

E100 系列可编程控制器

用户手册【CODESYS 软件篇】



宁波海天驱动技术有限公司

版本: V1.11

目录

第一章 CODESYS 简介	- 6 -
1.1 CODESYS 简介	- 6 -
1.2 软件安装	- 6 -
1.2.1 安装所需要的硬件要求	- 6 -
1.2.2 安装及版本管理	- 6 -
1.3 启动 CODESYS	- 7 -
1.3.1 设置管理员权限	- 7 -
1.3.2 启动编程软件	- 7 -
1.3.3 CODESYS 界面介绍	- 8 -
第二章 快速入门	- 10 -
2.1 启动 CODESYS	- 10 -
2.2 设备库安装	- 10 -
2.3 创建工程	- 12 -
2.4 固件更新	- 14 -
2.5 库的使用与安装	- 17 -
2.5.1 概述	- 17 -
2.5.2 库文件的管理	- 17 -
2.5.3 库文件的安装与调用	- 19 -
2.6 软件左侧栏介绍	- 21 -
2.7 PLC 用户程序编写	- 22 -
2.8 用户程序变量与端口的关联配置	- 24 -
2.9 任务配置	- 26 -
2.10 CodeSys 连接设置	- 29 -
2.10.1 网关配置	- 29 -
2.10.2 以太网通讯	- 29 -
2.10.3 控制器 IP 设置	- 31 -
2.10.4 USB 通讯	- 33 -
2.11 程序下载/读取	- 34 -
2.11.1 编译	- 34 -
2.11.2 登录及下载	- 34 -

2.12 在线监控	35 -
2.12.1 POU 在线监控	35 -
2.12.2 写入/强制变量	36 -
2.12.3 曲线采样	37 -
2.13 PLC 本机高速 IO	38 -
2.13.1 普通 IO	39 -
2.13.2 高速输入	39 -
2.13.3 高速输出	40 -
2.14 用 CODESYS 编写一个跑马灯样例工程	41 -
第三章 通讯配置	44 -
3.1 ModBus RTU 通讯设置	44 -
3.1.1 Modbus 接线配置	44 -
3.1.2 Modbus 主站配置	44 -
3.1.3 Modbus 从站配置	47 -
3.1.4 Modbus 从站映射地址	50 -
3.2 ModBus TCP 通讯设置	50 -
3.2.1 Modbus 接线配置	50 -
3.2.2 Modbus 主站配置	51 -
3.2.3 Modbus 从站配置	52 -
3.2.4 Modbus 从站映射地址	56 -
3.2.5 PLC 级联通讯 (Elink)	56 -
3.3 EtherCAT 主站通讯设置	58 -
3.3.1 E112C 与 Hi 驱动器通信举例	58 -
3.3.2 EtherCAT 单轴通讯控制	61 -
3.3.3 导入第三方从站设备	68 -
3.4 CANopen 主站通讯设置	72 -
3.4.1 E112C 与 Hi 驱动器通信举例	72 -
3.4.2 CANopen 参数配置	74 -
第四章 扩展模块	81 -
4.1 扩展模块类型	81 -
4.2 扩展模块组态	81 -
4.3 扩展模块的配置与映射	82 -

4.3.1 EM-AI04 模块.....	- 82 -
4.3.2 EM-AO04 模块.....	- 83 -
4.3.3 EM-DI16 模块.....	- 84 -
4.3.4 EM-DO16N 模块.....	- 84 -
4.3.5 EM-HC0401 模块.....	- 85 -
4.4 选配项.....	- 86 -
4.4.1 选配项类型与组态.....	- 86 -
4.4.2 选配项参数设置.....	- 87 -
第五章 公共元素及变量.....	- 89 -
5.1 公共元素.....	- 89 -
5.1.1 分界符.....	- 89 -
5.1.2 关键字.....	- 90 -
5.1.3 常数.....	- 91 -
5.1.4 句法颜色.....	- 93 -
5.1.5 空格和注释.....	- 93 -
5.2 变量的表示和声明.....	- 95 -
5.2.1 变量.....	- 95 -
5.2.2 标识符.....	- 95 -
5.2.3 变量声明.....	- 95 -
5.3 数据类型.....	- 96 -
5.3.1 标准数据类型.....	- 96 -
5.3.2 标准扩展数据类型.....	- 98 -
5.3.3 自定义数据类型.....	- 99 -
5.4 变量的类型和初始化.....	- 101 -
5.4.1 变量的类型.....	- 101 -
5.4.2 变量的初始化.....	- 102 -
第六章 常用指令.....	- 103 -
6.1 运动功能块.....	- 103 -
6.1.1 概览.....	- 103 -
6.1.2 状态图.....	- 104 -
6.1.3 功能块输入输出规则.....	- 105 -
6.1.4 BUFFER_MODE.....	- 106 -

6.1.5 MC_LIB 数据类型说明	- 107 -
6.1.6 系统功能块	- 115 -
6.1.7 单轴运动功能块	- 116 -
6.1.8 轴组(主/从轴)运动功能块	- 127 -
6.1.9 运动功能块错误说明	- 149 -
6.2 通讯功能块	- 154 -
6.2.1 概览	- 154 -
6.2.2 COM_LIB 数据类型说明	- 155 -
6.2.3 CANopen 通讯功能块	- 158 -
6.2.4 EtherCAT 通讯功能块	- 164 -
6.2.5 Modbus 通讯功能块	- 169 -
6.2.6 Module 通讯功能块	- 173 -
6.2.7 Socket 通讯功能块	- 181 -
6.2.8 通讯功能块错误说明	- 184 -

第一章 CODESYS 简介

1.1 CODESYS 简介

CODESYS 由德国 Smart Software Solution GmbH 公司开发，通常以 3S 公司简称，该公司的总部位于德国巴伐利亚肯不腾市。

CODESYS (Controller Development System) 是可编程逻辑控制 PLC 的完整开发环境，在 PLC 程序员编程时，它强大的编程语言提供了一个简单的方法，系统的编辑器和调试器的功能建立在高级编程语言的基础上（如 Visual C++）。

CODESYS 提供了所有 IEC61131-3 所定义的 5 中编程余元：结构化文本（ST）、顺序功能块（SFC）、功能块图（FBD）、梯形图（LD）和指令表（IL），此外还支持连续功能图（CFC）的编程语言。

1.2 软件安装

CODESYS 编程软件是标准的 Windows 界面，支持编程，调试及配置。

1.2.1 安装所需要的硬件要求

由于 CODESYS V3.5 软件比较大，需要处理的数据也比较多，因此对 PC 的硬件配置及系统环境有一定的要求。其要求最低配及推荐配置见表 1.1。

表 1.1 软件安装最低配置及推荐配置

描述	最低配置	推荐配置
操作系统	Windows 2000 (Windows Vista/Windows 7/8/10/11)	Windows 7/8/10/11 (32/64 位)
内存	512MB	4GB
硬盘空间	200MB	2GB
处理器	Pentium V, Centrino>1.8GHz, Pentium M>1.0GHz	Pentium V, Centrino>3.0GHz, Pentium M>1.5GHz

1.2.2 安装及版本管理

1. 安装

直接双击运行 Setup_CODESYSV35SP15.exe 安装文件即可进入安装，整个安装过程中安装助手都会引导用户进行安装。

在软件安装之前，程序会对安装环境进行检测，必须要有 Microsoft Visual C++ 及 Microsoft .NET Framework 4.6 的安装环境，如果之前没有安装过，该应用程序会自动安装这部分软件。安装前需同意遵守 3S 公司的软件使用规范。

2.版本管理

在 CODESYS 可以同时安装一个组件的多个不同版本，并可以组合使用，程序编辑器也可以安装与使用多个版本，而且无需更新整个版本就可以新增独立的功能。

1.3 启动 CODESYS

1.3.1 设置管理员权限

在 Windows7 系统下一管理员权限打开软件，在默认安装路径下找到 CODESYS.EXE 文件，选中该文件后鼠标单击右键，选择属性，然后在“CODESYS.exe Properties”对话框的“Compatibility”的选项卡中，选中“RUN this program as an administrator”复选框，单击“OK”按钮确认，确认后，每次运行 CODESYS，系统就会默认以管理员权限进入。

1.3.2 启动编程软件

在“开始”菜单中依次选择程序→3S CODESYS→CODESYS→CODESYS V3.5，或直接双击桌面图标  启动，CODESYS 启动后需要安装 E112C 控制器设备描述文件（E112C_DevDesc_1.1.8.0.xml）安装成功后即可在 CODESYS 中使用 E100 系列设备。具体操作如下：

选择菜单栏中的“工具”→“设备存储库”打开如图 1.1 所示界面，在对话框中点击“安装”选择所需要的设备 E112C_DevDesc_1.1.8.0.xml 设备描述文件，点击“打开”按钮确认后，新设备即添加到“设备存储库”中的设备目录。以上操作执行完成后可以通过菜单栏“工具”→“设备库”打开 PLC 查看设备库是否安装正确如图 1.2 所示。

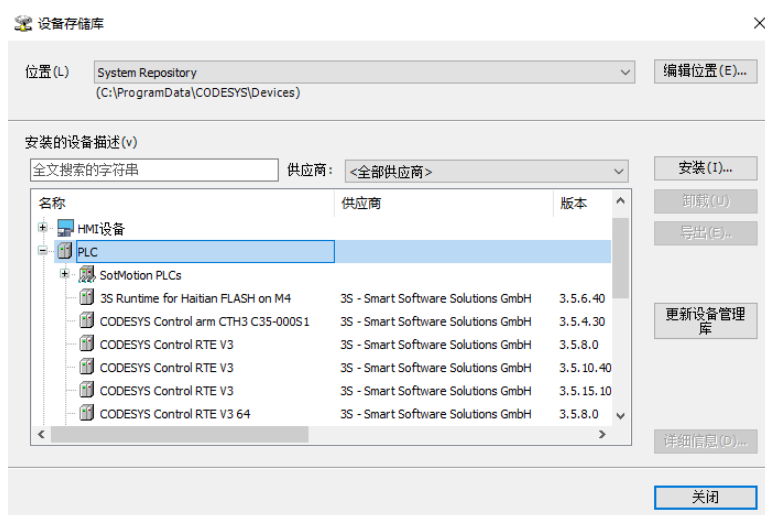


图 1.1 设备安装

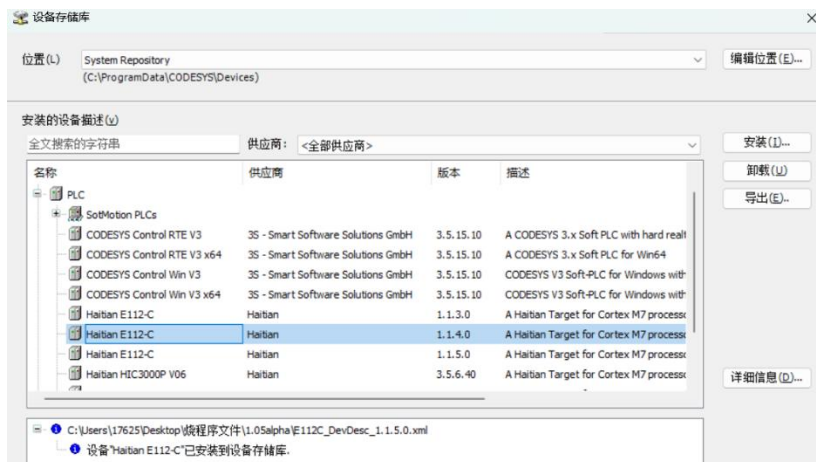
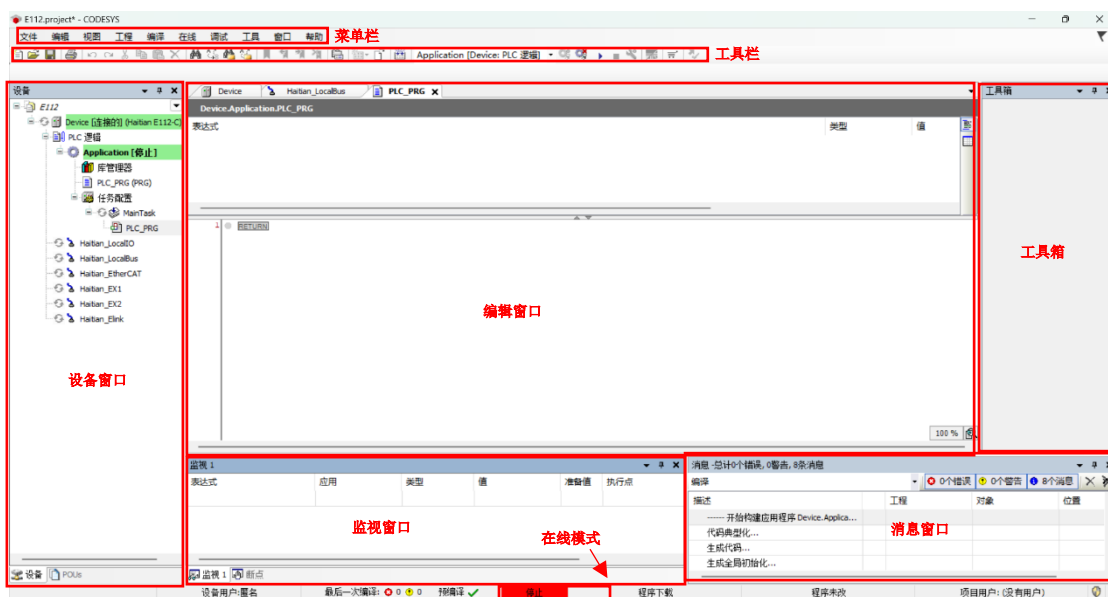


图 1.2 设备安装完成

1.3.3 CODESYS 界面介绍

如图 1.1 所示，为 CODESYS V3.5 SP15 的开发界面，标准的组件主要有菜单栏、工具栏、编辑窗口、监视窗口、消息窗口和在线模式等。



1. 菜单栏

在 CODESYS 中，菜单栏是使用最为频繁的操作选项，项目的新建及保存，程序的编译，登录及下载，调试时的断点设置及轻质写入等操作都需要通过菜单栏里面的功能来实现。

2. 工具栏

通过单击图标用户可以更快速的相应选择的命令。可以选择的图标将自动与激活窗口相适应。仅当鼠标在图标上单击然后释放时，才能够执行命令，如果用户将鼠标指针短时间内停留在工具栏上的一个图标时，则会在工具上提示中显示该图标的名称。

3. 编辑窗口

编辑窗口用于在相应的编辑器中创建特定的对象。一般所说的语言编辑器（如 ST-编辑

器。CFC-编辑器）是指余元编辑器和声明编辑器的组合，通常在下部为语言编辑器，上部为声明编辑器，在其他编辑器中还可以提供对话框。

4. 设备窗口

以树结构管理工程中的资源对象。在项目中，数据分层结构中以对象的形式进行保存。

5. 监视窗口

显示一个 POU 的监视视图，当程序登录后，可以用来监视 POU 中任意的表达式

6. 消息窗口

消息窗口显示预编译、编译、生成器、下载和程序检测等信息。如果用鼠标双击消息窗口内的一条消息或者按右键选择“跳转到代码处”，编辑器就会打开所选择行的对象。通过右键快捷菜单中的选项“后一个消息”（<F4>键）及“前一个消息”（<Shift+F4>）组合键，可以在各个错误消息之间快速跳转。消息窗口显示是滚动的。

7. 在线模式

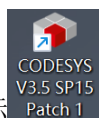
在线模式具体状态信息见表 1.2

表 1.2 通信状态具体信息

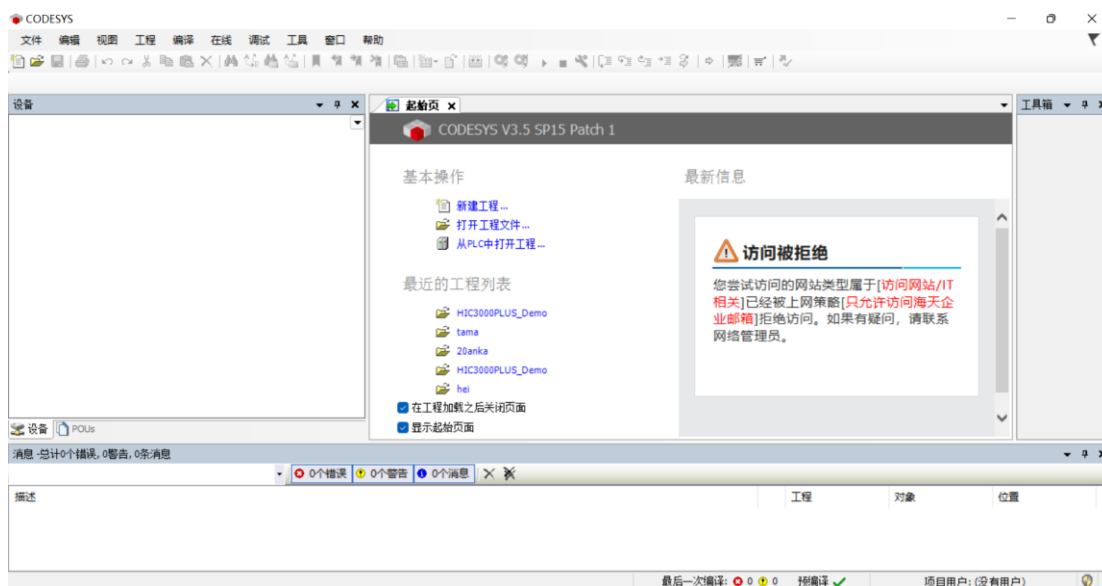
状态	描述
运行	程序正在运行
停止	程序已经停止
仿真	CODESYS 目前正处于无硬件仿真阶段
程序下载	程序已经下载到设备
程序未变	设备上的程序与编程系统中的相符
修改程序（在线）	设备上的程序与变成系统中的不同，需要在线修改

第二章 快速入门

2.1 启动 CODESYS



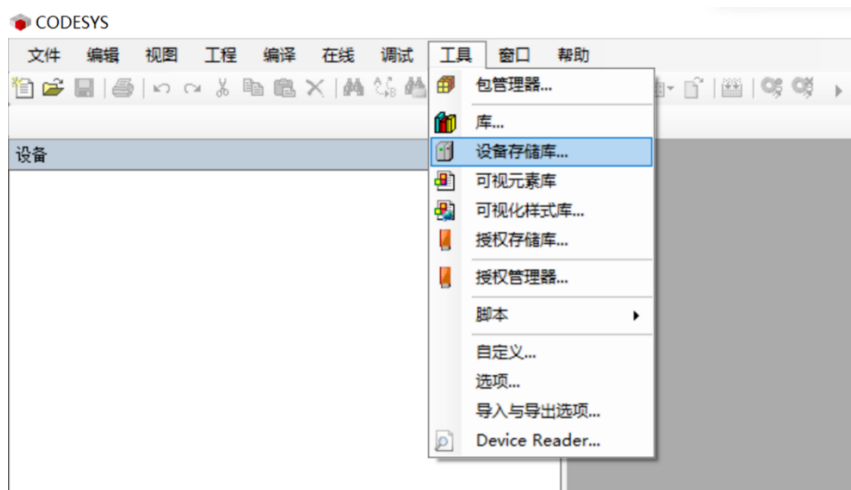
双击桌面 CODESYS 图标，打开后起始界面如下：



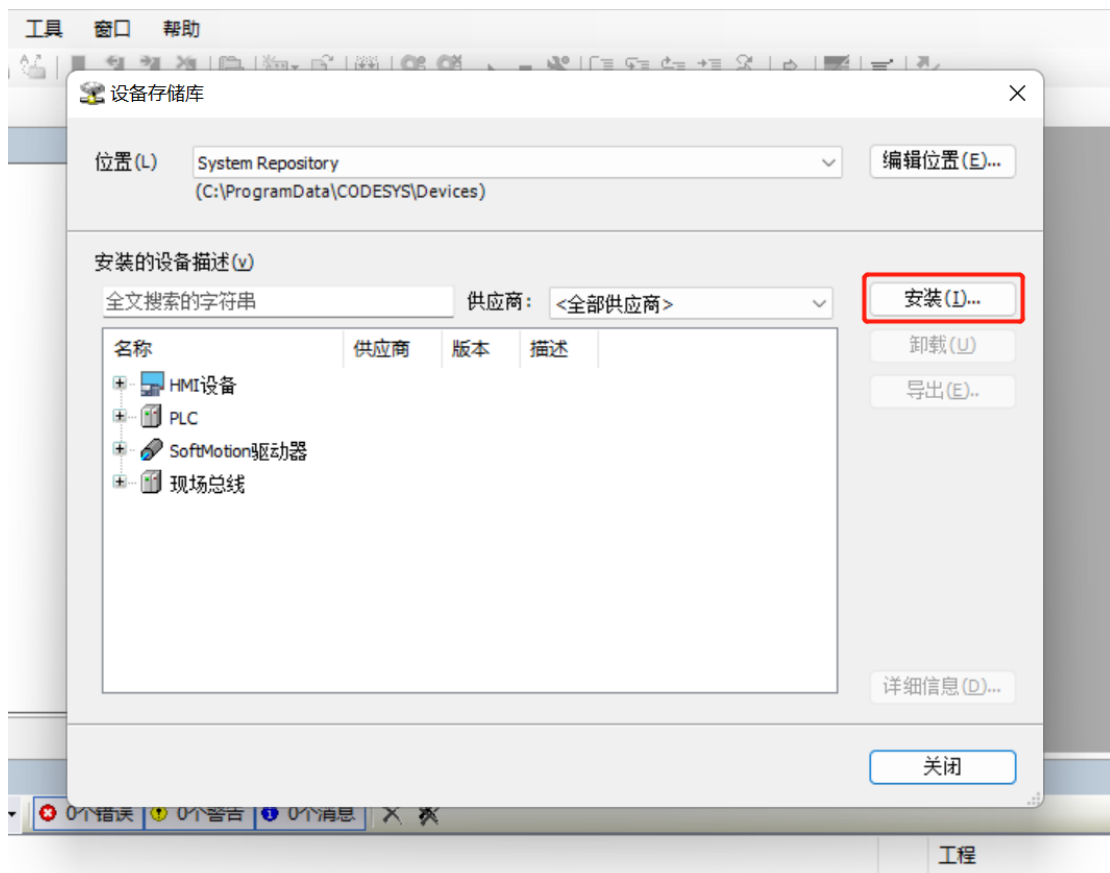
2.2 设备库安装

设备位于软件模型的最上层，它代表了一个具体的目标，即硬件对象。因此为了在 CODESYS 中使用 E112C 设备，我们首先须要在 CODESYS 软件中安装相应设备描述文件“E112C_DevDesc_1.1.8.0.xml”，该设备描述文件确定了使用设备的相关配置，可编程性和与其他设备的互联性。

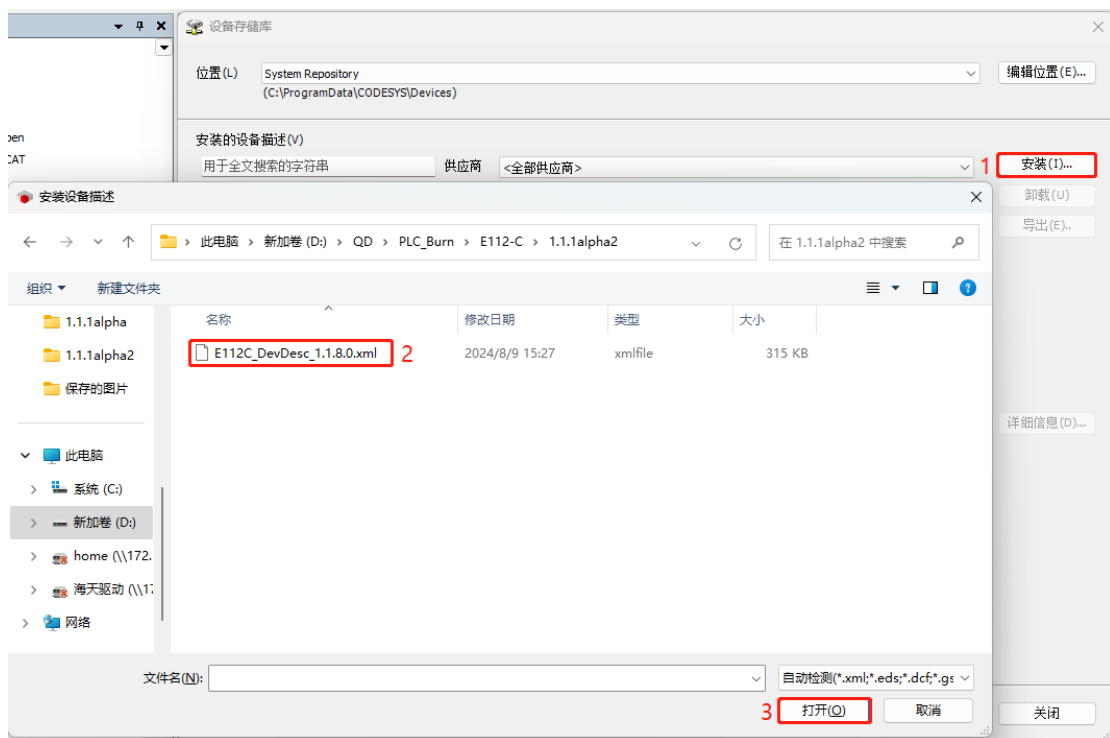
(1) 点击菜单栏中“工具”，随后点击“设备存储库”，如下图所示：



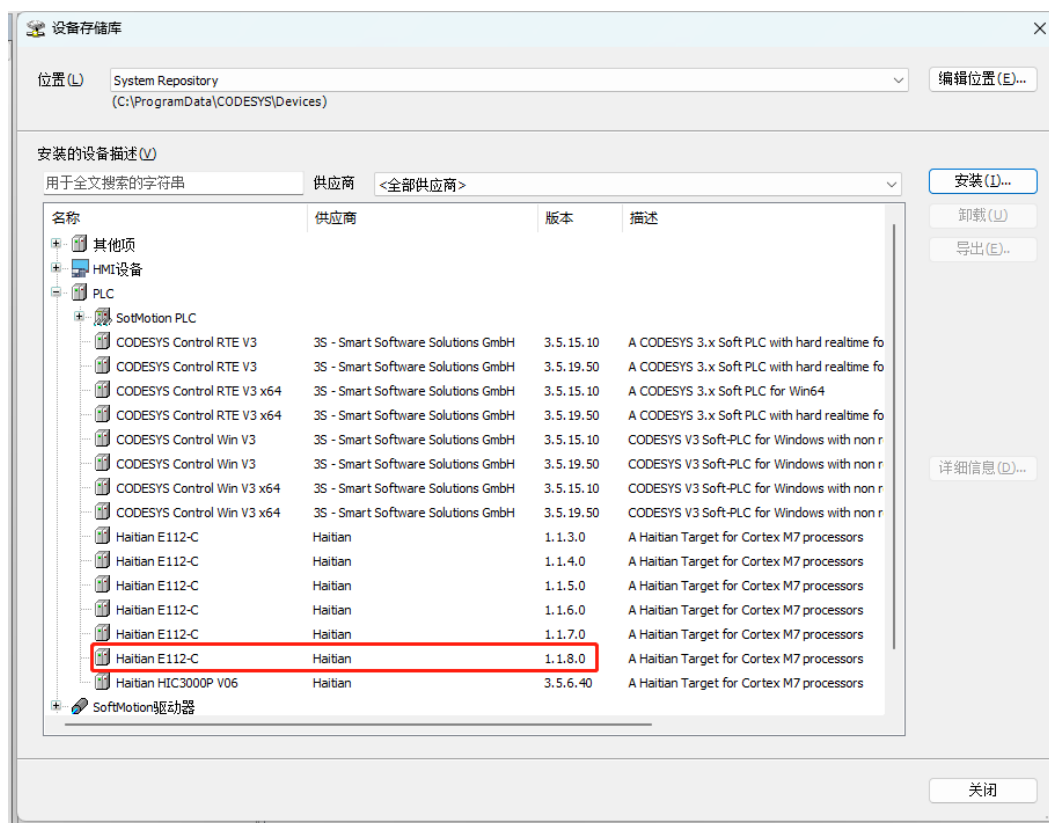
(2) 点击“安装”。



(3) 选择所需的设备 E112C_DevDesc_1.1.8.0.xml 设备描述文件后，点击“打开”完成设备描述文件安装操作。



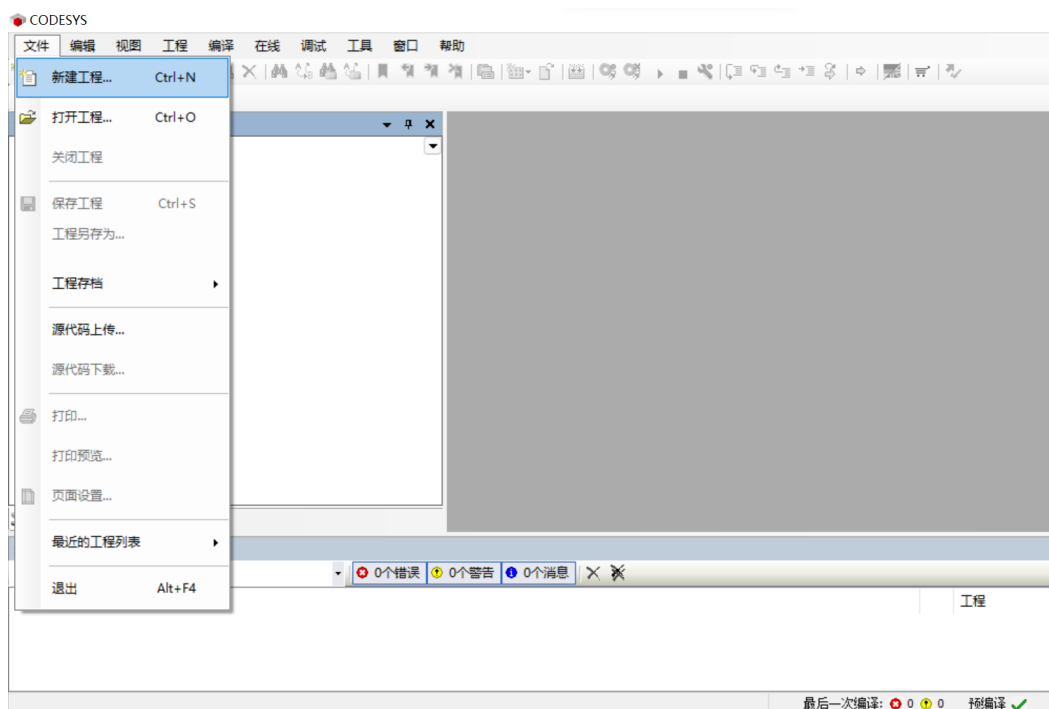
(4) 可重新点击“工具”→“设备存储库”后在 PLC 栏下找到相应设备。



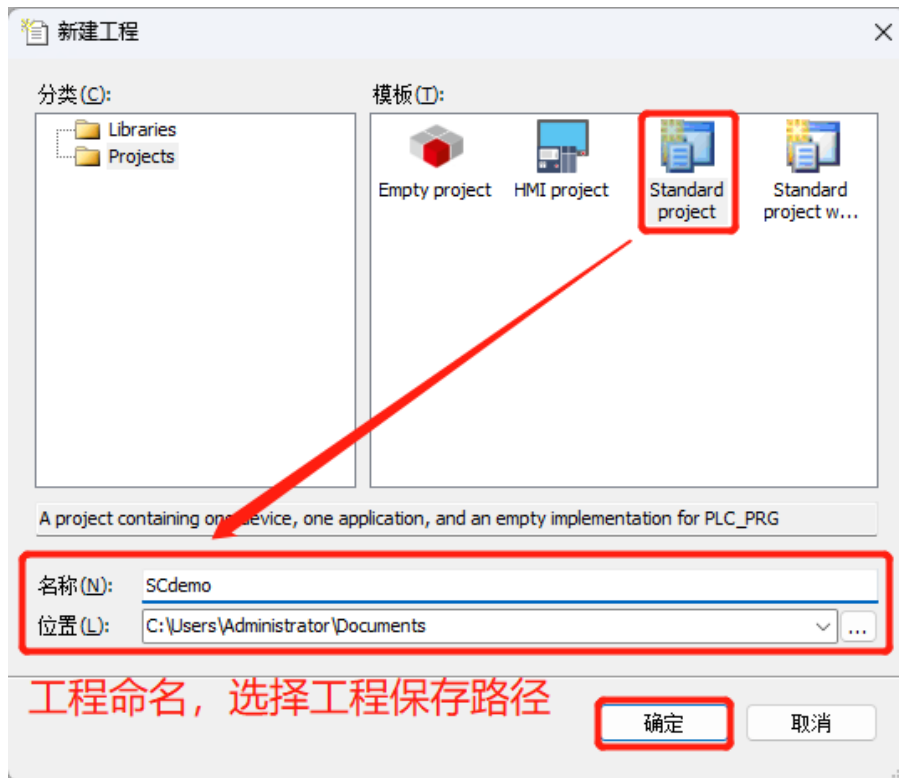
2.3 创建工程

设备添加完毕后，需要建立一个工程文件在其中完成程序编写等内容。

(1) 点击菜单栏中“文件”，随后点击“新建工程”。

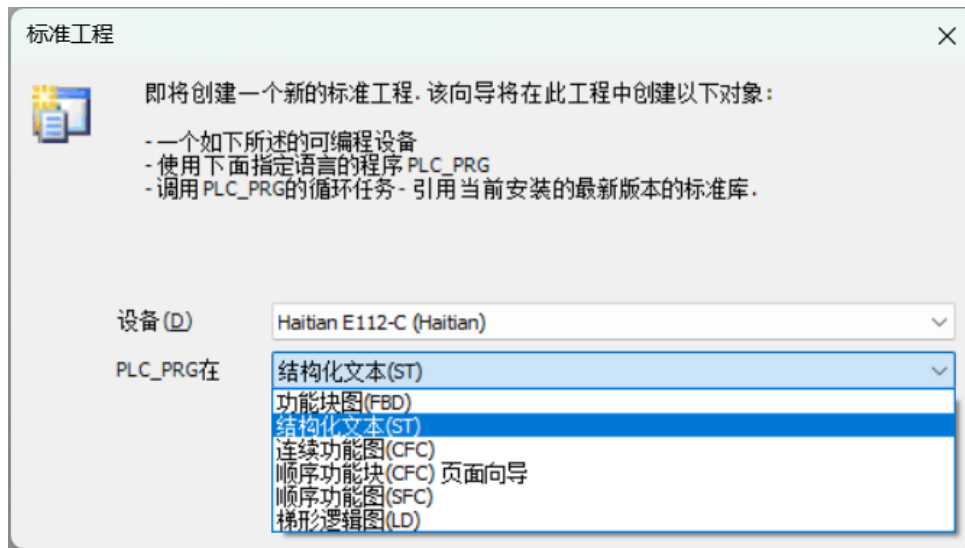


(2) 选择“标准工程” (“Standard project”) 后，为新建工程命名并选择保存路径，完成上述操作后点击“确定”。

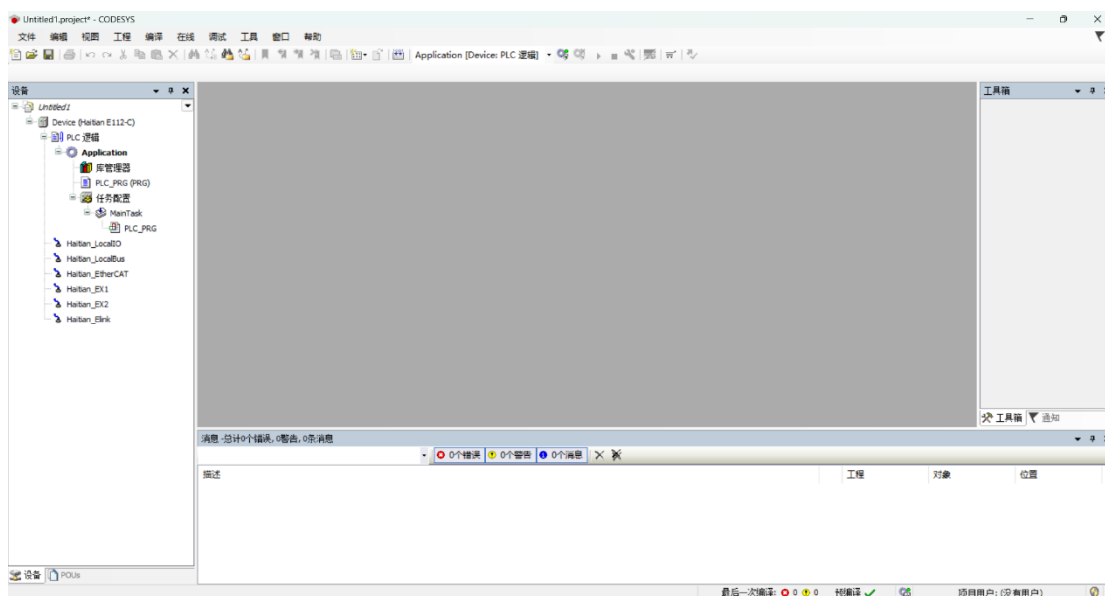


(3) 选择 2.2 中安装的 Haitian E112-C 设备，编程语言根据个人需求自行选择，本手册选用 ST 即结构化文本语言，完成后点击“确定”。



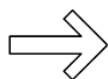
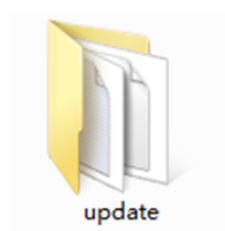


(4) 完成上述操作后界面如下：



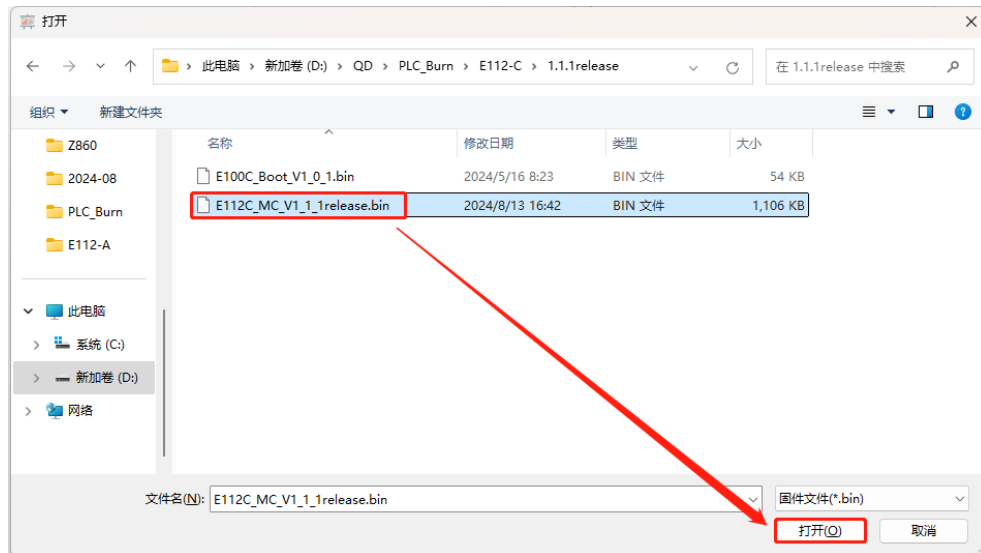
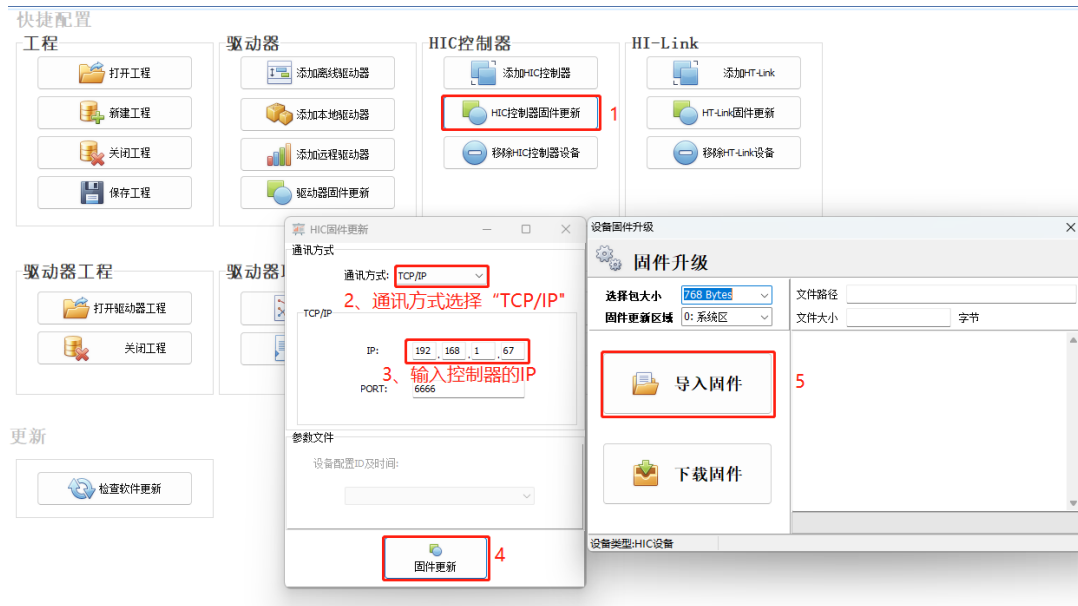
2.4 固件更新

- (1) 在 SD 卡一级目录下新建名称为“update”的文件夹。
- (2) 在上述“update”文件夹中，新建名为“config.txt”的 txt 文件，参考下图。

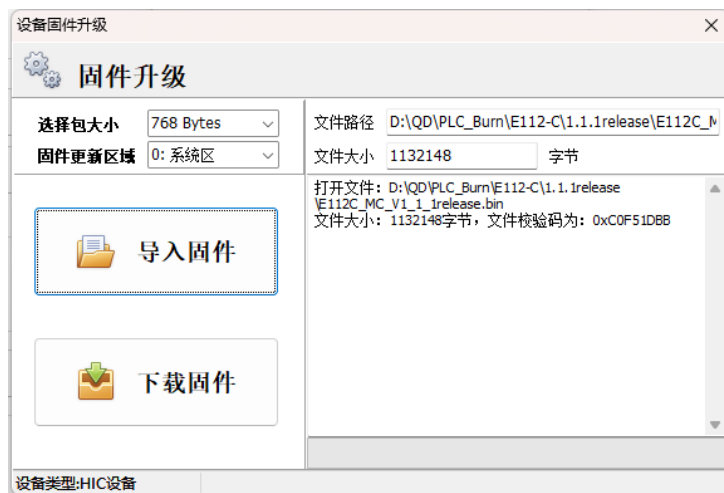


- (3) E100 系列控制器断电后,将 SD 卡插入卡槽, 控制器重新上电。
- (4) 打开 HIMPV3 上位机软件, 选择“HIC 控制器固件更新”, 输入控制器的 IP 地址, 导入

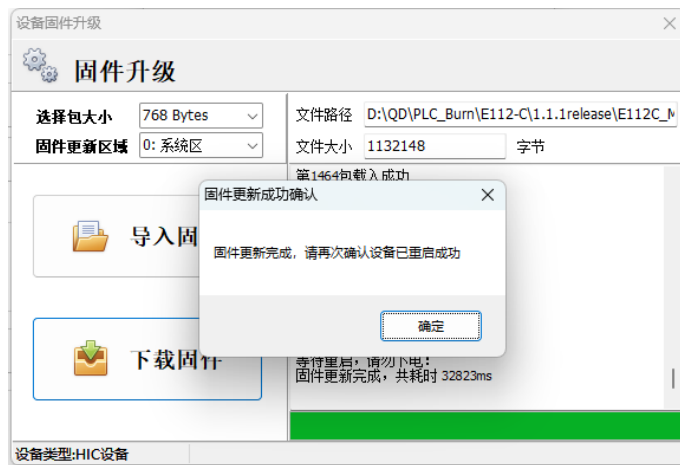
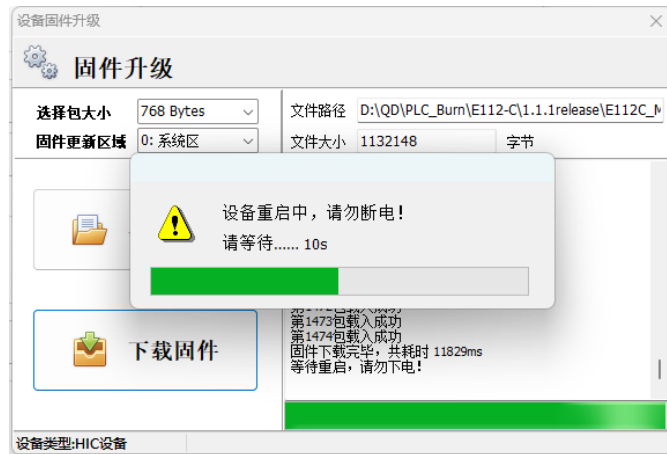
需要更新的固件



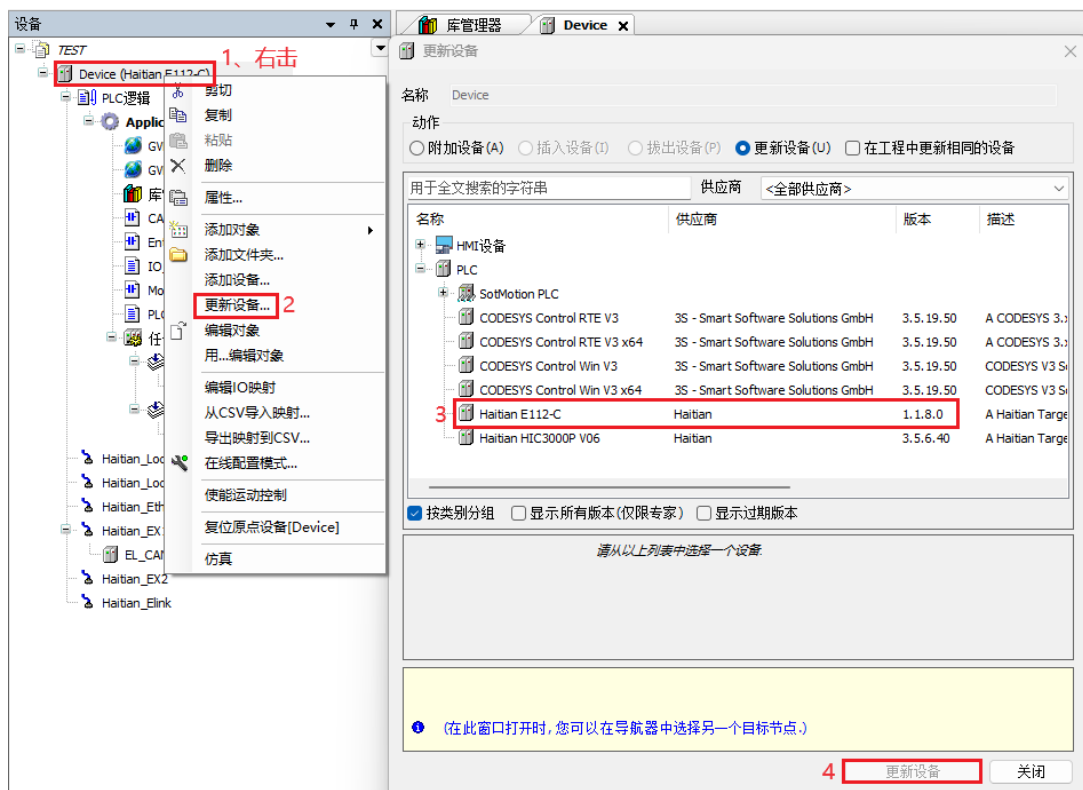
(5) 下载固件（注意：升级过程中不允许断电，升级完成后控制器会自动重启）



(6) 设备重启过程中，PR、RN、SF 三灯亮起，更新完成后，PR 常亮，RN 闪烁，SF 灭。



(7) 完成固件升级后，根据“2.2 设备库安装”将新的设备描述文件进行导入，然后按照图示更新设备。



2.5 库的使用与安装

2.5.1 概述

库文件用于存放程序中可多次使用的程序组织单元（POU）。这些 POU 可以从已有的项目中复制到库中，也可以由用户通过新建库项目自定义库。库文件从功能上划分，可以分为系统库文件、应用库文件、厂商自定义库文件及 IEC 动作库。CODESYS V3.5 中默认的库文件是“.library*”。

（1）系统库文件

系统库文件是一个支持 CODESYS 软件系统的文件，它包括对软件和语法编写的支持，以及标准 I/O 的支持。通常该库文件会在软件启动后自动导入到控制器中，不需要手动添加。

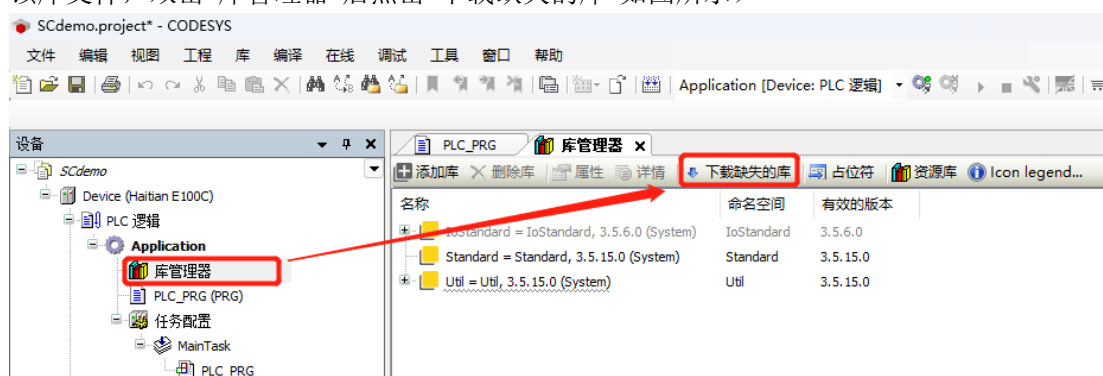
（2）应用库文件

支持基本应用的文件库如下。

1) Util: 包含各种数学运算、位操作指令等功能。

2) Standard: 包括定时器，计数器，边沿检测及双稳态触发器等函数功能块。

库 Standard 是作为一台 PLC 必备的功能，因此，在打开 CODESYS 后，会自动调用该库文件。其他库需要按照客户实际需求来进行添加。（客户首次使用 CODESYS 可能未添加该库文件，双击“库管理器”后点击“下载缺失的库”如图所示）

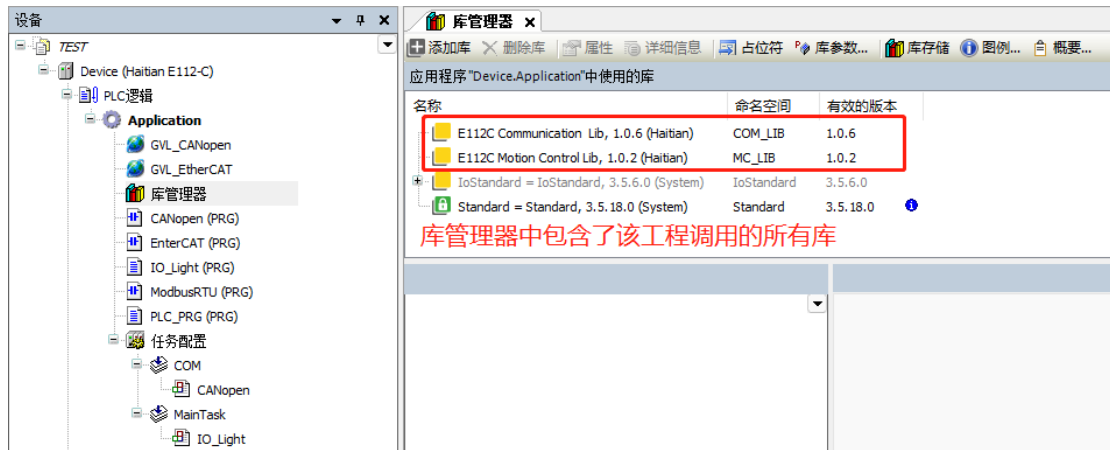


（3）厂商自定义库文件

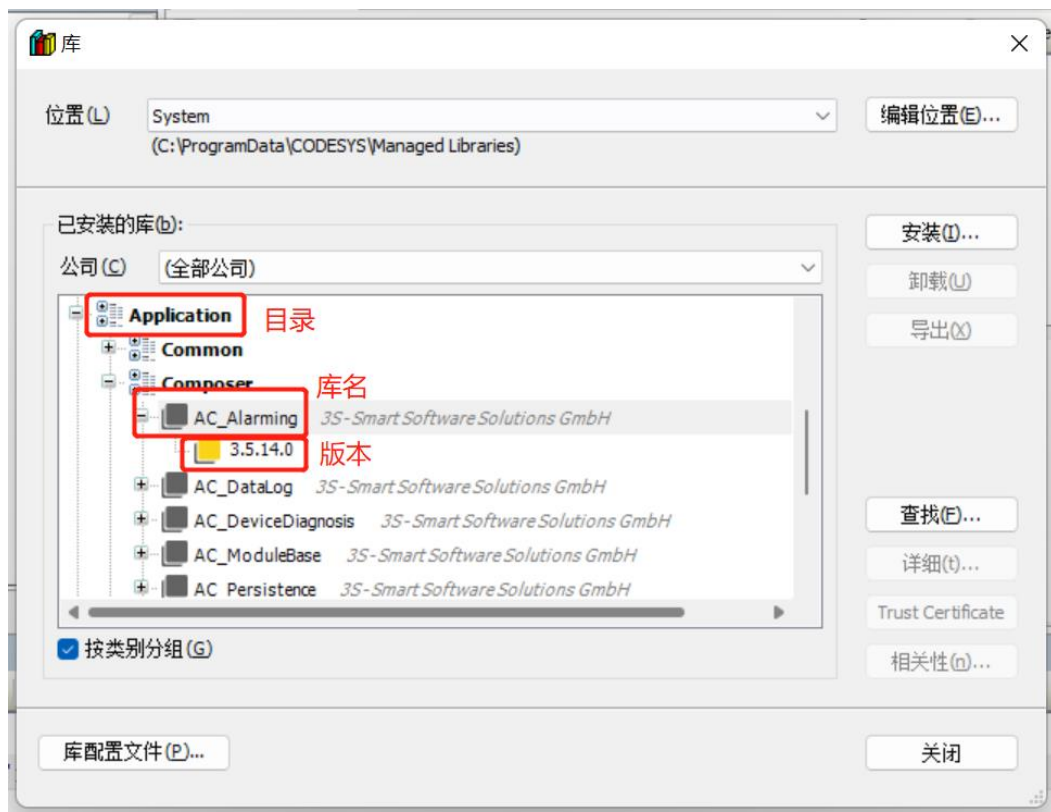
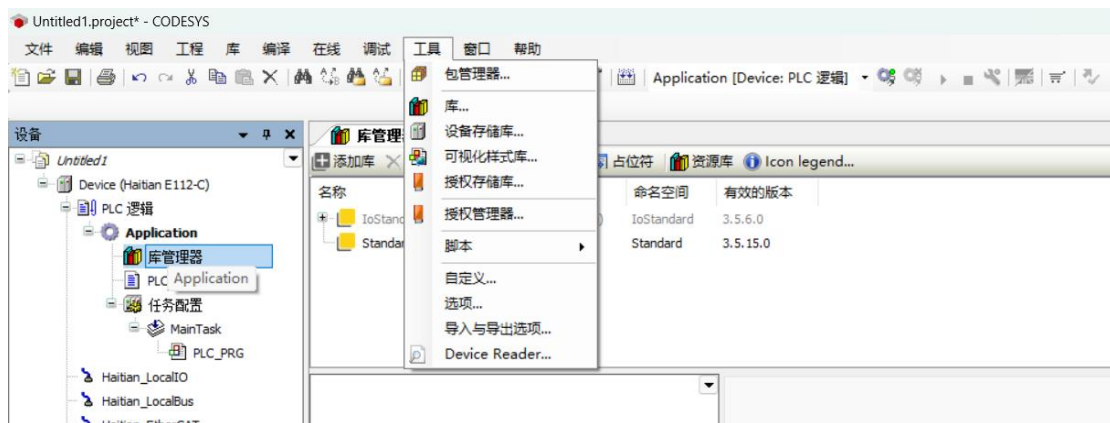
厂商自定义库是根据不同生产厂商硬件设备的环境而配置的应用库。通常只有使用该生产厂商的硬件才能匹配对应的库文件，因此使用前需要详细阅读库文件的说明文档。

2.5.2 库文件的管理

库管理器显示与当工程项目有关或调用的所有库，库管理器通过“库管理器”命令打开，包括库在内的有关信息和项目一起保存。



如果需要在计算机上安装其他供应商提供的库文件，则需要使用库文件管理器。库文件管理器通过使用菜单栏命令“工具”→“库”实现。

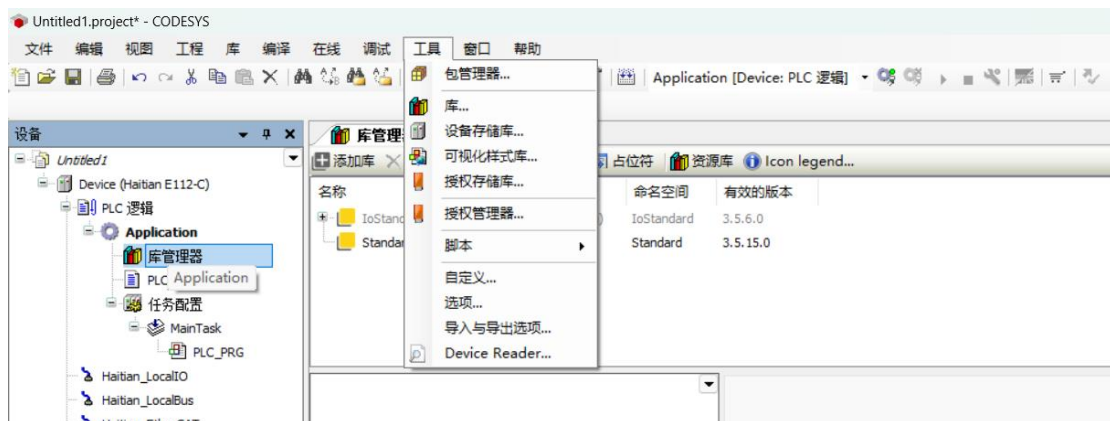


2.5.3 库文件的安装与调用

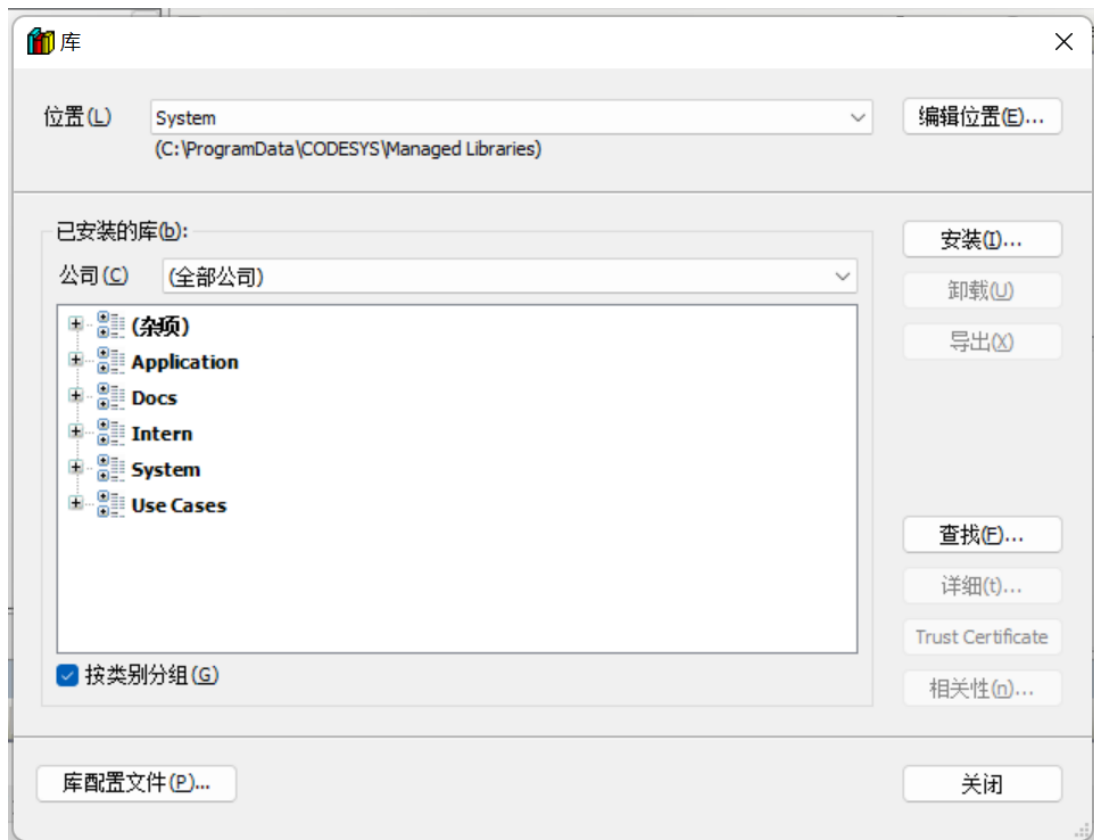
2.5.3.1 库文件的安装

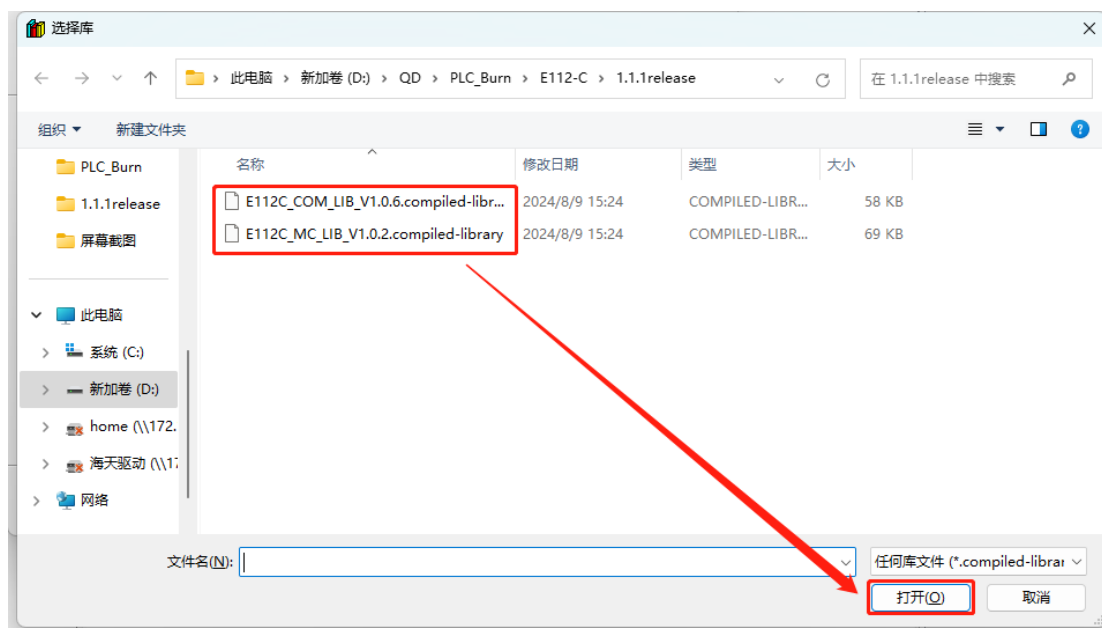
在使用一个库之前必须先先在“库”对话框进行“安装”，安装后才能工程项目中调用该库。
E112C 控制器共有两个库文件及各自版本分别为 E112C_COM_LIB_V1.0.6.compiled-library，
E112C_MC_LIB_V1.0.2.compiled-library，安装步骤如下：

(1) 首先点击菜单栏中的“工具”，选择“库”。



(2) 点击“安装”后，找到对应库文件选中后点击“打开”，依次安装上述两个库文件。

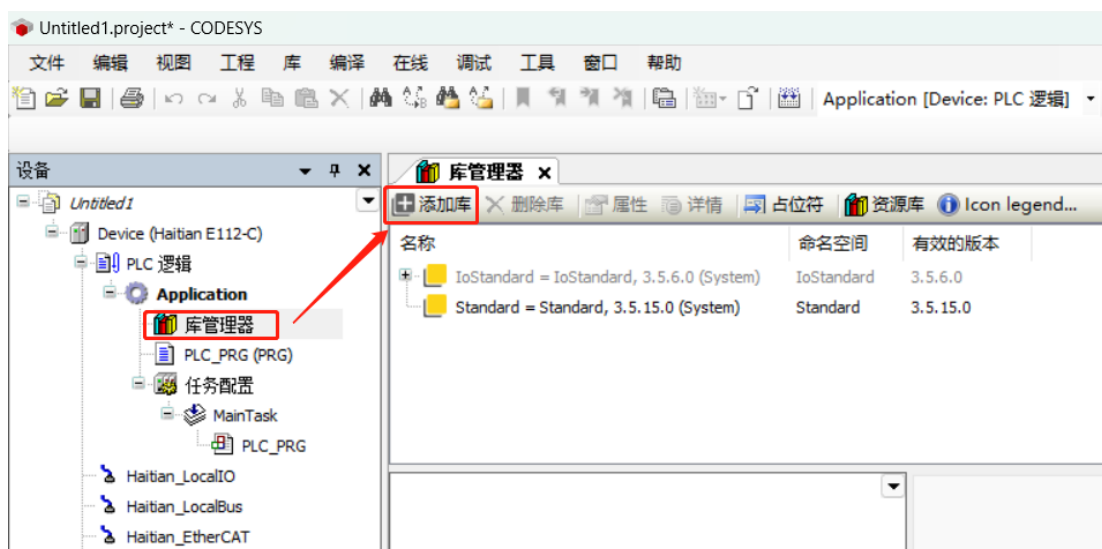




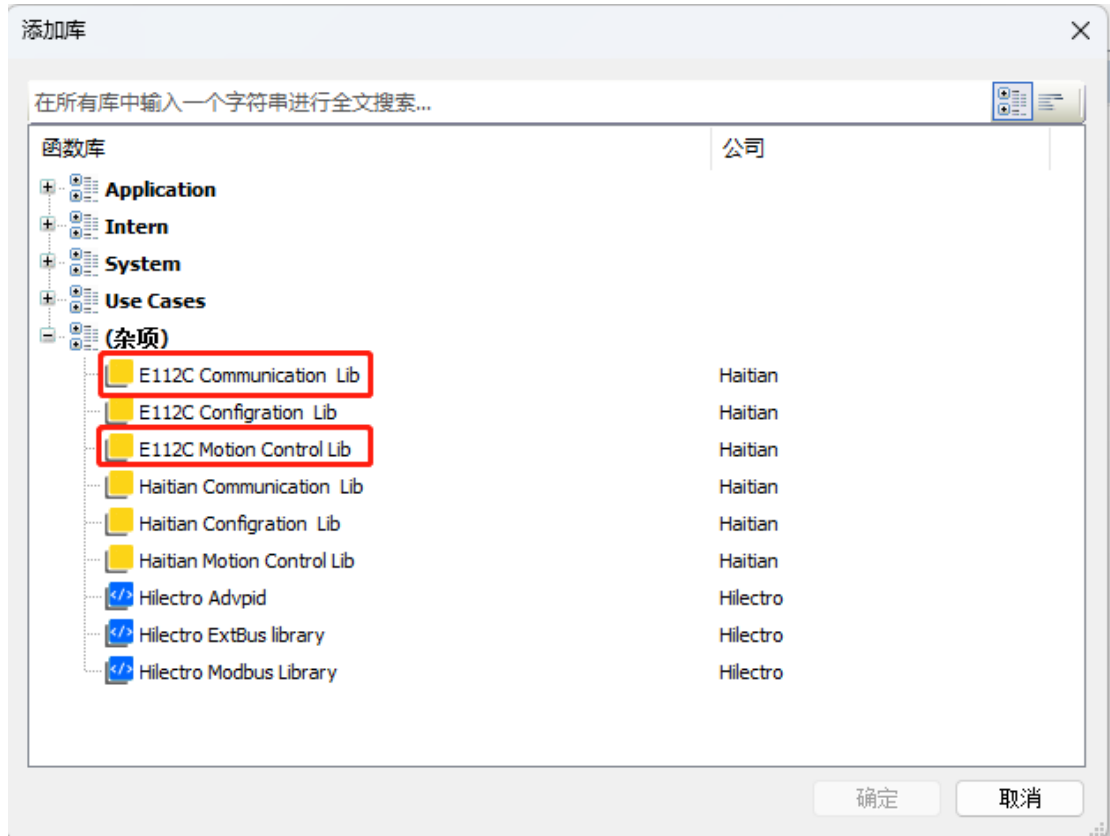
2.5.3.2 库文件的调用

安装过库文件后，需要在项目中对库文件进行添加才能调用其中的函数或功能块等，此时需要库管理器实现此功能。

(1) 在工程项目中双击“库管理器”，随后点击“添加库”。



(2) 当选择“添加库”后，用户可以选择之前已经安装过的库文件，并将其导入到项目中，可根据供应商名称、库文件、版本等功能进行选择。查找并选择上述所安装的两个库文件，点击“确定”导入到项目中。

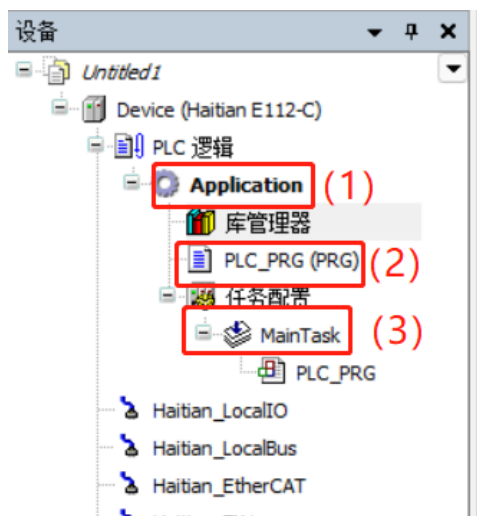


(3) 完成两个库导入后如图所示，导入后可查看库文件中的函数及功能块详细说明。



2.6 软件左侧栏介绍

CODESYS 的软件模型用分层结构表示，每一层隐含其下面层的许多特性。软件模型描述了基本的软件元素及其相互关系，这些软件元素包含设备、应用、任务、全局变量、访问路径和应用对象。所有的结构集中在 CODESYS 工程界面的左侧列表，一般习惯性地称其为设备树。



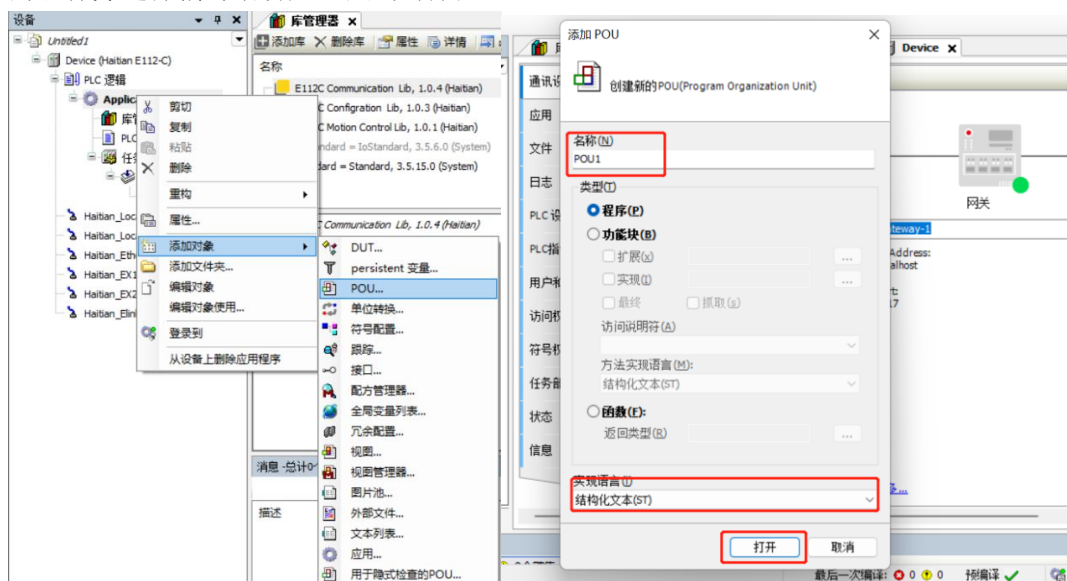
(1) 应用：在每一个设备中，有一个或多个应用，它为运行程序提供了一个支持系统，而且它反映了 PLC 的物理结构，在程序的 PLC 物理 I/O 通道之间提供了一个接口。

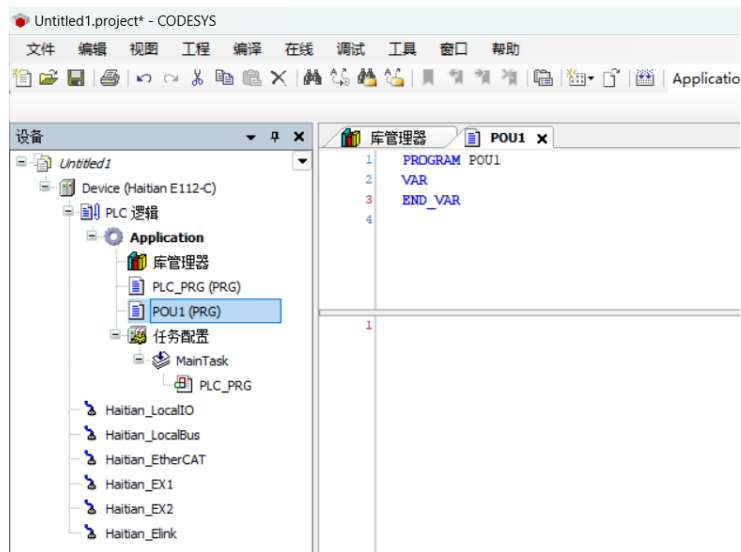
(2) 程序：程序组织单元（Program Organization Unit, POU）由声明区和代码区两部分组成，是用户程序的最小软件单元。

(3) 任务：任务用于规定程序组织单元在运行时的属性，它是一个执行控制元素，具有调用的能力。

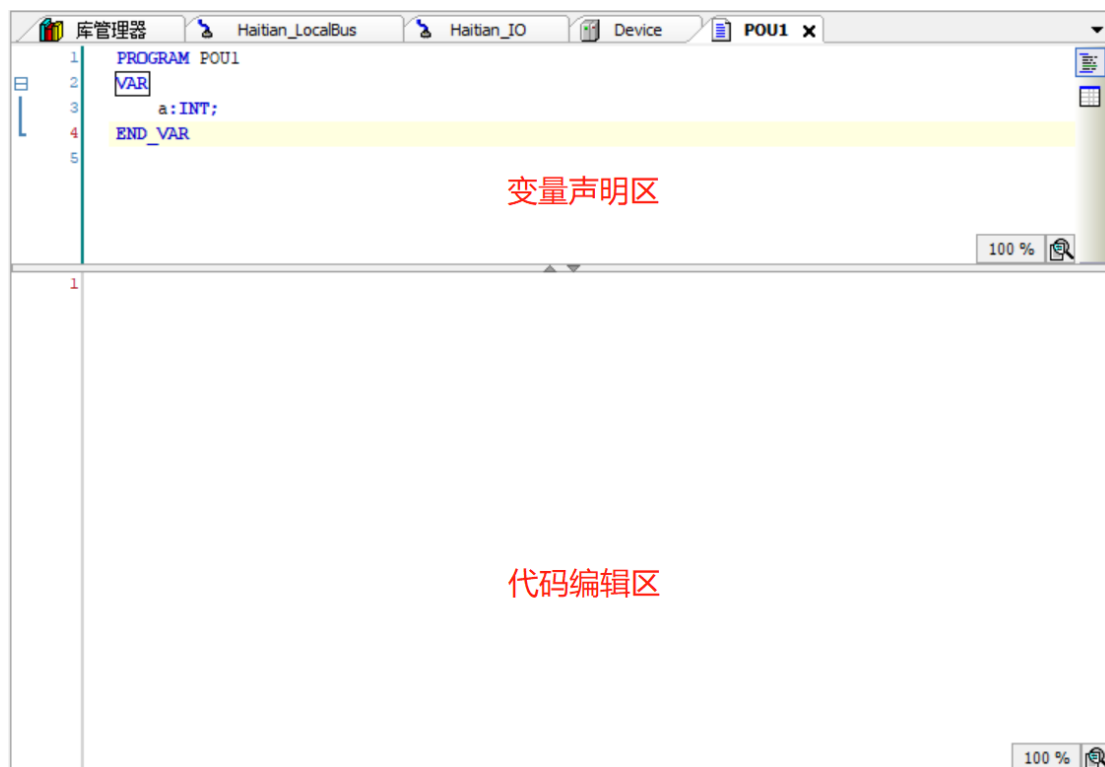
2.7 PLC 用户程序编写

新建一个工程项目后，系统会自动生成一个默认的周期性任务，任务中包含有一个默认的程序为“PLC_PRG”，该程序为基本的软件单元（一个 POU）。如果需要自行建立一个新的程序，可点击设备树中的“Application”右键在“添加对象”栏中选择“POU”，接着填写该 POU 的名称并选择编写语言后，点击“打开”。





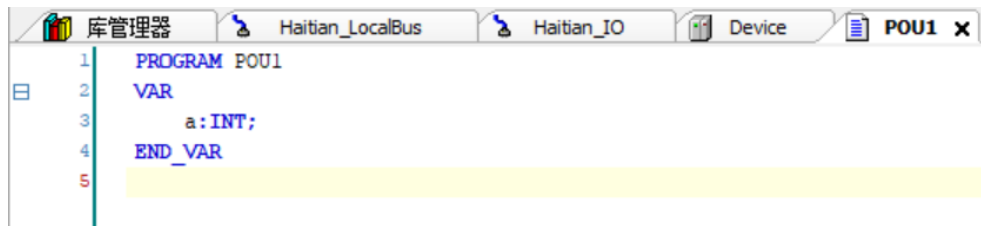
双击设备树中的“POU1”，自动在用户界面中部的编程语言编辑器中打开。编程语言编辑器包含变量声明区域和代码编辑区域，由一个可调分割线分开。



声明区域包括：显示在左侧边框的行号、POU 类型和名称（如“PROGRAM POU1”），以及关键字“VAR”和“END_VAR”。

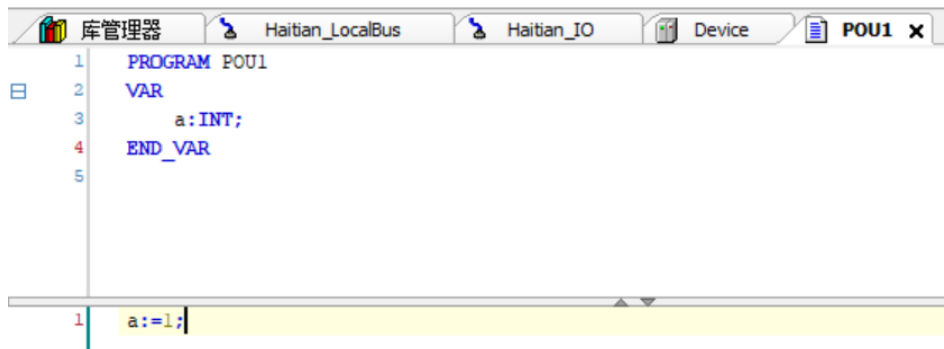
(1) 变量声明

在编辑器的声明部分，将鼠标移到 VAR 后，键入“enter”插入新的空行，声明 INT 类型变量 a，键盘输入为“a:INT;”。



(2) 程序编写

在代码编辑区键入“a:=1;”为变量 a 赋值为 1。

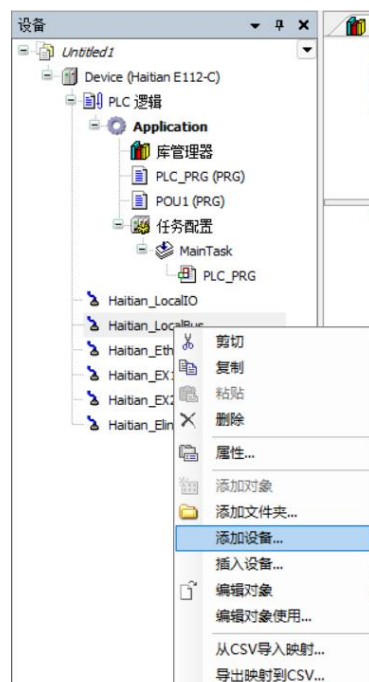


综上，通过在 Application 中添加一个名为 POU1 的程序，并在程序声明区和代码区输入相应代码完成一个简单的程序编写：为声明的 INT 型变量 a 赋值为 1。

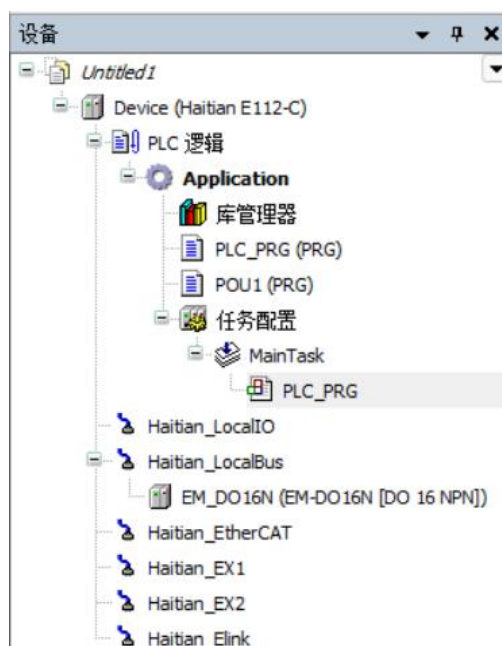
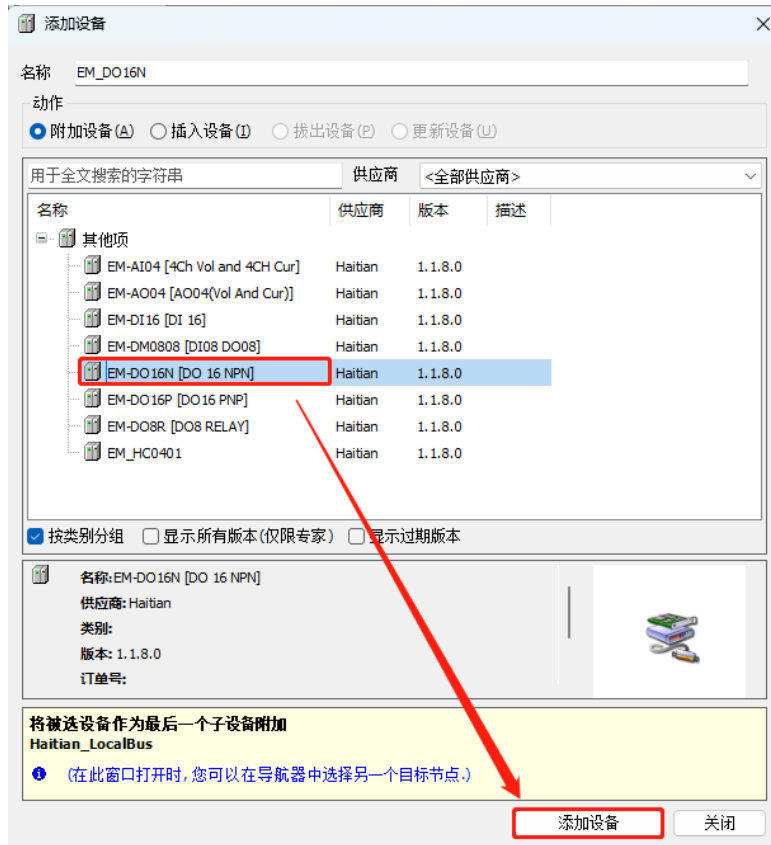
2.8 用户程序变量与端口的关联配置

E100 系列控制器可进行模块扩展，本章将说明如何通过编写程序实现将需要关联的硬件端口与用户程序中的变量进行关联，所扩展部分为 EM-DO16N 扩展模块。

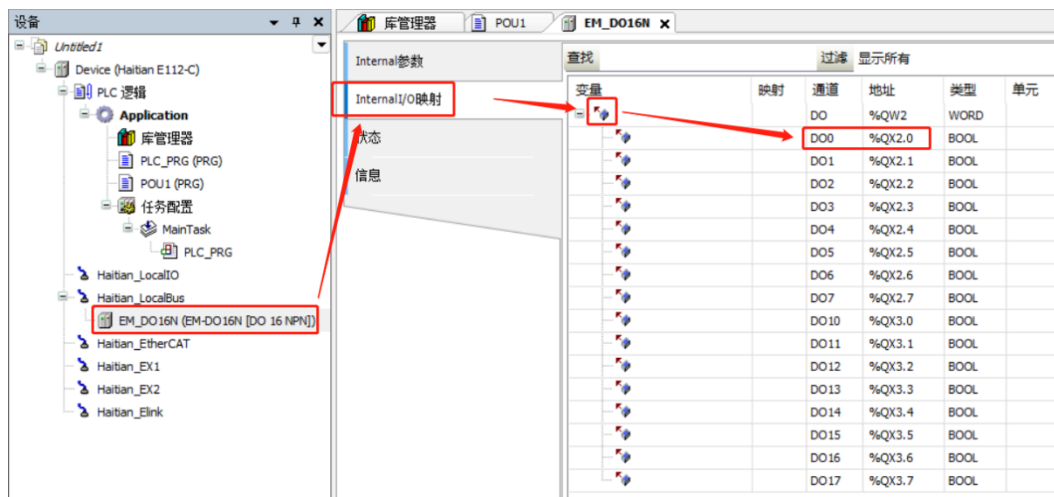
(1) 首先须要在设备树中选“Haitian_LocalBus”点击鼠标右键后选择“添加设备”。



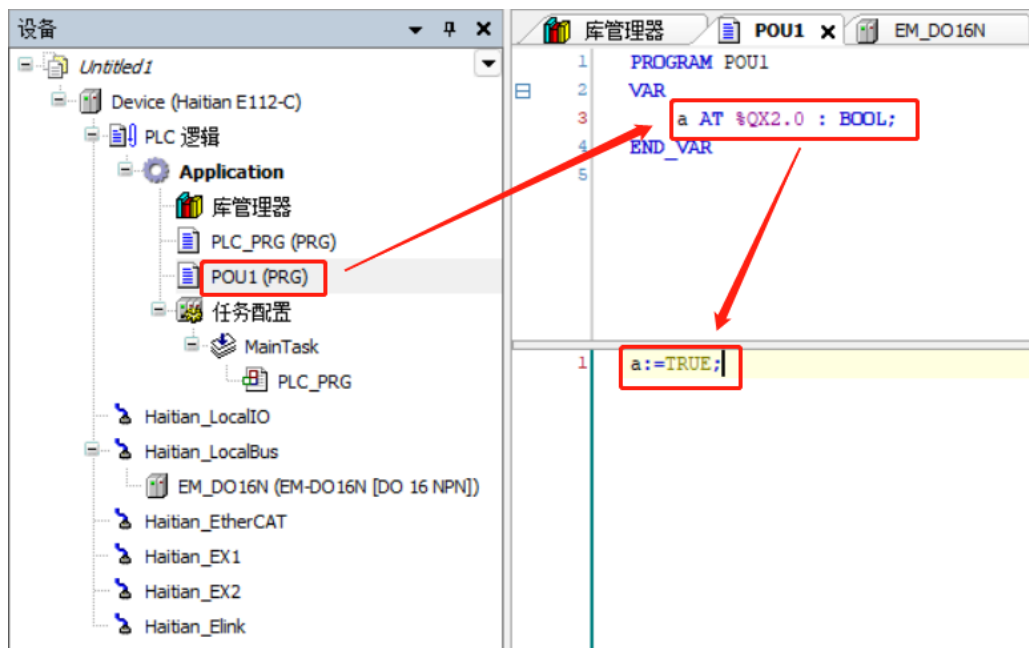
(2) 根据实际硬件扩展模块从左至右的顺序依次添加相关扩展模块设备，本例中硬件扩展为 EM-DO16N（只有一个扩展模块）。因此在 CODESYS 中首先添加 EM-DO16N 模块（如果有后续接入第二个模块需要自行按顺序添加相应设备），选择“EM-DO16N [DO 16 NPN]”，随后点击“添加设备”完成 EM-DO16N 模块添加。

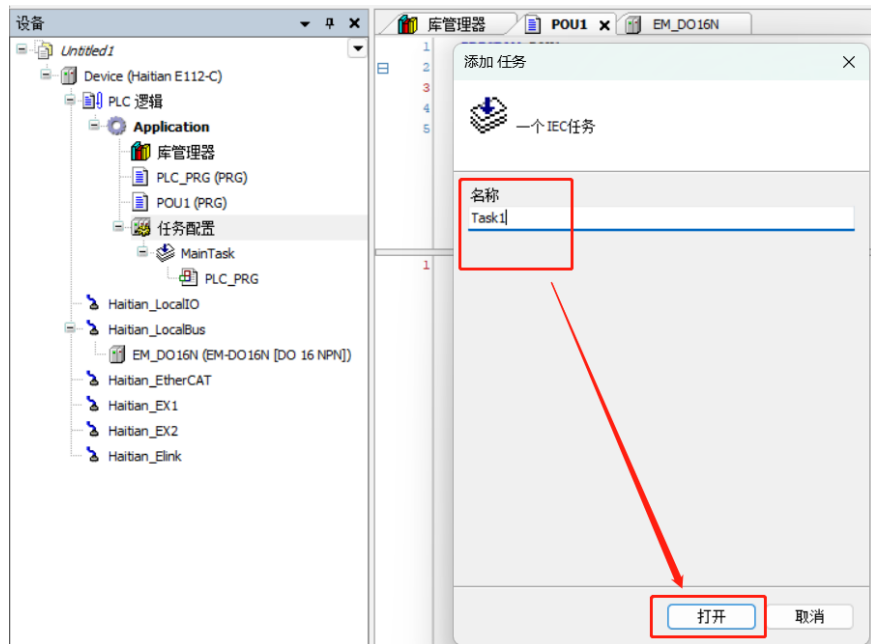


3) 双击设备树中“EM-DO16N [DO 16 NPN]”，点击“Internal I/O 映射”，在 DO0 组中查看 EM-DO16N 模块的第一个输出地址为“%QX2.0”。

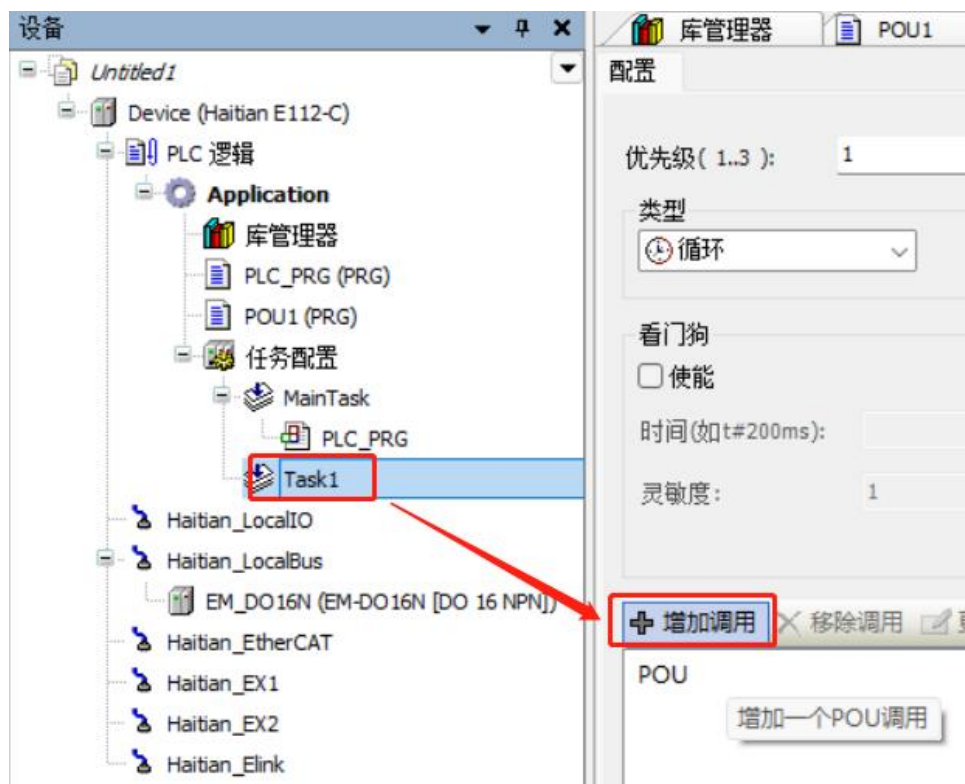


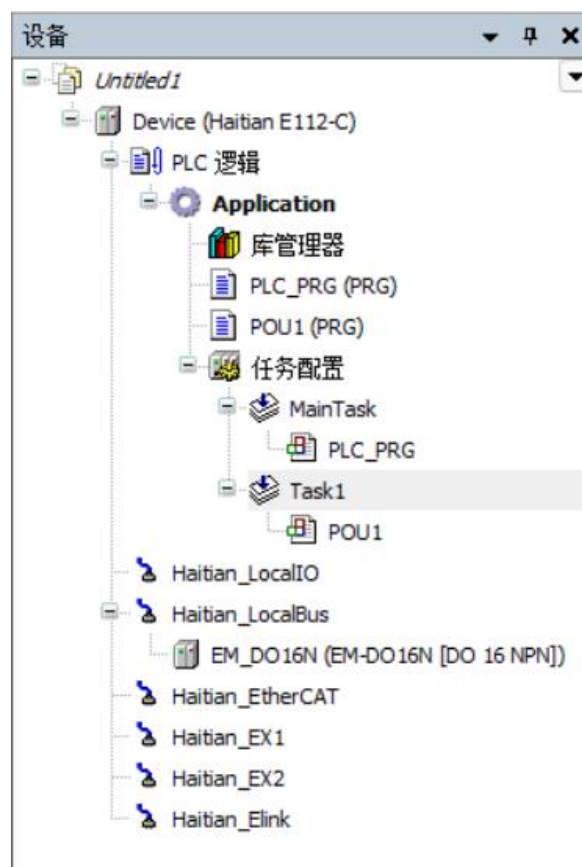
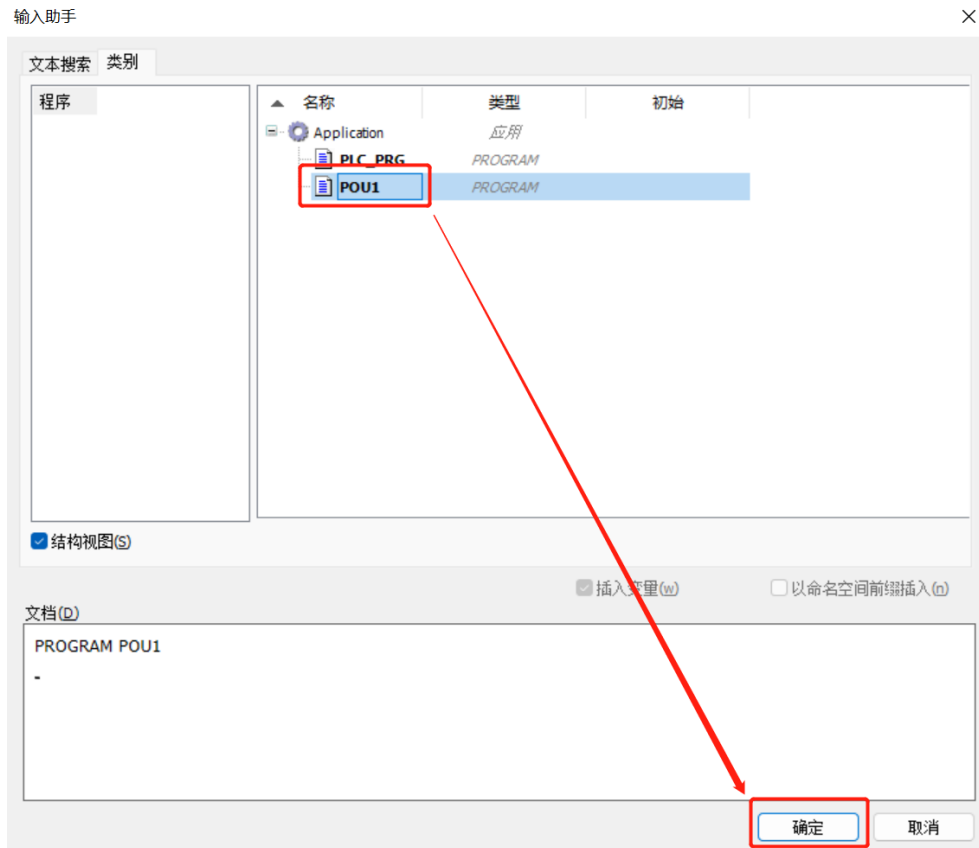
4) 双击设备树中 2.4.1 中创建的程序“POU1”，在声明变量区域将关键字“VAR”和“END_VAR”之间的内容清除，随后将变量 a 与 EM-DO16N 模块第一个输出端口关联，键入“a AT %QX2.0:BOOL;”，接着清空代码编辑区，键入“a:=TRUE;”。





(2) 将 2.4.2 中创建的“POU1”程序文件添加到“Task1”任务下：双击“Task1”，可修改该任务参数，本例选择创建时的默认参数，随后点击“增加调用”，选中 POU1 程序后点击“确定”。





2.10 CodeSys 连接设置

2.10.1 网关配置

Gateway Server 主要起到通信网关的作用,在程序下载之前,必须先使用 Gateway Server,该服务程序由 CODESYS 安装程序直接提供。

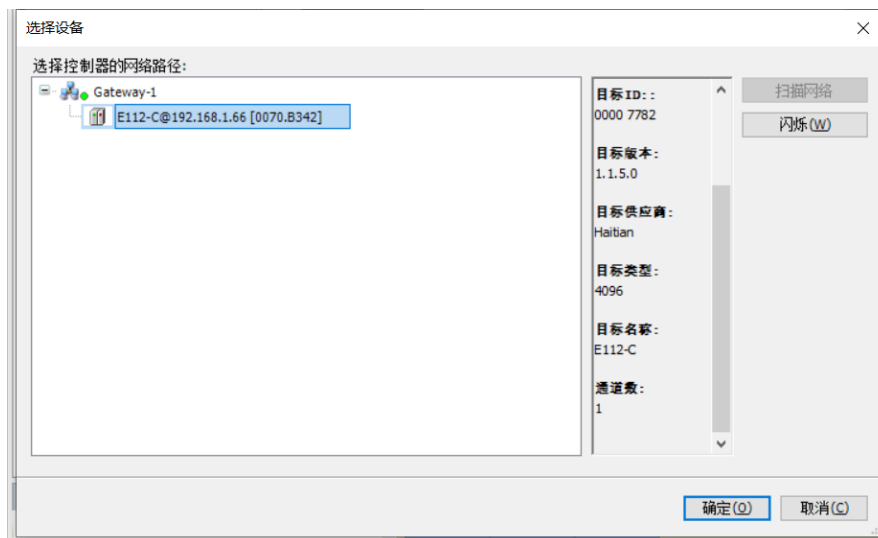
启动 Gateway Server: Gateway Server 作为服务程序一般在系统启动时就自动启动,无需再单独启动,可通过系统托盘查看是否有指示 Gateway Server 运行的图标  判断 Gateway Server 是否正常启动。

2.10.2 以太网通讯

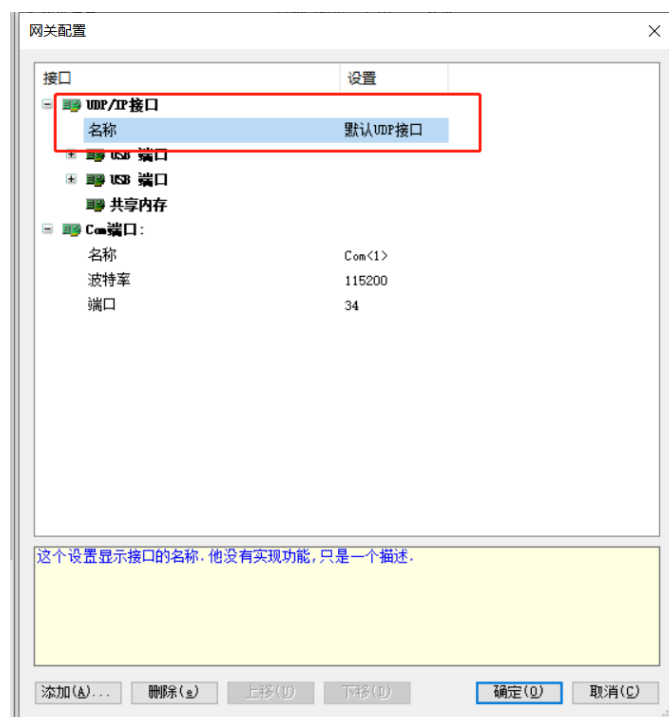
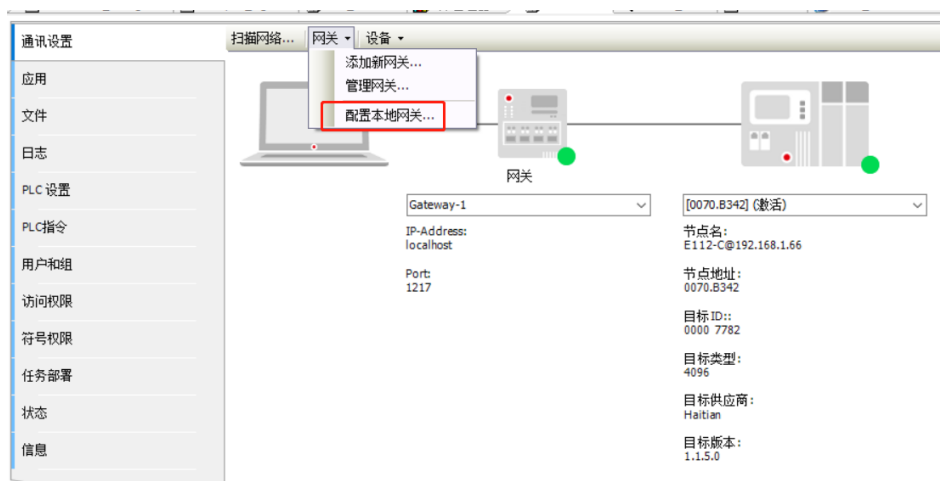
E112C 控制器默认的 IP 地址为“192.168.1.66”,计算机连接控制器时需要将计算机本地的 IP 地址设置成和控制器 IP 地址同一网段,例如可以将计算机 IP 地址设置成“192.168.1.111”,子网掩码设置成“255.255.255.0”。

将网线一头接在 E112C 控制器的 EtherNET 口,另一头接在电脑网口。网络添加完成后有时需要重启网关 Gateway Server,点击系统托盘  →点击“Stop Geteway”等候几秒,再点击“Start Geteway”后网关重启成功。双击设备树中“Device”,再点击设备对话框中的“扫描网络”即可扫描到当前连接的硬件设备。点击“确定”与硬件设备建立连接。此时 CODESYS 软件已与 E112C 控制器在通讯层面连接成功。可进行下一步操作。





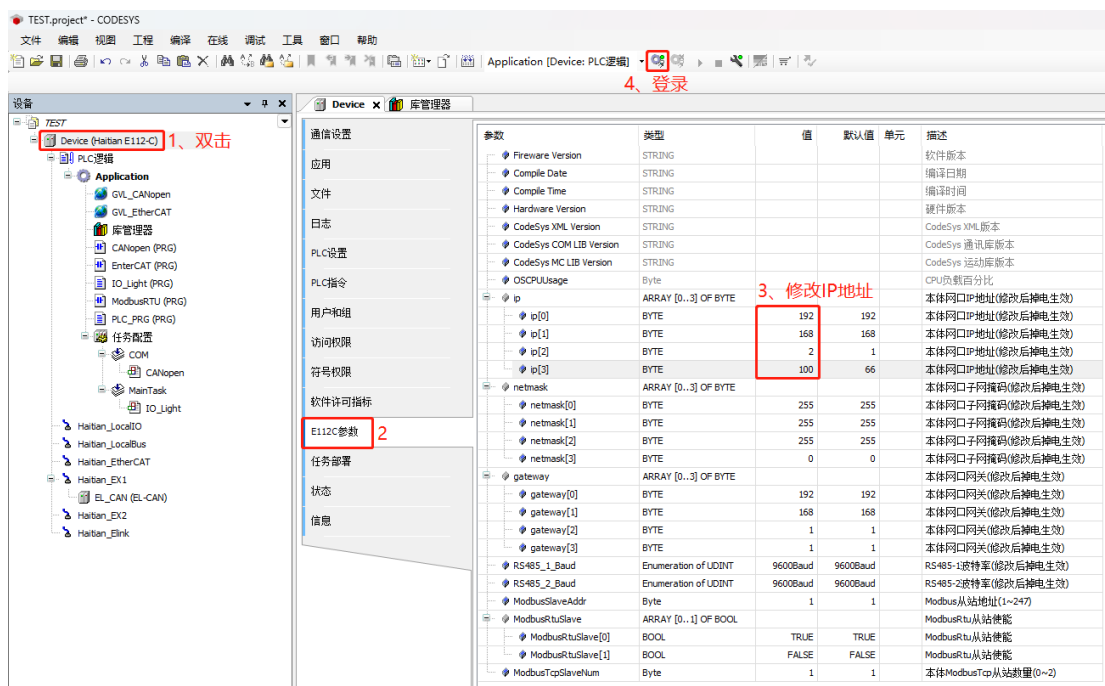
如果还是无法扫描到设备，请检查本地网关配置，默认为 UDP 接口，请参考下图配置：



2.10.3 控制器 IP 设置

2.10.3.1 通过 CODESYS 平台修改 IP

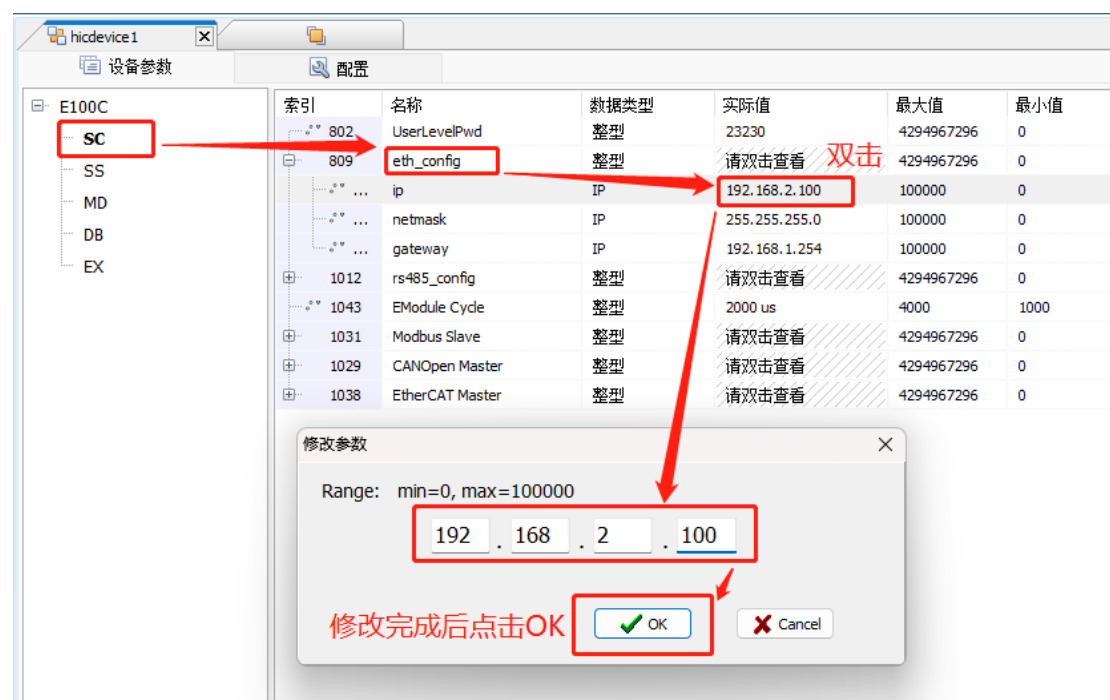
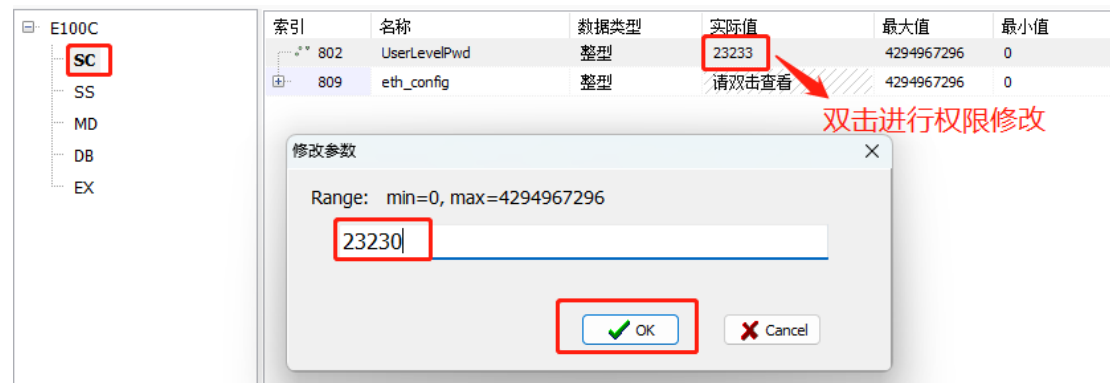
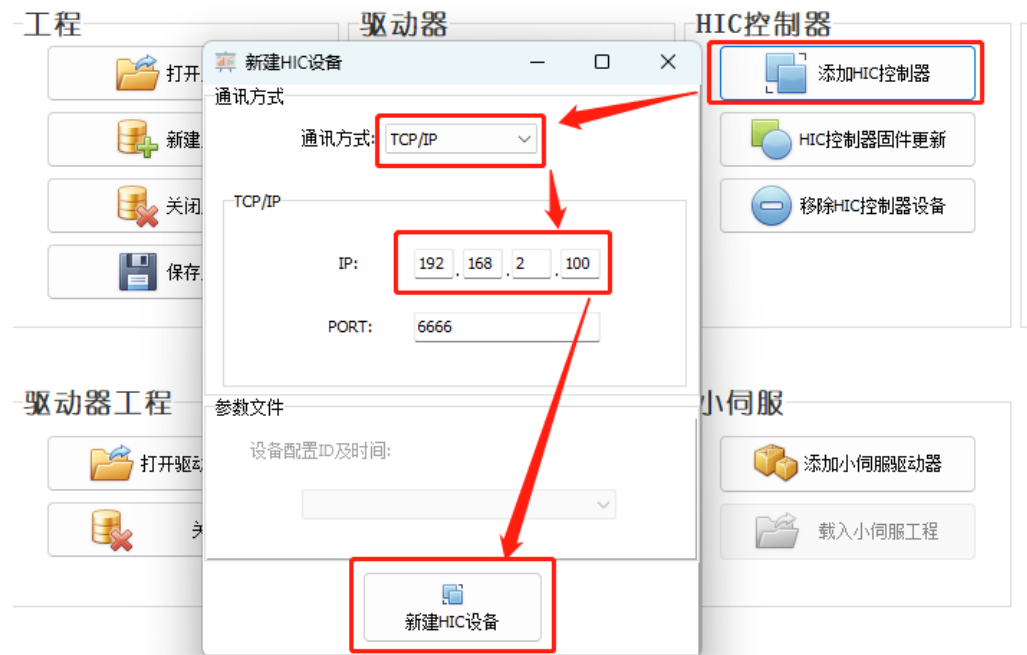
打开 CODESYS 工程，双击 Device，在“E112C 参数”中进行 IP 地址修改（如将 192.168.1.66 修改为 192.168.2.100），登录运行程序后，退出登录，由于改配置需要重启生效，请将 E112C 控制器重启上电。



2.10.3.2 通过 HIMPV3 上位机平台修改 IP

打开 HIMPV3 上位机，添加 HIC 控制器，通讯方式选择 TCP/IP，将 IP 设为控制器的 IP 地址，点击新建 HIC 设备。在设备下找到 SC 栏，找到 eth_config 选项下的 ip，双击对其进行修改，修改完成后，在空白处右键“保存修改数据”，由于该配置需要重启后生效，请将控制器重新上电。**注意：如果需要修改其他配置**，可以双击 UserLevelPwd 将用户权限进行修改，将 23233 改为 23230。

注意：请确认 HIMPV3 上位机版本为 V3.3.1.17 以上，否则无法修改。



以下步骤非常重要，请不要遗忘，保存修改后再重启上电。

索引	名称	数据类型	实际值	最大值	最小值
802	UserLevelPwd	整型	23230	4294967296	0
809	eth_config	整型	请点击查看	4294967296	0
1012	rs485_config	整型	请点击查看	4294967296	0
1043	EModule Cycle	整型	1000 us	4000	1000
1031	Modbus Slave	整型	请点击查看	4294967296	0
1029	CANOpen Master	整型	请点击查看	4294967296	0
1038	EtherCAT Master	整型	请点击查看	4294967296	0

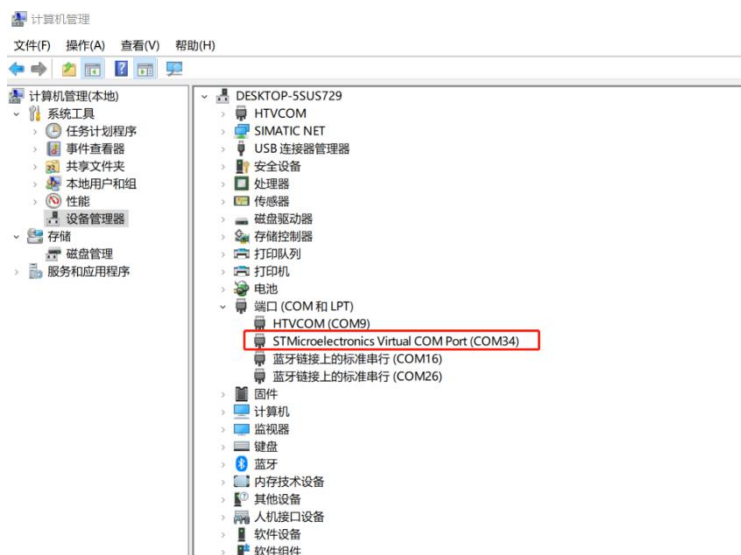


2.10.4 USB 通讯

使用 USB 线连接：首先请下载并安装 USB 串口驱动，下载地址：

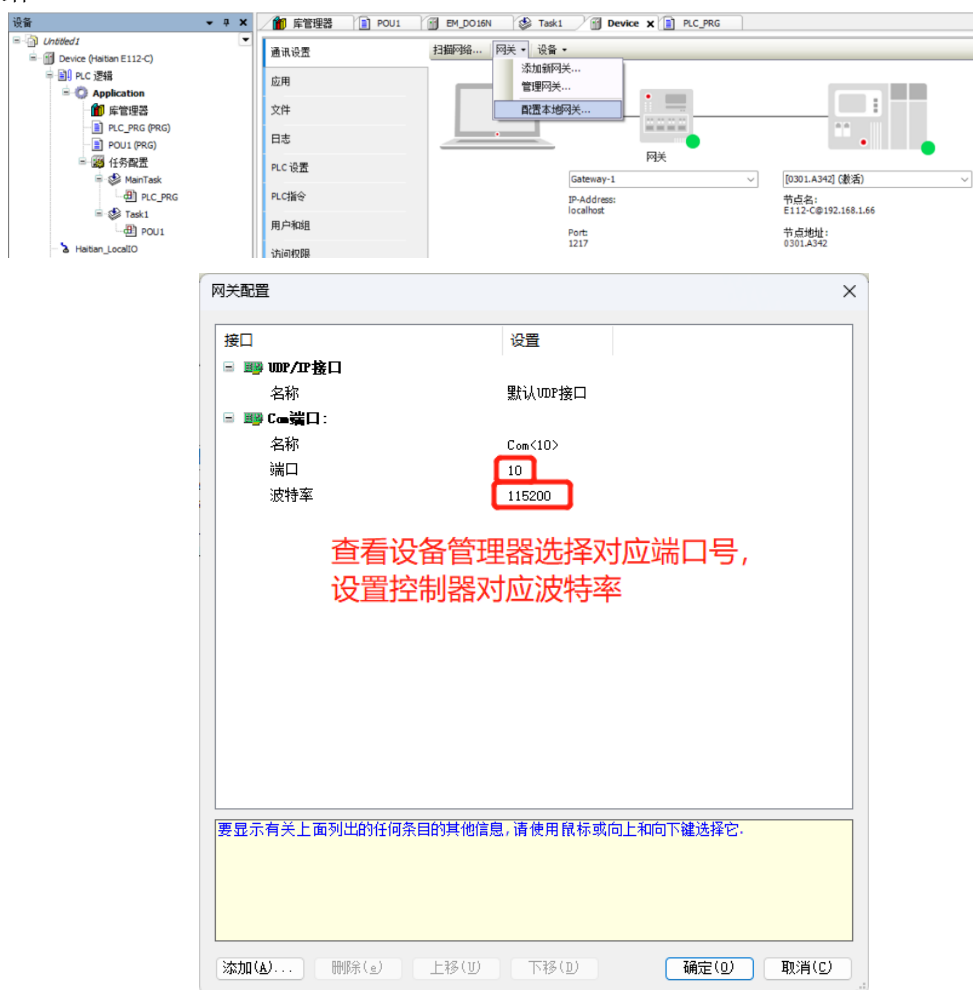
http://co.haitianiot.com:7010/uploadfile/2024-06-12/1718156013408_USB%E9%A9%B1%E5%8A%A8.rar

安装完驱动后将 TYPE-C 口接到控制器上，另一头 USB 口连接电脑，设备管理器会显示当前设备端口号，如下：



打开 CODESYS 软件，双击设备树中“Device”，再点击设备对话框中的“网关”下拉键，选择“配置本地网关”，点击左下“添加”选择“添加顶级接口”，配置好对应参数值（参考下图）。网络添加完成后有时需要重启网关 Gateway Server，点击系统托盘  → 点击“Stop Geteway”等候几秒，再点击“Start Geteway”后网关重启成功。双击设备树中“Device”

随后点击设备对话框中的“扫描网络”即可扫描到当前连接的硬件设备。点击“确定”与硬件设备建立连接。此时 CODESYS 软件已与 E112C 控制器在通讯层面连接成功。可进行下一步操作。



2.11 程序下载/读取

2.11.1 编译

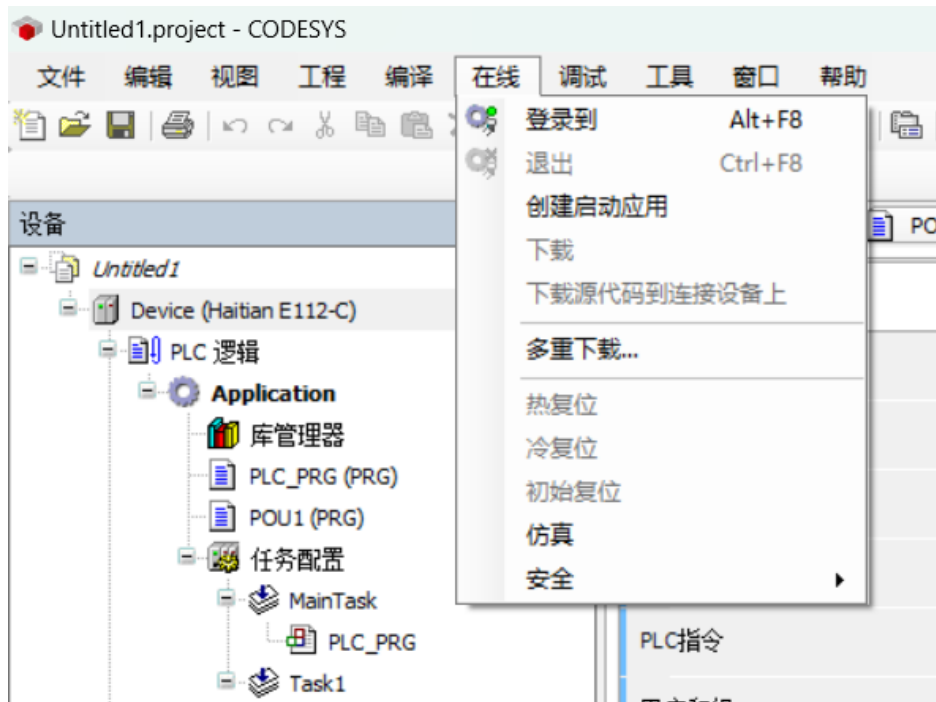
程序编写完成后，在下载硬件设备之前需要对程序进行编译，编译命令对编写的程序进行语法检测，如果创建的 POU 没有添加到任务中，编译命令不对该 POU 进行语法检测。编译功能可在菜单栏“编译”→编译中操作也可以在工具栏  按钮直接进行编译指令。编译通完成后会在消息栏有相关信息弹出，用户可鼠标左键双击消息栏信息直接跳转到错误代码处。如果编译通过后发现所需要运行测程序单元显示为灰色，请检查该单元是否已成功被任务所调用。

2.11.2 登录及下载

登录使应用程序与目标设备建立连接并进入在线状态。正确登录的前提条件是正确配置

设备的通信设置并且应用程序必须是无变异错误的。

对于以当前活动应用登录，生成的代码必须没有错误和设备通信设置必须配置正确，登录后系统会自动选择程序下载。



2.12 在线监控

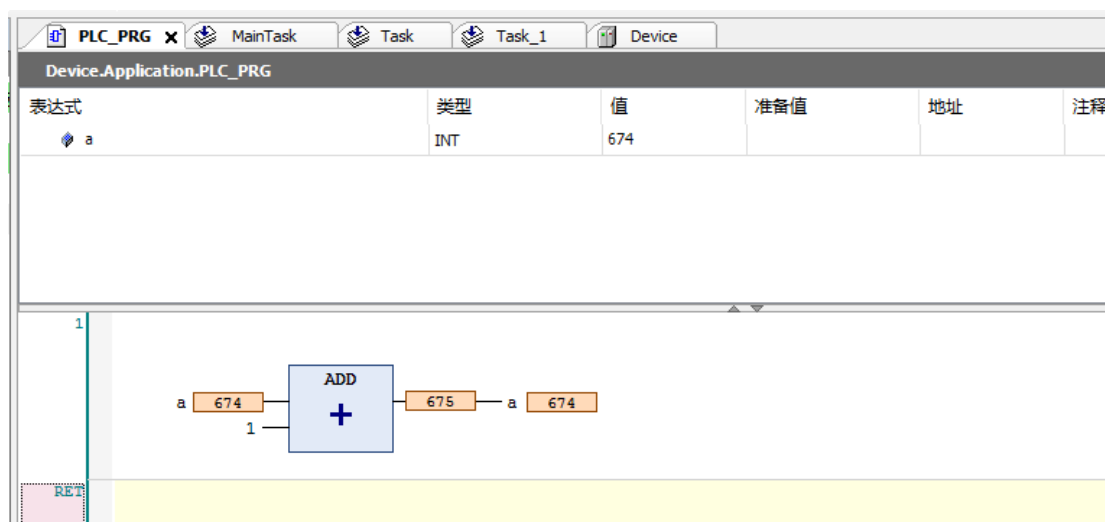
使用以下三种方法，可以在线监视应用程序中的变量。

1. POU 在线监控
2. 特殊变量在线监控
3. 写入和强制变量

2.12.1 POU 在线监控

双机设备窗口里的执行程序“PLC_PRG”或邮件单击该项，在弹出的快捷菜单中选择“编辑对象”命令，打开在线视图，出现相应的对话框，这里我们选择 POU_2 以在线模式进行查看。如图 2.10 所示

如图 2.10 所示，上半部分为变量表达式的状态，以表格形式显示，下半部分显示 POU_2 的监视表达式，由每一个变量后的内部监视窗口显示实际的值。

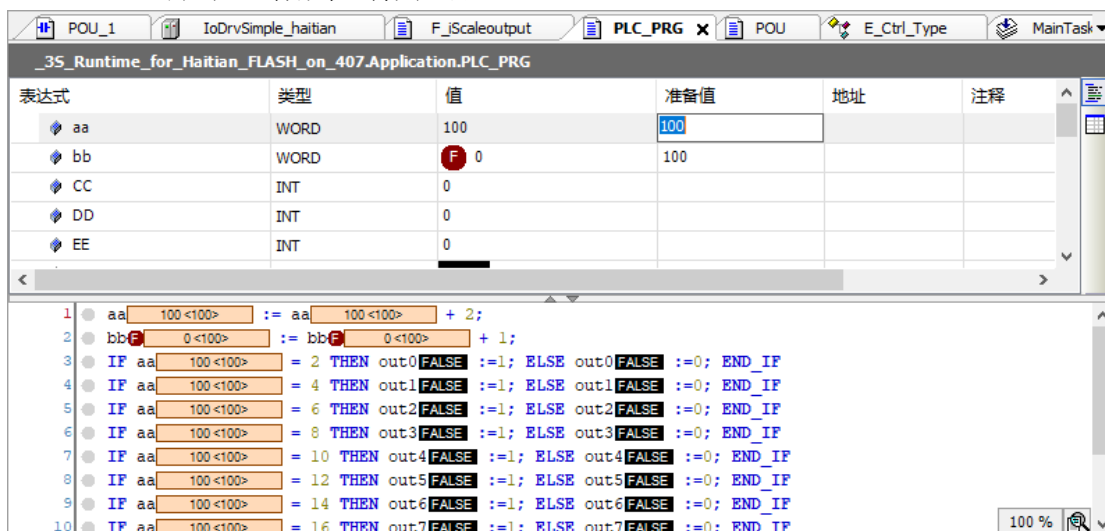


此外如果用户想要同时监控不同 POU 中的变量，把需要在同意情况下观察的变量集合在一起，一并比较，则可以在“视图”→“监视”在其中可配置 4 个监视列表，如单击“监视 1”则程序自动创建监视 1 列表。打开监视 1 后单击表达式下面的空栏，会出现“输入助手”选项，如通过输入助手可以选择要监控的变量并组成列表。在线登录后就可以对变量进行监视。

2.12.2 写入/强制变量

CODESYS 中有两种强制赋值方式供选择，一种为“写入值”，另一种为“强制值”。具体操作方式为点击菜单栏“调试”→“写入值”或“强制值”实现。

1. 写入值：只在下一个周期将变量改为准备中的数据。如果该变量未被其他数据赋值，则保持改数据，否则根据程序里面的赋值语句进行执行。通过“写入值”赋值的数据显示正常的黑色或蓝色。
2. 强制值：从下一个周期开始，在每个循环后再次设定为某个强制值，如果系统有程序给该变量赋值，该变量的值任为强制的值，强制优先级比较高。通过该方式写入的值在变量列表左侧有一个红色的“F”标识，如图 2.11 所示。强制值可通过“调试”→“释放值”将其还原。



2.12.3 曲线采样

采样功能类似于示波器，在程序调试和诊断过程中，他是个非常实用和有效的工具该功能可以把程序执行过程中需要采集的数据记录下来，并通过曲线方式展示出来。

右键单击“Application”在弹出快捷菜单中选择“添加对象”→“跟踪”按鼠标左键确认。如图 2.12 所示。随后弹出“添加跟踪”对话框，在“跟踪名称”文本框中填写跟踪的名称，如“trace1”，如图 2.13 所示，单击“打开”按钮。

进入曲线界面后点击右上角“配置”弹出跟踪配置对话框在任务出选择要跟踪的任务如图 2.14 所示。点击确定。然后点击“添加变量”弹出跟踪配置对话框，在变量处可通过输入助手选择要跟踪的变量如图 2.15 所示。

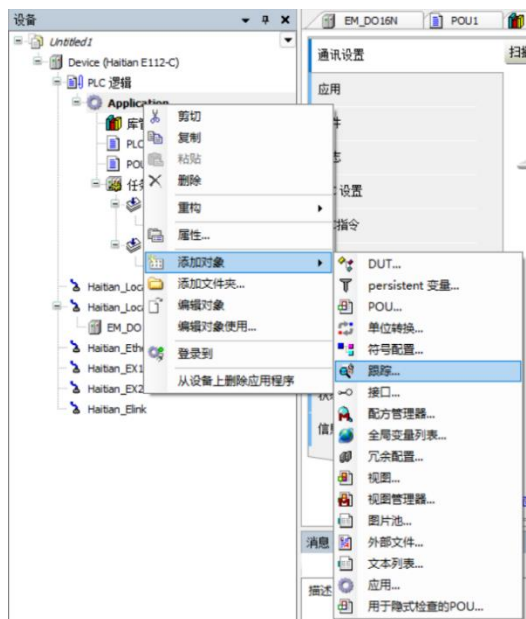


图 2.12 添加跟踪

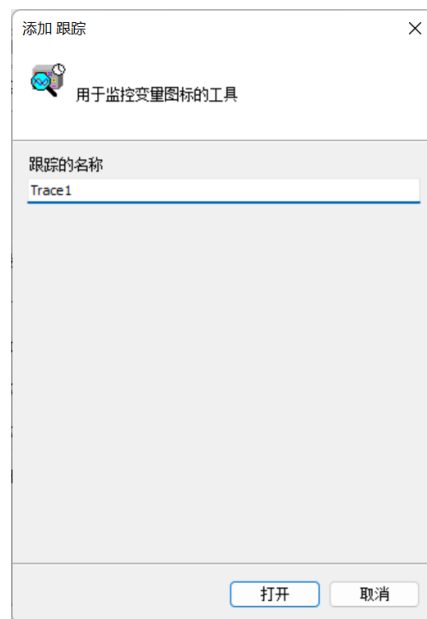


图 2.13 输入跟踪名称

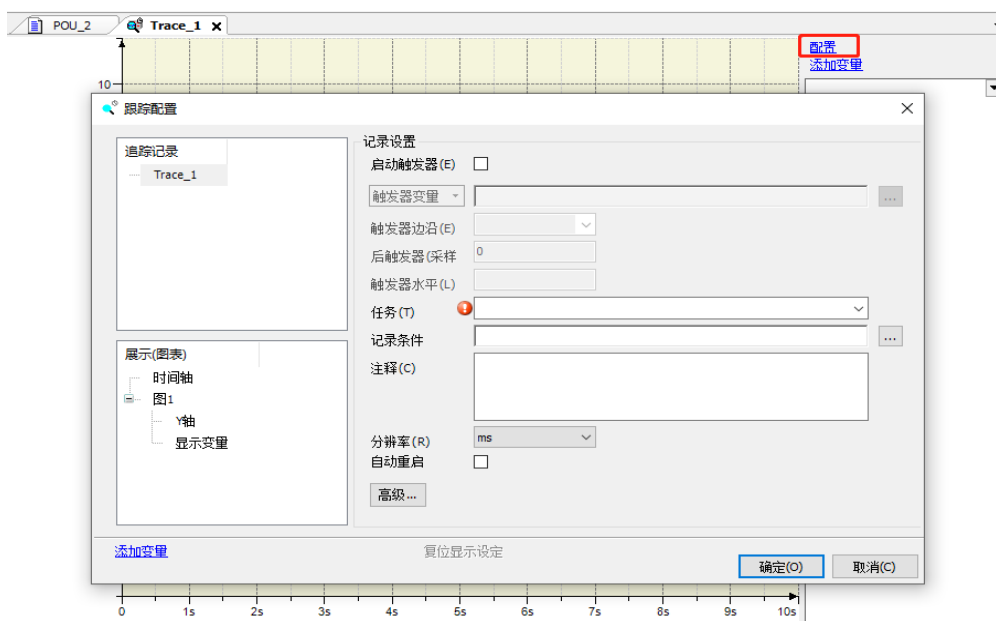


图 2.14 跟踪配置

配置完成后在曲线界面单击鼠标右键选择“下载跟踪”即可采集到所监视变量的曲线变化，如图 2.16 所示。

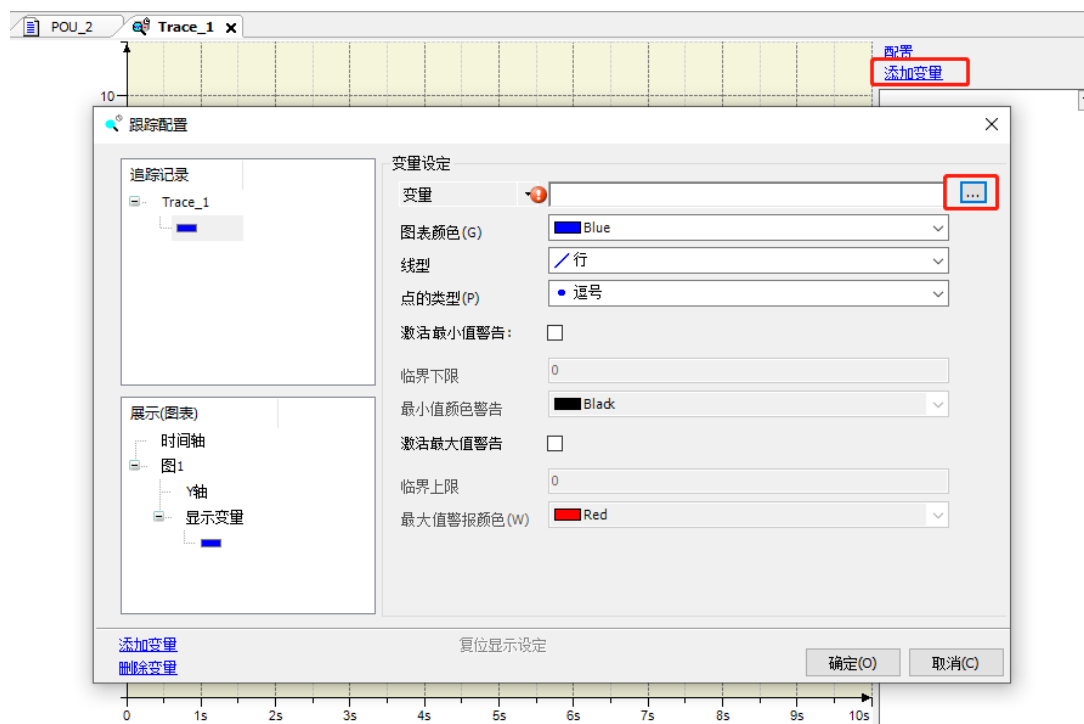


图 2.15 添加跟踪变量

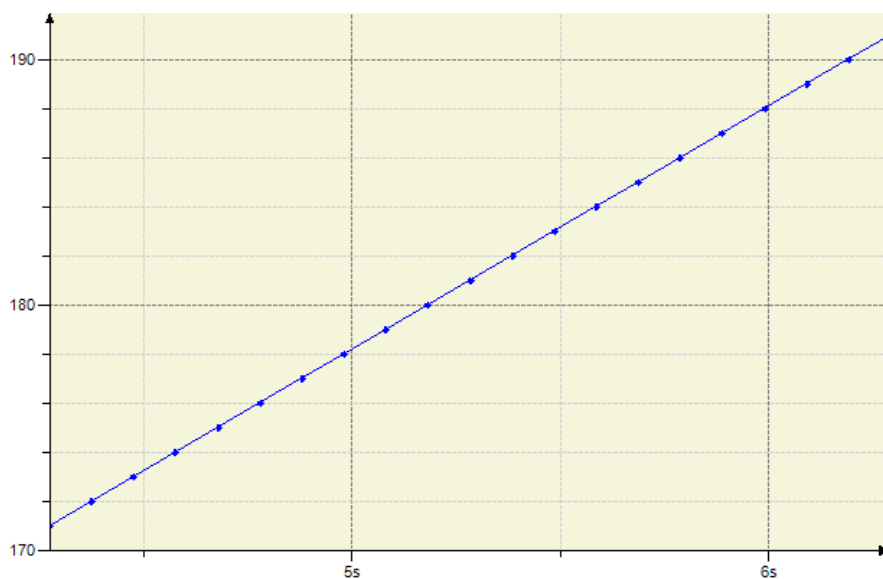


图 2.16 跟踪曲线

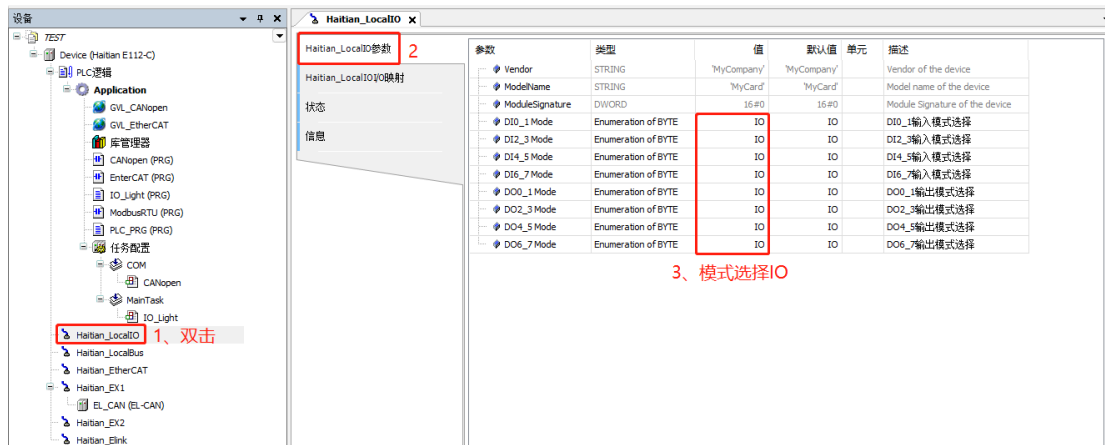
2.13 PLC 本机高速 IO

E112-C 本机自带单相 4 路 200K（AB 相）高速输入和 4 轴 200K（脉冲+方向）高速输出，高速输入输出的端口同样可以作为普通输入输出使用。

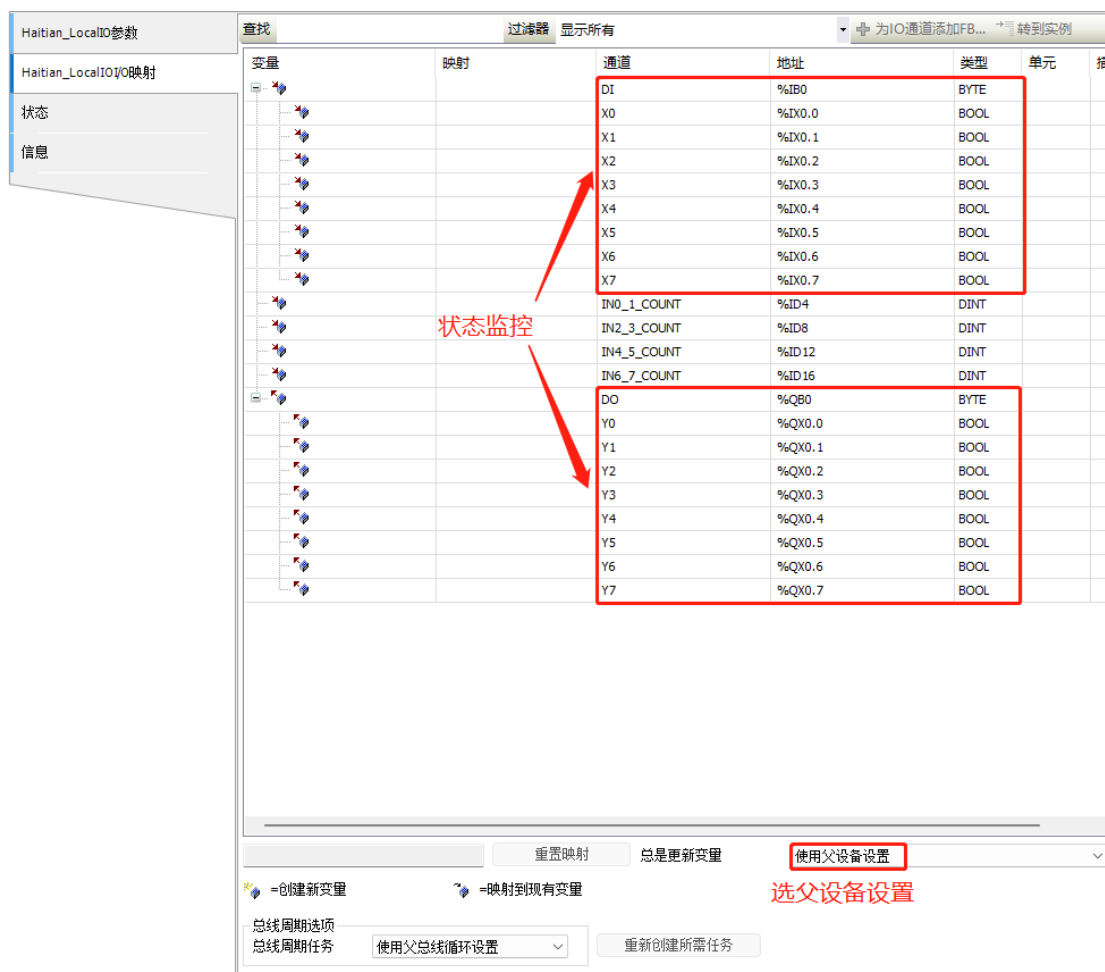
2.13.1 普通 IO

作为普通 IO 时，在软件上的组态步骤如下：

(1) 双击”Haitian_LocalIO”，选择“Haitian_LocalIO 参数”，将对应 IO 口的输入输出模式设为“IO”。



(2) 设置完成后，在“Haitian_LocalIO I/O 映射”下就可以监控当前本地 IO 点的状态。

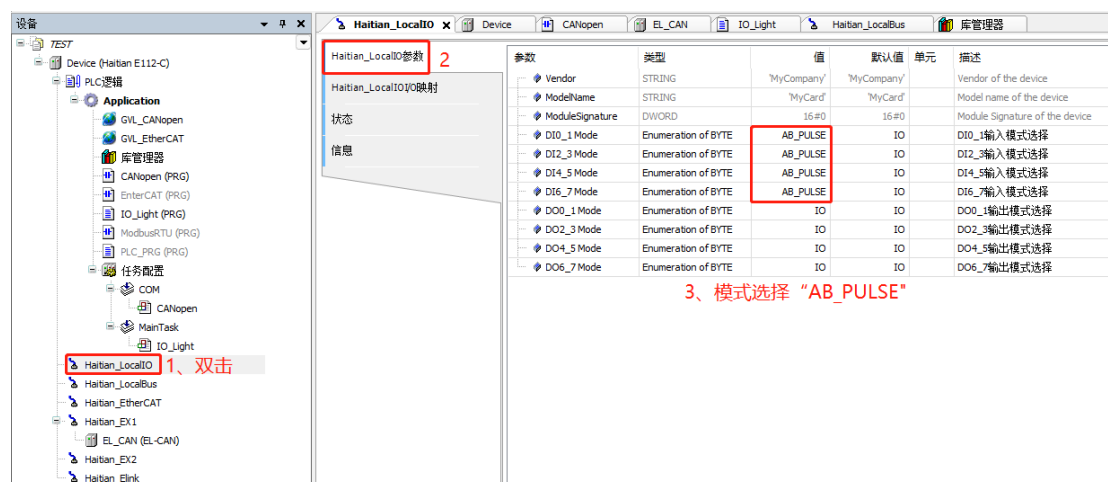


2.13.2 高速输入

作为高速输入时，在软件上的组态步骤如下：

(1) 双击”Haitian_LocalIO”，选择“Haitian_LocalIO 参数”，将对应 DI 口的输入模式设为

“AB_PULSE”。



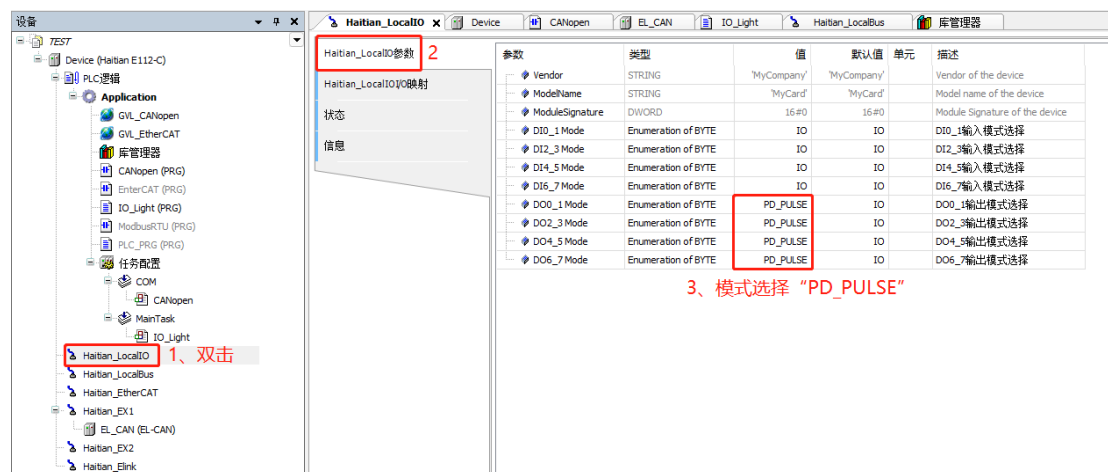
(2) 设置完成后，在“Haitian_LocalIO I/O 映射”下就可以监控当前本地 IO 点的状态。



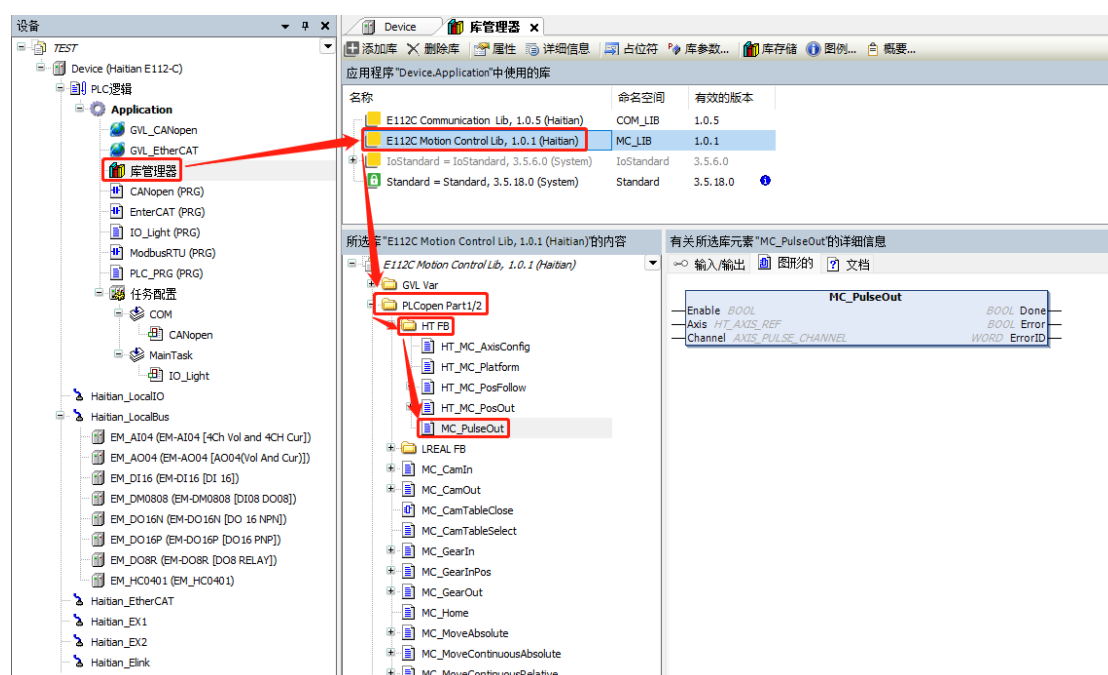
2.13.3 高速输出

作为高速输出时，在软件上的组态步骤如下：

(1) 双击”Haitian_LocalIO”，选择“Haitian_LocalIO 参数”，将对应 DO 口的输出模式设为“PD_PULSE”。



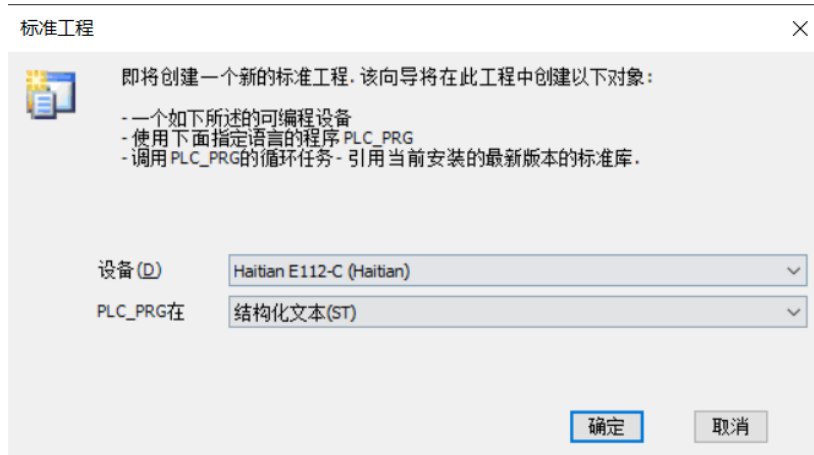
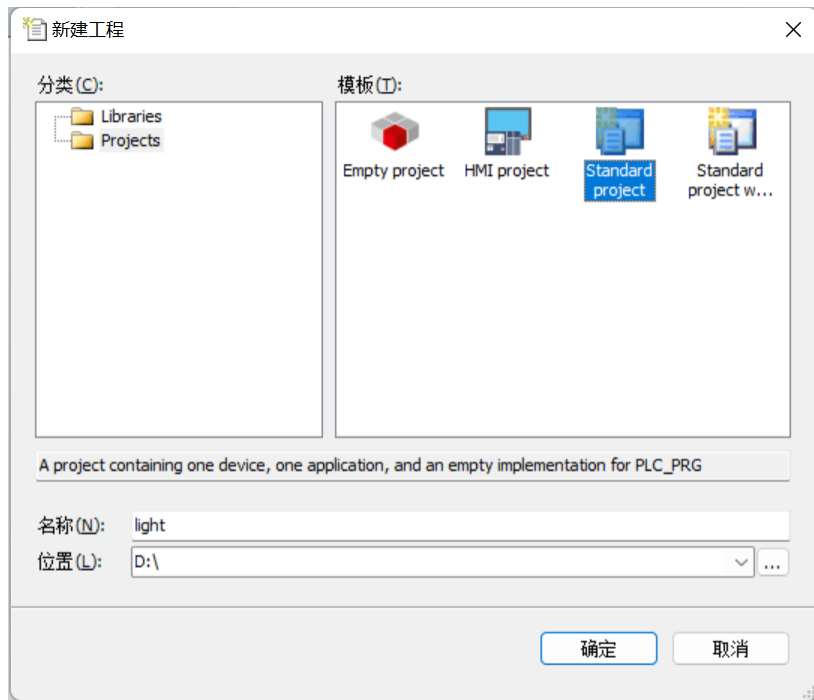
(2) 设置完成后，可通过调用库管理器中的“MC_PulseOut”模块，来控制脉冲输出。



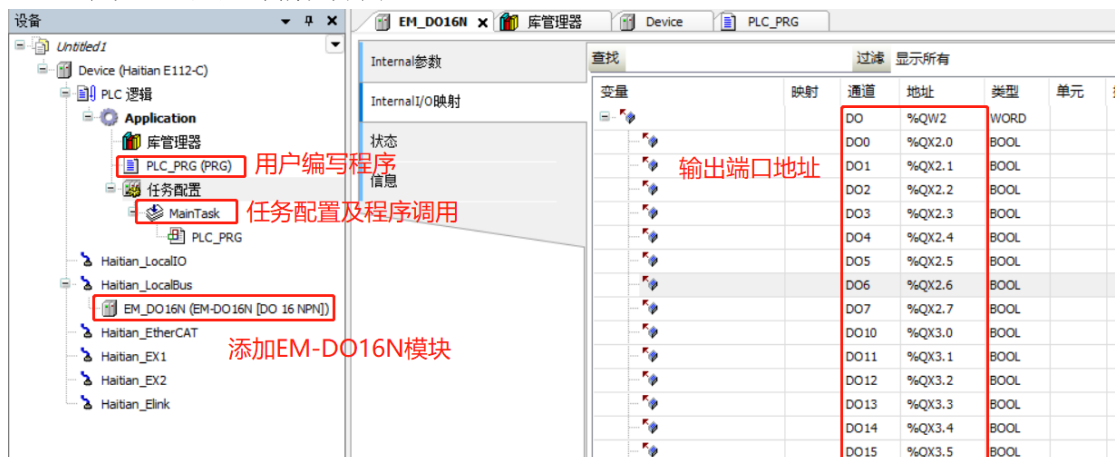
2.14 用 CODESYS 编写一个跑马灯样例工程

(1) 启动 CODESYS 编程环境

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

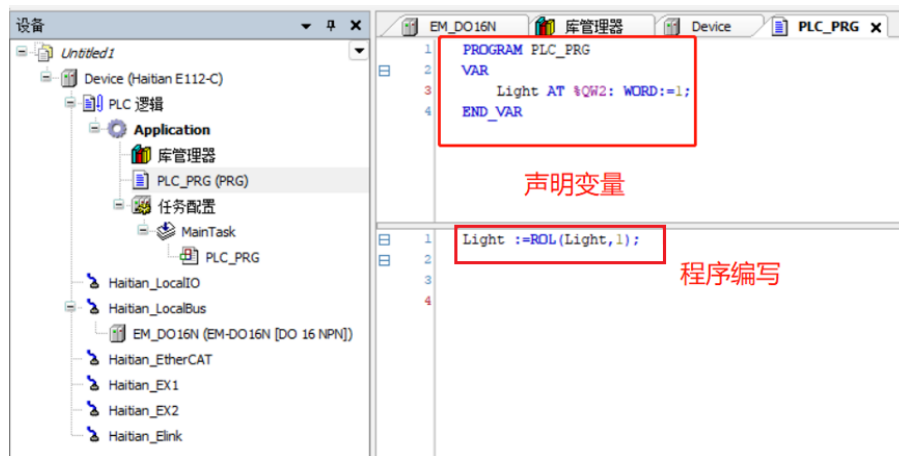


(2) 系统组态配置与编程界面

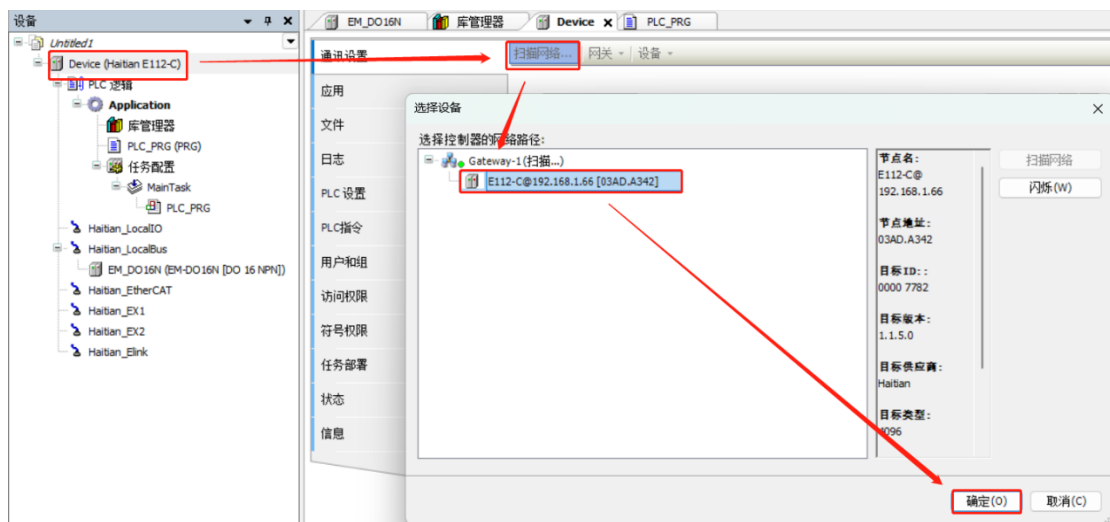


(3) ST 语言编写“跑马灯样例程序”

双击打开“PLC_PRG”程序组织单元



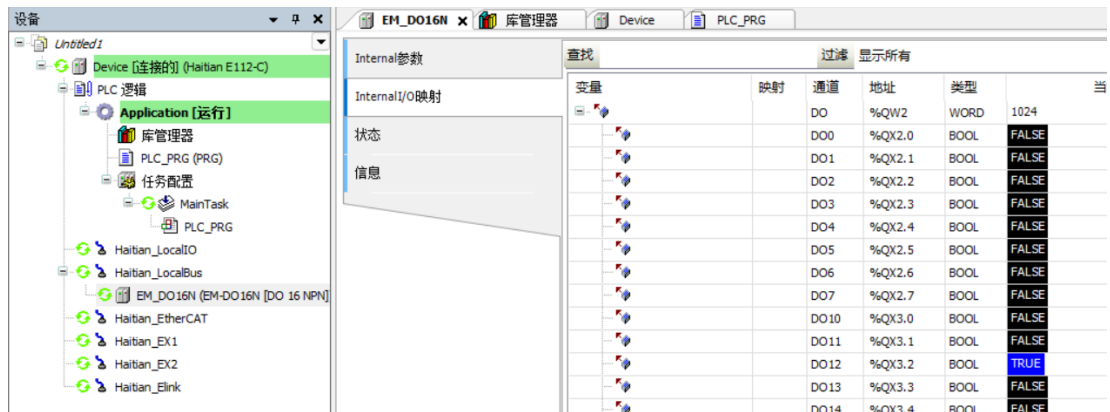
(4) 编译登录



点击“登录”



(5) 下载完成后，启动运行 PLC



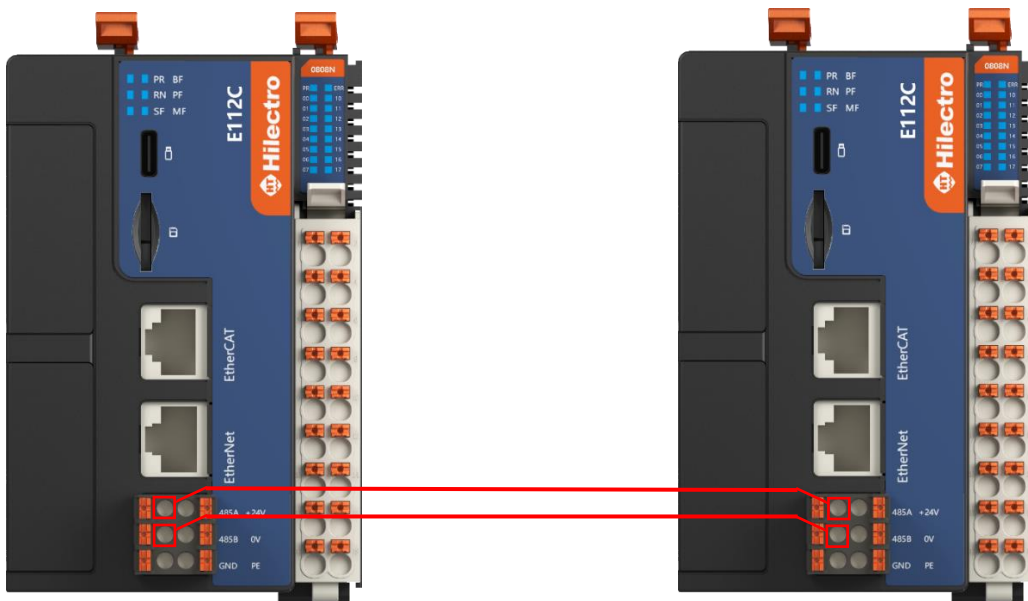
第三章 通讯配置

3.1 ModBus RTU 通讯设置

E112C 控制器本体支持 2 路 RTU 通讯，可通过扩展 EL-485 选配卡再增加 2 路 RTU 通讯，任意一路 RTU 都支持主从配置。

3.1.1 Modbus 接线配置

两台 E112C 控制器分别作为 Modbus 主站和 Modbus 从站，将两台控制器主控端子 485A、485B 接口对应相连接，如图所示。



3.1.2 Modbus 主站配置

(1) 打开 CODESYS 工程，双击“Device”，在“E112C 参数”中修改参数，ModbusRtuSlave[0] 和 ModbusRtuSlave[1] 分别对应 ModbusRTU 的两个通道的从站使能，值为 TRUE 时该通道视为从站开启。根据需求将作为主站的设备中的对应通道改为 FALSE。RS485_1_Baud 和 RS485_2_Baud 分别对应两个通道的波特率，根据需求修改相应波特率。登录运行程序后，退出登录，由于改波特率配置需要重启生效，请将 E112C 控制器重启上电。

1、双击

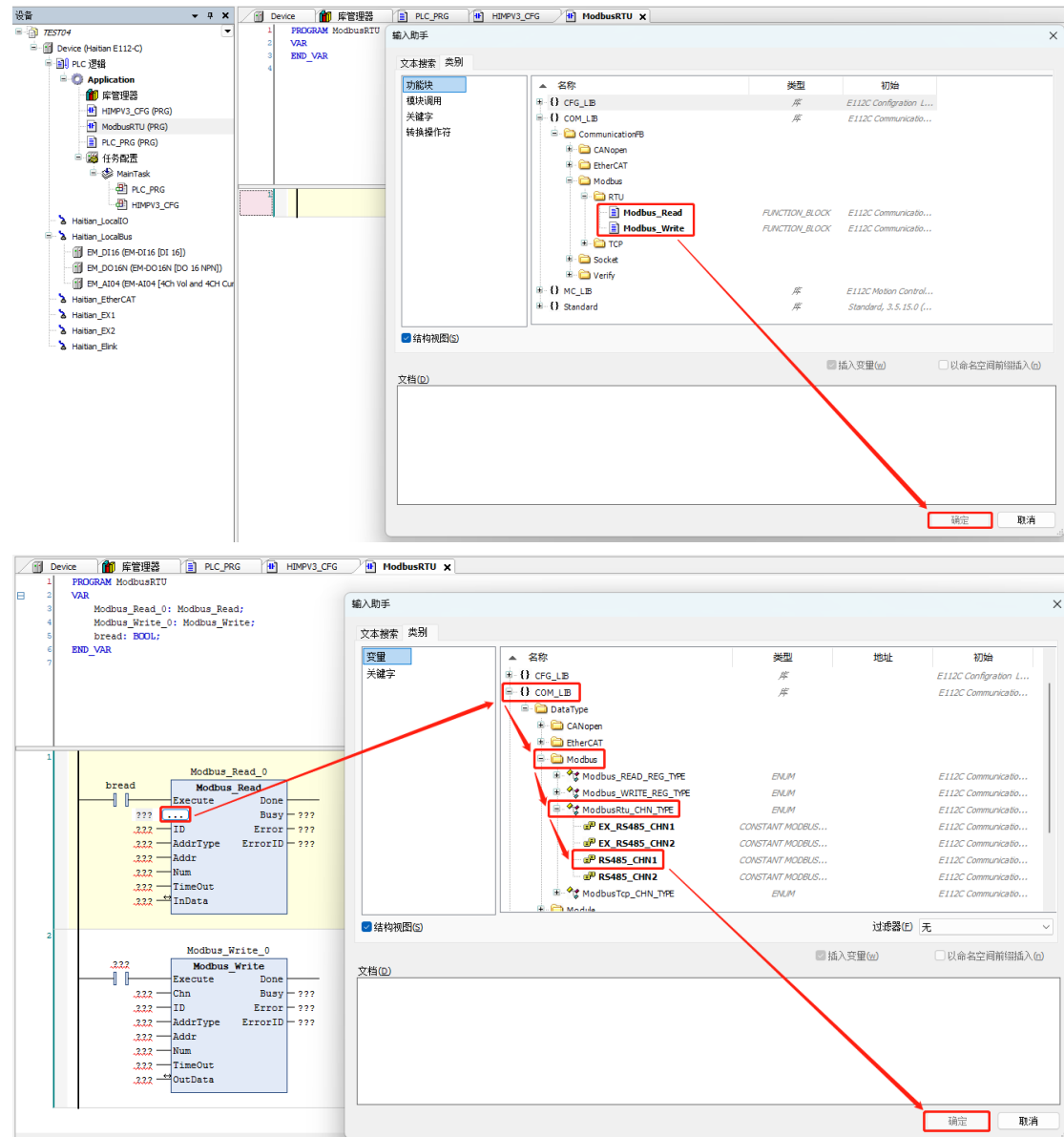
2、进入参数配置

3、设置对应通道波特率

4、关闭从站使能

参数	类型	值	默认值	单元	描述
Fireware Version	STRING				软件版本
Compile Date	STRING				编译日期
Compile Time	STRING				编译时间
Hardware Version	STRING				硬件版本
CodeSys XML Version	STRING				CodeSys XML 版本
CodeSys COM LIB Version	STRING				CodeSys 通讯库版本
CodeSys MC LIB Version	STRING				CodeSys 运动库版本
OSCPUUsage	Byte				CPU 负载百分比
ip	ARRAY [0..3] OF BYTE				本体网口 IP 地址 (修改后掉电生效)
ip[0]	BYTE	192	192		本体网口 IP 地址 (修改后掉电生效)
ip[1]	BYTE	168	168		本体网口 IP 地址 (修改后掉电生效)
ip[2]	BYTE	1	1		本体网口 IP 地址 (修改后掉电生效)
ip[3]	BYTE	67	66		本体网口 IP 地址 (修改后掉电生效)
netmask	ARRAY [0..3] OF BYTE				本体网口子网掩码 (修改后掉电生效)
gateway	ARRAY [0..3] OF BYTE				本体网口网关 (修改后掉电生效)
RS485_1_Baud	Enumeration of UDINT	9600Baud	9600Baud		RS485-1 波特率 (修改后掉电生效)
RS485_2_Baud	Enumeration of UDINT	9600Baud	9600Baud		RS485-2 波特率 (修改后掉电生效)
ModbusSlaveAddr	Byte	1	1		Modbus 从站地址 (1~247)
ModbusRtuSlave	ARRAY [0..1] OF BOOL				Modbus 从站使能
ModbusRtuSlave[0]	BOOL	FALSE	TRUE		Modbus 从站使能
ModbusRtuSlave[1]	BOOL	FALSE	FALSE		Modbus 从站使能
ModbusTcpSlaveNum	Byte	1	1		本体 ModbusTcp 从站数量 (0~2)

(2) 右击 Application 添加对象 POU，新建一个梯形逻辑图 (LD) 工程，并在程序中右键插入网络，在网络中分别插入 “Modbus_Read”、“Modbus_Write”两个功能块，根据所用到的通道配置四个功能块参数，修改两个模块中 ID 端口为从站地址 ID（示例里为“1”），同时需要修改 TimeOut 端口的超时时间（设置值应该大于任务周期时间，根据连接设备回复的快慢设置合理的超时时间。示例里为“50ms”）。将程序添加进任务中，登录并下载程序。



The screenshot shows the Siemens SIMATIC Manager interface. On the left, the project tree is visible, with 'ModbusRTU' selected under the 'HIMPV3_CFG' folder. The main workspace displays the ladder logic for the 'ModbusRTU' program. The 'Modbus_Read_0' function block is connected to the 'bread' variable, and the 'Modbus_Write_0' function block is connected to the 'bwrite' variable. The 'COM_LIB' library is used for both blocks. The 'Modbus_Read_0' block has inputs for 'Execute', 'Done', 'Chn', 'Busy', 'ID', 'Error', 'AddrType', 'ErrorID', 'Addr', 'Num', 'TimeOut', and 'InData'. The 'Modbus_Write_0' block has inputs for 'Execute', 'Done', 'Chn', 'Busy', 'ID', 'Error', 'AddrType', 'ErrorID', 'Addr', 'Num', 'TimeOut', and 'OutData'.

配置

优先级(1..3):

2

类型

循环

间隔(如t#200ms)

10

小于Timeout

ms

看门狗

☐ 使能

时间(如t#200ms)

ms

灵敏度

1

✚ 添加调用

✕ 移除调用

🔧 更改调用

⬆ 上移

⬇ 下移

🔗 打开POU

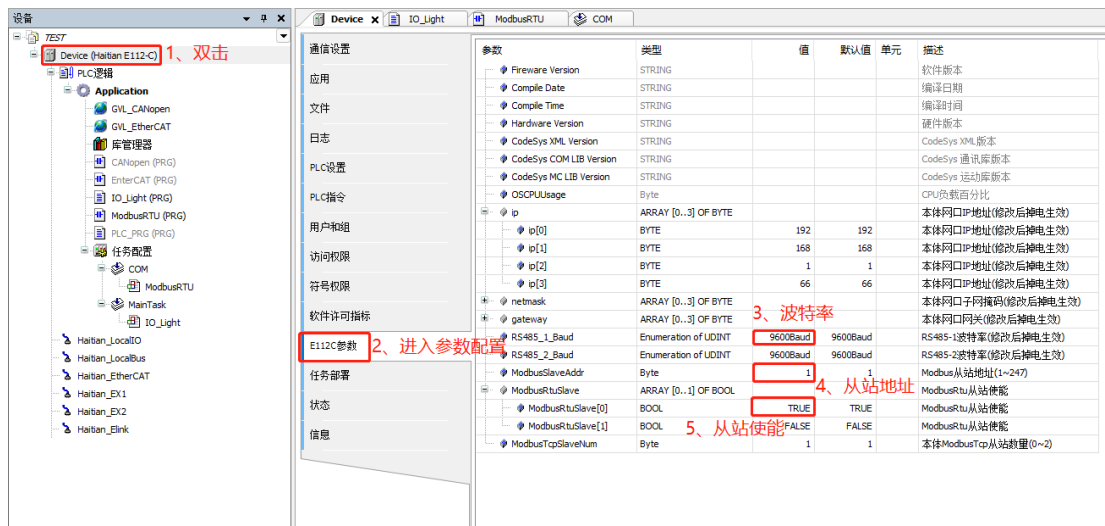
POU

ModbusRTU

注释

3.1.3 Modbus 从站配置

(1) 打开 CODESYS 工程，双击“Device”，在“E112C 参数”中修改参数，ModbusRtuSlave[0] 和 ModbusRtuSlave[1] 分别对应 ModbusRTU 的两个通道的从站使能，根据需求将作为从站的设备中的对应通道改为 TRUE。将 ModbusSlaveAddr 改为所需从站地址。RS485_1_Baud 和 RS485_2_Baud 分别对应两个通道的波特率，根据需求修改相应波特率。登录运行程序后，退出登录，由于改波特率配置需要重启生效，请将 E112C 控制器重启上电。

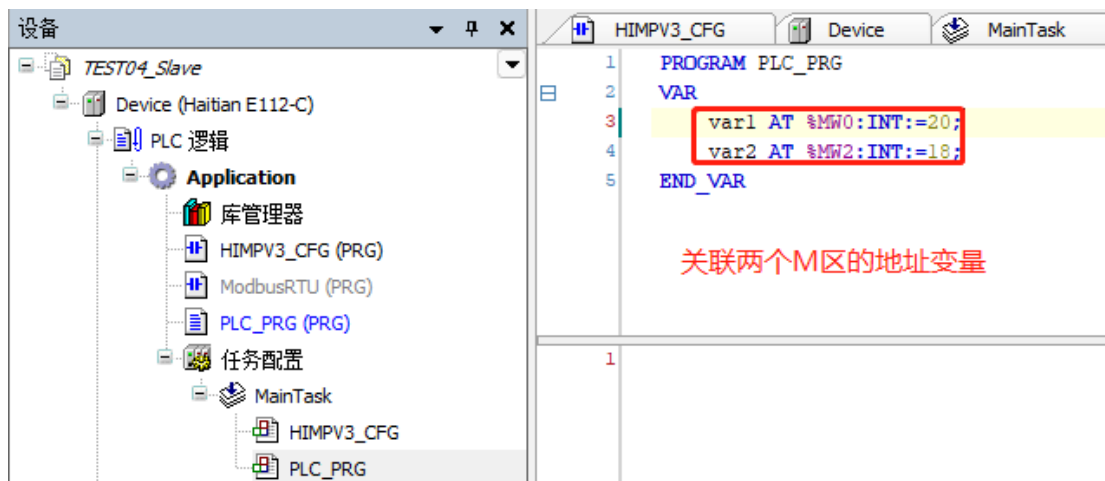


(2) 在 CODESYS 中新建一个 POU 程序，在应用程序声明区域声明存储单元变量，例如：

```
var1 AT %MW0:INT:=20;
```

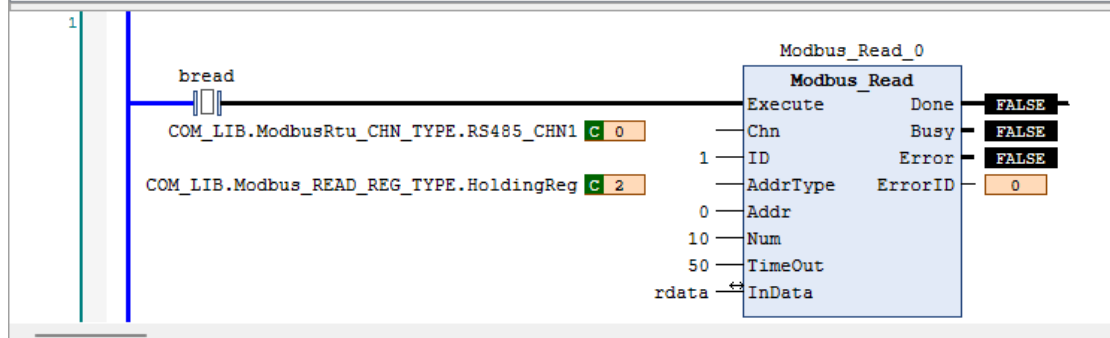
```
var2 AT %MW2:INT:=18;
```

将其添加到“MainTask”任务下随后编译下载程序。

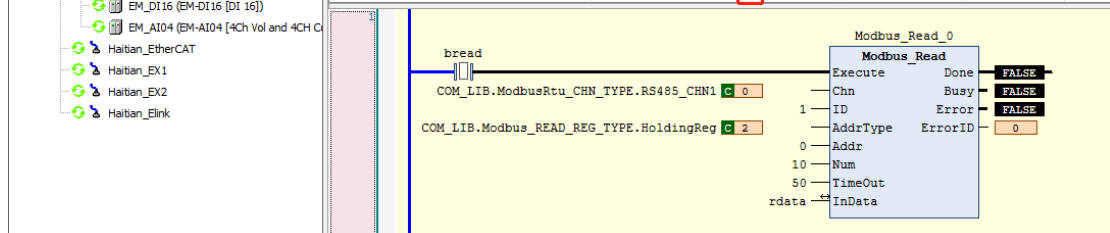


(3) 在 Modbus 主站工程下对从站设备数据进行读操作，对 Modbus_Read 模块的 Execute 引脚发送一个上升沿信号，随后可在“rdata”中发现主站已将从站数据读取到对应位置。对从站设备数据进行写操作，在“wdata”中写入从站所要获取的数据（一个 M 区地址长度为一个字节），对 Modbus_Write 模块的 Execute 引脚发送一个上升沿信号后可在从站工程对应关联地址变量下观察到数据已写入。

Device.Application.ModbusRTU						
表达式	类型	值	准备值	地址	注释	
bread	BOOL	FALSE			当读取模块未使能时，无法读取从站数据	
bwrite	BOOL	FALSE				
rdata	Modbus_READ_DATA					
data	ARRAY [0..199] OF ...					
data[0]	BYTE	0				
data[1]	BYTE	0				
data[2]	BYTE	0				
data[3]	BYTE	0				
data[4]	BYTE	0				
data[5]	BYTE	0				
data[6]	BYTE	0				
data[7]	BYTE	0				
data[8]	BYTE	0				



Device.Application.ModbusRTU						
表达式	类型	值	准备值	地址	注释	
Modbus_Read_0	Modbus_Read					
Modbus_Write_0	Modbus_Write					
bread	BOOL	FALSE			未触发读取模块时，无法获得当前从站的数据	
bwrite	BOOL	FALSE				
rdata	Modbus_READ_DATA					
data	ARRAY [0..199] OF BYTE					
data[0]	BYTE	0				
data[1]	BYTE	0				
data[2]	BYTE	0				
data[3]	BYTE	0				
data[4]	BYTE	0				
data[5]	BYTE	0				
data[6]	BYTE	0				
data[7]	BYTE	0				



设备 TEST04

Device [连接的] (Haitian E112-C)

Application [运行]

库管理器

HIMPV3_CFG (PRG)

ModbusRTU (PRG)

PLC_PRG (PRG)

任务配置

LOCALBUS

HIMPV3_CFG

MainTask

ModbusRTU

PLC_PRG

Haitian_LocalIO

Haitian_LocalBus

EM_DI16 (EM-DI16 [DI 16])

EM_AI04 (EM-AI04 [4Ch Vol and 4Ch C])

Haitian_EtherCAT

Haitian_EX1

Haitian_EX2

Haitian_Elink

Device.Application.ModbusRTU

表达式	类型	值	准备值	地址	注释
Modbus_Read_0	Modbus_Read				
Modbus_Write_0	Modbus_Write				
bread	BOOL	TRUE			
bwrite	BOOL	FALSE			
rdata	Modbus_READ_DATA				
data	ARRAY [0..199] OF BYTE				
data[0]	BYTE	20			
data[1]	BYTE	0			
data[2]	BYTE	18			
data[3]	BYTE	0			

上升沿触发后读取从站的默认值，读取模块的Done端口输出TRUE为读取成功。

Modbus_Read_0

Execute

Done TRUE

Busy FALSE

Error FALSE

ErrorID 0

Chn 0

ID 1

AddrType 0

Addr 0

Num 10

TimeOut 50

rdata InData

设备 TEST04

Device [连接的] (Haitian E112-C)

Application [运行]

库管理器

HIMPV3_CFG (PRG)

ModbusRTU (PRG)

PLC_PRG (PRG)

任务配置

LOCALBUS

HIMPV3_CFG

MainTask

ModbusRTU

PLC_PRG

Haitian_LocalIO

Haitian_LocalBus

EM_DI16 (EM-DI16 [DI 16])

EM_AI04 (EM-AI04 [4Ch Vol and 4Ch C])

Haitian_EtherCAT

Haitian_EX1

Haitian_EX2

Haitian_Elink

Device.Application.ModbusRTU

表达式	类型	值	准备值	地址	注释
Modbus_Read_0	Modbus_Read				
Modbus_Write_0	Modbus_Write				
bread	BOOL	FALSE			
bwrite	BOOL	TRUE			
rdata	Modbus_READ_DATA				
wdata	Modbus_WRITE_DATA				
data	ARRAY [0..99] OF BYTE				
data[0]	BYTE	66			
data[1]	BYTE	0			
data[2]	BYTE	68			

给从站写入两个数据

bwrite 触发上升沿信号

Modbus_Write_0

Execute

Done TRUE

Busy FALSE

Error FALSE

ErrorID 0

Chn 0

ID 1

AddrType 0

Addr 0

Num 10

TimeOut 50

wdata OutData

写入完成

设备 TEST04_Slave

Device [连接的] (Haitian E112-C)

Application [运行]

库管理器

HIMPV3_CFG (PRG)

ModbusRTU (PRG)

PLC_PRG (PRG)

任务配置

MainTask

HIMPV3_CFG

PLC_PRG

Haitian_LocalIO

Haitian_LocalBus

EM_DI16 (EM-DI16 [DI 16])

EM_DI16_1 (EM-DI16 [DI 16])

Haitian_EtherCAT

Haitian_EX1

Haitian_EX2

Haitian_Elink

Device.Application.PLC_PRG

表达式	类型	值	准备值	地址	注释
var1	INT	66		%MW0	
var2	INT	68		%MW2	

写入成功

RETURN

3.1.4 Modbus 从站映射地址

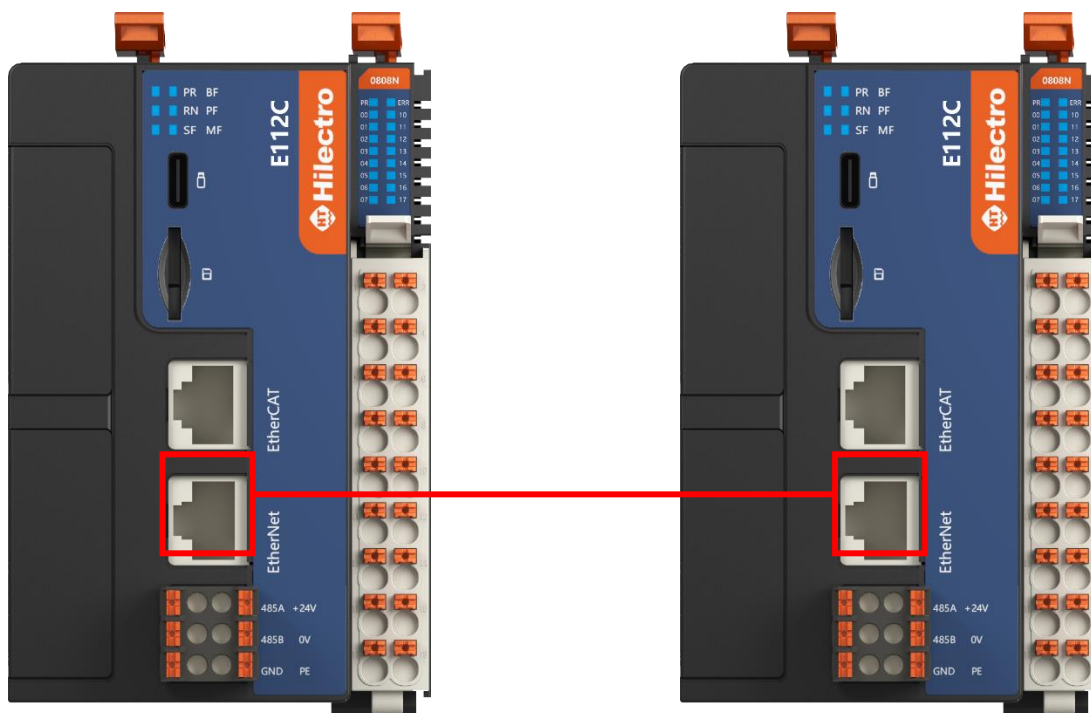
RAM:48K 0x30000000	32K(PLC的M区 , 需要映射到modbus地址)				
	M0—M13999	M14000—M17999	M18000—M21999	M22000—M31999	M32000—M32767
	M用户区	M掉电保存区(用户)	M特殊地址区(非用户)	M掉电保存区(非用户)	预留
	1K:Retain掉电保存区 (无PLC地址)		1K:Persistent Retain掉电保 存区 (无PLC地址)		预留14K
RAM:128K 0x20000000	128K(IO区、数据区)				
	I0—I1023	Q0—Q1023			
	I输入区	Q输出区	数据区		

3.2 ModBus TCP 通讯设置

E112C 控制器本体支持 2 路 Modbus TCP 通讯，可通过扩展 EL-ETH 选配卡再增加 8 路 Modbus TCP 通讯，任意一路 TCP 都支持主从配置。

3.2.1 Modbus 接线配置

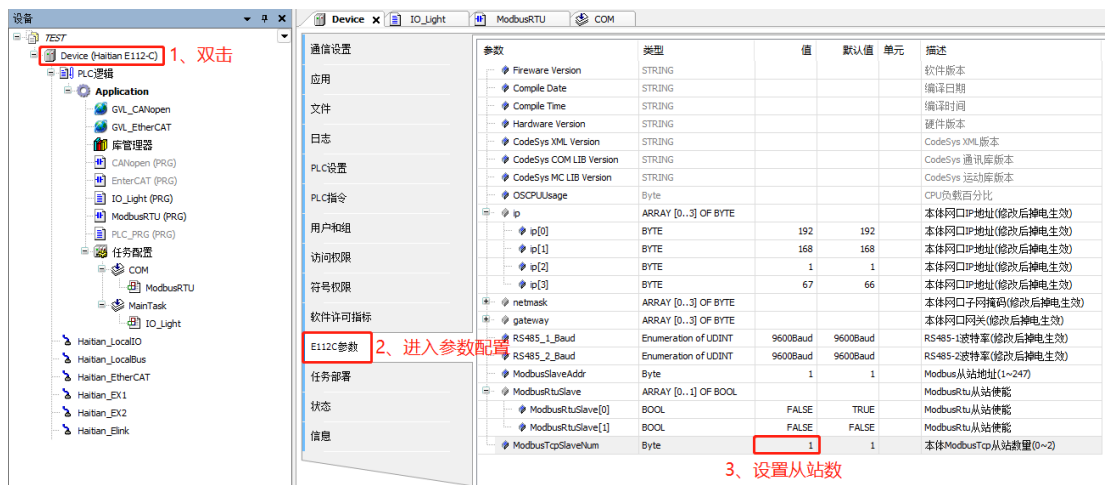
两台 E112C 控制器分别作为 Modbus 主站和 Modbus 从站，将两台控制器 EtherNet 口相连接，如图所示。



3.2.2 Modbus 主站配置

(1) 打开 CODESYS 工程，双击“Device”，在“E112C 参数”中修改参数，根据使用情况设置本体网口 ModbusTCP 从站个数以及扩展网口 ModbusTCP 从站个数，本例只使用本体网口的一个通道作为主站通道，在 ModbusTCPSlaveNum 中写入所需设置的从站个数。配置完成后，请登录并下载程序。

(注意：本体 EtherNET 网口支持 2 个通道，扩展 EtherNET 网口支持 8 个通道，每个网口中主站通道和从站通道总计不能大于该网口最大通道数（例：使用本体网口作 1 个主站时，则本体网口 ModbusTCP 从站个数只能为 0 或 1。）EX 网口同理。)



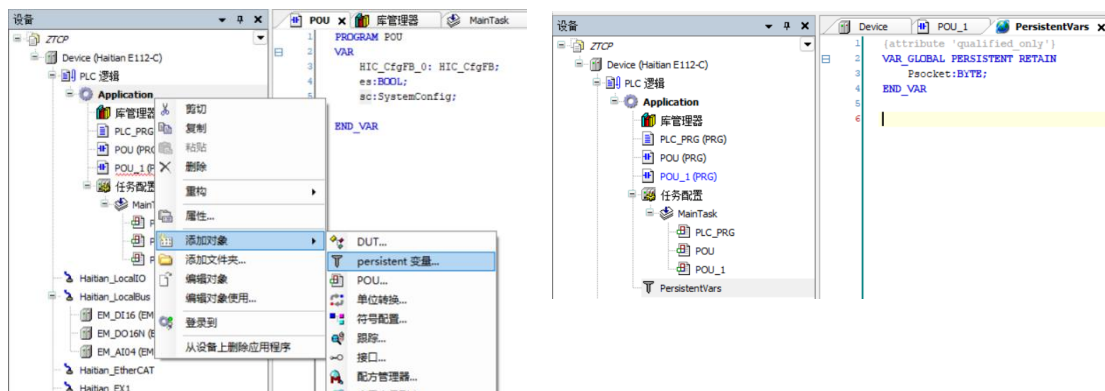
1、双击

2、进入参数配置

3、设置从站数

参数	类型	值	默认值	单元	描述
Fireware Version	STRING				软件版本
Compile Date	STRING				编译日期
Compile Time	STRING				编译时间
Hardware Version	STRING				硬件版本
CodeSys XML Version	STRING				CodeSys XML 版本
CodeSys COM LIB Version	STRING				CodeSys 通讯库版本
CodeSys MC LIB Version	STRING				CodeSys 运动库版本
OSCPUUsage	Byte				CPU 负载百分比
ip	ARRAY [0..3] OF BYTE				本体网口 IP 地址(修改后掉电生效)
ip[0]	BYTE	192	192		本体网口 IP 地址(修改后掉电生效)
ip[1]	BYTE	168	168		本体网口 IP 地址(修改后掉电生效)
ip[2]	BYTE	1	1		本体网口 IP 地址(修改后掉电生效)
ip[3]	BYTE	67	66		本体网口 IP 地址(修改后掉电生效)
netmask	ARRAY [0..3] OF BYTE				本体网口子网掩码(修改后掉电生效)
gateway	ARRAY [0..3] OF BYTE				本体网口网关(修改后掉电生效)
RS485_1_Baud	Enumeration of UDINT	9600Baud	9600Baud		RS485-1 波特率(修改后掉电生效)
RS485_2_Baud	Enumeration of UDINT	9600Baud	9600Baud		RS485-2 波特率(修改后掉电生效)
ModbusSlaveAddr	Byte	1	1		Modbus 从站地址(1~247)
ModbusRtSlave	ARRAY [0..1] OF BOOL				ModbusRt 从站使能
ModbusRtSlave[0]	BOOL	FALSE	TRUE		ModbusRt 从站使能
ModbusRtSlave[1]	BOOL	FALSE	FALSE		ModbusRt 从站使能
ModbusTcpSlaveNum	Byte	1	1		本体 ModbusTcp 从站数量(0~2)

(2) 新建一个 POU 程序，在程序中添加如图所示功能块及代码，IP 设置为从站 IP（示例里为“192.168.1.154”）。将程序添加进任务中，登录并下载程序。

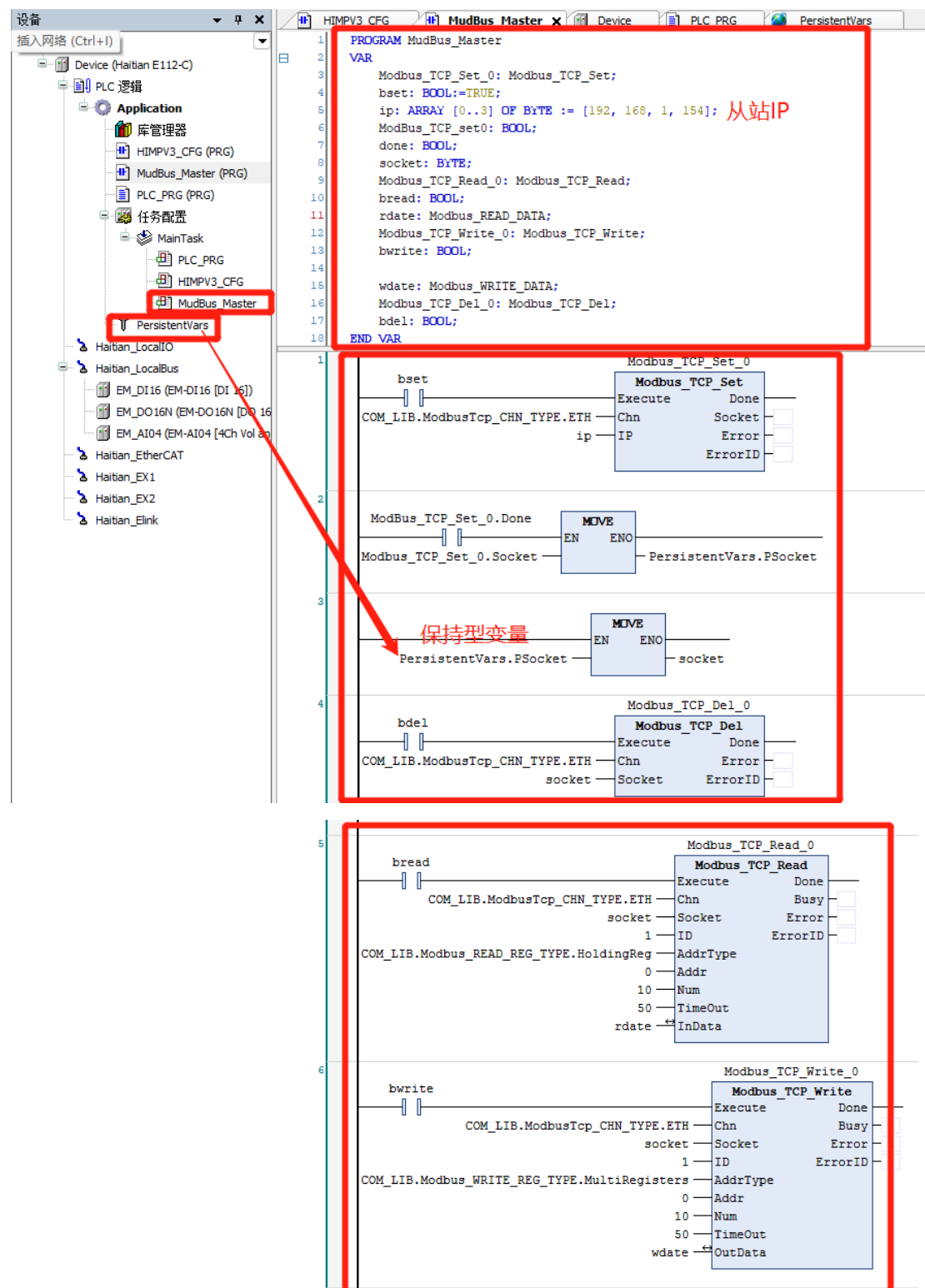


POU_1

```

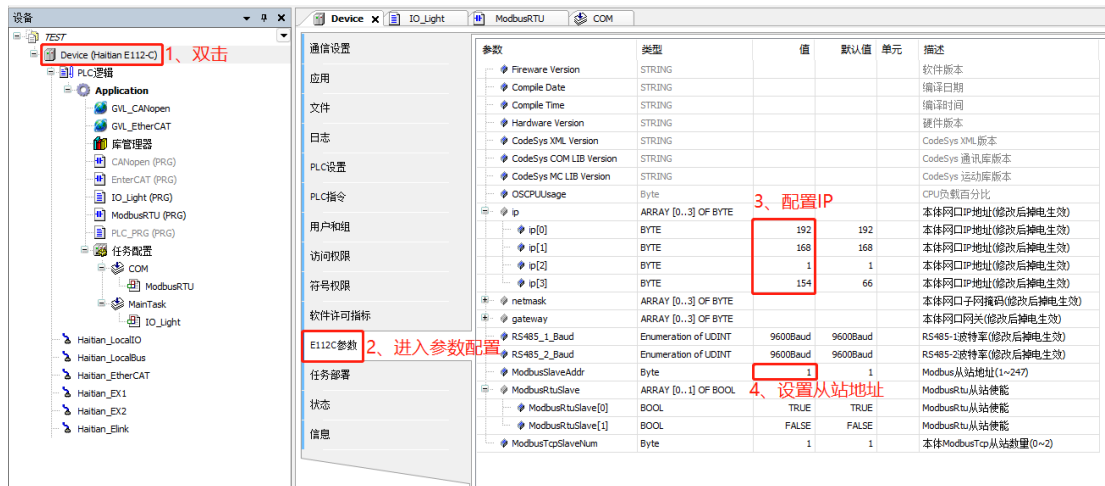
1 (attribute 'qualified_only')
2 VAR_GLOBAL PERSISTENT RETAIN
3   Packet:BYTE;
4 END_VAR
5
6

```



3.2.3 Modbus 从站配置

(1) 打开 CODESYS 工程，双击“Device”，在“E112C 参数”中修改参数，配置 IP 地址和从站 ID 号与主站功能块设置的从站 IP 和从站 ID 号一致。登录运行程序后，退出登录，由于改配置需要重启生效，请将 E112C 控制器重启上电。

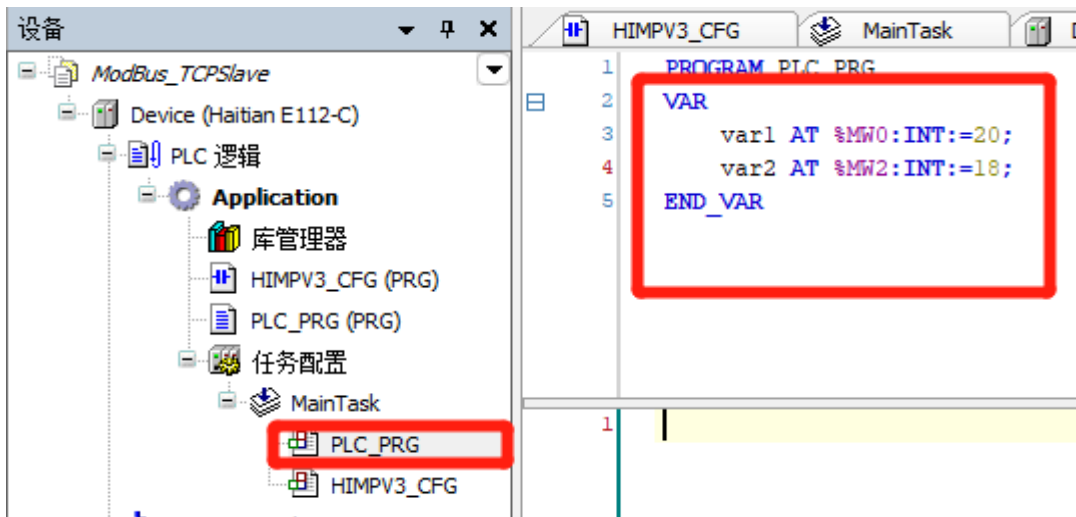


(2) 在 CODESYS 中新建一个 POU 程序，在应用程序声明区域声明存储单元变量，例如：

```
var1 AT %MW0:INT:=20;
```

```
var2 AT %MW2:INT:=18;
```

将其添加到“MainTask”任务下随后编译下载程序。



(4) 在 Modbus 主站工程下对从站设备数据进行读操作，对 Modbus_TCP_Read 模块的 Execute 引脚发送一个上升沿信号，随后可在“rdata”中发现主站已将主站数据读取到对应位置。对从站设备数据进行写操作，在“wdata”中写入从站所要获取的数据（一个 M 区地址长度为一个字节），对 Modbus_TCP_Write 模块的 Execute 引脚发送一个上升沿信号后可在从站工程对应关联地址变量下观察到数据已写入。

设备 test003

Device [连接的] (Haitian E112-C)

PLC 逻辑

Application [运行]

GVL

库管理器

HIMPV3_CFG (PRG)

Modbus_TCP_Master (PRG)

PLC_PRG (PRG)

任务配置

MainTask

PLC_PRG

HIMPV3_CFG

Modbus_TCP_Master

Haitian_LocalIO

Haitian_LocalBus

Haitian_EtherCAT

Haitian_EX1

Haitian_EX2

Device

Modbus_TCP_Master

库管理器

任务配置

HIMPV3_CFG

PLC_PRG

表达式

表达式	类型	值	准备值	地址	注释
* ip	ARRAY [0..3] OF BYTE				
* rdata	Modbus_READ_DATA				
data	ARRAY [0..199] OF ...				
data[0]	BYTE	0			
data[1]	BYTE	0			
data[2]	BYTE	0			
data[3]	BYTE	0			

Modbus_TCP_Read_0

bread

Modbus_TCP_Read

Execute Done FALSE

socket 2 Socket Busy FALSE

1 ID Error FALSE

2 AddrType ErrorID 0

0 Addr

10 Num

100 Timeout

rdata InData

Modbus_TCP_Write_0

bwrite

Modbus_TCP_Write

Execute Done FALSE

socket 2 Socket Busy FALSE

1 ID Error FALSE

1 AddrType ErrorID 0

0 Addr

10 Num

100 Timeout

wdata OutData

未触发读取模块时，无法获得当前从站数据。

设备 test003

Device [连接的] (Haitian E112-C)

PLC 逻辑

Application [运行]

GVL

库管理器

HIMPV3_CFG (PRG)

Modbus_TCP_Master (PRG)

PLC_PRG (PRG)

任务配置

MainTask

PLC_PRG

HIMPV3_CFG

Modbus_TCP_Master

Haitian_LocalIO

Haitian_LocalBus

Haitian_EtherCAT

Haitian_EX1

Haitian_EX2

Device

Modbus_TCP_Master

库管理器

任务配置

HIMPV3_CFG

PLC_PRG

表达式

表达式	类型	值	准备值	地址	注释
* ip	ARRAY [0..3] OF BYTE				
* rdata	Modbus_READ_DATA				
data	ARRAY [0..199] OF ...				
data[0]	BYTE	20			
data[1]	BYTE	0			
data[2]	BYTE	18			
data[3]	BYTE	0			

Modbus_TCP_Read_0

bread

Modbus_TCP_Read

Execute Done TRUE

socket 2 Socket Busy FALSE

1 ID Error FALSE

2 AddrType ErrorID 0

0 Addr

10 Num

100 Timeout

rdata InData

Modbus_TCP_Write_0

bwrite

Modbus_TCP_Write

Execute Done FALSE

socket 2 Socket Busy FALSE

1 ID Error FALSE

1 AddrType ErrorID 0

0 Addr

10 Num

100 Timeout

wdata OutData

上升沿触发后，读取到从站默认值。读取模块Done引脚显示读操作已经完成。

test003

Device [连接的] (Haitian E112-C)

Application [运行]

库管理器

HIMPV3_CFG (PRG)

Modbus_TCP_Master (PRG)

PLC_PRG (PRG)

任务配置

MainTask

PLC_PRG

HIMPV3_CFG

Modbus_TCP_Master

Haitian_LocalIO

Haitian_LocalBus

Haitian_EtherCAT

Haitian_EX1

Haitian_EX2

Device

Modbus_TCP_Master

库管理器

任务配置

HIMPV3_CFG

P

Device.Application.Modbus_TCP_Master

表达式	类型	值	准备值	地址	注释
bwrite	BOOL	TRUE			
wdata	Modbus_WRITE_DATA				
data	ARRAY [0..99] OF B...				
data[0]	BYTE	66			
data[1]	BYTE	0			
data[2]	BYTE	66			
data[3]	BYTE	0			

写入两个数据给从站

5

Modbus_TCP_Read_0

bread

socket 2

1 ID

2 AddrType

0 Addr

10 Num

100 TimeOut

rdata InData

6

Modbus_TCP_Write_0

bwrite

socket 2

1 ID

1 AddrType

0 Addr

10 Num

100 TimeOut

wdata OutData

提供上升沿信号

写操作完成

test004

Device [连接的] (Haitian E112-C)

PLC 逻辑

Application [运行]

库管理器

HIMPV3_CFG (PRG)

PLC_PRG (PRG)

任务配置

MainTask

HIMPV3_CFG

PLC_PRG

Haitian_LocalIO

Haitian_LocalBus

Haitian_EtherCAT

Haitian_EX1

Haitian_EX2

PLC_PRG

库管理器

Device

HIMPV3_CFG

Haitian_LocalBus

Device.Application.PLC_PRG

表达式	类型	值	准备值	地址	注释
var1	INT	66		%MW0	
var2	INT	66		%MW2	

数据已写入

1

RETURN

3.2.4 Modbus 从站映射地址

RAM:48K 0x30000000	32K(PLC的M区 , 需要映射到modbus地址)				
	M0—M13999	M14000—M17999	M18000—M21999	M22000—M31999	M32000—M32767
	M用户区	M掉电保存区(用户)	M特殊地址区(非用户)	M掉电保存区(非用户)	预留
	1K:Retain掉电保存区 (无PLC地址)		1K:Persistent Retain掉电保 存区 (无PLC地址)		预留14K
RAM:128K 0x20000000	128K(IO区、数据区)				
	I0—I1023	Q0—Q1023			
	I输入区	Q输出区	数据区		

3.2.5 PLC 级联通讯 (Elink)

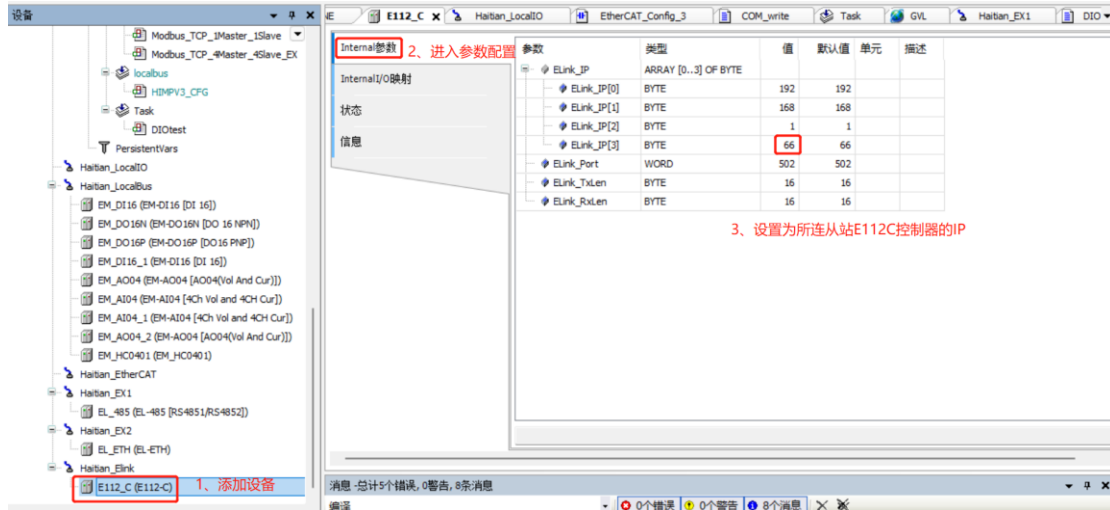
E112C 控制器支持 CPU 级联通讯, 以下根据两台 E112C 控制器举例说明如何建立 Elink 连接。

(1) 作为主站的 E112C 控制器必须扩展 EL-ETH 选配卡, 将作为主站的 E112C 控制器的 EL-ETH 选配卡上的网口通过网线连接到作为从站的 E112C 控制器的本体网口上。随后对配置参数进行设置, 扩展网口的 IP 地址不可与其他设备相同, 从站数量不可为最大值 8。在 Haitian_Elink 下添加 E112C 设备, 并将 Elink_IP 设置成所连从站 E112C 控制器的 IP。

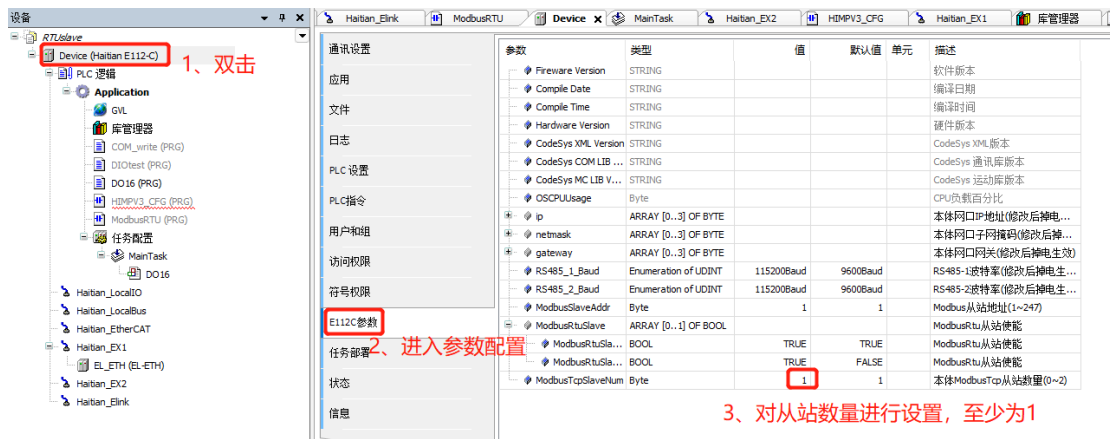
The screenshot shows the 'Internal Parameters' (内部参数) configuration window for the EL_ETH device. The window is divided into several sections:

- Left Panel (Device Tree):** Shows the hierarchy of devices, including 'Haitian_LocalIO', 'Haitian_LocalBus', 'Haitian_EtherCAT', 'Haitian_EX1', 'Haitian_EX2', 'Haitian_Elink', and 'E112_C (E112-C)'. The 'EL_ETH (EL-ETH)' device is highlighted under 'Haitian_Elink'.
- Center Panel (Parameters):** A table of parameters for the selected device.

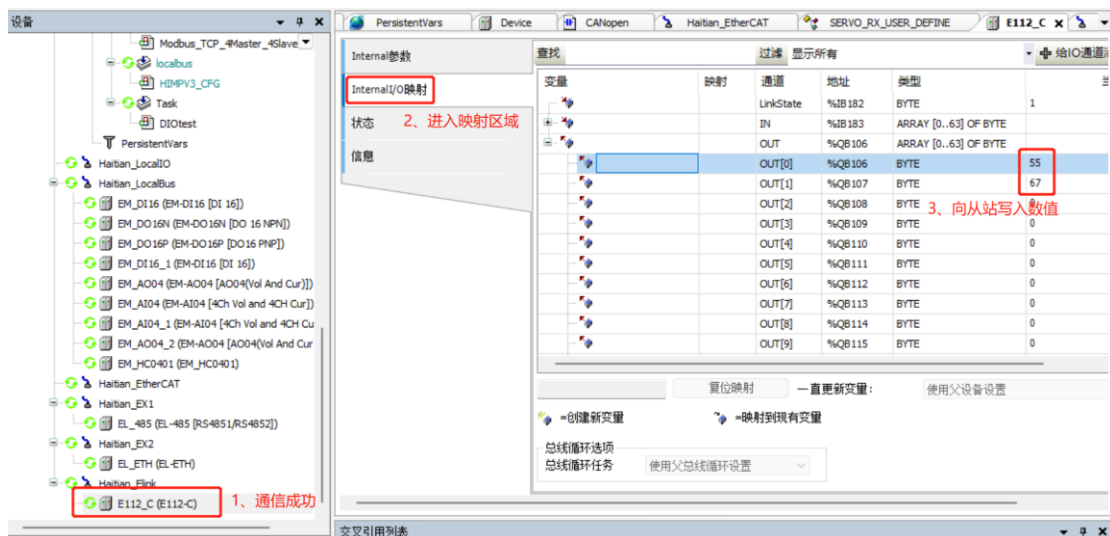
参数	类型	值	默认值	单元	描述
Vendor	STRING	MyCom...	MyC...		Vendor of the device
ModelName	STRING	MyCard	MyC...		Model name of the device
ModuleSignature	DWORD	15#403	15#403		Module Signature of the device
ip	ARRAY [0..3] OF BYTE				EX网口IP地址(修改后掉电生效)
ip[0]	BYTE	192	192		EX网口IP地址(修改后掉电生效)
ip[1]	BYTE	168	168		EX网口IP地址(修改后掉电生效)
ip[2]	BYTE	1	1		EX网口IP地址(修改后掉电生效)
ip[3]	BYTE	67	66		EX网口IP地址(修改后掉电生效)
netmask	ARRAY [0..3] OF BYTE				EX网口子网掩码(修改后掉电生效)
gateway	ARRAY [0..3] OF BYTE				EX网口网关(修改后掉电生效)
ModbusTcpSlaveNumEx	Byte	4	0		EX模块ModbusTcp从站数量
- Right Panel (Status/Log):** Shows the 'Internal Parameters' (内部参数) section with a '2、进入参数配置' (2. Enter parameter configuration) instruction. Below it, there are instructions for modifying the IP address (3、修改扩展网口IP) and the number of slave stations (4、修改从站通道数, 可建立主站数为“8减去从站数量”).



(2) 对作为从站的 E112C 控制器进行参数配置。



(3) 将作为主站的 E112C 控制器的 EL-ETH 选配卡上的网口通过网线连接到作为从站的 E112C 控制器的本体网口上，两台控制器上电（从站需要先上电）。在对应映射地址上可进行数据传输。



3.3.1 E112C 与 Hi 驱动器通信举例

E100 系列控制器作为主站，Hi 驱动器作为从站，驱动器供电（第三代驱动器电源端口 24 号端子为 24V，23 号端子为 0V），通过转换器连接上位机 HIMPV3

打开 HIMPV3, 点击添加“本地驱动器”, 找到驱动器参数组别 FB, 将 FB00 总线类型修改为 EtherCAT, FB73 同步信号源修改为 EtherCAT sync0, FB74 同步周期设置为所需值。E100 控制器 EtherCAT 网口与 Hi 驱动器 IN 网口连接。



FB:Fieldbus configuration

Index	ID	Describe	Localize	Value	Type
651	FB00	Field bus type	通信总线类型选择	3 (CAN bus)	DT_U
652	FB01	Field bus state	总线连接状态	0000 (无值)	DT_V
653	FB02	VARAN bus reconnect	VARAN总线重连	0	DT_U
654	FB03	VARAN bus reload	VARAN总线重启	0	DT_U
686	FB04	CAN bus ID	CAN总线节点ID	7	DT_U
687	FB05	CAN bus baud rate	CAN总线波特率	1 (500kbps)	DT_U
688	FB06	651 FB00 Field bus type	0发送周期	1 ms	DT_U
704	FB07		选择	1 (Little endian)	DT_U
700	FB08		使能	1 (Standard frame)	DT_U
701	FB09			1 (Enabled)	DT_U
689	FB10		点数	0	DT_U
690	FB11		1 ID	2	DT_U
691	FB12		2 ID	3	DT_U
692	FB13		3 ID	4	DT_U
693	FB14		4 ID	5	DT_U
694	FB15		5 ID	6	DT_U
695	FB16		1连接状态	0 (Disconnect)	DT_U
696	FB17		2连接状态	0 (Disconnect)	DT_U
697	FB18		3连接状态	0 (Disconnect)	DT_U
698	FB19		4连接状态	0 (Disconnect)	DT_U
699	FB20		5连接状态	0 (Disconnect)	DT_U
657	FB21	PDO input object number	接收PDO对象数目	0	DT_U
658	FB22	PDO input alive count	总线输入PDO计数	0	DT_U
659	FB23	PDO input 1 index	接收PDO1参数号	0	DT_U
660	FB24	PDO input 1 word number	接收PDO1数据长度	0	DT_U

FB:Fieldbus configuration

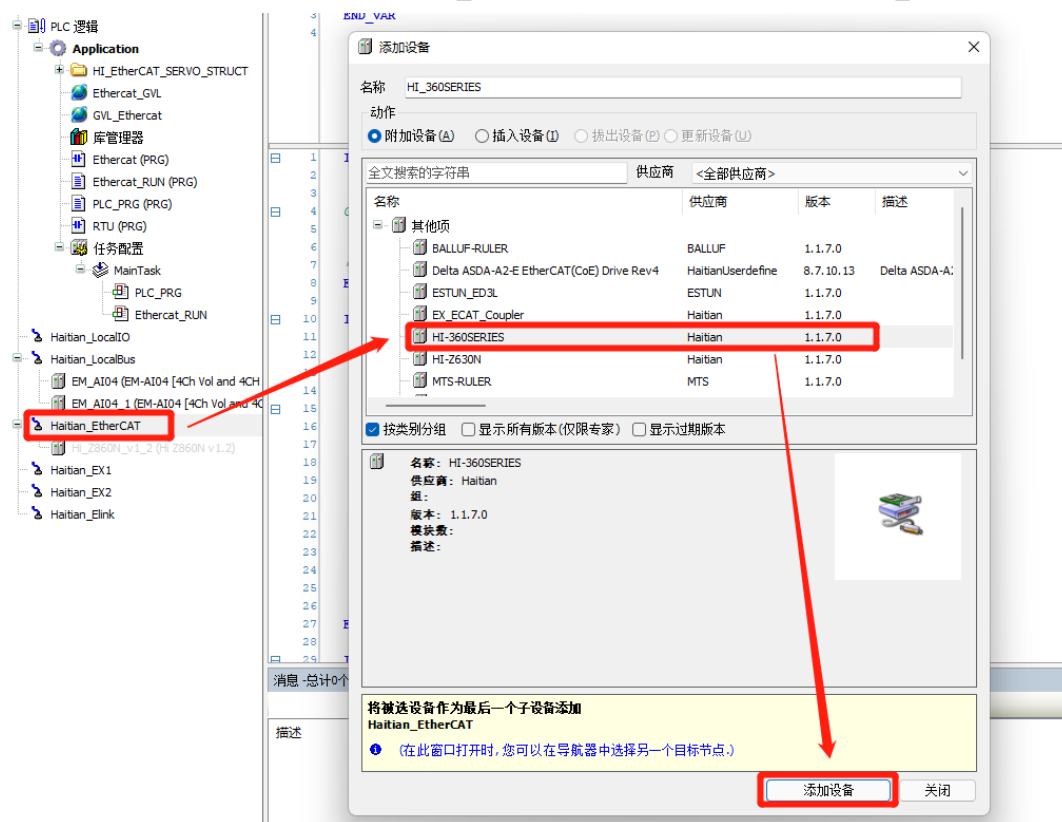
Index	ID	Describe	Localize	Value	Type
710	FB69	PDI error counter and processing unit error c...	过程数据接口错误计数器和处理单元...	0000	DT_U
717	FB70	Set watchdog time manually	手动设置看门狗时间	1 (Enabled)	DT_U
718	FB71	Watchdog time process data	过程数据看门狗超时时间	6 ms	DT_U
722	FB72	Synchronization status	同步状态	0010 (Sync timer out)	DT_U
723	FB73	Source for sync signal	同步信号源	1 (CANsync function module)	DT_U
724	FB74	Sync interval	同步周期	2000 us(2 ms)	DT_U
726	FB76			0.00 us	DT_U
727	FB77			0.00 us	DT_U
728	FB78			0	DT_U
729	FB79		总线同步信号和SM2事件...	1 (Enabled)	DT_U
730	FB80		的SM2事件早于应用	0.00 us	DT_U
734	FB85		限	30	DT_U

FB:Fieldbus configuration

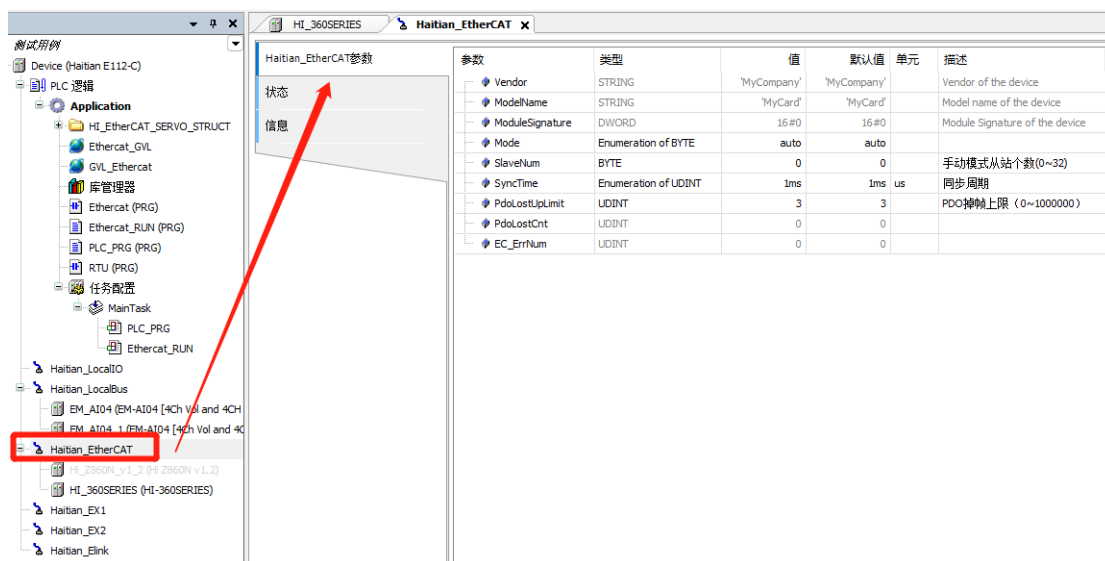
Index	ID	Describe	Localize	Value	Type
716	FB69	PDI error counter and processing unit error c...	过程数据接口错误计数器和处理单元...	0000	DT_U
717	FB70	Set watchdog time manually	手动设置看门狗时间	1 (Enabled)	DT_U
718	FB71	Watchdog time process data	过程数据看门狗超时时间	6 ms	DT_U
722	FB72	Synchronization status	同步状态	0040 (Sync tolerance error)	DT_U
723	FB73	Source for sync signal	同步信号源	1 (CANsync function module)	DT_U
724	FB74	Sync interval	同步周期	1000 us(1 ms)	DT_U
731	FB75	sync tolerance	同步允许偏差	13.33 us	DT_U
726	FB76			0.00 us	DT_U
727	FB77			0.00 us	DT_U
728	FB78			0	DT_U
729	FB79		总线同步信号和SM2事件...	1 (Enabled)	DT_U
730	FB80		的SM2事件早于应用	0.00 us	DT_U
734	FB85			30	DT_U
735	FB86			0	DT_U
761	FB87	CAN special mode	CAN总线特殊模式	0 (Normal mode)	DT_U

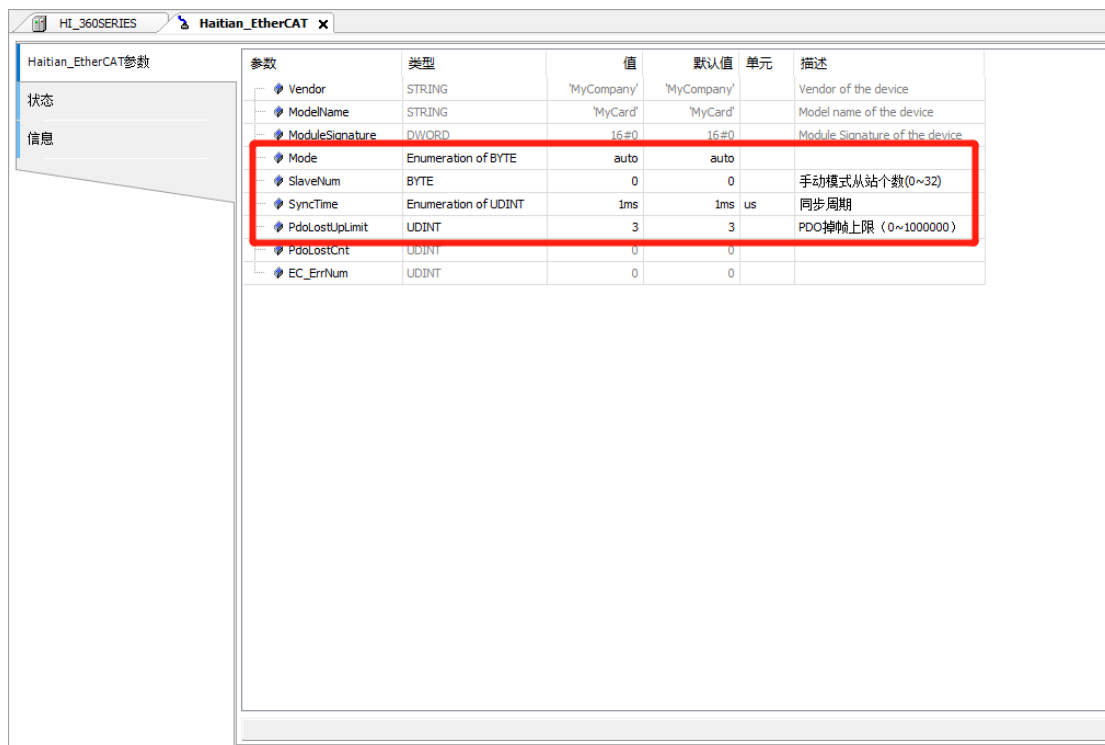
3.3.1.3 EtherCAT 参数配置

(1) 打开 CODESYS 软件, 在左侧 Hatian_EtherCAT 栏右键添加设备, 选择 HI_360SERIES。

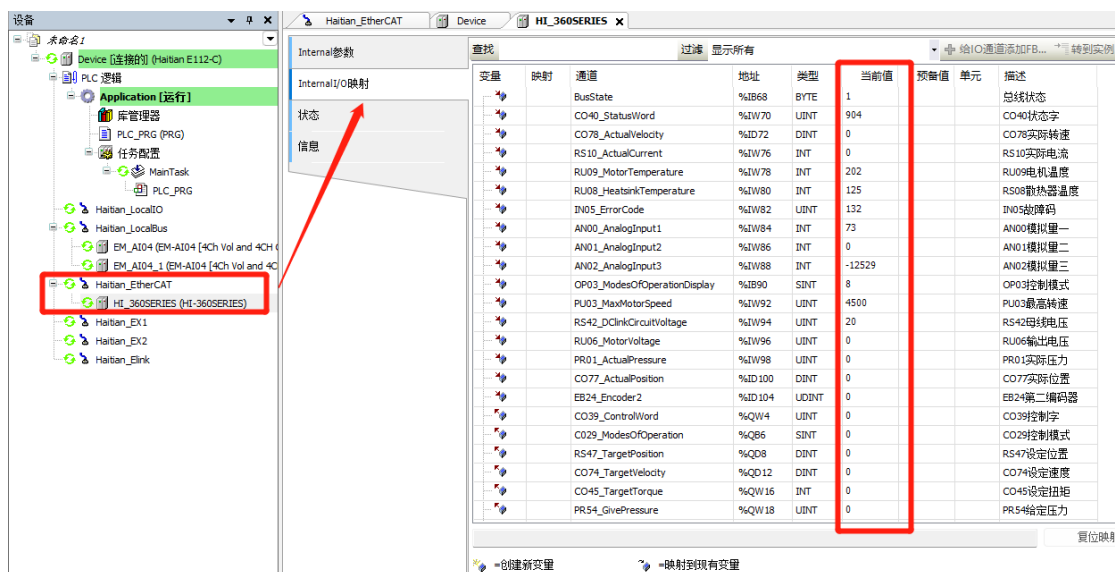


(2) 双击 Hatian_EtherCAT 栏, 在弹出的窗口选择 Haitian_EtherCAT 参数栏, Mode 默认为 auto 模式, 此模式下无需更改 SlaveNum, 根据需求更改同步周期 (此处同步周期要与驱动器同步周期相同), 本例采用默认配置。





(3) 登录控制器并下载程序，运行程序后 HI_360SERIES 处于绿色运行状态，PDO 持续收发，双击 HI_360SERIES，点击 Internal I/O 映射栏，若有数据读到，说明通讯成功。

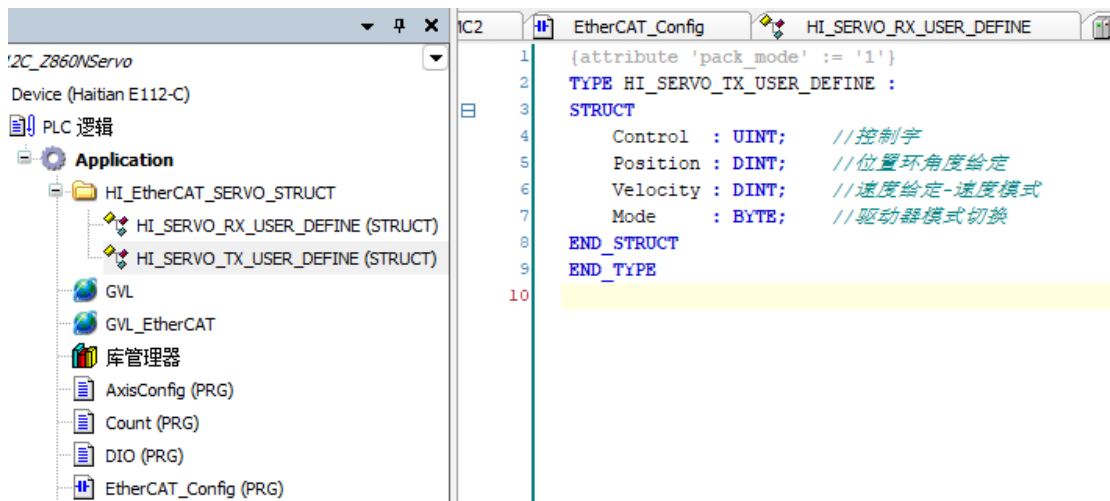
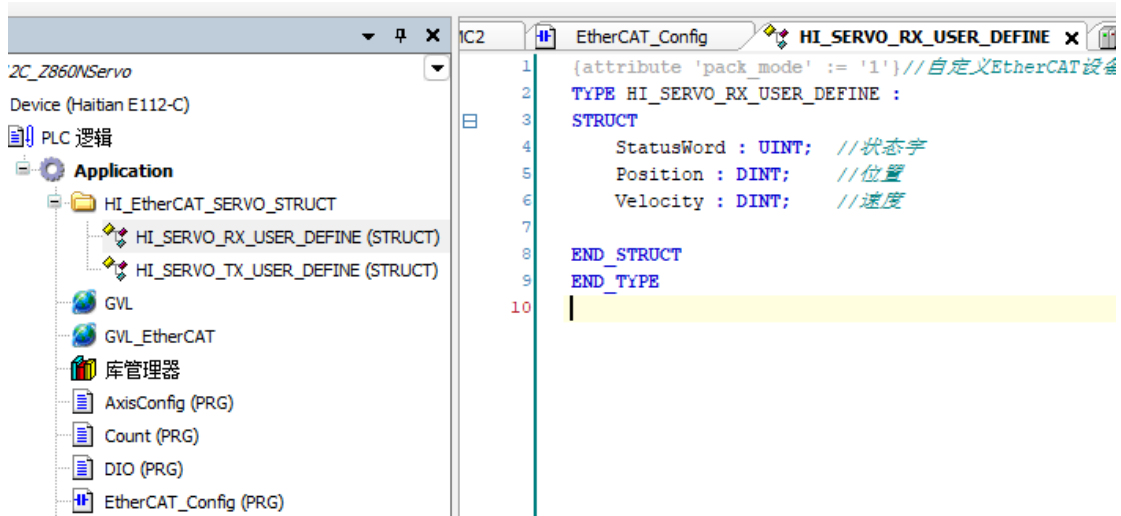
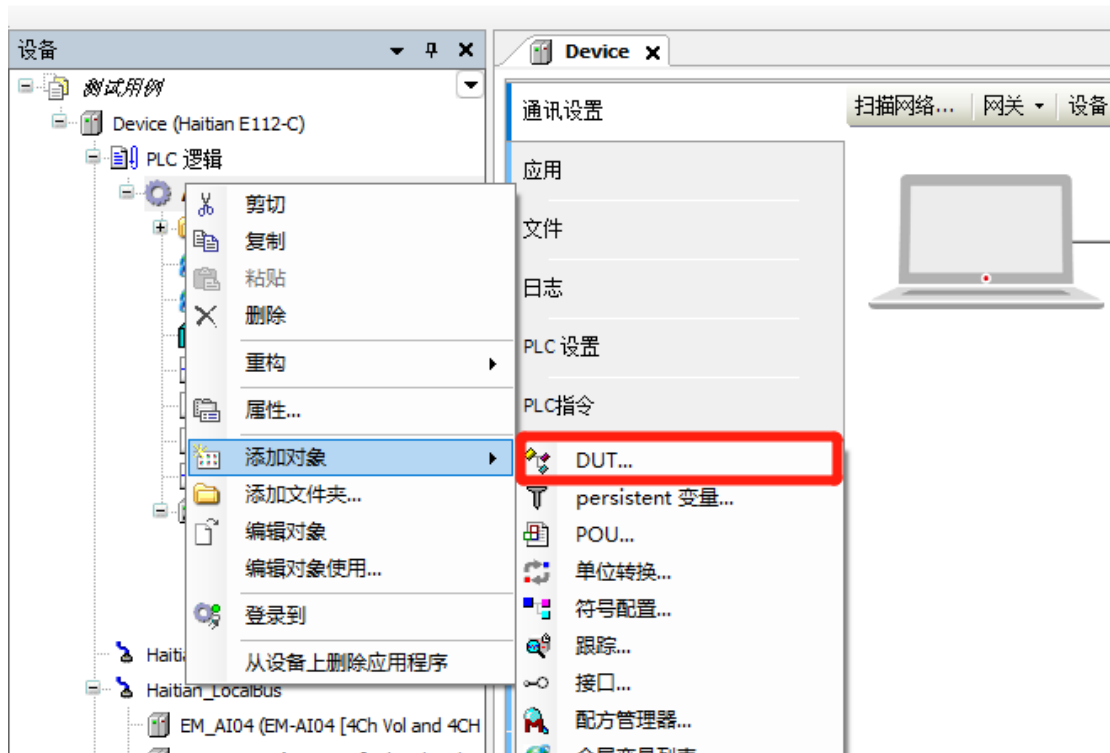


3.3.2 EtherCAT 单轴通讯控制

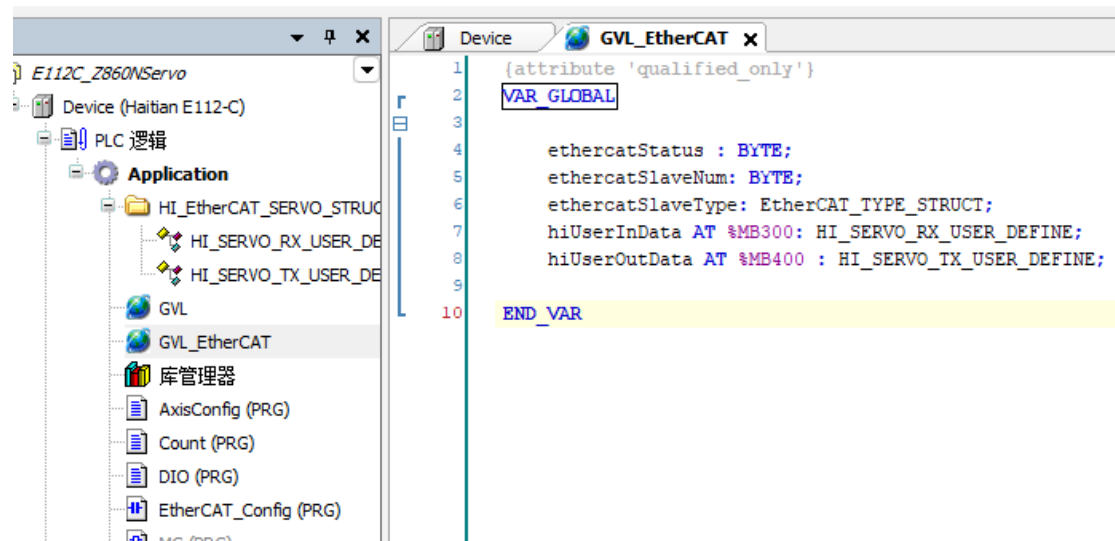
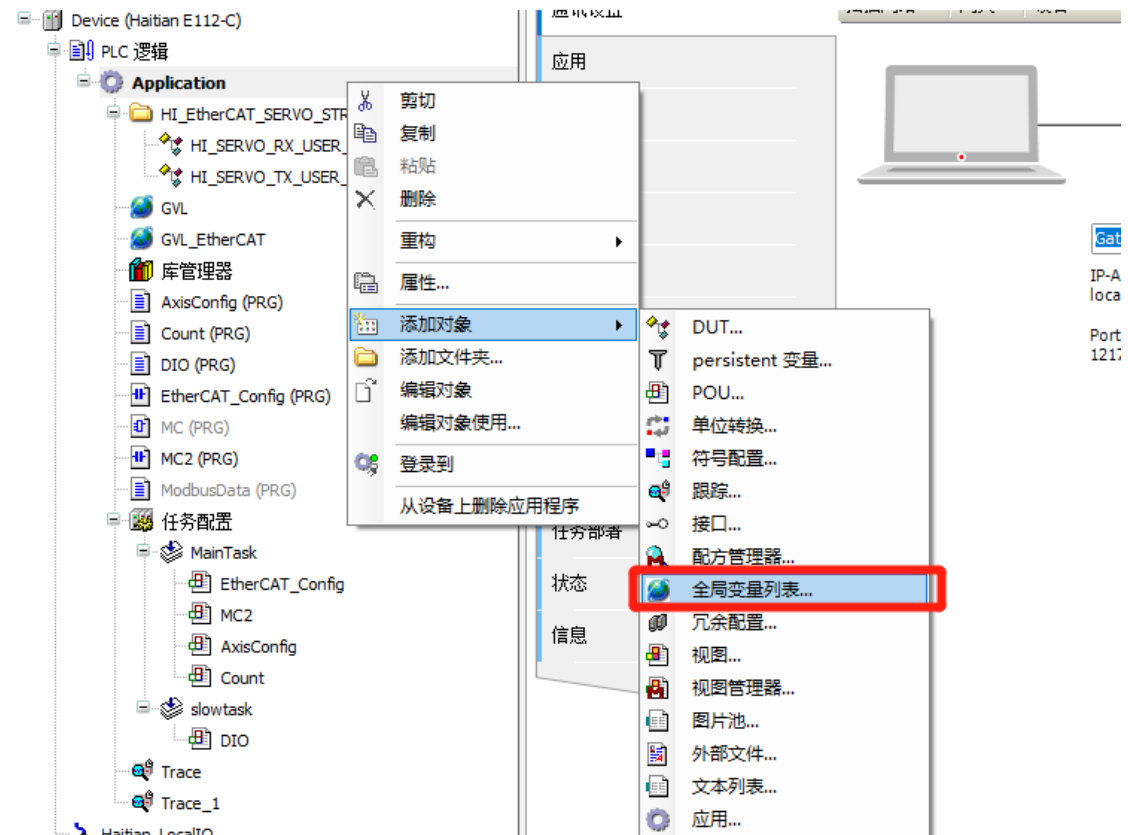
本例使用 Hi_680N 驱动器通过 EtherCAT 通讯控制单个轴进行往返运动

3.3.2.1 通讯配置

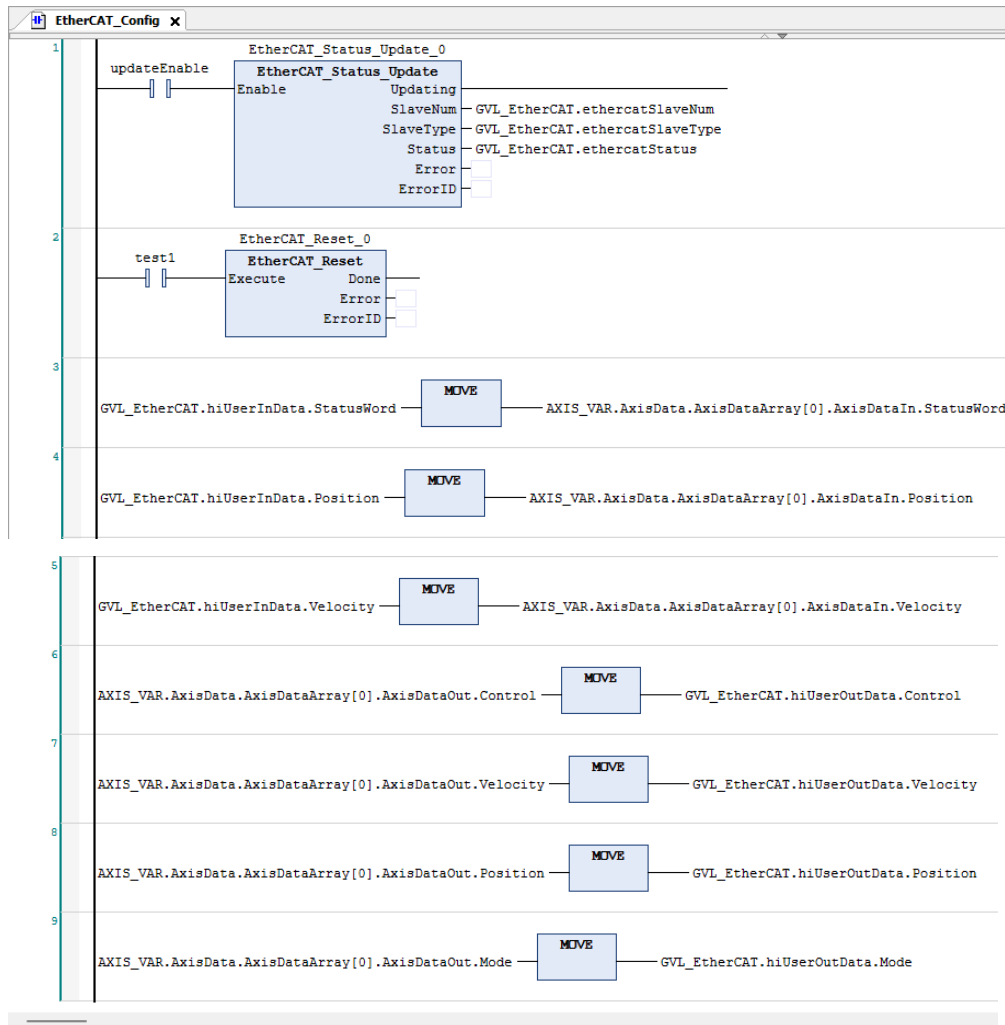
首先按照 3.3.1 章节测试控制器与 Hi_680N 驱动器是否能正常通讯，然后退出登录，在 Application 下创建两个数据收发结构体用于 PDO 数据的接收和发送，根据需要定义变量，本例定义结构体如下：



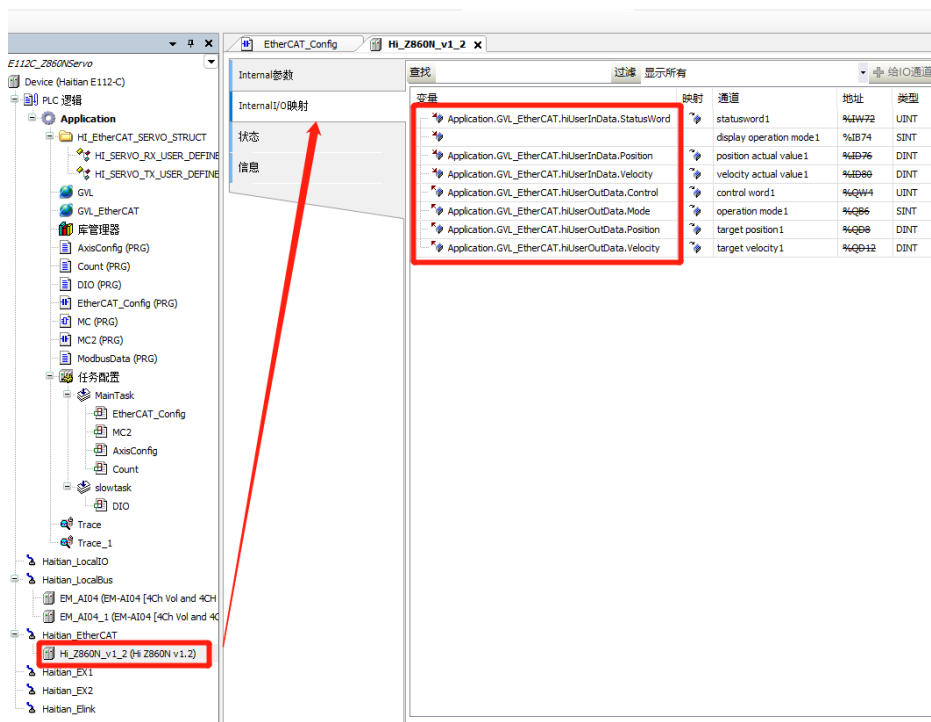
在 Application 下添加全局变量列表，在全局变量列表里定义 EtherCAT 状态、从站个数、从站类型三个变量，同时可以声明 PDO 接收和发送变量，数据类型为上面创建的两个数据收发结构体（单轴可以选择直接使用数据收发结构体而无需重复声明，重复声明是为方便多轴情况下使用）。



全局变量列表创建完成后，创建用于 EtherCAT 通讯的程序，调用 EtherCAT_Status_Update 功能块用于 EtherCAT 状态更新和读取，调用 EtherCAT_Reset 功能块用于总线复位，同时将 PDO 接收和发送变量与轴的特殊输入输出变量进行交互。



最后，在 Haitian_EtherCAT 栏找到 Hi_Z860 驱动器设备，点击 InternalI/O 映射栏，将用于 PDO 接收和发送的变量映射到对应的通道下。



3.3.2.2 轴配置

在 Application 下创建两个新的程序，分别用于轴的参数配置和运动功能块的调用，具体的轴数据结构体含义以及运动功能块描述参考 5.1 运动功能块章节。

(1) 轴参数配置程序

轴参数配置程序主要对轴的运动模式、速度、加速度、电子齿轮比、限位等数据进行设置，用户根据自己的实际需求进行配置，本例参数配置如下：

```

AxisConfig x
1  PROGRAM AxisConfig
2  VAR
3      i: INT;
4      varee: BYTE := 1;
5      mm: BOOL := 0;
6      maxvel: LREAL := 2000.0;
7      d : UINT;
8      vsdfv : byte := 1;
9      homeoffset: LREAL := 0.0;
10     home: LREAL := 0.0;
11     indexuse: BOOL := 1;
12 END_VAR
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
MC_LIB.AXIS_VAR.AxisData.BusStatus := vsdfv;
MC_LIB.AXIS_VAR.AxisData.AxisDataArray[0].VirtualEnable := mm; (*虚轴使能 0: 不使能 1: 使能*)
MC_LIB.AXIS_VAR.AxisData.AxisDataArray[1].VirtualEnable := mm;
MC_LIB.AXIS_VAR.AxisData.AxisDataArray[2].VirtualEnable := mm;
MC_LIB.AXIS_VAR.AxisData.AxisDataArray[3].VirtualEnable := mm;
MC_LIB.AXIS_VAR.AxisData.AxisDataArray[4].VirtualEnable := mm;
MC_LIB.AXIS_VAR.AxisData.AxisDataArray[5].VirtualEnable := mm;
MC_LIB.AXIS_VAR.AxisData.AxisDataArray[6].VirtualEnable := mm;
MC_LIB.AXIS_VAR.AxisData.AxisDataArray[7].VirtualEnable := mm;

FOR i:= 0 TO 7 BY 1 DO
    AXIS_VAR.AxisPara[i].AxisType.ControllerMode := 0; //位置模式
    AXIS_VAR.AxisPara[i].AxisType.ActiveFlag := 1;
    AXIS_VAR.AxisPara[i].AxisType.VelocityType := varee;
    AXIS_VAR.AxisPara[i].MotionData.GearScale := 10; //电子齿轮比 10u/r 一圈10u
    AXIS_VAR.AxisPara[i].MotionData.InputScale := 131072; //每转所需的脉冲数
    AXIS_VAR.AxisPara[i].MotionData.OutputScale := 131072;
    AXIS_VAR.AxisPara[i].MotionData.MaxVel := maxvel;
    AXIS_VAR.AxisPara[i].MotionData.MaxAcc := 200000.0;
    AXIS_VAR.AxisPara[i].MotionData.MaxDec := 200000.0;
    AXIS_VAR.AxisPara[i].MotionData.MaxError := 2000.0;
    AXIS_VAR.AxisPara[i].MotionData.MaxJerk := 200000.0;
    AXIS_VAR.AxisPara[i].MotionData.MaxPos := 999999999999.9;
    AXIS_VAR.AxisPara[i].MotionData.MinPos := -999999999999.9;
    AXIS_VAR.AxisPara[i].MotionData.MinError := 10000; //999999999999.9;
    AXIS_VAR.AxisPara[i].MotionData.MaxError := 10000; //999999999999.9;
END_FOR

AXIS_VAR.AxisPara[0].HomeData.HomeSearchVel := 0.0;
AXIS_VAR.AxisPara[0].HomeData.HomeLatchVel := 0.0;
AXIS_VAR.AxisPara[0].HomeData.HomeFinalVel := 0.0;
AXIS_VAR.AxisPara[0].HomeData.HomeOffset := homeoffset;
AXIS_VAR.AxisPara[0].HomeData.Home := home;
AXIS_VAR.AxisPara[0].HomeData.IndexUse := indexuse;

d := SIZEOF(MC_LIB.AXIS_VAR.AxisPara[0]); //计算字节长度
d := SIZEOF(MC_LIB.AXIS_VAR.AxisPara[0]);

```

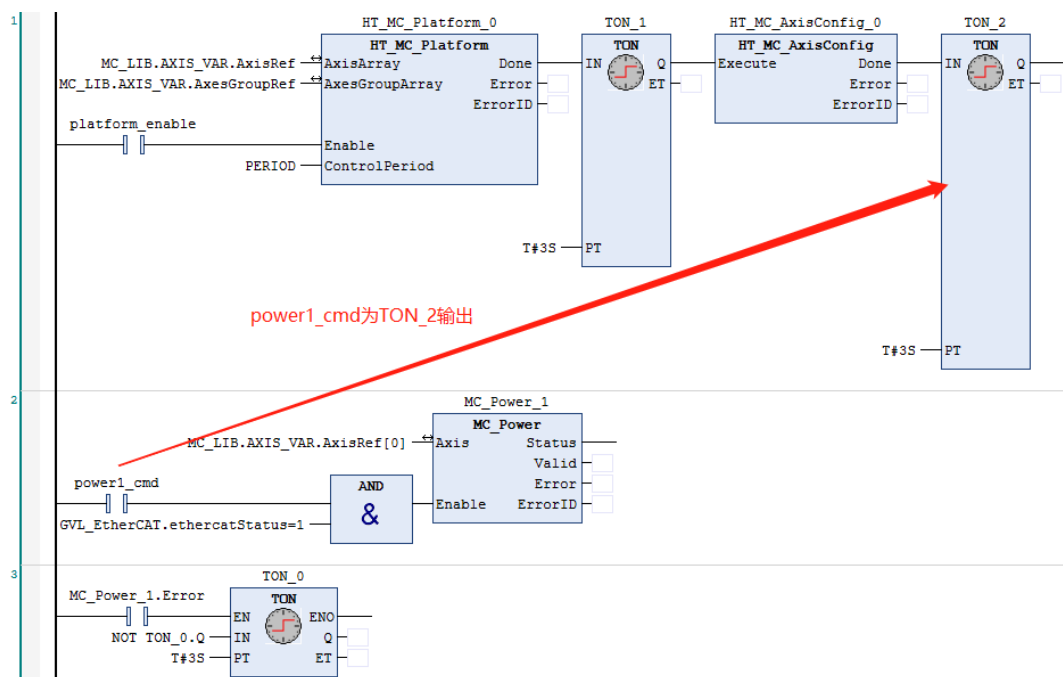
(2) 运动功能块的调用

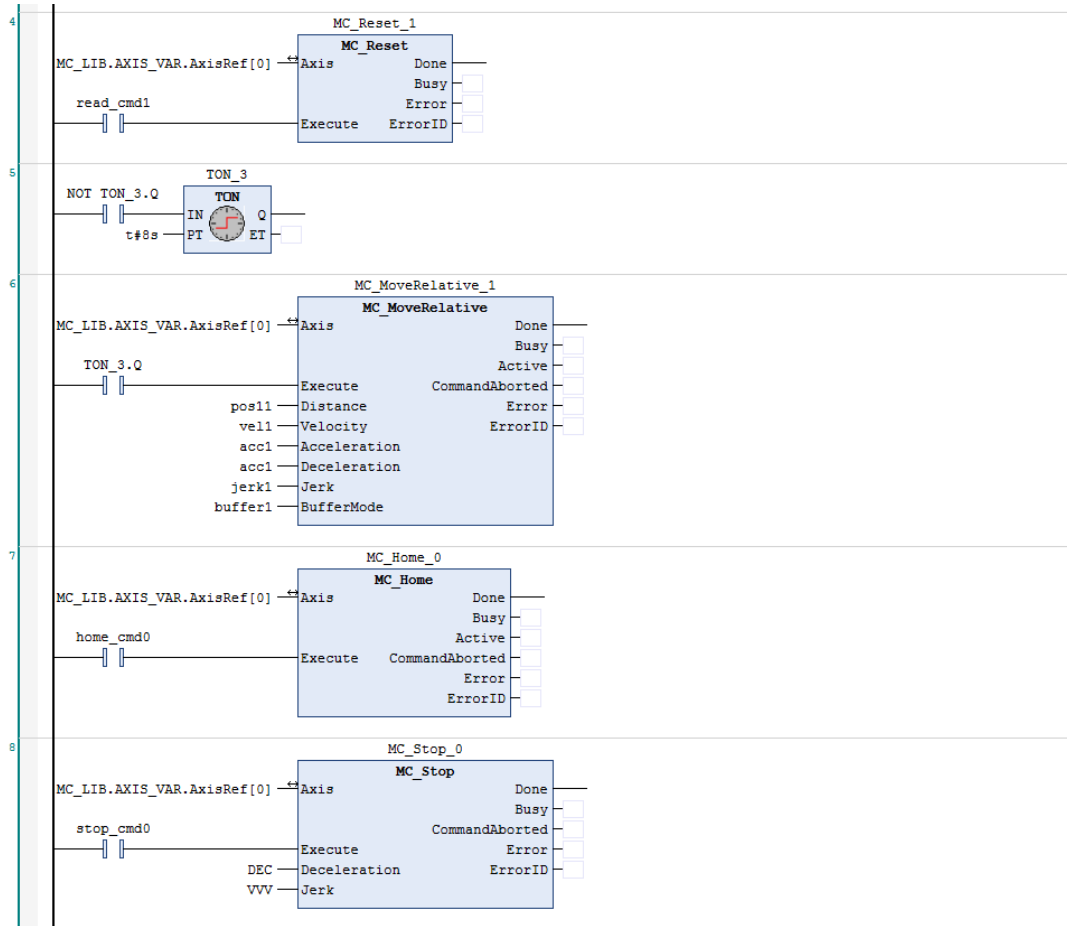
根据下图调用运动功能块

```

1  PROGRAM MC2
2  VAR
3      HT_MC_Platform_0: HT_MC_Platform;
4      platform_enable : BOOL := TRUE;
5      PERIOD: UDINT := 1000;
6      HT_MC_AxisConfig_0: HT_MC_AxisConfig;
7      power1_cmd : BOOL;
8      read_cmd1:BOOL;
9      home_cmd0 : BOOL;
10     MC_Power_1: MC_Power;
11     MC_Reset_1:MC_Reset;
12     pos11: REAL := -100;
13     vell: REAL := 33.33;
14     accl: REAL := 150.0;
15     jerk1: REAL := 1000.0;
16     buffer1 AT %MW100: HT_MC_BUFFER_MODE := 0;
17     vvv: INT;
18     MC_MoveRelative_1: MC_MoveRelative;
19     MC_Stop_0: MC_Stop;
20     DEC: REAL := 200.0;
21     MC_Home_0: MC_Home;
22     vel_cmd0 : BOOL;
23     stop_cmd0 : BOOL;
24     TON_1: TON;
25     TON_2: TON;
26     TON_0: TON;
27     TON_3: TON;
28 END_VAR

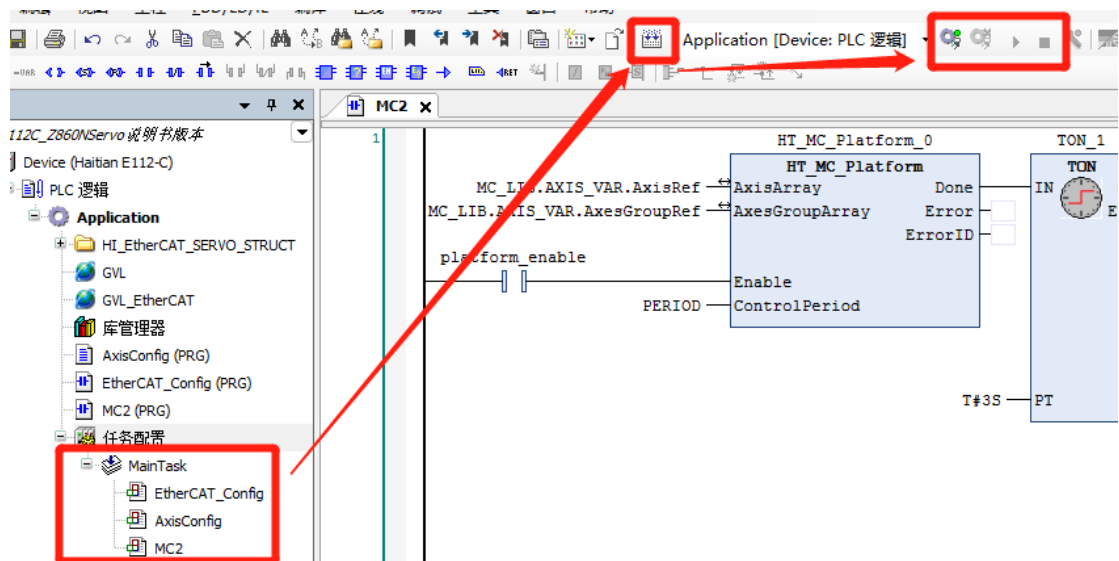
```





3.3.2.3 轴控制

将通讯程序，轴参数配置程序以及运动功能块调用程序添加到 MainTask 下，编译然后登录控制器并运行程序。



程序运行后首先执行执行 MC_Reset 对轴进行复位，复位完成后关闭功能块，然后执行 EtherCAT_Status_Update 以及 HT_MC_Platform 功能块，随后 MC_Power 功能块自动激活，轴开始进行往返运动，每隔 8S 换向一次。

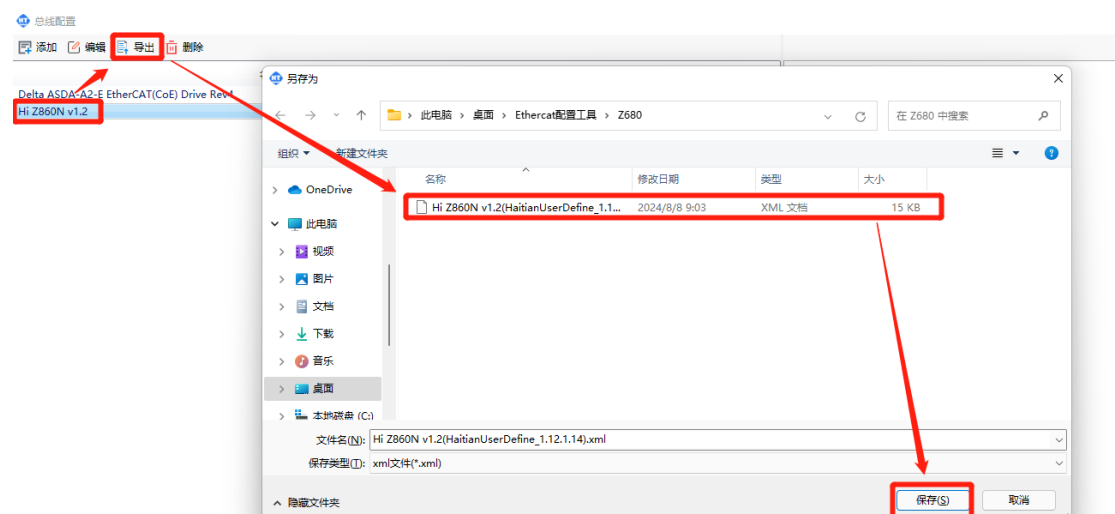
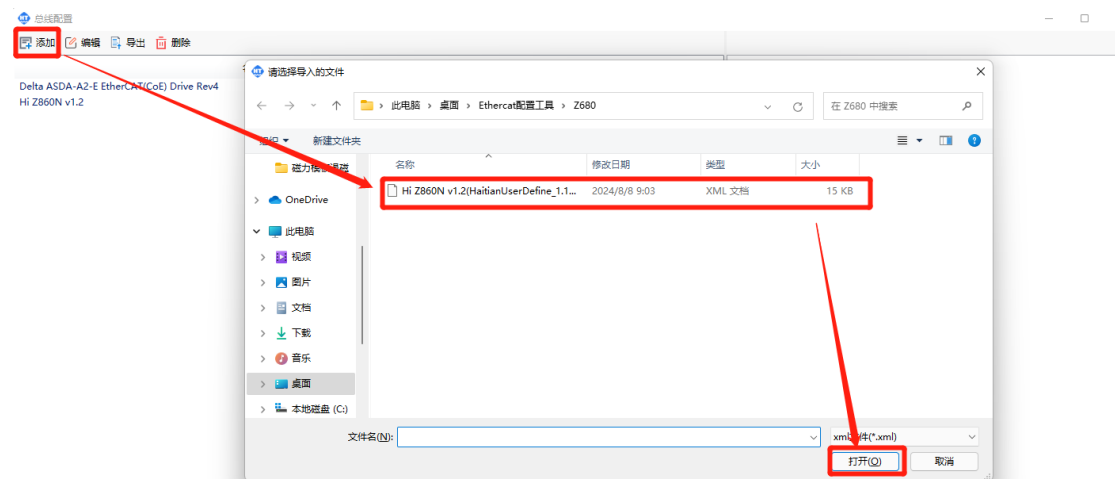
3.3.3 导入第三方从站设备

使用第三方驱动器时，不能将设备的 XML 文件直接导入使用，需要用配置工具进行转换，同时 XML 文件配置工具还可以对 PDO 进行配置。

配置工具获取网址：<http://co.haitianiot.com:7010/software/Hic%20BusAssistant.rar>

3.3.3.1 XML 文件导入

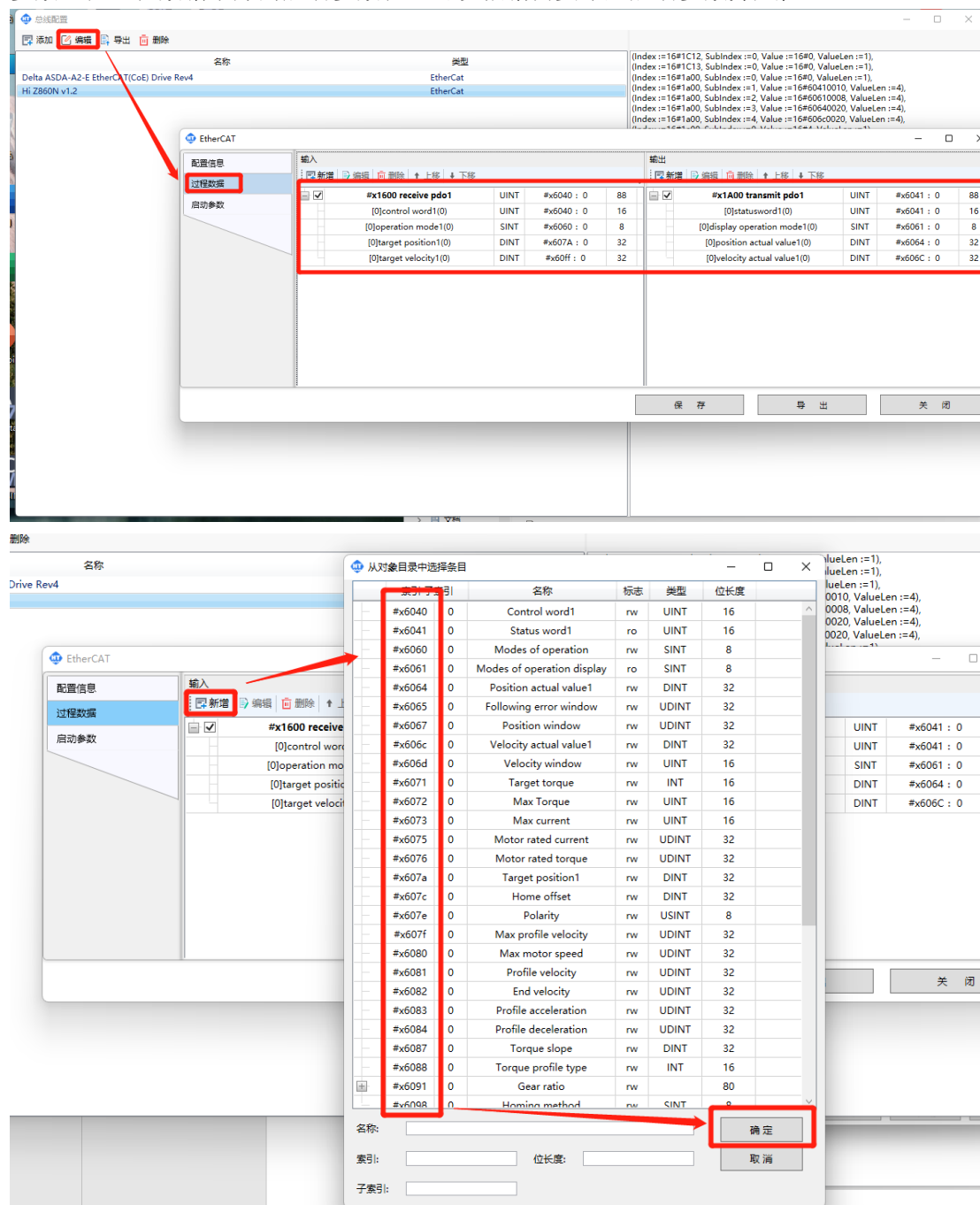
首先打开 XML 配置工具，点击左上角添加按钮，然后选择需要导入的设备的 XML 文件，点击打开。



打开文件后，窗口会显示已经导入的 XML 文件目录，点击刚才导入的文件，点击导出，然后保存，保存后的 XML 文件可以导入到 codesys 设备管理器中，导入后便可在 Haitian_EtherCAT 栏添加并使用该设备。

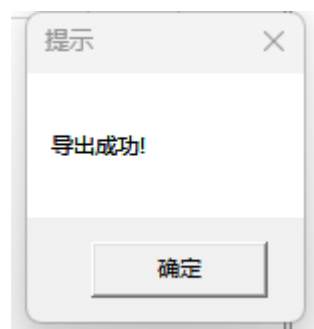
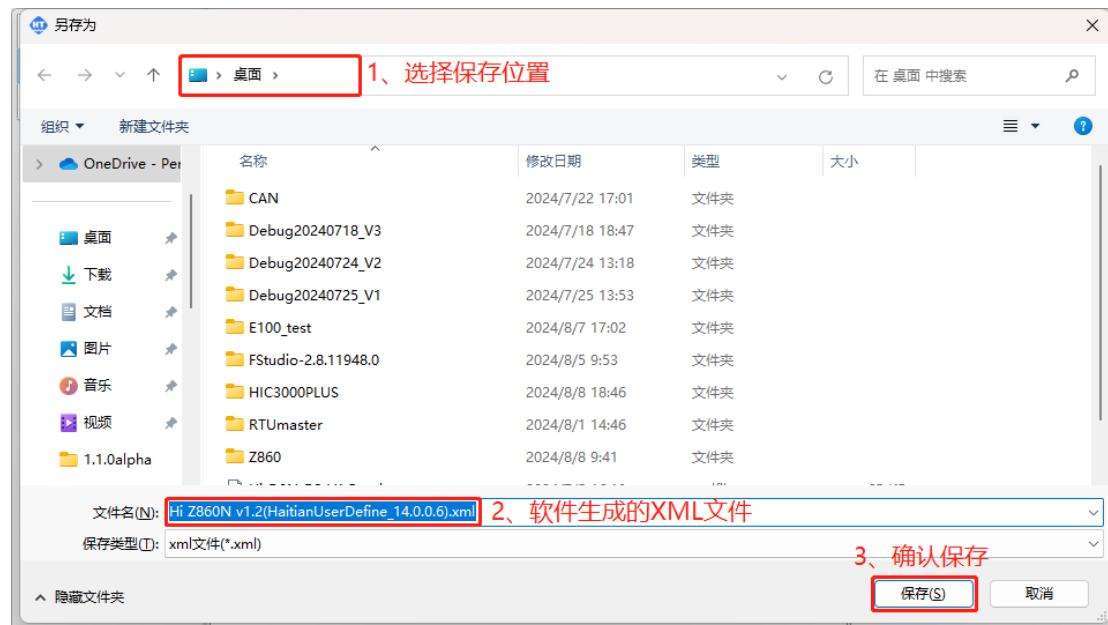
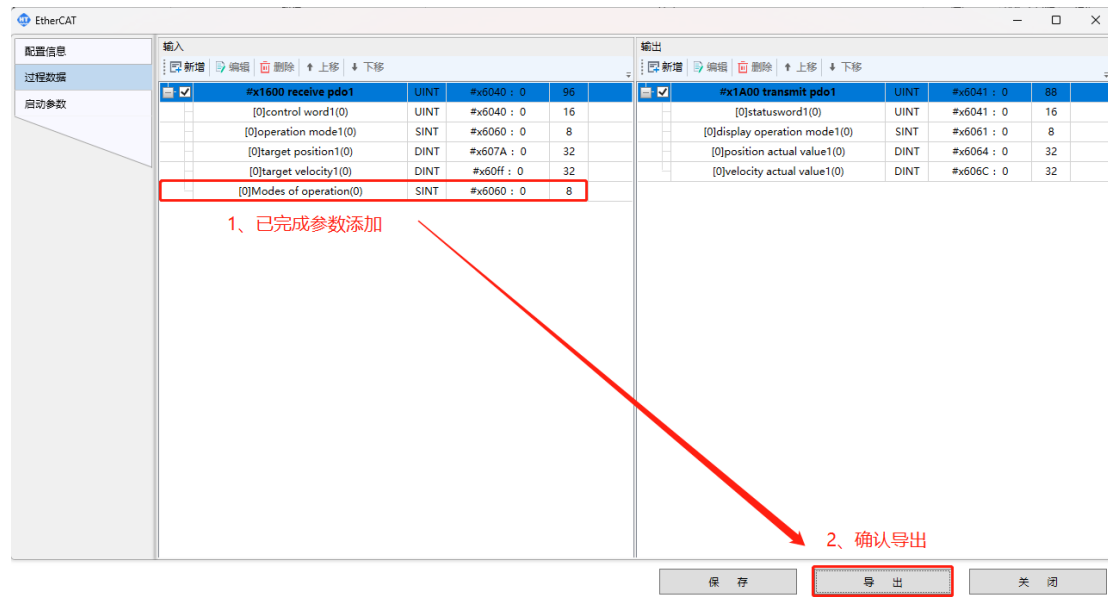
3.3.3.2 从站参数配置

在主页面选择需要配置的文件，点击编辑会弹出配置窗口，点击过程数据一栏会显示出传送 PDO 和接收 PDO 的数据内容，点击新增弹出对象字典，可以在目录中选择需要新增的参数，在过程数据下方的启动参数栏还可以根据需要添加启动参数并赋值。



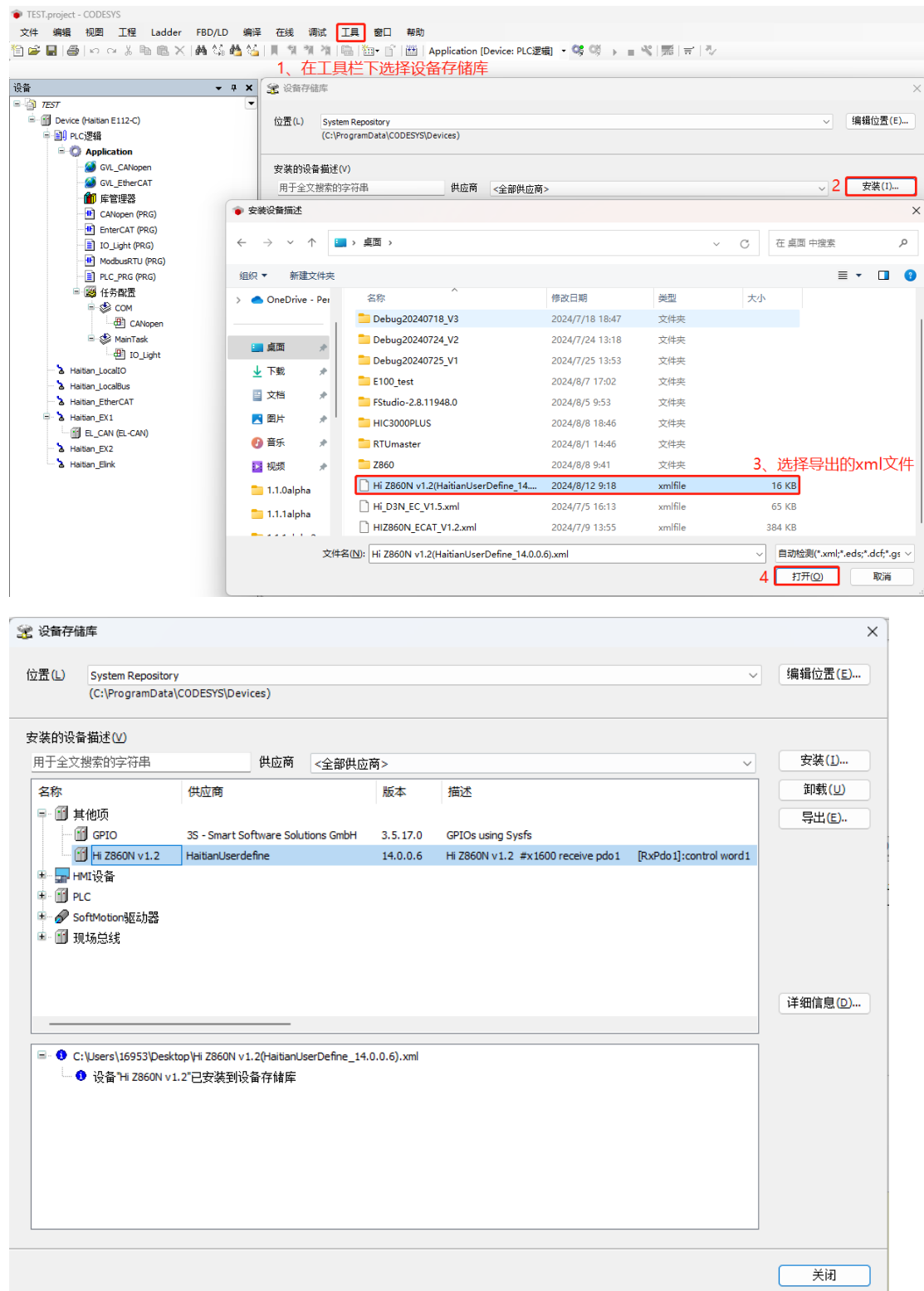
3.3.3.3 XML 文件导出

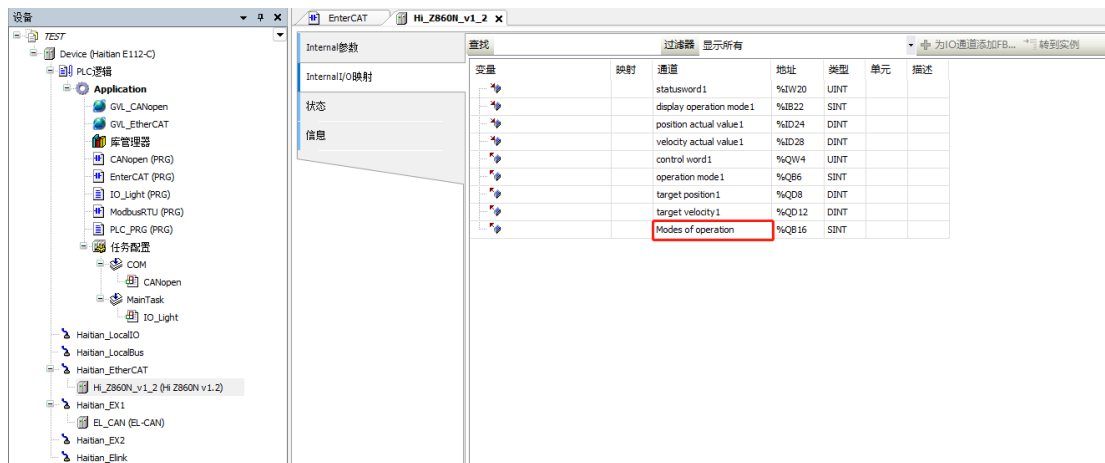
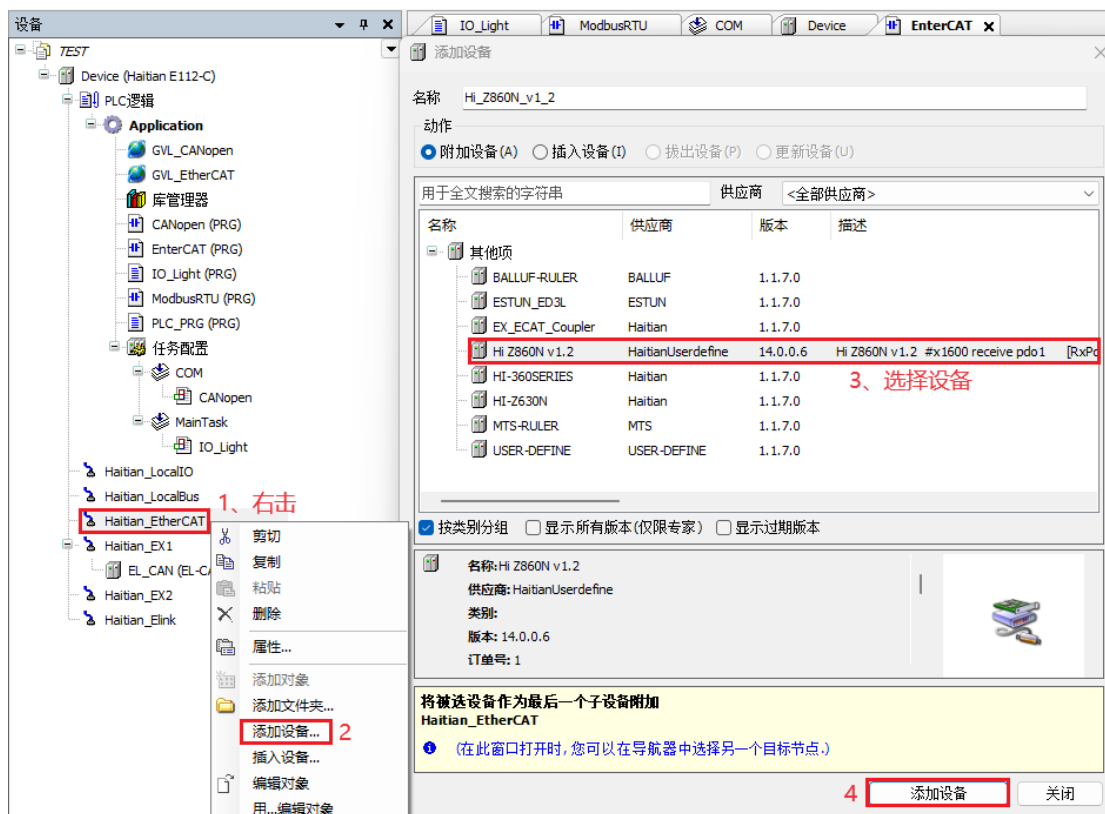
完成参数添加后，确认导出.XML 文件。



3.3.3.4 CodeSys 平台导入第三方从站

将生成的 xml 文件导入到 CodeSys 平台中，点击“工具”，选择“设备存储库”，点击“安装”，选择生成的 xml 文件，点击“打开”。导入成功后，会提示“已安装到设备存储库”，在“其他项”中可以看到导入的第三方设备。右击“Haitian_EtherCAT”,选择所需导入的第三方设备，进行配置。





3.4 CANopen 主站通讯设置

E112C 控制器本体默认不支持 CAN 通讯，需要扩展 EL-CAN 选配卡实现 CAN 通讯。

3.4.1 E112C 与 Hi 驱动器通信举例

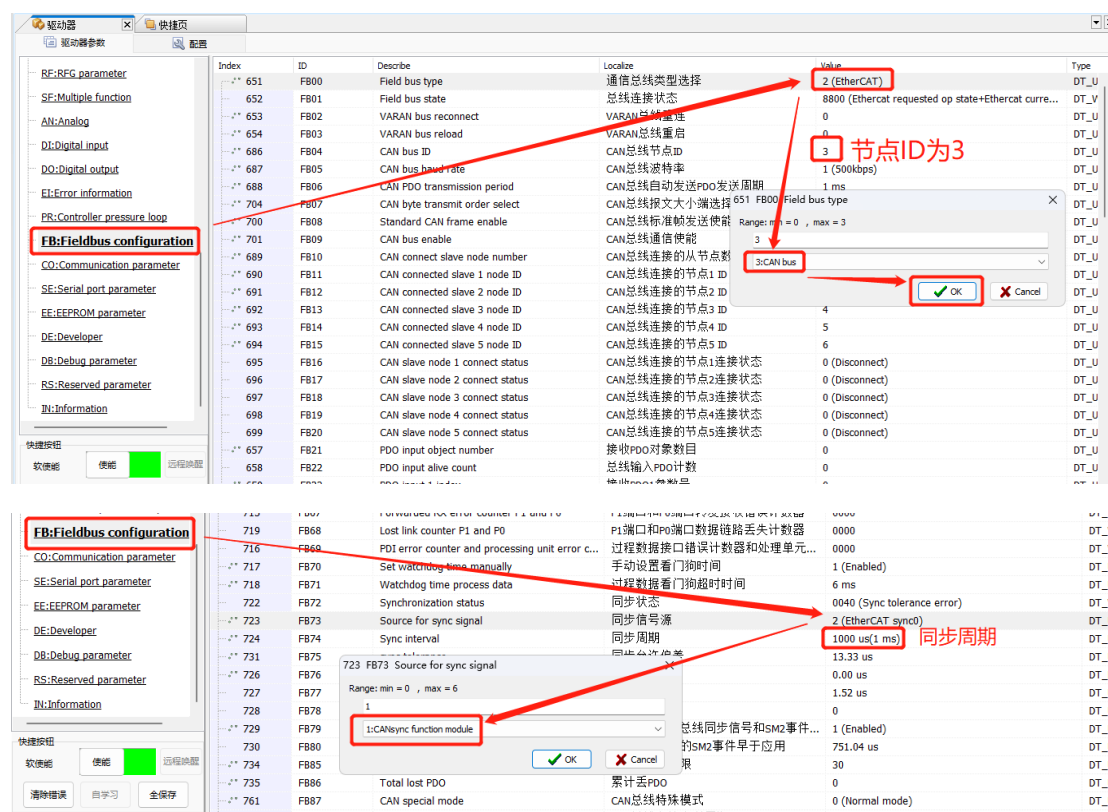
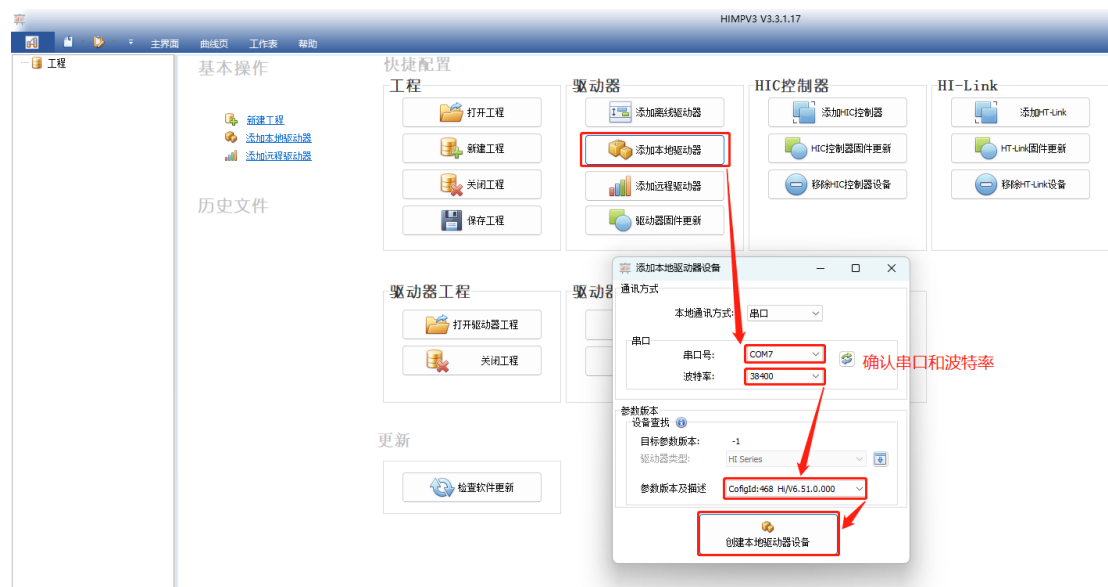
3.4.1.1 硬件配置准备

将 1 台 E112C 控制器作为 CANopen 主站，1 台 Hi 驱动器和一把 CAN 尺作为从站，将 E112C 控制器的 EL-CAN 扩展模块的 CANH、CANL 接口依次与 Hi 驱动器(第三代)的 22 号

端口 (CAN-H)、21 号端口(CAN-L)接口连接,并串接到 UCAN 工具的 5 (H)、4 (L) 接口。将第三代驱动器 24 号端口接 24v, 23 号端口接 0v 为驱动器供电。

3.4.1.2 驱动器参数配置

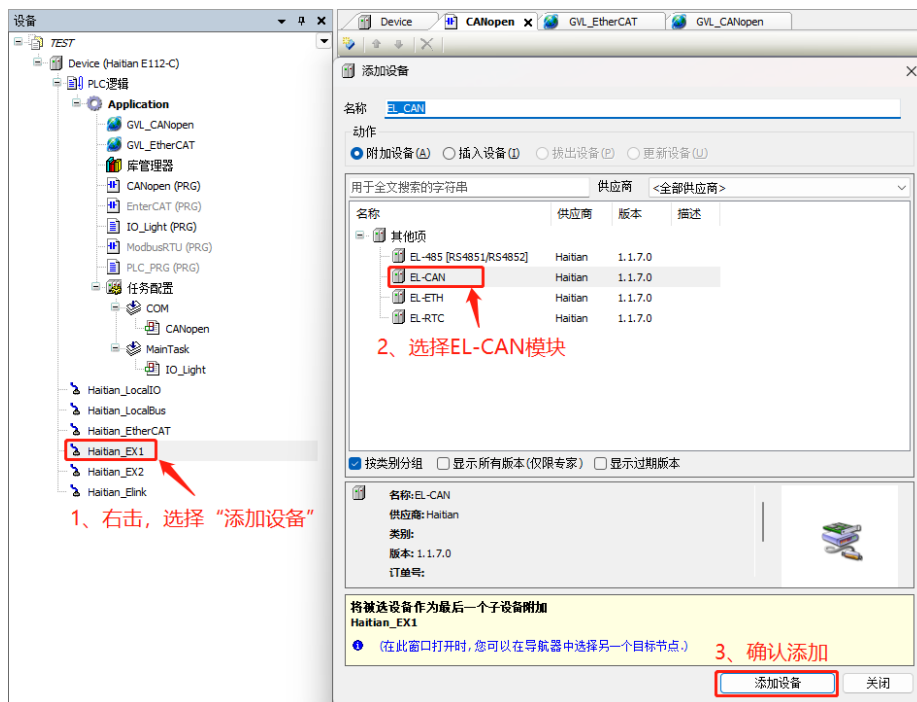
HIMPV3 上位机连接驱动器，添加本地驱动器，按图示进行创建本地驱动器设备，找到驱动器参数组别 FB，将 FB00 总线类型修改为 CAN bus，查看 FB04 CAN 总线节点 ID，FB73 同步信号源修改为 CANsync function module，FB74 同步周期设置为所需值。



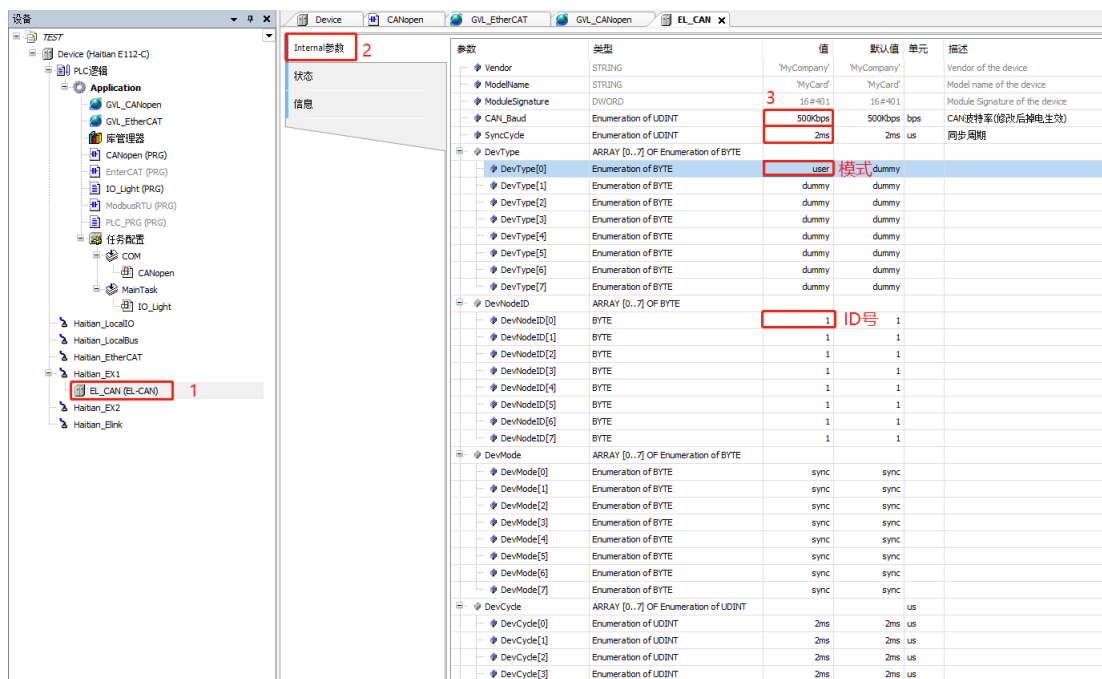
3.4.2 CANopen 参数配置

3.4.2.1 user 自定义设备

(1) 右击 Haitian_EX1(根据 EL-CAN 模块安装的位置选择 EX1 或 EX2), 添加 EL-CAN 设备。

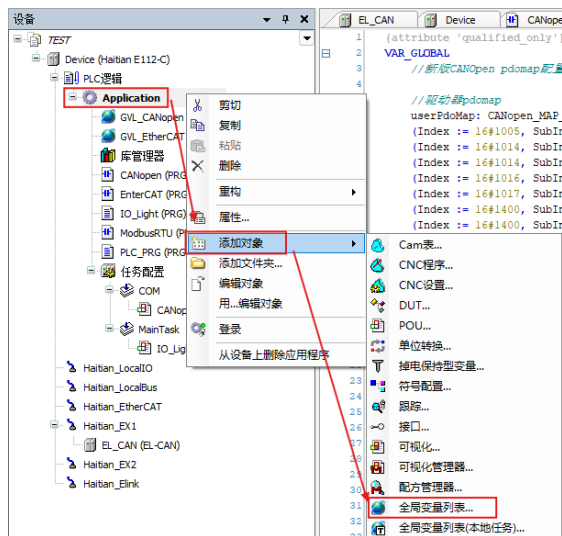


(2) 双击 EL-CAN 模块, 点击 Internal 参数, 设置 DevType[0]为 user, DevNodeID[0]为 Hi 驱动器 CAN 总线节点 ID。(注意: CAN_Baud 和 SyncCyde 需和 Hi 驱动器的参数对应, DevMode 下参数均为 sync, DevCycle 下参数均为 2ms。)



Index	ID	Describe	Localize	Value
651	FB00	Field bus type	通信总线类型选择	3 (CAN bus)
652	FB01	Field bus state	总线连接状态	0000 (无值)
653	FB02	VARAN bus reconnect	VARAN总线重连	0
654	FB03	VARAN bus reload	VARAN总线重启	0
686	FB04	CAN bus ID	CAN总线节点ID	1 ID
687	FB05	CAN bus baud rate	CAN总线波特率	1 (500kbps) 波特率
688	FB06	CAN PDO transmission period	CAN总线自动发送PDO发送周期	1 ms
FB72	Synchronization status		同步状态	0010 (Sync timer out)
FB73	Source for sync signal		同步信号源	1 (CANsync function module)
FB74	Sync interval		同步周期	2000 us(2 ms) 同步周期
FB75	sync tolerance		同步允许偏差	13.33333333333333 us

(3) 右击 Application，添加对象，新建一个全局变量列表，将 PdoMap 和 PdoNum 写入全局变量中。



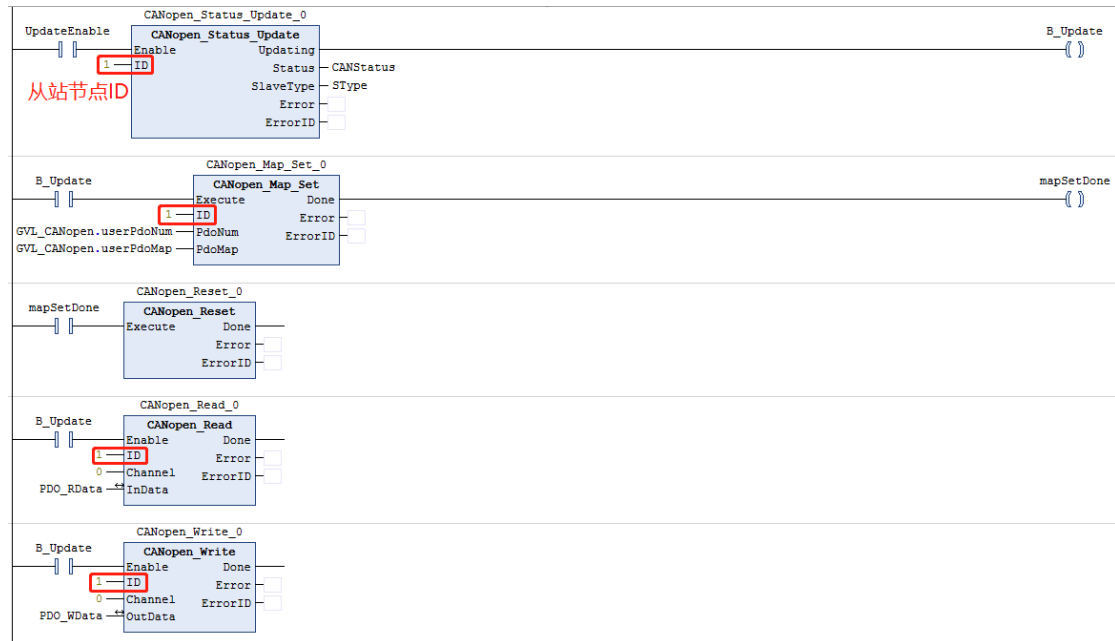
```
{attribute 'qualified_only'}
VAR_GLOBAL
//新版CANOpen pdomap配置方式

//驱动端pdomap
userPdoMap: CANOpen_MAP_ARRAY := (PDOMapArray := [
(Index := 16#1005, SubIndex := 0, value := 16#000000080, ValueLen := 4),
(Index := 16#1014, SubIndex := 0, value := 16#800000082, ValueLen := 4),
(Index := 16#1014, SubIndex := 0, value := 16#000000082, ValueLen := 4),
(Index := 16#1016, SubIndex := 1, value := 16#007F012C, ValueLen := 4),
(Index := 16#1017, SubIndex := 0, value := 16#0064, ValueLen := 2),
(Index := 16#1400, SubIndex := 1, value := 16#80000202, ValueLen := 4),
(Index := 16#1400, SubIndex := 2, value := 16#01, ValueLen := 1),
(Index := 16#1600, ValueLen := 1),
(Index := 16#1600, SubIndex := 1, Value := 16#60400010, ValueLen := 4), //控制字
(Index := 16#1600, SubIndex := 2, Value := 16#60600008, ValueLen := 4), //模式控制
(Index := 16#1600, SubIndex := 3, Value := 16#60FF0020, ValueLen := 4), //速度给定-速度模式
(Index := 16#1600, Value := 3, ValueLen := 1),
(Index := 16#1400, SubIndex := 1, value := 16#00000202, ValueLen := 4),
(Index := 16#1800, SubIndex := 1, value := 16#80000182, ValueLen := 4),
(Index := 16#1800, SubIndex := 2, value := 16#01, ValueLen := 1),
(Index := 16#1A00, ValueLen := 1),
(Index := 16#1A00, SubIndex := 1, Value := 16#60410010, ValueLen := 4), //状态字
(Index := 16#1A00, SubIndex := 2, Value := 16#60640020, ValueLen := 4), //实际位置
(Index := 16#1A00, Value := 2, ValueLen := 1),
(Index := 16#1800, SubIndex := 1, value := 16#00000182, ValueLen := 4),
(Index := 16#1801, SubIndex := 1, value := 16#80000282, ValueLen := 4),
(Index := 16#1802, SubIndex := 1, value := 16#80000382, ValueLen := 4),
(Index := 16#1803, SubIndex := 1, value := 16#80000482, ValueLen := 4),
(Index := 16#1401, SubIndex := 1, value := 16#80000302, ValueLen := 4),
(Index := 16#1401, SubIndex := 2, value := 16#01, ValueLen := 1),
(Index := 16#1601, ValueLen := 1),
(Index := 16#1601, SubIndex := 1, Value := 16#607A0020, ValueLen := 4), //位置模式下位置值
(Index := 16#1601, SubIndex := 2, Value := 16#60810020, ValueLen := 4), //位置模式下速度值
(Index := 16#1601, Value := 2, ValueLen := 1),
(Index := 16#1401, SubIndex := 1, value := 16#00000302, ValueLen := 4)
]);

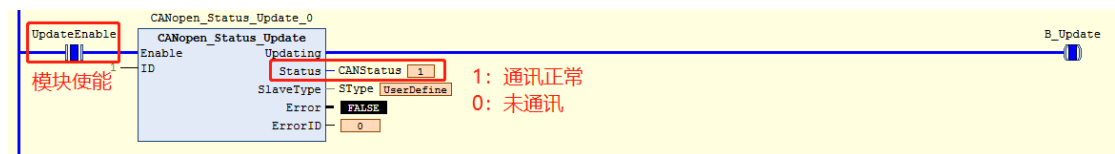
userPdoNum : BYTE:=30;

END_VAR
```

(4) 右击 Application, 添加对象 POU, 新建一个梯形逻辑图 (LD) 工程, 在程序中添加如图所示功能块及代码, 将程序添加进任务中, 登录并下载程序。



(4) 通信正常时 CANopen_Status_Update 模块 Status 引脚输出为 1, UCAN 可以看到 18X 的数据在变化。



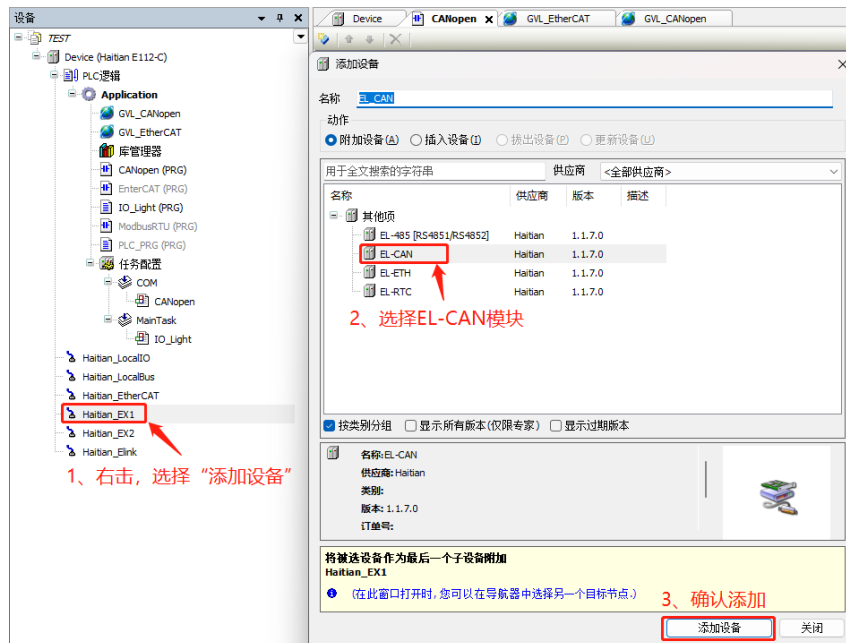
UCAN - [Receive/Transmit]

Client Edit Transmit Setup about

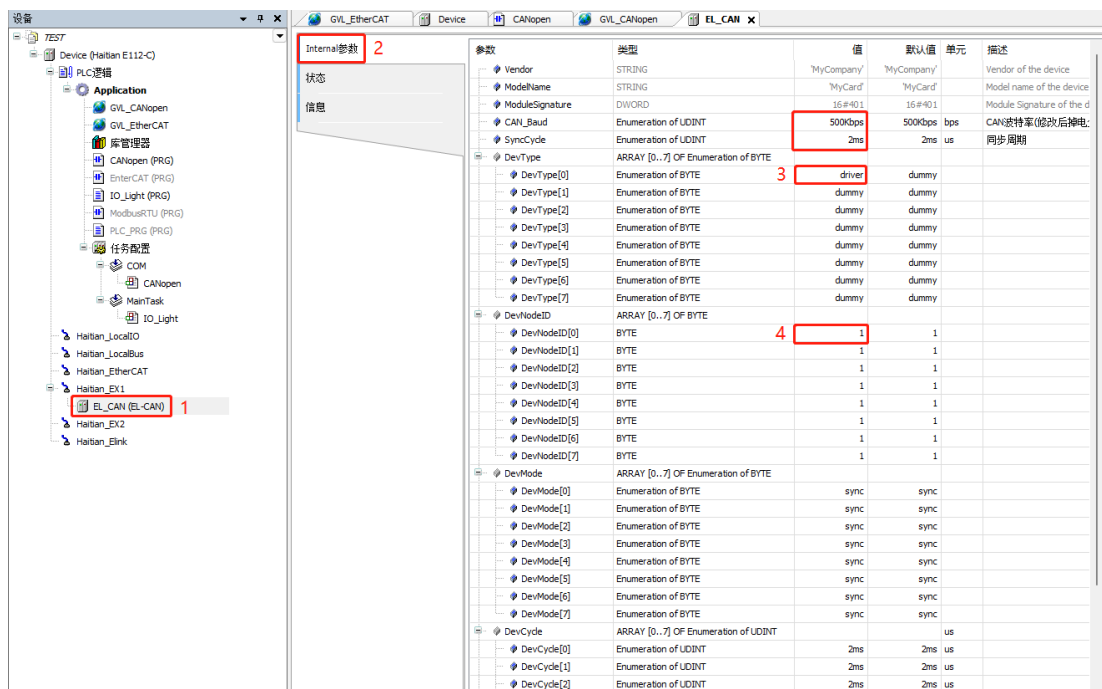
</

3.4.2.2 Hi Driver 标准设备

(1) 右击 Haitian_EX1(根据 EL-CAN 模块安装的位置选择 EX1 或 EX2)，添加 EL-CAN 设备。

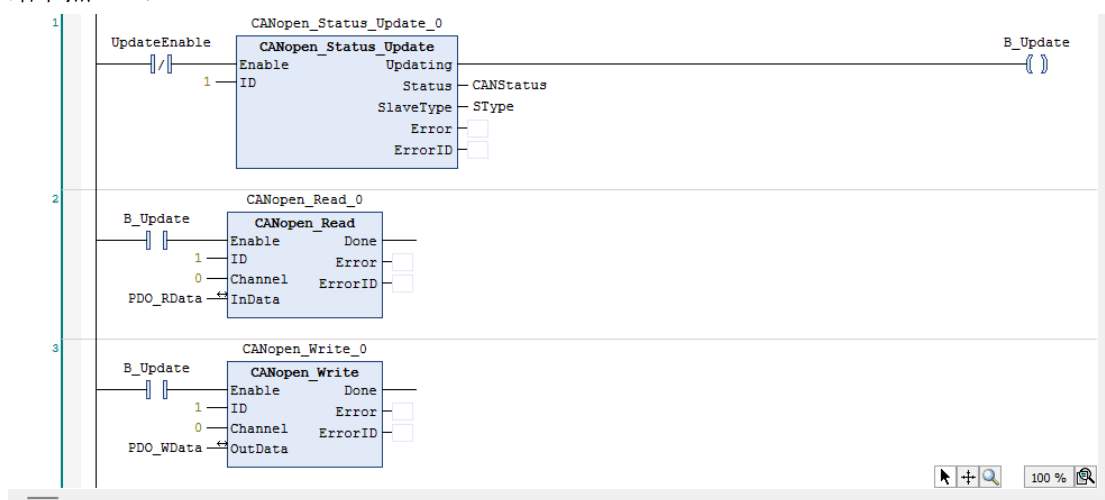


(2) 双击 EL-CAN 模块，点击 Internal 参数，设置 DevType[0]为 driver，DevNodeID[0]为 Hi 驱动器 CAN 总线节点 ID。(注意：CAN_Baud 和 SyncCyde 需和 Hi 驱动器的参数对应，DevMode 下参数均为 sync，DevCycle 下参数均为 2ms。)

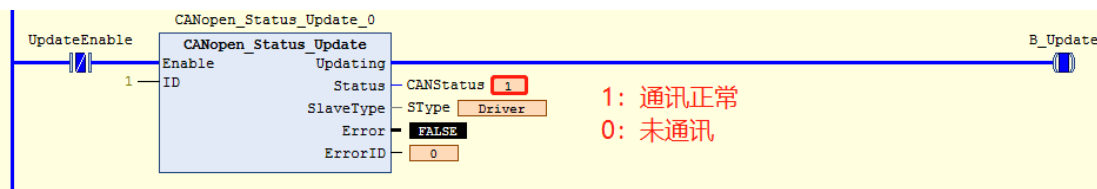


Index	ID	Describe	Localize	Value
651	FB00	Field bus type	通信总线类型选择	3 (CAN bus)
652	FB01	Field bus state	总线连接状态	0000 (无值)
653	FB02	VARAN bus reconnect	VARAN总线重连	0
654	FB03	VARAN bus reload	VARAN总线重启	0
686	FB04	CAN bus ID	CAN总线节点ID	1 ID
687	FB05	CAN bus baud rate	CAN总线波特率	1 (500kbps) 波特率
688	FB06	CAN PDO transmission period	CAN总线自动发送PDO发送周期	1 ms
FB72	Synchronization status		同步状态	0010 (Sync timer out)
FB73	Source for sync signal		同步信号源	1 (CANsync function module)
FB74	Sync interval		同步周期	2000 us(2 ms) 同步周期
FB75	sync tolerance		同步允许偏差	13.333333333333 us

(3) 右击 Application, 添加对象 POU, 新建一个梯形逻辑图 (LD) 工程, 在程序中添加如图所示功能块及代码, 将程序添加进任务中, 登录并下载程序。(注意: 功能块中的 ID 为从站节点 ID。)



(4) 通信正常时 CANopen_Status_Update 模块 Status 引脚输出为 1, UCAN 可以看到 18X 的数据在变化。



UCAN - [Receive/Transmit]

Client Edit Transmit Setup about

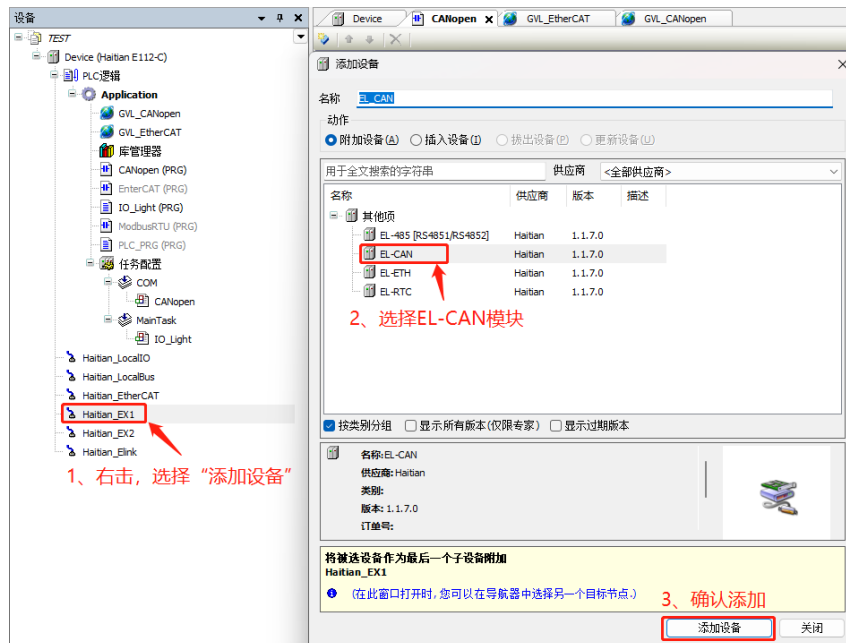
Receive

Message	Length	Data	Period	Count	RTR-Per.	RTR-CNT.
080h	0		2	6474	0	0
181h	6	88 00 00 00 00 00	2	6474	0	0
201h	7	00 00 00 00 00 00 00	2	6474	0	0
301h	8	00 00 00 00 00 00 00 00	2	6474	0	0
700h	1	05	100	130	0	0
701h	1	05	100	130	0	0

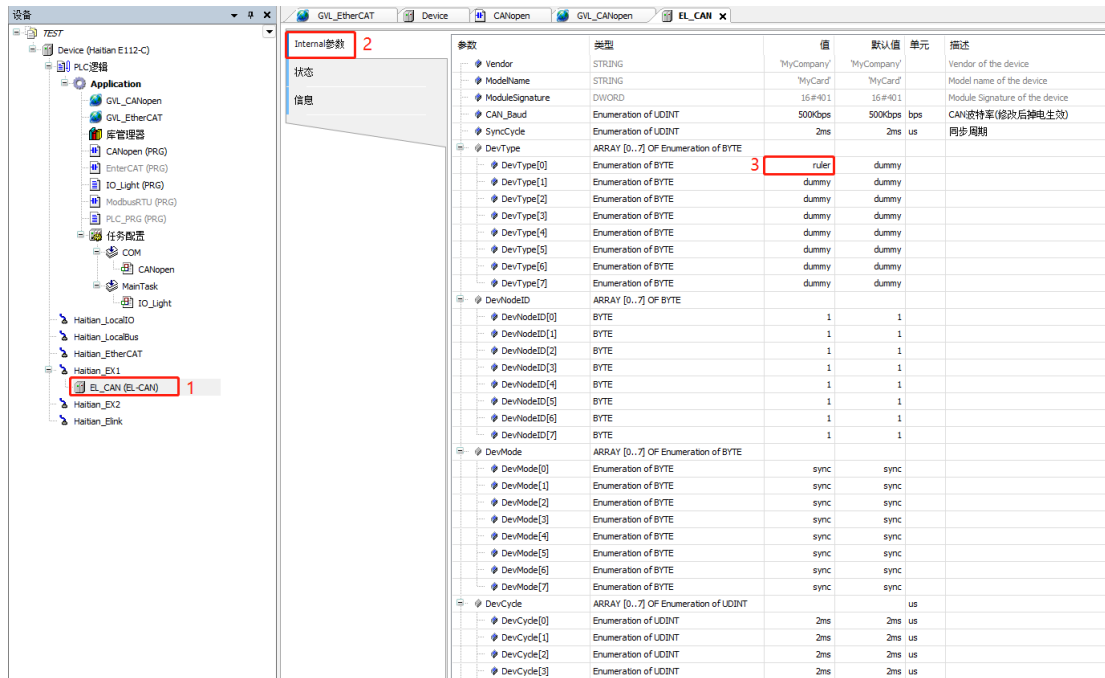
18X为驱动器的数据, 20X为控制器的数据, "X" 表示ID

3.4.2.3 ruler 位置尺

(1) 右击 Haitian_EX1(根据 EL-CAN 模块安装的位置选择 EX1 或 EX2)，添加 EL-CAN 设备。



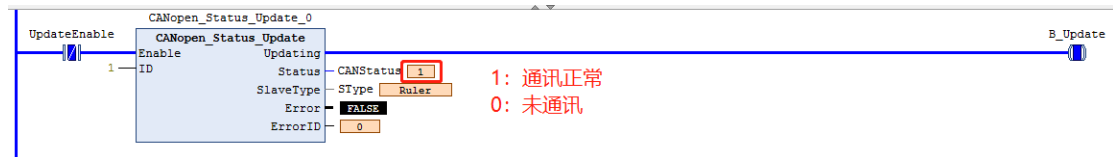
(2) 双击 EL-CAN 模块，点击 Internal 参数，设置 DevType[0]为 ruler。



(3) 右击 Application，添加对象 POU，新建一个梯形逻辑图（LD）工程，在程序中添加如图所示功能块及代码，将程序添加进任务中，登录并下载程序。



(4) 通信正常时 CANopen_Status_Update 模块 Status 引脚输出为 1。



第四章 扩展模块

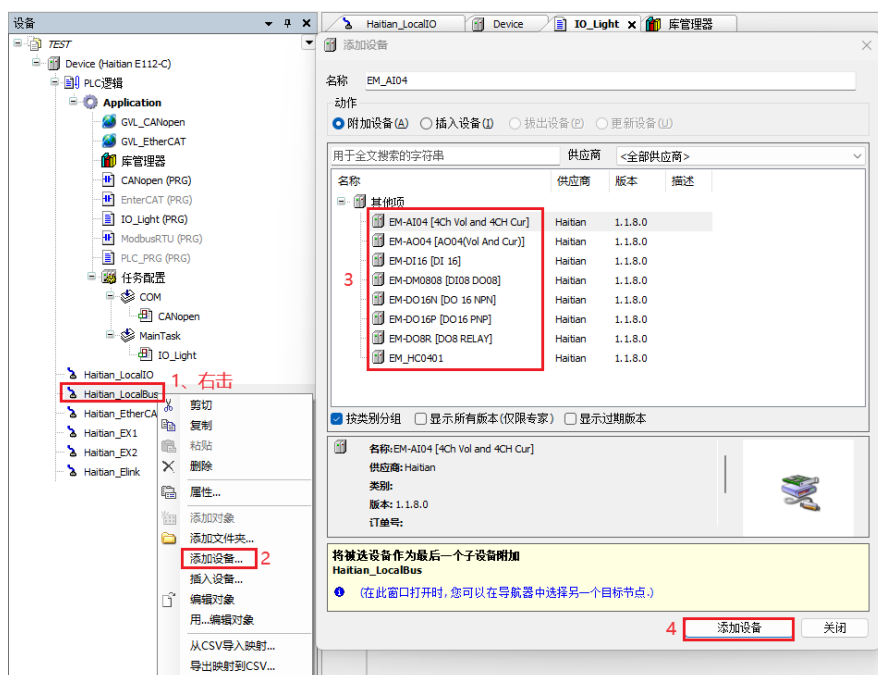
4.1 扩展模块类型

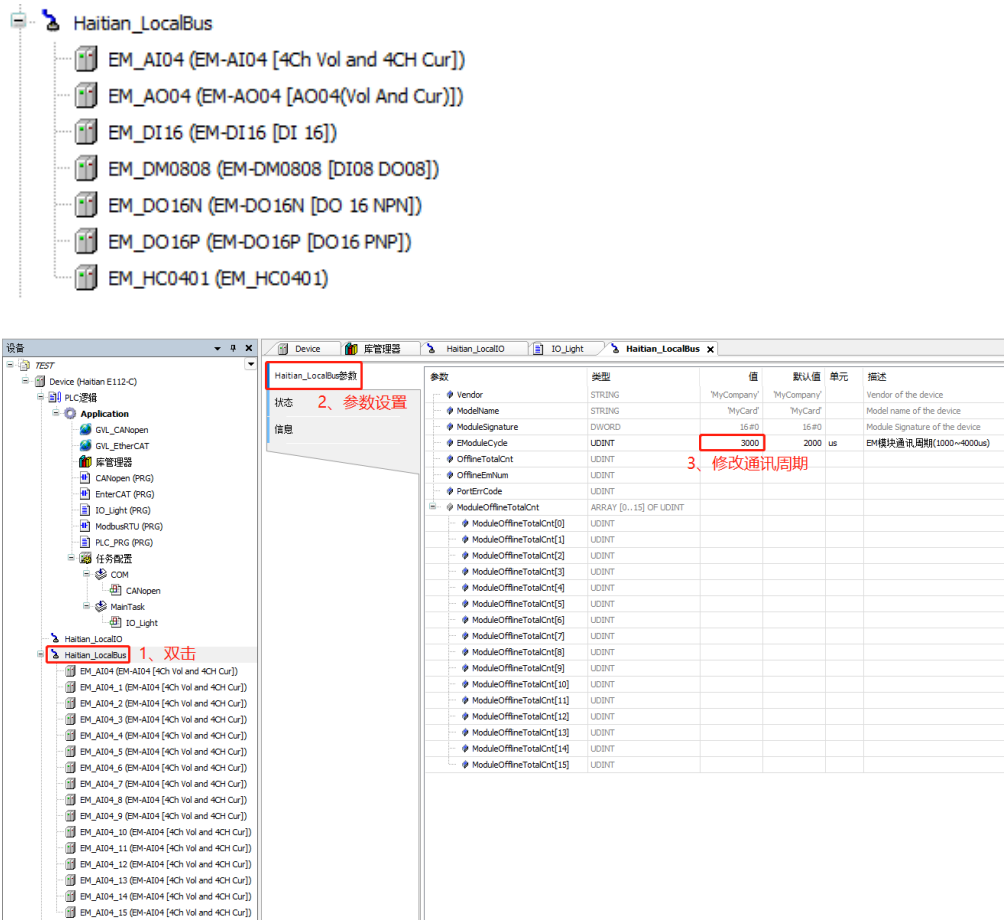
E112-C 本体可直接在右侧插入安装扩展模块，本体最多可扩展 16 个扩展模块。

模块类型	型号	描述
数字量输入模块	EM-DI16	16 路数字量输入模块
数字量输出模块	EM-DO16N	16 路数字量 NPN 输出模块
	EM-DO16P	16 路数字量 PNP 输出模块
	EM-DO8R	8 路数字量输出模块，继电器输出
数字量输入输出模块	EM-DM0808	8 路数字量输入 8 路数字量输出混合模块
模拟量输入模块	EM-AI04	4 路模拟量输入模块，电流/电压输入
模拟量输出模块	EM-AO04	4 路模拟量输出模块，电流/电压输出
高速计数模块	EM-HC0401	4 路单端 1 路差分高速计数模块

4.2 扩展模块组态

可通过右击“Haitian_LocalBus”添加设备，根据实际模块安装顺序，从左到右依次添加相应模块，添加完成后会在“Haitian_LocalBus”下显示相应模块。（**注意：**若添加的扩展模块数量较多，导致扩展模块通信失败，请双击“Haitian_LocalBus”，在“Haitian_LocalBus 参数”下，将“EM 模块通讯周期”视具体情况增大。（例如，改为 3000us。））



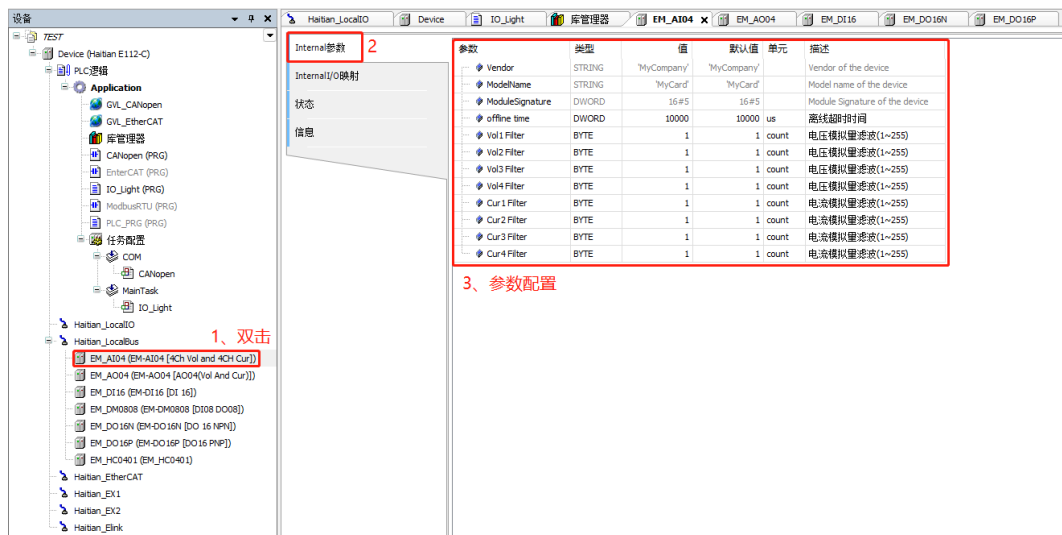


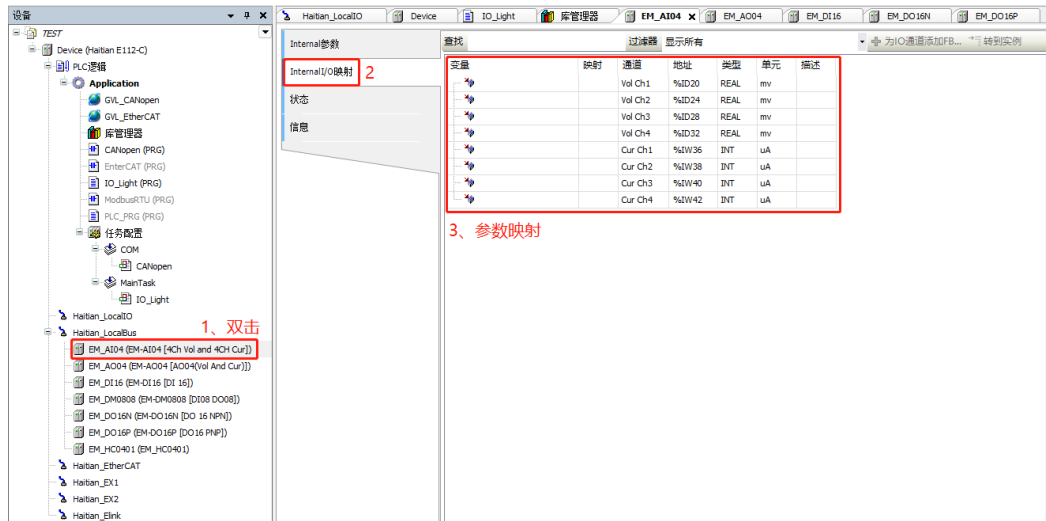
4.3 扩展模块的配置与映射

扩展模块可通过参数配置按需实现对应功能。

4.3.1 EM-AI04 模块

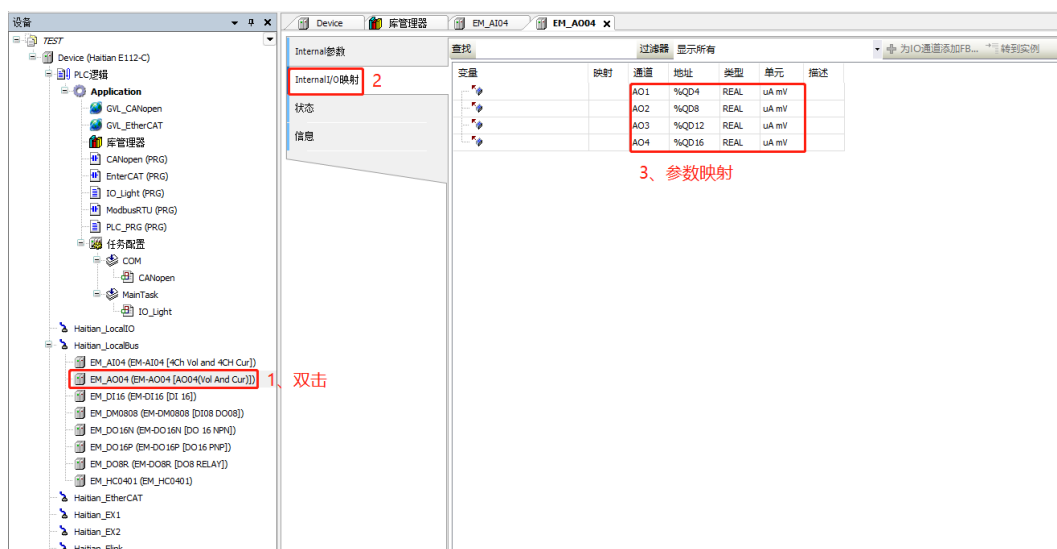
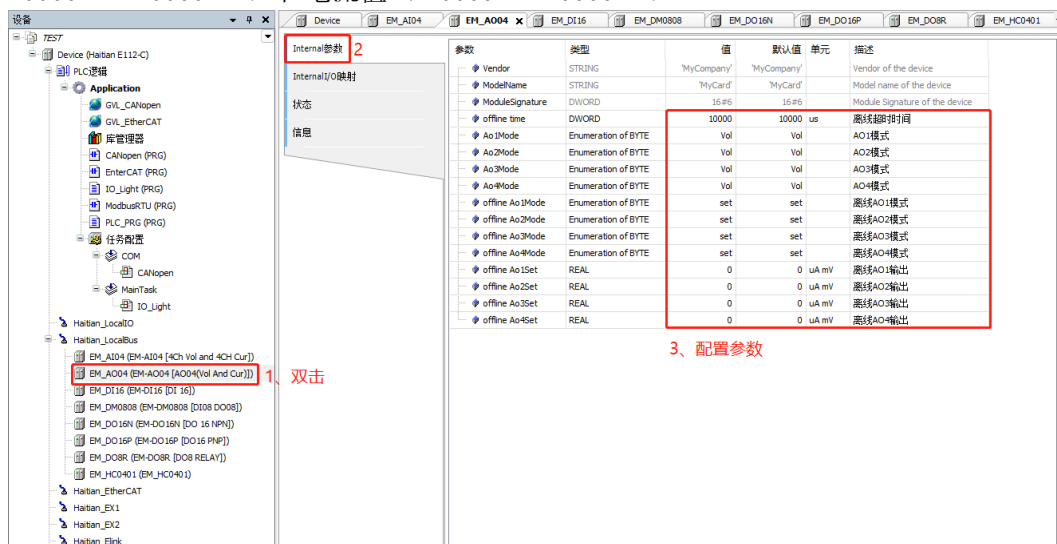
AI04 模块可读取输入的电压/电流模拟量，通过设置“电压/电流模拟量滤波”的次数，次数越多获取到的值越稳定，参数映射可显示对应通道输入的电压电流值。





4.3.2 EM-AO04 模块

AO04 模块可由四个通道输出电压/电流模拟量,将“AO 模式”中的值设置为“Vol”或“Cur”,来控制四个通道的输出模式,在参数映射中可设置对应通道输出的电压值 (-10000mV~10000mV) 和电流值 (-20000uA~20000uA)。



4.3.3 EM-DI16 模块

DI16模块用于读取输入的数字量的状态(TRUE/FALSE),设置不同的通道的滤波时间,时间越长获取到的值越稳定。

参数	类型	值	默认值	单元	描述
Vendor	STRING	'MyCompany'	'MyCompany'		Vendor of the device
ModelName	STRING	'MyCard'	'MyCard'		Model name of the device
ModuleSignature	DWORD	16#1	16#1		Module Signature of the device
offline time	DWORD	10000	10000	us	离线超时时间
CH1 filter	Enumeration of BYTE	FILTER_10MS	FILTER_10MS		CH1 滤波时间(X0~X3)
CH2 filter	Enumeration of BYTE	FILTER_10MS	FILTER_10MS		CH2 滤波时间(X4~X7)
CH3 filter	Enumeration of BYTE	FILTER_10MS	FILTER_10MS		CH3 滤波时间(X8~X11)
CH4 filter	Enumeration of BYTE	FILTER_10MS	FILTER_10MS		CH4 滤波时间(X12~X15)

变量	映射	通道	地址	类型	单元	描述
DI			%DI44	WORD		
DI0			%DI44.0	BOOL		
DI1			%DI44.1	BOOL		
DI2			%DI44.2	BOOL		
DI3			%DI44.3	BOOL		
DI4			%DI44.4	BOOL		
DI5			%DI44.5	BOOL		
DI6			%DI44.6	BOOL		
DI7			%DI44.7	BOOL		
DI10			%DI45.0	BOOL		
DI11			%DI45.1	BOOL		
DI12			%DI45.2	BOOL		
DI13			%DI45.3	BOOL		
DI14			%DI45.4	BOOL		
DI15			%DI45.5	BOOL		
DI16			%DI45.6	BOOL		
DI17			%DI45.7	BOOL		

4.3.4 EM-DO16N 模块

DO16N 模块用于输出数字量(输出 0V 为“TRUE”, 24V 为“FALSE”, DO16P 反之),在“离线 DO 模式”的设置中,“set”表示按程序设定值输出,“keep”表示按模块离线前的值输出。在“离线 DO 功能”的设置中,“invalid”表示初始输出为“0”,“valid”表示初始输出为“1”。

1、双击

2

3、参数配置

参数	类型	值	默认值	单元	描述
Vendor	STRING	'MyCompany'	'MyCompany'		Vendor of the device
ModelName	STRING	'MyCard'	'MyCard'		Model name of the device
ModuleSignature	DWORD	16#2	16#2		Module Signature of the device
offline time	DWORD	10000	10000	us	离线超时时间
offline DoMode	WORD	65535			离线DO模式
Bit0	BOOL	set	set		
Bit1	BOOL	set	set		
Bit2	BOOL	set	set		
Bit3	BOOL	set	set		
Bit4	BOOL	set	set		
Bit5	BOOL	set	set		
Bit6	BOOL	set	set		
Bit7	BOOL	set	set		
Bit8	BOOL	set	set		
Bit9	BOOL	set	set		
Bit10	BOOL	set	set		
Bit11	BOOL	set	set		
Bit12	BOOL	set	set		
Bit13	BOOL	set	set		
Bit14	BOOL	set	set		
Bit15	BOOL	set	set		
offline DoSet	WORD	0			离线DO功能
Bit0	BOOL	invalid	invalid		
Bit1	BOOL	invalid	invalid		
Bit2	BOOL	invalid	invalid		
Bit3	BOOL	invalid	invalid		
Bit4	BOOL	invalid	invalid		
Bit5	BOOL	invalid	invalid		
Bit6	BOOL	invalid	invalid		
Bit7	BOOL	invalid	invalid		
Bit8	BOOL	invalid	invalid		
Bit9	BOOL	invalid	invalid		
Bit10	BOOL	invalid	invalid		
Bit11	BOOL	invalid	invalid		
Bit12	BOOL	invalid	invalid		
Bit13	BOOL	invalid	invalid		
Bit14	BOOL	invalid	invalid		

1、双击

2

3、参数配置

变量	映射	通道	地址	类型	单元	描述
DO	%QW2	WORD				
DO0	%QW2.0	BOOL				
DO1	%QW2.1	BOOL				
DO2	%QW2.2	BOOL				
DO3	%QW2.3	BOOL				
DO4	%QW2.4	BOOL				
DO5	%QW2.5	BOOL				
DO6	%QW2.6	BOOL				
DO7	%QW2.7	BOOL				
DO10	%QW3.0	BOOL				
DO11	%QW3.1	BOOL				
DO12	%QW3.2	BOOL				
DO13	%QW3.3	BOOL				
DO14	%QW3.4	BOOL				
DO15	%QW3.5	BOOL				
DO16	%QW3.6	BOOL				
DO17	%QW3.7	BOOL				

4.3.5 EM-HC0401 模块

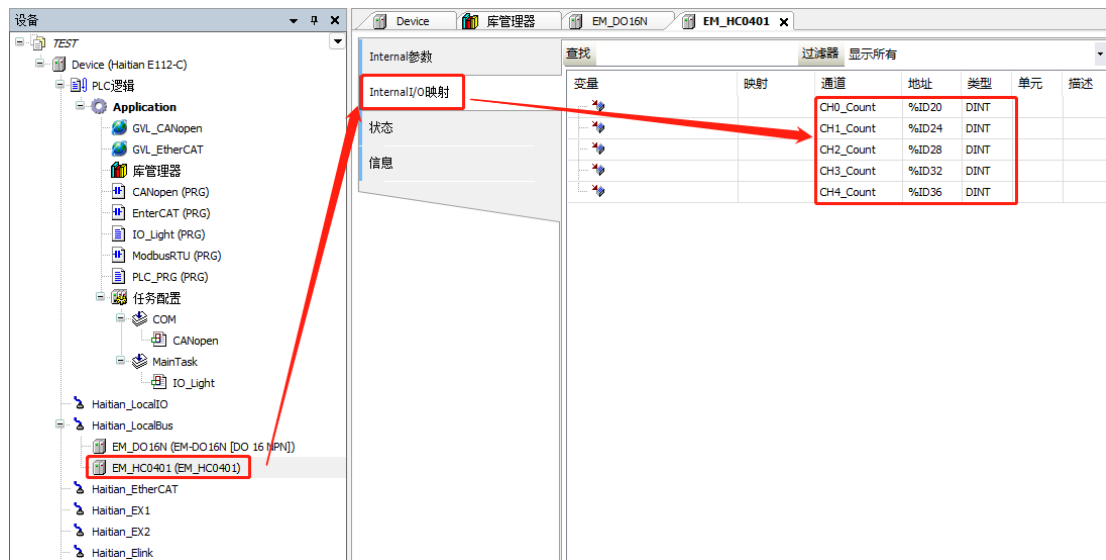
HC0401 模块含有四路单端输入（CH0~CH3）和一个差分输入（CH4），可用于连接编码器读取编码器的脉冲数。

1、双击

2

3、参数配置

参数	类型	值	默认值	单元	描述
Vendor	STRING	'MyCompany'	'MyCompany'		Vendor of the device
ModelName	STRING	'MyCard'	'MyCard'		Model name of the device
ModuleSignature	DWORD	16#8	16#8		Module Signature of the device
offline time	DWORD	10000	10000	us	离线超时时间
CH0_Type	Enumeration of BYTE	AB	AB		
CH1_Type	Enumeration of BYTE	AB	AB		
CH2_Type	Enumeration of BYTE	AB	AB		
CH3_Type	Enumeration of BYTE	AB	AB		
CH4_Type	Enumeration of BYTE	AB	AB		



4.4 选配项

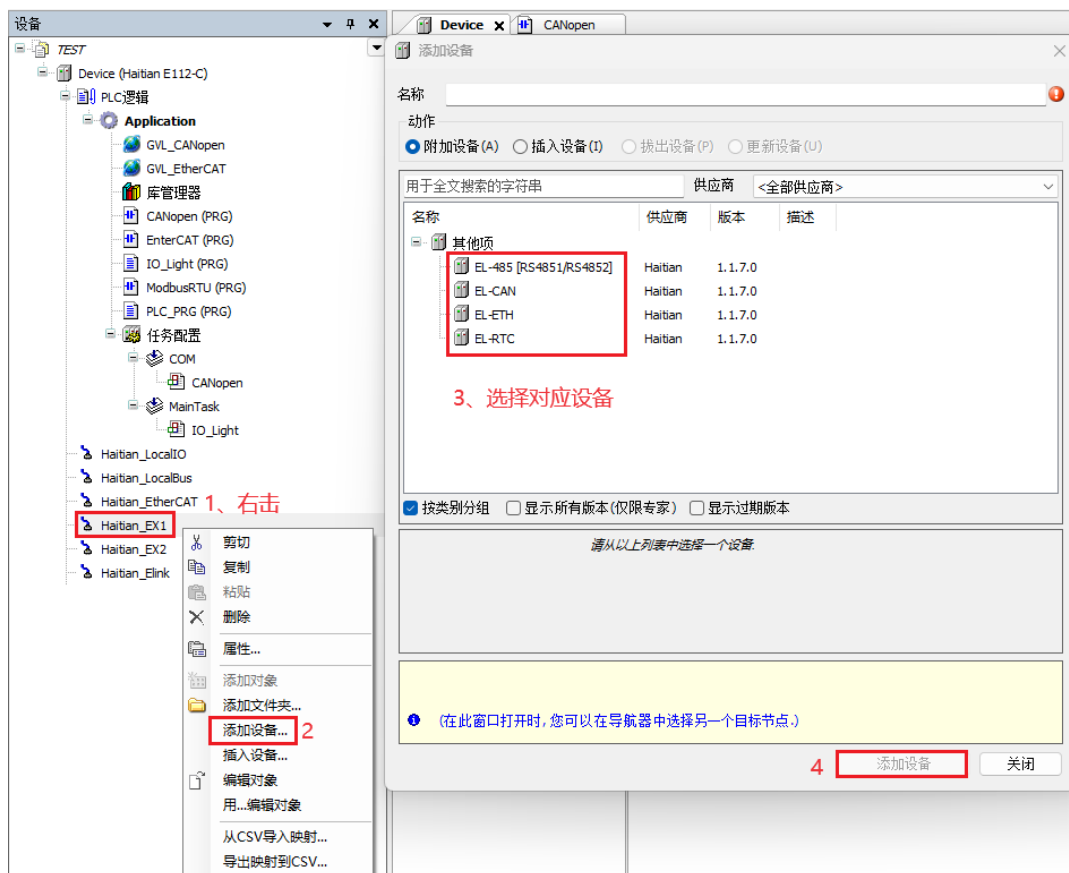
E112-C 本体除右侧可扩展模块外，本体也具备 EX1 和 EX2 两个选配项的扩展口，可供客户按需进行扩展。

4.4.1 选配项类型与组态

除 E112-C 控制器具备的通讯方式外，可通过选配项继续增加通讯通道及方式。

型号	描述
EL-485	485 模块，可扩展两个 485 通道
EL-CAN	CANopen 模块
EL-ETH	通讯网口模块
EL-RTC	

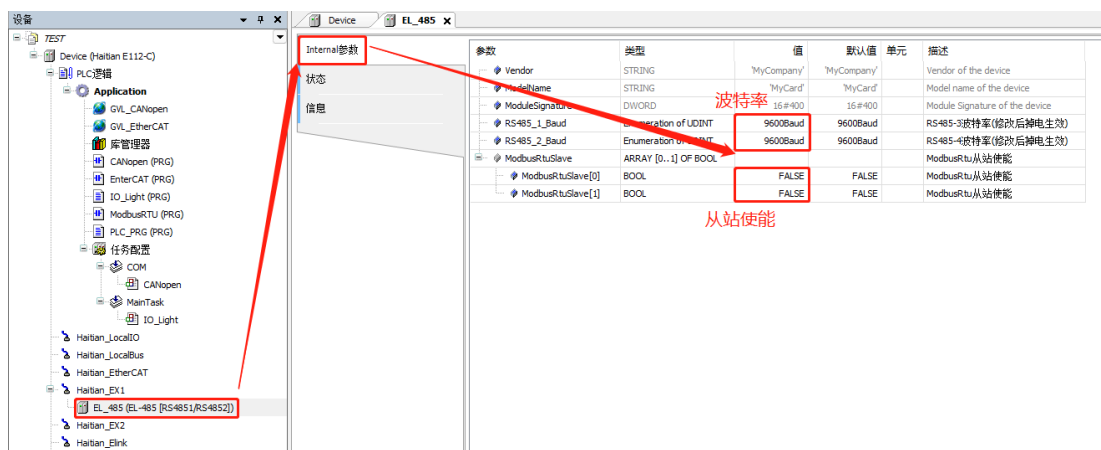
根据实际选配项安装情况，“I ex”对应右击“Haitian_EX1”，“II ex”对应右击“Haitian_EX2”，添加对应设备。



4.4.2 选配项参数设置

4.4.2.1 EL-485 模块

可按需配置扩展的两个 485 通道的波特率以及从站使能。



4.4.2.2 EL-CAN 模块

配置 CAN 通讯的“CAN_Baud”波特率、“SyncCycle”主站同步周期、“DevType”设备类型、“DevNodeID”从站节点 ID、“DevMode”同步 sync/异步 async 模式选择、“DevCycle”从站同步周期、CANopen 延时启动时间、CANopen 离线时间。

1、双击

2

3、CAN参数配置

参数	类型	值	默认值	单元	描述
Vendor	STRING	'MyCo...	'MyCom...		Vendor of the device
ModelName	STRING	'MyCard'	'MyCard'		Model name of the device
ModuleSignature	DWORD	16#403	16#403		Module Signature of the device
CAN_Baud	Enumeration of UDINT	500Kbps	500Kbps	bps	CAN波特率(修改后掉电生效)
SyncCycle	Enumeration of UDINT	2ms	2ms	us	同步周期
DevType	ARRAY [0..7] OF Enumeration of BYTE				
DevType[0]	Enumeration of BYTE	dummy	dummy		
DevType[1]	Enumeration of BYTE	dummy	dummy		
DevType[2]	Enumeration of BYTE	dummy	dummy		
DevType[3]	Enumeration of BYTE	dummy	dummy		
DevType[4]	Enumeration of BYTE	dummy	dummy		
DevType[5]	Enumeration of BYTE	dummy	dummy		
DevType[6]	Enumeration of BYTE	dummy	dummy		
DevType[7]	Enumeration of BYTE	dummy	dummy		
DevNodeID	ARRAY [0..7] OF BYTE				
DevNodeID[0]	BYTE	1	1		
DevNodeID[1]	BYTE	1	1		
DevNodeID[2]	BYTE	1	1		
DevNodeID[3]	BYTE	1	1		
DevNodeID[4]	BYTE	1	1		
DevNodeID[5]	BYTE	1	1		
DevNodeID[6]	BYTE	1	1		
DevNodeID[7]	BYTE	1	1		
DevMode	ARRAY [0..7] OF Enumeration of BYTE				
DevMode[0]	Enumeration of BYTE	sync	sync		
DevMode[1]	Enumeration of BYTE	sync	sync		
DevMode[2]	Enumeration of BYTE	sync	sync		
DevMode[3]	Enumeration of BYTE	sync	sync		
DevMode[4]	Enumeration of BYTE	sync	sync		
DevMode[5]	Enumeration of BYTE	sync	sync		
DevMode[6]	Enumeration of BYTE	sync	sync		
DevMode[7]	Enumeration of BYTE	sync	sync		
DevCycle	ARRAY [0..7] OF Enumeration of UDINT			us	
DevCycle[0]	Enumeration of UDINT	2ms	2ms	us	
DevCycle[1]	Enumeration of UDINT	2ms	2ms	us	
DevCycle[2]	Enumeration of UDINT	2ms	2ms	us	
DevCycle[3]	Enumeration of UDINT	2ms	2ms	us	
DevCycle[4]	Enumeration of UDINT	2ms	2ms	us	
DevCycle[5]	Enumeration of UDINT	2ms	2ms	us	
DevCycle[6]	Enumeration of UDINT	2ms	2ms	us	
DevCycle[7]	Enumeration of UDINT	2ms	2ms	us	

4.4.2.3 EL-ETH 模块

配置网口通讯的 IP 地址和 ModbusTCP 的从站数量。（注意：扩展的网口 IP 地址需和 PLC 本体的 IP 地址不同。）

1、双击

2

3、网口参数配置

参数	类型	值	默认值	单元	描述
Vendor	STRING	'MyCompany'	'MyCompany'		Vendor of the device
ModelName	STRING	'MyCard'	'MyCard'		Model name of the device
ModuleSignature	DWORD	16#403	16#403		Module Signature of the device
ip	ARRAY [0..3] OF BYTE				Ex网口IP地址(修改后掉电生效)
ip[0]	BYTE	172	172		Ex网口IP地址(修改后掉电生效)
ip[1]	BYTE	18	18		Ex网口IP地址(修改后掉电生效)
ip[2]	BYTE	57	57		Ex网口IP地址(修改后掉电生效)
ip[3]	BYTE	66	66		Ex网口IP地址(修改后掉电生效)
netmask	ARRAY [0..3] OF BYTE				Ex网口子网掩码(修改后掉电生效)
netmask[0]	BYTE	255	255		Ex网口子网掩码(修改后掉电生效)
netmask[1]	BYTE	255	255		Ex网口子网掩码(修改后掉电生效)
netmask[2]	BYTE	255	255		Ex网口子网掩码(修改后掉电生效)
netmask[3]	BYTE	0	0		Ex网口子网掩码(修改后掉电生效)
gateway	ARRAY [0..3] OF BYTE				Ex网口网关(修改后掉电生效)
gateway[0]	BYTE	172	172		Ex网口网关(修改后掉电生效)
gateway[1]	BYTE	18	18		Ex网口网关(修改后掉电生效)
gateway[2]	BYTE	57	57		Ex网口网关(修改后掉电生效)
gateway[3]	BYTE	1	1		Ex网口网关(修改后掉电生效)
ModbusTcpSlaveNumEx	Byte	0	0		Ex模块ModbusTcp从站数量

第五章 公共元素及变量

5.1 公共元素

PLC 程序是由一定数量的基本语言元素（最小单元）组成的，把他们组成在一起以形成说明或语句，包括分解符、关键字、标识符和注释。

5.1.1 分界符

分界符用于隔离语言元素的字符或字符组合。它是专门的字符，不同的分界符含义不同，表 4.1 列出各种分界符的应用示例。

表 4.1 分界符

分界符	应用场合	备注或示例
空格	允许在 PLC 程序中插入一个或多个空格	不允许在关键字、文字、标识符和枚举值中直接插入空格
Tab	允许在 PLC 程序中插入 Tab	不允许在关键字、文字、标识符和枚举值中直接插入 Tab
(*	注释开始	用户自定义注释，可在程序允许空格的任何位置输入注释，切 CODESYS 可以通过设置允许注释嵌套
*)	注释结束	
+	十进制数的前缀符号（正数）	+456; +1.23
	加法操作	25+36
-	十进制数的前缀符号（负数）	-789
	年、月、日的分隔符	D#2020-06-06
	减法操作符	36-14
#	基地数的分隔符	2#1001; 16#FF
	数据类型分隔符	INT#16
	时间文字的分隔符	T#1S; TOD#05:30:35:28; t#4m_12s
.	整数和小数的分隔符	3.14; 2.01
	分级寻址分隔符	%IX0.1
	结构元素分隔符	Axis [0]. ActVelocity; abc.number;
	功能块结构分隔符	TON_1.Q
E/e	实数分隔符	1.0e+16; 3.14E6
'	字符串开始/结束符	'Hello World'
\$	字符串中特殊字符的开始	'\$L'表示换行; '\$R'表示回车
:	时刻文字分隔符	TOD#12:21:35.22
	变量/类型分隔符	Bin_0: BOOL
:=	初始化操作符	Var1:int:=3;
	输入变量连续操作符	INT_2(SINGLE :=z2,PRIORITY:=1)
	赋值操作符	Var2:=100;
()	枚举表分隔符	V (B1_10V,UP_10V,IP_10V): =UP_10V
	子范围分隔符	DATA: INT (-32768..32767)
	初始化重复因子	ARRAY (1..2,1...3) OF INT:=1,2,3(4),6
	指令表修正符	(A > B)

	函数自变量	Var*LIMIT (Var1)
	子表达式分级	(A*(B-C)+D)
	功能块输入表分隔符	TON_1 (IN:=%IX5.1,PT:=T#500S) ;
[]	数组下标分隔符	Axis [0]. ActVelocity
	枚举表分隔符	V (B1_10V,UP_10V,IP_10V): =UP_10V
	初始化分隔符	ARRAY (1..2,1...3) OF INT:=1,2,3(4),6
	数组下标分隔符	ARRAY (1..2,1...3) OF INT:=1,2,3(4),6
	被声明变量分隔符	VAR aa,bb :WORD;END_VAR
,	功能块初始化分隔符	TON_1 (IN:=%IX5.1,PT:=T#500S) ;
	功能块输入表分隔符	SR_1(S1:=%IX0.0,RESET:=%IX0.1);
	操作数分隔符	ARRAY (1..2,1...3) OF INT:=1,2,3(4),6
	函数自变量表分隔符	LIMIT(MN:=4,IN:=%IW0,MX:=20);
	Case 值表分隔符	CASE STEP OF 1,5:DISPLAY:=FALSE;
	类型分隔符	TYPE R:REAL;END_TYPE
;	语句分隔符	QU:=5*(A + B);QD:=4*(A - B);
	子范围分隔符	ARRAY(1..2,1..3);
..	Case 范围分隔符	CASE STEP OF 1,5:DISPLAY:=FALSE;
%	直接表示变量的前缀	%ID0
=>	输出连续操作符	C10(CU:=BINPUT,Q=>Out);

5.1.2 关键字

关键字是语言元素特征化的词法单元。在 IEC61131-3 标准中，关键字作为编程语言的字用于定义不同结构或特征的软件元素。

部分关键字需配对使用，如“FUNCTION”与“END_FUNCTION”等，部分关键字可以单独使用，如“ABS”等。关键字不能用于任何其他目的，如不能作为变量名称或者扩展名，也就是说，既不能用 TON 作为变量名也不能用 VAR 做扩展名。

由于关键字是标准的标识符，因此不能包含空格。表 4.2 列出了 CODESYS 中常用的关键字和说明。

表 4.2 常数关键字和说明

关键字	说明	关键字	说明
PROGRAM	程序段开始	EN,ENO	使能输入/输出
END_PROGRAM	程序段结束	TRUE	逻辑真
FUNCTION	函数段开始	FALSE	逻辑假
END_FUNCTION	函数段结束	TYPE	数据类型开始
FUNCTION_BLOCK	功能段开始	END_TYPE	数据类型结束
END_FUNCTION_BLOCK	功能段结束	STRUCTURE	结构体开始
VAR	内部变量开始	END_STRUCTURE	结构体结束
END_VAR	内部变量结束	IF THEN EISIF	If 语句
VAR_INPUT	输入变量开始	ELSE END_IF	
END_VAR	输入变量结束	CASE OF FLASE	CASE 语句
VAR_OUTPUT	输出变量开始	END_CASE	
END_VAR	输出变量结束	FOR TO BY DO	FOR 循环
VAR_IN_OUT	输入输出变量开始		

END_VAR	输入输出变量结束	END_FOR	
VAR_GLOBAL	全局变量开始	REPEAT UNTIL	REPEAT 循环
END_VAR	全局变量结束	END_REPEAT	
CONSTANT	常数变量	WHILE DO	WHILE 循环
		END_WHILE	
ARRAY OF	数组	RETURN	跳转返回符
AT	直接地址	NOT/AND/OR/XOR	逻辑操作符

此外，下列功能模块和函数的标识符也被保留为关键字。

1. 标准数据类型：BOOL、REAL、INT 等；
2. 标准函数和功能块：SIN、COS、RS、TON 等
3. 指令表语言中的文本操作符：LD、ST、ADD、GT 等
4. 结构化文本语言这能怪的文本操作符：NOT、MOD、AND 等。

5.1.3 常数

数值文字用于定义一个数字，他可以是十进制或者其他进制的数。数值文字分为两类：整数文字和实数文字，其书写格式为。

<类型>#<数值>

<类型>:指定所需的数据类型。支持的类型有：BOOL、SINT、USINT、BYTE、INT、UINT、WORD、DINT、UDINT、DWORD、REAL、和 LREAL。

<数值>: 值定常数。输入的数据必须符合指定的数据类型<类型>。

1. 数值常数

数值常数可用二进制、十进制和十六进制数表示。假设一个整数不是十进制数，用户必须在该整数之前写出它的基数并加上数字符号（#），如二进制数 101 的表示方法为 2#101。

数值文字的表示和示例见表 4.3，这些数值可以实变量类型 BYTE、WORD、DWORD、SINT、USINT、INT、UINT、DINT、UDINT、REAL 或 LREAL。

表 4.3 数值常数的表示和示例

数值文字类型	表示方法	示例
整数	[整数类型名#]符号整数或二（八、十、十六）进制整数	INT#-34, UINT#32
	二进制整数	2#00100010（34）
	八进制整数	8#77（63）
	十六进制整数	16#E5（229）
实数	[实数类型名#]符号整数.整数[指数]	REAL#4.96, 6.3e-7
	符号整数.整数	3.14159267
布尔数	[布尔类型名#]0 或 1, False 或 True	0, 1, True, BOOL#1

2. BOOL 常数

BOOL 常数为逻辑值 TRUE（真）与逻辑值 FALSE（假），也可以用 1（真）和 0（假）表达，其意义相同。

3. TIME 常数

TIME 常数由一个其实符 T 或 t（或者 TIME 或 time）和数字标识符#，再加上实际时间表示，包括天（d）、时（h）、分（m）、秒（s）和毫秒（ms）。注意时间各项必须根据时间长度单元顺序进行设置顺序级别依次为（d>h>m>s>ms）单无需包含所有的时间长度。时间长度示例如表 4.4 所示。

表 4.4 时间长度示例

正确示例	错误示例
TIME1: =T#6m8S	TIME1: =T#5m68s (最低单位值溢出)
TIME1: =T#100S12MS	TIME1: = 100S12MS (没有 T#)
TIME1: =T#12h34m15s	TIME1: =T# 15s 12h34 m (输入的顺序错误)

4. DATE 常数

这些常数可以用来表示日期。声明一个 DATA 常数，起始符为 d、D、DATA 或者 data 后跟一个#，然后就可以按照“年-月-日”的格式表示任何日期。例如：

DATA: 2020-06-01

D#2020-06-01

5. TIME_OF_DAY 常数

使用这种类型可以保存一天中的不同时间。TIME_OF_DAY 常数的声明使用起始符 tod#、TOD#、TIME_OF_ADY#或 time_of_day#，后跟随一个时间格式为“时：分：秒”的时间。值可以是整数或者小数。例如：

TIME_OF_DAY#15:36:30.123

tod#00:00:01

6. DATA_AND_TIME 常数

日期常数和一天中的时间常数可合并起来构成一个所谓的 DATA_AND_TIME 常数。DATA_AND_TIME 常数的起始符为 dt#、DT#、DATA_AND_TIME#或 data_and_time#然后在日期之后用“-”字符连接时间，如：

DATA_AND_TIME#2020-06-01-20:00:00

dt#2020-06-01-20:00:00

7. REAL/LREAL 常数

REAL 和 LREAL 常数可使用十进制小数和指数形式表示，使用带小数点的美国格式表示实数 (REAL/LREAL)，如

7.4 取代 7,4

1.64e+009 取代 1,64e+009

8. STRING 常数

字符串是一个字符队列。STRING (字符串) 常数使用一个单引号作为其前缀和后缀。也可以输入空格和专用字符，这些字符将通所有其他字符一样进行处理，字符的表达示例如下：

‘CODESYS!!’

‘3s’

‘:-)’

9. 类型符

通常对 IEC 常数，尽可能少使用数据类型。如果必须使用另一种数据类型，则可借助类型而不需要显式声明常数。为此常数可以使用一个前缀表示，该前缀决定了其数据类型。格式如下：

<Type>#<Literal>

5.1.4 句法颜色

所有编辑器中不通类型文本有不同颜色，不但可以辨识错误，而且有助有快速开发。例如，注释没有被括上，从字体颜色上马上就会得到提示。句法颜色示例如图 4.1 所示

蓝色：关键字

绿色：编辑器中的注释

粉红色：特殊常数（如 TRUE/FALSE、T#3S、%IX0.0）。

红色：输入错误。

黑色：变量、常数、标点符号。

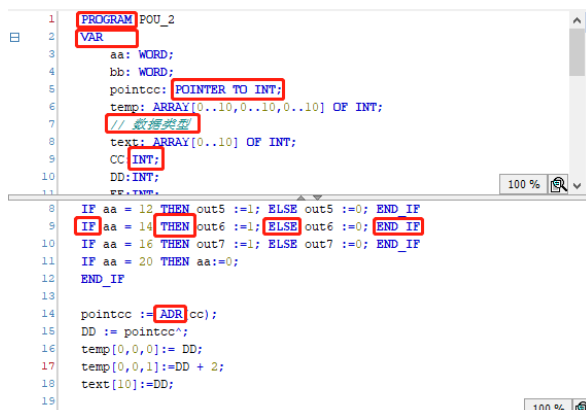


图 4.1 句法颜色

5.1.5 空格和注释

1. 空格

在程序文本的任何地方允许插入一个或多个空格。在关键字、标识符、分界符等内部不允许包含空格。表 4.5 是空格和不允许空格的示例。

表 4.5 空格应用示例

状态	示例
允许空格	LD%IX0.2: SR1 (SET1 : =Start, Reset: =Stop);
不允许空格	L D%IX0.2: SR1 (S ET1 : =Start, Reset: =Stop);

2. 注释

在程序中我们通常在逻辑性较强的地方插入注释，以标准这段程序的逻辑是怎么实现的，以便于自己或他们理解。合理的注释可提高程序代码的可读性。

(1) 结构化文本（ST）的注释

括号注释：以（*开始，以*）结束，可多跨行注释。

单行注释：以//进行标注没改注释方式只可注释一行。

(2) FBD、LD 和 IL 中的注释

依次选择“工具”→“选项”→“FBD、LD、和 IL 编辑器”，然后在“常规”选项卡中

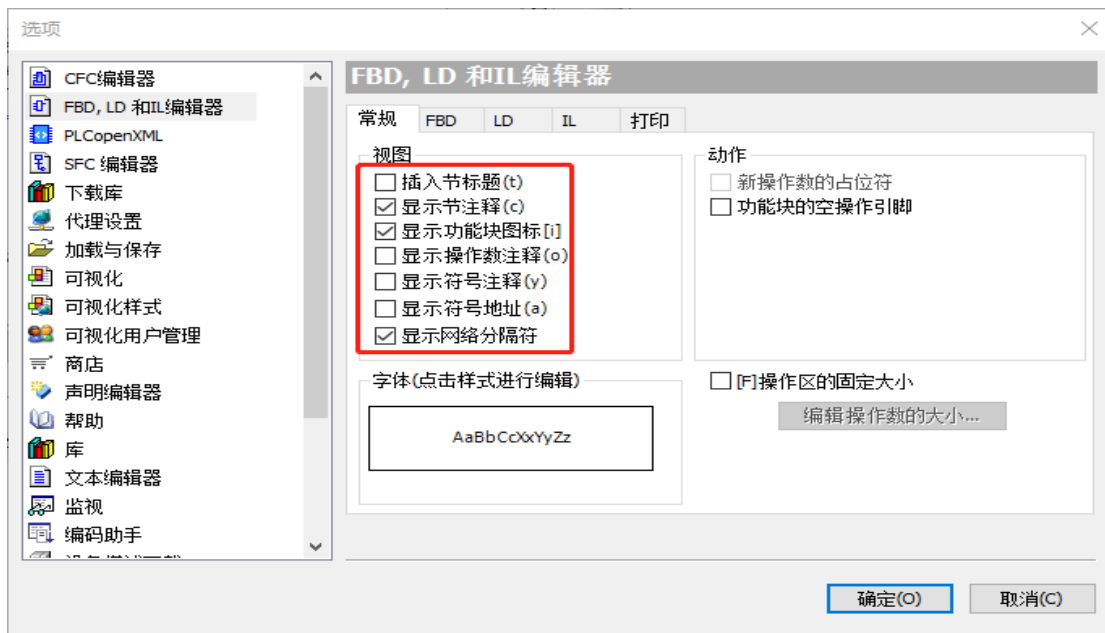


图 4.2 注释常规设置

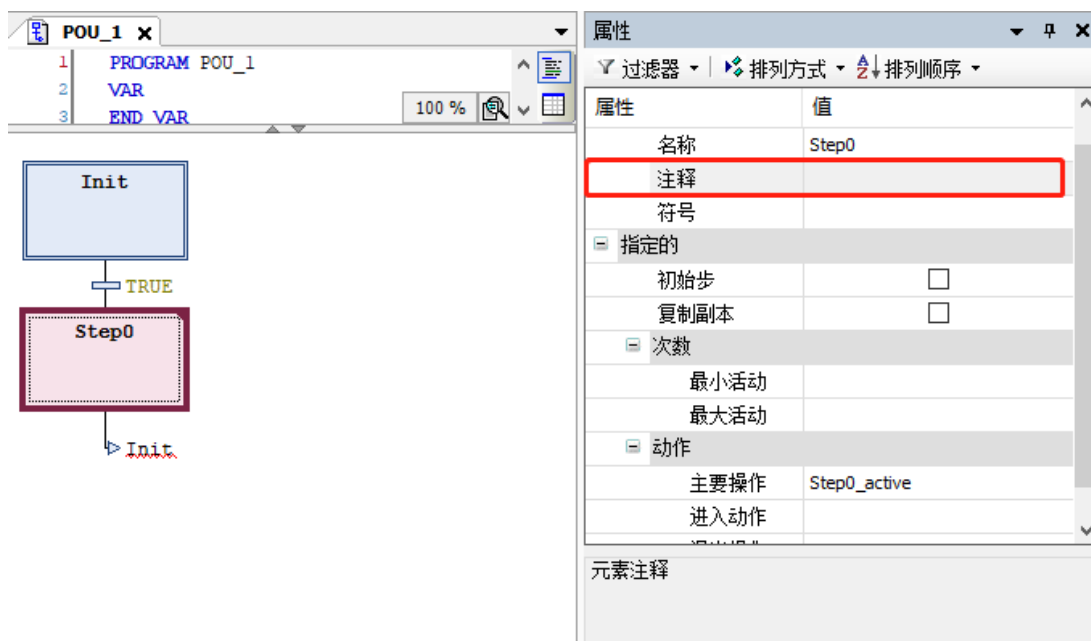
对 FBD、LD 和 IL 编辑语言进行注释视图设置，如图 4.2 所示

(3) CFC 中的注释

在 CFC 中可以随意放置注释选择工具箱中“ 注释”选项可进行添加注释。

(4) SFC 中的注释

在 SFC 编程语言中，能够在编辑步属性对话框中输入关于步的注释，如图 4.3 所示。



(5) 注释字体设置

(6) 一次选择“工具”→“选项”→“语法高亮显示”，在“语法高亮显示”界面中可以设

置注释的字体及颜色。

5.2 变量的表示和声明

变量用来表示一个数值、一个字符串或者一个数组等。CODESYS 将变量的数据类型分为标准类型、扩展类型及自定义类型。

5.2.1 变量

变量时保存在存储器中待处理的抽象的数据，是为了识别 PLC 的输入/输出、PLC 内部的存储区域而使用的名称，可以代替物理地址在程序中使用。

可以根据是要随时修改变量中所存储的数据，在程序执行过程中，变量的只可以发生变化，使用变量钱必须声明变量，以指定变量的类型和名称。变量具有名称、类型和值。变量的数据类型确定它所代表的内存大小和类型。变量名即指在程序源代码中的标识符。

5.2.2 标识符

标识符就是变量的名称，在定义标识符时，根据 IEC61131-3 标准，必须由字母、数字和下划线组成。此外，还应遵循如下规则。

标识符首字符必须是字母或者下划线，最后一个字符必须是字母或数字。中间允许字母数字下划线。

标识符中不区分字母大小写

下划线是标识符中的一部分，单标识符中不允许有连续两个或两个以上的下划线。

不得含有空格。

5.2.3 变量声明

变量声明就是指变量的名称、类型、以及赋初始值。变量的声明非常重要，使用未经声明的变量时不能够通过编译的，也无法保存在程序中。用户可以在程序组织单元（POU）、全局变量列表（GVL）中声明。

1. 普通变量声明

普通变量不需要和硬件或通讯进行关联的变量，仅供内部逻辑使用。普通变量符合以下规则：

<标识符>:<数据类型> {:=<初始值>};

2. 直接变量声明

当变量需要和 PLC 的 I/O 或者通讯进行直接映射时，需要采用此声明方式。使用关键字“AT”把变量直接连接到确定的地址。直接变量的声明符合以下规则。

AT<地址>;

<标识符> AT <地址>:<数据类型> {:=<初始值>};

使用“%”开始，随后是位置前缀符号和类型前缀符号，如果有分级，则用整数表示分级，并用小数点符号“.”表示，如%IX0.0，%QW0.直接变量声明的具体格式如图 4.4 所示。

图 4.4 中的位置前缀有如下定义：

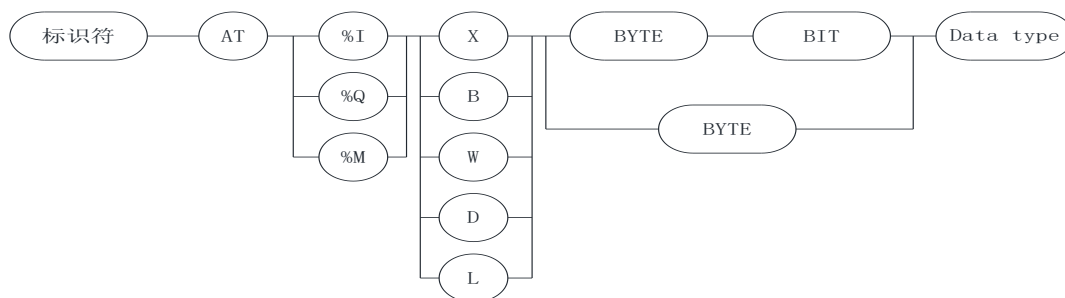


图 4.4 直接变量声明具体格式

I：表示输入单元

Q：表示输出单元

M：表示存储单元

类型前缀定义见表 4.6，内存映射示例见表 4.7

表 4.6 类型前缀定义

前缀符号	定义	约定数据类型
X	位（BIT）	BOOL
B	字节（BYTE）	BYTE
W	字（WORD）	WORD
D	双字（DWORD）	DWORD
L	长字（LWORD）	LWORD
*	为指定的内部变量，系统自动分配	

表 4.7 内存映射示例（字节）

%IX	12.0~12.7	13.0~13.7	14.0~14.7	15.0~15.7
%IB	12	13	14	15
%IW	12		14	
%ID	12			

5.3 数据类型

数据类型的标准化是编程语言开放的重要标志，CODESYS 的数据类型完全符合 IEC61131-3 所定义的标准，它将数据类型分为标准类型、扩展类型、和自定义类型。数据的类型将决定了它占用多大内存空间以及何种类型的值。

5.3.1 标准数据类型

标准数据类型分为 5 大类，分别是布尔类型、整型类型、实数类型、字符串类型和时间数据类型。表 4.8 列举了支持的标准数据类型。

表 4.8 标准数据类型

数据大类	数据类型	关键字	位数	取值范围
布尔	布尔	BOOL	1	FALSE (0) 或 TRUE (1)
整型	字节	BYTE	8	0~255
	字	WORD	16	0~65536
	双字	DWORD	32	0~4294967295
	长字	LWORD	64	0~($2^{64}-1$)
	短整型	SINT	8	-128~127
	无符号短整型	USINT	8	0~255
	整型	INT	16	-32768~32767
	无符号整型	UINT	16	0~65535
	双整型	DINT	32	$-2^{31} \sim 2^{31}-1$
	无符号双整型	UDINT	32	0~4294967295
实数	长整型	LINT	64	$-2^{63} \sim 2^{63}-1$
	实数	REAL	32	1.175494351e-38~3.402823466e+38
	长实数	LREAL	64	2.225.7385850e-308~1.7976931e+38
字符串	字符串	STRING	8×N	
时间	时间	TIME	32	T#0S~T#71528m47s295ms
		TIME_OF_DAY		TOD#0:0:0~TOD#1193:02:47: .295
		DATE		D#1970-01-01~2106-02-06
		DATA_AND_TIME		DT#1970-1-1-0:0:0~DT#2106-2-2-06:28:15

1. 布尔

布尔型变量用来表示 TRUE/FALSE 值，一个布尔型变量只有 TRUE 或 FALSE 两种状态，也可以用 0 或 1 来表示。

2. 整型

整型类型代表了没有小数点的整数类型，支持的整数类型如表 4.8 所示。

U 表示无符号数据类型。为 Unsigned 的缩写。

S 表示短数据类型，为 Short 的缩写。

D 表示双数据类型，为 Double 的缩写。

L 表示长数据类型，为 Long 的缩写。

3. 实数

实数也称为浮点数，用于处理含有小数的数值数据。实数类型包含 REAL 和 LREAL。REAL 实数占 32 位存储空间，而 LREAL 长实数占 64 位存储空间。在软件中，实数和长实数常量有两种表示形式。

(1) 十进制小数：由数字和小数点组成。例如：1.23

(2) 指数形式：123e3 或 123E3 都代表 123×10^3

4. 字符串

一个字符串是一个字符串队列，字符串常数使用单引号作为其前缀和后缀，其中可以输入空格和专用字符，这些字符将通所有其他字符一样进行处理。

5. 时间数据

时间数据包含 TIME、TIME_OF_DAY/TOD、DATE 和 DATA_AND_TIME/DT。系统内部处理这些数据的方式与双字近似。

- (1) TIME: 时间, 精度为毫秒, 范围为 0~71582m47s295ms。
- (2) TIME_OF_DAY/TOD: 时刻, 精度为毫秒, 范围为 0:0:0~1193:02:47.295.时刻声明使用“时: 分: 秒”格式。
- (3) DATE: 日期, 精度为天, 范围为 1970-01-01~2106-02-06。日期声明使用“年-月-日”格式
- (4) DATA_AND_TIME/DT, 日期和时间, 精度为秒, 范围为 1970-01-01-00:00:00~2106-02-06-06:28:15, 日期和时间声明使用“年-月-日-时: 分: 秒”的格式。

5.3.2 标准扩展数据类型

作为对 IEC61131-3 标准中数据类型的补充, CODESYS 隐含一些标准的扩展数据类型、有联合体、长时间、宽字节字符串、引用和指针等, 详见表 4.9 所示

表 4.9 IEC61131-3 标准的扩展数据类型

数据大类	数据类型	关键字	位数	取值范围
联合	联合体	UNION		自定义
时间数据	长时间	LTIME	64	纳秒~天
字符串	宽字节字符串	WSTRING	16×(N+1)	
引用	引用	REFEREFCE TO		自定义
指针	指针	POINTER TO		自定义

1. 联合体

有时候需要将集中不同类型的变量存放在同一段内存单元中, 如果可以把一个 INT 型变量、一个 BYTE 型变量和一个 DWORD 型变量放在同一地址开始的内存单元中。

2. 长时间

提供长时间类型数据作为高精度计时器的时间基量。它与 TIME 类型不同的是, TIME 的长度为 32 位切精度为毫秒 (ms) LTIME 的长度为 64 位且精度为纳秒 (ns)

3. 宽字符串

与字符串类型数据 (ASCII) 不同的是, 这一数据类型由 Unicode 解码。每个字符串占用的存储空间大小为 2×N+2。

4. 引用

引用是一个对象发的别名, 这个别名可以通过变量名读写, 它与指针的区别是, 引用所指向的数据将被直接改变, 因此引用的赋值和所指向的数据是相同的, 设置引用的地址时, 可用一个特定的赋值操作完成。一个应用是否指向一个有效的数据 (不等于 0), 可以使用一个专门的操作符来检查。

5. 指针

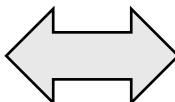
(1) 指针的概念

所谓指针就是一个地址。如果在程序中定义了一个变量, 然后对其进行编译, 系统则会给这个变量分配相应的内存单元并根据变量的类型分配相应的内存空间, 如个 BYTE 类型变量, 系统则为其分配 1 个字节的存储空间, 如果是一个 REAL 类型变量, 系统则为其分配 4 个字节的内存存储空间。内存以字节为单位, 以“地址”进行编号, 可以理解为酒店的房间号, 地址所标识的内存单元中存放着具体的数据, 这就相当于在宾馆中居住的客人。

一般而言, 程序都是通过变量名对内存单元进行存储操作的, 这是因为程序经过编译后,

已经将变量名转换为变量的内存地址，对变量的存储其实都是通过内存地址进行的。假设在程序中定义了 var1、var2 和 var3 这 3 个变量，在声明时将其都定义为 WORD 类型，经过系统编译后，系统分配给 var1 的内存空间为 2 字节，地址分别为 1000 和 1001, var2 为 1002 和 1003, var3 为 1004 和 1005。1000 和 1001 内存的具体数据原则是 var1 内的具体数据，var2 和 var3 也是相同的道理，示意图如表 4.10 所示

表 4.10 变量名对应内存地址查询变量

变量名	数据内容		内存地址	数据内容
Var1	Byte_Low		1000	Byte_Low
	Byte_High		1001	Byte_High
Var2	Byte_Low		1002	Byte_Low
	Byte_High		1003	Byte_High
Var3	Byte_Low		1004	Byte_Low
	Byte_High		1005	Byte_High

5.3.3 自定义数据类型

1. 数组

数组类型在 CODESYS 中被大量使用，使用数组可以有效地处理大批量数据，可以大大提高工作效率。数组是有序数据的结合，其中的每个元素都拥有相同的数据类型。

在 CODESYS V3 版本中可以定义一维、二维、三维数组作为数据类型。用户可以在 POU 的声明部分或全局变量中定义数组，声明数组的语法如下。

<数组名>: ARRAY[<I1>..I1>,<I2>..I2>,<I3>..I3>]OF<基本数据类型>

(1) 一维数组定义

一维数组是常用的一种数组类型，一维数组定义示例如下。

VAR

text: ARRAY[0..10] OF INT;

END_VAR

(2) 二维数组

二维数组可以看成是一个特殊的一维数组，他自身元素可以理解为一个新的数组。例如，可以把 a 看成是一个一维数组，它有 3 个元素：a[0]、a[1]、a[2]，而其中的每一个元素又可以包含 4 个元素的一维数组，

$$a \begin{cases} a[0] - - - - - a_{00} & a_{01} & a_{02} & a_{03} \\ a[1] - - - - - a_{10} & a_{11} & a_{12} & a_{13} \\ a[2] - - - - - a_{20} & a_{21} & a_{22} & a_{23} \end{cases}$$

二维数组应用示例

VAR

text: ARRAY[0..10,0..10] OF INT;

END_VAR

(3) 三维数组

理解二维数组，三维数组就不难理解了。下面将为三维数组定义示例

VAR

temp: ARRAY[0..10,0..10,0..10] OF INT;

END_VAR

2. 结构体

目前，已经介绍了基本类型的变量(如整型、实数和字符串等)，也介绍了数组，数组中的各个元素属于同一个类型。但仅有这些数据类型是不够的，有时需要将不同类型的数据组合成一个整体以便于引用。结构体(Struct)就是由一系列具有相同类型或不同类型的数据构成的数据集合。

结构体声明的语法如下

TYPE <结构体名>

STRUCT

<变量声明 1>

<变量声明 2>

<变量声明 n>

END_STRUCT

END_TYPE

结构体名是一种可以在整个工程中被识别的数据类型，可以像标准数据类型一样使用。结构体可实现嵌套。

添加结构体

右键单击“Application”在弹出的快捷菜单中依次选择“添加对象”→“DUT...”在“添加DUT”对话框中输入结构体名称，并选中“结构(S)”单选按钮，然后点击“打开”按钮，如图 4.5 所示。然后系统会自动进入 DUT 编辑器，如图 4.6 所示

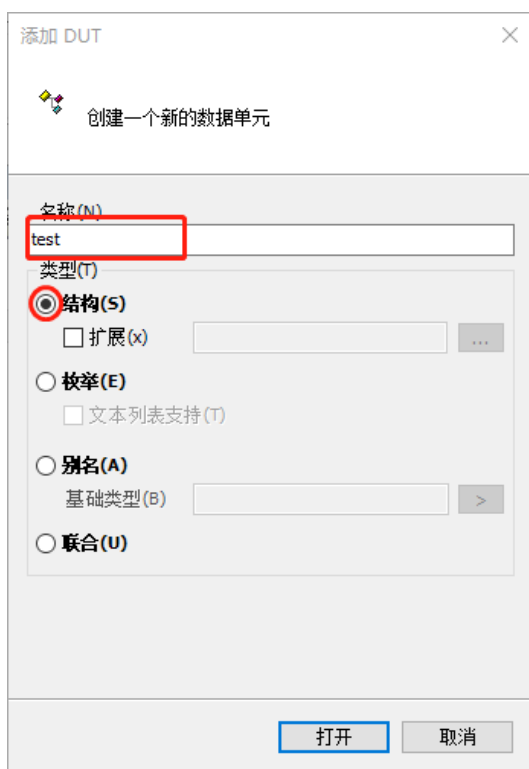


图 4.5 创建结构体

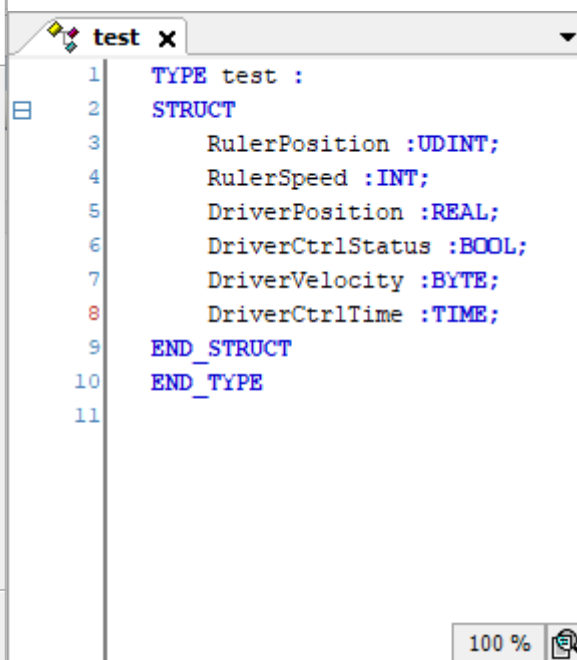


图 4.6 DUT 编辑器

结构体建立完成后，只需要在程序中新建一个变量，类型为刚刚新建结构体的名称，在程序中“变量名.”系统就会总出现结构体内对应的信息，通过鼠标选择，在配合赋值语句使用即可。

3. 枚举

如果种变量有几种可能的值，就可以定义为枚举类型。所谓“枚举”是将变量的值一一列举出来，变量的值只能在列举出来的值的范围内。例如，定义一个变量，该变量的值表

示一星期中的一天。该变量只能存储 7 个有意义的值。若要定义这些值，可以使用枚举类型。例如，一星期内可能取值的集合为：

{Sun, Mon, Tue, Wed, Thu, Fri, Sat}

枚举类型创建

右键单击“Application”在弹出的快捷菜单中依次选择“添加对象”→“DUT...”在“添加 DUT”对话框中输入结构体名称，并选中“枚举（E）”单选按钮，然后点击“打开”按钮，如图 4.7 所示。然后系统会自动进入枚举编辑器，如图 4.8 所示

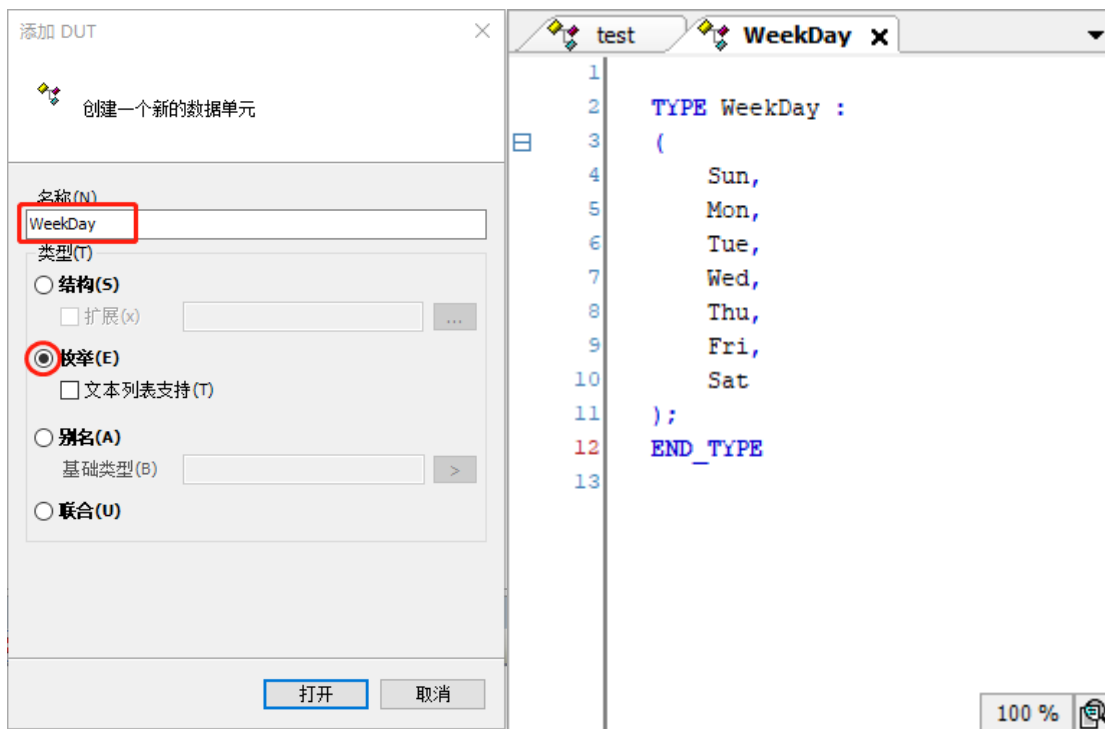


图 4.7 新建枚举

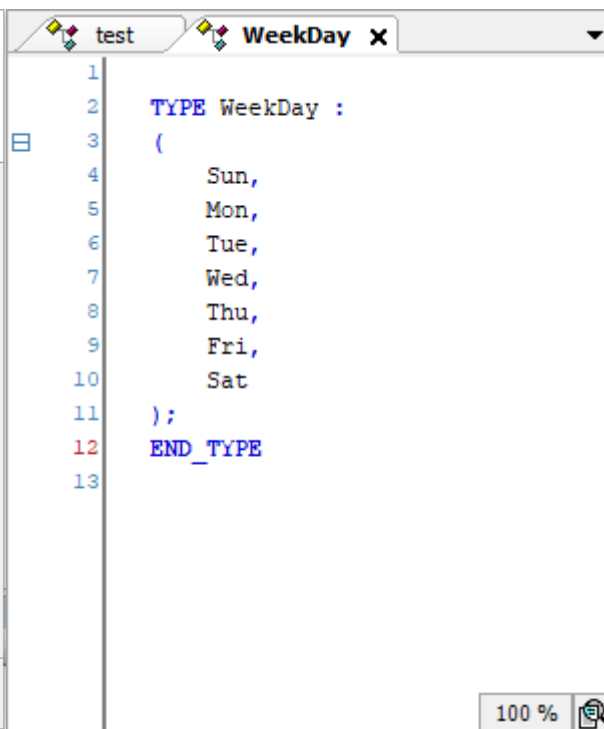


图 4.8 枚举编辑器

- 1) 枚举类型的基本数据类型默认是 INT 类型，也可由用户明确指定。
- 2) 枚举类型可以直接用来做判断条件，
- 3) 整数可以直接复制给一个枚举变量
- 4) 同样 POU 内，同样的枚举不得用两次，

5.4 变量的类型和初始化

CODESYS 根据 IEC61131-3 标准对变量定义了属性，通过设置变量属性将他们的有关性能赋予变量。变量可根据其应用范围进行分类，出变量之外，CODESYS 还提供了变量的附加属性。

5.4.1 变量的类型

CODESYS 支持的变量属性见表 4.11

表 4.11 变量属性

变量类型关键词	变量属性	外部读写	内部读写
---------	------	------	------

VAR	局部变量		R/W
VAR_INPUT	输入变量，由外部提供	R/W	R
VAR_OUTPUT	输出变量，由内部变量提供给外部	W	R/W
VAR_INOUT	输入-输出变量	R/W	R/W
VAR_GLOBAL	全局变量，能在所有的配置、资源中使用	R/W	R/W
VAR_TEMP	临时变量，程序或功能块内部存储使用的变量		R
VAR_STAT	静态变量		
VAR_EXTERNAL	外部变量，能在程序内部修改，但需由全局变量提供	R/W	R/W

在 CODESYS 中，变量附加属性见表 4.12

表 4.12 变量附加属性

变量附加属性关键字	变量附加属性
RETAIN	保持性变量，用于掉电保存
PERSISTENT	保持性变量，用于掉电保存
VAR RETAIN PERSISTENT VAR PERSISTENT RETAIN	两者功能一样，皆为保持性变量，用于掉电保存
CONSTANT	常量

5.4.2 变量的初始化

在程序中，有时候需要对一些变量预先赋予初值，CODESYS 允许在定义变量时对变量进行赋初值处理，在赋值运算符“:=”的左边是变量及变量的类型，该变量接受右边地址或表达式的值，例如：

```
VAR
    w_aa :WORD := 100;
    b_bb :BYTE := 20;
    B_cc :BOOL := TRUE;
    i_arr:ARRAY[1..5] OF INT := [1,2,3,4,5];
END_VAR
```

用户自定义初始值在控制器刚启动的第一个任务周期对该变量写一次。在程序中，如果直接在变量声明区修改了初始值，“登录到”PLC 后选择“在线修改”并不会带该变量的数据有任何影响，只有当 PLC 复位后重新运行程序才有用。

第六章 常用指令

6.1 运动功能块

6.1.1 概览

HIC 控制器运动功能块主要分为运动功能块和管理功能块。如表 1.1 所示

表 1.1 运动功能块概览

Administrative(管理功能块)		Motion(运动功能块)	
SignalAxis(单轴)	MultipleAxis(多轴)	SignalAxis(单轴)	MultipleAxis(多轴)
MC_Power (上使能)	MC_AddAxisToGroup (将轴加入轴组)	MC_Home (回零)	MC_CearIn (电子齿轮耦合)
MC_Reset (复位)	MC_RemoveAxisFromGroup (将轴从轴组中移除)	MC_Stop (停止)	MC_CearInPos (电子齿轮同步耦合)
MC_SetPosition (坐标偏移)	MC_UngroupAllAxes (解除轴组)	MC_MoveAbsolute_L REAL (绝对运动高精度)	MC_CearInPos_LREAL (电子齿轮同步耦合高精度)
MC_ReadStatus (读状态)	MC_GroupEnable (轴组上使能)	MC_MoveAbsolute (绝对运动)	MC_CearOut (电子齿轮解耦)
	MC_GroupDisable (轴组下使能)	MC_MoveRelative (相对运动)	MC_GroupHome (轴组回零)
	MC_GroupReset (轴组复位)	MC_MoveVelocity (连续运动)	MC_GroupStop (轴组停止)
		MC_MoveContinuousAbsolute (绝对定位连续运动)	MC_MoveLinearAbsolute (直线绝对运动)
		MC_MoveContinuousRelative (相对定位连续运动)	MC_MoveLinearRelative (直线相对运动)
			MC_MoveCircularAbsolute (圆弧绝对运动)
			MC_MoveCircularRelative

(1) 单轴状态图

每个轴始终位于状态图中的一个状态 AxisState: 0 Disabled 未使能; 1 StandStill 静止; 2 ErrorStop 错误停止; 3 Stopping 停止中 ; 4 Homing 回原点; 5 DiscreteMotion 离散运动 ; 6 ContinuousMotion 连续运动; 7 SynchronizedMotion 同步运动。如图 1.1 所示。

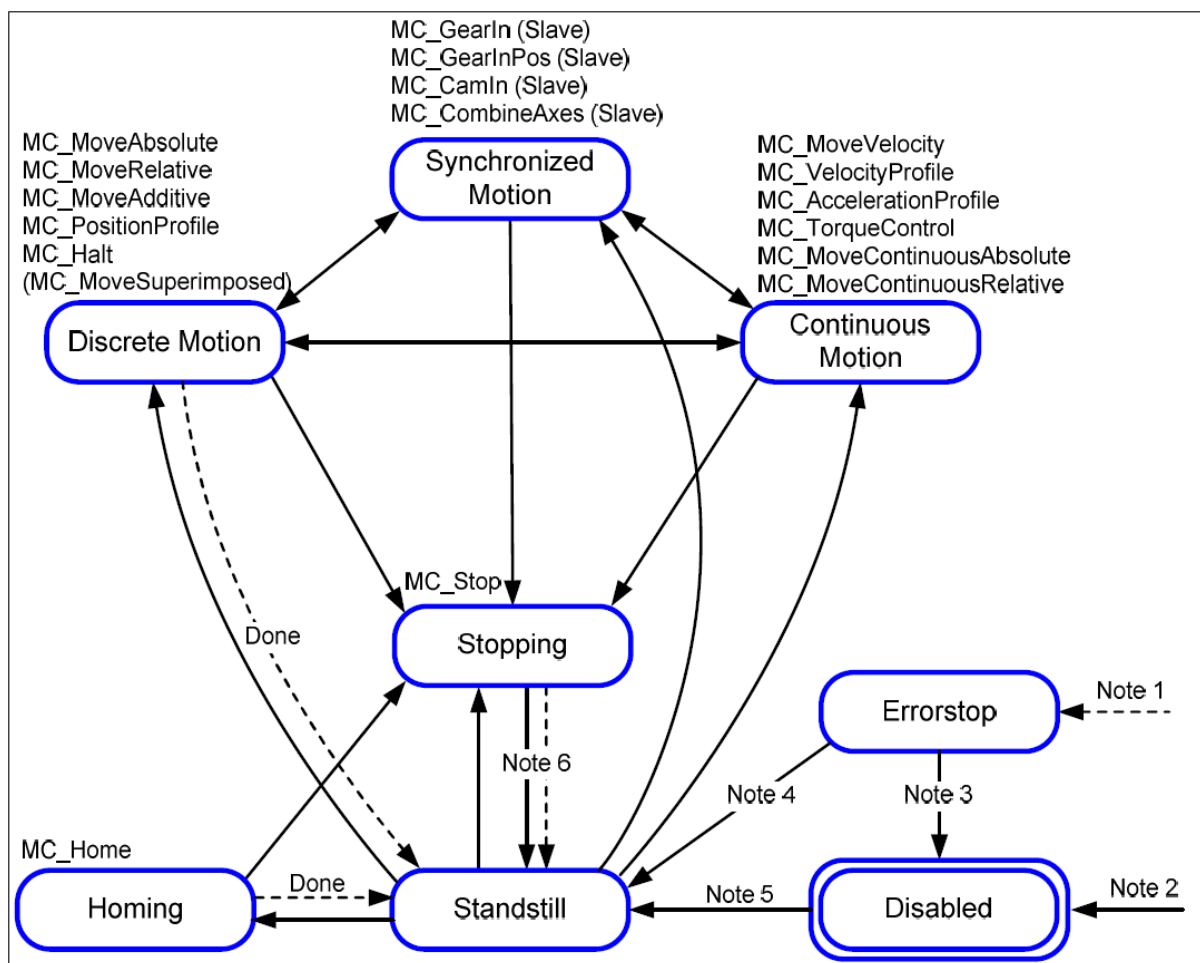


图 1.1 MC 单轴状态图

Note 4: MC_Reset AND MC_Power.Status = TRUE AND MC_Power.Enable = TRUE

Note 6: MC Stop.Done = TRUE AND MC Stop.Execute = FALSE

(2) 轴组状态图

每个轴组始终位于状态图中的一个状态 groupStatus: 0 GroupDisabled 未使能; 1 GroupStandBy 静止; 2 GroupErrorStop 错误停止; 3 GroupStopping 停止中; 4 GroupHoming 回原点; 5 GroupMoving 运动。如图 1.2 所示。

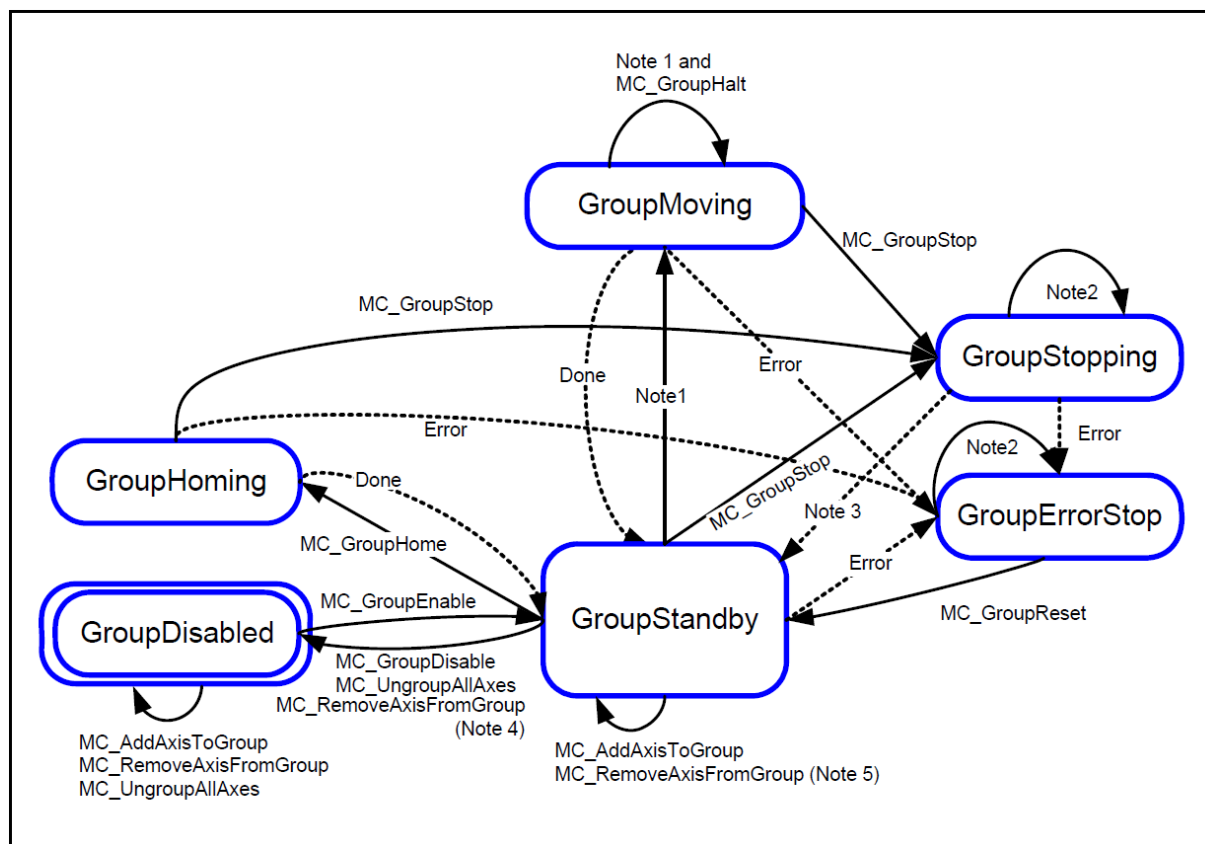


图 1.2 MC 轴组状态图

Note 1:适用于所有的运动型功能块:

Note 2: 在轴组 GroupErrorStop 或 GroupStopping 状态下, 任何功能块都可被调用, 但无法执行除了作用于 GroupErrorStop 状态的 MC_GroupReset 功能块。

Note 3: MC GroupStop.DONE = TRUE AND MC GroupStop.EXECUTE = FALSE.

Note 4:适用于 MC RemoveAixsFromGroup 将轴从轴组中移除后轴组为空。

Note 5:适用于 MC RemoveAixsFromGroup 将轴从轴组中移除后轴组不为空。

Note 6: MC_GroupDisable 和 MC_UngroupAllAxes 可在任何轴组状态下执行，并且将轴组切换到 GroupDisabled 状态。（暂时 MC_UngroupAllAxes 不支持）

6.1.3 功能块输入输出规则

PLCopen 功能块的输入输出状态需要满足标准的规则，如下表 1.2 所示。

表 1.2 功能块输入输出规则

参数类型	规则
输出 Output	•输出 Done、InVelocity、Error、ErrorID 和 CommandAborted 在输入 Execute 下降沿时被复位。注意：它们至少保持设置一个 PLC 周期（不管输入 Execute 已在此之前被复位与否）。 •输出 Busy、Done、Error 和 CommandAborted 在同一个功能块上同一时刻这些输出中只能有一个为 TRUE，当 Execute 为 TRUE 时，这些输出中必须有一个也为

	TRUE; •输出 Active、Error、Done 和 CommandAborted 中只能有一个在同一时间被设置。 •输出 Busy: 设置: Execute 上为上升沿; 复位: 输出 Done、CommandAborted 或 Error 中的一个为 TRUE。 •输出 Active: 只在具备输入 BufferMode 的功能块上才存在。输出 Active 在相关功能块控制相应轴的运动时被设置。注意: 在非缓冲模式中, 输出 Busy 和 Active 始终具有相同的值。 •输出 Done:设置: 在任务成功完成后被设置。多个功能块先后作用于同一轴时: 当轴正在进行的运动被同一轴的另外一个运动中中断, 而目标状态(位置)还未达到时, 第一个功能块的输出 Done 不被设置。 •输出 CommandAborted: 在一个正在进行的运动任务被另外一个运动任务或 MC_Stop 中断时被设置。CommandAborted 的复位行为与 Done 相同。当输出 CommandAborted 被设置时, 其他诸如 InVelocity 等输出被复位。 •输出 Valid: 设置: Enable=TRUE 且输出值有效; 与 Error 在同一时刻只有一个为真。 •输出 Error: 设置: 执行功能块时发生了一个错误。 •输出 ErrorID: 功能块发生错误的代码, 详见第 10 章。
输入 Input	•输入 Execute: 参数始终随着输入 Execute 的上升沿被采用。要更改任一输入参数, 必须输入正确的值, 然后用输入 Execute 的上升沿触发采用。 •输入 Enable: 在周期性向一个输出发出(设定)一个控制参数值的功能块上存在。只要 Enable 设为 TRUE, 相关输出参数就能获得当前的值。 •位置或距离: “位置”是一个坐标系统中的值; “距离”是一个相对的大小, 表示两个位置之差。 •符号: 输入 Acceleration、Deceleration 和 Jerk 始终为正值; 位置和距离可正可负。

6.1.4 BUFFER_MODE

HIC 控制器运动功能块定义了 6 种 BufferMode 类型, 如下表 1.3 及图 1.3 所示。

表 1.3 BufferMode 类型

HT_MC_BUFFER_MODE	缓存模式	描述
0	Aborting	不带缓冲的标准模式。下一个功能块中断正在进行的运动, 即命令直接作用于轴。
1	Buffered	下一个功能块在前一个运动完成后才生效。轴被停止在两个运动之间。
2	BlendingLow	下一个功能块在前一个功能块完全执行后才操纵轴, 轴在第一个功能块执行结束时的速度为两个功能块中较低的速度。
3	BlendingPrevious	下一个功能块在前一个功能块完全执行后才操纵轴, 轴在第一个功能块执行结束时的速度为第一个功能块的速度。
4	BlendingNext	下一个功能块在前一个功能块完全执行后才操纵轴, 轴在第一个功能块执行结束时的速度为第二个功能块的速度。
5	BlendingHigh	下一个功能块在前一个功能块完全执行后才操纵轴, 轴在第一个功能块执行结束时的速度为第一个功能块的速度。

一个功能块执行结束时的速度为两个功能块中较高的速度。

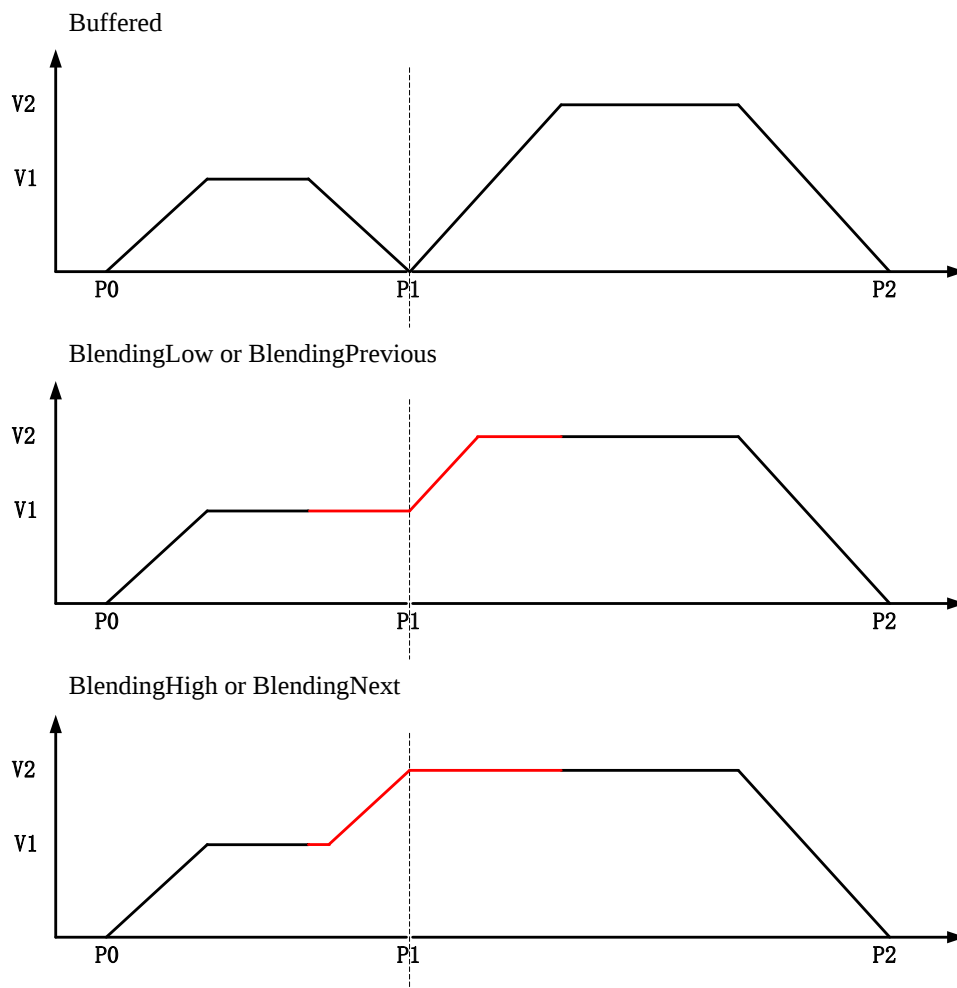


图 1.3 功能块运行变化图

6.1.5 MC_LIB 数据类型说明

MC_LIB 作为 HIC 控制器运动功能库，自带变量组 `AXIS_VAR`，其中共定义了四个全局变量，分别为 `AxisRef`、`AxisPara`、`AxisData`、`AxesGroupRef`，其中 `AxisRef` 为轴的状态，`AxisPara` 为轴的参数，定义在起始地址%MB3.22000 下(掉电保存)，`AxisData` 为轴的特殊输入输出数据，定义在起始地址%MB18000 下，`AxesGroupRef` 为轴组的状态，用户在使用时，直接用 MC_LIB 里面的全局变量即可。

- 变量名称为 `AxisPara`，对应结构体为 `HT_AXIS_CONFIG_ARRAY`
- 变量名称为 `AxisRef`，对应结构体为 `HT_AXIS_REF_ARRAY`
- 变量名称为 `AxisData`，对应结构体为 `HT_AXIS_DATA_ARRAY`
- 变量名称为 `AxesGroupRef`，对应结构体为 `HT_AXES_CONFIG_REF_ARRAY`

6.1.5.1 HT_AXIS_CONFIG_ARRAY 轴配置结构体

HT_AXIS_CONFIG_ARRAY 是轴参数的配置结构体，主要用于存储轴的配置参数。

(*运动类型结构体*)

```
HT_AXIS_TYPE : STRUCT
    ActiveFlag : BOOL;          (*激活标志 0: 不激活 1: 激活*)
    MotionType : BYTE;          (*轴类型 0: 线性轴 1: 非线性轴(预留) 2: 编码器轴*)
    VelocityType : BYTE;        (*规划模式 0: 直线规划 1: s 曲线规划*)
    ControllerMode : BYTE;      (*控制模式 0: 位置模式 1: 速度模式 2: 扭矩模式*)
    EncoderType : BYTE;         (*编码器类型 0: 增量编码器 1: 绝对编码器*)
END_STRUCT;
```

(*回零参数结构体*)

```
HT_HOME_DATA : STRUCT
    Home : LREAL;               (*原点位置*)
    HomeOffset : LREAL;         (*找到 Z 相脉冲位置*)
    HomeSearchVel : LREAL;      (*寻参速度*)
    HomeFinalVel : LREAL;       (*最终速度*)
    HomeLatchVel : LREAL;       (*寻 Z 相脉冲信号速度*)
    IndexUse : BOOL;            (*INDEX 信号是否使用 0: 不使用; 1: 使用*)
    IgnoreLimit : BOOL;         (*回零过程中限位开关忽略*)
END_STRUCT;
```

(*运动参数结构体*)

```
HT_MOTION_DATA : STRUCT
    IgnoreSoftwareLimit : BOOL; (*软件限位是否忽略 预留参数*)
    InputScale : DINT;          (*输入编码器线数 伺服旋转一圈反馈的脉冲数*)
    OutputScale : DINT;         (*输出编码器线数 伺服旋转一圈需要的脉冲数*)
    GearScale : LREAL;          (*电子齿轮比: 电机转一圈, 机械实际运行的距离*)
    BackLash : LREAL;           (*反向间隙*)
    ModuloValue : LREAL;        (*模数范围 预留参数*)
    MaxPos : LREAL;             (*正限位*)
    MinPos : LREAL;             (*负限位*)
    MaxVel : LREAL;             (*最大速度*)
    MaxAcc : LREAL;             (*最大加速度*)
    MaxDec : LREAL;             (*最大减速度*)
    MaxJerk : LREAL;            (*最大加加速度*)
    MinError : LREAL;           (*静态跟随误差*)
    MaxError : LREAL;           (*动态跟随误差*)
    MotorOffset : LREAL;        (*电机偏移*)
END_STRUCT;
```

END_STRUCT;

(*PID 参数结构体*)

HT_PID_DATA : STRUCT

PidPgain : REAL;	(*PID 比例增益*)
PidIgain : REAL;	(*PID 积分增益*)
PidDgain : REAL;	(*PID 微分增益*)
PidMaxOut : REAL;	(*PID 最大输出值*)
PidBias : REAL;	(*PID 稳态偏移*)
PidDeadBand : REAL;	(*PID 死区*)
PidKgain : REAL;	(*前馈增益 预留参数*)
PidEnable : BOOL;	(*PID 使用使能*)
PidErrorSource : BOOL;	(*PID 偏差来源 0: 控制器自动计算; 1: 偏差外部来源*)
PidType : BYTE;	(*PID 类型 0: 位置绝对; 1: 位置增量; 2: 增量+前馈(预留)*)

END_STRUCT;

(*预留参数结构体*)

HT_USER_DEFINE_DATA : STRUCT

UserDefine : ARRAY [0..99] OF BYTE;

END_STRUCT;

(*单轴配置结构体*)

HT_AXIS_CONFIG : STRUCT

AxisType : HT_AXIS_TYPE;	(*运动类型选择*)
MotionData : HT_MOTION_DATA;	(*运动参数*)
PidData : HT_PID_DATA;	(*PID 参数*)
HomeData : HT_HOME_DATA;	(*回零相关参数*)
UserDefine : HT_USER_DEFINE_DATA;	(*预留*)

END_STRUCT;

(*所有轴配置数组*)

HT_AXIS_CONFIG_ARRAY : STRUCT

AxisConfigArray : ARRAY [0..7] OF HT_AXIS_CONFIG;

END_STRUCT;

6.1.5.2 HT_AXIS_REF_ARRAY 轴状态更新结构体

HT_AXIS_REF_ARRAY 是轴状态更新结构体，主要用于存储轴的当前状态。

(*运动状态结构体*)

HT_AXIS_MOTION_STATE : STRUCT

Enabled : BOOL;	(*轴使能标志*)
-----------------	-----------

```

ServoEnabled : BOOL;           (*伺服使能标志位*)
Activated : BOOL;              (*轴激活标志*)
Homed : BOOL;                 (*轴已回零标志*)
AxisConstantVel : BOOL;       (*当前运动在恒速中*)
AxisAccelerating : BOOL;      (*当前运动在加速中*)
AxisDecelerating : BOOL;      (*当前运动在减速中*)
MotionPosDir : BOOL;          (*轴正向运动中*)
MotionNegDir : BOOL;          (*轴负向运动中*)
VirtualAxis : BOOL;           (*是否虚轴*)
AxisInGroupFlag : BOOL;       (*是否加入轴组中*)
CommunicationState : BYTE;     (*总线通讯状态 0: 通讯初始化; 1: 通讯准备好; 2: 通讯正常 3: 通讯错误*)

AxisNo : UINT;                (*运动轴号*)
AxisState : HT_AXIS_STATE;     (*轴状态: 0: disabled; 1: standstill; 2: ErrorStop; 3: Stopping; 4: Homing; 5: Discrete Mtion; 6: continous motion 7: Synchronized Motion*)

AxisErrorID : UINT;           (*运动错误 ID*)
CamState : UINT;              (*电子凸轮状态 0: 电子凸轮未执行 1: 电子凸轮同步过程中 2: 电子凸轮执行中 3: 电子凸轮表不是周期模式, 凸轮表一个周期执行完成*)

GearState : UINT;             (*电子齿轮状态 0: 电子齿轮未执行 1: 电子齿轮同步过程中 2: 电子齿轮执行中*)

PosFollowState:UINT;          (*位置跟随状态*)
SetTorQue : INT;              (*设定扭矩*)
ActTorQue : INT;              (*实际扭矩*)
MotionID : DINT;              (*运动 ID*)
SetPosition : REAL;           (*当前命令位置*)
ActPosition : REAL;           (*当前反馈位置*)
SetVelocity : REAL;           (*当前命令速度*)
ActVelocity : REAL;           (*当前反馈速度*)
SetAccleration : REAL;        (*当前命令加速度*)
ActAccleration : REAL;        (*当前反馈加速度 预留参数*)
SetJerk : REAL;               (*当前命令加加速度*)
ActJerk : REAL;               (*当前反馈加加速度 预留参数*)
Override : REAL;              (*当前速度修调值*)
CycleTime : REAL;             (*运动控制周期*)
SetPosition_LREAL : LREAL;    (*当前命令位置*)
ActPosition_LREAL : LREAL;    (*当前反馈位置*)
END_STRUCT;

(*单轴状态结构体*)
HT_AXIS_REF : STRUCT
    AxisMotionState : HT_AXIS_MOTION_STATE; (*运动状态*)
    AxisPara : HT_AXIS_CONFIG;              (*实际参数*)

```

```
END_STRUCT;
```

(*所有轴状态数组*)

```
HT_AXIS_REF_ARRAY : STRUCT
    AxisRefArray : ARRAY [0..7] OF HT_AXIS_REF;
END_STRUCT;
```

6.1.5.3 HT_AXIS_DATA_ARRAY 轴特殊结构体

HT_AXIS_DATA_ARRAY 是轴的特殊输入输出数据结构体。

(*轴特殊输入结构体*)

```
HT_AXIS_DATA_IN : STRUCT
    Home : BOOL; (*轴原点信号*)
    PosLimit : BOOL; (*轴正限位信号*)
    NegLimit : BOOL; (*轴负限位信号*)
    ServoVelUnit : BYTE; (*伺服反馈速度的单位 0: 0.1r/min; 1: pluse/s *)
    StatusWord : UINT; (*状态字*)
    Position : DINT; (*位置*)
    Velocity : DINT; (*速度*)
    Current : UINT; (*扭矩*)
    ZIndex : UINT; (*轴 Z 相信号*)
    PIDError : REAL; (*PID 偏差从外部输入的值,当轴参数配置为 PID
    数据从外部来时才有效*)
    PIDFeedForward : REAL; (*PID 前馈从外部输入的值,当轴参数配置为 PID
    数据从外部来时才有效*)
    EncoderAxisPos : LREAL; (*编码器轴外部输入的位置值, 当轴参数配置为
    编码器轴才有效*)
    EncoderAxisVel : LREAL; (*编码器轴外部输入的速度值, 当轴参数配置为
    编码器轴才有效*)
    Var1 : INT; (*预留*)
    Var2 : INT; (*预留*)
END_STRUCT;
```

(*轴特殊输出结构体*)

```
HT_AXIS_DATA_OUT : STRUCT
    Control : UINT; (*控制字*)
    Position : DINT; (*位置*)
    Velocity : DINT; (*速度*)
    Torque : UINT; (*扭矩*)
    Mode : BYTE; (*伺服控制模式*)
    TouchProbeFun : UINT;
    Var1 : INT; (*预留*)
```

```

        Var2 : INT;                                (*预留*)
END_STRUCT;

```

(*单轴特殊数据结构体*)

```

HT_AXIS_DATA : STRUCT
    VirtualEnable : BOOL;                          (*虚轴使能 0: 不使能 1: 使能*)
    SingleServoMode : BOOL;                        (*伺服单圈模式 0: 不使能 1: 使能*)
    AxisDataIn : HT_AXIS_DATA_IN;                 (*轴数据输入*)
    AxisDataOut : HT_AXIS_DATA_OUT;                (*轴数据输出*)
END_STRUCT;

```

(*所有轴特殊数据数组*)

```

HT_AXIS_DATA_ARRAY : STRUCT
    EmergencyStop : BYTE;                          (*急停输入,预留*)
    BusStatus : BYTE;                              (*总线状态*)
    AxisDataArray : ARRAY [0..7] OF HT_AXIS_DATA;
END_STRUCT;

```

6.1.5.4 HT_AXES_GROUP_REF_ARRAY 轴组状态结构体

HT_AXES_GROUP_REF_ARRAY 是轴组状态结构体，主要用于存储轴组的当前状态。

(*轴组中单轴状态结构体*)

```

AXIS_GROUP_STATE : STRUCT
    active : BYTE;                                (*轴组中轴是否激活*)
    drive_num : DINT;                             (*轴组中轴对应的实际轴*)
    home_sequence : DINT;                         (*轴组中轴回零顺序*)
END_STRUCT;

```

(*轴组结构体*)

```

HT_AXES_GROUP_REF : STRUCT
    GroupNo : BYTE;                               (*轴组组号*)
    GroupEnable : BOOL;                           (*轴组是否使能标志，对应轴组状态图*)
    GroupActive : BOOL;                           (*轴组 motion 是否使能，对应轴组控制器是否使能*)
    GroupStatus : HT_GROUP_STATUS;                (*轴组状态机： 0: GroupDisabled; 1: GroupStandby; 2: GroupErrorStop; 3: GroupStopping; 4: GroupHoming; 5: GroupMoving*)
    GroupActiveAxesNum : DINT;                    (*轴组中加入的轴数*)
    GroupErrorID : DINT;                          (*轴组错误 ID*)
    GroupSetVel : REAL;                           (*轴组速度*)

```

```

        GroupSetAcc : REAL;                (*轴组加速度*)
        GroupCurrentPos : ARRAY [0..7] OF REAL;    (*轴组加速度*)
        GroupGoalPos : ARRAY [0..7] OF REAL;    (*轴组目标位置*)
        GroupAxisState : ARRAY [0..7] OF HT_AXIS_GROUP_STATE; (*轴组状态*)
    END_STRUCT;

    (*所有轴组数组*)
    HT_AXES_GROUP_REF_ARRAY : STRUCT
        AxesGroupRefArray : ARRAY [0..0] OF HT_AXES_GROUP_REF;
    END_STRUCT;

    (*轴组插补位置结构体*)
    HT_GROUP_AXES_POS : STRUCT
        Position : ARRAY [0..7] OF REAL;
    END_STRUCT;

    (*轴组中标志结构体*)
    HT_IDENT_IN_GROUP_REF : STRUCT
        AxisNoInGroup : DINT;                (*轴在轴组中的轴号*)
        AxisHomeSequence : DINT;            (*回零顺序*)
    END_STRUCT;

    (*轴组过渡模式预留参数数组*)
    HT_MC_TRANSITION_PARA: STRUCT
        TransitionParameter : ARRAY [0..7] OF REAL;    (*目前仅用到第 1 个 REAL 类型,
                                                         代表轴组过渡打开距离*)
    END_STRUCT;

```

6.1.5.5 凸轮表结构体

(*凸轮 ID, 目前支持两个凸轮表*)

```
TYPE HT_CAM_ID :
```

```

(
    CamID0 := 0,
    CamID1 := 1
);
END_TYPE

```

(*凸轮启动模式*)

```
TYPE HT_CAM_START_MODE :
```

```

(
    McAbsolute := 0, (*绝对运动*)
    McRelative := 1  (*相对运动*)
);

```

END_TYPE

(*主轴位置数据来源*)

TYPE HT_MC_SOURCE :

(

McSetValue := 0, (*命令值*)

McActualValue := 1 (*反馈值*)

);

END_TYPE

(*凸轮表点结构体*)

HT_CAM_TABLE_POINT : STRUCT

X : REAL; (*主轴位置*)

Y : REAL; (*从轴位置*)

V : REAL; (*速度*)

A : REAL; (*加速度*)

J : REAL; (*加加速度*)

SegmentType : HT_SEGMENT_TYPE; (*凸轮表曲线类型: 0: Poly5, 目前支持五次曲线*)

END_STRUCT

(*凸轮表结构体*)

HT_MC_CAM_REF : STRUCT

CamPointNum : UINT; (*凸轮表点数*)

CamID : HT_CAM_ID; (*凸轮表 ID*)

CamPointData : ARRAY [0..49] OF HT_CAM_TABLE_POINT; (*凸轮表点数据*)

END_STRUCT

(*凸轮表 ID 结构体*)

HT_MC_CAM_ID : STRUCT

Periodic : BOOL; (*TRUE: 周期性执行; FALSE: 执行一个周期*)

MasterAbsolute : BOOL; (*TRUE: 主轴绝对模式; FALSE: 主轴相对模式*)

SlaveAbsolute : BOOL; (*该参数和 MC_CamIn.StartMode 共同决定从轴模式。
(MC_CamTableSelect. SlaveAbsolute = TRUE) AND (MC_CamIn.StartMode = 0): 从轴绝对模式; 其它情况下从轴为相对模式*)

MasterNum : UINT; (*主轴号*)

SlaveNum : UINT; (*从轴号*)

CamID : HT_CAM_ID; (*凸轮表 ID*)

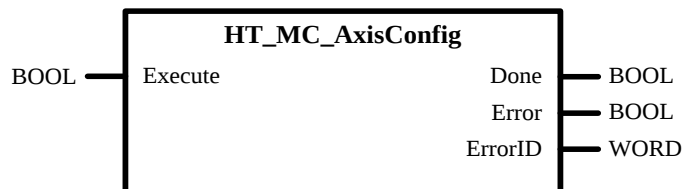
CheckValue : UINT; (*校验值*)

END_STRUCT

6.1.6 系统功能块

6.1.6.1 HT_MC_AxisConfig 轴配置功能块

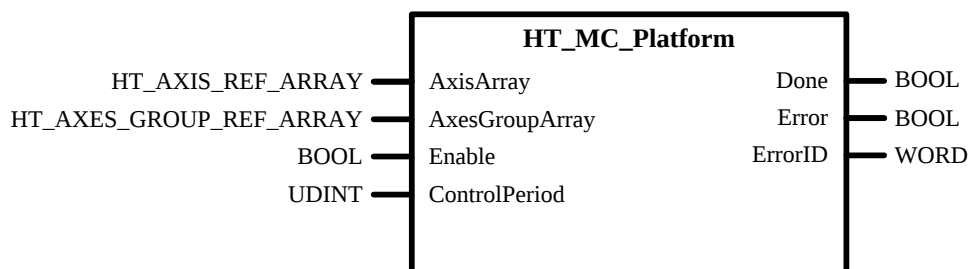
描述：HT_MC_AxisConfig 功能块对轴进行相关的参数配置，使用该功能块时轴必须处于 Disabled 或者 ErrorStop 状态。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.6.2 HT_MC_Platform 运动状态更新功能块

HT_MC_Platform 功能块对所有轴状态、所有轴组状态进行更新，如果要使用运动控制功能，该功能块必须放在所有运动功能块之前。



参数	数据类型	描述
VAR_IN_OUT		
AxisArray	HT_AXIS_REF_ARRAY	轴状态
AxesGroupArray	HT_AXES_GROUP_REF_ARRAY	轴组状态
VAR_INPUT		
Enable	BOOL	如果 Enable=TRUE，表示执行该功能块

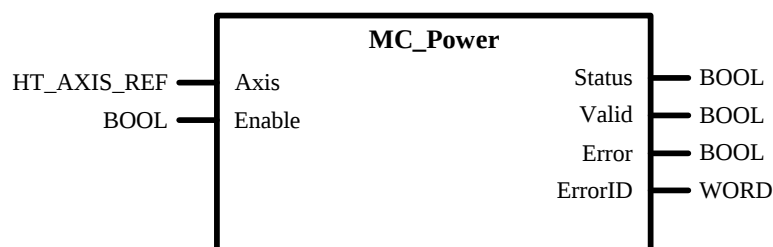
ControlPeriod	UDINT	运动控制周期(同调用该功能块的 PLC 任务周期, 单位 us)
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.7 单轴运动功能块

在本节中主要介绍 HIC 控制器 MC_LIB 中的单轴运动功能块, 包括 PLCopen Part1/2 部分。接下来将对每一个功能块进行介绍

6.1.7.1 MC_Power

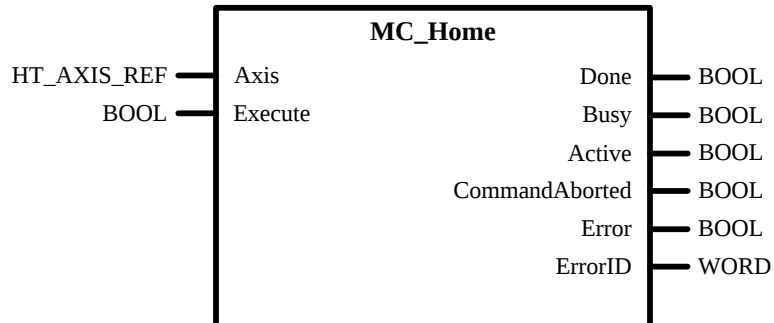
该功能块对轴进行使能操作, 当轴为虚轴时, 该功能块使能够让虚轴上使能; 当轴为实轴时, 将轴的控制字输出到伺服上时, 该功能块能够伺服使能, 轴上使能后, 状态由 Disable 切换到 Standstill。



参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Enable	BOOL	如果 Enable = TRUE, 表示轴上使能 如果 Enable = FALSE, 表示轴下使能
VAR_OUTPUT		
Status	BOOL	TRUE 表示轴上使能, FALSE 表示轴未使能
Valid	BOOL	预留参数
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.7.2 MC_Home

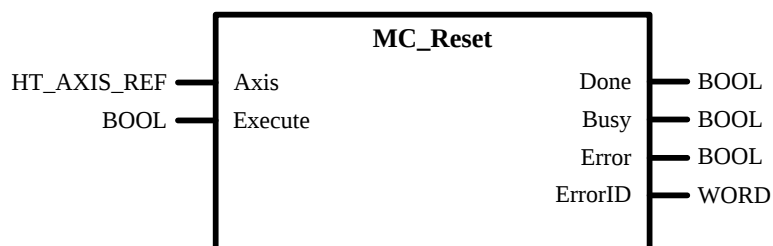
该功能块为轴的回零，在触发该功能块后，轴设定的回零方式进行回零。



参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.7.3 MC_Reset

该功能块为轴错误复位功能块，轴状态从 ErrorStop 转为 Standstill 或者 Disabled。

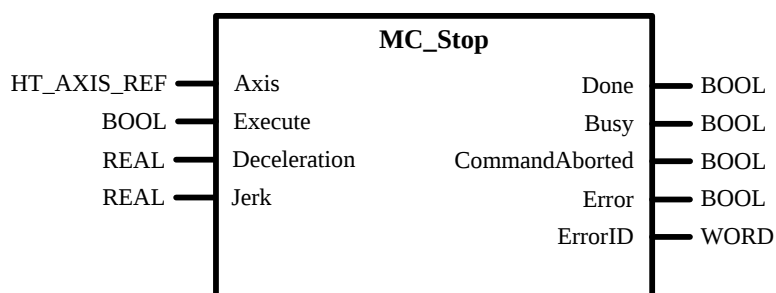


参数	数据类型	描述
----	------	----

VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.7.4 MC_Stop

该功能块使轴的运动停止，并将轴状态设置为 **Stopping**；处于 **Stopping** 状态的轴，其他功能块不能执行任何运动；当 **Execute** 为 **True** 时，轴一直处于 **Stopping** 状态；**Done** 为 **True** 且 **Execute** 为 **False** 时，状态变为 **Standstill**

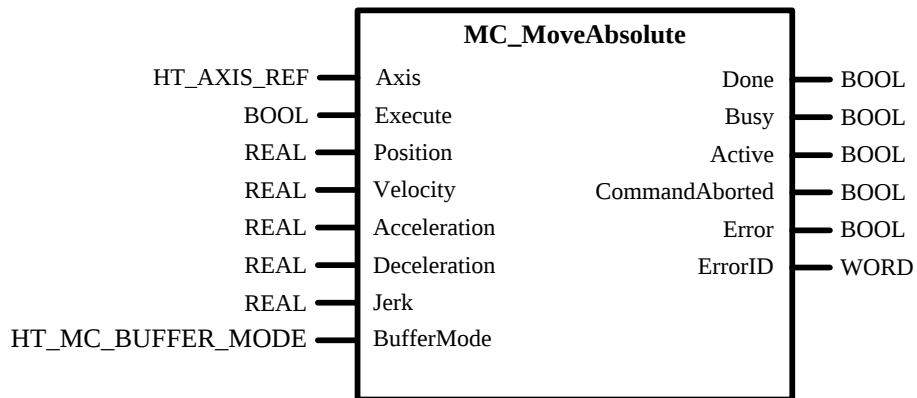


参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Execute	BOOL	上升沿触发
Deceleration	REAL	停止减速度
Jerk	REAL	预留参数
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误

ErrorID	WORD	错误代码
---------	------	------

6.1.7.5 MC_MoveAbsolute

该功能块为轴的绝对定位运动，在触发该功能块后，轴能够以设定的速度、加速度、加加速度运动到给定的 position 位置值。当 BufferMode 给定不同的值时，两个运动段之间有不同的过渡连接。

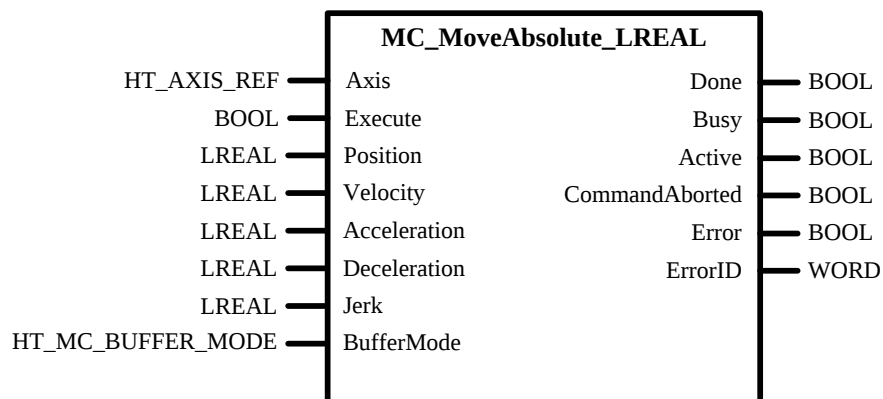


参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Execute	BOOL	上升沿触发
Position	REAL	运动的绝对目标位置，单位[u]（负或正）
Velocity	REAL	额定速度，单位[u]/[s]（始终为正）
Acceleration	REAL	额定加速度，单位[u]/[s ²]（始终为正）
Deceleration	REAL	额定减速度，单位[u]/[s ²]（始终为正）
Jerk	REAL	额定加/减速度斜率，单位[u]/[s ³] (始终为正，只有当规划为 S 曲线时该参数有效)
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了

Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.7.6 MC_MoveAbsolute_LREAL

该功能块为轴的高精度绝对定位运动，当 REAL 类型参数满足不了运动精度时，选择该功能块进行轴的绝对定位运动。

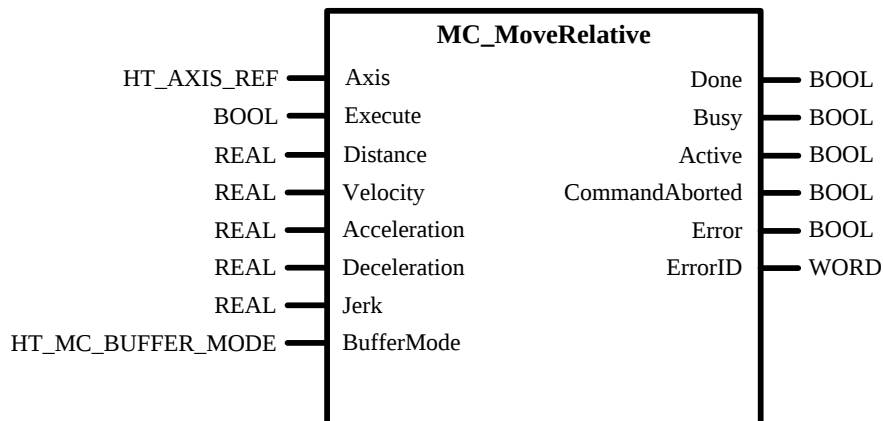


参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Execute	BOOL	上升沿触发
Position	LREAL	运动的绝对目标位置，单位[u]（负或正）
Velocity	LREAL	额定速度，单位[u]/[s]（始终为正）
Acceleration	LREAL	额定加速度，单位[u]/[s2]（始终为正）
Deceleration	LREAL	额定减速度，单位[u]/[s2]（始终为正）
Jerk	LREAL	额定加/减速度斜率，单位[u]/[s3] (始终为正，只有当规划为 S 曲线时该参数有效)
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了

Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.7.7 MC_MoveRelative

该功能块为轴的相对定位运动，运动距离通过 Distance 进行设置，当执行到该功能时，目标位置=当前位置+运动距离。

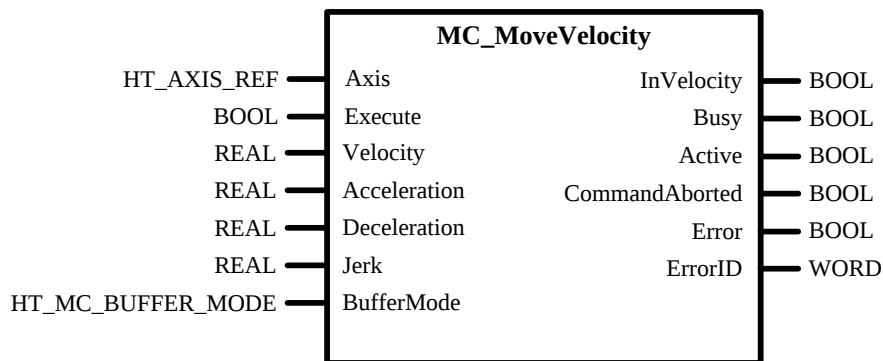


参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Execute	BOOL	上升沿触发
Distance	REAL	运动的相对距离，单位[u]（负或正）
Velocity	REAL	额定速度，单位[u]/[s]（始终为正）
Acceleration	REAL	额定加速度，单位[u]/[s ²]（始终为正）
Deceleration	REAL	额定减速度，单位[u]/[s ²]（始终为正）
Jerk	REAL	额定加/减速度斜率，单位[u]/[s ³] (始终为正，只有当规划为 S 曲线时该参数有效)
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误

ErrorID	WORD	错误代码
---------	------	------

6.1.7.8 MC_MoveVelocity

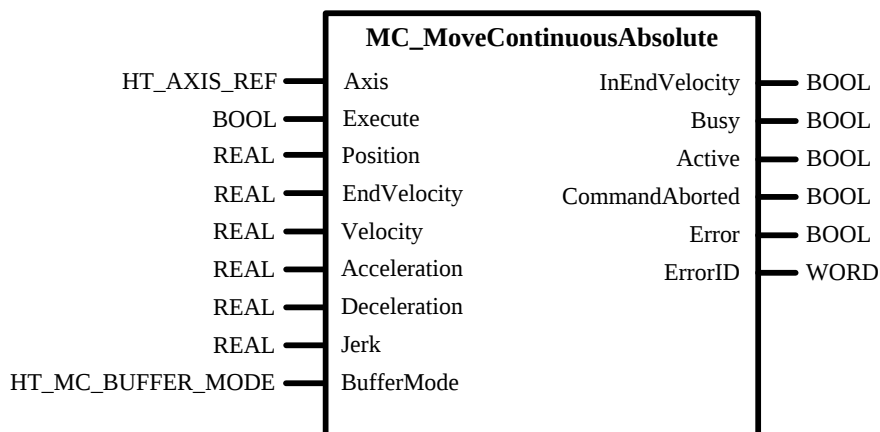
该功能块实现轴以设定的速度进行不停止的运动。该功能块不能自己停止，只能被其它功能块中断



参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Execute	BOOL	上升沿触发
Velocity	REAL	额定速度，单位[u]/[s]（始终为正）
Acceleration	REAL	额定加速度，单位[u]/[s ²]（始终为正）
Deceleration	REAL	额定减速度，单位[u]/[s ²]（始终为正）
Jerk	REAL	额定加/减速度斜率，单位[u]/[s ³] (始终为正，只有当规划为 S 曲线时该参数有效)
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为
VAR_OUTPUT		
InVelocity	BOOL	TRUE 表示达到设定速度
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.7.9 MC_MoveContinuousAbsolute

该功能块实现轴的绝对定位运动，到达目标位置时速度不为 0，为指定的速度 EndVelocity，且以 EndVelocity 速度继续运行。该功能块不能自己停止，只能被其它功能块中断。



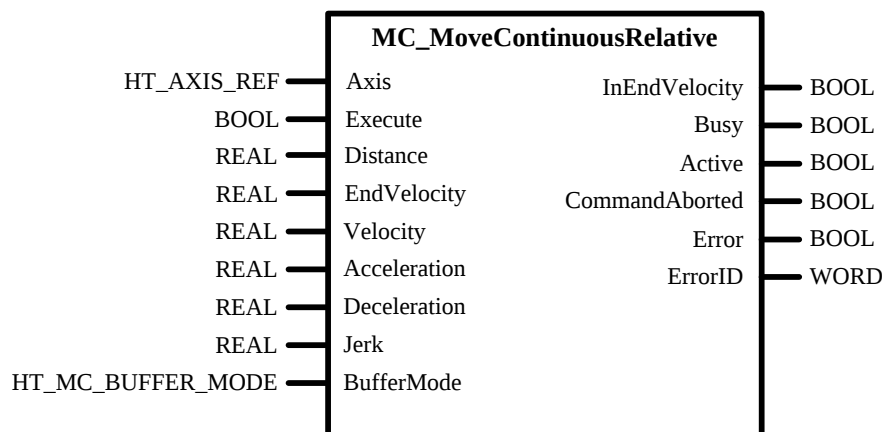
参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	参考轴
VAR_INPUT		
Execute	BOOL	上升沿触发
Position	REAL	运动的绝对目标位置，单位[u]（负或正）
EndVelocity	REAL	结束速度，单位[u]/[s]（负或正）
Velocity	REAL	额定速度，单位[u]/[s]（始终为正）
Acceleration	REAL	额定加速度，单位[u]/[s ²]（始终为正）
Deceleration	REAL	额定减速度，单位[u]/[s ²]（始终为正）
Jerk	REAL	额定加/减速度斜率，单位[u]/[s ³] (始终为正，只有当规划为 S 曲线时该参数有效)
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为（BufferMode 值只能是 0，如果当前正在执行该功能块，新的命令将打断功能块）
VAR_OUTPUT		
InEndVelocity	BOOL	TRUE 表示到达目标位置值，且以 EndVelocity 继续运动
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意：

- （1）功能块执行允许的轴状态：StandStill、DiscreteMotion、ContinuousMotion、SynchronizedMotion；
- （2）功能块执行时，轴状态：ContinuousMotion；
- （3）直线规划时：Jerk 参数无效；
- （4）S 曲线规划时：控制器默认将 Deceleration 参数赋值为 Acceleration 值，即加减速度值相同；
- （5）功能块执行时位置仍受到软限位限制；
- （6）BufferMode 值只能是 0，如果当前正在执行该功能块，新的命令将打断功能块。

6.1.7.10 MC_MoveContinuousRelative

描述：该功能块实现轴相对于实际位置的相对运动，运动距离通过 Distance 进行设置，当执行到该功能时，目标位置=当前位置+运动距离。到达目标位置时速度不为 0，为指定的速度 EndVelocity，且以 EndVelocity 速度继续运行。该功能块不能自己停止，只能被其它功能块中断。



参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	参考轴
VAR_INPUT		
Execute	BOOL	上升沿触发
Distance	REAL	运动的距离，单位[u]（负或正）
EndVelocity	REAL	结束速度，单位[u]/[s]（负或正）
Velocity	REAL	额定速度，单位[u]/[s]（始终为正）
Acceleration	REAL	额定加速度，单位[u]/[s2]（始终为正）
Deceleration	REAL	额定减速度，单位[u]/[s2]（始终为正）
Jerk	REAL	额定加/减速度斜率，单位[u]/[s3] (始终为正，只有当规

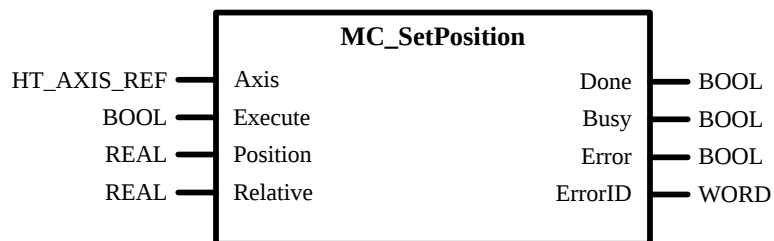
		划为 S 曲线时该参数有效)
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为（BufferMode 值只能是 0，如果当前正在执行该功能块，新的命令将打断功能块）
VAR_OUTPUT		
InEndVelocity	BOOL	TRUE 表示到达目标位置值，且以 EndVelocity 继续运动
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意：

- （1）功能块执行允许的轴状态：StandStill、DiscreteMotion、ContinuousMotion、SynchronizedMotion；
- （2）功能块执行时，轴状态：ContinuousMotion；
- （3）直线规划时：Jerk 参数无效；
- （4）S 曲线规划时：控制器默认将 Deceleration 参数赋值为 Acceleration 值，即加减速度值相同；
- （5）功能块执行时位置仍受到软限位限制；
- （6）BufferMode 值只能是 0，如果当前正在执行该功能块，新的命令将打断功能块。

6.1.7.11 MC_SetPosition

该功能块为轴的坐标偏移功能块，当该功能块的 Execute 为 TRUE 时，轴的当前位置设置为指定的值。



参数	数据类型	描述
VAR_IN_OUT		
Axis	HT_AXIS_REF	轴
VAR_INPUT		
Execute	BOOL	上升沿触发

Position	REAL	设置位置值(当为绝对模式时，轴的位置值设定为 position 值；当为相对模式时，轴的位置值设定为当前位置+position 值)
Relative	BOOL	FALSE:绝对模式；TRUE:相对模式
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

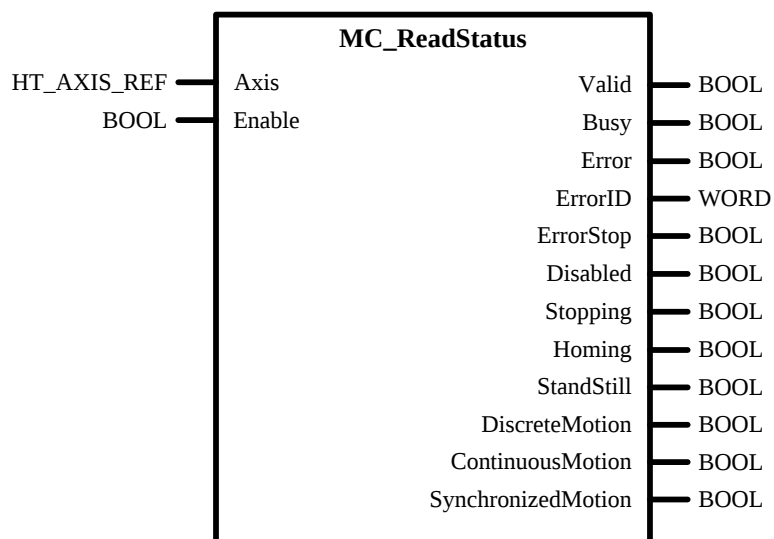
注意：

（1）该功能允许的轴状态：直线规划时：Standstill、DiscreteMotion、ContinuousMotion；S 曲线规划时：Standstill。

（2）当 Relative=FALSE 时，功能块将当前位置设为 Position 值；当 Relative=TRUE 时，功能块将当前位置设为（当前位置+Position）值；

6.1.7.12 MC_ReadStatus

描述：该功能块用于读取轴的详细状态。



参数	数据类型	描述
VAR_IN_OUT		
Axis	<u>HT_AXIS_REF</u>	参考轴
VAR_INPUT		
Enable	BOOL	只要 Enable = TRUE，当前运动状态将被持续输出
VAR_OUTPUT		

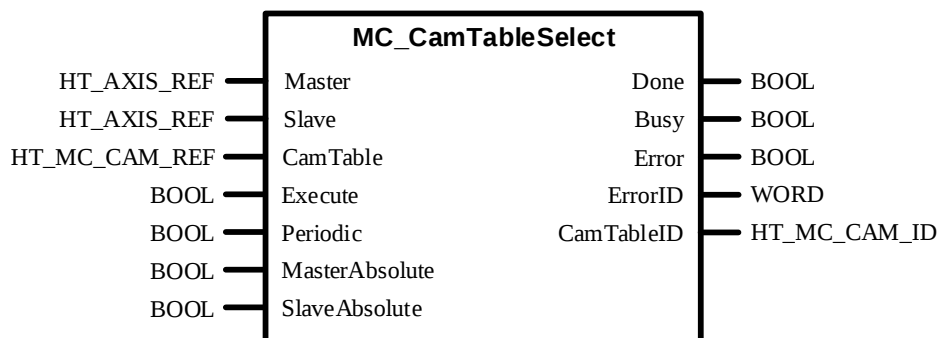
Valid	BOOL	TRUE 表示当前运动状态已输出
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码
Errorstop	BOOL	TRUE 表示轴处于“故障停止”状态
Disable	BOOL	TRUE 表示轴处于“未使能”状态
Stopping	BOOL	TRUE 表示轴处于“停止”状态
Homing	BOOL	TRUE 表示轴处于“回零”状态
StandStill	BOOL	TRUE 表示轴处于“静止”状态
DiscreteMotion	BOOL	TRUE 表示轴处于“离散运动”状态
ContinuousMotion	BOOL	TRUE 表示轴处于“连续运动”状态
SynchronizedMotion	BOOL	TRUE 表示轴处于“同步运动”状态

6.1.8 轴组(主/从轴)运动功能块

在本节中主要介绍 HIC 控制器 MC_LIB 中的单轴运动功能块，包括 PLCopen Part4 部分。接下来将对每一个功能块进行介绍。

6.1.8.1 MC_CamTableSelect

描述：该功能块将选择一个凸轮表文件连接到凸轮主从轴，使建立主轴、从轴、凸轮表三者的关系，并设定凸轮运行的周期、主轴从轴的位置模式（绝对或相对）。

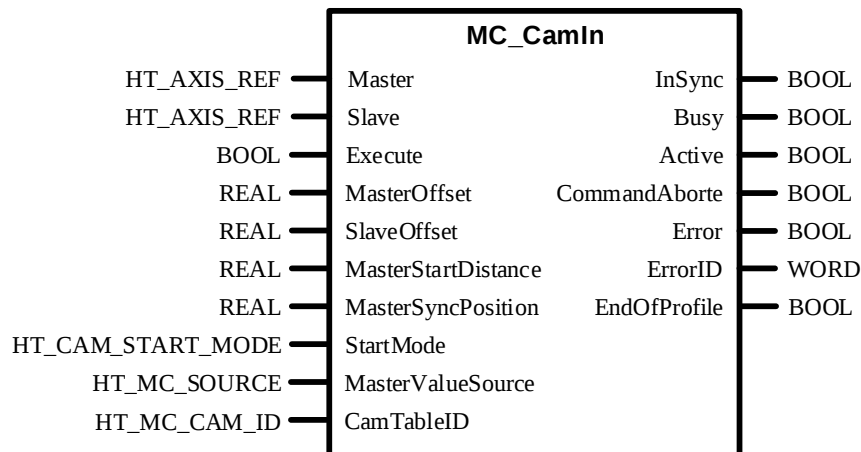


参数	数据类型	描述
VAR_IN_OUT		
Master	HT_AXIS_REF	主动轴
Slave	HT_AXIS_REF	从动轴
CamTable	HT_MC_CAM_REF	要选择打开的凸轮表

VAR_INPUT		
Execute	BOOL	上升沿触发
Periodic	BOOL	TRUE: 周期性执行; FALSE: 执行一个周期
MasterAbsolute	BOOL	TRUE: 主轴绝对模式; FALSE: 主轴相对模式
SlaveAbsolute	BOOL	该参数和 MC_CamIn.StartMode 共同决定从轴模式。 (MC_CamTableSelect. SlaveAbsolute = TRUE) AND (MC_CamIn.StartMode = 0): 从轴绝对模式; 其它情况下从轴为相对模式
VAR_OUTPUT		
Done	BOOL	凸轮表选择完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码
CamTableID	HT_MC_CAM_ID	凸轮表信息

6.1.8.2 MC_CamIn

描述: 该功能块使凸轮从轴根据凸轮主轴位置及凸轮表中的位置、速度、加速度关系进行同步运行, 轴状态切换到 Synchronized Motion, 该指令的执行对主轴运动没有任何影响。

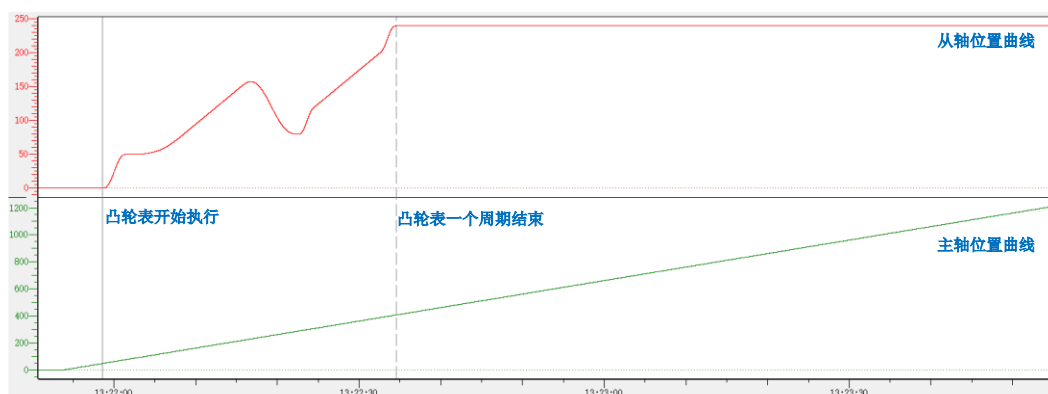


参数	数据类型	描述
VAR_IN_OUT		
Master	HT_AXIS_REF	主动轴

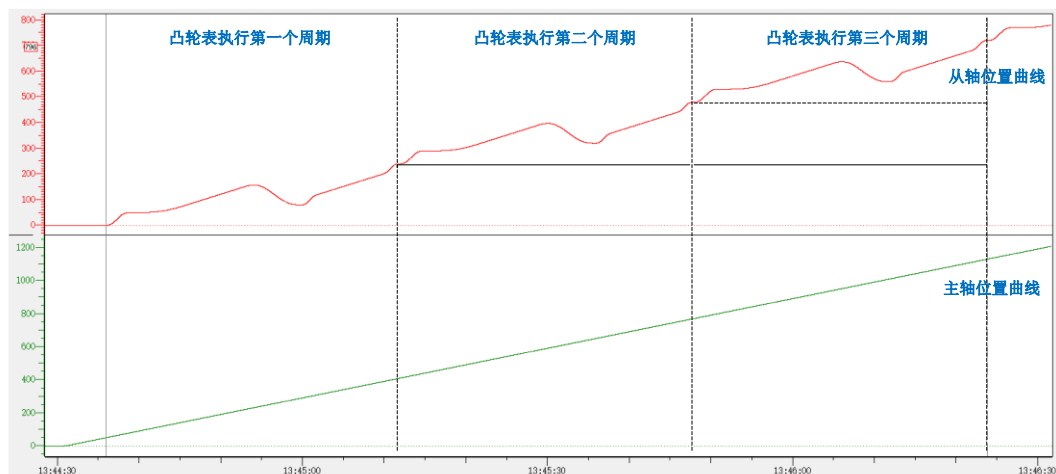
Slave	<u>HT_AXIS_REF</u>	从动轴
VAR_INPUT		
Execute	BOOL	上升沿触发
MasterOffset	REAL	主轴相对凸轮表偏移
SlaveOffset	REAL	从轴相对凸轮表偏移
MasterStartDistance	REAL	主从轴同步过程主轴的运动距离
MasterSyncPosition	REAL	主从轴同步结束时的主轴位置值
StartMode	<u>HT_CAM_START_MODE</u>	该参数和 MC_CamTableSelect. SlaveAbsolute 共同决定从轴模式。 (MC_CamTableSelect. SlaveAbsolute = TRUE) AND (MC_CamIn.StartMode = 0): 从轴绝对模式; 其它情况下从轴为相对模式
MasterValueSource	<u>HT_MC_SOURCE</u>	主轴位置来源, 0: 命令值; 1: 反馈值
CamTableID	<u>HT_MC_CAM_ID</u>	要执行凸轮表 ID (该参数为 MC_CamTableSelect 功能块输出的 CamTableID, 作为本功能块的输入)
VAR_OUTPUT		
InSync	BOOL	TRUE 表示主从轴位置同步
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码
EndOfProfile	BOOL	凸轮表执行结束, 输出一个脉冲信号

(1) 凸轮表的周期模式 (Periodic)

当 MC_CamTableSelect.Periodic = false 时, 为单周期凸轮运行状态, 即在运行完一个凸轮表周期后, 从轴脱离凸轮运行状态, 保持在当前位置, 如下图所示, 且输出 MC_CamIn.EndOfProfile = true。

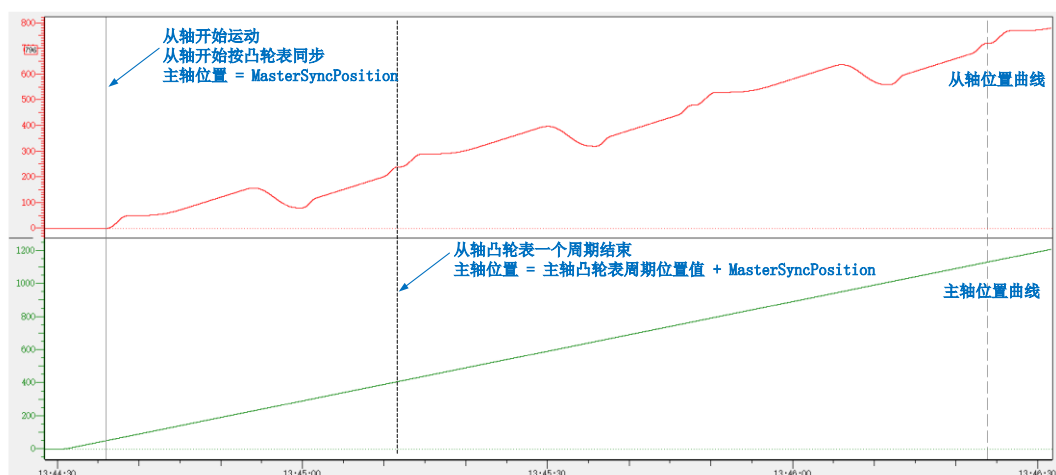


当 MC_CamTableSelect.Periodic = true 时，为周期性凸轮运行状态，即在运行完一个凸轮表周期后，从轴继续执行下一凸轮周期的运行，直到有命令中断使其退出凸轮运行状态，如下图所示。

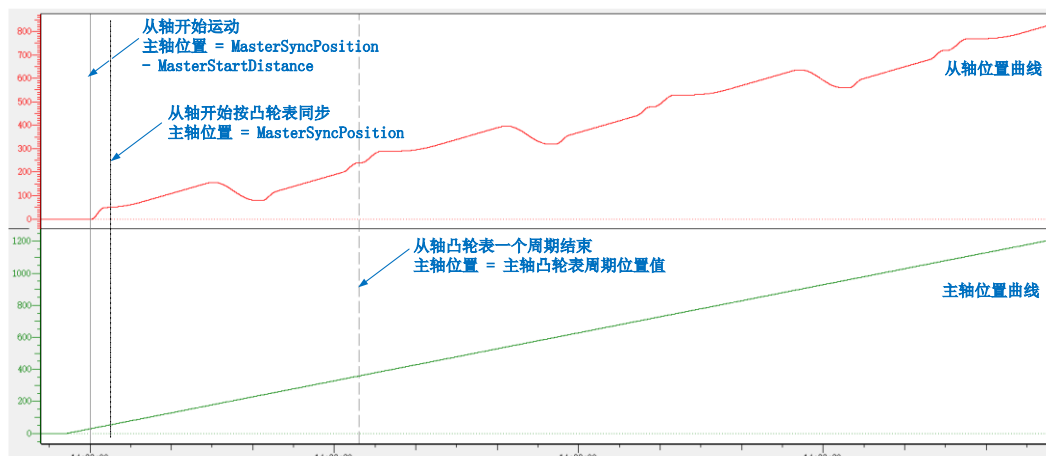


(2) 主从轴各运动模式时的运行状态 (MasterAbsolute\SlaveAbsolute\StartMode)

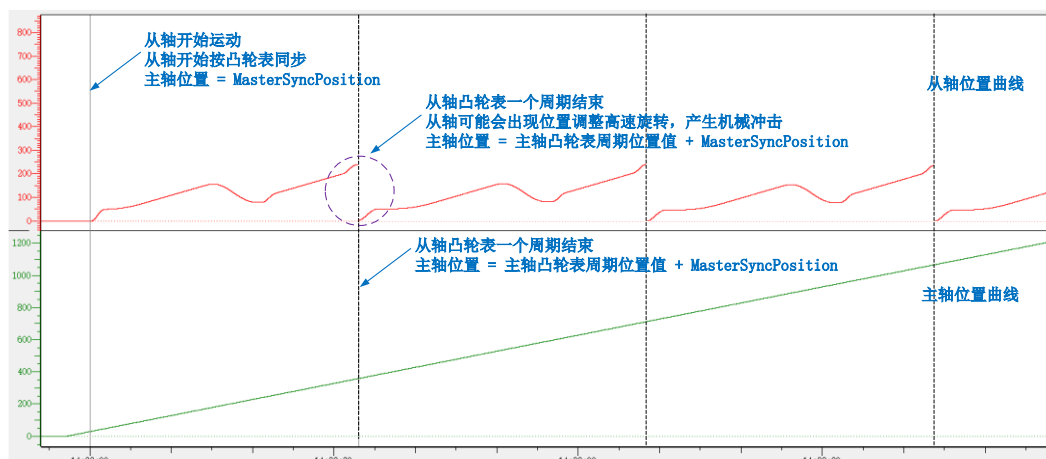
A、主从轴都为相对模式的运行状态 (MC_CamTableSelect.MasterAbsolute = false, MC_CamTableSelect.SlaveAbsolute = false 或 MC_CamIn.StartMode = 1)，如下图所示。



B、主轴为绝对模式，从轴为相对模式的运行状态 (MC_CamTableSelect.MasterAbsolute = true, MC_CamTableSelect.SlaveAbsolute = false 或 MC_CamIn.StartMode = 1)，如下图所示。

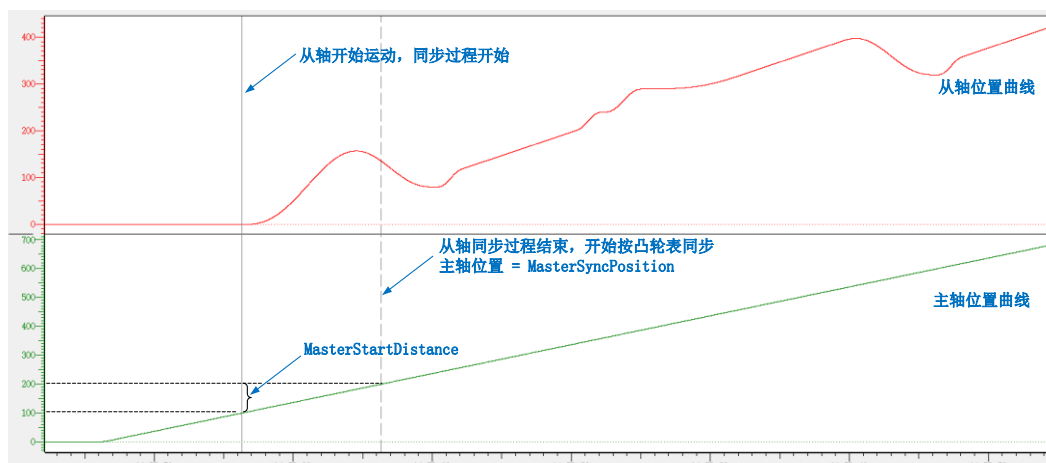


C、主轴为相对模式，从轴为绝对模式的运行状态（MC_CamTableSelect.MasterAbsolute = true, MC_CamTableSelect.SlaveAbsolute = true AND MC_CamIn.StartMode = 0），如下图所示。



（3）主从轴同步过程（MasterStartDistance\MasterSyncPosition）

MasterSyncPosition 给定主从轴同步过程结束时主轴的位置值，MasterStartDistance 给定主从轴同步过程的距离（电子凸轮开始同步位置=MasterSyncPosition - MasterStartDistance），当主轴到达 MasterSyncPosition 位置点时，同步过程结束，开始执行凸轮表。

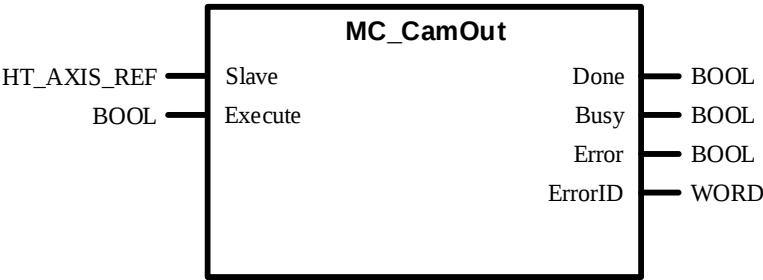


(4) 凸轮运行中 Offset 功能 (MasterOffset\ SlaveOffset)

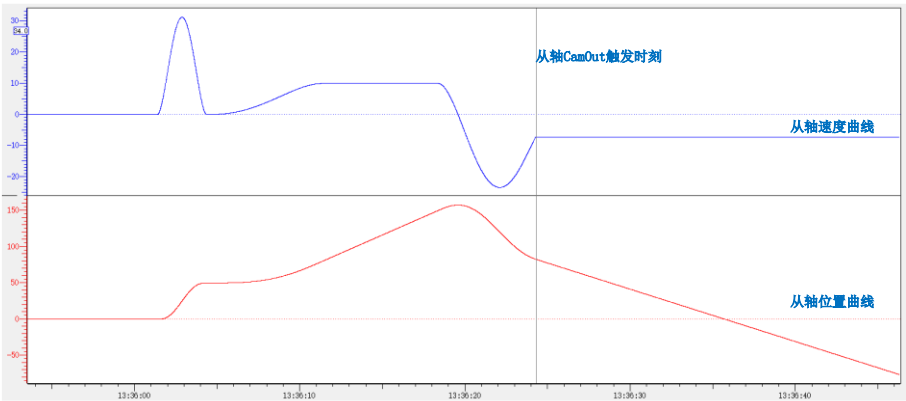
凸轮主轴和凸轮从轴的偏移参数能够在不改变凸轮表的情况下, 改变凸轮从轴的运动轨迹, 计算公式如下所示, 其中 fcam 表示电子凸轮表从轴和主轴的位置关系式:
SlavePosition = fcam(MasterPosition + MasterOffset) + SlaveOffset

6.1.8.3 MC_CamOut

描述: 该功能块解除从轴和主轴的电子凸轮耦合关系, 并使从轴以退出凸轮状态瞬间的速度保持连续运动, 轴状态由 Synchronized Motion 切换到 Continuous Motion 状态。

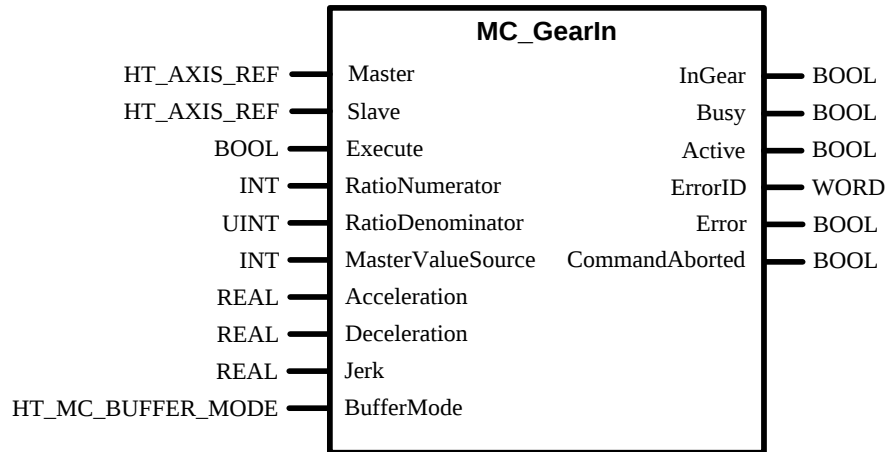


参数	数据类型	描述
VAR_IN_OUT		
Slave	HT_AXIS_REF	从动轴
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	主从轴解除凸轮耦合关系完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码



6.1.8.4 MC_GearIn

该功能块使主从轴按照设定的速度比例进行同步运行。



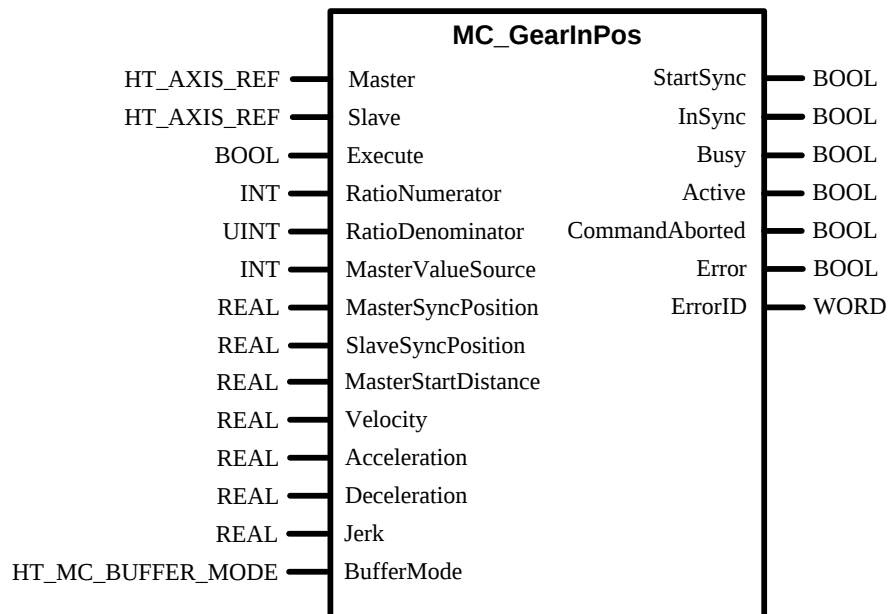
参数	数据类型	描述
VAR_IN_OUT		
Master	HT_AXIS_REF	主轴
Slave	HT_AXIS_REF	从轴
VAR_INPUT		
Execute	BOOL	上升沿触发
RatioNumerator	INT	电子齿轮分子
RatioDenominator	UINT	电子齿轮分母
MasterValueSource	INT	主轴数据来源，0：主轴命令速度；1：主轴反馈速度
Acceleration	REAL	主从同步过程中从轴最大加速度
Deceleration	REAL	主从同步过程中从轴最大减速度
Jerk	REAL	主从同步过程中从轴最大加加速度
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为
VAR_OUTPUT		
InGear	BOOL	TRUE 表示主从轴耦合中
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意:

- (1) 从轴的轴号必须大于主轴的轴号

6.1.8.5 MC_GearInPos

该功能块为轴的高精度电子齿轮位置耦合功能，使主从轴在设定的位置时，速度达到同步。

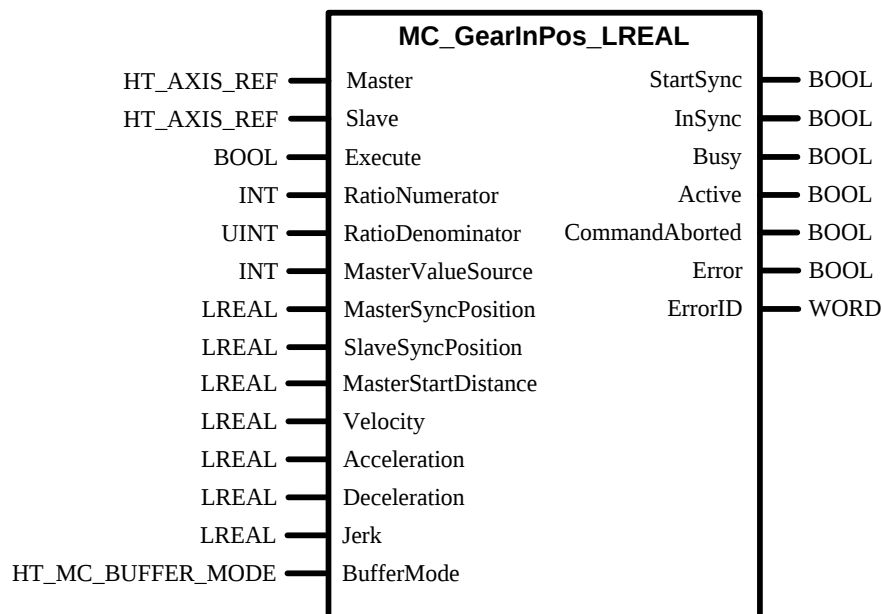


参数	数据类型	描述
VAR_IN_OUT		
Master	HT_AXIS_REF	主轴
Slave	HT_AXIS_REF	从轴
VAR_INPUT		
Execute	BOOL	上升沿触发
RatioNumerator	INT	电子齿轮分子
RatioDenominator	UINT	电子齿轮分母
MasterValueSource	INT	主轴数据来源，0：主轴命令速度；1：主轴反馈速度
MasterSyncPosition	REAL	同步结束时，主轴的位置值
SlaveSyncPosition	REAL	同步结束时，从轴的位置值
MasterStartDistance	REAL	主从轴同步过程的距离
Velocity	REAL	主从轴同步过程中从轴的最大速度
Acceleration	REAL	主从同步过程中从轴最大加速度

Deceleration	REAL	主从同步过程中从轴最大减速度
Jerk	REAL	主从同步过程中从轴最大加加速度
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为
VAR_OUTPUT		
StartSync	BOOL	TRUE 表示主从轴开始同步中
InSync	BOOL	TRUE 表示主从轴同步中
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.8.6 MC_GearInPos_LREAL

该功能块为轴的高精度电子齿轮位置耦合功能，当 REAL 类型数据不能满足精度要求时候，调用此功能块，在触发该功能块后，从轴和主轴进行耦合。

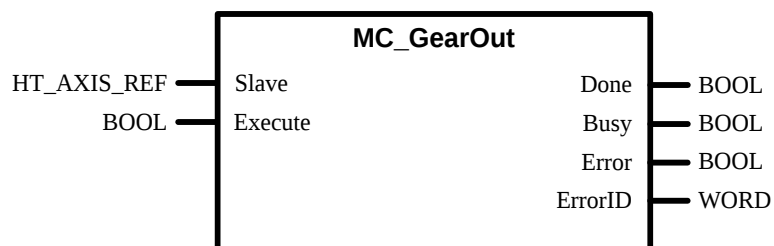


参数	数据类型	描述
VAR_IN_OUT		
Master	HT_AXIS_REF	主轴
Slave	HT_AXIS_REF	从轴

VAR_INPUT		
Execute	BOOL	上升沿触发
RatioNumerator	INT	电子齿轮分子
RatioDenominator	UINT	电子齿轮分母
MasterValueSource	INT	主轴数据来源，0：主轴命令速度；1：主轴反馈速度
MasterSyncPosition	LREAL	同步结束时，主轴的位置值
SlaveSyncPosition	LREAL	同步结束时，从轴的位置值
MasterStartDistance	LREAL	主从轴同步过程的距离
Velocity	LREAL	主从轴同步过程中从轴的最大速度
Acceleration	LREAL	主从同步过程中从轴最大加速度
Deceleration	LREAL	主从同步过程中从轴最大减速度
Jerk	LREAL	主从同步过程中从轴最大加加速度
BufferMode	HT_MC_BUFFER_MODE	该值用来设定轴的转换行为
VAR_OUTPUT		
StartSync	BOOL	TRUE 表示主从轴开始同步中
InSync	BOOL	TRUE 表示主从轴同步中
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.8.7 MC_GearOut

描述：该功能块解除从轴和主轴的电子齿轮耦合关系，并使从轴以退出齿轮状态瞬间的速度保持连续运动，轴状态由 Synchronized Motion 切换到 Continous Motion 状态。

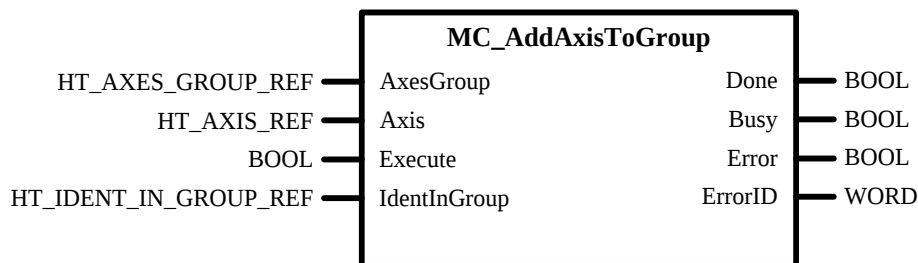


参数	数据类型	描述
VAR_IN_OUT		

Slave	HT_AXIS_REF	从轴
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.8.8 MC_AddAxisToGroup

描述：该功能块将轴加入到轴组中。



参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
Axis	AXIS_REF	参考轴
VAR_INPUT		
Execute	BOOL	上升沿触发
IdentInGroup	HT_IDENT_IN_GROUP_REF	轴在轴组中标志
VAR_OUTPUT		
Done	BOOL	加入轴组完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

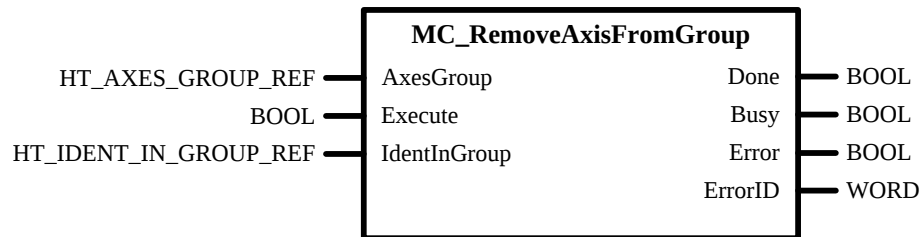
注意：

- (1) 轴加入轴组，允许的轴组状态：GroupStandBy、GroupDisabled、GroupErrorStop；
- (2) 轴加入轴组时，允许的轴状态：StandStill；

- (3) 加入轴组的轴必须先回零;
- (4) IdentInGroup.axisNo : 指定加入轴在轴组中的位置, 范围是 0-7 ;
IdentInGroup.homeSequence: 指定加入轴在轴组中的回零顺序, 范围是 0-7;
- (5) 加入轴组时, 轴运动队列必须为空;

6.1.8.9 MC_RemoveAxisFromGroup

描述: 该功能块将轴从轴组中移除。



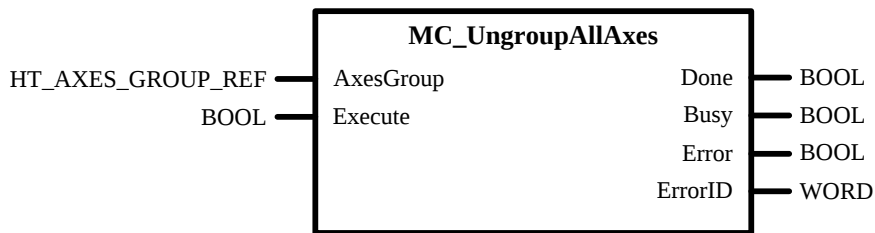
参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
IdentInGroup	HT_IDENT_IN_GROUP_REF	轴在轴组中标志
VAR_OUTPUT		
Done	BOOL	移除轴完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意:

- (1) 轴移除轴组, 允许的轴组状态: GroupStandBy、GroupDisabled、GroupErrorStop;
- (2) 如果轴移除后轴组中没有轴, 轴组状态切换到 GroupDisabled;
- (3) IdentInGroup.axisNo: 指定移除轴在轴组中的位置, 范围是 0-7;
- (4) 移除轴时, 轴组的运动队列必须为空。

6.1.8.10 MC_UngroupAllAxes

描述: 该功能块将轴组解除, 将轴组中所有轴移除。



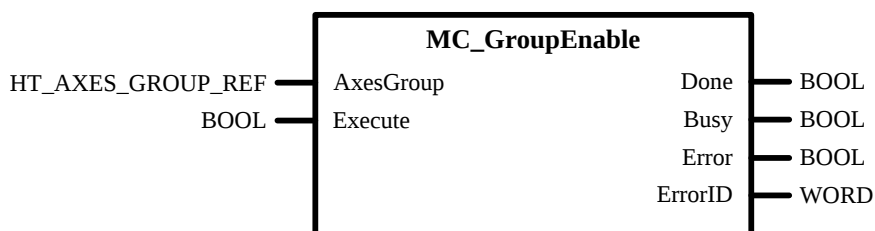
参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	解除轴组完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意：

- (1) 解除轴组，允许的轴组状态：GroupStandBy、GroupDisabled、GroupErrorStop；
- (2) 解除轴组，轴组状态切换到 GroupDisabled；
- (3) 解除轴组时，轴组的运动队列必须为空。

6.1.8.11 MC_GroupEnable

描述：该功能块将轴组上使能，使轴组状态从 GroupDisabled 切换到 GroupStandBy。



参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		

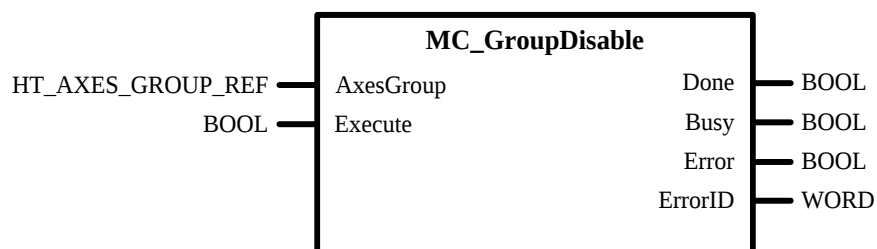
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	轴组上使能完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意：

- (1) 轴组上使能，允许的轴组状态： GroupDisabled;
- (2) 轴组上使能完成后，将轴组状态切换到 GroupStandby。

6.1.8.12 MC_GroupDisable

描述：该功能块将轴组下使能，使轴组状态从 GroupStandBy 切换到 GroupDisabled。



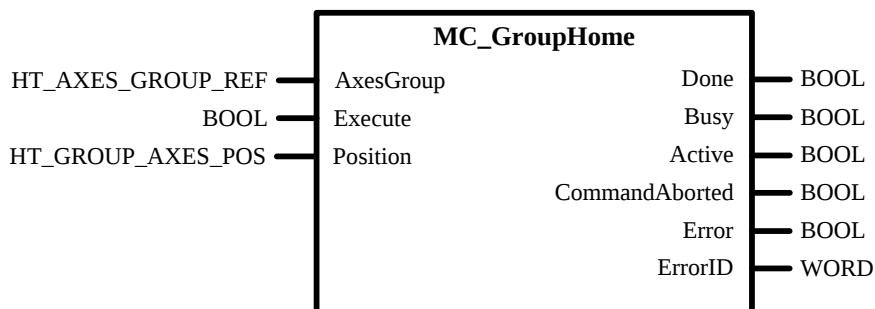
参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	轴组下使能完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意：

- (1) 轴组下使能，允许的轴组状态： 除了 GroupDisabled;
- (2) 轴组下使能完成后，将轴组状态切换到 GroupDisabled。

6.1.8.13 MC_GroupHome

描述：该功能块将轴组回到 Position 指定的位置。



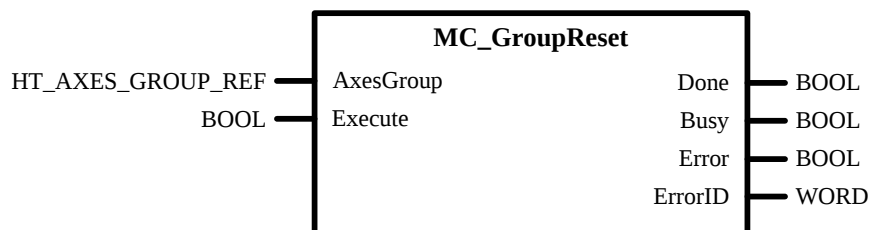
参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
Position	HT_GROUP_AXES_POS	各个轴需要回到的位置(轴组中没有加入轴的位置值无效)
VAR_OUTPUT		
Done	BOOL	轴组回原地完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意：

- (1) 轴组回零，允许的轴组状态： GroupStandby;
- (2) 轴组回零时，轴组状态： GroupHoming;
- (3) 轴组回零完成，轴组状态： GroupStandby;
- (4) 轴组回零时根据轴组中各轴的回零顺序，依次运动到 position 位置（如果回零顺序相同，则根据轴号决定回零顺序）;
- (5) 轴组回零时各轴运动到 position 位置的速度：当轴参数 search_vel 不等于 0 时，各轴以 search_vel 运动到 position 位置；当轴参数 search_vel 等于 0 时，各轴以 0.01*各轴的最大速度运动到 position 位置；
- (6) CoordSystem、BufferMode 为预留参数，暂未使用。

6.1.8.14 MC_GroupReset

描述：该功能块将轴组错误复位。



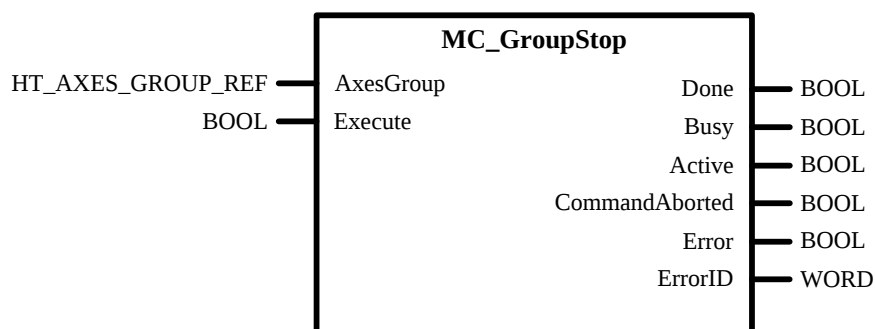
参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	轴组错误复位完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意：

- (1) 轴组下使能，允许的轴组状态： GroupErrorStop;
- (2) 轴组错误复位完成后，如果轴组在下使能状态，将轴组状态切换到 GroupDisabled; 如果轴组在使能状态，将轴组切换到 GroupStandBy。

6.1.8.15 MC_GroupStop

描述：该功能块将轴组运动停止，并中止任何正在执行的功能块。



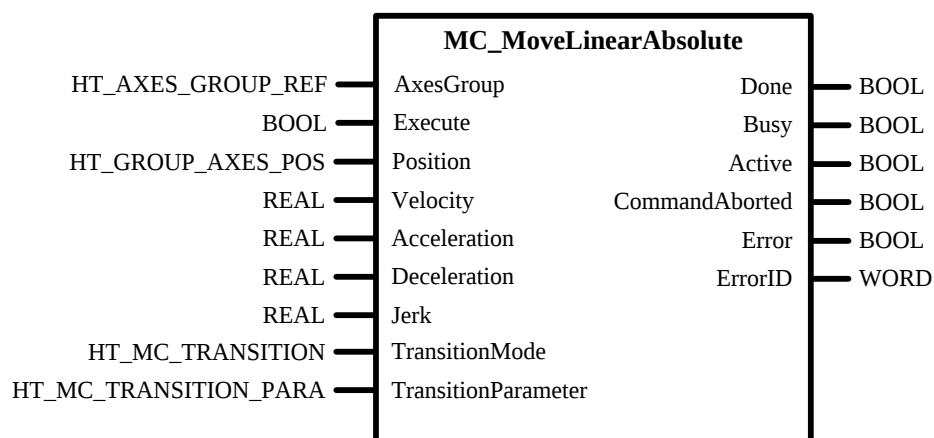
参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	轴组停止完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

注意：

- (1) 轴组停止功能块，允许的轴组状态： GroupStandBy、GroupHoming、GroupMoving；
- (2) 轴组停止时，只要轴组中各轴速度未到 0 或者 Execute 为 True，轴组状态仍处于 GroupStopping；
- (3) 轴组处于状态 GroupStopping 时，其他功能块不能使该轴组上执行任何运动；
- (4) 轴组停止执行完成后且 Execute 为 True，轴组状态一直处于 GroupStopping；
- (5) 轴组停止执行完成后且 Execute 为 False，轴组状态 GroupStopping 切换到 GroupStandBy；
- (6) 轴组停止功能块只能被 Mc_GroupDisable 中止。

6.1.8.16 MC_MoveLinearAbsolute

描述：该功能块实现轴组的直线插补绝对运动。

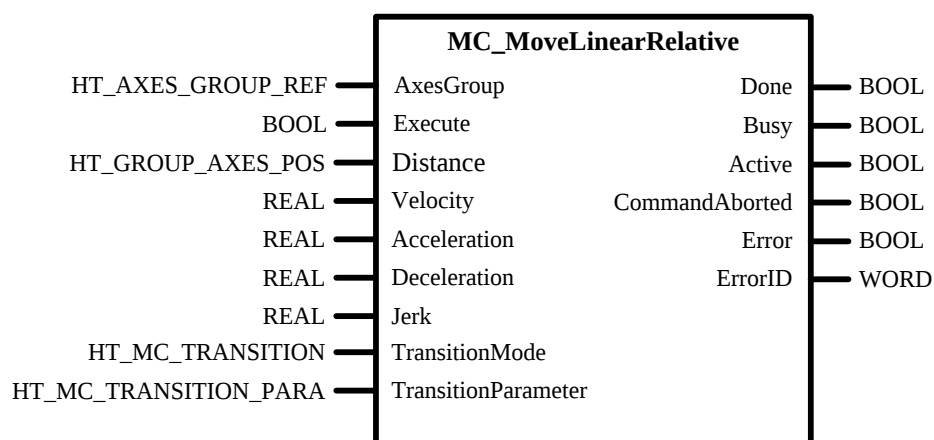


参数	数据类型	描述
----	------	----

VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
Position	HT_GROUP_AXES_POS	各个轴需要运动到的绝对位置(轴组中没有加入轴的位置值无效)
Velocity	REAL	运动速度
Acceleration	REAL	运动加速度
Deceleration	REAL	运动减速度 (暂未使用, 预留参数)
Jerk	REAL	运动加加速度 (暂未使用, 预留参数)
TransitionMode	HT_MC_TRANSITION	轴组过渡模式 (TMNone := 0, TMCornerRadius := 3)
TransitionParameter	HT_MC_TRANSITION_PARA	轴组过渡模式预留参数数组
VAR_OUTPUT		
Done	BOOL	轴组直线插补绝对运动完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.8.17 MC_MoveLinearRelative

描述：该功能块实现轴组的直线插补相对运动。

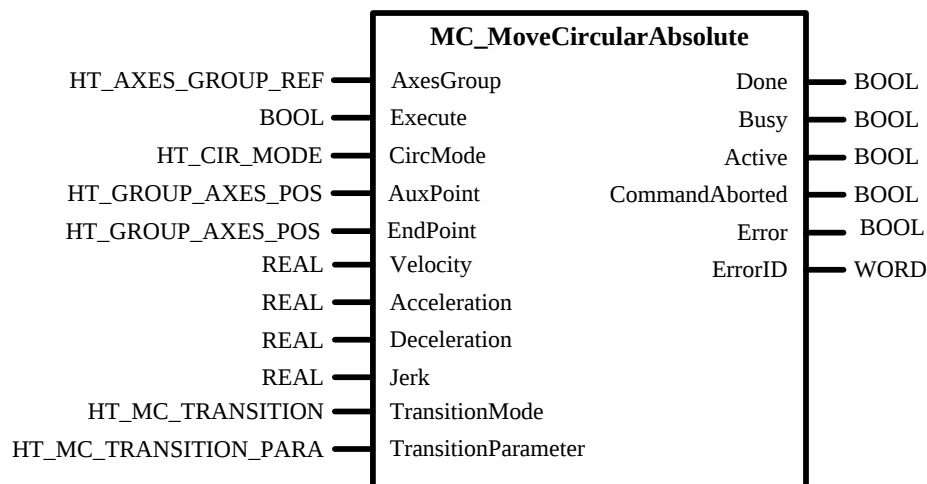


参数	数据类型	描述
----	------	----

VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
Distance	HT_GROUP_AXES_POS	各个轴需要运动到的相对位置(轴组中没有加入轴的位置值无效)
Velocity	REAL	运动速度
Acceleration	REAL	运动加速度
Deceleration	REAL	运动减速度 (暂未使用, 预留参数)
Jerk	REAL	运动加加速度 (暂未使用, 预留参数)
TransitionMode	HT_MC_TRANSITION	轴组过渡模式 (TMNone := 0, TMCornerDistance := 3)
TransitionParameter	HT_MC_TRANSITION_PARA	轴组过渡模式预留参数数组
VAR_OUTPUT		
Done	BOOL	轴组直线插补相对运动完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.8.18 MC_MoveCircularAbsolute

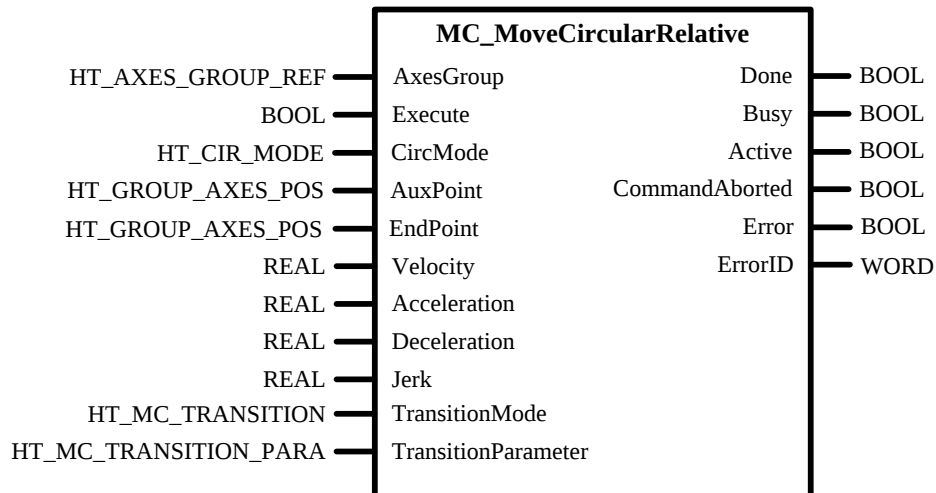
描述：该功能块实现轴组的圆弧插补绝对运动。



参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
CircMode	HT_CIR_MODE	目前圆弧运动只支持 BORDER 模式 (BORDER :=0)
AuxPoint	HT_GROUP_AXES_POSITION	圆弧的中间点 (取轴组中激活的前三个轴数据)
EndPoint	HT_GROUP_AXES_POSITION	圆弧的末端点 (取轴组中激活的前三个轴数据)
Velocity	REAL	运动速度
Acceleration	REAL	运动加速度
Deceleration	REAL	运动减速度 (暂未使用, 预留参数)
Jerk	REAL	运动加加速度 (暂未使用, 预留参数)
TransitionMode	HT_MC_TRANSITION	轴组过渡模式 (TMNone := 0, TMCornerDistance :=3)
TransitionParameter	HT_MC_TRANSITION_PARAMETER	轴组过渡模式预留参数数组
VAR_OUTPUT		
Done	BOOL	轴组圆弧插补绝对运动完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.8.19 MC_MoveCircularRelative

描述: 该功能块实现轴组的圆弧插补相对运动。

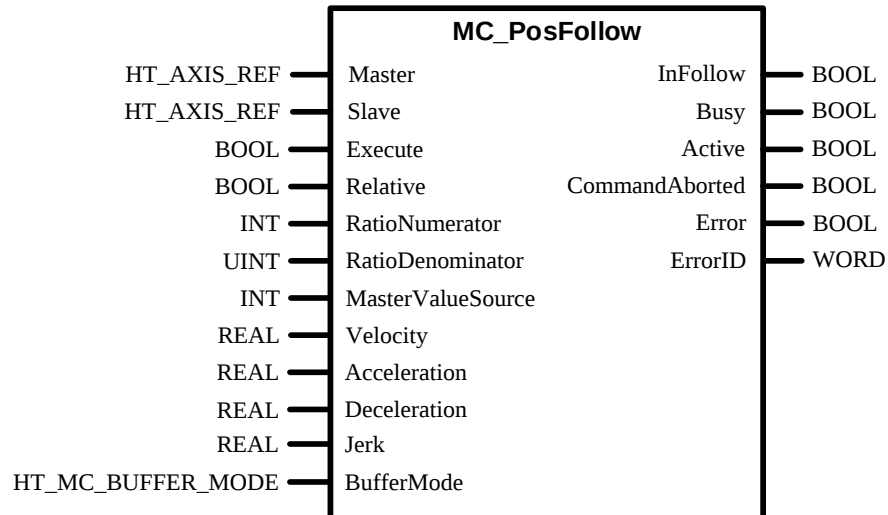


参数	数据类型	描述
VAR_IN_OUT		
AxesGroup	HT_AXES_GROUP_REF	参考轴组
VAR_INPUT		
Execute	BOOL	上升沿触发
CircMode	HT_CIR_MODE	目前圆弧运动只支持 BORDER 模式 (BORDER :=0)
AuxPoint	HT_GROUP_AXES_POS	圆弧的中间点 (取轴组中激活的前三个轴数据)
EndPoint	HT_GROUP_AXES_POS	圆弧的末端点 (取轴组中激活的前三个轴数据)
Velocity	REAL	运动速度
Acceleration	REAL	运动加速度
Deceleration	REAL	运动减速度 (暂未使用, 预留参数)
Jerk	REAL	运动加加速度 (暂未使用, 预留参数)
TransitionMode	HT_MC_TRANSITION	轴组过渡模式 (TMNone := 0, TMCornerDistance :=3)
TransitionParameter	HT_MC_TRANSITION_PARA	轴组过渡模式预留参数数组
VAR_OUTPUT		
Done	BOOL	轴组圆弧插补相对运动完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误

ErrorID	WORD	错误代码
---------	------	------

6.1.8.20 MC_PosFollow

描述：该功能块使主从轴进行耦合，从轴跟随运动。

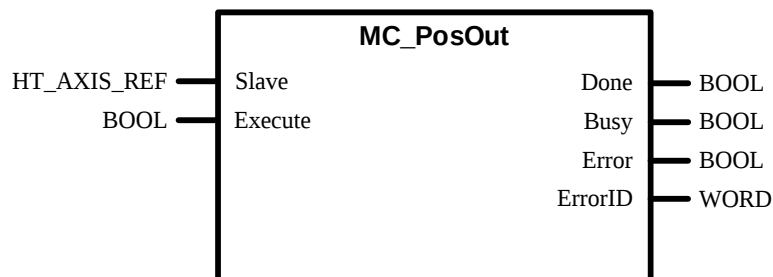


参数	数据类型	描述
VAR_IN_OUT		
Master	HT_AXIS_REF	主动轴
Slave	HT_AXIS_REF	从动轴
VAR_INPUT		
Execute	BOOL	上升沿触发
Relative	BOOL	位置跟随模式（0：相对，1：绝对）
RatioNumerator	INT	比例分子
RatioDenominator	UINT	比例分母
MasterValueSource	INT	主轴位置来源（0：命令值，1：反馈值）
Velocity	REAL	跟随时的最大速度
Acceleration	REAL	跟随时的最大加速度
Deceleration	REAL	跟随时的最大减速度
Jerk	REAL	跟随时的最大加加速度（暂未使用，预留参数）
BufferMode	HT_MC_BUFFER_MODE (INT)	该值用来设定轴的转换行为（暂未使用，预留参数）
VAR_OUTPUT		
InFollow	BOOL	TRUE 表示从轴跟随状态

Busy	BOOL	TRUE 表示功能块的执行还没有结束
Active	BOOL	TRUE 表示功能块正在执行中
CommandAborted	BOOL	TRUE 表示命令的执行被另外一个命令请求中断了
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.8.21 MC_PosOut

描述：该功能块解除从轴和主轴的位置跟随耦合关系。



参数	数据类型	描述
VAR_IN_OUT		
Slave	HT_AXIS_REF	从动轴
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	主从轴解除位置跟随耦合关系完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.1.9 运动功能块错误说明

当调用 MC_LIB 功能块时，有时会出现报错，具体的报错信息如下表所示。

ErrorID	详细说明	解决方法
1	不允许的状态	检查当前轴状态
2	无效的轴号	检查轴号是否正确
3	轴未配置	检查轴是否激活
4	轴未使能	检查轴是否上使能
5	无效的位置值 Pos	检查位置值是否正确
6	无效的速度值 Vel	检查速度值是否正确

7	无效的加速度值 Acc	检查加速度值是否正确
8	无效的减速度值 Dec	检查减速度值是否正确
9	无效的急动度值 Jerk	检查加加速度值是否正确
10	无效的方向值 Direction	检查方向是否正确
11	无效的 BUFF_MODE 值	检查 bufferMode 值是否正确
12	无效的参数号	检查参数号是否正确
13	错误的 AXIS_REF 结构体	检查 AXIS_REF 是否正确
14	无效的距离值	检查距离是否正确
15	伺服未下使能	检查伺服使能状态是否正确
16	无效的 SYS_STATUS 结构体	检查 SYS_STATUS 结构体是否正确
20	无效的运动类型值	检查运动类型是否正确
21	无效的软件限位值	检查软限位值是否正确
22	无效的最大速度值	检查最大速度值是否正确
23	无效的最大加速度值	检查最大加速度值是否正确
24	无效的最大减速度值	检查最大减速度值是否正确
25	无效的最大急动度值	检查最大加加速度值是否正确
26	无效的最大静态误差	检查最大静态误差值是否正确
27	无效的最大动态误差	检查最大动态误差值是否正确
28	无效的编码器输入线束	检查编码器输入参数是否正确
29	无效的编码器输出线束	检查编码器输出参数是否正确
30	无效的电子齿轮比	检查电子齿轮比参数是否正确
31	无效的电子齿轮同步位移	检查电子齿轮同步位移是否正确
32	电子齿轮比例错误	检查电子齿轮的速度比例值是否正确
33	无效的速度规划类型	检查速度规划参数是否正确
34	无效的电子凸轮偏移值	检查电子凸轮偏移值是否正确
35	无效的电子凸轮缩放值	检查电子凸轮缩放值是否正确
36	无效的电子凸轮启动模式	检查电子凸轮启动模式是否正确
37	无效的电子凸轮表	检查电子凸轮表是否正确
38	无效的电子凸轮同步位移	检查电子凸轮同步位移参数
39	轴没有执行电子凸轮	请检查当前轴是否在执行电子凸轮
40	无效的控制模式	请检查控制模式参数是否正确
41	轴没有执行电子齿轮	请检查当前轴是否在执行电子齿轮
42	轴未回零	请先将轴回零
43	MC_ABS 功能块写命令出错	请进行复位操作
44	MC_POWER 功能块写命令出错	请进行复位操作
45	MC_PLATFORM 功能块没有更新	请进行复位操作

46	MC_PLATFORM 更新状态出错	请进行复位操作
47	MC_HOME 功能块写命令出错	请进行复位操作
48	MC_RELATIVE 功能块写命令出错	请进行复位操作
49	MC_STOP 功能块写命令出错	请进行复位操作
50	MC_WRITE_DOUBLE_PARA 功能块写命令出错	请进行复位操作
51	MC_WRITE_BOOL_PARA 功能块写命令出错	请进行复位操作
52	MC_CONFIG 功能块写命令出错	请进行复位操作
53	AXIS_REF 结构体轴号错误	请检查轴号是否正确
54	MC_VELOCITY 功能块写命令出错	请进行复位操作
55	MC_RESET 功能块写命令出错	请进行复位操作
56	MC_CONTINOUS_ABS 功能块写命令出错	请进行复位操作
57	MC_CONTINOUS_RELATIVE 功能块写命令出错	请进行复位操作
58	MC_GEAR_IN 功能块写命令出错	请进行复位操作
59	MC_GEAR_OUT 功能块写命令出错	请进行复位操作
60	MC_GEAR_IN_POS 功能块写命令出错	请进行复位操作
61	MC_SET_POSITION 功能块写命令出错	请进行复位操作
62	MC_CAM_IN 功能块写命令错误	请进行复位操作
63	MC_CAM_OUT 功能块写命令错误	请进行复位操作
70	轴编码器错误	请进行复位操作
71	轴跟随误差错误	请进行复位操作
73	轴未使能超时错误	请进行复位操作
100	运动控制器有错误	请进行复位操作
101	运动控制使能前伺服未准备就绪	请进行复位操作
102	模式错误	请进行复位操作
103	运动控制器未使能错误	请进行复位操作
104	轨迹超安全区域	请进行复位操作
105	轨迹超硬限位区域	请进行复位操作
106	添加运动段错误	请进行复位操作
107	命令执行错误	请进行复位操作
108	在回零过程中不能执行 JOG 命令	请进行复位操作
110	外部伺服错误	请进行复位操作
111	DPRAM 数据通信错误	请进行复位操作

112	HAL 通信掉 OP	请进行复位操作
113	HAL 长时间不 OP	请进行复位操作
114	脉冲板从站反馈出现大值	请进行复位操作
115	脉冲板从站读/写 CRC 校验失败	请进行复位操作
116	脉冲板从站丢脉冲错误	请进行复位操作
117	未探测到点错误	请进行复位操作
118	自动模式下碰撞错误	请进行复位操作
119	摇杆模式下碰撞错误	请进行复位操作
120	回零模式下碰撞错误	请进行复位操作
121	自由模式下碰撞错误	请进行复位操作
122	摇杆模式下回退向量计算错误	请进行复位操作
123	自动模式下回退向量计算错误	请进行复位操作
124	添加直线段错误	请进行复位操作
125	添加圆弧段错误	请进行复位操作
126	添加攻丝段错误	请进行复位操作
127	插补计算超过软限位	请进行复位操作
128	命令处理错误	请进行复位操作
129	加入轴组的自由轴存在错误	请进行复位操作
130	轴正硬限位错误	请进行复位操作
131	轴负硬限位错误	请进行复位操作
132	轴正软限位错误	请进行复位操作
133	轴负软限位错误	请进行复位操作
134	轴伺服报警错误	请进行复位操作
135	轴跟随误差错误	请进行复位操作
136	轴超速错误	请进行复位操作
137	轴回零错误	请进行复位操作
138	轴超加速度错误	请进行复位操作
140	轴错误	请进行复位操作
141	无效的轴速度修调值	请检查速度修调值是否正确
142	无效的轴加速度修调值	请检查加速度修调值是否正确
143	无效的轴加加速度修调值	请检查加加速度修调值是否正确
144	写速度修调命令错误	请进行复位操作
145	轴组未下使能，而运动控制器下使能	请进行复位操作
146	插补过程中，轴出现超速错误	请进行复位操作
147	轴位置命令超过 INT 最大值	请进行复位操作，并将轴往反方向运动
200	功能块还在运行中	请等待功能块运行完成，再重新触发

201	轴组结构体错误	请检查结构体是否正确
202	轴组中轴标识错误	请检查轴标识是否正确
203	轴组中该轴没有激活，不能移除	请选择激活的轴进行操作
204	MC_RemoveAxisFromGroup 写命令错误	请进行复位操作
205	MC_AddAxisToGroup 写命令错误	请进行复位操作
206	MC_UngroupAllAxes 写命令错误	请进行复位操作
207	MC_group_Enable 写命令错误	请进行复位操作
208	MC_group_Disable 写命令错误	请进行复位操作
209	MC_group_Reset 写命令错误	请进行复位操作
210	MC_group_Stop 写命令错误	请进行复位操作
211	MC_MoveLinearAbsolute 写命令错误	请进行复位操作
212	MC_MoveCircularAbsolute 写命令错误	请进行复位操作
213	无效的轴组号	请检查轴组号是否正确
214	无效的轴组状态	请检查当前的轴组状态
215	直线运动终点和起点相同错误	请检查位置参数是否正确
216	圆弧模式错误	请检查输入参数是否正确
217	圆弧方向错误	请检查输入参数是否正确
218	圆弧需要的位置点不够	请检查激活轴数是否正确
219	圆弧三点共线	请检查位置参数是否正确
220	无效的 ident 值	请检查 ident 是否正确
221	在 ident 下已经存在轴，不允许加入新的轴	请确保 ident 下没有加入轴
222	在 ident 下不存在轴，不允许移除轴	请确保 ident 下存在轴
223	轴已经加入轴组中	请确保轴只能加入轴组一次
224	轴组未使能	请检查轴组是否使能
225	轴组中轴数错误	请检查轴组中的轴数是否正确
226	轴在轴组中，不允许执行单轴运动命令	请检查轴是否加入到了轴组中
227	轴标识结构体错误	请检查结构体是否正确
228	功能块输入位置参数错误	请检查位置参数输入是否正确
229	单轴队列不为空，不能加入轴组	请确保单轴队列为空后，再加入轴组
230	轴组队列不为空，不能将轴从轴组中移除	请确保轴组为空后，再将轴从轴组中移除
300	凸轮表文件名长度错误，小于 8 字符	请修改凸轮表文件名
301	凸轮表文件不存在	请检查凸轮表文件名是否正确
302	打开凸轮表数量超限	请先退出凸轮表执行
303	分配凸轮表内存错误	请进行复位操作
304	凸轮表类型错误	请检查凸轮表是否正确
305	凸轮表语法错误	请检查凸轮表是否正确

306	凸轮表 MC_CAM_ID 长度错误	请检查 MC_CAM_ID 结构体是否正确
307	凸轮表没有打开	请先打开凸轮表
308	计算凸轮表系数错误	请检查凸轮表是否正确
309	凸轮表 ID 错误	请检查打开的凸轮表是否正确
310	打开的凸轮表的主从轴轴号和 MC_CamIn 的主从轴轴号不相同	请检查 MC_CamTableSelect 或者 MC_CamIn 的主从轴 是否一致
311	凸轮表第一个点和最后一个点速度不同	请检查凸轮表是否正确
312	凸轮表第一个点和最后一个点加速度不同	请检查凸轮表是否正确
313	凸轮表主轴位置点没有递增或者小于 0	请检查凸轮表是否正确
314	从轴和主轴轴号相同错误	请检查主轴轴和从轴是否用了同一个轴
400	MC_POS_FOLLOW 功能块写命令错误	请检查 MC_POS_FOLLOW 输入是否正确
401	MC_POS_OUT 功能块写命令错误	请检查 MC_POS_OUT 输出是否正确
402	MC_TorqueOut 功能块写命令错误	请检查 MC_TorqueOut 输出是否正确
403	轴没有执行位置跟随	
404	主从轴位置偏差过大	请检查速度值和加速度值是否超出设定范围
405	无效的编码器类型	请检查编码器类型是否正确

6.2 通讯功能块

6.2.1 概览

HIC 控制器通讯功能块主要分 4 部分介绍，如表 2.1 所示

表 2.1 通讯功能块概览

CANopen	EtherCAT	Modbus	Module
CANopen_Status_Update (状态更新)	EtherCAT_Status_Update (状态更新)	Modbus_Read (读取)	HI_Module_AiFilterCfg (设置高速模拟量滤波)
CANopen_Read (读取)	EtherCAT_Read (读取)	Modbus_Write (写入)	HI_Module_CycleSet (设置高速通讯周期)
CANopen_Write (写入)	EtherCAT_Write (写入)		HI_Module_DiFilterCfg (设置高速数字量滤波)
CANopen_SDO_Read (SDO 读取)	EtherCAT_Map_Set (通用设备配置)		HI_Module_SSICfg (SSI 配置)
CANopen_SDO_Write (SDO 写入)	EtherCAT_Map_Get (通用设备读取)		HI_Module_StopAOSet (高速模拟量超时输出)
CANopen_Map_Set	EtherCAT_Reset		HI_Module_StopIOset

(通用设备配置)	(总线复位)		(高速数字量超时输出)
CANopen_Map_Get (通用设备读取)			Module_CycleSet (设置低速通讯周期)
CANopen_Reset (总线复位)			Module_StopAOSet (低速模拟量超时输出)
			Module_StopCycleSet (低速通讯超时周期)
			Module_StopIOSet (低速数字量超时输出)

6.2.2 COM_LIB 数据类型说明

COM_LIB 作为 HIC 控制器通讯功能库中，定义了一些特殊的数据类型，这节主要介绍包括 CANopen、EtherCAT、Modbus 部分的数据类型。

6.2.2.1 CANopen 配置结构体

(*CANopen 报文配置结构体*)

CANopen_MAP : STRUCT

Index : UINT; (*索引*)
SubIndex : BYTE; (*子索引*)
Value : UDINT; (*数值*)
ValueLen : BYTE; (*长度*)

END_STRUCT;

(*CANopen 报文配置数组*)

CANopen_MAP_ARRAY : STRUCT

PDOMapArray : ARRAY [0..29] OF CANopen_MAP;

END_STRUCT;

(*驱动器设备读取 PDO 结构体*)

CANopen_DRIVER_RX_PDO : STRUCT

Position : DINT; (*位置*)
StatusWord : UINT; (*状态字*)

END_STRUCT;

(*驱动器设备写入 PDO 结构体*)

CANopen_DRIVER_TX_PDO : STRUCT

Velocity : DINT; (*速度*)
ControlWord : UINT; (*控制字*)

END_STRUCT;

(*位置尺设备读取 PDO 结构体*)

```
CANopen_RULER_RX_PDO: STRUCT
    Position : DINT;          (*位置*)
    Velocity : INT;           (*速度*)
END_STRUCT;
```

(*通用设备 PDO 数据数组*)

```
CANopen_USER_DEFINE_PDO : STRUCT
    PdoBuffer : ARRAY [0..7] OF BYTE;
END_STRUCT;
```

(*CANopenPDO 数据读取联合体*)

```
CANopen_READ_UNION : UNION
    HiServo : CANopen_DRIVER_RX_PDO;
    Ruler : CANopen_RULER_RX_PDO;
    UserDefine : CANopen_USER_DEFINE_PDO;
END_UNION;
```

(*CANopenPDO 数据写入联合体*)

```
CANopen_WRITE_UNION : UNION
    HiServo : CANopen_DRIVER_TX_PDO;
    UserDefine : CANopen_USER_DEFINE_PDO;
END_UNION;
```

(*CANopenSDO 数据数组*)

```
CANopen_SDO_DATA : STRUCT
    data : ARRAY [0..9] OF BYTE;
END_STRUCT;
```

6.2.2.2 EtherCAT 配置结构体

(*EtherCAT 报文配置结构体*)

```
EtherCAT_MAP : STRUCT
    Index : UINT;             (*索引*)
    SubIndex : BYTE;          (*子索引*)
    Value : UDINT;            (*数值*)
    ValueLen : BYTE;          (*长度*)
END_STRUCT;
```

(*EtherCAT 报文配置数组*)

```
EtherCAT_MAP_ARRAY : STRUCT
    PDOMapArray : ARRAY [0..63] OF EtherCAT_MAP;
END_STRUCT
```

(*驱动器设备读取 PDO 结构体*)

```
HI_SERVO_RX_PDO : STRUCT
    StatusWord : UINT;          (*状态字*)
    Position : DINT;            (*位置*)
    Velocity : DINT;            (*速度*)
    Current : INT;              (*扭矩*)
    DI : UINT;                  (*数字量输入*)
    Voltage : UINT;
END_STRUCT;
```

(*驱动器设备写入 PDO 结构体*)

```
HI_SERVO_TX_PDO : STRUCT
    Control : UINT;             (*控制字*)
    Position : DINT;            (*位置*)
    Velocity : DINT;            (*速度*)
    Torque : INT;               (*扭矩*)
    Dout : UINT;                (*数字量输出*)
    Reserved : UINT;
END_STRUCT;
```

(*BALLUFF 位置尺设备读取 PDO 结构体*)

```
BALLUFF_RULER_RX_PDO : STRUCT
    Position : UDINT;           (*位置*)
    Velocity : DINT;            (*速度*)
    StatusWord : UINT;          (*状态字*)
END_STRUCT;
```

(*MTS 位置尺设备读取 PDO 结构体*)

```
MTS_RULER_RX_PDO : STRUCT
    StatusWord : UINT;          (*状态字*)
    Position : UDINT;           (*位置*)
    Velocity : DINT;            (*速度*)
END_STRUCT;
```

(*通用设备读取 PDO 数据数组*)

```
USER_DEFINE_RX_PDO : STRUCT
    RxPdo : ARRAY [0..63] OF BYTE;
END_STRUCT;
```

(*通用设备写入 PDO 数据数组*)

```
USER_DEFINE_TX_PDO : STRUCT
    TxPdo : ARRAY [0..63] OF BYTE;
END_STRUCT;
```

(*EtherCAT 数据读取联合体*)

```
EtherCAT_READ_UNION : UNION
    HiServo : HI_SERVO_RX_PDO;
    MtsRuler : MTS_RULER_RX_PDO;
    BalluffRuler : BALLUFF_RULER_RX_PDO;
    EtherCATModule : EtherCAT_MOULD_RX_PDO;
    UserDefine : USER_DEFINE_RX_PDO;
END_UNION;
```

(*EtherCAT 数据写入联合体*)

```
EtherCAT_WRITE_UNION : UNION
    HiServo : HI_SERVO_TX_PDO;
    EtherCATModule : EtherCAT_MOULD_TX_PDO;
    UserDefine : USER_DEFINE_TX_PDO;
END_UNION;
```

(*EtherCAT 从站类型数组*)

```
EtherCAT_TYPE_STRUCT :
STRUCT
    SlaveType : ARRAY [0..63] OF EtherCAT_SLAVE_TYPE; (*数据类型: 0: None; 1:
    HiServo; 2: BalluffRuler;
    3 : MtsRuler ; 4 :
    HicEthercatModule ; 5 :
    UserDefine; 6: EtherCatIo*)
END_STRUCT
```

6.2.2.3 Modbus 配置结构体

(*Modbus 读取数据数组*)

```
Modbus_READ_DATA :STRUCT
    data : ARRAY [0..199] OF BYTE;
END_STRUCT;
```

(*Modbus 写入数据数组*)

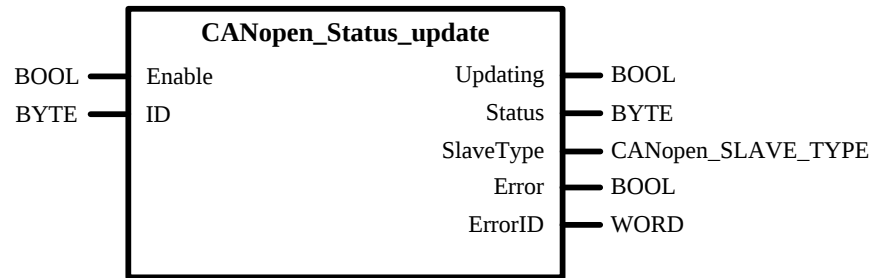
```
Modbus_WRITE_DATA :STRUCT
    data : ARRAY [0..99] OF BYTE;
END_STRUCT;
```

6.2.3 CANopen 通讯功能块

在本节中主要介绍 HIC 控制器 CANopen 通讯部分的功能块，接下来将对每一个功能块进行介绍

6.2.3.1 CANopen_Status_Update

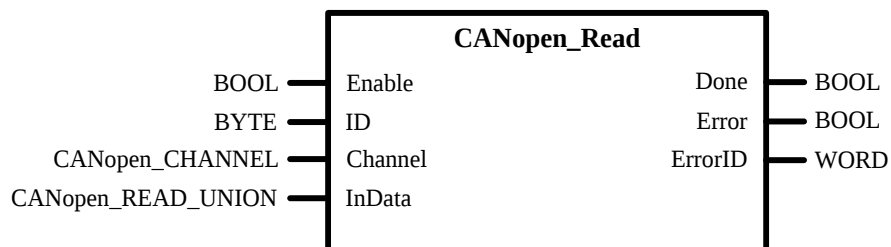
该功能块对 HIC 控制器的 CANopen 状态进行更新（非必要调用）。



参数	数据类型	描述
VAR_INPUT		
Enable	BOOL	如果 Enable = TRUE，表示执行该功能块
ID	BYTE	CANopen 从站 ID 号
VAR_OUTPUT		
Updating	BOOL	TRUE 表示功能块执行中
Status	BYTE	CANopen 从站通讯状态（未通讯:= 0，通讯正常:= 1）
SlaveType	CANopen_SLAVE_TYPE	CANopen 从站类型模式（None := 0,Ruler := 1,Driver := 2, UserDefine :=3）
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.3.2 CANopen_Read

该功能块对 HIC 控制器的 CANopen 从站 PDO 信息进行读取。

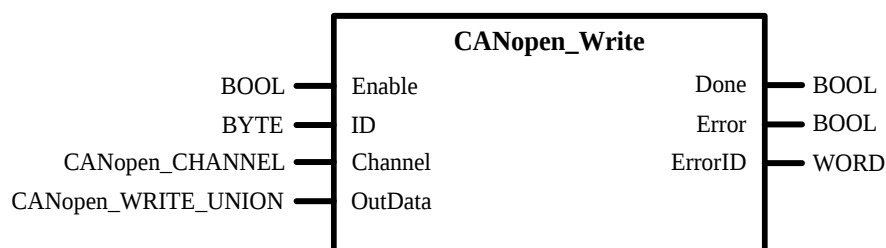


参数	数据类型	描述
VAR_IN_OUT		

InData	CANopen_READ_UNION	CANopen 读取的 PDO 数据信息
VAR_INPUT		
Enable	BOOL	如果 Enable = TRUE，表示执行该功能块
ID	BYTE	CANopen 从站 ID 号
Channel	CANopen_CHANNEL	通道号配置（ Ch1 := 0,Ch2 := 1,Ch3 := 2,Ch4 := 3 ）
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.3.3 CANopen_Write

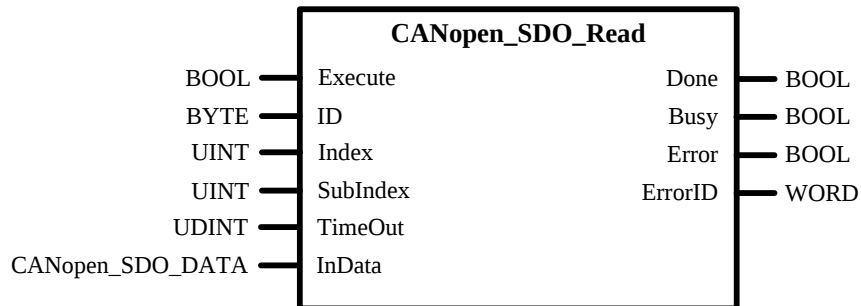
该功能块对 HIC 控制器的 CANopen 从站 PDO 信息进行写入。



参数	数据类型	描述
VAR_IN_OUT		
OutData	CANopen_WRITE_UNION	CANopen 写入的 PDO 数据信息
VAR_INPUT		
Enable	BOOL	如果 Enable = TRUE，表示执行该功能块
ID	BYTE	CANopen 从站 ID 号
Channel	CANopen_CHANNEL	通道号配置（ Ch1 := 0,Ch2 := 1,Ch3 := 2,Ch4 := 3 ）
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.3.4 CANopen_SDO_Read

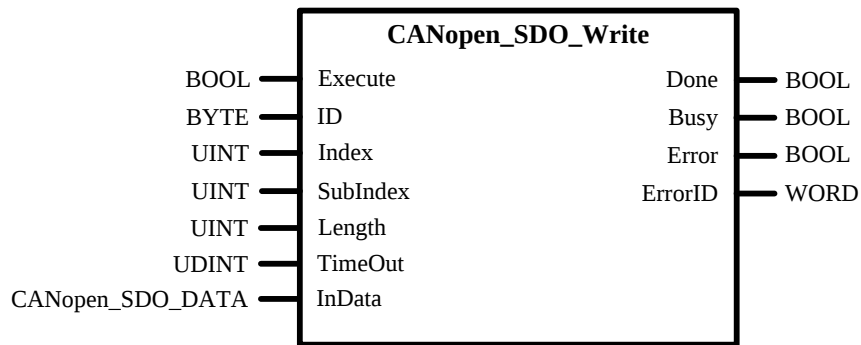
该功能块对 HIC 控制器的 CANopen 从站 SDO 信息进行读取。



参数	数据类型	描述
VAR_IN_OUT		
InData	CANopen_SDO_DATA	CANopen 读取的 SDO 数据信息
VAR_INPUT		
Execute	BOOL	上升沿触发
ID	BYTE	CANopen 从站 ID 号
Index	UINT	要读取的索引号
SubIndex	UINT	要读取的子索引号
TimeOut	UDINT	超时时间
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.3.5 CANopen_SDO_Write

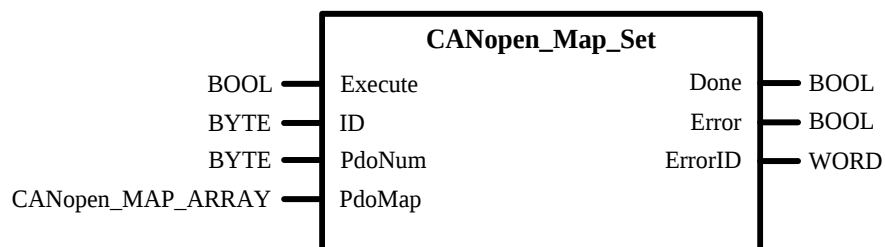
该功能块对 HIC 控制器的 CANopen 从站 SDO 信息进行写入。



参数	数据类型	描述
VAR_IN_OUT		
OutData	CANopen_SDO_DATA	CANopen 写入的 SDO 数据信息
VAR_INPUT		
Execute	BOOL	上升沿触发
ID	BYTE	CANopen 从站 ID 号
Index	UINT	要写入的索引号
SubIndex	UINT	要写入的子索引号
Length	UINT	长度
TimeOut	UDINT	超时时间
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.3.6 CANopen_Map_Set

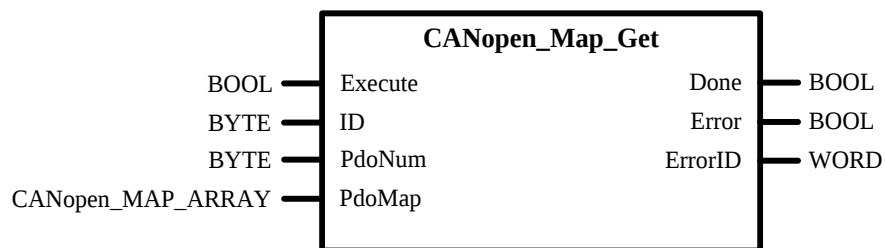
该功能块对 HIC 控制器的 CANopen 从站通用设备进行配置。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
ID	BYTE	CANopen 从站配置的 ID 号
PdoNum	BYTE	PDO 报文配置信息调用的个数
PdoMap	CANopen_MAP_ARRAY	PDO 报文配置信息
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.3.7 CANopen_Map_Get

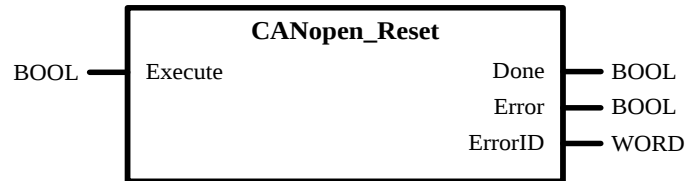
该功能块对 HIC 控制器的 CANopen 从站通用设备配置信息进行读取。



参数	数据类型	描述
VAR_IN_OUT		
PdoNum	BYTE	PDO 报文配置信息调用的个数读取
PdoMap	CANopen_MAP_ARRAY	PDO 报文配置信息读取
VAR_INPUT		
Execute	BOOL	上升沿触发
ID	BYTE	CANopen 从站配置的 ID 号
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.3.8 CANopen_Reset

该功能块对 HIC 控制器的 CAN 总线进行复位。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

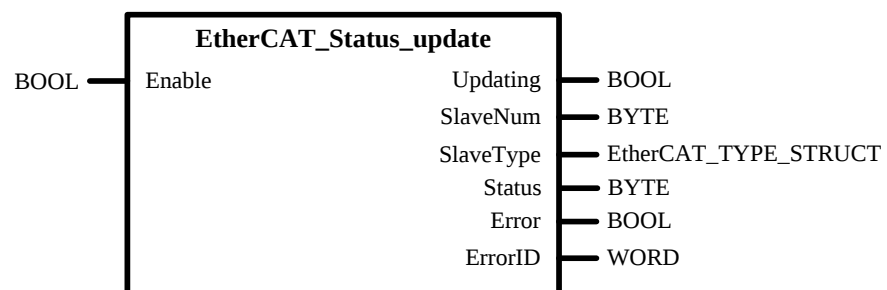
6.2.4 EtherCAT 通讯功能块

在本节中主要介绍 HIC 控制器 EtherCAT 通讯部分的功能块，接下来将对每一个功能块进行介绍

6.2.4.1 EtherCAT_Status_Update

该功能块对 HIC 控制器的 EtherCAT 状态进行更新。

注意：调用其它 EtherCAT 功能块之前，需要将该功能块一直使能，否则其它功能块将会报错。

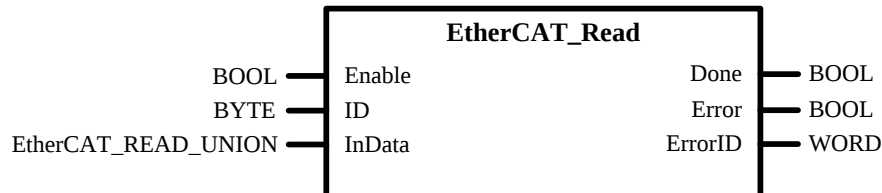


参数	数据类型	描述
VAR_INPUT		

Enable	BOOL	如果 Enable=TRUE，表示执行该功能块
VAR_OUTPUT		
Updating	BOOL	TRUE 表示功能块执行中
SlaveNum	BYTE	表示当前 EtherCAT 总线所连的从站个数
SlaveType	EtherCAT_TYPE_STRUC	EtherCAT 从站的类型
Status	BYTE	1 表示当前 EtherCAT 处于 OP 状态，0 表示当前 EtherCAT 处于非 OP 状态
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.4.2 EtherCAT_Read

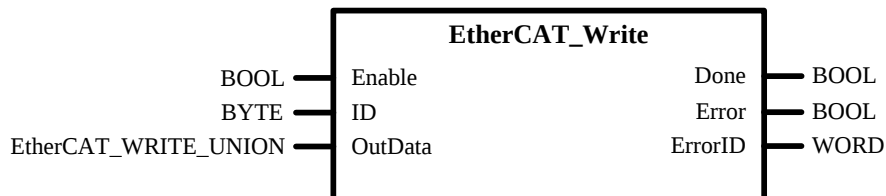
该功能块对 EtherCAT 数据进行读取，当从站类型为 HI SERVO 时，读取的数据存放于 InData. HiServo 中，当从站类型为 MTS 位置尺时，读取数据存放于 InData. MtsRuler 中，当从站类型为 BALLUFF 位置尺时，读取数据存放于 InData. BalluffRuler 中，当从站类型为通用设备时，读取数据存放于 InData. UserDefine 中。



参数	数据类型	描述
VAR_IN_OUT		
InData	EtherCAT_READ_UNION	EtherCAT 读数据内容
VAR_INPUT		
Enable	BOOL	如果 Enable=TRUE，表示执行该功能块
ID	BYTE	EtherCAT 从站的 ID 号
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.4.3 EtherCAT_Write

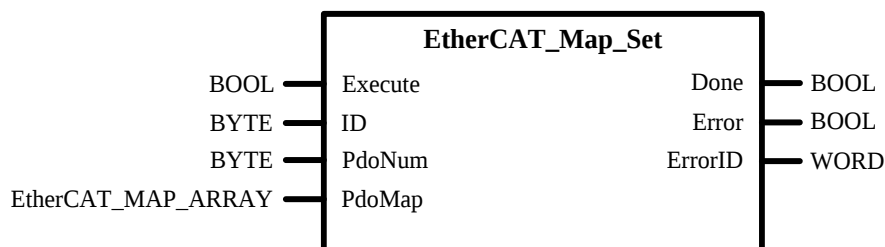
该功能块对 EtherCAT 数据进行写操作，当从站类型为 HI SERVO 时，写的数据存放于 OutData. HiServo 中，当从站类型为通用设备时，读取数据存放于 OutData. UserDefine 中。



参数	数据类型	描述
VAR_IN_OUT		
OutData	EtherCAT_WRITE_UNION	EtherCAT 写数据内容
VAR_INPUT		
Enable	BOOL	如果 Enable=TRUE，表示执行该功能块
ID	BYTE	EtherCAT 从站的 ID 号
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.4.4 EtherCAT_Map_Set

该功能块对 EtherCAT 设备进行对象字典配置。

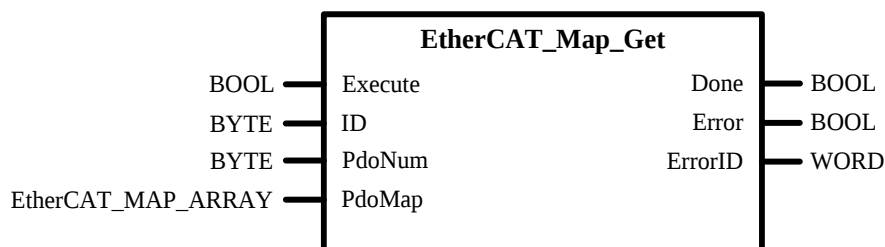


参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
ID	BYTE	EtherCAT 从站的 ID 号

PdoNum	BYTE	要配置的 PDO 数量
PdoMap	EtherCAT_MAP_ARRAY	要配置的 PDO 对象字典
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.4.5 EtherCAT_Map_Get

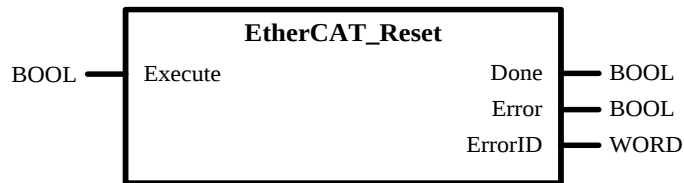
该功能块对 EtherCAT 设备进行对象字典配置信息进行读取



参数	数据类型	描述
VAR_IN_OUT		
PdoNum	BYTE	读到的 PDO 数量信息
PdoMap	EtherCAT_MAP_ARRAY	读到的 PDO 对象字典配置信息
VAR_INPUT		
Execute	BOOL	上升沿触发
ID	BYTE	EtherCAT 从站的 ID 号
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.4.6 EtherCAT_Reset

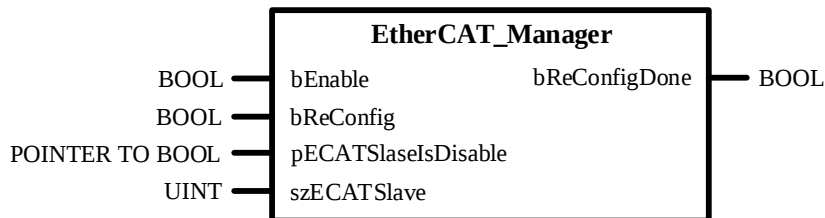
该功能块对 HIC 控制器的 EtherCAT 总线进行复位。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.4.7 EtherCAT_Manager

该功能块用于实现 EtherCAT 主站网络管理，可以控制从站是否启用



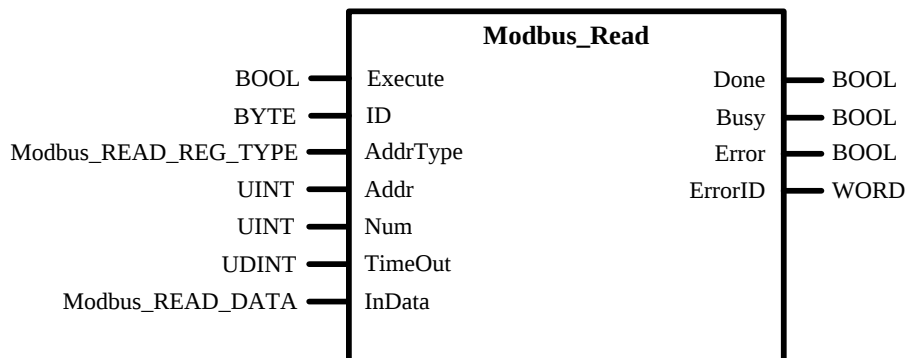
参数	数据类型	描述
VAR_INPUT		
bEnable	BOOL	TRUE: 启用功能块 FALSE: 关闭功能块
bReConfig	BOOL	TRUE: 重新配置主站
pECATSlavesDisable	POINT TO BOOL	从站数组使能状态，TRUE: 表示失能该从站
szECATSlave	UINT	指定从站数组结构的内存大小
VAR_OUTPUT		
bReConfigDone	BOOL	TRUE 表示功能块执行完成

6.2.5 Modbus 通讯功能块

在本节中主要介绍 HIC 控制器 modbus 通讯部分的功能块，接下来将对每一个功能块进行介绍

6.2.5.1 Modbus_Read

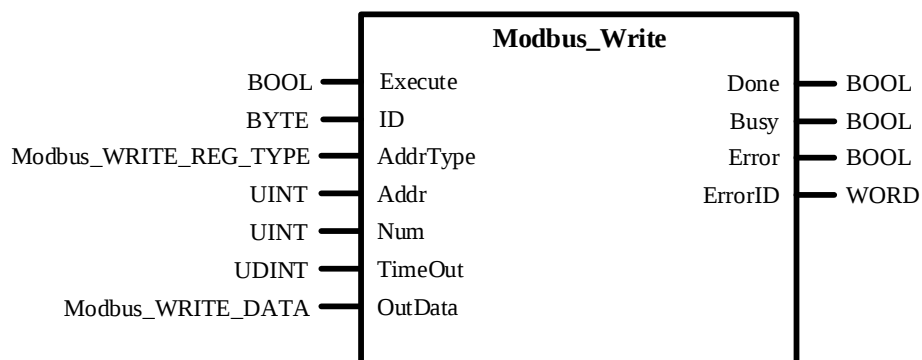
该功能块对 Modbus 从站进行数据读取。



参数	数据类型	描述
VAR_IN_OUT		
InData	Modbus_READ_DATA	Modbus 读数据内容
VAR_INPUT		
Execute	BOOL	上升沿触发
ID	BYTE	Modbus 从站的 ID 号
AddrType	Modbus_READ_REG_TYPE	寄存器类型（目前只支持 HoldingReg := 0, InputReg := 1）
Addr	UINT	从站起始地址
Num	UINT	Modbus 从站长度
TimeOut	UDINT	超时时间
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.5.2 Modbus_Write

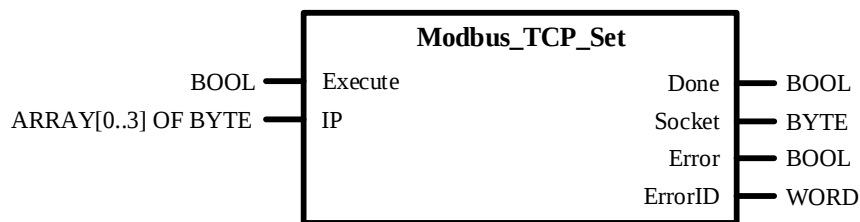
该功能块对 Modbus 从站进行数据写入。



参数	数据类型	描述
VAR_IN_OUT		
OutData	Modbus_WRITE_D ATA	Modbus 写数据内容
VAR_INPUT		
Execute	BOOL	上升沿触发
ID	BYTE	Modbus 从站的 ID 号
AddrType	Modbus_WRITE_REG_TYPE	寄存器类型（目前只支持 HoldingReg := 0）
Addr	UINT	从站起始地址
Num	UINT	Modbus 从站长度
TimeOut	UDINT	超时时间
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.5.3 Modbus_TCP_Set

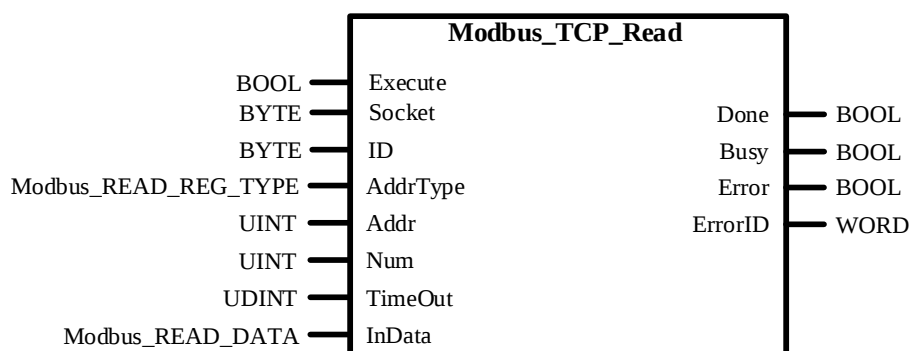
该功能块对 Modbus TCP 主站初始化。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
IP	ARRAY[0..3] OF BYTE	Modbus TCP 从站的 IP
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Socket	BYTE	Modbus TCP 主站初始化成功，获取到套接字
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.5.4 Modbus_TCP_Read

该功能块对 Modbus TCP 从站进行数据读取。

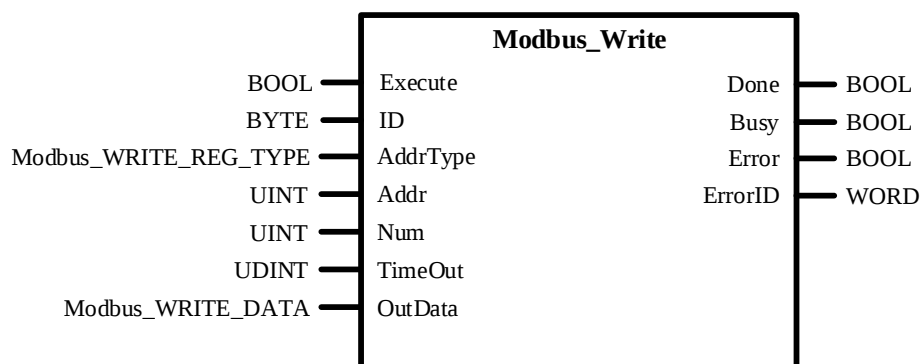


参数	数据类型	描述
VAR_IN_OUT		
InData	Modbus_READ_DATA	Modbus 读数据内容

VAR_INPUT		
Execute	BOOL	上升沿触发
Socket	BYTE	Modbus TCP 主站初始化成功，获取到的套接字
ID	BYTE	Modbus 从站的 ID 号
AddrType	Modbus_READ_REGISTER_TYPE	寄存器类型（目前只支持 HoldingReg := 0, InputReg := 1）
Addr	UINT	从站起始地址
Num	UINT	Modbus 从站长度
TimeOut	UDINT	超时时间
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.5.5 Modbus_TCP_Write

该功能块对 Modbus TCP 从站进行数据写入。

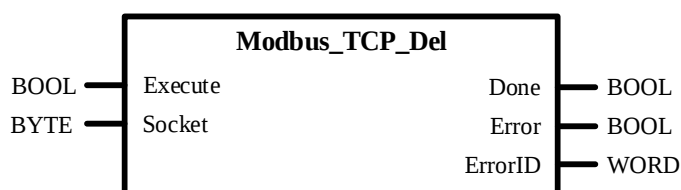


参数	数据类型	描述
VAR_IN_OUT		
OutData	Modbus_WRITE_DATA	Modbus 写数据内容
VAR_INPUT		
Execute	BOOL	上升沿触发
Socket	BYTE	Modbus TCP 主站初始化成功，获取到的套接字
ID	BYTE	Modbus 从站的 ID 号
AddrType	Modbus_WRITE_REGISTER_TYPE	寄存器类型（目前只支持 HoldingReg := 0）

	G_TYPE	
Addr	UINT	从站起始地址
Num	UINT	Modbus 从站长度
TimeOut	UDINT	超时时间
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.5.6 Modbus_TCP_Del

该功能块关闭 Modbus TCP 连接。

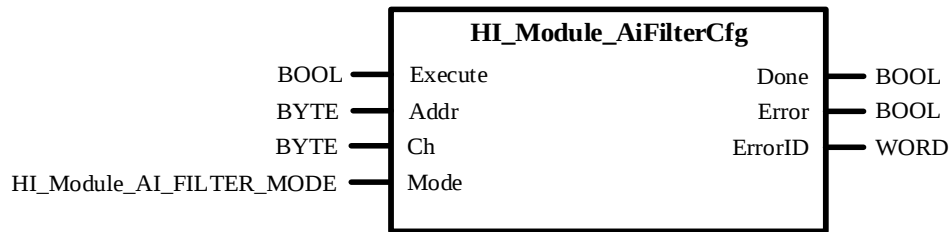


参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Socket	BYTE	Modbus TCP 主站初始化成功，获取到的套接字
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6 Module 通讯功能块

6.2.6.1 HI_Module_AiFilterCfg

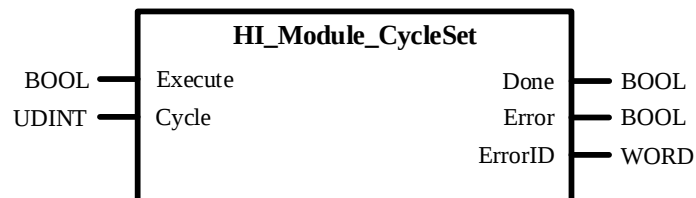
该功能块设置高速模块模拟量滤波。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Addr	BYTE	高速模块的 ID
Ch	BYTE	模拟量的通道号
Mode	HI_Module_AI_FILTER_MODE	滤波模式（Filter0 := 0,不滤波；Filter4 := 1,滤波 4 次）
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.2 HI_Module_CycleSet

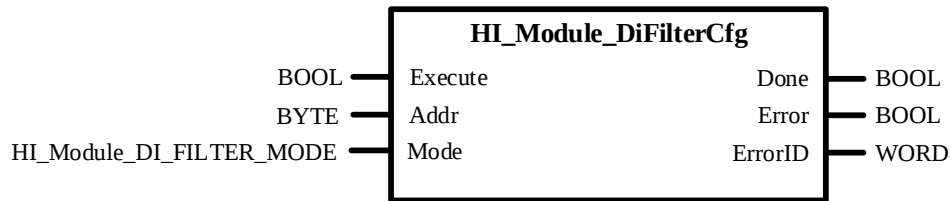
该功能块设置高速模块通讯周期时间。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Cycle	UDINT	高速模块通讯周期（250~4000，单位 us）
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.3 HI_Module_DiFilterCfg

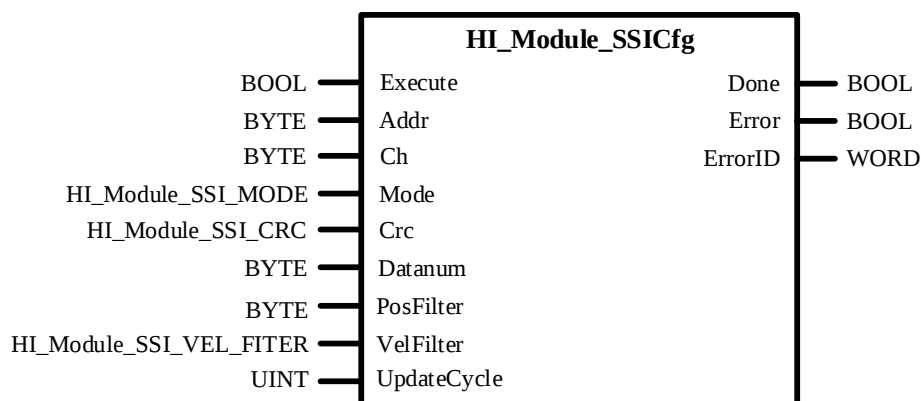
该功能块设置高速模块数字量滤波。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Addr	BYTE	高速模块的 ID
Mode	HI_Module_DI_FILTER_MODE	滤波模式（Filter0 := 0, Filter500us := 1, Filter1ms := 2, Filter2ms := 3）
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.4 HI_Module_SSICfg

该功能块设置高速模块 SSI 的相关配置。

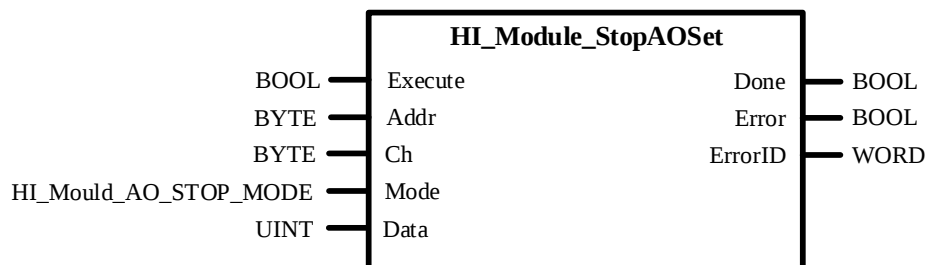


参数	数据类型	描述
VAR_INPUT		

Execute	BOOL	上升沿触发
Addr	BYTE	高速模块的 ID
Ch	BYTE	SSI 通道号
Mode	HI_Module_SSI_MO DE	设备类型（0:正交编码, 1:SSI 位置尺（格雷码）, 2:SSI 位置尺（二进制）, 3:ENDATA）
Crc	HI_Module_SSI_CR C	校验位（0:无校验,1:有校验）
Datanum	BYTE	位置尺位数：0~63
PosFilter	BYTE	位置平均次数 2 的幂次
VelFilter	HI_Module_SSI_VE L_FILTER	速度一阶级滤波系数（0: 100%; 1: 50%; 2: 25%; 3: 12.5%; 4: 6.25%; 5: 3.125%; 6: 1.5625%; 7: 0.78125%）
UpdateCycle	UINT	正交编码数据更新周期，单位：us
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.5 HI_Module_StopAOSet

该功能块设置高速模块通讯超时模拟量的输出值。

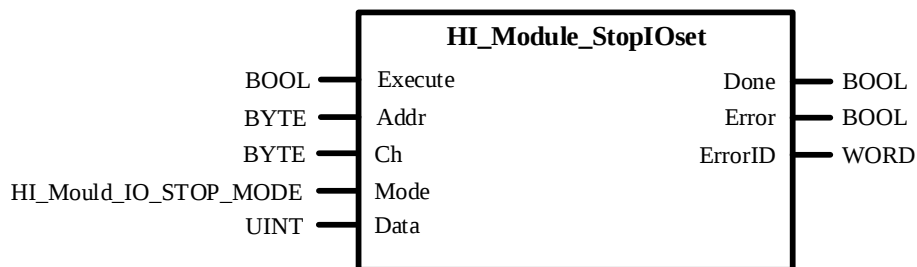


参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Addr	BYTE	高速模块的 ID
Ch	BYTE	模拟量的通道号
Mode	HI_Module_AO_ST OP_MODE	模式（KeepCurrentValue := 0, SetValveOut := 1）
Data	UINT	模拟量输出值
VAR_OUTPUT		

Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.6 HI_Module_StopIOset

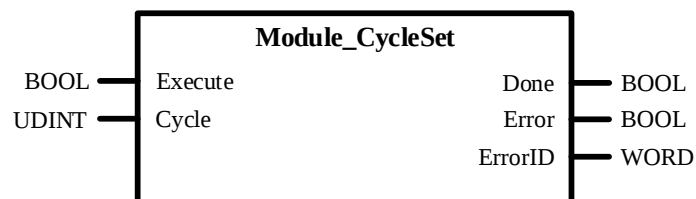
该功能块设置高速模块通讯超时数字量的输出值。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Addr	BYTE	高速模块的 ID
Ch	BYTE	数字量的通道号
Mode	HI_Module_IO_STOP_MODE	模式（KeepCurrentValue := 0,SetValveOut := 1）
Data	UINT	数字量输出值(0:FALSE;1:TRUE)
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.7 Module_CycleSet

该功能块设置低速模块通讯周期时间。

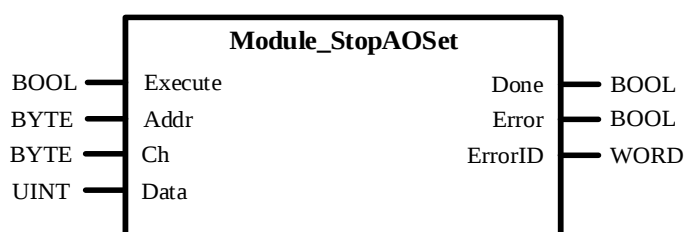


参数	数据类型	描述
----	------	----

VAR_INPUT		
Execute	BOOL	上升沿触发
Cycle	UDINT	低速模块通讯周期（1000~4000，单位 us）
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.8 Module_StopAOSet

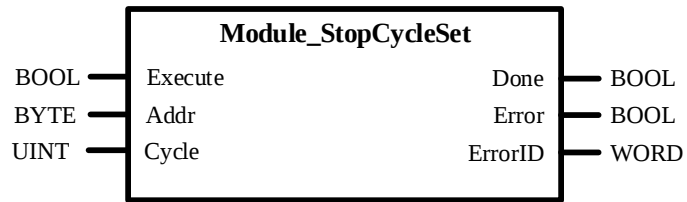
该功能块设置低速模块通讯超时模拟量的输出值。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Addr	BYTE	低速模块的 ID
Ch	BYTE	模拟量的通道号
Data	UINT	模拟量输出值
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.9 Module_StopCycleSet

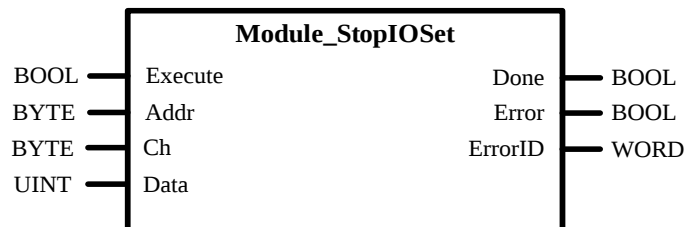
该功能块设置低速模块通讯超时时间(如果不设置，则低速模块默认通讯超时时间为 100ms)。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Addr	BYTE	低速模块的 ID
Cycle	UINT	低速模块超时时间(ms)
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.10 Module_StopIOSet

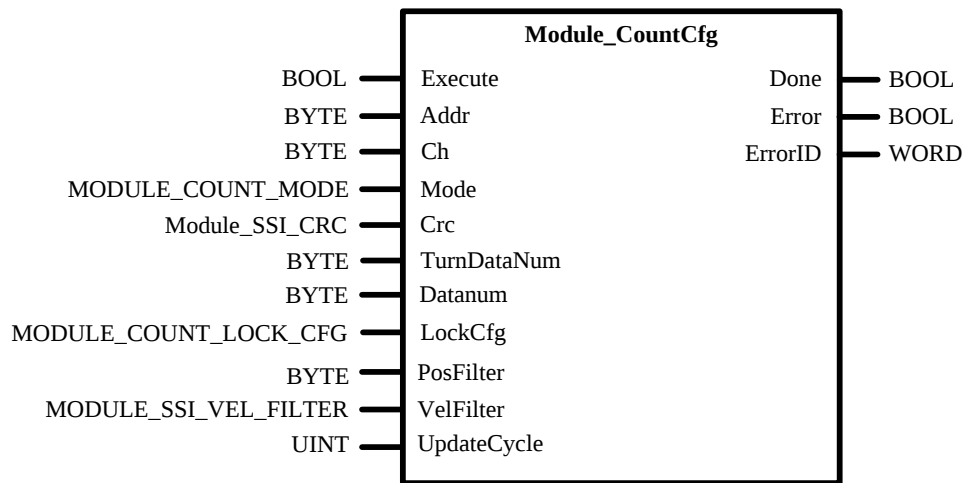
该功能块设置低速模块通讯超时数字量的输出值。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Addr	BYTE	低速模块的 ID
Ch	BYTE	数字量的通道号
Data	UINT	数字量输出值(0:FALSE;1:TRUE)
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.11 Module_CountCfg

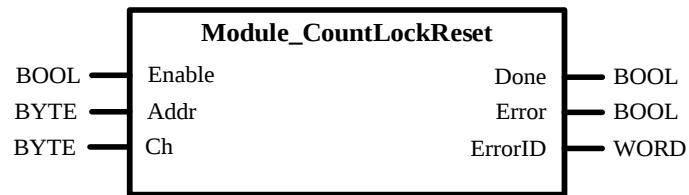
该功能块对低速模块计数进行配置。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Addr	BYTE	低速模块的 ID
Ch	BYTE	通道号
Mode	MODULE_COUNT_MODE	设备类型 (0: 正交编码; 1: SSI 位置尺 (格雷码); 2: SSI 位置尺 (二进制); 3: ENDATA; 4: BISS_C; 5: Counter)
Crc	MODULE_SSI_CRC	校验位 (0:无校验,1:有校验)
TurnDataNum	BYTE	圈数数据位数
Datanum	BYTE	数据位数: 0~63
LockCfg	MODULE_COUNT_LOCK_CFG	锁存配置 (0: RISING_1ST; 1:FALLING_1ST; 2: RISING_ALL; 3:FALLING_ALL)
PosFilter	BYTE	位置滤波使能 (0: Disable; 1: Enable)
VelFilter	MODULE_SSI_VEL_FILTER	速度滤波系数 (0: 100%; 1: 50%; 2: 25%; 3: 12.5%; 4: 6.25%; 5: 3.125%; 6: 1.5625%; 7: 0.78125%)
UpdateCycle	UINT	数据更新周期, 单位: us
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.6.12 Module_CountLockReset

该功能块对低速模块计数锁存进行复位。

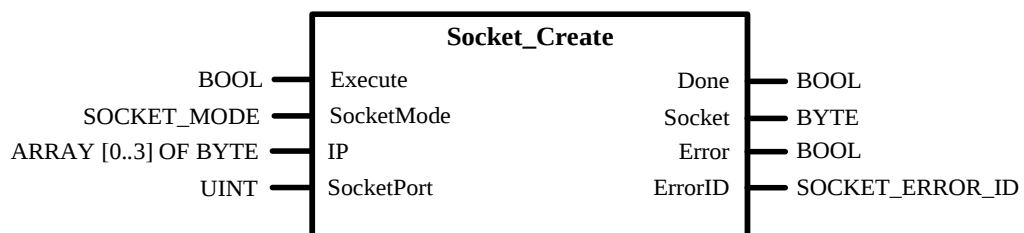


参数	数据类型	描述
VAR_INPUT		
Enable	BOOL	如果 Enable = TRUE，表示执行该功能块
Addr	BYTE	低速模块的 ID
Ch	BYTE	通道号
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.7 Socket 通讯功能块

6.2.7.1 Socket_Create

该功能块创建 Socket。

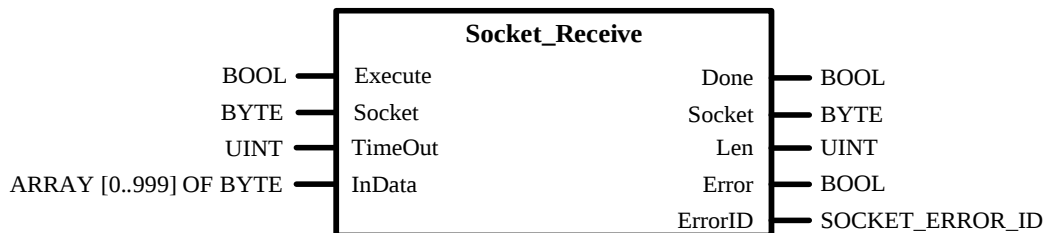


参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
SocketMode	SOCKET_MODE	模式（0: TCP Cilent; 1: TCP Server; 2: UDP）
IP	ARRAY [0..3] OF BYTE	目标 IP
SocketPort	UINT	目标端口号

VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Socket	BYTE	获取到的套接字
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	WORD	错误代码

6.2.7.2 Socket_Receive

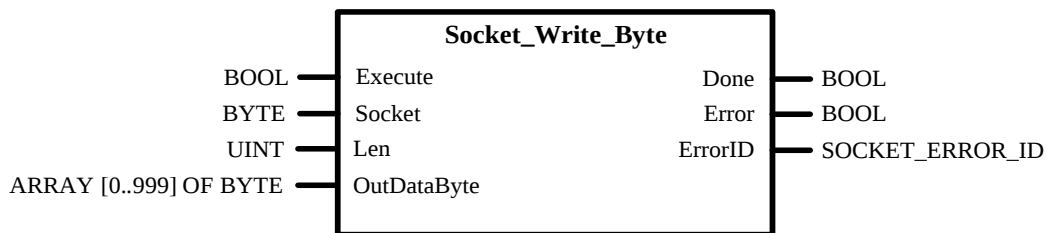
该功能块接收 Socket 数据。



参数	数据类型	描述
VAR_IN_OUT		
InData	ARRAY [0..999] OF BYTE	Socket 读取的数据信息
VAR_INPUT		
Execute	BOOL	上升沿触发
Socket	BYTE	获取到的套接字
TimeOut	UINT	超时时间
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Busy	BOOL	TRUE 表示功能块的执行还没有结束
Len	UINT	长度
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	SOCKET_ERROR_ID	错误代码

6.2.7.3 Socket_Write_Byte

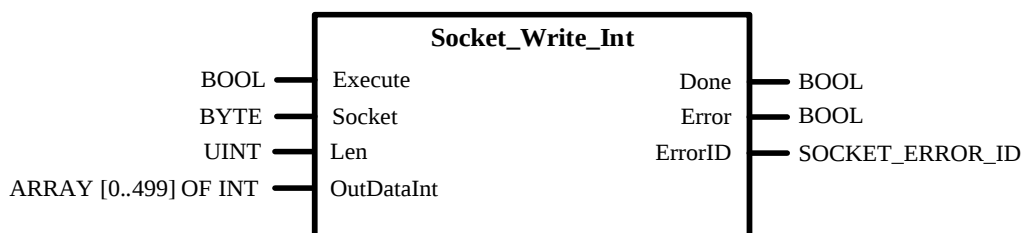
该功能块对 Socket 进行 Byte 类型数据写入。



参数	数据类型	描述
VAR_IN_OUT		
OutDataByte	ARRAY [0..999] OF BYTE	Socket 写入的数 Byte 类型据信息
VAR_INPUT		
Execute	BOOL	上升沿触发
Socket	BYTE	获取到的套接字
Len	UINT	长度，Byte 类型
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	SOCKET_ERROR_ID	错误代码

6.2.7.4 Socket_Write_Int

该功能块对 Socket 进行 Int 类型数据写入。

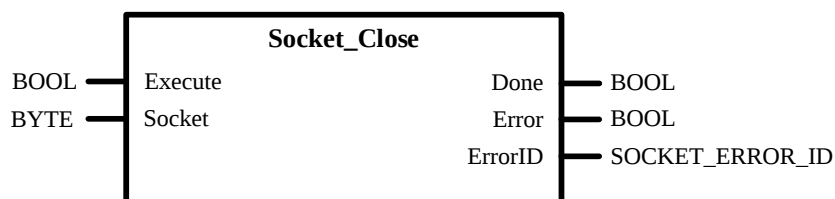


参数	数据类型	描述
VAR_IN_OUT		
OutDataInt	ARRAY [0..499] OF INT	Socket 写入的数 Int 类型据信息
VAR_INPUT		
Execute	BOOL	上升沿触发
Socket	BYTE	获取到的套接字

Len	UINT	长度, Int 类型
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	SOCKET_ERROR_ID	错误代码

6.2.7.5 Socket_Close

该功能块关闭 Socket 连接。



参数	数据类型	描述
VAR_INPUT		
Execute	BOOL	上升沿触发
Socket	BYTE	获取到的套接字
VAR_OUTPUT		
Done	BOOL	TRUE 表示功能块执行完成
Error	BOOL	TRUE 表示执行功能块时发生了一个错误
ErrorID	SOCKET_ERROR_ID	错误代码

6.2.8 通讯功能块错误说明

当调用 COM_LIB 功能块时, 有时会出现报错, 具体的报错信息如下表所示。

ErrorID	详细说明	解决方法
500	无效的模式配置	检查功能块模式配置
501	无效的模式配置	检查功能块模式配置
502	无效的模式配置	检查功能块模式配置
503	无效的模式配置	检查功能块模式配置
504	无效的模式配置	检查功能块模式配置
1000	Ecat 从站个数配置超过 32 个	请修改从站配置
1001	Ecat 液压从站个数超过 8 个	请修改从站配置

1002	Ecat 液压从站类型配置错误	请修改从站配置
1003	配置 IO 数量超限	请修改从站配置
1004	配置伺服个数超 16 个	请修改从站配置
1005	从站类型配置错误	请修改从站配置
1006	非 ECAT 版本不能配置伺服从站	请修改从站配置
1007	Ecat reset error	请修改从站配置
1050	CAN 从站个数配置超过 16 个	请修改从站配置
1051	CAN 液压从站个数超过 8 个	请修改 CAN 配置
1052	CAN 液压从站类型配置错误	请修改 CAN 配置
1053	配置 IO 数量超限	请修改 CAN 配置
1054	配置伺服个数超 16 个	请修改 CAN 配置
1055	从站类型配置错误	请修改 CAN 配置请修改 CAN 配置
1056	非 CAN 版本不能配置伺服从站	请修改 CAN 配置
1057	CAN 波特率错误	请修改 CAN 配置
1058	CAN 周期错误	请修改 CAN 配置
1100	PLC 初始化失败	
1101	PLC 读参数错误	
1102	PLC 轴参数 CRC 校验错误	
1103	PLC 读掉电保持错误	
1104	PLC 掉电保持数据 CRC 错误	
1105	PLC 掉电保持数据和工程 CRC 均错误	
1106	PLC 写掉电时数据写一半错误	
1107	PLC 写掉电时数据写一半和工程 CRC 错误	
1108	PLC 工程 CRC 错误	
1200	无效的 socket	
1201	socket 创建失败	
1202	socket 监听失败	
1203	socket 接收失败	
1204	socket 发送失败	
1205	socket 接收失败	
1206	socket 关闭失败	
1207	socket 连接失败	
1208	socket 发送超时	
1209	socket 接收超时	
1210	数组错误	
1211	socket 绑定失败	

1212	创建 accept 线程失败	
1213	创建 connect 线程失败	
1214	超过最大连接数	
1215	accept 处于错误状态	
1216	无效的 IP 地址	
1217	connect 处于错误状态	
1218	还未建立连接	
1219	还未建立 accept	
1220	写数据长度错误	
1221	TCP 操作无效的状态条件	
1222	TCP_Create 一次都没有调用过	
1223	TCP_Server_Create 一次都没有调用过	
1300	无效的波特率	
1301	串口错误状态	
1302	串口读取超时错误	
1303	无效的 BUFF	
1304	无效的串口状态	
1305	串口发送超时错误	
1306	串口设置不为自由通讯模式	
1308	串口只能设置一次	
1309	串口写长度错误	
1400	无效的 CAN BUFF	检查 CAN 读写功能块的数据数组
1401	无效的 CAN BUFF 长度	检查 CAN 读写功能块的数据数组长度
1402	第 1 通道 can 读错误	检查第 1 通道 can 接收设置
1403	第 2 通道 can 读错误	检查第 2 通道 can 接收设置
1404	第 3 通道 can 读错误	检查第 3 通道 can 接收设置
1405	第 4 通道 can 读错误	检查第 4 通道 can 接收设置
1406	第 1/2 通道 can 读错误	检查第 1/2 通道 can 接收设置
1407	第 1/3 通道 can 读错误	检查第 1/3 通道 can 接收设置
1408	第 1/4 通道 can 读错误	检查第 1/4 通道 can 接收设置
1409	第 2/3 通道 can 读错误	检查第 2/3 通道 can 接收设置
1410	第 2/4 通道 can 读错误	检查第 2/4 通道 can 接收设置
1411	第 3/4 通道 can 读错误	检查第 3/4 通道 can 接收设置
1412	第 1/2/3 通道 can 读错误	检查第 1/2/3 通道 can 接收设置
1413	第 1/2/4 通道 can 读错误	检查第 1/2/4 通道 can 接收设置
1414	第 1/3/4 通道 can 读错误	检查第 1/3/4 通道 can 接收设置

1415	第 2/3/4 通道 can 读错误	检查第 2/3/4 通道 can 接收设置
1416	第 1/2/3/4 通道 can 读错误	检查第 1/2/3/4 通道 can 接收设置
1417	can 发送错误	检查 CAN 数据收发状态与设置
1418	can 发送 id 错误	检查 CAN 数据收发状态与设置
1419	can 发送错误和 ID 错误	检查 CAN 数据收发状态与设置
1500	MODBUS 不是工作在主站模式	检查串口模式设置
1501	MODBUS 主站初始化错误	请重启控制器
1502	MODBUS 主站使能错误	请重启控制器
1503	MODBUS 主站命令错误	检查 Modbus 数据收发状态与设置
1504	MODBUS 主站没有使能	检查 Modbus 数据收发状态与设置
1505	MODBUS 主站处于 BUSY 状态	请重启控制器
1507	MODBUS 从站答复超时	检查 Modbus 数据收发状态与设置
1508	MODBUS 接收数据出错	检查 Modbus 数据收发状态与设置
1509	功能块使能失败	检查 Modbus 数据收发状态与设置
1510	功能块返回错误	检查 Modbus 数据收发状态与设置
1511	MODBUS 写长度错误	检查 MODBUS_RUN 功能块 Count 值
1512	MODBUS 主站只能设置一次	重启控制器重新设置
1520	MODBUS 功能块参数配置错误	检查功能块参数配置
1522	MODBUS 功能块寄存器配置错误	检查功能块参数配置
1523	MODBUS 当前状态错误	检查上次读写是否完成
1524	MODBUS 读功能块数据错误	检查 Modbus 通讯情况
1526	MODBUS 当前状态错误	检查上次读写是否完成
10002	ETHERCAT 总线在错误状态	检查 ETHERCAT 总线通讯情况
10003	ETHERCAT 读数组错误	检查 ETHERCAT 总线通讯情况
10004	ETHERCAT 写数组错误	检查 ETHERCAT 总线通讯情况
10005	ETHERCAT ID 错误	检查功能块配置的 ID 号
10006	ETHERCAT 状态未处于更新中状态	检查 EtherCAT 状态更新功能块
10007	ETHERCAT 通用设备数组信息配置错误	检查 EtherCAT 通用设备配置信息
10008	ETHERCAT 通用设备 PDO 数量配置错误	检查 EtherCAT 通用设备配置信息
10009	ETHERCAT 更新功能块状态错误	检查 EtherCAT 状态更新功能块
10010	ETHERCAT 通用设备配置添加失败	检查功能块配置信息
10011	ETHERCAT 通用设备配置获取失败	检查功能块配置信息
10012	ETHERCAT 报文内容更新失败	检查 EtherCAT 通讯状态
10013	ETHERCAT 报文长度更新失败	检查 EtherCAT 通讯状态
10014	ETHERCAT 通用设备个数超限	检查 EtherCAT 通用设备个数是否超 16 个
10103	读数组错误	检查通用设备 PDO 读数组大小是否 8 字节

10104	写数组错误	检查通用设备 PDO 写数组大小是否 8 字节
10106	CANopen 通用设备配置数组长度错误	检查 CANopen 通用设备配置
10107	CANopen 通用设备配置数组调用数错误	检查 CANopen 通用设备配置
10109	CANopen 通用设备配置序号错误	检查 CANopen 通用设备配置
10110	获取 CANopen 通用设备配置信息失败	检查 CANopen 通用设备配置
10111	ID 错误	ID 不在规定范围内
10112	CANopen 设备类型获取失败	检查 CANopen 设备类型的配置
10113	CANopen 设备配置信息获取失败	检查 CANopen 设备信息的配置
10114	CANopen 总线复位失败	请联系厂商处理
10115	SDO 读写长度错误	检查 SDO 读写配置长度
10116	SDO 读写应答失败	检查 CAN 总线通讯情况
10119	CANopen 的 SDO 状态错误	检查 CAN 总线 SDO 调用情况
10120	CANopen 的 SDO 读错误	检查 CAN 总线通讯情况
10121	CANopen 的 SDO 写错误	检查 CAN 总线通讯情况
10122	CANopen 设备类型错误	检查读写的设备类型
10123	CANopen 设备类型配置超限	检查设备配置情况
10124	CANopen 的 SDO 读写超时	检查 CAN 总线通讯情况