

## 运动控制-工艺组

应用笔记 049

608 847 48\_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](http://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   | Motion Control - Technology Group |
| Document type  | Application Note 049              |
| Translation of the original German language document |                                   |
| Document revision                                    | 1.00                              |
| Item number  | 608 847 48_00                     |
| Date of issue  | 2021-02-08                        |

## 目录

|       |                             |    |
|-------|-----------------------------|----|
| 1     | 介绍 .....                    | 1  |
| 1.1   | 前提条件 .....                  | 1  |
| 1.2   | 什么是工艺组? .....               | 1  |
| 1.3   | 耦合类型 .....                  | 1  |
| 1.3.1 | 电子齿轮 .....                  | 1  |
| 1.3.2 | 凸轮 .....                    | 1  |
| 1.3.3 | 实际值/设定点耦合 .....             | 2  |
| 1.4   | 主轴类型 .....                  | 2  |
| 1.5   | 从轴类型 .....                  | 3  |
| 2     | 创建一个工艺组 .....               | 4  |
| 2.1   | 如何创建一个工艺组 .....             | 4  |
| 2.2   | 配置工艺组 .....                 | 11 |
| 2.3   | 将 MC 文件下载到控制器 .....         | 14 |
| 3     | 在 Motion Setup 中应用工艺组 ..... | 16 |
| 3.1   | 操作工艺组 .....                 | 16 |
| 3.1.1 | 错误页面-概述 .....               | 18 |
| 3.1.2 | 状态页-概述 .....                | 19 |
| 3.1.3 | 其他窗口 .....                  | 19 |
| 4     | 在应用程序中使用工艺组 .....           | 20 |
| 4.1   | 将 Motion API 库集成到项目中 .....  | 20 |
| 4.2   | 重新启动程序 .....                | 21 |
| 4.2.1 | 重新启动控制器 .....               | 21 |
| 4.2.2 | 重新启动应用程序 .....              | 22 |
| 4.3   | 激活工艺组 .....                 | 24 |

---

|                          |           |
|--------------------------|-----------|
| 4.4 停用工艺组 .....          | <b>26</b> |
| 4.5 定位 .....             | 26        |
| 4.5.1 MovePtp .....      | 27        |
| 4.5.2 MoveVelocity ..... | 28        |
| 4.5.3 MoveHalt .....     | 29        |
| 4.5.4 MoveHome .....     | 30        |
| 4.6 诊断程序 .....           | 31        |

# 1 介绍

## 1.1 前提条件

本文的代码和项目示例以及屏幕截图使用以下版本的软件：

- JetSym 5.60

|  |  |
|--|--|
| <ul style="list-style-type: none"> <li>• JC-440MC, OS 1.08.0.00</li> <li>• Motion API 2.0.0.4</li> </ul> | <ul style="list-style-type: none"> <li>• JC-365MC, OS1.33.0.00</li> <li>• Motion API 1.0.0.16</li> </ul> |
|--|--|

在本应用笔记中，以上述不同的控制器来表示不同的 Motion API 1 + 2。

如果它们的用法有所不同，则会在文档中提及。

## 1.2 什么是工艺组？

- 在一个工艺组中，将各个轴组合起来以形成一个网络，以协调各个轴的运动。在一个工艺组中，主轴与若干从轴结合在一起。从轴可以动态耦合和解耦。
- 一工艺组总是由一个主轴和一到多个从轴组成。

## 1.3 耦合类型

### 1.3.1 电子齿轮

术语“电子齿轮”比喻为电子变速，其类似于机械变速器，可改变速度和位置。与机械变速器相比，电子齿轮可以连续改变传动比。电子齿轮可以更轻松地实现动态耦合和解耦过程以及位置偏移的校正。



- 当您定义工艺组时，将创建一个电子齿轮。激活工艺组时，主轴和从轴之间的传动比可以动态的并且仅通过一个比值指定。
- 从轴可以叠加运动。

### 1.3.2 凸轮

当一个或多个从轴的位置取决于主轴的位置并且必须协调循环运行的机器中的复杂运动时序时，就需要使用凸轮。机械凸轮通常用于此目的。

- 电子凸轮具有更大的灵活性，因为可以非常快速地切换到其他凸轮工艺，并且运动部件重量更小。
- 可实现更高的动态和更高的循环速度，同时驱动器更小，磨损减少。

- 电子凸轮可以更灵活地启用和关闭，并可以通过任何偏移量进行移动。使用机械凸轮盘时，只有通过用力旋转驱动轴上的凸轮体才能实现偏置。

### 1.3.3 实际值/设定点耦合

轴对象通常具有位置设定值和实际位置值。

- 位置设定值由 MCX 周期性的计算，并传送到轴的位置控制环进行控制。
- 实际位置值描述了轴当前所在的位置。

通过**实际值耦合**，从轴将耦合到主轴的实际位置值。

如果主轴是实轴或外部编码器，则即使在静止状态下，实际位置也不完全是恒定的。任何波动都会作为参考值传输到从轴，这可能导致不稳定的行为。为了减少这种影响，可以设置适当的用于平滑的滤波器。但是，滤波器也会导致从轴的反应延迟。

此外，将实际值传输到 MC 内核时产生的延迟，也会导致从轴的反应延迟。

这些延迟主要是在主轴速度变化期间引起的。

实际值耦合用于同步从轴，例如在摩擦轮（主轴）的帮助下，通过的产品或运输物体容易打滑。

通过**设定点耦合**，从轴被耦合到主轴的设定点位置。

由于 MCX 会计算位置设定值，因此该值是准确的，并且始终是恒定的，即使轴是静止的。因此，不需要设定用于平滑位置设定值的过滤器。

在同一个 MCX 计算周期中，新的位置设定值会对从轴生效，因此不会出现延迟。速度变化因此立即传递到从轴上。

因此，在大多数情况下，设定值耦合是首选的解决方案。如果使用外部编码器，例如摩擦轮，只能使用实际值耦合。

## 1.4 主轴类型

- 实轴
  - 有伺服驱动器连接，除非伺服驱动器用作模拟轴
  - 设定值由运动控制系统计算
  - 实际值由伺服驱动器返回
  - 可以采用所有 MCX 操作状态，例如“已启用”或“正在运行”。另请参阅 MCX 操作状态概述
- 虚拟轴
  - 没有连接伺服驱动器
  - 设定值由运动控制系统计算
  - 实际值=设定值
  - 可以采用所有 MCX 操作状态，例如“已启用”或“正在运行”。另请参阅 MCX 操作状态概述

- 外部轴
  - 具有计数器模块，也就是纯粹的编码器信号输入
    - 对于带有 EtherCAT 的 JC-440MC, JC-945MC 或 JC-975MC, 驱动器上的附加编码器输入 (例如 JM-3000) 用作计数器模块
    - 对于 JC-647MC, JC-365MC, JC-940MC 或 JC-970MC, 附加的伺服驱动器 (例如 JM-105, JM-108, JM-2xx) 用作计数器模块
  - 设定点未计算
  - 实际值由计数器模块报告
  - MC 操作状态始终为“不活动”
- 影子轴
  - 没有伺服驱动器
  - 实际值和设定值取自源轴
  - MC 操作状态始终为“不活动”

## 1.5 从轴类型

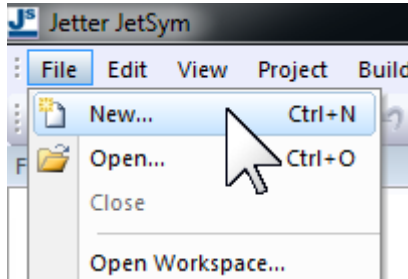
- 实轴
- 虚拟轴

## 2 创建一个工艺组

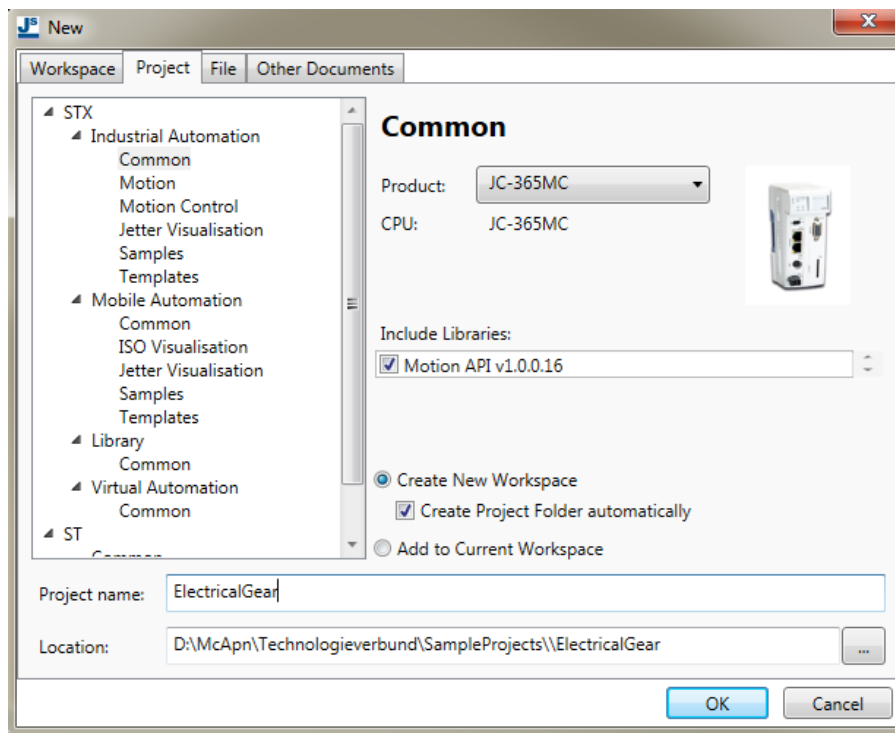
### 2.1 如何创建一个工艺组

下面的新项目示例向您展示了如何创建一个工艺组。

- 打开 JetSym
- 打开“File”菜单，然后单击按钮“New”



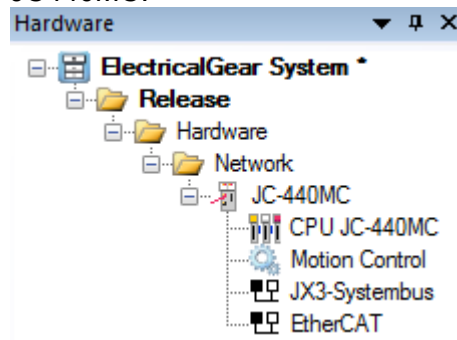
- 用于创建项目的对话框打开。  
选择“JC-440MC”或“JC-365MC”控制器，并包括建议的 Motion API 库。  
输入项目名称并选择用于保存项目的文件夹后，单击“OK”确认输入。



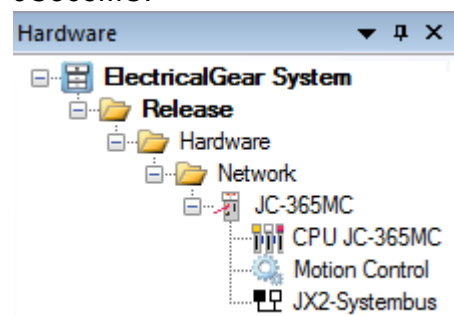


- 软件会自动创建一个基本结构。硬件选项卡包含此树结构。

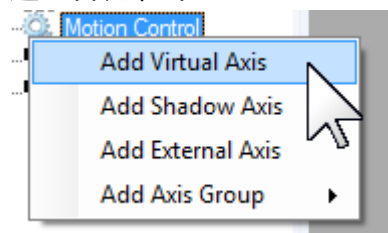
JC440MC:



JC365MC:

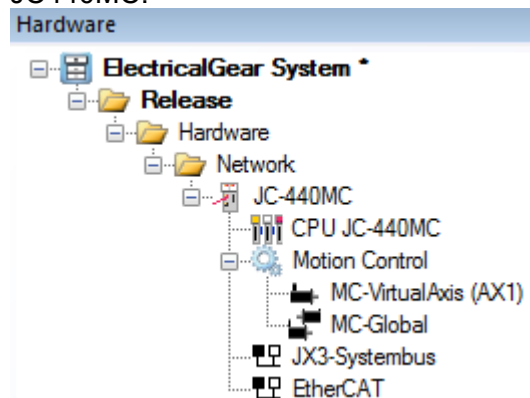


- 首先，创建一个虚拟轴作为主轴。通过“Add Virtual Axis”快捷菜单创建虚拟轴，该快捷菜单通过右键单击“MotionControl”节点打开。

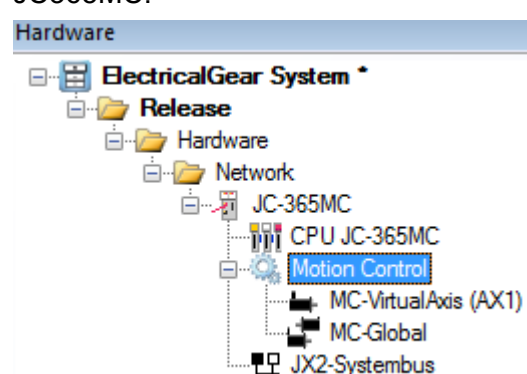


- 然后硬件树中会按照如下显示：

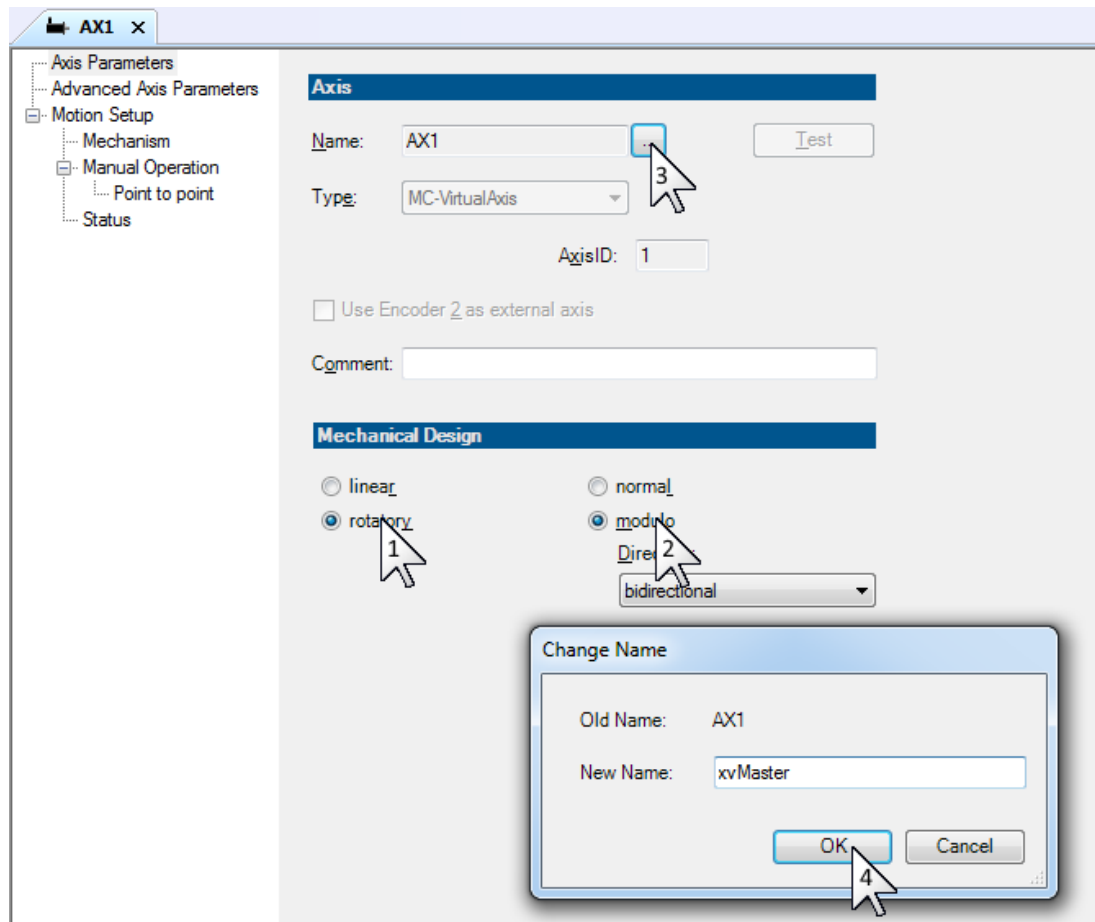
JC440MC:



JC365MC:



- 双击“MC-VirtualAxis (AX1)”节点以打开该虚拟轴的运动设置。



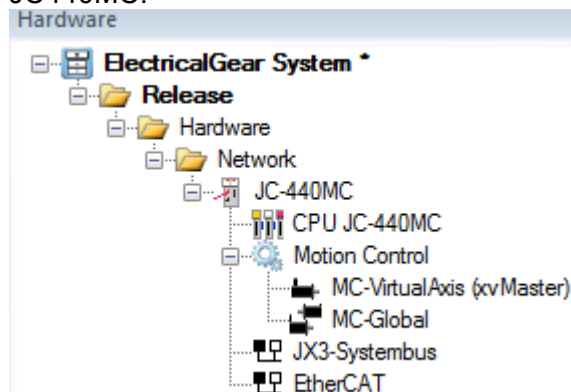
将“Mechanical Design”设置为“rotatory” (1) 和“modulo” (2) 。单击“...” (3) 按钮重命名轴。在此示例中，为轴分配新名称“ xvMaster axis”，然后单击“OK” (4) 确认。

单击“OK”确认后续的对话框。

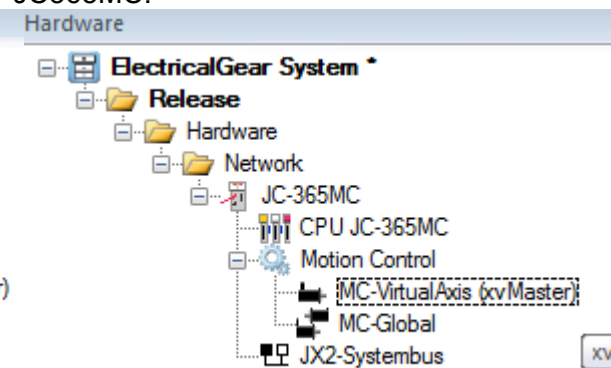
在 STX 项目中，将创建一个 MotionAPI 对象，该对象使您可以访问虚拟轴。

- 轴的新名称现在也出现在硬件树中。

JC440MC:

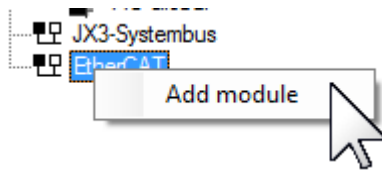


JC365MC:

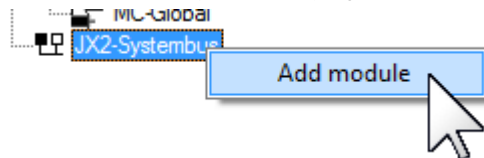


- 现在添加一个实轴。

JC440MC: 可以通过右键单击“EtherCAT”节点打开快捷菜单。



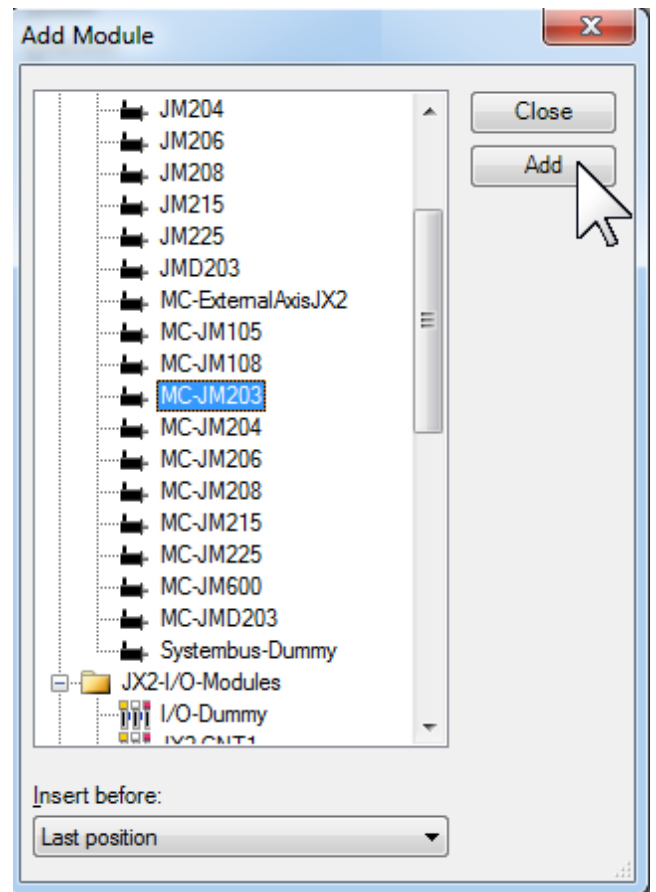
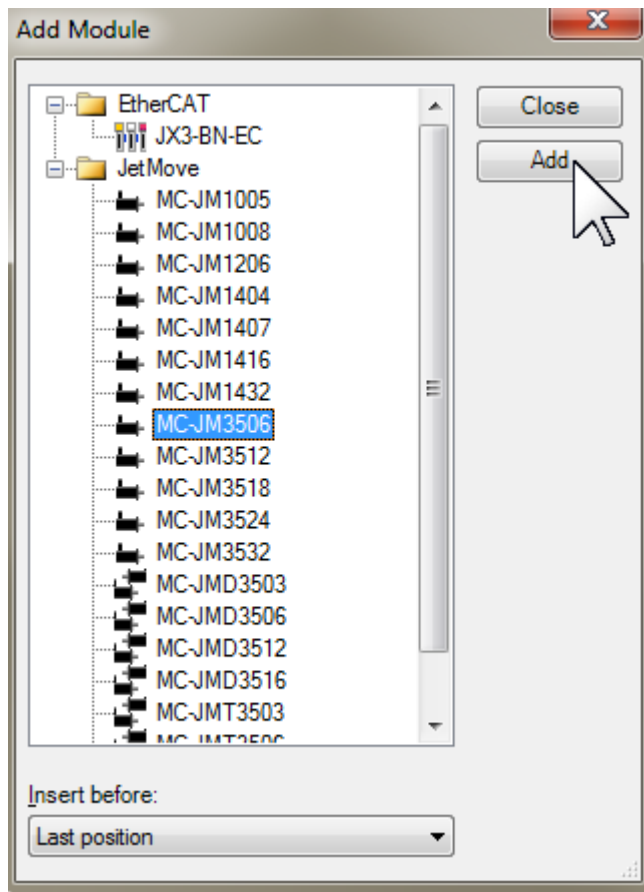
JC365MC: 可以通过右键单击“JX2-Systembus”节点来打开快捷菜单。



- 然后既可以添加所需轴。为此，请选择带有前缀“MC”的运动控制轴，例如“MC-JM203”，然后单击“Add”。对于本示例，一个轴就足够了，然后关闭对话框。

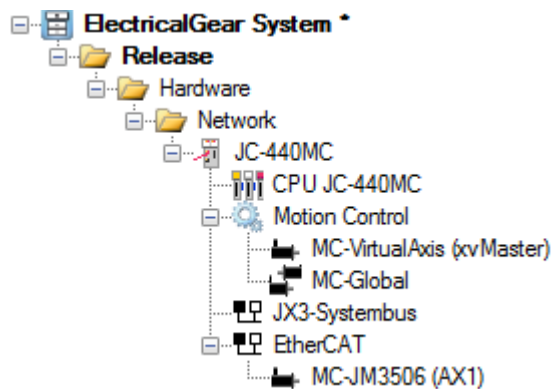
JC440MC:

JC365MC:

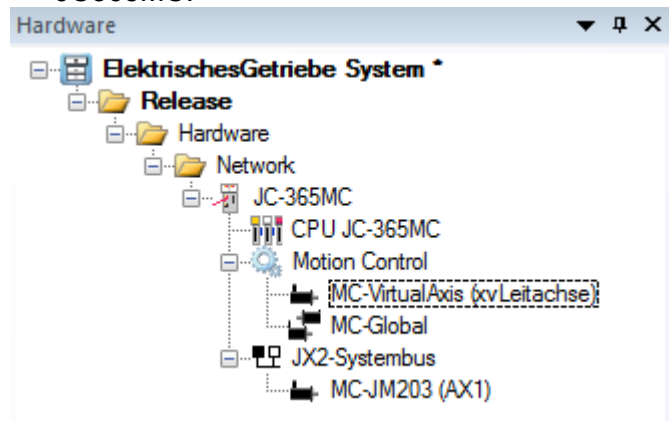


现在，硬件树将创建的轴显示为所选的总线的子节点。

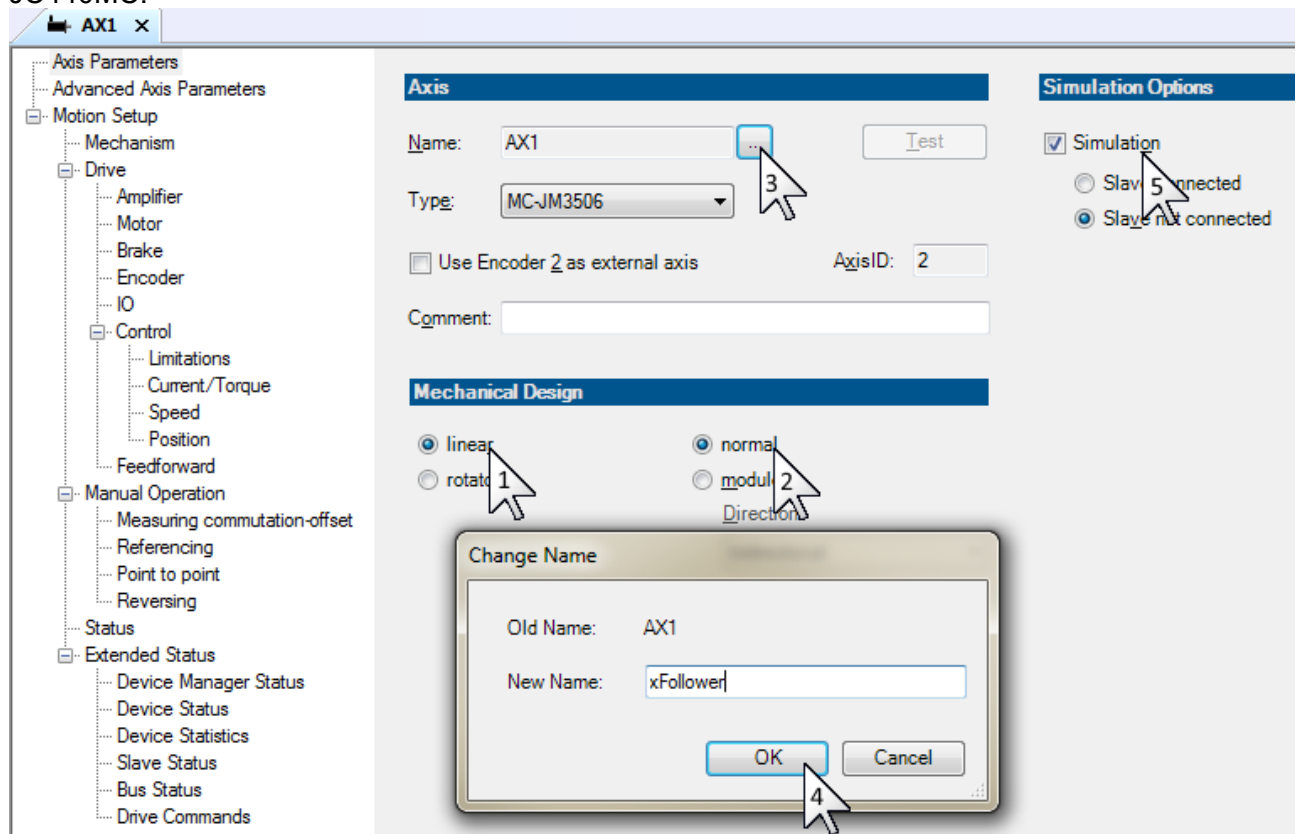
JC440MC:



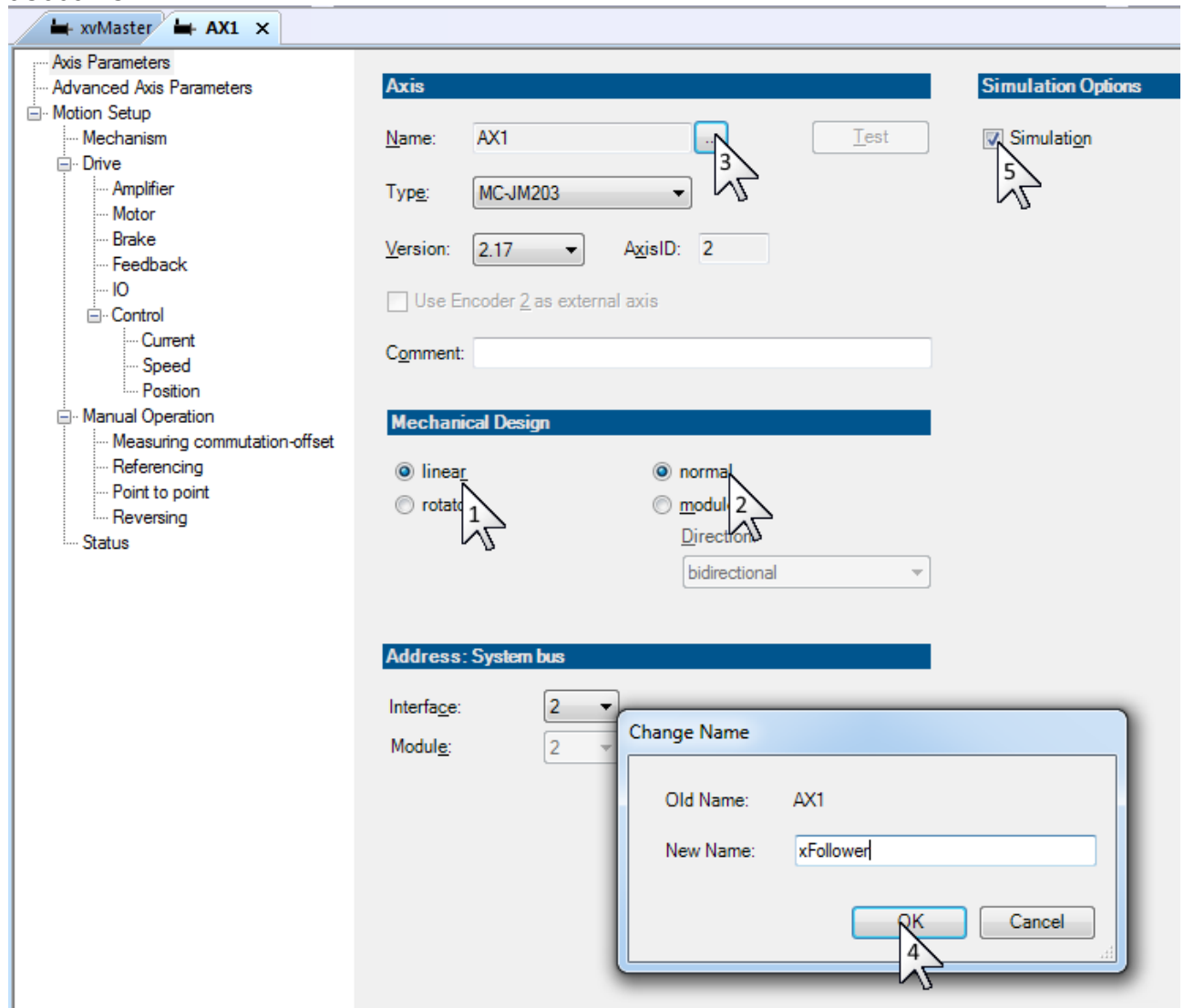
JC365MC:



- 要重命名和配置轴，请双击节点，以打开“AX1”轴的运动设置。
- JC440MC:



## - JC365MC:



根据您的应用选择适当的“Mechanical Design”，即“linear”或“rotary”(1) 和“normal”或“modulo” (2)。

单击“...” (3) 按钮重命名轴。在此示例中，将轴命名为“xFollower”，然后单击“OK” (4) 进行确认。

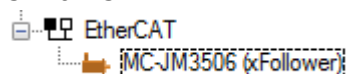
单击“OK”确认后续的对话框。

另外，检查“Simulation” (5) 以选择是否为仿真轴。

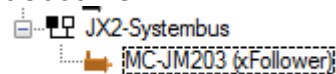
在 STX 项目中，将创建一个 MotionAPI 对象，该对象可用于寻址轴。

## - 现在，更改后的名称也会显示在硬件树中。

## JC440MC:

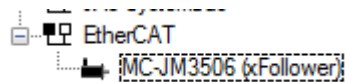


## JC365MC:

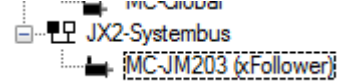


- 硬件树中轴符号的颜色显示该轴是仿真轴（橙色）还是实轴（黑色）。

JC440MC:

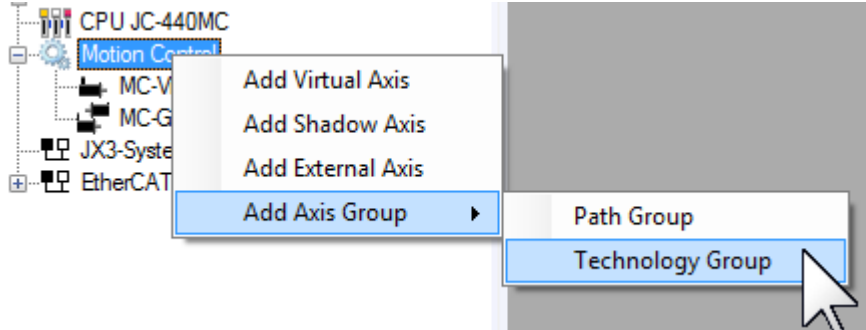


JC365MC:

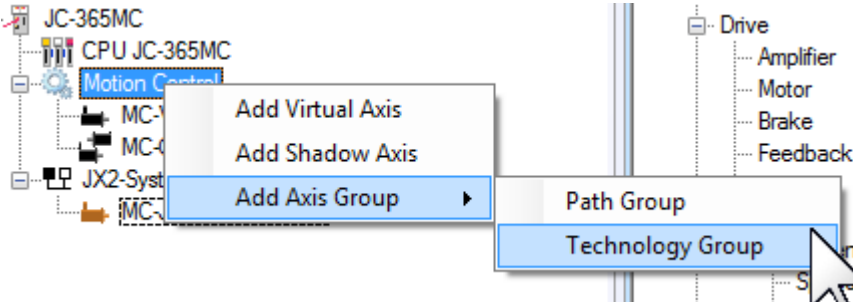


- 通过右键单击“MotionControl”节点打开快捷菜单，然后选择“Add Axis Group”。然后选择“Technology Group”。

JC440MC:

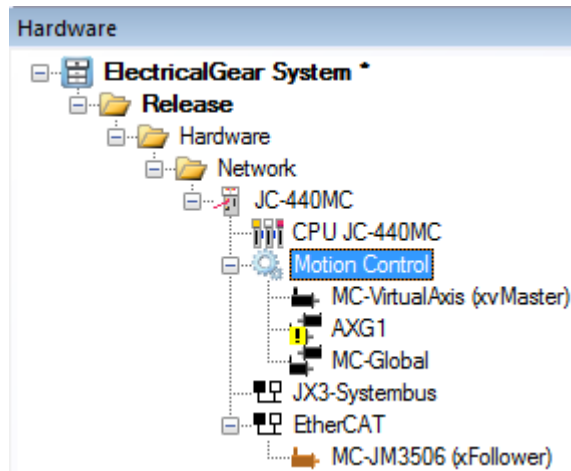


JC365MC:

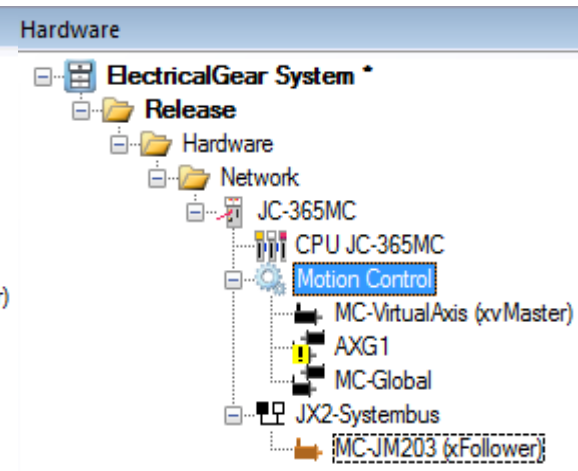


- 现在，新工艺组显示在硬件树中。由于尚未配置该工艺组，因此在硬件树中将其标记为警告标志。一旦有有效的配置可用，该警告标志就会消失。

JC440MC:

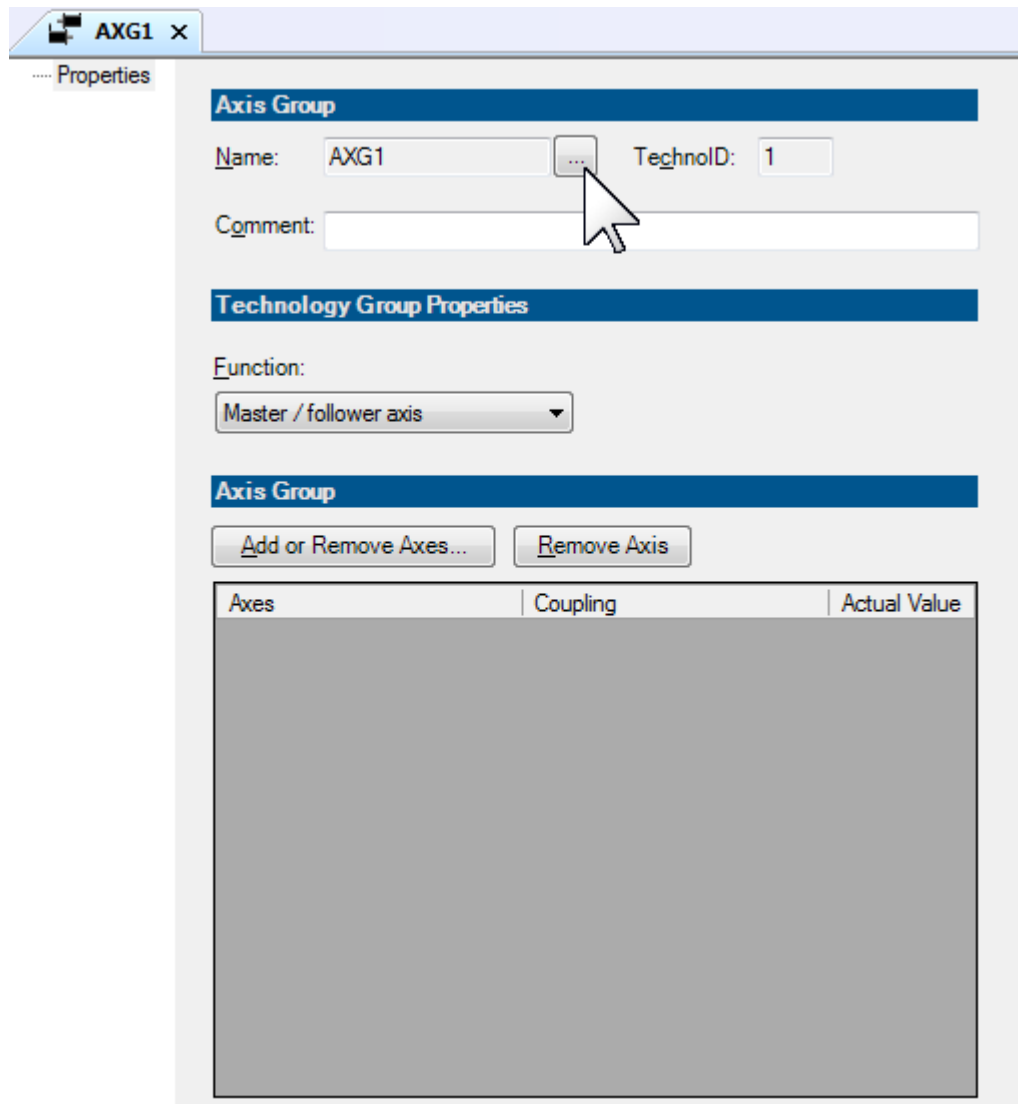


JC365MC:

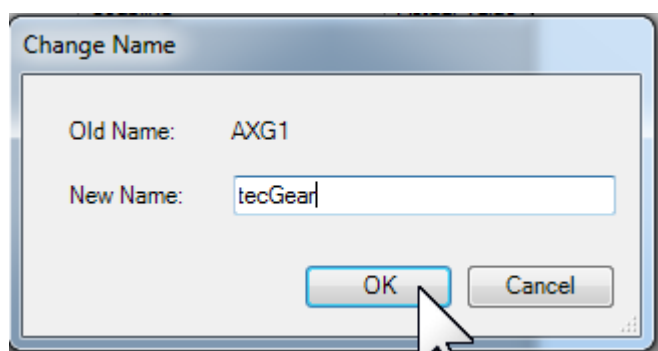


## 2.2 配置工艺组

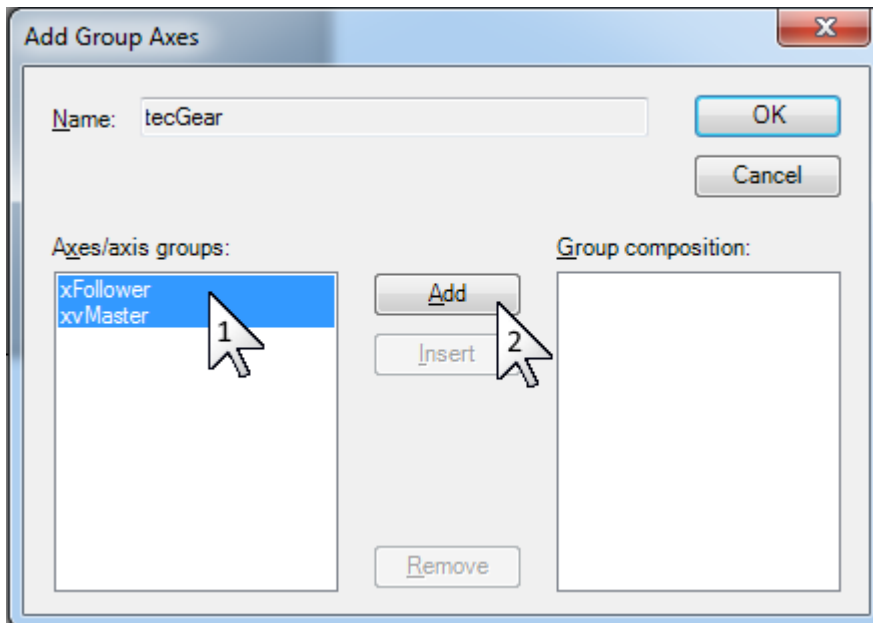
双击新创建的组以打开工艺组的设置。



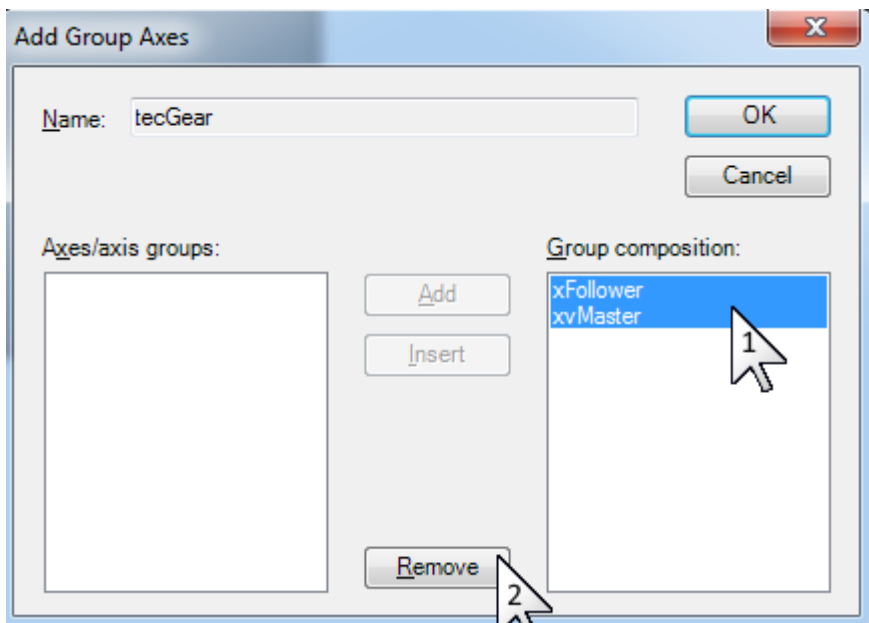
通过单击“...”按钮，您现在可以为工艺组分配一个描述性名称。在 STX 项目中同时会自动创建具有此名称的 MotionAPI 对象。通过 JetSym 确认所有相关对象的重命名，以便新名称也可以在 STX 项目中应用。



在“Axis Group”部分中，单击“Add or Remove Axes”以指定组中涉及的轴。系统将出现以下对话框：



通过单击“Add”或双击相应的轴来选择要包括的轴（1）。将它们移至“Group composition:”，其中列出了组中涉及的所有轴，各个轴的功能分配在上一级的视图中进行。

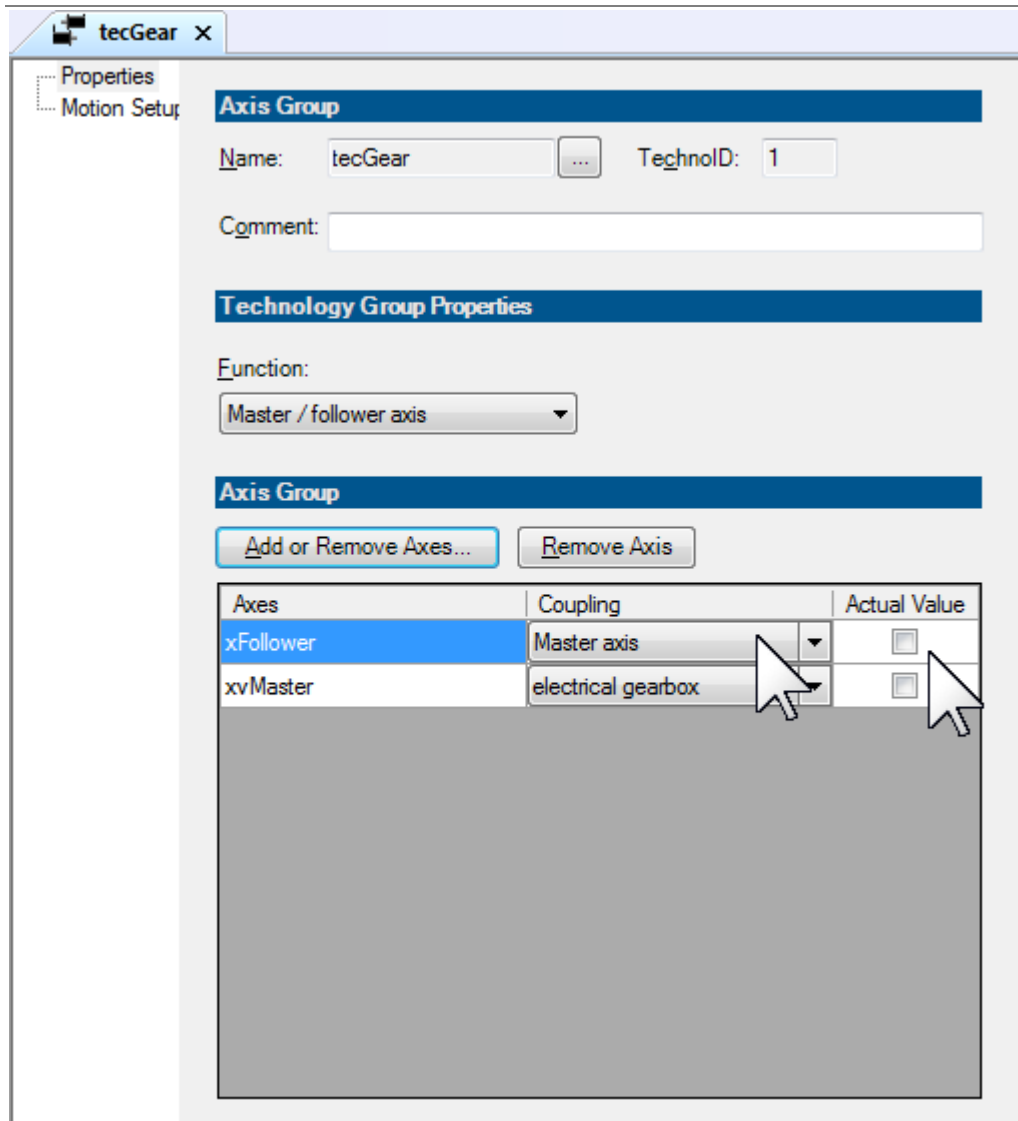


如果要从组中删除任何轴，则可以选择它们（1），然后通过“Remove”（2）从列表中将其删除，或者在“Group composition:”窗口中双击它们。

首次配置工艺组时，建议将列表中的第一个轴作为主轴。

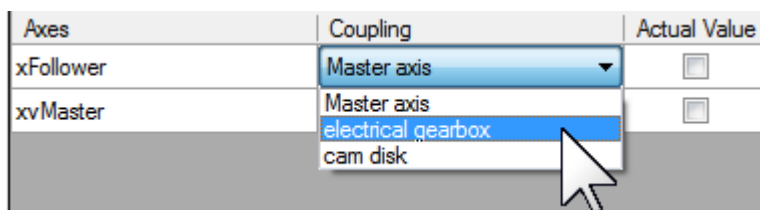
单击“OK”（3）确认您的选择。





现在，您可以设置耦合类型以及是否要为每个包含的轴使用实际值耦合。

在下面的屏幕中，将“xFollower”轴耦合类型设置为“electrical gearbox”，将“xvMaster”轴设置为“Master axis”。



这会生成以下配置：

| Axes      | Coupling           | Actual Value             |
|-----------|--------------------|--------------------------|
| xFollower | electrical gearbox | <input type="checkbox"/> |
| xvMaster  | Master axis        | <input type="checkbox"/> |

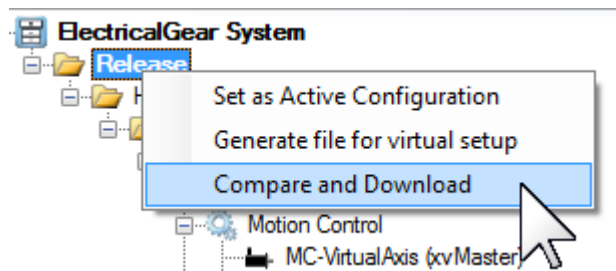
#### 请注意：

主轴不必一定在第一位置。顺序是任意的。

该配置仅支持一个主轴。 JetSym 将通过硬件树中的警告符号和鼠标提示来通知您的此类配置错误。该消息也会出现在“Compare and Download”对话框中。

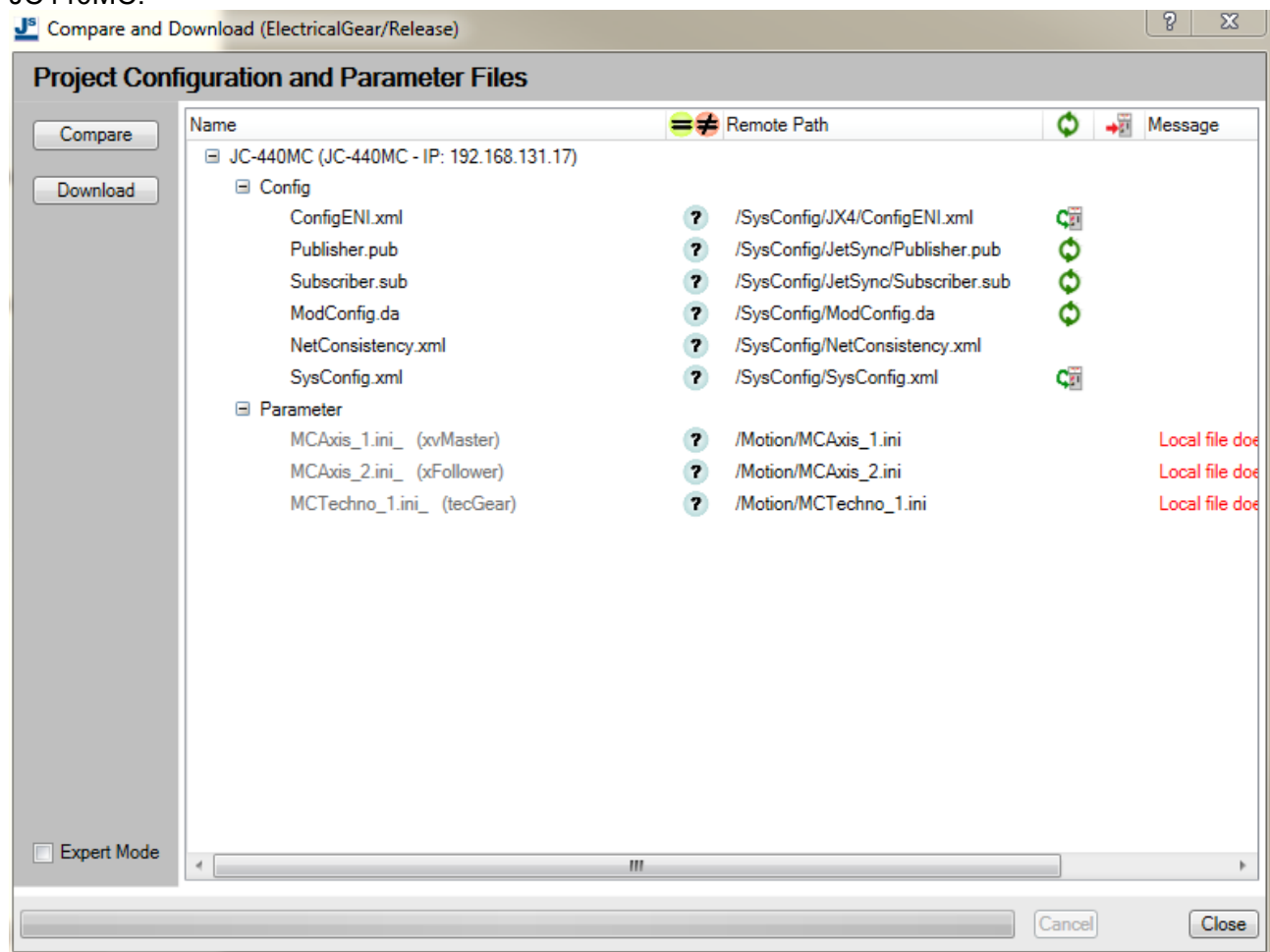
## 2.3 将 MC 文件下载到控制器

必须通过创建和下载必要的运动控制文件才能让控制器识别新的运动控制配置。重新引导后，控制器可以使用新配置。

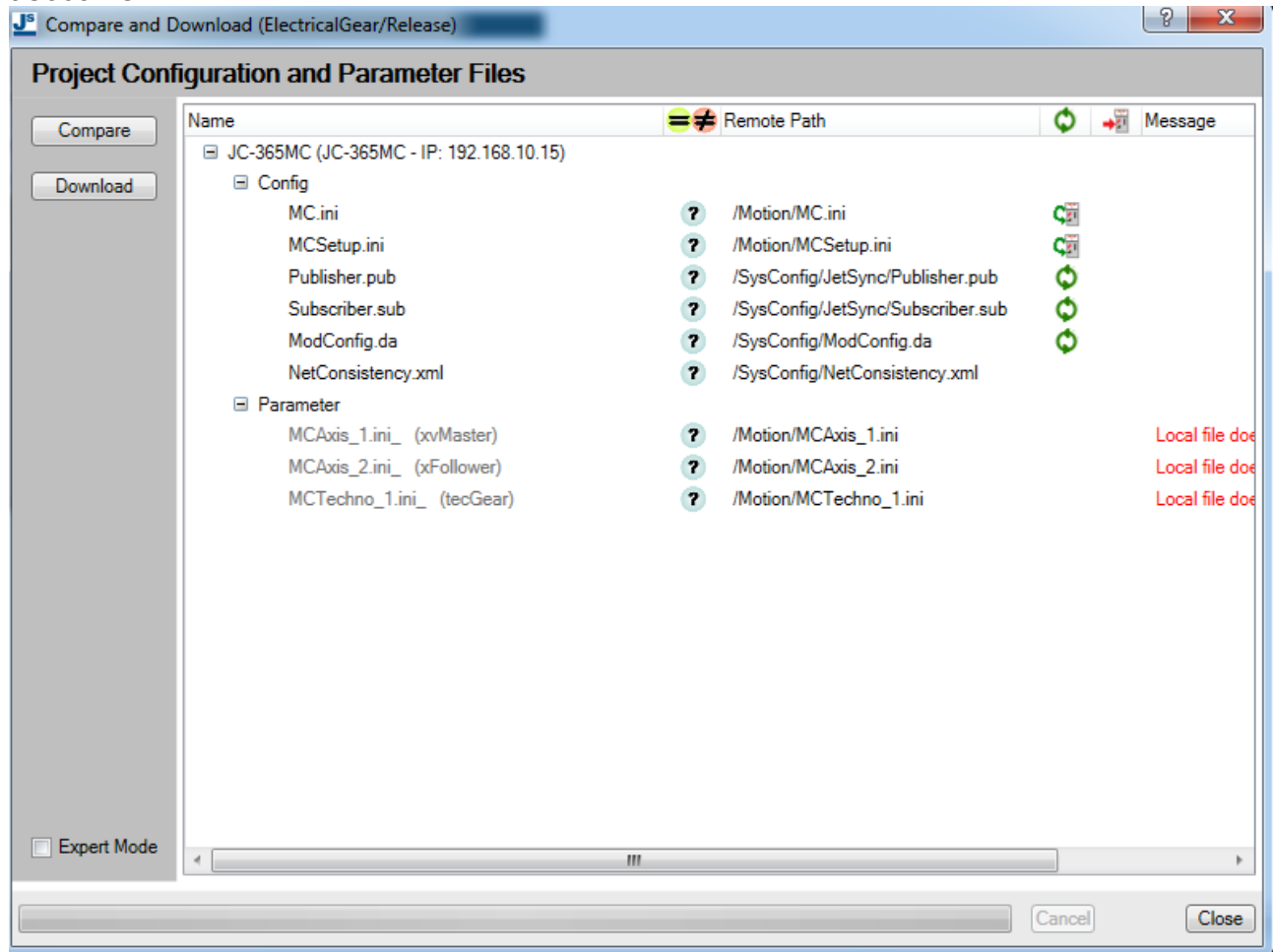


在配置文件夹的快捷菜单中选择“Compare and Download”（例如“Release”）。

JC440MC:



## JC365MC:



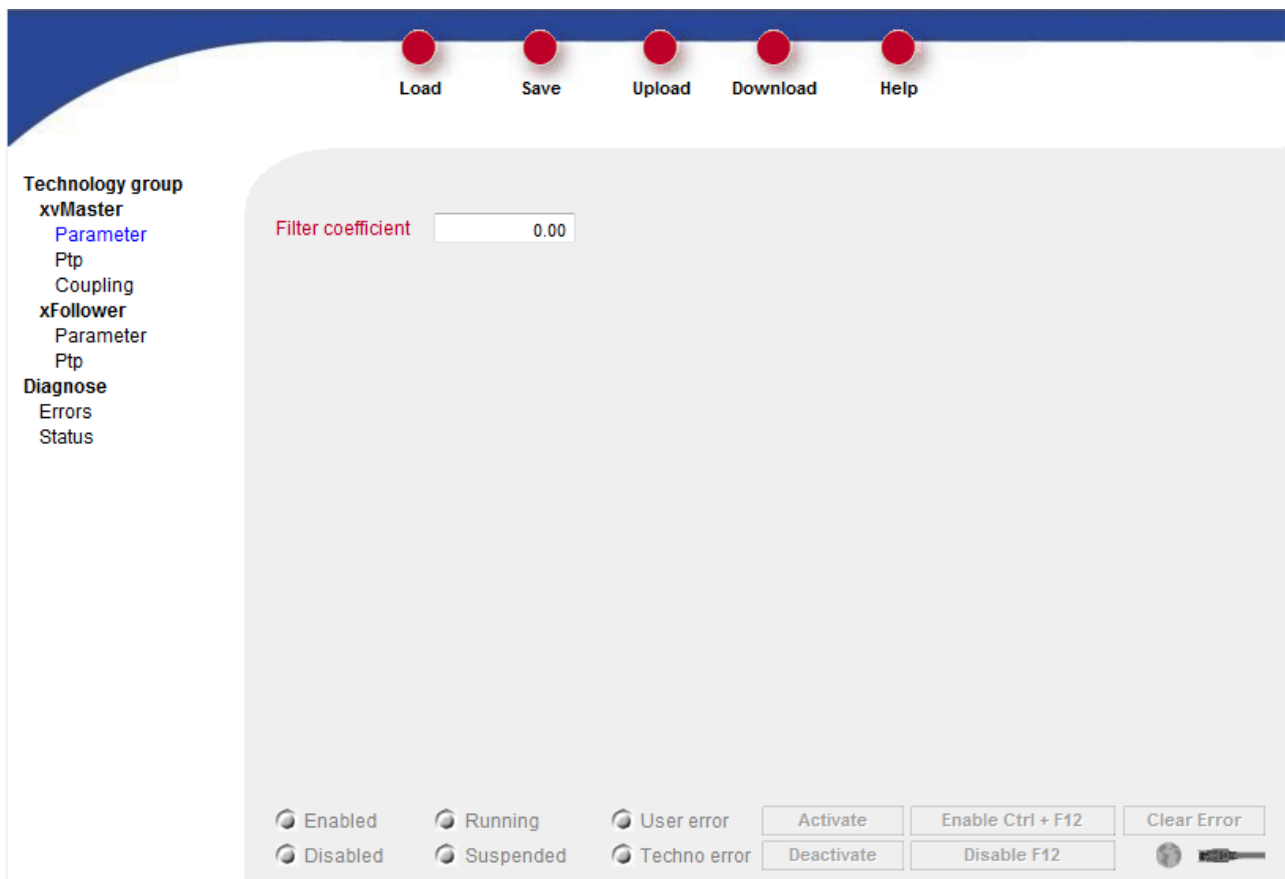
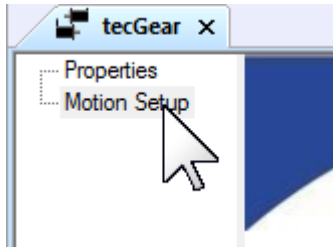
在“Compare and Download”对话框中，通过单击“Download”按钮将硬件配置传输到已配置的控制器。

如果需要重新启动控制器，则会显示一个相应的对话框，您可以选择重新启动控制器。

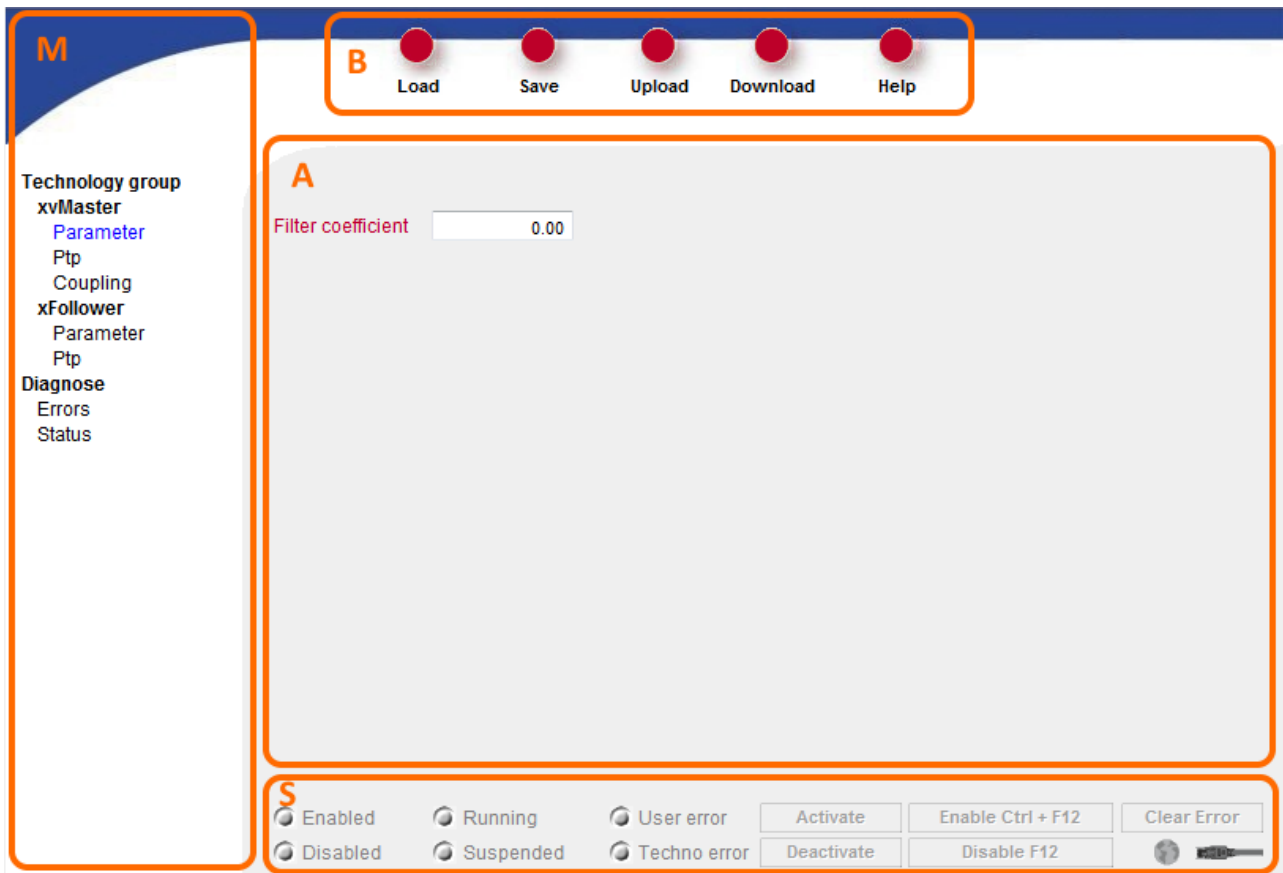
## 3 在 Motion Setup 中应用工艺组

### 3.1 操作工艺组

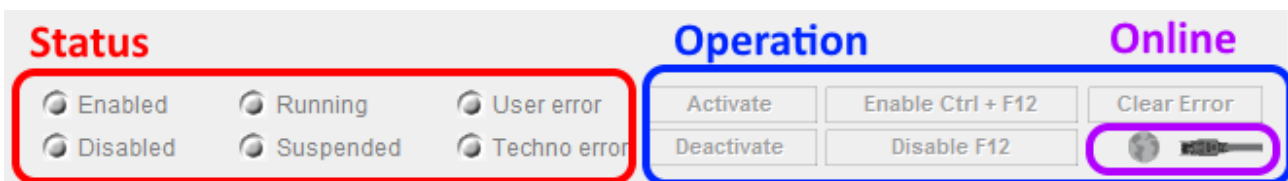
双击相应的工艺组，打开“Motion Setup”对话框。然后在配置菜单中切换到“Motion Setup”。





运动设置分为以下几个区域：



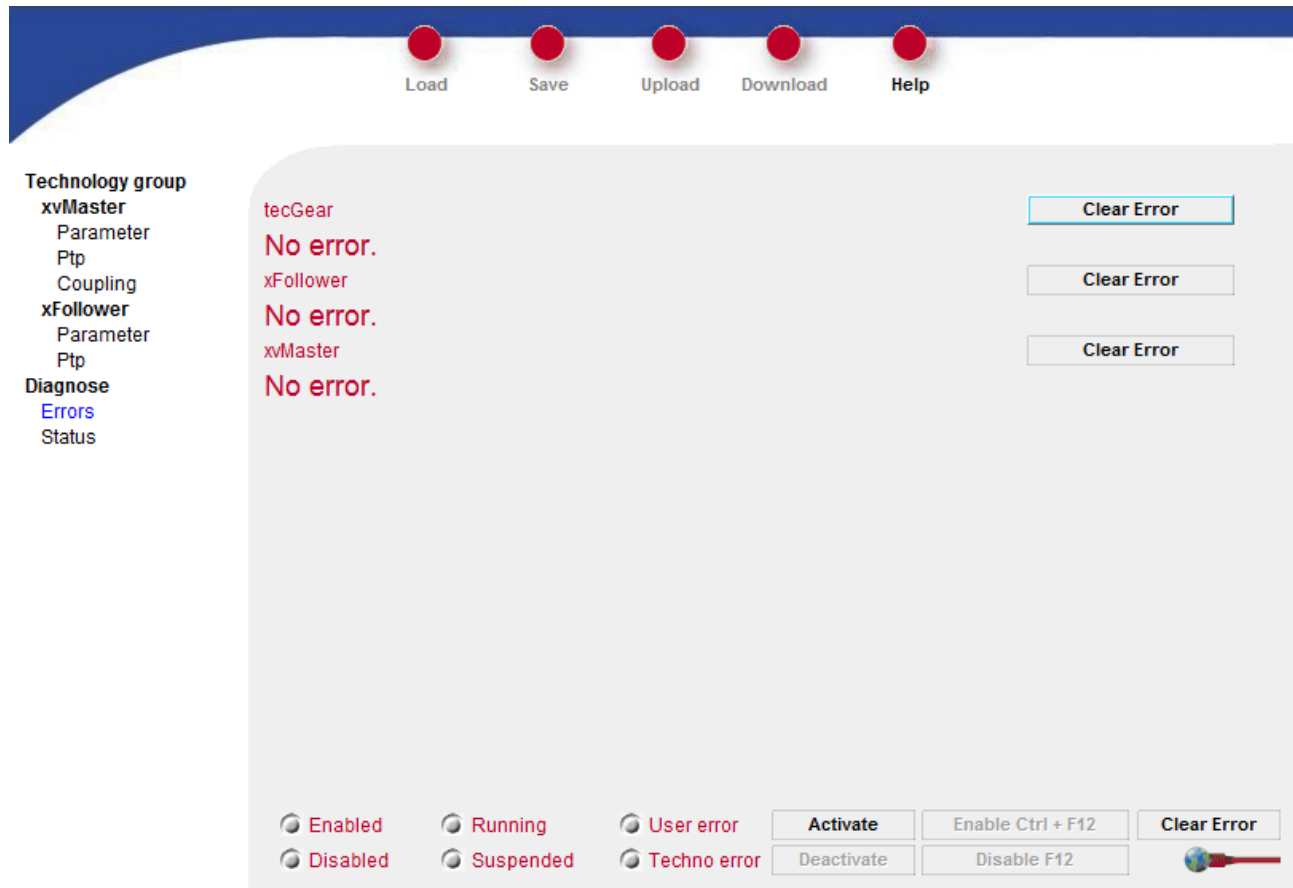
- M: 菜单-选择各个调试页面  
 B: 控制条  
 A: 指示区域  
 S: 状态



- 这些组件允许工艺组的更高层次的操作
  - 激活/停用工艺组
  - 使能/关闭（仅当组处于活动状态且没有错误时）
  - 清除错误：清除组和活动组中涉及的轴的错误
- 状态指示
  - 已使能
  - 未使能
  - 运行中
  - 挂起
  - 如果该组处于非活动状态，则状态显示为灰色
- 错误

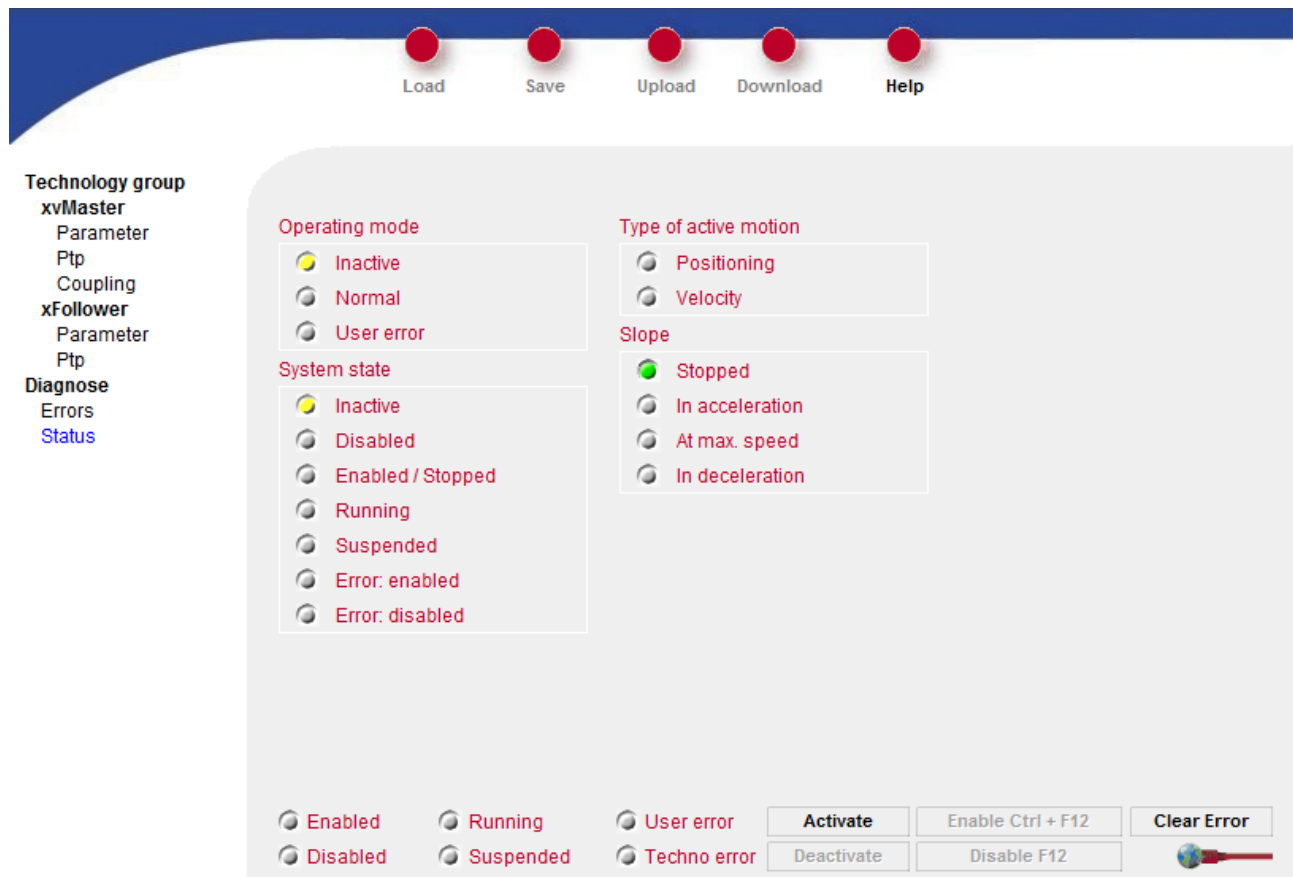
- 用户错误
- 工艺错误=工艺组错误
- 在线
  - 在线符号指示与控制器是否存在连接。
- 离线: 
- 在线: 

### 3.1.1 错误页面-概述



在这里，您可以看到该组及其成员轴的现有错误。除了可以使用控制栏中的“Clear Error”按钮清除所有错误之外，还可以在此处单独处理错误。

### 3.1.2 状态页-概述



工艺组显示操作模式，操作状态，运动类型和斜坡状态（=坡度）的状态。  
运动类型和斜坡状态指的是主轴的运动。

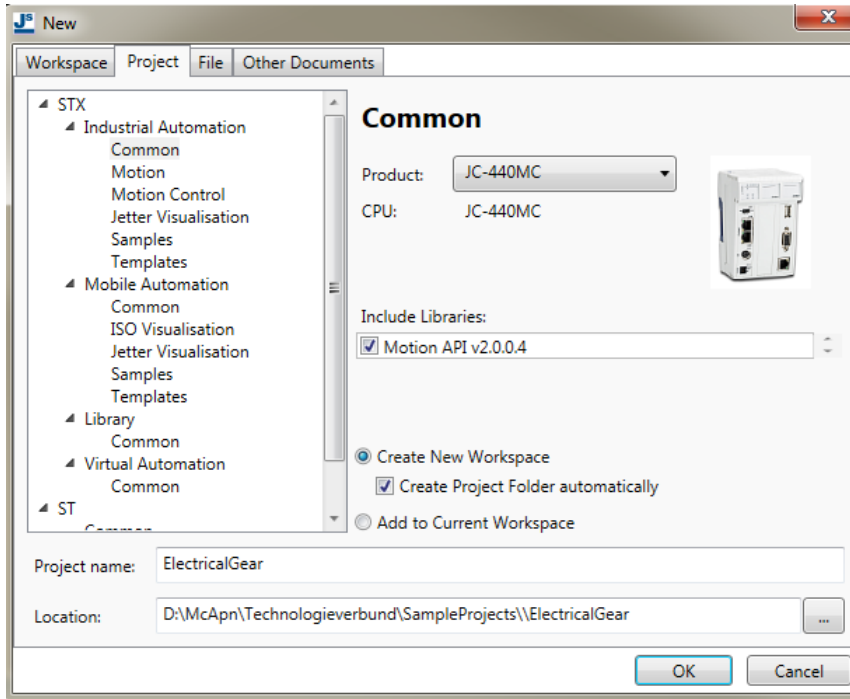
### 3.1.3 其他窗口

用于设置和测试工艺组的其他窗口将在与其相关的应用笔记中介绍。

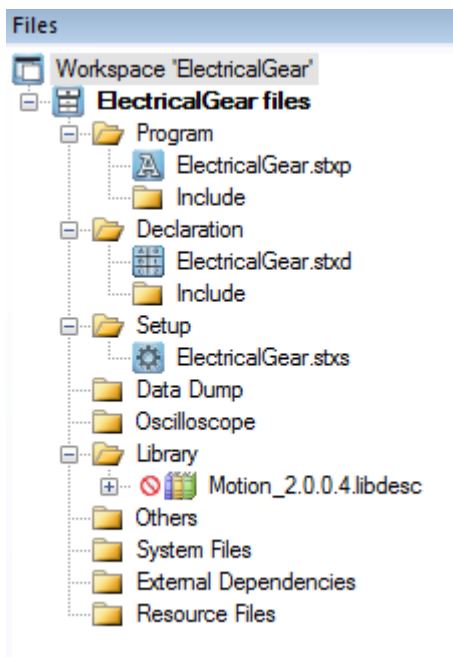
## 4 在应用程序中使用工艺组

### 4.1 将 Motion API 库集成到项目中

第一个选择是在创建新项目时直接选择 Motion API 库。



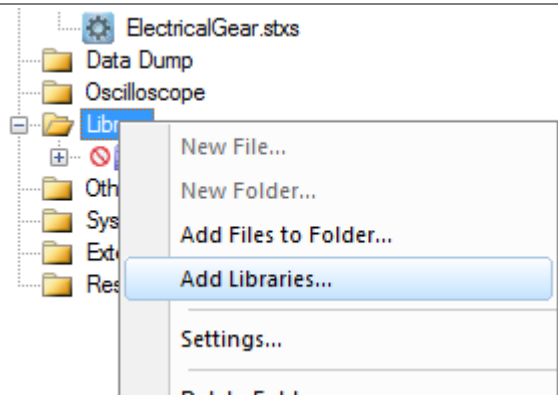
然后该库直接自动集成到项目树中，并包含在 STX 项目中。



通过创建硬件配置并使用 Motion API，将自动创建一个“HardwareConfig.stxp”，也可以通过项目设置中的“Include”设置。

在该程序文件中，将实例化运动对象并创建 MC Manager 列表。





在此视图中，还可以通过“Library”文件夹的快捷菜单更改和包含库。

一次只能包含一个 Motion API 库。如果要切换库，必须先删除现有链接，然后再添加所需的库。在此视图中删除库不会删除文件或文件系统中的库！

该文件是在编译期间创建的。为了在程序中的 IntelliSense 的帮助下使用程序中的运动对象，必须将程序编译一次。

对于没有现有代码的新项目，应创建以下简短代码以正确编译：

```
Task tMain autorun
End_Task;
```

正确编译以后，运动对象就会输入到 IntelliSense 中，它将在程序创建过程中为您提供便捷。

## 4.2 重新启动程序

启动程序时，分两种情况：

- 上电引导控制器
- 通过下载或直接通过 JetSym 重新启动应用程序

### 4.2.1 重新启动控制器

重新启动控制器时，将重新启动控制器操作系统的所有组件。这也包括运动控制内核。基于 MC.ini 和 MCSetup.ini 配置文件，运动控制内核创建其运动对象。它还连接到并同步已连接的伺服驱动器。然后，将加载与运动控制对象关联的配置文件，以便在启动应用程序时可以使用控制参数，传动比等。

通常，在启动阶段完成后，所有运动控制对象都将初始化，同步并且没有错误。当实轴，虚拟轴和仿真轴处于“locked”操作状态时，轴组处于非活动状态。这是程序运行开始的默认状态。

特殊情况：

- 运动控制内核的引导错误
  - MCSetup.ini 中的硬件配置与实际硬件不匹配，例如如果指定的轴数与所连接的驱动器不匹配，或者伺服驱动器的 IP 配置不正确。
  - 同步失败，例如由于伺服驱动器中通讯卡的操作系统错误。

- 伺服驱动器错误
  - 常见错误：
    - 编码器错误 (JM-200)：如果使用旋转变压器或 HIPERFACE 以外的编码器系统，则 JM-200 不会自动检测编码器类型，并且在初始化时提示编码器错误。可以通过“Clear Error”清除此错误，因为应该已经在 MCAxisXX.ini 中输入了正确的编码器类型。
    - 以太网同步错误 F43：主要由于重新启动控制器的时候没有同时重新启动 JM-200。重新启动控制器后，JM-200 会检测到同步丢失并发出错误 F43 信号。可以通过“Clear Error”清除该错误。

启动程序时，应考虑上述例外情况，以便一方面可以达到标准的初始状态，另一方面在出现错误的能够通知用户，以便他可以采取适当的措施。

#### 4.2.2 重新启动应用程序

请注意，重新启动应用程序时，控制器将根据配置文件加载最后可用的运动控制对象。或者是通过先前运行的应用程序，或是通过使用 Motion Setup 中配置的 Motion Control 对象。

在标准应用程序中应考虑到这一点。由于实际应用程序应从默认的初始状态开始，因此运动控制对象应转移到该初始状态。

这包括清除错误，锁定驱动器和停用组。

##### *重置运动控制对象*

重置运动控制对象的过程在启动控制器后和重新启动程序时均有效。

运动控制的结构导致可能发生这种情况的一般顺序。

- 重置组：
  - 禁用
  - 清除错误
  - 停用
- 重置轴
  - 禁用
  - 清除错误

建议在机器停止时重新启动程序或重新启动控制器。

在以下建议中，使用紧急停止斜坡来停止运动，这会导致硬性停止。如果要可控地停止运行轴，则必须相应地调整程序。在最简单的情况下，停止之前有针对性的“MoveHalt”就足够了，但是由于应用程序的各种要求，此方法会很快变得非常复杂，因此这里不能一概而论。

通常还希望再次重置在运行时或通过 Motion Setup 更改过的参数。这可以在清除所有错误后加载相应参数以实现。

#### 4.2.2.1 重新启动应用程序-建议 1:

```
//Resetting all technology groups
if not tecGear.State.IsInactive then
    if not tecGear.State.IsDisabled or not tecGear.State.IsDisabledError then
        tecGear.State.Transitions.Disable(MCStateTransitionDisableModes.VelocityControlSlope);
        when tecGear.State.IsDisabled or tecGear.State.IsDisabledError continue;
    end_if;

    tecGear.Diagnostics.ClearErrors();
    when tecGear.State.IsDisabled continue;

    tecGear.Deactivate();
    when tecGear.State.IsInactive continue;
end_if;

//Resetting all axes
if not xvMaster.State.IsDisabled or not xvMaster.State.IsDisabledError then
    xvMaster.Power.Quickstop();
    when xvMaster.State.IsDisabled or tecGear.State.IsDisabledError continue;
end_if;

xvMaster.Diagnostics.ClearErrors();
when xvMaster.State.IsDisabled continue;

//Optional
xvMaster.Settings.LoadFromFile();
when xvMaster.State.IsDisabled continue;

if not xFollower.State.IsDisabled or not xFollower.State.IsDisabledError then
    xFollower.Power.Quickstop();
    when xFollower.State.IsDisabled or xFollower.State.IsDisabledError continue;
end_if;

xFollower.Diagnostics.ClearErrors();
when xFollower.State.IsDisabled continue;

//Optional
xFollower.Settings.LoadFromFile();
when xFollower.State.IsDisabled continue;
```

在上述此示例中，为创建每个运动控制对象分别执行重置过程。

另请参见示例项目：

- [TechnologyStartUp1](#)

#### 4.2.2.2 重新启动应用程序-建议 2:

```
function TechnoResetAll()
var
```

```

nListIndex: int;
end_var;

for nListIndex := 0 to mcMgr.TechnoCount - 1 by 1 do
  if not mcMgr.TechnoList[nListIndex].State.IsInactive then
    if not mcMgr.TechnoList[nListIndex].State.IsDisabled or
      not mcMgr.TechnoList[nListIndex].State.IsDisabledError
    then
      mcMgr.TechnoList[nListIndex].State.Transitions.Disable(
        MCStateTransitionDisableModes.VelocityControlSlope);
      when mcMgr.TechnoList[nListIndex].State.IsDisabled or
        mcMgr.TechnoList[nListIndex].State.IsDisabledError continue;
    end_if;
    mcMgr.TechnoList[nListIndex].Diagnostics.ClearErrors();
    when mcMgr.TechnoList[nListIndex].State.IsDisabled continue;

    mcMgr.TechnoList[nListIndex].Deactivate();
    when mcMgr.TechnoList[nListIndex].State.IsInactive continue;
  end_if;
end_for;
end_function;

function AxisResetAll()
var
  nListIndex: int;
end_var;

for nListIndex := 0 to mcMgr.AxisCount - 1 by 1 do
  mcMgr.AxisList[nListIndex].State.Transitions.Disable(
    MCStateTransitionDisableModes.VelocityControlSlope);
  when mcMgr.AxisList[nListIndex].State.IsDisabled or
    mcMgr.AxisList[nListIndex].State.IsDisabledError continue;
  mcMgr.AxisList[nListIndex].Diagnostics.ClearErrors();
  when mcMgr.AxisList[nListIndex].State.IsDisabled continue;
//Optional
  mcMgr.AxisList[nListIndex].Settings.LoadFromFile();
  when mcMgr.AxisList[nListIndex].State.IsDisabled continue;

end_for;
end_function;

```

在此示例中，将为创建每个运动控制对象分别执行重置例程。您可以在其他项目中将这些功能作为标准功能包括在内。

另请参见示例项目：

- [TechnologyStartUp2](#)

## 4.3 激活工艺组

在默认的初始状态下，轴组处于非活动状态。因此，操作状态为“不活动”，可以使用“tecGroup.State.IsInactive”指令进行查询。

使用“tecGroup.Activate”指令激活工艺组。根据成员轴的先前状态，可以在激活之前使用状态为“Disabled”或“Enabled”的状态更改来查询成功激活。

在激活轴组之前，将根据成员轴的状态为该轴组指定运行状态为“Enabled”或“Disabled”。如果不是所有的带状态成员轴都具有相同的状态，则激活时会显示错误。

带状态成员轴是指那些可以处于“Inactive”以外状态的轴，可以是虚拟轴，实轴或仿真轴。

外部轴或影子轴始终处于“Inactive”状态，因此不考虑在内。

### 提示：



仅当有状态的成员轴处于“Enabled”或“Disabled”状态时，才能激活轴组。例如，如果至少一个带状态成员轴处于“Running”状态，则无法激活轴组。

### 示例：

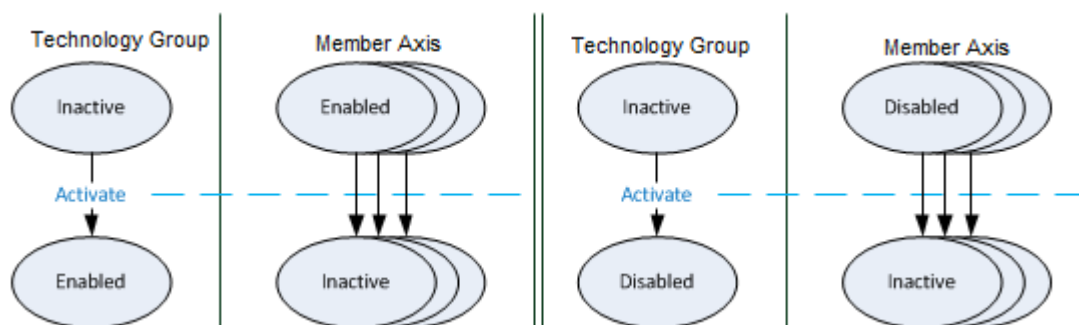
```
xvMaster.Power.Enable();
xFollower.Power.Enable();
when xvMaster.State.IsEnabled continue;
when xFollower.State.IsEnabled continue;

// Activating the group
tecGroup.Activate();
// Waiting for activation to succeed
when tecGroup.State.IsEnabled continue;
```

为简单起见，在此过程中假定成功激活。错误的处理可以在单独的任务中执行。

一个典型的错误是在若干带状态成员轴的处于不同操作状态下激活轴组。

激活轴组后，带状态成员轴的操作状态将变为“Inactive”。



激活工艺组后状态会转变

## 4.4 停用工艺组

使用“tecGroup.Deactivate”指令可以停用工艺组。可以通过查询“Inactive”来检查是否成功停用。或者，查询带状态成员轴的运行状态是否为“Enabled”或“Disabled”就足够了。

如果轴组在停用之前处于“Enabled”状态，则带状态成员轴将从“Inactive”更改为“Enabled”。

如果轴组在停用之前处于“Disabled”状态，则带状态成员轴将从“Inactive”变为“Disabled”。

注意：只有处于“Enabled”或“Disabled”状态的组才能被停用。

示例“Disabled”：

```
when tecGroup.State.IsDisabled continue;
// Deactivating the group
tecGroup.Deactivate();
// Waiting for deactivation to succeed
when tecGroup.State.IsInactive continue;
```

示例“Enabled”：

```
when tecGroup.State.IsEnabled continue;
// Deactivating the group
tecGroup.Deactivate();
// Waiting for deactivation to succeed
when xvMaster.State.IsEnabled continue;
when xFollower.State.IsEnabled continue;
```

停用轴组后，其操作状态将变为“Inactive”。

*禁用工艺组时的状态转换：*

另请参见示例项目：

- [TechnologyActivateDeactivate](#)

## 4.5 定位

当定位运动单个轴时，必须考虑将其作为单个轴还是组中的成员轴激活。

对于单个轴，定位功能直接在轴对象上调用。

示例：

```
xvMaster.MovePtp.Start(AxisPositioningModes.AbsModuloAuto, 100.0, 200.0, 1000.0, 1000.0, 1.0);
xvMaster.MoveVelocity.Start(Directions.Positive, 100.0, 200.0, 300.0);
xvMaster.MoveHalt.Start(MCAxisHaltModes.Normal, 1000.0);
xvMaster.MoveHome.Start();
xvMaster.MoveHome.SetReference();
```

如果该组处于活动状态，则通过该组调用定位功能：

示例:

```
tecGear.MovePtp.Start(xvMaster, MCTechnoPositioningModes.AbsModuloAuto, 100.0, 200.0, 1000.0,
1000.0, 1.0);
tecGear.MoveVelocity.Start(xvMaster, Directions.Positive, 100.0, 200.0, 300.0);
tecGear.MoveHalt.Start(xvMaster, MCTechnoHaltModes.Normal, 1000.0);
```

如果将定位功能发送到单轴，尽管它在一个组中处于活动状态，但会生成错误“ 10-Object inactive”，表示此处已访问了一个非活动对象。

另请参见示例项目：

- [TechnologyMoveInGroup](#)
- [TechnologyMoveNoGroup](#)

#### 4.5.1 MovePtp

单轴:

有两种方法可以查询定位是否完成：

- 操作状态查询：AxisName.State.IsEnabled
  - 执行定位后，轴的运行状态将变为“Running”，定位完成后将变为“Enabled”。
- 斜坡状态查询
  - 当执行定位时，斜坡状态从“Accelerating”，“ConstantSpeed”，“Decelerating”变为“Stopped”，表示轴已停止。
  - 加速中：AxisName.Mechanism.Slope.IsAccelerating
  - 恒定速度：AxisName.Mechanism.Slope.IsAtConstantSpeed
  - 减速中：AxisName.Mechanism.Slope.IsDecelerating
  - 已停止：AxisName.Mechanism.Slope.IsStopped

这些查询均参考轴的设定值。

实际位置是否也已到达目标窗口也应使用 Axis.IsInTargetWindow 指令检查。

通过指令例如 MoveHalt 提前停止的轴，操作状态会变为“Enabled”，斜坡状态会变为“Stopped”，但可能尚未达到目标位置！

组中涉及的轴:

在查询定位是否完成时，应仅考虑斜坡状态以及相应轴的目标窗口（如果适用）。轴组的运行状态是各个轴的运行状态的总和。一旦组中的某个成员开始运动后，操作状态就会变为“Running”，并且只有当所有成员都静止不动时，轴组的状态才会变为“Enabled”。因此，取决于应用程序，组中单个轴的状态可能是处于变化中的。

主轴的斜坡状态:

轴组本身也具有斜坡状态。这是主轴的斜坡状态！因此，在定位主轴时，斜坡状态查询也必须考虑到这一点。在轴组激活期间，主轴作为单个轴的斜坡状态始终保持“Stopped”，因此不能使用。

示例:

```
// Master axis
tecGear.MovePtp.Start(xvMaster, MCTechnoPositioningModes.AbsModuloAuto, 100.0, 200.0, 1000.0,
1000.0, 1.0);
when tecGear.Mechanism.Slope.IsStopped and xvMaster.IsInTargetWindow continue;

// Follower axis
tecGear.MovePtp.Start(xFollower, MCTechnoPositioningModes.AbsModuloAuto, 100.0, 200.0, 1000.0,
1000.0, 1.0);
when xFollower.Mechanism.Slope.IsStopped and xFollower.IsInTargetWindow continue;
```

## 4.5.2 MoveVelocity

单轴:

要查询是否开始了连续定位，可以检查运行状态“Running”。轴一但开始加速就会变为该状态。

要查询连续定位的阶段，请检查斜坡状态:

- AxisName.Mechanism.Slope.IsAccelerating
- AxisName.Mechanism.Slope.IsAtConstantSpeed

示例:

```
xvMaster.MoveVelocity.Start(Directions.Positive, 100.0, 200.0, 300.0);
when xvMaster.Mechanism.Slope.IsAtConstantSpeed continue;
```

组中涉及的轴:

与 MovePtp 一样，此处应将斜坡状态与单轴一起使用。

轴组的运行状态是各个轴的运行状态的总和。一旦组中的某个成员开始运动后，操作状态就会变为“Running”，并且只有当所有成员都静止不动时，轴组的状态才会变为“Enabled”。因此，取决于应用程序，组中单个轴的状态可能是处于变化中的。

主轴的斜坡状态:

轴组本身也具有斜坡状态。这是主轴的斜坡状态！因此，在定位主轴时，斜坡状态查询也必须考虑到这一点。在轴组激活期间，主轴作为单个轴的斜坡状态始终保持“Stopped”，因此不能使用。

示例:

```
// Master axis
tecGear.MoveVelocity.Start(xvMaster, Directions.Positive, 100.0, 200.0, 300.0);
when tecGear.Mechanism.Slope.IsAtConstantSpeed continue;

// Follower axis
tecGear.MoveVelocity.Start(xFollower, Directions.Positive, 100.0, 200.0, 300.0);
when xFollower.Mechanism.Slope.IsAtConstantSpeed continue;
```



### 4.5.3 MoveHalt

#### 单轴:

有两种查询停止是否完成的方法:

- 操作状态查询: AxisName.State.IsEnabled
  - o 执行定位后, 轴的运行状态将变为“Running”, 定位完成后将变为“Enabled”。
- 斜坡状态查询
  - o 停止过程, 当轴停止时, 斜坡状态从“Accelerating”, “ConstantSpeed”, “Decelerating”变为“Stopped”。
  - o 加速中: AxisName.Mechanism.Slope.IsAccelerating
  - o 恒定速度: AxisName.Mechanism.Slope.IsAtConstantSpeed
  - o 减速中: AxisName.Mechanism.Slope.IsDecelerating
  - o 已停止: Axis.Mechanism.Slope.IsStopped

#### 示例:

```
xvMaster.MoveHalt.Start(MCAxisHaltModes.Normal, 1000.0);
when xvMaster.Mechanism.Slope.IsStopped continue;
```

#### 组中涉及的轴:

查询停止是否完成时, 仅需考虑斜坡状态。

轴组的运行状态是各个轴的运行状态的总和。一旦组中的某个成员开始运动后, 操作状态就会变为“Running”, 并且只有当所有成员都静止不动时, 轴组的状态才会变为“Enabled”。因此, 取决于应用程序, 组中单个轴的状态可能是处于变化中的。

#### 主轴的斜坡状态:

轴组本身也具有斜坡状态。这是主轴的斜坡状态! 因此, 在定位主轴时, 斜坡状态查询也必须考虑到这一点。在轴组激活期间, 主轴作为单个轴的斜坡状态始终保持“Stopped”, 因此不能使用。

#### 示例:

```
// Master axis
tecGear.MoveHalt.Start(xvMaster, MCTechnoHaltModes.Normal, 1000.0);
when tecGear.Mechanism.Slope.IsStopped continue;

// Follower axis
tecGear.MoveHalt.Start(xFollower, MCTechnoHaltModes.Normal, 1000.0);
when xFollower.Mechanism.Slope.IsStopped continue;
```

#### 4.5.4 MoveHome

单轴:

*MoveHome.Start*

参考运行通常通过 *MoveHome.Start* 启动。

查询参考运行是否已完成时，应使用操作状态和参考状态：

- 操作状态查询：AxisName.State.IsEnabled
  - o 执行定位后，轴的运行状态将变为“Running”，完成后将变回“Enabled”。
- 查询参考运行状态
  - o 在参考运行期间，状态从“ReferenceRunStarted”，“ReferenceFound”，“SwitchFound”变为“ReferenceSet”。
  - o AxisName.MoveHome.IsReferenceRunStarted
  - o AxisName.MoveHome.IsSwitchFound：找到限位开关
  - o AxisName.MoveHome.IsReferenceFound：找到参考
  - o AxisName.MoveHome.IsReferenceSet：参考运行结束

示例：

```
xFollower.MoveHome.Start();  
when xFollower.State.IsEnabled and xFollower.MoveHome.IsReferenceSet continue;
```

*MoveHome.SetReference()*

如果仅设置参考，则查询参考状态“ReferenceSet”就足够了。

示例：

```
xvMaster.MoveHome.SetReference();  
when xvMaster.MoveHome.IsReferenceSet continue;
```

提示：

有关参考运行的详细说明，请参见 JM-200 伺服驱动器文档。

组中涉及的轴：

参考运行和设置参考在该组中不可用。

因此，在激活组之前，应单独设定所有组成员的参考。

## 4.6 诊断程序

### 提示:

有关错误诊断，警告和消息的更多信息，请阅读 Application Note 054 "Motion Control Basics"。

### 单轴

可以通过轴对象的标识符访问单个轴的诊断信息。查询和清除错误是通过可用的功能和属性完成的。

### 示例:

```
// Is there an error on the axis?
if xFollower.Diagnostics.IsErrorPending then
    // Clearing the axis error
    xFollower.Diagnostics.ClearErrors();
    // Waiting until error is cleared
    when xFollower.State.IsDisabled continue;
end_if;
```

### 工艺组

在工艺组中，使用与单轴相同的方式查询和清除错误。

如果组的成员轴有错误，例如跟踪错误，这也会使工艺组的进入错误状态。如果清除了工艺组错误，则也清除了成员轴的所有错误。

### 示例:

```
// Is there an error in the technology group?
if tecGear.Diagnostics.IsErrorPending then
    // Clearing the error of the technology group
    tecGear.Diagnostics.ClearErrors();
    // Waiting until error is cleared
    when tecGear.State.IsDisabled continue;
end_if;
```

如果关注单个成员轴的错误状态，则可以检查单轴对象。清除错误将重置相应单轴的错误。但是，仍必须明确清除工艺组的错误。

### 示例:

```
// Is there an error in the technology group?
if tecGear.Diagnostics.IsErrorPending then
    // Is there an error in the follower axis?
    if xFollower.Diagnostics.IsErrorPending then
        // Clearing the error in the follower axis
        xFollower.Diagnostics.ClearErrors();
        // Waiting until error is cleared
        when not xFollower.Diagnostics.IsErrorPending continue;
    end_if;
end_if;
```

```
// Clearing the error of the technology group
tecGear.Diagnostics.ClearErrors();
// Waiting until error is cleared
when tecGear.State.IsDisabled continue;
end_if;
```

**提示:**

这些示例假定可以清除错误。但是实际上，错误仍然可能未解决并且导致错误清除失败。例如，如果某个轴的实际编码器有缺陷，则即使在清除错误后，它仍然有缺陷。在此情况下，上述的查询将挂起在等待条件上。

Jetter AG  
Graeterstrasse 2  
71642 Ludwigsburg | Germany  
[www.jetter.de](http://www.jetter.de)

E-mail     [info@jetter.de](mailto: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](http://www.jetterat.cn)

[contact@jetterat.cn](mailto:contact@jetterat.cn)  
+86 21 5869 123

We automate your success.