福氏技术
PRACTEK

# 福氏技术控制平台CT65基础编程手册

V1.0.0.0
2023-12-04

# 目录

# IEC61131-3编程指南

## ➢ CoDeSys IDE介绍

CT65控制器编程环境由工业自动化中广泛使用的CoDeSys IDE提供。PRACTEK基于CoDeSys V3设计了CT65的相关PLC功能，以方便用户使用。

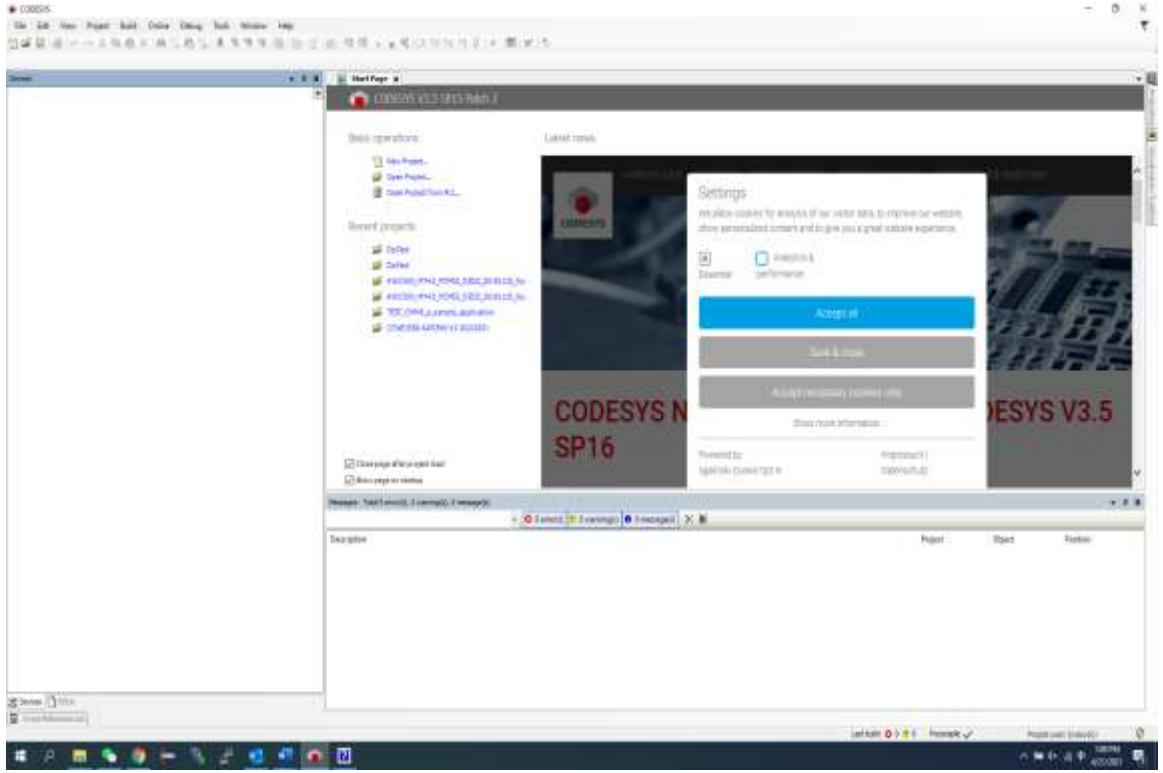
CoDeSys IDE支持IEC61131-3编程标准的所有五种编程语言，包括:

- 指令表(IL)
- 结构化文本(ST)
- 梯形图(LD)
- 功能块图(FBD)
- 顺序功能图(SFC)

# IEC61131-3编程指南

## ➢ CoDeSys IDE介绍

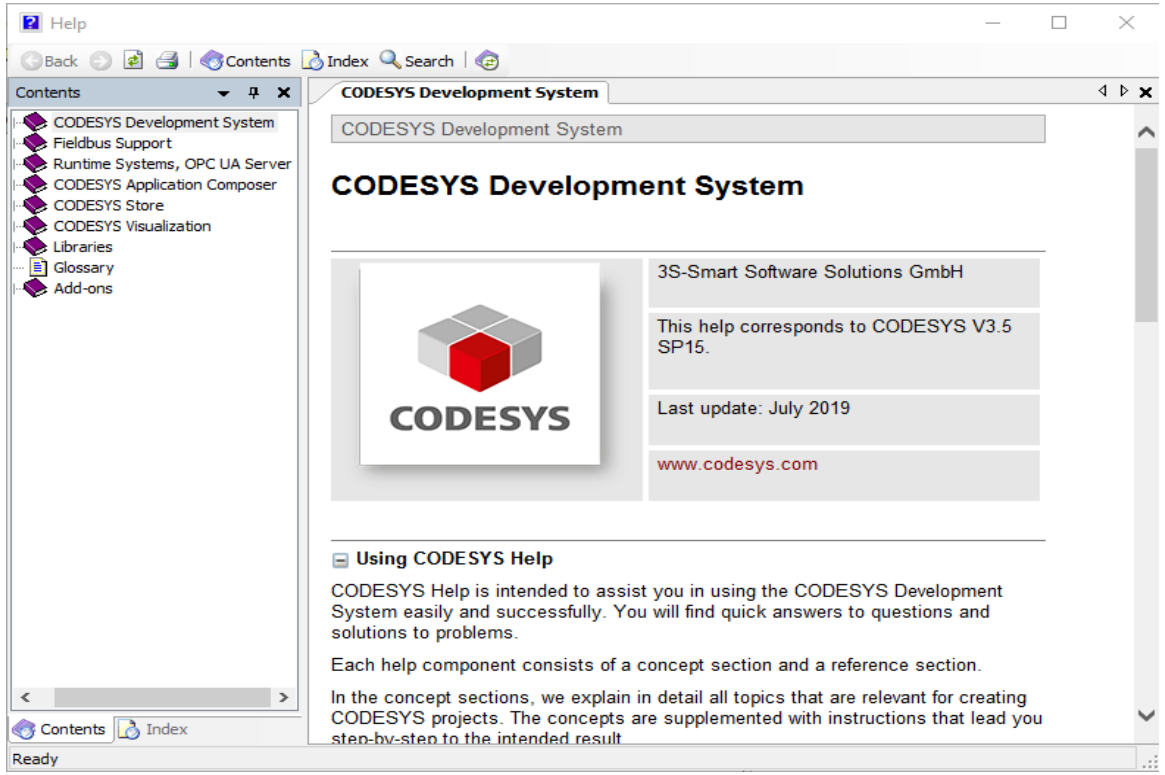
在代码开发电脑中安装了CoDeSys IDE并运行后，将显示下图所示的编程环境画面。



# IEC61131-3编程指南

## ➢ CoDeSys IDE介绍

编程套件，包括各种手册及在线帮助由CoDeSys提供，均包含了英文，德文以及中文三种语言版本。如下图所示。



# IEC61131-3编程指南

## ➢ CoDeSys IDE介绍

CoDeSys IDE包含但不限于以下功能:

- 监控所有变量
- 强制变量值并写入控制器进行
- 调试项目（断点、步进、单周期、呼叫堆栈）
- POUs和数据的无中断在线更改
- 采样跟踪
- 用户定义库的管理
- 离线模拟
- 图形PLC配置
- OPC服务器

# IEC61131-3编程指南

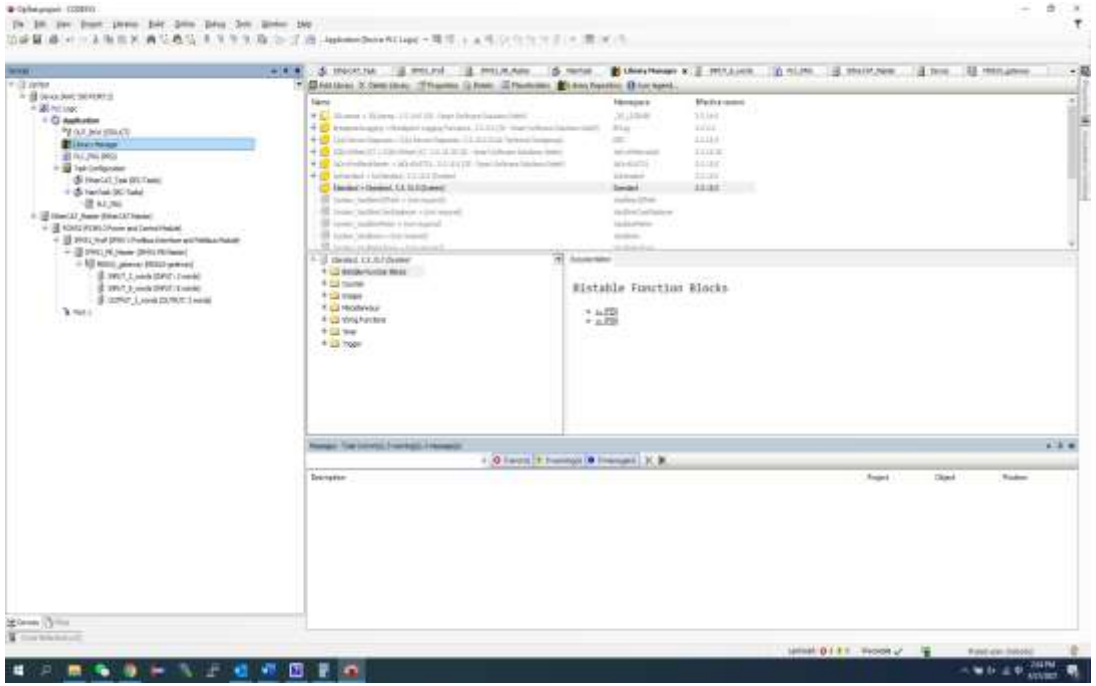
## ➢ CoDeSys IDE介绍

控制器中安装有CoDeSys实时运行系统，使用CoDeSys IDE或其他基于CoDeSys的产品编写的程序可以从其他PLC平台移植到CT65中。但通常代码在平台之间移植会有一些变化，较为普遍的是原平台库文件以及驱动程序中在CT65中不存在，需要重新开发。另外HMI会议问兼容性问题而需要全部重写。基于CoDeSys V2编写的PLC程序可以在CoDeSys V3中导入（仅在CoDeSys V3.4具有该功能），同样需要一些适应性更改。

# IEC61131-3编程指南

## ➢ CoDeSys库管理及标准库

CoDeSys IDE包含各种库文件方便用户使用，依据所需功能可以在项目中的库管理器(Library Manager)中进行添加，下图展示了新建项目中的库管理器的位置。



# IEC61131-3编程指南

## ➢ CoDeSys库管理及标准库

CoDeSys V3提供的标准库如下表所示，这些库不是AWP100新建项目默认添加的库文件，如有需要需要用户手动添加进项目中。

| 库名称 | 功能 |
| --- | --- |
| SysCom.library | 与目标设备进行串行同步通讯 |
| SysComAsync.library | 与目标设备进行串行异步通讯 |
| SysCpuHandling.library | IEC功能调用、测试和复位 |
| SysDir.library | 目标设备上的同步文件管理器 |
| SysDirAsync.library | 目标设备上的异步文件管理器 |
| SysEvent.library | IEC任务事件 |
| SysFile.library | 目标设备同步文件处理系统 |
| SysFileAsync.library | 目标设备异步文件处理系统 |

# IEC61131-3编程指南

➢ CoDeSys库管理及标准库

| 库名称 | 功能 |
| --- | --- |
| SysInt.library | 基于SysMem.library内存管理器提供终端功能 |
| SysPci.library | 连接到系统的PCI总线的访问权限 |
| SysPort.library | 通过端口地址(例如实时时钟等)与外部硬件模块同步通讯 |
| SysPortAsync.library | 通过端口地址(例如实时时钟等)与外部硬件模块异步通讯 |
| SysProcess.library | 目标系统上的流程处理 |
| SysSem.library | 创建、使用同步任务信号 |
| SysSemProcess.library | 创建、使用异步任务信号 |
| SysShm.library | 创建和访问共享内存 |
| SysSocket.library | 通过TCP/IP和UDP访问sockets上的同步通讯 |
| SysSocketAsync.library | 通过TCP/IP和UDP访问sockets上的异步通讯 |
| SysTask.library | 任务管理 |
| SysTime.library | 用于读取控制器实时时钟等 |

# IEC61131-3编程指南

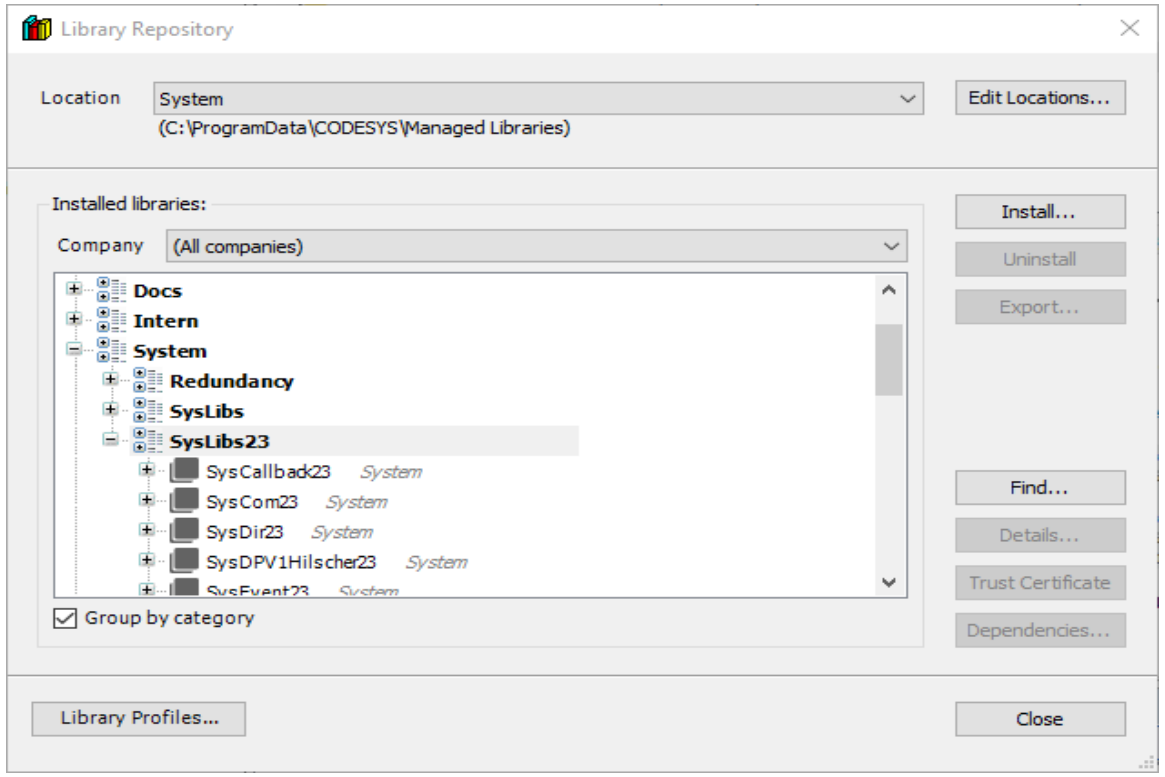
➢ CoDeSys库管理及标准库

| 库名称 | 功能 |
| --- | --- |
| SysTimer.library | 实现触发调用函数事件的分时器 |
| SysTypes.library | 实时系统平台综合 |

# IEC61131-3编程指南

## ➢ CoDeSys库管理及标准库

CoDeSys V3系统库中兼容了一部分CoDeSys V2系统库，相关的库在添加库Sys/SysLibs23目录项下，如下图，如有需要可在项目中添加调用相关库文件。



# IEC61131-3编程指南

## ➢ PRACTEK开发库

PRACTEK针对包括AWP100在内的控制器产品开发了相关的系统功能库。

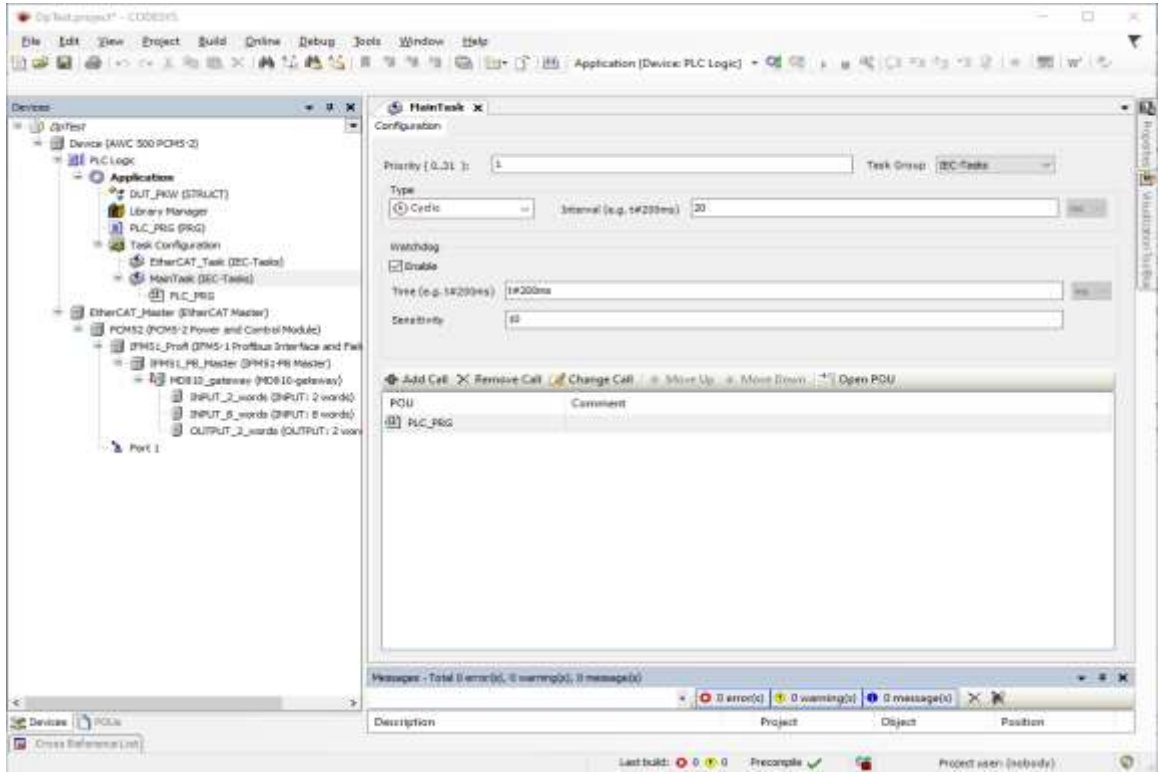
其中较为重要的是Pcm51Clib，它可以对磁盘空间进行监控，并实现CoDeSys实时系统的退出(将导致系统重启)，以及实现系统重启功能。包含表2.3.1所示的函数：

| 函数名称 | 功能 |
| --- | --- |
| **Pcm51DiskFree('/app')** | 该函数反馈设备磁盘空闲的字节数，/app表示Application文件系统的根目录，/mmc表示MMC/SD卡文件系统的根目录，/tmp表示RAM临时文件的根目录 |
| **Pcm51Exit()** | 退出CoDeSys实时系统，运行该函数将导致设备重启 |
| **Pcm51Reboot()** | 设备重启 |

# IEC61131-3编程指南

## ➢ Task任务看门狗功能

CT65具备对某一Task任务运行是否时间过长或导致CPU超载的检测和控制功能，其可以通过下图所示的看门狗设置功能启动。



# IEC61131-3编程指南

## ➢ Task任务看门狗功能

操作方法步骤如下:

- 勾选Wathdog下的Enable复选框
- 设置时间，例如T#200ms
- 设置敏感值，例如10

- **看门狗设置时间必须大于Task任务运行周期**
- **触发机制：例如：设置时间为T#200ms，敏感值为10，当单次任务运行时间超过200ms*10时，或当连续10次任务运行时间均超过200ms时，将导致看门狗触发**

# IEC61131-3编程指南

- **Task任务看门狗功能**

  当看门狗异常触发时，数字输出的状态将被设置为默认状态。

  禁用或重新使能看门狗功能也可以通过接口函数的方式在程序内部进行设置。

```
hIecTask := IecTaskGetCurrent(0);
IecTaskDisableWatchdog(hIecTask);
... // Code that is protected against watchdog
IecTaskEnableWatchdog(hIecTask);
```

# IEC61131-3编程指南

## ➢ Task任务看门狗功能

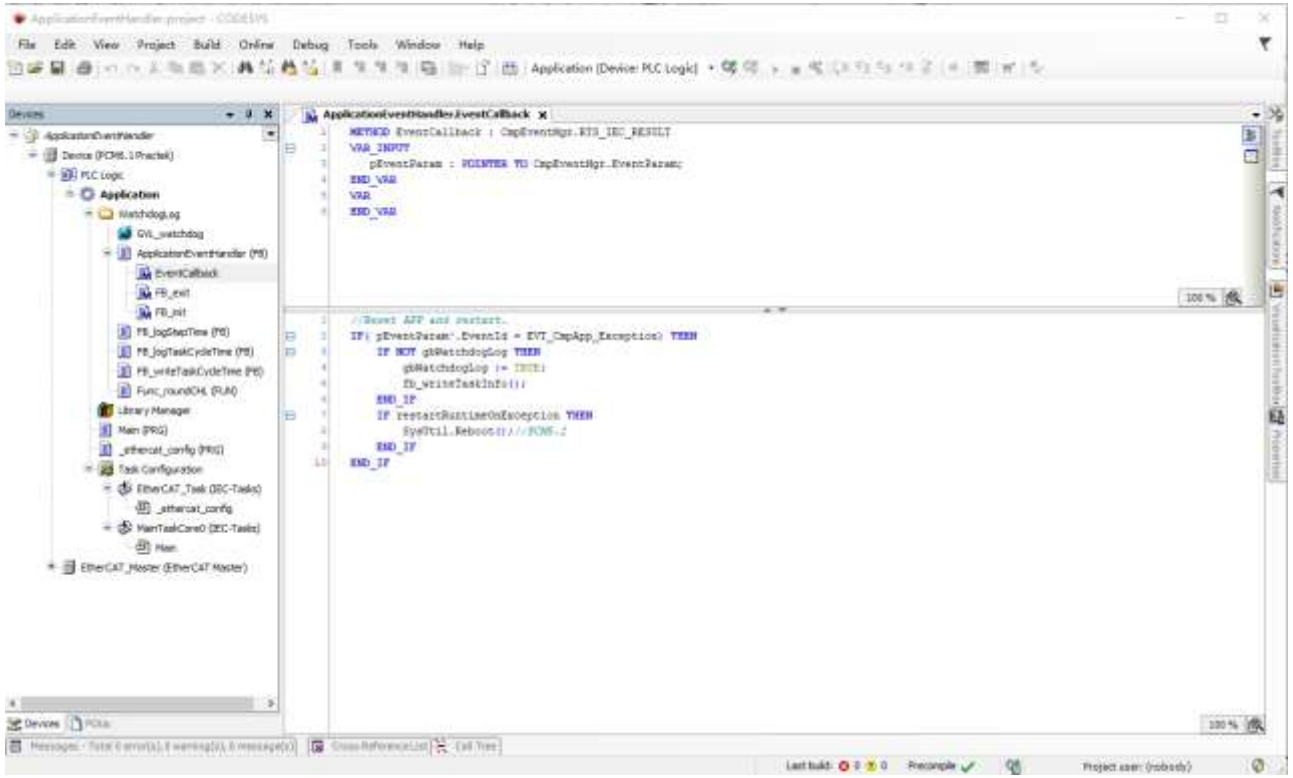
当包括看门狗在内的系统故障触发时，实时系统将停止运行并触发Exception故障，此时程序内变量将保持故障前状态不变。

- **看门狗的触发通常是由于代码中含有对空指针的调用、死循环、除零等异常情况。**
- **请务必评估当Exception故障时系统的输出是否会产生严重意外，并酌情进行必要的规避和处理**

# IEC61131-3编程指南

## ➢ Task任务看门狗功能

针对异常系统故障的处理，可以使用CoDeSys IDE提供的接口回调程序，并在回调程序中设置当出现异常系统故障是采用何种操作处理，例如重启。

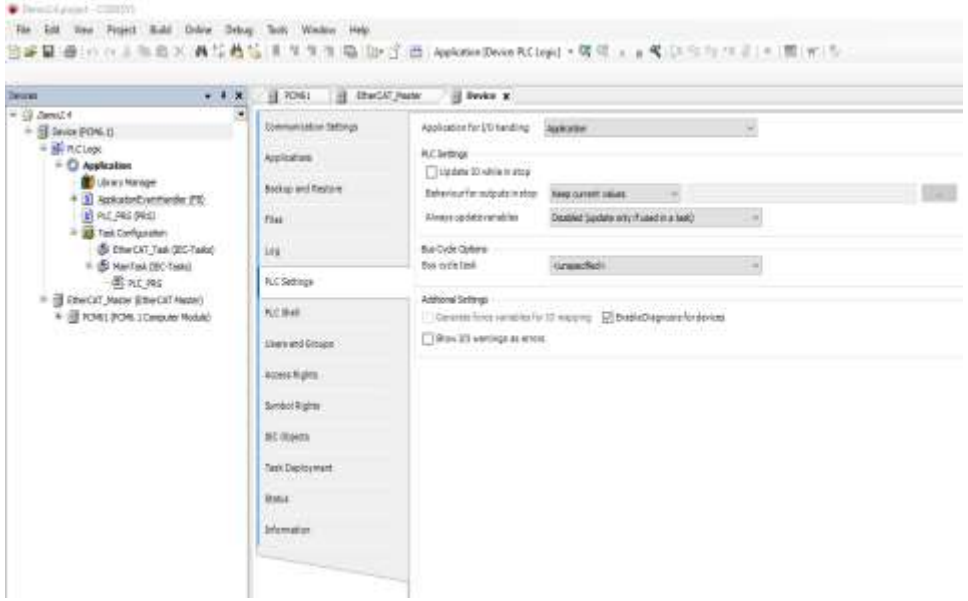


# IEC61131-3编程指南

## ➢ 默认IO输出状态

对于IO输出，当异常系统故障发生时可以被设置为默认状态，如下图所示，在Device页面PLC Setting项下：

- 使能Update IO while in stop
- 设置Behaviour for outputs in Stop为Set all outputs to default即可。



# IEC61131-3编程指南

## ➢ 硬件属性及状态监控

AWP100提供全系列产品的硬件属性及状态监控功能，常用功能块如表2.6所示。

| 功能块 | 参数/属性 | 功能/返回值 |
|---|---|---|
| **IoDrvEtherCAT** | NumberActiveSlaves | 返回实际连接的从站数量 |
| | xConfigFinished | 返回配置是否完成的状态 |
| | xError | 返回EtherCAT总线错误或故障 |
| | LastMessage | 返回EtherCAT总线最新的消息 |
| **ETCSlave** | VendorID | 返回供应商ID |
| | ProductID | 返回产品ID |
| | wState | 返回从站当前状态。枚举值如下:<br>0:ETC_SLAVE_BOOT<br>1:ETC_SLAVE_INIT<br>2:ETC_SLAVE_PREOPERATION<br>4:ETC_SLAVE_SAFEOPERATION<br>8:ETC_SLAVE_OPERATION |

# IEC61131-3编程指南

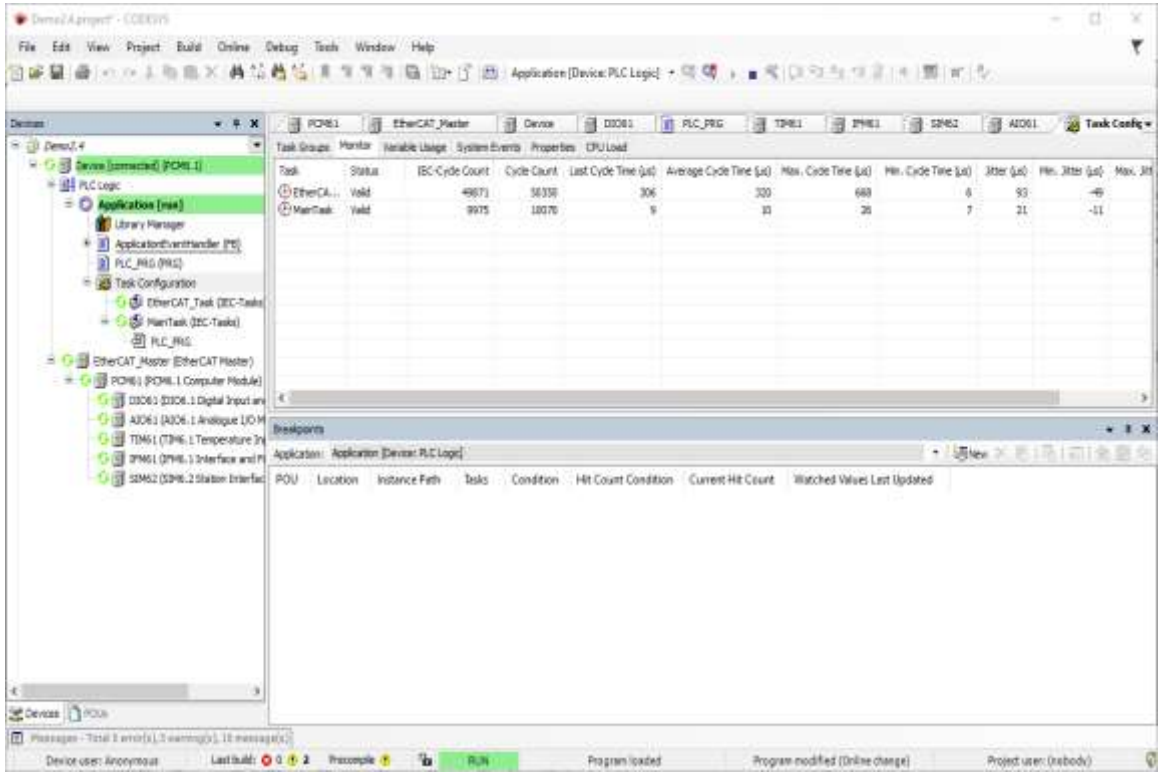
## ➢ 硬件属性及状态监控

| 功能块 | 参数/属性 | 功能/返回值 |
|---|---|---|
| AIO6·1 | AI x Under range | 当输入值低于通道检测下限时返回TURE |
| | AI x Over range | 当输入值高于通道检测上限时返回TURE |
| | AI x Error | 通道故障(包括超过通常检测上下限等故障时) |
| TIM6·1 | RTD x Under range | 无效 |
| | RTD x Over range | 当输入值高于通道检测上限时返回TURE |
| | RTD x Error | 通道故障(包括超过通常检测上限等故障时) |

# IEC61131-3编程指南

## ➢ 控制器负载监控

在CoDeSys中，可以通过Task Configuration/Monitor中查看任务周期时间和统计数据，如下图



# IEC61131-3编程指南

## ➢ 控制器负载监控

或者，也可以测量每个子系统的执行时间。

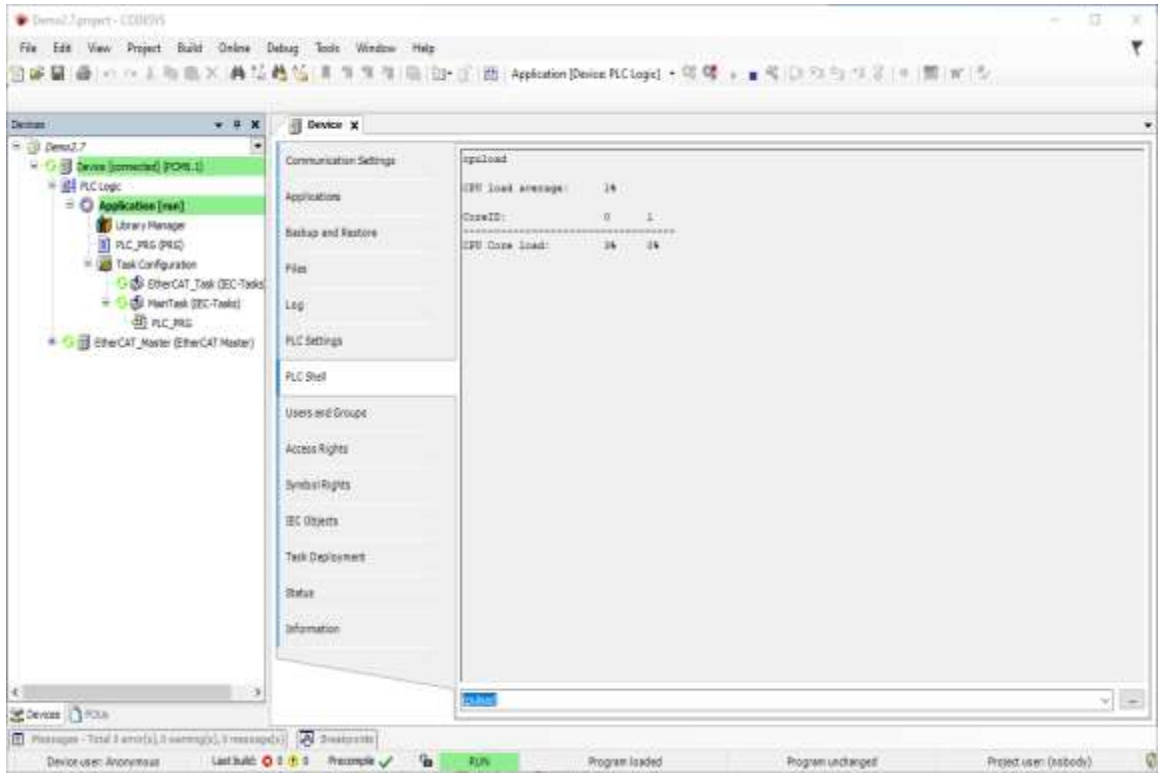
```
//循环时间统计的方法1，手动计时计算循环时间
//在任务代码开始和末尾分别获取时间戳然后做减法得到实际运行时间
//Task开始时间戳
SysTimeCore.SysTimeGetUs(pUsTime := start);

//循环时间统计的方法2，获取系统统计的循环时间结果，单位us
hTask := CmpIecTask.IecTaskGetCurrent(pResult:=ADR(Result));
IF hTask <> ISysTypes.RTS_INVALID_HANDLE AND hTask <> 0 THEN
    pTaskInfo2 := CmpIecTask.IecTaskGetInfo3(hIecTask:=hTask, pResult:=ADR(Result));
    dwInterval:= pTaskInfo2^.dwInterval;
    dwCycleTime := pTaskInfo2^.dwCycleTime;
    dwMinCycleTime := pTaskInfo2^.dwMinCycleTime;
    dwMaxCycleTime := pTaskInfo2^.dwMaxCycleTime;
    dwAvgCycleTime := pTaskInfo2^.dwAverageCycleTime;
ELSE
    dwCycleTime := 20000;
    dwMinCycleTime := 3000;
    dwMaxCycleTime := 3000;
    dwAvgCycleTime := 3000;
END_IF;

//循环消耗时间，增加代码运行时间
IF test THEN
    FOR cnt := 1 TO 65534 DO
        ;
    END_FOR
END_IF

//Task结束时间戳
SysTimeCore.SysTimeGetUs(pUsTime := end);

//Task实际时间消耗，单位us
cycletime := end - start;
```

需要确保实时运行任务的实际运行时间消耗小于任务运行周期，并尽可能的短，这是保证系统不至于因为任务超时导致系统故障的有效方法。

# IEC61131-3编程指南

## ➢ 控制器负载监控

在Debug或测试时，PLC的负载情况也可通过PLC Shell、网页或通过SSH登录到Linux系统后运行top指令查看。



# IEC61131-3编程指南

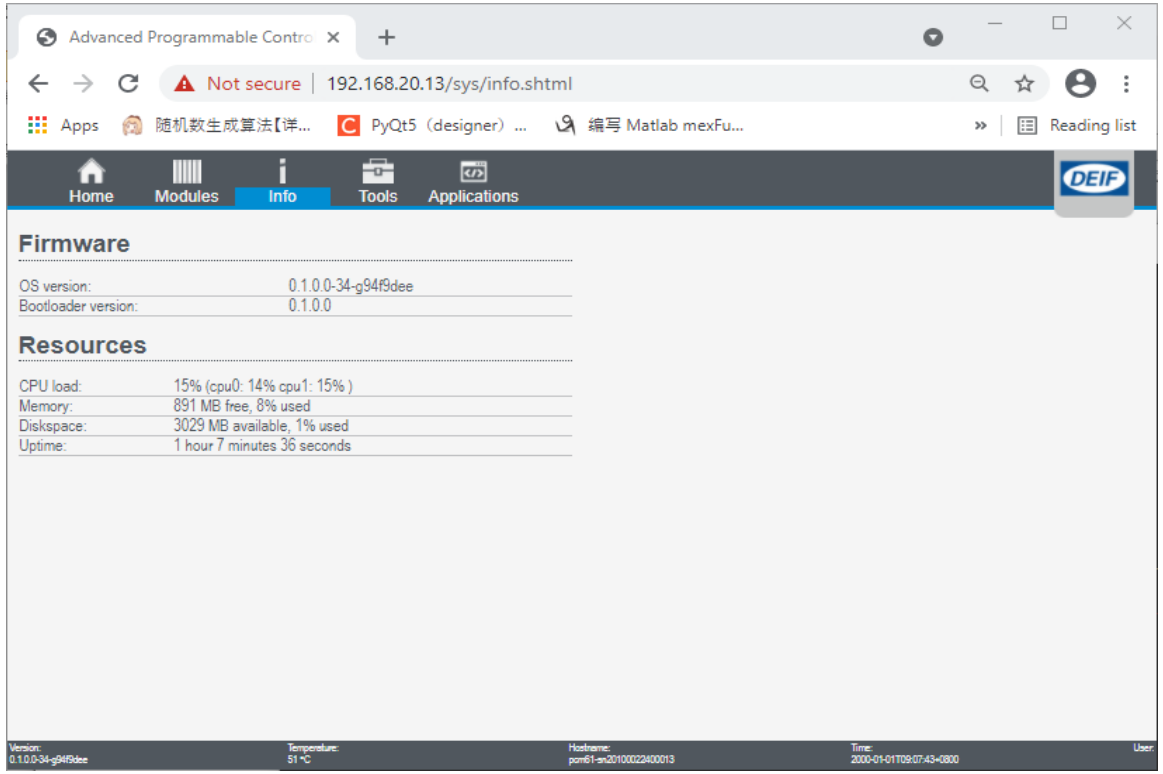
## ➢ 控制器负载监控

192.168.20.13 - PuTTY

```
login as: root
root@192.168.20.13's password:
Last login: Sat Jan  1 13:55:12 2000 from 192.168.20.16


BusyBox v1.25.1 (2021-02-17 16:34:25 UTC) built-in shell (ash)

~ # top
Mem: 111120K used, 917600K free, 932K shrd, 17408K buff, 37568K cached
CPU:  4.5% usr  9.0% sys  0.0% nic 86.3% idle  0.0% io  0.0% irq  0.0% sirq
Load average: 0.28 0.23 0.18 1/189 1599
  PID  PPID USER     STAT   VSZ %VSZ %CPU COMMAND
 1159  1158 root     S    32184  3.1  8.1 ./codesyscontrol
 1599  1596 root     R     2676  0.2  5.4 top
  685   395 root     S    38424  3.7  0.0 fnvramd -l -f -t 60 /nvram /data/.nvra
  745   400 root     S    11480  1.1  0.0 netc daemon -c /etc/netc.json -r 6
  607   409 root     S    10048  0.9  0.0 reset-button -s reset-button-system-re
  861   398 root     S     5032  0.4  0.0 lighttpd -D -f /etc/lighttpd.conf
 1590   815 root     S     4604  0.4  0.0 sshd: root@pts/0
  815   404 root     S     4604  0.4  0.0 /usr/sbin/sshd -D -e
  905   883 nobody   S     2684  0.2  0.0 tr -d
 1596  1590 root     S     2676  0.2  0.0 -sh
  770   764 root     S     2676  0.2  0.0 /sbin/getty 115200 /dev/console
  225   211 root     S     2584  0.2  0.0 s6-fdholderd -l -i data/rules
 1592   793 root     S     2544  0.2  0.0 sleep 300
 1593   794 root     S     2544  0.2  0.0 sleep 300
 1597   795 root     S     2544  0.2  0.0 sleep 30
  633   392 root     S     2544  0.2  0.0 inotifyd dupdate-inotifyd-agent /tmp/f
```

# IEC61131-3编程指南

## ➢ 控制器负载监控



# IEC61131-3编程指南

## ➢ EtherCAT硬件扫描

CT65系列产品支持模块配置的扫描功能，省去了手动逐个添加硬件模块的烦恼。方法如下：

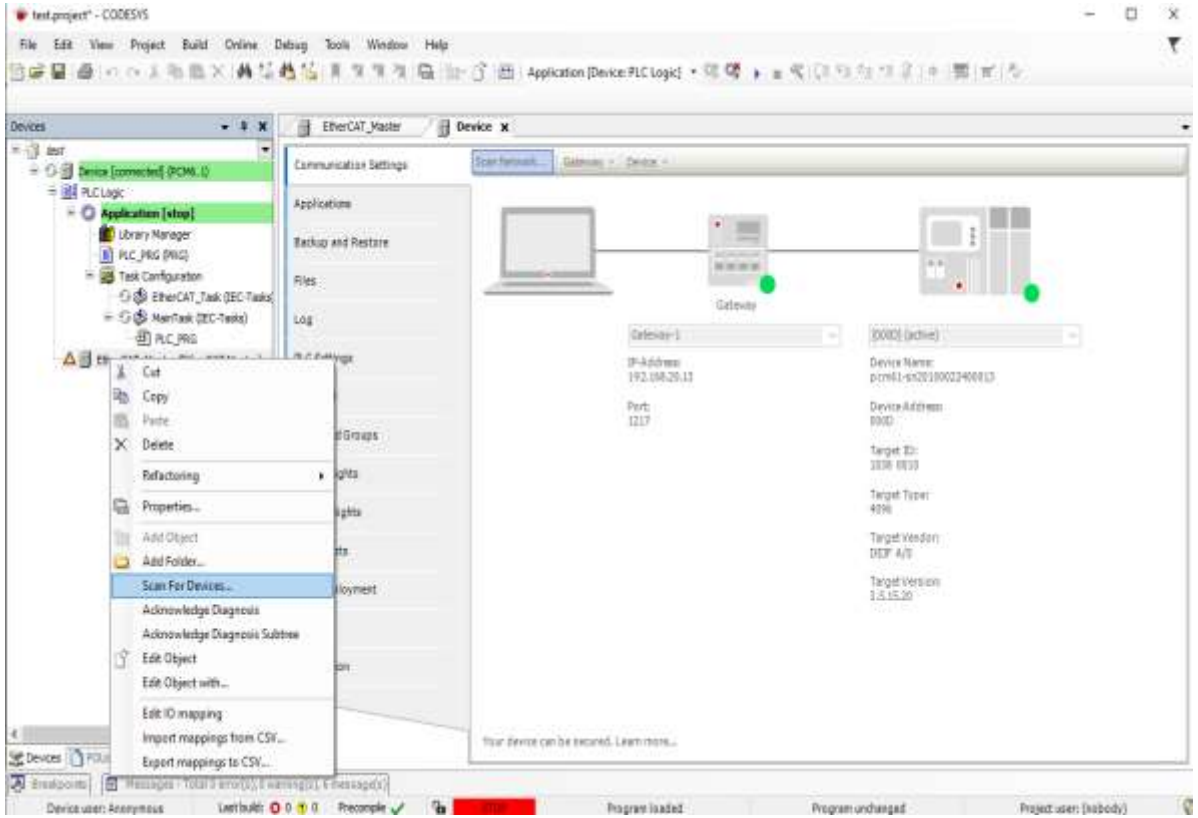
- 新建工程，选择默认的Standard project
- 添加设备，选择EtherCAT并选中EtherCAT Master
- 修改配置为Select network by name，并将Network Name修改为ecat0

- **默认的以MAC来搜索网络会使得开发的工程只能应用去该MAC地址设备，无法部署到其他设备中。这不利于工程的现场应用。**

# IEC61131-3编程指南

## ➢ EtherCAT硬件扫描

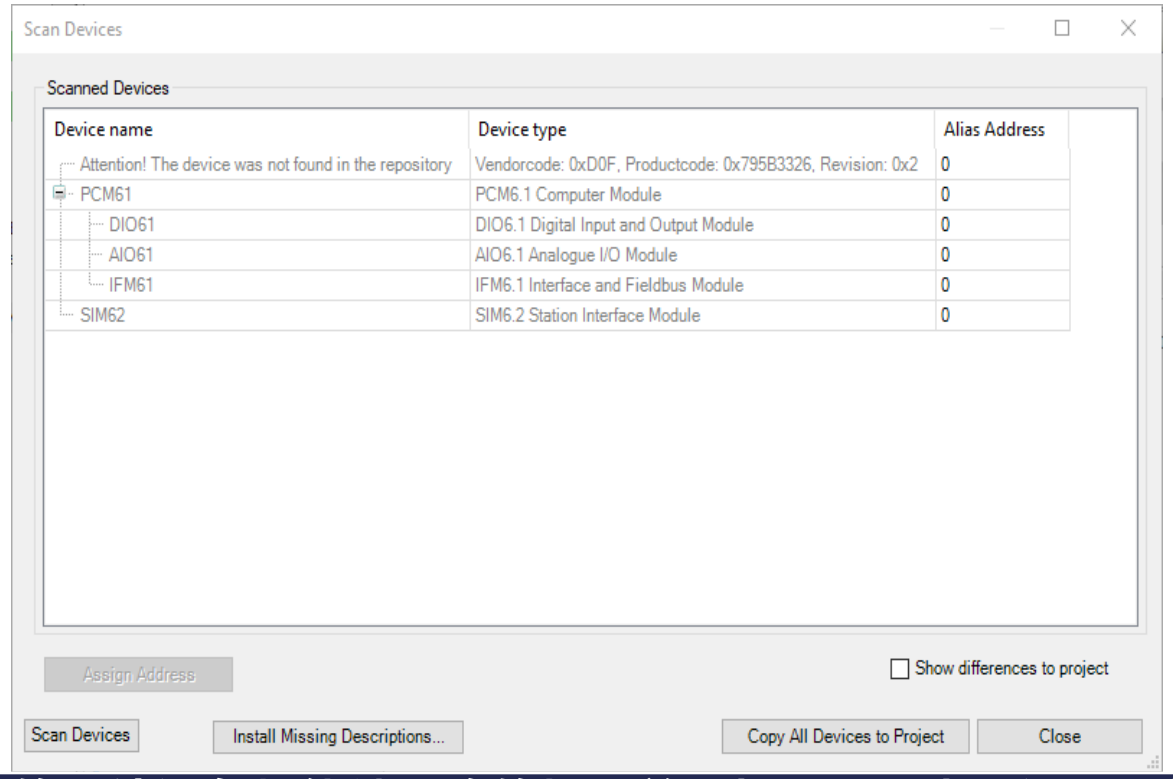
- Login，但不要运行工程
- 右击EtherCAT_Master选中Scan for Devices



# IEC61131-3编程指南

## ➢ EtherCAT硬件扫描

- 所有在线设备将自动被扫描出来，点击Copy All Devices to Project，相关配置将被导入到工程中。如下图



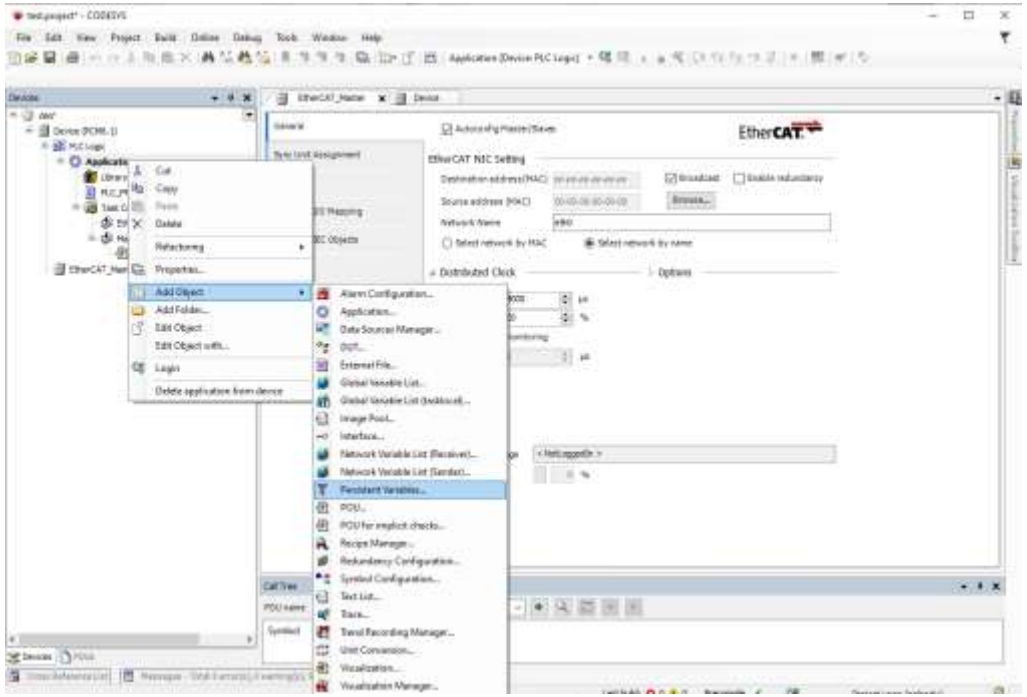

- **不是所有的硬件设备都能被正确的扫描并添加入工程中，设置因为设备故障导致其他设备不能被添加入工程。因此仍然建议核对已扫描添加的设备或手动添加设备。**

# IEC61131-3编程指南

## ➢ EtherCAT硬件扫描持久型变量(Persistent variables)

需要分配掉电不丢失的参数时可将其声明为持久型变量，这些变量会被实时储存，防止意外导致数据丢失。

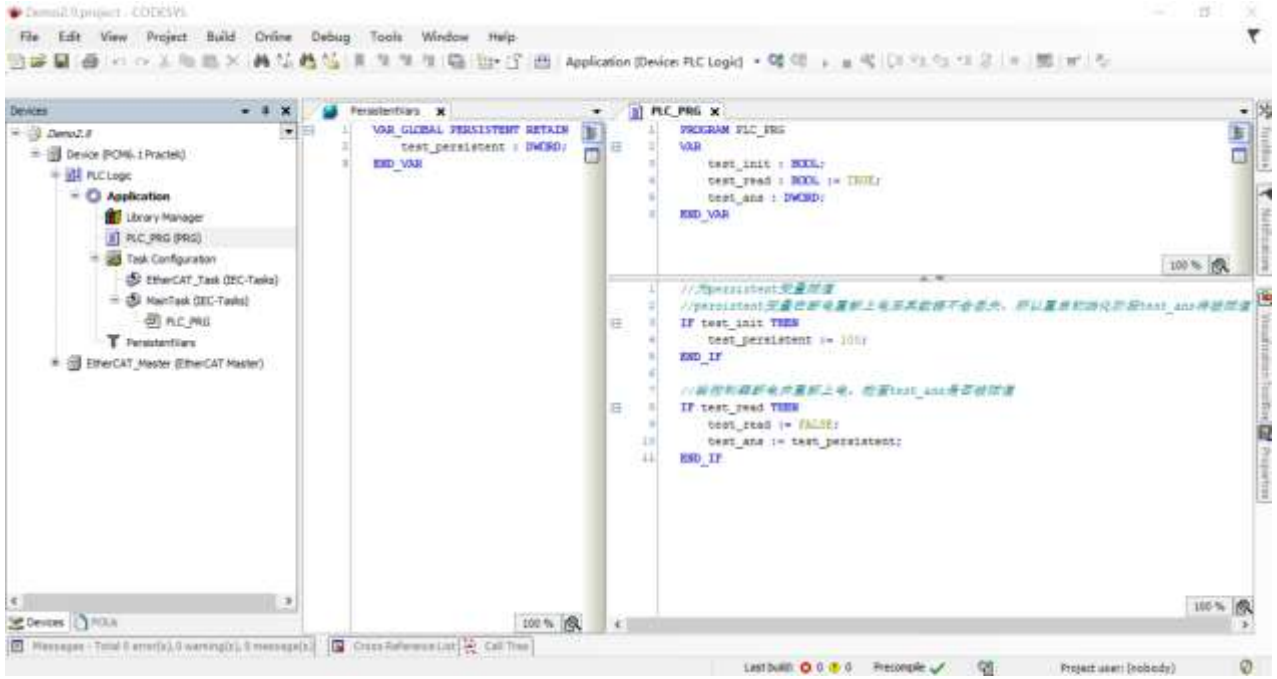
这些变量需要在Persistent Variables块内被声明才具有上述功能，添加Persistent Variables的方法如下图所示。



# IEC61131-3编程指南

## ➢ EtherCAT硬件扫描持久型变量(Persistent variables)

虽然持久型变量具有不易丢失的特性，但仍然建议在此基础上将相关数据保存在备份文件中，以防备可能的器件损坏，工程误更新等导致数据丢失。



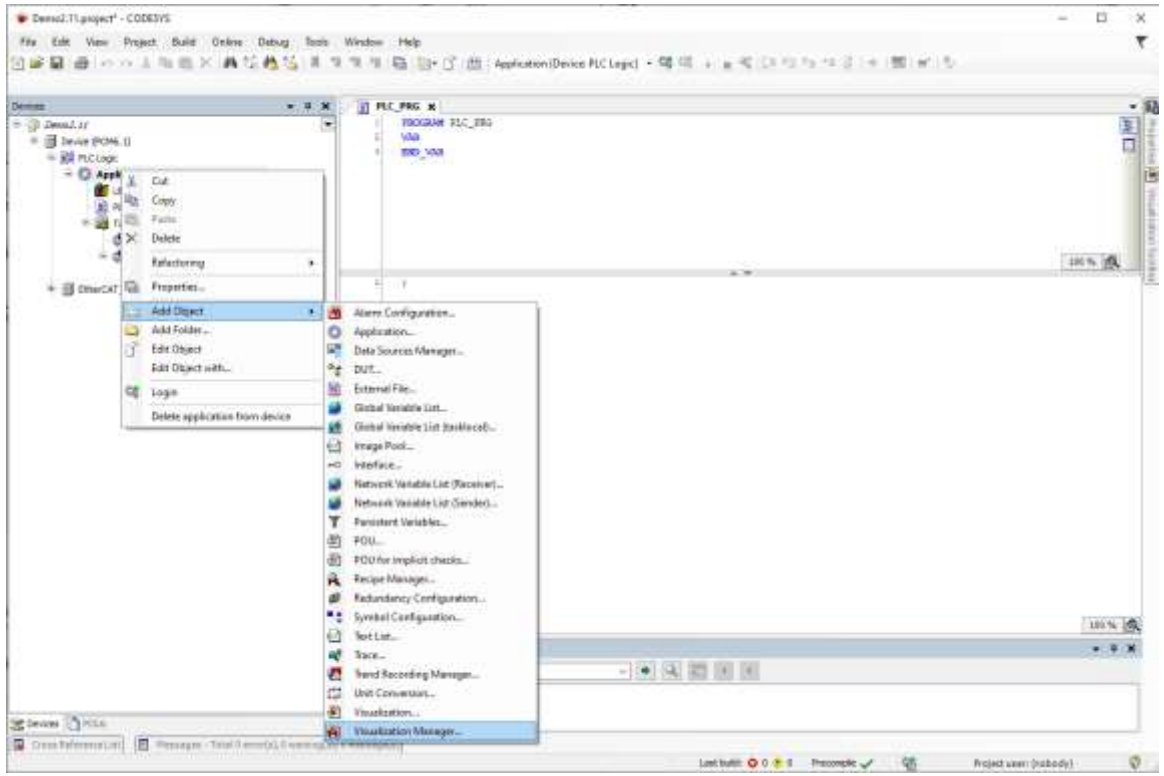

- **必须将程序生成启动应用，否则重启后程序丢失，persistent变量也不复存在。**

# IEC61131-3编程指南

## ➢ HMI创建及设置

CoDeSys IDE支持创建和编辑HMI，并支持通过网页对HMI的访问。在工程中添加HMI的方法如下：

- 新建工程，选择默认的Standard project
- 右击Application，并添加Visualization Manager，如图2.10.1



# IEC61131-3编程指南

## ➢ HMI创建及设置

- 默认选择直到VisualizationManger被加入到工程中

- **一个Application只能添加一个Visualization Manager，当前Application添加完成Visualization Manager后，该选项将从菜单中消失**

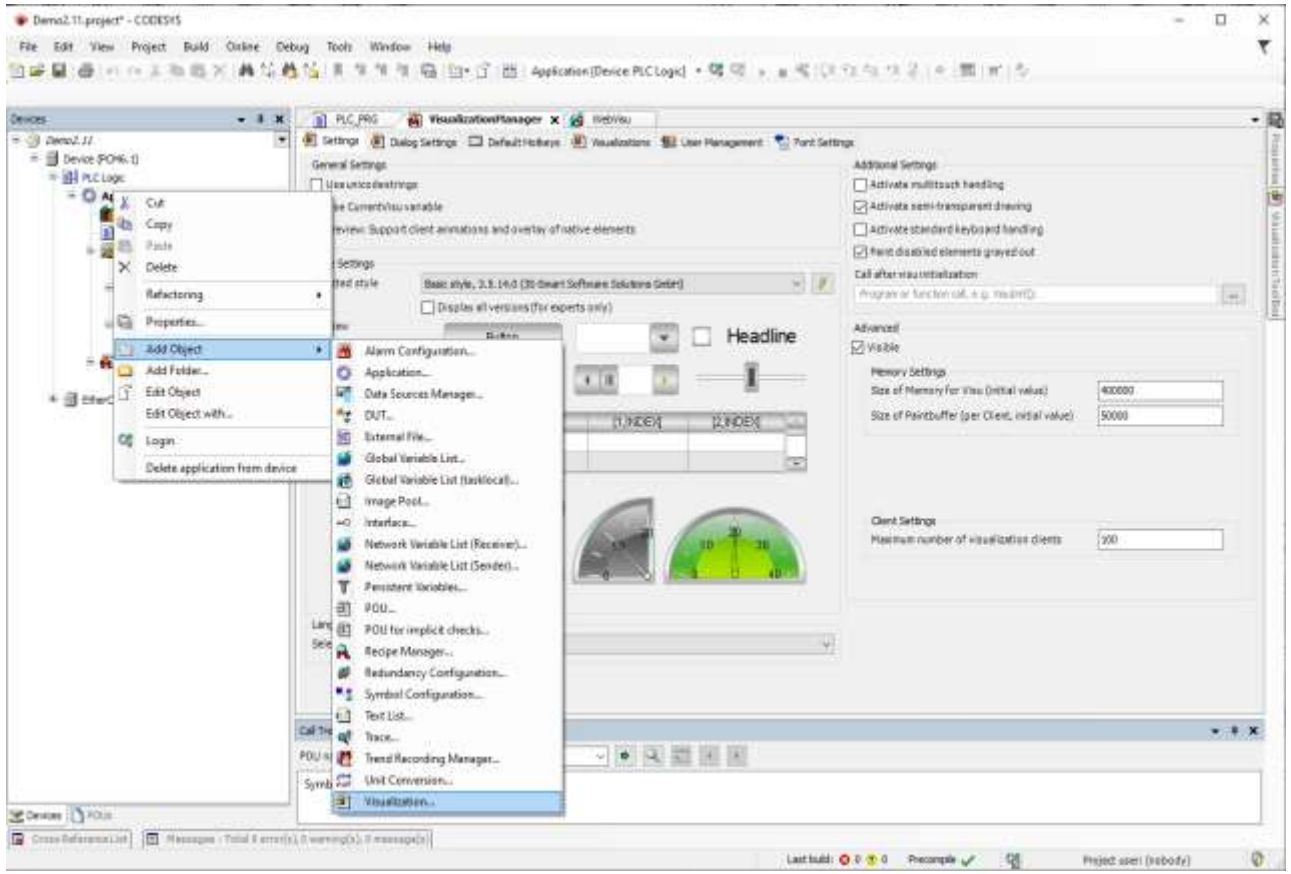
- 双击VisualizationManager，勾选Visible，并设置HMI存储区大小

- **HMI存储区大小通常采用默认设置即可。但当用户绘制的单一页面元素过多时，若存储区过小可能导致画面卡死或白屏，此时调整存储区大小时必要的。**
- **控制器作为服务器同时链接的用户数是有限的，同时链接大量用户也是导致画面卡死或白屏。因此限制同时访问的用户客户端数量也是必要的。**

# IEC61131-3编程指南

## ➢ HMI创建及设置

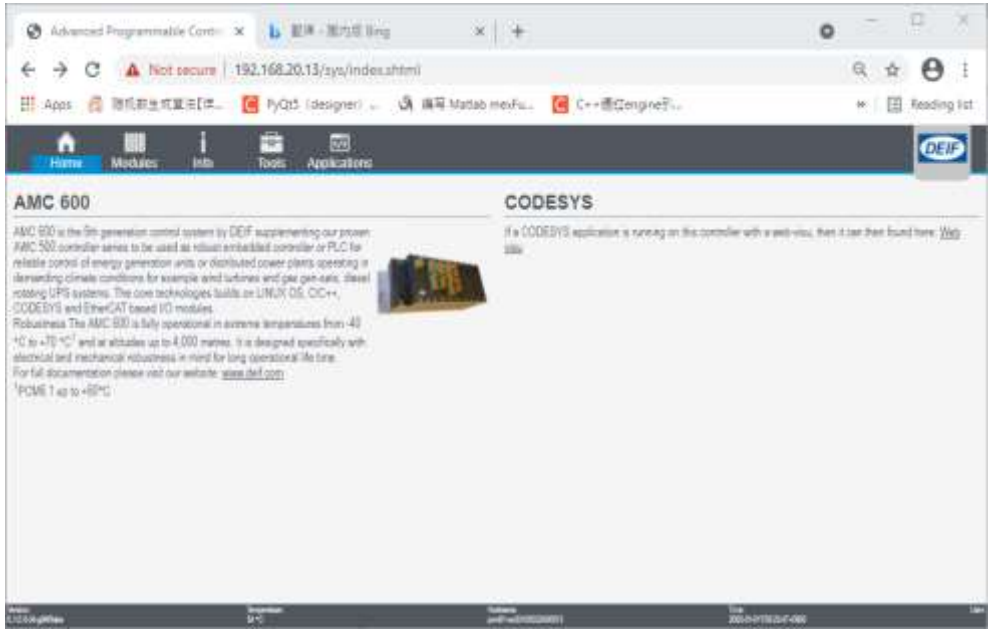
右击Application添加Visualization



# IEC61131-3编程指南

## ➢ HMI创建及设置

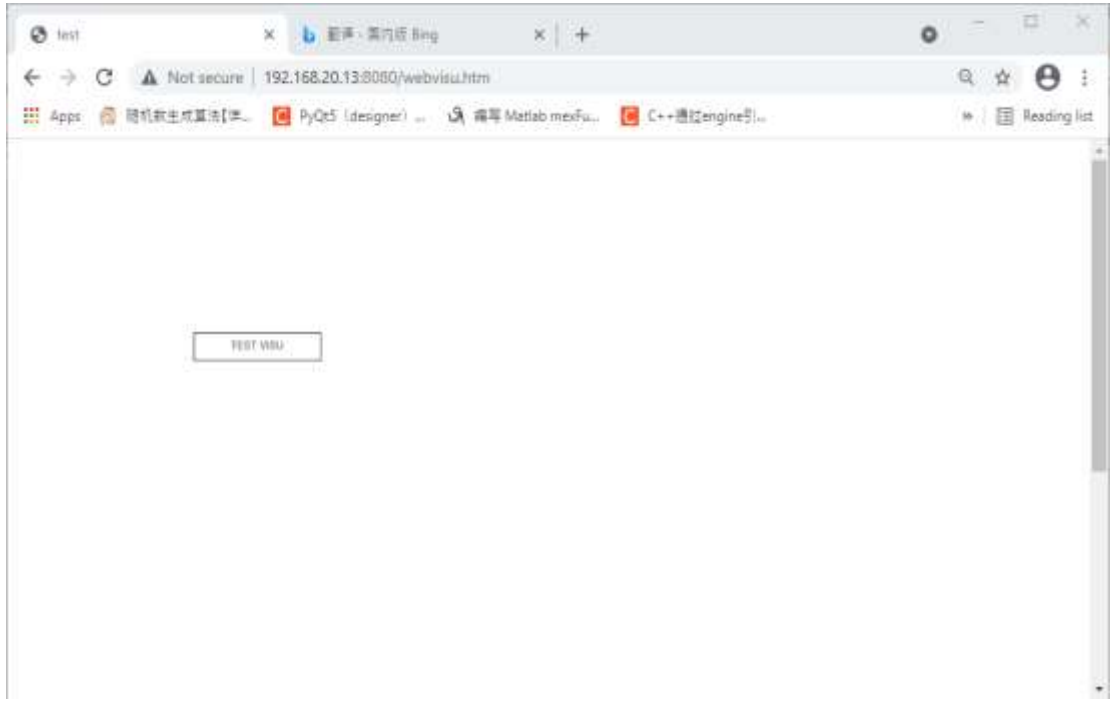
- 对其命名，并开始编辑界面
- 双击VisualizationManager下的WebVisu，可通过修改Start visualization来设置当用户登录到HMI界面时首先显示哪个界面。同时在WebVisu中还可对界面的刷新频率，界面缓存等参数进行设置。
- 编辑完成后将代码下载到控制器并运行，然后用户可通过访问控制器IP地址打开控制器网页



# IEC61131-3编程指南

- **HMI创建及设置**

- 在Home标签下点击右侧Web visu即可进入用户界面

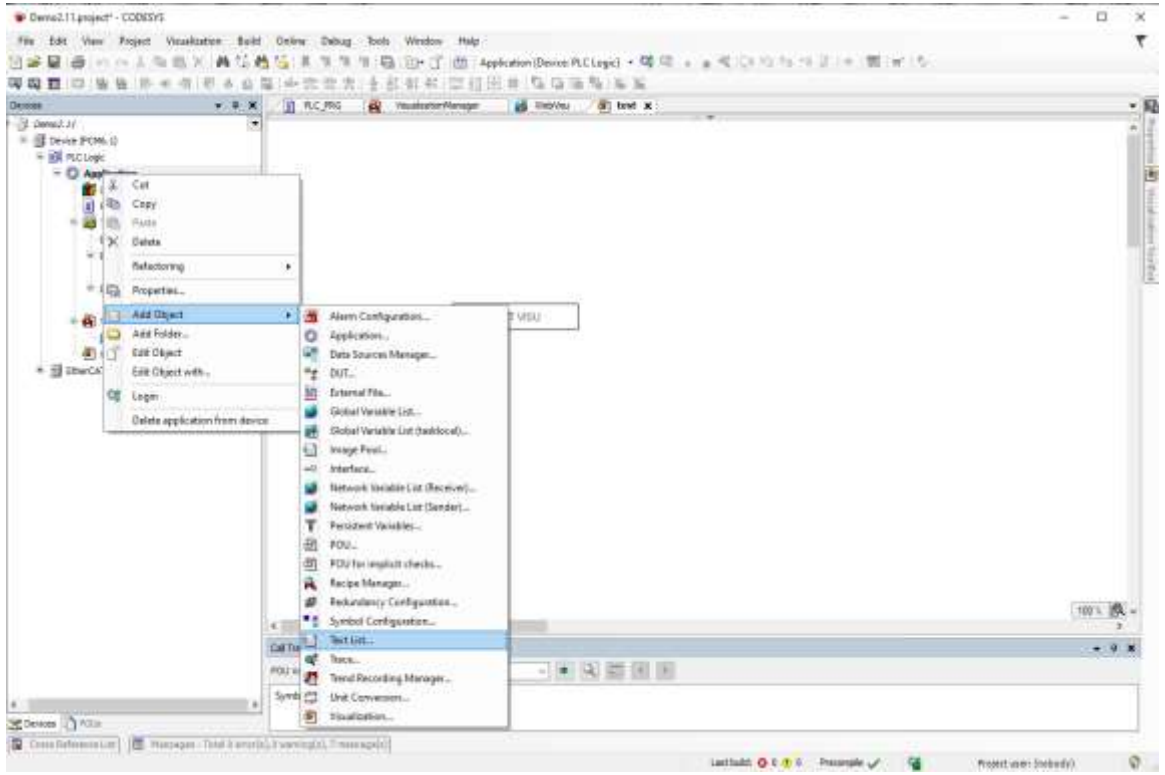



更多关于界面的设计，可以在CoDeSys IDE画面编辑中尝试使用CoDeSys提供的Visualization Toolbox中的多种组件实现。

# IEC61131-3编程指南

## ➢ HMI的交互语言

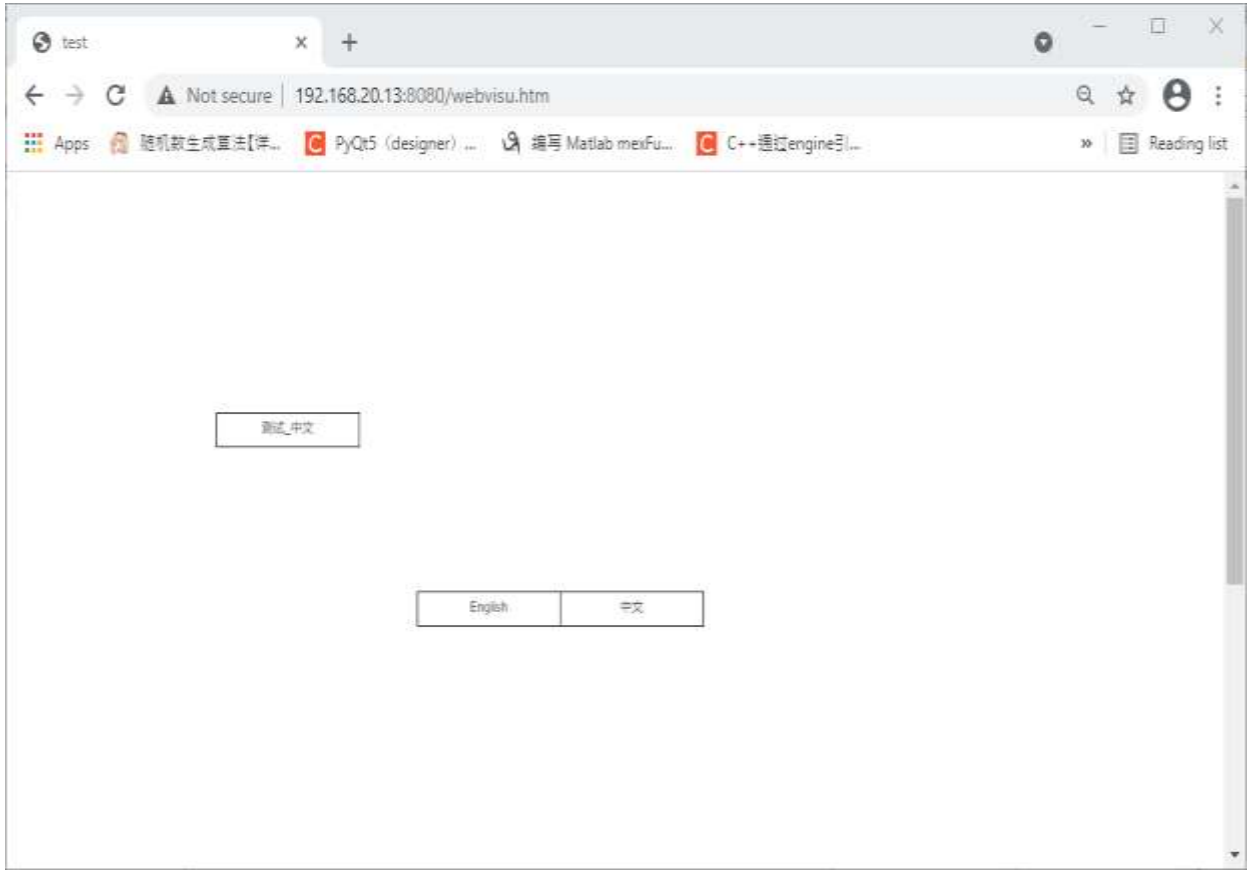
在项目开发过程中，添加到HMI中的所有静态文本都会在默认语言下自动添加到带有ID的全局文本列表中。图2.11展示了如何添加全局文本列表。



# IEC61131-3编程指南

## ➢ HMI的交互语言

其他语言可作为新列添加到此列表中，并可设置切换语言。



# IEC61131-3编程指南

## ➢ 创建启动应用及生成启动项目包

工程调试完毕正常运行后可通过菜单栏Online/Create Boot Application生成启动应用，之后控制器断电并重新上电原工程将自动运行。

为方便将工程部署到更多的设备当中，AWP100提供了制作启动项目包的工具：bootappBuilder

bootappBuilder_v9.0.0.1.zip

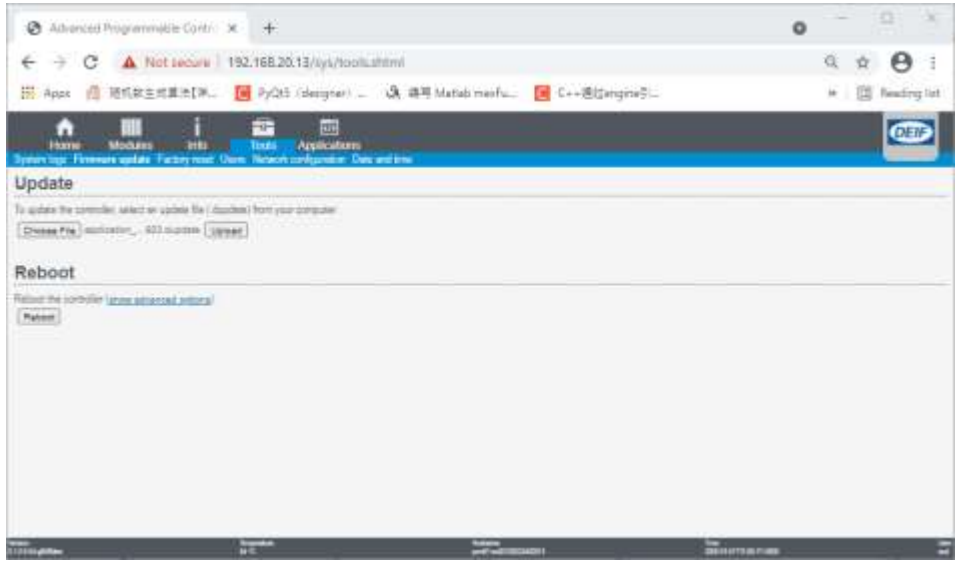
该工具是免安装的，解压后可直接使用，在config.txt中将IP地址修改为需要制作启动项目包的目标控制器，然后保存，双击运行批处理程序create_dupdate.bat，待运行完成将在同级文件夹内生成application_xxx_xxx.dupdate（xxx为生成的时间戳），该文件即为启动项目包

# IEC61131-3编程指南

## ➢ 部署启动项目包

AWP100提供了两种启动项目包的部署方法，其一是通过控制器网页更新，方法如下：

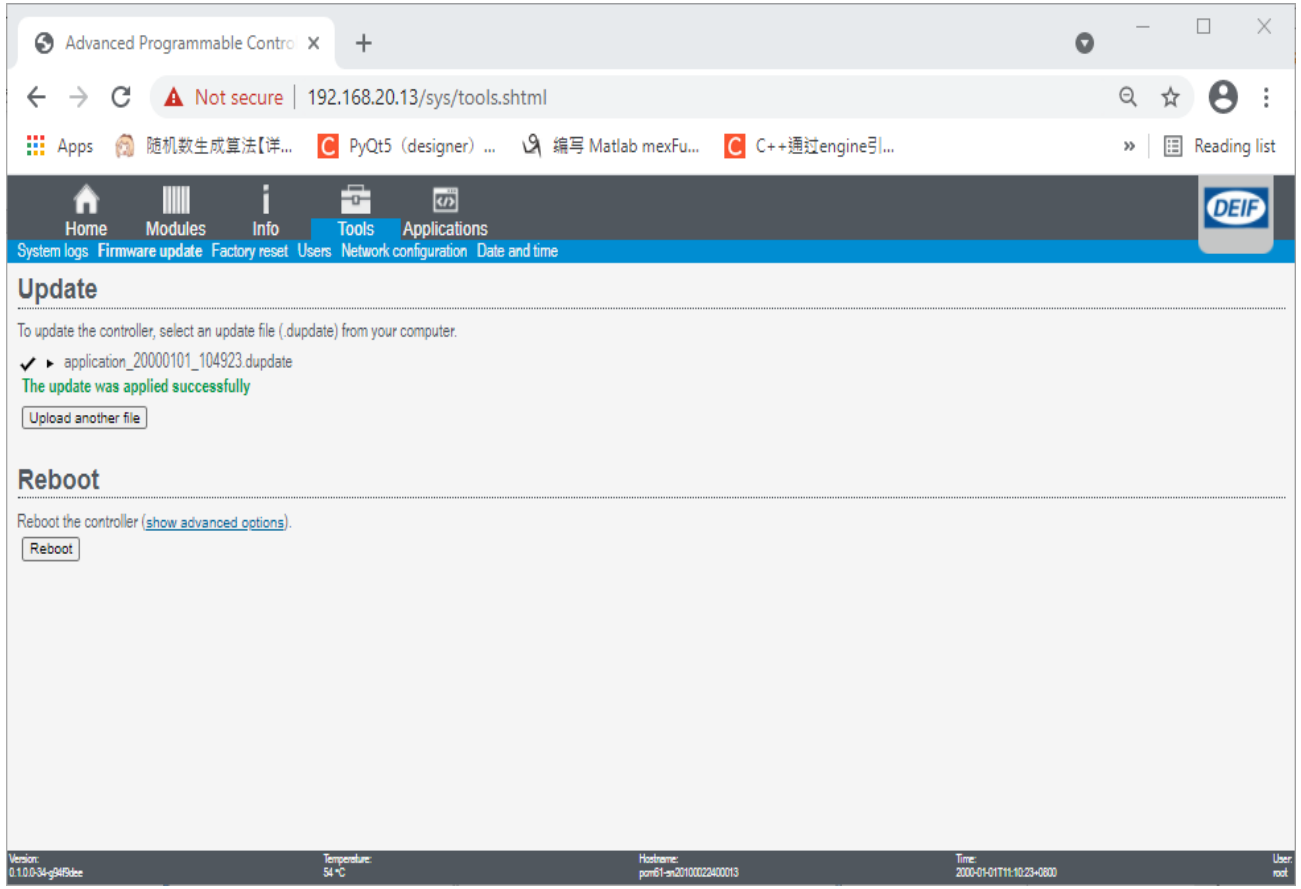
- 通过输入IP进入控制器网页界面，并转到Tools/Firmware update
- 在Update项下单击Choose File按钮并找到启动项目包路径，选中启动项目包
- 点击Upload



# IEC61131-3编程指南

## ➢ 部署启动项目包

- 更新完成后将显示绿色更新成功提示。



# IEC61131-3编程指南

## ➢ 部署启动项目包

点击Reboot

- **若需要输入用户名密码，请输入用户名：root，密码：deif7800，并点击确认**

输入成功后点击Reboot重启控制器即可

- **虽然启动项目包刷入完成后控制器能够正常运行，但是仍然建议重启一次以确保所有部署项完全生效**

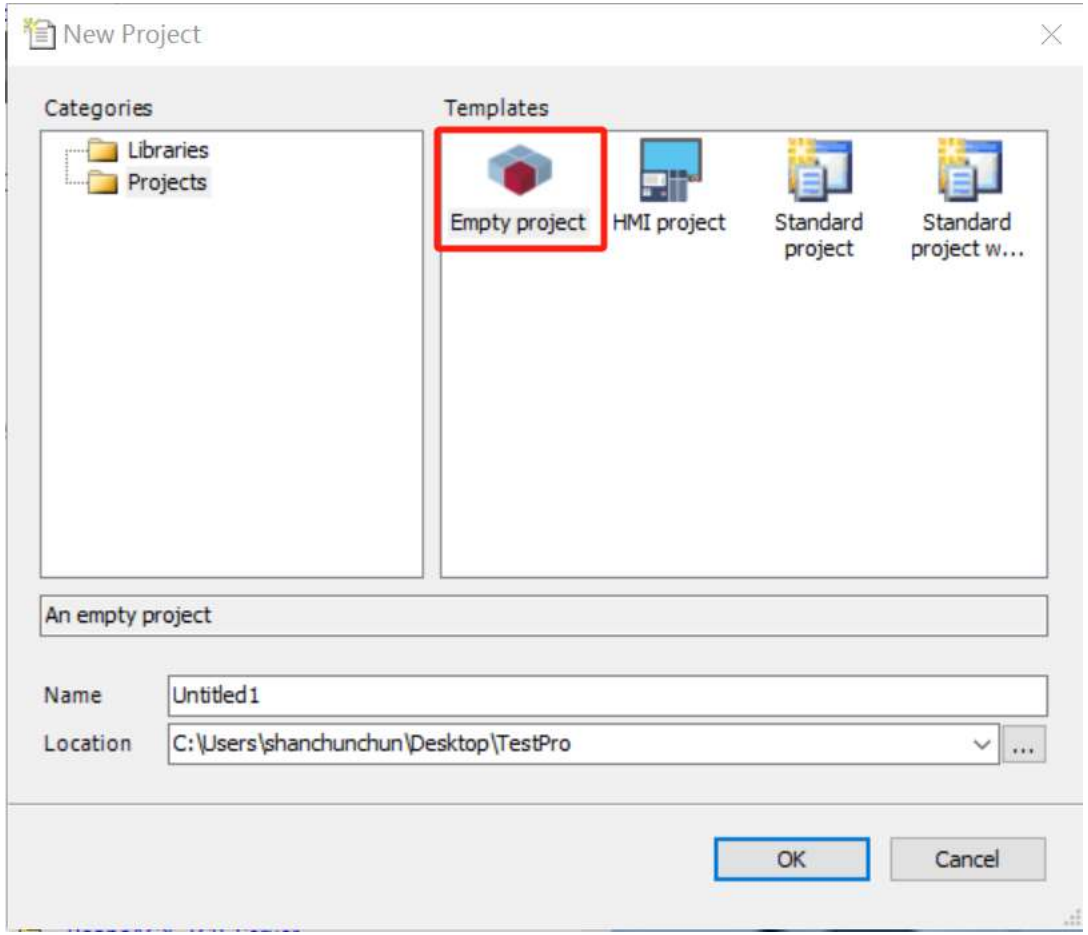
# IEC61131-3编程指南

## ➢ 多应用编程及监控

AWP100支持多Application同时运行在系统中，并且可以实现Application间的相互监控，当某一个Application出现异常时可由其他Application实现对异常Application的重启。

# 新建第一个CoDeSys工程

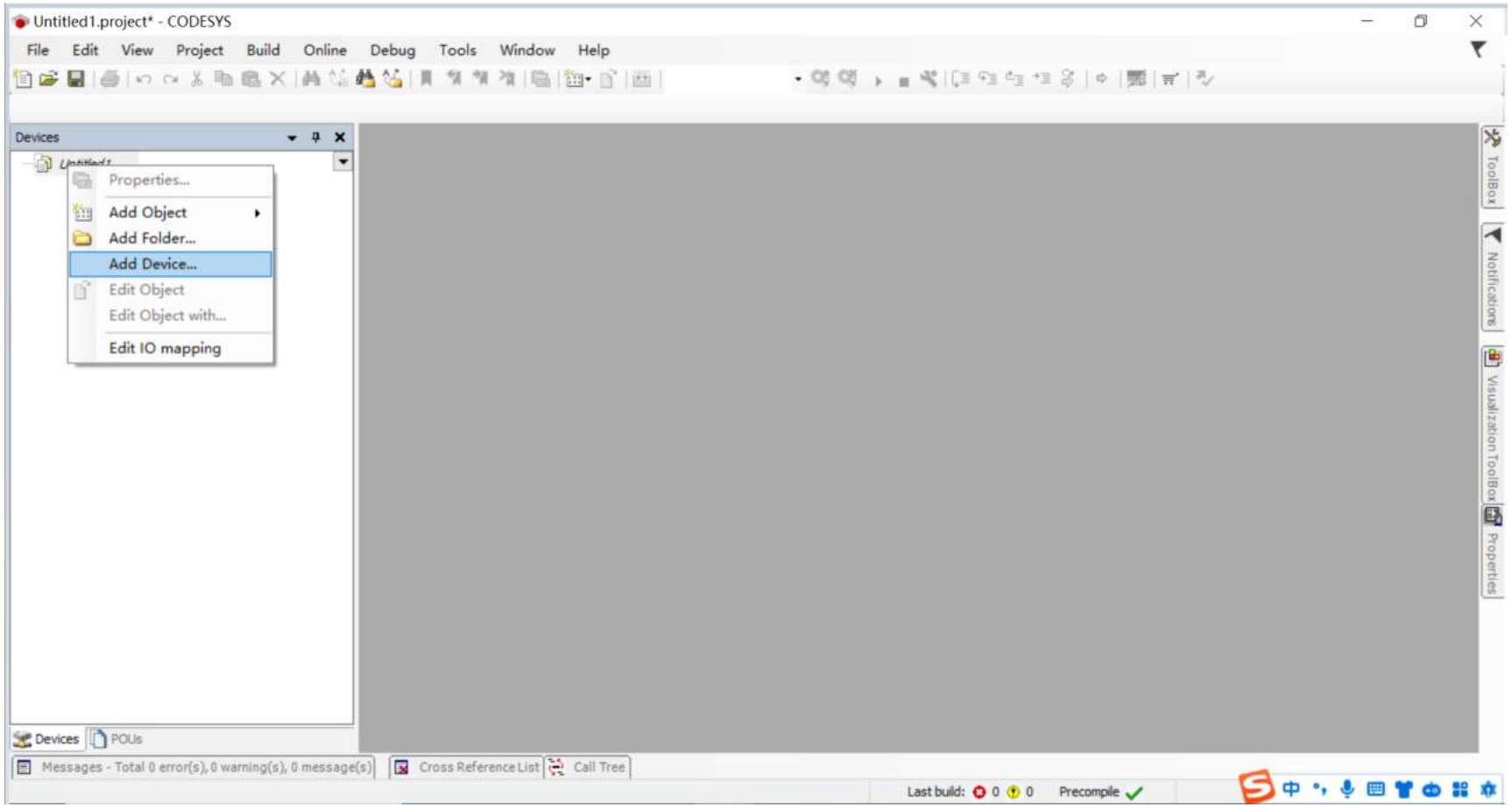
➢ 新建一个CoDeSys工程




也可以选择“Standard project”，直接开始编程

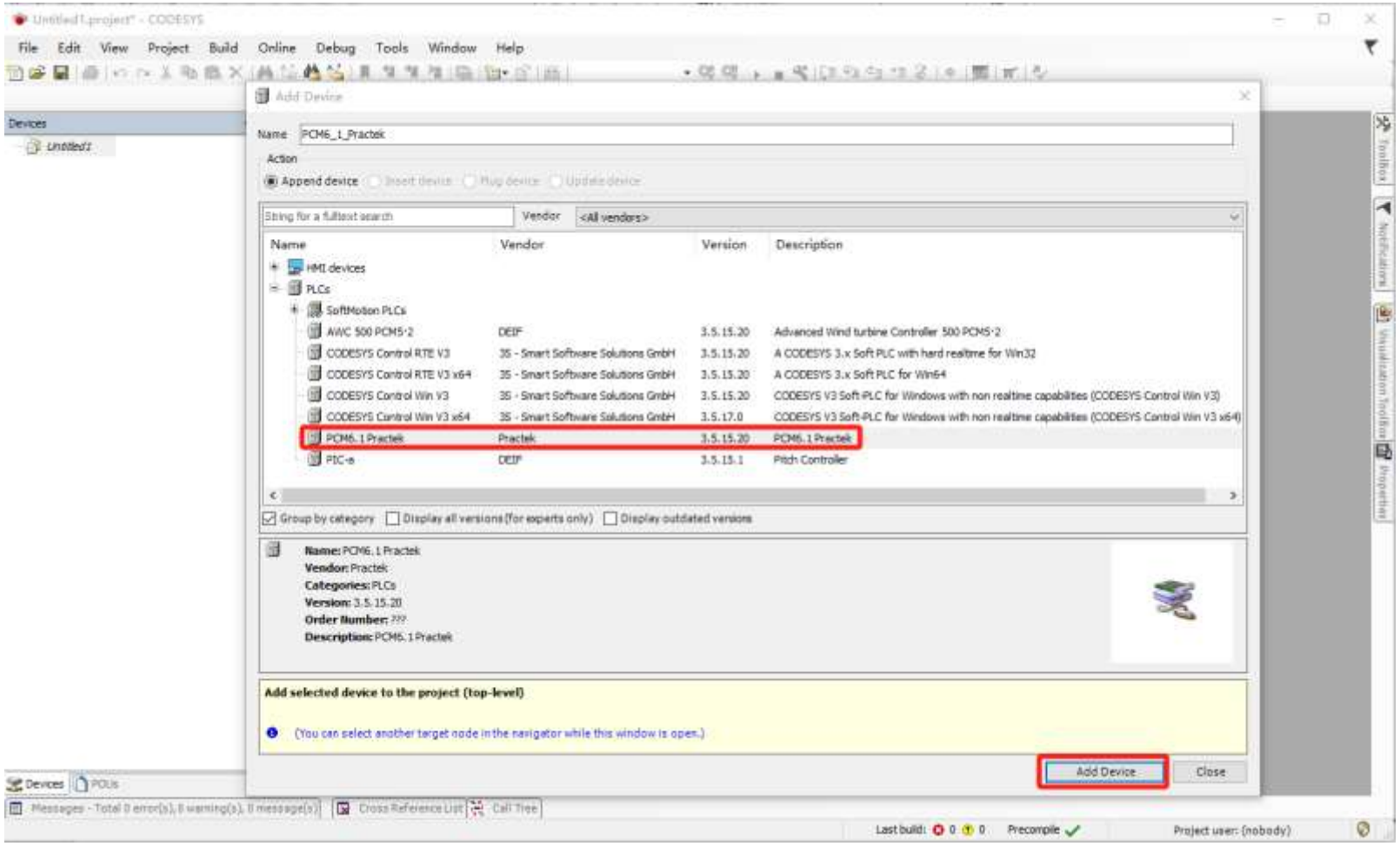
# 新建第一个CoDeSys工程

➢ 添加CT65设备



# 新建第一个CoDeSys工程

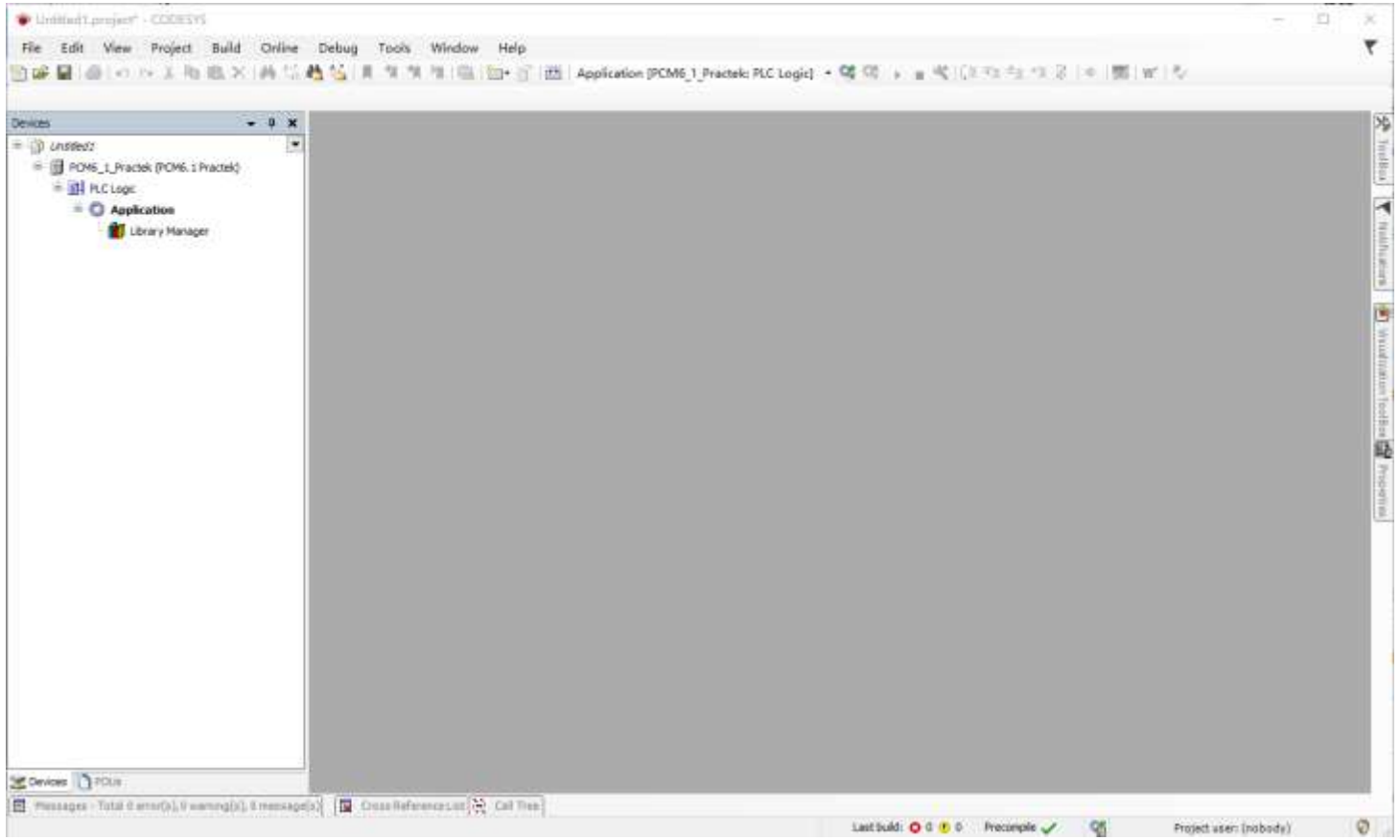
➢ 添加CT65设备



# 新建第一个CoDeSys工程

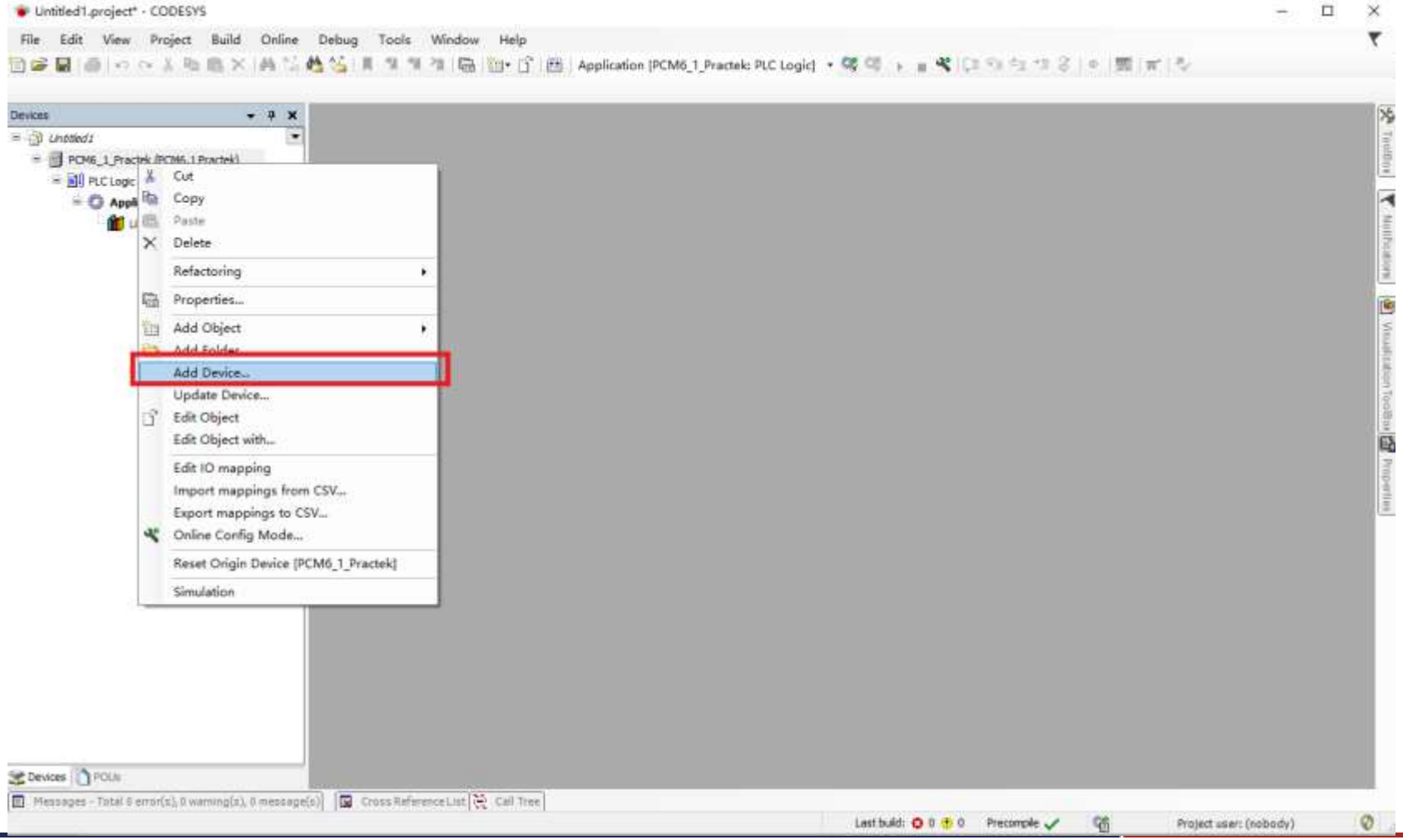
## ➢ 添加CT65设备

添加完工程如下图所示



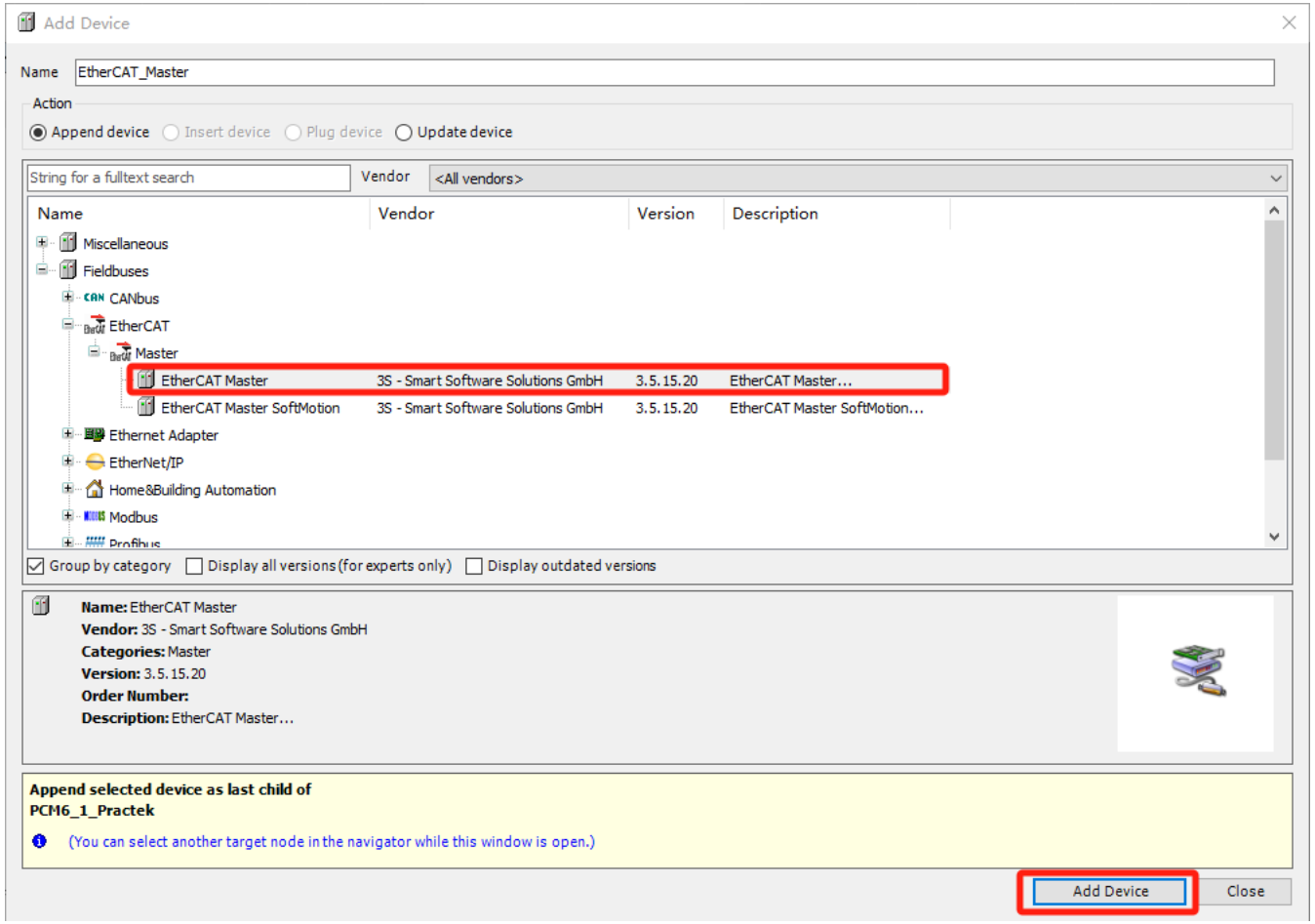
# 新建第一个CoDeSys工程

➢ 添加EtherCAT Master



# 新建第一个CoDeSys工程

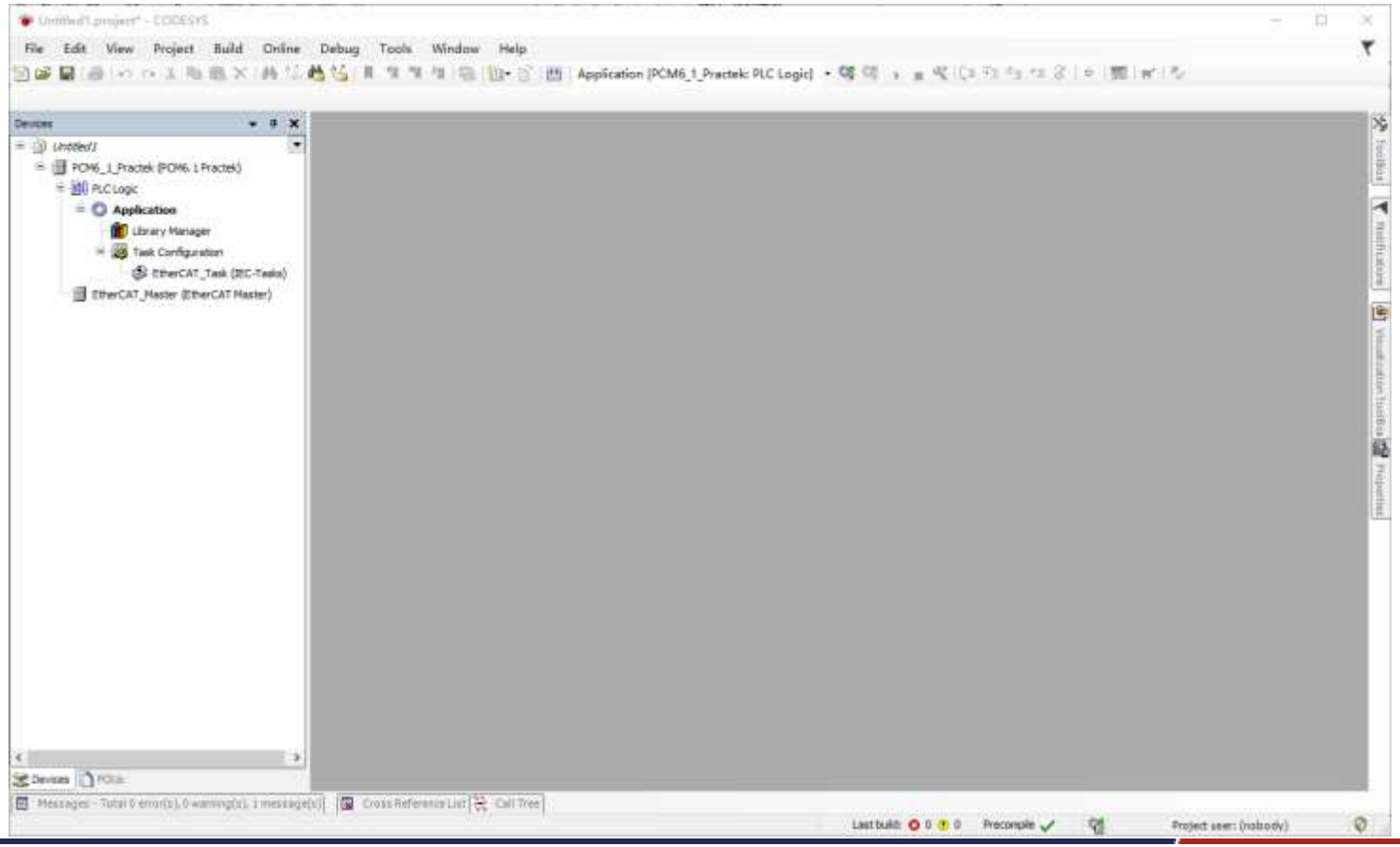
➢ 添加EtherCAT Master



# 新建第一个CoDeSys工程

## ➢ 添加EtherCAT Master

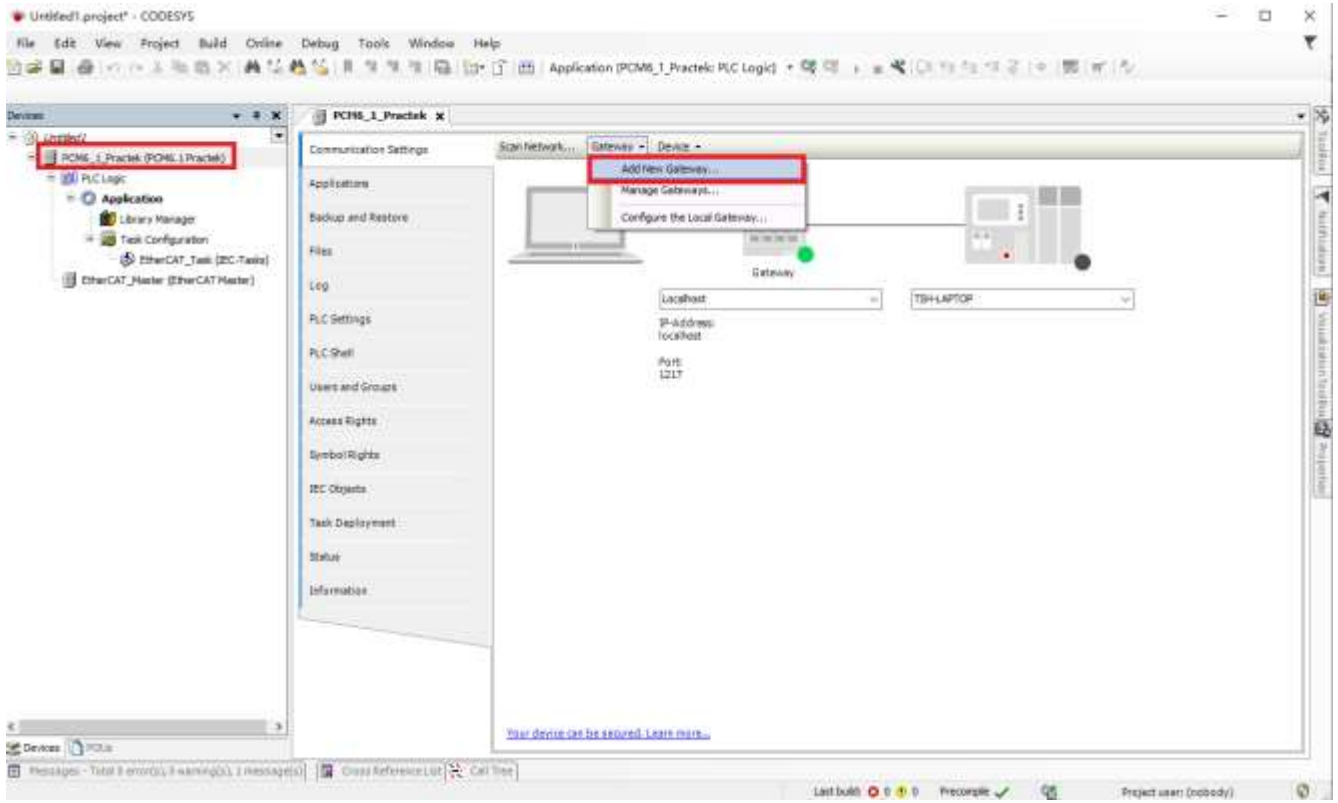
添加完EtherCAT Master，工程如下图所示



# 新建第一个CoDeSys工程

## ➢ 和CT65建立连接

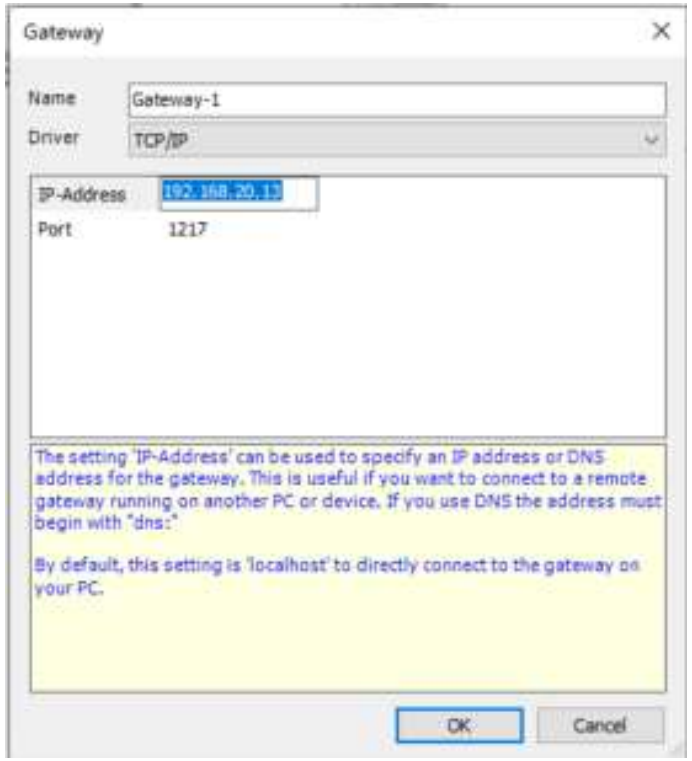
双击CT65设备，可以看到一个已经存在的Gateway，默认IP地址：Localhost；你也可以新建一个新的Gateway，点击Gateway -> Add new gateway。



# 新建第一个CoDeSys工程

## ➢ 和CT65建立连接

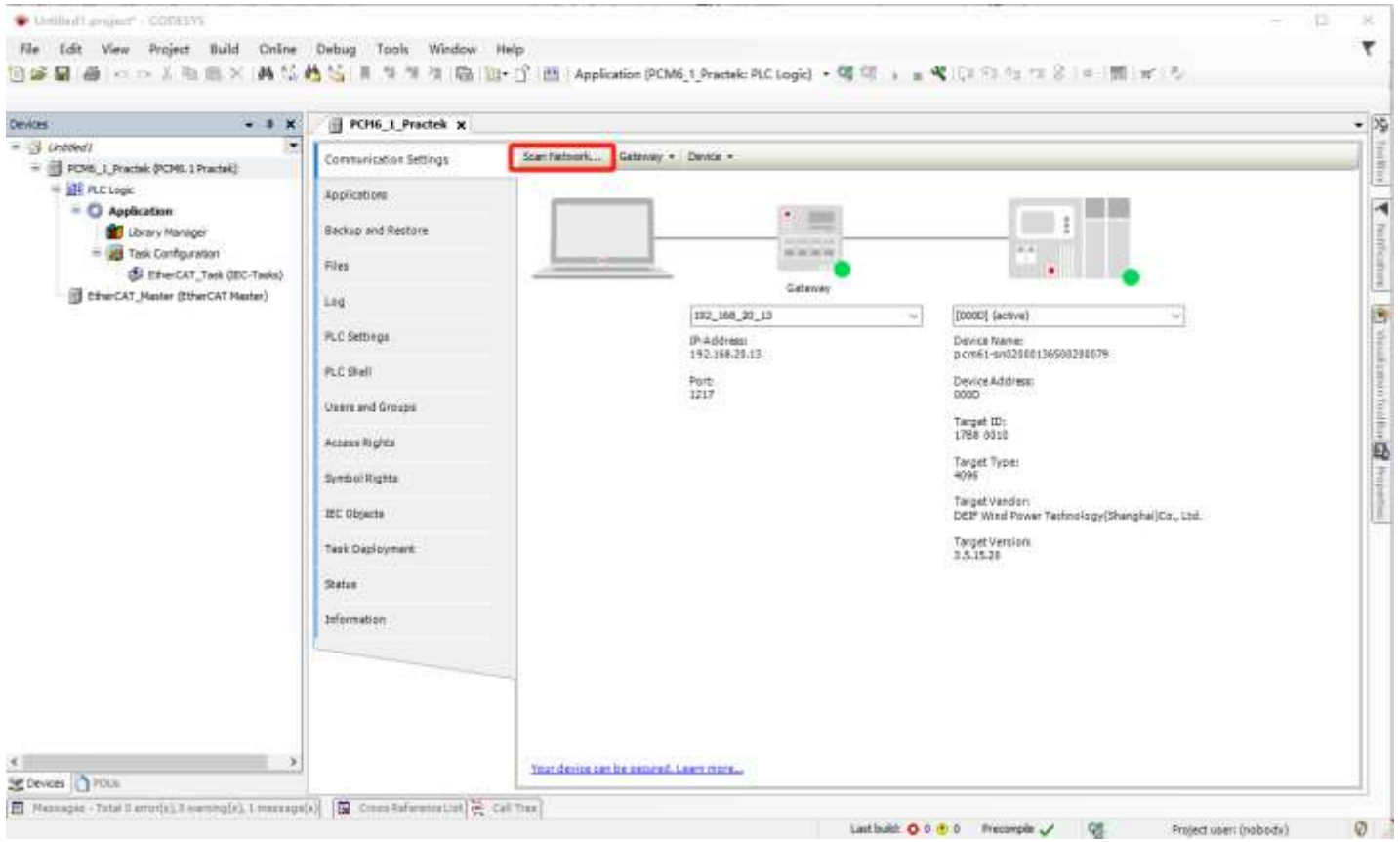
输入控制器的IP，例如：192.168.20.13。



# 新建第一个CoDeSys工程

## ➢ 和CT65建立连接

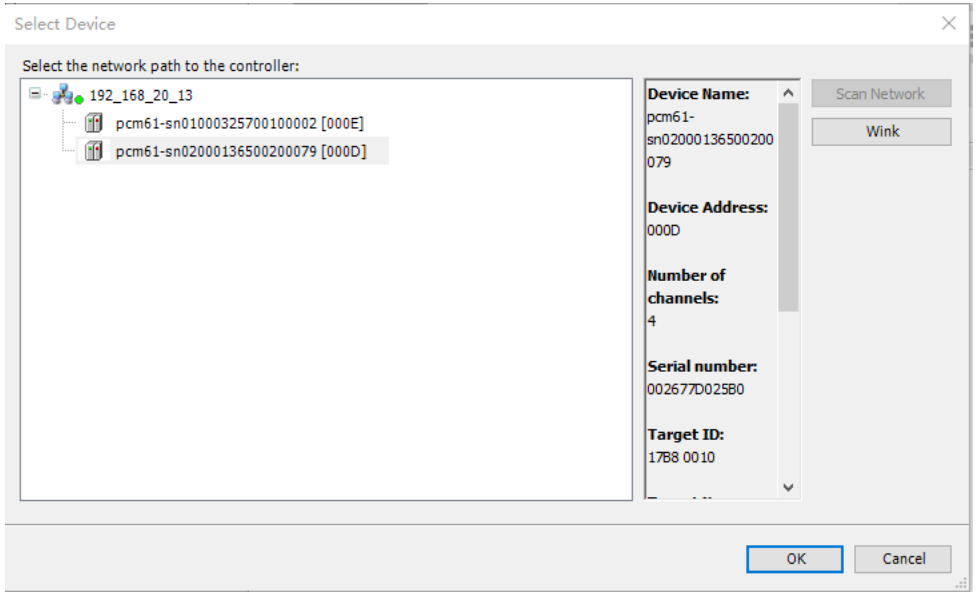
新建完新的Gateway后，点击Scan network，扫描CT65网络。



# 新建第一个CoDeSys工程

## ➢ 和CT65建立连接

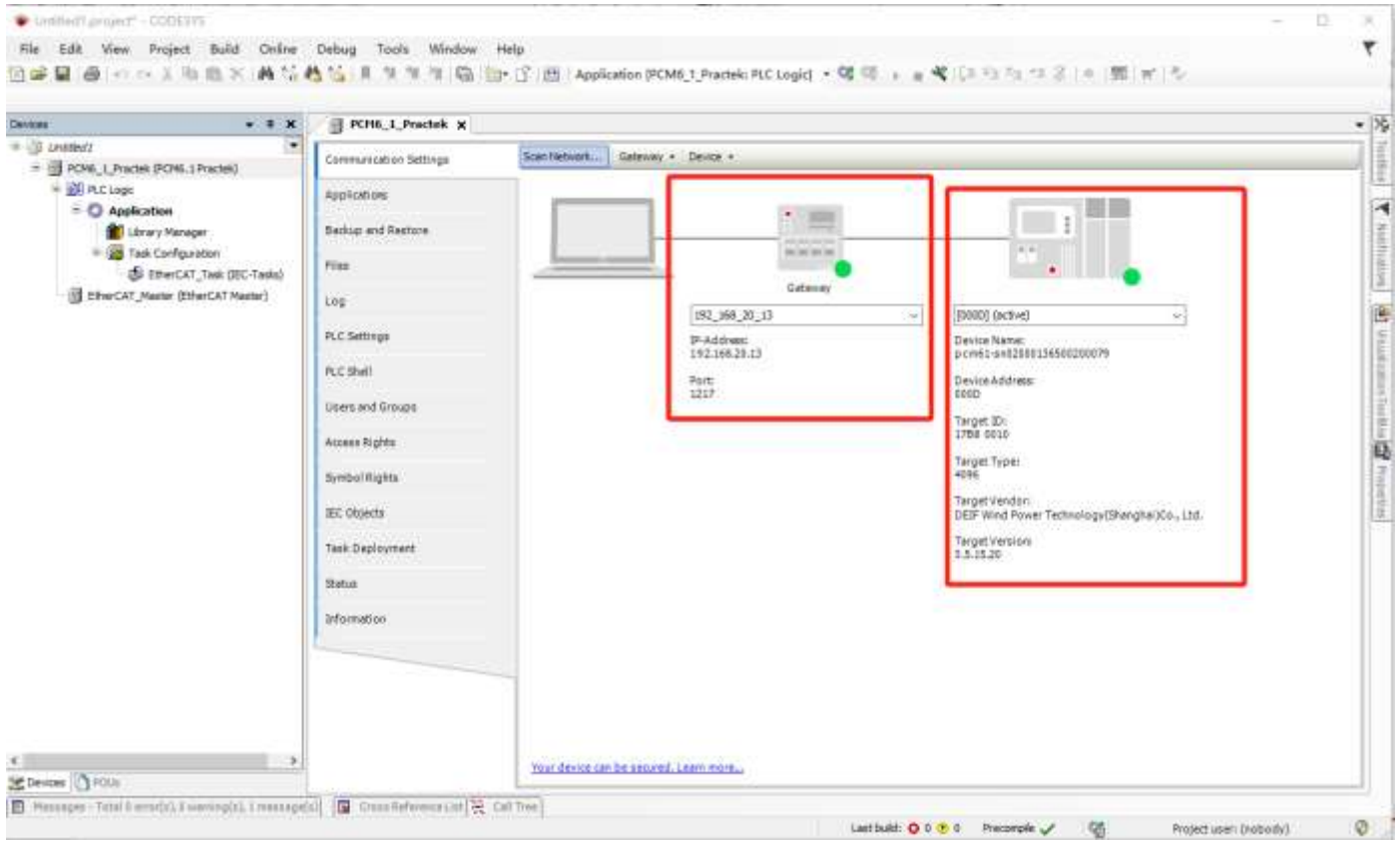
选择对应的设备，点击OK。



# 新建第一个CoDeSys工程

## ➢ 和CT65建立连接

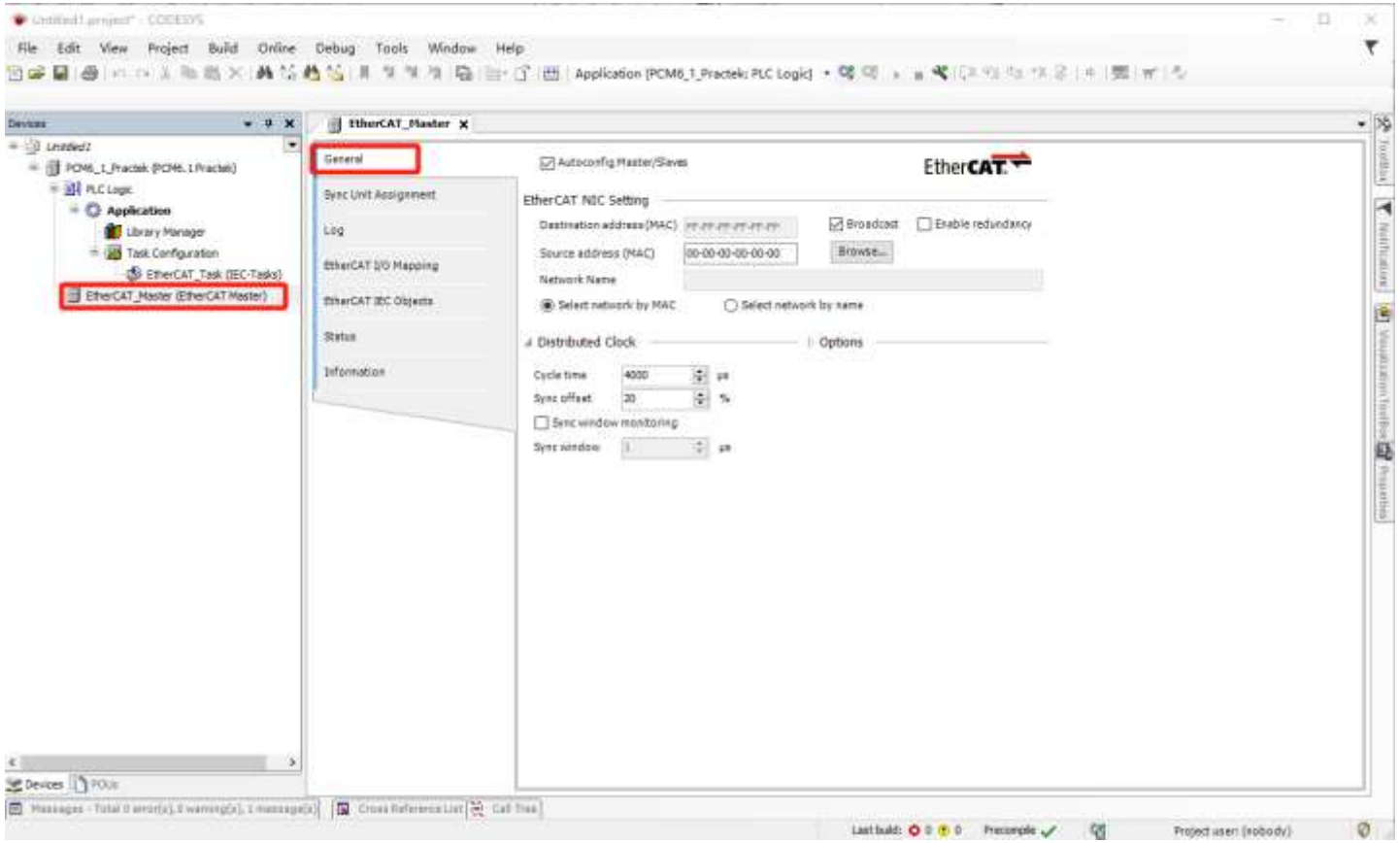
连接完成后，工程界面如下。



# 新建第一个CoDeSys工程

## ➢ 设置EtherCAT Master

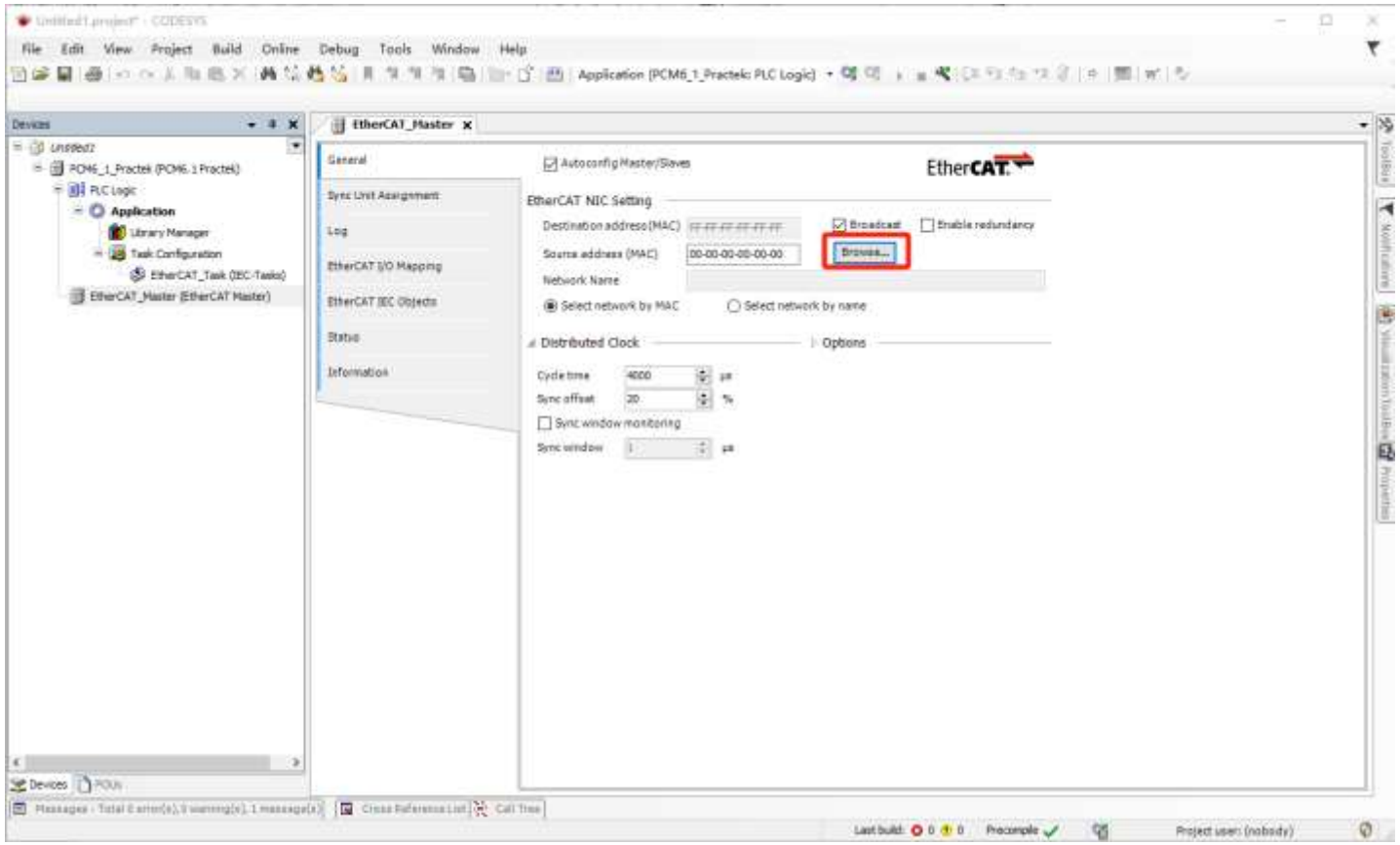
双击EtherCAT_Master。



# 新建第一个CoDeSys工程

➢ **设置EtherCAT Master**

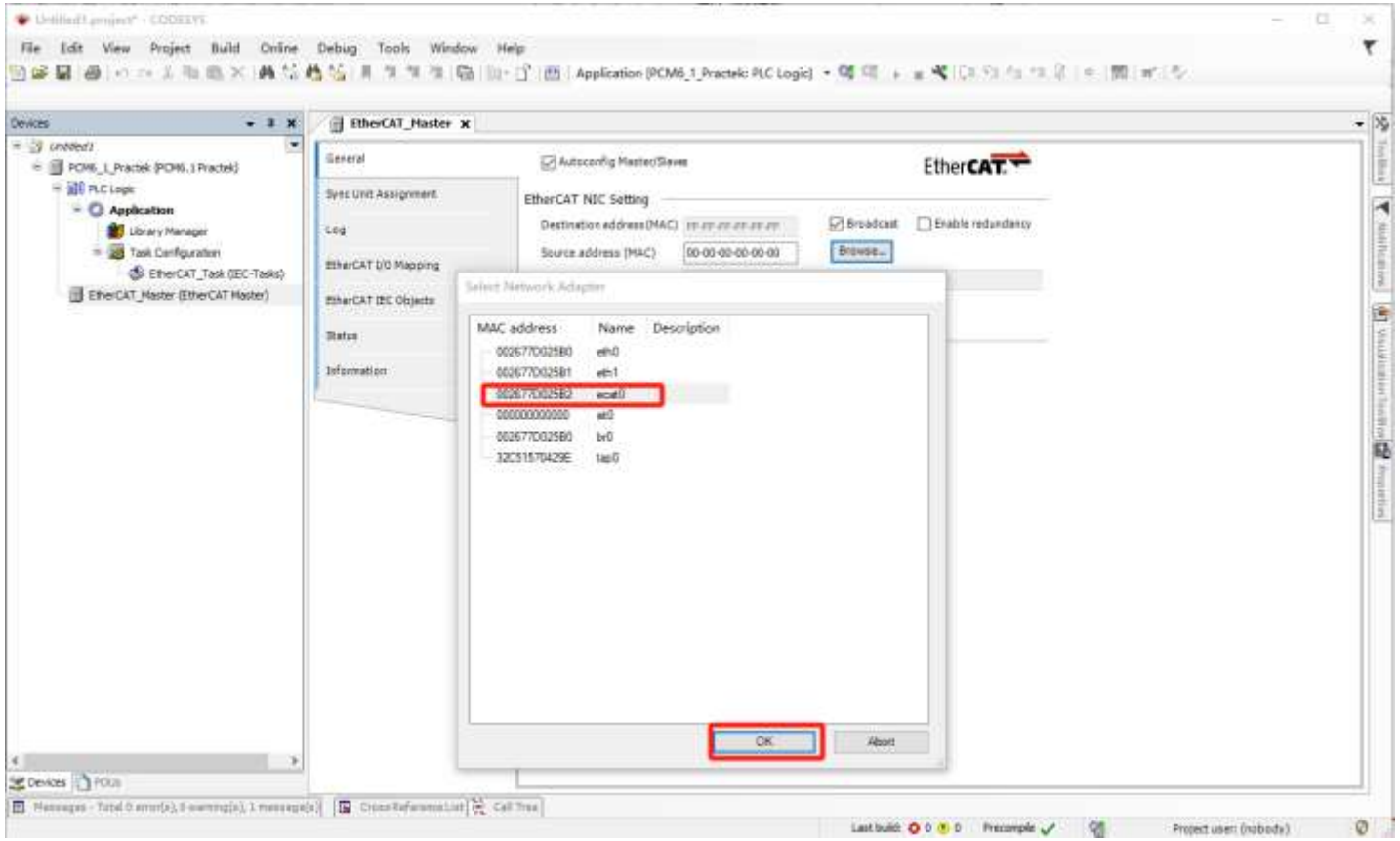
点击Browse…



# 新建第一个CoDeSys工程

➢ **设置EtherCAT Master**

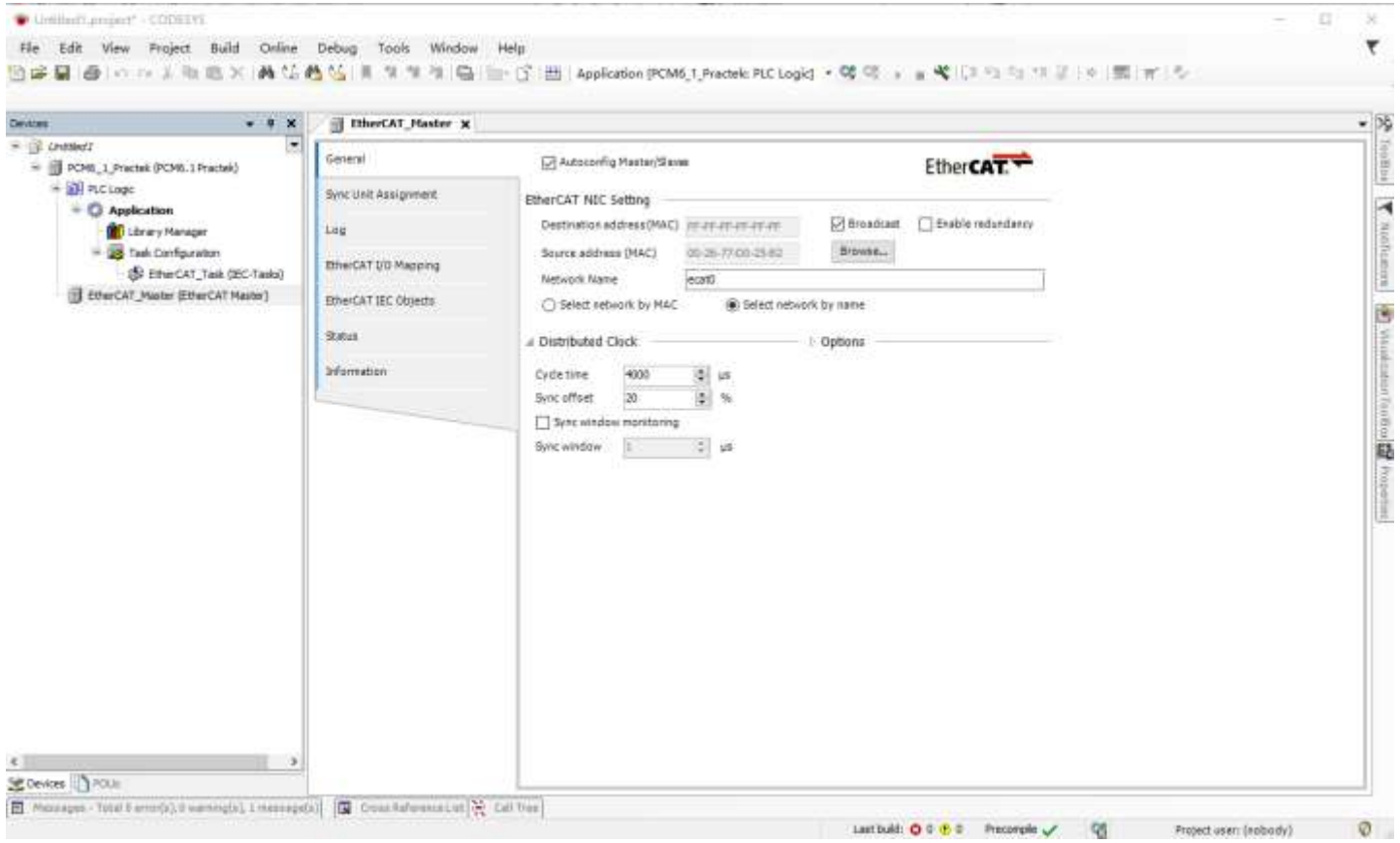
选择ecat0,点击OK。



# 新建第一个CoDeSys工程

## ➢ 设置EtherCAT Master

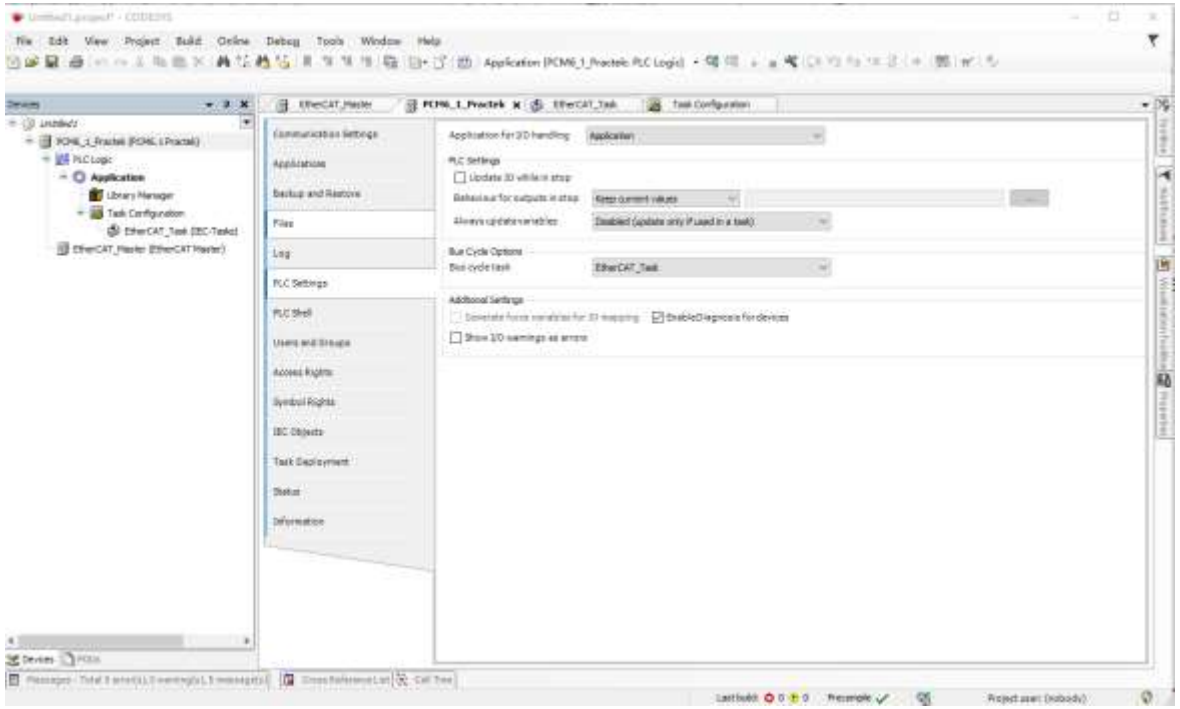
选择Select Network by Name,工程如下图。



# 新建第一个CoDeSys工程

## ➢ PLC Settings

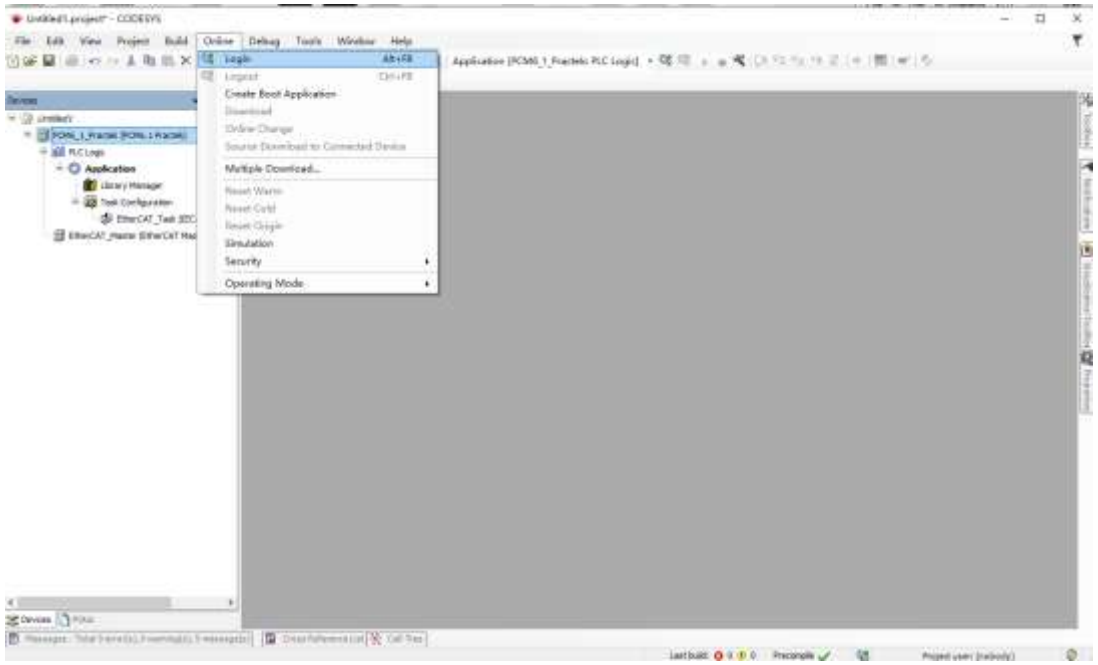
点击设备PCM61_1_Practek->PLC Settings，我们推荐把Bus cycle task 设置为EtherCAT_Master，同时用户可以根据需求设置Update IO while in stop、Behavior of the outputs at stop、Always update variables。

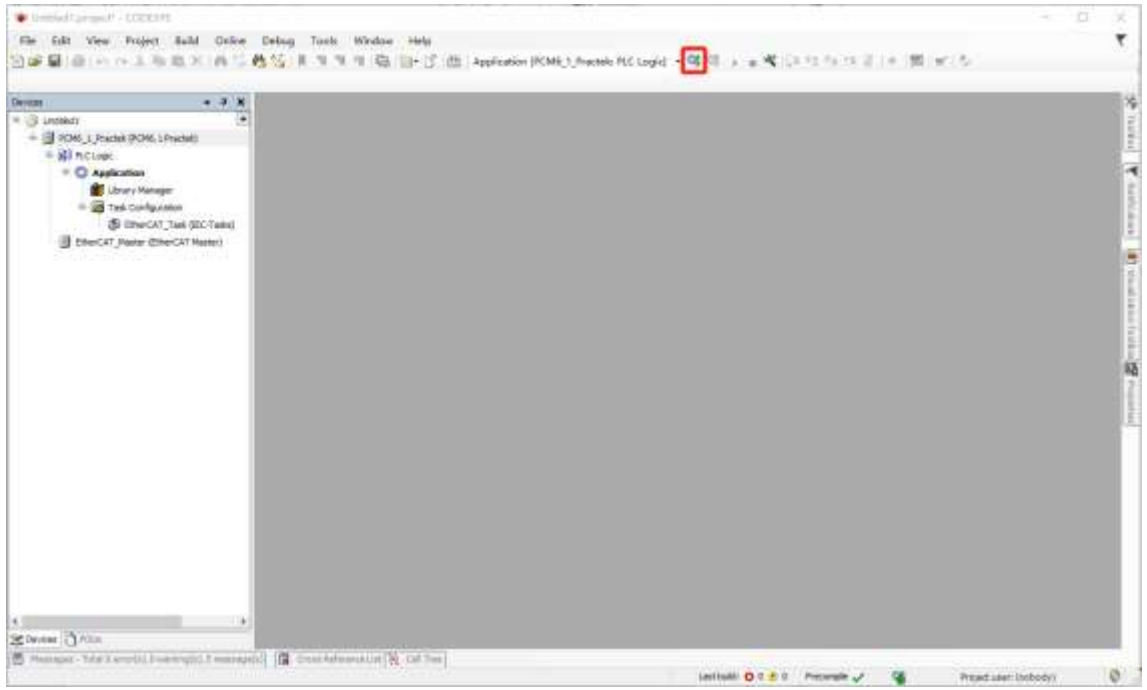


# 新建第一个CoDeSys工程

## ➢ 扫描模块

首先login

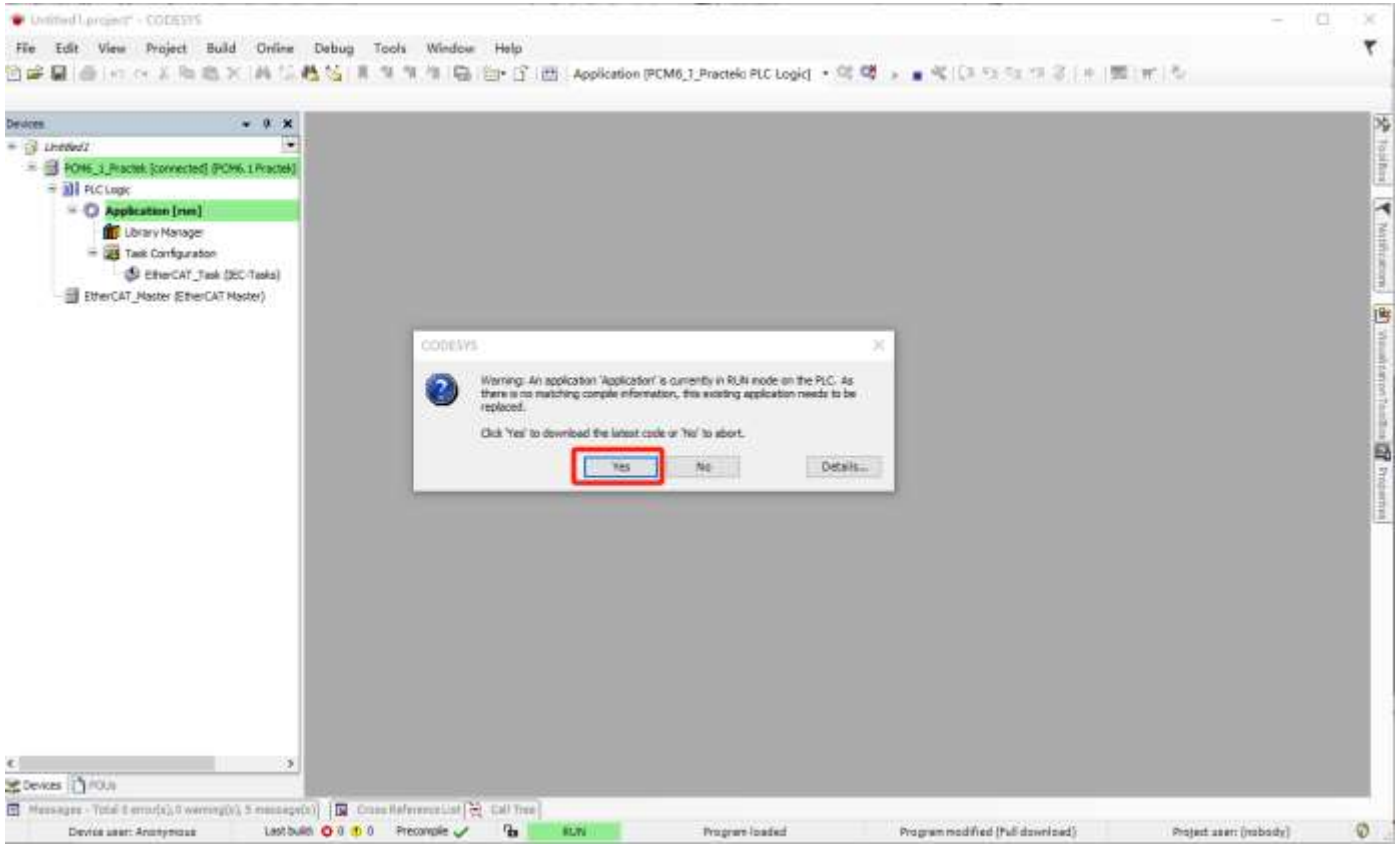


# 新建第一个CoDeSys工程

## ➢ 扫描模块

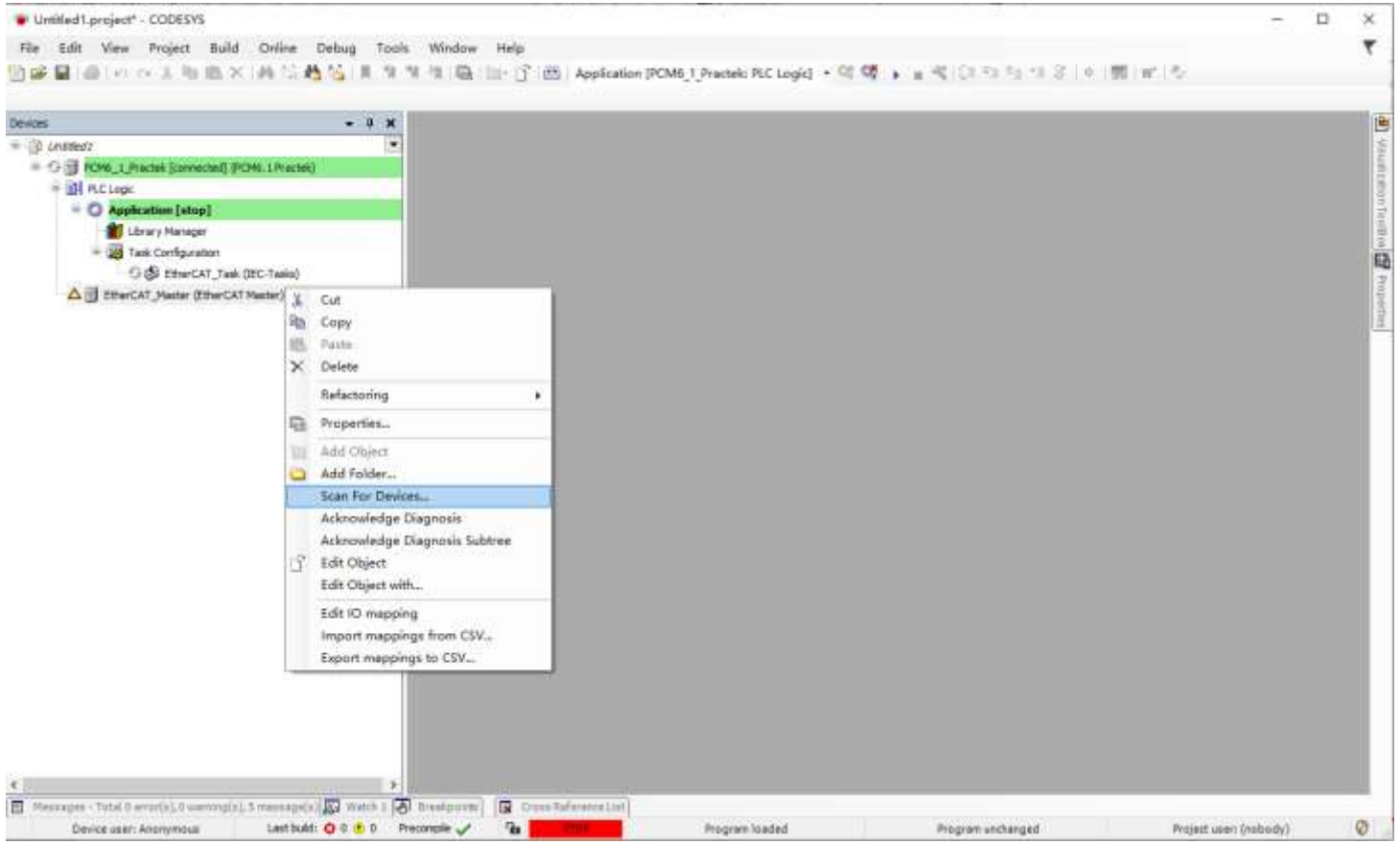
点击Yes



# 新建第一个CoDeSys工程

## ➢ 扫描模块

右击EtherCAT_Master，选择Scan For Devices…



# 新建第一个CoDeSys工程

- **扫描模块**

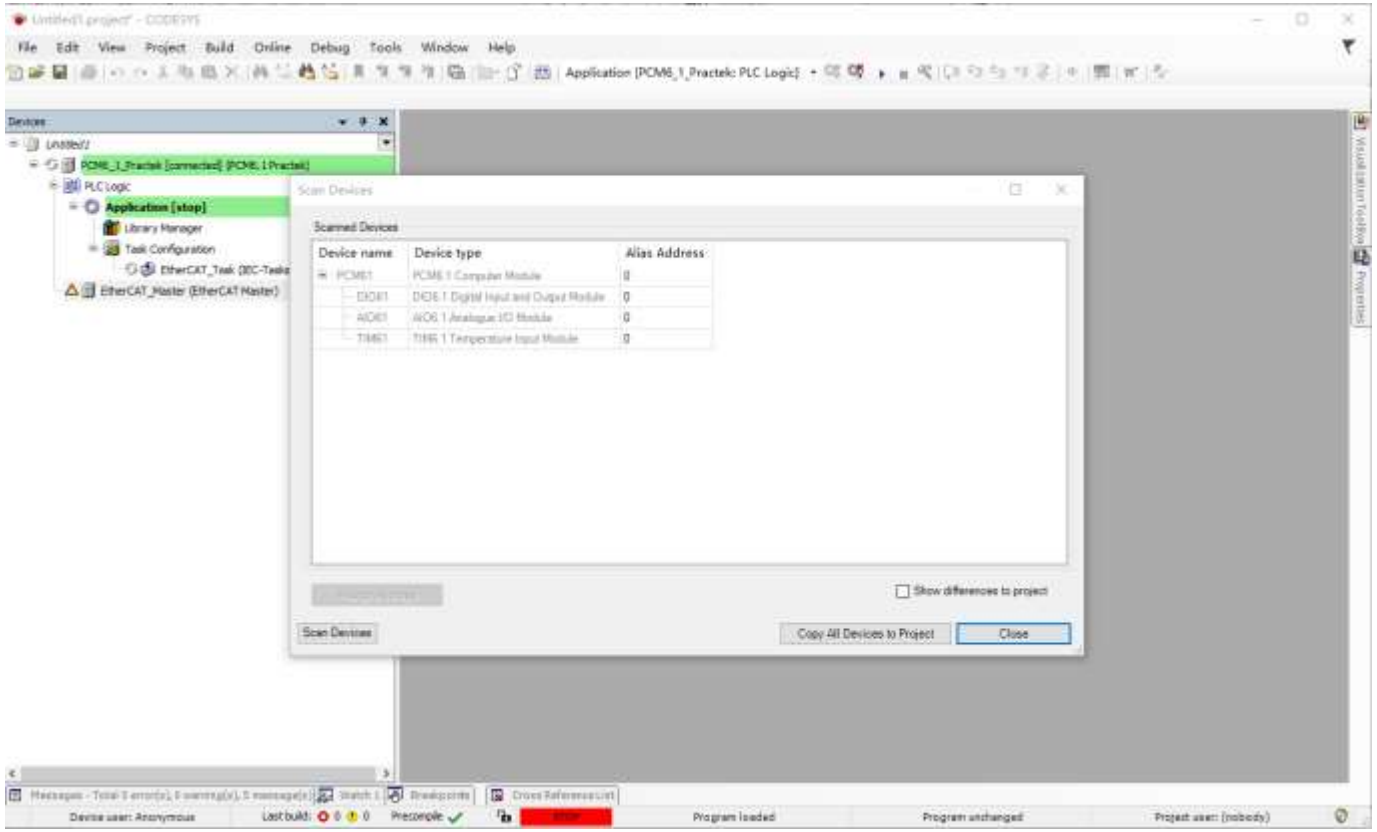
  可能无需登录CT65就可以扫描设备，它要求EtherCAT Master在CT65上运行就可以扫描。

  如果扫描失败，请执行Login和Reset cold，然后重新扫描。

# 新建第一个CoDeSys工程

## ➢ 扫描模块

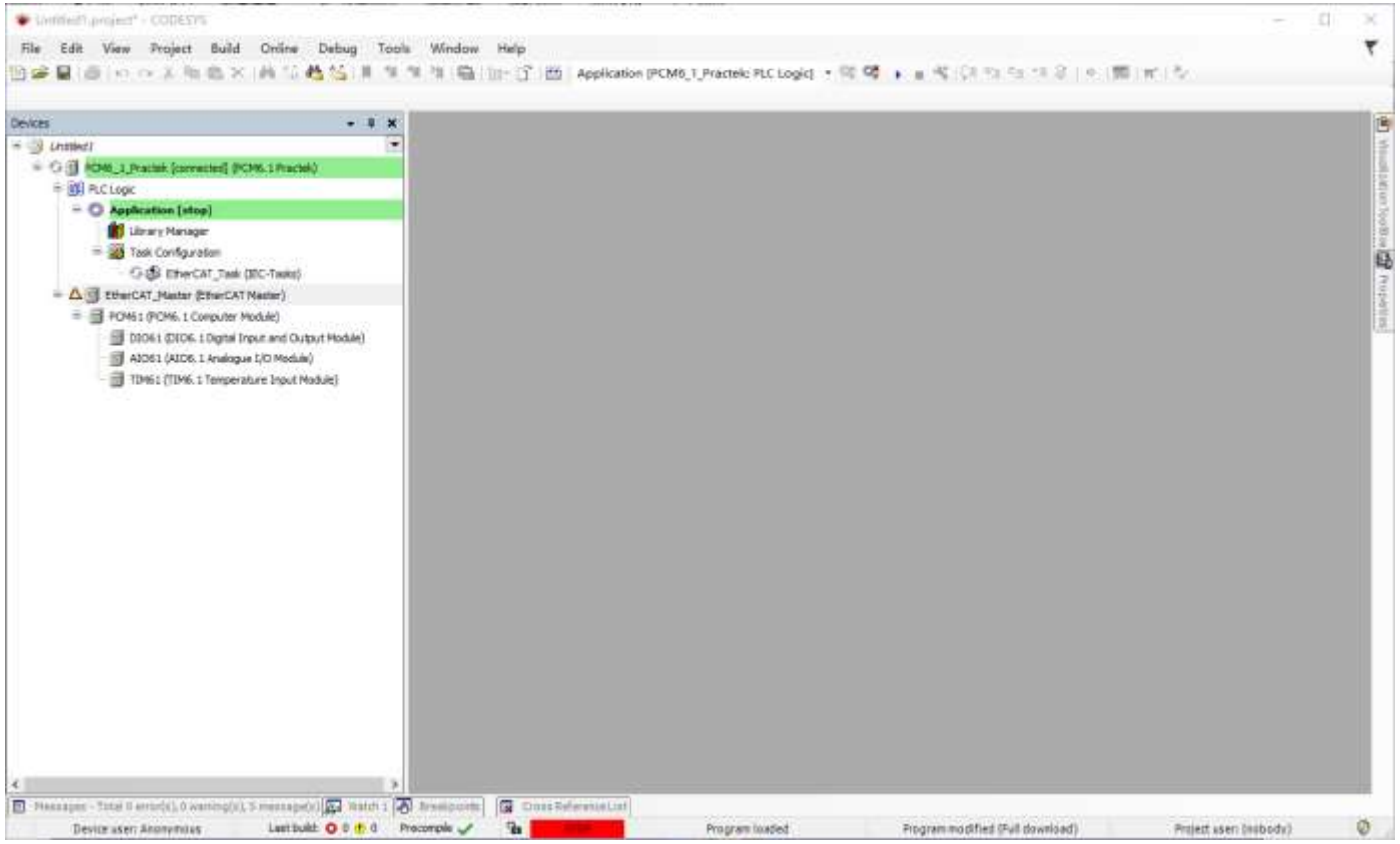
扫描到的模块会被显示出来，选择Copy ALL Devices to Project。



# 新建第一个CoDeSys工程

## ➢ 扫描模块

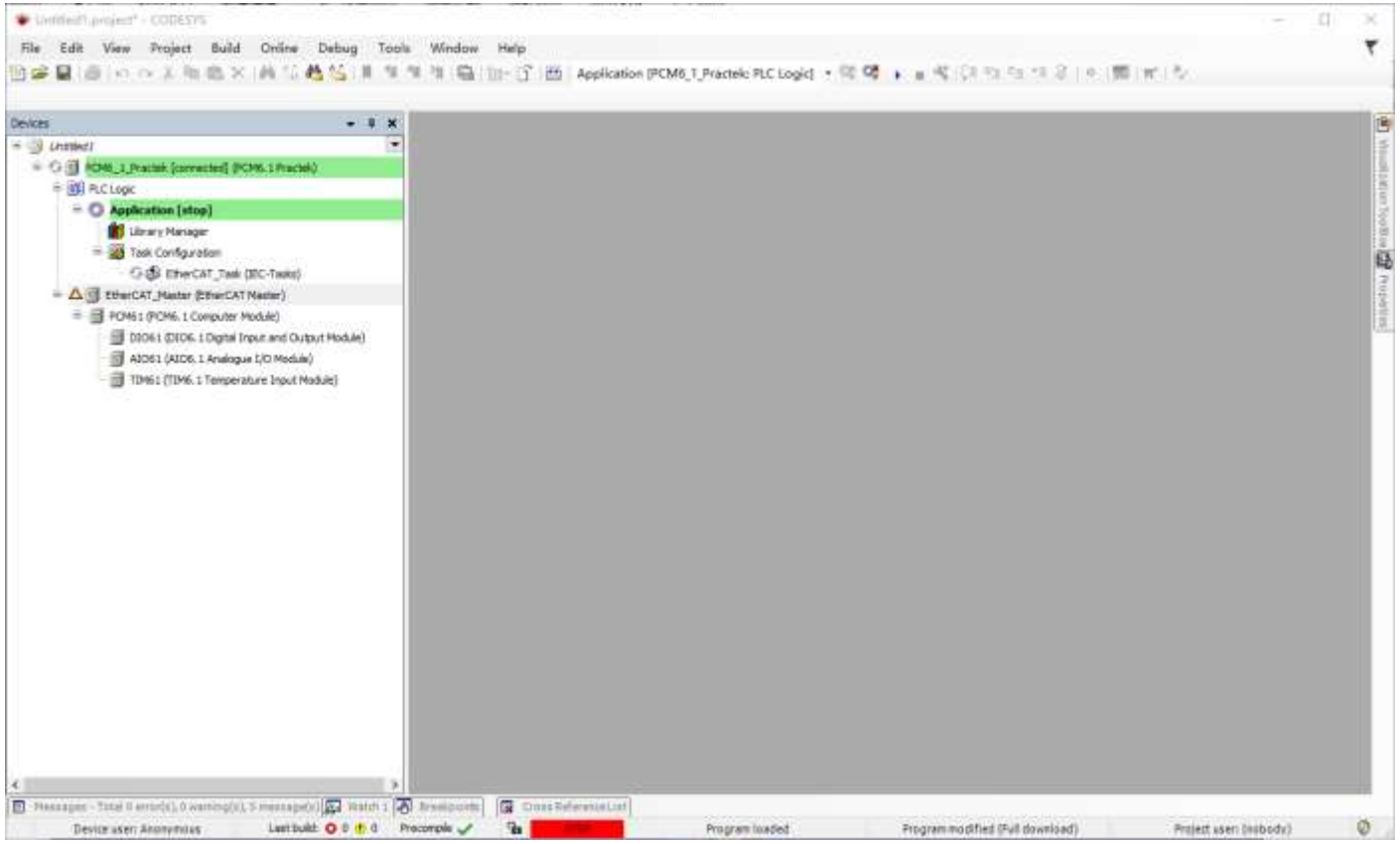
添加完模块，工程如下。



# 新建第一个CoDeSys工程

## ➢ 扫描模块

添加完模块，工程如下。



# 新建第一个CoDeSys工程

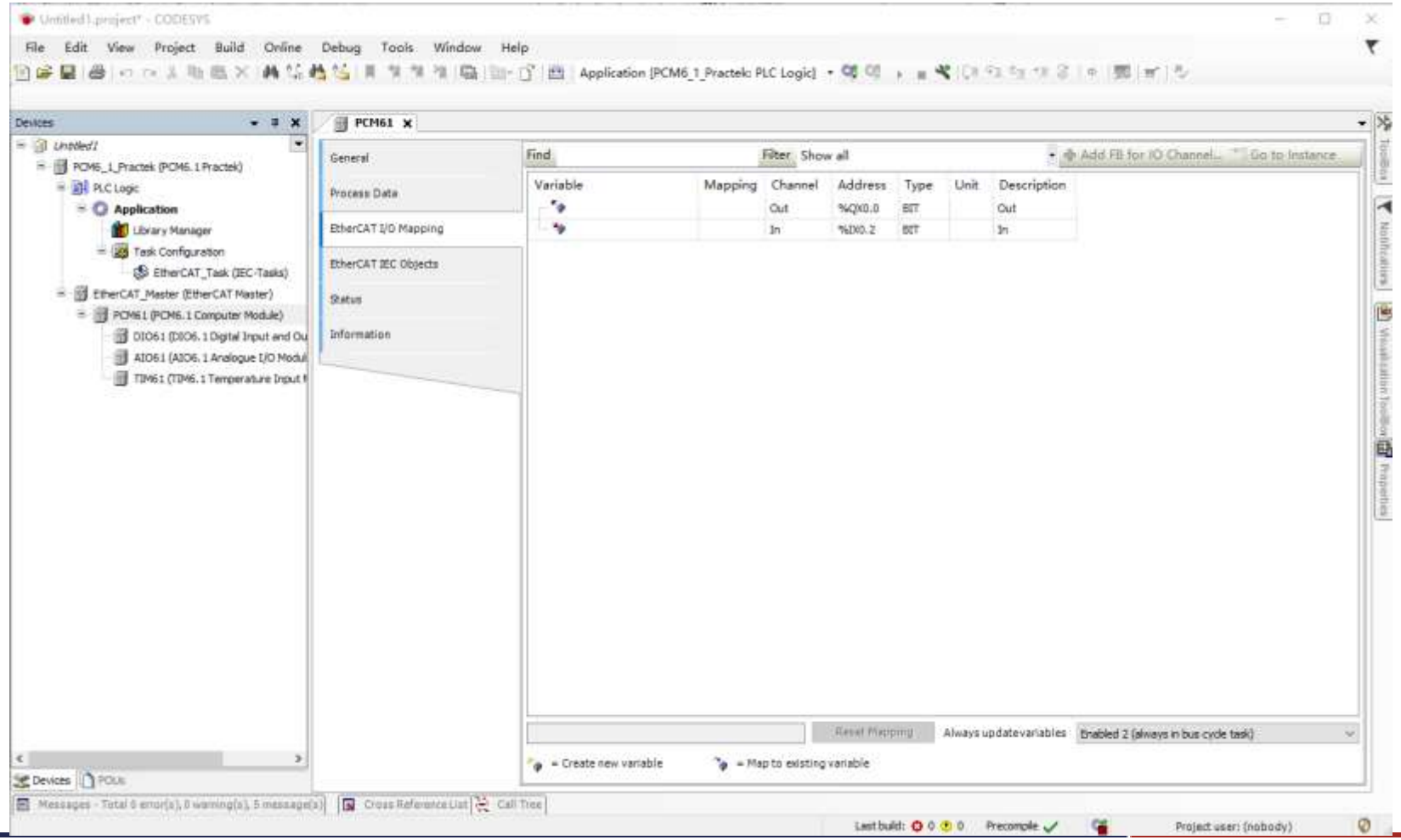
## ➢ 设置EtherCAT I/O Mapping

轮流选中刚添加的模块，点开EtherCAT I/O Mapping界面，将右下角的Always update variables，由Use parent device setting设置为Enabled 2 (always in bus cycle task)。

Always update variables也可以保持默认设置Use parent device setting，但是需要设置PCM61_1_Practek->PLC Settings中的Always update variables，设置为Enabled 2 (always in bus cycle task)。

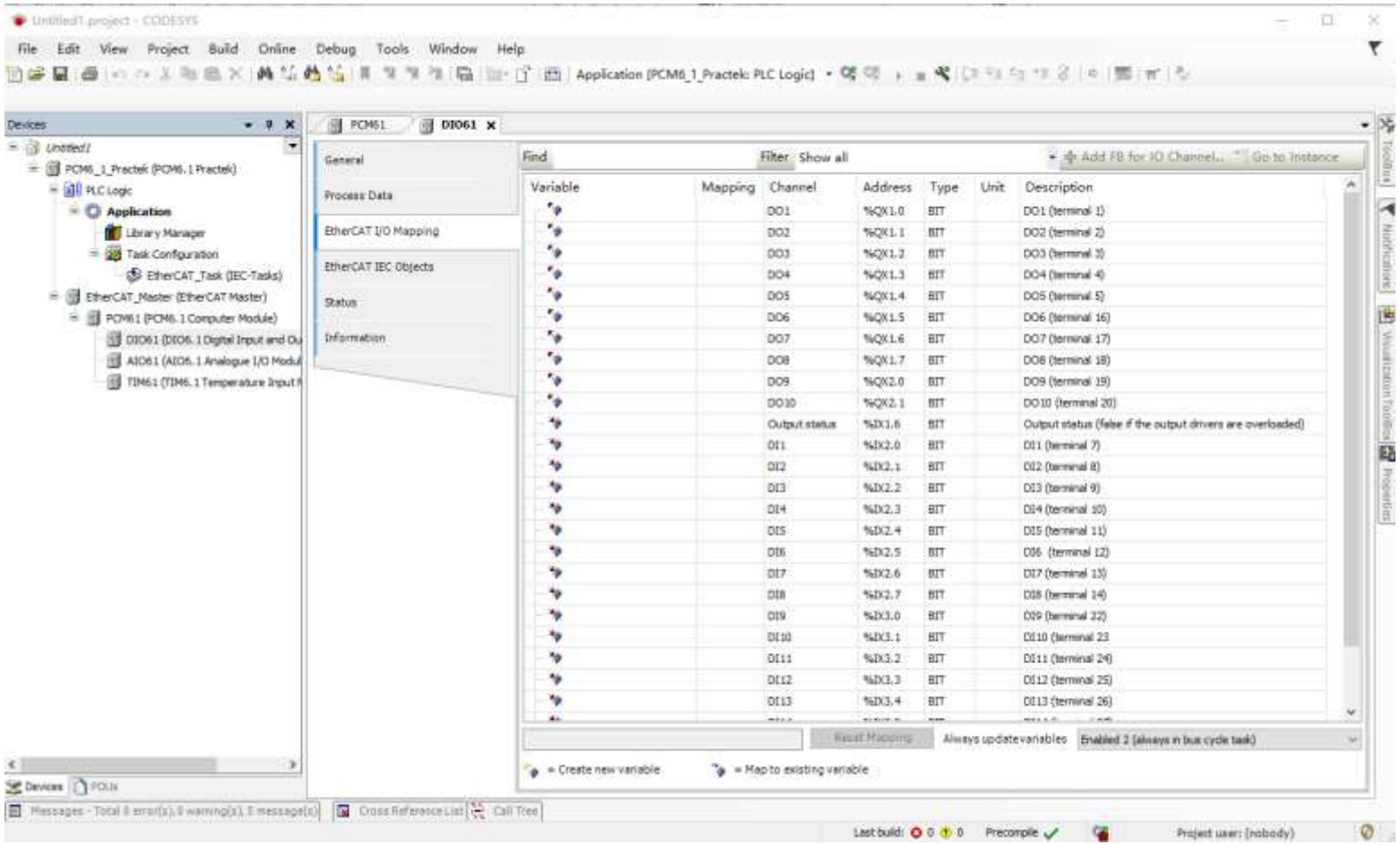
# 新建第一个CoDeSys工程

➢ 设置EtherCAT I/O Mapping



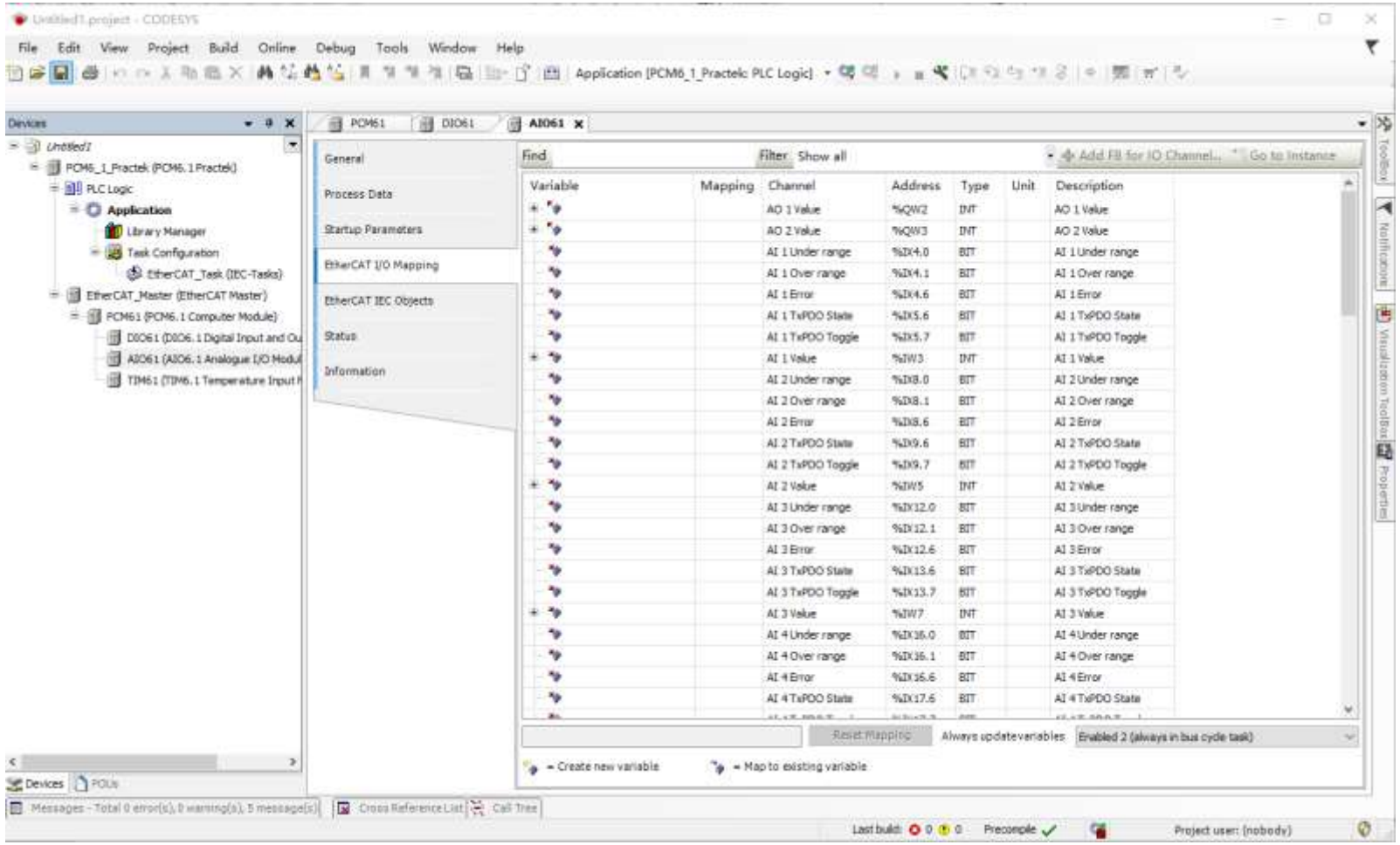
# 新建第一个CoDeSys工程

➢ 设置EtherCAT I/O Mapping



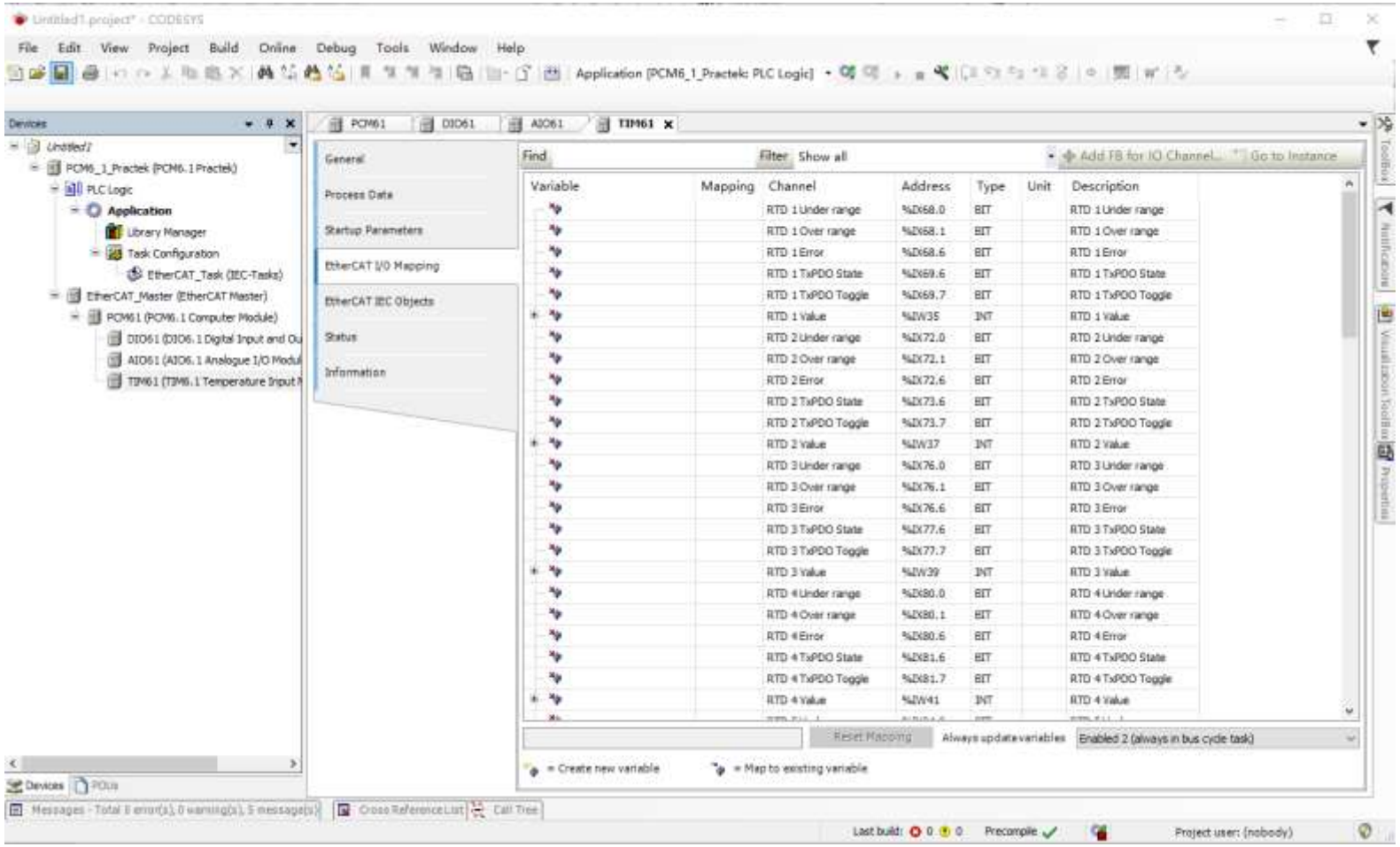
# 新建第一个CoDeSys工程

➢ 设置EtherCAT I/O Mapping



# 新建第一个CoDeSys工程

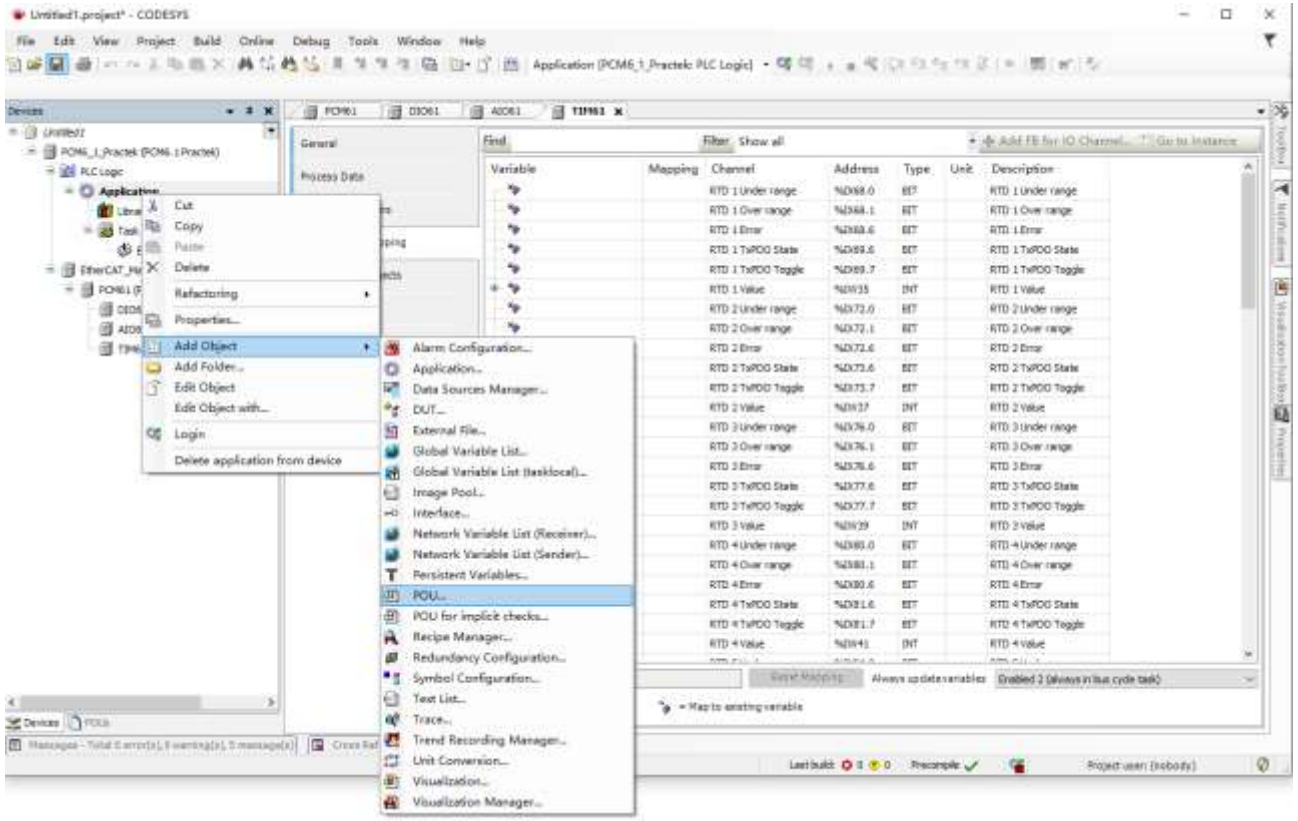
➢ 设置EtherCAT I/O Mapping



# 新建第一个CoDeSys工程

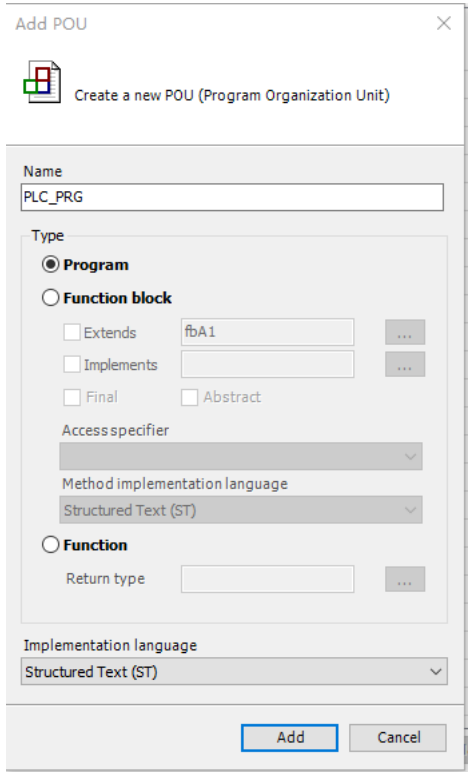
## ➢ 编写程序

添加POU，Application -> Add Object -> POU…。



# 新建第一个CoDeSys工程

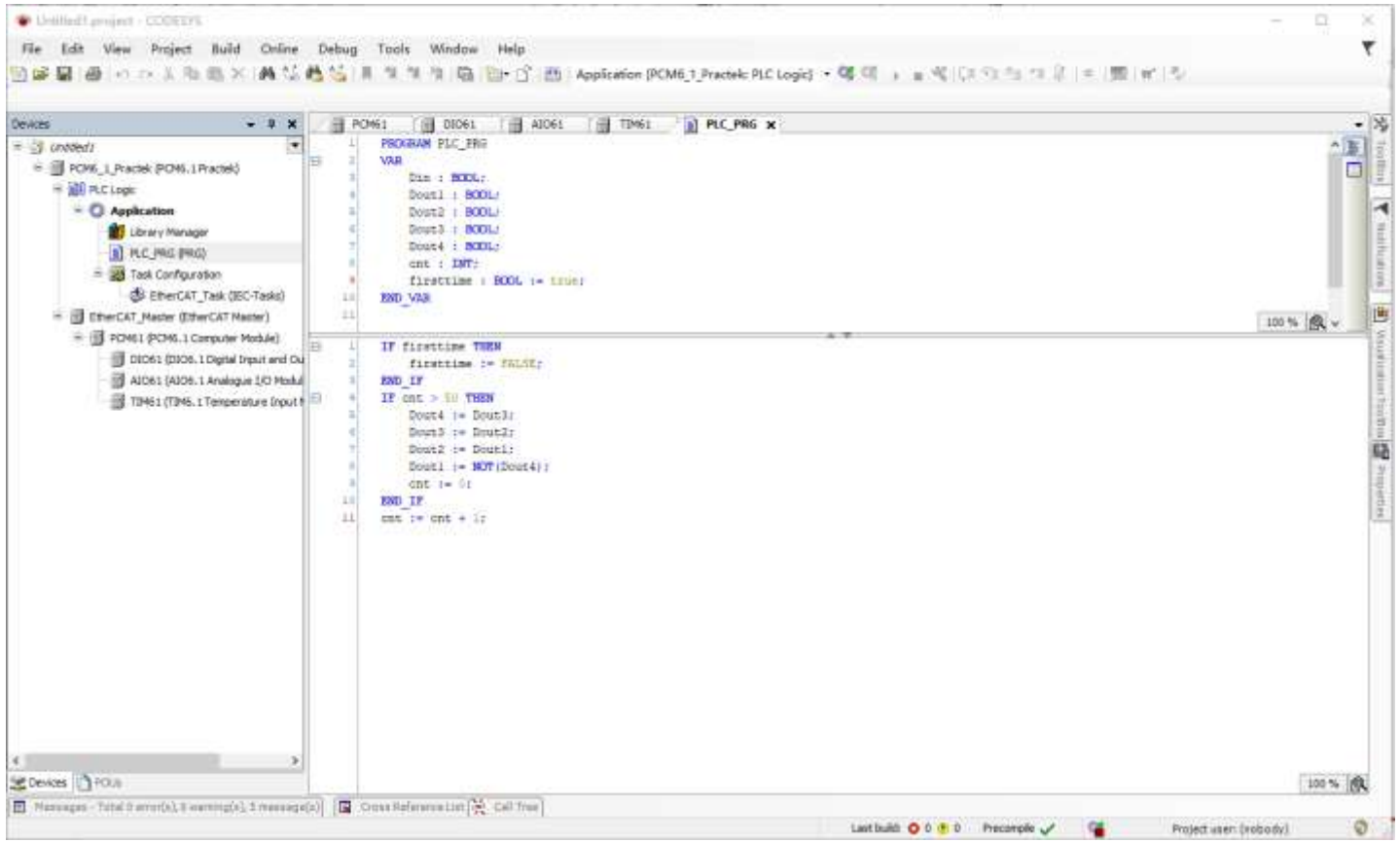
➢ 编写程序



# 新建第一个CoDeSys工程

## ➢ 编写程序

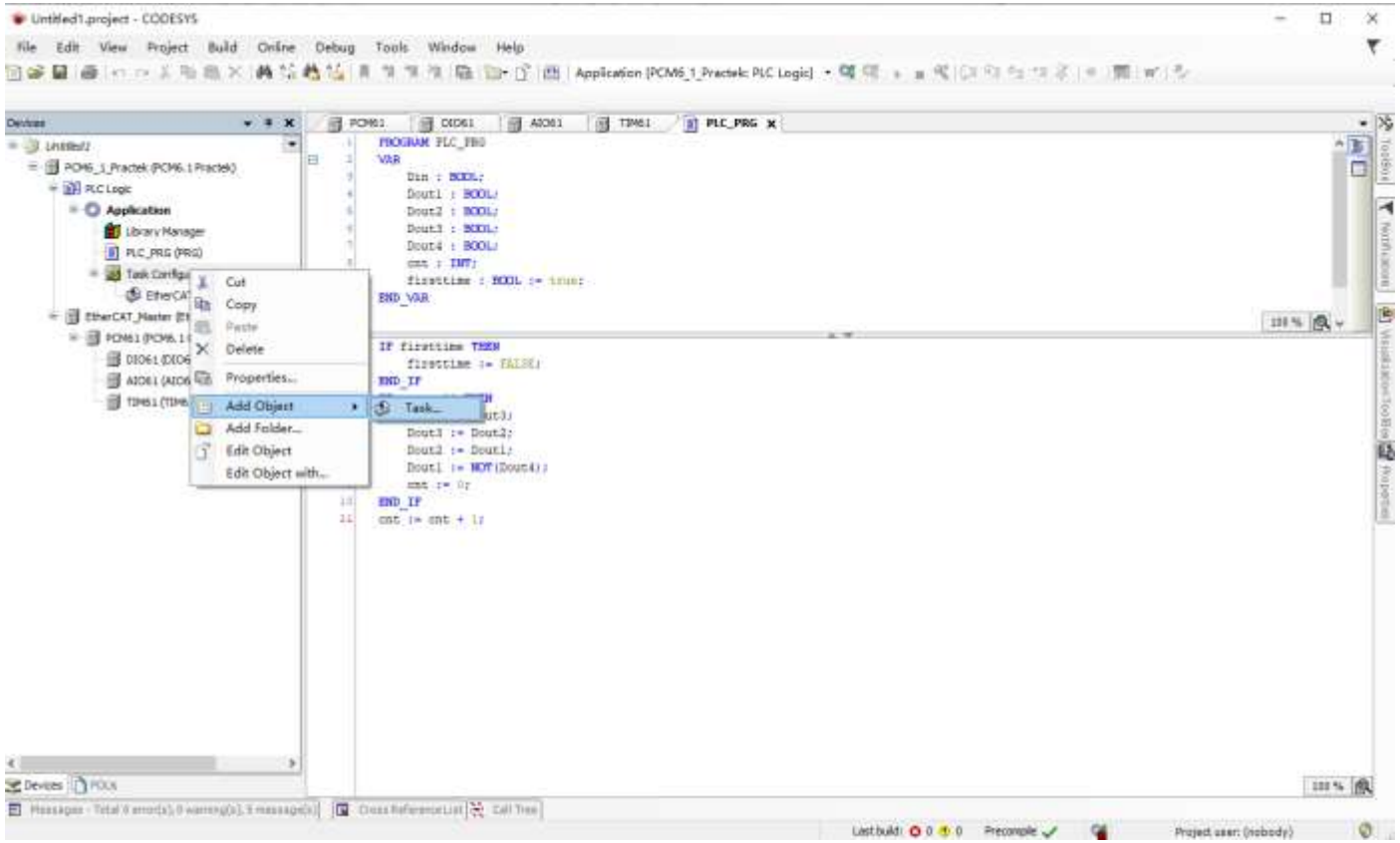
编写代码



# 新建第一个CoDeSys工程

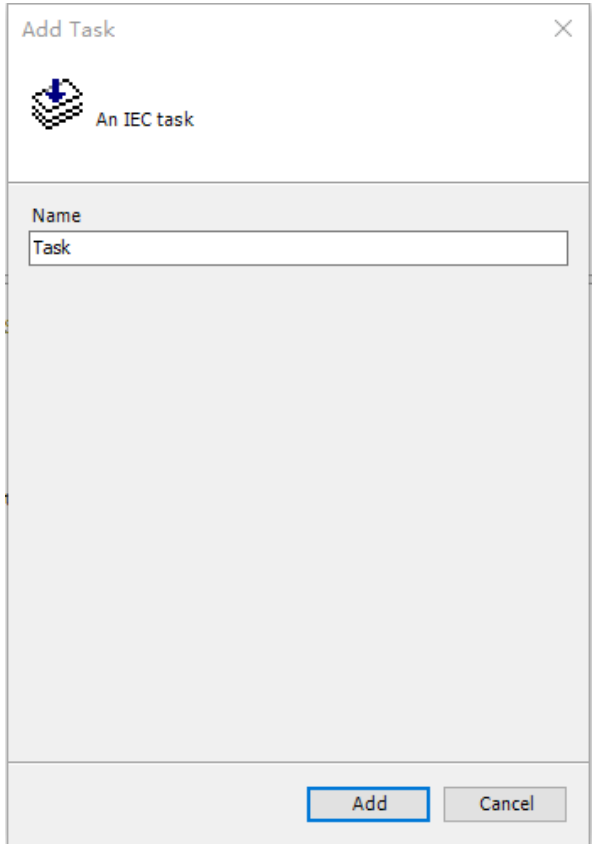
➢ **添加task**

Task Configuration -> Add Object -> Task…。



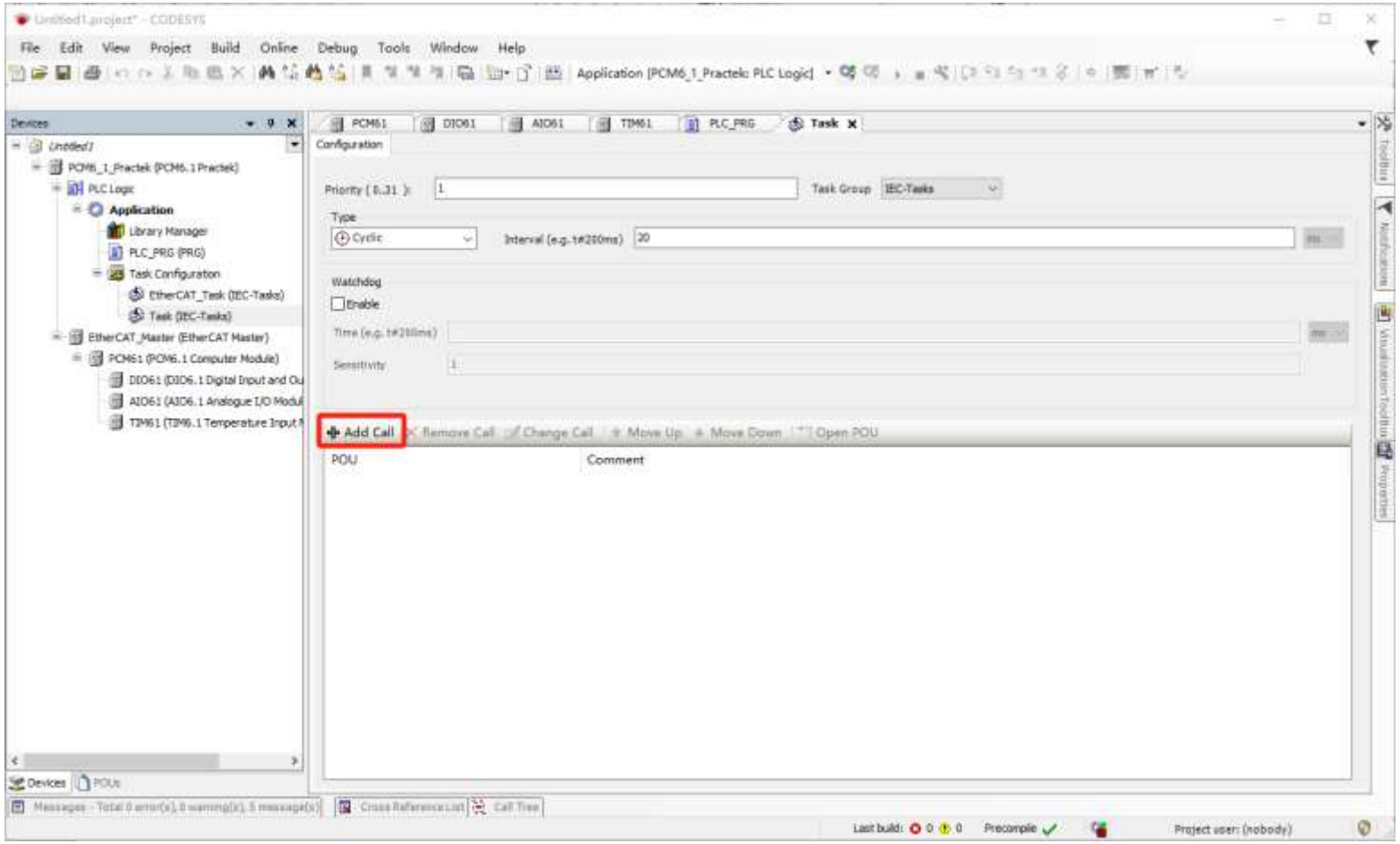
# 新建第一个CoDeSys工程

➢ 添加task



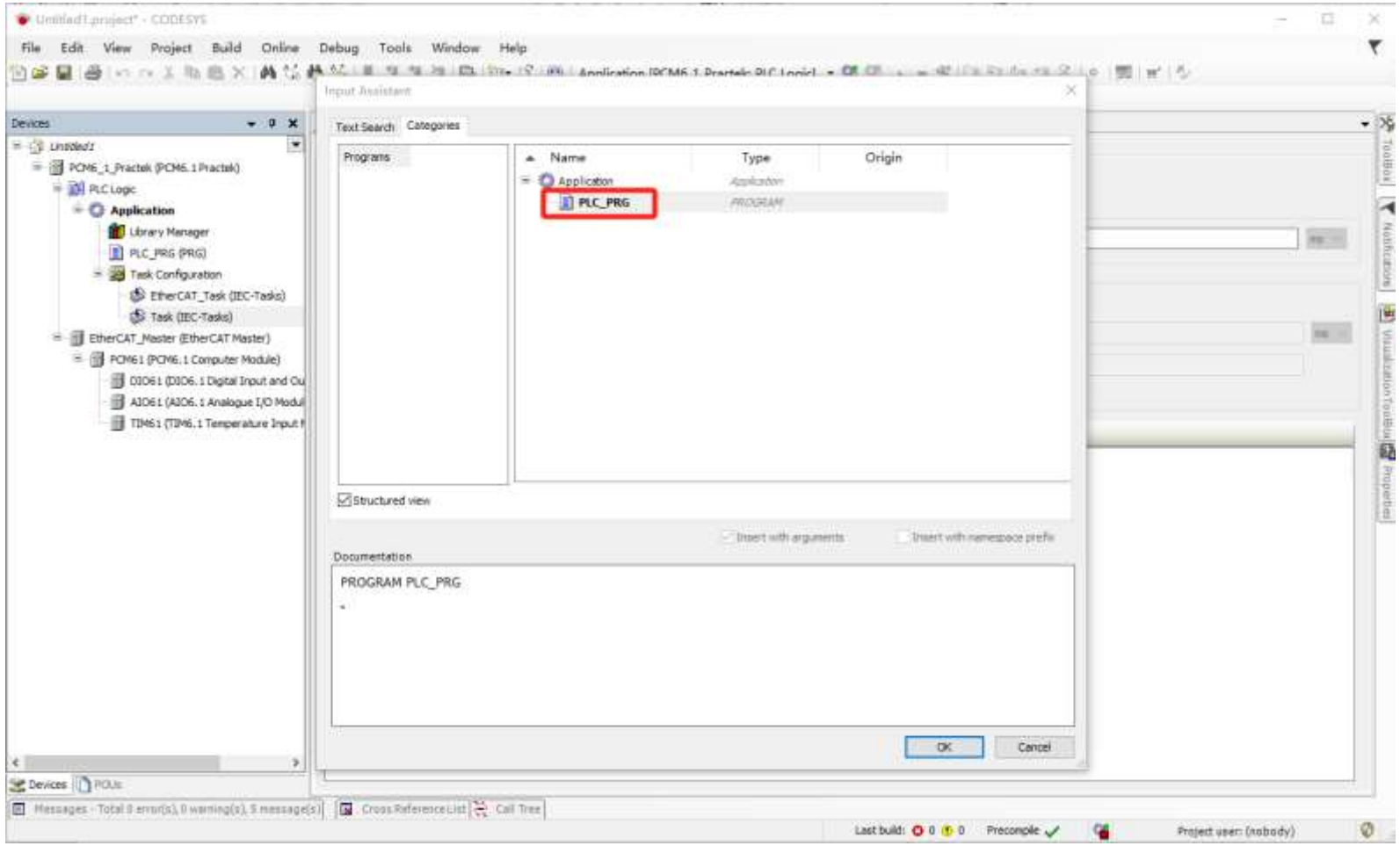
# 新建第一个CoDeSys工程

➢ 添加task



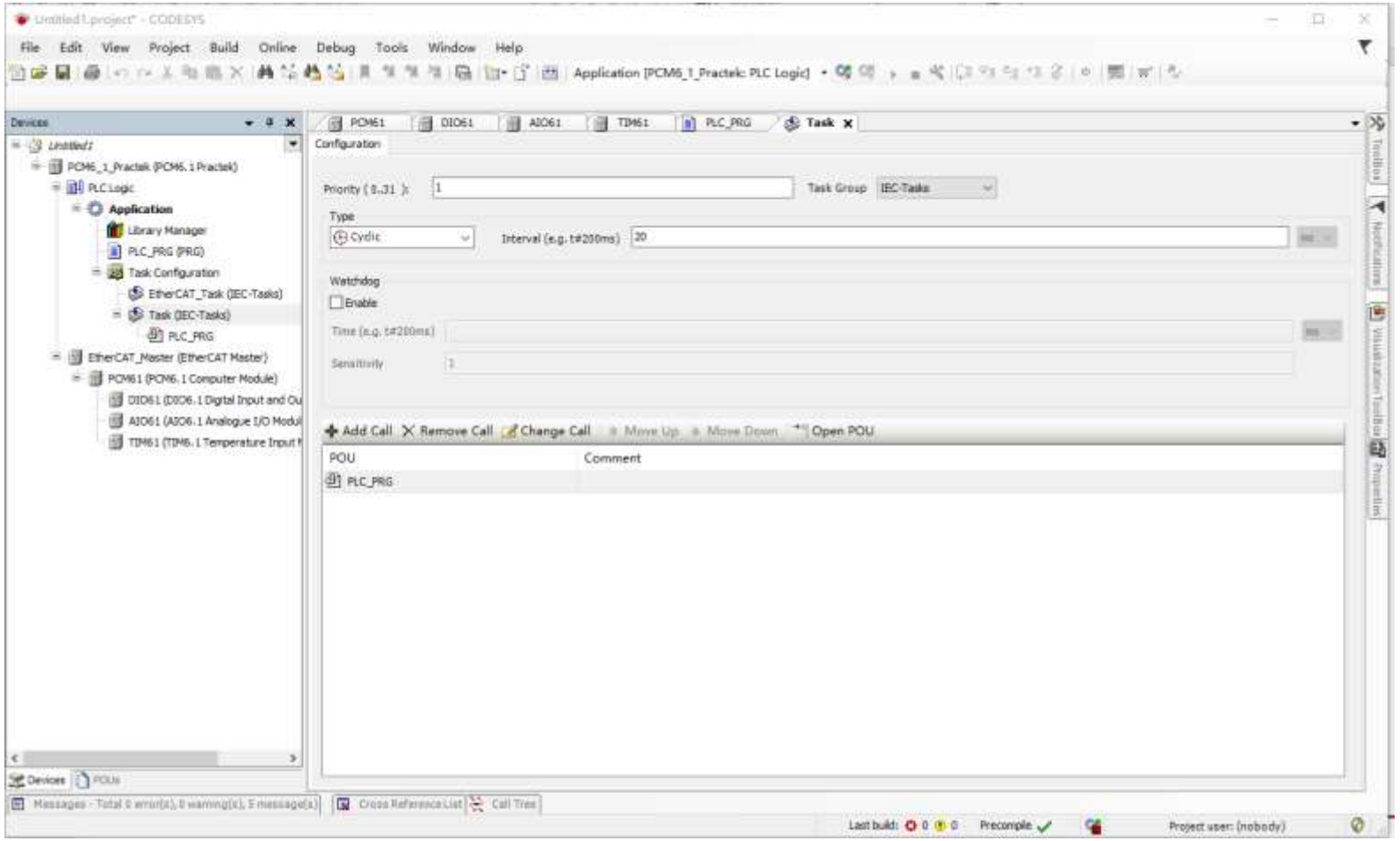
# 新建第一个CoDeSys工程

➢ 添加task



# 新建第一个CoDeSys工程

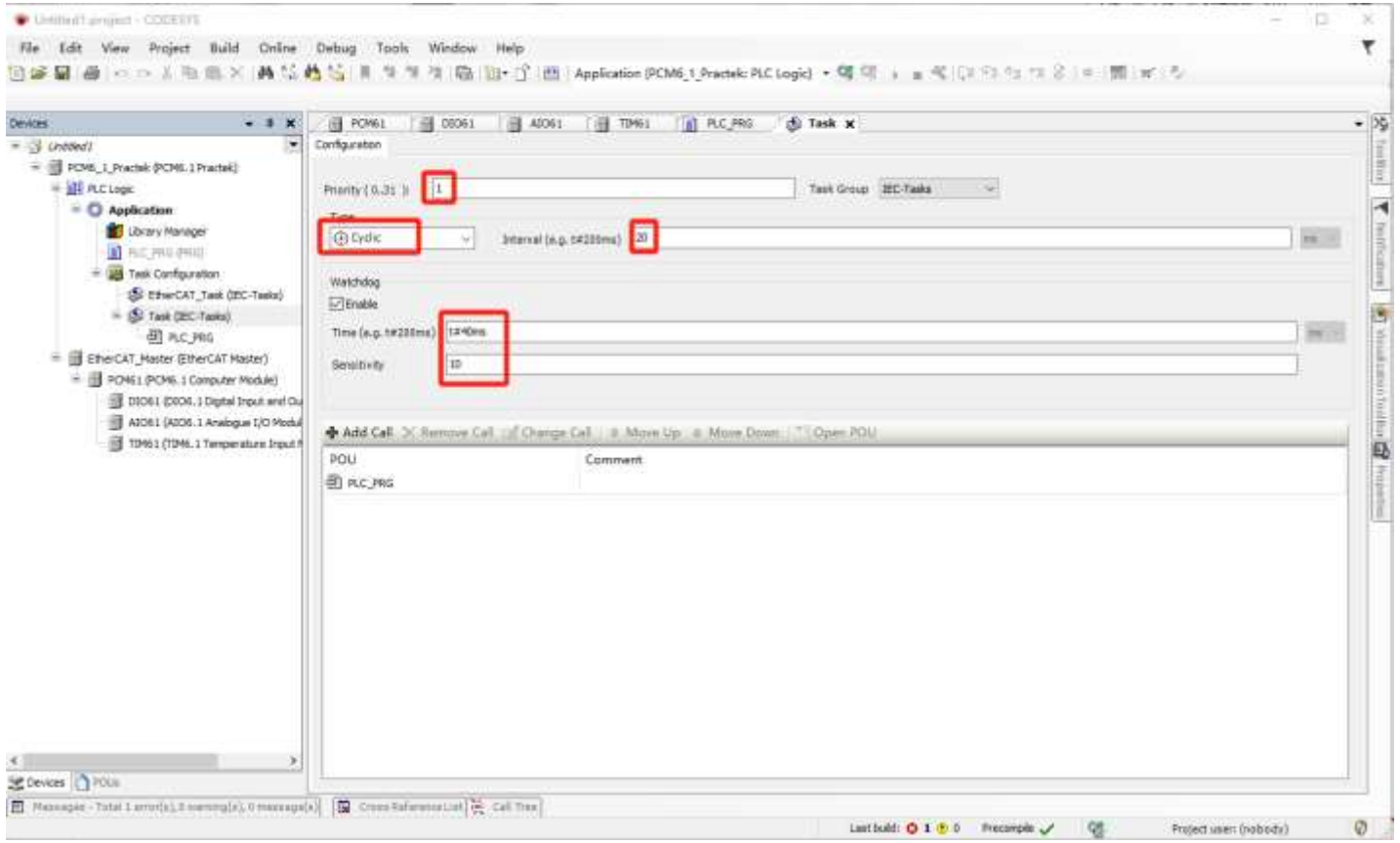
➢ 添加task



# 新建第一个CoDeSys工程

➢ **添加task**

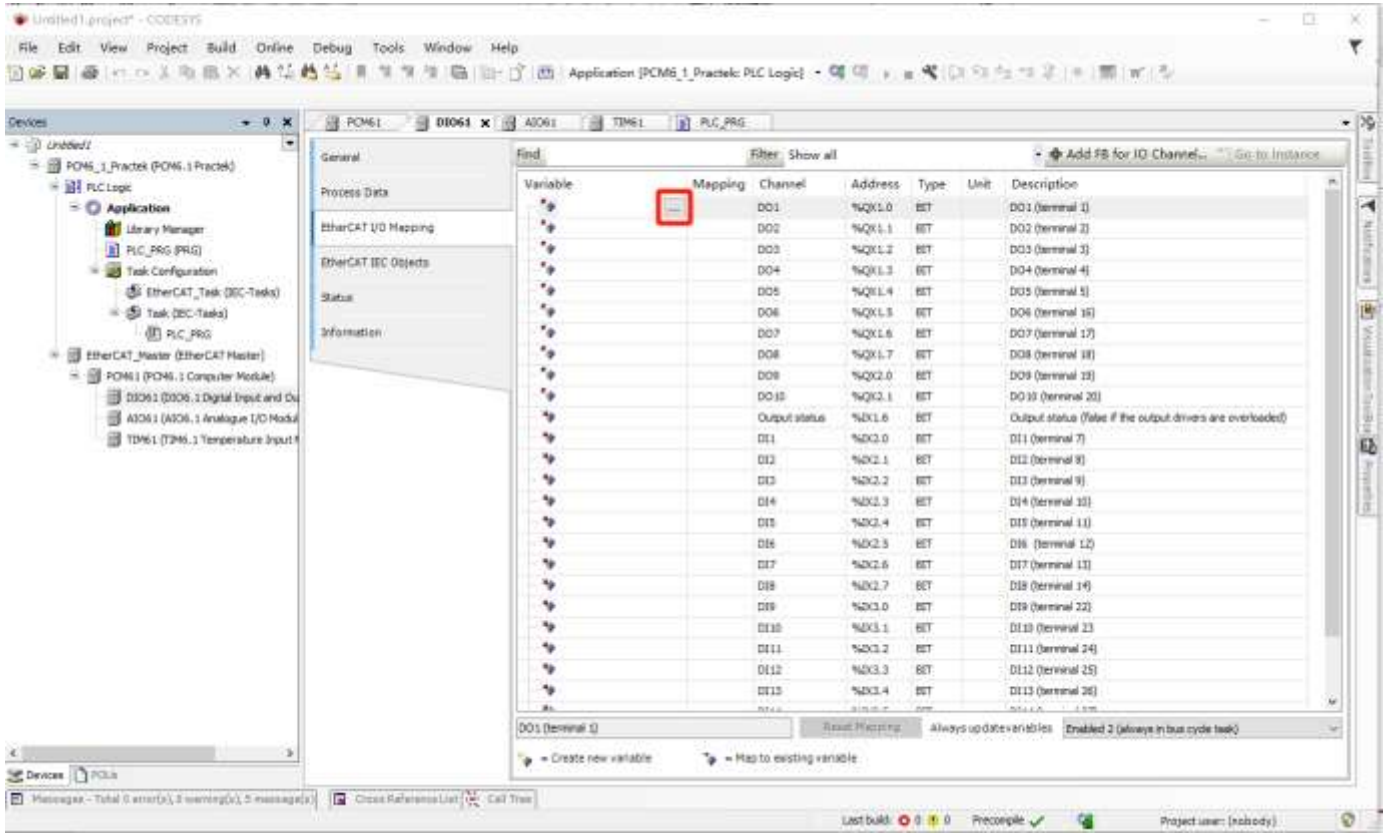
根据需要设置priority、type和Watchdog



# 新建第一个CoDeSys工程

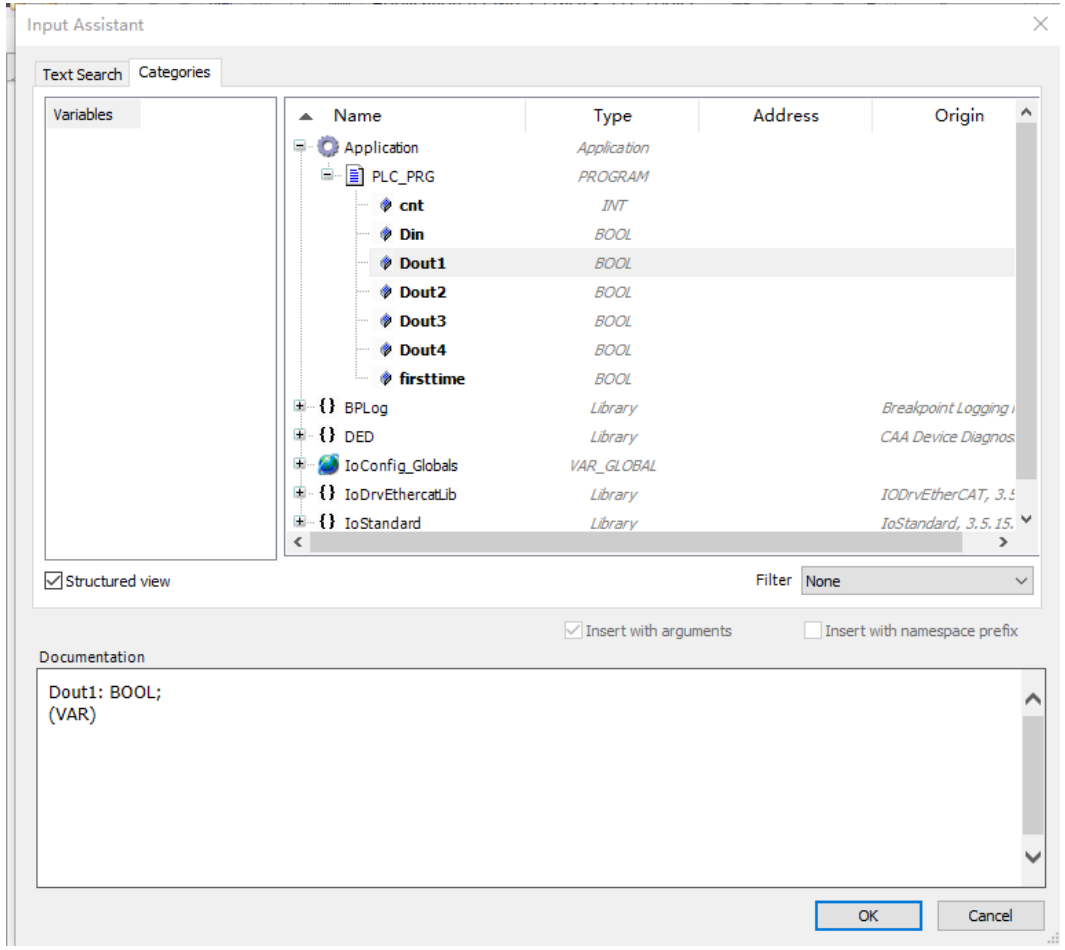
## ➢ 连接IO变量

在EtherCAT I/O Mapping界面，选择通道，点击...图标。



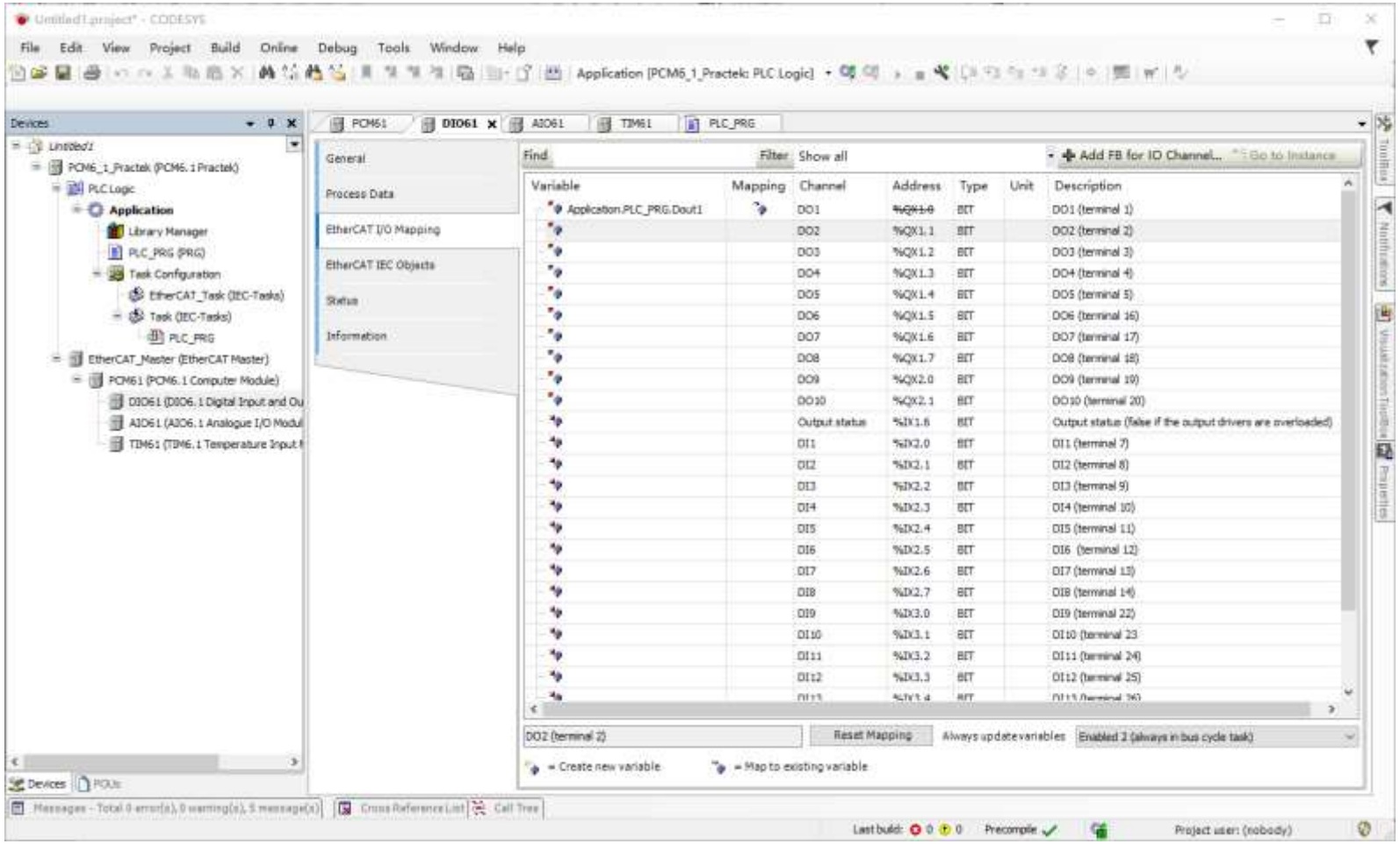
# 新建第一个CoDeSys工程

➢ 连接IO变量



# 新建第一个CoDeSys工程

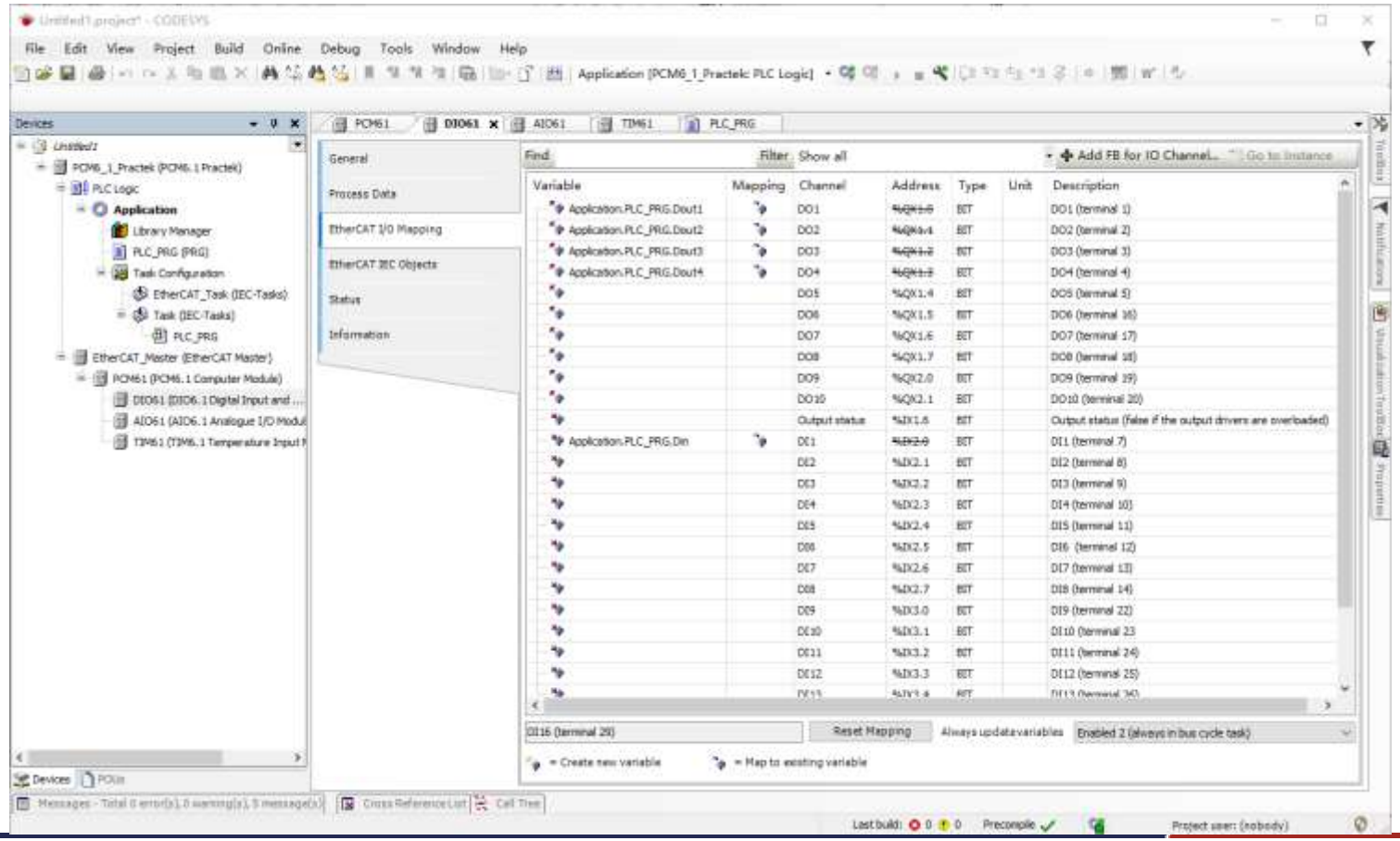
➢ 连接IO变量



# 新建第一个CoDeSys工程

## ➢ 连接IO变量

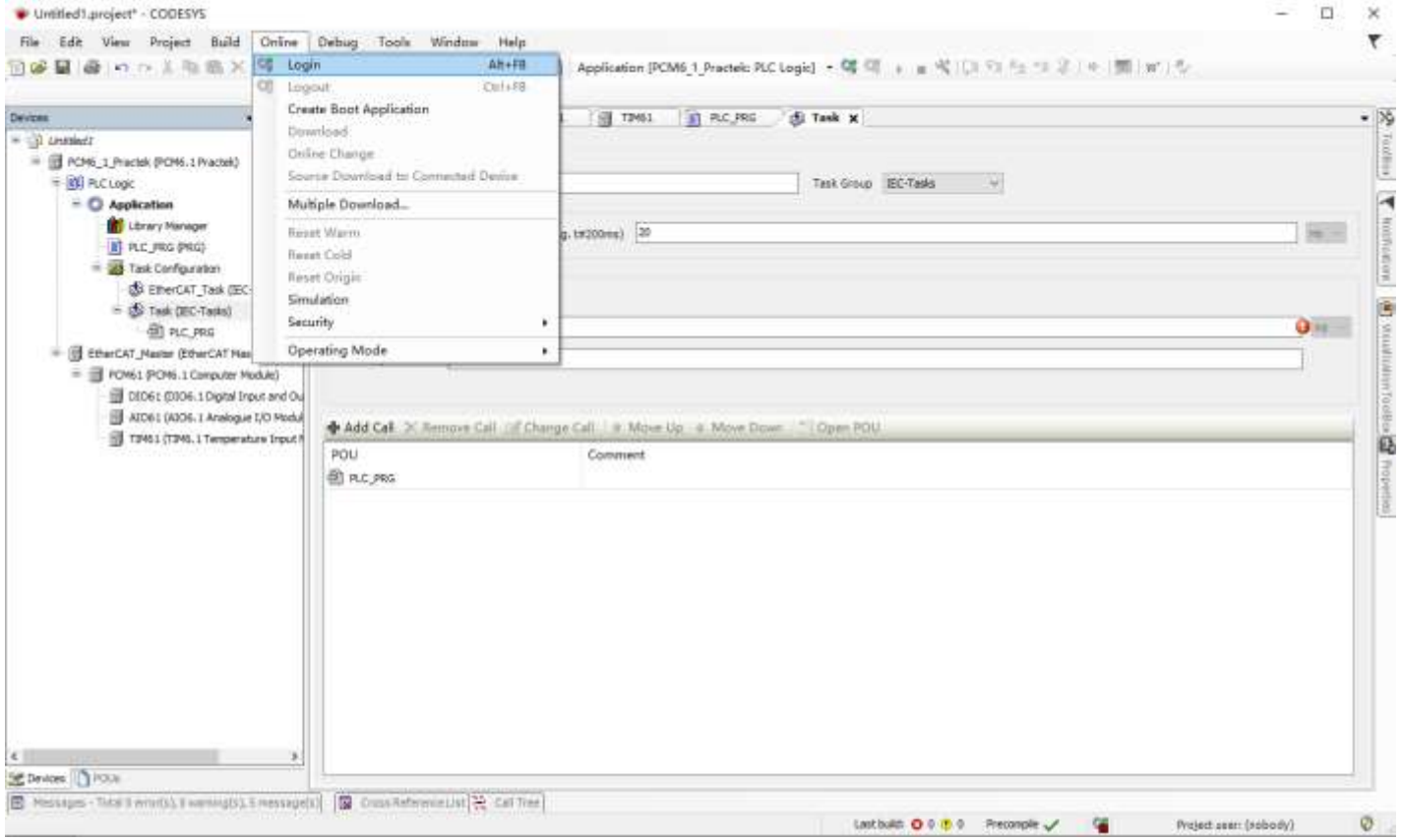
连接完所有的变量



# 新建第一个CoDeSys工程

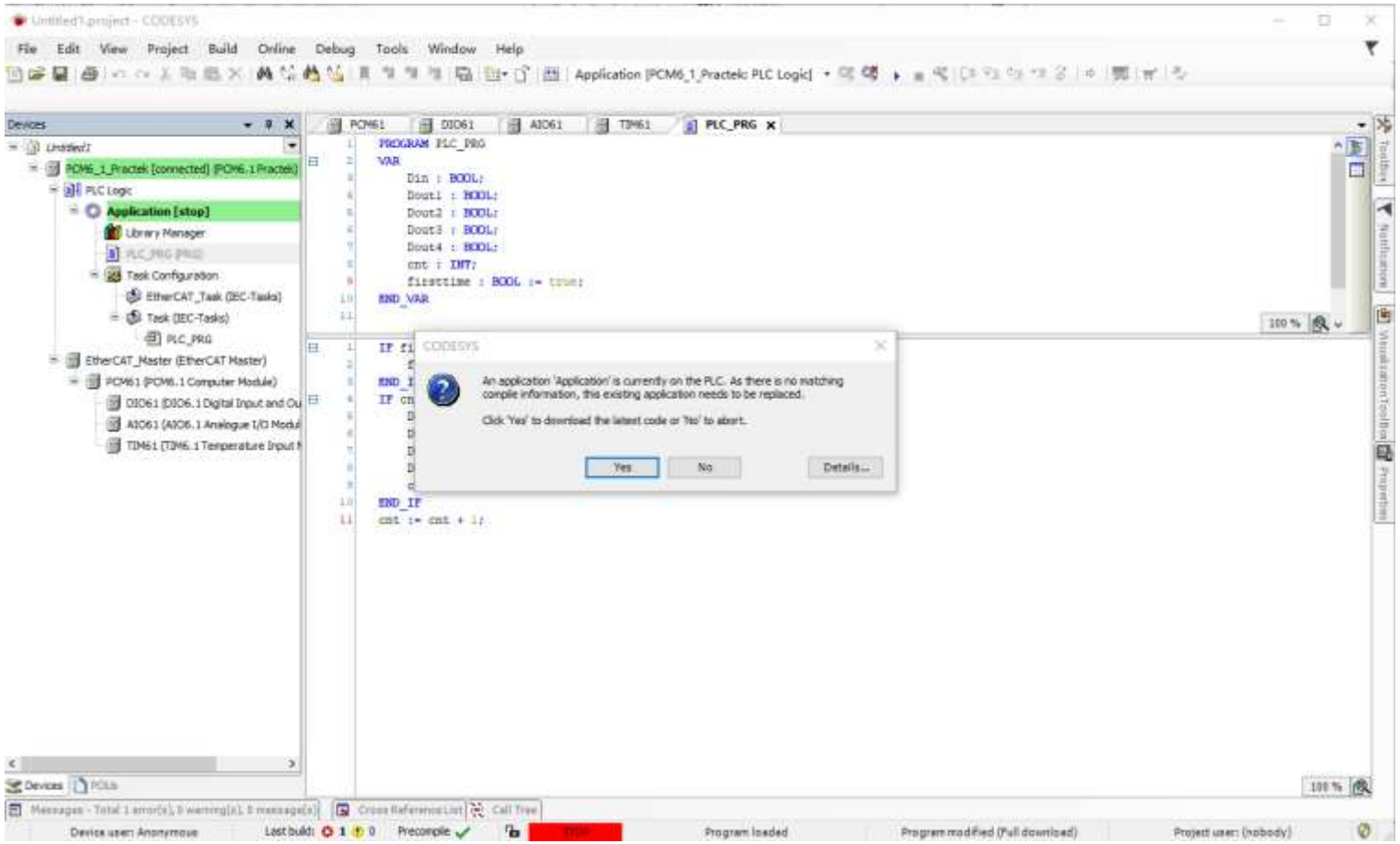
➢ **登录Login**

Onling -> Login



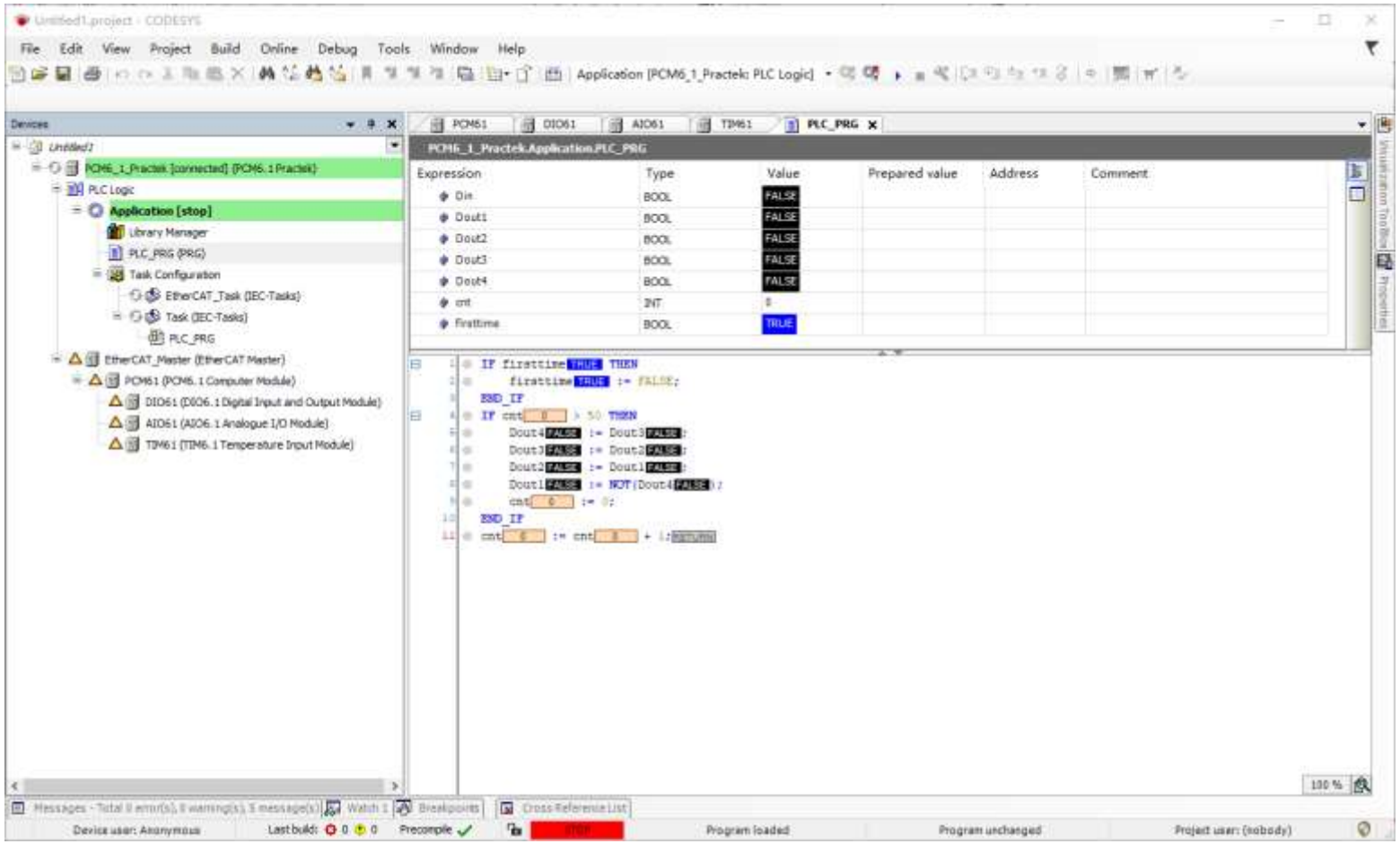
# 新建第一个CoDeSys工程

➢ 登录Login



# 新建第一个CoDeSys工程

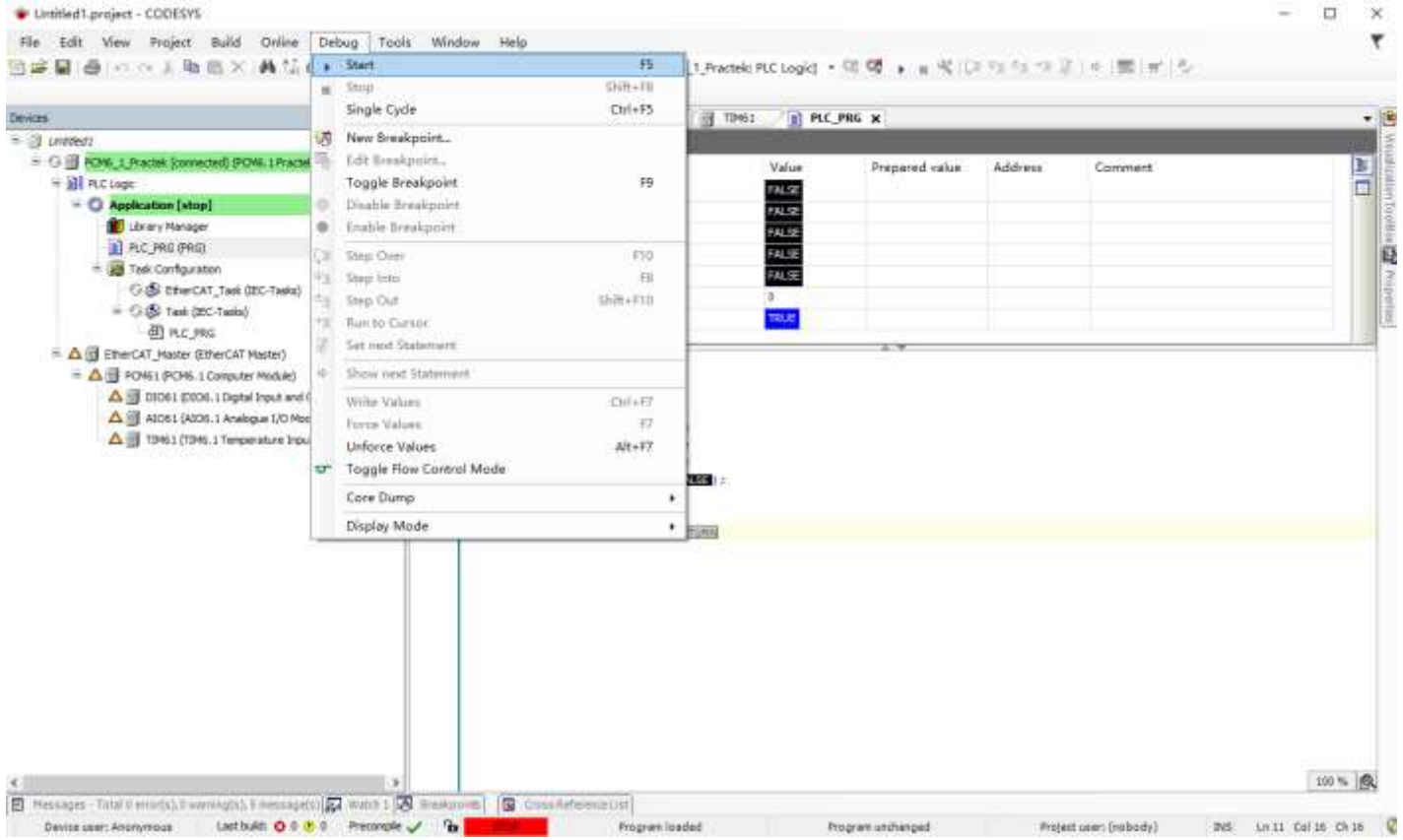
➢ 登录Login



# 新建第一个CoDeSys工程

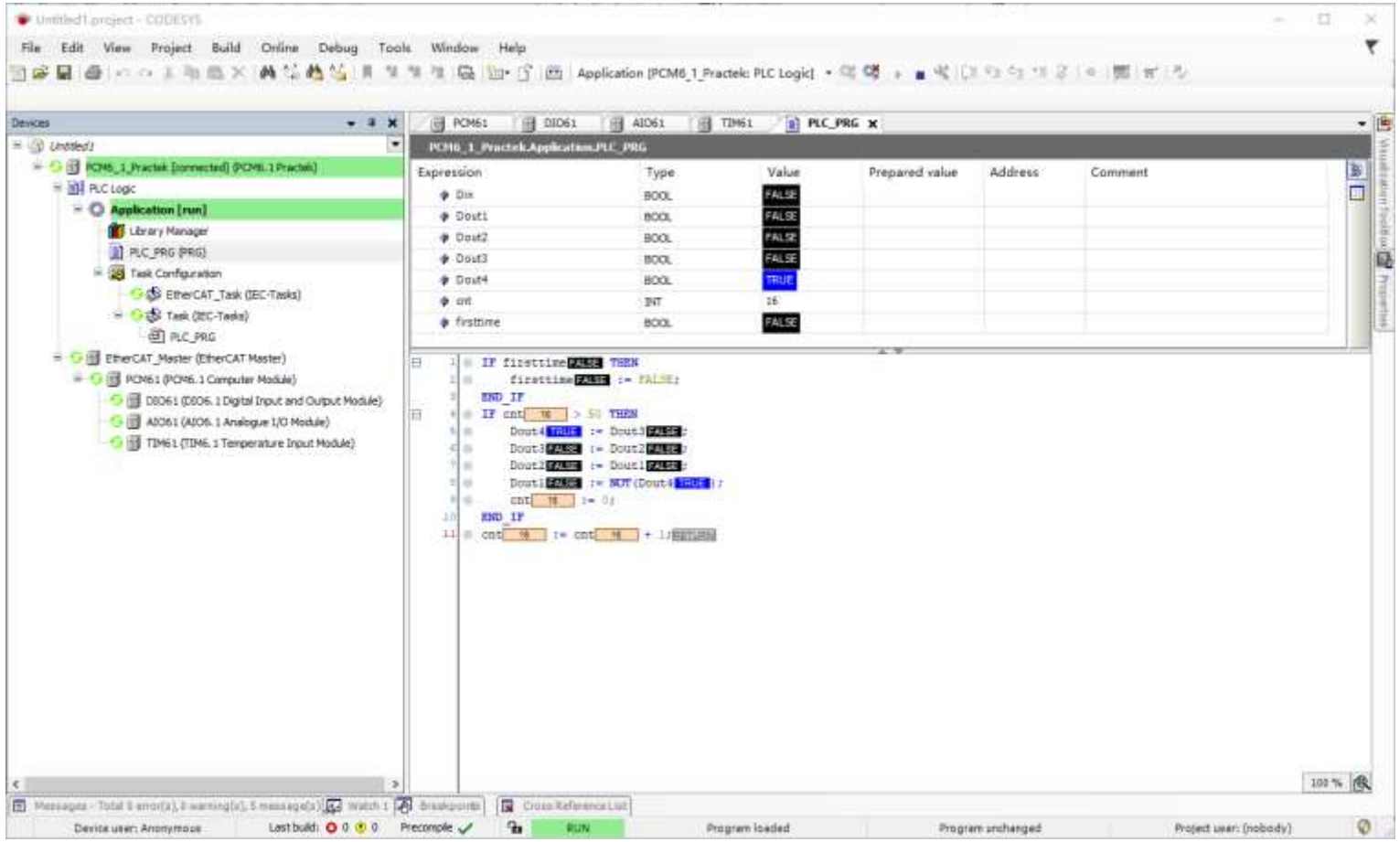
➢ **运行程序**

Debug -> Start



# 新建第一个CoDeSys工程

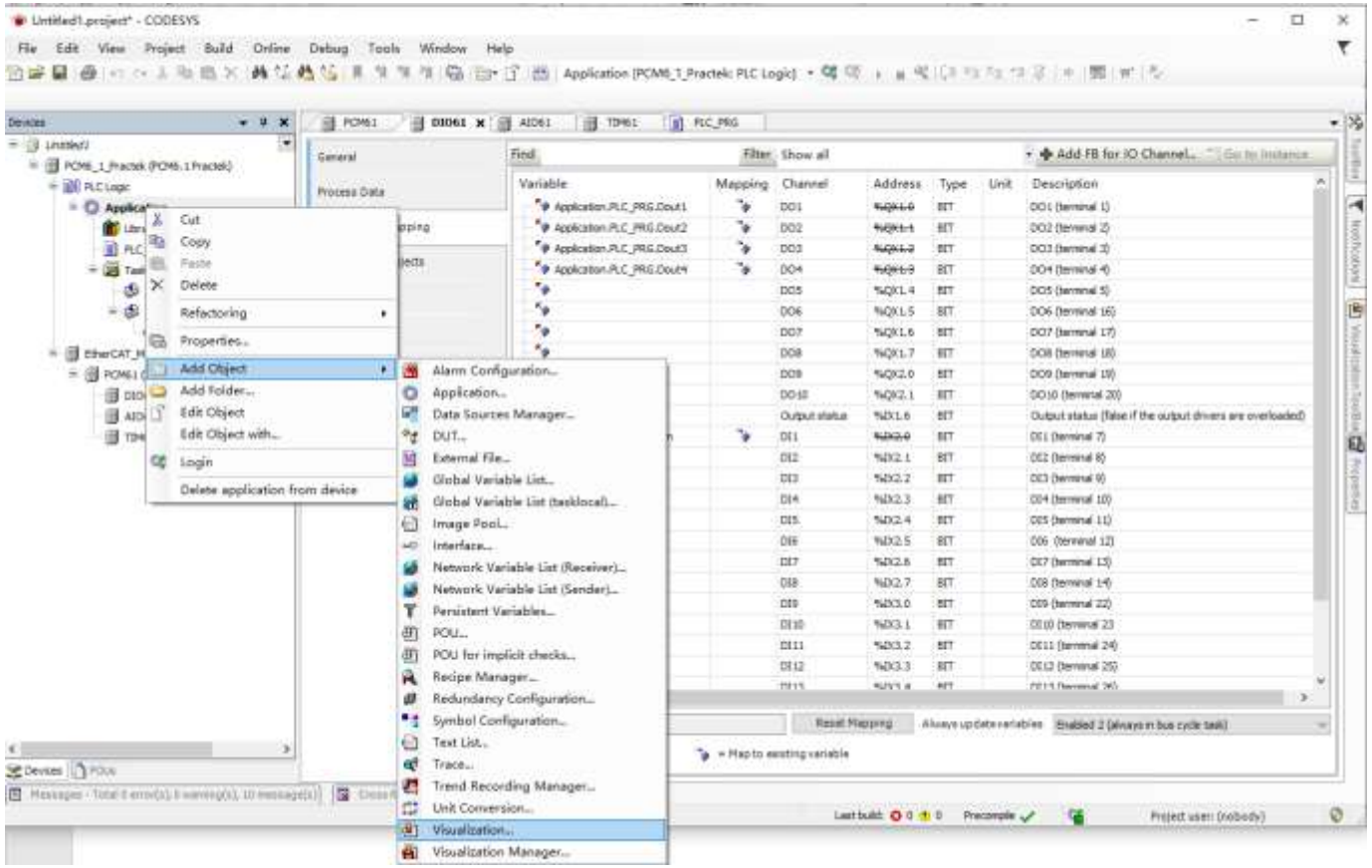
➢ 运行程序



# 新建第一个CoDeSys工程

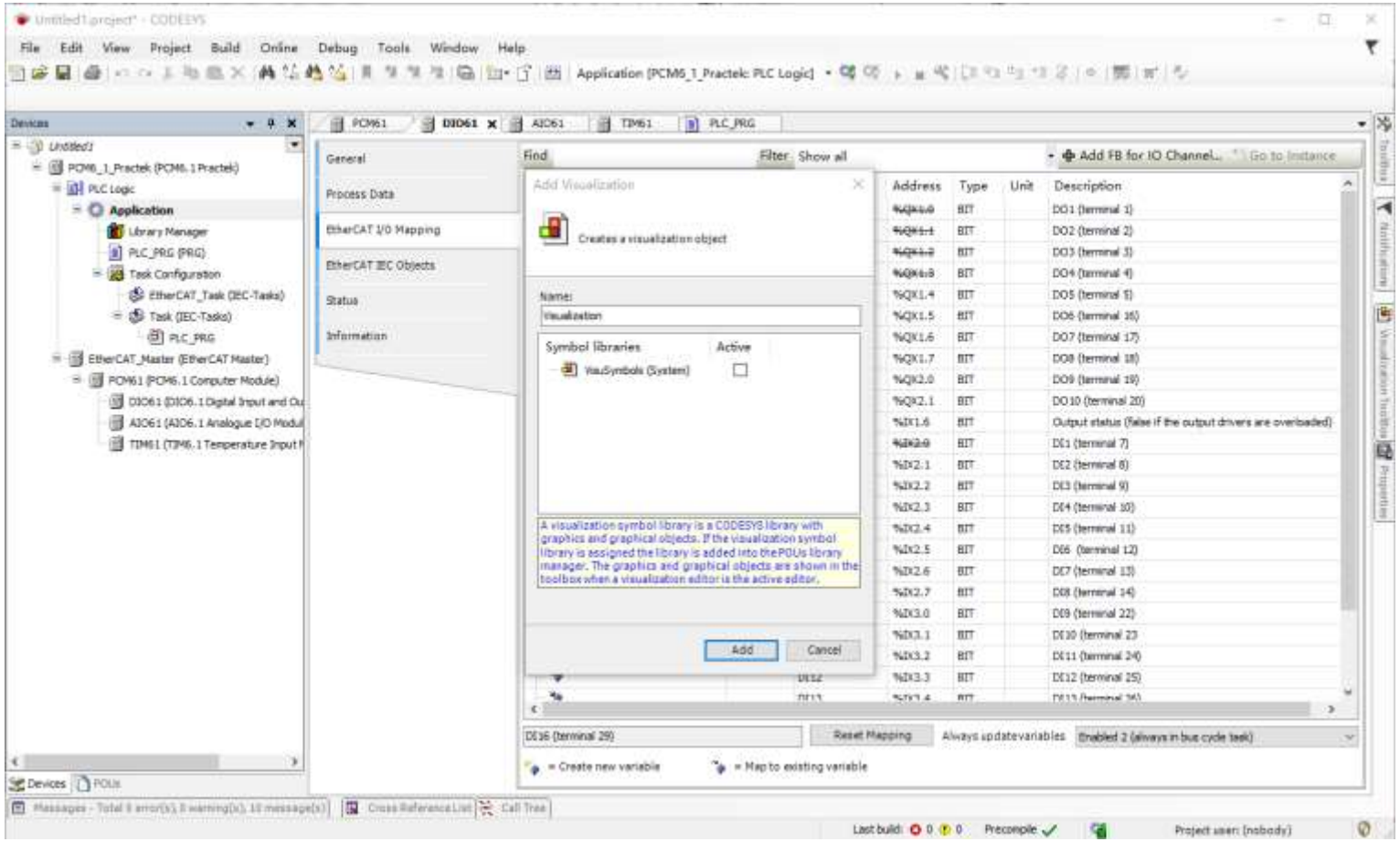
➢ **CoDeSys HMI**

点击Online -> Logout登出程序，点击Application -> Add Object -> Visualization…，创建界面



# 新建第一个CoDeSys工程

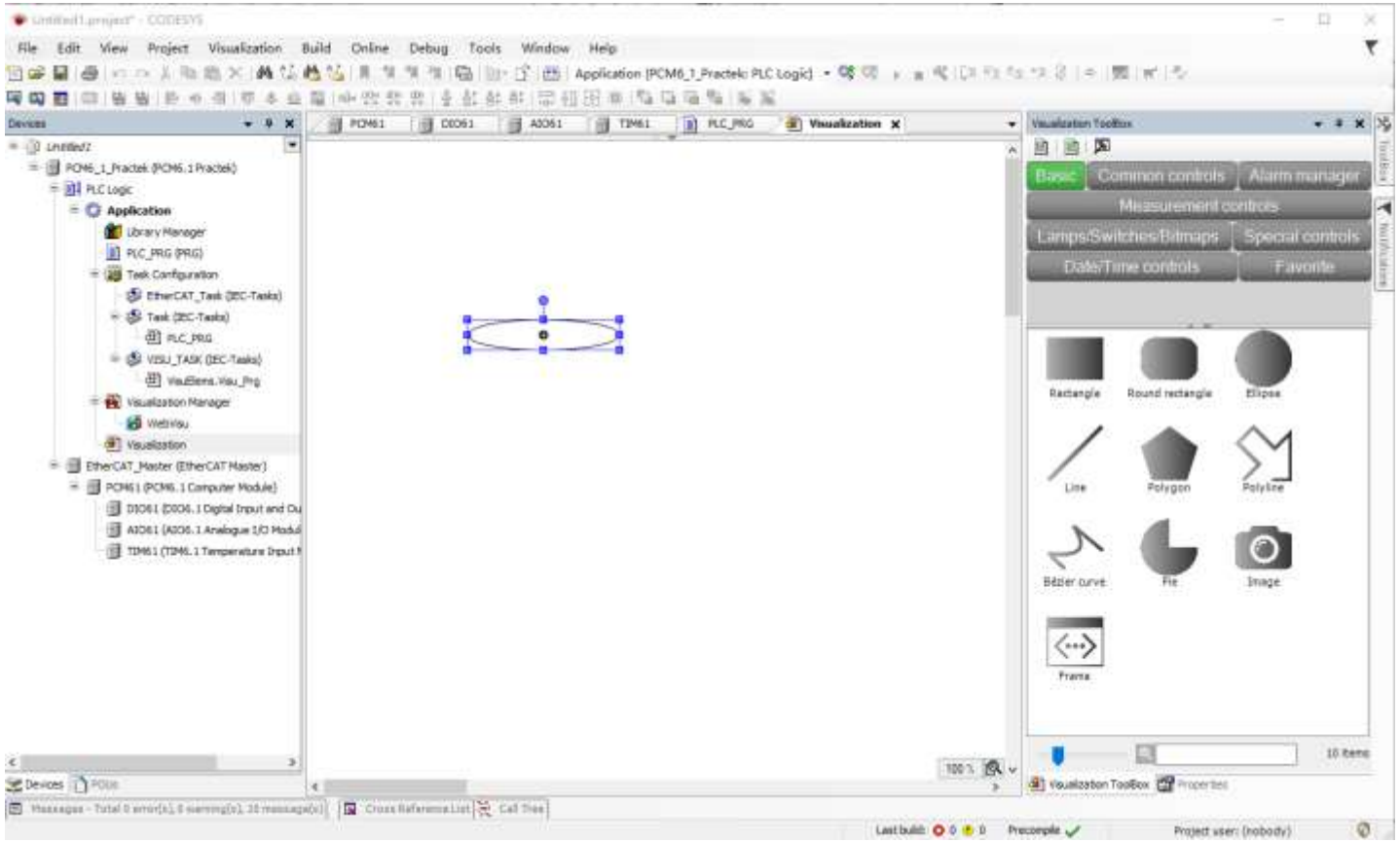
➢ CoDeSys HMI



# 新建第一个CoDeSys工程

➢ **CoDeSys HMI**

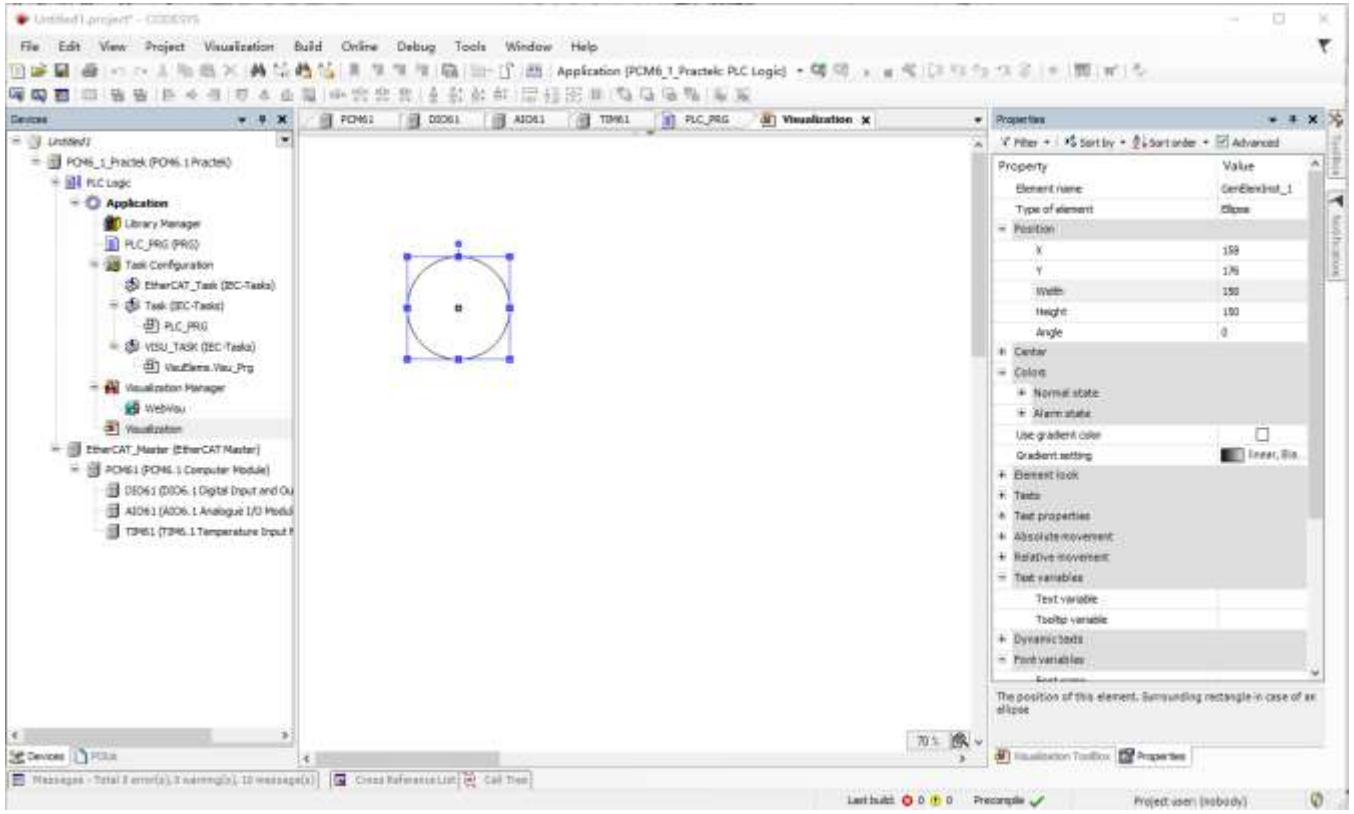
拖一个Eclipse控件



# 新建第一个CoDeSys工程

➢ **CoDeSys HMI**

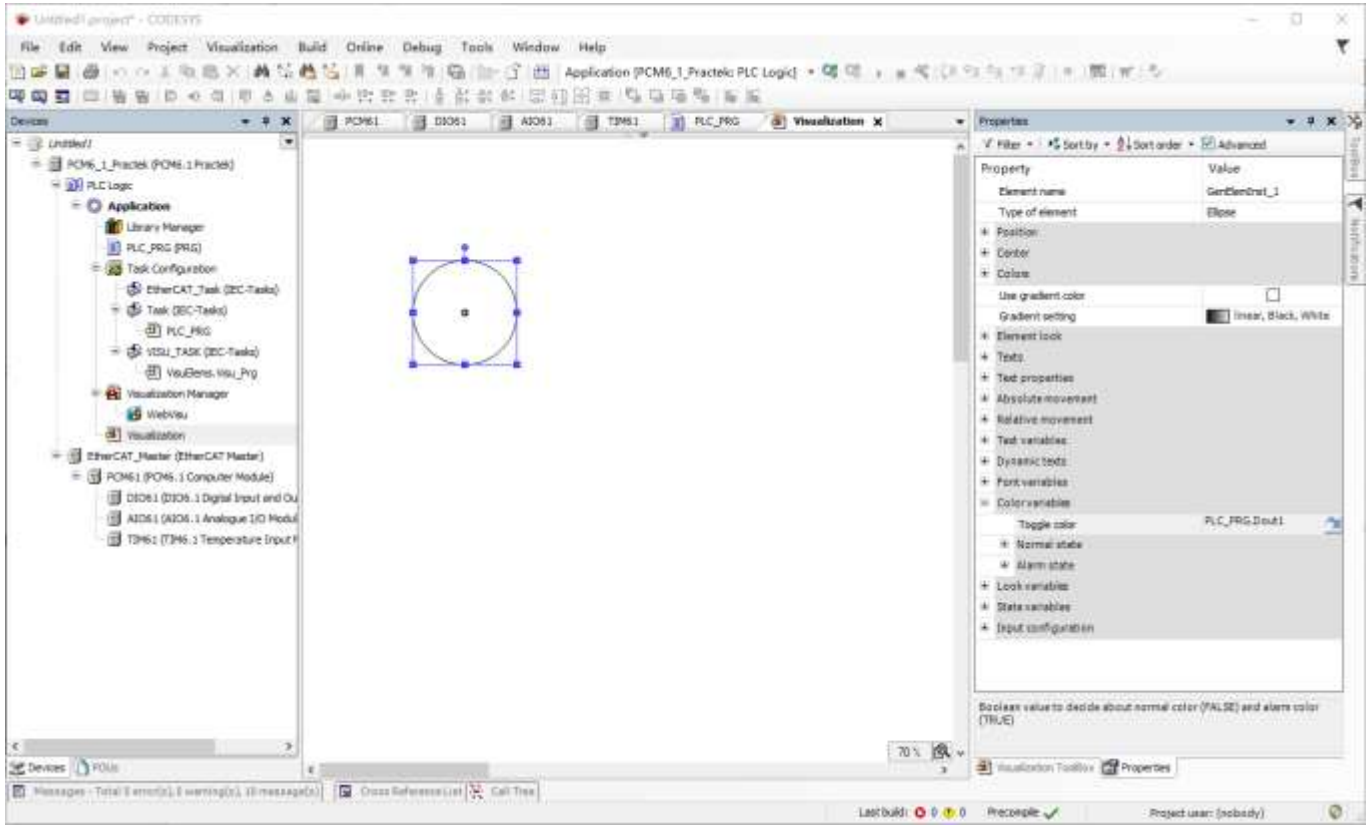
设置Eclipse控件大小属性，Width 150，Height 150。



# 新建第一个CoDeSys工程

➢ **CoDeSys HMI**

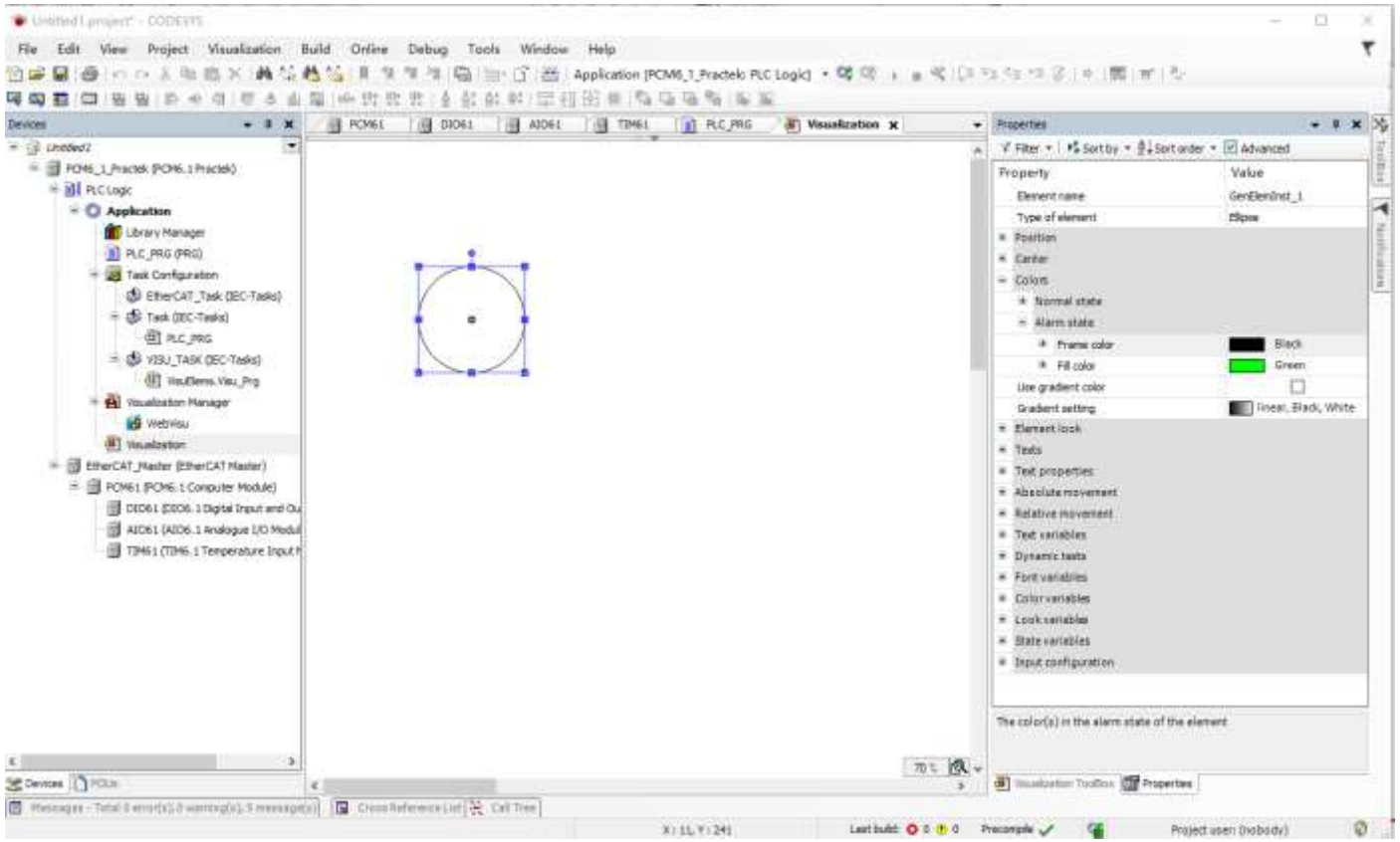
设置Toggle color，关联程序PLC_PRG中的变量Dout1。



# 新建第一个CoDeSys工程

## ➢ CoDeSys HMI

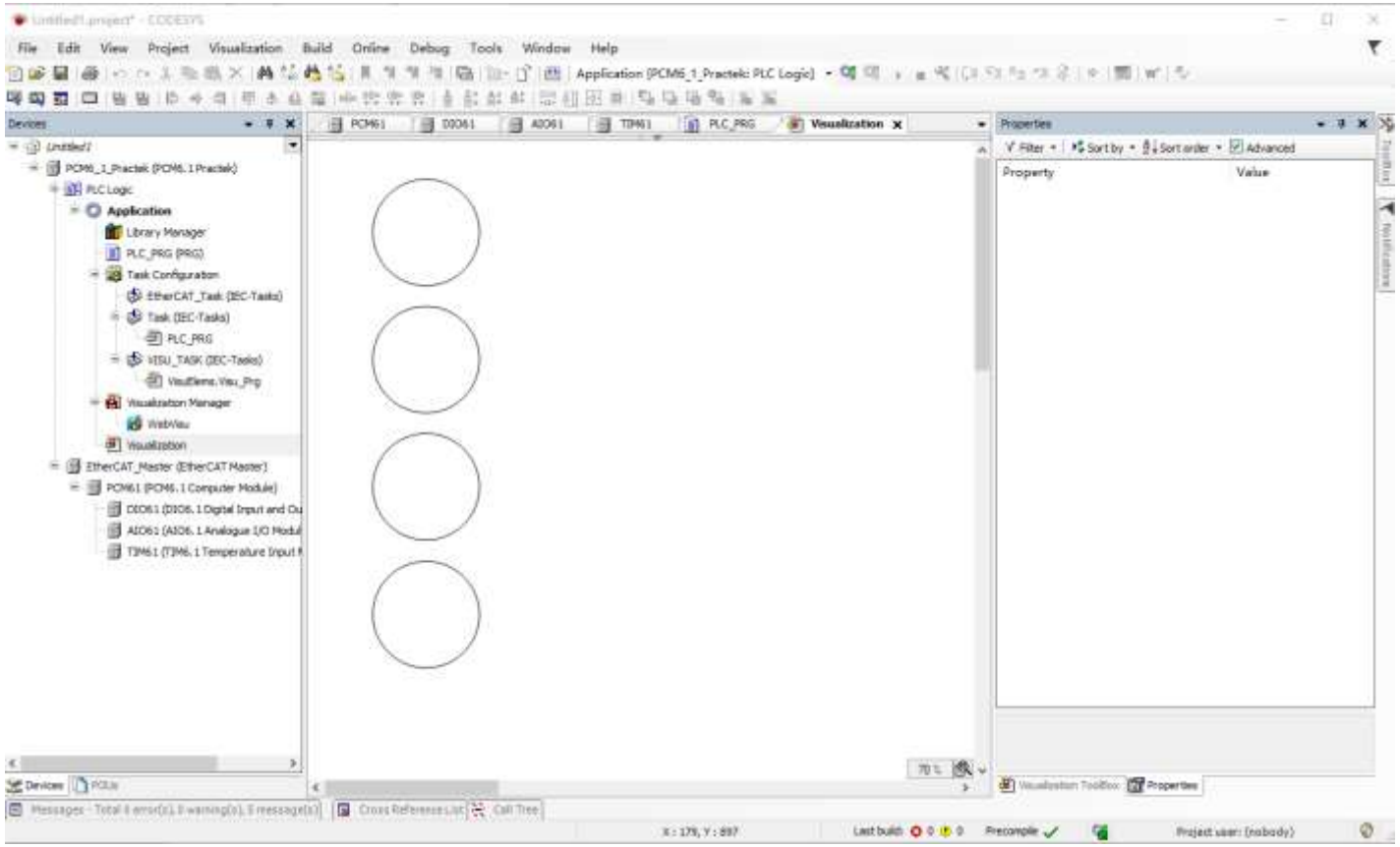
设置Colors的Alarm state。



# 新建第一个CoDeSys工程

➢ **CoDeSys HMI**

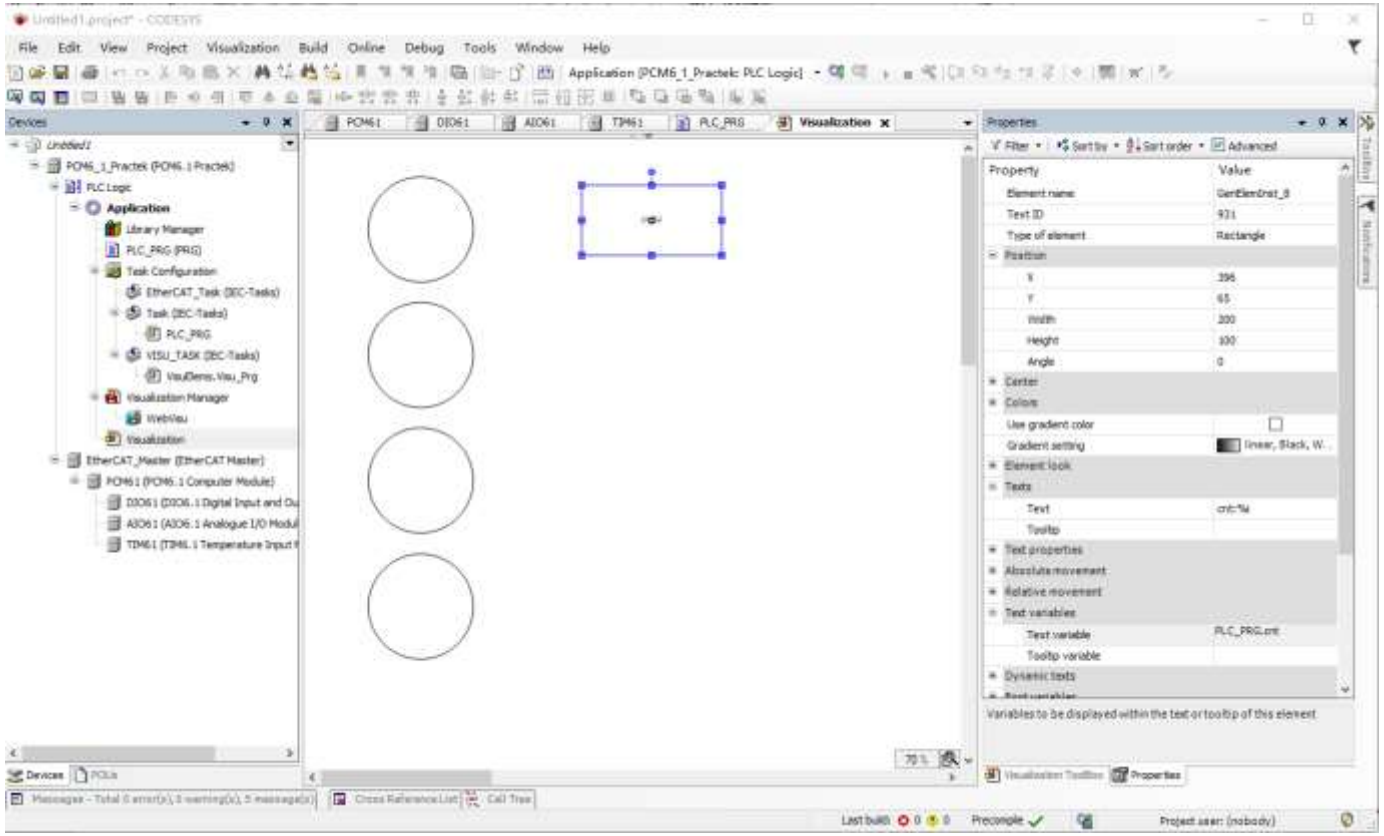
复制三个Eclipse控件，Toggle Color分别关联程序PLC_PRG中的变量Dout2、 Dout3、 Dout4。



# 新建第一个CoDeSys工程

➢ **CoDeSys HMI**

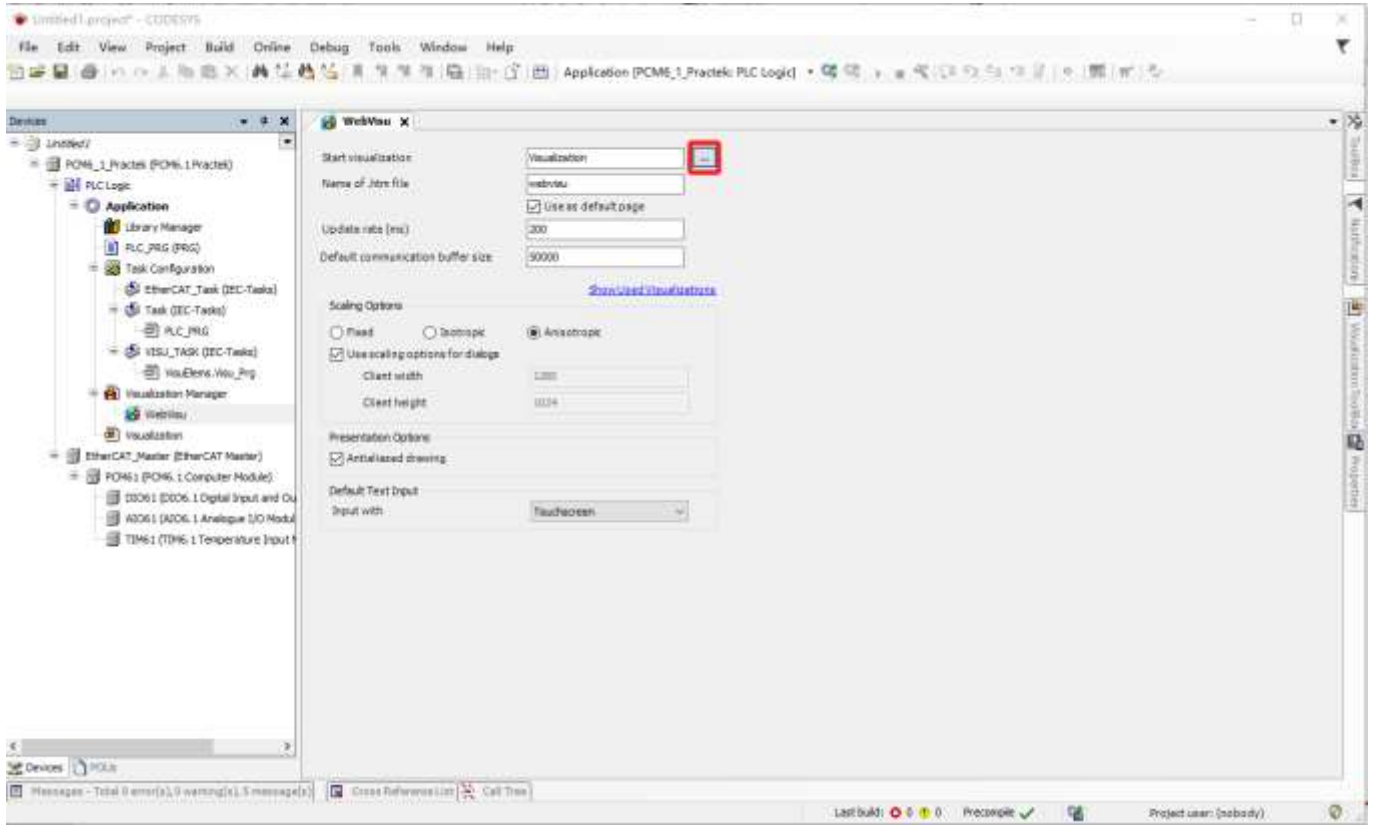
添加一个Rectangle空间，设置Position，Texts，Text variables。



# 新建第一个CoDeSys工程

## ➢ Web Visu设置

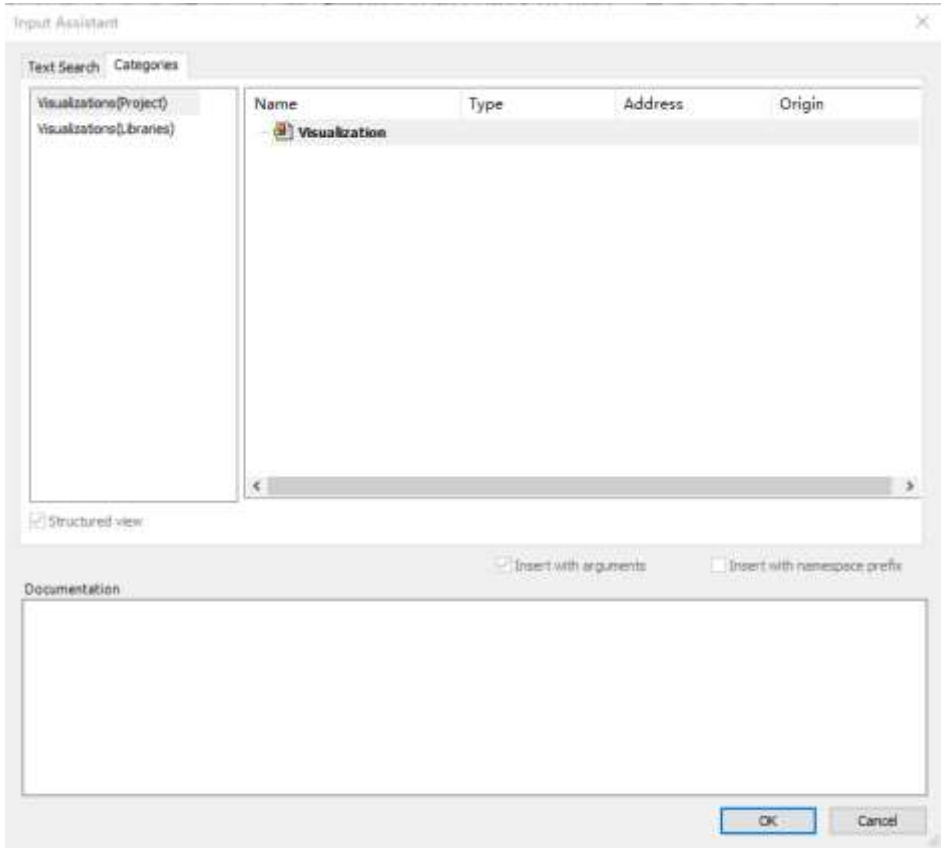
设置Start Visualization



# 新建第一个CoDeSys工程

## ➢ Web Visu设置

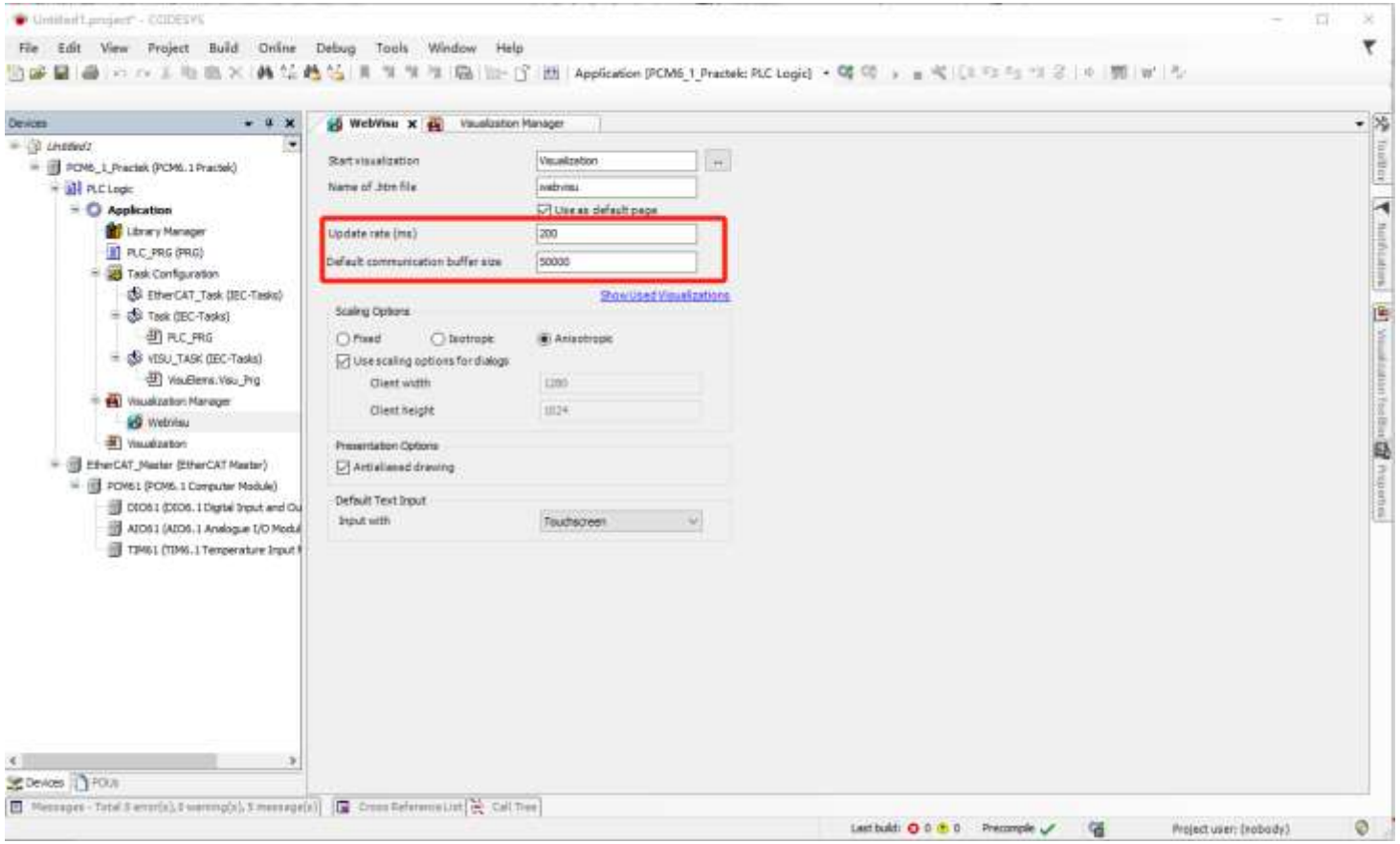
设置Start Visualization



# 新建第一个CoDeSys工程

## ➢ Web Visu设置

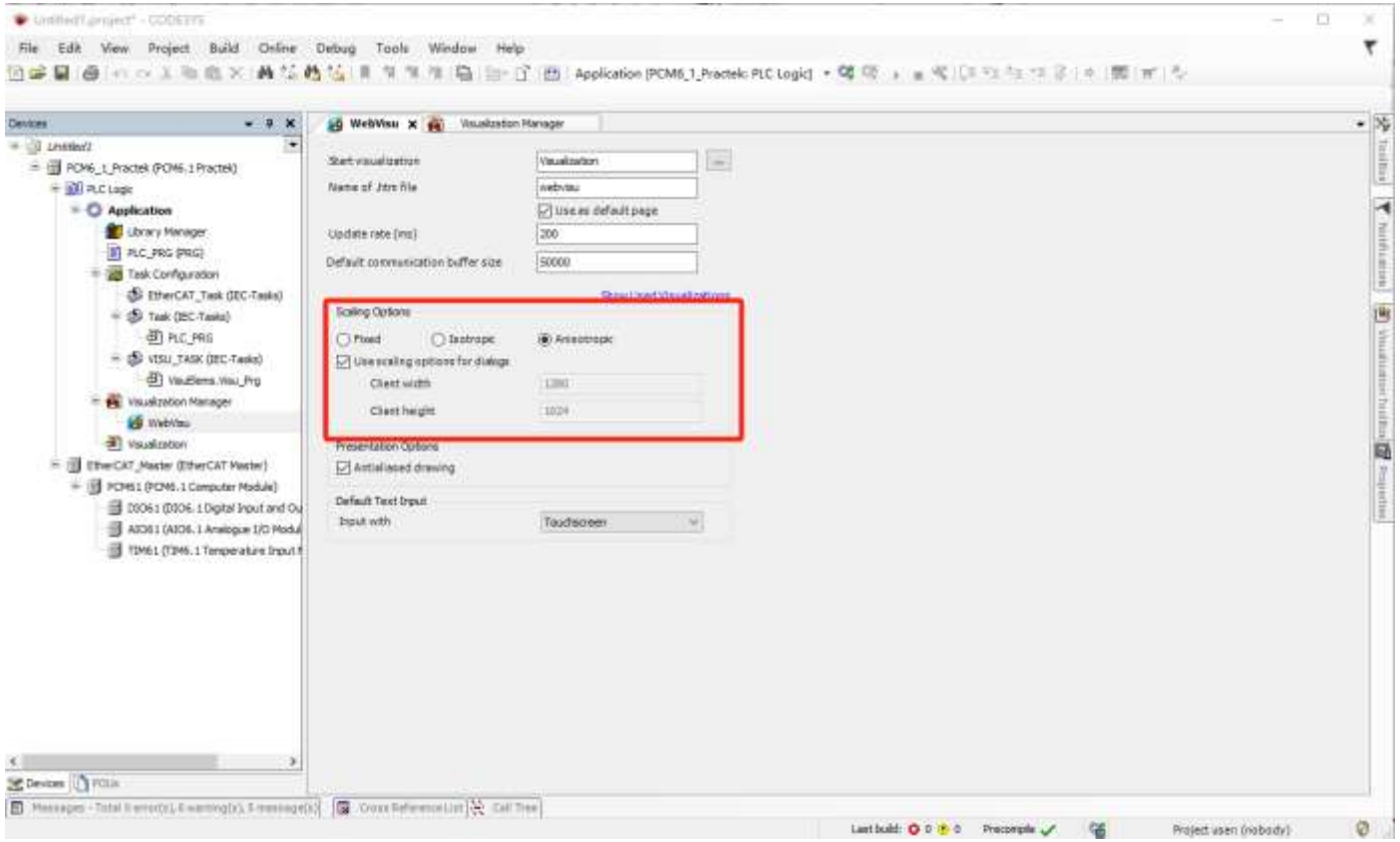
设置Update rate（ms）和Default communication buffer size。



# 新建第一个CoDeSys工程

## ➢ Web Visu设置

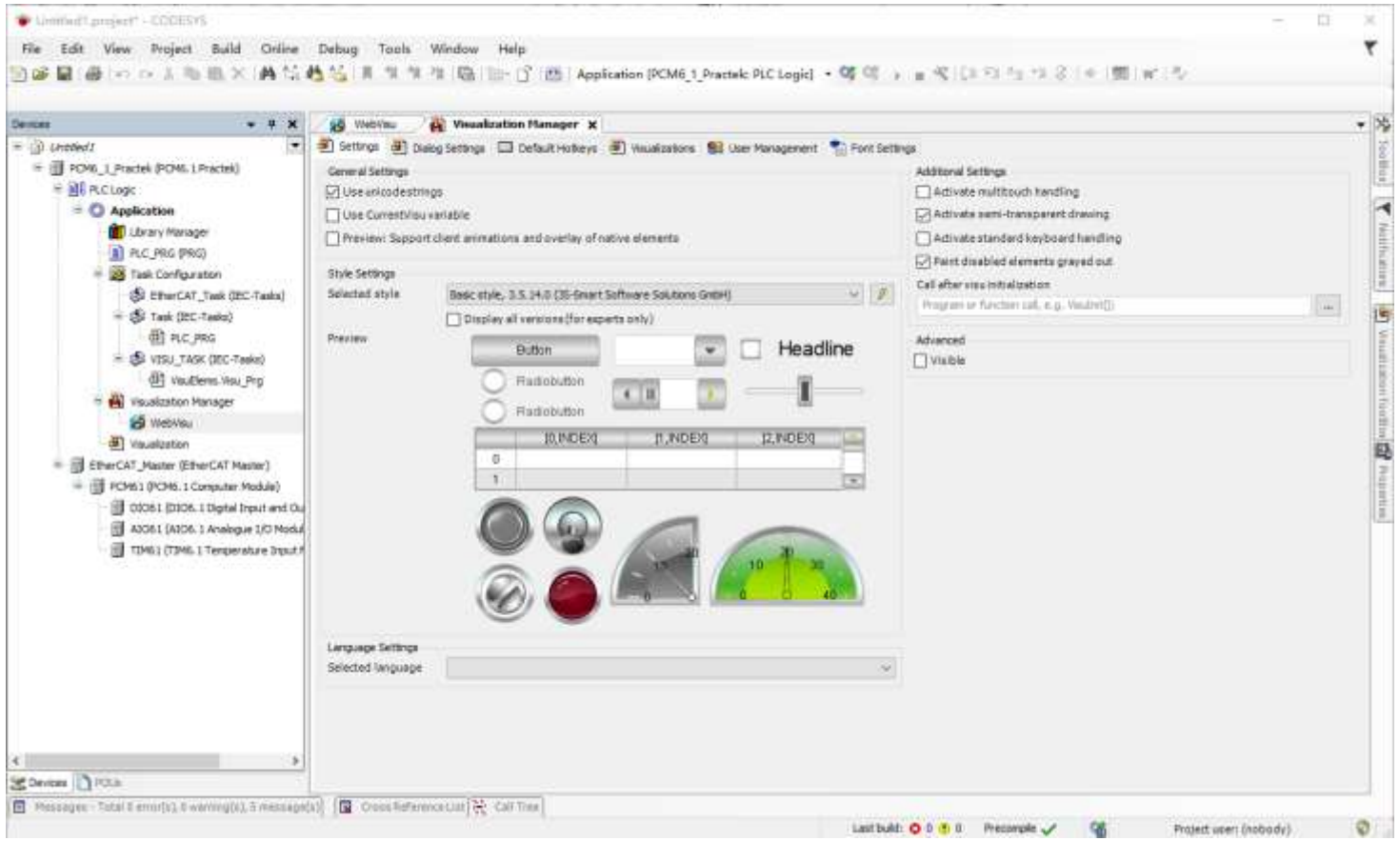
设置Scaling options,调节界面显示效果。



# 新建第一个CoDeSys工程

## ➢ Visualization Manager设置

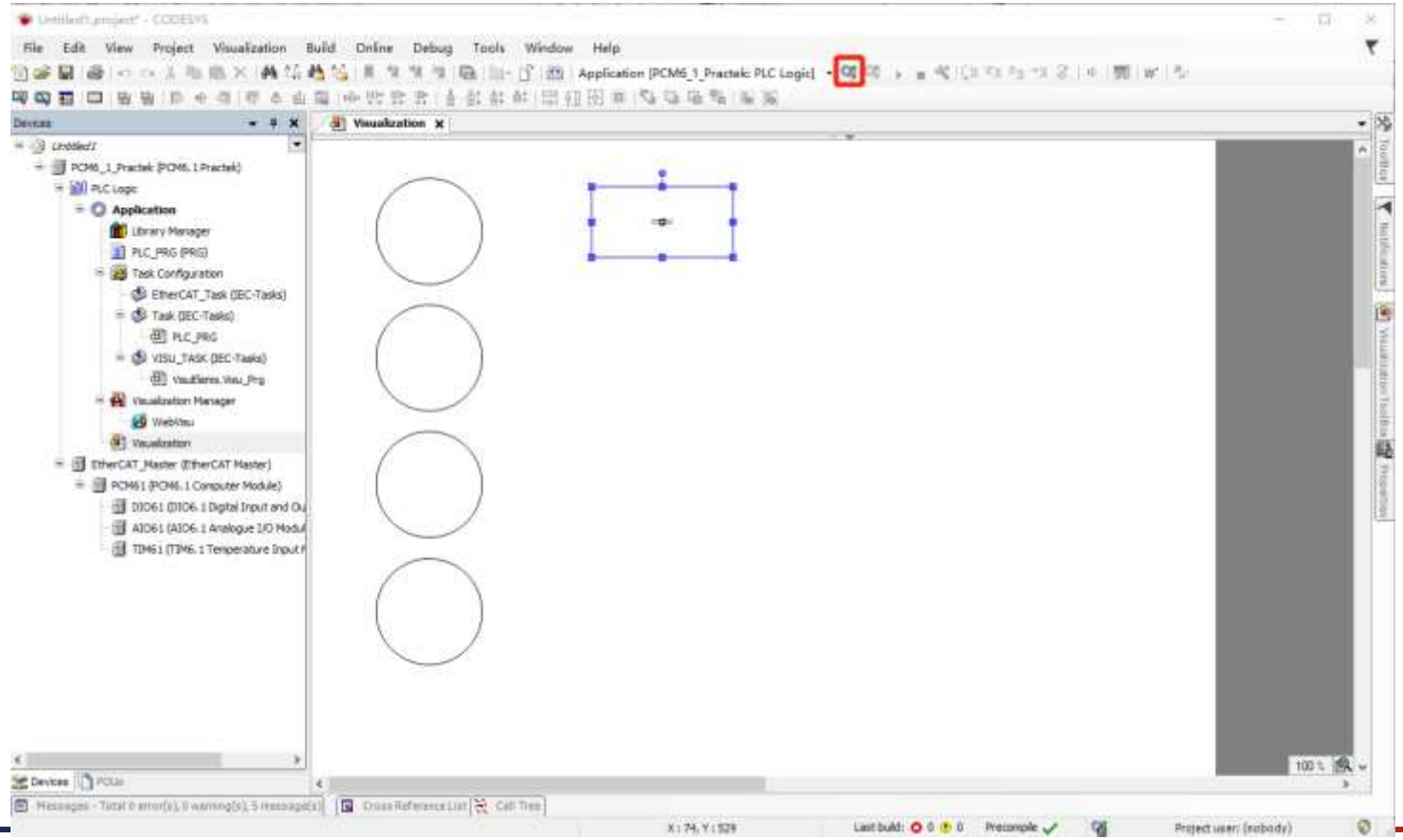
勾选Use unicode strings,可以使用unicode编码格式的字符串。



# 新建第一个CoDeSys工程

➢ **重新登录**

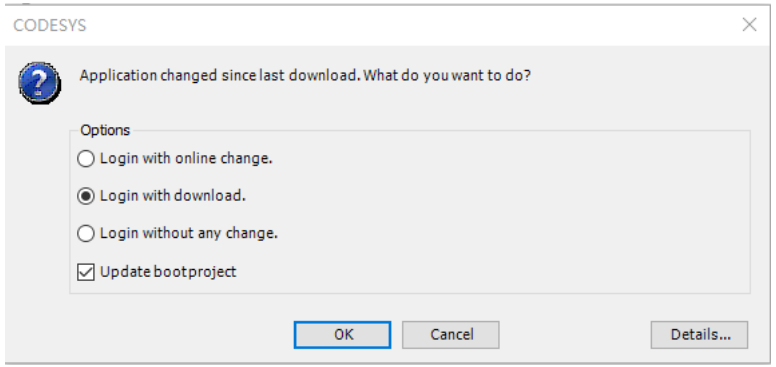
点击登录按钮。



# 新建第一个CoDeSys工程

## ➢ 重新登录

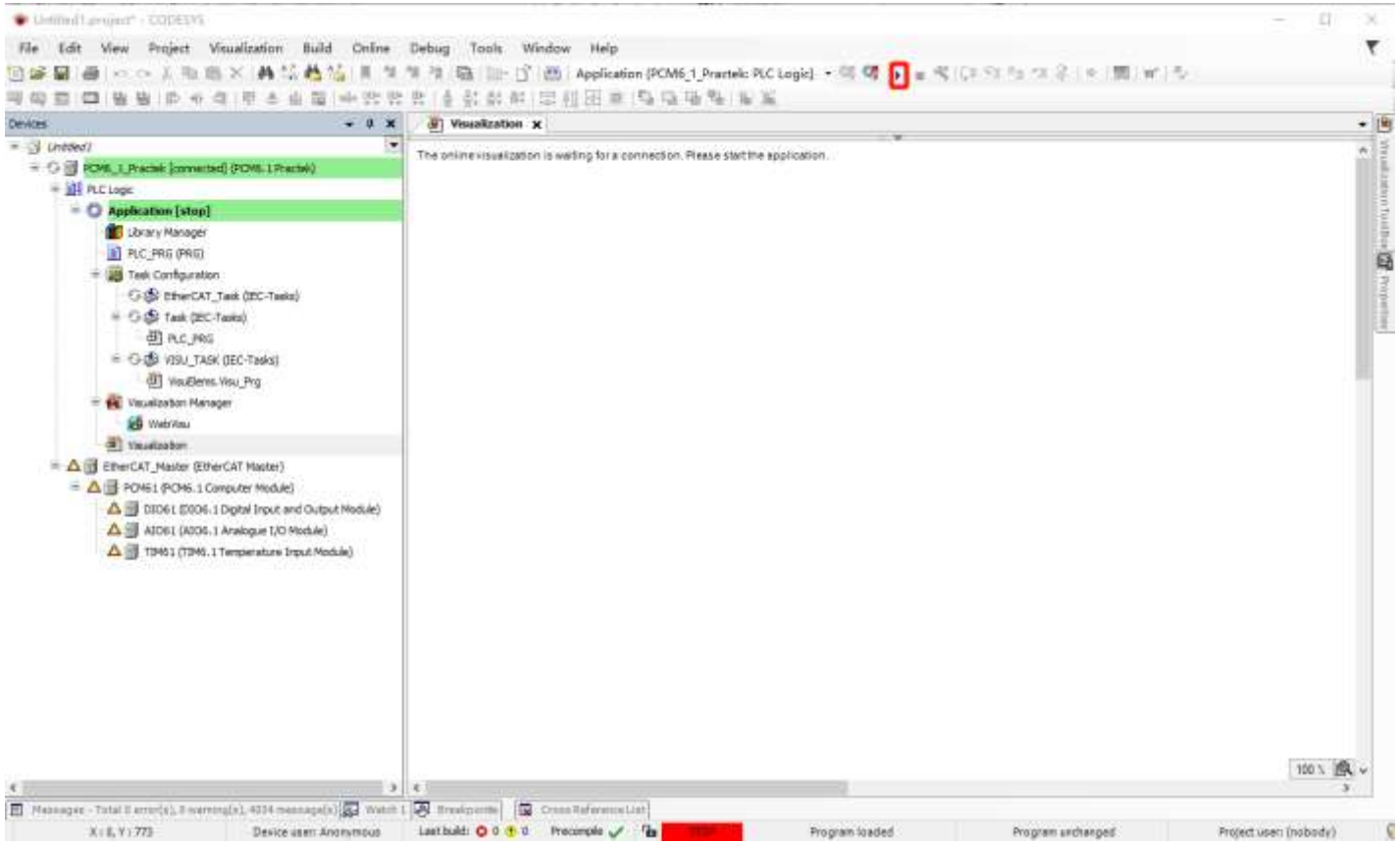
弹出窗口，选择Login with download。



# 新建第一个CoDeSys工程

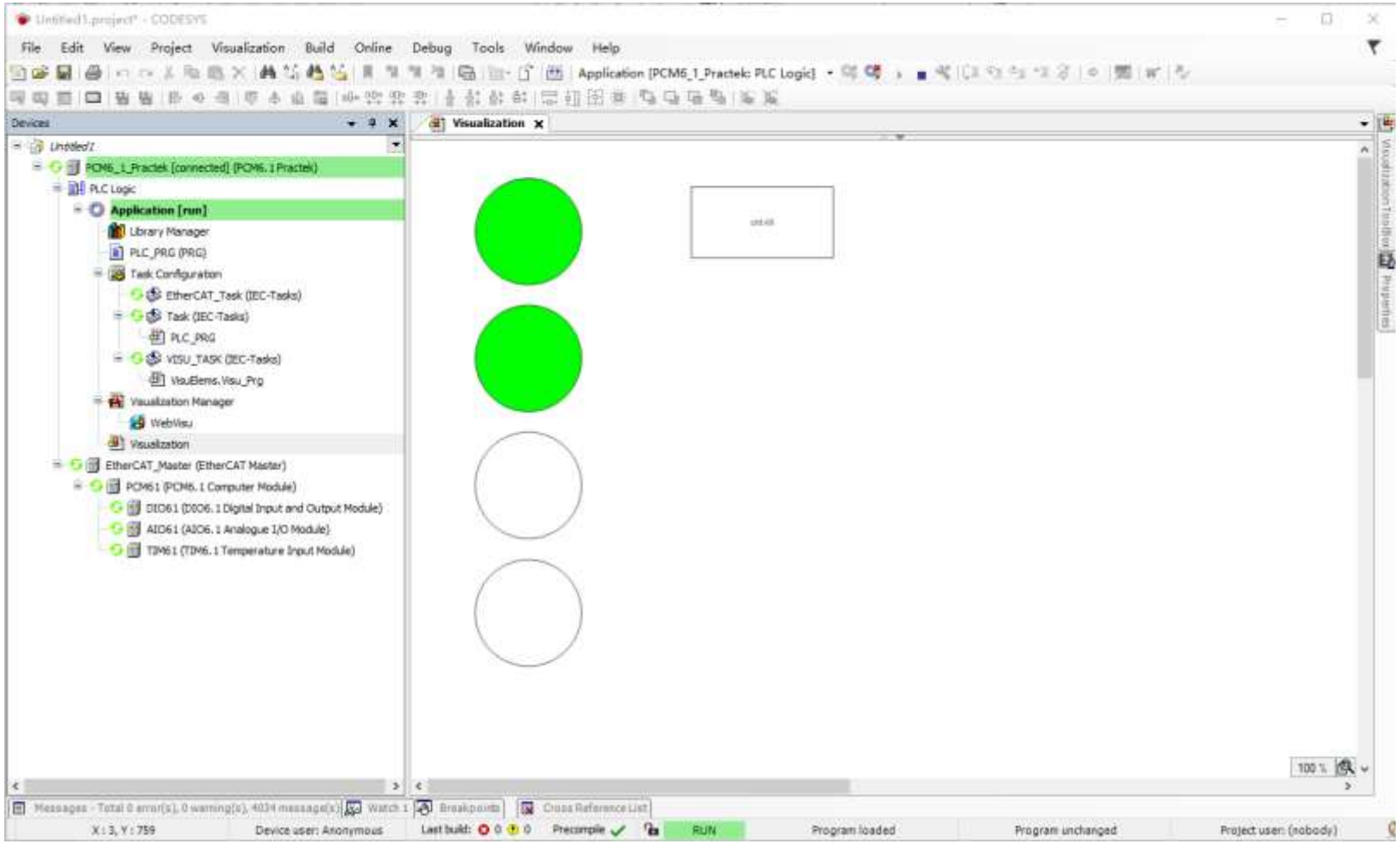
## ➢ Start程序

点击▸按钮，运行程序。



# 新建第一个CoDeSys工程

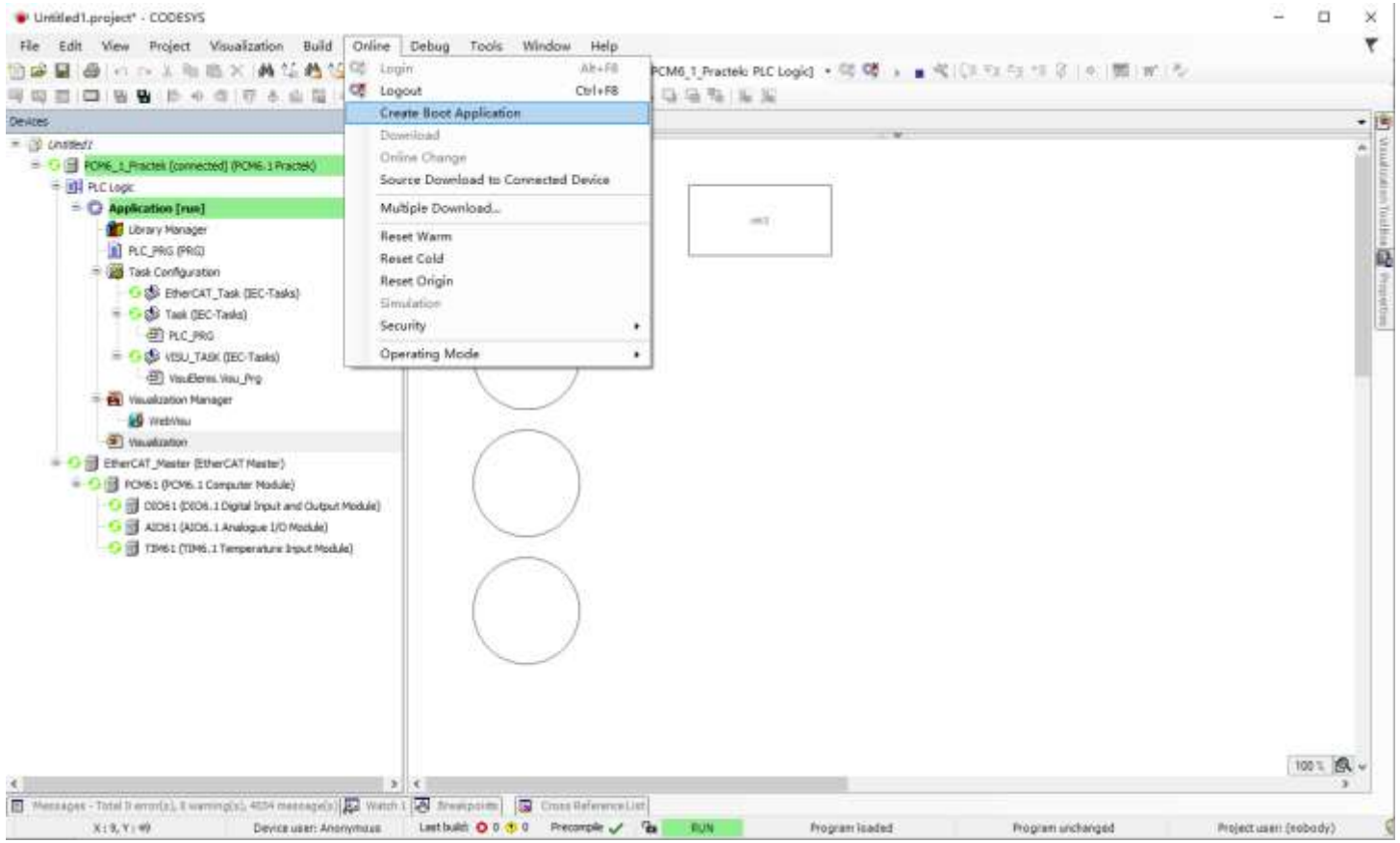
➢ **Start程序**



# 新建第一个CoDeSys工程

## ➢ 创建启动应用

点击Online -> Create Boot Application,创建启动应用，每次断电重启后，控制器自动运行应用程序。



# 新建第一个CoDeSys工程

## ➢ 创建程序dupdate文件

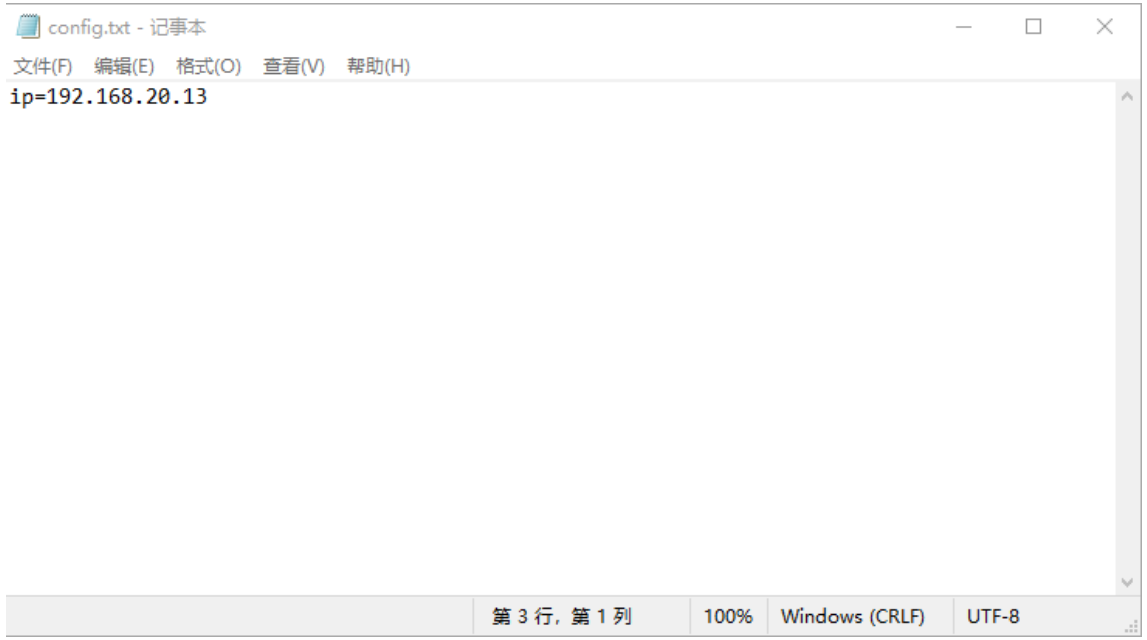
首先，程序创建启动应用，解压bootappBuilder_v9.0.0.1压缩包。

| | | | |
|---|---|---|---|
| tools | 2018/9/14 17:12 | 文件夹 | |
| config.txt | 2023/6/29 10:22 | 文本文档 | 1 KB |
| create_dupdate.bat | 2018/9/14 17:16 | Windows 批处理文件 | 1 KB |

# 新建第一个CoDeSys工程

## ➢ 创建程序dupdate文件

打开config.txt文件，将ip修改为控制器的IP，例如：192.168.20.13。



# 新建第一个CoDeSys工程

➢ **创建程序dupdate文件**

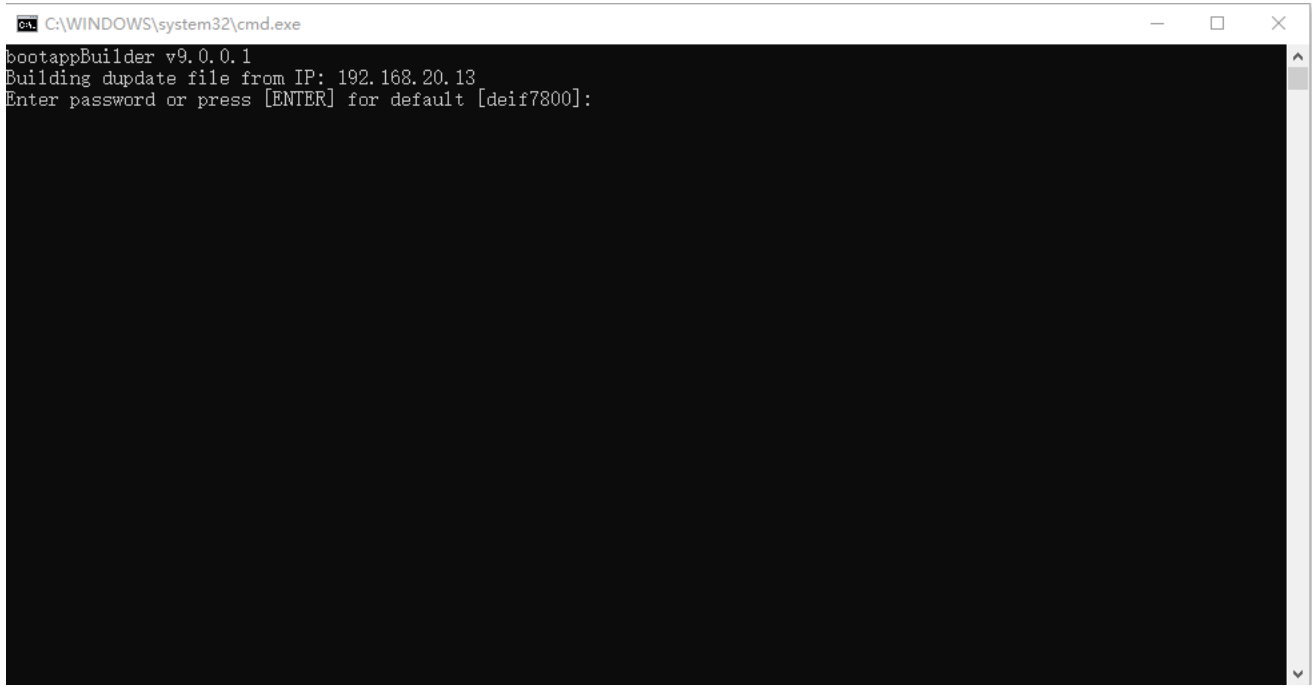
双击create_dupdate.bat文件。

| | | | |
|---|---|---|---|
| tools | 2018/9/14 17:12 | 文件夹 | |
| config.txt | 2023/12/5 16:12 | 文本文档 | 1 KB |
| create_dupdate.bat | 2018/9/14 17:16 | Windows 批处理文件 | 1 KB |

# 新建第一个CoDeSys工程

## ➢ 创建程序dupdate文件

在弹出如下窗口后，回车。



# 新建第一个CoDeSys工程

➢ **创建程序dupdate文件**

dupdate文件包生成。

| | | | |
|---|---|---|---|
| tools | 2018/9/14 17:12 | 文件夹 | |
| application_20231205_161514.dupdate | 2023/12/5 16:15 | DUPDATE 文件 | 2,562 KB |
| config.txt | 2023/12/5 16:12 | 文本文档 | 1 KB |
| create_dupdate.bat | 2018/9/14 17:16 | Windows 批处理文件 | 1 KB |

# 新建第一个CoDeSys工程

➢ **浏览器访问CodeSys Web界面**

打开浏览器，推荐火狐浏览器，在导航栏输入https://192.168.20.13:8443/webvisu.htm。

第一次连接时会弹出如下界面，点击高级。

# 新建第一个CoDeSys工程

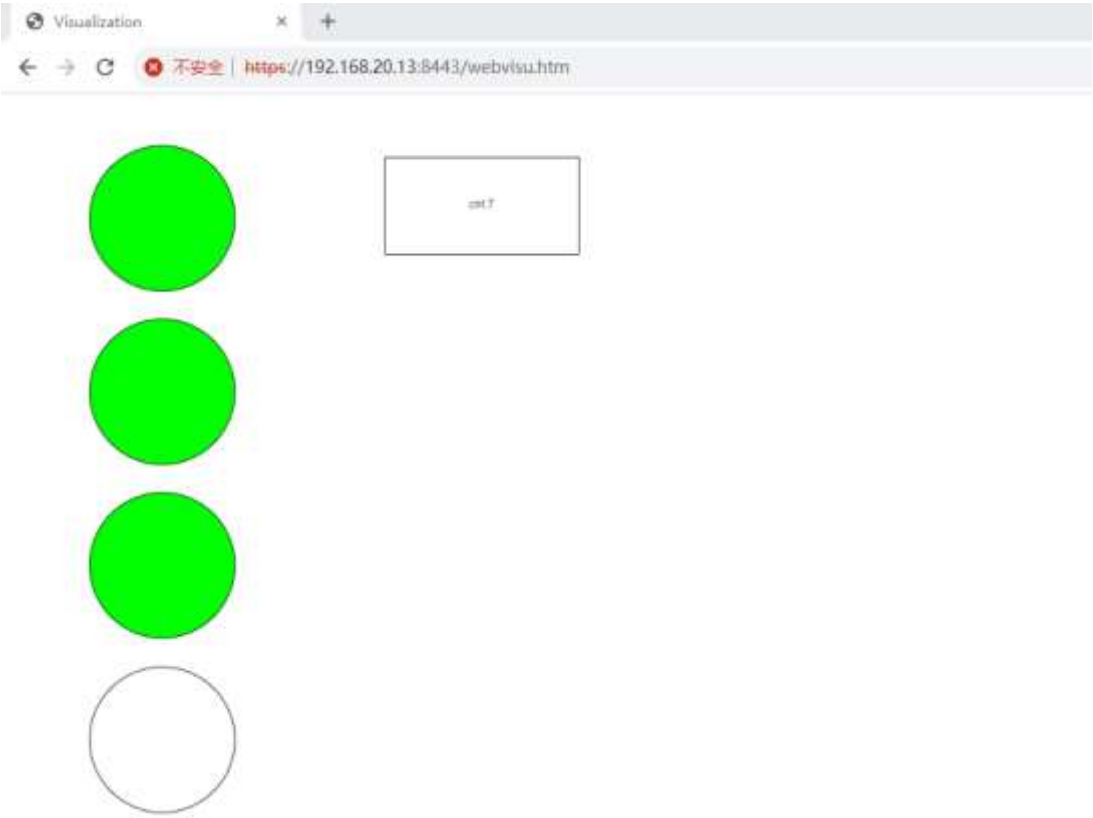
➢ **浏览器访问CodeSys Web界面**

选择继续前往192.168.20.13(不安全)。

# 新建第一个CoDeSys工程

➢ **浏览器访问CodeSys Web界面**

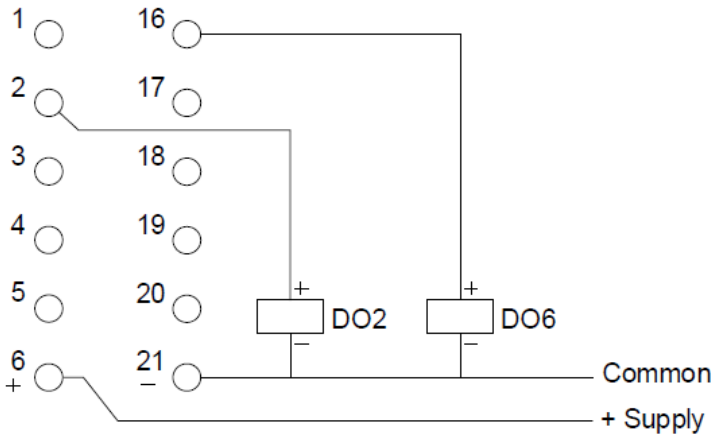
界面如下。



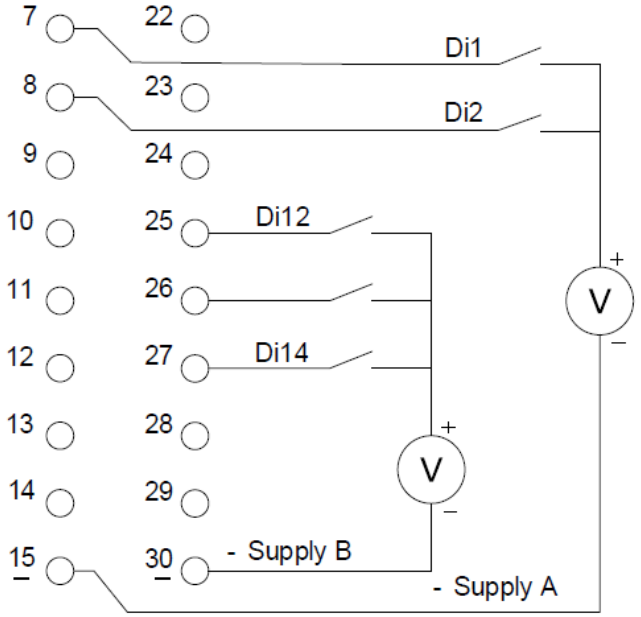
# CT65模块应用

## ➢ DIO6·1

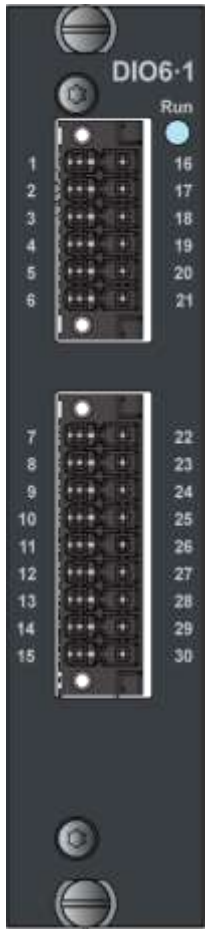
DIO6·1是数字量输入输出模块，拥有10路数字量输出通道，16路数字量输入通道。




Digital outputs

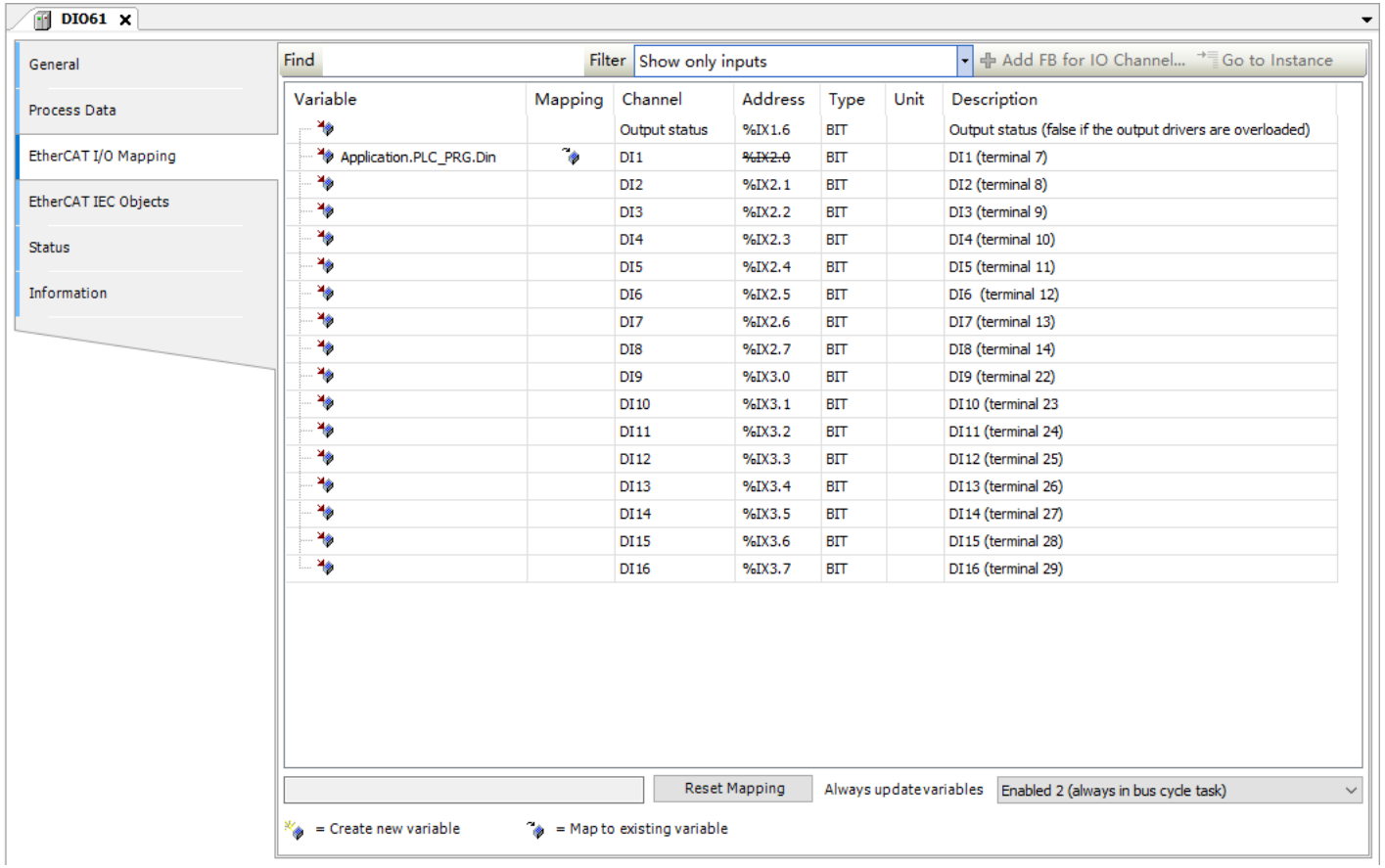



Digital inputs



# CT65模块应用

- DIO6·1 -DI
- DI通道变量映射




| Variable | Mapping | Channel | Address | Type | Unit | Description |
|---|---|---|---|---|---|---|
| | | Output status | %IX1.6 | BIT | | Output status (false if the output drivers are overloaded) |
| Application.PLC_PRG.Din | | DI1 | ~~%IX2.0~~ | BIT | | DI1 (terminal 7) |
| | | DI2 | %IX2.1 | BIT | | DI2 (terminal 8) |
| | | DI3 | %IX2.2 | BIT | | DI3 (terminal 9) |
| | | DI4 | %IX2.3 | BIT | | DI4 (terminal 10) |
| | | DI5 | %IX2.4 | BIT | | DI5 (terminal 11) |
| | | DI6 | %IX2.5 | BIT | | DI6 (terminal 12) |
| | | DI7 | %IX2.6 | BIT | | DI7 (terminal 13) |
| | | DI8 | %IX2.7 | BIT | | DI8 (terminal 14) |
| | | DI9 | %IX3.0 | BIT | | DI9 (terminal 22) |
| | | DI10 | %IX3.1 | BIT | | DI10 (terminal 23 |
| | | DI11 | %IX3.2 | BIT | | DI11 (terminal 24) |
| | | DI12 | %IX3.3 | BIT | | DI12 (terminal 25) |
| | | DI13 | %IX3.4 | BIT | | DI13 (terminal 26) |
| | | DI14 | %IX3.5 | BIT | | DI14 (terminal 27) |
| | | DI15 | %IX3.6 | BIT | | DI15 (terminal 28) |
| | | DI16 | %IX3.7 | BIT | | DI16 (terminal 29) |

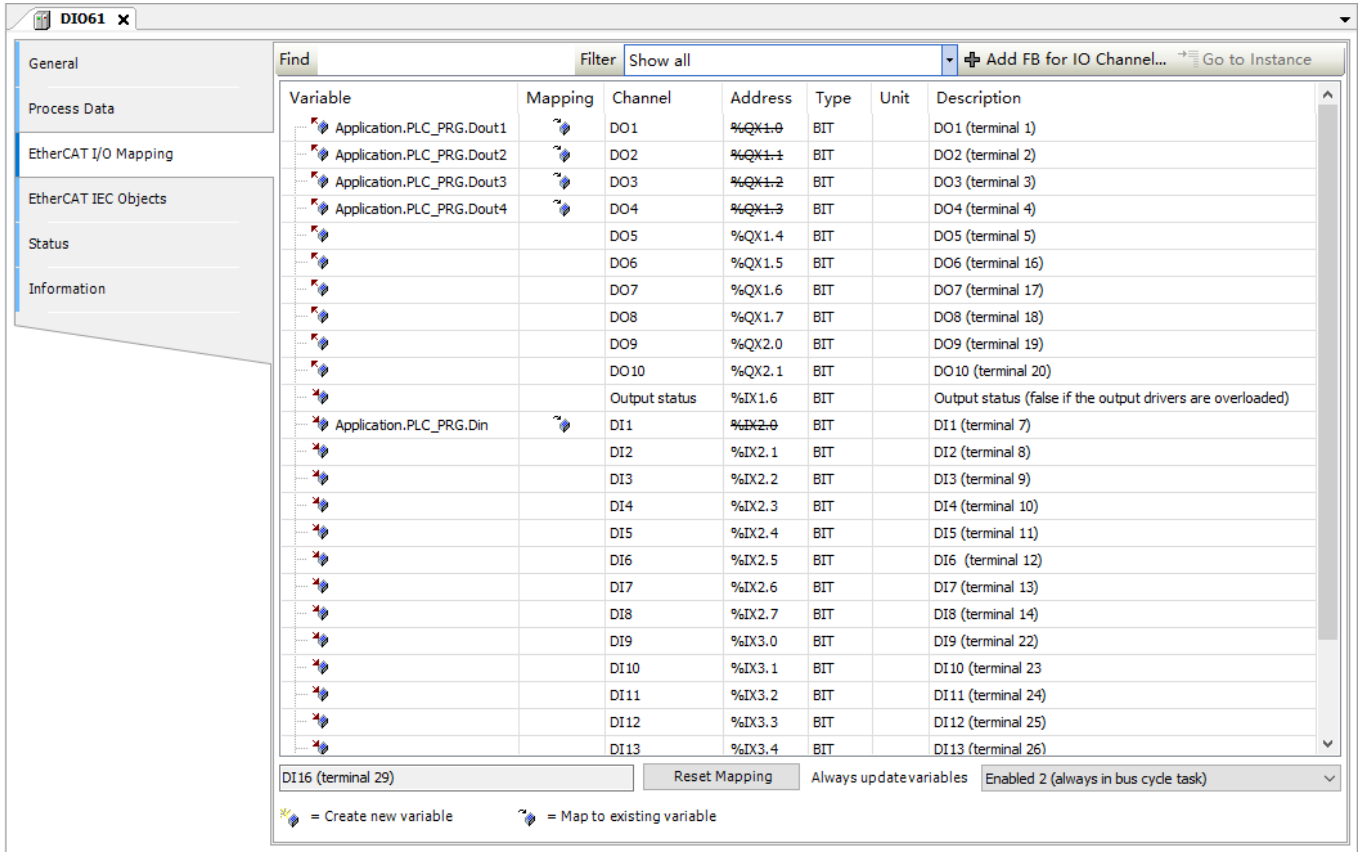
# CT65模块应用

- **DIO6·1 -DI**
- DIO6·1 DI Mapping 通道定义

| 通道名称 | 类型 | 描述 |
| --- | --- | --- |
| **DIx** | BIT | DI的数字量数值，激活状态（PNP高电平）数值为TRUE |

# CT65模块应用

- **DIO6·1 -DO**
- DIO6·1 DO 变量映射




| Variable | Mapping | Channel | Address | Type | Unit | Description |
|---|---|---|---|---|---|---|
| Application.PLC_PRG.Dout1 | | DO1 | ~~%QX1.0~~ | BIT | | DO1 (terminal 1) |
| Application.PLC_PRG.Dout2 | | DO2 | ~~%QX1.1~~ | BIT | | DO2 (terminal 2) |
| Application.PLC_PRG.Dout3 | | DO3 | ~~%QX1.2~~ | BIT | | DO3 (terminal 3) |
| Application.PLC_PRG.Dout4 | | DO4 | ~~%QX1.3~~ | BIT | | DO4 (terminal 4) |
| | | DO5 | %QX1.4 | BIT | | DO5 (terminal 5) |
| | | DO6 | %QX1.5 | BIT | | DO6 (terminal 16) |
| | | DO7 | %QX1.6 | BIT | | DO7 (terminal 17) |
| | | DO8 | %QX1.7 | BIT | | DO8 (terminal 18) |
| | | DO9 | %QX2.0 | BIT | | DO9 (terminal 19) |
| | | DO10 | %QX2.1 | BIT | | DO10 (terminal 20) |
| | | Output status | %IX1.6 | BIT | | Output status (false if the output drivers are overloaded) |
| Application.PLC_PRG.Din | | DI1 | ~~%IX2.0~~ | BIT | | DI1 (terminal 7) |
| | | DI2 | %IX2.1 | BIT | | DI2 (terminal 8) |
| | | DI3 | %IX2.2 | BIT | | DI3 (terminal 9) |
| | | DI4 | %IX2.3 | BIT | | DI4 (terminal 10) |
| | | DI5 | %IX2.4 | BIT | | DI5 (terminal 11) |
| | | DI6 | %IX2.5 | BIT | | DI6 (terminal 12) |
| | | DI7 | %IX2.6 | BIT | | DI7 (terminal 13) |
| | | DI8 | %IX2.7 | BIT | | DI8 (terminal 14) |
| | | DI9 | %IX3.0 | BIT | | DI9 (terminal 22) |
| | | DI10 | %IX3.1 | BIT | | DI10 (terminal 23 |
| | | DI11 | %IX3.2 | BIT | | DI11 (terminal 24) |
| | | DI12 | %IX3.3 | BIT | | DI12 (terminal 25) |
| | | DI13 | %IX3.4 | BIT | | DI13 (terminal 26) |
| DI16 (terminal 29) | | | | | | |

# CT65模块应用

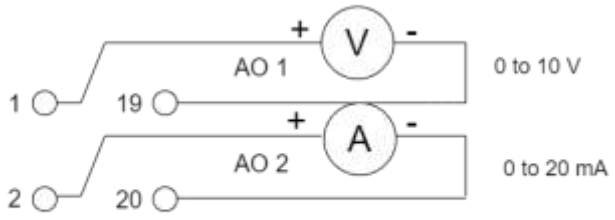
- **DIO6·1 -DO**
- DIO6·1 DO Mapping 通道定义

| 通道名称 | 类型 | 描述 |
| --- | --- | --- |
| **DOx** | BIT | DO的数字量数值，激活状态（PNP高电平）数值为TRUE |
| **Output status** | BIT | FALSE值表示DO输出过载，典型的过载工况为短路 |

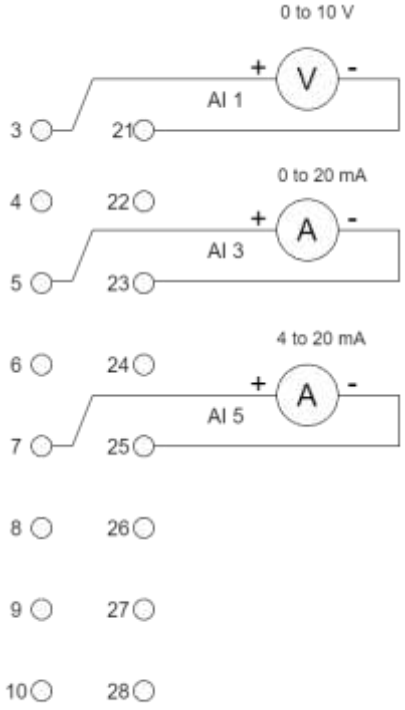
# CT65模块应用

## ➢ AIO6·1

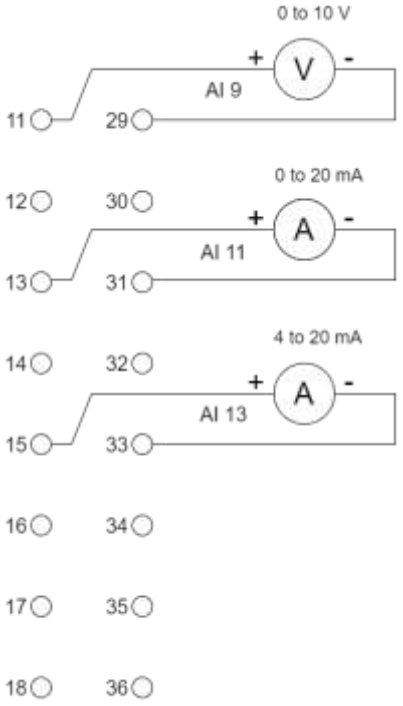
AIO6·1是模拟量输入输出模块，拥有2路模拟量输出通道(0至20 mA / 4至20 mA / 0至10 V)，16路模拟量输入通道(0至20 mA / 4至20 mA / 0至10 V )。



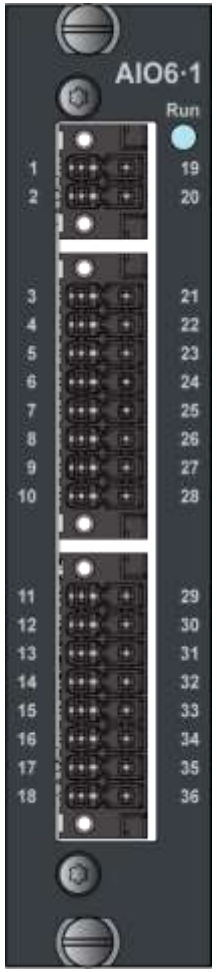

Analogue Outputs




Analogue Input 1 - 8



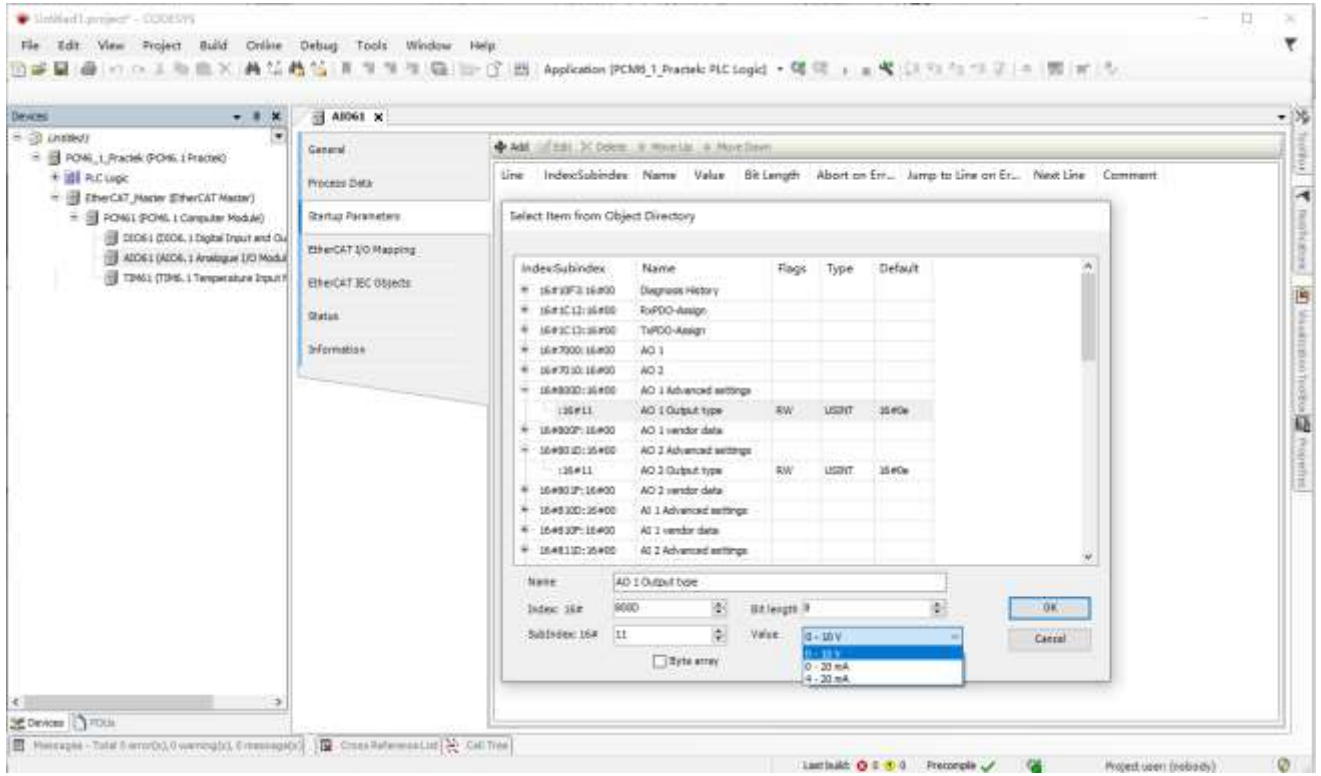

Analogue Input 9 - 16



# CT65模块应用

- **AIO6·1 – AO**
- AO通道

AIO6·1拥有2路AO通道，可以通过启动参数设置为0至20 mA / 4至20 mA / 0至10 V。



# CT65模块应用

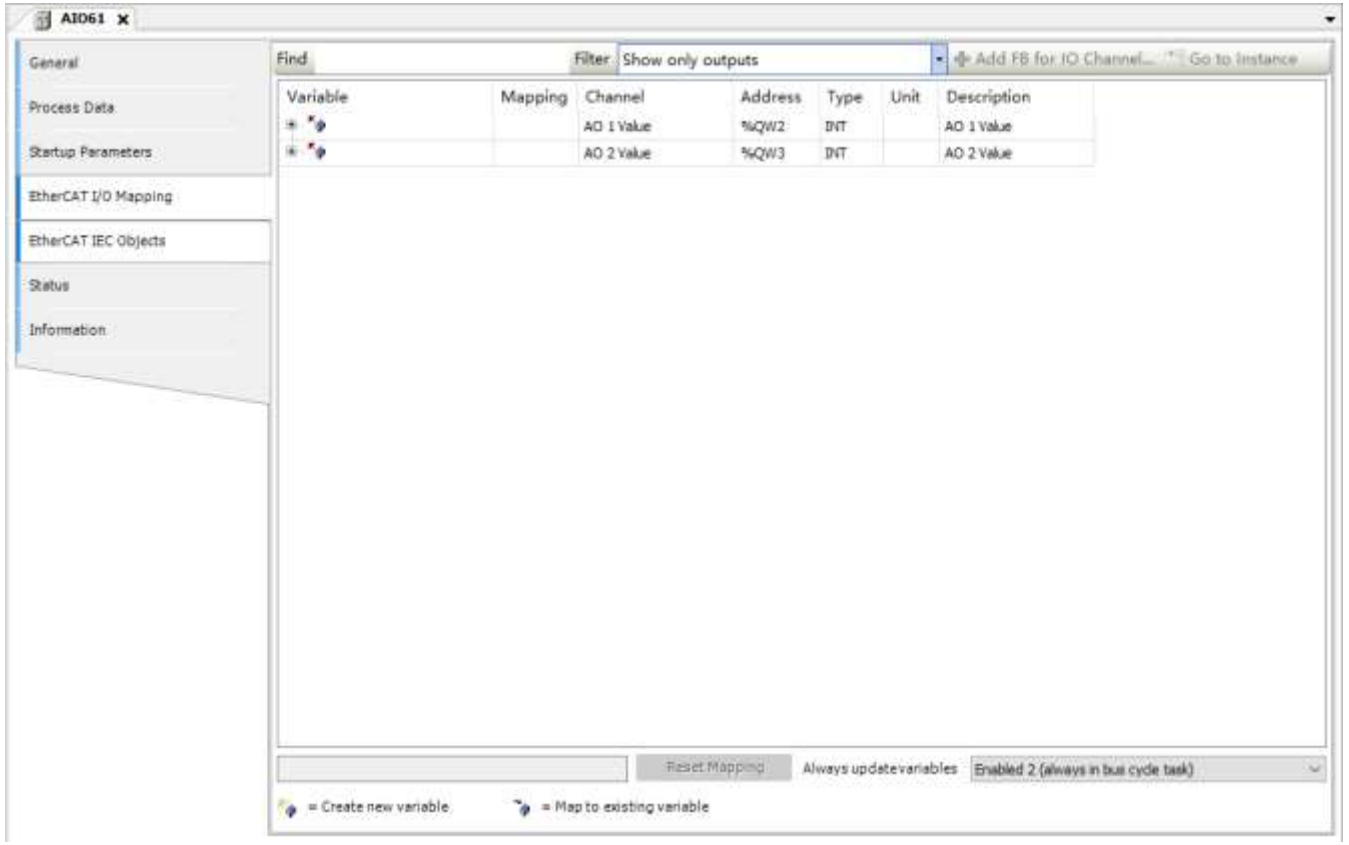
- **AIO6·1 - AO**
- AO通道

  启动参数定义

| 名称 | 类型 | 枚举名 | 值 |
|---|---|---|---|
| **AO x Output type** | USINT | 0-10V | 14 |
| | | 0-20mA | 18 |
| | | 4-20mA | 19 |

# CT65模块应用

- **AIO6·1 – AO**
- AO通道变量映射



# CT65模块应用

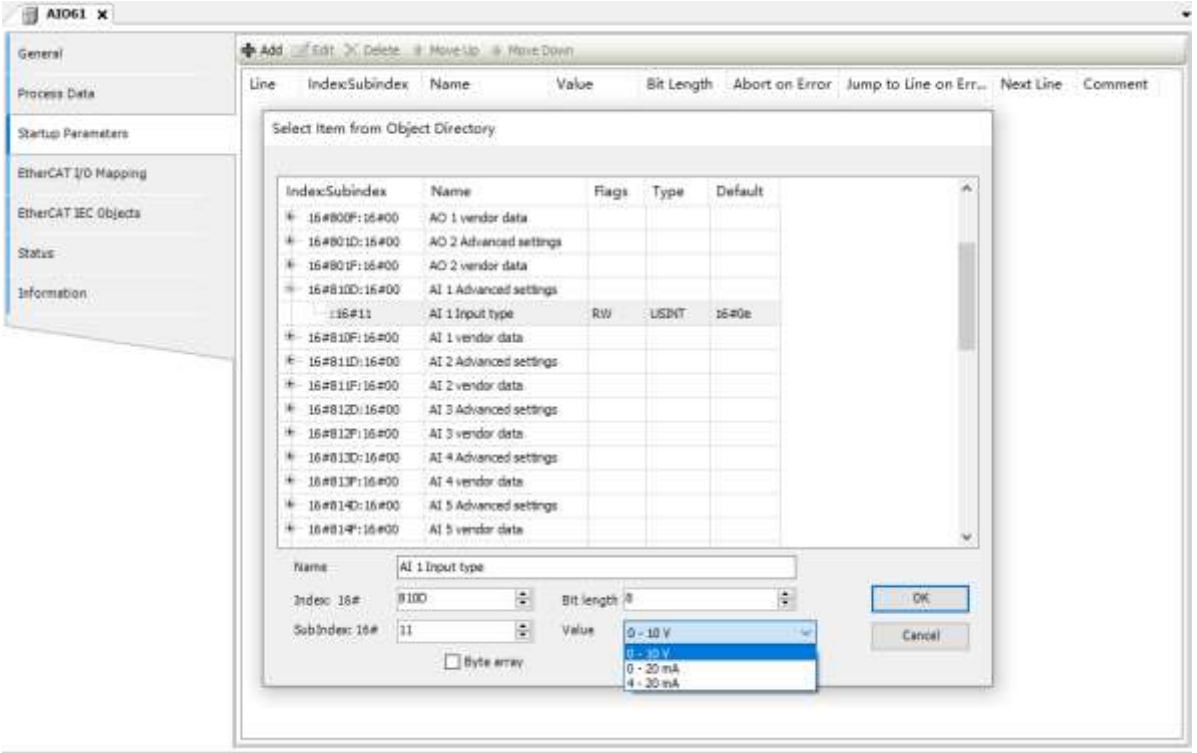
- **AIO6·1 - AO**
- AO通道变量映射

| 通道名称 | 类型 | 描述 |
| --- | --- | --- |
| **AO value** | INT | AO的模拟量数值，根据设定参数不同，数值0~32767线性对应全量程0~10V、0~20mA、4~20mA。 |

# CT65模块应用

- **AIO6·1 – AI**
- AI通道

AIO6·1拥有16路AI通道，可以通过启动参数设置为0至20 mA / 4至20 mA / 0至10 V。



# CT65模块应用

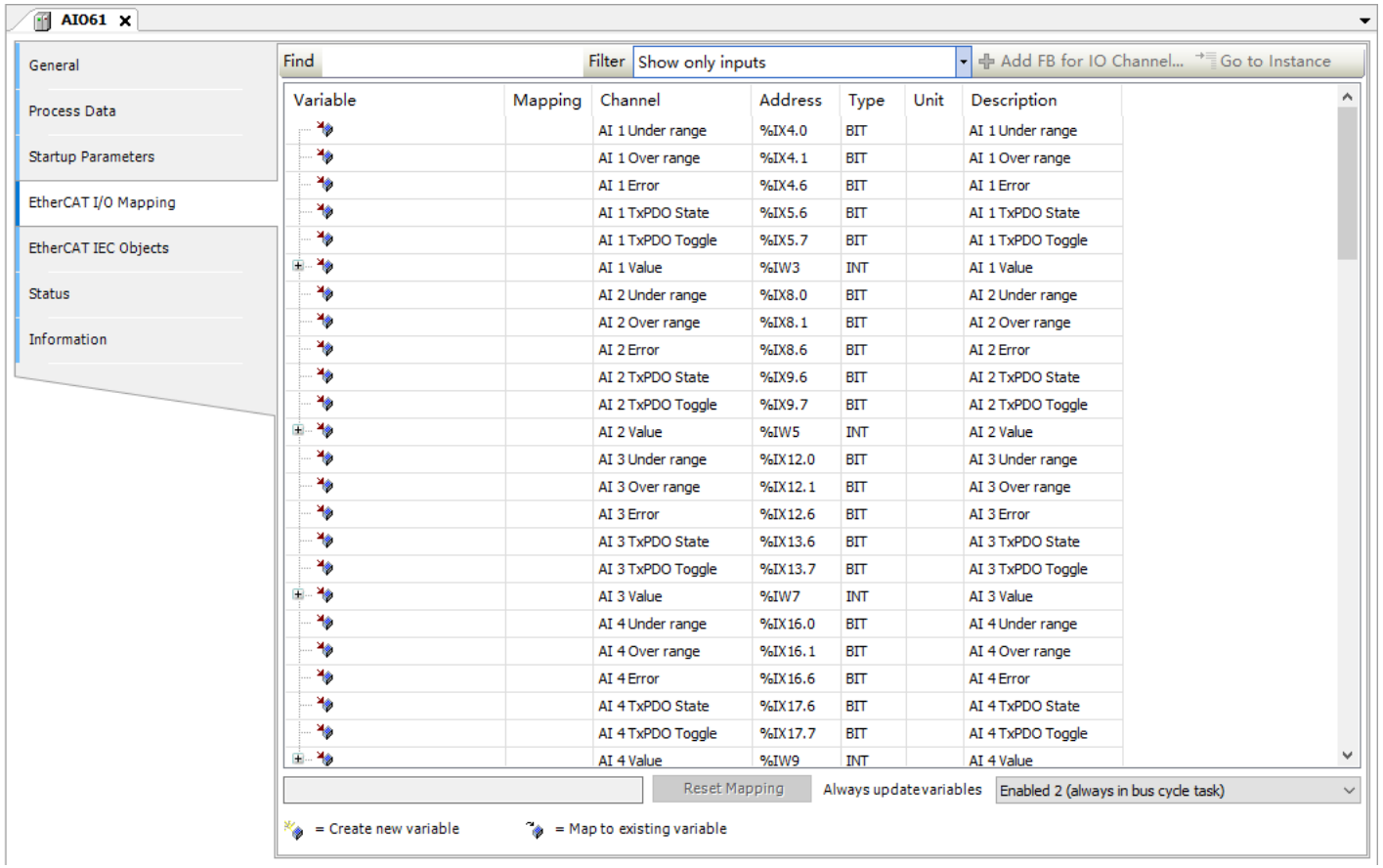
- **AIO6·1 - AI**
- AI通道

  启动参数定义。

| 名称 | 类型 | 枚举名 | 值 |
|---|---|---|---|
| **AI Input type** | USINT | 0-10V | 14 |
| | | 0-20mA | 18 |
| | | 4-20mA | 19 |

# CT65模块应用

- AIO6·1 – AI
- AI通道变量映射




| Variable | Mapping | Channel | Address | Type | Unit | Description |
|---|---|---|---|---|---|---|
| | | AI 1 Under range | %IX4.0 | BIT | | AI 1 Under range |
| | | AI 1 Over range | %IX4.1 | BIT | | AI 1 Over range |
| | | AI 1 Error | %IX4.6 | BIT | | AI 1 Error |
| | | AI 1 TxPDO State | %IX5.6 | BIT | | AI 1 TxPDO State |
| | | AI 1 TxPDO Toggle | %IX5.7 | BIT | | AI 1 TxPDO Toggle |
| | | AI 1 Value | %IW3 | INT | | AI 1 Value |
| | | AI 2 Under range | %IX8.0 | BIT | | AI 2 Under range |
| | | AI 2 Over range | %IX8.1 | BIT | | AI 2 Over range |
| | | AI 2 Error | %IX8.6 | BIT | | AI 2 Error |
| | | AI 2 TxPDO State | %IX9.6 | BIT | | AI 2 TxPDO State |
| | | AI 2 TxPDO Toggle | %IX9.7 | BIT | | AI 2 TxPDO Toggle |
| | | AI 2 Value | %IW5 | INT | | AI 2 Value |
| | | AI 3 Under range | %IX12.0 | BIT | | AI 3 Under range |
| | | AI 3 Over range | %IX12.1 | BIT | | AI 3 Over range |
| | | AI 3 Error | %IX12.6 | BIT | | AI 3 Error |
| | | AI 3 TxPDO State | %IX13.6 | BIT | | AI 3 TxPDO State |
| | | AI 3 TxPDO Toggle | %IX13.7 | BIT | | AI 3 TxPDO Toggle |
| | | AI 3 Value | %IW7 | INT | | AI 3 Value |
| | | AI 4 Under range | %IX16.0 | BIT | | AI 4 Under range |
| | | AI 4 Over range | %IX16.1 | BIT | | AI 4 Over range |
| | | AI 4 Error | %IX16.6 | BIT | | AI 4 Error |
| | | AI 4 TxPDO State | %IX17.6 | BIT | | AI 4 TxPDO State |
| | | AI 4 TxPDO Toggle | %IX17.7 | BIT | | AI 4 TxPDO Toggle |
| | | AI 4 Value | %IW9 | INT | | AI 4 Value |

# CT65模块应用

- **AIO6·1 – AI**
- AI通道变量映射

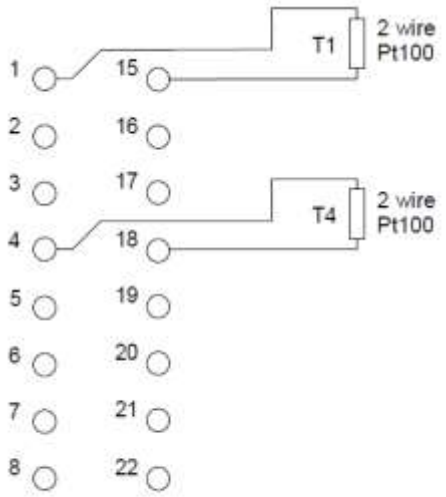
| 通道名称 | 类型 | 描述 |
|---|---|---|
| **AI x Under range** | BIT | TRUE值表示超出测量下限 |
| **AI x Over range** | BIT | TRUE值表示超出测量上限 |
| **AI x Error** | BIT | Under range或Over range触发时，AI Error触发，TRUE值表示Error |
| **AI x TxPDO State** | BIT | 备用，暂无实际意义 |
| **AI x TxPDO Toggle** | BIT | 当AI数据更新时，TRUE / FALSE跳反变化 |
| **AI x value** | INT | AI的模拟量数值，根据设定参数不同，数值0~32767线性对应全量程0~10V、0~20mA、4~20mA。 |

# CT65模块应用

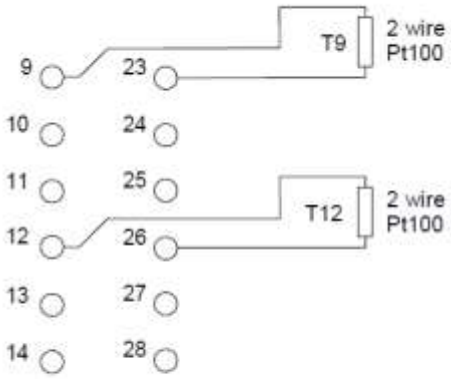
## ➢ TIM6·1

TIM6·1 是温度输入模块，可以测量14路两线PT100输入或者6路三线制PT100。

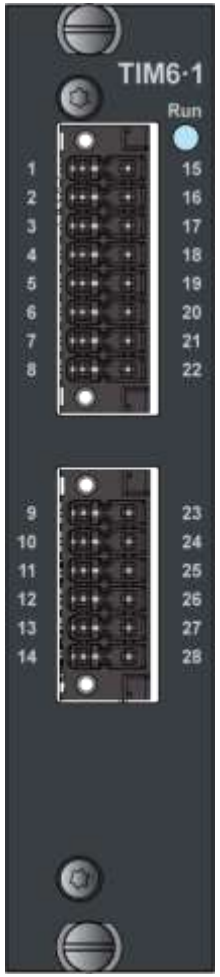
两线制接线图:




Temperature inputs 1 to 8 (Pt100)



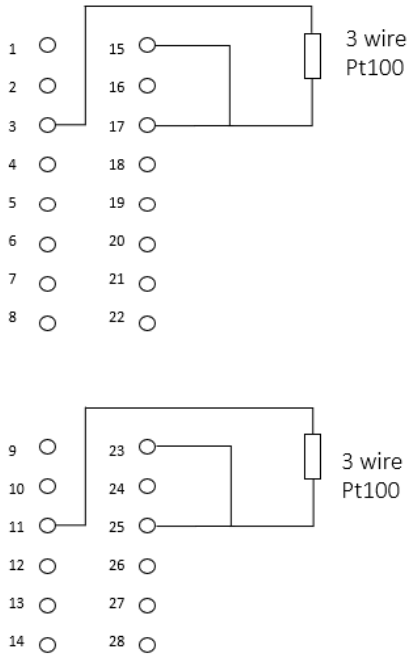

Temperature inputs 9 to 14 (Pt100)



# CT65模块应用

## ➢ TIM6·1

三线线制接线图:




三线制分组:

1/15/3/17同组，1空闲

2/16/4/18同组，2空闲

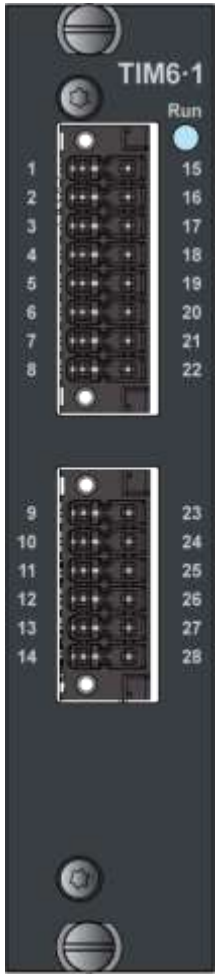
5/19/7/21同组，5空闲

6/20/8/22同组，6空闲

9/23/11/25同组，9空闲

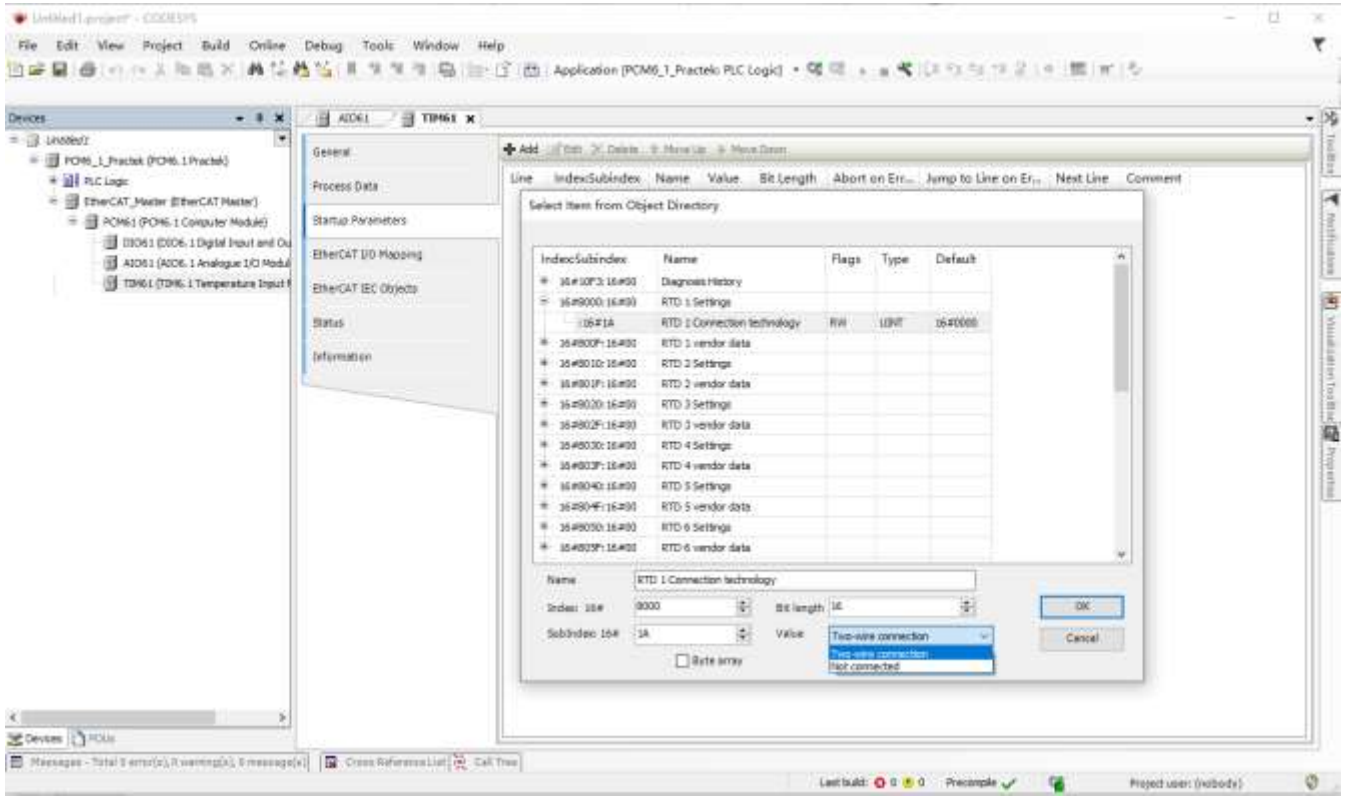
10/24/12/26同组，10空闲

Temperature input 3 wire Pt100



# CT65模块应用

- **TIM6·1 – 两线制**
- 启动参数设置



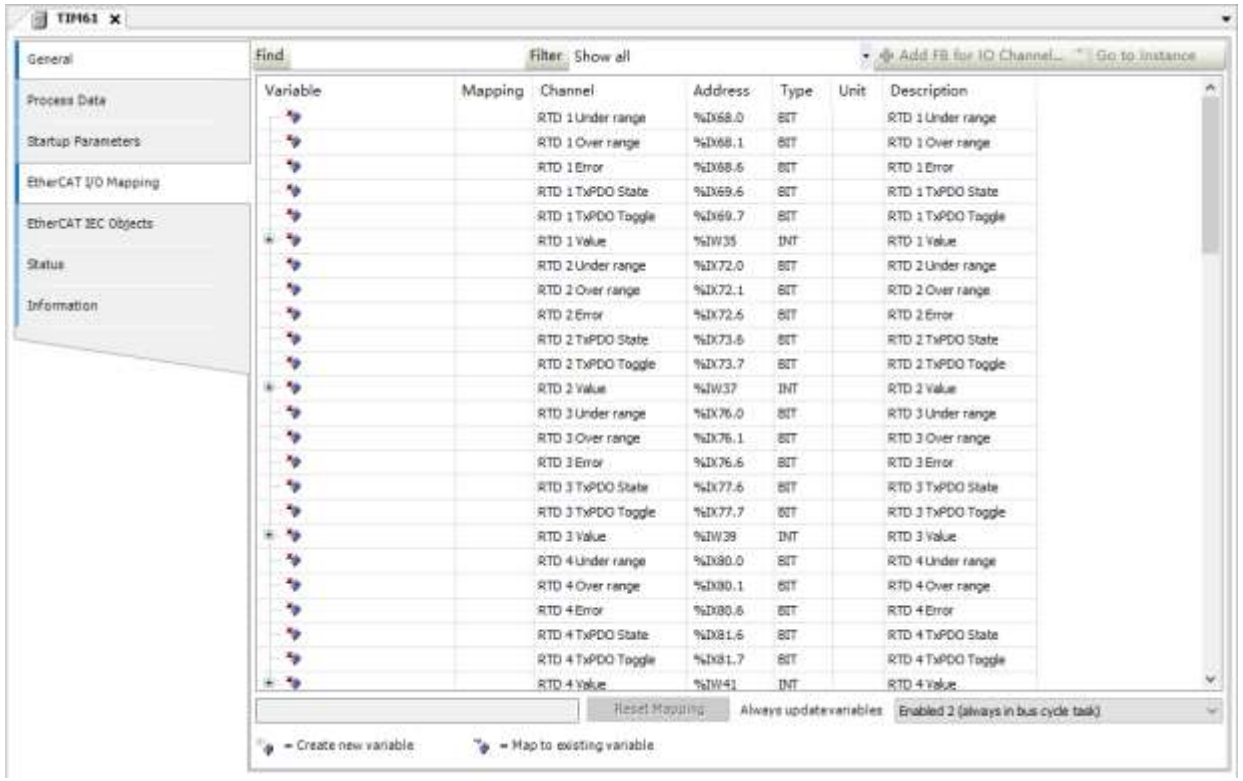
# CT65模块应用

- **TIM6·1 – 两线制**
- 启动参数设置

| 启动参数 | 类型 | 枚举值 | 值 |
|---|---|---|---|
| **RTD 1~14 Connection technology** | UINT（16位） | Two-wire connection | 0 |
| | | Not connected | 3 |

# CT65模块应用

- **TIM6·1 – 两线制**
- 变量映射



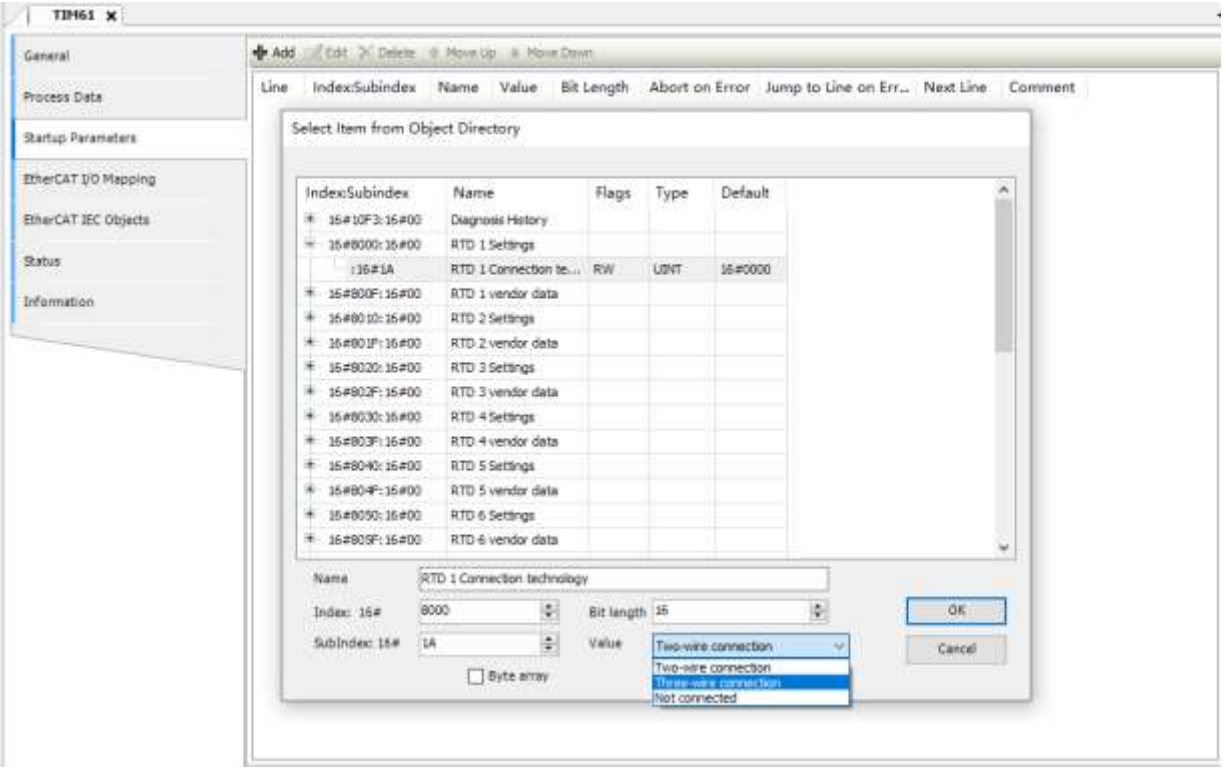
# CT65模块应用

- **TIM6·1 – 两线制**
- 变量映射

| 通道名称 | 类型 | 描述 |
|---|---|---|
| Under range | BIT | 超出测量下限，TRUE值报警 |
| Over range | BIT | 超出测量上限，TRUE值报警 |
| Error | BIT | Under range或Over range值触发时，Error值触发，TRUE值报警 |
| TxPDO State | BIT | 暂无实际意义 |
| TxPDO Toggle | BIT | 数据刷新标识，数据更新时TRUE/FALSE值翻转。 |
| Value | INT | 温度采集的值（实际值的10倍） |

# CT65模块应用

- **TIM6·1 – 三线制**
- 需要更新TIM6.1固件，设置启动参数



# CT65模块应用

- **TIM6·1 – 三线制**

- 启动参数设置

RTD 1~12 Connection technology参数可选Two-wire connection/Three-wire connection/ Not connected，对应的TEMP接线方式两线制/三线制/不使用。

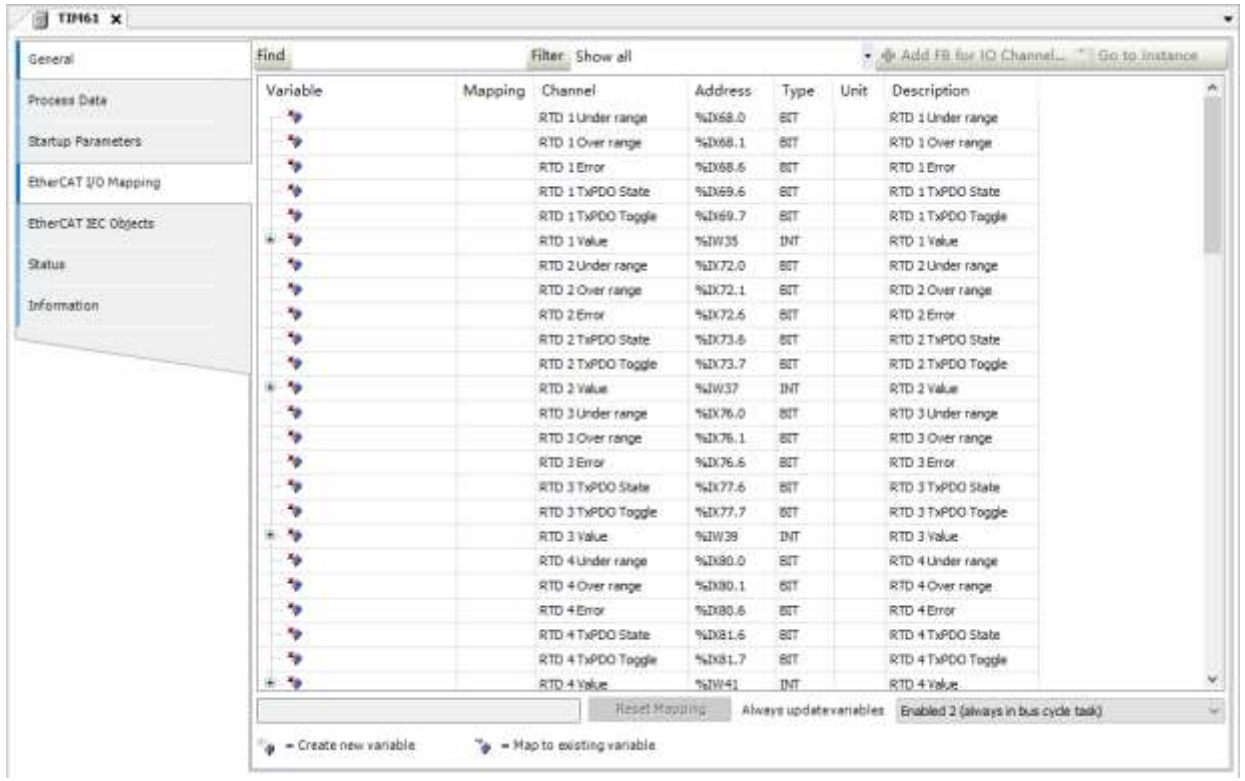
RTD 13, 14只可作为两线制TEMP使用。

当使用三线制时，固定的两路两线制合并成为一个三线制。如：使用1/15/3/17，将TEMP1和TEMP3合并为一路三线制TEMP，需要将TEMP1和TEMP3的启动参数都修改为Three-wire connection。

| 启动参数 | 类型 | 枚举值 | 值 |
|---|---|---|---|
| **RTD 1~14 Connection technology** | UINT（16位） | Two-wire connection | 0 |
| | | Three-wire connection | 1 |
| | | Not connected | 3 |

# CT65模块应用

- **TIM6·1 – 三线制**
- 变量映射




| Variable | Mapping | Channel | Address | Type | Unit | Description |
|---|---|---|---|---|---|---|
| | | RTD 1 Under range | %IX68.0 | BIT | | RTD 1 Under range |
| | | RTD 1 Over range | %IX68.1 | BIT | | RTD 1 Over range |
| | | RTD 1 Error | %IX68.6 | BIT | | RTD 1 Error |
| | | RTD 1 TxPDO State | %IX69.6 | BIT | | RTD 1 TxPDO State |
| | | RTD 1 TxPDO Toggle | %IX69.7 | BIT | | RTD 1 TxPDO Toggle |
| | | RTD 1 Value | %IW35 | INT | | RTD 1 Value |
| | | RTD 2 Under range | %IX72.0 | BIT | | RTD 2 Under range |
| | | RTD 2 Over range | %IX72.1 | BIT | | RTD 2 Over range |
| | | RTD 2 Error | %IX72.6 | BIT | | RTD 2 Error |
| | | RTD 2 TxPDO State | %IX73.6 | BIT | | RTD 2 TxPDO State |
| | | RTD 2 TxPDO Toggle | %IX73.7 | BIT | | RTD 2 TxPDO Toggle |
| | | RTD 2 Value | %IW37 | INT | | RTD 2 Value |
| | | RTD 3 Under range | %IX76.0 | BIT | | RTD 3 Under range |
| | | RTD 3 Over range | %IX76.1 | BIT | | RTD 3 Over range |
| | | RTD 3 Error | %IX76.6 | BIT | | RTD 3 Error |
| | | RTD 3 TxPDO State | %IX77.6 | BIT | | RTD 3 TxPDO State |
| | | RTD 3 TxPDO Toggle | %IX77.7 | BIT | | RTD 3 TxPDO Toggle |
| | | RTD 3 Value | %IW39 | INT | | RTD 3 Value |
| | | RTD 4 Under range | %IX80.0 | BIT | | RTD 4 Under range |
| | | RTD 4 Over range | %IX80.1 | BIT | | RTD 4 Over range |
| | | RTD 4 Error | %IX80.6 | BIT | | RTD 4 Error |
| | | RTD 4 TxPDO State | %IX81.6 | BIT | | RTD 4 TxPDO State |
| | | RTD 4 TxPDO Toggle | %IX81.7 | BIT | | RTD 4 TxPDO Toggle |
| | | RTD 4 Value | %IW41 | INT | | RTD 4 Value |

# CT65模块应用

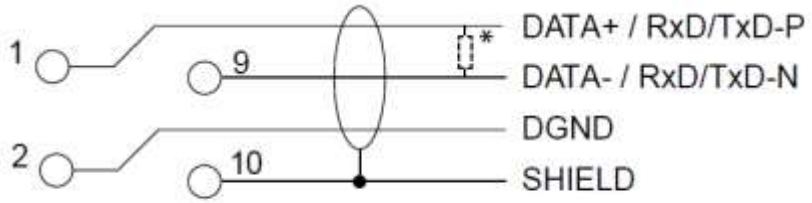
- **TIM6·1 – 三线制**
- 变量映射

| 通道名称 | 类型 | 描述 |
|---|---|---|
| Under range | BIT | 超出测量下限，TRUE值报警 |
| Over range | BIT | 超出测量上限，TRUE值报警 |
| Error | BIT | Under range或Over range值触发时，Error值触发，TRUE值报警 |
| TxPDO State | BIT | 暂无实际意义 |
| TxPDO Toggle | BIT | 数据刷新标识，数据更新时TRUE/FALSE值翻转。 |
| Value | INT | 温度采集的值（实际值的10倍） |

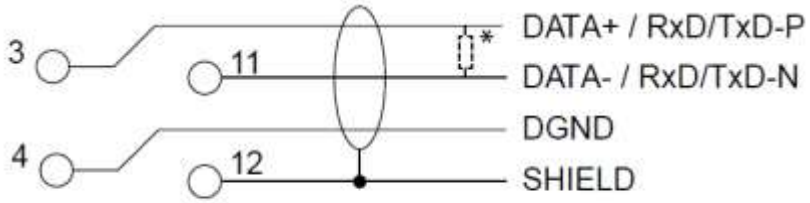
# CT65模块应用

## ➢ IFM6·1

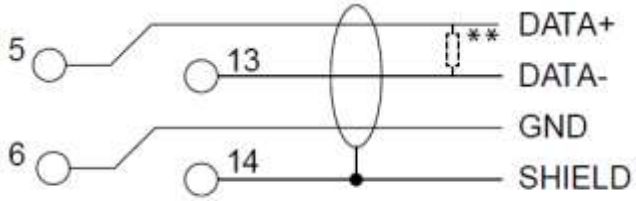
IFM6·1模块拥有两个Profibus DP Master和两个RS-485接口



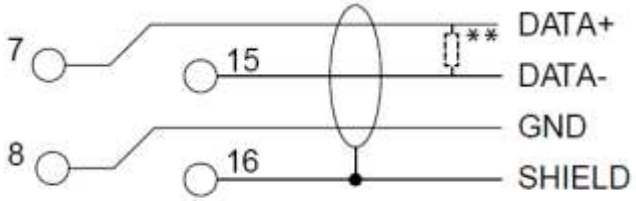

Profibus Master port 1




Profibus Master port 2




COM - RS-485 port 1




COM - RS-485 port 2

# CT65模块应用

- **IFM6·1 - Profibus-DP Master**

  IFM6·1模块拥有两个Profibus DP Master接口，每个接口可以连接最多五个DP slave。

# CT65模块应用

- **IFM6·1 - Profibus-DP Master**

- 重置过程数据

IFM6.1 通信模块具有 2 个 Profibus DP Master 端口，在进行 Profibus DP 变量链接之前，需要进行 Process Data 设置。IFM6.1 提供一个默认 122 字节的数组来实现与Profibus DP 子站的数据交互，该数组与 Profibus 通信数据的映射是由 PDO 实现的。

Process Data 设置需要将“16#1702”替换成“16#1600”（用于 slave1）、“16#1601”（用于 slave2）、“16#1602”（用于slave3）、“16#1603”（用于slave4）、“16#1604”（用于slave5）；

# CT65模块应用

- **IFM6·1 - Profibus-DP Master**
- 重置过程数据

| Profibus Outputs | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 16#1702 | | | | | 16#1703 | | | | |
| 16#1600 | 16#1601 | 16#1602 | 16#1603 | 16#1604 | 16#1640 | 16#1641 | 16#1642 | 16#1643 | 16#1644 |
| DP1 Slave1 | DP1 Slave2 | DP1 Slave3 | DP1 Slave4 | DP1 Slave5 | DP2 Slave1 | DP2 Slave2 | DP2 Slave3 | DP2 Slave4 | DP2 Slave5 |

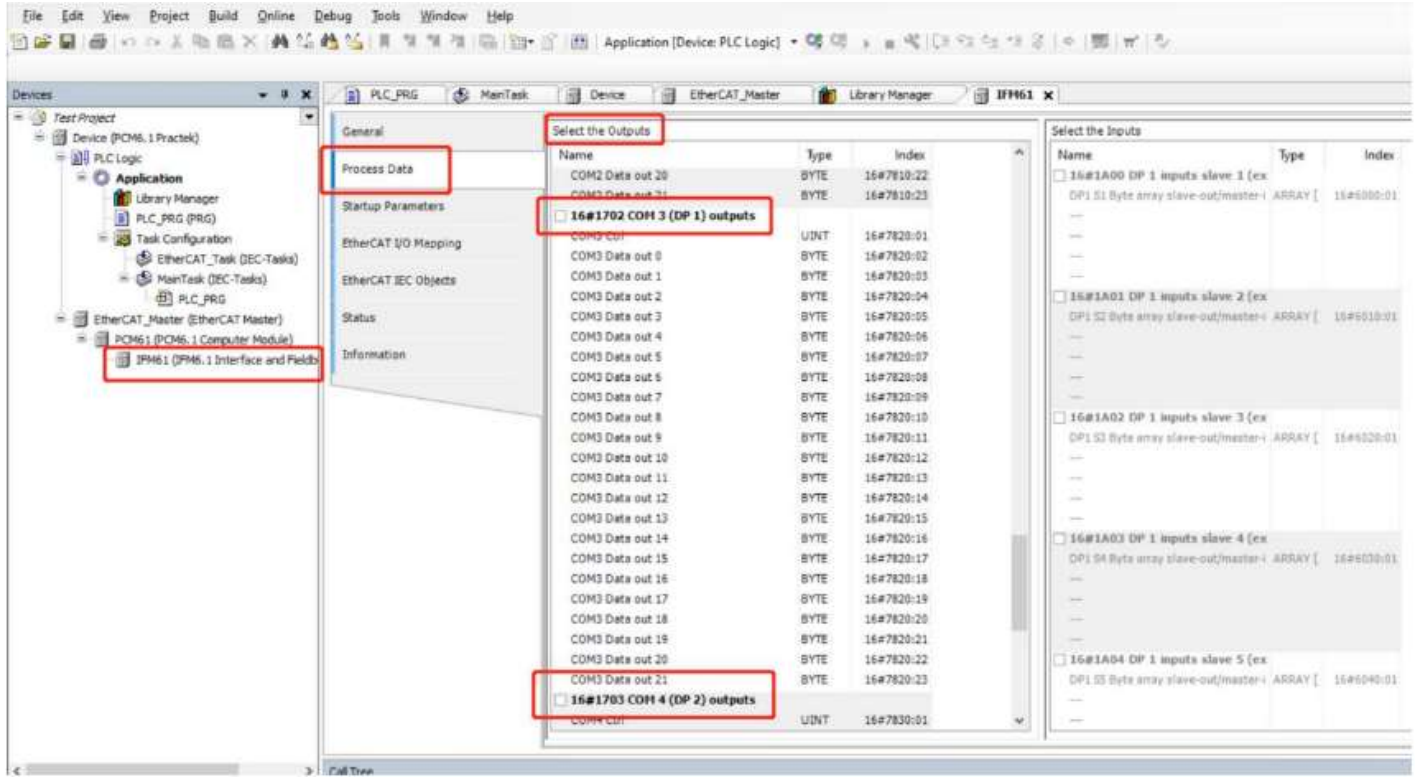
# CT65模块应用

- **IFM6·1 - Profibus-DP Master**
- 重置过程数据

单击“IFM61”、“Process Data”进入 Process Data 配置页面。在“select the Outouts”分组内取消“16#1702”和“16#1703”的勾选，勾选“16#1600”和“16#1640”，这样就设置了 Profibus Outputs 的 DP1 slave1 和 DP2 slave1。如果有两个子站同时连接主站，那么需要继续勾选“16#1601”和“16#1641”，设置 Profibus Outputs 的 DP1 slave2 和 DP2 slave2。

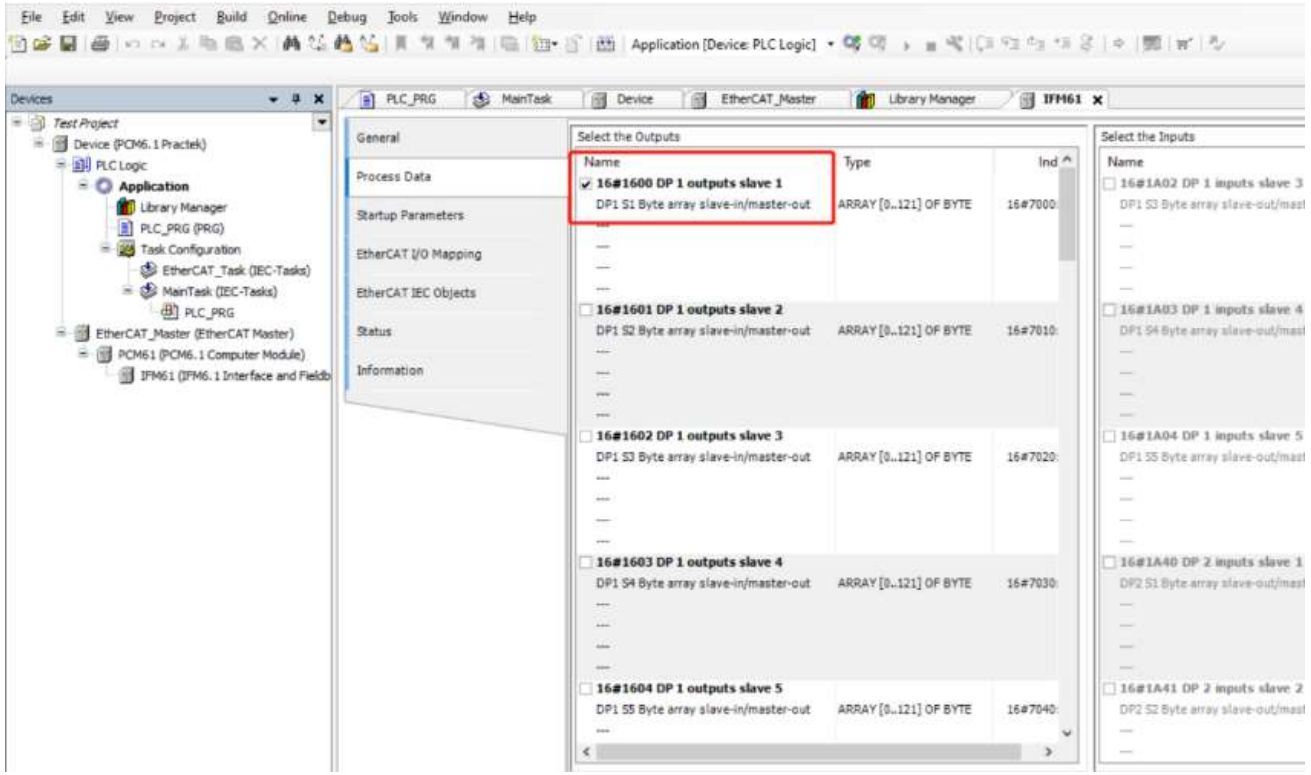
# CT65模块应用

- IFM6·1 - Profibus-DP Master
- 重置过程数据



# CT65模块应用

- IFM6·1 - Profibus-DP Master
- 重置过程数据



# CT65模块应用

- **IFM6·1 - Profibus-DP Master**
- 重置过程数据

将“16#1B02”替换成“16#1A00”（用于slave1）、“16#1A01”（用于slave2）、“16#1A02”（用于slave3）、“16#1A03”（用于slave4）、“16#1A04”（用于slave5），完成 Profibus DP1的slave 1、slave 2、slave 3、slave 4、slave 5 的 Process Data 设置。“16#1703”和 “16#1B03”设置方法与上面相同，用于 Profibus DP2 的数据通信。

# CT65模块应用

- **IFM6·1 - Profibus-DP Master**
- 重置过程数据

| Profibus Inputs | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 16#1B02 | | | | | 16#1B03 | | | | |
| 16#1A00 | 16#1A01 | 16#1A02 | 16#1A03 | 16#1A04 | 16#1A40 | 16#1A41 | 16#1A42 | 16#1A43 | 16#1A44 |
| DP1 Slave1 | DP1 Slave2 | DP1 Slave3 | DP1 Slave4 | DP1 Slave5 | DP2 Slave1 | DP2 Slave2 | DP2 Slave3 | DP2 Slave4 | DP2 Slave5 |

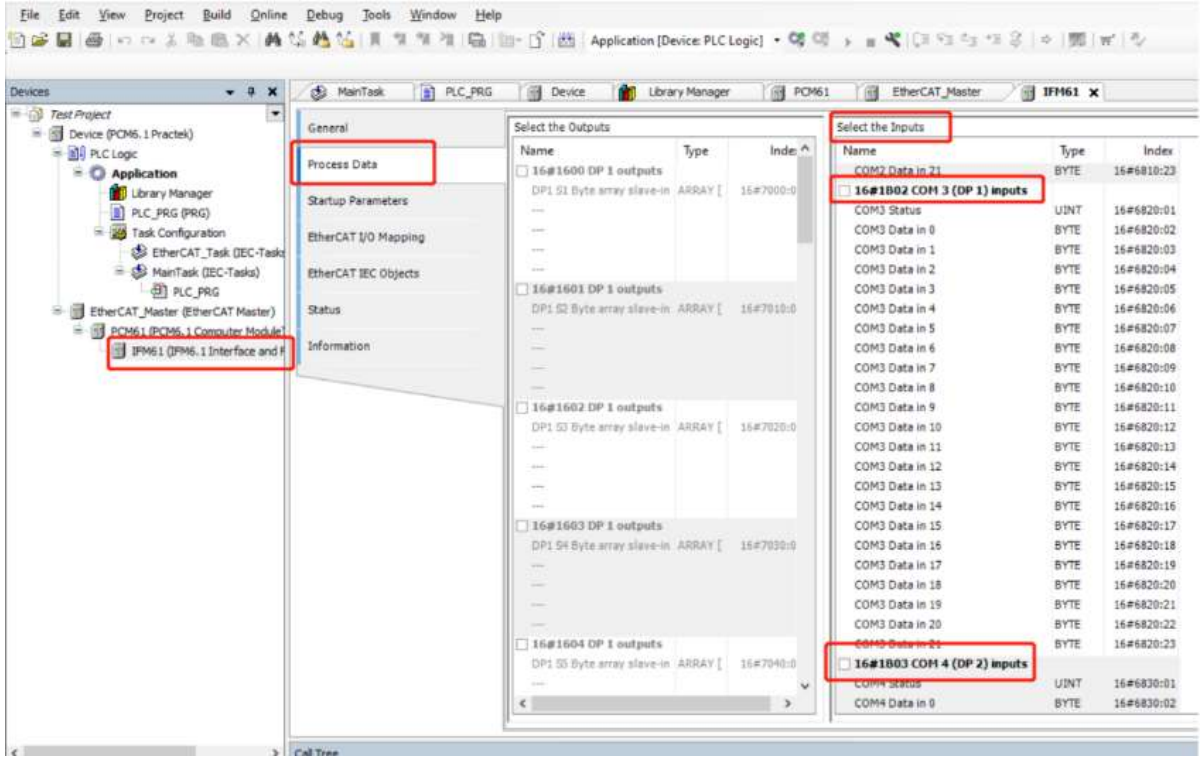
# CT65模块应用

- **IFM6·1 - Profibus-DP Master**
- 重置过程数据

单击“IFM61”、“Process Data”进入 Process Data 配置页面。在“select the Inputs”分组内取消“16#1B02”和“16#1B03”的勾选，勾选“16#1A00”、“16#1A40”。这样就设置了Profibus Inputs 的 DP1 slave1 和 DP2 slave1。如果有两个子站同时连接主站，那么需要继续勾选“16#1A01”和“16#1A41”，设置 Profibus Inputs 的 DP1 slave2 和 DP2 slave2。

# CT65模块应用

- IFM6·1 - Profibus-DP Master
- 重置过程数据



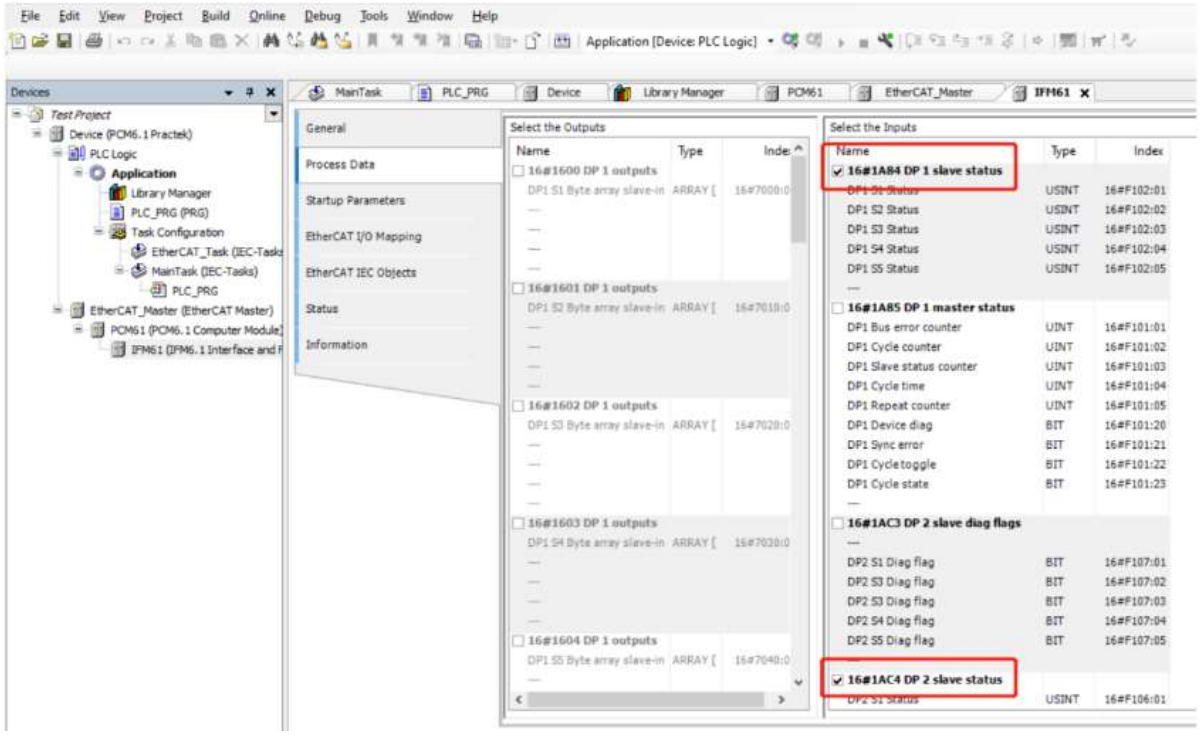
# CT65模块应用

- **IFM6·1 - Profibus-DP Master**
- 重置过程数据

在“select the Inputs”分组内勾选“16#1A84”、“16#1AC4”，这样就设置了Profibus Inputs 的 DP1 和 DP2 子站通信状态。

# CT65模块应用

- IFM6·1 - Profibus-DP Master
- 重置过程数据



# CT65模块应用

- **IFM6·1 - Profibus-DP Master**
- 设置启动参数

例程中与子站某品牌变桨控制器通信的Profibus DP启动参数设置，关于变桨控制器配置需要如下必须信息：

从站的站号，该品牌变桨控制器的站号为1；

Ident Number，该品牌变桨控制器的Ident Number为0x1810；

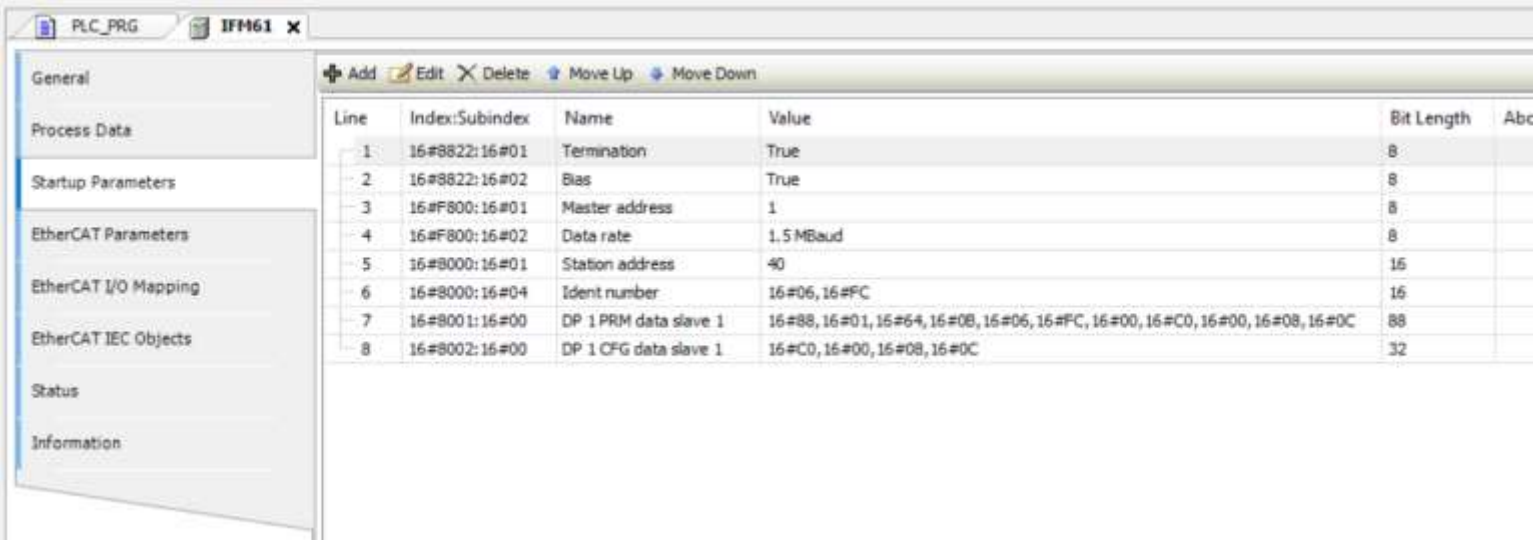
ParametersData，主要为需要设置通讯中的watchdog参数，包含watchdog1和watchdog2两个参数，满足持续时间 = Watch Dog1 * Watch Dog2 * 10ms，本例中设置watchdog1和watchdog2均为10，需要设置的ParametersData即为0x88 0x0A 0x0A 0x0B 0x18 0x10 0x00；其中0x88为固定参数，0x0A和0x0A分别为watchdog1和watchdog的值，0x0B为固定参数。0x18 0x10为Ident Number，0x00也为固定参数。

# CT65模块应用

- **IFM6·1 - Profibus-DP Master**

- 设置启动参数

CfgData，其为正常通讯后传输的数据的类型，根据通讯协议，该品牌变桨控制器传输内容数据类型配置为：16#10,16#10,16#10,16#5F,16#5F,16#5F,16#53,16#20,16#20,16#20,16#67,16#61,16#60




| Line | Index:Subindex | Name | Value | Bit Length |
|---|---|---|---|---|
| 1 | 16#8822:16#01 | Termination | True | 8 |
| 2 | 16#8822:16#02 | Bias | True | 8 |
| 3 | 16#F800:16#01 | Master address | 1 | 8 |
| 4 | 16#F800:16#02 | Data rate | 1.5 MBaud | 8 |
| 5 | 16#8000:16#01 | Station address | 40 | 16 |
| 6 | 16#8000:16#04 | Ident number | 16#06,16#FC | 16 |
| 7 | 16#8001:16#00 | DP 1 PRM data slave 1 | 16#88,16#01,16#64,16#0B,16#06,16#FC,16#00,16#C0,16#00,16#08,16#0C | 88 |
| 8 | 16#8002:16#00 | DP 1 CFG data slave 1 | 16#C0,16#00,16#08,16#0C | 32 |

# CT65模块应用

## ➢ IFM6·1 - Profibus-DP Master

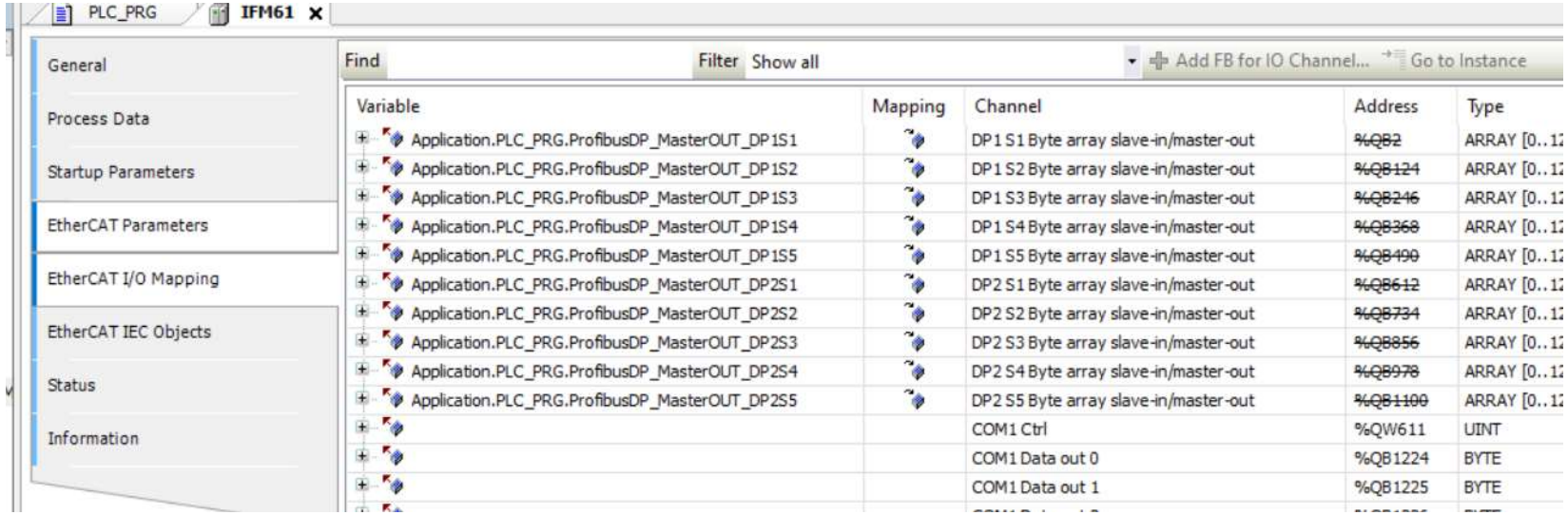
## ➢ 设置启动参数

| 名称 | 类型 | 说明 | | |
|---|---|---|---|---|
| Termination | USINT（8位） | 终端电阻，true为开启，false为关闭 | | |
| Master address | USINT（8位） | 主站地址 | | |
| Station address | UINT（16位） | 从站地址 | | |
| Data rate | USINT(8位) | 波特率 | 枚举名 | 值 |
| | | | 9.6 kBaud | 0 |
| | | | 19.2 kBaud | 1 |
| | | | 93.75 kBaud | 2 |
| | | | 187.5 kBaud | 3 |
| | | | 500 kBaud | 4 |
| | | | 1.5 MBaud | 6 |
| | | | 3 MBaud | 7 |
| | | | 6 MBaud | 8 |
| | | | 12 MBaud | 9 |
| | | | 45.45 kBaud | 11 |
| Ident number | UDINT（16位） | 厂商识别号 | | |
| PRM data | BYTE（88位） | 需要设置通讯中的watchdog参数，包含watchdog1和watchdog2两个参数，满足持续时间 = Watch Dog1 * Watch Dog2 * 10ms | | |
| CFG data | BYTE（32位） | 正常通讯后传输的数据的类型 | | |

# CT65模块应用

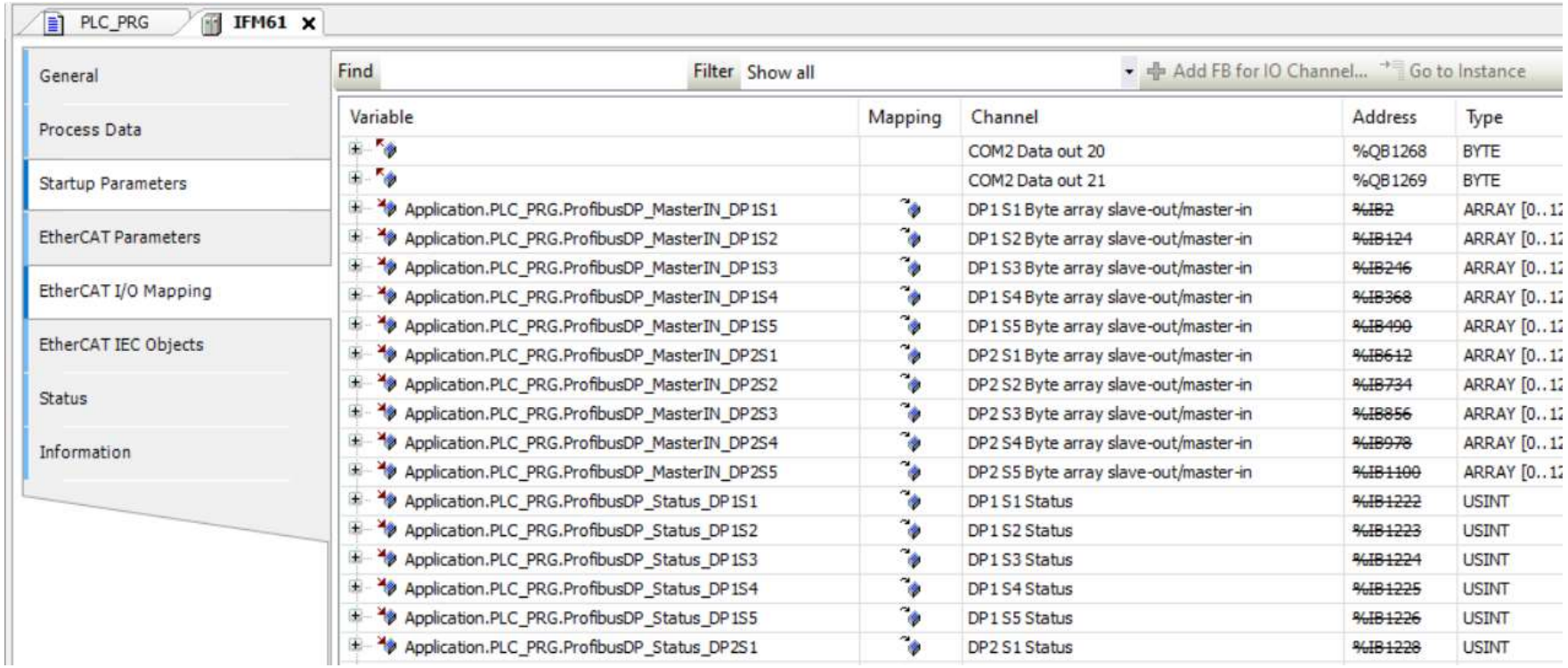
- **IFM6·1 - Profibus-DP Master**
- 链接变量

在EtherCAT I/O Mapping页中链接DP Byte array slave-out/master-in和DP Byte array slave-in/master-out数组及DP Status变量。



# CT65模块应用

- IFM6·1 - Profibus-DP Master
- 链接变量




| Variable | Mapping | Channel | Address | Type |
|---|---|---|---|---|
| | | COM2 Data out 20 | %QB1268 | BYTE |
| | | COM2 Data out 21 | %QB1269 | BYTE |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP1S1 | | DP1 S1 Byte array slave-out/master-in | %IB2 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP1S2 | | DP1 S2 Byte array slave-out/master-in | %IB124 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP1S3 | | DP1 S3 Byte array slave-out/master-in | %IB246 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP1S4 | | DP1 S4 Byte array slave-out/master-in | %IB368 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP1S5 | | DP1 S5 Byte array slave-out/master-in | %IB490 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP2S1 | | DP2 S1 Byte array slave-out/master-in | %IB612 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP2S2 | | DP2 S2 Byte array slave-out/master-in | %IB734 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP2S3 | | DP2 S3 Byte array slave-out/master-in | %IB856 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP2S4 | | DP2 S4 Byte array slave-out/master-in | %IB978 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_MasterIN_DP2S5 | | DP2 S5 Byte array slave-out/master-in | %IB1100 | ARRAY [0..12 |
| Application.PLC_PRG.ProfibusDP_Status_DP1S1 | | DP1 S1 Status | %IB1222 | USINT |
| Application.PLC_PRG.ProfibusDP_Status_DP1S2 | | DP1 S2 Status | %IB1223 | USINT |
| Application.PLC_PRG.ProfibusDP_Status_DP1S3 | | DP1 S3 Status | %IB1224 | USINT |
| Application.PLC_PRG.ProfibusDP_Status_DP1S4 | | DP1 S4 Status | %IB1225 | USINT |
| Application.PLC_PRG.ProfibusDP_Status_DP1S5 | | DP1 S5 Status | %IB1226 | USINT |
| Application.PLC_PRG.ProfibusDP_Status_DP2S1 | | DP2 S1 Status | %IB1228 | USINT |

# CT65模块应用

- **IFM6·1 - Profibus-DP Master**

- 解析数据

链接到DP1 S1 Byte array slave-in/master-out地址的ProfibusDP_MasterOUT_DP1S1变量数组，即为第一个DP接口上主站发送给第一个子站的数据，在程序中为这个变量数组赋值，即可将数据发送到子站。

链接到DP1 S1 Byte array slave-out/master-in地址的ProfibusDP_MasterIN_DP1S1变量数组，即为第一个DP接口上第一个子站发送给主站的数据，在程序中读取这个数组中的数值，即可获取子站发送的数据。

通讯建立后通过查看DP Status的值判断通讯状态，比如DP1 S1 Status为第一个DP接口第一个子站的通讯状态。DP Status数值为0，表示通信已正常连接进行数据交换。DP Status具体数值以及相对应的含义见下图。

# CT65模块应用

- IFM6·1 - Profibus-DP Master
- 解析数据

| DP Status 枚举值 | 描述 |
|---|---|
| 0 | 通讯正常 |
| 1 | 从站未使能 |
| 2 | 从站无响应 |
| 3 | 从站正与其他主站交换数据 |
| 4 | 不正确的从站响应，例如数据交互时，未激活服务 |
| 5 | 从站报告 prm 参数错误，通常是由于主站配置了错误的 ident number 或者 user parameter |
| 6 | 从站报告 DP 函数不支持 |
| 7 | 从站报告配置错误，通常是由于配置了错误的CFG参数，例如添加了错误的 modules |
| 8 | 从站未准备好数据交互 |
| 9 | 从站报告静态诊断 |

# CT65模块应用

- IFM6·1 - Profibus-DP Master
- 解析数据

| DP Status 枚举值 | 描述 |
|---|---|
| 10 | 备用 |
| 11 | 总线错误，例如 party 或者 checksum 错误，可能时总线受到干扰，出现错误帧 |
| 12~13 | 备用 |
| 14 | 响应错误，例如请求位被置位 |
| 15 | 从站报告无资源，通常是由于过长的PRM 或者 CFG 参数导致 |
| 16 | 从站报告 DP 服务未激活 |
| 17 | 意外的远程帧，例如等待从站响应时发现总线被占用，有可能是总线中存在其他主站或其他 节点发出的消息等 |
| 18 | 从站已经准备好进行数据交互，但没有数据通过 Ethercat 交互 |
| 19~255 | 备用 |

# CT65模块应用

- **IFM6·1 – Modbus RTU**
- 设置启动参数

  应按照串口实际使用情况设置启动参数中的终端电阻、偏置电阻、波特率及帧格式。

  各个通讯站点的波特率及帧格式应一致，通讯线路末端的站点应配置终端电阻。

# CT65模块应用

- **IFM6·1 – Modbus RTU**
- 设置启动参数

| 索引名称 | | | 子索引名称 | 类型 | 描述 |
|---|---|---|---|---|---|
| **COM * feature bits** | | | Termination | USINT(8位) | 终端电阻，true为开启，false为关闭 |
| | | | Bias | USINT(8位) | 偏置电阻，true为开启，false为关闭 |
| **COM * baud rate** | Baud rate | UDINT(32位) | 波特率<br>默认值为9600bps | 枚举名 | 值 |
| | | | | 2400bps | 2400 |
| | | | | 4800bps | 4800 |
| | | | | 9600bps | 9600 |
| | | | | 19200bps | 19200 |
| | | | | 38400bps | 38400 |
| | | | | 45450bps | 45450 |
| | | | | 57600bps | 57600 |
| | | | | 115200bps | 115200 |
| | | | | 230400bps | 230400 |
| | | | | 460800bps | 460800 |

# CT65模块应用

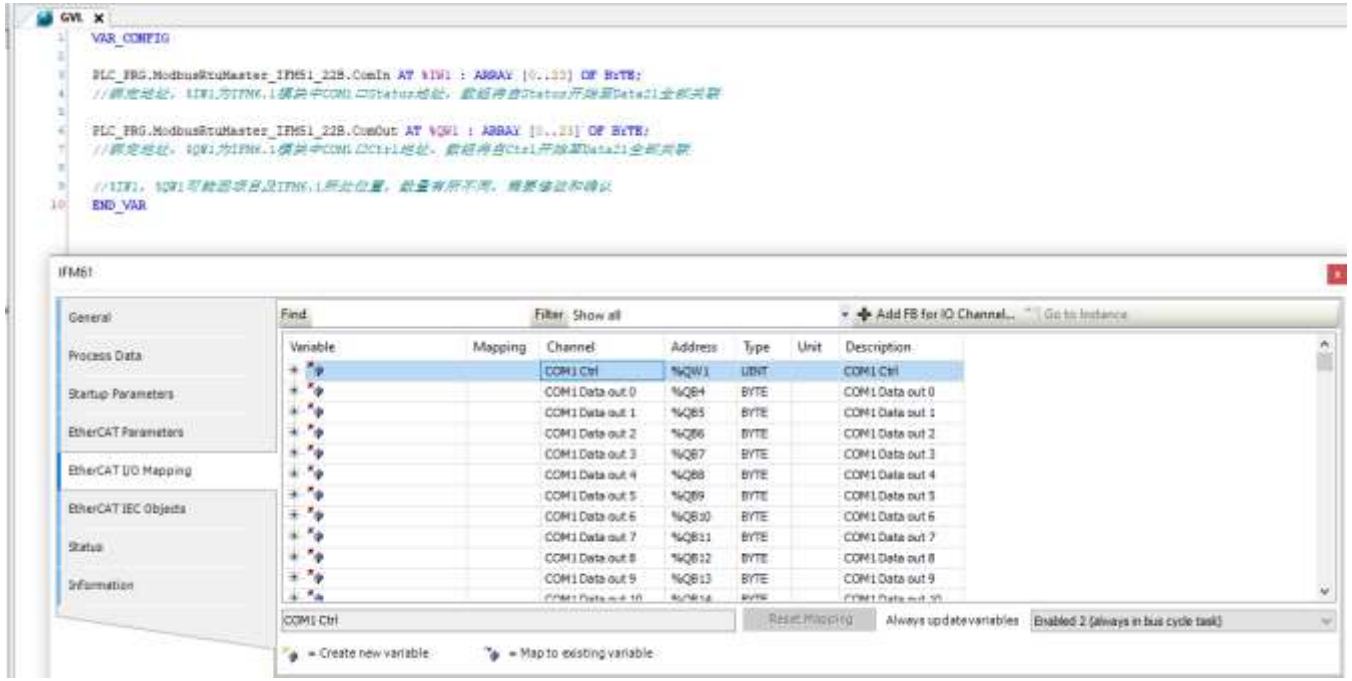
- IFM6·1 – Modbus RTU
- 设置启动参数

| 索引名称 | 子索引名称 | 类型 | 描述 | | |
|---|---|---|---|---|---|
| COM * data frame | Frame format | USINT(8位) | 帧格式<br>默认值为7ODD1 | 7EVEN1 | 0 |
| | | | | 7EVEN2 | 1 |
| | | | | 7ODD1 | 2 |
| | | | | 7ODD2 | 3 |
| | | | | 8NONE1 | 4 |
| | | | | 8NONE2 | 5 |
| | | | | 8EVEN1 | 6 |
| | | | | 8EVEN2 | 7 |
| | | | | 8ODD1 | 8 |
| | | | | 8ODD2 | 9 |

# CT65模块应用

- **IFM6·1 – Modbus RTU**
- 变量链接

在变量定义文件中，将Modbus RTU库中的通讯变量使用AT硬件地址的方式链接到设备通道。

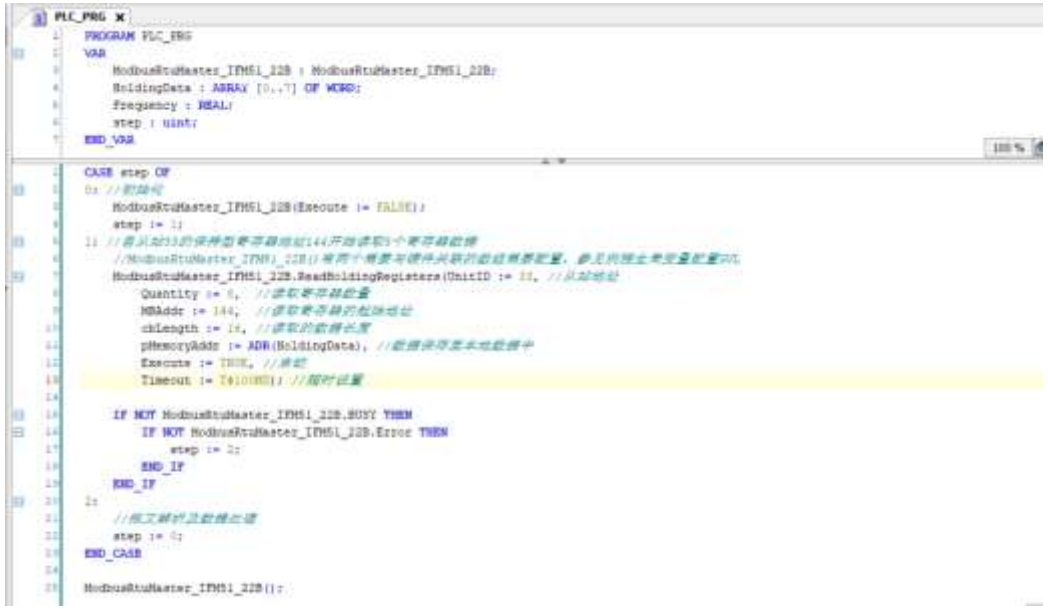


# CT65模块应用

- **IFM6·1 – Modbus RTU**

- 建立通讯

使用Modubus RTU库中的函数读写Modbus寄存器，如下图所示，使用ReadHoldingRegisters函数读取33号子站的保持寄存器。

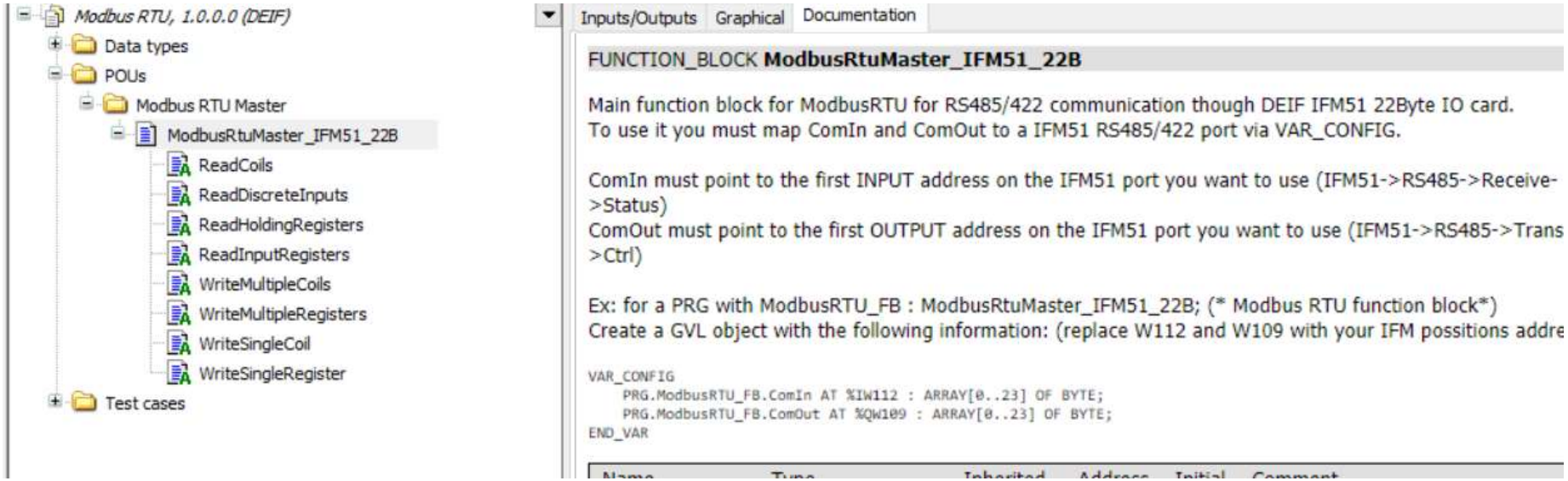


# CT65模块应用

- **IFM6·1 – Modbus RTU**

- 建立通讯

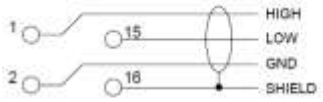
  Modbus RTU库所有的寄存器和线圈读取动作(Action)如下图所示，请根据实际的应用选择正确的动作。



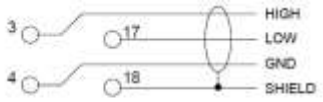
# CT65模块应用

## ➢ IFM6·2

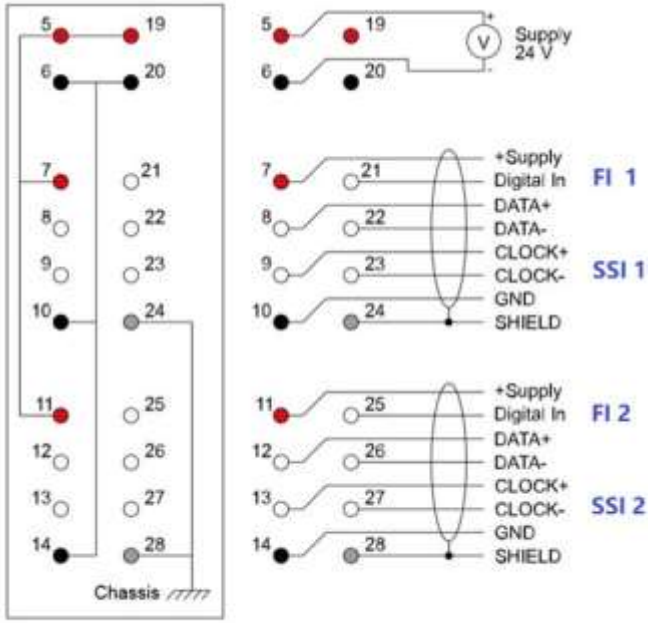
IFM6·2拥有两个CAN口，两个SSI，两个FI功能。



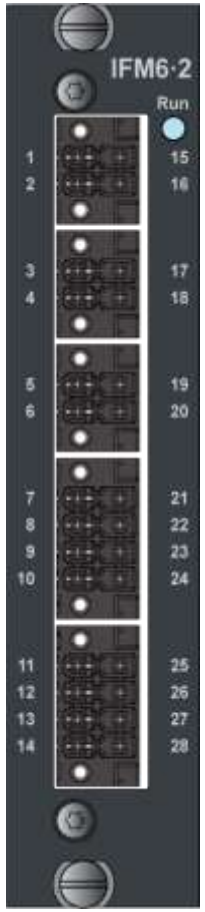

CAN port 1




CAN port 2




2 × SSI　2 × FI



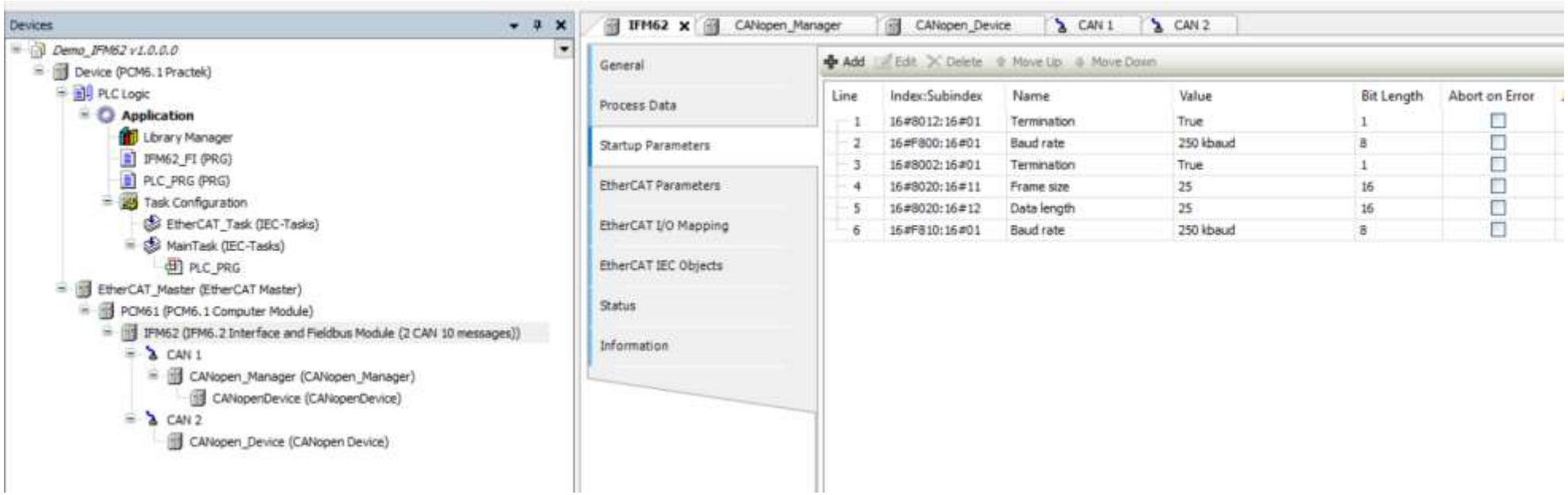
# CT65模块应用

## ➢ IFM6·2 – CANopen Master

当使用CAN口作为CANopen主站时，按以下使用方法操作。

## ➢ 设置启动参数

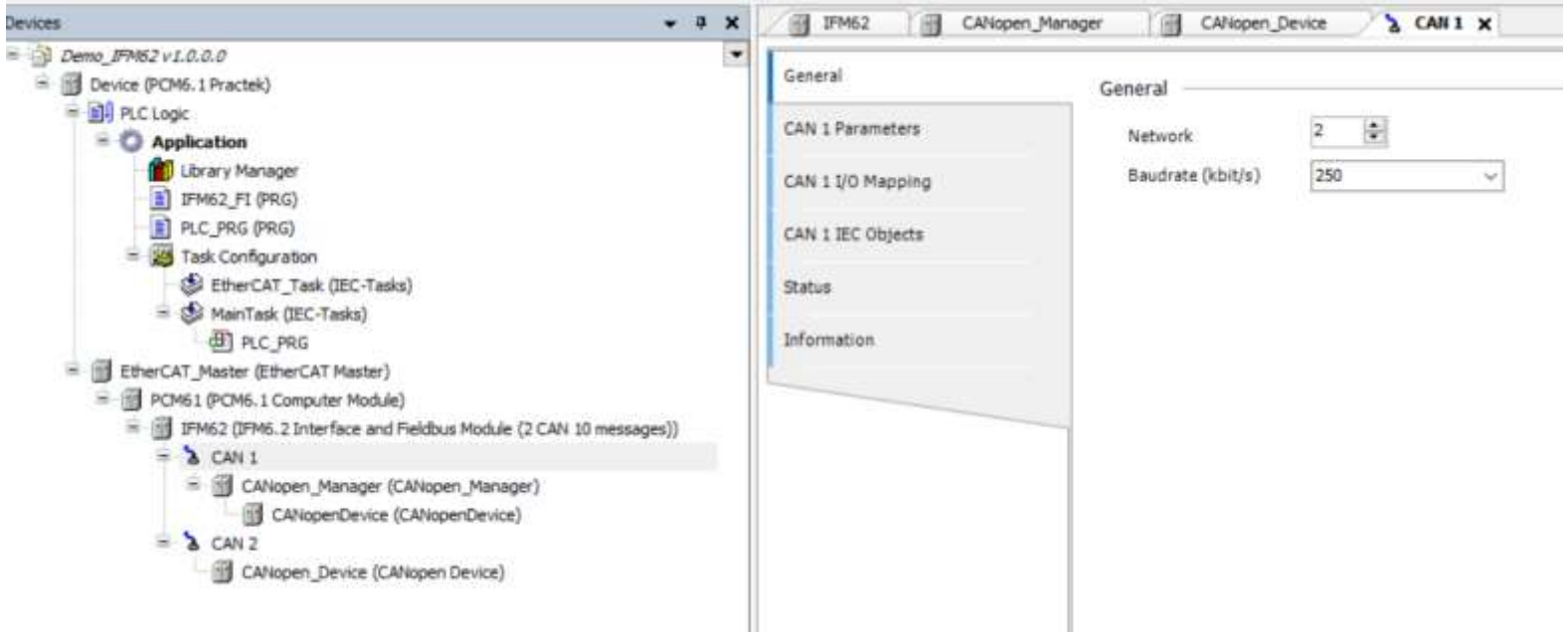
在IFM6.2的启动参数中为使用的CAN口设置波特率及终端电阻。



# CT65模块应用

- **IFM6·2 – CANopen Master**
- 设置启动参数

请注意在IFM6.2板卡的CAN port上也可为该CAN口设置波特率，当启动参数中未设置波特率时，CAN port上的波特率生效，如两处都设置的话，则需保证设置值一致。



# CT65模块应用

## ➢ IFM6·2 – CANopen Master

### ➢ 设置启动参数

CAN port上的Network数值无实际意义，但是不恰当的设置该值会导致下载程序时弹窗警告。可按以下规则进行设置：

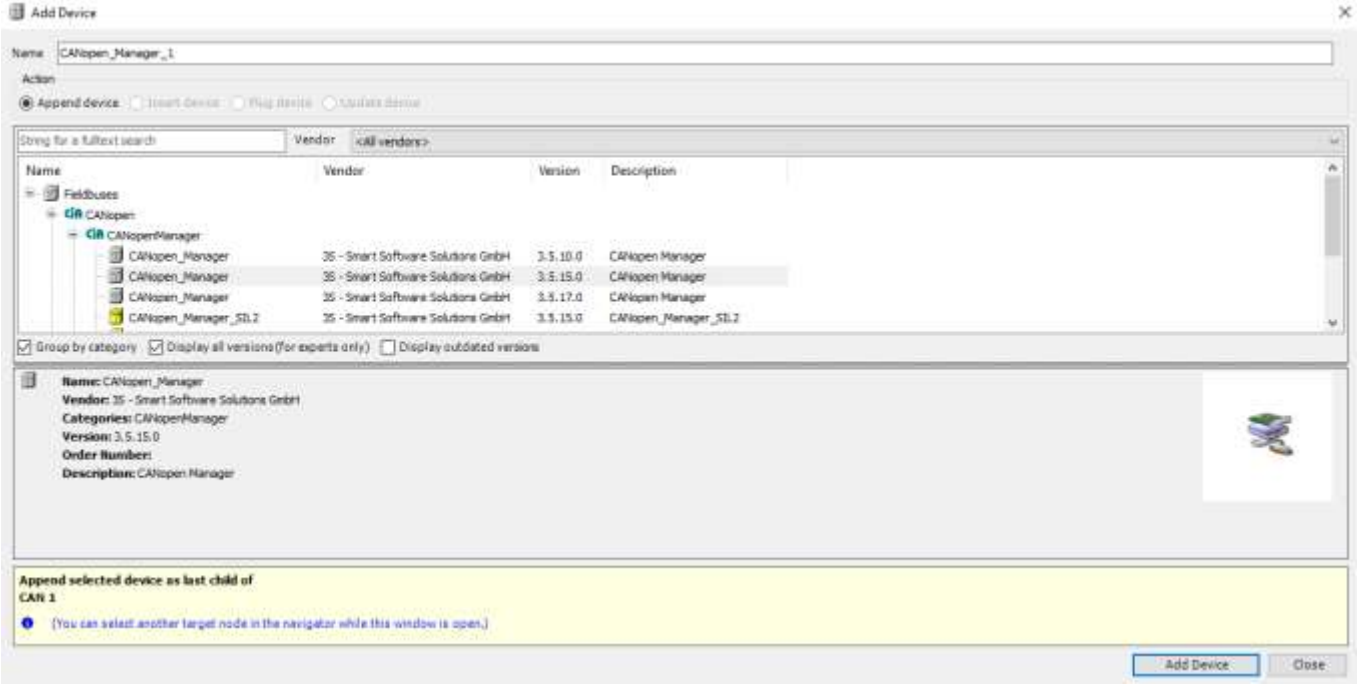
- Network的0值与1值被预留给PCM板卡，无论PCM板卡中的CAN口是否使用，IFM6.2板卡中的CAN port不应使用0与1。
- 第一块IFM6.2板卡上的CAN1 port的Network值应设置为2，CAN2 port的Network值应设置为3；第二块IFM6.2板卡则应顺延为4，5。依次向下排列。

# CT65模块应用

- **IFM6·2 – CANopen Master**

- CANopen Manager

在CAN port上右击，选择Add Device，在弹出的设备选择窗口中选择CANopen – CANopenManager – CANopen_Manager，点击窗口下方Add Device按钮，即可在CAN port上添加CANopen Manager。

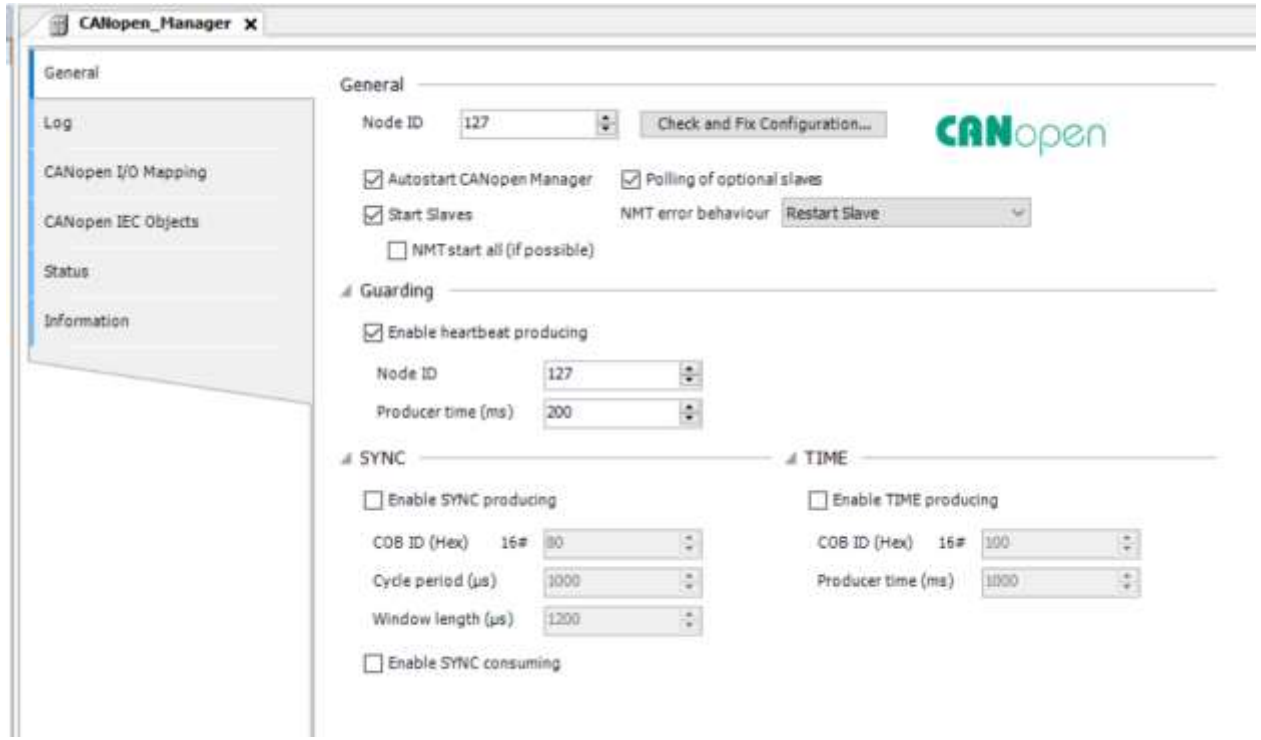


# CT65模块应用

- **IFM6·2 – CANopen Master**

- CANopen Manager

CANopen Manager – General页可对CANopen通讯主站进行各项设置。



# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen Manager - General区域

**Node ID：**主站站点号，默认127。Node ID的数值会包含在通讯报文COB-ID中，用以标识报文源/目标站点，所以在同一网络中，各设备Node ID不可相同。

**Check and Fix Configuration：**用以检查整个CANopen网络中是否存在Node ID重复或COB-ID重复。

**Autostart CANopen Manager：**用以配置主站是否自动切换至OP。当该选项勾选时，如果所有的子站配置完成进入OP状态，则主站自动进入OP状态；当该选项未勾选时，所有子站OP后，主站也不会自动进入OP状态，可以在代码中使用CiA405.NMT库切换主站的状态。当主站未OP时，所有的PDO都不会发送。

**Polling of optional slaves：**该选项勾选时，如果一个子站未响应配置，主站会以每秒一次的频率询问该子站。这种周期性询问会降低CANopen网络的实时性，在实时性要求高的网络中可以关闭该选项。当该选项关闭时，需要子站自动发送一个CANopen节点启动信息，主站才能侦测到该子站。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen Manager – General区域

**Start slaves：**当该选项勾选时，主站会启动子站；否则需要在代码中使用CiA405.NMT库去启动子站。

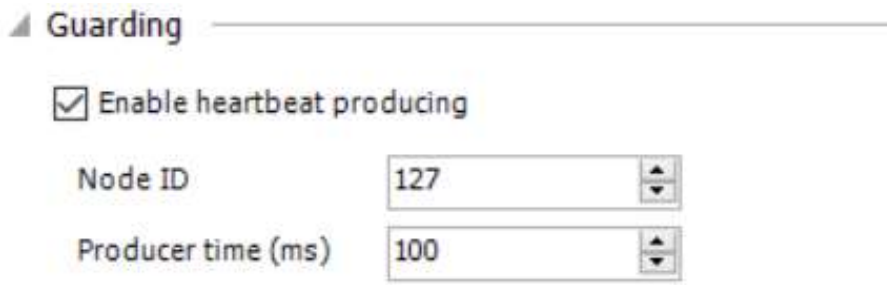
**NMT start all：**当该选项勾选时，如果所有可选子站都就绪，则会使用NMT Sart ALL指令启动所有子站，如果有子站未就绪，则单独启动每个子站；

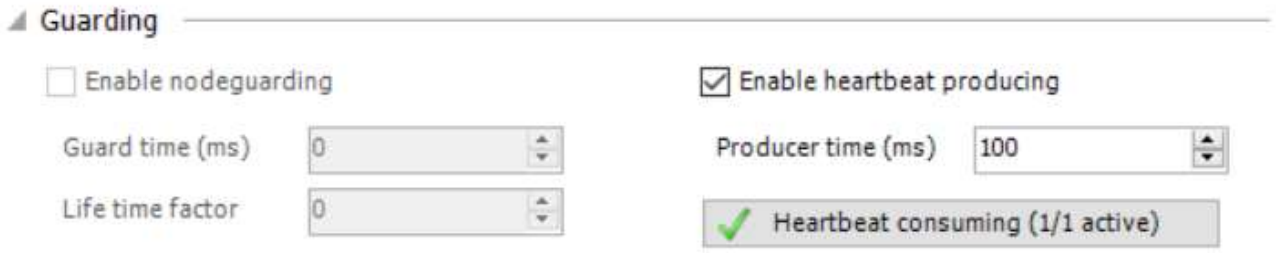
**NMT error behavior：**可以选择在发生NMT Error Event后对子站进行的操作，如选择Restart slave，则会使用NMT Rest + SDO Configuration + NMT start去启动子站；如选择Stop slave，在发生NMT Error Event后主站不再操作该子站，需要在代码中使用CiA405.NMT库手动管理子站。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen Manager - Guarding区域

**Enable hearbeat producing：**勾选后主站发送心跳信号，信号发送周期由Producer time定义。如果子站的EDS中列明子站支持心跳信号消费功能，则子站中可配置心跳消费（下图右侧Heartbeat consuming(1/1 active)除）。请注意下图左侧的主站生产心跳信号，和下图右侧的子站Heartbeat consuming心跳消费为对应的成对功能，实现主站发送，子站消费的心跳检测功能。下图右侧的子站Enable heartbeat producing指的是子站发送心跳，和它下方的Heartbeat consuming子站消费心跳是没有功能关系的。



# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen Manager - Guarding区域

**Node ID：**心跳信号的节点标识（主站发出的心跳信号，可以通过此处设置一个不同于主站节点号的标识号，此不同有何用处？建议不要修改该ID号，使用主站节点号即可）

**Producer time(ms)：**心跳信号发送周期

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen Manager - SYNC区域

**Enable SYNC producing：**勾选后主站发送SYNC同步报文，如子站PDO存在同步刷新类型，则需激活该功能。

**COB ID：**同步报文ID号，各设备一般按默认16#80配置。

**Cycle period：**同步报文发送周期。

**Window length：**同步报文时间窗口，该参数为同步PDO接收有效期，即主站在发送同步报文后，在该有效期内收到的PDO被视为有效数据，超出时间窗口后收到的PDO会被判定无效，进行丢弃处理。Window length和Cycle period不存在倍数关系，只需关注同步报文发送至从站，从站发送PDO传输至主站所需的时限。

**Enable SYNC consuming：**主站作为同步信号消费者，由其他设备发送同步信号。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen Manager - TIME区域

  **Enable TIME producing：**主站发送时间报文。

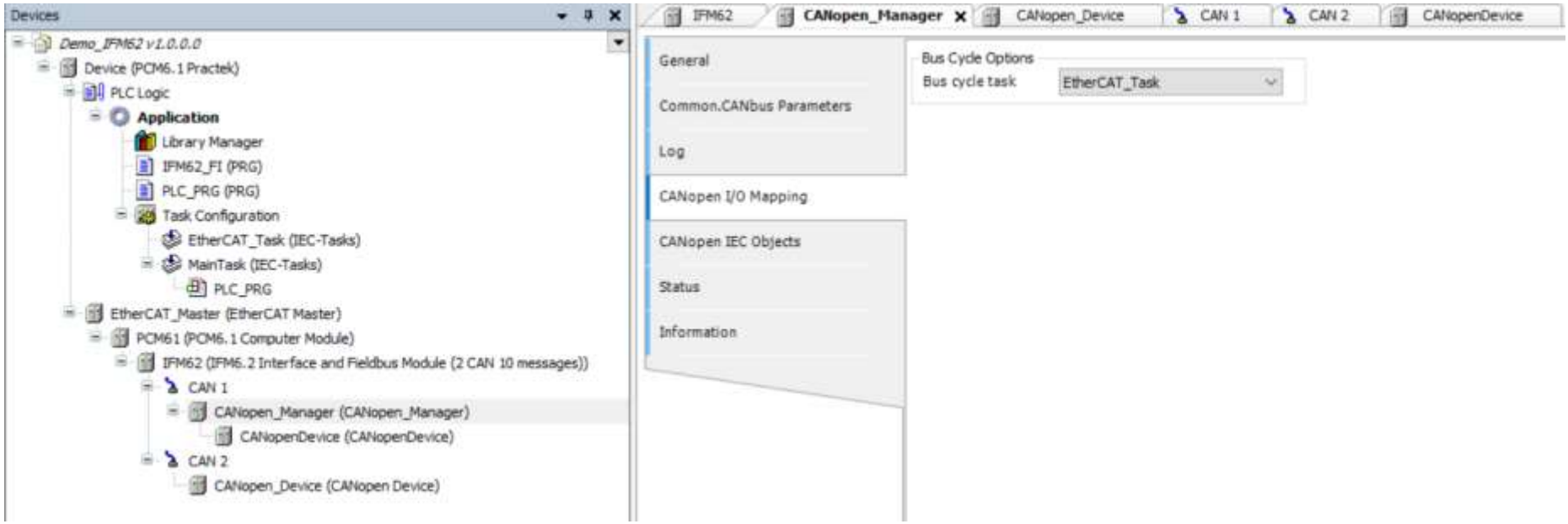
  **COB ID：**时间报文的cob id。

  **Producer time(ms)：**时间报文的发送周期。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- 为CANopen协议栈设置循环任务

在CANopen Manager的CANopen I/O Mapping页中可以为CANopen协议栈设置循环任务，建议CANopen协议栈任务的循环时间与EtherCAT总线同周期时长，也可直接使用EtherCAT总线循环任务。

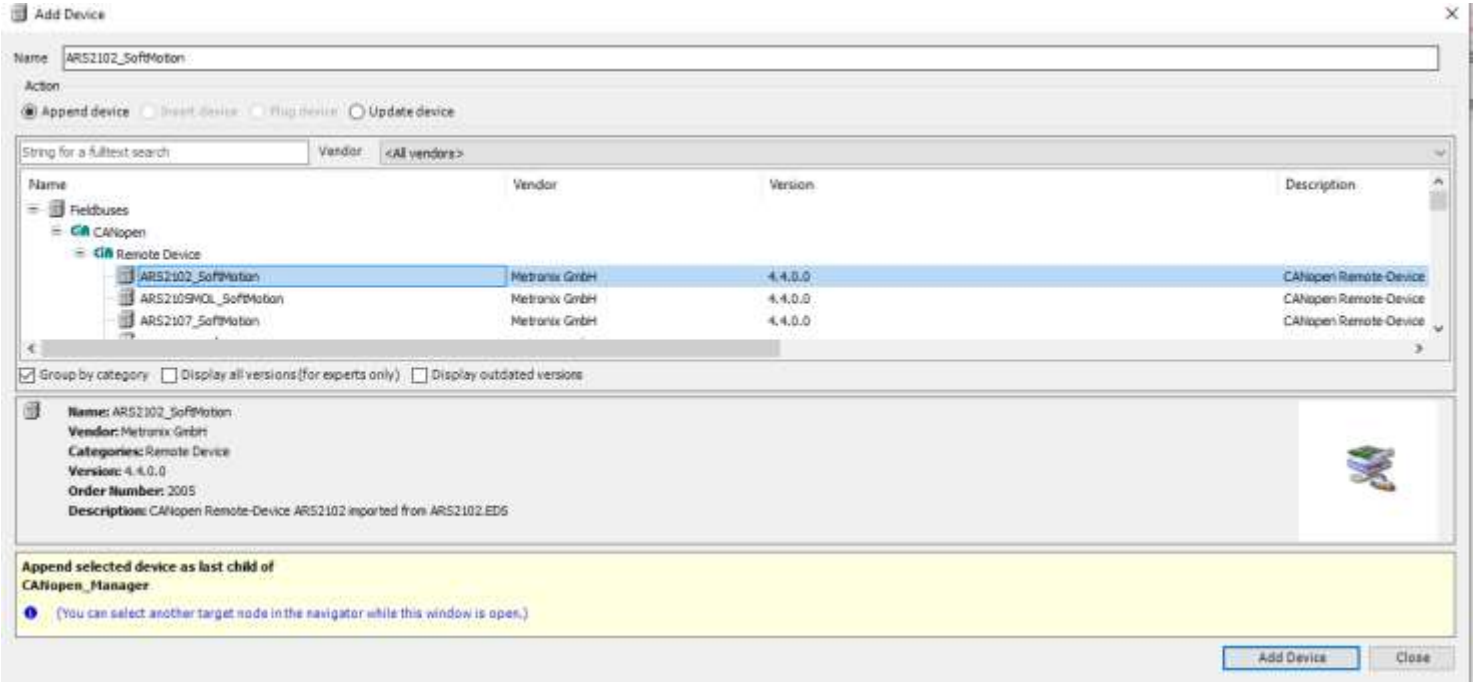


# CT65模块应用

- **IFM6·2 – CANopen Master**

- 添加CANopen remote device

在CANopen Manager上右击，选择Add device，在弹出的设备选择窗口中选择CANopen – Remote Device下希望添加的子站设备，点击窗口下方的Add Device，即可在CANopen Manager下添加子站设备。

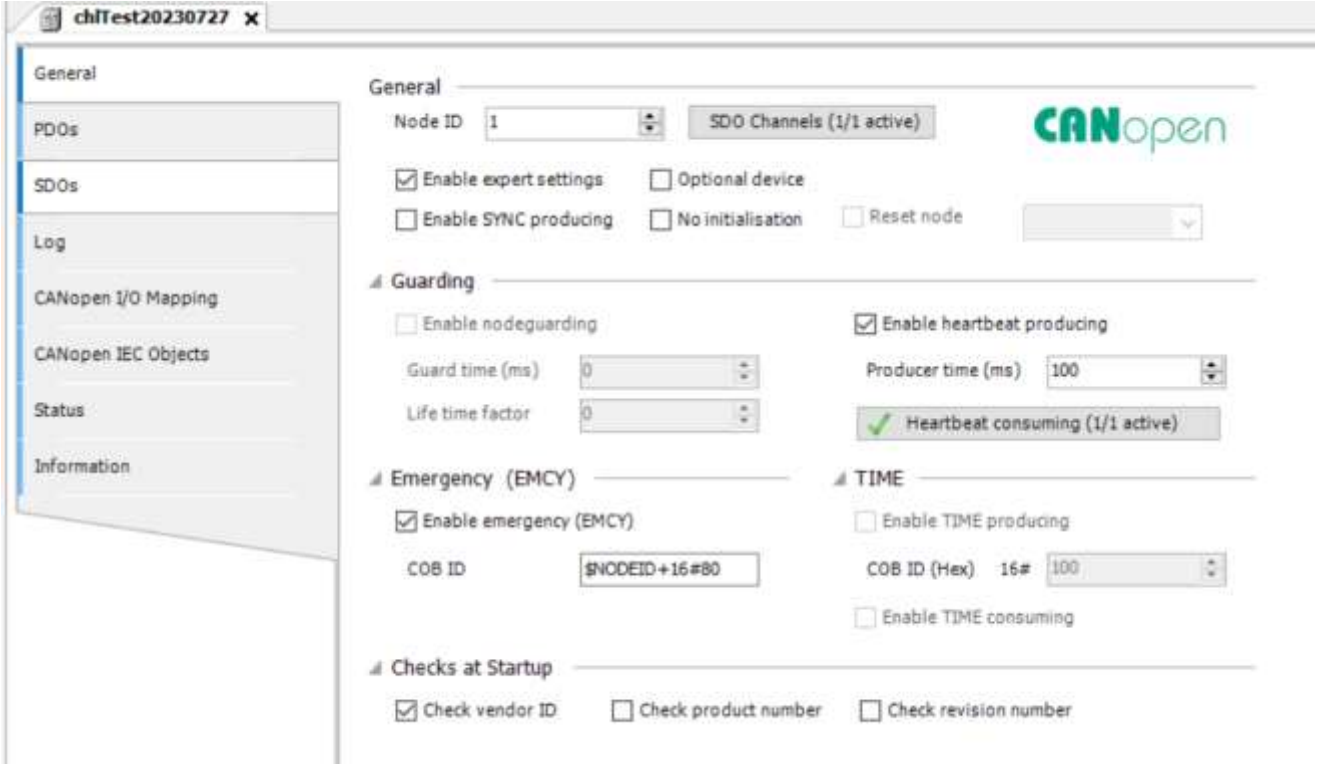


# CT65模块应用

- **IFM6·2 – CANopen Master**

- 添加CANopen remote device

  在子站设备General页，可以对子站的CANopen通讯进行各项设置。



# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **General区域**

**Node ID：**子站站点号

**SDO Channels(1/1 active)：**SDO通道配置

**Enable expert settings：**使能专家模式，勾选后可以看到所有选项（只是影响显示，不会影响是否生效）

**Optional device：**勾选后子站被作为可选设备对待，该子站的状态不再影响主站切换到运行模式。

**Enable SYNC producing：**子站作为同步报文生产者，需在主站激活Enable SYNC consuming并设置同步报文相关参数。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device - **General区域**

**No initialization：**勾选该选项后，主站不会对站点进行NMT管理，也不会通过SDO进行参数配置。PDO刷新和心跳、节点守卫功能会正常按配置生效。节点可以自动运行，或经由代码执行CiA.NMT库相关功能进行控制。

**Reset node：**当该选项被勾选时，依赖于右侧下拉框中的选项，有一部分参数不会被配置。Sub001:所有参数不可配置；Sub002:通讯参数不可配置(1000h-1FFFh)；Sub003:应用参数不可配置(6000h-9FFFh)；Sub004-Sub127:生产商指定区域不可设置；Sub128-Sub254:预留。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **Guarding区域**

**Enable node guarding：**该子站启动节点保护功能，勾选后主站以Guard time中设置的周期向该子站发送节点保护报文，如未收到子站回复，则重复发送，重复发送节点保护报文的次数由Life time factor参数设定，发送报文达到上限还未收到子站回复后，主站判定子站通讯故障。节点保护与心跳保护为互斥性设置，激活一个后，另一个设能设置变为灰色，不可激活。

**Guard time(ms)：**当节点保护功能开启后，主站以这个设定为周期发送节点保护询问报文。

**Life time factor：**当主站无法收到从站对节点保护报文的响应时，重复发送节点保护询问报文的次数。

**Enable heartbeat producing：**勾选后子站发送心跳报文，主站按Producer time * 1.5作为心跳超时的判断依据。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **Guarding区域**

**Producer time(ms)：**子站发送心跳报文的周期。

**Heartbeat consuming：**勾选后子站作为主站心跳报文的消费者。当该选项为灰色时，是因为子站EDS文件中1016h缺乏相应配置，默认该子站不支持心跳报文消费功能。当Heartbeat consuming为可操作状态时，可点击该选项打开配置框进行设置：Enable勾选时使能心跳消费；Node ID of guarded Node设置该子站消费哪个cob id的心跳信号（一个CAN网络中可能有多个设备发出心跳，作为心跳信号消费者，子站只能检测一个设备的心跳，一般检测主站心跳）；Consumer time为心跳消费时间，默认该值为主站心跳周期的1.5倍。

Heartbeat Consuming Properties

| Enable | Node ID of guarded Node | Consumer time (ms) |
|---|---|---|
| ☑ | 125 | 150 |

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **Emergency区域**

**Enable emergency(EMCY)：**当勾选该选项后，子站在发生内部错误时会发出一个紧急报文，可使用CiA405功能库读取该报文。

**COB ID：**紧急报文cob id号。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **TIME区域**

  **Enable TIME producing：**勾选后该子站发送时间报文。

  **COB ID(Hex)：**时间报文的cob id。

  **Enable TIME consuming：**勾选后子站作为时间报文消费者。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **Checks at Startup区域**

**Check vendor ID：**检查设备供应商ID，勾选后，SDO通讯配置阶段，主站会检查EDS文件中配置的ID是否与设备实际ID一致，如不一致，则报错并中断通讯配置。

**Check product number：**检查设备产品类型号码，勾选后，主站会检查EDS文件中配置的产品类型号与实际设备是否相符。

**Check revision number：**检查设备产品版本号，勾选后，主站会检查EDS文件中配置的产品版本号与实际设备是否相符。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device - **Checks at Startup区域**

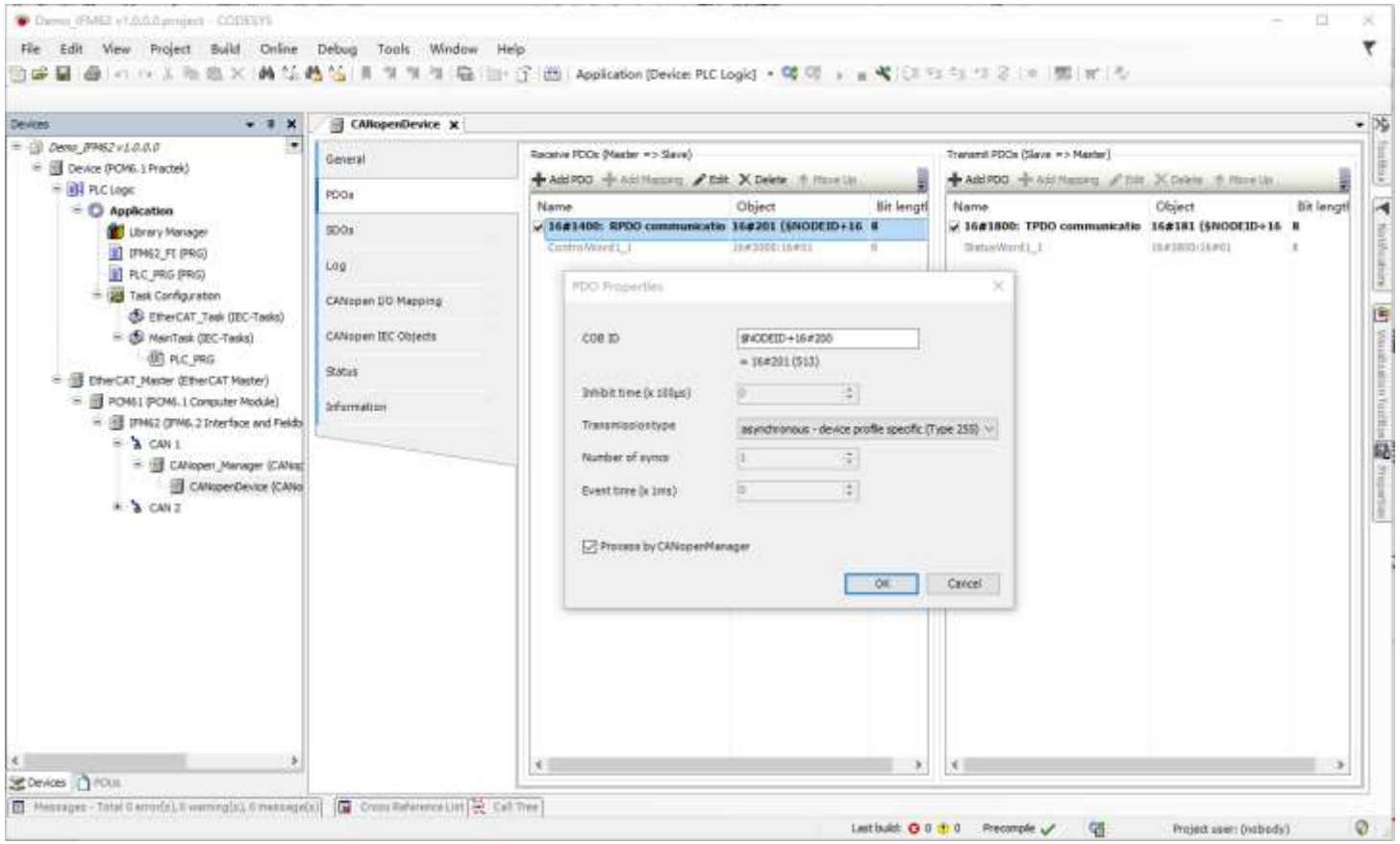
**Check vendor ID：**检查设备供应商ID，勾选后，SDO通讯配置阶段，主站会检查EDS文件中配置的ID是否与设备实际ID一致，如不一致，则报错并中断通讯配置。

**Check product number：**检查设备产品类型号码，勾选后，主站会检查EDS文件中配置的产品类型号与实际设备是否相符。

**Check revision number：**检查设备产品版本号，勾选后，主站会检查EDS文件中配置的产品版本号与实际设备是否相符。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **PDO Properties**



# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **PDO Properties**

COB ID：通讯对象标识（Communication Object Identifier），可以设置绝对数值，比如16#181，也可以设置公式，比如$NODEID+16#180，NODEID为该PDO所属站点号

RTR：TxPDO可勾选该选项，勾选后该PDO可由远程帧触发。

Inhibit time(x 100us)：抑制时间，该选项规定了两次PDO发送之间的最小时间间隔。仅在传输模式254，255中生效。传输模式254，255仅由事件驱动，如果事件触发频繁，PDO可能会以极高频率发送，造成CAN网络拥堵，故此需要这个参数来限制PDO最小发送间隔。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device - **PDO Properties**

Transmissiontype：传输类型。下拉可选acyclic-synchronous(Type 0)，同步非循环模式，事件发生之后，站点在收到同步信号后会传输数据，典型事件为数据变化或定时器中断；cyclic- synchronous(Type 1-240)，同步循环模式，在收到N个同步信号后发送数据，N由Number of syncs设置，设置范围1-240；synchronous-RTR only(Type 252)，同步RTR，收到同步信号后PDO更新但不发送，收到远程帧后发送数据；asynchronous-RTR only(Type 253)，异步RTR，收到远程帧后更新PDO并发送数据；asynchronous-manufacturer specific(Type 254)，供应商定义事件触发，通常是数据发生变化或达到定时时间；asynchronous-device profile specific(Type 255)，CiA协议规范指定事件触发。

# CT65模块应用

- **IFM6·2 – CANopen Master**
- CANopen remote device – **PDO Properties**

Number of syncs，同步信号数量

Event time(x 1ms)：事件计时器。仅在传输模式254，255中生效。传输模式254，255仅由事件触发，当事件一直不触发时，可通过设置该参数强制PDO每隔一段时间发送一次。当计时器未溢出，事件触发后，计时器清零。

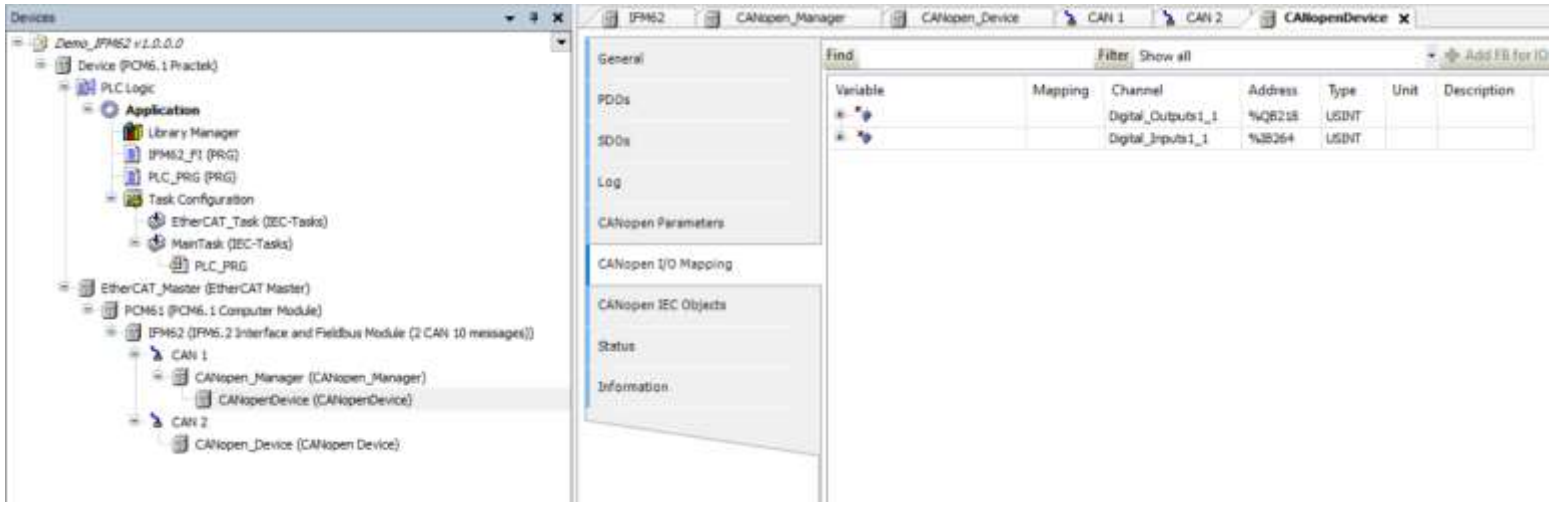
Process by CANopenManager：默认勾选，取消后CANopen Manager不再处理这个PDO，PDO不再自动发送或接收。

# CT65模块应用

- **IFM6·2 – CANopen Master**

- 链接变量

在添加的子站设备CANopen I/O Mapping页中，可以为子站链接变量，地址为%Q类型的变量为RxPDO相关数据，如截图中所示Digtal_Outputs1_1，在程序中向对应的变量写入数值，数值即由主站传输到子站；地址为%I类型的变量为TxPDO相关数据，如截图中所示Digtal_Inputs1_1，在程序中读取对应的变量值，即获取到由子站传输给主站的数据。



# CT65模块应用

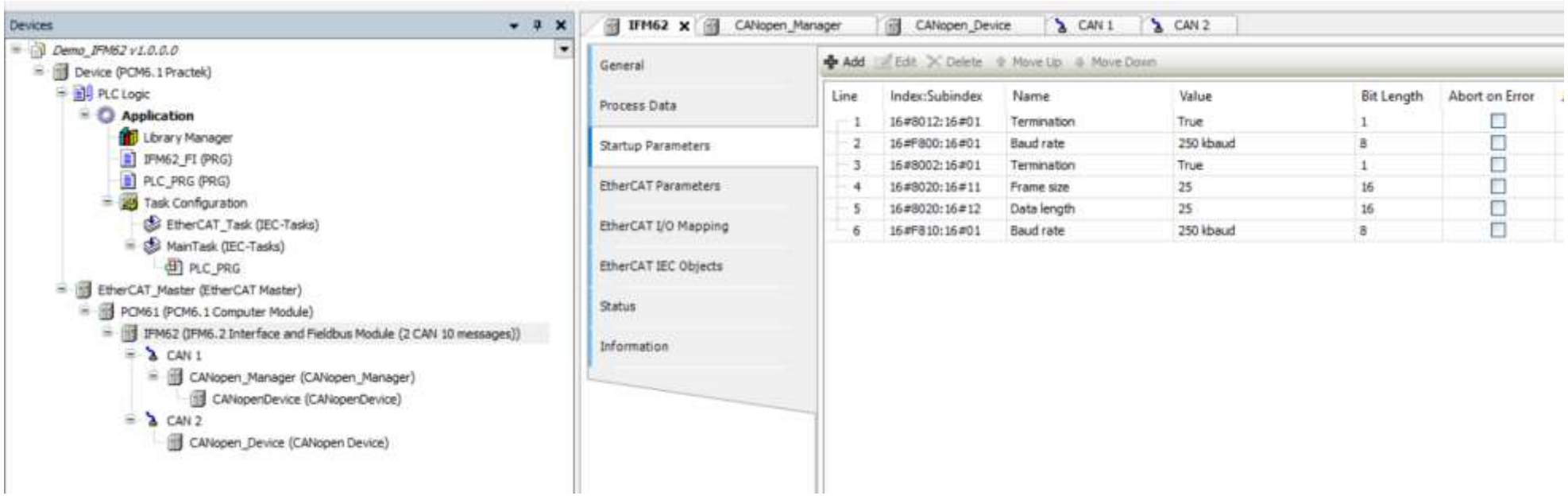
## ➢ IFM6·2 – CANopen Slave

当使用CAN口作为CANopen子站时，按以下使用方法操作。

# CT65模块应用

- **IFM6·2 – CANopen Slave**
- 设置启动参数

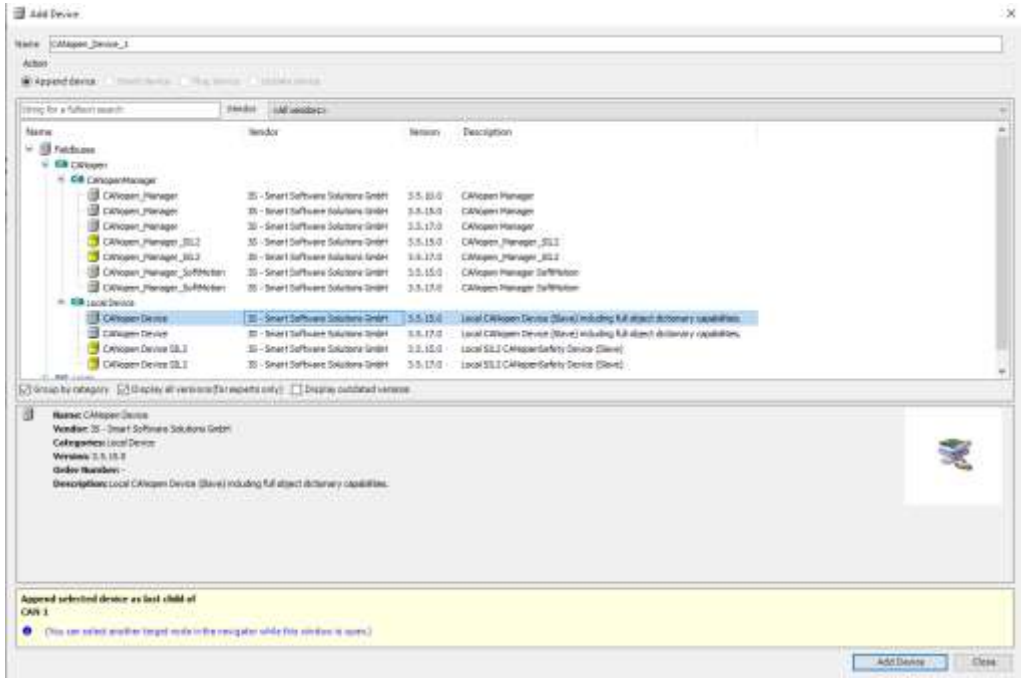
与主站操作相同，请参考本文档内相关内容。



# CT65模块应用

- **IFM6·2 – CANopen Slave**
- 添加CANopen Device

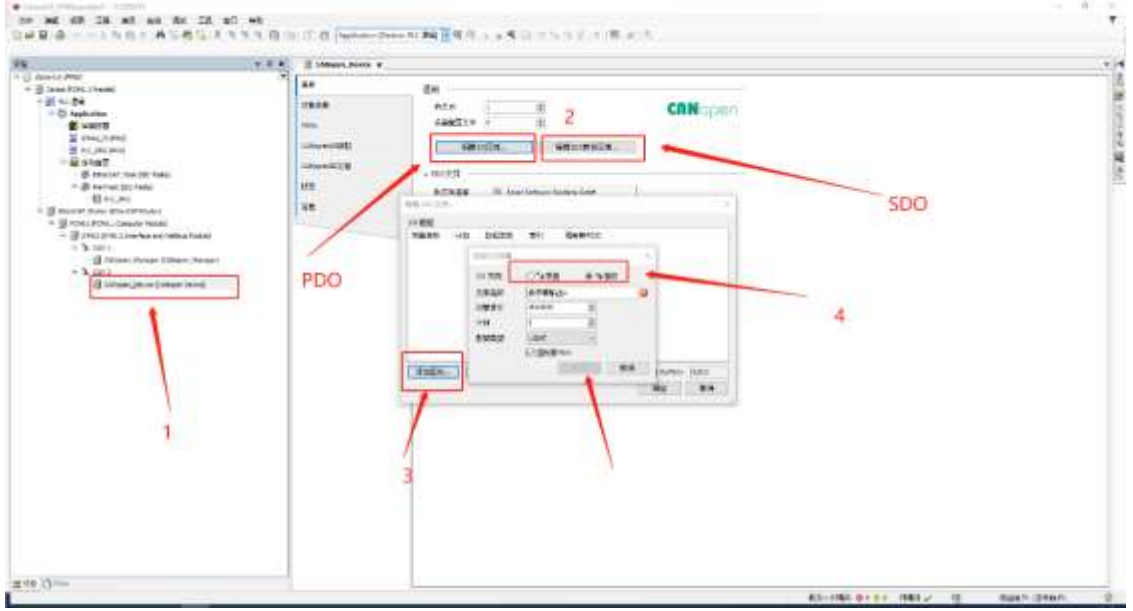
在CAN port上右击，选择Add Deive，在弹出的设备选择窗口中选中CANopen – Local Device – CANopen Device设备，点击窗口下方的Add Device按钮，即添加了CANopen子站。



# CT65模块应用

- **IFM6·2 – CANopen Slave**
- 添加PDO / SDO

在CANopen Device的General页，点击Edit I/O Arean按钮可编辑PDO，点击Edit设置SDO。点击添加，设置接收或者发送，并输入名称，该名称不可包含中文，若包含中文，则确定键为灰色。配置数据类型以及是否强制新PDO，在配置完成后，点击确定，完成PDO或SDO的添加。



# CT65模块应用

- **IFM6·2 – CANopen Slave**
- 添加PDO / SDO

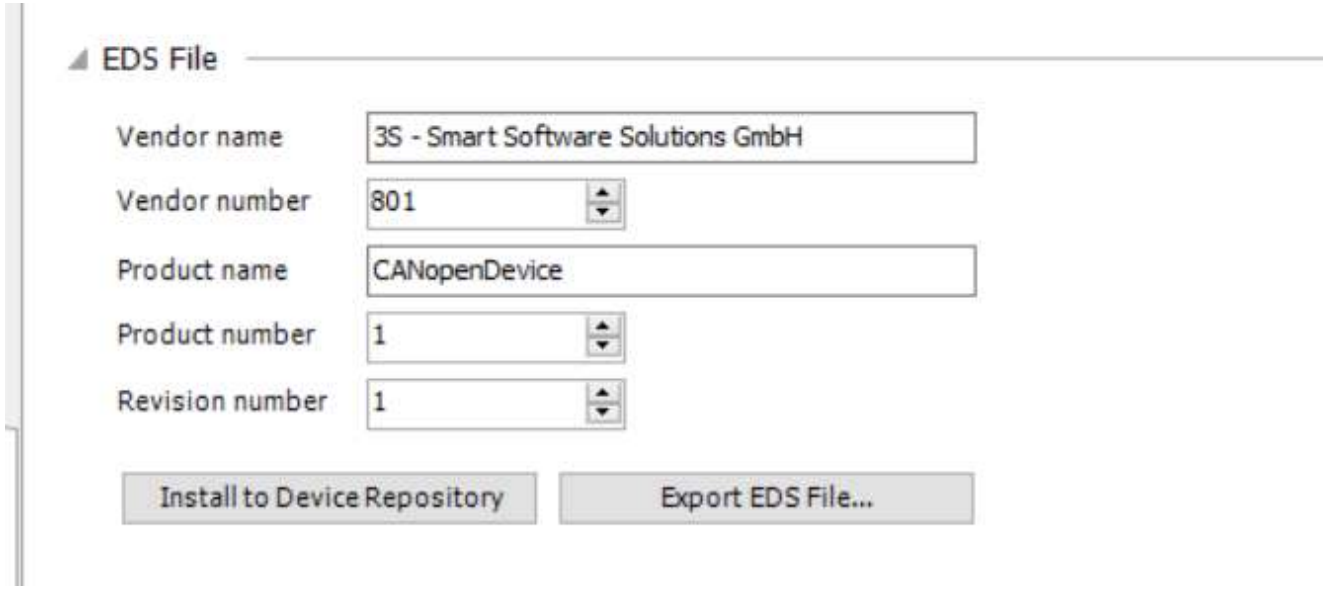
PDO和SDO配置页面描述可参考下表:

| 名称 | Name | 描述 |
|---|---|---|
| **发送** | Transmit | 主站接收数据，子站发送数据 |
| **接收** | Receive | 主站发送数据，子站接收数据 |
| **范围名称** | Range name | 自定义数据名称 |
| **计数** | Count | 选择每组数据数量 |
| **数据类型** | Data type | 选择每组数据类型 |
| **强制新PDO** | Force new PDO | 勾选表示新建一组数据，否则在原数据组添加 |

# CT65模块应用

- **IFM6·2 – CANopen Slave**
- 生成EDS文件

在General页面，点击Export EDS file导出从站的 EDS 文件，以提供给主站配置使用。Vendor name、Product name 等信息可根据实际情况 进行配置。



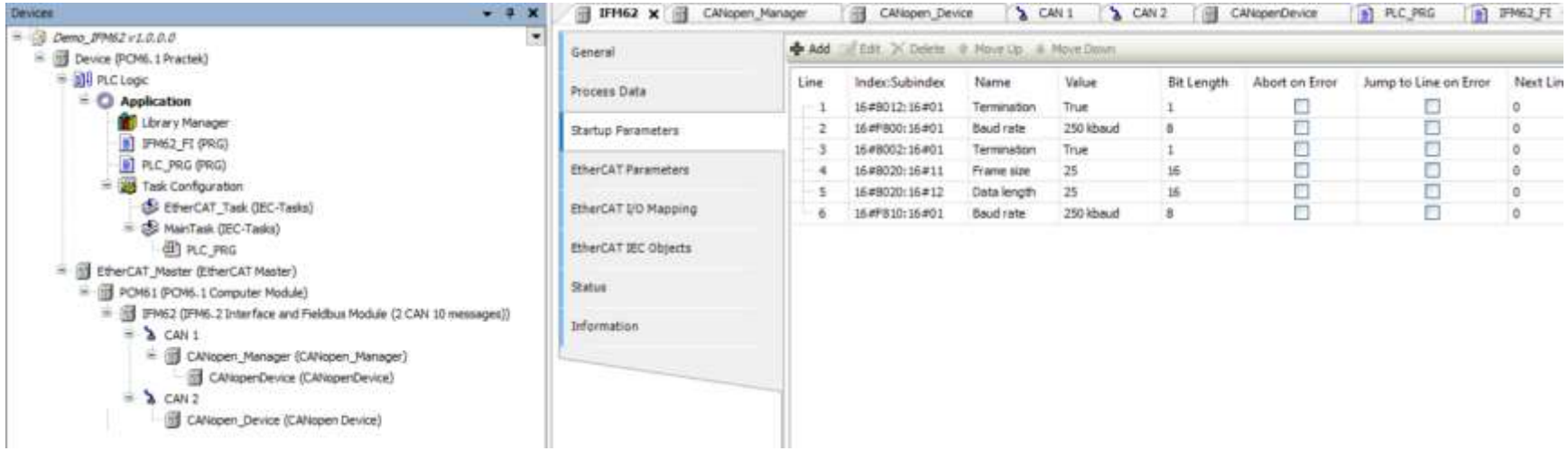
# CT65模块应用

- **IFM6·2 – CANopen Slave**
- 为CANopen device链接变量

在CANopen I/O Mapping页中，可为添加的PDO / SDO链接变量，即可在程序中读取由主站发送给子站的PDO / SDO数据；写入由子站发送给主站的PDO / SDO数据。

# CT65模块应用

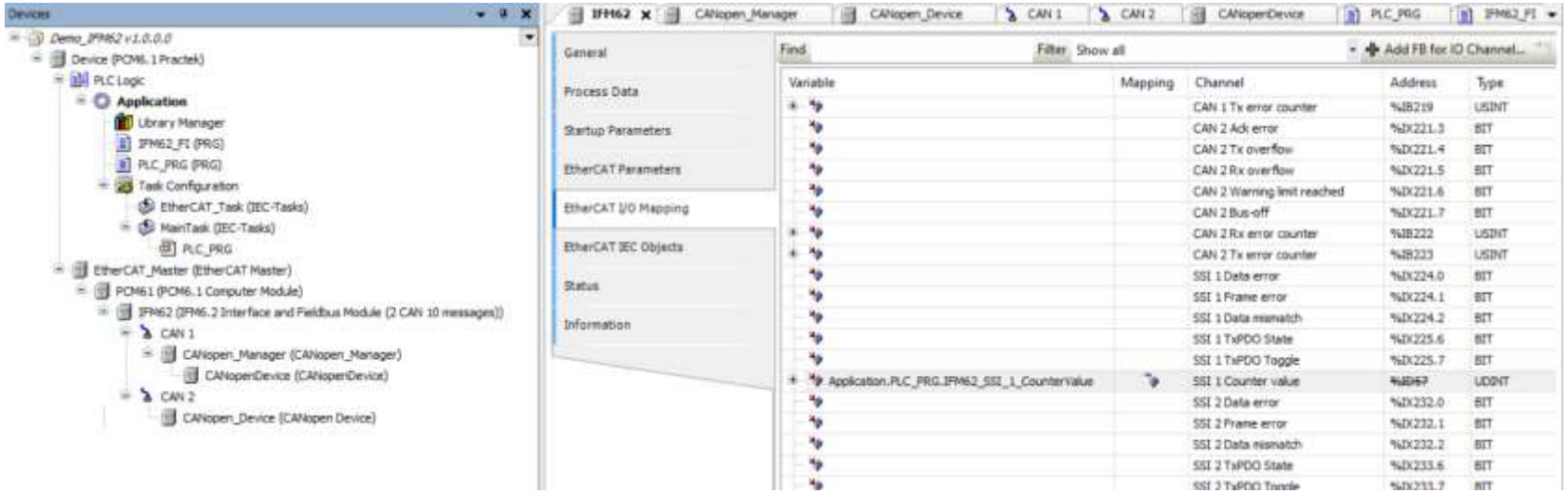
- IFM6·2 – SSI
- 设置启动参数




请按照实际设备的情况，设置对应的启动参数，如上图所示，设置SSI通信波特率250k，数据长度25位。

# CT65模块应用

- IFM6·2 – SSI
- 链接变量并使用

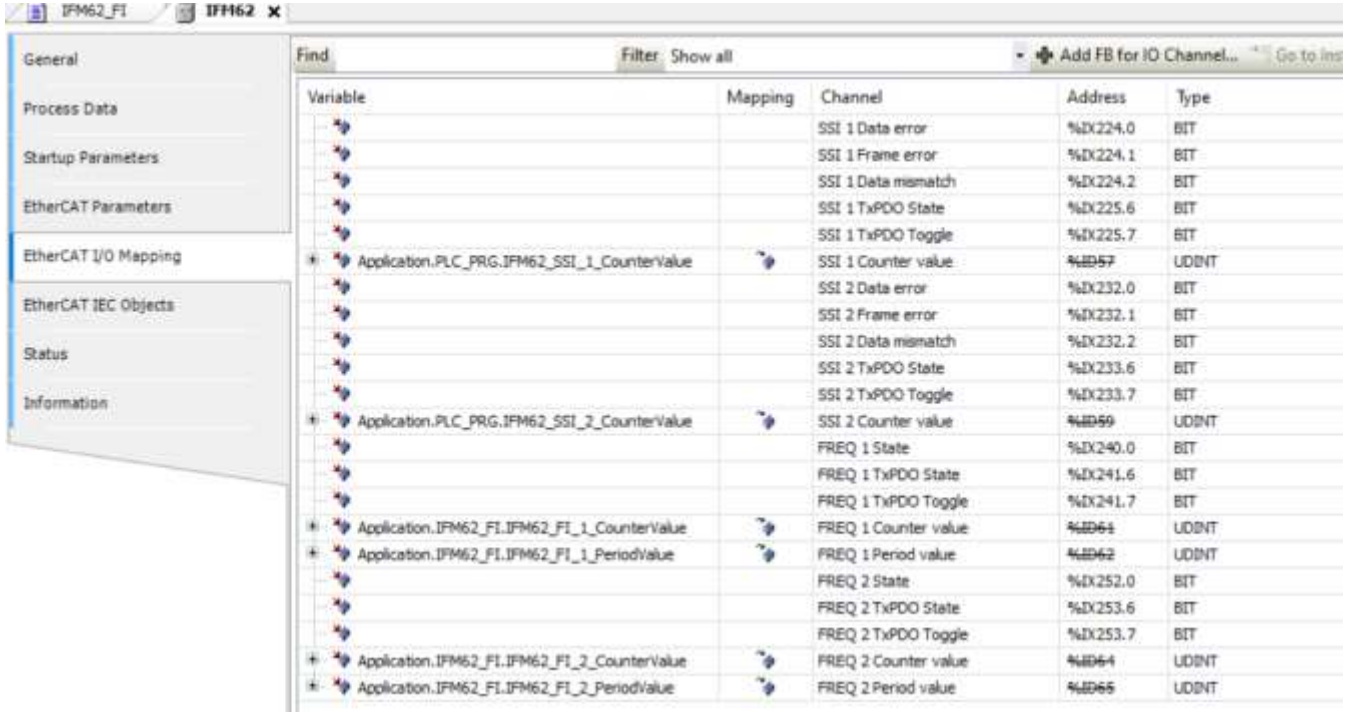



在EtherCAT I/O Mapping页中链接变量SSI Counter value，该变量即为传感器传回的编码器数值，正确的解析该变量即可获得编码器位置。

# CT65模块应用

## ➢ IFM6·2 – FI

通过Ethercat I/O Mapping页中FREQ Counter value及FREQ Period value通道数值即可计算FI信号的频率(Hz)及转速值(RPM)。



# CT65模块应用

## ➢ IFM6·2 – FI

FI计算代码示例如下：

IFM62_FI × IFM62

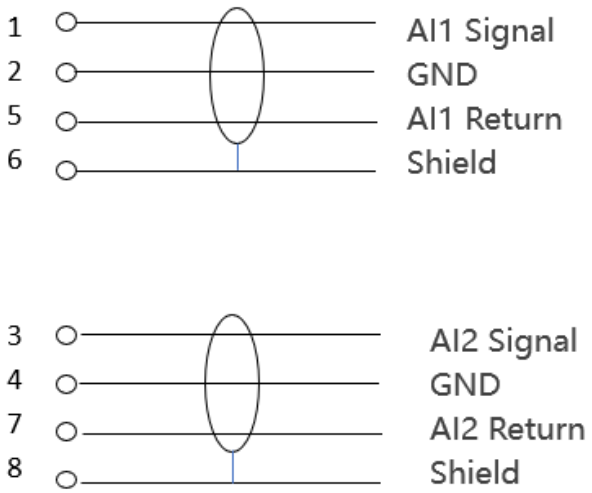
```
PROGRAM IFM62_FI
VAR
    IFM62_FI_1_CounterValue : UDINT;
    IFM62_FI_2_CounterValue : UDINT;
    IFM62_FI_1_PeriodValue : UDINT;
    IFM62_FI_2_PeriodValue : UDINT;

IF Last_CounterValue <> IFM62_FI_1_CounterValue THEN
    IF IFM62_FI_1_PeriodValue <> 16#FFFFFFFF AND  IFM62_FI_1_PeriodValue <> 16#00000000 THEN
        rSimSpeed_HZ :=100000000.0/UDINT_TO_LREAL(IFM62_FI_1_PeriodValue);   (*  f[Hz] =1/(10*10^-9*CounterValue)    *)
        rSimSpped_RPM :=  rSimSpeed_HZ *60.0/EncoderPPR;    (*Encoder  Pulse Per Revolution*)
    END_IF
    SpeedTimeoutCnt := 100;   (*Reset timeout for pulses   100*0.010 sec = 1sec    0.010sec is program scan period *)
ELSE
    IF SpeedTimeoutCnt  > 0 THEN
        SpeedTimeoutCnt := SpeedTimeoutCnt -1;
    ELSE
        rSimSpped_RPM := 0.0;
    END_IF
END_IF
Last_CounterValue:=IFM62_FI_1_CounterValue;
```

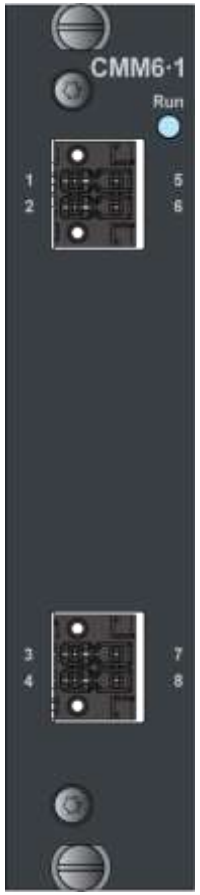
当FI有新的上电沿产生时，FREQ Counter value产生变化，新的周期及转速值可计算。FREQ Period value值为两个FI信号上电沿之间的时间片计数，单个时间片为10ns，故如上图所示，使用10E8除以Period Value值即可得FI信号频率值，通过频率值和传感器单圈信号分辨率(PPR)即可算得实时转速。

# CT65模块应用

## ➢ CMMM6·1

CMM6.1有两路高频模拟输入。



# CT65模块应用

- **CMMM6·1**

- 启动参数

CMM6.1需要设置4个启动参数，分别为Input type、Sensor excitation、Input range、Sample frequency。启动参数的详细可参见下表:

| 索引名 | 子索引名 | 类型 | 枚举名 | 值 | 描述 |
|---|---|---|---|---|---|
| **Input type Ch1~2** | Input type | USINT (8位) | AC mode | 0 | 设置外界传感器输入信号（交流或直流） |
| | | | DC mode | 1 | |
| **Sensor excitation Ch1~2** | Sensor excitation | USINT (8位) | 0 mA | 0 | 设置传感器响应电流。 |
| | | | 2 mA | 1 | |
| | | | 4 mA | 2 | |
| | | | 6 mA | 3 | |

# CT65模块应用

- **CMMM6·1**
- 启动参数

| 索引名 | 子索引名 | 类型 | 枚举名 | 值 | 描述 |
|---|---|---|---|---|---|
| **Input range Ch1~2** | Input range | USINT (8位) | -10...+20V DC | 0 | 需设置测量范围。 |
| | | | -10…10V | 1 | |
| | | | -5…+5V | 2 | |
| | | | -2.5…+2.5V | 3 | |
| | | | -1.25…+1.25V | 4 | |
| | | | -0.62…+0.62V | 5 | |
| | | | -0.31…+0.31V | 6 | |
| | | | -0.16…+0.16V | 7 | |
| | | | -0.08…+0.08V | 8 | |
| | | | -0.04…+0.04V | 9 | |
| | | | -0.02…+0.02V | 10 | |

# CT65模块应用

- **CMMM6·1**

- 启动参数

| 索引名 | 子索引名 | 类型 | 枚举名 | 值 | 描述 |
|---|---|---|---|---|---|
| **Sample frequency Ch1~2** | Sample frequency | USINT（8位） | 58.6KSMPS | 0 | 设置采样频谱。 |
| | | | 39.3KSMPS | 1 | |
| | | | 11.7KSMPS | 2 | |
| | | | 5.86KSMPS | 3 | |
| | | | 2.34KSMPS | 4 | |
| | | | 1.17KSMPS | 5 | |
| | | | 586SMPS | 6 | |
| | | | 234SMPS | 7 | |
| | | | 117SMPS | 8 | |
| | | | 58.6SMPS | 9 | |
| | | | 29.3SMPS | 10 | |
| | | | 11.7SMPS | 11 | |

# CT65模块应用

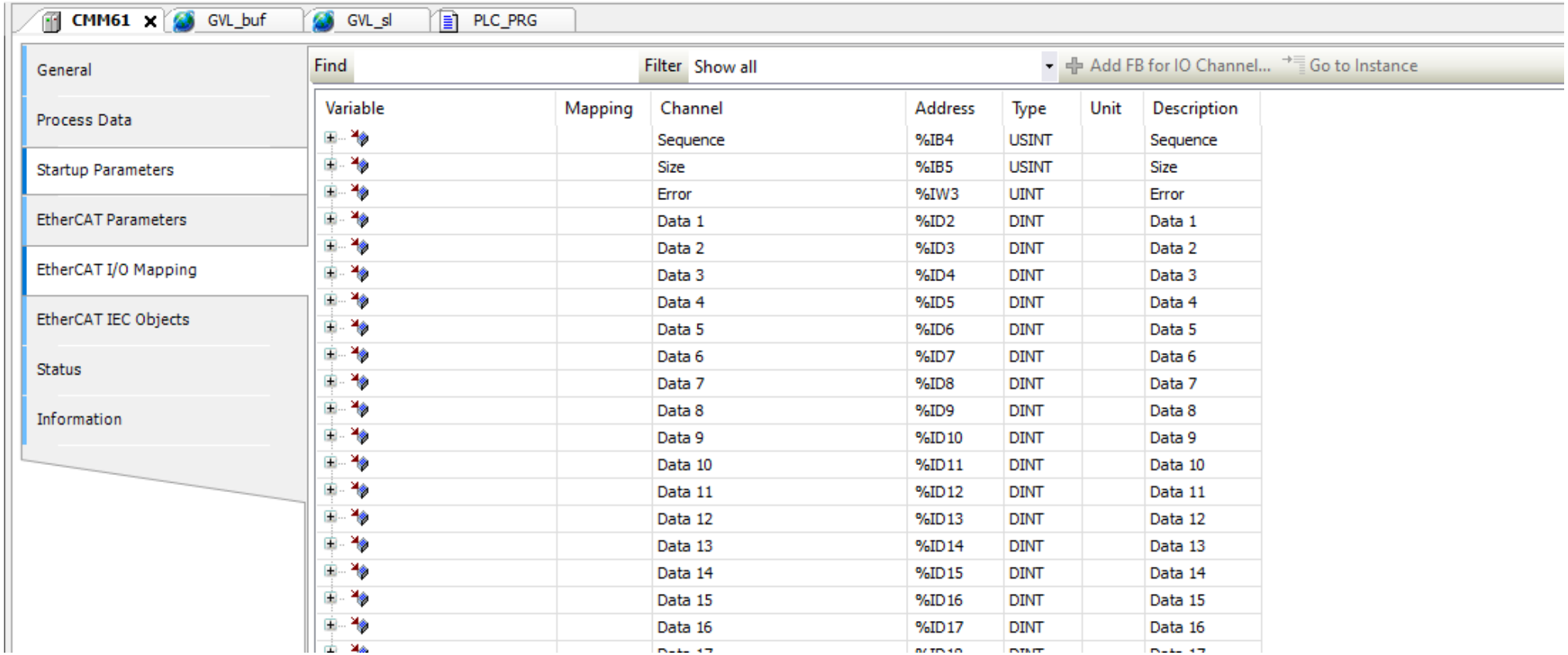
- **CMMM6·1**
- 变量映射

CMM6.1提供如下图所示数据通道，各通道含义如下所示

- Sequence: 数据序列值，当数据更新时，该数值自增
- Size: 更新数据数量，比如Size值为50，代表Data1~Data50中数值为本次更新数值
- Error: 错误标识位，当CMM6.1模块运行/采集数据错误时，Error数值为TRUE
- Data: 数据数组，Data1~Data122，提供振动数据通道。

# CT65模块应用

- CMMM6·1
- 变量映射

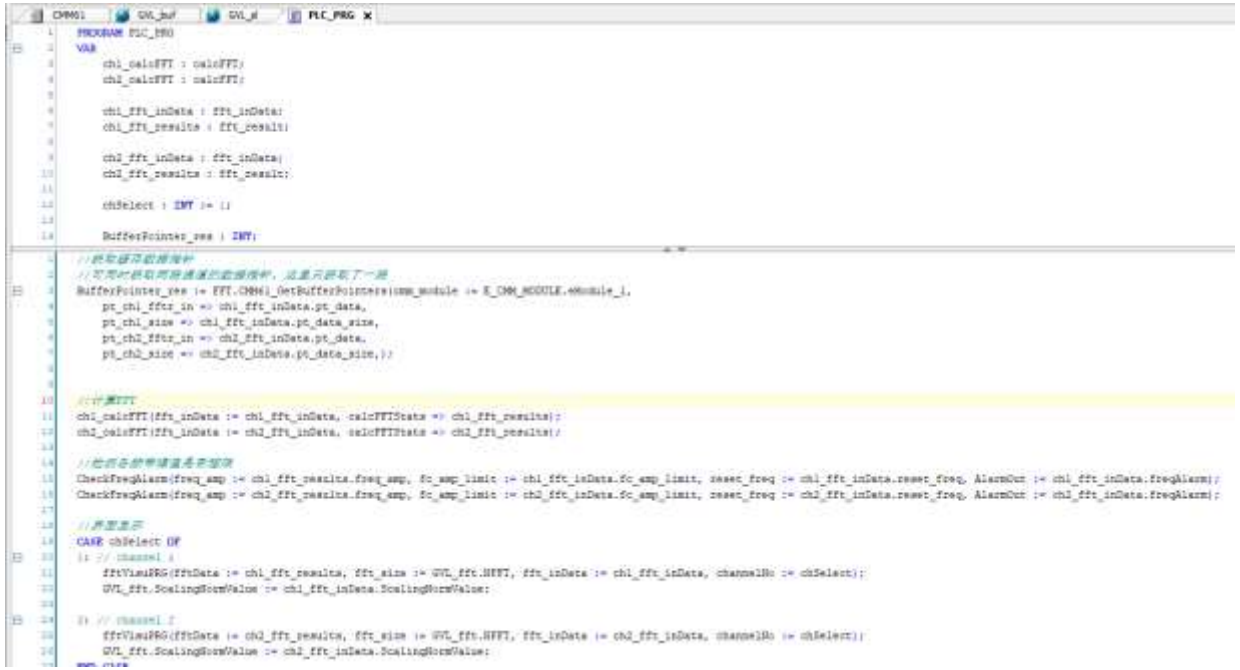



| Variable | Mapping | Channel | Address | Type | Unit | Description |
|---|---|---|---|---|---|---|
| | | Sequence | %IB4 | USINT | | Sequence |
| | | Size | %IB5 | USINT | | Size |
| | | Error | %IW3 | UINT | | Error |
| | | Data 1 | %ID2 | DINT | | Data 1 |
| | | Data 2 | %ID3 | DINT | | Data 2 |
| | | Data 3 | %ID4 | DINT | | Data 3 |
| | | Data 4 | %ID5 | DINT | | Data 4 |
| | | Data 5 | %ID6 | DINT | | Data 5 |
| | | Data 6 | %ID7 | DINT | | Data 6 |
| | | Data 7 | %ID8 | DINT | | Data 7 |
| | | Data 8 | %ID9 | DINT | | Data 8 |
| | | Data 9 | %ID10 | DINT | | Data 9 |
| | | Data 10 | %ID11 | DINT | | Data 10 |
| | | Data 11 | %ID12 | DINT | | Data 11 |
| | | Data 12 | %ID13 | DINT | | Data 12 |
| | | Data 13 | %ID14 | DINT | | Data 13 |
| | | Data 14 | %ID15 | DINT | | Data 14 |
| | | Data 15 | %ID16 | DINT | | Data 15 |
| | | Data 16 | %ID17 | DINT | | Data 16 |

# CT65模块应用

- **CMMM6·1**
- 代码示例

用户需要通过Sequence变化判断是否有数据更新，通过Size数值判断当前循环更新数据数量，再从Data中读取正确长度的数据。然后对累积的数据进行FFT等频谱分析，福氏提供了FFT库进行相应计算。



PRACTEK

远见 · 互重 · 雄心