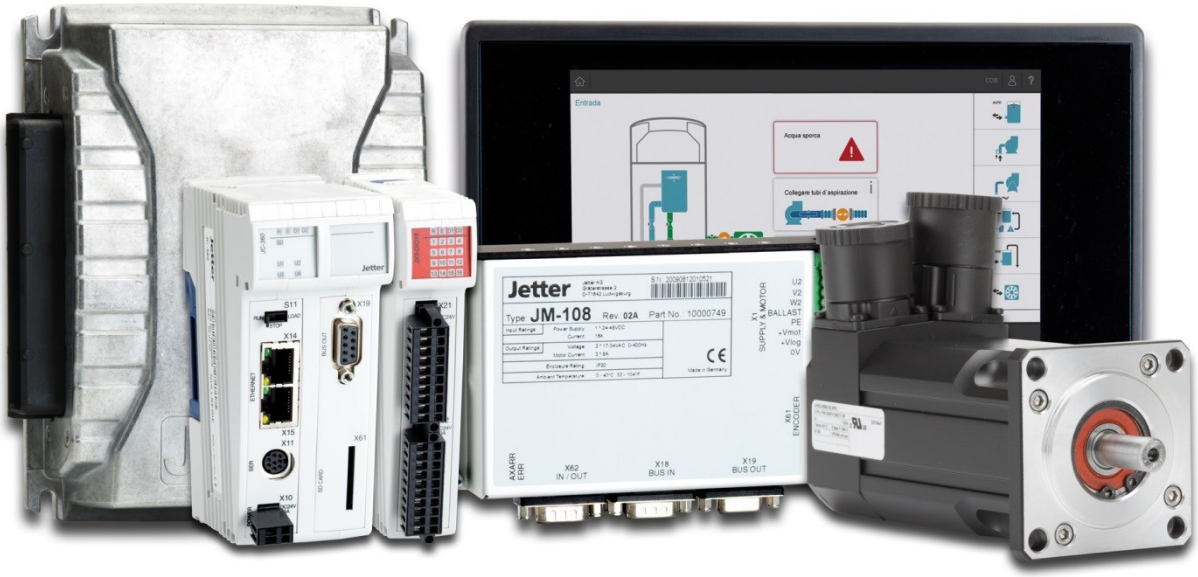
Jetter automation



MCX - 轴

应用笔记 057

608 864 47_00

We automate your success.

本文件是由 Jetter AG 基于已知的技术慎重地编制而成。产品的变更和技术开发未全部包含在修订文件中。Jetter AG 对内容和形式上的错误、缺少更新以及由此产生的损害或不利情况不承担任何法律责任。

Jetter AG
Graeterstrasse 2
71642 Ludwigsburg
Germany

www.jetter.de

Phone:

Switchboard	+49 7141 2550-0
Sales	+49 7141 2550-531
Technical hotline	+49 7141 2550-444

E-mail:	info@jetter.de
Technical hotline	hotline@jetter.de
Sales	vertrieb@jetter.de

Product name	MCX – Axes
Document type	Application Note 057

Translation of the original German language document

Document revision	1.00
Item number	608 864 47_00
Date of issue	2021-11-09

目录

1 前提条件

应用程序：

- JetSym 5.6.3
- 在本应用笔记中，假定鼠标采用右手配置：
  - 打开子菜单：右键单击
  - 双击：用鼠标左键快速单击两次
  - 选择：单击鼠标左键
- 轴和轴组的命名：
  - 基本上，用户可以为轴和轴组分配任何名称。
  - 在本应用笔记中，以下前缀用于命名目的：
    - 实轴："x"
    - 虚拟轴："xv"
    - 影子轴："xs"
    - 工艺组："tec"
    - 路径组："geo"
  - 前缀没有特殊代码效果，但可以快速向程序的读者指示正在使用的对象类型。

安全说明：

在轴的应用、调试、编程或测试过程中，运动的机械或者是电机和伺服驱动器上的电压可能会导致发生危险情况。请遵守伺服驱动器和控制组件的安全说明！

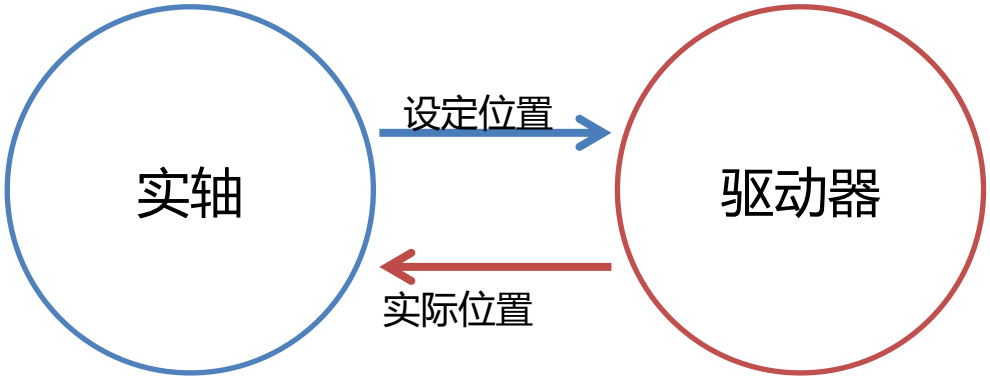
2 轴的类型

运动控制内核中可以定义 5 种类型的轴：

- 实轴
- 仿真轴
- 虚拟轴
- 外部轴
- 影子轴

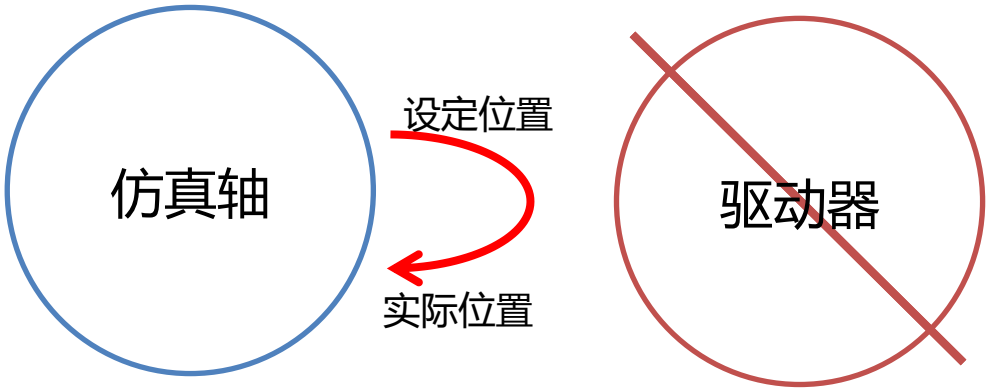
2.1 实轴

真正的轴 - 顾名思义 - 物理存在。伺服驱动器通过系统总线连接，通过总线传输设定点和命令。同时，实际值从伺服驱动器发送到运动控制内核。设定值在运动控制内核的每个循环中计算。



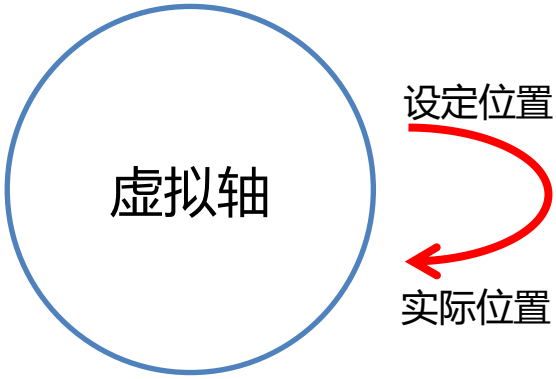
2.2 仿真轴

仿真轴是实轴的虚拟表示。由于在应用程序开发期间所需的硬件不一定连接到控制器，因此可以将实轴重新配置为仿真轴。这意味着这些过程也可以在没有伺服驱动器的情况下投入运行。



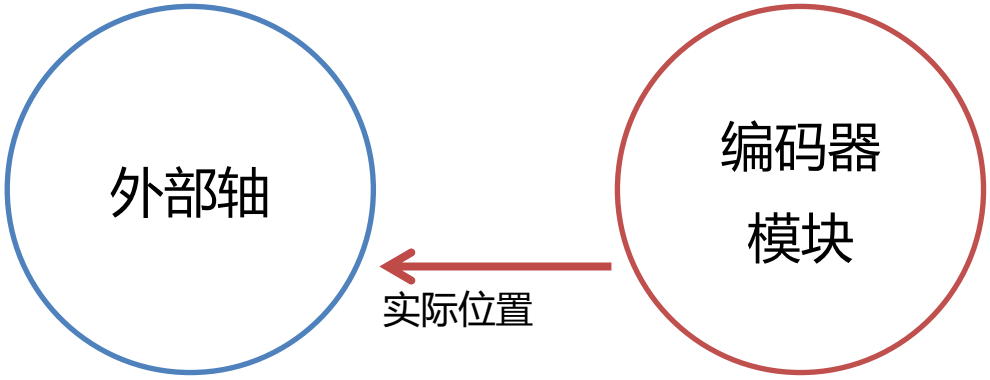
2.3 虚拟轴

虚拟轴仅用于运动控制内核，与伺服驱动器无关。它通常在工艺组中作为主轴。



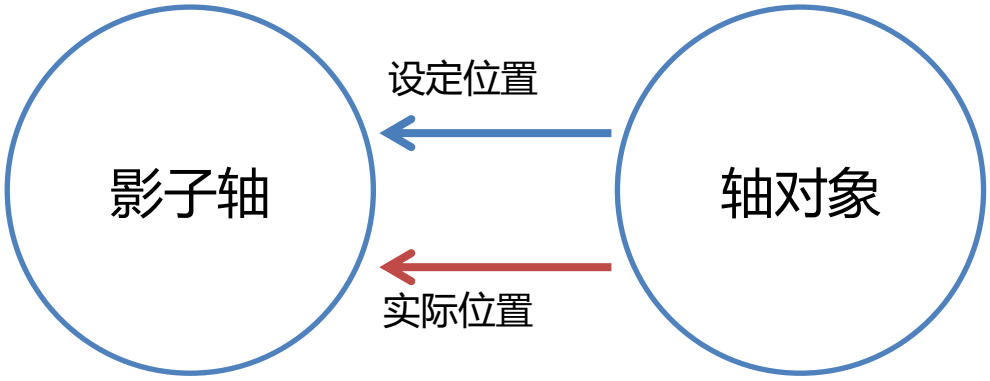
2.4 外部轴

例如，对于需要测量编码器轮的值的应用，则使用外部轴。这通常使用编码器模块或伺服驱动器的附加编码器输入。系统只检测模块发送的实际值。不会为此轴生成设定位置。
该轴的运行状态始终为"inactive"！

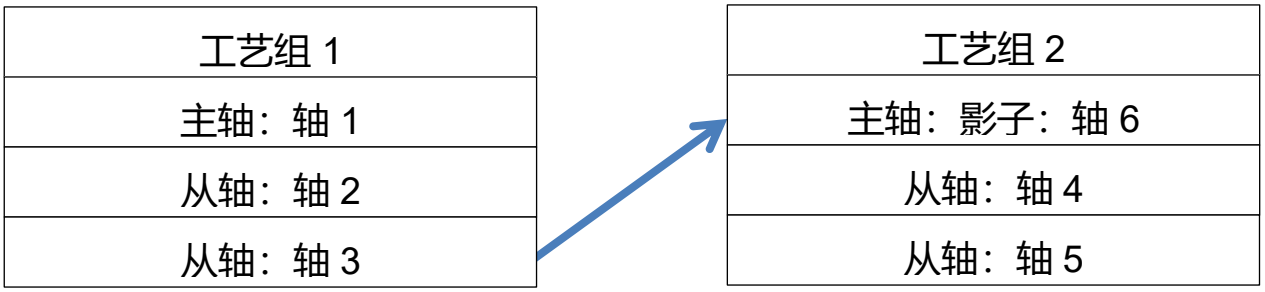


2.5 影子轴

影子轴是外部轴和内部轴的混合。影子轴链接到不同类型的轴对象。因此，该轴继承链接轴的设定位置和实际值。



影子轴始终是工艺组中的主轴。当源轴对象不能作为某个工艺组的主轴时，则使用影子轴，例如，因为该源轴对象本身是另一个组中的从轴，或者因为它要独立于某个工艺组使用。



在本例中，轴 3 是工艺组中的从轴。但是，由于轴 4 和 5 将跟随轴 3，因此使用影子轴（轴 6）作为轴 3 的表示。由于轴一次只能在一组中处于活动状态（它可以是多个组的成员，但一次只能在一组中激活），轴 3 可以独立于工艺组 2 操作。

3 机械设计

轴的应用和编程主要集中在操作点上。因此，运动控制内核必须知道机械是如何设计的。

3.1 线性 - 常规

线性常规机构，直线运动，有起点和终点，例如线性轴。

3.2 线性 - 模态，方向：双向

线性机构具有直线运动，可以无休止地连续移动，例如传送带。

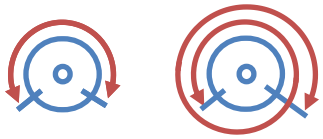
3.3 线性 - 模态，方向：单向正向

线性机构只能沿正方向移动

3.4 线性 - 模态，方向：单向负向

线性机构只能沿负方向移动

3.5 旋转 - 常规



一种在有限路径中绕圈转动的机构。行程不限于模数范围。

3.6 旋转 - 模态，方向：双向

一种可以在两个方向上无限连续旋转的机构。

3.7 旋转 - 模态，方向：单向正向

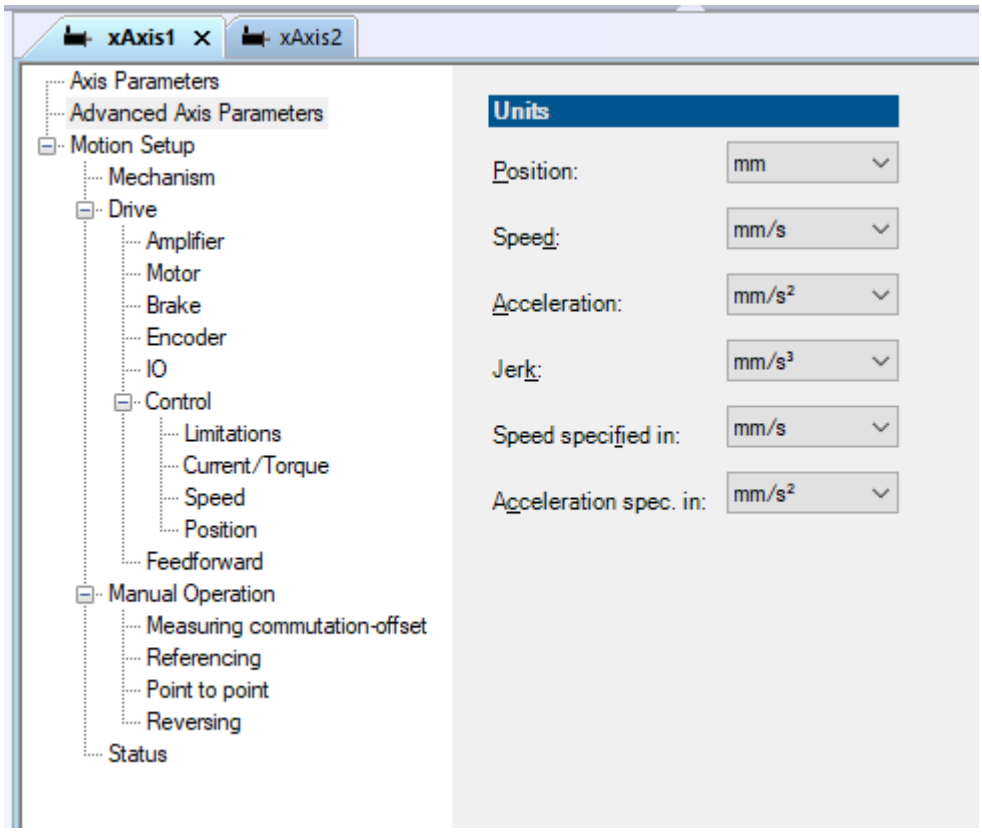
一种在圆周上无限转动的机构，只能沿正方向移动。

3.8 旋转 - 模态，方向：单向负向

一种在圆周上无限转动的机构，只能向负方向移动。

4 机械计量单位

可以为每个轴单独指定使用的计量单位：



这些单位构成了运动控制内核如何解释相应值的基础。

例如：

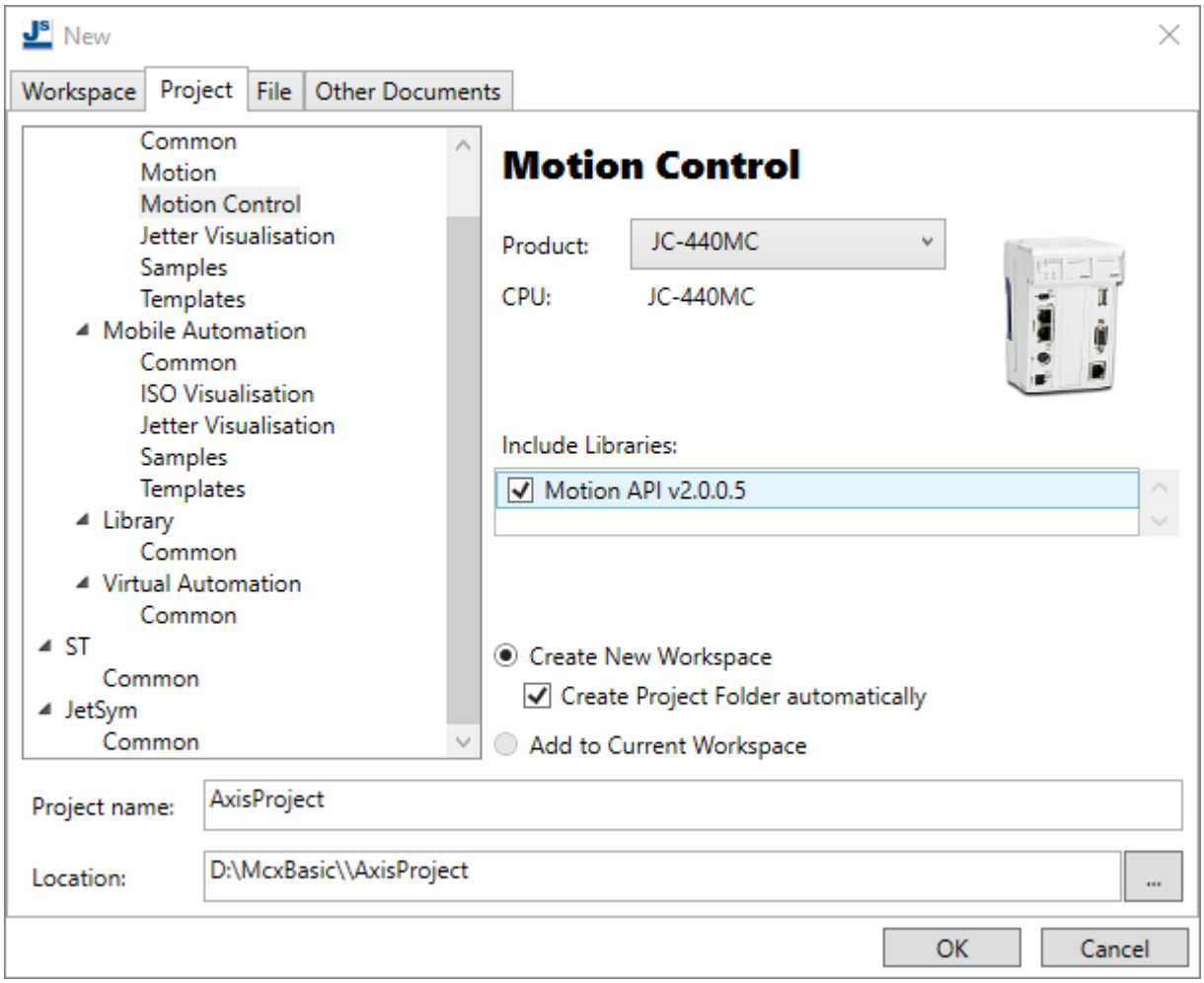
设置单位	数值	速度
mm/s	100	100 mm/s
m/s	100	100,000 mm/s

建议使用标准 SI 单位。

5 轴的应用

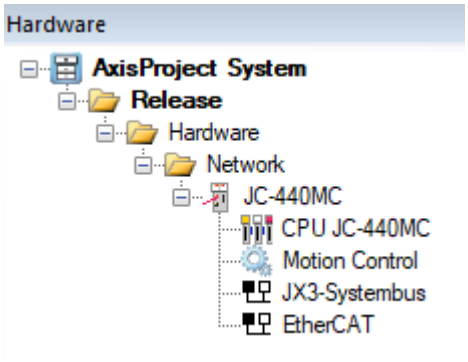
5.1 创建 MCX 项目

作为各个轴类型表示的基础，我们创建了一个项目，我们逐渐向其中添加不同的轴或对其进行配置。在 JetSym 中，通过"File -> New"打开项目向导。



选择一个 MCX 控制器并包含 Motion API。JetSym 始终自动推荐当前安装并适合所选控制器的版本。单击"OK"后，将创建项目树。

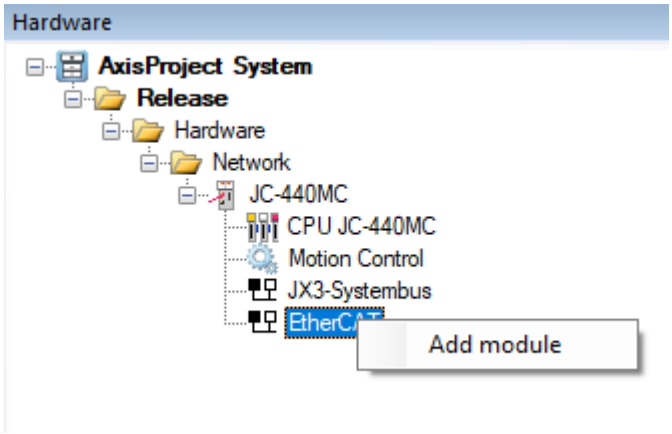
在硬件管理器中，控制器和基本节点现已自动创建，可供进一步配置。



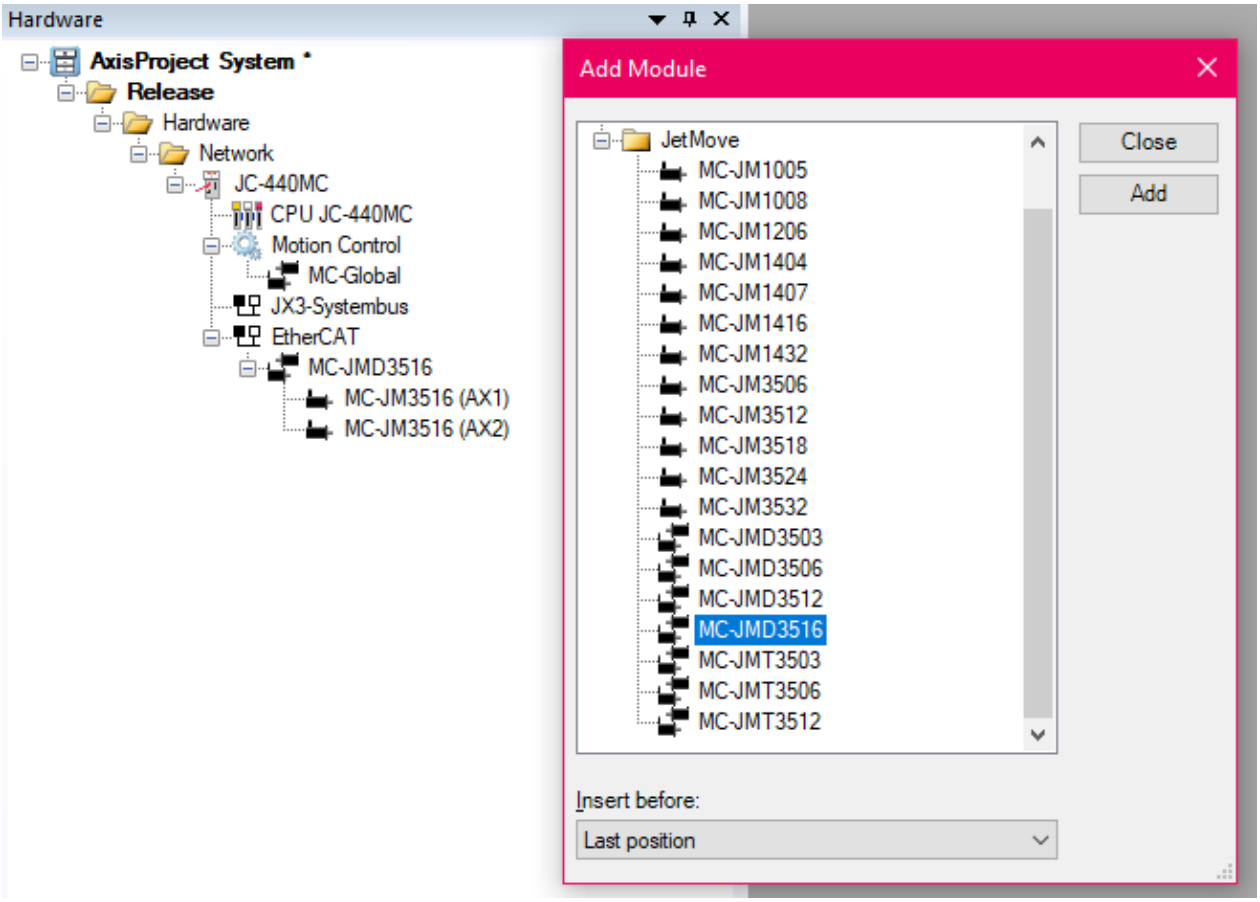
5.2 实轴

5.2.1 添加伺服驱动器

现在，您可以添加 EtherCAT 伺服驱动器。



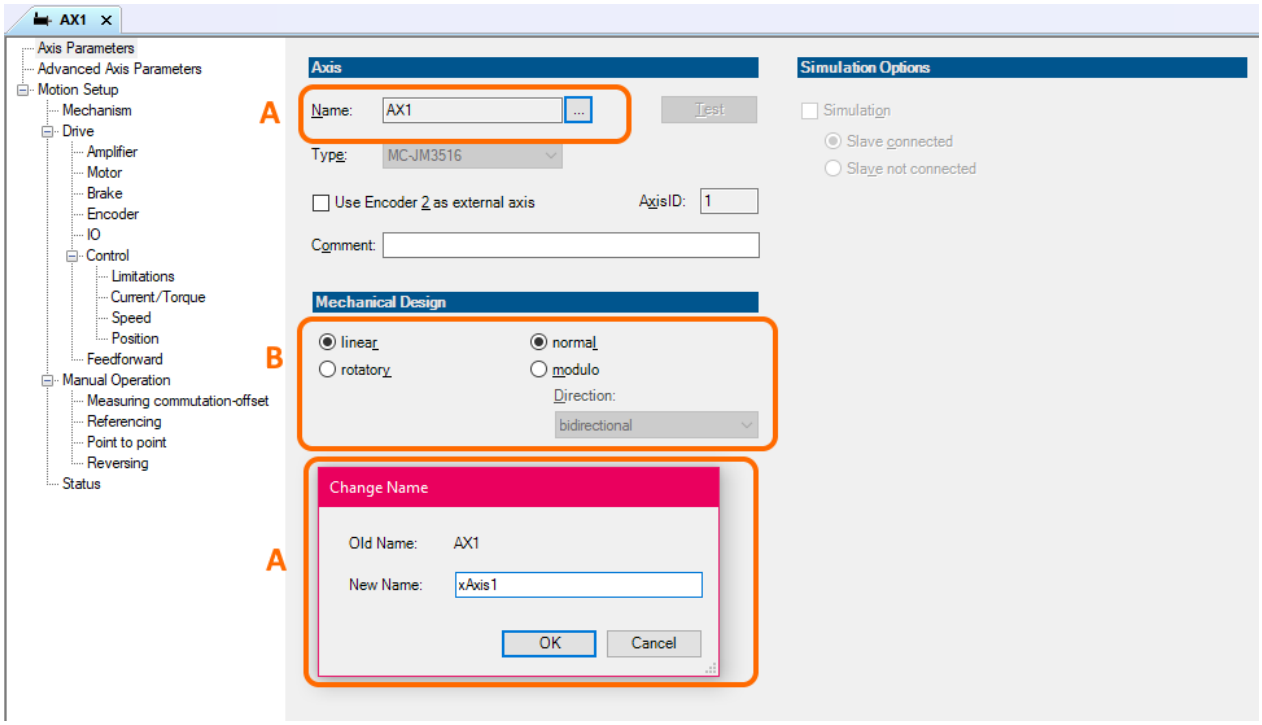
"EtherCAT"节点上的子菜单允许您通过"Add module"选择要添加到该节点的模块。



现在选择所需的伺服驱动器并通过双击或单击"Add"将它们添加到硬件配置中。

5.2.2 配置实轴

双击轴对象打开运动设置 - 配置和调试对话框。



在本例中，轴"AX1"的运动设置处于打开状态。

在 Motion Setup 开始页面上，可以定义轴对象的基本设置。

A. 建议指定一个对应用有意义的描述性名称，而不是建议的轴名称“AX1”。该名称还用于在 STX 程序中使用相应的轴对象。

   此对话框可让您更改轴名称。

B. 通过"Mechanical Design"对话框，您可以指定是旋转运动还是直线运动，以及轴是模态轴还是常规轴。

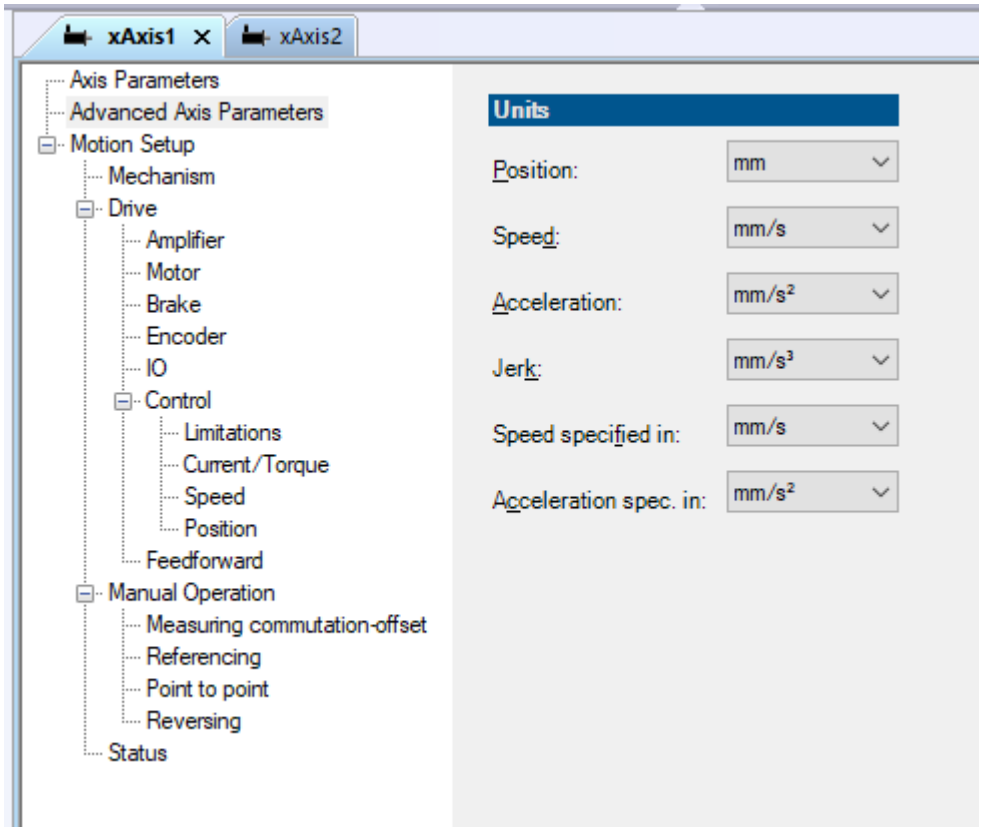
   - 常规轴：该轴具有无法超出的固定行程范围。

   - 模态轴：超出模数范围时，轴的位置跳回到起始位置。

   通常应该根据工作点的机械运动状态来设定此处。

5.2.3 轴的高级参数

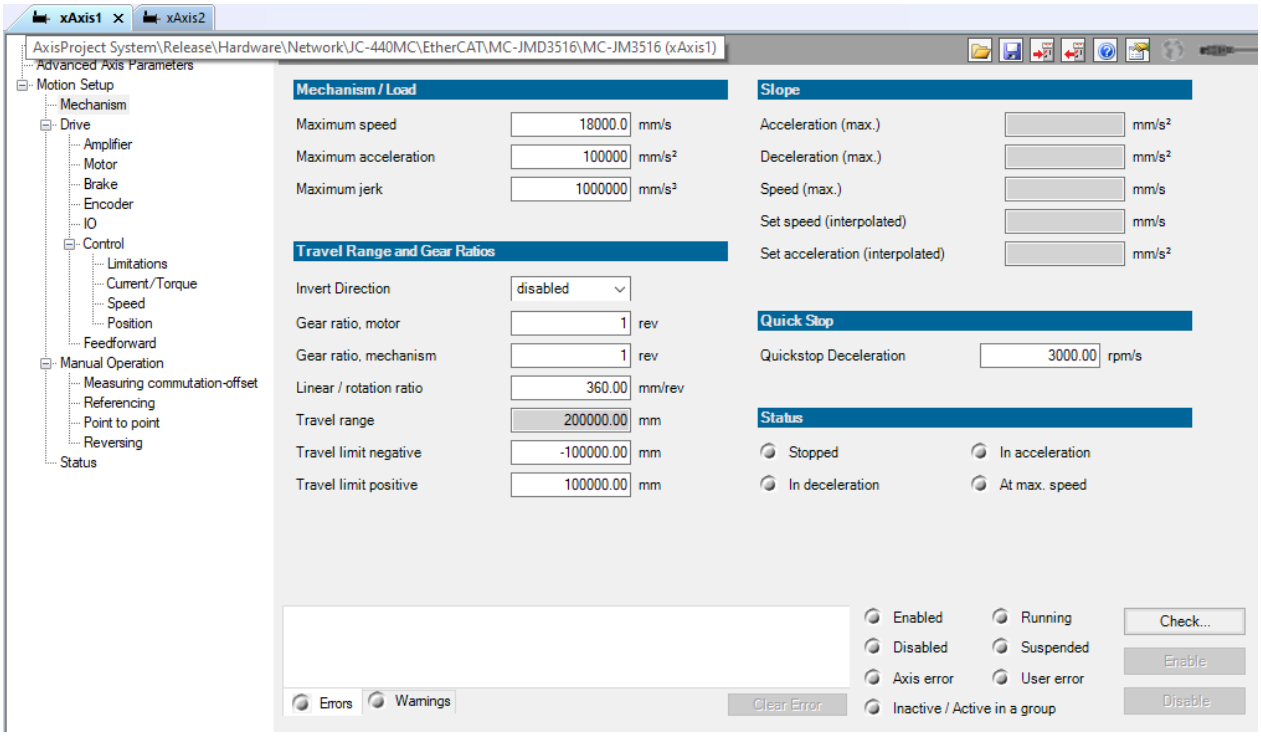
"Advanced Axis Parameters"对话框允许您设置用于轴的单位。



请参考机械计量单位

5.2.4 机械参数

运动设置页面"Mechanism"允许您预设一些重要参数，用于设定值计算。



这些参数定义了机械条件：

- 该机械结构可以/可能移动多快?
- 加速度多大是合适的?
- 旋转方向是否需要改变，例如当电机沿正方向运动时，机构沿逻辑负方向运动?
- 传动比是多少?
- 行程范围多大?

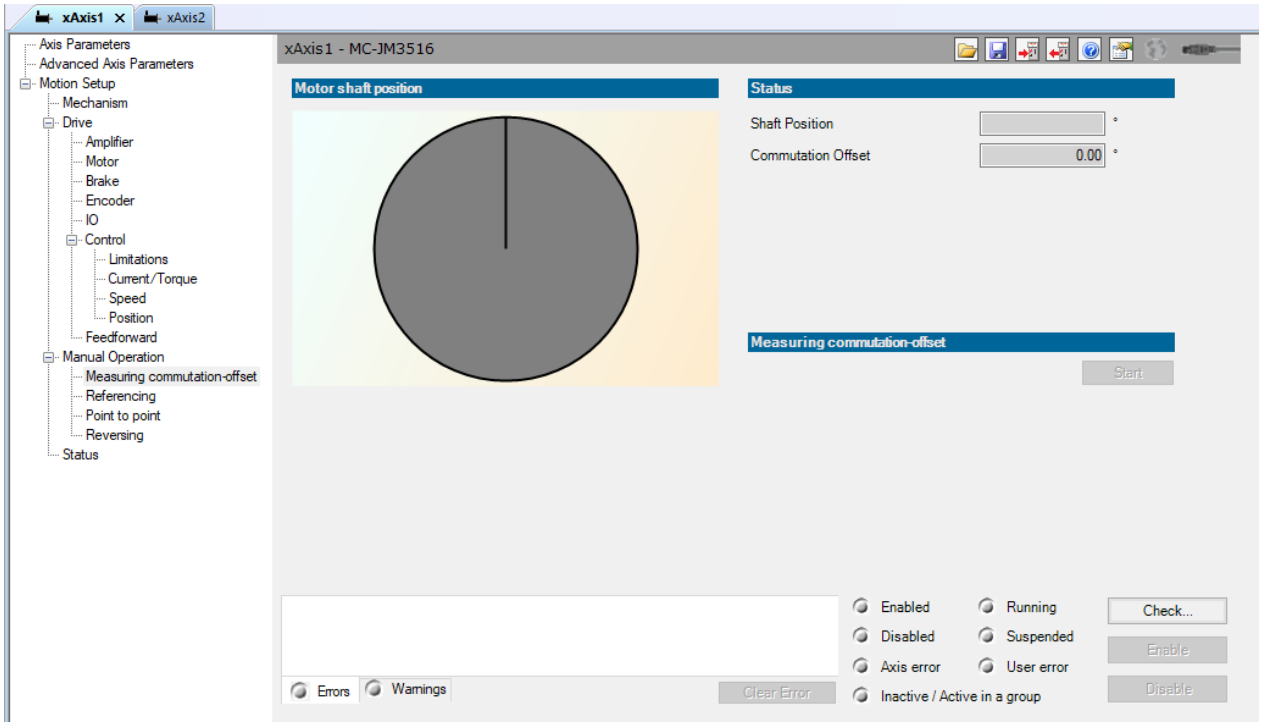
5.2.5 调试轴

更多的运动设置页面可用于调试轴。调试步骤请参考伺服驱动器的用户手册。

5.2.6 手动操作

5.2.6.1 测量换向角

该对话框允许您测量换向以确定换向角。



Jetter 伺服电机的标准换向角度为 0°。但是，对于其他制造商的电机，此角度可能不同。换向角可以通过这个页面测量来确定。

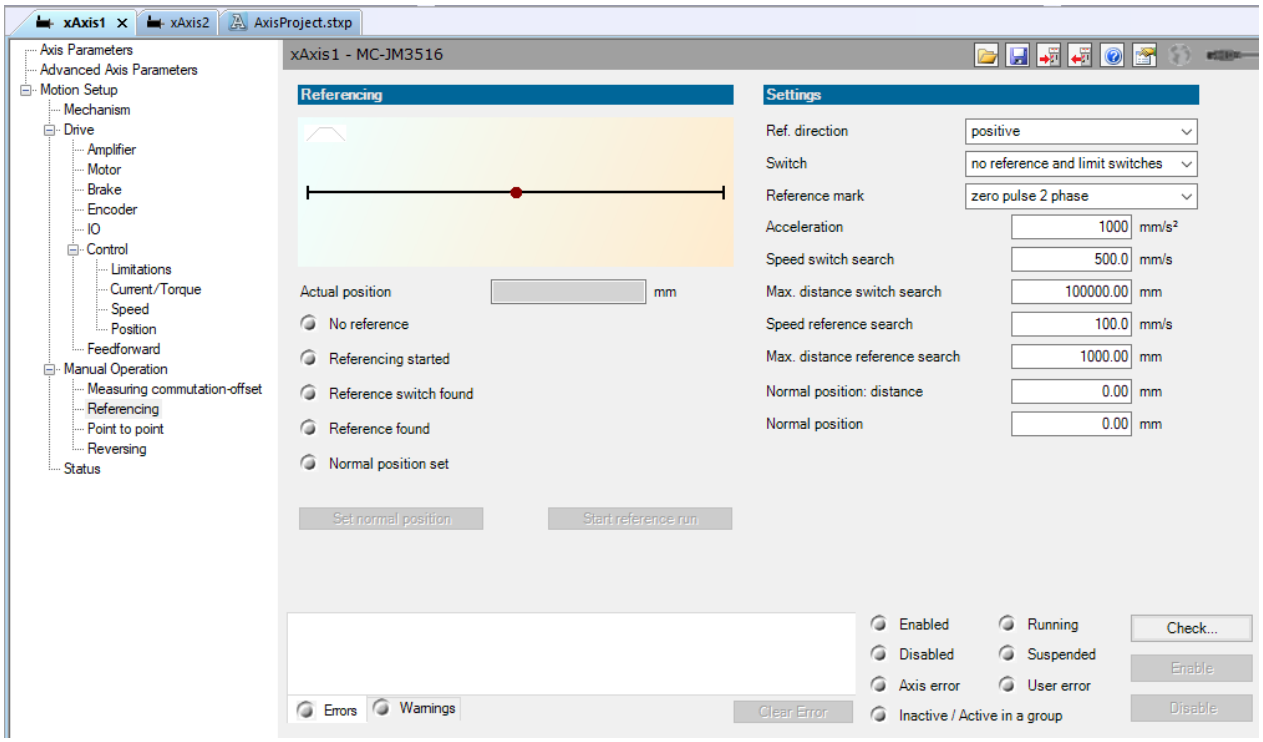
确定的换向角保存在轴的 .ini 文件中，并在控制器重新启动时加载。

也可以通过应用程序启动换向测量：

```
xAxis1.Drive.Motor.MeasureCommutationOffset(DriveCommOffsetMeasurementModes.DynamicMode);
when xAxis1.State.IsDisabled continue;
```

5.2.6.2 参考点

此页面允许您配置和测试参考点（回零）。



设置的机械参考点参数保存在轴的.ini 文件中，并在控制器重新启动时加载。
如果在没有传递参数的应用程序中启动参考点运动，将会使用 .ini 文件中的参数，前提是参考点参数没有被应用程序覆盖
注意：
在不带传递参数的情况下开始参考点运动，将使用最后设置的参数。如果参考点参数已在应用程序中更改，则使用更改的值。

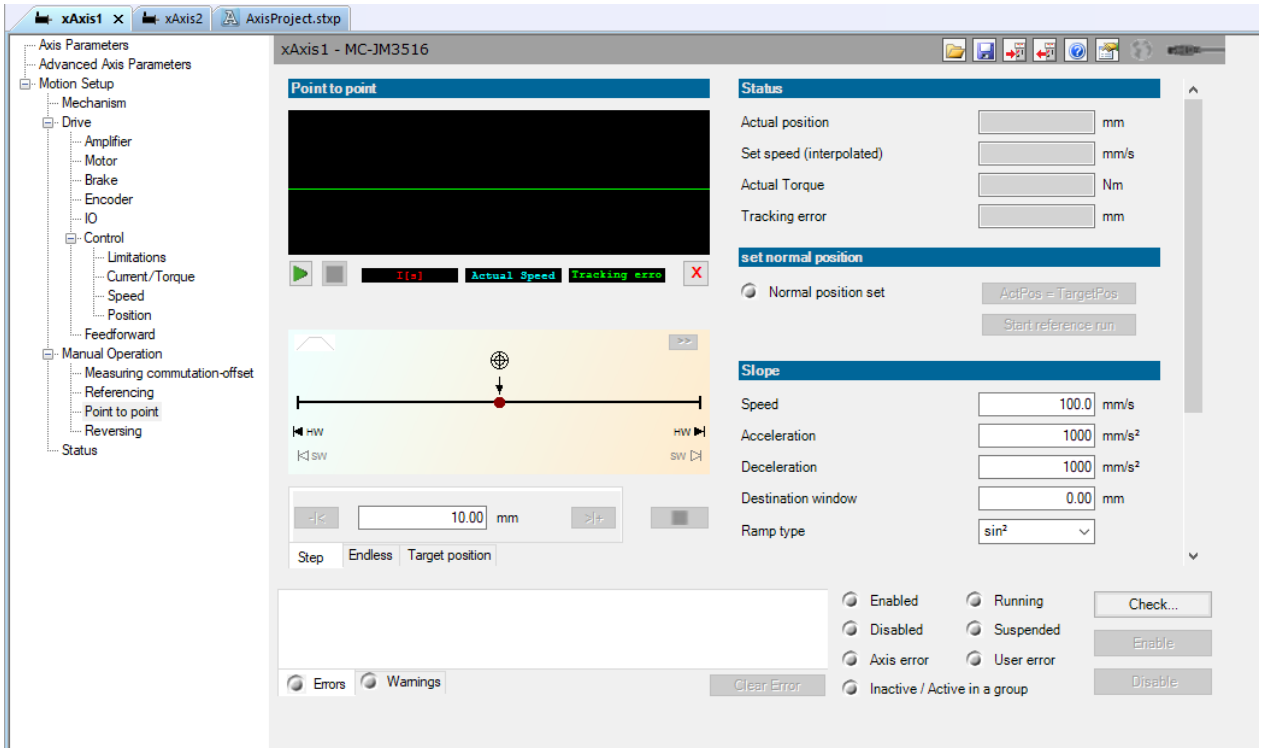
```
xAxis1.MoveHome.Start();
```

也可以在应用程序中单独设置参考点参数。

参考点参数和选项的详细说明，请参考伺服驱动器的用户手册和 JetSym 帮助。

5.2.6.3 点到点定位

此页面允许您配置和测试点到点定位。



设置的点到点参数保存在轴的 .ini 文件中，并在控制器重新启动时加载。

如果在没有传递参数的应用程序中启动点到点定位，将会使用 .ini 文件中的设置参数，前提是点到点参数未被应用程序覆盖。

注意：

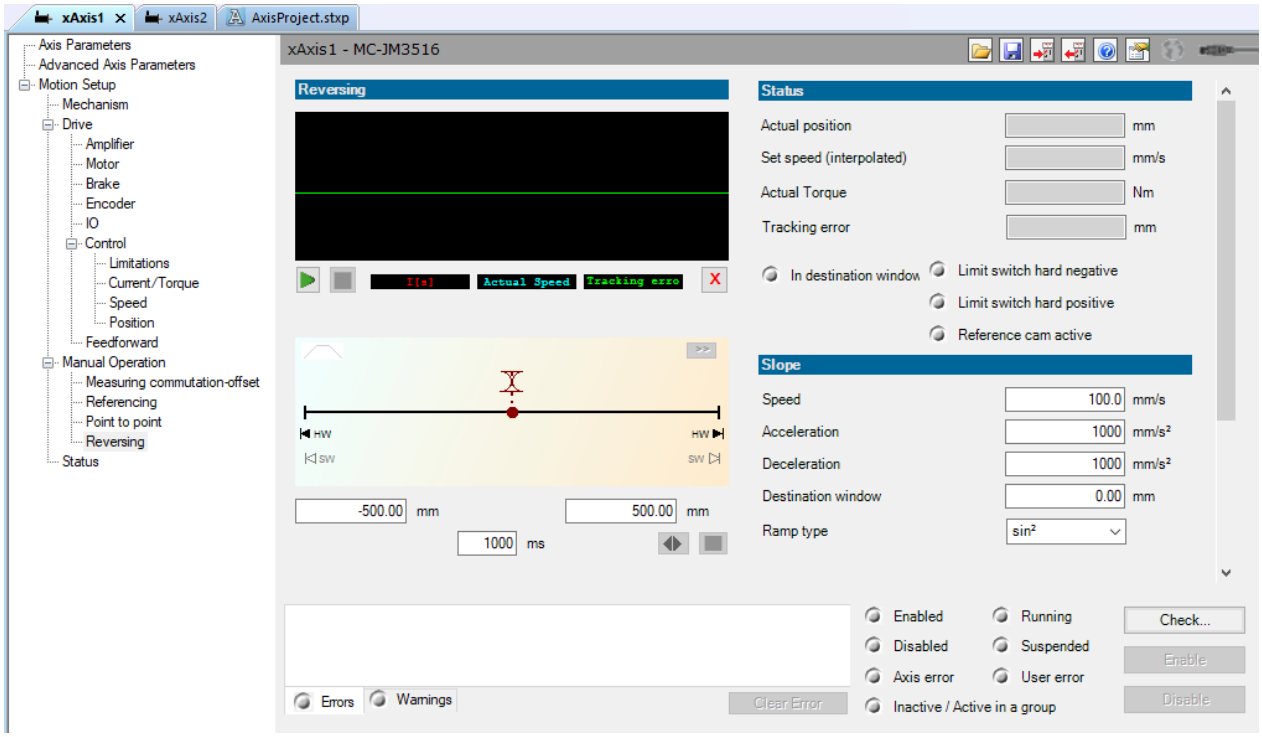
在不带传递参数的情况下开始点到点定位，将使用最后设置的参数。如果点到点参数在应用程序中已更改，则使用更改的值。

```
xAxis1.MovePtp.Start();
```

但是，点到点参数也可以在应用程序中单独设置。

5.2.6.4 往返运动

此页面可让您设置往返模式参数。往返运动常用于调整控制参数。



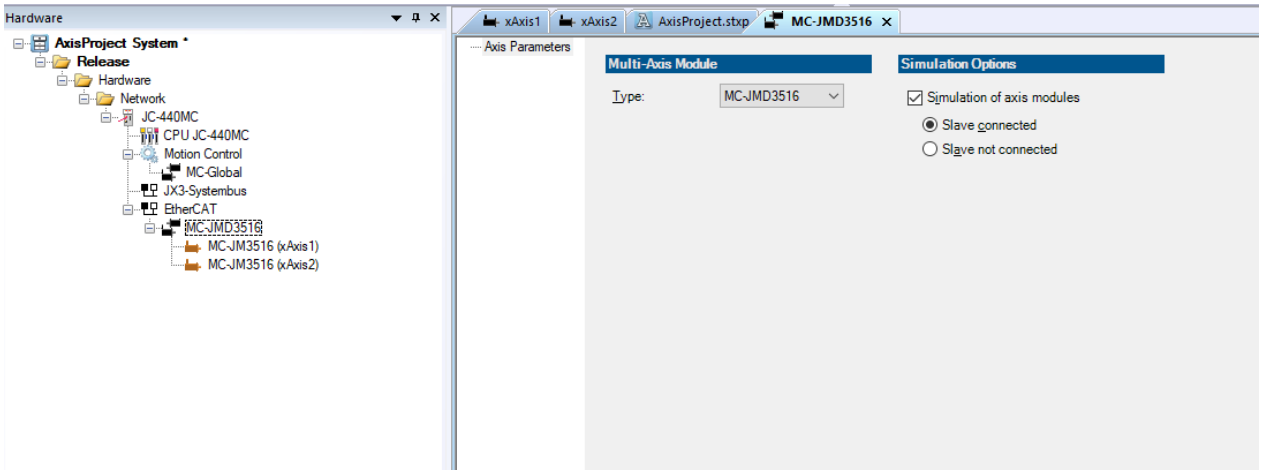
速度、加速度、减速度、目标窗口、斜坡类型等设置参数与点到点参数相同。这些参数保存在轴的 .ini 文件中，并在控制器重新启动时加载。

应用程序中没有单独的往返运动模式，但可以相应地进行编程，并且编程很简单，例如像这样：

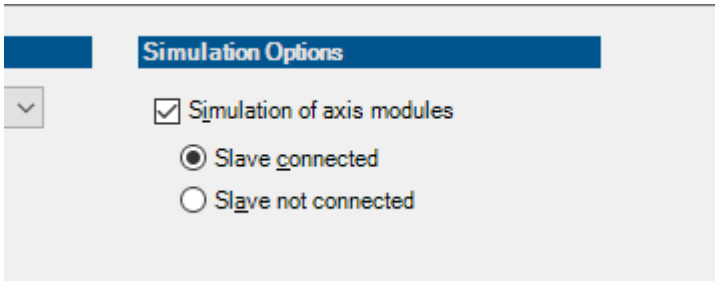
```
loop
       xAxis1.MovePtp.Start(,Target1);
       when xAxis1.State.IsEnabled continue;
       xAxis1.MovePtp.Start(,Target3);
       when xAxis1.State.IsEnabled continue;
end_loop;
```

5.3 仿真轴

仿真轴的使用可以使用户可以在没有或仅部分连接硬件的情况下开发应用程序。
如果机器不可用或只有部分可用，软件可以仿真缺少的驱动器。这使您可以在没有危险的情况下测试正在开发的过程或进行检查。为此，在相应伺服驱动器的轴参数中选择"Simulation"选项。

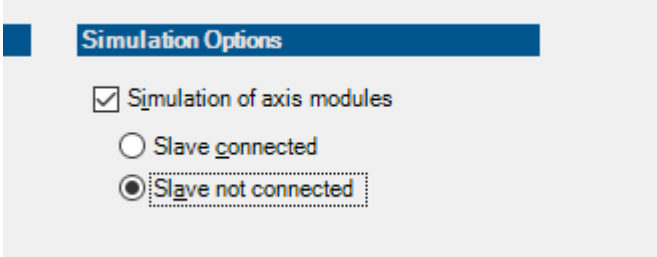


注意：
只能仿真整个模块。如果使用多轴驱动器，仿真选项设置在上级节点，适用于该模块的所有轴。
如果伺服驱动器已连接，选择"Slave connected"选项。



此选项很有用，例如驱动器硬件已连接但是未连接电机或在测试期间不得移动机械装置。

如果没有硬件连接，选择"Slave not connected"选项。

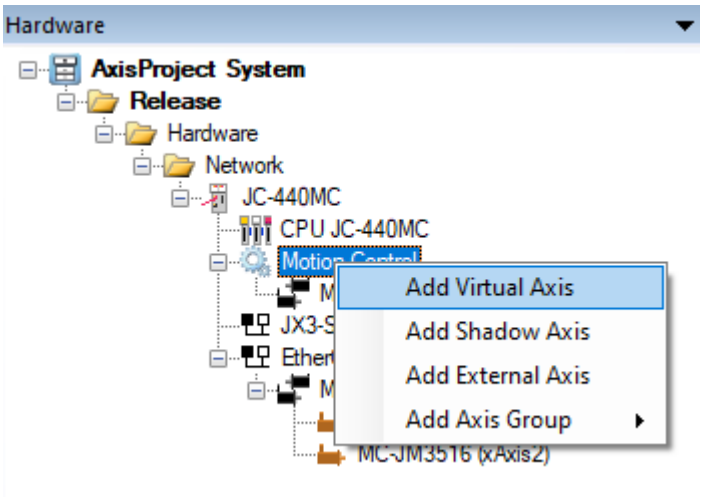


这个区别很重要，因为对于系统总线的初始化，必须明确连接了哪些模块。

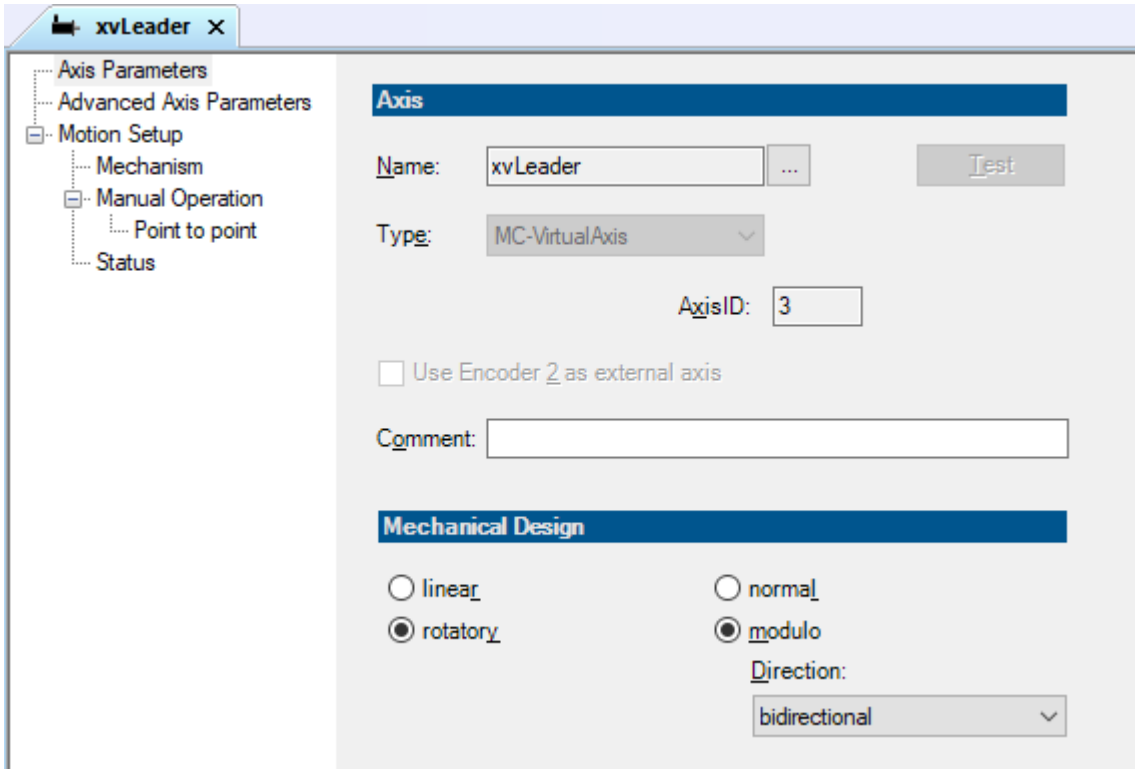
应用：
在应用程序中使用仿真轴，就像实轴一样。但是需要注意的是，不能直接访问轴。例如，如果读取了直流母线电压，将不会返回预期的结果。
外部反应也可能不存在。例如，真实机器在移动的过程中会触发一个光栅，这在仿真中不会发生。

5.4 虚拟轴

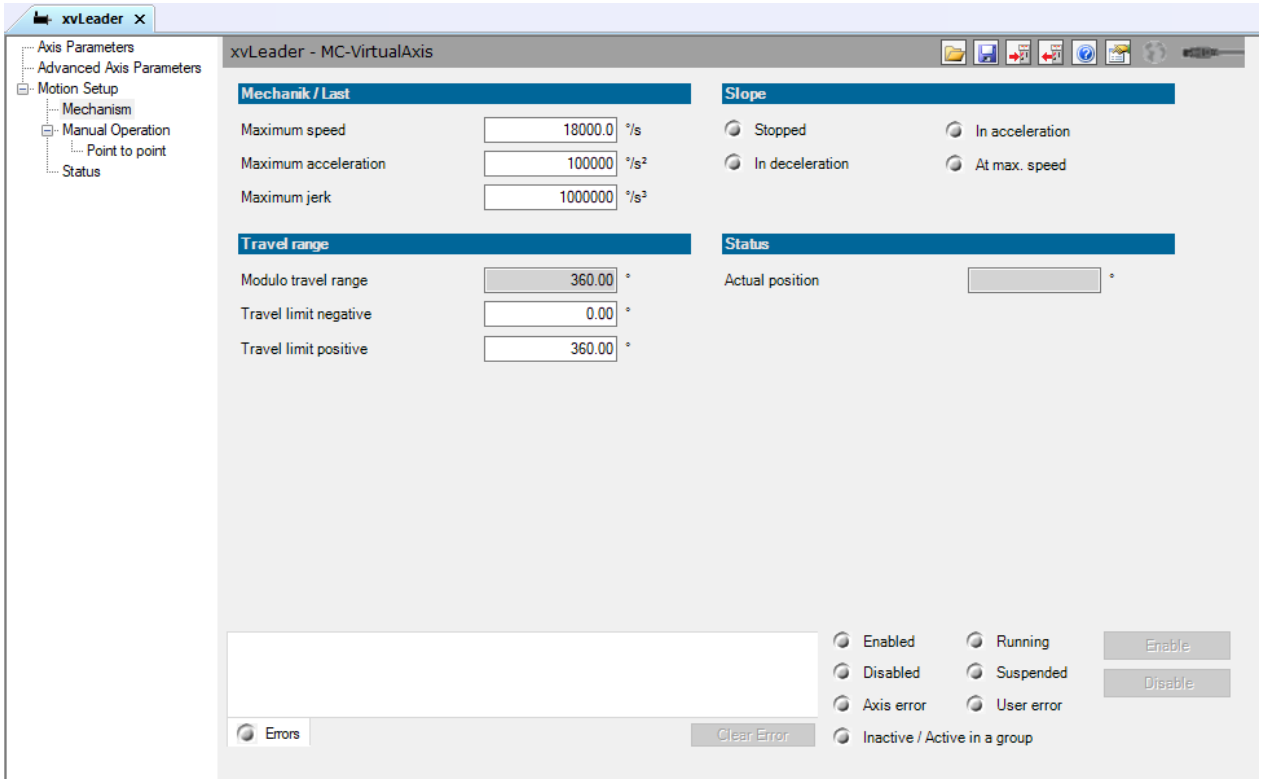
要将虚拟轴添加到硬件配置中，请转到运动控制节点并选择"Add virtual axis"。



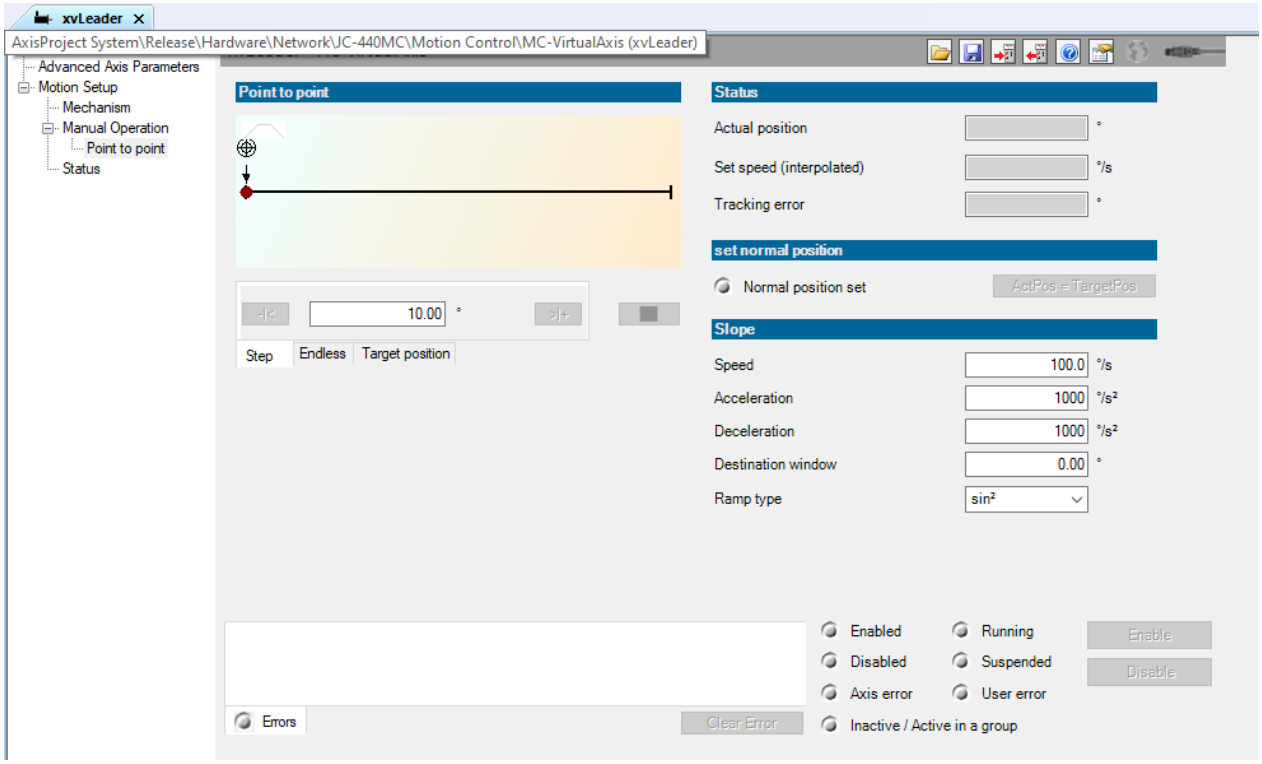
虚拟轴的配置方式与实轴相同，只是该轴没有硬件，因此不需要设置电机参数。



在配置过程中，选择所需的机械设计并设置行程范围和最大值。



出于测试目的，虚拟轴也可以通过运动设置进行定位运动。



设置的点到点参数保存在轴的 .ini 文件中，并在控制器重新启动时加载。

如果在没有传递参数的应用程序中启动点到点定位，将会使用 .ini 文件中的设置参数，前提是点到点参数未被应用程序覆盖。

注意：

在不带传递参数的情况下开始点到点定位，将使用最后设置的参数。如果点到点参数在应用程序中已更改，则使用更改的值。

```
xvLeader.MovePtp.Start();
```

但是，点到点参数也可以在应用程序中单独设置。

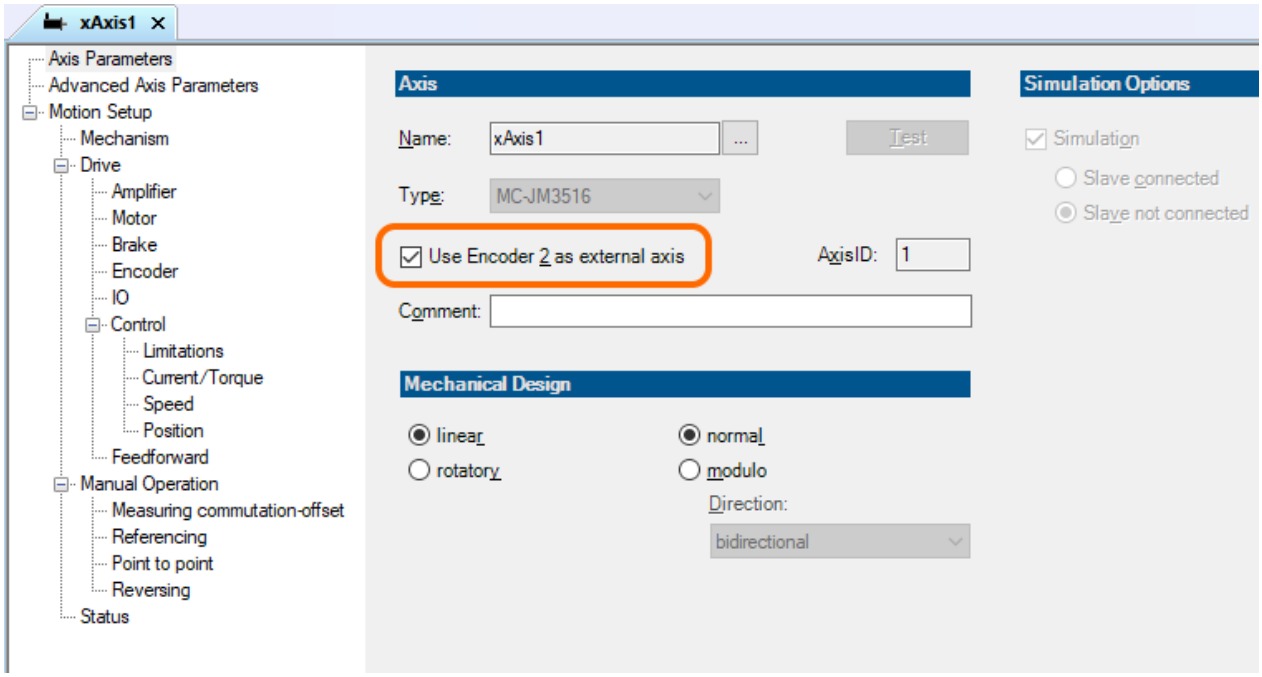
5.5 外部轴

要添加外部轴，请转到"Motion Control"节点并选择"Add external axis"。

5.5.1 带有 EtherCAT®驱动器的 JC-440Ext：

如果带有附加编码器输入的 EtherCAT®驱动器，操作如下：

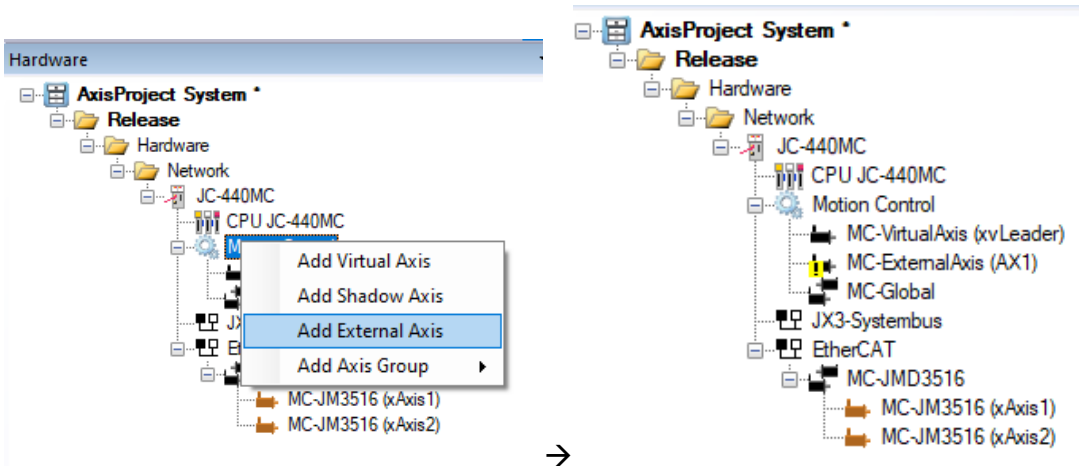
第 1 步：



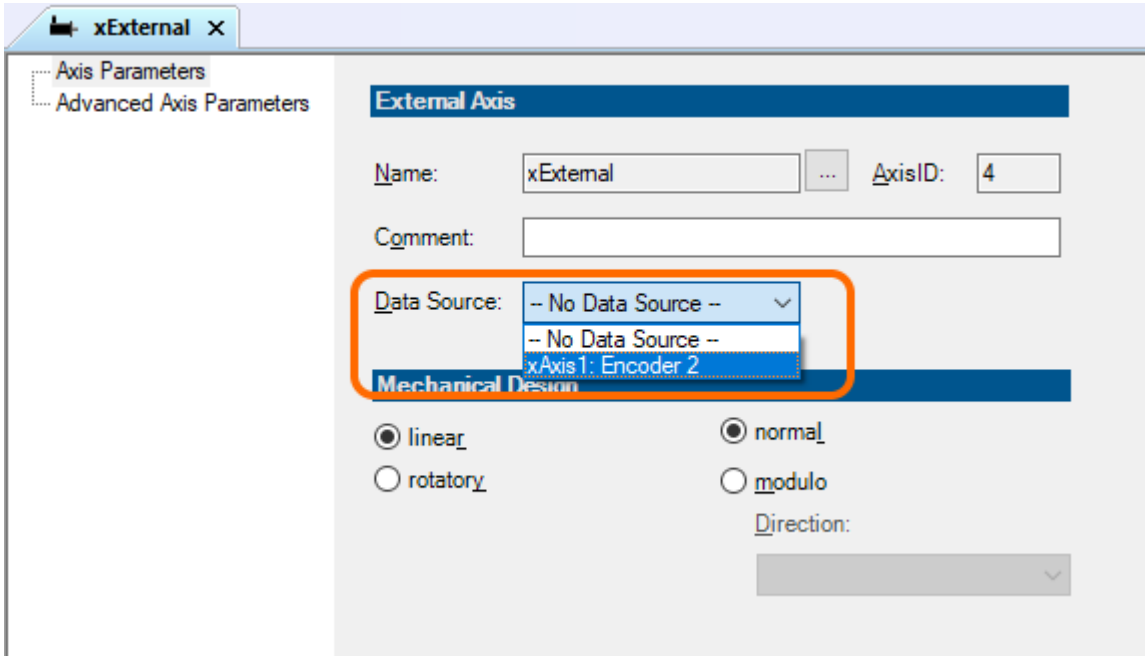
必须明确选择编码器输入 2 用于外部轴。

这意味着该输入不能用作第二个编码器用于控制实轴。

第 2 步：



在"Motion Control"节点的快捷菜单中选择"Add External Axis"。

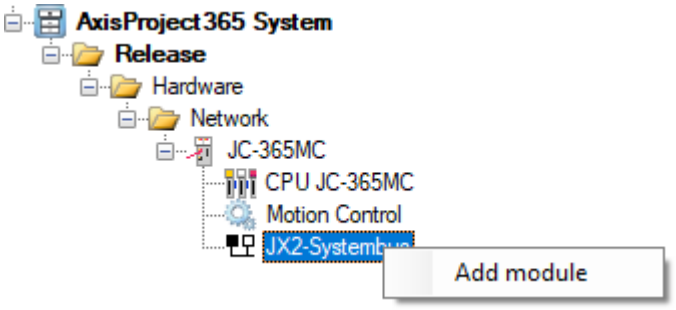


由于外部轴的编码器在步骤 1 中启用，现在可以在此处选择该编码器并将其分配给该轴对象。"Mechanical Design"也可以在这里设置。这将确定你将要使用线性轴还是旋转轴，是一个常规轴或者是模态轴。

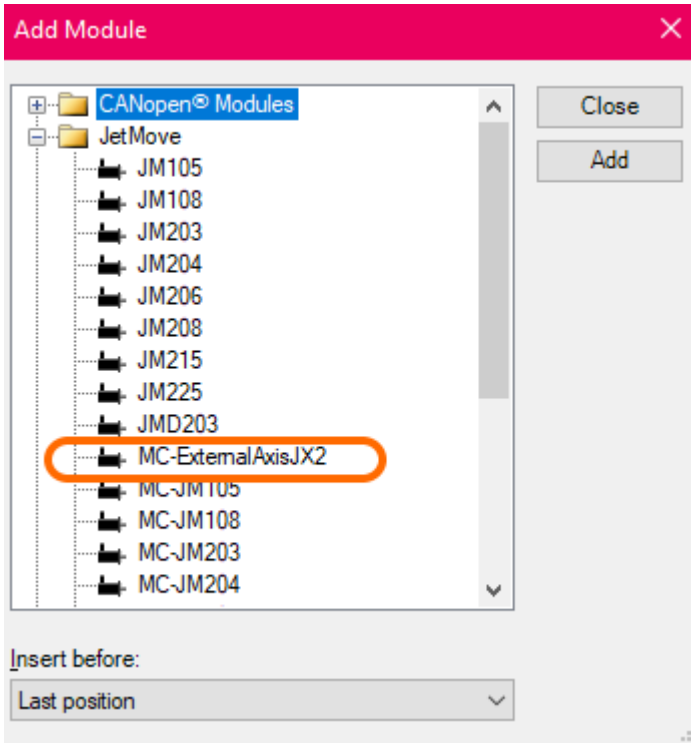
更多详细的信息请参考第三章：3. 机械设计

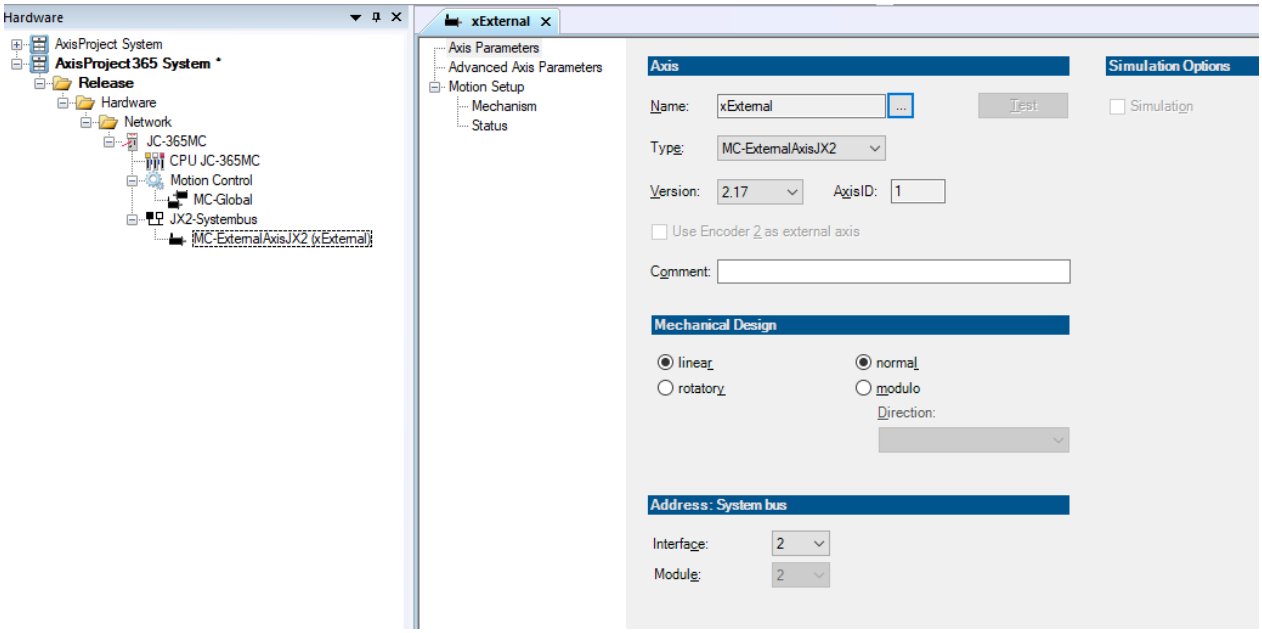
5.5.2 JC-365MC:

对于 JM-200 和 JM-105/108，如果带有第二个编码器输入，不可以用作外部轴。要为外部轴获取额外的编码器信号，JX2 系统总线上需要 JM-200 或 JM-105/JM-108。



为此，将"MC-ExternalAxisJX2"模块添加到系统总线。

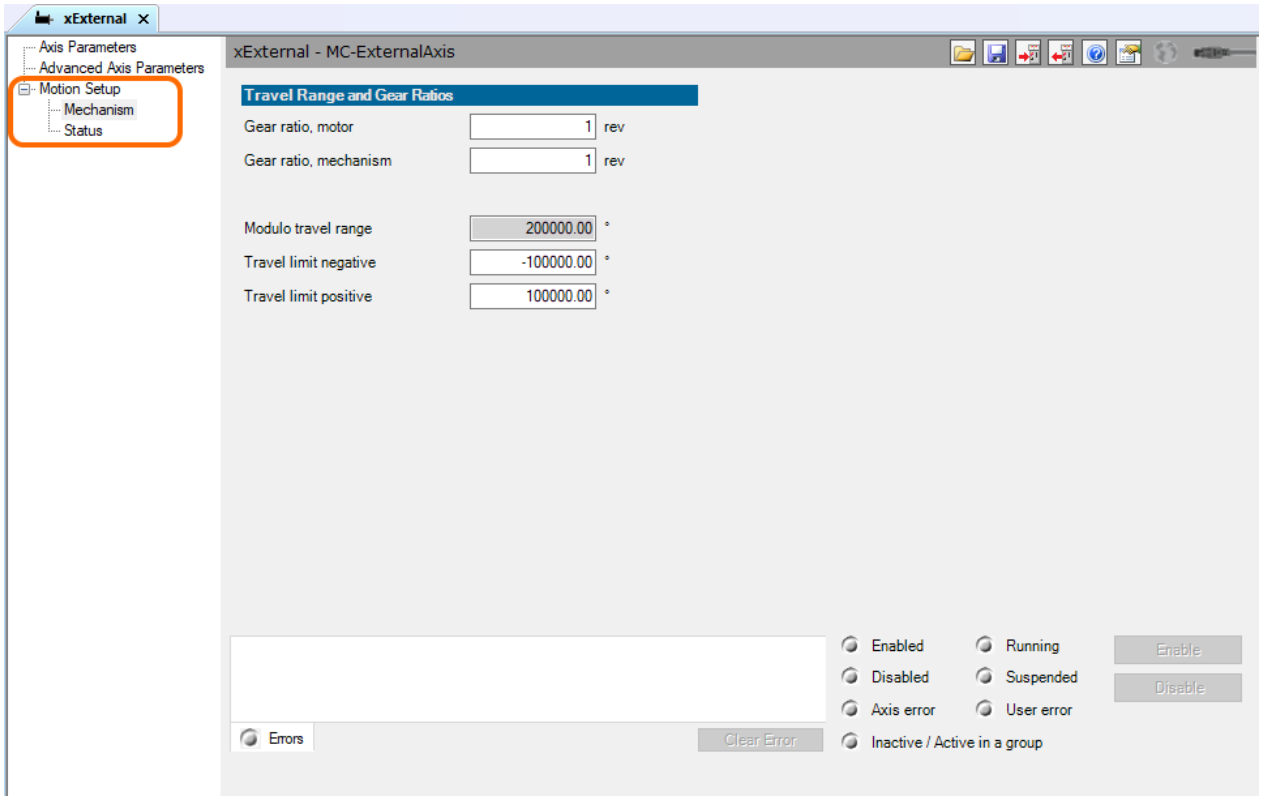




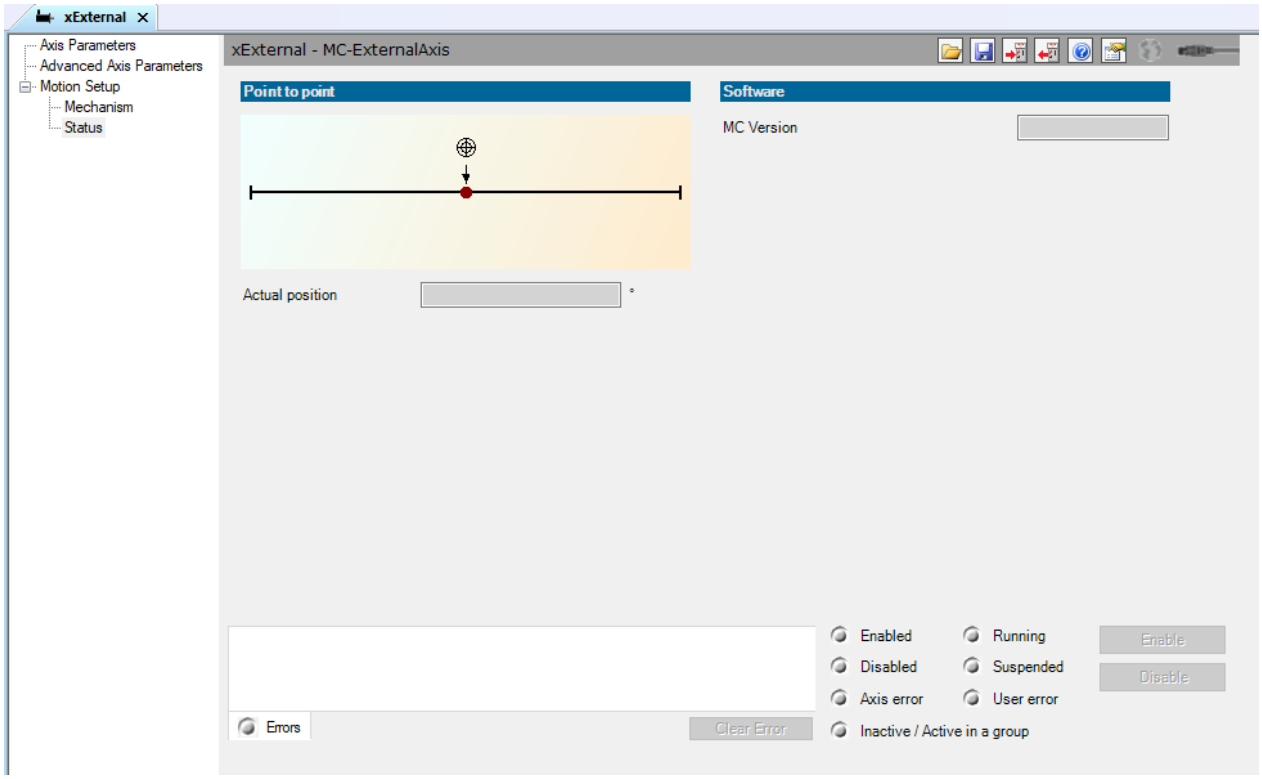
现在可以设置与 JC-440Ext 和 JM-3000 设置相同的参数。

5.5.3 参数

如果基本设置有效，菜单中将出现 Motion Setup 选项。



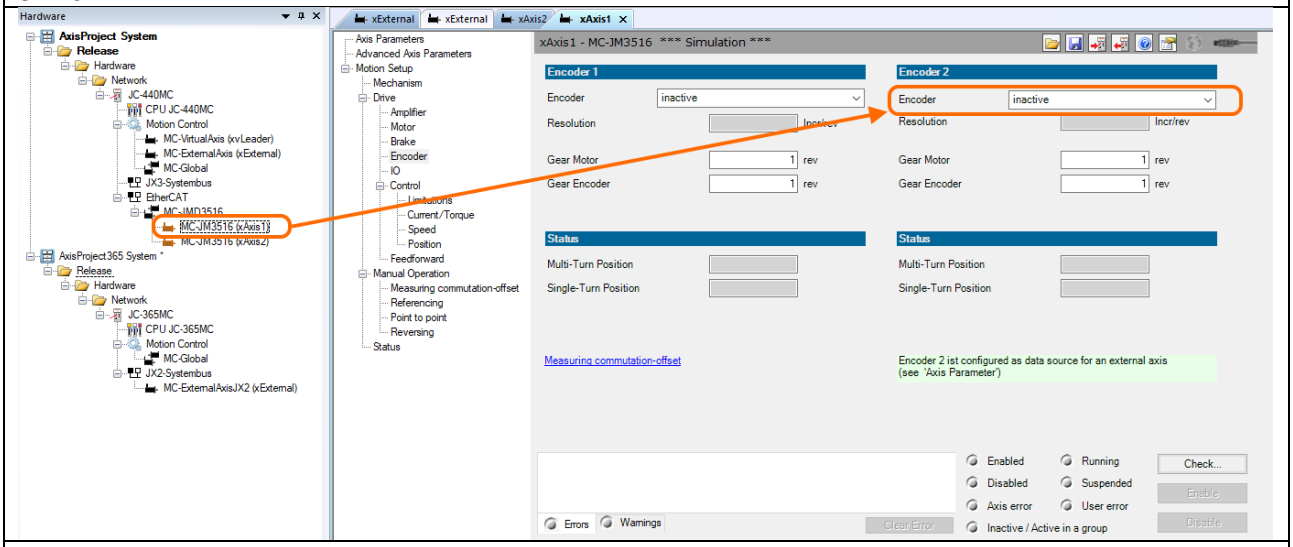
在 Mechanism 页面上，可以设置传动比和行程范围。



状态页面显示外部轴的实际位置，可让您检查外部编码器的功能。

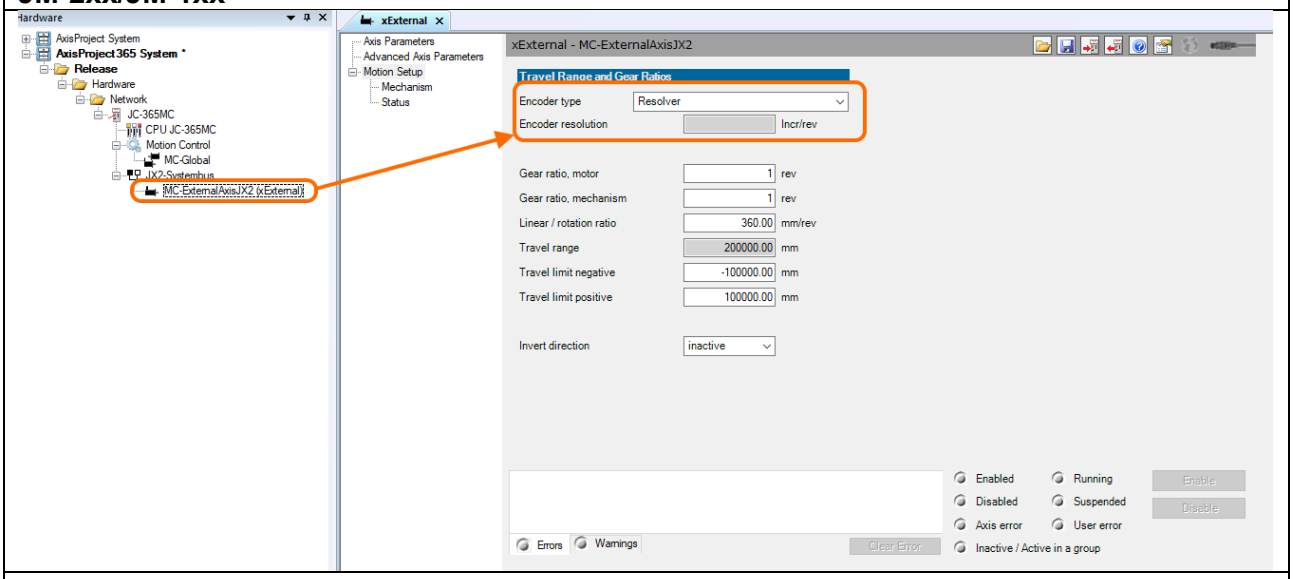
5.5.4 设置编码器类型

JM-3xxx



通过实轴的编码器页面设置编码器类型，该实轴的第二个编码器用于外部轴。

JM-2xx/JM-1xx

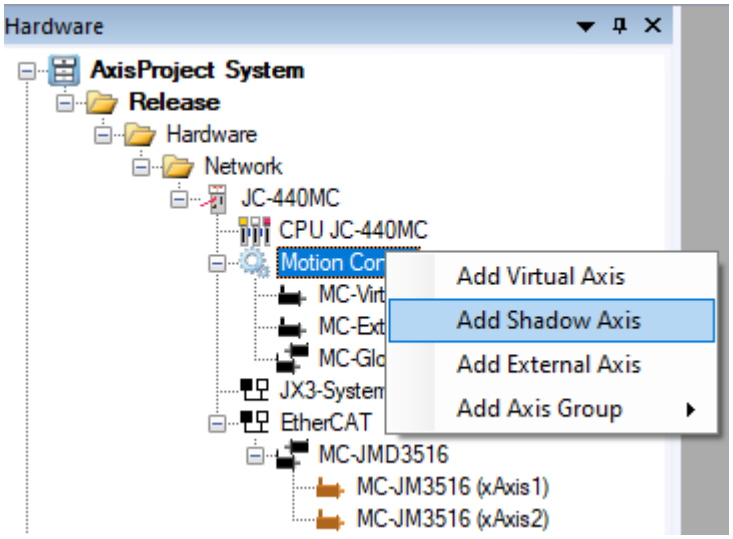


在外部轴的 Mechanism 页面上设置编码器类型。

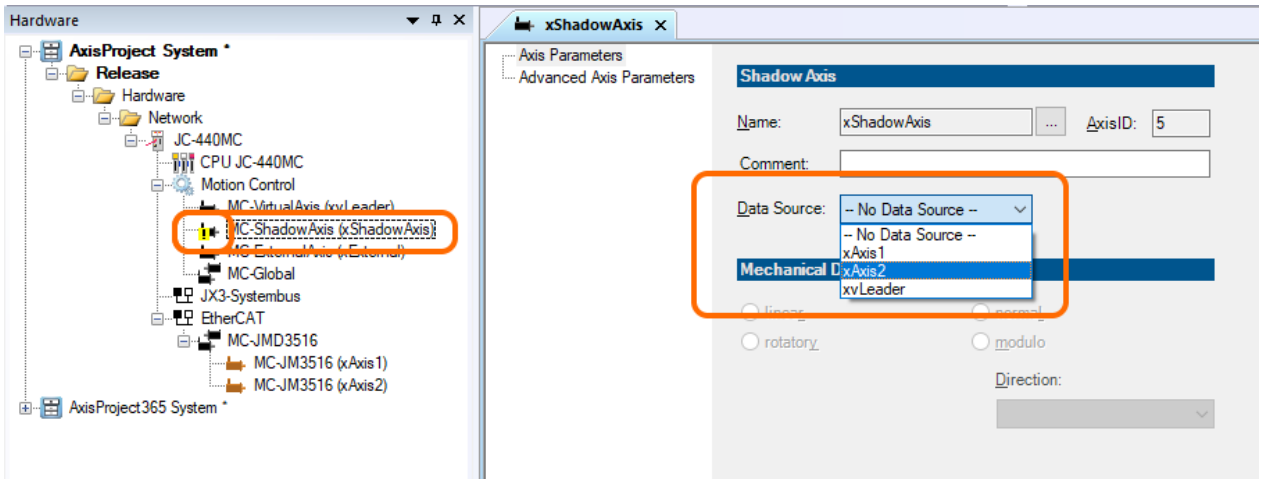
外部轴不需要任何其他设置，例如控制参数、电机类型等。此外，外轴始终处于"Inactive"运行状态。因此 Motion Setup 中不需要额外视图用于配置。

5.6 影子轴

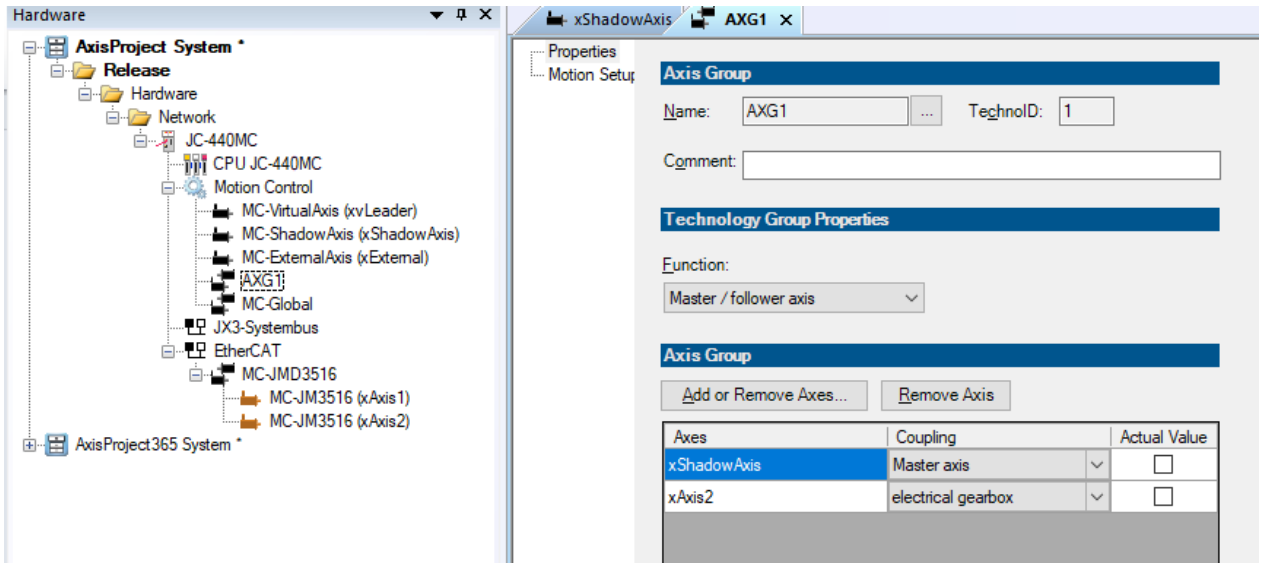
要添加影子轴，请转到"Motion Control"节点并选择"Add Shadow Axis"。



这将创建一个新的影子轴。默认情况下，影子轴有一个带有感叹号的电机图标，表示配置仍然不完整或不正确。

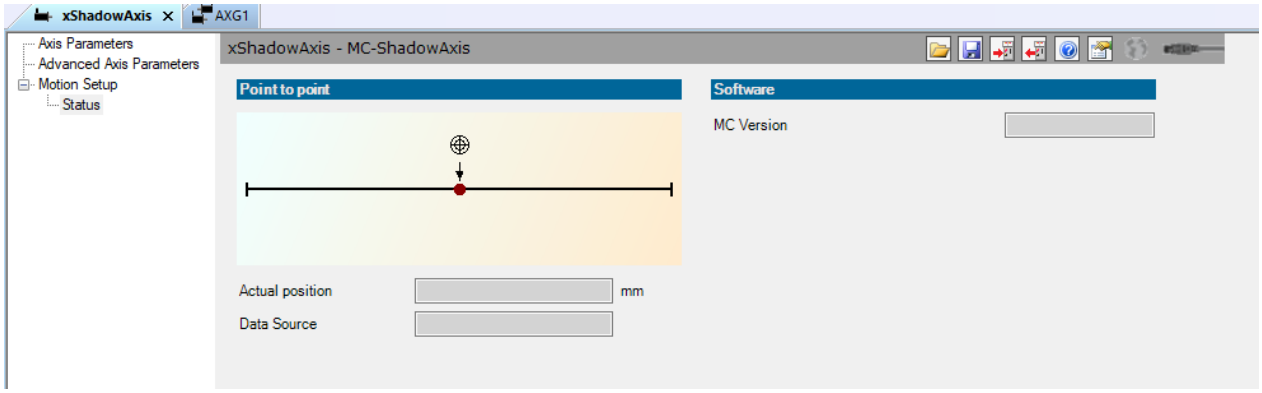


创建后，首先缺少的是在"Axis Parameters"页面上定义的影子轴的数据源。
由于影子轴只能在工艺组中用作主轴，因此只有将此影子轴作为主轴集成到工艺对象中后，配置才完成。



5.6.1 运动设置 - 状态

如果影子轴配置完成，实际位置可以在 Motion Setup 中查看。



影子轴继承了源轴的所有设置，例如行程范围。此外，影子轴始终处于"Inactive"操作状态。因此 Motion Setup 中不需要额外视图用于配置。

6 轴的编程

6.1 设置斜坡参数

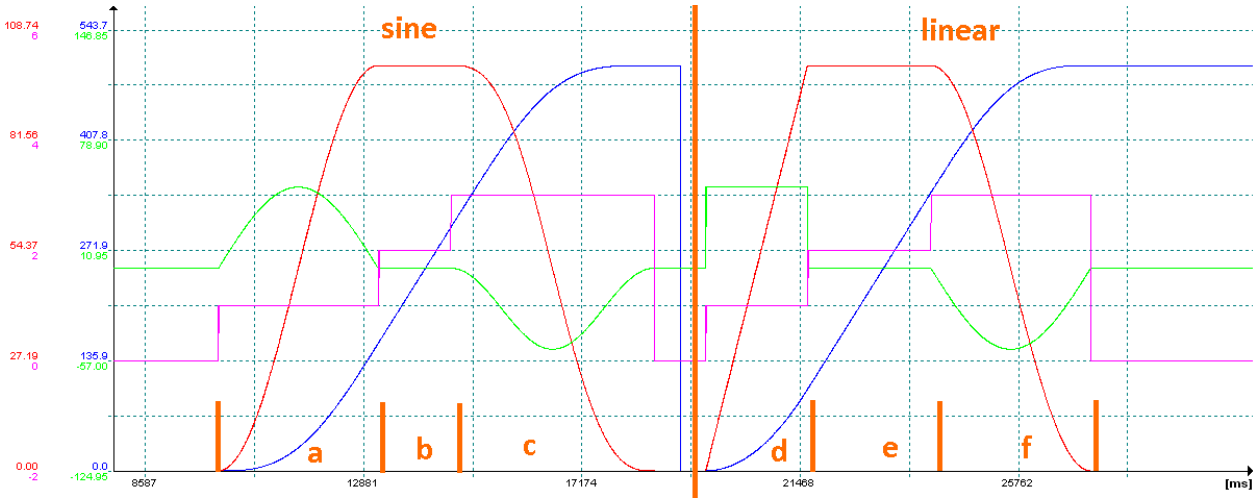
实轴	仿真轴	虚拟轴	外部轴	影子轴

6.1.1 斜坡类型

您可以在此处设置在加速或减速期间用于定位的斜坡类型。

```
xAxis1.Mechanism.Slope.SlopeType := AxisSlopeTypes.Sine;
xAxis1.Mechanism.Slope.SlopeType := AxisSlopeTypes.Linear;
```

对于斜坡类型 AxisSlopeTypes.Sine，使用正弦斜坡加速，使用正弦平方斜坡减速。
对于斜坡类型 AxisSlopeTypes.Linear，使用线性斜坡用于加速，正弦斜坡用于减速。



图例

- 设定位置
- 设定速度
- 设定加速度
- 斜坡（斜率）状态

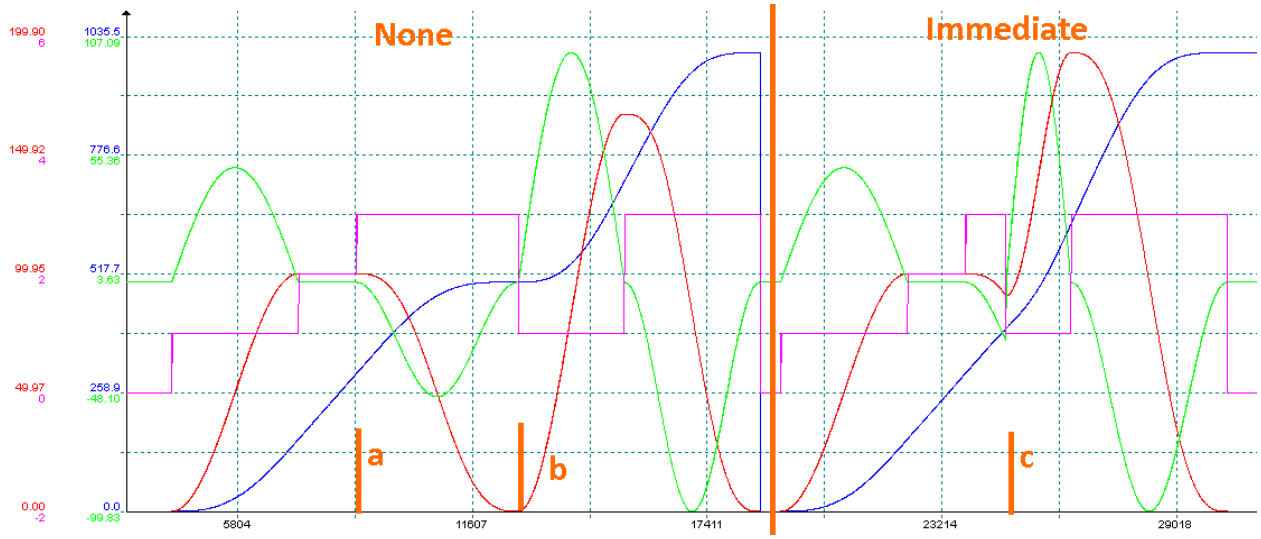
1. 定位*正弦斜坡*	2. 定位*线性斜坡*
a) 正弦加速度 -> 正弦速度增加	d) 线性加速度 -> 速度线性增加，恒定加速
b) 恒速阶段	e) 恒速阶段
c) 正弦²减速 -> 无冲击停止	f) 正弦减速

6.1.2 斜坡中断模式

在电机运动处于减速斜坡时，如果发出新的 MovePtp.Start 指令，可以使用 **Config.RampBreakMode** 属性更改设定位置生成的方式。

斜坡配置	行程对于减速是否足够?	效果
无	是	首先到达第一个设定位置。紧接着直接到达第二个设定位置
无	否	首先到达第一个设定位置。紧接着直接到达第二个设定位置
立即	是	减速斜坡被新设定的最大减速度中止，然后紧接着直接到达第二个设定位置
立即	否	首先到达第一个设定位置。紧接着直接到达第二个设定位置

```
xAxis1.Mechanism.Slope.Config.RampBreakMode := MCAxisRampBreakModes.None;
xAxis1.Mechanism.Slope.Config.RampBreakMode := MCAxisRampBreakModes.Immediate;
```



图例

- 设定位置
- 设定加速度
- 设定速度
- 斜坡（斜率）状态

定位 *无*:

a) 在代码中开始新定位的时间点。

b) 实际运行中开始新定位的时间点。

定位 *立即*:

c) 在代码和现实中开始新定位的时间点。系统取消当前的减速并切换到新定位的加速。

6.2 使能/关闭轴

实轴	仿真轴	虚拟轴	外部轴	影子轴

要使能和关闭轴，请使用 MotionAPI 的 Power 对象。

6.2.1 Power.Enable(), Power.Disable();

然后，可以检查状态以确保操作成功。

```
xAxis1.Power.Enable();
when xAxis1.State.IsEnabled continue;

xAxis1.Power.Disable(MCPowerDisableModes.Forced);
when xAxis1.State.IsDisabled continue;
```

6.2.2 Drive.IsReadyForOperation

对于实轴，建议在使能前检查伺服驱动器是否已准备好运行。然后可以提前更详细地分析并为错误提示做好准备，例如为什么伺服驱动器没有准备好运行。实际操作中，确实可以向未准备好运行的伺服驱动器发送使能指令，这会被确认然后生成轴错误。尤其是在多轴的应用中，所有的轴只有在实际可以使能的情况下才应该使能。

```
if xAxis1.Drive.IsReadyForOperation then
        xAxis1.Power.Enable();
        when xAxis1.State.IsEnabled continue;
else
        // inform user that axis is not operational
end_if;
```

6.2.3 Power.QuickStop()

作为直接关闭使能的替代方法，还可以使用快速停止。此处，速度控制回路中的紧急停止斜坡用于停止然后关闭轴。

```
xAxis1.Power.QuickStop();
when xAxis1.State.IsDisabled continue;
```

6.3 参考点

6.3.1 设置参考点

实轴	仿真轴	虚拟轴	外部轴	影子轴

使用 <Axis>.MoveHome.SetReference() 将轴参考点设置为指定位置。

```
xAxis1.MoveHome.SetReference(0.0);
when xAxis1.MoveHome.IsReferenceSet continue;
```

设置位置后，参考状态变为"IsReferenceSet"。

6.3.2 轴参考点运行

实轴	仿真轴	虚拟轴	外部轴	影子轴

使用 <Axis>.MoveHome.Start 开始参考点运行。例如，这里显示了一个函数调用。

```
xAxis1.MoveHome.Start(
        Directions.Positive,                          // Direction
        ReferencingSwitchConfigs.RefAndLimitSwitch,   // SwitchConfig
        ReferencingModes.SwitchingEdge_2Phase,        // Mode
        100.0,                                        // Acceleration
        50.0,                                         // SpeedSwitchSearch
        10000.0,                                      // MaxDistanceSwitchSearch
        10.0,                                         // SpeedReferenceSearch
        100.0,                                        // MaxDistanceReferenceSearch
        200.0,                                        // ReferenceOffset
        0.0                                           // NormalPosition
        );
when xAxis1.MoveHome.IsReferenceSet continue;
```

该函数调用的所有参数都是可选的。
如果在调试期间还通过 Motion Setup 测试了参考点运行，并设置了必要的参数，这些参数也保存在 .ini 文件中。将文件传输到控制器后，参考点参数已经自动设置，那么下面的调用就足够了。

```
xAxis1.MoveHome.Start();
when xAxis1.MoveHome.IsReferenceSet cotinue;
```

如果只是更改部分参考点参数，其余部分使用调试时的参数或着跟之前的配置相同，那么仅需要设定更改的参数即可。
在这个例子中，到参考点开关的距离和参考点位置将被改变：

```
xAxis1.MoveHome.Start(
        ,                                             // Direction
        ,                                             // SwitchConfig
        ,                                             // Mode
        ,                                             // Acceleration
        ,                                             // SpeedSwitchSearch
        ,                                             // MaxDistanceSwitchSearch
        ,                                             // SpeedReferenceSearch
        ,                                             // MaxDistanceReferenceSearch
        m_nNewReferenceOffset,                        // ReferenceOffset
        m_nNewNormalPosition                          // NormalPosition
        );
when xAxis1.MoveHome.IsReferenceSet continue;
```

参考点运行函数和顺序在 JetSym 的帮助页面 *"MotionHome Illustration"* 里有详细介绍。

6.3.2.1 机械参考点参数

参考点可以是零脉冲的位置，如果没有零脉冲的情况下执行参考点运行，那么参考点也可以是开关的边沿信号。

参考点参数不仅可以在 <Axis>.MoveHome.Start() 命令中配置，还可以单独设置。各个变量在括号中表示。

Direction (<Axis>MoveHome.Direction)

- Directions.Positive
- Directions.Negative

定义搜索参考点开关的方向。

SwitchConfigs (<Axis>MoveHome.SwitchConfig)

- ReferencingSwitchConfigs.NoSwitch
  - 如果仅使用 K0 作为参考点，请使用该项。
- ReferencingchConfigs.RefAndLimitSwitch
  - 带参考点开关和限位开关的参考点运行。如果搜索参考开关期间触发限位开关动作，系统将停下并反向搜索参考点开关。
- ReferencingSwitchConfigs.RefSwitch
  - 在参考点运行期间，仅考虑参考点开关。通常情况应用于是旋转轴或未安装限位开关。
- ReferencingSwitchConfigs.LimitSwitch
  - 限位开关用作参考点开关。使用哪个限位开关取决于参考点运行的方向。如果定为正向启动，正向限位开关为参考点开关。

Mode (<Axis>MoveHome.Mode)

- ReferencingModes.ZeroPulse_2Phase
  - 包含 2 个阶段：
    - 1) 粗略搜索阶段：通常速度配置较快(SpeedSwitchSearch)，以便快速找到零脉冲 K0
    - 2）精细搜索阶段：通常速度配置较慢（SpeedReferenceSearch），以便更精确地找到零脉冲 K0 的边沿
- ReferencingModes.SwitchingEdge_2Phase
    - 1）粗略搜索阶段：通常速度配置较快（SpeedSwitchSearch），以便快速找到参考开关

        - 2) 精细搜索阶段：通常速度配置较慢（SpeedReferenceSearch），以便更精确地找到参考开关的边沿
- ReferencingModes.ZeroPulse_1Phase
    - 一个搜索阶段：通常速度配置缓慢（SpeedReferenceSearch），以便更精确地找到零脉冲 K0 的边沿
- ReferencingModes.SwitchingEdge_1Phase
    - 一个搜索阶段：通常速度配置较慢（SpeedReferenceSearch），以便更精确地找到参考开关的边沿

Acceleration (<Axis>MoveHome.Acceleration)
参考点运行期间的加速和减速。

SpeedSwitchSearch (<Axis>MoveHome.SpeedSwitchSearch)
“粗略”搜索阶段的速度。通常选择高于 SpeedReferenceSearch 的速度，以便快速找到参考点。

MaxDistanceSwitchSearch (<Axis>MoveHome.MaxDistanceSwitchSearch)
在“粗略”搜索阶段找到参考点的最大距离。如果在参考点运行期间超过此距离，会产生错误。

SpeedReferenceSearch (<Axis>MoveHome.SpeedReferenceSearch)
“精细”搜索阶段的速度。这通常选择低于 SpeedSwitchSearch 来确定参考点的边沿。

MaxDistanceReferenceSearch (<Axis>MoveHome.MaxDistanceReferenceSearch)
在“精细”搜索阶段找到参考点的最大距离。如果在参考点运行期间超过此距离，会产生错误。

ReferenceOffset (<Axis>MoveHome.ReferenceOffset)
参考点偏移，即与找到的参考点的之间距离。如果参考点的边沿是在参考点运行时确定的，轴定位将偏移该值。例如，ReferenceOffset =0mm，使用边沿位置。

NormalPosition(<Axis>MoveHome.NormalPosition)
到达参考点偏移之后，指定当前轴的位置。例如：ReferenceOffset = 100 mm，NormalPosition = 20 mm：当在参考搜索过程中找到开关边沿时，轴将进一步移动 100 mm。通过设定该位置，可以使得轴在参考运行结束时，指定当前位置为 20 m。

6.3.2.2 监控参考点运行：

参考点运行的不同阶段映射到状态里，以便查询。

<Axis>MoveHome.IsReferenceRunStarted
参考点运行已经并开始搜索参考点的边沿。

<Axis>MoveHome.IsSwitchFound
在“粗略”搜索阶段找到了参考点开关的边沿。轴现在开始“精细”搜索边沿。

<Axis>MoveHome.IsReferenceFound
在“精细”搜索阶段找到参考开点关的边沿。轴现在移动到由参考点偏移确定的位置。

<Axis>MoveHome.IsReferenceSet
参考点运行完成并设置原点位置（正常位置）。

6.4 定位

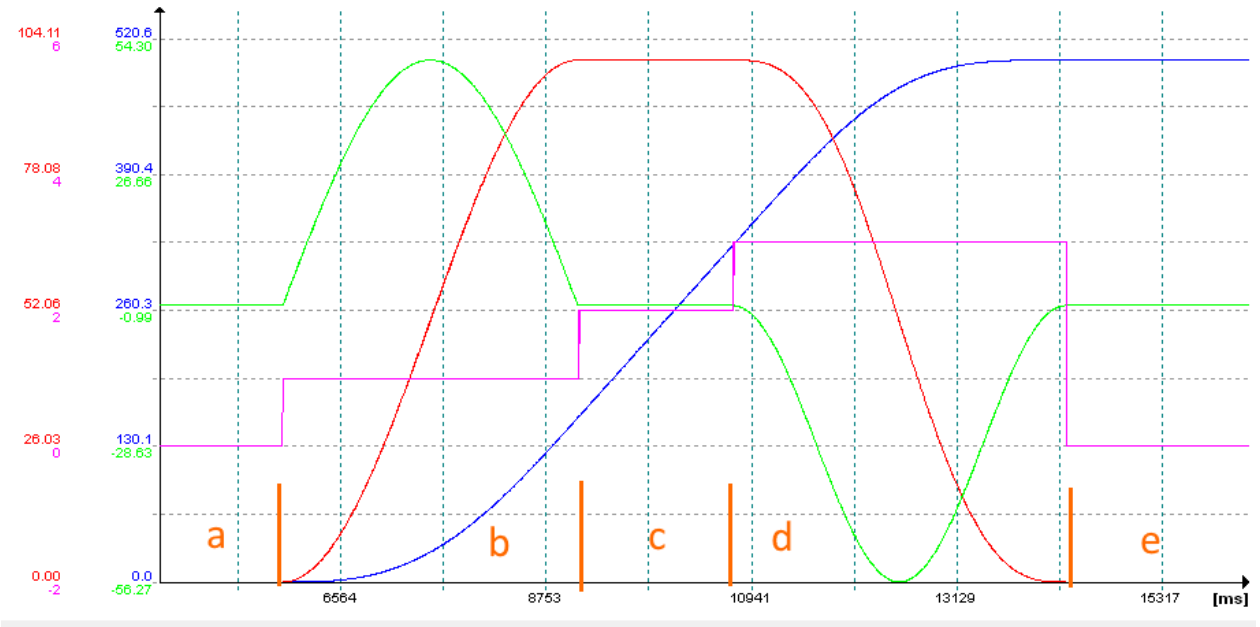
实轴	**仿真轴**	**虚拟轴**	外部轴	影子轴

可以在点到点 (MovePtp) 或连续模式 (MoveVelocity) 中定位轴。

6.4.1 斜坡状态

一个定位过程分为 4 个阶段：

- 停止
- 加速阶段
- 恒速阶段
- 减速阶段



图例

- 设定位置
- 设置设定速度
- 设定速度
- 斜坡（斜率）状态

a) 停止

- 轴停止

- 设定位置不会改变
- 设定速度和设定加速度为零
- 斜坡状态停止（0：IsStopped）

b）加速阶段

- 轴正在加速
- 设定位置正在改变
- 设定速度在增加
- 设定加速度上升和下降，直到达到目标速度
- 斜坡状态为"accelerating"（1：IsAccelerating）

c）恒速阶段

- 轴以恒定速度运行
- 设定位置现在线性增加
- 设定速度保持恒定
- 设定加速度为 0
- 斜坡状态为"moving at constant speed"（2：IsAtConstantSpeed）

d）减速阶段

- 轴正在减速
- 设定位置正在改变
- 设定速度降低
- 设定加速度上升和下降，直到达到目标
- 斜坡状态为"decelerating"（3：IsDecelerating）

e）参考 a）

根据定位参数的设置，如果有从加速到减速的直接过渡，恒速行程也可以省略。
在 STX 中查询：

```
if xAxis1.Mechanism.Slope.IsAccelerating then
elseif xAxis1.Mechanism.Slope.IsAtConstantSpeed then
elseif xAxis1.Mechanism.Slope.IsDecelerating then
elseif xAxis1.Mechanism.Slope.IsStopped then
elseif xAxis1.Mechanism.Slope.SlopeState == AxisSlopeStates.Stopped then
end_if;
```

6.4.2 点到点 (MovePtp)

对于点到点定位，使用 <Axis>.MovePtp.Start 或 <Axis>.MovePtp.StartNewTargetPosition 指令。
轴根据斜坡配置和输入的参数移动到指定的目标。

如果是<Axis>.MovePtp.Start 指令，可以指定模式、目标位置、速度、加速度、减速度和目标窗口参数。或者，如果运动速度、加速度和减速度不改变，也可使用 <Axis>.MovePtp.StartNewTargetPosition。

示例 <Axis>.MovePtp.Start：

```
xAxis1.MovePtp.Start(
        AxisPositioningModes.AbsNormal,      // PostioningMode
        500.0,                               // TargetPosition
        100.0,                               // Speed
        50.0,                                // Acceleration
        50.0,                                // Deceleration
        1.0                                  // TargetWindow
);
when xAxis1.State.IsEnabled and xAxis1.IsInTargetWindow continue;
```

示例 <Axis>.MovePtp.StartNewTargetPosition：

```
xAxis1.MovePtp.StartNewTargetPosition (
        AxisPositioningModes.AbsNormal,      // PostioningMode
        500.0,                               // TargetPosition
        1.0                                  // TargetWindow
);
when xAxis1.State.IsEnabled and xAxis1.IsInTargetWindow continue;
```

由于这两个函数的参数都是可选的，所以下面的例子也可以用来开始一个新的定位，其中速度、加速度和减速度保持不变。

```
xAxis1.MovePtp.Start(
        AxisPositioningModes.AbsNormal,      // PostioningMode
        500.0,                               // TargetPosition
        ,                                    // Speed
        ,                                    // Acceleration
        ,                                    // Deceleration
        1.0                                  // TargetWindow
);
when xAxis1.State.IsEnabled and xAxis1.IsInTargetWindow continue;
```

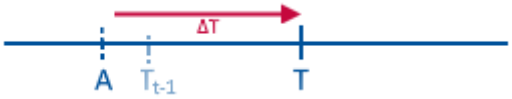
什么时候使用 MovePtp.Start，什么时候使用 MovePtp.StartNewTargetPosition？原则上，特别是上述例子中的功能可以被认为是冗余的和等效的。因此，使用哪个功能取决于程序开发人员的偏好。本应用笔记的作者仅使用 MovePtp.Start。

6.4.3 定位方式

- AxisPositioningModes.AbsNormal
  - 设定位置是绝对的
  - 仅适用于常规轴（非模态）

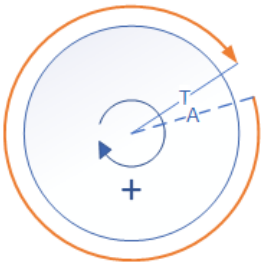
- AxisPositioningModes.RelTarget
  - 新设定位置相对于上一个设定位置
  - 例如，一个 Ptp 定位在到达目标之前就停止了，那么可以使用“Start(AxisPositioningModes.RelTarget, 0.0....)”将轴移动到最后一个目标位置

- AxisPositioningModes.RelActual
  - 新的目标位置相对于当前位置
  - 对应于要运行的路径



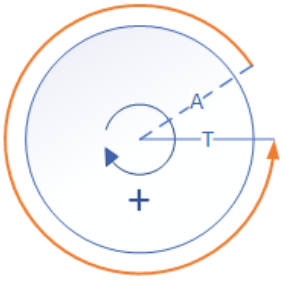
- AxisPositioningModes. AbsModuloPos
  - 应用于模态轴
  - 运动方向为正向

  例如：
  - 轴在 50.0°，目标是 40.0°，然后轴将正向移动 350°。



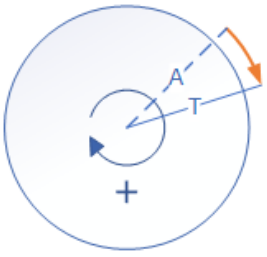
- AxisPositioningModes.AbsModuloNeg
  - 应用于模态轴
  - 运动方向为负向

  例如：
  - 轴在 40.0°，目标是 50.0°，然后轴将负向移动 350°。



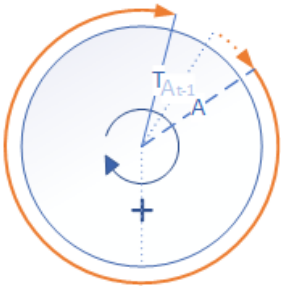
- AxisPositioningModes.AbsModuloAuto
  - 应用于模态轴
  - 运动方向是最短的路径

  例如：
  - 轴在 40.0°，目标是 50.0°，然后轴将正向移动 10°。



- AxisPositioningModes. AbsModuloKeepDirection
  - 应用于模态轴
  - 运动方向对应上次定位的方向

  例子：
  - 轴位于 40.0° 并正在移动到 50°，现在需要



使用 AbsModuloKeepDirection 移动到

20°：它将正向移动 330°。

6.4.3.1 更改速度 (<Axis>.MovePtp.NewSpeed)

要改变正在运动的轴的速度，可使用<Axis>.MovePtp.NewSpeed() 指令指定新速度。轴以指定的加速度加速或减速到新速度。

```
xAxis1.MovePtp.NewSpeed(
        100.0                               // Speed
);
```

使用 *Speed* 参数指定新速度。

6.4.3.2 更改加速度 (<Axis>.MovePtp.NewAcceleration)

要改变正在加速运动的轴的加速度，可使用<Axis>.MovePtp.NewAcceleration () 指令指定新值。

```
xAxis1.MovePtp.NewAcceleration(
        100.0                               // Acceleration
);
```

6.4.3.3 更改减速度 (<Axis>.MovePtp.NewDeceleration)

要改变正在减速运动的轴的减速度，可使用<Axis>.MovePtp.NewDeceleration () 指令指定新值。

```
xAxis1.MovePtp.NewDeceleration(
        100.0                               // Deceleration
);
```

6.4.4 连续定位（MoveVelocity）

可以使用 <Axis>.MoveVelocity.Start() 指令触发连续定位。在这种情况下，轴以设置的加速度加速到目标速度。

```
xAxis1.MoveVelocity.Start(
        Directions.Positive,  // Direction
        100.0,                // Speed
        100.0,                // Acceleration
        150.0                 // Deceleration
);
```

Direction
连续定位的方向。

Speed

连续定位的目标速度。该值必须始终 > 0。

Acceleration
该值用于加速到目标速度。该值必须始终 > 0。

Deceleration
停止连续定位时，使用该减速度，除非被停止指令覆盖。该值必须始终 > 0。

6.4.4.1 反转方向 (<Axis>.MoveVelocity.Reverse)

要反转运动方向，可以使用 <Axis>.MoveVelocity.Reverse() 指令。然后，轴以设置的加速度沿相反方向加速到相同速度。

```
xAxis1.MoveVelocity.Reverse();
```

作为上一个指令的替代，可以再次给出 <Axis>.MoveVelocity.Start() 指令，但指定方向为相反的方向。

```
xAxis1.MoveVelocity.Start(
        Directions.Negative,   // Direction
        100.0,                 // Speed
        100.0,                 // Acceleration
        150.0                  // Deceleration
);
```

6.4.4.2 更改速度 (<Axis>.MoveVelocity.NewSpeed)

要改变正在运动的轴的速度，可使用<Axis>.MoveVelocity.NewSpeed() 指令指定新速度。轴以指定的加速度加速或减速到新速度。

```
xAxis1.MoveVelocity.NewSpeed(
        100.0                              // Speed
);
```

使用 *Speed* 参数指定新速度。由于此值必须始终大于 0，因此可以更改速度，但不能更改移动方向！

6.4.4.3 更改加速度 (<Axis>.MoveVelocity.NewAcceleration)

要改变正在加速运动的轴的加速度，可使用 <Axis>.MoveVelocity.NewAcceleration () 指令指定新值。

```
xAxis1.MoveVelocity.NewAcceleration(
        100.0                              // Acceleration
);
```

6.4.4.4 更改减速度 (<Axis>.MoveVelocity.NewAcceleration)

要改变正在减速运动的轴的减速度，可使用 <Axis>.MovePtp.NewDeceleration () 指令指定新值。

```
xAxis1.MoveVelocity.NewDeceleration(
        100.0                                   // Deceleration
);
```

6.4.5 停止（MoveHalt）

可以使用 <Axis>.MoveHalt.Start 指令停止正在进行的运动。参数是可选的。如果没有指定任何参数，将会使用此时有效的参数。
轴以设置的减速度停止。如果没有指定这个参数，将会使用前一个移动指令所使用的减速度。

```
xAxis1.MoveHalt.Start(
        MCAxisHaltModes.Normal,         // Mode
        85.0                            // Deceleration
);
when xAxis1.State.IsEnabled continue;
```

使用<Axis>.State.IsEnabled 查询停止过程是否完成。

```
when xAxis1.Mechanism.Slope.IsStopped continue;
```

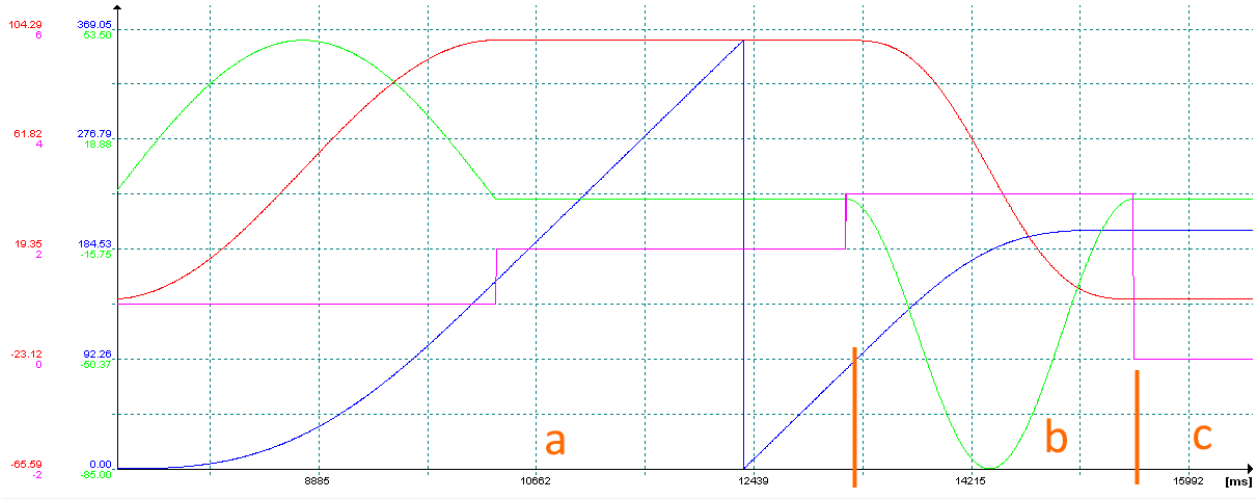
作为替代，您可以使用 <Axis>.Mechanism.Slope.IsStopped。
区别在于：

- 在 <Axis>.State.IsEnabled 的情况下，轴停止并启用
- 在<Axis>.Mechanism.Slope.IsStopped 的情况下，轴也静止不动，但运行状态不清楚。例如，状态可以是 IsEnabled，但也可以是 IsDisabled 或其他状态。

6.4.5.1 模式

MCAxisHaltModes.Normal

从指令起始这一点开始，正在进行的运动通过主动减速度停止。



图例

- 设定位置
- 设定速度
- 设定加速度
- 斜坡（斜率）状态

a) 开始连续定位。

b) 使用 MCAxisHaltModes.Normal 停止轴。

轴减速至静止。

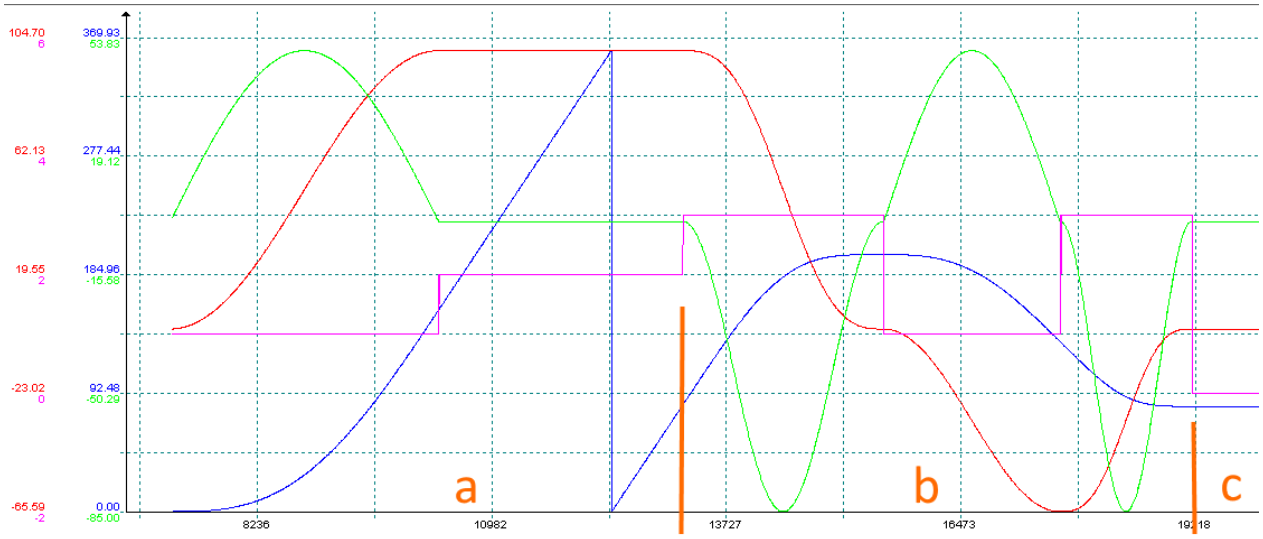
斜坡状态变为"IsDecelerating"。

c) 轴现在停止。

斜坡状态现在是"IsStopped"。

MCAxisHaltModes.AtActualPosition

停止正在进行的运动，轴返回到发出命令的点。



图例

- 设定位置
- 设定加速度
- 设定速度
- 斜坡（斜率）状态

a) 开始连续定位。

b) 使用 MCAxisHaltModes.AtActualPosition 停止轴。

首先，轴减速至静止。然后，轴移动到 b) 开头的位置。

斜坡状态变为"IsDecelerating"，然后在重新定位期间变为"IsAccelerating"和"IsDecelerating"。

c) 轴现在停在 b) 开头的位置。

斜坡状态现在是"IsStopped"。

Jetter automation

Jetter AG
Graeterstrasse 2
71642 Ludwigsburg
Germany
www.jetter.de

E-mail info@jetter.de
Phone +49 7141 2550-0

坚德自动化技术（上海）有限公司.
上海市浦东新区康桥路 787 号 6 号楼 105 室
邮编：201315
www.jetterat.cn

contact@jetterat.cn
+86 21 5869 1233

We automate your success.