COTRUST

CTH300-C 系列运动控制器

用户手册

版本：V1.80
发布日期：07/2020

权利声明

版权声明

深圳市合信自动化技术有限公司版权所有，并保留对本手册及本声明的最终解释权和修改权。未得到深圳市合信自动化技术有限公司的书面许可，任何单位及个人不得以任何方式或形式对本手册内的任何部分进行复制、摘录、备份、修改、传播、汇编、翻译成其它语言、将其全部或部分用于商业用途。

商标声明

®、TrustPLC®、MagicWorks®、COTRUST®均为深圳市合信自动化技术有限公司或其关联公司的注册商标，受法律保护，侵权必究。未经深圳市合信自动化技术有限公司或其关联公司的书面许可，任何单位及个人不得以任何方式或理由对该商标的任何部分进行使用、复制、永久下载、修改、传播、抄录或与其他产品捆绑销售，或注册为其域名或无线网址名称，或域名或无线网址的主要部分。

免责声明

本手册依据现有信息制作，其内容如有更改，恕不另行通知。深圳市合信自动化技术有限公司在编写该手册的时候已尽最大努力保证其内容准确可靠，但深圳市合信自动化技术有限公司不对本手册中的遗漏、不准确或印刷错误导致的损失和损害承担责任。

前 言

感谢您购买深圳市合信自动化技术有限公司的 CTH300-C 系列运动控制器产品！

在使用我公司 CTH300-C 系列运动控制器前，敬请您仔细阅读本手册，以便更清楚地掌握产品的特性，更安全地应用，充分利用本产品丰富的功能。

产品特性

CTH300-C 系列运动控制器是一款基于嵌入式系统的中央处理单元，它同时具备运动控制和逻辑控制功能。作为运动控制平台 CoMotion 的重要一员，CTH300-C 系列运动控制器及其扩展单元，通过处理各种运算和运行用户程序，对外设进行访问和控制以实现用户所需的机器动作。

适用对象

本手册提供关于 CTH300-C 系列运动控制器的安装和调试信息，为工程师、安装人员、维护人员和具有自动化常识的电工而设计。

服务支持

深圳市合信自动化技术有限公司建立了售后维修服务中心，并提供电话热线服务。客户在产品使用过程中遇到问题时，可随时与深圳市合信自动化技术有限公司各地的服务支持联系。各地服务支持热线电话请到 http://www.co-trust.com/Company/Contact/index.html 获取。此外，客户还可通过深圳市合信自动化技术有限公司网站 http://www.co-trust.com 或官方微信及时了解最新产品动态，以及下载需要的技术文档。

安全注意事项

在开始使用 CTH300-C 系列产品之前，请认真阅读用户手册的注意事项，以避免意外事故的发生。

所负责产品安装、操作的人员必须经过严格培训，遵守相关行业的安全规范，严格遵守该手册提供的相关设备注意事项和特殊安全指示，按正确的操作方法进行设备的各项操作。

为防止对人的危害和对财产的损害，对务必遵守的事项特做以下说明。对错误使用本产品而可能带来的危害和损害程度见相关符号说明。



警告

该标记表示

“由于没有按要求操作造成的危险，可能导致人身伤亡”



注意

该标记表示

“由于没有按要求操作造成的危险，可能会导致人身轻度或中度伤害和设备损坏”



提示

该标记表示

“对操作的描述进行必要的补充或说明”

注：必要时，可做针对性详细说明，细分为产品使用中、安装时、布线时、运行和保养时以及报废时的注意事项。

版本修订记录

发行日期	修订后版本	修订内容
2020 年 7 月	V1.80	• 新增 C37-02 产品规格

目录

权利声明.....II
前 言.....III
安全注意事项.....IV
版本修订记录.....V
1 系统概述.....1
1.1 CTH300-C 系列运动控制器.....1
1.2 CTH300-C 系列运动控制器的扩展单元.....2
1.3 网络架构.....4
1.4 功能应用.....4
1.4.1 运动控制功能.....4
1.4.2 逻辑控制功能.....5
2 使用入门.....6
2.1 连接 C37-01.....6
2.2 启动 CODESYS 软件.....7
2.3 设置通信.....12
2.4 编写 PLC 程序.....13
2.5 编译并运行程序.....14
2.6 监控与调试.....15
3 运动控制器的基本概念.....17
3.1 CTH300-C 系列运动控制器如何执行您的控制逻辑.....17
3.2 CTH300-C 系列运动控制器的数据存储区.....18
3.3 CTH300-C 系列运动控制器的特性.....19
3.3.1 通信功能.....19
3.3.2 运动控制功能.....20
3.3.3 逻辑控制功能.....21
3.3.4 实时时钟.....21
4 CTH300-C 系列运动控制器的技术规格.....22
4.1 技术参数.....22
4.2 接口定义.....24
5 安装.....26
5.1 安装注意事项.....26
5.2 安装尺寸.....27
5.3 使用机架.....28
5.4 安装方法.....30
5.5 接地和布线.....32
5.6 抑制电路.....32

6 CODESYS 的结构.....34
6.1 工程的构成.....34
6.2 语言.....36
6.2.1 指令表（IL）.....36
6.2.2 结构化文本（ST）.....38
6.2.3 顺序功能图表（SFC）.....43
6.2.4 功能模块图（FBD）.....46
6.2.5 连续功能图表（CFC）.....46
6.2.6 梯形图（LD）.....46
6.3 联机调试功能.....47
6.4 标准化.....48
7 使用指南.....49
7.1 总线通信实例.....49
7.1.1 示例组件.....49
7.1.2 网络连接.....49
7.1.3 操作步骤.....49
7.2 Modbus TCP 通信实例.....51
7.2.1 示例组件.....51
7.2.2 网络连接.....51
7.2.3 操作步骤.....52
7.3 Modbus RTU 通信实例.....53
7.3.1 示例组件.....53
7.3.2 网络连接.....54
7.3.3 操作步骤.....54
7.4 CANopen 通信实例.....55
7.4.1 示例组件.....56
7.4.2 网络连接.....56
7.4.3 操作步骤.....56
7.5 基于 CANopen 通信的单轴运动控制.....58
7.5.1 示例组件.....58
7.5.2 网络连接.....59
7.5.3 操作步骤.....59
7.6 EtherCAT 通信实例.....63
7.6.1 示例组件.....63
7.6.2 网络连接.....63
7.6.3 CPU 通信设置.....64
7.6.4 操作步骤.....65
7.7 基于 EtherCAT 通信的电子凸轮.....70
7.7.1 示例组件.....70
7.7.2 网络连接.....71
7.7.3 操作步骤.....71
7.7.4 跟踪（Trace）.....76
7.8 CNC 例程.....77
7.9 高速计数器模块使用举例.....77

7.9.1 示例组件.....77
7.9.2 网络连接.....78
7.9.3 操作步骤.....78

附录.....81

A 术语解释.....81
B 电源预算.....82
C 扩展模块技术规范.....83
C.1 电源模块.....83
C.2 数字量模块.....85
C.3 模拟量模块.....89
C.4 温度模块.....95
C.5 高速计数模块.....101
C.6 脉冲输出模块.....107
C.7 CAN-1T 主站模块（C3X-00 系列支持）.....108
C.8 中继模块.....110
C.9 EtherCAT 从站模块.....113
D IO 模块通道配置控制字意义.....115
D.1 数字量输入模块通道配置.....115
D.2 数字量输出模块通道配置.....115
D.3 AI 模块通道配置.....116
D.4 TC 模块通道配置.....116
D.5 RTD 模块通道配置.....117
D.6 模拟量输出模块通道配置.....119
E CT_MODBUS 库的使用介绍.....120
E.1 安装 CTModbus 库文件.....120
E.2 Modbus_RTU 库文件.....121
E.3 Modbus_TCP 库文件.....125
F 在 CODESYS 中进行的相关操作详述.....128
F.1 添加设备.....128
F.2 配置 CANopen 从站设备.....130
F.3 配置 EtherCAT 从站设备.....131
G 在 CODESYS 中安装设备描述文件.....136
G.1 安装 EtherCAT 伺服从站设备描述文件.....136
G.2 安装 PLC/402 轴设备描述文件.....137
G.3 安装 CANopen 从站 EDS 文件.....139
H 指令速查.....139
H.1 CODESYS 中的操作符总结.....140
H.2 Standard.lib 库.....142
H.3 Util.lib 库.....142
H.4 SoftMotion 库.....143
I 订货信息.....152

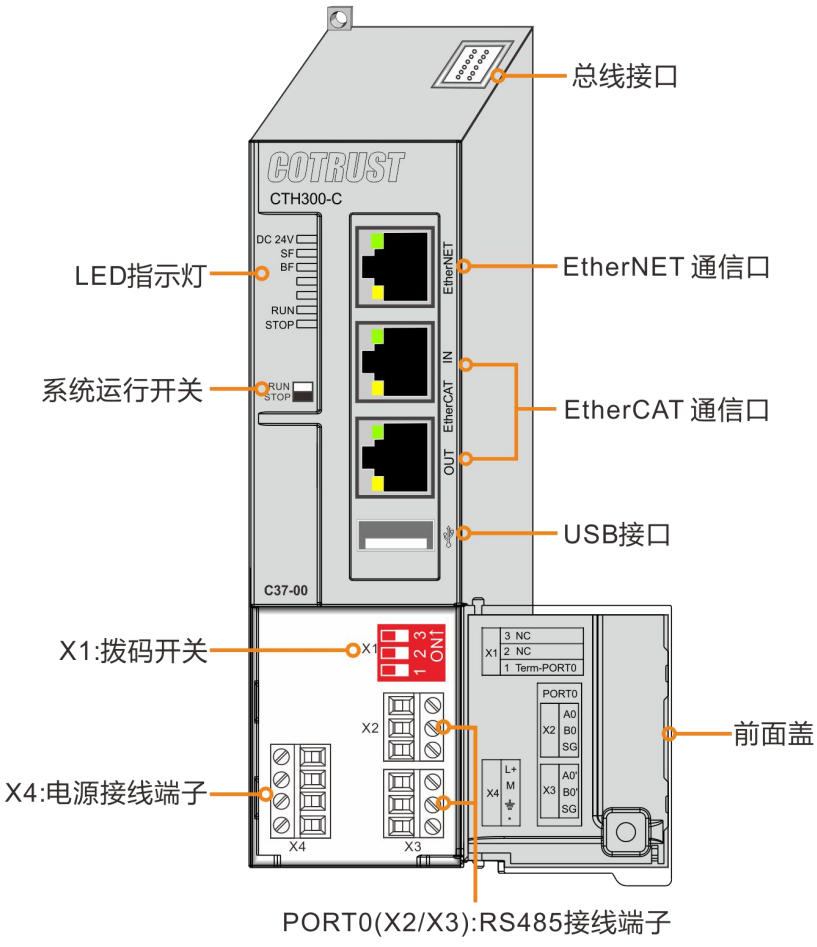
1 系统概述

CTH300-C 系列运动控制器是一款基于嵌入式系统的中央处理单元，它同时具备运动控制和逻辑控制功能。作为运动控制平台 CoMotion 的重要一员，CTH300-C 系列运动控制器及其扩展单元，通过各种运算的处理以及用户程序的运行，对外设进行访问和控制以实现用户所需的机器动作。

1.1 CTH300-C 系列运动控制器

CTH300-C 系列 C3X-00 运动控制器自带 3 个以太网通信口，C3X-01/C37-02 运动控制器自带 2 个以太网通信口和 1 个 CAN 通讯口。C3X-00 和 C3X-01、C37-02 运动控制器均提供 1 个可串联的 RS485 通信口与第三方设备进行通讯。

CTH300-C 系列 C3X-00 外观示意图如下：



CTH300-C 系列 C3X-01/C37-02 外观示意图如下：

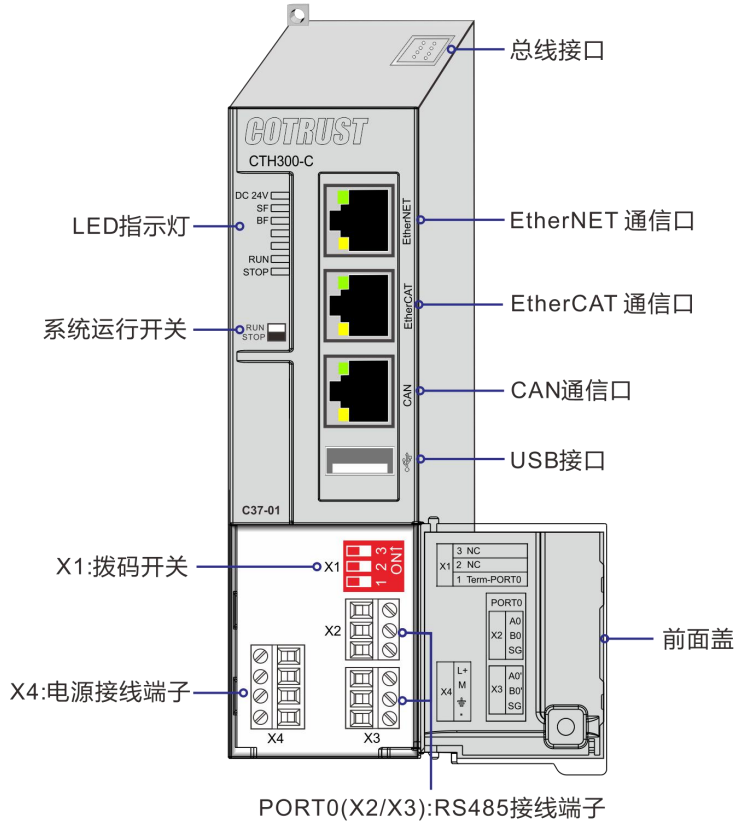


表 1-1　CTH300-C 运动控制器的基本规格

特性	C35-00	C36-00	C37-00	C35-01	C36-01	C37-01	C37-02
外形尺寸（W*H*D）	34×115×101.6 mm						
用户程序空间	32MB						
掉电保持空间	32KB			64KB			120KB
通讯口	3*RJ45（10/100 Mbit/s）、2*RS485、1*USB2.0						
系统时钟	掉电保持						
编程软件	CODESYS V3						
SoftMotion 功能	--		单轴/主从轴控制功能		单轴/主从轴控制功能/CNC 功能		
编程语言	IL、SFC、CFC、FBD、LD、ST						
位指令执行速度	0.015μs						

<备注> 有关 CTH300-C 运动控制器的具体特性描述请参考章节“3.3 CTH300-C 运动控制器的特性”。

1.2　CTH300-C 系列运动控制器的扩展单元

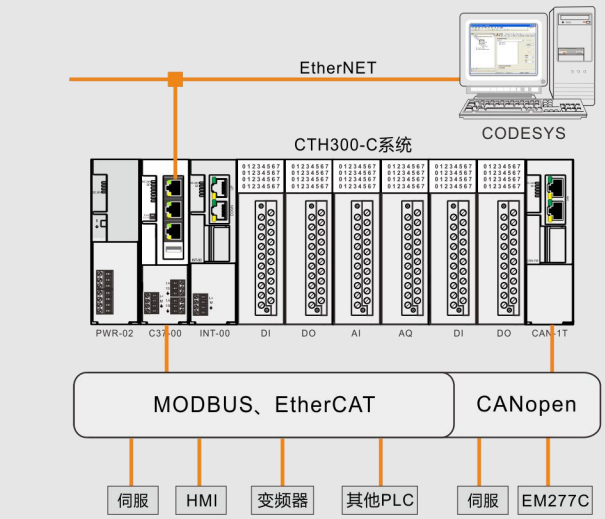
CTH300-C 系列运动控制器属于 CTH300 系列中型 PLC 家族，它可以支持 CTH300 系列的扩展模块，其模块类型包括但不仅限于：数字量输入输出、模拟量输入输出、温度采集、高速计数、运动控制、CAN 通信等，模块型号请参考下表：

	数字量输入模块 CTH3 DIT-080S1 CTH3 DIT-160S1 CTH3 DIT-320S1	数字量输出模块 CTH3 DQT-080S1 CTH3 DQT-160S1 CTH3 DQT-320S1 CTH3 DQR-080S1 CTH3 DQR-160S1	
	模拟量输入模块 CTH3 AIS-040S1 CTH3 AIV-080S1 CTH3 AIC-080S1	模拟量输出模块 CTH3 AQS-040S1 CTH3 AQS-080S1 模拟量输入输出模块 CTH3 AMS-060S1	温度模块 CTH3 AIT-040S1 CTH3 AIT-080S1 CTH3 AIR-040S1 CTH3 AIR-080S1
	高速计数模块 CTH3 HSC-020S1	 脉冲输出模块 CTH3 HSP-040S1	 CAN 主站模块 CTH3 CAN-1T0S1
	电源模块 CTH3 PWR-020S1	 中继模块 CTH3 INT-000S1	 EtherCAT 从站模块 CTH3 ECT-000S1

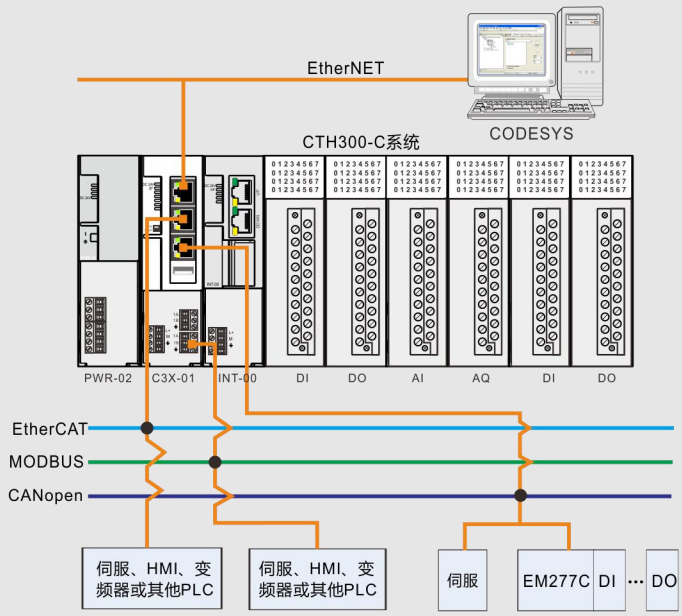
<备注> 请参考附录“C 扩展模块技术规范”获取扩展模块的相关规范。

1.3 网络架构

C3X-00 网络架构如下：



C3X-01/C37-02 网络架构如下：



编程

通过 CTH300-C 系列运动控制器的 EtherNET 接口与上位机进行通信。

CTH300-C 系统扩展

1、INT-00 扩展

最大允许扩展 32 个 I/O 模块(4 机架，每机架 8 个模块)

2、ECT-00 扩展

最大允许扩展 1032 个 I/O 模块（125 个 ECT，每个 ECT 可扩展 8 个模块）

通信

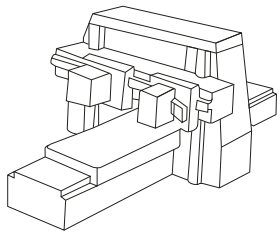
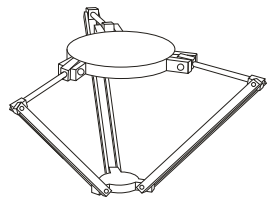
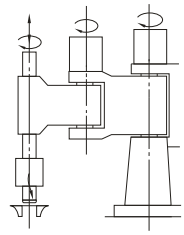
通过 CTH300-C 系列运动控制器的 EtherNET、RS485、EtherCAT、CAN 通信口（仅限 C3X-01 和 C37-02 系列 CPU）与伺服、HMI、其他 PLC 等进行通信；通过总线扩展 CAN-1T 主站模块（仅限 C3X-00 系列 CPU）实现 CANopen 通信。

1.4 功能应用

CTH300-C 系列运动控制器拥有两种控制功能：运动控制功能、逻辑控制功能。

1.4.1 运动控制功能

表 1-2　运动控制功能

基于 PLCopen 的单轴控制/主从轴控制			
单轴	回原、相对/绝对定位、位置/速度/加速度轨迹规划、连续运动、JOG		
主从轴	电子齿轮、电子凸轮、相移		
CNC 功能（仅限 C37-00、C37-01、C37-02）			
标准 CNC 功能	3-5 轴联动、GCODE 导入、坐标变换等		
面向应用的 CNC 功能	Gantry（龙门式工作台）	Parallel（并联机器人）	Scara（选择顺应性装备机器手臂）
			

1.4.2 逻辑控制功能

三种程序组织单元

- ❒ 程序（Program）
- ❒ 功能块（Function Block）
- ❒ 功能（Function）

六种编程语言

- ❒ 指令表（IL）
- ❒ 结构化文本（ST）
- ❒ 顺序功能图表（SFC）
- ❒ 功能模块图（FBD）
- ❒ 梯形图（LD）
- ❒ 连续功能图表（CFC）

<备注> 请参考章节“6.2 语言”获取编程语言的说明。

标准化的 PLC 功能

类型转换、数值功能、算术功能、移位功能、布尔运算功能、选择功能、比较功能、字符串功能、定时器、计数器、边沿检测、双稳态元素

2 使用入门

本章基于具体实例，介绍如何创建一个包含 PLC 程序的简单工程，以及如何加载这个程序到目标设备（C37-01 运动控制器），运行并监控此程序。

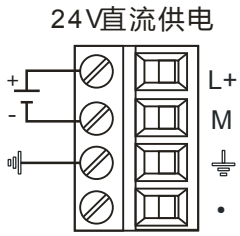
示例程序使用 LD 语言编写，包含一段程序：PLC_PRG。PLC_PRG 中包含两个定时器，其中定时器 1 在程序启动时开始计时，到达 5 秒后进行置位；定时器 2 在定时器 1 置位后开始计时，达到 5S 后定时器 2 开始复位；之后定时器 1 和定时器 2 在 5 秒间隔中进行周而复始的置位和复位操作。

2.1 连接 C37-01

只需给 C37-01 供电，然后在编程设备与 C37-01 之间连接通信电缆即可。

给 C37-01 供电

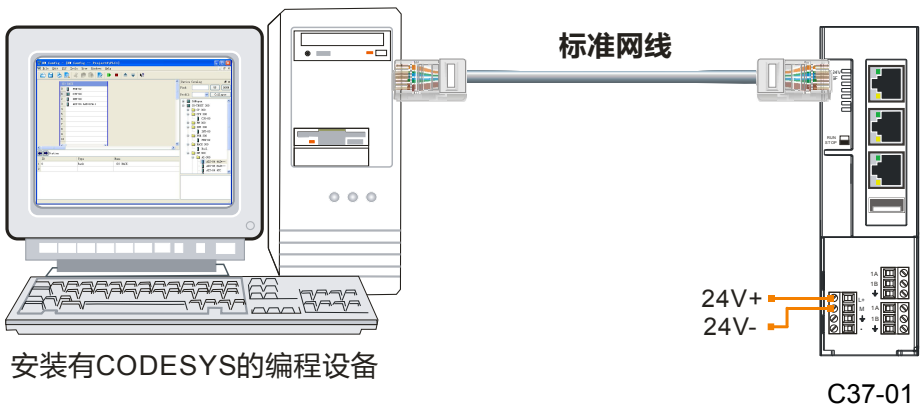
下图为 C37-01 的直流供电接线示意图：



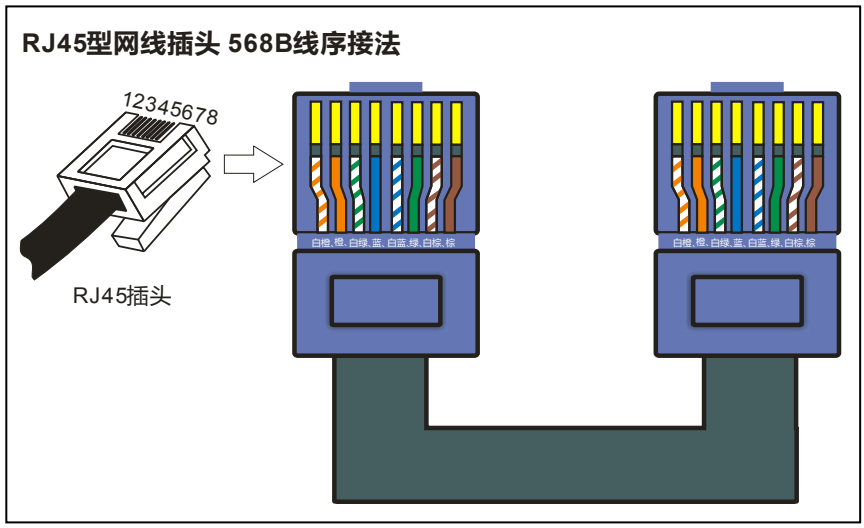
<警告> 在安装或拆除 CTH300-C 系列运动控制器之前，必须遵守相应的安全防护规范，并务必将其电源断开。

连接电缆

使用标准网线连接编程设备 PG/PC 与 C37-01（EtherNET 通信口）。



参考如下示意图制作标准网线：



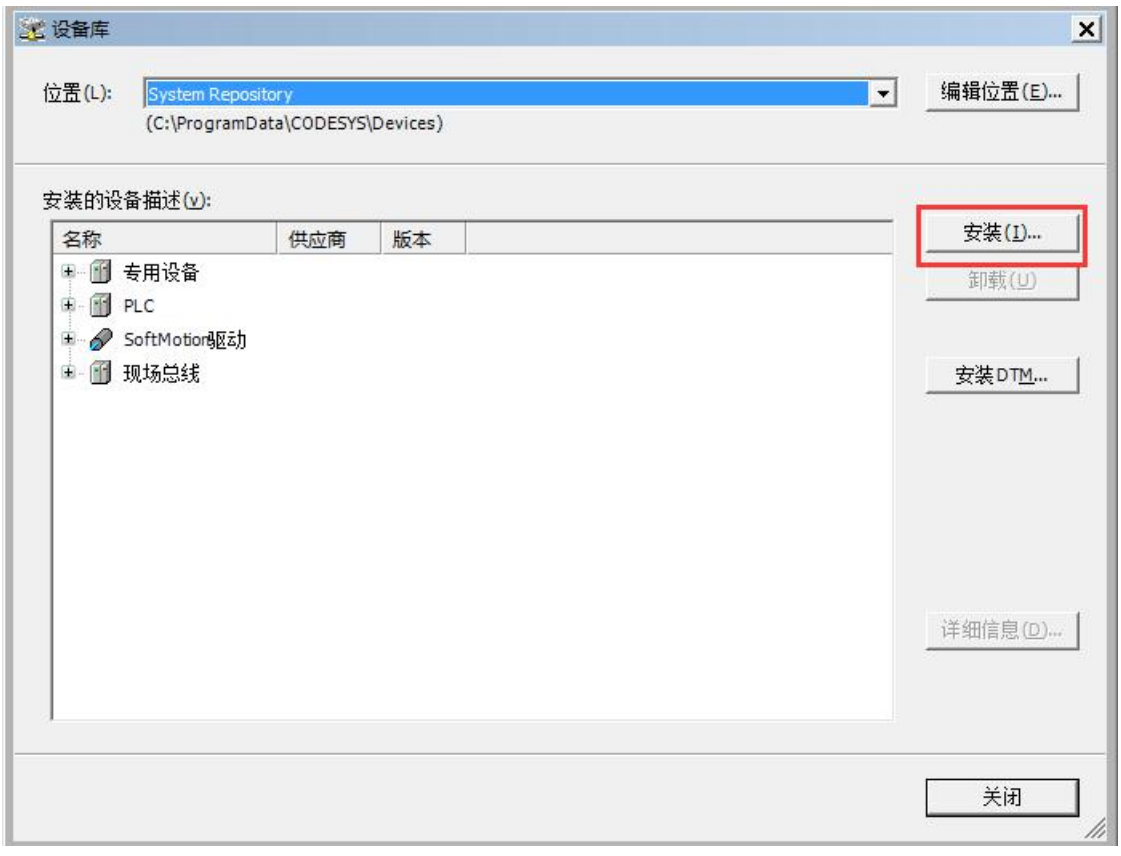
2.2 启动 CODESYS 软件

双击桌面图标 或在开始菜单中启动 CODESYS 软件，然后执行以下步骤：

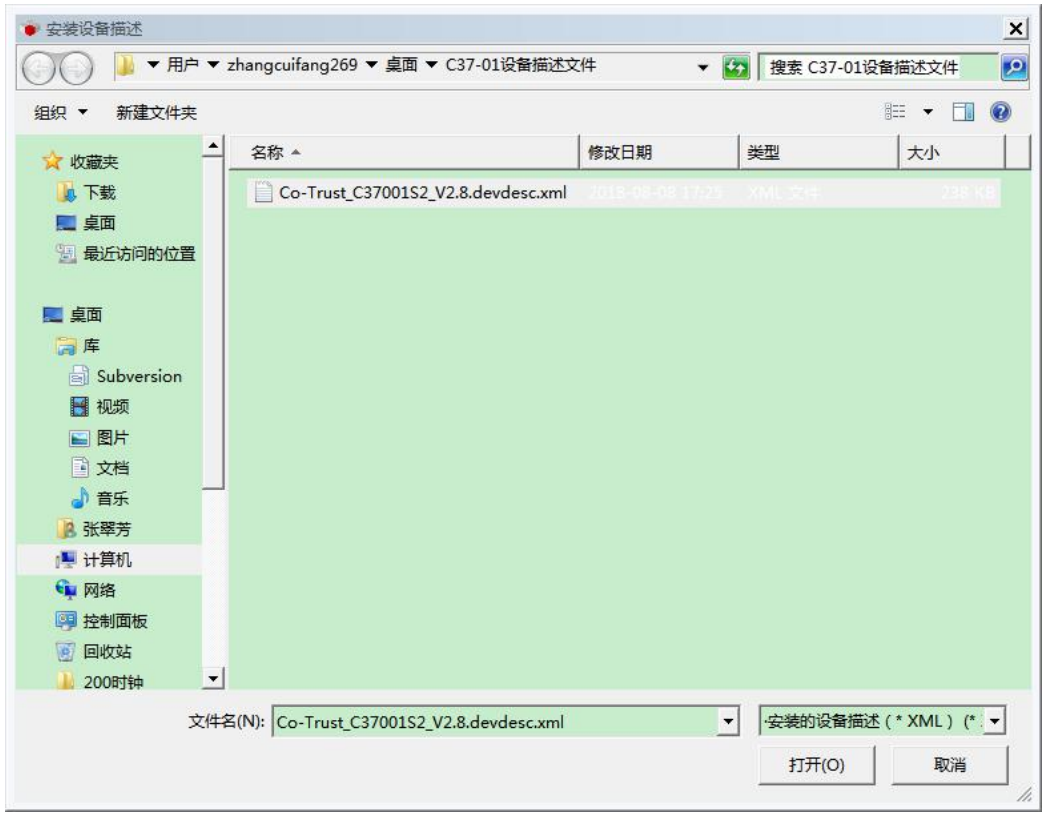
1、安装 COTRUST 设备描述文件（Co-Trust_C37001S2_V2.8.devdesc.xml）

首先需要安装 COTRUST 设备描述配置文件（Co-Trust_C37001S2_V2.8.devdesc.xml），安装成功后即可在程序系统中使用 COTRUST 设备，具体操作如下：

❒ 选择菜单项“工具”→“设备库”打开如下对话框，其中列出了系统中的设备描述，在该对话框里中点击“安装”。

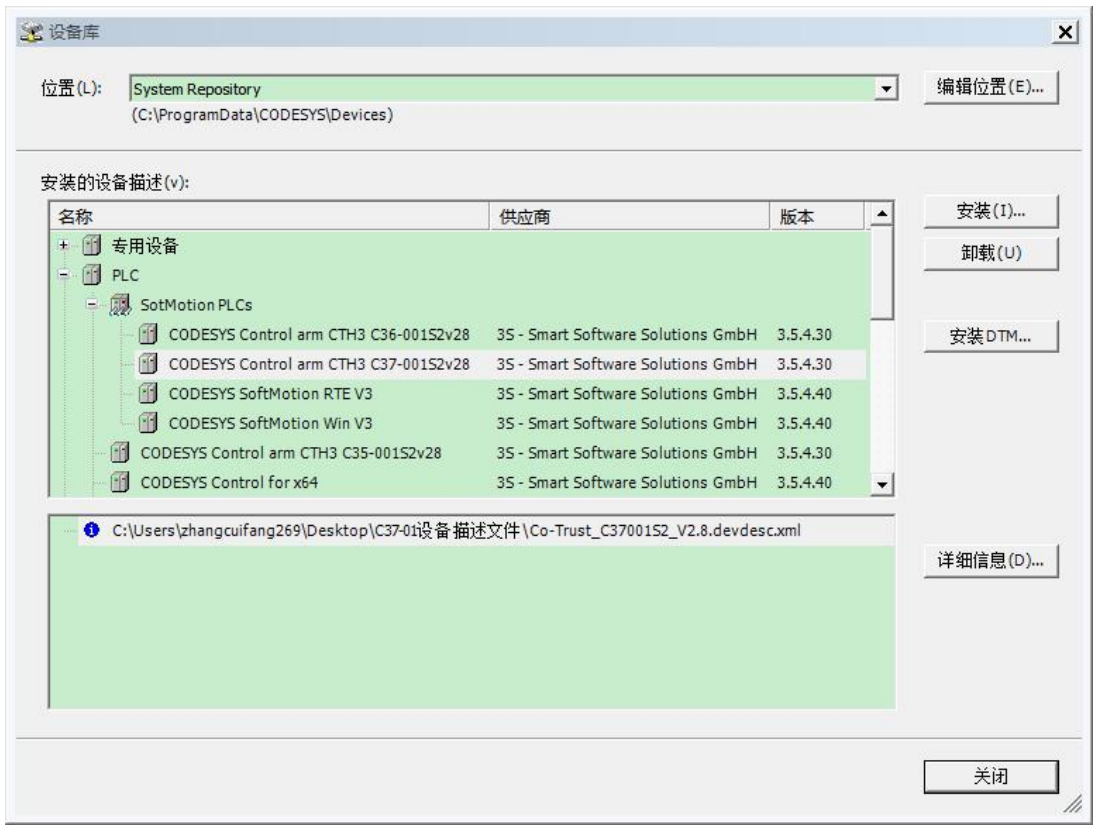


❒ 在打开的对话框中选择所需的 Co-Trust_C37001S2_V2.8.devdesc.xml，点击“打开”按钮确认选项后，新设备随即添加到“设备库”中的设备目录。由 COTRUST 提供的描述文件可通过设置相应的过滤器来选择。



以上操作执行完成后，可参考如下步骤进入设备目录查看文件是否正确安装：

选择菜单项“工具”→“设备库”打开设备库对话框，如下所示：



在设备库对话框中展开“PLC”→“SoftMotion PLCs”，即可在看到已经成功安装的 COTRUST 设备“CoDeSys Control arm CTH3 C37-001S2”。查看完毕，若还需要继续安装设备文件，请在“工具”→“选项”中取消勾选“使能设备库对话框”即可。

提示

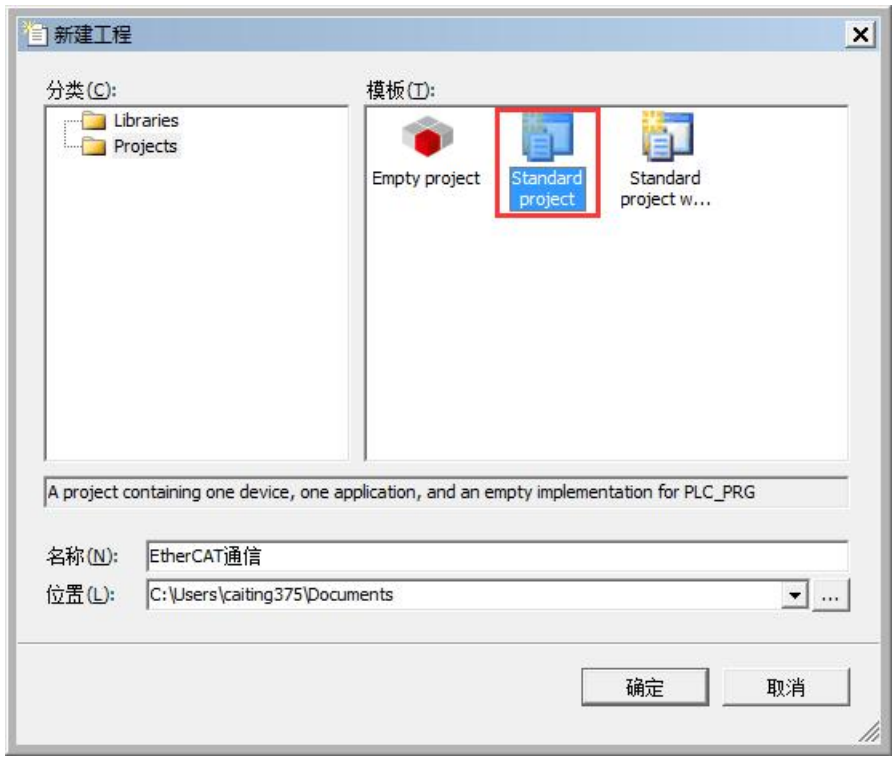
1）如果您还需要使用 COTRUST 的其它设备（CANopen 或 EtherCAT 从站设备），那么您还需要安装如下设备描述文件：

- CANopen 从站设备配置文件（*.eds）：Co-Trust E10.eds、Co-Trust EM277C.eds、Co-Trust H1A.eds；
- EtherCAT 从站设备配置文件（*.xml）：Co-Trust H1A.xml；
- Softmotion 设备配置文件（*.devdesc.xml）：H1A EtherCAT Drive(CoE) SoftMotion.devdesc、H1A CANopen Drive SoftMotion.devdesc；

2）请于本公司网站下载所需的设备配置文件：http://www.co-trust.com。

2、新建一个工程

（1）在 CODESYS 主页面选择菜单项“文件”→“新建工程”，然后在弹出的对话框中选择“Standard project”(标准工程)并设置文件名和存储目录。



（2）执行以上操作并确认后会弹出另一个对话框，在对话框中选择需要的文件设备（CTH3 C37）和编程语言，再点击“确定”即可完成工程的创建。



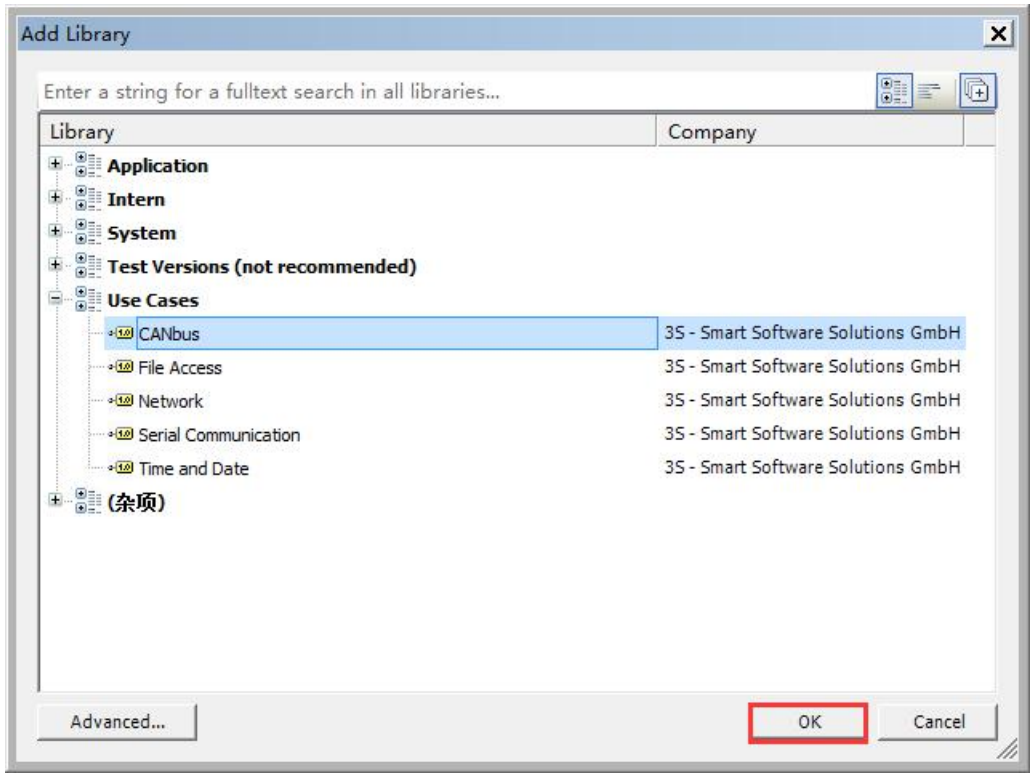
3、添加库

用户可根据具体需要在 CODESYS 中添加指令库，具体操作如下：

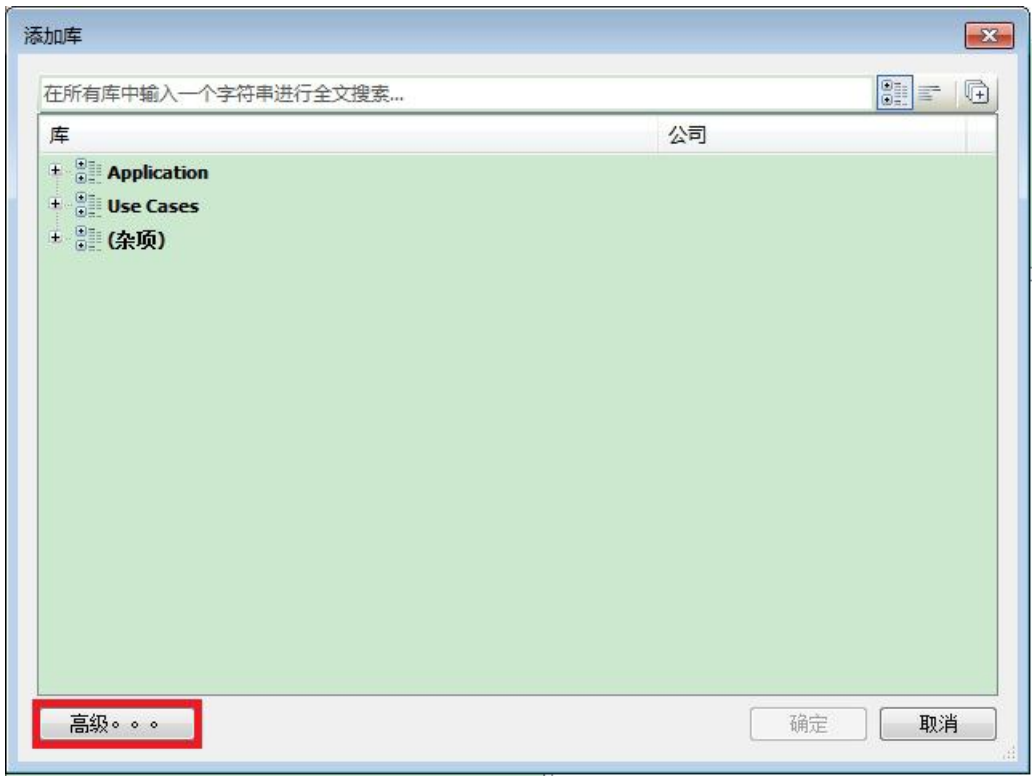
- 在项目树中打开“库管理器”并选择“添加库”。

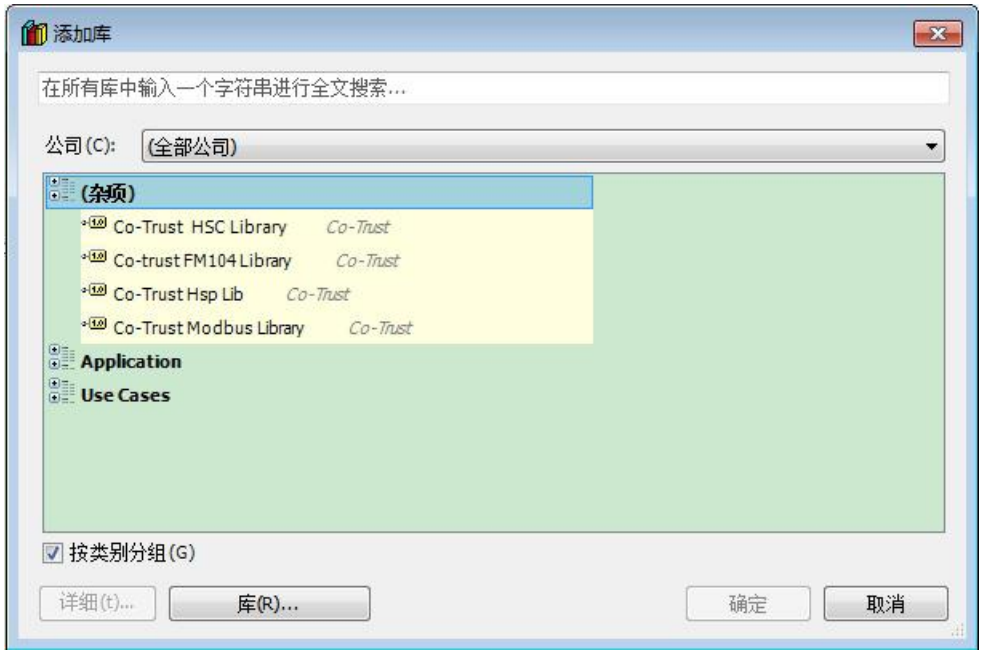


- 在显示的界面中选择要添加的库并单击“确定”，即可将所需库添加到库管理器中。



注意：对于 CODESYS SP4，在添加库时，需点击“高级”选项卡，然后在“杂项”中选择需要添加的库并单击“确定”，如下图所示：





2.3 设置通信

请参考如下步骤设置 C37-01 与上位机的通信：

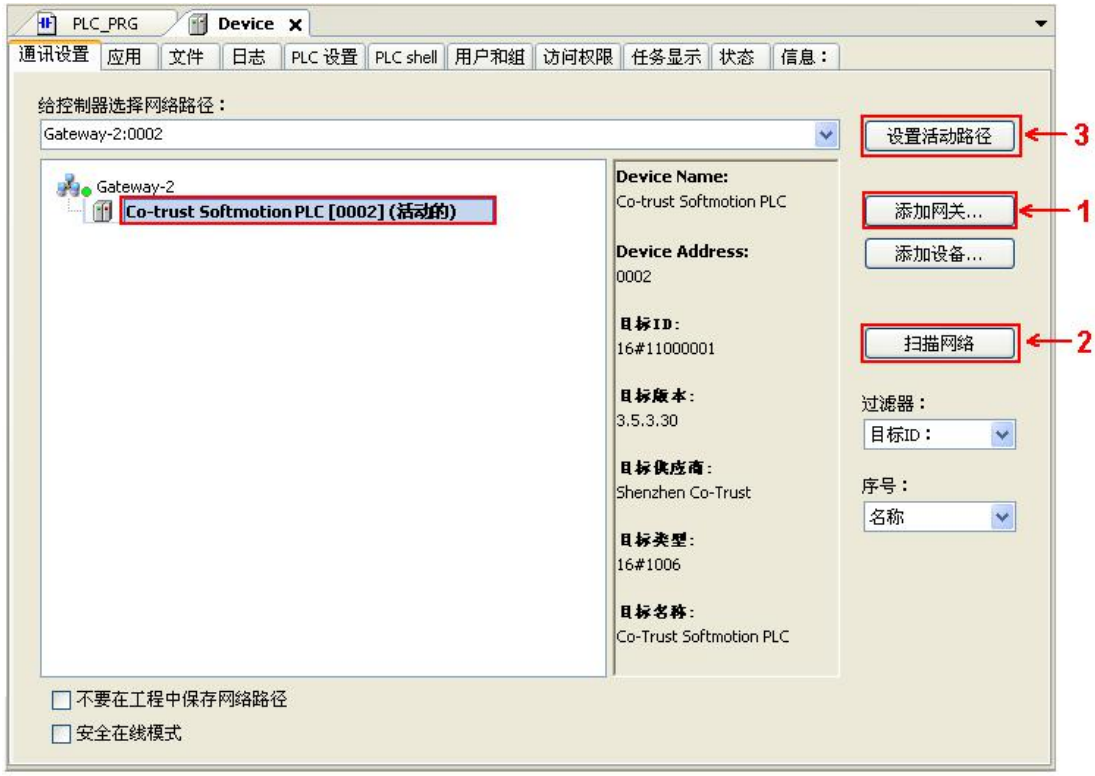
1、将编程设备的 IP 设为与 C37-01 同一个网段

设置通信前，需要将编程设备 PG/PC 的 IP 设置与 C37-01（IP：192.168.0.x）同一个网段，设置方法：将 PC 机的本地连接属性打开，双击 TCP/IP 协议，将“自动获得 IP 地址”更改为“使用下面的 IP 地址”，然后在 IP 地址中填写“192.168.0.X”即可。

2、在 CODESYS 设备视图中执行“通讯设置”

在设备视图中双击“Device (CoDessys Control from Co-Trust)”打开设备对话框，然后在选项卡“通讯设置”中点击“添加网关”按钮，在弹出的“网关”对话框中输入“名称”、“驱动器”选择“TCP/IP”、“IP 地址”选择“localhost”，最后点击“确定”关闭对话框，C37-01 即被添加到通讯对话框。

选中添加成功的 C37-01，然后点击“扫描网络”按钮，可以在本地网络上搜寻可用的设备。若搜索成功，则选中搜索到的设备并点击“设置活动路径”按钮，该操作会激活通讯通道设置，也就是所有与通讯相关的操作将与该通道关联。

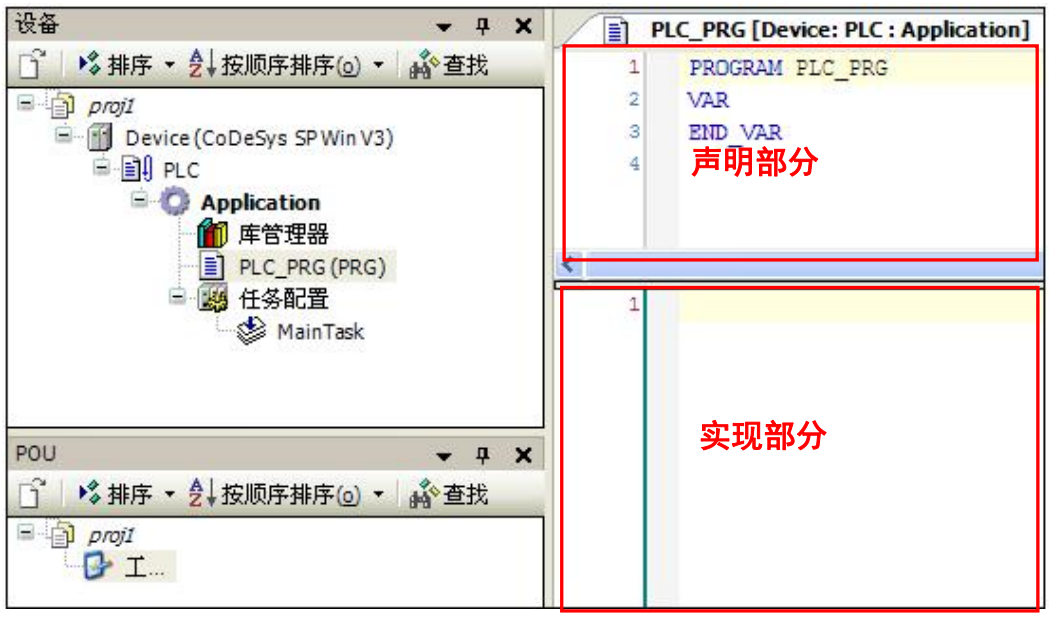


<备注> 系统启动时，系统托盘处会出现 CODESYS 相关服务程序（如、等），如无特殊要求，无需对托盘中的服务程序进行操作。

2.4 编写 PLC 程序

本例程序实现目的：定时器 1 和定时器 2 在 5S 间隔中周而复始的进行置位和复位操作。

在设备视窗中，缺省 POU 为“PLC_PRG”，双击设备视图中“PLC_PRG”，自动在 CODESYS 用户界面中部的 LD 语言编辑器中打开。LD 语言编辑器包含声明部分、实现部分。

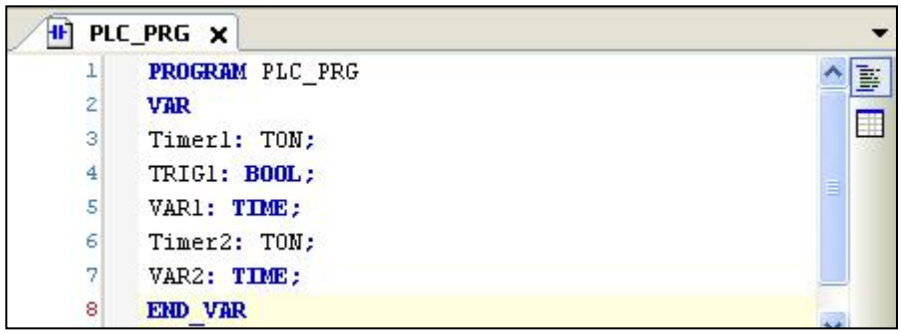


声明部分包括：显示在左侧边框中的行号、POU 类型和名称（如“PROGRAM PLC_PRG”），以及在关键字“VAR”和“END_VAR”之间的变量声明。

实现部分为空，仅显示行号 1。

1、在 PLC_PRG 中声明变量

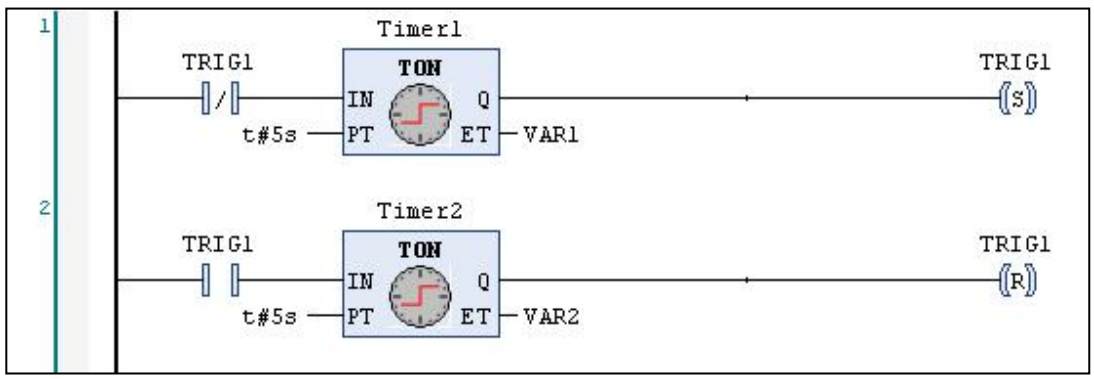
在编辑器的声明部分，将光标移到 VAR 后，点击回车，插入新的空行，声明您需要使用的变量。或者在程序的实现部分，使用自动声明功能：在程序的实现部分输入指令，点击回车，如果在新行中有未声明的变量，则系统打开自动声明对话框，在此可以进行声明设置。本例声明部分内容如下图：



2、在 PLC_PRG 的实现部分输入指令

在 LD 语言编辑器右侧的工具箱中展开“梯形图元素”，使用鼠标将“通电延时定时器 TON”拖曳到语言编辑器的实现部分，然后从工具箱的“梯形图元素”中拖曳一个复位线圈到 TON 指令输出点 Q 后方。使用同样的操作，在网络 2 中创建一个 TON 指令。

本例最终完成的 LD 程序如下：



2.5 编译并运行程序

1、保存当前工程并进行编译

程序编写完成后，保存当前工程，然后选择菜单项“编译”即可对当前对象进行语法检查，语法检查完成时，任何错误消息和警告会在“编译”类的消息窗口中显示出来。若无错误，则表示编译成功。

2、登录、下载和启动程序

选择菜单项“在线”→“登录...”使应用程序与 C37-01 建立起连接，并进入在线状态，如果已经进行了通讯设置，会弹出如下对话框提示：



点击“是”启动应用的编译和下载。

最后，选择菜单项“调试”→“启动”使 C37-01 中的应用程序开始运行，此时即可对当前工程进行监控和调试操作。

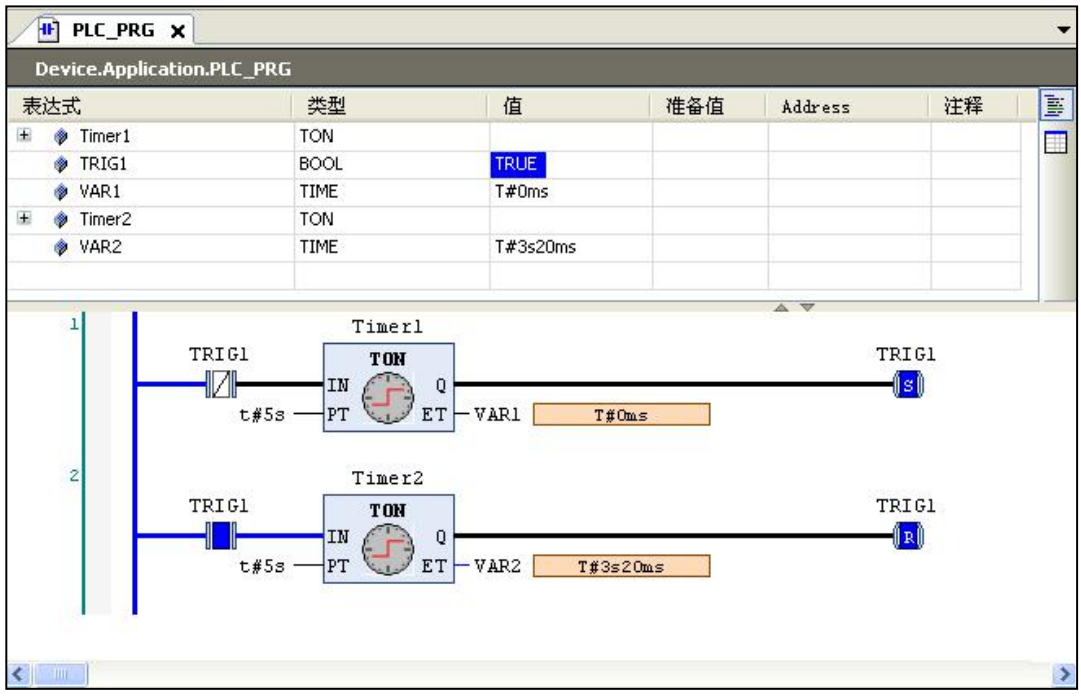
2.6 监控与调试

使用以下三种方法，可以监视应用程序中的变量：

- 包含已定义监视列表的监视窗口
- 写入和强制变量
- 特殊 POU 的在线视图

1、打开程序的示例窗口

双击打开 PLC_PRG，出现如下在线视图：上半部分显示对应于 PLC_PRG 实现部分视图的程序主体，由每一个变量后的内部监视窗口显示实际的值。



2、写入和强制变量

通过写入或强制的方式将某一“准备值”赋给变量 TRIG1，在下一周期开始，变量将显示为该值。在“准备值”的输入框输入所需的整数值，单击回车或者该区域外部，然后执行命令“写入值”或者“强制值”，即可将该值写入或强制到 PLC。

3、使用监视视窗

选择菜单项“视图”→“监视”→“监视 1”打开监视窗口。然后，鼠标点击表达式列第一行，打开编辑框，输入要监视的变量 TRIG1 的完整路径：“Device.Application.PLC_PRG.TRIG1”，随后即可对变量进行写入和强制值。



3 运动控制器的基本概念

本章将向您介绍有关 CTH300-C 系列运动控制器如何执行逻辑控制、存储区数据存取以及其特性等概念。

3.1 CTH300-C 系列运动控制器如何执行您的控制逻辑

CTH300-C 系列运动控制器循环执行程序中的控制逻辑和读写数据。

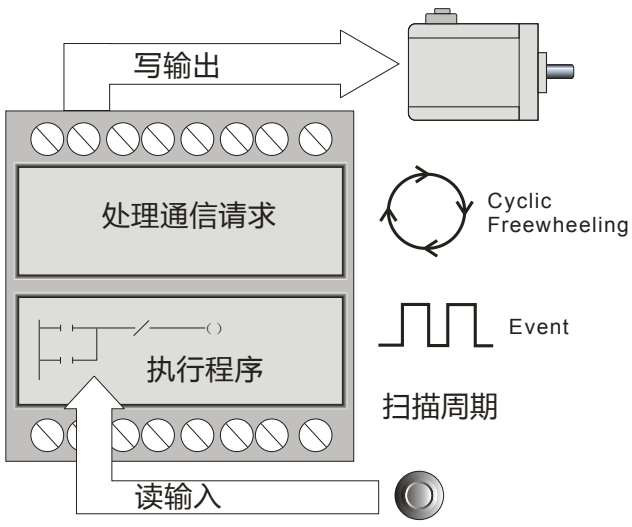
运动控制器将你的程序与物理输入输出点联系起来

- CPU 读输入状态
- CPU 中存储的程序利用输入执行逻辑控制，当程序运行时，CPU 刷新有关数据
- CPU 将数据写到输出

在扫描周期中完成它的任务

运动控制器可通过三种扫描方式执行它的任务：Cyclic、Event、FreeWheeling

- Cyclic（循环）：按照给定的时间间隔循环执行程序。
- Event（事件触发）：事件遇上升沿便开始执行程序。
- FreeWheeling（惯性滑行）：程序开始便执行，在结束一轮运行时将重新自动执行一个循环。



在一个扫描周期内，CTH300-C 系列运动控制器将执行以下部分或全部操作：

- 读输入（数字量/模拟量）
- 执行逻辑控制程序
- 执行通信请求
- 写输出（数字量）

选择操作模式

CTH300-C 系列运动控制器有两种操作模式：停止（STOP）和运行（RUN）。运动控制器前面板上的 LED 指示灯显示了当前的操作模式。在停止模式下，运动控制器不执行程序，您可以下载程序和组态；在运行模式下，运动控制器将运行程序。

• CTH300-C 系列运动控制器提供一个模式开关来改变操作模式，因此您可使用模式开关手动选择操作模式。将开关打在 RUN 模式，可以启动程序的执行；将开关打在 STOP 模式，可以停止程序的执行。

• CODESYS 允许您改变与之相连的运动控制器的操作模式。如果希望使用软件来改变操作模式，运动控制器的模式开关必须打在 RUN 模式，您可以用菜单命令“调试”→“停止”或“启动”来改变操作模式。

3.2 CTH300-C 系列运动控制器的数据存储区

CTH300-C 系列运动控制器将信息存储于不同的存储器单元，每个单元都有唯一的地址，您可以明确指出要存取的存储器地址，这就允许用户程序直接存取这个信息。

内存位置和大小是由特殊字符序列表示的，语法：%<范围前缀><长度前缀><数字|.数字|.数字....>

可以使用以下内存区域：

存储区	名称	大小（Bytes）
I	输入区(通过输入驱动的物理输入,“传感器”)	1280
Q	输出区(通过输出驱动的物理输出,“执行器”)	1280
M	内存区	1280

可以使用以下的长度前缀：

X	单个二进制位
None	单个二进制位
B	字节（8位）
W	字（16位）
D	双字（32位）

例如：

%QX7.5	输出区的地址7，第5位
%IW215	输入区的地址215，1个字
%QB7	输出区的地址7，1个字节
%MD48	内存区的地址48，双字。
%IW2.5.7.1	取决于 PLC 的设置
ivar AT %IW0 : WORD;	t 带有指定地址的变量声明的例子

请确定使用有效的地址：

要在应用中指定一个有效地址，首先您必须知道进程映像中的适当位置，也就是所使用的内存区：Input(I)，Output(Q)或者 Memory(M)区域，如上表所述。然后关于指定想要的大小：

当前设备配置和设定（硬件结构、设备描述、IO 设置）扮演了决定性的角色。尤其要考虑到，使用“字节寻址模式”的设备和使用面向字模式的“IEC 寻址模式”设备在地址解释之间差异。

所以依赖于大小和寻址模式，不同的内存单元可被相同的地址定义寻址。

见下表中字节寻址和面向字的 IEC 寻址对于 bit，bytes，words 和 dwords 的比较。这个表画出了字节寻址模式时重叠的内存区（见表后示例）。

关于写法，请注意对于 bit 地址，IEC 寻址模式总是面向字的，就是说在点号之前的区域对应于字的个数，名称之后的区域对应于 bit 的个数。

对于地址大小 D，W 和 X，面向字节和字的寻址的比较：

双字/字				字节	X（位）					
字节寻址		面向字的 IEC 寻址			字节寻址			面向字的 IEC 寻址		
D0	W0	D0	W0	B0	X0.7	...	X0.0	X0.7	...	X0.0
D1	W1			B1	X1.7	...	X1.0	X0.15	...	X0.8
...	W2		W1	B2				X1.7	...	X1.0
	W3			B3				X1.15	...	X1.8
	...	D1	W2	B4						
				B5						
			W3	B6						
				B7						
		D2	W4	B8						
		...		...						
		...								
		...								
D(n-3)		D(n/4)								
	W(n-1)		W(n/2)							
				Bn	Xn.7	...	Xn.0	X(n/2).15	...	X(n/2).8

n =字节数

在字节寻址模式时，内存重叠是变化的，例如：

D0 包括 B0-B3，W0 包括 B0 和 B1，W1 包括 B1 和 B2，W2 包括 B2 和 B3。为了避免重叠，W1 或者 D1,D2,D3 都不能用于寻址！

<注意> 如果没有指定明确的单个二进制位的地址，布尔值的长度将为一个字节。例如：改变 varbool1 AT %QW0 的值将影响从 QX0.0 到 QX0.7 的内容。

<注意> 使用指向地址的指针时注意，在线修改可能会改变地址中的内容！

3.3 CTH300-C 系列运动控制器的特性

3.3.1 通信功能

为网络选择通信接口

CTH300-C 系列运动控制器支持各种类型的通信网络，能够访问这些网络的接口包括：

- RS485 接口（RS485 接线端子）
- EtherCAT 接口（RJ45 标准网口）
- CANOpen 接口（RJ45 标准网口）
- EtherNET 接口（RJ45 标准网口）

为网络选择通信协议

CTH300-C 系列运动控制器支持的协议包括：

- 总线协议
- UDP 协议
- Modbus TCP 协议
- Modbus RTU 协议
- CANopen DS301 标准协议
- EtherCAT 协议

建立基于通信协议的网络

通信方式	通信接口和协议	在 CODESYS 上实现通信
总线通信	通过 CTH300-C 运动控制器的总线接口（支持总线协议）连接 CTH300-C 系列扩展模块，可实现总线通信。	通过设备目录的“Co-trust LocalBus”添加与实际扩展模块相符的设备，然后通过相应模块的 IO 映射读写内存。
EtherNet 通信	通过 CTH300-C 运动控制器的 EtherNET 通信口（支持 UDP 协议），经由通信电缆连接 CTH300-C 系列运动控制器与通信设备，可进行以太网编程。	
EtherCAT 通信	通过 CTH300-C 运动控制器的 EtherCAT 接口（支持 EtherCAT 协议），经由标准的以太网电缆连接 CTH300-C 运动控制器与 EtherCAT 从站，即可构建线型网络。	通过设备目录添加 EtherCAT 通信设备：EtherCAT 主站、EtherCAT 从站等。
Modbus 通信	通过 CTH300-C 运动控制器的 RS485 接线端子可进行 Modbus RTU 通信；使用其 EtherNET 通信口可进行 Modbus TCP 通信时。	使用 COTRUST 提供的 ct_Modbus 库文件可进行 MODBUS RTU 和 MODBUS TCP 通信，Ct_Modbus 库的使用请参考附录“E CT_MODBUS 库的使用介绍”。
CANopen 通信	1、CTH300-C 运动控制器通过总线扩展 CANoepn 主站模块（CAN-1T），然后 CANopen 主站模块经由标准网线与 CAN 从站相连，即可实现 CANopen 通信。	通过设备目录添加 CANopen 通信设备：CANBUS、CAN 主站、CAN 从站、CAN 从站模块等。

<备注> 请参考章节“7 使用指南”获取各种通信方式的相关使用。

3.3.2 运动控制功能

采用高性能运动控制软件平台，内嵌单轴或多轴运控算法和指令。

电子凸轮功能

- 支持电子齿轮/电子凸轮等同步功能，支持虚轴和编码器轴。
- 图形凸轮编辑器可实现对复杂运动控制轨迹的快速执行。可对凸轮表进行实时、动态修正。
- 可以通过运动控制库内的 PLCopen 功能块实现一般功能的运动控制。

CNC 功能

- 支持 DIN66025 G 代码及 DXF 文件导入。
- 支持 5 轴联动，支持并行机器人控制。
- CNC 路径视图，所有在 CNC 文本编辑器中变化的内容会在路径视图中自动更新。

3.3.3 逻辑控制功能

程序组织单元

CTH300-C 运动控制器由以下程序组织单元构成：

程序（Program）：程序是存放控制应用程序指令的位置，程序中的指令按照顺序、不同的扫描方式执行。

功能块（Function Block）：功能块是用户编写的有自己的存储区的块，每次调用功能块时需要提供各种类型的数据给功能块，功能块也要返回变量给调用它的块。

功能（Function）：功能（FC）属于个人编程的块。功能是一种不带存储器的逻辑块。属于 FC 的临时变量保存在本地数据堆栈中。由于 FC 本身没有存储器，因此，必须始终给它指定实际参数。不能给 FC 的本地数据分配初始值。

编程语言

支持所有在 IEC_61131 标准中规定的编程语言：

- 文本形式的语言：指令表（IL）、结构化文本（ST）
- 图形化的语言：顺序功能图表（SFC）、功能模块图（FBD）、梯形图（LD）、基于功能模块图的连续功能图表（CFC）

<备注> 请参考章节“6.2 语言”获取编程语言的说明。

标准化的 PLC 功能

通过使用相关指令可实现标准化的 PLC 功能，如：类型转换、数值功能、算术功能、移位功能、布尔运算功能、选择功能、比较功能、字符串功能、定时器、计数器、边沿检测、双稳态元素等。

3.3.4 实时时钟

- 掉电保持时间约 112 小时（典型值）
- 每月偏差<60 秒
- 通过指令 setrtc 读取/设置时钟

在 CODESYS 中设置或读取 CTH300-C 系列运动控制器的实时时钟：在设备视图双击打开“Device(CODESYS Control from Co-Trust)”，然后选择选项卡“PLC SHELL”：

1）在 PLC SHELL 中输入命令，如“setrtc 2013-09-23 08:30:00”，然后按 ENTER 键即可设置当前 RTC；

2）在 PLC SHELL 中输入命令“setrtc”，然后按 ENTER 键可读取当前 RTC；

<备注> CTH300-C 运动控制器本机出厂时拥有一个随机时间，因此您需要先手动设置当前时间，再次读取 RTC 时方可显示正确的时间。

4 CTH300-C 系列运动控制器的技术规格

4.1 技术参数

表 4-1 常规规格

产品型号	C35-00	C36-00	C37-00	C35-01	C36-01	C37-01	C37-02
物理特性							
尺寸（W×H×D）	34×115×101.6 mm						
功耗	19.2W						
内存							
用户程序空间	32MB						
掉电保持空间	32KB			64KB			120KB
电源特性							
额定输入电压	24V DC						
输入电压范围	20.4V~28.8V DC						
输入电流	0.8A						
极性反接保护	有						
总线电源电压	+5V DC						
总线电源电流	1.6A						
隔离	外部电源与系统电源之间隔离						
LED 指示灯特性							
24V 电源指示灯（绿色）	亮起：24VDC 供电正常，熄灭：无 24VDC 供电						
SF 指示灯（红色）	亮起：系统故障，熄灭：无错						
BF 指示灯（红色）	亮起：总线故障，熄灭：无错						
RUN 指示灯（绿色）	亮起：系统运行，熄灭：系统停止						
STOP 指示灯（橙色）	亮起：系统停止，熄灭：系统运行						
RJ45 网口指示灯	绿色	闪烁：连接，熄灭：未连接					
	黄色	亮起：100Mbps，熄灭：10Mbps					
CAN 口指示灯	绿色	--		数据接收指示灯			
	黄色	--		数据发送指示灯			
实时时钟							
掉电保持	掉电保持时间约 112 小时（典型值）						
精度	每月偏差<60 秒						
读取/设置实时时钟	通过指令 setrtc 读取/设置						
保护功能							
电源保护	供电电源端提供反接保护功能及浪涌吸收功能						
接口保护	通讯端口防雷保护						
扩展 I/O 能力							
1 个 CPU 支持 INT-00 机架数	4						
每机架支持最多模块数	除了电源、CPU、中继模块外最多支持 8 个模块						

1 个 CPU 支持 ECT-00 从站数	最多 125 个
每 ECT-00 支持模块数	最多支持 8 个模块
隔离	
电源隔离	外部电源与系统电源之间隔离
通讯隔离	以太网、RS485 隔离、CAN 隔离

表 4-2　通讯口规格

产品型号	C35-00	C36-00	C37-00	C35-01	C36-01	C37-01
EtherNET 通讯口						
通讯接口	1个标配以太网口					
协议类型	CODESYS 定制协议，基于 UDP 和 TCP （支持以太网编程和 CPU 间以太网通信）					
每段最大电缆长度	100m					
一个站点最大连接数	UDP 最多支持16个连接；Modbus_TCP 最多支持32个连接					
波特率	10/100Mbps 自适应					
隔离	通信口隔离					
EtherCAT 通讯口						
通讯接口	1个 EtherCAT 接口					
最大站点数	每个主站最多支持125个伺服从站					
协议类型	EtherCAT					
支持功能	支持分布时钟配置					
	支持启动参数配置					
	支持配置 PDO 参数和映射					
	支持配置总线循环周期					
每段最大电缆长度	100m					
波特率	10/100Mbps 自适应					
隔离	通信口隔离					
RS485通讯口						
通讯接口	1个 RS485通讯口					
最大站点数	每段32个站，每个网络126个站					
协议类型	Modbus 主站/从站协议					
Modbus 波特率	1200\4800\9600\19200\38400\57600\115200bps					
隔离	通信口隔离					
CANopen 通讯口						
通讯接口	--			1 个 CAN 通讯主站接口		
最大从站点数	1 个主站后面最多可连接 32 个从站					
协议类型	CANopen DS301 标准协议					
支持功能	支持自动启动 CANopen Manager					
	支持可选从站轮询					
	支持启动从站					
	支持 NMT					
	支持同步生产					
	支持同步消耗					
	支持心跳产生					

	支持激活时间创建						
传输速率（kbit/s）	1000	800	500	250	125	50	20
最大长度（m）	25	50	100	250	500	1000	2500
隔离	通信口隔离						

4.2 接口定义

表 4-3 电源接口（X4）信号定义

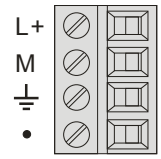
4 位可拆卸端子（X4）	符号	信号定义
	L+	24V 电源正
	M	24V 电源负
	⏚	大地
	--	--

表 4-4 EtherNET/EtherCAT 通信接口信号定义

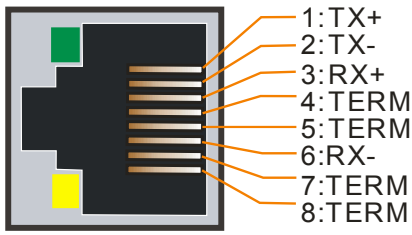
RJ45 网口	位号	信号	信号定义
	1	TX+	数据发送正端
	2	TX-	数据发送负端
	3	RX+	数据接收正端
	4	--	--
	5	--	--
	6	RX-	数据接收负端
	7	--	--
	8	--	--
	连接器外壳	PE	机壳接地

表 4-5 CAN 接口定义

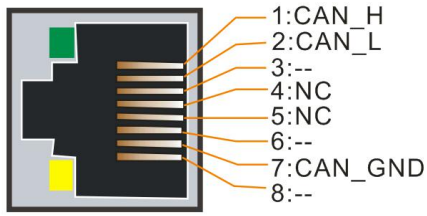
RJ45 网口	位号	信号	信号定义
	1	CAN_H	数据发送正端
	2	CAN_L	数据发送负端
	3	--	--
	4	--	--
	5	--	--
	6	--	--
	7	CAN_GND	CAN 的地
	8	--	--
	连接器外壳	PE	机壳接地

表 4-6　RS485 接线端子信号定义

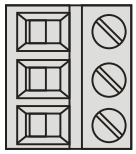
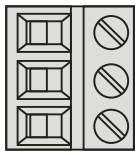
6 位可拆卸端子（PORT0）		信号	信号定义
X2		A0	RS485 信号 A
		B0	RS485 信号 B
		SG	RS485 信号地
X3		A0’	RS485 信号 A
		B0’	RS485 信号 B
		SG	RS485 信号地

表 4-7　系统运行开关拨码定义

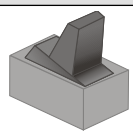
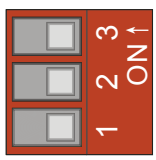
系统运行开关	信号	拨码方向	信号定义
	RUN	向上	系统运行
	STOP	向下	系统停止

表 4-8　DIP 开关（X1）拨码定义

DIP 开关（X1）	位号	信号	信号定义
	1	Term-PORT0	RS 485 终端电阻开关 ON：有终端电阻；OFF：无终端电阻
	2	NC	保留
	3	NC	保留

5 安装

CTH300-C 系列 PLC 的外形设计使其极易安装。在现场可以利用安装孔把模块固定在控制柜的背板上，或者利用设备上的 DIN 夹子，把模块固定在一个标准（DIN）的导轨上。

本节将提供 CTH300-C 系统的安装和接线指导。

5.1 安装注意事项

CTH300-C 系列 PLC 的安装须注意以下事项：

❒ 将 CTH300-C 系列 PLC 与加热装置、高电压和电子噪声隔离开

按照一般惯例，在安装设备器件时，总是把产生高电压和高电子噪声的设备与 CTH300-C 系列 PLC 这样的低压电子型的设备分隔开。

在控制柜的背板上安排 CTH300-C 系列 PLC 时，应考虑把电子器件安排在控制柜中温度较低的区域内。电子器件长期在高温环境下工作会缩短其无故障时间。

要考虑控制柜的背板布线，尽量避免把交流供电线、高能量、开关频率很高的直流信号线与低压信号线、通讯电缆设计在同一个线槽中。

❒ 为散热和接线留出适当的空间

CTH300-C 系列 PLC 的设计采用自然对流散热方式，在模块的上下方都必须留有至少 30mm 的空间，以便于正常的散热。前面板与背板的板间距离也应保持至少 80mm。

⚠注意

在垂直安装时，其允许的最高环境温度要比水平安装时低 10℃，而且 CPU 应安装在所有扩展模块的下方。

在安装 CTH300-C 系列 PLC 时，应留出足够空间用于接线和连接通讯电缆。

下图显示的是安装在多个机架上的 CTH300-C，其中显示了各机架与相邻组件、电缆槽、机柜之间的间距。通过电缆槽为模块接线时，屏蔽连接元件底部与电缆槽间的最小间距为 40 mm。

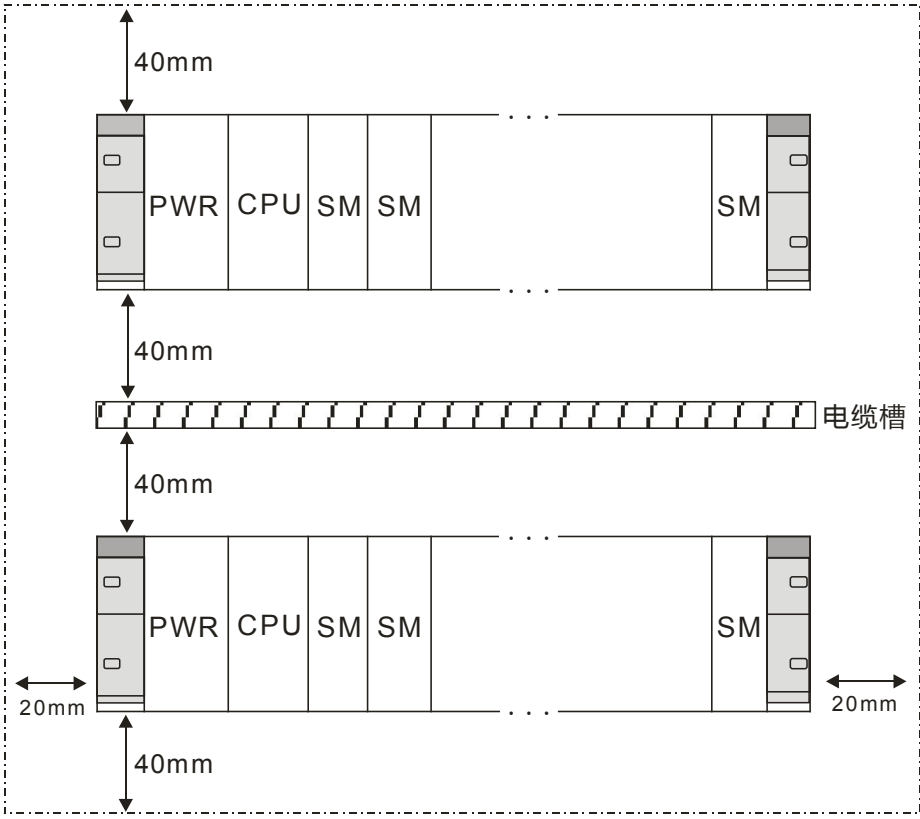


图 5-1　安装示意图

❒ 电源预算

选择好各机架的 CPU、电源模块、中继模块和扩展模块后，还需要确认系统总线的电流消耗和功率消耗是否满足以下条件：

条件 1：总线电流消耗确认

内部总线电压为 5VDC，电流由 CPU（无中继模块时）或中继模块提供。每个机架上扩展模块总线电流消耗之和不能超过 CPU 或中继模块允许的最大总线电流。

条件 2：功率消耗确认

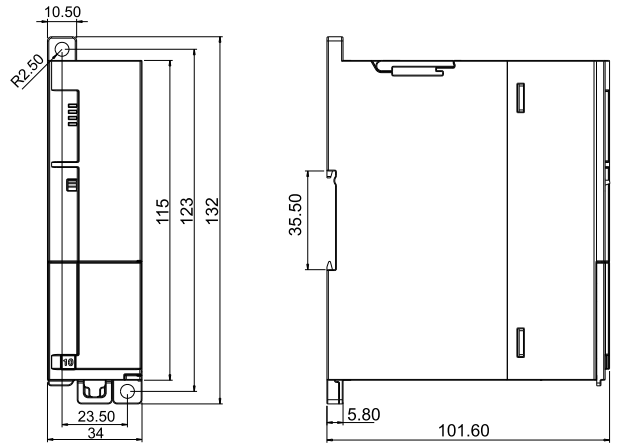
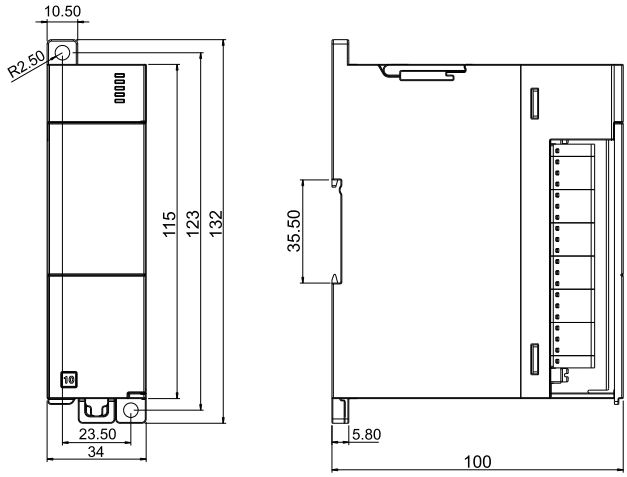
使用电源模块时，每个机架上其它模块的功率消耗之和不能超过电源模块允许的最大功耗。

使用外部电源时，根据所接的功率之和选择合适功率大小的型号。

<备注> CTH300-C 系列 PLC 的总线电源电流的预算请参考附录“B 电源预算”。

5.2　安装尺寸

CTH300-C 系列 PLC 都有安装孔，可以很方便地安装在背板上，各模块安装尺寸见下表。

型号	尺寸图（单位：mm）
C35-00、C35-01 C36-00、C36-01 C37-00、C37-01、C37-02 HSC-02、HSP-04 CAN-1T、ECT-00 PWR-02 INT-00	 正面视图　　侧立视图
DIT-08、DIT-16、DIT-32 DQT-08、DQT-16、DQT-32 DQR-08、DQR-16 AIS-04、AIC-08、AIV-08 AQS-04、AQS-08、AMS-06 AIT-04、AIT-08、AIR-04、AIR-08	 正面视图　　侧立视图

5.3 使用机架

❒ 中央机架和扩展机架

CTH300-C 系统由一个中央单元和一个或多个扩展机架组成。

包含 CPU 的机架是中央单元。配有模块并连接到中央机架形成了系统的扩展机架。

❒ 扩展机架的使用

如果中央机架已经用完了所有插槽，则可以使用扩展机架。

使用扩展机架时，除额外的机架和中继模块（INT）之外，可能还需要更多的电源模块。使用中继模块时，必须确保与扩展站相兼容。

❒ 机架上的模块布局

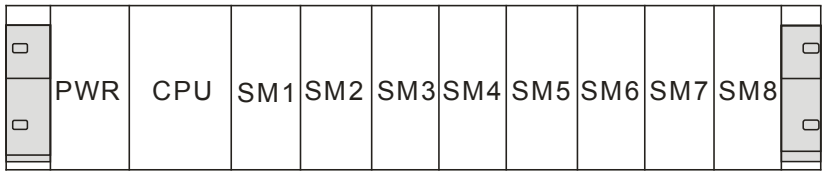
CTH300-C 的机架是一个装配导轨。可利用此导轨安装 CTH300-C 系统所属的模块。

① 一个机架上的模块布局

如果要在一个机架上安装模块，请注意以下几点：

- CPU 右侧安装的模块数不超过八个（SM、FM、CP）。
- 安装在机架上的模块的累积功耗在 CTH300-C 总线上不得超过 1.6A。

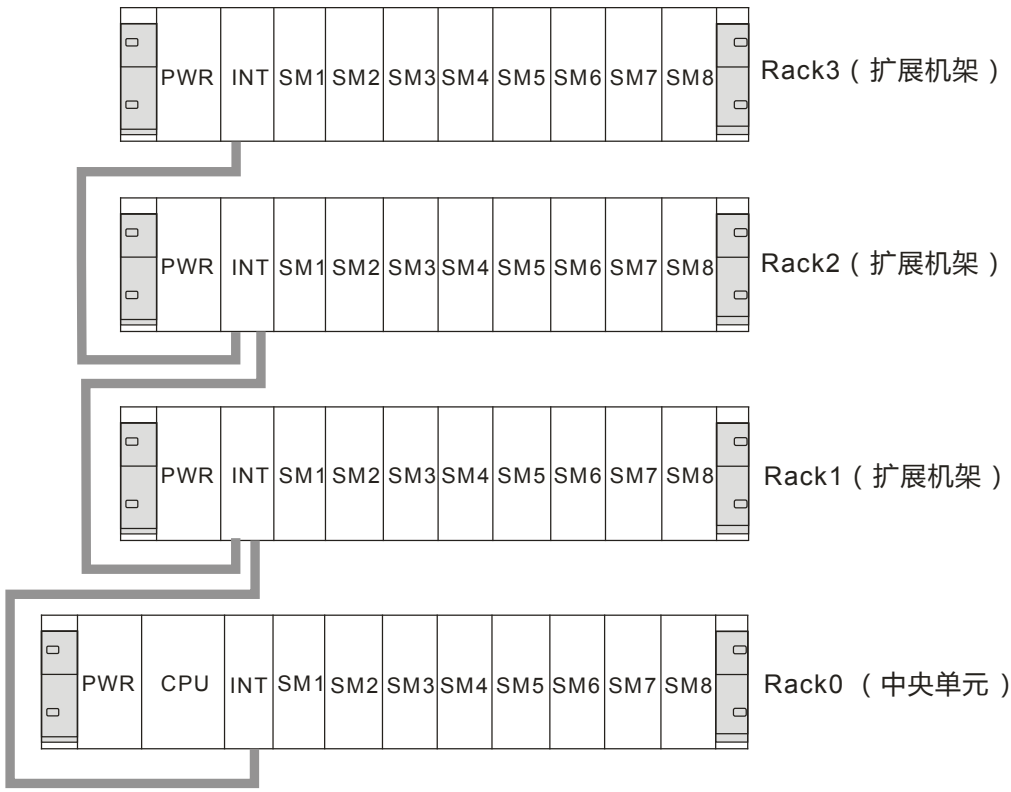
下图显示在一个 CTH300-C 装配中具有八个信号模块的布局。



② 四个机架的完整装配

如果计划在多个机架上执行装配，需要使用中继模块（INT）。中继模块将 CTH300-C 的总线连接到下一个机架。

下图显示 CTH300-C 系统中的各模块在 4 个机架上的排列情况。



如果要在多个机架上排列模块，请注意以下几点：

- INT 模块始终使用插槽 3（插槽 1：PWR；插槽 2：CPU，插槽 3：INT）
- 在插入第一个信号模块前它始终位于左侧。
- 每个机架上安装的模块数不超过八个（SM、FM、CP）。
- 模块（SM、FM、CP）数量受到 CTH300-C 总线上允许的电流消耗的限制。每个机架的累积功耗不得超过 1.6A。

5.4 安装方法

❒ 先决条件

在安装和拆卸 CTH300-C 系列 PLC 及其相关设备时，必须预先采取适当的安全措施并且确认 CTH300-C 系列 PLC 的供电被切断。

警告

试图在带电情况下安装或拆卸 CTH300-C 系列 PLC 及其相关设备有可能导致电击或者设备误动作。在安装和拆卸 CTH300-C 系列 PLC 及其相关设备时，如果未能切断所有电源，有可能造成死亡或严重的人身伤害和设备损坏。

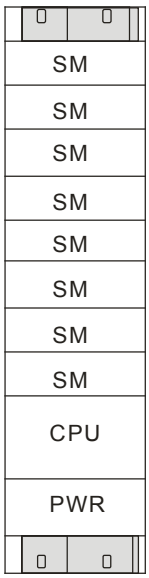
在更换或安装 CTH300-C 系列 PLC 时，要确定使用了正确或等同的模块。在更换 CTH300-C 系列 PLC 时，除了要使用相同的模块外，还要确保安装的方向和位置是正确的。

注意

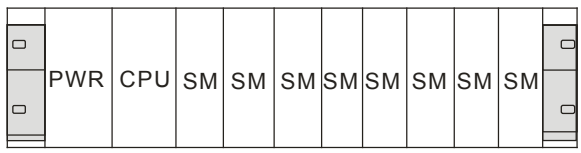
- 如果您安装了不正确的模块，CTH300-C 系列 PLC 的程序可能会产生错误的功能。
- 如果未能使用相同的模块按照相同的方向和顺序替换 CTH300-C 系列 PLC，有可能导致死亡或者严重的人身伤害和设备损坏。

❒ 安装方式

CTH300-C 系列 PLC 既可以安装在控制柜背板上，也可以安装在标准 DIN 导轨上；既可以水平安装，也可以垂直安装，安装时始终将 CPU 和电源模块安装在左侧或底部。



垂直安装



水平安装

❒ 安装与拆除操作

请按照以下方法安装或拆除 CTH300-C 系列 PLC。

● 安装面板

1）按照尺寸要求定位打孔；

2）用合适的螺钉将模块固定在背板上；

3）如果使用了扩展模块，将扩展模块的扁平电缆连到前盖下面的扩展口。

● DIN 导轨安装

1）将导轨固定在背板上，保持间距 80mm；

2）打开模块底部的 DIN 夹子，将模块背部卡在 DIN 导轨上；

3）如果使用了扩展模块，将扩展模块的扁平电缆连到前盖下面的扩展口；

4）旋转模块贴近 DIN 导轨，合上 DIN 夹子；

5）仔细检查模块上 DIN 夹子与 DIN 导轨是否紧密固定好；

6）为避免模块损坏，不要直接按压模块正面，而要按压安装孔的部分。

注意

当 CTH300-C 系列 PLC 在振动比较大的使用环境或者采用垂直安装方式时，应该使用 DIN 导轨挡块。如果系统处于高振动环境中，使用背板安装方式可以得到较高的振动保护等级。

● 拆卸 CPU 或者扩展模块

1）拆除 CTH300-C 系列 PLC 的电源；

2）拆除模块上的所有连线和电缆；

3）如果有其它扩展模块连接在您所拆卸的模块上，请打开前盖，拔掉相邻模块的扩展扁平电缆；

4）拆掉安装螺钉或者打开 DIN 夹子；

5）拆下模块，拆卸和安装端子排。

● 端子排的拆卸

1）打开端子排安装位置的上盖；

2）把螺丝刀插入端子块中央的槽口中；

3）用力下压并撬出端子排。

● 端子排的安装

1）打开端子排安装位置的上盖；

2）确保模块上的插针与端子排边缘的小孔对正；

3）将端子排向下压入模块。确保端子块对准了位置并锁住。

5.5 接地和布线

❒ CTH300-C 系列 PLC 接地和接线指南

合理的接地和接线对于所有的电器设备是至关重要的，它能够确保您的系统具备最优的操作特性，同时能够为您的系统提供更好的电子噪声保护。

在接地和接线之前，必须先确保设备的电源已被切断，也要保证与该设备相关的设备电源已被切断。

在对 CTH300-C 系列 PLC 及其相关设备接线时，必须确保遵从所有有效的电气编码规则。安装和操作所有设备要符合所有有效的国家或地区标准。同地区的权威保持联系，以确定哪些标准符合您的特殊需要。

警告

试图在带电情况下进行接地或接线，有可能造成死亡或严重的人身伤害和设备损坏。

在设计 CTH300-C 系统的接地和接线时必须考虑安全因素，否则有可能造成设备的误动作。因此，您应该执行所有的安全规定以避免人员伤害和设备损坏。

警告

控制设备有可能造成它所控制设备的误操作。这种误操作有可能导致死亡或者严重的人身伤害和设备损坏。因此 CTH300-C 系统中必须具备独立于 CTH300-C 系列 PLC 的急停功能、机电互锁或者其它冗余的安全设施。

5.6 抑制电路

在使用感性负载时，要加入抑制电路来限制输出关断时电压的升高。抑制电路可以保护输出点不至于因为高感抗开关电流而过早的损坏。另外，抑制电路还可以限制感性负载开关时产生的电子噪声。

注意

抑制电路的有效性取决于应用，您应该调整其参数以适应您的特殊应用。要确保所有器件参数与实际应用相符合。

❒ 晶体管输出和控制直流负载的继电器输出

晶体管输出有内部保护，可以适应多种应用。由于继电器型输出既可以连接直流负载，又可以连接交流负载，因而没有内部保护。

图 5-2 给出了直流负载抑制电路的一个实例。在大多数的应用中，用附加的二极管 A 即可，但如果您的应用中要求更快的关断速度，则推荐您加上齐纳二极管 B。确保齐纳二极管能够满足输出电路的电流要求。

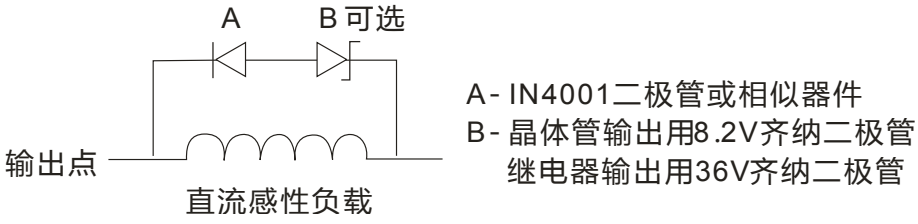


图 5-2 直流负载的抑制电路

❒ 交流输出和控制交流负载的继电器输出

交流输出对大部分应用都有内部保护，因为继电器可用于 DC 或 AC 负载所以不提供内部保护。

图 5-3 给出了交流负载抑制电路的一个实例。在大多数的应用中，附加的金属氧化物可变电阻(MOV)可以限制峰值电压，从而保护 CTH300-C 系列 PLC 内部电路。要确保 MOV 的工作电压比正常的线电压至少高出 20%。

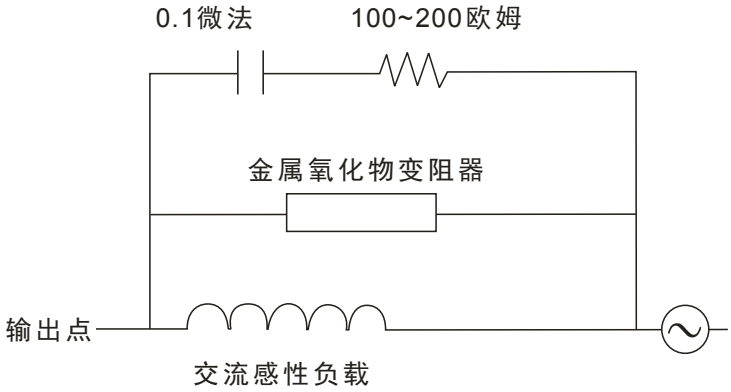
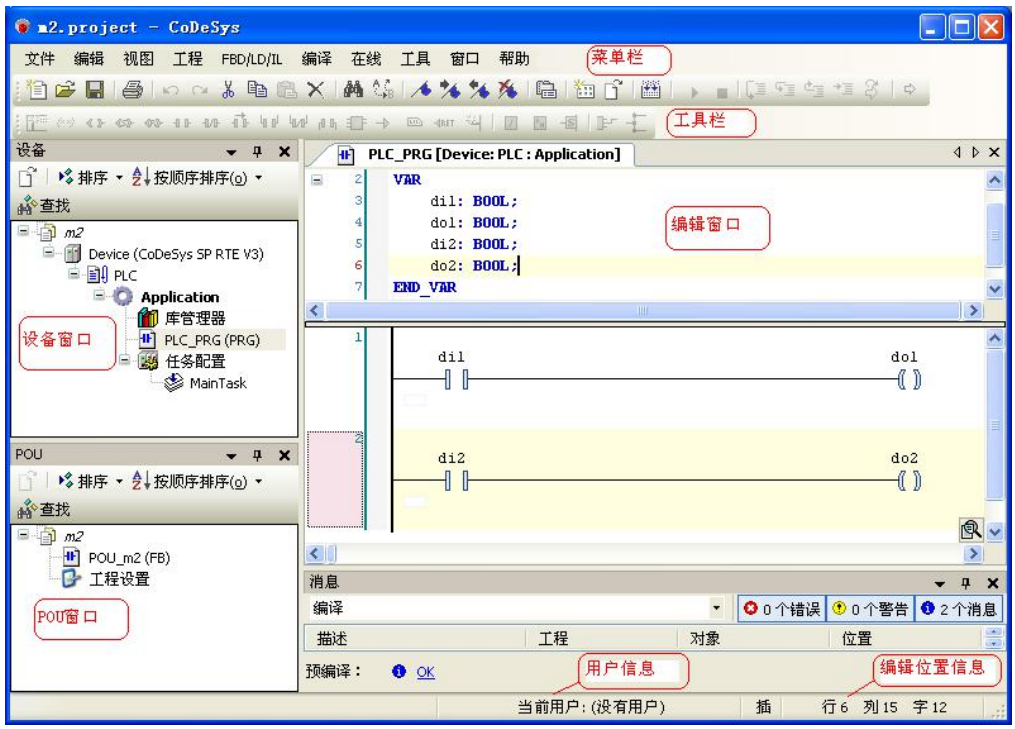


图 5-3 交流负载的抑制电路

6 CODESYS 的结构

CODESYS 是一种与设备无关的 PLC 编程系统。CODESYS 不仅支持所有符合 IEC 61131-3 标准的编程语言，还支持 C 语言。与 CODESYS 实时系统结合，可以在一个工程中对多个控制器设备进行配置。CODESYS 编程用户界面由“组件”排列组成，参考下图：



<备注> 本手册仅对 CODESYS 进行简单的应用介绍，有关 CODESYS 的详细使用，请参考 CODESYS 软件的在线帮助。

6.1 工程的构成

一个工程包含了 PLC 程序中的所有对象，以工程文件命名存储。工程中包含下列对象：POU，数据类型，可视化界面，资源及库。

POU

POU 是指“程序组织单元”，可以是程序、功能块或函数类型，它们可以随时增补。

每一个 POU 都包含一个局部和主题声明定义，主体部分可以用 IEC 的语言来编写，这些语言包括指令列表（IL），结构化文本（ST），顺序功能图（SFC），功能模块图（FBD），梯形图（LD）或连续功能图表（CFC）。

CODESYS 支持所有 IEC 标准的 POUs，如果在工程文件中使用这些 POU，必须在工程文件中包含标准库文件 standard.lib。POUs 可以调用其它的 POUs，但递归调用是不允许的。

函数

一个函数是一个 POU，它正确地产生一个数据元素（可以包含若干元素，比如，字段或结构体），在处理过程中，可以作为文本化语言表达式中的一个操作数来调用它。在声明一个函数的时候，一定要给它一个类型，这就是说，在函数名后面加上一个冒号然后跟一个数据类型。

一个正确的函数声明可以参考下面的例子：

FUNCTION Fct: INT

另外，必须分配给函数一个结果，即把函数名作为一个输出变量。

函数的声明从关键字 FUNCTION 开始。

在 IL 中函数的调用被安排在单步操作或单个转换之内。在 ST 中一个函数的调用可以作为表达式中的操作数。下面是一个在 IL 中一个函数带有三个输入变量并返回前两个变量的乘积与第三个变量相除的结果。

功能块

一个功能块是一个 POU，在程序中提供一个或多个值。与函数相反，一个功能块没有返回值。功能块的声明用关键字 FUNCTION_BLOCK 开始，可以创建功能块的备份或实例。

程序

一个程序是一个 POU，它在操作过程中返回几个值，程序在工程文件中是全局的。程序的所有最终值将保留直到下一次程序运行。

PLC_PRG

PLC_PRG 是一个特殊的预定义的 POU，每一个工程文件中必须包含一个这样的特殊的程序。实际上这个 POU 在每个控制循环中只调用一次。

在一个新工程文件创建之后，将首次使用“Project（工程）”→“Object Add（添加对象）”命令，在 POU 的对话框的缺省项目是一个名为 PLC_PRG 的程序类型的 POU。不能更改这些默认的设置。

如果定义了任务，那么工程中可以不包含 PLC_PRG，因为在这种情况下，程序的时序依赖于任务的分配。

<注意> 不要删除或者重命名 POU PLC_PRG(假如没有使用任务配置)，PLC_PRG 是一个单任务程序中的主程序。

动作

动作能够被定义并分配给功能块和程序，动作表现为一个更进一步的执行，它可以用其它的语言进行创建，每一个动作都有一个名称。

每一个动作都是和功能块或者程序中的数据一起工作的，动作使用相同的输入/输出变量和局部变量来执行。

资源

你需要用资源来配置和组织的工程文件和追踪变量的值。

- 全局变量用于全部工程文件或网络中。
- 库管理器用于添加库文件到工程中。
- 日志文件记录在线期间的动作。
- PLC Configuration 配置可编程控制器的硬件。
- 任务配置通过任务划分来引导程序的工作。
- 监视和接收管理器来显示变量值和设置默认变量值。

- 目标系统设置用来选择和在必要时进行目标系统的最终配置。
- 工作区作为工程选项的映像。

在 CODESYS 中构建工程需要的目标系统和目标系统设置，也可能用到下列资源：

- 用于变量值图形显示的采样追踪。
- 用于在同一个网络中与其它控制器交换数据的参数管理器。
- 作为控制器监视的 PLC 浏览器。
- 工具，与目标系统相关，在 CODESYS 内外调用外部工具程序。

库文件

可以在工程文件中包含一系列的库文件，可以象使用自定义的变量一样使用库文件的 POU，数据类型，和全局变量。库文件中的 standard.lib 和 util.lib 是可以自由调用的程序单元。

数据类型

参照标准的数据类型，用户可以定义自己的数据类型，可以建立结构体枚举类型和引用类型。

可视化界面

CODESYS 提供了可视化界面，因此可以显示工程的变量，通过可视化的帮助可以在离线的情况下绘制几何图形，在联机模式下能够按照特定变量的值而改变他们的形状，颜色和文本输出。

可视化的界面可以用作带 CODESYS 的 HMI 的 PLC 实用操作接口，或者作为一个网页或目标系统显示，通过因特网或 PLC 直接可视化。

6.2 语言

CODESYS 提供了对应的编辑器，支持所有在 IEC_61131 标准中规定的编程语言：

- 文本形式的语言：指令表（IL）、结构化文本（ST）
- 图形化的语言：顺序功能图表（SFC）、功能模块图（FBD）、梯形图（LD）、基于功能模块图的连续功能图表（CFC）

6.2.1 指令表（IL）

指令表中包含一系列的指令，依赖于操作的类型，每一条指令在一个新行开始并且包含运算符号和一个或多个用逗号隔开的操作数。在一个指令前面，还可以有一个标号，后缀一个冒号。注释部分在一行的最后，指令与指令之间可以插入空行。

在指令列表中将用到下面的操作符和限定符：

限定符：

C 与操作符 JMP, CAL, RET 连用：当前面的表达式处理的结果为 TRUE 时，才执行此指令。

N 与操作符 JMPC, CALC,RETC 连用：当前面的表达式处理的结果为 FALSE 时，才执行此指令。

N 用于其它情况：取操作数的反（不包括累加器）。

下面是操作符和它们可能的限定符以及相关的意义：

操作符及限定符意义	
LD N	使当前的值等于操作数

ST N	在操作数的位置保存当前值
S	当前的值为 TRUE 时，把布尔型操作数置为 TRUE
R	当前的值为 TRUE 时，把布尔型操作数置为 FALSE
AND N，（	位逻辑运算符号“与”
OR N，（	位逻辑运算符号“或”
XOR N，（	位逻辑运算符号“异或”
ADD （	加法
SUB（	减法
MUL（	乘法
DIV（	除法
GT（	>
GE（	>=
EQ（	=
NE（	<>
LE（	<=
LT（	<
JMP CN	跳转到标号
CAL CN	调用程序功能块
RET CN	离开 POU 并返回到调用的地方
）执行延时操作	

单击这里可以得到所有 IEC 操作符的列表。

例子：使用限定符编写的程序

LD TRUE (* 把 TRUE 加载到累加器中*)

ANDN BOOL1 (* 执行 AND 和 BOOL1 变量的取反后“与” *)

JMPC mark (* 当上面的结果为 TRUE 时，跳转到标号“mark”处*)

LDN BOOL2 (* 保存 BOOL2 的反 *)

ST ERG (* 把 BOOL2 保存在 ERG*)

Lable:

LD BOOL2 (* 保存 BOOL2 的值 *)

ST ERG (*把 BOOL2 保存在 ERG*)

在 IL 中也可以在操作之后放一个圆括号。圆括号内的值被认为是一个操作数。例如：

LD 2

MUL 2

ADD 3

Erg

这里 Erg 的值为 7，但是如果加一个圆括号：

LD 2

MUL (2

ADD 3

)

ST Erg

Erg 的结果是 10，当到达")"时操作 MUL 才开始计算；此时操作数 5 计算 MUL。

6.2.2 结构化文本（ST）

结构化文本中包含一系列的指令，这些用高级语言编写的指令能够被执行（例如 IF……THEN……ELSE）或者在循环（WHILE……DO）。

表达式

表达式是一个在运算后返回一个值的结构。表达式由运算符和操作数组成，操作数可以是常量、变量、函数调用或其它表达式。

表达式的计算

依照一定的规则来处理操作数可以计算出表达式的值，约束力最高的运算符首先参加运算，然后是约束力稍高的运算符，直到所有的运算符都被处理为止。相同约束力的运算符号的处理是从左到右的顺序。

下面是结构文本中运算符号约束力的级别排列：

操作	符号	约束力
放入圆括号	(表达式)	最强的约束力
函数调用	函数名（参数列表）	
求幂	EXPT	
取反	-	
乘法	*	
除法	/	
取模	MOD	
加法	+	
减法	-	
比较	<,>,<=,>=	
等于	=	
不等于	<>	
布尔运算“与”	AND	
布尔运算“异或”	XOR	
布尔运算“或”	OR 最弱的约束力	

下面这些是结构化文本中的其它指令，和例子一起安排在一个表中。

指令类型	例子
赋值	A：=B； CV ： = CV + 1 ； C：=SIN(X)；
调用一个功能块并使用功能块输出	CMD_TMR(IN ： = %IX5, PT ： = 300) ； A：=CMD_TMR.Q
RETURN	RETURN ；
IF	D：=B*B ； IF D<0.0 THEN C：=A； ELSIF D=0.0 THEN

	C：=B； ELSE C：=D； END_IF；
CASE	CASE INT1 OF 1：BOOL1：= TRUE； 2：BOOL2：= TRUE； ELSE BOOL1：= FALSE； BOOL2：= FALSE； END_CASE；

FOR	J：=101 ；
	FOR I：=1 TO 100 BY 2 DO
	IF ARR[I] = 70 THEN
	J：=I；
	EXIT；
	END_IF；
	END_FOR；

WHILE	J：=1 ； WHILE J<= 100 AND ARR[J] <> 70 DO J： =J+2； END_WHILE；
REPEAT	J：=-1 ； REPEAT J：=J+2； UNTIL J= 101 OR ARR[J] = 70 END_REPEAT；
EXIT	EXIT ；
空指令	；

赋值操作符

赋值符号左边是一个操作数（变量，地址），“：=”右边是赋予它的表达式的值，例如：

Var1 ：=Var2*10

在运算结束后，变量 Var1 就得到了 Var2 的 10 倍值。

在结构化文本中调用功能块

通过写功能块的实例名，以及随后在括号中给参数分配值来调用一个功能块。在下面的例子中，通过给两个参数 IN 和 PT 赋值来调用一个定时器，然后结果变量 Q 的值赋予变量 A。

RETURN 指令

返回指令可以用来按照条件离开一个 POU（ POU）。

IF 指令

IF 指令可以检验一个条件，根据这个条件，执行指令。

语法：

IF <Boolean_expression1> THEN

<IF_instructions>

{ELSIF <Boolean_expression2> THEN

<ELSIF_instructions1>

.

.

ELSIF <Boolean_expression n> THEN

<ELSIF_instructions n-1>

ELSE <ELSE_instructions>}

END_IF;

在{}中的部分是可选的。

如果布尔运算表达式<Boolean expression>返回 TRUE，只有 if 指令部分执行，其它部分不执行。否则，布尔运算表达式从<Boolean expression 2>开始，一个接一个的计算，直到某个布尔表达式返回为 TRUE。然后，在这个布尔运算表达式 2 之后，ELSE 或 ELSE IF 之前的部分被计算。

如果没有任何一个布尔运算表达式返回 TRUE，那么只计算 ELSE 下的指令。例如：

IF temp<17

THEN heating_on ：= TRUE；

ELSE heating_on ：= FALSE；

END_IF；

这里当温度降到 17 度以下时加热开始，否则保持关闭状态。

CASE 指令

使用 CASE 指令，可以在一个结构中，用同一个条件变量组合多个有条件判断的指令。句式：

CASE <Var1> OF

<Value1>: <Instruction 1>

Languages...

2-14 CoDeSys V2.3

<Value2>: <Instruction 2>

<Value3, Value4, Value5>: <Instruction 3>

<Value6 .. Value10>: <Instruction 4>

...

<Value n>: <Instruction n>

ELSE <ELSE instruction>

END_CASE;

CASE 指令根据下面的模式来处理：

如果变量 Var1 有值 Value1，那么执行指令 Instruction1。

如果变量 Var1 不是所指定的值，那么执行 ELSE Instruction。

如果有多个变量值要执行同一个指令，那么这些条件执行一个公共指令。

如果对于一个变量在一个值的范围内执行同一个指令，那么在初始值和最后值之间用两个句点隔开，所以可以规定公共条件。

例如：

```
CASE INT1 OF
1, 5: BOOL1 ：= TRUE;
BOOL3 ：= FALSE;
2: BOOL2 ：= FALSE;
BOOL3 ：= TRUE;
10..20: BOOL1 ：= TRUE;
BOOL3：= TRUE;
ELSE
BOOL1 ：= NOT BOOL1;
BOOL2 ：= BOOL1 OR BOOL2;
END_CASE;
```

FOR 循环

通过 FOR 循环程序可以编写重复执行的处理过程。

句式：

```
INT_Var :INT;
FOR <INT_Var> ：= <INIT_VALUE> TO <END_VALUE> {BY <Step size>} DO
<Instructions>
END_FOR;
```

{}内的部分是可选的。

只要计数器 INT_Var 不大于 END_VALUE，指令 Instructions 就一直执行，在执行 Instructions 之前首先检查计数器的值，如果 INIT_VALUE 比 END_VALUE 大的话 Instructions 将不在执行。

当 Instructions 执行后，INT_Var 通常要增加一个 Step size，Step size 可以是任何整型值，如果没有 Step size，它将设置为 1，当 INT_Var 大到一定值时，循环结束。

例如：

```
FOR Counter：=1 TO 5 BY 1 DO
Var1：=Var1*2；
END_FOR；
Erg：=Var1；
```

我们假设 Var1 的默认值是 1，那么在循环结束后它将得到值 32。

<注意> END_VALUE 一定不要大于等于与计数器 INT_VAR 的极限值，例如：如果变量计数器是一个 SINT 类型，并且 END_VALUE 为 127，那么这将是一个死循环。

WHILE 循环

WHILE 循环可以象 FOR 循环那样使用，不同之处在于 WHILE 循环的退出条件可以是任何布尔型表达式，当条件满足时，就会执行循环。句式：

```
WHILE <Boolean expression>
```

<Instructions>

END_WHILE;

只要 Boolean_expression 返回 TRUE，那么就重复执行 Instructions，如果 Boolean_expression 在首次计算出 FALSE，那么指令将不再执行，如果 Boolean_expression 从不出现 FALSE，Instructions 将没完没了的重复执行。

<注意> 程序员必须保证不出现死循环，这可以通过改变循环中指令部分的条件来实现，例如：可以通过计数器增加或减少。

例如：

WHILE counter<>0 DO

Var1 ：= Var1*2;

Counter ：= Counter-1;

END_WHILE

对于 WHILE 和 REPEAT 循环，在循环之前不必知道循环的次数，从这个意义上来说，这两种循环要比 FOR 要强大一些。

因此在此类情况下，可以用这两种循环。如果循环数比较明确，那么因为 FOR 没有死循环反而更好应用。

REPEAT 循环

REPEAT 循环和 WHILE 循环的不同之处在于它的中断条件是在循环执行之后才被检查，这就是说，循环至少要执行一次，不管中断是什么条件。

句式：

REPEAT

<Instructions>

UNTIL <Boolean expression>

END_REPEAT;

Instructions 一直执行到 Boolean expression 返回 TRUE，如果 Boolean expression 第一次就赋予真值，Instructions 只执行一次，否则 Instructions 将重复执行将会导致时间延迟。

<注意> 程序员可以通过改变循环中指令部分的条件来保证没有死循环出现，例如：可以通过计数器增加或减少。

例如：

REPEAT

Var1 ：= Var1*2

Counter ：= Counter-1；

UNTIL

Counter=0

END_REPEAT；

EXIT 指令

如果在 FOR，WHILE 或 REPEAT 循环中有 EXIT 指令，那么内循环就结束，不管中断是什么条件。

6.2.3 顺序功能图表（SFC）

顺序功能图表是基于图形化的语言，用它可以描述一个程序中不同动作的先后顺序。因为这些动作分配给单步元素，以及采用过度转换变量来控制处理的顺序。

步（Step）

用顺序功能图编写的 POU 包含了一系列的步，这些步之间是通过定向连接（转换条件）实现的。

有两种类型的步：

• 简单类型：每步包括一个动作和一个标记，这个标记用来表示此步是否激活。如果单步动作正在执行，那么在步的右上角方向会出现一个小三角形。

• IEC 类型：每步包含一个标记和一个或多个赋值的动作或布尔变量。相关的动作出现在步的右边。

动作（Action）

一个动作可以包含一系列的指令表或结构化文本指令，功能模块图或梯形图许多的网络，或者又包含另外顺序功能图。

在简单步中，动作经常是和步连接在一起的，为了能编辑一个动作，在步上双击鼠标或选择此步，再选择菜单命令“Extras”“Zoom Action / Transition”。另外，每一个步中只允许一个输入或输出动作。

IEC 步的动作是附加在顺序功能图- POU 内的对象管理器中，通过双击或者在它的编辑器中按 Enter 键可以加载它。也可以通过“Project”“Add Action”来创建一个新的动作。可以为一个 IEC 步分配最多九个动作。

进入和退出动作

可以额外的为一个步添加一个进入和退出的动作，在一个步激活后，一个进入动作只能执行一次。退出动作只在步失效之前执行一次。

带进入动作的步左下角一个“E”来表示，退出动作用右下角的“X”表示。

转换/转换条件

在步和步之间有所谓的转换。

转换条件的值必须是 TRUE 或 FALSE，因而它可以是一个布尔变量、布尔地址或布尔常量。在结构化文本句式（例如（I<=100）AND b）或者在任何一种期望的语言（参照‘Extras’‘Zoom Action/Transition’）中，它也能包括一系列有布尔结果的指令。转换中不能包括程序、功能块或赋值。

<注意> 除了转换外，也能用渐进模式跳到下一步，查看 SFCtip 和 SFCtipmode。

激活步

在调用顺序功能图的 POU 后，将首先执行初始化步的动作（被一个双边线包围）。正在执行的步动作，状态是激活的，在联机模式下，激活的步显示为蓝色。在一个控制循环中激活步的所有动作都将执行。所以，当激活步之后的转换条件是 TRUE 时，它之后的步被激活。当前激活的步将在下个循环中再执行。

<注意> 如果激活的步包含一个输出动作，譬如它下面转换条件是 TRUE，那么它只能在下个循环过程中执行。

IEC 步

在顺序功能图中可以使用标准的 IEC 步。

为了能使用 IEC 步，必须在工程文件中联接 Iesfc.lib 库文件。

一个 IEC 步中不能分配超过九个动作，IEC 的动作不象简单步那样固定地作为输入或输出到某个步的动作，而是和步分开存储并且能够在一个 POU 中重复使用多次。因此，它们必须用命令“Extras Associate action”和单个步联系在一起。

除了动作，布尔变量也能分配给步。

能够使用所谓的限定词来控制激活和未激活的动作和布尔变量。可能有时间延迟，如果一个动作依然激活中，而下一个步已经开始处理了，通过限定词 S（设置），可以取得并发的过程。

随着每一个顺序功能模块的调用，相关联的布尔变量被设置或复位，也就是说，随着每一次调用，这个值将在 TRUE 到 FALSE 之间来回变化。

IEC 步的关联动作在步右边的两长方形中表示，左边的区域包含了限定词，可能带有时间常量，右边的区域包含了动作名和各自的布尔变量名。

限定符

为了关联动作和 IEC 步，用到下面的限定词

N	非存储	动作和步一起激活
R	复位	动作是未激活的
S	设置	动作被激活，再复位前保持激活状态
L	时间限制	动作激活一段时间，最大值和步激活时间一致
D	时间延迟	如果步仍然激活，动作在一定时间后激活，然后只要步是激活的，它就保持激活
P	脉冲	如果步激活，动作只执行一次。
SD	存储和时间延迟	在一定时间之后动作激活并保持激活状态到下一个复位开始。
DS	延迟和保持	只要步仍然激活并且保持到下一个复位开始，那么在一定时间后动作被激活
SL	保持和时间限制	动作激活并保持一段时间

限定符 L、 D 、 SD 、 DS 和 SL 需要一个 TIME 常量格式的时间值。

<注意> 当一个动作失去激活时，它会再执行一次。这就是说每个动作至少执行两次。

SFC 中的隐含变量

在 SFC 中使用一些隐含声明的变量。

每一个步都有一个标记，标记中存储着步的状态。对于 IEC 步来说，步的标记（激活或未激活）被称为<StepName>.x 或者对一个简单的步来说称为<StepName>。当关联的步激活的时候这个布尔变量值为 TRUE，反之则值为 FALSE。它能够用在 SFC 模块中的每一个动作和转换中。

可以通过查询<ActionName>.x 来查询一个 IEC 步是否激活。

隐含变量<StepName>.t 能够用来查询步激活的时间。

隐含变量也能够被其它程序访问，例如，boolvar1：=sfc1.step1.x; 这里 step1.x 是隐含布尔变量，它代表了 POUsfc1 中的 IEC 步 step1 的状态。

SFC 标志符

标志符用来控制 SFC POU 操作，它在工程运行期间隐含创建。为了能读这些标志符，必须定义合适的全局变量或局部变量。例如，如果在一个 SFC POU 中一个步激活的时间超过了它定义的

属性，那么就会设置一个标志符，通过用一个“SFCError”变量可以访问到这个标志符（此时 SFCError 得到真值）。

可以定义下列标志符变量：

SFCEnableLimit:这个变量的类型是布尔型，当它的值为 TRUE 时，这一步的超时将会注册进 SFCError，其它的超时将被忽略。

SFCInit:当这个布尔变量值为 TRUE 时，顺序功能图复位到初始状态，其它的 SFC 标志符也会被复位。初始步保持激活，直到变量值为 TRUE 时，才开始执行。只有当 SFCInit 被重新设置为 FALSE 时，模块才能正常工作。

SFCReset:这个布尔变量，与 SFCInit 很相似，不同之处在于，进一步的处理发生在初始化步的初始化之后，因而，例如 SFCReset 标志符可以在初始化步中被复位到 FALSE。

SFCQuitError:当这个布尔变量得到 TRUE 时，SFC 的执行将会停止，因此，在变量 SFCError 中一个可能超时将复位，当这个变量呈现 FALSE 时，激活步中的所有时间都会复位，先决条件是在 SFC 中预先定义 SFCError，同时注册任何超时设定的标志符。

SFCPause:当这个布尔变量值为 TRUE 时，SFC 图表就停止执行。

SFCError:当在 SFC 图表中有超时，这个变量得到 TRUE。如果在这个之后还有其它的超时发生，除非是这个变量已经复位，否则，这些状态将不会记录。如果想使用其它的时间控制标志符 (SFCErrorStep,SFCErrorPOU, SFCQuitError,SFCErrorAnalyzation) 的前提条件是定义 SFCError。

SFCTrans:当一个转换被激活时，这个布尔变量得到真值。

SFCErrorStep:这个变量是一个字符串变量，如果 SFCError 中注册了一个超时，这个变量将存储这个超时步的名字。前提条件是在 SFC 中已定义了注册任何超时的标志符 SFCError。

SFCErrorPOU:这个字符串变量包含了发生超时的模块名字。前提条件是在 SFC 中已定义了注册任何超时的标志符 SFCError。

SFCCurrentStep:这个字符串变量存储了那个被激活步的名字，它与时间监控无关。在仿真的情况下，此步存储在外部权限的分支中。如果一个超时发生，超时将不再注册，而且 SFCError 也不会复位。

SFCErrorAnalyzationTable:这是数组型变量 ARRAY [0..n]，它提供一个转换表达式的分析结果。对有影响的每个元素转换注册为 FALSE，以及上一步超时记录，所以要把下列信息写进 ExpressionResult 结构中，写进：名称，地址，注释，当前值。

最大可以容纳 16 个元素，因此，数组的范围从（0~15）。

ExpressionResult 结构和隐含使用的分析模块都是由 AnalyzationNew.lib 库文件提供的，分析模块也能够被其它的不用 SFC 编写的 POU 显式使用。

上一步的超时注册是分析一个转换表达式的先决条件，因此在这里必须有一个时间监视的应用，而且 SFCError 必须在声明窗口被定义。

SFCTip, SFCTipMode:这个布尔变量允许 SFC 的渐进模式。当用在 SFCTipMode=TRUE 来切换它时，如果 SFCTip 设置值为 TRUE 它只可能跳到下一个步，只要 SFCTipMode 是设置为 FALSE 时，它可能跳过转换。

可选分支

在 SFC 中可以定义两个或两个以上的可选择分支。每一个分支的开始和结束必须带有一个转换。可选分支可以包含并行的分支和其它的选择分支，一个可选分支开始于一个水平线并终止于一个水平线（选择结束），或是一个跳跃。

如果在可选分支开始行前面的步是激活的，每一个可选分支的首次变换将从左到右被计算，最先的转换将从左边转换条件为 TRUE 的开始，然后下面的步被激活。

并行分支

在 SFC 中可以定义两个或两个以上的分支为并行分支。每一个并行分支在开始和结束处必须有一个步。并行分支可以包含可选择的分支或其它并行分支，一个并行分支开始于一个双划线，结束于一个双划线或者一个跳跃，它能提供一个跳跃标识。

如果并行分支的先前步是激活的，并且这个步之后的变换条件值是 TRUE 时，那么并行分支的第一步激活。这些分支彼此并行处理。当所有并行步激活并且这些步之后转换条件为 TRUE 时，那么并行分之末端线后的那一步被激活跳转。

跳转是对在跳转符号下面指明的步名的一个连接。当在不允许创建向上或互相交叉联络的时候，必须使用跳转。

6.2.4 功能模块图（FBD）

功能模块图是一种基于图形的编程语言，它用一串网络来工作，每一个网络包含一个提供算术或逻辑表达式、功能块的调用、跳转或返回指令的结构。

6.2.5 连续功能图表（CFC）

连续功能图表编辑器不象功能模块图那样操作，但是可以自由放置元素，它允许使用反馈。

6.2.6 梯形图（LD）

梯形图也是一种基于图形化的编程语言，它接近于电子电路的结构，一方面，梯形图很适合构建逻辑开关，另一方面，它也能创建象 FBD 中的网络图，所以梯形图在控制调用其它 POU 的时候是很有用的。

梯形图包含了一系列的网络，左右两边各有一个垂直的电流线，网络图仅限制于左右两母线之间的范围内，在中间是由线圈触点和连接线组成的电路图。

每一个网络包含左边的一系列触点，这些触点根据布尔变量值的 TRUE 和 FALSE 来传递从左到右的开和关的状态。每一个触点是一个布尔变量，如变量值为 TRUE，电路从左到右通过连接线就连通。否则右边接收到“关”的值。

触点

在梯形图中的每一个网络图的左边都有触点（触点是用两个并行线| |来表示），它用来表示电路的“开”“关”状态。

这些状态与布尔变量 TRUE 和 FALSE 相一致。布尔变量属于每一个触点。如果变量值为 TRUE，那么状态可以通过连接线从左传到右边。否则，右边接收到的是“断开”。

触点可以并联使用，其中的一个并联分支必须传递“开”状态时，并联分支才能传递“开”。或者触点串联连接，此时，触点必须传递“开”状态时，最后的触点才传递“开”，这些与串并联连接的电路一致。

线圈

在梯形网络图的右边有一些所谓的线圈，它们用（）表示，并且只能通过水平线来连接。线圈传递从左到右的连接状态，并且复制状态到布尔变量中，可以描述入口线的状态为“开”（对应布尔变量的 TRUE）或者“关闭”状态（对应布尔变量的 FALSE）。

触点和线圈也可以取反值（在上例中的触头 SWITCH1 和线圈%QX3.0 是取反值）。如果触点取反值（在触头符号中用“/”来表示），然后它复制反后的值到相应的布尔型变量中。如果一个触点取反值，仅当相应的布尔变量取到 FALSE 时，电路才能连通。

梯形图中的功能块

可以在网络图中添加功能块和程序，但它们必须具有布尔型值的输入和输出，并且可以象触点那样用在梯形图的左边。

触点的设置/复位

触点可以被定义为设置/复位触点。设置触点在触点符号中用“S”表示，它从不覆盖相应的布尔变量中的 TRUE 值，也就是说，如果变量一旦设置为 TRUE，它将保持状态不变。

复位触点用“R”来表示，它从不覆盖相应布尔变量中 FALSE 值，如果变量已经设置了 FALSE，它将保持这种状态不变。

当使用梯形图时候，可以用触点开关的结果来控制其它的 POU。一方面可以用线圈把结果输出给全局变量，这个全局变量可以在其它的地方使用。也可以通过引入一个带 EN 输入的 POU 来直接在梯形网络图中插入可能的调用。

这些 POU 是完整的正常的操作数、功能、程序或功能块。它们都有一个附加的输入标志符 EN。EN 输入是一个布尔变量，只有当 EN 值为 TRUE 时，带有 EN 输入的 POU 才会被计算。

6.3 联机调试功能

采样追踪

采样追踪允许追踪变量的连续变化的值，它依赖于所谓的触发事件，触发事件是先前定义的布尔变量（触发变量）的上升沿或下降沿。CODESYS 允许对 20 个变量进行追踪，每一个变量可以追踪 500 个值。

调试

CODESYS 的调试功能使得错误很容易被找到。为了调试，运行命令“Project”“Options”，在自动弹出的创建选项的对话框中选择已经激活的选项。

断点

断点是程序处理过程中停止的位置，因而它可以在程序中的特定位置观察变量值的变化。

断点可以在编辑器中设置，在文本编辑器中断点在行的编号处设置，在连续功能图中和梯形图中是在网络编号处设置，在 CFC 中是在 POU 处设置，在 SFC 中是在步处设置，在功能模块图的实例中不能设置断点。

单步

单步是：

在指令表中：执行程序直到运行 CAL LD 和 JMP 命令。

在结构化文本中：执行下一条指令。

在功能模块图梯形图中：执行下一步网络。

在顺序功能图中：继续目前的动作直到下一个步开始。

通过一步一步的运行可以检查程序中的逻辑错误。

单循环

如果选择了单循环，每一个循环结束，执行也就结束。

联机模式下改变值

在操作过程中，变量可以设置为一个特定的值（write value），或者在每一个循环之后重新定义为特定的值（force value）。在联机模式下可以通过双击变量的值来改变它的值，布尔变量从 TRUE 变为 FALSE，或从 FALSE 变为 TRUE。对于每一种类型的变量都可以打开 Write Variable xy 对话框，在这里可以编辑变量的真实值。

监视

在联机模式下，所有的变量可以从控制器中读出并及时的显示。可以在定义和程序编辑器中找到这些显示。也可以在观察和接收器中读出变量的当前值，并且可以看到它们。如果要监视功能块的实例中的变量，相应的实例块必须已经打开。

仿真

在仿真模拟过程中，创建的 PLC 程序不在实际的 PLC 中运行，而是在 CODESYS 系统中的计算器中运行。所有的联机功能都是可用的。它允许在无需 PLC 硬件的情况下检测逻辑的正确性。

<注意> 外部库文件的 POU 是不能运行在仿真模式的。

日志

日志记录着用户的操作、内部进程、状态变换和联机模式处理过程中发生的意外的情形。它用来监视和跟踪错误。

6.4 标准化

IEC61131-3 是一种国际标准化的 PLC 编程语言。在 CODESYS 中提供的可编程语言符合标准化的要求。根据这个标准，一个程序包含以下元素：

- 结构
- POU
- 全局变量

通用的语言的元素在标识符、地址、类型、注释和常量部分中已经讲述。CODESYS 程序的处理是从一个特殊的 POU（POU）PLC_PRG 开始的，POU PLC_PRG 能调用其它的 POU。

7 使用指南

本章基于几个具体实例，介绍如何配置CTH300-C 系列运动控制器，以及与其它设备进行Modbus、CANopen、EtherCAT 通信等。

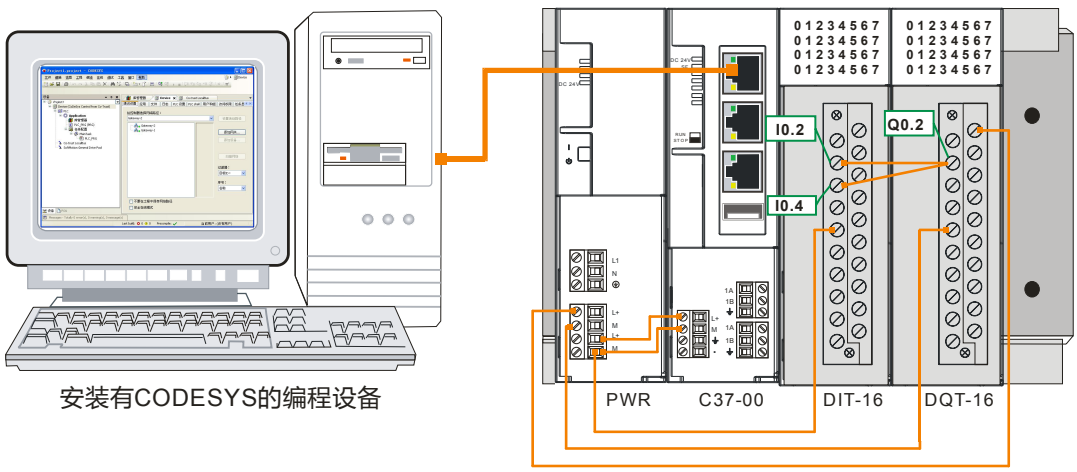
7.1 总线通信实例

本指南通过一个具体实例让大家了解如何使用 CODESYS 软件控制CTH300-C 系列运动控制器与扩展模块的总线通信。

7.1.1 示例组件

组件	功能
安装有 CODESYS V3.5 的编程设备	对 CTH300-C 系列运动控制器进行组态、编程和调试
装配导轨	CTH300-C 机架，用于固定系统中的各模块。
电源模块 PWR-02	提供 CTH300-C 系列运动控制器及其 24 VDC 负载电路。
C37-00 运动控制器	CPU 执行用户程序，向 CTH300-C 总线提供 5 V 电压，并通过以太网接口与其它模块进行通讯。
扩展模块 SM	SM 转换不同的过程信号电平，使其与 CTH300-C 系列运动控制器相匹配。 本例使用：数字量模块 DIT-16、DQT-16
标准网线	连接编程设备与C37-00

7.1.2 网络连接



7.1.3 操作步骤

1、为 C37-00、电源模块、数字量输入和输出模块接线

打开 C37-00、电源模块 PWR-02、数字量输入（DIT-16）和输出模块（DQT-16）的前面板，然后按照“7.1.2 网络连接”为它们接线。

警告

电源模块上电后或将电源电缆连接到主电源后不得触摸通电电缆，只有在关闭电源后才能进行接线工作！

2、连接电缆

参考“7.1.2 网络连接”连接各设备，连接操作描述如下：

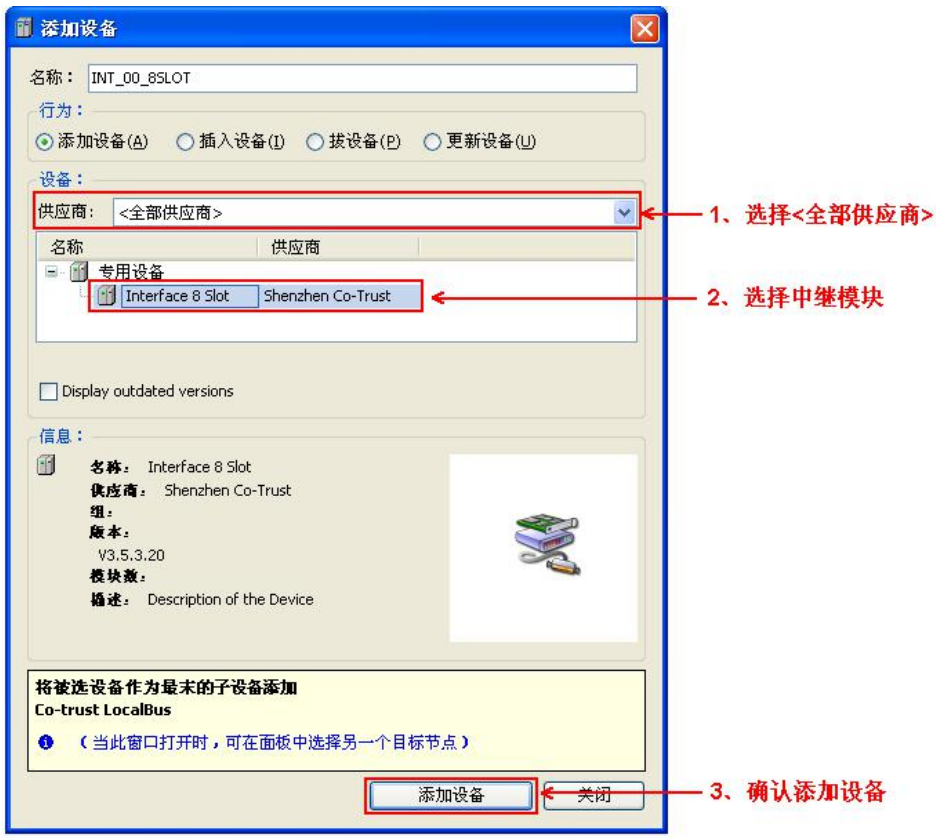
1）使用标准网线连接 PC 与 C37-00

2）DIT-16 通过总线方式连接至 C37-00

3）DIQ-16 通过总线方式连接至 DQT-16

3、在 CODESYS 中进行组态

1）在设备视图选中“Co-trust LocalBus”并点击右键选择“添加设备”，即可在弹出的对话框中选择并添加中继模块，参考如下示意图：



2）参照以上操作，在设备视图中右键点击中继模块选择“添加设备”，即可在“添加设备对话框”中选择添加扩展模块 DIT-16 和 DQT-16。或者右键点击中继模块选择“扫描设备”，与 C37-00 相连的扩展模块即会显示在中继模块下方。

<备注> C37-00 支持的扩展模块有中继模块、数字量输入/输出模块、模拟量输入/输出模块、温度模块、CAN 主站模块，扩展模块的规格信息请参考“C 扩展模块技术规范”。

4、参考章节“2.3 设置通信”设置 C37-00 与上位机的通信

5、运行、调试与监控

1）选择菜单项“在线”→“登录到...”使应用程序与 C37-00 建立起连接，并进入在线状态。然后，选择菜单项“调试”→“启动”使 C37-00 中的应用程序开始运行，此时即可对当前工程进行监控和调试操作。

2）在 CODESYS 软件的 Internal I/O 映射表中为%QX0.2 写入 1，则 DQT-16 模块的输出点 Q0.2 被点亮。同时，DIT-16 模块的输入点 I0.2 和 I0.4 被点亮，在 DIT-16 模块的 Internal I/O 映射表中查看%IX0.2 和%IX0.4 的值变为 1。

7.2 Modbus TCP 通信实例

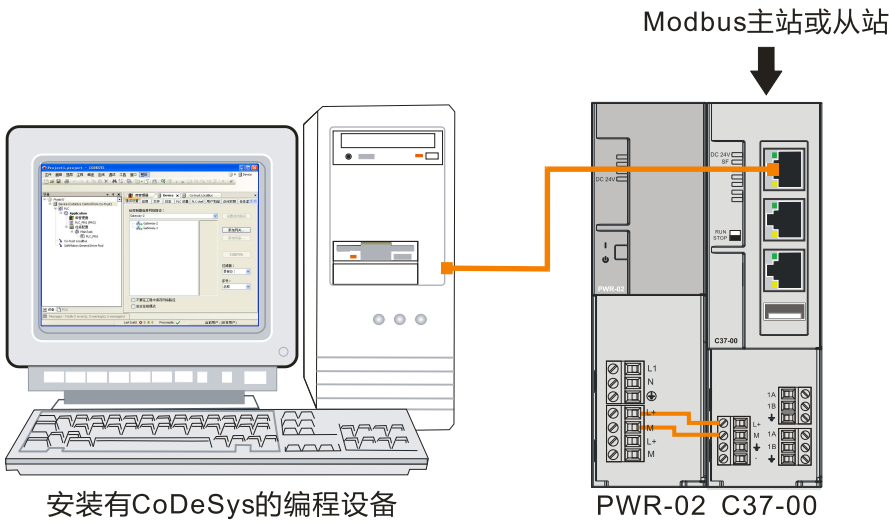
本指南通过一个具体实例来引导您建立一个应用程序，通过本例，您将熟悉 CTH300-C 系列运动控制器的 Modbus TCP 的通信功能。

7.2.1 示例组件

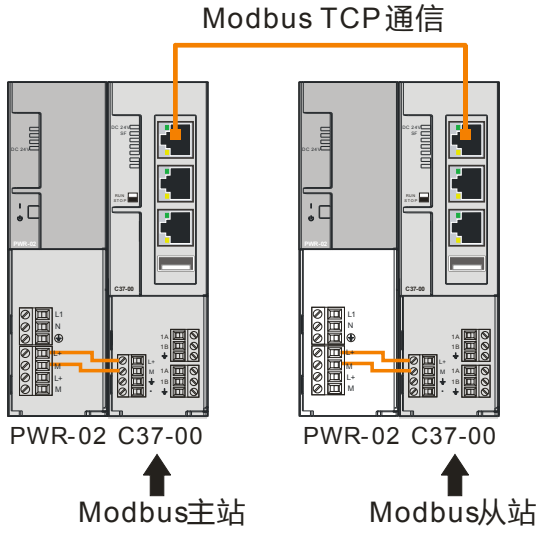
组件	功能
安装有 CODESYS V3.5 的编程设备	对 CTH300-C 系列运动控制器进行组态、编程和调试
装配导轨	CTH300-C 机架，用于固定系统中的各模块。
电源模块 PWR-02	提供 CTH300-C 系列运动控制器的 24VDC 工作电源
C37-00 运动控制器	执行用户程序，为总线提供 5 V 电压，并通过以太网接口与其它模块进行通讯。
标准网线	连接编程设备与C37-00

7.2.2 网络连接

使用标准网线连接编程设备与 C37-00，然后通过编程设备分别对作为 Modbus 主站、从站的 C37-00 进行编程：



使用标准网线将作为 Modbus 主站和 Modbus 从站的两台 C37-00 连接起来，继而可进行 Modbus TCP 通信：



7.2.3 操作步骤

1、连接电缆

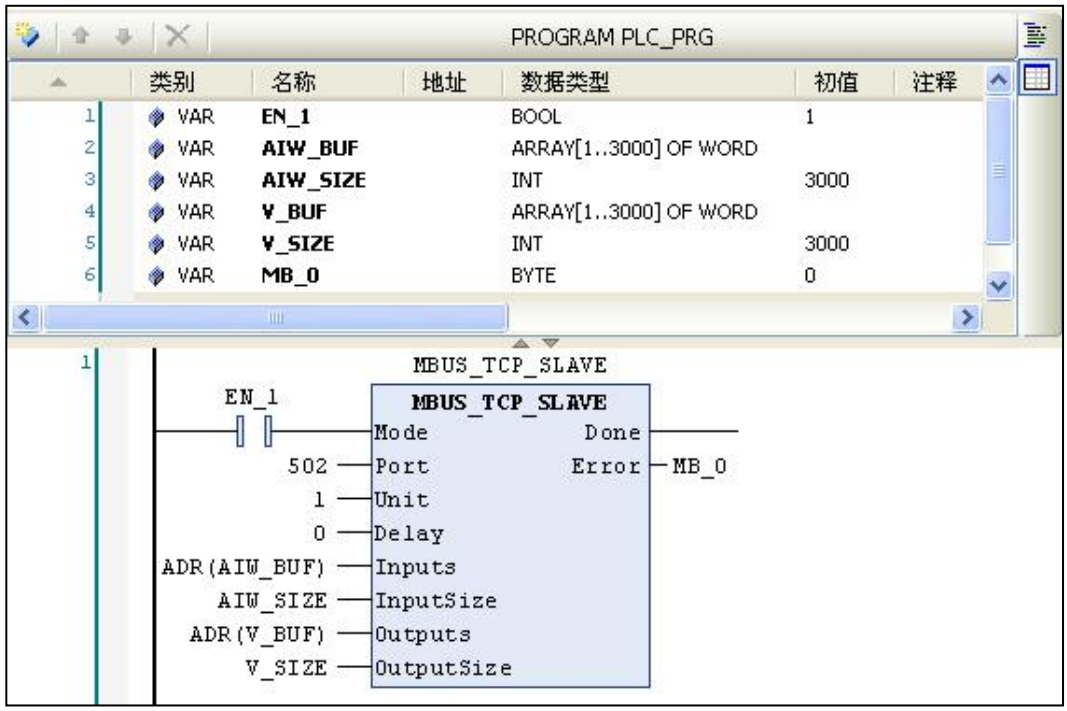
参考“7.2.2 网络连接”，使用一条标准网线连接 PC 与 Modbus 从站。

2、对 Modbus 从站 C37-00 进行编程

1）在 PLC SHELL 中输入命令，如“setip 192.168.0.1”按 ENTER 键可设置 IP 地址。

2）在 PLC_PRG(PRG)中为 Modbus 从站编程

以下指令为“CO-TRUST MODBUS LIBRARY”库文件中的“Modbus_TCP_Slave”库指令：



3）编译并下载程序到 Modbus 从站设备中

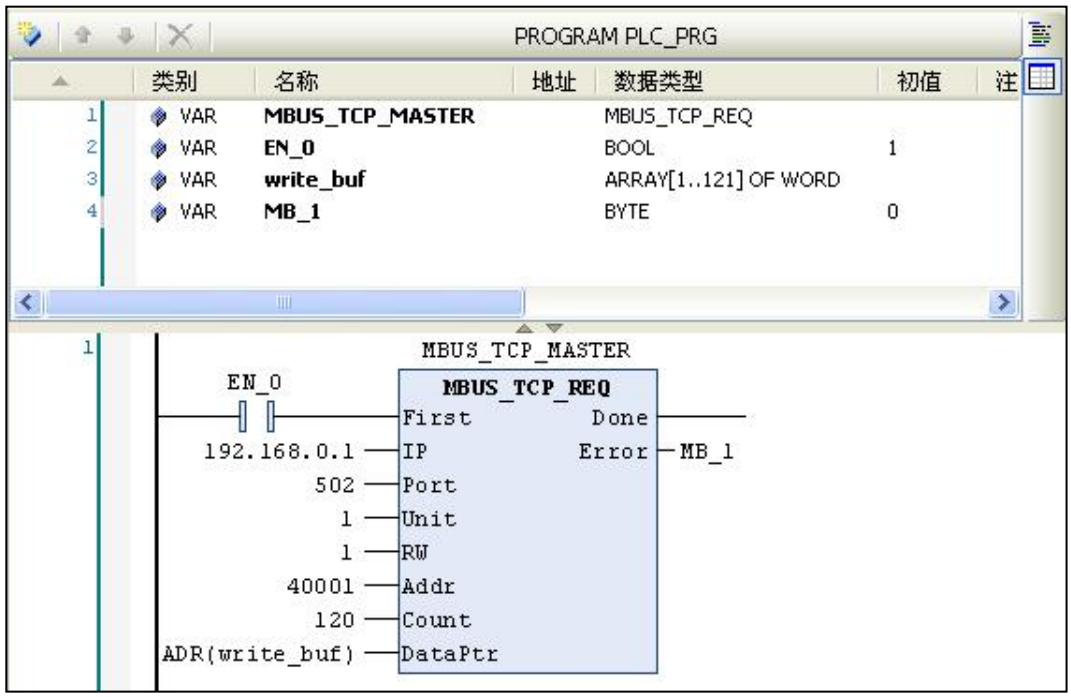
3、连接电缆

参考“7.4.2 网络连接”，使用标准网线连接 PC 与 Modbus 主站。

4、对 Modbus 主站 C37-00 进行编程

1）在 PLC_PRG(PRG)中为 Modbus 主站编程

以下指令为“CO-TRUST MODBUS LIBRARY”库文件中的“Modbus_TCP_REQ”库指令



5、Modbus 主站与 Modbus 从站互联通信

1）给 Modbus 主站、从站接通电源

2）使用标准网线连接 Modbus 主站与从站，即它们开始进行 Modbus TCP 通信。

7.3 Modbus RTU 通信实例

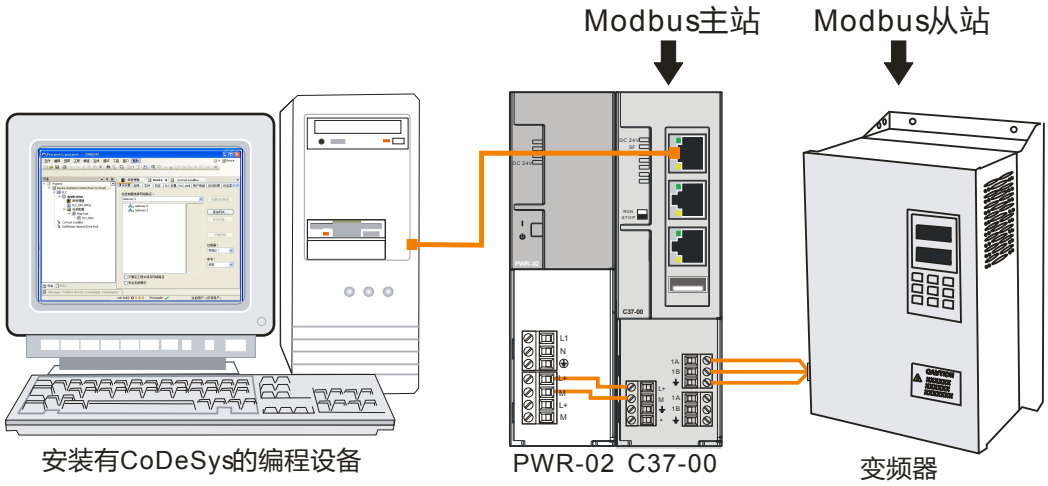
本指南通过一个具体实例来引导您建立一个应用程序，通过本例，您将熟悉 C37-00 的 Modbus RTU 的通信功能。

7.3.1 示例组件

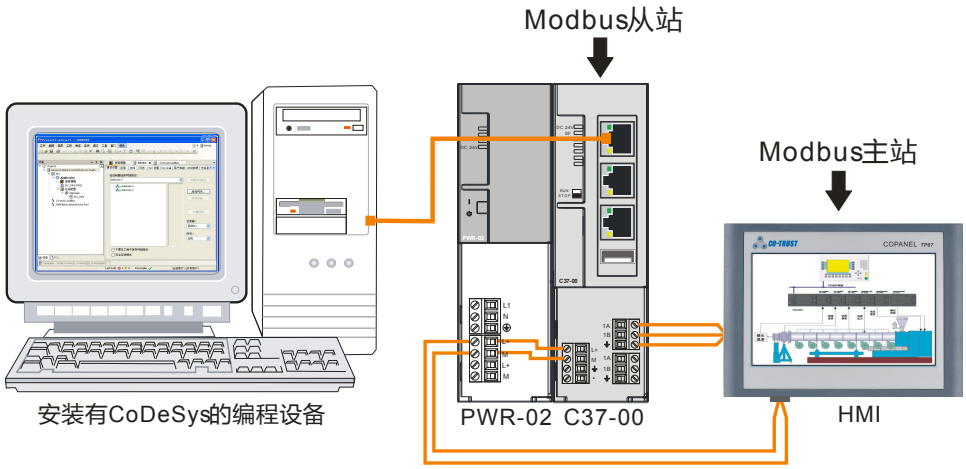
组件	功能
安装有 CODESYS V3.5 的编程设备	对 CTH300-C 系列运动控制器进行组态、编程和调试
装配导轨	用于固定系统中的各模块。
电源模块 PWR-02	提供 CTH300-C 系列运动控制器及 HMI 的 24VDC 工作电源
C37-00 运动控制器	CPU 执行用户程序，向 CTH300-C 总线提供 5V 电压，并通过以太网接口与其它模块进行通讯。
HMI	Copanel 系列 HMI，作为 Modbus 主站与 C37-00 通信
变频器	支持 Modbus 协议的变频器，作为 Modbus 从站与 C37-00 通信
标准网线	连接编程设备与C37-00

7.3.2 网络连接

◆ C37-00 作为主站与变频器通信



◆ C37-00 作为从站与 HMI 通信



7.3.3 操作步骤

1、为 C37-00、电源模块、HMI 和变频器接线

参考“7.3.2 网络连接”分别为 C37-00、电源模块 PWR-02、HMI 和变频器接线。

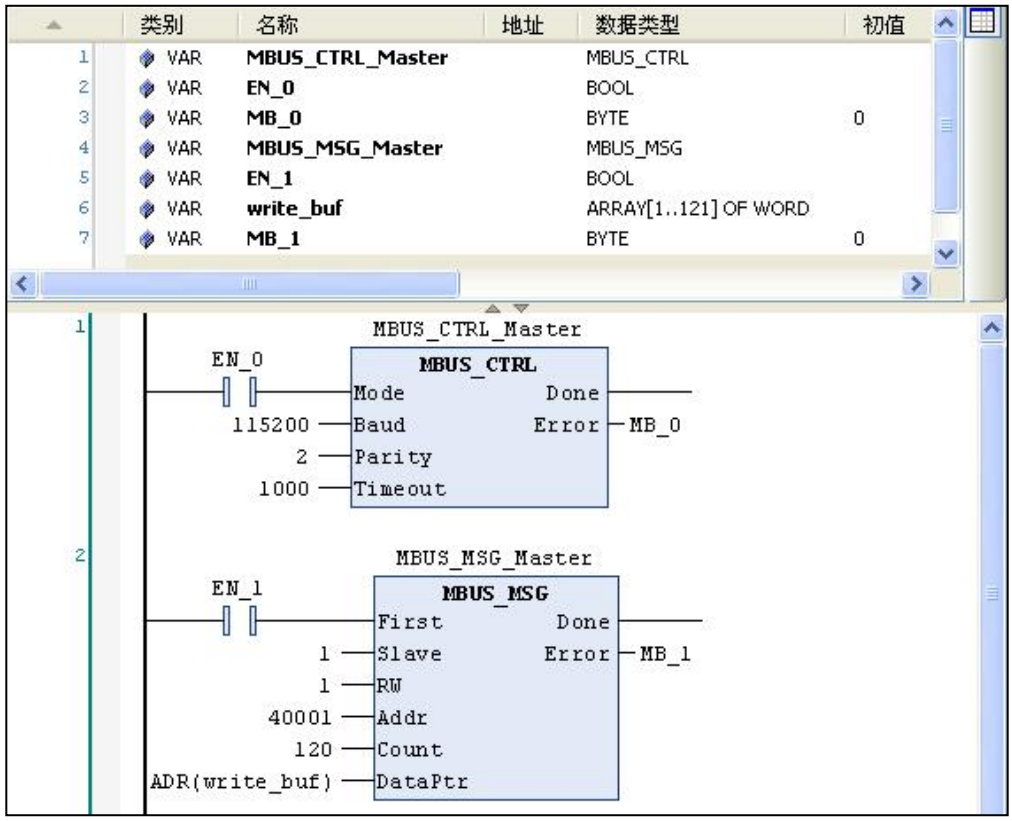
2、连接电缆

1）使用一条标准网线连接 PC 与 C37-00。

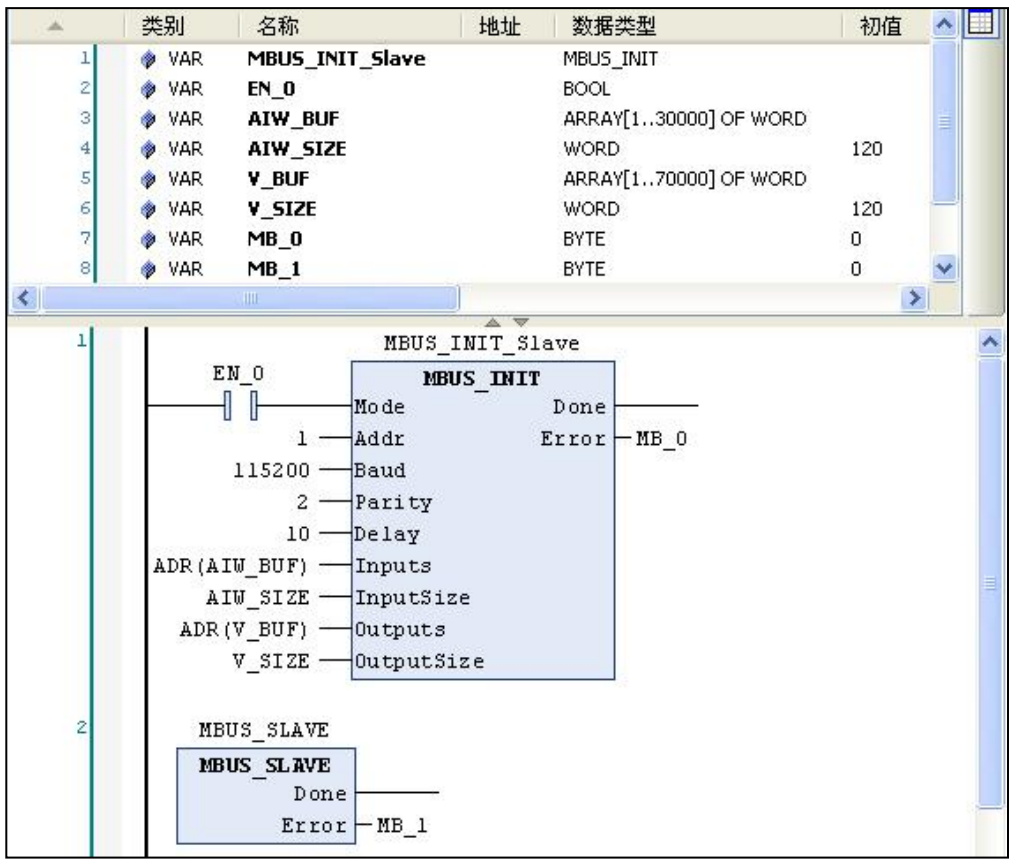
2）当 C37-00 作为 Modbus 主站与变频器通信时，将 C37-00 的 RS485 接线端子的 1A、1B 连接至变频器的 Modbus 通信口。

3）当 C37-00 作为 Modbus 从站与 HMI 通信时，将 C37-00 的 RS485 接线端子的 1A、1B 连接至 HMI 的 Modbus 通信口。

3、当 C37-00 作为 Modbus 主站与变频器通信时，为 C37-00 编程如下：



4、当 C37-00 作为 Modbus 从站与 HMI 通信时，为 C37-00 编程如下：



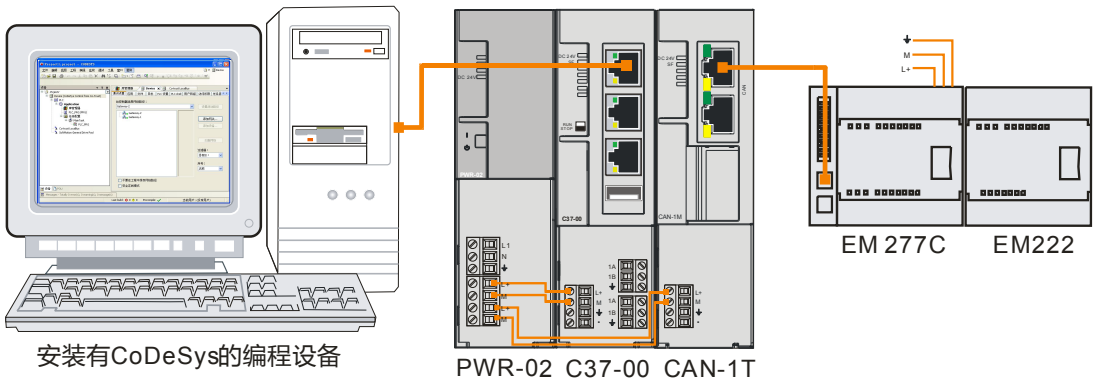
7.4 CANopen 通信实例

本指南通过一个具体实例使您熟悉 C37-00 通过 CAN-01 主站模块实现 CANopen 通信功能。

7.4.1 示例组件

组件	功能
安装有 CODESYS V3.5 的编程设备	对 CTH300-C 系列运动控制器进行组态、编程和调试
装配导轨	用于固定系统中的各模块。
电源模块 PWR-02	提供 CTH300-C 系列运动控制器及其 24 VDC 负载电路。
C37-00 运动控制器	执行用户程序，为总线提供 5 V 电压，并通过以太网接口与其它模块进行通讯。
CAN 主站模块 CAN-1T	作为 CAN 主站，与 CAN 从站进行通信
EM277C	作为 CAN 从站
扩展模块	CAN 从站的扩展模块，挂接在 EM277C 后面 本例使用：数字量模块 EM222-1HH32
标准网线	连接编程设备与C37-00 连接CAN-1T与CAN从站

7.4.2 网络连接



注意：如果采用 C35-01/C36-01/C37-01 作为主站，则无需 CAN 主站模块，直接通过其 CAN 接口连接 CAN 从站即可。

7.4.3 操作步骤

1、为各设备接通电源

打开电源模块 PWR-02、C37-00、EM277C 及 CAN-1T 的前面板，然后参考“7.4.2 网络连接”为它们接通电源。

2、连接电缆

参考“7.4.2 网络连接”连接各设备，具体操作如下：

1）使用一条标准网线连接 PC 与 C37-00

2）CAN-1T 主站模块通过总线方式连接至 C37-00

3）使用一条标准网线连接 CAN-1T 主站模块与 EM277C

4）数字量模块 EM222 通过总线方式连接至 EM277C

3、在 CODESYS 中进行 CANopen 组态

1）添加 CANbus

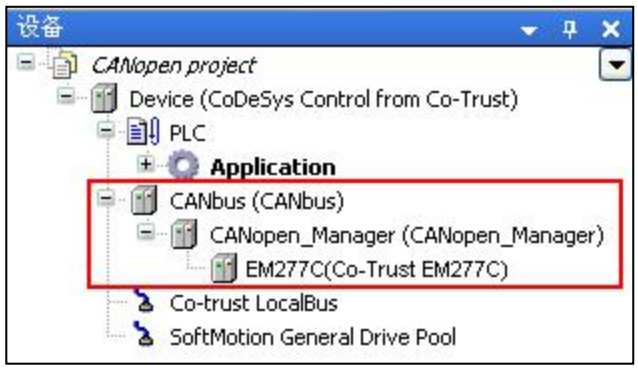
在设备视图中右键点击“Device(CodeSys control from Co-trust)”选择“添加设备”，即可在弹出的对话框中选择添加 CANbus：供应商选择“<全部供应商>”、现场总线选择“CANBUS”作为 CAN 总线。

2）添加 CAN 主站

在设备视图中选中已经成功添加的“CANBUS”并右键选择“添加设备”，即可在“添加设备”对话框中添加 CAN 主站：现场总线→CANopen→CANopen 管理器→CANopen_Manager。

3）添加或扫描 CAN 从站

在设备视图中选中已经成功添加的“CANopen_Manager”并右键选择“添加设备”，即可在“添加设备”对话框中添加 CAN 从站：现场总线→CANopen→远程设备→EM277C；或者右键点击“CANopen_Manager”选择“扫描设备”，即可将从站设备（EM277C）显示在 CAN 主站（CAN-1T 模块）下方。



4）CANbus 设置

CANbus 选项卡：本例中“网络”设置为 0、“波特率”设置为 1000000。

5）配置 CAN 主站

CANopen_Manager 主站中的 NODE_ID 设为 127。

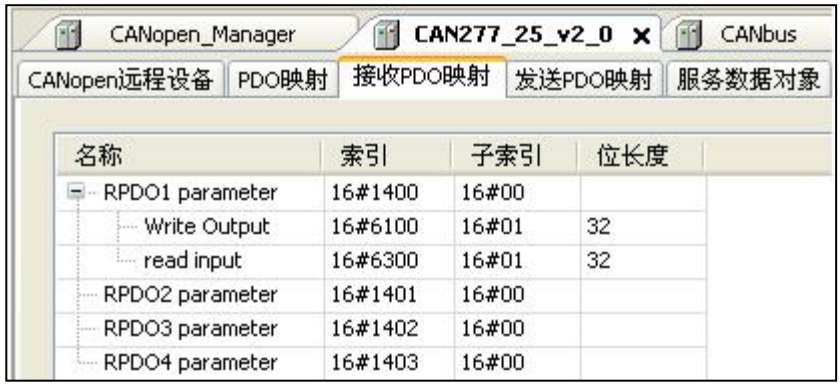
6）配置 CAN 从站（EM277C）

“CANopen 远程设备”选项卡：节点 ID 设为 4、勾选“使能专家设置”。

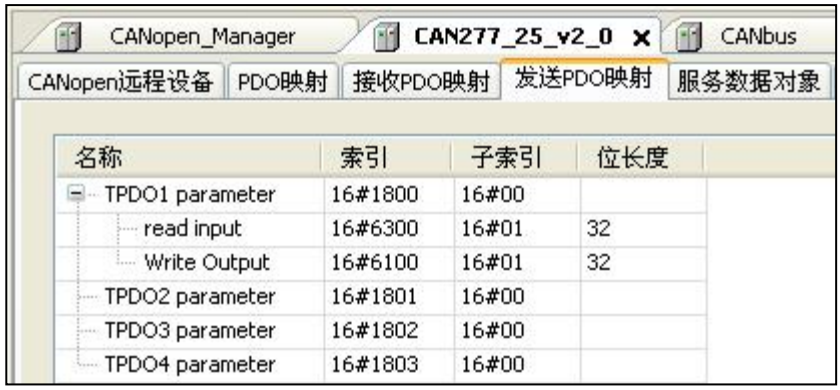
<备注> 需要将 EM277C 的拨码开关设为 4，具体配置方法可参考《CANopen 用户手册》。

“服务器数据对象”：新建添加 16#2200 的 16#01 参数；

“接收 PDO 映射”：在 RPDO1 parameter 中添加如下映射：



“发送 PDO 映射”：在 TPDO1 parameter 中添加如下映射：



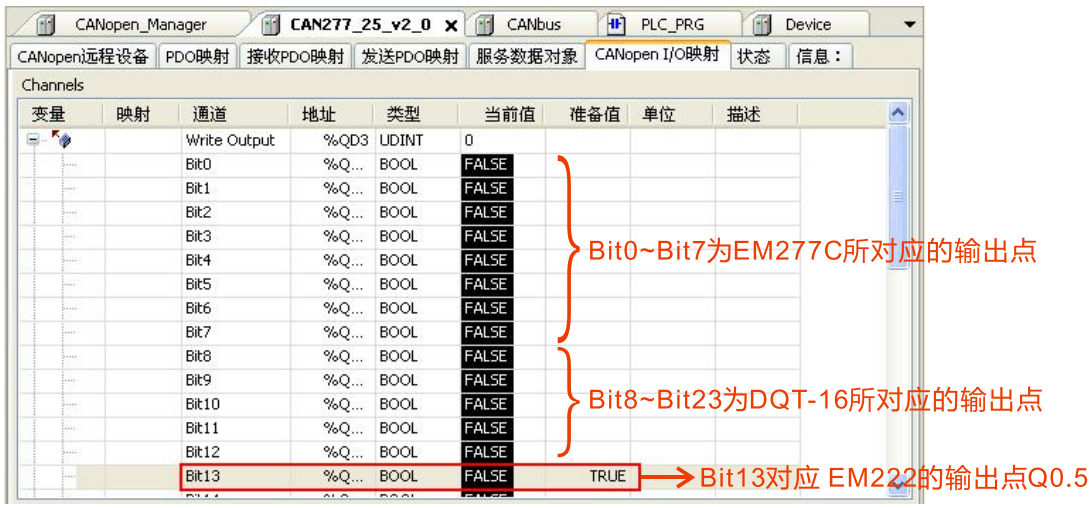
<备注> 请参考附录“F.2 配置 CANopen 从站设备”获取更多关于 CANopen 从站配置详情。

4、参考章节“2.3 设置通信”设置 C37-00 与上位机的通信

5、运行、调试与监控

1）选择菜单项“在线”→“登录到...”使应用程序与 C37-00 建立起连接，并进入在线状态。然后，选择菜单项“调试”→“启动”使 C37-00 中的应用程序开始运行，此时即可对当前工程进行监控和调试操作。

2）在 EM277C 从站的选项卡“CANopen I/O 映射”（勾选 always update variable）中调试其挂接的模块，本例选择的是数字量模块 EM222，则通过点亮其输出点 Q0.5，操作如下图：



结果：Bit13 写入 TRUE 后，EM222 模块的 Q0.5 即被点亮。

7.5 基于 CANopen 通信的单轴运动控制

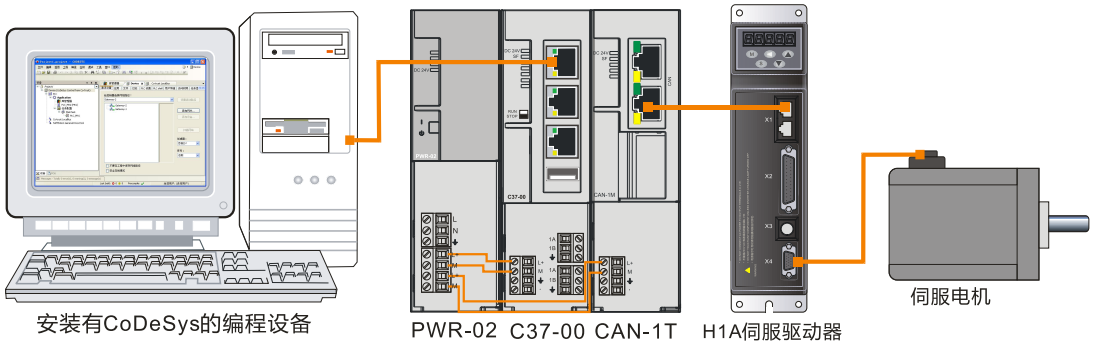
本指南通过一个具体实例来引导您建立一个应用程序，通过本节您将熟悉 C37-00 基于 CANopen 通信的单轴运动控制功能。

7.5.1 示例组件

组件	功能
安装有 CODESYS V3.5 的	对 CTH300-C 系列运动控制器进行组态、编程和调试

编程设备	
装配导轨	用于固定系统中的各模块。
电源模块 PWR-02	提供 CTH300-C 系列运动控制器及其 24 VDC 负载电路。
外部电源	提供 H1A 伺服驱动器工作电源
C37-00 运动控制器	执行用户程序，为总线提供 5 V 电压，并通过以太网接口与其它模块进行通讯。
CAN 主站模块 CAN-1T	作为 CAN 主站，与 CAN 从站进行通信
H1A 伺服驱动器及电机	本例使用：H1A 伺服驱动器及配套电机
标准网线	连接编程设备与C37-00 连接CAN-1T与H1A伺服驱动器
编码器电缆	连接H1A伺服驱动器与电机

7.5.2 网络连接



注意：C35-01/C36-01/C37-01 作为主站时无需 CAN-00 主站模块，直接通过其 CANopen 接口连接 CAN 从站。

7.5.3 操作步骤

1、为各设备接通电源

1）打开 C37-00、电源模块 PWR-02 的前面板，然后按照“7.5.2 网络连接”将 C37-00 连接到 PWR-02。

2）为 H1A 伺服驱动器接通主电源和控制电源。

2、使用电缆连接各设备

参考“7.5.2 网络连接”连接各设备，具体操作如下：

1）使用标准网线连接 PC 与 C37-00

2）使用标准网线连接 C37-00 与 H1A 伺服驱动器

3）使用编码器电缆连接 H1A 驱动器与电机

3、在 CODESYS 中进行 CANopen 组态

1）添加 CANbus

在设备视图中右键点击“Device(CodeSys control from Co-trust)”选择“添加设备”，即可在弹出的对话框中选择添加 CANbus 作为 CAN 总线。

2）添加 CAN 主站

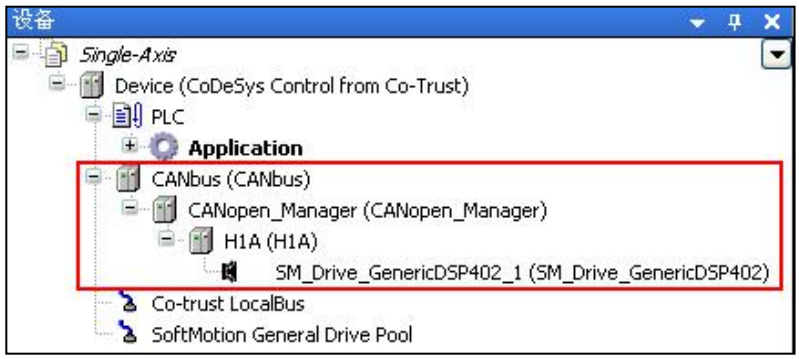
在设备视图中右键点击“CANbus”，在弹出的菜单项中选择“添加设备”，即可在“添加设备”对话框中添加 CAN 主站：现场总线→CANopen→CANopen 管理器→CANopen_Manager。

3）添加或扫描 CAN 从站

在设备视图中右键点击“CANopen_Manager”，在弹出的菜单项中选择“添加设备”，即可在“添加设备”对话框中添加 CAN 从站：现场总线→CANopen→远程设备→H1A；或者右键点击 CANopen_Manager（CAN-1T）选择“扫描设备”，即可将 CAN 从站（H1A）显示在 CAN 主站（CAN-1T）下方。

4）添加 CAN 从站（H1A）的驱动

若在第 3）步中添加了 H1A 作为 CAN 从站，那么此处可以为 H1A 添加驱动：在设备视图中选中 CAN 从站（H1A）并点击右键，在弹出的菜单中选择“Add softmotion-DSP402-axis”，即驱动添加成功。



5）配置 CANbus



6）配置 CANOpen 主站

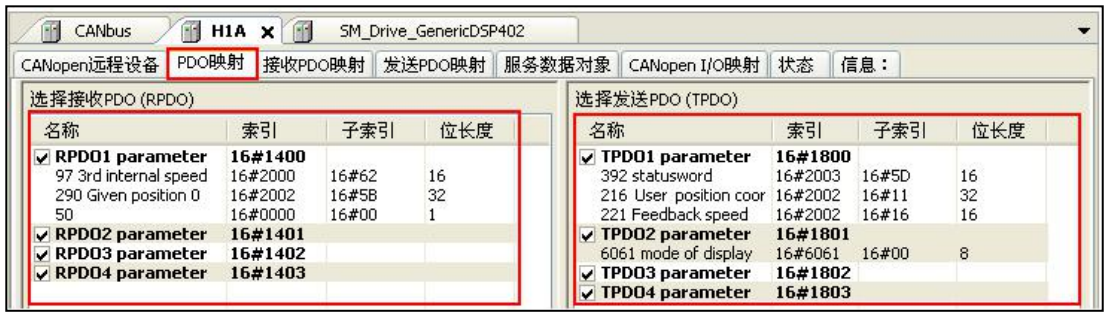
在主站的“CANopen 管理器”中输入 CAN 主站的节点 ID：127

7）配置 CANopen 从站（H1A）

首先，在“CANopen 远程设备”选项卡设置从站的节点 ID（即从站 H1A 的地址），然后勾选“使能专家设置”。



然后，在“PDO 映射”选项卡中选择需要使用的参数，本例设置如下图所示：



在“接收 PDO 映射”选项卡中添加映射：



在“发送 PDO 映射”选项卡中添加映射：

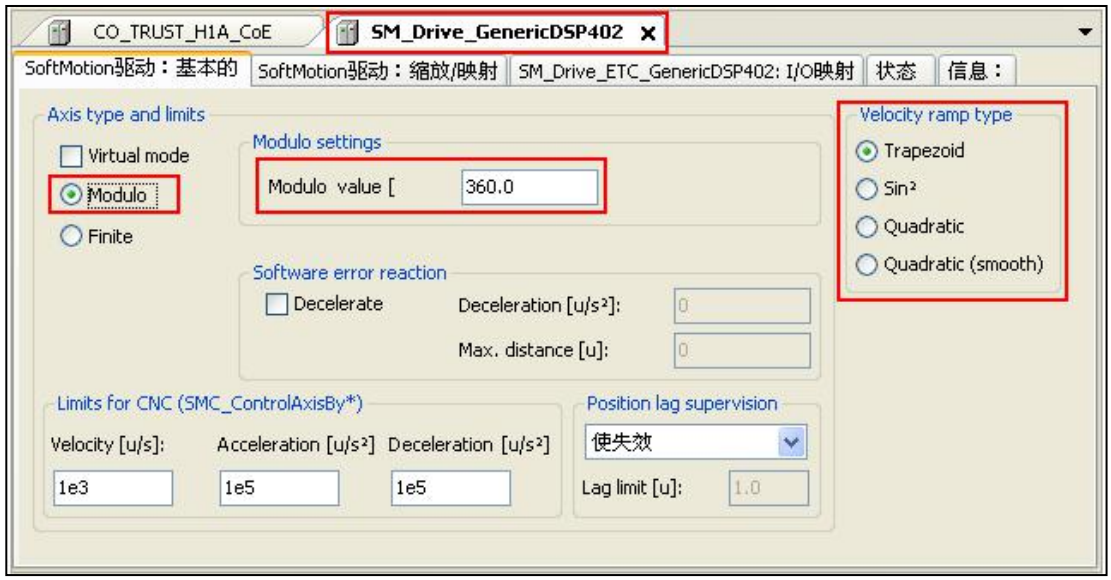


在“CANopen I/O 映射”选项卡中勾选“always update variable”。

<备注> 请参考附录“F.2 配置 CANopen 从站设备”获取更多关于 CANopen 从站配置详情。

8）配置 H1A 驱动（SM_Driver_GenericDSP402）

SM_Driver_GenericDSP402 的“Softmotion 驱动：基本的”选项卡设置如下：



运行轨迹选择为“Modulo”，Modulo 值设置为 360.0，Velocity ramp type（速度斜坡类型）选择为 Trapezoid（梯形）。

4、参考章节“2.3 设置通信”设置 C37-00 与上位机的通信

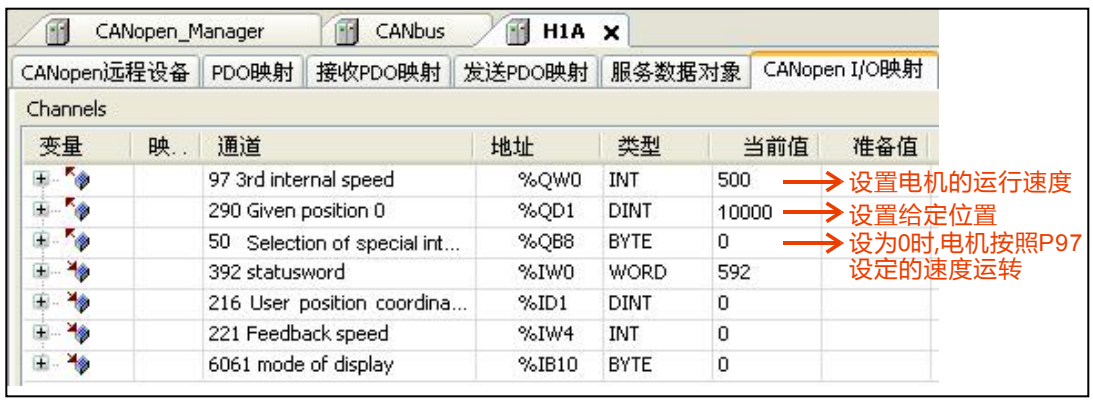
5、运行与调试

1）参考《H1A 使用说明书》对当前 H1A 伺服从站进行参数配置

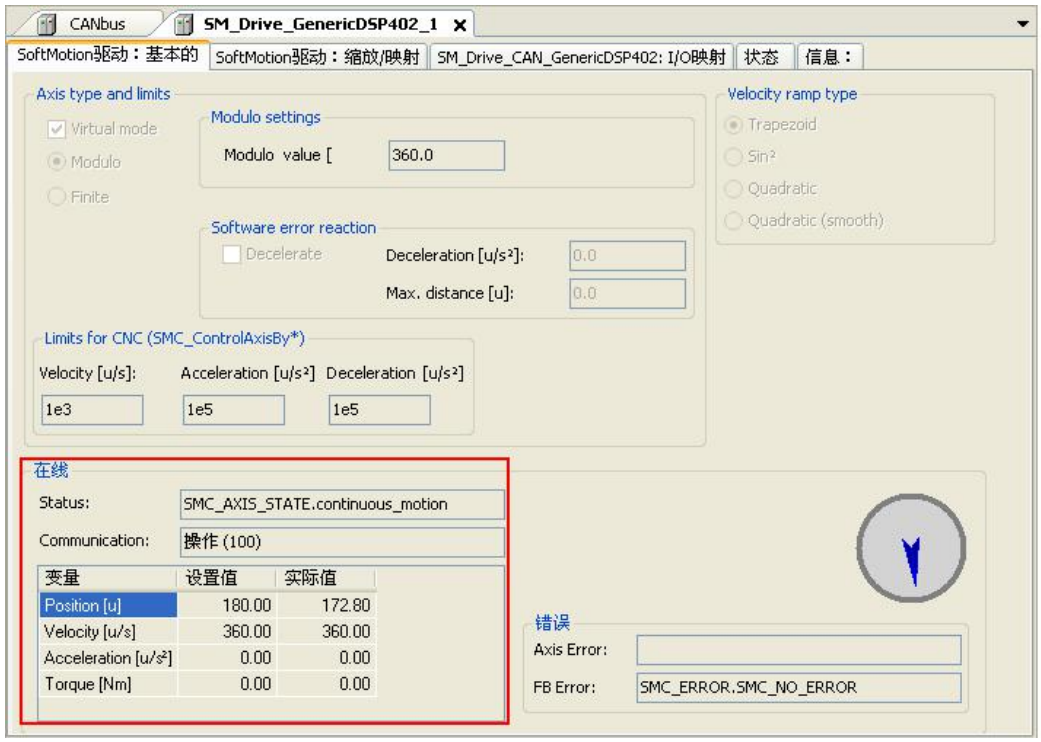
通过 H1A 伺服驱动器的前面板设置控制模式和 CANopen 波特率：P01=1（通信位置控制模式）；P11=1（CANopen 波特率设置为 1000Kbps）。

2）选择菜单项“在线”→“登录...”使应用程序与 C37-00 建立起连接，并进入在线状态；然后，选择菜单项“调试”→“启动”使 C37-00 中的应用程序开始运行。

3）在 H1A 的“CANopen I/O 映射”选项卡中设置 P97、P290 和 P50。其中 P97 设置电机的运行速度，P290 设置给定的位置大小，P50（通信模式指令插补模式选择）选择电机按 P97 设定的速度运行，还是按控制器内部速度运行；本例的 P50 设置为 0，即选择按 P97 设定的速度运行。参数配置如下图所示：



4）在设备视图中双击“SM_Drive_GenericDSP402”，即可在其“Softmotion 驱动”选项卡中监控电机的运行情况，本例中电机的运行情况如下图所示：



结果：与 H1A 伺服驱动器相连的伺服电机按照 P97 设定的速度（500）执行给定位置 10000。

7.6 EtherCAT 通信实例

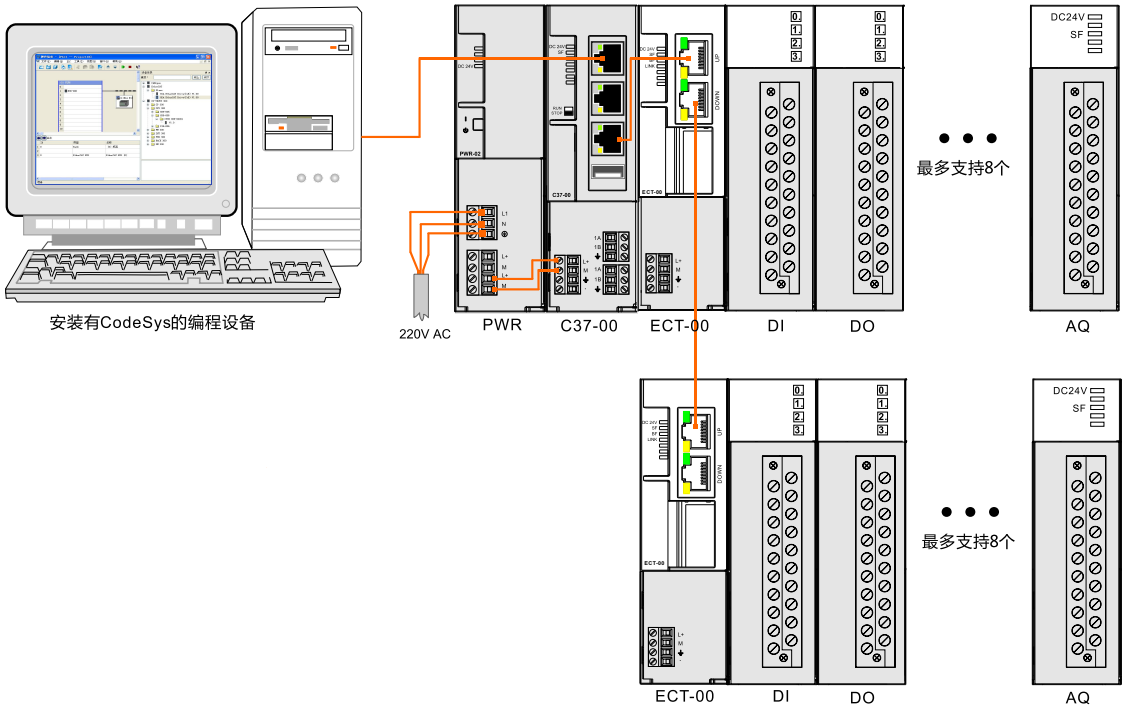
本指南通过一个具体实例来引导用户建立一个应用程序，并熟悉 C37-00 使用 EtherCAT 从站模块时的具体读写操作。

7.6.1 示例组件

组件	功能
编程设备 PG\PC	安装有 CODESYS V3.5 的编程设备，对 CTH300-C 系列运动控制器进行组态、编程和调试
装配导轨	用于固定系统中的各模块
电源模块 PWR-02	提供 CTH300-C 系列运动控制器及其 24 VDC 负载电路
C37-00 运动控制器	执行用户程序，为总线提供 5V 电压，并通过以太网接口与其它模块进行通讯
EtherCAT 主站模块	1 个 CTH300 系列 CPU，C37-00
EtherCAT 从站模块	2 个 ECT-00
扩展 I/O 模块	16 个 CTH300 系列数字量和模拟量 I/O 模块，两个 EtherCAT 从站模块后分别挂接 8 个
标准网线 3 根	● 连接C37-00与编程设备 ● 将C37-00的EtherCAT通信口的OUT接口连接到第一个ECT-00的IN口 ● 将第一个ECT-00的OUT口连接到第二个ECT-00的IN口

7.6.2 网络连接

下图为 EtherCAT 从站模块搭配 CTH300-C 系列 CPU 使用时的典型应用架构：

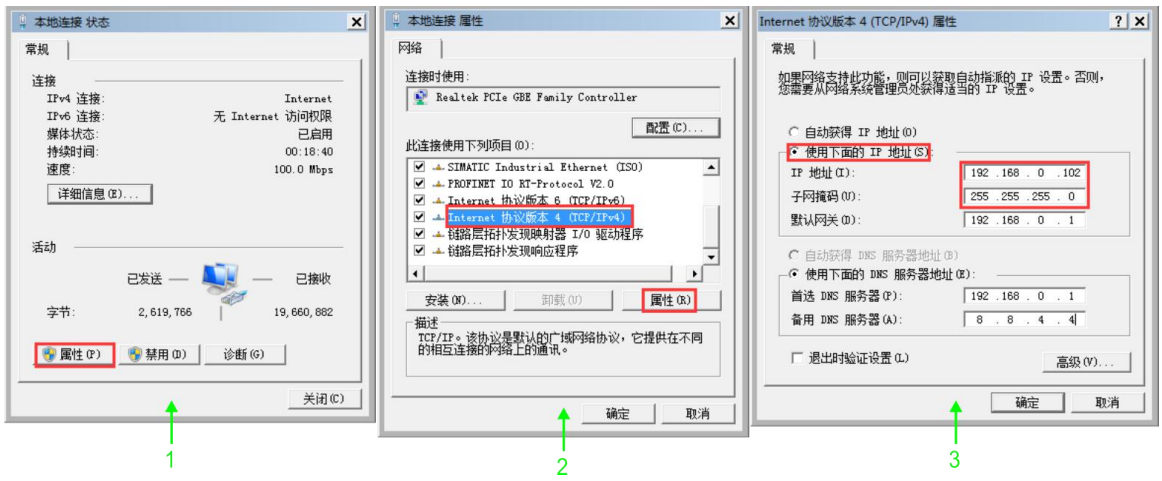


7.6.3 CPU 通信设置

在应用 EtherCAT 从站模块之前，请参照章节 7.6.2 连接 EtherCAT 通信架构并接通电源，然后执行以下操作来设置 CPU 通信。

1、设置通信 IP

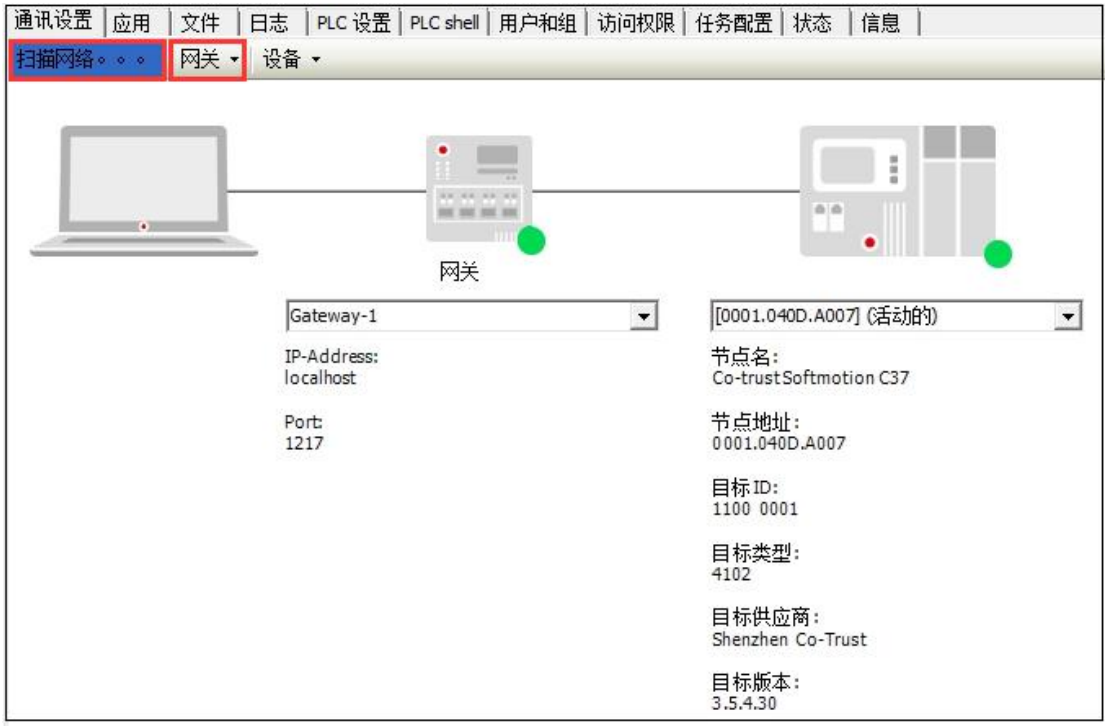
设置通信前，需要将编程设备 PG/PC 的 IP 设置与 C37-00（IP：192.168.0.x）同一个网段，设置方法：将 PC 机的本地连接属性打开，双击 TCP/IP 协议，将“自动获得 IP 地址”更改为“使用下面的 IP 地址”，然后在 IP 地址中填写“192.168.0.X”即可，如下图所示。



2、设置 CPU

在 Codesys 的设备视图中双击“Device (Codesys Control from Co-Trust)”打开设备对话框，如下图所示。在选项卡“通讯设置”中的“网关”下拉菜单中点击“添加网关”按钮，在弹出的“网关”对话框中输入“名称”、“驱动器”选择“TCP/IP”、“IP 地址”选择“localhost”，最后点击“确定”关闭对话框，C37-00 即被添加到通讯对话框。

C37-00 添加成功后点击“扫描网络”按钮，可以在本地网络上搜寻可用的设备。若搜索成功，则选中搜索到的设备并点击“设置活动路径”按钮，该操作会激活通讯通道设置，所有与通讯相关的操作将与该通道关联。



<备注> 系统启动时，系统托盘处会出现 CODESYS 相关服务程序（如、等），如无特殊要求，无需对托盘中的服务程序进行操作。

7.6.4 操作步骤

1、为各设备接通电源

1)打开 C37-00、电源模块 PWR-02 的前面板，然后按照“7.6.2 网络连接”将 C37-00 连接到 PWR。

2）为系统接通主电源和控制电源。

2、使用电缆连接各设备

参考“7.6.2 网络连接”连接各设备，具体操作如下：

1）使用标准网线连接 PC 与 C37-00 的 EtherNET 通信口；

2）使用标准网线连接 C37-00 的 EtherCAT 通信 OUT 口与第一个 EtherCAT 从站模块的 IN 口；

3）使用标准网线连接第一个 EtherCAT 从站的 OUT 口与第二个 EtherCAT 从站模块的 IN 口。

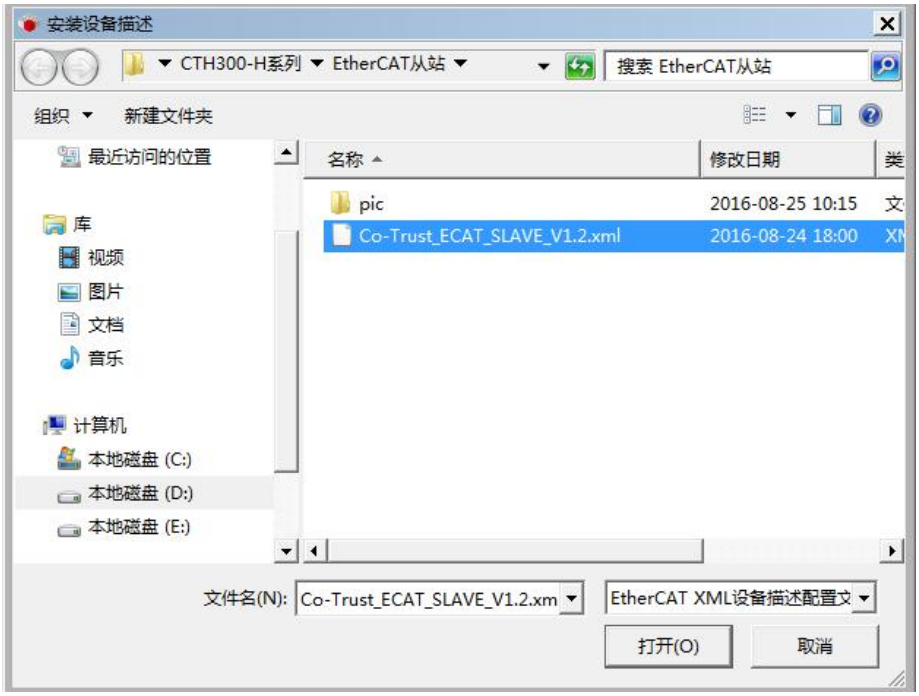
3、在 CODESYS 中进行 EtherCAT 组态

双击桌面图标或在开始菜单中启动 CODESYS 软件，然后执行以下步骤：

1) 安装设备描述文件

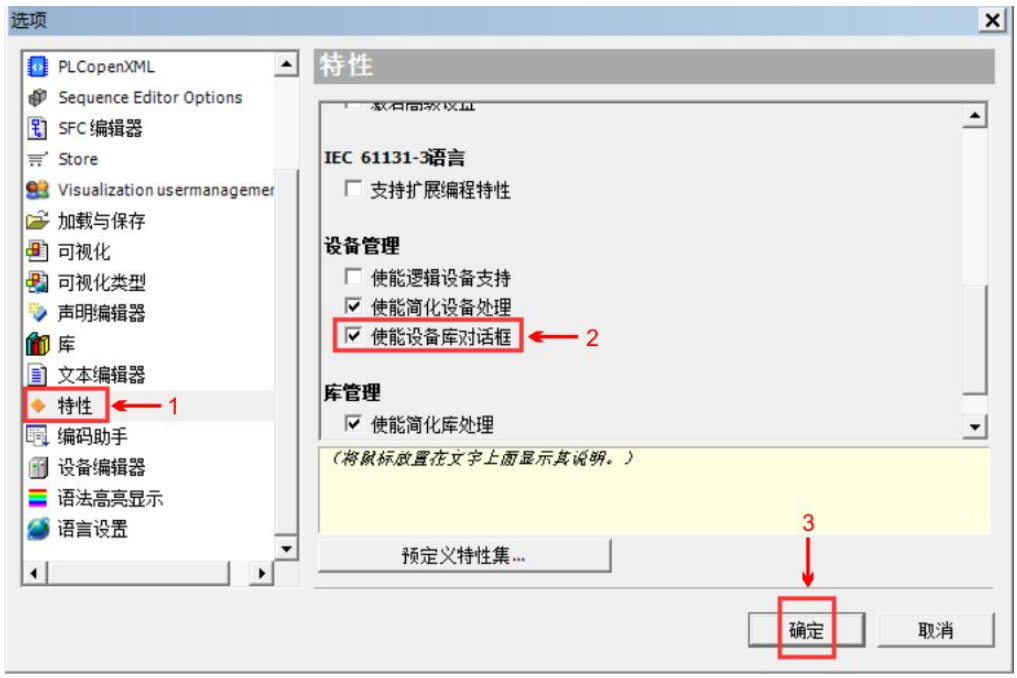
安装示例组件中所用 CPU C37-00 和 EtherCAT 从站模块的 COTRUST 设备描述文件（Co-Trust_C37_V2.3.devdesc.xml 和 Co-Trust_ECAT_SLAVE.V1.2.xml），安装成功后即可在程序系统中使用 COTRUST 设备，具体安装操作如下：

选择菜单项“工具”→“安装设备”打开如下对话框，在该对话框里可以浏览并安装系统中的设备描述。由 COTRUST 提供的描述文件可通过设置相应的过滤器来选择（**注意**：必须选择 EtherCAT XML 设备描述配置文件）。点击“打开”按钮确认选项，随即在“设备库”中添加新设备到设备目录。

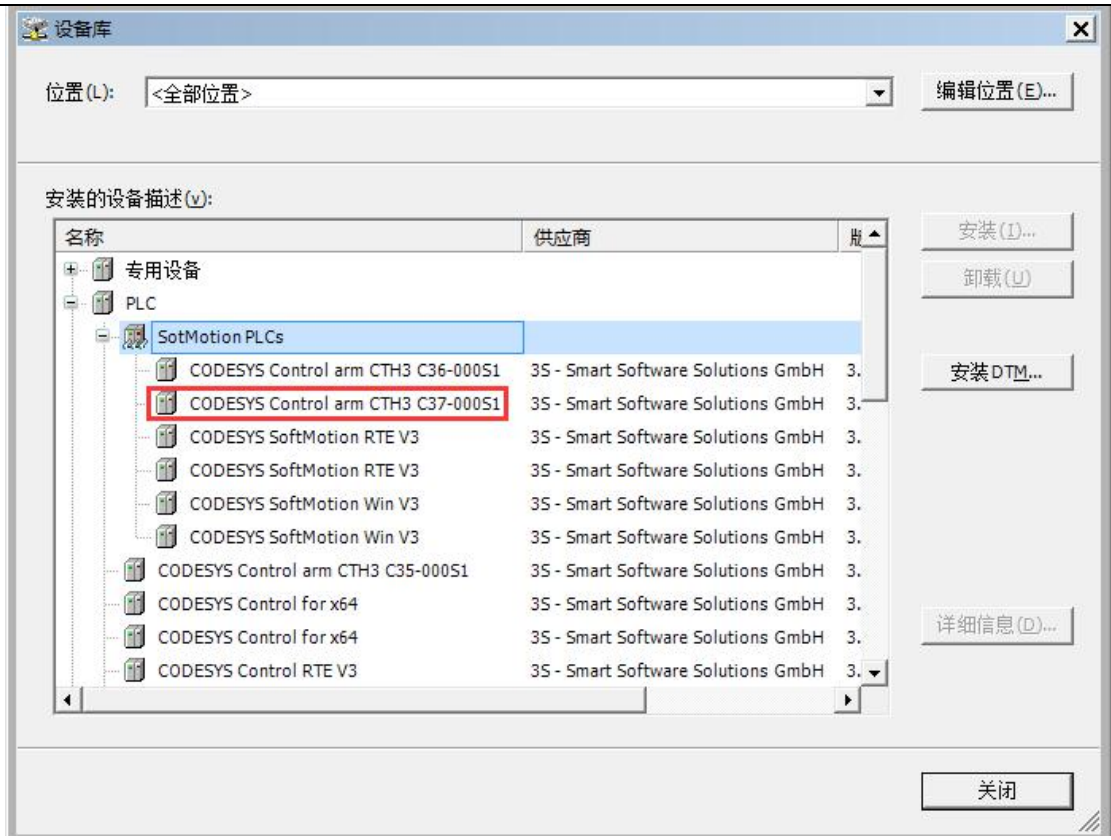


以上操作执行完成后，可参考如下步骤进入设备目录查看文件是否正确安装：

- 选择菜单项“工具”→“选项”→“特性”→“设备管理”→“使能设备库对话框”：



- 选择菜单项“工具”→“设备库”打开设备库对话框，如下所示：



在设备库对话框中展开“PLC”→“SoftMotion PLCs”即可看到已经成功安装的 COTRUST 设备“CodeSys Control from Co-Trust”；展开“现场总线”→“EtherCAT”→“从站”即可看到已经成功安装的 EtherCAT 从站“EtherCAT Slave”。查看完毕，若还需要继续安装设备文件，请在“工具”→“选项”中取消勾选“使能设备库对话框”即可。

提示

1）如果您还需要使用 COTRUST 的其它设备（CANopen 或其他 EtherCAT 从站设备），那么您还需要安装如下设备描述文件：

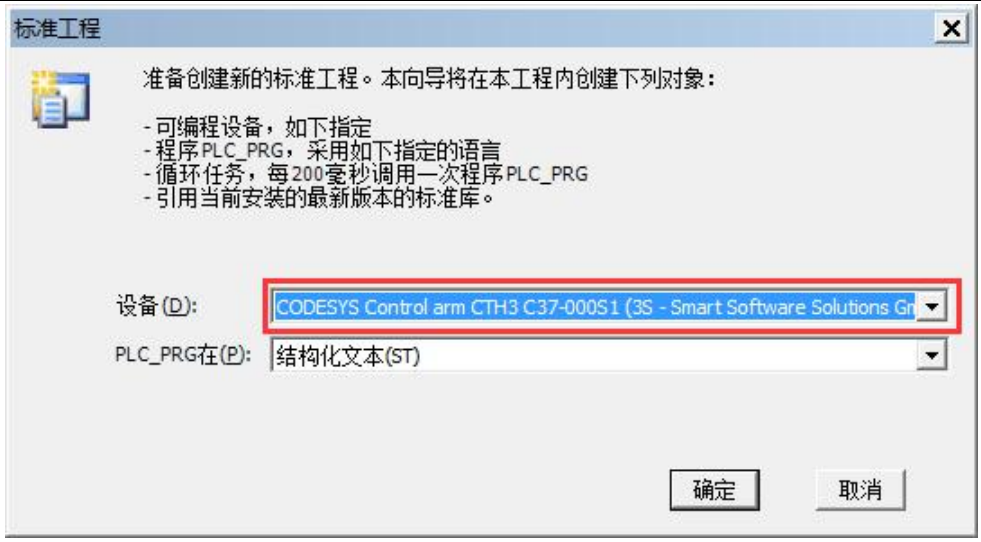
CANopen 从站设备配置文件（*.eds）：Co-Trust E10.eds、Co-Trust EM277C.eds、Co-Trust H1A.eds；

Softmotion 设备配置文件（*.devdesc.xml）：H1A EtherCAT Drive(CoE) SoftMotion.devdesc、H1A CANopen Drive SoftMotion.devdesc；

2）请于合信官网下载所需的设备配置文件：http://www.co-trust.com。

2）新建一个工程

- 在 CODESYS 主页面选择菜单项“文件”→“新建工程”，然后在弹出的对话框中选择“标准工程”并设置文件名和存储目录。
- 执行以上操作并确认后会弹出另一个对话框，在对话框中选择刚添加的 CPU 文件设备（CTH3 C37-000S1）和使用的编程语言，再点击“确定”即可完成工程的创建。

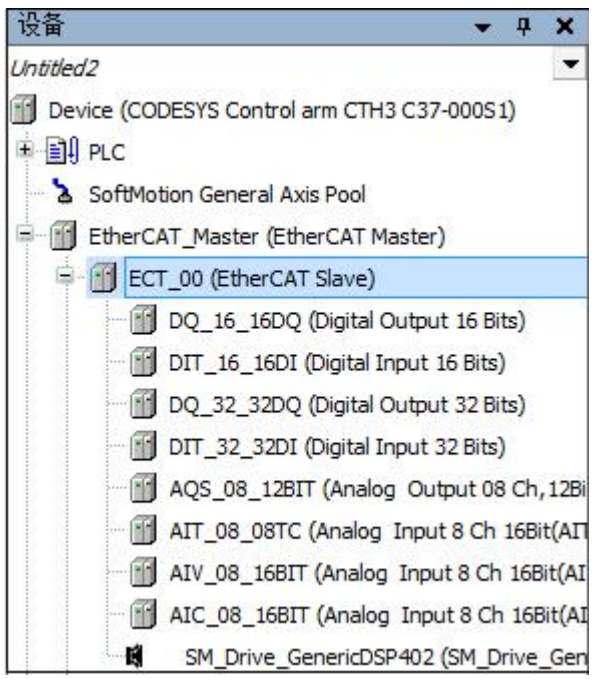


3) 添加 EtherCAT 主站

在打开的工程中，右键点击设备视图中的“Device (CodeSys control from Co-trust)”选择“添加设备”，即可在弹出的对话框中选择添加 EtherCAT 主站：供应商选择“<全部供应商>”、现场总线选择“EtherCAT”→“主站”→“EtherCAT Master”。

4) 添加 EtherCAT 从站

在设备视图选中 EtherCAT 主站并点击右键选择“添加设备”，即可在弹出的对话框中选择添加 EtherCAT 从站：供应商选择“<全部供应商>”、现场总线选择“EtherCAT”→“从站”→“ECT-00”，即可将连接到 C37-00 的 EtherCAT 从站（ECT-00）显示在 EtherCAT 主站（C37-00）下方。



5) 配置 EtherCAT 主站

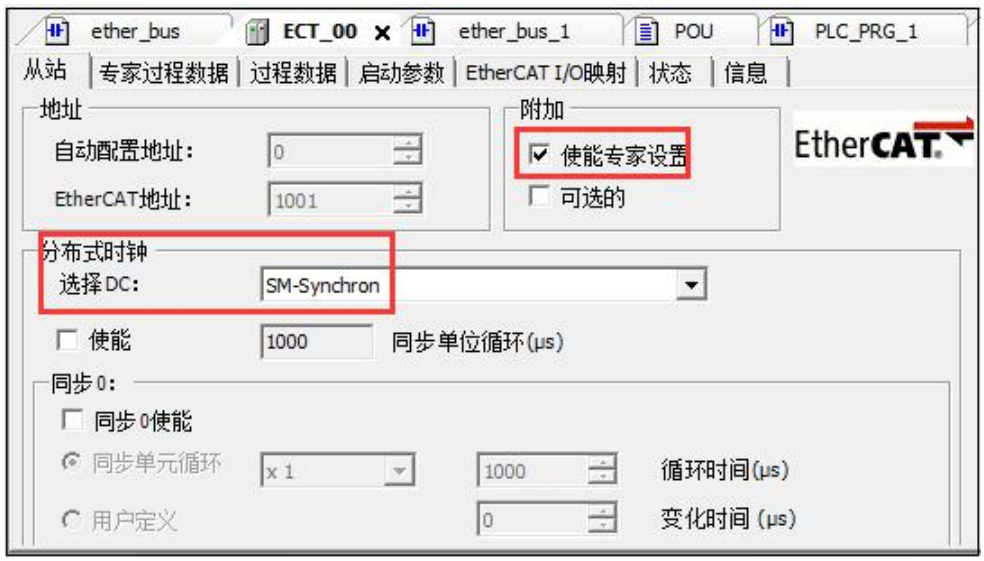
在“主站”选项卡中设置源地址（MAC）：点击“源地址（MAC）”右侧的“浏览”按钮，然后在弹出的对话框中选择相应的 MAC 地址。



<备注> 如果环型拓扑支持冗余，则可以勾选“Enable Redundancy”。通过使用这个功能，EtherCAT 网络功能即使在电缆破损时也可以被保存。如果勾选“Enable Redundancy”，必须定义“Redundancy EtherCAT NIC Setting”。

6) 配置 EtherCAT 从站（ECT-00）

在“从站”选项卡中勾选“使能专家设置”，分布式时钟选择 DC“SM-Synchron”。



<备注> 请参考附录“F.3 配置 EtherCAT 从站设备”获取更多关于 EtherCAT 从站配置详情。

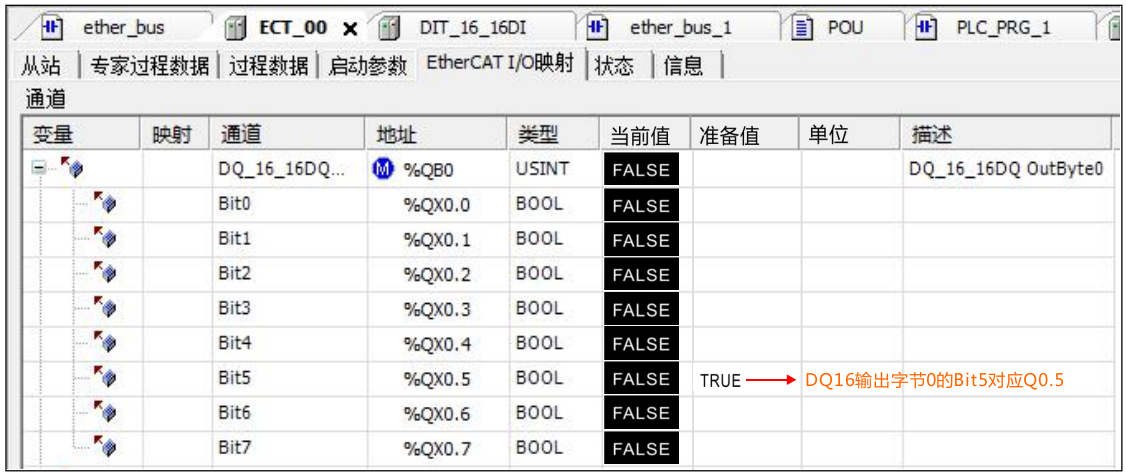
5、参考章节“7.6.3 CPU 通信设置”设置 C37-00 与上位机的通信

6、运行与调试

1）选择菜单项“在线”→“登录...”使应用程序与 C37-00 建立连接，并进入在线状态；然后，选择菜单项“调试”→“启动”使 C37-00 中的应用程序开始运行。

2）对当前工程进行监控和调试操作

在 ECT_00 从站的选项卡“EtherCAT I/O 映射”（勾选 always update variable）中调试其挂接的模块，以 Digital Output 16 Bits 数字量输入模块为例，点亮其输出点 Q0.5 操作如下：



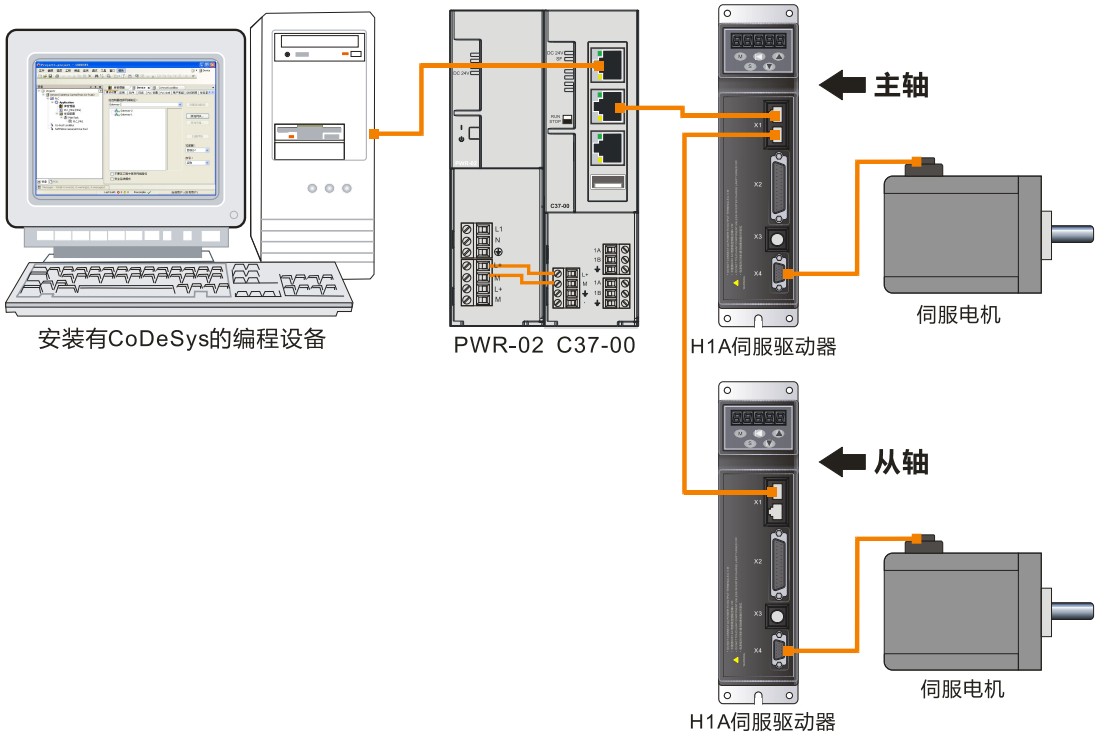
7.7 基于 EtherCAT 通信的电子凸轮

本指南通过一个具体实例来引导您建立一个应用程序，通过本节您将熟悉 C37-00 基于 EtherCAT 通信的电子凸轮控制功能。

7.7.1 示例组件

组件	功能
安装有 CODESYS V3.5 的编程设备	对 CTH300-C 系列运动控制器进行组态、编程和调试
装配导轨	用于固定系统中的各模块。
电源模块 PWR-02	提供 CTH300-C 系列运动控制器及其 24 VDC 负载电路。
外部电源	提供 H1A 伺服驱动器工作电源
C37-00 运动控制器	执行用户程序，为总线提供 5V 电压，并通过以太网接口与其它模块进行通讯。
H1A 伺服驱动器及电机	本例使用：H1A 伺服驱动器及配套电机
标准网线	◆ 连接编程设备与C37-00 ◆ 连接C37-00与H1A伺服驱动器 ◆ H1A伺服驱动器之间的互联
编码器电缆	连接H1A伺服驱动器与电机

7.7.2 网络连接



7.7.3 操作步骤

1、为各设备接通电源

1）打开 C37-00、电源模块 PWR-02 的前面板，然后按照“7.7.2 网络连接”将 C37-00 连接到 PWR-02。

2）为 H1A 伺服驱动器接通主电源和控制电源。

2、使用电缆连接各设备

参考“7.7.2 网络连接”连接各设备，具体操作如下：

1）使用标准网线连接 PC 与 C37-00

2）使用标准网线连接 C37-00 与 H1A 伺服驱动器

3）使用编码器电缆连接 H1A 驱动器与电机

3、在 CODESYS 中进行 EtherCAT 组态

1）添加 EtherCAT 主站

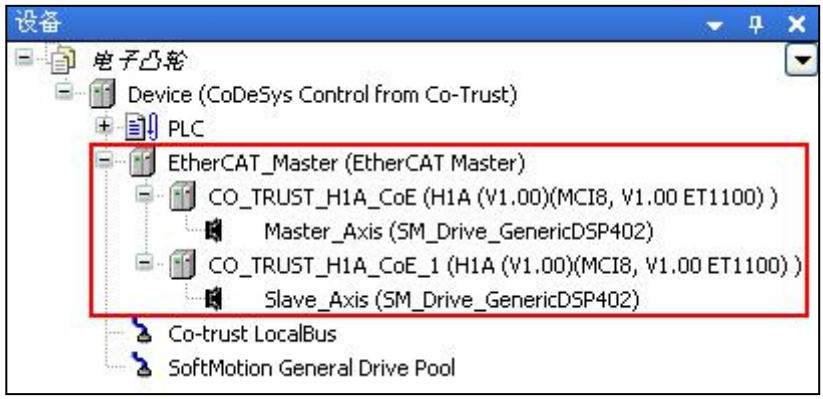
在设备视图中右键点击“Device(CodeSys control from Co-trust)”选择“添加设备”，即可在弹出的对话框中选择添加 EtherCAT 主站：供应商选择“<全部供应商>”、现场总线选择“EtherCAT”→“主站”→“EtherCAT Master”。

2）添加或扫描 EtherCAT 从站

在设备视图选中 EtherCAT 主站并点击右键选择“添加设备”，即可在弹出的对话框中选择添加 EtherCAT 从站：供应商选择“<全部供应商>”、现场总线选择“EtherCAT”→“从站”→“H1A”；或者右键点击 EtherCAT 主站（C37-00）选择“扫描设备”，即可将连接到 C37-00 的 EtherCAT 从站（H1A）显示在 EtherCAT 主站（C37-00）下方。

3）添加 EtherCAT 从站的驱动

在设备视图中选中 EtherCAT 从站（H1A）并点击右键，在弹出的菜单中选择“Add softmotion-DSP402-axis”，即驱动添加成功。



4）配置 EtherCAT 主站

在“主站”选项卡中设置源地址（MAC）：点击“源地址（MAC）”右侧的“浏览”按钮，然后在弹出的对话框中选择相应的 MAC 地址。



<备注> 如果环型拓扑支持冗余，则可以勾选“Enable Redundancy”。通过使用这个功能，EtherCAT 网络功能即使在电缆破损时也可以被保存。如果勾选“Enable Redundancy”，必须定义“Redundancy EtherCAT NIC Setting”。

5）配置 EtherCAT 从站（H1A）

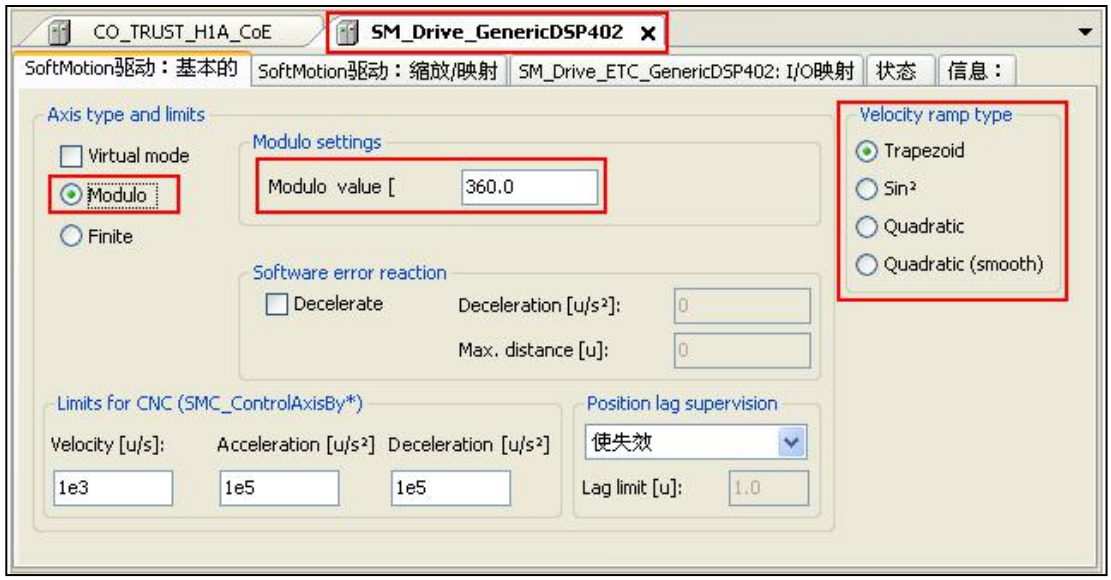
在“从站”选项卡中勾选“使能专家设置”，分布式时钟选择“DC for synchronization”。



<备注> 请参考附录“F.3 配置 EtherCAT 从站设备”获取更多关于 EtherCAT 从站配置详情。

6）配置 H1A 驱动（SM_Driver_GenericDSP402）

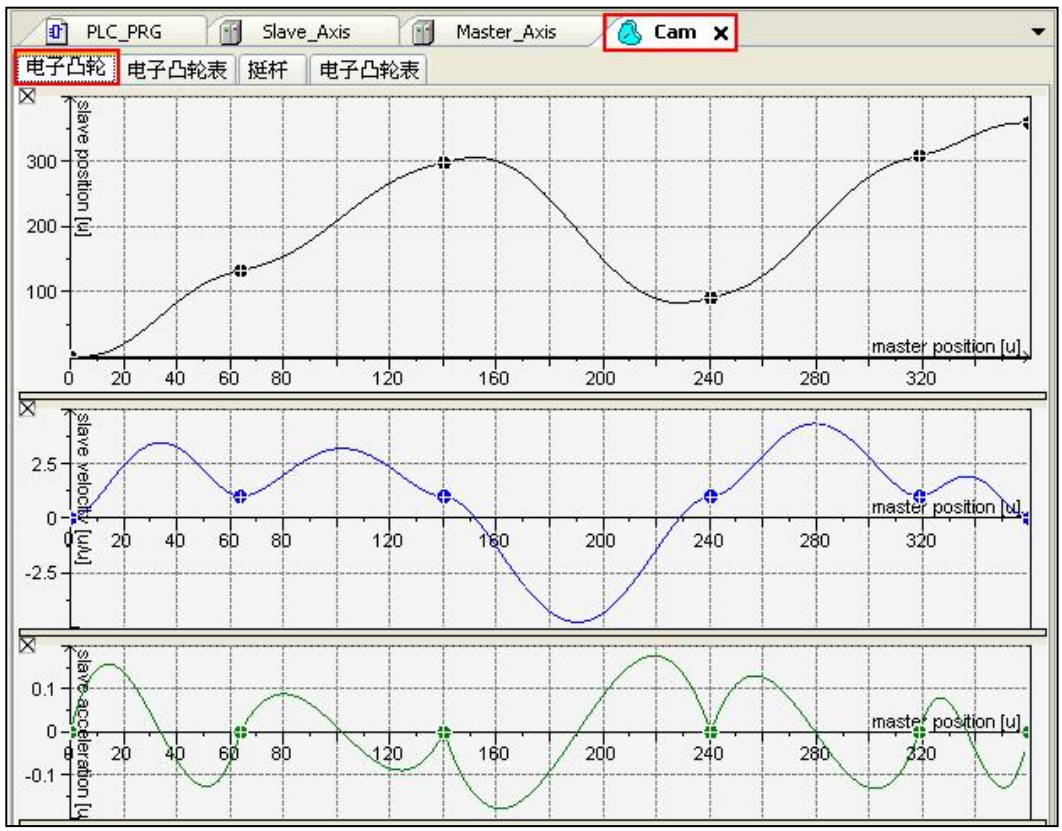
SM_Driver_GenericDSP402 的“Softmotion 驱动：基本的”选项卡设置如下：



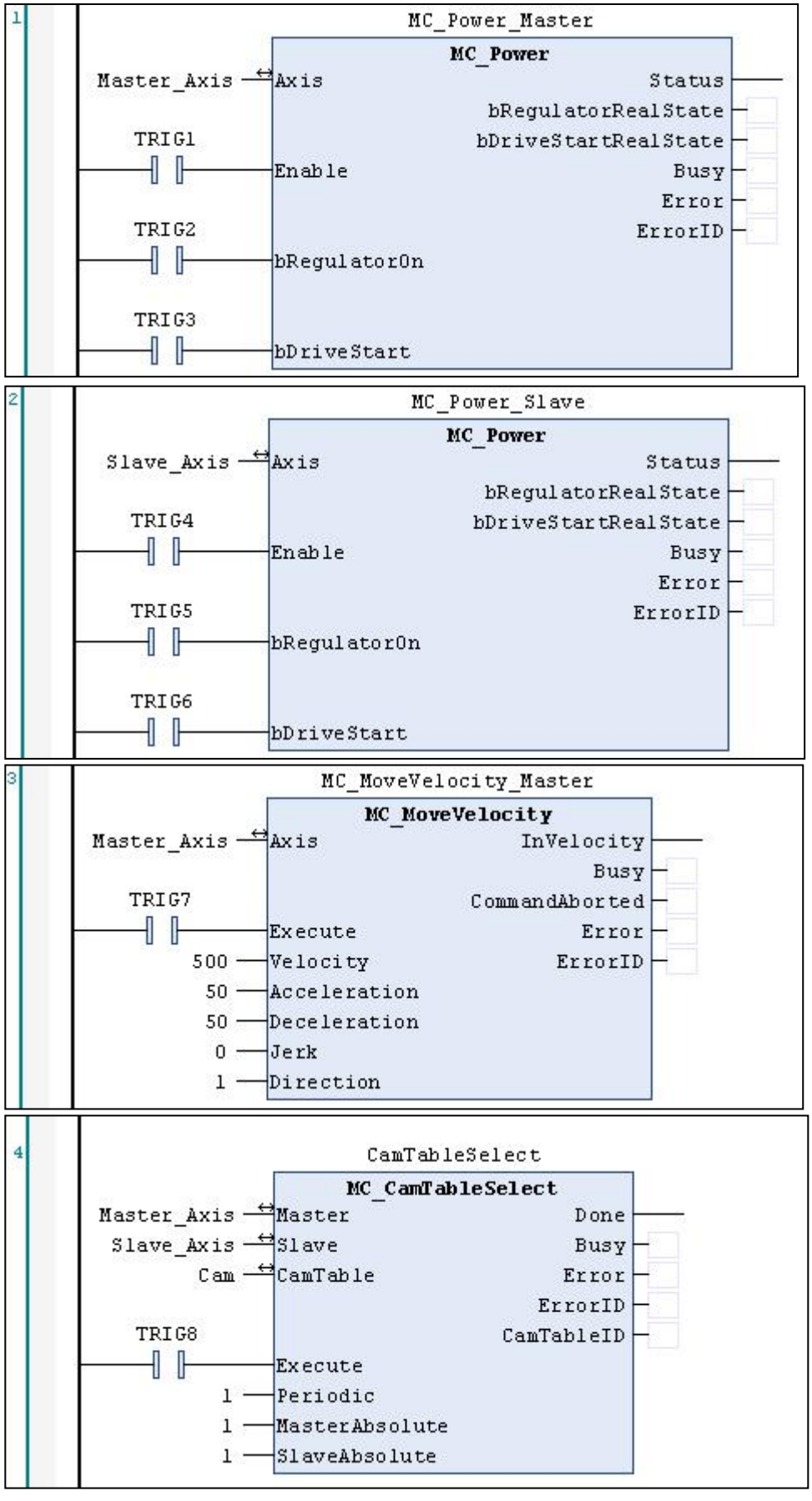
运行轨迹选择为“Modulo”，Modulo 值设置为 360.0，Velocity ramp type（速度斜坡类型）选择为 Trapezoid（梯形）。

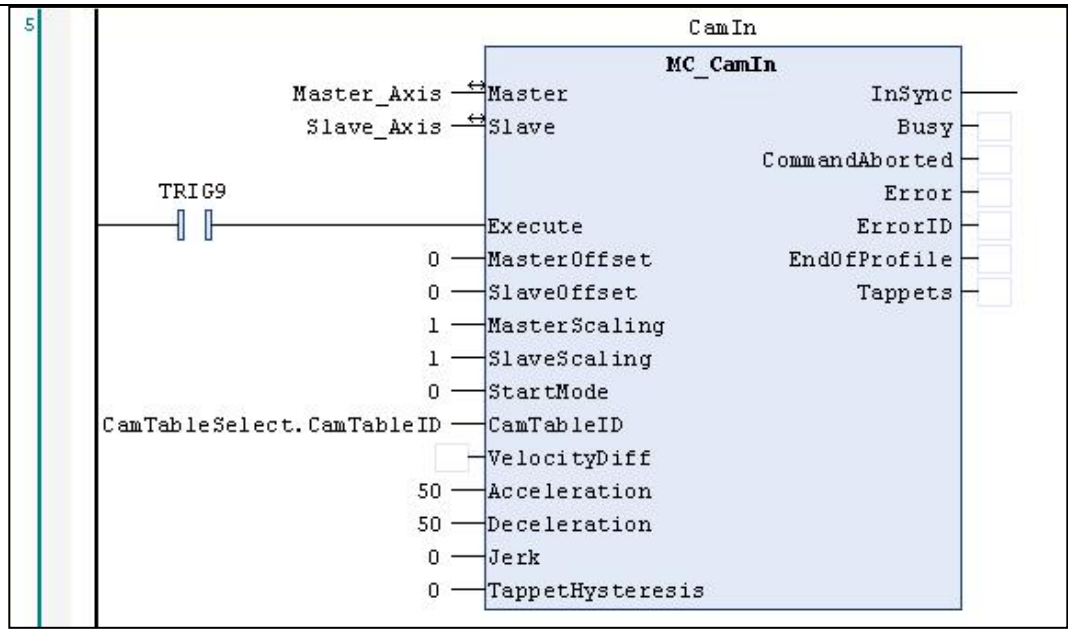
7）新建 CAM 表

在设备视图展开“Device”→“PLC”，然后右键点击“Application”选择“添加对象”→“电子凸轮表”，在弹出的对话框中输入表的名称，点击“打开”，则 Cam 表即被创建成功。双击打开刚创建的 Cam 表，将电子凸轮表调整成理想状况，如下所示：



4、编程


1
MC_Power_Master
MC_Power
Master_Axis
Axis
Status
bRegulatorRealState
bDriveStartRealState
TRIG1
Enable
Busy
Error
TRIG2
ErrorID
bRegulatorOn
TRIG3
bDriveStart
2
MC_Power_Slave
MC_Power
Slave_Axis
Axis
Status
bRegulatorRealState
TRIG4
bDriveStartRealState
Enable
Busy
Error
TRIG5
ErrorID
bRegulatorOn
TRIG6
bDriveStart
3
MC_MoveVelocity_Master
MC_MoveVelocity
Master_Axis
Axis
InVelocity
Busy
TRIG7
CommandAborted
Execute
Error
500
Velocity
ErrorID
50
Acceleration
50
Deceleration
0
Jerk
1
Direction
4
CamTableSelect
MC_CamTableSelect
Master_Axis
Master
Done
Slave_Axis
Slave
Busy
Cam
CamTable
Error
ErrorID
TRIG8
CamTableID
Execute
1
Periodic
1
MasterAbsolute
1
SlaveAbsolute



5、参考章节“2.3 设置通信”设置 C37-00 与上位机的通信

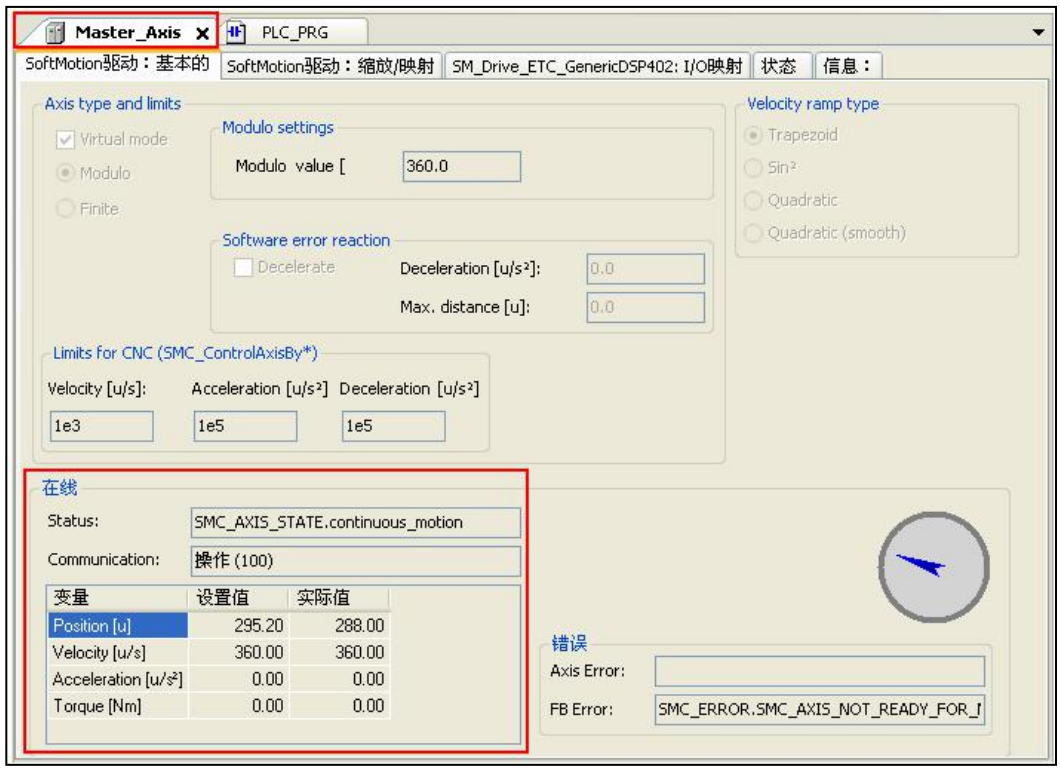
6、运行与调试

1）选择菜单项“在线”→“登录...”使应用程序与 C37-00 建立起连接，并进入在线状态；然后，选择菜单项“调试”→“启动”使 C37-00 中的应用程序开始运行。

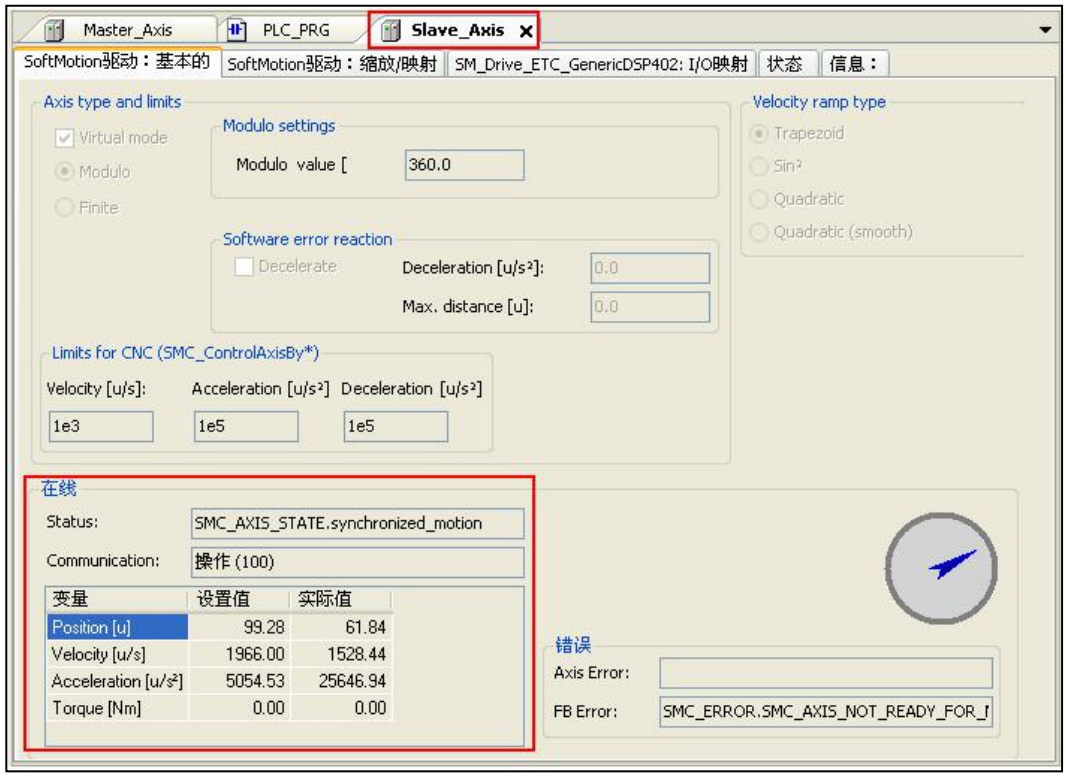
2）对当前工程进行监控和调试操作

在设备视图中双击“PLC_PRG(PRG)”打开程序表，依次触发并下载各指令。

在设备视图中双击打开主轴 Master_Axis，Master_Axis 的执行效果如下：



在设备视图中双击打开从轴 Slave_Axis，Slave_Axis 的执行效果如下：



7.7.4 跟踪（Trace）

1、创建跟踪表（Trace）

在设备视图展开“Device”→“PLC”，然后右键点击“Application”选择“添加对象”→“Trace”，在弹出的对话框中输入跟踪的名称，点击“打开”，跟踪即被创建成功。

2、添加跟踪对象（Add Variable）

为了在视图中显示被跟踪的变量，首先必须在工程中插入合适的跟踪对象。在这里可以配置这个被跟踪的变量、触发变量和跟踪本身（坐标轴，刷新率，等）。本例选择跟踪主轴 Master_Axis 的位置变量、从轴 Slave_Axis 的位置变量。

1）主轴 Master_Axis 的位置变量添加方法：点击“Add Variable”→“变量”→在输入助手对话框中选择“Categories”→“IoConfig_Globals”→“Master_Axis”→“fActPosition”。

2）按照以上步骤添加从轴 Slave_Axis 的位置变量，添加成功的变量会在右侧表框中出现。

3、配置跟踪（Configuration）

点击“Configuration”进行跟踪配置，在该对话框中输入需要触发变量、触发沿、触发采样、任务等。本例监控主轴 Master_Axis 的速度指令 MC_Movevelocity，则应该选择该指令的触发变量 TRIG7。

选择触发变量的步骤：“Configuration”→“触发变量”→“Categories”→“Application”→“PLC_PRG”→“TRIG7”。

触发沿：正向、触发采样：150、任务：EtherCAT_Master。

4、下载跟踪

1）按照以上步骤配置完成后，右键点击 Trace 表选择“下载跟踪”，则 Trace 表开始绘制曲线。

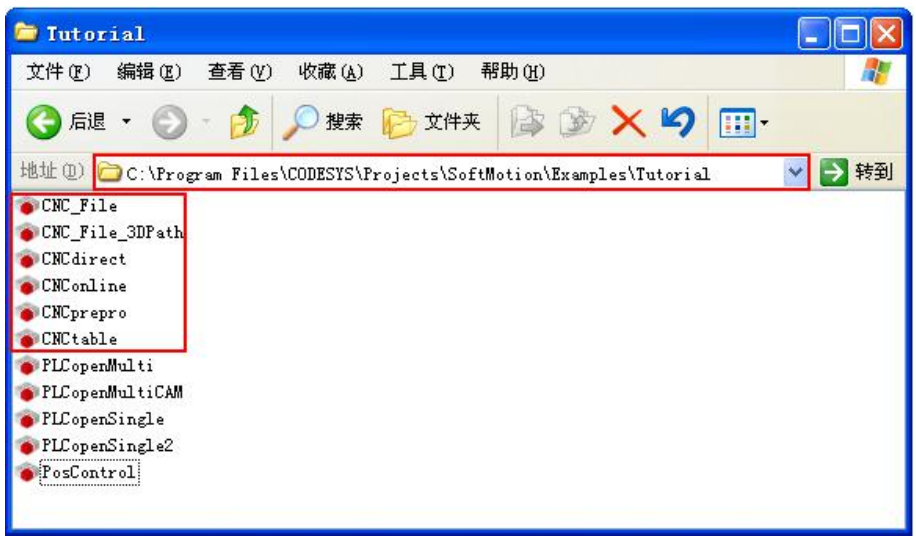
2）将 Trace 表与 CAM 表中的曲线图进行对比，即可查看电子凸轮的执行效果。



7.8 CNC 例程

请于 CODESYS 安装目录中获取有关 CNC 功能的例程：

C:\Program Files\co-trust\CODESYS\Projects\SoftMotion\Examples\Tutorial



7.9 高速计数器模块使用举例

本节基于具体实例，主要使用高速计数模块对伺服电机编码器反馈的脉冲信号进行计数和测速。完成本节后，您将会熟悉高速计数模块的基本功能。

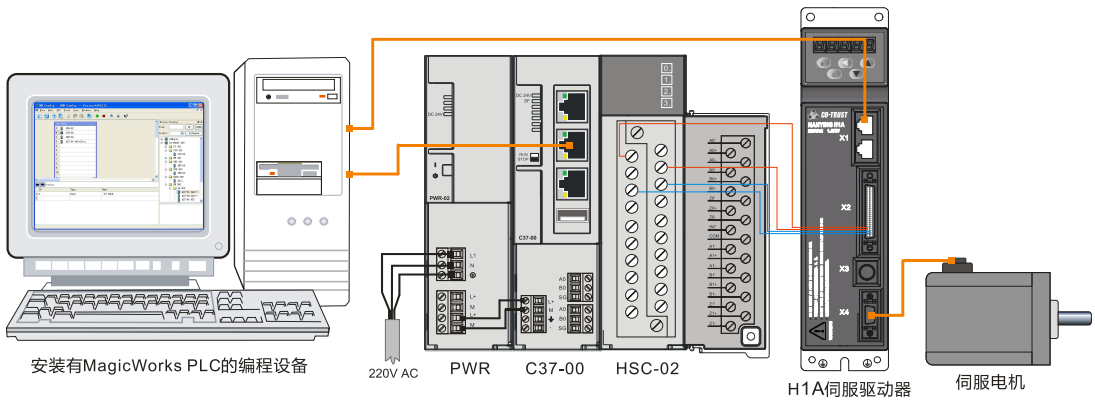
7.9.1 示例组件

表 7-8 示例组件

组件	功能
编程设备 PG\PC	安装有 CODESYS V3.5 的编程设备（V3.5 SP4 及以上版本），对

	CTH300-C 系列运动控制器进行组态、编程和调试
装配导轨	CTH300 系统机架，用于固定系统中的各模块。
电源模块 PWR-02	提供 CTH300-C 系列运动控制器及其 24 VDC 负载电路。
外部电源	提供 H1A 伺服驱动器工作电源
C37-00 运动控制器	执行用户程序，为总线提供 5 V 电压，并通过以太网接口与其它模块进行通讯。
HSC-02 高速计数模块	通过总线与 C37-00 进行连接，对电机速度、位置等进行计数。
H1A 伺服驱动器及电机	本例使用：H1A 伺服驱动器及配套电机
标准网线	◆ 连接编程设备与C37-00 ◆ 连接 C37-00 与 H1A 伺服驱动器
编码器电缆	连接 H1A 伺服驱动器与电机

7.9.2 网络连接



7.9.3 操作步骤

步骤 1：接线

打开 C37-00 和电源模块 PWR-02 的前面板，为它们接线。

步骤 2：连接电缆

参考“7.9.2 网络连接”连接各设备，具体操作如下：

1）使用标准网线连接 PC 与 C37-00

2）C37-00 与高速计数模块通过总线进行连接

3）使用标准网线连接 PC 与 H1A 驱动器

4）使用编码器电缆连接 H1A 驱动器与电机

5）为高速计数模块和 H1A 驱动器接线（H1A 伺服驱动器 X2 端子的的 A+、A-、B+、B-分别与 HSC-02 的 A0+、A0-、B0+、B0-一一对应接线）

步骤 3：设置 PLC 通信

参考章节“2.3 设置通信”设置通信。

步骤 4：在 CODESYS 中进行组态

在设备目录中右键点击“Co-Trust LocalBus”选择“添加设备”，然后在弹出的对话框中将供应商选择为“Co-Trust”，选择“专用设备 Interface 8 Slot”，最后点击“添加设备”按钮即可添

加该设备，添加成功的设备将显示在 Co-Trust LocalBus 下方，最后选择添加成功的 INT_00 并右键选择“添加设备”，选择添加 HSC-02 设备。

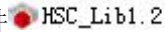


<备注> 双击设备 HSC-02 打开其属性对话框可以配置该模块的模式、控制字等参数，若在此处配置后，即无需使用指令 HSC_300、HSC_SETMODE 指令。

步骤 5：运行驱动系统

通过设置 H1A 伺服驱动器参数使电机开始正常运转，具体操作参考《H1A 系列交流伺服驱动器使用说明书》。

步骤 6：添加库文件 hsc_lib1

通过菜单项“工具”->“安装库”选择添加库文件

。

库文件添加成功后会弹出一个提示框，提示添加成功。然后将添加成功的库添加到库管理器中，请参考如下操作：

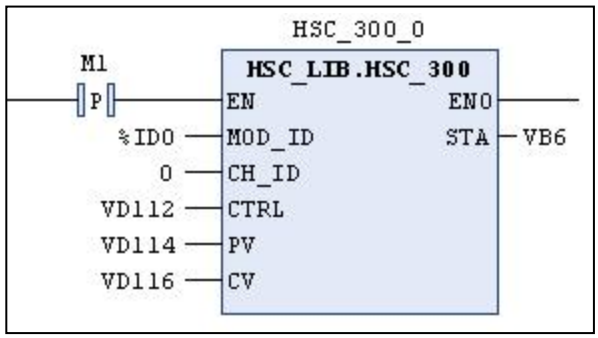
1）从设备目录中打开库管理器：PLC->Application->库管理器

2）然后在库管理器界面点击

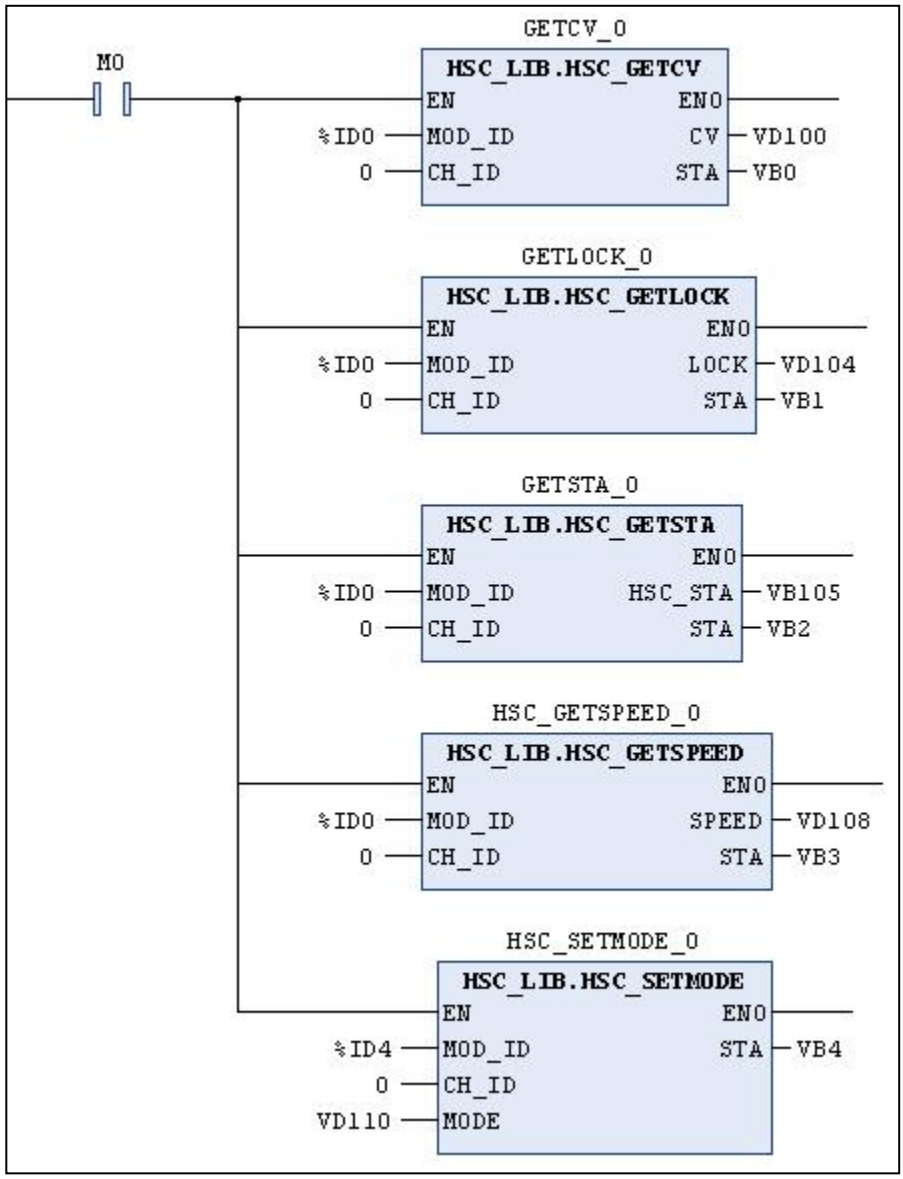
添加库文件“Co-Trust HSC Library”

步骤 7：使用 hsc_lib1 库进行编程

HSC-02 模块的库文件已经成功添加，可以直接通过指令读取该模块的速度、位置及状态等信息。



<备注> 若在 HSC-02 模块属性对话框中配置了控制字、模式等参数，则无需调用 HSC_300、HSC_SETMODE 指令。



步骤 8：调试与监控程序

1）选择菜单项“在线”→“登录...”使应用程序与 C37-00 建立起连接，并进入在线状态；然后，选择菜单项“调试”→“启动”使 C37-00 中的应用程序开始运行。

2）调试程序

通过程序中的 HSC_GETCV、HSC_GETSPEED 等指令读取伺服电机的当前位置、速度等值。

附录

您可以通过本附录快速查阅如下信息：

- ❒ 术语解释
- ❒ 电源预算
- ❒ 扩展模块技术规范
- ❒ IO 模块通道配置控制字意义
- ❒ CT_MODBUS 库的使用介绍
- ❒ 在 CODESYS 中进行的相关操作详述
- ❒ 指令速查
- ❒ 订货信息

A 术语解释

表 A-1 缩略术语解释

缩写	全称	解释
AC	Alternating Current	交流
DC	Direct Current	直流
ADC	Analog to Digital Converter	模数转换器
DAC	Digital-to-Analog Converter	数模转换器
CODESYS	Controlled Development System	受控的开发系统
PLC	Programmable Logic Controller	可编程逻辑控制器
CPU	Central Processing Unit	PLC 的中央处理单元
HMI	Human Machine Interface	人机界面
FB	Function Block	功能块
FC	Function	功能
HSC	High-speed counter	高速计数器
IEC	International Electro Technical Commission	国际电工委员会
SDO	Service Data Object	服务数据对象
PDO	Process Data Object	过程数据对象
RTD	Resistance Temperature Detector	电阻温度探测器
TC	Thermocouple	热电偶
UDP	User Datagram Protocol	用户数据包协议
IL	Instruction List	指令表
ST	Structure Text	结构化文本
SFC	Sequential Function Chart	顺序功能图

CFC	Continuous Function Chart	连续功能图表编辑器
FBD	Function Block Diagram	功能模块图
LD	Ladder Diagram	梯形图

B 电源预算

选择好各机架的 CPU、电源模块、中继模块和扩展模块后，还需要确认系统总线的电流消耗和功率消耗是否满足以下条件：

条件 1：总线电流消耗确认

内部总线电压为 5VDC，电流由 CPU（无中继模块时）或中继模块提供。每个机架上扩展模块总线电流消耗之和不能超过 CPU 或中继模块允许的最大总线电流。

条件 2：功率消耗确认

使用电源模块时，每个机架上其它模块的功率消耗之和不能超过电源模块允许的最大功耗。

使用外部电源时，根据所接的功率之和选择合适功率大小的型号。

表 B-1　5VDC 总线电流供给和消耗

产品型号	供给电流	消耗电流
C35-00	1600mA	--
C36-00	1600mA	--
C37-00	1600mA	--
C35-01	1600mA	--
C36-01	1600mA	--
C37-01	1600mA	--
C37-02	1600mA	--
INT-00	1600mA	--
DIT-08	--	60mA
DIT-16	--	80mA
DIT-32	--	130mA
DQT-08	--	70mA
DQT-16	--	120mA
DQT-32	--	210mA
DQR-08	--	45mA
DQR-16	--	60mA
AIS-04	--	50mA
AIC-08	--	30mA
AIV-08	--	30mA
AQS-04	--	40mA
AQS-08	--	40mA
AMS-06	--	50mA
AIT-04	--	50mA
AIT-08	--	50mA
AIR-04	--	50mA
AIR-08	--	50mA

HSC-02	--	100mA
HSP-04	--	100mA
CAN-1T	--	100mA

表 B-2　24VDC 总线电流供给和消耗

产品型号	供给电流	消耗电流
PWR-02	2000mA	--
C35-00	--	800mA
C36-00	--	800mA
C37-00	--	800mA
C35-01	--	800mA
C36-01	--	800mA
C37-01	--	800mA
INT-00	--	800mA
DIT-08	--	--
DIT-16	--	--
DIT-32	--	--
DQT-08	--	50mA
DQT-16	--	95mA
DQT-32	--	180mA
DQR-08	--	64mA
DQR-16	--	130mA
AIS-04	--	65mA
AIC-08	--	50mA
AIV-08	--	50mA
AQS-04	--	110mA
AQS-08	--	200mA
AMS-06	--	110mA
AIT-04	--	50mA
AIT-08	--	50mA
AIR-04	--	60mA
AIR-08	--	80mA
HSC-02	--	--
HSP-04	--	100mA
CAN-1T	--	100mA

C　扩展模块技术规范

C.1　电源模块

PWR-02 电源模块，最佳匹配于 CTH300 系统，专门为 CPU 及扩展模块（除数字量模块外）提供 24V DC 电源。每个机架请选配一个电源模块，数字量输入输出电源和传感器电源请选择其他供电电源。

表 C-1　电源模块 PWR-02 的基本属性

名称	规格描述	订货号
PWR-02电源模块	输入：85～264V AC 输出：24V DC/2A	CTH3 PWR-020S1

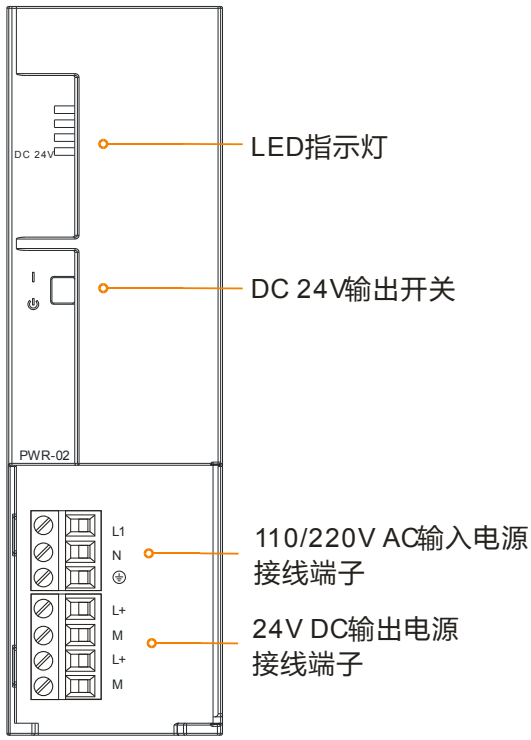
表 C-2　电源模块 PWR-02 的常规特性

物理特性	
尺寸（W×H×D）	34×115×101.6 mm
LED 指示灯特性	
24V 电源指示灯（绿色）	亮起：有 24V DC 输出，熄灭：无 24V DC 输出
开关特性	
开关	控制 24V DC 电源输出 ON：有 24V DC 输出，OFF：无 24V DC 输出

表 C-3　电源模块 PWR-02 的功能特性

输入电压特性	
电压范围	85～264VAC，宽电压输入
额定频率 频率范围	50Hz/60Hz 47Hz～63Hz
效率	75%
交流电流	0.9A/110V、0.5A/220V
浪涌电流（25℃最大）	≤20A/110V、≤35A/220V
泄露电流	≤5mA/220VAC
输出电压特性	
直流电压/额定电流	24VDC/2A
额定功率	48W
纹波和噪声（最大）	150mVp-p
电压输出范围	±5%
启动/上升/保持时间	≤2.5s/≤50ms/≥20ms
隔离（电源输入与输出）	110V/220V AC 与 24V DC 之间隔离
保护功能	
过载保护	105%～130%的额定输出功率，切断输出，故障排除自动恢复
过压保护	115%～135%Ue；保护方式：打嗝模式，故障排除自动恢复
浪涌保护	供电电源端提供浪涌吸收功能
过流保护	电源输出端提供过流保护
安全电磁兼容	
耐电压	输入~输出：1.5KVDC，输入-PE：1.5KVDC，输出-PE：500VDC
隔离电阻	输入~输出，输入-PE，输出-PE：100MΩ/500VDC
依据标准	安全参照UL60950和UL1950，电磁兼容参照EN55022

接口示意图



接口定义

表 C-4　PWR-02 的 220V AC 输入电源接口定义

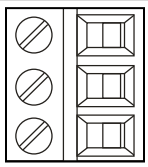
3 位可拆卸端子	信号	信号定义
	L	火线
	N	零线
	⏚	大地

表 C-5　PWR-02 的 24V DC 输出电源接口定义

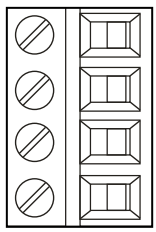
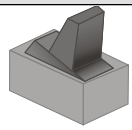
4 位可拆卸端子	信号	信号定义
	L+	24V 电源正
	M	24V 电源负
	L+	24V 电源正
	M	24V 电源负

表 C-6　PWR-02 的拨码开关定义

两态开关	位号	拨码方向	信号定义
	ON	向上	有 24V DC 输出
	OFF	向下	无 24V DC 输出

C.2　数字量模块

表 C-7　数字量模块的基本属性

名称	规格描述	订货号
DIT-080S1 数字模块	数字量输入 8 x 24VDC	CTH3 DIT-080S1
DIT-160S1 数字模块	数字量输入 16 x 24VDC	CTH3 DIT-160S1
DIT-320S1 数字模块	数字量输入 32 x 24VDC	CTH3 DIT-320S1
DQT-080S1 数字模块	数字量输出 8 x 24VDC	CTH3 DQT-080S1
DQT-160S1 数字模块	数字量输出 16 x 24VDC	CTH3 DQT-160S1
DQT-320S1 数字模块	数字量输出 32 x 24VDC	CTH3 DQT-320S1
DQR-080S1 数字模块	数字量输出 8 x 继电器	CTH3 DQR-080S1
DQR-160S1 数字模块	数字量输出 16 x 继电器	CTH3 DQR-160S1

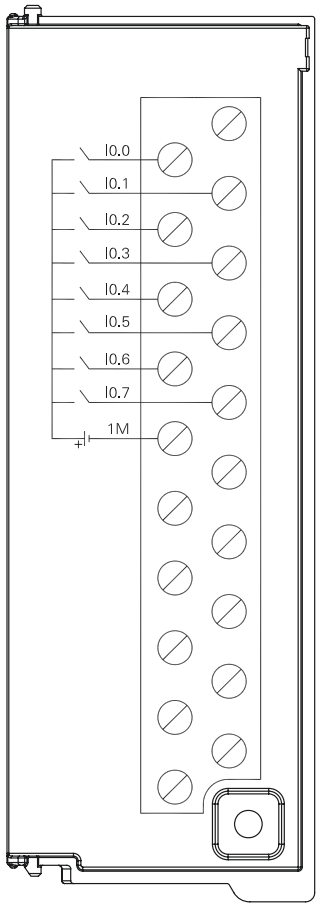
数字量输入模块

表 C-8　数字量输入模块规范

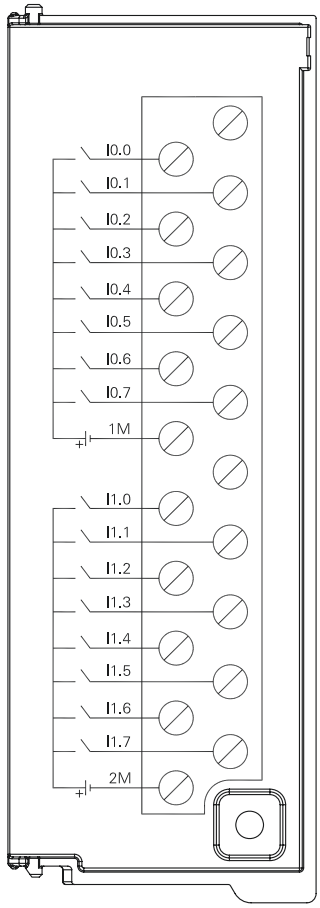
特性		DIT-080S1	DIT-160S1	DIT-320S1
尺寸（W×H×D）		34×115×100 mm		
输入点数		8	16	32
电流消耗	24V DC	4mA/通道	4mA/通道	4mA/通道
	+5V 总线	60mA	80mA	130mA
输入类型		漏型/源型（IEC 1 类漏型）		
输入额定电压		24V DC，6mA		
输入电压范围		20.4~28.8V DC		
浪涌电压		35V DC，持续 0.5s		
逻辑 1（最小）		15V DC，2.5mA，翻转电平：10.5V DC±15%		
逻辑 0（最大）		5V DC，1mA		
连接 2 线接近开关传感器（BERO） 允许的漏电流（最大）		1mA		
输入滤波		可配置，支持 0.2ms、0.4ms、0.8ms、1.6ms、3.2ms、6.4ms（默认）、12.8ms		
输入频率		1.5KHz，占空比 50%		
输入阻抗		6.6KΩ		
隔离		500V AC，持续 1min		
每组隔离点数		8	8	8
同时 ON 点数		8	16	32
电缆长度	屏蔽	500m		
	非屏蔽	300m		

接线规格

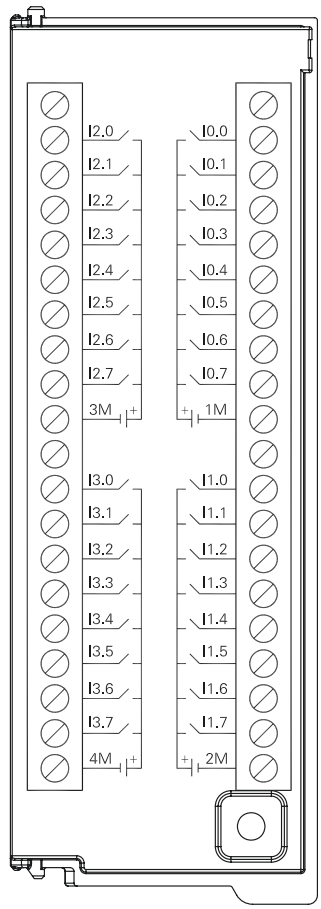
◆ DIT-080S1 接线图



◆ DIT-160S1 接线图



◆ DIT-320S1 接线图



数字量输出模块

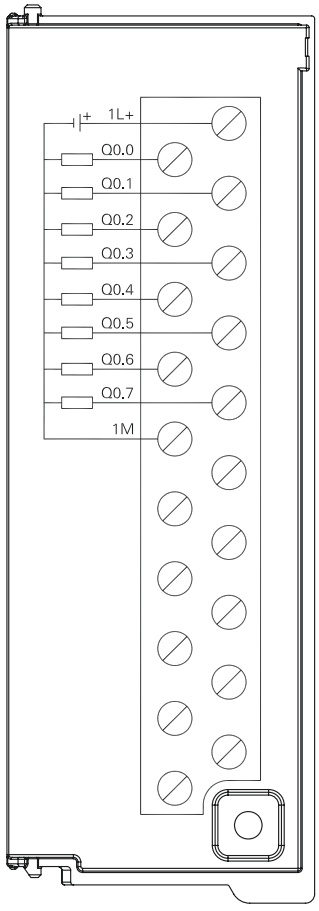
表 C-9　数字量输出模块规范

特性	DQT-080S1	DQT-160S1	DQT-320S1	DQR-080S1	DQR-160S1
尺寸（W×H×D）	34×115×100 mm				
输出点数	8	16	32	8	16
电流消耗 24V DC +5V 总线	 50mA 70mA	 95mA 120mA	 180mA 210mA	 64mA 45mA	 130mA 60mA
输出类型	固态-MOSFET，源型			继电器-干触点	
输出额定电压	24V DC			DC：24V，AC：110V/220V	
输出电压范围	20.4~28.8V DC			DC：5～30V，AC：5～250V	
逻辑 1（最小）	20V DC			-	
逻辑 0（最大）	0.1V DC，10KΩ负载			-	
输出电流（最大）	0.5A			2A	
每个公共端电流	最大 4A			最大 16A	
输出漏电流（最大）	15uA			-	
浪涌电流	8A，100ms			5A，4s@10%占空比	
灯负载	5W			DC：30W /AC：200W	

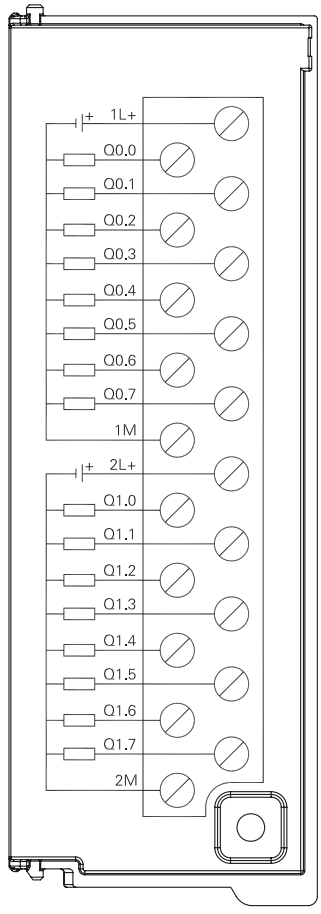
接触电阻		0.3Ω，最大 0.6Ω			新设备最大为 0.2Ω	
输出延迟		断开到接通（最大）：50us 接通到断开（最大）：200us			最大 10ms	
输出频率（最大）		1KHz			阻性负载：10Hz 灯负载：1Hz 感性负载：0.5Hz	
机械寿命（无负载）		-			10，000，000	
触点寿命（额定负载）		-			100，000	
隔离 现场到逻辑 线圈到触点		500V AC，持续 1min -			500V AC，持续 1min 1500V AC，持续 1min	
每组隔离点数		8	8	8	8	8
同时 ON 点数		8	16	32	8	16
两个输出并联		支持同组内两个输出并联				
电缆长度	屏蔽	500m				
	非屏蔽	150m				

接线规格

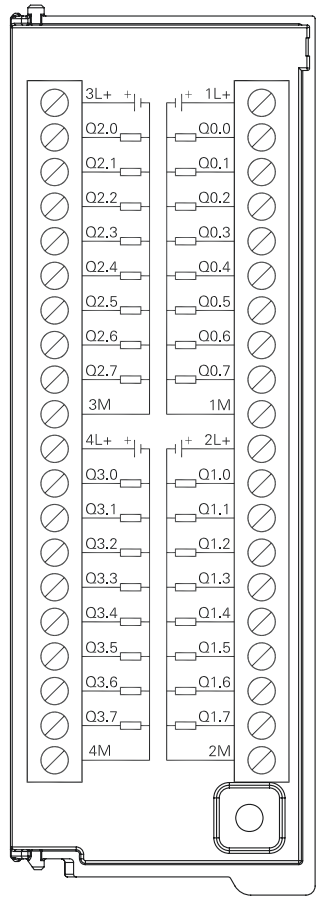
◆ DQT-080S1 接线图



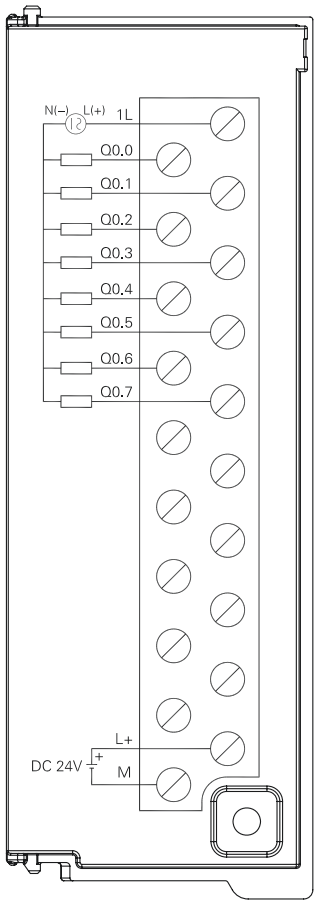
◆ DQT-160S1 接线图



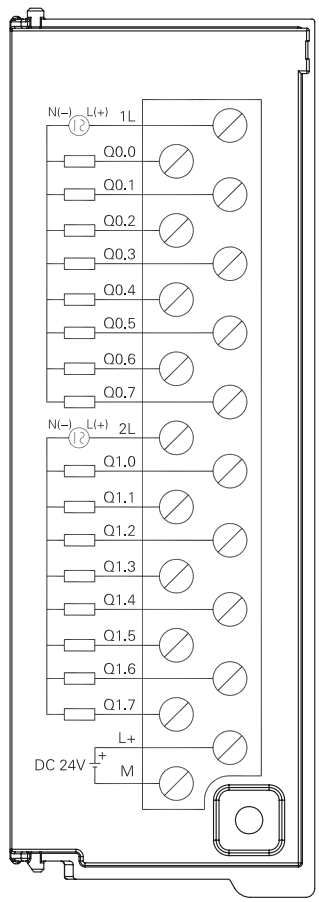
◆ DQT-320S1 接线图



◆ DQR-080S1 接线图



◆ DQR-160S1 接线图



C.3 模拟量模块

表 C-10 模拟量模块的基本属性

名称	规格描述	订货号
AIS-04模拟模块	模拟量电压电流输入，4AI x 12bit	CTH3 AIS-040S1
AMS-06模拟模块	模拟量电压电流输入输出，4AI x 12bit，2AQ x 12bit	CTH3 AMS-060S1
AIV-08 模拟量模块	模拟量电压输入，8AI x 16bit	CTH3 AIV-080S1
AIC-08 模拟量模块	模拟量电流输入，8AI x 16bit	CTH3 AIC-080S1
AQS-04模拟模块	模拟量电压电流输出，4AQ x 12bit	CTH3 AQS-040S1
AQS-08模拟模块	模拟量电压电流输出，8AQ x 12bit	CTH3 AQS-080S1

表 C-11 模拟量模块的常规特性

特性	AIS-040S1	AMS-060S1	AIV-080S1/ AIC-080S1	AQS-040S1	AQS-080S1
物理特性					
尺寸（W×H×D）	34×115×100 mm				
电源特性					
额定输入电压	24V DC				
输入电压范围	20.4V~28.8V DC				
输入电流	65mA	110mA	50mA	110mA	200mA
极性反接保护	有				

总线电源电压	+5V DC				
总线电源电流	50mA	50mA	30mA	40mA	40mA
LED 指示灯特性					
24V 电源指示灯	亮起：24VDC 供电正常，熄灭：无 24VDC 供电				
SF 指示灯	亮起：模块故障，熄灭：无错 闪烁：输入电流信号超限报警（仅针对 4-20mA）				

C-12 模拟量模块的功能特性

功能类别	功能项	描述
扩展功能	扩展功能	提供总线扩展功能
隔离功能	信号隔离	现场与总线之间隔离
保护功能	电源保护	供电电源端提供反接保护功能及浪涌吸收功能
滤波功能	滤波功能	采用硬件滤波与软件滤波相结合的方式
电源功能	电源功能	模块采用 24V DC 供电

模拟量输入模块

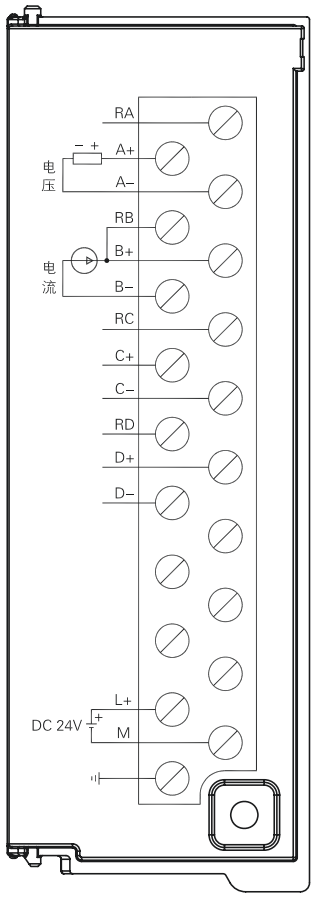
表 C-13 模拟量输入模块规范

特性	AIS-040S1	AIV-080S1	AIC-080S1
输入类型	电压或电流（差分输入）	电压输入	电流输入
输入点数	4	8	
输入量程 电压 电流	 单极性：0~5V，0~10V，双极性：±2.5V，±5V 0~20mA，4~20mA		
最大输入电压	30V DC		
最大输入电流	40mA		
输入阻抗 电压 电流	 ≥2MΩ 250Ω	 ≥2MΩ 250Ω	
数据格式	电压：0~+32000（单极性），-32000~+32000（双极性） 电流：0~+32000		
输入阶跃响应	4 通道 5ms（最快）	8 通道 50ms（最快）	
模块更新频率（所有通道）	4 通道支持 200Hz、100Hz、50Hz、20Hz、10Hz 配置 默认：50Hz 所有通道	8通道支持50Hz、20Hz、10Hz、5Hz、2Hz 配置 默认：10Hz 所有通道（50Hz 仅满足4通道）	
共模抑制	＞40dB		
通道串扰	＞60dB		
共模电压	-12V≤信号电压+共模电压≤+12V		
分辨率	单极性：12bit 双极性：11bit+符号位	单极性：16bit 双极性：15bit+符号位	
测量原理	逐次逼近	Sigma-delta（Σ-△）	
测量误差	0.5%（最大）	0.1%（最大）	
断线检测（仅针对 4~20mA）	断线标定：-32768， 32767 两值可选		
隔离			

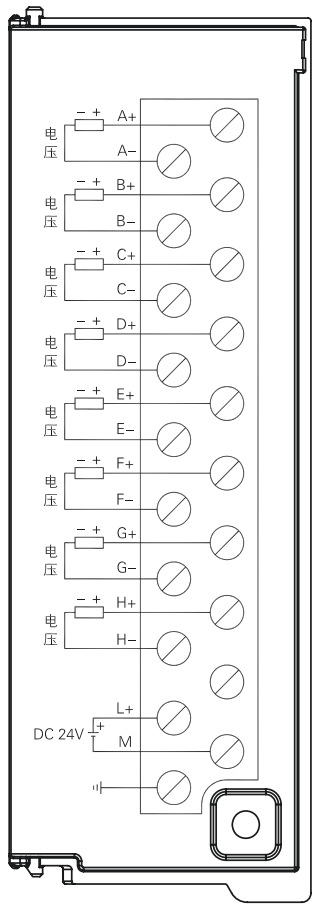
现场至逻辑	500V AC	
24VDC 至逻辑	500V AC	
诊断		
超负量程	单极性：0　双极性：-32768	单极性：0　双极性：-32768
超正量程	单极性：32760　双极性：32752	单极性：32767　双极性：32767
没有电源	32736	32766

接线规格

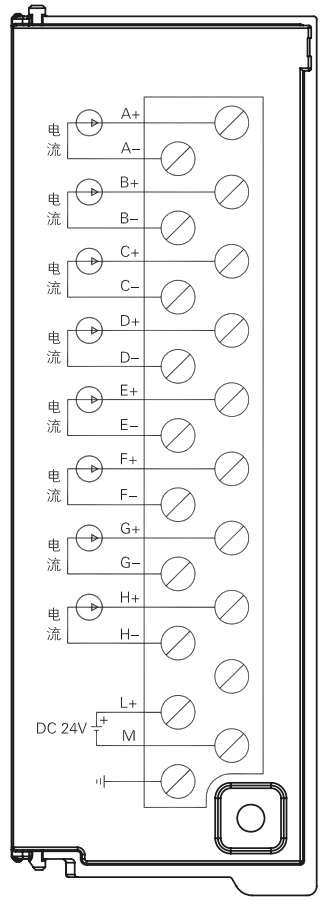
◆ AIS-040S1 接线图



◆ AIV-080S1 接线图



◆ AIC-080S1 接线图



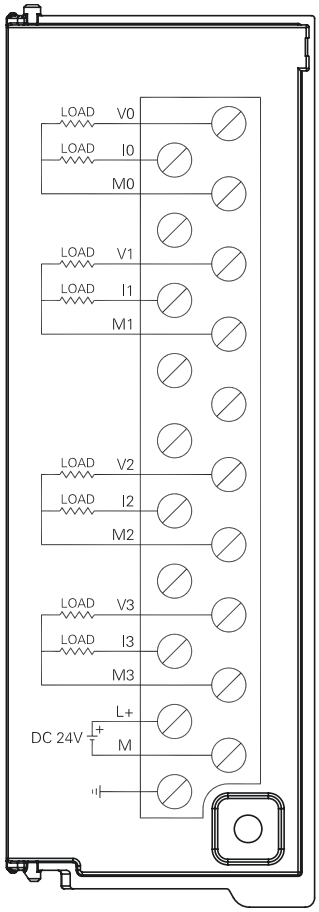
模拟量输出模块

表 C-14　模拟量输出模块规范

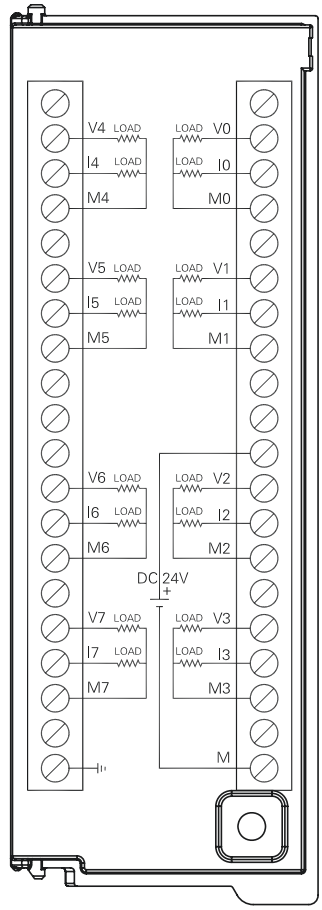
特性	AQS-040S1	AQS-080S1
输出类型	电压或电流	
输出点数	4	8
输出量程 电压 电流	 ±10V 0~20mA， 4mA~20mA	
保护 输出误接电压 电压短路保护	 最大 30V DC 有	
数据格式 电压 电流	满量程时 -32000~+32000 0~+32000	
建立时间 电压输出 电流输出	 100us 2ms	
负载阻抗 电压输出 电流输出	 5000Ω（最小） 500Ω（最大）	
分辨率	单极性：12bit 双极性：11bit+符号位	
精度 电压 电流	 典型值：满量程的±0.5%，最坏情况：满量程的±2% 典型值：满量程的±0.6%，最坏情况：满量程的±2%	
隔离 电源隔离 现场至逻辑	 500V AC 500V AC	

接线规格

◆ AQS-040S1 接线图



◆ AQS-080S1 接线图



模拟量输入输出模块

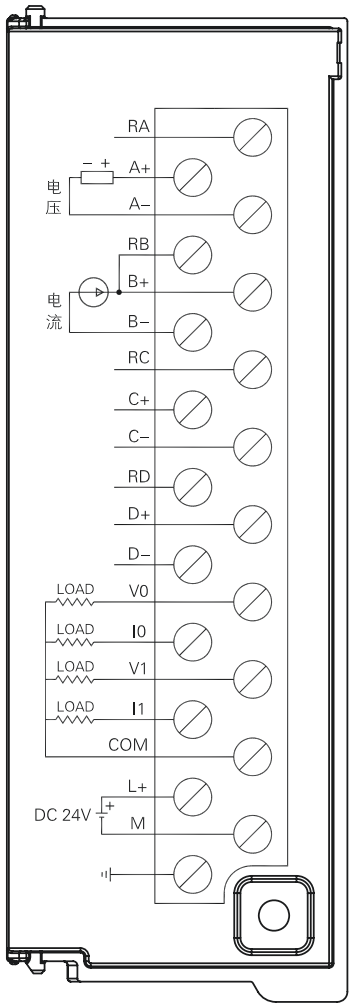
表 C-15　模拟量输入输出模块规范

输入特性	**AMS-060S1**	
输入类型	电压或电流（差分输入）	
输入点数	4	
输入量程 电压 电流	 单极性：0~5V，0~10V 双极性：±2.5V，±5V 0~20mA， 4~20mA	
最大输入电压	30V DC	
最大输入电流	40mA	
输入阻抗 电压 电流	 ≥2MΩ 250Ω	
数据格式	电压：0~+32000（单极性），-32000~+32000（双极性） 电流：0~+32000	
输入阶跃响应	4 通道 5ms（最快）	
模块更新频率（所有通道）	4 通道支持 200Hz、100Hz、50Hz、20Hz、10Hz 配置 默认：50Hz 所有通道	8 通道支持 50Hz、20Hz、10Hz、5Hz、2Hz 配置 默认：10Hz 所有通道

共模抑制	＞40dB
通道串扰	＞60dB
共模电压	-12V≤信号电压+共模电压≤+12V
分辨率	单极性：12bit 双极性：11bit+符号位
测量原理	逐次逼近
测量误差	0.5%（最大）
断线检测（仅针对4~20mA）	断线标定：-32768，32767 两值可选
隔离 现场至逻辑 24VDC 至逻辑	 500V AC 500V AC
诊断 超负量程 超正量程 无电源	 单极性：0，　双极性：-32768 单极性：32760，　双极性：32752 32736
输出特性	**AMS-060S1**
输出类型	电压或电流
输出点数	2
输出量程 电压 电流	 ±10V 0~20mA，　4mA~20mA
保护 输出误接电压 电压短路保护	 最大 30V DC 有
数据格式 电压 电流	满量程时 -32000~+32000 0~+32000
建立时间 电压输出 电流输出	 100us 2ms
负载阻抗 电压输出 电流输出	 5000Ω（最小） 500Ω（最大）
分辨率	单极性：12bit 双极性：11bit+符号位
精度 电压 电流	 典型值：满量程的±0.5%，最坏情况：满量程的±2% 典型值：满量程的±0.6%，最坏情况：满量程的±2%
隔离 电源隔离 现场至逻辑	 500V AC 500V AC

接线规格

◆ AMS-060S1 接线图



C.4 温度模块

表 C-16 温度模块的基本属性

名称	规格描述	订货号
AIT-04 温度模块	热电偶输入模块，4 路 TC，隔离型 16bit 精度	CTH3 AIT-040S1
AIT-08 温度模块	热电偶输入模块，8 路 TC，隔离型 16bit 精度	CTH3 AIT-080S1
AIR-04 温度模块	热电阻输入模块，4 路 RTD，隔离型 16bit 精度	CTH3 AIR-040S1
AIR-08 温度模块	热电阻输入模块，8 路 RTD，隔离型 16bit 精度	CTH3 AIR-080S1

表 C-17 温度模块的常规特性

特性	AIT-040S1	AIT-080S1	AIR-040S1	AIR-080S1
物理特性				
尺寸（W×H×D）	34×115×100 mm			
电源特性				
额定输入电压	24V DC			
输入电压范围	20.4V~28.8V DC			
输入电流	50mA	50mA	60mA	80mA
极性反接保护	有			
总线电源电压	+5V DC			
总线电源电流	50mA	50mA	50mA	50mA

LED 指示灯特性	
24V 电源指示灯	亮起：24VDC 供电正常，熄灭：无 24VDC 供电
SF 指示灯	亮起：模块故障，熄灭：无错，闪烁：输入信号错误

表 C-18　温度模块的功能特性

功能类别	功能名称、标识符	描述
I/O 功能	I/O 接口	提供 4 路/8 路温度传感器输入接口
扩展功能	扩展功能	提供总线扩展功能
隔离功能	信号隔离	现场与总线之间隔离
	电源隔离	外部电源与系统电源之间隔离
保护功能	电源保护	供电电源端提供反接保护功能及浪涌吸收功能
	断线检测	输入提供断线检测功能
滤波功能	滤波功能	采用硬件滤波与软件滤波相结合的方式
电源功能	电源功能	模块采用 24V DC 供电

表 C-19　温度模块的输入特性

特性	AIT-040S1	AIT-080S1	AIR-040S1	AIR-080S1
输入类型	悬浮型热电偶		模块参考接地热电阻	
输入点数	4	8	4	8
接线方式	-		支持 2 线制、3 线制、4 线制； 默认：3 线制	
输入范围	热电偶类型（选一种）： S、T、R、E、N、K、J 电压范围：±80mV 默认：K		热电阻类型（选一种）： Pt-100Ω，200Ω，500Ω，1000Ω （α=3850ppm，3920ppm， 3850.55ppm，3916ppm，3902ppm） Pt-10000Ω（α=3850ppm） Cu-9.035Ω（α=4720ppm） Ni-100Ω，120Ω，1000Ω （α=6720ppm，6178ppm） R-150Ω，300Ω，600ΩFS 默认：Pt-100Ω（α=3850ppm）	
隔离 现场至逻辑 现场至 24VDC 24VDC 至逻辑	 500VAC 500VAC 500VAC			
共模抑制	＞100dB@120VAC			
输入分辨率 温度 电压 电阻	 0.1℃/0.1℉ 15 位+符号位 -		 0.1℃/0.1℉ - 15 位+符号位	
测量原理	Sigma-Delta			
模块更新频率（所有通道）	4 通道支持 8Hz、4Hz、2Hz、1Hz 配置，默认：2Hz 所有通道			
	8 通道支持 4Hz、2Hz、1Hz、0.5Hz 配置，默认：1Hz 所有通道			
到传感器的导线长度	最大 100m			

导线回路电阻	100Ω	20Ω，Cu 型 2.7Ω
噪声抑制	85dB@50Hz/60Hz/400Hz	
数据字格式	电压：-27648~+27648	电阻：-27648~+27648
输入阻抗	＞10MΩ	＞10MΩ
最大输入电压	输入端能支持最大电压为 30V DC 的误接	
分辨率	15 位+符号位	
输入滤波衰减	-3dB@21KHz	-3dB@3.6KHz
基本误差	0.1%Fs（电压）	0.1%Fs（电阻）
重复性	0.05%Fs	
冷端补偿	可配置，默认有冷端补偿	-
冷端误差	±1.5℃	-
温度单位	摄氏度/华氏度可配置，默认摄氏度	
断线检测	热电偶：可配置，默认有断线检测；	热电阻：一直有断线检测，不可配置
	支持正负两个方向标定，默认正向标定	
诊断 断线 无模块电源	 32767（正向标定），-32768（负向标定） 32766	
是否集成有 PID 控制功能	无	-

热电偶特性

各种类型热电偶的温度范围（°C）和准确度

数据字 (1个数字位=0.1°C)		类型J	类型K	类型T	类型E	类型R,S	类型N	±80mV	
十进制	十六进制								
32767	7FFF	>1200.0°C	>1372.0°C	>400.0°C	>1000.0°C	>1768.0°C	>1300.0°C	>94.071mV	OF
↑	↑							↑	↑
32511	7EFF							97.071mV	OR
:	:								
27649	6C01							80.0029mV	
27648	6C00							80mV	NR
:	:								
17680	4510					1768.0°C			
:	:								
13720	3598		1372.0°C 超出范围						
:	:								
13000	32C8		1300.0°C				1300.0°C		
:	:								
12000	2EE0	1200.0°C							
:	:								
10000	2710				1000.0°C				
:	:								
4000	0FA0			-400.0°C		400.0°C			
:	:								
1	0001	0.1°C	0.1°C	0.1°C	0.1°C	0.1°C	0.1°C	0.0029mV	
0	0000	0.0°C	0.0°C	0.0°C	0.0°C	0.0°C	0.0°C	0.0mV	
-1	FFFF	-0.1°C	-0.1°C	-0.1°C	-0.1°C	-0.1°C	-0.1°C	-0.0029mV	
:	:					低于范围			
-500	FE0C					-50.0°C			
-1500	FA24	-150.0°C							
:	:								
-2000	F830	低于范围	-200.0°C						
:	:								
-2100	F7CC	-210.0°C							
:	:		低于范围						
-2550	F60A			-255.0°C	-255.0°C				
:	:			低于范围	低于范围				
-2700	F574		-270.0°C	-270.0°C	-270.0°C		-270.0°C		
:	:								
-27648	9400							-80mV	
-27649	93FF							-80.0029mV	UR
:	:								
-32512	8100							-94.071mV	
#	#							↓	↓
-32768	8000	<-210.0°C	<-270.0°C	<-270.0°C	<-270.0°C	<-50.0°C	<-270.0°C	<-94.07mV	UF
全量程范围的精度		S0.1%	S0.3%	S0.6%	S0.1%	S0.6%	S0.1%	S0.1%	
精度(无冷端补偿的额定范围)		S1.5°C	S1.7°C	S1.4°C	S1.3°C	S3.7°C	S1.6°C	S0.10°C	
冷端误差		S1.5°C	S1.5°C	S1.5°C	S1.5°C	S1.5°C	S1.5°C	NA	
*OF=上溢 OR=超出范围 NR=额定范围 VR=低于范围 UF=下溢									
↑表示所有大于该值但小于断线阀值的模拟量都报告为上溢出值，32767（10x7FFF） ↓表示所有小于该值但大于断线阀值的模拟量都报告为下溢出值，-32768（0x8000）									

热电阻特性

用于热电阻类型的温度范围（°C）和精度

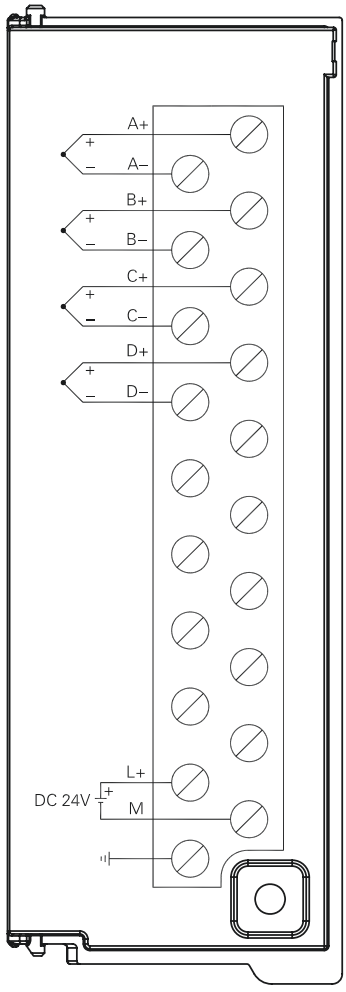
系统字 (1个数字位=0.1°C)		Pt10000	Pt100 Pt200 Pt500 Pt1000	Ni100 Ni120 Ni1000	Cu9.035	0-150Ω	0-300Ω	0-600Ω	
十进制	十六进制								
32767	7FFF								
32766	7FFE					↑	↑	↑	
32511	7EFF					176.383Ω	352.767Ω	705.534Ω	
29649	6C01					150.005Ω	300.011Ω	600.022Ω	
27648	6C00					150.000Ω	300.000Ω	600.000Ω	
25000	61AB								↑
18000	4650								OR
15000	3A98								
13000	32C8	↑	↑						
10000	2710	1000.0°C	1000.0°C						
8500	2134		850.0°C						
6000	1770	600.0°C			↑				
3120	0C30			↑	312.0°C				
2950	0B86			295.0°C					
2600	0A28				260.0°C				
2500	09C4			250.0°C					
1	0001	0.1°C	0.1°C	0.1°C	0.1°C	0.005Ω	0.011Ω	0.022Ω	
0	0000	0.0°C	0.0°C	0.0°C	0.0°C	0.000Ω	0.000Ω	0.000Ω	
-1	FFFF	-0.1°C	-0.1°C	-0.1°C	-0.1°C	（负值是不可能的）			
						↓	↓	↓	
-600	FDA8			-60.0°C					
-1050	FBE6			-105.0°C					
				↓					
-2000	F830	-200.0°C	-200.0°C		-200.0°C				
-2400	F6A0				-240.0°C				
-2430	F682	-243.0°C	-243.0°C		↓				
		↓	↓						
-5000	EC78								
-6000	E890								UR
-10500	D6FC								↓
-12000	D120								
-20000	4E20								
-32767	8001								
-32768	8000								
全量程范围的精度		±0.4%	±0.1%	±0.2%	±0.5%	±0.1%	±0.1%	±0.1%	
精度（额定范围）		±4°C	±1°C	±0.6°C	±2.8°C	±0.15°C	±0.3°C	±0.6°C	

*OF=上溢出 OR=超出范围 NR=额定范围 UR=低于范围 UF=下溢出

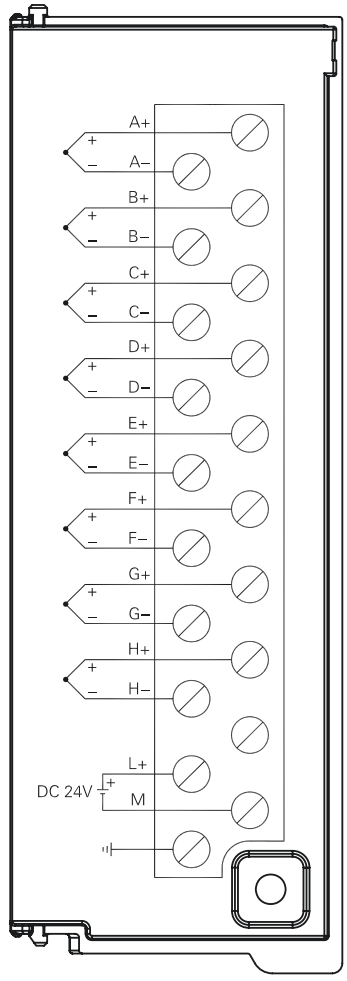
↑或↓表示所有超过或低于这个限制值的模拟量其测量值都报告为所选的熔断标定值，32767（0x7FFF）或-32768（0x8000）

热电偶接线规格

◆ AIT-040S1 接线规格

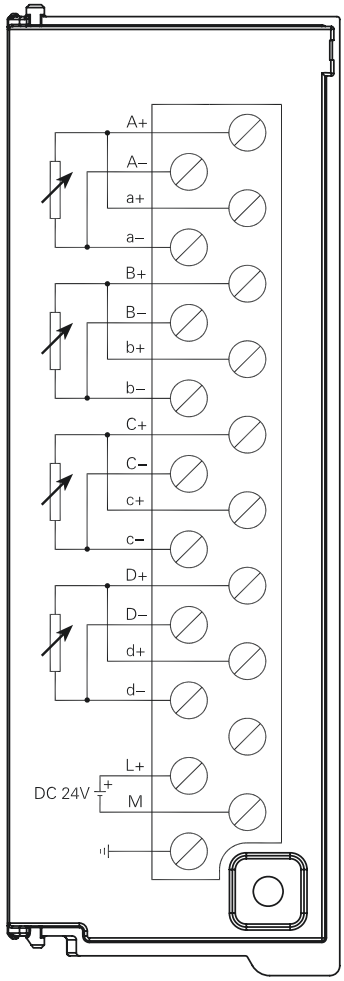


◆ AIT-080S1 接线规格

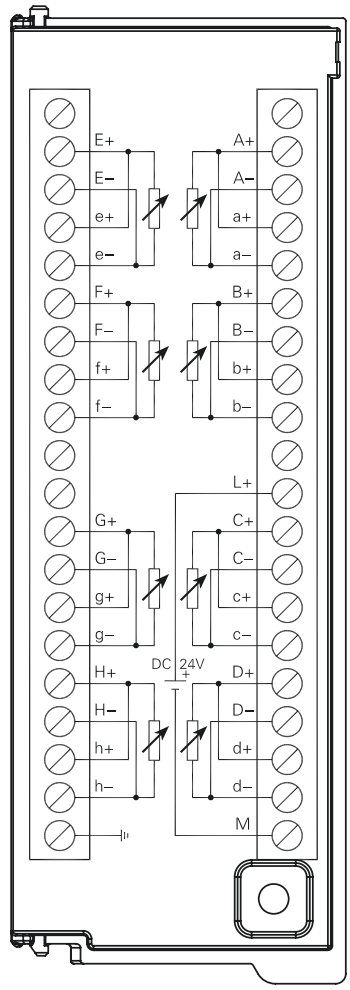


热电阻接线规格

◆ AIR-040S1 接线规格



◆ AIR-080S1 接线规格



C.5 高速计数模块

表 C-20 高速计数模块的基本属性

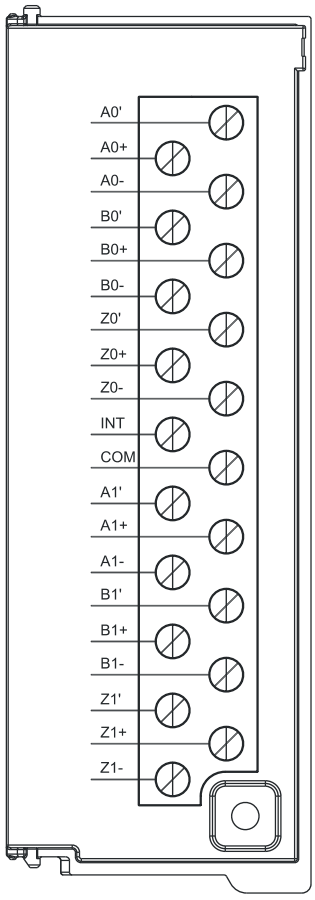
名称	规格描述	订货号
HSC-02高速计数模块	2路差分/单端信号输入	CTH3 HSC-020S1

表 C-21 HSC-02 的常规特性

物理特性		
尺寸（W×H×D）		34×115×100 mm
电源特性		
总线电源电压		+5V DC
总线电源电流		130mA
LED 指示灯特性		
信号指示灯		ON：有输入信号，OFF：无输入信号
传感器连接		
输入通道数		2
信号类型	差分输入	信号电压：5VDC 最高输入频率：2MHz

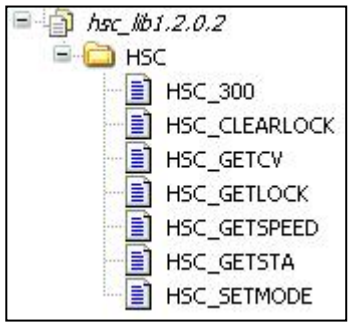
	单端输入	信号电压：24VDC 最高输入频率：500KHz 信号占空比允许范围：40%-60%
信号输入最大保护电压		30VDC
输入滤波		可配置，125KHz/250KHz/500KHz/1MHz/2MHz
正交易码		1、2、4 倍频
计数器格式		32 位
计数器清零功能		有，Z 信号
计数器捕捉功能		有，Z 信号
多计数器同步计数功能		有，INT 信号
INT 信号电压		24VDC
INT 信号最高输入频率		500KHz
INT 信号输入滤波		可配置，125KHz/250KHz/500KHz
光电隔离		500VAC，1min

接线图



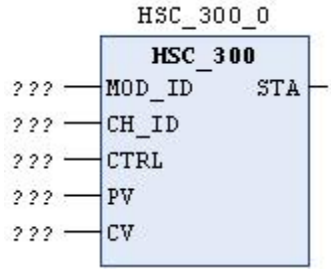
支持的库指令(HSC_LIB)

高速计数模块 HSC-02 支持指令库（HSC_LIB），各指令的描述参考如下说明。



1、设置计数器指令

函数名：HSC_300



功能：设置计数器。

参数

参数名	输入输出属性	参数描述	类型	数值范围	备注
MOD_ID	IN	模块地址	DWORD		模块映射字节中的 Module Id
CH_ID	IN	通道	BYTE	0~1	
CTRL	IN	控制字	BYTE		见下表
PV	IN	当前值	DWORD		
CV	IN	预设值	DWORD		
STA	OUT	返回状态	BYTE	0~255	0：OK，其它：访问模块出错

控制字（R/W）

7	6	5	4	3	2	1	0
hsc_en	hsc_cv_update	hsc_pv_update	hsc_dir_update	hsc_dir	quad_rate		reset_level

reset_level：复位电平，0-高电平复位，1-低电平复位

f quad_rate[1:0]：正交计数选择，00-4x 倍数，01-2x 倍速，10-1x 倍速

hsc_dir：计数方向，0-减计数，1-增计数

hsc_dir_update：计数方向更新，0-不更新，1-更新

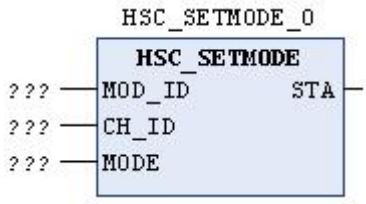
hsc_pv_update：预设值更新，0-不更新，1-更新

hsc_cv_update：当前值更新，0-不更新，1-更新

hsc_en：计数使能，0-不使能，1-使能

2、设置计数器模式

函数名：HSC_SETMODE



功能：设置计数器模式。

参数

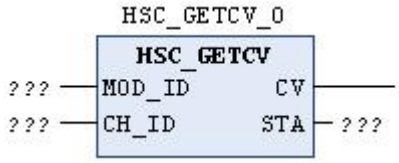
参数名	输入输出属性	参数描述	类型	数值范围	备注
MOD_ID	IN	模块地址	DWORD		模块映射字节中的 Module Id
CH_ID	IN	通道	BYTE	0~1	
MODE	IN	控制字	BYTE		Bit0~Bit3：HSC 计数模式（模式描述见表 9-7） Bit4：Z 信号锁存功能，0：锁存，1：不锁存 Bit5：Z 信号清零功能，0：清零，1：不清零 Bit6：预留 Bit7：锁存值清零 0：无效，1：有效
STA	OUT	返回状态	BYTE	0~255	0：OK，其它：访问模块出错

HSC 计数模式

模式	描述	输入			软件控制
	HSC0	A0	B0	Z0	INT
	HSC1	A1	B1	Z1	
0	具有内部方向控制的单相计数器	时钟			
1		时钟		重设	
2		时钟		重设	启动（外部同步）
3	具有外部方向控制的单相计数器	时钟	方向		
4		时钟	方向	重设	
5		时钟	方向	重设	启动（外部同步）
6	具有 2 个时钟输入的双相计数器	向上时钟	向下时钟		
7		向上时钟	向下时钟	重设	
8		向上时钟	向下时钟	重设	启动（外部同步）
9	A/B 相正交计数器	时钟 A	时钟 B		
10		时钟 A	时钟 B	重设	
11		时钟 A	时钟 B	重设	启动（外部同步）

3、获取当前计数值

函数名：HSC_GETCV



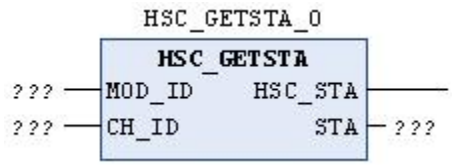
功能：获取当前计数值

参数

参数名	输入输出属性	参数描述	类型	数值范围	备注
MOD_ID	IN	模块地址	DWORD		模块映射字节中的 Module Id
CH_ID	IN	通道	BYTE	0~1	
CV	OUT	当前计数值	DWORD		当前计数值
STA	OUT	返回状态	BYTE	0~255	0：OK，其它：访问模块出错

4、获取当前计数状态

函数名：HSC_GETSTA



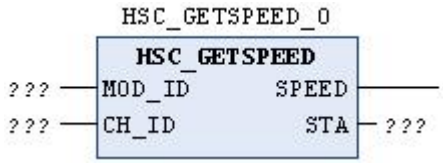
功能：获取当前计状态

参数

参数名	输入输出属性	参数描述	类型	数值范围	备注
MOD_ID	IN	模块地址	DWORD		模块映射字节中的 Module Id
CH_ID	IN	通道	BYTE	0~1	
HSC_STA	OUT	计数器状态	BYTE		Bit0~Bit3：当前模式 Bit4：预留 Bit5：HSC0当前计数方向位：1=增计数 Bit6=1：当前值等于预设值位 Bit7=1：当前值大于预设值位
STA	OUT	返回状态	BYTE	0~255	0：OK，其它：访问模块出错

5、获取当前速度

函数名：HSC_GETSPEED



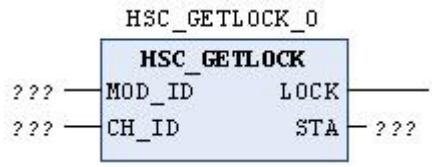
功能：获取当前速度

参数

参数名	输入输出属性	参数描述	类型	数值范围	备注
MOD_ID	IN	模块地址	DWORD		模块映射字节中的 Module Id
CH_ID	IN	通道	BYTE	0~1	
SPEED	OUT	当前速度	DWORD		Hz
STA	OUT	返回状态	BYTE	0~255	0：OK，其它：访问模块出错

6、获取当前锁存值

函数名：HSC_GETLOCK



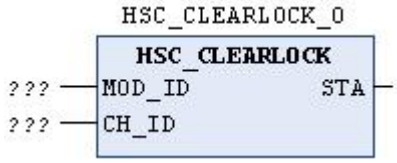
功能：获取当前计数值

参数

参数名	输入输出属性	参数描述	类型	数值范围	备注
MOD_ID	IN	模块地址	DWORD		模块映射字节中的 Module Id
CH_ID	IN	通道	BYTE	0~1	
LOCK	OUT	当前锁存值	DWORD		当前锁存值
STA	OUT	返回状态	BYTE	0~255	0：OK，其它：访问模块出错

7、清除锁存值

函数名：HSC_CLEARLOCK



功能：清除锁存值。

参数

参数名	输入输出属性	参数描述	类型	数值范围	备注
MOD_ID	IN	模块地址	DWORD		模块映射字节中的 Module Id

CH_ID	IN	通道	BYTE	0~1	
STA	OUT	返回状态	BYTE	0~255	0：OK，其它：访问模块出错

C.6 脉冲输出模块

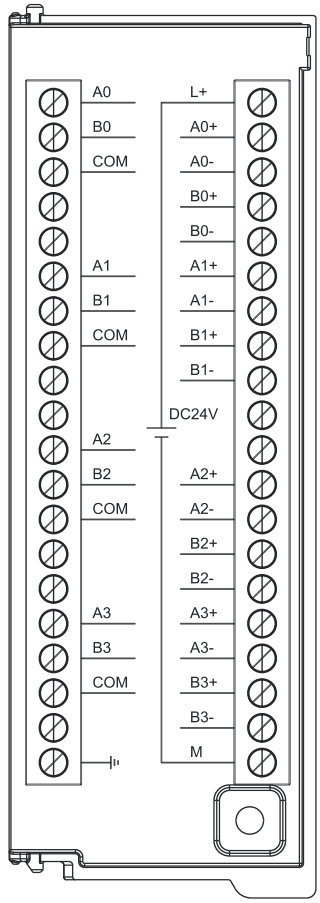
表 C-22 脉冲输出模块的基本属性

名称	规格描述	订货号
HSP-04脉冲输出模块	4路差分/单端信号输出	CTH3 HSP-040S1

表 C-23 HSP-04 的常规特性

物理特性		
尺寸（W×H×D）	34×115×101.6 mm	
电源特性		
额定输入电压	24V DC	
输入电压范围	20.4V~28.8V DC	
输入电流	100mA	
极性反接保护	有	
总线电源电压	+5V DC	
总线电源电流	100mA	
LED 指示灯特性		
信号指示灯	ON：有输入信号，OFF：无输入信号	
输出特性		
输出通道数	4	
输出类型	差分信号	单端（NPN）信号
最高输出频率	4MHz	500KHz
输出信号占空比	-	50%
额定输出电压	5VDC	5~24VDC
输出电压范围	0~5.5VDC	5~28.8VDC
输出信号逻辑“0”	3.8V（最小）	0.5V（最大）
输出信号逻辑“1”	0.3V（最大）	Vcc~0.5V（最小）
浪涌电流	8A，持续 100ms	
每点电流（最大）	20mA	20mA
每个公共端最大电流	无	160mA
漏电流（最大）	10μA	
隔离	500VAC，1min	

接线图



C.7 CAN-1T 主站模块（C3X-00 系列支持）

表 C-24 CAN-1T 模块的基本属性

名称	规格描述	订货号
CAN-1T 主站模块	CANopen 主站通信模块 1 个 CAN 通信口	CTH3 CAN-1T0S1

表 C-25 常规特性

物理特性	
尺寸（W×H×D）	34×115×101.6 mm
功耗	2.5W
电源特性	
额定输入电压	24V DC
输入电压范围	20.4V~28.8V DC
输入电流	100mA
极性反接保护	有
总线电源电压	+5V DC
总线电源电流	100mA
LED 指示灯特性	
24V 电源指示灯	亮起：24VDC 供电正常，熄灭：无 24VDC 供电
SF 指示灯	亮起：模块故障，熄灭：无错

表 C-26 功能特性

功能类别	功能项	描述
扩展功能	扩展功能	提供总线扩展功能
通讯功能	CAN 接口	提供 1 个 CANopen 主站接口
隔离功能	电源隔离	外部电源与系统电源之间隔离
	通讯隔离	CAN 口隔离
保护功能	电源保护	供电电源端提供反接保护功能及浪涌吸收功能
	接口保护	通讯端口防雷保护

表 C-27　通信口规范

CANopen 通讯							
通讯接口	1 个 CAN 通讯口						
最大主站点数	1 个 CPU 后面最多可连接 8 个主站						
最大从站点数	1 个主站后面最多可连接 32 个从站						
传输速率（kbit/s）	1000	800	500	250	125	50	20
最大长度（m）	25	50	100	250	500	1000	2500
隔离	通信口隔离						

CAN 通信口采用带屏蔽的网线作为通信线，可供选择的网线类型为 22AWG~25AWG，其规格标准如下表，电阻值为单根导线的直流电阻值，推荐使用全屏蔽五类线或者全屏蔽超五类线，24AWG。

表 C-28　通信口线材规格

AWG	外径公制 mm	外径英制 inch	截面积 mm^2	电阻值 W/km
22	0.643	0.0253	0.3247	54.3
23	0.574	0.0226	0.2588	48.5
24	0.511	0.0201	0.2047	89.4
25	0.44	0.0179	0.1624	79.6

接口示意图

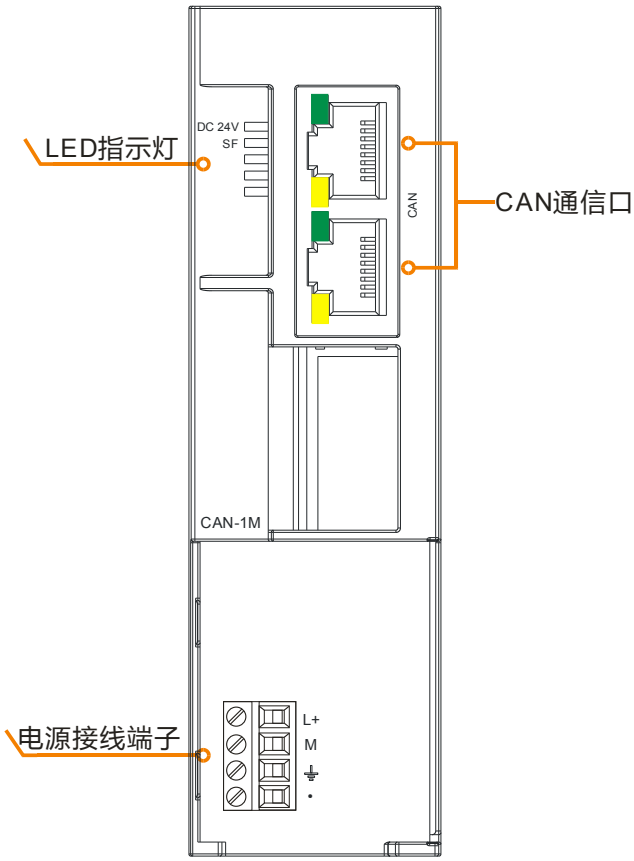
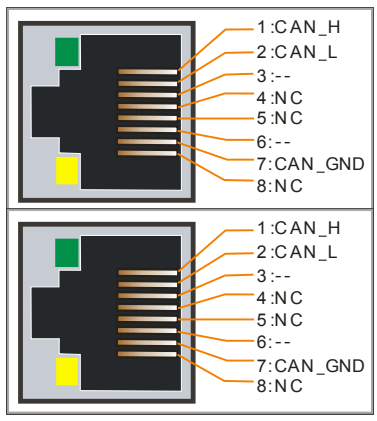


表 C-29　输入电源接线端子定义

4 位可拆卸端子	信号	信号定义
L+ M ⏚ •	L+	24V 电源正
	M	24V 电源负
	⏚	大地
	--	--

表 C-30　CAN 通信口定义



位号	信号	信号定义
1	CAN_H	CAN 数据发送正端
2	CAN_L	CAN 数据发送负端
3	--	--
4	--	--
5	--	--
6	--	--
7	CAN_GND	CAN 的地
8	--	--
连接器外壳	PE	机壳接地

C.8　中继模块

表 C-31　中继模块的基本属性

规格描述	订货号
INT-00中继扩展模块	CTH3 INT-000S1

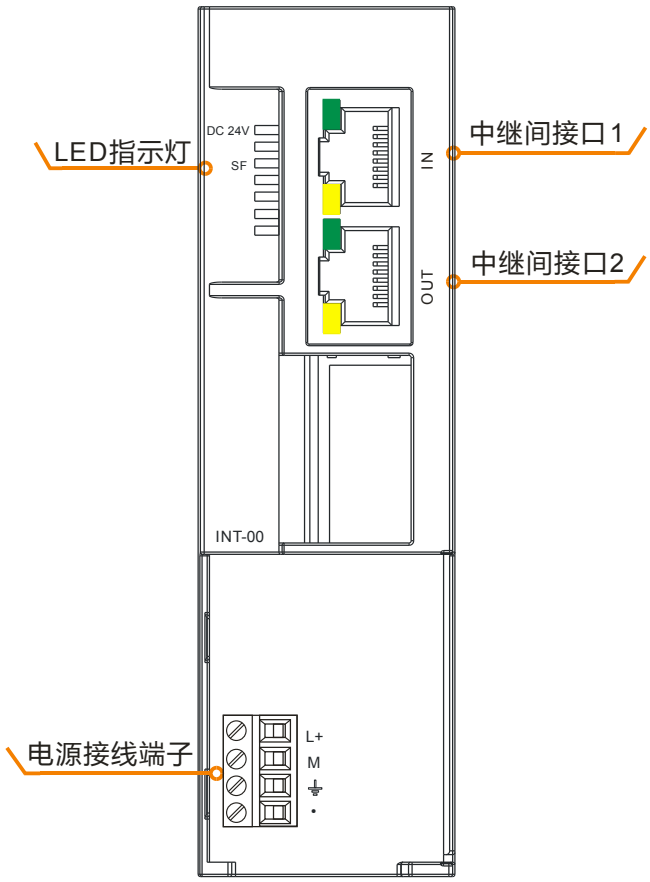
表 C-32　INT-00 的常规特性

物理特性	
尺寸（W×H×D）	34×115×101.6 mm
功耗	19.5W
电源特性	
额定输入电压	24V DC
输入电压范围	20.4V~28.8V DC
输入电流	0.8A
极性反接保护	有
总线电源电压	+5V DC
总线电源电流	1.6A
LED 指示灯特性	
24V 电源指示灯	亮起：24VDC 供电正常，熄灭：无 24VDC 供电
SF 指示灯	亮起：模块故障，熄灭：无错

表 C-33　INT-00 的功能特性

功能类别	功能项	描述
扩展功能	扩展总线接口	提供总线扩展功能
通讯功能	中继间接口	提供中继之间通信接口
扩展功能	扩展功能	提供允许扩展 8 个 I/O 模块
隔离功能	电源隔离	外部电源与系统电源之间隔离
保护功能	电源保护	供电电源端提供反接保护功能及浪涌吸收功能

接口示意图



请按照以下说明连接中继模块，如果接错则会导致通信失败。

中继间接口 1（IN）：连接上一个中继模块的接口（若为第一个中继模块，此口不接）。

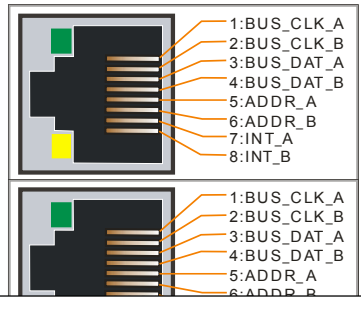
中继间接口 2（OUT）：连接下一个中继模块的接口（若为最后一个中继模块，此口不接）。

接口定义

表 C-34　INT-00 的电源接口定义

4 位可拆卸端子	位号	信号	信号定义
L+ M ⏚ •	1	L+	24V 电源正
	2	M	24V 电源负
	3	⏚	大地
	4	--	--

表 C-35　INT-00 的双 RJ45 接口定义



位号	信号	信号定义
1	BUS_CLK_A	总线时钟
2	BUS_CLK_B	总线时钟
3	BUS_DAT_A	总线数据
4	BUS_DAT_B	总线数据
5	ADDR_A	配置地址
6	ADDR_B	配置地址

	7	INT_A	中断
	8	INT_B	中断
	连接器外壳	PE	机壳接地

C.9 EtherCAT 从站模块

表 C-36 EtherCAT 从站模块的基本属性

名称	规格描述	订货号
EtherCAT 从站模块	最多可挂接8个扩展模块，仅限数字量模块、模拟量模块、温度模块	CTH3 ECT-000S1

关于该模块的具体应用，请访问合信官网下载《CTH300 系列 EtherCAT 从站模块使用手册》，网址：http://www.co-trust.com。

表 C-37 常规特性

物理特性	
尺寸（W×H×D）	34×115×100 mm
功耗	2.5W
电源特性	
额定输入电压	24V DC
输入电压范围	20.4V~28.8V DC
输入电流	0.8A
极性反接保护	有
总线电源电压	+5V DC
总线电源电流	1.6A
LED 指示灯特性	
24V 电源指示灯（绿色）	亮起：24VDC 供电正常，熄灭：无 24VDC 供电
SF 指示灯（红色）	亮起：EtherCAT 从站模块故障，熄灭：无错
BF 指示灯（红色）	亮起：扩展总线故障，熄灭：无错
LINK 指示灯（绿色）(从站状态指示灯)	亮起：正常工作(8)，闪烁：预操作(2)、安全操作(4)（参见**注 1**），熄灭：没有连接(0)、初始化(1)
RJ45 口指示灯(绿色)	亮起：与其他 EtherCAT 接口连接 熄灭：没有与其他 EtherCAT 接口连接 闪烁：与其他 EtherCAT 接口进行通信

注 1：当从站扩展总线无输出类型的模块时，从站与主站之间通信断开，从站不会切换至安全操作。

表 C-38 功能特性

功能类别	功能项	描述
硬件组态	Codesys 或其他第三方组态	添加的 ECT-00 模块支持 8 个槽位的扩展
扩展功能	扩展功能	允许扩展 8 个 I/O 模块，仅限数字量模块、模拟量模块、温度模块
通讯功能	总线接口	提供扩展模块接口，支持 CTH300 PLC 自定义 55MHZ 总线协议
	EtherCAT 接口	提供 EtherCAT 通信接口，支持 CANopen over EtherCAT (CoE)
隔离功能	电源隔离	外部电源与系统电源之间隔离

保护功能	电源保护	供电电源端提供反接保护功能及浪涌吸收功能

表 C-39 通信口规范

EtherCAT 通讯	
通讯接口	1 个双 RJ45 口
波特率	100Mbps
协议类型	CANopen over EtherCAT (CoE)
	支持 PDO 服务
	支持 SDO 服务
	支持 EtherCAT 状态机命令
	支持第三方 EtherCAT 主站
从站间通信距离最长	100m (100BASE-TX)
隔离	通信口隔离

EtherCAT 通信口采用带屏蔽网线作为通信线，可供选择的网线类型为 22AWG~25AWG，其规格标准如下表，电阻值为单根导线的直流电阻值，推荐使用全屏蔽五类线或者全屏蔽超五类线，24AWG。

表 C-40 通信口线材规格

AWG	外径公制 mm	外径英制 inch	截面积 mm^2	电阻值 W/km
22	0.643	0.0253	0.3247	54.3
23	0.574	0.0226	0.2588	48.5
24	0.511	0.0201	0.2047	89.4
25	0.44	0.0179	0.1624	79.6

推荐采用超五类屏蔽水晶头，如下图：



接口示意图

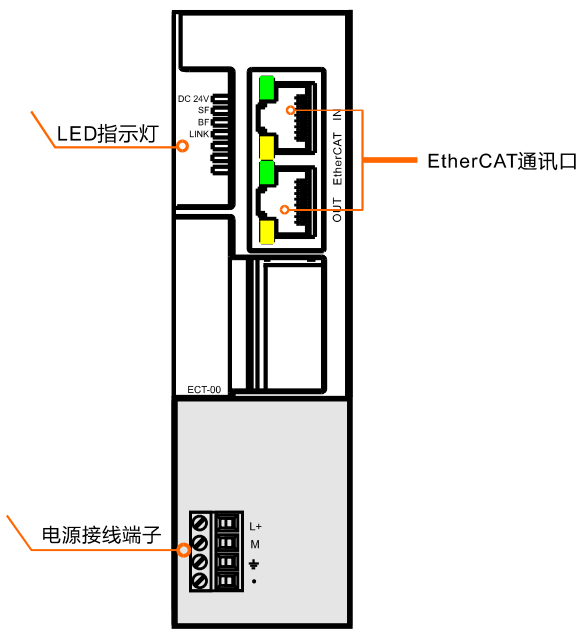


表 C-41　电源接口定义

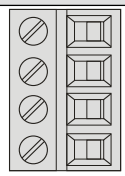
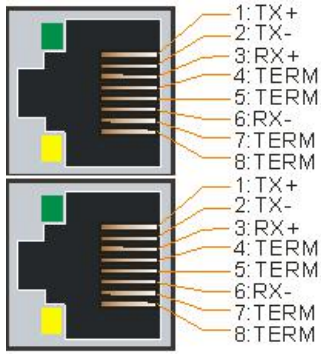
4 位可拆卸端子	位号	信号	信号定义
L+ M ⏚ • 	1	L+	24V 电源正
	2	M	24V 电源负
	3	⏚	大地
	4	--	--

表 C-42　双 RJ45 接口定义

双 RJ45 网口	位号	信号	信号定义
	1	TX+	数据发送正端
	2	TX-	数据发送负端
	3	RX+	数据接收正端
	4	--	--
	5	--	--
	6	RX-	数据接收负端
	7	--	--
	8	--	--
	连接器外壳	PE	机壳接地

D　IO 模块通道配置控制字意义

D.1　数字量输入模块通道配置

每组（8 通道）配置参数格式为：

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保留	通道 4-7 滤波时间			保留	通道 0-3 滤波时间		

滤波时间：

0：0.20ms

1：0.40ms

2：0.80ms

3：1.60ms

5：3.20ms

6：6.40ms（默认值）

7：12.8ms

通道控制字节默认值：16#66

D.2　数字量输出模块通道配置

每组（8 通道）配置参数格式为：

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保留	保留	保留	保留	保留	保留	保留	保留

通道控制字节默认值：16#00

D.3 AI 模块通道配置

模拟量输入一组参数配置格式：

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
采样周期			输入类型和量程				
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
保留	保留	保留	保留	保留	保留	保留	

输入类型和量程：

通过 bit4 来区分单极性和双极性，bit4 为 1 表示双极性；为 0 表示单极性，具体编码如下：

模块类型	输入类型	输入量程	量程编码（BIT4~0）
AI 模块	电压	0~5V	00000
		0~10V（默认值）	00001
		±2.5V	10000
		±5V	10001
	电流	0~20mA（默认值）	00010
		4~20mA	00011

采样周期：

模块类型	更新频率（采样周期）	采样周期编码（BIT7~5）
AI 模块 4 通道	200 Hz	000
	100 Hz	001
	50 Hz（默认值）	010
	20 Hz	011
	10 Hz	100
AI 模块 8 通道	50 Hz	000
	20 Hz	001
	10 Hz（默认值）	010
	5 Hz	011
	2 Hz	100

通道默认值：16#0041（电压 0～10V，4 通道，50Hz）

D.4 TC 模块通道配置

温度模块一组参数配置格式：

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
采样周期			输入类型和量程				
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
温度模块组关闭位	配置使能位，必须为 1	温度模块冷端补偿	温度模块正负标定	温度模块断线检测	温度模块温度单位	温度模块接线方式	

注：组关闭位 0 - 使能

输入类型和量程：

模块类型	输入类型	输入量程	量程编码（BIT4~0）
TC	TC	S	00000
		T	00001
		R	00010
		E	00011
		N	00100
		K（默认值）	00101
		J	00110
	电压	±80mV	10000

采样周期：

模块类型	更新频率（采样周期）	采样周期编码（BIT7~5）
TC 4 通道	8 Hz	000
	4 Hz	001
	2 Hz（默认值）	010
	1 Hz	011
TC 8 通道	4 Hz	000
	2 Hz	001
	1 Hz（默认值）	010
	0.5 Hz	011

接线方式	0：三线制（默认值） 1：二线制 2：四线制
温度单位	0：degrees Celsius（默认值） 1：degrees Fahenheit
断线检测	0：要断线检测（默认值） 1：不要断线检测， 温度模块要配置此项参数，AI 模块只有 4～20mA 量程段需要配置此项参数
正负标定	0：正标定（默认值） 1：负标定 温度模块要配置此项参数，AI 模块只有 4～20mA 量程段需要配置此项参数
冷端补偿	0：要冷端补偿（默认值） 1：不要冷端补偿
组使能	0：使能（默认值） 1：不使能

通道默认值：16#4045（K，4 通道，2Hz，degrees Celsius，要断线检测，正标定，要冷端补偿，使能）

D.5 RTD 模块通道配置

温度模块输入一组参数配置格式：

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
采样周期			输入类型和量程				

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
温度模块组关闭位	配置使能位，必须为 1	温度模块冷端补偿	温度模块正负标定	温度模块断线检测	温度模块温度单位	温度模块接线方式	

注：组关闭位 0 - 使能

输入类型和量程：

模块类型	输入类型	输入量程	量程编码（BIT4~0）
RTD	RTD	Pt-100Ω(α=3850ppm)（默认值 ）	00000
		Pt-200Ω(α=3850ppm)	00001
		Pt-500Ω(α=3850ppm)	00010
		Pt-1000Ω(α=3850ppm)	00011
		Pt-100Ω(α=3920ppm)	00100
		Pt-200Ω(α=3920ppm)	00101
		Pt-500Ω(α=3920ppm)	00110
		Pt-1000Ω(α=3920ppm)	00111
		Pt-100Ω(α=3850.55ppm)	01000
		Pt-200Ω(α=3850.55ppm)	01001
		Pt-500Ω(α=3850.55ppm)	01010
		Pt-1000Ω(α=3850.55ppm)	01011
		Pt-100Ω(α=3916ppm)	01100
		Pt-200Ω(α=3916ppm)	01101
		Pt-500Ω(α=3916ppm)	01110
		Pt-1000Ω(α=3916ppm)	01111
		Pt-100Ω(α=3902ppm)	10000
		Pt-200Ω(α=3902ppm)	10001
		Pt-500Ω(α=3902ppm)	10010
		Pt-1000Ω(α=3902ppm)	10011
		Pt-10000Ω(α=3850ppm)	10100
		Cu-9.035Ω(α=4720ppm)	10101
		Ni-10Ω(α=6720ppm)	10110
		Ni-120Ω(α=6720ppm)	10111
		Ni-1000Ω(α=6720ppm)	11000
		Ni-10Ω(α=6178ppm)	11001
		Ni-120Ω(α=6178ppm)	11010
		Ni-1000Ω(α=6178ppm)	11011
	电阻	R-150Ω	11100
		R-300Ω	11101
		R-600ΩFS	11110

采样周期：

模块类型	更新频率（采样周期）	采样周期编码（BIT7~5）
RTD 4 通道	8 Hz	000
	4 Hz	001
	2 Hz（默认值）	010

	1 Hz	011
RTD 8 通道	4 Hz	000
	2 Hz	001
	1 Hz（默认值）	010
	0.5 Hz	011

接线方式	0：三线制（默认值） 1：二线制 2：四线制
温度单位	0：degrees Celsius（默认值） 1：degrees Fahenheit
断线检测	0：要断线检测（默认值） 1：不要断线检测 温度模块要配置此项参数，AI 模块只有 4～20mA 量程段需要配置此项参数
正负标定	0：正标定（默认值） 1：负标定 温度模块要配置此项参数，AI 模块只有 4～20mA 量程段需要配置此项参数
冷端补偿	0：要冷端补偿（默认值） 1：不要冷端补偿
组使能	0：使能（默认值） 1：不使能

通道默认值：16#4040（Pt-100Ω(α=3850ppm)，4 通道，2Hz，三线制，degrees Celsius，要断线检测，正标定，要冷端补偿，使能）

D.6 模拟量输出模块通道配置

模拟量输出一组类型配置格式：

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保留		保留	电压/电流	量程（含极性）			
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
保留							

类型：

0：电压（默认值）

1：电流

量程：

0：±10V（电压）

1：0~20mA（电流）

2：4~20mA（电流）

默认值：±10V（电压）/0~20mA（电流）

通道默认值：16#0000

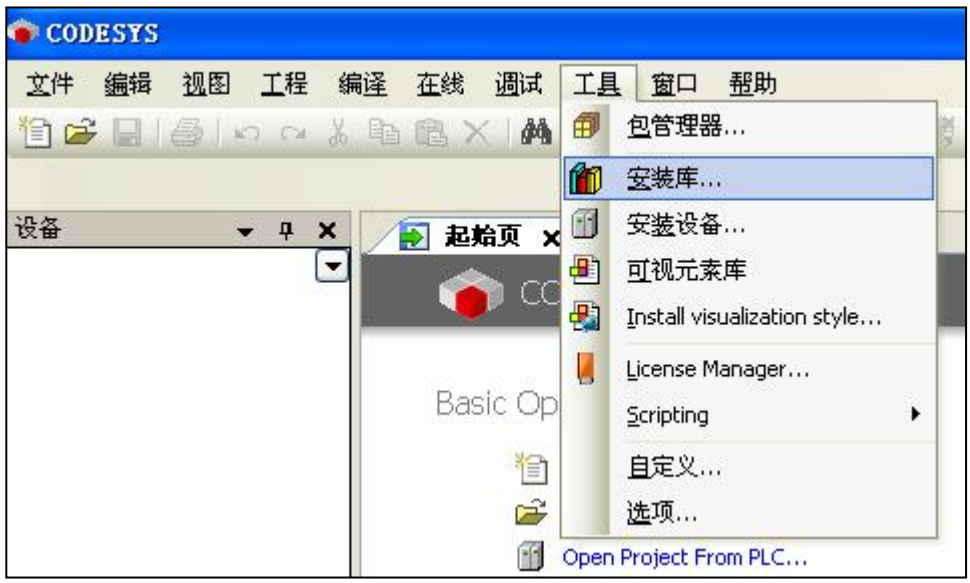
E CT_MODBUS 库的使用介绍

E.1 安装 CTModbus 库文件

选择进行 Modbus 通信时，首先请安装 CT_MODBUS 库文件（库文件请于 COTRUST 公司网站下载：http://www.co-trust.com），然后执行“添加库”即可使用该库指令。

1）安装库

启动 CODESYS 软件，选择菜单项“工具”→“安装库”：



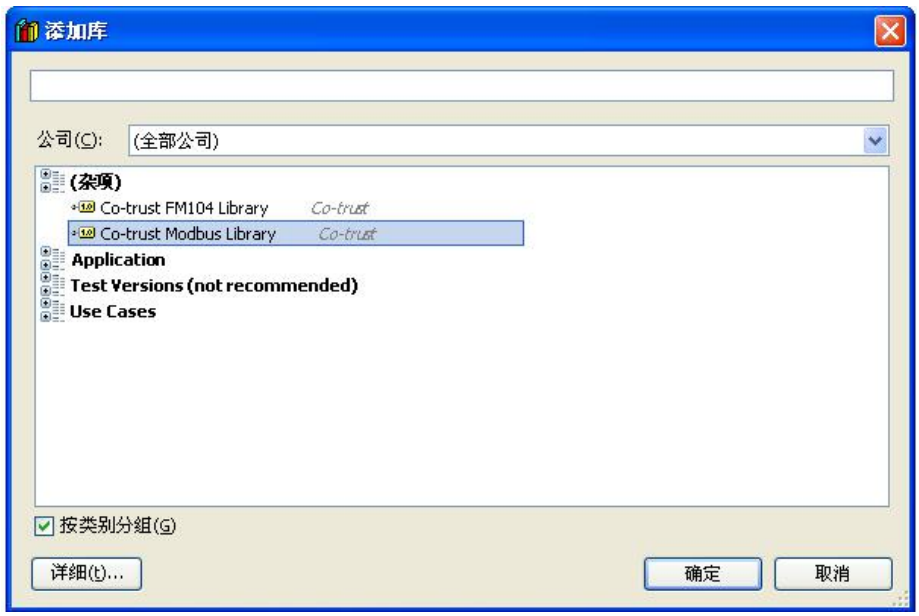
在弹出的窗口中浏览到 Ct_Modbus.library 所在的文件夹，选中“Ct_Modbus.library”，点击“打开”按钮。如果弹出如下提示，即表示 CT_MODBUS 库已安装成功。



2）添加库

CTMODBUS 库文件安装成功后，若需要在当前工程中使用该库，则需要执行“添加库”。具体操作步骤如下：

在设备视图中展开“Device”→“PLC”→“Application”，双击打开“库管理器”，然后在库管理器对话框点击“Add Library”按钮即弹出如下“添加库”对话框：



展开以上对话框中的“杂项”，选中子列表中的“Co-trust Modbus Library”并点击“确定”按钮，即可将 CT_MODBUS 库添加到当前工程中。

E.2 Modbus_RTU 库文件

1）Modbus_RTU 主站 MBUS_CTRL 使用说明

使用 MBUS_CTRL 指令完成主站的初始化，并启动其功能控制。

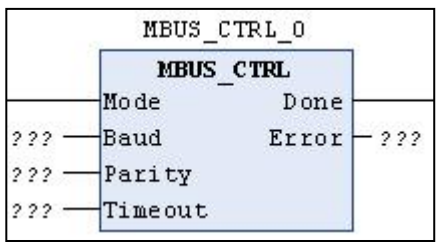


表 E-1 MBUS_CTRL 指令的参数说明

参数	说明	数据类型	范围
Mode	Modbus 使能/不使能 1：使能 Modbus 主站协议 0：不使能 Modbus 主站协议（重新启用 PPI 系统协议）	位	
Baud	设置波特率	双字	1200、2400、4800、9600 19200、38400、57600、 115200
Parity	设置校验 0：无校验 1：奇校验 2：偶校验	字节	
Timeout	主站等待从站响应的时间， 单位：毫秒 ms 典型的设置值为 1000 毫秒（1 秒）。 此值应该设为足够大，以便从站设备对所选的波特率及时响应。	整型	1~ 32767

	超时时间的设定用户判断从站设备是否响应请求。 在最后一个请求字节发出后，超时时间值决定了 Modbus 主站等待第一个字符回复的时间。		
Done	始终为 1	位	
Error	错误代码见下表（Done 为 1 时生效）	字节	

表 E-2　错误代码

错误代码	意义
0	无错误
1	校验选择非法
2	波特率选择非法
3	超时时间非法
4	模式选择非法
9	硬件错误

2）主站 MBUS_MSG 使用说明

使用 Modbus RTU 主站读写子程序 MBUS_MSG 指令，First 接通发送一个 Modbus 请求。同一时刻只能使能一个读写功能（即 MBUS_MSG）。

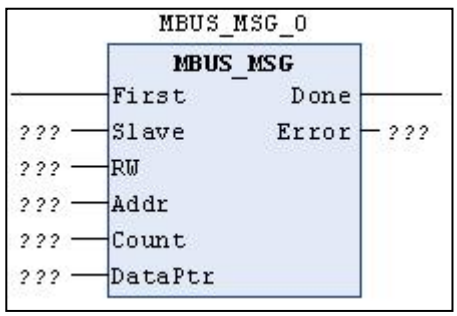


表 E-3　MBUS_MSG 指令的参数说明

参数	说明	数据类型	范围
First	消息读写请求标志，程序运行周期中设置为 1 表示有新的读写请求	位	
Slave	设置 Modbus 从站地址 0：广播地址 广播地址（0）仅能用于写请求。	字节	0~247
RW	R/W 读写 0：Read（读） 1：Write（写）	字节	
Addr	数字量输出和保持寄存器均支持读写操作。 数字量输入和模拟量输入寄存器只能读。 Addr Modbus 数据地址（i.e. 40001） 00001 ~ 00xxx：开关量输出 10001 ~ 10xxx：开关量输入 30001 ~ 30xxx：模拟量输入寄存器 40001 ~ 40xxx：保持寄存器	双字	

Count	通讯的数据个数（位或字的个数） 00xxx：读或写的位个数 10xxx：读或写的位个数 30xxx：输入寄存器读的字个数 40xxx：保持寄存器读或写的字个数 Modbus 主站最多可读或写 120 个字的数据。	整型	
DataPtr	数据指针 如果是读指令，读回的数据放到这个数据区中；如果是写指令，要写出的数据放到这个数据区中。 位数据（地址 0xxxx 和 1xxxx）区域可以作为包字节进行读写，8 位被打包进一个字节。	双字	
Done	信息完成位 0：信息执行中 1：完成	位	
Error	错误代码见下表（Done = 1 时生效）		

表 E-4　错误代码

错误代码	意义
0	无错误
1	响应校验错误
2	未用
3	接收超时（从站无响应）
4	请求参数错误（从站地址，Modbus 地址，count，RW）
5	Modbus/自由口未使能
6	Modbus 正在忙于其它请求
7	响应错误（响应非请求的操作）
8	响应 CRC 校验和错误
9	硬件错误
101	从站不支持请求的功能
102	从站不支持数据地址（起始地址的长度范围）
103	从站不支持数据类型
104	从站设备故障
105	从站接收了信息，但是响应被延迟
106	从站忙，拒绝了该信息
107	从站拒绝了信息
108	从站存储器奇偶校验错误

<备注> 同一时间，只能有一个 MBUS_MSG 可以被激活。如果多个 MBUS_MSG 同时被激活，MBUS_MSG 将返回错误代码 6。

3）从站 MBUS_INIT 使用说明

MBUS_INIT 指令用于初始化或禁止 Modbus 通讯。MBUS_INIT 指令必须无错误的执行，然后才能够使用 MBUS_SLAVE 指令。

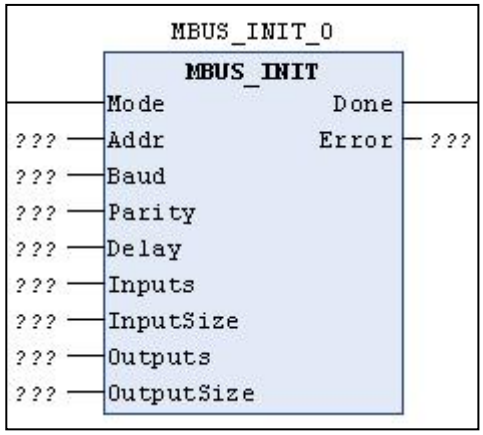


表 E-5　MBUS_INIT 指令的参数说明

参数	说明	数据类型	范围
Mode	设为 1 时初始化 modbus 从站，当程序由 STOP 转换为 RUN，就只需初始化一次。	位	
Addr	从站地址	字节	0 ~ 247
Baud	设置波特率	双字	1200、2400、4800、9600 19200、38400、57600、115200
Parity	设置校验 0：无校验 1：奇校验 2：偶校验	字节	
Delay	附加字符定时器延时	整型	0~32767
Inputs	输入寄存器的起始地址 (ADR(arr1))		
InputSize	输入寄存器最大容量	字	
Outputs	输出保持寄存器的起始地址(ADR(arr1))		
OutputSize	输出保持寄存器的最大容量	字	
Done	始终为 1	位	
Error	错误代码见下表	字节	

表 E-6　错误代码

错误代码	意义
1	存储区范围错误
2	非法波特率或校验
3	非法从站地址
4	非法协议（Modbus 参数）
5	保持寄存器与 Modbus 从站符号地址重叠
6	接收校验错误
7	接收 CRC 错误
8	非法功能请求
9	请求中有非法存储区地址
10	Modbus 未初始化

4）从站 MBUS_SLAVE 使用说明

MBUS_SLAVE 指令用于服务来自 Modbus 主站的请求，必须在每个循环周期都执行，以便检查和响应 Modbus 请求。

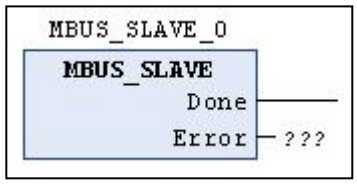


表 E-7 MBUS_SLAVE 指令的参数说明

参数	说明	数据类型	范围
Done	Done 为 1 时执行 Modbus 消息	位	
Error	错误代码见下表	字节	

表 E-8 错误代码

错误代码	意义
1	存储区范围错误
2	非法波特率或校验
3	非法从站地址
4	非法协议 (Modbus 参数)
5	保持寄存器与 Modbus 从站符号地址重叠
6	接收校验错误
7	接收 CRC 错误
8	非法功能请求
9	请求中有非法存储区地址
10	Modbus 未初始化

E.3 Modbus_TCP 库文件

1）Modbus_Tcp 主站 MBUS_TCP_REQ 使用说明

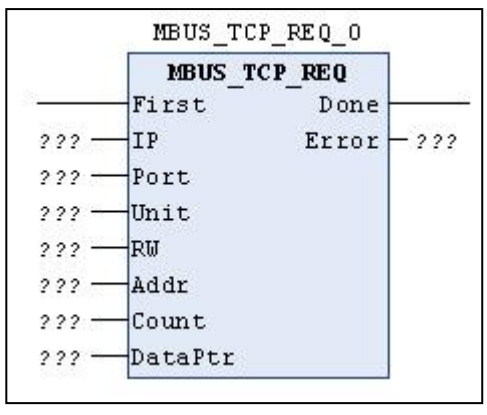


表 E-9 MBUS_TCO_REQ 指令的参数说明

参数	说明	数据类型	范围
First	消息读写请求标志，程序运行周期中设置为 1 表示有新的读写请求	位	
IP	目标 IP 地址，格式为 '192.168.0.125'		

Port	目标端口		
Unit	目标单元号		
RW	读写标志 0：读（Read） 1：写（Write）	字节	
Addr	读写的地址	双字	Addr Modbus 数据地址 (i.e. 40001） 00001 ~ 00xxx 数字量输出 10001 ~ 10xxx 数字量输入 30001 ~ 30xxx 输入寄存器 40001 ~ 40xxx 保持寄存器
Count	读写数据大小 最多可读或写 120 个字的数据	整型	
DataPtr	数据存放位置指针，指向一个字数组，例如（ADR(arr1)）	双字	
Done	始终为 1	位	
Error	错误代码（当 Done 为 1 时，Error 值才有效）	字节	

表 E-10 错误代码

错误代码	意义
0	无错误
1	保留
2	保留
3	接收超时（从站无响应）
4	请求参数出错（IP 地址、Modbus 地址、count、RW，Unit）
5	Modbus 未使能
6	Modbus 正在忙于其它请求
7	响应错误（响应不是请求的操作）
8	响应 CRC 校验和错误
9	硬件错误
101	从站不支持请求的功能
102	从站不支持数据地址（起始地址的长度范围）
103	从站不支持此种数据类型
104	从站设备故障
105	从站接收了信息，但是响应被延迟
106	从站忙，拒绝了该信息
107	从站拒绝了信息
108	从站存储器奇偶错误

2）Modbus_Tcp 从站 MBUS_TCP_SLAVE 使用说明

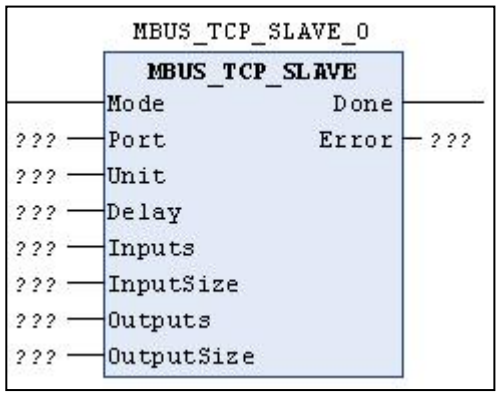


表 E-11　MBUS_TCP_SLAVE 指令的参数说明

参数	说明	数据类型	范围
Mode	设置为 1 时，初始化并启用 TCP MODBUS SLAVE；为 0 时停用 TCP MODBUS SLAVE。注意：若需要启用 TCP MODBUS SLAVE，应保持此值为 1。	位	
Port	本地监听端口 建议设置为 502，如果使用其他端口，建议设置值大于 1024		0~65535
Unit	本地单元号		1~247
Delay	应答延时，建议设置为 0	整型	0~32767
Inputs	输入寄存器起始地址指针，指针类型为字指针，例如 Inputs=ADR(arr1)。Arr1 为字数组。		
InputSize	输入寄存器大小，例如，Inputs=ADR(arr1)时，值应小于等于数组 arr1 的大小		
Outputs	保持寄存器起始地址阵阵，Outputs=ADR(arr3)		
OutputSize	保持寄存器大小，例如，Outputs=ADR(arr3)时，值应小于等于数组 arr3 的大小		
Done	始终为 1	位	
Error	错误码（Done=1 时有效）	字节	

表 E-12　错误代码

错误代码	意义
1	存储区范围错误
2	非法波特率/校验
3	非法的单元 ID
4	非法协议 (Modbus 参数)
5	保持寄存器与 Modbus 从站符号地址重叠
6	保留
7	保留
8	非法功能请求
9	请求中有非法存储区地址
10	Modbus 未初始化

F 在 CODESYS 中进行的相关操作详述

F.1 添加设备

使用 CTH300-C 系列控制器可以在 CODESYS 上进行 CANopen、EtherCAT、Modubs_TCP/RTU 方式的通信；其中 Modbus 通信通过 Modbus 指令库（参考“附录 E CT_MODBUS 库的使用介绍”）进行编程即可实现；而 CANopen、EtherCAT 通信则需要在 CODESYS 设备视图中添加通信设备方可进行指定通信。添加 CANopen、EtherCAT 通信设备的步骤可以参考以下描述。

1、在 CODESYS 中添加 CANopen 通信设备

1）添加 CANbus

在设备视图中右键点击“Device(CodeSys control from Co-trust)”选择“添加设备”，即可在弹出的对话框中选择添加 CANbus：供应商选择“<全部供应商>”、现场总线选择“CANbus”作为 CAN 总线。



2）添加 CAN 主站

在设备视图中选中已经成功添加的“CANBUS”并右键选择“添加设备”，即可在“添加设备”对话框中添加 CAN 主站：现场总线→CANopen→CANopen 管理器→CANopen_Manager。

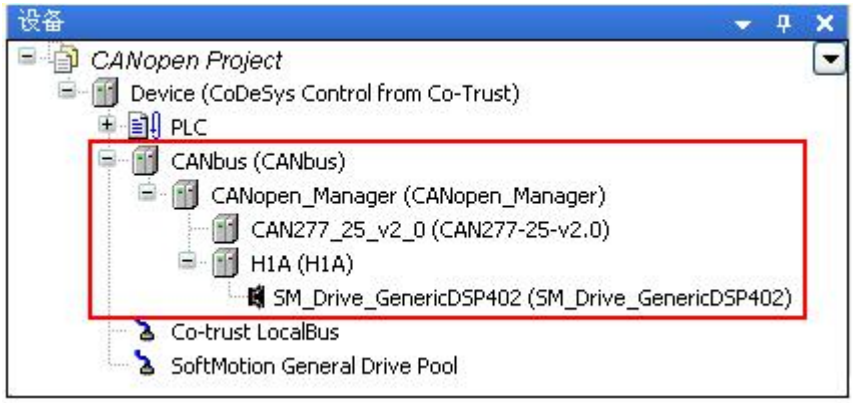
3）添加或扫描 CAN 从站

在设备视图中选中已经成功添加的“CANopen_Manager”并右键选择“添加设备”，即可在“添加设备”对话框中添加 CAN 从站：现场总线→CANopen→远程设备→EM277C/H1A；或者右键

点击“CANopen_Manager”选择“扫描设备”，与 CANopen 主站相连的从站设备即可显示在 CAN 主站下方。

4）添加 CANopen 从站的驱动

在设备视图中选中 CANopen 从站（H1A）并点击右键，在弹出的菜单中选择“Add softmotion-DSP402-axis”，即驱动添加成功。



2、在 CODESYS 中添加 EtherCAT 通信设备

1）添加 EtherCAT 主站

在设备视图选中右键点击“Device(CodeSys control from Co-trust)”选择“添加设备”，即可在弹出的对话框中选择添加 EtherCAT 主站：供应商选择“<全部供应商>”、现场总线选择“EtherCAT”→“主站”→“EtherCAT Master”。

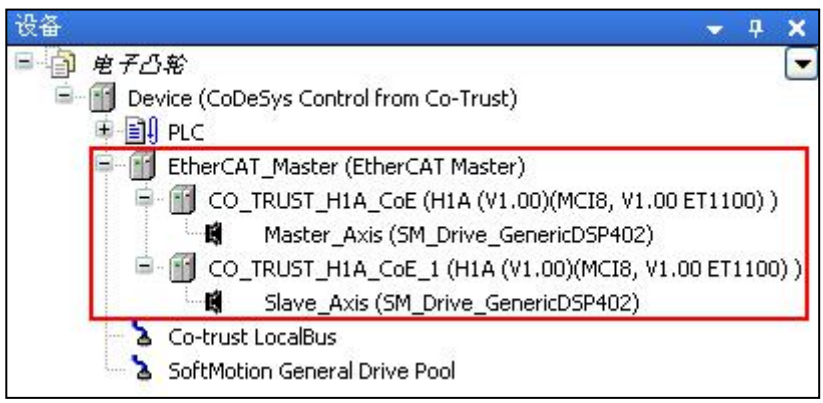
2）添加或扫描 EtherCAT 从站

在设备视图选中已经成功添加的 EtherCAT 主站并点击右键选择“添加设备”，即可在弹出的对话框中选择添加 EtherCAT 从站：供应商选择“<全部供应商>”、现场总线选择“EtherCAT”→“从站”→“H1A”；或者右键点击 EtherCAT 主站（C37-00）选择“扫描设备”，即可将连接到 C37-00 的 EtherCAT 从站（H1A）显示在 EtherCAT 主站（C37-00）下方。

注意：EtherCAT 从站模块无法通过“扫描设备”添加，只能手动选择添加。

3）添加 EtherCAT 从站的驱动

在设备视图中选中 EtherCAT 从站（H1A）并点击右键，在弹出的菜单中选择“Add softmotion-DSP402-axis”，即驱动添加成功。



F.2 配置 CANopen 从站设备

CAN 从站设备（E10/H1A/H2A/EM277C）被成功添加到 CODESYS 中后，需要对其进行相关配置操作。在设备视图中双击打开 CAN 从站，随后即可对其进行配置操作，参考下文：

① “CANopen 远程设备”选项卡：设置节点 ID、勾选“使能专家设置”；



② “PDO 映射”选项卡

在“选择接收 PDO”和“选择发送 PDO”配置框中勾选需要显示在“接收 PDO 映射”和“发送 PDO 映射”选项卡中的参数组。



③ “接收 PDO 映射”选项卡

在该选项卡中可以为选中的参数组添加参数映射，具体操作如下图所示：



<备注> 使用 COTRUST 伺服作为 CANopen 从站时，有关各参数的描述请参考相关伺服使用说明书，如：若使用 H1A 伺服作为 CANopen 从站，则需要参考《H1A 伺服使用说明书》。

④ “发送 PDO 映射”选项卡

与“接收 PDO 映射”操作相同，在该选项卡内选择需要操作的 PDO 参数组，然后点击“添加映射”即可为该组添加发送参数。

⑤ “服务器数据对象”选项卡

服务数据对象配置对话框中可以配置所需的 SDO，它可以定义需要传送对象的顺序，还可以定义在一个不完整的传输过程中将进行的操作。

⑥ “CANopen I/O 映射”选项卡

该对话框用于显示 CANopen I/O 映射，您可在本选项卡中对所映射的数据通道进行读/写操作。



⑦ “状态”选项卡

该对话框用于显示设备的状态信息（如运行、停止等）以及设备的诊断信息。

⑧ “信息”选项卡

该对话框用于显示当前设备的信息，如名称、供应商、类型、版本号、模块序号、描述等。

F.3 配置 EtherCAT 从站设备

EtherCAT 从站设备（E10/H1A/H2A/ECT-00）被成功添加到 CODESYS 中后，需要对其进行相关配置操作。

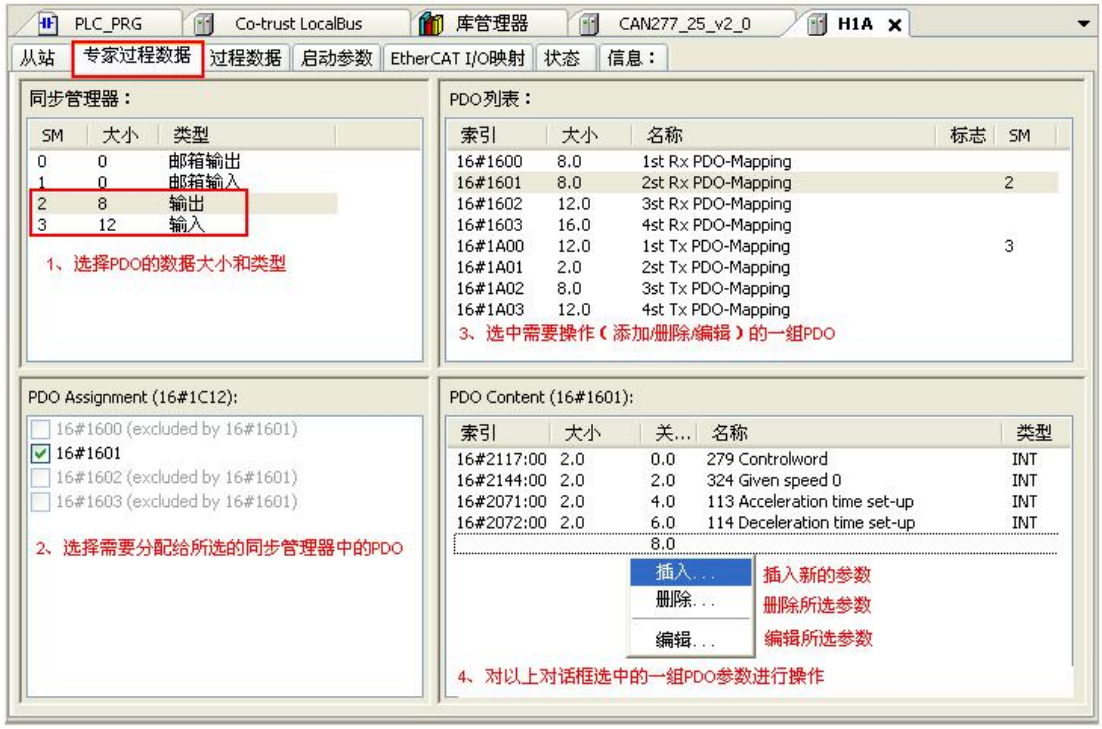
配置 E10/H1A/H2A

① “从站”选项卡：勾选“使能专家设置”，分布式时钟选择“DC for synchronization”。



② “专家过程数据”选项卡

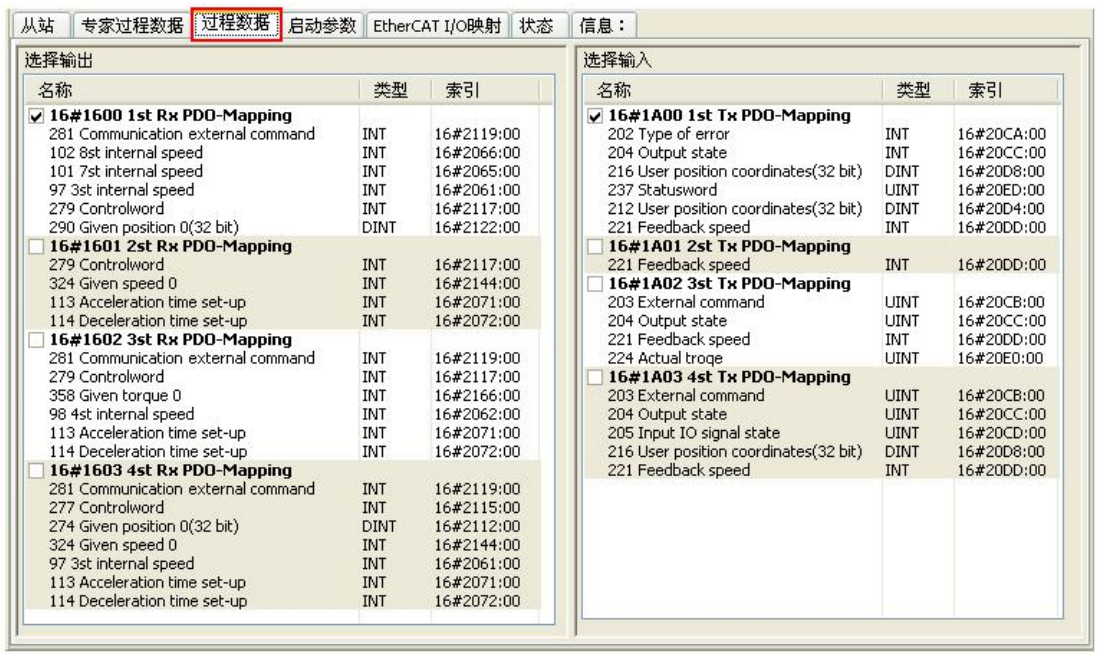
在“从站”选项卡中勾选“使能专家设置”后方可显示。具体操作参考以下图示：



<备注> 有关各参数的描述请参考相关伺服使用说明书，如：使用 H1A 伺服作为 EtherCAT 从站，则需要参考《H1A 伺服使用说明书》。

③ “过程数据”选项卡

显示从站的输入/输出过程数据，由设备描述文件的名称、类型、索引来定义。

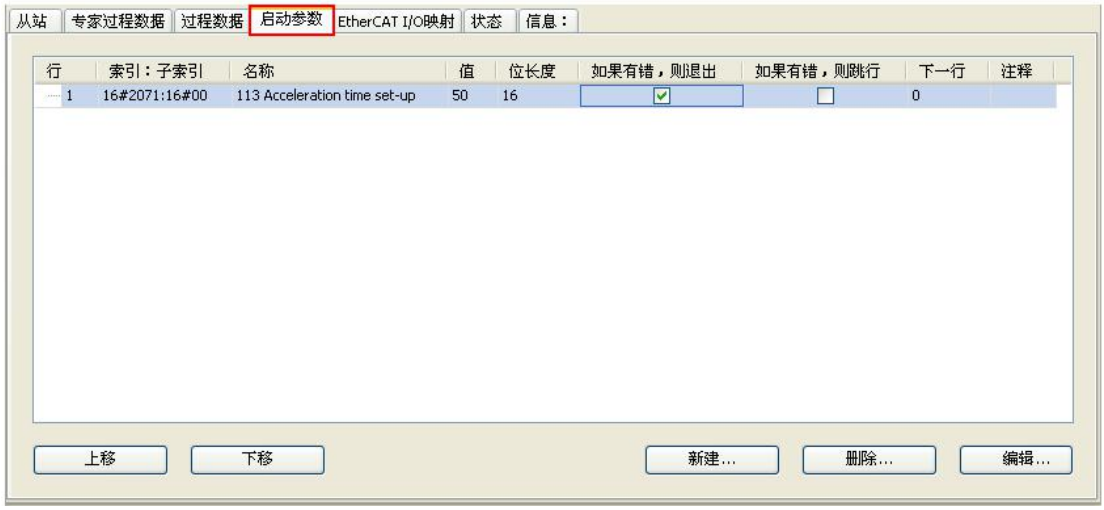


设备所选的输入（读）和输出（写）参数组可以在 I/O 映射对话框中作为 PLC（项目变量可能被映射）的输入和输出。

若要修改当前选择，你必须先用鼠标点击当前选择的参数组前面的方框来取消该选择，随后你即可设置另外一组。

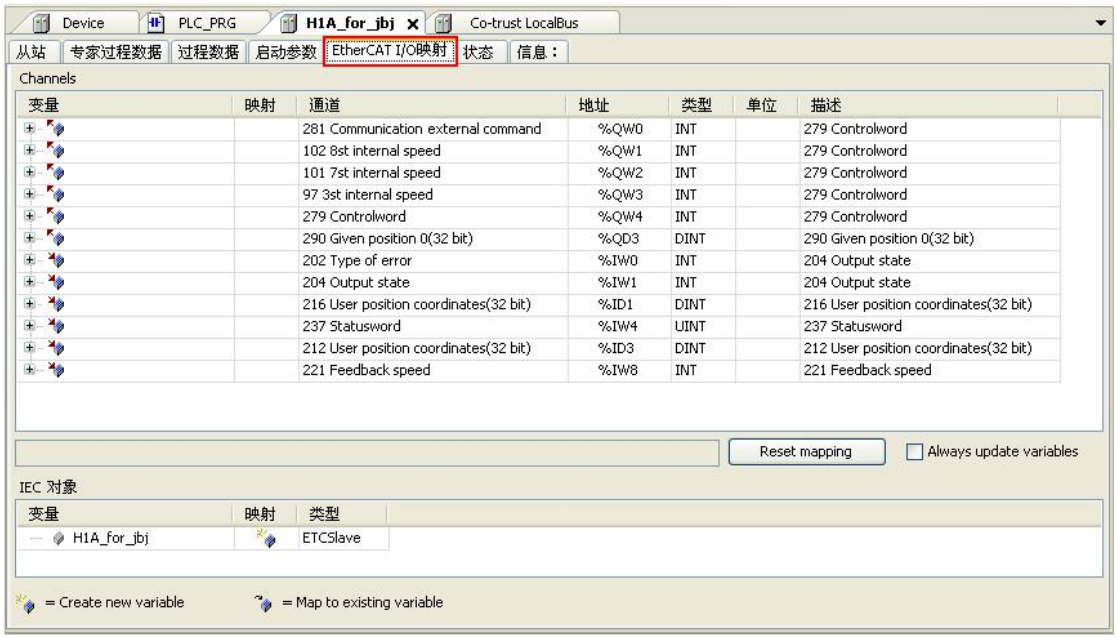
④ “启动参数”选项卡

为设备定义特定的参数，在系统启动时该参数由 SDO 或 IDN 传送。



⑤ “EtherCAT I/O 映射”选项卡

该对话框用于显示 EtherCAT I/O 映射，您可以在本选项卡中对所映射的参数进行读/写操作。



⑥ “状态”选项卡

该对话框用于显示设备的状态信息（如运行、停止等）以及设备的诊断信息。

⑦ “信息”选项卡

该对话框用于显示当前设备的信息，如名称、供应商、类型、版本号、模块序号、描述等。

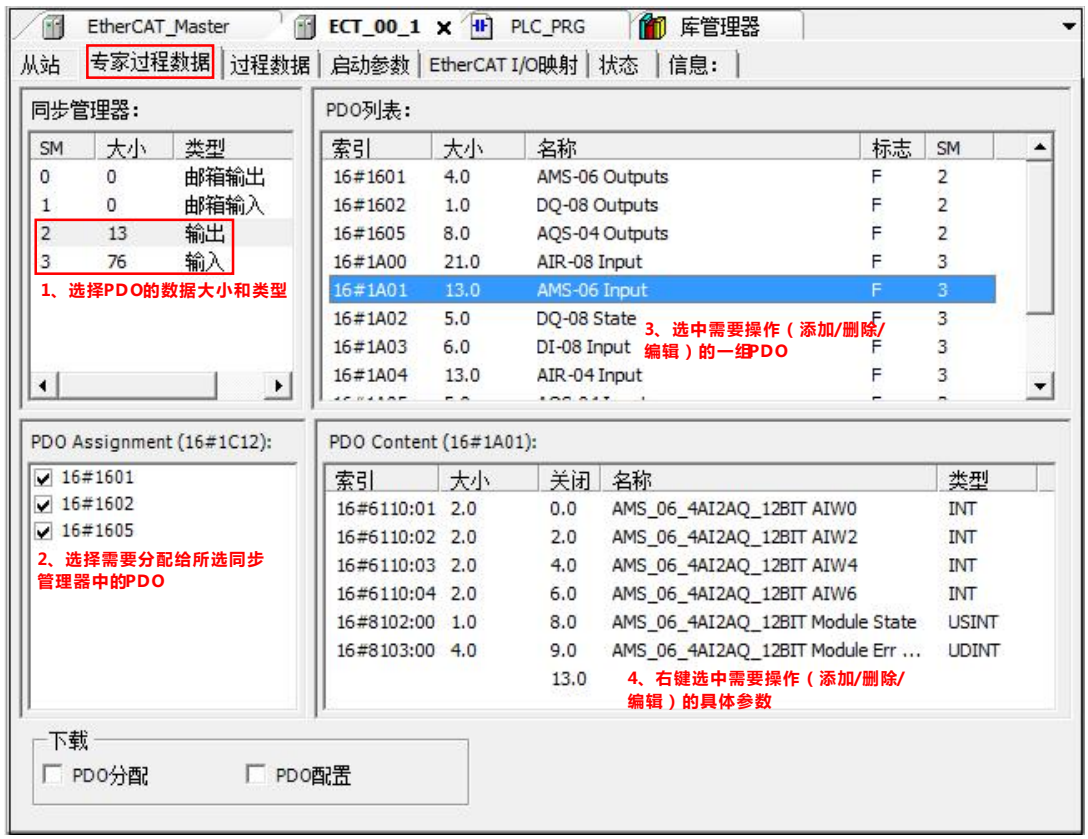
配置 ECT-00

① “从站”选项卡：勾选“使能专家设置”，分布式时钟选择“DC for synchronization”。



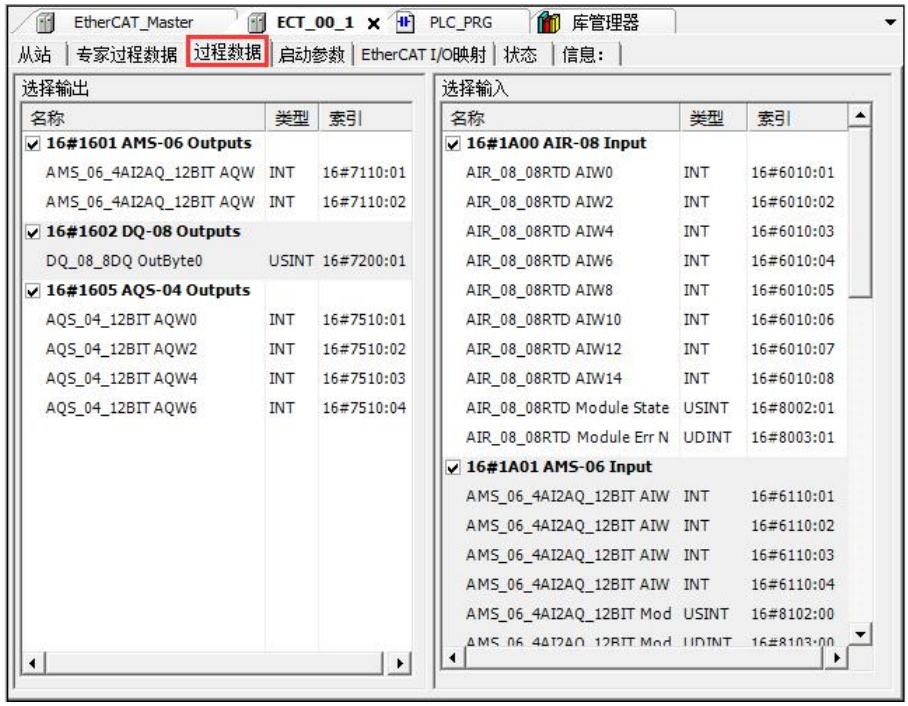
② “专家过程数据”选项卡

在“从站”选项卡中勾选“使能专家设置”后方可显示。具体操作参考以下图示：



③ “过程数据”选项卡

显示从站的输入/输出过程数据，由设备描述文件的名称、类型、索引来定义。

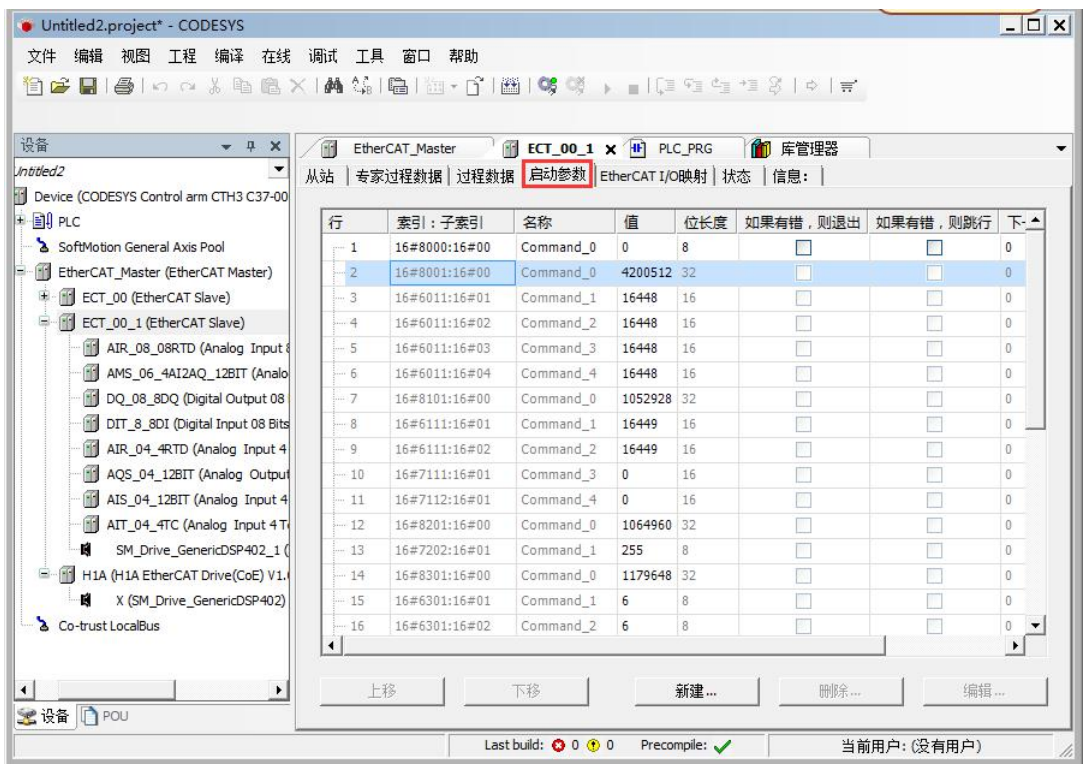


设备所选的输入（读）和输出（写）参数组可以在 I/O 映射对话框中作为 PLC（项目变量可能被映射）的输入和输出。

注意：EtherCAT 从站的过程数据默认全选，请勿更改，否则可能会导致通信故障。

④ “启动参数”选项卡

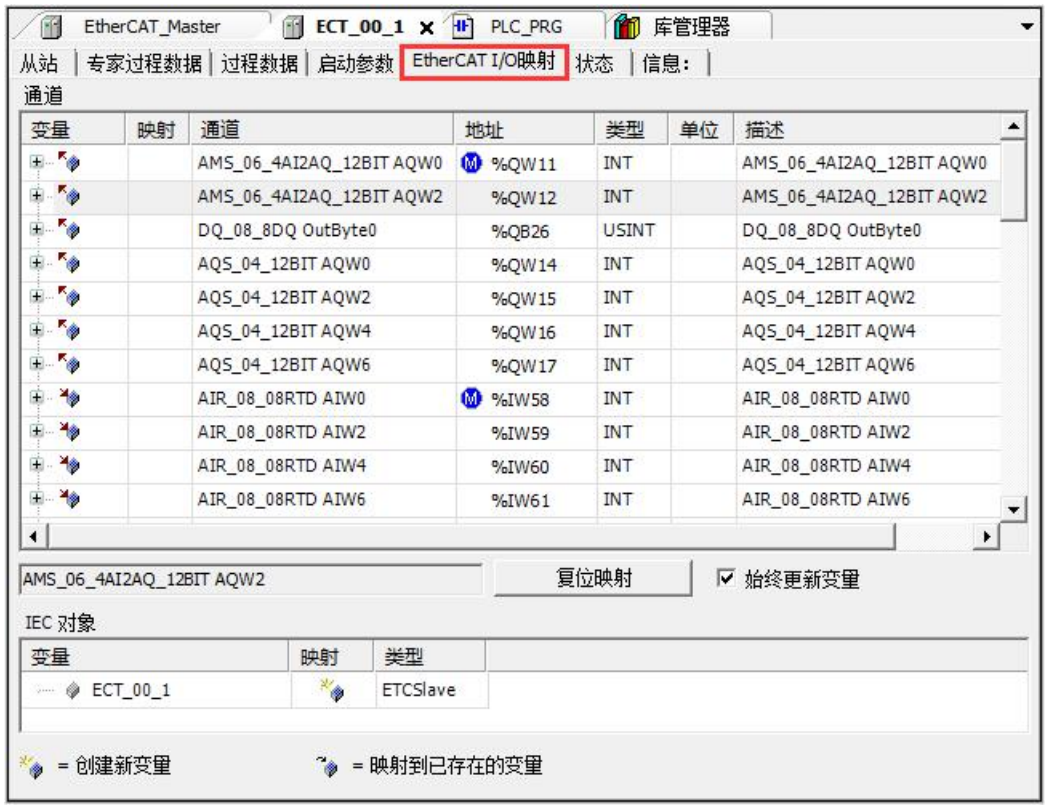
为设备定义特定的参数，在系统启动时该参数由 SDO 或 IDN 传送。



注意：EtherCAT 从站的启动参数无法修改，如需修改扩展模块控制字的配置，请在菜单树中双击打开相应扩展模块，在其中修改启动参数后将配置下载到 CPU 运行，相关配置更改随即生效。

⑤ “EtherCAT I/O 映射”选项卡

该对话框用于显示 EtherCAT I/O 映射，您可以在本选项卡中对所映射的参数进行读/写操作。

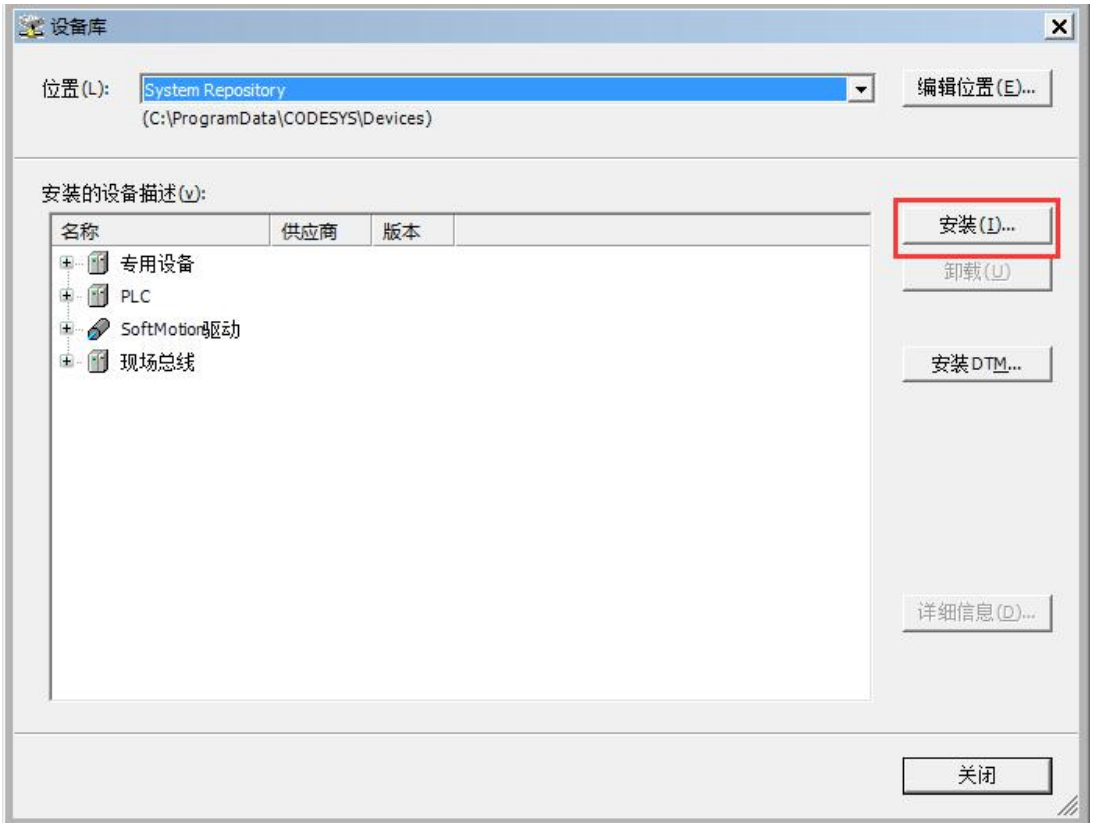


注意：各个扩展模块分配的地址并非固定不变，如果执行添加/删除模块操作，则相应分配的地址会发生变化。

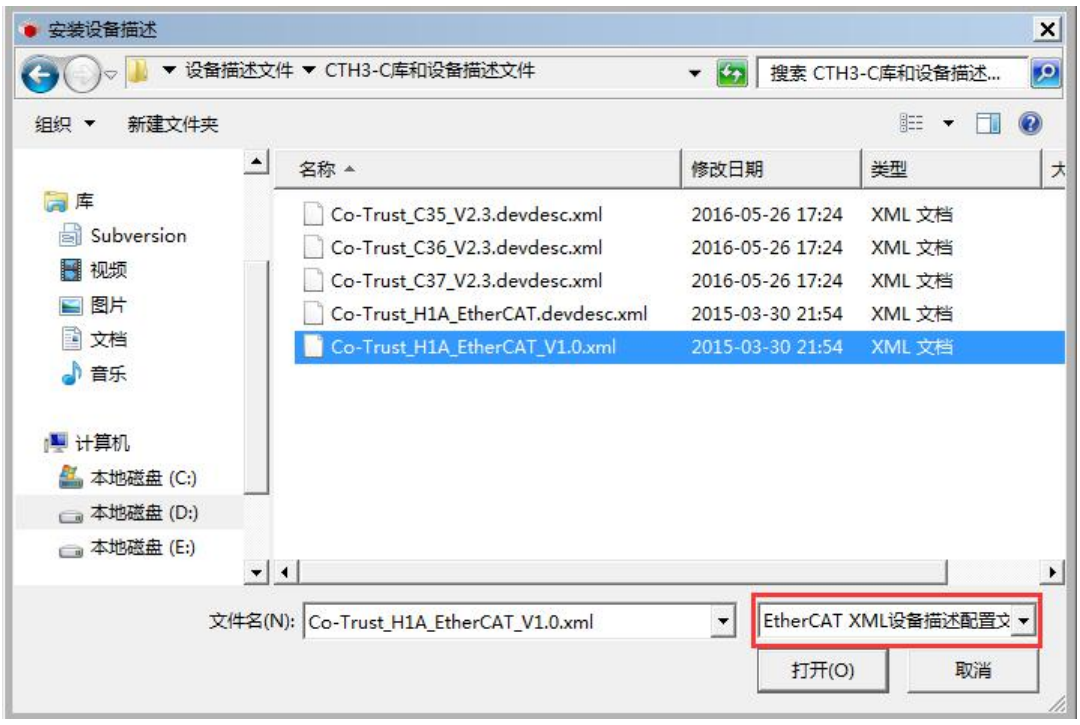
G 在 CODESYS 中安装设备描述文件

G.1 安装 EtherCAT 伺服从站设备描述文件

1、打开 CODESYS 软件，选择菜单项“工具”→“设备库”打开如下对话框，在该对话框里中点击“安装”。

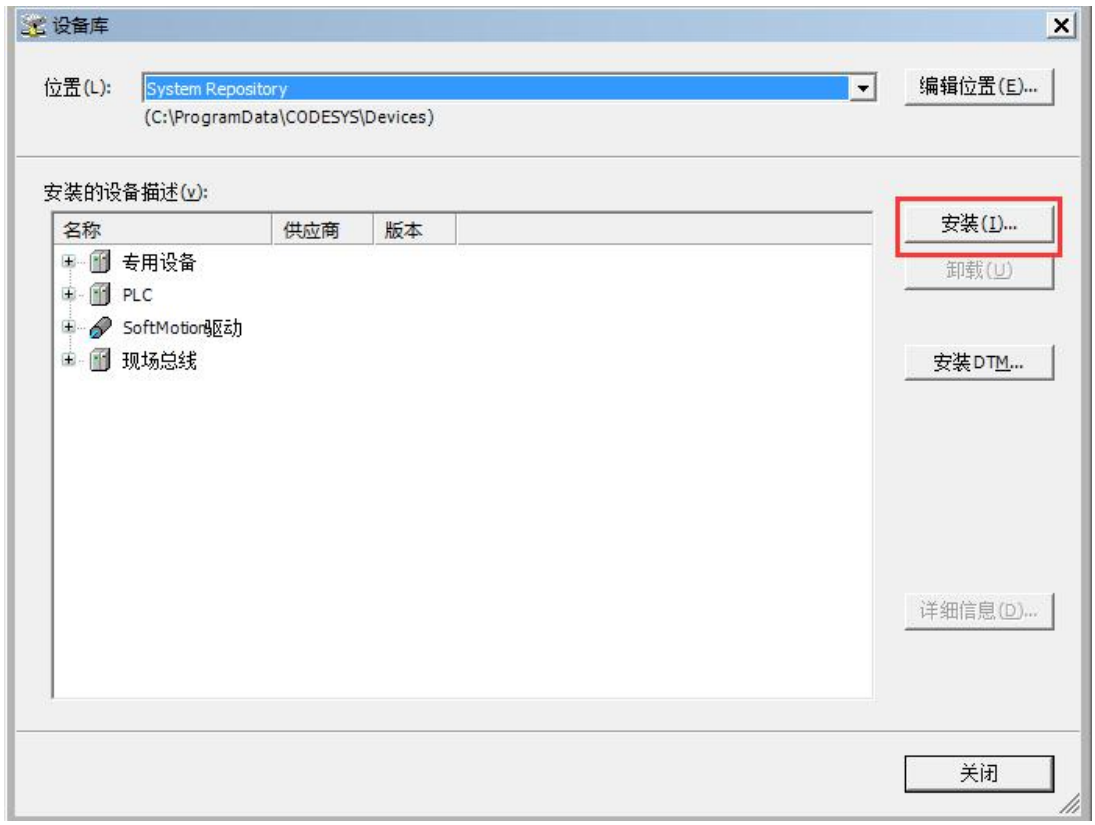


2、执行上述操作后随即弹出一个对话框如下，请选择文件类型“EtherCAT XML 设备描述配置文件（*.xml）”，选择文件“Co-Trust_H1A_EtherCAT_V1.0.xml”，点击“打开”即可安装成功。

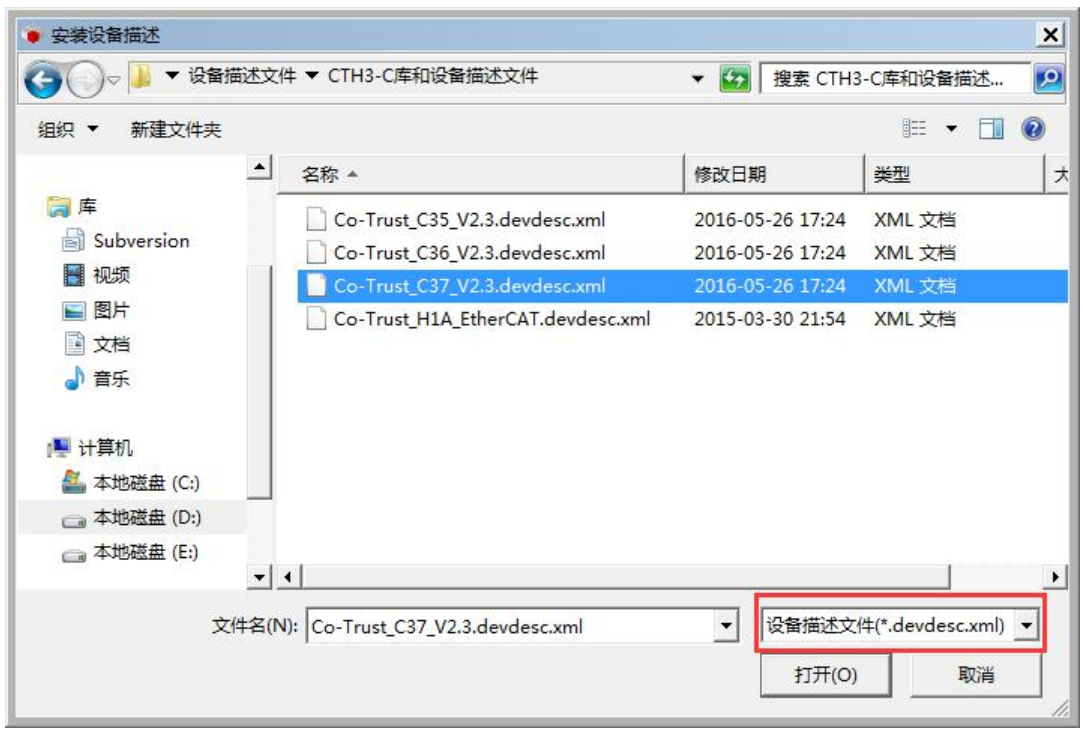


G.2 安装 PLC/402 轴设备描述文件

1、打开 CODESYS 软件，选择菜单项“工具”→“设备库”打开如下对话框，在该对话框里中点击“安装”。



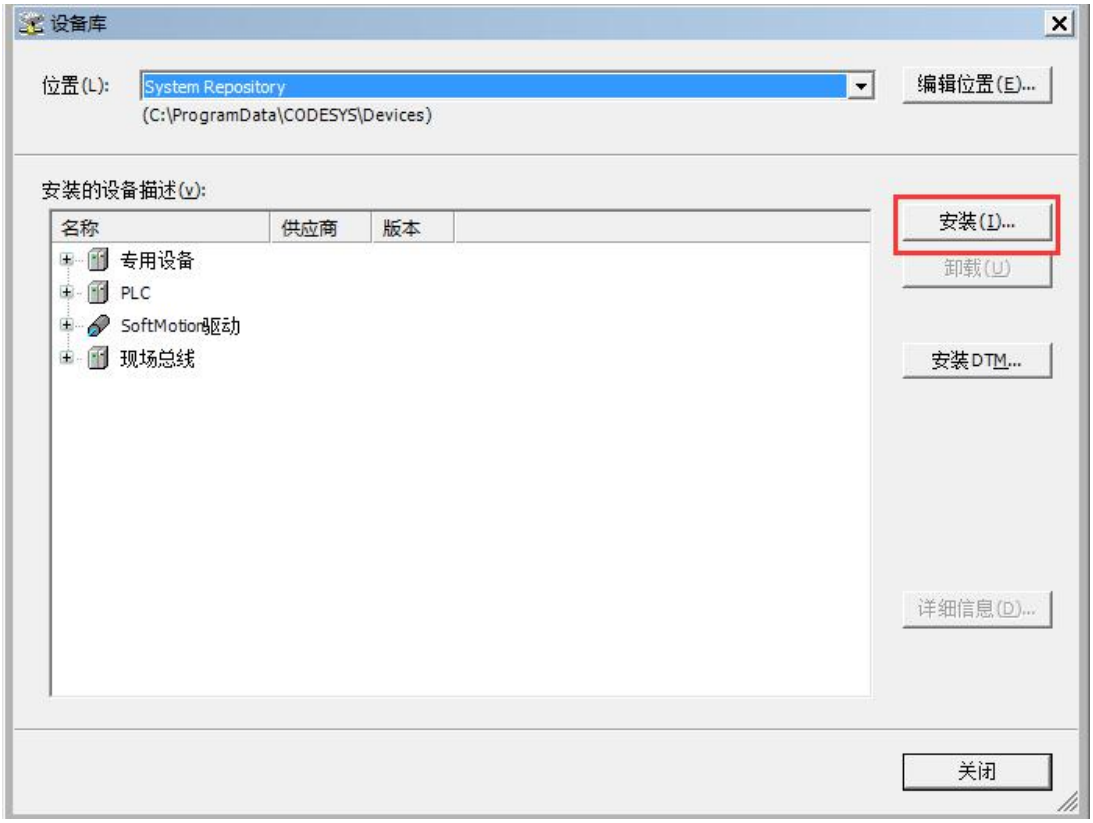
2、执行上述操作后随即弹出一个对话框如下，请选择文件类型“设备描述配置文件（*.devdesc.xml）”，选择文件“Co-Trust_C35_V2.3.devdesc.xml”或“Co-Trust_C36_V2.3.devdesc.xml”或“Co-Trust_C37_V2.3.devdesc.xml”或“Co-Trust_H1A_EtherCAT.devdesc.xml”，点击“打开”即可安装成功。



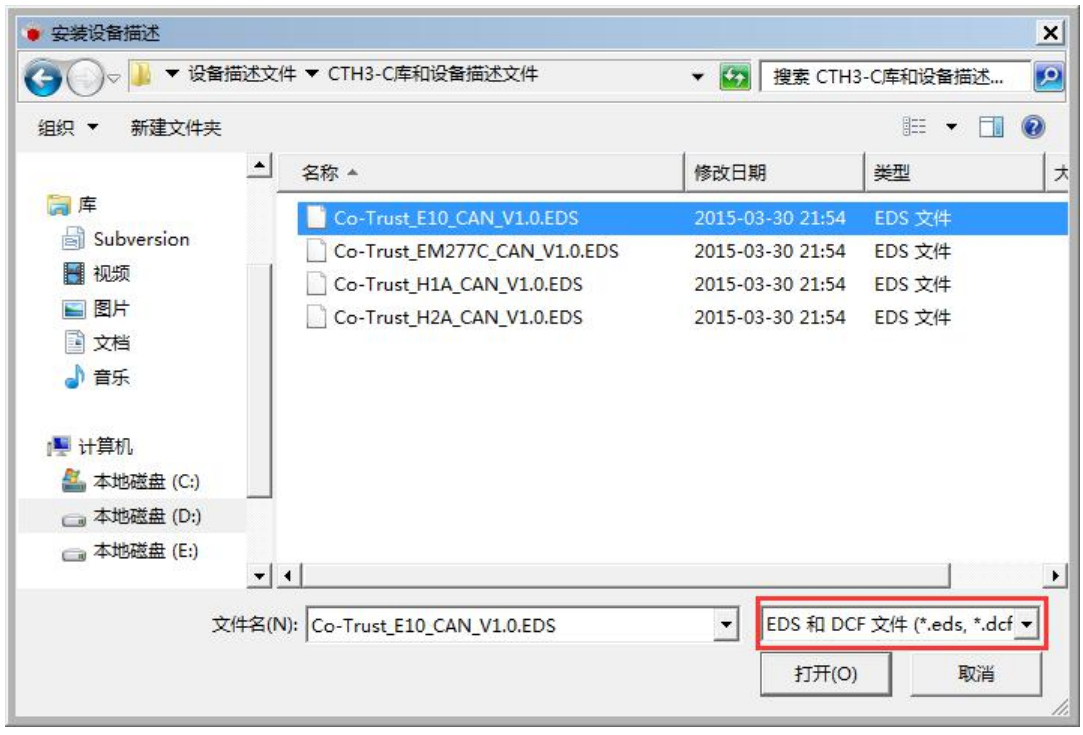
注意：如果使用的不是合信 H1A EtherCAT 伺服，请不用安装 402 轴文件，即“Co-Trust_H1A_EtherCAT.devdesc.xml”不用安装。

G.3 安装 CANopen 从站 EDS 文件

1、打开 CODESYS 软件，选择菜单项“工具”→“设备库”打开如下对话框，在该对话框里中点击“安装”。



2、执行上述操作后随即弹出一个对话框如下，请选择文件类型“EDS 和 DCF 文件（*.eds.*.dcf）”，选择“Co-Trust_H1A_CAN_V1.0.EDS”等后缀为.EDS 的文件，点击“打开”即可安装成功。



H 指令速查

H.1 CODESYS 中的操作符总结

下表为操作符总结，这些操作符分别都存在于 CODESYS 的 Standard.lib 和 Util.lib 程序库中。

注意‘IL 符’栏：只有使用此符号的行才被显示。一个必要条件是：第一组必要的操作数已被顺利载入前一行（如，LD in）

‘Mod.IL’栏显示 IL 中有可能出现的修饰符：

C	前行表达式为 TRUE 的结果的修饰符。
N	对于 JMPC，CALC，RETC：如前行的结果为 FALSE，则执行此命令。
N	否则：否定操作数（非累积器）
(	括弧中操作符：只有括号出现后操作才开始执行。

表 H-1 操作符总结表

在 ST 中	在 AWL 中	Mod. AWL	描述
'			字符串分隔符
.. **[ ]**			Array 值域大小
:			声明中操作数与类型间的分隔符。
;			说明终止。
^			指示符废止。
	LD var1	N	缓冲器中变量 1 值的载入。
:=	**ST var1**	N	将实际结果存入变量 1。
	S boolvar		当实际结果为 TRUE，准确设置操作数布尔变量为 TRUE。
	R boolvar		当实际结果为 FALSE，准确设置操作数布尔变量为 TRUE。
	JMP label	**CN**	直接跳至标签。
<Program name>	**CAL** prog1	**CN**	调用程序 prog1。
<Instance name>	**CAL** inst1	**CN**	调用功能程序 inst1。
<Fctname>(vx, vy,..)	**<Fctname> vx, vy**	**CN**	调用功能 fctname 和 transmit 变量 vx，vy。
RETURN	**RET**	CN	Leave POU and go back to caller
	(		括号后的值作操作数处理，括号前的运算在括号内的表达式之前不执行。
	)		现执行已被设置运算
AND	**AND**	N,(	位 AND
OR	**OR**	N,(	位 OR
XOR	**XOR**	N,(	位专用 OR
NOT	**NOT**		位 NOT
+	**ADD**	(	加法
-	**SUB**	(	减法
*****	**MUL**	(	乘法
/	**DIV**	(	除法
>	**GT**	(	大于
>=	**GE**	(	大于等于

=	**EQ**	(	等于
<>	**NE**	(	不等于
<=	**LE**	(	小于等于
<	**LT**	(	小于
MOD(in)	**MOD**		模相除
INDEXOF(in)	**INDEXOF**		POU in1 的内部指数
SIZEOF(in)	**SIZEOF**		已知 in 数据类型所需字节数
SHL(K,in)	**SHL**		操作符 in 位左移 k
SHR(K,in)	**SHR**		操作符 in 位右移 k
ROL(K,in)	**ROL**		操作符 in 位旋转 k 到左
ROR(K,in)	**ROR**		操作符 in 位旋转 k 到右
SEL(G,in0,in1)	**SEL**		in0（G 为 FALSE）和 in1（G 为 TRUE）两个操作数之间的两进制选择。
MAX(in0,in1)	**MAX**		2 个值中较大的一个的复原。
MIN(in0,in1)	**MIN**		in0 和 in1 两值之中较小一个的复原。
LIMIT(MIN,in,Max)	**LIMIT**		限制值域（in 重置为 MIN，或 MAX，以免超出范围）。
MUX(K,in0,...in_n)	**MUX**		一组值中（in 至 In_n）选择第 k 个值。
ADR(in)	**ADR**		DWORD 中操作数的地址。
ADRINST()	**ADRINST**()		调用操作符即功能块操作符的地址。
BITADR(in)	**BITADR**		DWORD 中操作数的位平衡。
BOOL_TO_<type>(in)	**BOOL_TO_<type>**		布尔操作数的类型转换。
<type>_TO_BOOL(in)	**<type>_TO_BOOL**		类型转换成 BOOL。
INT_TO_<type>(in)	**INT_TO_<type>**		INT 操作数类型转换位另一基本类型。
REAL_TO_<type>(in)	**REAL_TO_<type>**		REAL 操作数类型转换位另一基本类型。
LREAL_TO_<type>(in)	**LREAL_TO_<type>**		LREAL 操作数类型转换位另一基本类型。
TIME_TO_<type>(in)	**TIME_TO_<type>**		TIME 操作数类型转换位另一基本类型。
TOD_TO_<type>(in)	**TOD_TO__<type>**		TOD 操作数类型转换位另一基本类型。
DATE_TO_<type>(in)	**DATE_TO_<type>**		DATE 操作数类型转换位另一基本类型。
DT_TO_<type>(in)	**DT_TO_<type>**		DT 操作数类型转换位另一基本类型。
STRING_TO_<type>(in)	**STRING_TO_<type>**		字符串操作数类型转换位另一基本类型，in 必须含所求类型的有效值。
TRUNC(in)	**TRUNC**		从 REAL 转换为 INT。
ABS(in)	**ABS**		操作数 in 的绝对值。
SQRT(in)	**SQRT**		操作数 in 的平方根。
LN(in)	**LN**		操作数 in 的自然对数。
LOG(in)	**LOG**		操作数 in 的以 10 为底的对数。
EXP(in)	**EXP**		操作数 in 的幂方。
SIN(in)	**SIN**		操作数 in 的正弦。

COS(in)	**COS**		操作数 in 的余弦。
TAN(in)	**TAN**		操作数 in 的正切。
ASIN(in)	**ASIN**		操作数 in 的反正弦。
ACOS(in)	**ACOS**		操作数 in 的反余弦。
ATAN(in)	**ATAN**		操作数 in 的反正切。
EXPT(in,expt)	**EXPT** expt		操作数 in with expt 的幂乘方。

<备注> 请于 CODESYS 程序库的相关附录里获取详细的使用信息。

H.2 Standard.lib 库

表 H-2 Standard.lib 库指令

在 ST 中	在 AWL 中	描述
LEN(in)	**LEN**	操作数 in 的字符串长度
LEFT(str,size)	**LEFT**	已知字符串 str 大小的左边第一个字符串。
RIGHT(str,size)	**RIGHT**	已知字符串 str 大小的右边第一个字符串。
MID(str,size,pos)	**MID**	POS 处已知大小的部分 str 字符串。
CONCAT('str1','str2')	**CONCAT** 'str2'	两个后继字符串的合并。
INSERT('str1','str2',pos)	**INSERT** 'str2',p	在 pos 处把字符串 str1 插入字符串 str2
DELETE('str1',len,pos)	**DELETE** len,pos	删除部分字符串（长度 len），启动位置为 str1 的 pos 处。
REPLACE('str1','str2',len ,pos)	**REPLACE** 'str2',len,pos	替换长度字符串为 str2，起始位置为 str1 的 pos 处。
FIND('str1','str2')	**FIND** 'str2'	在 str1 中寻找部分字符串 str2。
SR	**SR**	双稳态的 FB 设置为显性。
RS	**RS**	双稳态的 FB 重设。
SEMA	**SEMA**	FB：臂板软件（可中断）
R_TRIG	**R_TRIG**	FB：上升边界被监测。
F_TRIG	**F_TRIG**	FB：下降边界被监测。
CTU	**CTU**	FB：升值计数。
CTD	**CTD**	FB：下降计数。
CTUD	**CTUD**	FB：上下计数。
TP	**TP**	FB：触发
TON	**TON**	FB：On-Delay 时钟。
TOF	**TOF**	FB：Off-Delay 时钟。
RTC	**RTC**	FB：现实时钟。

H.3 Util.lib 库

表 H-3 Util.lib 库指令

BCD_TO_INT	字节转换：BCD 转为 INT 格式。
INT_TO_BCD	字节转换：INT 转为 BCD 格式。
EXTRACT(in,n)	DWORD in 的第 n 位以 BOOL 类型还原。
PACK	多达 8 位被压入一个字节。
PUTBIT	DWORD in 一个位被设位一定的值。
UNPACK	字节被作为单个的位被复原。

DERIVATIVE	局部派生。
INTEGRAL	整数。
LIN_TRAFO	REAL 值的转换。
STATISTICS_INT	INT 格式的最小，最大和平均值。
STATISTICS_REAL	REAL 格式的最小，最大和平均值。
VARIANCE	变异。
PD	PD 控制器。
PID	PID 控制器。
BLINK	脉冲信号。
FREQ_MEASURE	布尔输入信号的测量频率。
GEN	周期性功能。
CHARCURVE	线性功能
RAMP_INT	限制供应功能（INT）减弱的上升。
RAMP_REAL	限制供应功能（REAL）减弱的上升。
HYSTERESIS	滞后作用。
LIMITALARM	注意输入值是否超出定义范围的限制。

H.4 SoftMotion 库

SM3_Basic Library

该库是 CODESYS SoftMotion 应用程序的基本库，因此它必须在 SoftMotion 工程中被包括。这将在一个 SoftMotion 设备插入时自动进行。

该库提供了以下功能块和函数：

● PLCopen 功能块既允许实现单轴运动控制，又允许实现两轴同步运动。除了状态检查、参数化和一般操作的库元素以外，还有以定义的速度、加速度参数来驱动轴的功能块。此外还有同步主、从轴和传动轴的模块。

● DriveInterface 基本驱动程序功能块（AXIS_REF_SM3, AXIS_REF_VIRTUAL_SM3, AXIS_REF_MAPPING_SM3）

● 工具函数，比如文件服务和错误报告

● 特定驱动器驱动程序的功能块或实现，基本驱动功能块的扩展。这些特殊驱动功能块（名为 AXIS_REF_*）通常被包括在一个单独的库中，该库被这个设备的相应设备描述所参考。

功能块整理如下：

● 驱动接口

指令名称	功能
AXIS_REF	这个驱动接口是一个功能块。它由“SM3_Basic.library”库提供，并包括作为驱动器的工作。每一个 SoftMotion 轴都是 AXIS_REF_SM3 实例的一个扩展。

● 数据类型，全局变量

指令名称	功能
MC_DIRECTION	它用作几个功能块的输入并指定运动方向。请注意，不是所有的模式都能够应用于所有的功能块和轴类型（模数/有限）。
MC_TAPPETMODE	它用于功能块 MC_DigitalCAMSwitch 以确定 tappet 值应参考

	主轴设定点（1），还是实际位置（2）。在自动检测（0）情况下，功能块的判断依赖于：驱动的控制状态是使用了设定值（bRegulatorOn = TRUE）还是实际值（bRegulatorOn = FALSE）。
SMC_BRAKESETSTATE	它用于功能块 SMC3_BrakeControl，决定驱动器如何设置机械制动。
SMC_CAMTAPPETACTION	它确定了当凸轮已被经过时，哪些开关动作被执行。
SMC_CAMTAPPETTYPE	它用于确定凸轮必须经过哪个方向才能被激活。
SMC_CONTROLLER_MODE	它描述了电机的哪个量被控制。
SMC_ERROR	它包括了 SoftMotion 功能块可能返回的所有错误号。
SMC_HOMING_MODE	在 SMC_Homing 使用时它定义了归航序列。
SMC_INT_STATUS	它描述插值器实例的状态。
SMC_LANGUAGE_TYPE	它用于函数 SMC_ErrorString 以确定语言。

- 数据类型，结构

指令名称	功能
MC_CAM_ID	它是一个描述 CAM 工作台的内部数据结构。它由 MC_CAMTableSelect 生成并做为 MC_CamIn 的输入。
MC_CAM_REF	它代表一个普通 CAM 并包含如下元素：
MC_CAMSWITCH_REF	它描述了 CAMSwitches 表并用作 MC_DigitalCAMSwitch 的输入。
MC_CAMSWITCH_TR	它描述了随行件的切换位置。
MC_OUTPUT_REF	它定义了 32 个布尔数组的形式的挺杆输出。 TYPE MC_OUTPUT_REF : ARRAY [0..31] OF BOOL; END_TYPE
MC_TRACK_REF	

- DriveBasic

指令名称	功能
SMC3_BrakeControl	如果被驱动装置和它的 SoftMotion 驱动程序支持，该功能块确定了机械刹车制动器的行为。
SMC3_BrakeStatus	它读取机械制动器的状态。
SMC_SetControllerMode	如果驱动设备支持，该模块可用来切换到其他控制器模式。
SMC_GetMaxSetAccDec	它用来测量一个轴的加速度（或减加速度）绝对值的最大值。
SMC_GetMaxSetVelocity	它用来测量一个轴速度最大值。
SMC_GetTrackingError	它测量用来补偿死区时间的实际和最大迟滞误差，这种误差可能在现场总线通讯时发生并将持续很多个周期 (byDeadTimeCycles)。
SMC_InPosition	它检查轴的实际位置是否在一段确定的时间内在某个位置值以内。
SMC_MeasureDistance	它用于一个旋转轴，以测量走过的距离（外壳也被计算在内）。
SMC_CheckLimits	它可用来检查驱动装置的实际设置点是否超过了在控制器中配置的最大值。检查的结果将被 bLimitsExceeded 输出指示。
SMC_FollowPosition	它将不作任何检查地把位置设置点写入轴。
SMC_FollowPositionVelocity	除了可能额外定义的速度以外，它的使用和

	SMC_FollowPosition 等效。
SMC_FollowSetValues	它将不作任何检查地把设置值写入轴。
SMC_FollowVelocity	它将不作任何检查地将设置速度写入轴。
SMC_ClearFBEror	它用来删除最老的功能块错误信息。
SMC_ReadFBError	它用来读取功能块错误的最老信息。
SMC_Homing	它执行轴的参考运动。做为参考开关，一个布尔值被使用，典型地该值是由一个硬件输入的。
SMC_ChangeGearingRatio	在该模块的帮助下，传动比和驱动设备类型可以被修改。

● Error

指令名称	功能
SMC_ErrorString	依赖于输入 ErrorID 和语言，函数 SMC_ErrorString 将返回一个字符串表达的错误。

● File

指令名称	功能
SMC_ReadCam	它用来在运行时下载 CAM 并使其对 MC_CamTableSelect 和 MC_CamIn 有效。
SMC_WriteCam	该功能块在运行时向*.CAM-file 文件写入一个在 CAM 编辑器中创建的 CAM。
SMC_AxisDiagnosticLog	它用来将属于一个轴的一系列参数值循环写入到文件中。这个输出文件是故障诊断的依据。

● PLCopen

指令名称	功能
SMC_ReadSetPosition	它用来读取驱动设备的当前设置位置。
SMC_SetTorque	如果驱动设备处于控制器模式“torque”时，它可用来创建一个力矩。
SMC_CAMBounds	它计算从设备的最大位置、速度和加速度值，该设备以绝对模式与一个主设备相连，主设备按规定的最大速度和加速度/减加速度移动。
SMC_CAMBounds_Pos	与 SMC_CAMBounds 相对的，该功能块仅计算从设备的位置最大、最小值，该设备以绝对模式与一个主设备相连，主设备按规定的最大速度和加速度/减加速度移动。
SMC_CAMRegister	它代表一个挺杆控制单元，它和 MC_CamIn 一样，作用于 MC_CAM_REF 结构，撤销原轨迹信息而只读挺杆信息。
SMC_GetCamSlaveSetPosition	如果（从）轴通过 CAM 与其他（主）轴的运动相连接，该模块计算从轴的当前目标位置。因此两个轴都不会被移动或影响。
SMC_GetTappetValue	它评估 MC_CamIn 功能块的输出 Tappet 并包含当前 tappet 状态。
MC_CamIn	它占用 CAM。
MC_CamOut	它将立即从主轴中释放从轴。
MC_CamTableSelect	它通过设置与相关工作台的连接来选择 CAM 工作台。
MC_GearIn	它通过设置与相关工作台的连接来选择 CAM 工作台。
MC_GearInPos	它通过设置与相关工作台的连接来选择 CAM 工作台。

MC_GearOut	它通过设置与相关工作台的连接来选择 CAM 工作台。它使从轴脱离主轴。
MC_Phasing	它将占用 CAM。
MC_AccelerationProfile	它用来控制一个时间-加速度已锁定的运动分布。
MC_Halt	它用来命令一个受控运动停止。功能块将终止所有进行中的功能块的执行。
MC_Home	它的执行引发轴执行“search home”序列。该序列的详情依赖于加工工艺并可以由轴的参数设置。
MC_MoveAbsolute	它用来命令受控运动到一个指定的绝对位置。
MC_MoveAdditive	它用来在离散运动状态下命令一个受控运动，该运动把指定相对距离与新近要求的位置相加。
MC_MoveRelative	它用来命令一个受控运动，在执行时该运动指定了相对于轴的实际位置的距离。
MC_MoveSuperImposed	它用来命令一个受控运动，该运动将指定的相对距离加到已有的运动中。已有的运动不会被中断，而是叠加附加运动。
MC_MoveVelocity	它用来以特定速度命令一个永不终止的受控运动。
MC_PositionProfile	它用来命令一个时间－位置锁定的运动分布。
MC_Power	它用来控制 power stage（“开”或“关”）。
MC_ReadActualPosition	它将返回参考轴的实际位置。
MC_ReadAxisError	它用来描述与功能块无关的普通轴错误。
MC_ReadBoolParameter	它返回厂家指定的 BOOL 类型的参数。
MC_ReadStatus	关于当前进程中的运动，它返回轴的详细状态。
MC_ReadParameter	它返回厂商指定的参数。
MC_Reset	它通过重置所有内部关于轴的错误，把 ErrorStop 转换成 StandStill 状态－这会影响功能块实例的输出。
MC_Stop	它的执行将导致受控运动停止并将轴设置为状态“Stopping”。
MC_VelocityProfile	它用来命令一个时间－速度锁定的运动分布。
MC_WriteBoolParameter	它可用来修改厂家设定的布尔类型的参数。
MC_WriteParameter	它可用来修改厂家设定的参数。
MC_AbortTrigger	它是用来终止与触发事件（例如 MC_TouchProbe）相连接的功能块。
MC_DigitalCamSwitch	和电动机轴上的开关类似：功能块控制一组离散输出位元，来开关一组就像是由机械凸轮控制的、与轴相连的开关。向前和向后运动都是允许的。
MC_ReadActualTorque	当 Enable 保持为 TRUE 时，功能块将返回实际转矩或力的值。
MC_ReadActualVelocity	当 Enable 保持为 TRUE 时，功能块将返回实际速度的值。
MC_SetPosition	它用来移动一个轴的坐标系。
MC_TouchProbe	它用来在触发事件发生时记录轴的位置。
SMC_MoveContinuousAbsolute	它执行绝对运动，但是与 MC_MoveAbsolute 不同，它不是以速度＝0 到达目标位置的，而是以指定速度。
SMC_MoveContinuousRelative	它执行相对运动，但是与 MC_MoveRelative 不同，它不是以速度＝0 到达目标位置的，而是以指定速度。

● SimpleTest

指令名称	功能

SMC_StartupDrive	它包含一组经常使用的功能块，它用来测试和试运转一个轴。

SM3_CNC Library

SM3_CNC.Library 库里的模块用来（依照 DIN 66025）解码在 CNC 编辑器中创建的轨迹。解码是为了转化轨迹到一个结构中，以使其能够被 IEC 程序所用（轨迹预处理），或能够被修改、被插补以及被转换以使其对于驱动设备是可读的（例如，SMC_NCDecoder，ToolCorr，SMC_AvoidLoop，SMC_SmoothPath，SMC_RoundPath，SMC_Interpolator）。

模块的功能故障允许某些组件－比如解码器－被为特殊需要而设计的模块所代替。此外，其他模块－比如某个轨迹预处理或者优化－可以被毫无问题地加入到已有组件中，这样不需要考虑库中组件间的适配性。因此，程序逻辑完全由 PLC 程序处理，只有纯动作信息才由库模块处理。

库 SM3_CNC.Library 的数据结构（例如 SMC_POSINFO，SMC_VECTOR3D）用来描述位置、轨迹段和向量，并管理 GEOINFO 对象（QUTQUEUE 结构）。

- 数据类型

指令名称	功能
SMC_CNC_REF	这个数据结构管理被解析了的 G 代码文件。
SMC_GCODE_TEXT	数据源与 SMC_NCDecoder 模块的同名输出相连。它用来向功能块 SMC_GCodeViewer 传递 CNC 文件最后一行已解码的字符串。
SMC_GCODEVIEWER_DATA	该结构体用一队数组作为SMC_GCodeViewer功能块的缓存；一个该类型的变量存储一条曲线轨迹对应的G代码文件。
SMC_GCODE_WORD	它包括某个 G 代码字如“N20”或“G1”的数据。
SMC_GEOINFO	该结构是被设计用来存储轨迹对象的，也就是已编程轨迹的一段，比如一条直线、一段圆弧或者样条曲线段。
SMC_M_PARAMETERS	它可以为当前活动的 M 函数定义附加参数，这些参数可被 SMC_GetMParameters 计值，以显示各个 M 函数的附加值。
SMC_OUTQUEUE	该结构用来管理一个已定义大小的表中的 GEOINFO 对象。
SMC_POSINFO	对一个特殊位置点该结构描述了它的坐标和附加轴的位置。
SMC_VECTOR3D	该结构体描述了三维的速度。

- 全局变量

指令名称	功能
SMC_AL_STATUS	它的值指明了 FB 实例的实际状态。
SMC_DEC_STATUS	它的值指示了 FB 实例的实际状态。
SMC_DIRECTION	它的值指明了功能块实例的实际插补方向。
SMC_INT_STATUS	它的值指示了 FB 实例的实际状态。可能的状态：
SMC_INT_VELMODE	它的值确定了速度分布图。
SMC_MOVTYP	它是 SMC_GEOINFO 的一个成员，它指明了对象的几何形状。
SMC_TC_STATUS	它的值代表 FB 实例的实际状态。
SMC_TOOLCORRMODE	它的值代表 SMC_Toolcorr 运行时的模式。
SMC_VECTORDIR	它作为SMC_CalcDirectionFromVector的输入变量，用来定义速度的倾角、反向或垂直。

功能块记录

- 在 SMC_CNC_POUs\File 文件夹内

指令名称	功能
SMC_READNCFILE	它从控制器的文件系统中读取一个 NC-ASCII 文档，使得该文档对 SMC_NCDecoder 可用。因此在运行时一个 NC 程序可被读入并被执行。
SMC_READNCQUEUE	它从 PLC 文件系统中读取一个由 CNC 编辑器创建的 OutQueue 文件，并提供一个 OutQueue 结构，典型地该结构是由解码器处理的。
SMC_VARLIST	IEC 61131-3 标准不可能通过字符串的符号名来决定一个变量的值。但这是必要的，假如变量的功能由 SMC_CNC_REF 提供给用户使用，也应该可以从一个文件中读取 CNC 程序。这个可以被结构 SMC_VARLIST 所管理。

● SMC_CNC_POUs\SoftMotion CNC\Coordinate Transformation 文件夹内

指令名称	功能
SMC_DETERMINECUBOIDBEARING	它用给出的 6（3/2/1）个点确定了一个立方体（角标志，边对齐）在空间中的位置。
SMC_COORDINATETRANSFORMATIONS3D	它计算在旧坐标系中给出、并转换到新坐标系下的点的坐标。坐标变换是由原点的变换和新单位向量（按旧坐标系给出）来定义的。
SMC_INVCOORDINATETRANSFORMATION3D	它计算一个在新坐标系下给出的点的坐标并将其换算到旧坐标系中。
SMC_TEACHCOORDINATESYSTEM	辅助用户计算新坐标系的向量。
SMC_UNITVECTORTORPY	从新坐标系的单位向量计算 RPY-角度（参考旧坐标系）
SMC_ROTATEQUEUE2D	在该模块执行时，存储在 poqDataIn 中的轨迹将被绕 Z 轴旋转给出的角度 dPhi[°]。正角度是指数学上的正向旋转（逆时针）。
SMC_SCALEQUEUE3D	在它执行时，在 poqDataIn 所包含的轨迹将被因数 fScaleFactor 拉伸。
SMC_TRANSLATEQUEUE3D	通过它的执行存储在 poqDataIn 中的路径将会按照 vec 转换，vec 是 SMC_VECTOR3D 的数据类型。

● SMC_CNC_POUs\SoftMotion CNC\Direct Axis Control 文件夹内

指令名称	功能
SMC_CONTROLAXIXBYPOS	用于向驱动结构写目标位置，以测试跳跃的结构。它被广泛用于 CNC 和 SMC_Interpolator。对于方向控制轴的设置值，应该使用始于 SMC_Follow*的一个功能块。
SMC_CONTROLAXIXBYPOSVEL	除了在添加时速度可以被定义以外，它的使用等同于 SMC_ControlAxisByPos。它通常用于 CNC 和 SMC_Interpolator。

● SMC_CNC_POUs\SoftMotion CNC\GCode Viewer 文件夹内

指令名称	功能
SMC_GCODEVIEWER	它可以与 SMC_NCDecoder 结合使用。

● SMC_CNC_POUs\SoftMotion CNC\Posinfo Functions 文件夹内

指令名称	功能
SMC_POSINFO	对一个特殊位置点该结构描述了它的坐标和附加轴的位置。

● SMC_CNC_POUs\SoftMotion CNC\OutQueue Functions 文件夹内

指令名称	功能
SMC_APPENDOBJ	该函数块用来处理 SMC_OUTQUEUE 结构体对象。
SMC_DELETEOBJ	该布尔型函数用来处理一个 SMC_OUTQUEUE 结构对象。
SMC_GETCOUNT	类型为 UNIT 的函数将返回存储在 SMC_OUTQUEUE 表（POQ）中的对象个数。
SMC_GETOBJ	假如输入 poq 被正确初始化并含有至少 N+1 个元素，函数将返回一个指向表中第 N 个 GEOINFO 对象的指针，计数从表的开头算起；所以对于 N=0 第一个元素被返回。
SMC_GETOBJFROMEND	假如输入 poq 被正确初始化并含有至少 N+1 个元素，函数将返回一个指向表中第 N 个 GEOINFO 对象的指针，计数从表的结尾算起；所以对于 N=0 最后一个元素被返回。
SMC_OUTQUEUEINIT	它设置 SMC_OUTQUEUE 数据结构的变量为它的默认值。
SMC_RESTOREQUEUE	该函数块将恢复一个已被插补的或已被处理了的结构。仅当整个轨迹能存储到由 poq 引用的表中时，这才是可能的。
SMC_SETQUEUECAPACITY	用于设置 SMC_OUTQUEUE 数据结构的变量空间。

● SMC_CNC_POUs\SoftMotion CNC\SoftMotion Function Blocks 文件夹内

指令名称	功能
SMC_AVOIDLOOP	该功能块被用来做轨迹预处理。它拷贝一个轨迹，但并不剪断其内部含有的环圈。
SMC_CHECKFORLIMITS	用于检查轨迹是否离开其特定的矩形范围。
SMC_CHECKVELOCITIES	该模块检查特殊轨迹段的轨道速度。
SMC_EXTENDEDVELOCITYCHECKS	它减少路径的速度，加速度和减速度，使附加轴(Z, P, Q, U,V, W)产生的速度加速度和减速的值不超过 SMC_GEOINFO 变量（adVelAddAx[0..6], adAccAddAx[0..6] and adDecAddAx[0..6]）的设置值。
SMC_INTERPOLATOR	这个功能块用来把由 SMC_GEOINFO 对象描述的一个连续轨迹转化为离散轨迹位置点，这些点考虑到了速度分布图和时间图。
SMC_INTERPOLATOR2DIR	从函数以及输入输出的配置看，该模块相当于功能块 SMC_Interpolator，但是它还能够反向地插补一个轨迹。
SMC_INTERPOLATOR2DIR_SLOWTASK	该模块负责产生反向轨迹。它被从 SMC_Interpolator2Dir 中分割出来，这样它就能在完全拉伸的低性能系统中被封装给一个低优先级任务。
SMC_LIMITDYNAMICS	该模块用于减小轨迹的速度、加减速度，以使附加轴（Z, P, Q, U, V, W）的速度、加减速值不超过 FB 的输入值 dMaxVel, dMaxAcc 和 dMaxDec。
SMC_LIMITCIRCULARVELOCITY	该模块检查 OutQueue 中的特定元素并限制圆形元素沿其半径反方向的轨迹速度。
SMC_NCDECODER	该模块用来将一个在 CNC 编辑器中被创建的 CNC 程序（DIN 6625，G 代码）转化为 SoftMotion-GEOINFO 结构对象的表。在每次循环中程序里的一行将被解码。

SMC_ROUNDPATH	SMC_ROUNDPATH 功能块与 SMC_SmoothPath 模块十分相象。它由两线相交圆整截得的圆弧的角度。
SMC_SMOOTHPATH	SMC_SmoothPath 功能块可用于轨迹预处理。它可使轨迹尖角(path angles)变得圆滑，从而生成一个圆滑的轨迹（slur path）。 它用于对轨迹的精度要求不高，而对速度要求较高，且不允许速度降到 0 的情况。
SMC_TOOLCORR	The SMC_ToolCorr 用于路径预处理。
SMC_XINTERPOLATOR	该SMC_XInterpolator功能块实现了CAM和CNC的混合。假设你想把一个工件切一个特定的形状（由G代码描述），根据工件由另一个进程的移动（例如，沿X轴移动）和其他的轴（y，z等）应当按照工件（X）当前位置控制，并且目标由路径要点给出。
SMC_GETMPARAMETERS	插补器正处理一个 M 函数，该模块用来轮询已为该 M 函数设置的参数（K，L，O，见 HERE）。
SMC_PREACKNOWLEDGEMFUNCTIONS	它用来在插补器到达该模块之前确认一个 M 函数。这样在 M 函数上的停顿就能够避免了。

● SM_Trafo_POUs\Additional FBs 文件夹内

指令名称	功能
SMC_CALCDIRECTIONFROMVECTOR	使用它时工具的方向（dAlpha）可被计算。

● SM_Trafo_POUs\Gantry Systems 文件夹内

指令名称	功能
SMC_TRAFO_GANTRY2	对于龙门系统不能进行变换，因此模块只能给 x,y 轴加上偏移量。
SMC_TRAFO_GANTRY2TOOL1	功能块用来反求龙门系统的刀具轴向偏置，也就是相对 Z 轴的轴向偏置，它们近似在一条直线上。
SMC_TRAFO_GANTRY2TOOL2	功能块用来反求龙门系统的刀具轴向偏置，也就是相对 Z 轴的轴向偏置，它们近似成直角。
SMC_TRAFO_GANTRY3	对于龙门系统不能进行变换，因此模块只能给 x,y,z 轴加上偏移量。
SMC_TRAFO_GANTRYCUTTER2	为一个正在被控制并指出沿着路径切线旋转轴的三维龙门系统计算一个逆向变换。
SMC_TRAFO_GANTRYCUTTER3	为一个正在被控制并指出沿着路径切线旋转轴的三维龙门系统计算一个逆向变换。
SMC_TRAFO_GANTRYH2	这个转变模块用来提供解决有着固定的驱动器的H龙门系统的反向转变。
SMC_TRAFOF_GANTRY2	功能模块用来操作一个没有动作的三维龙门操作系统，因此，该模块仅在X,Y轴各自增加了一个偏移。每一个SMC_TRAFOF_GANTRY2模块的实例都能与一个命名为SMC_VISU_GANTRY2的可视化模板相连接。
SMC_TRAFOF_GANTRY2TOOL1	功能块用来反求龙门系统的刀具轴向偏置，也就是相对 Z 轴的轴向偏置，它们近似在一条直线上。
SMC_TRAFOF_GANTRY2TO	功能块用来反求龙门系统的刀具轴向偏置，也就是相对 Z 轴

OL2	的轴向偏置，它们近似成直角。
SMC_TRAFOF_GANTRY3	功能模块用来操作一个没有动作的三维龙门操作系统，因此，该模块仅在X,Y,Z轴各自增加了一个偏移。每一个SMC_TRAFOF_GANTRY3模块的实例都能与一个命名为SMC_VISU_GANTRY3的可视化模板相连接。
SMC_TRAFOF_GANTRYCUTTER2	该模块将解决有着指向沿着切线方向的旋转轴的三维龙门系统的前向转变。
SMC_TRAFOF_GANTRYCUTTER3	它将解决指向当前轨迹的切线方向的旋转轴的三维龙门系统的前向转变。
SMC_TRAFOF_GANTRYH2	这个转变模块用来提供解决有着固定的驱动器的H龙门系统的前向转变。
SMC_TRAFOV_GANTRY2	反向转换模块被用来提供处理考虑轨迹速度轨迹方向作为轴的控制变量的二维龙门系统。
SMC_TRAFOV_GANTRY3	反向转换模块被用来提供处理考虑轨迹速度轨迹方向作为轴的控制变量的三维龙门系统。
SMC_TRAFOV_GANTRYCUTTER2	反向转换模块被用来处理考虑轨迹速度轨迹方向作为轴的控制变量的二维龙门系统。
SMC_TRAFOV_GANTRYCUTTER3	反向转换模块被用来提供处理考虑轨迹速度轨迹方向作为轴的控制变量的三维龙门系统。
SMC_TRAFOV_GANTRYH2	反向转换模块被用来提供处理考虑轨迹速度轨迹方向作为轴的控制变量的二维 H 龙门系统。

● SM_Trafo_POUs\Parallel Systems 文件夹内

指令名称	功能
SMC_TRAFO_BIPOD_ARM	转换模块用来提供处理双脚架臂的后向转换。
SMC_TRAFO_TRIPOD	转换模块用来提供处理三脚架的后向转换。
SMC_TRAFO_TRIPOD_ARM	转换模块用来提供处理三脚架臂的后向转换。
SMC_TRAFOF_BIPOD_ARM	转换模块用来提供处理双脚架臂的前向转换。
SMC_TRAFOF_TRIPOD	转换模块用来提供处理三脚架的前向转换。
SMC_TRAFOF_TRIPOD_ARM	转换模块用来提供处理三脚架臂的前向转换。

● SM_Trafo_POUs\Scara Systems 文件夹内

指令名称	功能
SMC_TRAFO_SCARA2	转换模块用来提供处理第二关节 Scara 系统的后向转换。
SMC_TRAFO_SCARA3	转换模块用来提供处理第三关节 Scara 系统的后向转换。
SMC_TRAFOF_SCARA2	转换模块用来提供处理第二关节 Scara 系统的前向转换。
SMC_TRAFOF_SCARA3	转换模块用来提供处理第三关节 Scara 系统的前向转换。

I 订货信息

表 I-1 订货信息一览表

规格描述	订货号
C35-00 运动控制器	
CTH300-C 系列运动控制型 CPU，32MB 程序数据空间，32KB 掉电保持数据，55M 扩展总线；1 个自由通信接口，2 个 EtherCAT 接口，1 个 EtherNET 接口，1 个 USB 口	CTH3 C35-000S1
C36-00 运动控制器	
CTH300-C 系列运动控制型 CPU，32MB 程序数据空间，32KB 掉电保持数据，55M 扩展总线；1 个自由通信接口，2 个 EtherCAT 接口，1 个 EtherNET 接口，1 个 USB 口，支持 SoftMotion	CTH3 C36-000S1
C37-00 运动控制器	
CTH300-C 系列运动控制型 CPU，32MB 程序数据空间，32KB 掉电保持数据，55M 扩展总线；1 个自由通信接口，2 个 EtherCAT 接口，1 个 EtherNET 接口，1 个 USB 口，支持 SoftMotion，CNC	CTH3 C37-000S1
C35-01 运动控制器	
CTH300-C 系列运动控制型 CPU，32MB 程序数据空间，64KB 掉电保持数据，55M 扩展总线；1 个自由通信接口，1 个 EtherCAT 接口，1 个 EtherNET 接口，一个 CANopen 口，1 个 USB 口	CTH3 C35-001S2
C36-01 运动控制器	
CTH300-C 系列运动控制型 CPU，32MB 程序数据空间，64KB 掉电保持数据，55M 扩展总线；1 个自由通信接口，1 个 EtherCAT 接口，1 个 EtherNET 接口，一个 CANopen 口，1 个 USB 口，支持 SoftMotion	CTH3 C36-001S2
C37-01 运动控制器	
CTH300-C 系列运动控制型 CPU，32MB 程序数据空间，64KB 掉电保持数据，55M 扩展总线；1 个自由通信接口，1 个 EtherCAT 接口，1 个 EtherNET 接口，一个 CANopen 口，1 个 USB 口，支持 SoftMotion，CNC	CTH3 C37-001S2
C37-02 运动控制器	
CTH300-C 系列运动控制型 CPU，32MB 程序数据空间，120KB 掉电保持数据，55M 扩展总线，1 个 RS485 通讯口（MODBUS 自由口协议），1 个 EtherCAT 接口，1 个 EtherNET 接口，1 个 CAN 口，1 个 USB 口，支持单轴运动控制，插补，电子齿轮和电子凸轮等 SoftMotion 指令集功能和 CNC 功能，支持 CODESYS 编程平台 SP11 版本	CTH3 C37-002S2
电源模块	
PWR-02 电源模块，输入 85～264V AC，输出 24V DC/2A	CTH3 PWR-020S1
中继模块	
INT-00 中继模块	CTH3 INT-000S1
高速计数模块	
HSC-02高速计数模块，2路差分/单端信号输入	CTH3 HSC-020S1
脉冲输出模块	
HSP-04脉冲输出模块，4路差分/单端信号输出	CTH3 HSP-040S1

规格描述	订货号
CAN 主站模块（C3X-00 系列支持）	
CAN-1T 主站模块，CANopen 主站通信模块，1 个 CAN 通信口	CTH3 CAN-1T0S1
EtherCAT 从站模块	
EtherCAT 从站模块，最多可挂接8个扩展模块	CTH3 ECT-000S1
数字量模块	
DIT-08 数字量模块，数字量输入 8 x 24VDC	CTH3 DIT-080S1
DIT-16 数字量模块，数字量输入 16 x 24VDC	CTH3 DIT-160S1
DIT-32 数字量模块，数字量输入 32 x 24VDC	CTH3 DIT-320S1
DQT-08 数字模块，数字量输出 8 x 24VDC	CTH3 DQT-080S1
DQT-16 数字量模块，数字量输出 16 x 24VDC	CTH3 DQT-160S1
DQT-32 数字量模块，数字量输出 32 x 24VDC	CTH3 DQT-320S1
DQR-08 数字量模块，数字量输出 8 x 继电器	CTH3 DQR-080S1
DQR-16 数字量模块，数字量输出 16 x 继电器	CTH3 DQR-160S1
模拟量模块	
AIS-04 模拟量模块，模拟量电压电流输入，4AI x 12bit	CTH3 AIS-040S1
AMS-06 模拟量模块，模拟量电压电流输入输出，4AI x 12bit，2AQ x 12bit	CTH3 AMS-060S1
AIV-08 模拟量模块，模拟量电压输入，8AI x 16bit	CTH3 AIV-080S1
AIC-08 模拟量模块，模拟量电流输入，8AI x 16bit	CTH3 AIC-080S1
AQS-04 模拟量模块，模拟量电压电流输出，4AQ x 12bit	CTH3 AQS-040S1
AQS-08 模拟量模块，模拟量电压电流输出，8AQ x 12bit	CTH3 AQS-080S1
温度模块	
AIT-04 温度模块，热电偶输入模块，4 点*TC，隔离型 16bit 精度	CTH3 AIT-040S1
AIT-08 温度模块，热电偶输入模块，8 点*TC，隔离型 16bit 精度	CTH3 AIT-080S1
AIR-04 温度模块，热电阻输入模块，4 点*RTD，隔离型 16bit 精度	CTH3 AIR-040S1
AIR-08 温度模块，热电阻输入模块，8 点*RTD，隔离型 16bit 精度	CTH3 AIR-080S1

服务热线

400-700-4858

深圳市合信自动化技术有限公司

深圳市南山区西丽镇茶光路南深圳成电路设计应用产业园 209

0755-86226822

sales@co-trust.com

www.co-trust.com

官方微信公众号