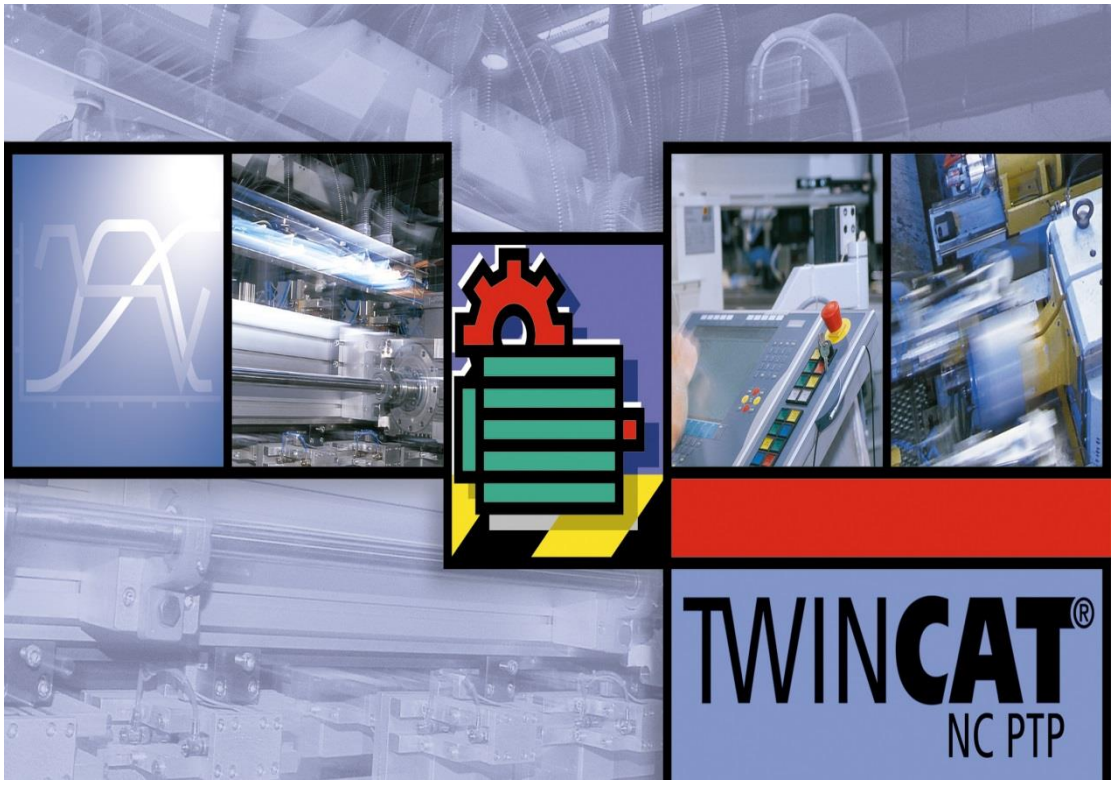
BECKHOFF



自动化新技术

TwinCAT2 运动控制入门教程

Version 1.71

毕孚自动化设备贸易（上海）有限公司

2019 年 **4** 月

前 言

进入 21 世纪，中国作为世界制造业基地的地位日益确立和稳固，产业机械迅猛发展。随着人力成本的不断升高，提高产业机械自动化程度成为业界的共识。

德国 BECKHOFF 公司推出了 TwinCAT NC ，这是一款基于 PC 的运动控制软件，不但实现了 NC 与 PLC 的无缝集成，而且支持几乎所有的伺服驱动器接口：脉冲、模拟量、现场总线和以太网，这意味着用户可以在伺服驱动器和电机的选择上拥有更多的自由。另一方面，即使在脱离伺服驱动器和电机的条件下，开发人员也可以在任意计算机上模拟调试自己的 PLC 和 NC 程序。辅以 BECKHOFF 公司提供的示波器软件 Scope View ，用户可以观察任意变量的曲线。

本书的目的是教您如何尽可能快捷地运用 TwinCAT NC PTP 编写程序，并假定您已经熟练掌握了 TwinCAT PLC 编程。本书中详细介绍了 TwinCAT NC PTP 的系统概述;如何在 System Manager 中独立硬件的 Axis 配置和调试界面;TwinCAT PLC Control 中独立于硬件的单轴及多轴运动控制程序的编写；完整的 AX5000 和电机参数的设置；以及电子凸轮表、飞锯、TwinCAT NC FIFO 等功能。

在本书的撰写过程中，要特别感谢广州分公司的同事陈利君，本书摘录了许多她撰写的技术文章。同时也希望本书能让更多的用户了解 TwinCAT NC，使用 TwinCAT 这个强大的工具，开发出更多高端的自动化设备。

科技，让生活更美好！

编者 邵伟栋 张立文

2015-10-14

目 录

一． TwinCAT NC PTP 系统介绍

本章目标：

通过本章节的学习，学员将了解：

- ☑ PLC轴、NC轴和物理轴的关系
- ☑ NC轴可控制的物理轴类型以及NC轴的数量
- ☑ NC轴的控制周期

TwinCAT NC PTP是Beckhoff公司的运动控制软件的名称，TwinCAT 是“The Windows Control and Automation Technology”的缩写，即基于Windows操作系统的自动化控制技术，而NC PTP是“Numerical Control Point To Point”的缩写，NC（Numerical Control）是自控领域的一个专业术语，类似MC（Motion Control）,也指运动控制，NC PTP就是点对点的运动控制。

TwinCAT NC 是基于 PC 的纯软件的运动控制，它的功能与传统的运动控制模块、运动控制卡类似。由于 TwinCAT NC 与 PLC 运行在同一个 CPU 上，运动控制和逻辑控制之间的数据交换更直接、快速，因此 TwinCAT NC 比传统的运动控制器更加灵活和强大。TWINCAT NC 的另一个特点是完全独立于硬件，用户可以选择不同厂家的驱动器和电机，而控制程序不变。程序的运动控制指令集遵循 PLCOpen 组织关于运动控制功能块的定义规范 V1.0 和 V2.0。

TwinCAT NC 有 PTP 和 NC I 两个级别，PTP 即点对点控制方式，可控制单轴定位或者定速，也可以实现两轴之间的电子齿轮、电子凸轮同步。在此基础上，Beckhoff 还提供 Dancer Control（张力控制）、Flying Saw（飞锯）、FIFO（先入先出）等多轴联动方式。此外，用户还可以在 PLC 程序中编写位置发生器，每个 PLC 周期都计算目标位置、速度和加速度，并发送给 TwinCAT NC 去执行。而 TwinCAT NC I 除了能够实现 TwinCAT NC PTP 的所有功能之外，还可以执行 G 代码，实现多轴之间的直线、圆弧和空间螺旋插补。

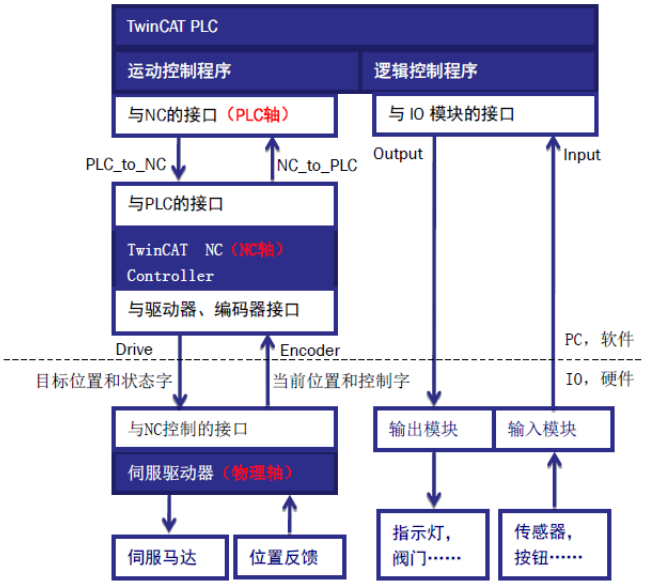
1. TwinCAT NC PTP与TwinCAT PLC的关系

TwinCAT NC PTP把一个电机的运动控制分为三层：PLC轴、NC轴和物理轴。

A、 PLC 程序中定义的轴变量，叫做 PLC 轴。

B、 在 NC 配置界面定义的 AXIS，叫做 NC 轴。

C、 在 IO 配置中扫描或者添加的运动执行和位置反馈的硬件，叫做物理轴。它们的关系如图所示：



由图可见，PLC程序对电机的控制，必须经过两个环节：PLC轴到NC轴；NC轴再到物理轴。

PLC轴的控制，是指PLC程序中编程，调用运动控制库的功能块。

NC轴不需要编程，它的运算分为轨迹规划、PID运算和IO接口处理。其中轨迹规划和PID运算是固定的，与硬件无关。IO接口处理随接口类型而不同。这些运算都在后台进行，用户只需要进行参数设置。这些参数可以固化在TwinCAT System Manager配置文件中，也可以在PLC程序中通过ADS指令读写。

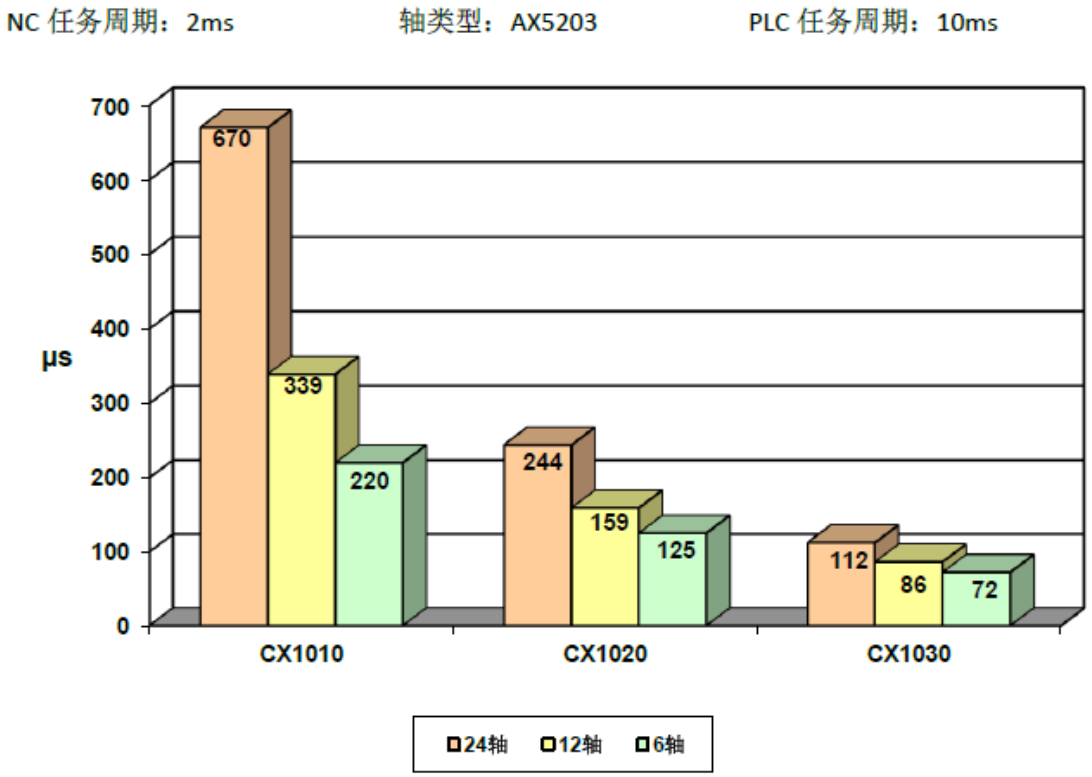
物理轴，指驱动器、电机和编码器。物理轴的配置，主是要驱动器的设置。在驱动器中，要配置好正确型号的电机、编码器、电子齿轮比，还要调整位置环、速度环、电流环的 PID 参数。如果是总线接口，还要设置好接口变量和通讯参数。

TwinCAT NC 做轨迹规划，是指接收到 PLC 指令以某个速度运动到某个位置后，计算出每个 NC 周期（比如：2ms）伺服轴应该到达的位置。IO 接口处理，是指根据轴的硬件类型和相应的参数设置，进行单位换算，将 NC 运算得出的目的位置，换算成驱动器可接受的输出变量值。

2．TwinCAT NC PTP控制的轴的类型和数量

和传统的硬件运动控制器和运动控制卡不同，TwinCAT NC PTP是纯软件的运动控制。理论上，最多可以驱动255个伺服轴。在实际应用中，一个EPC或者PC上运行的TwinCAT NC PTP软件能够控制的伺服轴数量，与PC或者EPC的CPU速度、内存以及NC任务的周期有关。

图：BECKHOFF 的控制器上运行 TwinCAT NC 的性能对比



TwinCAT NC 支持多种伺服轴类型，下面介绍几种常用类型：

总线接口

总线接口，又称数字接口，比如Sercos，CanOpen（DS402），Lightbus等。由不同厂家生产的同一种总线协议的伺服驱动器，在TwinCAT NC中视作同一种驱动器。值得一提的是，对于EtherCAT接口的驱动器，其协议层通常使用CanOpen，或者Sercos。在TwinCAT NC中，EtherCAT接口CanOpen协议的驱动器，与CanOpen接口CanOpen协议的驱动器，都视作同一种驱动器。同理，EtherCAT接口Sercos协议的驱动器，与Sercos光纤接口Sercos协议的驱动器，也视作同一种驱动器。

紧凑型驱动模块

这里主要是指Beckhoff公司的步进电机驱动模块KL2531/2541、EL7031/7041，伺服电机驱动模块EL7201等等。

高速脉冲接口

TwinCAT NC通过控制脉冲输出模块KL/EL2521的输出频率，控制伺服驱动器或者步进电机驱动器。同时，TwinCAT NC直接把KL/EL2521发出的脉冲数量，作为位置反馈信号。

模拟量控制

TwinCAT NC 通过控制电压输出模块 KL/EL4xxx 的电压，控制伺服驱动器和电机的速度。此时，必须配置编码器模块 KL/EL5xxx 作为位置反馈。

3．TwinCAT NC PTP的控制周期

通常说的NC周期，是指轨迹规划和PID运算的周期，是NC与伺服驱动器交换数据的周期，目标位置、当前位置、控制字、状态字都以这个频率更新。在TwinCAT System Manager中，叫做NC Task SAF任务周期，默认值为2ms，理论上最小设置为50us。当连接硬件运动轴时，以BECKHOFF的伺服驱动器AX5000为例，位置环周期为125us，所以NC周期设置为50us是没有意义的，实际上250us的NC周期已经是很高端的应用了。

另一个 NC 周期，是 NC 与 PLC 交换数据的周期，比如 NC 轴状态、当前位置、使能信号等等，都是以这个周期刷新的。在 TwinCAT System Manager 中，叫做 NC Task SVB 任务周期，默认值为 10ms，与 PLC 程序中默认的任务周期一致。

4．TwinCAT NC PTP的配置、编程、调试

在开发PC上安装TwinCAT时，如果选择TwinCAT NC PTP或者TwinCAT NC I 级别，安装完成后，运行TwinCAT System Manager，左边的树形结构中就包含了TwinCAT NC Configuration这一项。TwinCAT NC任务和轴的配置调试就是在这一项下进行。

TwinCAT NC任务的配置主要是设置任务周期，多数情况下，使用默认值即可。TwinCAT NC轴的配置包括：编码器（Enc）、驱动器（Drive）、NC控制器（Ctrl）、与PLC的接口（Inputs和Outputs）。Enc和Drive的配置决定了NC轴与哪个驱动器对应，而Inputs和Outputs则决定它对应PLC程序中的哪一个轴结构型变量。Ctrl中的设置则决定了PID运算的模型和参数。

TwinCAT NC轴的调试，分为单轴点动、指定方式动作和双轴齿轮或凸轮联动。这些动作都可以在TwinCAT System Manager的NC Configuration项下完成，不需要编写任何PLC程序。NC轴调试的目标，是确保电机能够按要求走得准、走得稳，消除单位设置、PID参数、传动机械方面的误差。

TwinCAT NC 轴的编程，在 TwinCAT PLC 中通过引用运动控制功能库 TcMC.Lib 或者 TcMC2.Lib，并调用其中功能块来实现。实际应用中，必须在 TwinCAT NC 轴调试完成后，才用 PLC 程序控制轴的动作，以达到设备的工艺要求。

二． System Manager 中 AX5000 的配置

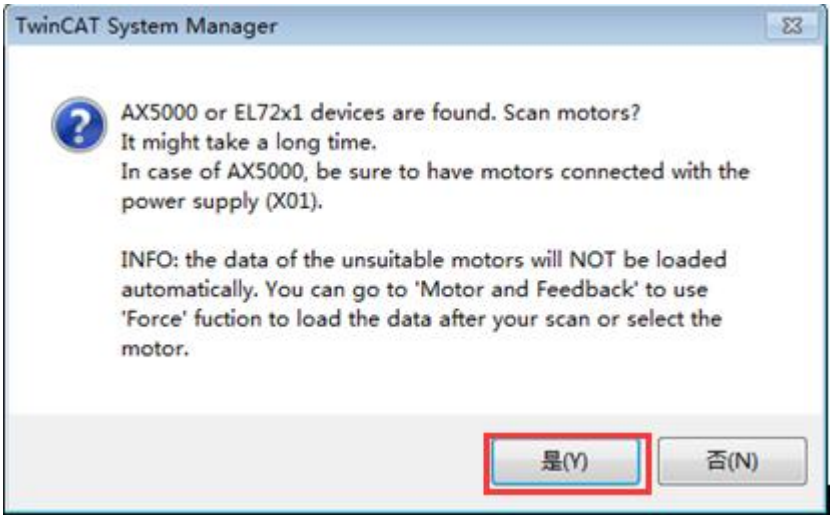
本章目标：

通过本章节的学习，学员将了解：

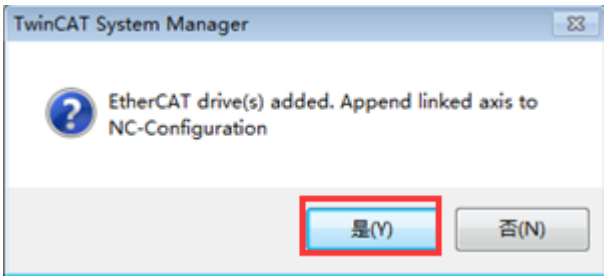
- ☑ 驱动器扫描方法
- ☑ NC 轴和物理轴的关系
- ☑ AX5000 的配置方法
- ☑ 通过System Manager软件对轴进行正反转调试
- ☑ 重要的NC参数设置
- ☑ 如何配置第三方伺服驱动
- ☑ 如何配置第三方伺服电机

1. 硬件扫描

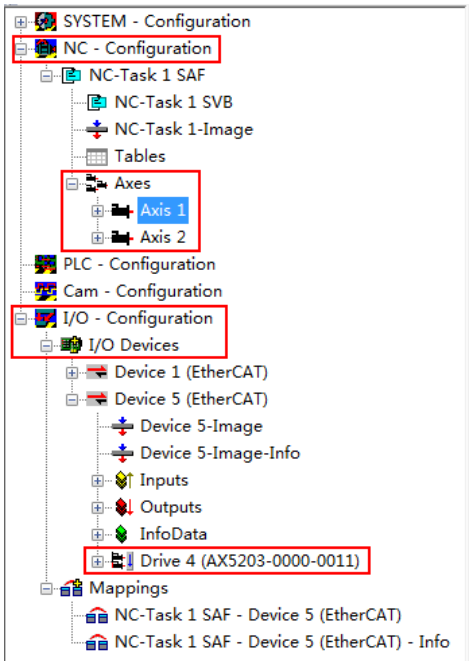
将培训器材上电后开始扫描硬件，培训室实验器材有两种，一种为：面板 PC+IO+AX5000 驱动+电机，简称**器材 A**，一种为：嵌入式 PC+IO+ AX5000 驱动+电机，简称**器材 B**，两种器材皆有 Beckhoff 的驱动和电机，在扫描硬件的时候会有下图提示，“AX5000 设备或 EL72X1 设备已经发现，是否要扫描驱动器所带的电机型号？”，点击“是”之后，软件会扫描驱动器下面所带的电机，会花一些时间。



然后会提示是否要添加 NC 轴和实际物理轴的链接，点击“是”。



扫描完成后可以在 NC-Configuration 中看到 3 根轴或者 2 根轴，器材 A 由于 IO 中还有 EL7041 步进电机模块，因此有 3 根轴，Axis1 对应步进电机，Axis2、Axis3 对应伺服驱动器控制的两台电机，IO-Configuration 中扫描到硬件驱动器 AX5203-0000-0011。



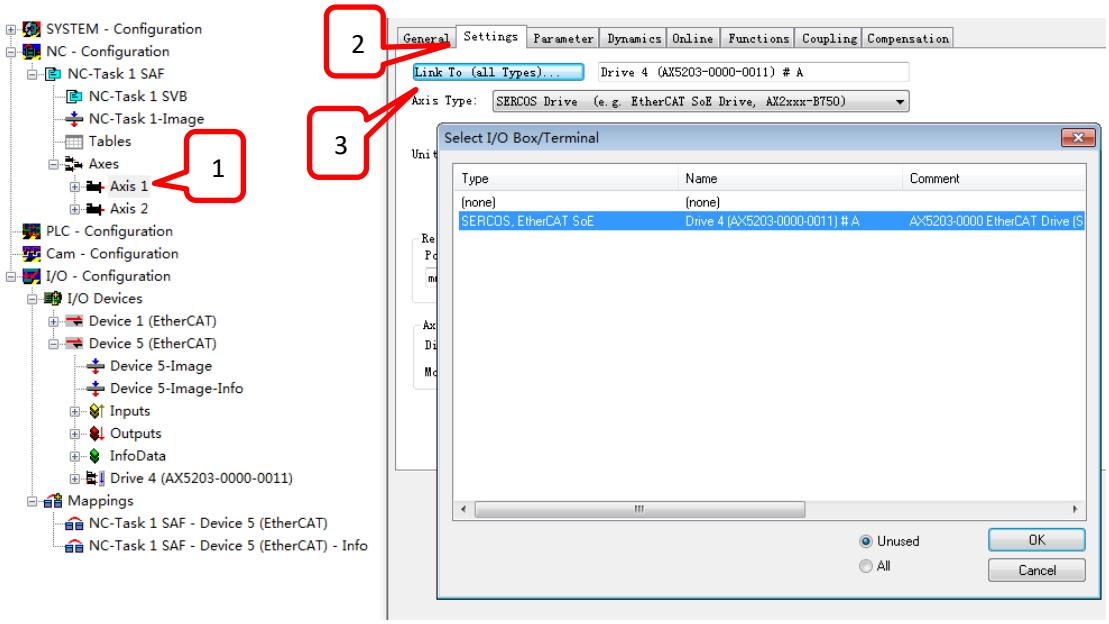
随堂问答：

学员提问：EL72x1是什么产品？

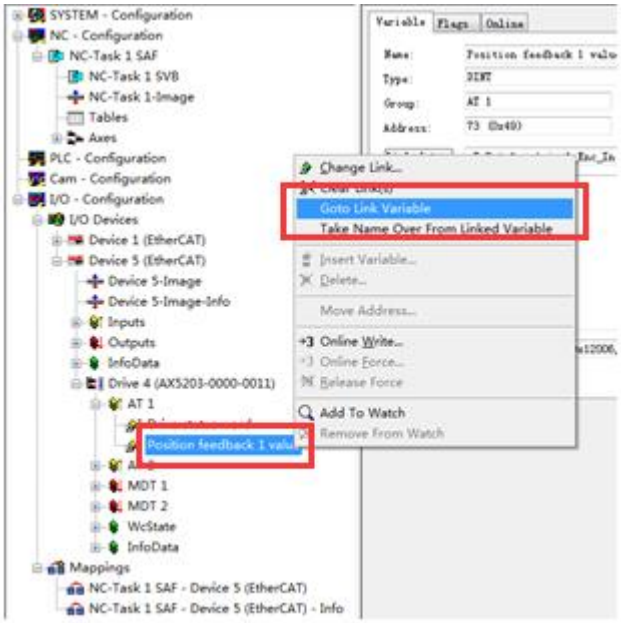
讲师解答：EL72x1 是伺服端子模块，可以直接连接伺服电机，不需要专门配置一个伺服驱动器，可以直接通过 System manager 软件扫描到 EL72x1 下所带的电机型号。

2. NC 轴和物理轴的关系

可以通过 Axsi1-Settings-Link to 来选择 NC 轴所关联的物理轴，这个链接在扫描硬件的时候自动添加，也可以手动右键 Axes，点击 Append axis 添加轴，将 NC 轴手动链接到物理轴上，这个窗口可以看到 NC 轴和物理轴的对应关系。



展开 I/O Devices 中 AX5203 下的变量，可以看到驱动器下面的变量都已经有关联了，右键其中的一个变量，点击 Goto link variable，可以看到此变量和 NC 轴中的变量链接。NC 轴与物理轴就是通过这些变量来交换数据的，每个周期将驱动器的数据读取到 NC 中，NC 处理完再将控制命令传给驱动器。



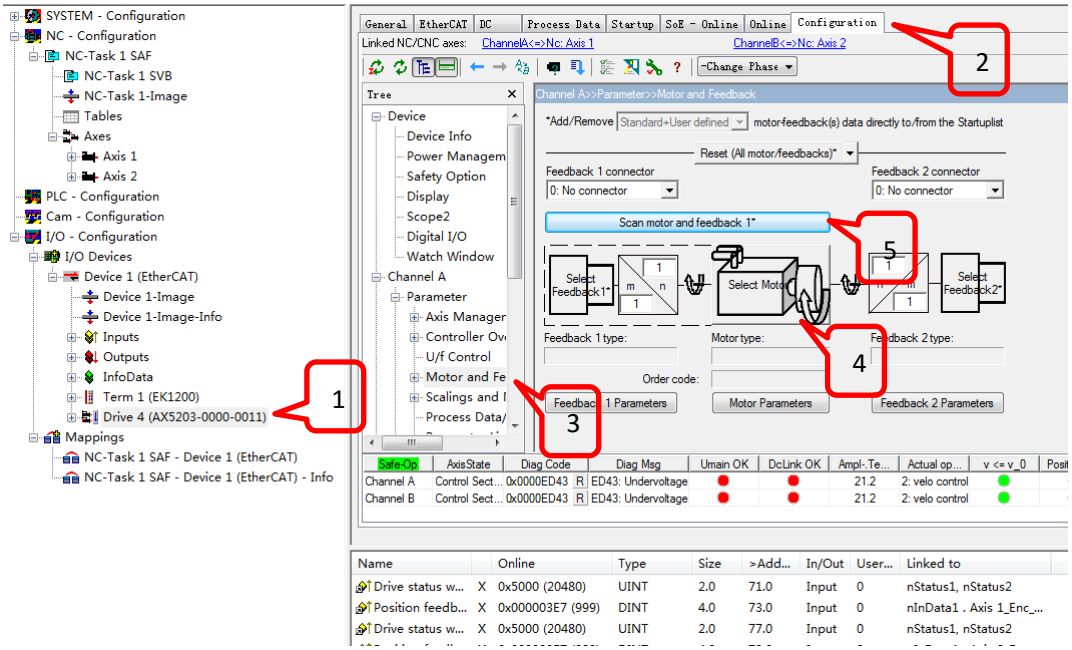
随堂问答：

学员提问：NC轴是否可以不链接到物理轴上？

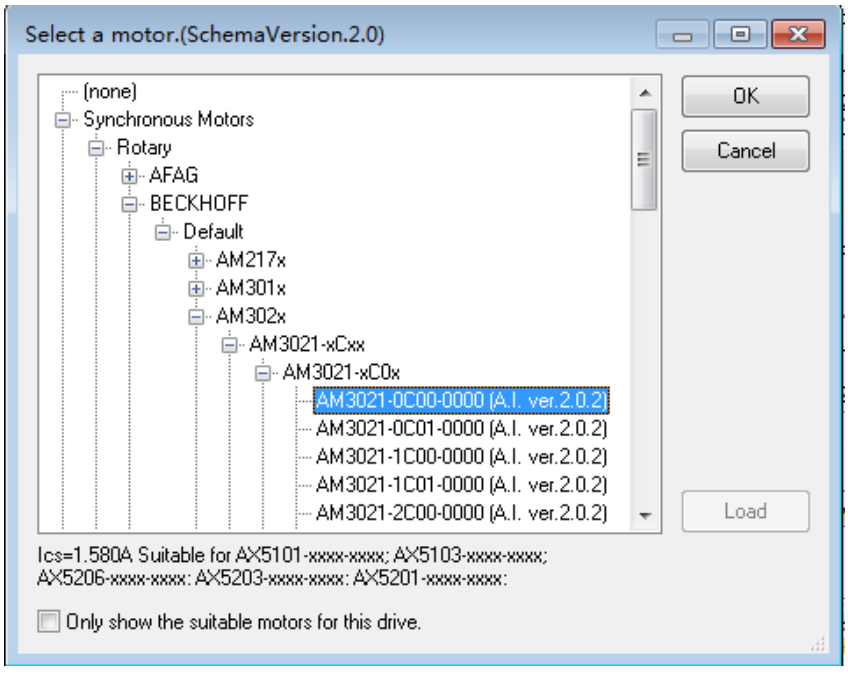
讲师解答：可以，不链接物理轴的 NC 轴是虚轴，如果现场还没有硬件设备，可以先添加虚轴调试程序，或者可以将虚轴做为耦合功能的主轴来使用。

3. AX5000 的配置

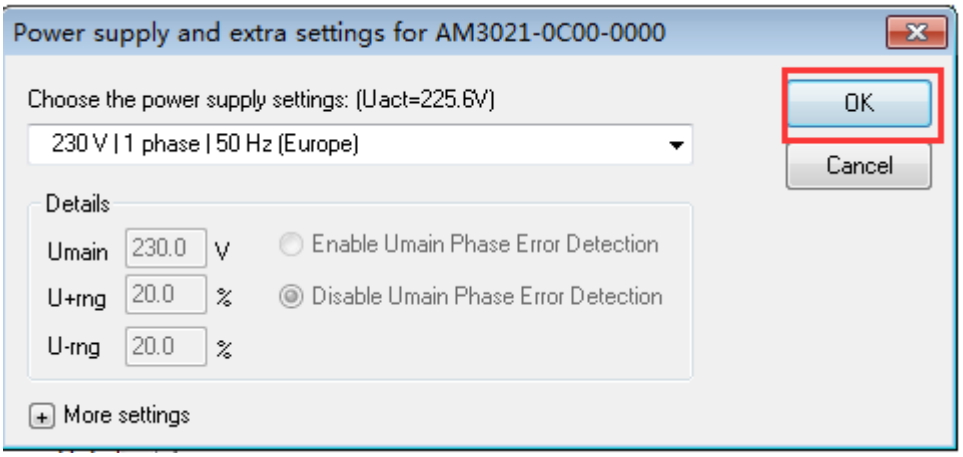
选中 Drive 4(AX5203-0000-0011)Configuration 选项卡，CHANNEL 下面的 MOTOR AND FEEDBACK，然后点击 Select Motor 来手动添加驱动器所带的电机型号，也可以点击 Scan motor and feedback 来自动获取电机型号，Configuration 是用来对 AX5000 驱动器进行配置的窗口，我们 AX5000 驱动器的配置软件没有额外的软件，直接通过 System manager 软件可以配置，并且不需要专门的电缆，只需要网线即可完成配置，有些电机无法通过自动扫描的方式获取型号，因此只能通过 Select Motor 手动选择电机。



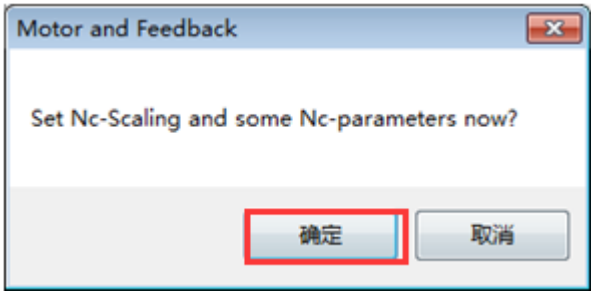
点击 select motor 之后，在弹出来的对话框中选择电机的型号



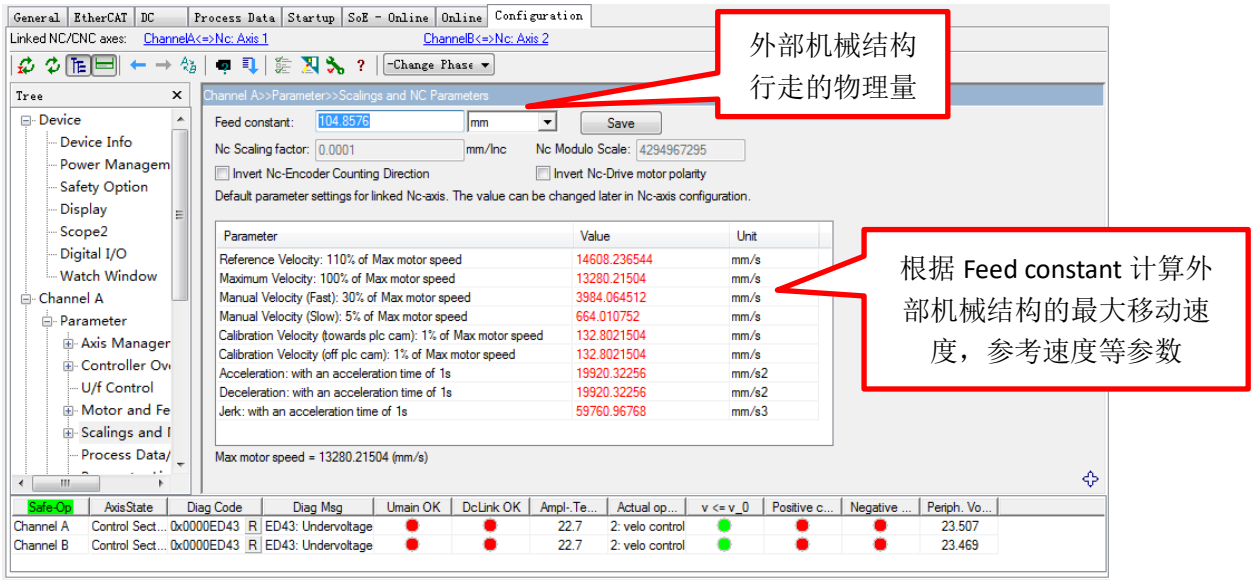
选择合适的电机型号之后点击 OK，接下来会提示如下窗口，选择驱动器实际的供电类型，点击 OK。



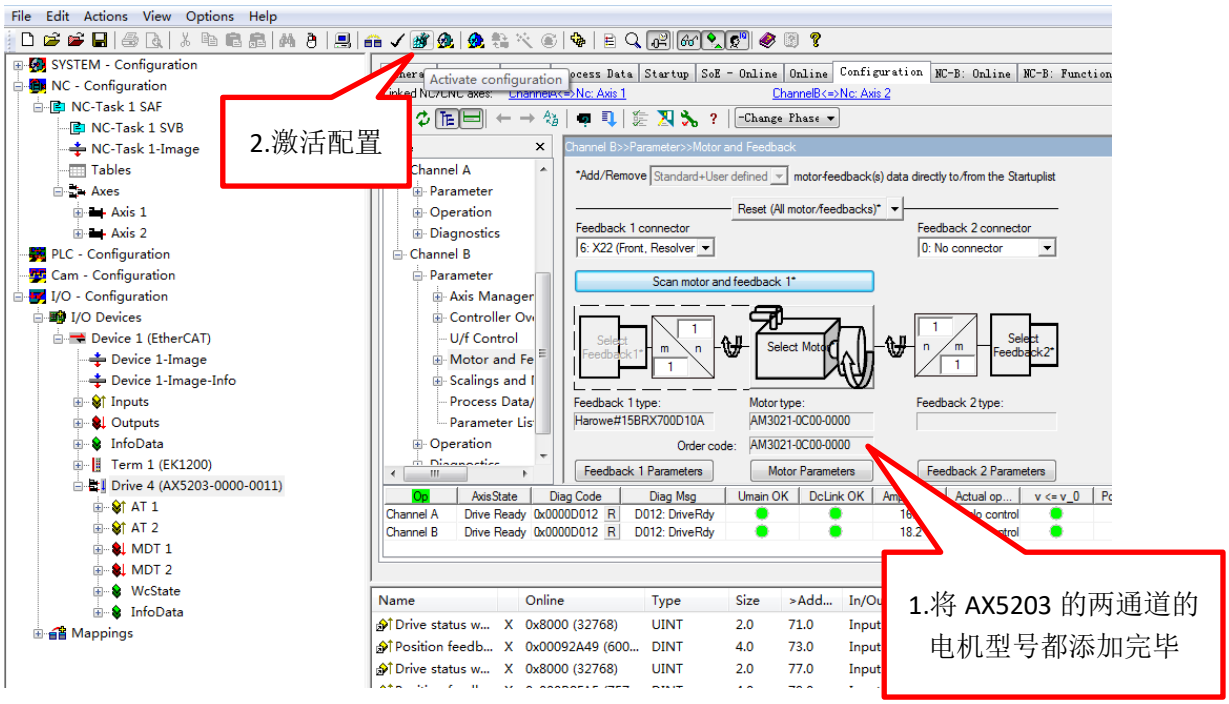
然后会提示是否设置 NC-标度以及一些 NC 的参数，这里可以点击确定或者取消。



如果点击确定，那么会提示如下窗口，这里主要是用来设置电机旋转一圈，实际外部机械结构行走的物理量，比如电机带了一个丝杠，如果电机旋转一圈，丝杠移动的位移为 20mm，那么就将 20mm 填入 Feed constant 这个参数里面，然后软件会自动根据 Feed constant 这个参数计算出丝杠移动的最大速度，参考速度等参数，点击 SAVE 即可保存，当然也可以不点 SAVE，用软件的默认参数，如果要让这些参数生效，那么一定要激活配置 (Activate configuration)。



将 CHANNEL B 的电机也选择好之后，然后激活配置 (activate configuration)



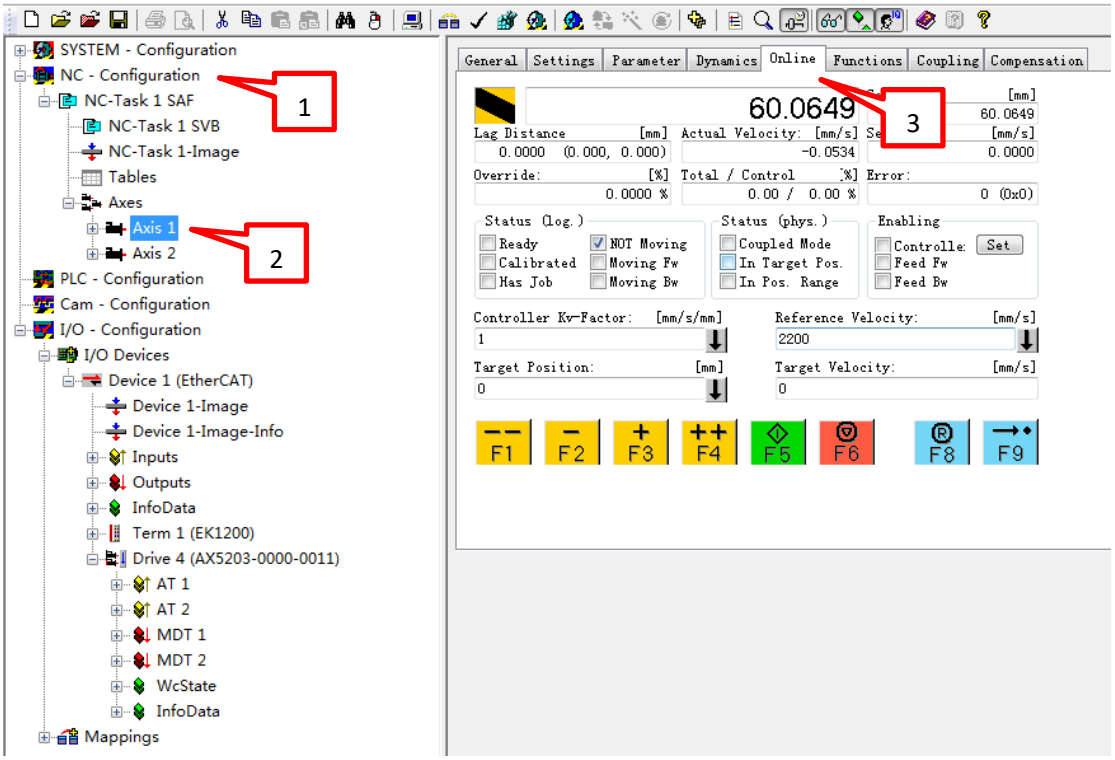
随堂问答：

学员提问：只需要添加电机型号与设置Feed constant就可以完成配置了吗？

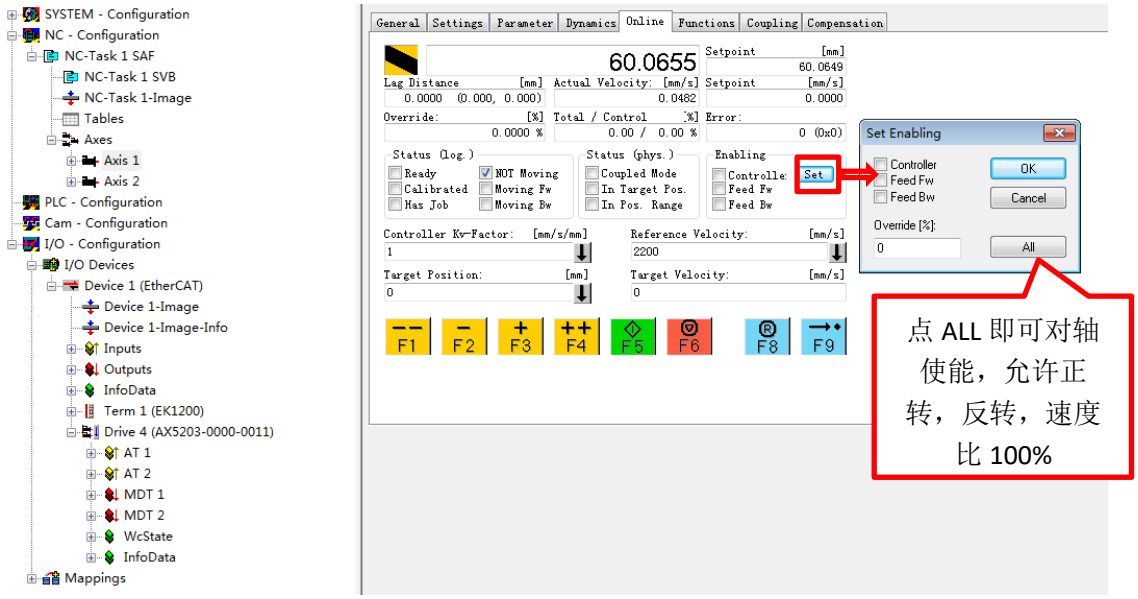
讲师解答：这只是最基本的配置，还需要调节驱动器的位置环，速度环的 PID 等参数（不支持自整定）。

4. AX5000 通过 system manager 软件调试

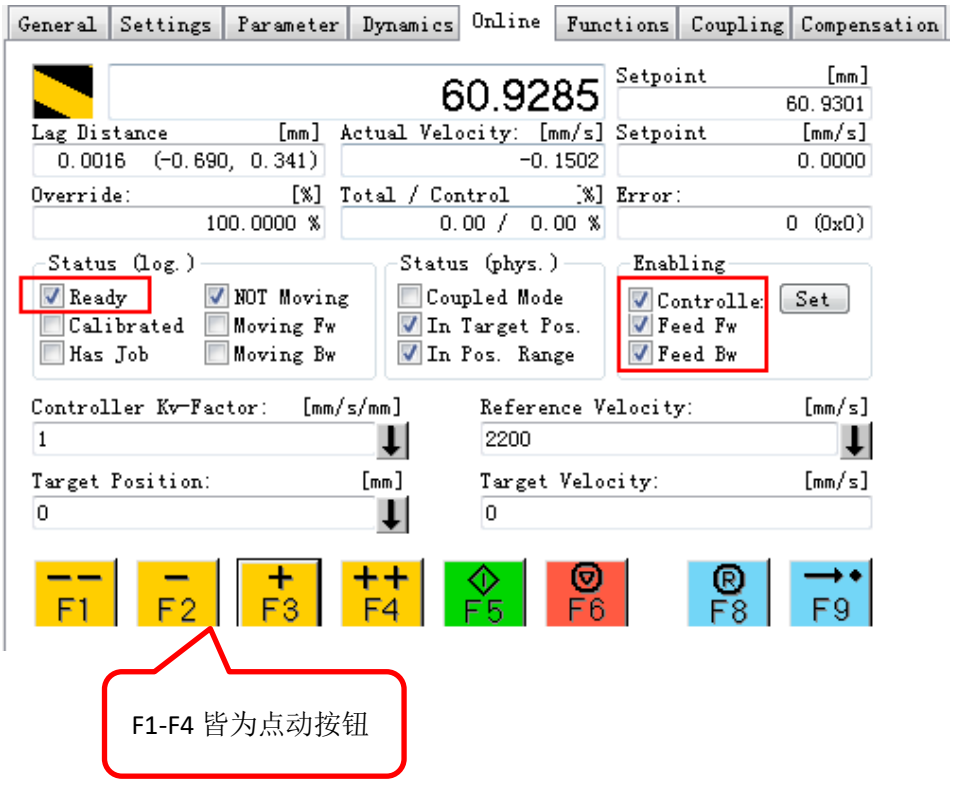
激活配置后，将 TwinCAT 切换到运行模式，然后点击 NC-configuration，点击 Axis1，点击 Online 选项卡，可以在这里对伺服轴进行调试（注：如果在 online 选项卡里面看不到轴的当前位置，那么请确保前面的电机型号添加以及激活配置等操作是否正常完成）



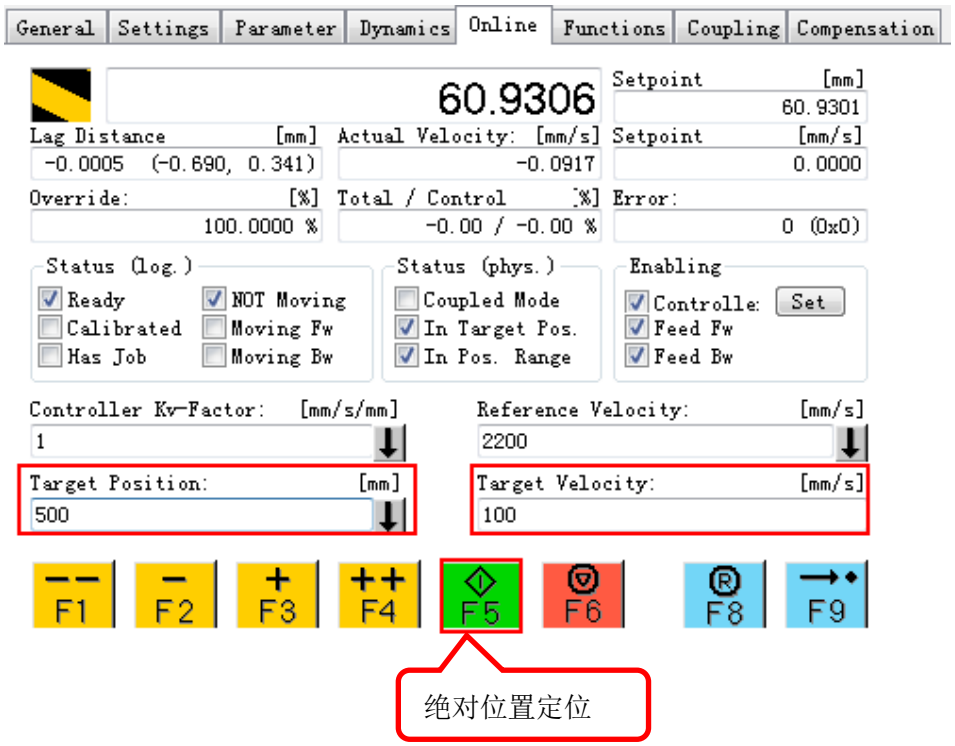
点击 SET，手动勾选 Controller，Feed Fw，Feed Bw，并设置 Override（速度比），然后点击 OK，或者直接点击 ALL 对轴进行使能，自动设置速度比为 100%。



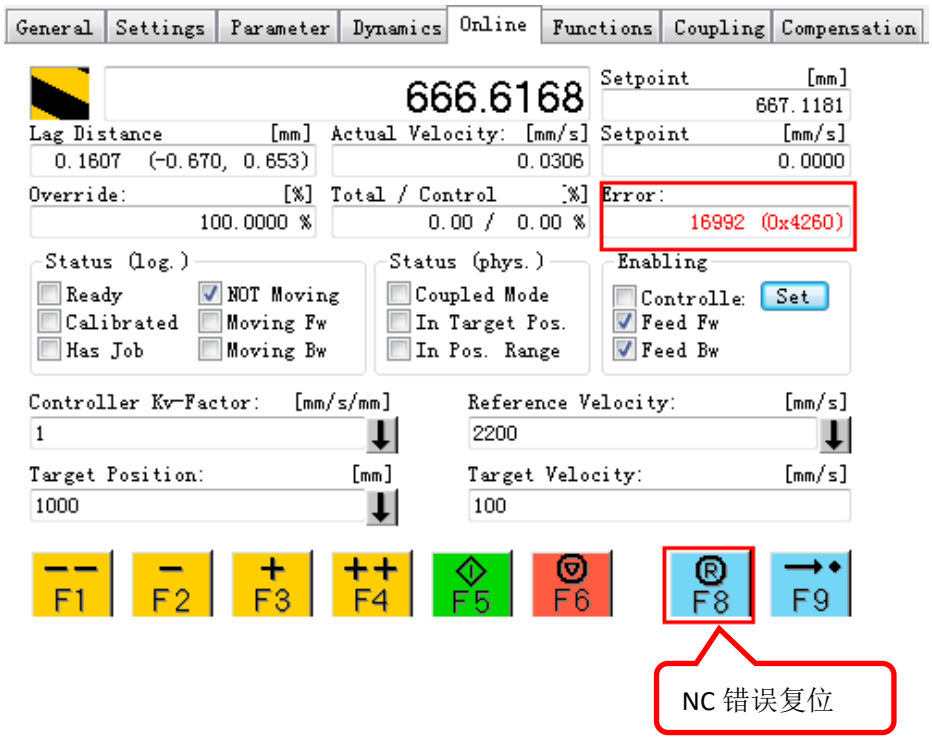
使能之后可以看到 Ready 状态会打勾，代表电机已使能，Controller,Feed Fw,Feed Bw 这些状态也会勾上，然后按下 F1 至 F4 即可对电机进行点动操作，按下 F1 点动，放开 F1 电机停止，点动速度在 parameter 选项卡中的 manual velocity 中设置，默认速度为 100mm/s 与 600mm/s，分别对应慢速点动和快速点动。



设置完 Target Position 和 Target Velocity 后按下 F5，即可实现位置控制，电机会以设定的目标速度走到目标位置，如当前位置 60，目标位置为 500，那么触发 F5 后，电机会从 60 的位置移动到 500，是绝对位置定位，定位的过程中可以使用 F6 停止。



当 NC 报错之后，Error 中会有错误代码，需要通过 F8 来对错误进行复位，否则轴无法继续动作，F9 是找原点的按钮，按下 F9 之后，轴位置会变成 99999……，并慢速移动，但是找原点的过程中需要一个外部的硬件信号做为原点信号，这个原点信号在 Online 窗口中无法捕捉，因此 F9 按钮一般不用，一般通过程序中编程来实现找原点的功能。

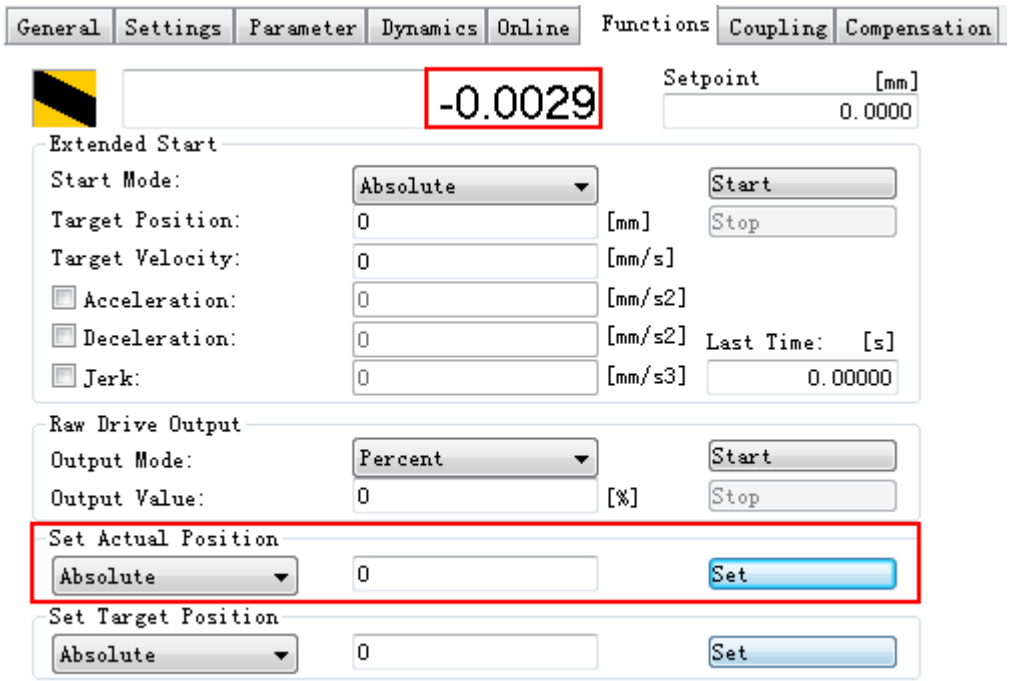


通过 Functions 里面的 Set Actual Position 可以修改轴的当前位置，如果将当前位置设置为 0，那么当前位置即为原点，此位置在 Twincat 重启之后会丢失，如果是绝对值编码器类型的反馈，那么重启之后以编码器的实际反馈位置为当前位置。

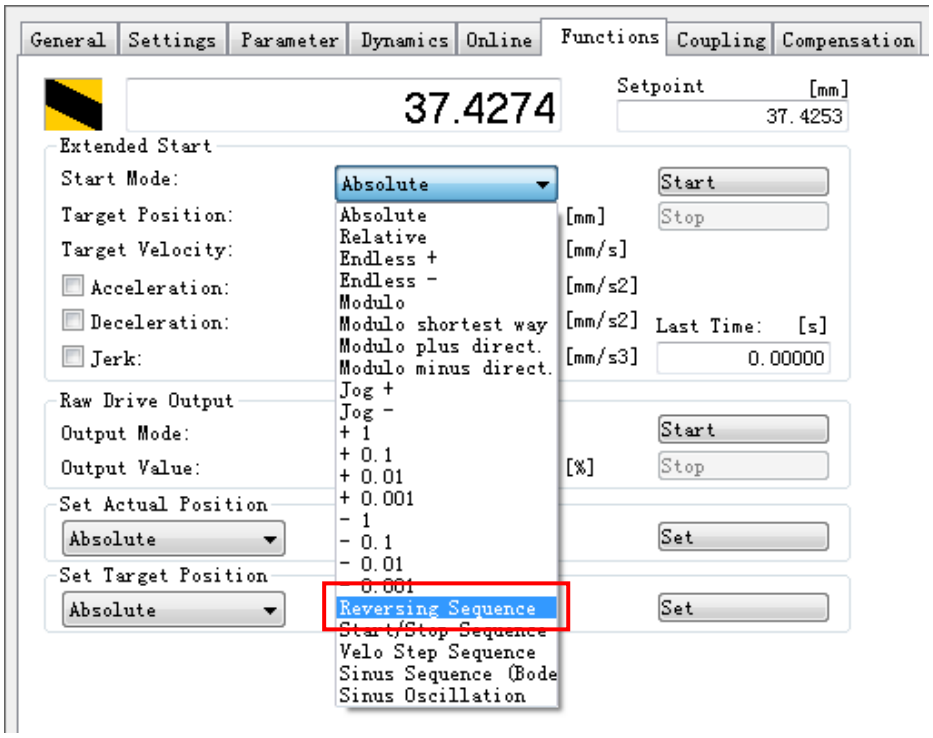
随堂问答：

学员提问：NC的Error代码16992如何查询对应的错误信息？

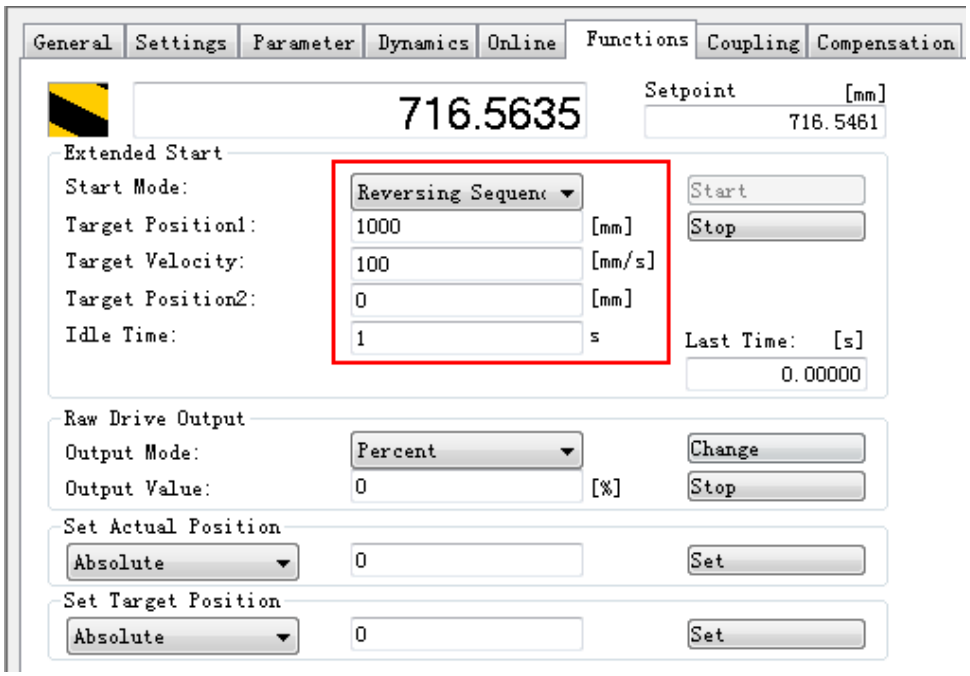
讲师解答：打开 Information system，搜索中输入 16992，点击列出主题即可搜到相关报错信息。



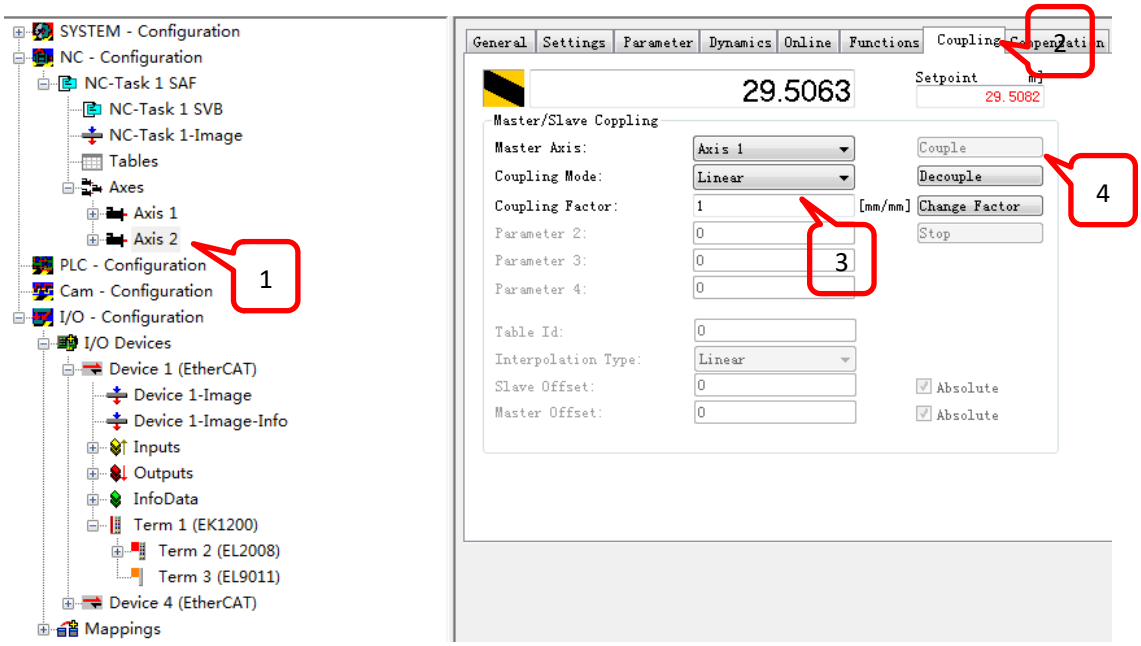
Functions——Start Mode 菜单中有很多对单轴的调试方法，常用的有 Absolute（绝对位置移动），Relative（相对位置移动），Endless+-（无限正反转），Modulo（模值移动），Reversing Sequence（往返序列），Start/Stop Sequence（启停序列），Velo Step Sequence（速度阶跃序列）



选择 Start Mode 为 Reversing Sequence，设置 Target Position1,Target Velocity,Target Position2,Idle Time（到达目标位置之后的等待时间）之后，点击 Start 即可让轴在 Position1 和 Position2 之间来回移动。

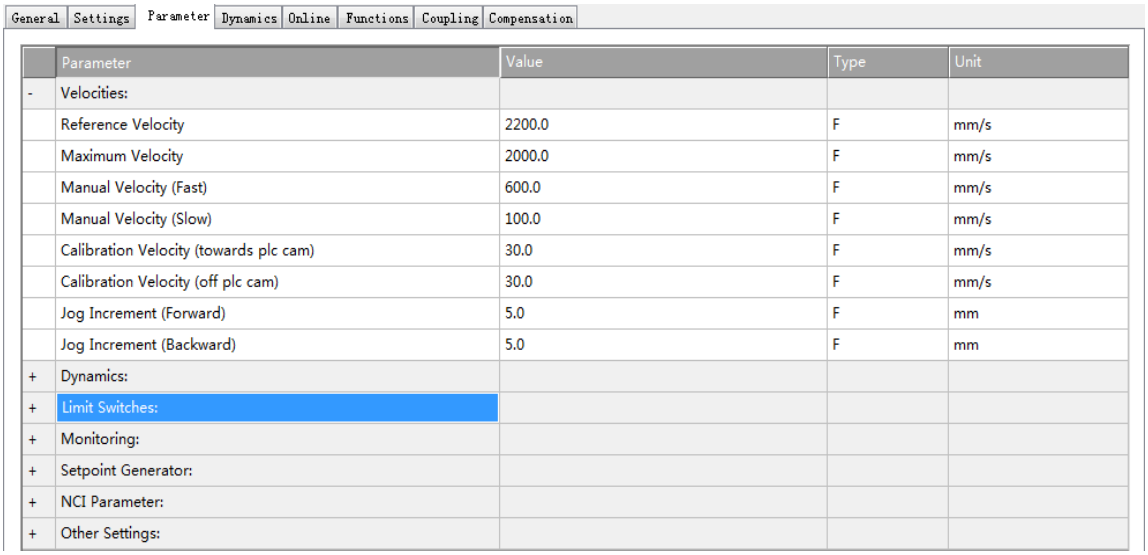


电子齿轮功能（主轴与从轴的速度保持比例关系，从轴跟随主轴移动）：首先将两根伺服轴都使能，然后选中 Axis2，Coupling 选项卡中，Master Axis 选择 Axis 1，Coupling mode 设置为 linear，Coupling Factor 设置为 1，然后点击 Couple 进行耦合，此时看到 Axis2 的 Setpoint 变为红色，代表 Axis2 已经作为从轴处于耦合状态，不能单独对 Axis2 进行控制了，此时控制 Axis1 轴动作的时候，Axis2 也会跟随动作，速度为 1:1，点击 Decouple 进行解耦，Change Factor 可以修改主从轴之间的速度比。



5. NC 参数设置

在 Parameter 选项卡中需要设置一些 NC 的参数，Reference Velocity 是参考速度，一般为 Maximum Velocity 的 110%，Maximum Velocity 是轴的最大速度，Manual Velocity(fast/slow)是点动的高速和慢速，Calibration Velocity 是寻参的速度，Dynamics 展开可设置加减速，Limit Switches 可以设置开启软限位，Monitoring 可以设置跟随误差的监视。



	Parameter	Value	Type	Unit
-	Velocities:			
	Reference Velocity	2200.0	F	mm/s
	Maximum Velocity	2000.0	F	mm/s
	Manual Velocity (Fast)	600.0	F	mm/s
	Manual Velocity (Slow)	100.0	F	mm/s
	Calibration Velocity (towards plc cam)	30.0	F	mm/s
	Calibration Velocity (off plc cam)	30.0	F	mm/s
	Jog Increment (Forward)	5.0	F	mm
	Jog Increment (Backward)	5.0	F	mm
+	Dynamics:			
+	Limit Switches:			
+	Monitoring:			
+	Setpoint Generator:			
+	NCI Parameter:			
+	Other Settings:			

Axis1——Axis 1_Enc 中的 Parameter 有 Scaling Factor 参数，用来进行定标，此参数比较重要，必须要设置，默认值是 0.0001。

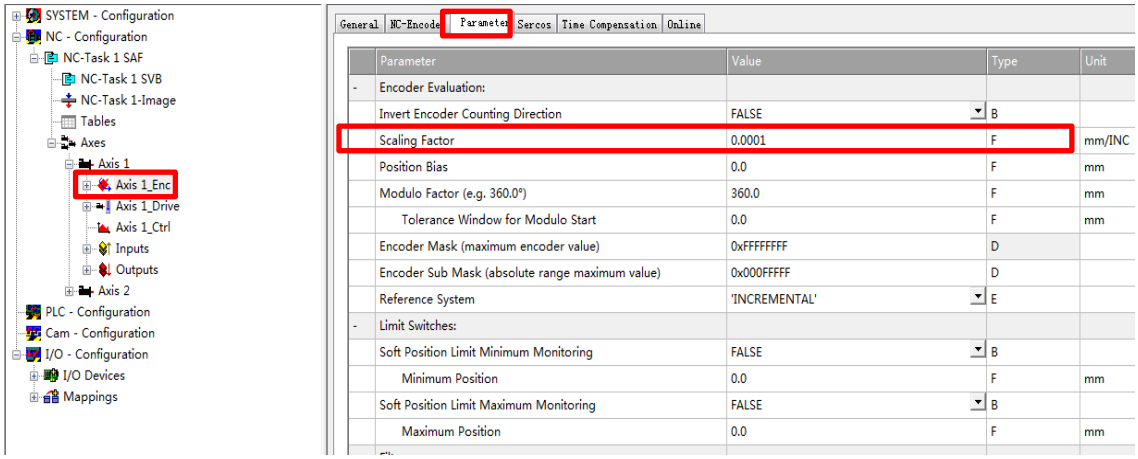
Scaling Factor=电机转一圈最终工件移动量/编码器反馈脉冲数

例如：电机转一圈，带动丝杠移动 5mm，AX5000 的编码器反馈为一圈 1048576，那么 scaling factor=5/1048576=0.00000476837158203125

例如：电机转一圈，带动一个圆形负载移动 360°，

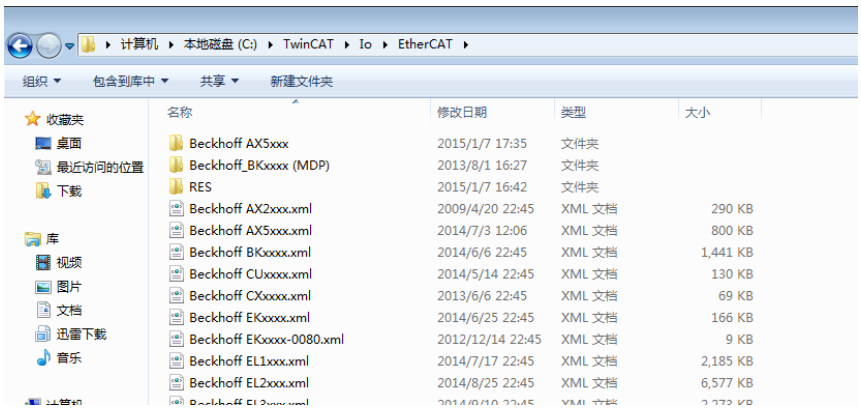
那么 scaling factor=360/1048576=0.00034332275390625（注：如用第三方伺服驱动器，那么编码器反馈不再是 1048576，需要根据第三方设备的实际反馈量来进行设置）

定标之后，NC 轴的位置和速度都是最终工件的位置和速度，用户可以直接通过 System Manager 或者编程控制最终工件，而不需要关注中间电机的转速和位置。



6. 添加第三方 EtherCat 总线的伺服驱动器

如果使用 BECKHOFF 的 PC 控制第三方的 EtherCAT 伺服驱动器，那么首先要将对方设备的从站描述文件——xml 文件拷贝到 C:\TwinCAT\Io\EtherCAT 路径中，然后重启 TwinCat 软件。

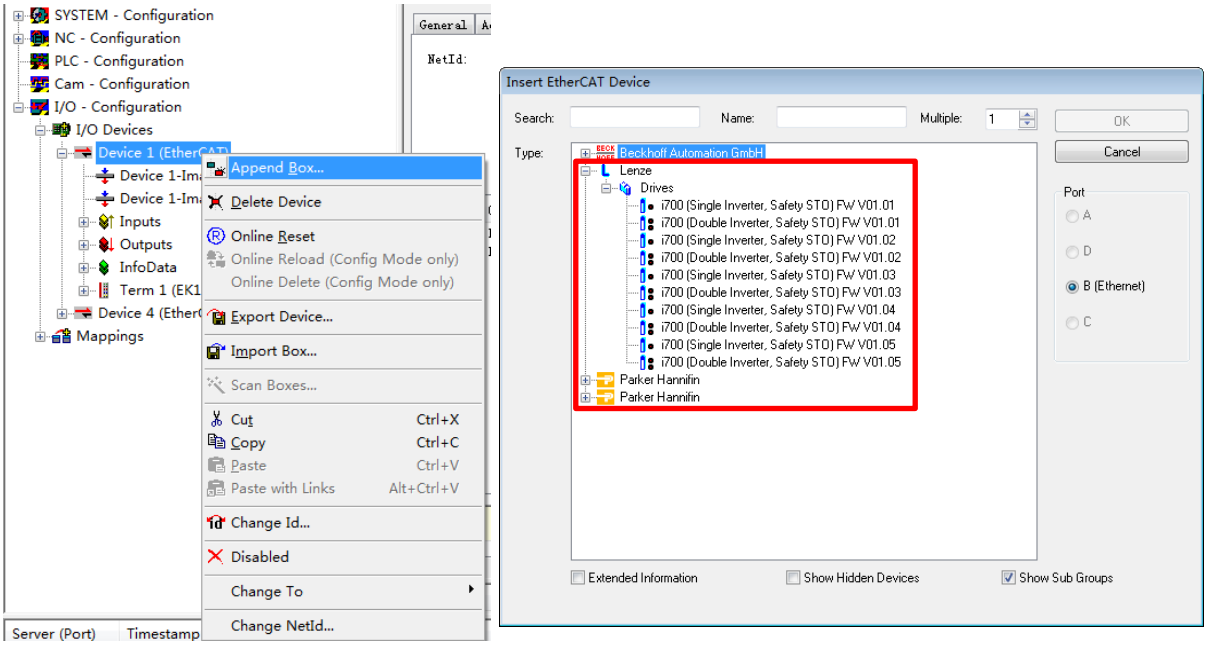


当 xml 文件拷贝到对应路径之后，可自动扫描到第三方的伺服驱动器，也可以手动添加第三方的伺服驱动器至 System manager 中进行配置。

随堂问答：

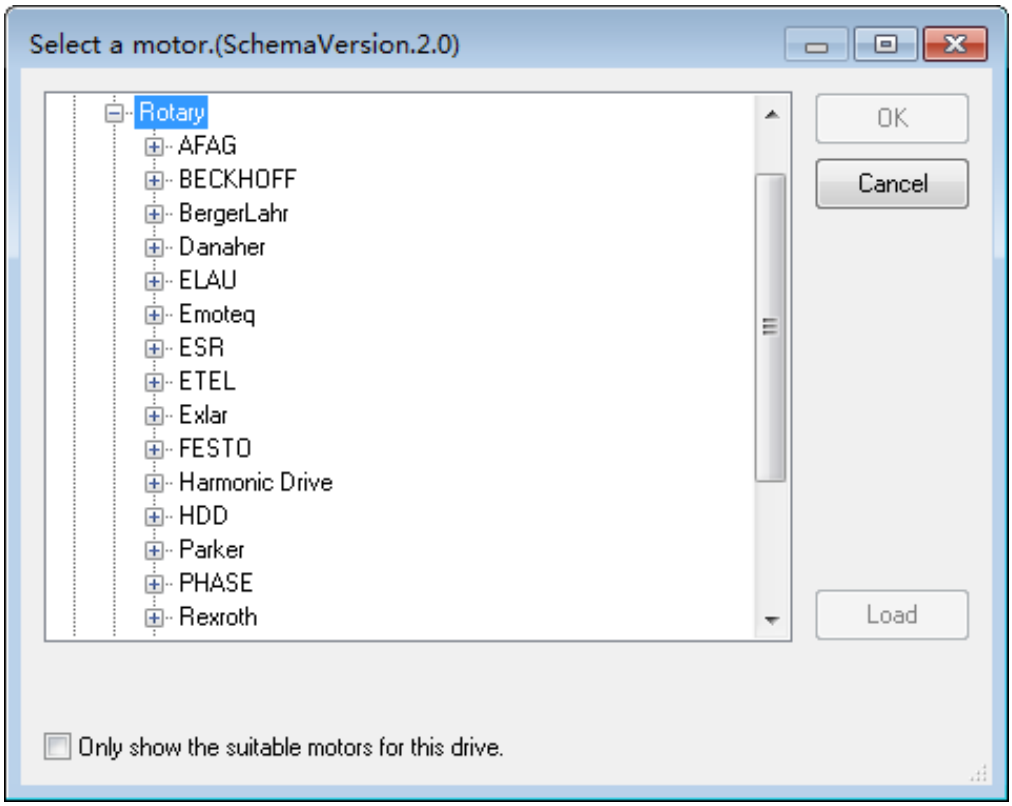
学员提问：CanOpen总线的驱动器如何添加？

讲师解答：将对方设备的 EDS 文件放到 C:\TwinCAT\Io\CanOpen 路径下即可。

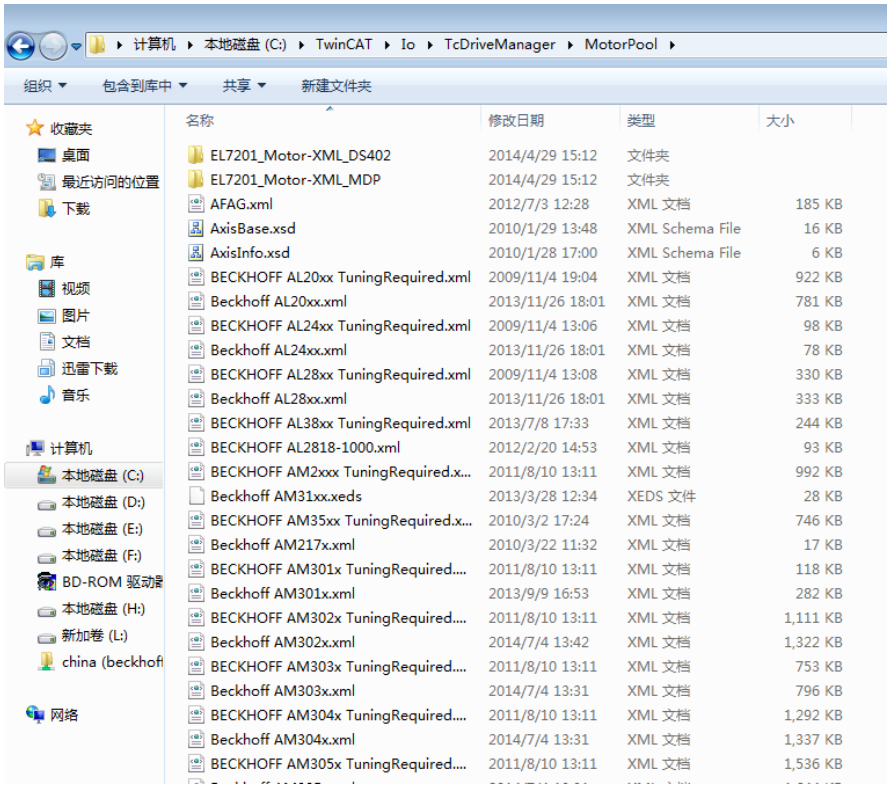


7. 添加第三方厂家的伺服电机

如果使用 AX5000 带第三方的伺服电机，并且此伺服电机可以在手动添加窗口中找到，那么选中此电机点击 OK 即可添加。



如果在手动添加电机的窗口中找不到此第三方电机，就需要填写“电机参数表”并提供给 BECKHOFF 技术工程师，BECKHOFF 工程师会制作此电机的 xml 文件给客户，客户将此电机文件放至 C:\TwinCAT\Io\TcDriveManager\MotorPool 路径中，重启 Twincat 软件即可，此时在手动添加电机的窗口中就可以找到该型号的电机了。（注：电机参数表需要问 BECKHOFF 的工程师索取）。



三． PLC Control 编程控制电机

本章目标：

通过本章节的学习，学员将了解：

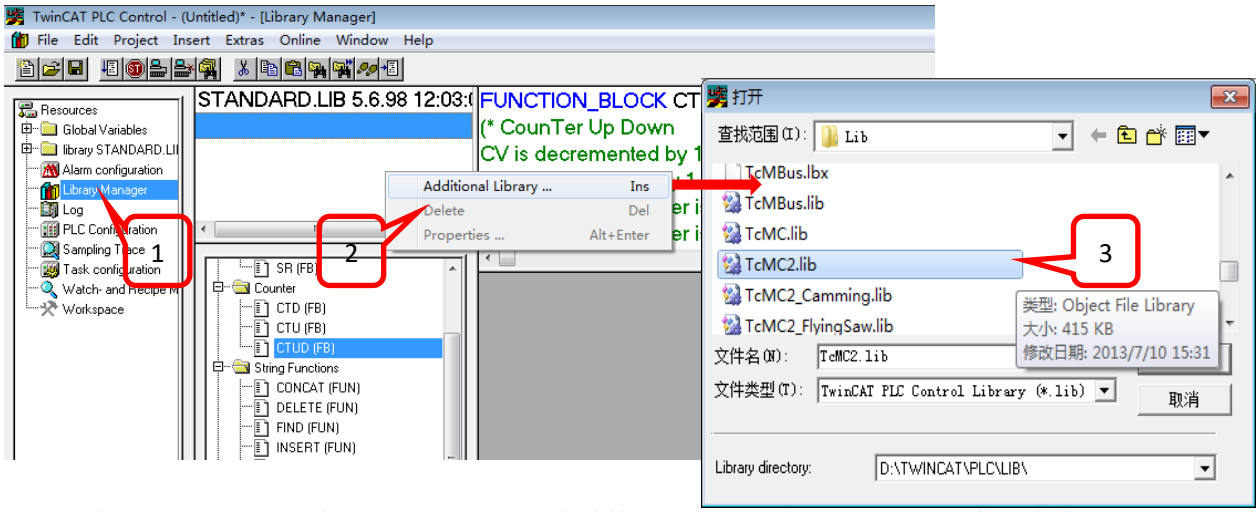
☑ NC PTP所需调用的库文件以及变量

☑ PLC轴变量和NC轴的链接

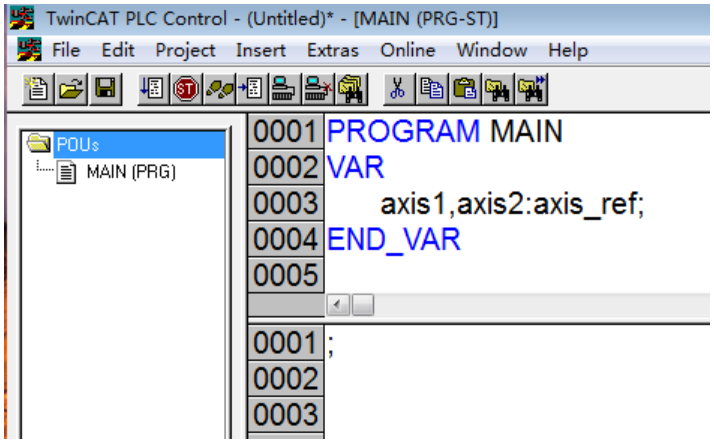
☑ 如何使用功能块对轴进行使能、点动、绝对定位、电子齿轮耦合、寻参等操作

1. 添加运动控制库文件以及轴类型变量

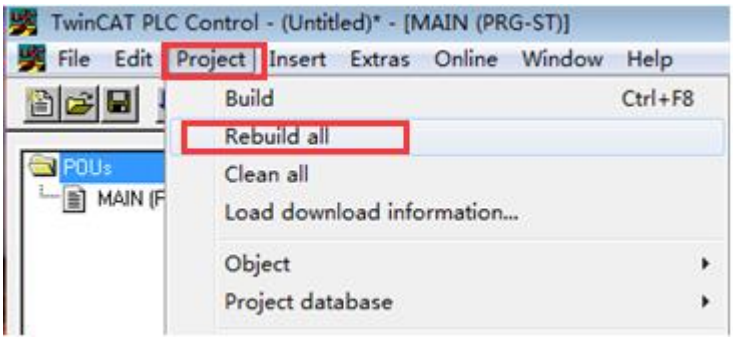
新建一个用 ST 语言编程的程序，然后在左下角工程栏 resource 选项卡中，双击 library manager，然后右键空白处点击 additional library，加载一个 TCMC2.LIB 的库文件，也有 TCMC.LIB 这个库文件，这个库文件是比较老的项目所用的库文件，这里不推荐新项目使用老的库文件编程。



新建两个 Axis_ref 类型的变量，axis_ref 是一个结构体，主要用来做 NC 和 PLC 数据交换用的，内部又嵌套了另外一些结构体，我们将 axis_ref 类型的变量称之为轴类型的变量，另外在程序编写窗口中输入一个“;”，否则编译会报错。

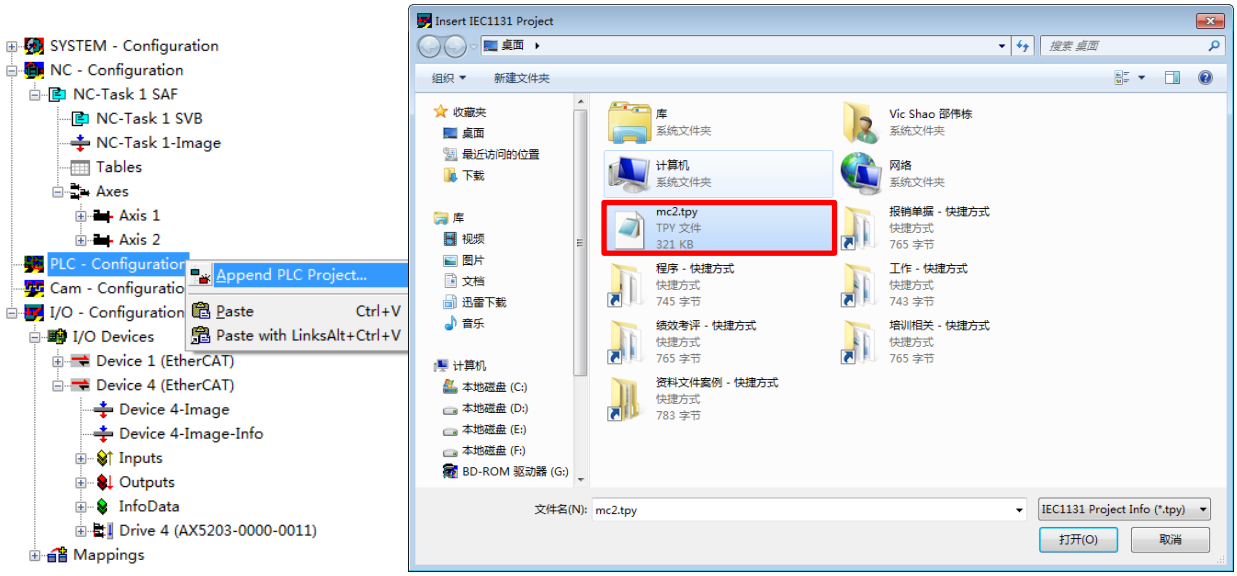


程序写完之后先“另存为”然后再“编译”，可以将程序保存到桌面或者手动指定一个文件夹，目的是生成一个 TPY 文件来做变量链接。（注：编译时注意是否有错误，有错误则无法生成 TPY 文件）

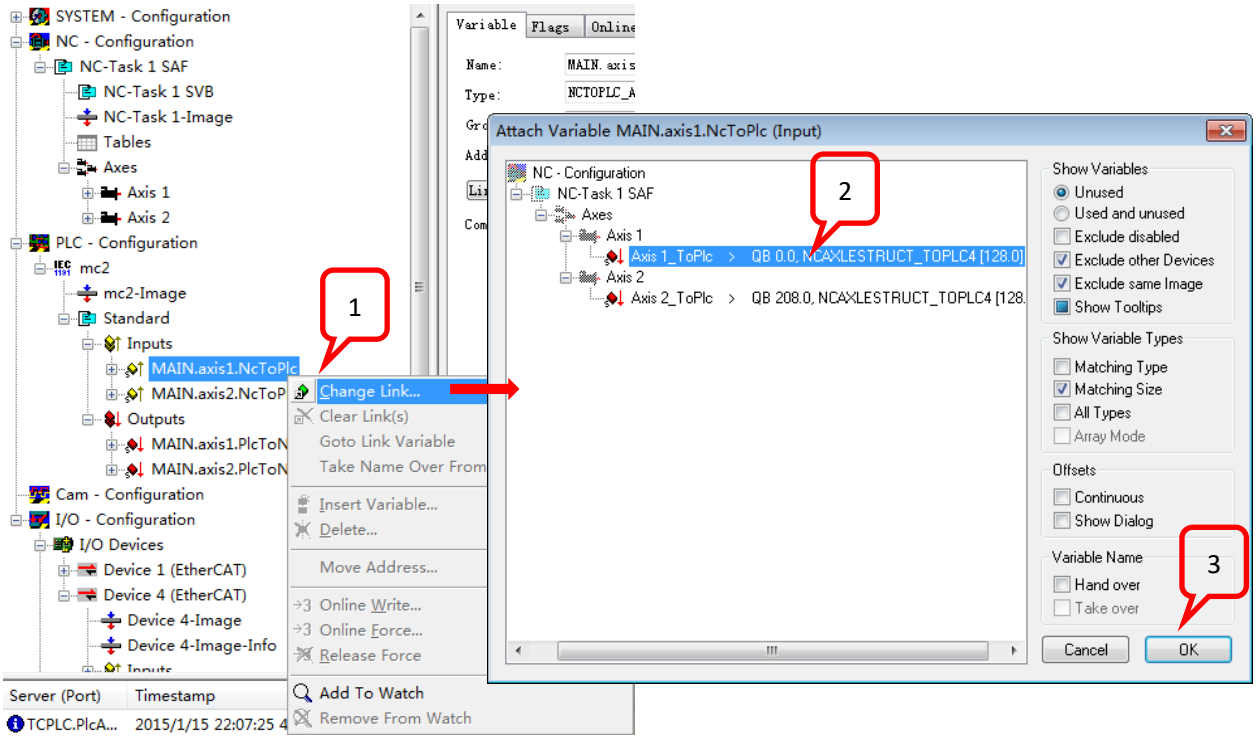


2. NC 与 PLC 的变量链接

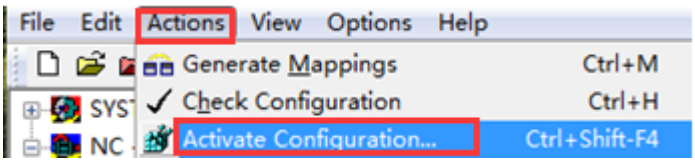
回到 system manager 软件，右键 PLC-Configuration，点击 Append PLC Project，找到前面生成的 TPY 文件进行添加。



然后将 PLC-Configuration 中的变量展开，依次右键 MAIN.AXIS1.NCTOPLC, MAIN.AXIS1.PLCTONC, MAIN.AXIS2.NCTOPLC, MAIN.AXIS2.PLCTONC 选择 chang link，将这些变量链接到 NC 轴下面的 Axis1_ToPlc,Axis1_FromPlc,Axis2_ToPlc,Axis2_FromPlc 变量中，NC 和 PLC 通过以上的链接交换数据，NC 将驱动器的状态位置反馈给 PLC，PLC 将功能块的控制数据写给 NC。

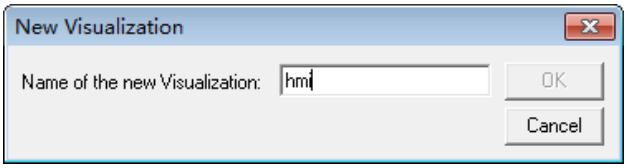


链接之后激活配置，并将 TwinCAT 重启为运行模式。

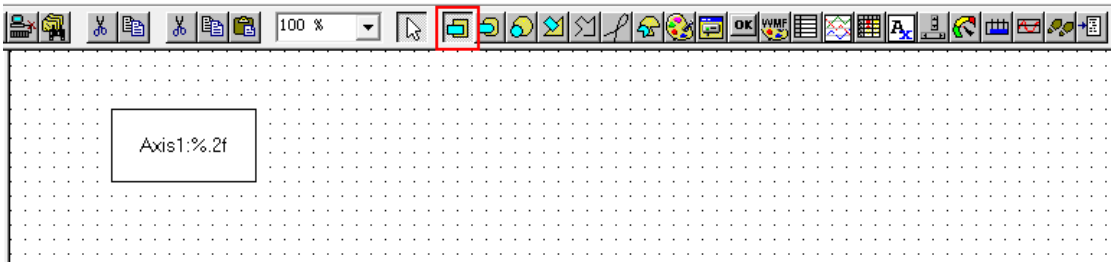


3. 调用功能块控制轴使能点动

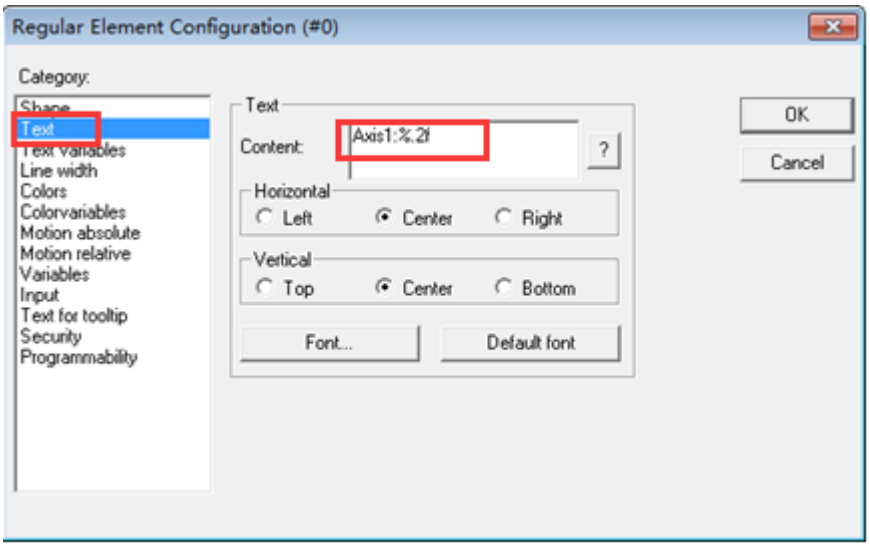
回到 Plc control 软件窗口后，点击左下角的 visualization 选项卡，右键空白的地方添加一个 hmi 画面，通过 Hmi 可以对轴进行更方便的调试。



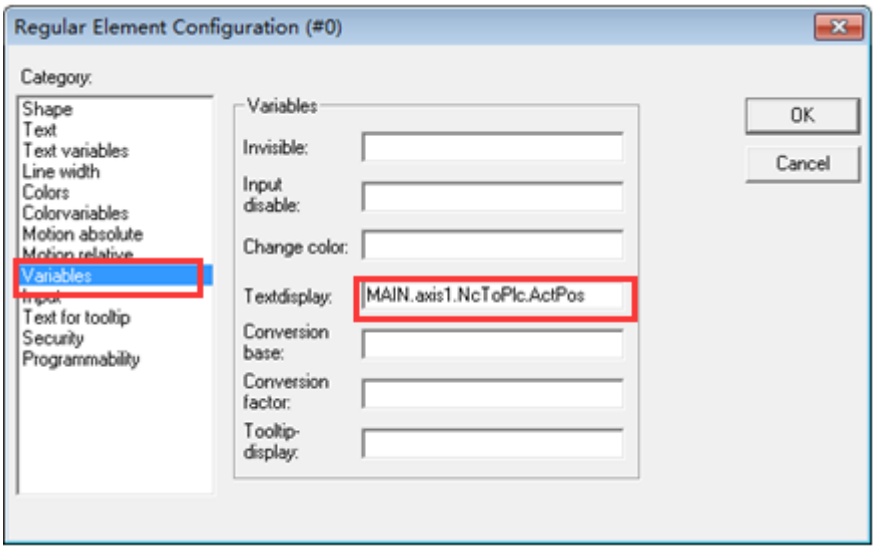
选中矩形框并在 HMI 里面拖出一个控件，双击矩形框控件进行设置。



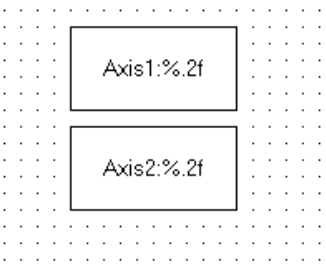
将矩形框里面的参数设置如下：Text-Content 里面输入 Axis1:%.2f，%.2f 代表以浮点数的数据类型显示关联变量（Variable-Textdisplay 所指向的变量）的值，并只保留小数点后两位。



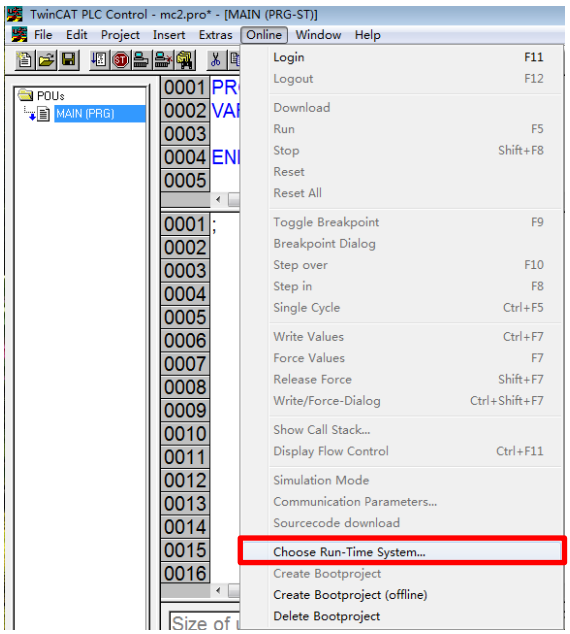
Variables 里面的 Textdisplay 中设置所关联的变量，这里选择轴 1 的实际位置。



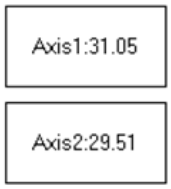
再做一个矩形框控件用来显示轴 2 的实际位置。



程序写好后需要 Login，在 login 之前选好目标设置，online-choose runtime system，然后将程序运行起来，选好 runtime 之后可以在 Plc control 下面状态看下所连的目标设备（Target:CX-17A5EC）



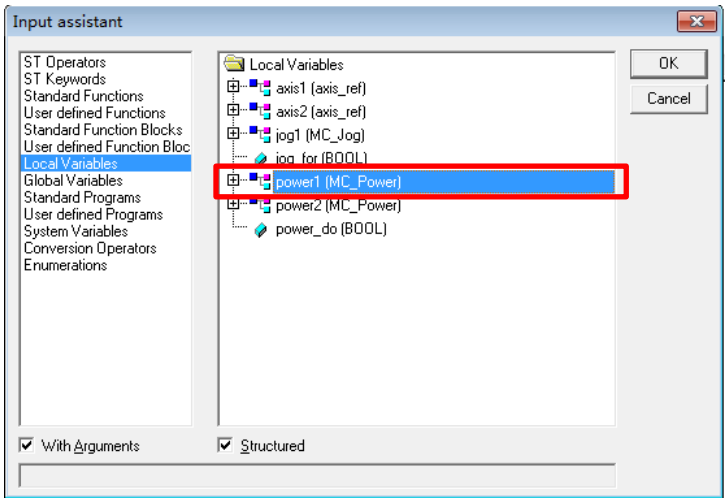
Login 之后把程序运行起来即可看到轴 1 和轴 2 的当前位置显示在 HMI 上面。



声明两个 MC_POWER 的功能块以及一个 MC_JOG 的功能块，MC_POWER 用来使能，MC_JOG 用来点动。

```
PROGRAM MAIN
VAR
    axis1,axis2:axis_ref;
    power1,power2:MC_Power;
    jog1:MC_Jog;
END_VAR
```

在程序编写窗口中按 F2，在 Local Variable 中选择 power1 点击 OK，将功能块调用到程序里面来，之后用同样的方法在程序中调用 Power2 与 Jog1 功能块。



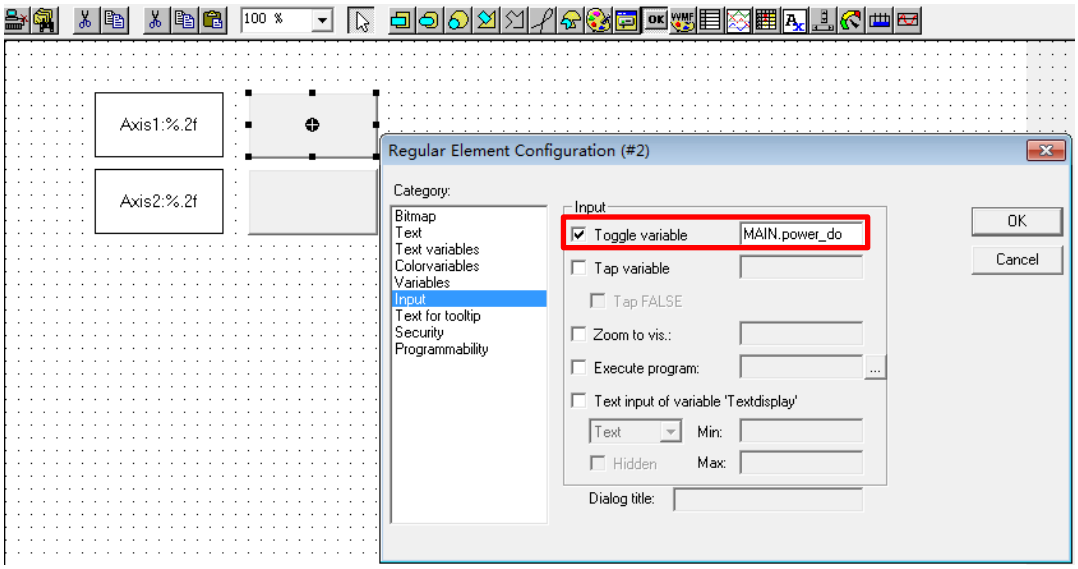
将功能块里面的参数填写完整，Enable 代表使能触发位，Enable_Positive 代表允许正转，Enable_Negative 代表允许反转，Override 代表速度比，Axis 代表对哪个轴进行操作。JogForward 代表正向点动位，另外再声明两个 bool 类型变量（power_do 和 jog_for）做为使能与点动功能块的触发位。

```
power1(
    Enable:=power_do ,
    Enable_Positive:= TRUE,
    Enable_Negative:=TRUE ,
    Override:=100 ,
    BufferMode:= ,
    Axis:=axis1 ,
    Status=> ,
    Busy=> ,
    Active=> ,
    Error=> ,
    ErrorID=> );
```

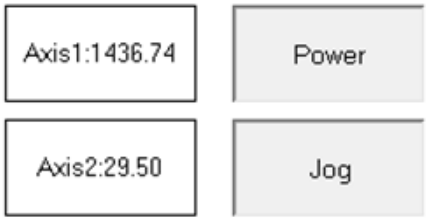
```
power2(
    Enable:=power_do ,
    Enable_Positive:=TRUE ,
    Enable_Negative:=TRUE ,
    Override:= 100,
    BufferMode:= ,
    Axis:=axis2 ,
    Status=> ,
    Busy=> ,
    Active=> ,
    Error=> ,
    ErrorID=> );
```

```
jog1(
    JogForward:= jog_for,
    JogBackwards:= ,
    Mode:= ,
    Position:= ,
    Velocity:= ,
    Acceleration:= ,
    Deceleration:= ,
    Jerk:= ,
    Axis:=axis1 ,
    Done=> ,
    Busy=> ,
    Active=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> );
```

HMI 中加入两个按钮控件，用来对轴进行使能以及点动，第一个按钮关联 MAIN.power_do 变量，第二个按钮关联 MAIN.jog_for 变量，两个按钮都选择 Toggle variable 类型（交替按钮），在按钮的 text 里面加上标签 Power 与 Jog。



首先将 Power 按钮按下，对轴进行使能，可以在 system manager 的 online 窗口查看轴的 ready 状态有没有勾选来判断轴是否已经使能上了，然后按下 Jog 即可看到轴在转动，再次按下 Jog 可以看到轴停止（注：先使能再点动，点动的时候不能撤除使能信号，否则轴会报错）



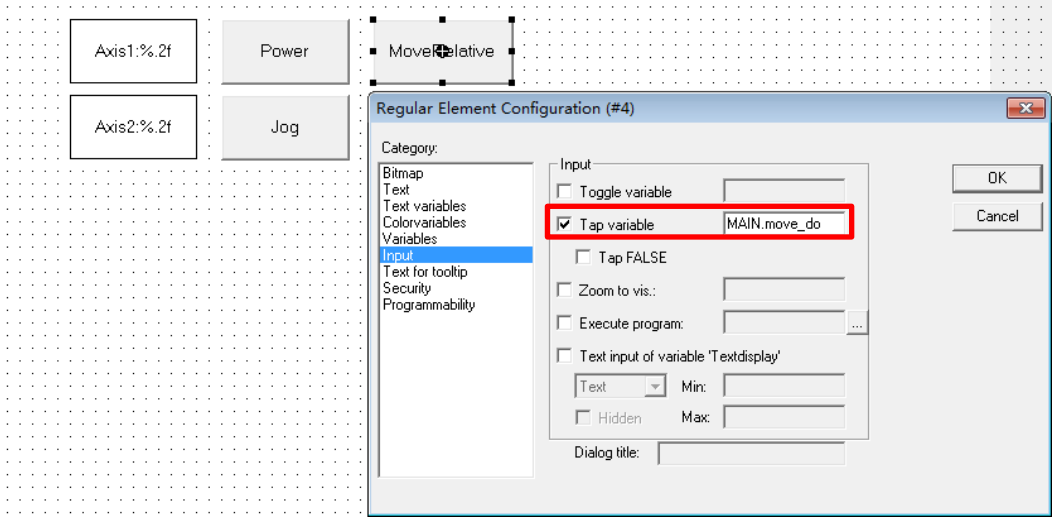
4. 调用功能块控制轴走相对位置

声明一个 MC_MoveRelative 功能块对轴进行位置控制，依然是按 F2 在程序编写窗口中调用功能块，然后将对应的参数填写完整，Execute 是功能块触发位，distance 和 velocity 是移动的距离以及速度，axis 代表控制哪根轴。

```
0006    power_do: BOOL;
0007    jog_for: BOOL;
0008    move_r:MC_MoveRelative;
0009    move_do: BOOL;
0010 END_VAR
```

```
0041 move_r(
0042    Execute:= move_do,
0043    Distance:= 1000,
0044    Velocity:=100 ,
0045    Acceleration:= ,
0046    Deceleration:= ,
0047    Jerk:= ,
0048    BufferMode:= ,
0049    Options:= ,
0050    Axis:= axis1,
0051    Done=> ,
0052    Busy=> ,
0053    Active=> ,
0054    CommandAborted=> ,
0055    Error=> ,
0056    ErrorID=> );
```

依然在 hmi 里面新建一个按钮，通过按钮来触发 Mc_MoveRelative 功能块。



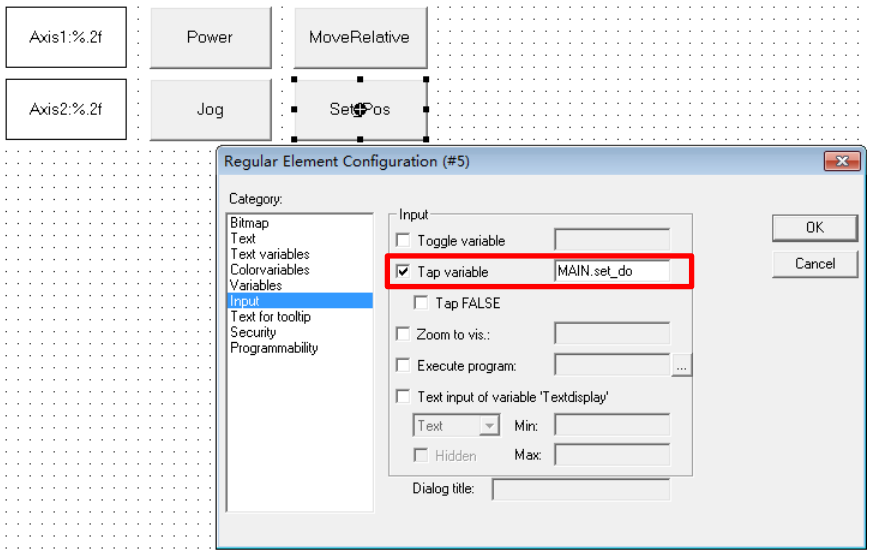
将程序 login 并 run，轴使能后按下 MoveRelative 按钮，即可让轴移动 1000 个位置，再次按下 MoveRelative，轴依然移动 1000 个位置。

5. 调用功能块控制轴改变当前位置

MC_SetPosition 可以设置轴的当前位置，声明功能块后，在编程编写窗口中调用这个功能块，并将功能块的参数填写完整，Position 设置为 0，触发功能块即将当前位置改为原点。

```
0010    setpos1,setpos2:MC_SetPosition;
0011    set_do: BOOL;
0012 END_VAR
0013
0057 setpos1(
0058    Execute:=set_do ,
0059    Position:=0 ,
0060    Mode:= ,
0061    Options:= ,
0062    Axis:= axis1,
0063    Done=> ,
0064    Busy=> ,
0065    Error=> ,
0066    ErrorID=> );
0067 setpos2(
0068    Execute:=set_do ,
0069    Position:=0 ,
0070    Mode:= ,
0071    Options:= ,
0072    Axis:= axis2,
0073    Done=> ,
0074    Busy=> ,
```

HMI 中加入一个按钮用来触发 Mc_SetPosition 功能块。



将程序运行，按下 Set_Pos 按钮之后即可设置轴 1 和轴 2 的当前位置为 0。

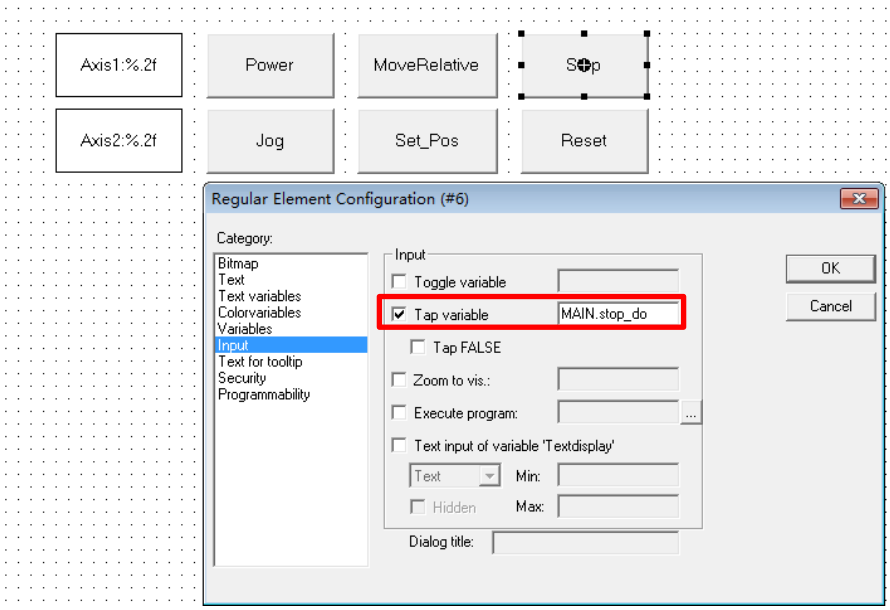
6. 调用功能块控制轴停止以及复位

MC_Stop 对轴进行停止，MC_Reset 对轴进行复位，声明之后在主程序中调用功能块，并将功能块的参数设置好。

```
0012      stop:MC_Stop;
0013      reset:MC_Reset;
0014      stop_do:BOOL;
0015      reset_do:BOOL;
0016  END_VAR

0077  stop(
0078      Execute:=stop_do ,
0079      Deceleration:= ,
0080      Jerk:= ,
0081      Options:= ,
0082      Axis:= axis1,
0083      Done=> ,
0084      Busy=> ,
0085      Active=> ,
0086      CommandAborted=> ,
0087      Error=> ,
0088      ErrorID=> );
0089  reset(
0090      Execute:= reset_do,
0091      Axis:= axis1,
0092      Done=> ,
0093      Busy=> ,
0094      Error=> ,
0095      ErrorID=> );
```

HMI 里面加入两个按钮，一个对轴进行停止，一个对轴进行复位，当轴动作的时候可以通过 Stop 按钮来停止，当 NC 轴报错的时候可以通过 Reset 来复位。



7. 调用功能块控制两轴电子齿轮耦合

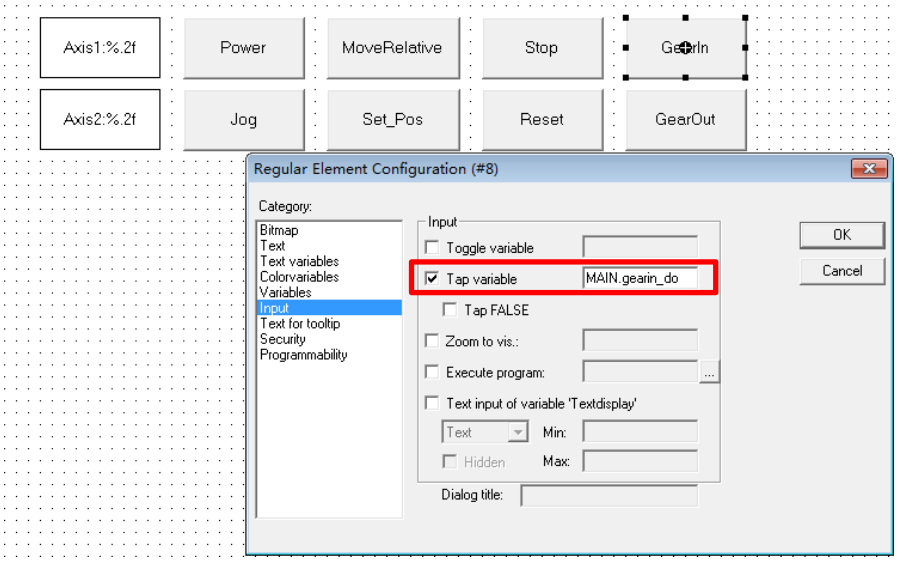
电子齿轮需要两个功能块，一个是耦合 MC_GearIn,一个是解耦 MC_GearOut，分别将两个功能块调用到主程序中，RatioNumerator 代表从轴的速度，RatioDenominator 代表主轴的速度，如 RatioDenominator 设为 2，RatioNumerator 设为 1，那么主轴速度是从轴的两倍，MASTER 设置哪个轴为主轴，Slave 设置哪个轴为从轴。

```
0016    gearin:MC_GearIn;
0017    gearout:MC_GearOut;
0018    gearin_do: BOOL;
0019    gearout_do: BOOL;
```

```
0096 gearin(
0097    Execute:=gearin_do ,
0098    RatioNumerator:= 1,
0099    RatioDenominator:=1 ,
0100    Acceleration:= ,
0101    Deceleration:= ,
0102    Jerk:= ,
0103    BufferMode:= ,
0104    Options:= ,
0105    Master:=axis1 ,
0106    Slave:=axis2 ,
0107    InGear=> ,
0108    Busy=> ,
0109    Active=> ,
0110    CommandAborted=> ,
0111    Error=> ,
0112    ErrorID=> );
```

```
0113 gearout(
0114    Execute:= gearout_do,
0115    Options:= ,
0116    Slave:=axis2 ,
0117    Done=> ,
0118    Busy=> ,
0119    Error=> ,
0120    ErrorID=> );
```

HMI 中加入两个按钮（Tap variable），一个按钮控制耦合，一个按钮控制解耦。



将程序登录进去并运行起来，首先通过 POWER 将两个轴使能，然后按下 GearIn 进行耦合，再按下 Jog 按钮后可以看到两个轴以 1:1 的速度转动，将 Jog 复位后，按下 GearOut 进行解耦。

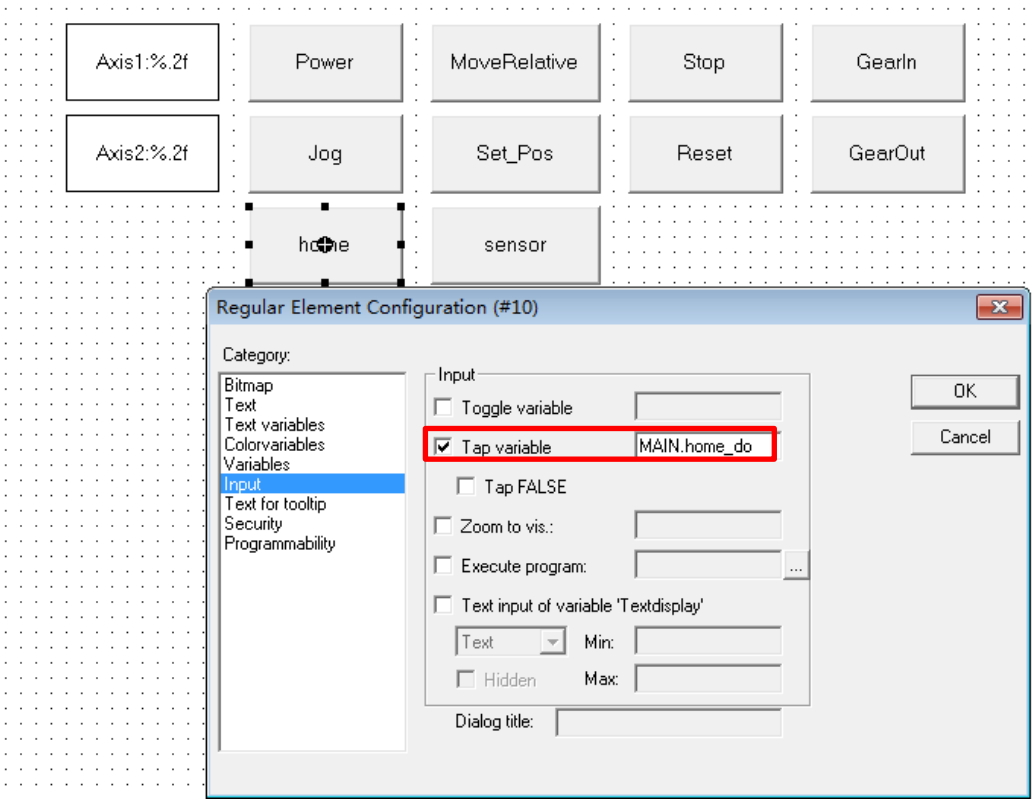
8. 调用功能块控制轴寻参

MC_home 功能块可以定位原点，home_do 是功能块的触发位，sensor 是外部接近开关的触发信号，可以用 hmi 的按钮来代替，或者链接到外部输入点，当轴碰到接近开关信号之后，NC 轴的位置变为 Position 参数中设置的值。

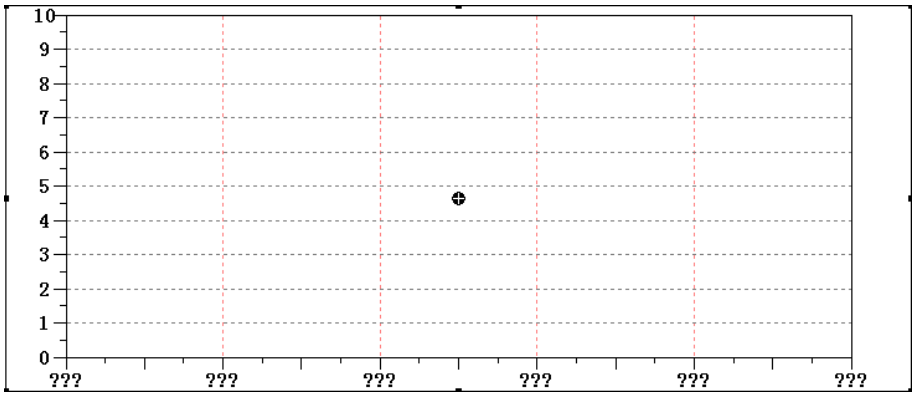
```
0021     home:MC_Home;
0022     home_do:BOOL;
0023     sensor:BOOL;
0024 END_VAR

0121 home(
0122     Execute:=home_do ,
0123     Position:=0 ,
0124     HomingMode:= ,
0125     BufferMode:= ,
0126     Options:= ,
0127     bCalibrationCam:=sensor ,
0128     Axis:=axis1 ,
0129     Done=> ,
0130     Busy=> ,
0131     Active=> ,
0132     CommandAborted=> ,
0133     Error=> ,
0134     ErrorID=> );
```

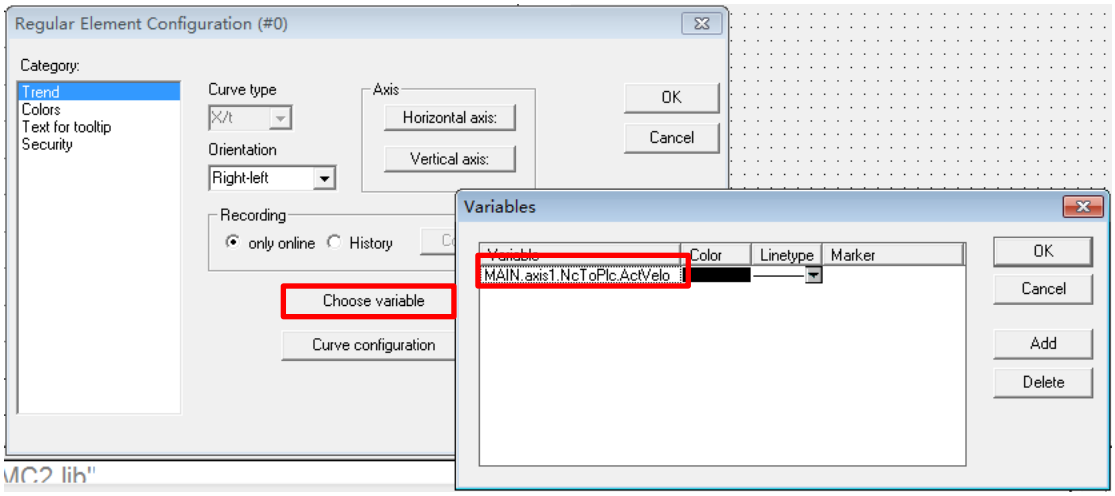
Hmi 里面加入两个按钮（Tap variable），分别用来触发 MC_home 功能块以及触发 sensor 信号。



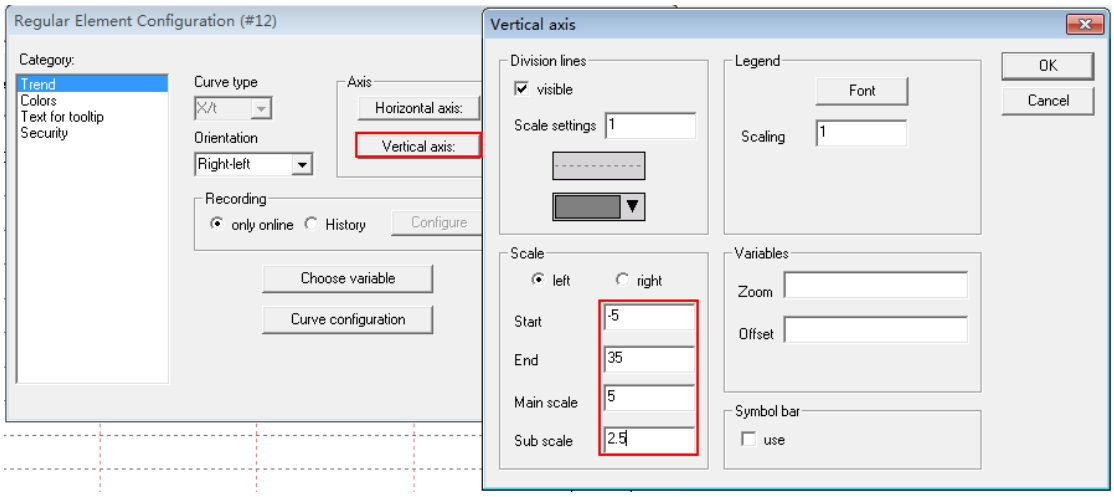
HMI 加入一个 Trend 趋势图控件，可以在上面的控件栏中找到 Trend，这里用趋势图来监视轴的速度变化，先将趋势图控件添加到 HMI 中。



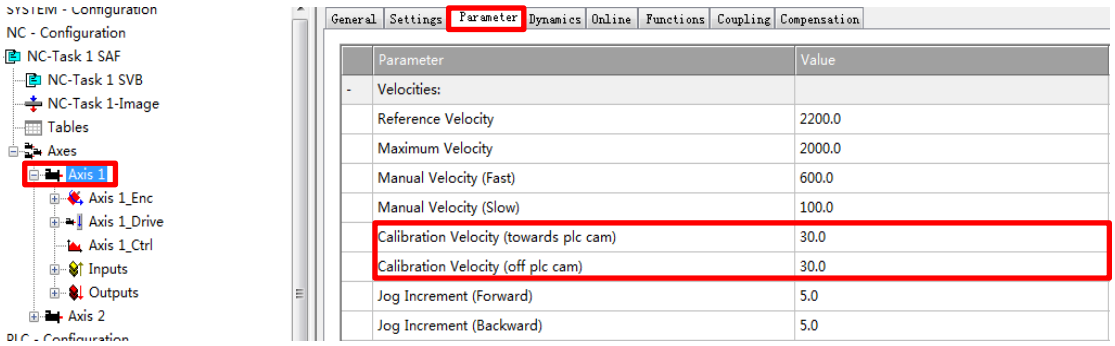
双击趋势图进行配置，首先需要选择趋势图所采样的变量，点击 Choose variable，会弹出一个窗口，按下 F2 之后选择 MAIN.axis1.NcToPlc.ActVelo 做为采样变量，此变量代表轴 1 的当前速度。



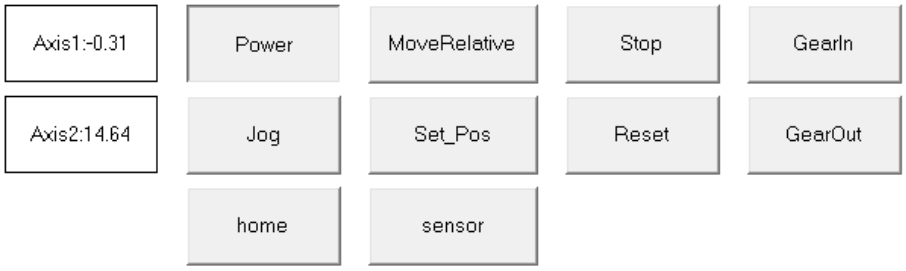
按下 Vertical axis 来设置趋势图 Y 轴的上下限，这里设置下限为-5，上限为 35，这两个值都是根据 Nc 中设置的寻参速度来决定的。

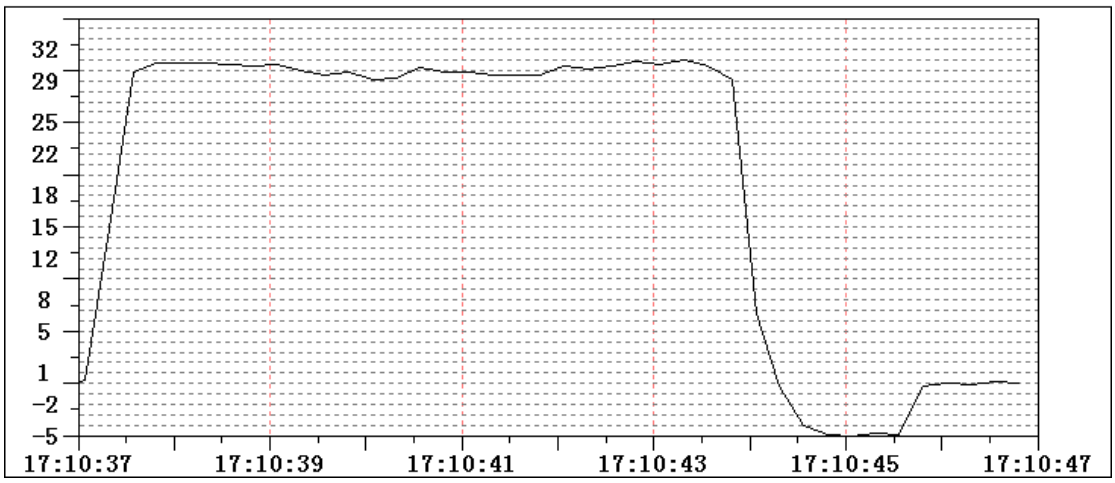


轴的寻参速度可以通过 AXIS-Parameter 中设置，默认为 30，这里有两个速度，一个是找原点的速度（towards plc cam）,另一个是碰到原点后反转的速度（off plc cam），这里可以将 off plc cam 的速度设置为 5，然后激活配置生效。



将程序 Login 之后，首先按下 power 按钮对轴进行使能，然后按下 Home 按钮，此时轴开始找寻原点，速度为 30 左右，当前位置变为-9999999999，然后按下 sensor 按钮不要放开，观察此时的轴会停止且反转，在放开 sensor 按钮的那一刻，将位置定为原点，找原点的流程如下：正向找原点=>碰到原点信号=>原点信号从 0 变为 1=>电机停止并反转=>脱离原点信号=>原点信号从 1 变为 0=>寻参完成且轴当前位置变成 0。





学员提问：寻参有没有其他模式？比如日系运动控制器寻参有很多模式？
讲师解答：只有一种寻参模式。

NC PTP 功能可以参照 U 盘中提供的例子
培训用 U 盘\运动控制培训\NC PTP

四．位置外部设定值发生器

本章目标：

通过本章节的学习，学员将了解：

☑ 位置外部设定值发生器可以实现的功能

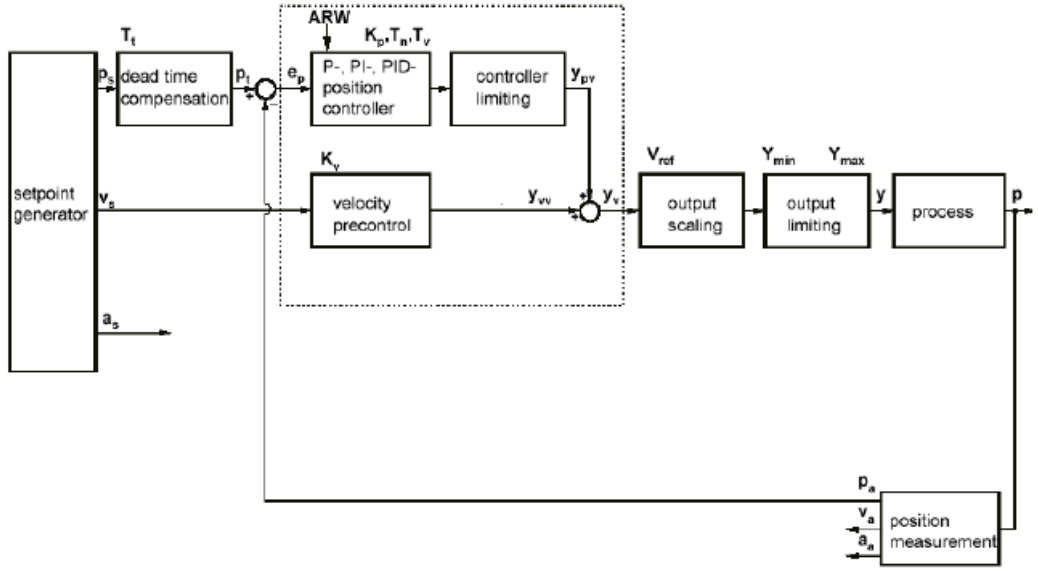
☑ 位置外部设定值所需要调用的功能块

☑ 位置外部设定值使用方法

1. 位置外部设定值发生器的功能

通常TwinCAT NC的设定位置（SetPosition）、设定速度(SetVelocity)、设定加速度(SetAcceleration)是由NC信号发生器（即下图的Setpoint Generator）产生的。每个NC周期（比如2ms）产生一套设定数据Setpoint。如果驱动器工作在位置模式，Setpoint中的位置信号，就会换算后发给驱动器，如果驱动器工作在速度模式，Setpoint中的速度信号，就会换算后发给驱动器。

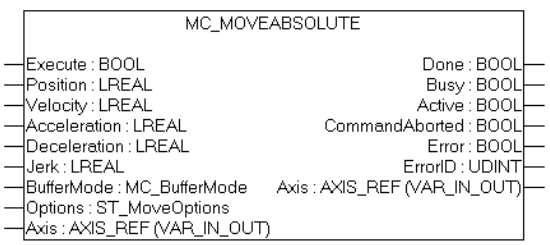
如图所示：



但在特殊情况下，用户需要自己的算法来给定每个NC周期的目标位置和目标速度。此时，可以在PLC程序中使用一个独立的设定值发生器（Setpoint Generator），取代NC位置发生器的功能。这大大增加了TwinCAT 轴的灵活性，可以应用于更广泛的场合。比如，电机转动与实际工件运动为非线性关系时，或者需要多种运动迭加的时候。

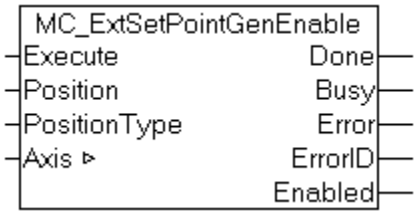
使用外部设定值发生器，需要三个步骤：启用——位置给定——停用，依次由功能块MC_ExtSetPointGenEnable、MC_ExtSetPointGenFeed 、MC_ExtSetPointGenDisable（TcMc2.lib）实现。

学员提问：外部位置给定和MC_MoveAbsolute功能块有什么区别？



讲师解答：MC_MoveAbsolute功能块执行之后，轴会根据指定的速度走到目标位置，期间不需要额外的对轴进行控制，外部位置给定则需要每个扫描周期都给出轴的目标位置，轴根据给出的目标位置来移动，用户可以使用高级语言算出轴的位置，然后通过外部位置给定功能块控制轴移动。

2.常用功能块说明



顾名思义，此功能块用于使能外部位置发生器。Execute上升沿生效。

输入变量：

Execute：BOOL；

Position：LREAL；

PositionType：给定位置类型，有三种类型可选：

POS_ABSOLUTE：绝对位置

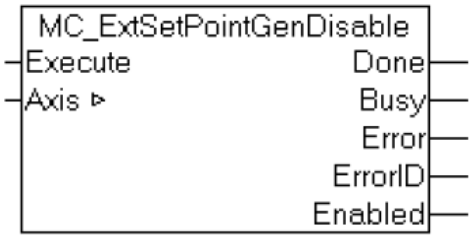
POS_RELATIVE：相对位置

POS_MODULO：模长内定位

输出变量：

Done：成功使能后该标记置位

注意：输入Postion并不是指让NC轴运动到该位置，而是到达该位置后，NC轴标记位InTargetPostion置位,该标记可以通过函数AxisIsAtTargerPosition获得。



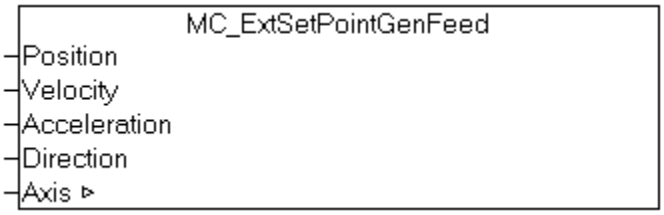
顾名思义，此功能块用于终止外部位置发生器。Execute上升沿生效。

输入变量：

Execute：BOOL；

输出变量：

Done：成功使能后该标记置位



注意，这是一个**Function**，在**PLC**程序中引用时，要在**Function**中去搜索。

此Function用于给定位置发生器的目标位置，仅当发生器使能“MC_ExtSetPointGenEnable”以后，才把输入变量复制到接口变量AxisRefOut结构体中。

输入变量：

Position：LREAL；复制到Axis_Ref.PlcToNc.ExtSetPos；

Velocity：LREAL；复制到Axis_Ref.PlcToNc.ExtSetVelo；

Acceleration：LREAL；复制到Axis_Ref.PlcToNc.ExtSetAcc；

Direction：方向选择；复制到Axis_Ref.PlcToNc.ExtSetDirection；

MC_Positive_Direction：正向

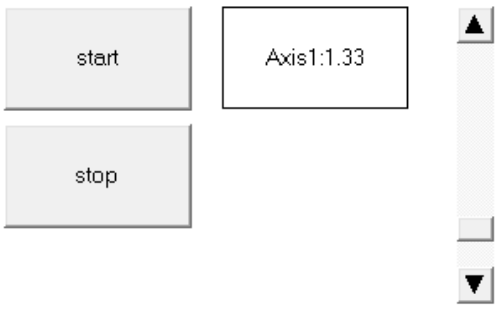
MC_Shortest_Way：最短距离

MC_Negative_Direction：反向

MC_Current_Direction：当前方向

3. 位置外部设定值发生器配置例程

将U盘中提供的例子程序和配置文件打开，将配置激活，程序登录，这里选择Local做为目标设备，程序Login之后点击run，打开人机界面，首先点击start开启外部位置给定功能，然后滑动界面右边的进度块，进度块绑定了MC_ExtSetPointGenFeed中的Position变量，当进度块滑动时，轴的当前位置也随着相应变化，此时通过Postion变量来控制轴的位置，点击Stop按钮关闭外部位置给定功能。




```
0031 MC_ExtSetPointGenFeed(
0032     Position:=SetPosEx,
0033     Velocity:=SetVelEx,
0034     Acceleration:=SetAccEx,
0035     Direction:=SetDirectionEx,
0036     Axis:=axis1);
```

```
SetPosEx = 1.333...
SetVelEx = 100
SetAccEx = 1000
SetDirectio...  = 1
```

外部位置给定功能可以参照 U 盘中提供的例子
培训用 U 盘\运动控制培训\外部位置给定

注意：

1， 启用外部设定值发生器之前，MC_ExtSetPointGenFeed中的设定位置必须与当前位置一致，否则会引起速度跳变，如果带着驱动器和电机，很容易发生事故，或者驱动器报警。

2， 启用外部设定值发生器之前，如果位置环放在TwinCAT NC中完成，则需要将Kv置为0。

3， 无论位置、速度、加速度中只给一个还是三个参数都给定，如果驱动器工作在位置模式，则Position生效，如果驱动器工作在速度模式，则Velocity生效，如果驱动器工作在力矩模式，则Acceleration生效。当然，程序中三个变量最好互相匹配。

4， 要用到外部设定值的程序，其轴变量所在的PLC周期、NC SVB任务周期都应调整为与NC SAF周期相等。否则即使位置给定平滑连续，电机也会抖动。

五．位置补偿

本章目标：

通过本章节的学习，学员将了解：

☑ 位置补偿所对应的功能块

☑ 位置补偿的应用场合

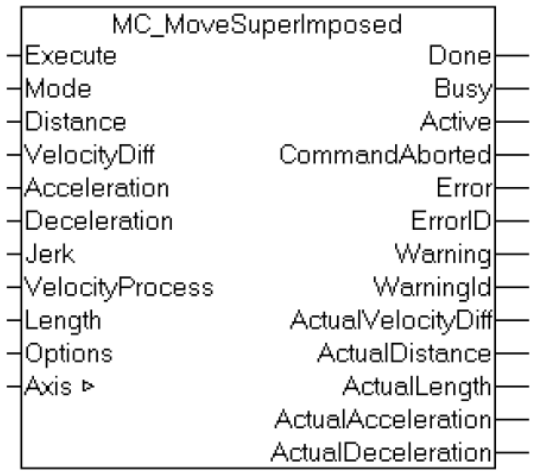
☑ 位置补偿使用方法

1.位置补偿功能块

TwinCAT NC PTP提供一个用于位置补偿的功能块MC_MoveSuperImposedExt。该功能块使运动中的NC轴同时执行一个位置叠加的动作。

如果NC轴是独立运动的轴、多轴联动的Master，或者电子凸轮Cam的Slave，都可以对轴进行补偿。TwinCAT Build20xx以前，电子齿轮Gear的Slave不能做位置补偿。但是最新测试的TwinCAT NC Build2224，做匀速运动的NC轴和齿轮从轴的NC轴都可以进行位置补偿了。

注意：如果 NC 轴的主动作是单向运动，位置补偿的结果可以预期。如果是往复运动，那么补偿动作触发的时机、补偿距离就会导致不同的结果，有时候可能补偿无法完成或者出现静止、反转的情况。这种情况下，使用外部设定值发生器（ExtGenerator），或者多主轴凸轮耦合（MC_CamIn_V2），是两个备选方案。



该功能块由输入变量Execute的上升沿触发。完成后输出变量Done置位。

VelocityProcess，指补偿过程匀速阶段的最大限值。

位置补偿的功能块MC_MoveSuperImposedExt的关键参数是补偿距离Distance、最大速度差VeloctityDiff、补偿区间Length以及补偿模式Mode。

补偿模式Mode用于选择生效的补偿速度差和补偿区间。一共有4种模式：

SUPERPOSITIONMODE_VELOREDUCTION_ADDITIVEMOTION ：

规定的区间Length+Distance内完成Distance的补偿，限定速度变化不超过VelocityDiff。

SUPERPOSITIONMODE_VELOREDUCTION_LIMITEDMOTION：

规定的区间Length内完成Distance的补偿，限定速度变化不超过VelocityDiff。

SUPERPOSITIONMODE_LENGTHREDUCTION_ADDITIVEMOTION,

以规定的最大速度差VelocityDiff完成补偿，补偿区间最短，以Length+Distance为限。

SUPERPOSITIONMODE_LENGTHREDUCTION_LIMITEDMOTION ,

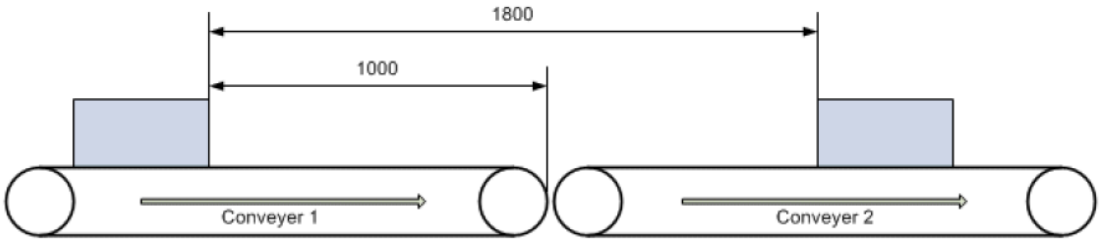
以规定的最大速度差 VelocityDiff 完成补偿，补偿区间最短，以 Length 为限。

2．适用位置补偿的场合

下面详细介绍几种不同的应用场合：

- 产品传送带上的位置补偿。

一条传送带分为若干段，每段由一个伺服轴驱动。传送带用于传送包装箱，包装箱之间必须保持正确的距离。如果不符合设定值，就要增加或者减小，包装箱必须在到达传送带终点之前，比前段传送带走得更慢或者更快，这就是位置补偿。



如图所示，当前测量距离是1800mm，需要缩短至1500mm。传送带1应加速，以缩短距离。距离补偿必须在传送带1到达终点之前完成，以免包装箱被推到速度更慢的传送带2上。

由于此时传送带1必须加速，传动系统要求给定速度差，在本例中假设为500mm/s。实际应用中，该值取决于传送带的最大速度和当前设置速度之差。

功能块MC_MoveSuperImposed的参数设置：

Distance = 1800 mm-1500 mm = 300 mm(补偿距离)

Length = 1000 mm(补偿距离，此处用包装箱到传送带终点的距离)

Mode = SUPERPOSITIONMODE_VELOREDUCTION_LIMITEDMOTION

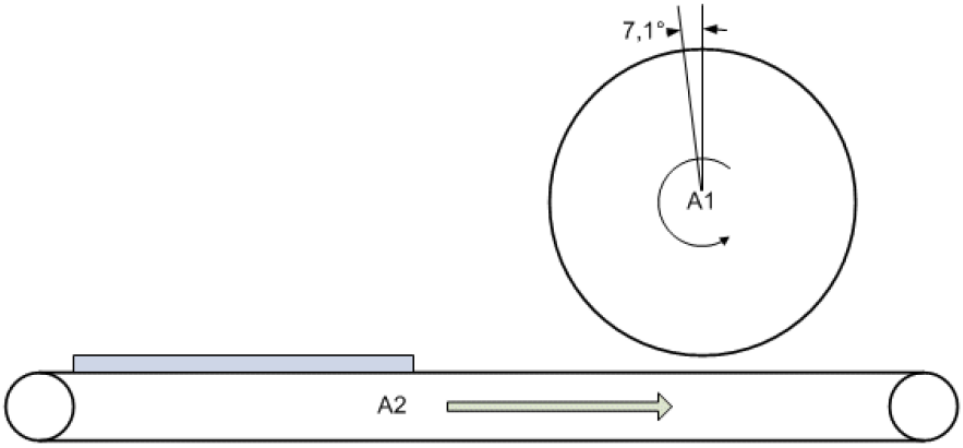
VelocityDiff = 500 mm/s

这种模式下，补偿距离为最大，以保持速度变化量为最小。此时速度差 VelocityDiff 的设定值是传送带 1 完成位置补偿的最大速度变化量。该值不能太小，以至传送带 1 用这个速度差走到终点还不能完成位置补偿。

另一种办法是让传送带 2 减速。此时，补偿位置 Distance 必须为负，而补偿距离 Length 为包装箱的右端到传送带 2 终点的距离。允许的最大速度差 VelocityDiff 相应改为传送带 2 的最大速度与当前速度之差。这样传送带 2 就可以减速，必要时甚至可以减为 0。

- 印刷辊移相。

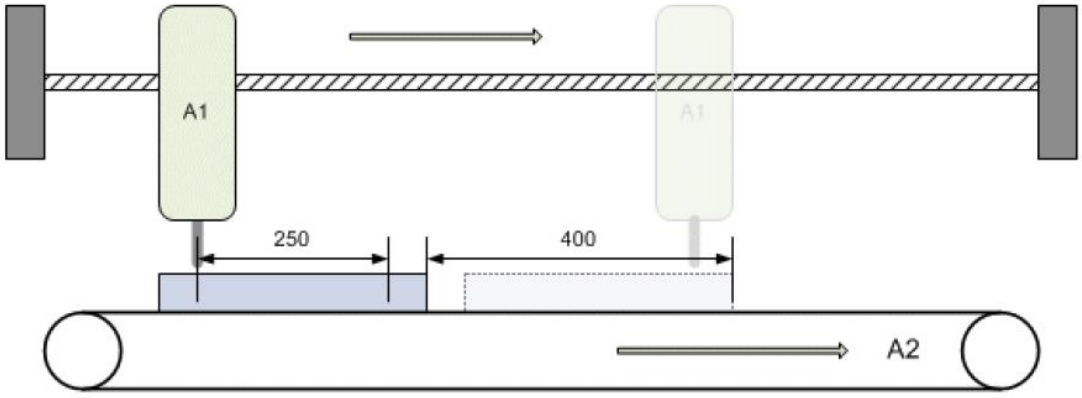
印刷辊轮保持与印刷工件所在的传送带相同的速度均速运动。如果辊轮上的印刷图案位置与工件上的设计印刷位置没有同步，印刷辊轮就必须补偿一个适当的角度（移相）。
如图所示：



移相可以有两种方式。
快速移相：在最短时间内修正相位角度，此时印刷辊轮必然发生速度冲击。
慢速移相：在尽可能长的距离内修正相位角度以减小速度冲击。比如，辊子转动完整一圈。
功能块MC_MoveSuperImposedExt的参数设置：
1.快速移相
Distance = 7.1°
Length = 360°(最大补偿距离)
Mode =SUPERPOSITIONMODE_LENGTHREDUCTION_LIMITEDMOTION
VelocityDiff = 30°/s(速度差)
此模式下，补偿距离尽可能短。此时补偿距离Length的设定值是辊轮完成移相的最大距离。该值不能太小，以至用最大速度也不能在这么短的距离内完成位置补偿。
也可选择模式SUPERPOSITIONMODE_LENGTHREDUCTION_ADDITIVEMOTION。此时，补偿距离为367.1°。由于补偿距离都是尽可能短，实际上对于这种应用，两种模式结果相同。
2. 慢速移相
Distance = 7.1°
Length = 360°(correction distance)
Mode =SUPERPOSITIONMODE_VELOREDUCTION_LIMITEDMOTION
VelocityDiff = 30°/s(速度差)
这种模式下，补偿距离为最大，以保持速度变化量为最小。此时速度差VelocityDiff的设定值是辊轮完成移相的最大速度变化量。该值不能太小，以至辊轮用这个速度差走完一圈还不能完成位置补偿。

- 钻削设备

钻头要在运动的工件上钻两个孔。第一个孔的同步是通过飞锯功能(MC_GearInPos)实现的，在此不再详述。完成第一个孔后，钻头必须相对于运动工件移动一定的距离。
如图所示：



图中设备完成第一个孔后，钻头必须相对于工件移动250mm，即两孔之间的距离。而在这段时间内，工件本身移动的距离是400mm。从这个位置开始，钻头再次与工件同步，然后钻第二个孔。
同样，这里也可以有两种模式可供选择，区别在于钻头的速度变化值。
功能块MC_MoveSuperImposed的参数设置：
1.快速补偿
Distance = 250 mm
Length = 400 mm
Mode = SUPERPOSITIONMODE_LENGTHREDUCTION_ADDITIVEMOTION
VelocityDiff = 500 mm/s(钻头移动速度最大变化量)
在此模式下，补偿距离尽可能短。此时补偿距离 Length 的设定值是钻头完成补偿的最大距离。该值不能太小，以至用最大速度也不能在这么短的距离内完成补偿。由于补偿距离是工件走过的距离加上相对位移，所以钻头实际上要走一个更长的距离。
2.慢速补偿
Distance = 250 mm
Length = 400 mm
Mode = SUPERPOSITIONMODE_VELOREDUCTION_ADDITIVEMOTION
VelocityDiff = 500 mm/s(钻头移动速度最大变化量)
这种模式下，补偿距离为最大，以保持速度变化量为最小。此时速度差 VelocityDiff 的设定值是钻头完成补偿的最大速度变化量。该值不能太小，以至钻头用这个速度差走完全程还不能完成位置补偿。在此过程中，工件走过距离 Length 为 400mm，钻头走过的距离就是 Length + Distance ，即 650mm。

3.位置补偿功能例程

将U盘中提供的例子程序和配置文件打开，将配置激活，程序登录，这里选择Local做为目标设备，程序Login之后点击run，打开人机界面，首先点击Move_Absolute按钮让轴1和轴2开始移动，然后点击SuperImposed按钮进行位置补偿，补偿模式设置为2，SUPERPOSITIONMODE_VELOREDUCTION_LIMITEDMOTION=>规定的区间Length内完成Distance的补偿，限定速度变化不超过VelocityDiff。根据功能块中设置的参数代表在2000的位置内完成500位置的补偿，两轴的速度差不超过500，当两个轴停止之后可以看到轴2走过的距离为5500，轴1走过的距离为5000，完成了500位置的补偿。


```
0043 superimposed(
0044      Execute:=superimposed_do ,
0045      Mode:= 2,
0046      Distance:=500 ,
0047      VelocityDiff:= 500,
0048      Acceleration:= ,
0049      Deceleration:= ,
0050      Jerk:= ,
0051      VelocityProcess:=500 ,
0052      Length:= 2000,
0053      Options:= ,
0054      Axis:= axis2, );
```

```
TYPE E_SuperpositionMode :
(
    SUPERPOSITIONMODE_VELOREDUCTION_ADDITIVEMOTION := 1,
    SUPERPOSITIONMODE_VELOREDUCTION_LIMITEDMOTION,
    SUPERPOSITIONMODE_LENGTHREDUCTION_ADDITIVEMOTION,
    SUPERPOSITIONMODE_LENGTHREDUCTION_LIMITEDMOTION,
    SUPERPOSITIONMODE_ACCREDUCTION_ADDITIVEMOTION,    (from TwinCAT 2.11)
    SUPERPOSITIONMODE_ACCREDUCTION_LIMITEDMOTION      (from TwinCAT 2.11)
);
END_TYPE
```

学员提问：VelocityProcess这个变量有什么用，怎么设置？

讲师解答：VelocityProcess指补偿过程匀速阶段的最大限值，设置过大的话会导致轴加减速过程过短，指定的补偿位置完成不了。

外部位置给定功能可以参照 U 盘中提供的例子

培训用 U 盘\运动控制培训\位置补偿

六. 从 PLC 程序修改 NC 轴的参数设置

本章目标：

通过本章节的学习，学员将了解：

☑ NC参数在Information System中的查询方法

☑ NC参数可以通过哪几种方式在程序中进行读写

☑ 查询NC参数Index-Group与Index-Offset的方法

1.读写 NC 轴参数的功能块

如果需要PLC程序动态地修改NC轴的参数，而不是驱动器参数，有两种方法：用专门的MC功能块，或者使用ADS通讯。

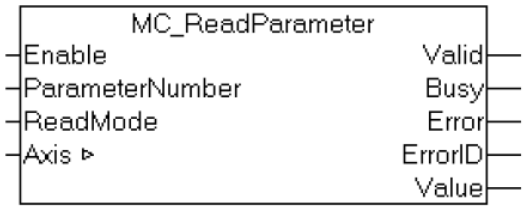
- 用MC功能块修改NC轴的参数

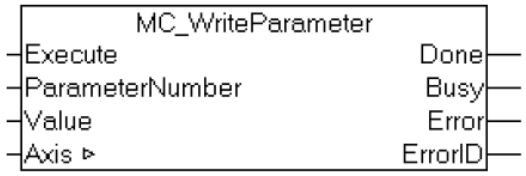
MC_ReadParameter,

MC_ReadBoolParameter,

MC_WriteParameter ,

MC_WriteBoolParameter ,





该功能块在输入变量Enable的上升沿，把输入变量Value的值写到Axis轴的参数号为ParameterNumber的NC变量中。完成后输出变量Done置位。

输入变量：

Enable：BOOL

ParameterNumber：UDINT

Value：LREAL；

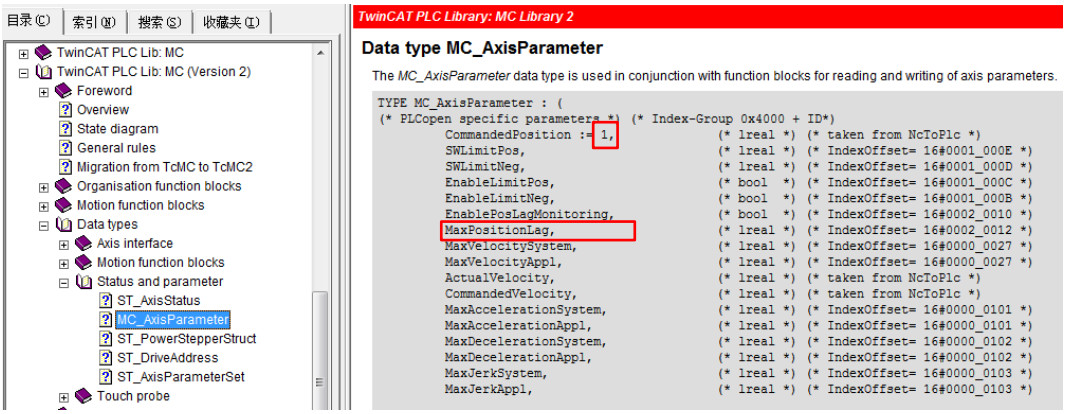
输出变量：

Done：BOOL；

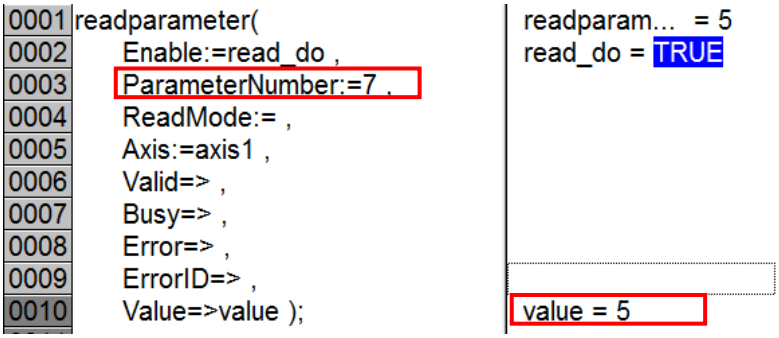
Error：BOOL；

ErrorID：UDINT；

如果要查询 MC_AxisPara 中的参数号说明，请打开帮助文件并定位到以下位置：



例如：最大跟随误差MaxPositonLag的参数号为：7，如果要从PLC程序修改此值，ParameterNumber就应填7。

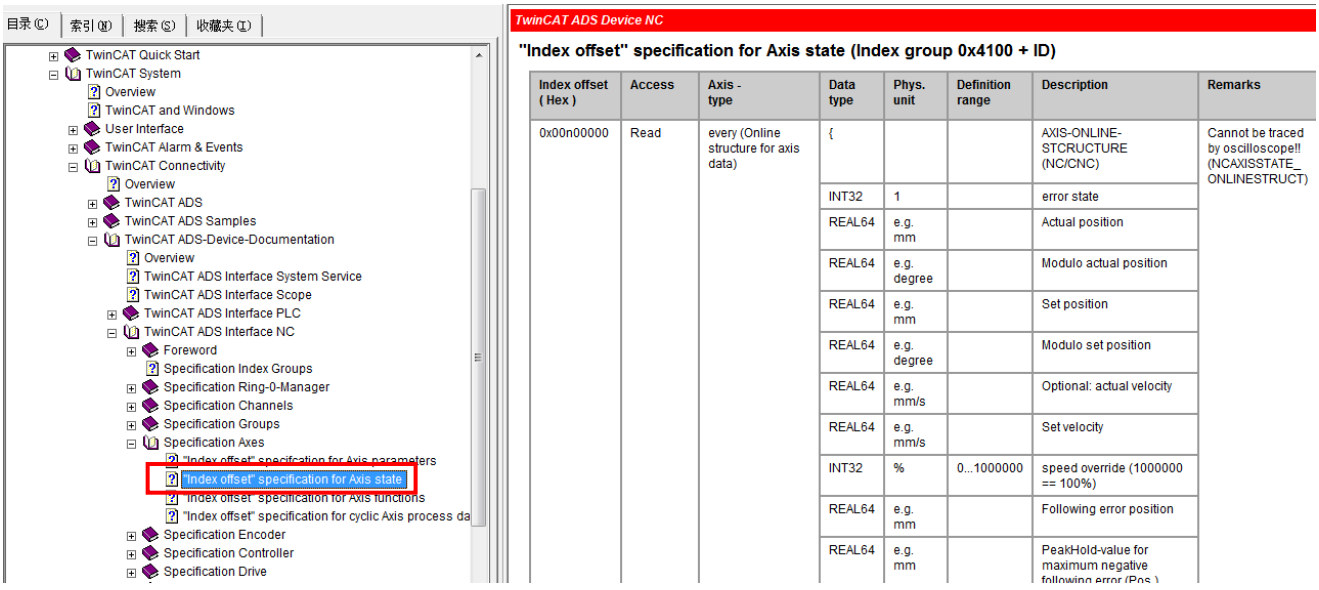


- 用ADSWrite or ADSRead，读写NC轴的参数

用ADS功能块读写NC轴参数时，NetID可以为空白（""），表示与PLC程序同一台控制器。Port填"501"，指TwinCAT NC端口。

Index和Offset，则需要依据NC轴的ID号和要修改的参数而定。

如果要查询 NC 轴参数的 Index 和 Offset 说明，请打开 Beckhoff Information system 帮助文件并定位到下图位置：



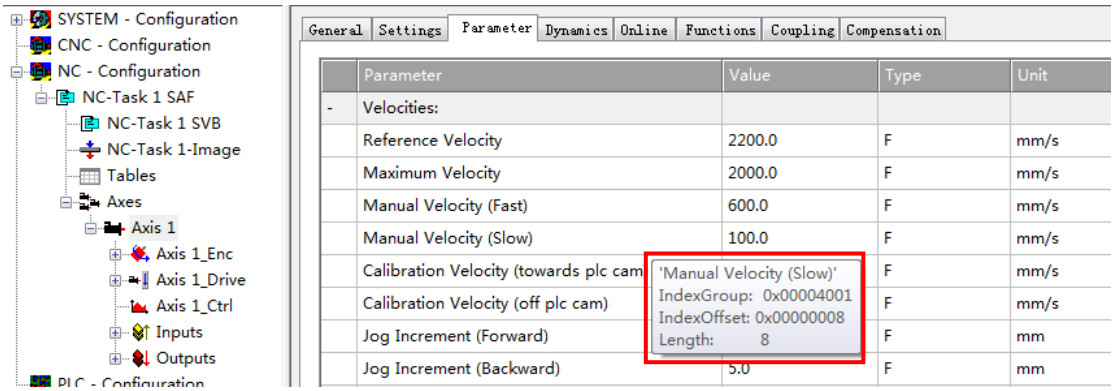
Index offset (Hex)	Access	Axis - type	Data type	Phys. unit	Definition range	Description	Remarks
0x00n00000	Read	every (Online structure for axis data)	{			AXIS-ONLINE-STCRUCTURE (NC/CNC)	Cannot be traced by oscilloscope!! (NCAXISSTATE_ONLINESTRUCT)
			INT32	1		error state	
			REAL64	e.g. mm		Actual position	
			REAL64	e.g. degree		Modulo actual position	
			REAL64	e.g. mm		Set position	
			REAL64	e.g. degree		Modulo set position	
			REAL64	e.g. mm/s		Optional: actual velocity	
			REAL64	e.g. mm/s		Set velocity	
			INT32	%	0...1000000	speed override (1000000 == 100%)	
			REAL64	e.g. mm		Following error position	
			REAL64	e.g. mm		PeakHold-value for maximum negative following error (Pos.)	

例如：

Controller Kv-Factor ， Index Group：16#6000+ID；Offset：16#0102

AxisMaxPosLagValue， Index Group：16#6000+ID；Offset：16#0012

也可以在 NC 的 Parameter 中找到参数的 Index Group 和Index Offset



首先加载 tcsystem.lib 库文件，调用 ADSREAD 功能块即可对 NC 的参数进行读取，Manual Velocity 的值通过 PLC 程序的功能块读取出来，值为 100，同样也可以通过 ADSWrite 对 NC 参数进行修改，IDXGRP 和 IDXOFFS 可以在 Information system 或者 system manager 中查询到。

```
0001 ADSREAD(
0002     NETID:= ,
0003     PORT:=501 ,
0004     IDXGRP:=16#4001 ,
0005     IDXOFFS:=16#8 ,
0006     LEN:= 8,
0007     DESTADDR:=ADR(manual_velo_slow) ,
0008     READ:=read_do ,
0009     TMOUT:= ,
0010     BUSY=> ,
0011     ERR=> ,
0012     ERRID=> );
```

```
manual_vel... = 100
read_do = TRUE
```

学员提问：如何读写AX5000的参数？
讲师解答：可以通过FB_SoeWrite和FB_SoeRead功能块读取AX5000驱动器的参数。

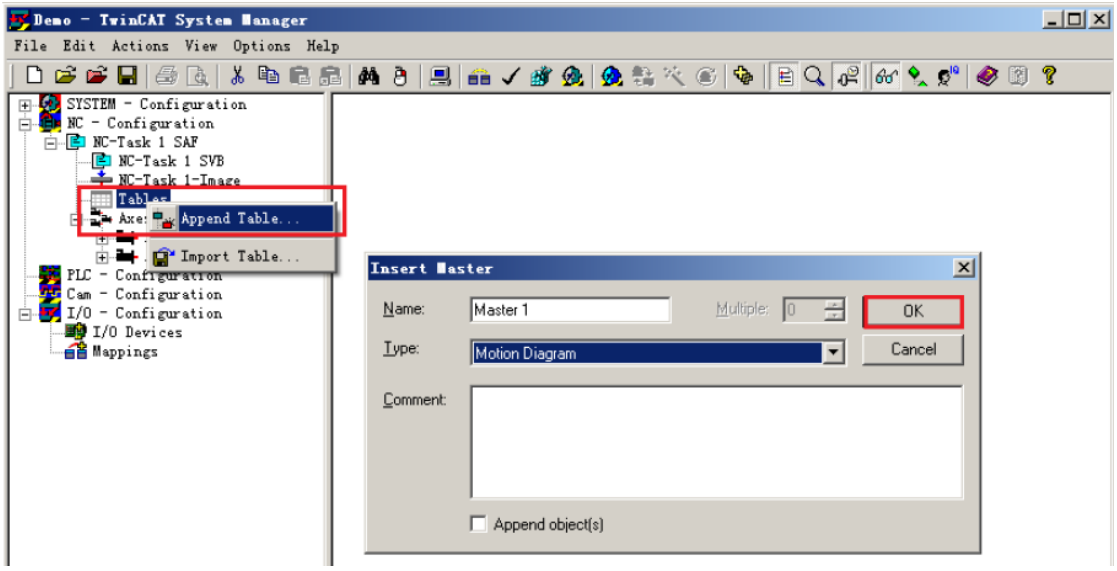
七. 电子凸轮表

本章目标：
通过本章节的学习，学员将了解：
☑ 电子凸轮表能实现的功能
☑ 使用电子凸轮表所需要安装的Supplement
☑ 如何通过System Manager软件生成电子凸轮表
☑ 如何在System Manager软件中调试凸轮表
☑ 通过功能块实现电子凸轮的耦合，解耦，读写关键点等功能
☑ 如何在程序中直接生成电子凸轮表

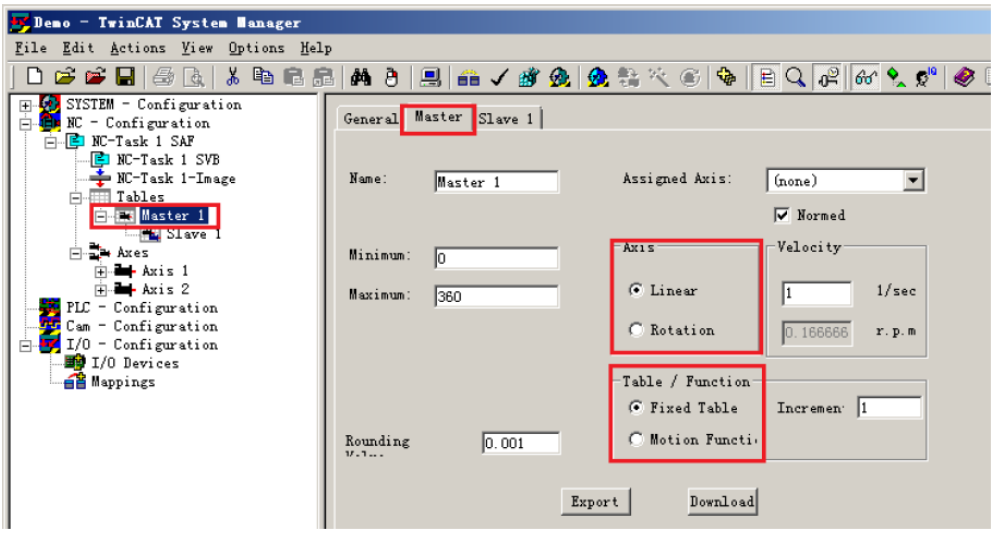
电子齿轮是主轴与从轴的速度保持比例关系，而电子凸轮则是主轴与从轴的位置保持对应关系。这个对应关系就是通过凸轮表（Cam Table）来表示的。TwinCAT System Manager 中提供的凸轮绘制界面。

1. System Manager中添加凸轮表：

准备工作：安装Cam Design Tool，需要购买授权（如果没有安装这个supplement，配置的凸轮表无法下载激活）安装路径：
http://www.beckhoff.com/forms/twincat3/warenkorb.aspx?lg=en&title=TS1510-CAM-Design-Tool&version=1.0.2
右键单击“Tables”，选择“Append Table”，就添加了一个主轴 Master 1，



在 Master 设置页面：



学员提问：Linear 与 Rotation 的区别？

讲师解答：图中 Axis 选项“Linear”和“Rotation”，对应主轴运动特征旋转型（Rotation）和直线型（Linear）。如果是 Rotation，当主轴位置超出凸轮表定义的范围后，从轴位置还会按照凸轮表的周期重复运动。如果是 Linear，从轴在表中找不到对应的位置就不再运动。

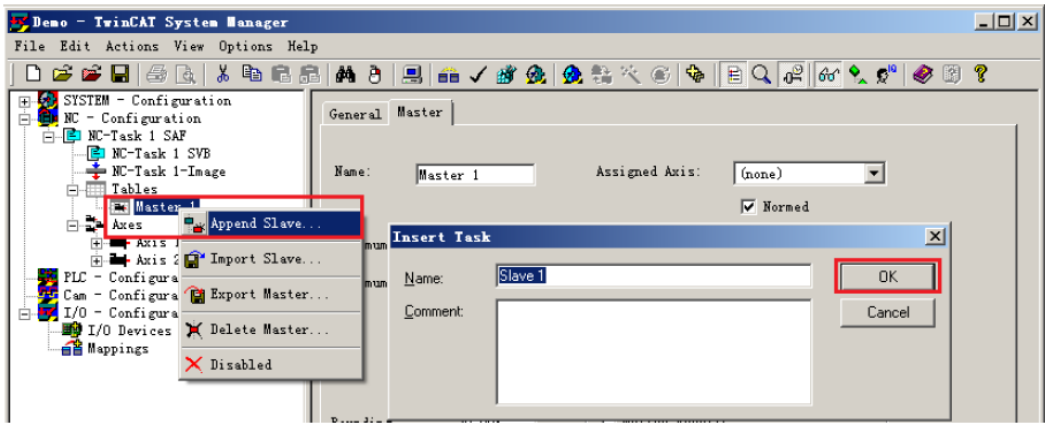
学员提问：Fixed Table 与 Motion Function 的区别？

讲师解答：传统上，把主轴和从轴之间非线性的电气耦合关系称为凸轮表。TwinCAT提供多种凸轮表：

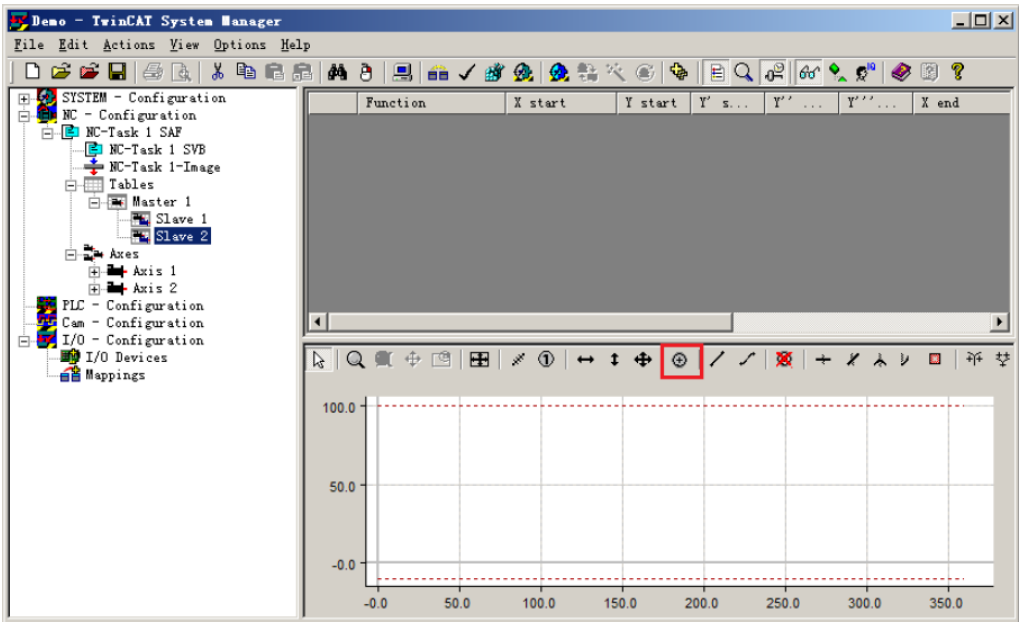
第一种是典型的位置表（FixedTable），位置表中用大量的点来描述一段凸轮曲线，每个主轴位置对应一个从轴位置，相邻点之间使用直线插补方式。这种方式的凸轮表过去用得很多，它的缺点是运行过程中很难修改。

为了弥补这个不足，TwinCAT 提供第二种凸轮表，即 MotionFuntion 型（MF）的凸轮表，它用另一种方式描述凸轮曲线。MF 型凸轮表通常只包含少量的关键点，然后用数据公式，比如 polynomial（多项式），来描述相邻两点之间的曲线。运行时根据数学公式实时计算从轴的位置。由于修改关键点就可以修改曲线，所以 MotionFuntion 型的凸轮表比 FixedTable 更容易在线修改。

右键 Master1 点击 Append Slave，添加凸轮表，这里添加两次就出现了两张凸轮表，分别对应 Table id 1 和 Table id 2。



2. System Manager中编辑凸轮表



可以使用上图中，曲线上方的一排按钮来增减或者移动关键点。这些按钮的功能如下：

 ：增加关键点

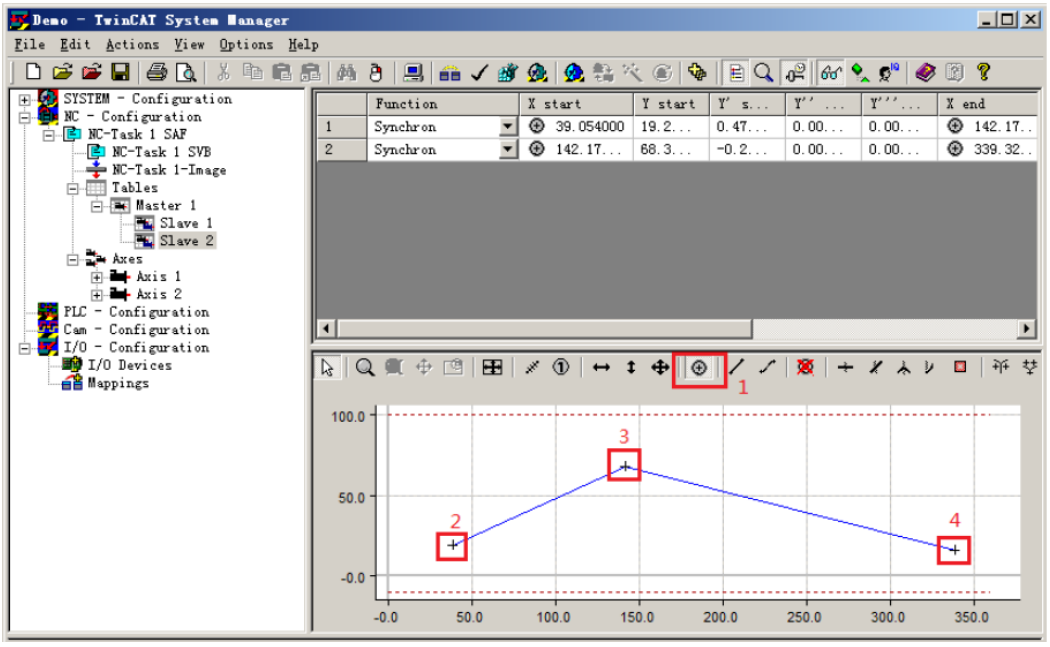
 ：删除关键点

 ：关键点移动

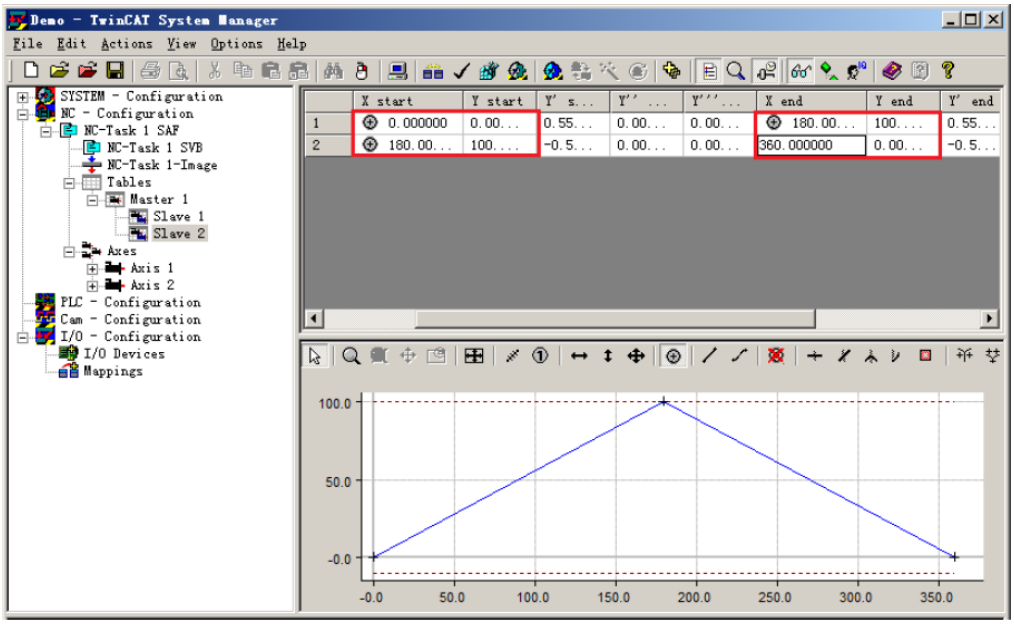
 ：以直线或者自动平滑曲线连接关键点

 ：视图缩放，取消缩放，视图平移，右上角小窗口显示

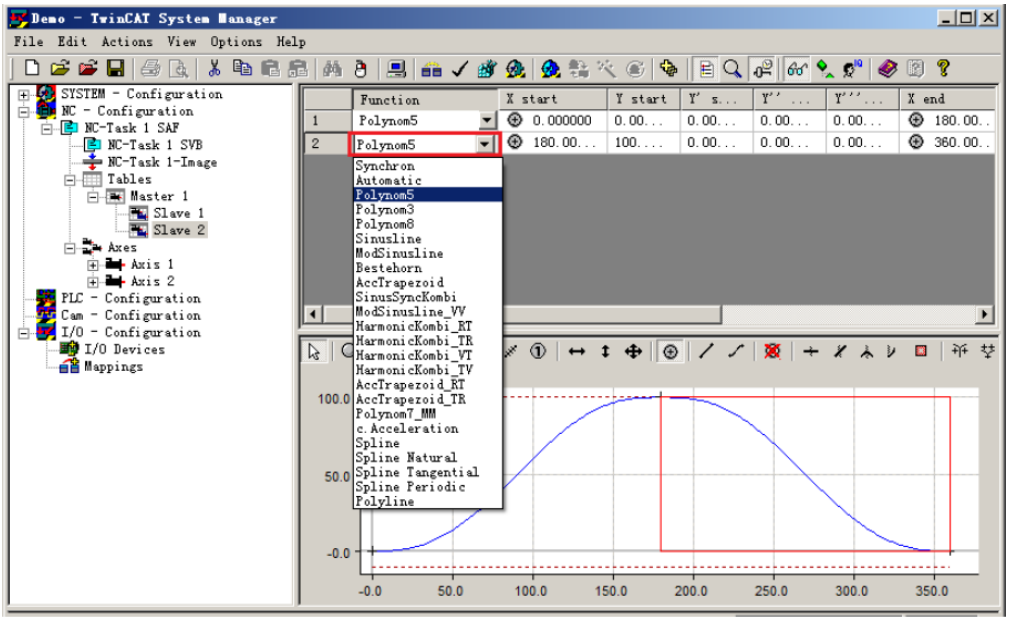
红线框内的按钮表示增加关键点。双击该按钮后，在下面的白色区域不同地方从左到右点击 3 次，就在图上增加了两个线段。



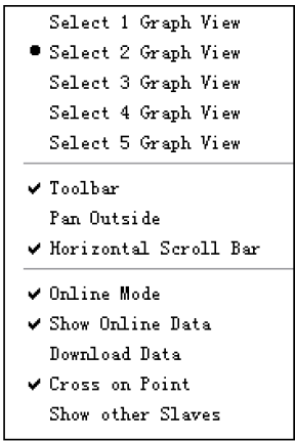
修改表格中各关键点的 X、Y 轴坐标。



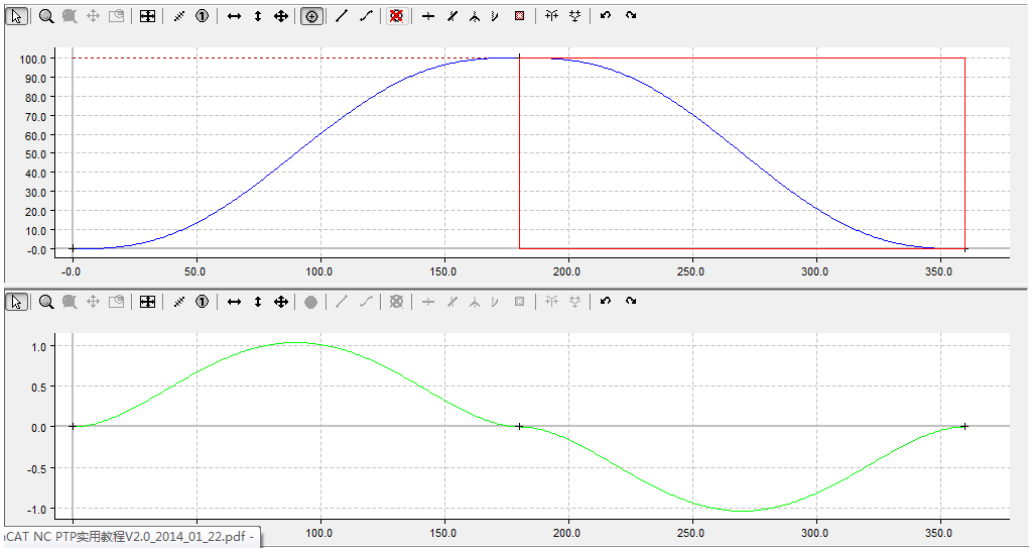
选择线段连接方式，这里可以选择不同的数据公式，具体哪种公式适用要经过现场测试。



在曲线显示区域单击右键，显示快捷菜单如下：



默认勾选“Select 1 Graph View”，显示一次曲线，即位置曲线，如果如上图所示，勾选“Select 2 Graph View”，则不仅显示一次，还显示二次曲线，即速度曲线。



同理，如果勾选“Select 3 GraphView”，还可显示即加速度曲线，勾选“Select 4 Graph View”，还可显示即加加速度（Jerk）曲线。此至，一个凸轮表就建好了。

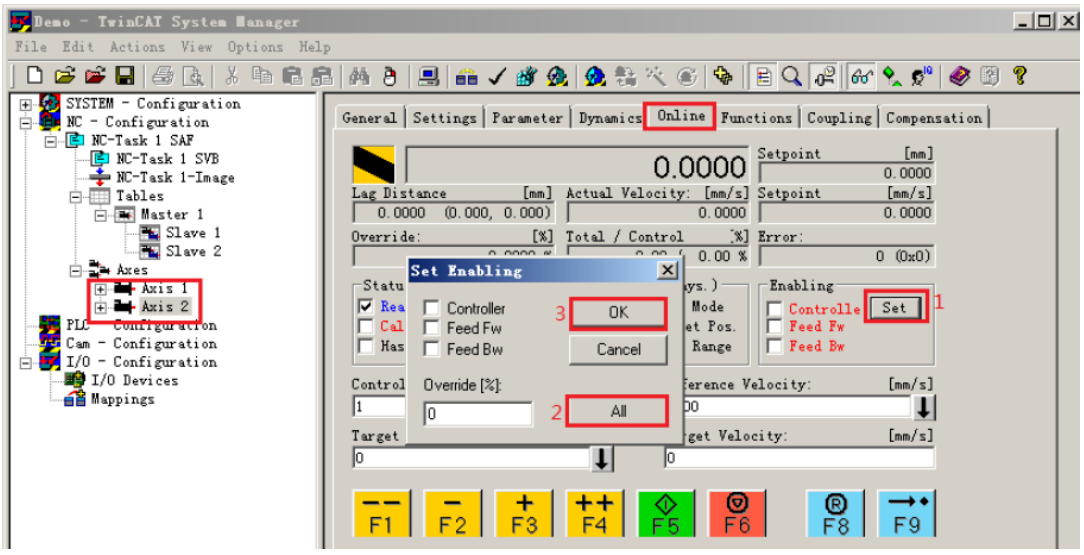
学员提问：怎么判断我画的凸轮表是否合适？
讲师解答：观察凸轮表的速度曲线，速度不能有阶跃，可以先用虚轴调试，用scope view软件监视轴的速度曲线和位置曲线。

3. system manager中激活配置

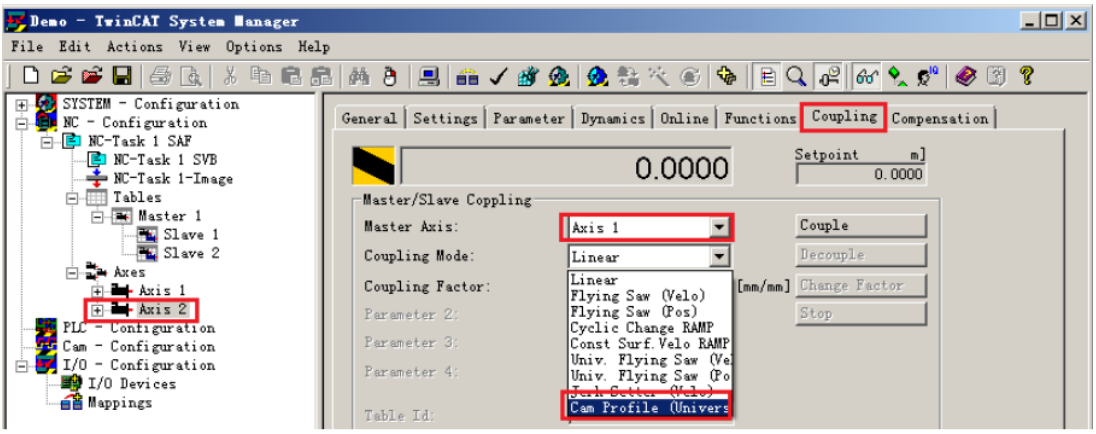
点击Activate configuration进行配置下载，并切换TwinCAT至运行模式。

4. system manager中调试凸轮表：

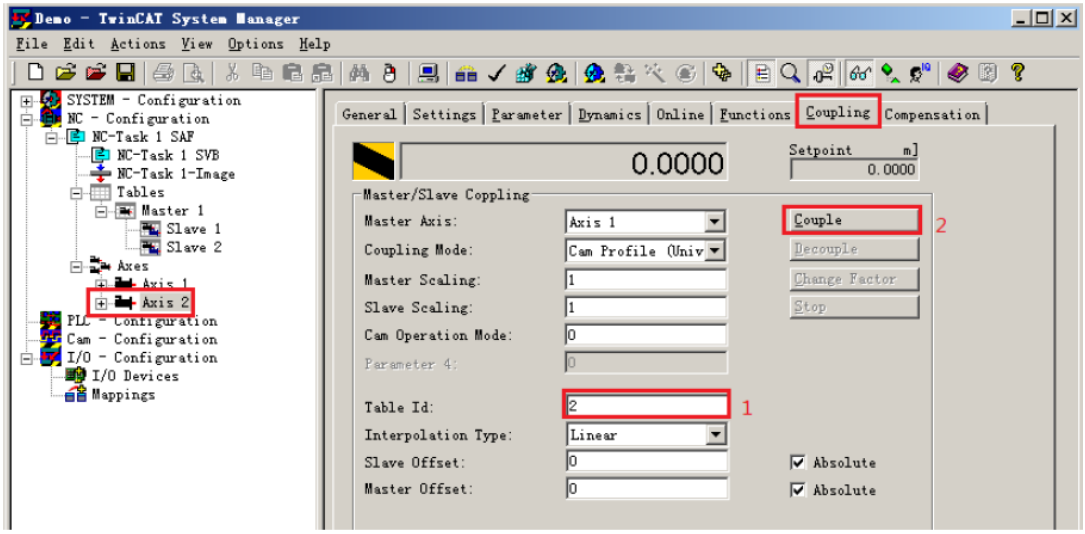
首先需要将主从轴进行使能



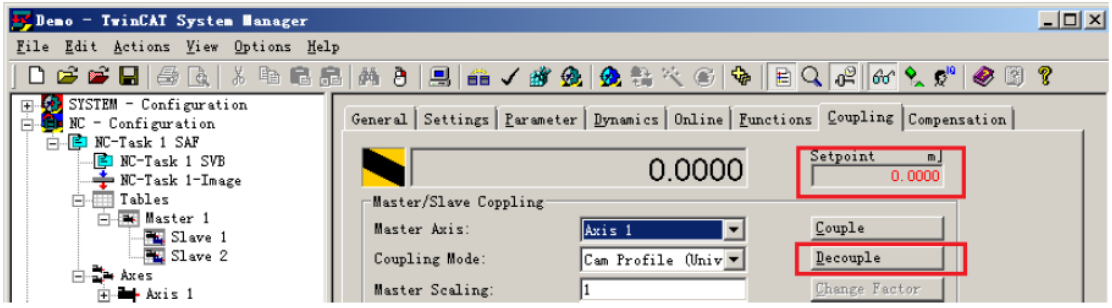
Coupling 中选择耦合关系为凸轮，设置主轴为 Axis1。



选择凸轮表，tableID 设置为 2，表示选择 Tables-Master1-Slave2 中的编辑的凸轮表，然后点击 Couple 进行耦合。



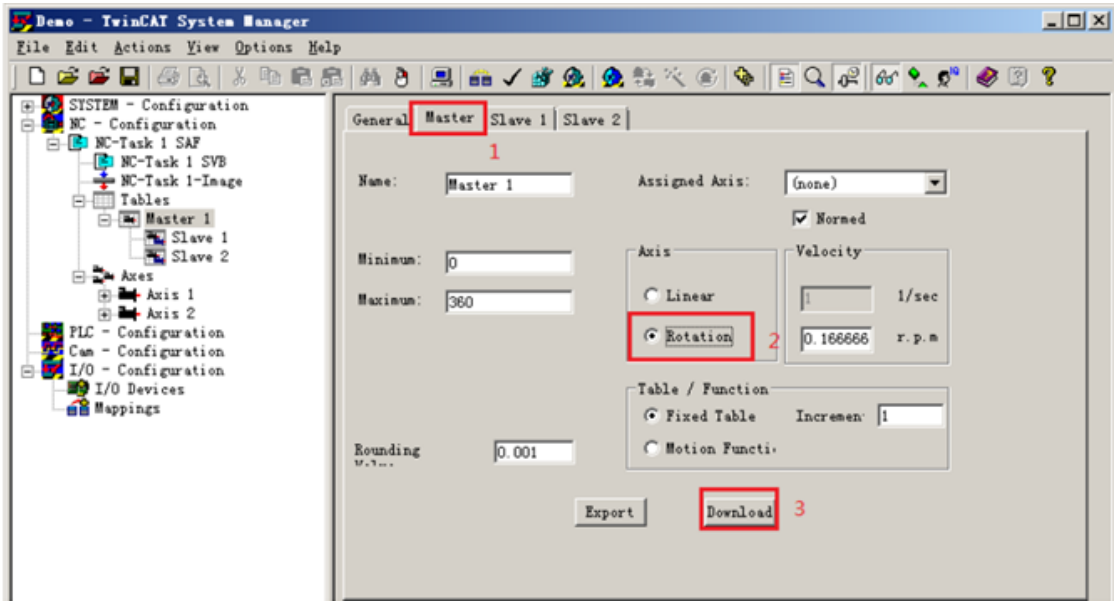
耦合之后可以看到 Axis2 变成了从轴，Setpoint 变为红色，代表此时 Axis2 不能单独进行控制，Axis2 只能跟随 Axis1 进行凸轮关系的运动。



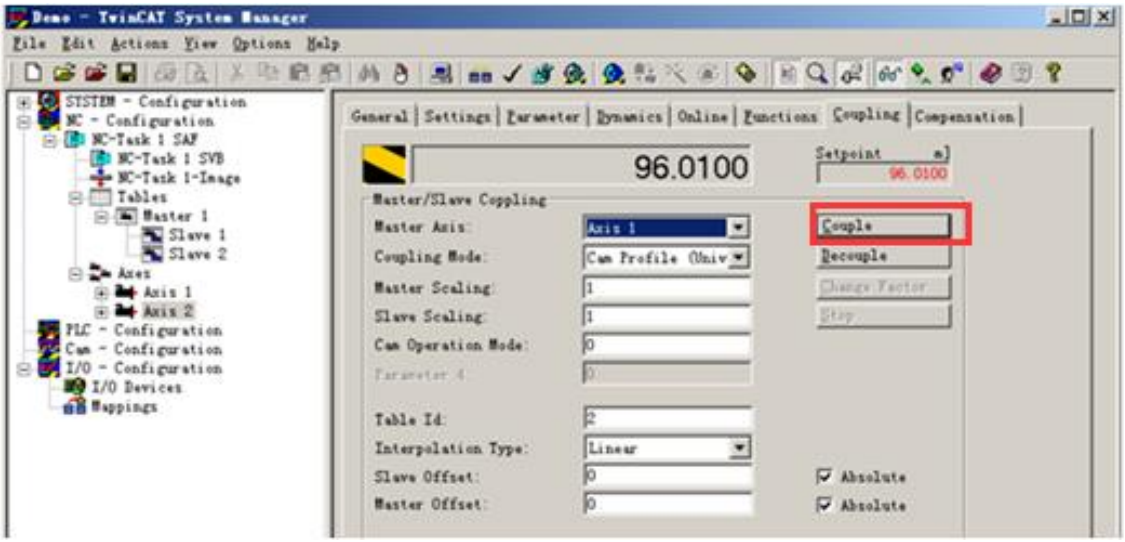
当Axis1作单向均速运动时，Axis2运动了一段时间，即停止不动了。
这是由于Master属性为Linear而不是Rotation。为了修改该属性，应先Decouple。

5. system manager中修改凸轮表：

修改Master属性，将linear修改为Rotation，然后点击Download。



回到 Axis 的 Coupling 页面。重新 Couple：



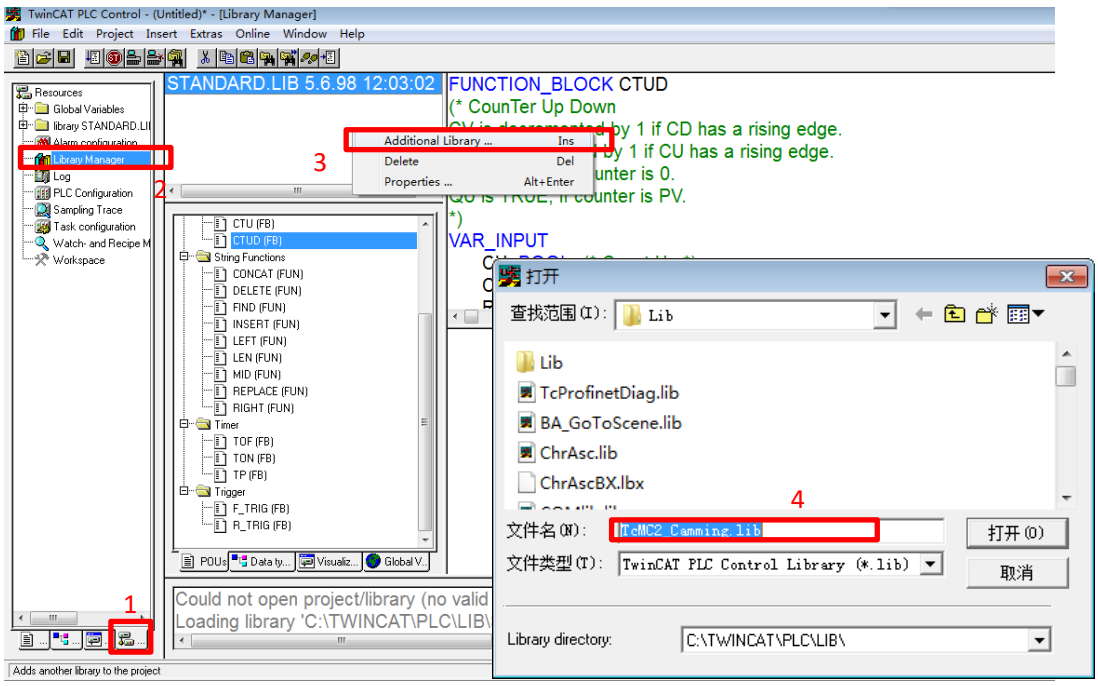
当 Axis1 作单向均速运动时，从轴也跟随运动，且周而复始，不再停止。

6. Plc Control编程实现电子凸轮功能

从PLC程序控制凸轮运动，需要调用TcMc2_Caming.lib，此库文件包含以下功能：（注：此库文件需要安装supplement才可获得）

序号	步骤	功能块	说明
1	准备凸轮点数据	Fixed Table（描点型凸轮）： 主从轴坐标，LReal 型。 Motion Function(关键点式凸轮): *MC_MotionFunctionPoint 型	如果在 tsm 文件中用 Cam Design Tool 编辑，则不用在 PLC 程序中创建和装载凸轮表 Cam Design Tool 是基于 TwinCAT 软件的一个 Supplement，需要购买授权并安装。其使用方法见第 3 章。
2	准备凸轮数据结构体	MC_CAM_REF 赋值	
3	把 MC_CAM_REF 装载到指定 TableID 的凸轮	MC_CamTableSelect	
4	凸轮耦合	MC_CamIn（单表耦合） MC_CamIn_V2(多表耦合)	
5	主轴运动		
6	修改关键点	MC_ReadMotionFunctionPoint MC_WriteMotionFunctionPoint	（可选）
7	修改周期、相位、偏移和幅值	MC_CamScaling MC_CamScaling_V2	（可选）
6	凸轮解耦	MC_CamOut	
7	从轴停止	MC_Stop	

新建一个 PLC Control 的项目，选择 pc or cx(x86)，编程语言选择 ST 语言，之后点击左边对象管理器中的 Resource 选项卡，双击 Library manager，右键 Standard.lib 下空白的地方，选择 Additional Library，添加 TCMC2_CAMMING.LIB 的库文件。



在 MAIN 的变量窗口中定义如变量以及功能块，axis_ref,mc_power,mc_jog 在前面的内容中已经介绍，这里不再介绍了，MC_camin_v2 与 mc_camout 是凸轮的耦合以及解耦的功能块。

```
0001 PROGRAM MAIN
0002 VAR
0003     axis1,axis2:axis_ref;
0004     power1,power2:MC_Power;
0005     jog1:MC_Jog;
0006     power_do:BOOL;
0007     jog_forward:BOOL;
0008     camin:mc_camin_v2;
0009     camout:mc_camout;
0010     camin_do:BOOL;
0011     camout_do:BOOL;
0012 END_VAR
```

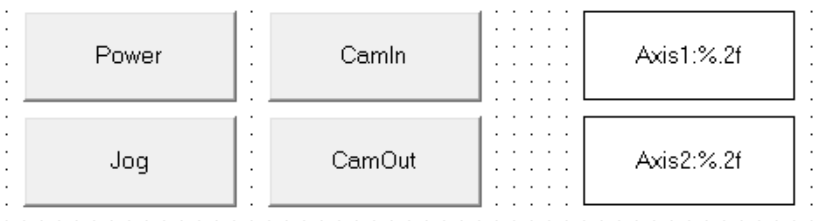
在程序编写窗口中调用 mc_camin_v2 的功能块，并将功能块的参数填写完整

```
0041 camin(
0042     Execute:=camin_do,
0043     ActivationMode:= ,
0044     ActivationPosition:= ,
0045     CamTableID:=1 ,
0046     Scaling:= ,
0047     Options:= ,
0048     Master:=axis1 ,
0049     Slave:=axis2 ,
0050     InSync=> ,
0051     Busy=> ,
0052     Active=> ,
0053     CommandAborted=> ,
0054     Error=> ,
0055     ErrorID=> );
```

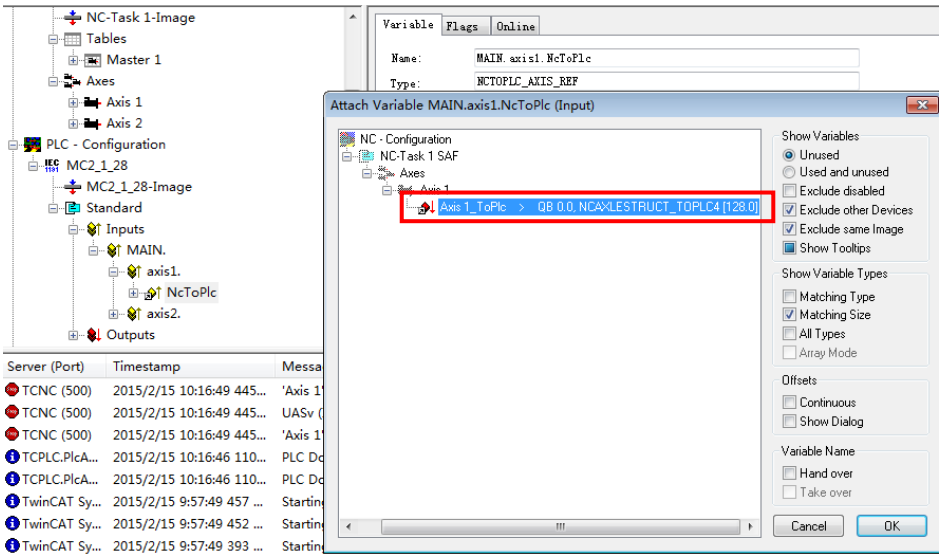
在程序编写窗口中调用 mc_camout 的功能块，并将功能块的参数填写完整。

```
0056 camout(
0057     Execute:=camout_do,
0058     Options:= ,
0059     Slave:=axis2 ,
0060     Done=> ,
0061     Busy=> ,
0062     Error=> ,
0063     ErrorID=> );
```

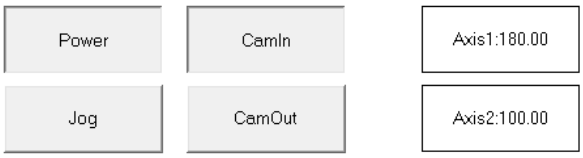
新建一个 HMI，里面新建两个矩形框来显示轴的位置，另外添加 4 个按钮分别对轴进行使能，点动，耦合以及解耦，使能和点动按钮选择 toggle variable（交替按钮），耦合和解耦选择 tap variable（瞬时按钮）。



程序编写完成之后先保存再编译，然后在 system manager 软件中调用 TPY 文件，并且做好变量链接，之后激活配置。



回到 PLC Control 软件中，将程序 login（先确认 choose runtime 选择的目标是否正确），之后 run 程序，先按下 power 按钮对轴进行使能，按下 CamIn 对主从轴进行耦合，然后按下 Jog 即可看到 Axis1 和 Axis2 以凸轮表中绘制的位置关系来移动。



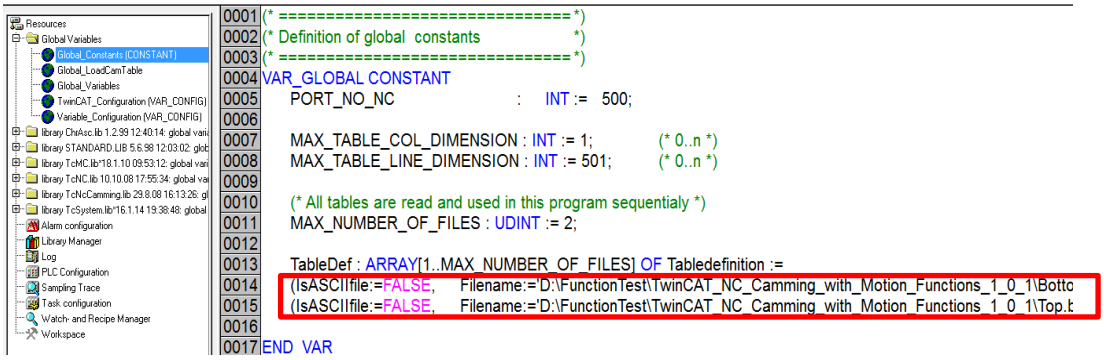
学员提问：主从轴处于耦合状态并移动的时候是否可以解耦？
讲师解答：不建议，运动的时候解耦，从轴会随着解耦时刻的速度一直移动，需要单独的停止命令来让从轴停止。

电子凸轮功能可以参照 U 盘中提供的例子
培训用 U 盘\运动控制培训\凸轮飞锯

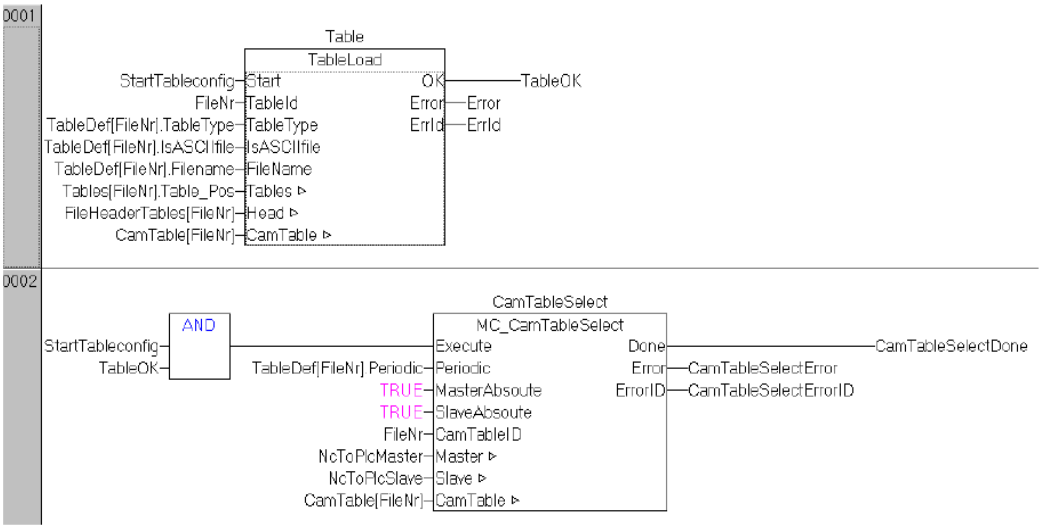
7. 在PLC程序中生成凸轮表

在PLC中生成凸轮表，可以从文件中装载数据，也可以在PLC中运算生成数据，然后通过MC_CamTableSelect功能块来生成凸轮表。

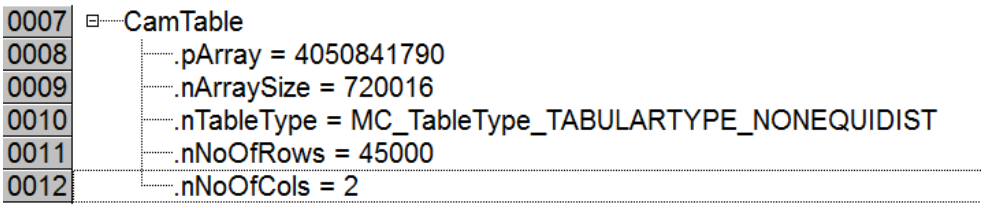
从文件装载凸轮表的过程包括：把文件读入指定格式的结构（MC_CAM_REF），然后把该结构定义的数据装载到 NC。如图：



上图中定义了位置表文件的路径和名称，注意，Filename 变量不接受中文字符。这里的路径，指文件在 PLC 控制器上的位置（注：如果是 txt 或者 csv 文件，那么 IsASCIIfile:=true,filename:=选择对应的凸轮数据文件）



上图中的功能块 TableLoad 并不包含在标准的 Beckhoff 所提供的 Lib 文件中，是例子程序中自带的一个功能块。

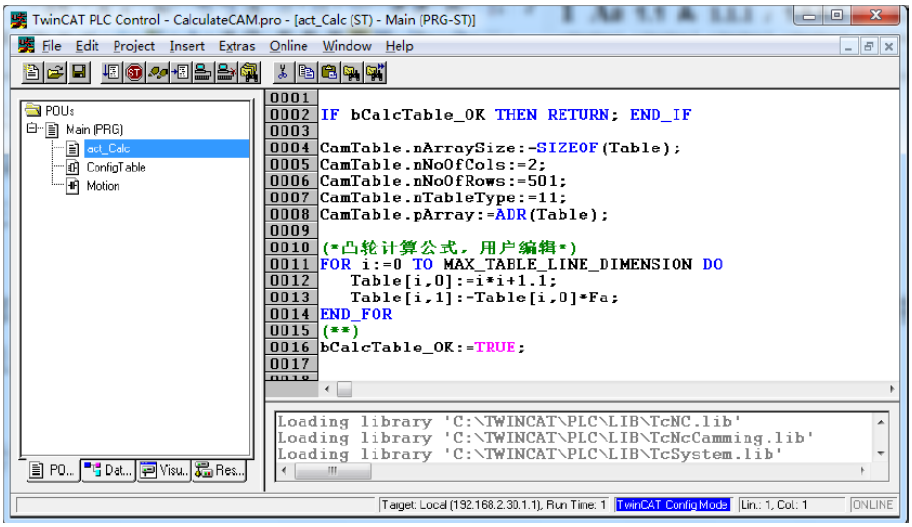


上图的 CamTable 即为 MC_CAM_REF 类型的一个结构体，当 TableLoad 功能块从文件中将凸轮数据装载出来之后会自动将 MC_CAM_REF 结构体填写好，pArray 是存放凸轮数据对应数组的地址，nArraySize 是数组的容量，nTableType 是凸轮表的类型（主轴位置间不等距），nNoOfRows 是数组中列的数量，nNoOfCols 是数组中行的数量。

如果需要从文件中导入凸轮表数据，可以参照 U 盘中提供的例子。

培训用 U 盘\运动控制培训\凸轮飞锯\电子凸轮_TableLoad

在 PLC 中运算生成凸轮表数据，就不需要 TableLoad 这个功能块，只要在程序中对 CamTable 中的位置数组进行赋值即可。如图所示：



八．通用飞锯

本章目标：
通过本章节的学习，学员将了解：
☑ 通用飞锯可以实现的功能
☑ 通用飞锯所需要安装的Supplement
☑ 通过功能块实现通用飞锯的调试

本节描述的Universal Fly Saw功能适用于TwinCAT V2.9 Build 248及以上版本。为区别于“传统飞锯”，称之为“通用飞锯”。新项目推荐使用“通用飞锯”的功能。

飞锯是指从轴可以同步到正在运动的主轴，并与主轴同步运行以完成一个加工周期。这种同步到主轴的运动，意味着工件可以在传输的过程中进行加工。启动飞锯后，从轴在静止或者运动状态均可耦合到运动的主轴上。从轴追上主轴（例如传输物料的运动轴），和物料保持速度同步，并持续一段时间，以便对物料进行加工。加工完成后，同步运行阶段结束，飞锯从轴反向运动，回到起始点，准备下一个周期。

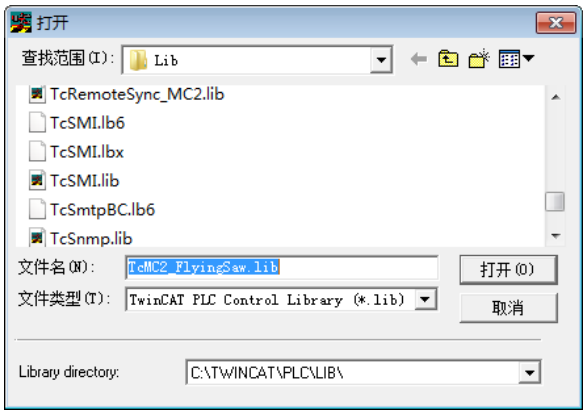
传统飞锯与通用飞锯的一个重要区别在于从轴切入同步运行要求的初始化条件。与传统飞锯不同，通用飞锯即使从轴不在停止状态也可以切入同步运行。此外还计算“改进的位置曲线”，用户可以通过更宽松的约束条件来调整该曲线。

通用飞锯有两种同步方法：速度同步和位置同步。速度同步时，从轴按耦合系数coupling factor尽快同步到主轴，因此主从轴的耦合位置，就是各个参数设定值允许的前提下最快达到同步的位置。位置同步时，用户通过参数设定主从轴的同步位置，主从轴将在最新指定的位置同步运行。

这两种同步方式都要求定义同步阶段的约束条件，即同步模式 SyncMode，以便根据工艺需求调整同步动作。

1. 通过程序来调试飞锯功能

在原先凸轮程序的基础上添加一个 TcMC2_FlyingSaw.lib 的库文件。（注：此库文件需要安装 supplement 才可获得）



在主程序中添加如下功能块以及变量，MC_GearInPos 是位置同步的飞锯功能块，MC_GearInVelo 是速度同步的飞锯功能块，MC_GearOut 是解耦的功能块。

```
0008      camin:mc_camin_v2;
0009      camout:mc_camout;
0010      camin_do:BOOL;
0011      camout_do:BOOL;
0012      gearin_pos:MC_GearInPos;
0013      gearin_velo:MC_GearInVelo;
0014      gearout:MC_GearOut;
0015      gearin_pos_do: BOOL;
0016      gearin_velo_do: BOOL;
0017      gearout_do: BOOL;
0018  END_VAR
```

在程序编写窗口中调用 mc_gearinpos 功能块用于位置同步耦合，将功能块的参数填写完整。

```
0064  gearin_pos(
0065      Execute:=gearin_pos_do ,
0066      RatioNumerator:= 1,
0067      RatioDenominator:=1 ,
0068      MasterSyncPosition:=1000 ,
0069      SlaveSyncPosition:=1000 ,
0070      SyncMode:= ,
0071      MasterStartDistance:= ,
0072      Velocity:= ,
0073      Acceleration:=,
0074      Deceleration:= ,
0075      Jerk:= ,
0076      BufferMode:= ,
0077      Options:= ,
0078      Master:= axis1,
0079      Slave:= axis2,
0080      StartSync=> ,
0081      InSync=> ,
0082      Busy=> ,
0083      Active=> ,
0084      CommandAborted=> ,
0085      Error=> ,
0086      ErrorID=> );
```

功能块参数的说明可以参照 information system 的文档，下图为功能块参数的截图，有些参数有默认值，可以不设置。

Execute	The command is executed with a rising edge at input *Execute*.
RatioNumerator	Gear ratio numerator. Alternatively, the gear ratio can be specified in the enumerator as a floating comma value, if the denominator is 1.
RatioDenominator	Gear ratio denominator
MasterSyncPosition	The master's synchronous position
SlaveSyncPosition	The slave's synchronous position
SyncMode	In the data structure SyncMode boundary conditions for the synchronisation process are specified via individual flags.
MasterStartDistance	Currently not implemented
Velocity	Maximum slave velocity in the synchronisation phase. If a velocity is not specified, the maximum velocity of the axis from the system manager data is used. **Warning:** The velocity given here is only checked if this checking is activated through the *SyncMode* variable.
Acceleration	Maximum slave acceleration in the synchronisation phase. If an acceleration is not specified, the maximum acceleration of the axis from the system manager data is used. **Warning:** The acceleration given here is only checked if this checking is activated through the *SyncMode* variable.
Deceleration	Maximum slave deceleration in the synchronisation phase. If a deceleration is not specified, the maximum deceleration of the axis from the system manager data is used. **Warning:** The deceleration given here is only checked if this checking is activated through the *SyncMode* variable.
Jerk	Maximum slave jerk in the synchronisation phase. If a jerk is not specified, the maximum jerk of the axis from the system manager data is used. **Warning:** The jerk given here is only checked if this checking is activated through the *SyncMode* variable.
BufferMode	Currently not implemented
Options	Currently not implemented

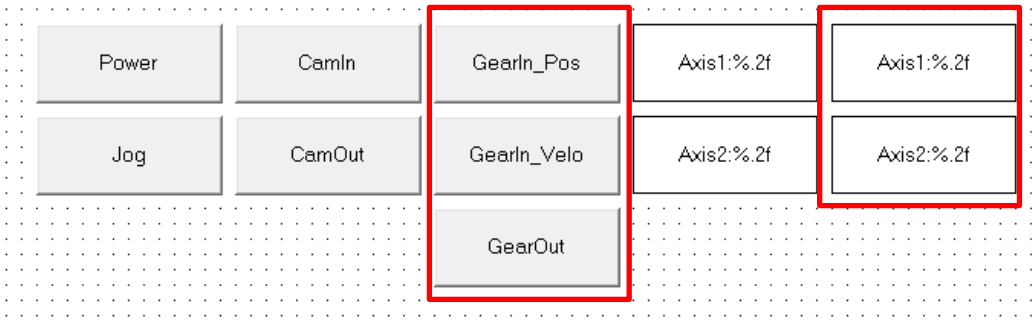
再在程序中调用 mc_gearinvelo 功能块用于速度同步耦合，将功能块的参数填写完整。

```
0087 gearin_velo(
0088     Execute:=gearin_velo_do ,
0089     RatioNumerator:=1 ,
0090     RatioDenominator:=1 ,
0091     SyncMode:= ,
0092     Velocity:= ,
0093     Acceleration:= ,
0094     Deceleration:= ,
0095     Jerk:= ,
0096     BufferMode:= ,
0097     Options:= ,
0098     Master:= axis1,
0099     Slave:= axis2,
0100     StartSync=> ,
0101     InSync=> ,
0102     Busy=> ,
0103     Active=> ,
0104     CommandAborted=> ,
0105     Error=> ,
0106     ErrorID=> );
```

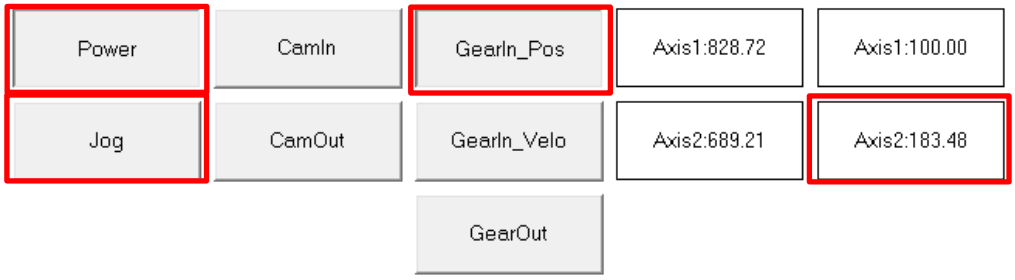
最后在程序编写窗口中调用 mc_gearout 功能块用于解耦，将功能块的参数填写完整。

```
0107 gearout(
0108     Execute:=gearout_do ,
0109     Options:= ,
0110     Slave:= axis2,
0111     Done=> ,
0112     Busy=> ,
0113     Error=> ,
0114     ErrorID=> );
```

HMI 中添加相应的一些控件，两个矩形框用来显示轴的速度，三个按钮用来对轴进行位置同步耦合，速度同步耦合以及解耦。



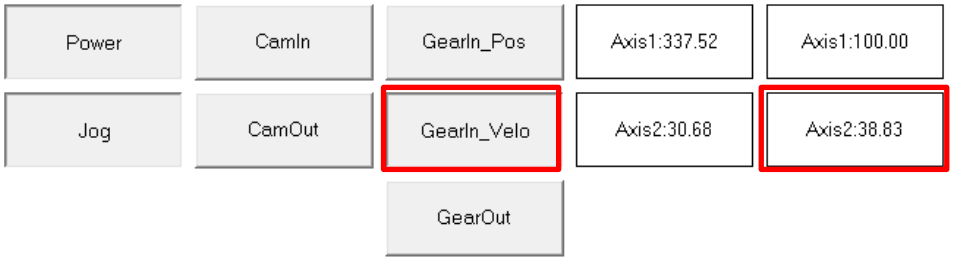
将程序 login 之后，先按下 Power 对轴进行使能，然后按下 JOG 对轴进行点动，此时 Axis1 以 100mm/s 的速度移动，当 Axis1 移动了一段距离后，按下 GearIn_Pos 按钮，可以看到 Axis2 以 183.48mm/s 的速度开始移动，此时 Axis2 相当于在追赶 Axis1。



在设定的 MasterSyncPosition：1000，SlaveSyncPosition：1000 这个位置，从轴赶上主轴并实现速度同步，两个轴的速度都变为 100mm/s，上述方式为位置同步，指定一个主从轴位置，在此位置实现速度同步。

按下 Jog 之后先让轴停下来，按下 Gearout 对轴进行解耦，然后通过 system manager 软件对两个轴的位置进行复位。

接下来测试速度同步耦合功能，首先按下 Jog 对轴进行点动，Axis1 依然是以 100mm/s 的速度移动，然后按下 GearIn_Velo 执行速度同步耦合，可以看到 Axis2 从 0mm/s 开始加速。



Axis2 会在最短的时间内加速到 100mm/s 与 Axis1 保持速度同步，这种方式为速度同步，不需要指定一个具体的位置让主从轴保持速度同步，只是让从轴以最快的加速度速度追上主轴。

通用飞锯功能可以参照 U 盘中提供的例子
培训用 U 盘\运动控制培训\凸轮飞锯

九． TwinCAT NC Fifo

本章目标：

通过本章节的学习，学员将了解：

☑ 先入先出轴可以实现的功能

☑ 先入先出轴对应的功能块

☑ 先入先出轴的使用方法

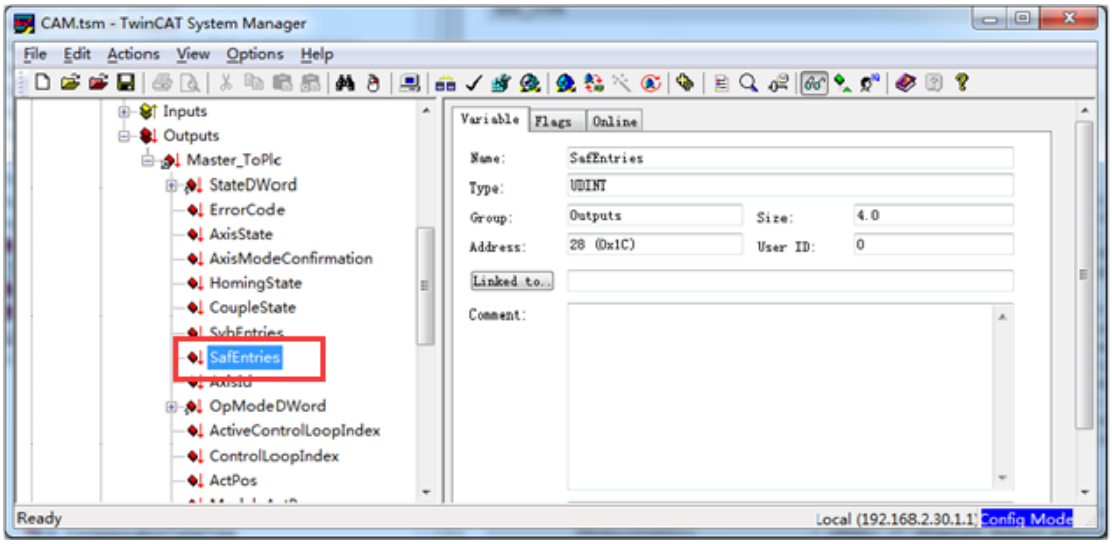
1. 先入先出轴可以实现的功能

FIFO是First Input First Output的缩写。TwinCAT NC FIFO是TwinCAT NC中的一个堆栈区。堆栈区中存放的是一个n维数组，数组中的值就是n个轴的位置序列。这些位置序列以先进先出的方式，依次作为设定位置发送给各个NC轴。

与单轴的External set value generation功能相比，这个功能类似于外部位置发生器（ExtSetpointGenerator），都是由用户自定义位置序列发送给NC轴作为设定位置，代替NC本身的位置发生器。区别有两点，一是FIFO功能允许同时给最多16个轴发送位置而ExtSetpointGenerator只能输出给一个轴；二是FIFO功能的位置序列允许自定义完成相邻两行位置之间的时间间隔，实际上就是NC会在相邻两行之间以NC周期插值。比如第1个点位置是0.0，第2个点位置是1.0，如果时间间隔为10ms，而NC周期为2ms，则NC会将10ms分为5段，每2ms发送一个设定位置，依次为0.2，0.4，0.6，0.8，1.0，保证在第10ms的时候，位置达到1.0。Fifo可使运动更加平稳。

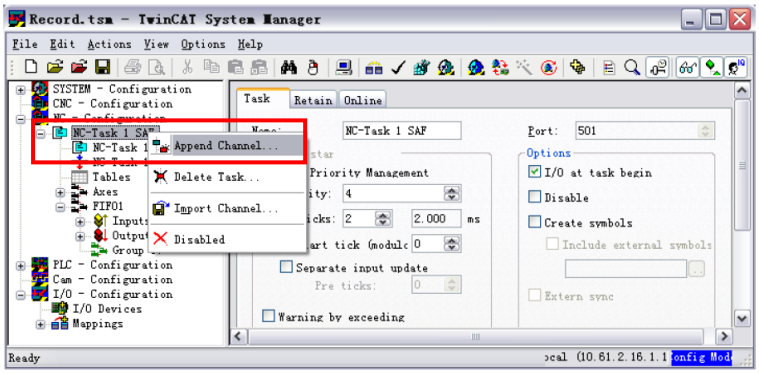
如果FIFO组中的轴数可以自定义，默认值为1。TwinCAT NC FIFO类似描点式（Fixed Table）电子凸轮表，但是Fifo消耗资源少，灵活性较差，不支持关键点的方式，也不能在线修改位置点。并且Fifo组内没有主从轴之分，不能根据主轴的速度变化调节从轴速度。FIFO运动不能反转，从堆栈中完成的位置序列不再保留，如果要重复动作，只能重新装载数据。

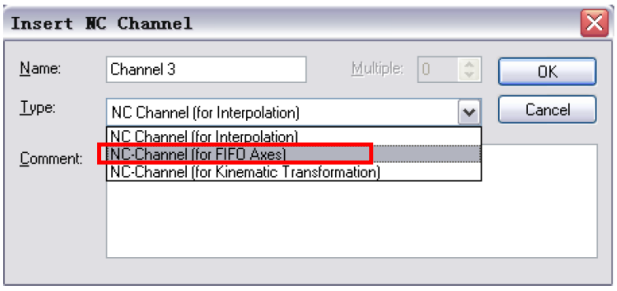
FIFO 堆栈区数据的大小可以自定义，默认值为 1000 行。堆栈数据的 input，是从 PLC 中经功能块 FiFoWrite 写入的。而堆栈数据的 output，是从 PLC 中经功能块 FiFoStart 流出的。写入数据一次可以写入多行，流出数据就只能按照定义好的时间间隔。比如间隔为 10ms，则 1 秒种流出 100 行数据。至于堆栈中还剩作多少行数据，可以从 FIFO 组内任意 NC 轴的接口变量 NCTOPLC_AXLESTRUCT 结构中找到 SafEntries，如图所示：

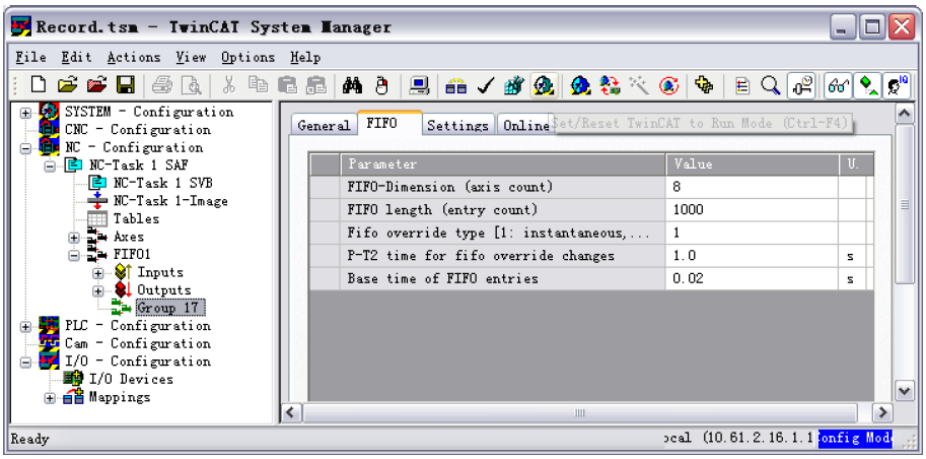


TwinCAT NC 中允许创建多个 FIFO 组，每个组的 ID 号是唯一的。在 PLC 程序中 FIFO 组的操作，都是通过 ID 号识别的。

2. 配置 TwinCAT NC Fifo 组







Fifo Dimension（Axis count）：Fifo组中的运动轴的数量。一个Fifo组最多可以控制16个轴,有的控制器一个Fifo组最多只能控制8个轴。

Fifo Length：Fifo位置表的Buffer容量。以图中所示为例，Base time为0.02s，Fifo Length为1000，则Buffer中的数据能维持运行20秒。缓存越小，则数据从PLC传送到NC Fifo就越频繁。Buffer越大，消耗计算机内存越多。

Fifo Override Type：通过Override，可以调节运动速度。Override的切换方式可以选择阶跃型（Instantaneous override）或者平滑型（PT-2 override）。

P-T2 time for override changes：当Fifo Override Type选择平滑型（PT-2 override）切换方式时，在此处设置切换时间。时间越长，切换越平滑。

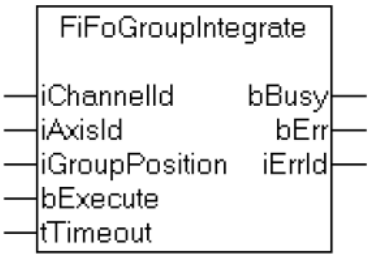
Base Time of Fifo entries：连续两个位置点之间的时间间隔，Base Time必须是NC周期的整数倍，NC会在连续两个位置点之间进行插补，并自动计算运动速度。

说明：

外部给定位置值列表，可以从文件中读取，也可以在 PLC 程序中在线生成。当 Fifo 表驱动实际硬件时，由于速度不是在程序中指定，而是由文件中位置表决定，所以建议先在

虚轴上运行，观察各轴运行 Fifo 表的最大速度，然后确认伺服驱动器和电机确实能够支持该速度。这一过程，又称为曲线校验。

3.Fifo 功能所对应的功能块



FiFoGroupIntegrate把一个独立的PTP轴集成到一个Fifo组中。变量iGroupPosition指定了它在Fifo组中的序号。

接口变量

iChannelId: FIFO channel的ID号.

iAxisId: 轴的ID.

iGroupPosition: 将轴集成到Fifo组后，它在该组中的序号，首序号为1).

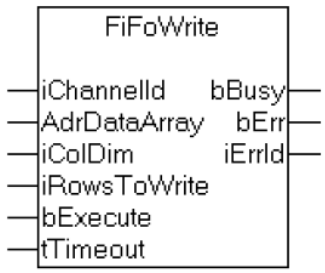
bExecute: 上升沿触发本功能块动作.

tTimeout: ADS timeout (约1 s).

bBusy: bExecute的上升沿bBusy置True, 完成后为False。

bErr: 指令执行过程中出错则置为True。.

bErrId: 错误代码(ADS 或NC错误代码).



FiFoWrite把指定数组中的数据，写到TwinCAT NC Fifo组的位置缓存表（Buffer）。

接口变量

iChannelId: FIFO channel的ID号.

AdrDataArray: 位置表数组的指针。该数组应该是一个二维数组。“列”表示轴的数量，“行”表示每个轴的位置点数量。

iRowsToWrite: 写入行数. 必须小于等于位置表数组的行数。

bExecute: 上升沿触发本功能块动作.

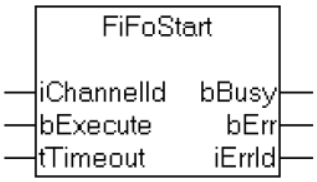
tTimeout: ADS timeout (约1 s).

bBusy: bExecute的上升沿bBusy置True, 完成后为False。

bErr: 指令执行过程中出错则置为True。.

bErrId: 错误代码(ADS 或NC错误代码).

FiFoStart



FiFoStart启动Fifo组内各轴按照此前接收并存储在Buffer中的位置表运动。

接口变量

iChannelId: FIFO channel的ID号.

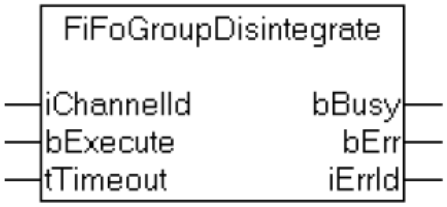
bExecute: 上升沿触发本功能块动作.

tTimeout: ADS timeout (约1 s).

bBusy: bExecute的上升沿bBusy置True, 完成后为False。

bErr: 指令执行过程中出错则置为True。.

bErrId: 错误代码(ADS 或NC错误代码



FiFoGroupDisintegrate将原先集成到Fifo组中的各轴释放为独立的PTP轴。

iChannelId: FIFO channel的ID号.

bExecute: 上升沿触发本功能块动作.

tTimeout: ADS timeout (约1 s).

bBusy: bExecute的上升沿bBusy置True, 完成后为False。

bErr: 指令执行过程中出错则置为True。.

bErrId: 错误代码(ADS 或NC错误代码).

注意：Fifo表的停止是通过将Override变为0，而不是通过功能块FifoStop。

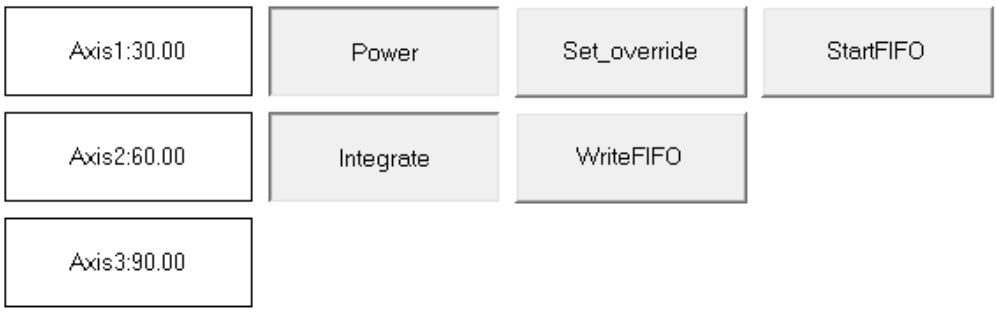
4.FIFO功能配置例程

先入先出轴功能可以参照 U 盘中提供的例子

培训用U盘\运动控制培训\FIFO先入先出轴

- 程序调试

将U盘中提供的例子程序和配置文件打开，将配置激活，程序登录，这里选择Local做为目标设备，打开HMI界面后直接可以调试功能，首先按下Power按钮对轴进行使能，然后按下Intergrate将轴分别加入Fifo功能通道，然后按下Set_override设置Fifo通道的速度比，WriteFIFO将位置数据写入FIFO通道，最后按下StartFIFO开启先入先出轴的功能，可以看到Axis1,Axis2,Axis3分别走到30,60,90位置。



- 程序以及配置分析

首先查看FIFO通道中的设置，此窗口中设置轴个数，缓存长度，Override切换方式以及FIFO基本事件，这里设置缓存位置为30，base time 为1s，那么30个缓存的数据一共可以控制FIFO通道运行30s。

General | FIFO | Settings | Online

Parameter	Value	Type	Unit
FIFO-Dimension (axis count)	3	D	
FIFO length (entry count)	30	D	
Fifo override type [1: instantaneous, 2: P-T2]	2	D	
P-T2 time for fifo override changes	1.0	F	s
Base time of FIFO entries	1.0	F	s

此功能块的作用是将三根NC轴添加到同一个FIFO通道

```
0039 FIFO_integrate(
0040     iChannelId:= 2,
0041     iAxisId:=1 ,
0042     iGroupPosition:=1 ,
0043     bExecute:=integrate_do ,
0044     tTimeout:= ,
0045     bBusy=> ,
0046     bErr=> ,
0047     iErrId=> );
```

integrate_do = TRUE

FIFO_Setoverride 设置 FIFO 通道的速度比为 100%

```
0066 FIFO_Setoverride(
0067     iChannelId:=2 ,
0068     iOverride:=1000000 ,
0069     bExecute:=setoverride_do ,
0070     tTimeout:= ,
0071     bBusy=> ,
0072     bErr=> ,
0073     iErrId=> );
```

setoverride... = FA...

通过 PLC 程序计算出 FIFO 通道所需要的位置数据，赋值到 Pos_arr 这个两维数组中，位置数据也可以存在 CSV 或者 TXT 文件中，通过功能块进行读取，也可以通过上位机通信写入。

```
0075 IF r_trig1.Q =TRUE THEN
0076 FOR n:=1 TO 30 BY 1 DO
0077 Pos_arr[n,1]:=pos_inc+1;
0078 Pos_arr[n,2]:=Pos_arr[n,1]*2;
0079 Pos_arr[n,3]:=pos_arr[n,1]*3;
0080 pos_inc:=pos_inc+1;
0081 END_FOR
```

```
r_trig1.Q = FALSE
n = 31
Pos_arr[n,1] = ???
Pos_arr[n,2] = ???
Pos_arr[n,3] = ???
POS_INC = 30
n = 31
```

```
n = 31
n = 31
n = 31
```

```
POS_INC = 30
Pos_arr[n,1] = ???
Pos_arr[n,1] = ???
```

FIFO_WRITE 将准备好的位置数据写入到 FIFO 通道的缓存中

```
0083 FIFO_write(
0084     iChannelId:=2 ,
0085     AdrDataArray:=ADR(pos_arr) ,
0086     iColDim:=3,
0087     iRowsToWrite:=30 ,
0088     bExecute:= write_do,
0089     tTimeout:= ,
0090     bBusy=> ,
0091     bErr=> ,
0092     iErrId=> );
```

```
write_do = FALSE
```

最终触发 FIFO_START 功能块即可让 FIFO 通道开始工作。

```
0093 FIFO_START(
0094     iChannelId:=2 ,
0095     bExecute:=start_do ,
0096     tTimeout:= ,
0097     bBusy=> ,
0098     bErr=> ,
0099     iErrId=> );
```

```
start_do = FALSE
```

学员提问：如何通过读文件的方式将位置数据读取到FIFO通道中？

讲师解答：建议使用 FB_XmlSrvRead 功能块进行读取，位置数据都写在一个 xml 文件中，触发 FB_XmlSrvRead 功能块就可以读取位置数据，也可以通过 FB_FileOpen,FB_FileRead,FB_FileClose 对 csv 或者 txt 文件进行读取。

十． 完整配置 AX5000 和电机

本章目标：

通过本章节的学习，学员将了解：

☑ 如何设置AX5000的供电范围以及制动电阻

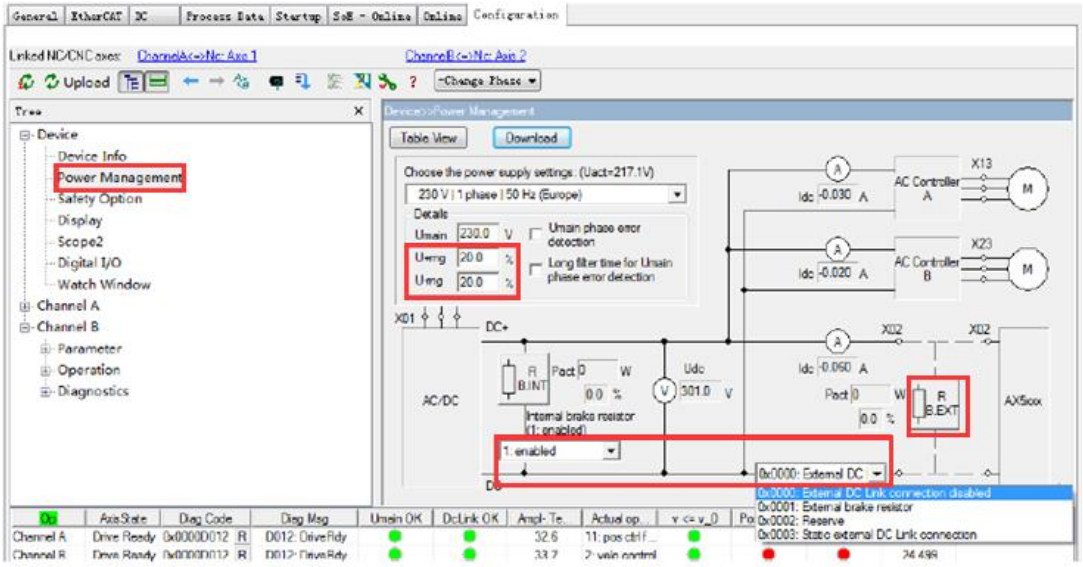
☑ 如何设置AX5000的工作模式

☑ 如何设置AX5000的PID参数

☑ 如何设置AX5000的启动参数

☑ 如何通过PLC读写驱动器的内部参数

1. 电源设置

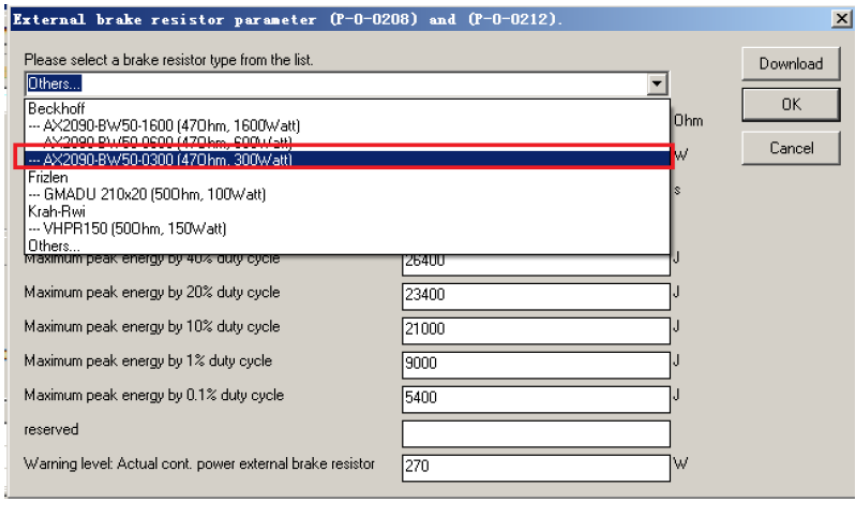


通常实际接入的电源电压等级，在扫描驱动器的时候系统会自动设置。

需要注意的是：

1， 电压波动范围的设置。默认的波动范围是+－10%，供电电压超过这个范围就会报错。但是根据国内的电网质量，设置为+－20%较为合适。

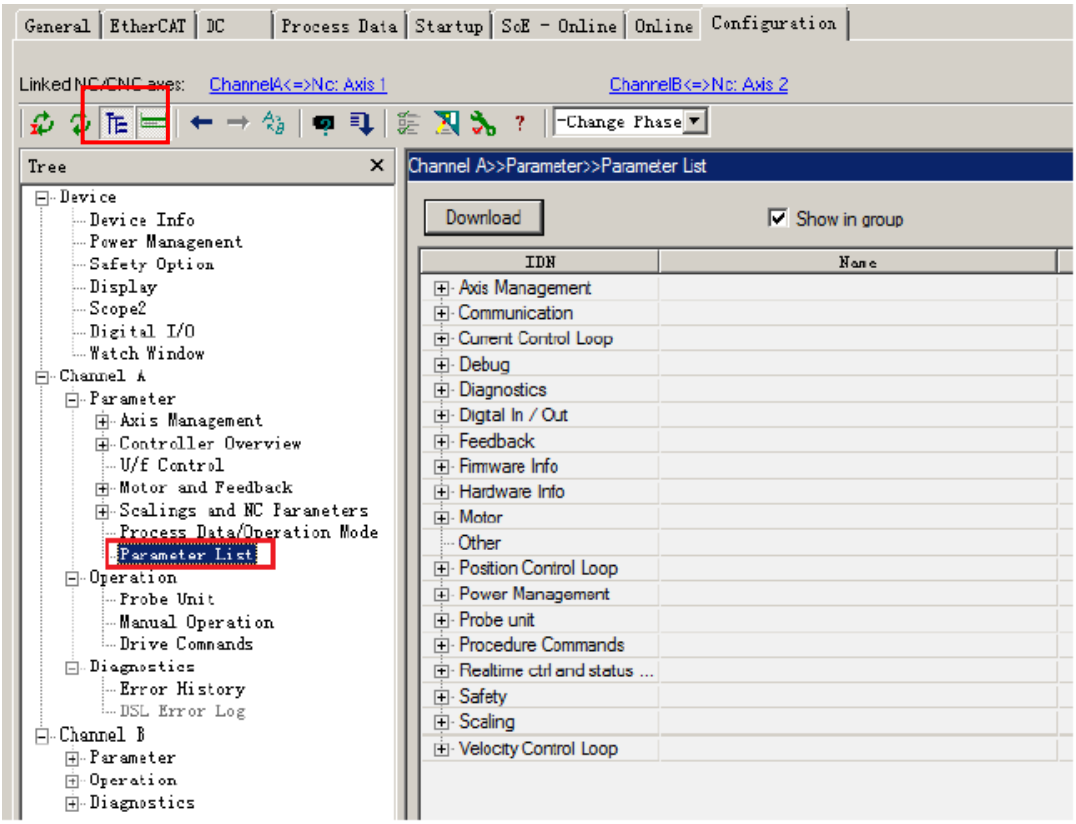
2， 制动电阻的设置。默认是启用AX5000内置的制动电阻。如果需要启用外部制动电阻，则需要在上图中选择“0x0001:External Brake Resistor”。然后点击按钮RB. EXT，进入制动电阻参数设置页面：



学员提问：是否可以使用第三方的制动电阻？
讲师解答：如果是倍福原厂电阻，直接选择型号即可。如果是国产电阻，则可以选择一个电阻和功率接近的原厂电阻，在此基础上再修改参数。

- 修改其它参数

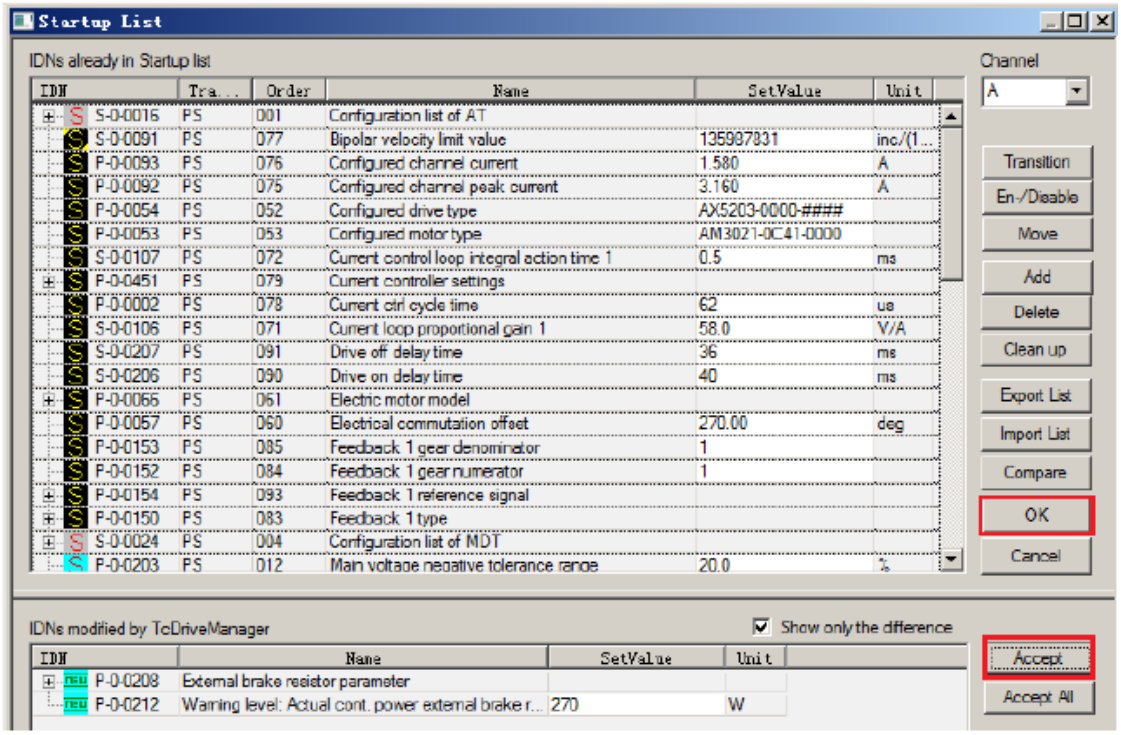
AX5000的总线接口协议为SercosOver EtherCAT。其内部参数是按照Sercos协议中规定的P参数、S参数来组织的。在AX5000的Configuration页面，确认顶部的红色框中两个按钮按下。分设设置Channel A和Channel B的参数。



可以展开左边的树形结构，选择目标项，进入图形化的设置界面。也可以双击上图中红线框内的“Parameter List”，进入纯参数的设置界面。
在这里不仅可以设置和查看参数，还可以手动松开或者合上抱闸（Manual Operation），或者手动发出校准磁偏角或者硬件复位的命令（Drive Command），以及查看历史报警信息（Error History）。当然，所有这些操作都可以通过 PLC 程序实现。

- 在Startup List中确认修改项

点击下图的按钮，进入 Startup 确认页面



在这个窗体设置的参数，直接点击OK，激活配置就生效了。此后每次TwinCAT启动时这些参数就会写入驱动器。

如果是在其它界面做的参数修改，在这个窗体的下半部就会显示修改项，点击“Accept”，就会自动把修改项添加到上半部的Startup List列表中。也是激活配置才生效。

此外，这个窗体上还有伺服通道参数的导入（Import List）、导出（Export List）和比较（Compare）功能。调试过程中，要尝试不同参数（比如三环 PID 参数等）的组合，就可以灵活使用这 3 个按钮。

2. AX5000的工作模式OP Mode

- 模式描述

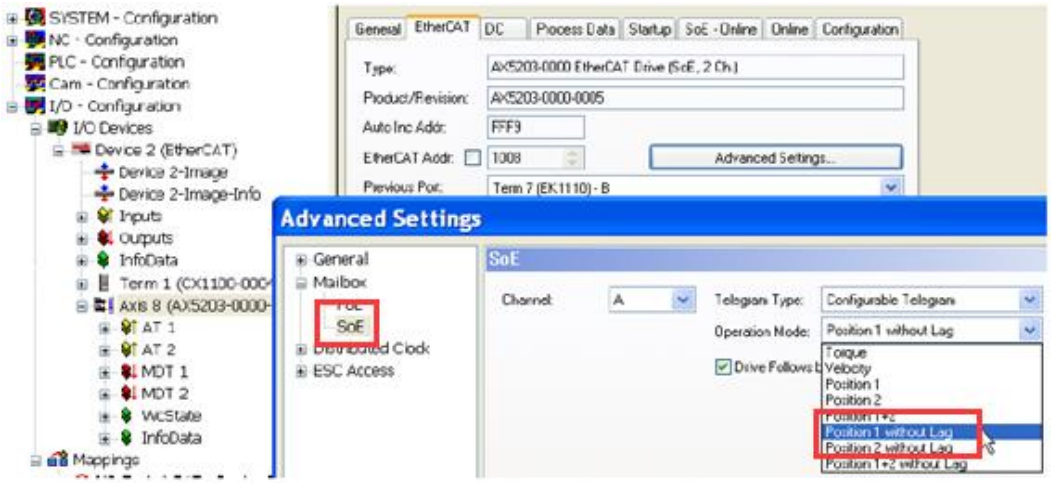
AX5000可以工作在位置模式、速度模式和转矩模式。

通常情况下，AX5000工作在位置模式，此时TwinCAT NC只负责路径规划，并在每个NC周期发送目标位置给AX5000。

如果AX5000工作在速度模式，那么TwinCAT NC不仅要做路径规划，还要完成位置环的PID控制。每个NC周期把PID的输出转换成目标速度发给AX5000。

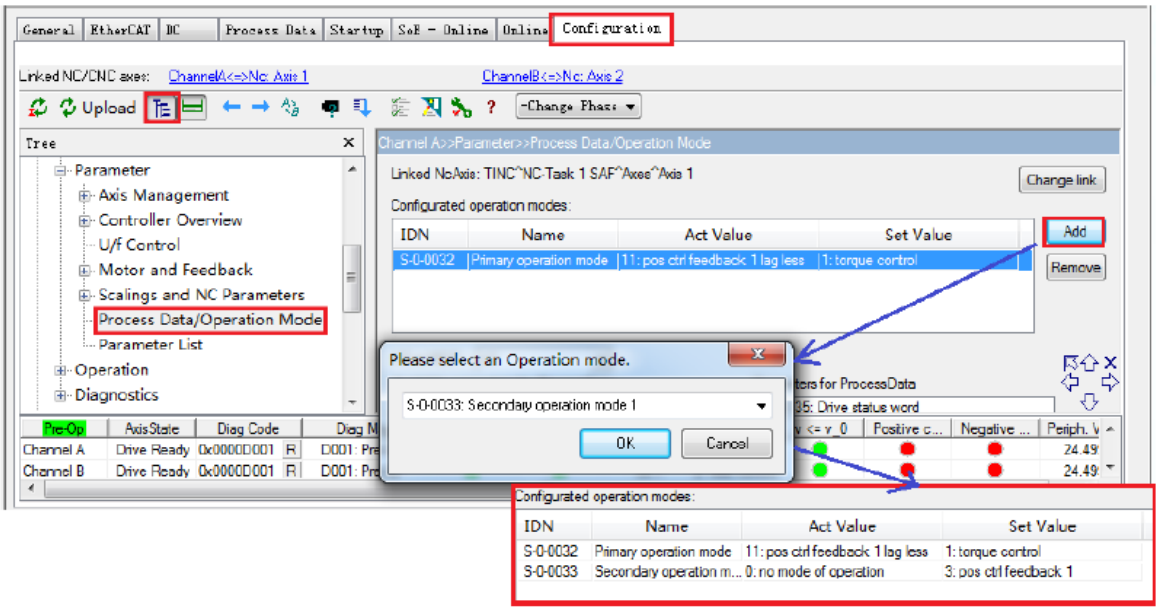
如果AX5000工作在转矩模式，TwinCAT NC就不能控制AX5000的速度或者位置了，而由PLC程序控制AX5000的转矩，TwinCAT NC就只有读取位置反馈信号并换算成位置和速度。

- 选择AX5000的工作模式：



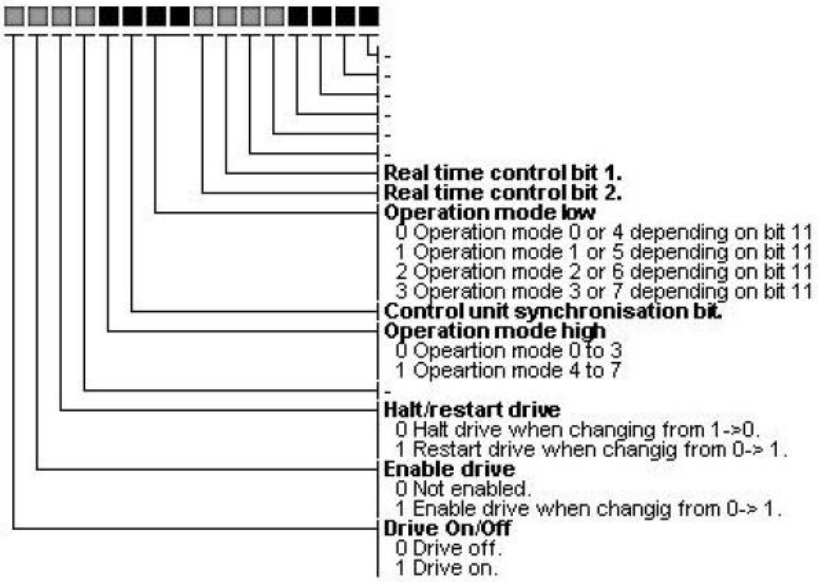
用这种方式修改OP Mode，Process Data会自动更新并链接到相应的NC轴。比如设置为位置模式，AX5000从NC轴接收的变量就是控制字和目标位置（S-0-0047），而送往NC轴的变量就包含了跟随误差（S-0-0189）。如果是速度模式，就不包含跟随误差，同时接收的变量也不是目标位置而是目标速度（S-0-0036）。如果选择了转矩模式，AX5000接收的变量就是控制字和目标转矩（S-0-0080）。

- 设置AX5000的第二操作模式：



第二操作模式可以由控制字Controlword的Bit 8、9来选择，如下图所示。

S-0-0134, Master control word



注意：

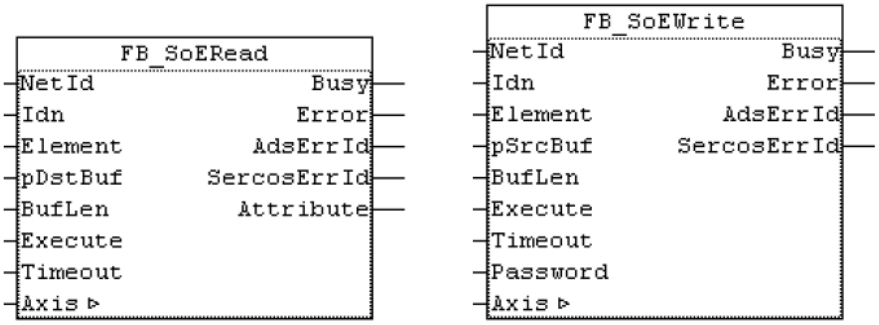
1， 通常AX5000需要模式切换时，是在力矩模式和速度模式之间切换。

2， 控制字不再是直接由NC轴出到AX5000，而是NC输出给PLC变量，PLC变量转换后再输出给AX5000。

3， 使能状态下从速度模式切换到力矩模式，目标力矩足以克服静摩擦后电机就会加速度转动。令目标力矩为0，电机停止。切回速度模式之前，应断开使能，模式切换成功后，再上使能。否则，电机会飞车，以最大速度返回到上次模式切换时的位置。

3.读写 SOE 参数

以 TcMc2Drive.lib 为例。



FB_SoERead 和FB_SoEWrite分别用于读SOE参数和写SOE参数。

输入变量：

NetId: 控制器的NetID，通常PLC程序都是操作本机控制的驱动器参数，留空，用默认值。

Idn: 参数号；“S_0_IDN + 33”表示S-0-0033，“P_0_IDN + 150”表示P-0-0150。

Element: 读取该参数的哪个属性, 通常是读取参数值，此处输入16#40。

pDstBuf:读回的值放到哪个地址，填ADR（DataIn），读回的值就赋给变量“DataIn”。

pSrcBuf:用哪个地址的值来写，填ADR（DataOut），把变量“DataOut”的值写入驱动器。

BufLen:变量长度。填写SizeOf（变量名）。

Execute: 上升沿触发读写操作。

Axis : AXIS_REF，要操作的驱动器所链接的PLC轴变量。

输出变量：

AdsErrId: 如果执行FB时发生错误，在此查看ADS错误代码。

SercosErrId: 如果执行FB时发生错误，在此查看Sercos错误代码。

TcMc2Drive.lib还提供一些读取常用参数的FB，它们都可以用FB_SoERead代替，方便之处在于不用查参数号。比如：

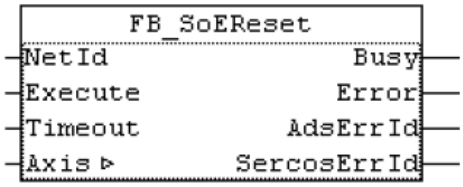
FB_SoEReadAmplifierTemperature，读驱动器温度，

FB_SoEReadMotorTemperature，读电机温度，

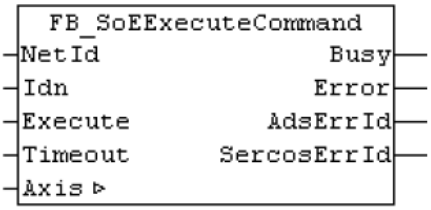
FB_SoEReadDcBusCurrent 和 FB_SoEReadDcBusVoltage 分别读取直流母线电流和电压。

4.从 PLC 程序控制 AX5000 执行硬件命令

在 Drive Manager 中执行的命令，比如驱动器重启、强制打开抱闸、校正磁偏角等动作，都可以通过 PLC 程序中调用特定的功能块实现。以 TcMc2Drive.lib 为例：



用于驱动器复位。

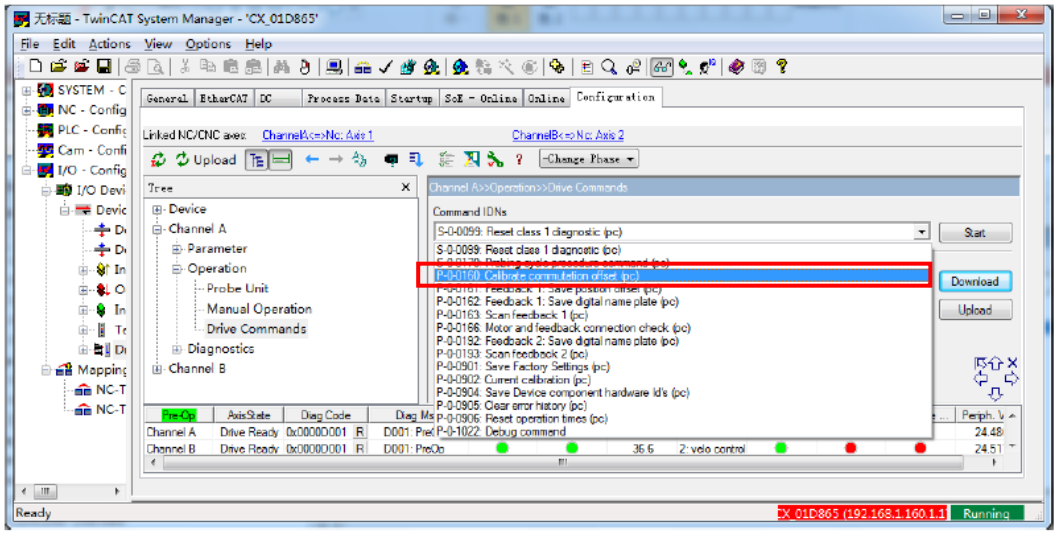


用于执行驱动器命令。

NetID：控制器的NetID，可以不填，表示操作本机所带的伺服驱动器。

Idn ： 命令参数号，比如“P_0_IDN + 160”表示执行寻磁偏角的动作，“S_0_IDN + 99”表示硬件复位动作。

FB_SoEExecuteCommand 等效于下图中的操作：

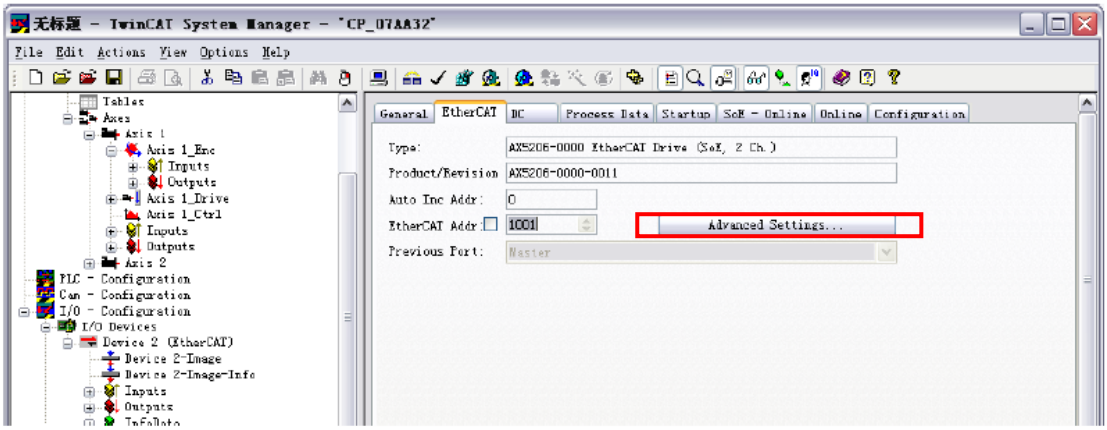


从上图也可以查出特定动作对应的参数号。

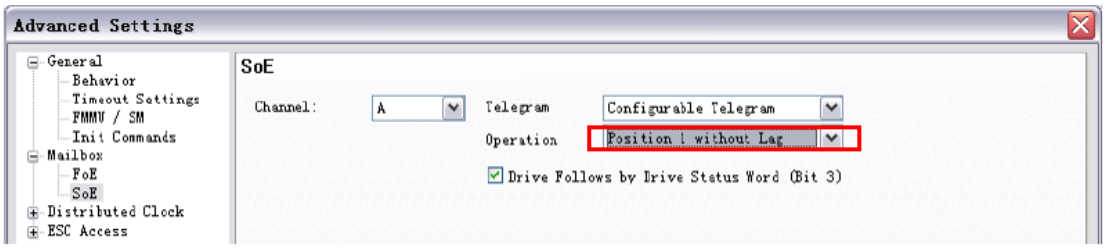
5.AX5000 的 PID 参数调整（速度环）

- 设置AX5000的OP Mode操作模式

在下图，点击 Advanced Settings



在 Mailbox |Soe|Channel A 的 Operation 选项，如图：

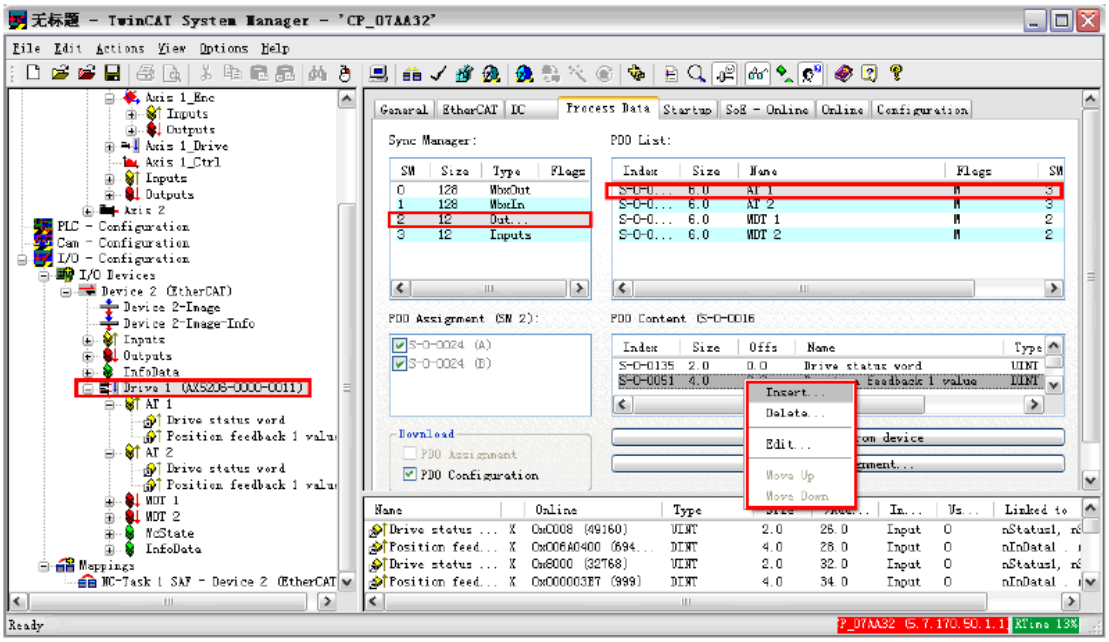


在做速度环阶跃响应时，应将AX5000的操作模式（Operation）设为Velocity模式

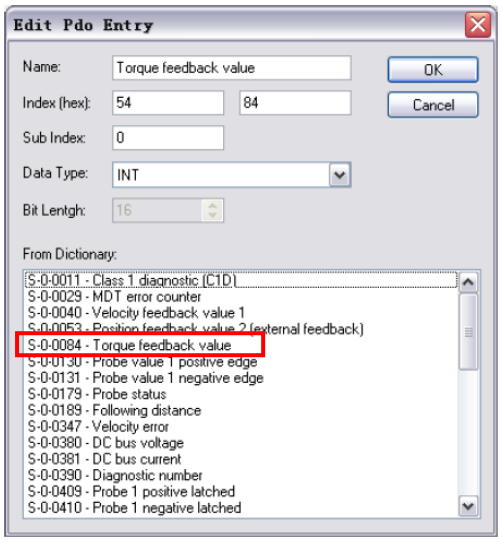
在位置环调节时，应设置为Position 1 without Lag。

- 在Process Data中增加Torque feedback Value

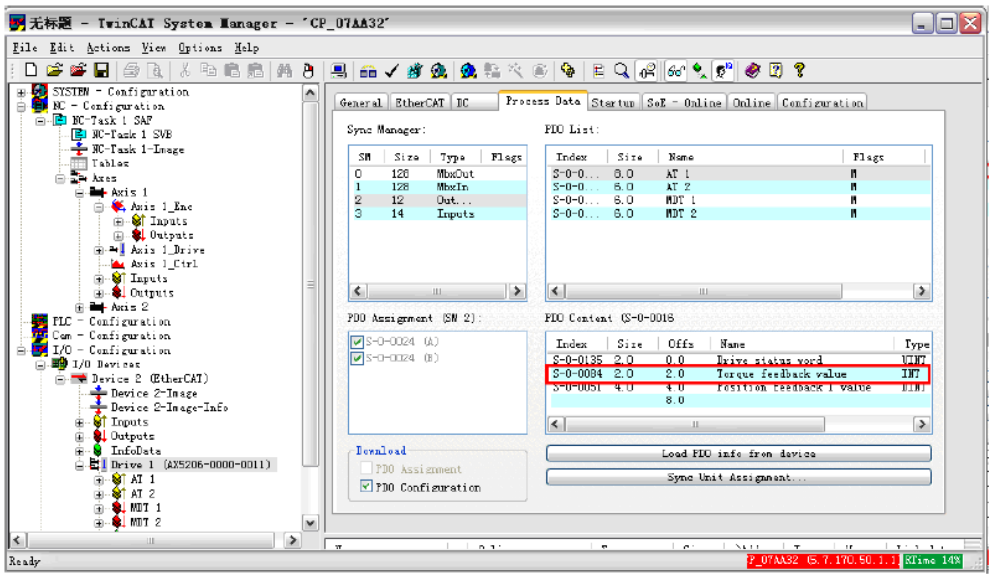
鼠标右键，选择 Insert



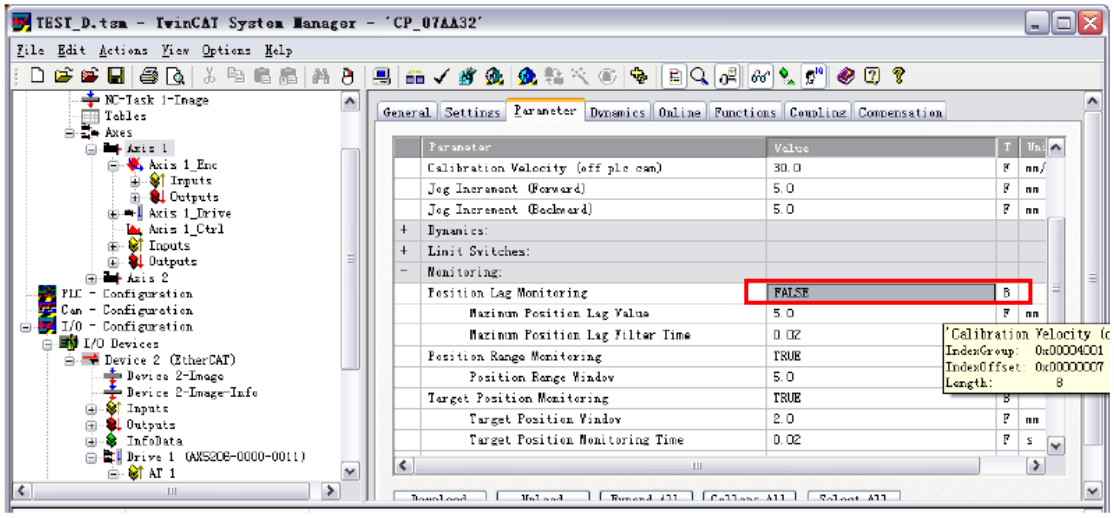
选择 S-0-0084



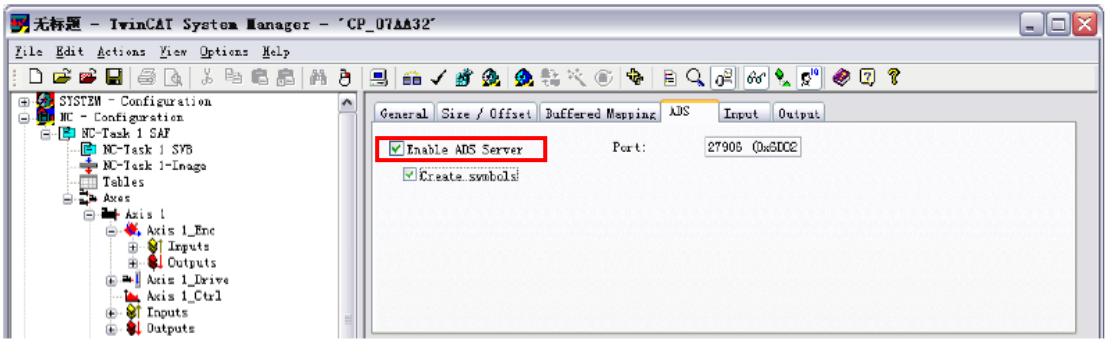
成功添加后，过程变量如图所示：



- 在NC Configuration中Axes的，Axis1,Axis2右侧选项卡Parameter中 展开Monitoring关闭Position Lag Monitor



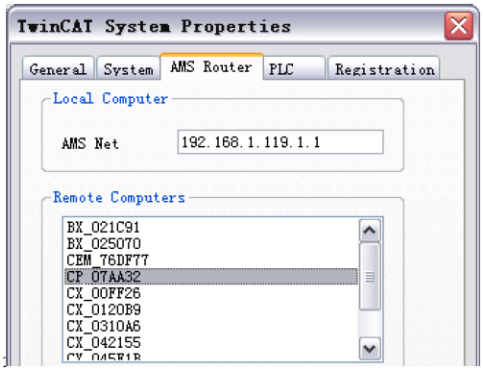
- 使能EtherCAT总线的ADS通讯

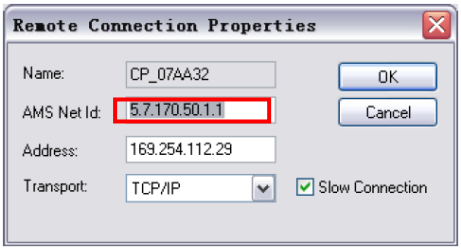


这是为了从 Scope View 中直接监视变量 Torque Feedback Value。

- 确认目标设备的ADS通讯地址

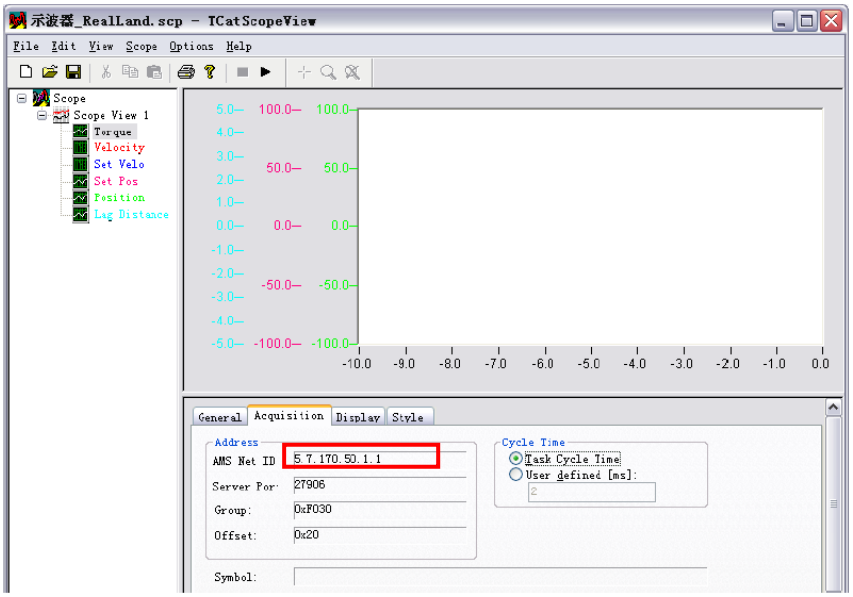
选择目标机器，点击 Properties，





以备 Scope View 中监视变量的设置

- 在 **Scope View** 中设置监视的变量



变量包括：

设置速度，AXES.Axis 1.SetVelo

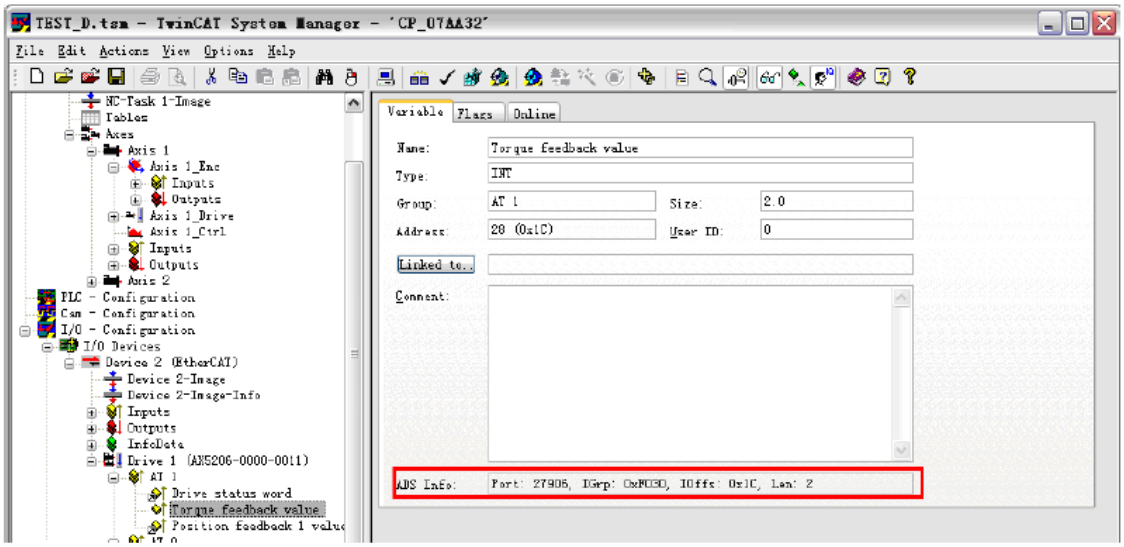
实际速度，AXES.Axis 1.ActVelo

目标位置，AXES.Axis 1.SetPos

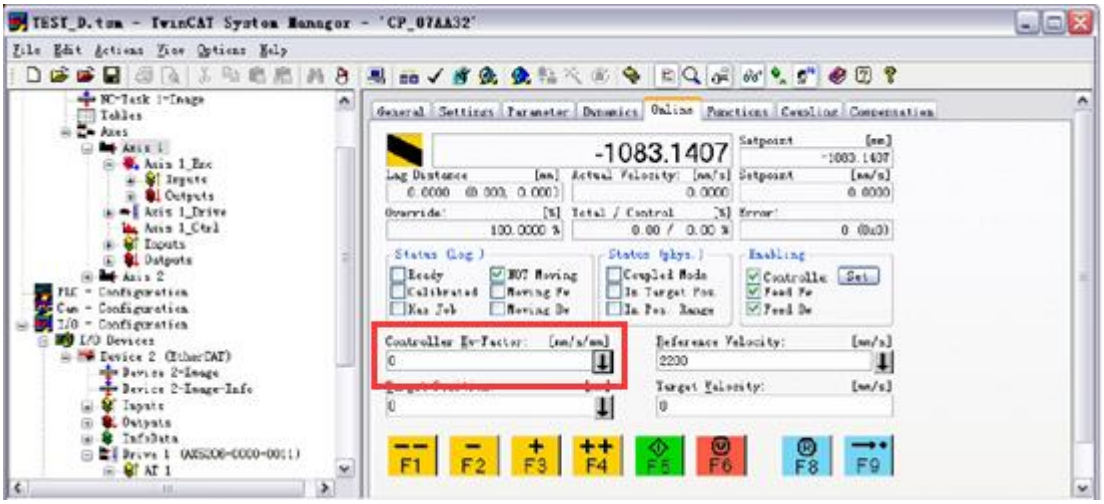
实际位置，AXES.Axis 1.ActPos

跟随误差，AXES.Axis 1.PosDiff

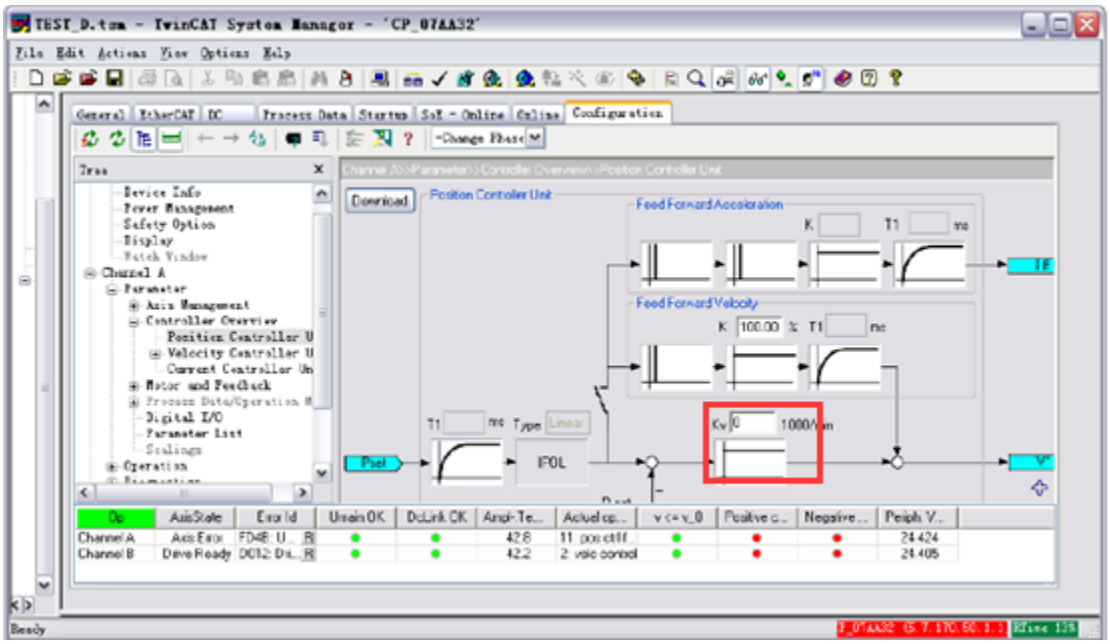
转矩反馈值，以 Index/Group 方式访问，此信息来自下图中显示的 ADS 信息。



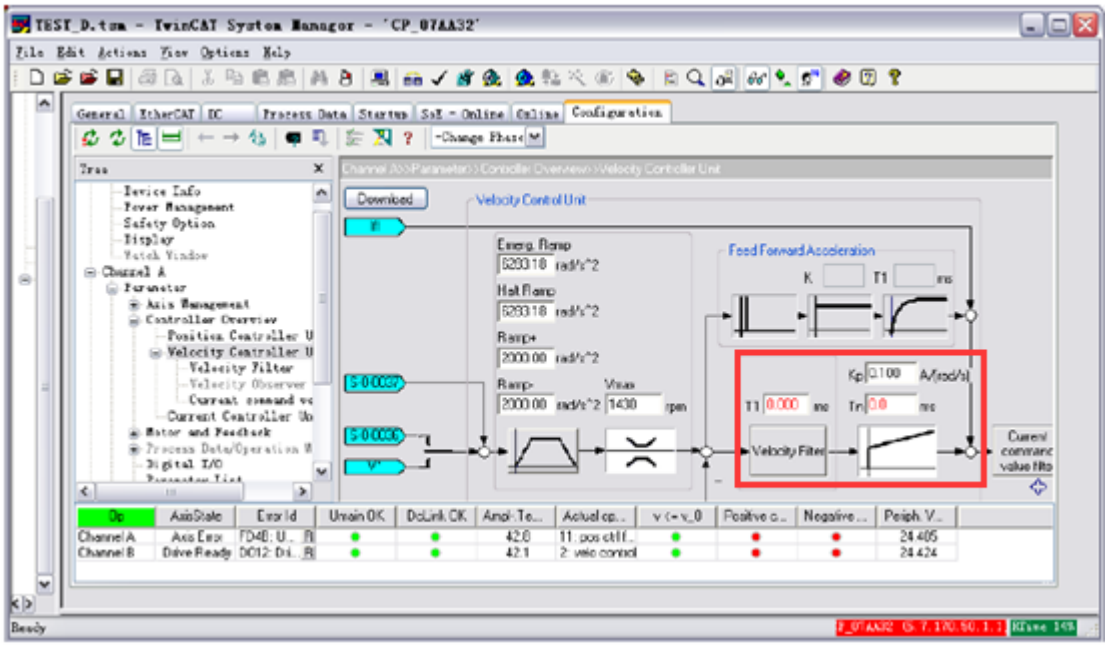
- Kv-Factor 设置为 0，并使能 Axis。



关闭位置环。把 Kv 值设为 0



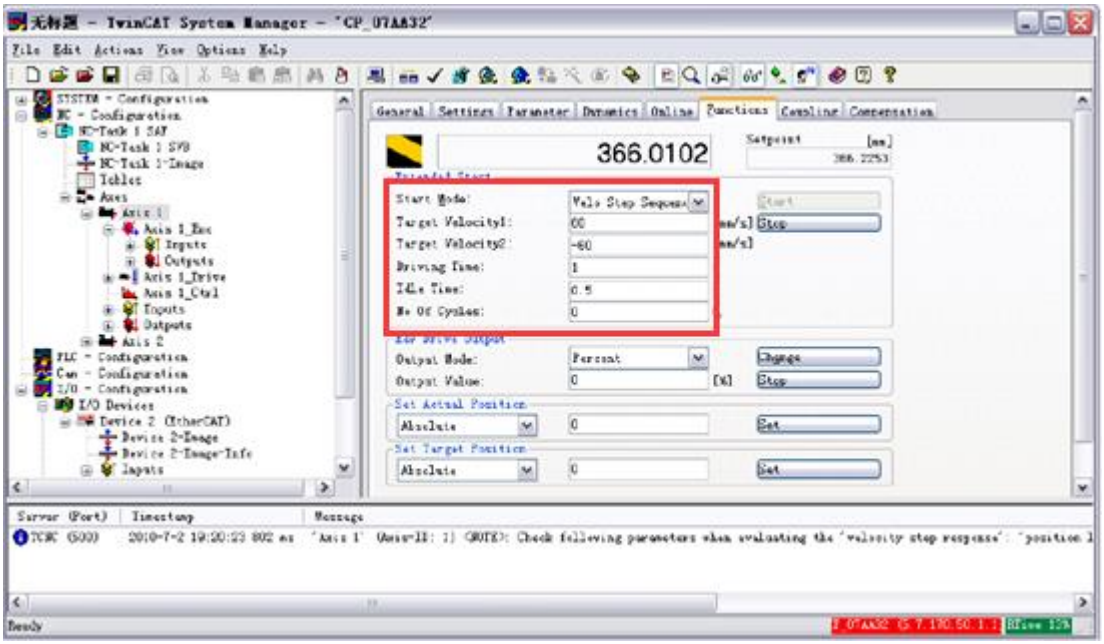
把速度滤波 T1 和积分时间 Tn 设为 0



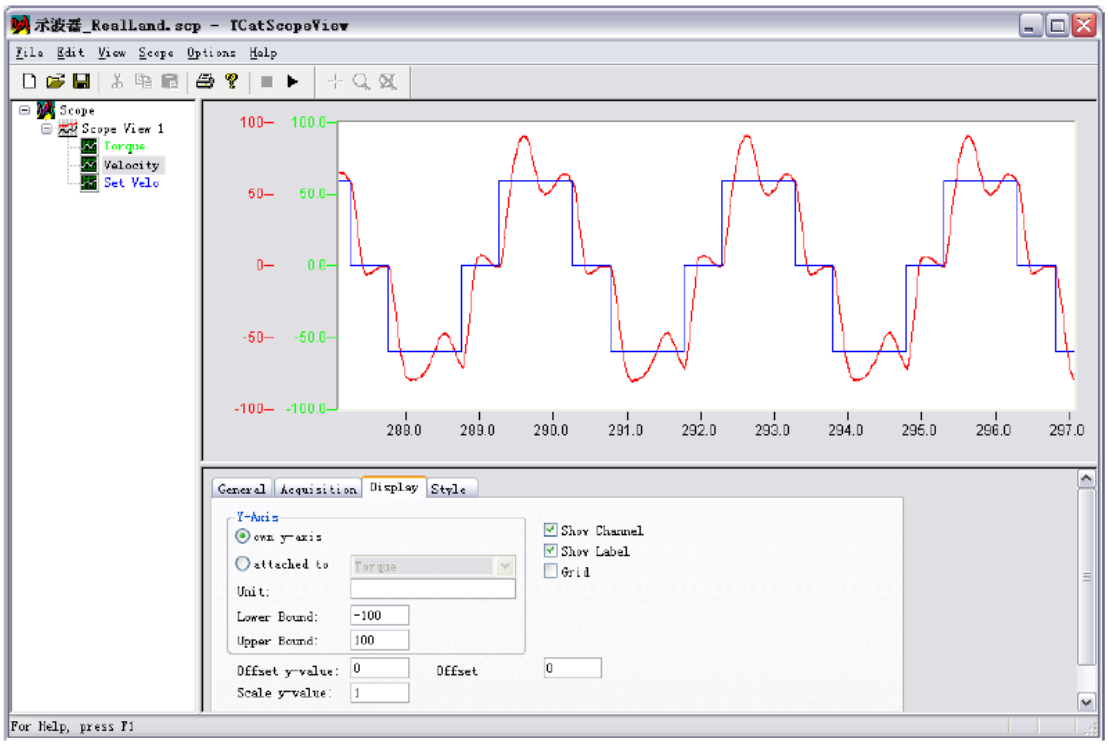
如果驱动器报错，就执行一次 S-0-0099，Reset。

- 阶跃响应

在 Function 页面，让 Axis 作正反速度动作，以调试速度环在阶跃响应下的性能。

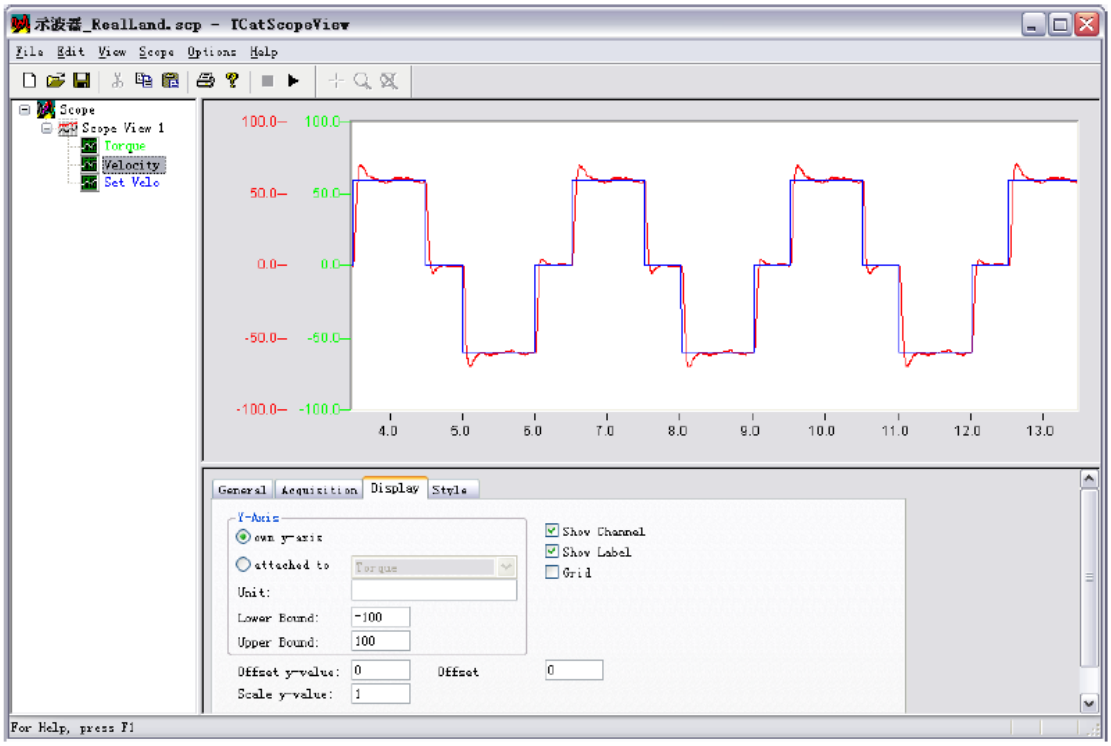


- 在Scope View中观察速度曲线。

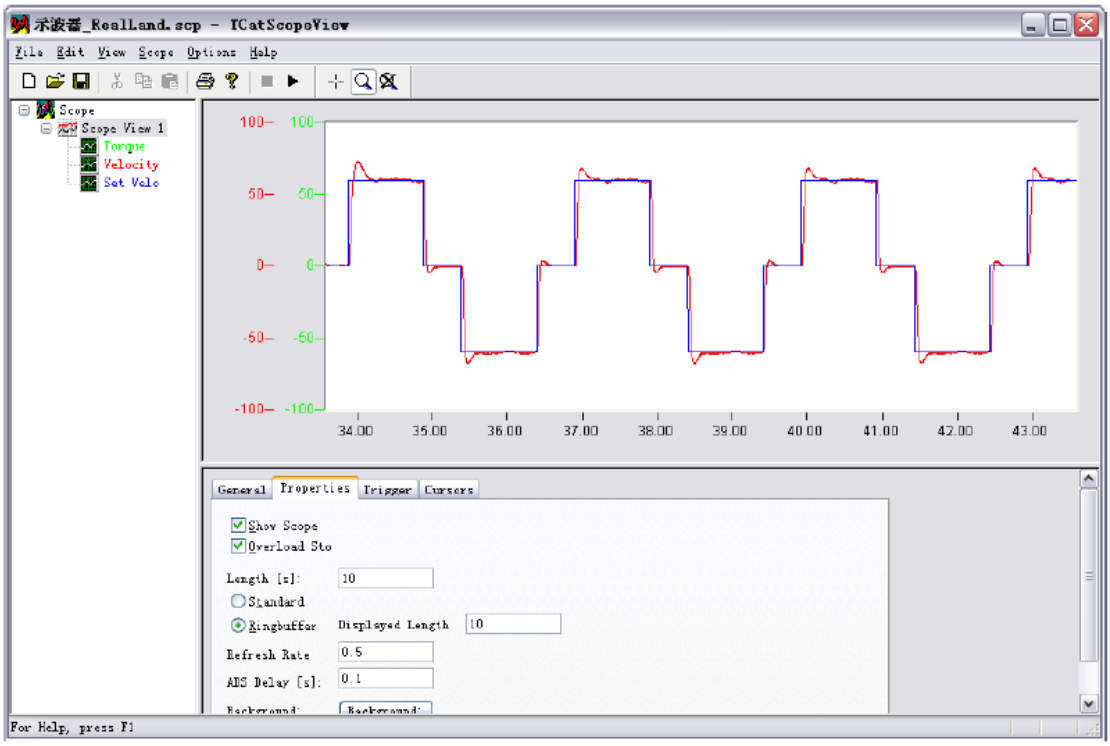


- 调节Kp值：缓慢增加Kp值，使设定值与反馈值尽量一致，但不应该有震荡；

例如：Kp：=0.7

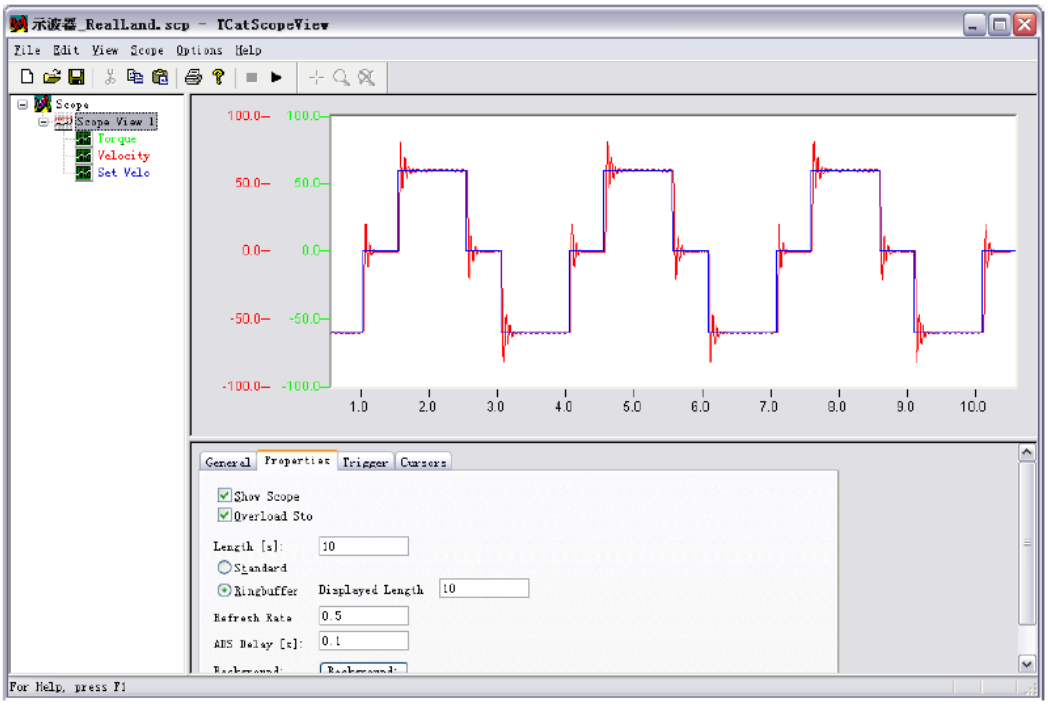


Kp:=1.0

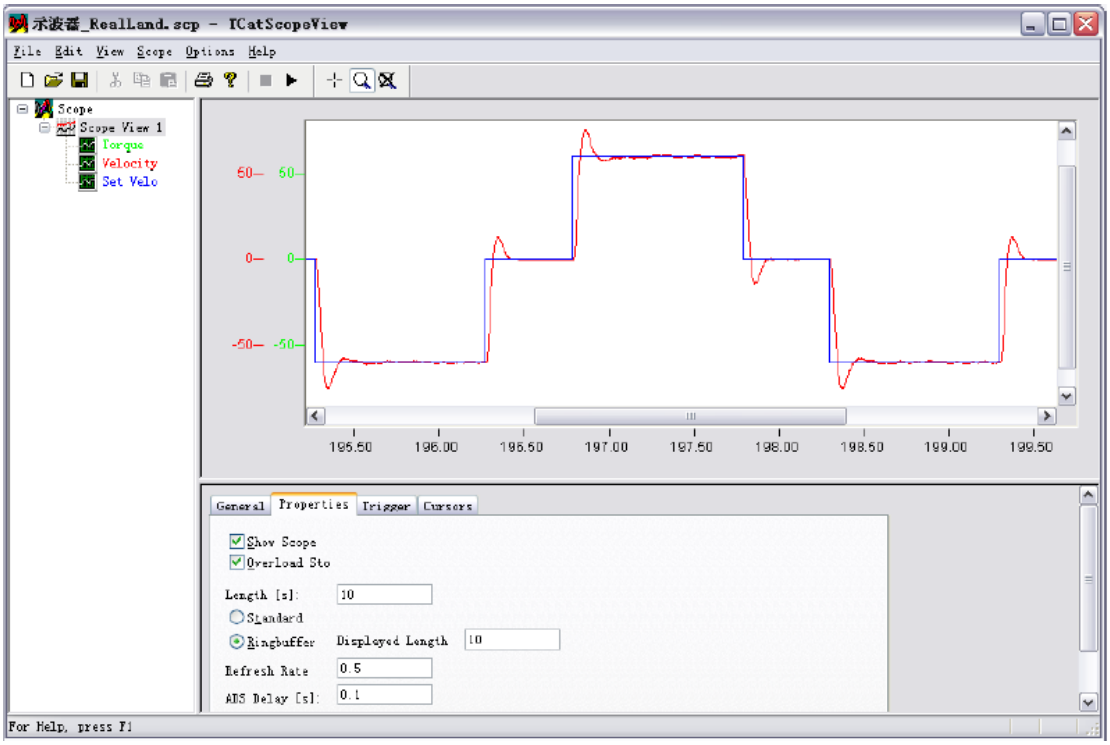


- 调节积分时间值Tn，直到出现10%到15%超调，但不应该有震荡；

Tn:=5ms（振荡）

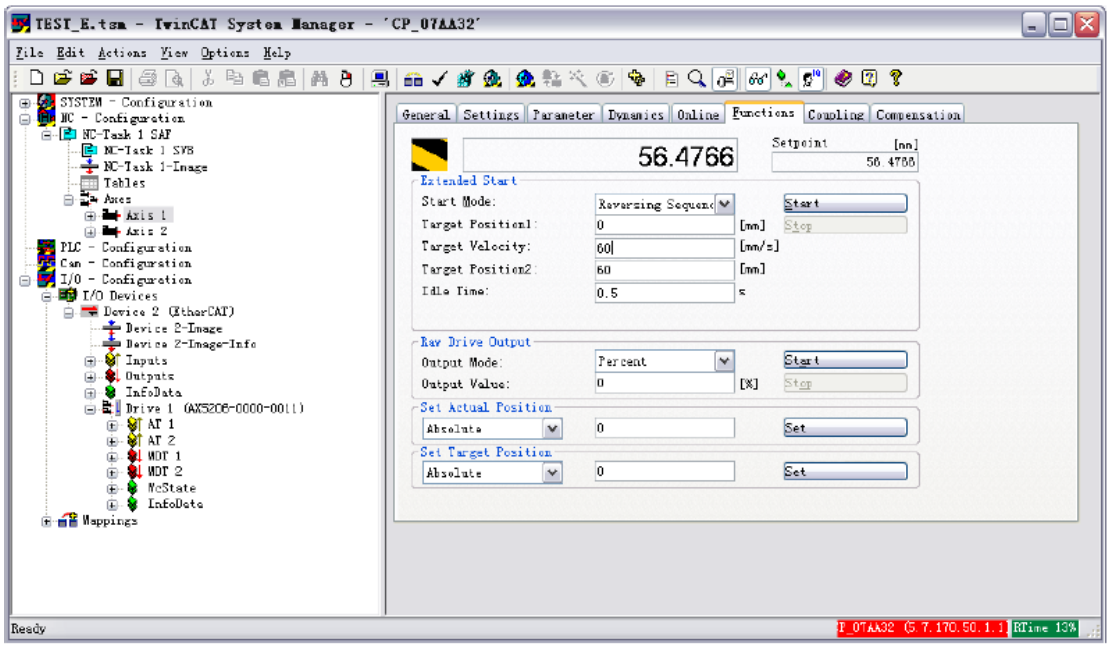


Tn:=50ms（OK）

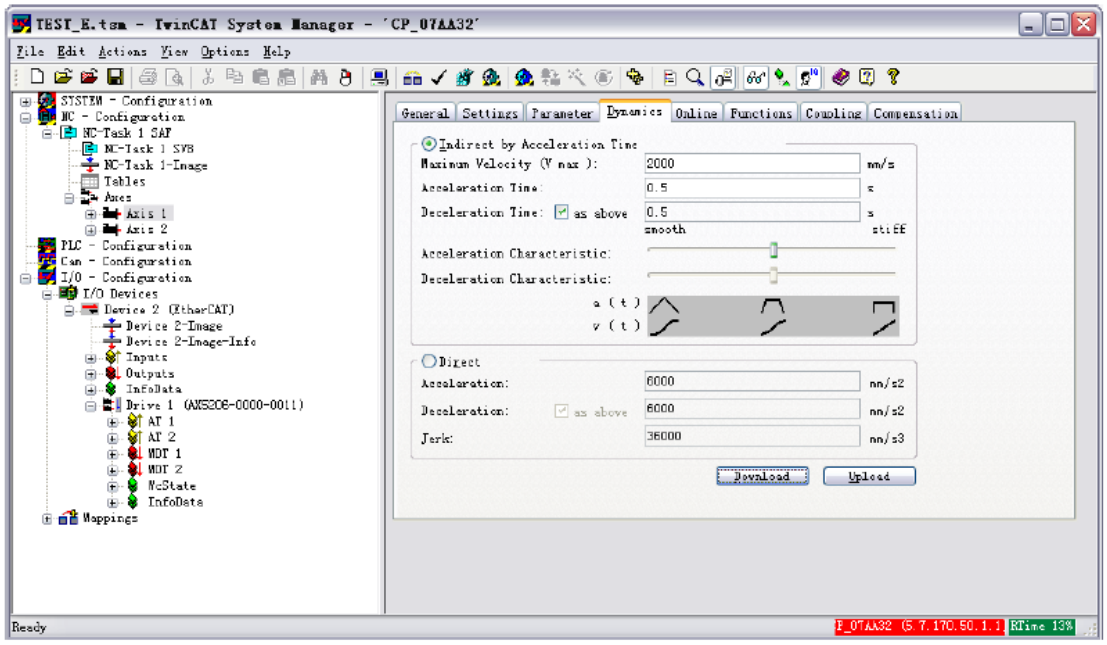


6.AX5000 的 PID 参数调节（位置环）

- 准备工作：把OP Mode设置为Poistion1 without Lag
- 在Function页面让电机作两点往复运动。

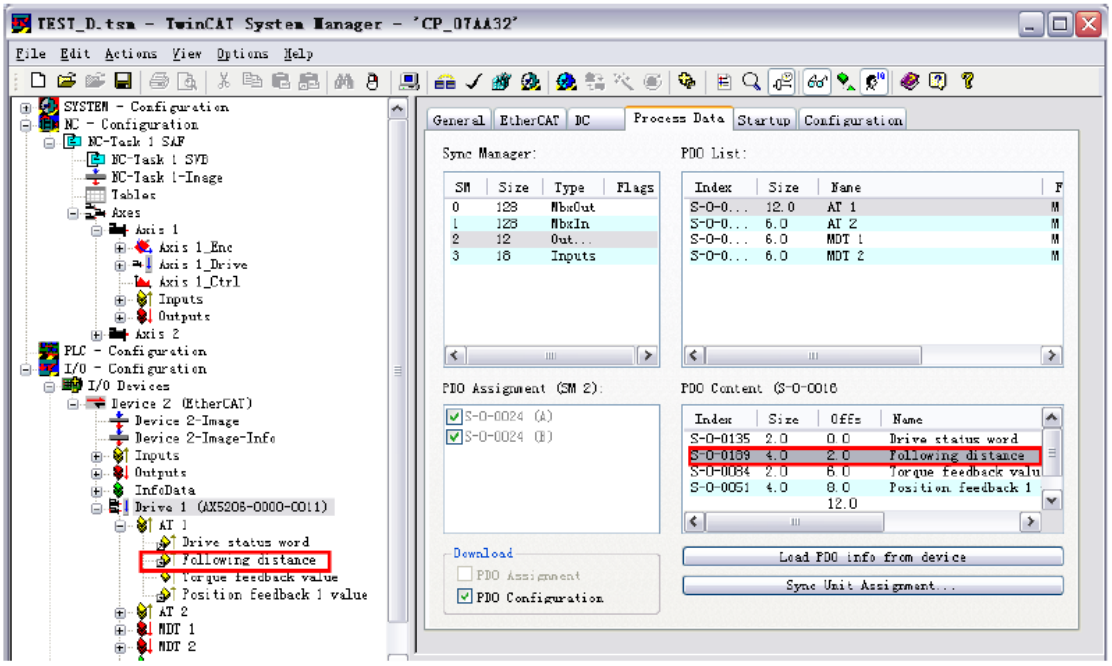


- 在Dynamics页面设置需要的最大加减速度

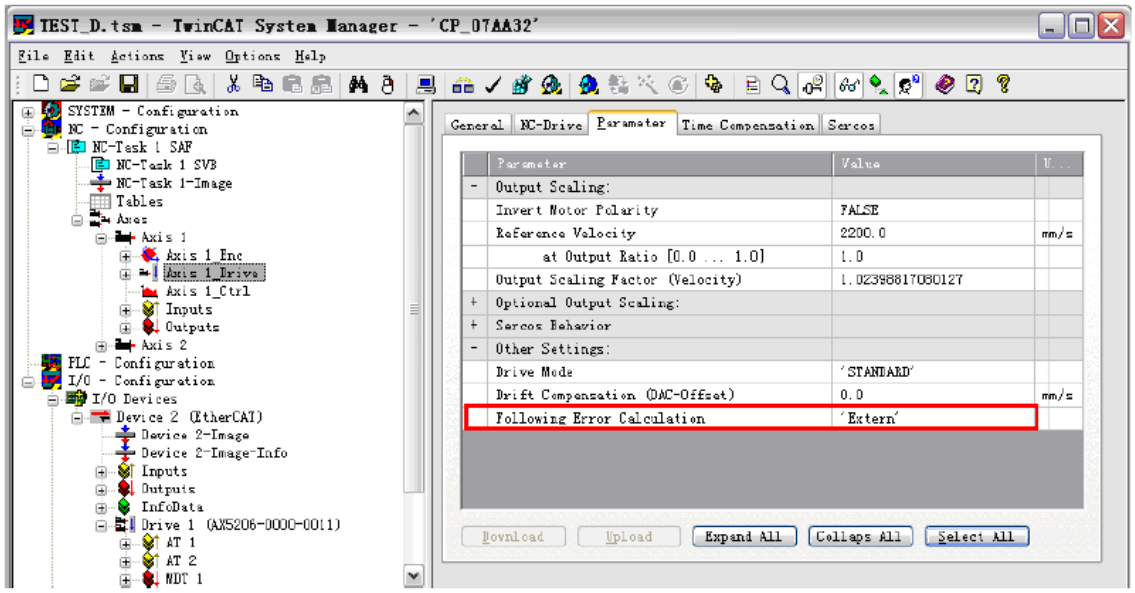


- 在Process Data中增加Following Distance

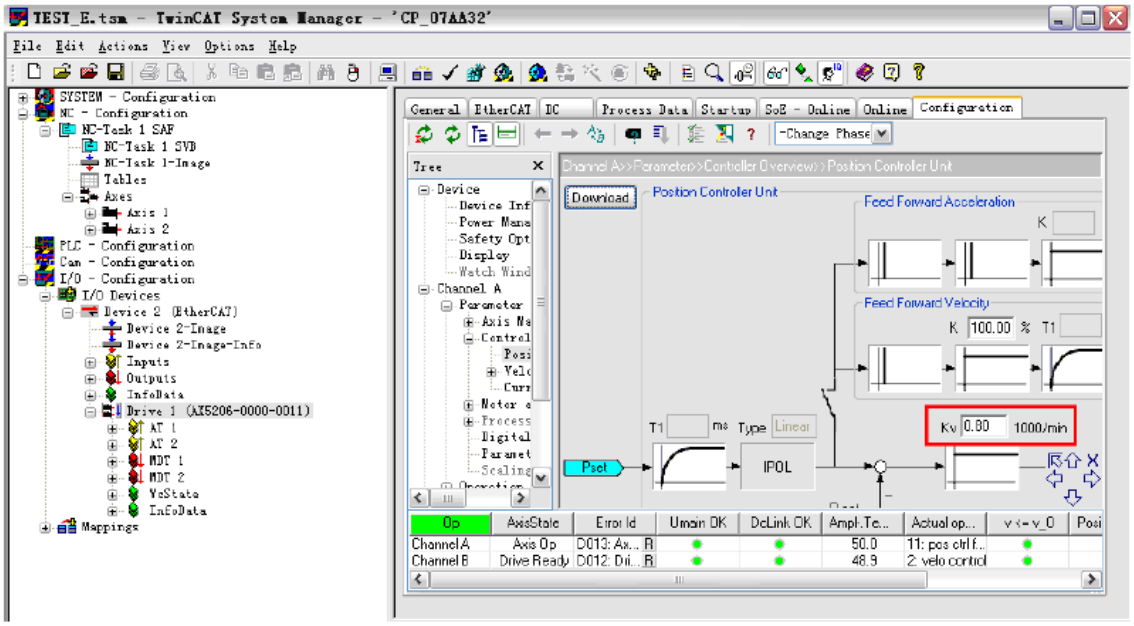
S-0-0189，Following Distance，即跟随误差。当 Process Data 中增加该变量以后，System Manager 会自动将其与 NC 轴中相应变量链接。如图所示：



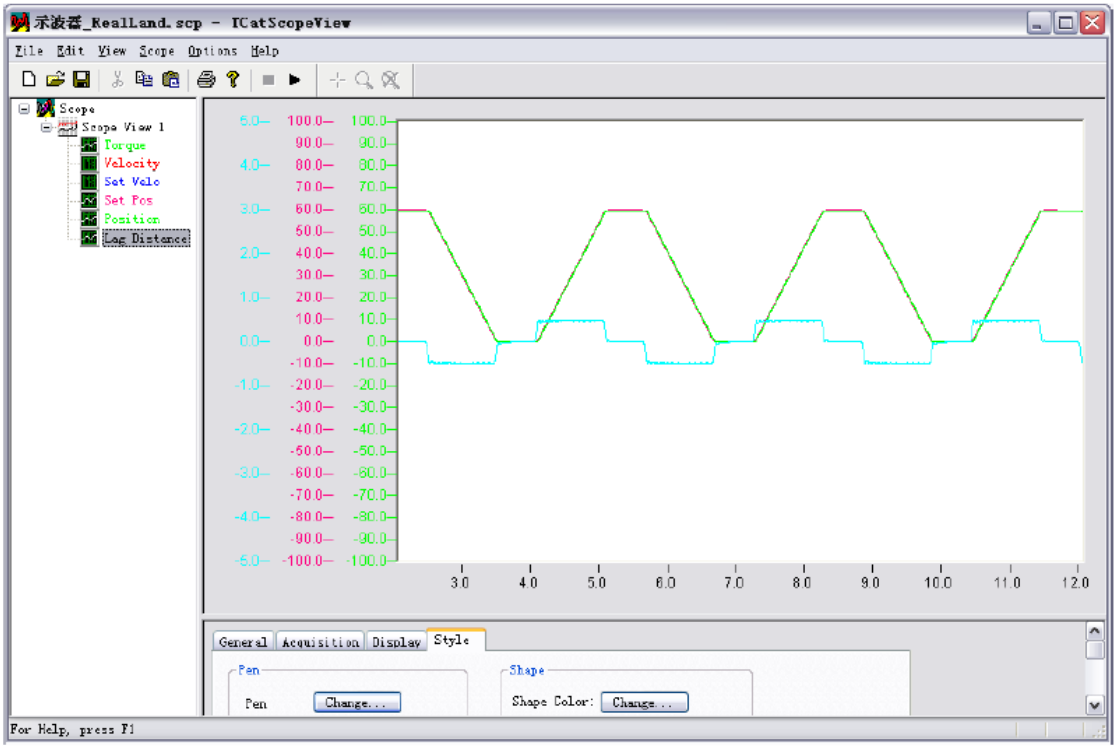
同时，NC 轴 Drive 的 Parameter 页面选项 Following Error Caculation 也自动由 Intern 变成 Extern，表示 NC 中的跟随误差来自 AX5000 位置环计算结果，因而更加准确。



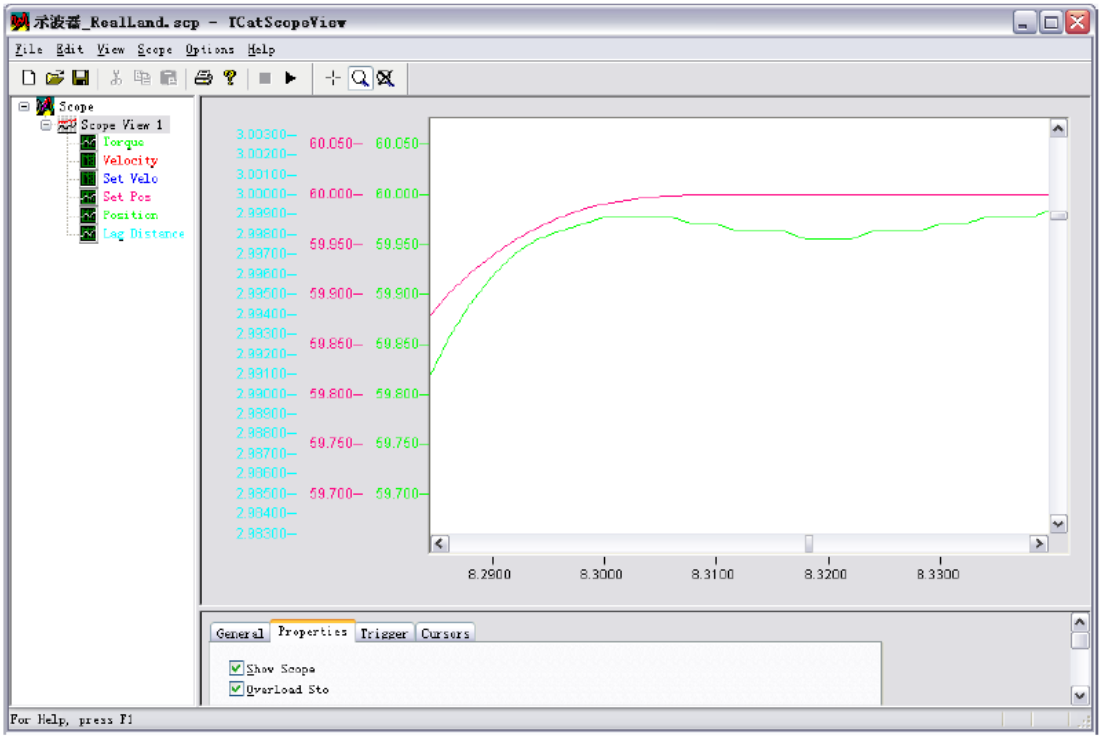
- 在AX5000位置环PID控制器调节页面，修改Kv值。



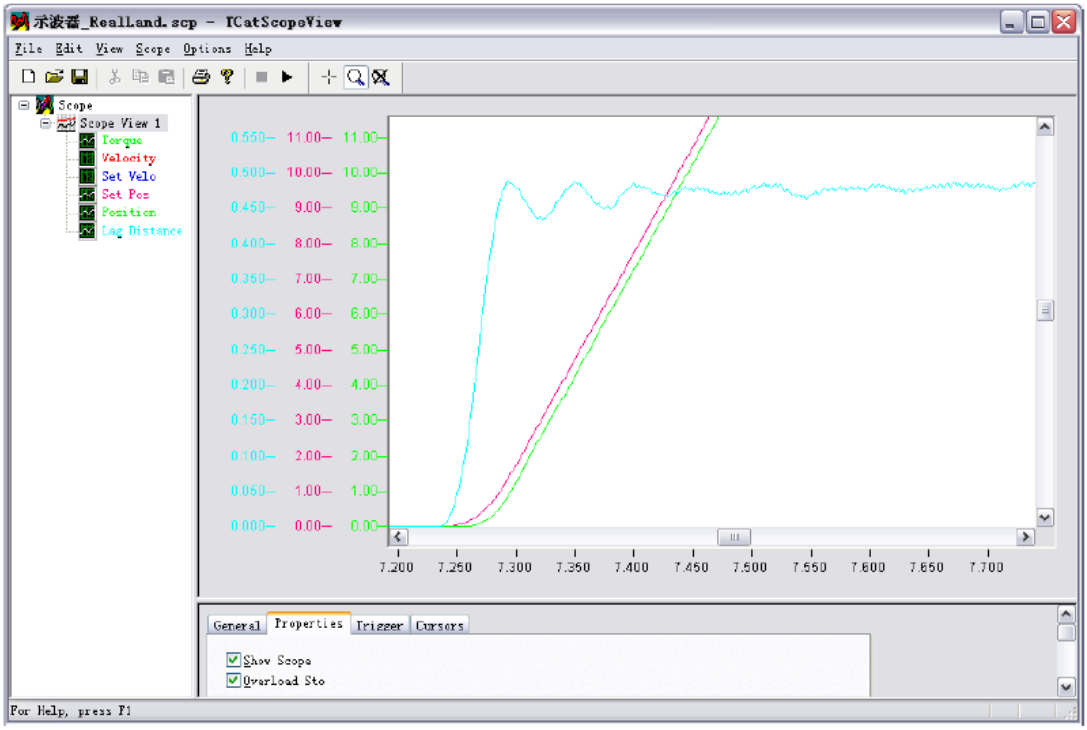
位置环只有P调节，所以只要调Kv值。注意每次修改都要点击Download按钮才生效。
缓慢增加Kv值，使设定值与反馈值尽量一致，但不应该有震荡；
例如：Kv：=5（有静差）



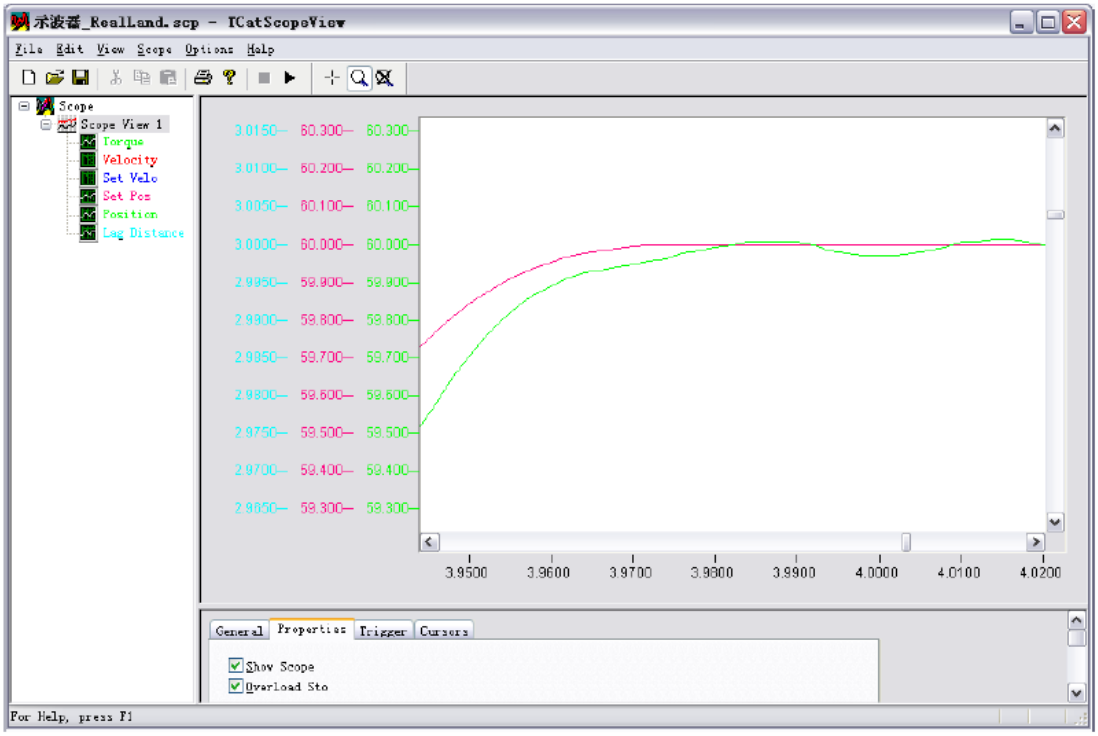
局部放大到达目标位置处的实际位置波形：有静差。



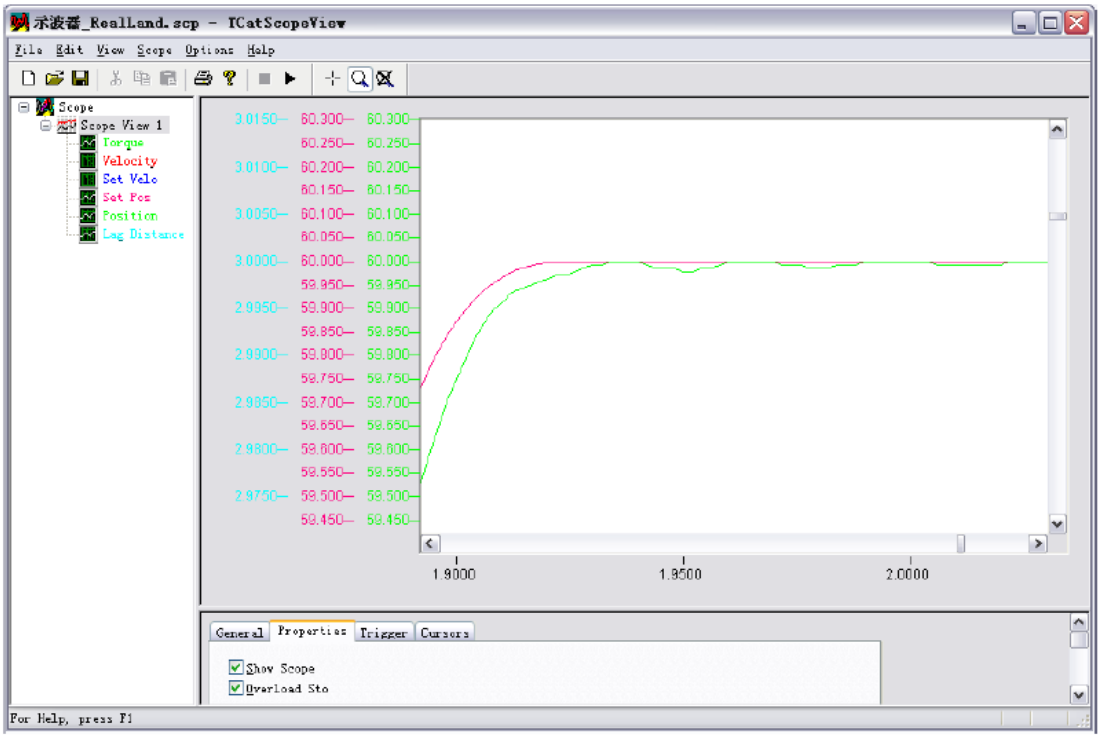
跟随误差的波形：



Kv=45（振荡）



Kv：=40，振荡临界点。最终取值 40*0.8=32。



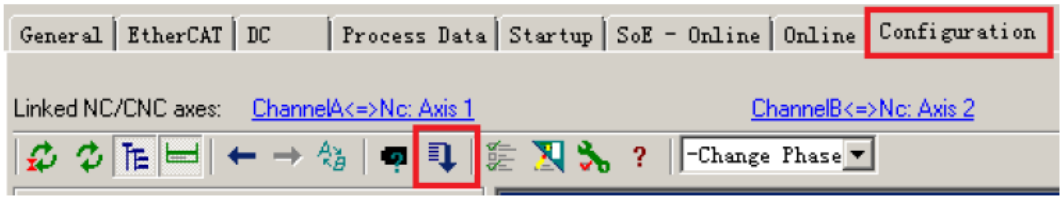
7.保存调试结果

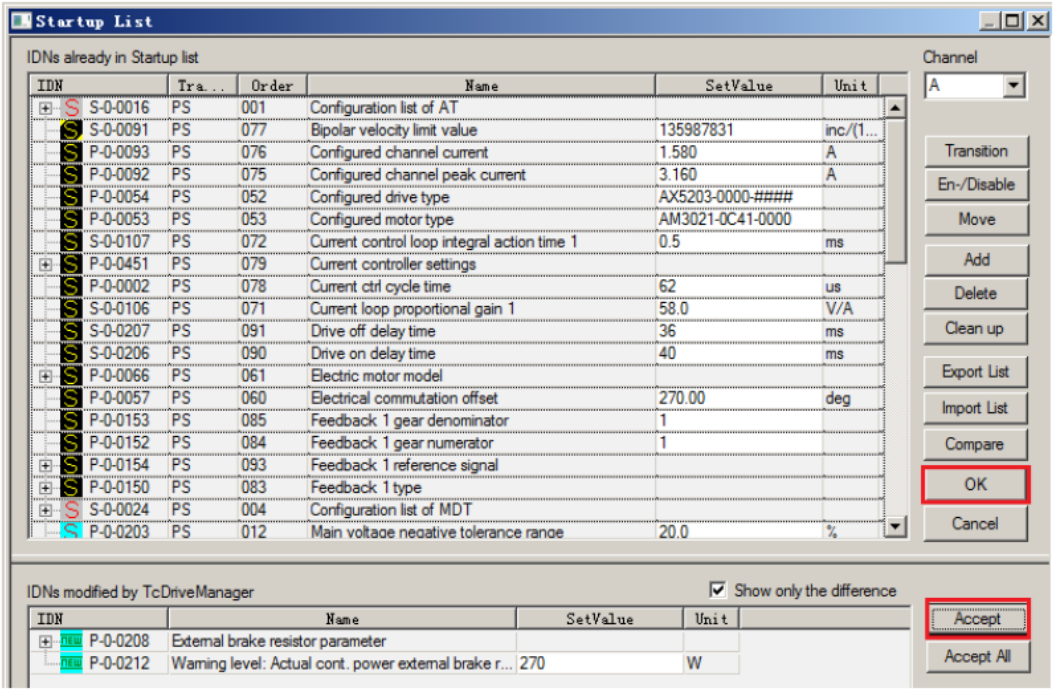
AX5000的Configuration界面做的任何参数修改，都必须经以下操作才能保持：

- 在AX5000的Configuration界面，点击Startup List按钮，点击Accept All，并点击OK；

Activate Configuration，激活配置，否则TwinCAT重启后，仍使用修改前的参数。

点击下图的按钮，进入 Startup 确认页面





在这个窗体设置的参数，直接点击 OK，激活配置就生效了。此后每次 TwinCAT 启动时这些参数就会写入驱动器。

学员提问：AX5000 驱动器更换之后，是否需要单独设置驱动器的参数？

讲师解答：AX5000 驱动器更换之后可以直接使用的，因为驱动器的参数都是通过工控机写入的，每次上电工控机都会把参数写进驱动器，因此更换一台驱动器，即使是出厂设置，上电后参数都会被修改为正常工作时的参数。

十一.　NCI 功能使用说明

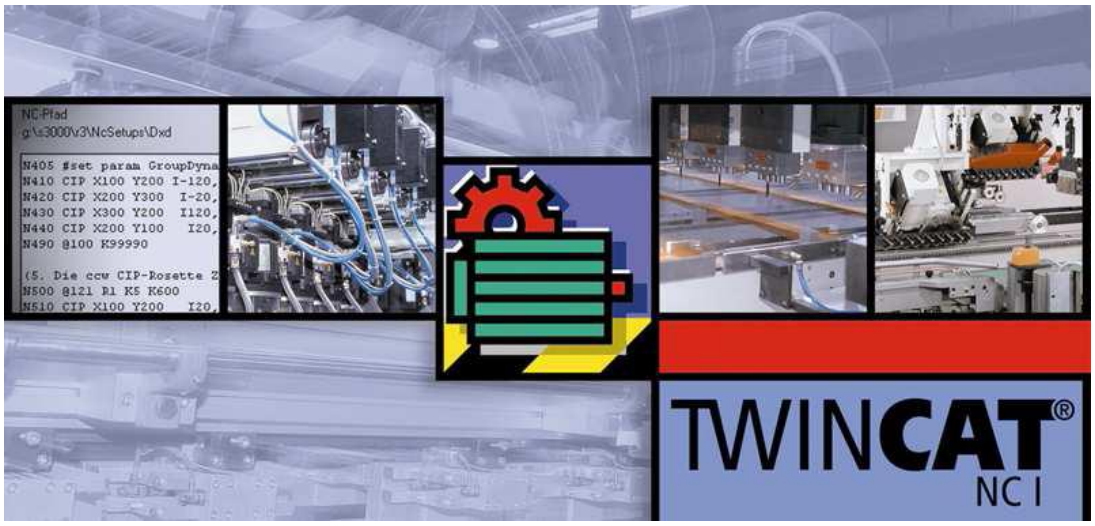
本章目标：

通过本章节的学习，学员将了解：

☑ NCI与NC PTP之间的区别

☑ NCI功能通过System Manager软件进行调试

☑ NCI功能通过PLC Control软件进行调试



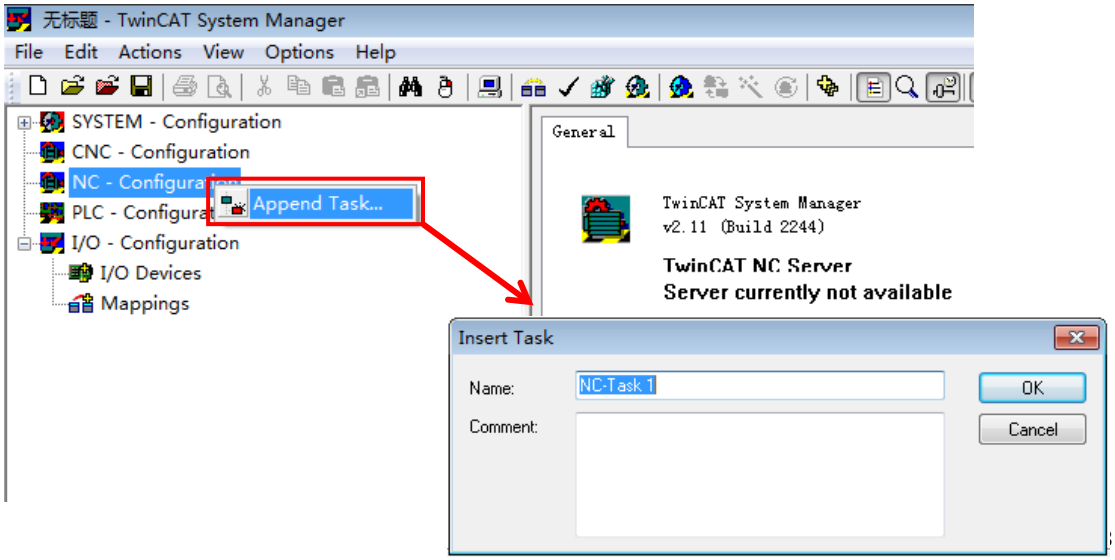
TwinCAT NCI

- 用于插补运动
- 通过符合 DIN 66025 G 代码编程运动
- 每个通道操作 3 跟插补轴（和 5 个辅助轴），最多 31 个通道

1. System Manager 中 NCI 的调试

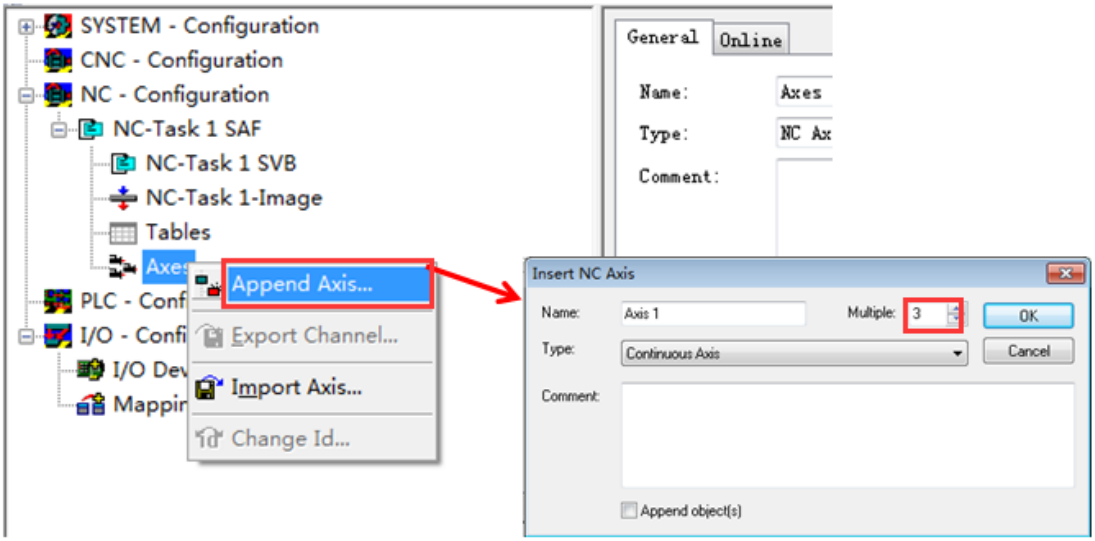
NCI 的配置方式和 NC PTP 有比较大的区别，NC PTP 可以直接通过 axis 的 online 窗口对物理轴进行调试，而 NCI 则必须通过调用 G 代码才可以让电机正反转，下面以虚轴为例，介绍一下如何在 System manager 软件中对 NCI 功能进行调试。

(一) 首先新建一个 NC 任务

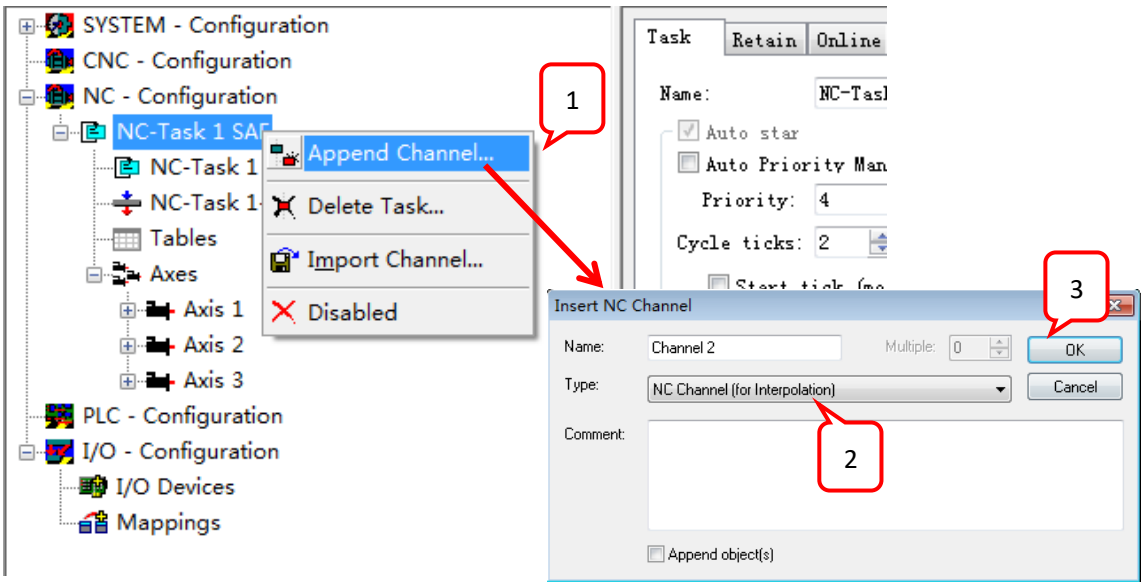


由于本次采用的是虚轴模拟的，所以需要先建三根虚轴：

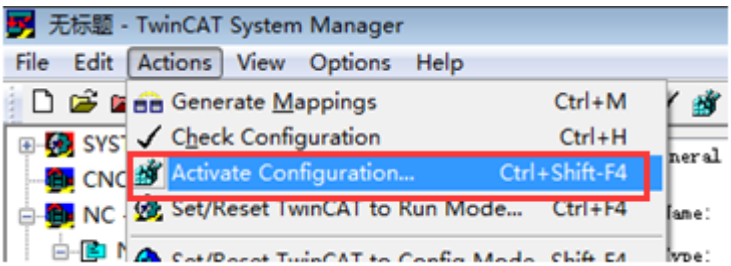
(1) 添加三根 NC 轴，直接在 Multiple 里面设置 3,可以一下子添加 3 根轴。



(2) 右键 NC_Task1 SAF，手动添加一个 NCI 的通道。

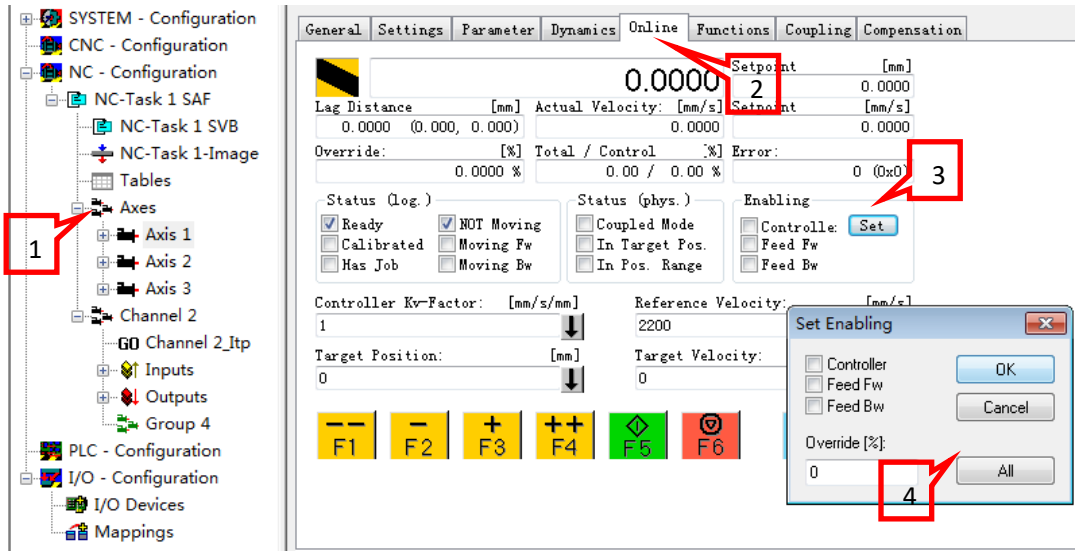


(3) 激活配置使前面添加的虚轴以及 NC 插补通道生效，并切换 TwinCAT 至运行模式。

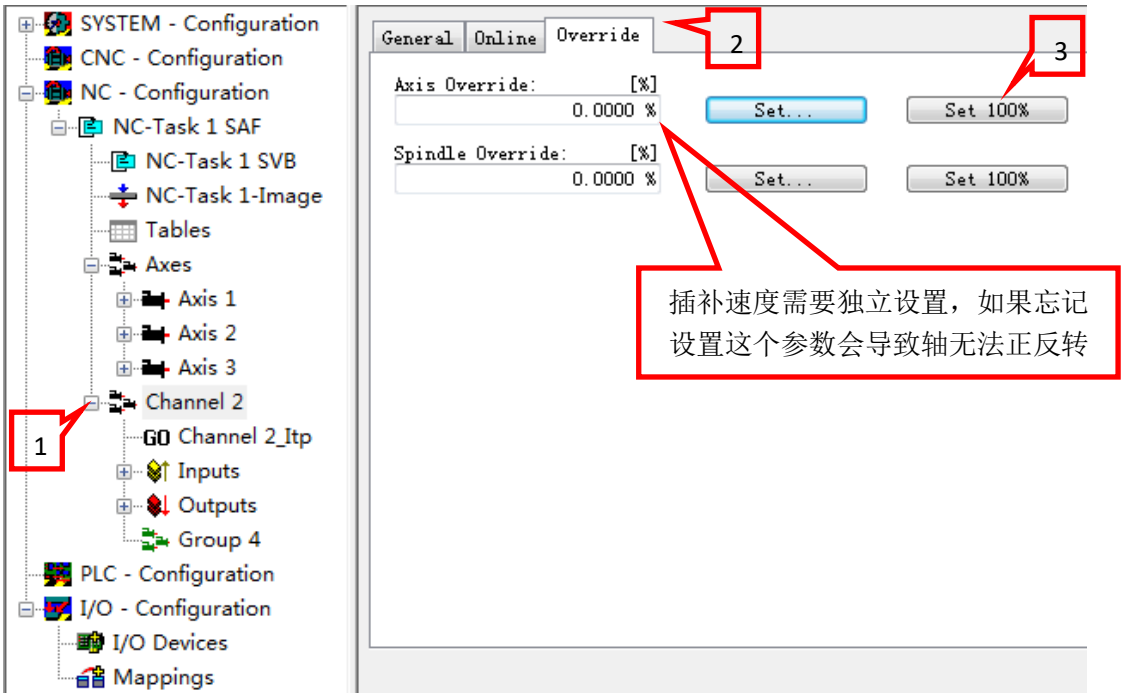


(二) 完成以上上述操作，就可以来尝试对轴进行初步调试工作：

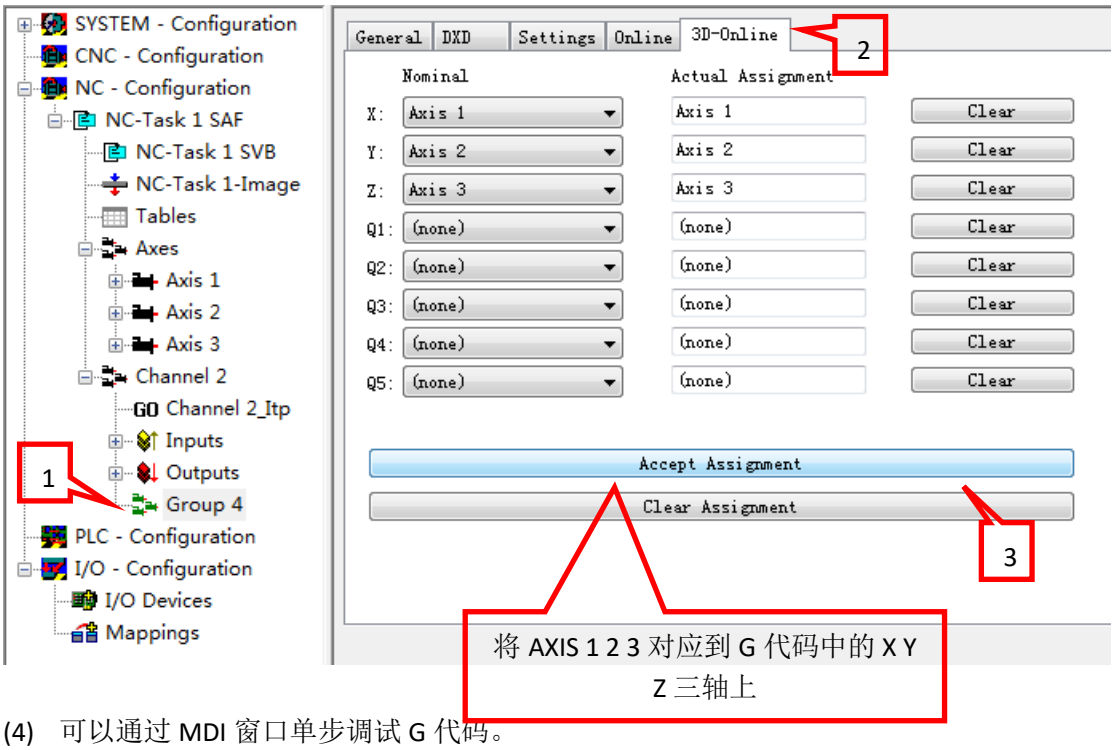
(1) 将三个虚轴使能，先选中 Axis 1，点击 Online 选项卡，点击 Set，点击 All 对轴进行使能，依次对 Axis2, Axis3 都使能上。



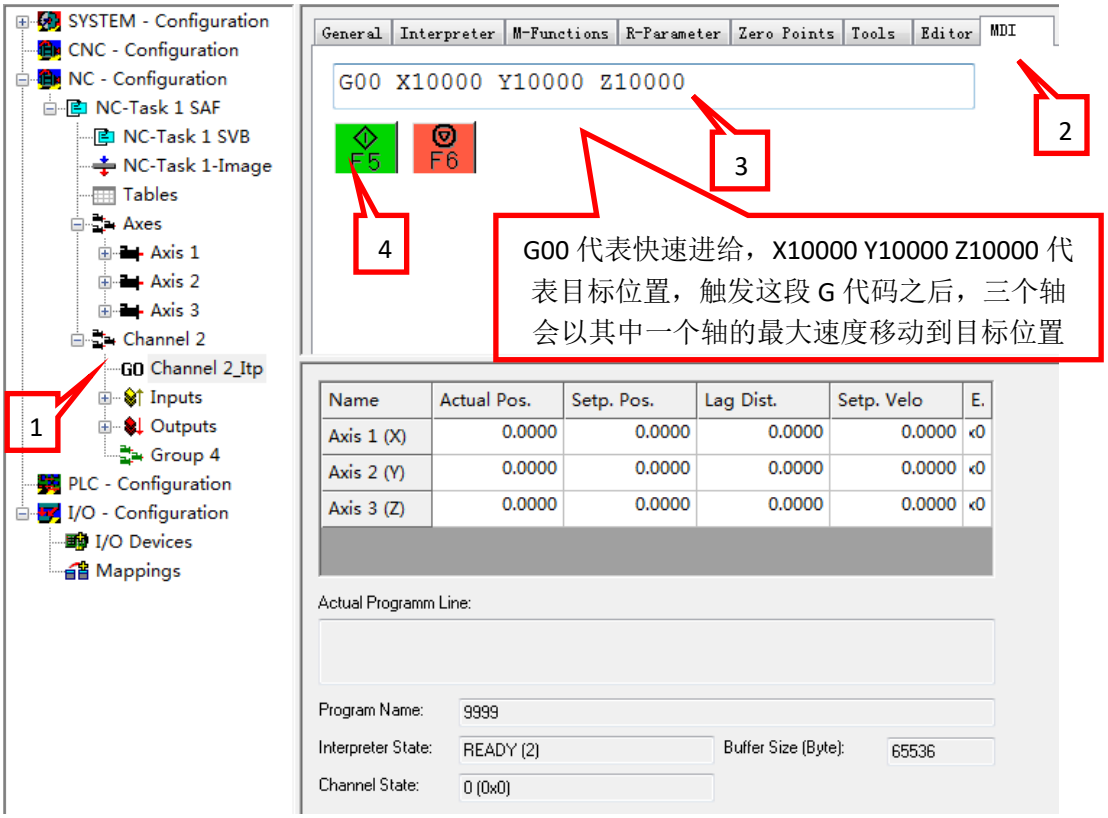
(2) 设置插补的速度比为 100%。



(3) 建立 G 代码中 XYZ 轴和 NC 轴的对应关系。



(4) 可以通过 MDI 窗口单步调试 G 代码。



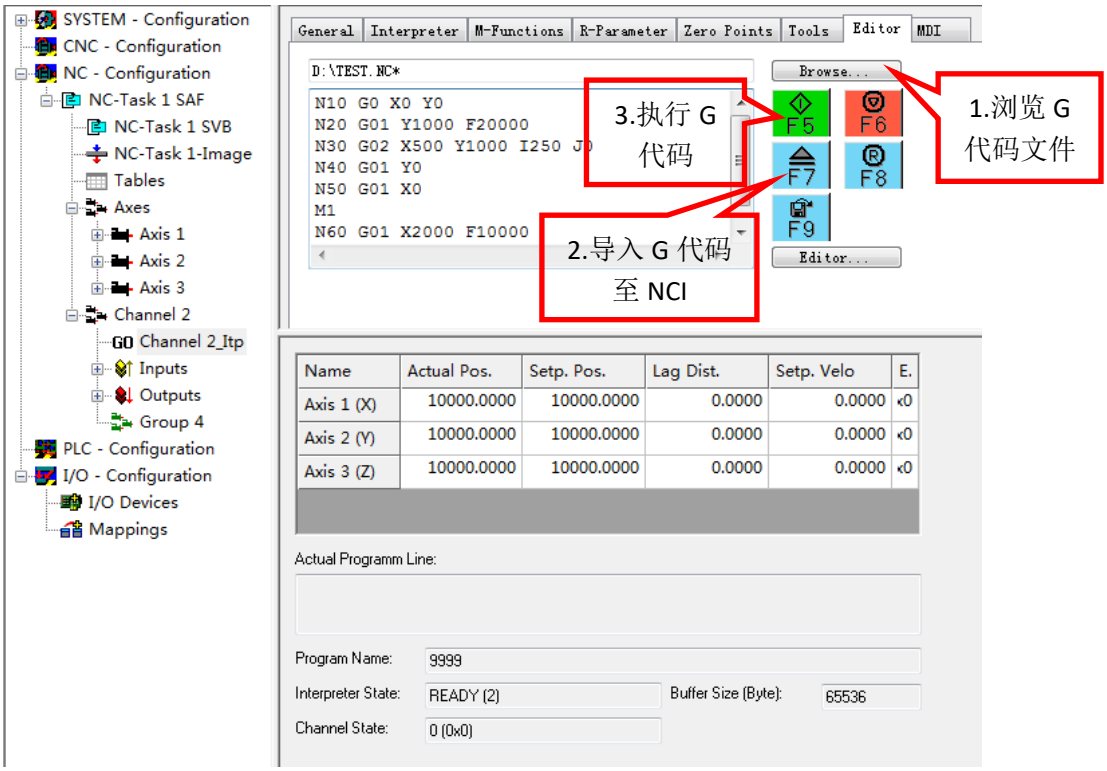
(5) Editor 窗口中可以对完整的 G 代码文件进行调试。

F5	执行 G 代码
F6	停止 G 代码
F7	加载 G 代码
F8	复位错误
F9	保存 G 代码至文件

调试步骤如下：

① 点击 Browse 选择 G 代码文件，按 F7 将 G 代码文件导入到 NCI 中准备执行，可以在 Program Name 中看到当前已经导入的 G 代码文件；

② 按 F5 执行 G 代码，此时可以看到 Actual Program line 中当前正在执行的 G 代码行，XYZ 轴会根据 G 代码的内容执行相应的动作。



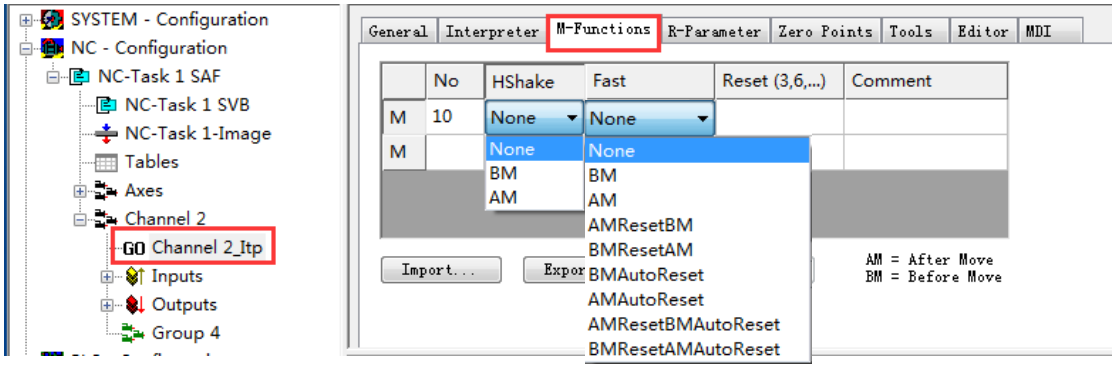
(三) M 指令的使用

在 G 代码程序中，还会用到 M 指令。M 指令起到 G 代码与 PLC 程序交互的作用，比如 G 代码执行过程中，需要 PLC 里面做一些处理，比如换刀，吹气等操作，可以通过 M 代码来完成，除了部分根据国际标准已经指定好固定用途的 M 指令，根据 M 指令是否打断 G 代码预读，可以被分为握手型 M 指令 Hshake 和快速 M 指令 Fast。握手型 M 指令需要 NCI 与 PLC 握手，M 指令在 NCI 通道中被触发，在 PLC 程序确认这个 M 指令之后，才能继续执行后面的 G 指令。而快速 M 指令不需要 PLC 程序去确认，只是起到通知 PLC 的作用。

M 功能	含义
0..159	可以任意定义的 M 功能（除了 2，17，30）
2	程序结束
17	子程序结束
30	程序结束并删除所有快速信号位方式的 M 功能

(1) M 指令的定义

路径在：Channel 2_Itp 下的 M-Functions：



其中：

No：M 指令的编号，最多可设置 159 个

Hshake：用来设置握手方式的，AM 表示 After Motion，即当 M 指令和 G 代码同行时，M 代码会在 G 代码执行完毕后被激活；BM 表示 Before Motion，即当 M 指令和 G 代码同行时，M 代码会在 G 代码还未被执行之前被激活；None 不需要 PLC 握手确认。

Fast：用于定义是否为 Fast 类型。其中有 autoreset 的代表前后动作指令复位该 M 指令，reset 则是用另一个 M 指令来复位这个指令，如果直接使用 AM 或者 BM，M 指令触发之后会一直保持为 true，复位需要使用 PLC 中的功能块 ItpresetMFuncEx，类似于 Hshake。

(四) R 参数的使用

G 代码中给定动作参数时可以使用固定的值，例如：X100 Y100 F1000，也可以使用 Lreal 类型的变量替代常量增加灵活性，NCI 中称为 R 参数。

如 G01 X100 Y200

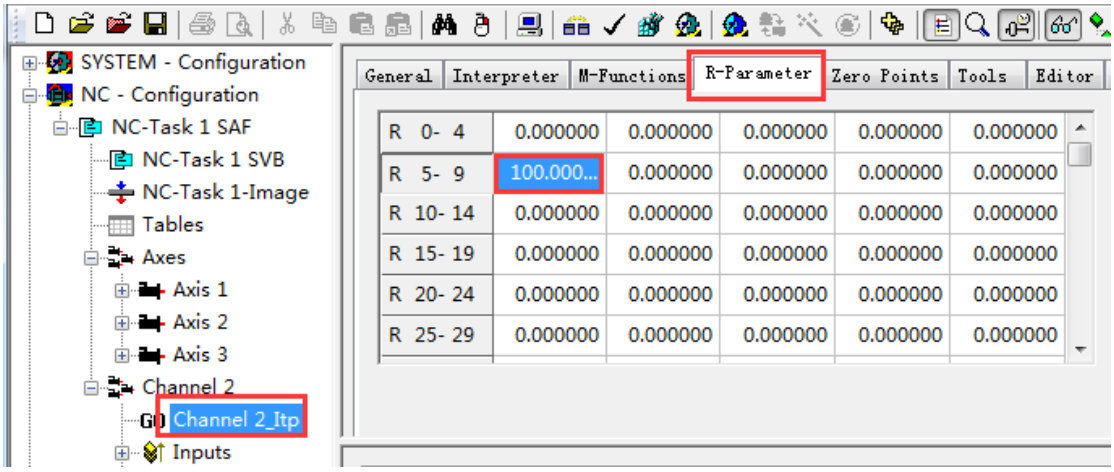
可改成：R5=100

G01 X=R5 Y=2*R5

其中每个 NCI 通道可设置 1000 个 R 参数，从 R0 到 R999，类型均为 Lreal 型，要注意的是 NCI 会对 G 代码进行预读，因此 R 参数需要提前修改。

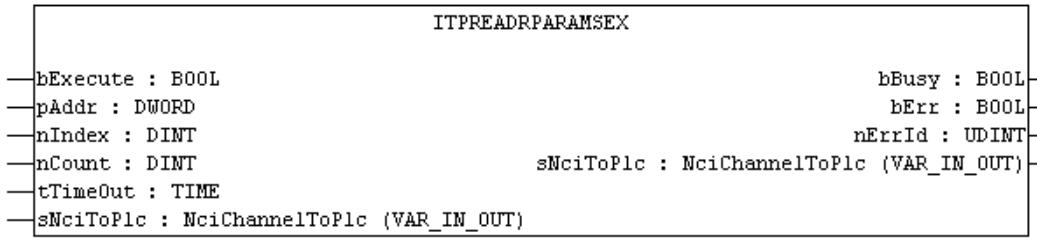
R 参数有三种访问方式：

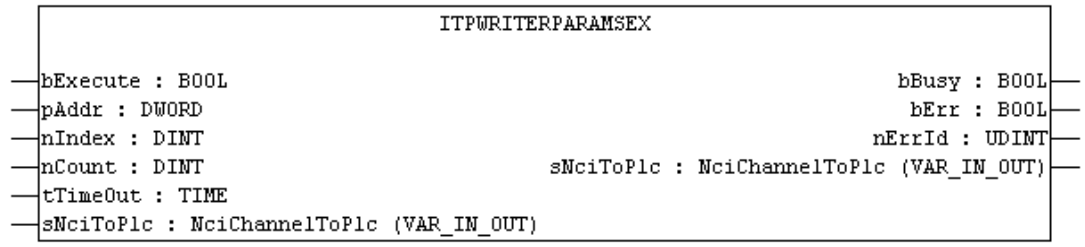
①System Manager 中进行访问，如下图：



②由 PLC 程序中进行访问，如下图：

采用功能块，ItpReadRParamsEx（读），ItpWriteRParamsEx（写）





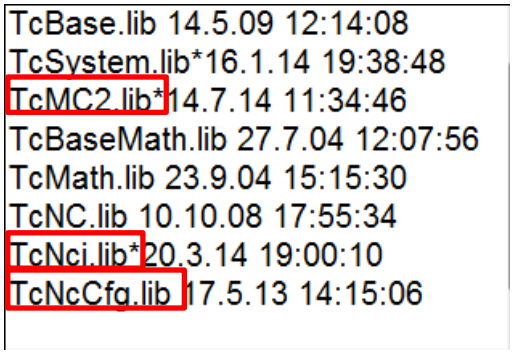
③G 代码的方式进行访问：

赋值　N10 R5=100

调用　G01 X=R5 Y=2*R5

2. Plc Control 编程配合 NCI 的调试

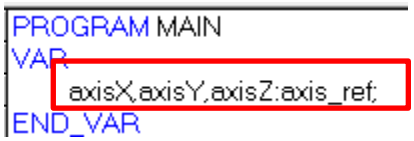
如果通过程序来做 NCI 控制，需要添加三个库文件：①TcMC2.lib ②TcNci.lib ③TcNcCfg.lib 下图所示的库文件，程序的作用主要是配合 G 代码，并不直接控制轴移动。



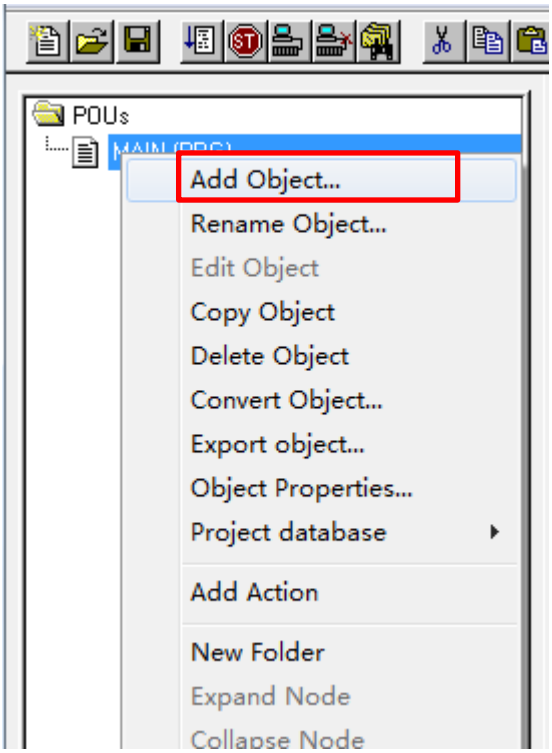
该介绍内容配合例程进行讲解：

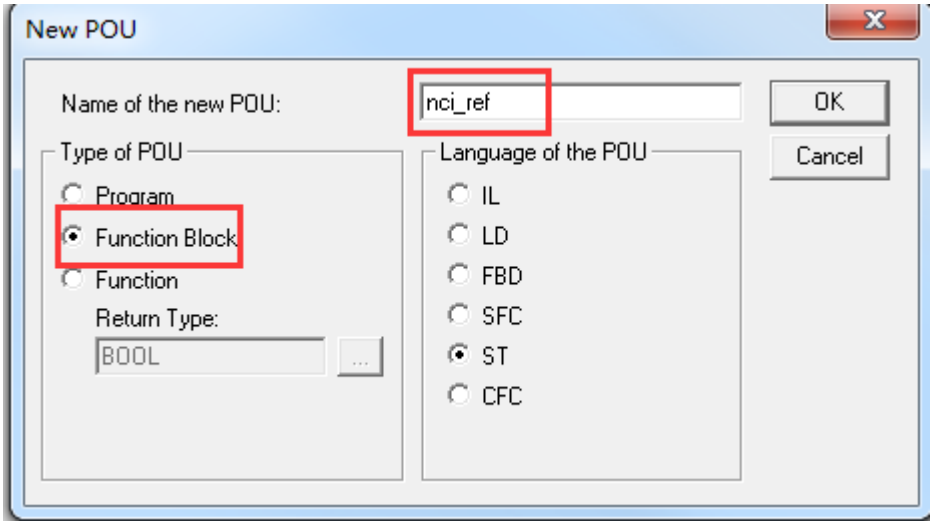
首先介绍基本所需用的框架程序：

(1)　在主程序区定义轴变量

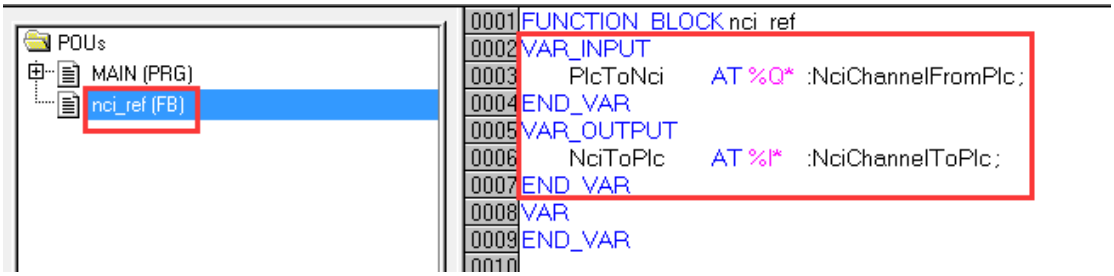


(2)　在对象管理窗口右击创建一个名字为“nci_ref”的 Function Block

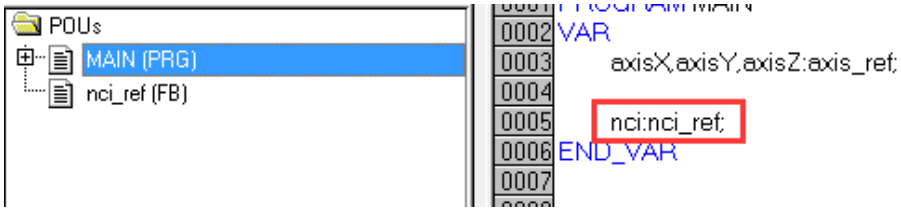




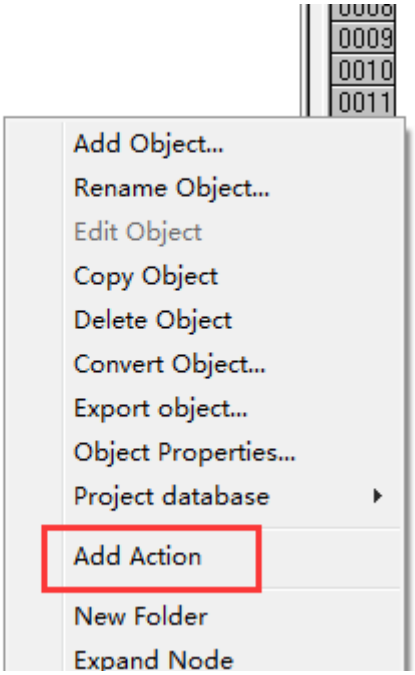
在该 FB 编程区定义如下：（该方式为了后续编程方便，仿照轴变量的定义方式）

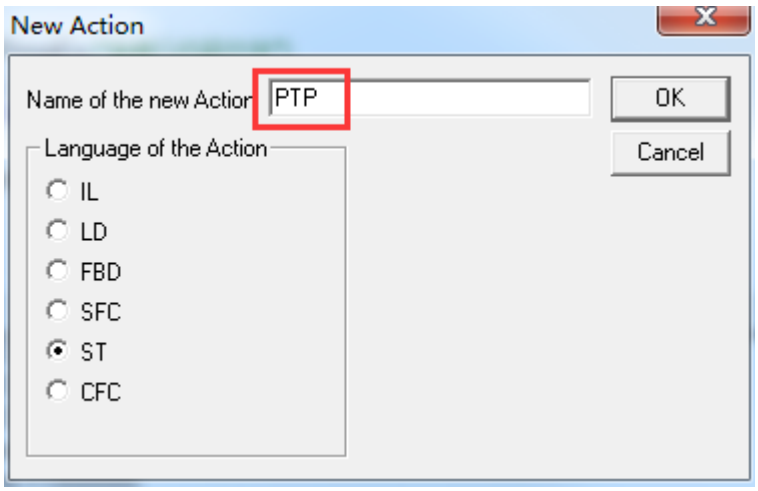


并在主程序区定义一个功能块，功能块类型选择刚才定义的 nci_ref。

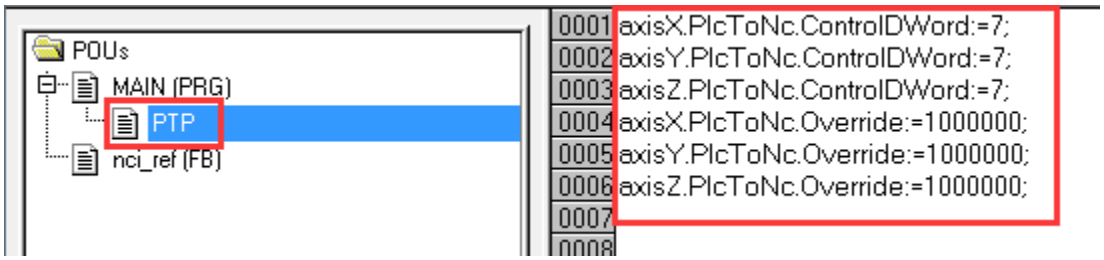


(3) 在对象管理窗口右击创建一个名字为“PTP”的 action，该 Action 是用于 X,Y,Z 三轴进行使能，和设置速比。



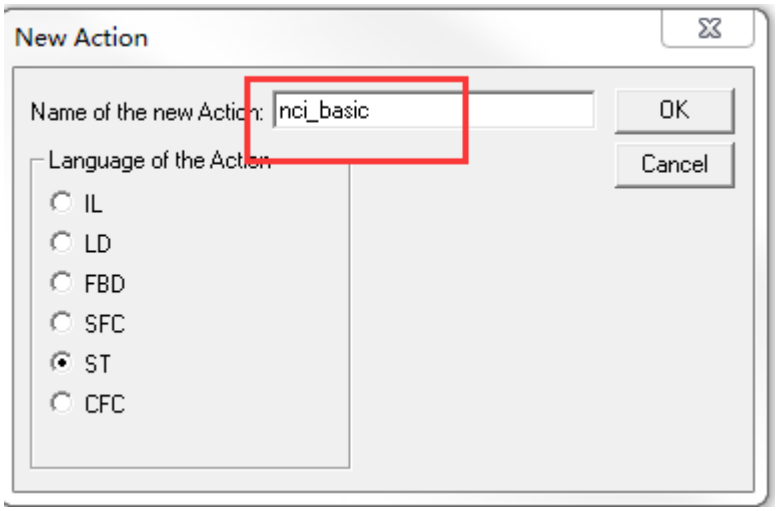


在该 action 编程区定义如下：



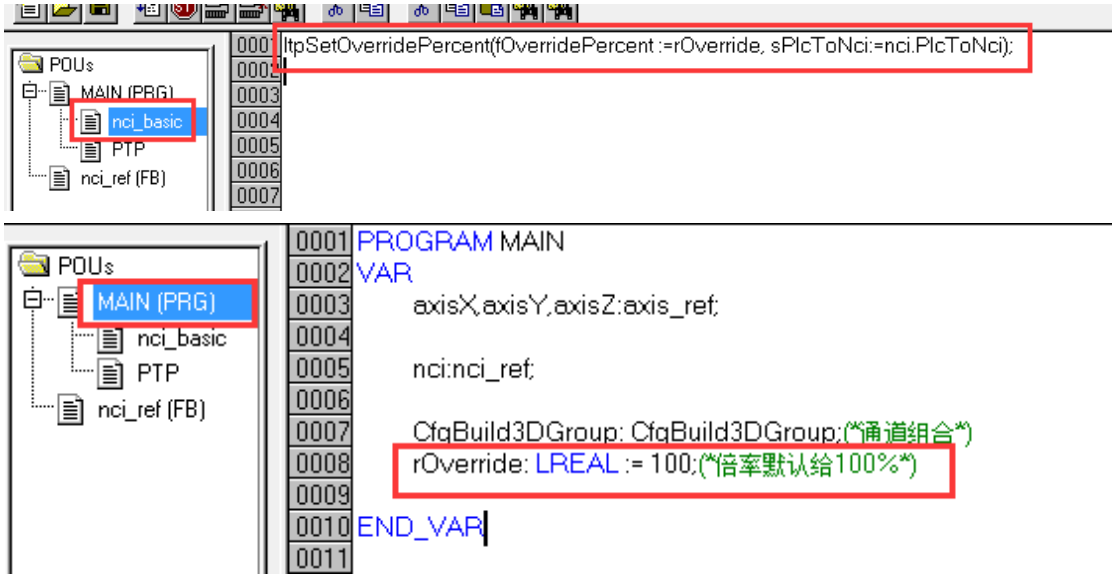
(4) 在对象管理窗口右击创建一个名字为“PTP”的 action，该 Action 是用于 X,Y,Z 三轴进行使能，和设置速比。变量、对三轴使能，并设置速度比

(5) 同（3）的方法类似，再创建一个 action，命名为 nci_basci，

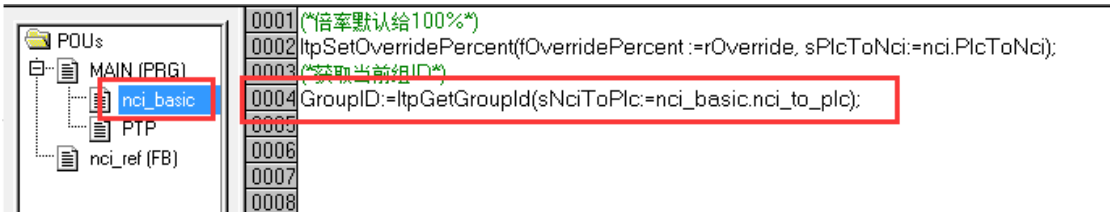


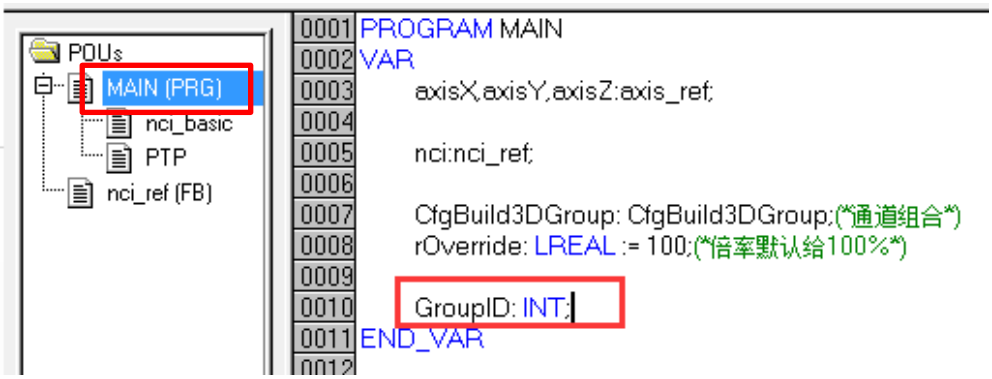
(6) 接下来在 nci_basic 区域定义一些基础的 NCI 功能，如通道组合、装载 G 代码文件等

1) 设置插补通道的倍率，调用 FUN：ItpSetOverridePercent。且 rOverride 定义为 Lreal 类型，赋初值 100；

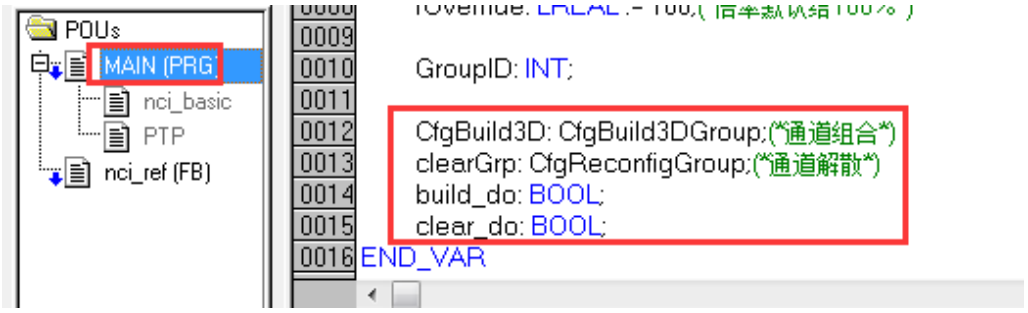


2) 获取当前 system manager 的 3D 组别 ID 号，调用 FUN：ItpGetGroupId，并将其得到的 INT 型数值赋给 GroupID。定义如下；

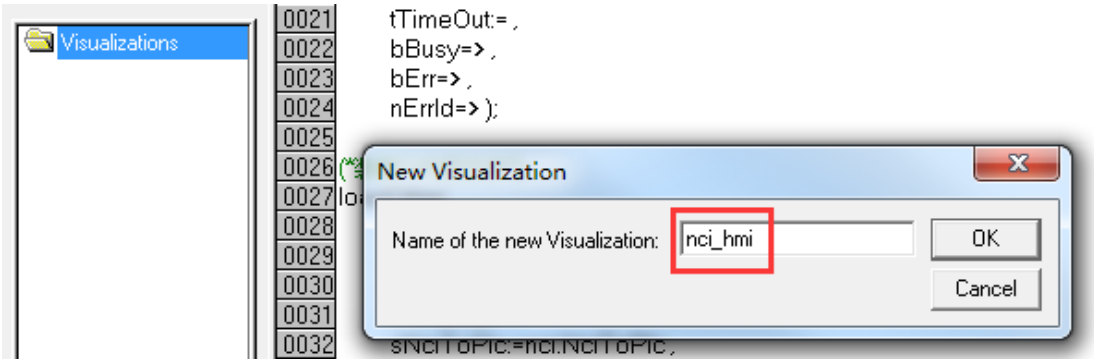




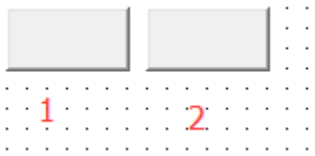
3) 定义通道组合功能块 CfgBuild3DGroup 和通道解散功能块 CfgReconfigGroup，并且对其编程。



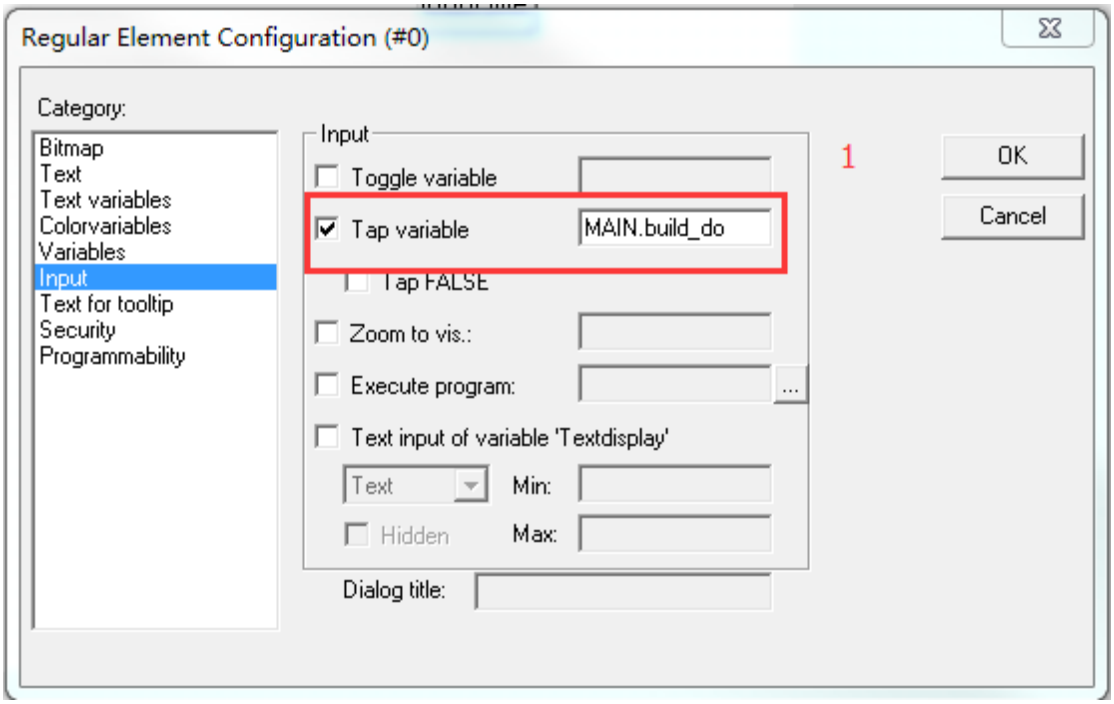
完成上述编程，创建一个名字为：nci_hmi 的人机界面

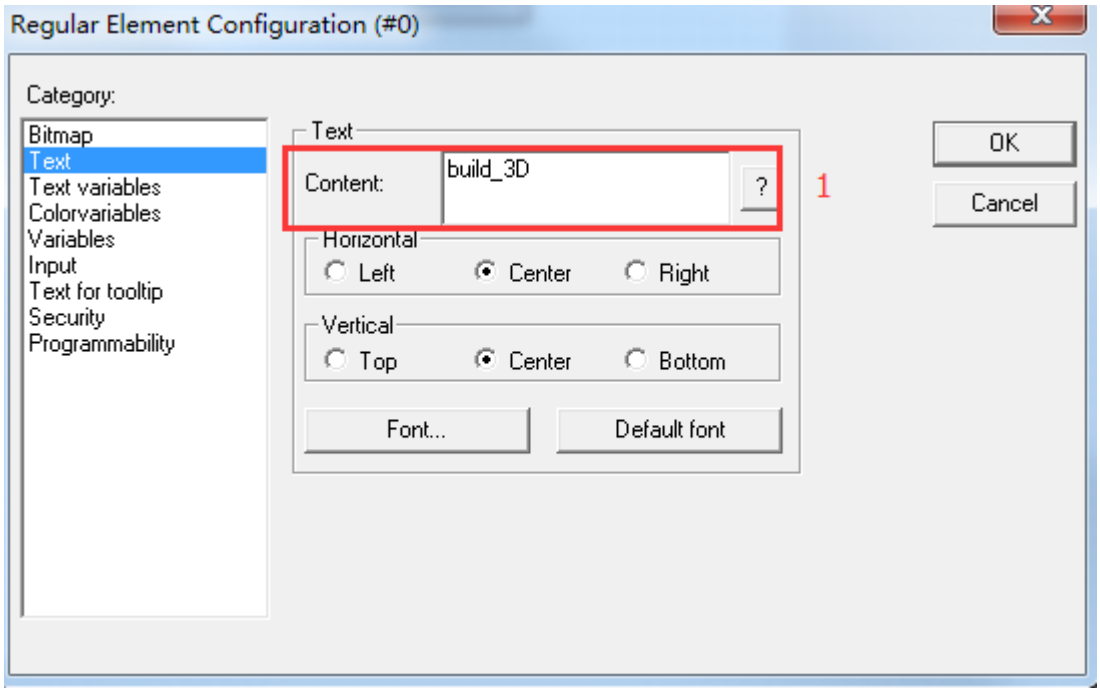


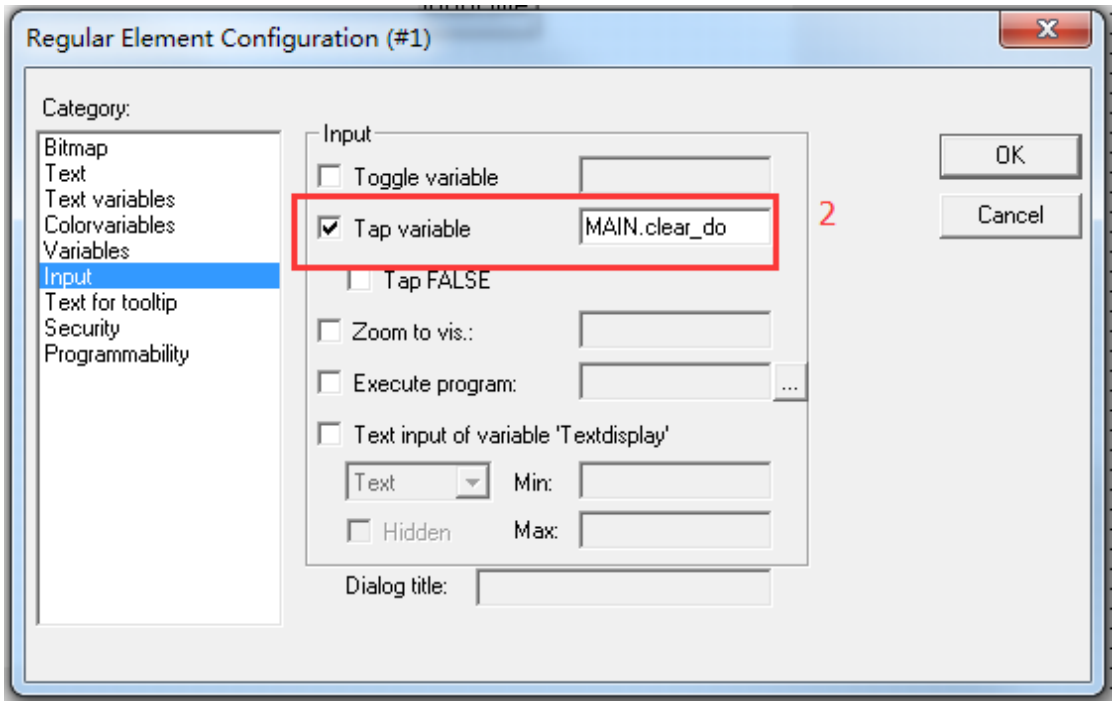
并且创建两个按钮  ，分别是：

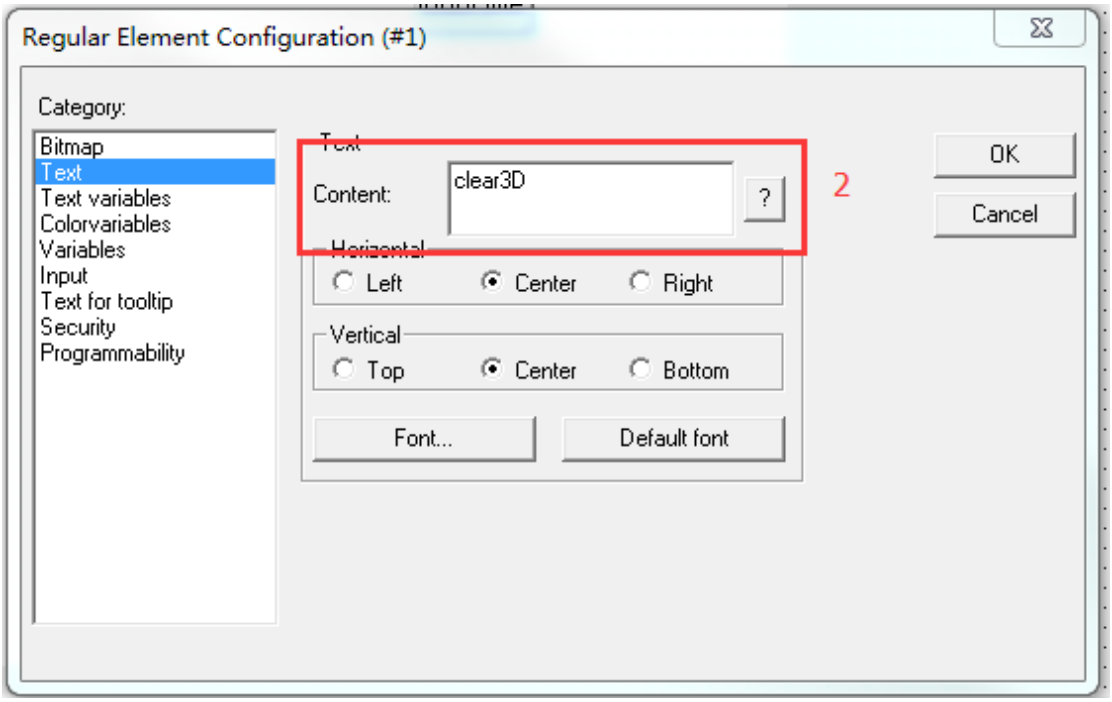


12 两个对应的按钮属性，设置如下：

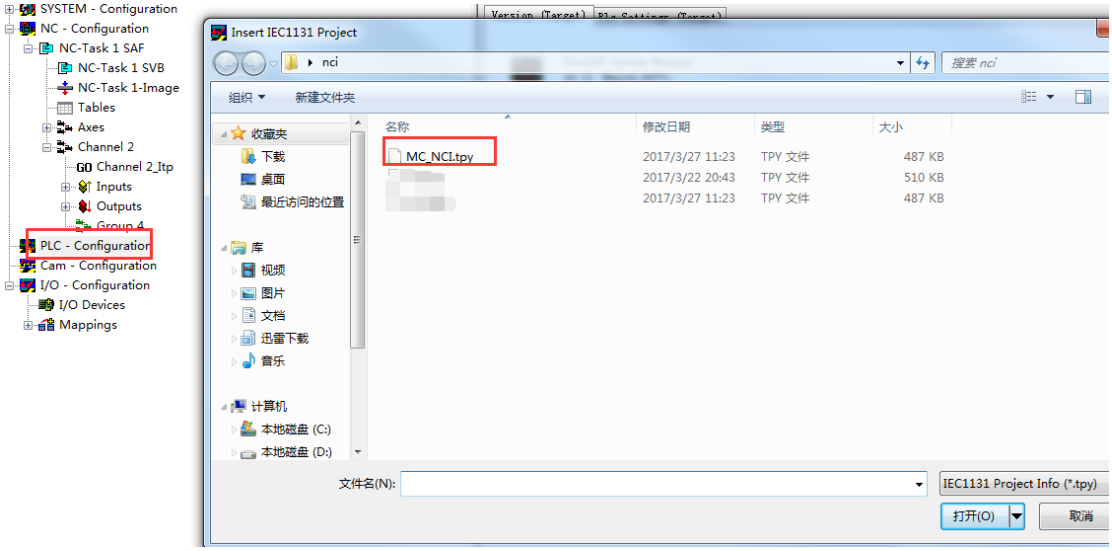



Regular Element Configuration (#0)
Category:
Bitmap
Text
Text variables
Colorvariables
Variables
Input
Text for tooltip
Security
Programmability
Text
Content:
build_3D
?
1
Horizontal
Left
Center
Right
Vertical
Top
Center
Bottom
Font...
Default font
OK
Cancel

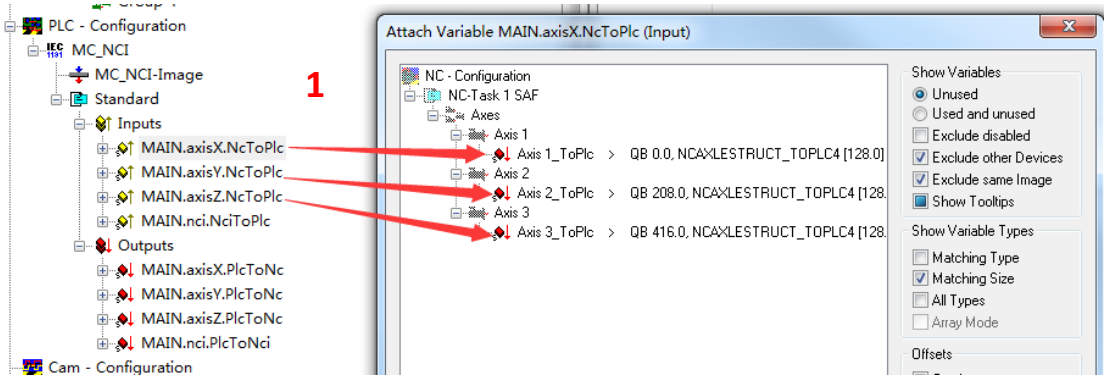

Regular Element Configuration (#1)
Category:
Bitmap
Text
Text variables
Colorvariables
Variables
Input
Text for tooltip
Security
Programmability
Input
Toggle variable
Tap variable
MAIN.clear_do
2
Tap FALSE
Zoom to vis.:
Execute program:
...
Text input of variable 'Textdisplay'
Text
Min:
Hidden
Max:
Dialog title:
OK
Cancel

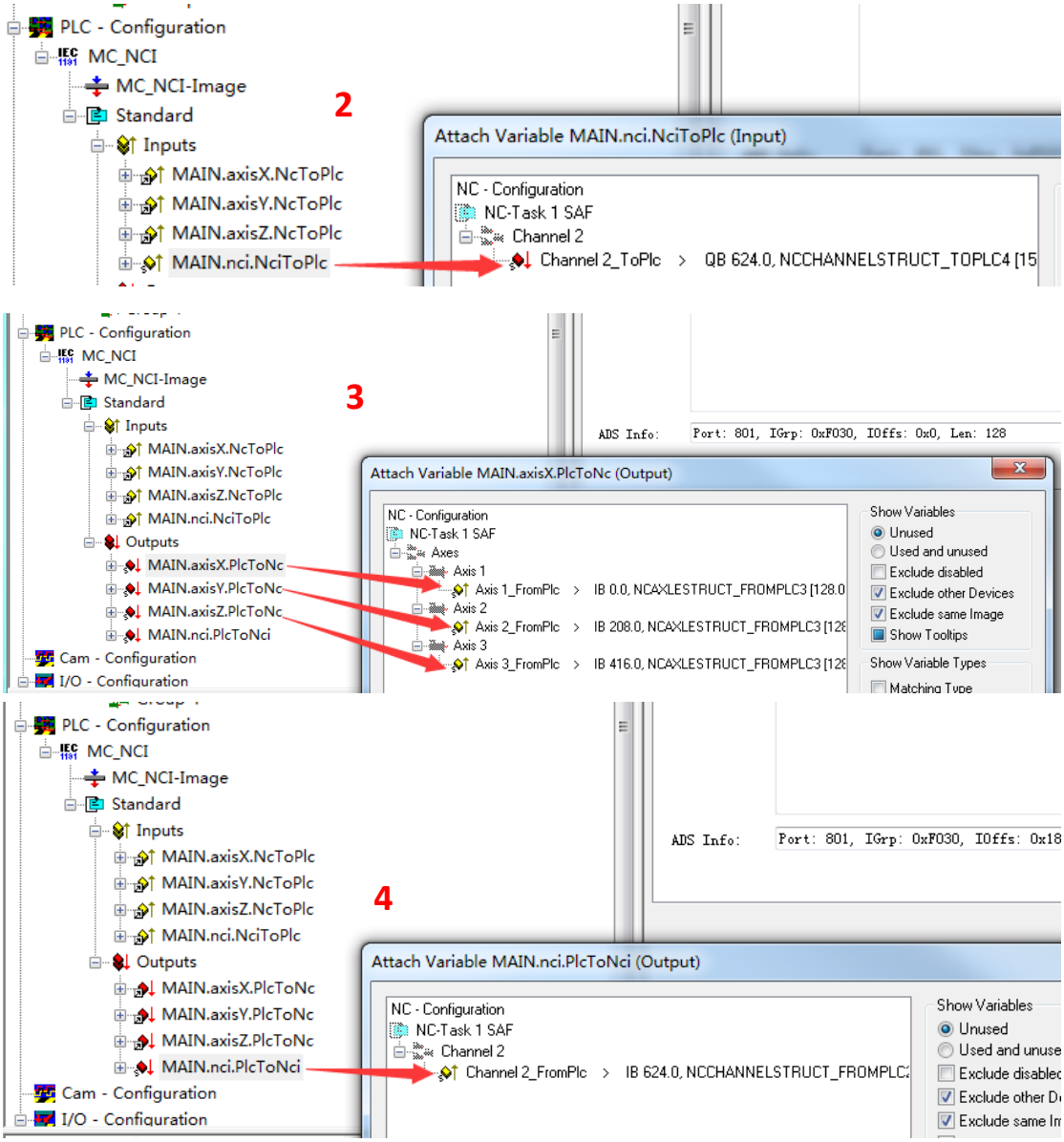


对程序编译，通过后生产 TPY 文件，然后在 System manager 中 PLC-Configuration 下添加该程序，如下：



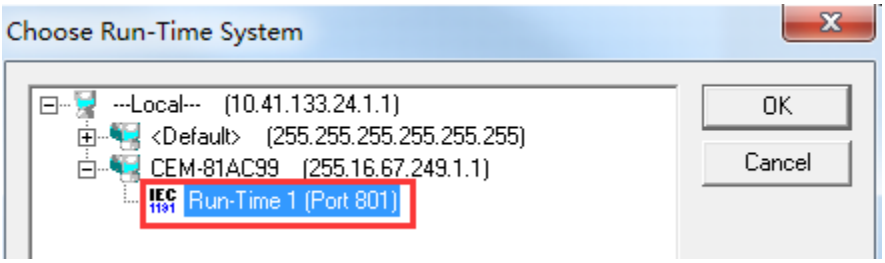
然后依次展开 MC_NCI，展开 standard，展开 Inputs，Outputs。





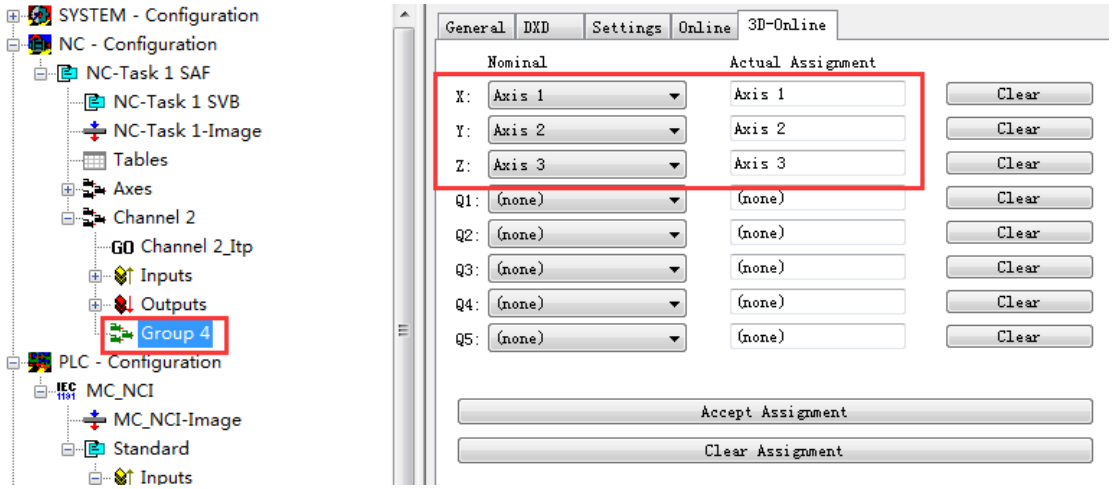
激活配置，单击  ，按照提示依次进行，将 twincat 软件切入 RUN 模式。

在 PLC Control 中，选择对应的 runtime，如下：



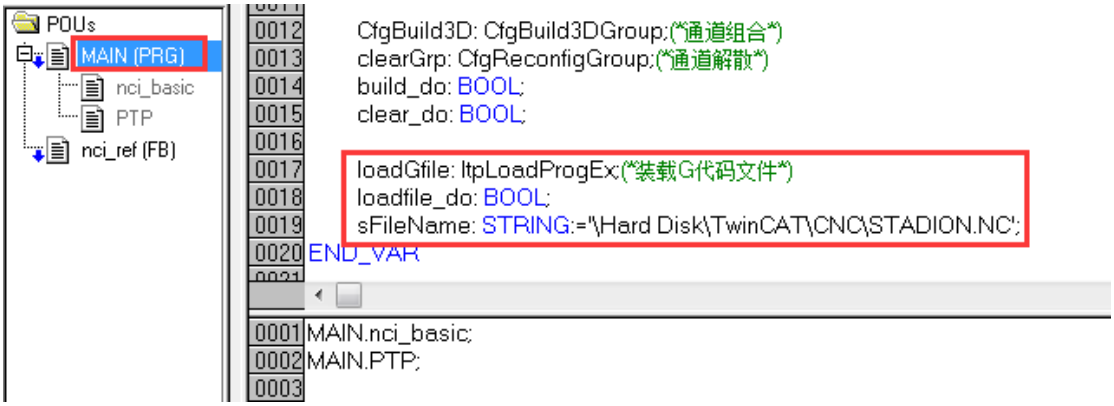
然后 Log In 程序  ，并且 run 程序  。

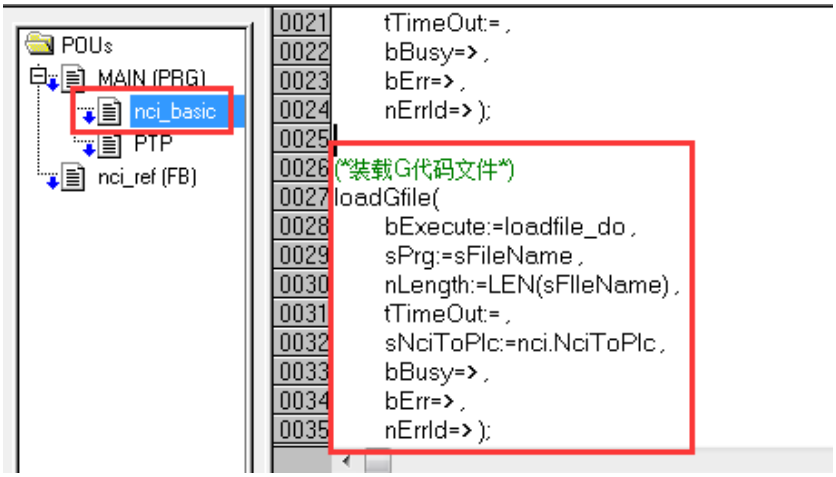
当按下人机界面中的“build_3D”按钮后，在 System manager 中，可以看见如下：



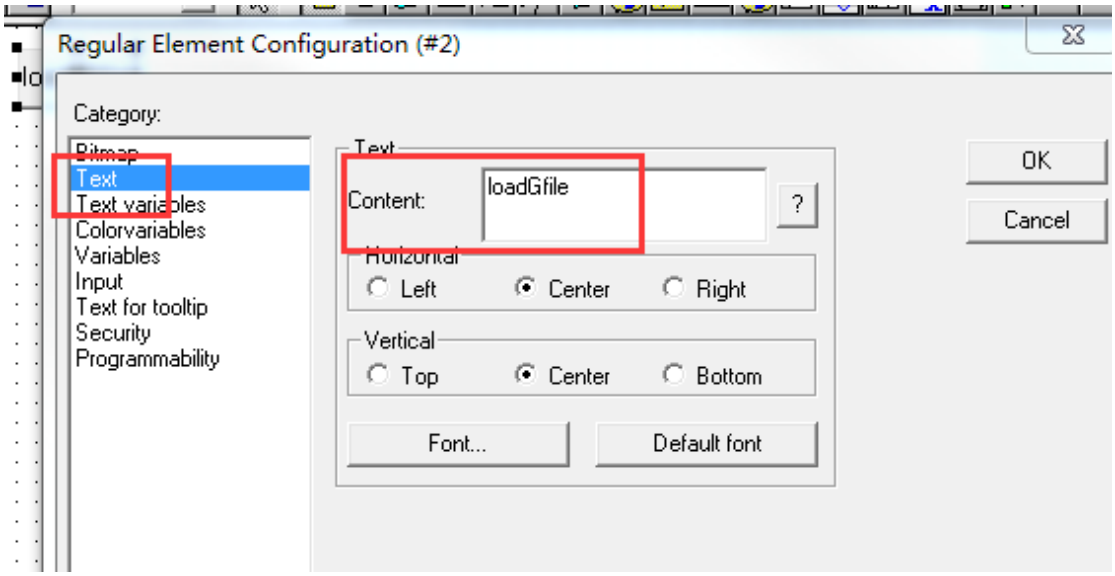
若需要解除三轴绑定，可按下人机界面中的”clear_3D”按钮。

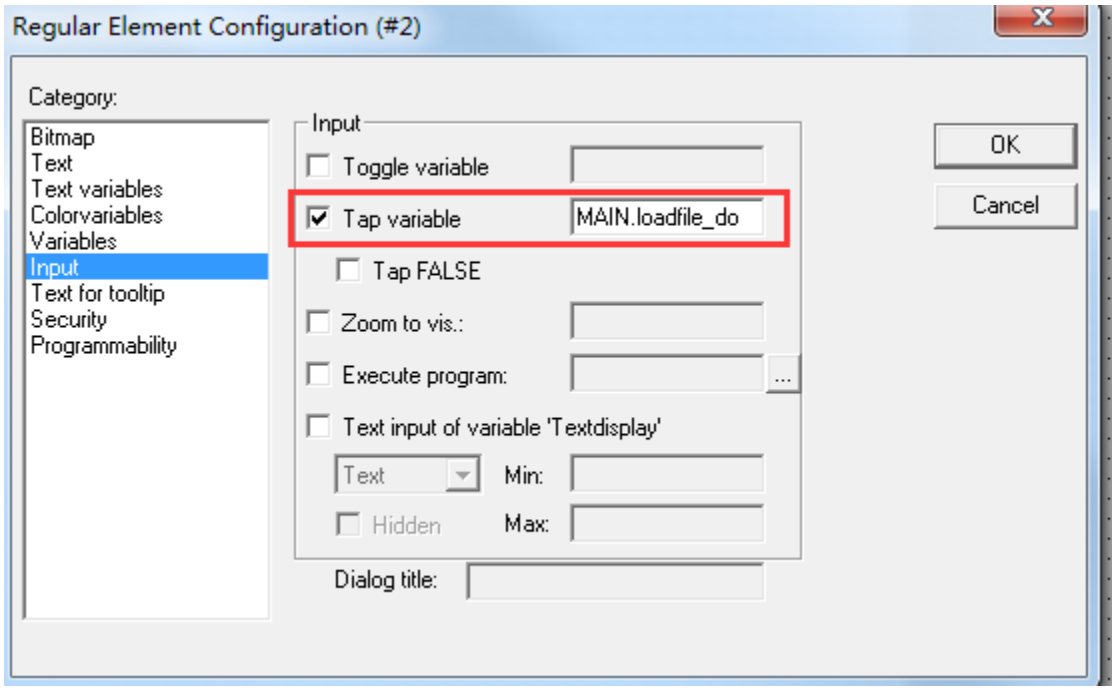
4) 装载 G 代码，调用 FB 功能块：ItpLoadProgEx，定义如下，其中 sFileName: STRING:='\Hard Disk\TwinCAT\CNC\STADION.NC' 表示：调用的.nc 的 G 代码文件所存的路径（根据自己存的地址编辑路径），此处需注意：若被控对象是控制器，请将 G 代码程序放置控制器中；若被控对象是虚拟机，请将 G 代码程序放置虚拟机中。



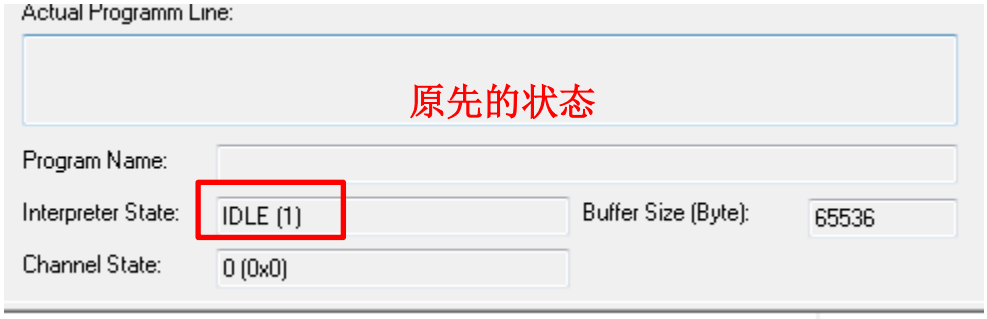


在人机界面中添加一个按钮  ，属性设置如下：

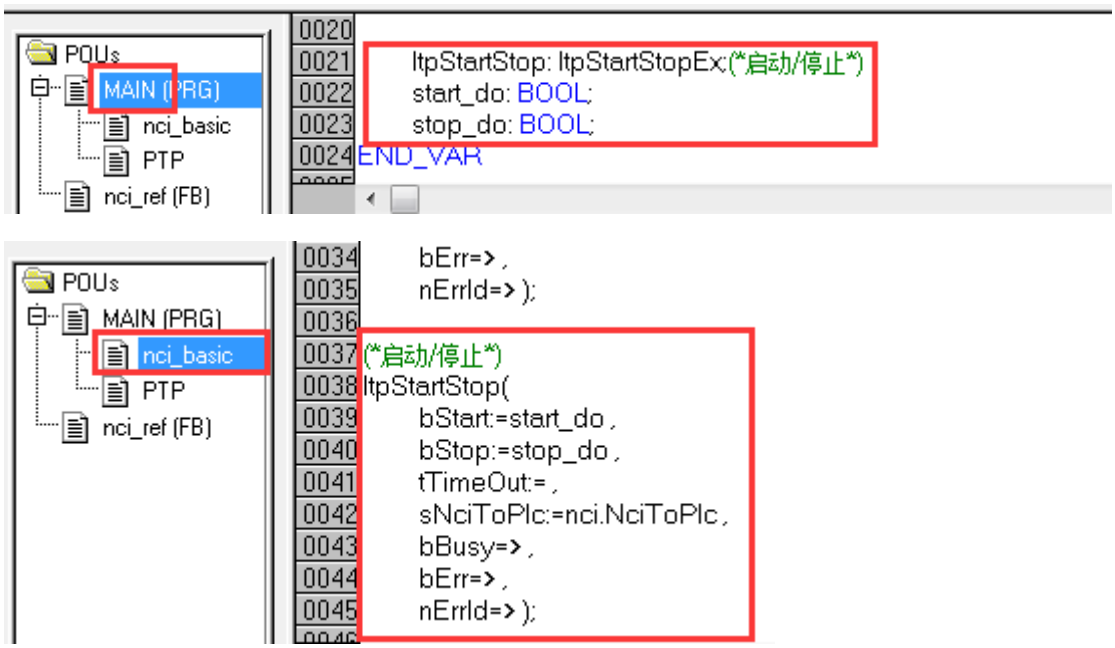




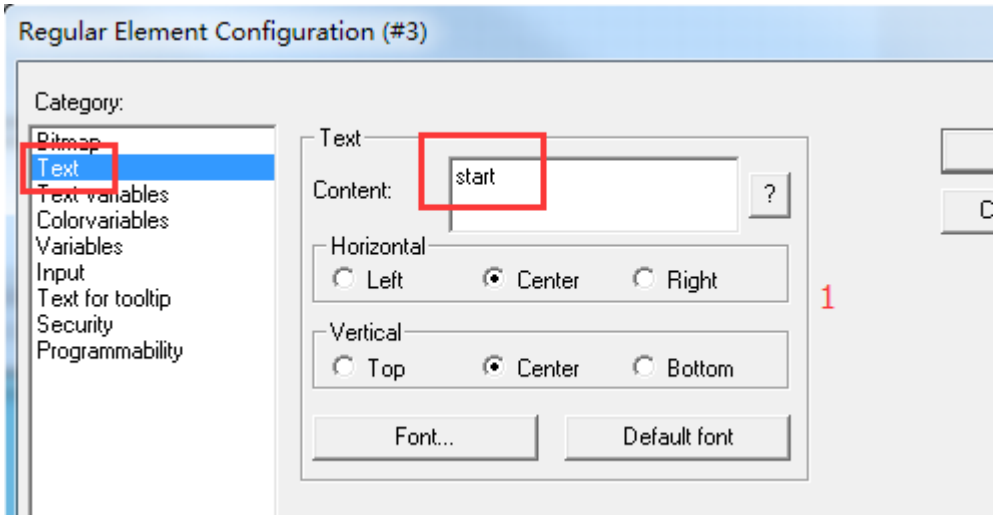
Log in 后，run 程序，可以在 system manager 中看见插补通道状态由 idle 变成 ready，说明 G 代码已经成功载入。

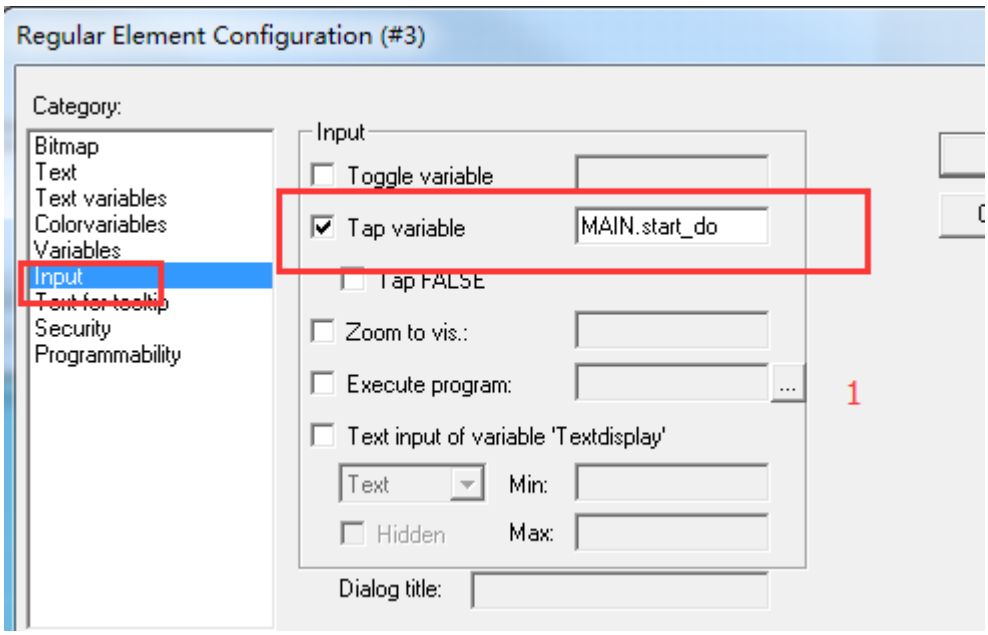


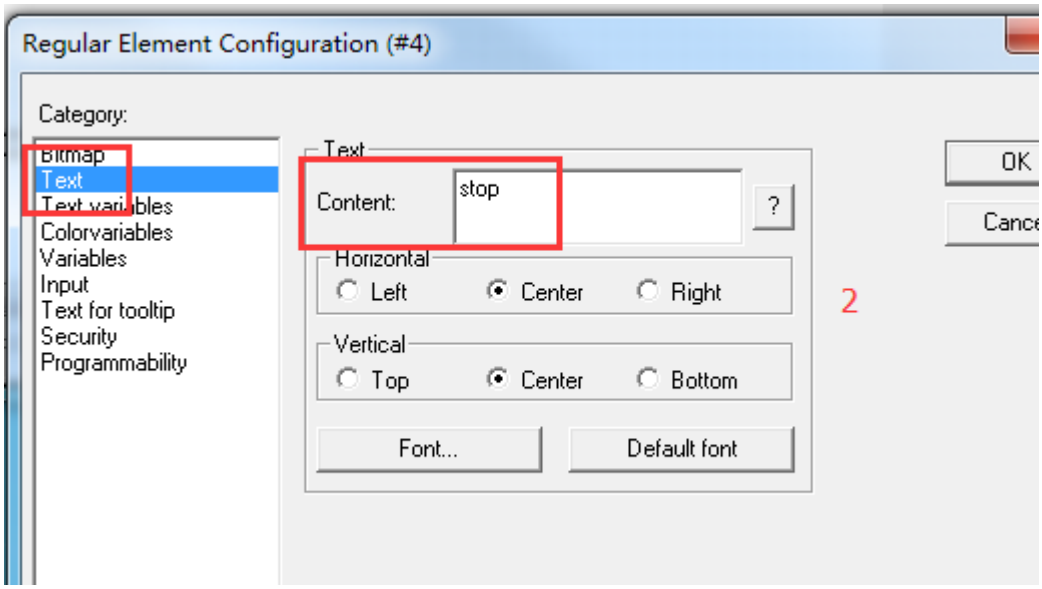
5) 执行/停止 G 代码的功能块 ItpStartStopEx，定义如下：

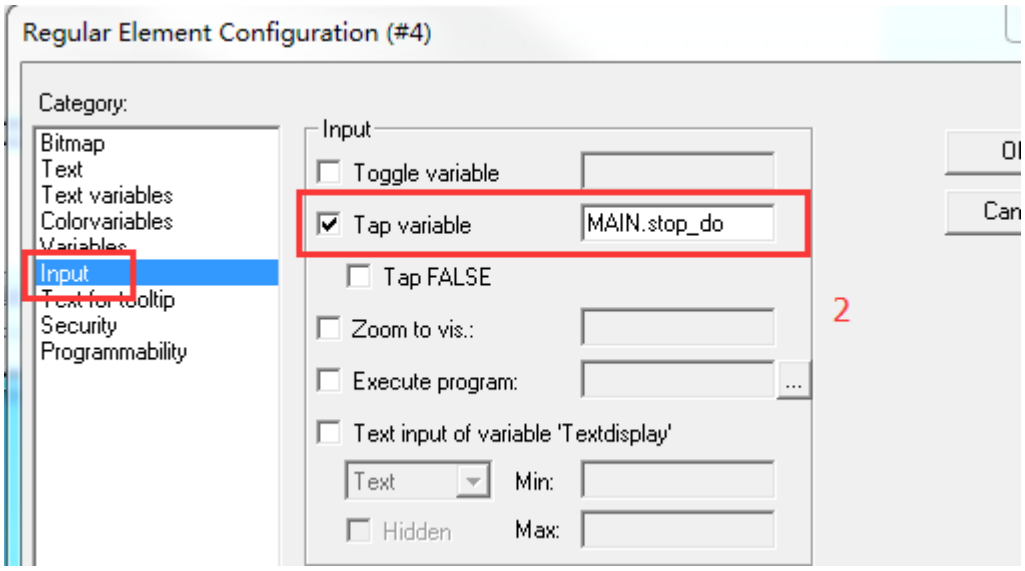


在人机界面中添加两个按钮  ，属性设置如下：

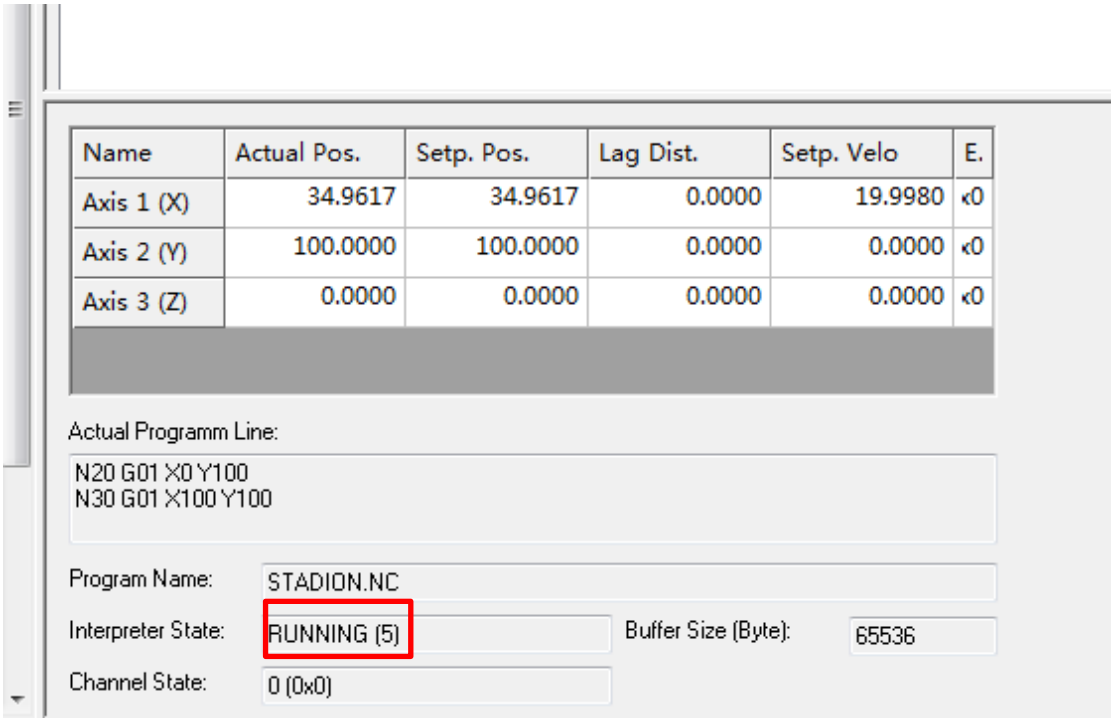



Regular Element Configuration (#3)
Category:
Bitmap
Text
Text variables
Colorvariables
Variables
Input
Text for tooltip
Security
Programmability
Input
Toggle variable
Tap variable
MAIN.start_do
Tap FALSE
Zoom to vis.:
Execute program:
Text input of variable 'Textdisplay'
Text
Min:
Hidden
Max:
Dialog title:
1

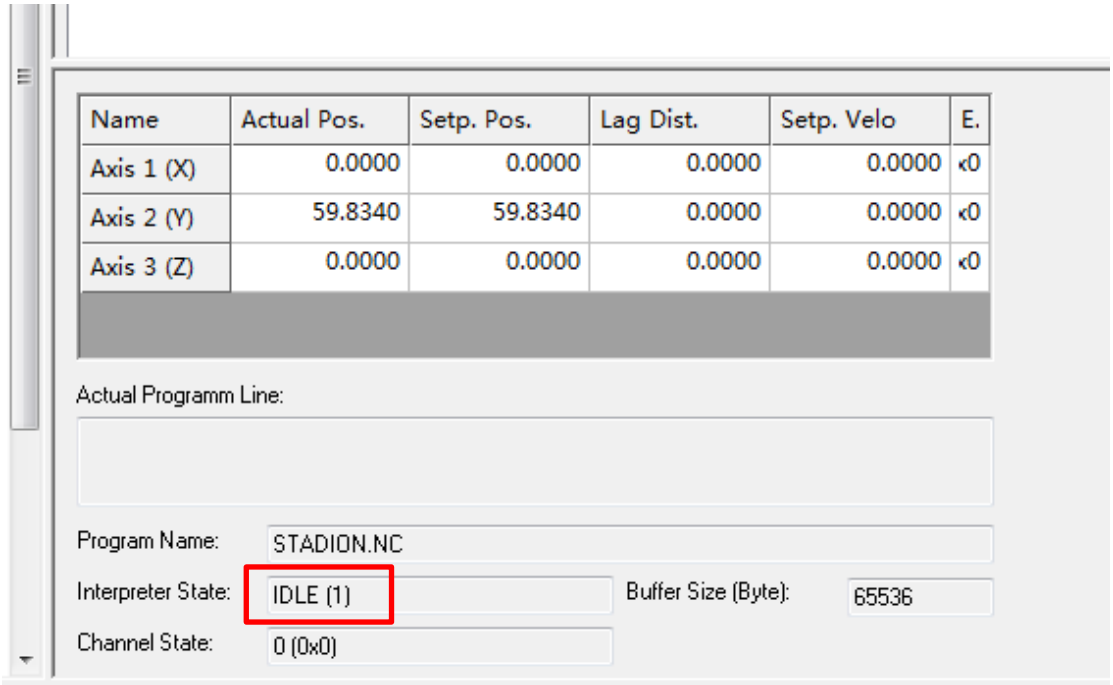

Regular Element Configuration (#4)
Category:
Bitmap
Text
Text variables
Colorvariables
Variables
Input
Text for tooltip
Security
Programmability
Text
Content:
stop
?
Horizontal
Left
Center
Right
Vertical
Top
Center
Bottom
Font...
Default font
OK
2


Regular Element Configuration (#4)
Category:
Bitmap
Text
Text variables
Colorvariables
Variables
Input
Text for tooltip
Security
Programmability
Input
Toggle variable
Tap variable
MAIN.stop_do
Tap FALSE
Zoom to vis.:
Execute program:
Text input of variable 'Textdisplay'
Text
Min:
Hidden
Max:
2

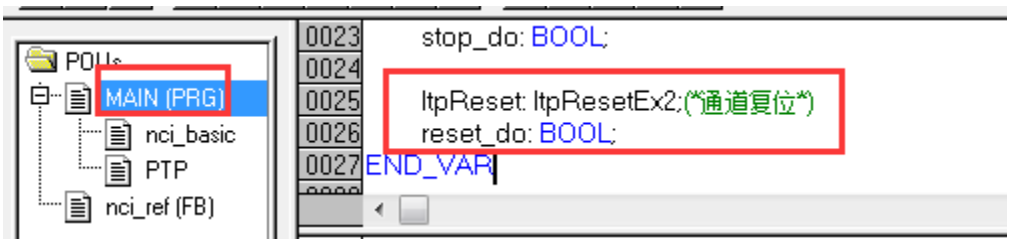
Log in 后，run 程序，单击 start 按钮，可以在 system manager 中看见插补通道如下状态，

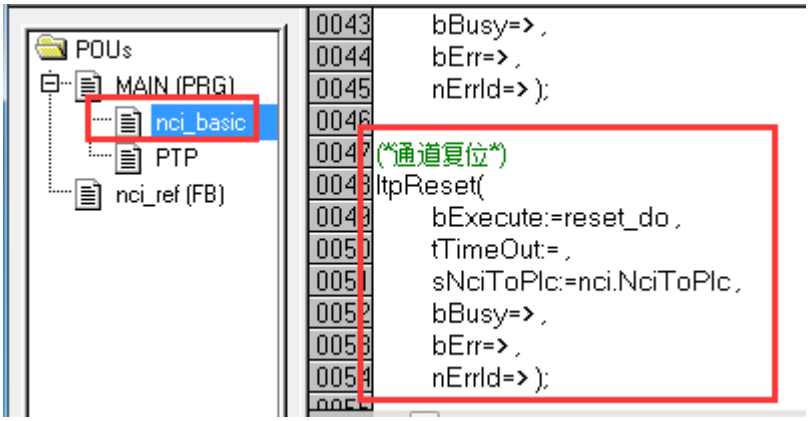


若在 G 代码运行的过程中，单击 stop 按钮，可看见如下状态：

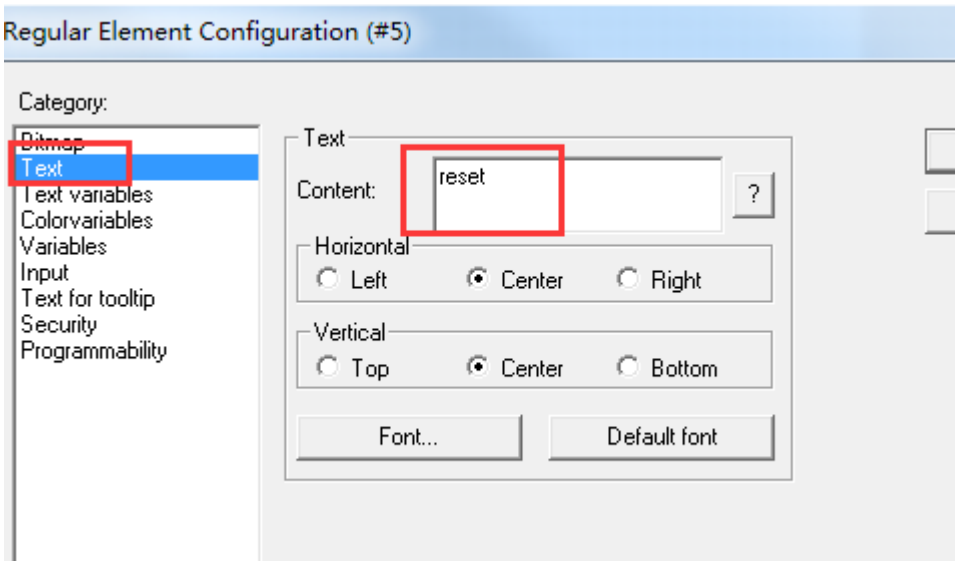


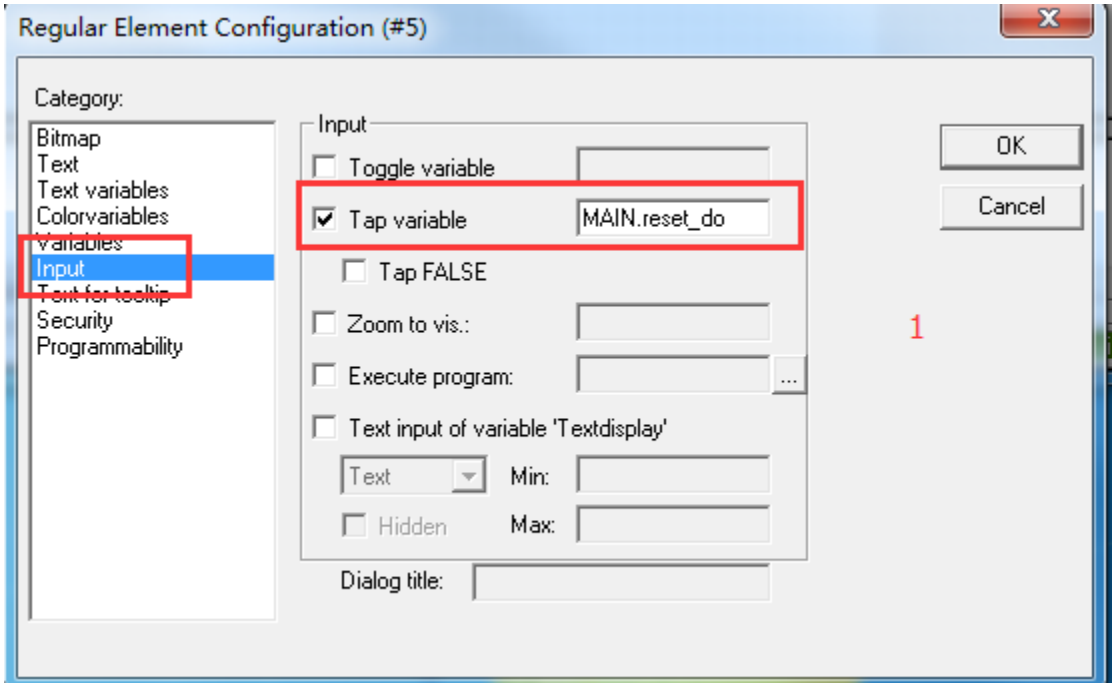
6) 若插补通道出现报错时，需要用到通道复位功能块：ItpResetEx2，定义如下：

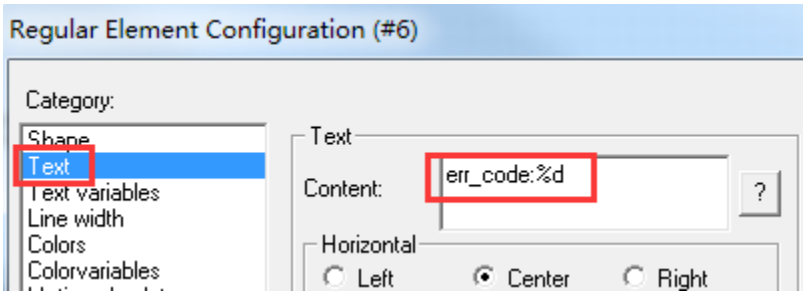


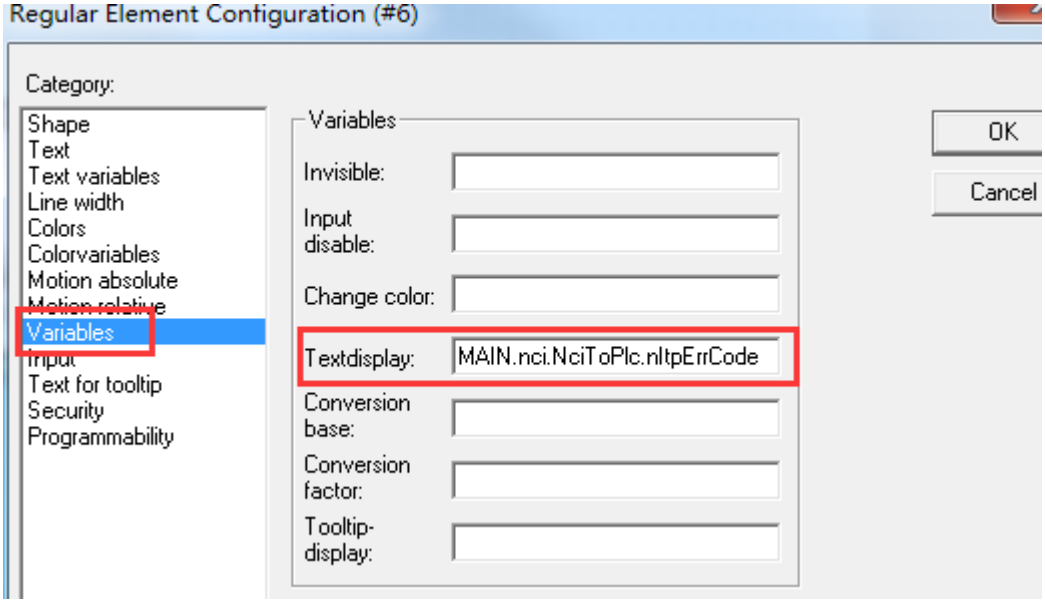


在人机界面中添加一个按钮，一个矩形框，属性设置如下：



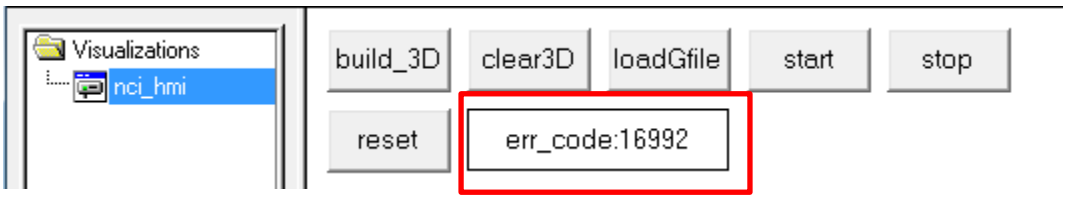




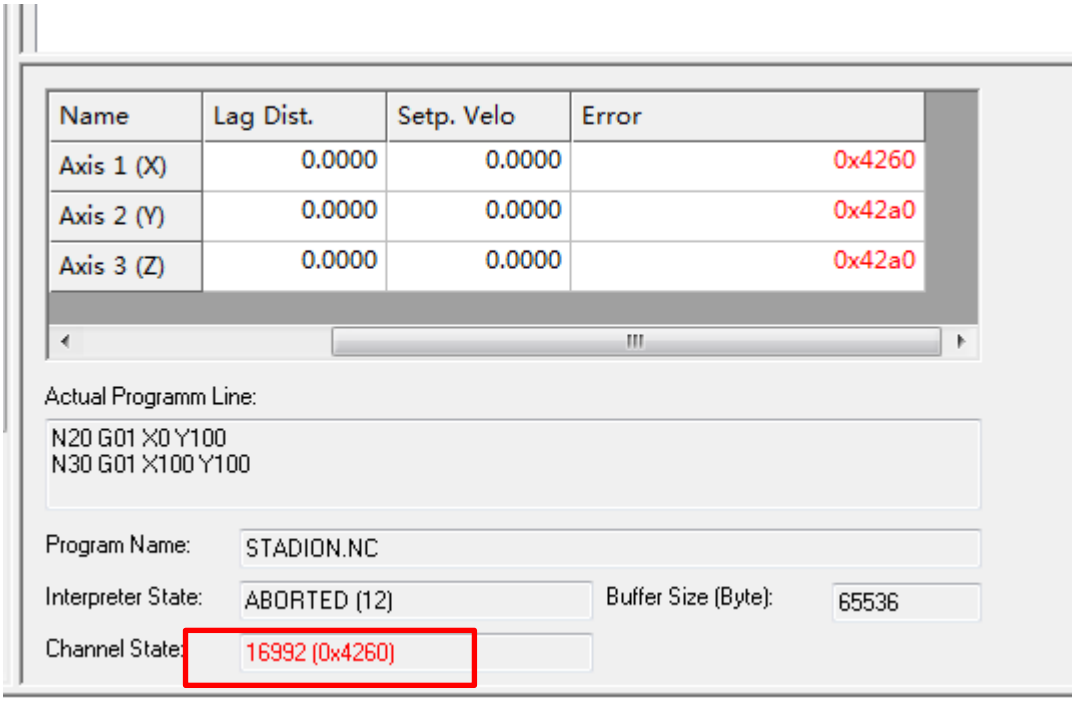


Log in 后，run 程序，单击 start 按钮，现人工使轴产生一个报错，报错结果如下，报错信息可以查看 information system 帮助文档。

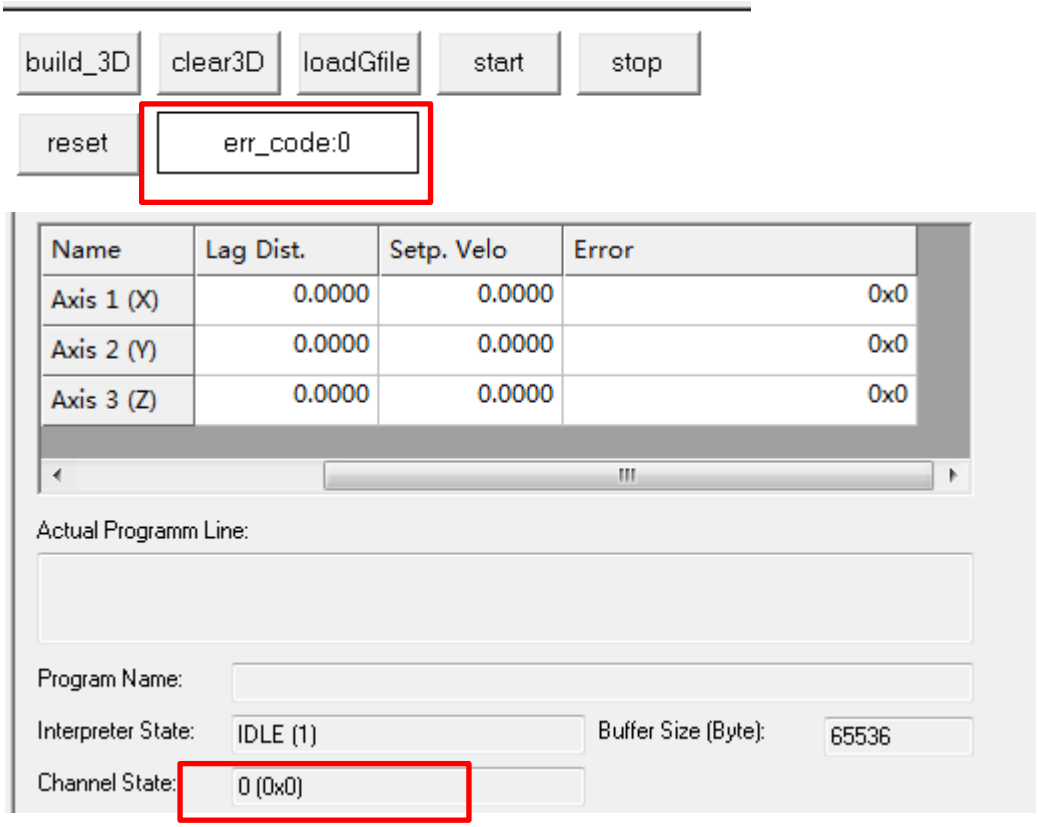
（可用人机界面查看）



（也可在 system manager 中查看插补通道）



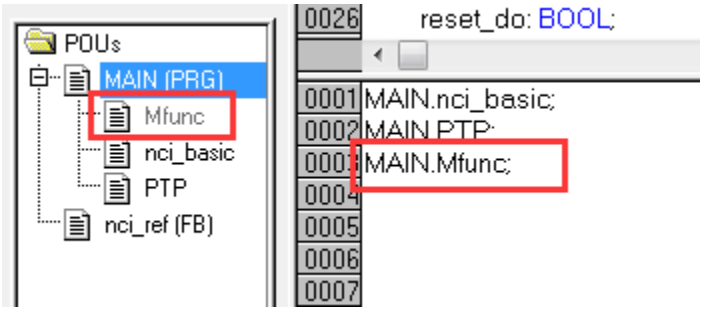
首先针对报错原因进行查找，并且改正。（16992 使能报错，由于 AXIS1 轴使能掉了造成的，现加回使能），接下来对插补通道进行复位。单击 reset。可发现错误信息被清除，轴已复位。



以上即为 nci_basic 这个基本 action 部分的介绍。

(7) PLC 中对握手类型的 M 指令复位

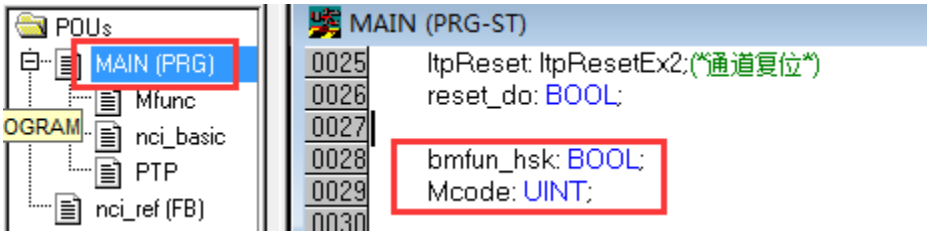
首先在左侧对象管理窗口中建立一个 Action，命名为 Mfunc，并在 MAIN 程序区进行调用。

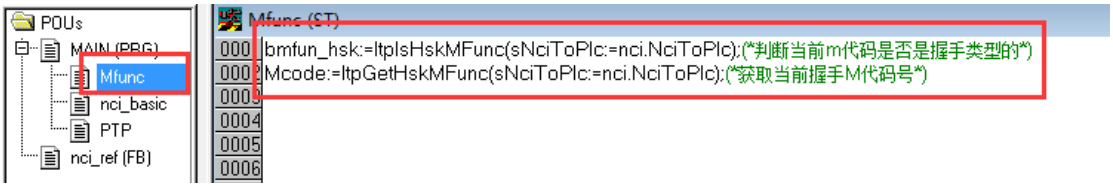


完成了框架部分，针对 M 指令部分进行编程：

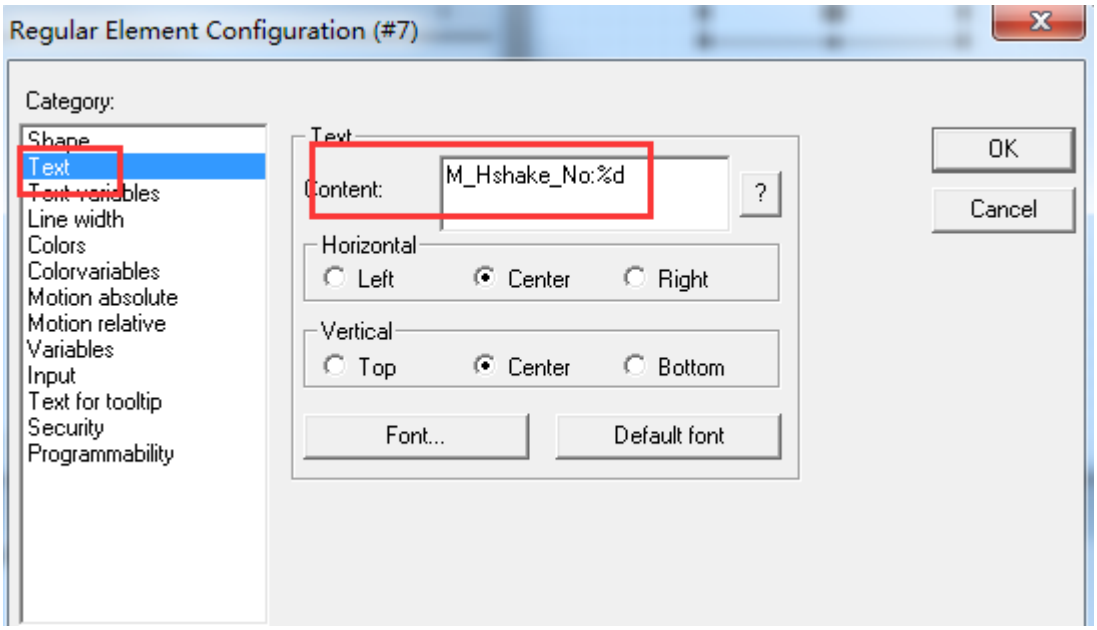
1) 调用 FUN 来获取当前 M 指令是否为握手类型，该功能为 ItpIsHskMFunc。若是握手类型，读取该握手指令的编号，该功能为 ItpGetHskMFunc。

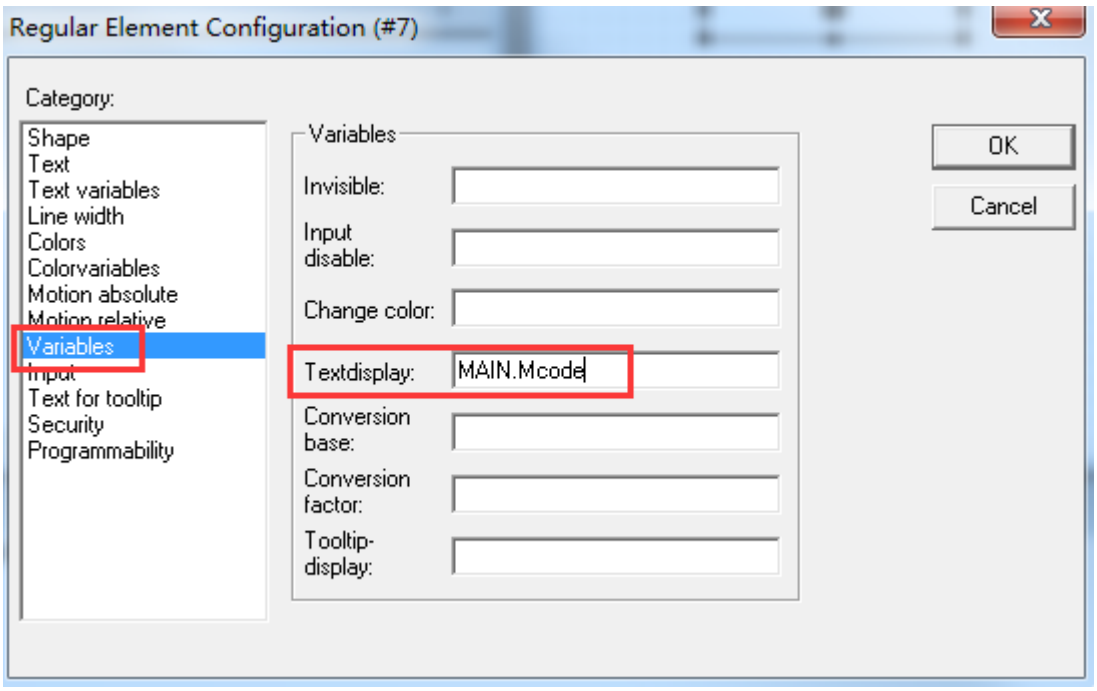
首先定义两个变量：bumfun_hsk 读取 M 指令是否为握手类型；Mcode 读取握手指令的编号。





同时在人机界面中，一个矩形框用于显示握手类型 M 指令的编码，矩形框属性设置如下：





现已经设置了 M10 为 BM 类型的握手指令。执行的 G 代码程序如下：

N10 G01 X0 Y0 F1200

N20 G01 X0 Y100

N30 G01 X100 Y100

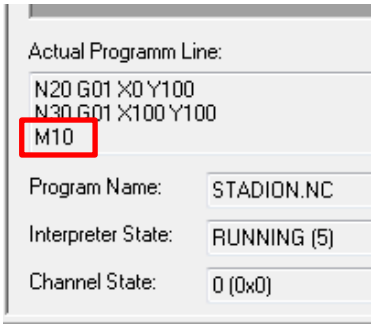
M10

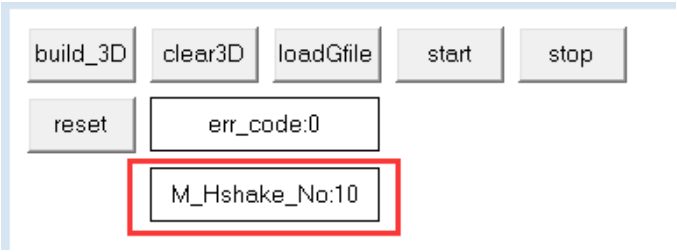
N40 M20 G01 X100 Y0

N50 G01 X0 Y0

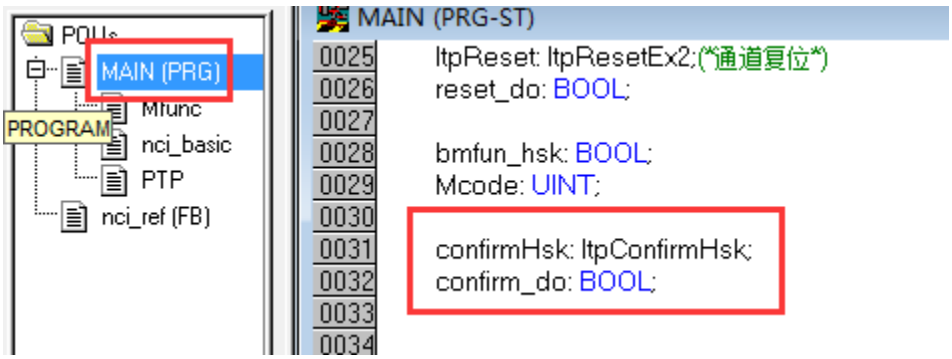
N50 M30

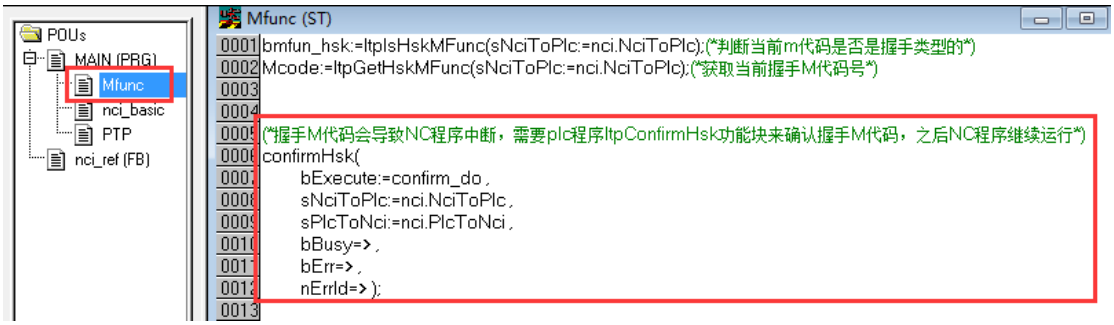
当 G 代码开始执行时，到达 M10 这一行的时候，如下显示：（左图是 System manager 软件的显示状态，右图是 PLC 中 nci_hm 界面的显示）



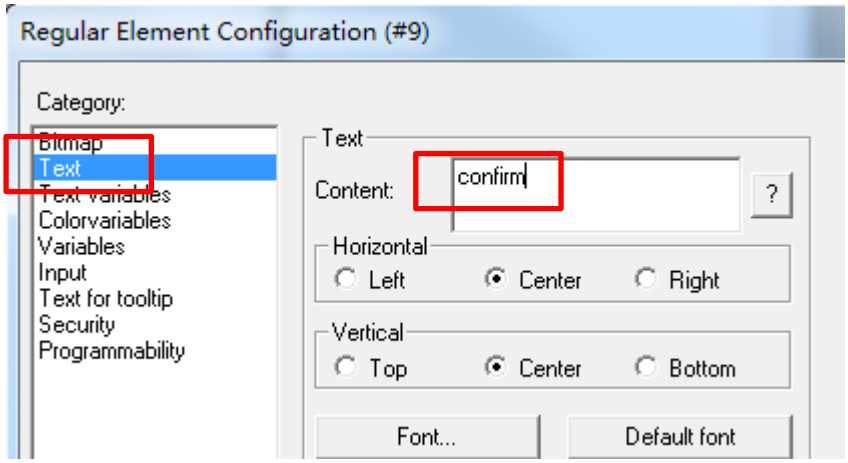


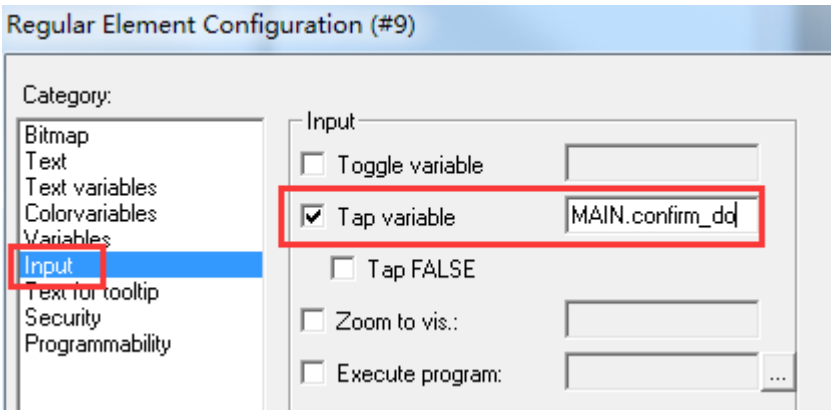
2) 假设 M10 在插补通道中被激活后，需要通过 HMI 界面上的按钮来确认，复位 M10 指令后，G 代码才可以继续执行，程序如下：





在 nci_hmi 中，添加一个按钮 OK，其属性如下：

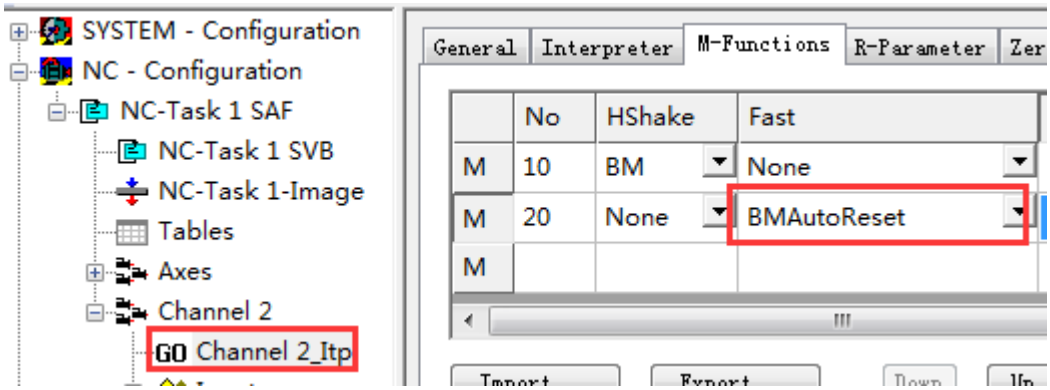




Log in 后，按下 confirm 按钮后，发现 G 代码程序继续向下执行，至结束。

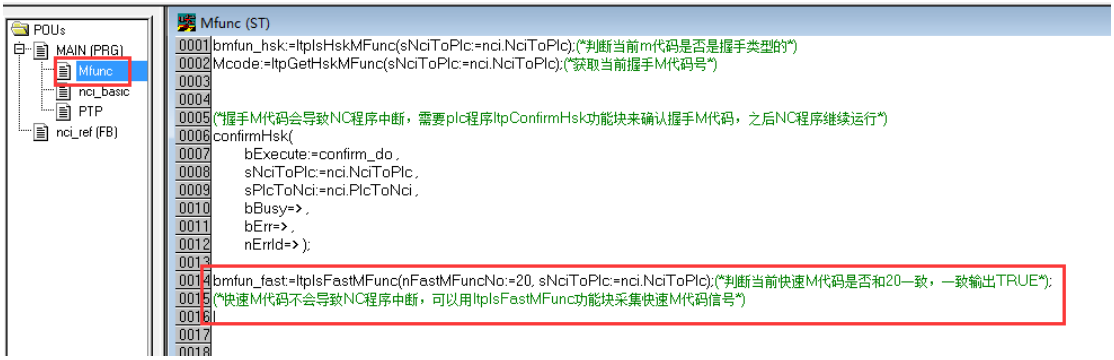
(8) PLC 中对快速类型的 M 指令检测

Fast 类型的 M 指令是不需要 PLC 确认的，但是可以用 PLC 程序检测此 M 指令是否生效，用于检测的功能为：ltpIsFastMFunc。现用 M20，定义如下：

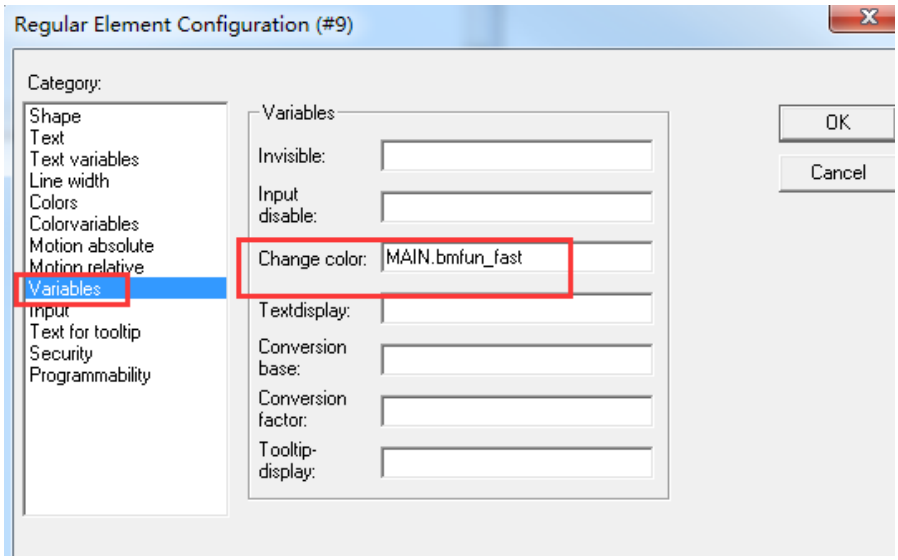


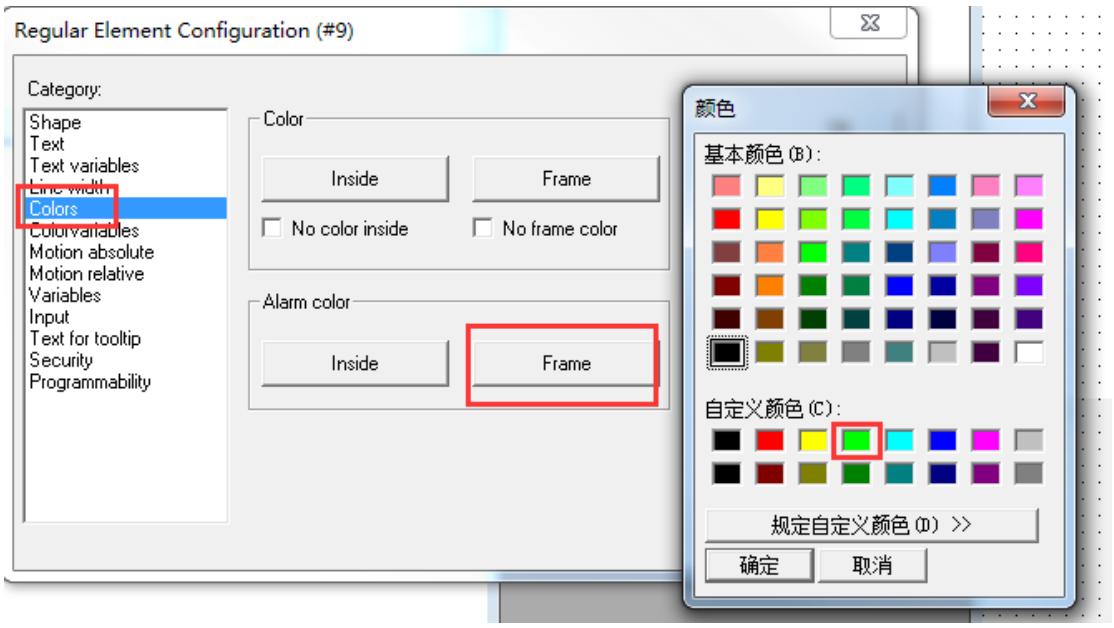
那么在程序去定义一个变量 bmfun_fast，该变量为 BOOL 型，若当前快速型 M 代码的编号如果是 20，那么该变量置为 TRUE。并在人机界面 nci_hmi 中创建一个圆形框，用于显示被触发的快速型 M 指令状态。

程序如下：其中将 bmfun_fast 定义成 BOOL 型；

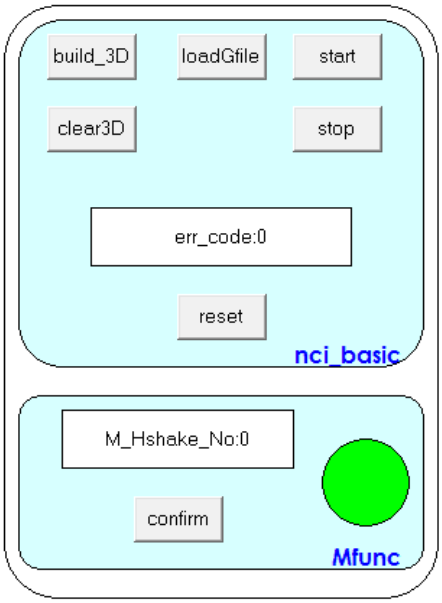


关于圆形框的设置如下：





Log in 后，当 G 代码执行到 M20 行处，可以发现在执行该行代码时，bmfun_fast 输出为 true 且圆形框变成绿色。



NCI 功能可以参照 U 盘中提供的例子：
培训用 U 盘\运动控制培训\NCI

十二.　AX5000 第二反馈配置步骤

本章目标：

通过本章节的学习，学员将了解：

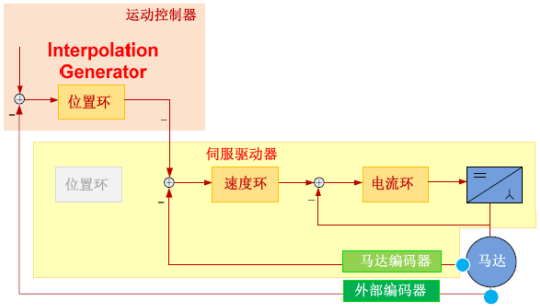
☑ 全闭环控制

☑ 第二反馈的配置方式

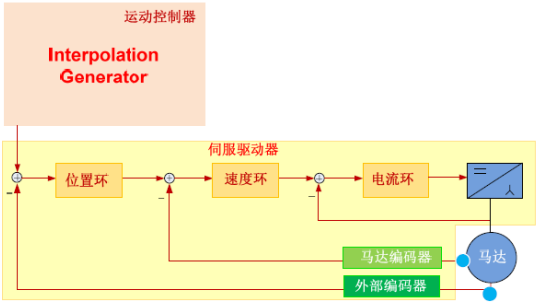
☑ 第二反馈的硬件接口

1. 第二反馈的应用场合

背隙、打滑等往往会造成设备精度的影响，常见的方式是在执行机构的终端上再加一个光栅尺或编码器形成一个全闭环回路控制，这些反馈信号可以接到EL/KL5xxx模块上，也可以接到AX5000空余的编码器接口上作为第二反馈，闭环回路控制可在运动控制器上实现，亦可在伺服驱动器上实现。这里介绍的是如何在伺服驱动器上实现闭环控制。



控制器全闭环控制



驱动器全闭环控制

2. 硬件准备以及接线

控制器：CX5020-0112

伺服驱动器：AX5206-0000-0200

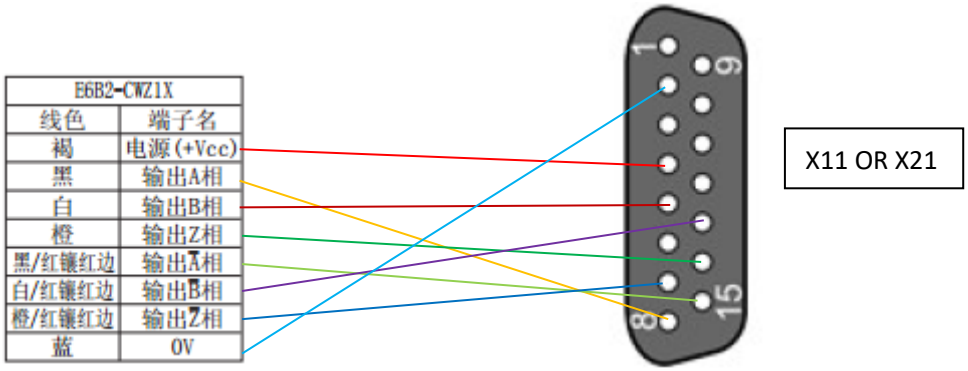
伺服电机：AM8021-0B20-0000，AM8021-0B10-0000

编码器：Omron E6B2-CWZ1X（TTL 增量式编码器）

驱动器的 X11 和 X21 可以支持多种类型的编码器反馈信号，本例中参照 TTL 管脚定义

Pin	EnDAT / BiSS	Hiperface	Sine / Cosine 1Vpp	TTL	output current
1	SIN	SIN	SIN	n.c.	
2	GND_5 V	GND_9 V	GND_5 V	GND_5 V	
3	COS	COS	COS	n.c.	
4	U_s_5 V	n.c.	U_s_5 V	U_s_5 V	
5	DX+ (Data)	DX+ (Data)	n.c.	B+	
6	n.c.	U_s_9V	n.c.	n.c.	
7	n.c.	n.c.	REFZ	REFZ	max 250 mA / chanel
8	CLK+ (Clock)	n.c.	n.c.	A+	
9	REFSIN	REFSIN	REFSIN	n.c.	
10	GND_Sense	n.c.	GND_Sense	GND_Sense	
11	REFCOS	REFCOS	REFCOS	n.c.	
12	U_s_5 V_Sense	n.c.	U_s_5 V_Sense	U_s_5 V_Sense	
13	DX- (Data)	DX- (Data)	n.c.	B-	
14	n.c.	n.c.	Z	Z	
15	CLK- (Clock)	n.c.	n.c.	A-	

编码器线序如下：需要根据两边的线序焊接专用接头。



由于伺服驱动器本身与 AM8021 连接的方式为 OCT（一根电缆连接方式）,驱动器自带的 X11 与 X21 接口没有使用，因此可以将编码器接到 X11 上当做第二反馈使用，如下图。

Q:TTL 的编码器使用需要注意什么事项？

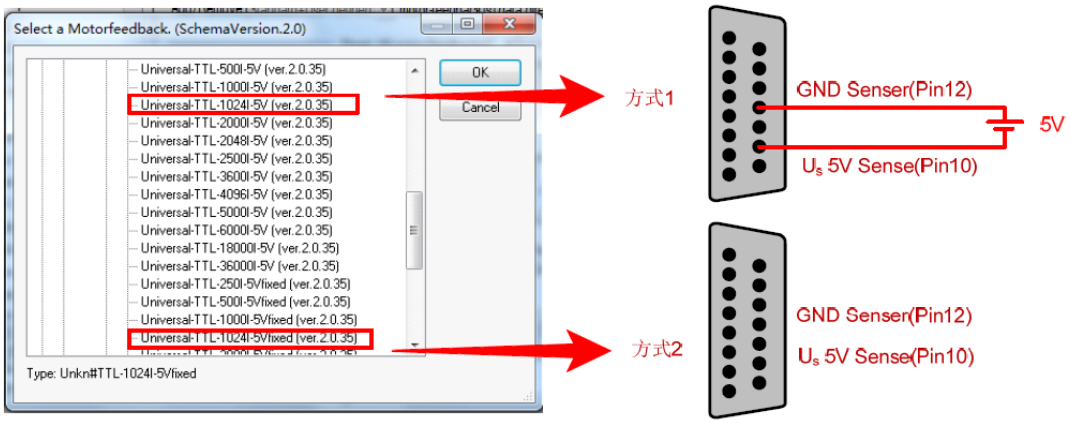
A:以 TTL 输出，1024 分辨率差分5V 输出编码器为例：

1）若采用Universal-XXX-5Vfixied 方式（见下图方式2），则伺服的Pin10,12 脚可不用接线，

此方式下，不连编码器驱动不会报警；

2）而采用Universal-XXX 方式（见下图方式1），则伺服编码器接口的Pin10，Pin12 要加上

5V 的电压，此方式下，不连编码器驱动器会报编码器断线错误 Lost feedback【F707】。



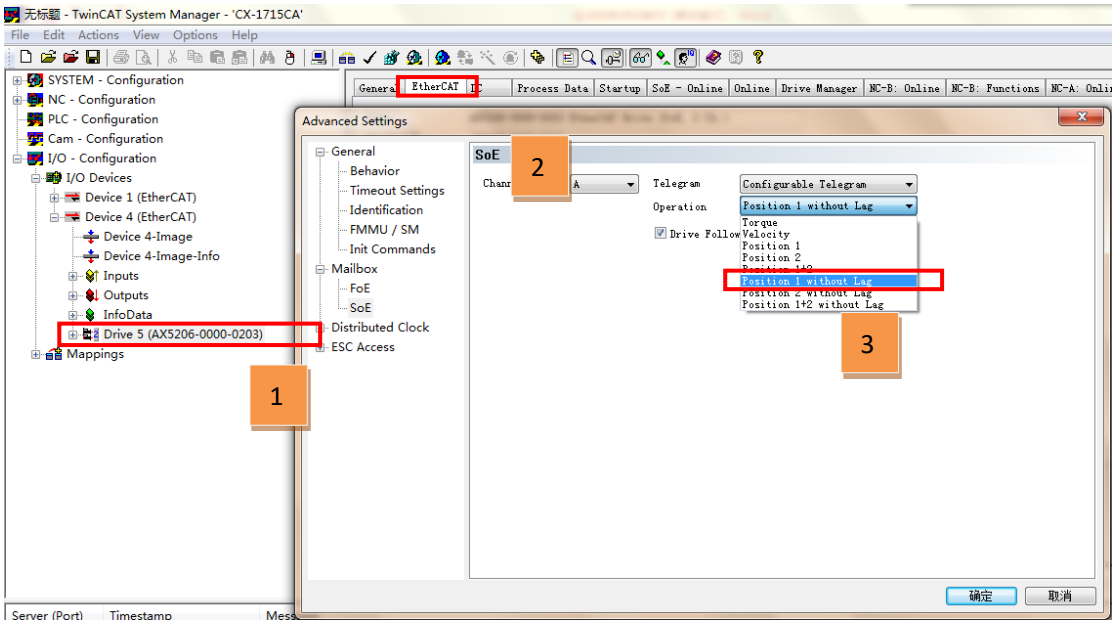
3. AX5000 中配置第二反馈

1） 配置电机与第一编码器反馈，检测电机是否正常；

打开 TwinCAT Manager，在Config Mode 下进行Scan Devices，配置好电机和电机编码器参数。并激活配置，检测是否可以正常电机操作。

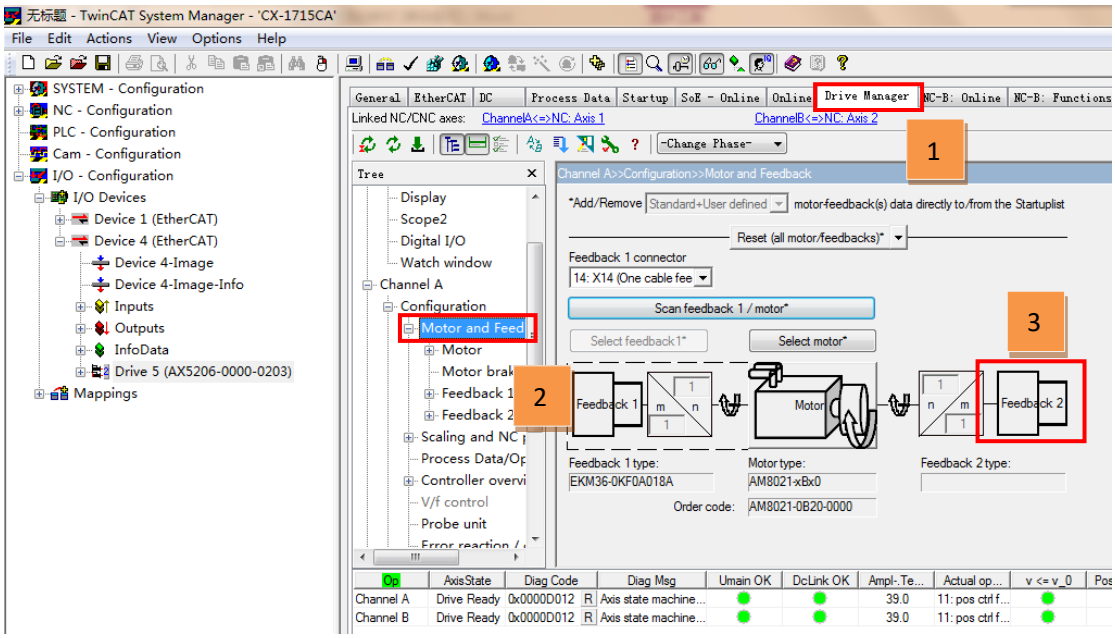
2） 将驱动器设置为 Position 2 without lag（12）模式；

点击【Drive 1(AX5203-0000-0201)】/【EtherCAT】/【MailBox】/【SoE】，由于本次测试使用通Channel B，因此选择channel【B】，将其模式设定为【Position 2 without lag】。

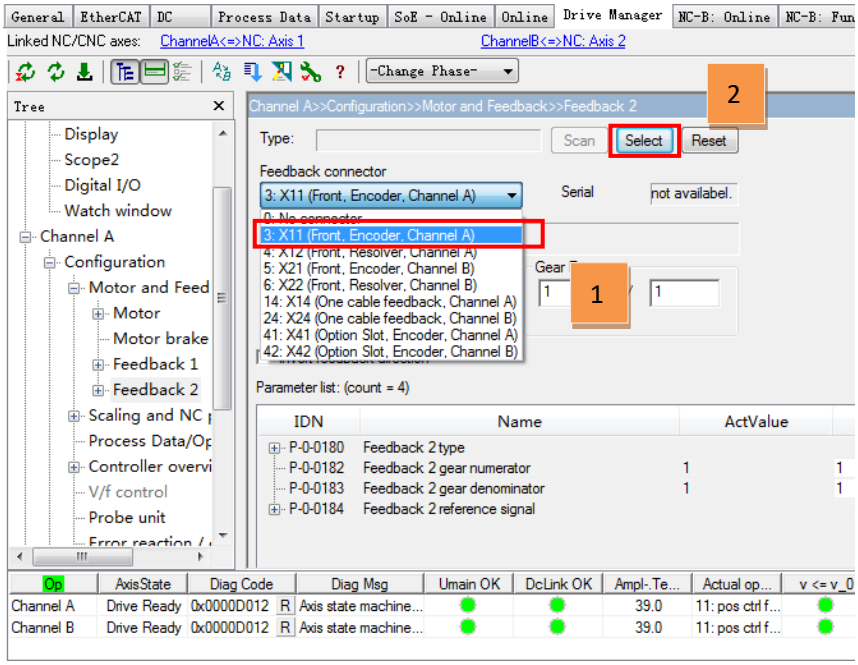


3） 在驱动器的 Drive Manager 窗口中配置第二反馈

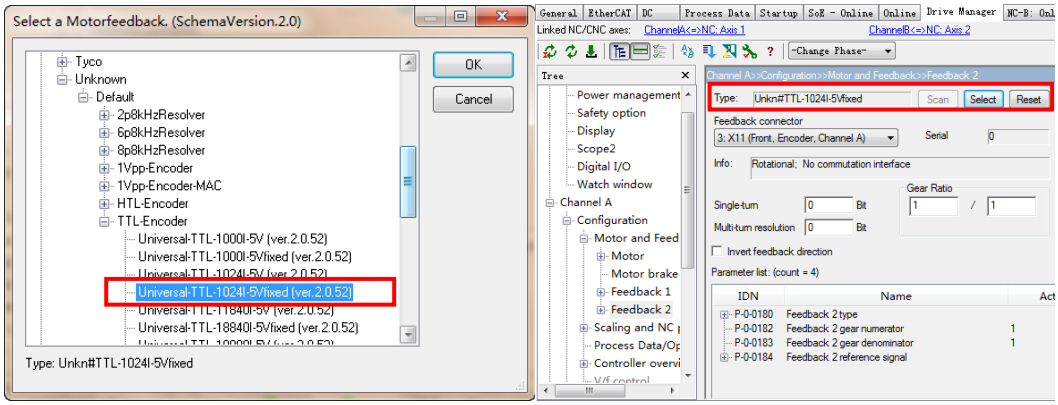
点击 Drive Manager——Motor and Feedback——Feedback 2 选择第二反馈



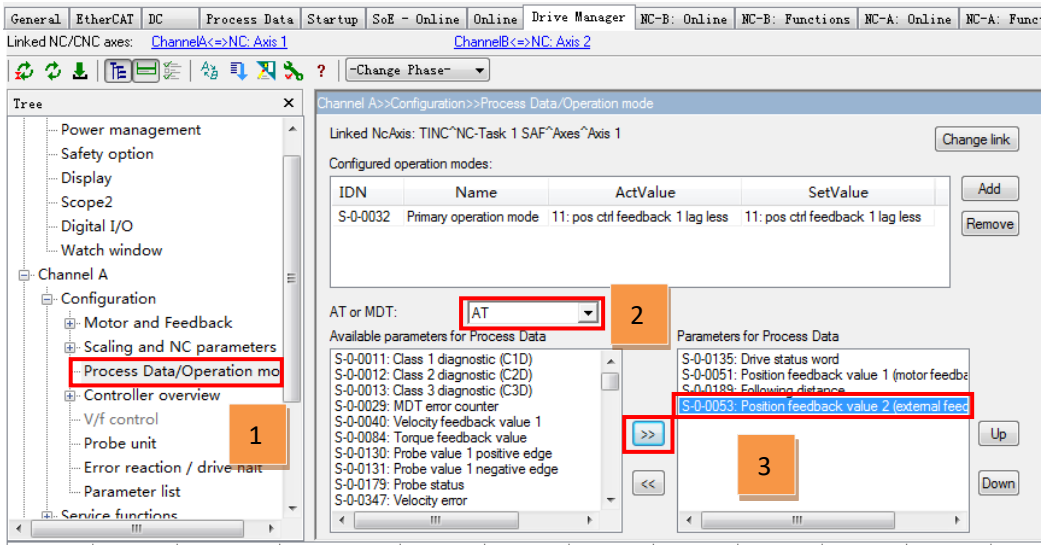
4） 选择编码器连接的实际接口，此例选择 X11，然后点击 select 选择对应的编码器类型



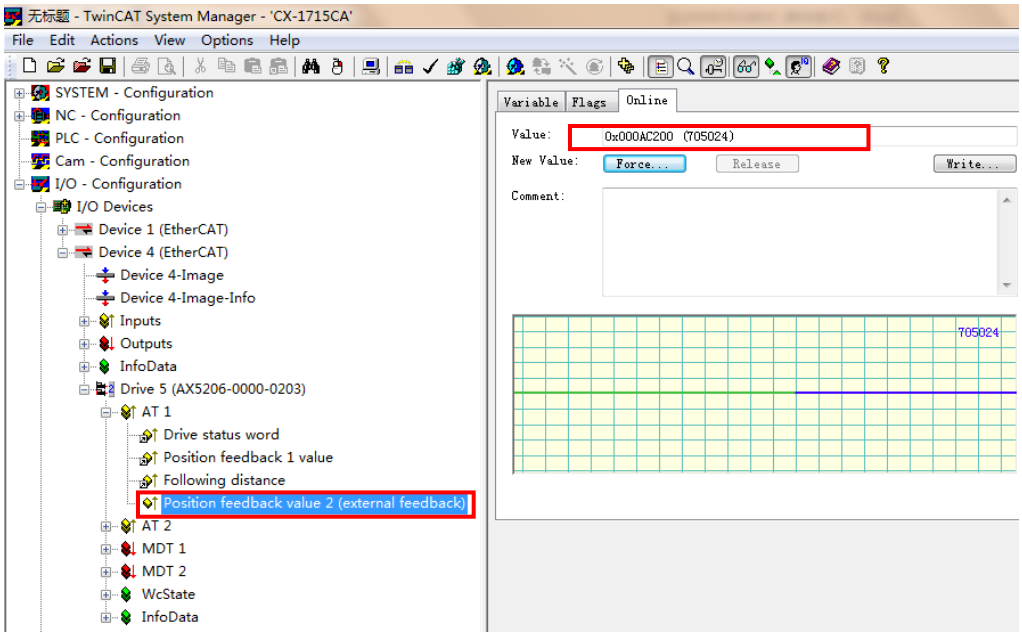
5） Rotational Motorfeedbacks——Unknown——Default——TTL-Encoder——Universal-TTL-2014l-5Vfixed(ver.2.0.52)



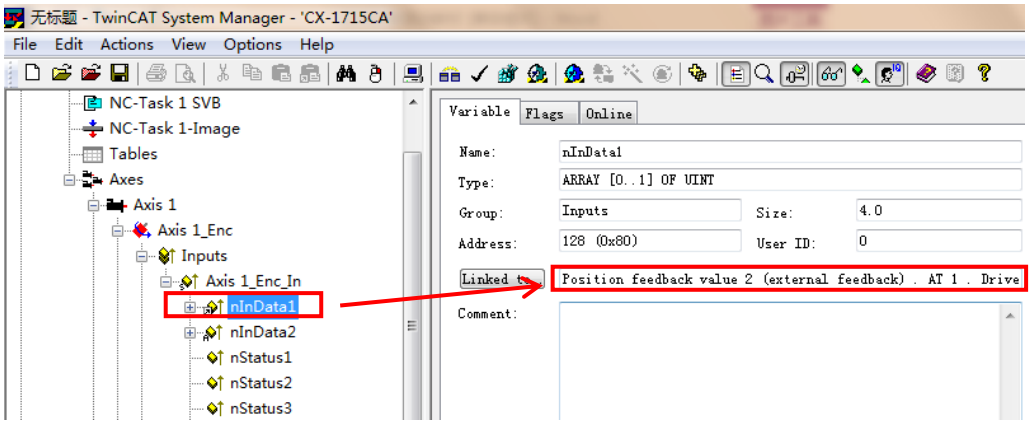
6） 点击 Channel A——Configuration——Process Data/Operation mode，将 S-0-0053 第二反馈添加至 process data 中



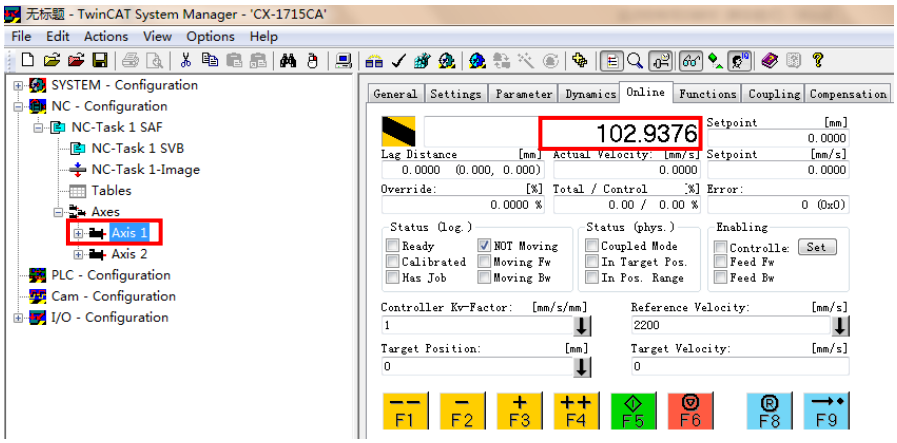
7） 激活配置  并重启 twincat 之后，在 AX5206-0000-0203——AT1——Position feedback value2(external feedback)可以查看第二反馈的当前位置，转动编码器一圈之后，value 的值变化 1048576(AX5000 默认一圈为 1048576 个脉冲，无论所连编码器实际反馈脉冲多少).



8） 将 AXIS1——Aixs1_ENC——inputs——AXIS 1_Enc_In 下的 nInData1 变量连接到 Position feedback value2 上面，再次激活配置并重启 twincat。



9） 此时可以在 NC——Axis1——Online 中可以看到轴的当前位置，转动编码器之后可以看到 NC 位置的变化。



十三.　AX5000 数字量 I/O 的使用

本章目标：

通过本章节的学习，学员将了解：

☑ AX5000驱动器I/O点如何接线

☑ AX5000驱动器I/O如何通过软件配置

☑ 硬件限位如何配置

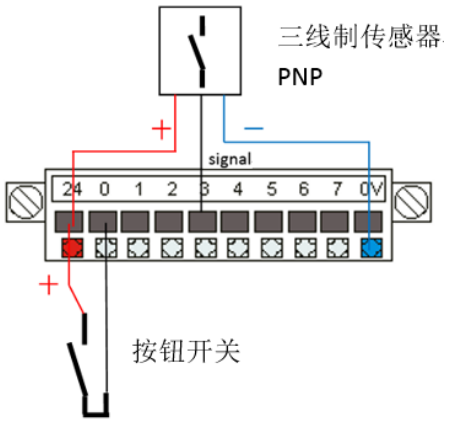
☑ AX5000驱动器I/O可以实现那些功能

1. 硬件接线与 I/O 信号获取

1) 首先需要对驱动器 X03 接口进行供电，将 DC 24V 接到 Up 与 Us，0V 接到 GND 上，Up 供电之后，驱动器的 X06 I/O 端子上的 24v 和 0v 即可输出 24V。

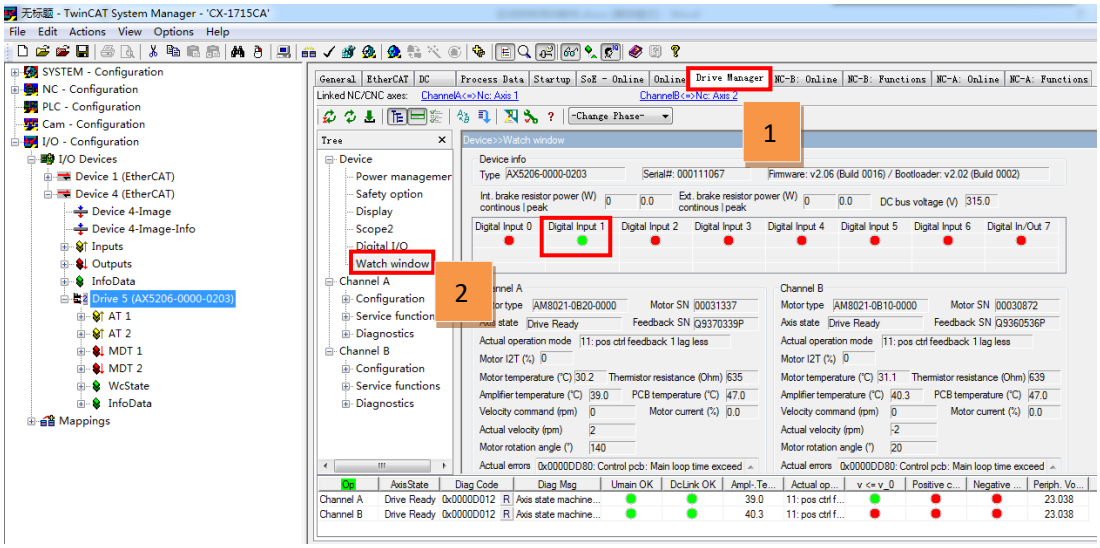
X03

2) 对于不同的输入信号可以参照下图的接线方式



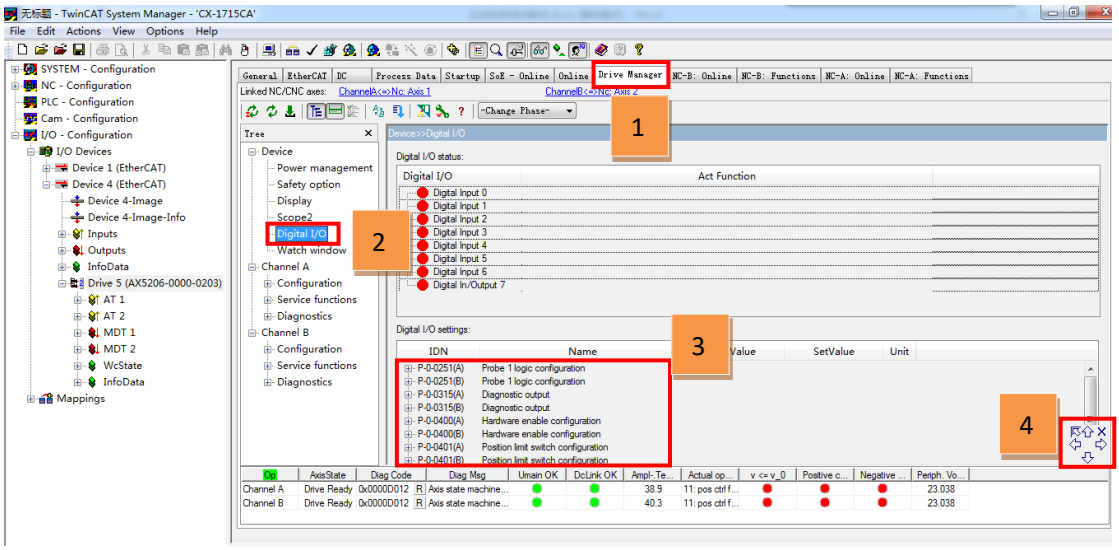
X06

3) 当外部输入信号高电平之后，可以在 Drive5——Drive Manager——Watch window 查看 IO 的 ON/OFF 状态（此窗口同样可以查看电机的转速，温度，报错等其他信息）

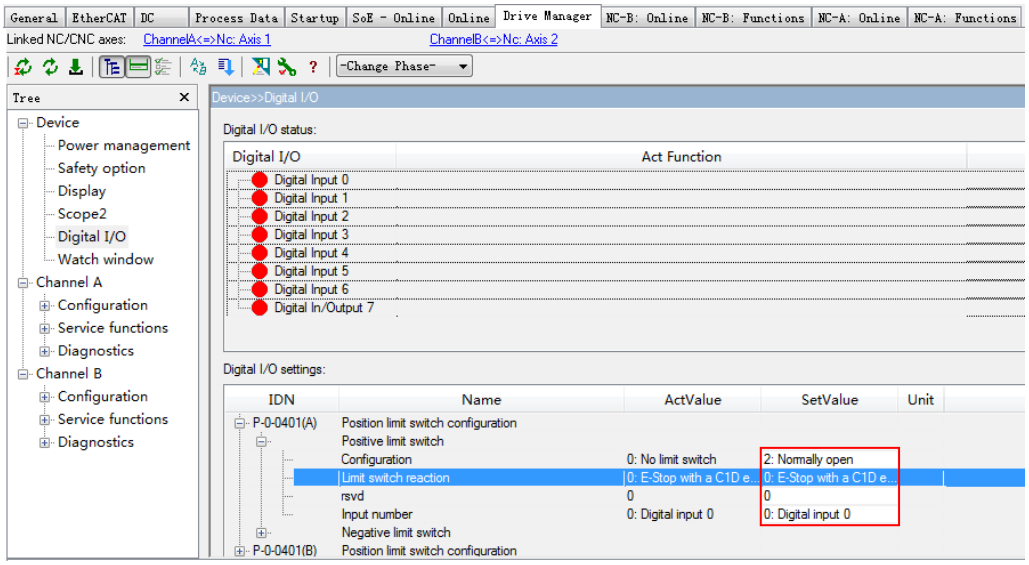


2. 设置硬件限位

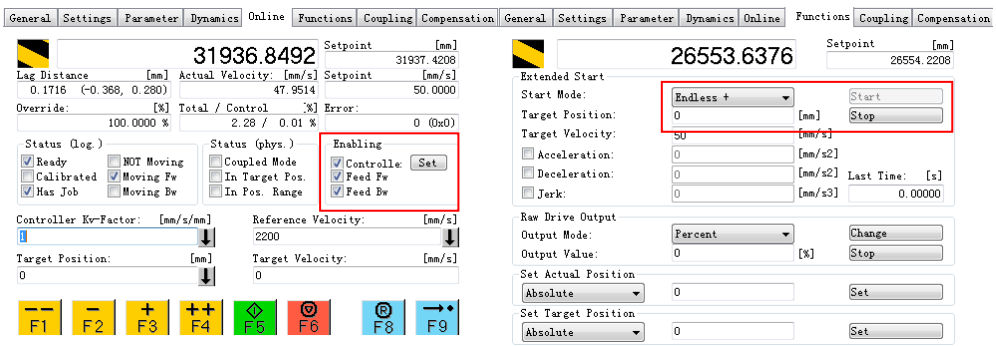
1) Drive5——Drive Manager——digital I/O 设置需要的功能，比如硬件限位，可以通过右下角的方向键来移动窗口。



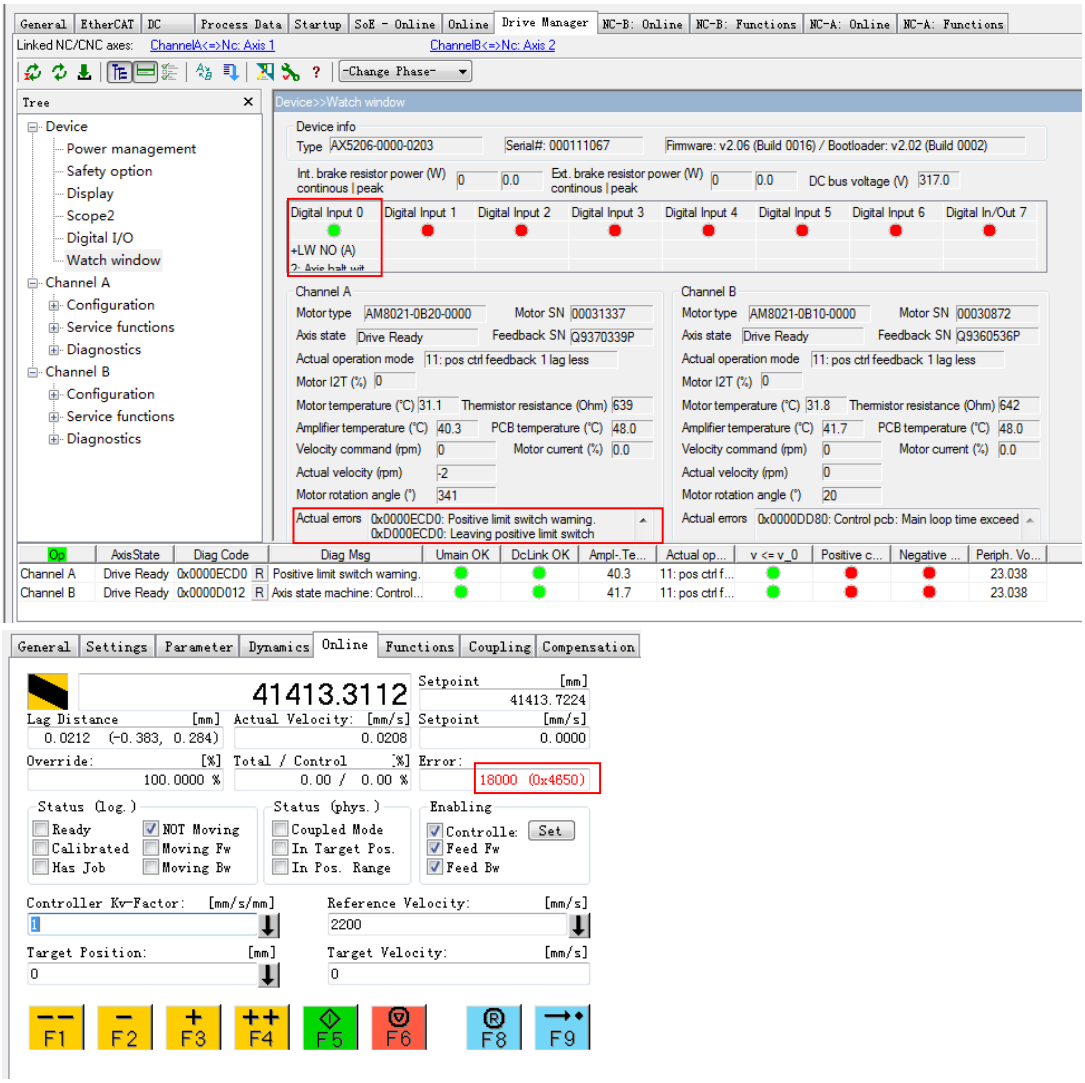
2) 选择 P-0-0401 位置限位设置，configuration 根据外部传感器类型选择，常开点或者常闭点，Limit switch reaction 设置当碰到限位信号后，驱动器的响应动作，0: E-Stop with a C1D error 驱动器报错，NC 报错，轴停止，1: E-Stop with a C2D warning，驱动器不报错，NC 报错，轴停止，2: Axis halt with a C2D warning，驱动器不报错，NC 报错，轴停止，input number 设置对应的输入点。



3) 配置完之后激活配置并重启 twincat，将轴使能并持续正转

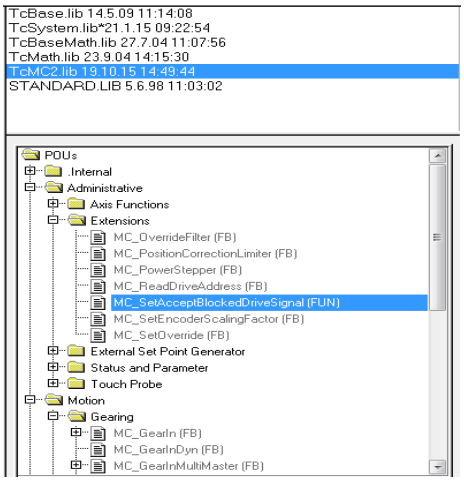


4) 此时触发极限信号代表碰到了正极限，可以看到驱动器出现警告（不报错），并停止转动，NC 报错 18000（此错误代表驱动器没有准备好）



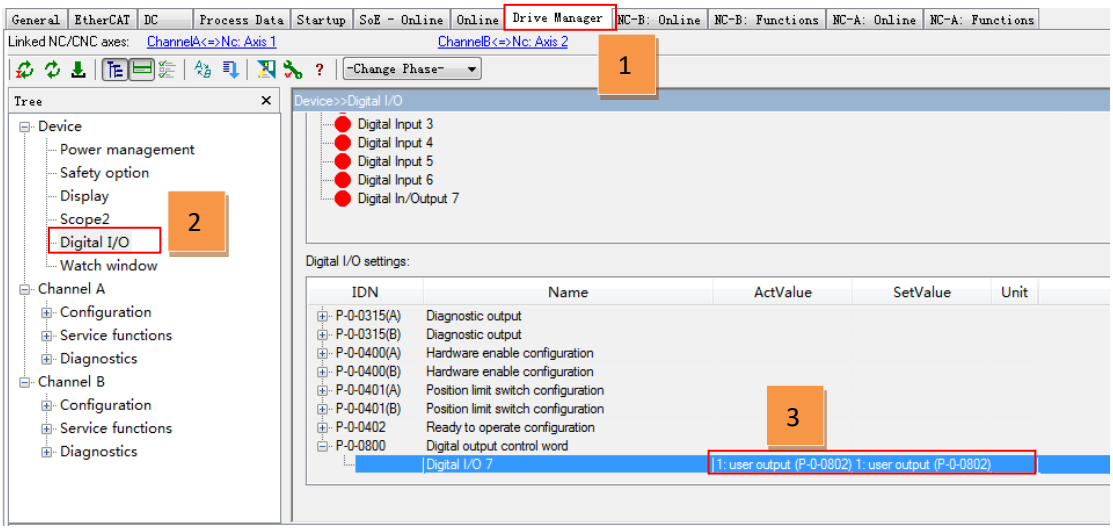
5) 复位并且反转

由于此时极限信号依然为 ON（驱动器无法处于就绪状态），NC 错误复位之后轴仍然不能反转，极限信号需要先 OFF，电机才能转动，但是这不符合现场的实际需求，客户往往希望碰到极限信号之后，可以反转，但是不能正转，此时需要使用 tcmc2.lib 库文件中的 FUNCTION AxisSetAcceptBlockedDriveSignal 来屏蔽极限信号，电机能反转不能正转。

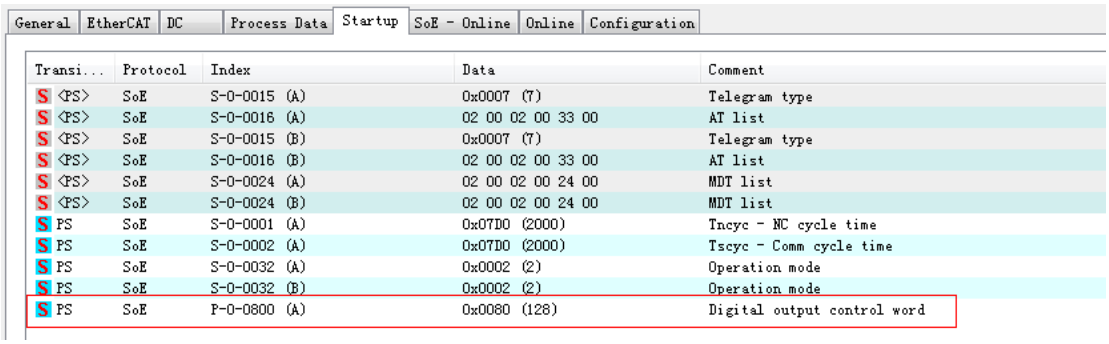


3. 驱动器输出点配置

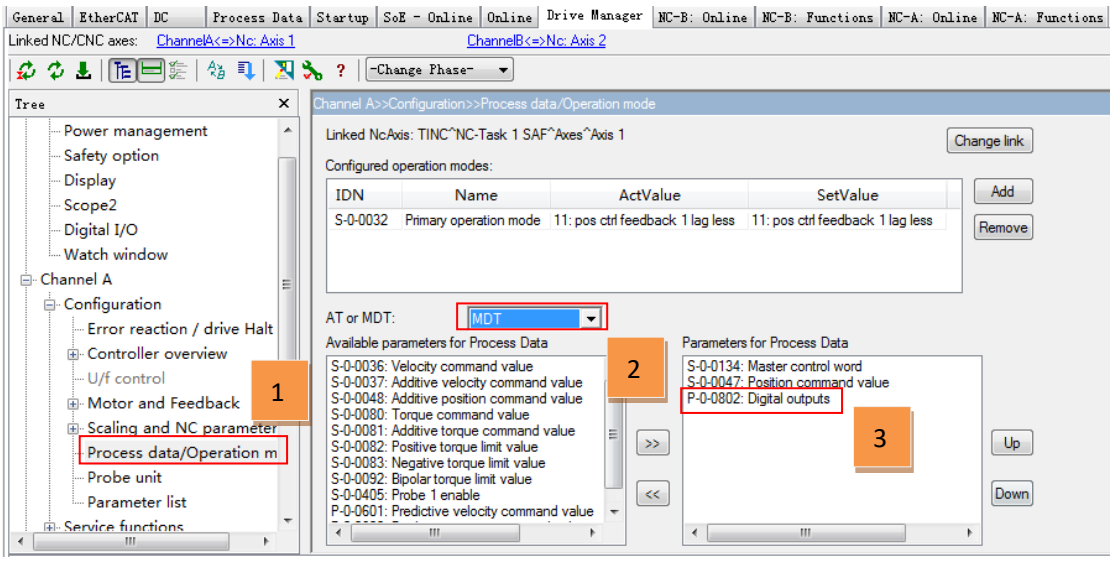
1) Digital I/O中，在P-0-0800中设为1：user output。



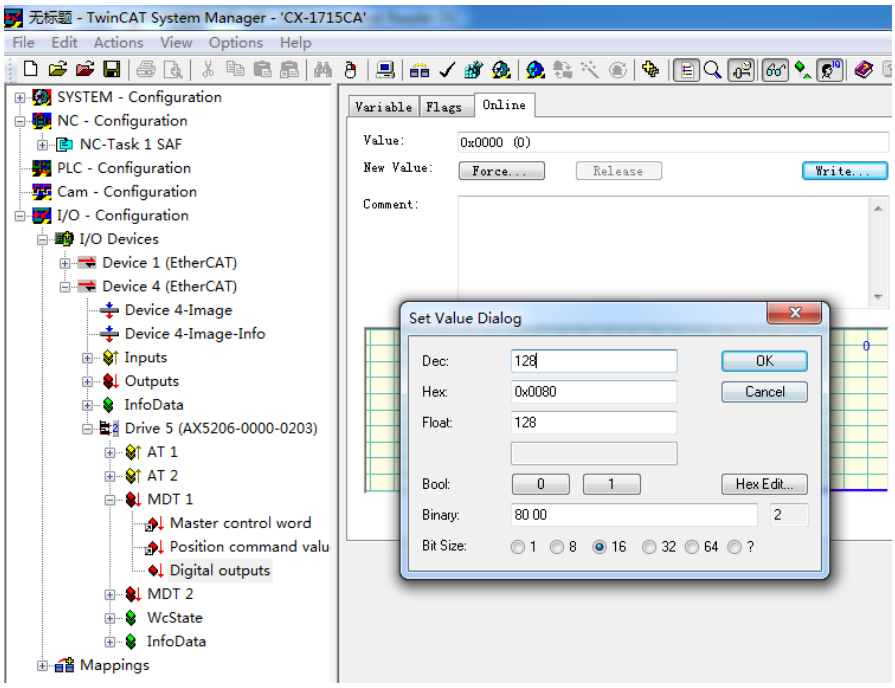
同时在 StartUP 中确认有 P-0-0800。



2) 在驱动器的Process Data的MDT1中添加Digital Outputs



3) 添加完成之后，MDT1中会出现Digital Outputs变量，激活配置并重启twincat后，将这个变量设为128（第8位为1），便可输出24V。



上海（ 中国区总部）
中国上海市静安区汶水路 299 弄 9号（市北智汇园）
电话：021-66312666　　传真：021-66315696　　邮编：200072

北京分公司
北京市西城区新街口北大街 3 号新街高和大厦 407 室
电话：010-82200036　　传真：010-82200039　　邮编：100035

广州分公司
广州市天河区珠江新城珠江东路16号高德置地G2603室
电话：020-38010300/1/2　传真：020-38010303　　邮编：510623

成都分公司
成都市锦江区东御街18号 百扬大厦2305 房
电话：028-86202581　　传真：028-86202582　　邮编：610016

请用微信扫描二维码
通过公众号与技术支持交流

倍福中文官网：
http://www.beckhoff.com.cn/
倍福虚拟学院：
http://tr.beckhoff.com.cn/

招贤纳士：job@beckhoff.com.cn
技术支持：support@beckhoff.com.cn
产品维修：service@beckhoff.com.cn
方案咨询：sales@beckhoff.com.cn