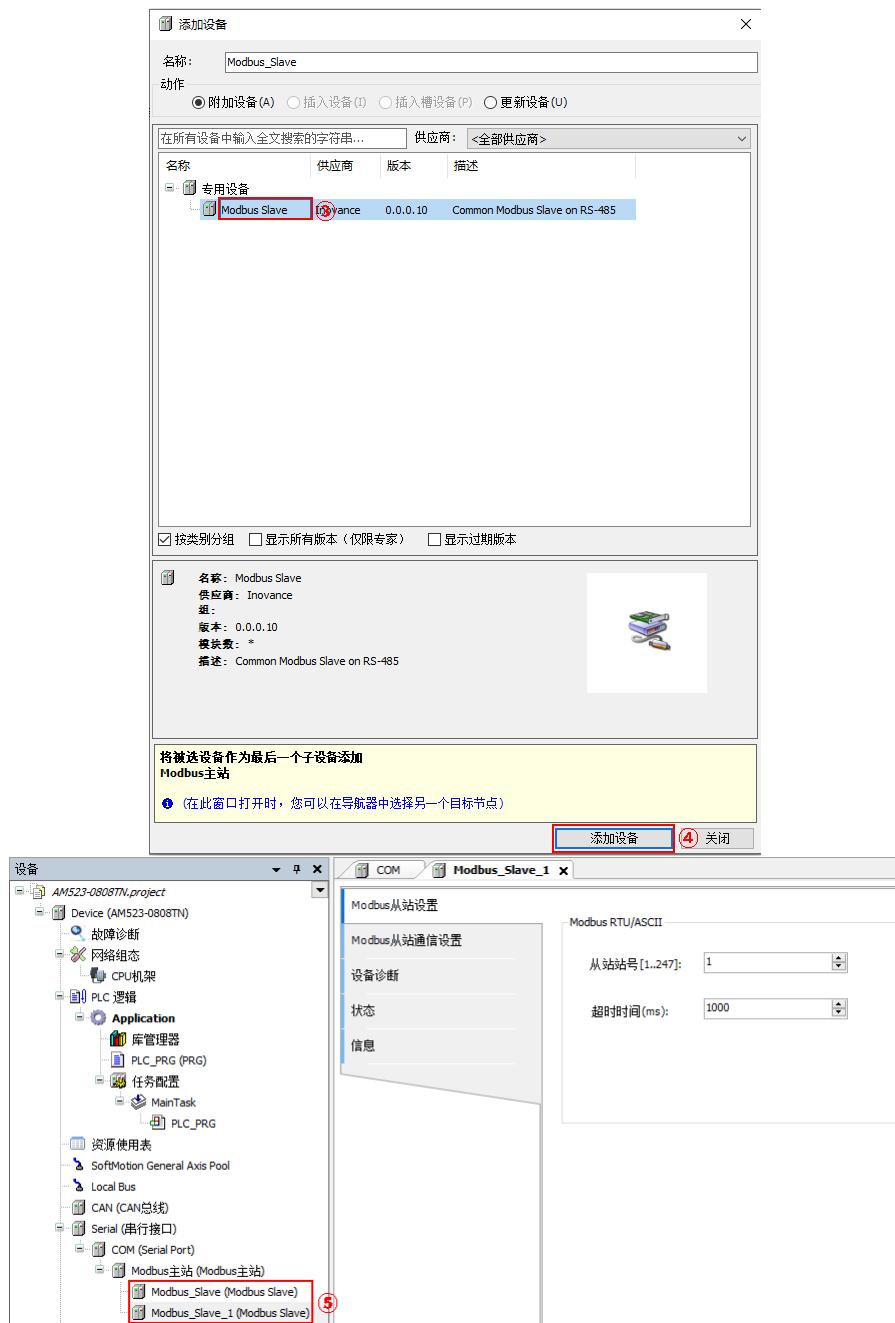
2. 使能PLC为 Modbus主站或者Modbus从站。

- 使能PLC为Modbus主站
  - a. 选择“Modbus 主站”，单击“添加设备”，使能PLC为Modbus主站。

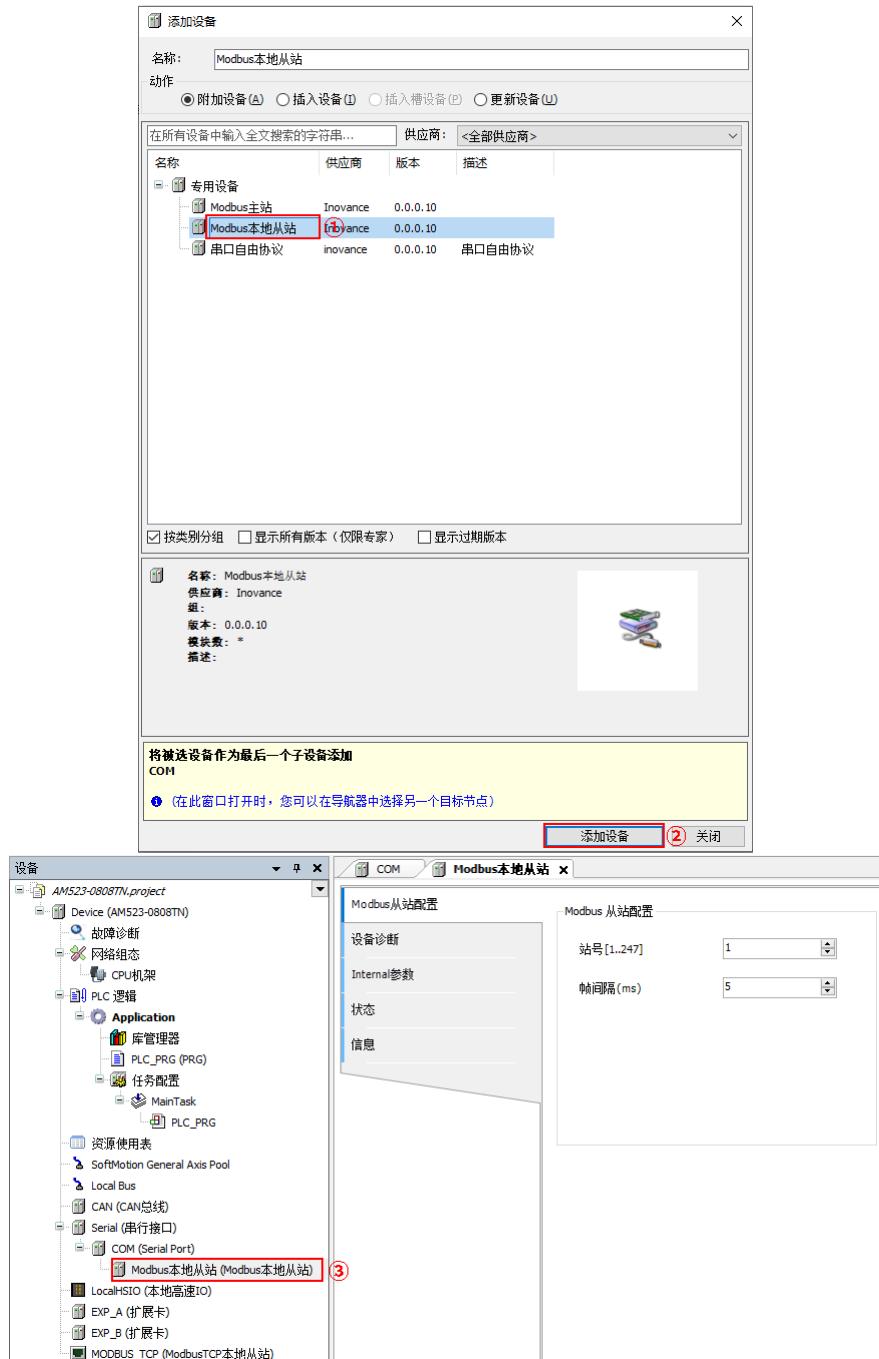


- b. 右键单击“Modbus主站(Modbus主站)”，选择“添加设备”，在打开的对话框中单击“Modbus slave”，单击“添加设备”，添加Modbus从站至网络中。





- 使能PLC为Modbus从站  
选择“Modbus 从站”，单击“添加设备”，使能PLC为Modbus从站。



PLC做Modbus从站设备时，可以被主站设备访问的地址范围定义如下：

- 位变量操作：功能码0x01、0x05、0x0F可读写%QX0.0-%QX8191.7，共65536个位变量；功能码0x02可读写%IX0.0-%IX8191.7，共65536个位变量。
- 寄存器变量操作：功能码0x03、0x04、0x06、0x10可读写MW0-MW65535，共65536个寄存器变量。
- 汇川HMI可以访问系统变量SM0-SM7999，寄存器变量SD0-SD7999。

#### 4.5.3 Modbus 主站配置

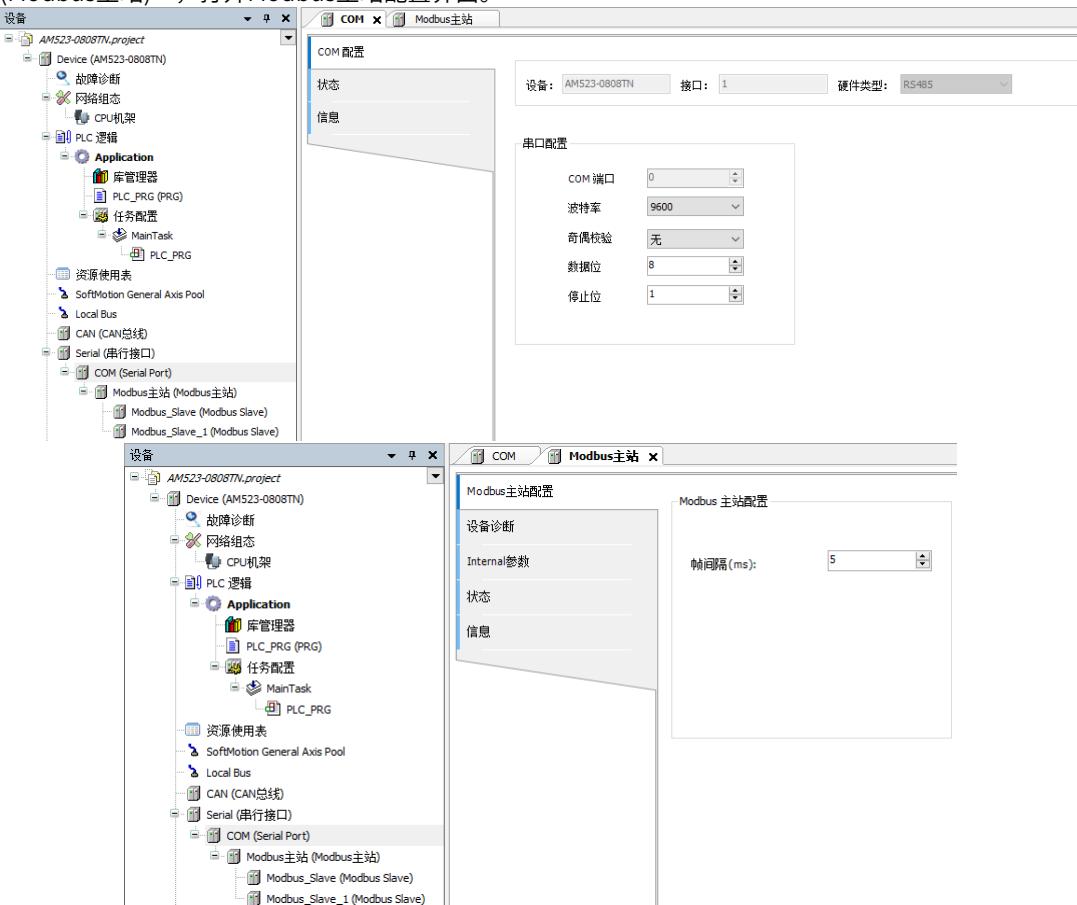
PLC做Modbus主站时，需要对Modbus主站进行配置，Modbus主从站通信参数配置一致时，才能正常通信。

### 1. 打开Modbus主站配置界面。

- AM400/AM600/AC700/A800：在左侧设备树中双击Modbus主站设备，打开Modbus主站配置界面。



- AM300/AM500：在左侧设备树中双击“COM(Serial Port)”，打开COM配置界面；双击“Modbus主站（Modbus Master）”，打开Modbus主站配置界面。



### 2. 配置Modbus主站参数，参数说明如下表所示。

参数名称	参数说明
COM端口	该主站物理连接选择串口0或串口1
波特率	通信时的速率
奇偶校验	通信帧的校验方式
数据位	通信帧包含的实际数据位
停止位	通信时标识单个包的最后位

参数名称	参数说明
传输模式	RTU
帧间隔	主站接收上一个响应数据帧到下一个请求数据帧之间等待的时间间隔

示例如下表所示。

参数名称	配置值
COM端口	0
波特率	115200
奇偶校验	偶校验
数据位	8
停止位	1
传输模式	RTU
帧间隔	2ms

#### 4.5.4 Modbus 主站通信配置

PLC做Modbus主站时，双击设备树中的从站设备打开Modbus从站设置窗口，如下图所示：

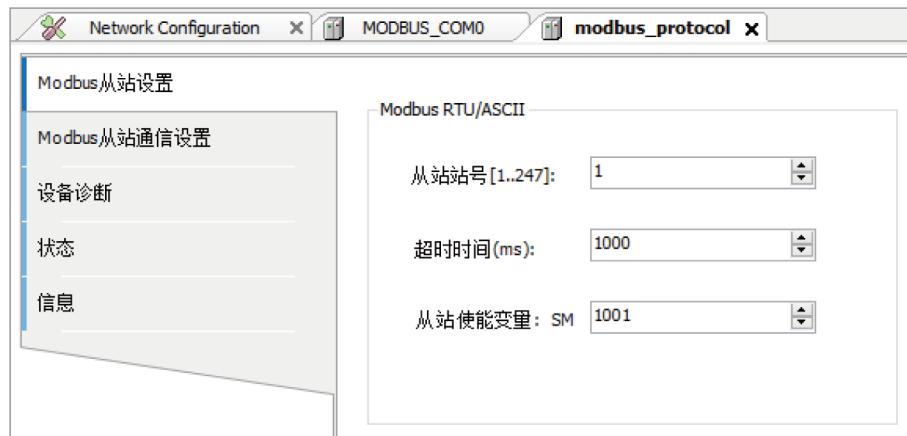


图4-15 端口作为主站时的Modbus从站配置

从站配置参数：

配置项	功能
从站站号	标识从站号，范围1~247
超时时间	主站发帧后，超过该时间从站未响应，主站报接收超时
从站使能变量	编程使能该变量后，主站开始向该从站发送通信帧

示例：

配置项	配置值
从站站号	11
超时时间	1000ms
从站使能变量	1001

切换页面到Modbus从站通信设置窗口，添加Modbus主从站通信配置，如下图所示。



配置表最多支持60条配置。

图4-16 端口做主站时的Modbus从站通信设置

图中每个通道代表一个独立的Modbus请求，其中，第4行定义了一个循环执行写单个寄存器（功能码0x06）的请求，向偏移量为0x0020的1个寄存器写一个字的数据。

单击“添加…”后会出现一个为Modbus从站添加新通道的对话框。单击“确定”可新建一个通道。

在“Modbus从站通道”列表中选择一个通道，单击“编辑…”，打开对话框，通过修改其中的参数可改变通道的配置，单击“确定”可更新通道设置。当添加或者编辑通道时，对话框中有以下参数可供使用：



图4-17 端口做主站时连接Modbus从站通信设置对话框

#### Modbus通信设置配置

配置项	功能	
名称	通道命名的字符串	
存取类型	读线圈状态（功能码01） 读输入状态（功能码02） 读保持寄存器（功能码03） 读输入寄存器（功能码04） 写单个线圈（功能码05） 写单个寄存器（功能码06） 写多个线圈（功能码15） 写多个寄存器（功能码16）	
触发器	循环执行：周期触发的请求	循环时间：设置时间再次执行
	电平触发：编程进行改变时触发	触发变量（SM）：设置触发SM元件，触发成功后，自动复位该元件
重发次数	本次发生通信故障未获得从站返回帧，则按重发次数进行重新发送。	
注释	可以对数据进行描述的简短文本区域	
读寄存器		
起始地址	读取的寄存器开始位置	
长度	读取的寄存器个数	
错误处理	保持最后的值：使数据保持最后一次的有效值 设置为0：使所有值归零	
写寄存器		
起始地址	写寄存器开始位置	
长度	写寄存器长度	

“长度”参数的有效范围取决于以下功能码：

功能码	类型访问	寄存器数
01	读线圈状态	1~2000
02	读输入状态	1~2000
03	读保持寄存器	1~125
04	读输入寄存器	1~125
05	写单个线圈	1
06	写单个寄存器	1
15	写多个线圈	1~1968
16	写多个寄存器	1~123

Modbus从站Internal I/O映射



图4-18 端口做主站时的Modbus从站Internal I/O映射

在Modbus从站通信设置中添加主从站通信配置后，Internal I/O映射中会自动分配每条配置的映射地址，如上图第一行的%IW1表示将读取的一个线圈数值映射到%IW1这个地址。另外，还可以通过输入助手或者直接输入示例变量路径，将程序中的自定义变量映射到I/O地址。

#### 4.5.5 Modbus 主站广播配置

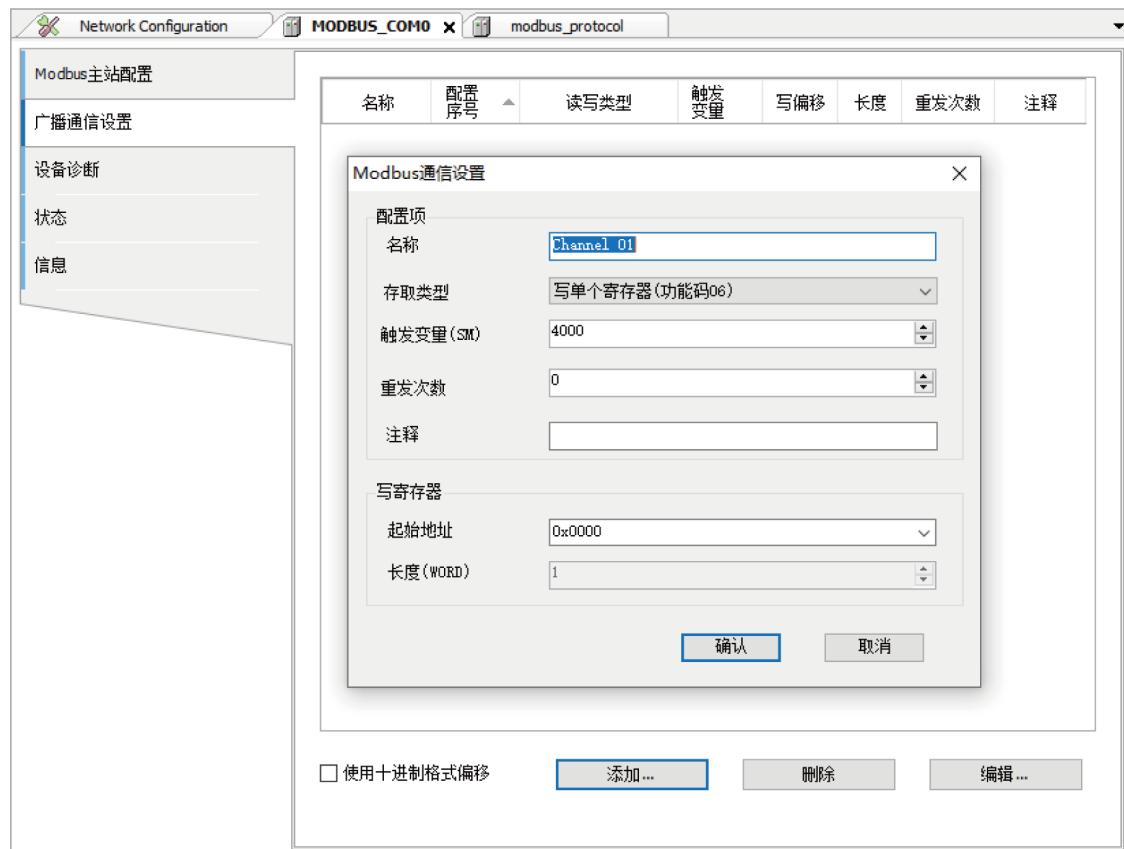


##### 注 意

仅AM400/AM600系列PLC才需要执行Modbus主站广播配置，其他系列PLC可跳过。

当Modbus主站连接多台Modbus从站，所有Modbus从站都接收写操作时，需要Modbus主站进行广播。

1. 在Modbus主站配置界面单击“广播通信设置”，进入“广播通信设置”界面。
2. 单击“添加”，打开“Modbus通信设置”对话框。



3. 配置相关参数，其中相关参数包括：名称、存取类型、触发变量（SM）、重发次数、注释，写寄存器包括起始地址和长度。  
存取类型包括多个功能码，有写单个线圈（功能码05）、写单个寄存器（功能码06）、写多个线圈（功能码15）、写多个寄存器（功能码16）。
  - 触发变量：Modbus主站进行通信的触发条件，只有触发变量为true时，Modbus主站才进行广播通信，需要在编程时对触发变量进行置位。
  - 重发次数：本次发生通信故障未获得从站返回帧，则按重发次数进行重新发送。
4. 单击“确认”，完成配置。

#### 4.5.6 Modbus 从站配置

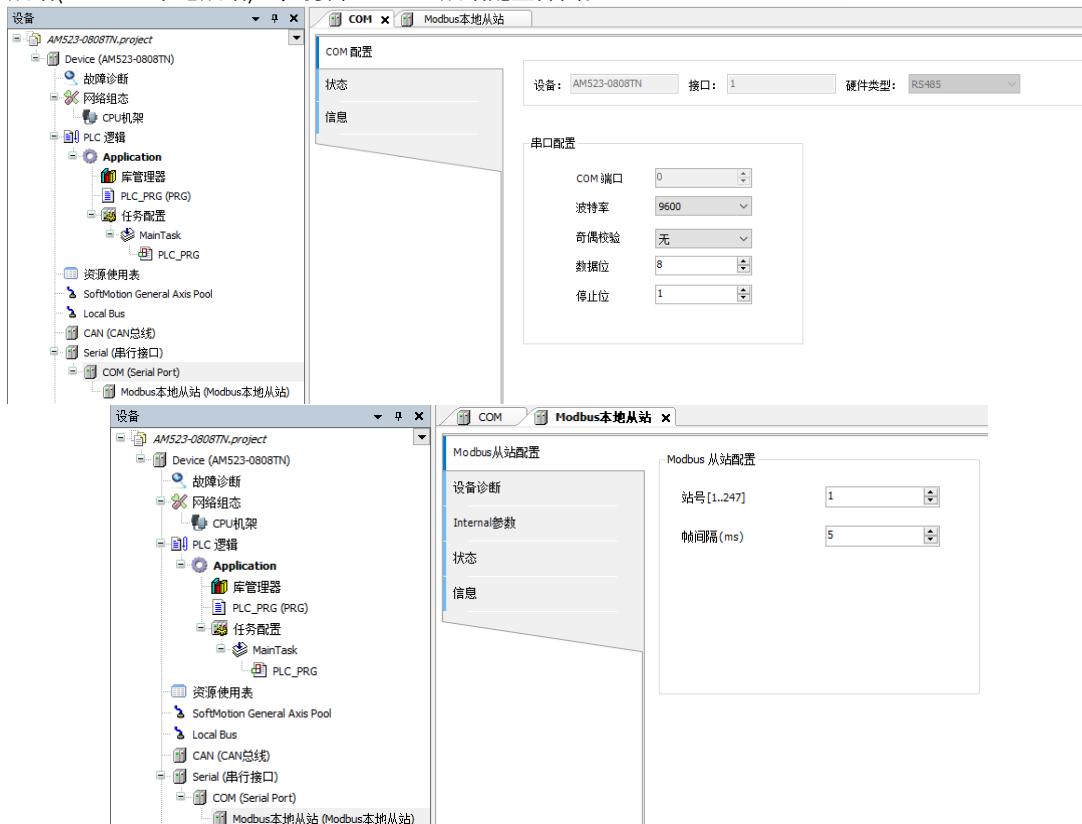
PLC做Modbus从站时，需要对Modbus从站进行配置，Modbus主从站通信参数配置一致时，才能正常通信。

1. 打开Modbus从站配置界面。

- AM400/AM600/AC700/A800：在左侧设备树中双击Modbus从站设备，打开Modbus从站配置界面。



- AM300/AM500：在左侧设备树中双击“COM(Serial Port)”，打开COM配置界面；双击“Modbus本地从站(Modbus本地从站)”，打开Modbus从站配置界面。



## 2. 配置Modbus从站参数。

Modbus从站配置参数中，串口配置部分与Modbus主站串口配置含义相同，Modbus从站配置站号是指本设备站号；帧间隔为收到主站发送的通信帧后，延时回复主站的具体时间段。

### 4.5.7 Modbus设备诊断

Modbus主站设备诊断可显示发生故障的从站和故障的通信配置项。

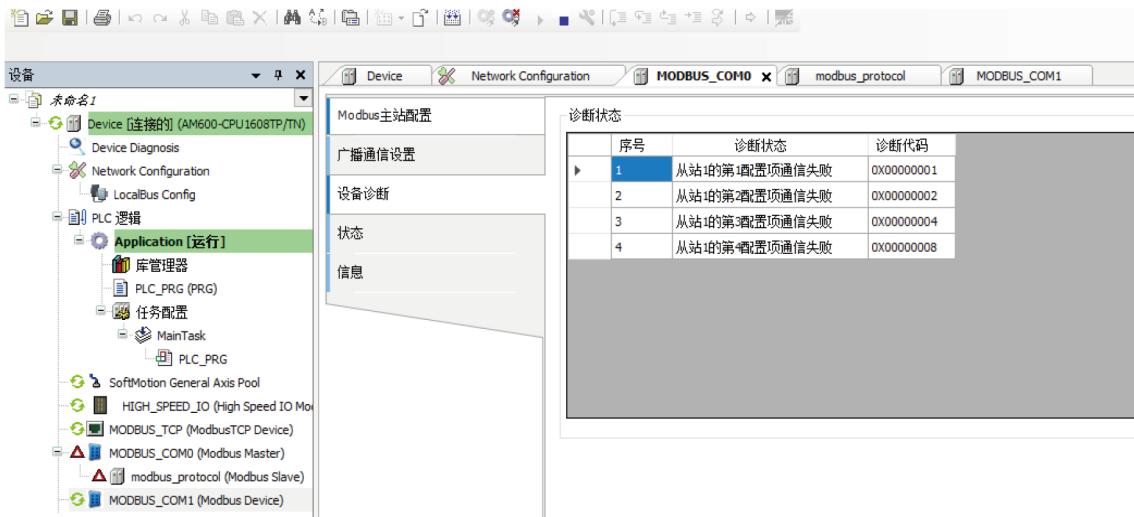


图4-19 Modbus主站设备诊断

### 4.5.8 Modbus常见故障

Modbus主站连接从站时发生的主要故障如下：

- Modbus主站与Modbus从站配置不一致，导致主站与从站通信无法建立。
- Modbus主站访问Modbus从站非法地址，返回错误应答。
- Modbus主站操作Modbus从站写寄存器，但是Modbus从站该寄存器只支持读不支持写操作，Modbus主站会收到Modbus从站返回的出错应答。

#### 错误响应帧格式及含义

错误响应：从机地址+（命令码+0x80）+错误码+CRC校验。

本错误帧适合所有的操作命令帧。

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	命令码+0x80	1个字节	错误命令码
3	错误码	1个字节	1~4

### 4.5.9 Modbus 变量编址

线圈：位变量，只有两种状态0和1。本PLC中包含Q区及SM区等变量。

变量名称	命令码	起始地址	线圈数量	说明
QX0.0-QX8191.7	0X01,0x05,0x0f	0	65536	通用标准Modbus协议都可以访问
IX0.0-IX8191.7	0x2	0	65536	通用标准Modbus协议都可以访问
SM0-SM7999	0x31,0x35,0x3f	0	8000	与汇川HMI的专用协议，使用不同的功能码

寄存器：16位（字）变量，本PLC中包含M区及SD区等变量。

变量名称	命令码	起始地址	寄存器数量	说明
MW0-MW65535	0x03,0x06,0x10	0	65536	通用标准Modbus协议都可以访问
SD0-SD7999	0x33,0x36,0x40	0	8000	与汇川HMI的专用协议，使用不同的功能码

说明：

汇川HMI的专用协议使用不同功能码：在访问SM时，使用0x31,0x35, 0x3f (在访问位变量的命令的基础上加了0x30)；在访问SD时，使用0x33,0x36,0x40 (在访问寄存器变量的命令的基础上加了0x30)。

AM600软元件有Q区，I区，M区这三种，均可以按位，按字节，按字和按双字进行访问，如：%QX、%QB、%QW、%QD，转换如下：

$$QB0 = (QX0.0 \sim QX0.7)$$

$$QW0 = (QB0 \sim QB1) = ((QX0.0 \sim QX0.7) + (QX1.0 \sim QX1.7))$$

$$\begin{aligned} QD0 = (QW0 \sim QW1) &= (QB0 \sim QB3) = ((QX0.0 \sim QX0.7) + (QX1.0 \sim QX1.7) + (QX2.0 \sim QX2.7) \\ &+ (QX3.0 \sim QX3.7)) \end{aligned}$$

### 寄存器地址索引规则

按bit寻址	按Byte寻址	按Word寻址	按Dword寻址	按bit寻址	按Byte寻址	按Word寻址	按Dword寻址
QX0.0	QB0	QW0	QD0	MX0.0	MB0	MW0	MD0
QX0.1				MX0.1			
QX0.2				MX0.2			
QX0.3				MX0.3			
QX0.4				MX0.4			
QX0.5				MX0.5			
QX0.6				MX0.6			
QX0.7				MX0.7			
QX1.0	QB1	QW1	QD1	MX1.0	MB1	MW1	MD1
QX1.1				MX1.1			
QX1.2				MX1.2			
QX1.3				MX1.3			
QX1.4				MX1.4			
QX1.5				MX1.5			
QX1.6				MX1.6			
QX1.7				MX1.7			
QX2.0	QB2	QW2	QD2	MX2.0	MB2	MW2	MD2
QX2.1				MX2.1			
QX2.2				MX2.2			
QX2.3				MX2.3			
QX2.4				MX2.4			
QX2.5				MX2.5			
QX2.6				MX2.6			
QX2.7				MX2.7			
QX3.0	QB3	QW3	QD3	MX3.0	MB3	MW3	MD3
QX3.1				MX3.1			
QX3.2				MX3.2			
QX3.3				MX3.3			
QX3.4				MX3.4			
QX3.5				MX3.5			
QX3.6				MX3.6			
QX3.7				MX3.7			
QX4.0	QB4	QW4	QD4	MX4.0	MB4	MW4	MD4
QX4.1				MX4.1			

AM600的Word型寄存器的起始地址为偶数Byte地址；DWord型寄存器的起始地址为偶数Word地址对齐，其索引号呈2倍关系，这样方便地址的计算。

#### 4.5.10 Modbus通信帧格式说明

- 读线圈

采用0x01命令码，可以读取Q变量；

采用0x31命令码，可以读取SM变量。

请求帧格式：从机地址+0x01+线圈起始地址+线圈数量+CRC检验

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x01/0x31 (命令码)	1个字节	读线圈
3	线圈起始地址	2个字节	高位在前，低位在后，见线圈编址
4	线圈数量N	2个字节	高位在前，低位在后
5	CRC校验	2个字节	高位在前，低位在后

响应帧格式：从机地址+0x01+字节数+线圈状态+CRC检验

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x01/0x31 (命令码)	1个字节	读线圈
3	字节数	1个字节	值：[ (N+7) /8]
4	线圈状态	[ (N+7) /8]个字节	每8个线圈合为一个字节，最后一个若不足8位，未定义部分填0。前8个线圈在第一个字节，最地址最小的线圈在最低位。依次类推
5	CRC校验	2个字节	高位在前，低位在后

- 读寄存器

采用0x03命令码，可以读取M变量；

采用0x33命令码，可以读取SD变量。

请求帧格式：从机地址+0x03+寄存器起始地址+寄存器数量+CRC检验

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x03/0x33 (命令码)	1个字节	读寄存器
3	寄存器起始地址	2个字节	高位在前，低位在后，见寄存器编址
4	寄存器数量	2个字节	高位在前，低位在后，N
5	CRC校验	2个字节	高位在前，低位在后

响应帧格式：从机地址+0x03+字节数+寄存器值+CRC检验

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x03/0x33 (命令码)	1个字节	读寄存器
3	字节数	1个字节	值：N*2
4	寄存器值	N*2个字节	每两字节表示一个寄存器值，高位在前低位在后。寄存器地址小的排在前面
5	CRC校验	2个字节	高位在前，低位在后

- 写单个线圈

采用0x05命令码，可以写Q变量。

采用0x35命令码，可以写SM变量。

请求帧格式：从机地址+0x05+线圈地址+线圈状态+CRC检验

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x05/0x35 (命令码)	1个字节	写单线圈
3	线圈地址	2个字节	高位在前，低位在后，见线圈编址
4	线圈状态	2个字节	低位在前，高位在后。非0即为有效
5	CRC校验	2个字节	高位在前，低位在后

响应帧格式：从机地址+0x05+线圈地址+线圈状态+CRC检验

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x05/0x35 (命令码)	1个字节	写单线圈
3	线圈地址	2个字节	高位在前，低位在后，见线圈编址
4	线圈状态	2个字节	低位在前，高位在后。非0即为有效
5	CRC校验	2个字节	高位在前，低位在后

- 写单个寄存器

采用0x06命令码，可以写M变量。

采用0x36命令码，可以写SD变量。

请求帧格式：从机地址+0x06+寄存器地址+寄存器值+CRC检验

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x06/0x36 (命令码)	1个字节	写单寄存器
3	寄存器地址	2个字节	高位在前，低位在后，见寄存器值编址
4	寄存器值	2个字节	高位在前，低位在后。非0即为有效
5	CRC校验	2个字节	高位在前，低位在后

响应帧格式：从机地址+0x06+寄存器地址+寄存器值+CRC检验。

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x06/0x36 (命令码)	1个字节	写单寄存器
3	寄存器地址	2个字节	高位在前，低位在后，见寄存器编址
4	寄存器值	2个字节	高位在前，低位在后。非0即为有效
5	CRC校验	2个字节	高位在前，低位在后

- 写多个线圈

采用0x0f命令码，可以写连续的多个Q变量。

采用0x3f命令码，可以写连续的多个SM变量。

请求帧格式：从机地址+0x0f+线圈起始地址+线圈数量+字节数+线圈状态+CRC检验。

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x0f/0x3f (命令码)	1个字节	写多个单线圈
3	线圈起始地址	2个字节	高位在前，低位在后，见线圈编址
4	线圈数量N	2个字节	高位在前，低位在后。最大为1968
5	字节数	1个字节	值：值： [ (N+7) /8]

序号	数据(字节)意义	字节数量	说明
6	线圈状态	[ (N+7) /8]个字节	每8个线圈合为一个字节，最后一个若不足8位，未定义部分填0。前8个线圈在第一个字节，最地址最小的线圈在最低位。依次类推
7	CRC校验	2个字节	高位在前，低位在后

响应帧格式：从机地址+0x05+线圈起始地址+线圈数量+CRC检验

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x0f/0x3f (命令码)	1个字节	写多个单线圈
3	线圈起始地址	2个字节	高位在前，低位在后，见线圈编址
4	线圈数量	2个字节	高位在前，低位在后。
5	CRC校验	2个字节	高位在前，低位在后

- 写多个寄存器

采用0x10命令码，可以写连续的多个M变量。

采用0x40命令码，可以写连续的多个SD变量。

请求帧格式：从机地址+0x10+寄存器起始地址+寄存器数量+字节数+寄存器值+CRC检验。

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x10/0x40 (命令码)	1个字节	写多个寄存器
3	寄存器起始地址	2个字节	高位在前，低位在后，见寄存器编址
4	寄存器数量	2个字节	高位在前，低位在后，数量为N
5	字节数	1个字节	值：N*2
6	寄存器值	N*2 (N*4)	
7	CRC校验	2个字节	高位在前，低位在后

响应帧格式：从机地址+0x05+线圈起始地址+线圈数量+CRC检验。

序号	数据(字节)意义	字节数量	说明
1	从机地址	1个字节	取值1~247
2	0x10/0x40 (命令码)	1个字节	写多个寄存器
3	寄存器起始地址	2个字节	高位在前，低位在后，见寄存器编址
4	寄存器数量	2个字节	高位在前，低位在后，N
5	CRC校验	2个字节	高位在前，低位在后

## 4.6 串口自由协议使用

### 4.6.1 概要

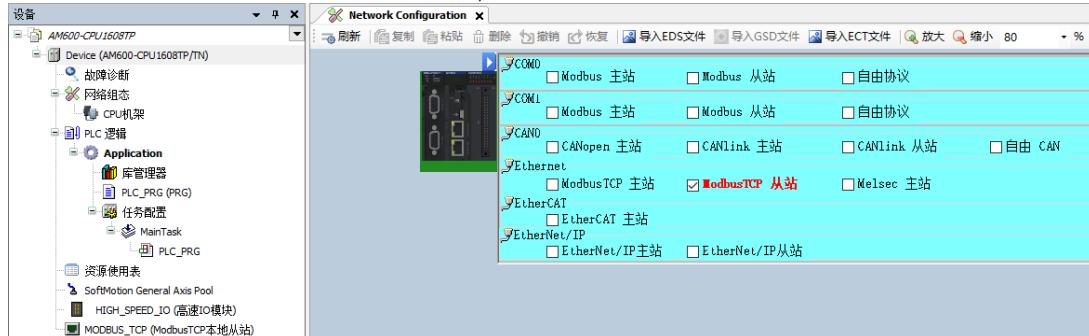
本节针对中型PLC的串口自由协议通讯进行说明，适用的版本要求如下：

设备名称	版本要求
InoProshop	1.2.0及以上
PLC固件	1.19.70以上

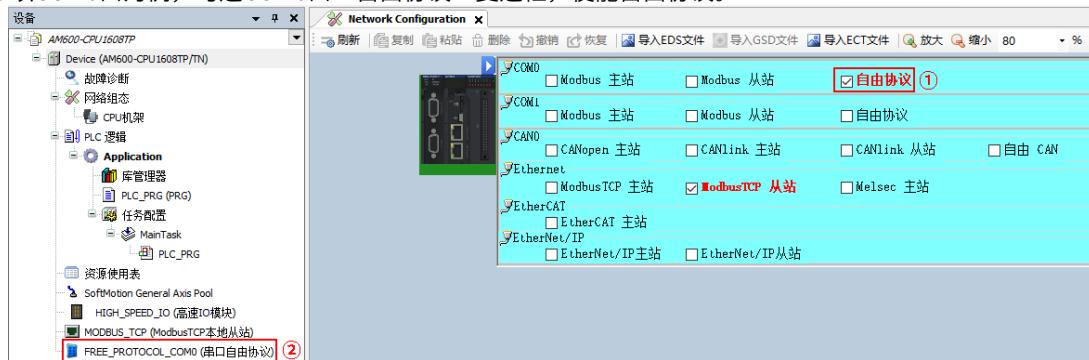
## 4.6.2 串口配置

- AM400/AM600/AC700/A800使能自由协议

- 在左侧设备树中双击“网络组态”，进入“Network Configuration”界面。
- 单击PLC设备图片，显示PLC内部所支持的主/从站和自由协议的使能信息。

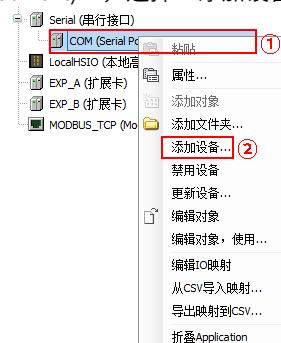


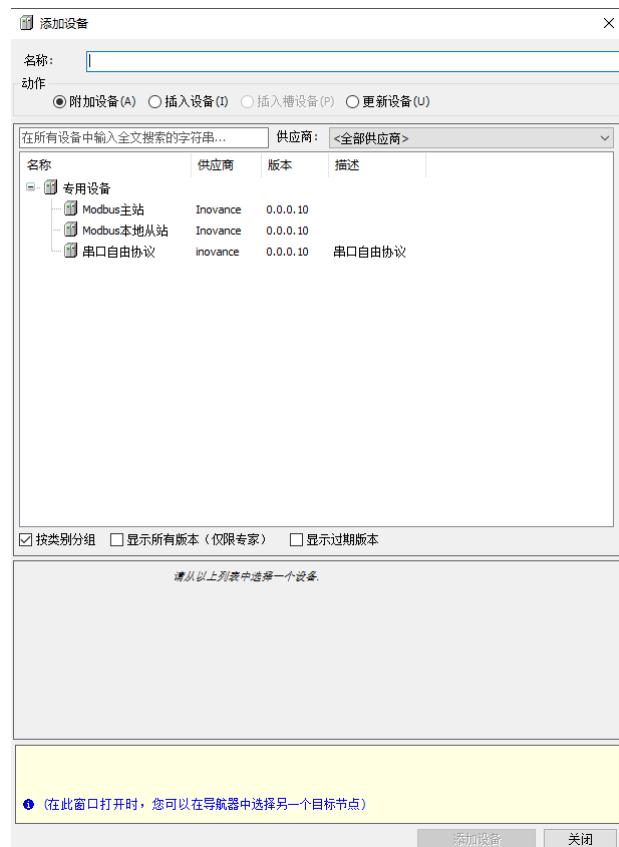
- 以COM0口为例，勾选COM0口“自由协议”复选框，使能自由协议。



- AM300/AM500使能自由协议

- 在左侧设备树中右键单击“COM(Serial Port)”，选择“添加设备”，打开“添加设备”对话框。





2. 选择“串口自由协议”，单击“添加设备”，使能PLC自由协议。





### 4.6.3 通讯配置



配置参数如下表所示：

配置项	功能
端口号	该主站物理连接选择串口0或串口1
波特率	通信时的速率
奇偶校验	通信帧的校验方式
数据位	通信帧包含的实际数据位
停止位	通信时标识单个包的最后位

配置【自由协议】串口的通讯格式，需要和链接的从站设备通讯格式一致，比如：9600 8 E 1。

### 4.6.4 数据发送与接收寄存器

端口作为自由协议的配置如下图：



图4-20 自由协议配置

配置项	功能
寄存器类型	可以选择%MW或SD
接收字节数寄存器	当有数据接收时此寄存器将显示接收字节数
接收数据起始地址	接收到数据缓存的起始寄存器
最大接收长度	最大缓存字节数 最大值为1024字节
发送字节数寄存器	此寄存器不为0时，将触发数据发送，发送完数据后自动复位
发送数据起始地址	将要发送的数据缓存起始寄存器
最大发送长度	一次最多发送的数据字节数 最大值为1024字节

上图已配置“%MW0”，则“%MW0”的数值就是记录接收到外部设备发送的数据帧长度；MW0 需要用户清零，如果不清零将会一直累加，直到最大接收后清零，并且接收缓存会从开始覆盖。

比如，接收数长度“%MW0”=10，那么接收数据的缓存是：“%MW1” ~ “%MW5”；（1个%MW占用了2个字节）。

比如，发送数据长度“%MW300”=8，那么发送数据的缓存是：“%MW301” ~ “%MW304”；当“%MW300”不为0时，会自动触发发送，发送完成后“%MW300”会自动清零。

#### 应用示例：

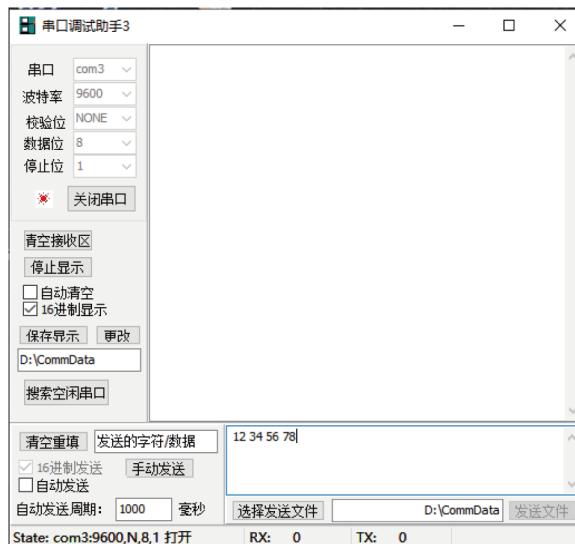
如果上图设置端口后，操作过程如下：

1. 当有数据接收时，“%MW0”会显示接收到的数据字节数，接收到的数据将存放在“%MW1”开始的寄存器里面。
2. 每次接收到数据后，用户需要手动把“%MW0”里面的数据清零，以便数据可以重新以“%MW1”开始重新缓存。如果不清除“%MW0”那么数据将依次缓存。
3. 当接收到的数据超过设置的最大接收长度256个字节时，“%MW0”会重新计数，接收到的数据也将从“%MW1”开始重新缓存。
4. 当有数据要发送时，首先把需要发送的数据写入以“%MW301”起始的寄存器里面。
5. 在“%MW300”中写入要发送的数据字节数后，便开始发送以“%MW301”开始的数据。
6. 数据发送完后，“%MW300”将会自动清零。
7. 当写入“%MW300”的数据个数大于设置的最大发送长度256时，数据将按照最大长度发送。

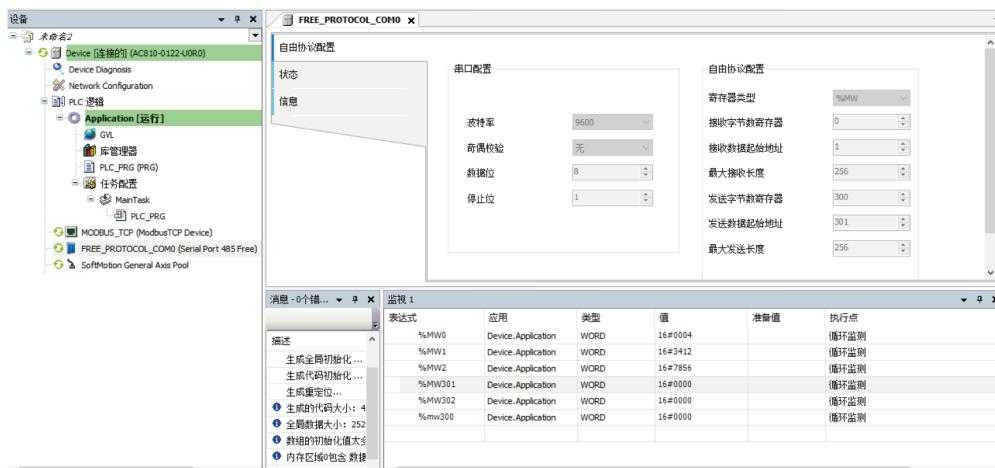
## 4.6.5 串口调试助手模拟通讯

### 1. PLC接收数据

首先打开【串口调试助手】，配置串口通讯参数。要求与后台串口自由协议参数配置保持一致，后台参数波特率=9600、校验方式=无、数据位=8、停止位=1，则对应串口参数配置如下图所示。



使用串口助手发送16进制“12 34 56 78”4个字节长度的数据，在监视PLC对应缓存区变量时，可以看到PLC接收到的数据长度“%MW0”=4个字节，接收到的数据缓存为：“%MW1~%MW2”。如果通过串口助手再次发送数据，PLC端下次接收前需要手动清零“%MW0”，否则接收缓存不会更新。



### 2. PLC发送数据。

如果PLC端要发送4个字节长度的数据，则对应发送数据缓存区%MW301~%MW302写入需发送的数据值，将PLC发送数据长度缓存区%MW300写入4，即可自动发送数据，发送成功后“%MW300”会自动清零，此时串口调试助手接收区域即能接收到PLC发送的对应数据。

## 4.7 Modbus TCP 设备编辑器

### 4.7.1 概述

单击网络组态中的PLC设备，将显示PLC内部所支持的主/从站的使能窗口，如下图所示，单击窗口中的复选框按钮来使能CPU所支持的主/从站功能，再从视图右侧的“网络设备列表”中单击“MODBUS\_TCP”将从站添加到网络中。

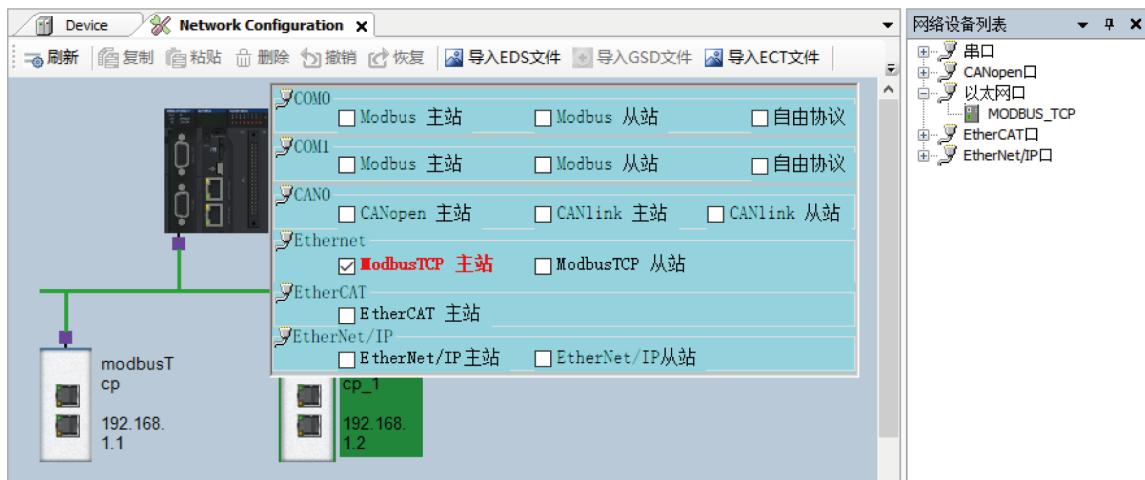


图4-21 Modbus TCP组态配置示例

此时，在界面左侧视图中将出现Modbus TCP组态配置对应设备树，如下图所示：

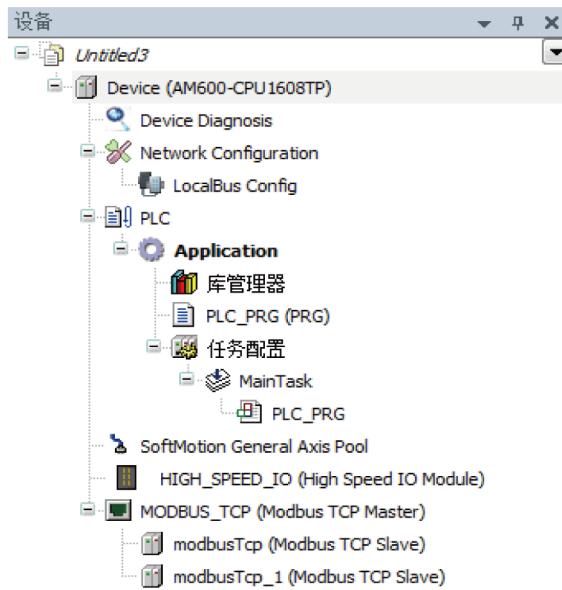


图4-22 Modbus TCP组态配置对应设备树示例

中型PLC的Modbus TCP通信规格，具体请参见各产品的《用户手册》

### 4.7.2 Modbus TCP主站配置

PLC做Modbus TCP主站时，双击设备树中的主站设备，打开Modbus主站配置窗口，如下图所示。

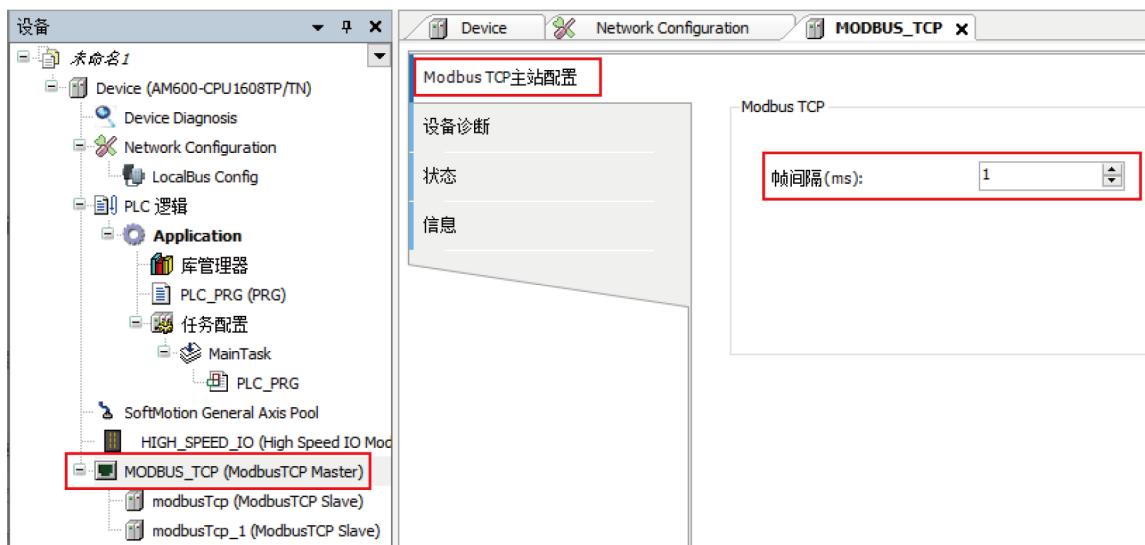


图4-23 Modbus TCP主站配置

帧间隔指主站接收上一个响应数据帧到下一个请求数据帧之间等待的时间间隔。这个参数可用于调节数据交换率。

### 4.7.3 Modbus TCP 主站通信配置

#### Modbus TCP从站设置

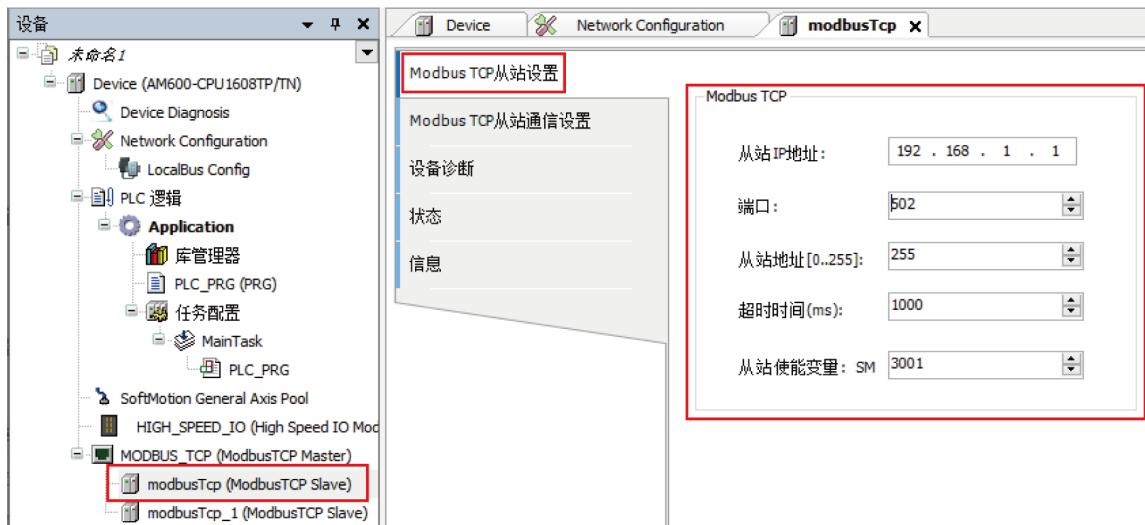


图4-24 Modbus TCP主站连接从站配置

配置参数：

配置项	功能
从站IP地址	主站连接Modbus TCP从站的IP地址
端口	主站连接Modbus TCP从站的TCP端口号
从站地址	主站连接Modbus TCP从站的协议站地址
超时时间	主站发帧后，超过该时间从站未响应，主站报接收超时
从站使能变量	编程使能该变量后，主站开始向该从站发送通信帧

示例：

配置项	配置值
从站IP地址	192.168.10.16
端口	502
从站地址	05
超时时间	1000
从站使能变量	3001

### Modbus TCP主站连接Modbus TCP从站通信设置

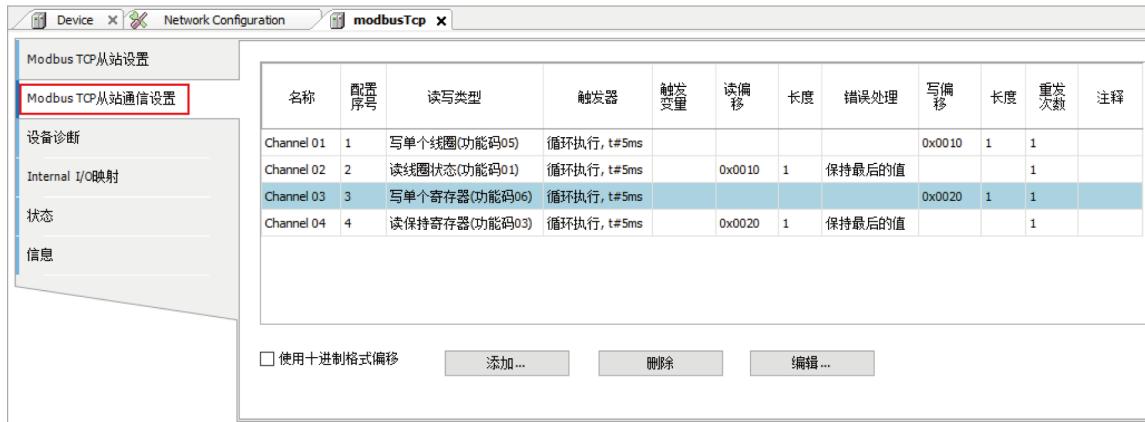


图4-25 Modbus TCP主站连接Modbus TCP从站通信设置

如上图所示，每一个通道代表一个独立的Modbus TCP请求，第三行定义了一个以5ms周期循环写单个寄存器（功能码06）的操作，向偏移地址为0x0020的寄存器写长度为2字节的数据（1个寄存器占2个字节）。

点击“添加…”后会出现一个为Modbus TCP从站添加新通道的对话框，点击“确定”按钮可新建一个通道。

在“Modbus TCP从站通道”列表中选择一个通道，单击“编辑…”按钮将会出现一个对话框，通过修改其中的参数可改变通道的配置，点击“确定”按钮可更新通道设置。

当添加或者编辑通道时，对话框中有以下参数可供使用：



图4-26 Modbus TCP主站连接Modbus TCP从站通信设置对话框

### Modbus 通信设置参数

配置项	功能	
名称	通道命名的字符串	
存取类型	读线圈状态 (功能码01) 读输入状态 (功能码02) 读保持寄存器 (功能码03) 读输入寄存器 (功能码04) 写单个线圈 (功能码05) 写单个寄存器 (功能码06) 写多个线圈 (功能码15) 写多个寄存器 (功能码16)	
触发器	循环执行：周期触发的请求	循环时间：设置时间再次执行
	电平触发：编程进行改变时触发	触发变量 (SM)：设置触发SM元件
重发次数	本次发生通信故障未获得从站返回帧，则按重发次数进行重新发送	
注释	可以对数据进行描述的简短文本区域	
读寄存器	—	
起始地址	读取的寄存器开始位置	
长度	读取的寄存器个数	
错误处理	保持最后的值：使数据保持最后一次的有效值 设置为0：使所有值归零	
写寄存器	—	
起始地址	写寄存器开始位置	
长度	写寄存器长度	

“长度”参数的有效范围取决于以下功能码：

功能码	类型访问	寄存器数
01	读线圈状态	1~2000
02	读输入状态	1~2000
03	读保持寄存器	1~125
04	读输入寄存器	1~125
05	写单个线圈	1
06	写单个寄存器	1
15	写多个线圈	1~1968
16	写多个寄存器	1~123

### Modbus TCP从站Internal I/O映射

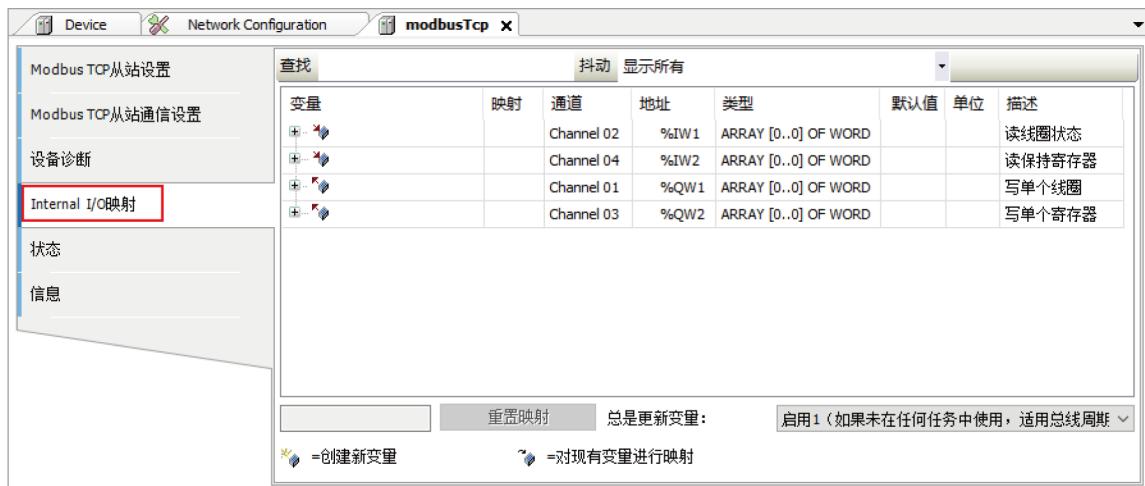


图4-27 Modbus TCP主站连接Modbus TCP从站Internal I/O映射

在Modbus TCP从站通信设置中添加主从站通信配置后，Internal I/O映射中会自动分配每条配置的映射地址，如上图第一行的%IW1表示将读取的一个线圈数值映射到%IW1这个地址。另外，还可以通过输入助手或者直接输入示例变量路径，将程序中的自定义变量映射到I/O地址。

#### 4.7.4 Modbus TCP 从站配置

PLC做Modbus TCP从站时，双击设备树中的从站设备，打开Modbus TCP从站配置窗口，如下图所示：

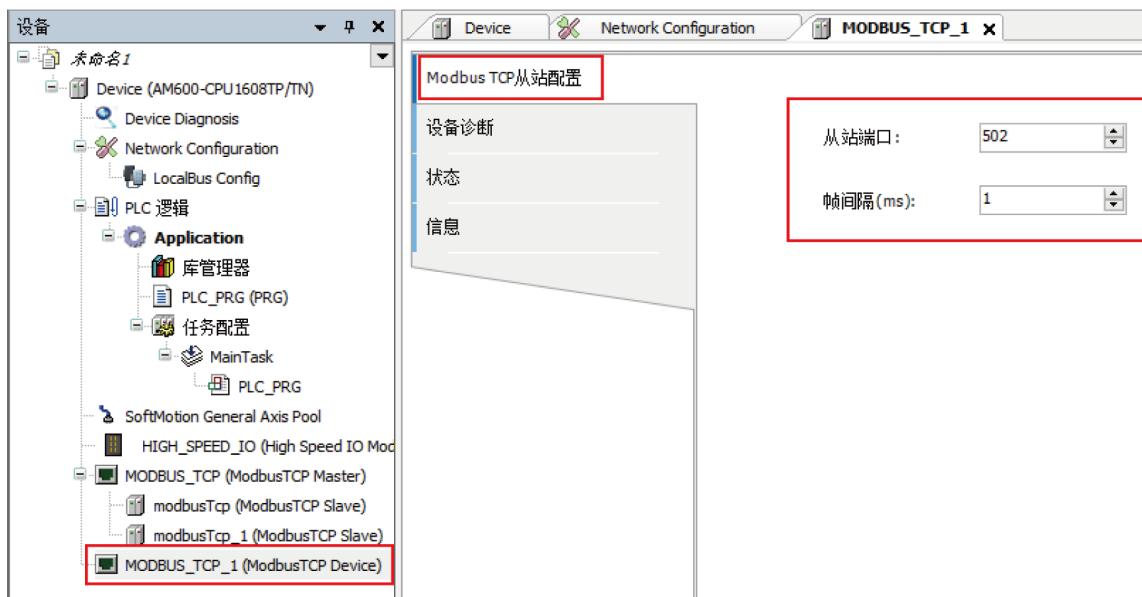


图4-28 Modbus TCP从站配置

配置参数：

配置项	功能
端口	Modbus TCP从站的TCP端口号
帧间隔	Modbus TCP从站接收通信帧后延迟时间发送回复帧

示例：

配置项	配置值
端口	502
帧间隔	1

#### 4.7.5 Modbus TCP 设备诊断

Modbus TCP主站设备诊断

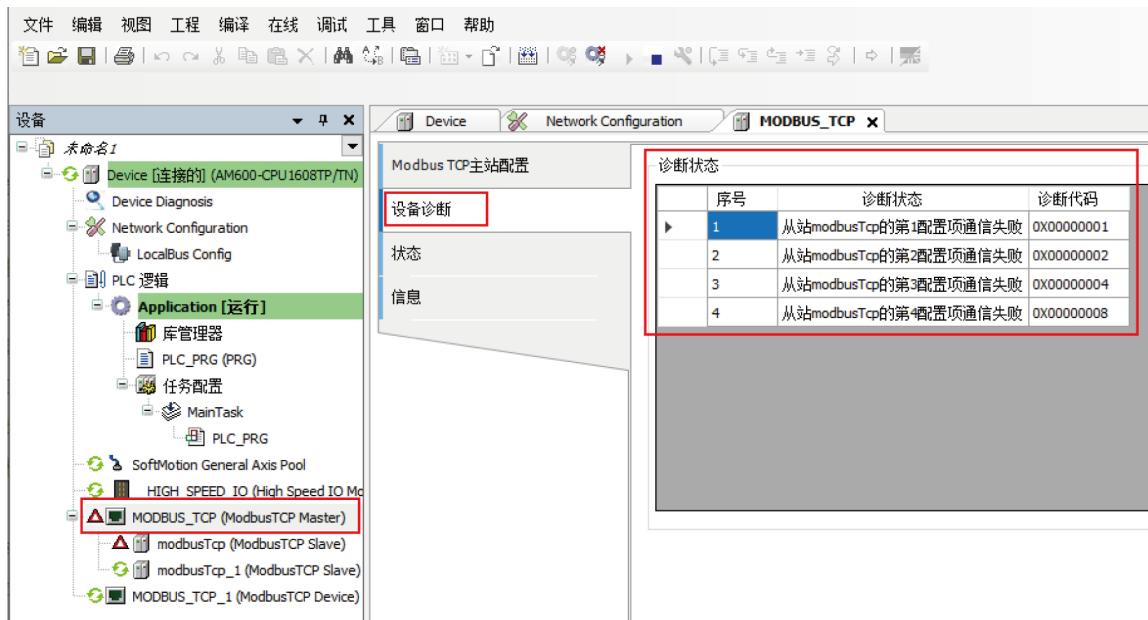


图4-29 Modbus TCP主站诊断

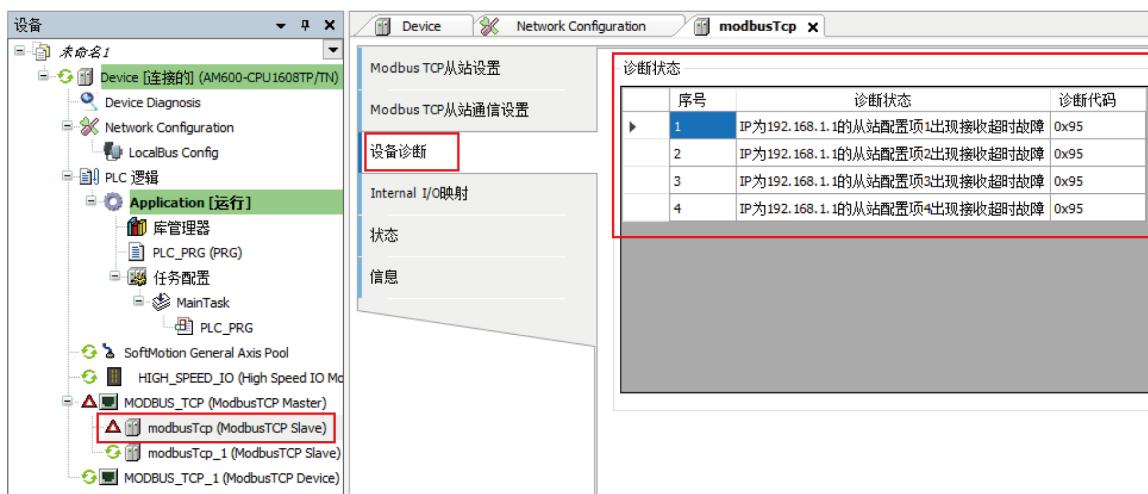


图4-30 Modbus TCP主站连接Modbus TCP从站诊断

Modbus TCP 从站诊断

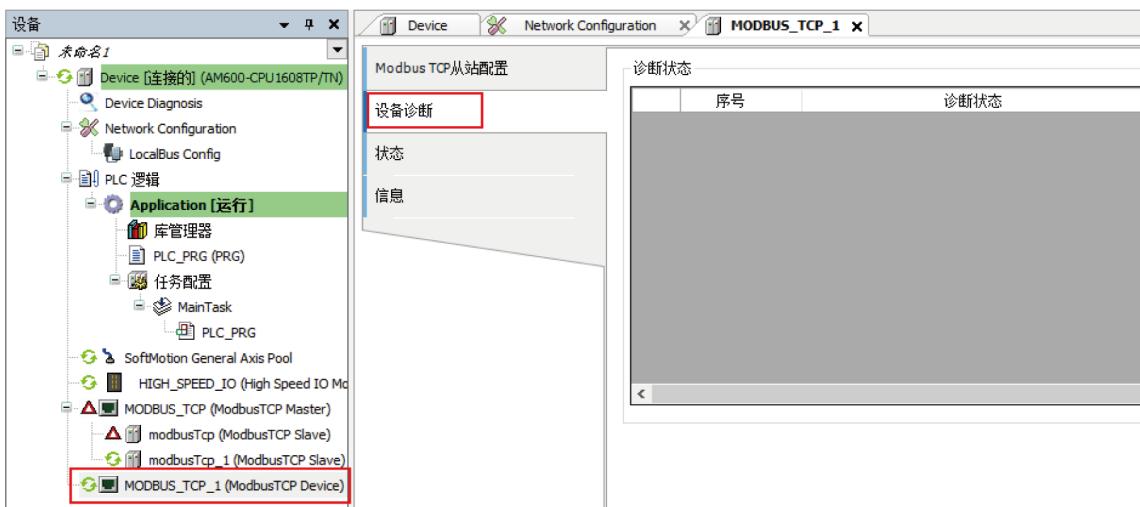


图4-31 Modbus TCP从站诊断

#### 4.7.6 Modbus TCP 常见故障

Modbus TCP主站连接Modbus TCP从站时发生的主要故障如下：

- Modbus TCP主站连接Modbus TCP从站配置IP不正确，导致主站与从站通信无法建立。
- Modbus TCP主站访问Modbus TCP从站非法地址，返回错误应答。
- Modbus TCP主站操作Modbus TCP从站写寄存器，但是Modbus TCP从站寄存器只支持读不支持写操作，Modbus TCP主站会收到Modbus TCP从站返回的出错应答。

错误帧格式：事务元标识符+协议标识符+长度+从机地址+（命令码+0x80）+错误码+CRC校验。

本错误帧适合所有的操作命令帧。

序号	数据（字节）意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	命令码 + 0x80	1个字节	错误命令码
6	错误码	1个字节	1~4

#### 4.7.7 Modbus TCP 变量编址

线圈：位变量，只有两种状态0和1。本PLC中包含Q区及SM区等变量。

变量名称	命令码	起始地址	线圈数量	说明
QX0.0-QX8191.7	0X01, 0x05, 0x0f	0	65536	通用标准Modbus协议都可以访问
IX0.0-IX8191.7	0x2	0	65536	通用标准Modbus协议都可以访问
SM0-SM7999	0x31, 0x35, 0x3f	0	8000	与汇川HMI的专用协议，使用不同的功能码

寄存器：16位（字）变量，本PLC中包含M区及SD区等变量。

变量名称	命令码	起始地址	寄存器数量	说明
MW0-MW65535	0x03, 0x06, 0x10	0	65536	通用标准Modbus协议都可以访问
SD0-SD7999	0x33, 0x36, 0x40	0	8000	与汇川HMI的专用协议，使用不同的功能码

说明：

汇川HMI的专用协议使用不同功能码：在访问SM时，使用0x31，0x35，0x3f(在访问位变量的命令的基础上加了0x30)；在访问SD时，使用0x33，0x36，0x40(在访问寄存器变量的命令的基础上加了0x30)。

AM600软元件有Q区，I区，M区这三种，均可以按位，按字节，按字和按双字进行访问，如：%QX、%QB、%QW、%QD，转换如下：

$$QB0 = (QX0.0 \sim QX0.7)$$

$$QW0 = (QB0 \sim QB1) = ((QX0.0 \sim QX0.7) + (QX1.0 \sim QX1.7))$$

$$\begin{aligned} QD0 = (QW0 \sim QW1) &= (QB0 \sim QB3) = ((QX0.0 \sim QX0.7) + (QX1.0 \sim QX1.7) + (QX2.0 \sim QX2.7) \\ &+ (QX3.0 \sim QX3.7)) \end{aligned}$$

#### 寄存器地址索引规则

按bit寻址	按Byte寻址	按Word寻址	按Dword寻址	按bit寻址	按Byte寻址	按Word寻址	按Dword寻址
QX0.0	QB0	QW0	QD0	MX0.0	MB0	MW0	MD0
QX0.1				MX0.1			
QX0.2				MX0.2			
QX0.3				MX0.3			
QX0.4				MX0.4			
QX0.5				MX0.5			
QX0.6				MX0.6			
QX0.7				MX0.7			
QX1.0	QB1	QW0	QD0	MX1.0	MB1	MW1	MD1
QX1.1				MX1.1			
QX1.2				MX1.2			
QX1.3				MX1.3			
QX1.4				MX1.4			
QX1.5				MX1.5			
QX1.6				MX1.6			
QX1.7				MX1.7			
QX2.0	QB2	QW1	QD1	MX2.0	MB2	MW2	MD2
QX2.1				MX2.1			
QX2.2				MX2.2			
QX2.3				MX2.3			
QX2.4				MX2.4			
QX2.5				MX2.5			
QX2.6				MX2.6			
QX2.7				MX2.7			
QX3.0	QB3	QW1	QD1	MX3.0	MB3	MW3	MD3
QX3.1				MX3.1			
QX3.2				MX3.2			
QX3.3				MX3.3			
QX3.4				MX3.4			
QX3.5				MX3.5			
QX3.6				MX3.6			
QX3.7				MX3.7			
QX4.0	QB4	QW2	QD1	MX4.0	MB4	MW2	MD1
QX4.1				MX4.1			

AM600的Word型寄存器的起始地址为偶数Byte地址； DWord型寄存器的起始地址为偶数Word地址对齐， 其索引号呈2倍关系， 这样方便地址的计算。

#### 4.7.8 Modbus TCP 通信帧格式说明

- 读线圈

采用0x01命令码， 可以读取Q变量。

采用0x31命令码， 可以读取SM变量。

请求帧格式： 事务元标识符+协议标识符+长度+从机地址+0x01+线圈起始地址+线圈数。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x01/0x31 (命令码)	1个字节	读线圈
6	线圈起始地址	2个字节	高位在前, 低位在后, 见线圈编址
7	线圈数量N	2个字节	高位在前, 低位在后

响应帧格式：事务元标识符+协议标识符+长度从机地址+0x01+字节数+线圈状态。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x01/0x31 (命令码)	1个字节	读线圈
6	字节数	1个字节	值: [ (N+7) /8]
7	线圈状态	[ (N+7) /8]个字节	每8个线圈合为一个字节, 最后一个若不足8位, 未定义部分填0。前8个线圈在第一个字节, 地址最小的线圈在最低位。依次类推

- 读寄存器

采用0x03命令码, 可以读取M变量。

采用0x33命令码, 可以读取SD变量。

请求帧格式：事务元标识符+协议标识符+长度+从机地址+0x03+寄存器起始地址+寄存器数量。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x03/0x33 (命令码)	1个字节	读寄存器
6	寄存器起始地址	2个字节	高位在前, 低位在后, 见寄存器编址
7	寄存器数量N	2个字节	高位在前, 低位在后

响应帧格式：事务元标识符+协议标识符+长度+从机地址+0x03+字节数+寄存器值。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x03/0x33 (命令码)	1个字节	读寄存器
6	字节数	1个字节	值: N×2
7	寄存器值	N×2个字节	每两字节表示一个寄存器值, 高位在前低位在后。寄存器地址小的排在前面

- 写单个线圈

采用0x05命令码, 可以写Q变量。

采用0x35命令码, 可以写SM变量。

请求帧格式：事务元标识符+协议标识符+长度+从机地址+0x05+线圈地址+线圈状态。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x05/0x35 (命令码)	1个字节	写单线圈
6	线圈地址	2个字节	高位在前，低位在后，见线圈编址
7	线圈状态	2个字节	低位在前，高位在后，非0即为有效

响应帧格式：事务元标识符+协议标识符+长度+从机地址+0x05+线圈地址+线圈状态。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x05/0x35 (命令码)	1个字节	写单线圈
6	线圈地址	2个字节	高位在前，低位在后，见线圈编址
7	线圈状态	2个字节	低位在前，高位在后，非0即为有效

- 写单个寄存器

采用0x06命令码，可以写M变量。

采用0x36命令码，可以写SD变量。

请求帧格式：事务元标识符+协议标识符+长度+从机地址+0x06+寄存器地址+寄存器值。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x06/0x36 (命令码)	1个字节	写单寄存器
6	寄存器地址	2个字节	高位在前，低位在后，见寄存器值编址
7	寄存器值	2个字节	高位在前，低位在后，非0即为有效

响应帧格式：事务元标识符+协议标识符+长度+从机地址+0x06+寄存器地址+寄存器值。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x06/0x36 (命令码)	1个字节	写单寄存器
6	寄存器地址	2个字节	高位在前，低位在后，见寄存器值编址
7	寄存器值	2个字节	高位在前，低位在后。非0即为有效

- 写多个线圈

采用0x0f命令码，可以写连续的多个Q变量。

采用0x3f命令码，可以写连续的多个SM变量。

请求帧格式：事务元标识符+协议标识符+长度+从机地址+0x0f+线圈起始地址+线圈数量+字节数+线圈状态。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x0f/0x3f (命令码)	1个字节	写多个单线圈
6	线圈起始地址	2个字节	高位在前, 低位在后, 见线圈编址
7	线圈数量N	2个字节	高位在前, 低位在后。最大为1968
8	字节数	1个字节	值: [ (N+7) /8]
9	线圈状态	[ (N+7) /8]个字节	每8个线圈合为一个字节, 最后一个若不足8位, 未定义部分填0。前8个线圈在第一个字节, 地址最小的线圈在最低位, 依次类推

响应帧格式: 事务元标识符+协议标识符+长度+从机地址+0x05+线圈起始地址+线圈数。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x0f/0x3f (命令码)	1个字节	写多个单线圈
6	线圈起始地址	2个字节	高位在前, 低位在后, 见线圈编址
7	线圈数量	2个字节	高位在前, 低位在后

- 写多个寄存器

采用0x10命令码, 可以写连续的多个M变量。

采用0x40命令码, 可以写连续的多个SD变量。

请求帧格式: 事务元标识符+协议标识符+长度+从机地址+0x10+寄存器起始地址+寄存器数量+字节数+寄存器值。

序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x10/0x40 (命令码)	1个字节	写多个寄存器
6	寄存器起始地址	2个字节	高位在前, 低位在后, 见寄存器编址
7	寄存器数量	2个字节	高位在前, 低位在后
8	字节数	1个字节	值: N×2
9	寄存器值	N×2 (N×4)	

响应帧格式: 事务元标识符+协议标识符+长度+从机地址+0x05+线圈起始地址+线圈数量。

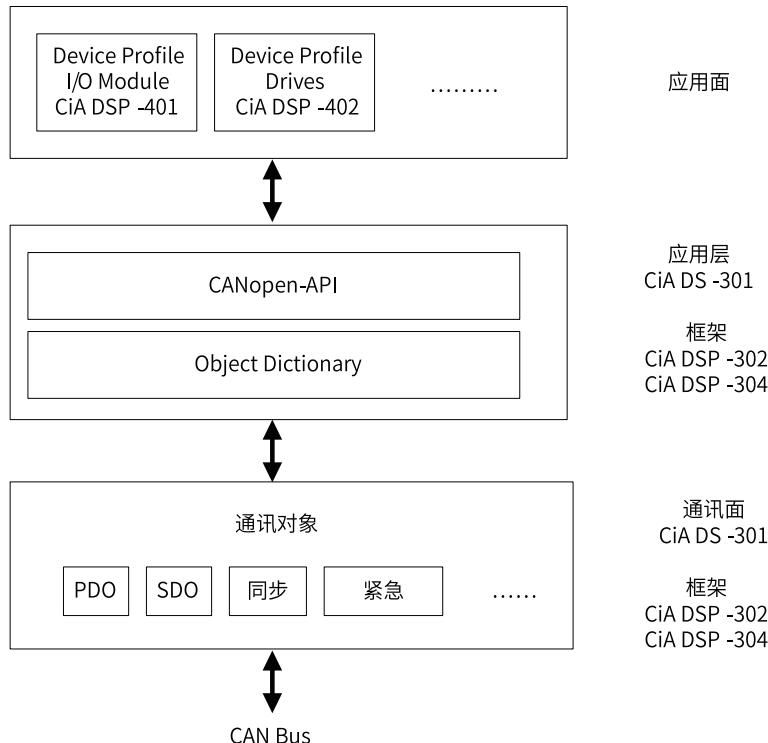
序号	数据(字节)意义	字节数量	说明
1	事务元标识符	2个字节	Modbus 请求/响应事务处理的识别码
2	协议标识符	2个字节	0=Modbus协议
3	长度	2个字节	以下字节的数量
4	从机地址	1个字节	取值1~247
5	0x10/0x40 (命令码)	1个字节	写多个寄存器
6	寄存器起始地址	2个字节	高位在前, 低位在后, 见寄存器编址
7	寄存器数量	2个字节	高位在前, 低位在后

## 4.8 CANopen 网络

### 4.8.1 CANopen 通信简介

#### CANopen总线简介

CANopen是基于CAN总线的分布式自动化控制设备的工业通信协议族。它由制造商与用户协会CiA（CAN in Automation）推广发展，并于2002年底标准化，标准号为CENELEC EN 50325-4。CANopen定义了应用层协议、通信层协议和多种应用协议。CANopen总体结构如下：



应用层为应用提供确认和不带确认的服务，并定义了通信对象。这些服务可用于从一个服务端请求数据。

通信对象用于数据交互。它们用于交换过程数据和服务数据，进程管理或系统时钟同步，错误状态管理和节点状态的控制和监测。这些对象由结构体，传输类型和CAN标识定义。通信对象的典型参数包括：用于数据传输的CAN标识、消息的传输类型、禁止时间或事件时间，通信协议对它们进行了定义。

每一个CANopen设备都以对象字典作为主要的数据结构。对象字典是应用层和CAN总线通信之间的首要数据交互介质。可以通过特定消息从应用层和CAN总线两边访问对象字典的入口。这些对象字典的入口，可以被看做是变量或者程序员定义的区域。

对象字典的每一个入口都具有一个索引和子索引。使用索引结构，可以准确的定位对象字典的入口。

CANopen协议栈提供标准的API函数来定义对象字典的入口，包括这些入口的读写属性。通过通信对象，还可以通过CAN总线访问对象字典。

在对象字典中，必须定义每一个入口的属性，包括数据类型、访问权限、过程数据对象的数据传输及变量范围等。

- PDO (Process data objects)，过程数据对象，用于过程实时数据的传输。
- SDO (Service Data Objects)，服务数据对象，用于访问一个设备的对象字典，对参数进行配置或非实时性数据的传输。
- SYNC消息不包含数据，由生产者向CAN网络周期性的发送，触发节点PDO数据的发送。

- Emergency消息由设备内部致命错误触发，由应急错误代码，错误寄存器和制造商特定错误构成。

CiA DS-301规范详细定义了应用层和通信协议，CiA DSP-302规范详细定义了可编程CANopen设备的框架，CiA DSP-304规范详细定义了安全冗余数据传输的框架，而特定设备的数据描述定义在由各自的设备协议构成的应用协议中，如CiA DSP-401规范定义了I/O模块的数据格式，CiA DSP-402规范定义了驱动控制的数据格式。

## 硬件端口

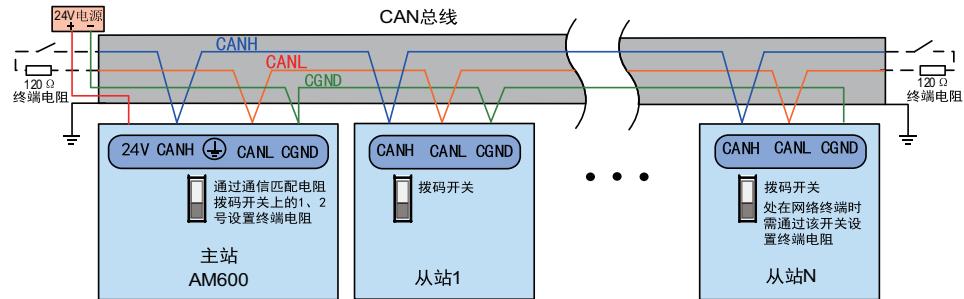
CANopen采用DB9接头进行数据传输，DB9引脚定义如下：

示意图	引脚	信号定义
9	PIN2	CANL
5	PIN7	CANH
4		
8		
3		
7		
2	PIN3	CGND
6		
1		

CAN总线推荐使用带屏蔽双绞线连接，总线两端分别连接两个120Ω终端匹配电阻防止信号反射，屏蔽层一般使用单点可靠接地，固定线缆时不要和交流电源线、高压线缆等捆扎在一起，避免通信信号受干扰影响。

## 组网示意图

CAN总线连接拓扑结构如下图所示：



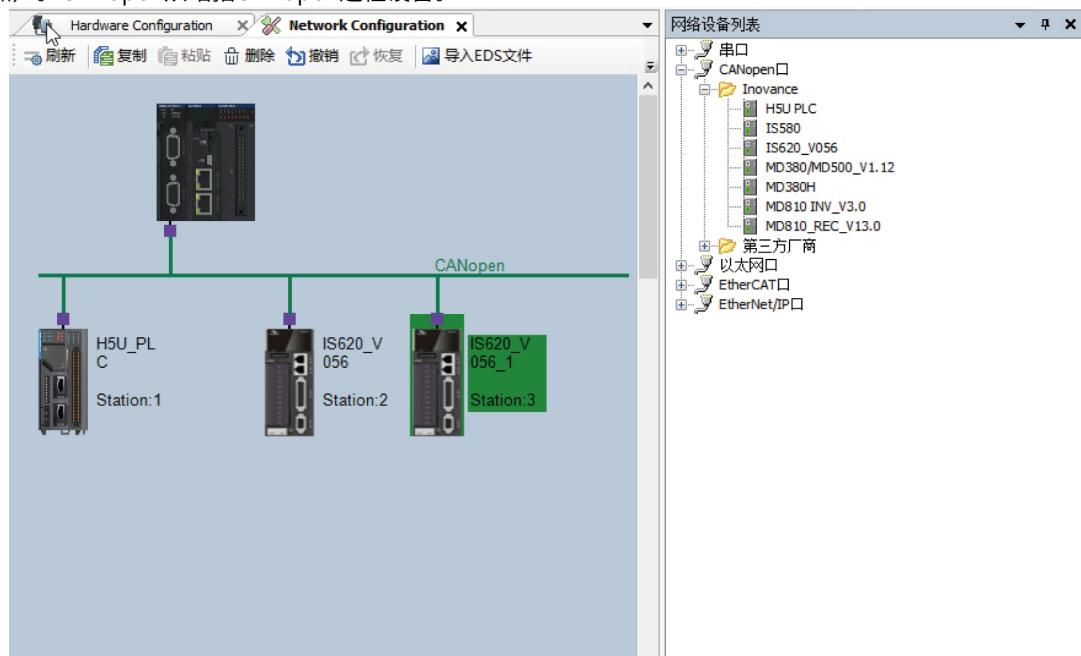
## CANopen使用流程

CANopen一般使用流程如下：

- 设计CANopen硬件网络结构。
- 在网络组态中激活CANopen总线。激活CANopen总线后，会自动添加CANopen主站，并且添加CANopen总线任务“CANopen”，默认CANopen总线使用此任务进行I/O刷新。
- 根据硬件结构在网络组态中添加CANopen从站及模块。如果是第三方从站，可以在网络组态中通过导入EDS文件导入第三方从站，然后添加第三方从站。



4. 如果是AM600从站，需要在硬件组态中添加I/O模块（如下图所示，双击右侧列表中的模块即可完成添加）。CANopen从站指CANopen远程设备。



5. 设置合适的主站配置参数、从站配置参数及模块配置参数。一般情况下，从站节点ID会自动生成，PDO及映射根据EDS文件会自动生成，一些特殊的配置需要手动修改。

配置主站参数及从站参数时，主站波特率与从站的节点ID需要和从站的波特率及从站节点ID拨码开关匹配。

## 网络配置

网络管理

节点 ID:	127	<input type="button" value="检查并修正配置"/>	<input type="button" value="故障停机设置"/>
波特率 (bit/s):	1000000		
<input type="checkbox"/> 程序运行过程中禁止 SDO,NMT 访问		<input checked="" type="checkbox"/> 启动从机	
<input checked="" type="checkbox"/> 自动启动 CANopenManager		<input type="checkbox"/> NMT 启动所有 (如果可能)	
<input checked="" type="checkbox"/> 轮询可选从机			

CANopen

同步

<input type="checkbox"/> 使能同步生产
COB-ID: 16# 80
同步周期 (us): 100000
窗口长度 (us): 0
<input type="checkbox"/> 使能同步消费

心跳

<input type="checkbox"/> 使能心跳生产
节点 ID: 127
生产时间 (ms): 300

从站参数配置

接收 PDO

发送 PDO

服务数据对象

调试

常规

节点 ID: 1
<input type="checkbox"/> 使能专家设置
<input type="checkbox"/> 使能同步发生器

CANopen

常规

节点 ID: 2	<input type="button" value="SDO 通道 (0/0 active)"/>
<input checked="" type="checkbox"/> 使能专家设置	<input type="checkbox"/> 可选设备
<input type="checkbox"/> 创建所有 SDO	<input type="checkbox"/> 未初始化
<input type="checkbox"/> 使能同步发生器	<input type="checkbox"/> 出厂设置
子索引:002 - 恢复通信相关参数	

CANopen

错误控制

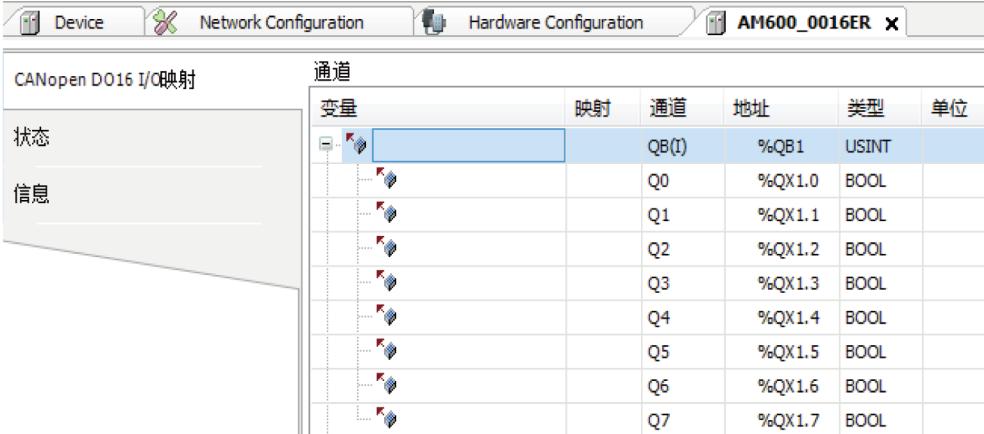
<input type="checkbox"/> 使能节点保护	
保护时间 (ms): 10	
生命周期因子: 2	
<input checked="" type="checkbox"/> 使能心跳生产	
生产时间 (ms): 1000	
<input type="button" value="改变心跳消费属性 (0/0 active)"/>	

使能紧急报文

紧急报文 COB-ID: \$NODEID+16#80

重启时检查

检查供应商 ID     检查产品号     检查版本



The screenshot shows a software interface for configuring CANopen I/O mapping. The main window title is "AM600\_0016ER". The left sidebar has tabs for "Device", "Network Configuration", "Hardware Configuration", and "CANopen DO16 I/O映射". The "CANopen DO16 I/O映射" tab is selected, displaying a table titled "通道" (Channels). The table columns are: 变量 (Variable), 映射 (Mapping), 通道 (Channel), 地址 (Address), 类型 (Type), and 单位 (Unit). The table lists 8 channels (Q0 to Q7) with their corresponding addresses (%QX1.0 to %QX1.7) and types (BOOL).

变量	映射	通道	地址	类型	单位
	QB(I)	Q0	%QX1.0	BOOL	
		Q1	%QX1.1	BOOL	
		Q2	%QX1.2	BOOL	
		Q3	%QX1.3	BOOL	
		Q4	%QX1.4	BOOL	
		Q5	%QX1.5	BOOL	
		Q6	%QX1.6	BOOL	
		Q7	%QX1.7	BOOL	

另外软件还提供了软元件用于获取CANopen从站状态及CiA-DSP405库用于从站管理和操作。

#### 4.8.2 CANopen 主站配置

##### 主站配置

如果主站设备是AC800系列PLC，则需要对转换模块进行配置，如下图所示。

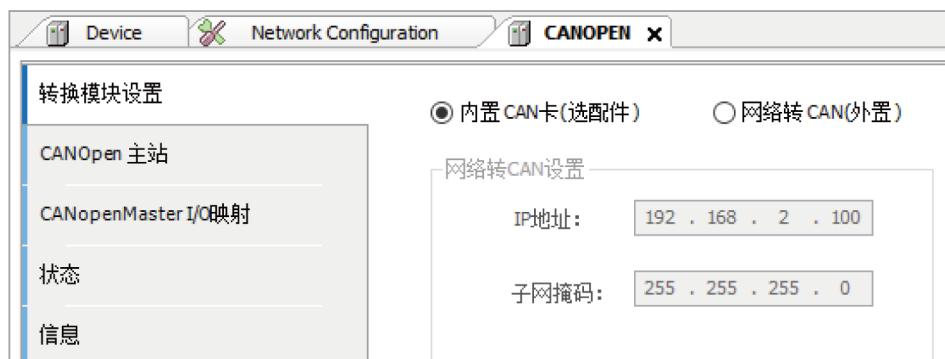


图4-32 CANopen主站配置界面

转换模块有两种：内置CAN卡和网络转CAN。使用内置CAN卡时只需配置模块类型为内置CAN卡；使用网络转CAN时需要配置模块类型、IP地址、子网掩码，网络转CAN目前支持设置的网段应与AC800的网口A或B网段保持一致，否则AC800扫描不到网络转CAN。

##### 说明

网络转CAN模块的IP地址和子网掩码发生修改后，登录时配置信息会生效，网络转CAN模块会重启并重连，请在网络转CAN模块重启完成后、SYS指示灯由红灯变为绿色并闪烁10秒后，再启动应用程序。



### 网络管理

- 节点ID：主站在CANopen网络唯一标示号， 默认127， 范围1-127， 必须是十进制进入数。
- 检查并修正配置：请参见“检查并修正配置”。
- 故障停机设置：请参见“故障停机设置”。
- 波特率：总线上用于传输的波特率。单位Kbit/s,可以设置以下的波特率：10, 20, 50, 100, 125, 250, 500, 800 及 1000。默认值500。

### 说明

如果CPU模块处于网络的首端或末端， CANopen端口的匹配电阻需要拨到ON的位置。通讯距离与波特率有关，要设置合理的波特率。

- 程序运行过程中禁止SDO， NMT 访问：如果这个复选框被激活，在应用程序运行的时候，用户不能通过SDO和NMT访问从站，例如：用户不能在用户程序中或者在从站调试页面通过SDO和NMT访问从站。
- 网络负载：总线运行过程中CANopen网络实时负载。在登陆PLC后才能显示网络负载。

### 同步

- 使能同步生产：如果启用这个选项（默认：禁用）， 主站将发送同步信息。一个CANopen总线系统只能有一个站启用同步生产。同步类型PDO在同步信息发送后根据设置类型发送信息。
- COB-ID：通信对象标识，此设置用于标识同步消息ID。值不能修改，为16#80。如果从站启用了同步生产，使用的也是此COB-ID。
- 同步周期(us)：同步信息以同步周期定义的时间间隔发送，同步周期的单位为微秒，范围为2000us-4294967000us，并且是总线任务时间的整数倍。
- 窗口长度(us)：用于同步PDO，以微秒为单位的时间窗长度。值为0不能修改。

### 心跳

心跳是另外一种节点保护机制：不同于节点守护功能，此功能可以由主站或者从站触发。通常情况下主站发送心跳到从站设备，从站设置消费的主站节点ID，实现从站对主站的监护。

- 使能心跳生产：如果启用这个选项（默认：禁用），主站将发送心跳信息。
- 节点ID：发送心跳信息的唯一标识符，默认为主站节点ID，范围1-127。
- 生产时间(ms)：心跳信息发送的时间间隔，单位为毫秒，范围为2ms-32767ms，并且是总线任务时间的整数倍。
- 窗口长度(us)：用于同步PDO，以微妙为单位的时间窗长度。值为0不能修改。

如果主站是AC800，则需要额外对外置转换模块进行配置，如下图所示。



图4-33 图 CANopen外置模块配置界面

## 检查并修正配置

当多个从站被添加到 CANopen 系统中，由于不同从站EDS文件可能默认配置了COB-ID或者修改了从站的节点ID，可能导致从站或者主站的节点ID重复或者COB-ID冲突。在CANopen主站配置界面点击“检查并修正配置”，再点击“对所有都应用此建议”，可以解决重复的节点ID或者冲突的COB-ID。



图4-34 检查与修正配置界面

重复节点ID。

该框中包含具有相同节点ID的所有从站设备的列表。用户可以通过编辑“节点ID”列来重新分配新的节点ID，修改完成后重复节点ID自动取消。

错误的COB-ID。

该框中显示所有冲突的和非法COB-ID的所有从站设备的列表。用户3种修改COB-ID方式：

- 通过编辑“错误的COB-ID”列，手动修改当前从站索引对应的COB-ID；
- 点击“自动建议”列对应的按钮，按照按钮显示值修改当前从站索引对应的COB-ID；
- 点击“对所有都应用此建议”按钮，按照所有“自动建议”按钮显示值修改所有错误的COB-ID 修改完成后，无错误的COB-ID从站自动取消。

修改完成后，无错误的COB-ID从站自动取消。

## 故障停机设置

故障停机功能，主要是当从站或者模块出现故障或者组态不一致时，从站是否停止运行。此设置为故障停机开关，此功能只支持AM600 CANopen从站。

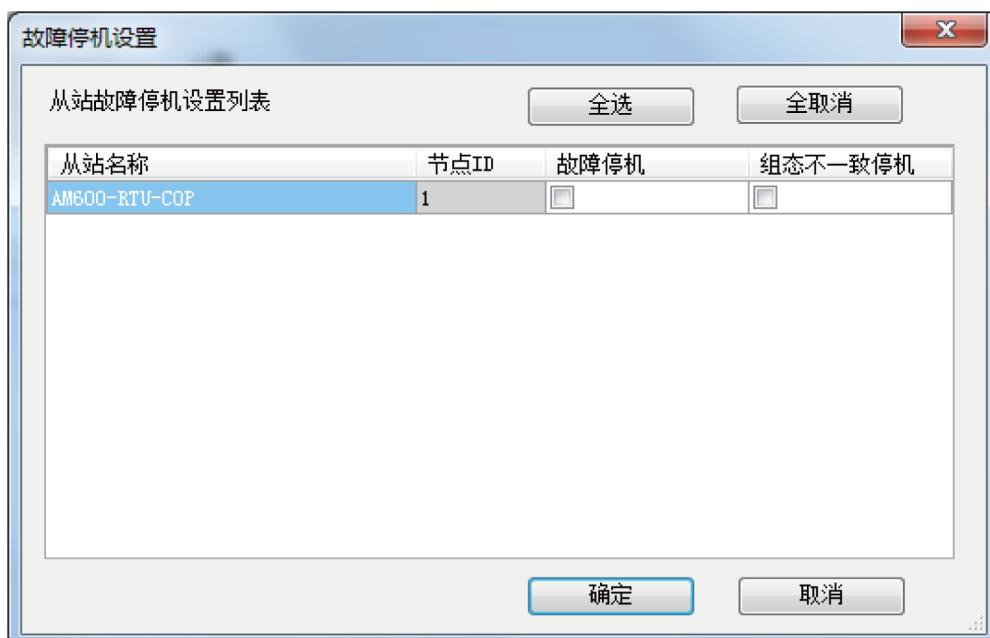


图4-35 故障停机设置

- 从站故障停机设置列表：显示并设置从站故障或者组态不一致时是否停机。
- “故障停机”列控制当从站或者模块出现故障时从站是否停机。如果选中了“故障停机”功能，对于从站，如果从站本身出现故障，从站停止运行；对于模块，如果I/O模块本身的诊断上报功能激活并且模块出现故障，从站停止运行；
- “组态不一致”列控制从站下I/O模块组态不一致从站是否停机，如果选中了“组态不一致停机”，当从站下I/O类型不匹配、模块少于实际模块数、多于实际模块数时，从站停止运行。
- 全选/全取消：激活/不激活故障停机列表所有从站设置。
- 确定/取消：保存/不保存故障停机设置。

## CANopenMaster I/O映射

I/O映射的总体描述和本对话框的使用请参照I/O映射（链接有误）链接。

### 状态

CANopen总线设备或模块的状态配置编辑器可显示状态信息（如：“运行”，“停止”）和内部总线系统 的状态。

### 信息

如果当前设备可用，将会显示下列基本信息：名称、供应商、类型、版本号，模块号和描述。

## 4.8.3 CANopen 从站配置

CANopen从站配置主要是配置从站基本参数、PDO配置、SDO配置及调试功能。

### 从站参数配置



图4-36 从站参数配置

### 常规

- 节点ID：从站在CANopen网络唯一标示号范围1-127（十进制），需要和从站本身标示（如拨码开关）一致。
- SDO通道：暂不支持。
- 使能专家配置：激活此选项，用户可以配置专家参数，如从站节点保护、心跳生产、应急报文、重启检查、PDO映射操作、系统SDO显示、SDO异常跳转。
- 可选设备：暂不支持。

- 创建所有SDO：创建对象字典中具有可写属性的SDO对象，如对象访问属性为rw, wo, rwr, rww。创建的SDO在服务数据对象界面显示。
- 未初始化：暂不支持。
- 使能同步发生器：如果启用这个选项（默认：禁用），此从站将发送同步信息。一个CANopen总线系统只能有一个启用同步生产。同步发送参数使用主站的同步配置参数。
- 出厂设置：如果激活此选项，在下载配置或者配置从站之前，从站参数将被复位。复位参数类型取决于复位类型列表的选择。定义规则如下：
  - “子索引:001”：所有参数都将会被重新复位。
  - “子索引:002”：相关通讯参数（索引1000h - 1FFFh 制造商指定的通讯参数）将会被重新复位。
  - “子索引:003”：相关应用参数（索引 6000h - 9FFFh 制造商指定应用参数）将会被重新复位。
  - “子索引:004” - “子索引:127”：制造商自定义的参数将被复位。
  - “子索引:128” - “Sub254”：保留。

复位参数类型列表内容基于当前的对象字典（EDS文件），来自于EDS文件索引1011的解析，子索引和上述定义规则一一对应。

### 错误控制

错误控制主要用于检测节点的在线状态，包括节点保护和心跳。

节点保护用于主站检测从站的在线状态，主站定时发送节点守护信息，从站响应此信息，如果在节点守护时间（保护时间x生命周期因子）内，从站没有响应，主站认为从站丢失。

心跳可以由从站生产，也可以由主站生产，生产者把心跳报文广播到CAN总线上，心跳消费者消费心跳，如果节点设置了心跳消费，在设定的心跳消费的时间内，没有检测到节点ID对应的心跳生产，则认为此节点丢失。一般从站消费主站的心跳，用于检测主站的在线状态。

- 使能节点保护：激活节点保护功能，节点保护和心跳生产是互斥的。主站在保护时间内定时发送节点保护帧，如果从站没有在节点守护时间（保护时间x生命周期因子）内给出包含特定防护COB-ID（通信对象标识）的响应，则从站认为掉线状态。
- 保护时间：主站定时发送节点保护帧间隔，范围为10ms-65535ms，并且为总线任务周期的整数倍。
- 生命周期因子：和保护时间共同使用，如果在节点守护时间（保护时间x生命周期因子）内，从站没有响应，主站认为从站丢失。范围为1-255。
- 使能心跳生产：激活从站心跳生产，从站以生产时间间隔定时发送心跳生产帧，和节点保护互斥。
- 生产时间：从站发送心跳生产帧间隔，范围为10ms-32767ms，并且为总线任务周期的整数倍。
- 改变心跳消费属性：打开一个对话框，设置从站消费的心跳生产者。通过设置心跳消费，此从站可以检查对应的心跳生产从站在线状态。一般从站消费主站的心跳生产。



图4-37 心跳消费界面

心跳消费配置列表用于配置被消费的心跳生产者。只有使能后才能配置。

使能后，“受保护的节点的NodeID”默认为主站心跳生产节点ID，如果主站没有使能心跳生产，此NodeID为0，范围1-127。心跳时间默认为主站心跳生产时间x 1.5，范围1-65535。

### 紧急报文

- 使能紧急报文：激活从站紧急报文功能，如果这个选项被激活，从站将通过紧急报文COB-ID发送紧急消息。这些紧急信息可以通过CiA405 library (RECV\_EMCY\_DEF, RECV\_EMCY) 函数库提供的函数获取紧急消息。
- 紧急报文COB-ID：从站发送紧急报文的COB-ID，默认为\$NODEID+16#80，NodeID为此从站的节点ID。此COB-ID格式为\$NODEID+16#+16进制数字、16#+16进制数字或者10进制数字。（示例说明）

### 重启时检查

- 检查供应商ID：激活供应商ID检查功能，如果这个选项被激活，从站将检查对象字典中供应商ID（索引1018，子索引01）和从站本身的供应商ID是否匹配，如果不匹配，从站不能正常运行。
- 检查产品号：激活产品号检查功能，如果这个选项被激活，从站将检查对象字典中产品号（索引1018，子索引02）和从站本身的产品号是否匹配，如果不匹配，从站不能正常运行。
- 检查版本：激活版本检查功能，如果这个选项被激活，从站将检查对象字典中版本（索引1018，子索引03）和从站本身的版本是否匹配，如果不匹配，从站不能正常运行。

## 接收PDO

PDO（过程数据对象）用于主站和从站之间的实时数据传输，接收PDO为主站向从站发送的实时数据。

PDO包含通信参数和映射参数。通信参数包括通信唯一标示COB-ID、传输类型、传输控制等。PDO映射参数表示此PDO传输数据来源于对象字典的索引和子索引。

接收PDO来自于对象字典中从索引1400开始到索引1600结束的对象，每个PDO对应的通信参数默认值分别来自于其对应的子索引，接收PDO可映射的对象来自于对象字典中可写访问权限的对象，如访问权限为RW、RWW、WO。

---

### 说明

- 在非专家模式下只能修改PDO通信参数，不能增加、删除PDO及PDO映射；
  - AM600从站只能修改PDO通信参数，不能增加、删除PDO及PDO映射， PDO映射随着AM600 I/O的添加而增加。
- 

接收PDO映射AM600输出模块，每个模块和固定的索引对应，对应关系如下表：

模块类型	索引（16进制）	子索引（16进制）	数据类型
DO16	6300	01-10	unsigned short int
DA4	6411	01-10	short int

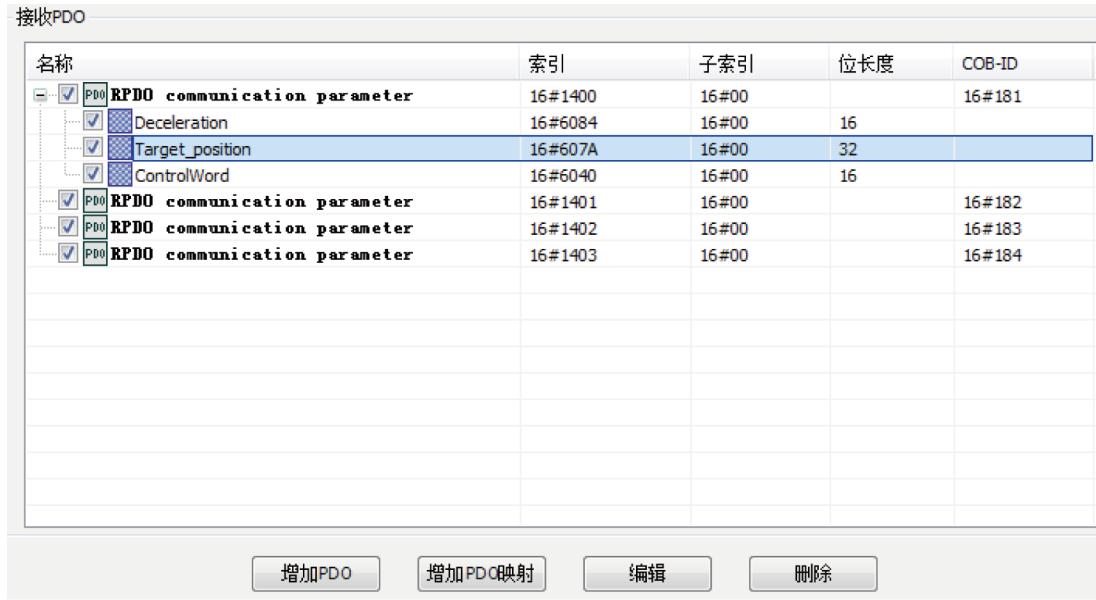


图4-38 接收PDO界面

- 增加PDO：增加一个PDO，增加的PDO添加到最前面。从站中最大接收PDO个数由对象字典索引1400-1600个数决定的，超过最大个数不能添加。添加的PDO名称和索引从对象字典根据使用顺序自动获取，不能修改。  
添加PDO会弹出一个PDO属性设置对话框，详见PDO属性。
- 增加PDO映射：在当前所选的PDO中增加一个PDO映射，此PDO映射增加到当前PDO的后面。PDO映射最大64位，超过64位不能添加。 PDO映射来自于对象字典，接收PDO可映射的对象来自于对象字典中可写访问权限的对象，如访问权限为RW、RWW、WO。对于非AM600从站，添加一个接收PDO映射，会在CANopen添加PDO映射弹出添加对象对话框，详见添加对象。
- 编辑：编辑当前所选的PDO通信参数或者PDO映射参数。如果所选为PDO，编辑PDO通信参数，如果所选为PDO映射，则编辑PDO映射。如果是AM600从站，只能编辑通信参数。
- 删除：删除当前所选的PDO或者PDO映射。如果所选为PDO，则删除PDO，如果所选为PDO映射，则编辑PDO映射。如果是AM600从站，不能执行删除操作。

## 发送PDO

PDO（过程数据对象）用于主站和从站之间的实时数据传输，发送PDO为从站向主站发送的实时数据。

发送PDO来自于对象字典中从索引1800开始到索引1A00结束的对象，每个PDO对应的通信参数默认值分别来自于其对应的子索引，发送PDO可映射的对象来自于对象字典中可读访问权限的对象，如访问权限为RW、RWR、RO、CONST。

### 说明

- 在非专家模式下只能修改PDO通信参数，不能增加、删除PDO及PDO映射；
- AM600从站只能修改PDO通信参数，不能增加、删除PDO及PDO映射， PDO映射随着AM600 I/O的添加而增加。

发送PDO映射AM600输入模块，每个模块和固定的索引对应，对应关系如下表：

模块类型	索引 (16进制)	子索引 (16进制)	数据类型
DI16	6100	01-10	unsigned short int
AD4	6401	01-10	short int

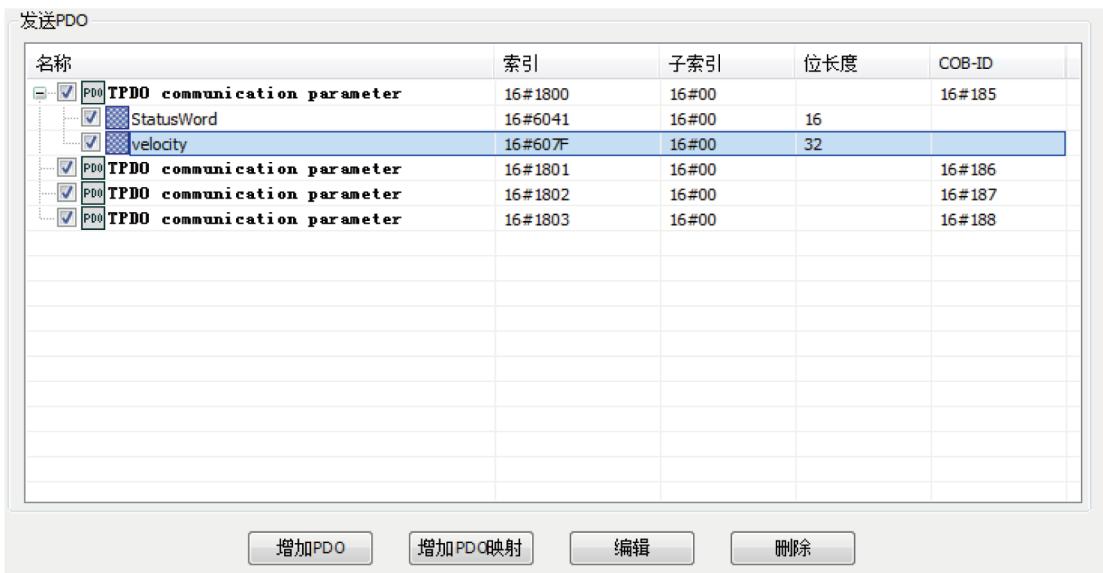


图4-39 发送PDO界面

- 增加PDO：增加一个PDO，增加的PDO添加到最后面。从站中最大发送PDO个数由对象字典索引1800-1A00个数决定的，超过最大个数不能添加。添加的PDO名称和索引从对象字典根据使用顺序自动获取，不能修改。  
添加PDO会弹出一个PDO属性设置对话框，详见PDO属性。
- 增加PDO映射：在当前所选的PDO中增加一个PDO映射，此PDO映射增加到当前PDO的后面。PDO映射最大64位，超过64位不能添加。PDO映射来自于对象字典，发送PDO可映射的对象来自于对象字典中可读访问权限的对象，如访问权限为RW、RWR、RO、CONST。  
添加PDO映射弹出添加对象对话框，详见添加对象。
- 编辑：编辑当前所选的PDO通信参数或者PDO映射参数。如果所选为PDO，编辑PDO通信参数，如果所选为PDO映射，则编辑PDO映射。如果是AM600从站，只能编辑通信参数。
- 删除：删除当前所选的PDO或者PDO映射。如果所选为PDO，则删除PDO，如果所选为PDO映射，则编辑PDO映射。如果是AM600从站，不能执行删除操作。

## 服务数据对象

服务数据对象（SDO）可以在从站初始化过程中进行数据传输，也可以在从站运行过程中进行数据传输，本页面所有配置用于在从站初始化时写入从站。

服务数据对象配置界面可配置选中的SDO，修改SDO的传输顺序，并定义SDO在传输过程中出现错误时的处理方式。

行号	索引:子索引	名称	值	位长度	有错则退出	有错则跳转到行	下一行
32	16#1800:16#01	Disable PDO	16#80000181	32	<input type="checkbox"/>	<input type="checkbox"/>	0
33	16#1801:16#01	Disable PDO	16#80000281	32	<input type="checkbox"/>	<input type="checkbox"/>	0
34	16#1802:16#01	Disable PDO	16#80000381	32	<input type="checkbox"/>	<input type="checkbox"/>	0
35	16#1803:16#01	Disable PDO	16#80000481	32	<input type="checkbox"/>	<input type="checkbox"/>	0
36	16#1804:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
37	16#1805:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
38	16#1806:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
39	16#1807:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
40	16#1808:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
41	16#1809:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
42	16#180A:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
43	16#180B:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
44	16#180C:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
45	16#180D:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
46	16#180E:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
47	16#180F:16#01	Disable PDO	16#80000000	32	<input type="checkbox"/>	<input type="checkbox"/>	0
48	16#6411:16#07	Analog_out_CH 7	0	16	<input type="checkbox"/>	<input type="checkbox"/>	0
49	16#6320:16#02	Digital32_out 2	0	32	<input checked="" type="checkbox"/>	<input type="checkbox"/>	0
50	16#1010:16#03	Save Application Parameters	0	32	<input type="checkbox"/>	<input type="checkbox"/>	0

SDO超时:

图4-40 服务数据对象列表界面

服务数据对象列表：此列表显示从站初始化时写入的所有SDO，其中灰色的SDO为系统自动添加的SDO，位于列表的最上面最先配置，此SDO为从站配置界面参数自动产生，如心跳、节点守护、应急信息、PDO及PDO映射等配置自动产生的。也可以通过下面的“增加”按钮自定义添加SDO。自定义添加的SDO可以修改和上下移动。

可以双击自定义添加的SDO“值”单元格，修改SDO值。

在配置SDO出现错误时可以设置出错处理方式。可以激活“有错则退出”，当配置此SDO出现错误时直接退出配置，下面的SDO不再配置；也可以激活“有错则跳转到行”，当配置此SDO出现错误时，直接跳转到指定的行开始向下执行。如果没有激活这两个选项，则按默认处理方式，直接执行下一个。

## 说明

- 只有在专家模式系统SDO和错误处理才能显示。
- SDO配置错误处理时，需要警惕跳转行处理，如果向上跳转时可能出现死循环配置SDO。

项目	描述
向上	把选择的SDO上移一行，只有自定义添加的SDO才以上移
向下	把选择的SDO下移一行，只有自定义添加的SDO才可以下移
增加	在所选择的SDO上方增加一个SDO。此按钮会弹出“添加对象”对话框。虽然“添加对象”界面和添加PDO界面相似，但添加SDO对象界面会显示SDO值编辑框和注释，可以在此编辑框中编辑SDO值，修改SDO注释
编辑	编辑所选择的SDO，此按钮会弹出“添加对象”对话框，通过此对话框修改SDO信息。只有自定义添加的SDO才可以编辑
删除	删除所选择的SDO，只有自定义添加的SDO才可以删除
SDO超时	设置配置一条SDO的超时时间，默认1000ms，范围0ms-4294967ms

## 调试

调试界面用于从站NMT控制、SDO读写和诊断信息获取。



图4-41 调试页面

### NMT

NMT用于提供网络管理（如初始化、启动和停止节点，侦测失效节点）服务。这种服务是采用主从通讯模式（所以只有一个NMT主节点，主站）来实现。

从站在启动过程中其状态可以用下面的节点状态转换图表示。

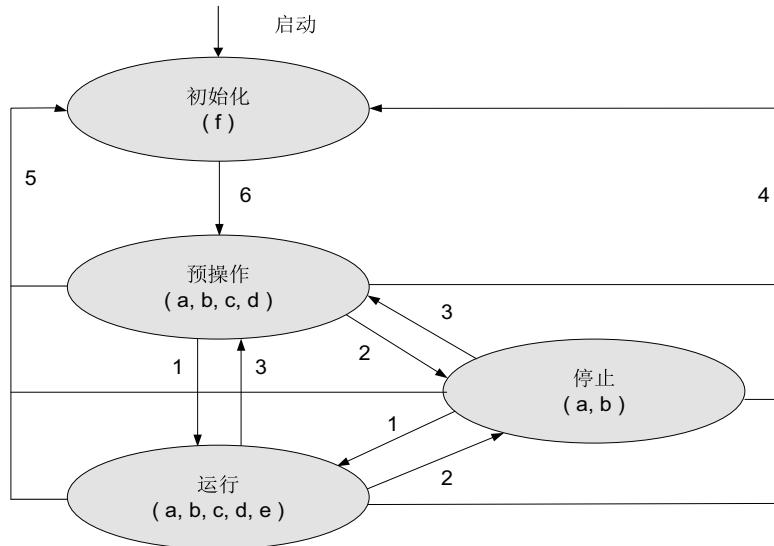


图4-42 从站节点转换图

说明：

a. NMT, b. Node Guard, c. SDO, d. Emergency, e. PDO, f. Boot-up

状态转移（1—5由NMT服务发起）序列，NMT命令字（在括号中）：

- 1: Start\_Remote\_node (0x01, 启动节点)
- 2: Stop\_Remote\_Node (0x02, 停止节点)
- 3: Enter\_Pre-Operational\_State (0x80, 进入试运行)
- 4: Reset\_Node (0x81, 复位节点)
- 5: Reset\_Communication (0x82, 复位通信)
- 6: 设备初始化结束, 自动进入Pre\_Operational状态, 发送Boot-up消息

初始化包括应用数据初始化和通信初始化, 复位节点复位从站节点所有数据, 而复位通信只复位通信数据。

在任何时候NMT服务都可使所有或者部分节点进入不同的工作状态。

- 启动节点: 运行从站节点, 在运行状态下才能进行PDO通信。当从站处于预运行状态或者停止状态时, 启动节点会把从站置于运行状态 ([第263页 “4-42 从站节点转换图”](#) 状态1)。
- 停止节点: 停止从站节点运行, 在此状态下会停止除节点守护和心跳之外的所有通信。当从站处于预运行状态或者运行状态时, 停止节点会把从站置于停止状态 ([第263页 “4-42 从站节点转换图”](#) 状态2)。
- 进入试运行: 从站进入试运行状态, 在此状态下可以进行SDO通信不能进行PDO通信。从站初始化完成后会自动进入试运行状态。当从站处于运行或者停止状态时, 进入试运行会把从站置于试运行状 ([第263页 “4-42 从站节点转换图”](#) 状态3)。
- 复位节点: 重置从站配置数据。首先复位应用配置, 然后复位通信配置数据, 复位完成后自动进入预运行状态 ([第263页 “4-42 从站节点转换图”](#) 状态4)。
- 复位通信: 重置从站通信配置数据。只复位通信配置数据, 复位完成后自动进入预运行状态 ([第263页 “4-42 从站节点转换图”](#) 状态5)。

### 服务数据对象 (SDO)

SDO用来在设备之间传输大的低优先级数据, 典型的是用来配置CANopen网络上的设备, 本页面主要用于从站节点运行过程中, 读取或者写入SDO对象值。读取或者写入一个SDO对象需要确定SDO对象的索引、子索引、位长度, 写入时还需要写入的值。

- 索引: SDO读/写索引, 范围16#0-16#FFFF。
- 子索引: SDO读/写子索引, 范围16#0-16#FF。
- 位长度: SDO读/写位长度, 范围8,16,24,32。
- 数据: SDO读/写的值, 前一个编辑框为16进制值, “=” 后为10进制值。在写入SDO时, 值范围和位长度有关, 最小值0, 最大值为位长度所能表示的最大值。
- 结果: SDO读/写结果, 如果读/写异常, 为异常信息。

### 诊断

显示从站节点的运行状态和应急信息。

- 在线状态: 显示从站在线状态, 从站在线显示绿色背景, 内容为“在线”, 从站离线显示红色背景, 内容为“离线”。
- 运行状态: 显示从站运行状态, 前面图标显示从站状态, 图标后为状态文字。状态对应的图标和文字如下表所示:

状态	图标	文字信息
运行		运行
停止		停止
试运行		预操作

状态	图标	文字信息
初始化		已初始化
未连接		断开连接
正在连接		正在连接...
正在准备		正在准备...
复位节点		复位节点...
复位通信		复位通信...
扫描节点		扫描从站...
配置节点		配置从站...
启动节点		启动从站...
未知状态		未知状态

- 诊断字符串：显示从站当前诊断信息。
- 最近的应急信息：显示当前未确认的第一条应急信息。当由应急信息过来时，如果上一条没有确认，此应急信息不显示，直到上一条确认后，才显示当前应急信息。

一个应急报文由8字节组成，格式如下：

sender receiver(s) COB-ID	Byte 0-1	Byte 2	Byte 3-7
0x080+Node_ID	应急错误代码	错误寄存器 (对象0x1001)	制造商特定的错误区域

16进制的应急错误代码如下表所示。应急错误代码中‘xx’部分由相应的设备子协议定义。

应急错误代码 (16进制)	代码功能描述
00xx	Error Reset 或No Error
10xx	Generic Error
20xx	Current
21xx	Current, device input side
22xx	Current, inside the device
23xx	Current, device output side
30xx	Voltage
31xx	Mains voltage
32xx	Voltage inside the device
33xx	Output voltage
40xx	Temperature
41xx	Ambient temperature
42xx	Device tempearture
50xx	Device hardware
60xx	Device software
61xx	Internal software
62xx	User software
63xx	Data set
70xx	Additional modules
80xx	Monitoring
81xx	communication
8110	CAN overrun
8120	Error Passive

应急错误代码 (16进制)	代码功能描述
8130	Life Guard Error 或Heartbeat Error
8140	Recovered from Bus-Off
82xx	Protocol Error
8210	PDO no processed Due to length error
8220	Length exceedd
90xx	External error
F0xx	Additional functions
FFxx	Device specific

应急信息包含时间、错误码、错误寄存器和制造商指定代码。

- 时间：软件获取应急信息的时间，不是故障发生时间。
- 错误码：应急错误码，鼠标悬浮上，会提示应急错误码对应的应急信息。
- 错误寄存器：应急信息错误寄存器。定义如下表：

位错误寄存器位定义 Bit	错误类型
0	Generic
1	Current
2	Voltage
3	Temperature
4	Communication
5	Device profile specific
6	Reserved(=0)
7	Manufacturer

- 制造商指定代码：应急信息制造商指定代码
- 确认：确认应急信息。应急信息只保留一条，只有确认后，后来的应急信息才接收显示。

## PDO属性

PDO属性用于设置PDO通信的通信参数，包括COB-ID（通信对象标示符）、通信传输方式、抑制时间和事件时间。在编辑或者添加PDO时弹出此对话框。



图4-43 PDO属性对话框

- COB-ID： PDO通信对象标示符。一个CANopen总线内是唯一的，不能和其它通信对象标示符重复（如其它PDO COB-ID、应急COB-ID、心跳COB-ID）。 PDO COB-ID范围为16#180-57F, 681-6DF，如果不合法

可以手动修改或者通过主站检查并修正配置修正。每个PDO默认COB-ID来自于对象字典对应PDO子索引01的对象，如果对象字典格式为\$NodeID+值，则此COB-ID随着从站节点ID的更改自动更改，当手动更改此COB-ID后，自动随节点ID更改特性消失。如果COB-ID对应的对象字典访问权限无写权限，则此COB-ID不能更改。

- RTR：接收远程帧，接收到远程帧后触发PDO发送，只有发送PDO才显示。
- 传输类型： PDO通信传输方式。

PDO可以有多种传送方式：

1. 同步（通过接收SYNC对象实现同步）

非周期：由远程帧触发传送，或者由设备子协议中规定的对象特定事件触发传送。

周期：传送在每1到240个SYNC消息后触发。

2. 异步

由远程帧触发传送；由设备子协议中规定的对象特定事件触发传送。

下表给出了由传输类型定义的不同PDO传输模式，传输类型为PDO通讯参数对象的一部分，由8位无符号整数定义。

传输类型	触发PDO通信条件 (B = both needed O = one or both)			PDO传输
	SYNC	RTR	Event	
0	B	-	B	同步， 非循环
1-240	O	-	-	同步， 循环
241-251	-	-	-	Reserved
252	B	B	-	同步，在RTR之后
253	-	O	-	异步，在RTR之后
254	-	O	O	异步，制造商特定事件
255	-	O	O	异步，设备子协议特定事件

说明：

SYNC –接收到SYNC-object。

RTR –接收到远程帧。

Event –例如数值改变或者定时器中断。

传输类型为：1到240时，该数字代表两个PDO之间的SYNC对象的数目）。

每个PDO默认传输类型来自于对象字典对应PDO子索引02的对象，如果其对应的对象字典访问权限无写权限，则传输类型不能更改。

- 同步数：和传输类型有关，只有传输类型选择1-240时可以修改，表示从站接收到同步数个同步帧后开始处理PDO数据传输。
- 抑制时间:同一个PDO传输两条信息之间的最小间隔时间与100μs的乘积，该值域才可修改。该设置可避免当一个值改变的时候PDO发送太频繁。默认值为0，范围0~65535。只有发送PDO并且传输方式为254或者255才能设置。每个PDO默认抑制时间来自于对象字典对应PDO子索引03的对象。如果抑制时间对应的对象字典访问权限无写权限，则此抑制时间不能更改。
- 事件时间:同一个PDO传输两条信息之间的间隔时间，单位ms，默认值为0，范围0-65535。只有发送PDO并且传输方式为254或者255才能设置。每个PDO默认事件时间来自于对象字典对应PDO子索引05的对象。如果事件时间对应的对象字典访问权限无写权限，则此事件时间不能更改。

## 添加对象

添加对象对话框用于添加、编辑接收PDO映射、发送SDO映射或者SDO。在对SDO操作时，此对话框会增加SDO值编辑框和SDO注释输入框。

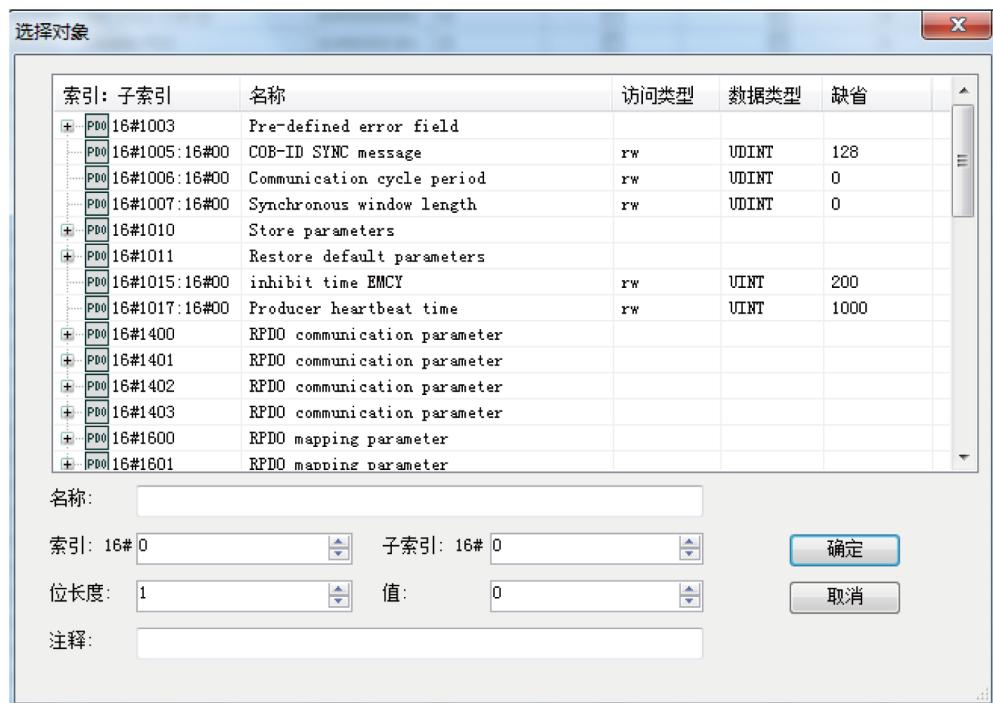


图4-44 添加对象对话框

对象列表：此列表来自于EDS文件中的对象。在接收PDO映射编辑时，只有访问权限为RW、RWW、WO并且索引大于16#2000的对象才能显示；在发送PDO映射编辑时，只有访问权限为RW、RWR、RO、CONST并且索引大于16#2000的对象才能显示；在SDO编辑时，只有访问权限为RW、RWW、RWR、WO的对象才能显示。

## 说明

对于AM600从站SDO编辑时，索引为16#2000-16#40df之间的对象不能显示（这些参数用于模块配置，而模块配置参数在模块本身已经配置）。

- 索引：对象索引，范围16#0-16#FFFF，当选中对象列表对象时，自动显示选中对象索引。
- 子索引：对象子索引，范围16#0-16#FF，当选中对象列表对象时，自动显示选中对象子索引。
- 位长度：对象位长度，范围0-32，当选中对象列表对象时，自动显示选中对象位长度。
- 值：要配置的SDO值，只有编辑SDO时才显示，范围和选中对象数据类型有关，当选中对象列表对象时，自动显示选中对象值。
- 注释：SDO注释，只有编辑SDO时才显示，最大50个字符。

## CANopen从站I/O映射

此页面只有在非AM600从站时显示，AM600从站对应的I/O映射和AM600 I/O模块对应，不再此显示。I/O映射的总体描述和本对话框的使用请参照I/O映射链接。

## 状态

CAN总线设备或模块的状态配置编辑器可显示状态信息（如：“运行”，“停止”）和内部总线系统的状态。

## 信息

如果当前设备可用，将会显示下列基本信息：名称、供应商、类型、版本号，模块号、描述、图像。

### 4.8.4 CANopen模块

#### 模块化设备和非模块化设备

在CANopen从站配置中，CANopen从站节点可接入两种类型的模块设备：

**模块化设备：**接入CANopen从站节点下，模块自带I/O映射列表，不需要使用CANopen从站I/O映射对话框，从站节点PDO映射随着模块添加自动添加。目前AM600 I/O模块为此类型。

**非模块化设备：**从站节点对话框包含I/O映射对话框，PDO映射不能自动配置。

#### AM600 CANopen I/O模块

AM600 CANopen I/O模块在硬件组态中添加，添加一个I/O，自动添加一个PDO映射，和PDO映射关系见接收PDO和发送PDO。添加I/O后，可以配置I/O参数，也可以给I/O添加I/O映射，实现数据刷新，具体见CPU下I/O模块。

### 4.8.5 CANopen参数配置

组态是汇川Inoproshop编程软件中重要一部分，Inoproshop目前仅支持IS620P-CO型号，如需使用其他厂商的驱动器，必须先需要导入其他厂商EDS文件。并确认其他厂商驱动器程序是否严格按照CANopen通讯标志和CIA402标准的设计。调用CANopen402功能块之前，必须完成正确的参数设置。详细如下：

#### 主站配置

主站主要配置有通讯波特率、同步模式、同步时间、心跳、心跳间隔时间。

- **通讯波特率：**波特率越高，通讯效率越高，但是波特率越高，通讯距离越短。一般情况下配置波特率为500k。
- **同步模式：**必须勾选周期同步功能，否则功能块无法工作。
- **同步周期时间：**在只有CANopen轴的功能中，推荐配置是4ms：3个从站，从站PDO配置接收、发送都少于8个字节。这个时间建议与程序中CANopen任务扫描周期时间相同。
- **心跳：**心跳是主站每个心跳时间发送心跳帧，用于从站监控主站是否掉线。此功能必须配合从站同时使用，有些从站未设计心跳检查功能，默认不需要心跳。
- **心跳时间：**主站每隔此设置时间长度，给从站发送心跳帧，默认300ms，在勾选心跳条件下生效。

## 网络配置

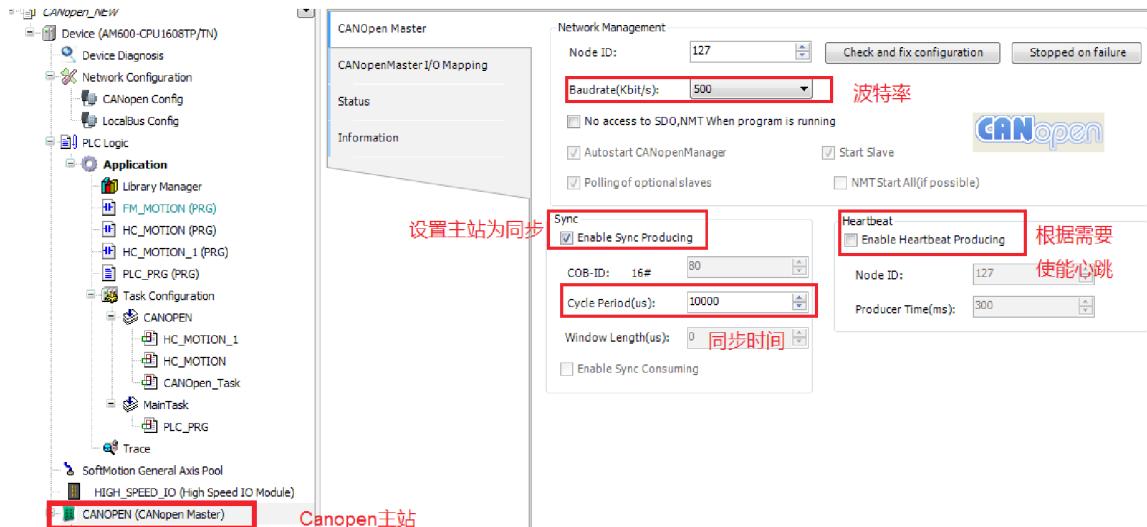


图4-45 CANopen主站配置

## 从站配置

从站主要配置节点ID、心跳、心跳时间、PDO同步方式、同步对象字典配置。

- 节点ID：又称为站号，从站通讯的基本参数，站号与实际的物理站号保持一致。
- 心跳：从站向指定站号发送心跳帧，主要用于其他站点监控本站点的通讯状态，默认勾选。
- 心跳时间：从站每隔此设置时间长度，给指定站号发送心跳帧，默认1000ms，在勾选心跳下生效。
- PDO同步方式：默认为异步方式，需要更改为同步周期模式。
- PDO对象字典配置：PDO对象字典保证从站与主站数据每个总线周期交互一次，数量越多，主站与从站状态交互越高效，但是PDO越多会导致总线负载越大，可能导致总线数据传输滞后、严重情况下会导致掉线。对于轴控设备，根据应用经验，发送和接收PDO必须配置项有6040, 6041, 6060, 6061, 607A, 6064 (6063) , 60FF, 606C, 6081。选配项为6083, 6084, 607C, 6098。

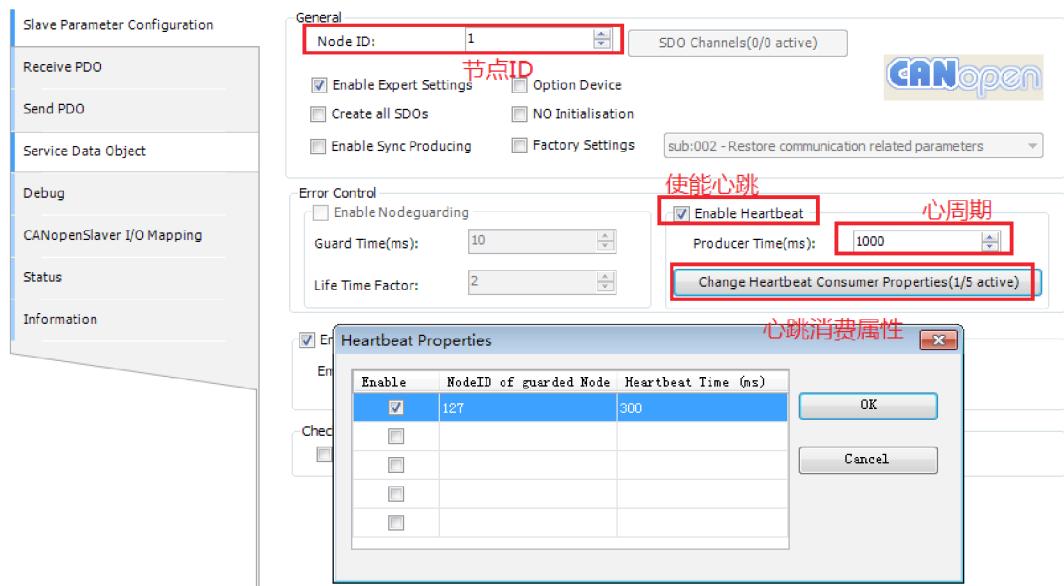


图4-46 CANopen从站参数配置

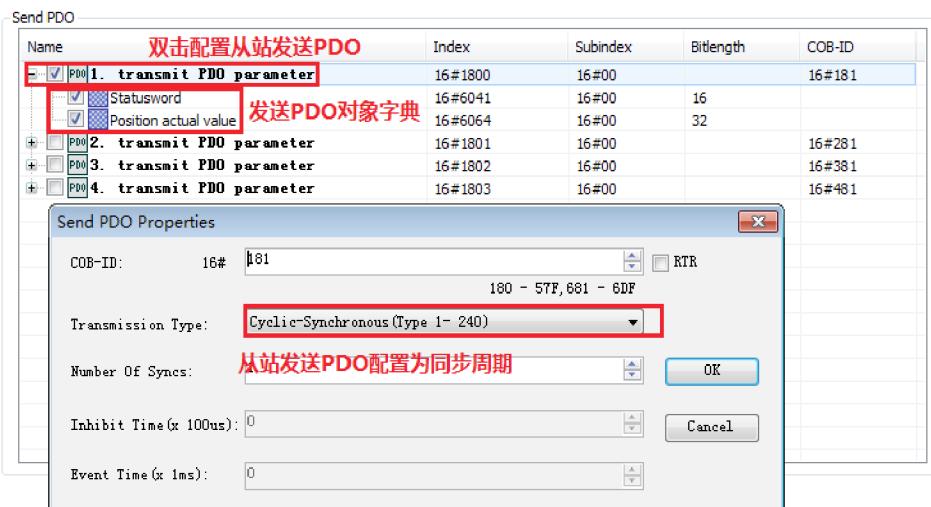


图4-47 CANopen从站发送PDO配置

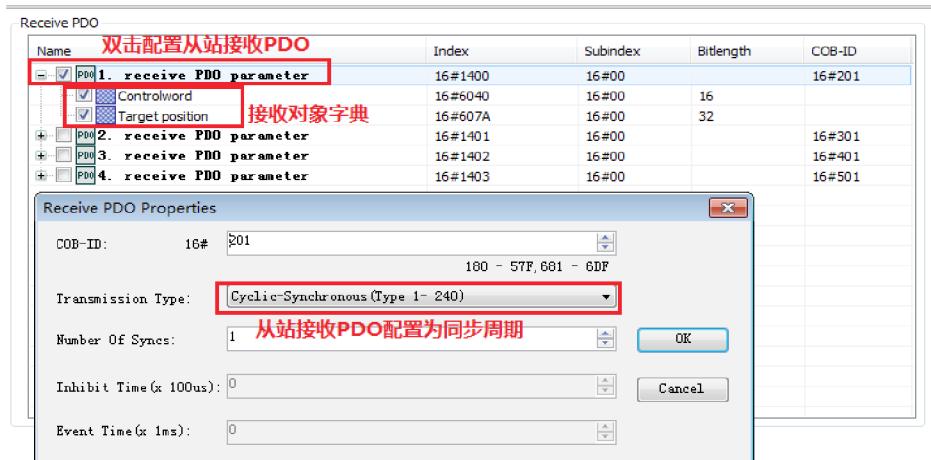


图4-48 CANopen从站接收PDO配置

## CANopen轴配置

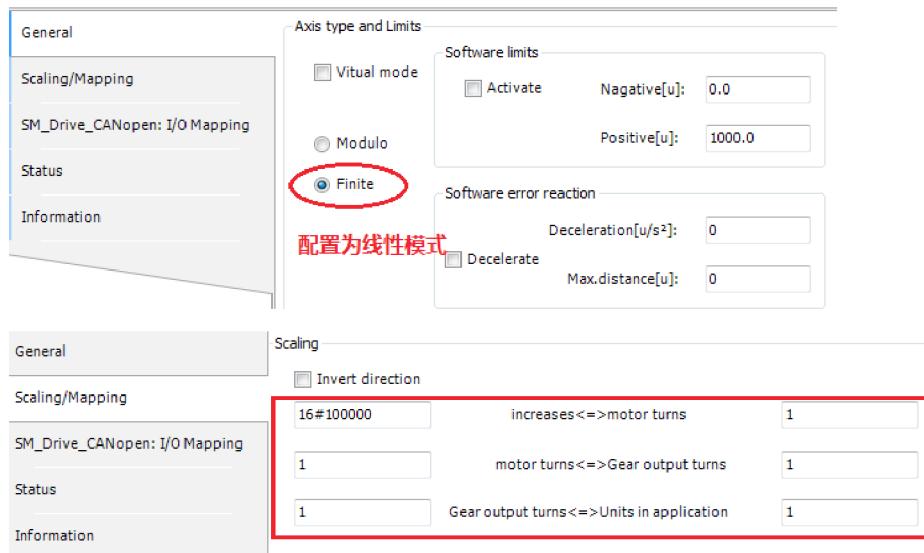


图4-49 CANopen轴配置

### 4.8.6 编程接口

请参见《中型PLC指令手册》中的6.3章节CANopen通信指令。

## 4.9 CANlink 3.0 配置编辑器

### 4.9.1 概述

CANlink协议是汇川技术股份有限公司基于CAN2.0总线协议制定的CAN实时总线应用层协议。主要用于汇川技术产品PLC，变频器，伺服控制器和远程扩展模块等产品之间进行高速实时数据交互。在使用AM600系列PLC的CANlink功能前，请仔细阅读本章节内容。

CANlink3.0采用主/从模式，在一个网络中必须具备并只有一个主站，从站数量为1~62个，所有主从站点号范围为1~63，且站号必须唯一。

1. 支持心跳监控主/从站运行状态；
2. 支持总线占有预警和实时总线占有率监控；
3. 支持掉线重连功能；
4. 支持热接入方式；
5. 主站支持发送配置（包括时间触发、事件触发、同步触发）发送数据共256条；
6. 单个从站支持发送配置（包括时间触发、事件触发、同步触发）16条，从站总计最多支持256条配置；
7. 每个站点支持接收其它8个站点发送的点对多数据；
8. 支持主/从式数据交互和从/从式数据交互；
9. 主站支持同步写最多128条。

## 4.9.2 CANlink3.0 网络组成

### 硬件端口

有关CANlink通信端口定义请参见[第250页 “硬件端口”](#)。

### 通信距离

一个CANlink3.0网络由一台主站、以及若干从站组成，最多支持的从站数目为62个（与波特率有关）。

波特率	最大通信距离	通信电缆线径	可接入站点数
1000Kbps	20m	$\geq 0.3 \text{ mm}^2$	18
500 Kbps	80m	$\geq 0.3 \text{ mm}^2$	32
250 Kbps	150m	$\geq 0.3 \text{ mm}^2$	63
125 Kbps	300m	$\geq 0.5 \text{ mm}^2$	63
100 Kbps	500m	$\geq 0.5 \text{ mm}^2$	63
50 Kbps	1000m	$\geq 0.7 \text{ mm}^2$	63

以上数据是在使用标准屏蔽双绞线前提下，可接入站点数是当前波特率下网络中允许的最大节点数（主站和从站总数）。

### 支持CANlink3.0的设备

在一个CANlink3.0网络中，必须有且只有一个主站，这个主站可能是AM400、AM600或AC800等系列PLC。在该网络中，可以存在1~62个从站，这些从站包括AM400、AM600或AC800（查看D8280，为300表示支持）、远程扩展模块(51210或52210以上版本)、214非标IS500伺服（功能码H00-02=214.xx）、IS620P（H01-00=6.0或以上）、IS700（H01-00=301.05）、MD310（F7-11=u37.18或以上）、MD380（F7-11=4.71.06或以上）等。部分产品需配置专用的CANlink通讯扩展卡才能使用CANlink功能，具体请参照各产品的用户手册。

### AM600支持CANlink3.0的特殊元件说明

#### 说明

AM600的CANLINK功能使用的软元件名称是SD和SM，与小型PLC的D和M元件类似，但它们之间没有对应关系。

SD区间分配	功能	SM区间分配	功能
0 - 7000	用户通用的字软元件区 (CANLINK配置表中能使用到的范围)	0 - 3071	用户通用的位软元件区 (CANLINK配置表中能使用到的范围)
7000-7999	用户通用的字软元件区	3072-7999	用户通用的位软元件区
8000 - 8999	系统定义的特殊寄存器元件区	8000 - 8999	系统定义的特殊位元件区：目前只有CANLINK使用
9000 - 9999	系统定义的特殊寄存器元件区： 目前只有 高速I/O使用	9000 - 9999	系统定义的特殊寄存器元件区： 目前 只有 高速I/O使用

AM600 CANLINK功能涉及的特殊软元件定义如下（具体定义请参见CANLINK3.0标准）：

特殊软元件	属性
SD8100 - SD8163	SD8100 本机节点的当前状态；SD81xx表示站点号为xx的节点的当前状态，例如SD8101代表站点号为1的节点状态。寄存器值含义： 0：未配置；1：有配置；2：在线；5：离线
SD8164 - SD8239	保留
SD8240	总线负载率（后台监控用）
SD8241	CAN3.0从站状态
SD8242	CAN3.0从站状态
SD8243	CAN3.0从站状态
SD8244	CAN3.0从站状态
SD8245	接收帧数
SD8246	接收错误计数
SD8247	发送错误计数
SD8287	厂商代码
SD8288	产品系列
SD8289	产品型号
SD8290	固件版本
SD8307	CAN3.0报错（命令异常码）
SD8308	CAN3.0报错(配置异常码)

### 4.9.3 CANlink 一般使用流程

CANlink一般使用流程如下：

1. 设计Canlink硬件网络结构。
2. 在网络组态中激活CANlink总线。AM600 CPU既可以作为CANlink主站也可以作为CANlink本地从站。激活总线后，会自动添加总线设备。
3. 如果AM600 CPU作为CANlink主站，可以在CANlink网络配置向导中，配置主站参数和添加、删除从站（远程从站，后面单独说从站指远程从站）。如果AM600 CPU作为CANlink本地从站，也可以配置其参数
4. 设置合适的发送配置参数、接收配置参数及同步配置参数。
5. 在程序中通过操作软元件控制CANlink传输数据。
6. 在网络管理页面控制主从站启停、监控主从站运行状态。

### 4.9.4 CANlink 网络配置

在配置CANlink网络之前，需要在网络组态激活CANlink总线。如果激活CANlink主站，在设备树中添加CANlink主站节点，第一个配置CANlink网络时，双击此节点显示网络配置向导；如果激活CANlink从站，在设备树中添加CANlink从站节点，双击此节点显示本地从站配置界面。

#### CANlink3.0网络配置向导

第一次配置CANLink总线或者在网络管理界面，单击“站点管理”，弹出网络配置向导。网络配置向导配置主站参数和从站参数。

## 主站配置

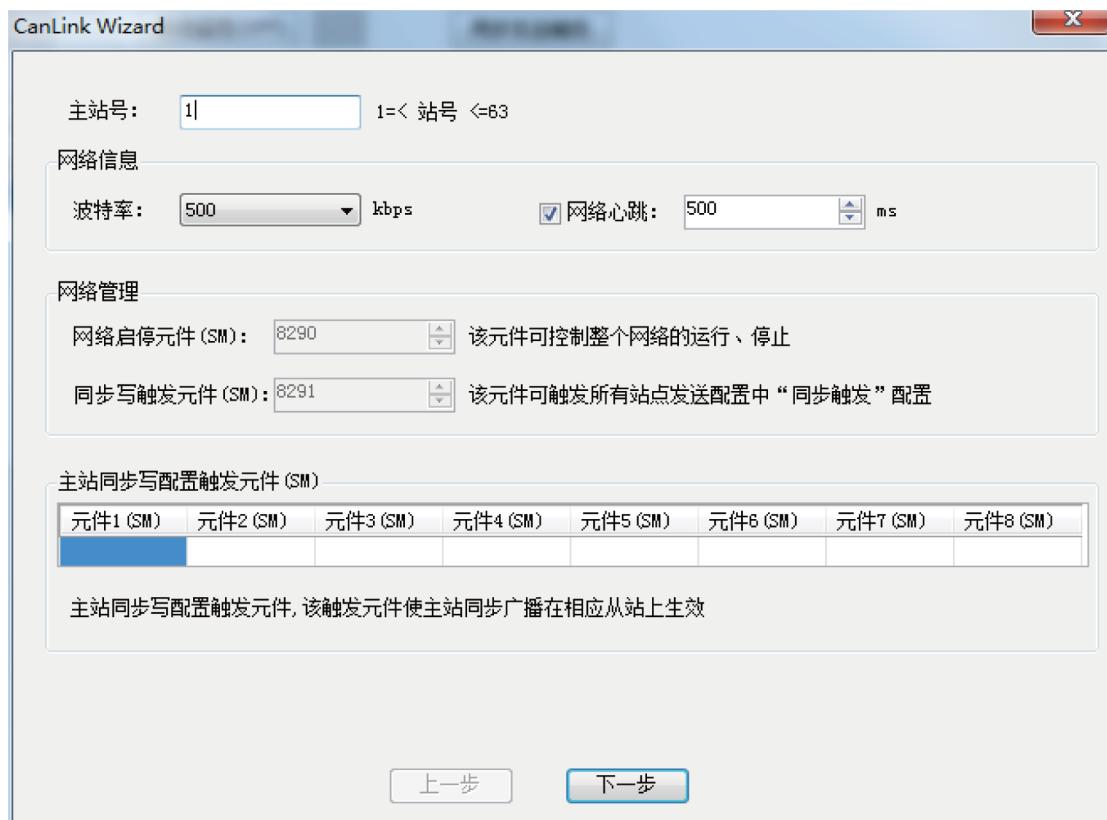


图4-50 主站参数配置界面

- 主站号：主站号作为网络中主站的标示，表明该站号的站点为主站，由主站对整个CANlink3.0网络进行管理和监控。注意，主站号必须与实际下载配置的PLC一致。

### 网络信息

- 波特率：选择网络使用的波特率，要求所有站点的波特率均与此处选择的一致，即使是没有进行配置，但线路接入网络中的站点，其波特率也需一致。
- 网络心跳：网络心跳设置范围为10~20000ms，主从站按设定值定时向网络中发送心跳，主从站通过心跳监控各自通信情况，当出现通信异常时告警并作出相应的处理。网络心跳设定值越小，监控越灵敏，但同时对网络占用率也更高，当心跳对网络占用率超过10%时将自动告警。

### 说明

如果将网络心跳前的勾去掉，则网络心跳功能将取消，将无法对网络进行监控。

### 网络管理

- 网络启停元件：控制CANlink网络启动和停止，使用软元件SM8290，当SM8290值为TRUE时表示启动CANlink，否则停止CANlink。
- 同步写触发元件：控制所有站点发送配置的同步写功能，使用软元件SM8291，具体请参见[第281页“同步触发配置”](#)。

### 主站同步写配置触发元件

主站同步写触发元件用于主站同步配置。最多可以设置8个同步触发元件。当网络中不需主站同步配置时，此处可不填写，具体参见第283页“4.9.8 主站同步写”。

## 增加和删除从站

此界面用于添加、修改和删除CANlink从站。

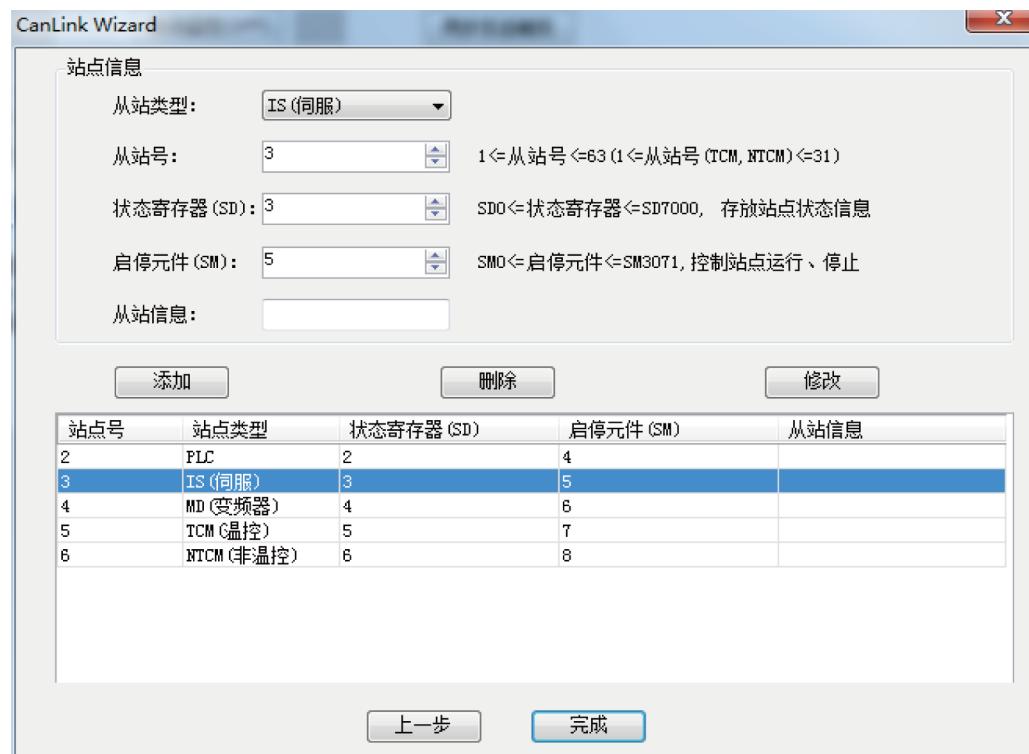


图4-51 从站添加和删除界面

### 站点信息

- 从站类型：支持的从站类型，当前支持PLC、伺服、变频器、温控模块、非温控模块从站。
- 从站号：作为网络中从站的标示，不能和主站或者其它从站号重复。
- 状态寄存器：表示相应从站运行状态的SD元件，这些元件不能和其它从站状态寄存器元件重复，这些状态主要包括从站的运行、是否有故障（不包括掉线）等。
- 启停元件：启停元件是主站控制从站启停的SM元件，网络运行时可通过改变这些SM元件来控制各从站通信的启停，这些元件不能和其它从站启停元件、同步写触发元件重复。
- 从站信息：表示从站信息，给从站增加注释信息，最大32字符。
- 添加：添加站点信息对应的从站到从站列表，添加从站时会检查状态寄存器、启停元件的唯一性。
- 删除：删除从站列表所选从站。
- 修改：修改从站列表所选从站点信息，这些信息从站点信息设置中获取。修改从站时会检查状态寄存器、启停元件的唯一性。

完成所有从站的设定后，点击“完成”即进入网络管理界面。

## 4.9.5 网络管理

网络管理界面可以管理网络的启停、同步发送触发、启停监控、启停从站，也可以进入站点配置向导，同样可以修改站点网络信息，另外可以清空所有配置。

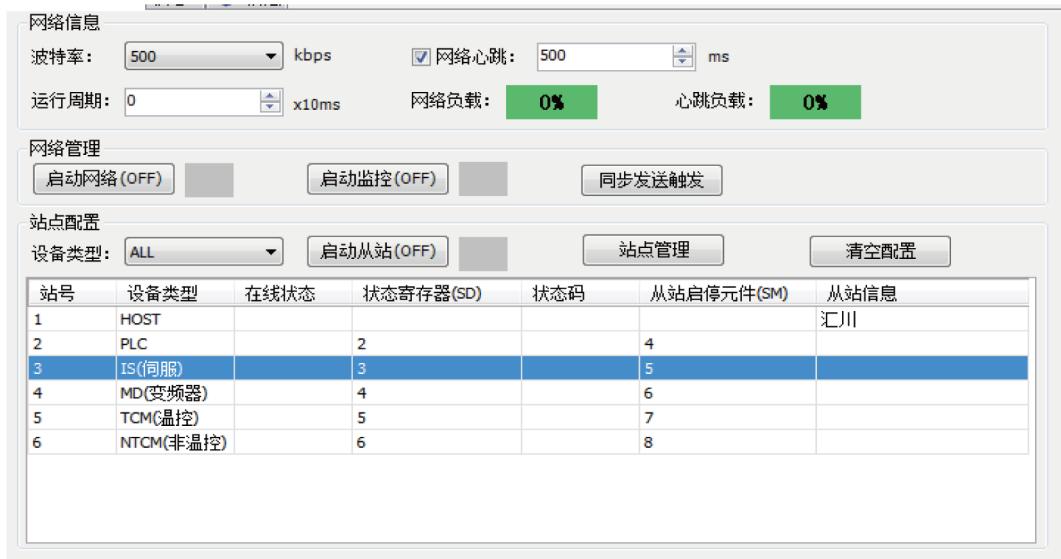


图4-52 网络管理界面

### 网络信息

- 波特率：选择网络使用的波特率，要求所有站点的波特率均与此处选择的一致，即使是没有进行配置，但线路接入网络中的站点，其波特率也需一致。
- 网络心跳：网络心跳设置范围为10~20000ms，主从站按设定值定时向网络中发送心跳，主从站通过心跳监控各自通信情况，当出现通信异常时告警并作出相应的处理。网络心跳设定值越小，监控越灵敏，但同时对网络占用率也更高，当心跳对网络占用率超过10%时将自动告警。

### 说明

如果将网络心跳前的勾去掉，则网络心跳功能将取消，将无法对网络进行监控。

- 运行周期：用于预估CANlink负载率。
- 网络负载：CANlink网络负载，包括CANlink接收、发送、心跳所有负载，监控条件下获取CANLink总线实时负载（获取SD8240寄存器值），非监控条件下预估网络负载。
- 网络负载背景颜色：

颜色	范围 (%)	负载文字
绿色	0~50	负载百分比，如10%
黄色	51~75	负载百分比，如55%
红色	76~90	负载百分比，如78%
红色	大于90	ERR

- 心跳负载：CANlink心跳负载，心跳负载通过预估计算得到（登陆后获取的也是预估值）。
- 心跳负载背景颜色：

颜色	范围 (%)	负载文字
绿色	0~10	负载百分比, 如10%
红色	大于10	ERR

### 网络管理

网络管理功能必须在登陆PLC后才能操作。

- 启动网络：启停CANlink网络，通过控制网络启停元件SM8290来启动或者停止网络。
- 启动监控：开始CANlink网络监控，定时获取CANlink主从站运行状态。在站点列表中会显示站点在线状态更新。
- 同步发送触发：触发一次同步写功能。

### 站点配置

- 设备类型：选择站点列表要显示的设备类型。
- 启动从站：把从站置于运行状态，通过更改从站启停元件的值为TRUE，来使从站运行。
- 站点管理：弹出网络配置向导对话框，配置CANLink网络。
- 清空配置：清空CANlink所有配置信息，把主站配置信息置为默认值。
- 站点列表：显示CANlink站点信息和站点状态。包含站号、设备类型、在线状态、状态寄存器、状态码、从站启停元件和从站信息。双击列表每行对应的从站，弹出站点发送配置、接收配置和主站同步写界面。
- 站号：站点唯一标示符。
- 设备类型：站点设备类型，包括PLC、伺服、变频器、温控模块、非温控模块。
- 在线状态：登陆PLC后，点击“启动监控”，显示从站状态。从站在线状态通过SD8241、SD8242、SD8243、SD8244在线状态寄存器获取从站是否在线和离线，如果从站在线、通过获取从站启停元件值，判断从站是否运行或者停止。在线状态寄存器为只读，不允许对其进行修改。在线状态寄存器每位和站号对应关系如下表。

软元件	位	从站号
SD8244	Bit15~bit0	32~47
SD8243	Bit15~bit0	48~63
SD8242	Bit14~bit0	1~15
SD8241	Bit15~bit0	16~31

- 状态寄存器：保存从站运行状态的运行状态寄存器。运行状态寄存器为只读，不允许对其进行修改。运行状态寄存器定义见下表：

位域	说明
bit0	故障标示，“1”表示节点设备故障、“0”表示无故障
bit1	运行标示，“1”表示运行、“0”表示停机
bit2	设备就绪，“1”表示就绪、“0”表示未就绪(伺服专用)
...	保留
bit15	保留

- 状态码：从站状态寄存器值，表示从站定义的状态码。
- 从站启停元件：控制从站的运行和停止。
- 从站信息：从站注释，可以定义从站说明信息。

## 4.9.6 发送配置

CANLink发送配置分时间触发配置、事件触发配置、同步触发配置三种，其中事件触发配置又有PLC事件触发和非PLC事件触发两种。

## 发送配置编辑器



图4-53 发送配置编辑器

发送配备编辑界面用于配置站点发送设置。发送列表分两部分，上半部分显示本站发送给别的站点配置，后半部分显示，别的站点发送给本站点的配置。

对于主站能配置的发送条数最大为256条，对于从站最大发送条数为16条。

发送列表包含触发方式、触发条件、发送站、发送寄存器、接收站、接收寄存器和寄存器个数。

项目	描述
触发方式	站点发送数据触发类型，包括时间触发、事件触发和同步触发
触发条件	站点发送数据触发条件，满足条件时开始发送配置数据。当触发方式为时间触发时，范围为0~30000，站点按照触发条件值定时发送配置数据；当触发方式为事件触发时，如果站点为Host或者PLC类型，触发条件为SM软元件，范围为0~3071，当触发条件为TRUE时站点发送配置数据，如果为伺服、变频器、温控模块、非温控模块时，范围为0~30000，表示定时发送配置数据；当触发方式为同步触发时，当软元件SM8291值为TRUE，执行发送动作
发送站	发送配置对应的站点，不能更改
发送寄存器	发送配置站对应的寄存器，如果发送站为PLC类型，范围（此范围表示寄存器值加寄存器个数，本小节的范围都是此意义）为0~7000；如果发送站为温控模块，范围为0~499，700~722；如果发送站为非温控模块，范围为0~63，700~722；如果发送站为伺服或者变频器，范围为0~65535
接收站	接收发送配置的站点，只能为已经添加的站点
接收寄存器	接收发送配置的站点中的寄存器。如果是点对多通信（发送站和接收站相同），范围为0~65535；如果接收站为PLC类型，范围为0~7000；如果接收站为温控模块，范围为0~499，721~722；如果接收站为非温控模块，范围为0~63，721~722；如果接收寄存器个数站为伺服或者变频器，范围为0~65535
寄存器个数	发送或者接收寄存器个数，范围1~4，发送寄存器或者接收寄存器总长度不能超过范围限制值

## 时间触发配置

时间触发配置是一种常用配置，发送站按配置的时间循环将源地址数据写入目标站的目标地址中（即相当于目标站循环读取发送站的源地址数据，并存放于目标地址中）。

若发送站和接收站相同，则该条是点对多配置，网络中所有配置接收该站点对多数据的站点均可接收该类数据（请参见第282页“4.9.7 接收配置”）。



图4-54 时间触发配置界面

时间触发方式中，时间设定范围为1~30000ms，循环时间设定数值越小，数据更新速度越快，网络负载也会越高。

如上图中，第3行配置表示1号站以100ms的循环周期将SD102、SD103两个寄存器的值写入到3号站伺服的H0200、H0201寄存器中。第4行配置中，发送站和接收站均为1号站，则表示该配置为点对多配置，1号站以100ms的周期将SD110-SD113四个寄存器的值以点对多的形式发送到网络中，所有配置接收1号站点对多数据的站点都可以接受到该帧。

## 事件触发配置

事件触发是一种实时性的触发配置，当满足条件时即时发送，若条件不满足无论距离上次发送多久也不会触发。根据产品的特点，PLC的事件触发与其他产品的事件触发略有不同。

### PLC事件触发配置

当作为触发条件的SM被置ON时，触发相应的事件。触发事件的范围为SM0~3071，超出后会自动提示。



图4-55 PLC事件触发配置

如上图中，当SM100=1时，1号站将SD100的值发送到2号站PLC的SD100中，发送后SM100会自动复位。

### 非PLC设备事件触发配置

除了PLC外，变频器、伺服、扩展模块等的事件触发均是采取寄存器值改变，且距离上次发送满足最小间隔时间时触发的方式。



图4-56 非PLC事件触发配置

如上图，4号站变频器的H3000发生变化时，如果距离上次该配置发送的时间达到100ms，则立即发送，如距离上次发送不足100ms则等待直到时间足够才发送，触发条件中的间隔时间越小，实时性越强，同时也对网络影响越大。

最小间隔时间的设置范围为1~30000ms，超出范围将自动提示。

## 同步触发配置

同步触发配置是主站检测到同步发送触发元件SM8291被置位时，向网络中广播命令，要求网络中所有站点的同步触发配置（触发条件为SM8291的配置）全部依次触发的一种配置。主站发送命令帧后将自动复位SM8291。如下图，1号主站、2号PLC从站与4号MD380从站均有同步触发配置，若主站SM8291被置位，这三个站的同步触发都将发送。



图4-57 同步触发配置

SM8291的使用方法与PLC事件触发的SM元件一致，同时也可以在网络运行时在CANlink3.0主界面点击“同步发送触发”执行相应操作。

只有主站的SM8291才有意义，对从站PLC的SM8291执行操作不会产生任何作用。

同步触发实际上就是网络中所有站点共用主站SM8291为事件的事件触发。

## 时间、事件、同步触发区别

项目	时间触发	事件触发	同步触发
发送方式	定时循环发送	事件发生或数据改变时发送，发送完成后自动清除事件	主站使能发送，完成后自动清除事件
实时性	与设定时间有关，时间越短实时性越高	事件发生即触发，实时性好	主站M8291置位即触发，实时性略好于事件触发
执行次数	定时发送	事件发生时触发一次	事件发生时触发一次
PLC中编程	定时发送，无需程序设计	需在程序中设计相应的时序逻辑	需在程序中设计相应的时序逻辑
网络占用率	高	低	低
适用场合	无特殊的时序要求，可以持续写入的数据	有时序要求，只需一次写入或持续写入会造成故障的数据	主站在特定场合时需多个从站集体返回的数据

## 4.9.7 接收配置

接收配置是配置各站接收网络中点对多数据的配置。



图4-58 接收配置对话框

如上图，2号从站接收配置中有1和3号站，表示2号从站可以接收到网络中所有1号站和3号站发出的点对多数据帧，而其他站发出的点对多数据则不会接收。如接收到的点对多数据的目的寄存器地址符合2号站的定义，则接收的数据生效，如不符合则即使接收也会将其丢弃。

每个站点最多允许接收其他8个站点发出的所有点对多数据。每个站点发出的点对多数据没有限制可以接收的站点数目。

通过点对多可以实现一个站点对多个站点同一功能码地址的修改，并可以实现同步的功能。

#### 4.9.8 主站同步写

主站同步配置是主站特有的一个配置，主要用于主站对一台或多台从站的多个寄存器或功能码同时进行写入操作。譬如，主站可以通过同步写多台伺服的功能码H31-00来控制它们同时启动或停止。

##### 触发元件

如在“配置向导”中的“主站同步写触发元件(M)”已填写相应的触发元件，则可以在主界面双击主站打开主站配置，在“同步配置”中“触发条件”下拉切换不同的配置条件进行配置。如在前面配置向导中没有填写，此处可打开但无法填写或下拉切换条件，可以点击主界面的“站点管理”重新进入向导增加/删除触发元件。



图4-59 主站同步写对话框

当主站的某个同步写触发元件置ON时，该触发元件下的所有配置将依次全部发送出去，从站接到这些数据后，会存放于缓冲区。主站检测到触发条件下的同步写已全部成功发送时将自动广播发送一个生效指令，使所有接收到数据的从站将数据从缓冲区中取出并生效。

主站同步写最多允许8个不同的触发条件，每个触发条件最多可以配置16条同步写，同时单个从站最多接收8条同步写配置（超出后后台会自动提示）。

### 对伺服32位寄存器的操作

CANlink3.0中是对16位寄存器和功能码进行操作，如需对32位寄存器或功能码进行操作，有如下方法：

主站同步写32位寄存器或功能码的高低16位地址。

例如：假设在伺服中的功能码H11-12（第1段移动位移），如下操作可将主站的SD1000作为低16位，SD1001作为高16位写入到3号伺服的H11-12中。

编号	发送站	发送寄存器	接收站	接收寄存器
1	1 HOST	SD1000 十进制	3 IS (伺服)	110C 十六进制
2	1 HOST	SD1001 十进制	3 IS (伺服)	110D 十六进制
3	1 HOST	十进制		
4	1 HOST	十进制		
5	1 HOST	十进制		
6	1 HOST	十进制		

图4-60 两个16位寄存器合并32值

建议用M元件的上下沿导通触发元件的SET语句。操作32位地址的功能码时，不允许只对低/高16位地址进行操作，也不允许把32位功能码的高低16位地址拆分到不同的触发条件下操作。

另外，也可以用普通发送配置将连续的两个SD元件的值一次性写入到伺服32位功能码的低地址位中，如下图：

编号	触发方式	触发条件	发送站	发送寄存器	接收站	接收寄存器	寄存器个数
1	时间 (ms)	100	1 HOST	1000	十进制	3 IS (伺服)	110C 十六进制 2
2			1 HOST		十进制		
3			1 HOST		十进制		
4			1 HOST		十进制		

图4-61 一个32位值写入到两个16位寄存器

#### 4.9.9 本地从站配置

本地从站指PLC作为CANlink从站，远程从站指主站下挂接的从站，如伺服、变频器、温控模块，甚至PLC。一台PLC可以作为主站或者作为本地从站，只能选择其一。

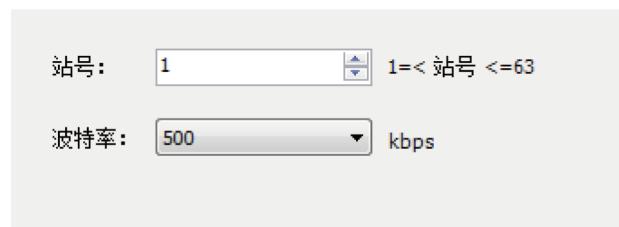


图4-62 本地从站配置对话框

站号：本地从站站点标示符，不能和CANlink网络中其它站点重复，范围1~63。

波特率：本地从站数据通信波特率，必须和主站波特率设置一致。

#### 4.9.10 设备接入CANlink3.0 网络

##### AM600系列PLC接入CANlink3.0网络

AM600的站号和波特率只支持软件设置，设置完成生效需要重启机器或者重新下载运行程序。

在同一个网络中，站点号需唯一，如出现重复站号，后接入的站将会自动关闭通信功能。同时不允许网络中存在0号站。

**网络信息**

波特率 :	500 Kps	<input checked="" type="checkbox"/> 网络心跳 : 500 ms
20		
50		
100		
125		
250		
500		
800		
1000		

该元件触发可控制整个网络的运行、停止

该元件可触发所有站点发送配置中“同步触发”配置

主站号 :  1 <= 主站号 <= 63

**主站同步写触发元件 (M)**

元件1 (M)	元件2 (M)	元件3 (M)	元件4 (M)	元件5 (M)	元件6 (M)	元件7 (M)	元件8 (M)
<input checked="" type="checkbox"/>							

主站同步写触发元件, 该触发元件使主站同步广播在相应从站上起效

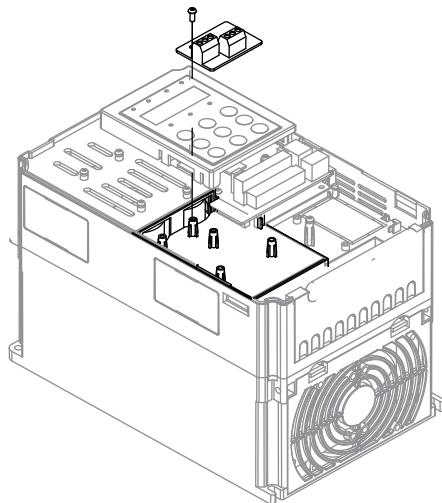
[上一步](#) [下一步](#)

## MD380/500变频器接入CANlink3.0网络

### 变频器MD38CAN1扩展卡安装

将MD38CAN1卡嵌入汇川技术的变频器中，MD38CAN1扩展卡（CANlink）不允许带电拆装，安装前请关断变频器供电电源，10分钟后等变频器充电指示灯彻底熄灭后才能进行安装。请参考下图的安装示意进行安装。

将MD38CAN1卡插入变频器后，固定相应的螺钉。



### 变频器MD380/500相关的设定

如需设定变频器的启停由CANlink网络控制，则需设定变频器的命令源为通信命令通道，即，将F0-02设为2。如不需要，按实际需要设定即可。

### 站号设置

设置功能码Fd-02可设定站号，设定范围1~63，其它值CANlink3.0将不能访问，修改后即时生效。网络中站号需唯一，如有重复，后接入的重复站点将自动关闭通信功能。

### 波特率設定

设置功能码Fd-00可设定波特率，其中千位对应CANlink的波特率，对应关系如下：

功能码地址	名称	设定范围	默认值
HFd-00	波特率	个位：Modbus 十位：Profibus-DP 百位：保留 千位：CANlink 0：20Kbps 1：50Kbps 2：100Kbps 3：125Kbps 4：250Kbps 5：500Kbps 6：1000Kbps	5005

CANlink3.0网络中所有站点的波特率必须一致，否则不能正常通信。

---

### 说明

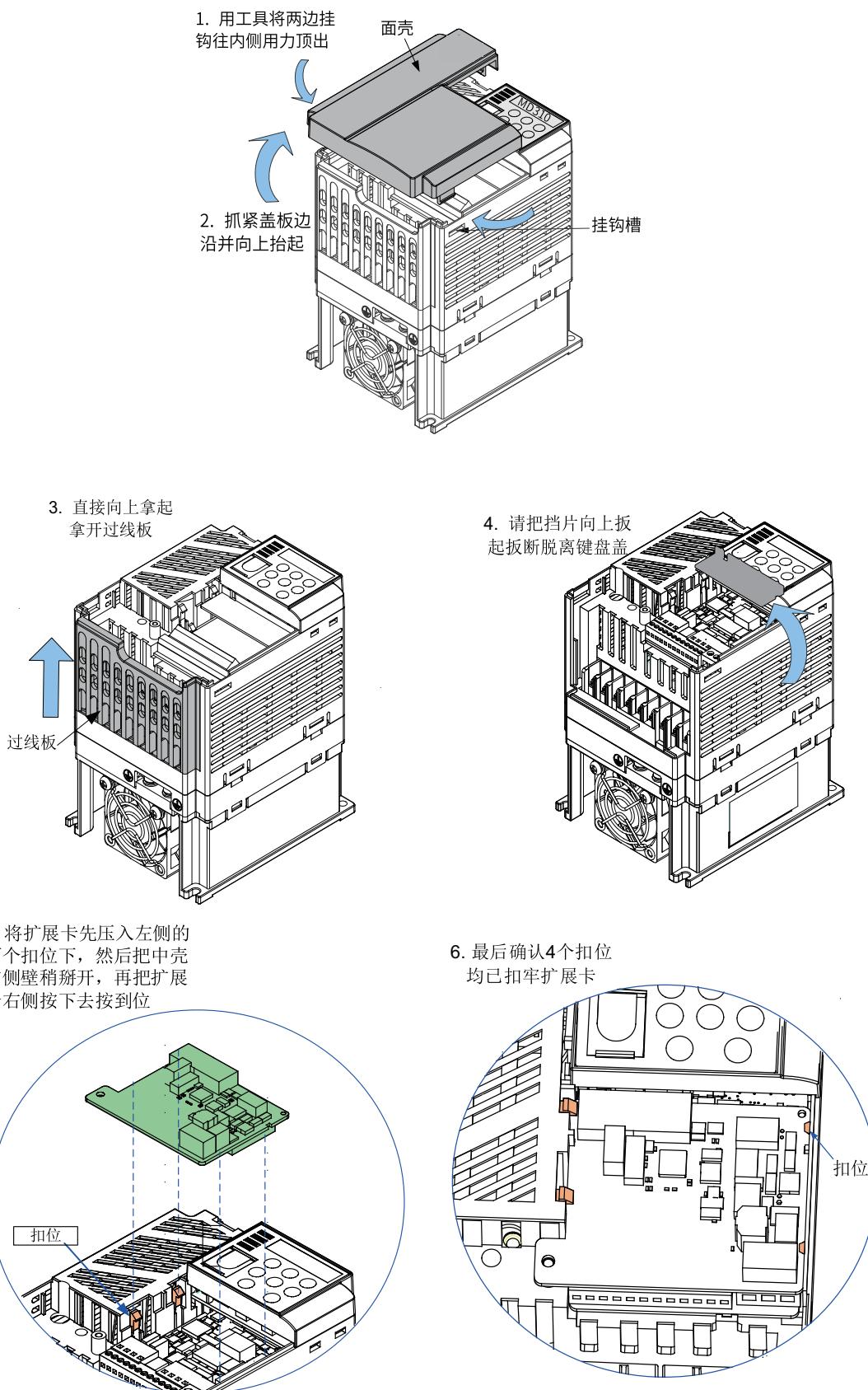
MD380/500不支持800Kbps波特率。

---

## MD310变频器接入CANlink3.0网络

### MD310-CANL卡安装

将MD310-CANL卡嵌入汇川技术的变频器中，安装前请关断变频器供电电源，10分钟后等变频器充电指示灯彻底熄灭后才能进行安装。请参考下图的安装示意进行安装。



### MD310相关设定

MD310在使用CANlink3.0时，站号及波特率均由CAN扩展卡上的拨码开关进行设定，MD310-CANL的拨码开关S1、S2组成10位拨码开关，用于设置CAN总线通讯波特率与通讯设备地址。拨码开关编号如下图所示，其

中Bd1、2、3用于设置波特率，Adr1~7用于设置CANlink地址。拨码打到“ON”表示“1”，打到下面表示“0”。波特率及站号的修改会立即生效。

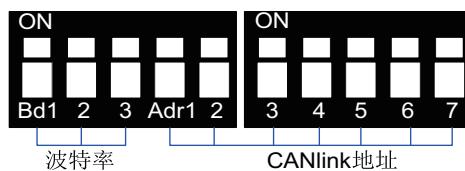


图4-63 MD310-CANL拨码开关

### 波特率设置

拨码与波特率的对应关系如下表所示，可设置8种波特率。

MD310-CANL波特率

拨码号Bd			波特率
1	2	3	
0	0	0	20Kbps
0	0	1	50Kbps
0	1	0	100Kbps
0	1	1	125Kbps
1	0	0	250Kbps
1	0	1	500Kbps
1	1	0	800Kbps
1	1	1	1Mbps

### CANlink设备地址

MD310-CANL提供7位拨码开关用于CANlink通讯地址设置，拨码“Adr1”表示最高位，拨码“Adr7”表示最低位。拨码Adr1~7对应一个地址站号的b6~b0位。拨码开关有效地址设置范围是1~63，如下表所示，0地址以及64~127为保留地址，不允许使用，设置为保留地址时MD310-CANL卡将不工作。

MD310-CANL拨码地址

拨码号Adr							地址
1	2	3	4	5	6	7	
0	0	0	0	0	0	0	保留
0	0	0	0	0	0	1	1
0	0	0	0	0	1	0	2
0	0	0	0	0	1	1	3
.....							...
0	1	1	1	1	1	1	63
1	x	x	x	x	x	x	保留

关于MD310-CANL扩展卡的其它使用注意事项，请参见《MD310-CANL通讯扩展卡说明书》。

## 伺服接入CANlink3.0网络

### 站号设置

修改功能码H0C-00的值可修改伺服的站号，设定范围为1~63，其它值CANlink3.0将不能访问，修改后即时生效。网络中的站号需唯一，如有重复，后接入的重复站点将自动关闭通信。

### 波特率设定

修改功能码H0C-08的值可修改伺服的波特率，对应关系为：

功能码地址	设定范围	默认值
H0C-08	0: 20Kbps	5
	1: 50Kbps	
	2: 100Kbps	
	3: 125Kbps	
	4: 250Kbps	
	5: 500Kbps	
	6: 800Kbps <sup>[1]</sup>	
	7: 1000Kbps	

IS620P不支持800Kbps波特率，选择6时是1000Kbps。

设定后即时生效，网络中所有站点的波特率必须一致，否则不能正常通信。

## 4.10 CAN自由协议

### 4.10.1 概述



#### 注意

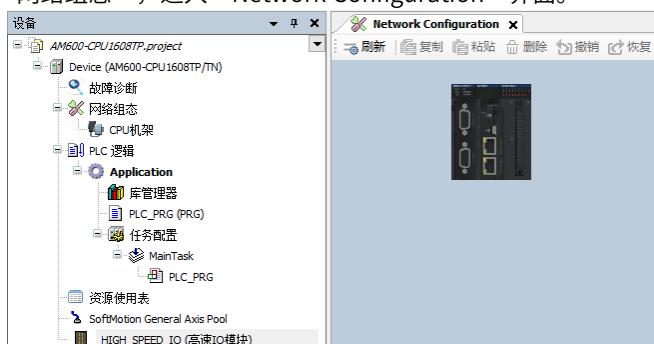
仅AM400/AM600/AM300/AM500支持CAN自由协议功能

AM400/AM600/AM300/AM500采用CAN2.0B协议，支持标准帧和扩展帧格式，适用于应用层协议自定义的场景，波特率从10kbit/s到1000kbit/s可设，应用灵活。

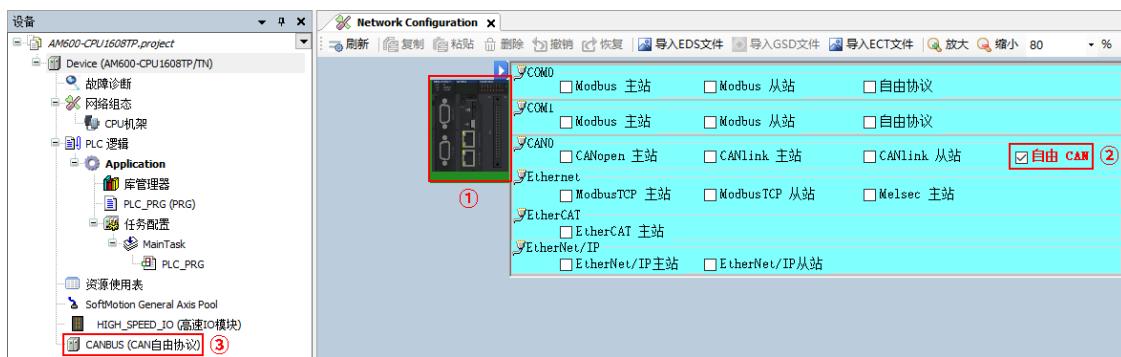
### 4.10.2 网络组态

#### AM400/AM600网络组态

1. 在左侧设备树中双击“网络组态”，进入“Network Configuration”界面。

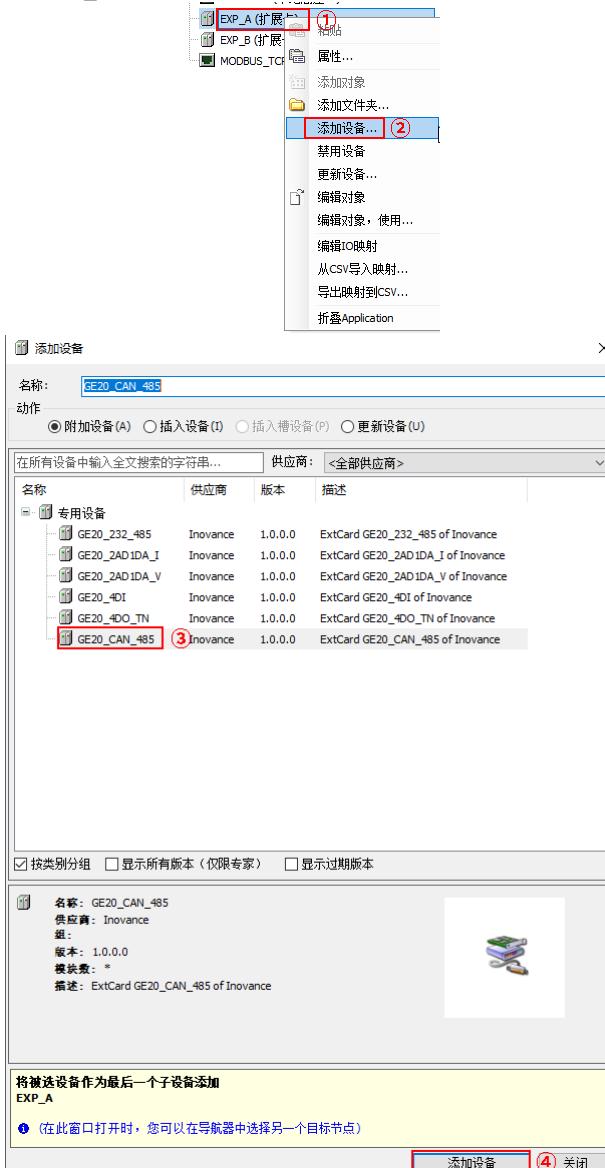


2. 单击PLC设备图片，勾选“自由 CAN”复选框，在左侧设备树中自动生成“CANBUS(CAN自由协议)”，完成“CAN自由协议”网络组态。



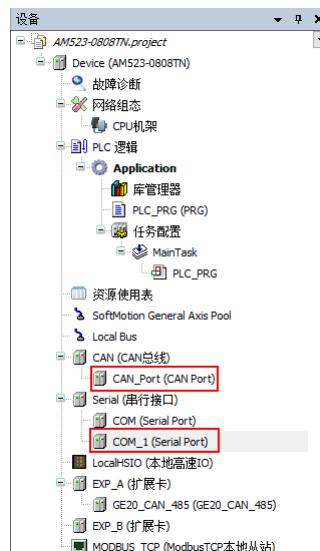
## AM300/AM500网络组态

1. 在左侧设备树中右键单击“EXP\_A（扩展卡）”，选择“添加设备”，打开“添加设备”对话框。

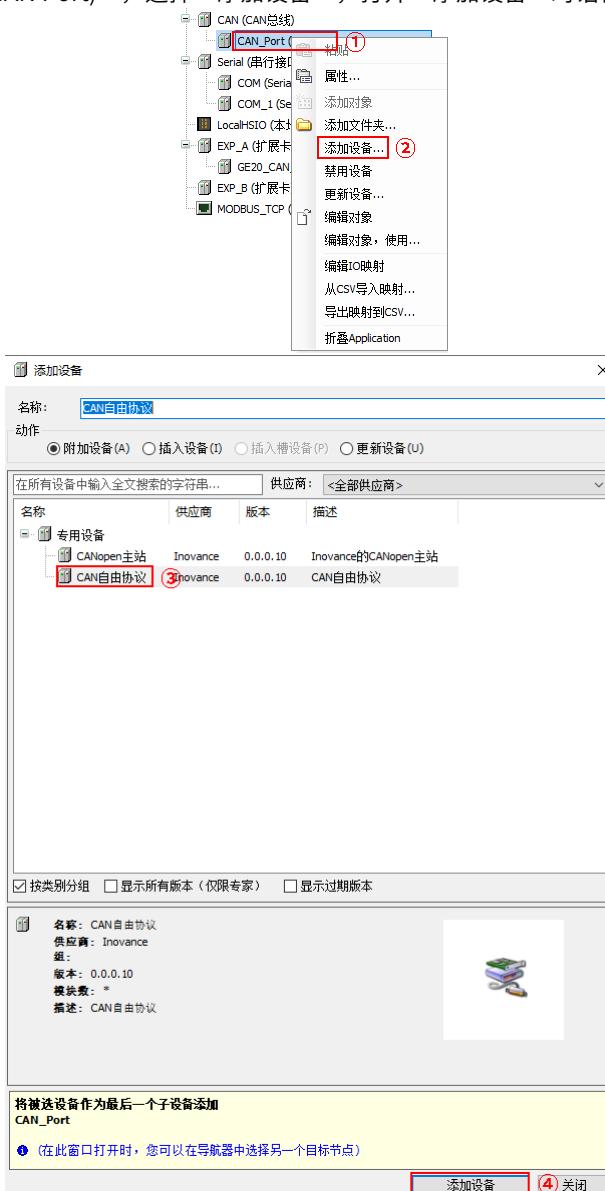


2. 选择“GE20\_CAN\_485”，单击“添加设备”。

添加扩展卡后，自动生成对应CAN口“CAN\_Port(CAN Port)”和串口“COM\_1(Serial Port)”。



3. 右键单击“CAN\_Port(CAN Port)”，选择“添加设备”，打开“添加设备”对话框。

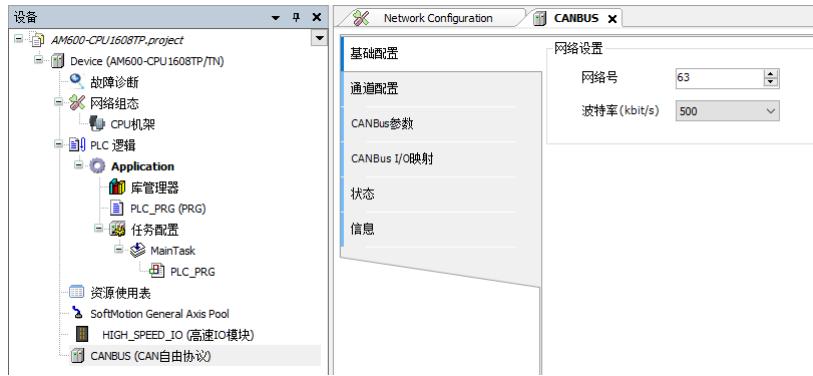


4. 选择“CAN自由协议”，单击“添加设备”，完成“CAN自由协议”网络组态。

### 4.10.3 配置CAN自由协议

AM400/AM600和AM300/AM500系列PLC配置方式略有差异，在具体步骤中进行说明，本节界面截图以AM600系列PLC为例进行介绍。

- 在左侧设备树中双击“CANBUS(CAN自由协议)”（AM300/AM500界面为“CAN自由协议(CAN自由协议)”），打开CAN自由协议配置界面。



- 配置网络。

设置网络号和波特率。

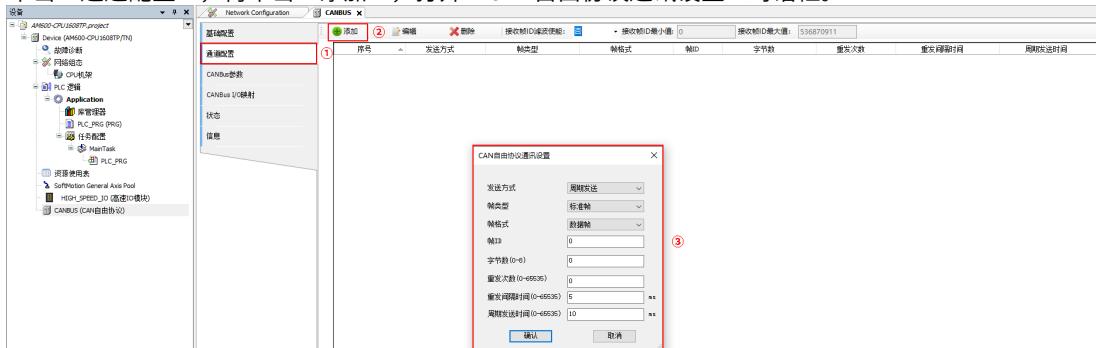


#### 注 意

- 使用功能块时，网络号作为唯一ID，在功能块中用于识别特定CANBus设备。
- AM300/AM500系列PLC请跳过本步骤。

- 配置通道。

- 单击“通道配置”，再单击“添加”，打开“CAN自由协议通讯设置”对话框。



- 配置CAN自由协议通讯参数，参数说明如下表所示。

参数名称	如何理解	如何设置
发送方式	数据发送方式。 <b>说明：</b> 触发方式模式使用CANBus_DevTxMsg功能块实现。	设置为“周期发送” 缺省值：周期发送
帧类型	帧类型包括标准帧和扩展帧。	根据实际情况设置。 缺省值：标准帧
帧格式	帧格式包括数据帧和远程帧。 <b>注意：</b> 当设置为远程帧时，无论“字节数”设置为多少，最终都按照0字节数数据发送。	根据实际情况设置。 缺省值：数据帧

参数名称	如何理解	如何设置
帧ID	十进制设置格式，内含范围检测功能，标准帧11bit有效，扩展帧29bit有效。	根据实际情况设置。 缺省值：0
字节数（0-8）	字节数量。	根据实际情况设置。 取值范围：0-8 缺省值：0
重发次数（0-65535）	数据发送失败时重新发送的次数，0表示不重新发送。	根据实际情况设置。 取值范围：0-65535 缺省值：0
重发间隔时间（0-65535）	数据发送失败时重新发送的间隔时间。	根据实际情况设置。 取值范围：0-65535 缺省值：5
周期发送时间（0-65535）	周期发送的间隔时间。 <b>注意：</b> 周期发送时间过小会导致CPU使用率过高。	根据实际情况设置。 取值范围：0-65535 缺省值：10

## 说明

- 由于CANBus属于非实时任务，受其他高优先级实时任务（如MainTask）的影响，CAN报文发送时间存在误差。当高优先级任务（例如MainTask）执行时间小于1ms时，帧间隔误差不超过±1ms。
- 部分应用中，CAN自由协议帧间隔误差会超过±1ms，实际上是系统调度过程中高优先级实时任务抢占CPU资源导致，可通过以下方式降低帧间隔时间误差。
  - 增加高优先级任务周期（如MainTask），降低CPU负载率。
  - 优化应用程序，降低每个高优先级任务（如MainTask）的执行时间。

c. (可选) 配置接收帧ID滤波使能，界面如下图所示，参数说明如下表所示。



参数名称	如何理解	如何设置
接收帧ID滤波使能	使能或禁止接收帧ID过滤机制。	根据实际情况设置。 缺省值：否
接收帧ID最小值	当“接收帧ID滤波使能”设置为“是”时有效。	根据实际情况设置。 缺省值：0
接收帧ID最大值	当“接收帧ID滤波使能”设置为“是”时有效。	根据实际情况设置。 缺省值：536870911

d. (可选) 程序下载并登录PLC设备后，分别单击“CANBus参数”、“CANBus I/O映射”或“状态”页签，分别查看CANBus参数、CANBus I/O映射和状态信息。

## 4.10.4 CANBus库

### 4.10.4.1 CANBus库枚举类型

**CANBus\_BaudrateType\_Enum: CANBus波特率枚举类型**

ENUM CANBus_BaudrateType_Enum						
名称	类型	继承自	地址	初始化	注释	
CANBUS_BAUDRATE_10KBPS	UINT			10		
CANBUS_BAUDRATE_20KBPS	UINT			20		
CANBUS_BAUDRATE_50KBPS	UINT			50		
CANBUS_BAUDRATE_100KBPS	UINT			100		
CANBUS_BAUDRATE_125KBPS	UINT			125		
CANBUS_BAUDRATE_250KBPS	UINT			250		
CANBUS_BAUDRATE_500KBPS	UINT			500		
CANBUS_BAUDRATE_800KBPS	UINT			800		
CANBUS_BAUDRATE_1000KBPS	UINT			1000		

**CANBus\_CtrlCmdType\_Enum: CANBus控制命令枚举类型**

ENUM CANBus_CtrlCmdType_Enum						
名称	类型	继承自	地址	初始化	注释	
CANBUS_CTRLCMD_INVALID	UINT			0	无效的命令	
CANBUS_CTRLCMD_RESTART	UINT			1	重启CANBus	
CANBUS_CTRLCMD_START	UINT			2	启动CANBus	
CANBUS_CTRLCMD_STOP	UINT			3	停止CANBus	
CANBUS_CTRLCMD_CLEAR	UINT			4	清零CANBus参数	

**CANBus\_ErrorType\_Enum : CANBus错误枚举类型**

ENUM CANBus_ErrorType_Enum						
名称	类型	继承自	地址	初始化	注释	
CANBUS_ERR_OK_NOERR	UINT			0	正常或者无错误	
CANBUS_ERR_CFG_ERR	UINT			1	配置错误	
CANBUS_ERR_DRV_ILLEGAL_ERR	UINT			2	驱动类非法错误	
CANBUS_ERR_DRV_NO_ACK_ERR	UINT			3	驱动类没有应答错误	
CANBUS_ERR_DRV_BUSOFF_ERR	UINT			4	驱动类总线关闭错误	
CANBUS_ERR_BUS_LOAD_OV_ERR	UINT			5	总线负载高错误	
CANBUS_ERR_TX_BUSY_ERR	UINT			16	发送忙错误	
CANBUS_ERR_TX_TIMEOUT_ERR	UINT			17	发送超时错误	
CANBUS_ERR_TX_BUFF_FULL_ERR	UINT			18	发送缓存区满错误	
CANBUS_ERR_RX_BUFF_EMPTY_ERR	UINT			32	接收缓存区空错误	
CANBUS_ERR_RX_BUFF_FULL_ERR	UINT			33	接收缓存区满错误	
CANBUS_ERR_OTHER_UNDEF_ERR	UINT			65535	其他未定义错误	

**CANBus\_FBFErrorType\_Enum : CANBus功能块错误枚举类型**

ENUM CANBus_FBFErrorType_Enum						
名称	类型	继承自	地址	初始化	注释	
CANBUS_FBERR_NO_ERROR	UINT			0	功能块运行正常或者无错误	
CANBUS_FBERR_EXE_FAILED	UINT			1	功能块运行失败	
CANBUS_FBERR_INVALID_DEVICEID	UINT			2	无效的设备ID, 该ID未使能	
CANBUS_FBERR_DEVICE_FAULT	UINT			3	设备存在故障	
CANBUS_FBERR_WRONG_PARAMETER	UINT			4	功能块输入参数错误	
CANBUS_FBERR_BUSY_ERROR	UINT			5	功能块忙	
CANBUS_FBERR_ABORT_ERROR	UINT			6	功能块中止错误	
CANBUS_FBERR_TIME_OUT_ERROR	UINT			7	功能块超时错误	
CANBUS_FBERR_POINTER_NULL	UINT			8	功能块指针空错误	
CANBUS_FBERR_BUFF_FULL	UINT			9	功能块缓存满错误	
CANBUS_FBERR_UNKNOWN_ERROR	UINT			255	16#000A~16#00FE, 保留 功能块未知错误	

**CANBus\_FBStateType\_Enum : CANBus功能块状态枚举类型**

ENUM CANBus_FBStateType_Enum						
名称	类型	继承自	地址	初始化	注释	
CANBUS_FBSTATE_DORMANT	UINT				功能块等待启动	
CANBUS_FBSTATE_EXECUTING	UINT				功能块周期运行	
CANBUS_FBSTATE_ABORTING	UINT				功能块中止运行	
CANBUS_FBSTATE_DONE	UINT				功能块运行完成	
CANBUS_FBSTATE_ERROR	UINT				功能块运行错误	
CANBUS_FBSTATE_RESETTING	UINT				功能块复位中	

## CANBus\_StateType\_Enum : CANBus状态枚举类型

ENUM CANBus_StateType_Enum						
名称	类型	继承自	地址	初始化	注释	
CANBUS_STATE_UNKNOWN	UINT			0	未知状态	
CANBUS_STATE_INITIALIZE	UINT			1	初始化状态	
CANBUS_STATE_RUNNING	UINT			2	运行状态	
CANBUS_STATE_STOPPING	UINT			3	停止状态	
CANBUS_STATE_FAULTRUN	UINT			4	故障运行状态	
CANBUS_STATE_FAULTSTOP	UINT			5	故障停止状态	

### 4.10.4.2 CANBus库结构体类型

#### CANBus\_IoDrvVer\_t: CANBus使用的驱动组件版本

STRUCT CANBus_CmpDrvVer_t						
名称	类型	继承自	地址	初始化	注释	
dwCmpDrvID	DWORD				组件ID	
dwCmpDrvVer	DWORD				组件版本号	
dwCmpDrvDate	DWORD				组件创建的日期	

CANBus\_IoDrvVer\_t类型中的参数统一使用十六进制格式。

#### CANBus\_DiagInfo\_t: CANBus诊断信息结构体

STRUCT CANBus_DiagInfo_t						
名称	类型	继承自	地址	初始化	注释	
xEnableFlag	BOOL				使能标志: FALSE-禁止; TRUE-使能	
xErrorFlag	BOOL				错误标志: FALSE-正常; TRUE-错误	
uiNetId	UINT				CANBus设备的网络ID, 范围[1,63]	
eStateType	CANBus_StateType_Enum				CANBus设备运行状态	
eErrorType	CANBus_ErrorType_Enum				CANBus设备错误类型	
uiBaudrate	UINT				CAN总线波特率, 单位kbps	
uiBusload	UINT				CAN总线负载率, 千分比	
uiTxErrCnt	UINT				发送错误计数器TEC	
uiRxErrCnt	UINT				接收错误计数器REC	
udiTxMsgNumInQueue	UDINT				队列中的发送帧数	
udiTxMsgNumFinish	UDINT				已完成的发送帧数	
udiTxBitCntPerSecond	UDINT				每秒发送位计数器	
udiRxMsgNumInQueue	UDINT				队列中的接收帧数	
udiRxMsgNumFinish	UDINT				已完成的接收帧数	
udiRxBitCntPerSecond	UDINT				每秒接收位计数器	

**CANBus\_FrameMsg\_t: CANBus帧信息结构体**

STRUCT CANBus_FrameMsg_t					
名称	类型	继承自	地址	初始化	注释
xExtFlag	BOOL				扩展帧标志: FALSE标准帧; TRUE扩展帧
xRtrFlag	BOOL				远程帧标志: FALSE数据帧; TRUE远程帧
dwCanId	DWORD				CANID: 标准帧低11bit有效; 扩展帧低29bit有效
usiDataLen	USINT				数据长度: 数据帧0-8; 远程帧0
abyDataStr	ARRAY [0..7] OF BYTE				数据区: CAN帧数据区最大字节数为8

**CANBus\_RxFilter\_t: CANBus接收帧过滤结构体**

STRUCT CANBus_RxFilter_t					
名称	类型	继承自	地址	初始化	注释
xExtFlag	BOOL				扩展帧标志: FALSE标准帧; TRUE扩展帧
xRtrFlag	BOOL				远程帧标志: FALSE数据帧; TRUE远程帧
dwCanId	DWORD				CANID: 标准帧低11bit有效; 扩展帧低29bit有效
xFilterFlag	BOOL				过滤机制使能标志: FALSE禁止过滤机制, 按照单一ID接收; TRUE使能过滤机制
xFilterExt	BOOL				扩展帧过滤: FALSE禁止使能xExtFlag过滤; TRUE使能xExtFlag过滤
xFilterRtr	BOOL				远程帧过滤: FALSE禁止xRtrFlag过滤; TRUE使能xRtrFlag过滤
dwFilterCanId	DWORD				CANID按位过滤: bitn=FALSE, 禁止dwCanId.bitn过滤; bitn=TRUE, 使能dwCanId.bitn过滤

**4.10.4.3 CANBus功能块**

CANBus功能块名称、含义和触发方式如下表所示。

名称	含义	触发方式
CANBus_CtrlDevState	控制设备的状态	边沿触发 (上升沿)
CANBus_DevRxMsg	设备接收单帧信息	电平触发 (高电平), 单帧缓存
CANBus_DevRxMultiMsg	设备接收多帧信息	电平触发 (高电平), 多帧缓存
CANBus_DevTxMsg	设备发送信息	边沿触发 (上升沿)
CANBus_GetDevDiagInfo	获取设备的诊断信息	电平触发 (高电平)
CANBus_GetIoDrvVer	获取CANBus组件版本信息	电平触发 (高电平)
CANBus_SetDevBaud	设置设备的波特率	边沿触发 (上升沿)

CANBus功能块相关说明如下：

- 功能块中的输入参数uiDevId设备ID，必须与CANBus设备的网络ID对应。
- 边沿触发功能块的输入条件为xExecute，表示上升沿触发，低电平中止/复位。
- 电平触发功能块的输入条件为xEnable，表示高电平触发，低电平停止/复位。
- 同一时刻，最多支持使能128个实例化接收功能块，包括CANBus\_DevRxMsg和CANBus\_DevRxMultiMsg。
- 同一时刻，仅能使能单个实例化CANBus\_DevTxMsg功能块，否则会出现发送忙错误。

**CANBus\_CtrlDevState控制设备的状态**

## FUNCTION\_BLOCK CANBus\_CtrlDevState

名称	类型	继承自	地址	初始化	注释
xExecute	BOOL				功能块触发标志：上升沿触发，低电平停止/复位
uiDevId	UINT				实例化CANBus设备的网络ID
eCtrCmd	CANBus_CtrlCmdType_Enum				控制设备状态切换命令
xDone	BOOL			FALSE	完成标志
xBusy	BOOL			FALSE	运行标志
xError	BOOL			FALSE	错误标志
eErrorID	CANBus_FBEErrorType_Enum				错误码
eDevState	CANBus_StateType_Enum				设备状态

## CANBus\_DevRxMsg设备接收单帧信息



## FUNCTION\_BLOCK CANBus\_DevRxMsg

名称	类型	继承自	地址	初始化	注释
xEnable	BOOL			FALSE	功能块激活标志：高电平激活，低电平停止/复位
uiDevId	UINT				实例化CANBus设备的网络ID
stRxFilterMsg	CANBus_RxFilter_t				接收过滤信息设置
xDone	BOOL			FALSE	完成标志
xBusy	BOOL			FALSE	运行标志
xError	BOOL			FALSE	错误标志
eErrorID	CANBus_FBEErrorType_Enum				错误码
udiFrameCnt	UDINT				帧计数器，满足条件的CAN接收帧
stRxMsg	CANBus_FrameMsg_t				接收信息，最近一次接收CAN报文

参数名称	参数说明
xEnable	高电平使能功能块，低电平复位功能块。
uiDevId	设备ID，必须与CANBus设备的网络ID对应。
stRxFilterMsg	设置接收参数和过滤器参数。
udiFrameCnt	帧计数器，表示当前接收到满足条件的帧数。
stRxMsg	保存满足条件的接收CAN帧信息（仅缓存最新一帧接收CAN报文）。

## CANBus\_DevRxMultiMsg 设备接收多帧信息



FUNCTION_BLOCK CANBus_DevRxMultiMsg						
名称	类型	继承自	地址	初始化	注释	
xEnable	BOOL			FALSE	功能块激活标志: 高电平激活, 低电平停止/复位	
xBuffResetFlag	BOOL			FALSE	复位缓存标志: TRUE-清除缓存, 缓存从头开始填充, 功能块自动清零	
xBuffReadFlag	BOOL			FALSE	读缓存标志: TRUE-应用层从缓存区读取数据, 该状态避免功能块写缓存	
uiDevId	UINT				实例化CANBus设备的网络ID	
stRxFilterMsg	CANBus_RxFilter_t				接收过滤信息设置	
xDone	BOOL			FALSE	完成标志	
xBusy	BOOL			FALSE	运行标志	
xError	BOOL			FALSE	错误标志	
eErrorID	CANBus_FBErroType_Enum				错误码	
uiBuffFrameMaxNum	UINT		0		接收缓存帧最大值: 0-没有缓存区; >0-缓存最大帧数	
uiBuffFrameNum	UINT		0		接收缓存帧数: 此值小于uiBuffFrameMaxNum, 缓存满后清零重新计数	
udiFrameCnt	UDINT		0		帧计数器, 满足条件的CAN接收帧	
stRxMsg	CANBus_FrameMsg_t				接收信息, 最近一次接收CAN报文	
astRxFrameBuff	POINTER TO CANBus_FrameMsg_t				接收帧缓存: 一维可变长度结构体数组, 数组元素个数不能超过65535	

参数名称	参数说明
xEnable	高电平使能功能块, 低电平复位功能块。
xBuffResetFlag	复位缓存标志, TRUE-清除缓存, 缓存从头开始填充, 功能块自动清零。 <ul style="list-style-type: none"><li>• 手动复位接收缓存数据。</li><li>• 在功能块使能的条件下触发。</li><li>• 高电平执行, xBuffResetFlag标志会自动复位。</li></ul>
xBuffReadFlag	读缓存标志, TRUE-应用层从缓存区读取数据, 该状态避免功能块写缓存。 <ul style="list-style-type: none"><li>• 在读缓存时, 可以通过置位该标志来防止新数据覆盖缓存中的数据。</li><li>• 特别注意的是, 在读缓存的过程中, 内部寄存器只保存最近一次接收报文。</li><li>• 为了避免接收出现丢帧, 应尽可能短的置位此标志并及时复位。</li></ul>
uiDevId	设备ID, 必须与CANBus设备的网络ID对应。
stRxFilterMs	设置接收参数和过滤器参数。
uiBuffFrameMaxNum	接收缓存最大帧数, 根据结构体数组自动计算。
uiBuffFrameNum	接收缓存帧数, 缓存满后自动清零, xBuffResetFlag置位可清零。
udiFrameCnt	帧计数器, 表示当前接收到满足条件的帧数。
stRxMsg	保存满足条件的接收CAN帧信息 (仅缓存最新一帧接收CAN报文)。
astRxFrameBuff	接收帧缓存, 一维可变长度结构体数组, 数组元素个数不能超过65535。 <ul style="list-style-type: none"><li>• astRxFrameBuff, 接口类型为结构体指针 (POINTER TO CANBus_FrameMsg_t)。实际上为可变长度的结构体数组 (ARRAY[*] OF CANBus_FrameMsg_t)。</li><li>• 传递参数时要用结构体数组, 使用其他数据类型, 程序编译时会给出错误提示。</li></ul>

同一时刻, 最多支持使能128个实例化接收功能块, 包括CANBus\_DevRxMsg和CANBus\_DevRxMultiMsg。

## CANBus\_DevTxMsg 设备发送信息

CANBus_DevTxMsg						
名称	类型	继承自	地址	初始化	注释	
xExecute	BOOL				功能块触发标志：上升沿触发，低电平停止/复位	BOOL xDone
uiDevId	UINT				实例化CANBus设备的网络ID	BOOL xBusy
uiTimeOutMs	UINT				超时时间，单位ms	BOOL xError
stTxMsg	CANBus_FrameMsg_t				发送信息，发送CAN报文	CANBus_FBErroType_Enum eErrorID
xDone	BOOL			FALSE	完成标志	
xBusy	BOOL			FALSE	运行标志	
xError	BOOL			FALSE	错误标志	
eErrorID	CANBus_FBErroType_Enum				错误码	

参数名称	参数说明
xEnable	上升沿使能功能块，下降沿和低电平中止并复位功能块。
uiDevId	设备ID，必须与CANBus设备的网络ID对应。
uiTimeOutMs	发送超时时间设置，单位ms。
stTxMsg	发送信息，包括扩展帧标志、远程帧标志、CANID、数据长度和数据。

- 若超时时间设为0，功能块运行时默认按照50ms检测发送超时。
- 同一时刻，仅能使能单个实例化CANBus\_DevTxMsg功能块，否则会出现发送忙错误。

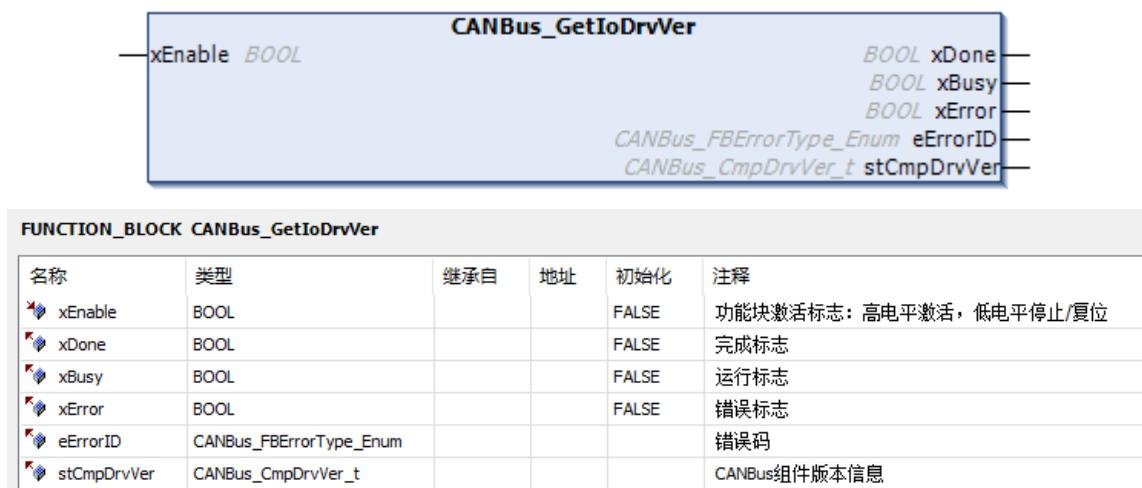
## CANBus\_GetDevDiagInfo 获取设备的诊断信息

CANBus_GetDevDiagInfo						
名称	类型	继承自	地址	初始化	注释	
xEnable	BOOL			FALSE	功能块激活标志：高电平激活，低电平停止/复位	BOOL xDone
uiDevId	UINT				实例化CANBus设备的网络ID	BOOL xBusy
xDone	BOOL			FALSE	完成标志	BOOL xError
xBusy	BOOL			FALSE	运行标志	CANBus_FBErroType_Enum eErrorID
xError	BOOL			FALSE	错误标志	CANBus_DiagInfo_t stDevDiag
eErrorID	CANBus_FBErroType_Enum				错误码	
stDevDiag	CANBus_DiagInfo_t				设备诊断信息	

参数名称	参数说明
stDevDiag	诊断信息实时更新，包括但不限于：运行状态、错误状态、DevID、运行波特率、总线负载率、收发计数器等。

## CANBus\_GetIoDrvVer 获取CANBus组件版本信息



stCmpDrvVer结构体中的参数统一使用十六进制格式。

## CANBus\_SetDevBaud 设置设备的波特率



eSetBaud CAN总线设置波特率，使用枚举类型可避免输入设备不支持的波特率。

### 4.10.4.4 CANBus功能块错误码

CANBus功能块错误码如下表所示。

## CANBus\_CtrlDevState控制设备的状态错误码

错误类型	错误码	含义	解决方法
CANBUS_FBERR_NO_ERROR	0x0000	功能块运行正常或者无错误	功能块正常执行
CANBUS_FBERR_EXE_FAILED	0x0001	功能块运行失败	输入DevID无效
CANBUS_FBERR_INVALID_DEVICEID	0x0002	无效的设备ID，该ID未使能	输入DevID无效
CANBUS_FBERR_DEVICE_FAULT	0x0003	设备存在故障	设备正在启动中

错误类型	错误码	含义	解决方法
CANBUS_FBERR_WRONG_PARAMETER	0x0004	功能块输入参数错误	输入CtrlCmd无效
CANBUS_FBERR_BUSY_ERROR	0x0005	功能块忙	-
CANBUS_FBERR_ABORT_ERROR	0x0006	功能块中止错误	-
CANBUS_FBERR_TIME_OUT_ERROR	0x0007	功能块超时错误	-
CANBUS_FBERR_POINTER_NULL	0x0008	功能块指针空错误	-
CANBUS_FBERR_BUFF_FULL	0x0009	功能块缓存满错误	-
CANBUS_FBERR_UNKNOWN_ERROR	0x00FF	功能块未知错误	-

### CANBus\_DevRxMsg 设备接收单帧信息

错误类型	错误码	含义	解决方法
CANBUS_FBERR_NO_ERROR	0x0000	功能块运行正常或者无错误	功能块正常执行
CANBUS_FBERR_EXE_FAILED	0x0001	功能块运行失败	参数地址被修改
CANBUS_FBERR_INVALID_DEVICEID	0x0002	无效的设备ID，该ID未使能	输入DevID无效
CANBUS_FBERR_DEVICE_FAULT	0x0003	设备存在故障	-
CANBUS_FBERR_WRONG_PARAMETER	0x0004	功能块输入参数错误	-
CANBUS_FBERR_BUSY_ERROR	0x0005	功能块忙	实例化功能块超限
CANBUS_FBERR_ABORT_ERROR	0x0006	功能块中止错误	-
CANBUS_FBERR_TIME_OUT_ERROR	0x0007	功能块超时错误	-
CANBUS_FBERR_POINTER_NULL	0x0008	功能块指针空错误	-
CANBUS_FBERR_BUFF_FULL	0x0009	功能块缓存满错误	-
CANBUS_FBERR_UNKNOWN_ERROR	0x00FF	功能块未知错误	-

### CANBus\_DevRxMultiMsg 设备接收多帧信息

错误类型	错误码	含义	解决方法
CANBUS_FBERR_NO_ERROR	0x0000	功能块运行正常或者无错误	功能块正常执行
CANBUS_FBERR_EXE_FAILED	0x0001	功能块运行失败	参数地址被修改
CANBUS_FBERR_INVALID_DEVICEID	0x0002	无效的设备ID，该ID未使能	输入DevID无效
CANBUS_FBERR_DEVICE_FAULT	0x0003	设备存在故障	-

错误类型	错误码	含义	解决方法
CANBUS_FBERR_WRONG_PARAMETER	0x0004	功能块输入参数错误	最大缓存数为0
CANBUS_FBERR_BUSY_ERROR	0x0005	功能块忙	实例化功能块超限
CANBUS_FBERR_ABORT_ERROR	0x0006	功能块中止错误	-
CANBUS_FBERR_TIME_OUT_ERROR	0x0007	功能块超时错误	-
CANBUS_FBERR_POINTER_NULL	0x0008	功能块指针空错误	未指定输入输出参数或者缓存地址
CANBUS_FBERR_BUFF_FULL	0x0009	功能块缓存满错误	接收缓存满
CANBUS_FBERR_UNKNOWN_ERROR	0x00FF	功能块未知错误	-

**CANBus\_DevTxMsg 设备发送信息**

错误类型	错误码	含义	解决方法
CANBUS_FBERR_NO_ERROR	0x0000	功能块运行正常或者无错误	功能块正常执行
CANBUS_FBERR_EXE_FAILED	0x0001	功能块运行失败	输入DevID无效
CANBUS_FBERR_INVALID_DEVICEID	0x0002	无效的设备ID，该ID未使能	输入DevID无效
CANBUS_FBERR_DEVICE_FAULT	0x0003	设备存在故障	设备未运行或故障
CANBUS_FBERR_WRONG_PARAMETER	0x0004	功能块输入参数错误	CANID无效，数据长度超限 (数据帧大于8，远程帧大于0)
CANBUS_FBERR_BUSY_ERROR	0x0005	功能块忙	发送请求冲突
CANBUS_FBERR_ABORT_ERROR	0x0006	功能块中止错误	-
CANBUS_FBERR_TIME_OUT_ERROR	0x0007	功能块超时错误	超时未发送成功
CANBUS_FBERR_POINTER_NULL	0x0008	功能块指针空错误	-
CANBUS_FBERR_BUFF_FULL	0x0009	功能块缓存满错误	-
CANBUS_FBERR_UNKNOWN_ERROR	0x00FF	功能块未知错误	-

**CANBus\_GetDevDiagInfo 获取设备的诊断信息**

错误类型	错误码	含义	解决方法
CANBUS_FBERR_NO_ERROR	0x0000	功能块运行正常或者无错误	功能块正常执行
CANBUS_FBERR_EXE_FAILED	0x0001	功能块运行失败	-
CANBUS_FBERR_INVALID_DEVICEID	0x0002	无效的设备ID，该ID未使能	输入DevID无效
CANBUS_FBERR_DEVICE_FAULT	0x0003	设备存在故障	-

错误类型	错误码	含义	解决方法
CANBUS_FBERR_WRONG_PARAMETER	0x0004	功能块输入参数错误	-
CANBUS_FBERR_BUSY_ERROR	0x0005	功能块忙	-
CANBUS_FBERR_ABORT_ERROR	0x0006	功能块中止错误	-
CANBUS_FBERR_TIME_OUT_ERROR	0x0007	功能块超时错误	-
CANBUS_FBERR_POINTER_NULL	0x0008	功能块指针空错误	-
CANBUS_FBERR_BUFF_FULL	0x0009	功能块缓存满错误	-
CANBUS_FBERR_UNKNOWN_ERROR	0x00FF	功能块未知错误	-

### CANBus\_GetIoDrvVer 获取CANBus组件版本信息

错误类型	错误码	含义	解决方法
CANBUS_FBERR_NO_ERROR	0x0000	功能块运行正常或者无错误	功能块正常执行
CANBUS_FBERR_EXE_FAILED	0x0001	功能块运行失败	-
CANBUS_FBERR_INVALID_DEVICEID	0x0002	无效的设备ID，该ID未使能	-
CANBUS_FBERR_DEVICE_FAULT	0x0003	设备存在故障	-
CANBUS_FBERR_WRONG_PARAMETER	0x0004	功能块输入参数错误	-
CANBUS_FBERR_BUSY_ERROR	0x0005	功能块忙	-
CANBUS_FBERR_ABORT_ERROR	0x0006	功能块中止错误	-
CANBUS_FBERR_TIME_OUT_ERROR	0x0007	功能块超时错误	-
CANBUS_FBERR_POINTER_NULL	0x0008	功能块指针空错误	-
CANBUS_FBERR_BUFF_FULL	0x0009	功能块缓存满错误	-
CANBUS_FBERR_UNKNOWN_ERROR	0x00FF	功能块未知错误	-

### CANBus\_SetDevBaud 设置设备的波特率

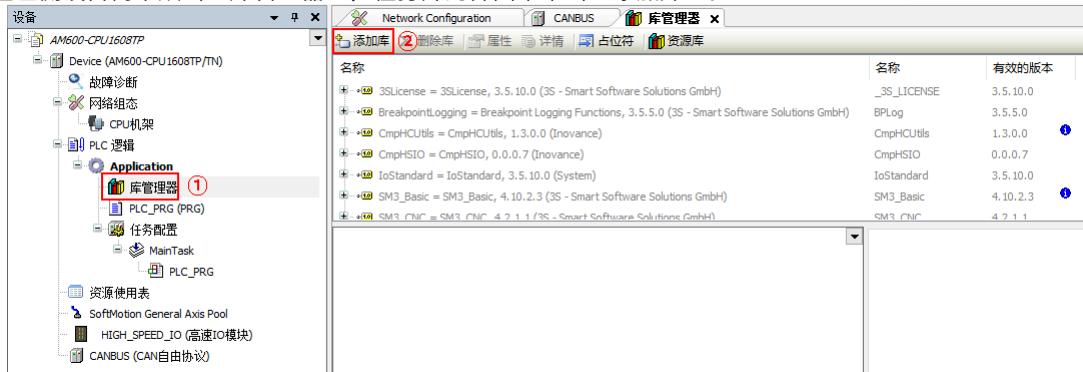
错误类型	错误码	含义	解决方法
CANBUS_FBERR_NO_ERROR	0x0000	功能块运行正常或者无错误	功能块正常执行
CANBUS_FBERR_EXE_FAILED	0x0001	功能块运行失败	波特率更改失败
CANBUS_FBERR_INVALID_DEVICEID	0x0002	无效的设备ID，该ID未使能	输入DevID无效
CANBUS_FBERR_DEVICE_FAULT	0x0003	设备存在故障	-

错误类型	错误码	含义	解决方法
CANBUS_FBERR_WRONG_PARAMETER	0x0004	功能块输入参数错误	波特率输入错误
CANBUS_FBERR_BUSY_ERROR	0x0005	功能块忙	-
CANBUS_FBERR_ABORT_ERROR	0x0006	功能块中止错误	-
CANBUS_FBERR_TIME_OUT_ERROR	0x0007	功能块超时错误	-
CANBUS_FBERR_POINTER_NULL	0x0008	功能块指针空错误	-
CANBUS_FBERR_BUFF_FULL	0x0009	功能块缓存满错误	-
CANBUS_FBERR_UNKNOWN_ERROR	0x00FF	功能块未知错误	-

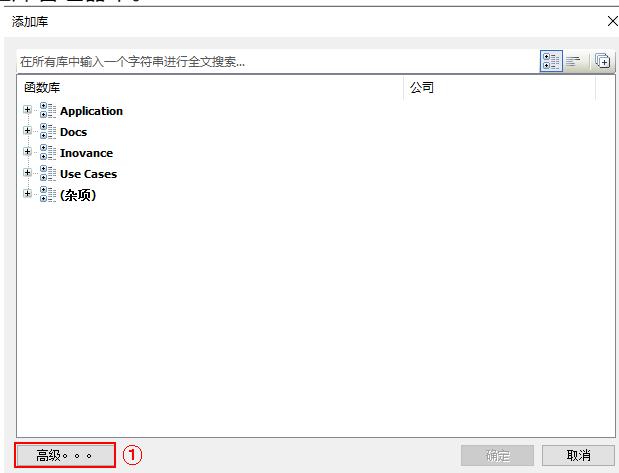
#### 4.10.4.5 CANBus功能块使用示例

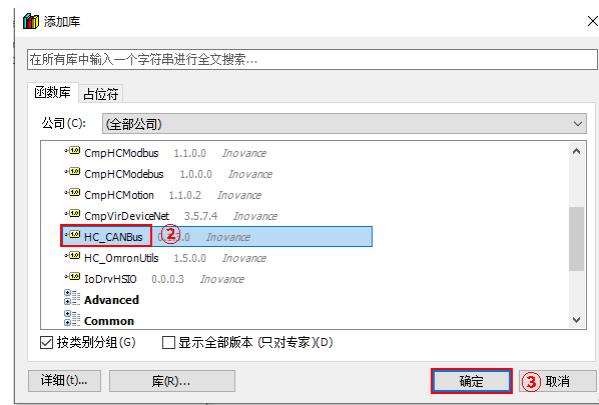
##### 添加CANBus库

1. 在左侧设备树中双击“库管理器”，在打开的界面中单击“添加库”。



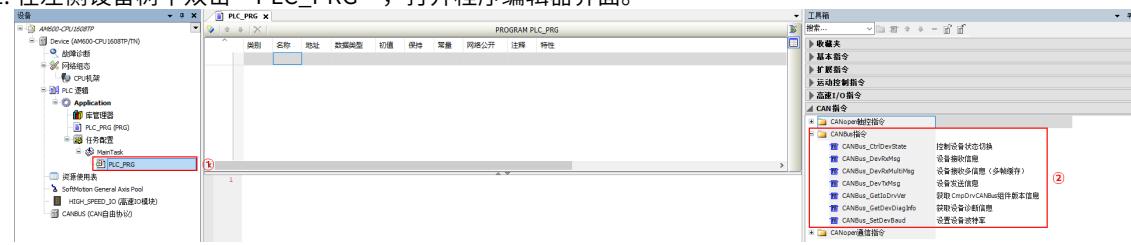
2. 在打开的对话框中单击“高级”，在打开的界面中展开“Inovance”，选择“HC\_CANBus”，单击“确定”，添加CANBus库至库管理器中。





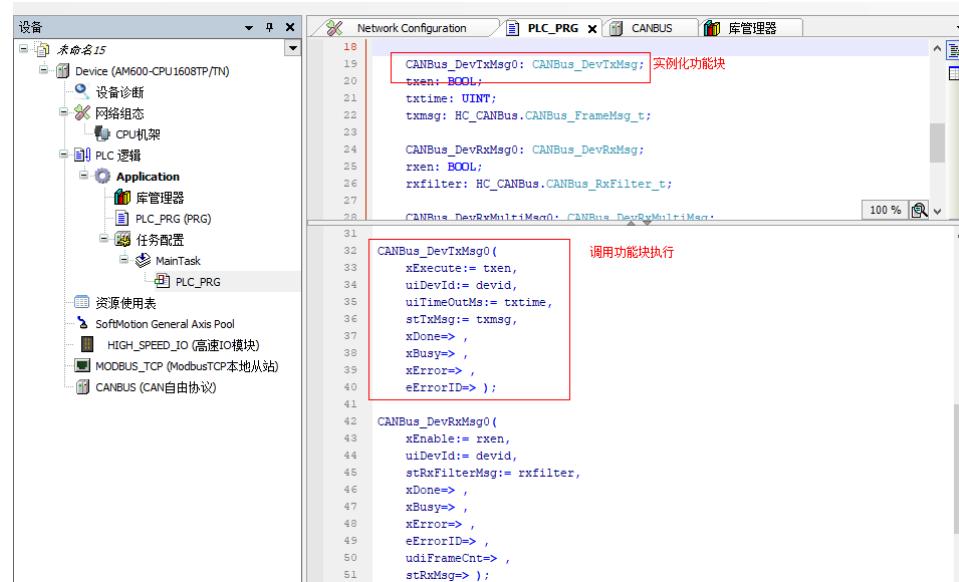
## 使用CANBus功能块指令

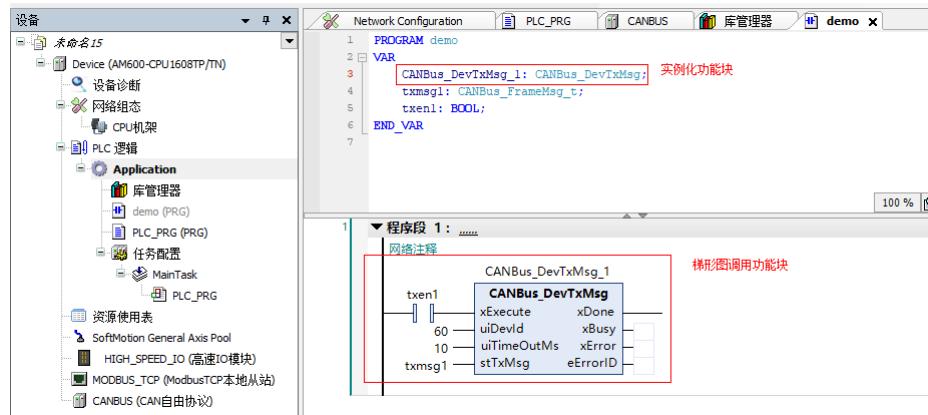
1. 在左侧设备树中双击“PLC\_PRG”，打开程序编辑器界面。



2. 在工具箱展开“CAN指令 > CANBus指令”，双击相应CANBus功能块指令使用该指令并进行实例化。

CANBus功能块指令支持ST语言和LD语言，分别如下图所示。

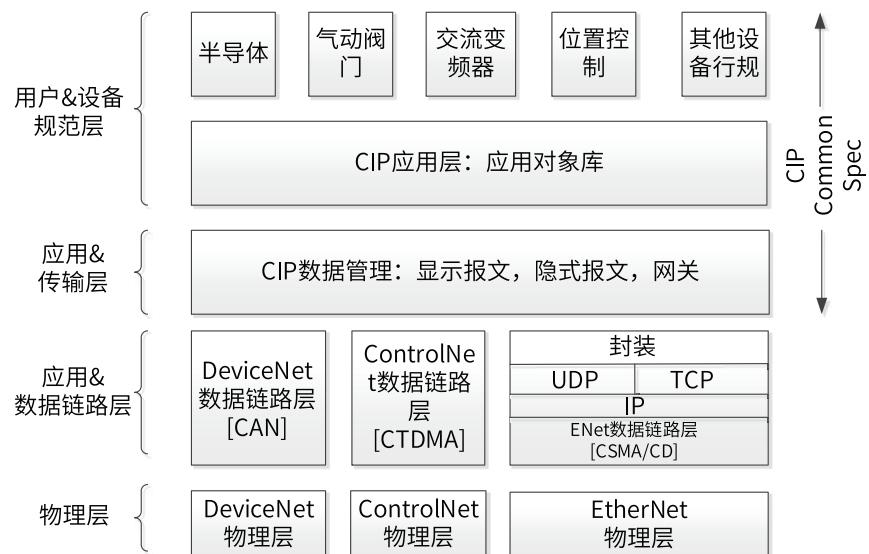




## 4.11 EtherNet/IP 通信

### 4.11.1 协议概述

EtherNet/IP通讯协议是适合工业环境应用的协议体系（IP是“Industrial Protocol”的简称），基于传统的以太网协议和标准的TCP/IP协议，可以实现工业设备之间应用信息的高效交换。EtherNet/IP应用层协议使用标准的面向对象的CIP协议<sup>[1]</sup>。各层结构如图所示。



EtherNet/IP协议的技术特点：

1. 标准化。EtherNet/IP建立在标准的TCP/UDP协议之上，完全符合标准的IEEE802.3U标准，所有有标准以太网节点的设备均可加入此网络。
2. 实时性<sup>[2]</sup>。EtherNet/IP数据传输分为隐式报文通讯（Implicit Messaging）和显式报文通讯（Explicit Messaging）。隐式通讯用于传输实时性数据，采用周期循环方式；显式通讯用于传输非实时性数据（如配置信息等），采用请求-应答的方式。
3. 通讯效率高。EtherNet/IP采用生产者/消费者技术，允许网络上的节点同时存取同一个源的数据。在生产/消费模式中，对每个数据均分配一个唯一的标识，每个数据源将数据一次性的发送到网络上，其他节点选择性的读取这些数据，从而极大的提高系统的通信效率。

4. EtherNet/IP设备应用广泛。EtherNet/IP通讯设备包括简单的I/O设备、传感器（扫码枪/相机）、执行器以及复杂的控制设备（机器人/PLC/焊机等）。

InoProShop1.5.0及以上版本支持EtherNet/IP功能，其通讯规格如下：

- 中型PLC均支持一路EtherNet/IP主站和从站功能，可同时支持主从站。
- 主从站均支持Class1标签或者实例路径连接，从站支持Class3/Ucmm服务消息标签连接。
- 最小循环通讯周期（RPI）为5ms。
- 单个连接最大支持1400个字节数据读写。
- EtherNet/IP主站最大支持64个从站。
- EtherNet/IP从站最大支持连接个数为32个。

## 说明

- [1]: 协议详情请参考EIP-CIP-V1-1.0、EIP-CIP-V2-1.0官方标准文档。
- [2]: 在同时有EtherCAT和EtherNet/IP网络的组网工程中，由于默认EtherCAT优先级为最高，故会降低EtherNet/IP网络的通讯实时性。

### 4.11.2 EtherNet-IP通信规格

项目			AM300系列	AM400系列	AM500系列	AM600系列	AC700系列	AC800系列
CIP服务	隐式(I/O)报文通信	发起端IO连接数量 <sup>[1]</sup>	16	16	16	16	32	64
		目标端IO连接数量 <sup>[2]</sup>	16	16	16	16	32	64
		RPI (通信周期)	5~50000 (单位：ms)					
		隐式(I/O)报文通信允许通信带宽	(@4 Byte)	6400 (pps) <sup>[3]</sup>	6400 (pps)	6400 (pps)	12800 (pps)	25600 (pps)
			(@250 Byte)	4800 (pps)	4800 (pps)	4800 (pps)	12800 (pps)	25600 (pps)
			(@500 Byte)	3200 (pps)	3200 (pps)	3200 (pps)	12800 (pps)	25600 (pps)
			(@1400 Byte)	1000 (pps)	1000 (pps)	1000 (pps)	12800 (pps)	25600 (pps)
	显式报文通信	每个连接的最大数据大小 <sup>[4]</sup>	1400 Bytes	1400 Bytes	1400 Bytes	1400 Bytes	1400 Bytes	1400 Bytes
		组播过滤功能 <sup>[5]</sup>	支持 (IGMP客户端功能)					
		目标端Class3标签数量 <sup>[6]</sup>	128	128	128	128	128	128
		目标端UCMM标签数量 <sup>[6]</sup>	128	128	128	128	128	128

[1] “发起端IO连接”包括：

- 消费者标签：作为发起端，以生产者标签的名称作为连接路径，请求建立连接
- 发起端通用IO连接：作为发起端，以目标端通用IO连接的实例ID作为连接路径，请求建立连接

[2] “目标端IO连接”包括：

- 生产者标签：作为目标端，以生产者标签的名称作为连接，响应建立连接的请求
- 目标端通用IO连接：作为目标端，以目标端通用IO连接的实例ID作为连接路径，响应建立连接的请求

[3]通信带宽pps是Packet Per Second 的简称，pps是网络吞吐率的单位，此处表示1秒内可处理的发送和接收的分组数据包的数量总和，计算公式：通信带宽pps = 1000ms/RPI \* 连接数量 \* 2

[4]保证连接内的数据同时性。数据大小在 509 字节以上（大于509）时，所用设备支持 Large Forward Open (CIP 选项规格)

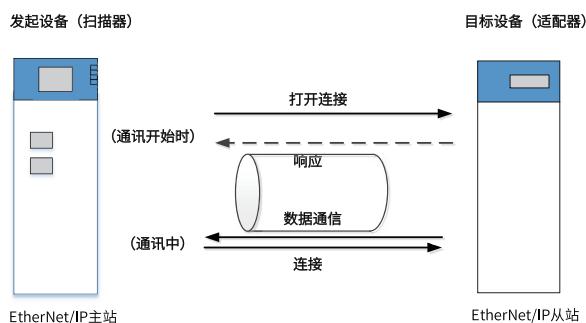
[5]EtherNet/IP 单元具有 IGMP 客户端功能，因此，使用支持 IGMP Snooping 的以太网交换机，可过滤掉不关注的组播数据包

[6]关于标签数量规格，说明如下：

- 发起端标签包括：消费者标签、发起端通用IO连接、发起端Class3标签、发起端UCMM标签，最大数量：32个
- 目标端标签包括：生产者标签、目标端通用IO连接、目标端Class3标签、目标端UCMM标签，最大数量：32个，其中，目标端通用IO连接的最大数量为16个

### 4.11.3 PLC 作为EtherNet-IP 主站的配置

通讯开始时，打开连接的一端称为发起设备，也叫扫描器（Scanner），为通常意义的EtherNet/IP主站。被打开的一端称为目标设备，也叫适配器（Adapter），也就是通常意义上的EtherNet/IP从站。

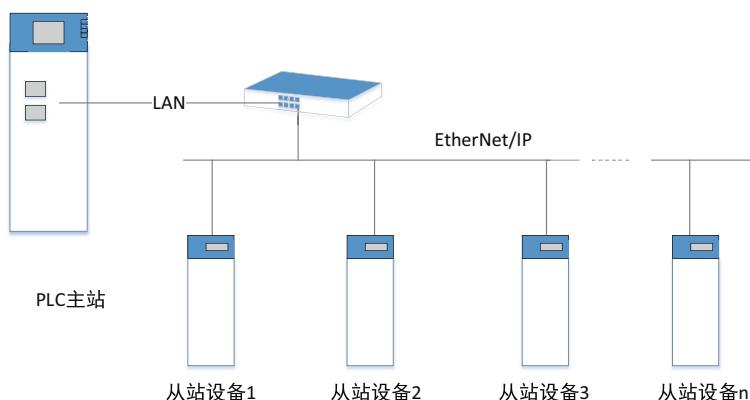


本章节将详细介绍在InoProShop后台以汇川技术中型PLC为EtherNet/IP主站的配置方法。

#### EtherNet/IP设备IP设置

EtherNet/IP支持的拓扑结构有总线型结构、星型结构、混合结构和环形结构。

对于星型结构，所有的节点都连接在网络集线器上，网络材料价格低廉，搭接容易，市面上可以找到很多合适的设备，且增减节点和维修都很方便，是目前经常采用的网络结构。在星型结构中，同一个EtherNet/IP网络中的所有设备IP地址都需要设置在同一个网段，且保证所有设备的IP地址不重复。



例如，网络中的设备IP可以设置为如下的形式：

EtherNet/IP主站的IP地址：192.168.1.100

EtherNet/IP从站1的IP地址：192.168.1.101

EtherNet/IP从站2的IP地址：192.168.1.102

.....

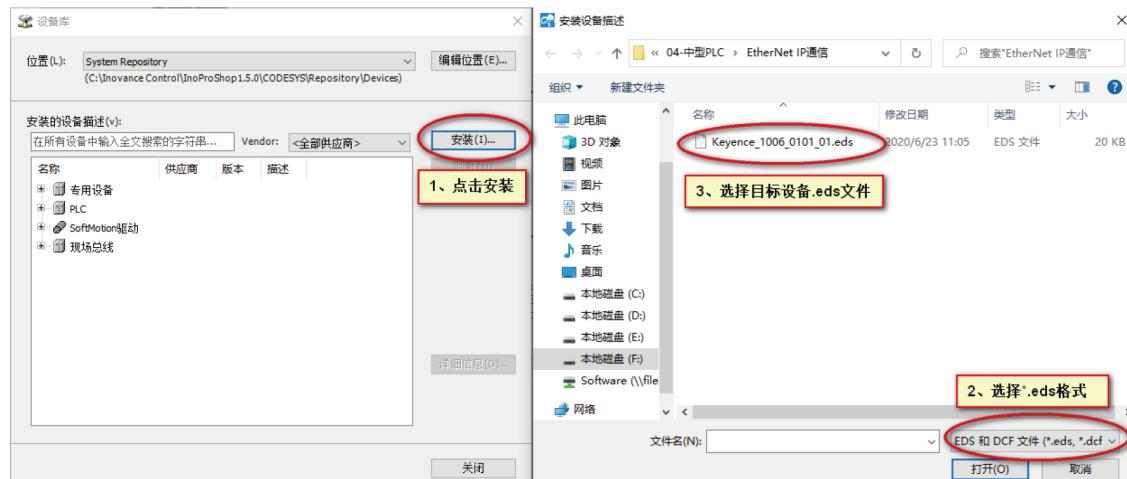
EtherNet/IP从站n的IP地址：192.168.1.XXX

## 添加EtherNet/IP远程从站

### 1. 导入eds从站设备描述文件

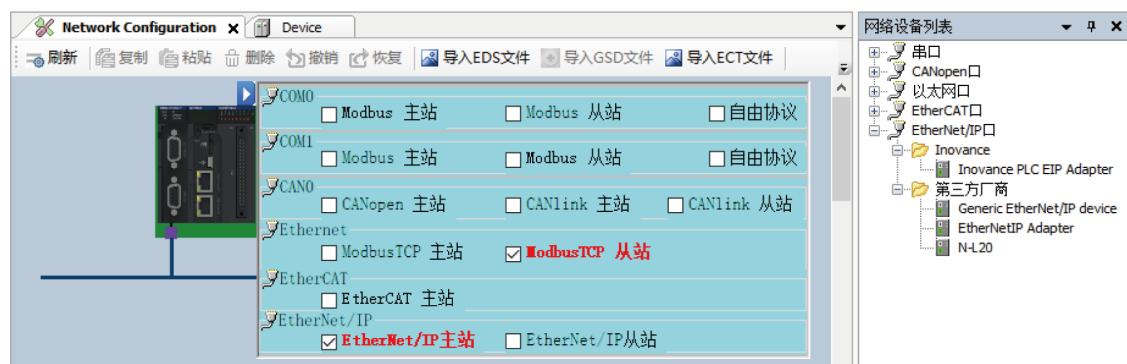
InoProShop后台提供两种方法导入EtherNet/IP从站设备的eds文件。一种是通过设备库添加，点击菜单栏->工具->设备库，点击“安装”导入；另外一种是直接在“Network Configuration”网络组态界面，选择“导入eds文件”导入。

以基恩士N-L20 EtherNet/IP从站模块设备为例，按照第一种方法，参考下图三个步骤，可以导入第三方设备的eds描述文件。



### 2. 勾选“EtherNet/IP”主站

鼠标单击主站设备，在弹出的界面中，勾选“EtherNet/IP主站”。

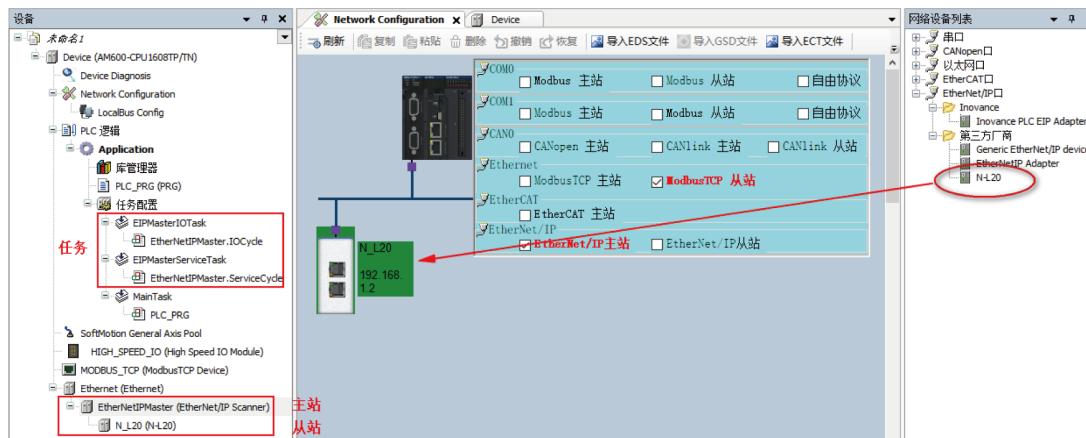


### 3. 添加EtherNet/IP远程从站

- 方式一：通过“网络设备列表”界面来添加

在右侧“网络设备列表”界面中，在“EtherNet/IP口列表”下，找到第一步导入的第三方厂商设备，鼠标双击，将该从站设备加入网络组态。

如下图所示，右边的设备树中，主站下生成了N\_L20从站设备，如果有多个EtherNet/IP远程从站设备，可以依次添加。



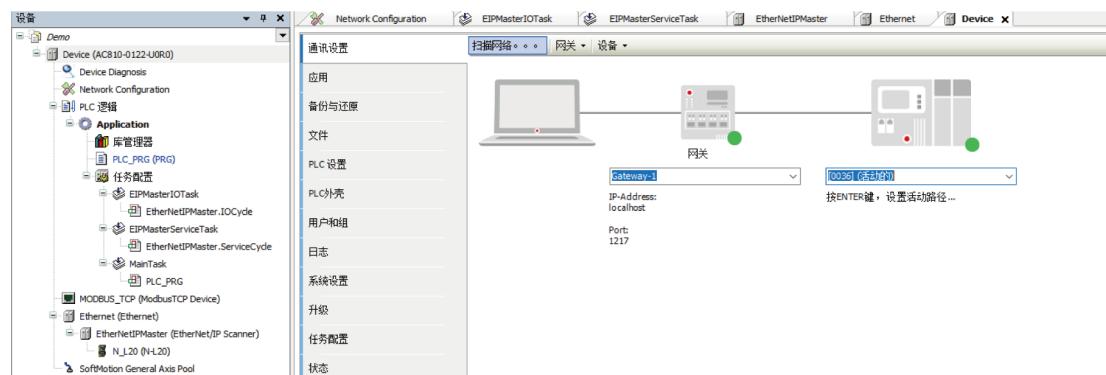
在加入EtherNet/IP设备后的任务配置下，系统自动生成了EIPMaster.IOTask和EIPMaster.ServiceTask两个任务，用于更新EtherNet/IP的循环通讯数据和服务数据。EIPMaster.IOTask任务优先级默认为0，用户可根据实际情况进行调整，比如工程中另外有EtherCAT任务，其优先级必须指定为0（最高），那么EIPMaster.IOTask的优先级可以修改为1。

- 方式二：通过EtherCAT扫描功能来添加

在添加从站eds文件之后，可登录PLC，选择设备树“EtherNetIPMaster”，右键选择“扫描设备”功能，扫描当前网络下的EtherNet/IP设备，再通过“拷贝所有扫描设备”功能添加从站，详细操作请参考EtherCAT扫描功能。

## EtherNet通用设置

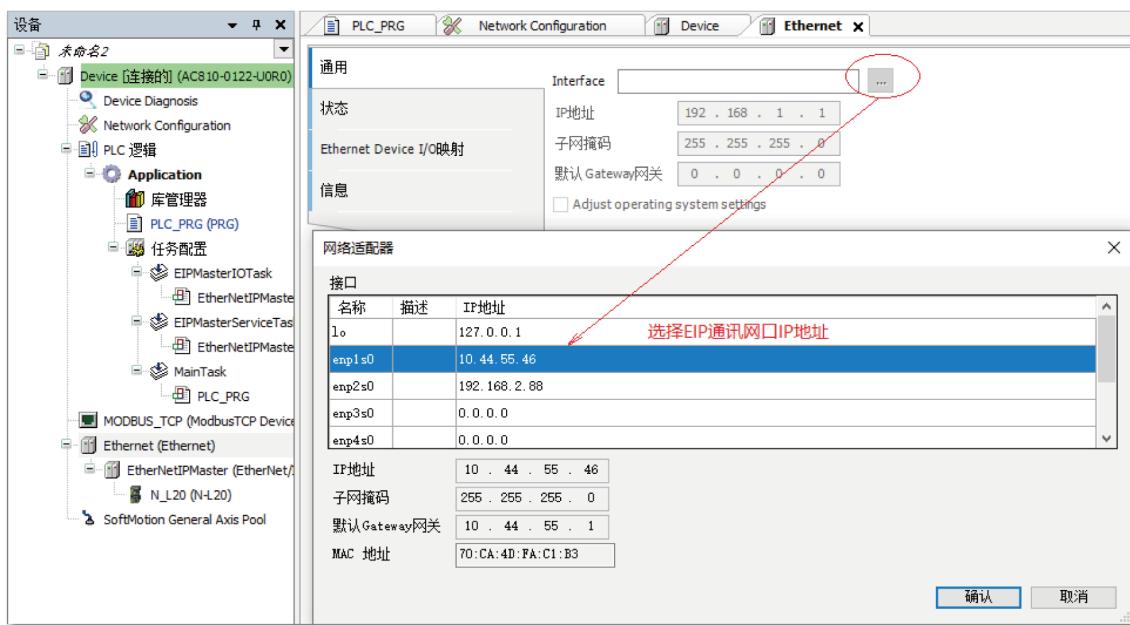
在EtherNet通用设置之前，首先配置PLC网关，可实现通过后台登录连接PLC。



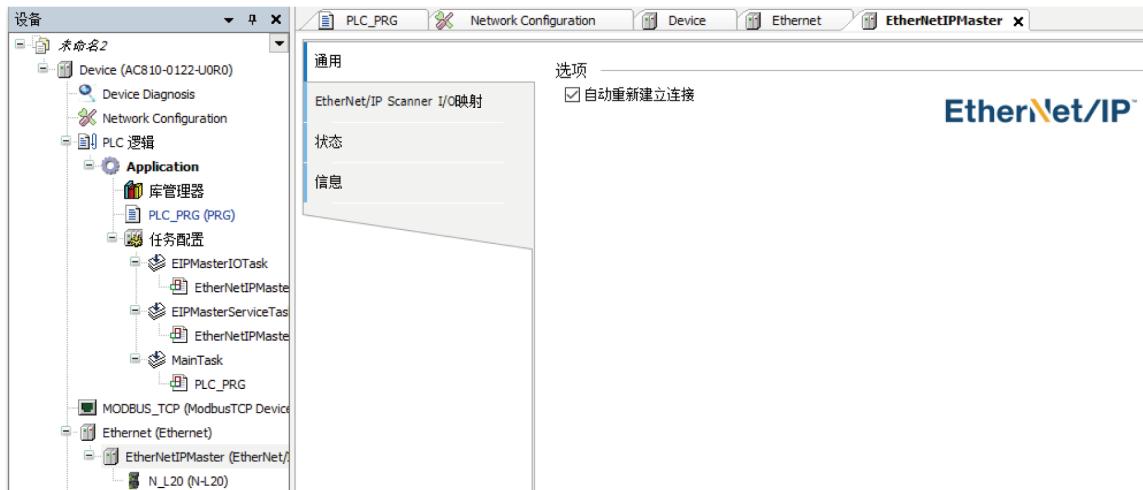
双击Ethernet设备，进入“通用”设置界面，如下图，打开“网络适配器”界面。

选择对应网络接口的主站IP地址。

注意：该IP地址是指PLC作为主站与外部设备连入EtherNet/IP网络的IP地址。对于AC800系列，具备双网口，请注意区分连接设备的IP和网口对应是否一致；对于AM600系列，只有一个以太网口，请确保其IP地址和登陆PLC的IP地址相一致。在设备初始界面中未扫描确认的PLC设备无法获取网口适配器对应的IP等信息。

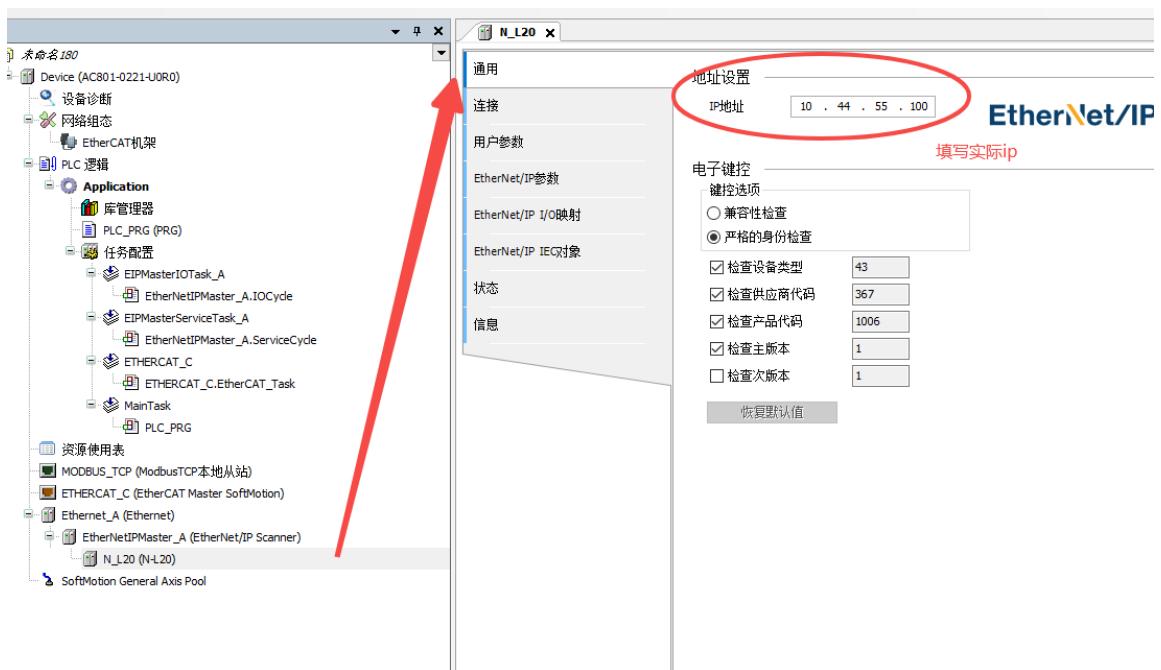


另外,主站通用设置界面,默认勾选自动重新建立连接,在从站链路丢失恢复时可以自动重连,如图所示。



## EtherNet/IP远程从站设置

双击需要设置的EtherNet/IP从站“N-L20”，进入设置界面。需要设置从站IP地址为远程从站设备实际的IP地址。

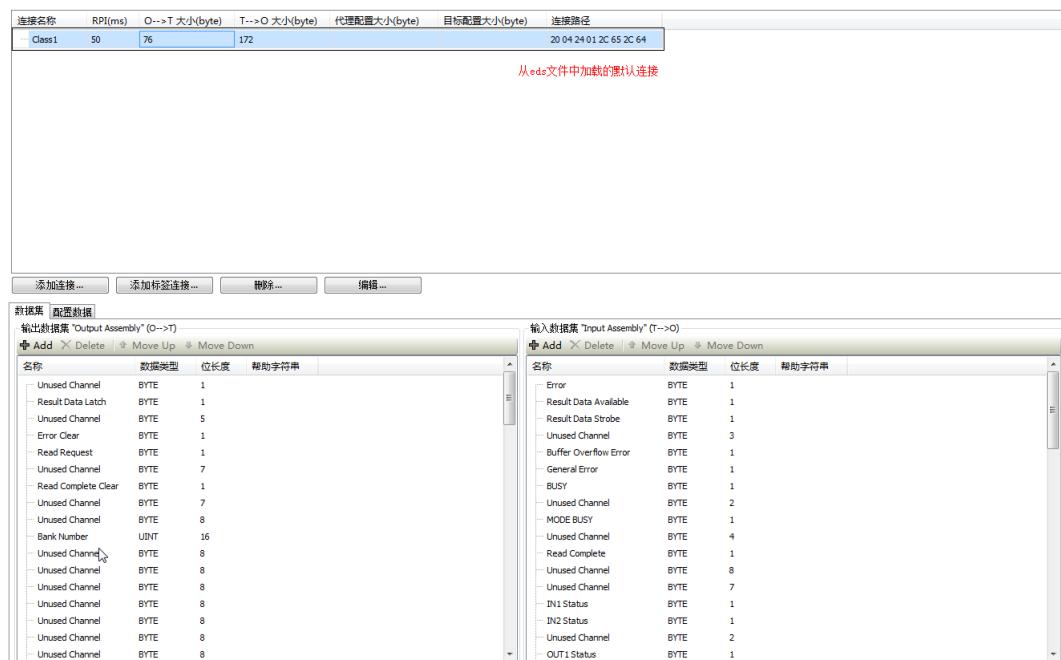


在电子键控一栏显示了从从站eds文件中加载的EtherNet/IP产品设备信息，在通讯连接请求时将默认严格的身份检查，即检查实际产品信息和请求连接的信息是否匹配一致，如果不匹配将会产生连接故障。实际使用时也可以根据需要勾选或则修改产品检查相关信息

## 连接设置

### 1. 添加连接

EtherNet/IP远程从站支持实例ID路径和标签路径的连接，其中实例ID路径的连接可以通过“添加连接”进行创建。EtherNet/IP远程从站eds描述文件中一般包含一个或者多个连接信息，在添加EtherNet/IP网络组态后，远程从站的连接界面会加载第一条连接为默认的连接，如下所示。



其中该界面数据集部分可展示默认定义的参数名，参数类型以及位长度。在未定义参数名的eds连接中，用户可根据数据传输长度自由组合数据类型以及参数命令

通过“添加连接”选择设置eds文件中默认预定义的其他连接路径。如下图所示。



## 2. 编辑连接

在上图中单击“编辑连接”，可进入连接设置界面，一般情况下默认预定义路径的连接除了RPI（通讯周期）、传输字节大小等参数需要根据具体应用做修改以外，其他参数均直接使用默认值。

编辑连接中包含了EtherNet/IP主站请求连接的主要配置参数，如图所示。下面对其中重要参数进行说明。



- 连接路径：该参数规定了一帧字节流的格式和连接实例。例如连接路径：20 04 2C C6 2C 68（详情参考EIP-CIP-V1-1.0, Appendix C: Data Management）。

20: Logical Segment、ClassID、8bit logical address

04: 表示Assembly Object (04H)

2C: Logical Segment、Connection Point、8bit logical address

C6: Assembly Object的实例的ID-C6H

2C: Logical Segment、Connection Point、8bit logical address

68: Assembly Object的实例的ID-68H

注意：连接路径因厂家而异，请根据具体的从站手册来配置。

- RPI (ms) : Requested Packet Interval的简称，以ms为单位的通讯传输间隔周期，各个节点的RPI可单独设置，互不影响。



### 注意

主站RPI周期必须为任务周期的整数倍。

- 传输字节大小

O->T Size (Bytes) : 从生产（发起设备）到消费（目标设备）传输的数据量，以byte为单位。

T->O Size (Bytes) : 从消费（目标设备）到生产（发起设备）传输的数据量，以byte为单位。

- 传输类型 (Transport Type)

专有所有者 (Exclusive Owner) :可同时设定“从发起设备到目标设备的数据发送”和“从目标设备到发起设备的数据接收”。

冗余所有者 (Redundant Owner) :允许多个发起设备对同一个目标设备建立相对独立的、相同的连接。

只输入 (Input Only) :此连接只能设定“从目标设备到发起设备的数据接收”。

只监听 (Listen Only) :应用此连接类型监听组播数据，而不提供配置或调度信息的 EtherNet/IP设备。

- 触发类型 (Trigger Type)

循环的 (Cyclic) :定期触发数据传输。

状态改变 (Change-Of-State) :检测到应用对象状态发生改变时传输数据。

应用程序 (Application Object) :应用对象触发时传输数据。

- 模式 (Connection Type)

组播 (Multicast) :多台扫描器同时接收一台目标设备的数据。

点对点 (Point-to-Point) :扫描器以一对一方式接收目标设备的数据。

### 3. 添加通用连接

在“连接”页面单击“添加连接”，打开连接设置界面，选择“通用连接”如下图所示。建议使用默认的连接配置（可根据从站规格自定义创建连接路径，但需要使用者有一定的CIP协议基础）。针对汇川PLC从站，只需要修改组合配置实例ID，组合消耗（O->T）数据集实例ID，组合消耗（T->O）数据集实例ID即可。



#### 4. 添加标签连接

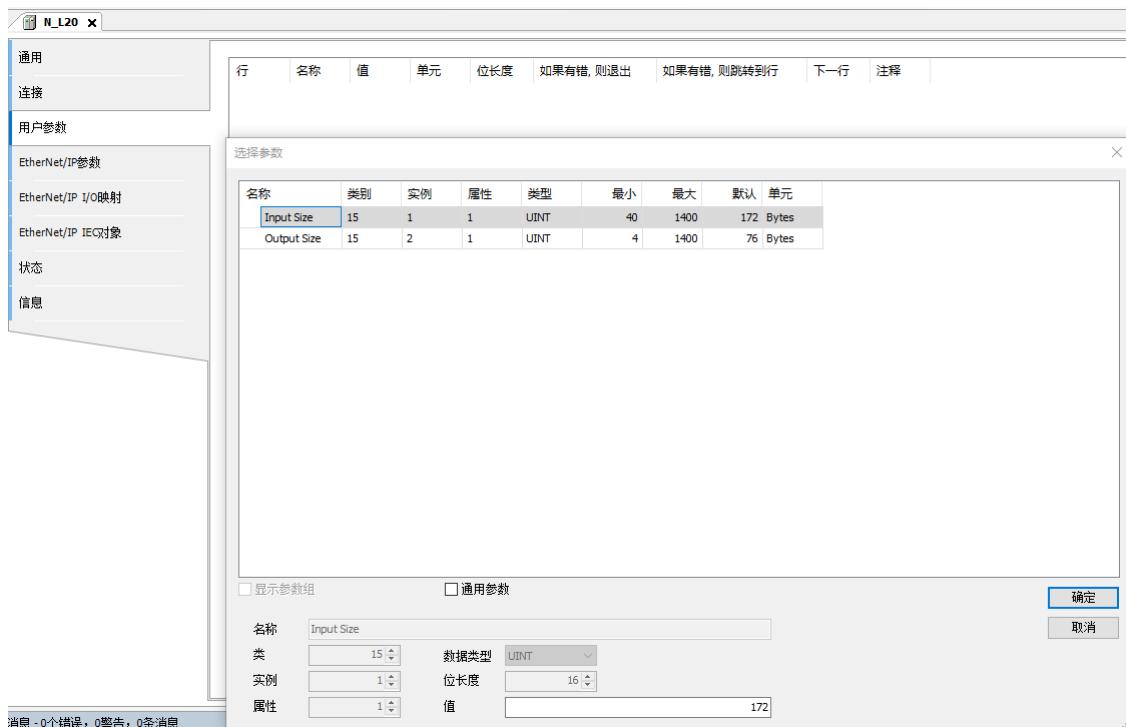
在“连接”页面单击“添加标签连接”，可快速添加一条标签路径的连接。标签连接类型默认Input Only，只读取消费从站发送的生产者标签。可在当前条目中直接修改T->O大小和连接路径（标签名），其他连接参数根据需要可在编辑连接中详细修改。

连接名称	RPI(ms)	O-->T 大小(byte)	T-->O 大小(byte)	代理配置大小(byte)	目标配置大小(byte)	连接路径
Class1	50	76	172			20 04 24 01 2C 65 2C 64
消费者标签	50	0	4			tag0

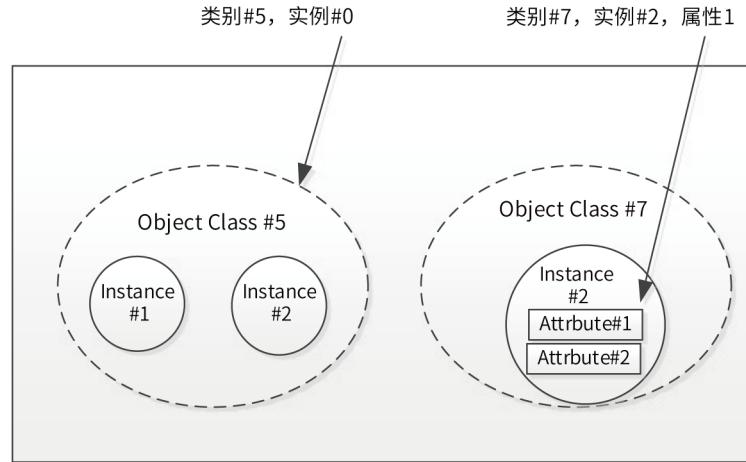
添加连接...    添加标签连接...    删除...    编辑...

## 设置用户参数

如需根据从站设备要求额外配置一些EtherNet/IP总线通信参数，可通过此选项进行配置（此操作需要一定程度的CIP协议基础），大多情况下无需配置。配置完成后，从站设备每次通讯启动或者重启时，都会把这些配置的用户参数向主站发送一次。

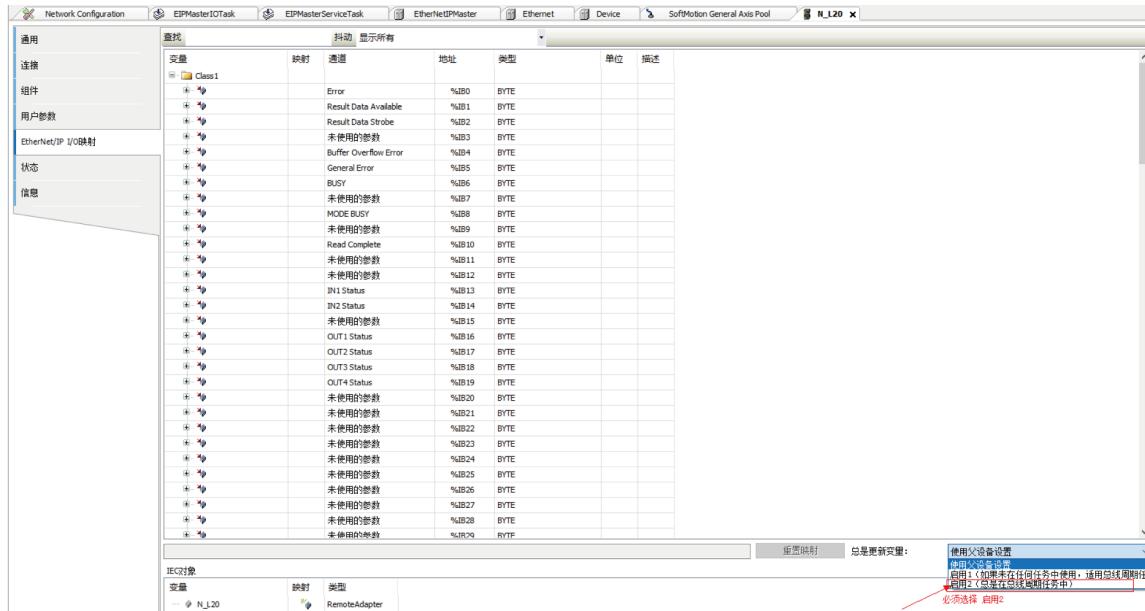


- 名称：参数的名称。
- 类别 (Class)：网络中所有可访问的对象类别都有一个唯一的整数值标识号。如图，类别#5, #7。
- 实例 (Instance)：一个物体的具体的和真实的（物理的）出现。例如：新建新西兰是对象类国家的一个实例，如图Instance#1和Instance#2。（CIP实例ID值零为0，用来表示与类内的特定实例的引用。）
- 属性 (Attribute)：对物体的外部可见特征或特征的描述。通常属性提供状态信息或控制对象。如图 Attribut#1和Attribut#2。



## 在程序中IO变量映射

变量的EtherNet/IP I/O映射界面如下图所示，通过地址设置方式可以设置输入、输出变量。直接选中地址选项进行编辑，可修改系统自动分配的地址，从而实现与程序中变量的映射。



**注意**

“一直更新变量”保持默认值“使能2（一直在总线循环任务重）”不变。

## 禁用/启用EtherNet/IP远程从站

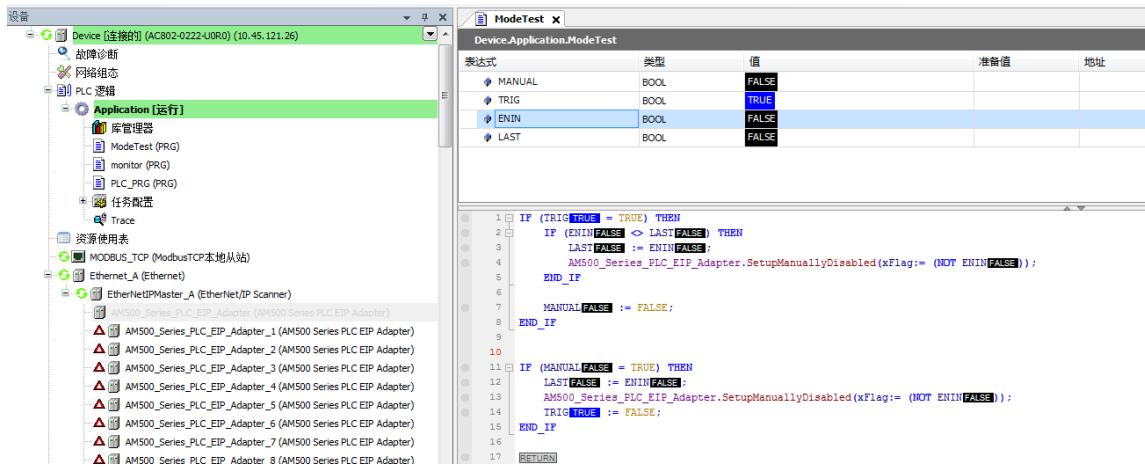


**注意**

InoProShop V1.8.1.0及以上版本支持该功能。

在应用编程时支持禁用或启用EtherNet/IP远程从站，指令格式为“SetupManuallyDisabled(xFlag:= )”，属于FC方法调用，在程序中调用时需要引用从站对象和指定特定从站才能执行。

- 当xFlag输入TRUE时，表示禁用EtherNet/IP远程从站。
- 当xFlag输入FALSE时，表示启用EtherNet/IP远程从站。



### 注意

如果启用EtherNet/IP远程从站时需要满足QuickConnect 500ms要求，那么务必满足以下4个条件：

- 从站启动就绪时间应低于350ms。
- 单个从站配置1条连接。
- EIPMasterIOTask任务周期不超过50ms。
- 同一时段，建议最多切换10个从站的启用/禁用状态。

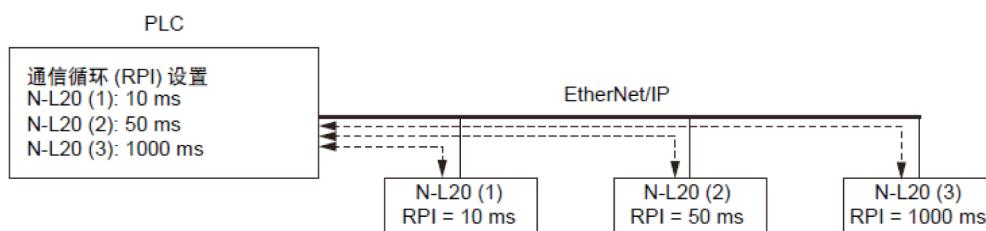
#### 4.11.4 PLC 作为EtherNet-IP 主站的配置例程

##### EtherNet/IP主站工程一般配置步骤

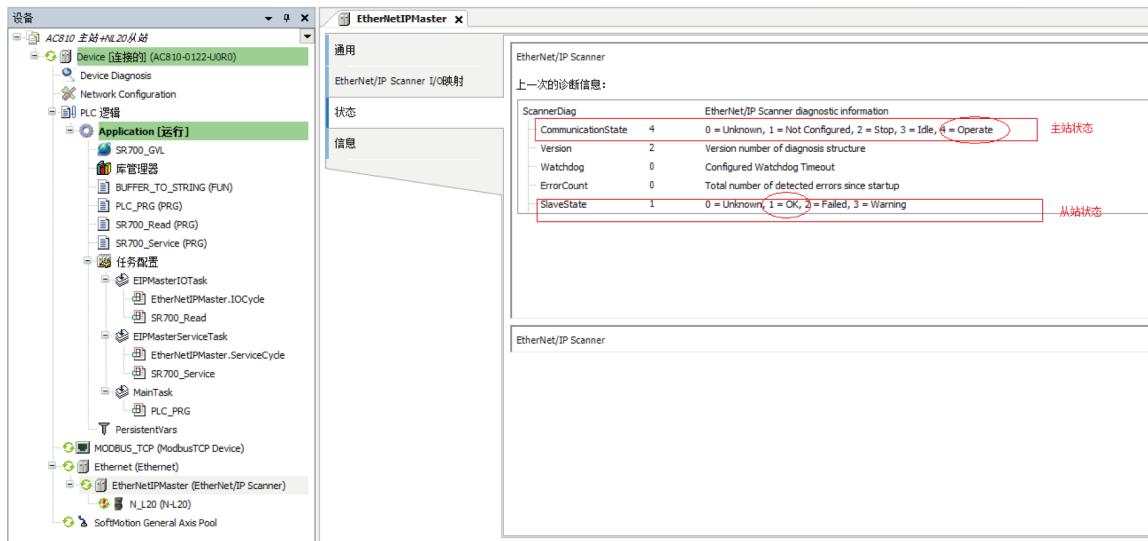
- 新建工程，在“Network Configuration”组态配置中勾选主站。
- 导入第三方eds文件，将EtherNet/IP从站加入组网工程。
- 设置EtherNet通用设置的IP地址，以及从站通用界面的IP地址，确保均在一个局域网内。
- 添加从站默认预定义连接/标签，配置 RPI、任务周期以及连接相关参数。
- 在远程从站EtherNet/IP I/O映射中进行参数映射。
- 根据需求使用编写用户POU程序。

##### EtherNet/IP主站循环数据通讯工程实例

在循环通信，可根据发送和接收数据的优先级设置RPI（通信周期），以发送和接收整体通信负载调整后的数据，如下图所示。



本实例工程以AC810为EtherNet/IP主站，基恩士NL-20+SR700扫码器为EtherNet/IP从站。按照上述设置步骤，配置完成后进行默认连接，启动运行程序。设备树状态全部显示为绿色时，表示启动通讯连接成功。同时，在主站状态中可以查看CommunicationState=4，为OP状态，从站SlaveState=1，为OK状态，ErrorCount=0。如下图所示。

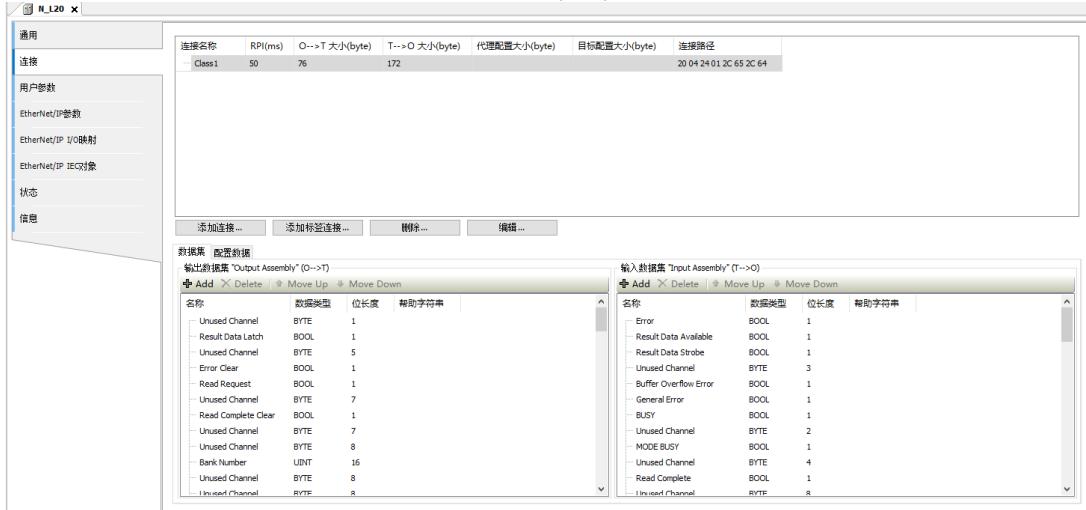


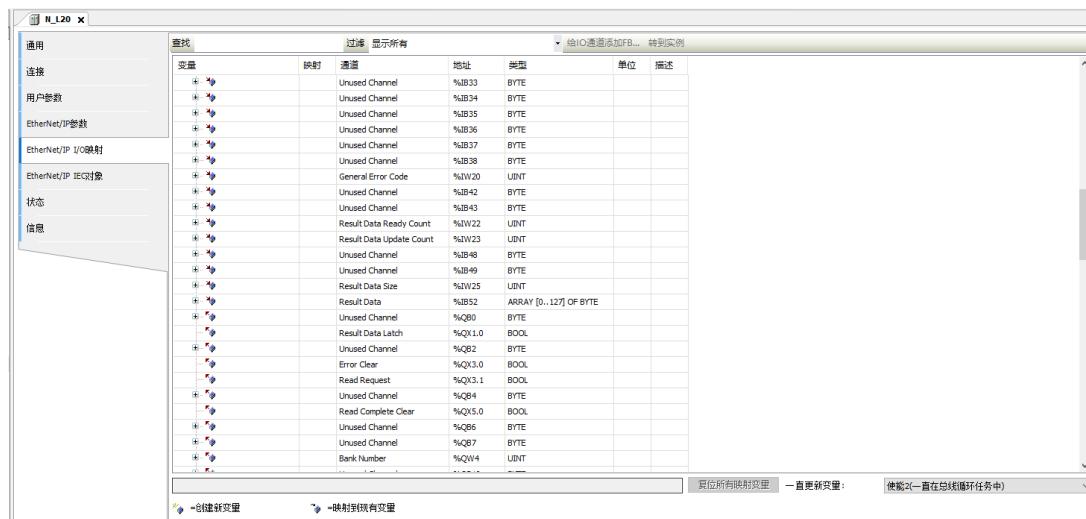
## 说明

在用户程序中判断主从站是否正常通讯，可通过EtherNetIPMaster.eState =6（RUNNING）进行状态判断，从站通过具体实例化的从站名称N\_L20.eState = 8（RUNNING）进行状态判断。

- 基恩士NL20和扫码模块循环通讯实例测试步骤如下：

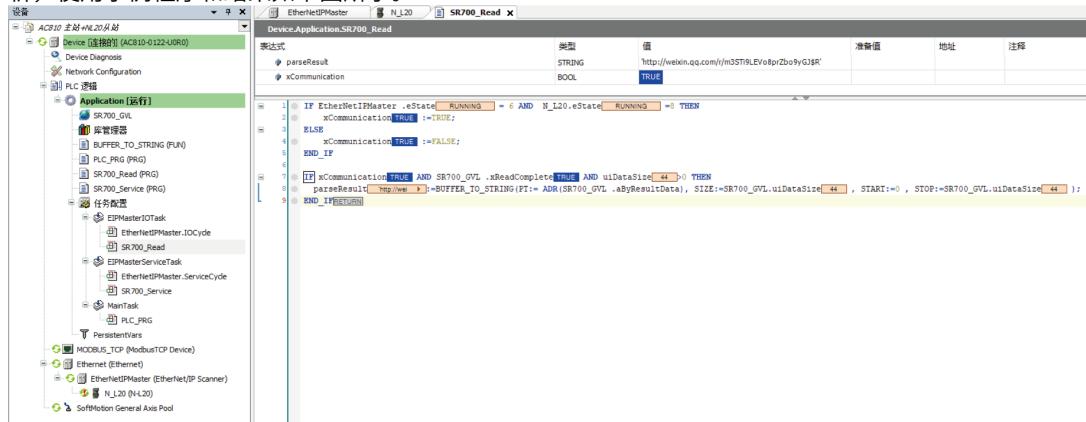
- 映射从站模块的操作变量，定义全局变量与EtherNet/IP I/O映射相关联，如下图所示。





2. 读取条码时, 将在“读取数据”中写入数据。此时, “Read Complete” 变为TRUE(1)。
3. 确认“Read Complete”为TRUE (1) 后, 将“Read Complete Clear”设为TRUE (1)。
4. 将“Read Complete Clear”设为TRUE (1) 时, “Read Complete” 变为FALSE (0)。
5. 确认“Read Complete”为FALSE (0) 后, 将“Read Complete Clear”设为FALSE (0)。

用户程序中在每个循环周期判断通讯状态, 当通讯连接状态正常, 传感器扫码完成后, xReadComplete标志置TRUE, 返回扫描数据不为0, 通过BUFFER\_TO\_STRING对数据结果进行解析, 使用示例程序和结果如下图所示。

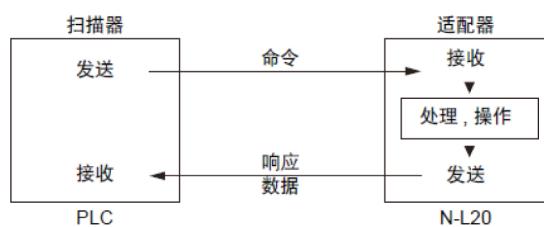


扫码器扫出的二维码字符串结果如图。

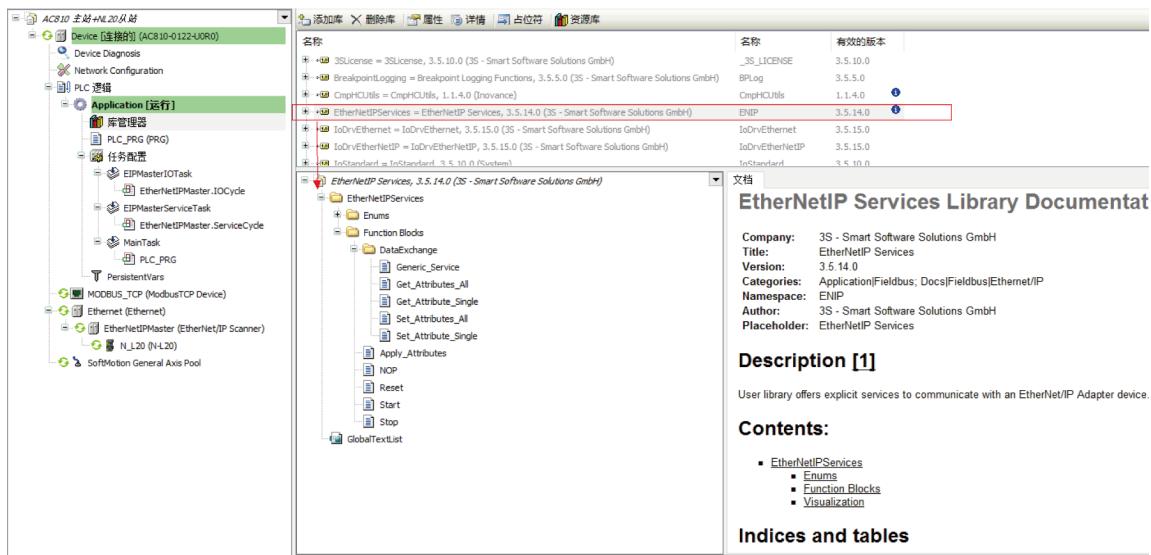


## EtherNet/IP主站服务数据通讯工程实例

在服务信息通信中, 通过命令/响应控制时序, 如下图所示。



根据服务数据通讯发送命名格式，通常需要指定Service code、Class ID、Instance、Attribute ID、Service data。在应用程序中可以通过EtherNet/IP服务功能块库EtherNetIPService来实现，如下图所示。



该功能块库提供了CIP通讯协议大部分公用的服务（CIP Common Services），服务ID和名称如下表所示。

表4-1 表 CIP公用服务ID和名称

ID	名称
00	Reserved for future use
01	Get_Attributes_All
02	Set_Attributes_All Request
03	Get_Attribute_List
04	Set_Attribute_List
05	Reset
06	Start
07	Stop
08	Create
09	Delete
0A	Multiple Service Packet
0B-0C	Reserved for future use
0D	Apply_Attributes
0E	Get_Attribute_Single
0F	Reserved for future use
10	Set_Attribute_Single
11	Find_Next_Object_Instance
12-13	Reserved for future use
14	Error Response (used by DeviceNet only)

通过Get\_Attribute\_Single和Generic\_Service两个功能块，可以获取用户程序中的属性和服务数据。

- Get\_Attribute\_Single

基恩士 NL-20模块手册上提供了特殊的属性 ID，来获取当前读码器的状态，如下表所示。

实例ID	属性ID	名称	相应参数	
			数量	说明
1 (0x01)	100 (0x64)	Read Status	UINT	bit0 : Error bit1 : Result Data Available bit2 : Result Data Strobe bit3 至5 : Reserved bit6 : Buffer Overflow Error bit7 : General Error bit8 : BUSY bit9 至10 : Reserved bit11 : MODE BUSY bit12 : ERR BUSY bit14 至15: Reserved
				UINT
				bit0: Read Complete
				UINT
1 (0x01)	108 (0x6C)	IN/OUT Status	UINT	bit0 : IN1 bit1 : IN2 bit2 至3 : Reserved bit4 : OUT1 bit5 : OUT2 bit6 : OUT3 bit7 : OUT4 bit8 至15 : Reserved
				UINT
				Result Data Ready Count
				UINT
				General Error Code
				UINT
111 (0x6F)	110 (0x6E)	General Error Code	UINT	General Error Code
				UINT
128 (0x80)		Result Data Ready Count	UINT	Result Data Ready Count
				UINT
129 (0x81)		Result Data Update Count	UINT	Result Data Update Count
				UINT

利用功能块Get\_Attribute\_Single，可以实现上表中的相应参数数据，如下图所示。



通过实例化Get\_Attribute\_Single功能块，获取返回结果次数Result Data Count（属性为0x6E）。实例代码如下。其中AutoID Communication Unit Object =16#69，不是EtherNet/IP 标准内的目标，而是KEYENCE 开发的使N-L20更加便于操作的目标。

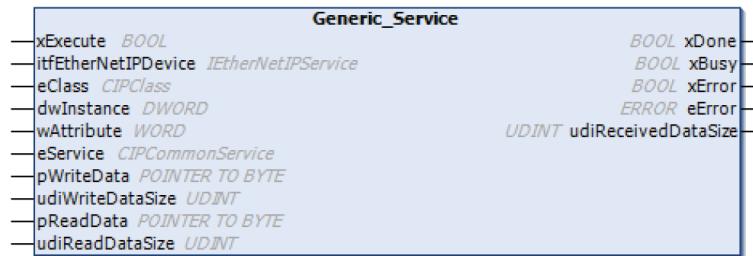
- Generic\_Service

除了公用的服务，还有一些EtherNet/IP设备厂商提供的特殊服务类型，例如基恩士NL-20模块，提供了如下服务代码用于获取设备特殊的交互数据。

实例ID	服务代码	服务数据		名称	说明
		数据类型：数据			
1 (0x01)	14 (0x0E)	-	Get_Attribute _Single	获取属性的一个项目。	
	16 (0x10)	-		获取属性的一个项目。	
	75 (0x4B)	UINT: Bank Number	Read Start	开始读取。	
	76 (0x4C)	-	Read Stop	停止读取。	
	83 (0x53)	-	Error Clear	清除错误。	
	85 (0x55)	UINT: Result Data Size UINT: Offset	Get Result Data	获取读取数据。 响应数据 UINT : 结果数据大小 UINT : 缓存到N-L20的剩余数据大小 BYTE[] : 结果数据	
	86 (0x56)	-	Sequence Reset	清除以下信息： Result Data Ready Count Result Data Update Count Main unit statistical information Buffering data Sequence bit	
	90 (0x5A)	-	Read Status Clear	读取完成通知和读取失败通知的清除	

-324-

可通过功能块Generic\_Service（功能块如下图），获取以上服务数据。通过提供详细的ClassID、实例ID、服务代码等，获取指定厂商提供的接口数据。



通过服务码85（0x55）来读取扫码器的数据，示例代码如下。

	类别	名称	地址	数据类型	初值	保持	常量	网络公开	注释	特性
1	VAR	genericService		ENIP.Generic_Service		<input type="checkbox"/>	<input type="checkbox"/>	默认		
2	VAR	xGenericService_Trigger		BOOL		<input type="checkbox"/>	<input type="checkbox"/>	默认		
3	VAR	dUIntWriteData		ARRAY[1..256] OF UINT		<input type="checkbox"/>	<input type="checkbox"/>	默认		
4	VAR	dUIntReadData		ARRAY[1..256] OF UINT		<input type="checkbox"/>	<input type="checkbox"/>	默认		

```

1  dUIntWriteData[0] := SIZEOF(dUIntReadData);
2  dUIntWriteData[1] := 0;
3
4  genericService(
5      xExecute:= xGenericService_Trigger,
6      xDone=> ,
7      xBusy=> ,
8      xError=> ,
9      itfEtherNetIPDevice:= NL_20,
10     eClass:= 16#69,
11     dwInstance:= 1,
12     eError=> ,
13     wAttribute:= 110,
14     eService:= 16#55,           //ENIP.CIPCommonService.SET_ATTRIBUTE_SINGLE
15     pWriteData:= ADR(dUIntWriteData),
16     udiWriteDataSize:= SIZEOF(dUIntWriteData),
17     pReadData:= ADR(dUIntReadData),
18     udiReadDataSize:= SIZEOF(dUIntReadData),
19     udiReceivedDataSize=> );

```

#### 4.11.5 PLC 作为EtherNet-IP 从站的配置

汇川中型PLC支持EtherNet/IP本地从站功能，支持Class1类型的实例ID连接和标签连接响应，同时可支持Class3/Ucmm类型的服务消息标签通讯（最大32个），从站具体配置说明如下。

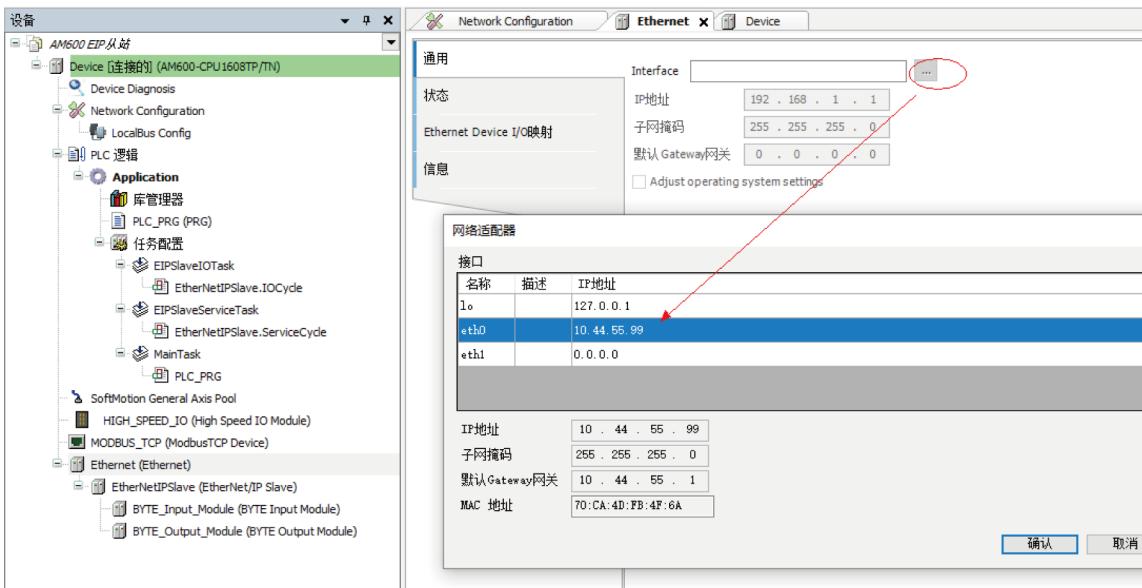
##### 勾选EtherNet/IP 从站

鼠标单击主站设备，在弹出的界面中，勾选“EtherNet/IP从站”。



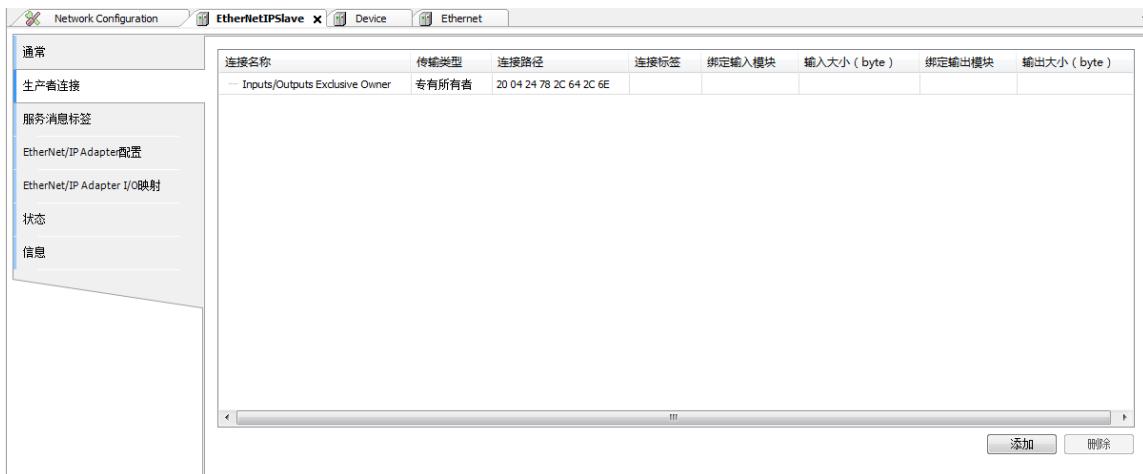
## 设置从站IP地址

后台登录PLC网关，选择Ethernet模块，双击进入通用界面，手动设置从站的IP地址。该IP地址设置是实际连接到EtherNet/IP主站设备的IP地址。

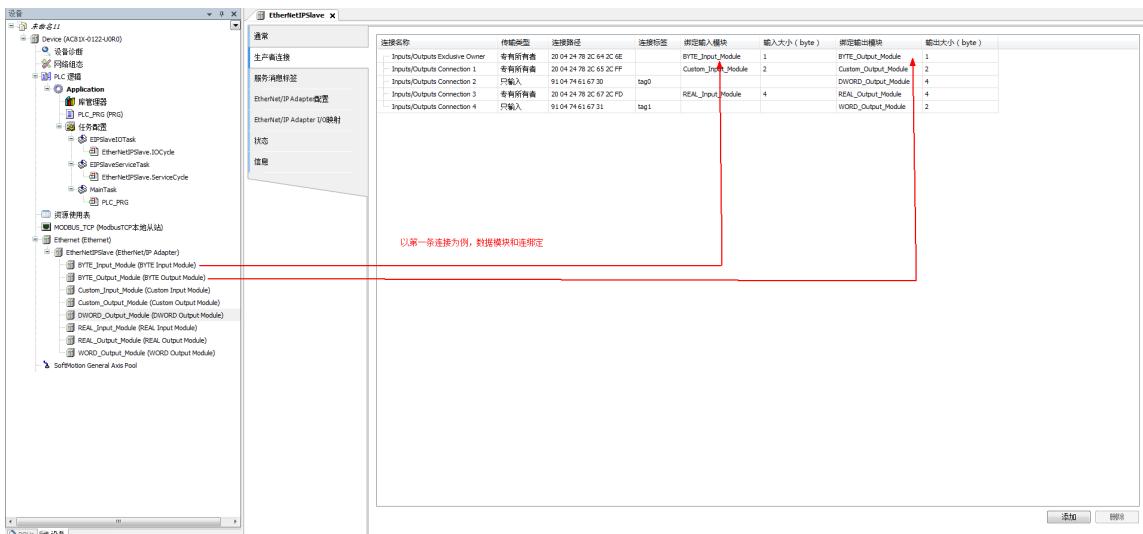


## 设置从站生产者连接

EtherNet/IP从站支持实例ID路径和标签路径连接，且可以响应多个主站（扫描器）设备发起的连接请求。如下图所示，“生产者连接”界面默认只有一条实例ID路径连接，通过“添加”按钮添加多条连接。连接路径中“20 04 24 78 2C 64 2C 6E”，配置实例ID为16#78，输入数据集实例ID为16#64，输出数据集实例ID为16#6E。



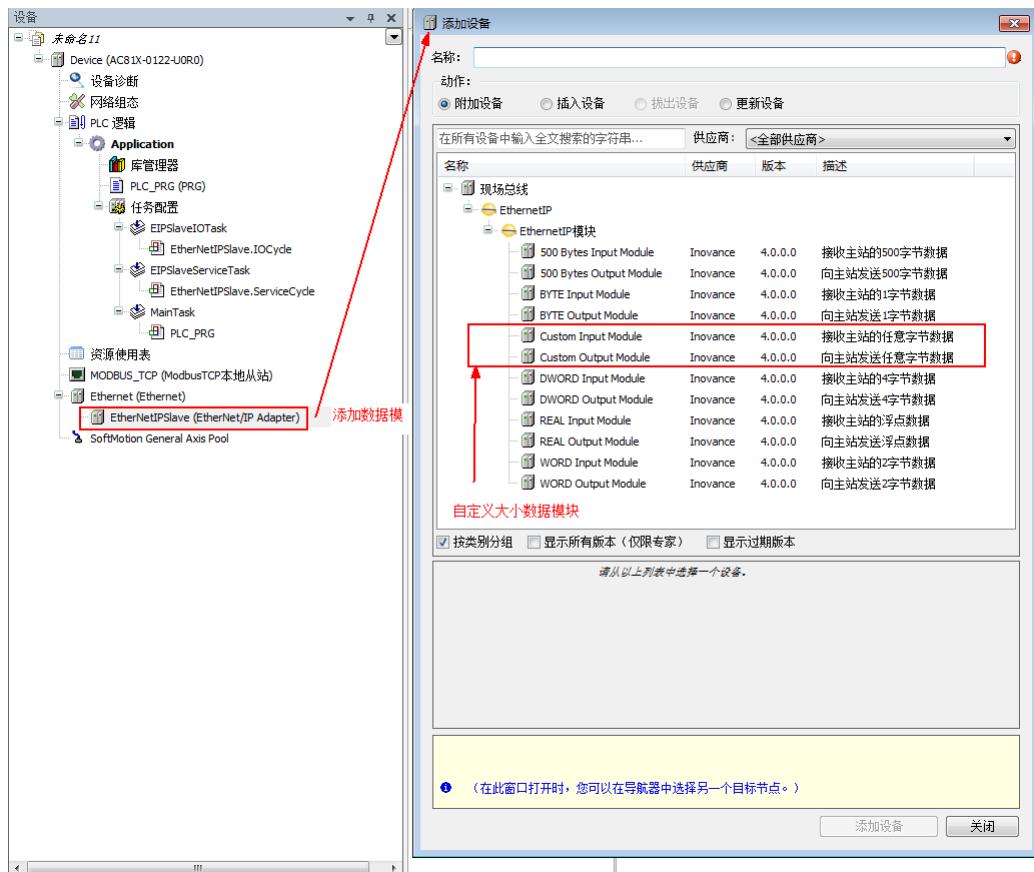
每条连接必须要和数据输入输出模块进行绑定，否则不生效。在添加完数据模块后，该界面自动刷新，可以显示绑定的数据模块名称以及数据总大小，如下图所示：



如果要设置生产者标签连接，可在连接路径中手动输入生产者标签名称，生产者标签默认数据为只输出，因此只可以绑定输出连接。

## 添加输入输出模块

选择EtherNetIPSlave，右键添加设备，添加从站输入输出模块。用户可以根据数据参数传输需要自由组合模块。在模块中 Custom Input Module 和 Custom Output Module 为自定义大小输入输出模块。其他数据模块为固定字节大小的输入输出模块，用户可以添加多个数据模块自由组合数据总大小。

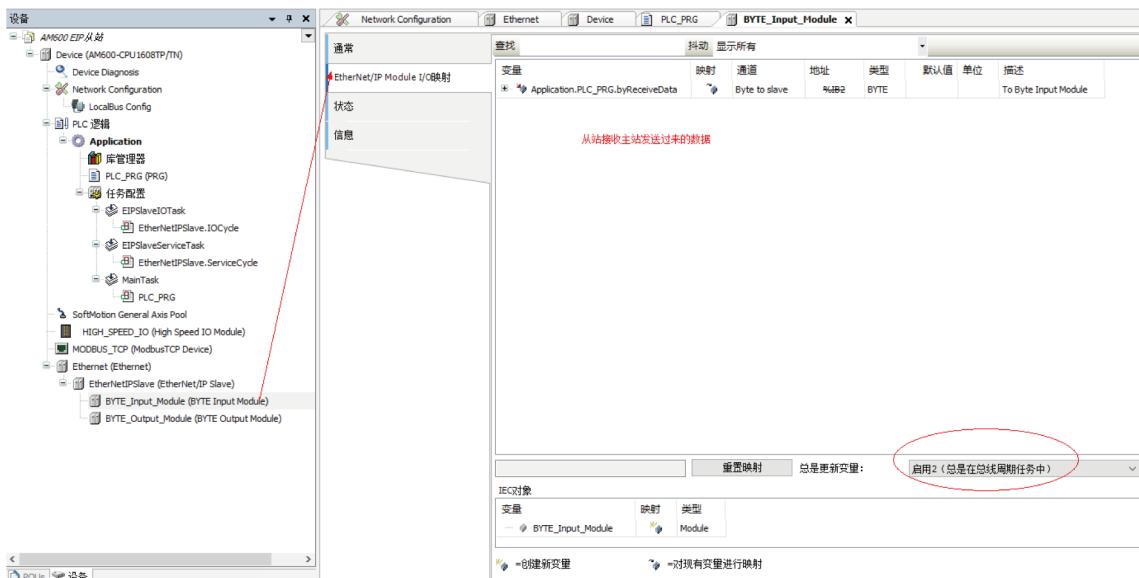


如下图自定义输入模块（Custom\_Input\_Module）配置界面所示，在该界面中可以自由绑定生产者连接和修改数据大小字节数。在输入数据集中可以自定义参数名，数据类型，灵活创建用户结构体参数。

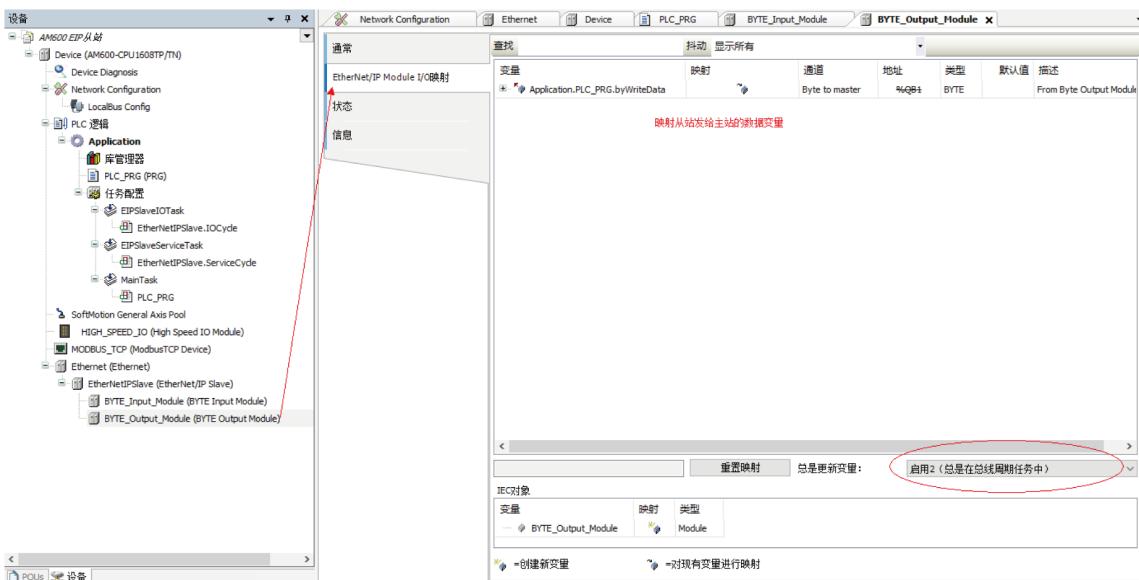


## 配置数据模块IO映射

以Byte Input Module和Byte Output Module数据模块为例，PLC自动分配数据类型为BYTE的输入输出映射，当PLC的EtherNet/IP从站收到数据，则在InputData中映射变量刷新，如下图byReceiveData，数据类型为Byte。



当PLC的EtherNet/IP从站发送数据，则在OutputData中映射变量将数据写入，如下图byWriteData，数据类型为Byte。



## 说明

勾选从站配置后，后台会自动生成任务配置，建议设置从站工程中的EIPMasterIOTask任务周期与主站的RPI时间一致，以保证数据传输同步。

## 导出从站配置的eds文件

如图在EtherNet/IP从站的通用设置中可以点击“导出定制EDS文件”导出上述添加配置的各个连接和输入输出数据集以及从站设备信息。

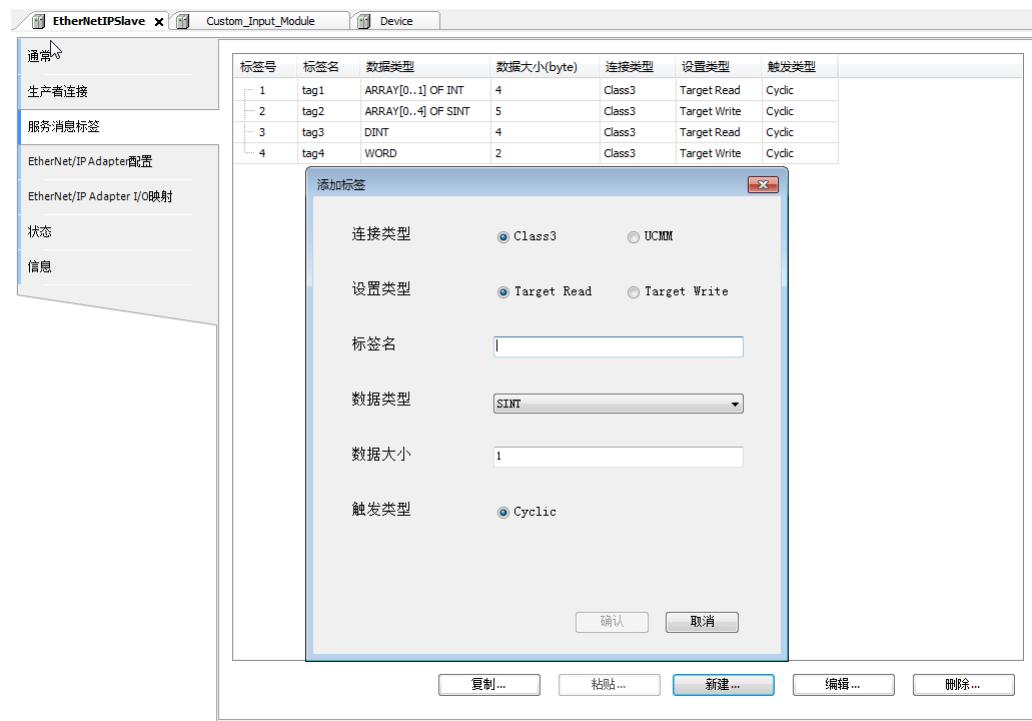


## 说明

当导出界面为灰色不可选时，原因为从站数据配置没有刷新，可以先关闭从站窗口再次打开即可。

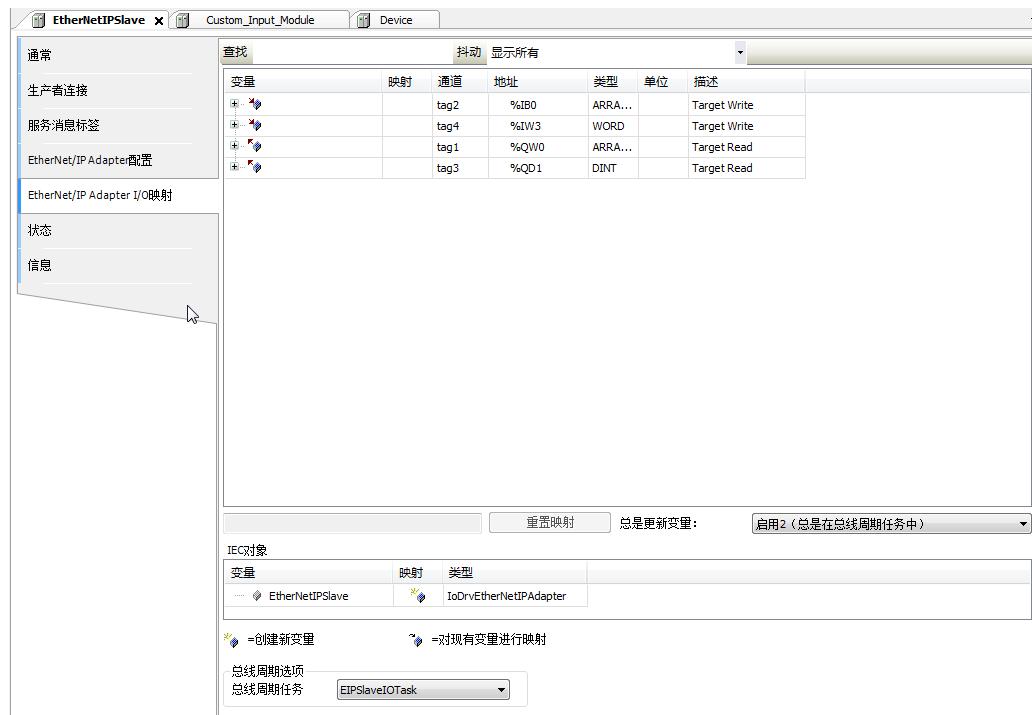
## 服务消息标签配置

该界面实现Class3/Ucmm类型的显示消息通讯配置，新建一条显示消息标签如下图所示。



其中，设置类型为Target Read 可以响应EtherNet/IP主站（扫描器）设备发起的读数据请求，服务码为4C。设置类型为Target Write 可以响应EtherNet/IP主站（扫描器）设备发起的写数据请求，服务码为4D。标签名，数据类型以及数据大小设定需和发起设备的参数一致，否则会产生通讯故障。

显示消息通讯读写数据的更新在EtherNet/IP Adapter I/O映射中循环刷新，任务刷新周期为EIPSlaveServiceTask的设定周期。



## 全局变量表EIP 配置

### 背景信息

标签变量的访问支持结构体成员访问和结构体整体访问（V1.8.1.0及以上版本支持），结构体成员只支持基础数据类型，变量成员规格如下表所示；支持嵌套访问，嵌套层数支持16层（包含两个前缀，前缀支持“application.gvl.” 和 “application\_gvl\_\_” ，优先选择 “application\_gvl\_\_” ），即实际嵌套层数支持14层。

数据类型名称	数据类型编码	数据类型描述	支持数组维数	结构体成员支持的数据类型
BOOL	C1	TRUE/FALSE	3	<input type="checkbox"/>
SINT	C2	8位整形	3	<input type="checkbox"/>
INT	C3	16位整形	3	<input type="checkbox"/>
DINT	C4	32位整形	3	<input type="checkbox"/>
LINT	C5	64位整形	3	<input type="checkbox"/>
USINT	C6	无符号8位	3	<input type="checkbox"/>
UINT	C7	无符号16位	3	<input type="checkbox"/>
UDINT	C8	无符号32位	3	<input type="checkbox"/>
ULINT	C9	无符号64位	3	<input type="checkbox"/>
REAL	CA	浮点32位	3	<input type="checkbox"/>
LREAL	CB	浮点64位	3	<input type="checkbox"/>
STRING	D0	字符串	0	<input type="checkbox"/>
BYTE	D1	8位	3	<input type="checkbox"/>
WORD	D2	16位	3	<input type="checkbox"/>
DWORD	D3	32位	3	<input type="checkbox"/>
LWORD	D4	64位	3	<input type="checkbox"/>
ARRAY	A3	数组	3	<input type="checkbox"/>
STRUCT	A0 · 02 · CRC · CRC	结构体	3	<input type="checkbox"/>

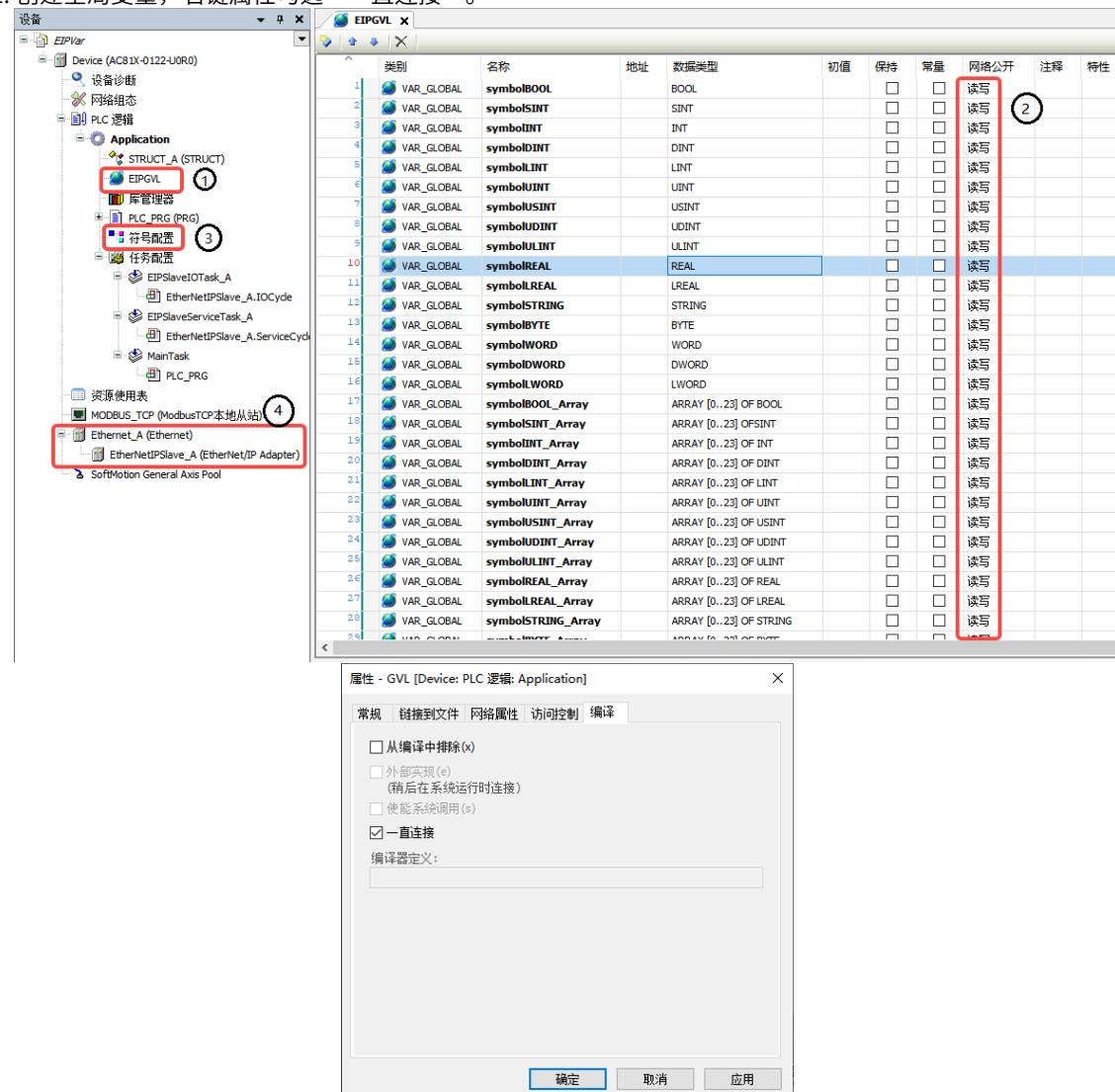
同时支持从站服务消息标签配置和全局变量勾选两种方式，区别如下表所示。

项目	服务消息标签配置	全局变量表
访问格式	标签名	Application+全局变量表名称+变量名
导入导出	不支持	支持
数据类型	不支持BOOL, STRING, DUT和多维数组	见表格
读写访问	不支持	支持
单次访问最大数据量	1400个字节	1000个字节
中文标签	不支持	支持（符号配置表，设置“字符编码使用UTF-8”）
数组索引	支持	支持
多标签服务	支持	支持

应用案例场景：HMI、EIP上位机API、PLC和IM/MES。

### 操作步骤

#### 1. 创建全局变量，右键属性勾选“一直连接”。



#### 2. 设置“网络公开”属性。

- 读写：标签变量可读可写。

- 只读：标签变量只能读访问，写访问返回错误。
- 只写：标签变量只能写访问，读访问返回错误。

3. 添加符号配置。

- 方式1：编译或者登录下载会自动添加。
- 方式2：手动添加。

4. 网络组态勾选“EtherNet/IP从站”，设置EtherNet\_A/EtherNet\_B网口IP。

5. (可选) 如标签变量中包含中文字符，在“符号配置”页签中单击“设置”，选择“字符编码使用UTF-8”编码格式。



#### 4.11.6 PLC 作为EtherNet-IP 从站的配置例程

##### EtherNet/IP从站工程一般配置步骤

- 新建工程，在“Network Configuration”组态配置中勾选从站。
- 根据应用需求配置生产者连接（Class1）或者显示服务消息连接（class3/Ucmm）。
- 如从站配置为生产者连接（实例ID/标签），添加输入输出模块并绑定到对应连接。
- 在从站输入输出模块或者从站I/O映射中进行变量参数映射。
- 根据需求编写用户POU程序。

按照上述EtherNet/IP从站配置步骤创建从站例程，本文以AM600工程和欧姆龙NJ501 EtherNet/IP主站工程进行通讯连接测试。

##### 从站工程与欧姆龙NJ501主站通讯

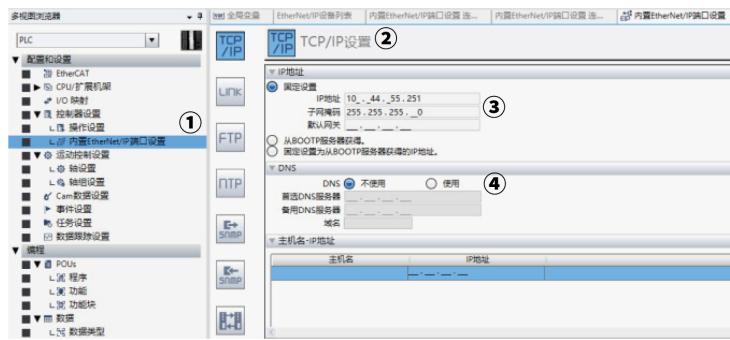
测试欧姆龙PLC EtherNet/IP为主站和汇川AM600从站工程进行通讯。

- NJ501 EtherNet/IP主站工程配置

- 新建欧姆龙NJ501标准工程。



- 配置和设置中，选择内置EtherNet/IP端口设置，TCP/IP设置中设置主站EtherNet/IP通讯端口IP地址，确保该IP地址和汇川AM600从站属于同一网段。

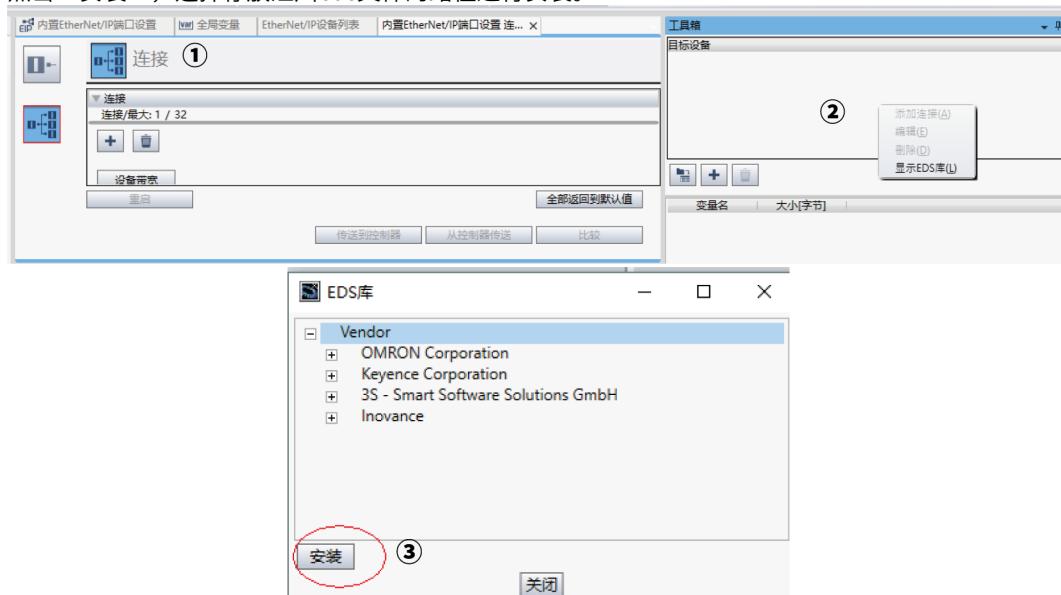


### 3. 新建全局变量。

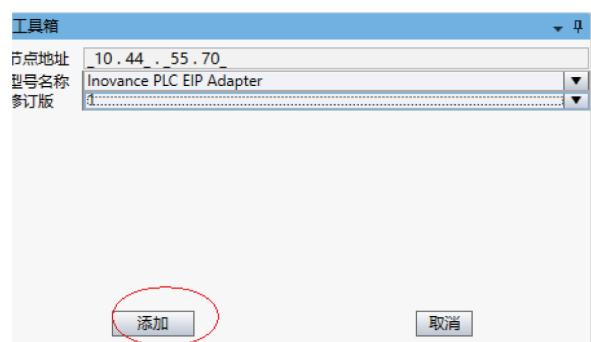


### 4. 设置连接路径。

- a. 导入汇川leds文件。在连接界面工具箱中，右键->“显示EDS库（L）”，在弹出的EDS库目录列表中点击“安装”，选择存放汇川leds文件的路径进行安装。



- b. 添加从站设备。安装完成后，在工具箱列表下方选择“添加目标设备”，在型号名称下拉列表中即可找到刚才导入EDS库中的汇川EtherNet/IP从站设备名，选中即可。再设置连接到汇川EtherNet/IP从站设备的节点IP地址和修订版本（该IP地址应和欧姆龙PLC主站在同一网段），最后点击“添加”即可。



- c. 设置连接。如下图所示，设置目标设备、连接类型和输入输出变量等参数。

连接										
连接名称	连接设备	连接I/O类型	输入/输出	目标变量	大小[字节]	起始变量	大小[字节]	连接类型	RPI(毫秒)	超时值
10.44.55.70 Inovance PLC E	default_001	Inputs/Outputs	输入	110	1	RecvVar	1	Point to Point	10	RPI x 4
			输出	100	1	SendVar	1	Point to Point	4	

## 说明

标变量输入输出110和100分别为汇川从站数据集输出和输入的实例ID。

连接名称	传输类型	连接路径	连接标签	绑定输入模块	输入大小 (byte)	绑定输出模块	输出大小 (byte)
... Inputs/Outputs Exclusive Owner	专有所有者	20 04 24 78 2C 64 2C 6E		BYTE_Input_Module	1	BYTE_Output_Module	1

输入模块实例ID                           输出模块实例ID

## EtherNet/IP主从通讯功能测试

通过以上设置后，将欧姆龙PLC工程进行编译，同步后运行，分别查看状态，确认主站和从站通讯正常。如下图所示。

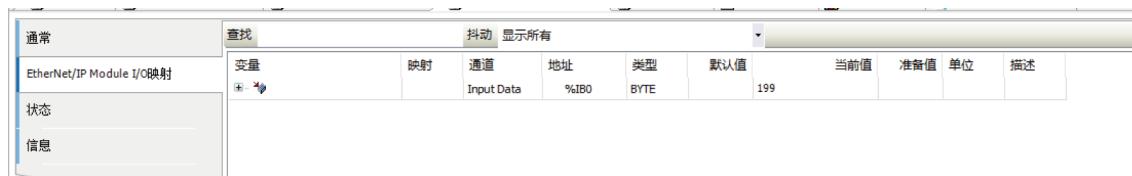
The screenshot shows two main windows. On the left is the '内置EtherNet/IP端口设置' (Built-in EtherNet/IP Port Settings) window, which displays network status, connection status, and various configuration parameters for the master station. On the right is the 'AM600 EtherNet/IP 从站' (AM600 EtherNet/IP Slave) configuration window, which shows the device tree with nodes like 'Device连接的' (AM600-CPU1608TP/TN), 'Network Configuration', 'Localbus Config', and the 'Application' section containing tasks for data exchange.

修改NJ501 EtherNet/IP标签通讯SendVar的值，查看AM600 EtherNet/IP从站接收变量的变化，如果能跟随修改值变化，表示通讯测试正常。如下图。

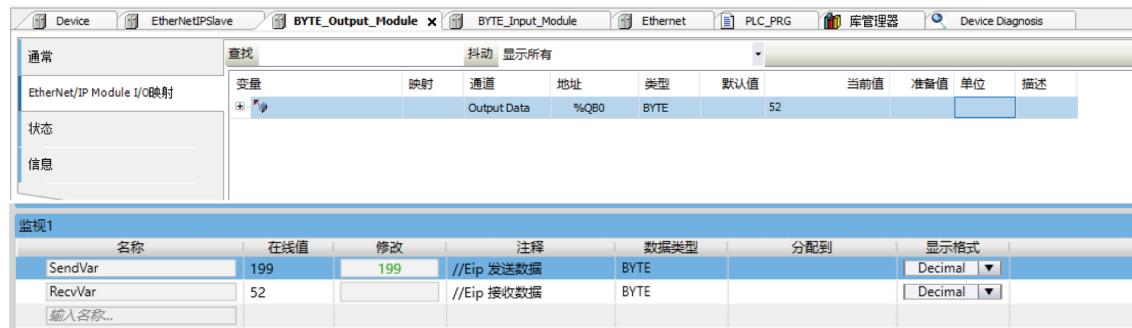
This screenshot shows the NJ501 monitor interface. The top part is a table for variable monitoring, where 'SendVar' is set to 101 and 'RecvVar' is set to 0. Below it is a search bar and a table for I/O mapping, showing 'EtherNet/IP Module I/O映射' with 'SendVar' mapped to 'Input Data' at address '%IB0'. The bottom part shows the status and information sections of the monitor.

SendVar在线修改为199，对应AM600映射变量也更新为199。

This screenshot shows the NJ501 monitor after the value was changed. The 'SendVar' row now shows '修改' (Modified) with the value '199', indicating the update was successful. The 'RecvVar' row remains at 0.



修改AM600 EtherNet/IP从站Output变量发送值为52， NJ501 EtherNet/IP通讯RecvVar值也更新为52。



#### 4.11.7 EtherNet-IP通讯状态诊断

##### EtherNet/IP设备通讯状态诊断

中型PLC EtherNet/IP设备在应用中可以分别做主站、从站或者同时主从，配置完成后在线登录到设备，EtherNet/IP设备树前显示绿色图标即通讯成功。在程序中的状态诊断可调用协议库中变量，EtherNet/IP主站通讯状态可以通过程序EtherNetIPMaster.eState获取，从站通讯状态可以通过EtherNetIPSlave.eState获取，远程从站的通讯状态可以通过“远程从站名.eState”（例如AM400\_Series\_PLC\_EIP\_Adapter.eState）。

可参考以下示例程序：

```

VAR
    eSlaveState : IodrvethernetIPAdapter.AdapterState;
    eMasterState : Iodrvethernetip.ScannerState;
    eRemoteSlave: Iodrvethernetip.AdapterState;
END_VAR

eSlaveState := EtherNetIPSlave.eState; //本地从站状态
eMasterState := EtherNetIPMaster.eState;//本地主站状态
eRemoteSlave := AM400_Series_PLC_EIP_Adapter.eState;//远程从站状态

//判断本地从站状态是否正常
IF eSlaveState = iodrvethernetIPAdapter.AdapterState.RUNNING THEN
    //TODO ; 添加用户逻辑
END_IF

//判断本地主站状态是否正常
IF eMasterState = iodrvethernetip.ScannerState.RUNNING THEN
    //TODO;添加用户逻辑
END_IF

//判断远程从站状态是否正常
IF eRemoteSlave = IoDrvEthernetip.AdapterState.RUNNING THEN
    //TODO;添加用户逻辑
END_IF

```

## EtherNet/IP日志错误诊断码

EtherNet/IP作为主站时，可以为主站添加从站（远程从站）。在主站和从站的配置界面中，都有一个“设备诊断”界面。主站诊断信息主要是用于标示配置项出现故障，不包含具体故障原因，因此“设备诊断”界面没有故障码。

当通讯发送连接错误时，可以通过后台设备->日志查找对应EtherNet/IP组件设备诊断信息。EtherNet/IP作为主站或者从站时其诊断码及诊断信息都是一致的，见下表EtherNet/IP诊断码。

诊断信息ID	状态码	诊断信息说明
SUCCESS	0	通讯成功
DUPLICATE_FWD_OPEN	16#100	重复的连接请求
TRANSPORTCLASSTRIGGER_NOT_SUPPORTED	16#103	传输类型和触发类型不支持
OWNERSHIP_CONFLICT	16#106	主站冲突
TARGET_CONNECTION_NOT_FOUND	16#107	目标从站连接未找到
INVALID_NETWORK_CONNECTION_PARAMETER	16#108	无效的网络连接参数
INVALID_CONNECTION_SIZE	16#109	无效的配置大小/T->O 大小/O->T大小
TARGET_FOR_CONNECTION_NOT_CONFIGURED	16#110	目标从站连接未配置
RPI_NOT_SUPPORTED	16#111	RPI不支持
RPI_NOT_ACCEPTABLE	16#112	RPI设置不通过
OUT_OF_CONNECTIONS	16#113	连接类型超出范围

诊断信息ID	状态码	诊断信息说明
VENDORID_OR_PRODUCT_CODE_MISSMATCH	16#114	厂商/产品ID不匹配
PRODUCT_TYPE_MISSMATCH	16#115	产品类型不匹配
REVISION_MISSMATCH	16#116	软件版本不匹配
INVALID_PATH	16#117	无效的路径
INVALID_CONFIGURATION_PATH	16#118	无效的配置路径
NON_LISTEN_ONLY_CONNECTION_NOT_OPEND	16#119	无non_Listen连接路径
TARGET_OBJECT_OUT_OF_CONNECTIONS	16#11A	超出目标设备最大连接个数
RPI_SMALLER_THAN_PRODUCTION_INHIBIT_TIME	16#11B	RPI设置小于生产者禁止时间
TRANSPORT_CLASS_NOT_SUPPORTED	16#11C	传输类型或者触发参数不支持
PRODUCTION_TRIGGER_NOT_SUPPORTED	16#11D	触发类型不支持
DIRECTION_NOT_SUPPORTED	16#11E	传输方向设置不支持
INVALID_OT_FIXVAR_VALUE	16#11F	无效的O->T网络连接参数固定/变化标识
INVALID_TO_FIXVAR_VALUE	16#120	无效的T->O网络连接参数固定/变化标识
INVALID_OT_PRIORITY	16#121	无效的O->T优先级设定
INVALID_TO_PRIORITY	16#122	无效的T->O优先级设定
INVALID_OT_CONNECTION_TYPE	16#123	无效的O->T连接类型
INVALID_TO_CONNECTION_TYPE	16#124	无效的T->O连接类型
INVALID_OT_REDUNDANT_OWNER_FLAG	16#125	无效的冗余所有者标志
INVALID_CONFIGURATION_SIZE	16#126	无效的配置大小
INVALID_OT_SIZE	16#127	无效的O->T大小
INVALID_TO_SIZE	16#128	无效的T->O大小
INVALID_CONFIG_APPL_PATH	16#129	无效的配置路径
INVALID_CONSUMING_APPL_PATH	16#12A	无效的消费者路径
INVALID_PRODUCING_APPL_PATH	16#12B	无效的生产者路径
CONFIG_SYMBOL_DOES_NOT_EXIST	16#12C	配置标签不存在
CONSUMING_SYMBOL_DOES_NOT_EXIST	16#12D	消费者标签不存在
PRODUCING_SYMBOL_DOES_NOT_EXIST	16#12E	生产者标签不存在
INCONSISTENT_APPL_PATH_COMBINATION	16#12F	配置/生产者/消费者路径不一致
INCONSISTENT_CONSUME_DATA_FORMAT	16#130	消费者数据格式不一致
INCONSISTENT_PRODUCE_DATA_FORMAT	16#131	生产者数据格式不一致
NULL_FWDOPEN_NOT_SUPPORTED	16#132	空配置的连接请求不支持
CONNECTION_TIMEOUT_MULTIPLIER_NOT_ACCEPTABLE	16#133	连接超时倍增不通过
MISMATCHED_TO_CONNECTION_SIZE	16#134	连接数据大小不匹配
MISMATCHED_TO_CONNECTION_FIXVAR	16#135	连接参数固定/变化标志不匹配
MISMATCHED_TO_CONNECTION_PRIORITY	16#136	连接参数优先级设定不匹配

诊断信息ID	状态码	诊断信息说明
MISMATCHED_TRANSPORT_CLASS	16#137	传输类型不匹配
CONNECTION_TIMED_OUT	16#203	连接超时
UNCONNECTED_REQUEST_TIMED_OUT	16#204	无连接请求超时
PARAM_ERROR_IN_UNCONNECTED_REQUEST	16#205	无连接请求参数错误
MESSAGE_TOO_LARGE	16#206	消息长度太大
UNCONNECTED_ACK_WITHOUT_REPLY	16#207	无连接应答未收到回复
NO_BUFFER_MEMORY_AVAILABLE	16#301	无可用的缓存空间
NETWORK_BANDWIDTH_NOT_AVAILABLE	16#302	网络无法分配足够的带宽
PORT_NOT_AVAILABLE	16#311	端口不可用
LINK_ADDRESS_NOT_VALID	16#312	端口数据段的链路地址无效
INVALID_SEGMENT_IN_CONNECTION_PATH	16#315	连接路径中存在无效的数据段

## 4.12 Profibus-DP 总线

### 4.12.1 概述

#### 总线简介

PROFIBUS是一种国际化，开放式，不依赖于设备生产商的现场总线标准。广泛适用于制造业自动化，流程工业自动化和楼宇、交通、电力等其它领域自动化，1996年成为欧洲工业标准，1999年成为国际标准，2001年被批准成为中华人民共和国工业自动化领域行业标准的现场总线标准。

PROFIBUS利用了现有国际标准，它的协议结构以国际标准OSI系统互连模型为基础，如图3-82所示，因此符合了开放性和标准化的要求。在这个模型中，每个传输精确处理所定义的任务。第一层物理层定义了物理的传输特性，第二层数据链路层定义了总线的存取协议，第七层应用层定义了应用功能。

PROFIBUS-DP使用了第一层、二层和用户接口，第三层到第七层未加描述，这种流体结构确保了数据传输的快速和有效、直接数据链路映像DDLM (direct data link mapper) 提供了易于进入第二层的服务用户接口，该用户接口规定了用户与系统以及不同设备可调用的应用功能，并详细说明了各种不同PROFIBUS-DP设备的设备行为，还提供了RS-485 传输技术或光纤。



图4-64 PROFIBUS-DP现场总线模型

PROFIBUS-DP用于现场层的高速数据传输，主站周期地读取从站的输入信息并周期地向从站发送输出信息。除周期性用户数据传输外，PROFIBUS-DP还提供智能化现场设备所需的非周期性通讯以进行组态、诊断和报警处理。

#### 4.12.2 Profibus-DP 使用的一般过程

Profibus-DP一般使用流程如下：

1. 设计Profibus-DP硬件网络结构。
2. 在网络组态中激活Profibus-DP总线。激活总线后，会自动添加Profibus-DP主站。
3. 根据硬件结构在网络组态中添加Profibus DP从站及模块。如果是第三方从站，可以在网络组态中通过导入GSD文件导入第三方从站，然后添加第三方从站；如果是AM600从站需要在硬件组态中添加I/O模块。Profibus-DP从站指DP远程设备。
4. 设置合适的主站配置参数、从站配置参数及模块配置参数。一般情况下，从站节点ID自动生成，IO映射根据GSD文件会自动生成，一些特殊的配置需要手动修改。

配置主站参数及从站参数时，主站波特率与从站的波特率是自适应的，所配置的从站的节点ID需要和实际的从站节点ID拨码开关匹配。

### 4.12.3 Profibus-DP 主站配置

#### 主站参数配置

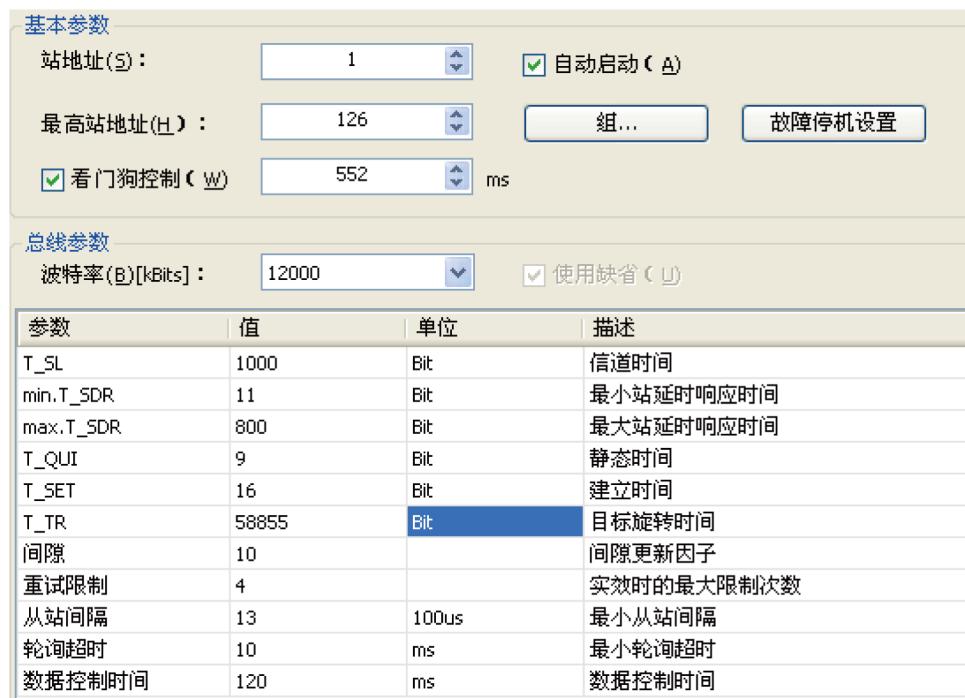


图4-65 Profibus-DP主站配置界面

#### 基本参数

- 节点ID：主站在Profibus-DP网络唯一标示号，默认1，范围1-126，必须是十进制数。
- 最高站地址：令牌帧传递的最大站地址，默认126。
- 看门狗控制：传递给从站的看门狗时间，用来确认从站与主站之间的连接。

#### 总线参数

波特率：总线上用于传输的波特率。单位Kbit/s,可以设置以下的波特率：9.6, 19.2, 45.45, 93.75, 187.5, 500, 1500, 3000, 6000 及 12000；默认值1500。

#### 说明

根据不同的通讯距离及通讯的站点个数，要设置合理的波特率，只有在总线配置文件参数相互匹配时，PROFIBUS子网才能正常工作。因此，只有在熟悉PROFIBUS总线配置文件的参数分配时，才能更改缺省值，建议用户使用默认的总线参数。

#### 故障停机设置

故障停机功能是指当从站或者模块出现故障或者组态不一致时，从站是否停止运行。此设置为故障停机开关，此功能只支持AM600 Profibus-DP从站。

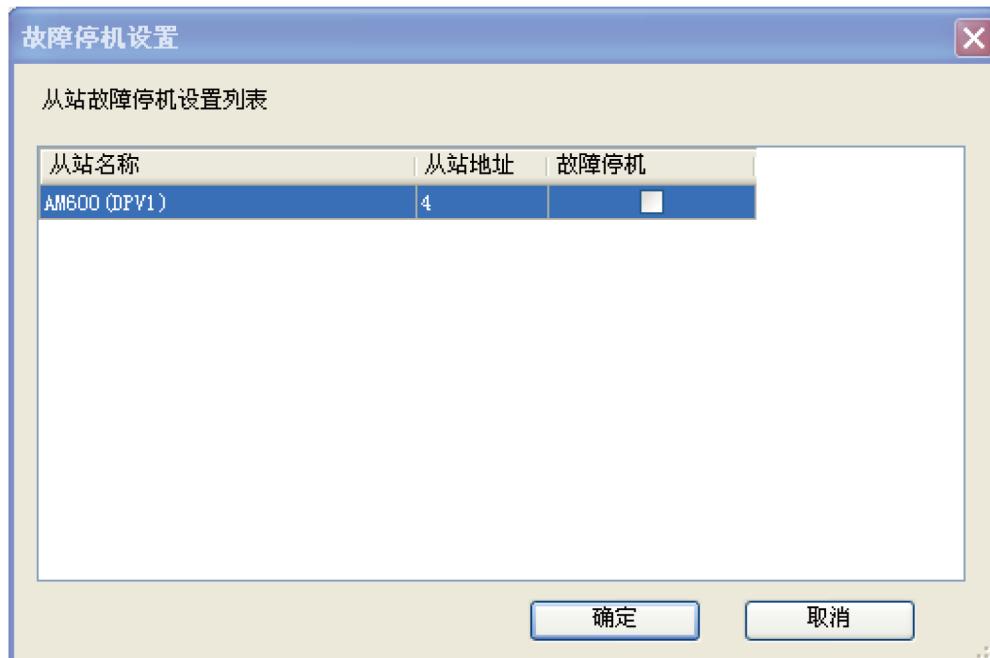


图4-66 故障停机设置

- 从站故障停机设置列表：显示并设置从站故障或者组态不一致时是否停机。  
“故障停机”列控制当从站或者模块出现故障时从站是否停机。如果激活了故障停机功能，如果从站本身出现故障，从站停止运行；如果I/O模块本身的诊断上报功能激活并且模块出现故障，从站停止运行。
- 确定/取消：保存/取消保存故障停机设置。

### Profibus-DP 主站 I/O映射

I/O映射的总体描述和相关对话框的使用请参照I/O映射链接。

#### 状态

DP总线设备或模块的状态配置编辑器可显示状态信息（如：“运行”，“停止”）和内部总线系统的状态。

#### 信息

如果当前设备可用，将会显示下列基本信息：名称、供应商、类型、版本号，模块号、描述。

### 4.12.4 Profibus-DP 从站配置

Profibus-DP从站配置主要是配置从站基本参数及GSD中声明的从站参数。

## 从站参数配置

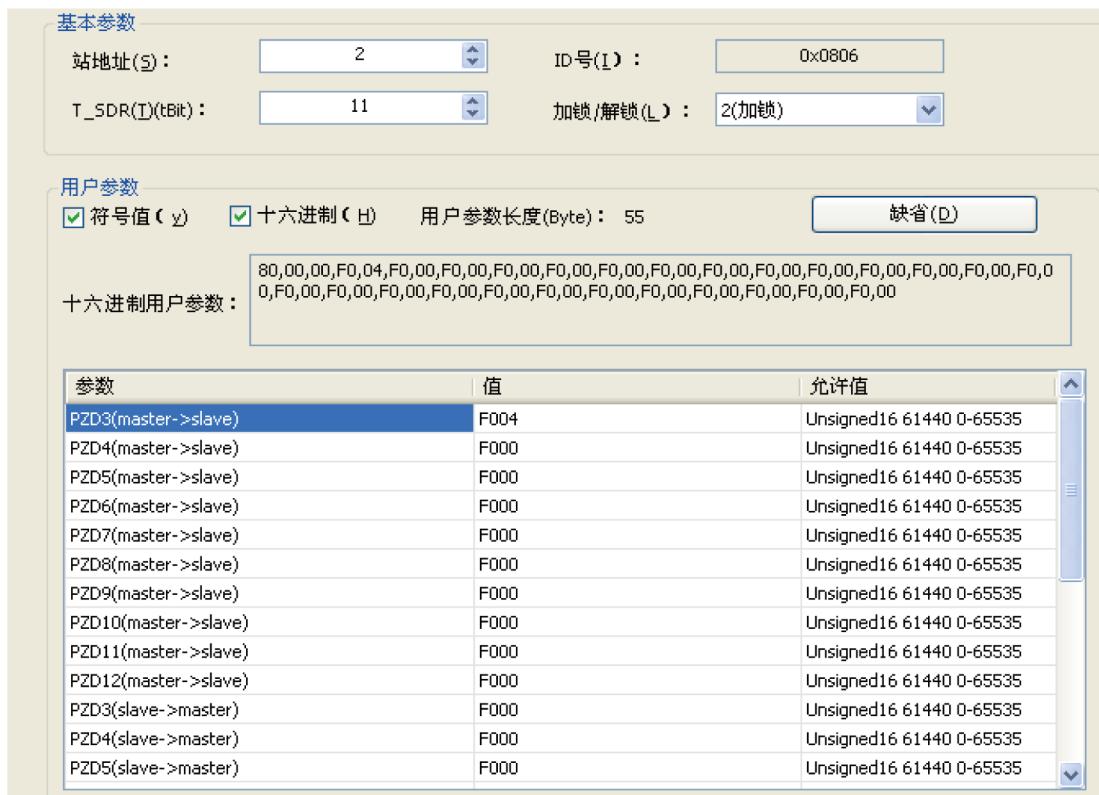


图4-67 Profibus-DP从站参数配置

### 基本参数

- 节点ID：从站在DP网络唯一标示号，范围1-125，默认2，必须是十进制数，要求与从站本身标示号（如拨码开关）一致。
- ID号：是从站设备唯一的识别符，由GSD文件决定。
- T\_SDR：从站的最小响应间隔，指示从站在收到主站数据后的最小回复间隔。
- 加锁/解锁：指示从站当前状态。

### 用户参数

用户参数是由GSD文件定义的，可以选择10进制或者16进制显示，具体含义请参考设备商提供的手册。

## 从站诊断

诊断指示了从站节点的运行状态。



诊断的解析：

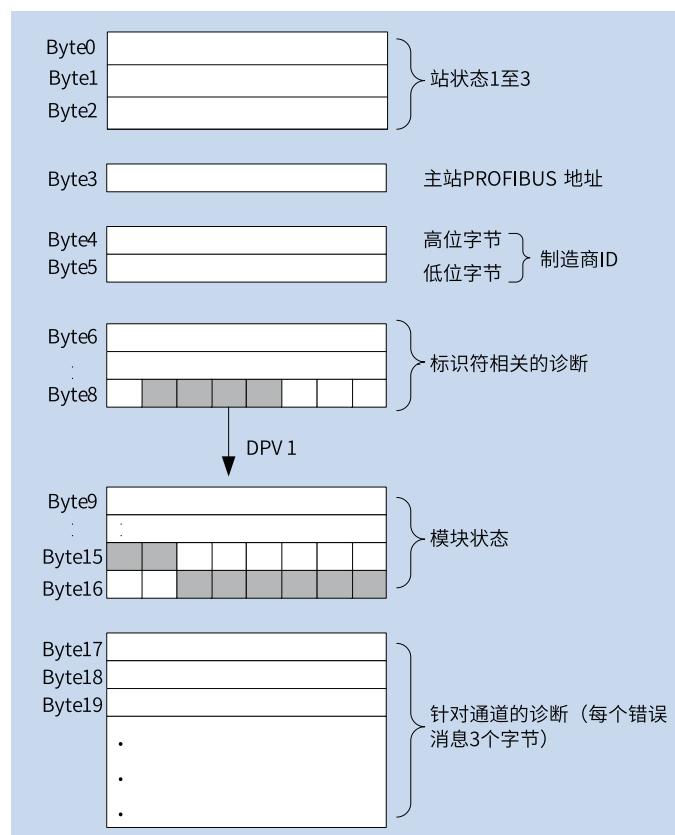


图4-68 诊断报文结构

表4-2 站状态1的含义

位	含义		原因/解决方法
0	0:	DP主站无法寻址DP从站， DP从站中该位始终为0。	DP从站上设置的PROFIBUS地址是否正确？ 总线连接器与FOC是否已连接？ DP从站的电压是多少？ RS485中继器的设置是否正确？ 是否复位了DP从站（打开/关闭）？
1	1:	DP从站尚未准备好进行交换数据。	请等待DP从站完成启动。
2	1:	DP主站发送到DP从站的组态数据与DP从站的实际组态不匹配。	在组态软件中输入的站类型或DP从站的组态是否正确？
3	1:	外部诊断可用。	评估标识符相关的诊断、模块状态和/或通道相关的诊断。纠正了所有错误后，位3即会复位。 在上面所示的诊断字节中，出现新的诊断消息时该位复位。
4	1:	DP从站不支持请求的功能。	检查组态。
5	1:	DP主站无法解释DP从站的响应。	检查总线组态。
6	1:	DP从站类型与软件组态不相符。	站类型的组态软件设置是否正确？
7	1:	其他DP主站（不是当前访问DP从站的DP主站）已组态DP从站。	在通过编程设备或其他DP主站访问DP从站的情况下，该位始终为1。 组态DP从站的DP主站的PROFIBUS地址位于“主站PROFIBUS地址”诊断字节中。

表4-3 站状态2的含义

位	含义	
0	1:	必须重新组态DP从站。
1	1:	从站处于启动阶段。
2	1:	DP从站中该位始终为“1”。
3	1:	已为此DP从站启用响应监视。
4	1:	DP从站已接收到“FREEZE”控制命令。
5	1:	DP从站已接收到“SYNC”控制命令。
6	0:	该位始终为0。
7	1:	DP从站被取消激活，即已将其从当前处理中删除。

表4-4 站状态3的含义

位	含义	
0~6	0:	该位始终为“0”。
7	1:	特定于通道的诊断消息数量超过诊断帧中可以容纳表示的消息。

- 主站Profibus地址：标明该Profibus-DP主站已组态DP从站并且对该DP从站有读写访问权限。值为FF，则DP主站尚未组态DP从站。
- 制造商ID：描述DP从站类型，由设备商声明，在GSD中体现。

#### 4.12.5 Profibus-DP 模块

##### 模块化设备和非模块化设备

在Profibus-DP从站配置中，DP从站节点可接入两种类型的模块设备。

模块化设备：接入DP从站节点下，模块本身自带I/O映射列表，不需要使用Profibus-DP从站I/O映射对话框，从站节点数据随模块自动添加。目前AM600 I/O模块为此类型。

非模块化设备：从站节点对话框包含I/O映射对话框，数据不能自动配置。

## AM600 Profibus-DP I/O模块

AM600 Profibus-DP I/O模块在硬件组态中添加，添加模块后可以配置相关I/O参数，也可以添加I/O映射实现数据刷新，详情请参见CPU下的I/O模块。

## 4.13 FINS TCP&UDP通信



### 注 意

InoProShop V1.8.1.0及以上版本支持该功能。

### 4.13.1 概述

FINS (Factory Interface Network Service) 协议是一种工业自动化控制网络协议，用于客户端和服务器之间进行通信。FINS协议是应用层协议，当使用以太网通信时，通信数据以UDP/IP包或TCP/IP包的形式发送和接收，默认通信端口为9600。

目前仅AC800系列PLC集成FINS协议通信功能，支持存储区域的数据读写功能。AC800系列PLC只能作为FINS TCP/UDP服务器，与欧姆龙PLC、基恩士PLC等建立通信，实现数据交互。AC800系列PLC支持的变量类型为%MW。

AC800系列PLC默认关闭FINS服务器功能，由用户应用编程控制，通过FINS\_Server功能块可开启/关闭FINS协议通信功能，PLC停止运行后不能访问。

### 功能码与地址

- 通过FINS协议命令访问AC800系列PLC的I/O存储区域DM区，通信命令码如下表所示。

FINS协议通信命令码				
类型	一级命令	二级命令	功能	描述
I/O存储区域读写(DM区)	01	01	内存读取	读取连续I/O内存区域数据
	01	02	内存写入	向连续I/O内存区域写入数据

- 将PLC软元件映射在DM区（存储区代码固定为0x82），编址范围为“0~32767”，统一以字元件形式访问，支持FINS协议访问的软元件地址映射表。

变量名称	地址范围	软元件数量	说明
%MW0~%MW32767	0x0000 ~ 0x7FFF (0~32767)	32768	16位寄存器

**注 意**

读写软元件较多时，建议FINS TCP/UDP客户端分包进行数据读写。

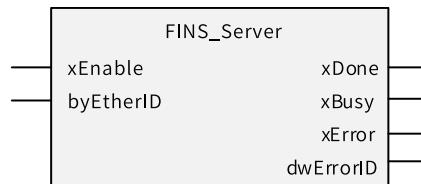
**FINS协议使用约束**

- 默认通信端口9600不可修改。
- FINS TCP/UDP客户端最大连接数为32个。
- FINS协议最大网络节点数为254。
- FINS协议最大连续读取元件数为2000字。
- IP地址最后一个字节设置范围为1~254，不能设置为0和255。

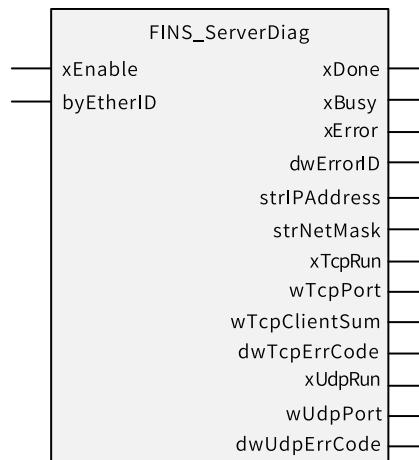
命令	RD-CMD（读命令）	RD-ACK（读应答）	WR-CMD（写命令）	WR-ACK（写应答）
FINS TCP	34字节	30字节 + 2*N	34字节 + 2*N	30字节
FINS UDP	18字节	14字节 + 2*N	18字节 + 2*N	14字节

**说明**

N不超过2000。

**FINS\_Server功能块**

FINS_Server: 开启FINS服务器				
输入输出	接口参数	数据类型	初始值	备注
Input	xEnable	BOOL	FALSE	FINS_Server功能块采用电平方式控制。 当xEnable=TRUE时，表示开启FINS服务器功能，此时客户端均能通过UDP/TCP连接读写PLC数据； 当xEnable=FALSE时，表示关闭FINS服务器功能，此时停止与客户端的数据交互，包括UDP/TCP连接。
Input	byEtherID	BYTE	0	byEtherID表示以太网口ID号，有效值从0开始， 最大值与PLC以太网口个数有关。 例如AC802支持2个EtherNET网口，有效输入值为 0-1，0表示LANA，1表示LANB。
Output	xDone	BOOL	FALSE	-
Output	xBusy	BOOL	FALSE	-
Output	xError	BOOL	FALSE	-
Output	dwErrorID	DWORD	0	• 0: 无错误 • 1: 输入参数错误（网络ID配置异常） • 2: 网络冲突错误 • 3: 执行错误（执行异常）

**FINS\_ServerDiag诊断功能块**

FINS_ServerDiag: 获取FINS服务器诊断信息				
输入输出	接口参数	数据类型	初始值	备注
Input	xEnable	BOOL	FALSE	FINS_ServerDiag功能块采用电平方式控制。 当xEnable=TRUE时，刷新FINS服务器诊断信息；当xEnable=FALSE时，停止读取FINS服务器诊断信息。
Input	byEtherID	BYTE	0	byEtherID表示以太网口ID号，有效值从0开始，最大值与PLC以太网网口个数有关。 例如AC802支持2个EtherNET网口，有效输入值为0-1，0表示LANA，1表示LANB。
Output	xDone	BOOL	FALSE	-
Output	xBusy	BOOL	FALSE	-
Output	xError	BOOL	FALSE	-
Output	dwErrorID	DWORD	0	• 0: 无错误 • 1: 输入参数错误（网络ID配置异常） • 2: 网络冲突错误 • 3: 执行错误（执行异常）
Output	strIPAddress	STRING(17)	“192.168.1.88”	网口IP地址
Output	strNetMask	STRING(17)	“255.255.255.-0”	网口子网掩码
Output	xTcpRun	BOOL	FALSE	TCP运行状态
Output	wTcpPort	WORD	9600	TCP端口号
Output	wTcpClientSum	WORD	0	TCP客户端连接数
Output	dwTcpErrCode	DWORD	0	TCP错误码，见下表
Output	xUdpRun	BOOL	FALSE	UDP运行状态
Output	wUdpPort	WORD	9600	UDP端口号
Output	dwUdpErrCode	DWORD	0	UDP错误码，见下表

FINS\_ServerDiag诊断功能块中UDP/TCP错误码如下表所示。

FINS UDP/TCP错误码		
错误码	描述	备注
0	无错误	-
1	FINS帧头错误	FINS TCP数据帧帧头应为0x46494E53
2	FINS错误码异常错误	FINS TCP握手命令帧错误码应为0x00000000
3	FINS客户端节点超限	FINS客户端节点不能超过254
4	FINS握手失败	FINS TCP客户端握手未成功
5	FINS固定值填充错误	FINS固定值数据填充错误，不符合FINS命令帧
6	FINS访问存储区类型错误	FINS读写命令帧访问的存储区类型不支持
7	FINS命令帧格式错误	FINS命令帧长度错误
8	FINS读/写数量超限	FINS命令帧读写长度超过2000或者为0
9	FINS访问地址范围错误	FINS访问地址范围错误
10	FINS功能请求码错误	FINS功能码不支持
12	FINS帧长度异常	帧长度异常，不是FINS帧

#### 4.13.2 FINS TCP&UDP通信配置

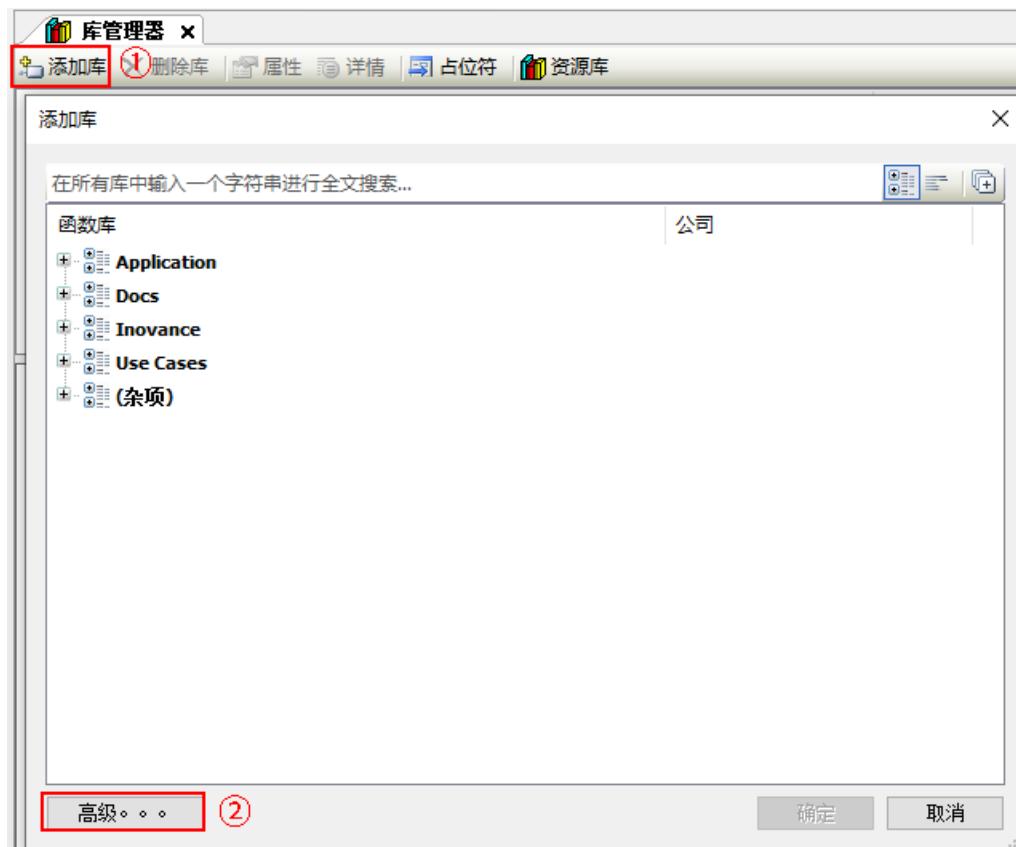
本节以HMI7100E作为FINS客户端，AC802作为FINS服务器，通过FINS TCP协议通信实现寄存器数据读功能为例进行介绍。

##### 1. 添加HC\_FINS库。

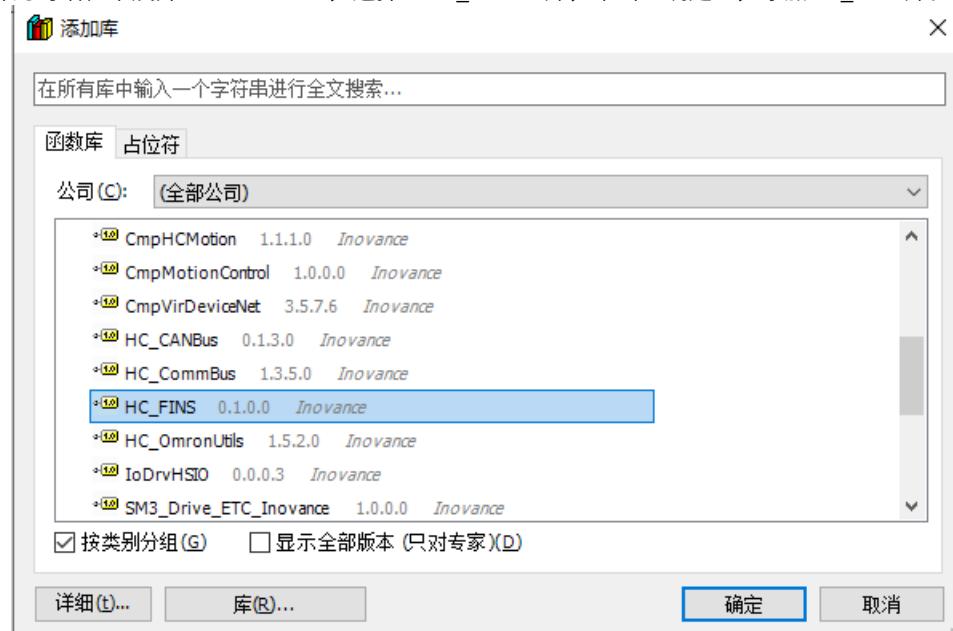
- a. 在设备树中双击“库管理器”，打开“库管理器”界面。



- b. 单击“添加库”，在打开的对话框中单击“高级。。。”。



c. 在打开的对话框中展开“Inovance”，选择“HC\_FINS”库，单击“确定”，添加HC\_FINS库。



2. 开启FINS服务器功能。

- 在设备树中双击“PLC\_PRG”，实例化FINS\_Server功能块，“xEnable”置为“TRUE”，开启LANA 网口的FINS服务器功能。
- 实例化FINS\_ServerDiag功能块，“xEnable”置为“TRUE”，监测LANA 网口的FINS协议状态。



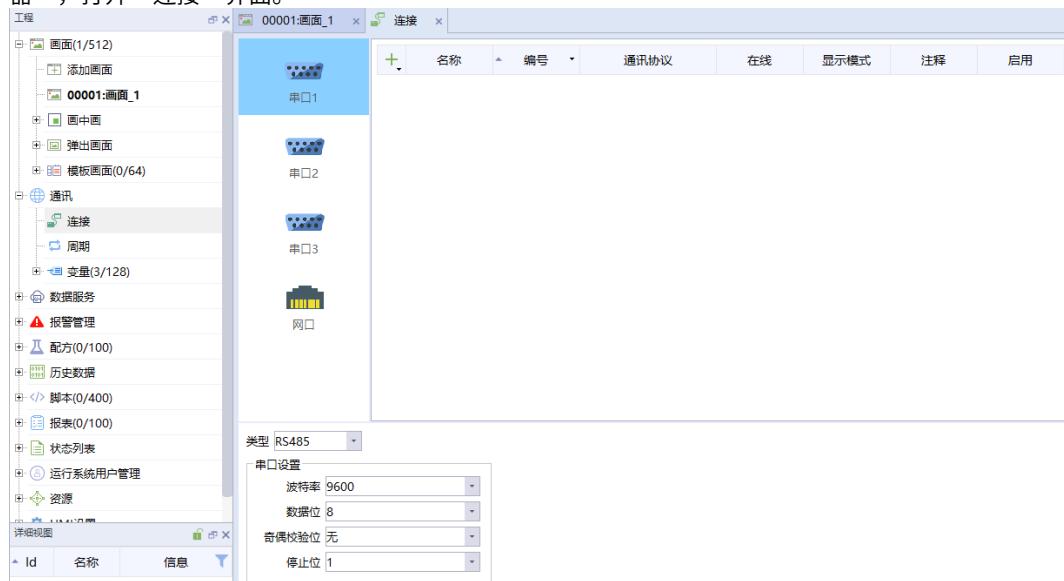
The screenshot shows the Device Application PLC\_PRG configuration interface. It displays two sets of parameters under the '表达式' (Expression) column:

表达式	类型	值	准备值	地址	注释
FINS_Server0	FINS_Server	TRUE			功能块激活标志: ...开启, 低电平关...
xEnable	BOOL	TRUE			Ethernet端口ID: 值从0开始, 根据...
byEtherID	BYTE	0			功能块完成标志
xDone	BOOL	TRUE			功能块运行标志
xBusy	BOOL	FALSE			功能块错误标志
xError	BOOL	FALSE			功能块错误码
dwErrorID	DWORD	0			
FINS_ServerDian0	FINS_ServerDian				
xEnable	BOOL	TRUE			功能块激活标志: ...激活, 低电平停...
byEtherID	BYTE	0			Ethernet端口ID: 值从0开始, 根据...
xDone	BOOL	TRUE			功能块完成标志
xBusy	BOOL	FALSE			功能块运行标志
xError	BOOL	FALSE			功能块错误标志
dwErrorID	DWORD	0			功能块错误码
strIPAddress	STRING(17)	'10.45.121.26'			FinsServer诊断信息: IP地址
strNetMask	STRING(17)	'255.255.255.0'			FinsServer诊断信息: 子网掩码
xTcpRun	BOOL	TRUE			FinsServer诊断信息: TCP运行状态
wTcpPort	WORD	9600			FinsServer诊断信息: TCP端口号
wTcpClientSum	WORD	0			FinsServer诊断信息: TCP客户端数
dwTcpErrCode	DWORD	0			FinsServer诊断信息: TCP错误码
xUdpRun	BOOL	TRUE			FinsServer诊断信息: UDP运行状态
wUdpPort	WORD	9600			FinsServer诊断信息: UDP端口号
dwUdpErrCode	DWORD	0			FinsServer诊断信息: UDP错误码

### 3. 开启FINS客户端功能并读寄存器数据。

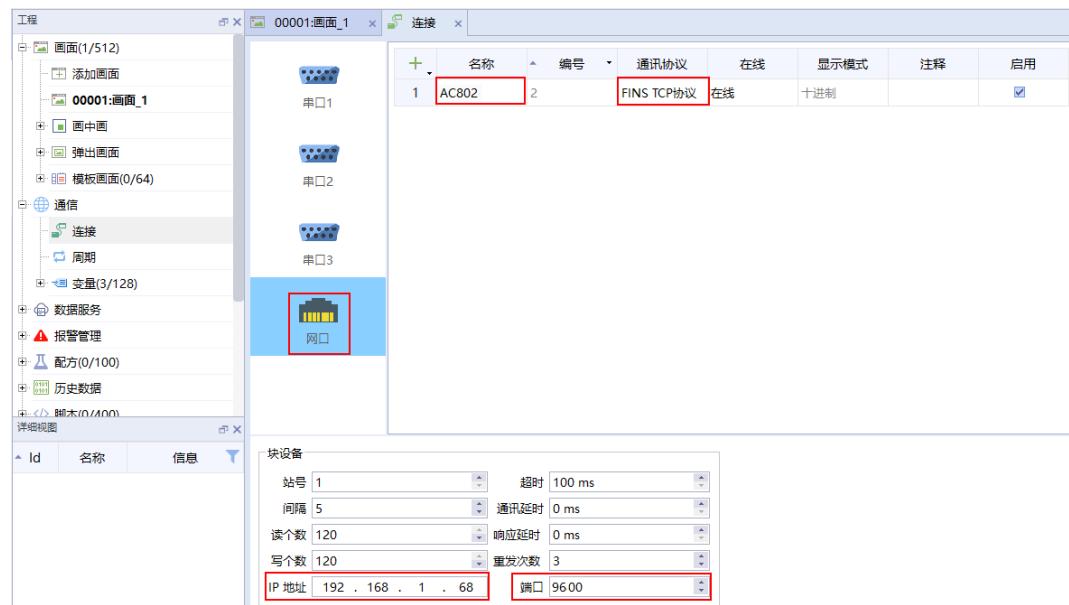
#### a. 创建连接。

- 1). 在InoTouchPad工程树中展开“通信”节点，双击“连接”或右键单击“连接”，选择“打开编辑器”，打开“连接”界面。



- 2). 单击“网口”，单击`+`，输入连接自定义名称，“通信协议”选择“FINS TCP协议”，设置FINS TCP服务器IP地址和端口。

## 网络配置



b. 创建变量。

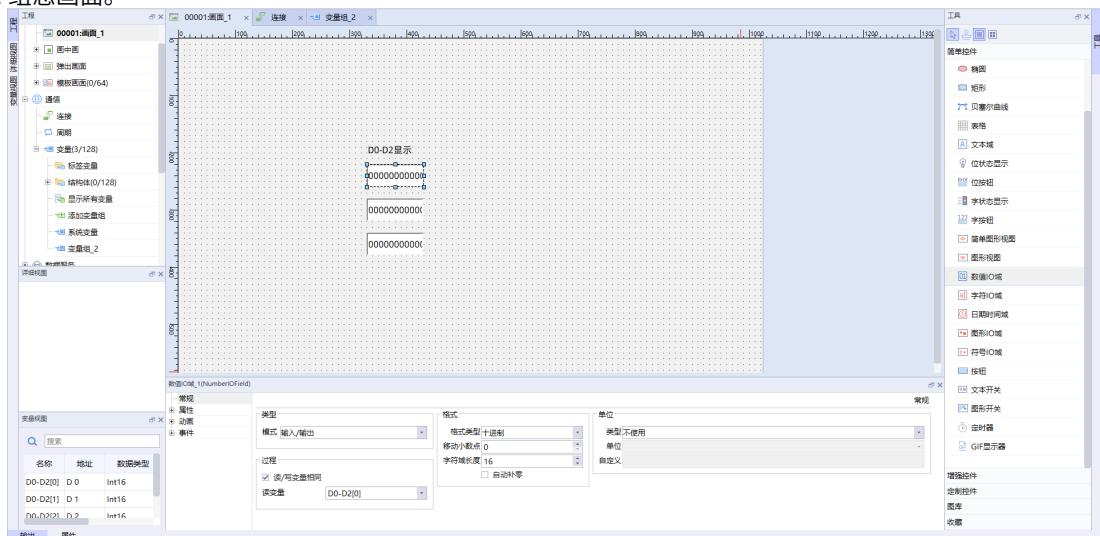
- 1). 在“通信”节点下展开“变量”，双击“变量组\_2”或右键单击“变量组\_2”，选择“打开编辑器”，打开“变量组\_2”界面。



- 2). 单击 **+**，输入名称，“数组计数”设置为“3”，“采集周期”设置为“100ms”。

名称	编号	连接	数据类型	长度	数组计数	地址	采集周期	采集模式
1 D0-D2	1	AC802	Int16	6	3	D 0	100ms	循环使用

c. 组态画面。



- 1). 在工程树中展开“画面”，双击“画面\_1”，进入“画面\_1”界面。
- 2). 在“工具”区拖入“文本域”至编辑区中，命名为“D0-D2显示”。

- 3). 在“工具”区拖入“数值IO域”3次至编辑区中，并垂直对齐，“读变量”分别关联“D0-D2[0]”、“D0-D2[1]”和“D0-D2[2]”。
- d. 编译并下载工程至HMI中运行。
- 1). 在菜单栏选择“编译(M) > 编译(M)”，或在工具栏单击，或按快捷键F5，编译工程。
  - 2). 在菜单栏选择“编译(M) > 下载工程...”，或在工具栏单击，或按快捷键F7，下载工程至HMI中运行。
4. 在HMI显示屏中查看画面“数值I/O域”显示框中数值分别为“1”、“2”和“3”，读取FINS TCP服务器“%MW0”、“%MW1”和“%MW2”的值成功。

## 4.14 与HMI通信配置

### 4.14.1 通信配置

本驱动构件通过InoTouch Editor软件组态，基于Modbus TCP/IP协议读写汇川中型系列PLC设备的各种寄存器的数据。

HMI支持01、31、03、33、05、35、06、36、0F、3F、10、40功能码，有关功能码的详细信息，请参见相应HMI产品用户手册。

配置项目	说明
通讯协议	采用汇川AM600系列PLC内置MODBUS TCP IP协议
通讯方式	一主一从、一主多从方式。驱动构件为主，设备为从
通讯设备	以太网子设备，须挂接在“通用TCP IP父设备”下才能工作

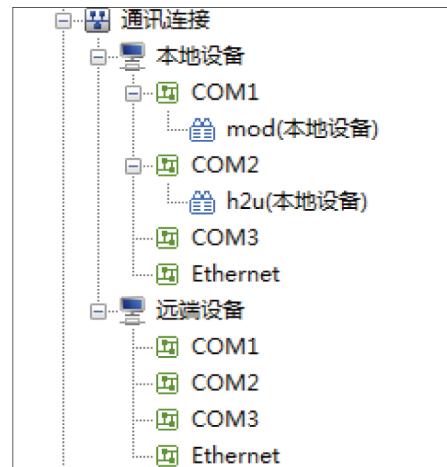
### 硬件连接

InoTouch Editor软件与设备通讯之前，必须保证硬件通讯连接正确。

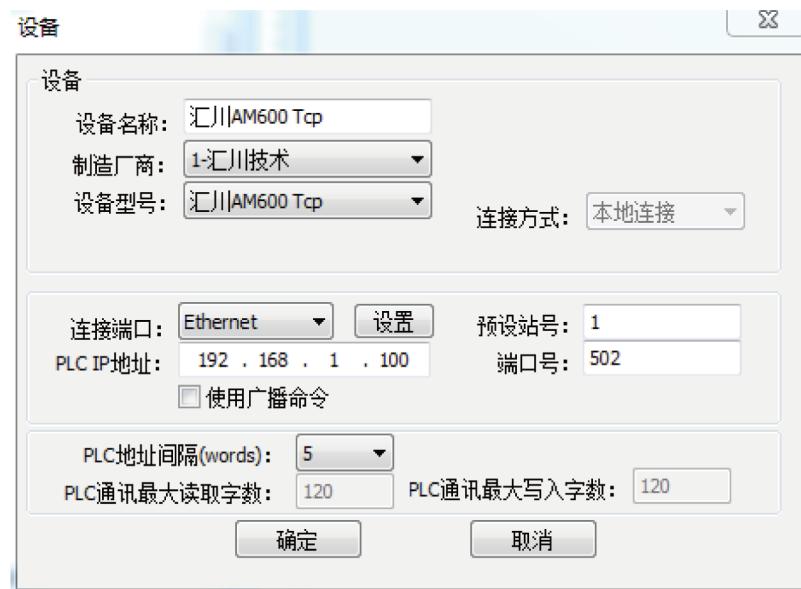
连接方式：采用RJ-45网线直连HMI与PLC（直通网线或使用HUB交换）。

### 设备通讯参数

InoTouchEditor 客户端通讯参数设置如下：



通讯连接中，选择Ethernet,增加设备如下：



- PLC IP地址：本栏需按PLC设定实际情况正确填写PLC IP地址。
- 端口号：上位机与下位机数据帧发送/接收端口，默认情况下为502，建议用户不要修改默认设置值。其他条目采用默认设置。

## 通信设置

“汇川AM600-ModbusTCP”子设备参数设置如下：



响应延时：发送帧发送后到启动接收的时间间隔，默认为0ms。

## 采集通道

### 通讯状态

通讯状态值	意义
通讯正常	表示当前通讯正常
命令失败	表示设备命令读写操作失败错误
校验失败	表示采集数据校验错误
通讯超时	表示采集无数据返回错误

## 内部属性

用户可通过内部属性添加通道，本驱动构件可支持汇川AM600系列PLC内置ModbusTCP寄存器类型及对应功能码如下：

寄存器	数据类型	读取功能码	写入功能码	操作方式	通道示例
[SM区]输入线圈	BT	31	35、3F	读写	“SM只读0000” 表示SM区地址0
[Q区]输出线圈	BT	01	05、0F	读写	“Q读写0001” 表示Q区地址1
[SD区]输入寄存器	16-bit-BCD 32-bit-BCD 16-bit-unsigned 16-bit-signed 32-bit-unsigned 32-bit-unsigned 32-bit-float	33	36、40	读写	“SD只读0002” 表示SD区地址2
[M区]输出寄存器	16-bit-BCD 32-bit-BCD 16-bit-unsigned 16-bit-signed 32-bit-unsigned 32-bit-unsigned 32-bit-float	03	06、10	读写	“M读写0003” 表示M区地址3

功能码：[SD区]在双字(32位)数据写操作或批量写入多个数据时，使用40功能码；

[M区]在双字(32位)数据写操作或批量写入多个数据时，使用10功能码。

## 说明

在内部属性中添加通道时，起始地址均为0，这是遵从汇川AM600内置ModbusTCP协议的，即所谓的“协议地址”。

### 4.14.2 通信示例

#### 读写位变量

分别位状态指示灯和位状态切换开关各一个控件，如下所示：

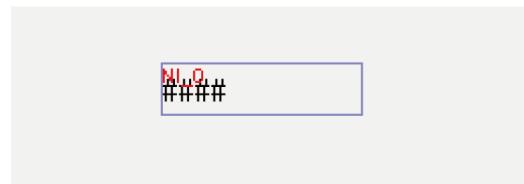


修改位状态指示灯控件一般属性，设置地址为Q\_bit(0:0)。



## 读写字变量

选择数值输入控件放入页面，如下图所示：



修改数值输入控件一般属性，设置地址为MW0。



#### 4.14.3 常见故障分析

故障现象	分析	处理建议
通讯超时	采集初始化错误 或采集无数据返回 (通讯硬件连接、参数设置问题)	1、检查网络设备参数设置是否正确
		2、检查串口是否被其他程序占用
		3、检查通讯电缆是否正确连接
		4、读取数据地址超范围
命令失败	读写操作失败	1、读取数据地址超范围
		2、通讯电缆太长，做短距离测试
		3、现场干扰太大，避免周围环境干扰

## 驱动构件支持的寄存器及功能码

寄存器	读取功能码	写入功能码	功能码说明
[SM区]输入线圈	31	35 3F	31：读取输入线圈状态 35：强制单个输入线圈 3F：强制多个输入线圈
[Q区]输出线圈	01	05 0F	01：读取输出线圈状态 05：强制单个输出线圈 0F：强制多个输出线圈
[SD区]输入寄存器	33	36 40	33：读输入寄存器 36：预置单个寄存器 40：预置多个寄存器
[M区]输出寄存器	03	06 10	03：读保持寄存器 06：预置单个寄存器 10：预置多个寄存器

说明：

1. 本驱动构件支持01、31、03、33、05、35、06、36、0F、3F、10、40等常用功能码，对于其它非数据通讯用功能码暂不支持。
2. 以上功能码均以16进制标注。功能码0x0F和0x10分别对应10进制的15和16。
3. “[SM区]输入线圈”在批量写入多个继电器时，使用3F功能码。
4. “[Q区]输出线圈”在批量写入多个继电器时，使用0F功能码。
5. “[SD区]输出寄存器”在双字(32位)数据写操作或批量写入多个数据时，使用40功能码。
6. “[M区]输出寄存器”在双字(32位)数据写操作或批量写入多个数据时，使用10功能码。

## 说明

添加寄存器通道时，起始地址均为0，这是遵从汇川AM600内置Modbus协议的，即所谓的“协议地址”。

## 数据类型表

BTdd	位 (dd范围：00—15)
16-bit-BCD	16位 BCD
32-bit-BCD	32位 BCD
16-bit-unsigned	16位 无符号
16-bit-signed	16位 有符号
32-bit-unsigned	32位 无符号
32-bit-signed	32位 有符号
32-bit-float	32位 浮点数

## 32位数在寄存器中的存储

1. PLC工程中变量名称与寄存器地址对应关系（寄存器16位）
  - 变量%MW5—对应寄存器地址5。
  - 变量%MD5—对应寄存器地址是10。
 变量名称前缀：W—字 (16 Bits); D—双字 (32 Bits)。
2. 驱动设备命令以及通道采集寄存器地址都是从0开始编址，0就代表寄存器地址为0。

3. 当读取或者写入32位数据时，都是从对应寄存器地址开始占两个寄存器（即32位）。

# 5 编程基础

## 5.1 概述

操作数是用户程序中操作符、功能、功能块或者程序操作的对象，可以作为输入、输出和中间保存结果。在 CoDeSys 中，常见的操作数包含直接地址、常量和变量。

与其他高级语言类似，CoDeSys 也有常量和变量的概念。所谓常量就是数值不变的数；变量是由用户定义的标识符。变量的存储位置可由用户指定为 %I 区、%Q 区、%M 区的特定地址，亦可不指定地址，由系统自行分配，用户不需要关注这些变量的存储位置。

## 5.2 直接地址

### 5.2.1 定义语法

此类型固定地址也叫直接变量，直接映射到 PLC 设备的具体地址。地址信息包含了变量在 CPU 的存储位置，存储大小及存储位置对应的偏移。

语法：%< 存储器区前缀>< 大小前缀>< 数字>|.< 数字>

- < 存储器区前缀 >：编程系统支持以下存储区前缀
  1. I：输入，物理输入，“传感器”
  2. Q：输出，物理输出，“执行器”
  3. M：存储位置
- < 大小前缀 >：编程系统支持以下大小前缀
  1. X：bit，一位
  2. B：Byte，一个字节
  3. W：Word，一个字
  4. D：Double Word，四个字节（双字）
- < 数字 >|.< 数字 >

第一个数字是存储区前缀的偏移地址，如果定义 BOOL 类型变量，需使用< 数字>.< 数字> 格式，“.” 后的数字是偏移地址的偏移位数。

示例：

%QX7.5 输出区域偏移7个字节的第六位 (bit5)

%QB17 输出区域偏移17个字节

%IW215 输入区域偏移215个字

%MD48 内存区域偏移48个双字

iVar AT %IW10: WORD;//iVar 变量是字类型，映射到输入区域偏移10字的位置

## 说明

- 大小前缀为X类型变量代表的数据类型为BOOL型，偏移地址应具体到位。
- 大小前缀和数据类型是匹配的，大小前缀为B类型的变量应声明为一个字节的数据类型，如BYTE, SINT, USINT；大小前缀为W类型的变量应声明为一个字的数据类型，如WORD, INT, UINT；大小前缀为D类型的变量应声明为一个双字的数据类型，如DWORD, DINT, UDINT。

## 5.2.2 PLC 直接地址存储区域

不同PLC提供的直接存储区域不同。对于PLC数据，%I、%Q区地址不能掉电保存，对于%M区可以掉电保存。

AM600、AM610、AM401、AM402编程系统提供128KB (Byte) 的输入区域 (I区) , 128KB (Byte) 输出区域 (Q区) 和512KB存储区域 (M区) , 其中存储区域中的前480KB用户可以直接使用，后32K为系统使用区域（主要用作软元件），用户不要直接使用。编程时，用户可以直接访问地址，也可以定义变量后把变量映射到地址间接访问。存储区域定义及使用的地址范围如下表。

区域	用途	大小	地址范围
I区 (%I) 128KB	用户使用区域	64KWords	%IW0-%IW65535
Q区 (%Q) 128KB	用户使用区域	64KWords	%QW0-%QW65535
M区 (%M) 512KB	用户使用区域	240KWords	%MW0-%MW245759
	SD元件	10000Words	%MW245760-%MW255759
	SM元件	10000Bytes	%MB511520-%MB521519
	保留	2768Bytes	%MB521520-%MB524287

AC800系列编程系统提供128KB (Byte) 的输入区域 (I区) , 128KB (Byte) 输出区域 (Q区) 和5MB存储区域 (M区) 。AC800系列不支持SD和SM软元件，%M区地址可以随意使用。存储区域定义及使用的地址范围如下表。

区域	用途	大小	地址范围
I区 (%I) 128KB	用户使用区域	64KWords	%IW0-%IW65535
Q区 (%Q) 128KB	用户使用区域	64KWords	%QW0-%QW65535
M区 (%M) 5MB	用户使用区域	2.5MWords	%MW0-%MW2321439

## 5.3 变量

### 5.3.1 概述

变量可以在POU的定义部分或者通过自动声明对话框定义，也可以在DUT 或者 GVL 编辑器定义，通过变量类型关键字来标识变量类型，例如通过VAR和END\_VAR来标识它之间定义的变量为本地变量。

变量类型包括本地变量(VAR)，输入变量(VAR\_INPUT)，输出变量(VAR\_OUTPUT)，输入输出变量(VAR\_IN\_OUT)，全局变量(VAR\_GLOBAL)，临时变量(VAR\_TEMP)，静态变量(VAR\_STAT)，配置变量(VAR\_CONFIG)。

### 5.3.2 变量定义

变量可以在声明编辑器中定义。声明编辑器有两种显示形式：文本视图和表格视图。结构体和数组复杂数据类型，支持变量定义支持数组元素注释，支持复杂数据类型地址递归显示。

文本声明如下图：

```

VAR_GLOBAL
END_VAR
VAR_GLOBAL RETAIN PERSISTENT
    (attribute 'ElemComment':='A_0([(ERT),9()),A_1([10())])')
    A AT%MW0:DUT := (A_0 := [2(FALSE), TRUE, 7(FALSE)]);
END_VAR
VAR_GLOBAL
    (attribute 'ElemComment':='A_0([(EEE),9()),A_1([10())])')
    B AT%MW200:DUT;
    C AT%MW400:DUT;
END_VAR

```

图5-1 文本声明

表格声明如下图：

类别	名称	地址	数据类型	初值	保持	常量	注释	特性
VAR_GLOBAL RETAIN PERSISTENT	<b>A</b>	%MW0	DUT	(A_0 := [2(FALSE), TRUE, 7(FALSE)])	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
VAR_GLOBAL	<b>B</b>	%MW200	DUT		<input type="checkbox"/>	<input type="checkbox"/>		
VAR_GLOBAL	<b>C</b>	%MW400	DUT		<input type="checkbox"/>	<input type="checkbox"/>		

图5-2 表格声明

表格声明中，可以对变量的各类属性进行编辑设置。下表是对表格声明的具体说明。

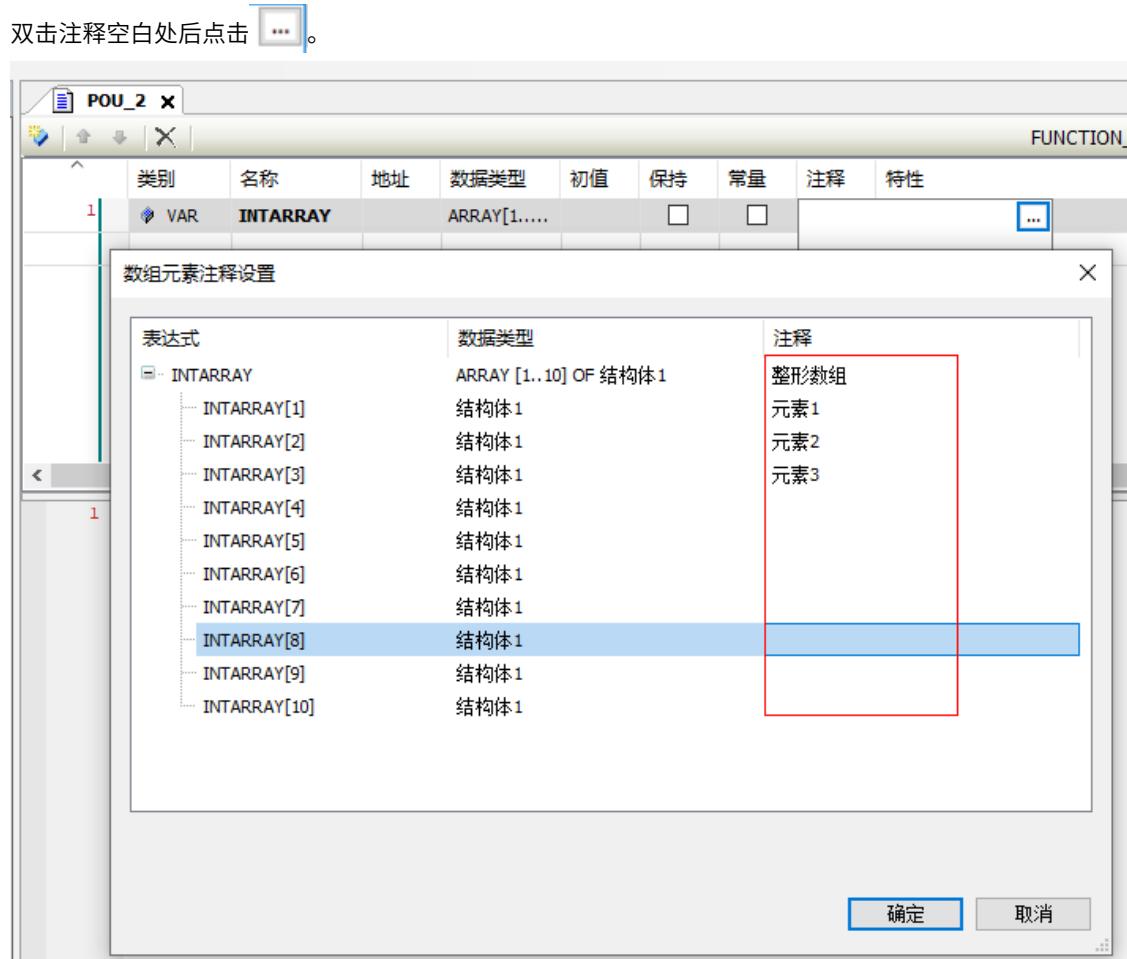
项目	描述
类别	变量的类别（如本地变量，输入变量，输出变量，临时变量等）
名称	变量的名称
地址	变量编译后的地址
数据类型	即变量的数据类型（如INT，BOOL等）
初值	变量的初始值
保持	标识变量是否为保持变量
常量	标识定义的变量是否为常量
注释	变量的注释
特性	变量的特性

### 变量定义支持数组元素注释和实例注释

#### 1. 数组元素注释

表格声明注释设置界面如下图所示。

POU_2									
	类别	名称	地址	数据类型	初值	保持	常量	注释	特性
1	VAR	INTARRAY		ARRAY[1.....]	<input type="checkbox"/>	<input type="checkbox"/>			
按 Ctrl+Enter添加新的行									



设置完成后文本声明效果如下图所示（同样也可以用文本直接声明）：

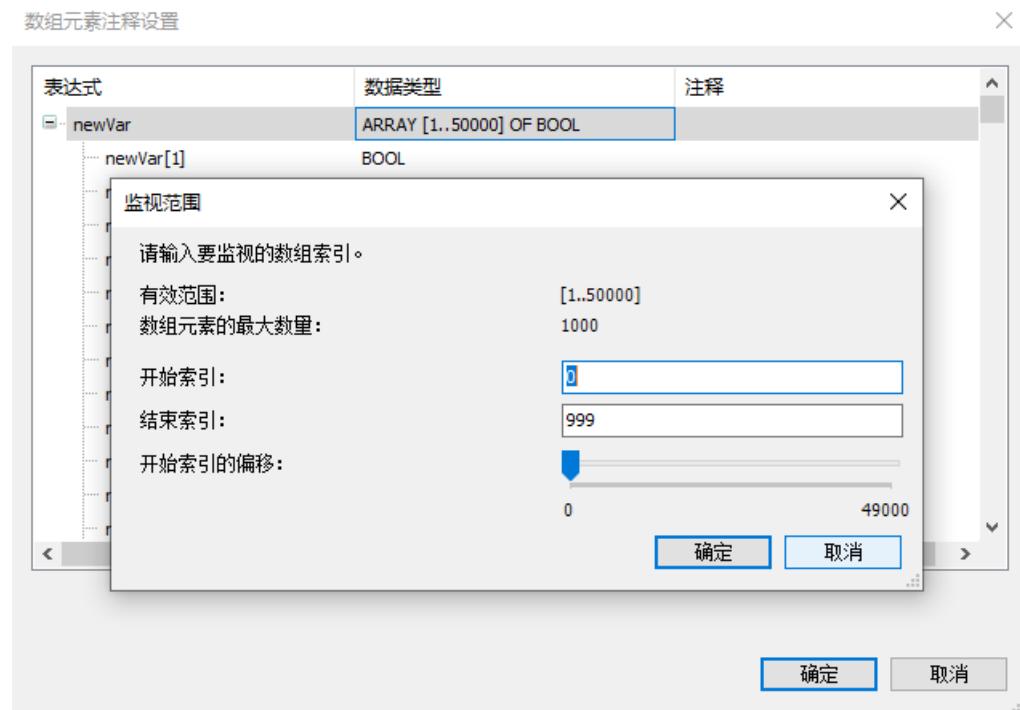
```

1 FUNCTION_BLOCK POU_2
2 VAR_INPUT
3   END_VAR
4 VAR_OUTPUT
5   END_VAR
6 VAR
7   // 整形数组
8   {attribute 'ElemComment':='(元素1),(元素2),(元素3),7()' }
9   INTARRAY: ARRAY[1..10] OF 结构体1;
10 END_VAR
11

```

- 可以通过表格和文本进行数组元素注释编辑。
- 表格在注释列，弹框（同 initialValue 操作）显示当前元素及子元素注释编辑界面。
- 文本编辑器编辑格式如下：
  - 数组本身：使用标准的注释编辑方式。
  - 数组元素：{attribute 'ElemComment':='1(子元素1注释),1(子元素2注释),n(相同子元素注释)' }。
- 如果是表格模式，声明数组类型变量时，默认注释为空（默认增加特性格式）。
- 表格模式的数组元素注释，只显示数组本身注释，不显示元素注释。
- 特性列中，去掉元素注释特性显示（数组元素注释是用特性来实现的，特性是标记在变量上的信息）。
- 表格视图数组长度变更时，存量数组元素的注释也会跟随保存。

- 表格视图中数组维度变更时，数组元素注释按扩展维度的最小下标对注释进行平移保存。  
如数组 INT\_ARRAY:ARRAY[1..2,2..3]维度变更为ARRAY[1..2,2..3,3..4],原来的数组元素INT\_ARRAY[1,2]注释会迁移到新数组元素INT\_ARRAY[1,2,3]中。
- 如 INT\_ARRAY:ARRAY[1..2,2..3]维度变更为ARRAY[1..2]，原来的数组元素INT\_ARRAY[1,2]注释会迁移  
到新数组元素INT\_ARRAY[1]中。
- 表格视图中，数据类型由数组类型变更为非数组类型时，会清空数组元素注释。
- 数组元素注释编辑界面最大显示1000个元素，双击数组所在的行中‘数据类型’列，可调节编辑显示范围。



## 2. 实例注释

在Prg（程序）和GVL（全局变量表）中申明的变量或申明为VAR\_STAT（静态）类型的变量，可以不受限制地展开变量内部成员编辑注释，保存注释时所有内部成员的注释会标记在该变量上，这种注释称为变量的实例注释。

如下图所示，数据结构体内的成员注释都可以标记并保存在变量数组结构体上。



```

1 VAR_GLOBAL
2 END_VAR
3 VAR_GLOBAL
4 // i第一层
5 (attribute 'ElemComment':='(i1[a1(ia1),a2(ia2),a3(ia3[(ia31),9()]),a4(ia4[(ia41),9()])],(i2[a1(i21),a2(i22),a3(i23),a4(i24)]),(i3),7())'
6 数组结构体: ARRAY[1..10] OF 结构体1 := [2((a1 := TRUE)), 8(())];
7 END_VAR
8

```

- 内部成员为FB类型时，只会显示输入、输出、输入输出变量，其他类型的变量不会展示。
- 表格中，数据类型由非数组类型变更为数组类型时，会清空实例注释。
- 变量的数组类型成员最大显示1000个元素，可调节编辑显示范围。

### 3. 注释的显示

在初始值编辑界面，监控变量表界面，梯形图，鼠标悬停显示注释等涉及到变量注释显示的功能，注释显示以实例注释优先，如果变量没有实例注释，则显示变量的类型注释。

梯形图中涉及到数组元素的注释显示的，以 数组注释 元素注释 合并显示；但同样采用上面的优先级顺序规则进行显示。

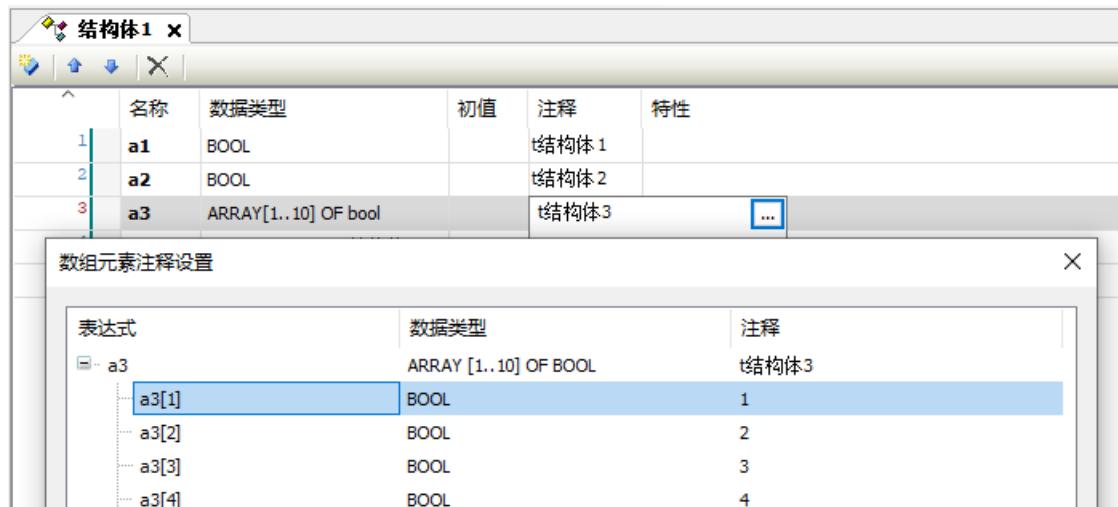
例如下图显示数组元素 数据结构体[1].a3[1] 的注释。



实例注释：



因为有实例注释，定义在类型中的数组注释和元素注释此时不显示。如下图所示。



## 变量的初始值设置

初始值设置界面如下。



- 变量声明表格模式，点击“初值”列，可直接编辑初始值，也可以点展开按钮展开编辑。
  - 当设置的初始值与默认值不一样时，初始值以粗体显示。
  - 初始值设置界面会把变量的成员逐层展开，中间变量不能直接设置，只有终端变量才能设置初始值，能设置初始值的‘表达式’列以粗体显示。
  - 功能块内部的不是输入、输出类型变量的成员以灰底色显示，并且不能编辑。
- 例：FB2中的成员newVar不是输入、输出类型的变量。

FUNCTION_BLOCK FB2								
	类别	名称	地址	数据类型	初值	保持	常量	
1	VAR	newVar		ARRAY[1..3] OF FB1	[[A := 111, B := 222, ARR2 := 666, ARR1 := [111, 222, 333], SSS := 444), 2(0)]]	<input type="checkbox"/>	<input type="checkbox"/>	

FB2类型的newVar成员展开时，无法编辑初始值（显示灰色底）。

表达式	初始值	数据类型	注释
newVar1		FB2	
newVar		ARRAY [1..3] OF FB1	
newVar[1]		FB1	
A	111	INT	
B	222	INT	
C	0	INT	
ARR2	666	INT	
ARR1		ARRAY [1..10] OF INT	
TTT	5	INT	
SSS	444	INT	
newVar[2]		FB1	
newVar[3]		FB1	

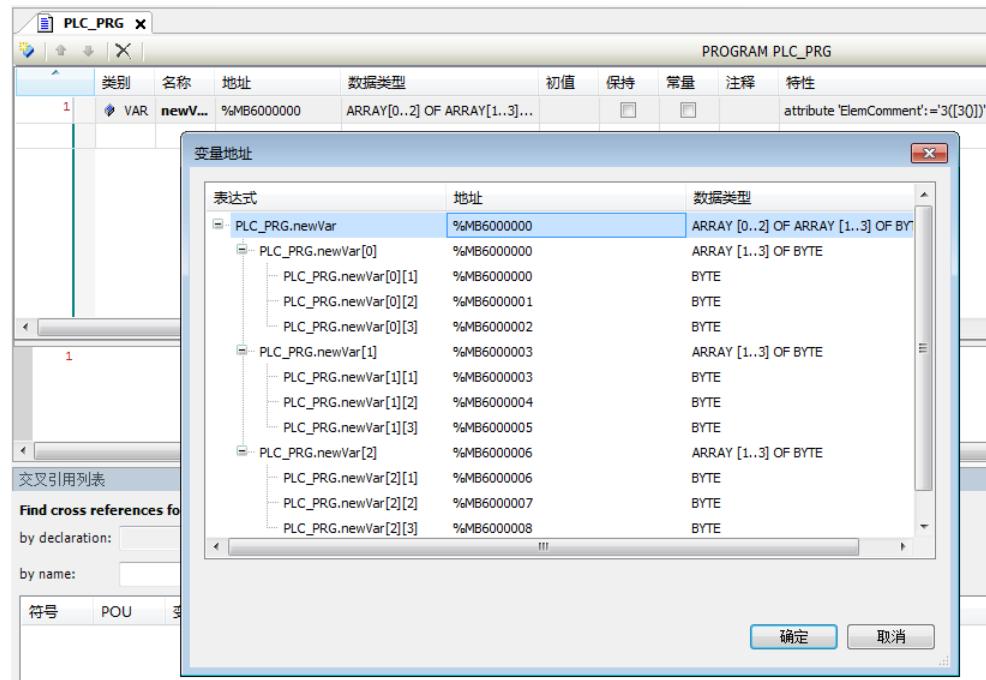
不同于默认初始值的以粗体显示。  
功能块的非输入输出变量以灰底显示。

**确定**    **取消**

- 如果展开项目中有数组，数组最大显示1000个元素，可双击数组所在行的“数据类型”单元格调节编辑显示范围。

## 变量定义的子元素显示地址信息

地址显示界面如下：



- 变量声明表格模式，在地址列，编辑，如果变量是不包含功能块的块类型（数组，结构体，联合体、别名）弹出编辑框，显示子元素地址。
- 在地址列文本框中输入地址，地址显示界面只读。
- 弹出的地址编辑界面地址修改后，可以同步到变量地址中。
- 最大显示元素个数（数组中最大显示1000个元素，可调节显示范围）。

## 标识

标识即变量的名称。变量命名应注意以下事项：

- 不能包含空格或者特殊字符。
- 不能是预定义的关键字。
- 名称不区分大小写。
- 名称长度没有限制。
- 名称不能重复定义。

定义的本地变量名称可以和全局变量重名，默认使用本地变量，该变量可以用来表示全局变量，也可以通过全路径变量名来指定具体的变量。例如：本地变量iVar:=1;全局变量 iVar:=2;全路径变量globlist1 iVar:=3。

命名时需考虑部分命名建议：如变量名应准备表达其意义及数据类型，变量最好采用匈牙利命名法（变量名=属性+类型+对象描述）。

## AT 地址

AT地址为直接地址，具体内容请参见[第360页“5.2.1 定义语法”](#)。

## 数据类型

数据类型分为标准数据类型和用户自定义数据类型。

### 1. 标准数据类型

标准数据类型分为布尔类型、整形、浮点型、字符串、时间类型。

类型	关键字	范围	占用内存
布尔类型	BOOL	TRUE, FALSE, 0, 1	8bit
bit类型	bit	TRUE, FALSE, 0, 1, 只能在结构体或者功能块中使用	1bit
整数	Byte	0~255	8bit
	WORD	0~65535	16bit
	DWORD	0~4294967295	32bit
	LWORD	0 ~2 <sup>64</sup> -1	64bit
	SINT	-128~127	8bit
	USINT	0~255	8bit
	INT	-32768~32767	16bit
	UINT	0~65535	16bit
	DINT	-2147483648~2147483647	32bit
	UDINT	0~4294967295	32bit
浮点类型	LINT	-2 <sup>63</sup> ~2 <sup>63</sup> -1	64bit
	ULINT	0 ~2 <sup>64</sup> -1	64bit
浮点类型	REAL	1.401e-45~ 3.403e+38	32bit
	LREAL	2.2250738585072014e~308 - 1.7976931348623158e+308	64bit
字符串	STRING	只支持ASCII码字符（不支持中文字符）。默认最大长度80字符，超过最大长度会被去掉。可以声明字符最大长度，如str:STRING(35):=’ This is a String’；字符串函数最大支持255字符	ASCII形式存储字符串，用一个字节存储结束符
	WSTRING	只支持UNICODE字符（支持中文字符）。默认最大长度80字符，超过最大长度会被去掉。可以声明字符最大长度，如wstr:WSTRING(35):=” This is a WString”；	UNICODE形式存储字符串，用两个字节存储结束符
时间	TIME	时间常量，日，时，分，秒，毫秒	32bit, 内部按Dword处理
	TIME_OF_DAY(TOD)	一天的时间常量	32bit, 内部按Dword处理
	DATE	日期常量，从1970年1月1日开始	32bit, 内部按Dword处理
	DATE_ANI_TIME(DT)	日期和时间常量，从1970年1月1日开始	32bit, 内部按Dword处理

## 2. 用户自定义数据类型

用户自定义数据类型包括数组/结构体/枚举/联合/别名/子集/引用/指针。在中型PLC编程软件InoProShop中，可以通过右键应用-添加对象-DUT来添加结构体、枚举、联合和别名4种自定义数据类型。

### 数组

语法：<Array\_Name>:ARRAY [<ll1>..<ul1>,<ll2>..<ul2>,<ll3>..<ul3>] OF <elem. Type>

ll1, ll2, ll3 定义区域的下限，ul1, ul2 ul3 定义上限，数值必须为整数，elem. Type 为每个数组元素数据类型

### 初始化及示例

Card\_game: ARRAY [1..13, 1..4] OF INT;

arr1 : ARRAY [1..5] OF INT := [1,2,3,4,5];

arr2 : ARRAY [1..2,3..4] OF INT := [1,3(7)]; (\*数组值 1,7,7,7\*)

arr3 : ARRAY [1..2,2..3,3..4] OF INT := [2(0),4(4),2,3]; (\*数组值 0,0,4,4,4,4,2,3\*)

arr1 : ARRAY [1..10] OF INT := [1,2]; (\*数组部分初始化，没有初始化元素为默认值0\*)

### 数组结构初始化示例

结构定义:

```
TYPE STRUCT1
```

```
STRUCT
```

```
    p1:int;
```

```
    p2:int;
```

```
    p3:dword;
```

```
END_STRUCT
```

```
END_TYPE
```

数组结构初始化:

```
arr1:ARRAY[1..3] OF STRUCT1:=[(p1:=1,p2:=10,p3:=4723),(p1:=2,p2:=0,p3:=299),(p1:=14,p2:=5,p3:=112)];
```

访问联合元素语法:

```
<Array-Name>[Index1,Index2].
```

例子:

```
Card_game [9,2]
```

### 结构

语法:

```
TYPE <structurename> | EXTENDS DUTTYPE:
```

```
STRUCT
```

```
    <declaration of variables 1>
```

```
    ...
```

```
    <declaration of variables n>
```

```
END_STRUCT
```

```
END_TYPE
```

<structurename>是一个类型，可作为数据类型使用的。EXTENDS DUTTYPE 是可选项，表示继承了DUTTYPE中的成员，可以通过structurename类型变量访问DUTTYPE的成员。这里的DUTTYPE可以为结构类型，联合类型或者别名。

### 初始化及示例

Polygonline类型结构定义:

```
TYPE Polygonline:
```

```
STRUCT
```

```
    Start:ARRAY [1..2] OF INT;
```

```
    Point1:ARRAY [1..2] OF INT;
```

```
    Point2:ARRAY [1..2] OF INT;
```

---

```

Point3:ARRAY [1..2] OF INT;
Point4:ARRAY [1..2] OF INT;
End:ARRAY [1..2] OF INT;
END_STRUCT

```

END\_TYPE

初始化：

```
Poly_1:polygonline := ( Start:=[3,3], Point1:=[5,2], Point2:=[7,3], Point3:=[8,5], Point4:=[5,7], End:=[3,5]);
```

访问结构元素语法：

```
<structurename>.<variable>
```

例子：

```
Poly_1.Start
```

### 枚举

枚举类型是由若干串常量组成的，这些常量被称为枚举类型值。

语法：

```
TYPE <identifier>:(<enum_0>,<enum_1>,...,<enum_n>) |<base data type>;
```

```
END_TYPE
```

identifier:自定义的枚举类型； enum\_n：枚举类型对应的常量值，每个常量可以声明其对应值，如果不声明使用默认值； base data type枚举常量对应数据类型，可以不用声明，默認為整数。

---

### 说明

同枚举变量一在多个库中同时存在时，需添加库名前缀，否则编译报错。

---

### 联合

语法：

```
TYPE <unionname>:UNION
```

```
<declaration of variables 1>
```

```
...
```

```
<declaration of variables n>
```

```
END_UNION
```

```
END_TYPE
```

<unionname>是一个类型，并且可作为数据类型使用的。联合中的所有变量具有相同的存储位置，联合类型变量分配的空间大小为其中占用最大空间的变量分配的大小。

### 示例

```
TYPE union1: UNION
```

```
a : LREAL;
```

```
b : LINT;
END_UNION
```

END\_TYPE

访问数组元素语法：

<unionname>.<variable>

示例

union1.a

### 别名

把一种数据类型用一种别名来表示。

语法：

TYPE <aliasname>: basetype END\_TYPE

aliasname为别名类型名，用作数据类型。basetype可以是标准数据类型，也可以是用户自定义数据类型。

示例

TYPE alias1 : ARRAY[0..200] of Byte; END\_TYPE

初始化及访问方式和其对应的基本类型一致。

### 子集

子集数据类型是其定义的基本数据类型的子集，可以通过增加DUT来增加一个子集类型，也可以直接声明一个变量为子集类型。

DUT对象语法：

TYPE <name> : <Inttype> (<ug>..<og>) END\_TYPE;

name：有效的IEC标示符

Inttype：是数据类型SINT, USINT, INT, UINT, DINT, UDINT, Byte, WORD, DWORD (LINT, ULINT, LWORD)中的一种。

ug：是一个常数，必须符合基本类型对应的下边界范围。下边界本身包含在此范围内。

og：是一个常数，必须符合基本类型对应的上边界范围。上边界本身包含在此范围内。

DUT对象声明示例

TYPE

SubInt : INT (-4095..4095);

END\_TYPE

变量直接声明示例

VAR

i : INT (-4095..4095);

ui : UINT (0..10000);

END\_VAR

## 引用

引用是一个对象的别名，操作引用就如同操作对象。

语法：

<identifier> : REFERENCE TO <data type>

identifier：引用标示符。data type：引用对象的数据类型。

示例及初始化

```
ref_int : REFERENCE TO INT;
```

```
a : INT;
```

```
b : INT;
```

```
ref_int REF= a; (* ref_int引用a *)
```

```
ref_int := 12; (* a值为12 *)
```

```
b := ref_int * 2; (* b值为24 *)
```

```
ref_int REF= b; (* ref_int引用b *)
```

```
ref_int := a / 2; (* b值为6 *)
```

## 说明

不能引用BIT类型，即不允许定义ref1:REFERENCE TO BIT。

## 指针

指针保存的是一个对象的地址，指针可以指向任何数据类型（bit类型除外）

语法：

<identifier>: POINTER TO <data type>;

identifier：指针标示符。data type：指针指向的数据类型。

通过地址操作符对指针进行操作。地址操作符包括ADR（获取变量地址）和^（变量地址对应的值）

示例及初始化

VAR

```
pt:POINTER TO INT; (* 声明指向INT类型的指针pt*)
```

```
var_int1:INT := 5;
```

```
var_int2:INT;
```

END\_VAR

```
pt := ADR(var_int1); (* 变量varint1的地址分配给指针pt *)
```

```
var_int2:= pt^; (* 通过^地址操作符获取指针对应的值*)
```

```
pt^:=33; (*给指针对应的变量var_int1赋值*)
```

## 初始值

变量初始化默认值为0，用户可以在变量声明时通过赋值运算符“:=”来增加自定义初始化值。初始化值为有效的ST表达式。ST表达式由操作符，操作数，赋值表达式组成，操作符主要是加（+），减（-），乘（\*），除（/）等组成；操作数主要是指常量，变量和函数；赋值表达式指ST表达式中的赋值运算符“:=”。因此，初始化可以为常量，变量或者函数，只是要确保使用的变量已经被初始化。

示例：

```
VAR
var1:INT := 12; (* 整形变量初始值12*)
x : INT := 13 + 8; (*常量表达式定义初始值*)
y : INT := x + fun(4); (*初始值包含函数调用*)
z : POINTER TO INT := ADR(y); (*指针通过地址函数ADR来初始化*)
END_VAR
```

---

### 说明

- 全局变量表（GVL）一般在POU本地变量定义之前已经初始化；
  - 定义时初始化指针时，如果是在线修改默认，指针将不会被初始化（指针还是指向在线修改之前的变量）。
- 

### 5.3.3 变量类型

主要的变量类型包括：本地变量（VAR）、输入变量（VAR\_INPUT）、输出变量（VAR\_OUTPUT）、输入输出变量（VAR\_IN\_OUT）、全局变量（VAR\_GLOBAL）、临时变量（VAR\_TEMP）、静态变量（VAR\_STAT）以及配置变量（VAR\_CONFIG）。

变量类型声明语法：<type\_key> |attribute\_key

```
variable1;
variable2;
...
END_VAR
```

type\_key：类型关键字，包括VAR（本地变量），VAR\_INPUT（输入变量），VAR\_OUTPUT（输出变量），VAR\_IN\_OUT（输入输出变量），VAR\_GLOBAL（全局变量），VAR\_TEMP（临时变量），VAR\_STAT（静态变量），VAR\_CONFIG（配置变量）。

attribute\_key：属性关键字，包括RETAIN，PERSISTENT，CONSTANT，用于明确变量的范围。

---

### 说明

- 有关RETAIN，PERSISTENT变量的详细信息，请参见[第384页“5.5.3 掉电保持变量表”](#)；
  - 有关CONSTANT的详细信息，请参见[第380页“5.4 常量”](#)。
- 

### 本地变量（VAR）

在POU内部VAR和END\_VAR之间的变量都为本地变量，不能被外部访问。

赋值格式：

本地变量:=值

示例

VAR

iLoc1:INT; (\* 本地变量\*)

END\_VAR

## 输入变量 (VAR\_INPUT)

在POU内部VAR\_INPUT和END\_VAR之间的变量都为输入变量，可以在调用位置给输入变量赋值。

POU调用格式：

本地变量:=调用者输入值

示例

VAR\_INPUT

iIn1:INT; (\* 输入变量\*)

END\_VAR

### 说明

输入变量在POU内部也是可以修改的，即使增加了CONSTANT属性。

## 输出变量 (VAR\_OUTPUT)

在POU内部VAR\_OUTPUT和END\_VAR之间的变量都为输出变量。输出变量可以在调用时返回给调用者，调用者可以做进一步处理。

POU调用格式：

输出变量=>调用者匹配类型变量

示例

VAR\_OUTPUT

iOut1:INT; (\* 输出变量\*)

END\_VAR

### 说明

- 对于功能 (FUNCTION) 和方法 (METHOD) 除了返回值外，可以有另外的输出变量，但是必须在调用时分配调用者接收变量。例如fun(iIn1 := 1, iIn2 := 2, iOut1 => iLoc1, iOut2 => iLoc2);
- 对于功能块，可以在调用后把功能块输出变量赋值给调用者。

## 输入输出变量 (VAR\_IN\_OUT)

在POU内部VAR\_IN\_OUT和END\_VAR之间的变量都为输入输出变量。输入输出变量不仅可以传入被调用的POU内，并且可以在被调用的POU内部修改。实际上传递给被调用POU内的变量是调用者变量的引用。

## 示例

```
VAR_IN_OUT
    iInOut1:INT; (* 输入输出变量*)
END_VAR
```

## 说明

- 因传递给被调用POU内的变量是调用者变量的引用，所以不能直接访问功能块实例中的输入输出变量，也就是不能直接使用<FBinstance>.<InOutVariable>，因输入变量已经是调用者变量的引用，已经发生了变化；
- 输入输出变量不能是常量和Bit类型直接变量（如xBit0 AT %I2.0:BOOL）。如果需要声明输入输出常量，可以加CONSTANT属性（VAR\_IN\_OUT CONSTANT）。如果需要Bit类型直接变量，需要增加一个中间变量作为输入输出变量，然后把中间变量值给Bit类型直接变量。

Bit类型直接变量示例：

```
VAR_GLOBAL
    xBit0 AT %MX0.1 : BOOL;(*声明Bit类型直接变量*)
    xTemp : BOOL; (*中间变量*)

END_VAR
//带有输入输出变量(xInOut)的功能块
FUNCTION_BLOCK FB_Test
VAR_INPUT
    xIn : BOOL;
END_VAR
VAR_IN_OUT
    xInOut : BOOL;
END_VAR
IF xIn THEN
    xInOut := TRUE;
END_IF
//程序中调用功能块
PROGRAM Main
VAR
    xIn : BOOL;
    l1 : FB_Test;
    l2 : FB_Test;
END_VAR
//使用Bit类型的直接地址变量，编译报错
```

```
//I1(xIn:=xIn, xInOut:=xBit0);
//通过中间变量xTemp把xBit0的值传递给功能块，然后把中间变量赋值给xBit0
xTemp := xBit0;
I2(xIn:=xIn, xInOut:=xTemp);
xBit0 := xTemp;
```

输入输出常量(VAR\_IN\_OUT CONSTANT)只能读不能写，而输入变量在当前版本是能够修改的，即使增加了常量属性，所以可以用输入输出常量来把变量属性变为不可修改。

输入输出常量示例：

```
PROGRAM PLC_PRG
VAR
    sVarFits : STRING(16);
    sValFits : STRING(16) := '1234567890123456';
    iVar: DWORD;
END_VAR

POU(sReadWrite:='1234567890123456', scReadOnly:='1234567890123456', iVarReadWrite:=iVar);
//POU(sReadWrite:=sVarFits, scReadOnly:=sVarFits, iVarReadWrite:=iVar);
//POU(sReadWrite:=sValFits, scReadOnly:=sValFits, iVarReadWrite:=iVar);
//POU(sReadWrite:=sVarFits, scReadOnly:='23', iVarReadWrite:=iVar);
```

```
FUNCTION POU : BOOL
VAR_IN_OUT
    sReadWrite : STRING(16); (* 在此POU内此字符串可读可写 *)
    iVarReadWrite : DWORD; (*在此POU内此字此变量可读可写*)
END_VAR

VAR_IN_OUT CONSTANT
    scReadOnly : STRING(16); (*在此POU内此字符串只读的*)
END_VAR

sReadWrite := 'string_from_POU';
iVarInPOU := STRING_TO_DWORD(scReadOnly);
```

## 全局变量 (VAR\_GLOBAL)

定义在VAR\_GLOBAL和END\_VAR之间的变量为全局变量。一般变量，常量，保留变量都可以声明为全局变量。在AM600编程软件InoPro中可以通过右键应用-添加对象-添加全局变量表来添加全局变量表，然后在全局变量表中添加全局变量。

示例

---

```

VAR_GLOBAL
    iGlobVar1:INT; (* 全局变量*)
END_VAR

```

---

### 说明

- 本地变量如果和全局变量具有相同的名称，直接对此变量名进行操作时表示操作的为本地变量，可以在变量名前加全局范围操作符(.)来操作全局变量，如. iGlobVar1；
  - 全局变量总是在局部变量之前初始化。
- 

## 临时变量 (VAR\_TEMP)

定义在VAR\_TEMP和END\_VAR之间的变量为临时变量，临时变量在每次调用时会进行初始化。

示例

```

VAR_TEMP
    iTemp1:INT; (*临时变量*)
END_VAR

```

---

### 说明

- 临时变量只能在程序和功能块中声明；
  - 临时变量只能声明的程序或者功能块中使用。
- 

## 静态变量 (VAR\_STAT)

定义在VAR\_STAT和END\_VAR之间的变量都为静态变量。静态变量在第一次调用时被初始化，在每次调用完此POU后，变量值依然保持下来。

示例

```

VAR_STAT
    iStat1:INT; (*静态变量*)
END_VAR

```

---

### 说明

- 静态变量只能在功能块，功能和方法中声明，不能在程序中声明；
  - 静态变量只能声明的POU中使用。
- 

## 配置变量 (VAR\_CONFIG)

定义在VAR\_CONFIG和END\_VAR之间的变量都为配置变量。配置变量是直接变量，一般是映射到功能块定义的不确定地址直接变量。在功能块内可以定义一个不确定地址的变量，此变量的地址通过“\*”来表示不确定的地址（任意的地址），然后添加一个配置变量表（通过添加全局变量表方式），把所有功能块实例中不确定地址变量添加到配置变量表中，在此变量表中把所有的不确定地址给明确下来，这样就可以集中管理所有功能块中不确定地址变量。

功能块不确定地址变量定义语法：

<标识符> AT %<I|Q|M><sup>\*</sup> : <数据类型>

地址的最终确定是在全局变量列表的“变量配置”中完成：

#### 示例

```
FUNCTION_BLOCK locio
VAR
    xLocIn AT %I*: BOOL := TRUE;
    xLocOut AT %Q*: BOOL;
END_VAR
```

这里定义了两个 I/O 变量，一个本地输入变量 (%I\*) 和一个本地输出变量 (%Q\*)。

然后添加“全局变量列表”对象（GVL）。在关键词 VAR\_CONFIG 和 END\_VAR 之间输入实例变量声明的具体地址，这里的实例变量指包含 POU 完整的实例路径，具体地址对应于功能块中不确定指定的地址（%I\*，%Q\*），另外数据类型必须与功能块的声明一致。

配置变量定义语法：

<实例变量路径> AT %<I|Q|M><位置> : <数据类型>;

#### 示例

```
PROGRAM PLC_PRG
VAR
    locioVar1: locio;
    locioVar2: locio;
END_VAR

VAR_CONFIG (*正确的变量配置表*)
PLC_PRG.locioVar1.xLocIn AT %IX1.0 : BOOL;
PLC_PRG.locioVar1.xLocOut AT %QX0.0 : BOOL;
PLC_PRG.locioVar2.xLocIn AT %IX1.0 : BOOL;
PLC_PRG.locioVar2.xLocOut AT %QX0.3 : BOOL;
END_VAR
```

#### 说明

- 一般是不需要配置变量的，因为对于 I/Q 地址输入/输出可以在对应模块的 I/O 映射界面通过输入助手（或者直接输入实例变量路径）把变量映射到 I/Q 地址；
- 配置变量一般映射到功能块中不确定地址变量，它也可以映射程序中的不确定地址变量；
- 只存在不确定地址变量或者只存在配置变量都会编译报错，两者是配合使用。

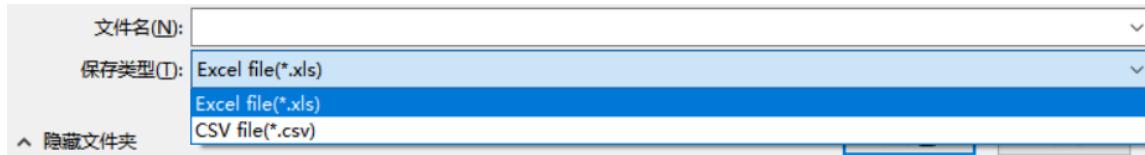
### 5.3.4 变量导入与导出

支持变量导入与导出，导出文件类型为“XLS 工作表 (.xls)”，呈现为 Excel 表格形式，可在外部进行增加、删除或其他对变量的编辑后再导入到 InoProShop 编程软件。

如下图所示。

	类别	名称	地址	数据类型	初值	保持	常量	注释	特性
1	VAR_GLOBAL RETAIN PERSISTENT	A_0		BOOL		<input checked="" type="checkbox"/>	<input type="checkbox"/>	变量A	
2	VAR_GLOBAL CONSTANT	A_1		INT		<input type="checkbox"/>	<input checked="" type="checkbox"/>		
3	VAR_GLOBAL	A_2		BYTE		<input type="checkbox"/>	<input type="checkbox"/>		
4	VAR_GLOBAL	A_3		WORD	100	<input type="checkbox"/>	<input type="checkbox"/>		
5	VAR_GLOBAL	A_4		DWORD		<input type="checkbox"/>	<input type="checkbox"/>		
6	VAR_GLOBAL	A_5		ARRAY[0..9] OF REAL		<input type="checkbox"/>	<input type="checkbox"/>	AAA	

在变量表中增加部分变量，并右键选择导出变量表类型Excel/CSV（CSV是纯文本文件，excel包含有格式信息。CSV文件较小，创建分发读取较方便，适合存放结构化信息。CSV文件在windows平台默认的打开方式是excel，本质是文本文件。），两种格式文件对变量的编辑没有区别。



打开导出的文件，并编辑（添加新变量A\_6、A\_7、A\_8、A\_9）后导入，效果如下图所示。

Type	Name	Address	DataType	InitValue	Comment	Attribute
VAR_GLOBAL RETAIN PERSISTENT	A_0		BOOL		变量A	
VAR_GLOBAL CONSTANT	A_1		INT			
VAR_GLOBAL	A_2		BYTE			
VAR_GLOBAL	A_3		WORD	100		
VAR_GLOBAL	A_4		DWORD			
VAR_GLOBAL	A_5		ARRAY [0..9] OF REAL			
VAR_GLOBAL RETAIN PERSISTENT	A_6		BYTE			
VAR_GLOBAL CONSTANT	A_7		WORD			
VAR_GLOBAL	A_8		DWORD	200		
VAR_GLOBAL	A_9		ARRAY [0..9] OF REAL		变量B	

	类别	名称	地址	数据类型	初值	保持	常量	注释
1	VAR_GLOBAL RETAIN PERSISTENT	A_0		BOOL		<input checked="" type="checkbox"/>	<input type="checkbox"/>	变量A
2	VAR_GLOBAL CONSTANT	A_1		INT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
3	VAR_GLOBAL	A_2		BYTE		<input type="checkbox"/>	<input type="checkbox"/>	
4	VAR_GLOBAL	A_3		WORD	100	<input type="checkbox"/>	<input type="checkbox"/>	
5	VAR_GLOBAL	A_4		DWORD		<input type="checkbox"/>	<input type="checkbox"/>	
6	VAR_GLOBAL	A_5		ARRAY [0..9] OF REAL		<input type="checkbox"/>	<input type="checkbox"/>	
7	VAR_GLOBAL RETAIN PERSISTENT	A_6		BYTE		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
8	VAR_GLOBAL CONSTANT	A_7		WORD		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
9	VAR_GLOBAL	A_8		DWORD	200	<input type="checkbox"/>	<input type="checkbox"/>	
10	VAR_GLOBAL	A_9		ARRAY [0..9] OF REAL		<input type="checkbox"/>	<input type="checkbox"/>	变量B

## 5.4 常量

在PLC编程的时候，会用到一些数值不变的参数，如定时器的时间、换算的比例等，这些数值不变的参数称为常量。

常量声明语法：

VAR CONSTANT

<identifier>:<type> := <initialization>;

END\_VAR

## 示例

VAR CONSTANT

```
c_iCon1:INT:=12;
```

END\_VAR

CoDeSys支持多种数据类型的常量，常见的常量包括布尔型、整形、时间型和字符串等。具体常量见下表。

类型	描述	示例
布尔类型	有两个值TRUE和FALSE（也可以用1和0），1表示TRUE，0表示FALSE	TRUE, FALSE, 1
BIT类型	和布尔类型相似，只能在结构体（占用位数）或者功能块（映射BOOL类型的直接地址）中使用	TRUE, FALSE, 0
整数	整数常量的数值可以是二进制、十进制、八进制和十六进制。如果整数值不是十进制值，可以用“进制”加符号“#”放在整数值前面来表示。十进制的10至15在十六进制中表示为A至F	十进制：66 二进制：2#101 八进制：8#72 十六进制：16#3A 类型常数： INT#22 BYTE#204
浮点类型	浮点常量用十进制小数和指数来表示，遵循标准的科学计数法格式	7.4 2.3e+9 REAL#3.12
ASCII字符串	ASCII字符串常量在两个单引号之间，可以包含空格和特殊字符。一个字符用一个字节表示，只支持ASCII码字符（不支持中文字符）。默认最大长度80字符，超过最大长度会被去掉。可以声明字符最大长度，如str: STRING(35):=' This is a String'；字符串函数最大支持255字符。	\$做转义字符例： '\$30' : 0, 字符0, 16进制30对应的ASCII字符 \$\$: \$, 美元字符 \$: ', 单引号
UNICODE字符串	UNICODE字符串常量在两个双引号之间，一个字符占两个字节，只支持UNICODE字符（支持中文字符）。默认最大长度80字符，超过最大长度会被去掉。可以声明字符最大长度，如wstr:WSTRING(35):=" This is a WString"；	“Unicode string”
时间	时间常量一般用来操作时间，由“T#”（或“t#”）加上“时间值”构成，时间值的单位包括天（d）、小时（h）、分（m）、秒（s）和毫秒（ms）。	t#12h34m15s;
时刻	一天的时间范围，语法：TOD#时间值。	TOD#15:36:30.123
日期	从1970年1月1日开始，语法：d#日期。	d#2015-02-12
日期时刻	日期常量和时刻常量合并起来称为日期时刻常量，由从1970年1月1日开始，语法：dt#日期。	dt#2004-03-29-11:00:00

## 说明

除BOOL、BIT和字符串类型外，其他类型都可以用关键字#常量值来表示某一类型常数。

## 5.5 掉电保持变量

### 5.5.1 概述



仅在编译器3.5.11.71及以上版本支持掉电保持变量功能。可通过菜单栏选择“工程 > 工程设置”打开“工程设置”对话框，在“编译选项”界面查询编译器版本。

掉电保持变量可在PLC掉电或程序下载后继续保留原来的值。该变量用来定义工程中重要的参数，防止PLC突发掉电或者程序下载而导致的重要参数丢失。

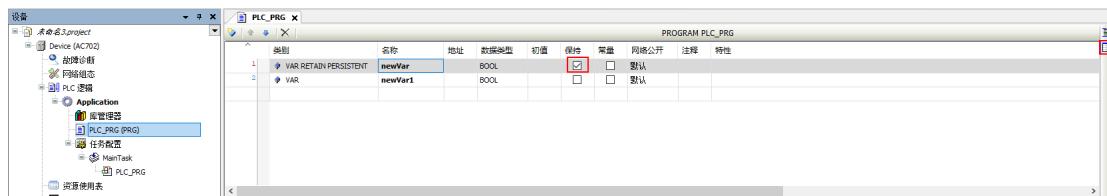
### 5.5.2 变量定义

#### 定义变量

掉电保持变量支持在全局变量表（GVL）、程序（PRG）、功能块（FB）、函数（FUN，仅限静态类别变量）中进行定义，不支持在方法（METH）、属性（Prop）、结构体（STRUCT）、联合体（Union）、枚举（Enum）、别名（Alias）中进行定义，可通过表格和文本两种方式进行定义，以在程序（PRG）中定义为例。

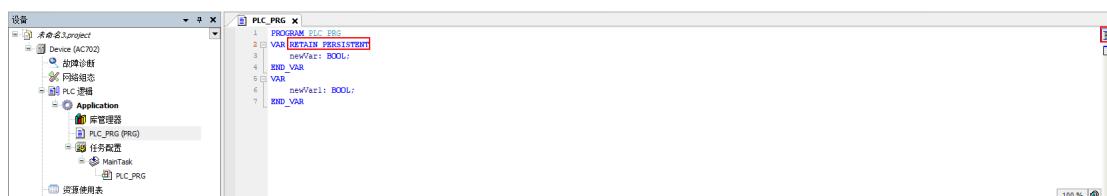
- 表格方式

在“PLC\_PRG(PRG)”编程界面表格方式下，勾选“保持”列中的复选框，定义该变量为掉电保持变量。



- 文本方式

在“PLC\_PRG(PRG)”编程界面文本方式下，通过对变量增加“RETAIN PERSISTENT”或“PERSISTENT RETAIN”关键字，定义该变量为掉电保持变量。



#### 变量类别

掉电保持变量支持设置为本地变量、输入变量、输出变量、静态变量等变量类别，不支持设置为输入输出变量、临时变量、常量、配置变量等变量类别。

以在程序（PRG）中表格方式设置变量类别为例，双击变量行对应的“类别”列，在打开的界面中设置支持的变量类别。

PROGRAM PLC_PRG									
类别	名称	地址	数据类型	初值	保持	常量	网络公开	注释	特性
VAR RETAIN PERSISTENT	newVar		BOOL		<input checked="" type="checkbox"/>	<input type="checkbox"/>	默认		
	newVar1		BOOL		<input type="checkbox"/>	<input type="checkbox"/>	默认		

## 变量类型

掉电保持变量不支持定义为指针类型、引用类型、功能块（FB）类型整体（包括嵌套的场景，如嵌套指针类型、嵌套引用类型、嵌套功能块类型），支持定义为除此之外的其他类型。

## 变量映射地址

定义变量为掉电保持变量后，编译后将自动生成一个映射至M区的地址，该地址支持手动编辑。

类别	名称	地址	数据类型	初值	保持	常量	网络公开	注释	特性
VAR RETAIN PERSISTENT	newVar	%MB131072	BOOL		<input checked="" type="checkbox"/>	<input type="checkbox"/>	默认		
VAR	newVar1		BOOL		<input type="checkbox"/>	<input type="checkbox"/>	默认		

```

PROGRAM PLC_PRG
2 VAR RETAIN PERSISTENT
3 newVar AT %MB131072 : BOOL;
4 END_VAR
5 VAR
6 newVar1: BOOL;
7 END_VAR

```



掉电保存变量只支持设置为M区地址，不支持设置为非M区地址，例如不支持I区和Q区。

同时在设备树中原来如果无“PersistentVars”对象节点，则将生成“PersistentVars”对象节点；原来如果有“PersistentVars”对象节点，则将更新内部掉电保持变量，并将工程中所有的掉电保持变量收集到该对象视图中。

The screenshot shows the Device tree on the left with a project named "未命名2.project". Under "PLC 逻辑", there is an "Application" folder which contains a "PersistentVars" node. On the right, there is a "PersistentVars" configuration window with tabs for "刷新" (Refresh), "掉电存储区" (Power-off Memory), "清零存储数据" (Clear Memory Data), and "修复冲突地址" (Repair Conflict Address). A note at the top says: "编译/生成代码成功后，变量节点才可以正常展开！" (After compilation/generation of code is successful, the variable node can be normally expanded!). Below the tabs is a table:

名称	地址	类型	初始值	注释
PLC_PRG.newVar	%MB131072	BOOL		

## 变量响应动作

执行复位、掉电等动作时不同掉电保持变量的响应动作如下表所示。

动作	VAR	VAR PERSISTENT RETAIN或者 VAR RETAIN PERSISTENT	VAR RETAIN
掉电	初始化	保持原值	保持原值
热复位	初始化	保持原值	保持原值
冷复位	初始化	保持原值	初始化
初始值复位	初始化	初始化	初始化
程序下载	初始化	保持原值	初始化
在线修改	保持原值	保持原值	保持原值

## 说明

- RETAIN变量和PERSISTENT RETAIN变量都属于保持变量，但保持特性不同；
- 映射到%M地址的直接变量可以声明为掉电保持变量，而映射到%I和%Q的直接变量不能声明为掉电保持变量。

### 5.5.3 掉电保持变量表

如果工程中定义了掉电保持变量，则必须生成一个掉电保持变量表，否则定义的变量不具有掉电保持功能。掉电保持变量表可通过以下两种方式生成：

- 手动添加：通过右键“应用” - “添加对象” - “掉电保持变量”来添加掉电保持变量表；
- 自动添加：当声明了掉电保持变量，编译时会自动创建掉电保持变量表。

掉电保持变量表包含两种模式：传统模式和标准模式（推荐使用）。传统模式即旧模式，具体使用与原来保持不变。标准模式下掉电保持变量表如下图：

The screenshot shows the '掉电存储区' configuration window. At the top, there are buttons for 刷新 (Refresh), 掉电存储区 (Power Failure Memory), 清零存储数据 (Clear Memory Data), and 修复冲突地址 (Repair Conflict Address). A message box says '存在多结构变量时, 需要生成代码来保证子变量地址正确分配!' (When multiple structured variables exist, code needs to be generated to ensure correct allocation of sub-variable addresses!). Below is a table:

名称	地址	类型	初始值	注释	配方	配方1
PLC_PRG.persData	%MB131072	INT	INT#1		1	2
PLC_PRG.persData2	%MB131074	STRING	'666'		'1222'	'222'
PLC_PRG.persData3	%MX131155.0	BOOL			FALSE	TRUE
PLC_PRG.persData4	%MB131156	WORD			2555	4555
PLC_PRG.persData5	%MB131158	ARRAY [1..100] OF BOOL				
PLC_PRG.persData5	%MB131158	BOOL			TRUE	TRUE
PLC_PRG.persData5	%MB131159	BOOL			FALSE	TRUE
PLC_PRG.persData5	%MB131160	BOOL			TRUE	TRUE
PLC_PRG.persData5	%MB131161	BOOL			TRUE	TRUE
PLC_PRG.persData5	%MB131162	BOOL			FALSE	TRUE
PLC_PRG.persData5	%MB131163	BOOL			FALSE	FALSE
PLC_PRG.persData5	%MB131164	BOOL			FALSE	TRUE
PLC_PRG.persData5	%MB131165	BOOL			TRUE	FALSE
PLC_PRG.persData5	%MB131166	BOOL			FALSE	TRUE

其中，工具栏选项的功能如下：

选项	功能	说明
刷新	将外部工程的持久性变量添加到持久性变量表中，并对未进行地址分配的持久性变量进行地址分配 刷新变量配方数据结构 检测所有掉电保持变量地址合法性	-
掉电存储区	实现模式切换和标准模式存储区域地址范围设置	区域地址范围设置请参见 <a href="#">第388页“地址区域设置”</a>
清零存储数据	用于在在线状态下清零掉电保持变量在设备内存中的数据	-
修复冲突地址	对掉电保持变量的冲突地址进行重新分配	-

列表条目说明如下：

条目	功能	属性
名称	显示掉电保持变量的来源和变量名	不可编辑
地址	显示掉电保持变量的地址	可编辑
类型	显示掉电保持变量的类型	不可编辑
初始值	显示掉电保持变量声明时的初始值 选择初始值列，执行菜单命令“当前值->初始值”命令时，掉电保持变量在线值会被写入初始值列，并同步到工程中的掉电保持变量声明中	不可编辑
注释	显示掉电保持变量的注释信息	不可编辑
配方	显示并保存掉电保持变量配方值 用户可按需添加	可编辑

## 5.5.4 掉电保持规则



### 注意

AC700/AC800系列1.26.14.0及以上固件版本和AM400/AM600系列1.40.8.0及以上固件版本才支持该规则。

掉电保持变量相关属性变化时变量值的规则如下：

- 变量名变化时，掉电保持变量的值将被初始化为初始值。
- 变量地址、初始值、注释、变量类别和特性发生变化时，掉电保持变量的值将保持原来的值。
- 变量类型变化，包括变量的类型名称变化、变量的类型成员变化、变量的类型成员类型变化。
  - 掉电保持变量的类型名称变化时，掉电保持变量的值将被初始化为初始值。

### 说明

在FB功能块成员名称、类型和实例对象名称未变化的前提下，修改FB功能块的类型名称，FB成员变量的值将保持原来的值。

- 掉电保持变量的类型名称未发生变化，但类型成员发生变化时：
  - 结构体/功能块：没有变化的成员的值将保持原来的值，变化的成员的值将被初始化为初始值。
  - 联合体/枚举：当其类型大小减少时将被初始化为初始值。

- 数组：当数组类型变量的基类型发生变化时，数组变量将被初始化为初始值（数组类型变量的基类型为结构体且仅增删改结构体成员时，没有变化的结构体成员将保持原来的值）；当数组类型变量的基类型未发生变化时，相同索引的元素值将保持原来的值。
- 当掉电保持变量引起的拷贝数据超过20万行代码时，将导致拷贝的掉电保持变量的值被初始化为初始值。

### 5.5.5 掉电保持模式

#### 模式对比

为了保持工程与软件版本的兼容性，工程中保留了掉电保持变量旧编辑模式，即传统模式。

当打开的旧工程含有“持久变量”时，系统仍可按传统模式来显示和编辑工程；如果原工程不存在“掉电保持变量”，则新建的掉电保持变量将遵循“标准模式”。

目前掉电保持变量包含标准模式和传统模式（旧模式），二者的差异对比如下表所示：

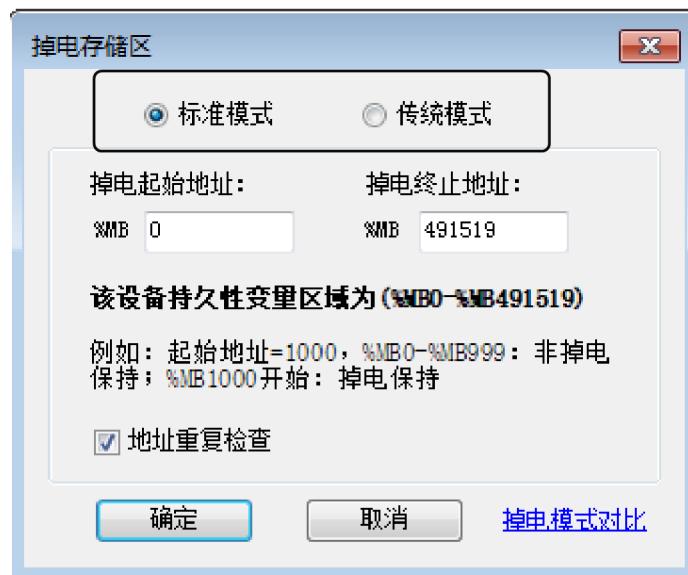
选项	标准模式	传统模式（旧模式）
数据来源	全部来源于掉电保持变量表外部带有 Persistent Retain特性或分配到M区掉电保存区范围的变量	掉电保持变量表内部定义变量和外部带 Persistent Retain特性变量
地址映射	所有的变量均需强关联到M区掉电保存区	不需强关联
中间插入或删除变量	由于变量是通过地址映射，因此插入或删除变量对其他变量保存值无影响	意外操作可能引起数据丢失（例：清除全部、PLC设备更新，编译选项修改等）
配方功能	内部融合配方功能	只能使用外部独立配方功能

#### 说明

仿真模式下，如果掉电保持变量为标准模式，执行“冷复位”命令，变量值将会被初始化。

#### 传统模式切换标准模式

掉电保持变量表可以通过工具栏“掉电存储区”选项进行模式切换。界面如下图所示：



“传统模式”切换为“标准模式”时，传统模式中自定义的掉电保持变量会保存到新生成的GVL程序中。重新编译或点击刷新时，系统会将GVL中的掉电保持变量添加到标准模式数据表中，并为其分配地址。

## 说明

如果用户重新切换回“传统模式”，原数据保持不变。但新生成的GVL程序不会被删除。因此，在切回“传统模式”时，需手动删除新生成的GVL程序。

## 模式兼容性处理

对于掉电保持模式，PLC固件和工程版本的兼容性存在以下两种情况：

- 如果PLC支持掉电保持模式切换，下载工程后，PLC自动切换为与工程设置相同的保持模式。

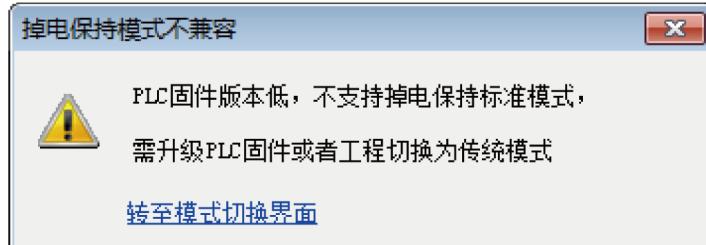
## 说明

若工程版本和PLC模式不一致，第一次下载工程时PLC会自动切换掉电保持模式，可能导致数据不符合预期。

支持掉电保持模式的PLC固件版本列表如下：

PLC型号	版本限制
AM600	1.24.20.0（含）以上版本
AM401	21.24.20.0（含）以上版本
AM402	41.24.20.0（含）以上版本
AM403	81.24.20.0（含）以上版本
AP700	1.13.30.0（含）以上版本
AC810/AC801/AC802	1.13.30.0（含）以上版本

- 如果PLC不支持掉电保持模式切换，则需要升级PLC固件或者将工程掉电保持模式切换为传统模式。



转至模式切换界面：进入掉电保持变量表，并打开模式切换界面。

## 5.5.6 地址分配

若用户程序中定义了掉电保持变量，但并未给变量分配地址，在标准模式下，只需单击工具栏中的“编译”或掉电保持界面的“刷新”，系统会自动为其分配地址。

考虑用户Modbus地址使用范围，在初次分配地址时会避开其常规使用区域（%MB0~%MB131071），以%MB131072开始，向后分配；只有当尾部地址使用完或者无法再为变量分配地址，再从%MB0~%MB131071范围从头向后分配。



例如，当前可用最大地址为%MB50000，要为Real型变量Var进行地址分配：

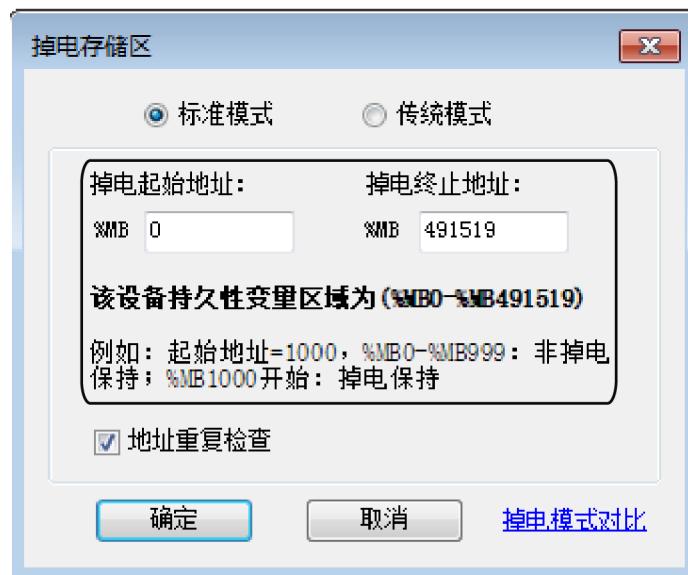
由于Real类型变量空间大小为4Byte，起始地址计算方法为：首先在“%MB131072~%MB50000”范围内开始选择地址，首先选取“%MB131072”为起始地址，再与现有分配的地址进行冲突检测，如果发生地址冲突，则抛弃该地址，从下一个可用地址再次重新计算地址，再次检测，直至地址不发生冲突为止；如果在“%MB131072~%MB50000”范围内找不到一块完整的区域保存该变量，则开始在“%MB0~%MB1310721”范围内开始查找，选取“%MB0”为起始地址，同样按照前边方式进行冲突检测，直至找到合法地址为止。

在对变量进行地址分配时，有几种数据类型需要满足四字节对齐原则，即变量初始地址能够被4整除。变量类型如包括：

- 自定义数据结构。
- 枚举类型。
- Real或LReal类型。
- 基类型数据为上述三种类型的数组Array类型。

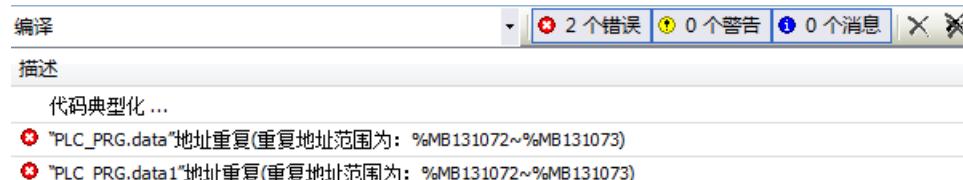
## 地址区域设置

地址分配区域为PLC的M区，具体的地址范围可以通过工具栏中“掉电存储区”选项查看和设置。如下图所示：

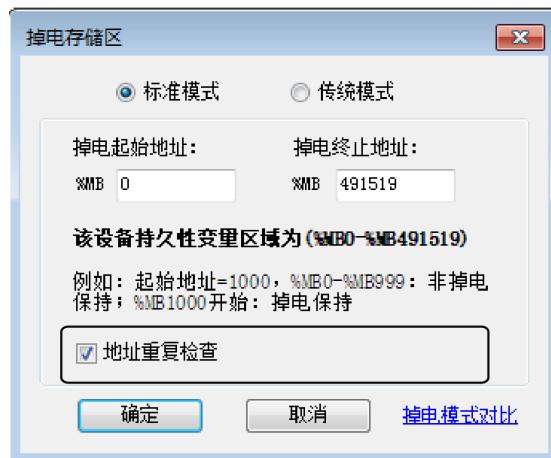


## 地址重复检查

标准模式下，为了保证掉电保持变量数据保存的正确性，原则上每个变量应该对应唯一的M区地址，默认情况下，系统会对所有掉电保持变量进行地址重复检查，当两个或两个以上变量地址发生冲突，会在输出窗口进行错误提示。



如果用户存在需求，变量可以地址重复，那么用户可以通过工具栏中“掉电存储区”选项中去除默认勾选项“地址重复检查”即可。如下图所示。



## 修复冲突地址

当掉电变量地址发生了地址冲突，用户可以通过手动修改变量地址的方法，自助对冲突地址进行调整。除此之外，也可以通过工具栏中“修复冲突地址”选项来对冲突地址进行修复。界面如下图所示。



表格中显示了所有存在地址冲突的变量。表格每列分别显示变量的名称、地址、类型、地址范围和建议修改的地址。

其中，建议地址是针对单个变量进行的地址修复功能。如果用户想要一次性修复，可以单击“一键修复”来整体修复地址。

## 批量删除变量地址

当需要调整掉电保持区域范围或因为地址冲突太多，想要对所有变量地址进行重新分配。可以通过右键菜单“清除全部变量地址”选项，来清除全部变量地址，再通过编译对所有变量进行地址重新分配。

### 说明

当掉电保持变量数据已下载到PLC中，无论是手动还是通过地址修复等操作对PLC中已存在的掉电保持变量进行地址修改，都会导致其原来地址保存的数据无法被使用。同时，如果新分配地址存在数据，那么该数据会被应用在新关联的变量上，可能出现无法预知的问题。为了保证后续数据的有效性，均需对初始值进行初始化操作。

需要调整地址的情况如下：

- 数组大小变化

```
VAR PERSISTENT RETAIN
    dataArray AT %MB131072 : ARRAY [1..100] OF INT;
    data AT %MX131272.0 : BOOL;
END_VAR
```

数组变量dataArray最初范围为1~100，分配地址为%MB131072，bool类型变量data分配地址为%MX131272.0，两个变量的地址是连续的。当因为实际需要，dataArray变量的范围调整为1~200，起始地址并未改变。

```
VAR PERSISTENT RETAIN
    dataArray AT %MB131072 : ARRAY [1..200] OF INT;
    data AT %MX131272.0 : BOOL;
END_VAR
```

此时dataArray变量的地址范围为%MB131072~%MB131471，与data变量的地址%MX131272发生了重复。

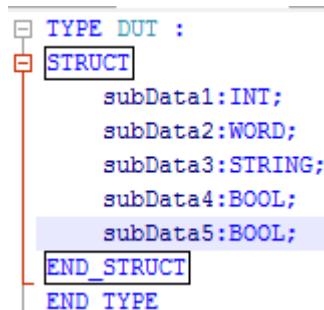
- 数据结构成员变化

定义数据结构变量dataDUT，其中DUT的数据结构如下图所示：

```
TYPE DUT :
STRUCT
    subData1:INT;
    subData2:WORD;
    subData3:STRING;
END_STRUCT
END_TYPE

VAR PERSISTENT RETAIN
    dataDUT AT %MB131072 : DUT;
    dataBool AT %MX131158.0 : BOOL;
END_VAR
```

最初DUT中包含三个成员，起始地址为%MB131072，当后续程序调整，DUT数据成员中增加两个成员变量，变量dataDUT实际占用空间变大，与后续dataBool变量地址发生冲突。



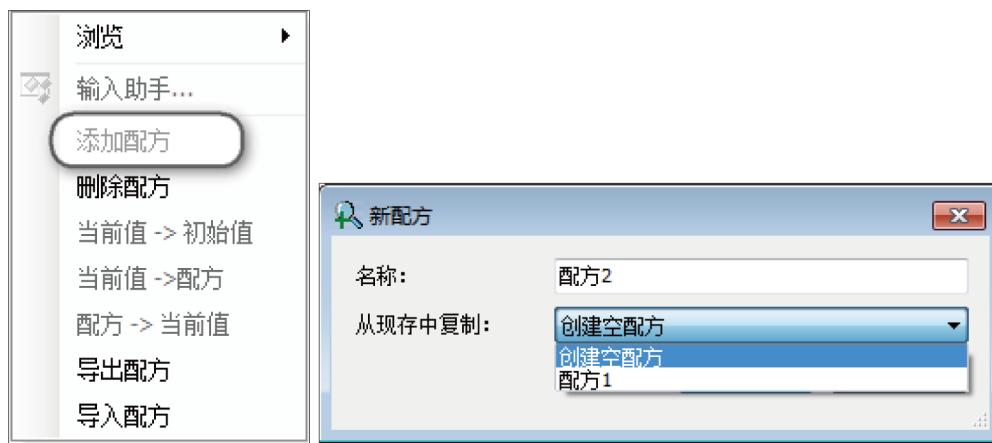
## 5.5.7 配方操作

配方可以保存一组变量的数据，作为掉电保持变量写入值。

在掉电保持标准模式下，用户可以在掉电保持变量表中右键单击“配方”条目来执行“添加配方”、“删除配方”、“当前值->配方”、“配方->当前值”、“导出配方”和“导入配方”等操作。

### 添加配方

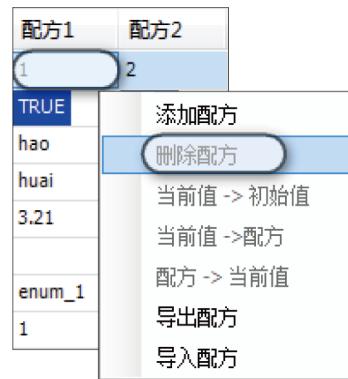
在掉电保持变量表中右键选择“添加配方”：



- 功能：此操作在掉电保持变量表最后一列后添加新的配方列。在弹出的“新配方”设置对话框中设置新配方列的名称，也可选择从已有的配方列中复制。
- 使能条件：存在标准模式的掉电保持变量表。
- 注意事项：首次添加配方时，需要先执行“刷新”，否则会弹框提示进行刷新。

## 删除配方

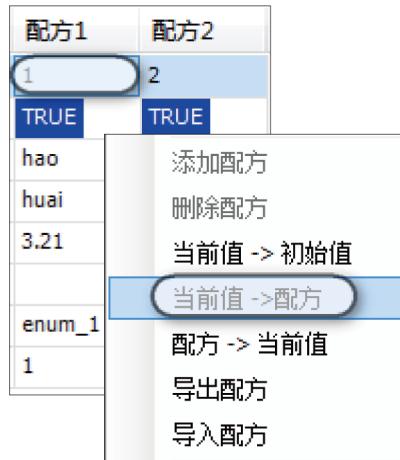
在掉电保持变量表中选中配方列，右键单击并选择“删除配方”：



- 功能：此操作将删除鼠标所选中的配方列。
- 使能条件：存在标准模式的掉电保持变量表，且鼠标选中要删除的配方列。

## 当前值->配方

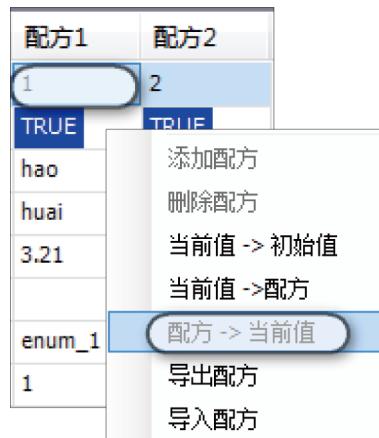
在掉电保持变量表中选中配方列，右键单击并选择“当前值->配方”：



- 功能：此操作将掉电保持变量的在线值保存到鼠标所选的配方列。
- 使能条件：登录工程，存在标准模式的掉电保持变量表，且鼠标选中对应的配方列。
- 注意事项：初始值列和配方列的数据结构要一致，才可以正常执行该命令，否则会弹出对话框提示刷新。

## 配方->当前值

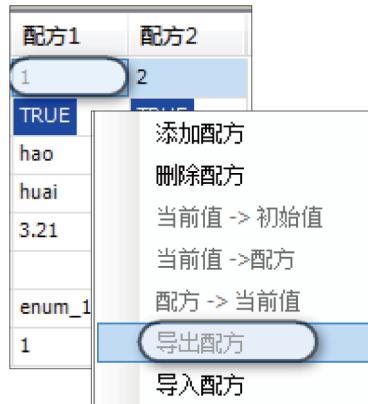
在掉电保持变量表中选中配方列，右键单击并选择“配方->当前值”：



- 功能：此操作将鼠标所选配方列中的掉电保持变量配方值写入在线值列。
- 使能条件：登录工程，存在标准模式的掉电保持变量表，且鼠标选中对应的配方列。
- 注意事项：初始值列和配方列的数据结构要一致，才可以正常执行该命令，否则会弹出对话框提示刷新。

## 导出配方

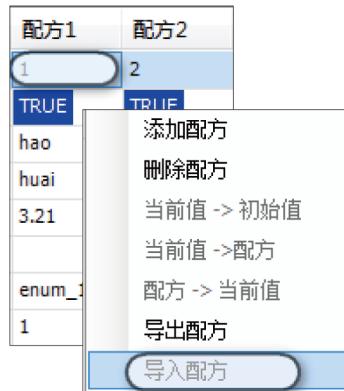
在掉电保持变量表中选中配方列，右键单击并选择“导出配方”：



- 功能：此操作将鼠标所选配方列的值导出到后缀为“.txtrecipe”的新文件中，并将该文件保存到指定位置。
- 使能条件：存在标准模式的掉电保持变量表，且鼠标选中对应的配方列。

## 导入配方

在掉电保持变量表中选中配方列，右键单击并选择“导入配方”：



- 功能：此操作将指定位置中所选文件（后缀为“.txtrecipe”）的配方值导入到鼠标选中的配方列。
- 使能条件：存在标准模式的掉电保持变量表，且鼠标选中对应的配方列。
- 注意事项：导入配方中的变量需包含在掉电保持变量表中，否则系统会提示不匹配。

## 5.5.8 使用说明

### 标准模式

标准模式下，当变量结构发生变化时，当变量起始地址%MB131072时（Modbus常用地址范围为%MB0~%MB131071），系统会保存原结构掉电保持变量已存在数据，对新增变量则会进行初始化操作。例如：

定义数据结构掉电变量dtData，如下所示。

```
VAR_GLOBAL PERSISTENT RETAIN
  dtData AT %MB131072 : DUTData;
END_VAR
```

变量dtData的数据结构如下所示。

```

TYPE DUTData :
STRUCT
    perData:INT;
    perData1:BOOL;
    perData2:WORD;
END_STRUCT
END_TYPE

```

将工程下载到PLC中后，系统在掉电时，会对变量dtData的数据进行存储。

当在DUTData中新增一个BOOL类型变量perData3，初次下载最新程序，该结构变量的数据变化为。

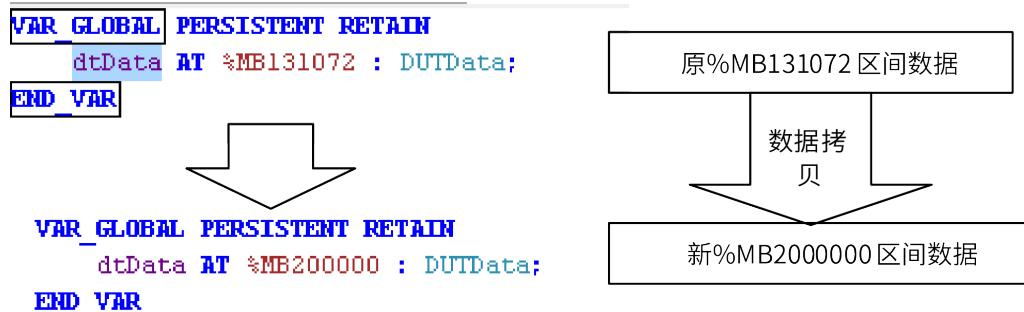
```

TYPE DUTData :
STRUCT
    perData:INT;
    perData1:BOOL;
    perData2:WORD;
    perData3:BOOL; ] 原掉电数值保持
    ] 首次下载，数值初始化
END_STRUCT
END_TYPE

```

## 变量地址变化

当掉电保持变量的地址发生变化，系统会将原地址数据拷贝到新地址内存个中，不会影响掉电保持变量数据变化。



## FB掉电保持变量

FB类型变量不支持具有掉电保持特性，但FB子成员变量可以具有掉电保持特性。

如下功能块FBData中，子成员变量fbData、fbData1、fbData2可以具有PERSISTENT RETAIN特性。但FBData类型变量fbVar不允许具有PERSISTENT RETAIN特性。

```

FUNCTION_BLOCK FBData
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR PERSISTENT RETAIN
    fbData:INT;
    fbData1:BOOL;
    fbData2:STRING;
END_VAR
VAR_GLOBAL
    fbVar:FBData;
END_VAR

```

# 6 编程语言

## 6.1 InoProShop 支持的编程语言简介

编程软件支持下列PLC编程语言：

- Ladder diagram(LD) 梯形图
- Function block diagram(FBD) 功能块图
- Structured text (ST) 结构化文本
- Sequential function chart (SFC) 顺序功能图
- Continuous function chart (CFC) 连续功能图

其中，LD、FBD、ST、SFC是基于IEC 61131-3标准，CFC是IEC 61131-3标准的一个扩展。

不论用户选用哪种语言，编程界面中的基本编辑方法是通用的，为编程带来很大方便。

- 标准的编辑器功能，如支持“复制”（Ctrl+C）、“粘贴”（Ctrl+V）和“删除”（Del）等快捷键；
- 标准的<Ctrl>、<Shift>按键多项选择；
- 支持功能键<F2>启动输入助手，系统根据具体环境提供相对应的输入提示或选择。

## 6.2 结构化文本语言(ST)

### 6.2.1 概述

结构化文本是一种文本化的高级语言，跟PASCAL或C类似。程序代码由指令组成，指令由关键字和表达式组成。不同于IL语言，ST语句循环中可以包含众多的语句，因此允许开发复杂的结构。

例如：

```
IF value < 7 THEN
    WHILE value < 8 DO
        value := value +1;
    END_WHILE;
END_IF;
```

### 6.2.2 表达式

表达式是一种结构，对它求值后，这个值可以在指令中使用。

表达式由操作符和操作数组成。一个操作数可以是一个常量，变量，功能调用或其他表达式。例如：

- 常量，例如：20, t#20s, 'string'
- 变量，例如：iVar, Var1[2,3]
- 功能调用，值为调用返回值，例如：Fun1(1,2,4)
- 其它表达式：10+3, var1 OR var2, (x+y)/z, iVar1:=iVar2+22

表达式的求值以特定的操作符优先权定义的顺序，按操作符对操作数进行求值。表达式中具有最高优先权的操作符应首先进行求值，接着是下一个较低优先权的操作符等，从高到低依次求值完成。优先权相等的操作符应按表达式中书写的从左到右的顺序进行。

例子：若A、B、C和D属于类型INT，并分别具有值1、2、3、4，那么 $A+B-C*ABS(D)$ 应等于-9，而 $(A+B-C)*ABS(D)$ 应等于0。

当操作符具有两个操作数时，应首先对最左边的操作数求值。例如，在表达式 $SIN(A)*COS(B)$ 中，应先对表达式 $SIN(A)$ 求值，其次是对 $COS(B)$ 求值，最后是积的求值。

下表记录了ST语言的操作符：

操作	符号	优先权
括号	(表达式)	
函数调用	函数名(参数列表，由逗号分隔)	
求幂	EXPT	
求负值	-	
求补	NOT	
乘	*	
除	/	
取余	MOD	
加	+	
减	-	
比较	<,>,<=,>=	
等于	=	
不等于	<>	
逻辑与	AND	
逻辑异或	XOR	
逻辑或	OR	

高  
↓  
低  
依次降低

### 6.2.3 ST 指令

整个ST程序由指令构成，指令由分号“；”分隔。这些指令由关键字和表达式组成，ST指令如下表。

指令	说明	示例
$:=, S=, R=$	赋值，置位，复位	$A:=B; C S= cond0; b1 R=cond1;$
功能块调用	功能块调用和输出	$CMD\_TMR:TON$ ( $CMD\_TMR.Q$ 为定时器输出状态) $CMD\_TMR(IN := %IX5, PT := 300); A:=CMD\_TMR.Q$
RETURN	返回（退出当前POU）	RETURN;
IF	选择	$D:=B*B;$ $IF D<0.0 THEN$ $C:=A;$ $ELSIF D=0.0 THEN$ $C:=B;$ $ELSE$ $C:=D;$ $END\_IF;$

指令	说明	示例
CASE	多重选择	CASE INT1 OF 1: BOOL1 := TRUE; 2: BOOL2 := TRUE; ELSE BOOL1 := FALSE; BOOL2 := FALSE; END_CASE;
FOR	FOR循环	J:=101; FOR I:=1 TO 100 BY 2 DO IF ARR[I] = 70 THEN J:=I; EXIT; END_IF; END_FOR;
WHILE	WHILE循环	J:=1; WHILE J<= 100 AND ARR[J] <> 70 DO J:=J+2; END WHILE;
REPEAT	REPEAT循环	J:=-1; REPEAT J:=J+2; UNTIL J= 101 OR ARR[J] = 70 END_REPEAT;
EXIT	退出循环	EXIT;
CONTINUE	继续循环下次执行	CONTINUE;
JMP	跳转	label: i:=i+1; JMP label;
;	空语句	;

## 赋值指令

赋值指令用于变量赋值，也就是赋值关键字的左边是变量，右侧为要赋的值，通过赋值关键字进行赋值，赋值关键字包含三种：“:=”、“S=”、“R=”。

- “:=” 为一般赋值，右值直接赋给左值，左值和右值相等。

例如：Var1 := Var2 \* 10;

完成执行后，Var1值为Var2的10倍。

- “S=” 为置位赋值，表示如果右值为TRUE，左值变量变为TRUE（置位），直到调用R=命令来初始化。
- “R=” 为复位赋值，表示如果右值为TRUE，左值变量变为FALSE（复位）。用于复位S=指令置位的变量。

例如：a S= b;

一旦b为 TRUE后，a会保持 TRUE，即使b 变为 FALSE后。

## 功能块的调用

**语法：** <FB 实例名>(FB输入变量:=<值和地址>|, <更多FB 输入变量:=<值和地址>|...更多 FB 输入变量);

调用语法： 调用一个延时功能块(TON)的实例，分配输入参数IN和PT，功能执行后，可以把结果Q赋值到变量A。

注意：TON功能块通过“TMR:TON”实例化。

实例化语法：<FB instance name> :<FB variable>;

TMR(IN := %IX5, PT:= T#300MS, Q=> q1, ET=>et1);

A:=TMR.Q;

## RETURN指令

RETURN指令表示当前置条件为TRUE时，离开此POU。

**语法：**

RETURN;

示例

IF b=TRUE THEN

RETURN;

END\_IF;

a:=a+1;

如果b是TRUE，语句“a:=a+1;”不会被执行，POU会立即被返回。

## IF指令

通过IF关键字，可以判断执行条件，根据执行条件，执行相应的指令。

**语法：**

IF <布尔表达式1> THEN

<IF\_指令

{ELSIF <布尔表达式2> THEN

<ELSIF\_指令1>

ELSIF <布尔表达式n> THEN

<ELSIF\_指令-1>

ELSE

<ELSE\_指令}>

END\_IF;

{}内部分是可选的

如果 <布尔表达式1> 为 TRUE, 那么只有 <IF\_指令> 被执行, 其它不被执行, 否则, 从 <布尔表达式2>开始, 一个一个计算布尔条件表达式直到其中一个表达式值为TRUE, 然后执行此表达式对应的指令, 如果没有表达式值为TRUE, 那么执行 <ELSE\_指令> 对应的指令。

### 示例

```
IF temp<17
THEN heating_on := TRUE;
ELSE heating_on := FALSE;
END_IF;
```

这里, 当温度低于17度时加热打开, 否则它保持关闭。

## CASE指令

使用CASE指令, 可以根据一个条件变量, 根据其对应的多个值罗列处理对应的命令。条件变量只能是整数。

### 语法:

```
CASE <Var1> OF
<value1>: <Instruction 1>
<value2>: <Instruction 2>
<value3, value4, value5>: <Instruction 3>
<value6 .. value10>: <Instruction4>
...
<value n>: <Instruction n>
ELSE <ELSE Instruction>
END_CASE;
```

CASE指令根据以下流程处理:

- 如果变量<Var1>的值为 <value1>, 那么<Instruction 1>会被执行。
- 如果 <Var1>没有匹配任何一个值, 那么<ELSE Instruction>被执行。
- 如果同一个指令在几个变量值时执行, 那么可以把这些值一个接一个的写出来, 用逗号隔开, 共同执行。
- 如果同一个指令会在一个变量范围内执行, 可以写上初始值和结束值, 中间用两个点隔开。

### 示例

```
CASE iVar1 OF
  2:c1:=c1+1;
  1, 6:c1:=c1-7;
  7..20:c1:=c1+5;
ELSE
  c1:=c1-1;
END_CASE
```

## FOR循环

通过FOR循环, 可以编写重复处理逻辑。

**语法:**

```
FOR <INT_Var> := <INIT_VALUE> TO <END_VALUE> {BY <Step size>} DO
<instructions>
END_FOR;
```

{} 内的部分是可选的。

<INT\_Var> 是计数器，是整数类型，只要计数器<INT\_Var>不大于<END\_VALUE>，<Instructions>会被执行。在执行<Instructions>之前首先要检查该条件，如果 <INIT\_VALUE> 大于 <END\_VALUE>，<instructions> 不会被执行。

当<Instructions>执行一次后，<INT\_Var>自动增加<Step size>。<Step size>可以是任意整数值，如果不写此参数，默认值为1。当<INT\_Var>大于<END\_VALUE>时，循环停止。

**示例**

```
FOR Counter:=1 TO 5 BY 1 DO
Var1:=Var1*2;
END_FOR;
Erg:=Var1;
```

假设Var1默认值是2，经过FOR循环后，它的值是32。

**WHILE循环**

WHILE循环和FOR循环一样可以作为循环处理使用，但和FOR循环不同是循环条件可以是任意布尔表达式。一旦循环条件满足，循环就执行，否则退出循环。

**语法:**

```
WHILE <boolean expression> DO
<instructions>
END WHILE;
```

当<Boolean\_expression>值为TRUE时，<Instructions>指令开始执行，直到<Boolean\_expression>值为FALSE。如果<Boolean\_expression>第一次值为FALSE，<Instructions>永不会被执行。如果<Boolean\_expression>永远为TRUE，<Instructions>重复执行不停止，进入死循环状态，编程时一定确保不要出现死循环。

**示例:**

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

在一定意义上来说，WHILE循环和REPEAT循环比FOR循环功能更强大，因为不需要在执行循环之前计算循环次数。因此，在有些情况下，用WHILE循环和REPEAT循环两种循环就可以了。然而，如果清楚知道循环次数，那么FOR循环更好。

## REPEAT循环

REPEAT循环不同于WHILE循环，因为循环条件是在循环指令执行后才检查的，这意味着，循环至少执行一次，不管循环条件值如何。

### 语法:

```
REPEAT
<instructions>
UNTIL <Boolean expression>
END_REPEAT;
```

### 执行逻辑:

<Instructions>一直执行直到<Boolean expression>值为TRUE。如果<Boolean expression>在第一次值TRUE，那么<Instructions>只被执行一遍。如果<Boolean expression>值永远是FALSE，那么<Instructions>永远执行不停，导致死循环。

### 示例:

```
REPEAT
Var1:=Var1*2;
Counter:=Counter-1;
UNTIL Counter=0;
END_REPEAT;
```

## CONTINUE语句

CONTINUE指令在FOR, WHILE和REPEAT循环中使用，用于提前结束本轮循环，并重新开始下一轮循环。

### 示例:

```
FOR Counter:=1 TO 5 BY DO
INT1:=INT1/2;
IF INT1=0 THEN
CONTINUE;
END_IF
Var:=Var1/UBT1L
END_FOR;
Erg:=Var1;
```

## EXIT语句

EXIT指令用于退出FOR, WHILE, 或REPEAT循环。

## JMP语句

JMP指令可用于无条件的跳转到指定标签处的代码行。

**语法:**

<label>:

JMP <label>;

<label>标签名位于程序行的开始处，JMP指令必须有一个跳转目标，也就是预定义的标签。到达JMP指令后，程会跳转到指定的标签处开始执行。

**示例:**

```
aaa :=0;  
_label1:aaa:=aaa+1;  
(*instructions*)  
IF (aaa < 10) THEN  
JMP _label1;  
END_IF;
```

变量aaa初始为0，只要其小于10，程序就会跳转到label1处重新执行，因此它会影响JMP指令和标签之间的程序的重复执行。

这样的功能也可以通过WHILE或REPEAT循环来实现。一般应慎用跳转指令，因为它降低了代码的可读性。

## 注释

在结构化文本中有两种写注释的方法。

- 单行注释：用“//”开始，用“//”结束。例如：“// This is a comment.”
- 多行注释：用“(\*”开始，用“\*)”结束。例如：“(\*This is a comment.\*)”

注释可以在ST编辑器声明或实现部分的任意地方。

注释的嵌套：注释可以放置在其他注释中。

**示例:**

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

## 6.2.4 ST编辑

### 6.2.4.1 ST工具箱

工具类别界面，如下：



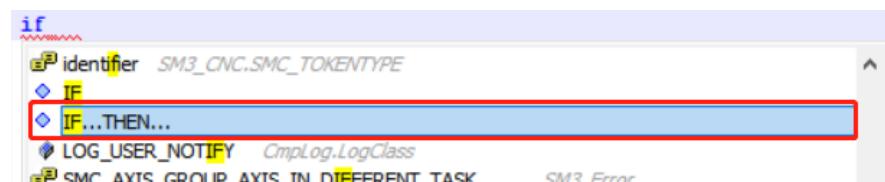
工具箱中包含ST语句、逻辑运算、定时器、计数器、数学函数、数据处理、数据转换、数据移位，可通过拖拽的形式拖入程序编写区例如ST语句、IF语句、FOR语句、WHILE语句、REPEATED、CASE语句、CONTINUE、JMP、EXIT、RETURN，插入时自动插入语句模板。

#### 6.2.4.2 智能输入

- 关键字匹配

输入ST语句类型关键字，能够自动匹配，语句包含IF语句、WHILE、FOR、CASE、REPEATED，格式化模板见附录语句模板。

如下图所示，输入IF能够弹出响应联想语句。



- TAB键快捷功能

- 能够对功能块、函数、方法、动作、程序自动格式化输入、输出
- 能够对功能块实例及实例的方法、动作自动格式化输入、输出
- 能够对IF、WHILE、FOR、CASE、REPEATED语句自动格式化，格式化模板见附录语句模板，入功能类型名、功能块实例等输入后按Tab键自动格式化

---

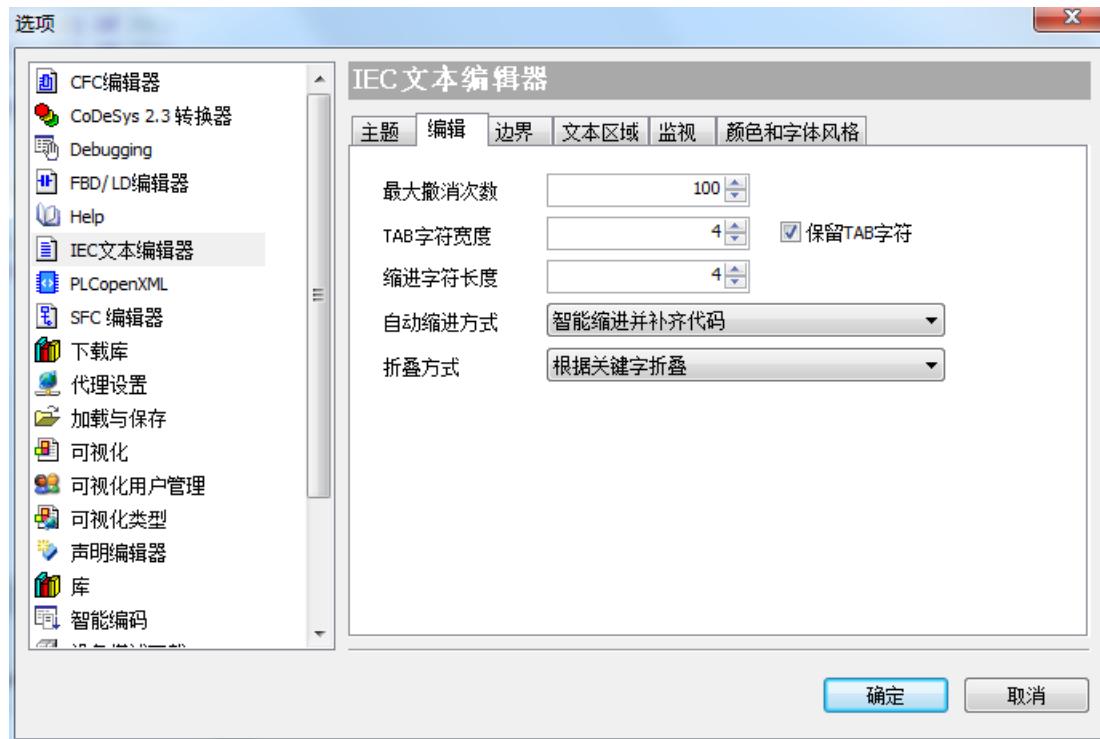
#### 说明

格式要求：Tab键快捷输入，行开始到光标位置为整体作为关键字，如果不匹配关键字，不会自动补全。

---

### 6.2.4.3 折叠和缩放功能

- 折叠方式支持关键字：关键字包含VAR、VAR\_INPUT、VAR\_GLOBAL、VAR\_OUTPUT、VAR\_IN\_OUT、VAR\_TEMP、VAR\_STAT、VAR\_EXTERNAL、CASE、FOR、REPEATED、IF/ELSE/ELSIF、WHILE、STRUCT、UNION、TYPE、\_\_TRY、\_\_CATCH、\_\_FINALLY。
- 如果选择自动缩进功能中的智能缩进，根据上述的关键字，自动加Tab长度，如果选择智能缩进并自动补齐，自动给关键字补齐结尾，如VAR，FOR，WHILE自动补齐结束标记，支持嵌套。
- 智能缩进时，如果上行是关键字，换行后自动增加Tab字符，如果非关键字，和上行同缩进。
- 块高亮显示。括号、中括号、WHILE、FOR、IF、ELSE、CASE、REPEAT、STRUCT、UNION、TYPE、TRY等类别之间显示块高亮信息，在文本边界和文本区域都有高亮显示标记。



### 6.2.4.4 IEC文本编辑器界面颜色

ST界面颜色每种都是一个模板，通过模板配置，也可以通过【工具】-【选项】-IEC文本编辑器-主题，颜色配置包含基本配置、IEC61131类型及标识符配置、在线配置及字体风格配置。

可根据用户喜好进行自定义设置，可设置参数具体如下表所示。

类别	说明
基本配置	主要设置基本界面基本颜色，主要包含： 背景色 前景色（默认文本颜色） 行高亮颜色 文本块高亮颜色 符号高亮颜色 光标颜色 焦点状态下选择文本背景颜色 非焦点状态下选择文本背景颜色 边界默认文本颜色 边界背景颜色 边界扩展背景 边界行高亮颜色 焦点状态分割线 非焦点状态分割线 折叠状态颜色 增量搜索颜色

类别	说明
IEC61131类型及标识符	主要设置数据类型和标识符颜色，主要包含： BOOL类型常数 时间类型常数 整数类型常数 浮点数类型常数 字符串类型常数 注释类型 特性类型 直接地址 全局变量 静态变量 输入变量 输出变量 输入输出变量 常量 临时变量 Persistent变量 Retain变量 外部变量 配置变量 FB 方法 动作 函数 结构体 枚举类型 枚举值 联合体 接口 操作符 关键字 错误
在线配置	监视框背景和文本 流控监视框背景和文本
字体风格	数据类型和标识符字体格式，如粗体，下划线等

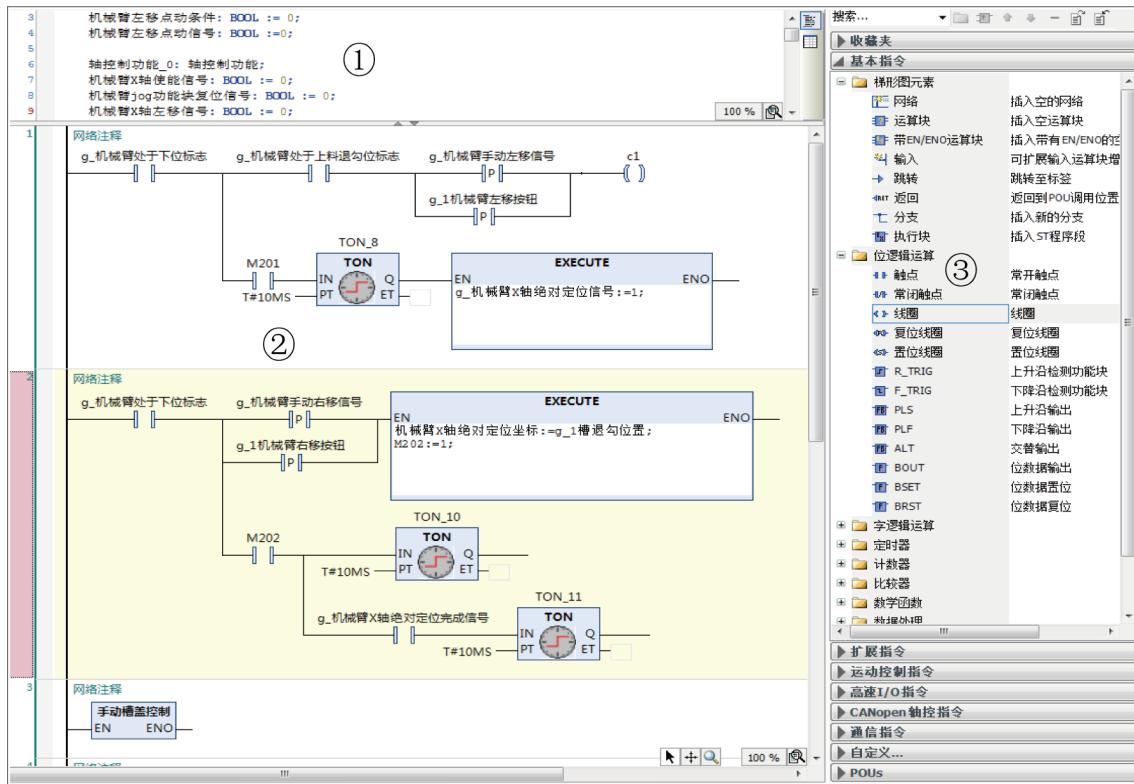
掉电保持、保持及常量类型和变量是修饰关系，两者只能选择一个，优先级关系如下：全局变量、输入、输出、输入/输出掉电保持（保持）本地、临时常量静态变量、配置变量、外部变量、枚举变量种变量都可以颜色设置，由于掉电和常量是特性，和变量类型是并列的，所以需要优先控制显示。

## 6.3 梯形图(LD)

### 6.3.1 概述

梯形图是图形化的编程语言，跟电路图的结构相近。梯形图包括一系列网络（也叫节，下文统一以“网络”代替），每个网络由左侧的竖线（电源轨，能流线）开始。一个网络由触点、线圈、运算块（函数、功能块、程序、执行块、动作、方法）、跳转、标签和连接线等构成。

网络左侧母线为能流线，其状态永远为TRUE，母线后会连接触点、运算块、线圈等元素。每个触点均分配布尔变量。如果变量值为TRUE，相当于开关闭合，条件会从沿着连接线从左向右传递，否则开关断开。在网络右侧的线圈，接收到从左侧传来的“开”或“关”信号，相应的TRUE或FALSE会被写到与线圈关联的布尔变量中。梯形图编辑界面如下图。

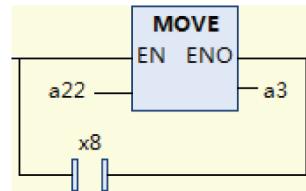


### 说明

- ①- 变量定义区；② - 梯形图编程区；③ - 工具箱。
- 梯形图语言中支持嵌入EXECUTE执行块，用于插入ST语言程序段。

梯形图主要元素包括触点、线圈、运算块、分支、注释等。通过插入、拖拽、划线、复制粘贴操作在网络中添加这些元素，形成梯形图执行逻辑。对于梯形图界面字体、操作数及注释显示可以通过【工具】-【选项】-【FBD/LD编辑器】设置。

梯形图支持监控、写入值、强制值、断点等在线调试功能。



### 6.3.2 梯形图元素

梯形图元素包括网络、触点、线圈、运算块、执行块、分支、跳转、标号、返回。

触点、线圈、运算块输入输出都和操作数关联，操作数可以是变量、常量(TRUE、FALSE、1,2等)、地址，具体见变量定义。

LD元素位于工具箱（菜单命令【视图】 - 【工具箱】）中，如下图所示。在工具箱中除了常规下元素、梯形图元素和IEC标准操作符（如布尔操作符、数学操作符）外，还包括功能块，当前程序中定义的POU。

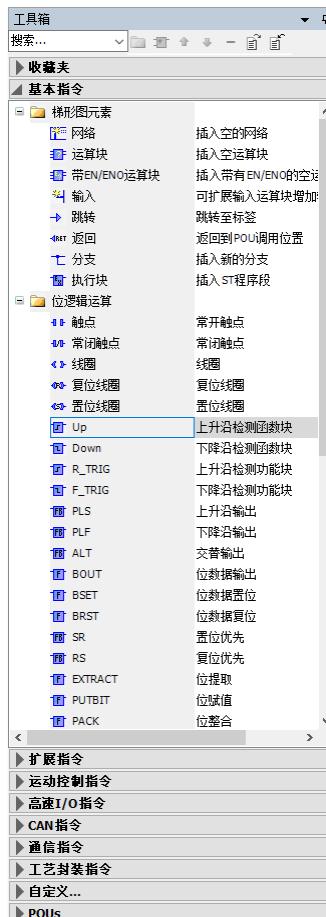


图6-1 LD工具箱

### 说明

当编程语言为梯形图时，在工具箱的位逻辑运算中将显示Up和Down指令块，其中Up指令块为上升沿检测函数块，Down指令为下降沿检测函数块，在梯形图中调用该指令可以实现不需进行实例化声明也可以实现边沿触发检测功能。

## 网络

### 图标 -

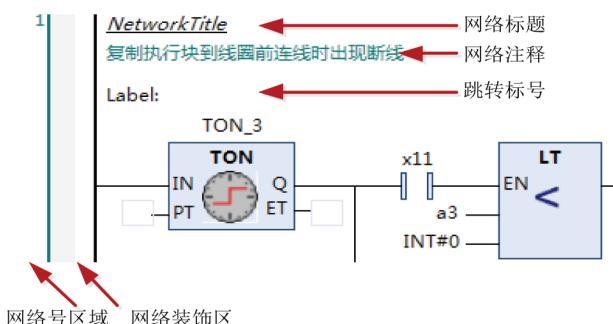
梯形图由一系列网络组成，其它所有的梯形图元素都位于网络内。每个网络都由左边的网络序号指示。

网络中可插入标题（网络总结性说明）和网络注释（网络比较详细的说明）。网络标题、网络注释是否显示通过选项配置(【工具】 - 【选项】 - 【FBD/LD编辑器】 - 【常规】)来控制。

网络中可插入标签，标签位于网络标题和网络注释下面，作为跳转目标。

网络也可以处于注释状态，通过菜单命令【切换网络注释状态】来使能或者禁用网络。

网络序号和网络内容之间有个区域叫网络装饰区，用来显示断点标志和书签位置。



## 触点

### 图标 -

触点分常开触点和常闭触点。触点传递ON (TRUE) 和OFF (FALSE) 值，触点为BOOL型变量，如果变量值为TRUE，常开触点向右传递ON (TRUE)，否则传递OFF (FALSE)，常闭触点传递值相反。

触点可以增加延信号功能，选中触点右键菜单命令【边沿检测】，可以把触点变为上升沿触发触点（触点变量值由FALSE变为TRUE时触点向右传递ON）或者下降沿触发触点（变量值由TRUE变为FALSE时触点向右传递ON）。

## 线圈

### 图标 -

线圈位于网络的末尾。左侧逻辑运算结果，赋值给线圈变量。线圈变量只能为BOOL类型，TRUE表示 (ON)，FALSE表示 (OFF)。线圈仅支持向上或者向下插入并联线圈。

线圈分为线圈、取反线圈、置位线圈、复位线圈。通过右键菜单命令或者快捷键可以进行4种线圈类型之间的切换。

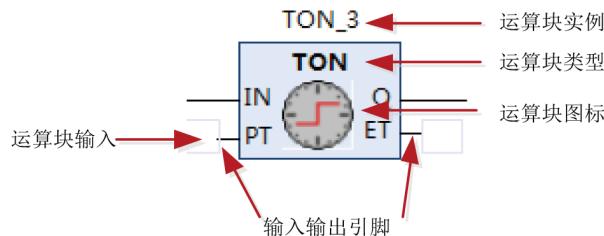
- 线圈：左侧逻辑运算结果直接赋值给线圈变量。
- 取反线圈：把左侧逻辑运算结果取反赋值给线圈变量。
- 置位线圈：如果线圈左侧的状态值为ON (TRUE)，则把线圈变量的值设置为ON (TRUE)，并且一直保持这个状态，直到下次该变量被复位线圈重置为OFF (False)。
- 复位线圈：对置位线圈进行复位。

## 运算块

### 图标 -

运算块可以是操作符、函数、功能块、程序、动作、方法。如果为功能块类型，则运算块框上面会增加编辑框来显示功能块实例。

一个运算块至少包含一个输入和一个输出。运算块主要有由下图组成：



运算块分普通运算块和En/Eno运算块。

- EN/ENO类型运算块：除了包含运算块本身所带的输入和输出外，还增加EN输入和ENO输出。EN/ENO运算块执行逻辑为：当EN为TRUE时执行运算块逻辑，执行完成后ENO为TRUE，如果EN为FALSE，不执行运算块，ENO为FALSE。注意：EN/ENO运算块输入连线只能连接在EN引脚，输出连线只能连接在ENO引脚。
- 运算块输入输出引脚：BOOL型输入引脚可以增加取反、上升沿、下降沿信号。运算块输出连引脚可以增加取反信号。
- 多输入连线运算块：有多个输入且多个输入均连接到能流线上，如下图为两个输入连线的多输入连线运算块。由于多输入连线运算块有多个连线和能流线相连，所以多输入连线运算块不能再并联分支中，并且只能在第一个分支中。

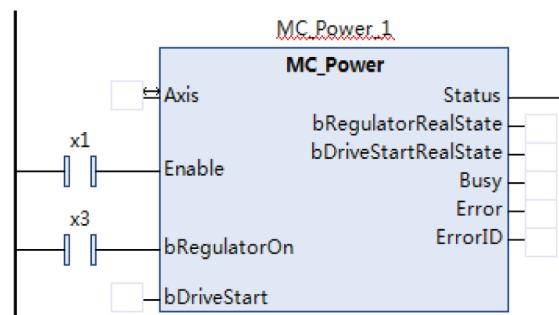
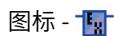


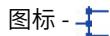
图6-2 多输入连线运算块

## 执行块



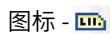
执行块是一个可以插入内嵌ST的块，在块中可以编辑ST语句。执行块可以放大缩小，最大为1000\*400。

## 分支



分支形成一个非闭合的并联逻辑。

## 标签



标签标明跳转位置，位于网络头部，通过跳转元素跳转到标签位置。跳转标签是字符串，需符合标识符命名规则。

## 跳转

图标 - 

当跳转元素左侧的输入为TRUE时，跳转到指定的标签位置执行。跳转元素位于网络的最右侧。

## 返回

图标 - 

当返回元素左侧的输入为TRUE时，当前程序立即退出执行。返回元素位于网络的最右侧。

### 6.3.3 LD 编辑器选项

LD编辑器选项用来控制LD界面的显示、单键命令设置、打印时显示方式。LD编辑器选项通过菜单命令【工具】-【选项】-【FBD/LD】打开。LD编辑器选项包含3个选项卡：常规、LD和打印。

#### 常规设置

常规选项卡设置如下图。



LD编辑器常规选项卡

#### 视图

- 插入网络标题：如果激活此选项，梯形图每个网络可以插入标题、编辑标题。如果已经插入了标题会在当前网络最上面增加一行，用来显示标题，如果没有标题，标题行不会显示。插入标题通过菜单命令实现。
- 显示网络注释：如果激活此选项，梯形图每个网络可以编辑网络注释。如果增加了网络注释，在网络标题下增加一行显示网络注释，如果网络注释不存在，不会产生空白行显示网络注释。编辑网络注释通菜单命令实现。
- 显示功能块图标 - 如果激活此选项，如果运算块定义了图标，运算块中间会显示图标。标准操作符（如 ADD、SUB）和功能块（如TON、TOF）都定义了图标，用户自定义的函数、功能块或者程序可以通过【右键对象】-【属性】-【位图】-【点击此处选择与工程有关的位图】来添加图片，形成运算块图标。

- 显示操作数注释：如果激活此选项，梯形图界面每个操作数上都可以编辑和显示操作数注释。操作数是一个编程概念，如变量、常量、地址都是操作数。由于梯形图中不一定都使用变量，如常量或者地址，这时可以通过操作数注释对它们进行注释。编辑操作数注释通过选择操作数字符串，然后右键菜单实现编辑。
- 显示变量注释：如果激活此选项，梯形图界面变量上，会显示变量声明时的注释。变量注释来自于变量声明，不能编辑。



### 字体

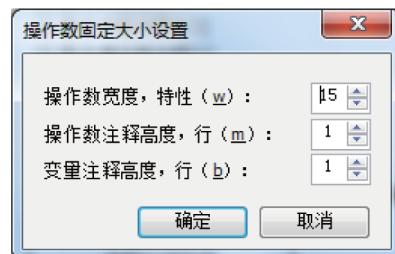
点击样本文字，弹出编辑器字体选择框，设置梯形图文字字体。默认字体为微软雅黑，使用小五号字体。字体范围主要包括操作数字符、注释。执行块字体使用文本编辑器字体（ST文本、变量声明文本）。

### 动作

- 新操作符的占位符：未实现
- 添加运算块时默认空引脚：如果激活此选项，新增加运算块时，运算块输入和输出引脚用空字符，如果未激活此选项，运算块输入和输出引脚用“???”字符。

### 操作数固定大小设置

如果激活此选项，可以设置操作数固定宽带、操作数注释高度和变量注释高度。如下图。



### 操作数固定大小设置

- 操作数宽度：设置操作数固定字符个数，默认15个。
- 操作数注释高度：设置操作数注释固定高度行数，默认1行。
- 变量注释高度：设置变量注释固定高度行数，默认1行。

### LD选项设置

LD选项卡设置如下图。



### 单键设置

单键功能，通过单个按键进行编辑操作，包括连线上执行的单键和元素上执行的单键。连线上执行单键命令插入串行元素，元素上执行单键命令插入并行元素。

另外选择元素时可以对元素功能进行切换，如取反切换、延信号切换、置位/复位切换。取反切换作用对象为触点、线圈，使用“/”按键；延信号切换作用对象为触点，使用空格按键；置位/复位切换作用对象为线圈使用空格按键。

单键设置，设置单键功能按键字符。每种功能都有默认单键字符，但是可以根据个人喜好自行设置。

### 说明

连线上单键功能对应字符或者元素上单键功能对应字符不能相同，但是连线上单键字符和元素上单键字符可以相同。

### 打印

打印选项卡设置如下图。



### 布局选项

打印大小适配方式：

- Poster，正常比例打印，打印时如果当前页面高度显示不了此网络使用，使用下一个页面打印，如果页面宽度显示了整个网络，使用下一个页面打印剩余的。
- Shrink to fit the widest network，表示打印时压缩显示内容使所有网络都一个页面宽度内显示，并且如果一个页面高度显示不完整个网络，显示部分网络，剩下的网络在下一个页面显示。
- 避免元素切割：如果激活了此选项，表示当一个元素在两个页面之间都有显示，则把当前元素在下一个页面显示，只能在Poster打印方式设置。
- 相邻页上的标记连接：表示设置了避免元素切割后，为了表示前后连接关系，增加连接标记。

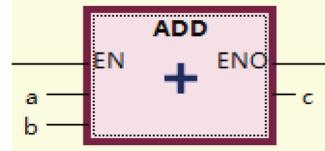
### 6.3.4 元素选择

选择是编辑的基础。选择对象可以是元件或者连线，选择可以单选也可以按下Ctrl、Shift多选，可以连续选择，也可以非连续选择。

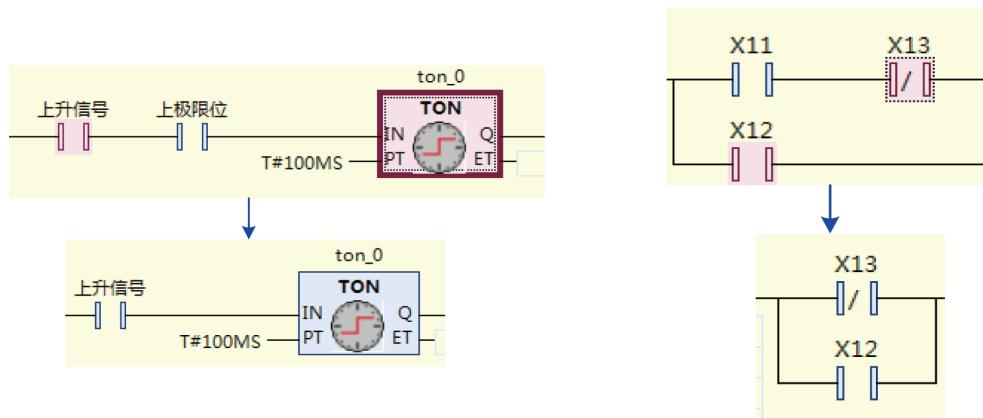
元素选择后元件会处于高亮状态，如触点被选中时显示形式为 $\boxed{\text{TT}}$ ，其中外框的虚线表示触点处于焦点状态。选择的元素可以和其它元素粘贴并联，拖拽串并联。由于多选元素可能不连续，这就需要说明选择元素形成的结果逻辑。选择结果逻辑的原则是保证原来逻辑的一致性。支持鼠标框选、Ctrl和Shift多选、全选。

#### 元素选择形成的结果逻辑

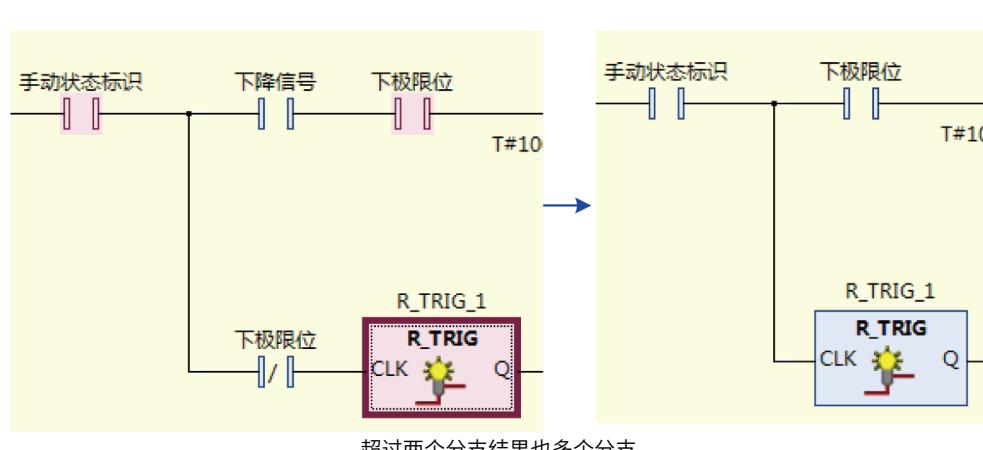
- 单选时：选择触点、线圈，只包含选择的触点、线圈；选择运算块包括运算块本身、无连线输入操作数和输出操作数。如下图，选择ADD的运算块，粘贴时会包含ADD运算块本身、输入a、输入b、输出c。

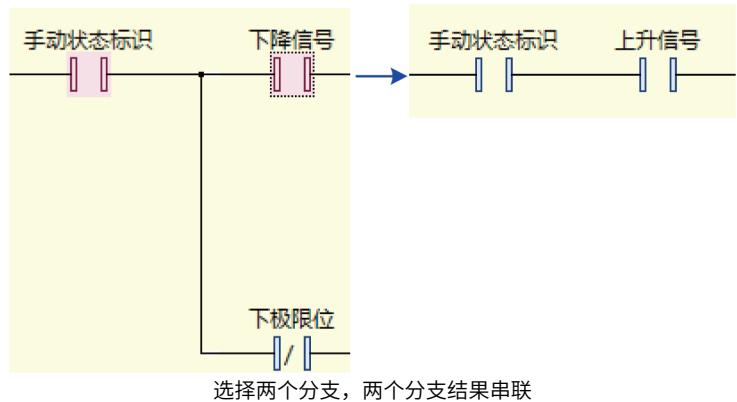


- 多选时：选择一条线上的串联（包含非连续选择）；选择并行线上元素（包含非连续选择），形成并行结果。

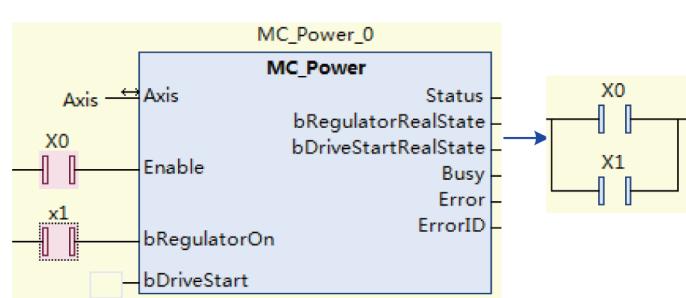
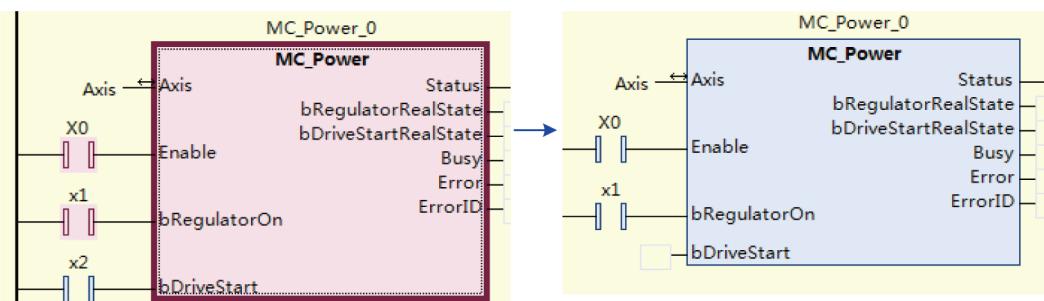


- 多选时：如果选择元素跨多分支，形成结果也跨多分支，不改变原来逻辑，但是如果只选择两个分支的，结果会把两个分支元素串联。



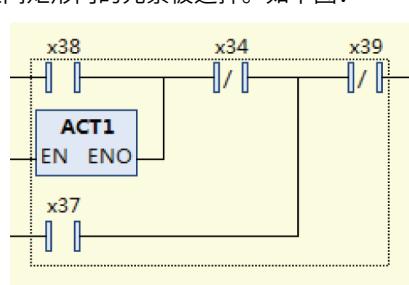


- 多选时：如果同时选择多输入连线运算块本身和多个输入连线上的元素，结果和选择一致；如果只选择多输入连线上元素，没有选择运算块，则把多个输入连线元素形成打开并联。



## 框选

从鼠标按下位置到鼠标松开位置之间矩形内的元素被选择。如下图：



## 说明

框选只能单网络，不能跨网络；鼠标按下的空白处网络开始。

## Ctrl和Shift多选

Ctrl和Shift多选符合标准多选方式：

- Ctrl多选：按下Ctrl时，如果当前元素没有选择则把当前元素增加到选择列表中；如果已经选择，则从选择列表中去除。
- Shift多选：从上次选择元素到本次选择元素矩形内的元素进行选择。

## 全选

使用Ctrl+A快捷键进行全选功能。全选会把所有网络选择。

### 6.3.5 标准编辑命令

梯形图标准编辑主要是我们常规编辑操作，如复制、粘贴、删除、剪切、撤销、恢复。使用标准的编辑快捷键。

## 复制

对选择的元素进行复制。复制的结果也就是选择的元素。详见元素选择章节。

梯形图中可以复制的元素包括网络、触点、线圈、运算块、字符串、分支连线。

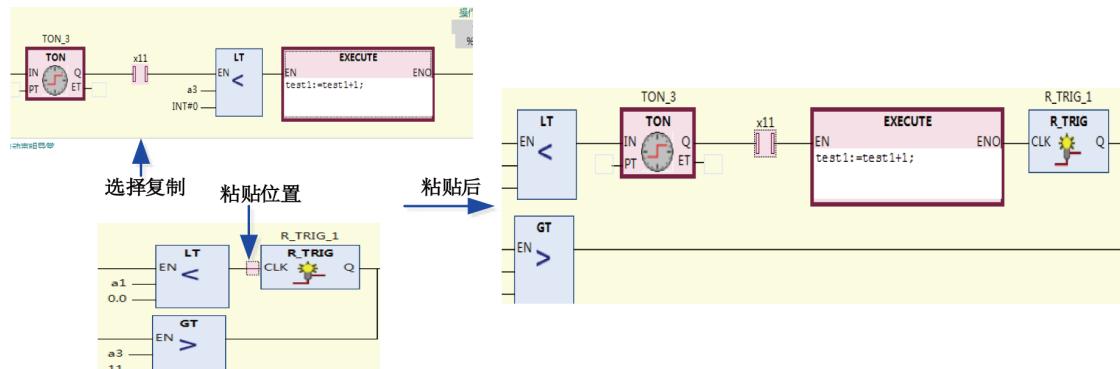
复制可以复制连续选择的元素，也可以复制非连续的，和选择有关。但是复制的元素只能单个网络内，如果跨网络选择，只能复制焦点元素网络选择的数据。

## 粘贴

粘贴是对复制的元素进行粘贴。粘贴规则如下：

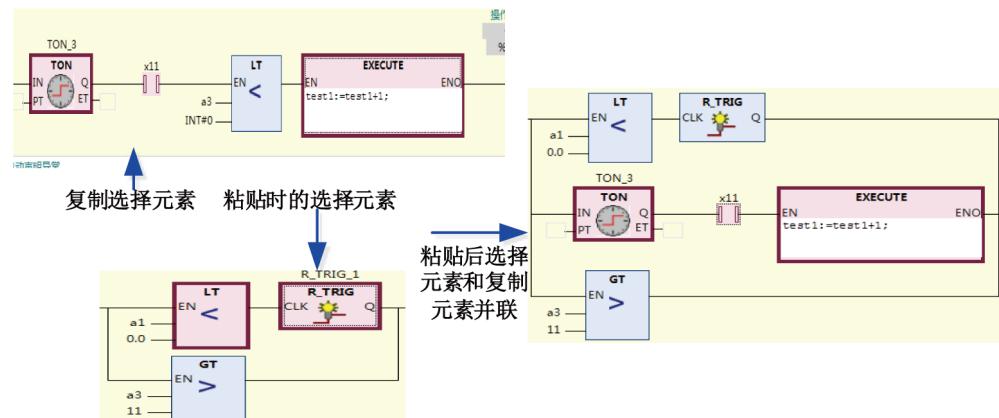
- 连线上粘贴。

在选择连线上粘贴时，连线位置插入复制元素，形成串联关系。



- 元素上粘贴。

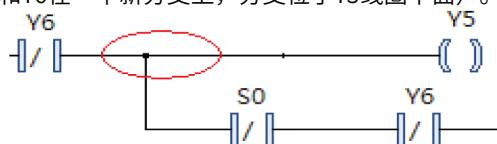
在选择元件上粘贴时，和所选择的批量元件生成并联关系。在形成并联关系时，被选择元件需要满足可并联条件，才能粘贴，即选择的元件必须满足：在一条线上、必须连续、不能跨越分支、开始元素和结束元素不能再并联分支内部和外部。



- 线圈位置粘贴。

由于线圈右侧不能存在元素，所以粘贴时有些特殊规则。跳转元素、返回元素规则和线圈相同，下面仅说明线圈。

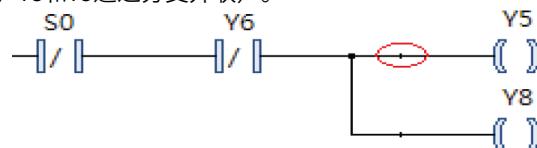
- 如果选择为线圈（选择元素并联），粘贴的元素位于一个新分支上，此分支位于线圈下面，如下图所示（选中Y5粘贴S0和Y6，S0和Y6在一个新分支上，分支位于Y5线圈下面）。



- 如果选择为线圈之前的连线（选择连线串联），粘贴时，根据复制的元素内容，分为复制内容仅包含一个分支和复制内容包含多个分支。

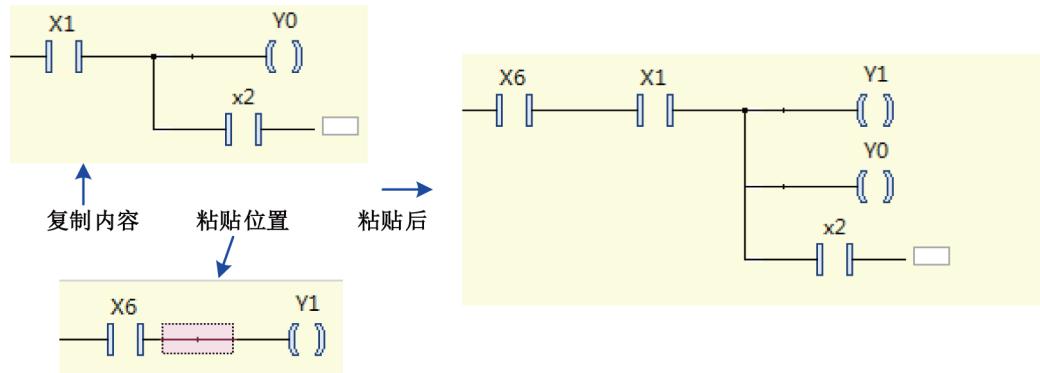
#### 复制内容仅包含一个分支

- 如果复制内容不包含线圈，复制元素直接和线圈串联。
- 如果复制元素包含线圈，则复制元素线圈之前的数据插入到连线选择位置，复制元素中的线圈和选择连线之后的线圈形成一个非闭合并联，如下图（选择Y5之前连线粘贴S0、Y6、Y8，粘贴后，Y8之前S0、Y6串联在Y5之前连线上，Y8和Y5通过分支并联）。



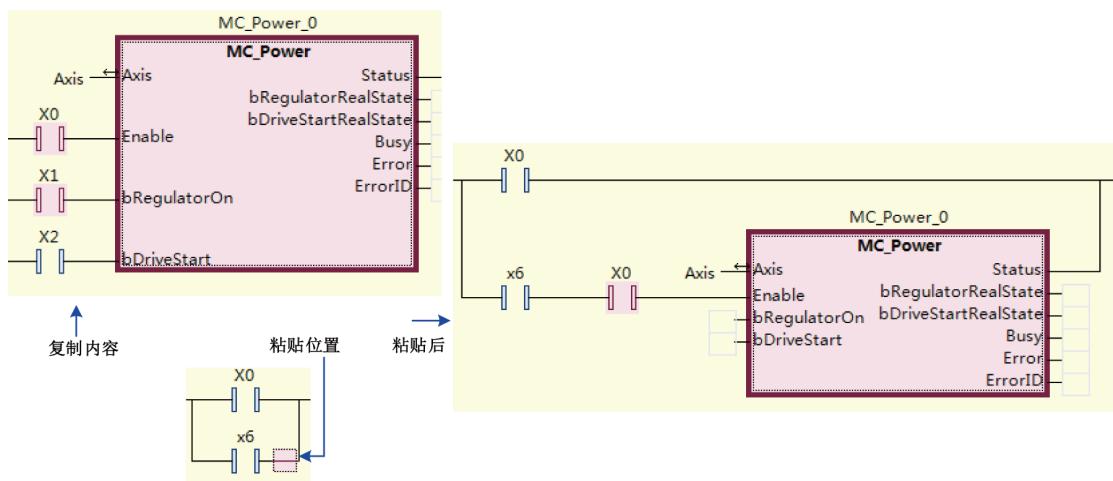
#### 复制内容包含多分支

- 如果复制内容水平最后一个分支不包含线圈，在选择连线位置插入复制内容。
- 如果复制内容水平最后一个分支包含线圈，则把复制内容水平最后一个分支的线圈和选择连线之后的线圈并联，其它数据串并联不变，如下图：复制内容为X1、Y0、X2粘贴在Y1线圈之前的连线位置，粘贴后，Y0位于Y1线圈下，X1位于Y1之前的连线上。



- 多输入连线运算块粘贴。

多输入连线运算块只能在第一个分支非并联位置，所以在粘贴位置，如果不能包含多连线运算块，会删除复制运算块从第二个输入连线开始的连线元素；如下图，复制的运算块输入X0、X1触点，粘贴时X1被删除。

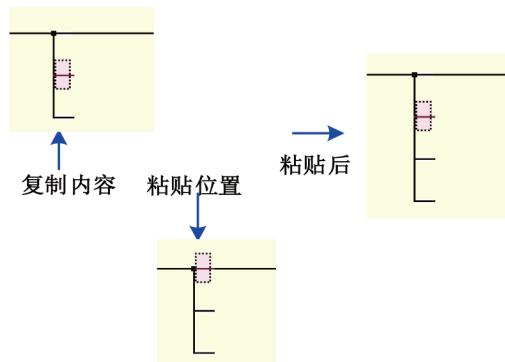


- 网络粘贴。

可以单选或者多选网络，进行复制粘贴，只有选择了网络才能粘贴复制的网络。

- 单分支连线粘贴。

单分支连线粘贴，用于增加单分支，和向上/下插入分支功能相同。复制单分支连线，可以在连线位置粘贴，粘贴在选择连线的下面，如下图所示：

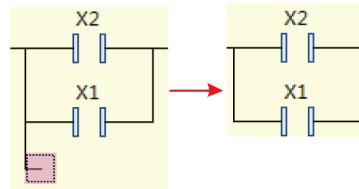


## 删除

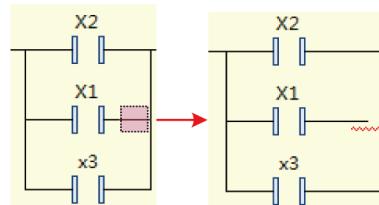
删除是对选择的元素进行删除。元素删除后，会选择下一个元素，以保证操作的连贯性。

删除可分为元素删除和连线删除：

- 元素删除 - 删除本身。
- 连线删除 - 删除分支连线、垂直连线及垂直连线相连的左侧连线。  
删除分支连线- 空分支连线删除后，此分支也同时会被删除。



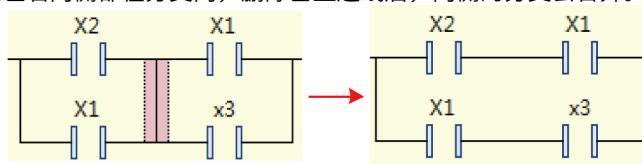
删除垂直连线相连的左侧连线 - 删除垂直连线左侧相连的连线，会把连线断开，分支打开。



删除垂直连线 - 原则上，删除垂直连线后，此垂直连线不再存在。垂直连线删除后，会产生三个结果：合并分支、打开分支、分支左移。

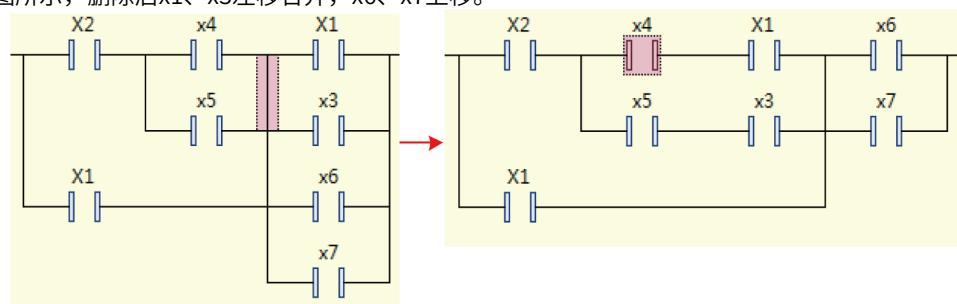
### 1. 合并分支

如果选择的垂直连线左右两侧都在分支内，删除垂直连线后，两侧的分支会合并。



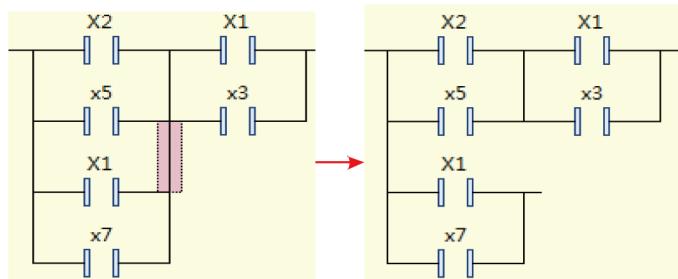
注意：如果删除垂直连线后产生桥式电路，这时会把右侧上下分支左移合并，右侧其它分支下移。

如下图所示，删除后x1、x3左移合并，x6、x7上移。



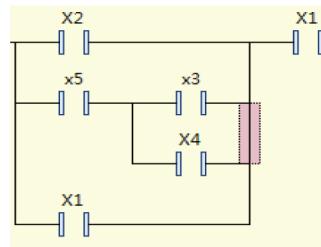
### 2. 打开分支

如果选择的垂直连线左侧在分支内，右上存在分支，右下不存在，删除垂直连线后会把左侧选择之下的分支断开。



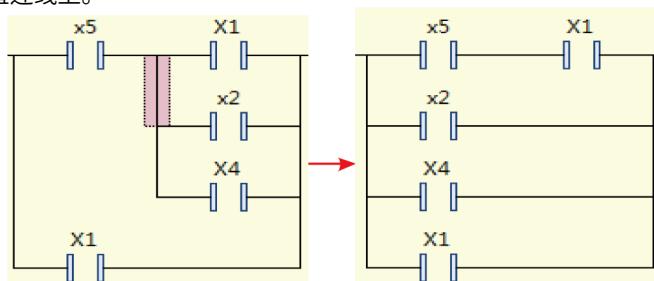
## 说明

如果删除垂直连线后产生桥式电路，则不能将其删除。



### 3. 分支左移

如果垂直连线右侧在分支内，左上存在分支，左下不存在，删除垂直连线后会把右侧选择之下的分支左移到上一个垂直连线上。



## 剪切

剪切功能是把选择的元素复制，以备粘贴，然后删除。

## 撤销/恢复

撤销：撤退到上一步编辑状态，并且恢复上一步选择元素。

恢复：恢复到下一步编辑状态，并且恢复下一步的选择元素。

## 6.3.6 LD 菜单命令

菜单命令可以通过右键菜单或者LD工具栏菜单执行的梯形图命令。

### 插入网络

包含插入网络命令和插入网络(在下方)两个菜单命令，可以通过拖动工具箱中“网络”来插入网络。

命令执行条件：先选择网络，在选择的网络上方/下方插入新网络。

插入网络：图标 - 快捷键：Ctrl + I，表示在选择网络上面插入一个空网络。

插入网络（在下方）：图标 - 快捷键：Ctrl + T，表示在选择网络下方插入一个空网络。

### 切换网络注释状态

图标 - 快捷键：Ctrl + O，把一个网络在注释状态和非注释状态进行切换。

在注释状态，整个网络的代码无效，代码不会执行，执行块也不能编辑。

命令执行条件：选择网络。

## 插入网络头信息

网络头主要包含网络标题、网络注释和标号。

插入网络头相关命令包含插入标号、编辑网络标题和编辑网络注释。

- 编辑网络标题：图标 - ，编辑选择网络的标题。
- 命令执行条件：选择网络，并且选项中“显示网络标题”使能
- 编辑网络注释：图标 - ，编辑选择网络的注释。
- 命令执行条件：选择网络，并且选项中“显示网络注释”使能。
- 插入标号：图标 - ，给选择网络插入跳转标签，作为跳转元素跳转位置。
- 命令执行条件：选择网络。

## 插入运算块

包含插入运算块、插入空运算块、插入带EN/ENO的运算块、插入带有EN/ENO的空功能块和插入并行运算块（在下方）5个菜单命令，用来插入操作符、功能、功能块和程序，也可以通过拖动工具箱中“运算块”或者“带有EN/ENO的运算块”来插入运算块。

前四个命令用来插入串联运算块，最后一个命令用于插入和选择元素并联的运算块。

插入位置：

1. 选择水平连线，在水平连线处插入一个运算块。
2. 选择元素，在选择元素左侧插入运算块。

- 插入运算块：图标 -  快捷键：Ctrl + B，弹出输入助手选择一个要插入的运算块。
- 插入空运算块：图标 -  快捷键：Ctrl + Shift + B，插入一个空的运算块，不弹出输入助手。可以在运算块类型处输入运算块类型。
- 插入带EN/ENO运算块：图标 -  快捷键：Ctrl + Shift + E，弹出输入助手选择一个要插入的运算块，此运算块带有EN/ENO输入和输出。带EN/ENO的运算块，当EN为TRUE时运算块才执行，为FALSE时不执行，ENO和EN结果相同。
- 插入带有EN/ENO的功能块：插入一个空的运算块，不会弹出输入助手，此运算块带有EN/ENO输入和输出。
- 插入并行运算块（在下方）：在选择的元素下方插入一个空的运算块。选择元素可以是触点、运算块。

## 插入执行块

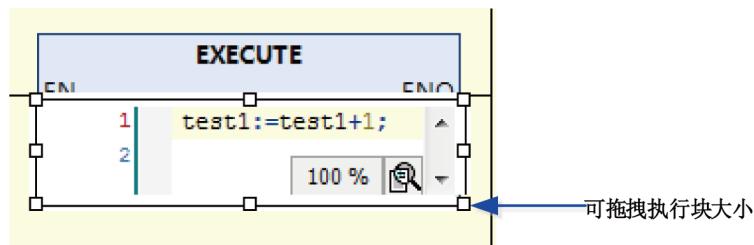
插入执行块：图标 - ，在当前选择位置插入串联执行块，也可以通过拖动工具箱中“执行块”来插入。

插入位置：

1. 选择水平连线，在水平连线处插入一个执行块；
2. 选择元素，在选择元素左侧插入运算块。

执行块是一个可以编辑ST语句的块，单键文本区域进行编辑；执行块只有EnEno输入和输出。

执行块大小拖拽：在编辑状态可以拖拽执行块边框，实现执行块大小控制，如下图：



## 插入输入

插入输入：图标 - 快捷键：Ctrl + Q，给可变输入运算块增加输入。

可变输入运算块：ADD、+、MUL、\*、SEL、AND、&、OR、|、XOR、MAX、MIN、MUX。

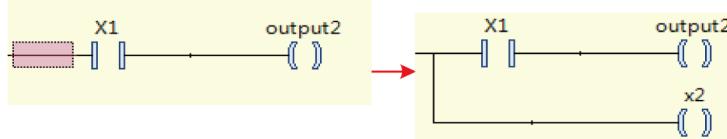
插入位置：当选择输入引脚时，在当前输入引脚之前加入一个输入；选择运算块，增加的输入引脚位于最后位置。

## 插入线圈

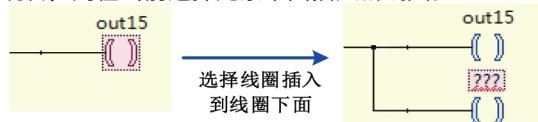
包含插入线圈、插入置位线圈和插入复位线圈三个菜单命令，也可以通过拖动工具箱中“线圈”、“置位线圈”和“复位线圈”元素来插入线圈。

- 命令执行条件：选择位置不能位于并联分支中，也不能位于多输入连线运算块输入位置。
- 插入线圈：图标 快捷键：Ctrl + Shift + A，在当前位置输出一个线圈。
- 插入位置：

1. 选择水平连线或者元素，在水平连线处或者元素左侧插入一个线圈，此线圈和连线通过非闭合分支处理。



2. 如果选择线圈、返回或者跳转，则在当前选择元素下面插入新线圈。



插入的线圈缺省变量为“???”，需要输入所需的变量或常量，可以使用输入助手（功能键<F2>），直接从变量列表中选择输入。

- 插入置位线圈：图标 ，表示在当前位置插入一个置位线圈。操作方式和上述“插入线圈”相同。
- 插入复位线圈：图标 ，表示在当前位置插入一个复位线圈。操作方式和上述“插入线圈”相同。

## 插入触点

包含插入触点、插入常闭触点、插入并联下触点和插入并联上触点四个菜单命令，也可以通过拖动工具箱中“触点”、“常闭触点”元素来插入触点。

- 插入触点：图标 快捷键：Ctrl + K，表示在当前位置前串联插入一个常开触点。
- 插入位置：

1. 选择水平连线，在水平连线处插入一个触点；
2. 选择一个网络，那么新触点插入到最后；

### 3. 选择元素，新触点插入到元素左侧。

触点缺省变量名为“???”。点击文本输入所需的变量或常量，可以使用输入助手（功能键<F2>），直接从变量列表中选择输入。

- 插入常闭触点：图标 - ，表示在当前位置串联插入一个常闭触点。操作方式和上述“插入触点”相同。
- 插入并联下触点：图标 - ，快捷键：Ctrl + R，表示在选择元素下并联插入一个常开触点。选择元素可以是触点或者运算块。
- 插入并联上触点：图标 - ，快捷键：Ctrl + P，表示在选择元素上面并联插入一个常开触点。操作方式和上述“插入并联下触点”相同。

## 插入分支

包含插入分支、在上面插入分支两个菜单命令，也可以通过拖动工具箱中“分支”元素来插入分支。插入的分支是一个非闭合的线。

- 插入分支：图标 - ，快捷键：Ctrl + Shift + V，表示在所选连线位置插入一个分支。  
插入位置：

1. 选择连线，在连线下方插入一条分支。
2. 选择触点或者线圈，在选择元素之前插入。

如下图，每个选择位置表示一个分支。

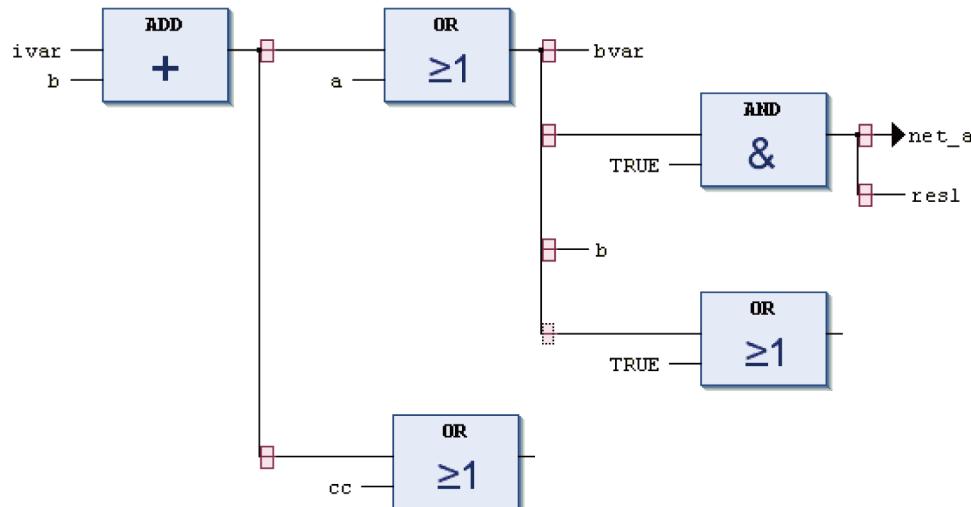


图6-3 分支标记

- 在上面插入分支：图标 - ，表示在所选分支上面增加一个分支。选择分支连线时，才能执行。

## 跳转和返回

包含插入跳转命令和插入返回两个菜单命令。跳转和返回属于程序执行顺序控制命令，正常程序按照网络顺序从上到下，从左到右依次执行。可以通过拖动工具箱中“跳转”来插入跳转元素或者工具箱中的“返回”来插入返回元素。

跳转、返回和线圈一样，必须在最右侧，所以插入跳转和插入返回的规则和插入线圈一样，详见插入线圈命令。

- 插入跳转：图标 - 快捷键：Ctrl + L，表示插入一个跳转元素，跳转到指定标号位置。跳转位置为网络中的标号，也就是说可以从一个网络跳转到另一个网络。当跳转前的输入条件满足时才能执行跳转。
- 插入返回：图标 -，表示插入一个返回元素。当输入条件满足时，当前POU执行返回，返回到调用它的POU。

## 取反

图标 - 快捷键：Ctrl + N，对运算块输入、运算块输出、跳转条件、返回条件、触点值或者线圈取反。

取反命令可以在两种位置执行：

- 元素取反：主要是触点和线圈。取反后触点和线圈内增加一个斜线 (/)。
- 连线取反：主要包含运算块输入连线、运算块输出连线、线圈输入连线、跳转输入连线、返回输入连线，取反后连线处增加一个圆圈。

可取反位置如下图：

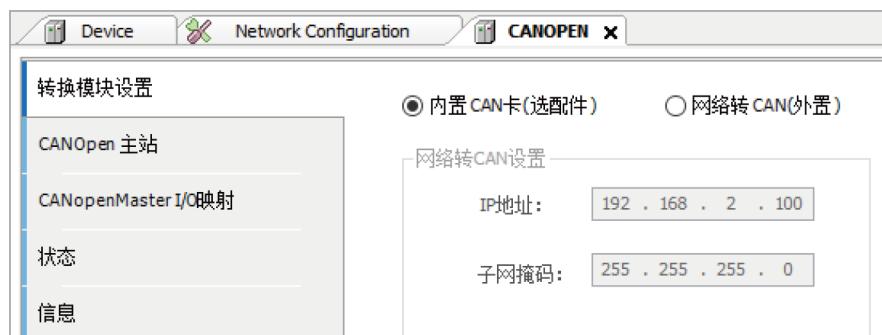


图6-4 可“取反”位置

再次执行取反命令，返回到初始状态。

## 边沿检测

图标 - 快捷键：Ctrl + E，表示对触点、运算块输入连线、线圈输入连线、跳转元素输入连线、返回元素输入连线增加边沿触发功能。

上升沿检测相当于R\_TRIG功能块，下降沿相当于F\_TRIG功能块。

边沿检测命令可以在两种位置执行：

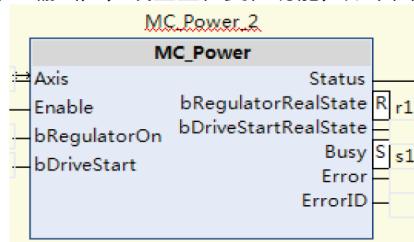
- 触点边沿检测：选中触点执行边沿检测命令，触点增加边沿检测功能， 表示上升沿； 表示下降沿。
- 连线增加边沿检测：运算块输入连线、线圈输入连线、跳转元素输入连线、返回元素输入连线，执行块边沿检测命令，连线上增加延信号符号，上升沿检测符号为；下降沿检测符号为。只有BOOL型输入连线才能添加边沿检测功能。

## 置位/复位

图标 - 快捷键：Ctrl + M，该命令用于增加置位或者复位输出功能。置位输出显示为“S”，复位输出显示为“R”。可以多次执行此命令，在置位、复位和正常输出之间切换。

置位/复位命令可以在两种位置执行：

1. 选择线圈，执行此命令变为置位、复位线圈。置位线圈：。复位线圈：
2. 选择运算块BOOL型输出连线（非主输出），设置置位复位功能，如下图：



## 设置输出连接

图标 -  快捷键：Ctrl + W，当运算块有多个输出时，可以修改主输出的引脚（一个运算块只有一个主输出，主输出和后继的元素相连）。如下图：

选择要修改的输出引脚，执行此命令，修改输出连接。

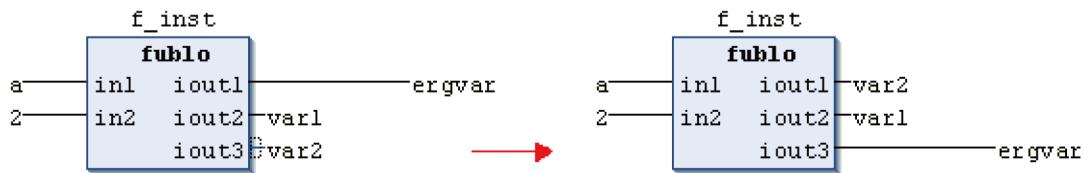


图6-5 更改输出连接

## 更改输入、输出引脚显示

包含更新参数和删除未使用的FB调用参数两个菜单命令。

更新参数：图标 -  快捷键：Ctrl + U，表示更新所选的运算块的输入和输出参数。如果运算块的输入或者输出参数发生变化时，通过执行“更新参数”命令，更新运算块的输入和输出参数。

删除未使用的FB调用参数：图标 - ，删除未使用的运算块输入引脚和输出引脚，也就是一个运算块中输入或者输出为“???”或者空时，这些输入和输出将不再显示。

## 批量更新运算块

当前打开的是梯形图编辑器时，如果当前编辑器包含的任何运算块的输入或输出参数发生变化，单击工具栏上的“批量更新运算块”，将批量更新当前编辑器中的运算块的输入和输出参数。此命令执行时会对当前编辑器中所有运算块的参数先检查再更新。



### 示例：

修改FBConVerTVa的输入输出参数：删除输入参数Time1, Time2, Time3, Alway1和输出参数Signal1, Signal2。

```

1  FUNCTION_BLOCK FBConvertVa
2  VAR_INPUT
3    Open: BOOL;
4    Close: BOOL;
5    In1: BOOL;
6    Alway: BOOL; //真空气氛
7    Time0: DINT; //关真空气氛OFF /ms
8    Time1: DINT; //信号延时ON /ms
9    Time2: DINT; //信号延时OFF /ms
10   Time3: DINT; //异常延时ON /ms
11   Always1: BOOL; //真空报警确认
12 END_VAR
13 VAR_OUTPUT
14   Opened: BOOL;
15   Closed: BOOL;
16   Signal11: BOOL; //打开信号
17   Signal12: BOOL; //关闭信号
18 END_VAR
19 VAR_IN_OUT
20   Out1: BOOL; //开真空

```

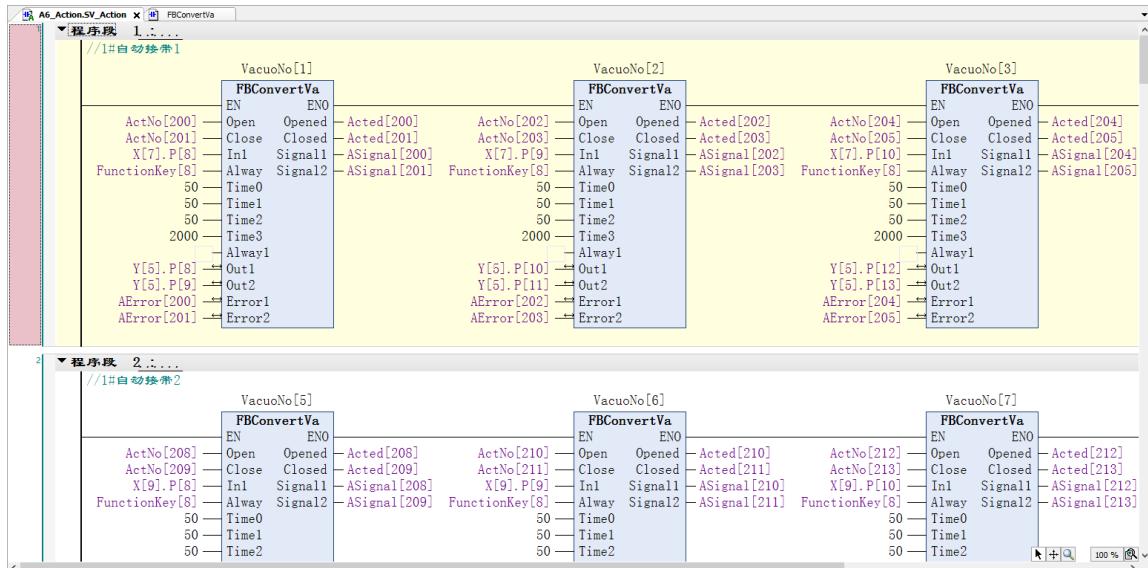
100% 10

```

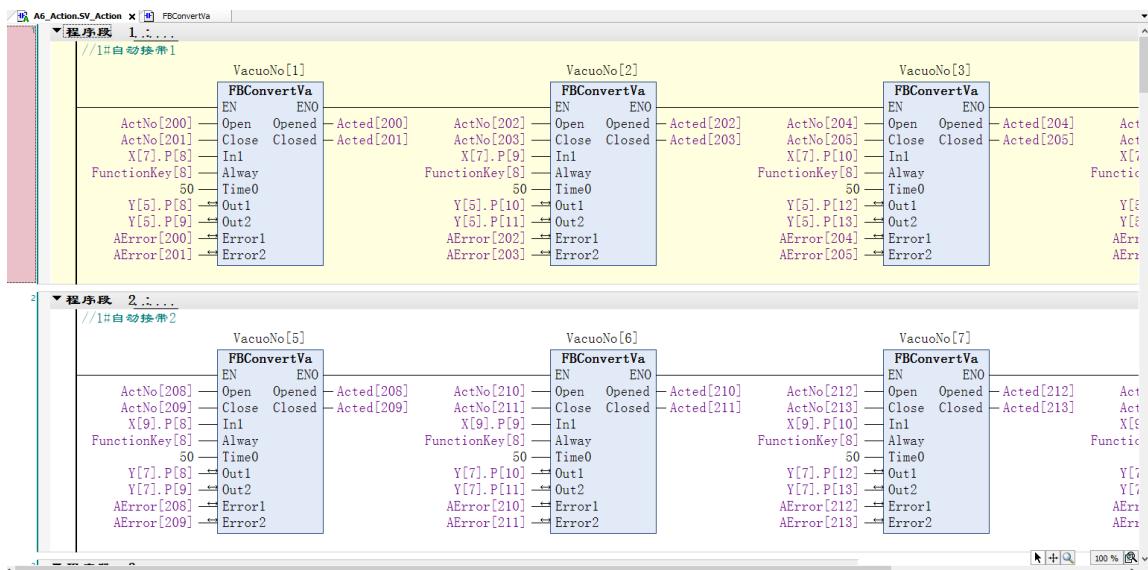
FUNCTION_BLOCK FBConvertVa
VAR_INPUT
  Open: BOOL;
  Close: BOOL;
  In1: BOOL;
  Alway: BOOL; //真空气氛
  Time0: DINT; //信号延时ON /ms
  Time1: DINT; //信号延时OFF /ms
  Time2: DINT; //异常延时ON /ms
  Time3: DINT; //异常延时OFF /ms
  Always1: BOOL; //真空报警确认
END_VAR
VAR_OUTPUT
  Opened: BOOL;
  Closed: BOOL;
  Signal11: BOOL; //打开信号
  Signal12: BOOL; //关闭信号
END_VAR
VAR_IN_OUT
  Out1: BOOL; //开真空

```

更新前：



更新后：



## 说明

执行批量更新运算块命令时，会对当前激活梯形图类型的编辑器中可更新的运算块执行批量更新操作，根据各运算块最新定义的输入、输出、输入输出参数更新其输入、输出、输入输出引脚，可更新的运算块包含程序PRG、功能块FB、函数FUN、行为Action及方法Meth。

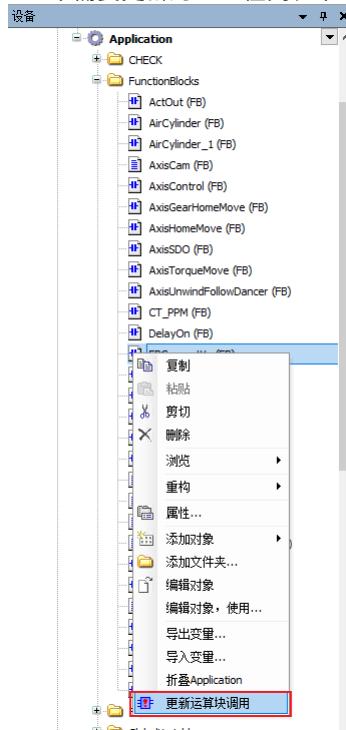
## 更新运算块调用

在设备树中选择POU节点（只包括函数、功能块、功能块方法、程序）后，在弹出的右键菜单中选择“更新运算块调用”，将更新工程中的所有梯形图中（ST暂不支持）调用该POU的运算块的输入输出引脚。

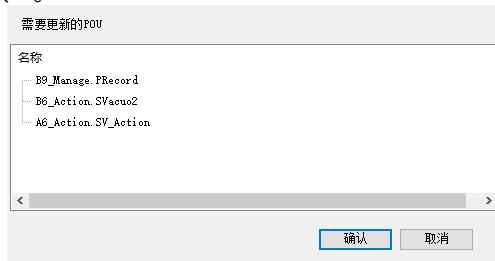
选择“更新运算块调用”后，会查找所有调用该POU的其他POU并对引脚重新计算。如果要做更新会弹出修改列表确认窗口，列出所有需要修改的POU，单击“执行更新，单击”取消“不做更新；如果没有需要更新的POU，不会弹出该窗口。

### 示例：

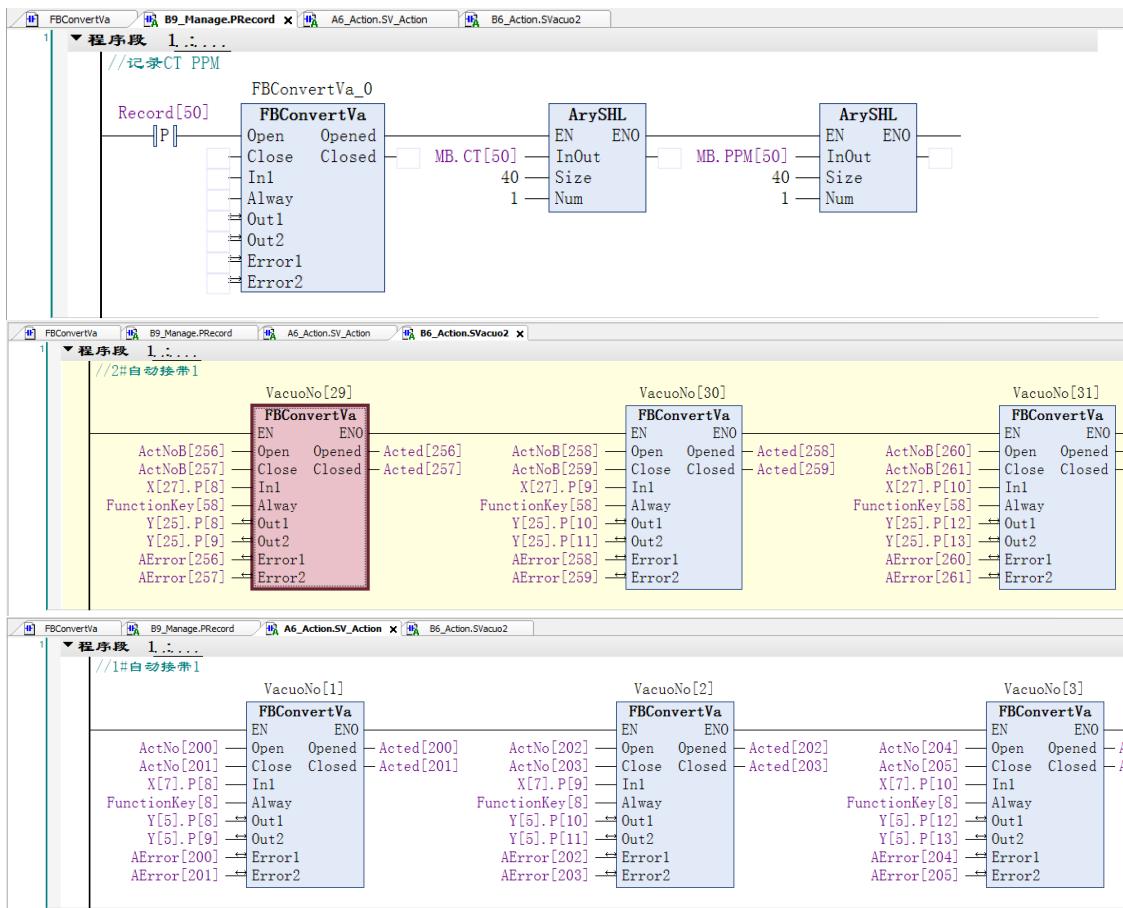
1. 在设备列表中右键单击“FBConvertVa”，选择“更新运算块调用”会将B9\_Manage.PRecord, B6\_Action.SVacuo2, A6\_Action.SV\_Action中需要更新的POU在列表中显示。



2. 在打开的对话框中单击“确认”。



所有调用FBConvertVa的运算块（B9\_Manage.PRecord、B6\_Action.SVacuo2、A6\_Action.SV\_Action）输入输出引脚将更新。



## 说明

对列表中的POU更新引脚时，只会更新所选择POU的运算块调用的引脚，不会更新其他引脚；但如果选择的是FB功能块并且FB功能块存在多个Action，会将FB调用和FB所有的Action调用一起做更新（因为Action的输入输出默认与FB功能块相同）。

## 转换为LD语言

显示为梯形图逻辑：快捷键：Ctrl + 2。把FBD/IL转换为LD语言；由于FBD、IL暂时不再支持，旧工程可以通过此命令把FBD/IL转换为LD语言显示。

## 跳转网络

转到…：跳转到指定的网络。弹出跳转网络输入框，指定跳转的网络编号。



## 编辑操作数注释

编辑操作数注释：编辑选择的操作数的注释。

命令执行条件：

- 在FBD/LD选项中，激活选项“显示操作数注释”。
- 需要选择操作数字符串。

操作数是逻辑概念，输入变量、常量、地址都是操作数，如运算块输入变量、触点关联变量、线圈关联变量、运算块实例等。

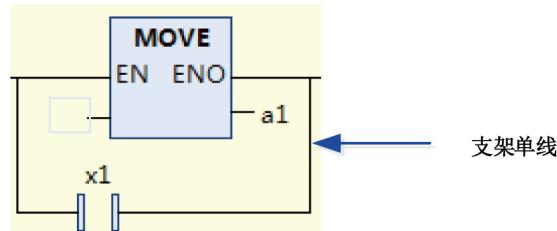
选择操作数字符串，执行此命令显示编辑操作数注释对话框，对操作数注释进行编辑，如下图。



## 并联模式切换

Toggle Parallel Mode：切换并联分支并联模式。并联模式分为顺序型并联分支和短路型并联分支。

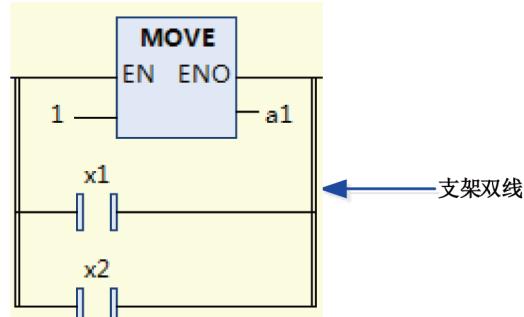
- 顺序型并联的并联支架使用单线，分支输出结果为单个分支输出取或操作，如下图，通过OR来形成分支结果。



- 短路型并联的并联支架使用双线，分支输出结果需要考虑每个分支是否包含非运算块。非运算块的分支作为条件，如果非运算块的分支有一个结果为True，都不去执行有运算块的分支（可以理解为触点短路运算块）。如下图，只有X1分支和X2分支结果都不是TRUE，才去执行第一个Move分支指令。

非运算块分支需要同时满足以下条件：

- 此分支只包含触点或者操作符运算块。
- 触点不能具有延信号。
- 操作符运算块不能是EnEno类型，操作符运算块输入连线不能包含取反、或者延信号。



## 说明

考虑到短路型分支的复杂性，不建议使用短路型分支。

## 设置分支起点/终点

设置分支起点/终点：图标 - ，快捷键：Ctrl + D。

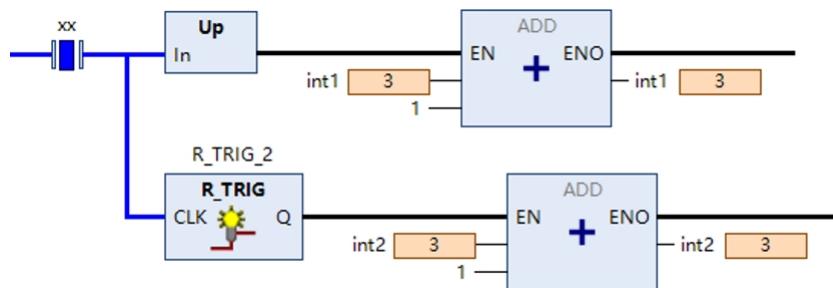
设置分支启动/终点命令用于把启动和终止连接起来，功能同划线功能。

连接两个点，首先执行此命令，设置起点，这时起点位置显示 ，表示连接开始点，然后再选择终点，执行此命令，这时会把启动和终点连接起来，具体连接逻辑见划线功能。

## 插入一个Up运算块

图标 ：表示插入一个Up指令块，可以实现不需要实例化声明也可以实现上升沿检测功能。

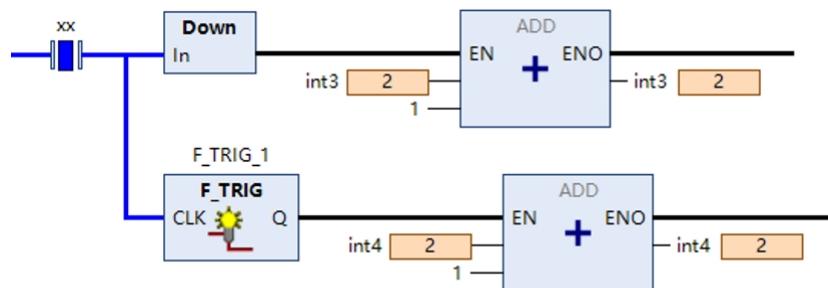
- 命令执行条件：选择位置为水平连线。
- 插入位置：选择水平连线，在水平连线处插入一个Up运算块。



## 插入一个Down运算块

图标 ，表示插入一个Down指令块，可以实现不需要实例化声明也可以实现下降沿检测功能。

- 命令执行条件：选择位置为水平连线。
- 插入位置：选择水平连线，在水平连线处插入一个Down运算块。



## 6.3.7 单键命令

单键命令通过单个字符快捷键进行快速编辑。可以在连线上执行单键命令，也可以在元素上执行单键命令。  
连线上执行单键插入串行元素；元素上单键命令用于插入并行元素或者元素功能切换。

每个命令对应的字符，可以在【选项】 - 【FBD/LD】 - 【LD】页面设置，具体见选项设置。

### 连线上执行的单键

- 插入触点：默认单键为“C”。
- 插入常闭触点：默认单键为“/”。
- 插入线圈：默认单键为“Q”。
- 插入复位线圈：默认单键为“R”。
- 插入置位线圈：默认单键为“S”。
- 插入空运算块：默认单键为“F”。
- 插入空EnEno运算块：默认单键为“E”。
- Set/Reset/延信号切换：默认单键为“空格”。用于运算块BOOL型输入、输出连线切换；当选择运算块BOOL型输入连线，执行延信号切换；选择运算块非主输出BOOL型连线，执行Set/Reset切换。

### 元素上执行的单键

- 插入并行触点：默认单键为“C”。选择的元素可以是触点、运算块。
- 插入并行空运算块：默认单键为“F”。选择的元素可以是触点、运算块。
- 插入并行空EnEno运算块：默认单键为“E”。选择的元素可以是触点、运算块。
- 插入线圈：默认单键为“Q”。选择的元素可以是线圈、返回、跳转元素。
- 元素取反切换：默认单键为“/”。选择触点进行常开和常闭触点切换；选择线圈对线圈进行取反切换。
- 元素Set/Reset/延信号切换：默认单键为“空格”。选择触点时，进行上升沿、下降沿和正常信号切换。选择线圈进行Set、Reset和正常线圈切换。

## 6.3.8 划线功能

划线功能主要是把划线起点和终点两个位置连接起来。划线功能首先要满足以下条件：

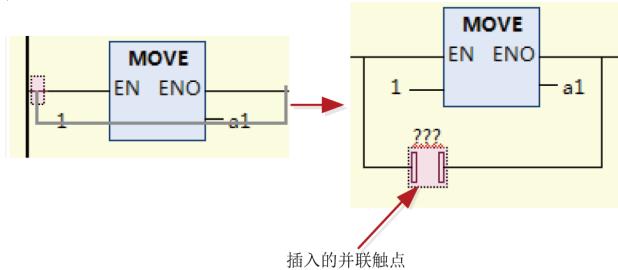
- 起点或者终点位置必须是连线，并且是可以选择的位置（能流线除外）。
- 运算块输入、输出引脚可以拖拽互换位置，所以靠近运算块引脚连线区域（大概11个像素）是拖拽引脚区域，不能划线。

从划线结果来看，划线功能分为三类：划线并联（增加并联分支），闭合分支、拆分分支。

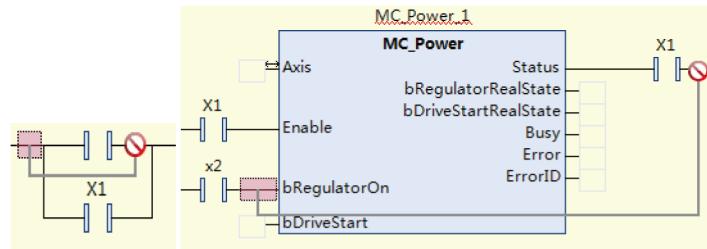
### 划线并联

当划线起点和终点在一个分支上时，自动增加一个触点和起点、终止并联，如下图：

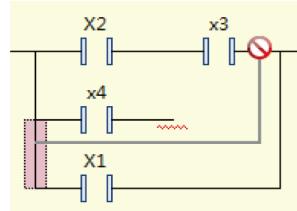
- 如果在一个分支内划线，在开始和结束位置之间自动并联一个触点；
- 如果从打开分支终止连线开始划线到其它分支，表示把打开分支闭合；
- 如果相邻两个上下划线，表示把上下两个分支拆分。



- 划线并联起点和终点必须满足可并联条件，不能跨越并联分支内外、不能从多连线运算块输入到输出。

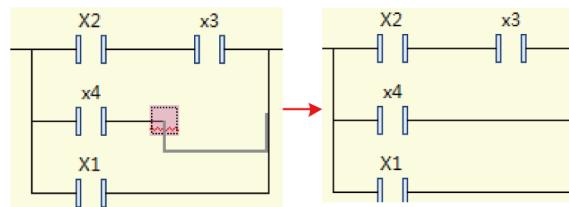


- 不能跨越打开分支。

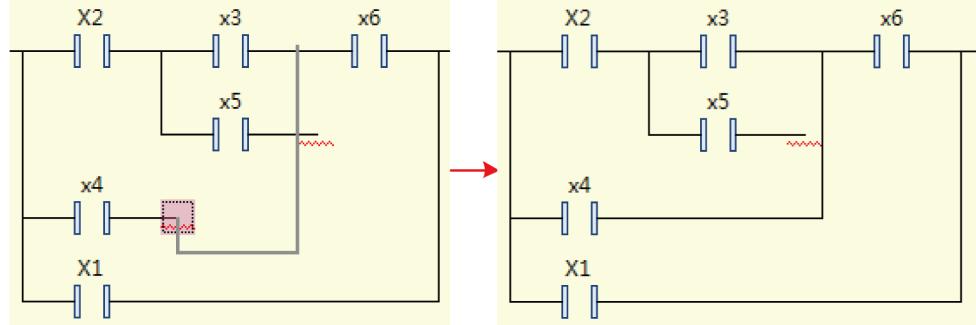


## 闭合分支

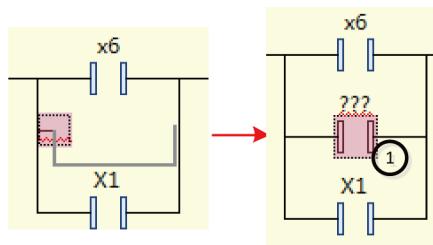
当划线起点为打开分支终止连线时，划线到另外一个可以闭合的连线上会把当前打开分支闭合，如下图：



- 闭合分支功能可以跨越打开的分支。

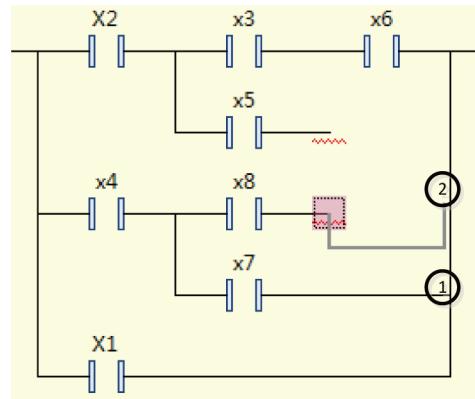


- 如果打开分支只有一个终止连线，闭合时，自动添加空触点。



①：自动添加空触点

- 如果打开分支对应的层级分支是闭合的，此打开分支只能和右侧垂直连线划线闭合。

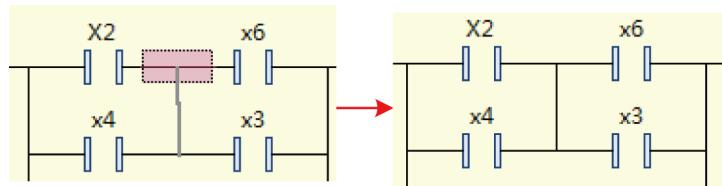


①：打开分支对应的层级分支是闭合的

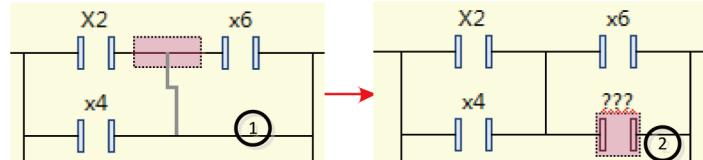
②：只能向右侧垂直连线划线闭合

## 拆分分支

当划线起点和终点为并联两相邻分支时，划线时会把起点、终点连线两侧拆分为两个并联。



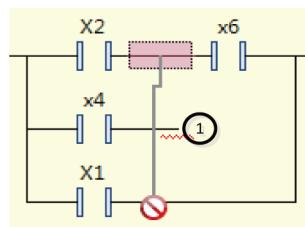
- 如果起点或者终点左侧或者右侧没有元素，自动在没有元素的一侧增加空触点。



①：右侧没有元素。

②：增加的空触点。

- 拆分分支不支持跨越打开分支。



①：打开的分支。

## 6.3.9 拖拽操作

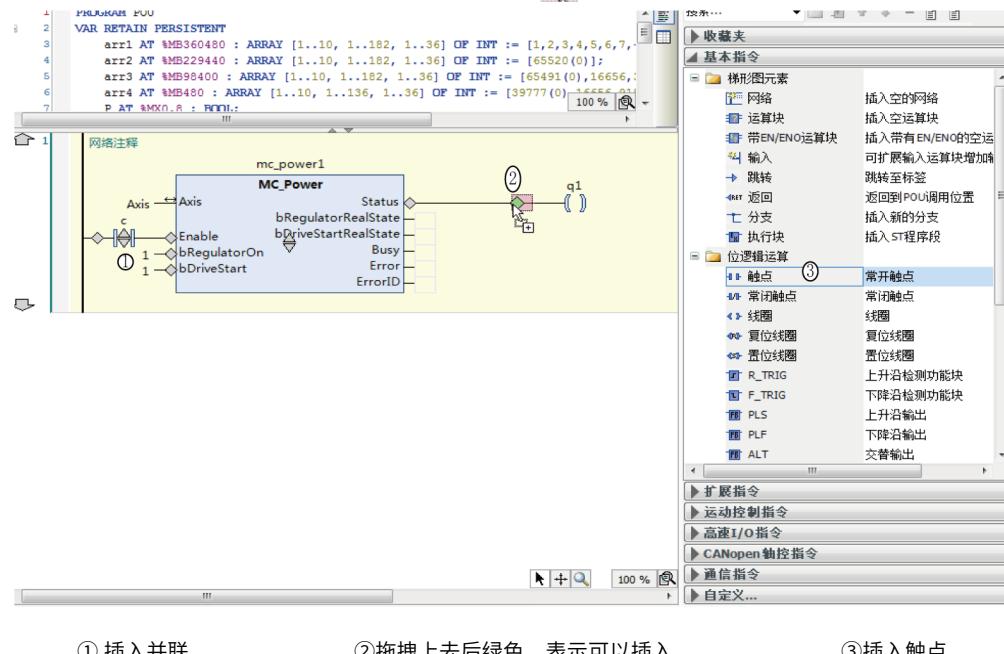
梯形图可以进行元素拖拽，主要包括工具箱拖拽元素到网络、梯形图界面内元素拖拽及跨界面拖拽。

当拖拽元素时，梯形图界面会显示可拖拽位置。可以拖拽位置有三种显示形式：

- 菱形显示：使用菱形 ，表示可拖拽到当前位置串联插入。
- 上下三角显示：使用上下三角箭头 ，表示在当前元素上方或者下方插入并联元素。

- 上下箭头显示：使用上下箭头，表示向上或者向下增加一个网络，并拖拽到新添加的网络。

当拖拽元素到插入位置时，每种图形内部会变成绿色，如，表示要插入到此位置。拖拽显示如下图所示。



## 工具箱元素拖拽

工具箱中的元素可以拖拽到梯形图编辑器中。

工具箱元素主要包含基本指令、扩展指令、运动控制、高速I/O, CANOpen轴控指令、通讯指令、POUs；另外用户可以自定义类别并添加指令，还可以把指令添加到工具箱中。

梯形图元素在基本指令中。

POUs主要包含当前工程中定义的程序、功能块、函数、方法、动作。最大显示不能超过200个，如果工程中超过200个，为防止显示混乱，不再显示POUs的内容。

拖拽时，每个元素有限定的拖拽位置，可拖拽规则如下：

- 触点可以拖拽到触点、运算块（包含执行块）上并联，拖拽到连线上串联。
- 运算块可以拖拽到触点、运算块（包含执行块）上并联，拖拽到连线上串联。
- 线圈可以拖拽到非闭合并联、非多连线运算块输入连线上串联，可以拖拽到线圈、返回、跳转上面或者下面。

## 编辑界面元素拖拽

在梯形图界面，可以拖拽选择的元素从一个位置到另外一个位置。拖拽元素可以在本编辑界面内，也可以拖拽到其它梯形图编辑界面。

拖拽的元素是选择的元素（见元素选择章节），可以多选、单选。

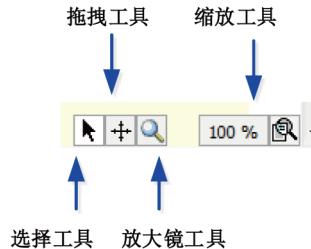
拖拽包含正常拖拽和复制式拖拽（按下Ctrl，拖拽）。正常拖拽，选择的元素拖拽过去后，会把原来选择的元素删除；复制式拖拽，选择的元素拖拽过去后，选项的元素保留。

拖拽功能都是按照标准操作方式实现。

于多选或者单选的拖拽规则，和标准编辑命令（粘贴）一致。

### 6.3.10 图形显示工具

梯形图图形显示工具，用来控制梯形图显示模式，包含选择工具、拖拽工具、放大镜工具和缩放工具，默认梯形图是选择工具模式。图形显示工具位于梯形图界面右下侧，如下图：



- 选择工具

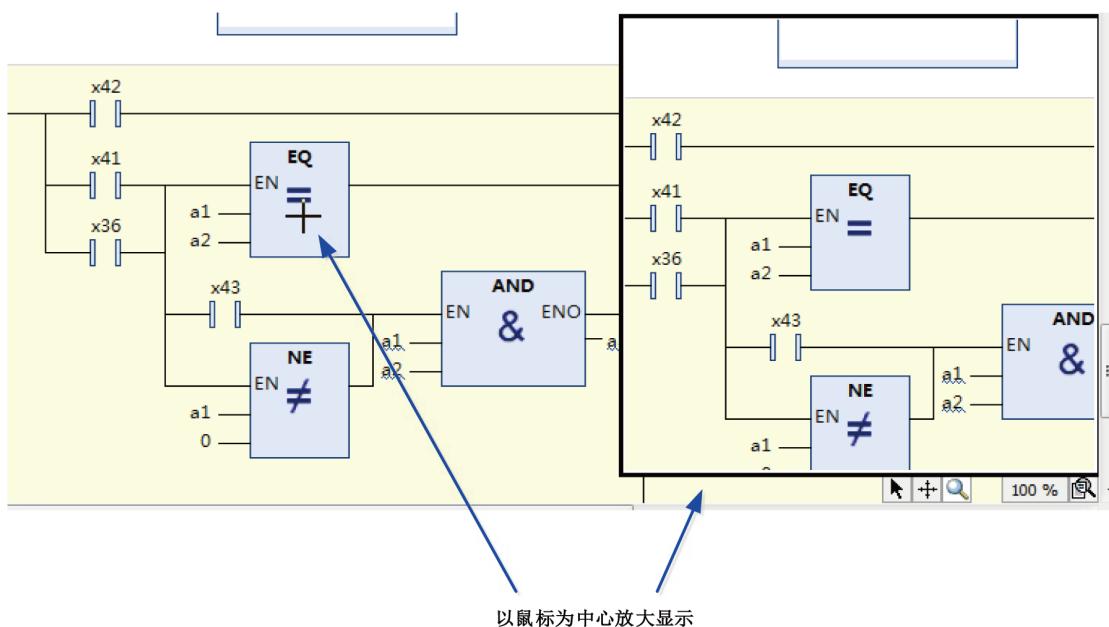
选择工具是默认显示工具，在选择工具模式下，鼠标样式为 $\text{手}$ ，可以进行选择元素，从而进行编辑操作。

- 拖拽工具

拖拽工具模式下，鼠标样式为 $\text{手+箭头}$ ，可以对区域进行拖拽显示操作。

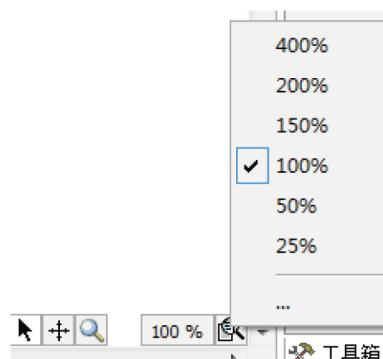
- 放大镜工具

放大镜模式下，鼠标样式为 $\text{放大镜}$ ，以鼠标为中心，进行放大显示，有放大镜作用。如下图：



- 缩放工具

缩放工具可以显示当前界面缩放比例，也可以设置缩放比例，如下图：



另外点击“...”，弹出缩放比例设置对话框，输入希望的缩放比例，如下图：

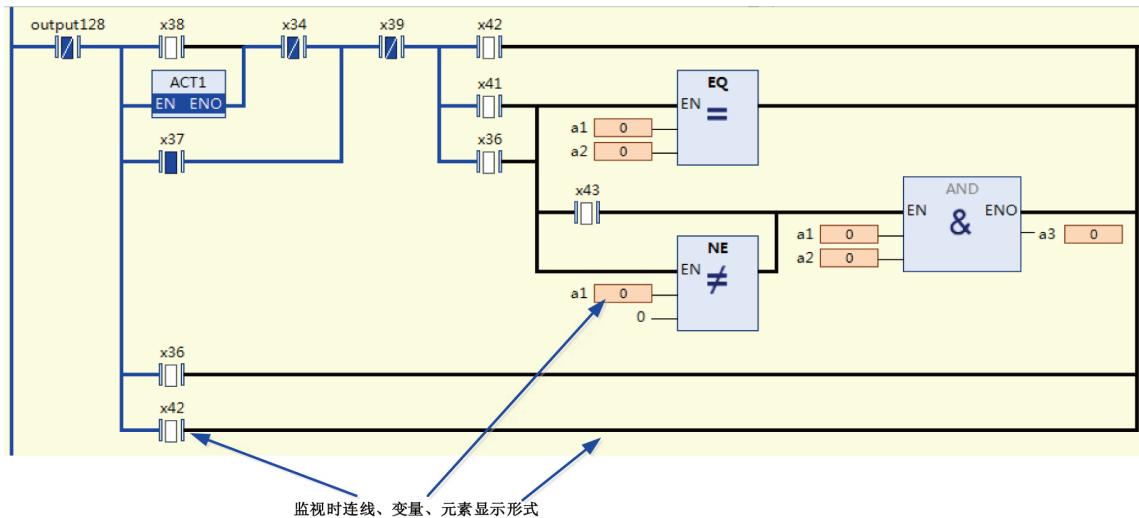


### 6.3.11 LD调试

梯形图提供了强大的调试功能，除了已有的监控表监控，梯形图还提供了在线模式下的监视、操作数写入、强制值写入、断点和单步调试功能。

#### 监视

在线模式下，梯形图界面中的连线、元素、操作数变量等通过特定的形式来表达执行结果。如下图。



- 监控连线

1. 对于BOOL型值连线，当导通时（值为TRUE）时，显示蓝色粗线，没有导通时显示黑色粗线。
2. 非BOOL型值连线（运算块输入、输出中的整形变量、时间类型变量、浮点数变量等），使用细线，并且值为零时使用黑色细线；不为零使用蓝色细线。

- 监控元素

1. 触点导通时，常开触点显示 || | 或者常闭触点显示 | | ||；触点不导通时，常开触点显示 | | | | 或者常闭触点显示 | | | | |。
2. 线圈导通时，正常线圈显示 ( ) 或者取反线圈显示 ( ) | ；线圈不导通时，正常线圈显示 ( ) | 或者取反线圈显示 ( ) | | 。
3. 对于EnEno运算块，由于EnEno运算块只有En为True，才执行运算块本身逻辑，为了使EnEno运算块本身能够一目了然了解此运算块是否执行（运算块是否使能），对运算块类型文本显示做了区分，如果此运算块执行了（En输入为True），运算块类型显示黑色字体，没有执行显示灰色字体（运算块禁用），如下图：

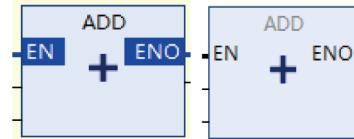


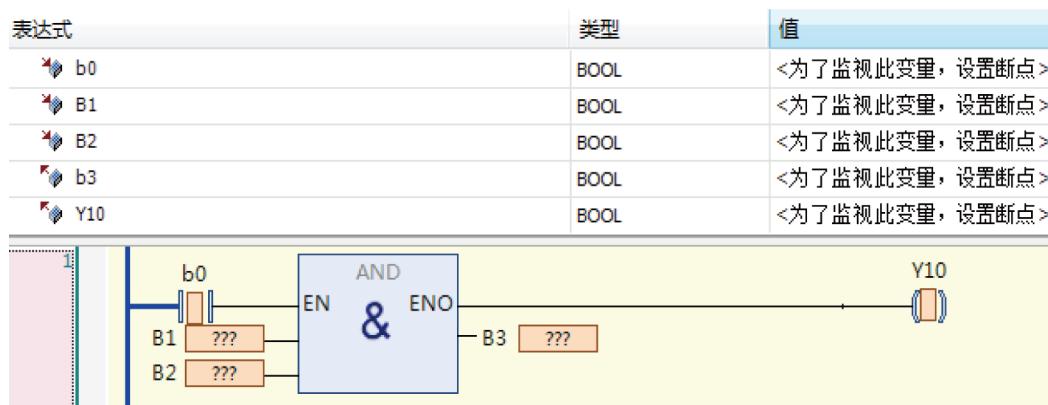
图6-6 运算块执行显示 运算块未执行显示

- 监控变量

1. 监控变量根据类型不同显示宽度不同，以减少占用空间，对于不定长的，如字符串、枚举类型（显示枚举名），默认长度为12个字符，如果显示不完，使用…替换，通过信息提示来完整显示；对于定长的，如整数、浮点数等根据最大长度显示。
2. 可以拖拽监控变量到监控变量列表。
3. 可以改变量显示模式，执行菜单命令：菜单【调试】 - 【显示模式】。

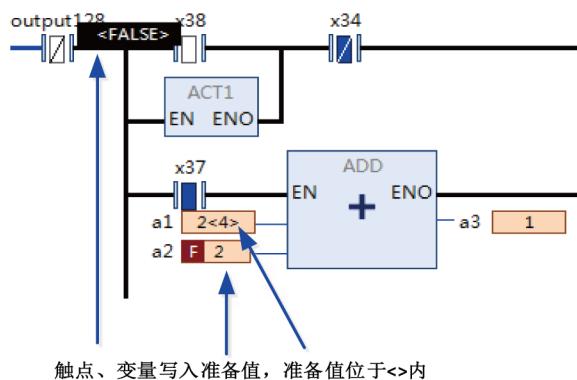
## 说明

由于函数和方法是即时执行，只有临时数据，所以登录后，方法和函数不能直接监控。如果需要监控函数和方法，需要在函数和方法中添加断点，中断执行，才能监视，如下图。



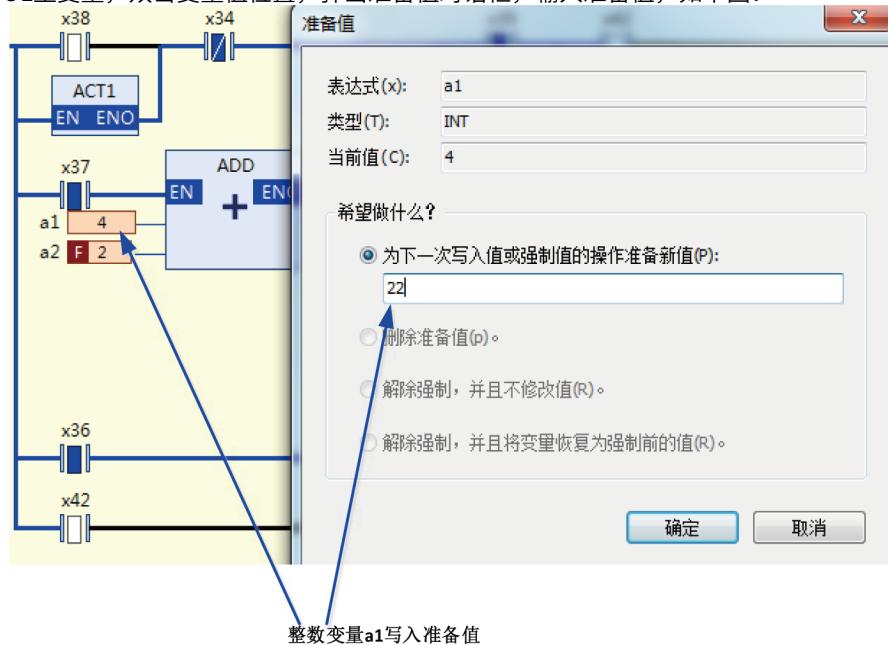
## 写入和强制

梯形图触点、线圈和变量可以写入准备值，然后执行调试菜单下的“写入值”命令或者“强制值”命令，给变量写入值或者强制值。在写值或者强制值之前，需写入准备值，如下图：



- 对于触点、线圈和BOOL型变量，通过双击元素位置或者变量值位置，进行TRUE、FALSE准备值切换。如双击触点或者线圈中间位置，准备值进行切换。

- 对于非BOOL型变量，双击变量值位置，弹出准备值对话框，输入准备值，如下图：

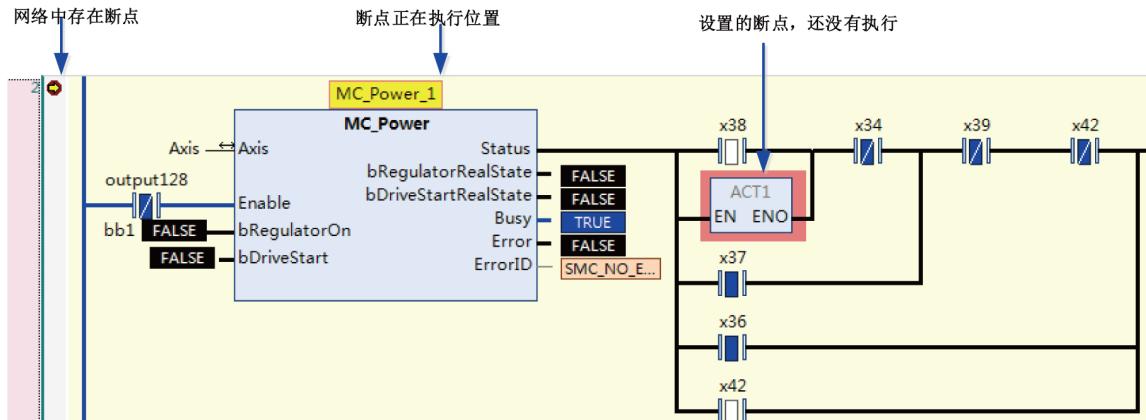


- 写入强制值后，值的最前方，增加**F**标识，标识此值是强制值。
- 把强制值释放，通过菜单命令【在线】-【释放值】

## 断点

LD支持断点功能，如果添加断点后，程序执行到断点位置自动中断，可以进行程序调试，支持跳入、跳过、跳出、运行到光标位置等操作。

添加断点后，添加断点的位置（元素）用一个浅红色的矩形框表示，当执行到断点位置，正在执行的断点位置用一个黄色的矩形框表示；如果网络中存在了断点，则网络装饰区域的圆点，如下图。



由于梯形图是图形化的，而断点在有逻辑语句的地方才可以添加，梯形图为了优化性能，并不是所有地方都会有逻辑语句，也就是，不是所有地方都能添加断点，例如触点位置、非EnEno操作符运算块位置不能添加断点。

断点一般在变量值可能发生变化的地方、程序的分支处或者另外一个POU调用的地方，如POU，输出变量赋值等地方。可以打开断点对话框（菜单【视图】-【断点】）查看所有可能的断点位置。

断点主要可以在以下位置添加：

- 网络开始位置，表示网络中第一个可能的断点位置，给网络增加断点时，自动增加到第一个断点位置。
- 不包括非EnEno操作符的运算块，如FB、动作、程序调用、执行块等。

- 线圈、返回、跳转元素位置。

### 6.3.12 梯形图数据更新

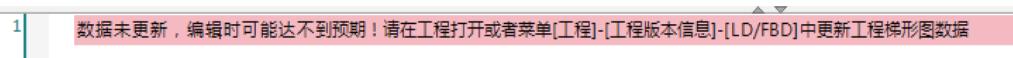
对于InoProShop(V1.4.0及之前的版本),如InoProShop(V0.0.9.10), InoProShop(V1.1.0), InoProShop(V1.2.0) InoProShop(V1.2.60.0), InoProShop(V1.2.70.1), 打开的梯形图数据需要升级,才能编辑使用梯形图优化版本的功能。

梯形图数据升级有两种方式:

- 在工程打开时,弹出的工程版本信息对话框-【LD/FBD】选项卡,选择下图中所有更新标志,然后点击“确定”按钮;
- 通过菜单命令【工程】-【工程版本信息…】,弹出工程版本信息对话框,切换到【LD/FBD】选项卡,选择下图中所有更新标志,然后点击“确定”按钮。



如果梯形图数据没有更新,在第一个网络,显示更新说明信息,如下图:



### 说明

使用前梯形图必须更新。

# 7 诊断

## 7.1 诊断简介

诊断是为了快速定位PLC运行过程出现的错误，通过错误信息和状态找出应对的解决方案。InoProShop上的诊断界面只有登录PLC后才能获取和显示。

InoProShop编程系统支持各种通信设备的诊断，可以根据各通信设备实际运行状态生成故障、离线等信息。

故障诊断涉及的模块类型主要包括：CPU模块、Modbus、ModbusTCP、EtherCAT、CANopen、CANlink、Profibus-DP等。

InoProShop编程系统主要提供了四种获取诊断方式：组态诊断、诊断信息列表、设备自身诊断信息列表和诊断编程接口。

所有诊断都是通过诊断码解析获取，诊断码和诊断编程接口相对应。

## 7.2 组态诊断

### 7.2.1 概述

组态分为网络组态和硬件组态，相应的诊断也分为网络组态诊断和硬件组态诊断。组态中，每个通信模块诊断状态通过不同的图标来呈现不同状态：运行状态、停止状态、离线状态和故障状态，即：

：运行状态，设备无故障。

：停止状态，设备没有运行，处于停止状态。

：离线状态，设备没有连接或者设备不存在。

：故障状态，设备处于故障状态，不能正常运行。

在组态中能直接看到设备的运行状态。

### 7.2.2 网络组态诊断

网络组态可以配置一个PLC总线系统，激活总线并添加从站。在登录状态下打开网络组态，可以看到其中各个通信设备的诊断状态，如下图所示：

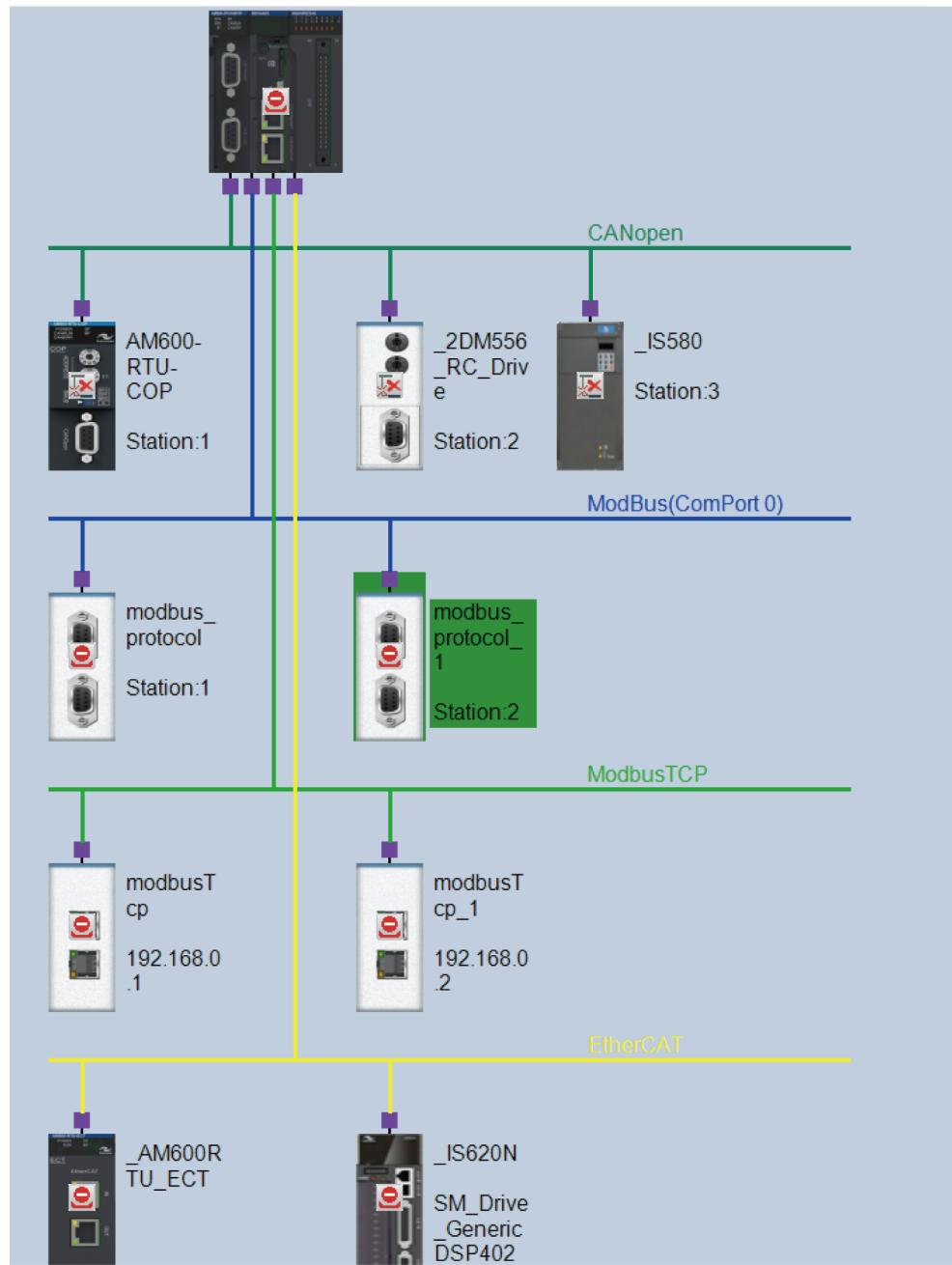


图7-1 网络组态诊断

登录后，网络组态中的每个从站或者CPU会显示运行、故障或者离线状态，如需了解网络组态操作，请参见硬件组态。

### 7.2.3 硬件组态诊断

硬件组态主要用于添加总线对应的扩展模块，包括本地IO硬件组态、EtherCAT硬件组态、CANopen硬件组态；而CANlink、Modbus和ModbusTCP只在网络组态中显示。双击网络组态子节点或者网络组态中的从站模块可以打开硬件组态，也可以在一个硬件组态中选择另外一个硬件组态。硬件组态诊断基本相似，CANopen硬件组态如下图所示：

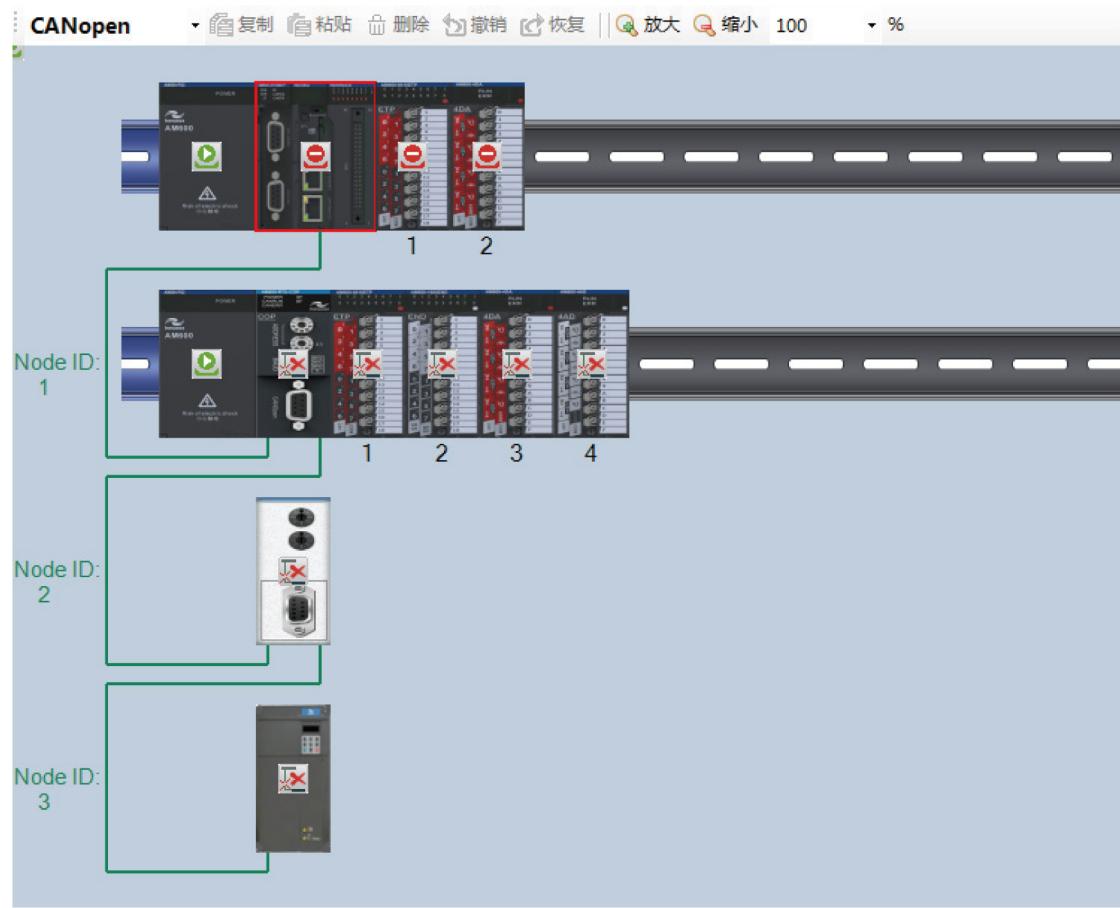


图7-2 CANopen硬件组态诊断

### 7.3 故障诊断

故障诊断用于显示所有设备出现的故障信息，并提供相关故障信息的详细说明、原因排查和解决措施；同时针对特殊情况还可以提供更详细的诊断信息。

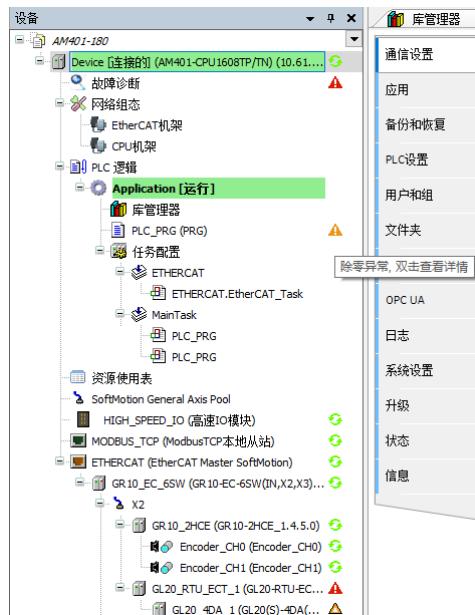
设备创建连接后，在左侧设备树中双击“故障诊断”，默认进入“实时诊断”界面，如下图所示。



故障诊断包括三个界面：

- 实时诊断：该登录设备目前存在的所有故障。
- 诊断缓冲区：该登录设备的所有历史故障。
- 用户事件：该登录设备与用户相关的所有历史操作记录。

如果设备存在故障，在设备树右侧出现故障图标，鼠标悬停时会提示出故障信息，双击故障图标将跳转到“实时诊断”界面，并定位到该设备的故障行。



## 实时诊断

- 单击“实时诊断”，进入“实时诊断”界面。
- 根据过滤条件筛选所需显示故障信息的模块。

- 在“模块”下拉选项选择所需显示故障信息的模块类型（所有、CPU模块、Modbus模块、ModbusTCP模块、本地模块和EtherCAT模块等），默认选择所有模块类型。
- 在文本框中输入某故障模块名或故障信息的关键字，单击“筛选”。

故障信息列表将根据设置的模块类型或筛选关键字进行显示，模块故障信息包括模块名称、故障信息和操作。

## 说明

- 刷新列表中模块故障信息：单击“刷新”，故障信息列表显示当前最新的模块故障信息。
- 清空列表中模块故障信息：单击“清空”。
- 导出列表中模块故障信息：单击“导出”。

3. 在故障信息列表单击某一条故障，界面下方显示“详细”和“深入诊断”信息，根据对应的诊断信息排查故障。

- 单击“详细”页签，该界面将显示原因排查、解决措施和更多的诊断信息。
- 单击“深入诊断”页签，该界面将显示某些复杂故障更多的诊断信息。

## 说明

EtherCAT部分相关故障才显示“深入诊断”页签信息，例如GL10和GL20模块。

4. (可选) 在“操作”列单击相应跳转链接，跳转至相应界面。

## 诊断缓冲区

1. 单击“诊断缓冲区”，进入“诊断缓冲区”界面。

界面顶部显示不同故障等级的故障数量，故障等级分为异常、错误和警告。

故障诊断					
实时诊断 诊断缓冲区 用户事件					
等级	时间	事件ID	位置	信息	操作
错误	2023-09-19 20:14:02	0x10050031	InoSV660N_7	从站链接断开，从站地址(1008)，别名地址(1040)，链接断开端口(1)	
错误	2023-09-19 20:14:00	0x10050031	GL20_RTU_ECT32	从站链接断开，从站地址(1010)，别名地址(333)，链接断开端口(0)	
错误	2023-09-19 20:14:00	0x10050031	InoSV660N_8	从站链接断开，从站地址(1009)，别名地址(1041)，链接断开端口(1)	
错误	2023-09-19 20:13:33	0x10F424200008	CmpDn	DeviceNet组件状态机进入故障状态，状态机为7	
错误	2023-09-19 20:13:33	0x10F424200003	CmpDn	DeviceNet组件状态机进入故障状态，状态机为7	
错误	2023-09-19 20:13:22	0x10F424200008	CmpDn	DeviceNet组件状态机进入故障状态，状态机为7	
错误	2023-09-19 20:13:22	0x10F424200003	CmpDn	DeviceNet组件状态机进入故障状态，状态机为7	
错误	2023-09-19 20:13:02	0x10F424200008	CmpDn	DeviceNet组件状态机进入故障状态，状态机为7	
错误	2023-09-19 20:13:02	0x10F424200003	CmpDn	DeviceNet组件状态机进入故障状态，状态机为7	
错误	2023-09-19 17:53:31	0x10050031	GL20_RTU_ECT32	从站链接断开，从站地址(1008)，别名地址(333)，链接断开端口(0)	
错误	2023-09-19 17:53:31	0x10050031	InoSV660N_6	从站链接断开，从站地址(1007)，别名地址(1039)，链接断开端口(1)	
错误	2023-09-19 17:52:33	0x10314E21	Axis_6	驱动器不在OP模式下，轴通信状态(1100)	
错误	2023-09-19 17:52:33	0x10314E21	Axis_5	驱动器不在OP模式下，轴通信状态(1100)	
错误	2023-09-19 17:52:33	0x10314E21	Axis_4	驱动器不在OP模式下，轴通信状态(1100)	

## 说明

如没有显示相关故障信息，单击“刷新”。

2. 根据过滤条件筛选所需显示的故障信息。

- 单击不同故障等级按钮，可切换该故障等级的故障信息显示或不显示。

- 在“组件”下拉选项选择所需显示故障信息的模块类型（所有、CPU模块、Modbus模块、ModbusTCP模块、本地模块和EtherCAT模块等），默认选择所有模块类型。
- 在“时间”文本框输入所需显示故障信息的起始时间和终止时间，在“内容”文本框输入故障信息的关键字，单击“筛选”。

故障信息列表将根据设置的模块等级、模块类型、筛选时间范围或筛选关键字进行显示，故障信息包括故障等级、发生时间、事件ID、设备位置、故障信息和操作。

## 说明

- 刷新列表中模块故障信息：单击“刷新”，故障信息列表显示当前最新的模块故障信息。
- 导出列表中模块故障信息：单击“导出”。

- 在故障信息列表单击某一条故障，界面下方显示详细信息，包括原因排查和解决措施，根据详细信息进行故障排查和解决。
- (可选) 在“操作”列单击相应跳转链接，跳转至相应界面。

## 用户事件

- 单击“用户事件”，进入“用户事件”界面。

界面顶部显示事件的数量，事件分为事件和信息两个等级，信息等级默认不显示。

等级	时间	事件ID	位置	信息	操作
事件	2023-09-19 20:15:49	0x10F420001203	Appolo	PLC开始运行	
事件	2023-09-19 20:15:48	0x10F420002006	Appolo	应用程序Application“热复位”完成	
事件	2023-09-19 20:15:48	0x10F420000101	Appolo	MRetain变量恢复“成功”	
事件	2023-09-19 20:15:48	0x10F42000200F	Appolo	应用程序Application退出完成，原因：“复位”	
事件	2023-09-19 20:15:48	0x10F420000100	Appolo	MRetain变量备份“成功”	
事件	2023-09-19 20:15:48	0x10F420000100	Appolo	MRetain变量备份“成功”	
事件	2023-09-19 20:15:48	0x10F420001208	Appolo	PLC停止运行	
事件	2023-09-19 20:15:48	0x10F420002004	Appolo	应用程序Application停止完成，原因：“操作停止运行”	
事件	2023-09-19 20:15:09	0x10F420001203	Appolo	PLC开始运行	
事件	2023-09-19 20:15:06	0x10F420002006	Appolo	应用程序Application“热复位”完成	
事件	2023-09-19 20:15:06	0x10F420000101	Appolo	MRetain变量恢复“成功”	
事件	2023-09-19 20:15:06	0x10F42000200F	Appolo	应用程序Application退出完成，原因：“复位”	
事件	2023-09-19 20:15:06	0x10F420000100	Appolo	MRetain变量备份“成功”	
事件	2023-09-19 20:15:06	0x10F420000100	Appolo	MRetain变量备份“成功”	
事件	2023-09-19 20:15:06	0x10F420001208	Appolo	PLC停止运行	
事件	2023-09-19 20:15:06	0x10F420002004	Appolo	应用程序Application停止完成，原因：“操作停止运行”	
事件	2023-09-19 20:14:54	0x10F420001203	Appolo	PLC开始运行	
事件	2023-09-19 20:14:53	0x10F420002006	Appolo	应用程序Application“热复位”完成	
事件	2023-09-19 20:14:53	0x10F420000101	Appolo	MRetain变量恢复“成功”	
事件	2023-09-19 20:14:53	0x10F42000200F	Appolo	应用程序Application退出完成，原因：“复位”	
事件	2023-09-19 20:14:53	0x10F420000100	Appolo	MRetain变量备份“成功”	
事件	2023-09-19 20:14:53	0x10F420000100	Appolo	MRetain变量备份“成功”	
事件	2023-09-19 20:14:53	0x10F420001208	Appolo	PLC停止运行	
事件	2023-09-19 20:14:53	0x10F420002004	Appolo	应用程序Application停止完成，原因：“操作停止运行”	

## 说明

如没有显示相关事件信息，单击“刷新”。

- 根据过滤条件筛选所需显示的事件信息。

- 单击“事件”，可切换事件等级的信息显示或不显示。
- 在“组件”下拉选项选择所需显示事件信息的类型，默认选择所有组件类型。
- 在“时间”文本框输入所需显示事件信息的起始时间和终止时间，在“内容”文本框输入事件信息的关键字，单击“筛选”。

事件信息列表将根据设置的事件等级、组件类型、筛选时间范围或筛选关键字进行显示，事件信息包括事件等级、发生时间、事件ID、组件位置、事件信息和操作。

## 说明

- 刷新列表中事件信息：单击“刷新”，事件信息列表显示当前最新的事件信息。
- 导出列表中事件信息：单击“导出”。

3. (可选) 在“操作”列单击相应跳转链接，跳转至相应界面。

# 7.4 在线诊断

## 7.4.1 概述

在线诊断指当用户连接PLC设备后，在线实时显示程序、设备及系统相关诊断信息。该功能用于帮助用户快速定位错误和及时解决问题，保证设备正常运行。

## 7.4.2 诊断流程

### 整体流程

首先通过设备扫描选择通信设备；待登录时，启动诊断信息监听功能；当收到诊断信息时，则进行数据刷新。



### 扫描流程

诊断功能的通信通道分为两种：**标准通信通道**和**诊断通信通道**。其中，标准通信通道即为现有通信通道，支持所有通信功能。诊断通信通道，则为当系统runtime出现异常，无法通过标准通信通道进行通信时，但PLC系统还可进行正常网络通信时，所采用用于诊断功能的通信通道。

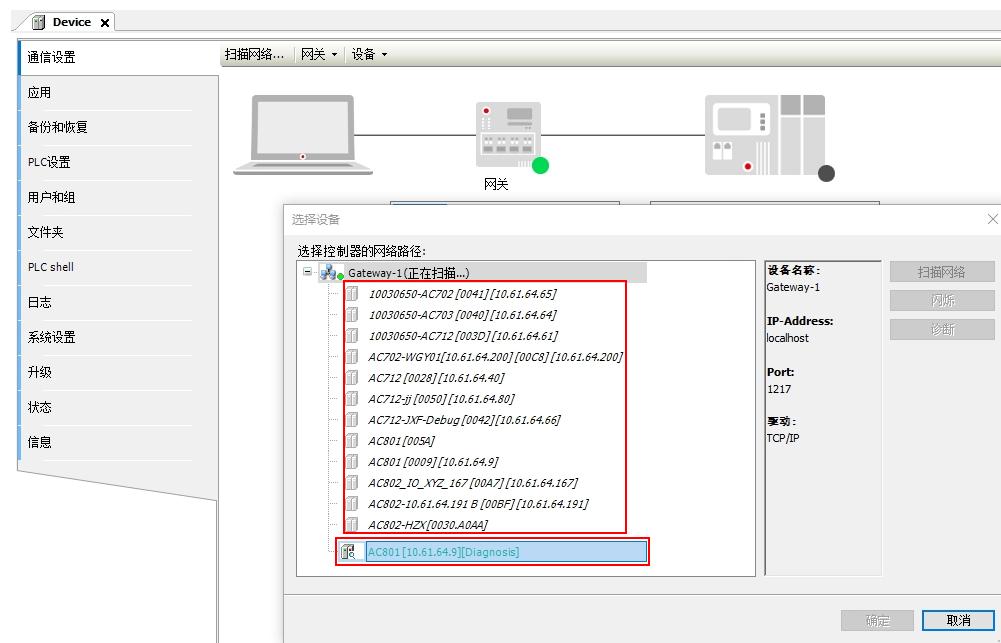
在进行设备扫描时，系统会同时打开两个通信通道进行设备扫描，系统显示标准通信通道所有扫描到的设备，及标准通信通道扫描不到但诊断通信通道可以显示的设备。

当用户选择哪个设备，系统会根据设备所属的通信通道进行通信。目前诊断通信通道只支持在线诊断功能和登录状态切换，其余通信功能暂不支持。不同通信通道支持的通信功能请参见下表。

通信功能	标准通信通道	诊断通信通道
程序下载	支持	不支持
在线修改	支持	不支持
监控	支持	不支持
日志刷新	支持	不支持
在线诊断	支持	支持
登录状态切换	支持	支持

### 7.4.3 扫描设备

单击“扫描网络”选项，打开“选择设备”界面，标准通信通道和诊断通信通道会同时启动设备扫描功能，将扫描到的设备显示在界面中，如下图所示。

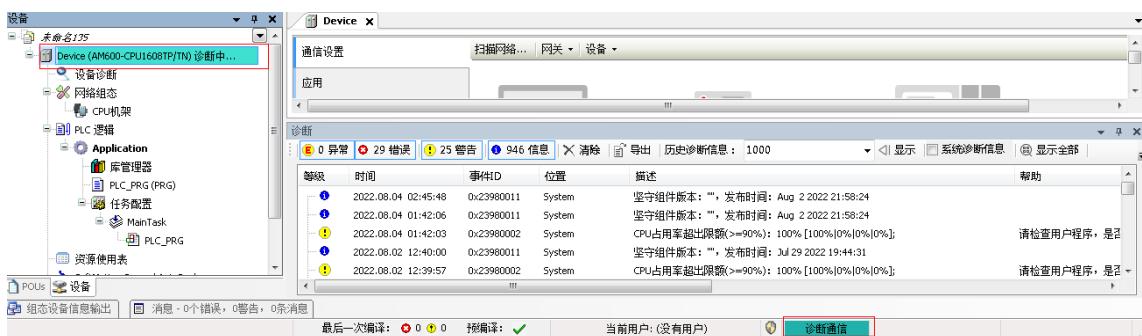


界面中黑色字体部分表示通过标准通信通道的扫描结果，淡绿色字体部分表示通过诊断通信通道的扫描结果。如果扫描的设备处于诊断通信通道，单击“诊断”，上载诊断日志，便于定位具体问题。

#### 说明

- 只有新版本才支持设备诊断扫描功能（版本标记为5），低版本不支持。
- 诊断通信通道不支持USB模式通信。

当选择诊断通信扫描设备后，状态栏的通信模式显示为“诊断通信”模式。当PLC登录时，设备与PLC建立通信，系统会启动诊断通信，此时设备树节点会显示“诊断中…”，系统会获取诊断数据，如下图所示。



由于诊断通道下的设备，出现故障早于通信，故障信息可能无法显示，可以通过查看历史诊断信息的方式获取诊断信息。

### 7.4.4 登录PLC

单击登录 ，当软件与PLC实现登录连接后，会启动诊断功能，诊断界面如下图所示。

## 诊断



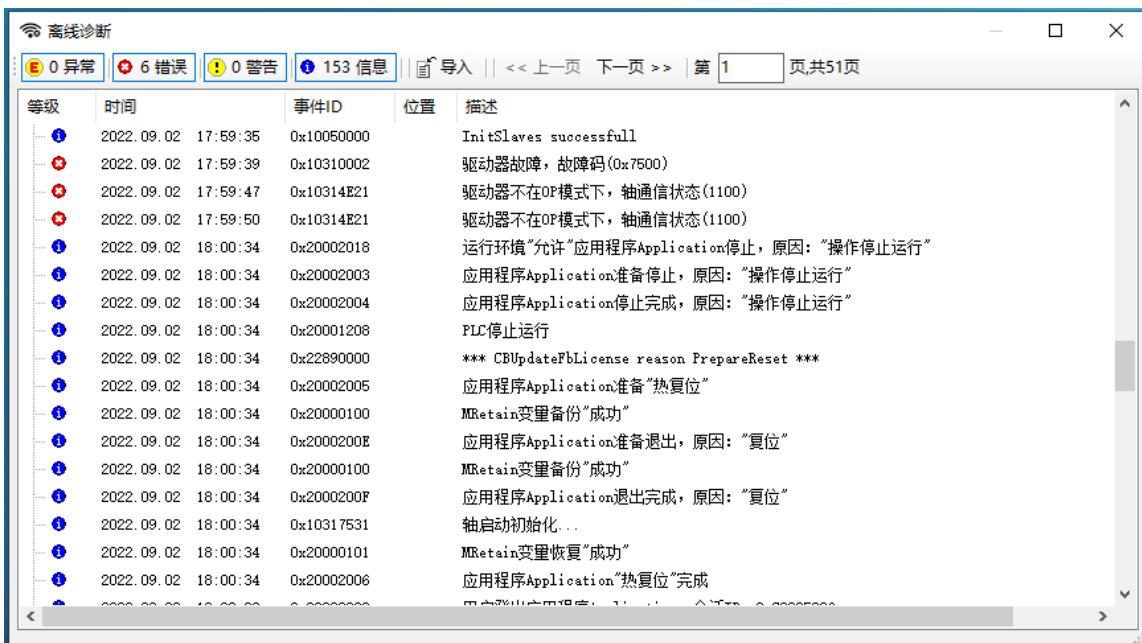
界面中工具条各参数含义请参见下表。

参数名称	参数说明
异常/错误/警告/信息	用于对诊断中不同等级信息进行显示和过滤操作。
清除	用于清除当前显示的诊断信息。清除后，再有新的诊断信息时，系统会刷新最新信息。
导出	用于导出所有诊断信息，导出文档格式为CSV格式。
历史诊断信息	用于查看登录之前的诊断信息。
系统诊断信息	在诊断信息中存在一些需要上载异常信息的诊断信息，系统将其标注为系统诊断信息。
显示全部	查看全部诊断信息。
离线诊断	用于在离线状态下导入保存的CSV格式的诊断信息。

界面中表格各参数含义请参见下表。

参数名称	参数说明
等级	显示信息的等级。
时间	显示信息发生的时间。
事件ID	显示信息的事件ID。
位置	显示故障的发生位置。当某个诊断信息支持跳转功能时，位置信息的字体则为下划线字体，双击该行，可以跳转到相应位置；当存在异常文件需要上载获取时，也可以通过双击所在行，上载异常文件。
描述	描述故障出现的现象和原因。
帮助	故障的解决方法或处理操作。

在线状态下可以通过导出功能，将在线诊断信息全部保存成CSV格式。之后可以通过离线诊断界面导入保存的数据，显示全部诊断信息，离线诊断界面如下图所示。



## 7.5 设备自身诊断信息列表

### 7.5.1 CPU 诊断

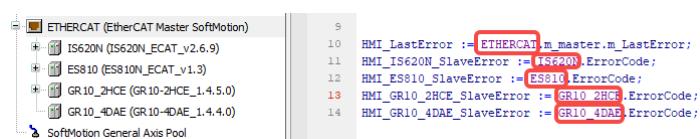
CPU本身不存在诊断界面，可以在诊断信息列表中查看诊断信息。

CPU诊断码及诊断信息请参见第507页“9.8.2 CPU 诊断码”。

### 7.5.2 EtherCAT 诊断

EtherCAT诊断用于记录和描述总线错误信息，其中包括主站诊断、从站诊断、从站模块诊断、从站伺服驱动器诊断。EtherCAT诊断仅对汇川技术的从站设备错误内容进行解析，诊断方法请参见[第442页“7.3 故障诊断”](#)，诊断的错误ID具体内容请参见[第512页“9.8.7 EtherCAT 诊断码”](#)。

有些应用场景需要通过触摸屏显示错误ID，只需把EtherCAT总线故障ID变量m\_LastError和EtherCAT从站故障ID变量分别赋值给关联HMI地址的变量即可。HMI\_LastError及HMI\_IS620N\_SlaveError等是关联HMI地址的一个WORD类型变量。触摸屏上即可显示EtherCAT诊断的总线故障ID和从站故障ID，如下图所示。



AM600 EtherCAT从站诊断

EtherCAT AM600从站对应的CANopen Emergency帧格式如下：

应急错误代码		错误寄存器	制造商特定的错误区域				
BYTE0	BYTE1	BYTE2	BYTE3	BYTE4	BYTE5	BYTE6	BYTE7
BaseInfo	SlaveError	0x80	InterCommError	ConformanceError		IOModulePosError	

## 说明

错误寄存器为0x80，表示从站Emergency帧。

其中BaseInfo暂时未使用，其余诊断码及诊断信息请参见第页“”。如果诊断格式符合此格式则解析为对应的诊断信息，否则解析为“设备出现故障”。

### 7.5.3 IO 诊断

I/O可以添加在CPU、CANopen AM600从站、Profibus-DP AM600从站和EtherCAT AM600从站下。其诊断信息基本相同，具体诊断码及诊断信息请参见第508页“9.8.3 IO 模块诊断码”。

关于自身诊断界面的描述，请参见设备自身诊断信息列表简介。

### 7.5.4 Profibus-DP 诊断

Profibus-DP诊断主要是指Profibus-DP从站诊断，数据在诊断数组中。每个从站都包含一个“从站诊断”界面，如下图所示，其中显示从站诊断信息。对于从站下的非AM600模块，诊断信息显示在此诊断界面中，对于AM600 Profibus-DP从站I/O模块，诊断信息显示在I/O模块本身界面中，详情请参见第450页“7.5.3 IO 诊断”。

对于Profibus-DP通道诊断，包括已定义的通道诊断和GSD文件定义的通道诊断，详情请参见Profibus-DP诊断简介中的通道诊断。

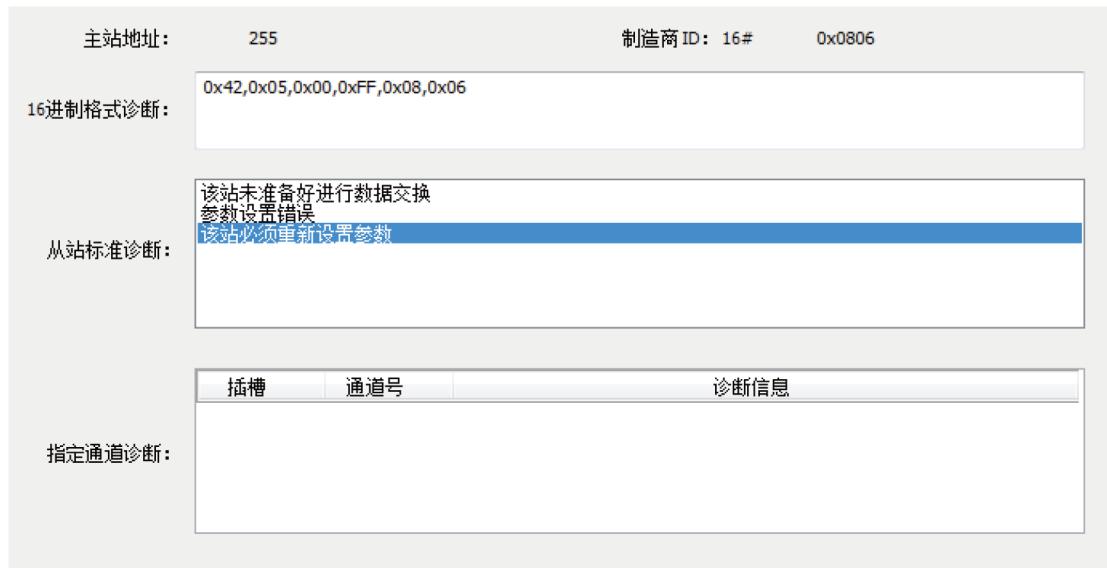


图7-3 Profibus-DP从站诊断

- 主站地址：主站地址，诊断数组中的第4个字节。
- 制造商ID：从站自定义的ID，诊断数组中的第5和第6个字节。GSD文件也定义了此ID。
- 16进制格式诊断：诊断数组数据16进制显示。
- 从站标准诊断：从站基本诊断信息和扩展诊断信息中状态诊断和标示符诊断，详见Profibus-DP诊断简介。

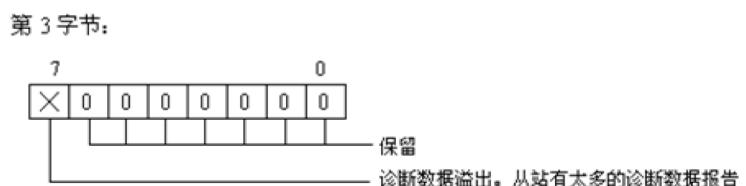
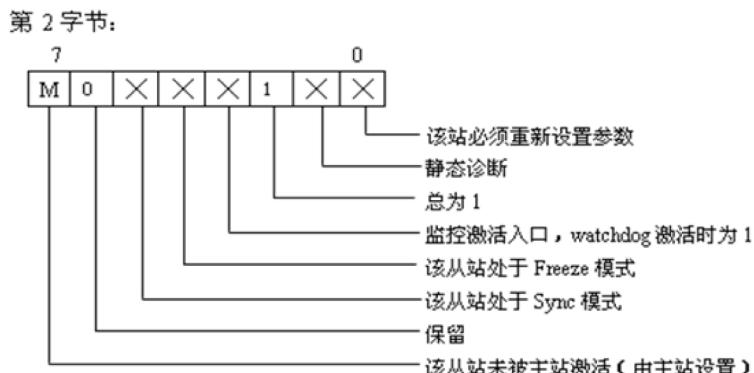
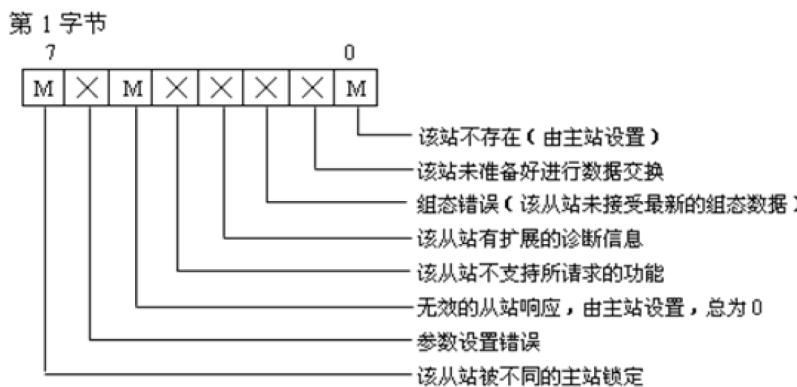
- 指定通道诊断：从站扩展诊断中的通道诊断，包括已定义的通道诊断和GSD文件定义的通道诊断，详见DP诊断简介中的通道诊断。

诊断数组结构如下表：

表7-1 诊断数组结构

前6字节为基本诊断信息（必选）	报警或状态信息块（4-63字节）（可选）	标志模块诊断信息块（可选）	通道诊断信息块（每个通道3个字节）（可选）
1----- ----- -6基本诊断信息部分	7----- -----		244 扩展诊断信息部分
DU单元最少6字节，最多可有244字节			

## 1. 基本诊断信息



第4字节：为主站地址。范围为0~7Dh(0~125)。当其值为FFh(255)时，表明该从站未被任何从站控制或未进行参数设置。

第 5 字节：该从站设备的 PROFIBUS ID 号高字节。范围为 0~FFh (0~255)。

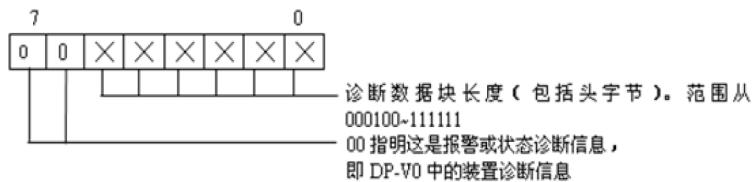
第6字节：该从站设备的PROFIBUS ID号低字节，范围为0~FFh（0~255）。

## 2. 扩展诊断信息

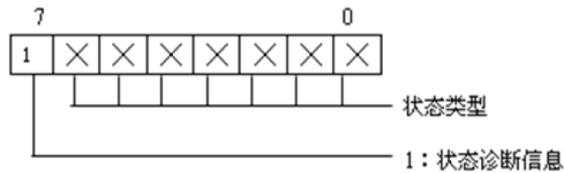
扩展信息诊断包括状态诊断、标示符诊断和通道诊断。

- 状态诊断

## 第7个字节



## 第8个字节



当位7为1时，指明的是状态诊断信息，这时位0~位6所指定的状态信息类型：

0：保留。

1：表示在状态详细特点信息字节后是状态信息。

2：表示在状态详细特点信息字节后是模块状态信息（影响第9字节后的字节内容）。

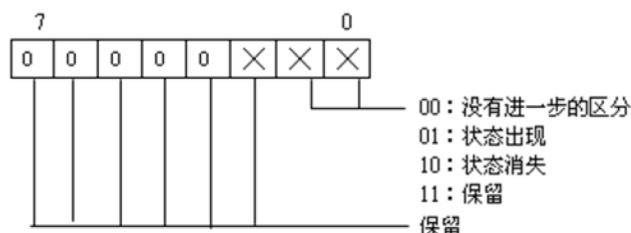
3~31：保留。

32~126：表示在状态详细特点信息字节后是制造商特殊数据。

127：保留。

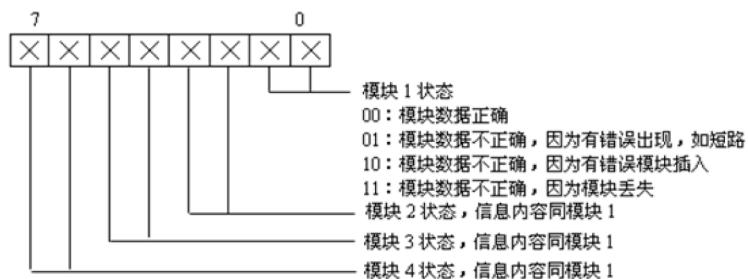
第9个字节：用来指明报告状态异常的从站设备的槽号，范围：0~254。

第10个字节：用来指定状态的详细特点。

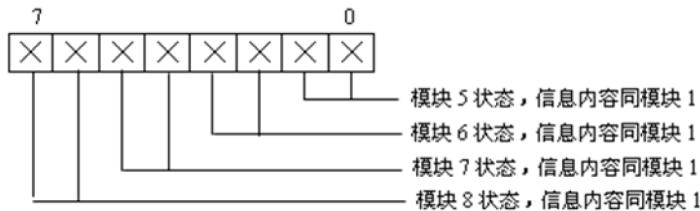


第11个字节以后：为用户数据字节。

若第8个字节中的状态类型为2，即模块状态信息，则第9字节为0，即从站槽号为0。从而第11字节以后就不再是用户数据字节了，其具体结构和含义如下：



第12字节：描述模块5~模块8的状态

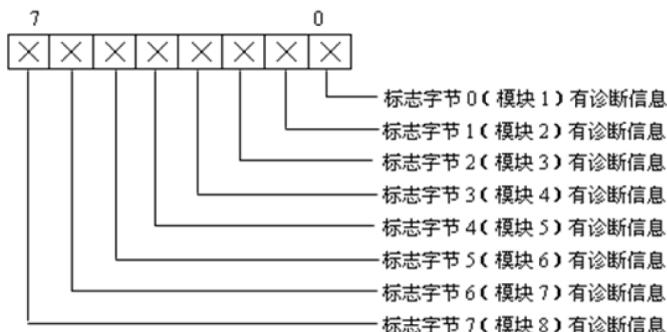
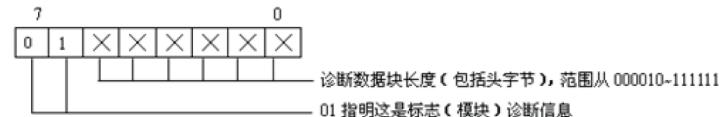


以后的字节可以依照上述的排列规则继续下去，直至将所有模块的信息写完。

#### ● 标示符诊断

头字节	和标志(模块)有关的诊断数据字节
01xxxxxx	1~62 字节

和上面的头字节类似，头字节指明“标志诊断信息”类型和长度(即有几个字节的诊断信息)，该长度包括头字节。头字节结构及含义如下：



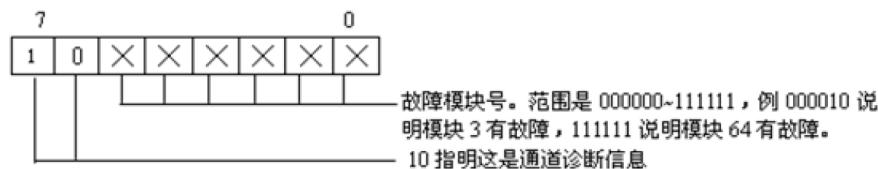
如果该设备的模块数多于8个，则可以继续使用接下去的字节指明标志字节号(或模块号)

#### ● 通道诊断

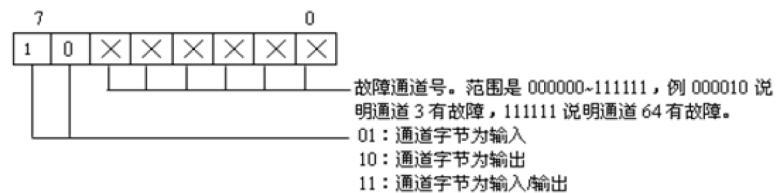
每个通道诊断信息由3个字节组成。通道诊断包含多个通道诊断信息，一个通道诊断信息的结构如下：

头字节	和通道有关的诊断数据字节
10xxxxxx	2字节(包括头字节有3个字节)

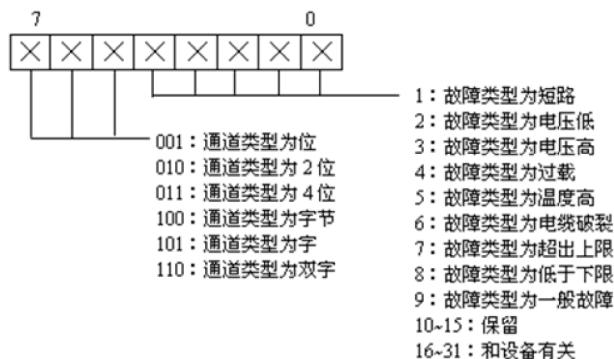
头字节指明“通道诊断信息”类型和发生故障的模块号。头字节的具体结构和含义如下：



第 2 个字节：说明通道字节类型，具体结构和含义如下：



第3个字节说明通道字节长度和故障类型，具体结构和含义如下：



### 7.5.5 ModbusRTU 诊断

ModbusRTU支持Modbus串口0和串口1两个串口总线。Modbus串口0或者串口1可以作为Modbus主站或者Modbus从站。

Modbus串口作为主站时，可以为主站添加从站（远程从站）。在主站和从站的配置界面中，都有一个“设备诊断”界面。主站诊断信息主要是用于标示从站配置项出现故障，不包含具体故障原因，因此“设备诊断”界面没有故障码；在从站“设备诊断”页面中详细说明了哪个配置项出现了具体的故障信息。

Modbus串口作为从站时，也有一个“设备诊断”界面，显示此从站和主站通信出现的故障，此界面显示内容详见设备自身诊断信息列表。

Modbus串口作为主站或者从站时其诊断码及诊断信息都是一致的，请参见[第511页 “9.8.6 Modbus 诊断码”](#)。

### 7.5.6 ModbusTCP诊断

AM600 PLC 可以作为ModbusTCP主站，也可以作为ModbusTCP从站。

ModbusTCP作为主站时，可以为主站添加从站（远程从站）。在主站和从站的配置界面中，都有一个“设备诊断”界面。主站诊断信息主要是用于标示从站配置项出现故障，不包含具体故障原因，因此“设备诊断”界面没有故障码；在从站“设备诊断”页面中详细说明了哪个配置项出现了具体的故障信息。

ModbusTCP作为从站时，也有一个“设备诊断”界面，显示此从站和主站通信出现的故障，此界面显示内容详见设备自身诊断信息列表。

ModbusTCP作为主站或者从站时其诊断码及诊断信息都是一致的，请参见[第511页 “9.8.6 Modbus 诊断码”](#)。

### 7.5.7 CANlink 诊断

CANlink本身没有“设备诊断”界面，但是在CANlink的网络管理功能中启动监控后，可以查看从站的在线和运行状态，详情请参见CANlink网络管理。

通过软元件可以获取CANlink站点状态，详情请参见[第274页 “4.9.4 CANlink 网络配置”](#)。

登陆PLC后，可以在诊断信息列表中查看CANlink的具体诊断信息。CANlink诊断码及诊断信息请参见[第273页 “4.9.2 CANlink3.0 网络组成”](#)。

关于自身诊断界面的描述，请参见设备自身诊断信息列表简介。

## 7.6 诊断编程接口

### 7.6.1 概述

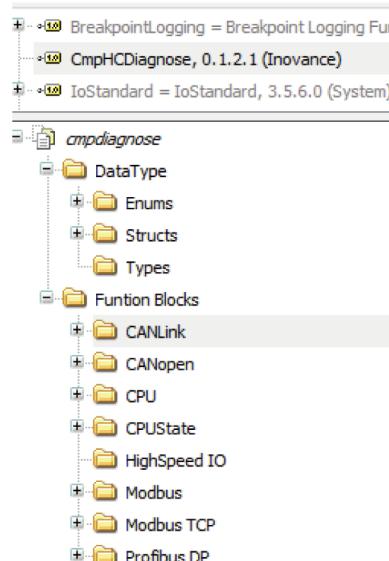
诊断接口库CmpHCDiagnose只支持AM400和AM600系列PLC，其中CANopen诊断功能块GET\_CANOPEN\_SALVE\_DIAGNOSE，V1.3.0及以后InoProShop软件版本不再适用，如需诊断从站状态，需添加CmpHCCiA402库，调用GET\_STATE接口。

SysHCPlcInfo库支持获取主机故障诊断、Modbus及ModbusTCP诊断、系统硬件及软件等相关信息。

### 7.6.2 诊断编程接口简介

诊断编程接口提供了在用户程序中获取诊断的解决方案：可以在用户程序中判断各个设备模块的诊断信息，从而作出相关的处理。

诊断编程接口以库的形式存在，可以在“库管理器”添加，添加后如下图所示。



编程接口中提供了CANlink、CANopen、CPU、Modbus、ModbusTCP、Profibus-DP对应的诊断编程接口，每种诊断对应一组功能块，用于获取对应的诊断码。

在获取诊断数据时，自定义的诊断结果和诊断状态在“DataType”中定义。获取每个诊断时一般都有一个HC\_error类型表示诊断是否获取成功，具体的定义见下表。

枚举器	值（十进制）	说明
NO_ERROR	0	无错误。
WRONG_PARAMETER	1	参数错误。
UNKNOWN_DEVICEID	2	无此设备ID。
INVALID_DEVICEID	3	设备ID无效。
INVALID_IO_POS	4	无效IO位置。
UNSUPPORT_DIAGNOSE	5	不支持诊断。
TIME_OUT	6	超时。

枚举器	值（十进制）	说明
INTERNAL_FB_ERROR	7	内部功能块错误。
UNKNOWN_ERROR	8	未知错误。
INVALID_IP	9	无效IP。

### 7.6.3 CPU 诊断编程接口

#### CPU自身诊断编程接口

获取CPU诊断数据 GET_CPU_DIAGNOSE			
<b>GET_CPU_DIAGNOSE</b> —xEnable BOOL —xDone BOOL HC_enumERROR eError HC_tagDIAGNOSE_DATA_CPU sCPUDiagnoseData			
参数名称	参数类型	初始值	参数作用
输入参数			
xEnable	BOOL	FALSE	功能块的使能位，电平触发
输出参数			
xDone	Bool	FALSE	获取诊断结果是否完成
eError	HC_enumERROR	NO_ERROR	获取诊断结果是否成功
sCPUDiagnoseData	HC_tagDIAGNOSE_DATA_CPU		CPU诊断数据

HC\_tagDIAGNOSE\_DATA\_CPU为结构体数据类型，如下表，每个数据诊断码和诊断信息关系，详细信息请参见[第507页“9.8.2 CPU 诊断码”](#)。

名称	类型
SDCardError	BYTE
FlashError	BYTE
SystemError	BYTE
InterCommError	BYTE
ConformanceError	WORD
IOModulePosError	WORD
FunctionErrorCode	WORD

#### 示例

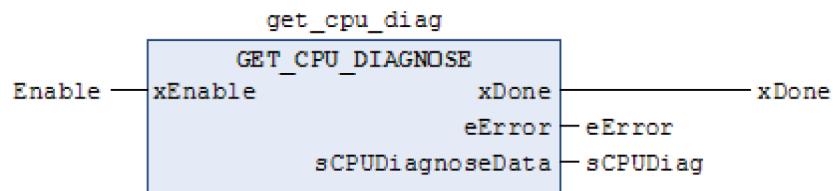
PROGRAM POU

VAR

```

get_cpu_diag: GET_CPU_DIAGNOSE;
  Enable: BOOL;
  eError: HC_enumERROR;
  xDone: BOOL;
  sCPUDiag: HC_tagDIAGNOSE_DATA_CPU;
END_VAR

```



## CPU本地IO扩展模块

获取CPU下IO诊断 GET_CPU_IOMODULE_DIAGNOSE			
<b>GET_CPU_IOMODULE_DIAGNOSE</b> <input type="checkbox"/> xEnable BOOL <input type="checkbox"/> byModulePos BYTE(1..16) <input type="checkbox"/> xDone BOOL <input type="checkbox"/> HC_tagDIAGNOSE_DATA_IOMODULE eError HC_enumERROR <input type="checkbox"/> sIODiagnoseData HC_tagDIAGNOSE_DATA_IOMODULE			
参数名称	参数类型	初始值	参数作用
输入参数			
xEnable	BOOL	FALSe	功能块的使能位, 电平触发
byModulePos	BYTE (1..16)	0	获取的IO位置
输出参数			
xDone	Bool	FALSE	获取诊断结果是否完成
eError	HC_enumERROR	NO_ERROR	获取诊断结果是否成功
sIODiagnoseData	HC_tagDIAGNOSE_DATA_IOMODULE		IO诊断数据

HC\_tagDIAGNOSE\_DATA\_IOMODULE为结构体数据类型, 如下表, 每个数据诊断码和诊断信息关系, 详细信息请参见[第508页“9.8.3 IO 模块诊断码”](#)。

结构成员	类型	说明
ModuleError	BYTE	模块错误。
ChannelError	ARRAY[0..3] OF BYTE	通道错误。

### 示例

PROGRAM POU

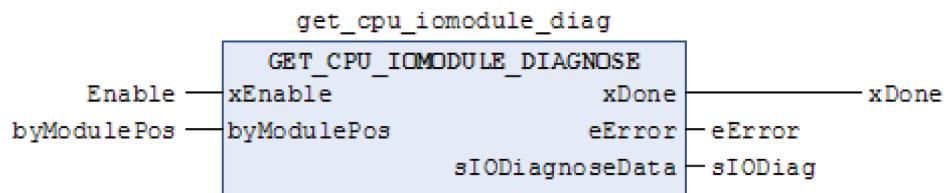
VAR

```

get_cpu_iomodule_diag: GET_CPU_IOMODULE_DIAGNOSE;
Enable: BOOL;
eError: HC_enumERROR;
xDone: BOOL;
byModulePos: BYTE (1..16);
sIODiag: HC_tagDIAGNOSE_DATA_IOMODULE;

```

END\_VAR



## 7.6.4 CANopen 诊断编程接口

如需诊断CANopen从站通讯状态，请在库管理器中添加CmpHCCiA405库，调用GET\_STATE接口进行获取。

## 7.6.5 Profibus-DP 诊断编程接口

### Profibus-DP从站诊断编程接口

获取DP从站诊断数据 GET_DP_SLAVE_DIAGNOSE			
GET_DP_SLAVE_DIAGNOSE		BOOL xDone HC_enumERROR eError HC_tagDIAGNOSE_DATA_SLAVE_DP sSlaveDiagnoseData	
参数名称	参数类型	初始值	参数作用
输入参数			
xEnable	BOOL	FALSE	功能块的使能位，电平触发
bySlaveID	BYTE (1..125)	0	获取诊断从站的站地址，范围1-125
输出参数			
xDone	Bool	FALSE	获取诊断结果是否完成
eError	HC_enumERROR	NO_ERROR	获取诊断结果是否成功
sSlaveDiagnoseData	HC_tagDIAGNOSE_DATA_SLAVE_DP		DP从站诊断数据

HC\_tagDIAGNOSE\_DATA\_SLAVE\_DP为结构体数据类型，如下表，每个数据诊断码和诊断信息关系，详细内容请参见[第509页 “9.8.4 DP 诊断码”](#)。

结构成员	类型	说明
Length	BYTE	诊断数据长度。
ExtDiagData	ARRAY[0..243]OF BYTE	诊断数据。

### 示例

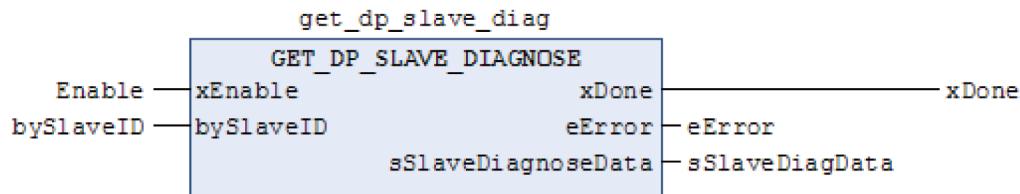
PROGRAM POU

VAR

```

get_dp_slave_diag: GET_DP_SLAVE_DIAGNOSE;
Enable: BOOL;
eError: HC_enumERROR;
xDone: BOOL;
bySlaveID: BYTE (1..125);
sSlaveDiagnoseData: HC_tagDIAGNOSE_DATA_SLAVE_DP;
END_VAR

```



## Profibus-DP从站下IO诊断编程接口

获取DP从站IO诊断 GET_DP_IOMODULE_DIAGNOSE				
参数名称		参数类型	初始值	参数作用
输入参数				
xEnable	BOOL	FALSE	功能块的使能位, 电平触发	
bySlaveID	BYTE (1..125)	0	从站的站地址, 范围1~125	
byModulePos	BYTE (1..16)	0	获取诊断的IO位置	
输出参数				
xDone	Bool	FALSE	获取诊断结果是否完成	
eError	HC_enumERROR	NO_ERROR	获取诊断结果是否成功	
sIODiagnoseData	HC_tagDIAGNOSE_DATA_IOMODULE		IO诊断数据	

HC\_tagDIAGNOSE\_DATA\_IOMODULE为结构体数据类型, 如下表, 每个数据诊断码和诊断信息关系, 详细内容请参见[第508页“9.8.3 IO 模块诊断码”](#)。

ModuleError	BYTE
ChannelError	ARRAY [0..3] OF BYTE

### 示例

PROGRAM POU

VAR

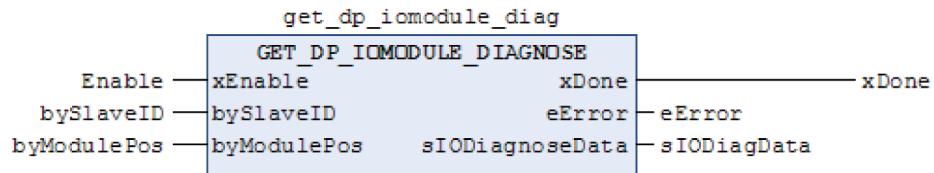
```

get_dp_iomodule_diag: GET_DP_IOMODULE_DIAGNOSE;

Enable: BOOL;
eError: HC_enumERROR;
xDone: BOOL;
bySlaveID: BYTE (1..125);
byModulePos: BYTE (1..16);
sIODiagData: HC_tagDIAGNOSE_DATA_IOMODULE;

```

END\_VAR



## 7.6.6 CANlink 诊断编程接口

获取CANlink诊断数据 GET_CANLINK_DIAGNOSE			
GET_CANLINK_DIAGNOSE			
xEnable ---	BOOL ---	BOOL xDone ---	
byStationID ---	BYTE (1..63) ---	HC_enumERROR eError ---	
		HC_tagDIAGNOSE_DATA_CANLINK sCanlinkDiagnoseData	
参数名称			
输入参数			
xEnable	BOOL	FALSE	功能块的使能位，电平触发
byStationID	BYTE (1..63)	0	获取的站节点ID，范围1-63
输出参数			
xDone	Bool	FALSE	获取诊断结果是否完成
eError	HC_enumERROR	NO_ERROR	获取诊断结果是否成功
sCanlinkDiagnoseData	HC_tagDIAGNOSE_DATA_CANLINK		CANlink站点诊断数据

HC\_tagDIAGNOSE\_DATA\_CANLINK为结构体数据类型，如下表所示，每个数据诊断码和诊断信息关系，详细内容请参见[第510页 “9.8.5 CANlink 诊断码”](#)。

结构成员	类型	说明
IsUsed	BOOL	是否使用。
IsMaster	BOOL	是否是主站。
StationStatus	WORD	CANlink站状态。
CfgFrameError	WORD	配置帧错误。
CmdFrameError	WORD	命令帧错误。

示例

PROGRAM POU

VAR

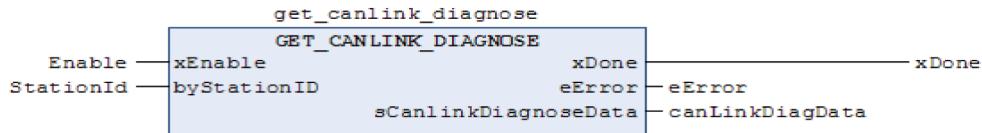
```

get_canlink_diagnose:GET_CANLINK_DIAGNOSE;

Enable: BOOL;
StationId: BYTE (1..63);
eError: HC_enumERROR;
canLinkDiagData: HC_tagDIAGNOSE_DATA_CANLINK;
xDone: BOOL;

```

END\_VAR



## 7.6.7 ModbusRTU诊断编程接口

- ModbusRTU本地从站诊断编程接口，具体请参见《中型PLC指令手册》中“SysHC\_ModbusRtuDeviceDiagnose”指令。
- ModbusRTU远程从站诊断编程接口，具体请参见《中型PLC指令手册》中“SysHC\_ModbusRtuSlaveDiagnose”指令。

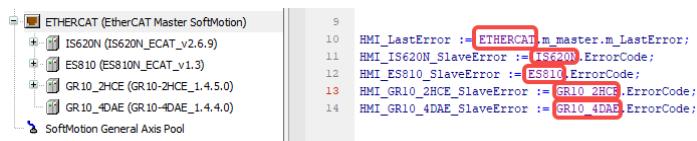
## 7.6.8 ModbusTCP 诊断编程接口

- ModbusTCP本地从站诊断编程接口，具体请参见《中型PLC指令手册》中“SysHC\_ModbusTcpDeviceDiagnose”指令。
- ModbusTCP远程从站诊断编程接口，具体请参见《中型PLC指令手册》中“SysHC\_ModbusTcpSlaveDiagnose”指令。

## 7.6.9 EtherCAT 诊断编程接口

EtherCAT诊断用于记录和描述总线错误信息，其中包括主站诊断、从站诊断、从站模块诊断、从站伺服驱动器诊断。EtherCAT诊断仅对汇川技术的从站设备错误内容进行解析，诊断方法请参见[第442页 “7.3 故障诊断”](#)，诊断的错误ID具体内容请参见本文附录。

有些应用场景需要通过触摸屏显示错误ID，只需把EtherCAT总线故障ID变量m\_LastError和EtherCAT从站故障ID变量分别赋值给关联HMI地址的变量即可。HMI\_LastError及HMI\_IS620N\_SlaveError等是关联HMI地址的一个WORD类型变量。触摸屏上即可显示 EtherCAT诊断的总线故障ID和从站故障ID，如下图所示。



## 7.6.10 CPU 停止控制

### 功能块描述

停止应用程序 STOP_APPLICATION									
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td colspan="2" style="text-align: center;">STOP_APPLICATION</td></tr> <tr> <td style="width: 20%;">xExecute</td><td style="width: 20%;">BOOL</td><td style="width: 20%;">BOOL</td><td style="width: 20%;">xDone</td></tr> </table>				STOP_APPLICATION		xExecute	BOOL	BOOL	xDone
STOP_APPLICATION									
xExecute	BOOL	BOOL	xDone						
参数名称	参数类型	初始值	参数作用						
输入参数									
xExecute	BOOL	FALSE	功能块的使能，上升沿触发						

停止应用程序 STOP_APPLICATION			
输出参数			
xDone	BOOL	FALSE	执行完成输出

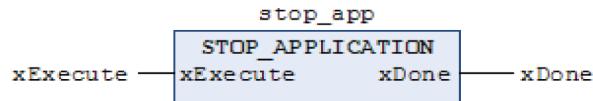
## 功能块示例

PROGRAM POU

VAR

```
stop_app: STOP_APPLICATION;
xExecute: BOOL;
```

END\_VAR



## 7.6.11 轴诊断

### 错误码

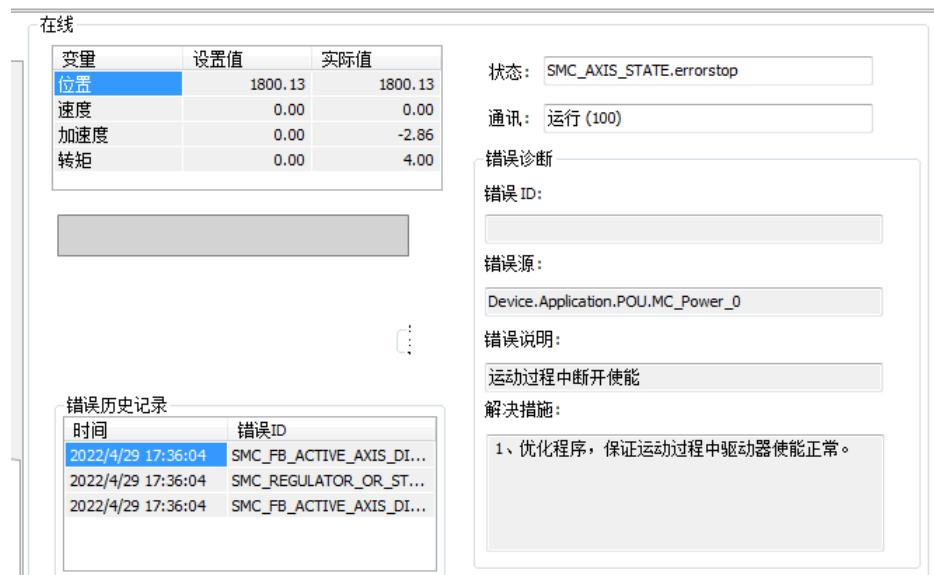
轴诊断用于记录和描述轴在初始化、启动和运行过程中的各种相关错误信息，比如从站通讯错误信息、轴运行过程中轴内部错误，如软限位、伺服报警，或者功能块使用不合理的报警等信息。轴诊断的错误 ID 具体内容请参见本文附录。

轴故障按照等级划分错误、警告和信息。

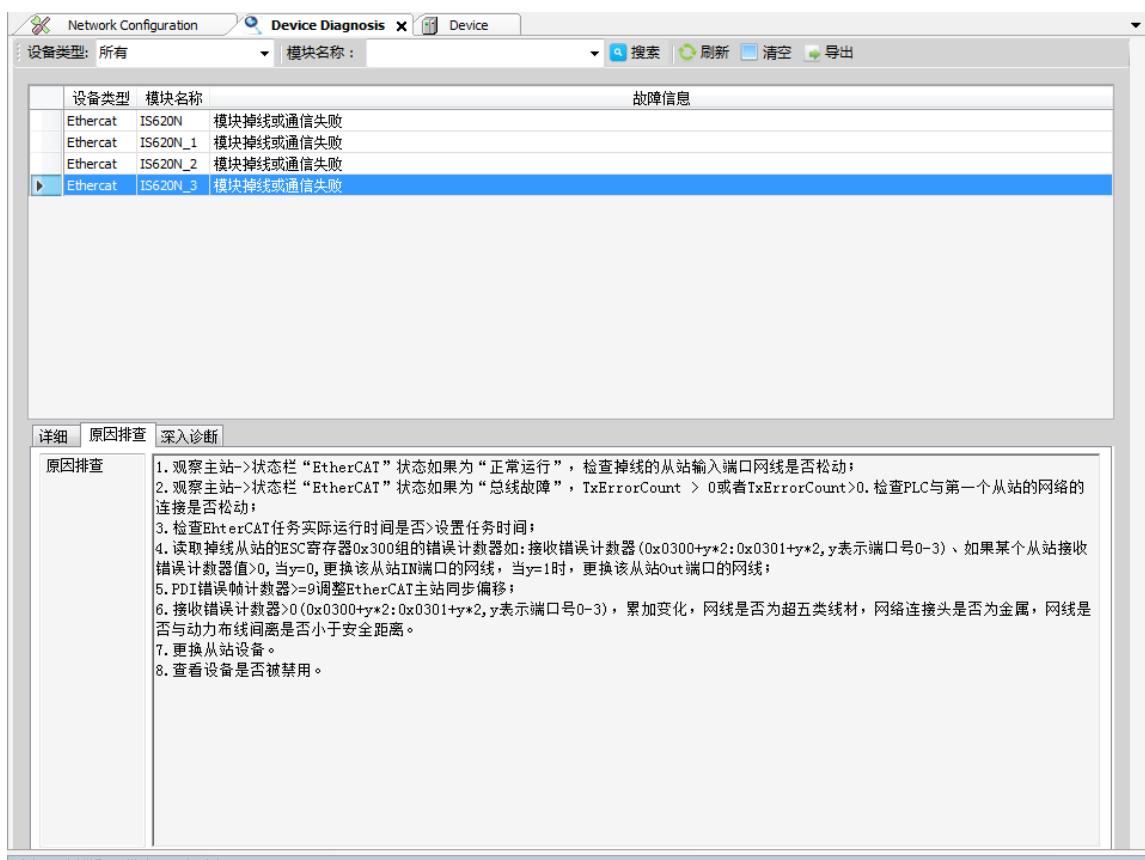
- **错误（严重）**：伴随轴停止的运行错误，轴处于运动中，运动被中断，总线错误，伺服故障，功能块运行错误，SDO 通讯等。
- **警告（一般）**：不伴随轴停止的运行错误，standstill、errorstop、poweroff 下的运动指令输入产生的错误，或者不影响动作功能及当前状态的其他内部错误。
- **信息（记录）**：启动记录、其他关键过程记录。

### 诊断界面

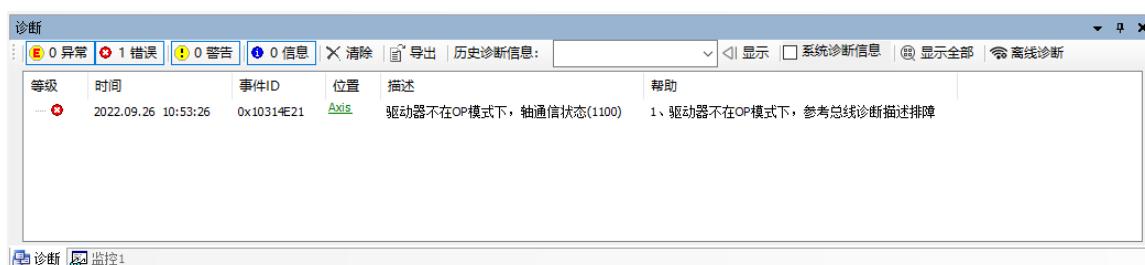
轴/设备诊断界面（在线）如下图所示。



设备诊断界面如下图所示。



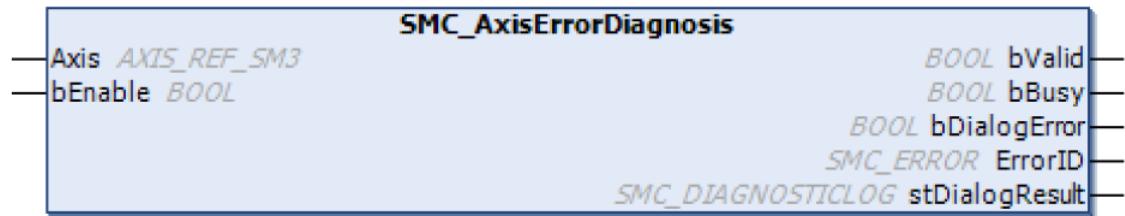
快速诊断接口如下图所示。



诊断结果结构体SMC\_DIALOGRESULT（快速诊断）请参见下表。

变量成员	数据类型	初始值	变量说明
DeviceType	UINT	-	SoftMotion
uiModuleID	UINT	-	轴ID标识
tTimeStamp	TIME	0	运行时间戳
ErrorID	SMC_ERROR	-	错误ID编号
eErrorClass	SMC_ERRORCLASS	-	故障等级（严重、中等、轻微）
eErrorType	SMC_ERRORTYPE	-	错误类型
StErrorInstance	STRING(60)	-	错误源
strErrorNotes	STRING(60)	-	错误说明
strErrorShooting	STRING(255)	-	解决措施

诊断功能块如下图所示。



- 功能说明

轴的实时诊断信息，诊断信息保存在诊断结果结构体SMC\_DIALOGRESULT中。

- 输入输出参数说明

输入输出变量说明请参见下表。

输入输出变量	名称	数据类型	取值范围	初值	描述
Axis	轴	AXIS_REF_SM3	-	-	映射到轴，即AXIS_REF_SM3的一个实例

输入变量说明请参见下表。

输入变量	名称	数据类型	取值范围	初值	描述
Enable	执行条件	BOOL	TRUE, FALSE	FALSE	输入高电平启动功能块的诊断功能

输出变量说明请参见下表。

输出变量	名称	数据类型	取值范围	初值	描述
Valid	完成	BOOL	TRUE, FALSE	FALSE	指令完成时为TRUE
Busy	执行中	BOOL	TRUE, FALSE	FALSE	当前指令正在执行中，置为TRUE
Error	错误	BOOL	TRUE, FALSE	FALSE	异常发生时，置为TRUE
ErrorID	错误代码	DWORD	-	0	异常发生时，输出错误代码
stDialogResult	诊断结果	SMC_DIALOGRESULT	-	-	诊断结果结构体

# 8 FAQ

## 8.1 CPU占有率过高

### 8.1.1 CPU 占有率定义

0%~89%：PLC运行比较稳定。逻辑执行、总线同步、IO刷新、数据同步、数据保存都有时间保证。

90%~100%：PLC运行稳定性降低。主要影响：

- EtherCAT运行稳定性难以保证，可能出现EtherCAT从站掉线、同步丢失。
- 严重的可能使PLC处于“假死”状态，不能扫描登录PLC。
- 掉电存储数据不能保存。
- CANopen、CANlink、Modbus/TCP存在数据刷新、断线风险。
- 在线修改或者下载PLC程序可能变慢，并且有可能失败。
- 监视的PLC变量值，存在刷新缓慢或者无法刷新的风险。

#### 说明

CPU占有率问题，目前适用于AM600、AM400系列PLC，对AC800、AP700系列暂不定义。

### 8.1.2 分析步骤

1. 查看PLC CPU占有率。

登录PLC，通过后台状态条能查看CPU占有率；如下图。



2. 查看任务执行时间，并计算执行时间在任务中占比。

登录PLC后，打开【任务配置】 - 【监视】界面，查看任务的平均循环时间，如下图。

任务	状态	IEC-循环计数	循环计数	最后循环时间(μs)	平均循环时间(μs)	最大循环时间(μs)	最小循环时间(μs)	抖动(μs)	最小抖动(μs)	最大抖动(μs)
ETHERCAT	有效的	1878500	1879064	264	272	824	11	734	-704	30
MainTask	有效的	1876537	1876537	3623	3575	3673	3471	111	-61	50

#### 说明

如果登录之前已经打开，登录后再次打开时，需要右键任务，执行菜单命令【复位】，恢复初始计算状态。

上图中，EtherCAT任务和MainTask任务循环周期都是4ms，而MainTask任务占比约3575/4000约等于89%，也就是说，MainTask占有过多的执行逻辑。

3. 优化任务中程序。

优化程序首先找到执行时间过长的程序，然后再找到程序中执行比较久的代码段。

找到占用过多CPU时间的程序，一般通过删减任务下的程序来判断。如果删除任务下的程序后，任务执行时间明显减少，表示此程序可能需要优化。

找到程序后，需要找到此执行比较久的代码，也是通过删除程序中的代码来判断。

### 8.1.3 常见优化方式

- 增大任务扫描周期  
任务扫描周期增加后，任务中程序执行次数减少，相应的占用CPU时间会减少。
- 批量数据处理代码优化  
一般程序是循环执行的，对于批量数据的处理，可以考虑多个周期处理。例如初始化代码、对时效性要求不是很高的逻辑，都可以多周期执行。
- 增加IF条件  
程序中功能块和函数，如果不增加条件，每个周期都会一直执行的。实际情况下，可能需要某个条件才需要执行，可以增加IF条件，满足条件才执行。在ST中可以考虑增加IF条件，在LD中，把运算块变为EnEno类型。
- 更换更高性能PLC

## 8.2 PLC运行异常

### 8.2.1 概述

汇川技术中型PLC（AM系列、AC系列）开发语言是基于IEC61131-3国际标准而设计的一种编译型语言。

编译型语言不同于解析型语言（小型PLC一般使用），编译型语言的程序在执行之前需要一个专门的编译过程，把程序编译成为机器语言的文件，运行时不需要重新翻译，直接使用编译的结果。编译型语言程序编写方式灵活、执行效率高，对编程人员能力要求相对较高（有C/C++基础比较合适）。

用户在编写程序时需要注意指针非法访问、除数为0、数组越界、类型隐式转换、死循环、全局变量保护等规避事项，否则用户程序将大概率造成PLC运行异常，甚至死机。

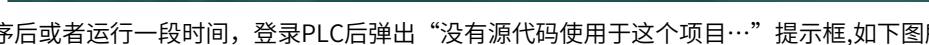
本文档重点介绍PLC运行用户程序异常(下载异常、死机)的主要原因、定位步骤、解决方法。

### 8.2.2 现象描述

#### 说明

- 问题定位后，建议手动删除所有隐含检查函数，因为隐含检查函数会消耗一部分CPU资源。
- AM系列PLC的固件版本必须在V1.22.0.0及以上。编程软件InoProShop版本必须在V1.3.2及以上。

- AM系列LED数码管显示停止刷新（正常情况显示“00”），AC系列液晶显示屏显示内容“Runtime crash”。
- 编程软件无法扫描到相应的PLC设备，PLC重新上电后正常，运行一段时间又无法扫描到PLC。
- 下载程序后或者PLC运行一段时间，登录PLC后信息显示栏，提示程序“停止”，错误内容“程序下载-异常”，如下图所示。  

- 下载程序后或者运行一段时间，登录PLC后弹出“没有源代码使用于这个项目…”提示框，如下图所示。  




### 8.2.3 原因分析及解决方法

#### 指针非法访问

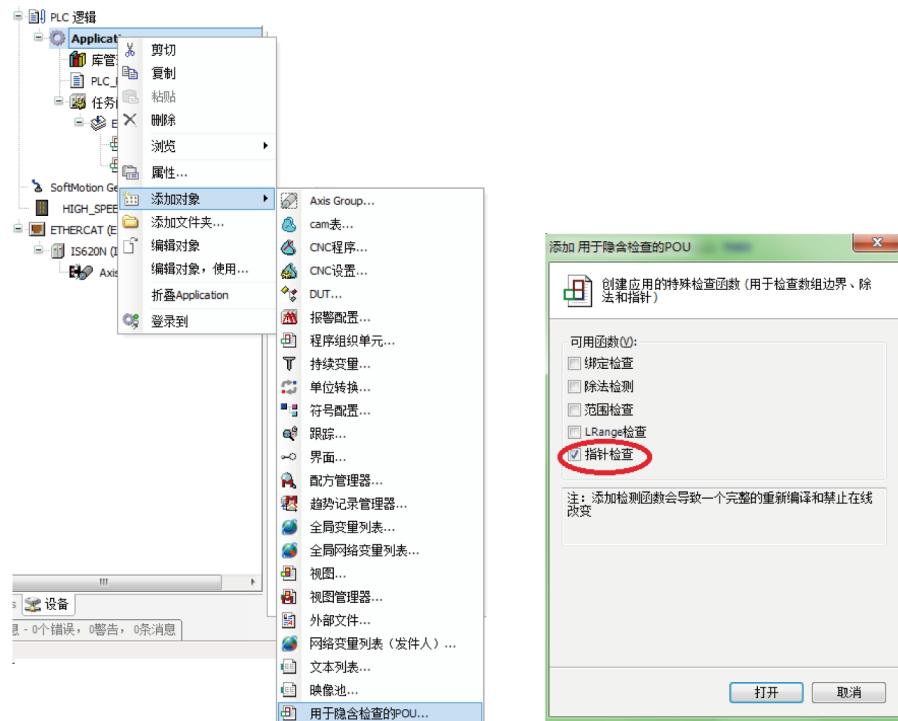
分为两种情形：空指针（指针指向地址值为0x00000000）、指针指向不合法区域（指针指向的地址与操作系统内部地址冲突）。

PLC操作系统不能执行空指针（地址0x00000000是一些单片机系统启动地址，会导致单片机软重启），空指针比较容易定位。

指针指向不合法区域会与系统其他正在运行的程序冲突，执行异常，这一类指针比较难查找原因，建议初级用户能用数组尽量用数组，不要使用指针。

- 定位步骤

在设备树中右键单击“Application”，选择“添加对象 > 用于隐含检查的POU”，并勾选“指针检查”。如下图所示：



添加“指针检查”后，编程软件会在“Application”设备树下默认添加“CheckPointer”函数，在函数中手动添加调试代码，如下图所示：

```

1 // 没有标准的实现方式。在此处输入代码
2 FUNCTION CheckPointer : POINTER TO BYTE
3 VAR_INPUT
4     ptToTest : POINTER TO BYTE; // 指针的目标地址
5     iSize : DINT;             // 指针指向的类型大小。（例如：POINTER TO ARRAY [0..9] OF INT为20）
6     iGran : DINT;             // 指针访问的粒度，即指针指向的类型中最大的非结构数据类型的大小。（例如：POINTER TO ARRAY [0..9]
7     bWrite : BOOL;            // 指示读或写权限，TRUE=写权限。
8 END_VAR
9
10
11 // 没有标准的实现方式。在此处输入代码
12 IF ptToTest = 0 THEN           新增的代码
13     CheckPointer := ptToTest;
14 END_IF
15
16 CheckPointer := ptToTest;

```

用户程序每执行一次指针调用，系统将默认执行“CheckPointer”隐含检查函数一次。通过在函数内部添加程序断点的调试方法，可以定位空指针在用户程序中使用的位置（断点调试方法请参考附录）。

登录PLC，在新增代码处“CheckPointer := ptToTest;”右键添加断点并激活，手动启动运行PLC(AM系列调试阶段需要将PLC的运行开关拨到“STOP”，如果是AC系列，需要在小屏幕设置为“程序开机运行”），目前只能定位指针为0的情况，如下图。



- 解决方法

用户程序指针调用处增加指针非0判断条件，如下图：

```

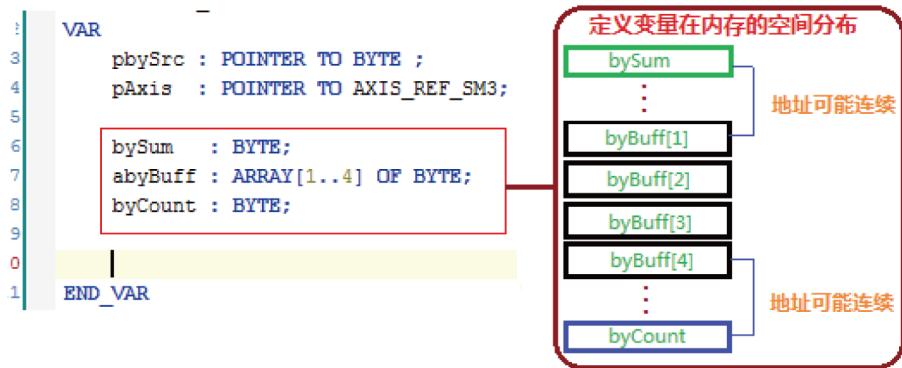
1 IF ( pAxis <> 0) THEN
2     IF pAxis^.nAxisState = SMC_AXIS_STATE.standstill THEN
3         // 用户执行代码
4     END_IF
5 END_IF
6

```

## 数组越界

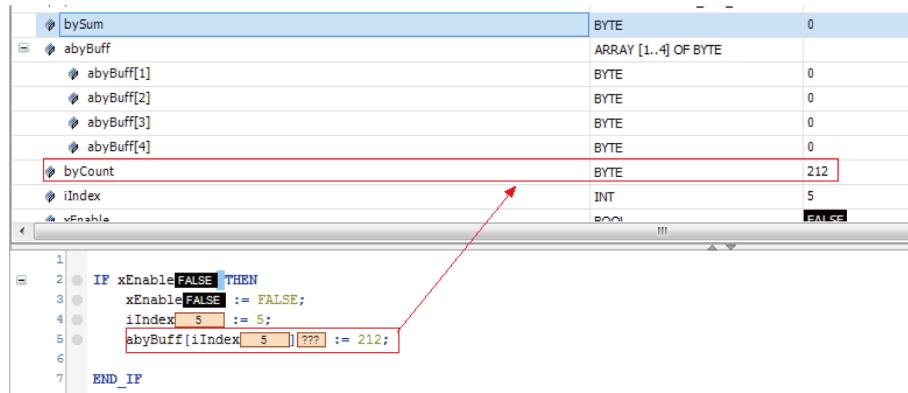
数组越界，分为上越界、下越界。程序中调用越界数组，可能导致数组相邻的变量值被调用的越界数组值覆盖。

程序中POU变量定义和数组在系统的内存分布大致如下图中bySum、abyBuff、byCount之间的关系。



变量定义区，byBuff[1]和变量bySum相邻，byBuff[4]与byCount相邻。如果用户程序对byBuff[0]、byBuff[5]进行写操作，那么bySum、byCount值可能分别被数组的byBuff[0]、byBuff[5]值覆盖，共用相同内存地址（编译时，系统已经所有变量分配内存地址。由于内存地址分配有优化算法，因此变量之间地址可能连续，可能不连续。一般情况下，数组上界与变量内存地址连续的可能性较大）。

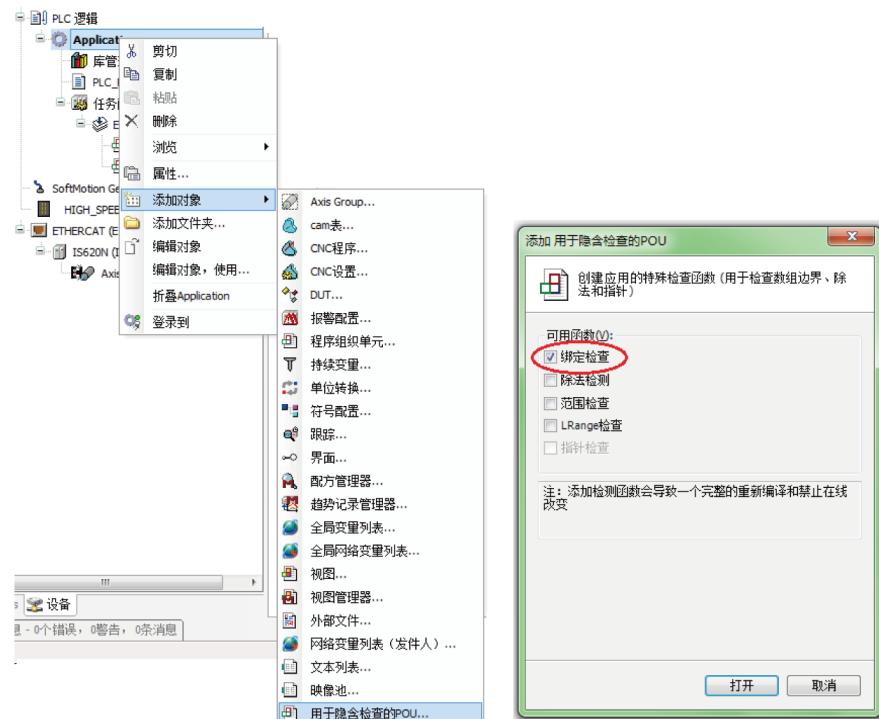
举例：iIndex作为数组变量，当iIndex等于5时，byBuff[iIndex]赋值“212”，则byCount等于212（如下图），因为byCount指向的地址与byBuff[5]地址连续导致。



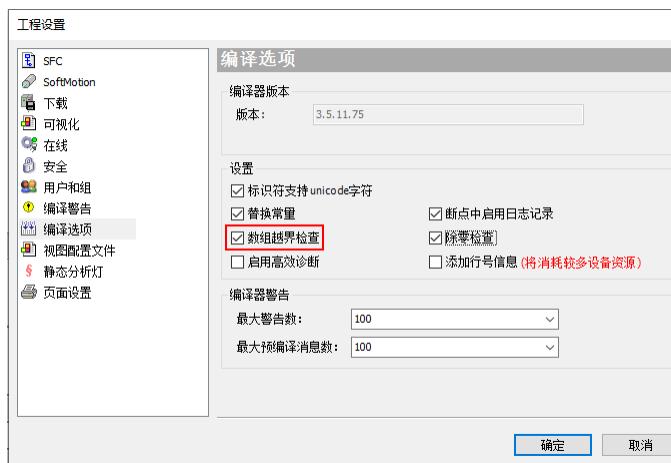
当iIndex等于0时，byBuff[iIndex]赋值“212”，则bySum等于0，因为bySum指向的地址与byBuff[0]地址不连续导致。

- 定位步骤

与指针异常步骤类似，在设备树中右键单击“Application”，选择“添加对象 > 用于隐含检查的POU”，并勾选“绑定检查（纯音译结果，正确翻译应该是检查界限/边界）”，系统将在“Application”设备树下默认添加“CheckBounds”函数，如下图所示。



或在菜单栏选择“工程 > 工程设置”，在“工程设置”界面单击“编译选项”，勾选“数组越界检查”。



## 说明

- 如需显示数组越界日志，则需要在“库管理器”中添加“CmpLog”库。
- 如对PLC的CPU负载要求较低，满足在极致性能下的使用需求，则可勾选“启用高效诊断”，同时需要在“库管理器”中添加“CmpApp”库。开启该功能后，最大输出1条该诊断信息，不开启则最大输出5条该诊断信息。

登录PLC，在代码处“CheckBounds := lower;和“CheckBounds := upper;”，增加断点并激活。通过单步调试（F10）定位到发生异常的用户程序位置，对比当前数组变量是否在数组定义的范围，如下图所示，abyBuff[5]不在定义byBuff[1..4]范围内。

```

1 // 隐含生成代码: 这里只是对代码实现的建议
2 IF index[5] < lower[1] THEN
3   CheckBounds[0] := lower[1];
4 ELSIF index[5] > upper[4] THEN
5   CheckBounds[0] := upper[4];
6 ELSE
7   CheckBounds[0] := index[5];
8 END_IF

```

## 单步调试 (F10) 直到进入用户程序

```

1
2 IF xEnable[FALSE] THEN
3   xEnable[FALSE] := FALSE;
4   iIndex[5] := 5;
5   abyBuff[iIndex[5] ???] := 212;
6
7 END_IF

```

- 解决方法

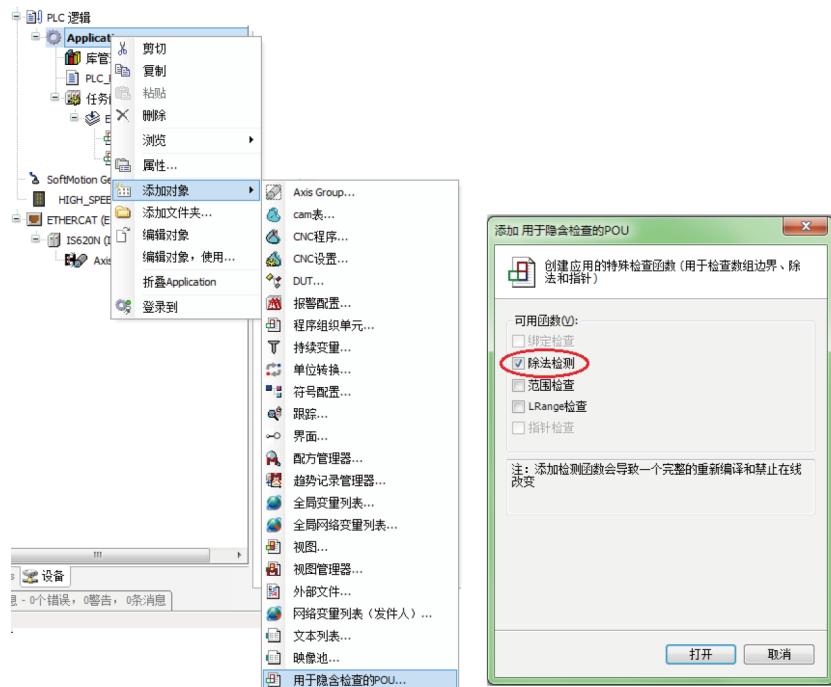
无，必须在代码中解决。

## 除数为0

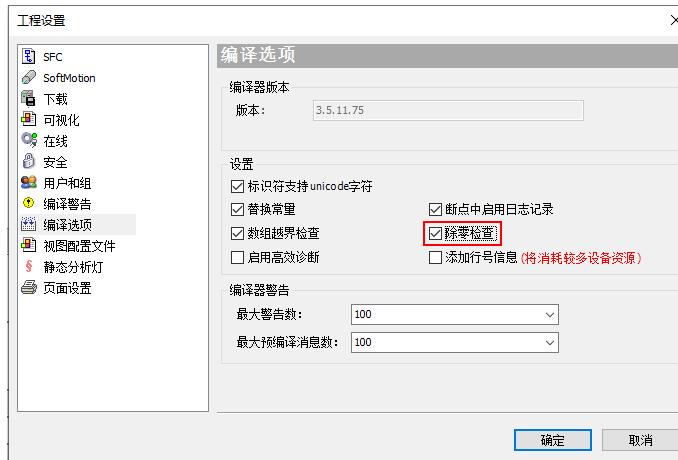
变量未初始化、用户程序变量初始化在调用之后、全局变量在多个任务或者多个POU被赋值。

- 定位步骤

在设备树中右键单击“Application”，选择“添加对象 > 用于隐含检查的POU”，并勾选“除法检查”，如下图所示。



或在菜单栏选择“工程 > 工程设置”，在“工程设置”界面单击“编译选项”，勾选“除零检查”。



## 说明

- 如需显示除零检查日志，则需要在“库管理器”中添加“CmpLog”库。
- 如对PLC的CPU负载要求较低，满足在极致性能下的使用需求，则可勾选“启用高效诊断”，同时需要在“库管理器”中添加“CmpApp”库。开启该功能后，最大输出1条该诊断信息，不开启则最大输出5条该诊断信息。

系统在“Application”设备树下默认添加“CheckDivDInt”、“CheckDivLInt”、“CheckDivLReal”、“CheckDivReal”四个函数。

登录PLC，在四个函数代码“CheckDivDInt:=1;”、“CheckDivLInt:=1;”、“CheckDivLReal:=1;”、“CheckDivReal:=1;”位置增加断点并激活。通过单步调试（F10）定位到用户程序位置（具体操作参考前面指针异常调试步骤）。

- 解决方法

参与运算的代码处增加“除数等于0”的条件判断，32bit变量通常用 $10E-6(0.000001)$ 作为判断门槛值，64bit变量根据需要调整精度，最小 $\geq 10E-15$ ，如下图。

```

10  IF ABS(fDiv) >= 0.000001 THEN
11      fRet := (fSum / fDiv);
12  END_IF
13

```

## 数据类型隐式转换

具有相同数据宽度的有符号和无符号变量之间赋值运算，强制赋值可能导致变量的值不在预期范围，变量在其他程序段被引用，可能会导致程序执行异常，如下图所示。

iTemp	INT	有符号整型	-32749
uiTemp	UINT	无符号整型	32787
1			
2	iTemp -32749 := uiTemp 32787;		
3			
4			
5	RETURN		

- 定位步骤

无

- 解决方法

用户程序在编译过程中会生成警告信息，重视编译警告信息的具体内容，保证参与赋值运算左右两边的数据类型相同。

## 死循环

程序中for、while、repeat循环条件使用不当，系统会一直在执行循环体中程序代码段，无法执行循环体以外代码，比如：PLC的LED数码管显示刷新、PLC与编程软件之间的通信，而EtherCAT任务下的死循环程序会导致掉电保持数据功能无法正常工作，丢失掉电保持数据。

- 定位步骤

for、while、repeat循环中引入计数变量，当计数变量到达一定值，循环跳出。登录PLC，在循环跳出位置“EXIT”增加断点，一旦系统循环计数超出预期，程序就会运行到断点处，断点使用方法参考帮助手册，

如下图所示，udiCnt循环执行次数检测变量，当循环体while下的代码执行100001次之后，程序运行到“EXIT”处。

```

13
14 udiCnt 1000001 := 0;          进入循环前清零变量udiCnt
15 WHILE (TRUE) DO
16   udiCnt 1000001 := udiCnt 1000001 + 1;
17   IF udiCnt 1000001 > 1000000 THEN
18     EXIT;                      新增断点位置
19   END_IF
20
21   //用户代码
22 END_WHILE
23 RETURN

```

- 解决方法

按照定位步骤解决，或者在循环体内增加定时器，当定时时间到达，程序跳出循环体。

## 功能块实例在多个任务调用

功能块实例的内部变量在扫描周期内执行一次后，第二次执行时的变量仍保持上一次的值（类似C/C++语言的静态变量）。两个任务A、B同时调用相同功能块实例（包括方法、动作、属性、转移）时，会出现A任务中功能块实例未执行完毕，被优先级高的B任务抢占执行，等待B任务执行完毕，A任务再执行时候，功能块实例的内部变量的数值与被抢占前的数值可能不完全一致，最后影响功能块在A、B任务的正常执行。

- 定位步骤

无。

- 解决方法

无，必须在前期代码设计时规避。

## 全局变量在多个任务调用

两个任务A、B中都对同一变量进行写访问操作，会出现A任务正在写全局变量，被优先级高的B任务抢占CPU资源，并写全局变量的值，等到B任务执行完毕，A任务再执行时候，全局变量的值可能与被抢占之前的值不一致，最后影响功能块正常执行。

例如：wTemp是一个具备2个字节的全局变量，wTempH、wTempL分别为wTemp高、低位字节。当A任务执行写完wTempL还未写wTempH时，被高优先的B任务抢占CPU资源，B任务写完wTempL和wTempH后，释放CPU资源后，A任务接着执行写wTempH。由于wTempL被任务改写，wTemp在A任务的值不在预期范围内。

- 定位步骤

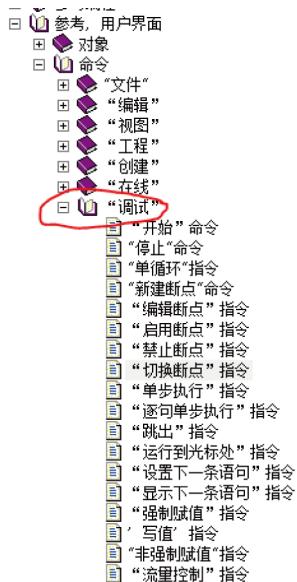
无

- 解决方法

全局变量只能在一个任务执行写访问，必须在代码设计时候解决。（汇川技术中型PLC可以通过互斥信号量解决此类问题，但这仅仅适合对操作系统有深入理解的高级开发人员，普通用户不要轻易尝试，使用不当会造成PLC死机）。

## 其他

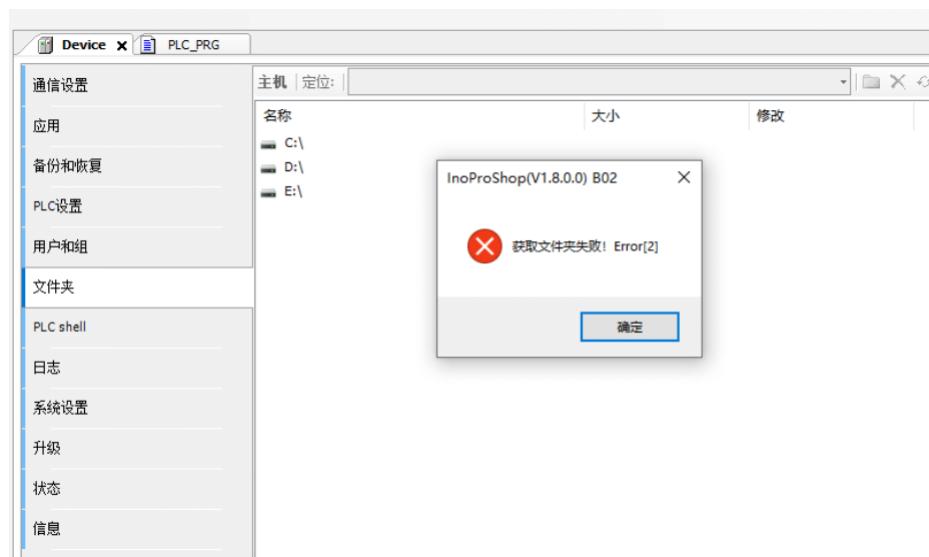
用户程序断点、单步等调试方法请参考帮助手册，相关内容在帮助手册路径如下图。



## 8.3 获取文件夹失败

### 现象描述

在InoProShop软件“Device”界面中查看文件夹，弹出“获取文件夹失败！”提示框。

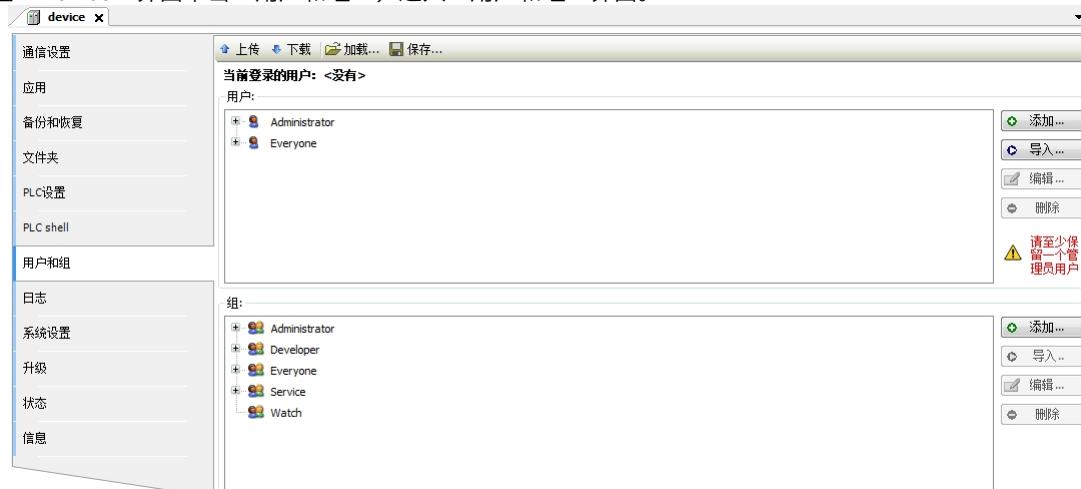


## 原因分析

用户使用默认的Everyone用户登录PLC，对“用户和组”进行配置前，Everyone用户默认拥有管理员权限访问文件夹；对“用户和组”进行配置后，Everyone用户不再拥有管理员权限访问文件夹。

## 解决方法

- 在“Device”界面单击“用户和组”，进入“用户和组”界面。



- 在“组”区域中选择“Administrator”，单击“编辑”，打开“编辑组 Administrator”对话框。



- 勾选“用户 Everyone”，单击“确定”，将“Everyone”用户添加至“Administrator”管理员组。
- 单击“下载”，使配置生效。

## 8.4 CPU实时负载超限



注意

AC700/AC800系列PLC固本版本1.26.15.6及以上和AM400/AM600系列PLC固本版本1.40.9.3及以上支持此功能。

### 现象描述

实时IEC任务异常高占用（当前默认配置是99%）某个CPU超过一定的时间（AM400/600系列PLC默认为10s，AC700/800系列PLC默认为5s），将触发ProcessorLoad异常，APP停止运行，设备显示屏显示“ER.2E”、“CPU ProcessorLoad exception”，并记录异常诊断日志。



## 说明

AC700/AC800系列PLC在实时IEC任务死循环场景下会触发ER27和ER2E两个错误。AC700系列PLC显示屏交替闪烁“2E”和“27”两个错误码，AC800系列PLC按【确认】键显示屏显示“Er.2E”和“Er.27”两个错误码。

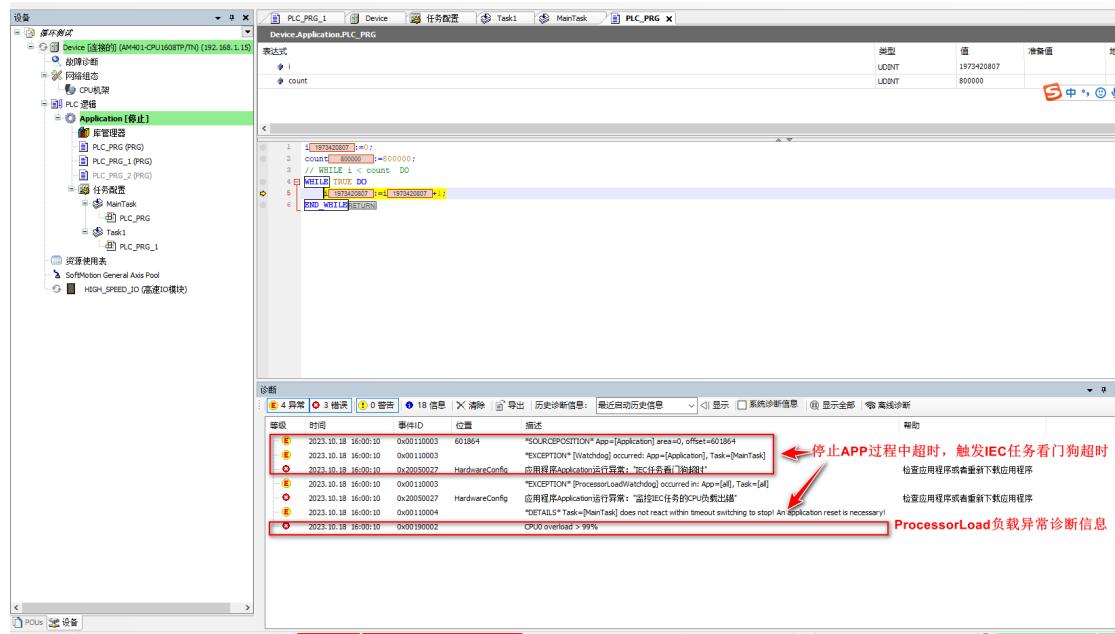
## 原因分析

用户程序存在不合理的循环逻辑，特别是在实时IEC任务中易造成实时任务对CPU占用超100%的情况，系统中其他一些关键低优先级实时或者非实时任务线程（如设备扫描，网络收发线程等）无法正常响应和执行，PLC可能出现无法ping通，InoProShop无法正常扫描和连接的“不可控”状态。

## 解决方法

可通过InoProShop诊断异常日志和InoProTool工具收集的日志“StdLogger.csv”进行分析。

- 在“诊断”界面查看CPU负载超限的诊断信息，单击“显示全部”，查看详细日志，以下为异常日志分析方法示例。



- 通过InoProTool工具收集的日志“StdLogger.csv”进行离线分析。

时间	事件ID	位置	描述
2023.10.18 16:00:10	0x00110003	601864	"SOURCEPOSITION" App=<Application> area=0, offset=601864
2023.10.18 16:00:10	0x00110003		"EXCEPTION" [Watchdog] occurred: App=<Application>, Task=<MainTask>
2023.10.18 16:00:10	0x20050027	HardwareConfig	应用软件Application运行异常，已“任务看门狗超时”
2023.10.18 16:00:10	0x00110003		"EXCEPTION" [ProcessorLoad/watchdog] occurred in: App=<[a]>, Task=<[a]>
2023.10.18 16:00:10	0x20050027	HardwareConfig	应用软件Application运行异常：“监控IEC任务的CPU负载超限”
2023.10.18 16:00:10	0x00110004		"DETAILS" Task=<MainTask> does not react within timeout switching to stop! An application reset is necessary!
2023.10.18 16:00:10	0x00190002	CPU0 overload > 99%	
2023.10.18 16:00:10	0x00190002		ProcessorLoad 负载异常诊断信息

# 9 附录

## 9.1 各通信端口的通信协议简介

### 9.1.1 概述

中型PLC系列提供Mini-USB口、串行通信口、Ethernet网口、EtherCAT网口、Mini-SD卡插槽、CAN通信口、Profibus-DP通信口、高速I/O接口和本地总线扩展接口等。

### 9.1.2 Mini-USB 端口及其内置通信协议

Mini-USB口的主要用途是下载PLC用户程序、进行监控调试，因此该端口的通信协议是固定的，用户无需选择通信协议，只要PC安装了相应的USB驱动程序，PC在InoProshop中就可以随时与中型PLC进行用户程序下载或监控。

Mini-USB端口内置的下载协议是汇川公司的专有协议，不支持第三方编程软件对中型PLC的程序下载。

首次安装InoProshop编程软件后，USB的驱动程序会自动安装。同一PC安装不同的版本的InoProshop，必须使用不同的安装目录。

### 9.1.3 COM通信端口及其内置协议

COM0、COM1口为PLC对外通信的基本端口，两个通信端口集成在同一个DB9物理端口上，其主要用途是RS485通信或者Modbus通信。

COM0、COM1通信端口支持的协议与设置单元定义如下表：

COM0/COM1协议	半双工/全双工模式	通信格式	波特率	数据位	停止位	校验
Modbus-RTU主站	半双工	固定	4800bits/s 9600bits/s	7bit 8bit	1bit 2bit	NONE ODD EVEN
Modbus-RTU从站	半双工		19200bits/s 38400bits/s 57600bits/s 115200bits/s	8bit		
RS485自由协议	半双工	不固定		8bit		
Modbus-ASCII主站	不支持	-	-	-	-	-
Modbus-ASCII从站	不支持	-	-	-	-	-

这几种协议简介如下：

- Modbus主站协议

PLC作为控制主机，常用该协议与变频器、伺服等下位机进行通信，或读取智能仪表、传感器的信息等。  
PLC之间采用Modbus通信，也会带来通信的灵活性。

- Modbus从站协议

当有上位机需要读取PLC的内部数据时，常采用Modbus协议，PLC作为通信从站。当PLC将端口设置为Modbus从站后，PLC根据上位机的通信命令，自动进行响应处理并给与应答。

- **自由通信协议**

PLC内置通信协议以外的协议，都称为“自由通信协议”，要以自由协议进行通信，编程人员必须完整理解该协议的帧结构定义。编程人员根据从站通信协议，以及要求的通信操作，在用户程序中事先准备好寄存器区中待发送的数据串，系统将指定寄存器区域的数据，自动向串口依次发送；然后串口进入接收状态，将串口接收到的数据存放于指定区域，当接收到指定长度的数据后，通过系统标志，通知用户程序，以便用户程序对接收的数据，按照协议规定，去解析得到所要要求的数据。

AM600系列自由通信协议中操作寄存器，相当于用户程序直接访问通信缓冲区，借助用户程序对通信收发缓冲区的处理，实现自定义协议的通信。实际编程时，需要作一些串行通信的配置和准备，如设定串口的收发模式、波特率、位数、校验位、软件协议的设定、超时判断条件、收发缓冲区的数据准备、收发标志处理等，才能按预期的要求进行通信。

#### 9.1.4 CANopen 通信协议

CANopen通信采用功能块读写SDO/PDO方式实现通信。将需要的通信变量（协议中的对象字典数据）赋值给功能块对应的输入参数，触发执行条件即可访问从站设备数据，AM600仅支持CANopen主站。

#### 9.1.5 CANlink通信协议

CANlink通信可采用配置表格法，将需要的通信变量、希望的通信频度、触发条件等，采用填表方式，事先进行设定。作为网络CANlink主站时，可连接汇川各种远程扩展模块、MD380/500系列变频器、IS620系列伺服等从站设备；同时也可作为CANlink网络从站接入其他设备。

CANlink3.0通信协议中，提供了如下通信帧：

- 定时触发和条件触发的通信帧，用于普通从站的通信数据交互；
- 同步触发，用于多个具有同步控制的高实时性设备控制，例如多个伺服的位置同步控制；
- 心跳帧，用于监视CANlink网络各从站的通信状态，便于对控制系统的异常状态作及时响应，避免造成更大损失。

#### 9.1.6 Ethernet 端口及通信协议

Ethernet端口主要有以下两个用途。

- 与Mini-USB口一样，用于下载PLC用户程序、进行监控调试。
- 以太网通信，包括标准的TCP/IP Modbus通信和自由通信。

Modbus协议通过后台直接配置相关通讯功能码和相应的地址映寄存器，用户程序中访问寄存器值即可与远端的Modbus设备进行数据交互。自由通信协议只能通过操作标准的socket功能块实现数据交互。

#### 9.1.7 EtherCAT 端口及通信协议

EtherCAT口主要用途用户标准的EtherCAT协议通讯，线性拓扑结构，全双工通讯，波特率1Mbit/s，从站节点之间通讯距离最大支持100m。主站支持同步事件、DC模式，AM600机型任务最大抖动在120us（典型值）。

### 9.1.8 高速IO 接口

高速I/O接口支持高速脉冲控制、高速脉冲计数功能。

- 高速脉冲控制用于控制脉冲式伺服驱动器、步进驱动器控制等设备；
- 高速脉冲计数用于AB相、单相、CW/CCW等脉冲信号频率和计数采集。

### 9.1.9 Mini-SD 卡插槽

Mini-SD插槽主要用于PLC底层固件升级（不对外开放）和PLC用户程序升级（对外开放）。

### 9.1.10 本地总线扩展接口

本地总线扩展接口实现PLC直接连接IO模块功能。PLC通过内部总线周期刷新IO模块的数据在PLC的映射地址。

用户仅需访问映射地址就可以对IO模块进行操作。

### 9.1.11 Profibus-DP端口

Profibus-DP端口和CAN端口集成在同一个DB9硬件端口上，目前DP功能仅在AM610中使用，在其他产品保留未使用。

## 9.2 软元件概述

软元件是编程系统预定义全局变量，编程时可以直接使用，不需要定义。软元件是直接变量，映射到M区（%M），并且具有RETAIN特性（掉电保存特性）。AM600编程系统包含两种软元件：SD软元件和SM软元件。SD软元件是INT类型全局直接变量，SM为BOOL类型全局直接变量。

M区（%M）共512kB，前480kB为用户使用区域，后32KB时系统使用区域，用户不要直接使用地址。最后32kB中的前30000Bytes为SD和SM软元件使用区域，用于实现CANlink、CANopen、高速I/O指令、ModbusS等特殊功能，具体如下表，用户可以访问这些软元件。

SD区间分配	功能	SM区间分配	功能
0~7999	用户使用寄存器元件：0~7000为CANlink使用（CANlink配置，兼容小型PLC）	0~7999	用户使用的位元件： 0 - 3071，8000-8511为CANlink使用（CANlink配置，兼容小型PLC） 0-7999为Modbus/Modbus TCP触发变量，从站使能变量使用。
8000~8999	系统使用的寄存器元件：CANlink/CANopen	8000~8999	系统使用的位元件：CANlink/CANopen
9000~9999	系统使用的寄存器元件：目前只有高速I/O使用	9000~9999	系统使用的位元件：目前只有高速I/O使用

### 说明

- Modbus触发变量在置位后，系统会自动复位，编程时请注意。
- 系统软元件，用户只可以读取不能写入，否则系统可能会出现异常。

关于软元件具体使用参见CANlink软元件和Modbus及Modbus TCP软元件说明。

### 9.3 基本指令速查表

指令类型	指令说明	指令名	指令分类
算术运算指令	加法指令	ADD	函数
	乘法指令	MUL	函数
	减法指令	SUB	函数
	除法指令	DIV	函数
	取余指令	MOD	函数
数据处理指令	赋值指令	MOV	函数
	数据批量传送	BMOV	函数
	数据一对多传输	FMOV	函数
	获取数据指定位的状态	BON	函数
	ON位总数	SUM	函数
	字节单位的数据结合	BTOW	函数
	字节单位的数据分类	WTOB	函数
	高低字节交换	SWAP	函数
	数据交换	XCH	函数
字逻辑指令	与指令	AND	函数
	或指令	OR	函数
	异或指令	XOR	函数
	取非指令	NOT	函数
(CmpHCUtils)	上升沿输出	PLS	功能块
	下降沿输出	PLF	功能块
	交替输出	ALT	功能块
	位数据输出	BOUT	函数
	位数据置位	BSET	函数
	位数据复位	BRST	函数
移位指令	左移指令	SHL	函数
	右移指令	SHR	函数
	循环左移指令	ROL	函数
	循环右移指令	ROR	函数
	带进位的循环右移	RCR	函数
	带进位的循环左移	RCL	函数
	位数据向左拷贝	SFTL	函数
	位数据向右拷贝	SFTR	函数
	字数据向左拷贝	WSFL	函数
	字数据向右拷贝	WSFR	函数
	先进先出的数据读出	SFRD	函数
	先进先出的数据写入	SFWR	函数
选择指令	二选一指令	SEL	函数
	取最大值指令	MAX	函数
	取最小值指令	MIN	函数
	极限值指令	LIMIT	函数
	多选一指令	MUX	函数

指令类型	指令说明	指令名	指令分类
比较指令	大于指令	GT	函数
	小于指令	LT	函数
	大于等于指令	GE	函数
	小于等于指令	LE	函数
	等于指令	EQ	函数
	不等于指令	NE	函数
数学基本运算指令	绝对值指令	ABS	函数
	平方根指令	SQRT	函数
	自然对数指令	LN	函数
	常用对数指令	LOG	函数
	指数指令	EXP	函数
	正弦指令	SIN	函数
	余弦指令	COS	函数
	正切指令	TAN	函数
	反正弦指令	ASIN	函数
	反余弦指令	ACOS	函数
	反正切指令	ATAN	函数
	幂指令	EXPT	函数
	角度值转换成弧度值	RAD	函数
数学辅助运算指令 (Util库)	微分指令	DERIVATIVE	功能块
	积分指令	INTEGRAL	功能块
	整形统计指令	STATISTICS_INT	功能块
	实型统计指令	STATISTICS_REAL	功能块
	平方偏差	VARIANCE	功能块
类型转换指令	布尔类型转换指令	BOOL_TO_<TYPE>	函数
	字节类型转换指令	BYTE_TO_<TYPE>	函数
	日期类型转换指令	DATE_TO_<TYPE>	函数
	长整型转换指令	DINT_TO_<TYPE>	函数
	日期时间类型转换指令	DT_TO_<TYPE>	函数
	双字类型转换指令	DWORD_TO_<TYPE>	函数
	整数类型转换指令	INT_TO_<TYPE>	函数
	字类型转换指令	WORD_TO_<TYPE>	函数
	实数类型转换指令	REAL_TO_<TYPE>	函数
	短整型转换指令	SINT_TO_<TYPE>	函数
	字符类型转换指令	STRING_TO_<TYPE>	函数
	时钟类型转换指令	TIME_TO_<TYPE>	函数
	时间类型转换指令	TOD_TO_<TYPE>	函数
	无符号长整型转换指令	UDINT_TO_<TYPE>	函数
地址运算指令	取地址指令	ADR	函数
	取地址内容指令	^	函数
	位地址指令	BITADR	函数
	索引指令	INDEXOF	函数
	数据类型大小指令	SIZEOF	函数
调用指令	调用指令	CAL	函数
初始化操作指令	初始化操作指令	INI	函数

指令类型	指令说明	指令名	指令分类
字符串处理指令 (Standard库)	取字符串长度指令	LEN	函数
	左边取字符串指令	LEFT	函数
	右边取字符串指令	RIGHT	函数
	中间取字符串指令	MID	函数
	合并字符串指令	CONCAT	函数
	插入字符串指令	INSERT	函数
	删除字符串指令	DELETE	函数
	替换字符串指令	REPLACE	函数
	查找字符串指令	FIND	函数
双稳态指令 (Standard库)	置位优先双稳态器	SR	功能块
	复位优先双稳态器	RS	功能块
触发器指令 (Standard库)	上升沿检测触发器	R_TRIG	功能块
	下降沿检测触发器	F_TRIG	功能块
计数器 (Standard库)	递增计数器	CTU	功能块
	递减计数器	CTD	功能块
	递增递减计数器	CTUD	功能块
定时器 (Standard库)	普通定时器	TP	功能块
	通电延时定时器	TON	功能块
	断电延时定时器	TOF	功能块
	实时时钟	RTC	功能块
BCD转换指令 (Util库)	BCD码转整形指令	BCD_TO_INT	函数
	整形转BCD码指令	INT_TO_BCD	函数
位/字节操作指令 (Util库)	位提取指令	EXTRACT	函数
	位整合指令	PACK	函数
	位拆分指令	UNPACK	功能块
	位赋值指令	PUTBIT	函数
控制器指令 (Util库)	比例微分控制器指令	PD	功能块
	比例积分微分控制器指令	PID	功能块
	比例积分微分控制器指令	PID_FIXCYCLE	功能块
信号发生器指令 (Util库)	脉冲信号发生器	BLINK	功能块
	周期性信号发生器	GEN	功能块
机器人操作指令 (Util库)	特征曲线指令	CHARCURVE	功能块
	整形限速指令	RAMP_INT	功能块
	实型限速指令	RAMP_REAL	功能块
模拟量处理指令 (Util库)	滞后指令	HYSTeresis	功能块
	上下限报警指令	LIMITALARM	功能块

指令类型	指令说明	指令名	指令分类
PLC系统信息指令  (SysHCPlInfo库, 具体指令使用详见帮助)	获取系统硬件相关信息	SysHC_HWInfo	功能块
	获取系统软件相关信息	SysHC_SWInfo	功能块
	获取CPU相关信息	SysHC_CPUInfo	功能块
	获取CPU相关故障诊断信息	SysHC_CPUDiagnose	功能块
	获取ModbusRTU从站设备故障诊断	SysHC_ModbusRtuDeviceDiagnose	功能块
	获取ModbusRTU主站访问从站故障诊断	SysHC_ModbusRtuSlaveDiagnose	功能块
	获取ModbusTCP从站设备故障诊断	SysHC_ModbusTcpDeviceDiagnose	功能块
	获取ModbusTCP主站访问从站故障诊断	SysHC_ModbusTcpSlaveDiagnose	功能块
	设置网络信息	SysHC_NetworkConfig	功能块
	获取网络信息	SysHC_NetworkInfo	功能块
	获取U盘路径信息	SysHC_UDiskPath	功能块
	获取Boot版本号	GetBootVersion	函数
	获取PLC版本号	GetPLCVersion	函数
	获取设备名	GetProductName	函数
	获取Runtime版本号	GetRuntimeVersion	函数
	获取SN号	GetSerialNumber	函数
	保存掉电保存信息	SysHC_SaveAllRetains	函数
时间和日期  (CmpHCUtis)	设置当前系统时钟	SetSystemDate	功能块
	获取当前系统时钟、时区	GetSystemDate	功能块
	获取系统运行时间, 计时单位分别为毫秒、微妙、纳秒	GetSystemTime	功能块
表格和区间  (CmpHCUtis)	死区控制指令	BZAND_TAB	函数
	数据平均值计算	MEAN_TAB	函数
	区域控制指令	ZONE_TAB	函数
	全部数据复位	ZRST_TAB	函数
	表格坐标获取	SCL_TAB	函数
	表格数据排序	SORT_TAB	函数
	斜坡指令	RAMP_TAB	功能块
通讯指令  (CmpHCUtis)	数据总和计算	WSUM_TAB	函数
	创建TCP服务器端通信服务	TCP_Server	功能块
	创建TCP客户端通信服务	TCP_Client	功能块
	创建TCP连接, 并连接到服务器	TCP_Connect	功能块
	TCP通信数据接收	TCP_Recieve	功能块
	TCP通信数据发送	TCP_Send	功能块
	创建UDP通信连接	UDP_Peer	功能块
	UDP通信数据接收	UDP_Receive	功能块
	UDP通信数据发送	UDP_Send	功能块

指令类型	指令说明	指令名	指令分类
滤波指令 (CmpHCUtis)	限幅滤波	LimitingFilter	功能块
	中位值滤波	MedianFilter	功能块
	算术平均滤波	ArithmeticAverageFilter	功能块
	递推平均滤波	RecursiveAverageFilter	功能块
	中位值平均滤波	MedianAverageFilter	功能块
	限幅平均滤波	LimitingAverageFilter	功能块
	一阶滞后滤波	FirstOrderLagFilter	功能块
	加权递推平均滤波	WeightRecursiveAverageFilter	功能块
	消抖滤波	DebounceFilter	功能块
	限幅消抖滤波	LimitingDebounceFilter	功能块
获取系统运行时间，计时单位分别为毫秒、微妙、纳秒		GetSystemTime	功能块
队列 (CmpHCUtis)	先入先出队列	FIFO	功能块

## 9.4 PLC编程软件升级

### 9.4.1 版本说明

- 编程软件InoProshop V1.1.0及之前版本在持久性变量、硬盘分区、高速I/O功能、EtherCAT总线IO模块等方面无法与新版本兼容，建议用户升级到最新版本。此外，本文描述中未涉及的版本（V1.3.2以下）不推荐使用，如有特殊要求请联系本地供应商。
- 从站设备文件，如EtherCAT描述文件（.xml）、CANopen描述文件（.eds）、Profibus-DP描述文件（.gds），应与从站设备固件版本相匹配，如有疑问，请联系本地供应商。V1.3.2版本中默认未安装的从站设备，可通过安装相应的设备文件获得支持。
- 如需了解AM400/AM600/AC800系列产品具体使用方法，请参见相关硬件手册或与供应商联系。

### 9.4.2 升级方法

- 应用软件安装

电脑配置要求Windows7或Windows10，内存不小于4G，强烈推荐使用64位中英文操作系统（32位系统不建议使用）。

请按照安装导向进行安装，或者根据需求在安装过程中对安装路径进行设置，默认安装路径为：C:\Inovance Control\InoProShop。

#### 说明

不可与其他版本安装在同一个文件夹中。

- 用户工程

如果打开旧版本工程，会弹出“工程版本信息”窗口。若不希望更新工程，选择“不更新”即可直接编辑使用，但是梯形图必须更新。

显示“工程版本信息”窗口有两种方式：

- 旧工程打开时自动弹出。
- 使用菜单命令【工程】 - 【工程版本信息…】打开。

有两种更新工程形式：全部更新和部分更新。

1. 全部更新，在弹出“工程版本信息”窗口时，选择“全部设置成最新版本”按钮，然后点击“确定”按钮。
2. 部分更新，在弹出“工程版本信息”窗口时，选择需要更新的类型对应的选项卡，然后点击“确定”按钮。

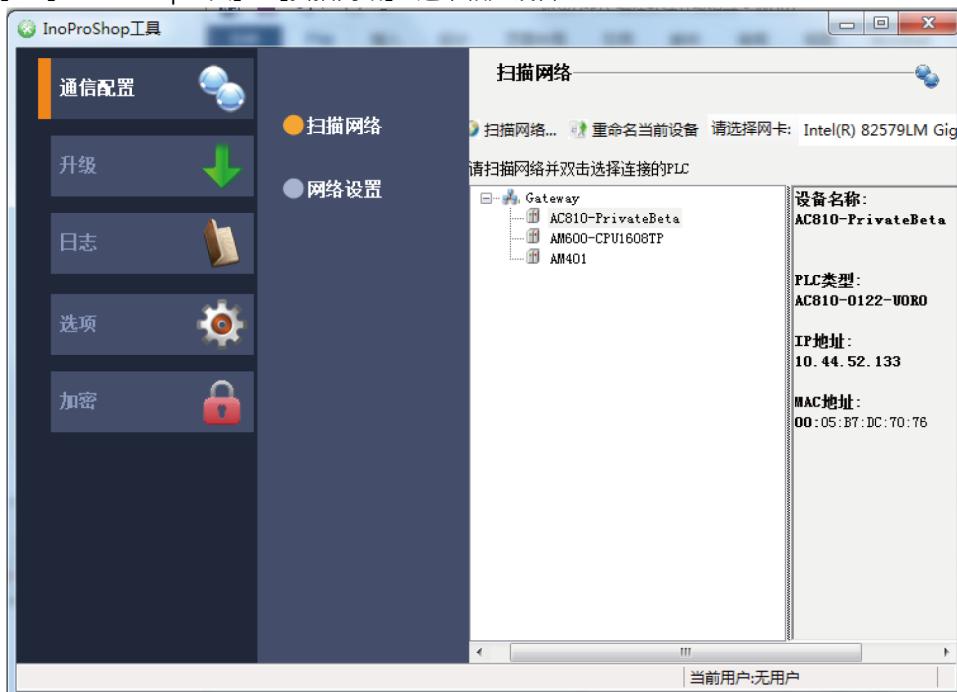
## 说明

梯形图必须更新。

## 在线固件升级

PLC设备（CPU模块）升级

1. 【工具】 - 【InoProshop工具】 - 【扫描网络】 - 选中相应设备



2. 【升级】 - 【选择固件升级包】 - 【升级】



## EtherCAT模块升级

1. 点击【设备】 - 【概述】，勾选“启动专家设置”，然后点击【登录下载】 - 【运行】。



2. 点击【设备】 - 【在线】，在【状态机】中选择“引导”。进入引导状态后，选择【通过EtherCAT进行文件访问】 - 【下载】。在弹出的框中，找到要升级的后缀名为“.bin”固件文件路径，选中固件文件启动升级。



## 库升级

参考下文“常见问题”中的“如何在工程添加编译库”。

## EtherCAT设备文件升级

1. 点击【设备】 - 【概述】，勾选“启动专家设置”，选择【登录下载】 - 【运行】。



2. 点击【设备】 - 【在线】 - 【写EEPROMXML】，在弹出的框中，找到要升级的XML文件的路径，选中该文件后启动升级。



## 9.4.3 常见问题

### 如何查看版本

在设备树中双击“Device (XXX)”，单击“升级”页签，再单击“获取PLC信息”。



## 目标系统与连接设备不匹配

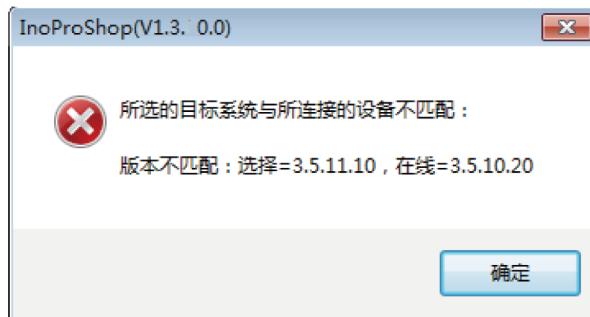


图9-1 目标系统与连接设备不匹配

**问题原因：**出现此错误的原因是InoProShop的PLC设备版本为“3.5.11.10”，而实际PLC设备版本为“3.5.10.20”，InoProShop上的设备版本不能大于实际版本！

- **解决方案1：**更新PLC固件，升级到与PLC设备版本匹配的固件版本“3.5.11.10”。

1. 右键单击“Device (XXX)”，选择“更新设备”，在弹出的窗口勾选“显示所有版本（仅限专家）”，找到更新设备的对应版本，单击“更新设备”。如果在设备列表中找不到与实际硬件相同的版本，选择前三个数字一样的版本即可。

如下图所示，列表中不存在与硬件设备相同的“3.5.10.20”，可以选择“3.5.10.40”（前面的三个数字版本相同）并更新设备。

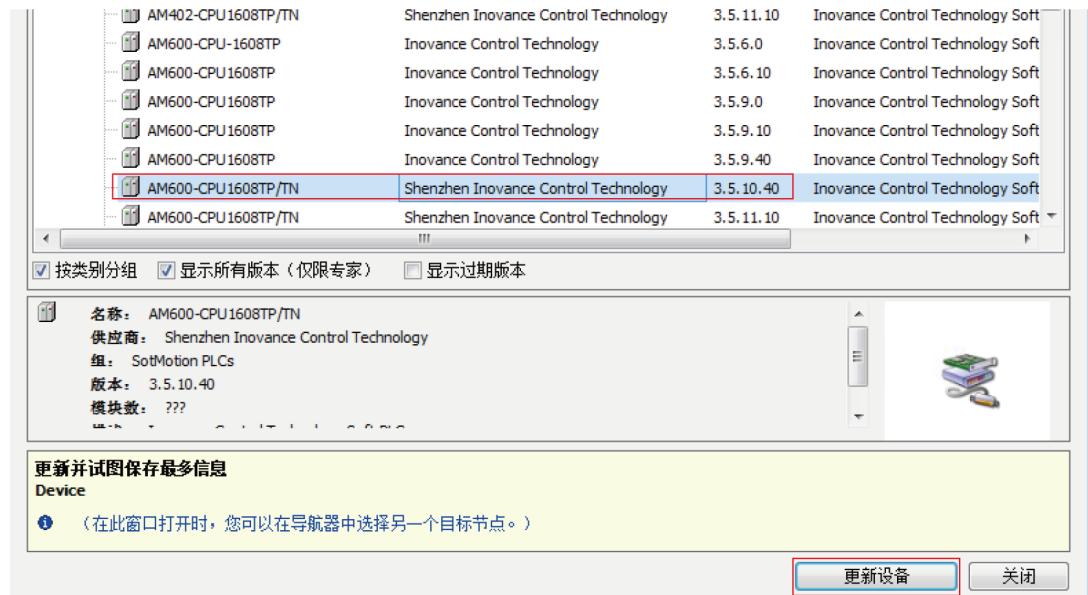


图9-2 更新PLC设备版本

2. 重新扫描并选中相应设备，系统将不会弹出错误窗口，如下图所示。

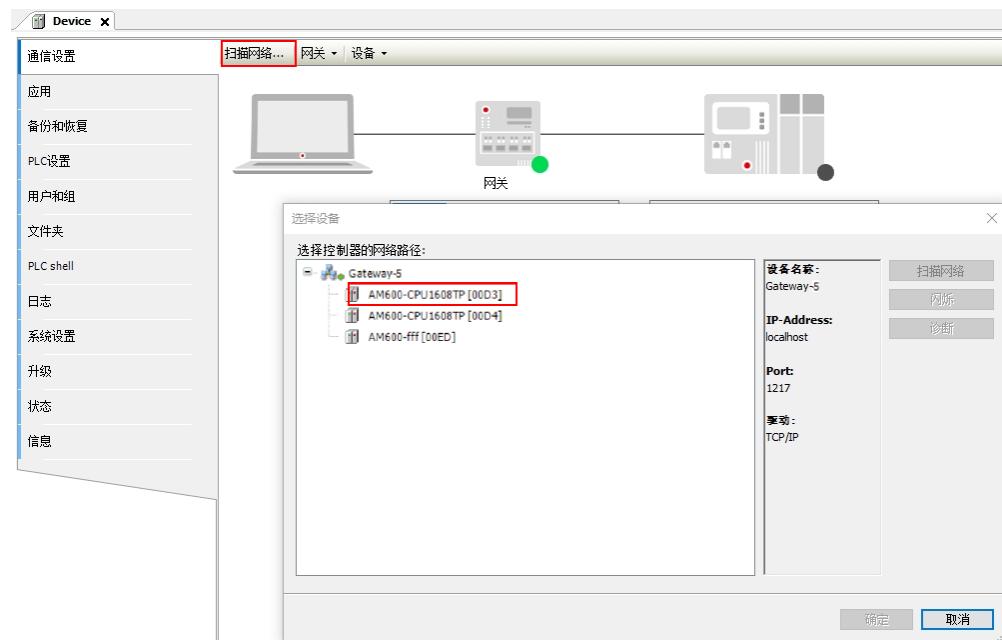


图9-3 连接PLC设备

3. 单击“升级”页签，在“固件升级”区域在线升级固件。

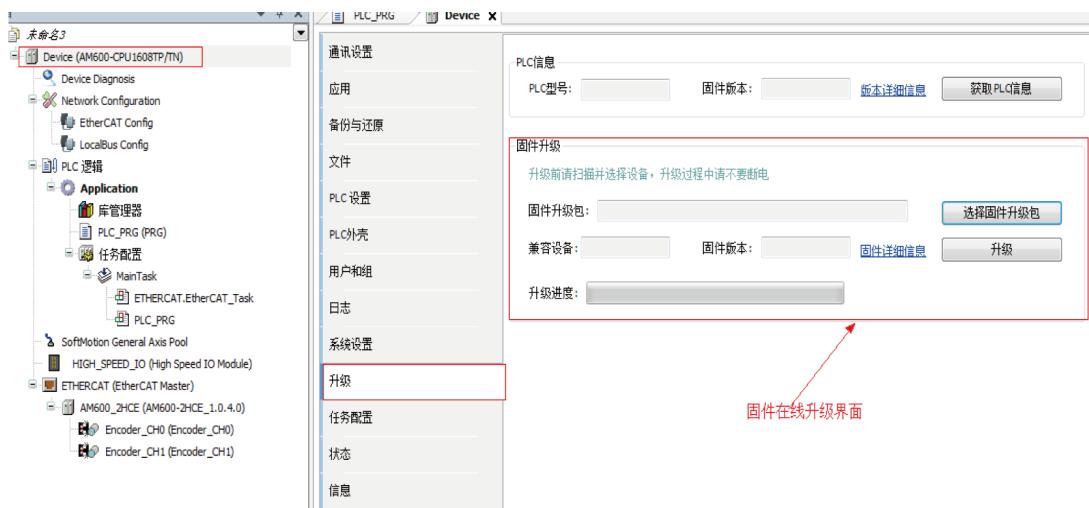


图9-4 在线升级固件

4. 固件升级完毕后，再按照步骤1将PLC设备版本更新到版本“3.5.11.10”，即可使用最新版本的PLC设备和PLC固件。

- 解决方案2：**更新设备版本，降低到与固件匹配的设备版本。

执行解决方案1中的第1步即可达到目的，但低版本的PLC设备文件只能跟该版本配套IEC库使用。

工程中添加IEC库时，由于IEC库添加的规则是默认最新版本，编译程序时可能出现编译报错，原因是库版本与PLC设备版本匹配不一致，可以通过手动更改IEC库版本解决。

## 新编程软件版本中添加库时编译报错

V1.3.2打开V1.3.0以前版本上的工程（以V1.2.0为例），添加IEC库“CmpBasic”，并使用库里面的“MC\_ResetDrive”功能块时，结果编译报错。

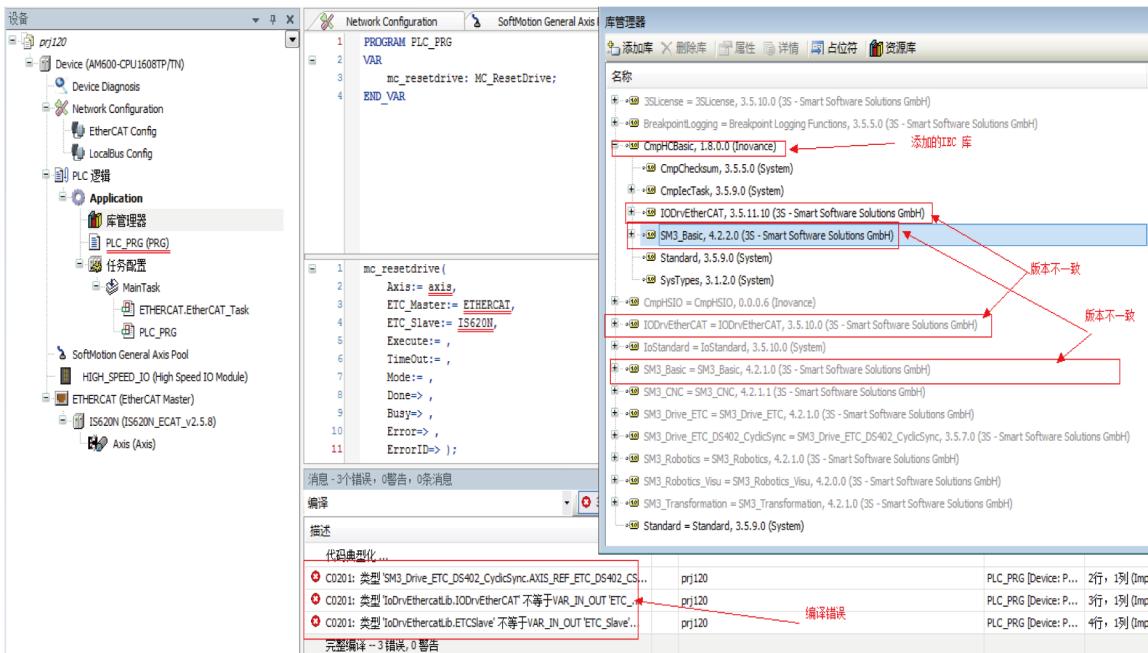


图9-5 新添加库编译报错

**问题原因：**CmpBasic库内依赖的特定库（SM3\_Basic、IODrvEtherCAT）与工程库管理中依赖的特定库版本不统一。CmpBasic（V1.8.0.0）版本依赖的“IODrvEtherCAT”版本为“V3.5.11.10”，而工程引用的版本是“V3.5.10.0”；CmpBasic（V1.8.0.0）版本依赖的“SM3\_Basic”版本为“V4.2.2.0”，而工程引用的版本是“V4.2.1.0”。

#### 解决方案：

- 在设备树中双击“库管理器”，打开“库管理器”界面，在库列表中选“CmpBasis”库，此时版本为“1.8.0.0”。
- 在“库管理器”界面中选择“属性”选项，在弹出的窗口内找到“版本”选项卡，在下拉菜单中选择与PLC设备对应的IEC库版本为1.6.0.0（V1.2.0新建的工程），然后单击“确定”。

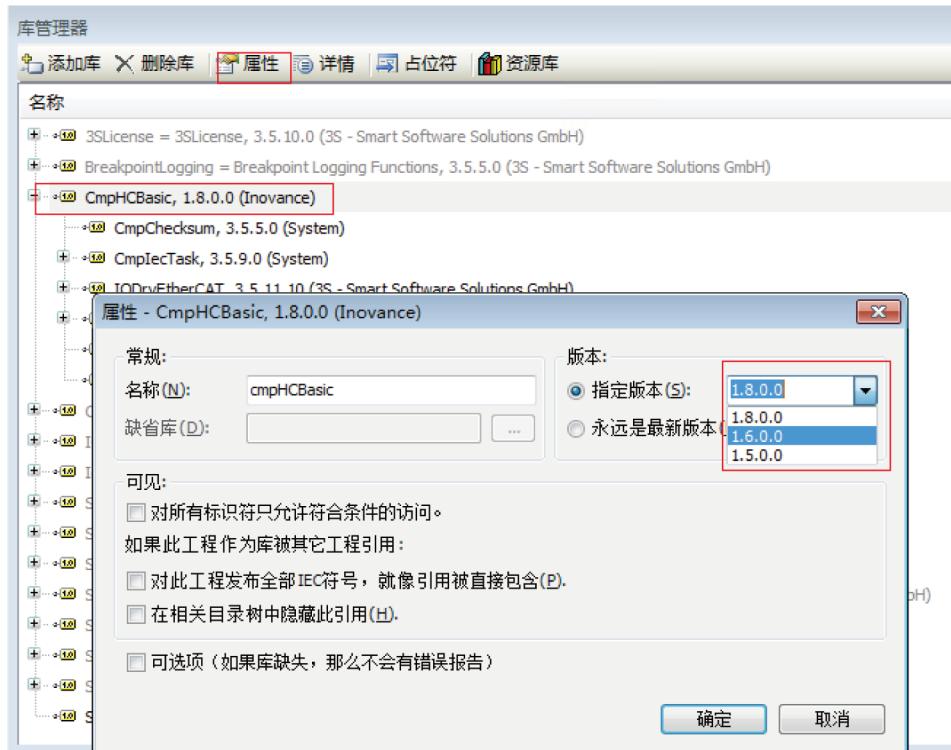


图9-6 手动更新IEC库

3. 编译工程，如下图所示。

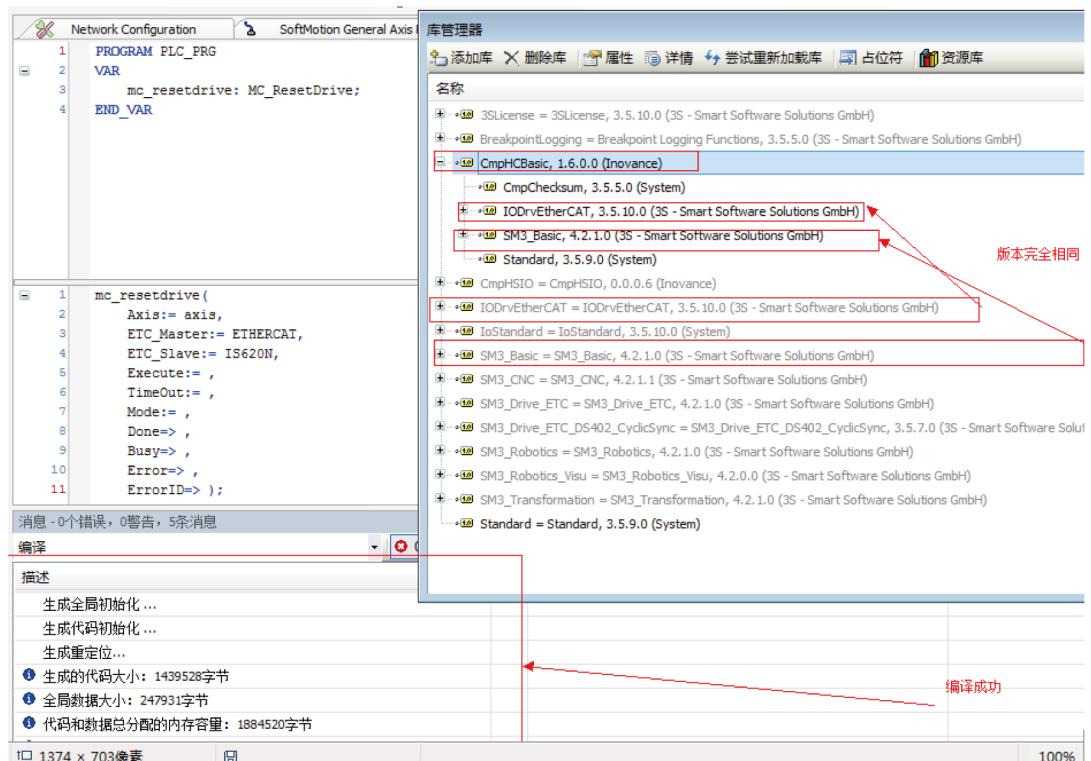


图9-7 更新后的IEC库、编译信息

## 说明

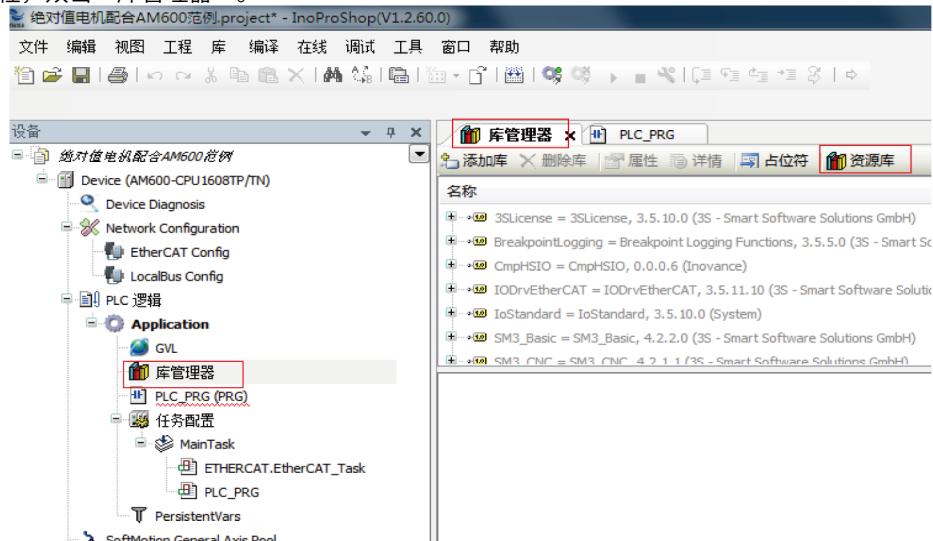
库不兼容的主要原因是使用库内依赖系统库与前工程中包含的系统库版本不一致。常见库有IODrvEtherCAT库、SM3\_Basic库。

## 如何在工程添加编译库

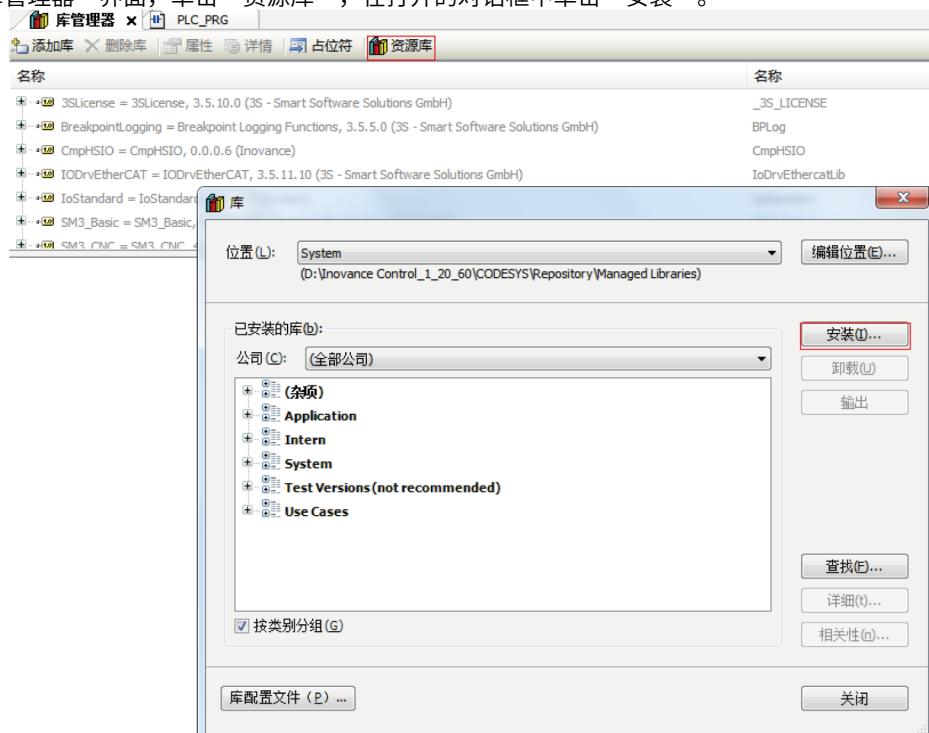
安装库以“CmpBasic.compiled-library”版本V1.11.0.0为例，后台软件版本以V1.2.60为例。其他后台软件版本操作步骤与范例相同。

### 1. 安装编译库。

#### a. 打开工程，双击“库管理器”。



#### b. 在“库管理器”界面，单击“资源库”，在打开的对话框中单击“安装”。

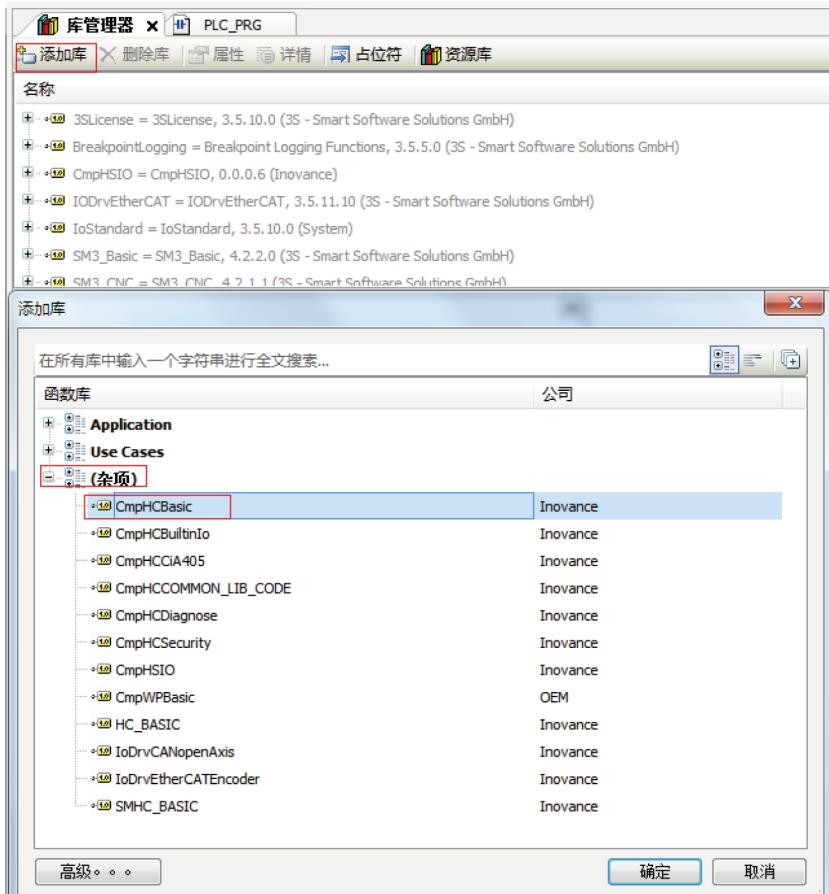


#### c. 找到需要安装的编译库存储路径 (CmpBasic.compiled-library V1.11.0.0)，选中并单击“打开”。

CmpHCBasic(V1.9.0.0 Base 3.5.9.30).compiled-library	2018/4/24 9:37	COMPILED-LIBR...	26 KB
CmpHCBasic(V1.10.0.0 Base 3.5.10.10).compiled-library	2018/4/24 9:30	COMPILED-LIBR...	39 KB
<b>CmpHCBasic(V1.11.0.0 Base 3.5.11.10).compiled-library</b>	2018/4/24 9:20	COMPILED-LIBR...	30 KB

## 2. 工程添加库。

a. 在“库管理器”界面，单击“添加库”，在打开对话框中选择“杂项”，单击“+”。



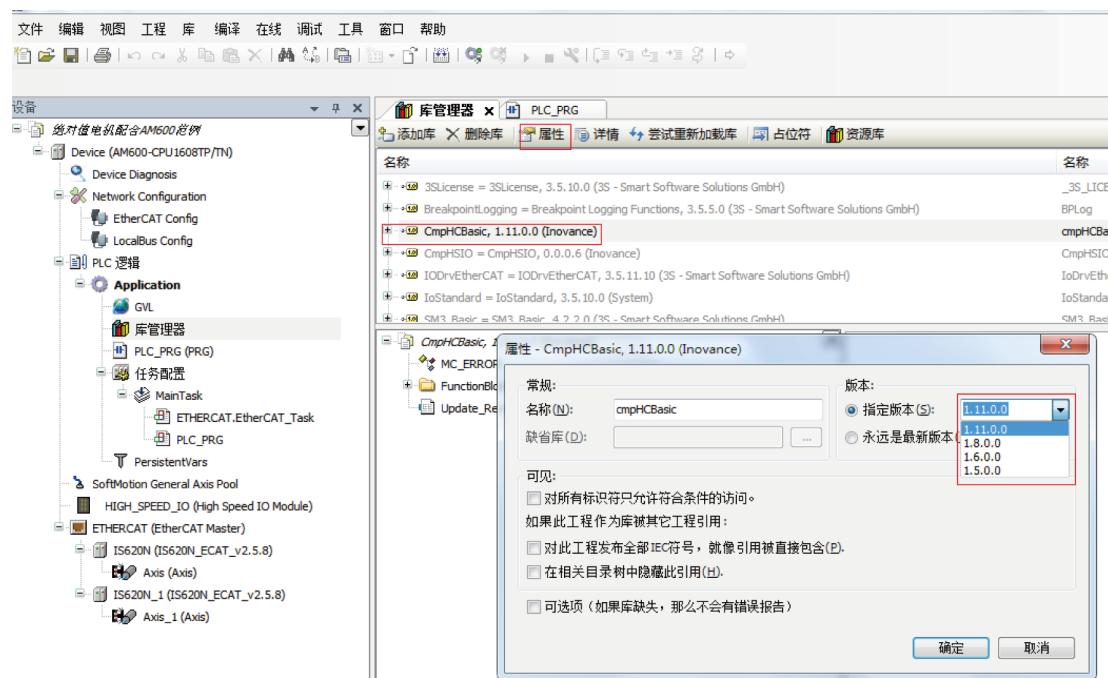
b. 选择对应的库“CmpHCBasic”并单击“确定”。默认将最高的版本添加到工程。

名称	名称	有效的版本
3SLicense = 3SLicense, 3.5.10.0 (3S - Smart Software Solutions GmbH)	_3S_LICENSE	3.5.10.0
BreakpointLogging = Breakpoint Logging Functions, 3.5.5.0 (3S - Smart Software Solutions GmbH)	BPLog	3.5.5.0
<b>CmpHCBasic, 1.11.0.0 (Inovance)</b>	<b>cmpHCBasic</b>	<b>1.11.0.0</b>
CmpHSIO = CmpHSIO, 0.0.0.6 (Inovance)	CmpHSIO	0.0.0.6
IODrvEtherCAT = IODrvEtherCAT, 3.5.11.10 (3S - Smart Software Solutions GmbH)	IODrvEtherCATLib	3.5.11.10
IoStandard = IoStandard, 3.5.10.0 (System)	IoStandard	3.5.10.0
SM3_Basic = SM3_Basic, 4.2.2.0 (3S - Smart Software Solutions GmbH)	SM3_Basic	4.2.2.0

Below the table, a tree view shows the contents of the 'CmpHCBasic, 1.11.0.0 (Inovance)' library, including 'MC\_ERROR', 'FunctionBlock', and 'Update\_Record'.

## 3. 手动选择库版本。

a. 在“库管理器”界面选中需要更新库“CmpBasic”，单击“属性”。



- b. 在打开的对话框中“版本”区域指定版本下拉选项中选择任意版本（注意库版本与后台的匹配关系，不匹配的情况使用库里面的功能会出现编译报错）。

## 说明

- 工程中添加或更新库时，必须在菜单栏选择“编译 > 清除全部”，再编译工程。
- 工程中添加或更新库，编译无错误后，登录必须重新下载（在线下载可能导致PLC异常）。

## 9.5 PLC用户程序升级

### 9.5.1 通过InoProShop工具升级

操作步骤：

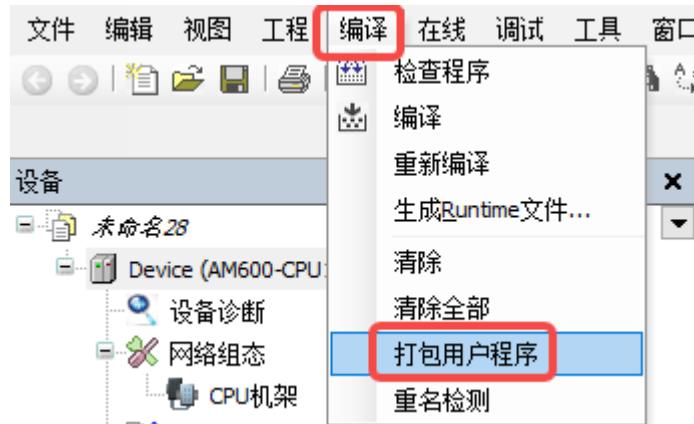
1. 在菜单栏选择“工具 > InoProShop工具”，打开 InoProShop 工具界面。
2. 单击“帮助”，打开帮助手册。
3. 升级PLC用户程序，具体操作请参见帮助手册。

### 9.5.2 通过SD卡升级

前提条件：准备一张TF卡（≤32GB）。

操作步骤：

1. 打开编译完成的工程，在菜单栏下单击“编译 > 打包用户程序”选项。



2. 在弹出的用户打包程序界面中单击“打包”完成打包用户程序操作。



3. 在选择保存的位置会生成下面 Application.userprg 文件，将这个文件复制放入SD卡根目录下。

## 说明

存放到SD卡的其他目录将无法被识别。

4. 在PLC断电状态下，将SD/TF卡插入PLC的SD卡槽。
5. 拨码开关处于Stop状态，将PLC上电，用户程序自动开始升级。  
此时数码管交替闪烁0，此过程大约持续20秒左右。



6. 待码管停止交替闪烁0，表示升级过程结束，拔出SD/TF卡，再次断电重启PLC，用户程序更新完成。
7. 拨码开关处于RUN状态下加载运行升级后的用户程序。

## 9.6 AM400/600高速I/O接线指导

AM400/600/610 CPU模块支持高速I/O数据处理功能，自带一个高密度端口，具有16路高速输入，其中前6路支持24V单端输入或差分输入，后10路支持24V单端输入。本文档对高速I/O信号接口及转接使用进行说明指导。

高密度端口（端口丝印：CN5）示意图如下：

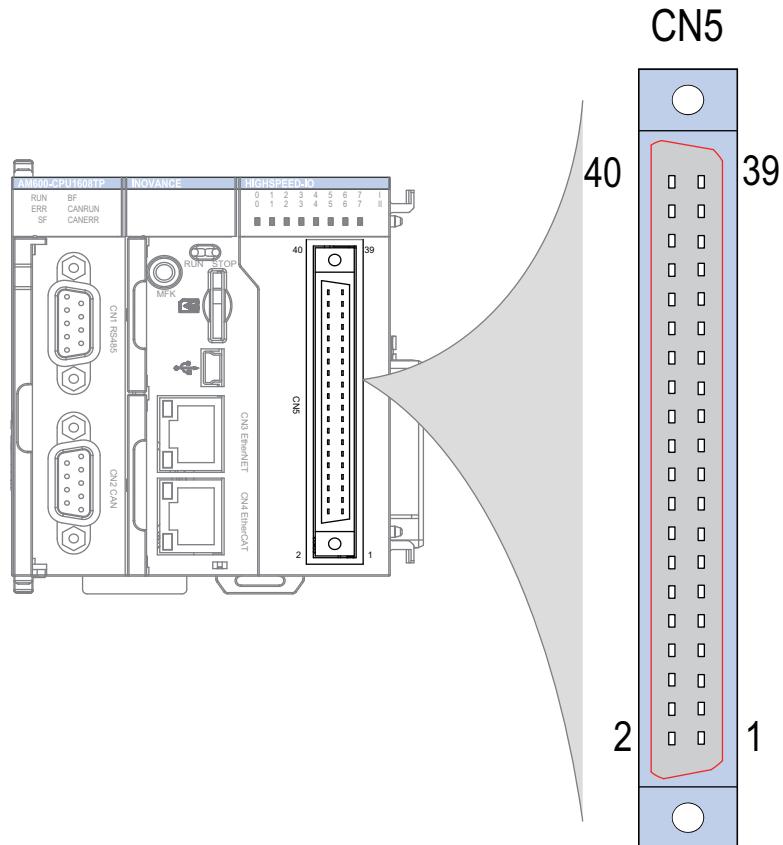
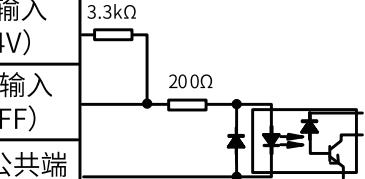
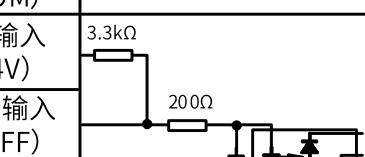
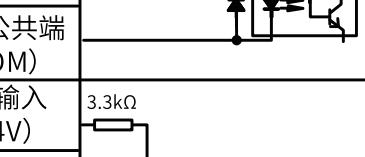
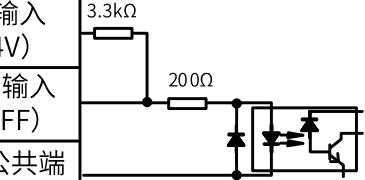
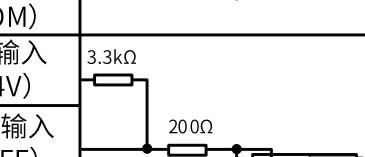
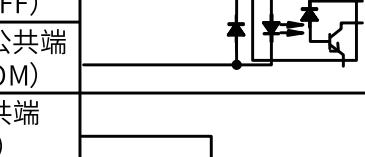
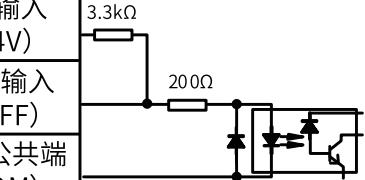
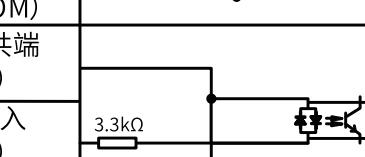
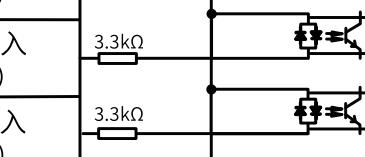
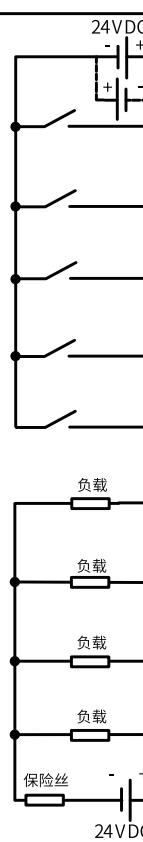
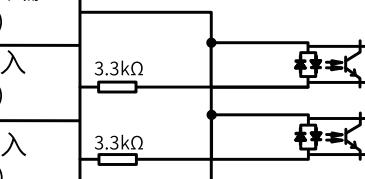
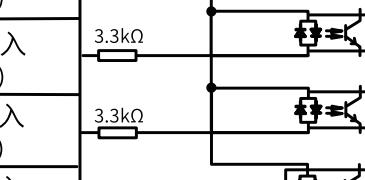
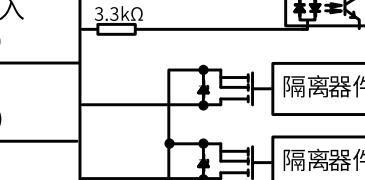
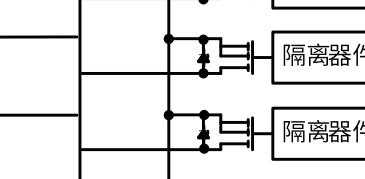
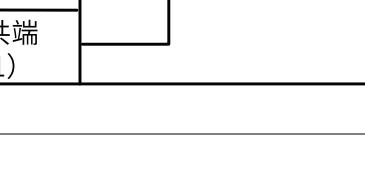
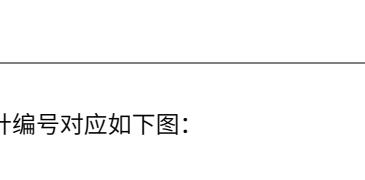
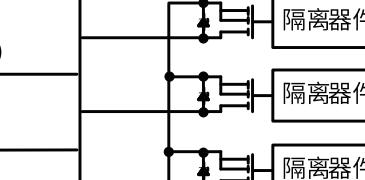
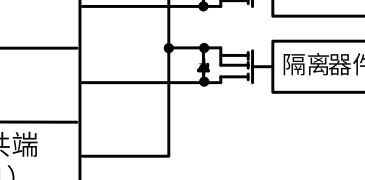
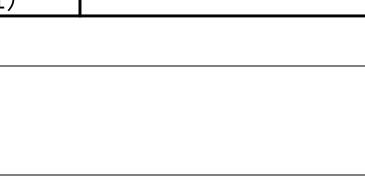
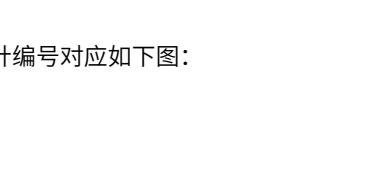


图9-8 AM600/610 CPU模块高密度端口示意图

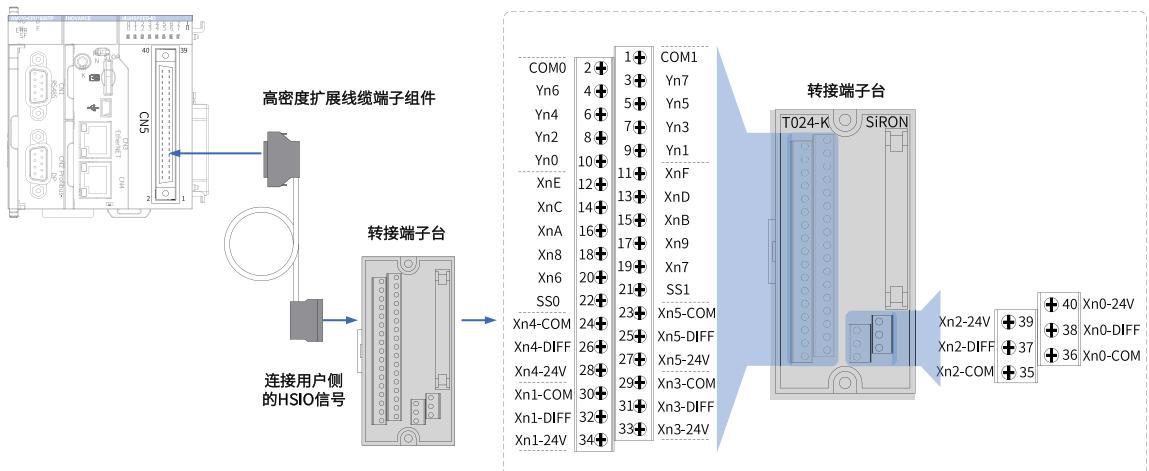
端口的内部电路及外部配线要求如下，请根据实际需求完成相应配线：

外部配线	信号名称	CN5针编 号		信号名称	内部电路
	B列		A列		
参照应 用举例	高速24V输入 (Xn0-24V)	40	39	高速24V输入 (Xn2-24V)	
	高速差动输入 (Xn0-DIFF)	38	37	高速差动输入 (Xn2-DIFF)	
	高速输入公共端 (Xn0-COM)	36	35	高速输入公共端 (Xn2-COM)	
参照应 用举例	高速24V输入 (Xn1-24V)	34	33	高速24V输入 (Xn3-24V)	
	高速差动输入 (Xn1-DIFF)	32	31	高速差动输入 (Xn3-DIFF)	
	高速输入公共端 (Xn1-COM)	30	29	高速输入公共端 (Xn3-COM)	
参照应 用举例	高速24V输入 (Xn4-24V)	28	27	高速24V输入 (Xn5-24V)	
	高速差动输入 (Xn4-DIFF)	26	25	高速差动输入 (Xn5-DIFF)	
	高速输入公共端 (Xn4-COM)	24	23	高速输入公共端 (Xn5-COM)	
	输入公共端 (SS0)	22	21	输入公共端 (SS1)	
	标准输入 (Xn6)	20	19	标准输入 (Xn7)	
	标准输入 (Xn8)	18	17	标准输入 (Xn9)	
	标准输入 (XnA)	16	15	标准输入 (XnB)	
	标准输入 (XnC)	14	13	标准输入 (XnD)	
	标准输入 (XnE)	12	11	标准输入 (XnF)	
	输出 (Yn0)	10	9	输出 (Yn1)	
	输出 (Yn2)	8	7	输出 (Yn3)	
	输出 (Yn4)	6	5	输出 (Yn5)	
	输出 (Yn6)	4	3	输出 (Yn7)	

## 说明

前6路高速DI接线使用请参见下文使用案例进行接线使用，避免出现错误接线。

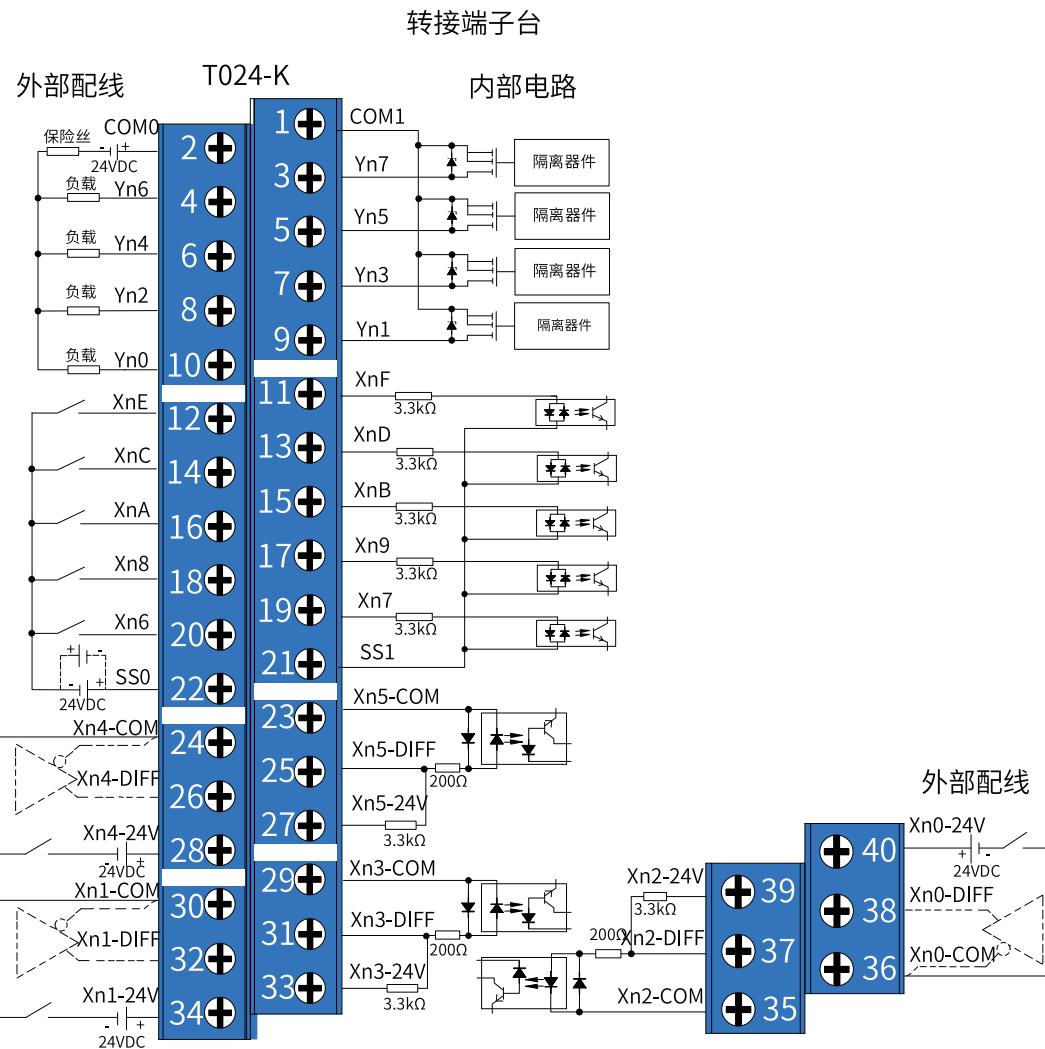
如果采用SIRON T024-K转接端子进行转接接线，端子序号与模块CN5针编号对应如下图：



其中如上图所示，汇川公司提供“①高密度扩展线缆、②连接插头（自制线缆时需要）、③转接端子台”可供选购，订货信息如下：

序号	订货编码	描述
①	15300119	40PIN FCN转MIL高密度扩展线缆 (500mm, 含两个40PIN FCN连接插头)
②	15050180	40PIN FCN连接插头 (如不选购上述线缆, 用户可选购此插头后自制线缆)
③	15020452	40PIN MIL转螺钉接线端子台

SIRON T024-K转接端子台配线示意如下：



## 说明

以上为CPU模块高密度端口针脚定义及接线说明，请详细阅读后再进行配线操作。

## 应用举例

高速I/O前4路DI支持高速单端及差分信号，使用时需注意接线正确，以Xn0为例进行应用举例说明如下。

1. 当输入为PNP型，24V电平输入时；

输入类型	外部配线	编号	信号名称	内部电路
PNP集电极型 (24V电平)		40	高速24V输入 (Xn0-24V)	3.3kΩ -> 200Ω -> * = <
		38	高速差动输入 (Xn0-DIFF)	
		36	高速输入公共端 (Xn0-COM)	

2. 当输入为NPN型，24V电平输入时；

输入类型	外部配线	编号	信号名称	内部电路
NPN集电极型 (24V电平)		40	高速24V输入(Xn0-24V)	
		38	高速差动输入(Xn0-DIFF)	
		36	高速输入公共端(Xn0-COM)	

3. 当输入为差分信号，5V电平输入时；

输入类型	外部配线	编号	信号名称	内部电路
差分信号 (5V电平)		40	高速24V输入(Xn0-24V)	
		38	高速差动输入(Xn0-DIFF)	
		36	高速输入公共端(Xn0-COM)	

## 9.7 高速IO兼容性

### 9.7.1 新旧界面介绍

CmpHCBuiltinIo和CmpHSIO分别表示旧高速I/O功能块库和新高速I/O功能块库

InProShop V1.2.0版本（临时版本中1.1.60.0）、AM600固件版本V1.19.70.0、FPGA版本A624以上支持高速I/O功能块的新库。



图9-9 高速I/O旧版本界面



图9-10 高速I/O新版本界面

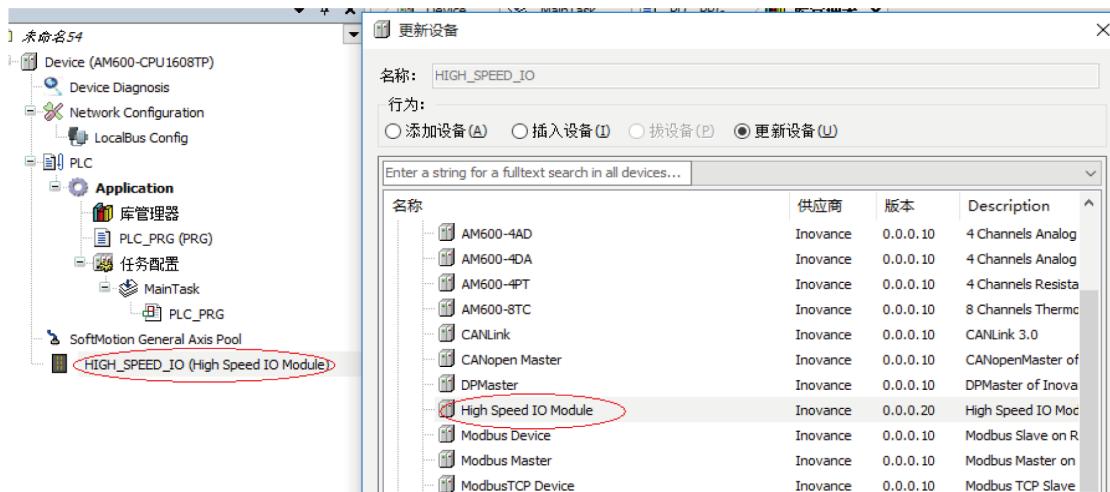
- 新高速I/O设备和新高速I/O库匹配、旧高速I/O设备和旧高速库匹配，两者不能混用。
- 新高速I/O功能需要新版本PLC固件和FPGA同时升级。

- 新高速和旧高速I/O功能可以相互切换（回原功能不兼容），如果新高速I/O和旧高速I/O混用（新高速I/O工程和旧PLC高速I/O、旧高速I/O工程和新PLC高速I/O），会提示切换PLC，登录时提示如下图所示，需PLC切换高速I/O版本，重新上电才生效。

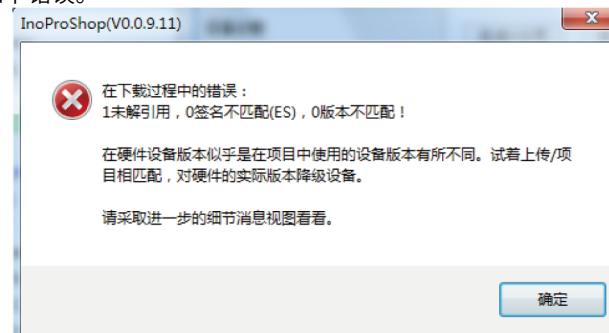


- 如果不想要切换PLC，可以切换工程的高速I/O版本，使用“更新设备”命令。

如下图所示，旧高速I/O设备版本为“0.0.0.10”，新高速I/O设备对应版本为“0.0.0.20”。



- 在旧版本后台（如1.1.0或者0.0.9.10版本）中，将旧高速I/O工程下载到新固件版本的PLC（1.19.70.0以上版本）中时会提示如下错误。



解决策施：安装最新后台，使用新后台打开旧工程（未使用高速I/O功能）后下载到新固件版本的PLC中时，不会有提示，可以正常下载使用。

## 9.7.2 高速I/O诊断

### 新版本高速I/O

基本格式为：

库 + 功能块 + 错误码

## 3 3位

- 库：高速I/O默认是0。
- 功能块：从01开始。
- 错误码：从01开始，错误码如下表所示。错误码小于500是严重错误，大于500是功能块错误。例如：14506, 14指的是HC\_WriteParameter, 506指设置参数错误；31520, 31指的是MC\_WriteParameter\_P, 520指设置参数错误。

表9-1 计数器错误码列表

错误码	定义	说明
001	ERR_COUNTERID_INVALID	输入通道号无效，有效范围是[0,7]
003	ERR_CNT_OVERFLOW	计数器上溢/下溢出错
004	ERR_COUNTER_NOT_CHOSEN	高速功能未选中，在后台配置中选择
007	ERR_COUNTER_NOT_ENABLED	计数器HC_Counter未使能
101	ERR_WRITEINTERRUPTPARAMETER_UNVALIDAD	写中断参数无效
102	ERR_INTERRUPT_NOT_CHOSE	未在后台选择“中断输入”
501	ERR_SETCOMPARE_IMREFRESHCYCLE_OVERFLOW	比较值ImRefreshCycle超过30000，有效范围是[0,30000]
502	ERR_SETCOMPAREM_NUMBERS_OVERFLOW	HC_SetCompareM的Number范围是[1,100]
503	ERR_PREWR_VALUE_OVERFLOW	预置值超限
504	ERR_AVERAGE_PARA_UNVALIDAD	频率和旋转速度设置平均参数无效
505	ERR_ROTATION_PULSES_UNIT_UNVALIDAD	转速测量的每转脉冲数设置无效
506	ERR_WRITEBOOLPARAMETER_UNVALIDAD	设置参数无效，HC_WriteBoolParameter
507	ERR_READBOOLPARAMETER_UNVALIDAD	获取参数无效HC_ReadBoolParameter
508	ERR_MEASURE_WIDTH_OVERFLOW	测量宽度无效
509	ERR_SETCOMPAREM_IMREFRESHCYCLE_OVERFLOW	比较值ImRefreshCycle超过30000，有效范围是[0,30000]
510	ERR_PRESET_TRIGGERTYPE_OVERFLOW	预置参数无效
511	ERR_WRITEPARAMETER_UNVALIDAD	设置参数无效，HC_WriteParameter
513	ERR_FUNC_COUNTERID_INVALID	特殊功能通道号无效，有效范围是[0,3]
514	ERR_COUNTER_NOT_CHOSE_EXETERNAL_X	计数器后台没有选择“外部触发输入”
515	ERR_CNT_FORMAT_NOT_RING	环形计数类型不对，在后台配置中选择
516	ERR_RING_DOWNVAL_BEYOND_UPVAL	环形计数下限值≥上限值
517	ERR_SAMPLE_VALUE_LESS	采样时间太小，采样时间范围是10~65535(10ms~65535ms)
518	ERR_RING_VALUE_OVERFLOW	环形计数超限

表9-2 高速轴错误码列表

错误码	定义	说明
001	ERR_NOT_POWER	MC_Power没使能
002	ERR_UP_SOFTWARE_LIMIT	当前位置超出软件行程限制+
003	ERR_DOWN_SOFTWARE_LIMIT	当前位置超出软件行程限制-
004	ERR_AXIS_FUNC_UNUSED	高速轴功能未使能，在后台使能
005	ERR_INPUT_CHANNAL_NUM_INVALID	轴号无效，有效范围是[0,3]
006	ERR_DEST_POS_OVER_SOFT_UP_LIMIT	目标位置超出软件上限
007	ERR_DEST_POS_OVER_SOFT_DOWN_LIMIT	目标位置超出软件下限

错误码	定义	说明
010	ERR_POS_DECPOINT_OVERFLOW	减速点无效：位置模式下，重新定位时，减速长度大于实际距离
011	ERR_VEL_DECPOINT_OVERFLOW	减速点无效：速度模式下，切换为定位时，减速长度大于实际距离
012	ERR_POS_PLNUM_OVERFLOW	超过PLNUM最大定位长度2147483647
013	ERR_POS_DECPOINT2_OVERFLOW	2次求减速点出错
501	ERR_ACC_SET_OVERFLOW	加速度超出允许范围，超出MC_WriteParameter_P设定的最大加速度
502	ERR_ACC_SET_LOW	加速度设置太小，低于MC_WriteParameter_P设定的最小加速度
503	ERR_DEC_SET_OVERFLOW	减速度超出允许范围，超出MC_WriteParameter_P设定的最大减速度
504	ERR_DEC_SET_LOW	减速度设置太小，低于MC_WriteParameter_P设定的最小减速度
505	ERR_VEL_SET_OVERFLOW	速度设置超出允许范围，后台或者MC_WriteParameter_P设置
506	ERR_VEL_SET_LOW	速度设置太小
508	ERR_VEL_LESS_THAN_STARTVEL	速度小于启动偏置速度，后台设置启动偏置速度
509	ERR_STARTVEL_SET_LOW	起始速度太小
510	ERR_FBD_MOVMODE_INVALID	功能块的运动模式无效，状态不对
511	ERR_WASNT_STANDSTILL	当前状态不是STANDSTILL
512	ERR_WASNT_DISABLED	当前状态不是DISABLE
513	ERR_IN_ERRORSTOP	当前状态是ERRORSTOP
514	ERR_NOT_READY_FOR_MOTION	轴没准备好，不能运行
515	ERR_INVALID_VELOCITY_MODE	无效的速度模式
516	ERR_INVALID_POSITION_MODE	无效的位置模式
520	ERR_AXIS_WRITEPARAMETER_UNVALID	MC_WriteParameter_P参数无效
521	ERR_AXIS_READPARAMETER_UNVALID	MC_ReadParameter_P参数无效
522	ERR_HOME_MODE_UNVALID	回原模式无效，后台选择
523	ERR_AXIS_WRITEPARAMETER_HOME_MODE_UNVALID	回原模式设置无效

错误分为：轴错误和功能块错误

置为ErrorStop状态的条件为：

- 发生轴错误。
- 在DiscreteMotion、ContinuousMotion和Homing状态时发生功能块错误。

## 旧版本高速I/O

高速I/O自身诊断界面显示高速I/O诊断信息，关于自身诊断界面的描述，请参考设备自身诊断信息列表简介。

高速I/O诊断主要通过获取高速I/O诊断软元件获得的。高速I/O诊断包括通道出错、通道报警、轴出错、轴报警和其它故障五类。通道出错、通道报警、轴出错、轴报警使用软元件表示诊断状态和诊断码。诊断状态表示此是否出现诊断信息，诊断码表示出现错误的代码。每种类型对应的软元件表、诊断码诊断信息表如下：

- 通道出错

通道号和错误标志软元件、错误诊断码软元件关系如下表所示。

通道号	出错标志软元件	出错诊断码软元件
0	SM9030	SD9007
1	SM9080	SD9017
2	SM9130	SD9027
3	SM9180	SD9037
4	SM9230	SD9047
5	SM9380	SD9057
6	SM9330	SD9067
7	SM9380	SD9077

诊断码和诊断信息对应关系如下表所示。

诊断码	诊断信息
1001	通道类型不匹配
1002	计数器溢出
1003	脉冲宽度测量溢出
1011	环形计数器下限值超过上限值
1012	计数器类型不匹配
1013	高速计数功能未使用
1014	高速计数器功能选择不匹配
1015	预置值超限
1016	平均参数无效
1017	转速测量的每转脉冲数设置无效

#### ● 通道报警

通道号和报警标志软元件、报警诊断码软元件关系如下表所示。

通道号	报警标志软元件	报警诊断码软元件
0	SM9031	SD9008
1	SM9081	SD9018
2	SM9131	SD9028
3	SM9181	SD9038
4	SM9231	SD9048
5	SM9381	SD9058
6	SM9331	SD9068
7	SM9381	SD9078

诊断码和诊断信息对应关系如下表所示。

诊断码	诊断信息
1501	采样值上溢

#### ● 轴出错

轴号和错误标志软元件、错误诊断码软元件关系如下表所示。

轴号	出错标志软元件	出错诊断码软元件
0	SM9405	SD9105
1	SM9425	SD9125
2	SM9445	SD9145
3	SM9465	SD9165

诊断码和诊断信息对应关系如下表所示。

诊断码	诊断信息
2001	正方向硬件限位
2002	负方向硬件限位
2003	始动时停止命令ON
2004	正向软件限位

诊断码	诊断信息
2005	负向软件限位
2006	运行中cpu模块切换为stop状态
2007	驱动模块就绪OFF
2008	零点信号ON
2009	没有实施机械原点回归
2010	重试出错
2011	ABS传送超时
2012	ABS传送sum
2013	速度0出错
2014	加减速时间设置超时
2015	减速停止时间设置超时
2016	速度位置切换控制中移动量设置超出了允许范围
2017	禁止速度位置切换启动
2018	非轴停止状态更改当前值
2019	加减速时间设置为0
2020	始动时轴未停止
2021	启动时遇轴停止命令

- 轴报警

轴号和报警标志软元件、报警诊断码软元件关系如下表所示。

轴号	报警标志软元件	报警诊断码软元件
0	SM9406	SD9106
1	SM9426	SD9126
2	SM9446	SD9146
3	SM9466	SD9166

诊断码和诊断信息对应关系如下表所示。

诊断码	诊断信息
2501	超出速度范围
2502	禁止目标位置更改
2503	禁止速度更改

- 其它故障

其它故障当前定义了高速I/O功能块输入参数无效诊断。此诊断不能通过软元件获取，可以在高速I/O自身诊断界面和诊断信息列表界面查看。其诊断代码和诊断信息如下表所示。

诊断码	诊断信息
1018	高速输入功能块的通道号配置无效
1019	高速输入功能块的输入参数配置无效
2022	高速输出功能块的通道号配置无效
2023	高速输出功能块的输入参数配置无效

## 9.8 诊断码和诊断信息

### 9.8.1 概述

每个诊断码都有名称，与诊断编程接口对应类型名称匹配，具体请参见[第455页“诊断编程接口”](#)。

## 9.8.2 CPU 诊断码

名称	诊断码	诊断信息
SDCardError	1	SD卡错误
FlashError	1	Flash错误
SystemError	0 x 40	高速I/O接口板连接错误
InterCommError	0x11	无IO扩展模块(板间通信错误:读校验失败)
	0x12	无IO扩展模块(板间通信错误:写校验失败)
	0x13	无IO扩展模块(板间通信错误:ACK为高电平)
	0x14	无IO扩展模块(板间通信错误: ACK低电平)
	0x21	实际IO扩展模块数少于组态配置(板间通信错误:读校验失败)
	0x22	实际IO扩展模块数少于组态配置(板间通信错误:写校验失败)
	0x23	实际IO扩展模块数少于组态配置(板间通信错误:ACK为高电平)
	0x24	实际IO扩展模块数少于组态配置(板间通信错误: ACK低电平)
	0x31	实际IO扩展模块数多于组态配置(板间通信错误:读校验失败)
	0x32	实际IO扩展模块数多于组态配置(板间通信错误:写校验失败)
	0x33	实际IO扩展模块数多于组态配置(板间通信错误:ACK为高电平)
	0x34	实际IO扩展模块数多于组态配置(板间通信错误: ACK低电平)
	0x41	IO扩展模块类型错误(板间通信错误:读校验失败)
	0x42	IO扩展模块类型错误(板间通信错误:写校验失败)
	0x43	IO扩展模块类型错误(板间通信错误:ACK为高电平)
	0x44	IO扩展模块类型错误(板间通信错误: ACK低电平)
ConformanceError (每位表示一个模块故障)	1	插槽1对应的IO模块和实际IO模块组态不一致
	2	插槽2对应的IO模块和实际IO模块组态不一致
	4	插槽3对应的IO模块和实际IO模块组态不一致
	8	插槽4对应的IO模块和实际IO模块组态不一致
	16	插槽5对应的IO模块和实际IO模块组态不一致
	32	插槽6对应的IO模块和实际IO模块组态不一致
	64	插槽7对应的IO模块和实际IO模块组态不一致
	128	插槽8对应的IO模块和实际IO模块组态不一致
	256	插槽9对应的IO模块和实际IO模块组态不一致
	512	插槽10对应的IO模块和实际IO模块组态不一致
	1024	插槽11对应的IO模块和实际IO模块组态不一致
	2048	插槽12对应的IO模块和实际IO模块组态不一致
	4096	插槽13对应的IO模块和实际IO模块组态不一致
	8192	插槽14对应的IO模块和实际IO模块组态不一致
	16384	插槽15对应的IO模块和实际IO模块组态不一致
	32768	插槽16对应的IO模块和实际IO模块组态不一致

名称	诊断码	诊断信息
IOModulePosError (每位表示一个模块故障, 因具体故障在IO模块已经显示, 所以此诊断信息不显示, 只作标志状态)	1	插槽1对应的IO模块出现故障
	2	插槽2对应的IO模块出现故障
	4	插槽3对应的IO模块出现故障
	8	插槽4对应的IO模块出现故障
	16	插槽5对应的IO模块出现故障
	32	插槽6对应的IO模块出现故障
	64	插槽7对应的IO模块出现故障
	128	插槽8对应的IO模块出现故障
	256	插槽9对应的IO模块出现故障
	512	插槽10对应的IO模块出现故障
	1024	插槽11对应的IO模块出现故障
	2048	插槽12对应的IO模块出现故障
	4096	插槽13对应的IO模块出现故障
	8192	插槽14对应的IO模块出现故障
	16384	插槽15对应的IO模块出现故障
	32768	插槽16对应的IO模块出现故障
FunctionErrorCode (每位表示一个总线出现故障, 只作标志状态)	0x01	DP总线出现故障
	0x02	EtherCAT总线出现故障
	0x04	CANopen总线出现故障
	0x08	CANlink总线出现故障
	0x10	ModbusTCP出现故障
	0x20	Modbus串口0出现故障
	0x40	Modbus串口1出现故障
	0x80	高速I/O出现故障

## 说明

因EtherCAT为Codesys实现, PLC暂时无法直接获取EtherCAT诊断, EtherCAT总线标志暂时没用。

### 9.8.3 IO 模块诊断码

名称	模块类型	诊断码	诊断信息
BaseInfo	所有模块	64	出现故障, 系统停机
ModuleError (每位表示一种模块故障)	AI	2	无外部负载电压
		4	模拟芯片连接错误
	AO	2	无外部负载电压
		4	模拟芯片连接错误
		8	模拟芯片过热

名称	模块类型	诊断码	诊断信息
ChannelError[i] (数组每个元素表示每个通道 诊断代码，每位表示一种故 障)	AI	2	上溢
		4	下溢
		8	超上限
		16	超下限
		32	断线
	AO	2	上溢
		4	下溢
		8	电流断线
		16	电压短路
		32	DAC通道硬件故障

#### 9.8.4 DP 诊断码

名称	诊断码	诊断信息
ExtDiagData[0] (每位表示一种故障)	0x02	未准备好进行数据交换
	0x04	组态错误
	0x08	该从站有扩展的诊断信息
	0x10	该从站不支持所请求的功能
	0x20	无效的从站响应
	0x40	参数设置错误
	0x80	被不同的主站锁定
ExtDiagData[1] (每位表示一种故障)	0x01	必须重新设置参数
	0x02	静态诊断
	0x08	看门狗监控被激活
	0x10	该从站处理锁存模式
	0x20	该从站处理同步模式
	0x80	该从站未被主站激活
ExtDiagData[2] (每位表示一种故障)	0x80	诊断数据溢出
ExtDiagData[3] (主站地址)	-	-
ExtDiagData[4]和ExtDiagData[5] (从站ID)	-	-

#### 说明

诊断前6个字节为基本诊断，后面的诊断数据位扩展诊断，具体解析详见DP诊断。

### 9.8.5 CANlink 诊断码

名称	诊断码	诊断信息
CmdFrameError (诊断码为除以100的余数)	1	命令码非法
	2	命令码地址异常
	3	数据值不在允许范围内
	4	命令码操作无效
	5	命令码数据长度无效
	6	命令码超时
CfgFrameError (诊断码为除以100的余数)	1	配置编码出错
	2	配置索引出错
	3	配置信息出错
	5	配置数据长度出错
	6	配置帧未响应

## 9.8.6 Modbus 诊断码

名称	诊断码	诊断信息
DiagData	0x70	Modbus从站地址设置错误
	0x71	数据帧长度错误，串口0 (COM0)
	0x72	非法数据地址
	0x73	CRC校验错误
	0x74	不支持的命令码
DiagData	0x75	接收超时
	0x76	非法数据值
	0x77	缓冲区溢出
	0x78	帧错误
	0x79	串口协议错误
	0x7C	地址错误
	0x7D	未收到数据
	0x7E	从站返回错误数据
	0x80	Modbus从站地址设置错误
	0x81	数据帧长度错误，串口1 (COM1)
	0x82	非法数据地址
	0x83	CRC校验错误
	0x84	不支持的命令码
	0x85	接收超时
	0x86	非法数据值
	0x87	缓冲区溢出
	0x88	帧错误
	0x89	串口协议错误
	0x8C	地址错误
	0x8D	未收到数据
	0x8E	从站返回错误数据
	0x90	Modbus从站地址设置错误
	0x91	数据帧长度错误，以太网 (MODBUS TCP)
	0x92	非法数据地址
	0x93	CRC校验错误
	0x94	不支持的命令码
	0x95	接收超时
	0x96	非法数据值
	0x97	缓冲区溢出
	0x98	帧错误
	0x99	串口协议错误
	0x9A	从站无连接
	0x9B	协议标示符不正确
	0x9C	地址错误
	0x9D	未收到数据
	0x9E	从站返回错误数据
	0x9F	客户端连接数超限
	0xA0	非法数据值

### 9.8.7 EtherCAT 故障码

EtherCAT故障ID分为EtherCAT总线故障ID和EtherCAT从站故障ID，其中EtherCAT总线故障ID包含主站故障和从站故障，变量为m\_LastError，EtherCAT从站故障ID对从站故障进行描述，变量为ErrorCode。

名称	诊断码	诊断信息
m_LastError	0x1	主站通信异常，连续丢失100帧以上数据
	0x2	部分从站掉线，在线从站数量与配置数量不一致
	0x3	DC时钟异常，参考时钟一直不变化
	0x4	网卡打开失败
	0x5	冗余网卡打开失败
	0x6	冗余网卡打开失败，冗余功能配置了同一网卡
	0x7	从站初始化错误，启动过程中从站不存在，或者无法建立通信
	0x8	供应商ID不匹配，配置的与实际的不一致
	0x9	产品ID不匹配，配置的与实际的不一致，或主站读取该从站产品ID失败
	0xA	从站数量不匹配，配置的从站数量大于实际从站数量
	0xB	SDO下载失败
	0xC	SDO下载超时
	0xD	从站紧急事件错误
	0xE	SOE下载失败
	0xF	SOE下载超时
	0x10	主站请求状态机超时
	0x20	别名地址冲突，实际网络多个从站启用相同的别名地址
	0x21	从站IN/OUT连接错误，
	0x22	EEPROM访问失败，启动过程主站访问从站EEPROM失败
	0x30	持续丢帧故障
	0x31	从站端口链接断开
	0x32	偶发丢帧警告
	0x64	切换通信状态失败
	0x65	从站未知错误
	0x66	从站邮箱申请内存失败
	0x6A	固件匹配错误，从站固件版本EEPROM存储信息不一致
	0x6B	从站更新固件失败
	0x75	状态机错误
	0x76	从站接收到未知状态改变请求
	0x77	状态机错误，从站不支持引导模式
	0x78	固件程序无效
	0x79	邮箱配置错误，从站引导状态下检测到邮箱配置错误
	0x7A	邮箱配置错误，从站预运行状态检测到邮箱配置错误
	0x7B	同步管理器错误，从站检测到同步管理器配置无效
	0x7C	输入数据无效
	0x7D	输出数据无效
	0x7E	同步错误
	0x7F	同步管理器看门狗超时
	0x80	同步管理类型无效
	0x81	输出PDO配置无效
	0x82	输入PDO配置无效
	0x83	看门狗配置无效
	0x84	从站需要冷启动
	0x85	需要初始化状态
	0x86	需要预操作状态
	0x87	需要安全操作状态
	0x88	输入映射无效，从站不支持输入PDO参数配置

名称	诊断码	诊断信息
m_LastError	0x89	输出映射无效，从站不支持输出PDO参数配置
	0x8A	从站不一致的设置
	0x8B	模式配置错误，从站不支持自由运行模式
	0x8C	模式配置错误，从站不支持同步运行模式
	0x8E	参数配置错误，从站自由运行模式需要配置3个缓冲区
	0x8F	输入和输出无效
	0x90	DC同步错误，从站DC模式下，Sync0看门狗超时
	0x91	DC同步错误，从站安全模式到运行模式过程未检测到Sync0中断信号
	0x92	DC同步错误，从站同步周期时间太小 从站地址({Addr})
	0x94	DC同步配置无效
	0x95	DC锁存配置无效
	0x96	PLL错误，从站同步丢失主站失败
	0x97	DC无效
	0x98	DC超时错误
	0x99	同步循环时间错误
	0x9A	Sync0配置错误，从站Sync0周期的超出范围
	0x9B	Sync1配置错误，从站Sync1周期的超出范围
	0xA5	从站MBX_AOE错误
	0xA6	从站MBX_EOE错误
	0xA7	从站MBX_COE错误
	0xA8	从站MBX_FOE错误
	0xA9	从站MBX_SOE错误
	0xB3	从站MBX_VOE错误
	0xB4	从站EEPROM 地址不能访问
	0xB5	从站EEPROM 错误
	0xB6	从站外部硬件未准备好
	0xC4	从站已在本地重新启动
	0xD4	从站配置错误，从站耦合器挂载模块配置与实际不一致
ErrorCode	0X08	供应商ID不匹配故障
	0X09	产品不匹配故障
	0X20	别名地址冲突
	0X21	IN/OUT反接故障
	0X22	EEPROM访问失败
	0X30	连续丢帧故障
	0X31	从站OUT端口链接断开
	0X32	偶发丢帧故障
	0X64	切换通信状态失败

### 9.8.8 轴诊断码

诊断码	诊断信息
0x1	驱动器总线通讯故障
0x2	驱动器故障
0x3	总线DC同步丢失
0xA	软件限位超限
0xB	硬件限位超限

诊断码	诊断信息
0xC	轴在线性模式下位置超出最大允许范围
0xD	驱动器不支持快速急停或暂停功能
0xE	无
0xF	无
0x10	目标位置与实际位置偏差超出滞后限制值
0x11	驱动器回零故障
0x12	无
0x14	轴处于未使能状态，执行了运控功能块
0x15	功能块执行中检查到不支持的控制模式，无法执行
0x19	功能块不支持对逻辑轴的操作
0x1B	无
0x1E	执行中的功能块在运动期间未被调用
0x1F	功能块输入参数轴类型错误
0x20	功能块执行过程中轴实例发生变化
0x21	功能块执行过程中断开使能
0x22	功能块触发时轴状态不满足PLCopen状态机的要求
0x23	轴运动过程驱动器故障
0x28	虚轴速度超过限制值
0x29	虚轴加速度超过限制值
0x2A	虚轴减速度超过限制值
0x32	上位机回零输入参数无效
0x33	上位机回零未配置硬件限位
0x3C	文件读写缓存通道已满，可注册的句柄为空
0x41	SDO多通道通信初始化失败，未获取到激活的APP
0x42	无效的IEC任务句柄
0x43	SDO多通道中存在有太多的任务
0x44	SDO多通道底层接口调用错误
0x45	无
0x46	功能块或驱动器不支持该控制模式
0x47	功能块触发时轴处于特定状态不能更改控制模式
0x48	控制模式切换时被中断
0x4B	当前控制模式不在同步力矩模式，实例不能运行。
0x50	功能块复位轴过程复位失败
0x51	功能块复位轴过程中初始化失败
0x55	功能块输入错误的轴类型
0x56	功能块输入参数无效
0x5A	功能块输入参数为零
0x5B	功能块无法在驱动器使能状态下执行
0x5C	旋转轴周期值设置无效
0x5D	旋转位置周期不是整数值
0x6E	任务循环时间设置为0
0x78	功能块无错误可复位
0x79	驱动器复位请求未响应
0x7A	驱动器故障不能被复位
0x7B	复位时驱动器响应超时
0x7C	总线通讯错误无法复位
0x82	功能块输入未知参数
0x83	功能块读驱动器参数错误

诊断码	诊断信息
0x84	功能块输入参数不在映射表中
0x85	功能块内部数据转换出错
0x8C	功能块输入未知参数
0x8D	功能块写驱动器参数错误
0x8E	功能块输入参数不在映射表中
0x8F	功能块内部数据转换出错
0xAA	轴不在Standstill状态
0xAB	回原指令写参数失败
0xAC	回原指令读参数未响应
0xAD	无
0xAE	回零时轴在ErrorStop状态下
0xB4	停止功能块在停止过程中被中止
0xB5	停止功能块减速速度输入值无效的
0xB6	方向为shortest 不可用
0xB7	轴在ErrorStop状态下
0xB8	执行的停止功能块在总线周期中未被调用
0xB9	功能块执行在Stopping状态
0xC8	任务周期时间设置为0
0xC9	功能块输入速度加速度值无效
0xCA	功能块方向参数无效
0xE2	功能块输入速度加速度值无效
0xE3	功能块方向参数无效
0xFB	功能块输入速度加速度值无效
0xFC	功能块方向参数无效
0x114	功能块输入速度加速度值无效
0x115	功能块方向参数无效
0x116	功能块无效的执行顺序
0x12C	无
0x12D	功能块输入速度加速度值无效
0x12E	功能块输入方向参数不支持
0x145	功能块输入ArraySize不合法
0x146	功能块输入时间参数不合法
0x15E	功能块输入ArraySize不合法
0x15F	功能块输入时间参数不合法
0x177	功能块输入ArraySize不合法
0x178	功能块输入时间参数不合法
0x190	探针通道被占用
0x191	驱动器不支持窗口探针功能
0x192	探针通讯错误
0x19A	探针不能被终止
0x1AA	功能块输入速度加速度值无效
0x1AB	功能块输入方向参数无效
0x1C3	功能块输入速度加速度值无效
0x1C4	功能块输入方向参数无效
0x1C5	功能块输入方向参数不支持
0x1DB	功能块执行只能在standstill或者power_off的状态下
0x1DC	功能块输入速度加速度值无效

诊断码	诊断信息
0x258	凸轮表中无挺杆
0x259	挺杆数量设置太多
0x25A	一个凸轮CAM表超过32个激活的挺杆操作
0x271	凸轮表为空
0x272	凸轮表主轴位置超出凸轮周期范围
0x273	凸轮动态耦合未输入速度、加速度值
0x274	凸轮关键点超出范围
0x275	在一个周期内激活太多的挺杆
0x280	功能块输入凸轮类型参数不支持
0x2A3	输入齿轮比分母为0
0x2A4	输入无效的加速度
0x2A5	输入无效的减速度
0x2A6	主轴使能状态改变
0x2A7	功能块的输入参数Jerk不合法
0x2D5	功能块输入速度加速度值无效
0x2D6	轴旋转周期值为零
0x2EE	输入非凸轮表结构体类型参数
0x2EF	凸轮表关键点不在主站范围内
0x2F0	凸轮表主轴起始值大于的结束值
0x2F1	凸轮表无效的主轴位置
0x2F2	凸轮表无效的从轴位置
0x307	主轴方向发生改变
0x308	从轴无法避免反转
0x309	功能块输入参数不支持线性轴
0x30A	Buffered模式主站开始距离必须为0
0x30B	不能开启运动同步
0x320	补偿间隙值太大
0x339	轨迹生成内部错误，算法不收敛
0x33A	轨迹生成内部错误，无效的参数值
0x33B	轨迹生成内部错误，轴计算无结果
0x33C	轨迹生成内部错误，下限持续时间减少
0x33D	轨迹生成内部错误，相关轴找不到共同的持续时间
0x33E	轨迹生成内部错误，无效的结果间隔
0x33F	生成S形速度轮廓需要更多的相位参数
0x340	轨迹生成内部错误
0x352	功能块执行轴的状态不在Standstill或者poweroff
0x353	功能块输入无效的参数
0x354	功能块执行轴的状态不在Standstill或者poweroff
0x355	功能块输入无效的位置模式和周期值
0x356	功能块输入轴不是虚轴
0x79F	运动缓存中前一个运动不支持Blending
0x7A0	运动缓存中前一个运动不支持BufferMode
0x7A1	上一条缓存指令当前周期没有被激活调用
0x7A2	轴的运动没有功能块接管
0x7A3	输入不支持的BufferMode参数
0x7A4	运动缓存中前一个运动发生错误
0x7A5	功能块实例已在运动缓存队列，无法再次添加

诊断码	诊断信息
0x4E20	驱动器DC时钟没有同步
0x4E21	驱动器不在OP模式下
0x4E22	启动过程中等待DC同步失败
0x4E23	驱动器缩放比配置为0
0x4E24	SMC_Basic版本太低
0x4E25	没有SoftMotion授权
0x4E26	没有检测到激活的App
0x4E27	轴设备被禁用
0x4E28	没有找到轴设备
0x4E29	当前轴和逻辑轴不在相同的任务中
0x4E2A	获取轴后台配置参数失败
0x4E2B	无效的设备类型
0x4E2C	任务周期设置太大
0x4E2D	获取轴映射参数失败
0x5015	力矩限制指令读写伺服参数过程从站未响应
0x5016	力矩限制指令读写伺服参数过程超时
0x501F	读取数字量输入过程从站未响应,
0x5020	读取数字量输入超时
0x5028	力矩转位置控制模式失败
0x5032	力矩指令执行过程中输入斜率或输入速度超过有效范围
0x5033	转矩模式下飞车
0x5034	转矩模式下叠加无效
0x5035	转矩指令未配置最大轮廓速度PDO
0x5036	转矩指令未配置期望力矩PDO
0x503C	偏差复位指令执行过程中轴状态出错
0x503D	偏差复位指令不能在stopping状态下执行
0x503E	偏差复位指令执行过程中被打断
0x5046	力矩转位置控制模式失败
0x5050	中止叠加指令执行顺序出错, 或程序中无叠加指令
0x5051	中止叠加指令输入变量无效
0x505A	探针相关PDO设定不足
0x505B	超出中断定长指令缓存模式范围
0x505C	中断定长指令线性模式下上边界大于下边界
0x505D	探针已被占用
0x505E	绝对、相对或速度控制过程发生错误
0x505F	中断定长过程发生错误
0x5060	探针未触发
0x5061	中断定长指令旋转模式下上下边界差超过旋转周期
0x5062	进给距离输入无效
0x5063	中断定长指令输入变量无效
0x5064	主轴在错误状态下机下
0x5065	从轴在错误状态下机下
0x5066	主轴在错误控制模式下
0x5067	从轴在错误控制模式下
0x5068	加减速段之和不能超过1
0x5069	加减速段输入超限
0x506A	主轴位移输入不合理

诊断码	诊断信息
0x506B	从轴位移输入不合理
0x506C	重复触发后不允许等待
0x506D	叠加运动过程中主轴运动方向改变
0x506E	同步叠加指令从轴叠加位移方向出错
0x506F	曲线类型不得超过可选范围
0x5078	轴组中单轴不允许执行单轴指令
0x754E	SDO读/写错误
0x754F	SDO读/写错误

## 9.9 同步工程信息

### 9.9.1 概述

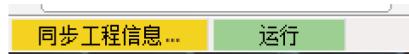
同步工程信息，主要是将工程数据与下载数据同步，保证准确登录PLC。

同步工程信息主要解决非预期的下载问题：

- 只拷贝工程文件
- 一个工程对多个PLC调试
- 多人同步编程调试。
- 工程“清除全部”，需要下载

### 9.9.2 自动下载同步工程信息

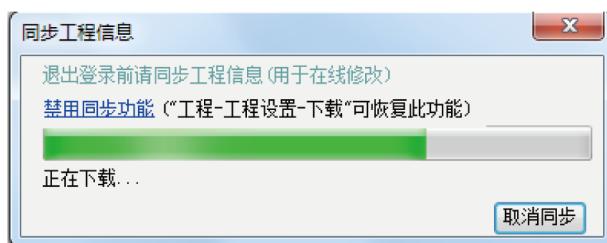
如果勾选菜单【工程】 - 【工程设置…】 - 【下载】 - 【下载工程信息】，当登录下载完用户程序后，会自动启动工程信息同步，在同步过程中，状态栏【同步工程信息…】橙色闪烁，如下图所示：



如果双击状态栏【同步工程信息…】，则显示当前同步过程，如下图所示：



如果同步工程信息没有完成，退出登录时，会弹出同步工程信息状态框，如下图所示：



- 同步完成后，同步工程信息界面自动关闭，退出登录。

- 如果点击“X”，关闭界面，取消退出登录。
- 如果选择【取消同步】，同步工程过程中断，状态界面关闭，退出登录。
- 如果点击【禁用同步功能】，则当前工程的“同步工程信息”功能禁用，状态界面关闭，退出登录。

### 9.9.3 手动下载同步工程信息

通过菜单命令【在线】-【同步工程信息…】，可以同步工程信息。此命令在登录PLC后才能执行。

执行此命令后，显示同步状态，如下图：



手动下载工程信息，会强制下载，不管工程是否禁用了“同步工程信息”功能。

### 9.9.4 同步工程信息的特殊说明

- 如果CPU正常运行负载比较高（超过85%），可能“同步工程信息”过程相对较慢
- 如果需要对单PLC进行多次调试，可先禁用同步功能，调试完手动操作“同步工程信息”。

## 9.10 SVN功能

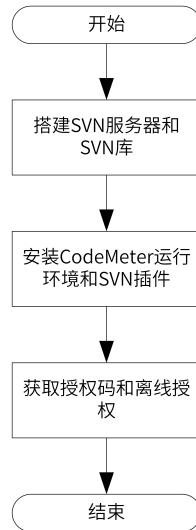
### 9.10.1 概述



InoProShop V1.8.0.0及以上版本支持该功能。

InoProShop软件支持将工程文件纳入SVN库进行管理，例如将工程文件导入至SVN库或将SVN库的工程文件导出至InoProShop软件。

使用SVN功能之前，需要先按照以下安装配置流程搭建SVN环境。



### 9.10.2 搭建SVN服务器和SVN库

搭建SVN服务器（VisualSVN Server）和SVN库（SVN Repository），具体操作方法请自行访问[CODESYS官网](#)下载并参考《CODESYS\_SVN》手册或联系贵司IT部门获得支持。

### 9.10.3 安装CodeMeter运行环境和SVN插件

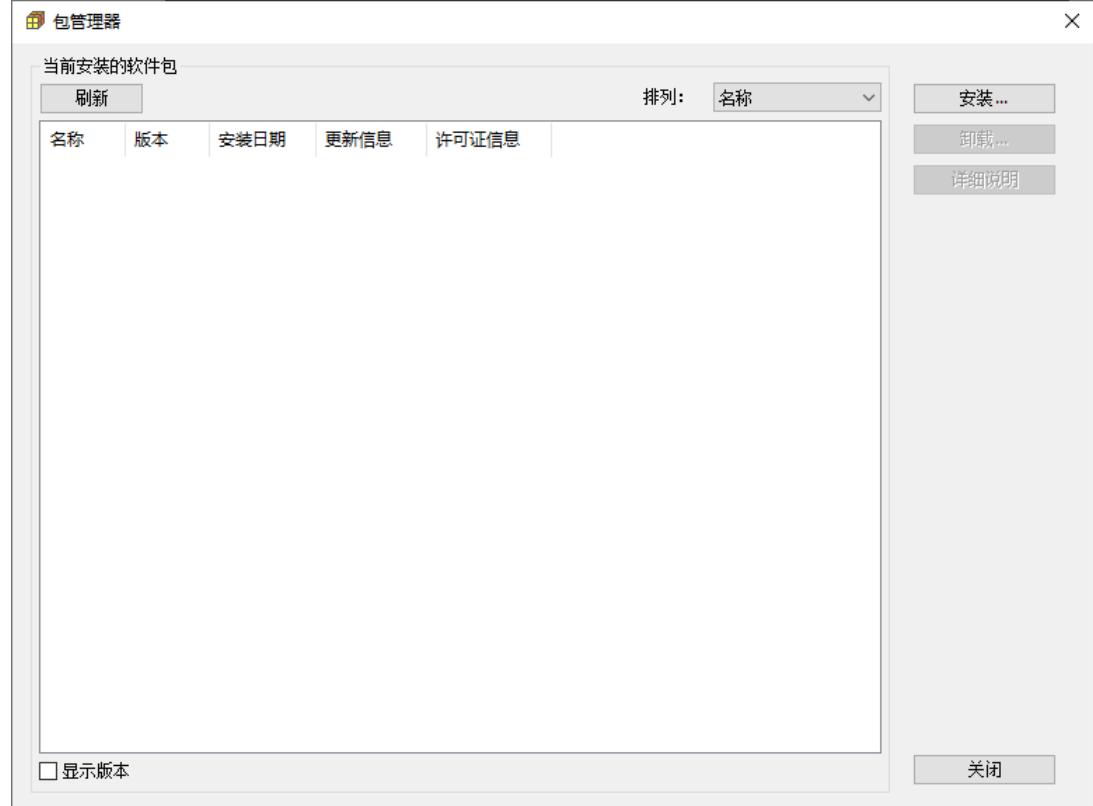
1. 安装CodeMeter运行环境。

- 在本地PC机解压InoProShop软件安装包，在“CodeMeterRuntime”文件夹双击“InstallCodeMeterRuntime.bat”批处理文件，自动识别本地PC机系统类型（32位或64位）并静默安装。
- 在“CodeMeter\_WiBu”文件夹双击“ImportLicenses.bat”批处理文件，打开“CodeMeter控制中心”界面。

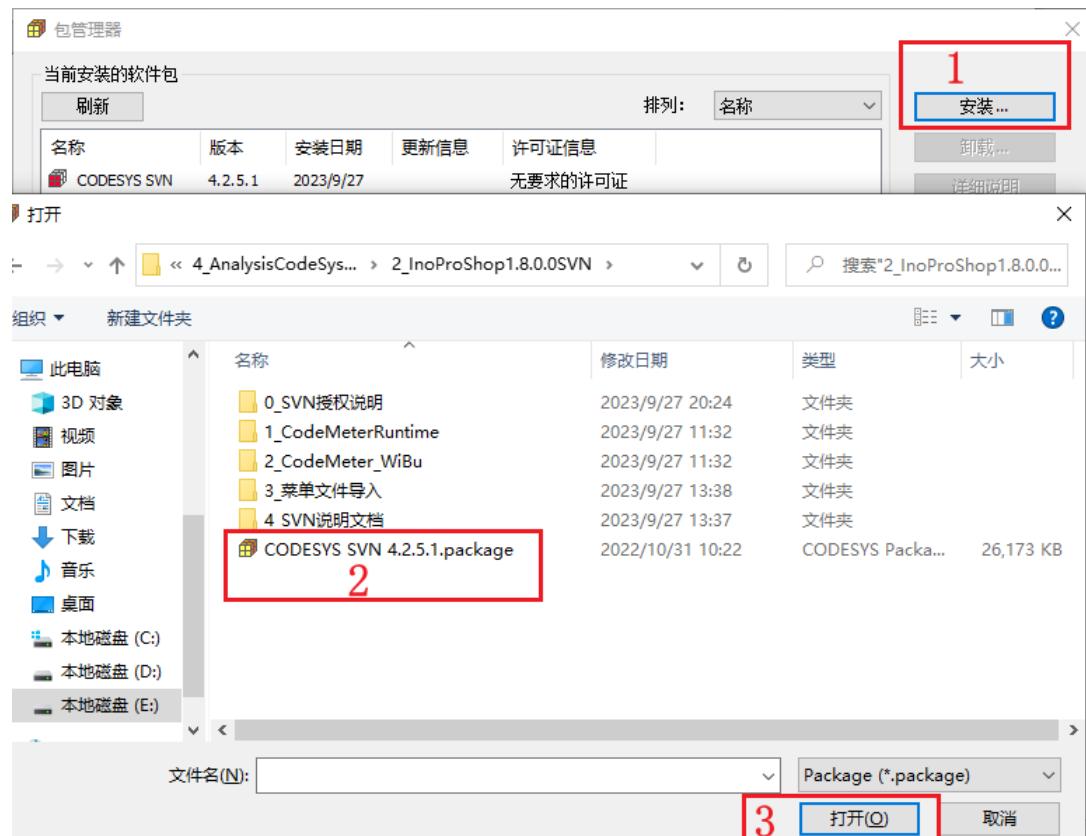


- c. 将“CodeMeter\_WiBu”文件夹中“3S-Smart\_Software\_Solutions\_Softlicenses.wbb”、“CmFirm.wbc”、“Patch\_Protection\_Only.wbb”和“Patch\_ProtectionUpdateFile.WibuCmRaU”文件拖入至上图中“许可”页签进行授权。
2. 安装SVN插件。

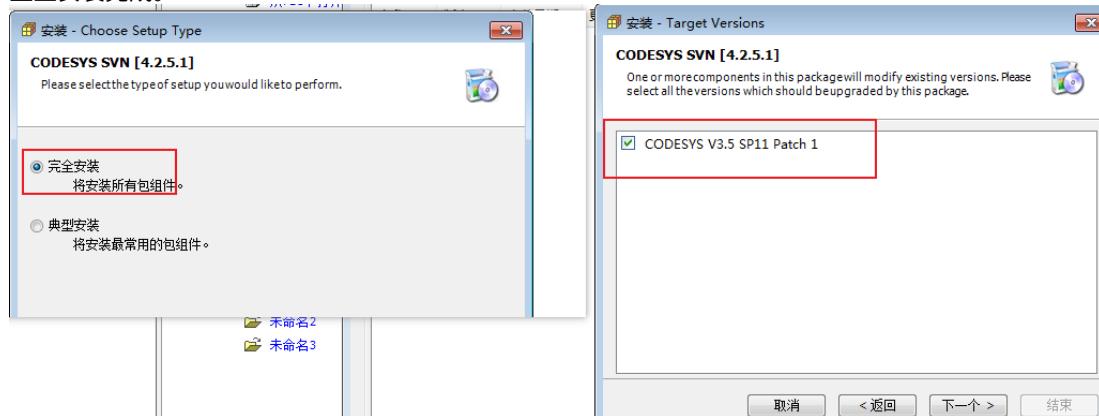
a. 在菜单栏选择“工具 > 包管理器”，打开“包管理器”界面。



b. 单击“安装”，在打开的对话框中选择InoProShop软件安装包解压后文件夹中“CODESYS SVN 4.2.5.1 package” SVN插件包，单击“打开”。



c. 在打开的向导对话框中选择“完全安装”和勾选“CODESYS V3.5 SP11 Patch 1”，按照界面向导操作直至安装完成。



安装成功后插件包相关信息将显示在“当前安装的软件包”列表中。



- d. 关闭InoProShop软件。
- e. 在InoProShop软件安装目录“..\\Inovance Control\\InoProShop1.8.0.0\\CODESYS\\Common”下双击“InstallSVN.bat”批处理文件，配置SVN插件，执行成功后显示如下图所示。

```
C:\Windows\system32\cmd.exe
D:\Program Files (x86)\Inovance Control\InoProShop1.5.6\CODESYS\Common>cd D:\Program Files (x86)\Inovance Control\InoProShop1.5.6\CODESYS\Common\
Installation and Profile Manager
Copyright © 1994-2015 by 3S-Smart Software Solutions GmbH. All rights reserved.

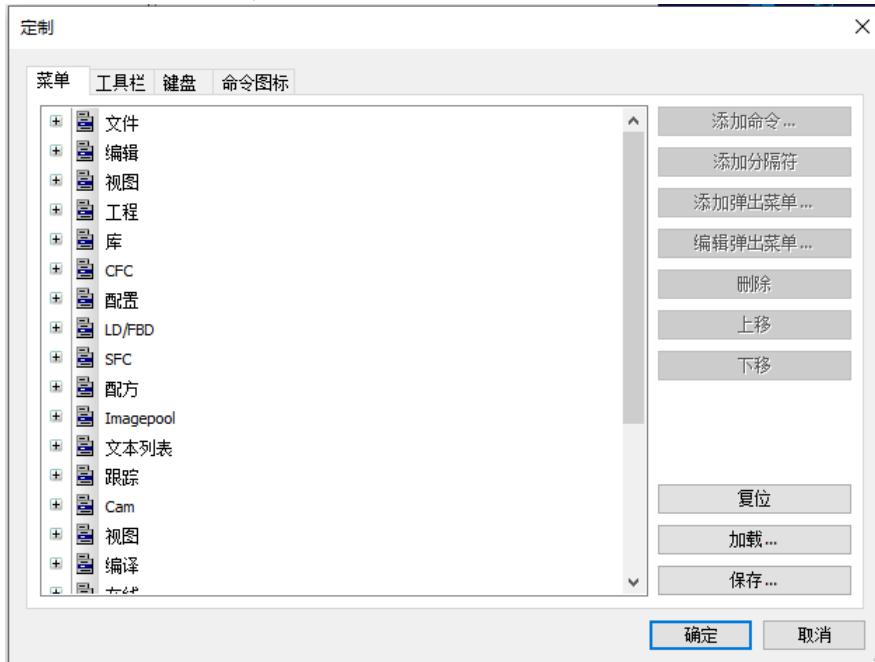
Installation and Profile Manager
Copyright © 1994-2015 by 3S-Smart Software Solutions GmbH. All rights reserved.

请按任意键继续... -
```

- f. 关闭批处理命令窗口，并重新打开InoProShop软件。

- g. 自定义菜单栏SVN功能。

1). 在菜单栏选择“工具 > 自定义”，打开“定制”界面。



2). 单击“加载...”，在打开的对话框中选择InoProShop软件安装包解压后文件夹中“菜单工具栏的SVN配置.opt.menu”菜单配置文件。



### 9.10.4 获取授权码和离线授权

在使用SVN功能之前，需要先获取授权码和进行离线授权。

#### 获取授权码

以获取“CODESYS Professional Developer Edition (Demo)”授权码为例，试用期为30天。如需更长授权天数，请购买并获取“CODESYS Professional Developer Edition”授权码。

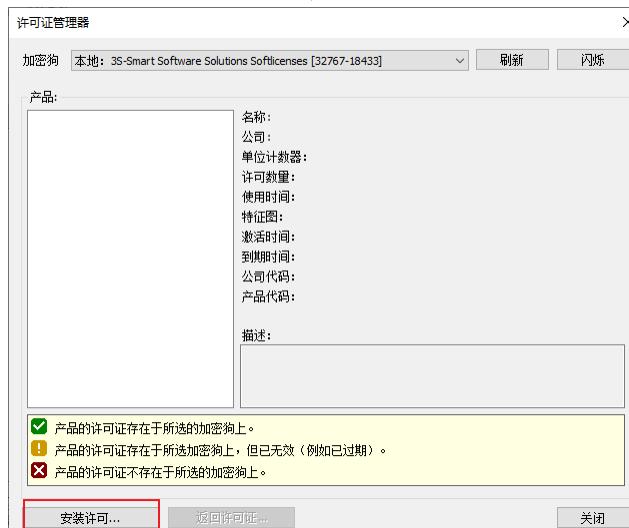
1. 登录[CODESYS国际商城](#)，创建账号。

2. 在CODESYS国际商城首页搜索框中搜索“CODESYS Professional Developer Edition (Demo)”，添加至购物车并下单。

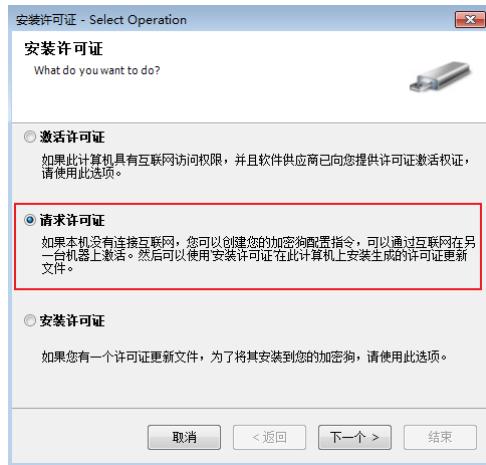
下单成功后页面将显示授权码。

## 离线授权

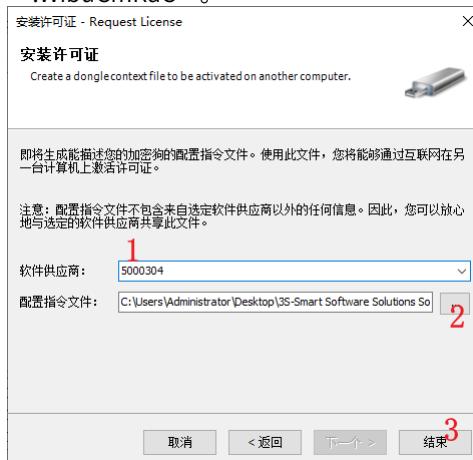
1. 在InoProShop软件菜单栏选择“工具 > 授权管理”，打开“许可证管理器”对话框。



2. 单击“安装许可”，在打开的界面中选择“请求许可证”，并单击“下一个”。



3. “软件供应商”选择“5000304”，单击“配置指令文件”后“...”选择离线授权文件生成目录，单击“结束”，生成离线请求授权文件“\*.WibuCmRaC”。



4. 登录[CODESYS官网License中心](#)，输入已获取的授权码，单击“NEXT”。  
**Welcome to CodeMeter License Central WebDepot**

Welcome to CodeMeter License Central WebDepot. You can transfer your licenses to your CmContainer using this WebDepot. Please enter your ticket and click "Next".

Ticket:

L368H-JGR47-766YP-4PY49-BK243

NEXT

5. 单击“ACTIVATE LICENSES”，激活License。  
**Licenses**

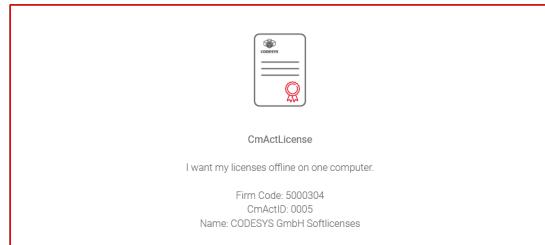
Name	Ticket	Activated On	CmContainer	Status
CODESYS Git & SVN (Demo)	L368H-JGR47-766YP-4PY49-BK243	-		Available
CODESYS Profiler (Demo)	L368H-J0R47-766YP-4PY49-BK243	-		Available
CODESYS Static Analysis (Demo)	L368H-JGR47-766YP-4PY49-BK243	-		Available
CODESYS Test Manager (Demo)	L368H-J0R47-766YP-4PY49-BK243	-		Available
CODESYS UML (Demo)	L368H-J0R47-766YP-4PY49-BK243	-		Available

ACTIVATE LICENSES

6. 选择“CmActLicense”License容器类型，在同一台电脑上进行授权。

**Available Licenses - Select the Container Type for Your Licenses**

There are different ways to activate your licenses. Please select the type of the container you want to use for the storage of your licenses.



**7. 选择“File-based license transfer” 基于文件的授权形式。**

**Available Licenses**

<input checked="" type="checkbox"/> Name	Ticket	Activated On	CmContainer	Status
<input checked="" type="checkbox"/> CODESYS Git & SVN (Demo)	FTAG9-DRNQH-3FDWQ-PB3Y3-3VPRU	-		Available
<input checked="" type="checkbox"/> CODESYS Profiler (Demo)	FTAG9-DRNQH-3FDWQ-PB3Y3-3VPRU	-		Available
<input checked="" type="checkbox"/> CODESYS Static Analysis (Demo)	FTAG9-DRNQH-3FDWQ-PB3Y3-3VPRU	-		Available
<input checked="" type="checkbox"/> CODESYS Test Manager (Demo)	FTAG9-DRNQH-3FDWQ-PB3Y3-3VPRU	-		Available
<input checked="" type="checkbox"/> CODESYS UML (Demo)	FTAG9-DRNQH-3FDWQ-PB3Y3-3VPRU	-		Available

Select CmContainer  
32767-60225 (3S-Smart Software Solutions Softlicenses)

[ACTIVATE SELECTED LICENSES NOW](#)

[File-based license transfer](#)

**8. 单击“浏览...”，选择步骤3中生成的离线请求授权文件“\*.WibuCmRaC”。**

Upload Request Download Update Upload Receipt

<input checked="" type="checkbox"/> Name	Ticket	Activated On	CmContainer	Status
<input checked="" type="checkbox"/> CODESYS Git & SVN (Demo)	PTFQC-86H76-U5V25-AR006-RG007K	-		Available
<input checked="" type="checkbox"/> CODESYS Profiler (Demo)	PTFQC-86H76-U5V25-AR006-RG007K	-		Available
<input checked="" type="checkbox"/> CODESYS Static Analysis (Demo)	PTFQC-86H76-U5V25-AR006-RG007K	-		Available
<input checked="" type="checkbox"/> CODESYS Test Manager (Demo)	PTFQC-86H76-U5V25-AR006-RG007K	-		Available
<input checked="" type="checkbox"/> CODESYS UML (Demo)	PTFQC-86H76-U5V25-AR006-RG007K	-		Available

Select an already used CmContainer  
No CmContainer found!

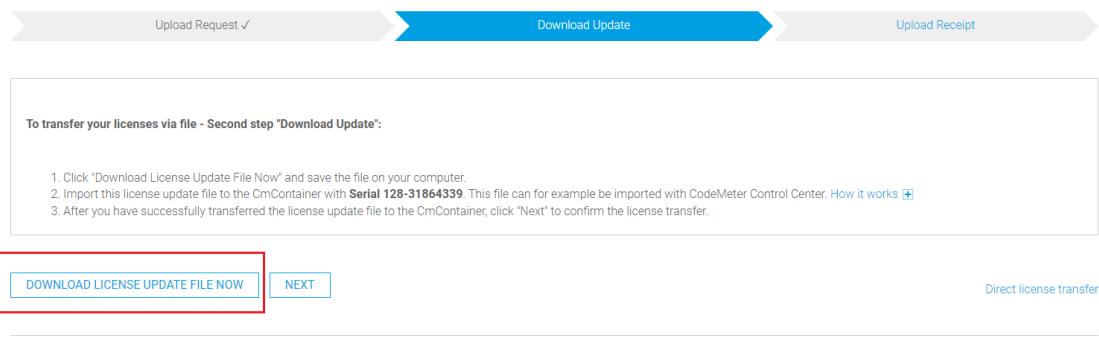
or

Pick a license request file (\*.WibuCmRaC) of another CmContainer

[START ACTIVATION NOW](#) [Direct license transfer](#)

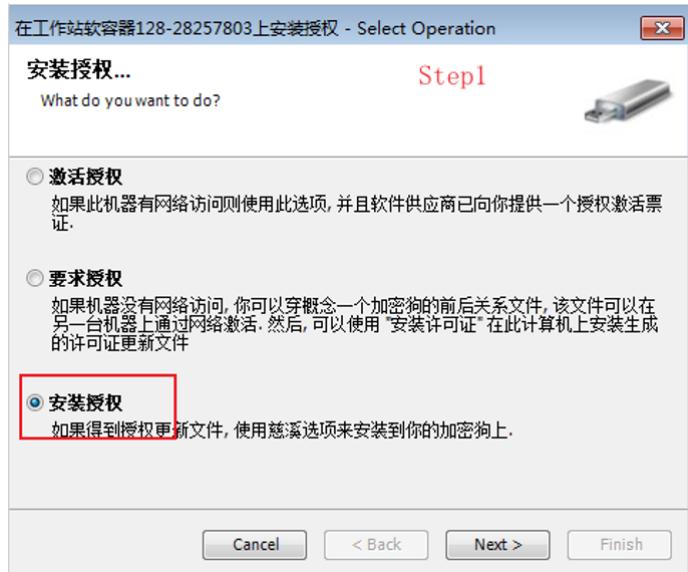
**9. 单击“DOWNLOAD LICENSE UPDATE FILE NOW”，下载生成的离线请求授权文件“\*.WibuCmRaU”，并保存至本地PC机中。**

### Download License Update File

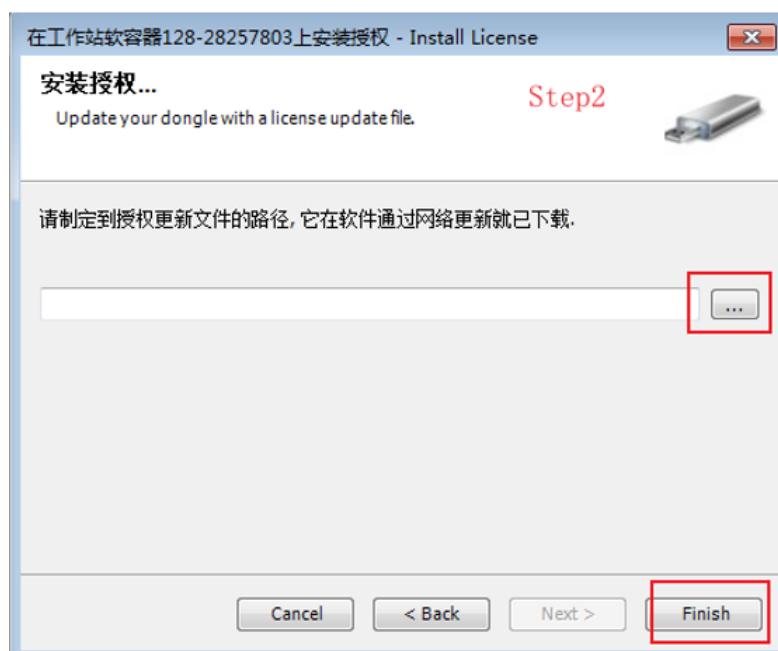


Licenses

10. 重复执行步骤1，在打开的界面中选择“安装授权”，在打开的界面中选择“请求许可证”，并单击“下一个”。

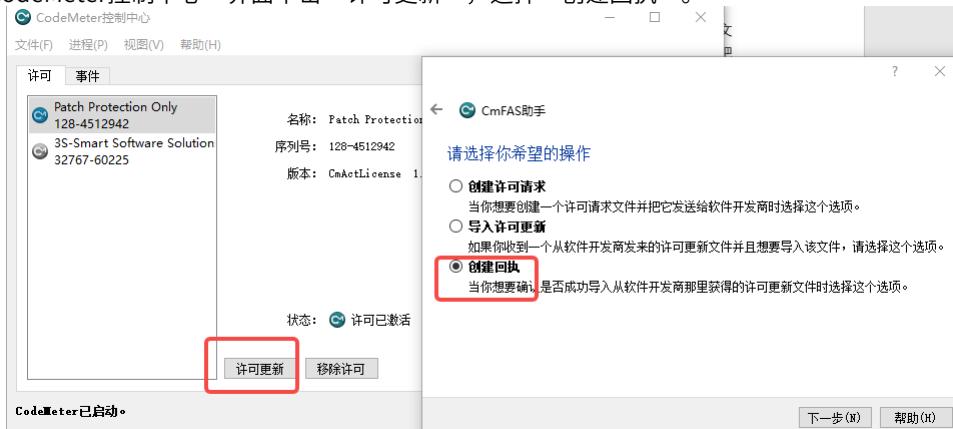


11. 单击“...”，在打开的对话框中选择步骤9生成的离线请求授权文件“\*.WibuCmRaU”，单击“结束”，完成授权。

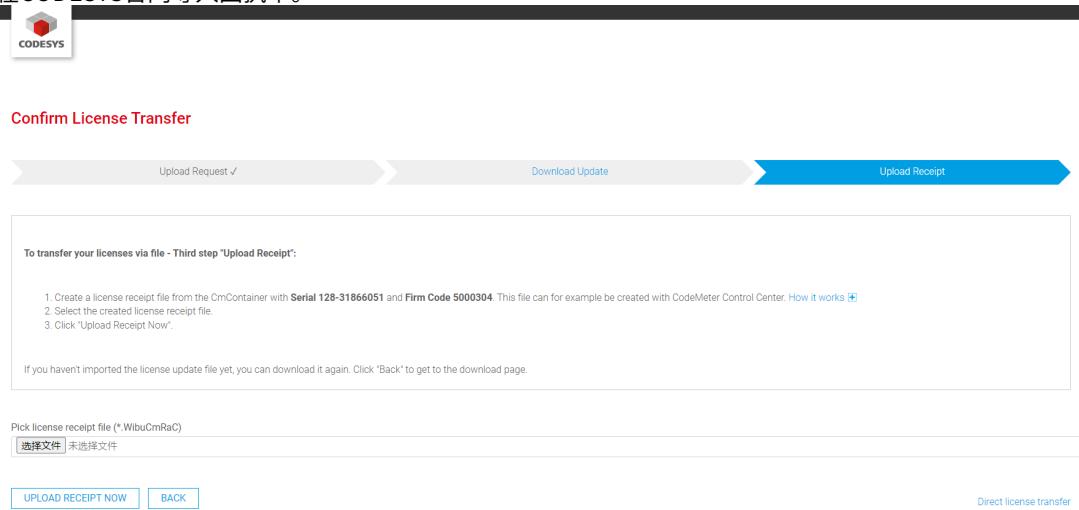


12. (可选) 可根据需要创建回执并上传回执给授权公司。

- a. 在“CodeMeter控制中心”界面单击“许可更新”，选择“创建回执”。



- b. 在CODESYS官网导入回执单。



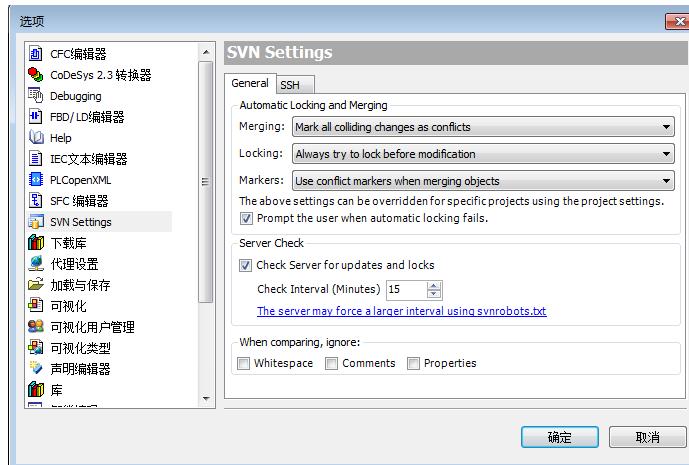
### 9.10.5 SVN操作指导

本节介绍SVN常用功能的操作，更多功能的操作具体请参见[CODESYS官网](#)下载的《CODESYS\_SVN》手册。

#### 设置SVN

- 设置选项中的SVN设置

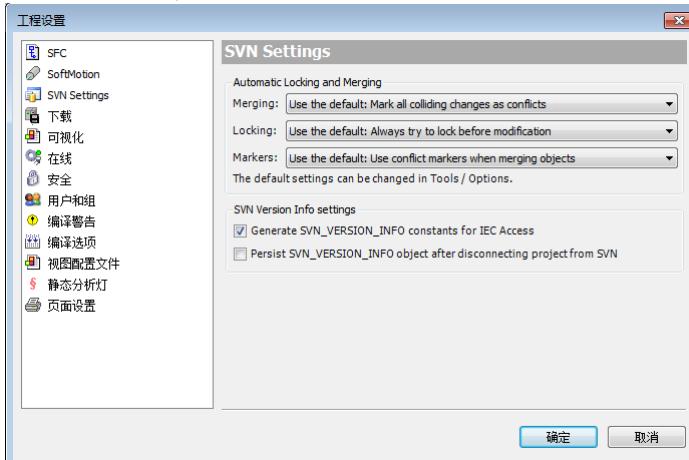
1. 在菜单栏选择“工具 > 选项”，打开“选项”对话框。



2. 单击“SVN Settings”，具体设置请参见《CODESYS\_SVN》手册。

### ● 设置工程设置中的SVN设置

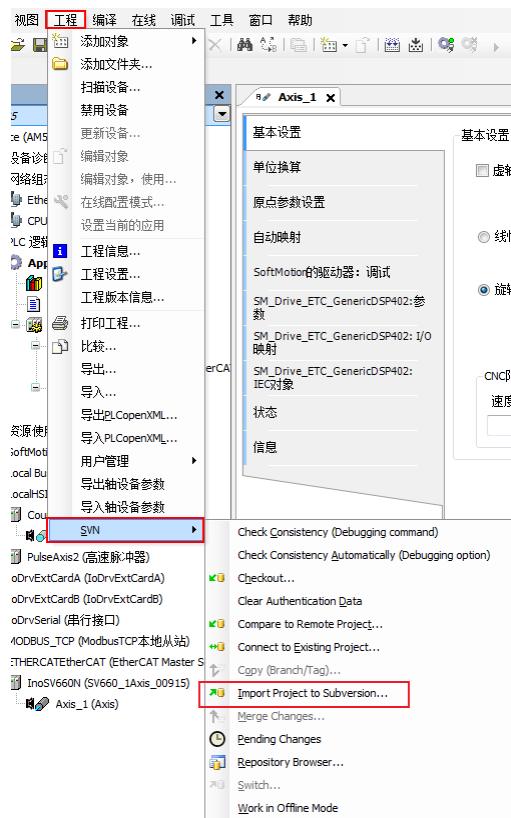
1. 在菜单栏选择“文件 > 页面设置”，打开“工程设置”对话框。



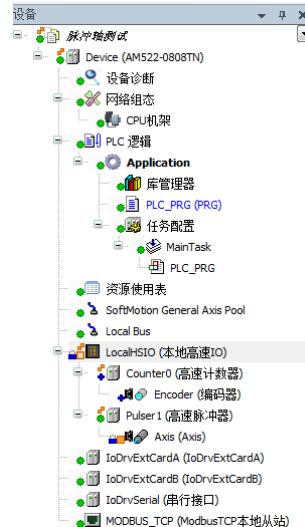
2. 单击“SVN Settings”，具体设置请参见《CODESYS\_SVN》手册。

## 导入工程至SVN库分支

在菜单栏选择“工程 > SVN > Import Project to Subversion”，导入工程至SVN库分支。

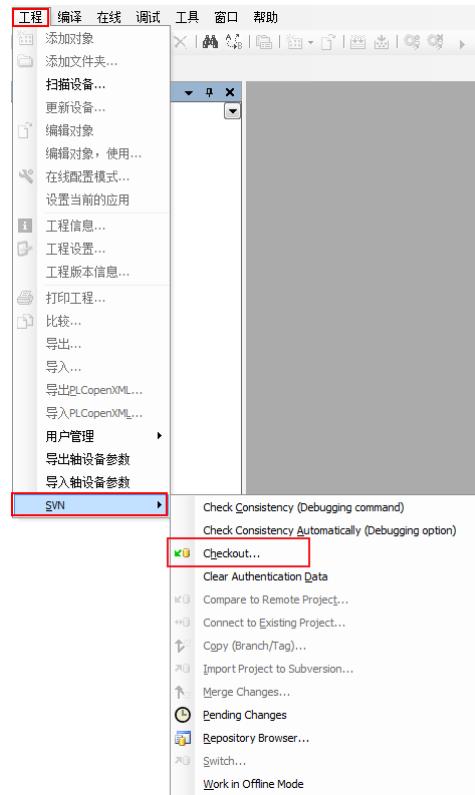


工程导入至SVN库分支后设备树将显示SVN标签。

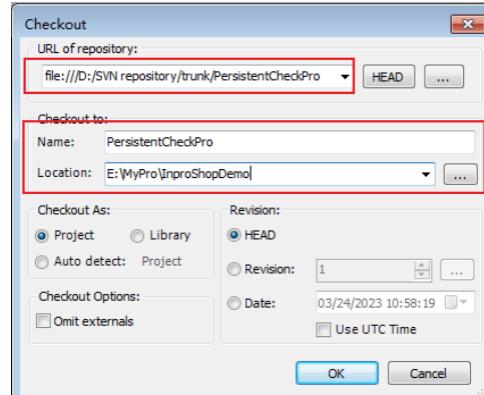


## 导出SVN库分支中的工程至本地

- 在菜单栏选择“工程 > SVN > Checkout”，导出SVN库分支中的工程至本地。

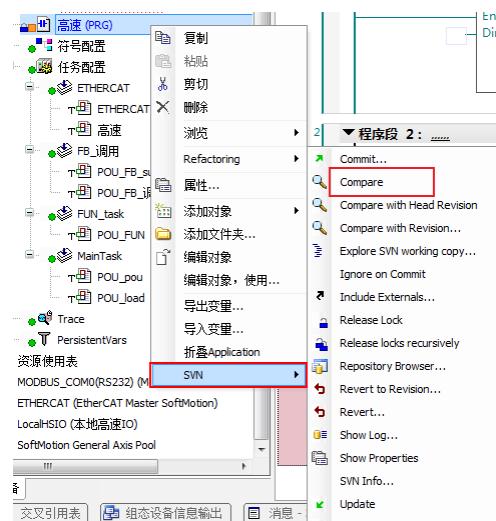


2. 在打开的对话框中设置SVN库分支路径、本地保存文件夹名称和路径，单击“OK”。



### 本地工程文件与SVN工程文件比较

在设备树中右键选择待比较的工程文件，在打开的右键菜单中选择“SVN > Compare”，比较本地工程文件与SVN工程文件之间的差异。



## 9.10.6 卸载SVN插件

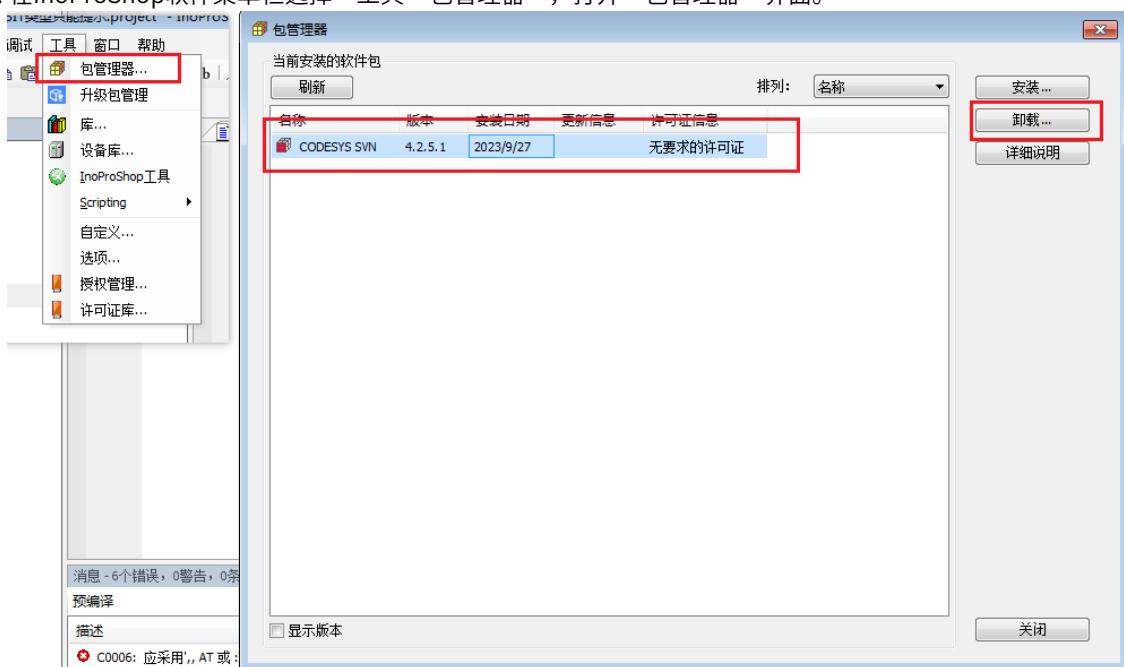
1. 关闭InoProShop软件。
2. 在InoProShop软件安装目录“..\\Inovance Control\\InoProShop1.8.0.0\\CODESYS\\Common”下双击“UninstallSVN.bat”批处理文件，卸载SVN插件，执行成功后显示如下图所示。

```
C:\Windows\system32\cmd.exe
D:\Program Files (x86)\Inovance Control\InoProShop1.8.0.0\CODESYS\Common>cd D:\Program Files (x86)\Inovance Control\InoProShop1.8.0.0\CODESYS\Common
Installation and Profile Manager
Copyright © 1994-2015 by 3S-Smart Software Solutions GmbH. All rights reserved.

Installation and Profile Manager
Copyright © 1994-2015 by 3S-Smart Software Solutions GmbH. All rights reserved.

请按任意键继续... -
```

3. 关闭批处理命令窗口，并重新打开InoProShop软件。
4. 在InoProShop软件菜单栏选择“工具 > 包管理器”，打开“包管理器”界面。



5. 选择SVN插件，单击“卸载”，根据界面引导进行卸载。

6. 卸载成功后关闭InoProShop软件。



19010334B07

由于本公司持续的产品升级造成的内容变更，恕不另行通知

版权所有 © 深圳市汇川技术股份有限公司

Copyright © Shenzhen Inovance Technology Co., Ltd.

---

**深圳市汇川技术股份有限公司**  
Shenzhen Inovance Technology Co., Ltd.

[www.inovance.com](http://www.inovance.com)

---

**地址：**深圳市龙华新区观澜街道高新技术产业园  
汇川技术总部大厦

**总机：**(0755) 2979 9595    **传真：**(0755) 2961 9897  
**客服：**4000-300124

---

**苏州汇川技术有限公司**  
Suzhou Inovance Technology Co., Ltd.  
[www.inovance.com](http://www.inovance.com)

---

**地址：**苏州市吴中区越溪友翔路16号  
**总机：**(0512) 6637 6666    **传真：**(0512) 6285 6720  
**客服：**4000-300124