

BECKHOFF GUANGZHOU

# TwinCAT 3.0 入门

---

LizzyChen

Update: 2015-09-25

## Version 1.01

安装文件及培训资料下载: <http://pan.baidu.com/s/1gd1zbnN>

(本文最新版本: )

更新记录:

2015.09.23 V1.01 增加 OOP 编程。

2015.09.08 基于《TwinCAT 2.0 从入门到精通 V2.01》修改

## 0. 1 本书读者对象

- Beckhoff 的 CX、CPxxxx、Cxxxx 系列控制器的用户。  
这些用户的共同点是，控制软件已经预装在订购的控制器上，用户需要用自己的电脑对控制器进行编程。控制器是基于 PC 的架构，并安装 Windows 操作系统。书中表述的 CX、CX 控制器、控制器，是由于文字编辑时期不同，表述有所差别，实际所指适用于所有基于 Windows 平台的 TwinCAT 控制系统。
- TwinCAT3.0 软件用户。  
这些用户的特点是，TwinCAT 控制软件需要自己安装在运行 Windows7 或者 Windows 操作 XP 系统的工控机上。用户可以在工控机上编程，也可以用自己的笔记本电脑对工控机进行编程。

## 0. 2 本书主要内容

本书基于 TwinCAT2.0 的两本教程《TwinCAT 2.0 从入门到精通》和《TwinCAT NC PTP 实用教程》。TwinCAT 3.0 中的 NC PTP 部分几乎完全兼容 TwinCAT 2.0，所以《TwinCAT NC PTP 实用教程》仍然适用于 TwinCAT 3 的用户。

本书讲解 TwinCAT3.0 的系统配置、PLC 编程、各种 Beckhoff 硬件、常用控制功能、通讯功能的实现。本书不涉及 TwinCAT 3.0 下的 C 语言编程和 Matlab 仿真功能，所以书名改为《TwinCAT 3.0 入门》。

- 上册为《TwinCAT 3.1 入门》的内容为：
  - 第 1 章，系统概述，包括原理、选型、安装和接线
  - 第 2 章，编程入门，实现用最简单的程序控制一套硬件。
  - 第 3 章，TwinCAT 3 开发环境的深入介绍，不做练习，仅供查询。
  - 第 4 章，TwinCAT 3 扩展功能，面向对象编程，制作库文件等。
  - 第 5 章，操作系统和硬件，包括系统备份、桌面接管等工具。
  - 第 6 章，常用功能：包括掉电保持、数据存储、配方功能等
  - 第 7 章，TwinCAT 库文件，重点介绍温控、PID、OS 功能扩展、EtherCAT 诊断和配置
  - 第 8 章，连接 IO 模块，介绍各种 IO 模块的特殊用法。
  - 第 9 章，连接其它 TwinCAT 系统，包括 ADS 通讯和 Realtime Ethernet
  - 第 10 章，连接第三方设备，包括与仪表、驱动、触摸屏、视觉系统的各种通讯。
  - 第 11 章，连接第三方 PLC，TwinCAT 作为现场总从站集成到其它 PLC 中。
  - 第 12 章，HMI 解决方案，包括触摸屏、组态软件、高级语言程序和 TwinCAT HMI
  - 第 13 章，连接企业数据库，通过 Tc Database Server 实现 PLC 与数据库的通讯。
  - 第 14 章，特殊 IO 模块（待填充）
  - 第 15 章，从 TwinCAT 2 到 TwinCAT 3。讲解 Tc 3.0 与 Tc 2.0 的区别。
  - 第 16 章，附录，包括 PLC 编程手册、简明安装指南、Codesys 中文帮助。

## 0. 3 本书的使用方法

- 项目考察阶段，可阅读“第 1 章，系统概述”，以及本书目录。

- 初学者必须依次阅读第 2 章的所有小节。
- 功能测试阶段，可根据目录找到相应的章节，每个章节在“配套文档”中都有对应的文件夹，里面有相关的例程、工具、文档说明。
- 项目开发阶段，预先阅读“第 3 章，TwinCAT 3 开发环境的深入介绍”及“第 4 章 和 TwinCAT 3 扩展功能”，并根据所使用的 IO 模块和设备，详细阅读第 8 章的相关内容。
- 项目结束阶段，可查阅第 5 章，操作系统和硬件。

## 0.4 版本说明

本书所提供的操作截图、程序代码都基于 VS Shell 2013 下的 TC3.1.4018.5。截至目前，由于 BECKHOFF 公司的 TwinCAT 软件仍然会持续升级和更新，我们不排除后续版本的操作界面会发生变化，而例程中的代码也有可能不适用于后续版本。

由于本书还在编辑阶段，作者对于 TwinCAT 3 的使用经验也还在积累，升级此书的目的是为了众多 TwinCAT 2 的用户能够顺利切换到 TwinCAT 3，并充分发挥多核 CPU、64 位操作系统的性能，以及 TwinCAT 3 开发工具的新功能。

接下来还需要对文字进行整理，修饰，对配套文档进行精简、条理化。那将是一个漫长的过程，视工作繁忙程度而定。欢迎对本书的结构、内容提出意见和建议，请发邮件至 [L.Chen@Beckhoff.com.cn](mailto:L.Chen@Beckhoff.com.cn)。

作者 2015-09-03 于广州

## 0. 5 更新记录

2015.09.23 第 1 遍 第 5 章 增加 OOP 编程的内容

2015.09.08 第 1 遍 基本内容更新至 TwinCAT 3

第 1 章 系统概述

第 2 章 编程入门

第 3 章 TC3 开发环境的深入介绍

第 4 章 TC3 扩展功能

第 12 章 HMI 解决方案（升级 TC3 HMI 部分）

第 15 章 从 TwinCAT 2 到 TwinCAT 3

其它章节沿用《TwinCAT 2.0 从入门到精通\_V2.01\_1112》

# 目 录

1. 系统概述.....	14
1.1. TwinCAT 3 Runtime 的运行条件 .....	14
1.2. TwinCAT 3 功能介绍 .....	15
1.2.1. TwinCAT PLC 的实时性.....	17
1.2.2. TwinCAT PLC 的数据区.....	17
1.2.3. TwinCAT PLC 的数据存储.....	18
1.2.4. TwinCAT 与外设 IO 的连接.....	19
1.3. 选型设计 .....	21
1.3.1. 控制器.....	21
1.3.2. 系统扩展模块.....	25
1.3.3. I/O 系统 .....	26
1.4. 安装和接线 .....	28
2. 编程入门.....	30
2.1. 概述 .....	30
2.2. 在编程 PC 上安装 TwinCAT 开发环境 .....	30
2.2.1. 在 PC 上安装 TwinCAT 开发环境.....	30
2.2.2. 升级 TwinCAT 开发环境.....	31
2.2.3. 在 TC3 和 TC2 之间切换.....	31
2.3. 初步认识开发环境.....	32
2.3.1. TC 3 图标和 TC 3 Runtime 的状态.....	32
2.3.2. TC 3 快捷菜单的功能.....	32
2.3.3. 启动 TC3 的帮助系统.....	33
2.3.4. TC3 Quick Start 教程 .....	34
2.3.5. 启动示例程序.....	34
2.3.6. 获取和注册正版授权.....	34
2.4. 编程准备：添加路由（Add ADS Router） .....	37
2.4.1. 设置 IP 地址 .....	37
2.4.2. 设置 NetID.....	38
2.4.3. 在 TC3 的 System   Routes 中添加路由 .....	38
2.5. 开发第一个 PLC 项目 .....	43
2.5.1. 建一个 TwinCAT 项目 .....	43
2.5.2. 建一个 PLC 项目 .....	47
2.5.3. PLC 变量映射和激活配置 .....	52
2.5.4. 设置开机自启动.....	55
2.6. 上传、下载和比较.....	58
2.6.1. PLC 程序的上传下载和比较 .....	58
2.6.2. TwinCAT 项目的上传和比较 .....	59
3. TwinCAT 3 开发环境的深入介绍 .....	60
3.1. 变量声明 .....	61
3.1.1. 变量声明的基本语法为.....	61
3.1.2. 变量类型.....	62

3.1.3.	变量声明中的绝对地址.....	64
3.1.4.	变量声明中的赋初值.....	64
3.1.5.	为 IO 变量自动分配地址.....	65
3.1.6.	变量声明时的缩写输入法.....	65
3.1.7.	变量的属性.....	66
3.1.8.	地址对齐.....	68
3.2.	编程语言和新增功能.....	69
3.2.1.	ST 中增加了 Continue 和 Jump 语句。.....	69
3.2.2.	指令: BitAdr(),用于定位到 Bit。.....	69
3.2.3.	UML ChartSate 编程.....	70
3.2.4.	指针操作增加.....	70
3.2.5.	支持变量作为 Bit 值访问.....	71
3.2.6.	程序注释.....	71
3.3.	诊断和调试功能.....	72
3.3.1.	搜索和替换按钮.....	72
3.3.2.	TwinCAT Live Watch 怎么用.....	72
3.3.3.	ADS Symbol Watch 怎么用?.....	73
3.3.4.	Command Window.....	73
3.3.5.	独立于程序的 Watch List.....	74
3.3.6.	Clean 之后不能再 Online Change.....	74
3.4.	任务和程序.....	74
3.4.1.	新建任务.....	75
3.4.2.	把程序指定到任务.....	76
3.4.3.	编译和试运行.....	79
3.4.4.	为 Task 指定 CPU、优先级、周期等等。.....	81
3.4.5.	Task with image:.....	83
3.4.6.	IO at task beginning.....	84
3.4.7.	Task 特别提示.....	84
3.5.	隐含的变量和函数.....	84
3.5.1.	TwinCAT_SystemInfoVarList.....	84
3.5.2.	除零等校验.....	85
3.5.3.	隐含的函数.....	86
3.6.	编程环境的设置.....	87
3.6.1.	TwinCAT 快捷键.....	87
3.6.2.	Smart Coding.....	88
3.6.3.	编程环境的其它设置.....	88
3.7.	兼容 TC2 的功能.....	90
3.7.1.	多语言混合编程 (Action).....	90
3.7.2.	可供使用的操作符、函数和功能块.....	92
3.7.3.	结构和枚举.....	103
3.7.4.	数组和指针.....	107
3.7.5.	项目加密和对象加密 (TC2).....	108
3.8.	禁止 TwinCAT 的开机自启动.....	113

3.9.	其它提示 .....	114
3.9.1.	弹出窗和提示。 .....	114
3.9.2.	TC2 的控制器可以刷 TC3 的 IMAGE 试用 .....	114
4.	TwinCAT 3 扩展功能 .....	115
4.1.	引用库文件 .....	115
4.1.1.	Add Library .....	115
4.1.2.	自定义库文件 .....	115
4.1.3.	引用别人的库 .....	120
4.1.4.	命名空间 .....	120
4.2.	Measurement .....	121
4.2.1.	TC3 Scope Server, 免费版与收费版的功能差别 .....	121
4.2.2.	Scope 导出数据 .....	121
4.2.3.	常见问题 .....	122
4.3.	程序归档 .....	123
4.3.1.	Measurement 项目的存储路径 .....	123
4.3.2.	TwinCAT 项目的存储路径 .....	123
4.3.3.	项目打包和解包 .....	124
4.3.4.	PLC 程序的打包和解包 .....	125
4.3.5.	FB 等对象的导出和导入 .....	126
5.	面向对象编程 .....	127
5.1.	概述 .....	127
5.1.1.	什么是面向对象编程 .....	127
5.1.2.	关键名词: Function Block 和 Interface .....	127
5.1.3.	关键词: Extend .....	128
5.1.4.	面向对象编程的 3 个用法 .....	130
5.2.	建立一个带 Method 和 Property 的 FB .....	130
5.2.1.	示例 .....	130
5.2.2.	关于 Method 和 Property 的 FAQ .....	136
5.3.	建立一个 FB 的扩展 FB (Extend) .....	138
5.3.1.	示例 .....	138
5.3.2.	关于 Extend 的 FAQ .....	143
5.4.	建立一个 Interface 并实现 (Impement) .....	143
5.4.1.	示例 .....	143
5.4.2.	关于 Interface 的 FAQ .....	149
5.5.	其它说明 .....	150
5.5.1.	静态变量 (Static) 和临时变量 (Temp) .....	150
5.5.2.	特殊的 Method: FB_Init, FB_Exit, FB_Reinit .....	150
6.	操作系统和硬件 .....	151
6.1.	概述 .....	151
6.2.	Windows CE 操作系统 .....	152
6.2.1.	系统备份和还原 .....	152
6.2.2.	远程桌面连接 .....	153
6.2.3.	中文语言包的安装 .....	154

6.2.4.	开启 FTP Server 与 PC 交换文件.....	155
6.2.5.	显示器分辨率设置及屏幕校准.....	158
6.3.	Windows XPe 及 Windows XP 操作系统.....	159
6.3.1.	系统备份和还原.....	159
6.3.2.	远程桌面连接.....	162
6.3.3.	中文语言包的安装.....	162
6.3.4.	突然断电对操作系统的损坏.....	164
6.3.5.	经共享文件夹与 PC 交换文件.....	168
6.3.6.	显示器分辨率设置及屏幕校准.....	168
6.4.	UPS 硬件.....	168
6.4.1.	CX 系列的 UPS.....	168
6.4.2.	IPC 上的 UPS.....	168
6.4.3.	CX50xx 上的 1s-UPS.....	168
7.	数据存储、配方和文件 (TC2).....	169
7.1.	掉电保持数据.....	170
7.1.1.	用 Persistent 变量实现掉电保持.....	170
7.1.2.	用 NOVRAM 区实现变量的掉电保持.....	174
7.1.3.	清除、备份和恢复 NOVRAM 区的数据.....	178
7.2.	数据存储到文件.....	179
7.2.1.	读写二进制文件.....	180
7.2.2.	读写 CSV 文件.....	182
7.2.3.	读写 wtc 文件.....	182
7.2.4.	读写 XML 文件.....	192
7.3.	配方功能.....	194
7.3.1.	用 XML 文件实现配方.....	194
7.3.2.	用 wtc 文件实现配方.....	194
7.3.3.	用 Persistent 变量实现配方.....	195
8.	TwinCAT 库文件.....	196
8.1.	温控库.....	196
8.1.1.	简介.....	196
8.2.	控制工具箱 TcPlcControllerToolbox.lib.....	196
8.2.1.	滤波.....	196
8.2.2.	PID.....	198
8.2.3.	PWM 输出.....	199
8.2.4.	设定点发生器 SetpointGeneration.....	200
8.3.	调用 Windows 和 TwinCAT 功能的库 TcUtility.lib.....	201
8.3.1.	调用 Windows 的功能.....	201
8.3.2.	读取 IP 地址和修改注册表.....	202
8.3.3.	启动和停止应用程序.....	203
8.3.4.	内存操作.....	204
8.3.5.	调用 TwinCAT System Manager 的功能.....	205
8.3.6.	BCD 码转换.....	205
8.4.	EtherCAT 主站和从站的控制 TcEtherCAT.lib.....	206

8.4.1.	EtherCAT 状态切换.....	206
8.4.2.	EtherCAT 从站的参数设置.....	210
8.4.3.	EtherCAT 数据包统计.....	211
8.4.4.	EtherCAT 诊断.....	211
8.5.	其它有用的库.....	211
9.	I/O 模块、总线主站和 EtherCAT.....	212
9.1.	KL 模块.....	212
9.1.1.	KL 模块的 Process Data.....	213
9.1.2.	KL 模块的参数设置.....	214
9.1.3.	KL 模块的错误诊断和恢复.....	218
9.2.	EL 模块.....	218
9.2.1.	EL 模块的 Process Data——控制信号.....	218
9.2.2.	EL 模块的 Process Data——EtherCAT 诊断信息.....	219
9.2.3.	EL 模块的参数访问.....	221
9.2.4.	EL 模块的错误诊断和优化配置.....	226
9.2.5.	EtherCAT 从站升级 Firmware.....	227
9.3.	现场总线主站.....	229
9.3.1.	概述.....	229
9.3.2.	Profibus DP.....	230
9.3.3.	CanOpen.....	235
9.3.4.	Realtime Ethernet.....	250
9.4.	EtherCAT.....	253
9.4.1.	配置 EtherCAT 主站.....	253
9.4.2.	配置 EtherCAT 从站.....	253
9.4.3.	优化 EtherCAT 网络——同步单元配置.....	254
9.4.4.	优化 EtherCAT 网络——设置热连接 Hot Connect.....	257
9.4.5.	优化 EtherCAT 网络——设置网络冗余 Redundancy.....	259
10.	TwinCAT 之间的实时和非实时通讯.....	262
10.1.	概述.....	262
10.2.	ADS 通讯协议.....	262
10.2.1.	ADS 设备的识别.....	262
10.2.2.	ADS Router 路由表.....	263
10.2.3.	ADS 设备的数据访问.....	267
10.2.4.	从 PLC 程序实现 ADS 通讯.....	269
10.2.5.	从高级语言实现 ADS 通讯.....	269
10.3.	Realtime Ethernet.....	270
10.3.1.	运行 Realtime Ethernet 的软件和硬件要求.....	271
10.3.2.	配置 Realtime Ethernet 的步骤.....	271
10.4.	EtherCAT Slave.....	281
10.4.1.	EtherCAT 从站侧的设置.....	281
10.4.2.	EtherCAT 主站侧的设置.....	283
10.5.	桥接模块 EL6692.....	284
10.5.1.	网络拓扑图.....	284

10.5.2.	配置步骤.....	284
11.	与现场仪表 RS232/485 及 TCP/IP 通讯.....	286
11.1.	串行通讯的硬件准备.....	286
11.1.1.	串行通讯的物理接口.....	286
11.1.2.	Modbus RTU 通信介绍.....	286
11.2.	串口通讯的硬件准备.....	287
11.2.1.	硬件接线.....	287
11.2.2.	配置通讯接口.....	288
11.3.	TwinCAT PLC 的自由口通讯程序.....	291
11.3.1.	编写 PLC 程序.....	292
11.3.2.	在 PLC 变量和 Process Data 之间建立映射.....	296
11.3.3.	激活配置。.....	298
11.3.4.	调试 PLC 程序。.....	298
11.4.	TwinCAT Modbus RTU 程序.....	299
11.4.1.	作为 ModbusRTU Slave 与触摸屏通讯.....	299
11.4.2.	作为 ModbusRTU Master 与与温控表、变频器等通讯.....	301
11.4.3.	在 PLC 变量和 Process Data 之间建立映射.....	304
11.4.4.	激活配置。.....	306
11.4.5.	调试 PLC 程序。.....	306
11.5.	TwinCAT TCP/IP Server.....	307
11.5.1.	安装 Supplement.....	307
11.5.2.	编写 TCP/IP 通讯的程序.....	308
11.5.3.	自行编写 Tcp/IP 通讯程序.....	308
11.5.4.	借用 Tcp/IP 通讯的 Demo 程序.....	311
11.5.5.	引用 UDP 通讯的示例程序.....	314
11.5.6.	变量值转换成 BCD 字符串。.....	315
11.6.	TwinCAT Modbus TCP Server.....	316
11.6.1.	TwinCAT Modbus TCP Server 的安装.....	317
11.6.2.	Modbus TCP 地址与 PLC 地址的映射关系.....	317
11.6.3.	测试 TwinCAT Modbus TCP Server.....	319
11.7.	TwinCAT Modbus TCP Client.....	320
11.7.1.	引用 TcModbsuSrv.Lib.....	320
11.7.2.	调用 Modbus TCP Client 的功能块.....	320
11.7.3.	存盘, 编译。.....	322
11.7.4.	测试运行.....	322
12.	作为总线从站集成到第三方 PLC.....	325
12.1.	概述.....	325
12.2.	Profibus DP Slave.....	325
12.2.1.	EtherCAT 从站侧 (TwinCAT) 的设置.....	325
12.2.2.	Profibus DP 主站侧 (Siemens PLC) 的设置.....	329
12.2.3.	EL6731-0010 的诊断.....	333
12.3.	CanOpen Slave.....	334
12.4.	Profinet IO Slave.....	334

12.4.1.	概 述.....	334
12.4.2.	软件和硬件准备.....	335
12.4.3.	设置步骤: .....	335
12.5.	Ethernet IP Slave.....	340
12.5.1.	概述.....	340
12.5.2.	软件和硬件准备.....	340
12.5.3.	从站侧 (TwinCAT) 的配置.....	341
12.5.4.	主站侧 (RSLogix) 的配置.....	345
12.5.5.	结果验证.....	347
12.5.6.	FAQ.....	348
12.6.	DeviceNet Slave.....	348
12.6.1.	软件和硬件准备.....	349
12.6.2.	DeviceNet 从站侧 (TwinCAT) 的设置.....	349
12.6.3.	DeviceNet 主站侧的设置.....	351
12.6.4.	EL6752-0010 的诊断.....	354
13.	HMI 解决方案.....	357
13.1.	概 述.....	357
13.2.	经 RS232/S485 连接触摸屏.....	357
13.3.	经以太网连接触摸屏.....	357
13.4.	组态软件访问 TwinCAT PLC.....	357
13.5.	从高级语言访问 TwinCAT PLC.....	358
13.5.1.	概述.....	358
13.5.2.	ADS 通讯.....	358
13.5.3.	OPC 通讯.....	361
13.5.4.	TwinCAT IO 与 C++的实时通讯.....	372
13.6.	兼容 TC2 的 TwinCAT HMI.....	373
13.6.1.	概述.....	373
13.6.2.	基本图元编辑.....	373
13.6.3.	子画面的重复使用.....	376
13.6.4.	背景画面的重复使用。.....	381
13.6.5.	动态文本的显示.....	386
13.6.6.	实例 1:显示中文报警信息.....	388
13.6.7.	实例 2:用户管理器.....	388
13.7.	实例 3:TwinCAT 3 HMI.....	388
13.7.1.	添加画面.....	<b>Error! Bookmark not defined.</b>
13.7.2.	使 HMI 与 PLC 分离.....	403
13.7.3.	中文显示.....	405
13.7.4.	改变风格.....	405
14.	连接企业数据库.....	406
14.1.	概 述.....	406
14.2.	软件模型.....	407
14.3.	TwinCAT Database Server 的安装和配置.....	408
14.3.1.	软件安装.....	408

14.3.2.	Database Server 配置.....	408
14.3.3.	自动记录数据.....	415
14.3.4.	从 PLC 程序操作数据库.....	416
14.3.5.	建立和断开链接.....	416
14.3.6.	插入记录行.....	416
14.3.7.	数据库操作.....	417
14.4.	案例.....	418
14.4.1.	案例一.....	418
14.4.2.	案例二.....	418
15.	特殊 IO 模块.....	419
15.1.	测量模块.....	419
15.1.1.	电力测量模块 KL3403 和 EL3403.....	419
15.1.2.	称重模块 KL3356 和 EL3356.....	419
15.1.3.	示波器模块 KL3361.....	419
15.1.4.	示波器模块 KL3361.....	419
15.2.	运动控制模块.....	420
15.3.	XFC 超高速模块.....	421
15.3.1.	快速 IO 模块 EL1262/EL2262.....	421
15.3.2.	时间戳模块 EL1262/EL2262.....	421
15.3.3.	超采样模块 EL37x2/EL47x2.....	421
15.4.	高速脉冲输入及编码器接口模块.....	422
16.	从 TwinCAT 2 到 TwinCAT 3.....	423
16.1.	概述.....	423
16.2.	TC3 的新功能.....	423
16.2.1.	TC3 的继承性.....	423
16.2.2.	TC2 与 TC3 的适用范围.....	424
16.3.	TC2 转换 TC3 的解决方案.....	424
16.3.1.	先在 TwinCAT 2 中打包.....	424
16.3.2.	在 TwinCAT 3 中装载.....	425
16.3.3.	转换结果.....	427
16.3.4.	HMI 的转换.....	428
16.3.5.	TC3 ADS.....	428
16.4.	TwinCAT 3 实训文档.....	429
17.	附录.....	430
17.1.	附录 1: PLC Control 编程手册 2011.....	430
17.2.	附录 2: 简明安装手册.....	430
17.3.	附录 3: 常用 Lib 文件.....	430
17.4.	附录 4: CodeSys 中文帮助.....	430

# 1. 系统概述

TwinCAT 是德国 Beckhoff 公司的基于 PC 平台和 Windows 操作系统的控制软件。它的作用是把工业 PC 或者嵌入式 PC 变成一个功能强大的 PLC 或者 Motion Controller 控制生产设备。

1995 年 TwinCAT 首次推出市场，现存版本有两种：TwinCAT 2 和 TwinCAT 3，以下简称 TC2 和 TC3。TC2 是针对单 CPU 及 32 位操作系统开发设计的，其运行核不能工作在 64 位操作系统。对于多 CPU 系统，只能发挥单核的运算能力。TC3 考虑了 64 位操作系统和多核 CPU，并且可以集成 C++编程和 Matlab 建模，所以 TC3 的运行核既可以工作在 32 位操作系统，也可以工作在 64 位操作系统，并且可以发挥全部 CPU 的运算能力。对于 PLC 控制和运动控制项目，TC3 和 TC2 除了开发界面有所不同之外，编程、调试、通讯的原理和操作方法都几乎完全相同。

TwinCAT 是一套纯软件的控制器，完全利用 PC 标配的硬件，实现逻辑运算和运动控制。TwinCAT 运行核安装在 Beckhoff 的 IPC 或者 EPC 上，其功能就相当于 1 台计算机加上 1 个逻辑控制器“TwinCAT PLC”和 1 个和运动控制器“TwinCAT NC”。对于运行在多核 CPU 上的 TC3，还可以集成机器人等更多更复杂的功能。

TwinCAT PLC 的特点：与传统的 PLC 相比，CPU、存储器和内存资源都有了数量级的提升。运算速度快，尤其是传统 PLC 不擅长的浮点运算，比如多路温控、液压控制以及其它复杂算法时，TwinCAT PLC 可以轻松胜任。数据区和程序区仅受限于存储介质的容量。随着 IT 技术的发展，用户可以订购的存储介质 CF 卡、Cfast 卡、内存卡及硬盘的容量越来越大，CPU 的速度越来越快，而性价比越来越高。因此 TwinCAT PLC 在需要处理和存储大量数据比如趋势、配方和文件时优势明显。

TwinCAT NC 的特点：与传统的运动控制卡、运动控制模块相比，TwinCAT NC 最多能够控制 255 个运动轴，并且支持几乎所有的硬件类型，具备所有单轴点动、多轴联动功能。并且，由于运动控制器和 PLC 实际上工作于同一台 PC，二者之间的通讯只是两个内存区之间的数据交换，其数量和速度都远非传统的运动控制器可比。这使得凸轮耦合、自定义轨迹运动时数据修改非常灵活，并且响应迅速。TC3 虽然可以用于 64 位操作系统和多核 CPU，现阶段仍然只能控制 255 个轴，当然这也可以满足绝大部分的运动控制需求。

归根结底，TwinCAT PLC 和 TwinCAT NC 的性能，最主要还是依赖于 CPU。尽管 Beckhoff 的控制器种类繁多，无论是安装在导轨上的 EPC，还是安装在电柜内的 Cabinet PC，还是集成到显示面板的面板式 PC，其控制原理、软件操作都一模一样，同一套程序可以移植到任何一台 PC-Based 控制器上运行。移植后的唯一结果是 CPU 利用率的升高或者降低。

## 1.1. TwinCAT 3 Runtime 的运行条件

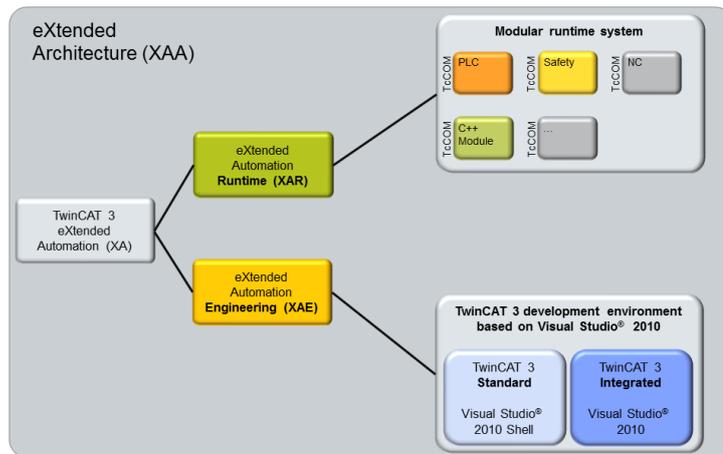
用户订购 Beckhoff 控制器时必须决定控制软件使用 TC2 还是 TC3 的运行核，软件为

出厂预装，用户不能自行更改。TC3 的运行核的控制器，必须使用 TC 3 开发版编程。

TwinCAT 运行核，分为 Windows CE 和 Windows Standard 两个版本，Windows Standard 版本包括 Windows XP、Windows Xpe、Windows NT、Windows 7、WES 7。由于 Windows CE 系统小巧轻便，经济实惠，相对于传统 PLC 而言，功能上仍然有绝对的优势，所以在工业自动化市场上，尤其是国内市场，Window CE 显然更受欢迎。

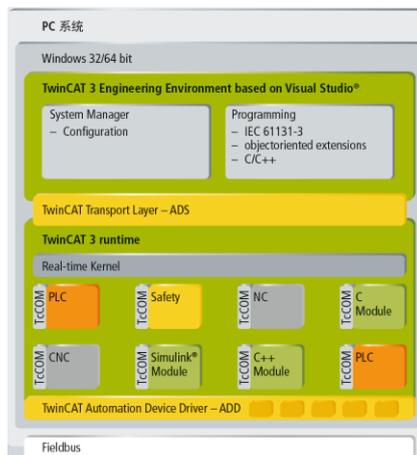
## 1.2. TwinCAT 3 功能介绍

TwinCAT 3 软件的结构可以用下图来描述：



TwinCAT 运行核是 Windows 底层优先级最高的服务，同时它又是所有 TwinCAT PLC、NC 和其它任务的运行平台。TC3 分为开发版（XAE）和运行版（XAR）。XAE 安装运行在开发 PC 上，既可以作为一个插件集成到标准的 Visual Studio 软件，也可以独立安装（with VS2010 Shell）。XAR 运行在控制器上的，必须要购买授权且为出厂预装。

在运行内核上，TC 3 首次提出了 TcCOM 和 Module 的概念。基于同一个 TcCOM 创建的 Module 有相同的运算代码和接口。TcCOM 概念的引入，使 TwinCAT 具有了无限的扩展性，Beckhoff 公司和第三方厂家都有可能把自己的软件产品封装成 TcCOM 集成到 TwinCAT 中。目前已经发布的 TcCOM 包括：



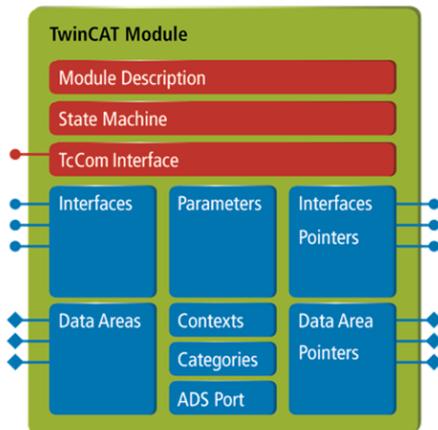
PLC 和 NC：这是与 TwinCAT 兼容的两种基本类别的 TcCOM；

Safety 和 CNC：这也是 TC2 中已经有的软件功能，在这里以 TcCOM 的形式出现；

C 和 C++ Module：TC3 新增的功能，允许用户使用 C 和 C++ 编辑 Real-time 的控制代码和接口。C++ 编程支持面向对象（继承、封装、接口）的方式，可重复利用性好，代码的生成效率高，非常适用于实时控制。广泛用于图像处理、机器人、仪器测控。

Simulink Module：TC 3 新增的功能，允许用户事先在 Matlab 中创建控制模型（模型包含了控制代码和接口），然后把模型导入到 TwinCAT 3。利用 Matlab 的模型库和各种调试工具，比 TwinCAT 编程更容易实现对复杂的控制算法的开发、仿真和优化，通过 RTW 自动生成仿真系统代码，并支持图形化编程。

基于一个 TcCOM，用户可以重复创建多个 Module。每个 Module 都有自己的代码执行区、接口数据区，此外还有数据区、指针、端口等：



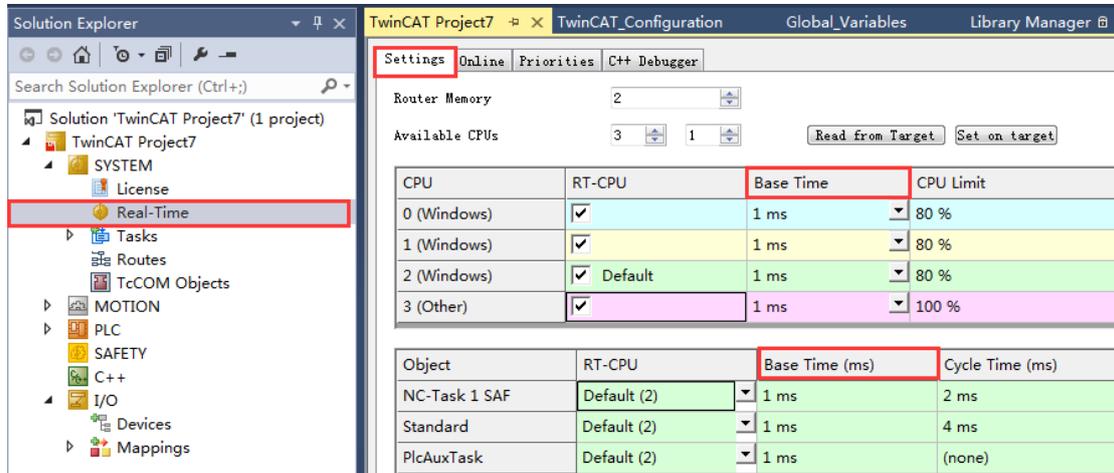
Module 可以把功能封装在 Module 里面而保留标准的接口，与调用它的对象代码隔离开来，既便于重复使用，又保证代码安全。一个 Module 可以包含简单的功能也可以包含复杂的运算和实时任务甚至一个完整的项目。TC3 运行内核上能够执行的 Module 数量几乎无限，可以装载到一个多核处理器的不同核上。

TwinCAT 3 的运行核多核 CPU，使大型系统的集中控制成为可能。与分散控制相比，所有控制由一个 CPU 完成，通讯量大大减少。在项目开发阶段，用户只要编写一个 Project，而不用编写 32 个 Project 还要考虑它们之间的通讯。在项目调试阶段，所有数据都存放在一个过程映像，更容易诊断。在设备维护阶段，控制器的备件、数据和程序的备份都更为简便。Beckhoff 公司目前的最高配置 IPC 使用 32 核 CPU，理论上可以代替 32 套 TwinCAT 2 控制器。随着半导体技术的发展，预计到 2020 年，CPU 最多可以达到 128 核，TwinCAT 3 将能胜任扩展的更多更复杂的任务。

在开发环境方面，TwinCAT 3 也做了全新的改版。最显著的改变是将 TwinCAT 3 开发环境集成在 Microsoft Visual Studio 中，成为后者的一个插件。在 TwinCAT 2 时代分别由 PLC Control、System Manager 和 Scope View 等 3 种软件实现的编程、配置、电子视波器功能，现在都可以集中在一个软件中实现。除了增加 C/C++ 和 Matlab®/Simulink® 的支持外，在 PLC 编程方面增加了面向对象的扩展功能，即 OOP 编程。

## 1.2.1. TwinCAT PLC 的实时性

TwinCAT PLC 的 CPU 实际上就是计算机的 CPU，是通过一个操作系统底层的实时核计算机的 CPU 上划分出一部分运算能力，用于执行 PLC 任务。如图所示：



在 TC3 中，针对多核 CPU，可以指定供 TwinCAT 使用的核及分别的 CPU Limit。上图中 CPU 3(Other)的 Limit 为 100%，表示这是 TwinCAT 独占的 CPU。默认所有任务都在 Default CPU 上运行，所以多核控制器上，用户要手动分配 Task 运行的 CPU。

根据默认设置，实时核首先把计算机的 CPU 时间划分成 1ms 的小片断，在每个 ms 优先执行 TwinCAT 实时任务，然后再响应操作系统的其它程序请求。如果到时间片的 80%处，TwinCAT 任务还没有执行完毕，则线程挂起，CPU 转去执行操作系统的普通任务。

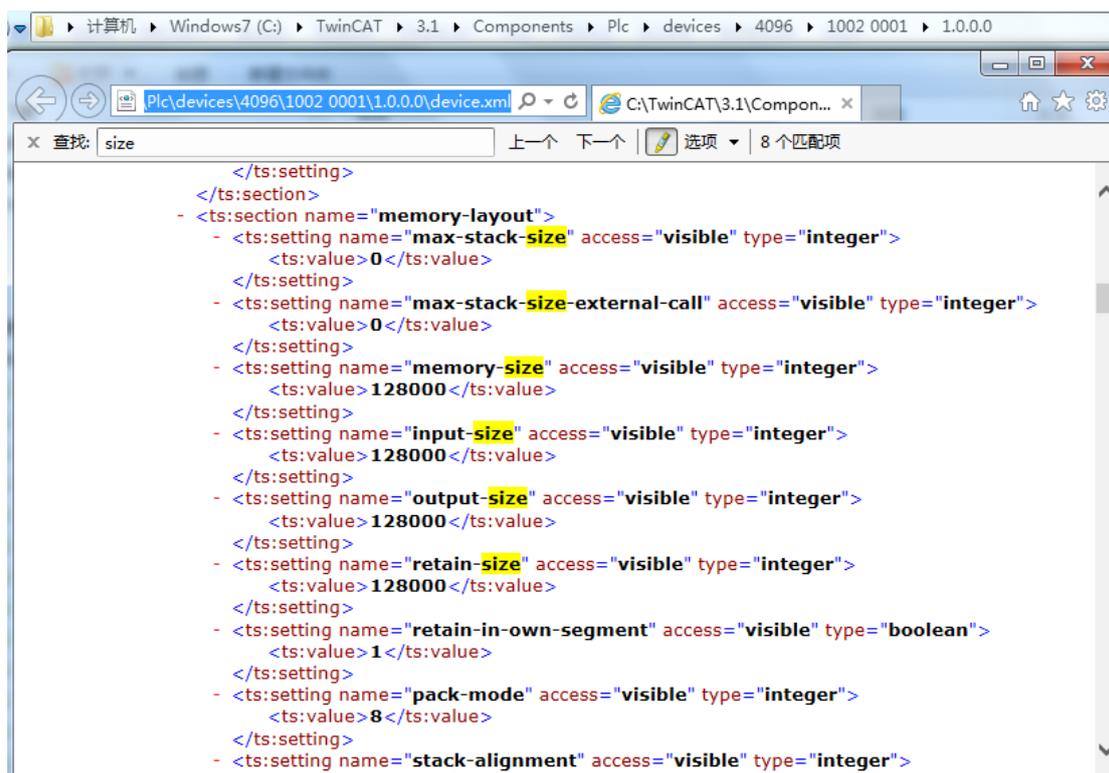
时间片 1ms 可以最小修改到 50us，而执行 TwinCAT 任务的 CPU 运算时间比例 80%也可以根据项目需要作出修改。通常情况下，TwinCAT 任务并不需要 80%的 CPU 运算能力。至于实际占用了多少，用户可以从开发工具或者 PLC 程序中访问。

## 1.2.2. TwinCAT PLC 的数据区

TwinCAT 3 中 PLC 的绝对地址区包括 Input、Output、Memory，它们都是嵌入式计算机内存的一部分，默认设置大小均为 128kByte。

如果用户可以通过以下 xml 文件修改 TC3 中的 PLC 地址区大小：

C:\TwinCAT\3.1\Components\Plc\devices\4096\1002 0001\1.0.0.0\Device.xml



Beckhoff公司的PC-Based控制器内存最小为128M,最大可以扩展到2G,所以TwinCAT PLC的内存相对于传统PLC而言,几乎是无限的。

Input区用于存放来自外部设备的输入信号,默认为128kBytes,理论上可接收6.4万路模拟量或者100万个开关量。同理Output区用于存放发送给外部设备的输出信号,默认为128kBytes,理论上可控制6.4万路模拟量或者100万个开关量。Memory用于存储中间变量。声明Input、Output、Memory区的变量时必须指定地址,它们在内存中的位置是确定的,可以按所在数据区的地址偏移量访问。

提示:

- 1, 如果没有给Input和Output区连接外部设备信号的变量分配确定的地址,程序就无法获取现场设备的状态以及控制现场设备。
- 2, TwinCATPLC提供函数Adr()来获取变量的地址,用于指针赋值。这个函数既可用于获取任意变量的地址,包括Data区的变量。
- 3, Retain数据区是掉电保持的,但是由于使用时要求苛刻,如果当前PLC程序与Retaint区保存的变量类型或者数量不一致,就会导致PLC程序启动失败。因此不推荐用户使用这个数据区。

### 1.2.3. TwinCAT PLC 的数据存储

TwinCAT PLC使用EPC或者IPC的CF卡、Cfast卡或者硬盘来存储数据。无论是程序还是数据,实际上都是存储介质上的一个文件,而目前可供货的CF卡最大容量已经达到16Gbytes,硬盘则高达320Gbytes,所以TwinCAT PLC的存储空间几乎没有限制。

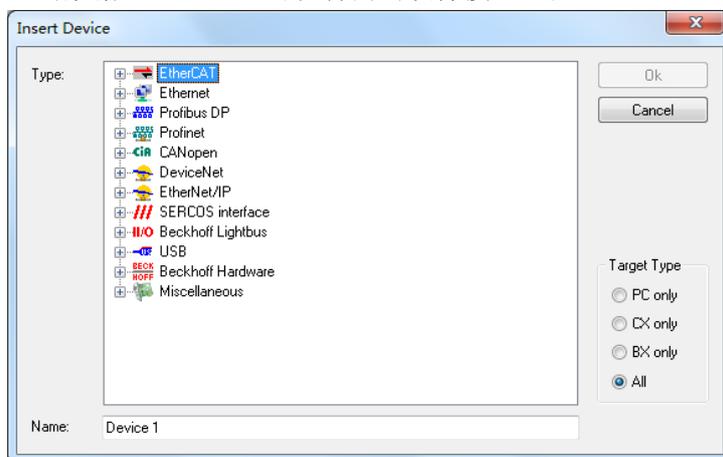
对于程序，不仅可以在 PLC 上保存机器码，而且可以下载源代码。需要的时候，工程师可以从控制器上载源代码，以确保机器上运行的程序与源代码的一致性。上载的源代码与工程师电脑上保存的文件完全相同，不仅包含基本的逻辑，还包括代码注释、调试画面、所有变量声明。

对于数据，TwinCAT PLC 没有一个固定的掉电保持区，当声明变量为掉电保持型之后，通过一定的操作，它的值就保存在存储介质上的一个文件中。此外 PLC 数据还可以通过文件读写的方式，按指定格式保存到存储介质中，然后复制到其它应用程序(比如 Excel、Notepad)中观察和分析并集中保存。TwinCAT PLC 还支持 XML 文件读写，这使得配方保存更加灵活方便。

TwinCAT PLC 的所有运行数据都在 RAM 里面，掉电即清零。需要掉电保持的变量，必须用一定的方法写入 CF 卡或者硬盘上文件，或者保存在一种特殊的硬件“NovRAM”中。具体方法参考第五章“数据存储、配方和文件”。

## 1.2.4. TwinCAT 与外设 IO 的连接

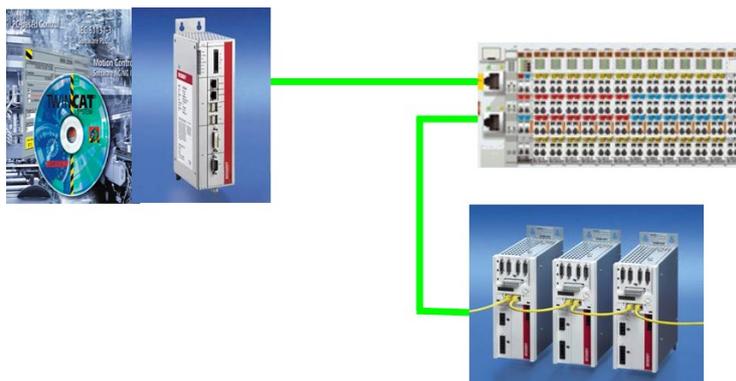
TwinCAT 与外设的物理连接，实际上就是 IPC 或者 EPC 的主板与外设的连接。根据控制器的种类不同，主板上提供的接口包括 PCI、PCIe 或者 PC104，以及所有控制器主板都具备的 Ethernet 网口。物理连接之后，TwinCAT 必须提供对物理接口的驱动程序，才能访问这些接口的数据。TwinCAT 可以访问的硬件接口包括：



其中最常用的是 EtherCAT 接口。从下面两种典型应用中，可以形象地说明 TwinCAT PLC 是如何与外设 IO 连接的。

第一种，以 IPC 带 EtherCAT 为例：

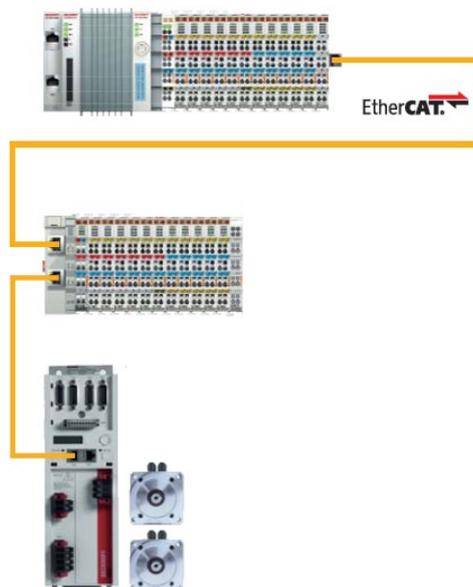
实际上，自 2004 年 Beckhoff 主导推出的工业以太网 EtherCAT 问世以来，它的高性能低成本获得了市场认可，新实施的项目通常用下图的 I/O 连接方式：



控制器直接将主板集成的网卡作为 EtherCAT 主站,通过一条网线连接 EtherCAT 从站设备。

第二种,以 CX 带 EtherCAT 为例:

对于导轨安装的 CX 系列控制器,同样也是把主板集成的网卡作为 EtherCAT 主站。典型的 IO 连接方式如下:



上图中,控制器使用一个**内置**的以太网口作为 EtherCAT 主站。控制器与电源模块拼装完成后,主板集成的 EtherCAT 主站就与第 1 个 EtherCAT 从站——电源模块 CX1100-0004——连接完成了。电源模块与相邻的 IO 端子的连接,和 IO 端子之间的连接一样,都是 EtherCAT 从站与从站的连接。

提示:

- 1, 初学者容易犯的错误是,把电源模块看成 EtherCAT 主站,或者把第一条 EtherCAT 网线的起点“EK1110”看成 EtherCAT 主站。
- 2, EtherCAT 耦合器所带的 IO 端子 EL 模块之间连接即背板总线称为 E-Bus, 这是为了区别于现场总线耦合器所带的 IO 端子 KL 模块之间的背板总线 K-Bus。实际上, E-Bus 就是 EtherCAT, 因为每个 EL 模块都是一个 EtherCAT 从站。

## 1.3. 选型设计

一个完整的控制系统包括控制器、I/O 系统和人机界面。如果设备不是单独工作，还要考虑与其它控制系统的连接，比如有没有上一层的主系统，或者下一层的子系统。

在 I/O 系统中，如果是标准电信号，可以接入相应的 IO 模块。如果是通讯方式，比如 RS485 接口的温控表、CanOpen 接口的变频器、TCP/IP 接口的机器视觉等等，那么在设备控制系统时还需要准备相应的通讯接口以及从 PLC 程序使用这些接口的软件包。

人机界面部分，虽然不参与直接的设备控制，但方案设计时必须清楚人机界面的硬件、软件、与 PLC 的通讯方式。如果 HMI 软件与 TwinCAT 要运行在同一个硬件平台上，那么在控制器选型时，就要注意 CPU、内存、硬盘是否足够，以及操作系统是否合适。

### [选型示例](#)

本节分别介绍控制器和 I/O 系统的选型。

### 1.3.1. 控制器

Beckhoff 的 PC-Based 控制器包括 EPC 和 IPC 两大类。选择控制器，首先要确定安装方式，也就是确定产品系列，然后在一个系列产品中选择适当的 CPU 和操作系统，也就确定了控制器的基本型号。具体说，选择控制器可依照以下步骤：

第 1 步，确定安装方式：

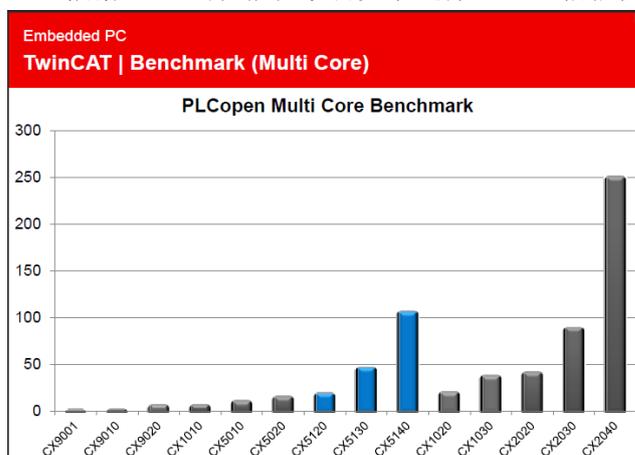
导轨安装，CX 系列

机柜安装，C69 系列

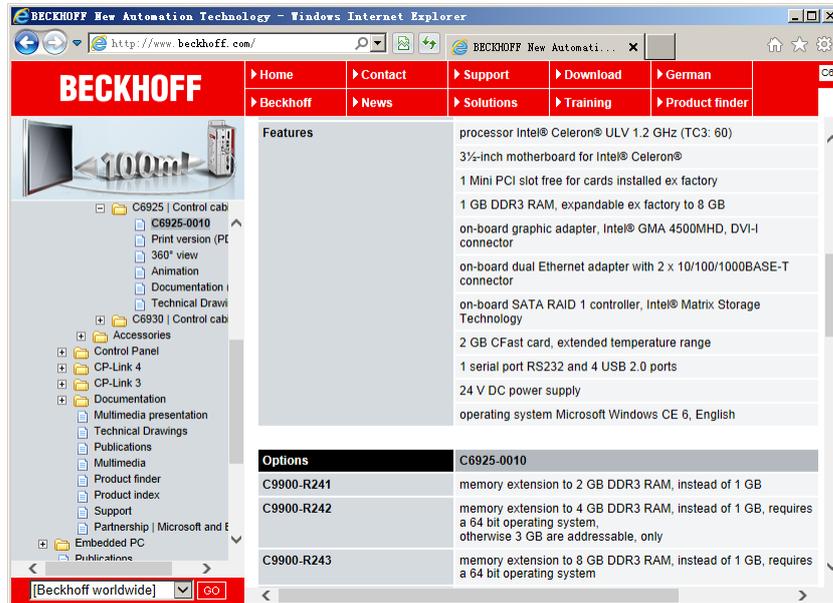
带显示面板：CP62、CP22 系列

第 2 步：选择 CPU

根据 CPU 的性能及项目要求选择 CPU，倍福控制器的相对性能比较如下：



确定了产品系列和 CPU 之后，就能在选型样本中找到正确的控制器型号了。最准确的信息是在 Beckhoff 官网上，搜索该型号，找到“Features”中的标准配置，如果标配不能满足要求，可以在“Options”项搜索需要的选项。如图所示：



### 第 3 步，确定操作系统

TwinCAT 运行核依赖于操作系统和硬件平台，不是任意控制器都可以运行任意操作系统，也不是任意控制器和操作系统的组合都可以运行 TC3。对于使用硬盘作为系统盘的工控机来说，用户可以任意选择使用 TC2 还是 TC3，但是嵌入式控制器 CX 系列，选择控制软件时还有以下限制：

Embedded PC CX   Operating System Overview							Embedded PC CX   TwinCAT3 Availability		
Device	CE5	CE6	CE7	WES 2009	WES7 32 Bit	WES7 64 Bit	Device	TC2	TC3
CX9000	✓						CX9000	✓	
CX9001	✓	✓					CX9001	✓	
CX9010	✓	✓					CX9010	✓	
CX80x0		✓					CX80x0	✓	
CX9020			✓				CX9020	✓	✓
CX1010				✓			CX1010	✓	
CX5010		✓		✓			CX5010	✓	✓
CX5020		✓		✓	✓		CX5020	✓	✓
CX51x0			✓		✓	✓	CX51x0	✓	✓
CX1020/ CX1030		✓		✓			CX1020/ CX1030	✓	
CX20x0			✓		✓	✓	CX20x0	✓	✓

CX80xx、CX90xx 系列，只能选择 WindowsCE。

操作系统安装在硬盘上的工控机和面板 PC，只能安装 Windows 7 或者 Windows XP。

CX10xx、CX50x0 系列，以及操作系统安装在 CF 卡或者 Cfast 卡上的工控机或者面板 PC，就需要选择使用 Windows XPe 或者 Windows CE。

下面列表说明两种操作系统的优点和缺点。

项目	Windows XPe	Windows CE
启动速度	慢	快
Tc 实时核效率	高	低
CF 卡的空间要求	>=2G	<64M
价格	高	低
程序开发和维护	既可本地编程，也可以另用 PC 机远程编程	只能用 PC 机远程编程
HMI 的开发和运行	与 IPC 相同	必须使用 CE 版的开发平台

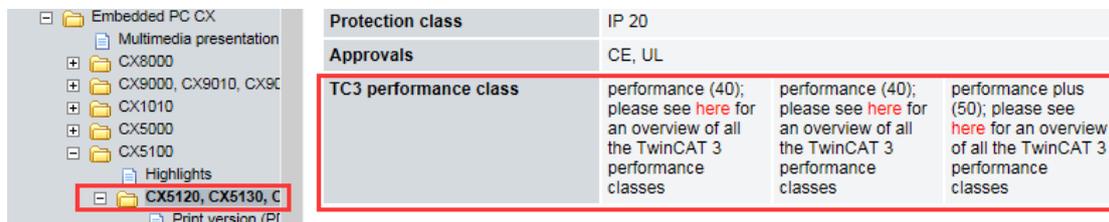
特别注意:选择 Windows Xpe, 必须订购 CF 卡或者 Cfast 卡选件, 容量不低于 2GB。而 CX51xx 系列控制器, 则完全不包含存储卡, 用户必须增加 Cfast 卡选项。

#### 第 4 步, 确定 TwinCAT 3 的基本软件功能

订货号	TwinCAT Level	功能
TC1200-00x0	TwinCAT PLC	软 PLC
TC1250-00x0	TwinCAT PLC 及 TwinCAT NC PTP(10 轴以内)	软 PLC 点对点的运动控制
TC1260-00x0	TwinCAT PLC 及 TwinCAT NC PTP(10 轴以内) TwinCAT NC I	软 PLC 点对点的运动控制 3 轴插补功能。
TC1270-00x0	TwinCAT PLC 及 TwinCAT NC PTP(10 轴以内) TwinCAT NC I TwinCAT CNC	软 PLC 点对点的运动控制 3 轴插补功能 5 轴以内插补

注意:

- 即使是对于 CX 控制器, TC3 的 Runtime 也必须单独订购。
- 软件订货号不再区分操作系统 CE 或者 Win7。
- 软件订货号根据控制器性能级别不同而不同, “-00x0” 中的 x 就表示控制器的 Performance 级别, 性能越强的 CPU 级别越高, 价格越贵。单独订购 TC3 软件, 该值为 “-0090”, 即按最高性能级别计算。在 Beckhoff 官网及选型手册上, 每款控制器都有 TC3 Performance Class 标注。如图所示:



- 如果是包含 NC 控制, 还必须确定轴的数量范围。

NC 轴数	软件订货号	说明
1-10 轴	TC1250	等效于 TC1200+TF5000
11-25 轴	TC1200+TF5010	

26-255 轴	TC1200+TF5020	
----------	---------------	--

#### 第 5 步，确定 TwinCAT 3 的扩展软件功能

TC2 中的 supplement，在 TC3 称为 Function，必须为每个控制器购买需要的 Function 授权，即使只是 lib 也不例外，比如温控库、扩展的 Motion 库、各种通讯库等等。虽然这些库的收费也很便宜，多数为几百元人民币，但必须订购才能使用，其价格与控制器的性能等级相关。

以下在 TC2 平台时国内用户经常免费使用的 Supplement，注意在 TC3 平台下需要订购：

订货号	软件名称	功能简述
TF4100	TC3 Controller Toolbox	PID , Filter, Ramp Generator 等
TF4110	TC3 Temperature Controller	温控库
TF5050	TC3 MC Camming	凸轮
TF5055	TC3 MC Flying Saw	飞锯
TF5060	TC3 NC Fifo Axes	FIFO 轴组
TF5065	TC3 Motion Control XFC	常用于 EL1252、EL2252 等 XFC 模块完成的 Touch Probe 及 Cam Switch 功能。
TF6200/6210	TC3 OPC UA/OPC DA	常用于与 HMI 应用程序通讯
TF6250/6255	TC3 Modbus TCP/Modbus RTU	常用于与触摸屏或者仪表经 Modbus 通讯
TF6310	TC3 TCP IP	与视觉系统等第三方设备的以太网通讯
TF6340	TC3 Serial Communication	RS232 或者 RS485 串口通讯库，处理 EL60x1 或者经 PC 的 Com 口与第三方设备通讯。
TF6421	TC3 XML Server	常用于配方等掉电数据保存。
???	TC3 HMI	TwinCAT 自带的组态软件，可以独立于 PLC 运行。目前仅支持 Win7 或者 XP

#### 第 6 步，确定扩展选件

通常控制器的存储介质和内存容量是可以扩展的，有的系列甚至可以配成双硬盘、双 CFast 卡或者硬盘+CFast 卡。当然如果标配已经满足要求，就不必扩展了。

**CF 卡选件：**对于，WinCE 操作系统，CF 卡扩展选项不是必须的。对于 WindowsXPe 操作系统，CF 卡至少要 2G，而 Win7 系统，Cfast 卡至少要 16G。如果标配是 64M 的 CF 或者 Cfast 卡，就必须订购存储容量扩展选件了。

**内存选件：**内存扩展选项不是必须的，对于 WindowsXPe 操作系统，由于 OS 本身占用内存大，如果 HMI 复杂的话，建议扩展内存到 512M 或者更大。

**CPU 选件：**对于 IPC，如果标配的 CPU 性能不够，还可以升级。

#### 第 7 步，电源、UPS 和电池

如果操作系统选择了 Win 或者 Xpe，由于 PLC 允许随时断电，而 PC 随时断电可能导致文件损坏，所以通常会配上 UPS 和电池。

对于工控机和面板式 PC, 电源是标配自带的, 如果需要 UPS 必须同时订购 C9900-U330 (UPS) 和 C9900-U209 (电池)。

在 CX 系列支持 TC3 的控制器中, 只有 CX20x0 需要配置电源模块。如图所示:

Technical data	CX2100-0004	CX2100-0014	CX2100-0904	CX2100-0914
Power supply	24 V DC (-15 %/+20 %)			
Max. output	45 W	90 W	45 W	90 W
I/O connection	E-bus (EtherCAT Terminals) or K-bus (Bus Terminals), automatic recognition			
Current supply E-bus/K-bus	2 A			
UPS	-	-	capacitively integrated	external Smart Battery
Charge	-	-	75 As	dependent on battery

CX2100-0xy4 是电源型号,

x 为 0 表示不带 UPS, x 为 9 表示不带 UPS,

y 为 0 表示功率 45w, 用于 CX2020 和 CX2030, 为 1 表示 90w, 用于 CX2040.

CX2100-0914 是用于 CX2040 带 UPS 的电源模块, 这个模块还必须购买电池 CX2900-0192。

## 1.3.2. 系统扩展模块

系统扩展模块, 包括串行通讯模块、现场总线模块等。对于使用 EtherCAT 的系统, 这两种模块都有相应的 EL 模块来代替。由控制器主板上扩展出的串口模块速度快、价格便宜, 缺点是串口故障时需要整个控制器返修。而通过主板扩展的现场总线模块, 已经完全被 EL 模块代替。

- 串行通讯模块

CX 系列嵌入式 PC 最多可以扩展 2 个串口模块, 即最多 4 个串口。其订货号如下:

CXxxxx-N030	RS232: Com1+Com2
CXxxxx-N041	RS485: Com3+Com4
CXxxxx-N031	RS485: Com1+Com2
CXxxxx-N040	RS232: Com3+Com4

注意: 如果只扩一个模块就选 Com1 和 Com2。如果要扩两个模块就“Com1+Com2”和“Com3+Com4”各选一个。扩展的 Com 口与普通 PC 机的 Com 口完全兼容, 在 TwinCAT PLC 中的用法也完全相同。

实际应用中, 不建议用户扩展 COM 口, 而是使用串行通讯模块 EL60xx 代替。好处是备件方便。

- 现场总线模块

EtherCAT 接口的现场总线接口模块和普通的 E-Bus 端子模块一样, 可以位于 EtherCAT 网络的任何位置, 数量也不受 CPU 限制。

订货信息	主站	从站
LightBus	EL6720	
ProfibBus	EL6731	EL6731-0010
CanOpen	EL6751	EL6751-0010
DeviceNet	EL6752	EL6752-0010

### 1.3.3. I/O 系统

在 I/O 选型之前，必须确定控制器与 IO 连接的总线种类。在新实施的项目中，通常使用 EtherCAT。

I/O 系统的选型包括 5 项内容：

- 1, 现场总线主站模块和从站耦合器
- 2, 信号模块
- 3, 系统模块
- 4, 电缆和接头
- 5, 总线分支选件（可选）

下面分别说明

- 1, 现场总线主站模块和从站耦合器

对于 EtherCAT 系统，主站不用选择。因为控制器上已经集成了 EtherCAT 主站网卡。RJ45 接口的从站耦合器标准型号为 EK1100，或者 EK1101（带 DIP 开关）。特别需要可参考厚样本。

- 2, 信号模块

DI: 信号类型是 PNP 还是 NPN?

DO: 有输出时是高电平还是低电平? 输出电流 0.5A 还是 2.0A?

AI: 信号类型是什么? 12 位分辨率是否足够? 可选 16 位和 24 位模块，价格较贵。如果是温度信号，是热电阻还是热电偶，测温范围是多少?

AO: 信号类型是什么? 12 位分辨率是否足够? 可选 16 位和 24 位模块，价格较贵。

编码器模块: SSI 还是增量编码器? 单端还是差分信号? 是否需要 C 相或者 Latch 点? 如果不需要 C 相或者 Latch 点可以选择相对经济型的双通道模块。

PWM 输出模块: 用于驱动什么负载，输出电流多少，切换频率如何? 用普通 DO 模块代替是否可以?

高速计数模块: 信号类型是 NPN 还是 PNP? 最高频率如何? 用普通 DI 模块代替是否可以?

智能仪表模块: 是否需要电力测量、称重、气压或者其他毫伏信号。倍福提供 IO 模块式的智能仪表。测量结果直接传送至 PLC。

通讯网关模块: 是否要接入其它现场总线（Profibus 或者 CanOpen 等等），通讯的对方是从站还是主站? 是否需要扩展 RS232 或者 RS485 接口?

驱动模块: 200W 以内的电机可以由 IO 模块直接驱动。

XFC 模块：XFC 技术可以实现数字量 1MHz、模拟量 100Khz 的采样频率，100KHz 内的信号发生器、计数器都可以轻松实现。而时间戳功能则使探针和凸轮输出独立于伺服驱动器和位置反馈装置

### 3, 系统模块

终端模块：对于 KL 模块组而言，最重要的系统模块是终端模块 KL9010。对于 EL 模块组来说，终端模块是 EL9011。

负载电源模块：总线耦合器或者电源模块，最多能提供 24V、10A 的负载容量。对于 DO 通道较多的 IO 站，要仔细核算负载电源的容量，如果超过 10A，就要增加负载电源模块 EL/KL9100。

控制电源模块：总线耦合器或者电源模块，最多能提供 5V、2A 的控制电源容量。选型手册(厚样本)上每个 KL 或者 EL 模块都标注了消耗的 E-bus 电流或者 K-bus 电流。把每个 IO 站上电流消耗相加，如果超出 2A 就必须增加控制电源模块 EL/KL9410。

其它电源模块：以上 3 种系统模块是必须的，不是必须但可能用到的电源模块请参考厚样本。

### 4, 电缆和接头

对于 EtherCAT，有预装好的定长电缆，型号为：ZK1090-9191-xxxx。比如

ZK1090-9191-0005: 0.5 米

ZK1090-9191-0010: 1 米

ZK1090-9191-0020: 2 米



如果不能确定每条网线的长度，又不想预留太多，用户可以订购总长若干米的长网线和 RJ45 接头。

EtherCAT 接头	ZS1090-0003	4 针, IP 20, 每盒 10 个
EtherCAT 电缆	ZB9010	固定安装, CAT 5e



如果是其它总线，或者**光纤**形式的 EtherCAT，请参考厚样本或者官网内容选型。

[http://www.beckhoff.com/english/ethercat/accessories\\_ethercat.htm?id=443317296563](http://www.beckhoff.com/english/ethercat/accessories_ethercat.htm?id=443317296563)

### 5, 总线分支选件（可选）

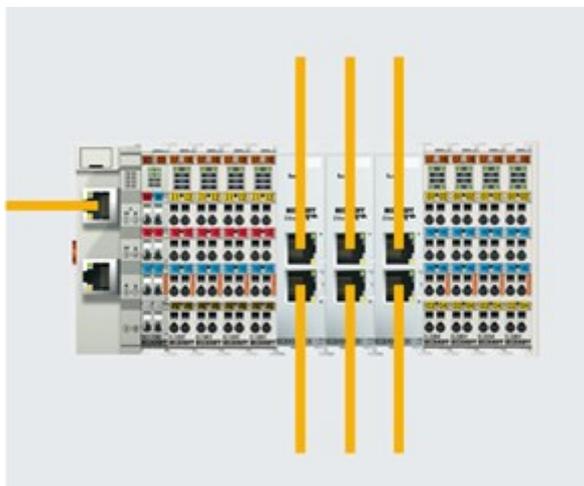
EtherCAT 最简单的拓普方式是链式。除 IO 模块之外，所有 EtherCAT 从站都有 In 和 Out

两个 EtherCAT 接口，只要各个从站的 In 和 Out 接口串连即可，不需要其它硬件。

假如用户需要树形或者星形连接，就需要专门的 EtherCAT 分支选件。其中 CU1128 是独立安装独立供电的 8 口 EtherCAT 交换机，可以方便地实现各 IO 站的星形连接。



EK1122 是作为模块安装在 IO 站中，实现 EtherCAT 主干道上两路分支。如图所示：



选型：见 [《Main Catalog》](#)（又称厚样本），或者 [《Product Overview》](#)（又称薄样本）。

## 1.4. 安装和接线

详细的安装要求，参考附件

[《简明安装手册》](#)

对于硬件的安装尺寸、安装要求、接线说明，可以查阅硬件的用户手册和图纸：

用户手册： I/O 端子，厂家并不随货提供纸质说明书，用户可以到 [Http://www.Beckhoff.com](http://www.Beckhoff.com) 或者 [Http://www.Beckhoff.com.cn](http://www.Beckhoff.com.cn) 下载，或者安装 Beckhoff 套装 DVD 中的 TwinCAT PLC 以及帮助文件，即可从 C:\TwinCAT\InfoSystem\1033 目录下找到所有硬件的技术文档。

图纸：用户可以到 [Http://www.Beckhoff.com](http://www.Beckhoff.com) 或者 [Http://www.Beckhoff.com.cn](http://www.Beckhoff.com.cn) 下载，或者在“Beckhoff 套装 DVD/Technical\_Drawings/”路径下找到相应类别/型号的 AutoCAD 图纸。

需要强调的是电源模块的接线:

为安全起见, 凡是供入电源的地方, 包括 CX 电源模块、耦合器 EK110x、EL9410, 其控制电源 ( $U_s$ ) 和负载电源 ( $U_p$ ) 应独立供电, 并且两组电源的 24V 和 0V 进模块之前都应该分别加上保险丝。 $U_s$  的保险丝熔断电流为 CPU 和模块 E-bus/K-bus 额定功耗折算成 24V 供电时电流的 1.2 倍。而  $U_p$  的保险丝熔断电流则为计算负载总电流的 2 倍左右。

## 2. 编程入门

### 2.1. 概述

TwinCAT 3 软件分为开发版和运行版。在“[第 1 章 系统概述](#)”中，简单介绍了 TwinCAT 的原理和若干特点，都是指的 TwinCAT 运行版 (XAR)，又称为 TwinCAT Runtime，它是控制系统的核心。运行版是用户订购，并在出厂前就预装好的。

本章要介绍的是 TwinCAT 3 开发版 (XAE) 的使用，包括安装过程、配置编程环境以及一些常用的基本操作步骤。TwinCAT 3 开发版是免授权的，可以从 Beckhoff 任意分支机构获取 TwinCAT 套装 DVD，也可以从 [Beckhoff 官网下载](#)，然后安装在工程师的编程 PC 上。

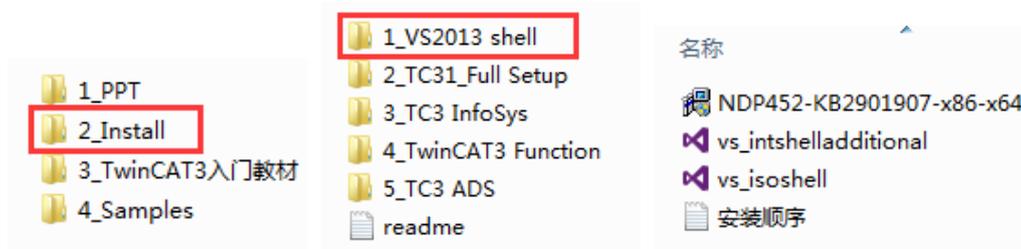
### 2.2. 在编程 PC 上安装 TwinCAT 开发环境

说明：安装 TwinCAT 之前，建议关闭杀毒软件，尤其是“360”。

#### 2.2.1. 在 PC 上安装 TwinCAT 开发环境

“TwinCAT 3 培训资料”可以从百度云下载：<http://pan.baidu.com/s/1gd1zbnN>

安装文件位于以下路径：



因为 XP 不支持 VS2012 及以上，单独安装 TC3 会自动安装 VS2010 Shell，所以 Win 7 系统推荐按顺序安装，Win XP 可跳过第 1) 步。

- 1) 先装 VS2012 Shell 或者 VS2013 Shell  
依次安装

NDP452-KB2901907-x86-x64-AllOS-ENU.exe  
vs\_isoshell.exe  
vs\_intshelladditional.exe

- 2) 安装 TC31-Full-Setup.3.1.4018.4 (XAE)  
选 complete 安装，

安装包会自动识别出 PC 中 VS 的版本，勾选 Microsoft Visual Studio 2013 Shell 即可。

### 3) 安装 TC3-InfoSys

//以下为可选安装项目，初次使用不安装。

### 4) 根据项目需要，安装 Function

强烈推荐客户直接去 Beckhoff 官网下载最新的 Function。因为 TC3 的 Function 更新很快，随时与官网公布版本保持一致，没有坏处。

以下是倍福广办申请的公共下载帐户：

下载地址：<http://www.beckhoff.com/>

帐号：346064941

密码：346064941

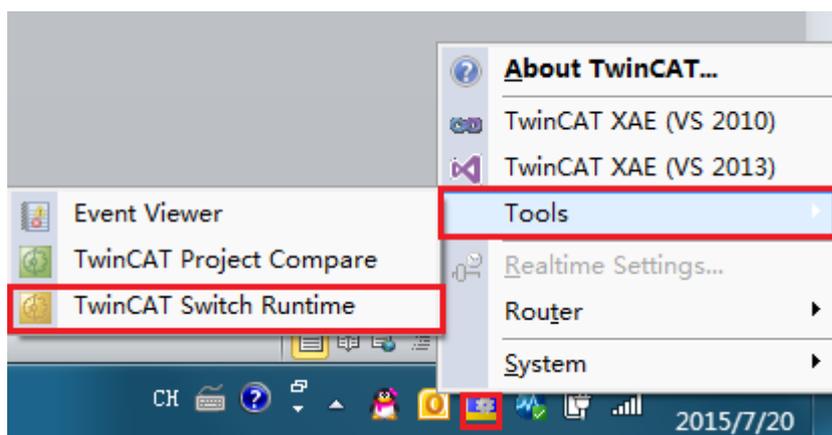
5) 根据项目需要，在仅运行组态软件或者高级语言开发的 HMI 程序的 PC 上安装 TC3 ADS。

## 2.2.2. 升级 TwinCAT 开发环境

升级 TwinCAT 不需要卸载当前版本，只要找到新版本的安装文件，直接完装即可，然后重启操作系统。升级后的 TwinCAT 沿用之前的设置参数和授权，扩展功能包 Supplement 也不需要重装安装。

## 2.2.3. 在 TC3 和 TC2 之间切换

很多用户同时使用 TC2 和 TC3 的开发版，但两个服务不能同时运行。TC3 中提供了切换小工具，如图所示：



如果要从 TC2 切换到 TC3，需要直接运行或者建立快捷方式：

C:\TwinCAT\TcSwitchRuntime\TcSwitchRuntime.exe

## 2.3. 初步认识开发环境

### 2.3.1. TC 3 图标和 TC 3 Runtime 的状态

TwinCAT 安装成功并重启后，编程 PC 桌面右下角有会出现 TwinCAT 图标。



[小技巧：在桌面菜单条显示 TwinCAT 图标的方法](#)

此处图标的颜色代表了编程 PC 上的 TwinCAT 工作模式：



蓝色图标表示配置模式，



绿色图标表示运行模式。



红色图标表示停止模式。

任何运行了 TwinCAT Runtime 的 PC-Based 控制器上都有这三种模式。如果用传统的硬件 PLC 来比喻 TwinCAT Runtime 的三种模式，可以表述为：

配置模式——PLC 存在，但没有上电。所以不能运行 PLC 程序，可以装配 IO 模块。

运行模式——PLC 存在，已经上电，可以运行 PLC 程序，但不能再装配 IO 模块。

停止模式——PLC 不存在。

**注意：**

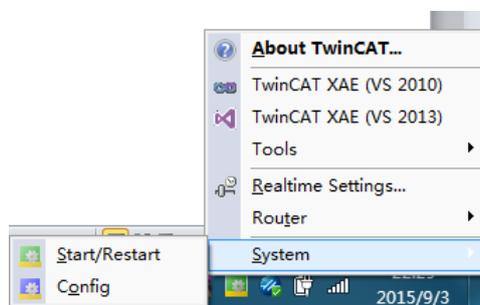
如果编程 PC 上的 TwinCAT 处于停止模式，就不能对其它 PLC 编程。

如果控制器上的 TwinCAT 处于停止模式，就不接受任何 PC 的编程配置。

### 2.3.2. TC 3 快捷菜单的功能

- 编程 PC 的 TwinCAT 状态切换

点击 TwinCAT 图标，在弹出的菜单中选择 System，就显示出左边的子菜单：



然后点击“Start/Restart”，编程 PC 就进入仿真运行模式。点击“Config”，就进入配置模式。状态切换失败，或者服务启动失败，才会进入停止模式。

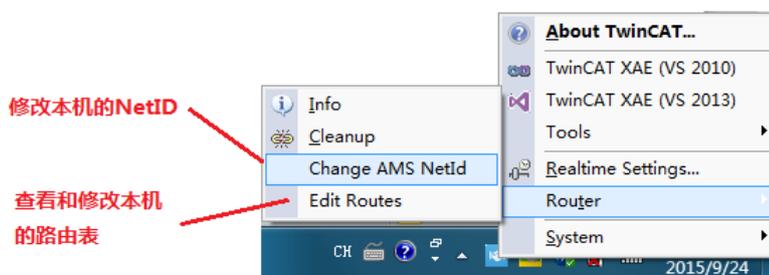
- 进入 TwinCAT 开发环境的快捷方式

如果开发 PC 上安装有多个 Visual Studio 版本，点击右下角的 TC3 图标，就可以选择进入哪个版本的 Visual Studio 中的 TwinCAT 3。



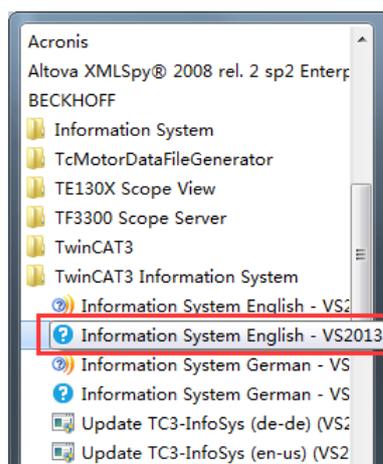
对于 Windows 8 系统，会提示 0x4115 错误，提示到 TwinCAT\3.1\下找 Win8.....bat 文件，运行并重启电脑。（文字有记录沈强补充，暂时找不到）

- 本机的 ADS 路由信息查看和编辑

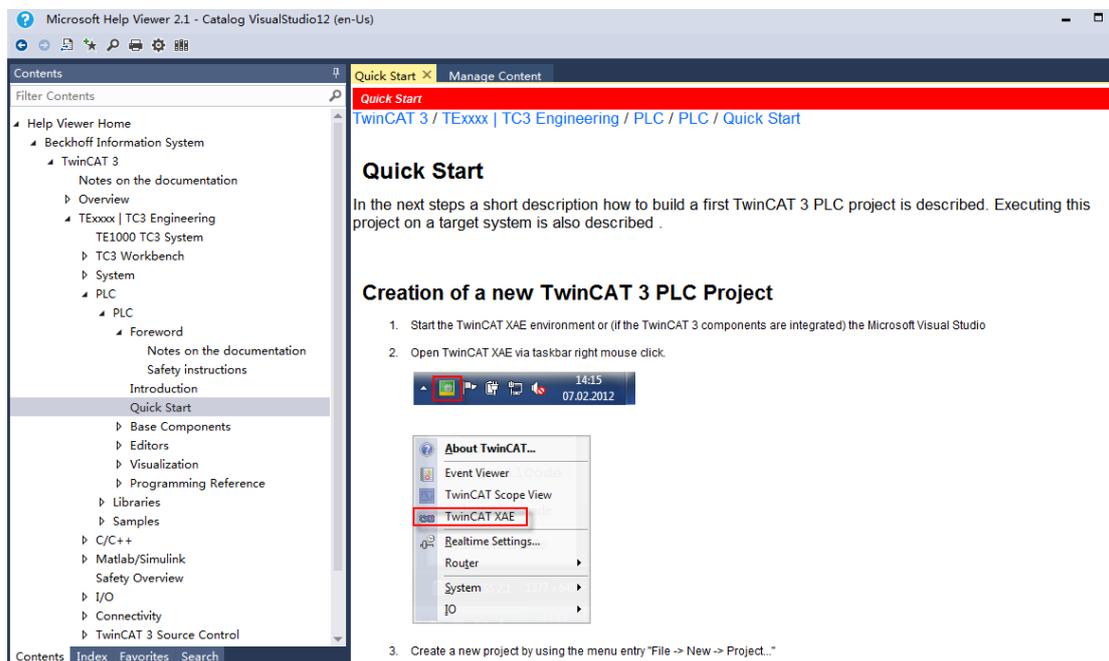


### 2.3.3. 启动 TC3 的帮助系统

在 VS2013 Shell 的开发环境下，按 F1。  
或者从以下界面进入：



## 2.3.4. TC3 Quick Start 教程



## 2.3.5. 启动示例程序

Beckhoff 公司提供的完整的 TC3 示例程序是:

“\TC3\_培训资料\4\_Samples\ TC3\_SortingSystem.zip”,

该示例的使用方法如下:

- 激活配置 Activate the TwinCAT configuration
- PLCs 程序 Login 并启动运行: TC3\_SortingSystem\_PLC 和 TC3\_SortingSystem\_Simu
- 通过 TC3\_SortingSystem\_PLC 的画面操作项目
- 例如: 1. 主电源开关, 2. 自动模式, 3. 启动

## 2.3.6. 获取和注册正版授权

本节内容节摘自“\TwinCAT 3 入门教材\11 TC3 第一章 安装和激活”，作者：杨煜敏。

### 试用版激活方式

在激活配置运行程序时，系统会提示需要的试用授权，用户只需要根据提示输入字符即可。试用授权可以用 7 天。之后会再次收到提示，重复以上动作可以再用 7 天，以此类推。所以用户无须购买开发版授权就可以完成项目开发。

### 完整版激活方式

(1) 首先选择 manager licenses, 在 Add License 中勾选所需要的 License,

Order No	License	Add License
TC1000	TC3 ADS	<input checked="" type="checkbox"/> cpu license
TC1100	TC3 IO	<input type="checkbox"/> cpu license
TC1200	TC3 PLC	<input type="checkbox"/> cpu license
TC1210	TC3 PLC / C++	<input type="checkbox"/> cpu license
TC1220	TC3 PLC / C++ / MatSim	<input type="checkbox"/> cpu license
TC1250	TC3 PLC / NC PTP 10	<input type="checkbox"/> cpu license
TC1260	TC3 PLC / NC PTP 10 / NC I	<input type="checkbox"/> cpu license
TC1270	TC3 PLC / NC PTP 10 / NC I / CNC	<input type="checkbox"/> cpu license
TC1300	TC3 C++	<input type="checkbox"/> cpu license
TC1320	TC3 C++ / MatSim	<input type="checkbox"/> cpu license
TE1300	TC3 Scope View Professional	<input type="checkbox"/> cpu license
TE1400	TC3 Target For Matlab Simulink	<input type="checkbox"/> cpu license
TE1410	TC3 Interface For Matlab Simulink	<input type="checkbox"/> cpu license
TE1500	TC3 Valve-Diagram-Editor	<input type="checkbox"/> cpu license

Ignore Project Licenses

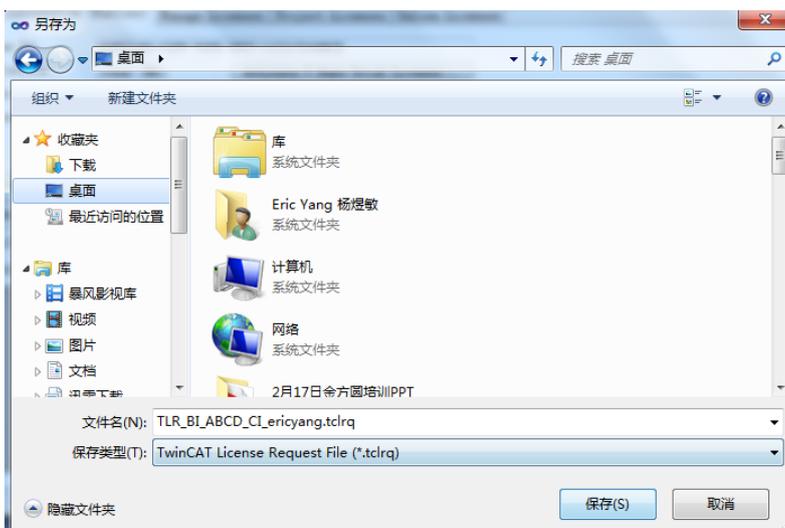
(2) 随后回到 Order Information:

Beckhoff License 中输入购买 License 时候的订单号, 例如 ABCD,

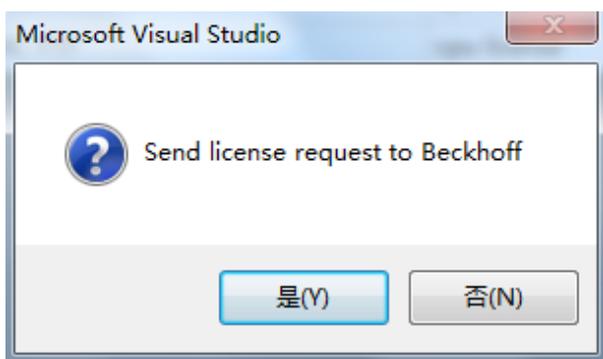
Custom Id 中可以任意输入一些数字或者字母, 例如下图中输入我的名字,

Order No	License	Instances
TC1200	TC3 PLC	cpu license
TF5000	TC3 NC PTP	cpu license

(3) 随后点击 Generate License Request File, 系统会生成一个 license 的请求文件, 保存到任意位置。



(4) 随后弹出对话框询问是否要把此文件发送到 bechhoff 德国总部。

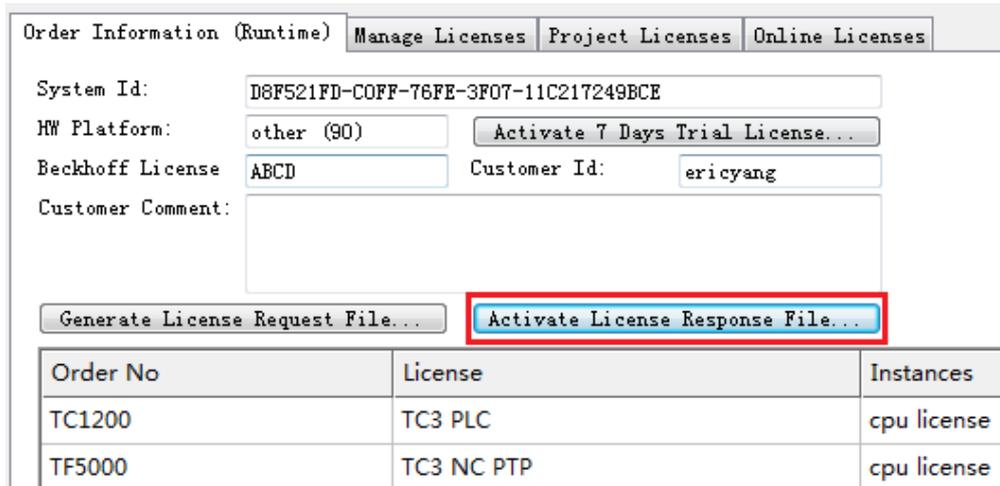


点击“是(Y)”。

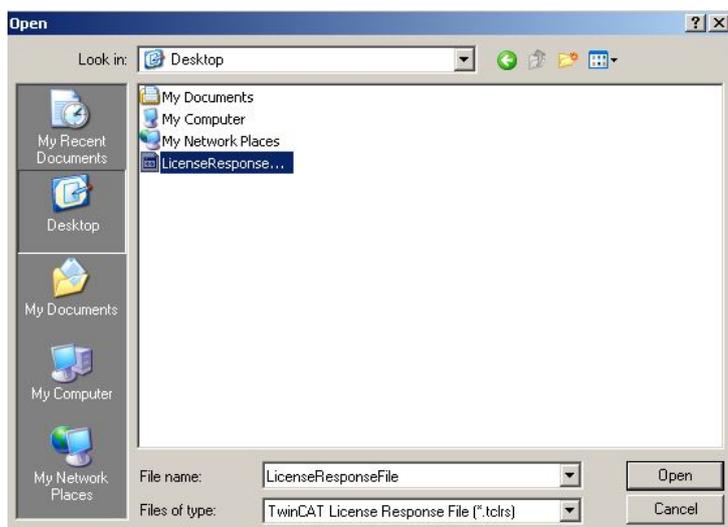
(5) 随后系统就会以邮件方式帮您发送到一个固定邮箱 `tclicense@beckhoff.com`，邮件内容包含了你所申请的 license 和申请文件，申请文件中包括了您的订单号信息，您的用户名信息，您所要激活的这台电脑的 HW 等。



(6) 如果德国收到邮件后核实相关信息没有错误后，就会返回一个邮件给你，并返回一个激活文件给你，随后点击 Activate License Response File 把激活文件导入就注册完整版成功。



(7) 导入返回的激活文件，



(8) 提示以下对话框就说明注册成功，点击 OK 即可。



(9) 最后注意下这个 license 文件会自动复制到 C:\TwinCAT\3.x\Target\License 文件下，请备份以免激活文件丢失。

## 2.4. 编程准备：添加路由（Add ADS Router）

### 2.4.1. 设置 IP 地址

编程 PC 总是通过以太网对 PC-Based 控制器进行编程和配置，和其它 PC 之间的通讯一样，通讯双方必须处于同一个网段。为此，必须先确定控制器的 IP 地址，才可能把编程 PC 和控制器的 IP 地址设置为相同网段。

控制 Beckhoff 控制器的 IP 地址有以下方法：

方法一：适用于新购控制器或者重刷过操作系统的控制器。

控制器出厂时，IP 分配方式为 DHCP，即由外接路由器分配地址。如果网内没有路由器，则默认 IP 地址为：169.254.X.X。如果把 PC 机的 IP 地址也设置成 169.254.X.X，掩码为 255.255.0.0。

方法二：适用于已经使用过的控制器，没有显示器，但不确认 IP 地址，WinCE 操作系统。

掉电，拔出 CF 卡，用读卡器删除文件夹 Document and Setting，删除\TwinCAT\Boot\下所有文件。注意删除之前应做好备份。

然后插回 CF 卡，重新上电，按默认设置的情况处理。

方法三：适用于带 DVI 接口并且连接显示器的控制器。

从显示器进入 Control Panel，找到 Network setting 项，修改 IP 设置。

方法四：适用于所有情况，

用第三方工具软件，比如 Wireshark。网线连接 PC 和控制器后，将控制器掉电，开启 PC 网卡的 Frame Capture，然后再控制器上电。观察数据包，可以见到除了 PC 的 IP 之外，另有一个 IP 会发送数据包，那个就是控制器的 IP。

确定控制器的 IP 地址之后，用适当的方法修改编程 PC 或者 TwinCAT 控制器的 IP 地址，使二者处在同一个网段。并在开发 PC 上启用命令模式，运行“Cmd”，然后用 Ping 指令验证局域网是否连通。

关闭杀毒软件的防火墙，以及操作系统的网络连接防火墙，或设置 TwinCAT 为例外。

## 2.4.2. 设置 NetID

### ● 什么是 NetID?

理论上，编程 PC 可以对所在局域网内的任意 TwinCAT 控制器进行编程调试。假定局域网内除了普通 PC 之外，多台装有 TwinCAT 运行版的控制器，以及安装了 TwinCAT 开发版的编程 PC。这些 PC 之间何区分呢？简单地说，所有 PC 之间以 IP 地址区分，而 TwinCAT 控制器及开发 PC 之间以 AMSNetID 区分。

AMSNetID 简称 NetID，NetID 是 TwinCAT 控制器最重要的一个属性，编程 PC 根据 TwinCAT 的 NetID 来识别不同的控制器。

说明：

*NetID 是 Beckhoff 公司定义的 TwinCAT 设备之间通讯的 ADS 协议规范中的一个名词。完整的理解 NetID 必须了解 ADS 通讯协议。有兴趣的用户可以参考附件“[ADS 通讯协议](#)”。*

NetID 是一个 6 段的数字代码。TwinCAT 控制器 NetID 的最后两段总是“1”，而前 4 段可以自定义。从 Beckhoff 订购的控制器出厂有一个默认的 NetID，用户可以修改，也可以维持。而编程 PC 安装了 TwinCAT 之后也有一个默认的 NetID。**必须确保同一个局域网内的 NetID 没有重复。**

说明：习惯上把编程 PC 的 NetID 设置为 IP 地址加上后缀“.1.1”。比如 IP 地址是“192.168.1.119”，NetID 就设置为“192.168.1.119.1.1”。Net ID 修改后，系统会要求重启电脑，选择“YES”。这是由于 Net ID 是存储在操作系统的注册表中，每次开机 TwinCAT 服务启动时，就已经确定了 Net ID 与 IP 地址的绑定关系。二者的前 4 段可以相同，也可以不同，但必须是唯一对应关系。

## 2.4.3. 在 TC3 的 System | Routes 中添加路由

### ● 什么是路由?

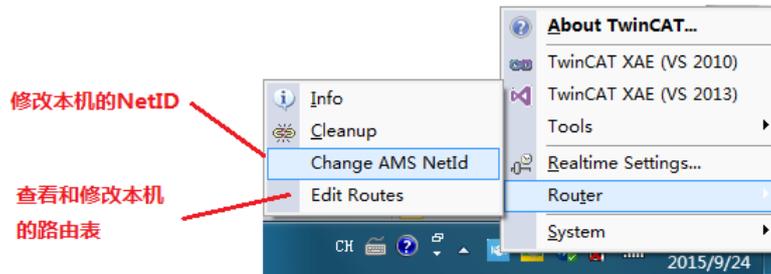
“路由”，即“AMS Router”，是 Beckhoff 公司定义的 TwinCAT 设备之间通讯的 ADS 协议

规范中的一个名词。每个 TwinCAT 控制器都有一个路由表，在路由表中登记了可以与之通讯的 TwinCAT 系统的信息，包括 IP 地址（或者 Host Name）、NetID、连接方式等。

完整的理解 AMS Router 必须了解 ADS 通讯协议。有兴趣的用户可以参考附件“[ADS 通讯协议](#)”。

- 如何查看路由表？

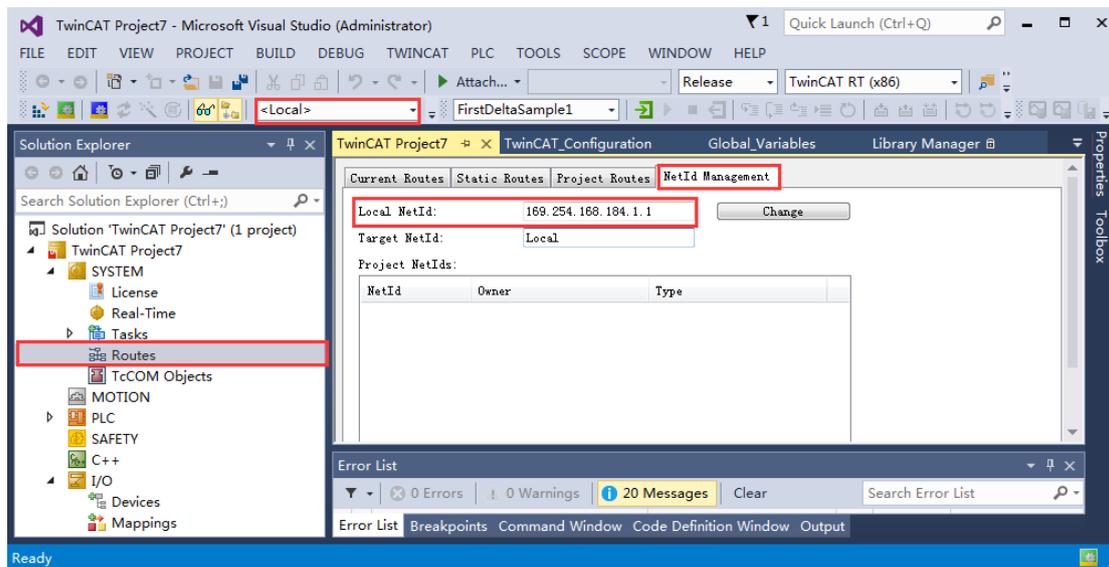
本机的路由表，可以从 TC3 图标右键快捷菜单的 Router 访问。



从这里修改路由表里，不需要输入用户名和密码。

下面的方法则本机和目标控制器都适用：

新建一个空白 TwinCAT 项目，选中目标系统为 Local，就可以从以下界面看到本机的 NetID 及路由表：



点击 Current Routes 页面：

Route	AmsNetId	Address	Type	Comment
GUANGZHOUTEC...	169.254.54...	192.168.1.127	TCP_IP	
CX-1DC32C	5.29.195.44...	169.254.188...	TCP_IP	
CX-1DBE2E	5.29.190.46...	169.254.211...	TCP_IP	

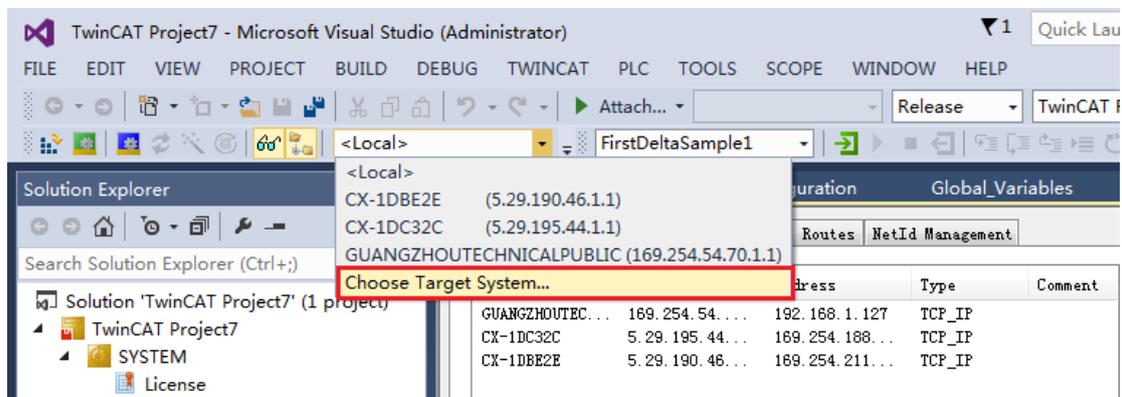
上图中可以看到编程 PC 的当前路由表，一共有 3 行，每行表示一台 TwinCAT 控制器。第 1 列是控制器的名字，第 2 列是 NetID，第 3 列是 IP 地址，第 4 列连接类型。编程 PC 只能对自己路由表中的控制器进行编程。

从这个页面，不仅可以查看路由表，也可以删除（Remove）或者添加（Add）路由表项。

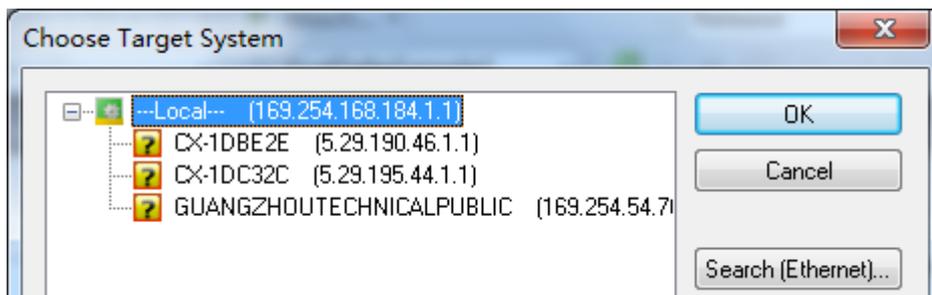
实际上，每个 TwinCAT 控制器都有一个路由表，每个控制器只接受自己路由表中的 PC 编程。控制器的路由表要“2.4.3 添加路由表”完成以后才能从 System Manager 页面看到。

- 如何添加路由表？

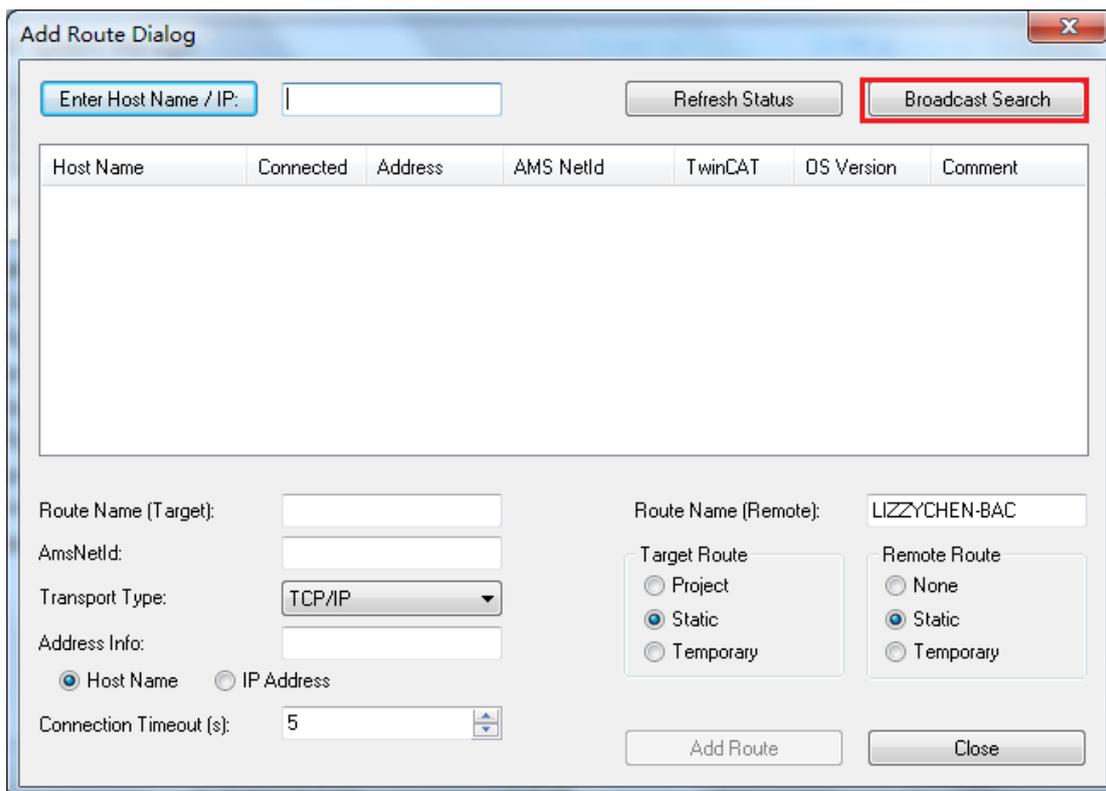
按照 2.4.1 和 2.4.2 中的说明设置好 IP 地址和 NetID 后，就可以添加路由表了。方法如下：在下图中点击“Choose Target System”：



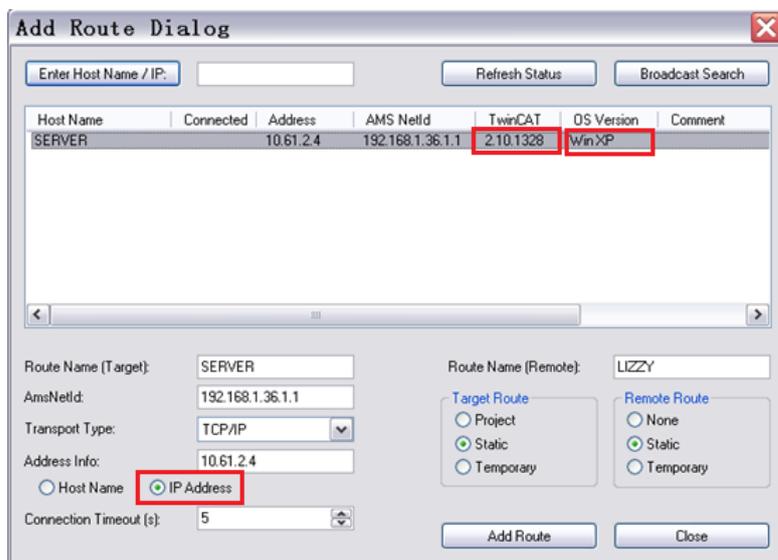
弹出“Choose Target System”窗体，点击 search Ethernet



在弹出的窗体中点击“Broadcast Search”



选中目标机器:



由于 CE 系统只支持 IPAddress 方式，并且这种方式当连接中断后再恢复时速度比较快。所以初学者推荐选用 IP Address 的方式。

点击 Add Router，进入用户登陆验证：

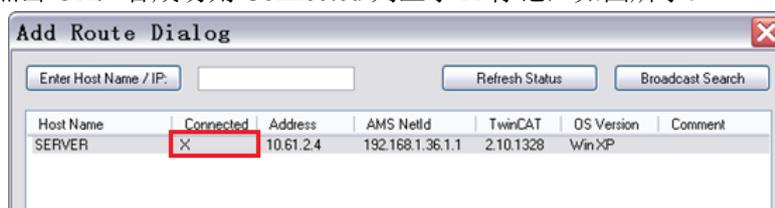


输入用户名和密码，

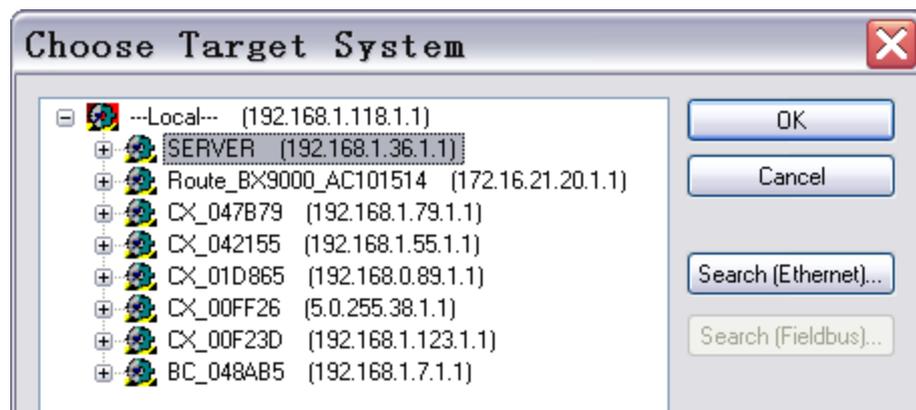
出厂设置：Windows XPe ， 用户名： administrator， 密码： 1；

出厂设置：Windows CE ， 用户名和密码均为空白。

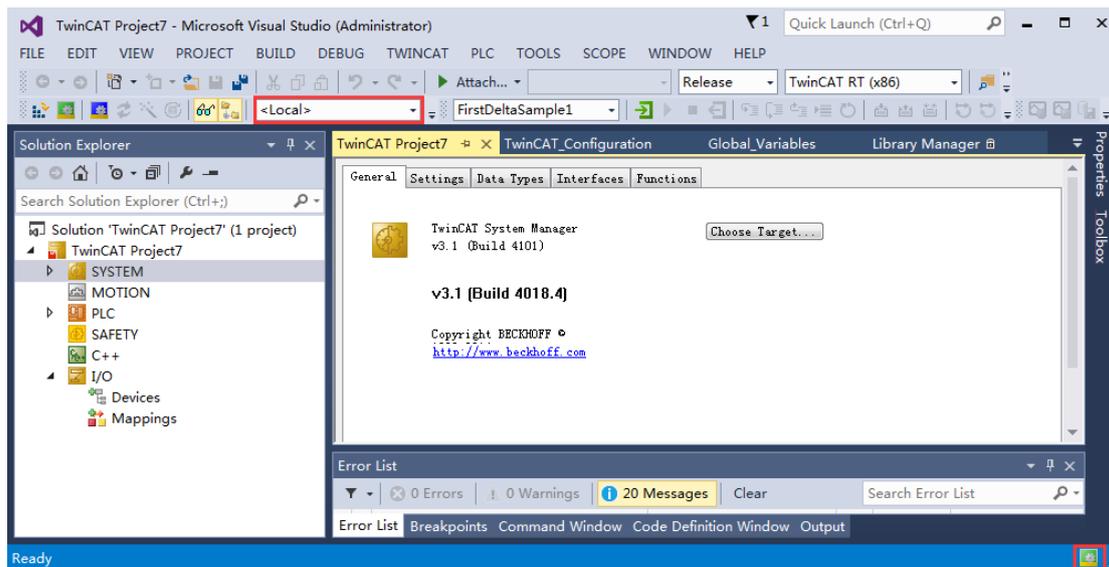
输入密码后点击 OK，若成功则 Connected 列显示 X 标记，如图所示：



点击 Close，返回前一窗体，可以见到刚刚添加的路由表项出现在列表中：



选中要配置的控制器，点击 OK，



如图所示，在窗体的右下角红色框中显示目标控制器的 TC3 运行核的状态，左上部红色框中显示当前登陆的目标控制器“Target”的名称和 NetID。

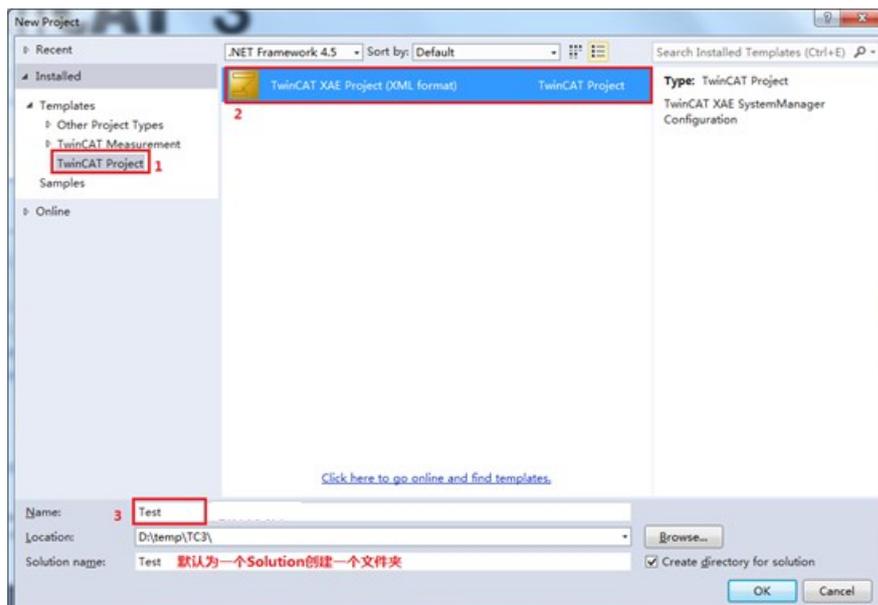
提示：

- 1, 如果仍然扫描不到或者加上不路由, 请参考“[小技巧: 加不上路由的若干可能性 V1.02.pdf](#)”
- 2, 连接中断后, 只要 System Manager 没有选择其它 Target, 编程 PC 就会连续尝试恢复与目标控制器的连接, 直到与连接恢复。中间可能导致对其它应用程序的响应缓慢甚至半死机状态。如果想中断连接尝试, 可以从编程 PC 的 TwinCAT 图标快捷菜单中将本机的 TwinCAT 状态切换到 Stop, 再切回 Config.

## 2.5. 开发第一个 PLC 项目

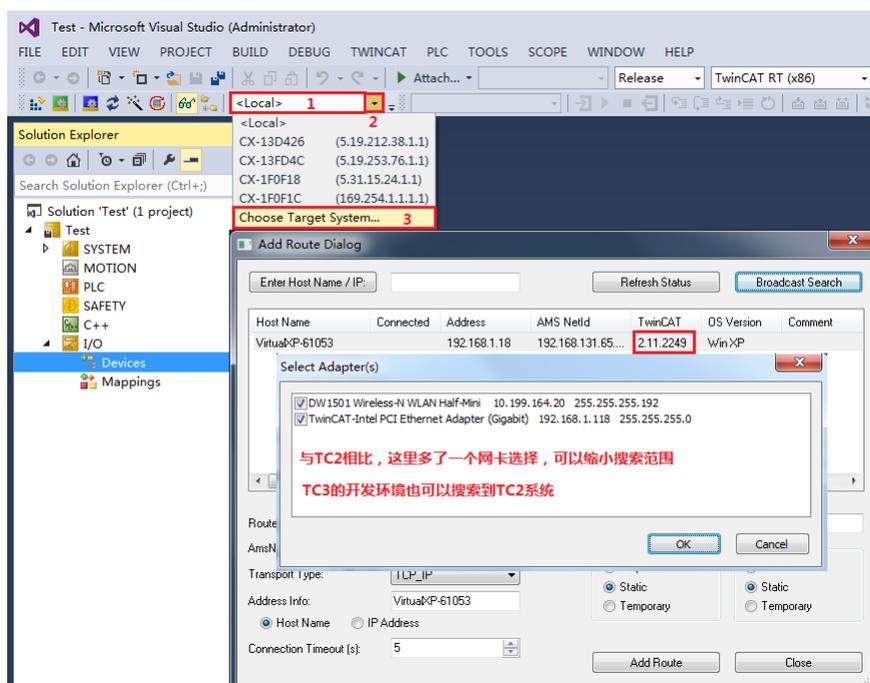
### 2.5.1. 建一个 TwinCAT 项目

Step 1. File|New|Project 进入以下页面：

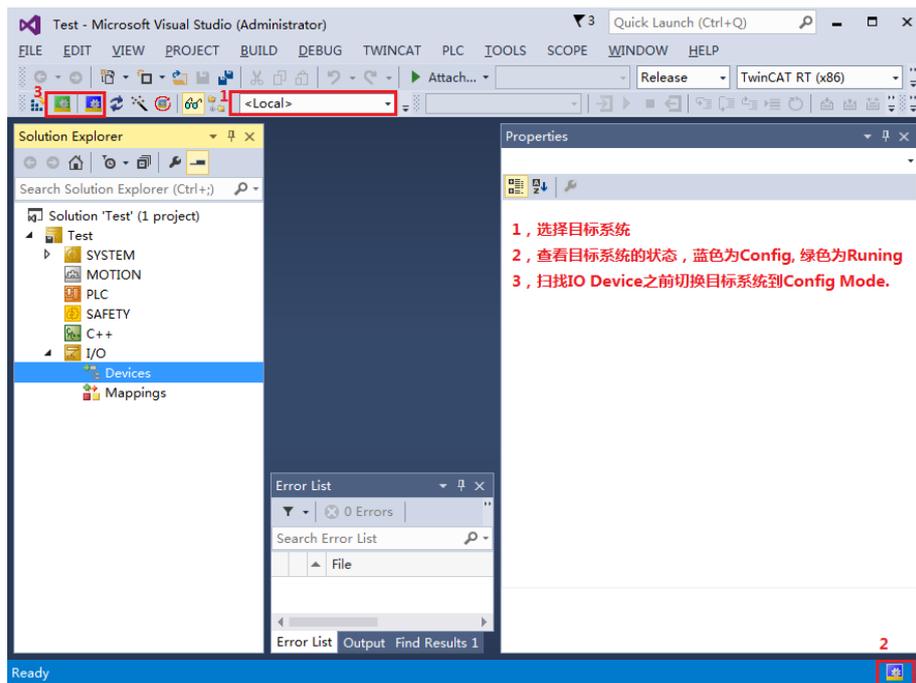


## Step 2. 选择目标系统 (Choose Target)

进入以下页面:

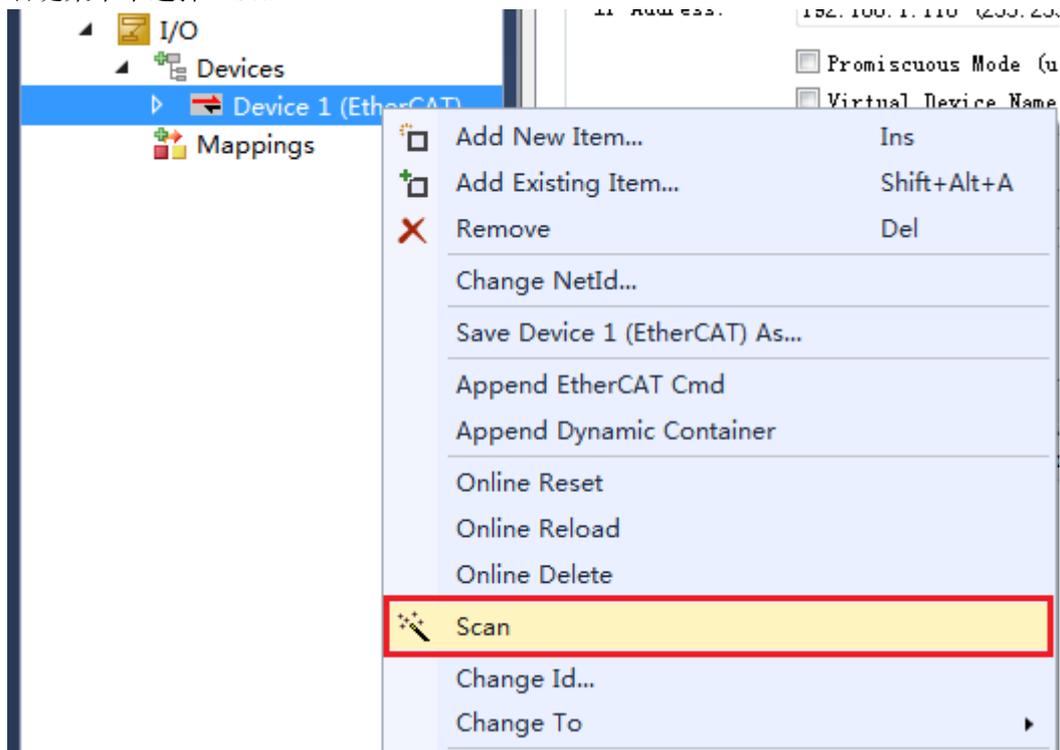


从这里可以进入熟悉的 Choose Target 及 Add Router 页面。

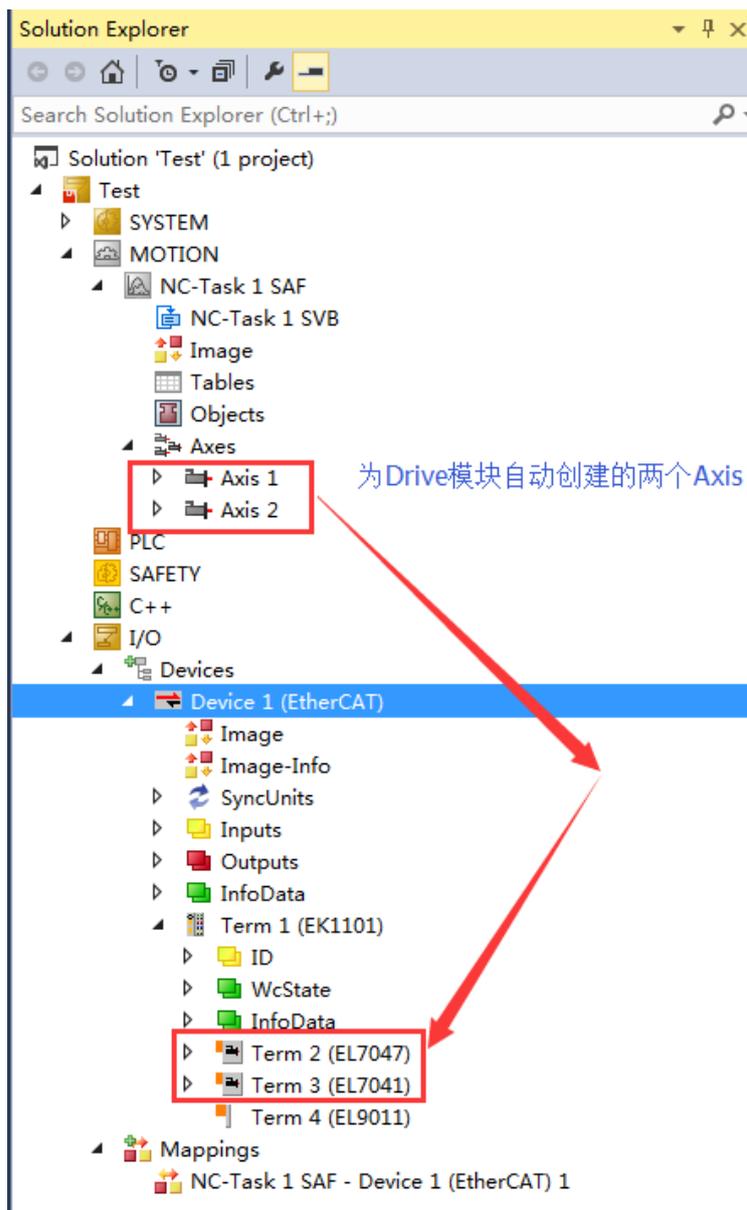


Step 3. 扫描 IO (Scan)

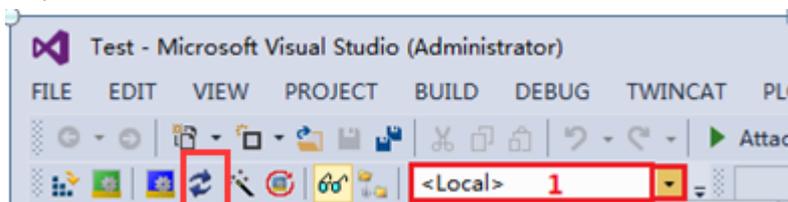
右键菜单中选择“Scan”



扫描到所有 EtherCAT 从站，并自动为 EL704x 步进驱动模块创建了两个 Axis。



#### Step 4. Free Run 模式测试通讯和 IO 模块



和 TC2 中一样，Free Run 的作用是脱离 PLC 程序测试通讯和 IO 模块是否正常，上图左边的红线框内图标，就是 Free Run 按钮。

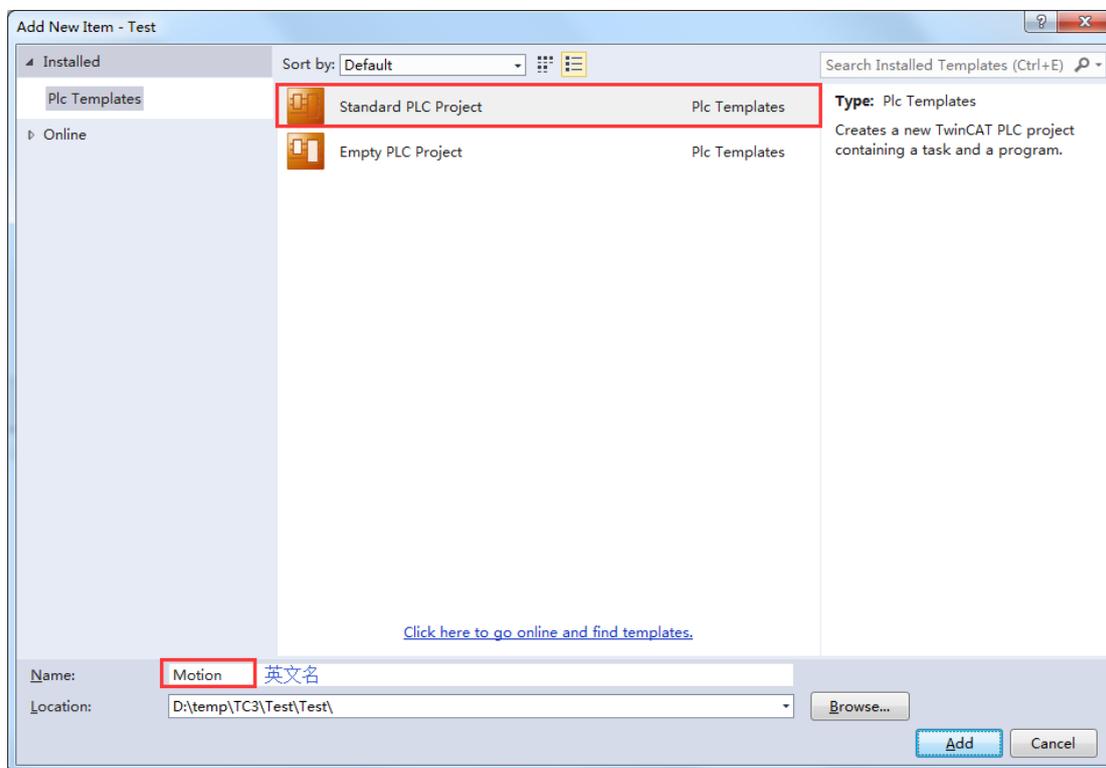
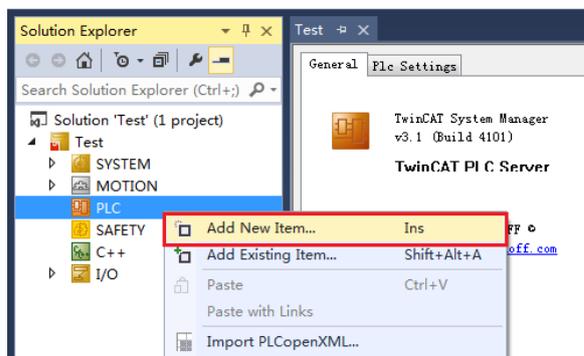
Free Run 还可以用于验证现有项目配置的 IO 与实际硬件是否相符。方法如下：

打开项目——Choose Target——Free Run

然后就可以在 EtherCAT 的 Online 页面查看通讯状态，以及手动 Write Output 变量了。

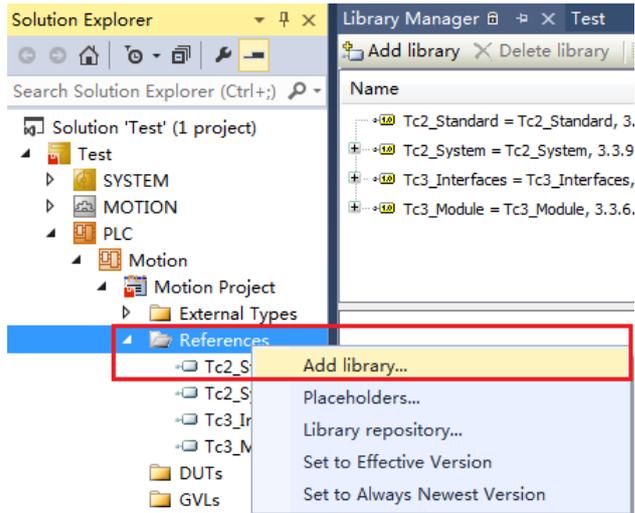
## 2.5.2. 建一个 PLC 项目

### Step 5. 添加一个 PLC Project

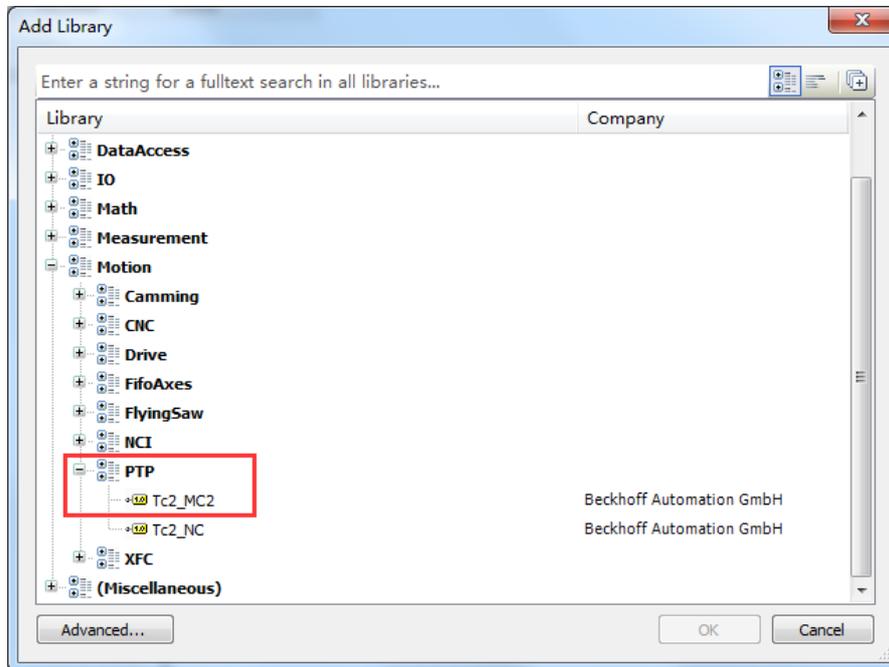


### Step 6. 添加库文件

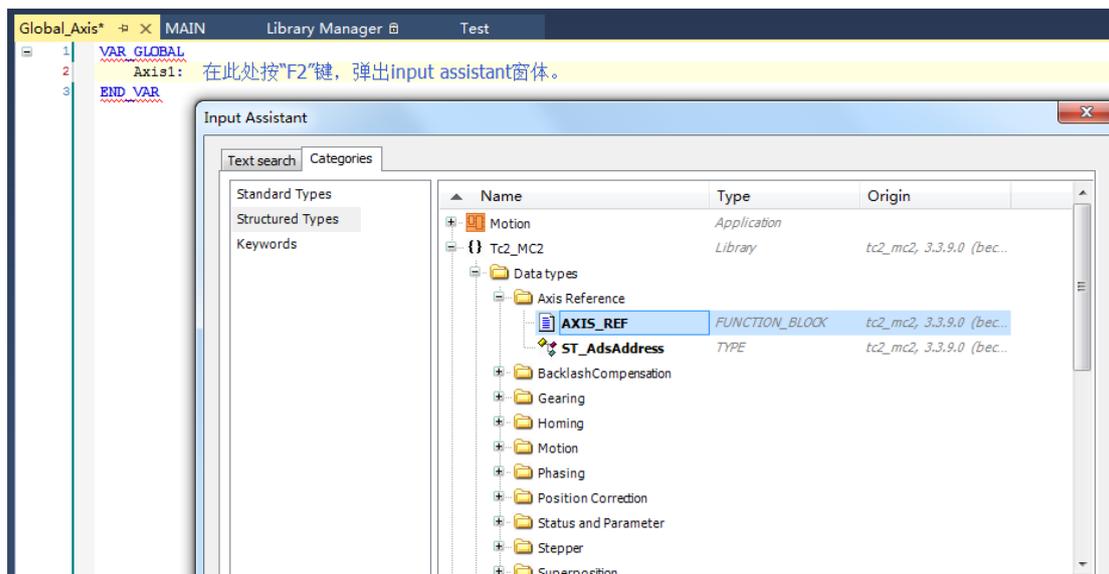
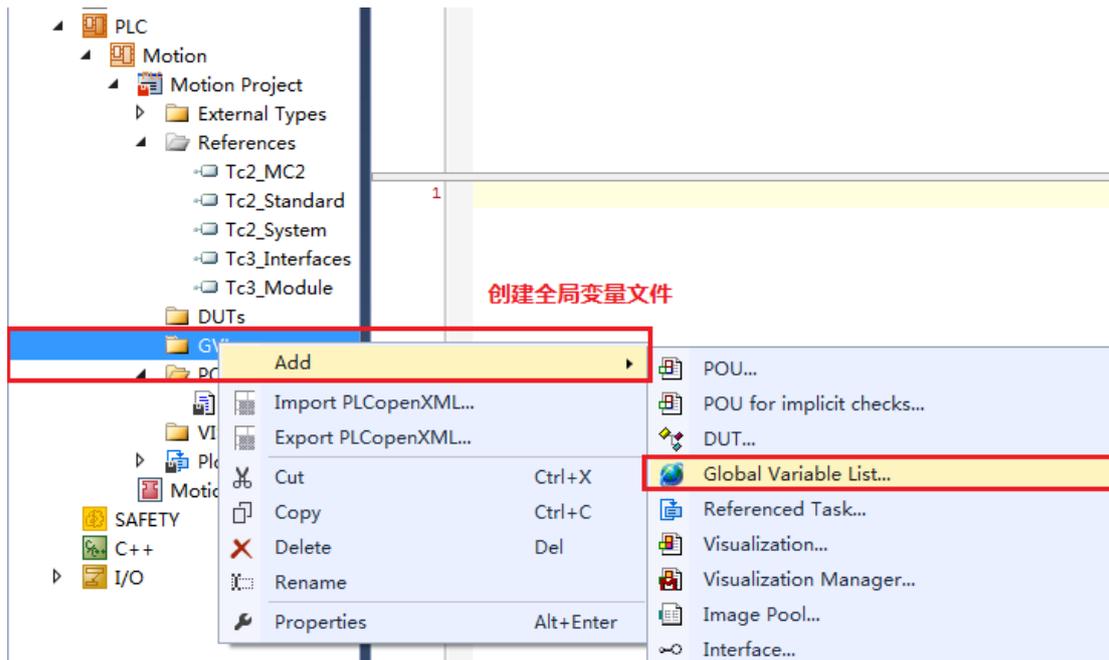
因为本例中是编写一个运动控制程序，所以要先引用 TC2\_MC2 库。如果是普通的 PLC 程序，可以直接跳到 2.5.11，编写其它逻辑。



选择库文件:

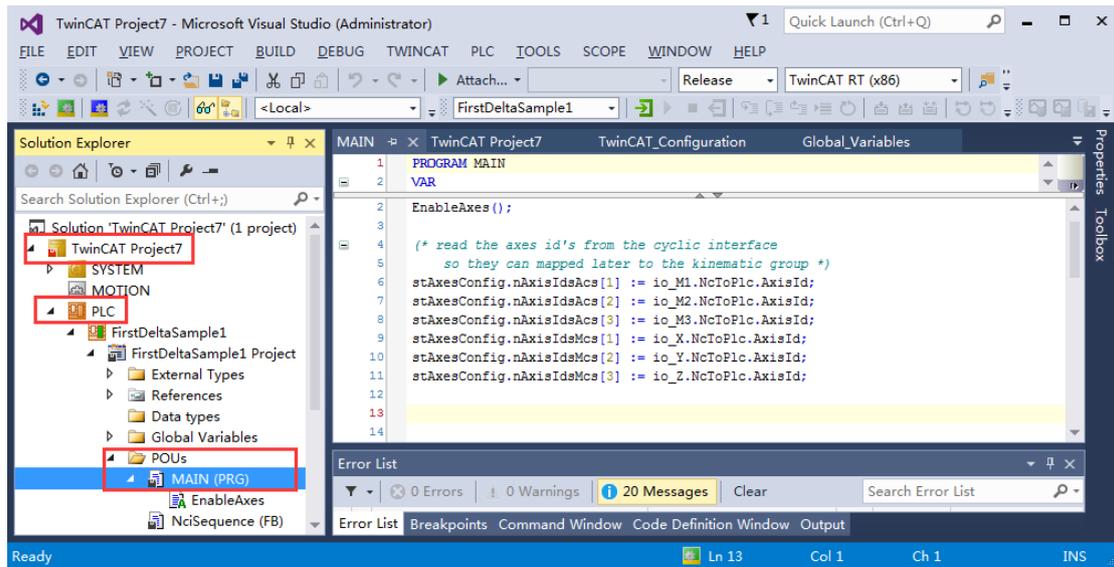


Step 7. 创建全局变量



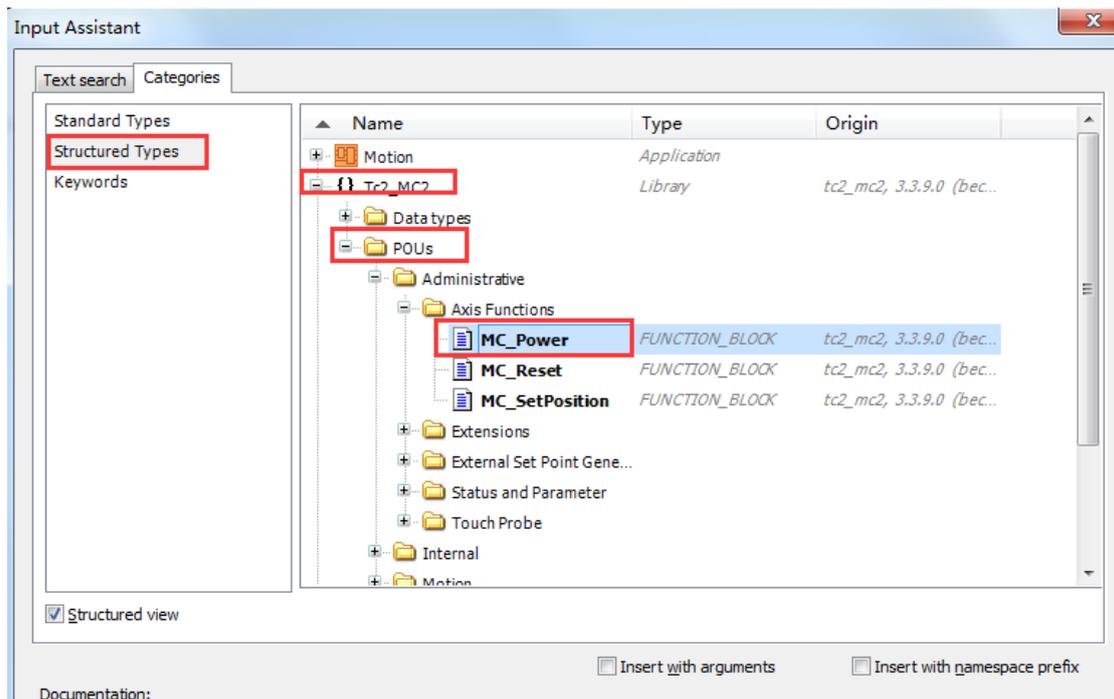
### Step 8. 声明 FB 实例

在 MAIN 的局部变量声明区，声明 FB 实例



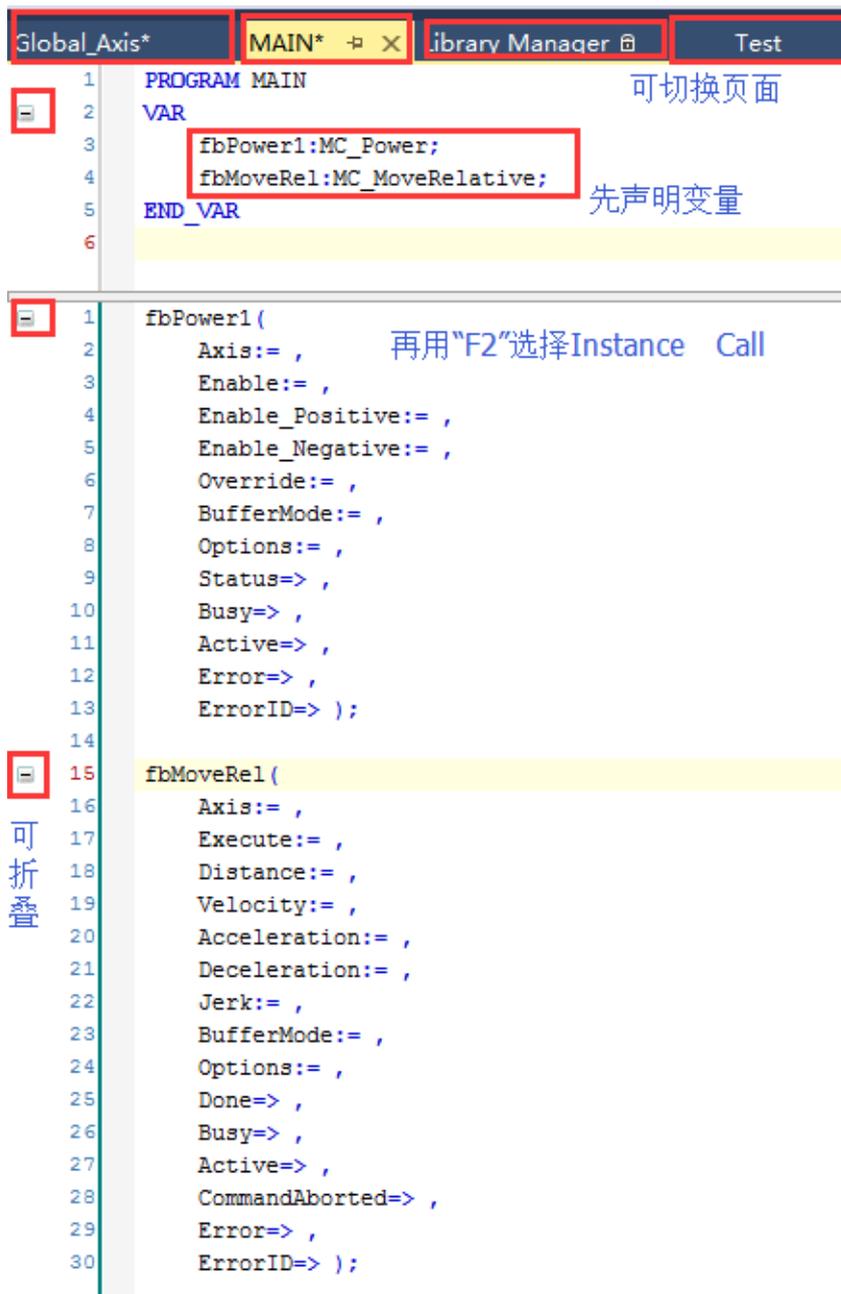
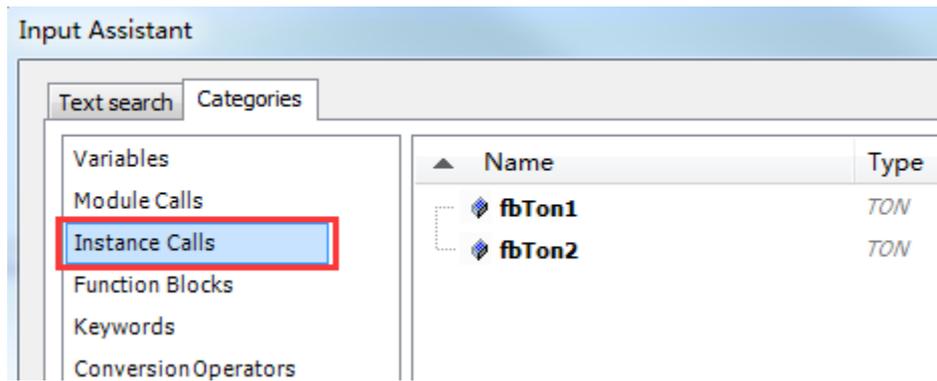
```
fbPower1:MC_Power;
fbMoveRel:MC_MoveRelative;
```

可以用手动输入或者 F2 辅助输入的方法:



### Step 9. Call Instance

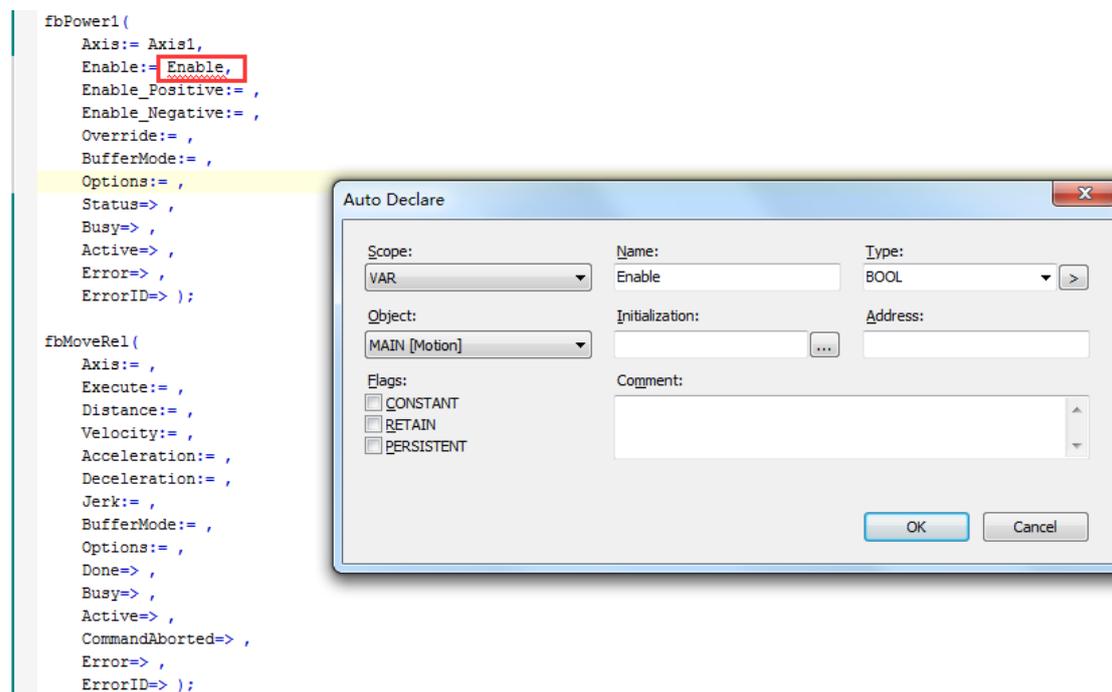
在 MAIN 程序的代码区, 按 F2 键, 选择 Call Instance, //要插入 Instance Call 才会有接口变量显示, 如果象 TC2 一样插入 Local 变量, 就没有



提示：编辑器实际上并没有 FB 识别功能，而是根据代码行的缩进来决定如何折迭的。

**Step 10. 填写接口变量**

方法与 TC2 完全相同：

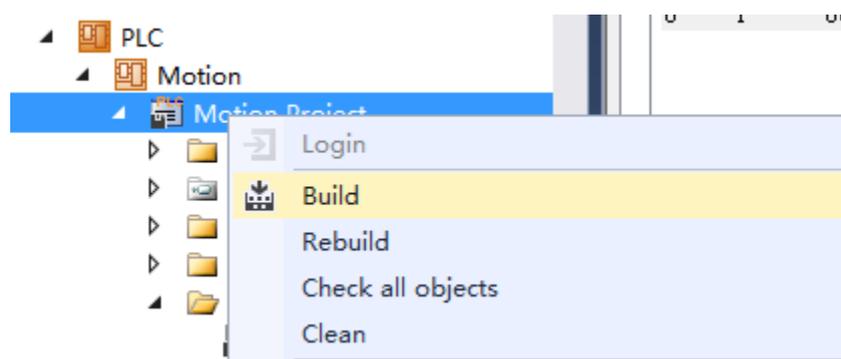
**Step 11. 编写其它逻辑**

至此，一个最简单的程序就编写完成了，再次存盘。

*说明：PLC 编程时对变量名的不区分大小写。*

**Step 12. 编译和试运行**

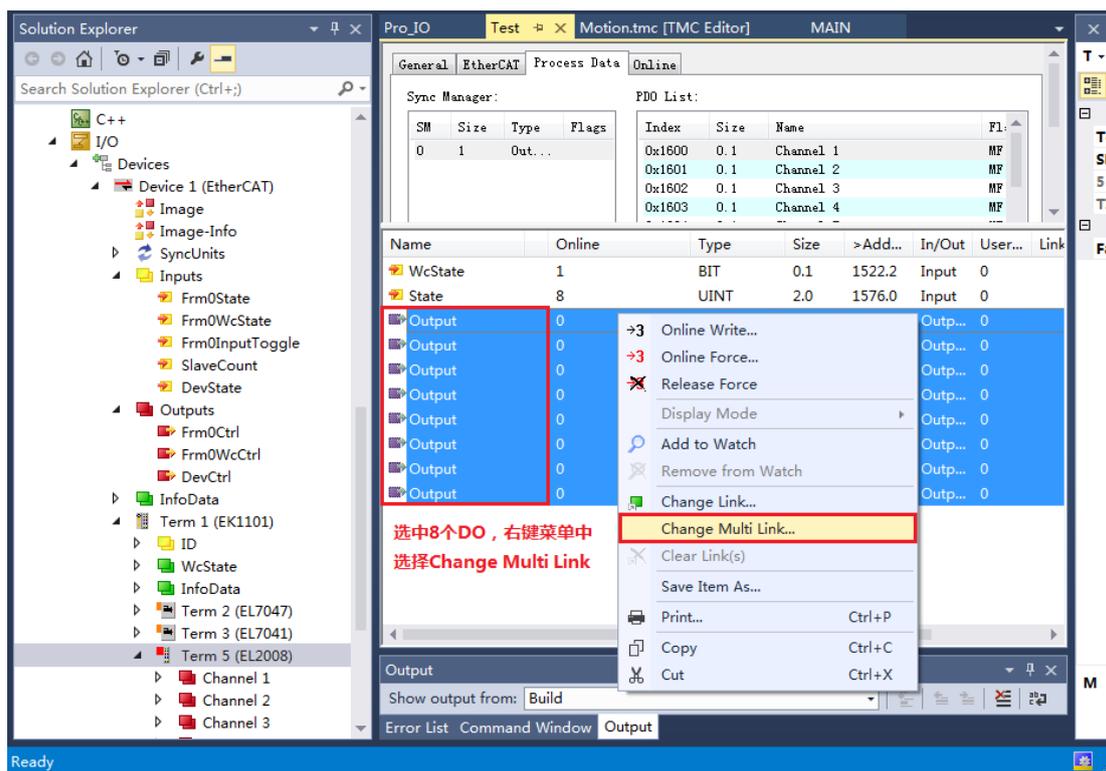
先编译 PLC Project



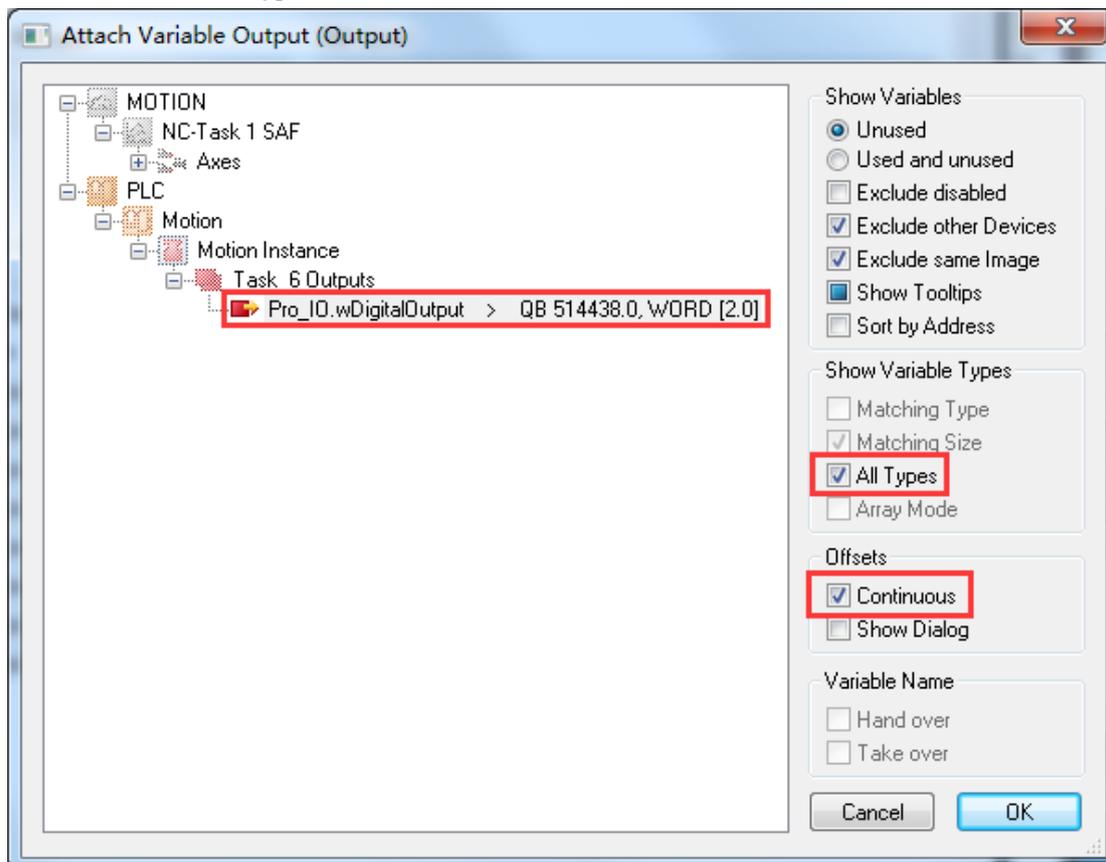
编译成功才会出现 PLC 程序名的 Instance，比如本例中的 Motion Instance.

**2.5.3. PLC 变量映射和激活配置**

Step 13. 把 8 个 DO 通道链接到 PLC 的输出变量：



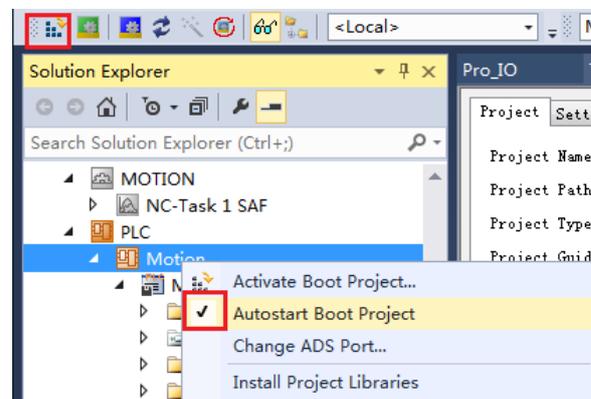
过滤器是勾选“**All Type**”和“**Continous**”，选中 PLC 变量。



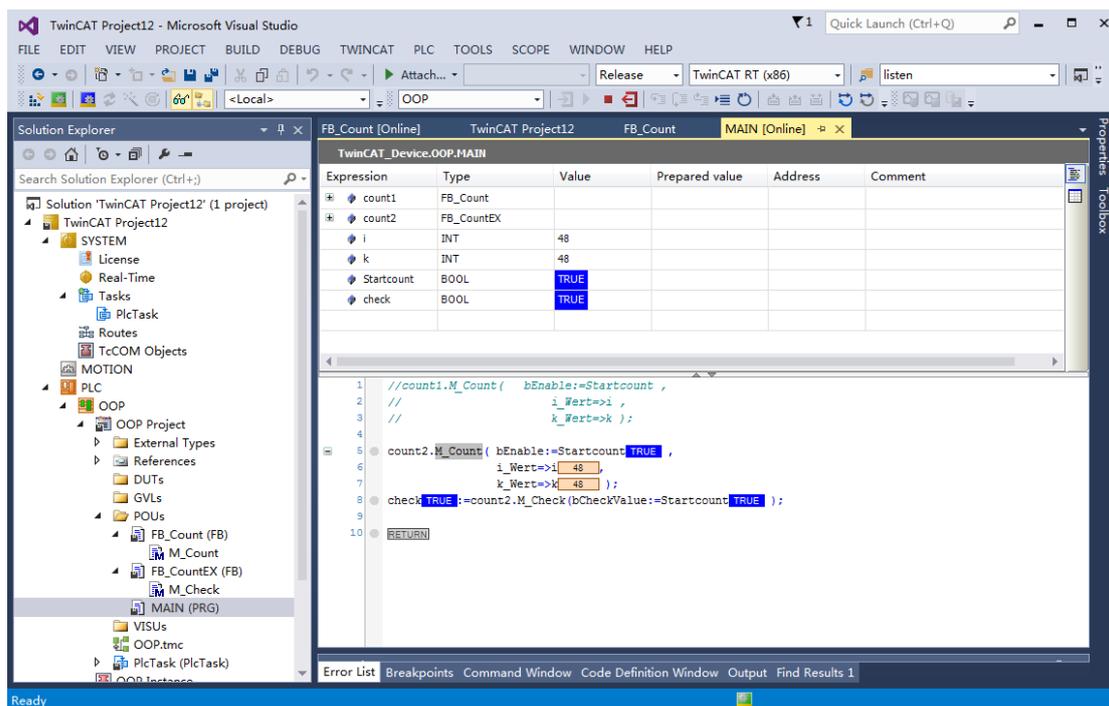
就可以看到 8 个 DO 都链接上了：

Name	Online	Type	Size	>Add...	In/Out	Use
WcState	1	BIT	0.1	1522.2	Input	0
State	8	UINT	2.0	1576.0	Input	0
Output	X 0	BIT	0.1	67.0	Outp...	0
Output	X 0	BIT	0.1	67.1	Outp...	0
Output	X 0	BIT	0.1	67.2	Outp...	0
Output	X 0	BIT	0.1	67.3	Outp...	0
Output . Channel 3 . Term 5 (EL2008) . Device 1 (EtherCAT) . Devices	X 0	BIT	0.1	67.4	Outp...	0
Output	X 0	BIT	0.1	67.5	Outp...	0
Output	X 0	BIT	0.1	67.6	Outp...	0
Output	X 0	BIT	0.1	67.7	Outp...	0

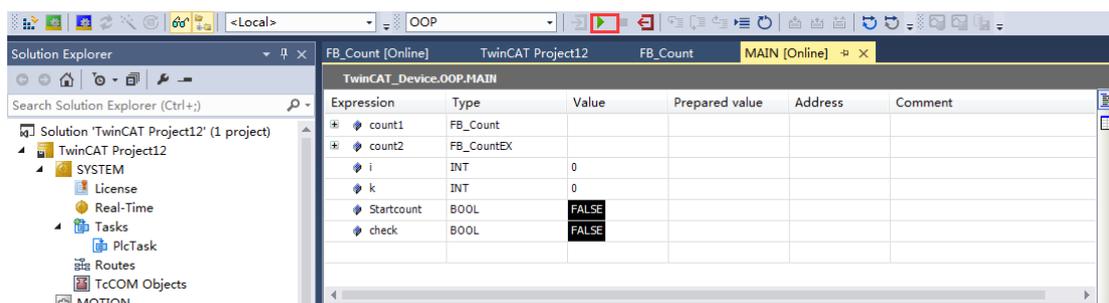
Step 14. 激活配置，运行程序：



上图中勾选了“AutoStart Boot Project”，程序就会自动运行。



如果没有勾选“AutoStart Boot Project”，就会看到如下界面：



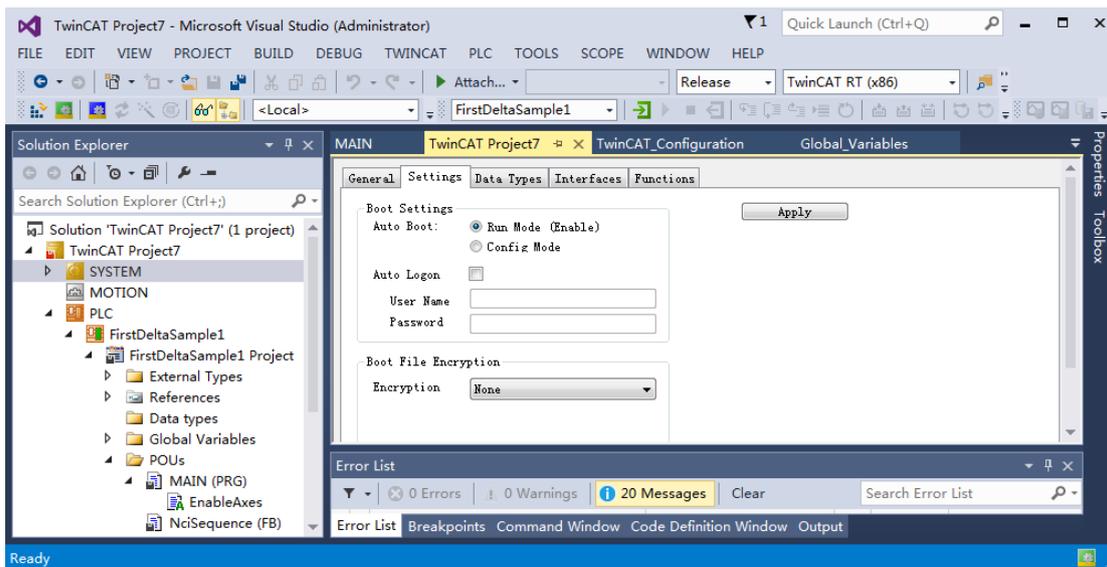
此时，需要点击红框中的运行按钮，或者“F5”，程序才会运行。

## 2.5.4. 设置开机自启动

在前面的调试阶段，Login 时，程序只是下载到内存运行。为了让控制器断电重启后还能按调试成功的程序运行，必须把程序执行码下载到控制器存储卡或者硬盘的指定路径。经过若干设置步骤，控制器重启后，TwinCAT 会自动到该路径下找到指定文件，装载到内存里运行。这个操作就称为“设置开机自启动”。

设置开机自启动的步骤为：

第 1 步：TwinCAT 启动模式。



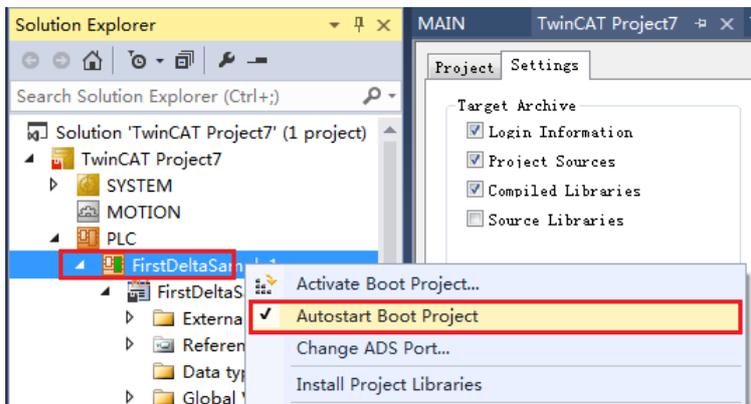
将 Boot Settings 的 Auto Boot 项目设置为 Run Mode (Enable) , 点击 Apply:



对 CE 系统, User name 和 Password 都为空白。对 XPe 系统, 默认的 User name 为 “Administrator”, Password 为 “1”。如果是 IPC, 则输入操作系统上具有管理员权限的某个用户名和密码。

如果设置成功, 下次 Window 启动完成后, TwinCAT 将自动进入 Runing 模式。

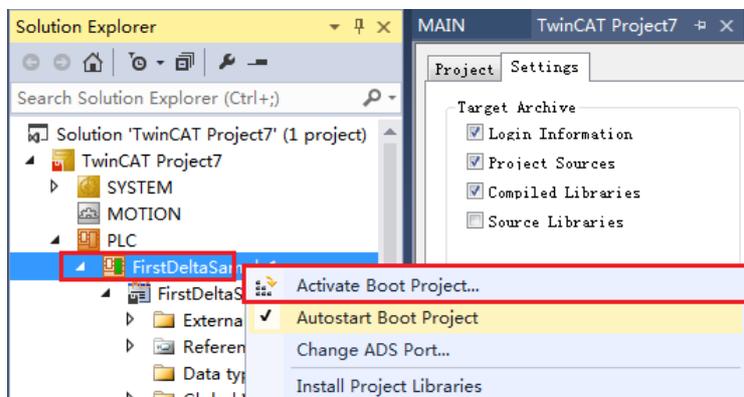
第 2 步, PLC 启动设置。



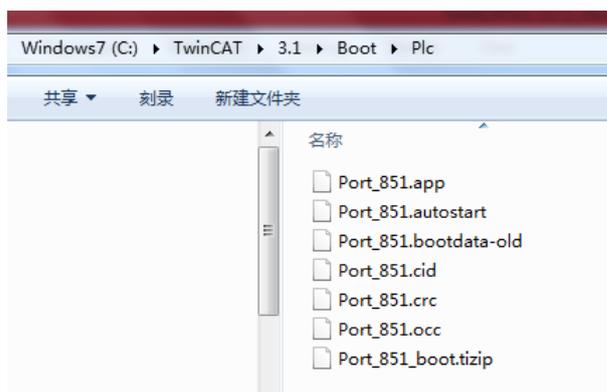
如果设置成功, 下次 TwinCAT 进入 Runing 模式后, 将自动到指定路径下找到指定文件, 装载到内存里运行。

提示：特殊情况下，不想让 PLC 程序自启动，也可以在此取消设置。

### 第 3 步：创建引导程序



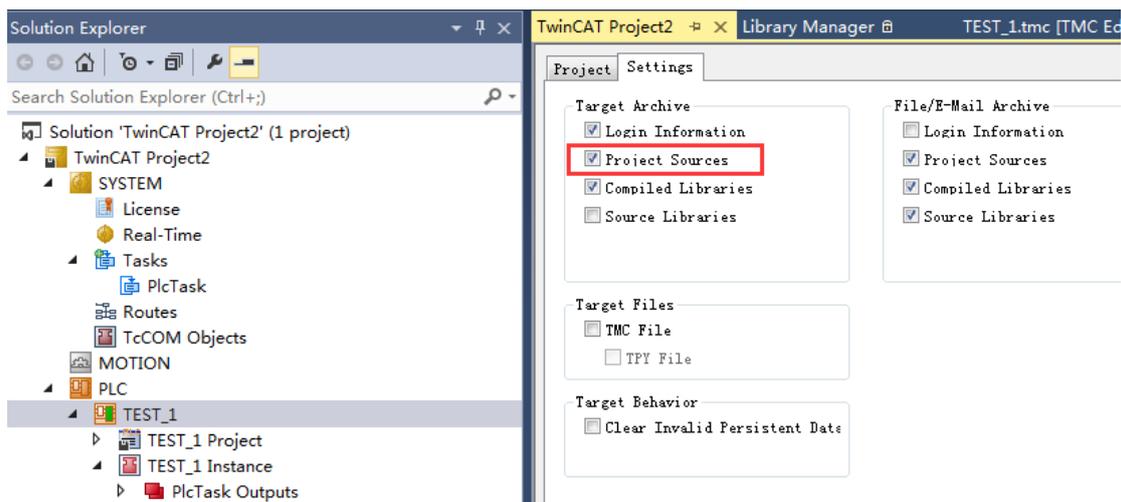
Login 状态下，点击上图中的 Activate Boot Project 菜单项，开发 PC 就会把程序执行码下载到控制器的存储卡或硬盘的指定路径\TwinCAT\3.1\Boot\Plc。



## 2.6. 上传、下载和比较

### 2.6.1. PLC 程序的上传下载和比较

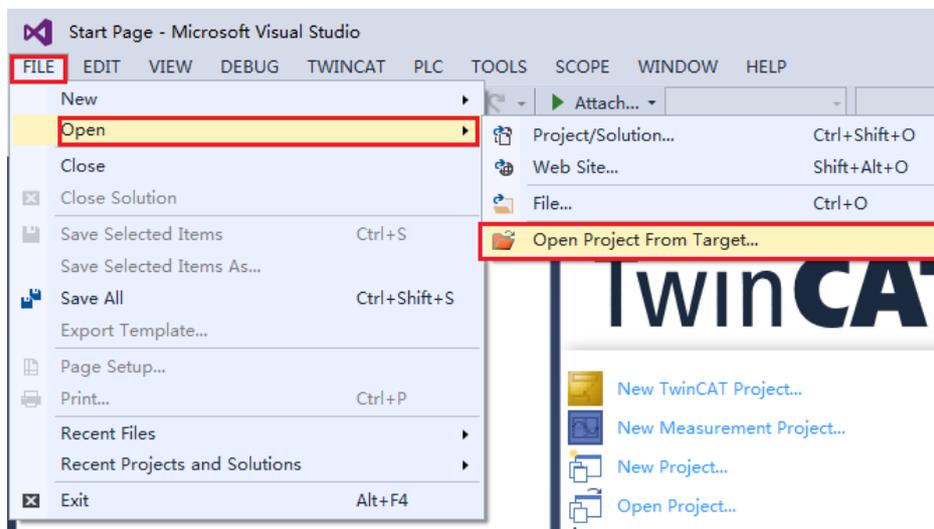
- 下载源代码



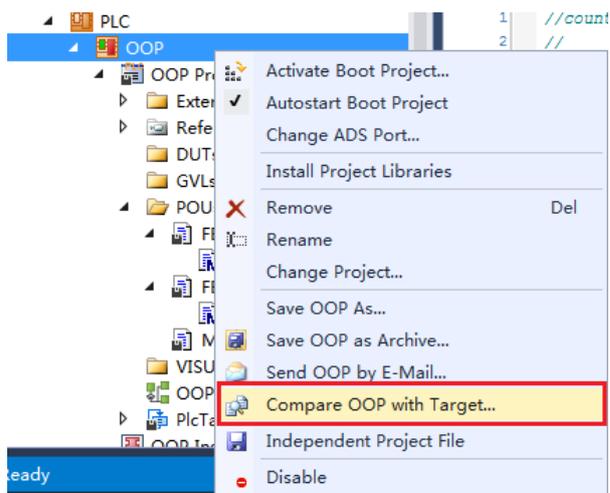
Tips : PLC Setting 的默认源代码下载和 Creat Boot Project 都是启用的，建议全部关闭。

- 上传源代码

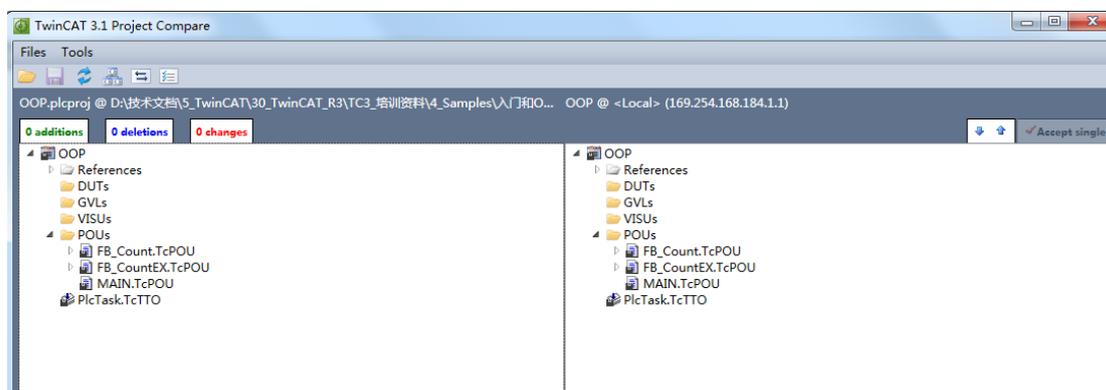
必须关闭 VS，再重新打开。然后从 FILE、OPEN、选择 Open From Target



- PLC 程序的比较



比较的结果:



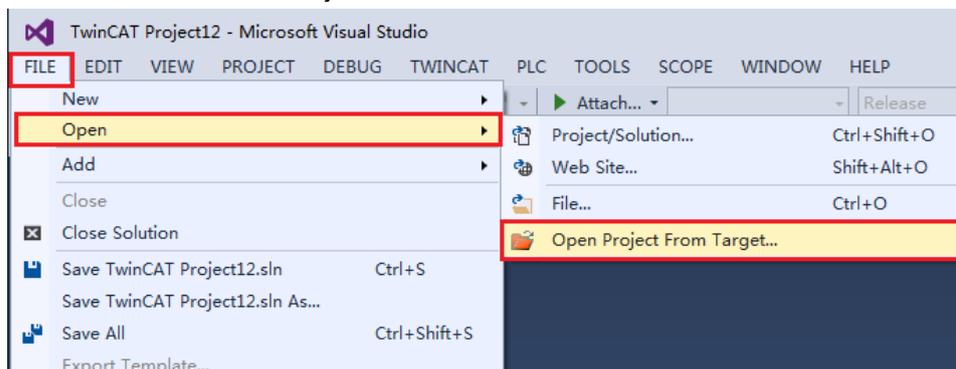
## 2.6.2. TwinCAT 项目的上传和比较

- 下载

TwinCAT 项目的下载，实际上就是激活运行的过程。

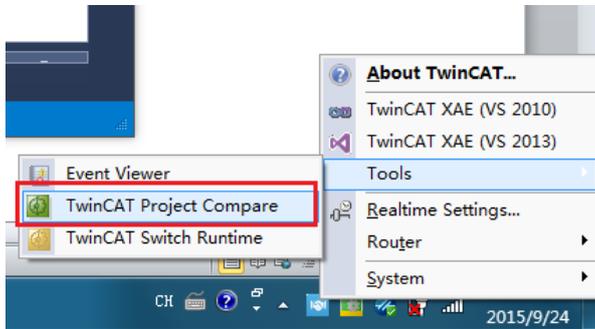
- 上传

新建一个空白项目，从主菜单 File | Open | Open Project FromTarget, 就可以上传整个当前运行的 TwinCAT Project。

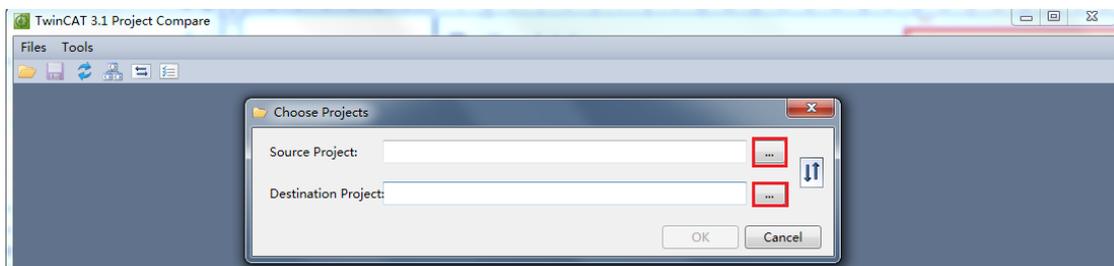


- 比较

整个 TwinCAT 项目的比较是在 VS 开发环境之外独立进行的。从 TC3 图标的右键菜单处进入：



用户可以指定任意两个项目进行比较。



### 3. TwinCAT 3 开发环境的深入介绍

本章不作重点描述但初学者必须掌握的内容有：

菜单、按钮和工作区的识别

IEC61131-3 的 5 种编程语言语法详解

操作符和操作数，比如数学运算、字符运算、比较指令、移位指令等

Standard.lib 中的 FB 用法，比如定时器、计数器、上升沿、下降沿

组态画面编辑器 Target Visual 的使用



任何类型的变量声明都包含这 4 个基本要素。  
不需要具体寻址的中间变量，有这 4 个元素就够了。

### 3.1.2. 变量类型

TC3 中新增了以下数据类型：

64 位数据：LINT，ULINT，LWORD 分别 64 位有符号整数、无符号整数、8 字节 WORD。  
时间

LTIME	tTime:= LTIME#12s34ms2us44ns
LDATE	DT:= CONCAT_DATE_TOD(DATE, TOD);
LDATE_AND_TIME	DATE:= CONCAT_DATE(YEAR, MONTH, DAY);
LTIME_OF_DAY	

UNION：多种数据类型共享内存区。类似于结构，但 Union 里面的各元素首地址是对齐的，而 Structure 里的各元素首地址是首尾相接的（不考虑字对齐的情况）。

WSTRING：Unicode 格式的字符，比如显示中文。

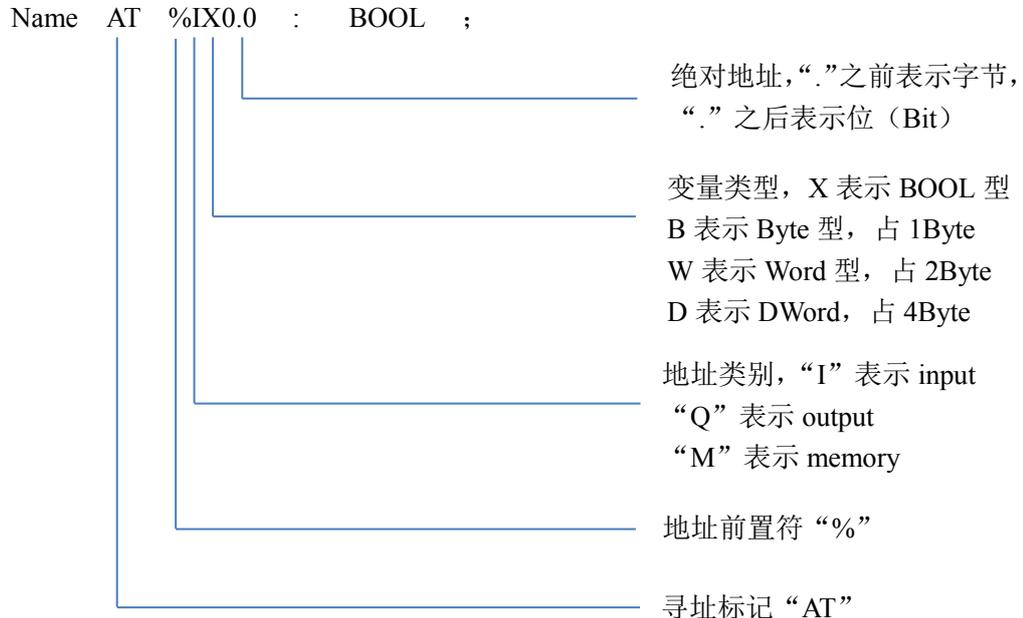
以下类型兼容 2 中的规范：

变量类型	最小值	最大值	内存占用
BOOL	False	True	1Bit
BYTE	16#00	16#FF	1 Byte
WORD	16#0000	16#FFFF	2 Byte
DWORD	16#0000 0000	16#FFFF FFFF	4 Byte
SINT	-128	127	1 Byte
USINT	0	255	1 Byte
INT	-32768	32767	2 Byte
UINT	0	65535	2 Byte
DINT	-2147483648	2147483647	4 Byte
UDINT	0	4294967295	4 Byte
REAL	-3.402823 x 1038	3.402823 x 1038	4 Byte
LREAL	1.79769313486231 e308	1.79769313486231 e308	8 Byte
STRING	String(35)表示35个英文字符，String()表示255字节的英文字符。		
TIME	T#0ms	T#71582m47s295ms	4 Byte
DATE	D#1970-01-01	D#2106-02-06	4 Byte
以下为自定义数据类型			
ARRAY	数组	Array[1..100]	不确定
POINTER	指针	Pointer to DataType	8 Byte

ENUM	枚举		2 Byte, 类似整数
STRUCT	结构		不确定

### 3.1.3. 变量声明中的绝对地址

输入变量、输出变量和需要按地址访问的 M 区变量，声明时需要增加“地址”部分，语法如下：



注意，

- 1, 对于超出 4 个字节的变量类型，统一使用“B”，即 Byte。
- 2, 不同类型的变量，只要地址值相同，就是指向同一个数值。

比如：

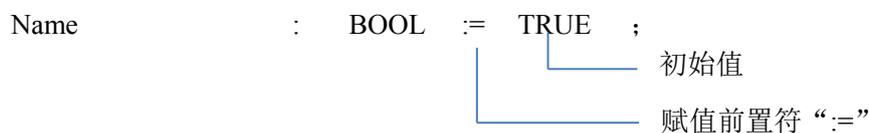
```

%ID100, 包含%IW100 和%IW102,
%IW100, 包含%IB100 和%IB101,
%IB100, 包含了%IX100.0 到%IX100.7

```

### 3.1.4. 变量声明中的赋初值

输入变量、输出变量和需要按地址访问的 M 区变量，声明时需要增加“地址”部分，语法如下：



如果是数组或者结构型变量，需要为多个元素赋初值，则以逗号分隔。多个相同变量以（）标记。比如：

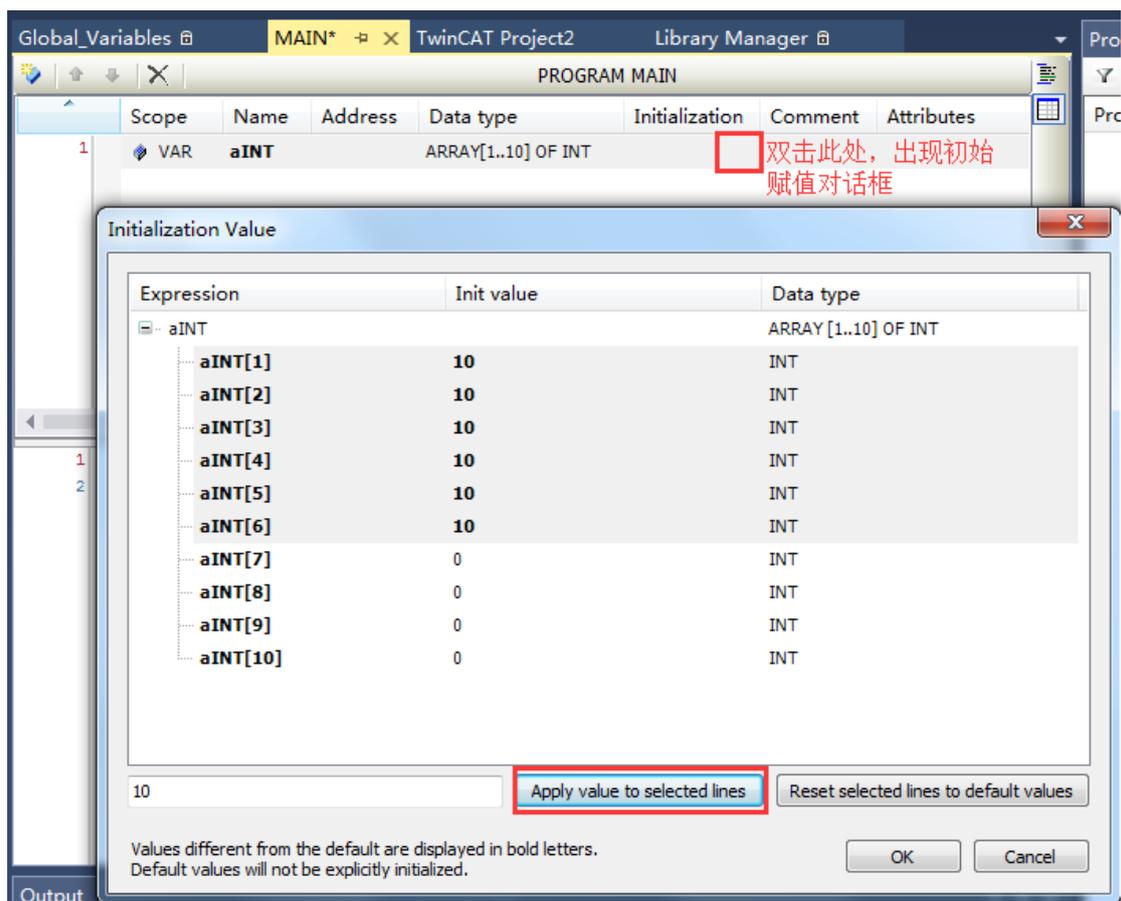
```

Array1 : Array[1..6] OF INT:= 1,2,3,3(10);

```

语句声明一个 10 个元素的 INT 型数组，10 个元素的初值分别为 1, 2, 3, 7, 7, 7。

在 TwinCAT 3 中，数组赋初值有更简捷的方法：



对数组很方便，但对结构来说，还是程序处理比较方便。

### 3.1.5. 为 IO 变量自动分配地址

在 TwinCAT PLC 中，理论上只有需要与硬件对应的 IO 变量，才需要分配绝对地址。并且由于 PLC 的 Input 区和 Output 区和 IO 模块的安装位置无关，所以即使是 IO 变量，编写程序时也可以没有确切的地址。

在 TC3 中，编译 PLC 程序不会产生 TC2 中的 Tpy 文件，但编译成功会出现 PLC 的 TMC 接口对象，这时就已经为 IO 变量自动分配好 PLC 地址了。

### 3.1.6. 变量声明时的缩写输入法

B or BOOL	ergibt	BOOL
I or INT	ergibt	INT
R or REAL	ergibt	REAL
S or string	ergibt	STRING

Shortcut	resulting declaration
A	A: BOOL;
AB I 2	A, B: INT := 2;
ST S 2; A string	ST:STRING(2); (* A string *)
X %MD12 R 5 Real Number	X AT %MD12: REAL := 5.0;(* Re

除了这个没试出来

### 3.1.7. 变量的属性

属性用{}括起来。

```
// Main logical
{This is my Comment for Main Local Variable}
PROGRAM MAIN
VAR
  {attribute 'displaymode':='hex'}
  aINT : ARRAY[1..2] OF INT := [2(12)];

  para1: ARRAY[1..3] OF int := [3(24)];
END_VAR
```

显示为表格就是:

Scope	Name	Address	Data type	Initialization	Comment	Attributes
1	VAR aINT		ARRAY[1..2] OF INT	[2(12)]		attribute 'displaymode':='hex'
2	VAR para1		ARRAY[1..3] OF int	[3(24)]		

attribute 'displaymode':='hex'

关键词不能写错。一共有多少关键词:

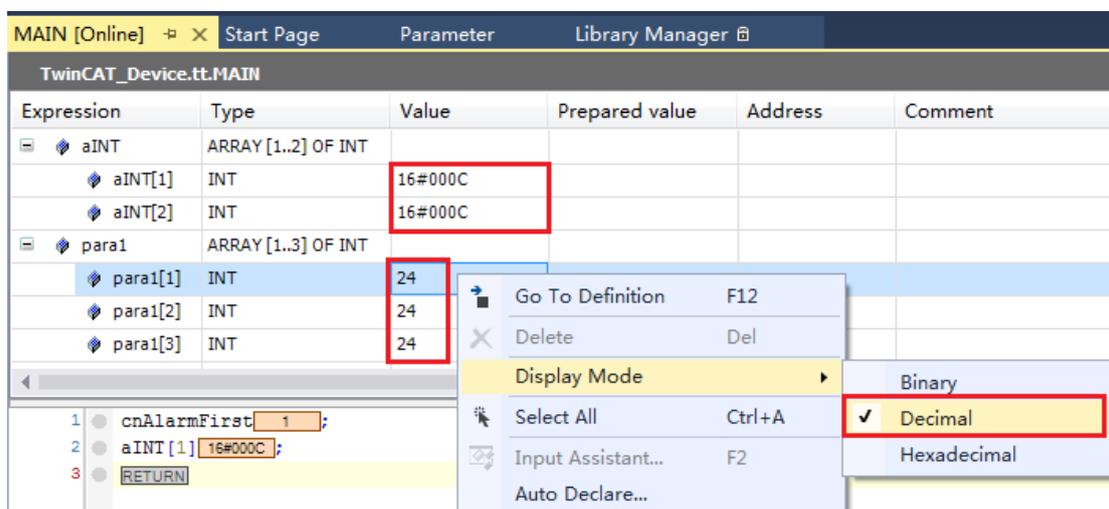
可自定义的属性

- User-defined Attributes
- Attribute Call\_after\_init
- Attribute displaymode
- Attribute ExpandFully
- Attribute Global\_init\_slot
- Attribute Hide
- Attribute hide\_all\_locals
- Attribute Initialize\_on\_call
- Attribute Init\_Namespace
- Attribute Init\_On\_Onlchange
- Attribute Instance-path
- Attribute linkalways
- Attribute Monitoring
- Attribute No\_check
- Attribute No\_Copy
- Attribute No-exit
- Attribute no\_init, no-init, noint
- Attribute No\_virtual\_actions

- Attribute Obsolete
- Attribute Pack\_mode
- Attribute Parameterstringof
- Pingroup attribute
- Attribute Qualified\_Only
- Attribute Reflection
- Attribute TcCallAfterOutputUpdate
- Attribute TcContextID
- Attribute TcContextName
- Attribute TcInitSymbol
- Attribute TcLinkTo / TcLinkToOSO
- Attribute TcNcAxis
- Attribute TcDisplayScale
- Attribute TcGvlVarNames
- Attribute tc\_no\_symbol



在线显示时，定义了属性的变量，按指定的格式显示，而其余变量，按右键菜单 Display Mode 的选择显示。

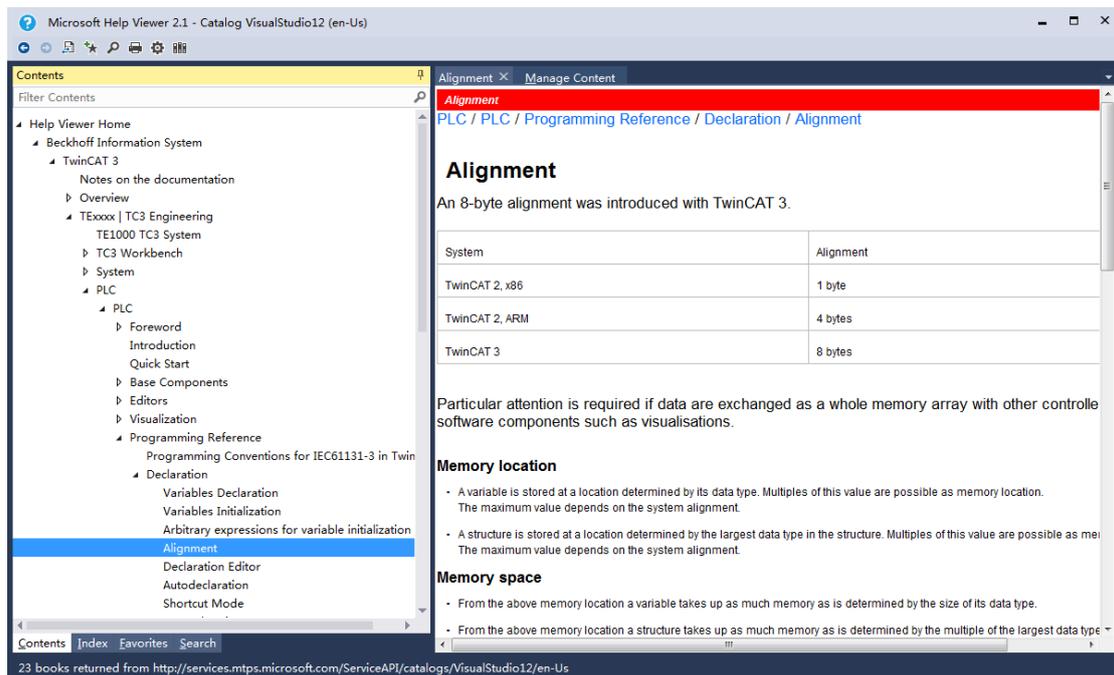


### 3.1.8. 地址对齐

DWords / Words				Bytes	X (bits)					
byte addressing		word oriented IEC addressing			byte addressing			word oriented IEC addressing		
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
	W4	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 = byte number

关于对齐，TC3 的 InfoSys 讲得极详细：



ARM: 4 字节对齐。

TC2: 1 字节对齐。

TC3: 8 字节对齐。(以结构里最长的元素)

%I\*的地址，在哪里查看。变量%qb0 的地址，看到是 20000 多。

## 3.2. 编程语言和新增功能

### 3.2.1. ST 中增加了 Continue 和 Jump 语句。

#### CONTINUE instruction

As an extension to the IEC 61131-3 standard the CONTINUE instruction is supported. CONTINUE makes the execution proceed with the next loop-cycle.

Example:

```
FOR Counter:=1 TO 5 BY 1
DO
  INT1:=INT1/2;
  IF INT1=0 THEN
    CONTINUE; (* to avoid division by zero *)
  END_IF
  Var1:=Var1/INT1; (* only executed, if INT1 is not "0"
*)
END_FOR;
Erg:=Var1;
```

#### Example

```
aaa:=0;
_label1: aaa:=aaa+1;
(*instructions*)
IF (aaa < 10) THEN
  JMP _label1;
END_IF;
```

注释：单行注释“//”，多行注释（\*\*）

### 3.2.2. 指令：BitAdr(),用于定位到 Bit。

VAR

var1 AT %IX2.3:BOOL;

bitoffset: DWORD;

END\_VAR

bitoffset:=BITADR(var1); (结果为 19)

Real\_To\_Int:四舍五入, 结果是 INT。

Trunc:结果是 DINT

Trunc\_Int: 结果是 INT

ANY\_NUM\_TO\_<numeric data type>

ANY\_TO\_ <any data type>

Example:

Conversion from a variable of data type REAL to INT:

```
re : REAL := 1.234;
```

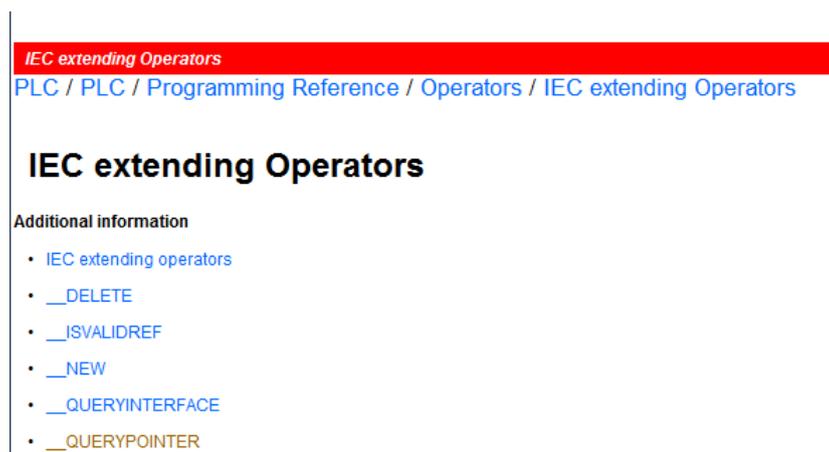
```
i : INT := ANY_TO_INT(re)
```

EXPT, SQRT

### 3.2.3. UML ChartSate 编程

除了原来的 IEC 编程方式外，还增加了 UML ChartSate 编程

### 3.2.4. 指针操作增加



The screenshot shows a web page with a red header bar containing the text "IEC extending Operators". Below the header is a breadcrumb trail: "PLC / PLC / Programming Reference / Operators / IEC extending Operators". The main title is "IEC extending Operators". Underneath, there is a section "Additional information" followed by a bulleted list of operators: "IEC extending operators", "\_\_DELETE", "\_\_ISVALIDREF", "\_\_NEW", "\_\_QUERYINTERFACE", and "\_\_QUERYPOINTER".

枚举的元素调用;

```
color := Colors.Blue; // Access to enum value Blue in type Colors  
feeling := Feelings.Blue; // Access to enum value Blue in type Feelings
```

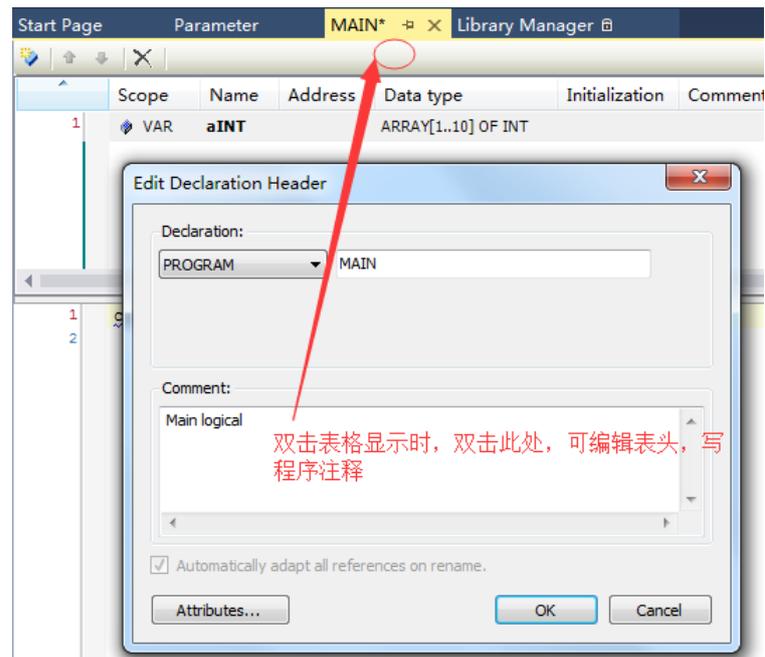
### 3.2.5. 支持变量作为 Bit 值访问

在 TwinCAT 2 中，要把一个 INT 或者 WORD 型变量的某一位取出来用，Varname.n 中的 n 必须是个常数，在 TC3 中，允许用变量作为 Bit 值访问了。

假如 enable:=2;

xxx.enable:=true; (\* -> the third bit in variable xxx will be set TRUE \*)

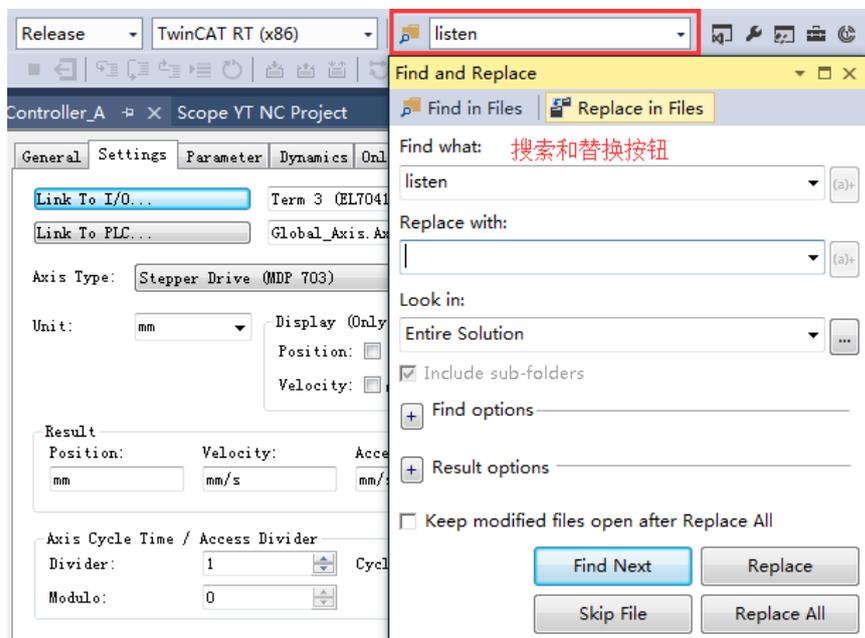
### 3.2.6. 程序注释



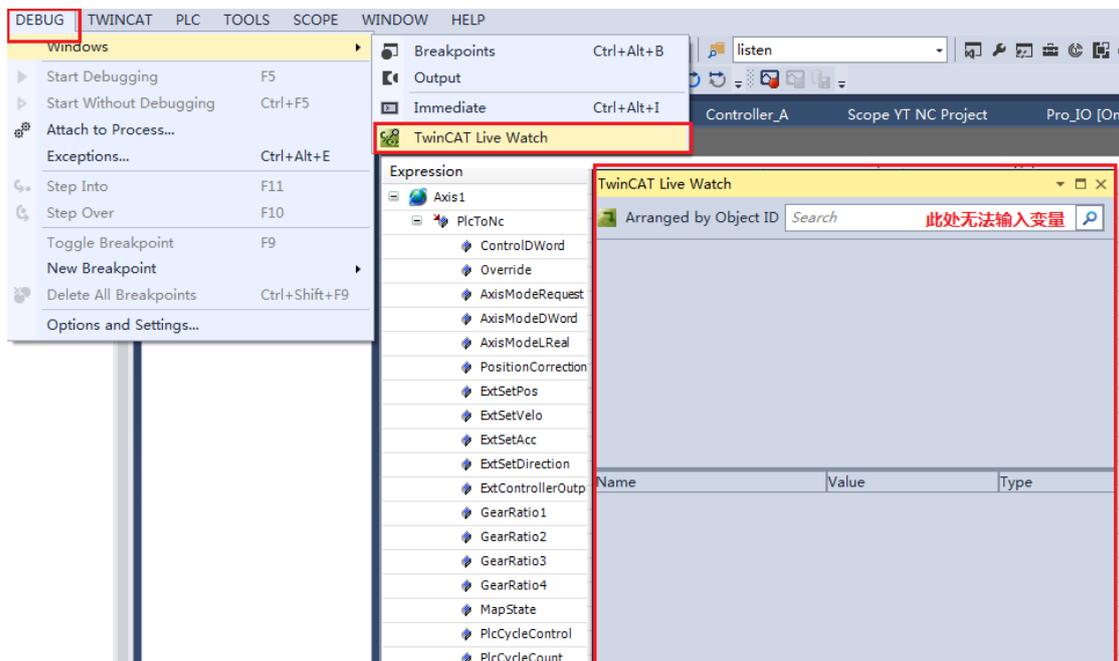
通过程序注释还可以有很多特殊功能，比如取消编译等，Infosys 上有详细说明。

## 3.3. 诊断和调试功能

### 3.3.1. 搜索和替换按钮

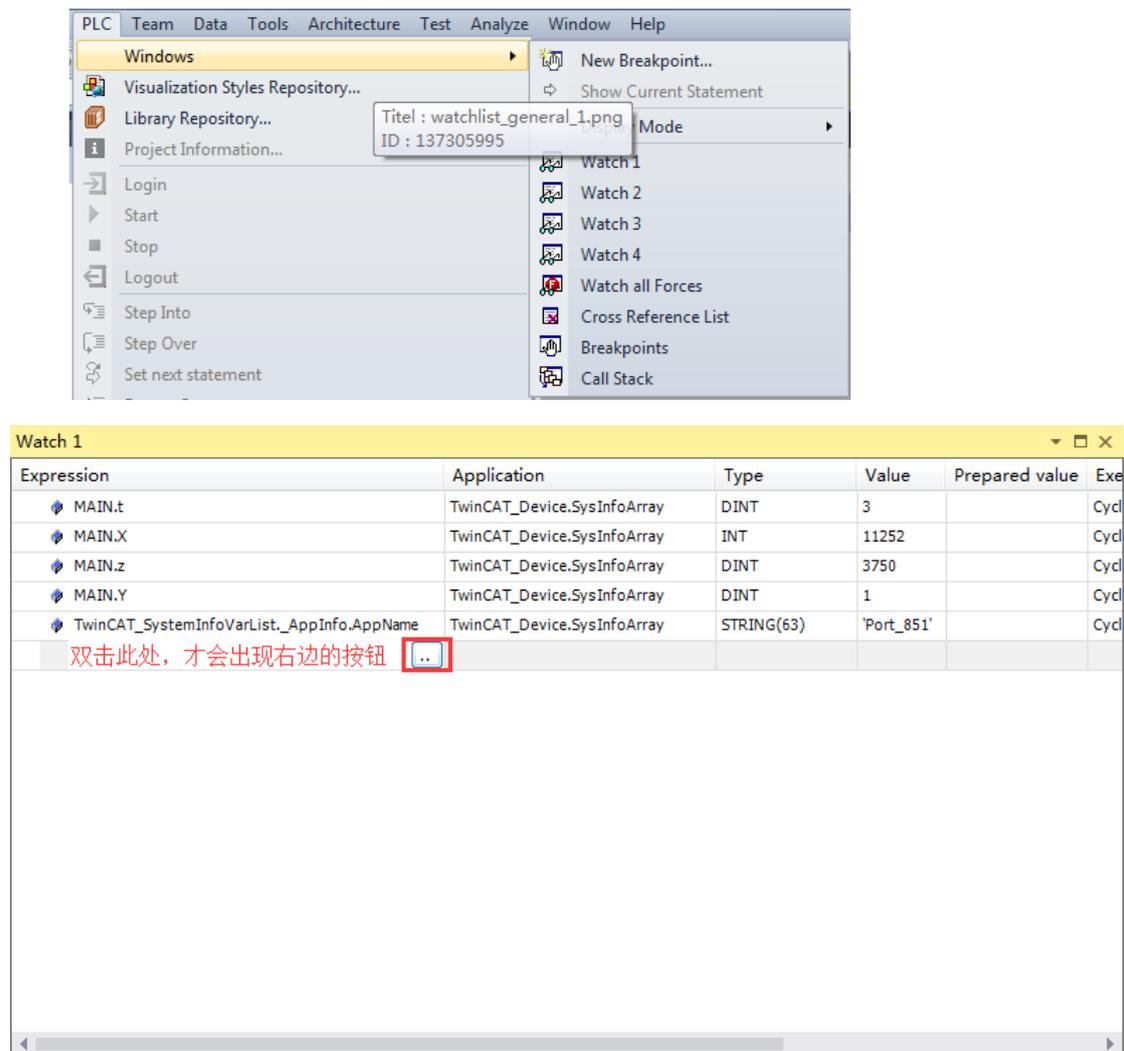


### 3.3.2. TwinCAT Live Watch 怎么用





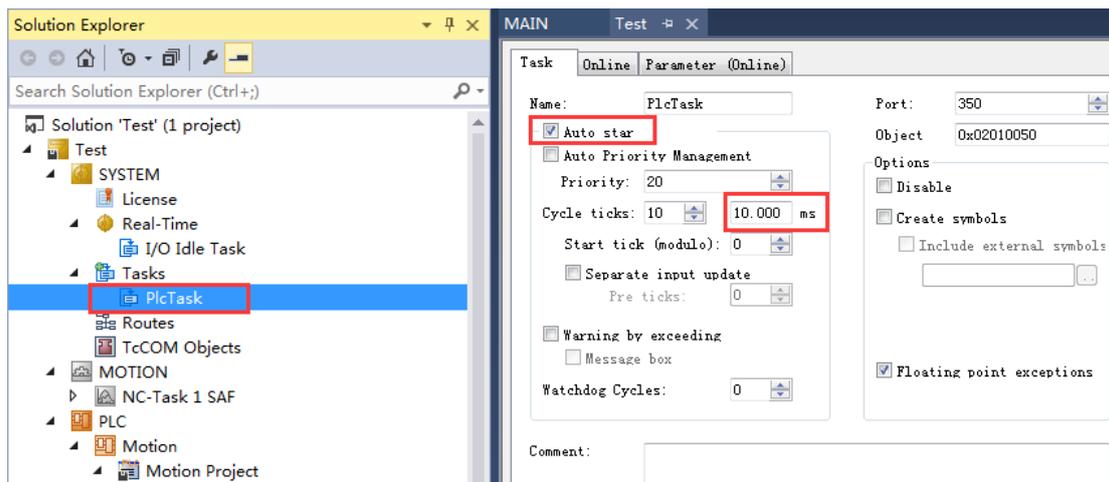
### 3.3.5. 独立于程序的 Watch List



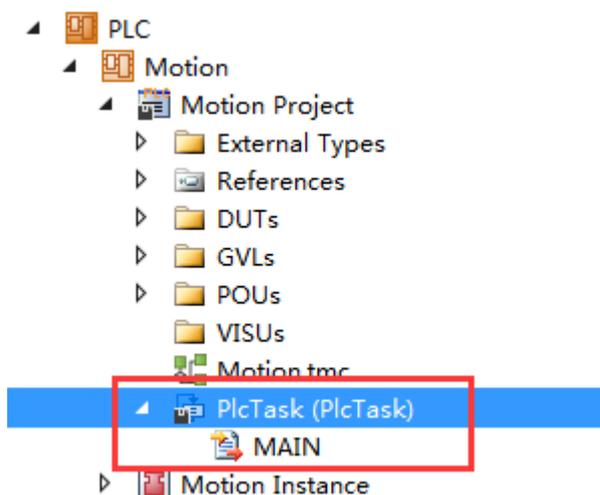
### 3.3.6. Clean 之后不能再 Online Change.

## 3.4. 任务和程序

在新建 PLC 项目时，系统会自动创建一个任务：

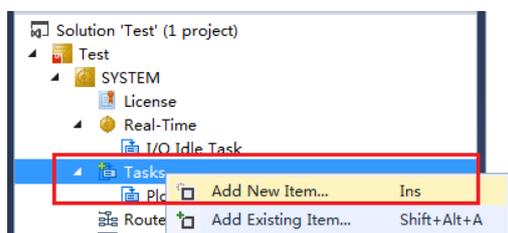


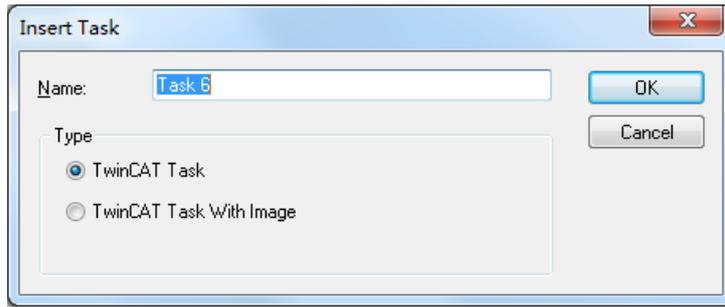
而自动新建的 Main 程序就被这个任务自动调用：



那么如何新建多个任务，并分别调用不同的 POU 呢？

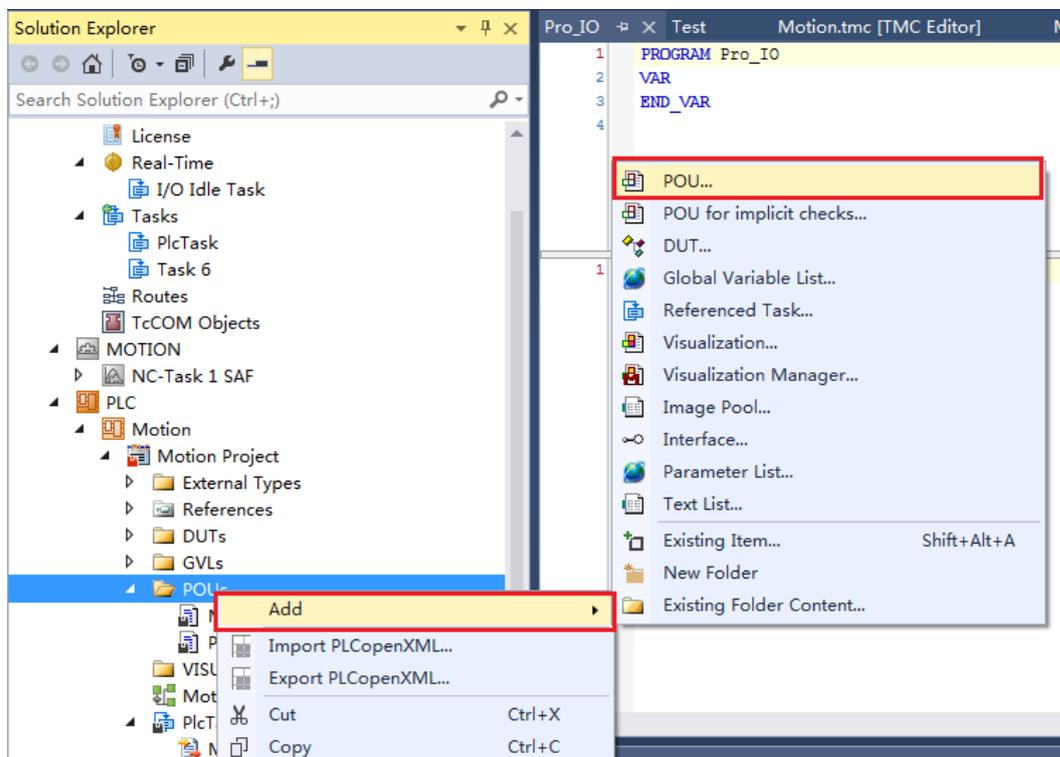
### 3.4.1. 新建任务

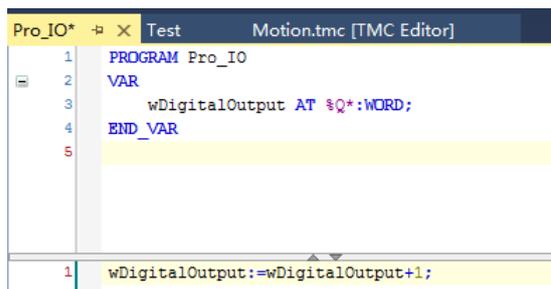
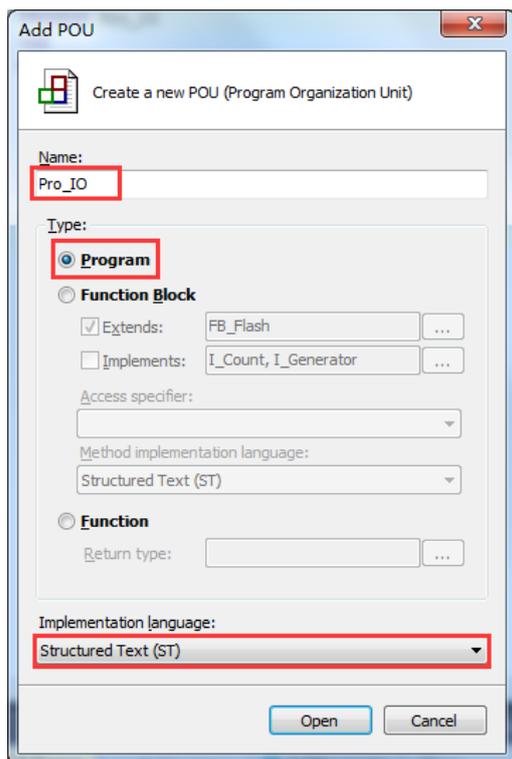




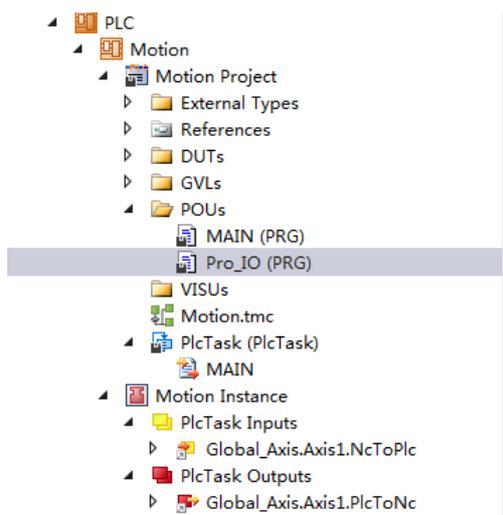
### 3.4.2. 把程序指定到任务

先增加一个程序：





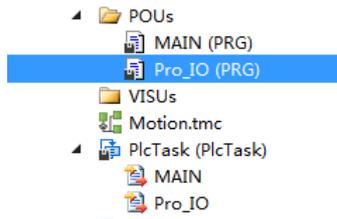
编译，发现 TMC 中没有 Pro\_IO 的接口变量。



这时，需要把 Pro\_IO 指定到某个任务。

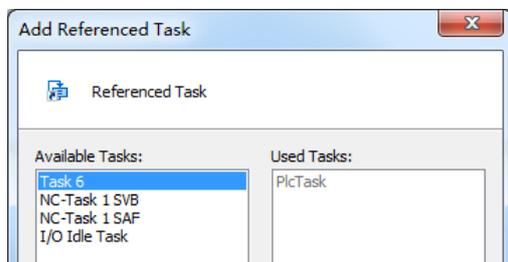
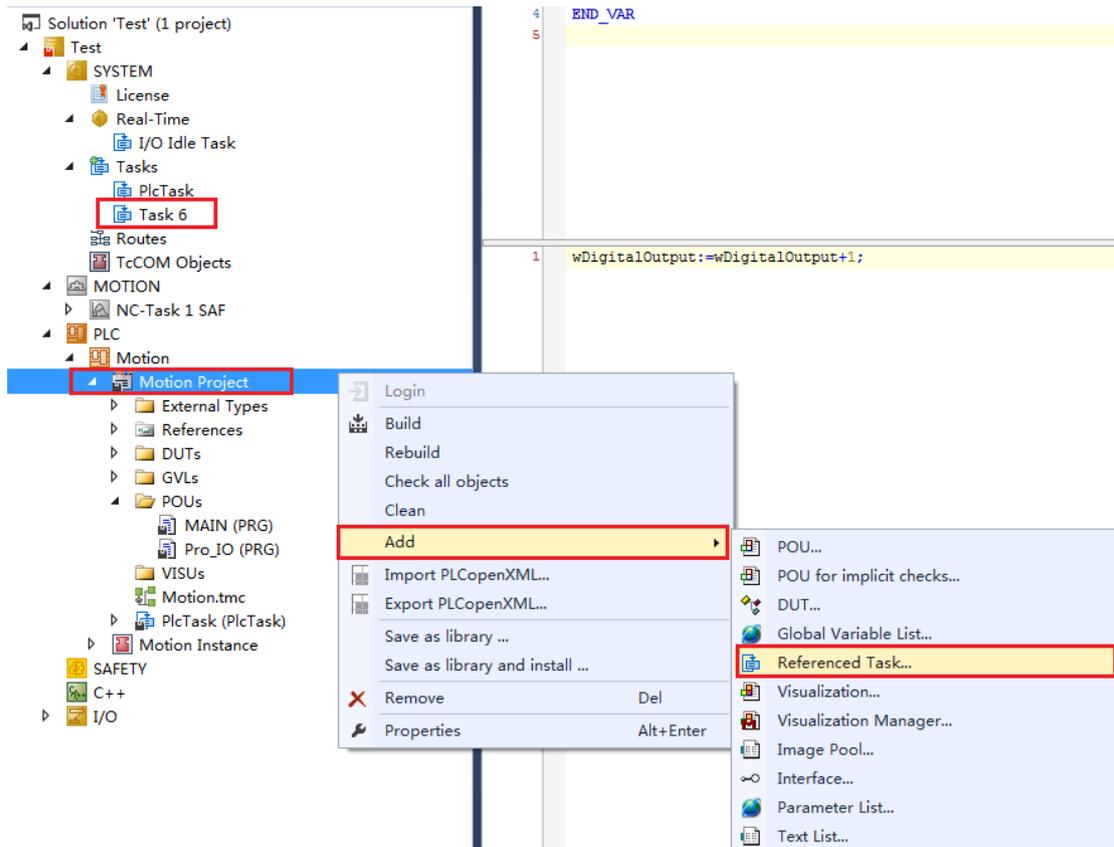
如是指定到原先的 PlcTask，与 Main 同样周期运行，最简单的方法是拖动：

选中 POUs 下的 Pro\_IO，直接拖放到 PlcTask 下面。



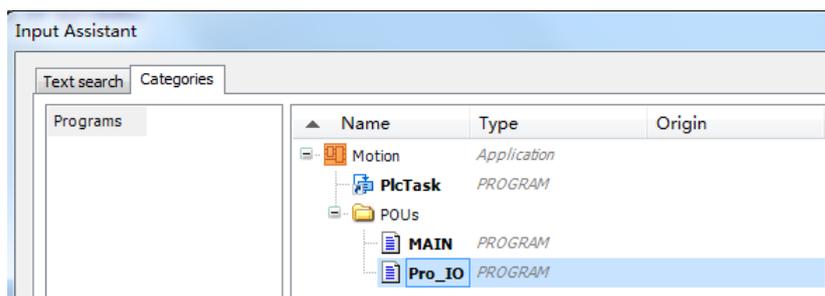
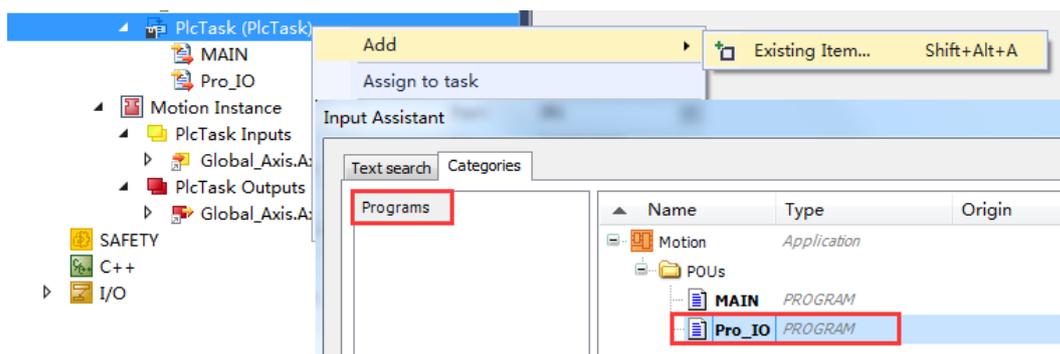
如是指定到其它 Task，则按以下操作：

先引用已新建的 Task，比如例中的 Task6，如图所示：

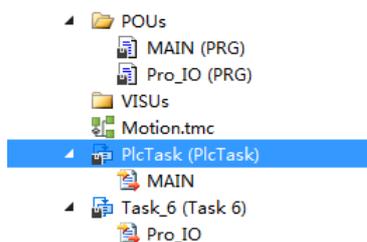


然后 PLC 项目 Motion 中就会出现引用的 Task 6。把程序 Pro\_IO 指定给 Task6 运行：简便方法是选中 POUs 下的 Pro\_IO，直接拖放到 Task 6 下面。

完整的方法是：

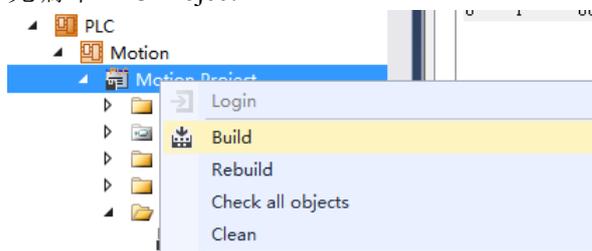


结果：

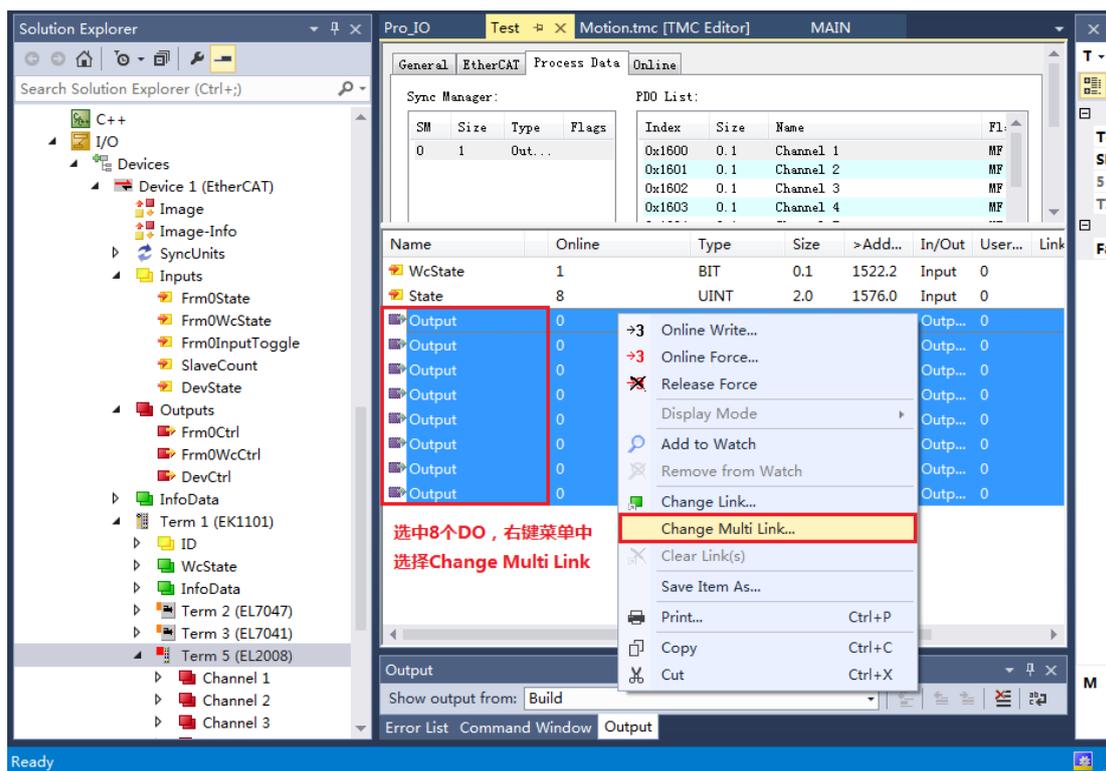


### 3.4.3. 编译和试运行

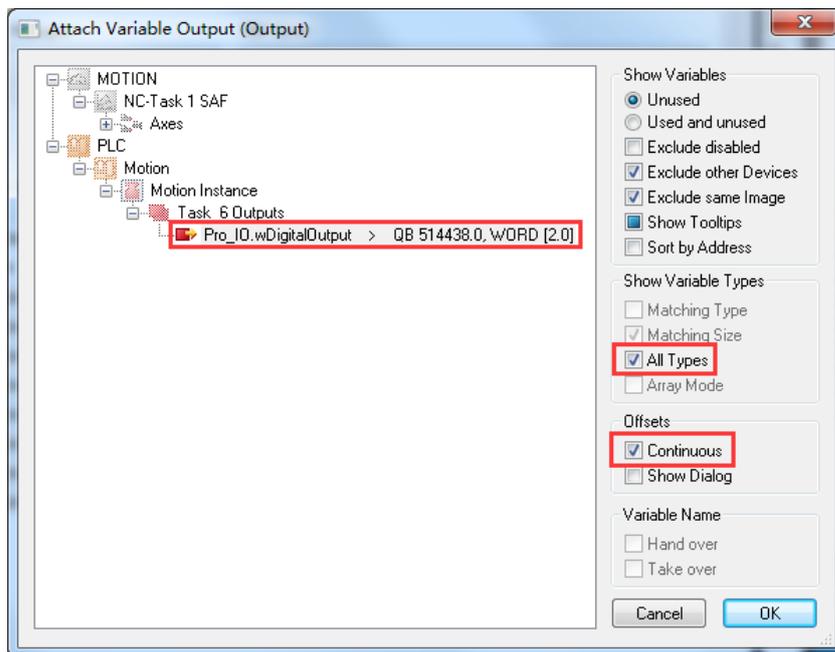
先编译 PLC Project



再把 8 个 DO 通道链接到 PLC 的输出变量：



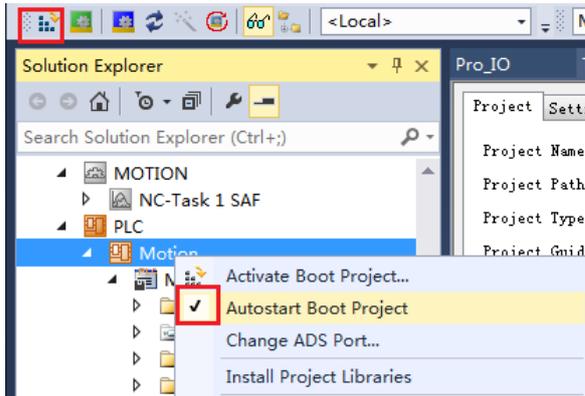
过滤器是勾选“**All Type**”和“**Continous**”，选中 PLC 变量。



就可以看到 8 个 DO 都链接上了：

Name	Online	Type	Size	>Add...	In/Out	Use
WcState	1	BIT	0.1	1522.2	Input	0
State	8	UINT	2.0	1576.0	Input	0
Output	X 0	BIT	0.1	67.0	Outp...	0
Output	X 0	BIT	0.1	67.1	Outp...	0
Output	X 0	BIT	0.1	67.2	Outp...	0
Output	X 0	BIT	0.1	67.3	Outp...	0
Output . Channel 3 . Term 5 (EL2008) . Device 1 (EtherCAT) . Devices						
Output	X 0	BIT	0.1	67.5	Outp...	0
Output	X 0	BIT	0.1	67.6	Outp...	0
Output	X 0	BIT	0.1	67.7	Outp...	0

激活配置，运行程序：



上图中勾选了“AutoStart Boot Project”，程序就会自动运行。

### 3.4.4. 为 Task 指定 CPU、优先级、周期等等。

CPU	RT-CPU	Bas...	CPU Limit	Latency Warning
0 (Wi...	<input type="checkbox"/>			
1 (Wi...	<input type="checkbox"/>			
2 (Ot...	<input checked="" type="checkbox"/>	De...	100 %	(none)
3 (Ot...	<input checked="" type="checkbox"/>	1...	100 %	(none)

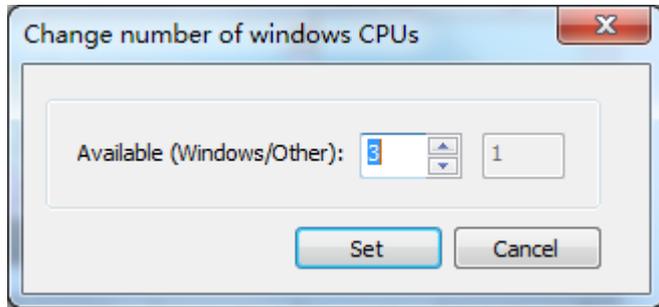
  

Object	RT-CPU	Base Ti...	Cycle Ti...	Cycle Tic...	Priority
Task 6	CPU 2	1 ms	20 ms	20	1
NC-Task...	CPU 3	1 ms	2 ms	2	4
I/O Idle ...	Defau...	1 ms	1 ms	1	11
PlcTask	CPU 3	1 ms	10 ms	10	20
PlcAuxT...	Defau...	1 ms	(none)	0	50

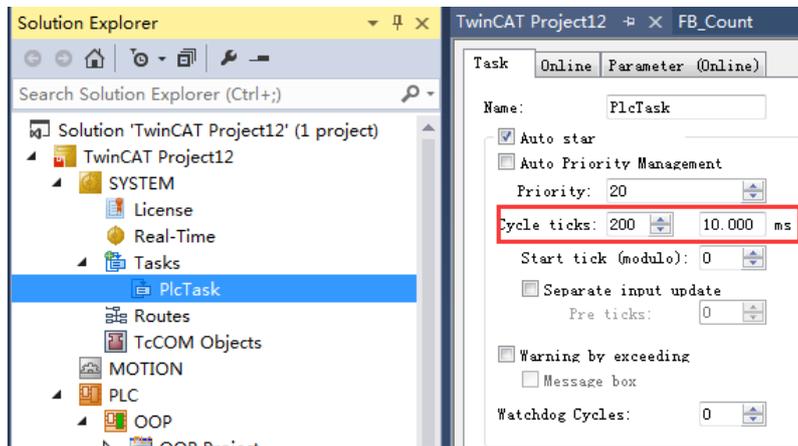
- 指定 CPU

图中的设置表示，控制器 CPU 一共有 4 个核。CPU0 和 1 给 Windows 使用，CPU2 和 3 给 TwinCAT 独占使用。因为是独占，所以 CPU 利用率允许到 100%。

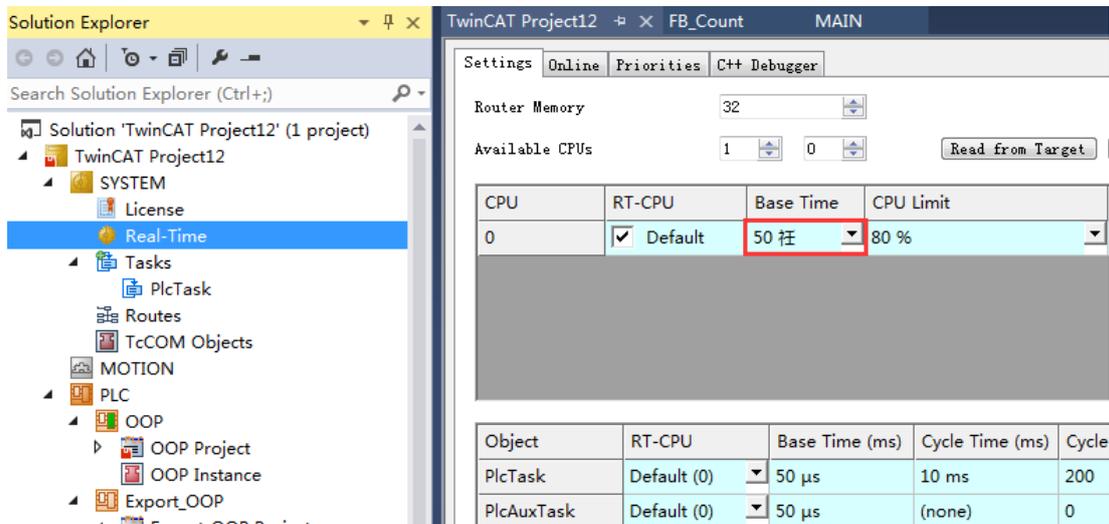
点击上图中的 Set On Target，可以设置操作系统使用的 CPU 和其它（TwinCAT）的 CPU 划分：



### ● 设置任务周期



默认的 Time Base 为 1ms，可以满足绝大多数的应用。如果要求 PLC 或者 NC 的任务周期低于 1ms，就需要修改 Time Base 的设置。



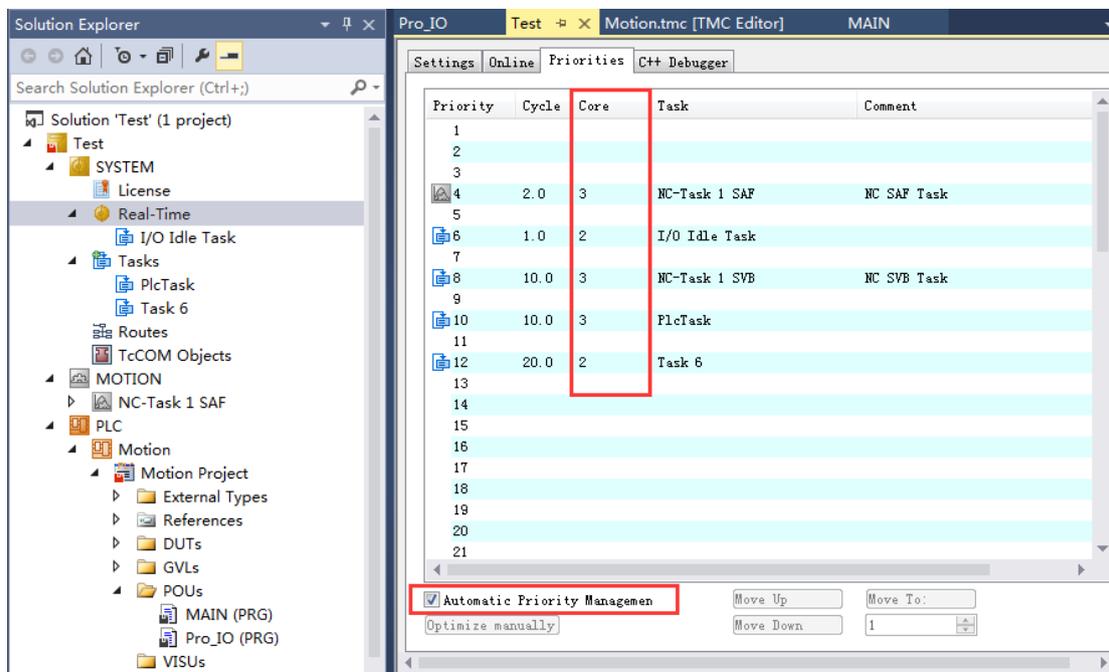
Time Base 可选的最小值为 50us, 最大值为 1ms。所有任务周期必须是 Time Base 的整数倍。

同样的程序, 同样的任务周期, Time Base 越小, 由于线程切换频繁, 所以 CPU 消耗越大。因此, 在满足任务周期设定的前提下, Time Base 应尽可能设置得大, 比如, 等于最小任务周期。

当设置 Time Base 小于 1ms 的任务时, 应考虑硬件 CPU 的运算能力。根据经验, CX2020 及以上控制器、工控机可以执行 50us 但程序量少的 PLC 任务, 而 CX80x0、CX50x0、CX90x0 系列 EPC 一旦 Time Base 设置为 50us, 即使代码行为空, TwinCAT PLC 一旦运行起来, CPU 利用率也会直接飙高甚至崩溃。

- 设置任务优先级

由于 TwinCAT3 手动增加的任务优先级不再受限于 PLC, 所以必须谨慎考虑优先级排列。默认手动增加任务的优先级总是高于 NC 任务, 但用作 PLC 任务时可能周期远大于 NC 周期。这在触发 IO 总线通讯时可能会产生问题, 最好是使用“自动优先级排列”, 如图所示:



说明: 在 TwinCAT 3 的 Real Time|Priorities 优先级排列时, 有了“Core”这一列显示使用 CPU 的哪个核。但总的优先级仍然是 1-61, 与 TwinCAT 2 相同。

### 3.4.5. Task with image:

类似 TC2 的 Additional Task。仅 TwinCAT IO 功能。对于 TC3 仅有 IO 级别的, TC1100-0090(第三方工控机), 400EURO。

对于 TC1100-0040(CX5130 及以下), 75Euro。CX5140 和 CX2020 则 100Euro。

### 3.4.6. IO at task beginning

从 POU 的注释里，通过特殊的注释，设置 IO at task beginning \NO CHECK 等功能。  
其它见帮助文件

### 3.4.7. Task 特别提示

同一套 PLC 程序的多个任务必须指定到同一个 CPU 核，否则可能引起优先级错误。

Sample: A POU which will be called by an task with a low priority can be assigned to an independent core. In this case exists the possibility, that the task with the lower priority finishes the execution earlier than the tasks with the high priority.

## 3.5. 隐含的变量和函数

### 3.5.1. TwinCAT\_SystemInfoVarList

在 TC2 中有一些全局变量比如 First Cycle 标记等，是需要引用 TcSystem.lib 等库才能使用的。在 TC3 中不需要引用任何库，可以直接从全局变量 TwinCAT\_SystemInfoVarList 结构体中获得。

该变量包括 1 个 \_AppInfo 结构体和 1 个 \_TaskInfo 结构型数组。

\_AppInfo 结构体包含以下信息：

AppName  
AppTimeStamp  
OnlineChangeCnt : 在线修改次数  
.....

\_TaskInfo 结构型数组元素包含以下信息：

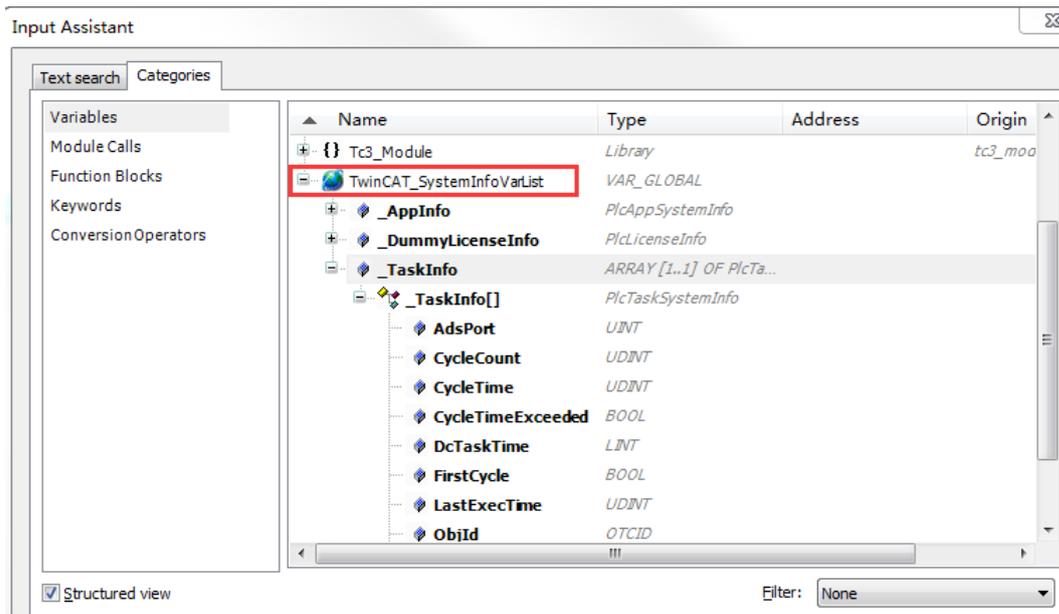
FirstCycle : 首周期标志  
LastExcuteTime : 上周期任务执行时间  
CycleTimeExceeded: 任务未成设定时间内完成  
.....

这两个结构体的完整的元素描述，可以参考在线帮助，或者 Infosystem.

注意，为了读取正确的任务编号的信息，首先要获取当前程序代码所在的任务编号，需要用到一下函数：

GetCurTaskIndex()

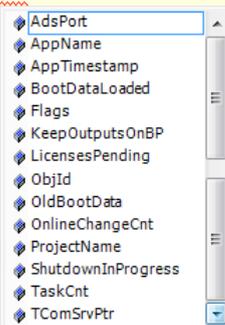
## ● 调用图示



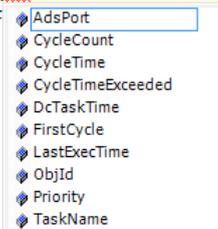
```
Y:=GETCURTASKINDEXEX(); 获取任务编号
TwinCAT_SystemInfoVarList._TaskInfo[Y].CycleTime;
TwinCAT_SystemInfoVarList._AppInfo.;
```

隐含全局变量

**TwinCAT\_SystemInfoVarList**  
的子元素



```
Y:=GETCURTASKINDEXEX();
TwinCAT_SystemInfoVarList._TaskInfo[Y].;
TwinCAT_SystemInfoVarList._AppInfo.Task
```



注意这两个 AdsPort 是不一样的。TaskInfo 里的 AdsPort 是针对任务的，AppInfo 里的 AdsPort 才是针对 PLC 变量的。与 HMI 通讯的，通常是指后者 \_AppInfo 里的端口，比如默认的 851。

```
TwinCAT_SystemInfoVarList._AppInfo.AppTimestamp DT#2015-8-6-14:6.5 ;RETURN
```

此处指PLC开始运行的时间，而不是当前时

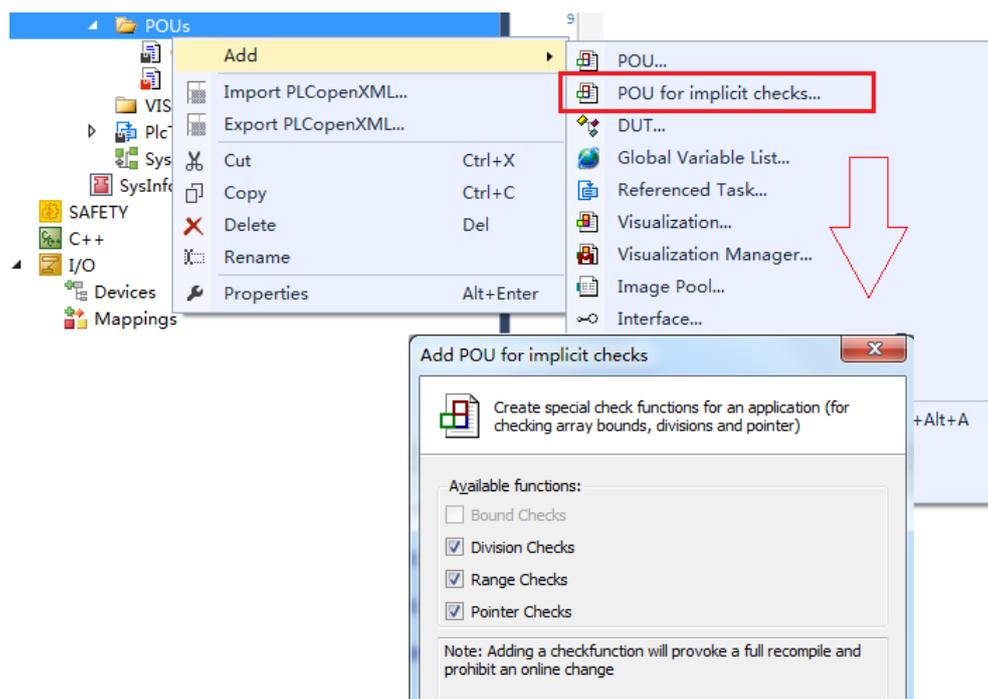
### 3.5.2. 除零等校验

在 TC2 中要校验除零、数组下标超限、指针错误等，必须引用 SysFunction.lib，或者自己建立一个 Check Bounds 类似的函数。在 TC3 中，默认已打开这些校验功能：

第 85 页 共 430 页

2015-9-25

本文纯属个人经验，非 Beckhoff 公司正式发行，如有疑问，请直接联系作者：BCGZ Lizzy Chen



如果要关闭，可以在图中取消勾选。

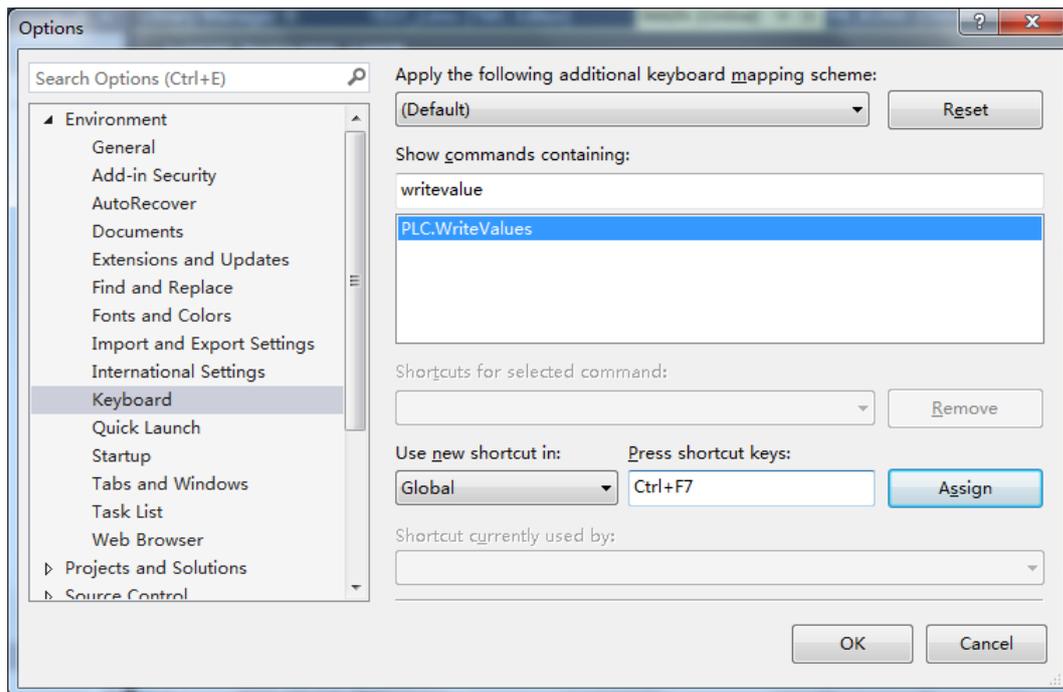
### 3.5.3. 隐含的函数

system:=TIME();获取当前时间。

## 3.6. 编程环境的设置

### 3.6.1. TwinCAT 快捷键

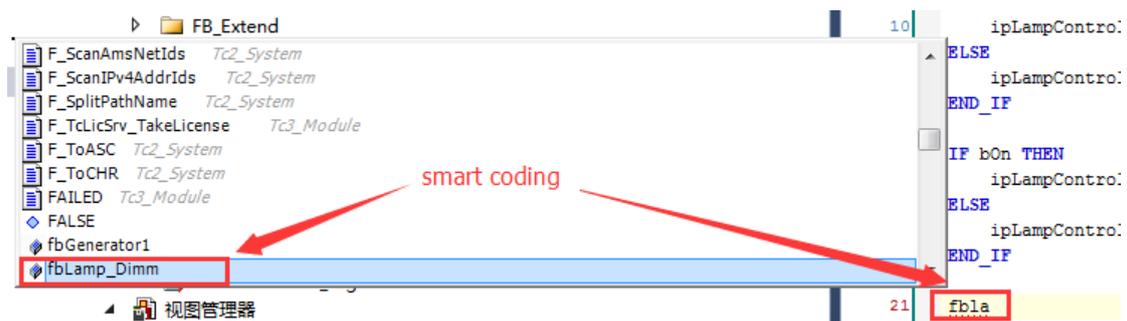
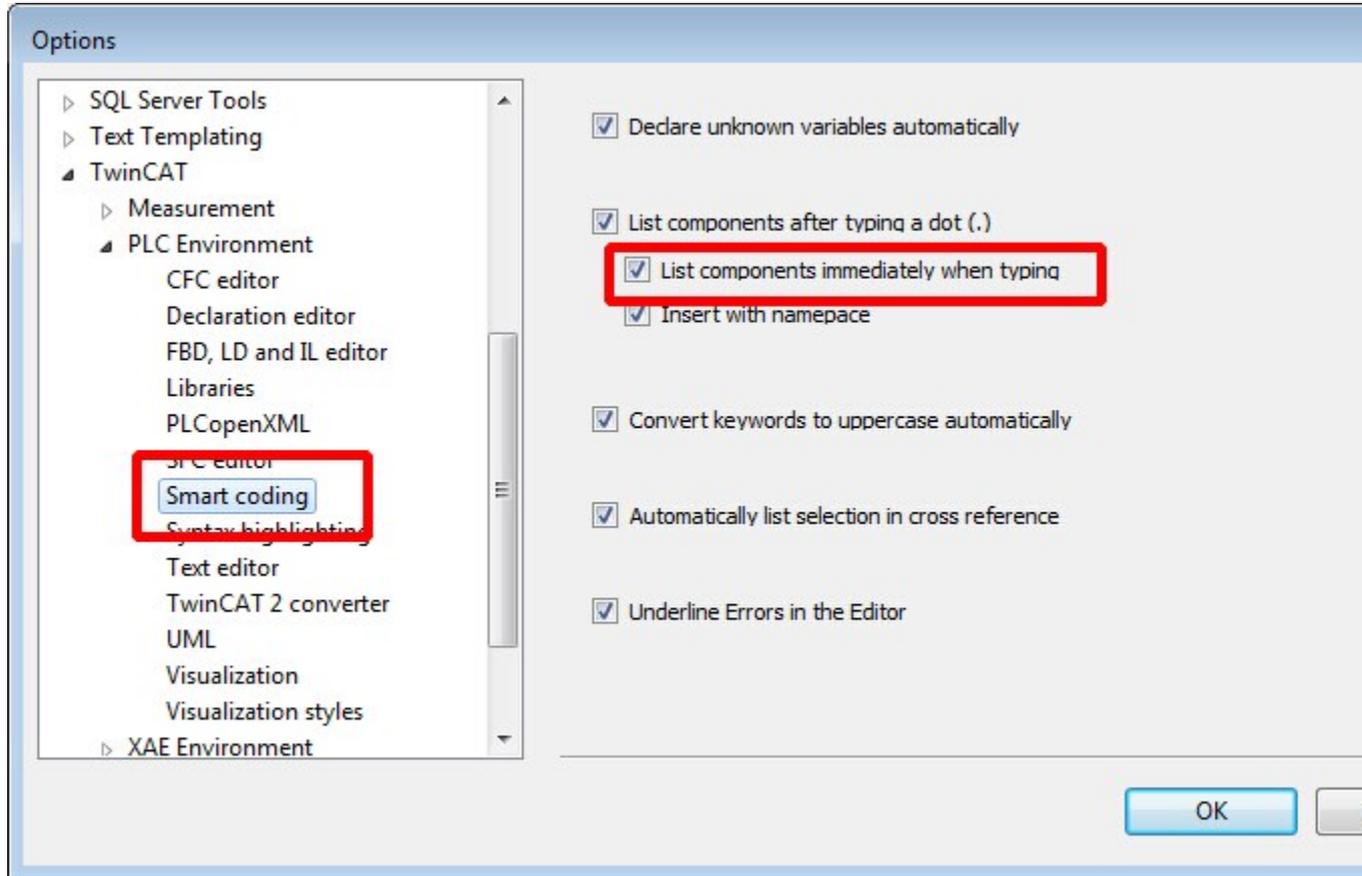
Tool | Option



默认的快捷键 F2 为 Input Assistant

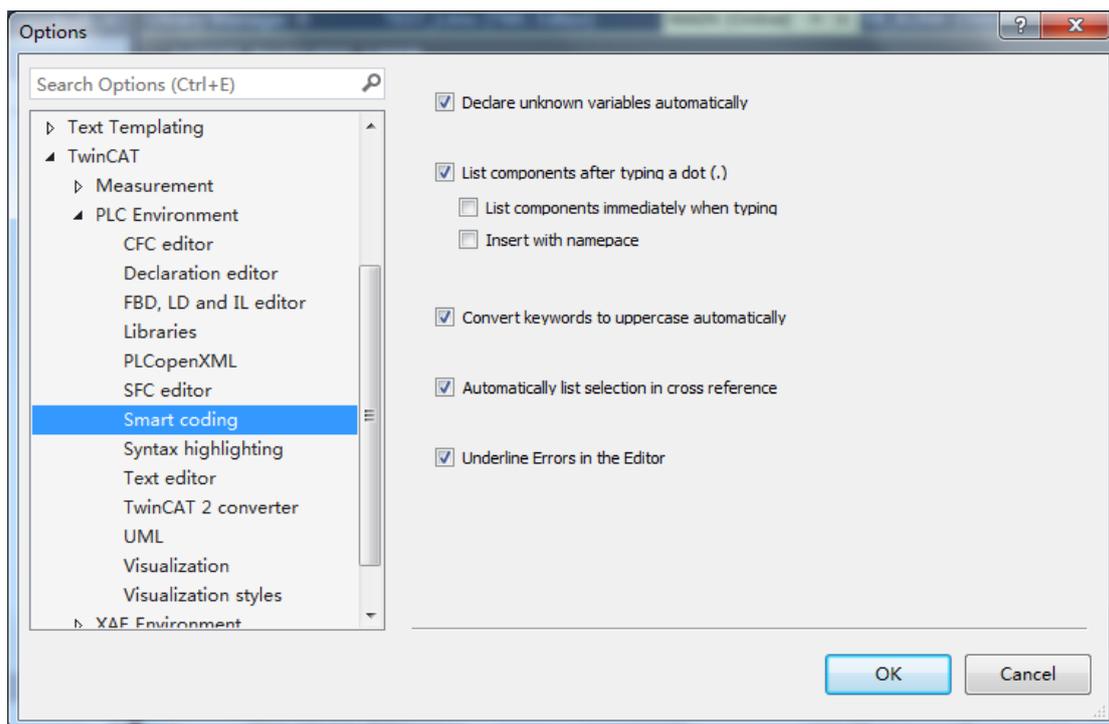
### 3.6.2. Smart Coding

变量名输入提示。



如图所示，输入首字母就有变量提示了。

### 3.6.3. 编程环境的其它设置



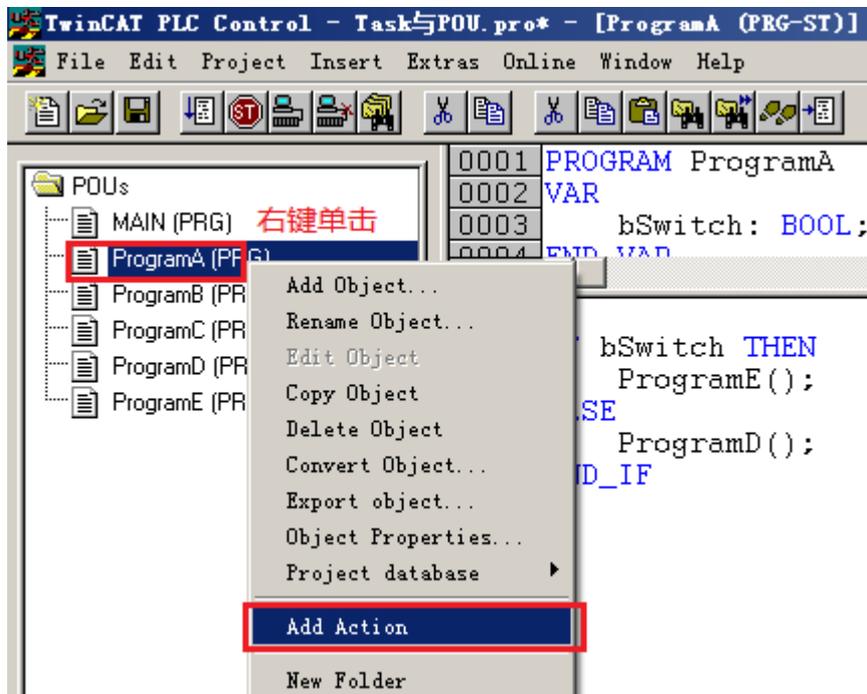
## 3.7.兼容 TC2 的功能

### 3.7.1. 多语言混合编程（Action）

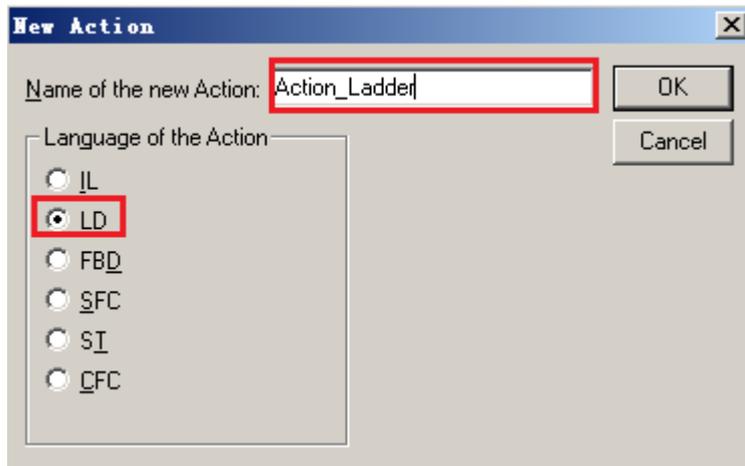
一个程序（PRG）中的代码可能包含逻辑运算，也包含数学运算。前者适合用梯形图，而后者用结构文本写起来就比较方便。TwinCAT PLC Control 中提供向 POU 中添加 Action 子程序的功能。在 POU 的 Action 中，可以直接使用所在程序的所有变量，可以选用跟所在程序相同的语言，也可以另外选一种语言，这就是多语言混合编程。

使用 Action 的操作方法如下：

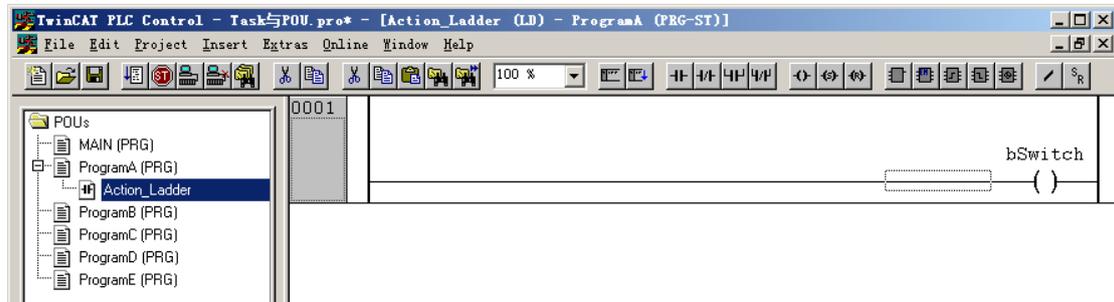
第 1 步：添加 Action



填写 Action 的名称

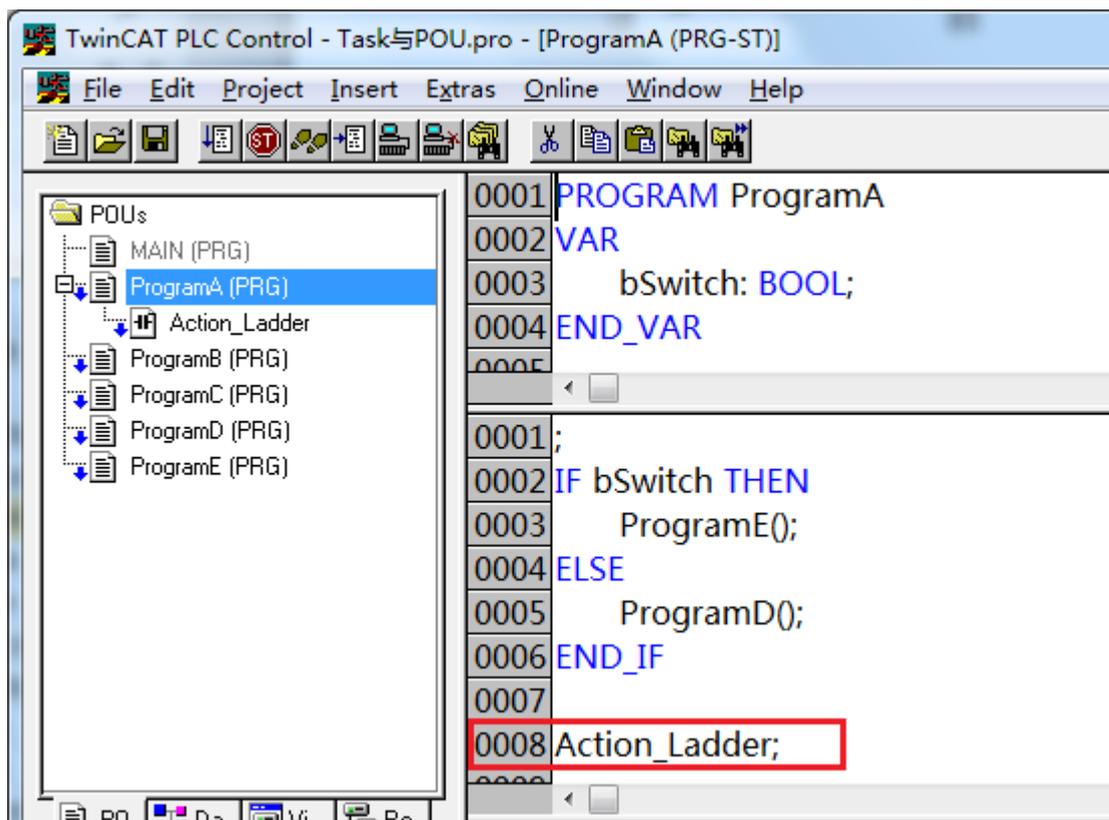


第 2 步：在 Action 中编写代码



第 3 步：在 PRG 程序中调用 Action。

和调用其它程序一样，调用 Action 也只需要直接写上其名称，此处应写“Action\_Ladder”，等效于“ProgramA.Action\_Ladder”。



提示:

- 1, 主程序用结构文本 ST 编程, 调用子程序会比较简洁方便。
- 2, Action 不仅用来换一种编程语言, 还可以把实现不同功能的代码放在不同的 Action 中, 增加主程序的可读性。

### 3.7.2. 可供使用的操作符、函数和功能块

PLC 程序可以使用 IEC61131-3 标准规定的函数和操作符, 以及新建的 PLC 程序时默认引用的库文件 Standard.lib 中的 Function 和 Function Block。

这些默认可以使用的操作符、Function 和 Function Block 包括:

用于结构文本 ST 的操作符	操作符 AWL	含义
:=	ST var1	赋值, 保存实际结果到 var1
RETURN	RET	离开 POU 并返回到调用程序
AND	AND	按位与 AND
OR	OR	按位或 OR
XOR	XOR	按位异或 XOR
NOT	NOT	按位取反 NOT

+	ADD	加
-	SUB	减
*	MUL	乘
/	DIV	除
>	GT	大于
>=	GE	大于或等于
=	EQ	等于
<	LT	小于
<>	NE	不等
<=	LE	小于或等于
in1 MOD in2	MOD	“取模”运算
INDEXOF(in)	INDEXOF	POU in1 的内部索引; [INT]
SIZEOF(in)	SIZEOF	数据类型 in 所需要的字节数
SHL(in,K)	SHL	操作数 in 按位左移 K 次
SHR(in,K)	SHR	操作数 in 按位右移 K 次
ROL(in,K)	ROL	操作数 in 按位循环左移 K 次
ROR(in,K)	ROR	操作数 in 按位循环右移 K 次
SEL(G,in0,in1)	SEL	在 2 个操作数 in0 (G 是 FALSE) 和 in1 (G 是 TRUE) 之间的二进制选择
MAX(in0,in1)	MAX	返回两个值之间的大者
MIN(in0,in1)	MIN	返回两个值 in0 和 in1 之间的小者
LIMIT(Min,in,Max)	LIMIT	限制值范围 (在超过该范围时, in 返回到 MIN 或 MAX)
MUX(K, in0,.. in_n)	MUX	从数值组中(in0 到 In_n)选出第 K 个值
ADR(in)	ADR	取变量的地址
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 的指数
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	带 expt 的操作数 in 的指数
LEN(in)	LEN	操作数 in 的串长度
LEFT(str, size)	LEFT	串 standard.lib 给定的左起始串
RIGHT(str, size)	RIGHT	串 standard.lib 给定长度的右起始串
MID(str, len, pos)	MID	串 str 中给定长度的部分串
CONCAT(str1, str2)	CONCAT	两个子串 standard.lib 的合并
INSERT(str1, str2, pos)	INSERT	在串 standard.lib 位置插入串 str1
DELETE(str1, len, pos)	DELETE	删去部分串 (长度为 len), 于 str1 的 pos standard.lib 位置开始
REPLACE(str1, str2, len, pos)	REPLACE	由 str2 替换部分的串 (长度为 len), 在 str1 的 pos 位置开始
FIND(str1, str2)	FIND	在 str1 中搜索串 str2 部分
SR	SR	以稳态功能块, 复位优先
RS	RS	双稳态功能块, 置位优先
SEMA	SEMA	FB:Semaphor 软件(可中断的)
R_TRIG	R_TRIG	FB:检测上升沿
F_TRIG	F_TRIG	FB:检测下降沿
CTU	CTU	FB:加计数
CTD	CTD	FB:减计数
CTUD	CTUD	FB:加/减计数
TP	TP	FB:触发器
TON	TON	FB:接通延迟定时器
TOF	TOF	FB:断开延迟定时器

以及变量类型转换函数

BOOL_TO_<type>(in)	BOOL_TO_<type>	布尔操作数的类型转换
<type>_TO_BOOL(in)	<type>_TO_BOOL	转换为布尔的类型转换
INT_TO_<type>(in)	INT_TO_<type>	INT 操作数, 转换类型为另一个基本类型
REAL_TO_<type>(in)	REAL_TO_<type>	REAL 操作数, 转换类型为另一个基本类型
LREAL_TO_<type>(in)	LREAL_TO_<type>	LREAL 操作数, 转换类型为另一个

		基本类型
TIME_TO_<type>(in)	TIME_TO_<type>	TIME 操作数, 转换类型为另一个基本类型
TOD_TO_<type>(in)	TOD_TO_<type>	TOD 操作数, 转换类型为另一个基本类型
DATE_TO_<type>(in)	DATE_TO_<type>	DATE 操作数, 转换类型为另一个基本类型
DT_TO_<type>(in)	DT_TO_<type>	DT 操作数, 转换类型为另一个基本类型
STRING_TO_<type>(in)	STRING_TO_<type>	STRING 操作数, 转换类型为另一个基本类型

此外, 倍福公司提供了许多针对不同应用的库文件。

TcEtherCAT.lib	访问 EtherCAT 主站和从站设备
TcMath.lib	数学运算函数
TcSystem.lib	ADS 通讯, 文件处理, 操作系统接口, 定义系统结构
TcSUPSTc.lib	使用 1 秒 UPS 需要的函数和功能块.
TcUtilities.lib	各种有用的服务, 比如 PLC 启停或者 NT 关机
TcMC2.lib	PLC Open 提供的标准运动控制功能块 Version2.
TcMC2_Camming.lib	基于 MC2 的凸轮
TcMC2_Drive.lib	直接通过 MC2 的 Axis_Ref 访问 SoE 驱动的硬件
TcMC2_FlyingSaw.lib	基于 MC2 的飞锯 (NC Flying Saw)
TcController Toolbox.lib	控制工具箱, 包括 PID、斜坡发生器、各种滤波等
TcTemCtrl.lib	带自整定的温控功能块
TcNcFifoAxis.lib	控制 NC FIFO 轴的各种专用功能块
TcMC2_XFC.lib	XFC 模块用于运动控制的专用功能块

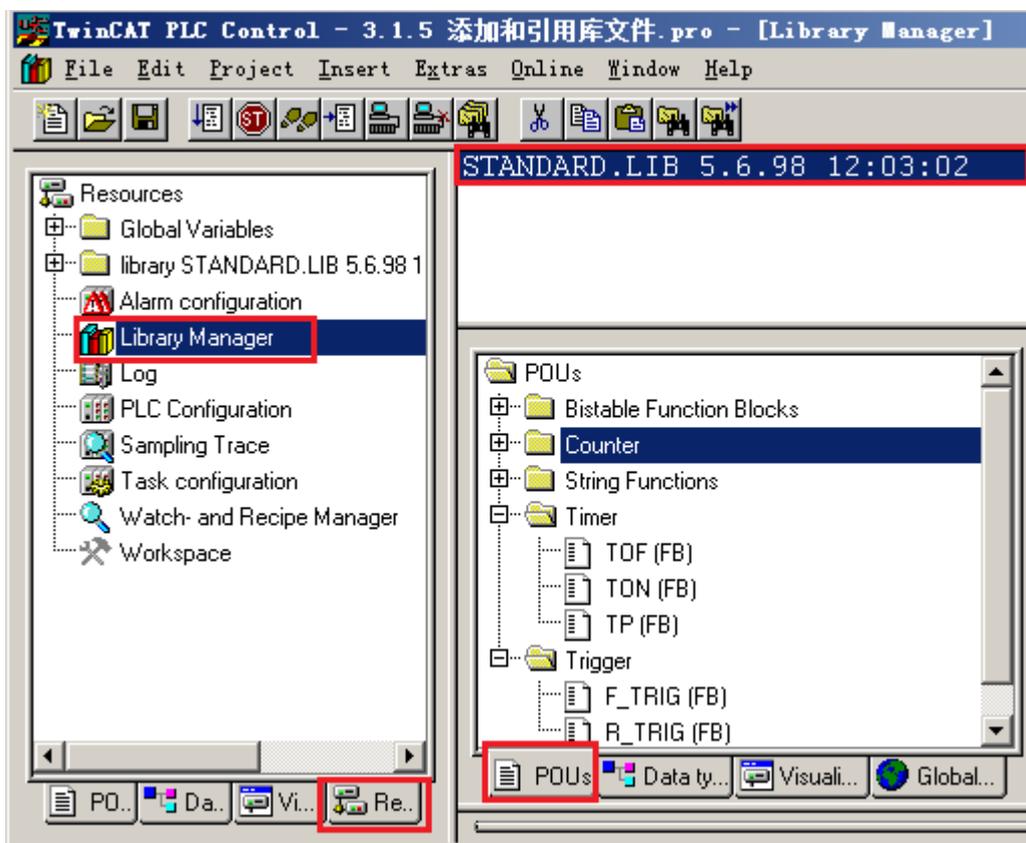
通讯类:

ModbusRTU.lib	通过总线端子或者 IPC 上的 Com 口进行 Modbus RTU 串行通讯的专用功能块
COMlibV2.lib	通过总线端子或者 IPC 上的 Com 口进行自由协议串行通讯的专用功能块
TcpIP.lib	通过总线端子或者 IPC 上的以及网口进行自由协议 Ethernet 通讯的专用功能块
TcRFID.lib	控制 RFID 读卡器的专用功能块

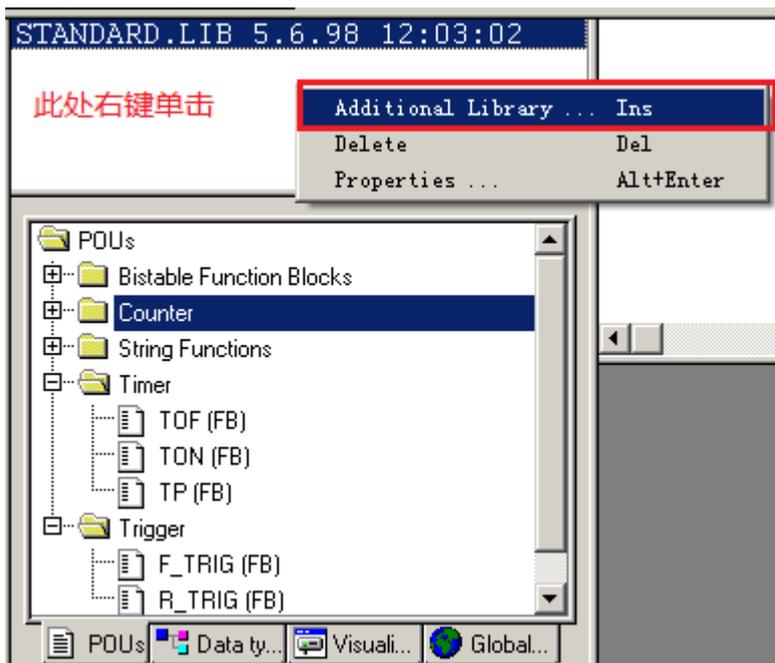
要使用其中的功能块和函数, 必须先的项目中引用对应的库。具体操作如下:

## 第 1 步：认识 Library Manager

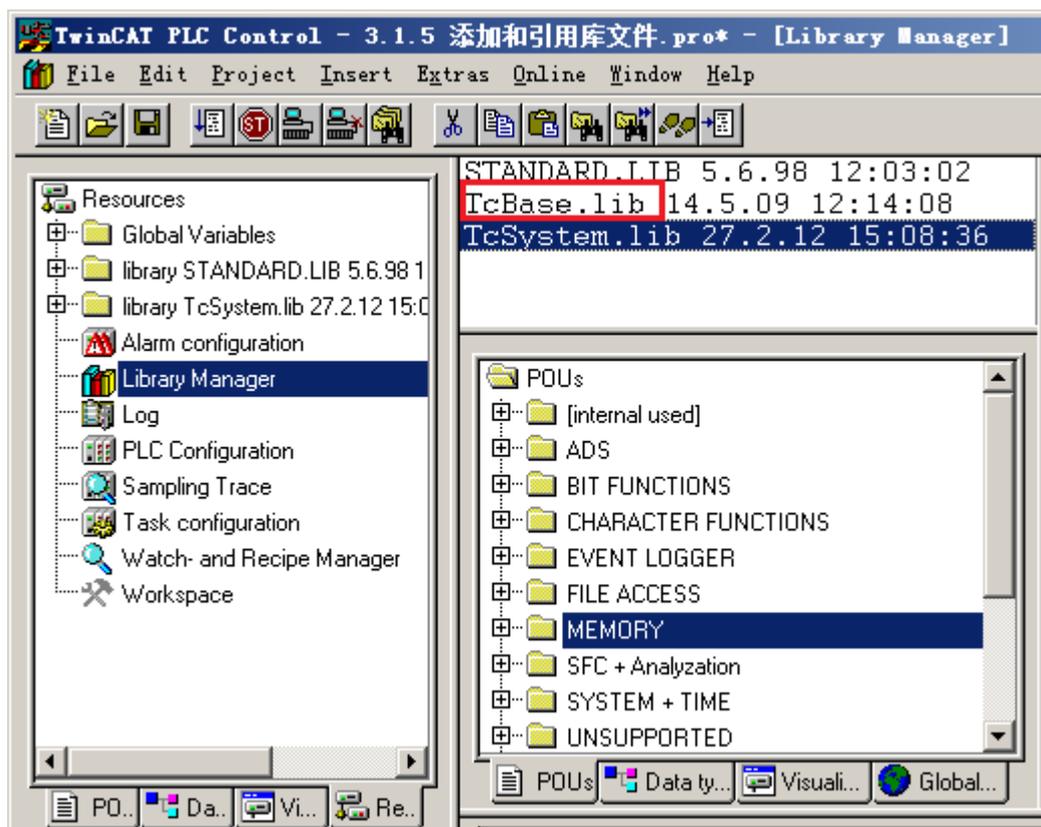
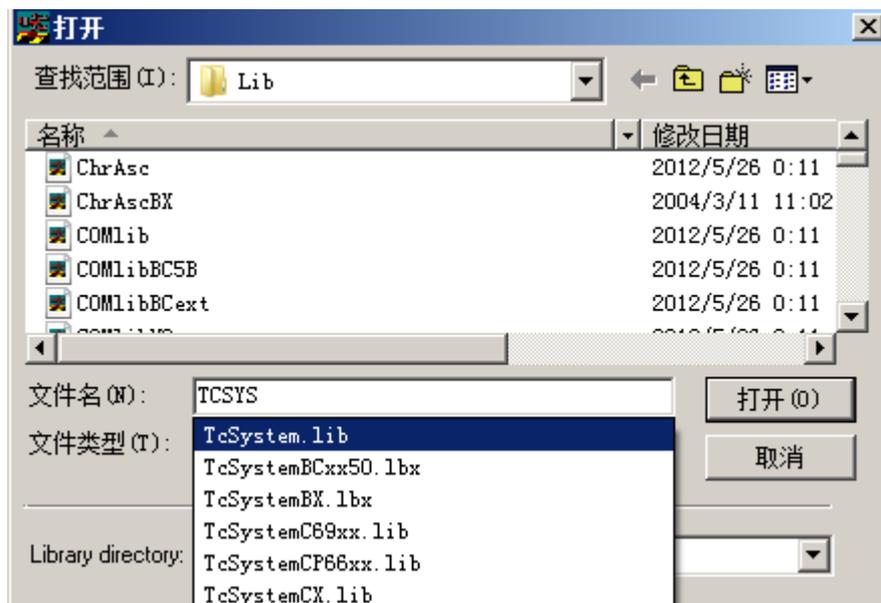
在 Library Manager 中可以查看库中所有的 FB、FC、



## 第 2 步：添加库文件



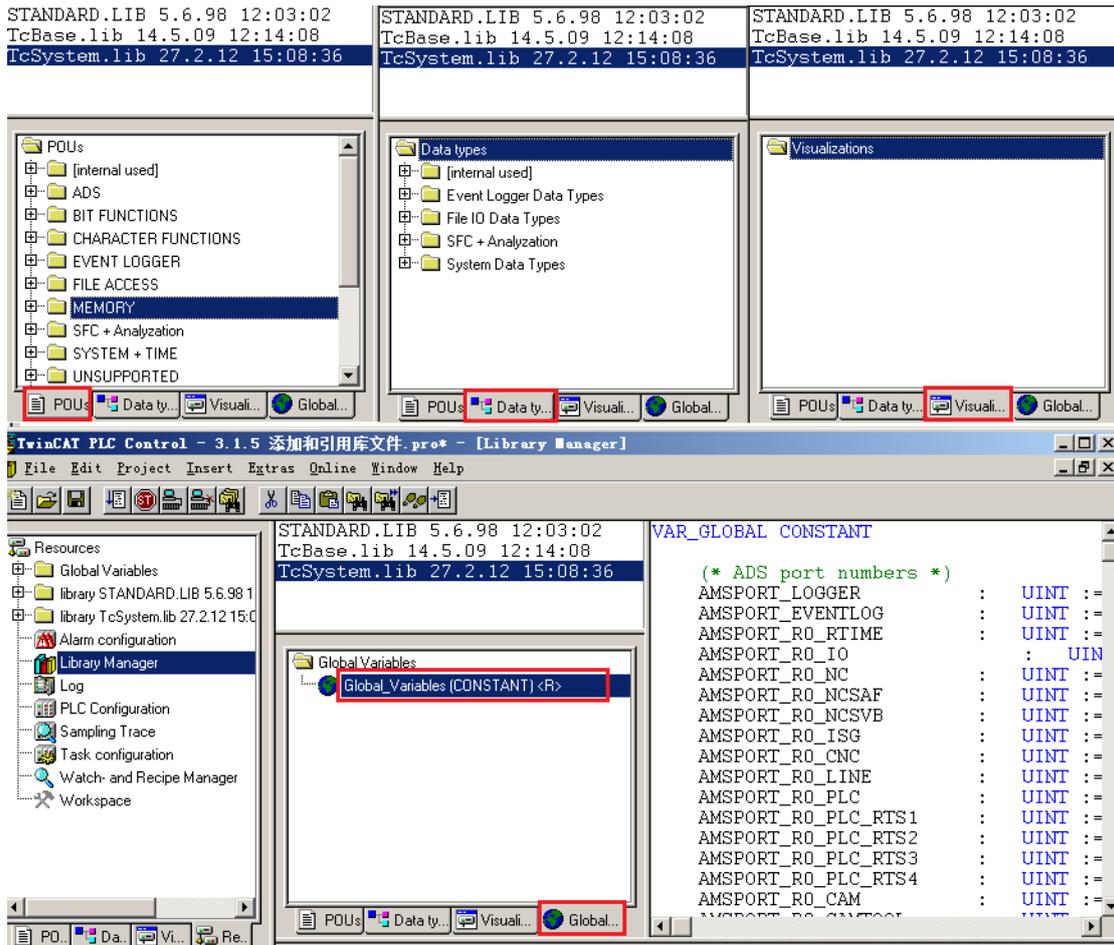
以 TcSystem.Lib 为例：



图中的 TcBase.lib 是由 TcSystem.lib 引用的，所以添加 TcSystem.lib 时自动添加进来。

第 3 步：查看库文件中的内容

和 PRO 程序一样，LIB 库文件中也有 4 个 TAB 页面：POUs、Data Type、Visualization 和 Global。

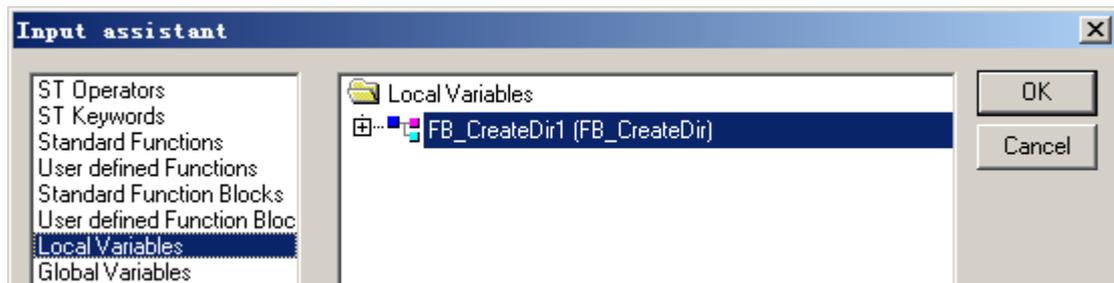


第 4 步：引用 LIB 中的功能块（以结构文本 ST 为例）

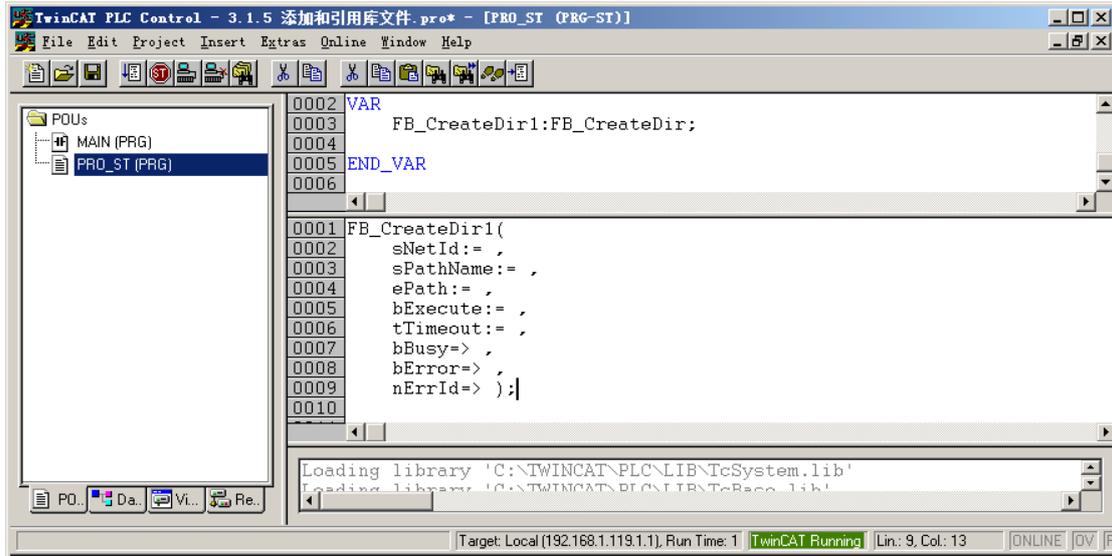
以功能块 FB\_CreateDir 为例，先复制字符串“FB\_CreateDir”，

然后新建一个程序 PRO\_ST，声明一个局部变量“FB\_CreateDir1:FB\_CreateDir;”

在程序代码区，按辅助输入键“F2”，选择 Local Variables，选择局部变量 FB\_CreateDir1，

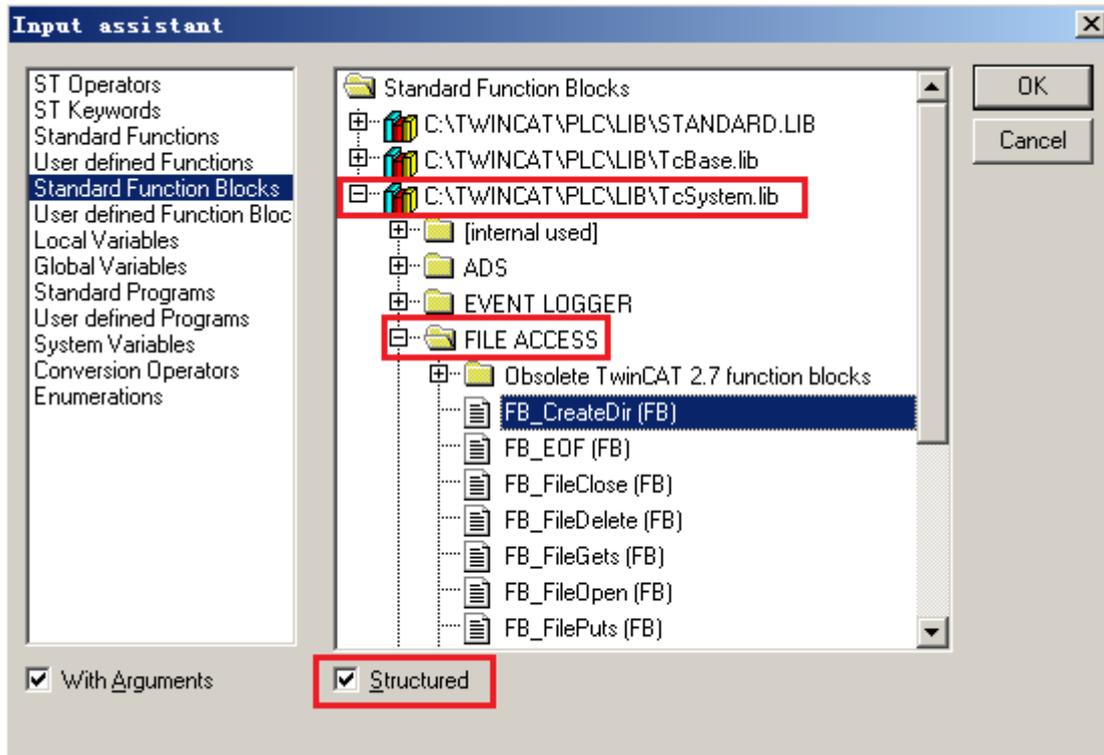


结果如图：

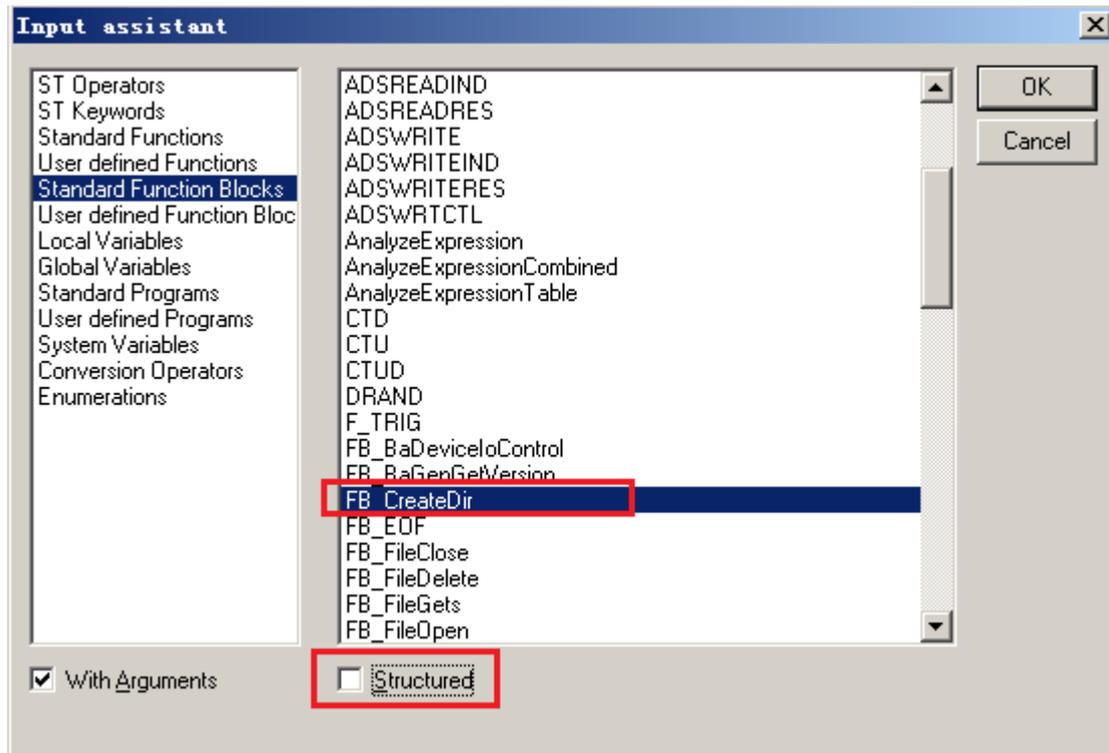


说明：

如果没有提前声明这个类型的局部变量，就要在代码区按辅助输入键“F2”，然后在弹出的窗体中选择。

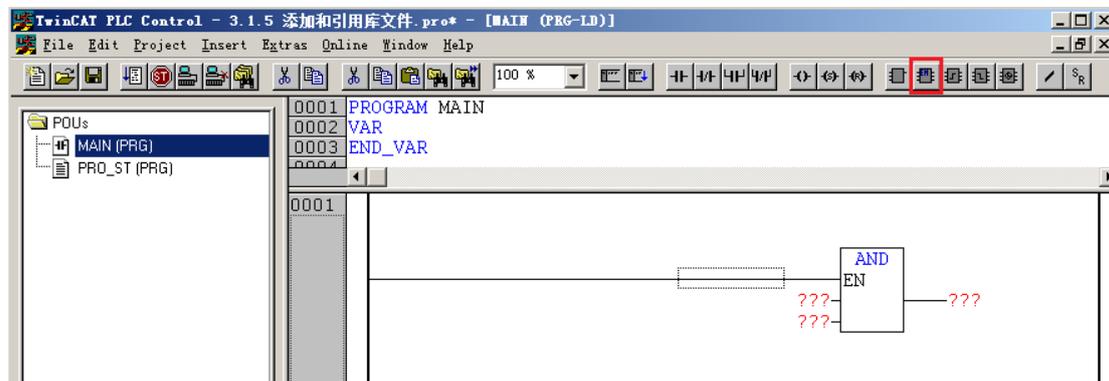


如果不记得 FB 的位置，就得取消“Structured”，然后按字母顺序查找：

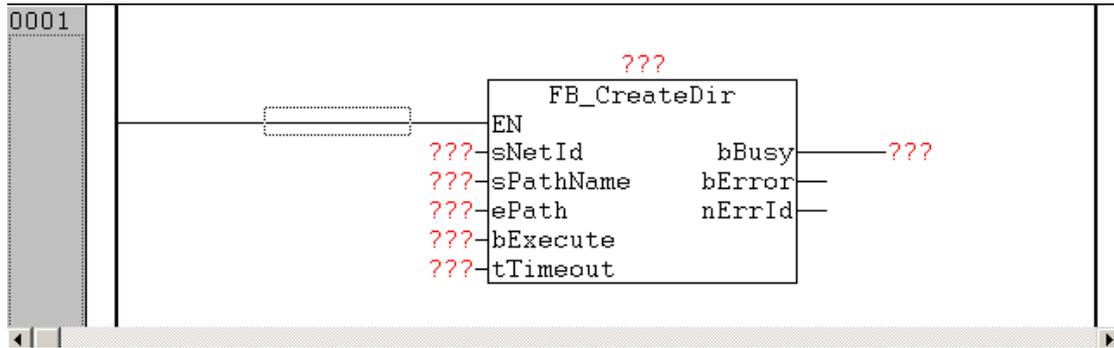


第 5 步：引用 LIB 中的功能块（以梯形图 LD 为例）

插入一个 BOX with En. 



选中“AND”，编辑字符为“FB\_CreatDir”，回车：

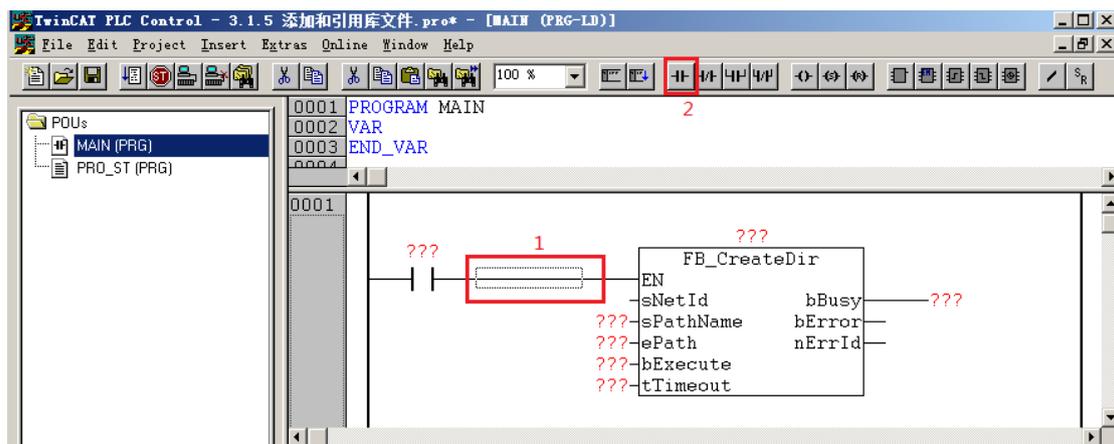


以这种方式插入的功能块，只有当 EN 输入点为 TRUE 时才会执行，对输入输出变量的类型没有要求。

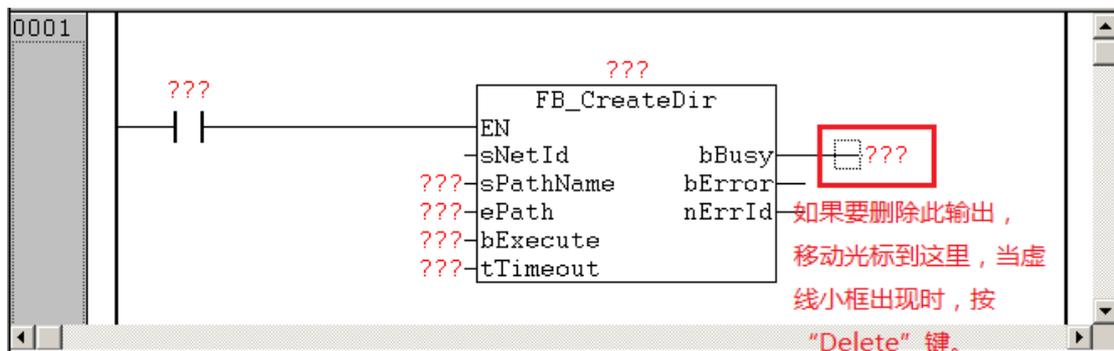
与此相比，以 BOX 方式插入的功能块，要求第一个输入变量和第一个输出变量必须是 BOOL 型变量，而功能块是每个 PLC 周期都无条件运行的。所以本例中的 FB“FB\_CreateDir”，不能以 BOX 方式插入。

给 EN “运行使能” 加入条件：

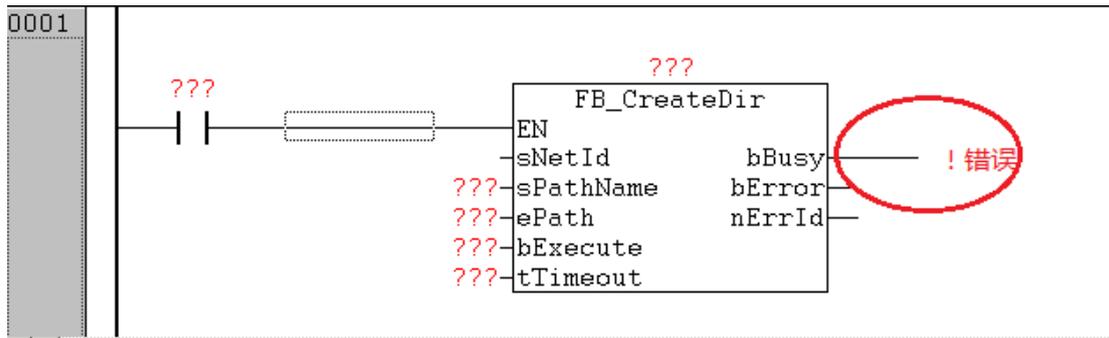
光标定位到“1”处，出现虚线框，然后点击“2”处的图标



默认 FB 的第一个输出要赋给一个程序变量，此时只要把下图中红线框中的“???”改成变量名即可。如果不需要赋给任何变量，则将光标定位到图中红线框的位置，出现一个小的虚线框，然后按“DEL”键，即可删除。



特别提示：直接删除变量占位符“???”而保留一条长线，编译将报错：

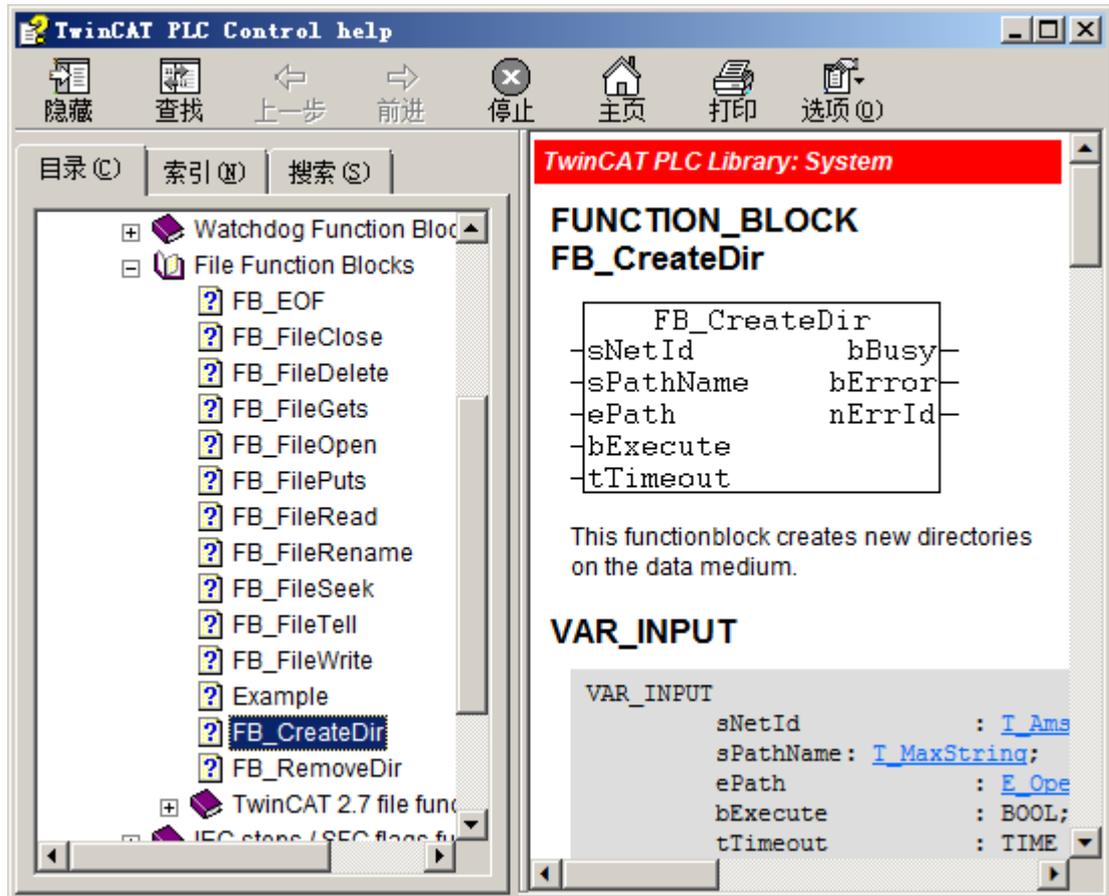


第 6 步：填写接口变量

如果安装了 Beckhoff Information System，就可以查找帮助文档，填写接口变量。要快速查看某个 FB 的帮助信息，可以选中该 FB 的名字，然后按快捷键“F1”，如图：



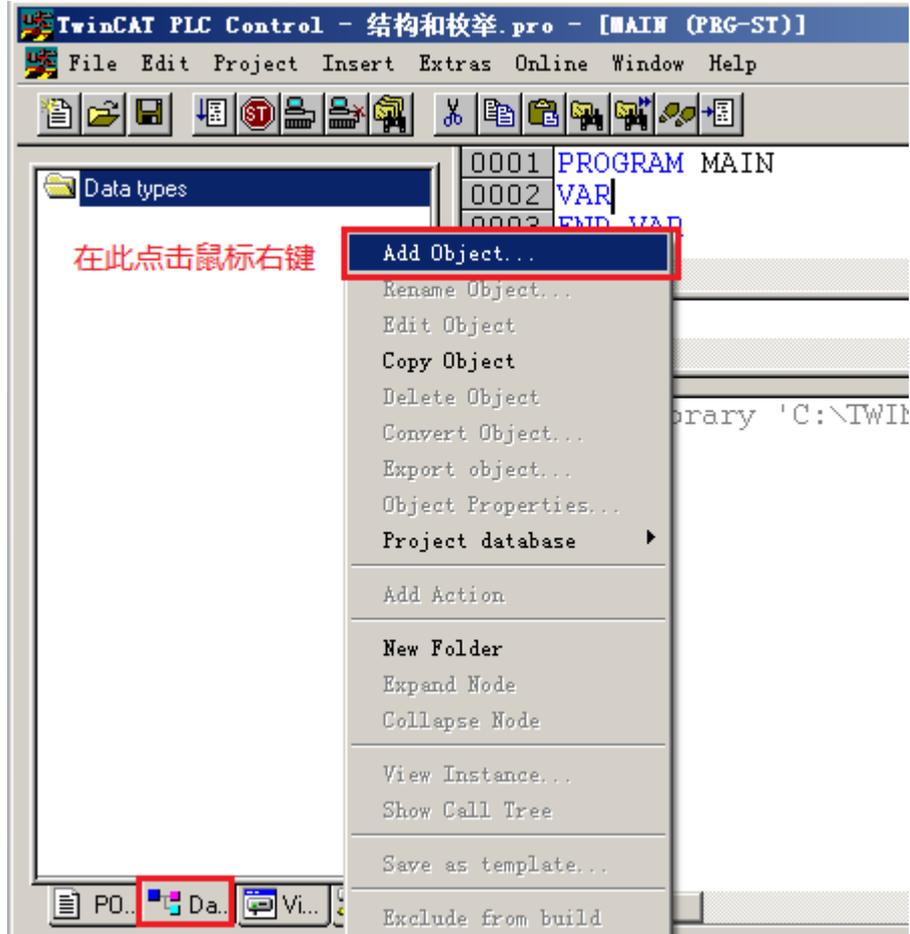
结果启动帮助系统，并定位到相应的 FB 描述位置：



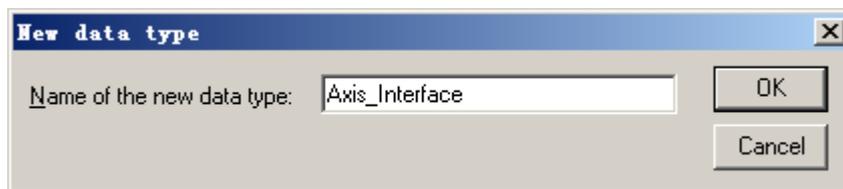
### 3.7.3. 结构和枚举

经过以下步骤，用户可以自己定义项目专门的结构和枚举。

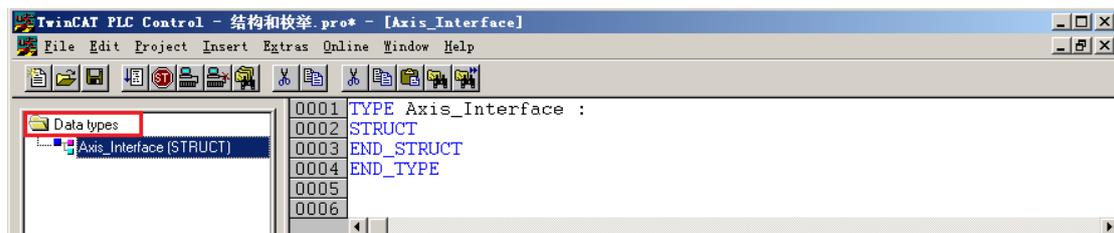
第 1 步：定义一个结构 “Axis\_Interface”



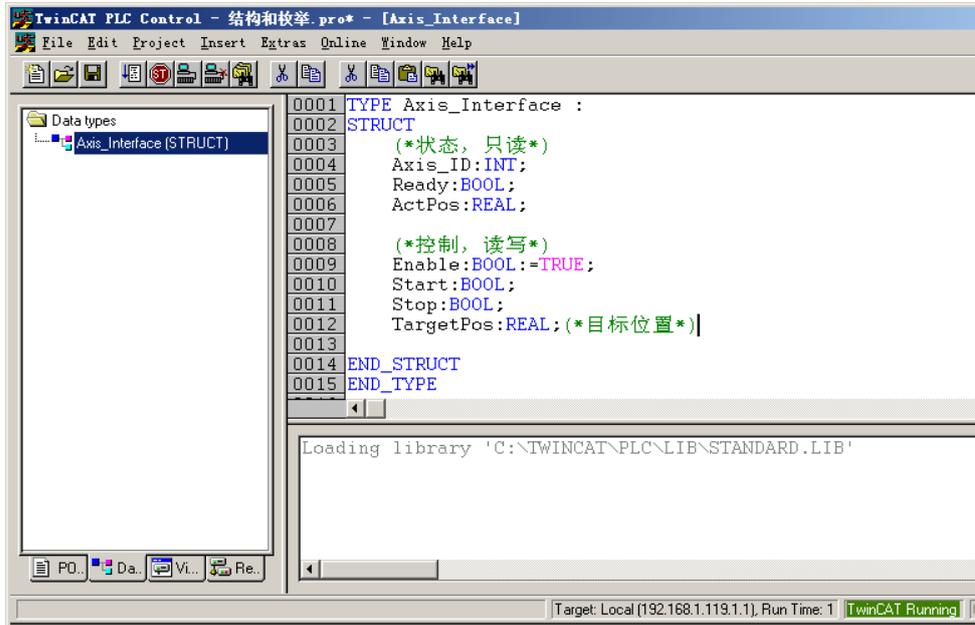
例如，要定义一个运动轴的接口：



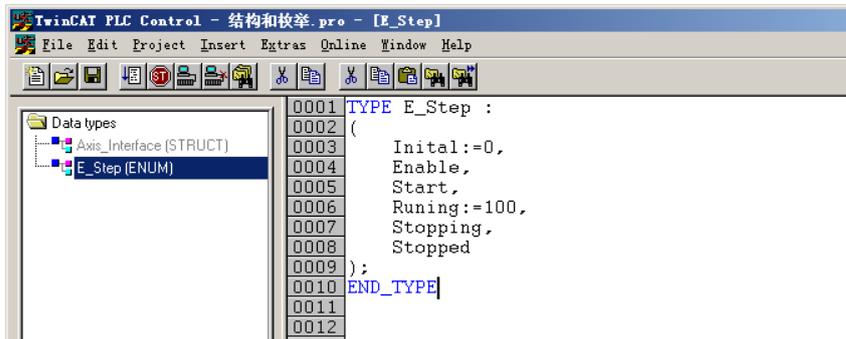
如果新建的结构没有出现，就双击红线框中的位置：



结构中的元素应该写在 STRUCT 和 END\_STRUCT 之间，如图：



第 2 步：定义一个枚举 “E\_Step”

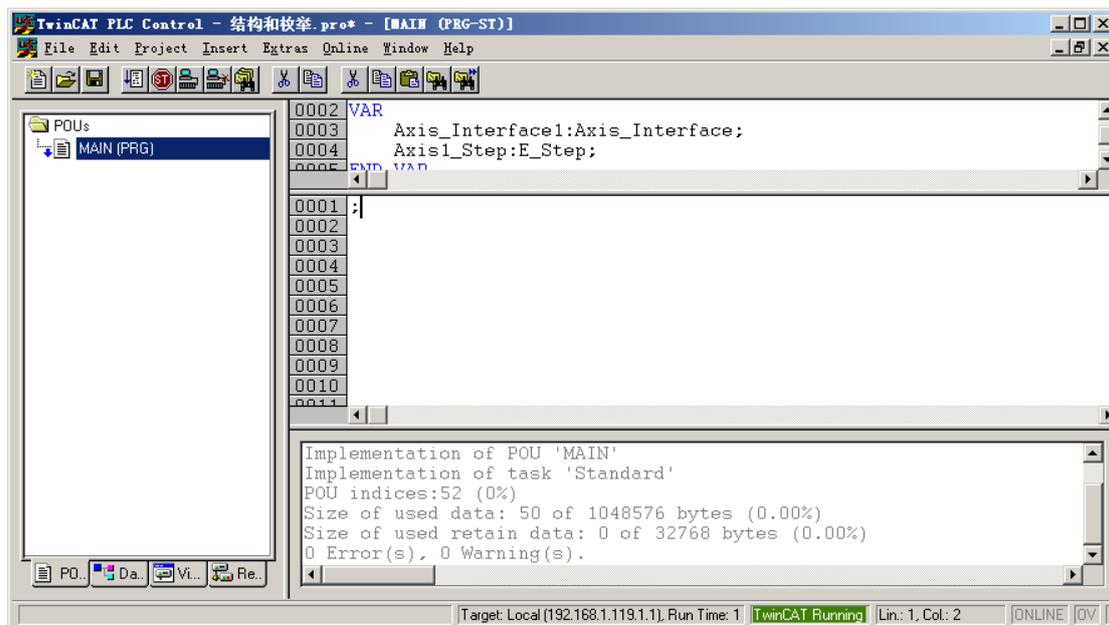
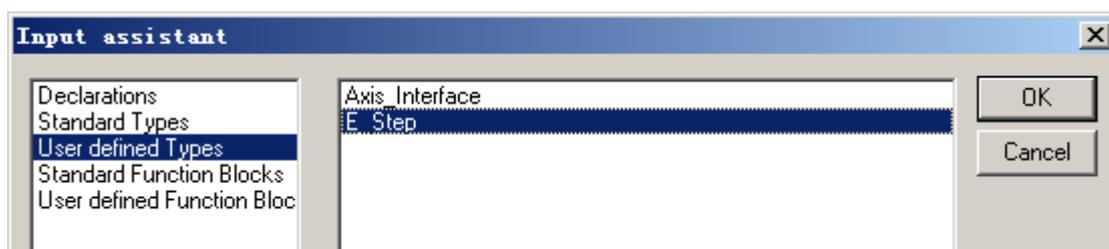
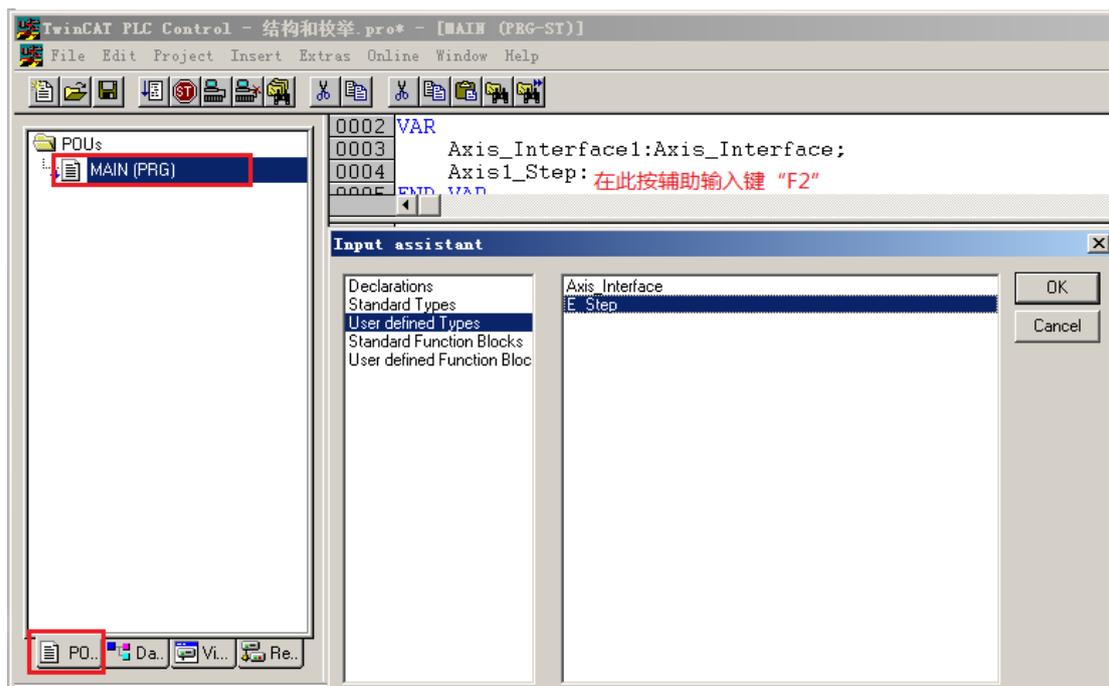


此枚举意为：

- 值为 0，枚举字符为 Initial
- 值为 1，枚举字符为 Enable
- 值为 2，枚举字符为 Start
- 值为 100，枚举字符为 Runing
- 值为 101，枚举字符为 Stopping
- 值为 102，枚举字符为 Stopped

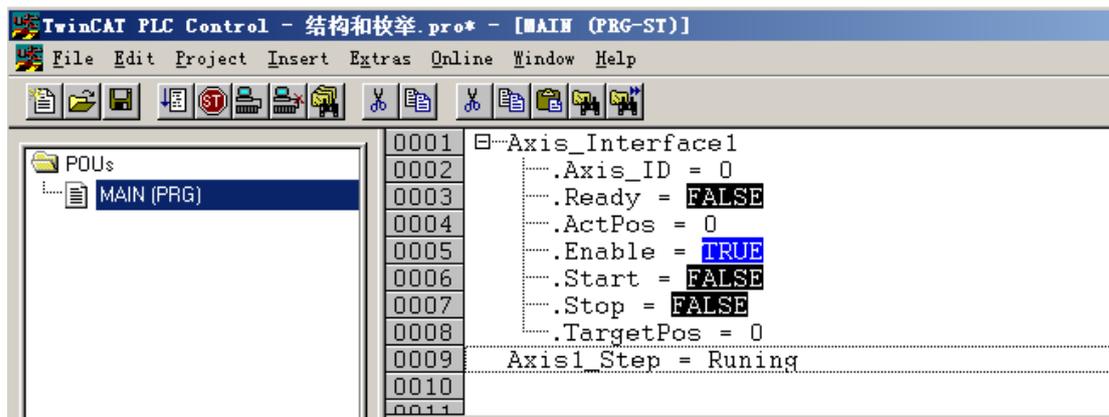
第 3 步：声明一个结构变量和枚举变量

声明变量时，既可以手动输入变量类型，也可以先写变量名称，加“:”，然后用辅助输入键“F2”，在弹出窗体中选择：



第 4 步：联机运行，查看结构和枚举变量。

在变量声明区，双击 Axis\_Interface1 前的 “+”

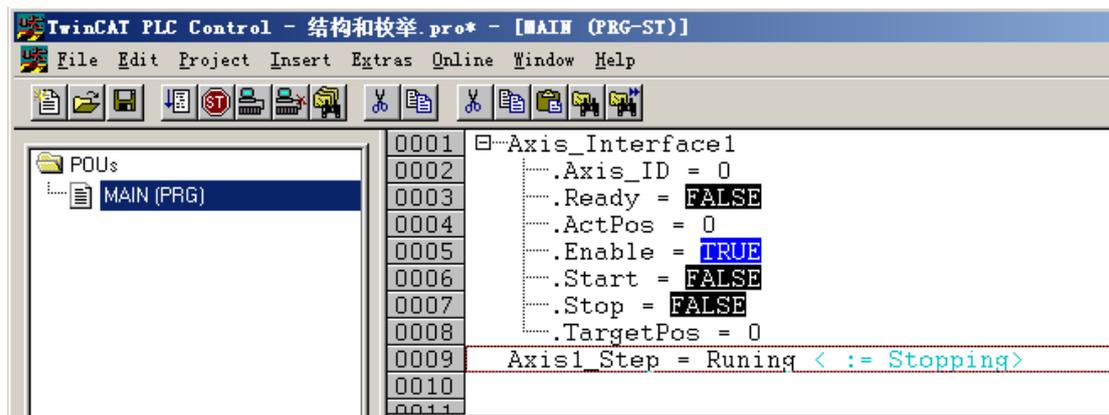


可见该变量的内部元素，正是结构 Axis\_Interface 定义的元素。

双击 Axis1\_Step，输入 “101”



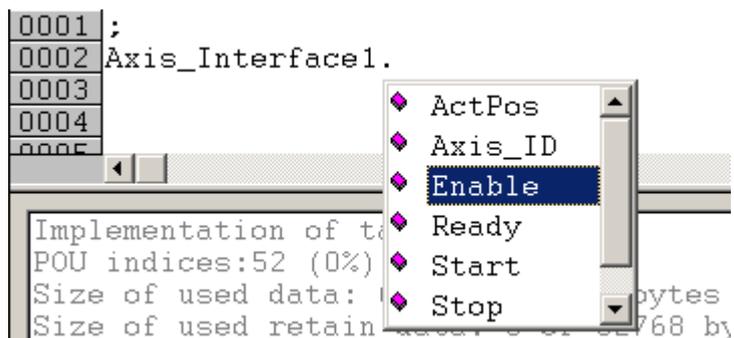
结果：



图上可见，要显示的数据变成了 Stopping，正是枚举定义时 101 对应的字符串。

第 5 步：访问结构变量的内部元素。

在程序代码区，输入变量名，加 “.”，就会弹出内部元素。



### 3.7.4. 数组和指针

数组的定义:

```
比如 : Motors : Array[1..100] OF Axis_Interface ;
       Points : Array[1..10, 1..2] OF Real ;
```

Motors 就是一个一维数组，每个元素都是上节定义的结构 Axis\_Interface 类型。

Points 就是一个二维数组，每个元素都是 Real 类型，相当于一个两行 10 列的矩阵。

数组的访问:

```
Motors[1], Motors[2], .....
Points[1,1], Points[1,2], .....
```

指针的定义:

```
pMotor : Pointer to Axis_Interface ;
pPoint : Pointer to Real ;
```

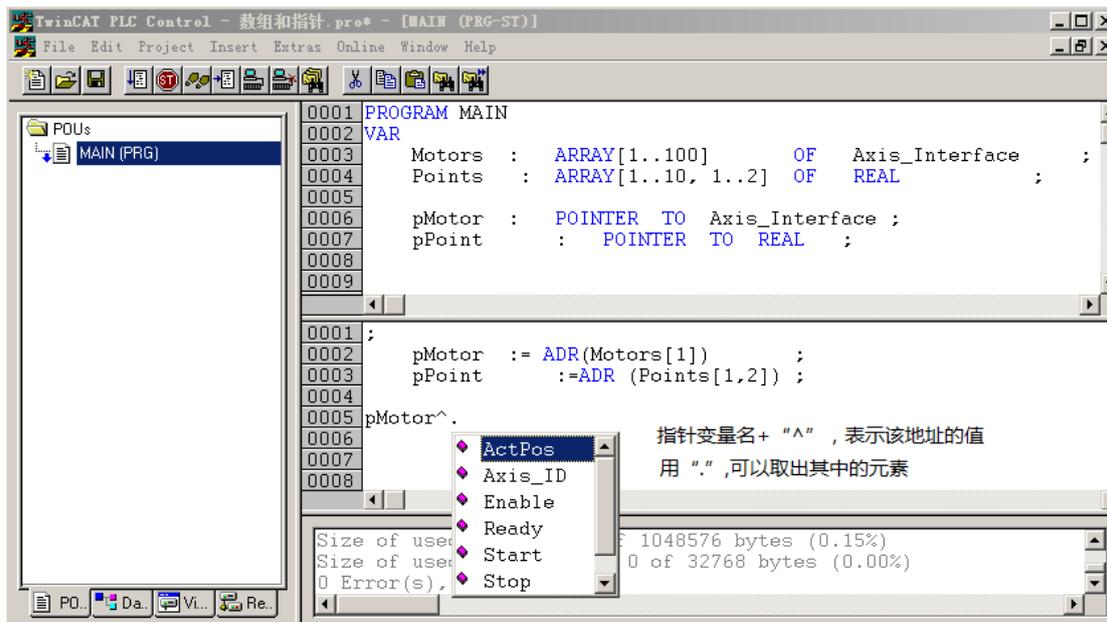
指针的赋值:

```
pMotor := ADR(Motors[1]) ;
pPoint := ADR(Points[1,2]) ;
```

指针的取值:

后置 “^”，可以取出指针中的值。

如果是指向结构的指针，还可以通过 “.” 取出子元素。



提示：指针一定要初始化，即指定一个地址给它，否则初始值为 0。

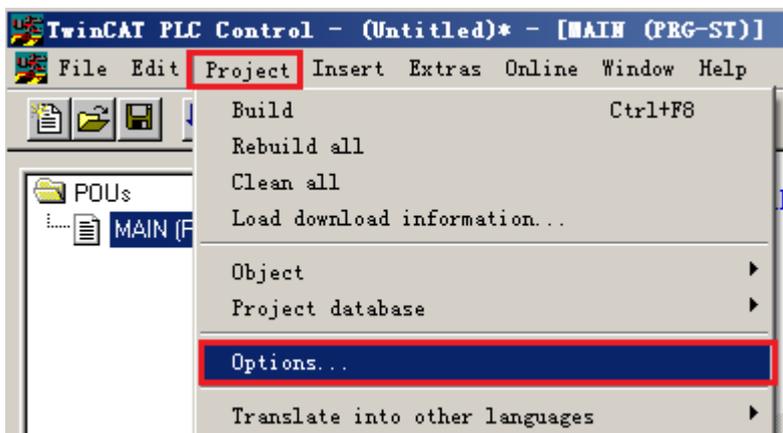
### 3.7.5. 项目加密和对象加密（TC2）

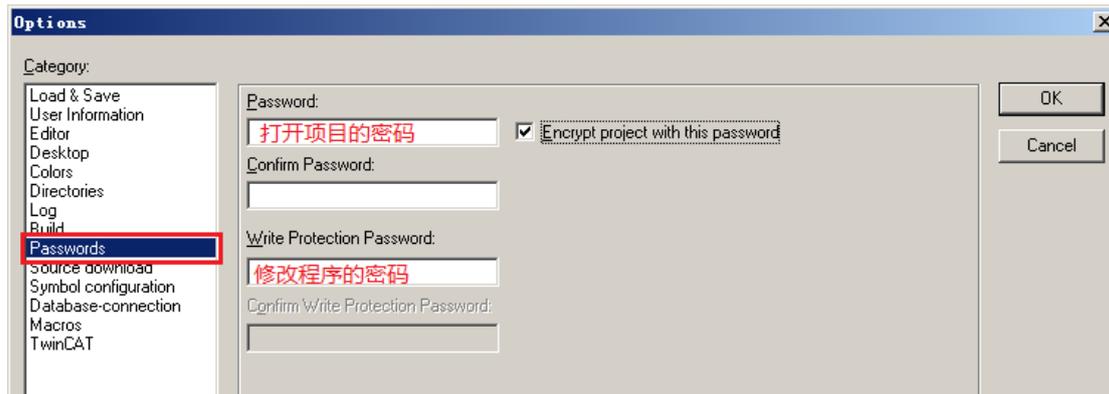
以 MyLib.lib 为例。

第 1 步：打开 MyLib.lib。

第 2 步：设置项目加密

项目加密：





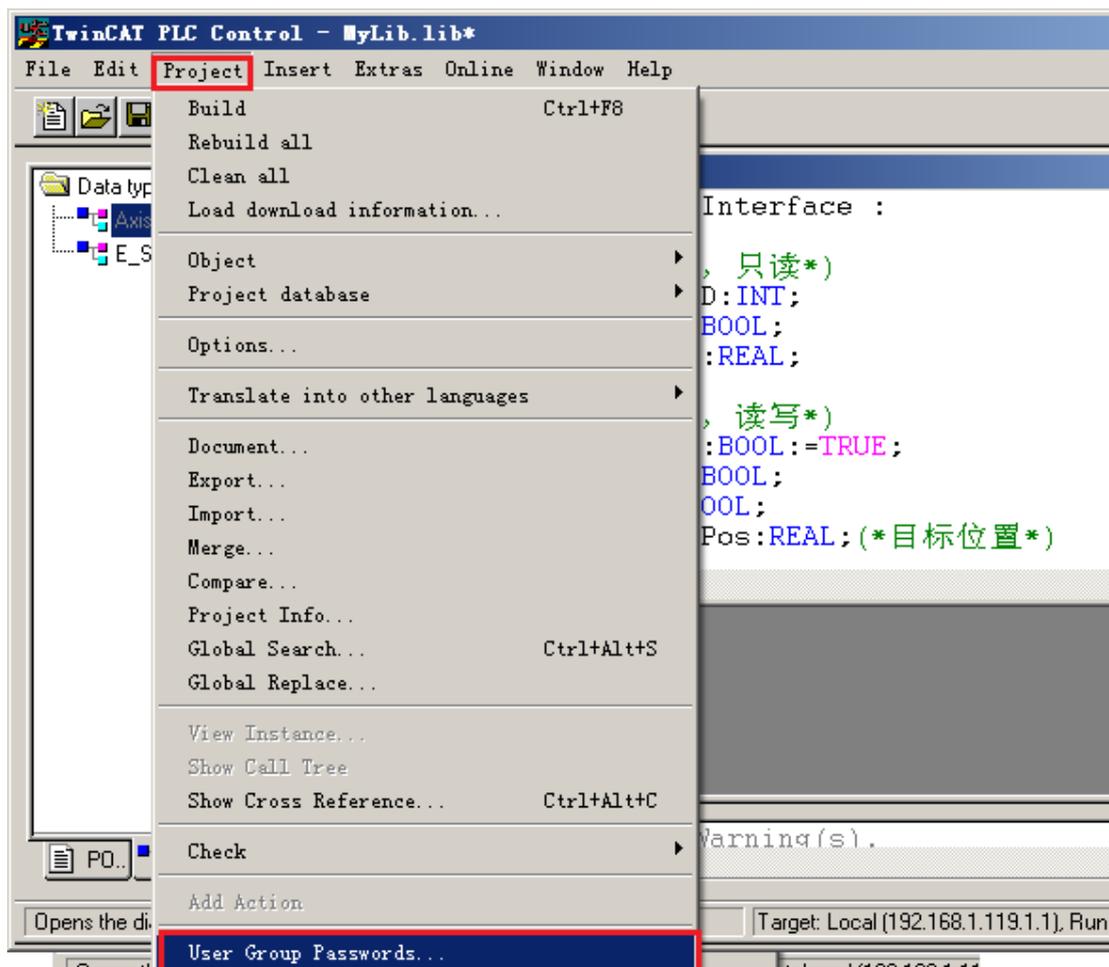
第 3 步：设置对象加密。

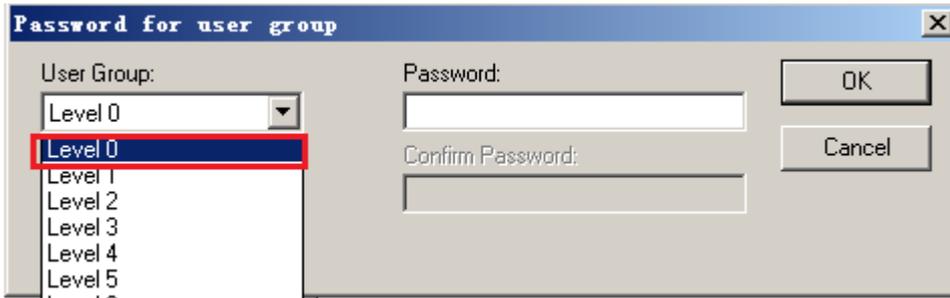
对象的加密是通过对象属性和用户组密码配合实现的。

程序分为 0 到 7，共 8 个用户组，分别设置密码。

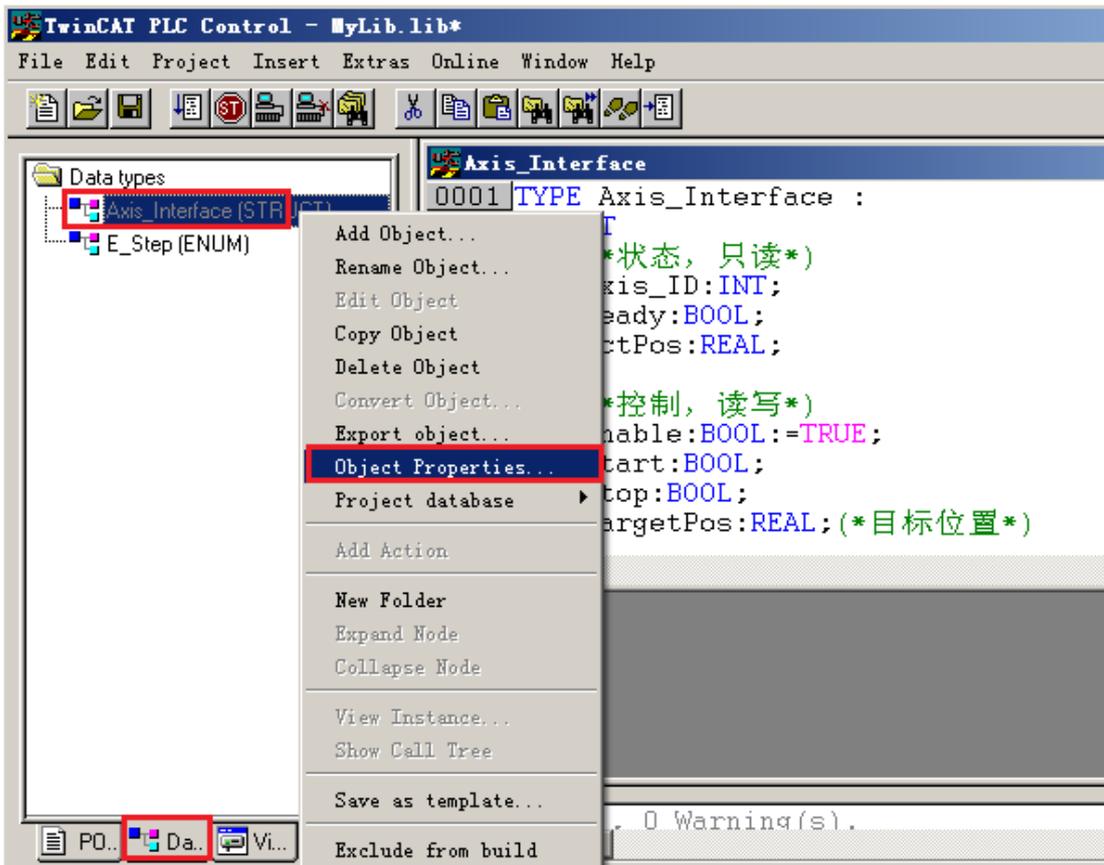
在每个对象（Object）的属性（Property）里，可以设置允许哪些用户组访问。

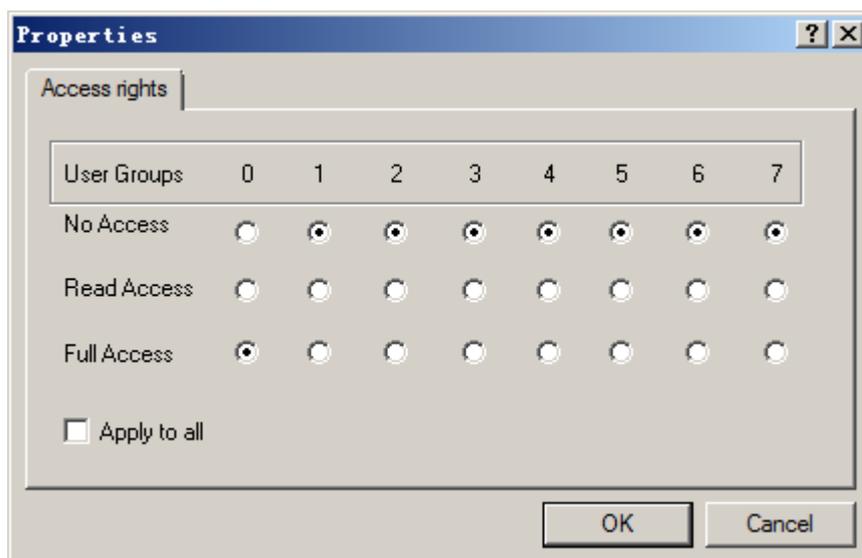
程序单元（Object）密码





设置密码为“123”，点 OK。



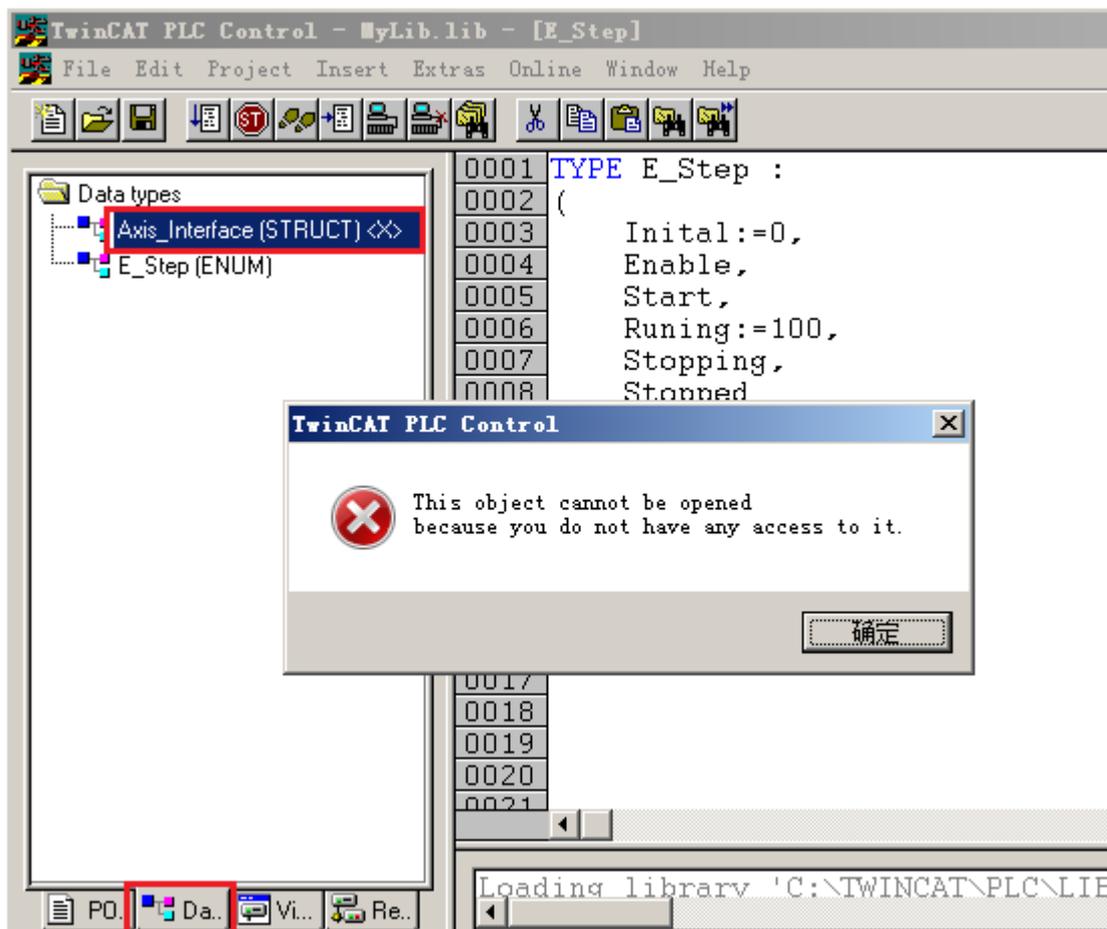


第 4 步，关闭程序。

第 5 步，打开程序。



假如不知道 Level 0 的密码，则选择 Level 1，直接点 OK.

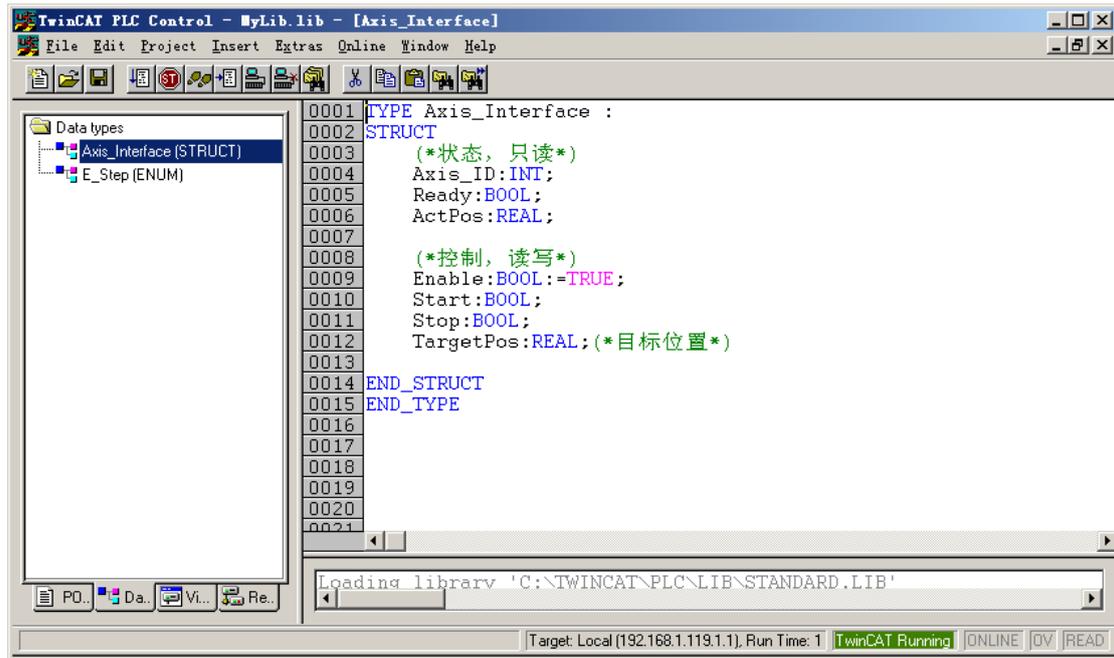


图中，枚举 E\_Step 没有加密，双击即可显示内容。而 Axis\_Interface 的属性中有设置只有 Level 0 才能访问，而由于不知道 Level 0 的密码，选择以无须密码的 Level 1 打开程序。所以此时，Axis\_Interface 的内容不可见，系统弹出报警。

关闭退出程序，重新打开，选择 Level 0，输入密码“123”，



可以看到自定义类型 Axis\_Interface 的内容：

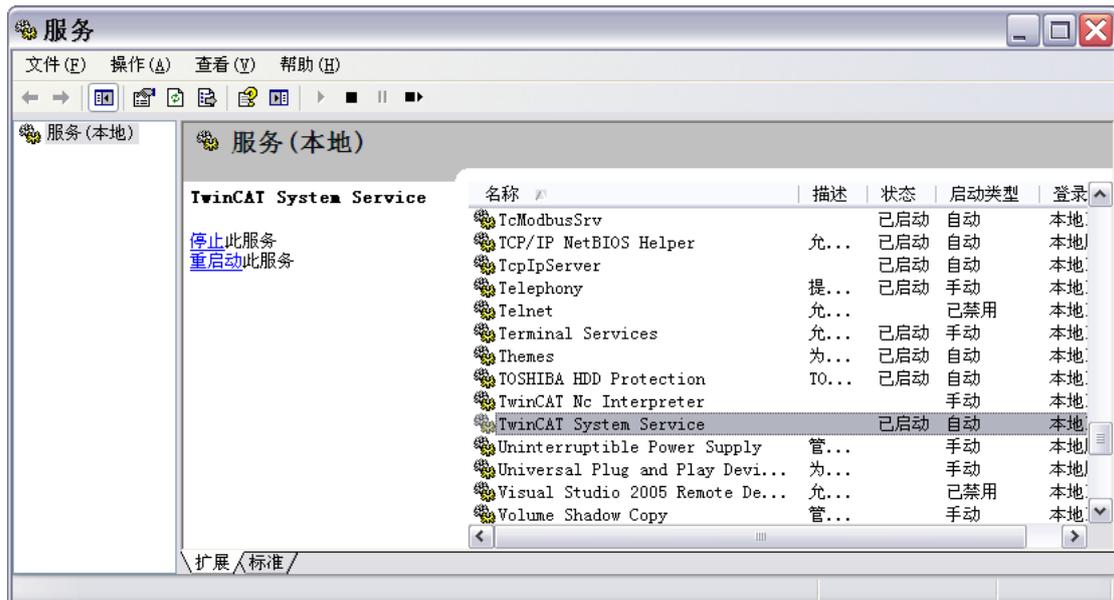


### 3.8. 禁止 TwinCAT 的开机自启动

对于同时安装了 TC2 和 TC3 的开发 PC，同时最多只能运行二者之一。如果要切换请参考“2.2.3 在 TC3 和 TC2 之间切换”。如果开机时两个都要禁用，则按下面的说明。

TwinCAT 作为 Windows 系统下优先级别最高的服务，默认为开机自启动，所以每次开机启动的时间都比较长。对于不是经常使用 TwinCAT 开发的用户，为节约时间，可以将 TwinCAT 服务设为手动启动。方法如下：

打开“控制面板”——“管理工具”——“服务”，进入服务窗口：



选中 TwinCAT System Service，双击，



修改启动类型为：手动。按确定，退出。

## 3.9. 其它提示

### 3.9.1. 弹出窗和提示。

因为 TC2 的消息窗大家都很熟悉了，种类也不多，常常凭记忆操作。但 TC3 中的消息窗多得多，使用初期也还不熟悉，所以要认真观察再决定点“Yes”还是。

例如：使用自动分配地址（%I\*），需要 Clean All/Rebuild All 操作的时候就要注意看提示窗，问“是否清除地址”时要选择“No”。

### 3.9.2. TC2 的控制器可以刷 TC3 的 IMAGE 试用

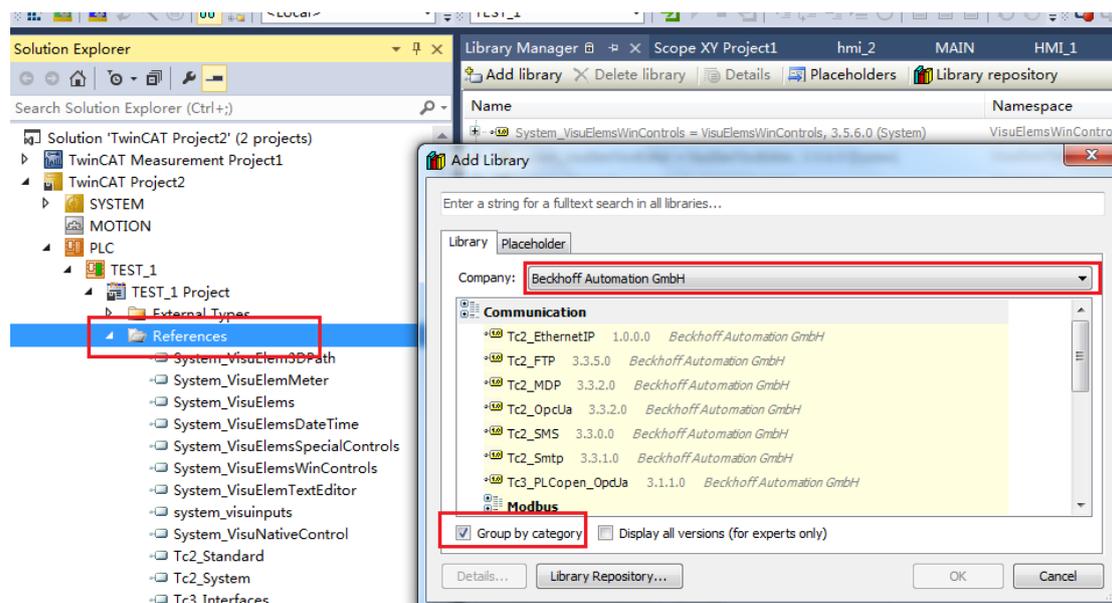
但只有 7 天试用期。反之，如果购买控制器时是 TC3 的授权，就不能刷成 TC2 的试用版。

## 4. TwinCAT 3 扩展功能

### 4.1. 引用库文件

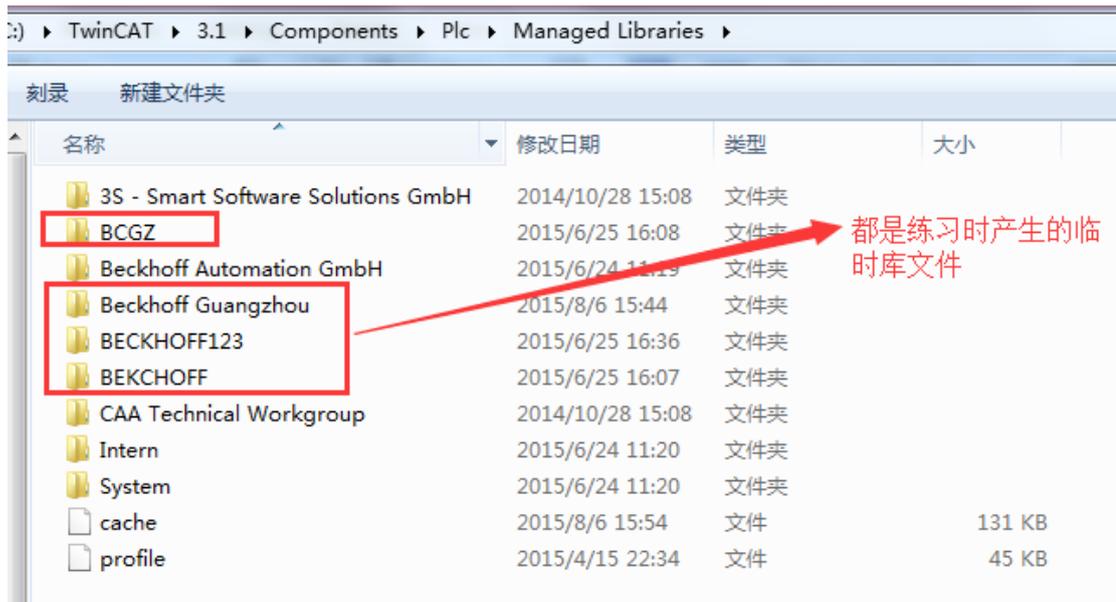
#### 4.1.1. Add Library

Advanced 方式显示的库

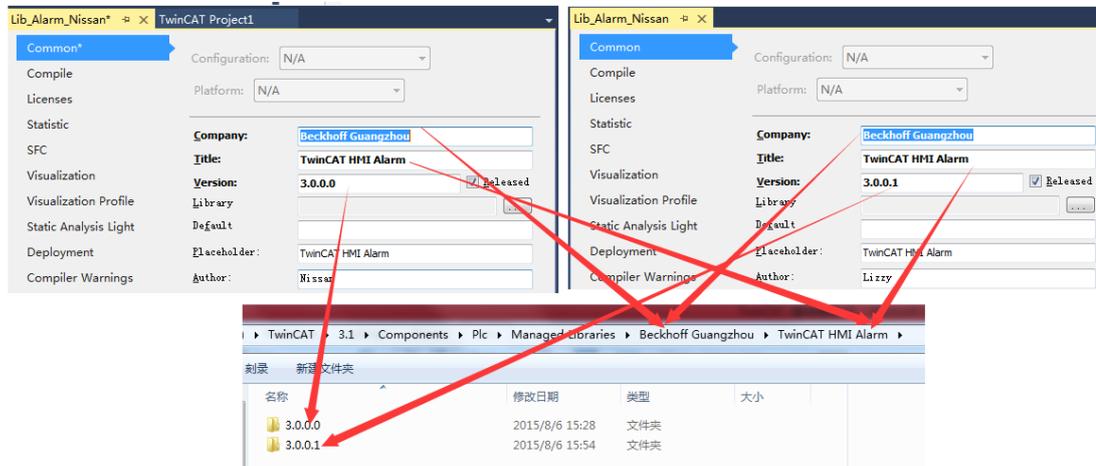


#### 4.1.2. 自定义库文件

在 Save And Install 库文件的时候，实际上在是把 Library 复制到了  
“C:\TwinCAT\3.1\Components\Plc\Managed Libraries”。



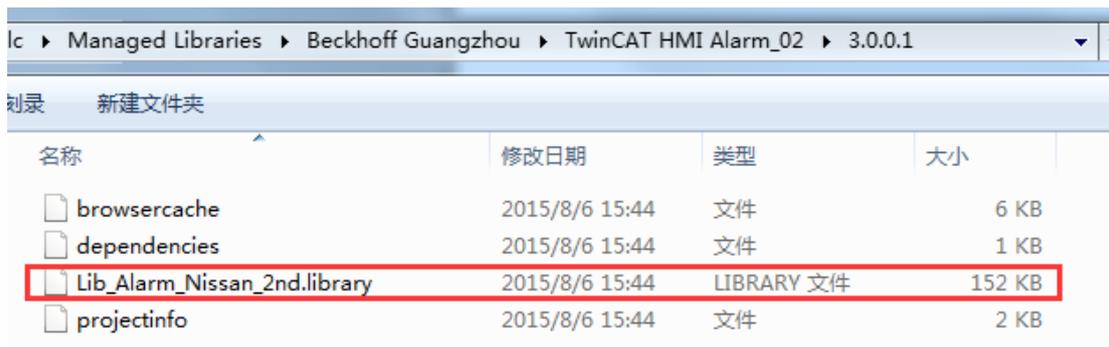
依据属文件中的属性设置：Company、Title 和 Version，依次定位 3 层路径。由于同样功能不同版本的库是存放在不同的目录下，所以 TwinCAT 3 不会出现 TC2 里面的库文件版本换了，引起程序不兼容的情况。



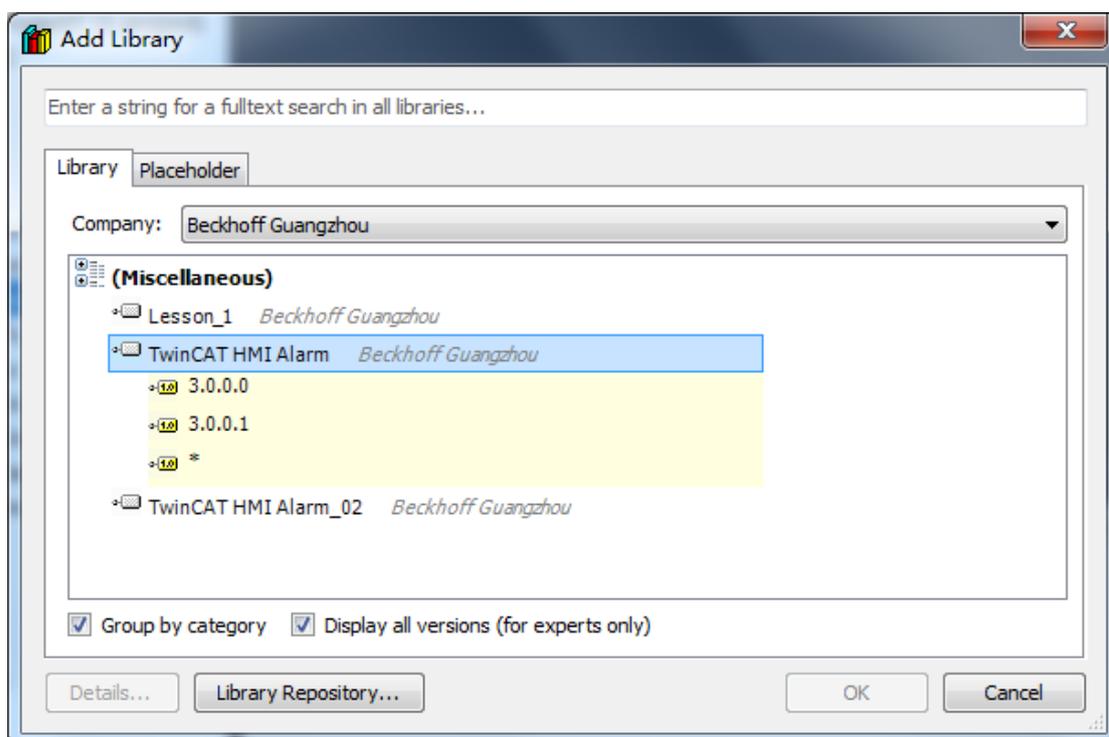
每一个版本下，都包含了 1 个 Library 文件和 3 个描述文件。

browsercache	2015/8/6 15:48	文件	6 KB
dependencies	2015/8/6 15:48	文件	1 KB
<b>Lib_Alarm_Nissan.library</b>	2015/8/6 15:48	LIBRARY 文件	155 KB
projectinfo	2015/8/6 15:48	文件	2 KB

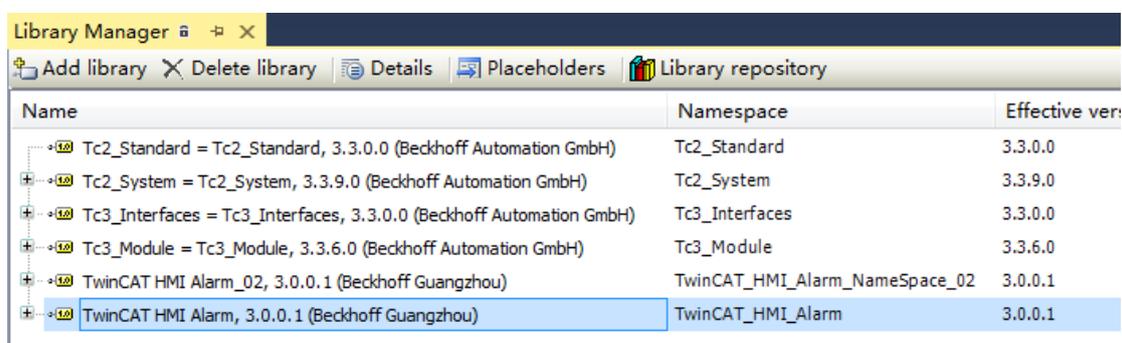
又比如：



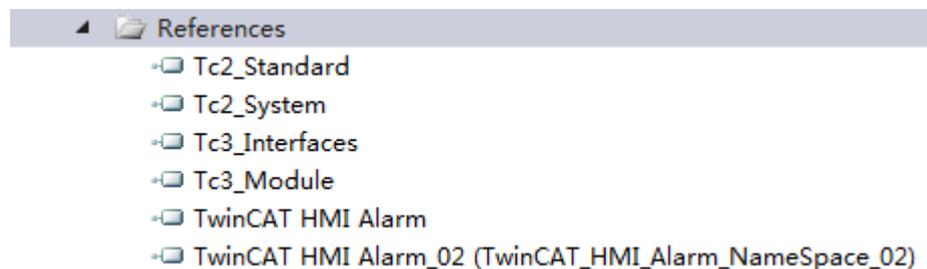
引用库时，可以依据 Company 和 Title 索引：



如果有自定义的与 Title 不同的 NameSpace，在 Library 中可以体现：



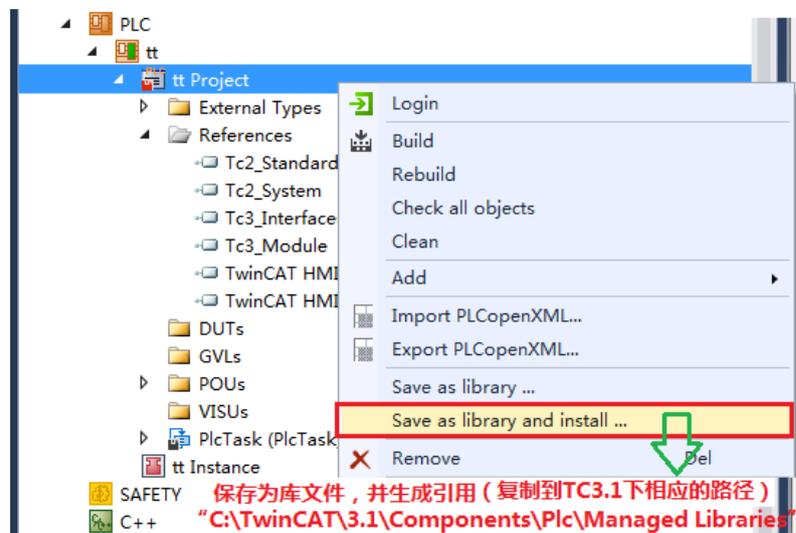
如果有自定义的与 Title 不同的 Placeholder, 在项目的 Reference 中体现为:



括号外面是 Placeholder, 括号里面是 NameSpace。

如果同一 Title 和命名空间, 不同的版本, 使用不同的 Placeholder, 就比单纯的版本号更容易识别。比如 Nissan 用的库, 和 Techlong 用的库, 和五菱用的库, 各更改装之后都不相同。

生成库文件。

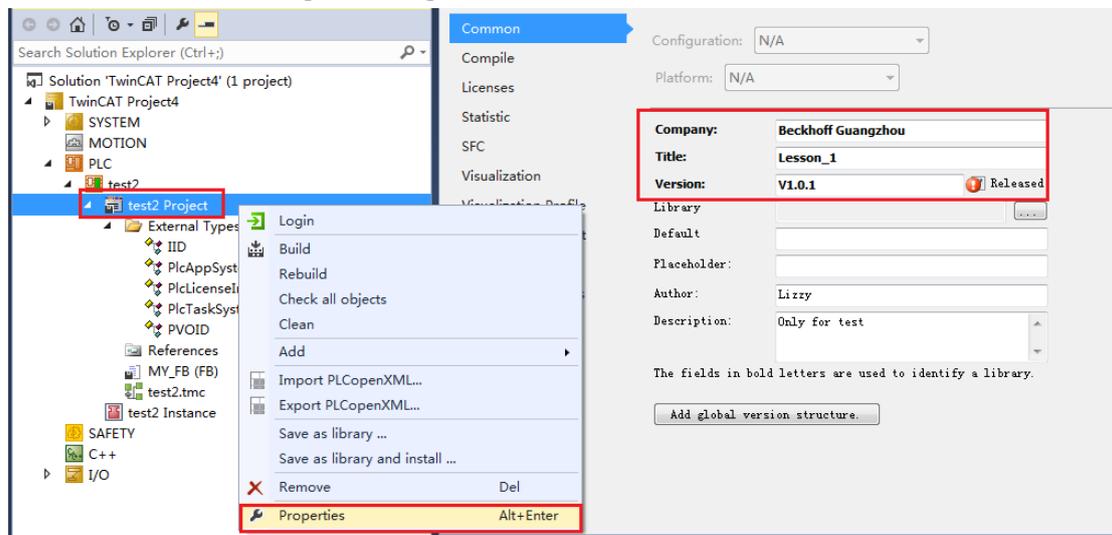




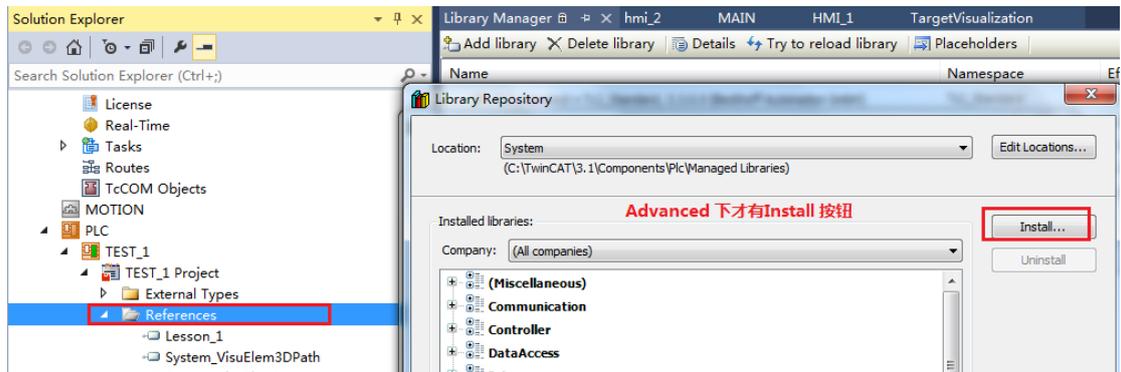
如果没有重复的库，就可以直接引用全局变量名：



另存为时可以选择存为普通的库还是编译过的库。如果是编译过的库，就不能访问代码。如果是普通库，就可以 StepIN 和 Step Over.



### 4.1.3. 引用别人的库



### 4.1.4. 命名空间

可以用不同的命名空间区分同一个库的不同版本。不同版本的库里有同样的变量和 FB，引用时要先注明命名空间。然后通过.号引出。

用法 1：同名的功能块在不同的库里面，以库的名称来区分。

用法 2：禁止调用。

## 4.2. Measurement

### 4.2.1. TC3 Scope Server, 免费版与收费版的功能差别

for the configuration or recordings.

#### Product level / feature list

This table shows which functions are available with which TwinCAT Scope level with the corresponding licensing.

Features	Scope Base		Scope Professional			
	Server	View	Server		View	
			Full	7-day test version	Full	7-day test version
General:						
Free of charge	✓	✓	✗	✓	✗	✓
Local record	✓	✓	✓	✓	✓	✓
Remote Record using Target Server	✗	✓	✓	✓	✓	✓
Remote Record using Local Server	✓	✓	✓	✓	✓	✓
Scope Control Integration	-	✗	-	-	✓	✓
Long Time Records > 1h	-	✗	-	-	✓	✗
Application Settings	-	✗	-	-	✓	✓

23 books returned from <http://services.mtps.microsoft.com/ServiceAPI/catalogs/VisualStudio12/en-Us>

### 4.2.2. Scope 导出数据

TwinCAT Project2 - Microsoft Visual Studio

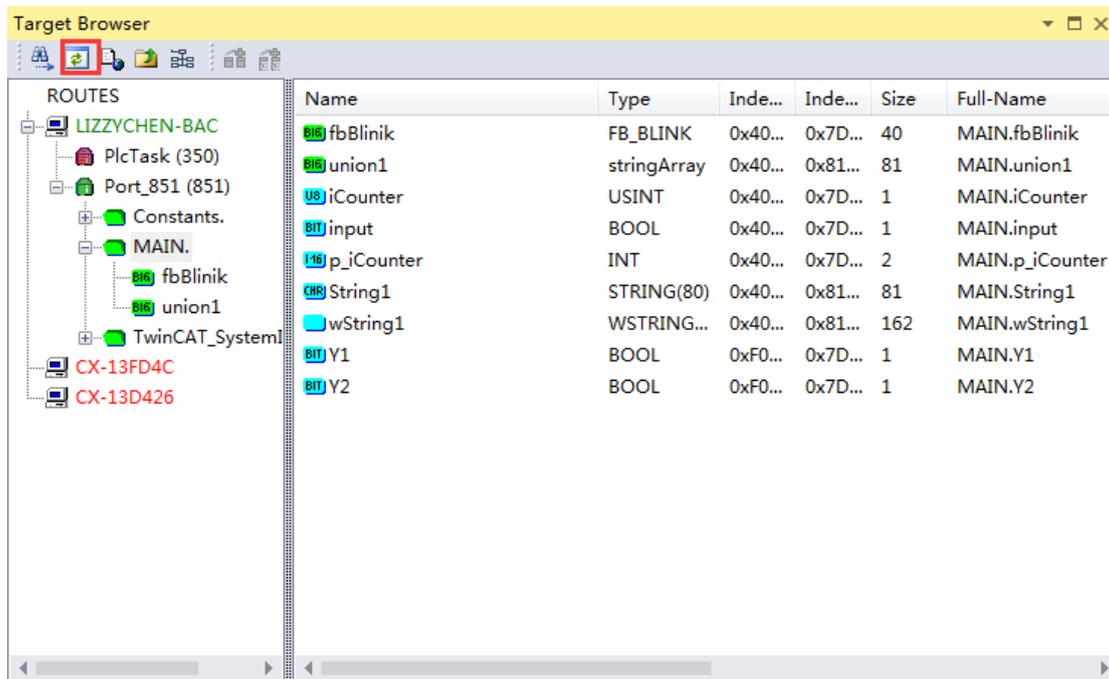
FILE EDIT VIEW PROJECT BUILD DEBUG TWINCAT PLC TOOLS **SCOPE** WINDOW HELP

Target Browser  
Cursor Window  
Apply Defaults  
Send Project By E-Mail...  
Clear Error List  
Change Ads Symbol...  
Change Index Group...  
New Axis  
New Empty Channel  
Copy Ctrl+C  
Delete Del  
**Export to CSV**  
Export to Binary  
Export to DAT  
Export to TDMS  
Stacked Axes  
Scope Messages  
Local Scope Server...  
Graphic Library  
Options

### 4.2.3. 常见问题

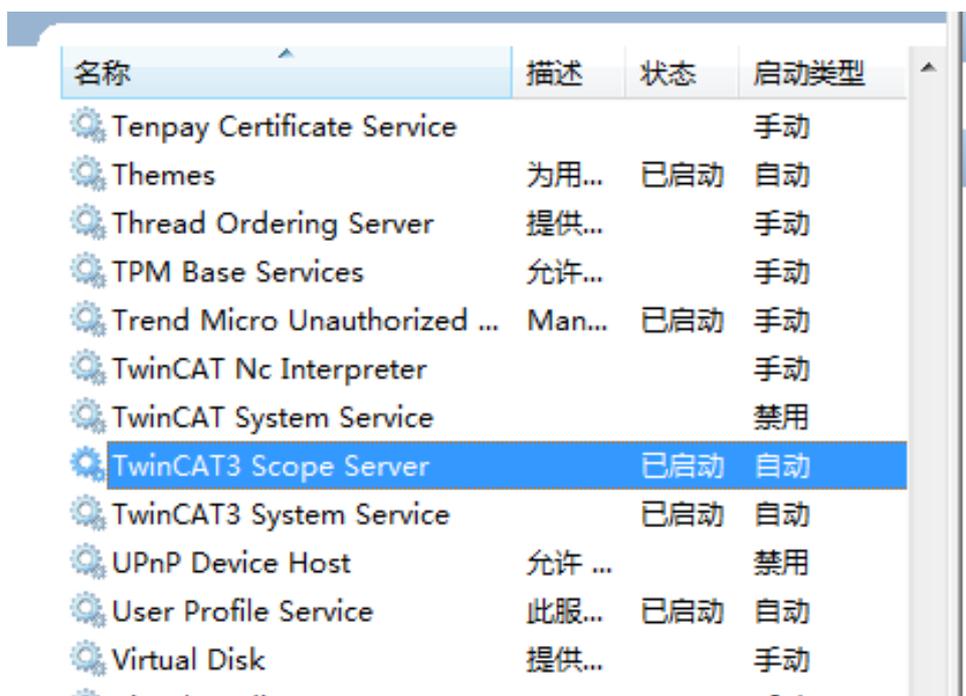
- 刷新变量

如果改变了变量，在 Measurement 里建变量的时候，要刷新一下变量。否则显示的还是以前 PLC 程序变量。



- Scope Server 启动错误

在计算机管理，服务设置里手动启动



## 4.3. 程序归档

### 4.3.1. Measurement 项目的存储路径

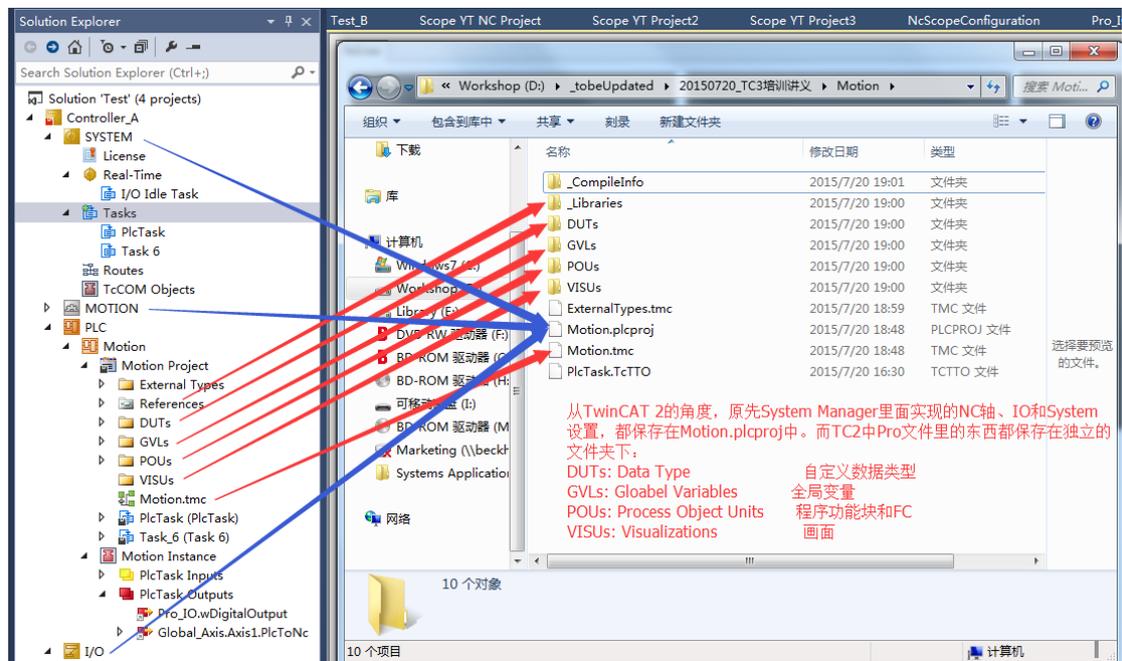
一个 Measurement 项目包括项目本身和若干个 Scope 窗体设置，都独立保存为一个文件。



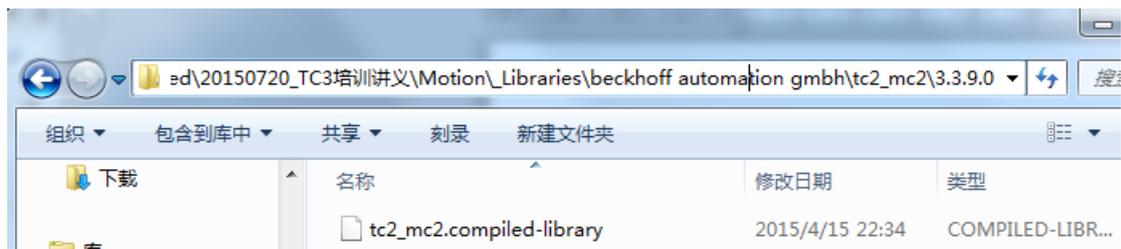
记录的数据可以保存，临时指定存放位置和名字。

### 4.3.2. TwinCAT 项目的存储路径

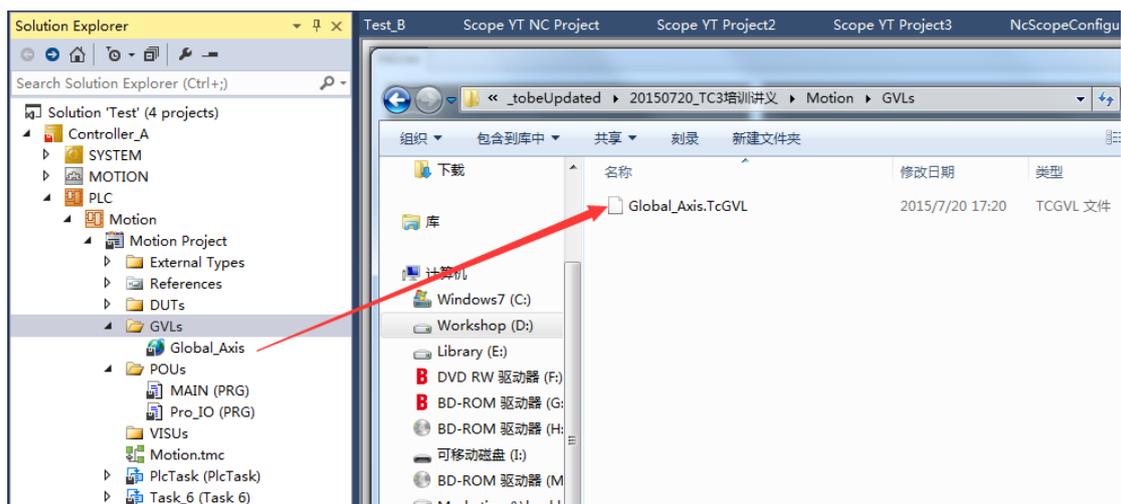
一个 TwinCAT 项目包括，



说明：引用的库文件，编译完成后是存到了项目中。不同版本的库可以共存。



全局变量文件是独立保存的。

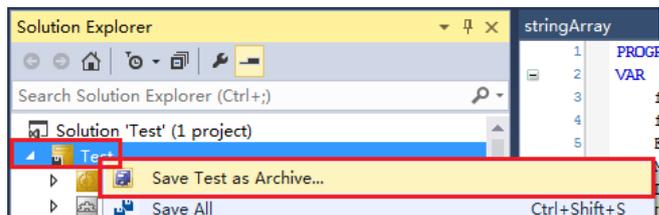


### 4.3.3. 项目打包和解包

如果单独复制其中一个文件夹,再在新项目中 Add Exist Item 也可以。如果要整个项目打包,就用下面的方法。

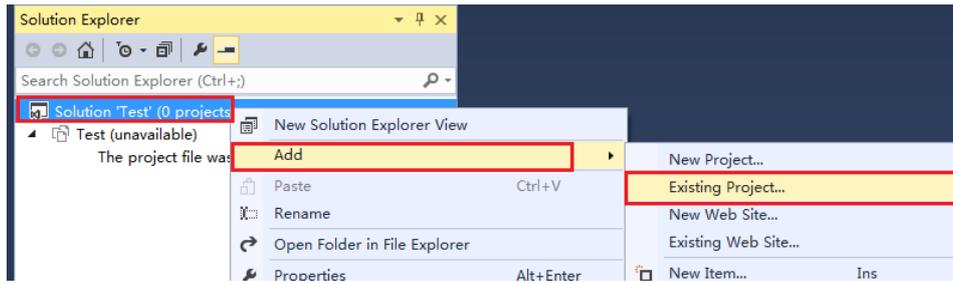
- 打包

如果要迁移到另一台 PC,推荐用 Save <project> As Archive。这样才会把所用到的库都一起导出。

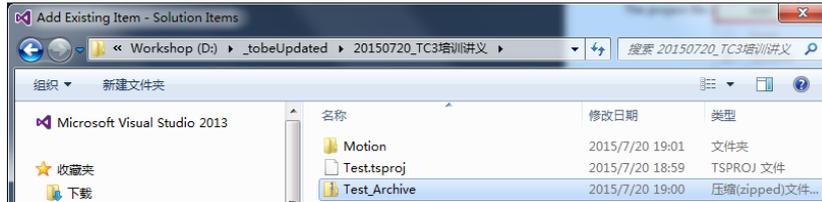


- 解包

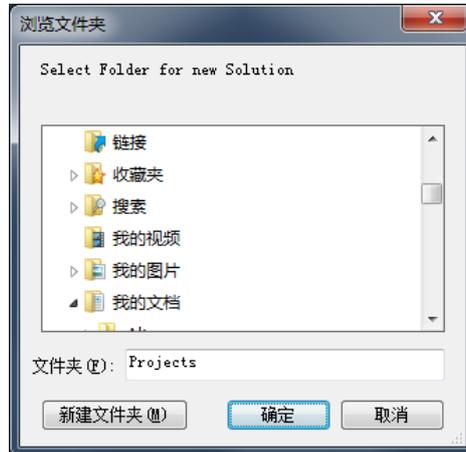
项目归档是把整个项目文件夹压缩存储为 tzip,解包就是从压缩的文件夹中打开项目。



选择 tzip 文件



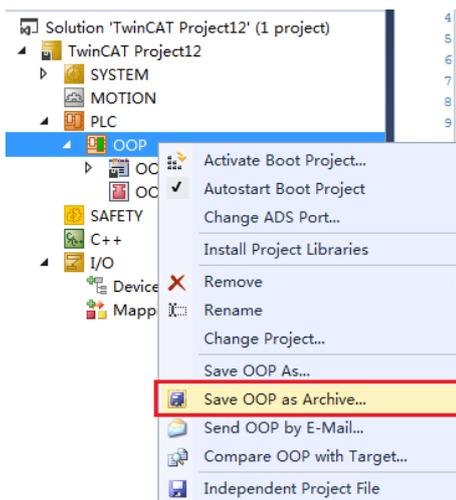
选择解压后的文件夹，可以放到新建项目中：



#### 4.3.4. PLC 程序的打包和解包

- 打包

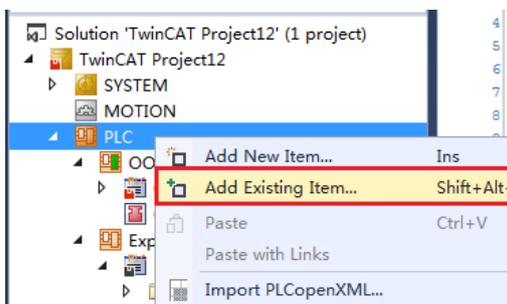
要把现有的 PLC 程序打包，先选中 PLC 名称，右键菜单中选择“Save As Archive”



按提示选择压缩文件 **tpzip** 的名字和存放路径即可。

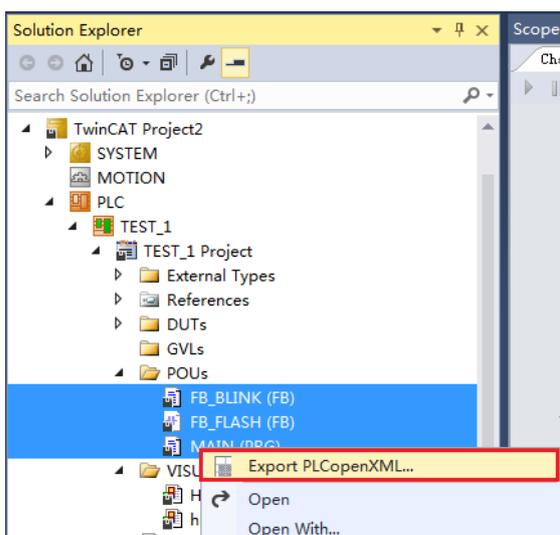
#### ● 解包

要在一个项目中插入一个 PLC 程序，



实际上这个方法也可插入未经打包的 PLC 项目，包括 .plcproj 文件和相关的文件夹。

### 4.3.5. FB 等对象的导出和导入



需要导入时则在同一右键菜单选择 Import 即可。

## 5. 面向对象编程

### 5.1. 概述

#### 5.1.1. 什么是面向对象编程

面向对象编程，即 Object Oriented Programme，简称 OOP 编程，是 TwinCAT 3 编程工具的新功能之一，也是 IEC61131-3 第 3 版的新增内容。

在 TwinCAT 2.0 的 PLC 编程中，需要重复使用的代码，用户可以写成一个 FB，一个 FB 下面又可以编写若干 Action，根据 Input 的变化来决定执行哪些 Action，Action 的执行结果影响了 Output。调用 FB 时需要“实例化”。一个实例，就是一个“对象 (Object)”。

OOP 编程，就是针对 FB 的 Input、Output 和 Action 的另一种实现思路，它的特点是：先确定被控设备有哪些动作 (FB 的 Method) 和特征值 (FB 的 Property)，然后编程实现动作和特征值之间的逻辑关系。FB 的变量和代码与外部隔离，通过属性 (Property) 和方法 (Method) 与其它对象或者程序主体交互。

需要澄清的是，OOP 编程是 TC3 的新功能，但不是必须使用的功能。换句话说，TC2 的用户如果不使用 OOP 编程而是沿用原有的编程思路，也一样可以完成项目。

#### 5.1.2. 关键名词：Function Block 和 Interface

OOP 编程里面有两个关键的名词功能块 (*Function Block*) 和接口 (*Interface*)。Function Block 在 TC2 中已经存在，但在 OOP 编程中却具有了许多新的特点。而 TC2 中完全没有 Interface 的概念，现在则是标准的 OOP 编程术语。

OOP 编程中，与 *Function Block* 紧密相关的关键词是 *Method* 和 *Property*。OOP 编程是用“方法 (Method)”和“属性 (Property)”取代了传统 PLC 编程时使用的 Input 和 Output 变量以及内部的 Action。

“方法 (Method)”，就是“对象可执行的动作”，比如一个指示灯，有“开”和“关”的动作。

“属性 (Property)”，就是“对象的特征值”，包括状态、参数等。比如一个指示灯，有“亮/灭”的状态。

Method 触发的动作，以及 Property 与内部变量的关系，都需要在 FB 中写代码来实现。但它们与 FB 的内部变量和代码是隔离的。

OOP 编程中，与 *Interface*，紧密相关的关键词是 *Implement*。前面讲到“对象”之间相互作用，只能通过 Method 和 Property 而不涉及对方的内部变量和逻辑。如果把 Method 和

Property 从 FB 中独立出来，用一个新的概念来表达，就是“Interface”，Interface 不包含任何代码，它的代码必须在 FB 中实现，所以一个 Interface 至少要被一个 FB 引用才有意义，这个引用关系就用关键词“Implement”表示。

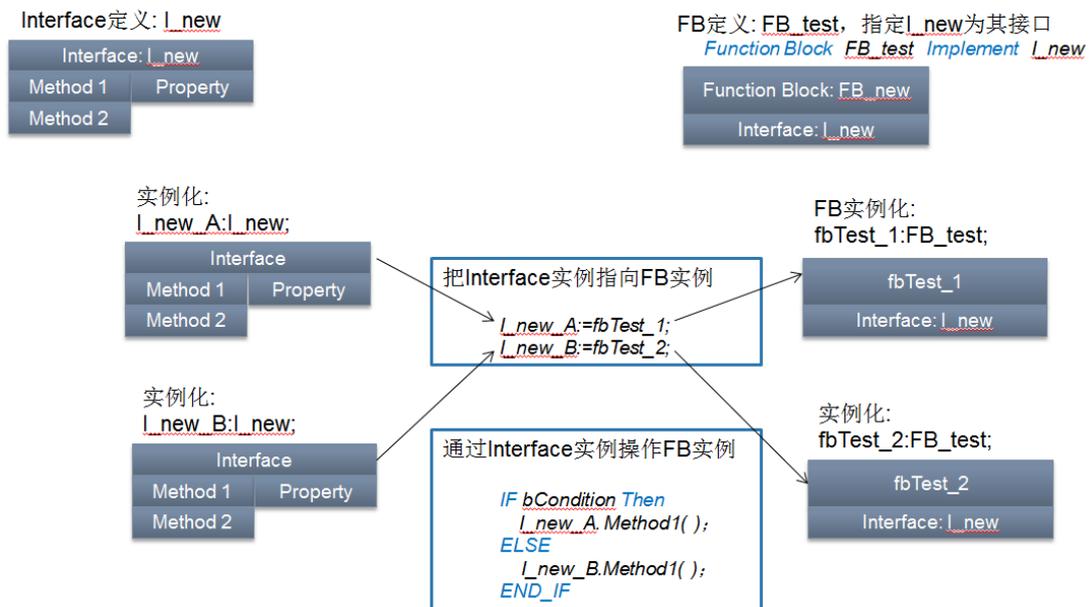
### FB 与 Interface 的关系：

FB 可以有一个或者多个 Interface，也可以不用 Interface。但 Interface 只有被一个或者多个 FB 引用才有意义。

Interface 和 FB 一样，也需要实例化才能使用。但是 FB 的实例占用一块内存，而 Interface 的实例只是一个指针。如果不指向某个地址，或者指向地址存放的内容与 Interface 定义不匹配，这个指针就无效。简言之，**FB 是传值的，而 Interface 是传址的。**

可以通过 Interface 的实例操作 FB 的实例，前提是这个 Interface 实例指向了这个 FB 实例，相当于把 Interface 指针向了 FB 实例接口的内存区。这个原理类似于在 TC2 中通过指针操作它所指向的地址中的变量值。

### FB 与 Interface 的关系图：



用户可以基于一个 Interface 建立多个 FB，而它们的内部代码各不相同。实际上，也只有需要重复利用的接口，才是 OOP 编程里面特指的 Interface。因为假如一个 Interface 只用一次，就没必要为此专门建一个 Interface。

## 5.1.3. 关键词：Extend

TwinCAT 3 编程里面新增了一个关键的动词：继承 (*Extend*)。与 Extend 紧密相关的关键词是 *This* 和 *Super*。

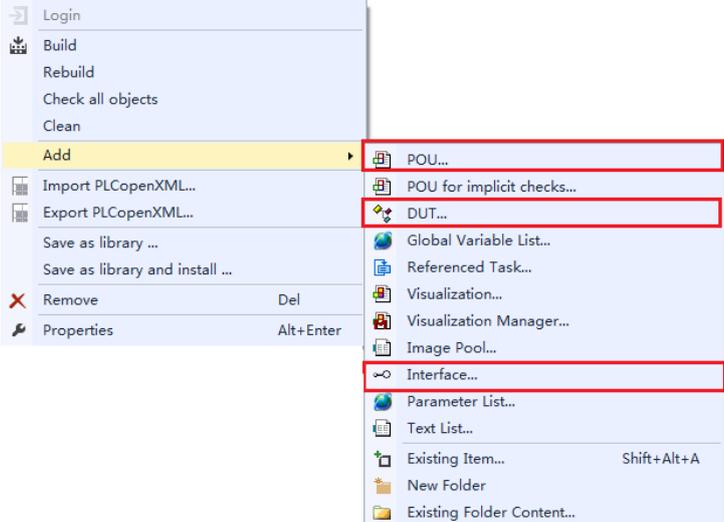
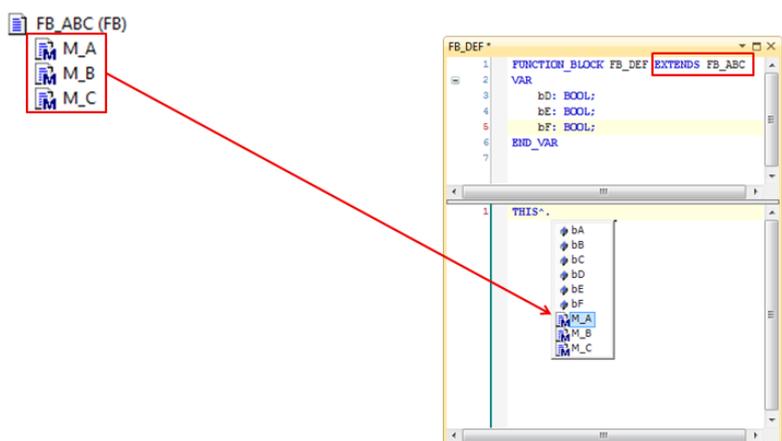
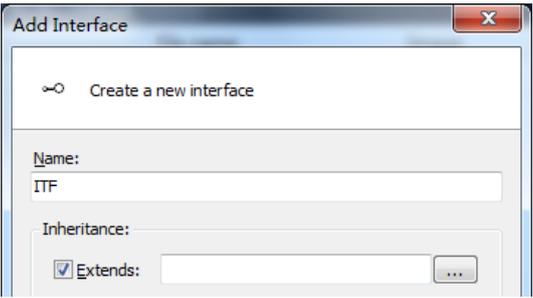
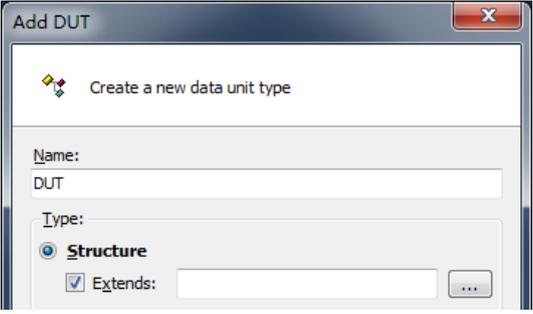
可以使用 Extend 功能的对象包括：

- a) Function Block

b) Interface

c) Structure

在 TC3 的 PLC 项目中添加这 3 种元素时，会有 Extends 选项出现：

<p>以 Extends 方式建立对象时，都要求指定 Extends Base 对象，即父对象。</p>	
<p>以 Extends 方式建立的 FB，继承了父对象所有的 Method、Property、内部变量和代码。</p>	
<p>以 Extends 方式建立的 Interface，继承了父对象所有的 Method、Property。</p>	
<p>以 Extends 方式建立的 Structure，继承了父对象所有的元素。</p>	

使用 `Extends`，可以由一个简单的对象，派生出若干复杂的对象。因为有了继承的关系，对象和父对象之间，就有了 `This` 和 `Super` 的说法。`This` 特指对象本身，而 `Super` 指它的父对象。当继承的 `Method`、`Property` 或者变量做了重置，对象和它的父对象下的同名元素（`Method`、`Property` 或者变量）就有了不同的含义，此时就需要前置 `This` 和 `Super` 来区分。如果没有前置这两个关键词，默认使用本对象的定义。

## 5.1.4. 面向对象编程的 3 个用法

面向对象编程总是指 FB 的写法。总结起来，在 TC3 中创建功能块时有 3 个方式：

- d) 建立一个空白的 FB
  - 定义 `Method` 和 `Property`，就是 OOP 编程。
  - 没有 `Method` 和 `Property`，仅定义 `Input` 和 `Output` 变量，就是传统的 PLC 编程。
- e) 建立一个基于 `Interface` 的 FB，使用预定义的 `Method` 和 `Property`。
- f) 建立一个 FB 的扩展 FB（`Extend`），继承所有 `Method`、`Property` 和代码。

使用 `Method` 和 `Property` 是面向对象编程（OOP）的基本特征。而 `Interface` 和 `Extend` 则是扩展应用。灵活运用这些新功能，可以大大提高程序的安全性、可移植性和可读性。后面的章节就详细说明这 3 种应用。

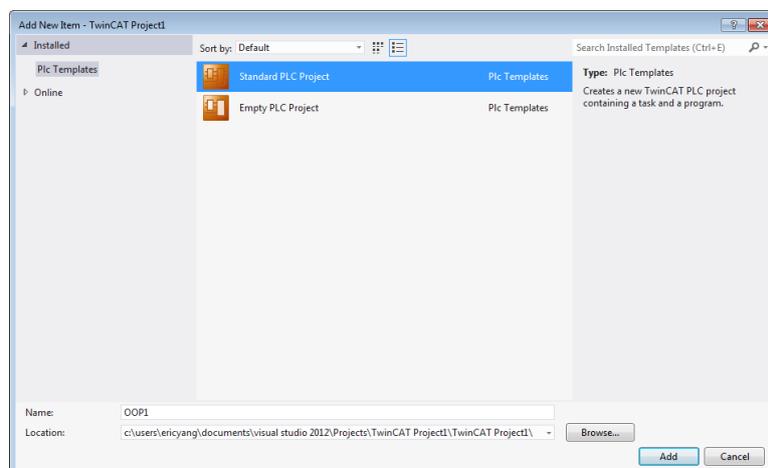
## 5.2. 建立一个带 `Method` 和 `Property` 的 FB

### 5.2.1. 示例

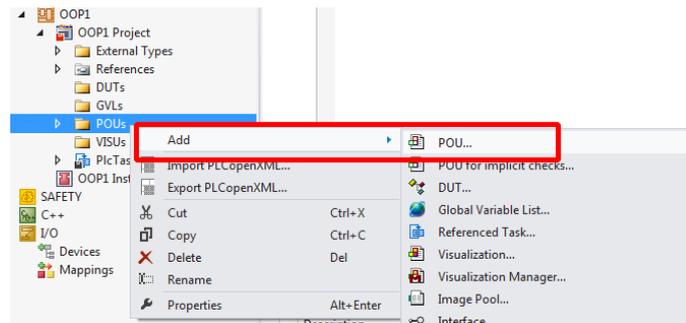
以下摘自“\TwinCAT 3 入门教材\11 TC3 第十一章 TwinCAT3-OOP 编程”，作者：杨煜敏。

#### 1: `method`, `property`, `super`, `extends` 的用法

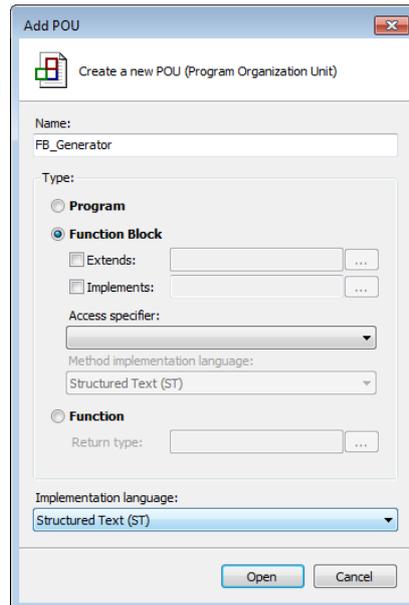
- (1) 首先还是新建 PLC 程序，并且取名为 `OOP1`。



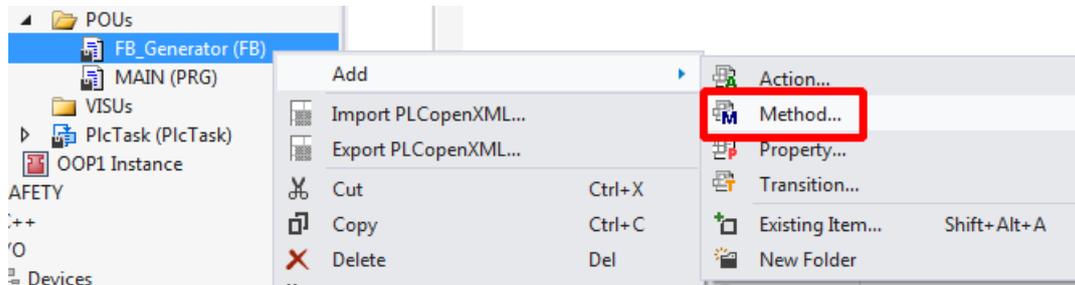
- (2) 右键 `POUs` 添加 `POU`。



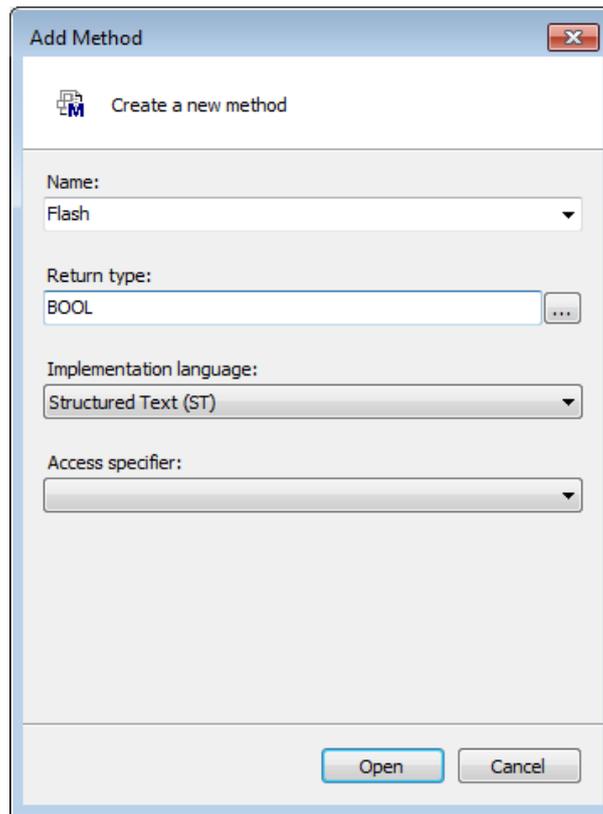
(3) 创建功能块，并且取名为：FB\_Generator，编写语言是 ST。



(4) 右键功能块 Add Method。



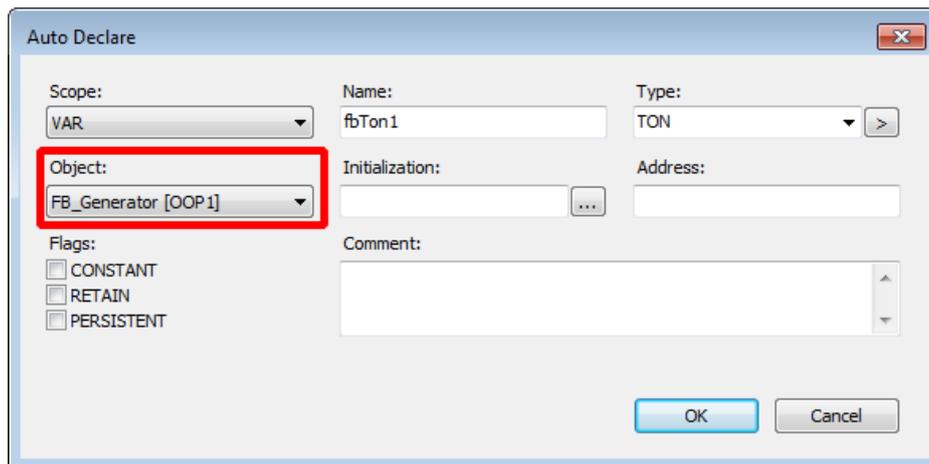
(5) 修改 Method 名为 Flash，并且选择返回类型为 BOOL。



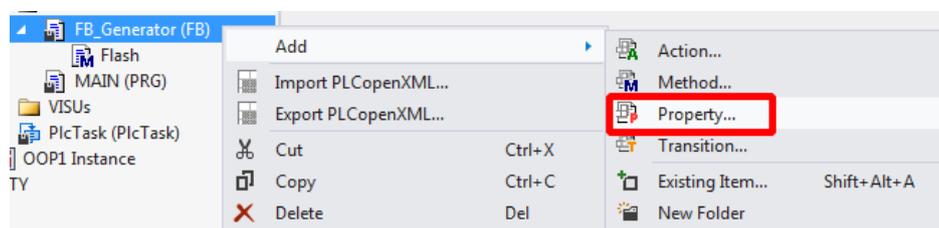
(6) 在 Flash 这个 Method 中编写程序。

```
FB_Generator.Flash  # X
1  METHOD Flash : BOOL
2  VAR_INPUT
3  END_VAR
4
5
6
7  fbTon1(IN:=NOT fbTon1.Q , PT:=_tCycletime , Q=> , ET=> );
8
9  IF fbTon1.Q THEN
10     bFlashVar := NOT bFlashVar;
11 END_IF
12
13 Flash := bFlashVar;
```

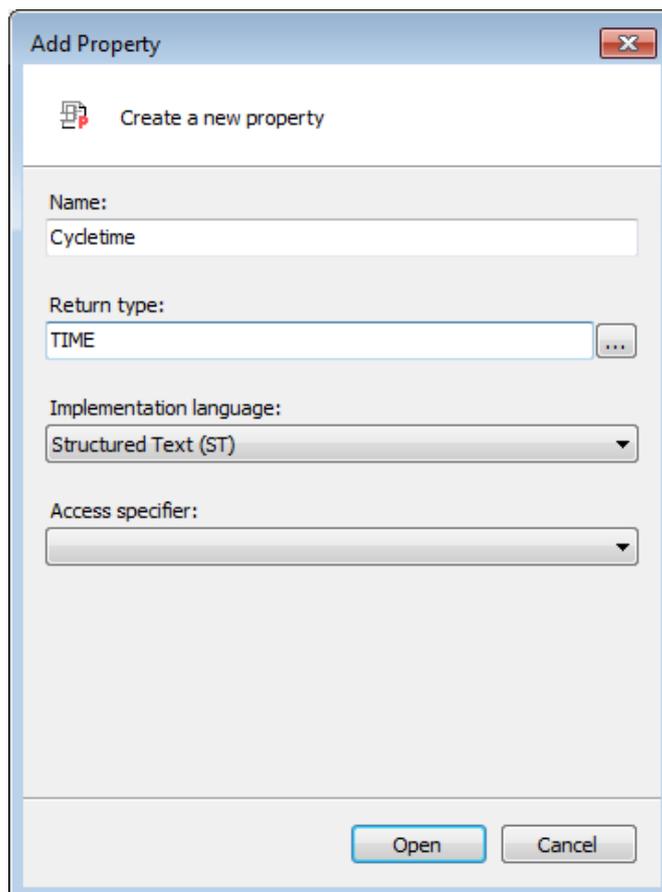
(7) 并且注意所有的变量声明都声明在 FB\_Generator 中，而不是 Flash 中，如下图所示，Object 都选择 FB\_Generator。



(8) 右键功能块 Add Property。

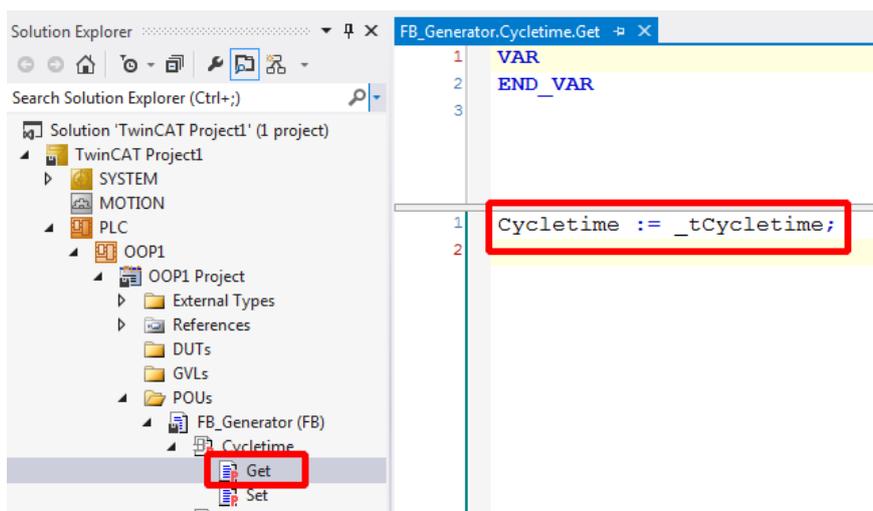


(9) 修改 property 名为 Cycletime，返回类型设置为 TIME。

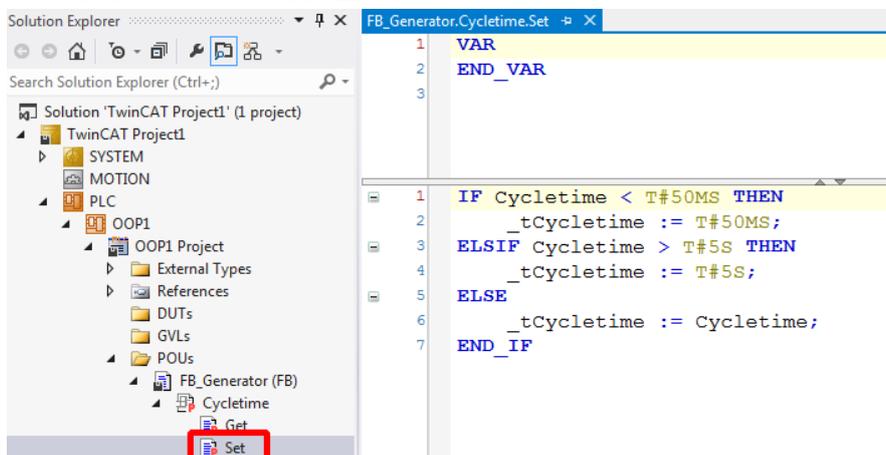


(10) 分别对应 property 中 2 个 method, set 和 get 进行编程使得 property 所对

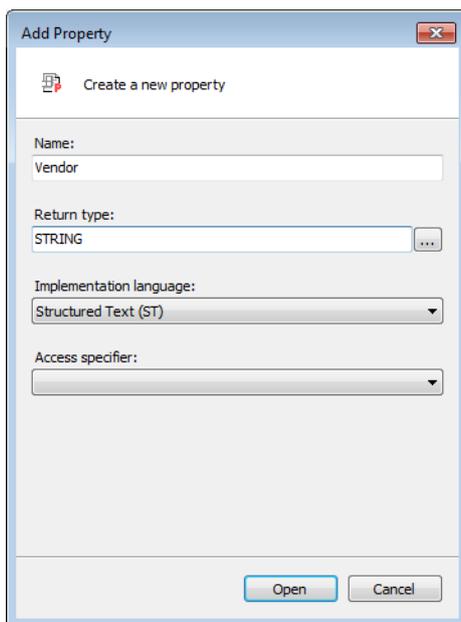
应的变量可读可写。



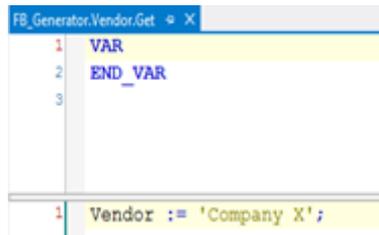
(11) Set 可以设限，让写更安全。



(12) 随后再添加一个 property，并且设置 property 名为 Vendor，返回类型设置为 STRING。



(13) Vendor.Get 代码编写

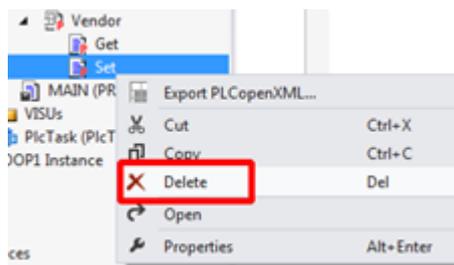


```

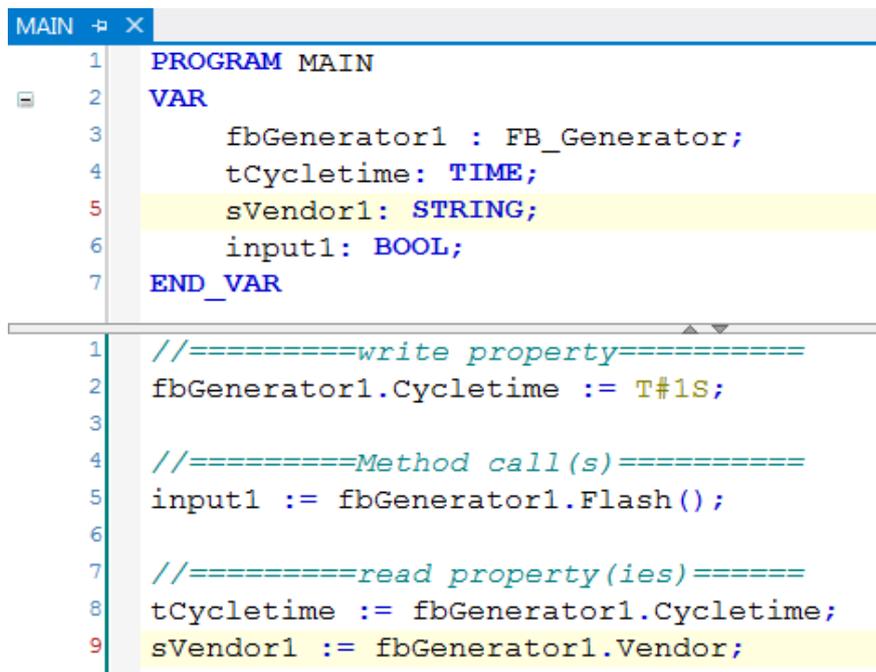
1  VAR
2  END_VAR
3
1  Vendor := 'Company X';

```

(14) 因为 vendor 只是一个标签，所以只读不能写，因此把 set 给删除，这样 vendor 这个属性就不支持写



(15) 这样一个简单的类就创建好了，里面有 1 个 method 和 2 个 property，我们可以先在 MAIN 程序中实例化这个类模拟一下，



```

1  PROGRAM MAIN
2  VAR
3      fbGenerator1 : FB_Generator;
4      tCycletime : TIME;
5      sVendor1 : STRING;
6      input1 : BOOL;
7  END_VAR
8
9  //=====write property=====
10 fbGenerator1.Cycletime := T#1S;
11
12 //=====Method call(s)=====
13 input1 := fbGenerator1.Flash();
14
15 //=====read property(ies)=====
16 tCycletime := fbGenerator1.Cycletime;
17 sVendor1 := fbGenerator1.Vendor;

```

(16) 运行后可以发现读取到了 property 的 2 个变量。

Expression	Type	Value	Prepared value	Address	Comment
fbGenerator1	FB_Generator				
tCycletime	TIME	T#1s			
sVendor1	STRING	'Company X'			
input1	BOOL	TRUE			

```

1 //=====write property=====
2 fbGenerator1.Cycletime := T#1S;
3
4 //=====Method call(s)=====
5 input1 TRUE := fbGenerator1.Flash();
6
7 //=====read property=====
8 tCycletime T#1s := fbGenerator1.Cycletime;
9 sVendor1 'Company X' := fbGenerator1.Vendor; RETURN

```

(17) 并且通过 scope view 观察变量 input1 为 1 秒循环闪烁。



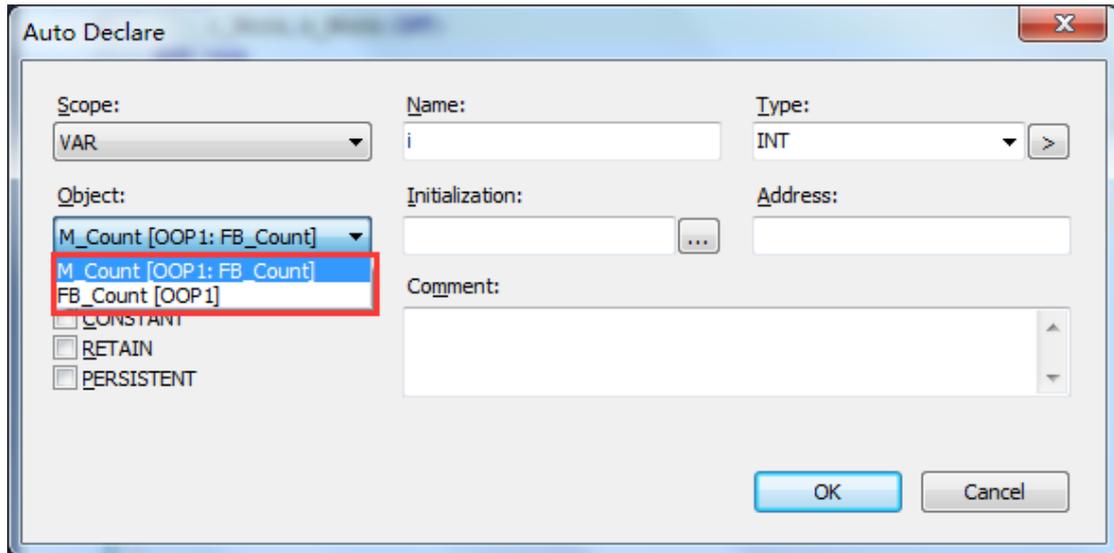
## 5.2.2. 关于 Method 和 Property 的 FAQ

- Method

Method 必须有自己的接口变量(Input, Output, IN\_Out)

Method 可以有反馈变量, 也可以没有。当它使用反馈变量的时候, 类似 Function。

在 Method 中变量声明时, 要选择是 Method 的变量还是所在 FB 的变量:



Method 中的代码可以使用 Method 变量、所在 FB 的局部变量以及项目的全局变量。

Method 变量，如果不声明为 Static，默认都是临时变量，每次调用它的时候都会初始化。

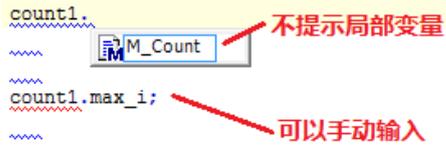
无论接口变量还是内部变量都不出现在 FB 的实例内部变量中：

TwinCAT_Device.OOP1.MAIN			
Expression	Type	Value	P
Count1	FB_Count		
i	INT	749	
max_i	INT	755	
k	INT	2258	
max_k	INT	2265	
Count2	FB_Count_ex		
i	INT	302	
max_i	INT	755	
k	INT	302	
max_k	INT	2265	
Startcount	BOOL	TRUE	

#### ● Property 说明

Property 表示对象的特征值，包括状态、参数等。在 FB 中添加一项 Property 之后，就自动出现了 Get 和 Set 的 Action，里面可以编辑属性与内部变量之间的关系。Get( )里面必须编写代码，如果只有 Get( )没有 Set( )代码，表示这是“Read Only”的属性。

外部编写代码访问 FB 的实例的内部变量时，不能通过“.”取出内部变量了，只有 Method 和 Property 出现在提示框中：



从外部给 FB 实例的属性赋值，用 “:=”。

两个属性可以对应 1 个局部变量，比如属性 1（重量：克），属性 2（重量：千克），那么属性 1 的值为 1000 时，属性 2 的值为 1.0。

## 5.3. 建立一个 FB 的扩展 FB（Extend）

### 5.3.1. 示例

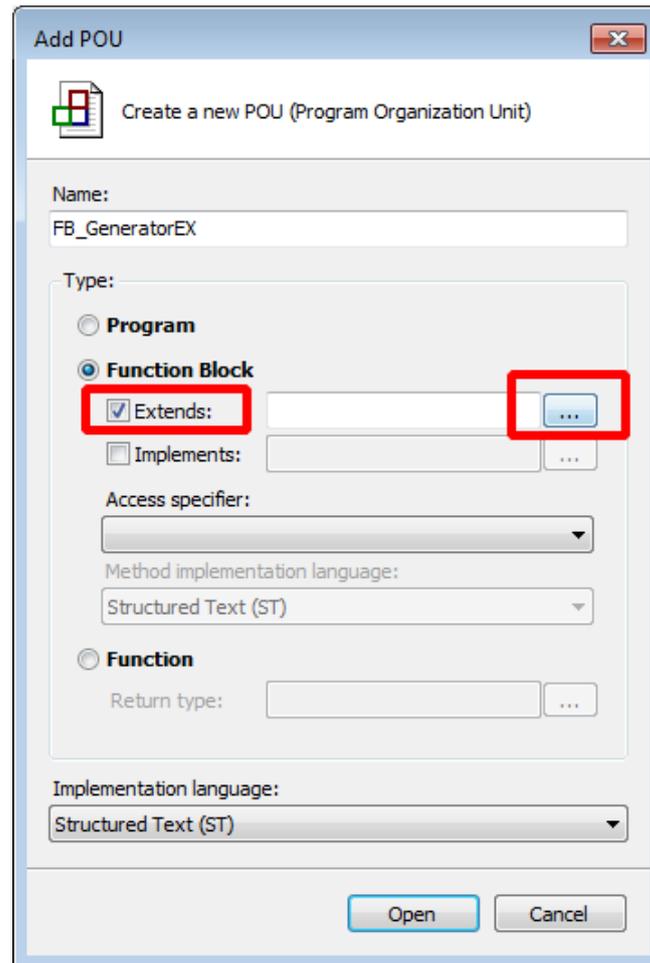
在上一节例子的基础上，增加 Extend 功能，将上例建议的 FB 作为父对象。

以下摘自 “\TwinCAT 3 入门教材\11 TC3 第十一章 TwinCAT3-OOP 编程”，作者：杨煜敏。

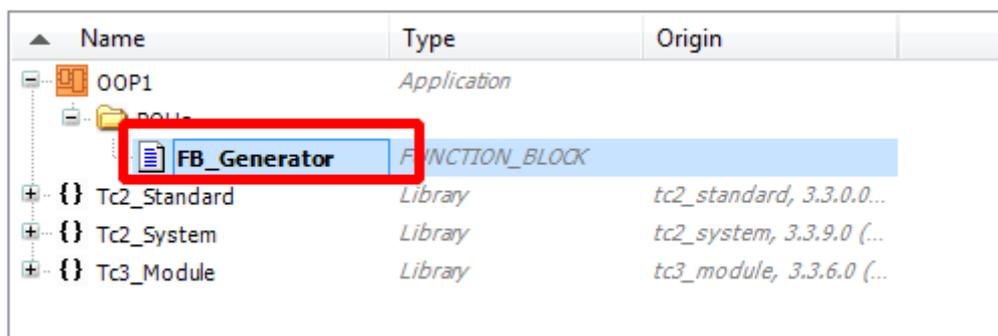
#### 1: method, property, super, extends 的用法

（18）紧接着来看下 extends 和 super 的使用。

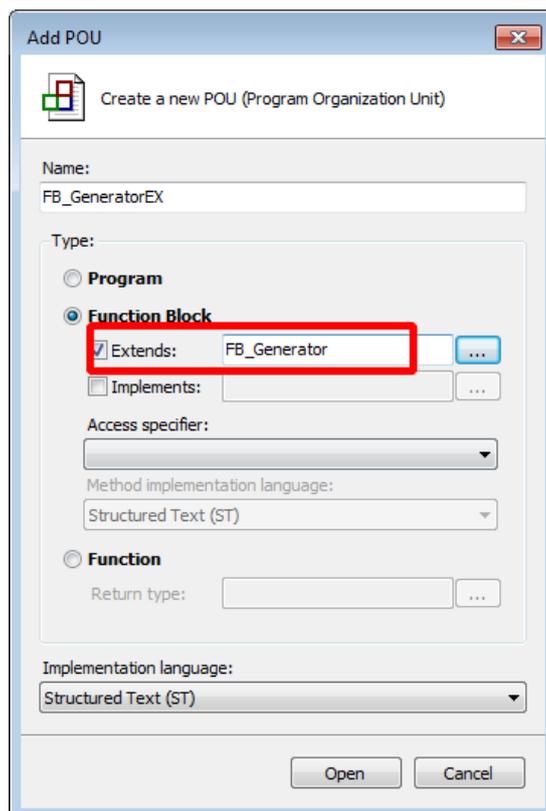
新建功能块，并且取名为 FB\_GeneratorEX，勾选 Extends，点击选项框。



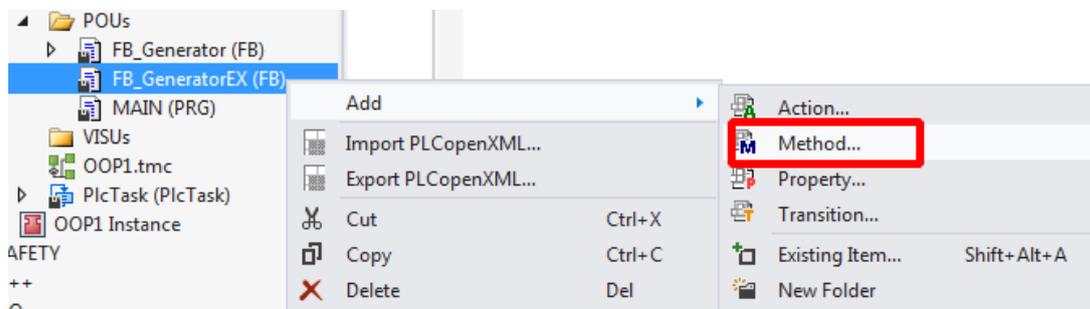
(19) 在选项框中找到被扩展的功能块 FB\_Generator。



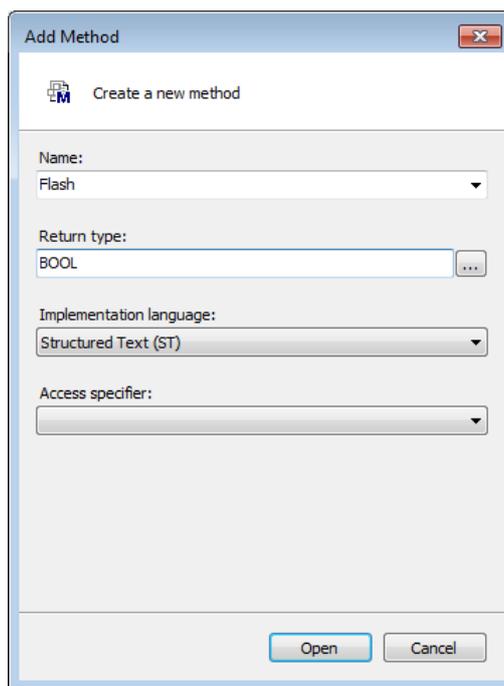
(20)选中后点击 OK 可以发现被扩展功能块名出现在 Extends 框中,点击 Open。



(21) 功能块 FB\_GeneratorEX 已经继承了 FB\_Generator 所有变量, 方法和属性, 我们可以对于 FB\_GeneratorEX 功能块进行添加新的方法和属性进行扩展, 当然也可以对于所继承的原有的方法和属性进行重写, 接下来就演示如何进行重写右键 FB\_GeneratorEX 新建 Method。



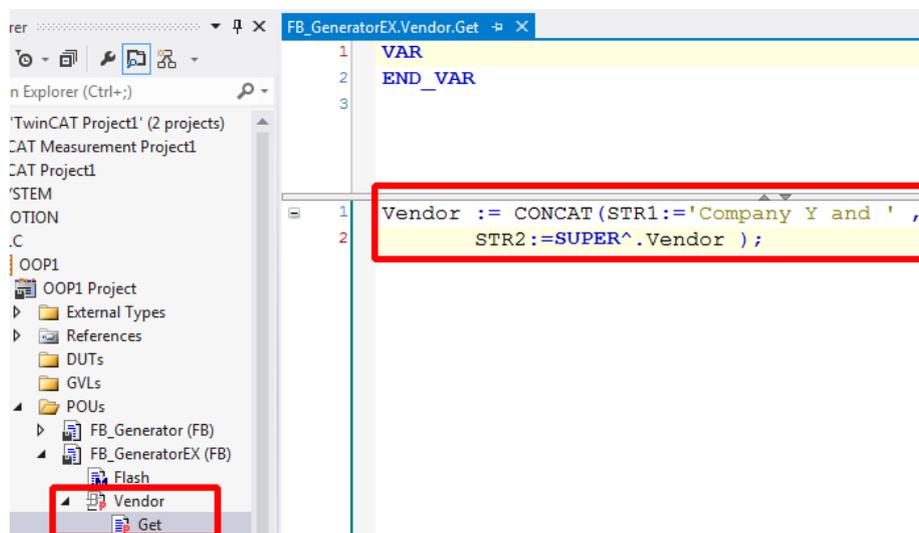
(22) 修改此 Method 名也为 Flash, 并且返回类型也是 BOOL。



(23) 随后重写编写 Flash 代码，实现的方法变为 1/3 为 true，2/3 为 false。

```
FB_GeneratorEX.Flash  ▸ ×
1  METHOD Flash : BOOL
2  VAR_INPUT
3  END_VAR
4
1  fbTon1(IN:=NOT fbTon1.Q , PT:=_tCycletime , Q=> , ET=> );
2
3  Flash := fbTon1.ET < fbTon1.PT / 3 ;
```

(24) 再重写 vendor 这个属性，重写 vendor.get，其中使用到 super 可以直接访问到被扩展功能块 FB\_Generator 中的 Vendor，并且使用 CONCAT 函数进行字符串合并。



(25) 重写一个方法和一个属性完成后，在程序中实例化这个功能块来观察效果在 MAIN 程序中只需要修改功能块声明和 `cycletime` 写入值为 3S 即可：

```

MAIN  ▸ ×
1  PROGRAM MAIN
2  VAR
3  fbGenerator1 : FB_GeneratorEX;
4  tCycletime: TIME;
5  sVendor1: STRING;
6  input1: BOOL;
7  END_VAR

1  //=====write property=====
2  fbGenerator1.Cycletime := T#3S;
3
4  //=====Method call(s)=====
5  input1 := fbGenerator1.Flash();
6
7  //=====read property(ies)=====
8  tCycletime := fbGenerator1.Cycletime;
9  sVendor1 := fbGenerator1.Vendor;

```

(26) 下载程序并且 login 后可以观察到 vendor 读取到的字符串为 Company Y and Company X。

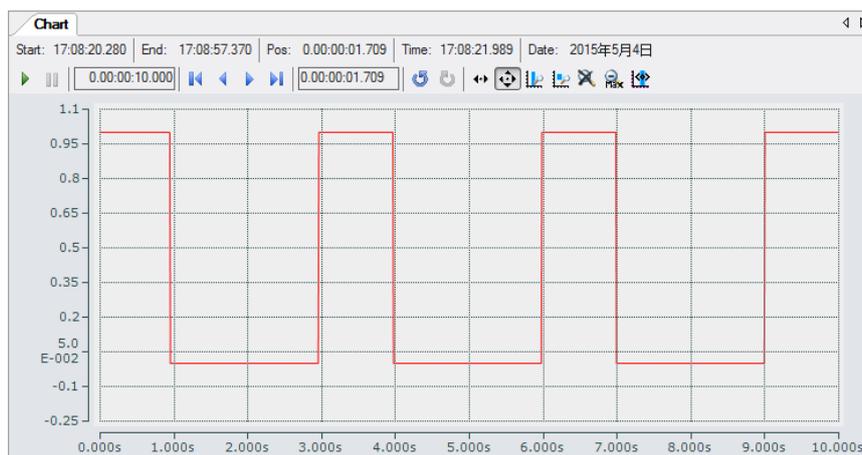
Expression	Type	Value	Prepared value	Address	C
fbGenerator1	FB_GeneratorEX				
tCycletime	TIME	T#3s			
sVendor1	STRING	'Company Y and Company X'			
input1	BOOL	TRUE			

```

1  //=====write property=====
2  fbGenerator1.Cycletime := T#3S;
3
4  //=====Method call(s)=====
5  input1 TRUE := fbGenerator1.Flash();
6
7  //=====read property(ies)=====
8  tCycletime T#3s := fbGenerator1.Cycletime;
9  sVendor1 'Company Y' := fbGenerator1.Vendor; RETURN

```

(27) 并且通过 scope view 观察到 input1 为 1 秒 true，2 秒 false。



## 5.3.2. 关于 Extend 的 FAQ

- 功能块的继承：包括 FB 下的静态变量、Method、Property、Interface 都可以继承。
- 继承和重置  
父对象的 Method，子对象可以原封不动地继承使用，也可以重新编写代码重置该方法。引用时，不作特别说明，即是指重置后的 Method。
- This 和 Super  
重置过的 Method，就会出现同样名字，但父对象和子对象含义不同的情况。某此时候如果要引用的是父对象中 Method 的代码功能，就要前置 Supper.Method 来实现。

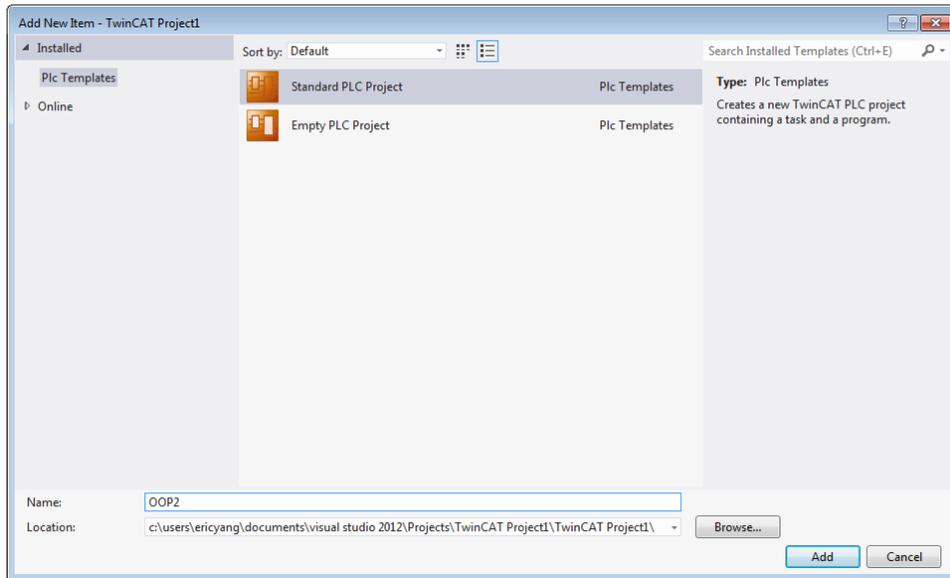
## 5.4. 建立一个 Interface 并实现 (Impement)

### 5.4.1. 示例

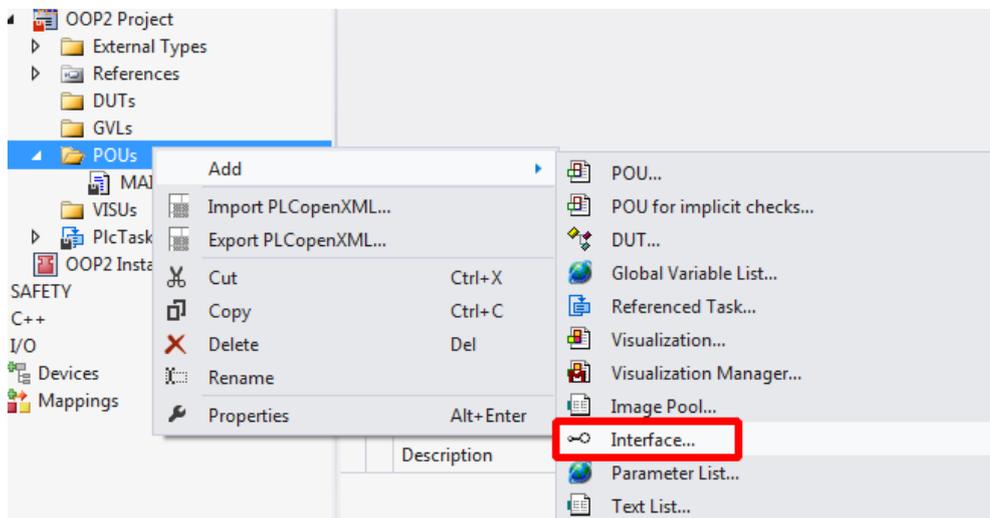
以下摘自“\TwinCAT 3 入门教材\11 TC3 第十一章 TwinCAT3-OOP 编程”

2：interface 和 implements 用法介绍。

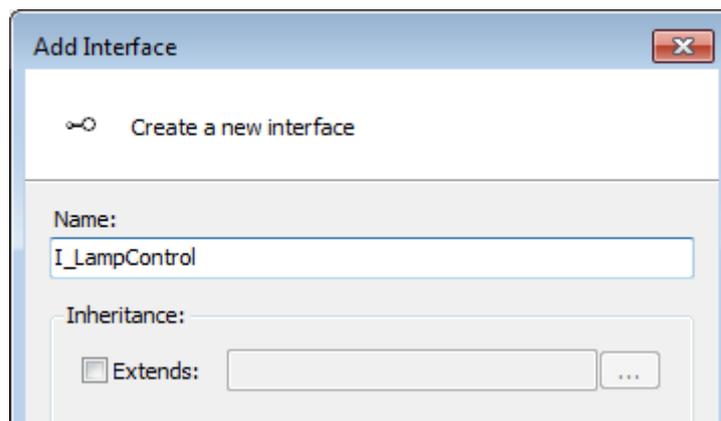
(1) 新建 PLC 项目，取名为 OOP2。



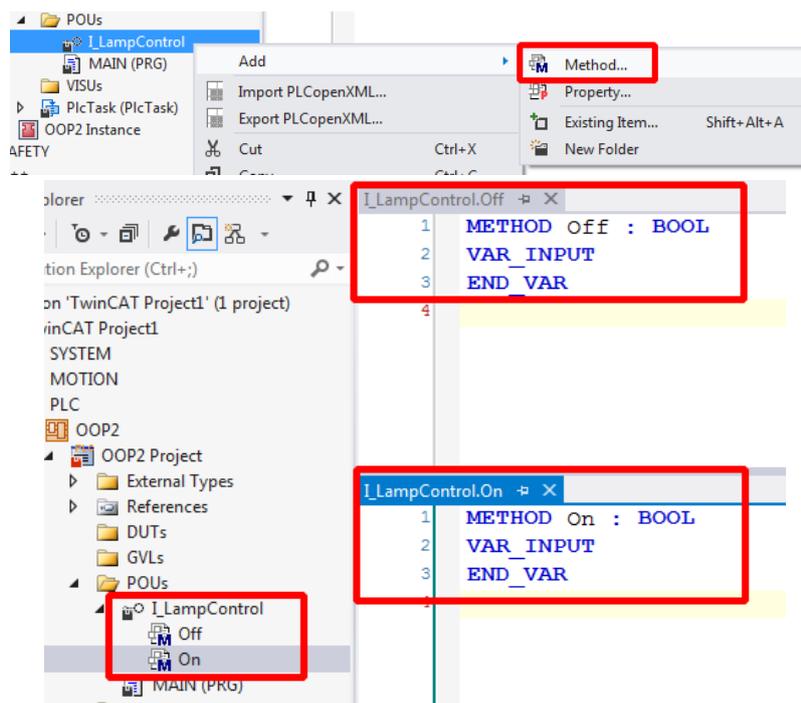
(2) 右键 POU 添加 Interface。



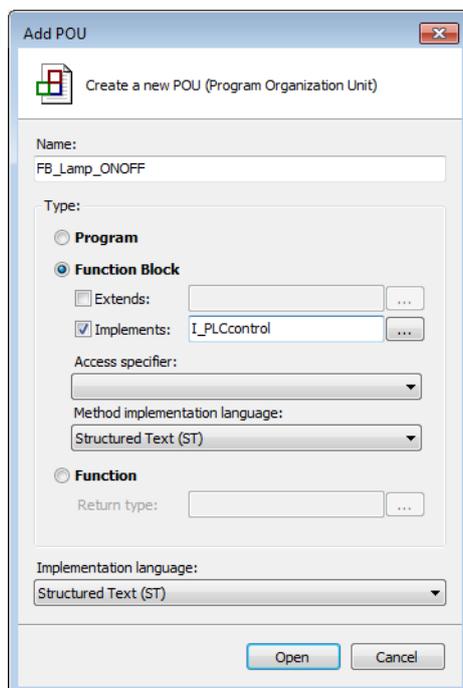
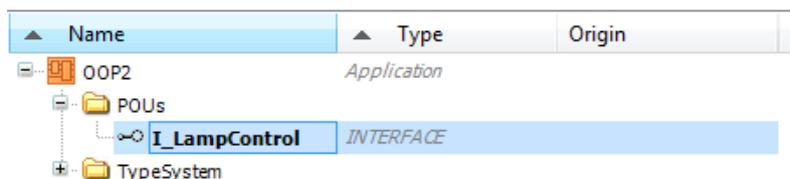
(3) 取名为 I\_LampControl 并且点击 OK 进行创建。



(4) 右键 I\_LampControl 添加 2 个方法，on 和 off 分别代表灯的 2 个功能，开灯和关灯，返回类型是 BOOL。



(5) 接口创建好了后开始创建功能块实现这些接口并且编写实现代码，右键 POU 新建功能块，取名功能块为 FB\_Lamp\_ONOFF，并且勾选 Implements，在选项框选择之前创建好的接口 I\_LampControl。



(6) 完成实现接口后就可以在 On 和 Off 中写实现代码，当然所有变量都需要声

明在 FB\_Lamp\_ONOFF 中，而不是 method 中。

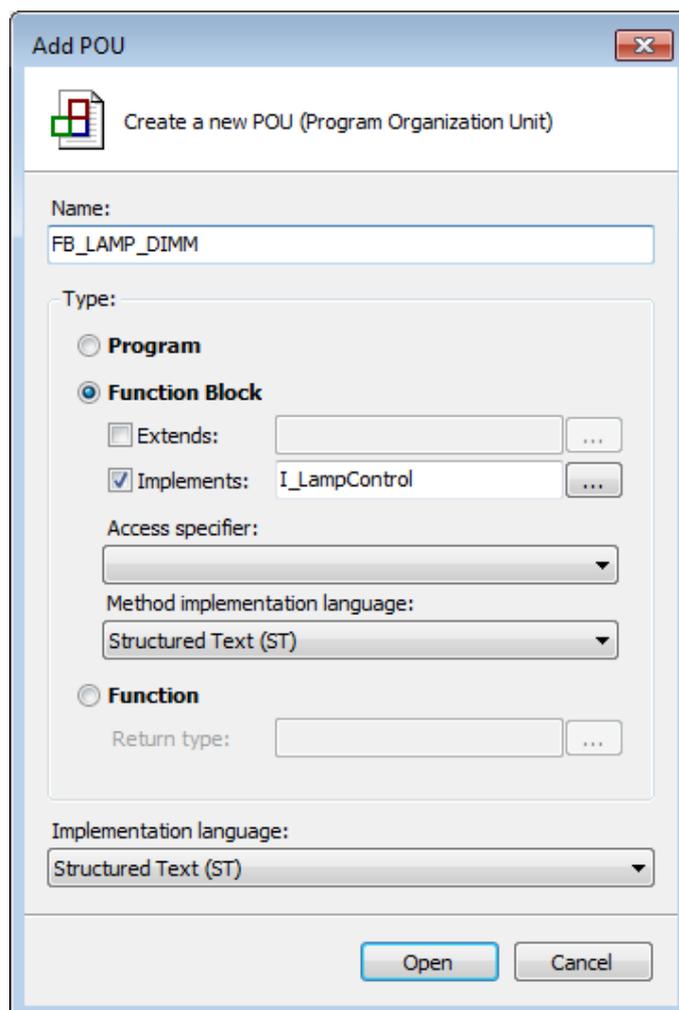
```

FB_Lamp_ONOFF.On
1 {attribute 'object_name' := 'On'}
2 METHOD On : BOOL
3
1 _bState := TRUE;
2 // Output
3 bOutHW := TRUE;

FB_Lamp_ONOFF.Off
1 {attribute 'object_name' := 'Off'}
2 METHOD off : BOOL
3
1 _bState := FALSE;
2 // Output
3 bOutHW := FALSE;

FB_Lamp_ONOFF
1 FUNCTION_BLOCK FB_Lamp_ONOFF IMPLEMENTS I_LampControl
2 VAR_INPUT
3 END_VAR
4 VAR_OUTPUT
5 END_VAR
6 VAR
7     _bState: BOOL;
8     bOutHW AT%Q* :BOOL;
9 END_VAR
  
```

(7) 接下来再见一个功能块实现这个接口，修改功能块名为 FB\_LAMP\_DIMM。



(9) 接着编写 On 和 Off 中实现代码。

```
FB_LAMP_DIMM.On  ×
1 {attribute 'object_name' := 'On'}
2 METHOD On : BOOL
3

FB_LAMP_DIMM.On
1 _bState := TRUE;
2 _fOut :=_fOut + _fInc;
3 IF _fOut > 100.0 THEN
4   _fOut :=100.0;
5 END_IF
6 iOutHw := LIMIT (0,REAL_TO_INT(_fOut),100);
7

FB_LAMP_DIMM.Off  ×
1 {attribute 'object_name' := 'Off'}
2 METHOD Off : BOOL
3

FB_LAMP_DIMM.Off
1 _bState := FALSE;
2 _fOut :=_fOut - _fInc;
3 IF _fOut < 0.0 THEN
4   _fOut :=0.0;
5 END_IF
6 iOutHw := LIMIT (0,REAL_TO_INT(_fOut),100);

FB_LAMP_DIMM  ×
1 FUNCTION_BLOCK FB_LAMP_DIMM IMPLEMENTS I_LampControl
2 VAR
3   _bState: BOOL;
4   _fOut: REAL;
5   _fInc: REAL := 0.5;
6   iOutHw AT %Q*: INT;
7 END_VAR
```

(9) 因此可以发现虽然是实现同一个接口，但实现代码可以截然不同，随后在程序中就可以利用接口的实例化实现多个对象的切换

```

MAIN # x
1 PROGRAM MAIN
2 VAR
3     fbLamp1 : FB_Lamp_ONOFF;
4     fblamp2Dimm : FB_LAMP_DIMM;
5     ipLampControll : I_LampControl;
6     change : BOOL;
7     bOn : BOOL;
8 END_VAR

1 IF change THEN
2     ipLampControll := fbLamp1;
3 ELSE
4     ipLampControll := fblamp2Dimm;
5 END_IF

7 IF bOn THEN
8     //fbLamp1.On();
9     ipLampControll.On();
10 ELSE
11     //fbLamp1.Off();
12     ipLampControll.Off();
13 END_IF

```

(10) 如果对象很多，当然也可以通过创建一个接口数组，并且利用 FOR 循环实现一条语句对于所有对象进行操作。

```

1 PROGRAM MAIN_OPP2
2 VAR CONSTANT
3     nNROFLAMPS : UINT := 4;
4 END_VAR
5 VAR
6     fbLamp1 : FB_Lamp_ONOFF;
7     fbLamp2 : FB_Lamp_ONOFF;
8     fbDimmerLamp1 : FB_LAMP_DIMM;
9     fbDimmerLamp2 : FB_LAMP_DIMM;
10    iLamp : ARRAY[0..(nNROFLAMPS - 1)] OF I_LampControl;
11    ni: INT;
12    bSwitch: BOOL;
13 END_VAR

1 ilamp[0]:= fbLamp1;
2 ilamp[1]:= fbLamp2;
3 ilamp[2]:= fbDimmerLamp1;
4 ilamp[3]:= fbDimmerLamp2;
5
6 FOR ni := 0 TO nNROFLAMPS -1 DO
7     IF ilamp[ni] <> 0 THEN
8         IF bSwitch THEN
9             ilamp[ni].On();
10        ELSE
11            ilamp[ni].Off();
12        END_IF
13    END_IF
14 END_FOR

```

## 5.4.2. 关于 Interface 的 FAQ

- 接口的内容

一个接口，用于不同的功能块。

接口只有变量、方法、属性、临时变量和接口变量，但没有代码和静态变量。

- 接口和 FB 的实例化

接口是一个内存区

```
PROGRAM MAIN
VAR
    Count1:FB_Count;
    Count2:FB_Count_ex;
    iCount:I_Count;
```

- 指针的确定

```
IF NOT input THEN
    iCount:=Count1;
ELSE VAR MAIN.iCount : I_Count
    iCount:=Count2;
```

- 可以用一个接口，处理所有的内象。

```
13 iCount.M_Count(bEnable:= Startcount, i_Work=>i , k_Work=>k );
14 iCount.P_Max_i:=100;
15 max_i:=icount.P_Max_i;
16 iCount.P_Max_k:=300;
17 max_k:=icount.P_Max_k;
18
```

- 建 FB 时选择 Implement 的含义

建 FB 时可以选择 Extend 及 Implement，后者表示基于接口去写代码和定义局部变量

```
1 FUNCTION_BLOCK FB_Count IMPLEMENTS I_Count
2 VAR_INPUT
3 END_VAR
4 VAR_OUTPUT
5 END_VAR
6 VAR
7     i: INT;
8     max_i: INT := 755;
9     k: INT;
10    max_k: INT := 2265;
11 END_VAR
12
```

这里表明了 FB 和 Interface 关联。即使代码相同，变量相同，有或者没有 Implement，在于是否允许指针赋值：

```

1  PROGRAM MAIN
2  VAR
3      Count1:FB_Count;
4      Count2:FB_Count_ex;
5      iCount:I_Count;
6
7  //Count2.M_Count (bEnable:=Startcount, i_Work=>i , k_Work=>k );
8  //bCheck:=Count2.M_Check (bCheckValue:=bCheckValue );
9
10 //Count2.P_Max_i:=100;
11 //Count2.P_Max_k:=300;
12
13 IF NOT input THEN
14     iCount:=Count1;
15 ELSE
16     iCount:=Count2;
17     bCheck:=Count2.M_Check (bCheckValue:=bCheckValue );
18 END_IF

```

如果没 Impement 语句，此处代码为非法。

## 5.5. 其它说明

### 5.5.1. 静态变量（Static）和临时变量（Temp）

VAR（局部变量）、VAR\_TEMP（临时变量）、VAR\_STA（静态变量），在不同 POU 下的区别。

	VAR	VAR_Temp	VAR_Sta
PRG 和 FB	有记忆	没记忆	有记忆
FC 和 Method	没记忆	每次调用都初始化	首次调用时初始化

### 5.5.2. 特殊的 Method: FB\_Init, FB\_Exit, FB\_Reinit

TwinCAT 3 中的 FB，新增了 3 个特殊的 Method: FB\_Init, FB\_Exit, FB\_Reinit。只要新建的 Method，命名为其中之一，就会在特定的时间自动执行：

FB\_init

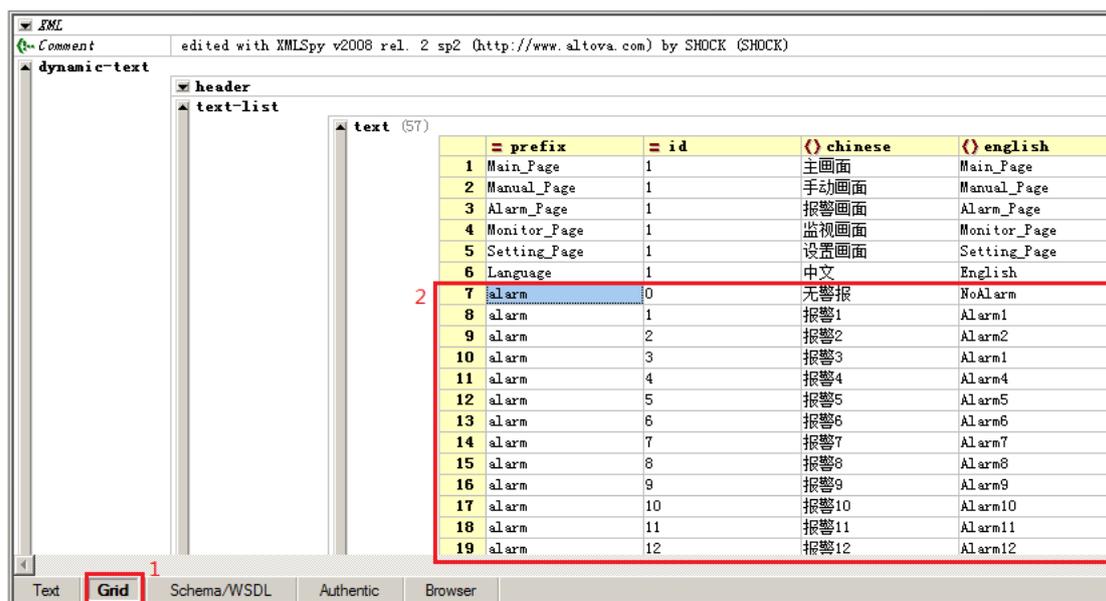
- 包括功能块的隐含初始化代码
- PLC 程序启动前执行。(SAFEOP => OP)
- 用户定义的输入

FB\_reinit

- 包括功能块的重新初始化代码。
- 功能块实例复制时（Online Change）执行。

FB\_exit

- 资源释放时的代码。
- Download、Reset 或者 Online Change 时执行。



先点击“1”处，以表格形式显示参数项。“2”处就指定了当变量为 1 到 12 时分别对应的中文字符和英文字符。

说明：实际的做法是，动态文本的所在 XML 文件总是用 DEMO 中的 XML 文件来修改。

### 13.6.6. 实例 1:显示中文报警信息

“配套文档\第 11 章\_HMI 解决方案\6 TwinCAT HMI\5\_实例 1 显示中文报警信息”

### 13.6.7. 实例 2:用户管理器

“配套文档\第 11 章\_HMI 解决方案\6 TwinCAT HMI\6\_实例 2 用户管理 EXP”

## 13.7. 实例 3:TwinCAT 3 HMI

在 PLC Control 中的 Target Visual 页面编辑的画面，除了用户开发工程师的调试界面外，还可以下载到 PLC 中全屏运行，接上显示设备，就可用作组态软件。此时需要在 PLC 上安装不同的软件包，才能支持画面全屏运行。

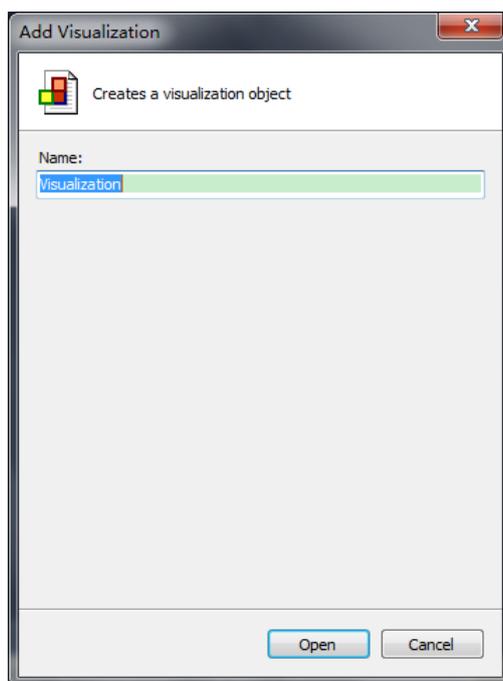
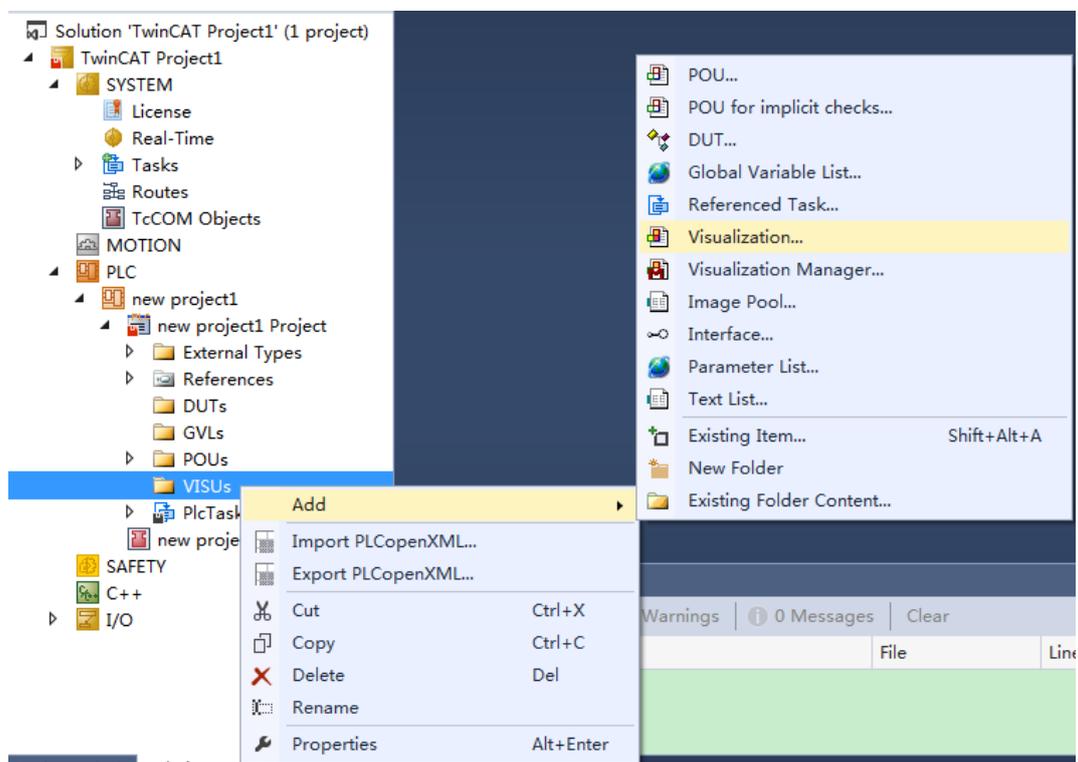
TwinCAT HMI CE，未发布。

TwinCAT HMI，安装在 Windows XP 或者 Win 7 操作系统上，接显示设备显示。

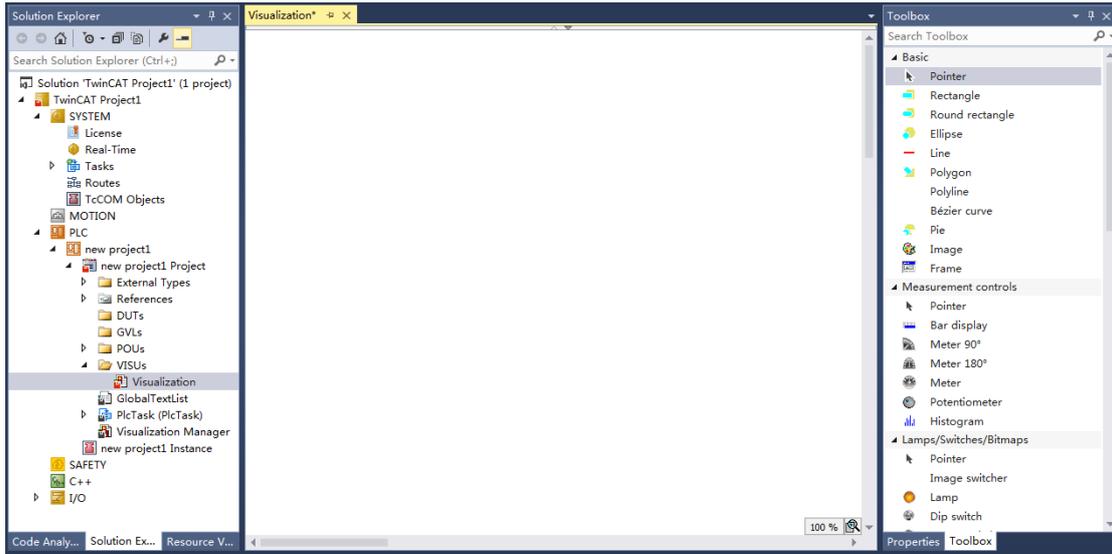
## 13.7.1. 基本操作步骤

以下摘自“\TwinCAT 3 入门教材\05TC3 第五章 TwinCAT3HMI 可视化编程”，作者：杨煜敏。

- (1) 新建一个 PLC 取名叫“new project1”把这个新项目展开找到 VISUs 并右击新建一个新的可视化取名叫“visualization”，过程如下



## (2) 新建完后 TWINCAT 会直接进入 visualization 的界面



画面最右侧会出现工具栏，其中包含两个选项卡 toolbox 和 properties，可以进行控件选择和控件属性的调整。

补充：要让可视化运行，需要一个小程序，这里变量 a 作为输入，b 作为输出。程序如下：

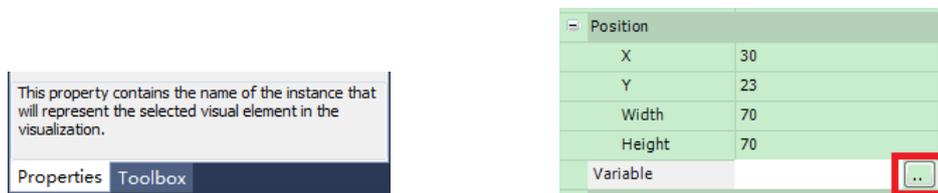
```
PROGRAM MAIN
VAR
    a   :   BOOL;
    b   :   BOOL;
END_VAR

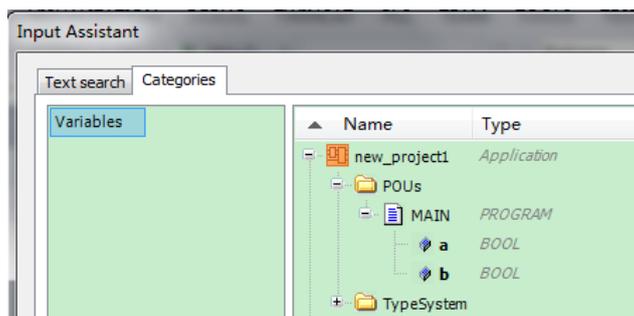
b := a;
```

## (3) 这里，我们添加一个按钮和一个位灯，对应地选择 Push switch LED 和 Lamp，



## (4) 我们单击加入的位灯，会在右侧显示它的属性栏

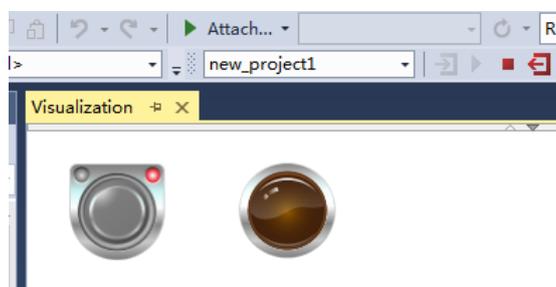
(5) 接下来我们要把程序里的变量和可视化控件进行连接。在属性栏里找到 Variable 标签，在空白栏右侧点击，在弹出的对话框里找到程序中的变量，选择 Bool 型变量 b



(6) 同样的，我们对按钮进行变量连接，在属性栏里找到 **Variable** 标签，选择 **Bool** 型变量 **a**

(7) 上述操作完成后，点击登入  Login 再运行  Start，效果如下：

初始状态



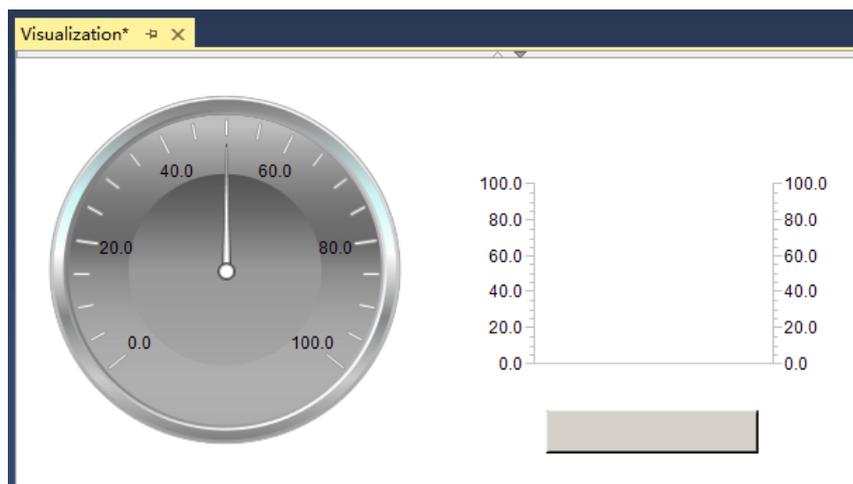
程序运行后的状态



(8) 再介绍一下关于计量仪和柱状图的添加和显示。这里需要添加三个控件，一个计量仪，一个柱状图，一个启动按钮。分别在右边 toolbox 里选用 **Measurement Controls**→**Meter**、**Histogram** 以及 **Common Controls**→**Button**



控件添加完成后显示界面如下：



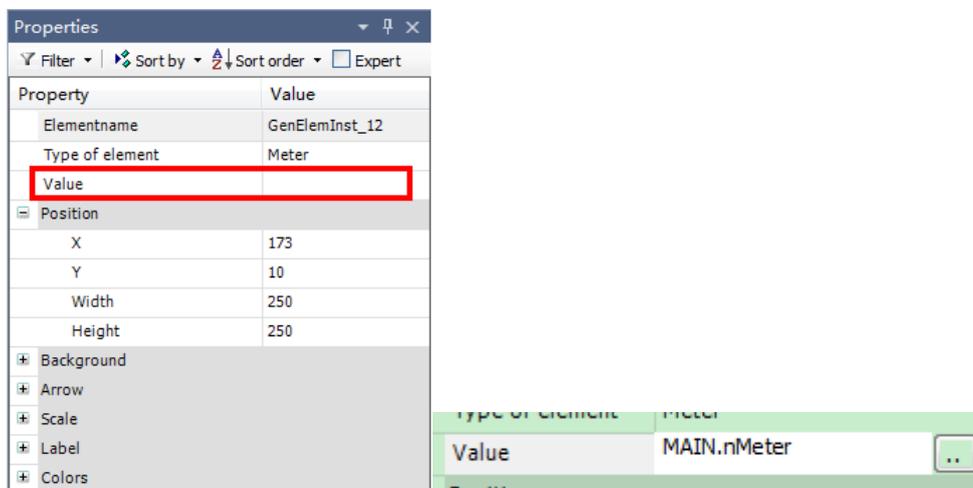
补充：这里我们同样需要一个小程序，程序如下：

```

MAIN* Visualization*
1 PROGRAM MAIN
2 VAR
3     nMeter      : INT:=0;
4     aHistogram : ARRAY[0..100] OF INT;
5     xBtn        : INT;
6     i           : INT;
7 END_VAR
8
9 IF xBtn=1 THEN
10    nMeter:=nMeter+1;
11    IF nMeter>99 THEN
12        nMeter:=0;
13    END_IF
14    FOR i:=0 TO 99 BY 1 DO
15        aHistogram[i]:=i;
16    END_FOR
17 END_IF

```

- (9) 选中计量仪，在属性栏里找到 Value 标签，在空白栏右侧点击 ，在弹出的对话框里找到程序中的变量，选择 Int 型变量 nMeter

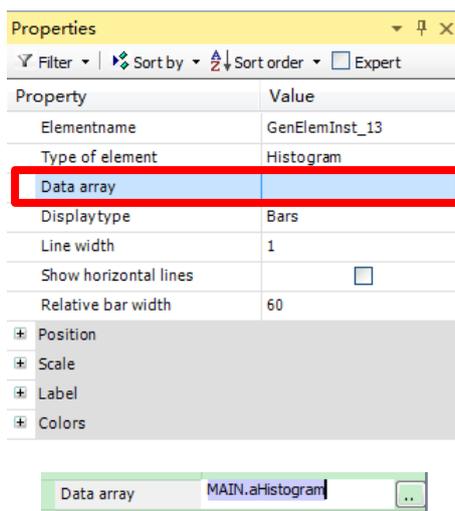


计量仪的参数是可以进行修改的，找到 Scale，展开菜单，红框中的内容是重点需要修改的

参数，这四个参数分别对应开始刻度、终止刻度、主刻度、副刻度

Scale	
Sub scale position	Outside
Scale type	Lines
Scale start	0
Scale end	100
Main scale	20
Sub scale	5
Scale line width	1
Scale color	<input type="text"/> Scalecol...
Scale in 3D	<input checked="" type="checkbox"/>
Show scale	<input checked="" type="checkbox"/>
Frame inside	<input type="checkbox"/>
Frame outside	<input type="checkbox"/>

- (10) 选中柱状图，在属性栏里找到 **Data array** 标签，在空白栏右侧点击 ，在弹出的对话框里找到程序中的变量，选择 Int 型数组变量 **aHistogram**



Properties

Filter | Sort by | Sort order | Expert

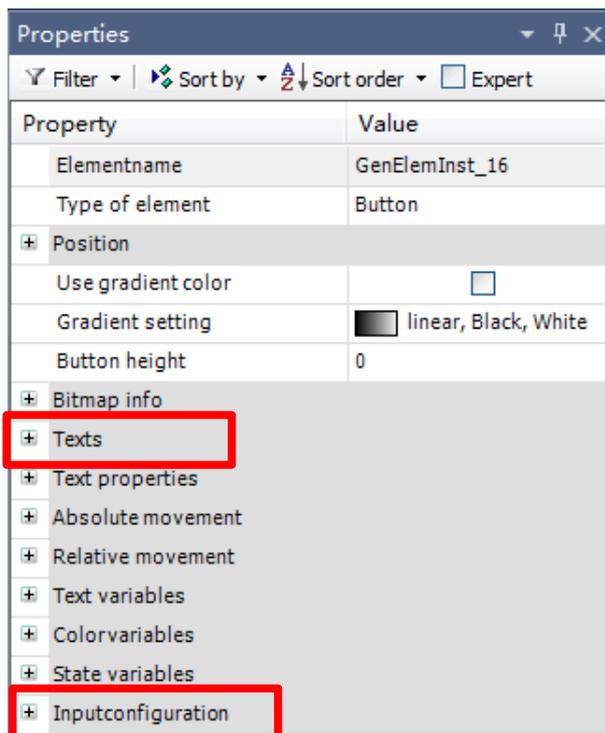
Property	Value
Elementname	GenElemInst_13
Type of element	Histogram
<b>Data array</b>	
Displaytype	Bars
Line width	1
Show horizontal lines	<input type="checkbox"/>
Relative bar width	60
Position	
Scale	
Label	
Colors	

Data array: MAIN.aHistogram

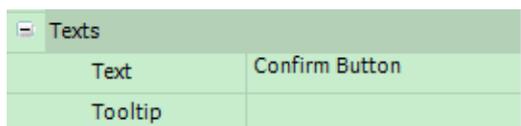
柱状图的参数同样可以修改，找到 **Scale** 展开菜单，红框中的内容是重点需要修改的参数，这四个参数分别对应开始刻度、终止刻度、主刻度、副刻度

Scale	
Scale start	0
Scale end	100
Main scale	20
Sub scale	5
Scale color	<input type="text"/> 192, 192, 192
Base line	0

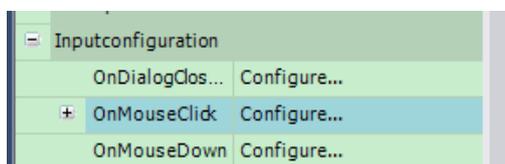
- (11) 选中按钮，会在界面右侧显示其属性栏



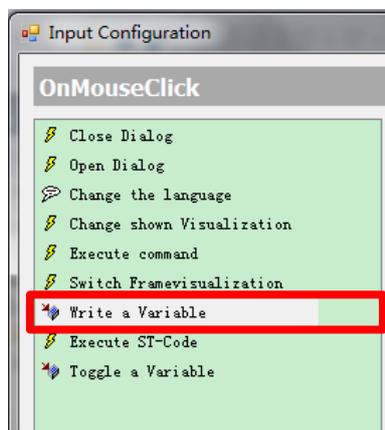
在属性栏里找到 Texts 标签，展开后在 Text 旁的空白栏中输入 Confirm Botton，给按钮命名一个名称



接下来找到 Inputconfiguration 标签，展开后找到 OnMouseClicked

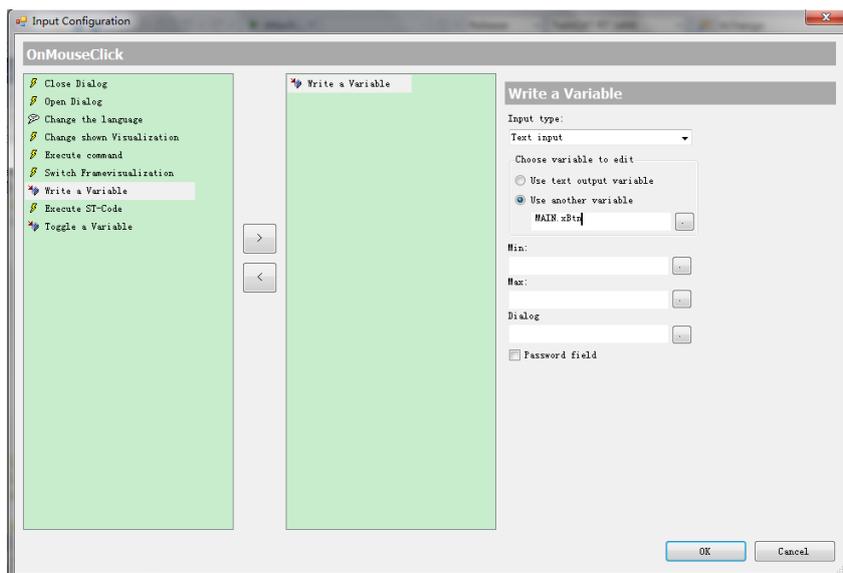


单击 Configure 后会弹出一个对话框，在对话框左侧的是功能栏，这里我们选择 Write a Variable

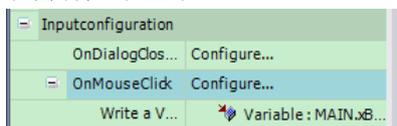


(12) 双击后左侧出现参数设定栏，在 Choose variable to edit 栏目下选择 Use another variable, 下方的空白区被点亮，点击  在弹出的对话框里找到程序中的对应变量，

## 选择 Int 型变量 xBtn

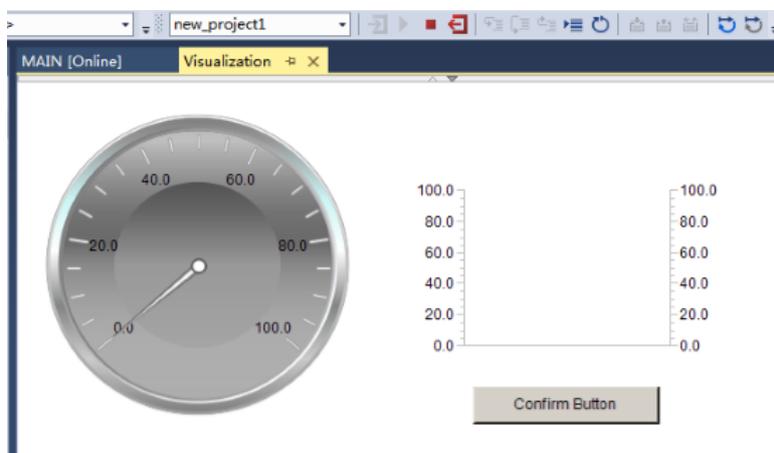


设定完成后 properties 选项卡中会有如下显示：

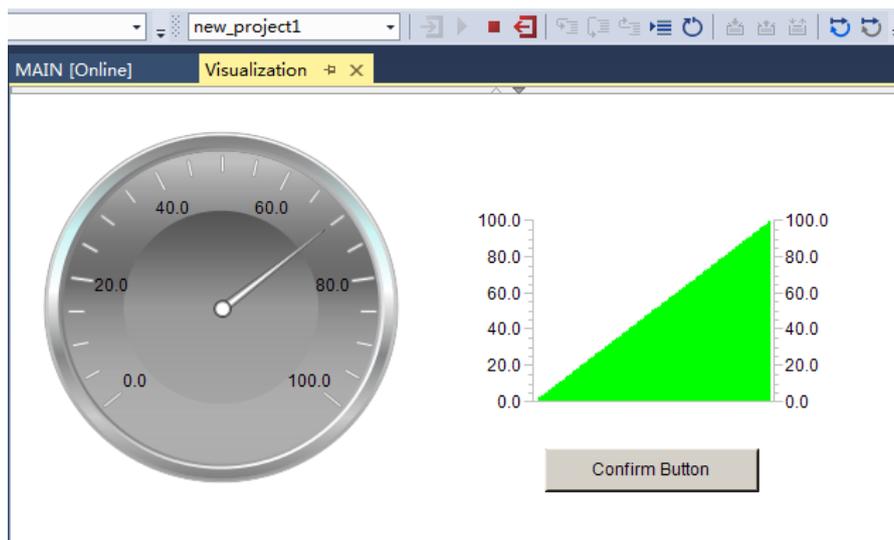


(13) 上述操作完成后，点击登入  Login 再运行  Start ，效果如下：

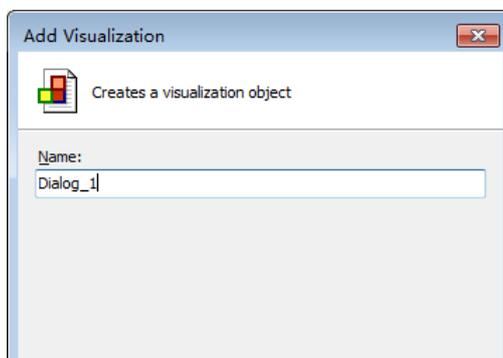
初始状态



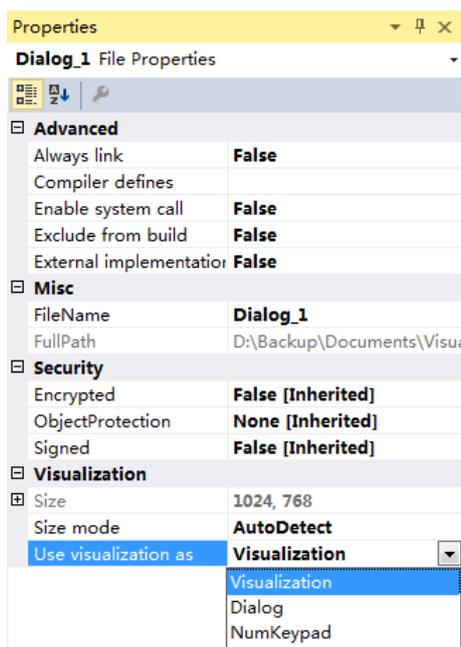
程序运行后的状态



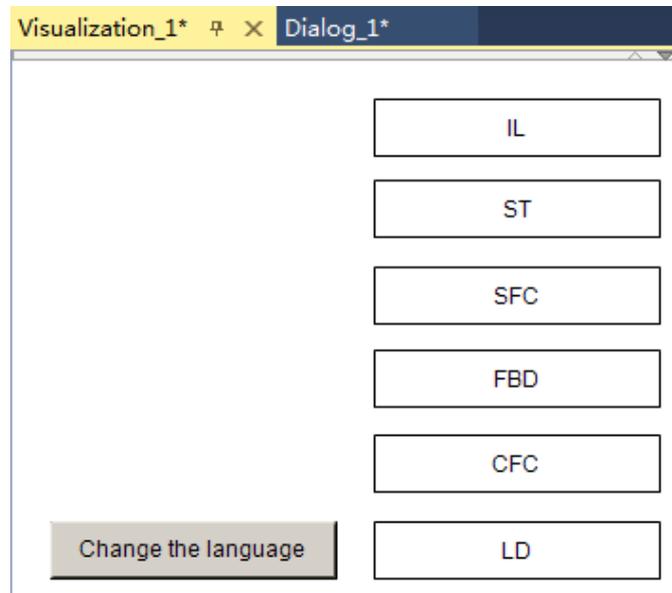
- (14) 再介绍两个鼠标操作的小功能：按下后弹出对话框以及按下后改变语言。我们要先新建一个可视化文件，将其命名为 Dialog\_1，然后确定



选中 Dialog\_1，窗口的右侧会出现这个文件的属性栏，这里需要在 Use visualization as 标签中进行文件的属性修改，单击下拉菜单选择 Dialog

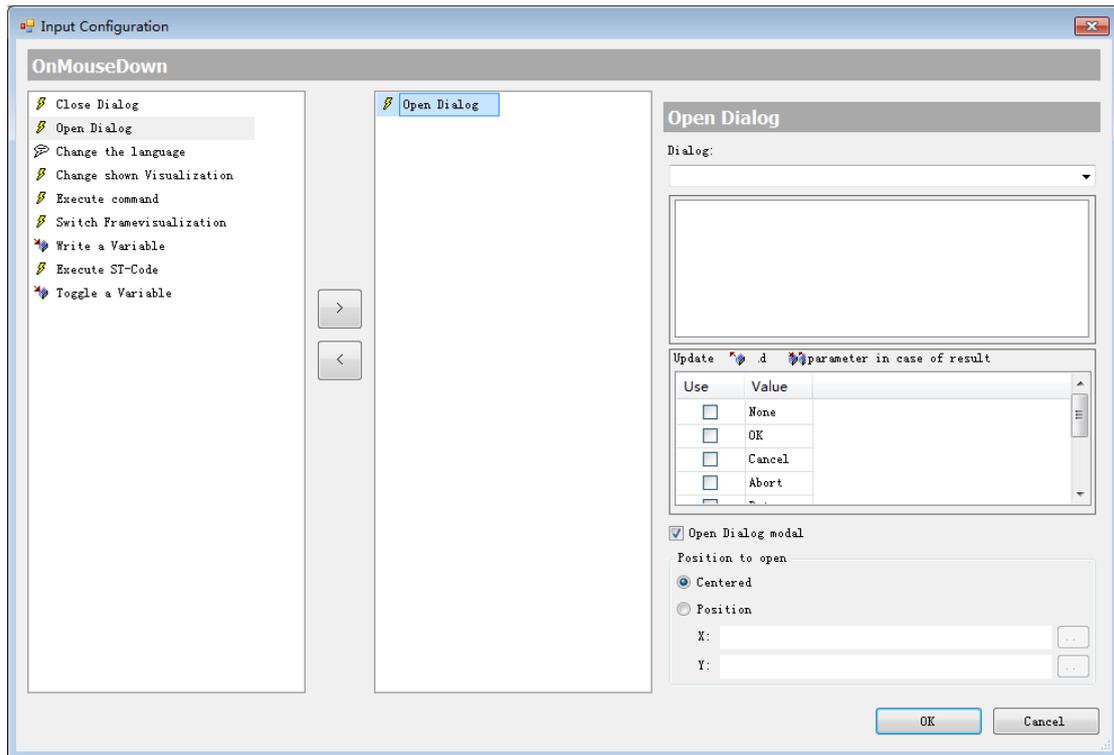


- (15) 在 Visualization 文件中编辑一些需要转换的文字，图中的矩形框体使用的是在 Toolboxes 中 Basic→Rectangle，再添加一个按钮，在 Text 中命名其为 Change the language

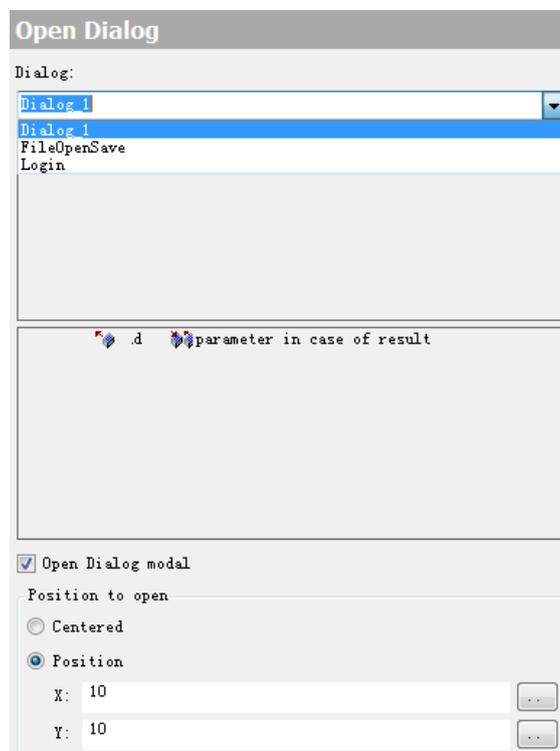


选中这个按钮，在右侧的属性栏中找到 Inputconfiguration 展开，单击 Configure 后在弹出的对话框左侧选项中双击 Open Dialog，对话框右侧会显示其属性

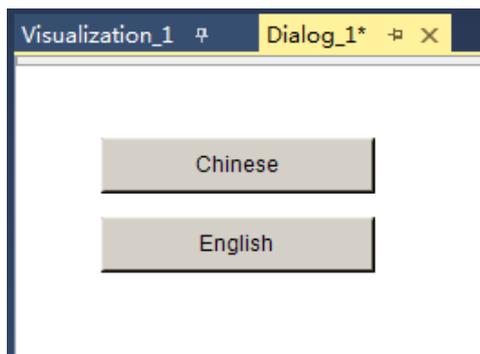




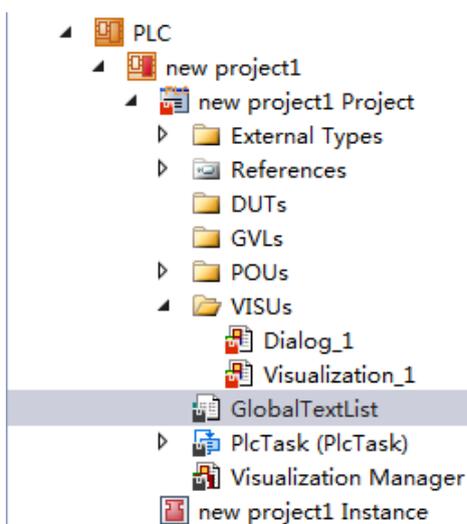
打开 Dialog 的下拉菜单，选中 Dialog\_1，这个就是我们刚刚更改了属性的文件  
下方 Position 可以更改对话框在页面中的显示位置



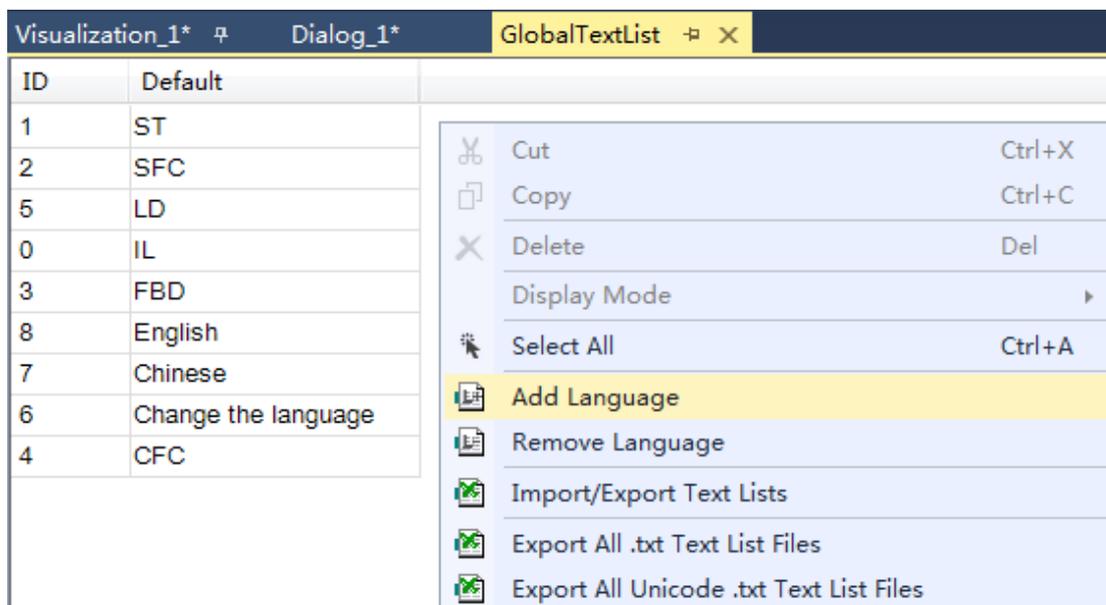
(16) 同时在 Dialog\_1 这个文件中，定义两个按钮，命名为需要转换的语言

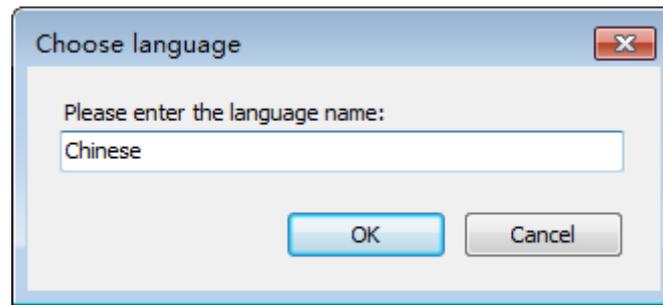


(17) 接着，在左侧的管理窗口 VISUs 文件夹下找到 GlobalTextList 这个文件夹



双击该文件后会出现之前在 Visualization 中已经编辑过的所有文字，在空白处右键，在菜单中找到 Add Language，单击后在弹出的对话框中给需要转换成的语言命名

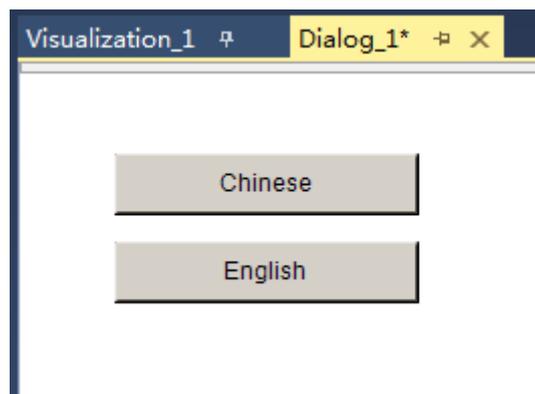


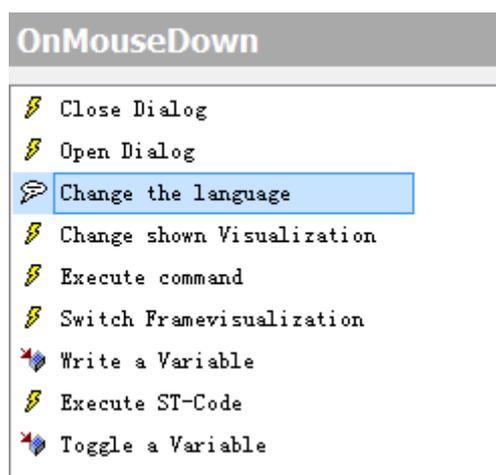


在新建语言栏下输入对应的文字

ID	Default	Chinese
4	CFC	连续功能图编辑器
6	Change the language	转换语言
7	Chinese	中文
8	English	英语
3	FBD	功能块图
0	IL	指令表
5	LD	梯形图
2	SFC	顺序功能图
1	ST	结构化文本

(18) 回到 Dialog\_1 界面，选中 Chinese 按钮，在左侧属性栏里将 Inputconfiguration 展开，单击 Configure 后在弹出的对话框左侧选项中双击 Change the language，对话框右侧会显示其属性

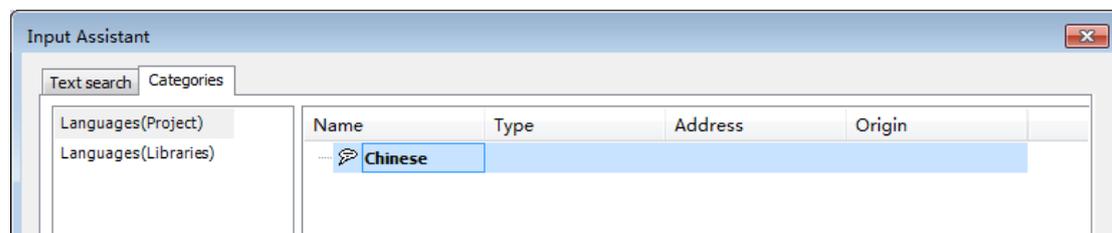




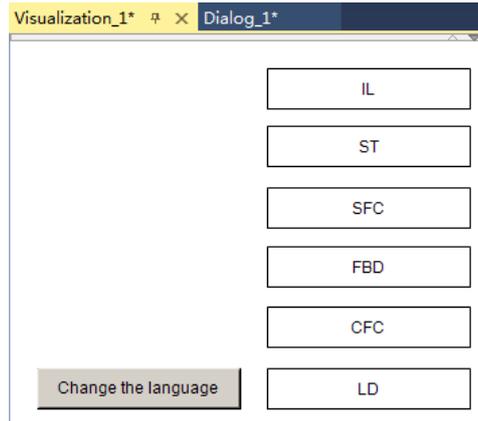
在左侧属性栏点击 Language 下空白处左边的小按钮



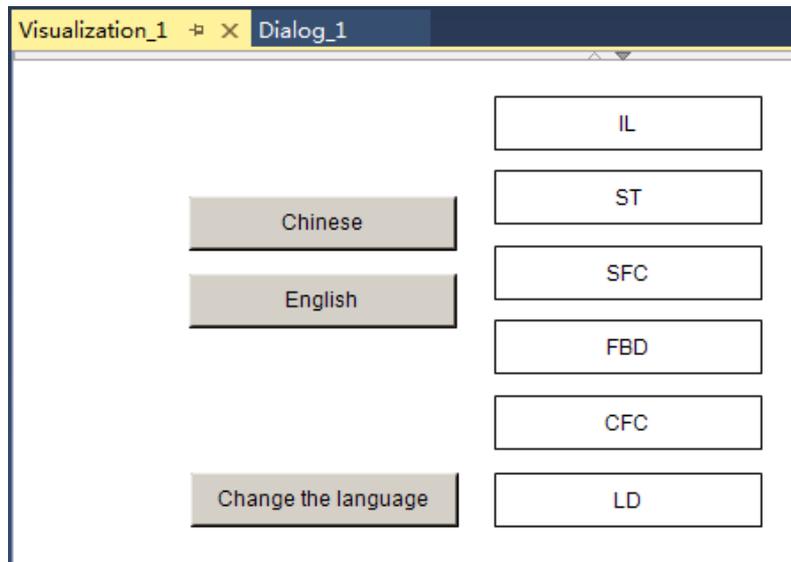
在弹出的对话框中会显示我们刚刚编辑好的语言，选中它，点击确定



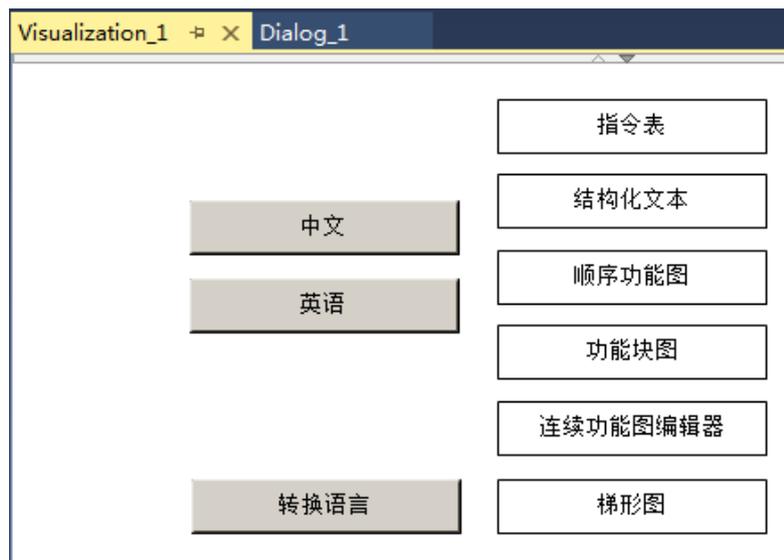
(18) 上述操作完成后，点击登入  Login 再运行  Start ，效果如下：  
初始状态



单击 Change the language 按钮后出现对话框选择按钮



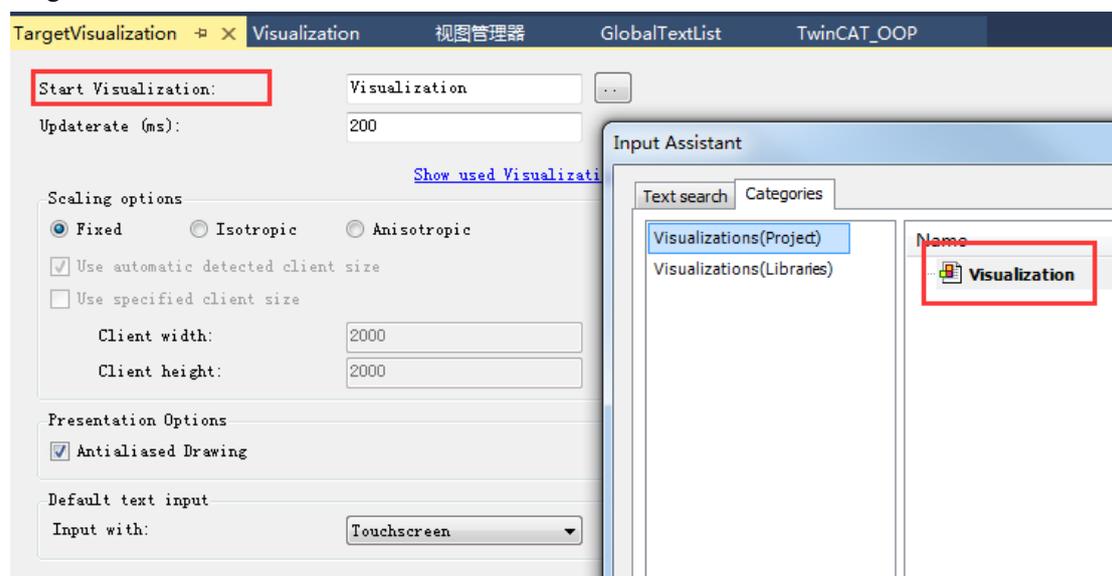
单击 Chinese 按钮后，所有文字显示为中文



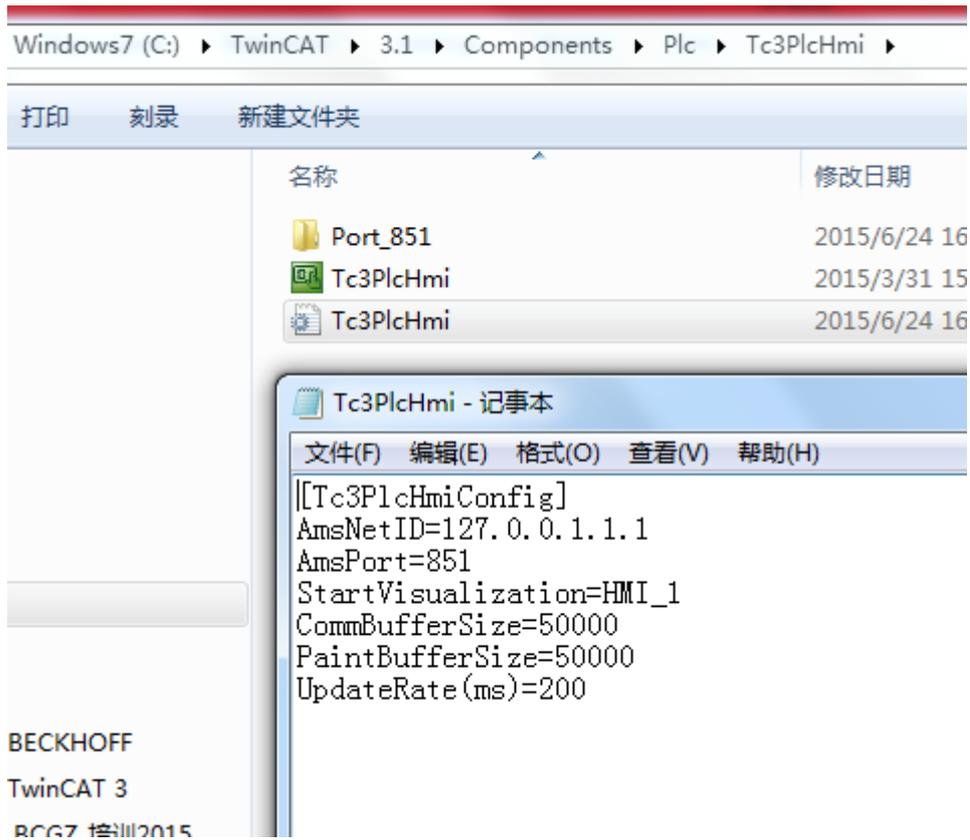
## 13.7.2. 使 HMI 与 PLC 分离

HMI 可以独立于 PLC 程序运行在本机或者另一台 PC 上。与 TC2 相比，进步之处在于不必把 PRO 文件的源代码都放在控制器上。

HMI 要能独立运行，必须在视图管理器中配置画面，Add Target Visualisation，以便创建一个 Target Visualization Client，然后选择初始画面。



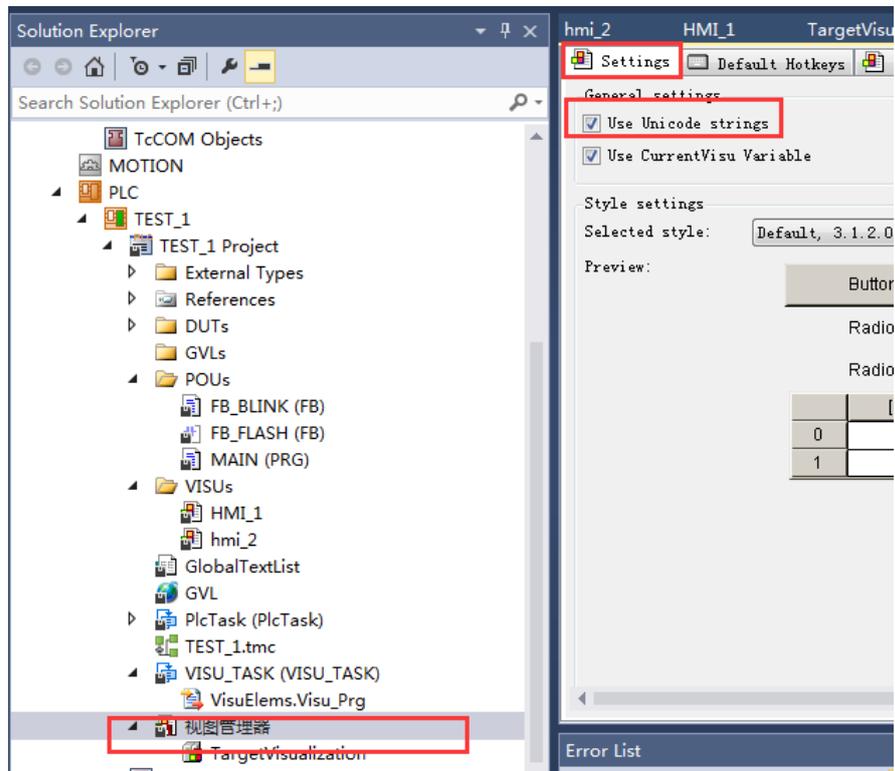
如果不在控制器上运行 HMI 而是在另一台 PC 上运行，PC 上安装 TC3 的 XAR 安装包，并与 PLC 互加路由。（上一版本说是只要装 TC3 ADS，证明是错误的）把控制器上的 EXE 和 INI 可以复制到运行画面的 PC，



然后设置好控制器的 NetID，和初始画面的名称。比如例中项目的“Visualization”

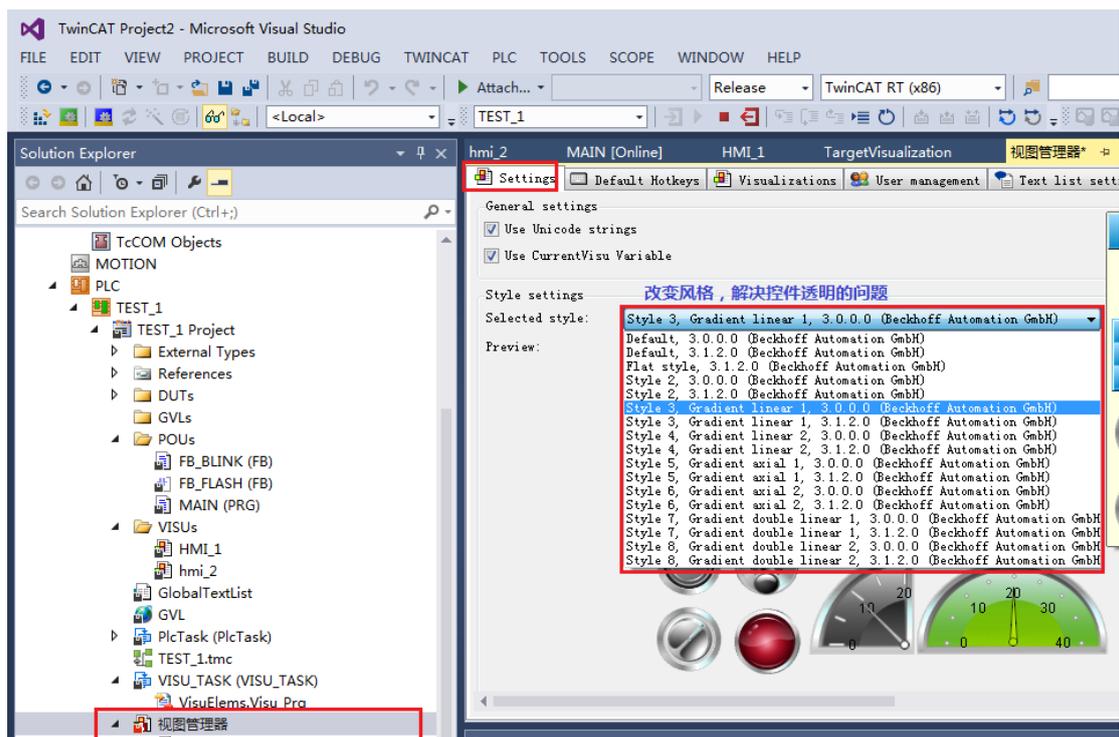
TIP: 8月12日 Lizzy Chen 陈利君测试，XP下，HMI显示本机或远程的PLC程序画面都不行。

### 13.7.3. 中文显示



### 13.7.4. 改变风格

可解决控件透明的问题



我的 DELL 电脑上，只有 Default 3.0.0.0 可用，换成其它风格就不行了。

## 14. 连接企业数据库

声明：TC3 的 ADS 通讯，PLC 的端口号做了改动，默认 PLC 项目的端口号为 851，所以本章所有示例中的 Port 801，应改为 Port 851。

### 14.1. 概述

TwinCAT 是基于 PC 和 Windows 的控制系统，可以很方便地集成到企业局域网。当用户需要把生产记录数据定期存放到服务器上的数据库，或者从数据库获取工艺配方时，就需要 PLC 访问企业数据库。

Beckhoff 提供扩展功能包 TwinCAT Database Server，用于实现 TwinCAT PLC 到 Database 的数据中转。与 TwinCAT OPC Server 类似，TwinCAT Database Server 也是通过 ADS 通讯收集 PLC 数据，然后转发至指定数据库（Database）中的表（Table）。

通常，TwinCAT Database Server 应安装在数据库所在的 PC 上，此外，该 PC 上还应安装 TwinCAT Demo 版或者 TwinCAT CP，以提供 ADS 通讯接口。

## 16. 从 TwinCAT 2 到 TwinCAT 3

### 16.1. 概述

TwinCAT 2 的软件是 20 世纪 90 年代开发完成的，虽然软件的功能模块不断增加，但其程序框架是基于微软公司 32 位的 Windows 操作系统，以及单核 CPU 的硬件平台，这也是为什么在 64 位操作系统的开发 PC 上 TwinCAT 2 不能仿真运行的原因。随着 IT 技术的发展，计算机 CPU 的单核运算能力已经趋于稳定，其性能的提升主要通过多核并行来实现。而 TwinCAT 2 的软件架构决定了它只能运行于一个 CPU 内核，无法完全发挥多核 CPU 的性能。

### 16.2. TC3 的新功能

TC3 的软件设计考虑了 32 位和 64 位操作系统，单核和多核 CPU。换言之，TC3 运行核既可以工作在 32 位操作系统，也可以工作在 64 位操作系统。Beckhoff 既提供 CE 下的 TC3 运行核，也提供 Win 7 下的 TC3 运行核。但是目前 CX10x0 还没有 TC3 的 Runtime，估计今后也不会推出。CX50x0、CX20x0 的 TC3 运行核已经推出，所有 Beckhoff 工控机都可以选配 TC3 运行核。

TC3 的软件容量比 TC2 扩大了至少一个数量级。TC2 的 62 个任务优先级和 255 个 NC 轴已经可以满足绝大部分工厂自动化的用户需求。而 TC3 可以任意指定 CPU 核，软件设计上支持最多 128 个 CPU。这使得 TwinCAT 3 有可能运用到工厂自动化之外的其它领域，比如测量、机器视觉、港口码头、石油化工等等。TC3 下的 NC，最多可以控制 255 个轴，几乎可以用于任何设备、项目的控制系统。

TwinCAT 3 除了解决 64 位操作系统和多核 CPU 的问题之外，还增加了 C++编程和 Matlab 建模功能，强化了面向对象编程的理念。

用户可以用 C++编写自己的类 (Class)，再创建若干这种类 (Class) 的对象 (Object)。这些对象 (Object) 有相同的属性、接口和动作，可以指定其执行周期。Matlab 建立的模型是另一种类 (Class)，它们与 C++类的区别在于，它们是在 Matlab 中创建然后导入到 Tc3 的，而 C++的类可以在 Tc3 中从零开始创建。Tc3 中导入 Matlab 中建好的类以后，就可以创建若干这种类的对象，其操作和 C++的类相同。

#### 16.2.1. TC3 的继承性

TC3 中的 PLC 任务和 NC 任务可以看作“厂家预置”的标准类，考虑到用户的使用习惯，与 TC2 中的 PLC 和 NC 编程、配置方法都保持一定的兼容性。尤其是 NC 的调试界面，与

TC2 中的 System Manager 中的调试界面完全相同。

TC3 延用了 TC2 中的 IO 硬件扫描、配置和变量映射的界面。

TC3 延用了 TC2 中的 ADS 通讯方式。不同的对象之间的数据交换，如果不是直接映射双方的接口变量，就采用 ADS 通讯的方式。TC3 与 HMI 程序，可以存在于同一个 Project，但它们之间并不能共享全局变量。HMI 要访问 PLC 数据，通过 ADS 通讯。

鉴于以上原因，如果用户只是用 TC3 来实现 TC2 的功能，只需要经过半天或者一天的简单培训就可以熟悉开发环境了。但是 TC3 的安装步骤比 TC2 要繁琐得多，需要依照“安装指南”按步操作。

## 16.2.2. TC2 与 TC3 的适用范围

对于绝大多数的工厂自动化用户，单核 CPU 和 32 位操作系统已经可以满足要求，因此可以继续使用 TwinCAT 2。

用户升级到 TwinCAT3 通常有以下原因：

- 需要用到 C++编程，比如一些特殊算法，原先已经有 C++代码。
- 需要用到 Matlab 建模
- 控制器是多核 CPU，程序量大，单核不能满足要求。
- 偏好 TwinCAT3 的开发环境

## 16.3. TC2 转换 TC3 的解决方案

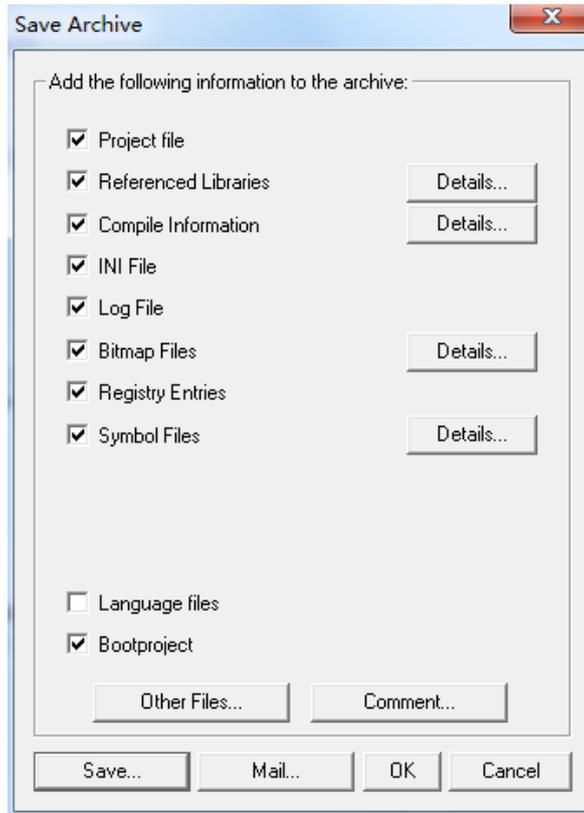
本章内容来自 BECKHOFF 的 TwinCAT 3 培训文档，作者：杨煜敏。  
见附件《(15) TC2 转换 TC3 解决方案.doc》

### 16.3.1. 先在 TwinCAT 2 中打包

方法一：（正规做法，By TC3 产品经理 杨煜敏 on 2015-06-26）

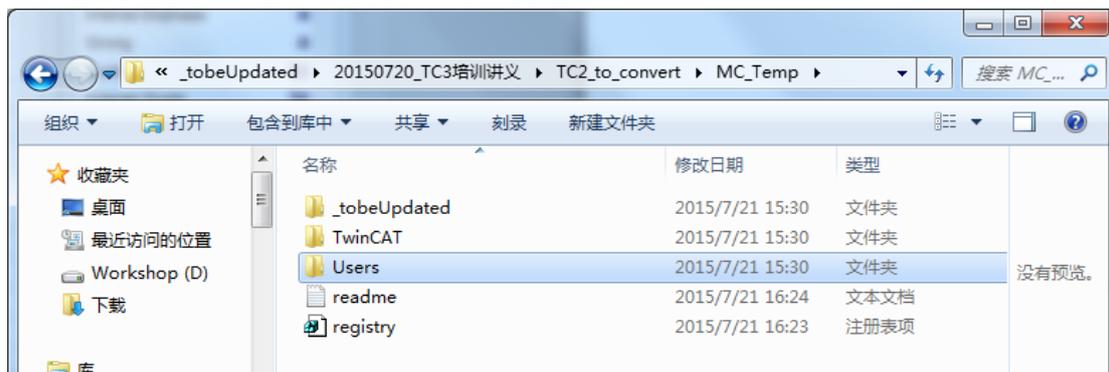
- 把 TSM 文件和 PRO 文件放在同一路径下，
- PRO 文件编译，
- 在 TwinCAT System Manager 中引用，
- 确认链接正确，
- 存盘，
- 目标系统选本机，激活配置（为了产生 CurrentConfig.tsm）。
- 在 PLC Control 中全编译，
- 保存。

然后在 PLC Control 中 File|Save Mail Archive，选择全部文件。生成 zip 压缩包。



- 说明：当前为 TC2 模式才会生成正确的压缩包。如果当前实际是 TC3 模式，就不会生成 CurrentConfig.tsm 文件。

把压缩包复制到 TC3 所在的开发 PC 中，并解压。



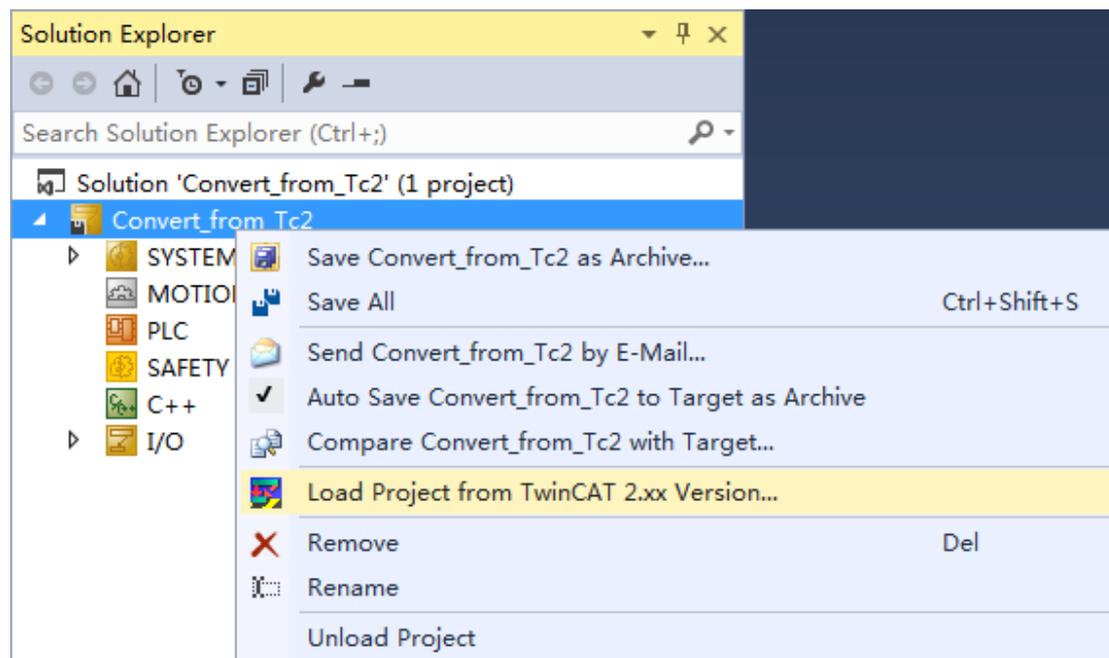
方法二：（Tested By Lizzy on 2015-07-21, TC3.1 build 4018.4）

不用打包，只要把 tsm 和 pro、tpy 等相关文件夹放在同一路径下，tsm 文件中引用 tpy 文件时使用相对路径，就等效于方法一中的 Save Archive。Lib 文件夹不是必须的。

## 16.3.2. 在 TwinCAT 3 中装载

必须是在 TwinCAT 3 中新建一个空白的 TwinCAT Project，才会出现菜单“Load Project from

TwinCAT 2.xx Version”



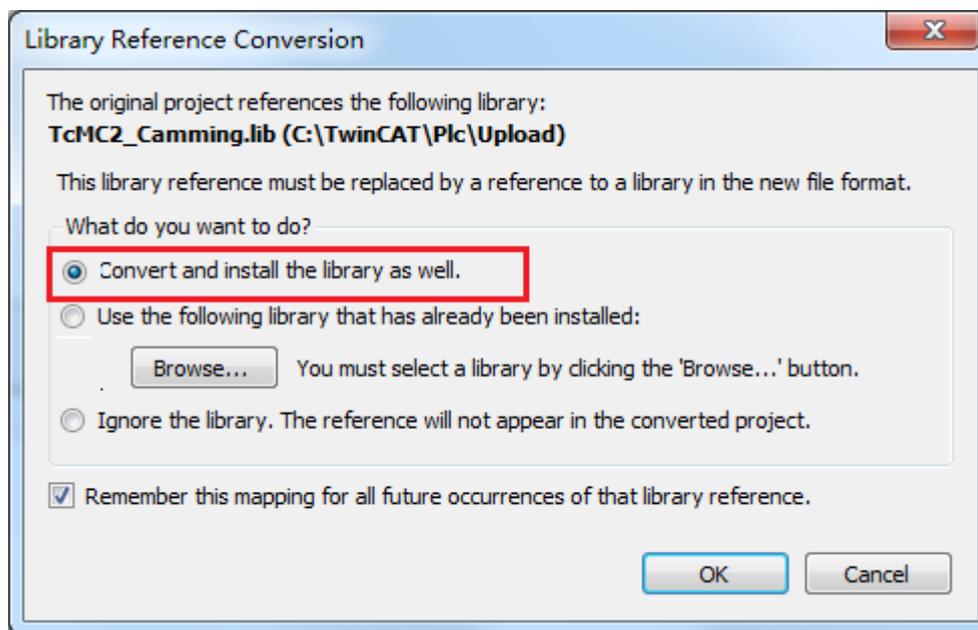
选择“MC\_Temp\Users\LIZZYC~1.BEC\AppData\Local\Temp\CurrentConfig.xml”，

装载过程中，最重要的是 Lib 文件的替换，如果是 Beckhoff 提供的库，按以下规则替换：

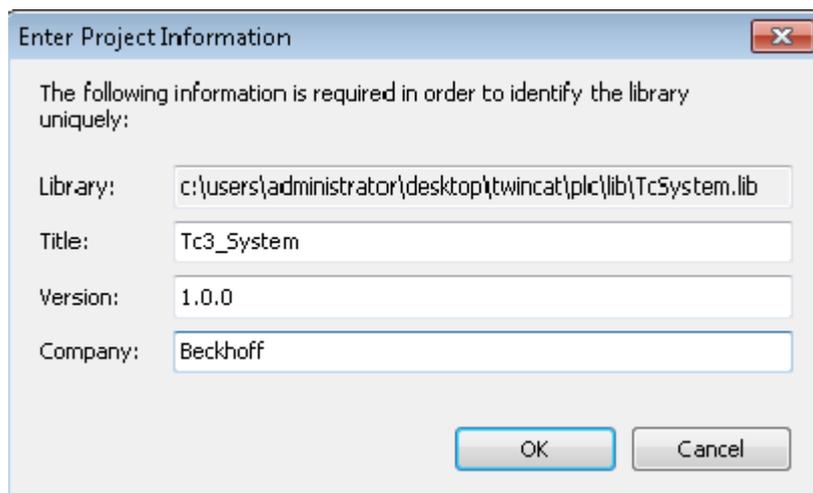
#### Standard Libraries

TwinCAT2 PLC Lib	Category	TwinCAT 3 PLC Library
standard	System	Tc2_Standard
TcBase	System	Tc2_System
TcSystem	System	Tc2_System
TcTestAndSet	System	Tc2_System
TcUtilities	System	Tc2_Utilities
TcFloatPC	System	Tc2_Utilities
TcMDP	System, Communication	Tc2_MDP
TcSystemC69xx	System IPC-Series	Tc2_SystemC69xx
TcSystemCX	System CX-Series	Tc2_SystemCX
TcSystemCX1000	System CX-Series	Tc2_SystemCX
TcSystemCX1010	System CX-Series	Tc2_SystemCX
TcSystemCX1020	System CX-Series	Tc2_SystemCX
TcSystemCX1030	System CX-Series	Tc2_SystemCX
TcSystemCX5010	System CX-Series	Tc2_SystemCX
TcSystemCX5020	System CX-Series	Tc2_SystemCX
TcSystemCX9000	System CX-Series	Tc2_SystemCX
TcSystemCX9010	System CX-Series	Tc2_SystemCX
TcloFunctions	IO	Tc2_loFunctions
TcRaidController	IO	Tc2_loFunctions
TcEtherCAT	IO	Tc2_EtherCAT
TcSUPS	IO	Tc2_SUPS
TcPlcCoupler	IO	Tc2_Coupler

如果程序全部使用 Beckhoff 的库文件，这样就算完成了。如果程序使用了自定义的或者是第三方的 lib 文件，就需要先把 TwinCAT 2 的 lib 转换成 TwinCAT 3 的库。在下面步骤中应选择 “Covert and install the library as well”

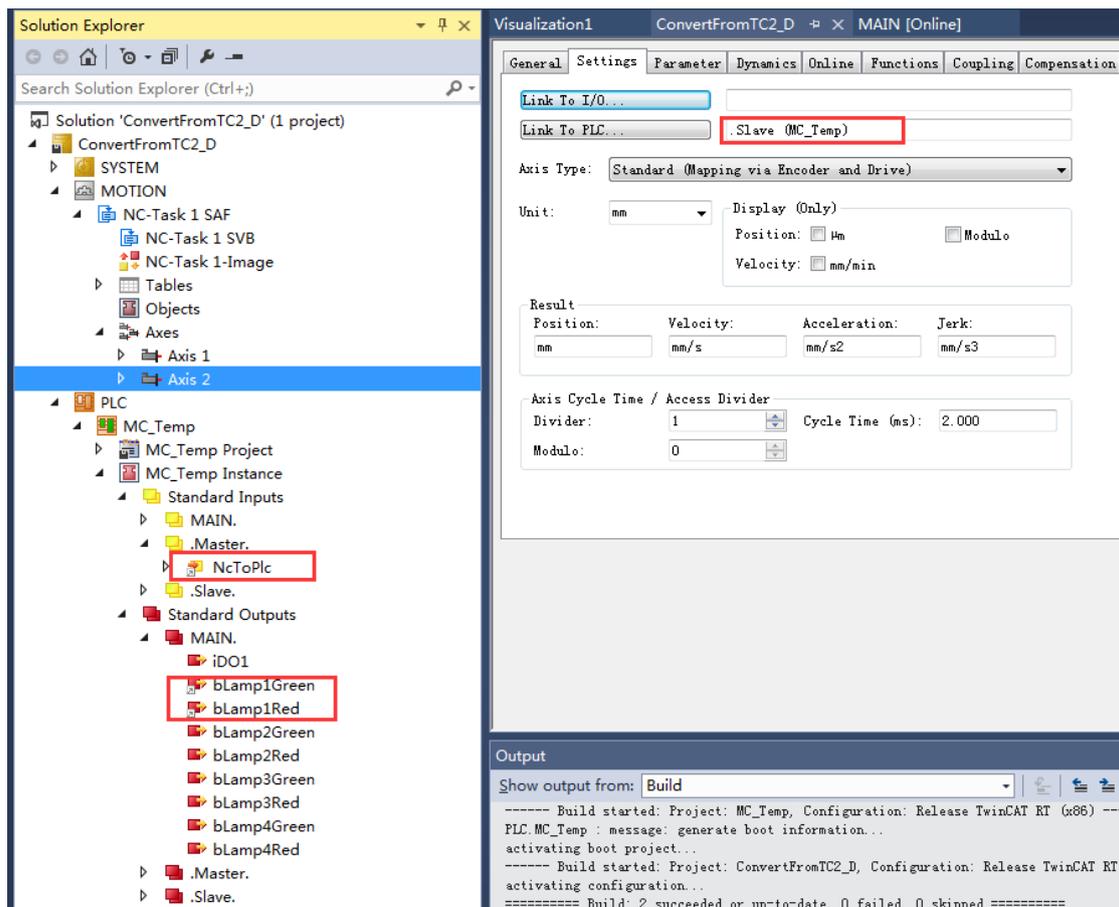


转换时要求填写转换库的信息



### 16.3.3. 转换结果

库文件替换成功后，TC3 中就有全部的代码、配置和任务了。并且可以看到所有映射都自动转换成功。



检查 IO 配置是否一致，原来 TC2 的硬件平台通常不同于 TC3 的 IO 硬件平台，按 TwinCAT 2 中相同的方法，调整 IO 与实际一致。  
编译 OK 就可以激活运行了。

但是也有极个别 TC2 功能库转换成 TC3 时会出现 FB 名字或者接口变量类型有变化的情况，此时只能根据自己的理解手动调整代码。

### 16.3.4. HMI 的转换

TC2 升级到 TC3 的时候，除了转换 PLC 程序之外，如果有触摸屏或者 HMI 软件经 ADS 与 PLC 通讯，那么触摸屏或者 HMI 软件中不仅要改端口号（例如：801 改成 851），还要修改变量写法。ADS 访问 TC2 的全局变量写为 “.Varname”，Tc3 下，要把变量名所在的变量文件写出来，比如 GLV1.Varname。

### 16.3.5. TC3 ADS

TC3 里 ADS 的不同。有培训视频  
如果是高级语言写的 HMI，那么 HMI 所在的 PC 上至少要安装 TC3 ADS。其功能类似 TwinCAT 2 CP。

- 经 ADS 访问 TC3 的 PLC 时，变量的写法与 TC2 不同。
- HMI 软件经 ADS 访问 TC3 时，HMI 所在的 PC 上建议安装 TC3 ADS 3.1。  
虽然装一个完整的 TC3 也可以，但是没有必要。

## 16.4. TwinCAT 3 实训文档

“TC3\_培训资料\3\_TwinCAT3 入门教材”

