运动控制产品

应用说明

把 ABB 伺服驱动器用于主编码器输入

AN00241

Rev C (CN)

利用 EtherCAT 的强大功能和灵活性，通过 ABB 伺服驱动器上的辅助编码器通道将一个或多个主编码器连接到 AC500 运动系统

介绍

可以使用 AC500 PLC（PM585 和 PM59x）通过 EtherCAT 执行 ABB 伺服驱动器的实时运动控制。在某些应用中，需将这些驱动器的运动与主编码器同步（例如，定长切断，使用一个编码器跟踪物料的送进长度）。

本应用说明详细介绍了如何使用 Automation Builder 定义相关硬件设置，以在连接的 MicroFlex e190 或 MotiFlex e180 伺服驱动器上，是由某通道作为主编码器通道，以及如何使用此编码器值创建 PLCopen 运动应用程序中的（虚拟/模拟）主轴。与使用 CD522 双通道编码器模块相比，它具有一些性能优势。

- 根据所使用的编码器通道号，编码器输入频率最高可达 8MHz（正交），而 CD522 的频率仅为 300kHz
- 编码器位置更新与 EtherCAT 周期同步（使齿轮轴运动更平稳）。
- 能够通过驱动器参数过滤探针（LATCH）数据（使 PLC 应用程序的代码开发更快更简单）。

本文档还详细介绍了如何结合 PS552-MC-E 运动库中包含的探针功能块使用 e190 和 e180 驱动器，来锁存连接的主编码器的位置。本应用说明还包含了 Automation Builder 项目的示例，以说明本应用说明涵盖的所有主题。

前提条件

软硬件需满足足下要求，以执行 EtherCAT 驱动器同步运动：

- 版本为 5860 或更新的 Mint Workbench（参见 new.abb.com/motion 了解更新的下载和支持信息）
- 固件版本为 5868 或更高的 MicroFlex e190 或 MicroFlex e180 驱动器
- 运行 Automation Builder 2.1.1 或更高版本的 PC 或笔记本电脑
- 已安装（有许可证）的 ABB PLCopen 运动控制库（PS552-MC-E v3.2.0 或更新）。
- AC500 PLC 处理器 PM585、PM590、PM591、PM592 或 PM595 中的一种（PLC 处理器应该运行 2.5.1 或更新版本的固件）。PM595 提供集成 EtherCAT 连接器（它应该运行 4.2.32.2 或更新版本的固件）。所有其它处理器需要 CM579-ECAT 通讯模块（必须运行 2.6.9 或更新版本的固件，但最好是 4.3.0.2 或更新的版本）。联系你当地的 ABB PLC 支持团队详细了解如何检查这些要求并在必要时更新，或访问 http://new.abb.com/plc/programmable-logic-controllers-plcs 并选择“软件”链接。在本应用说明的正文中，我们假设使用的是带 CM579-ETHCAT 连接器的 PM591 PLC。
- 用于把 EtherCAT 连接器连接到驱动器的直通以太网电缆（插塞式电缆）（不要在插塞和跨接电缆之间切换，因为这可能影响到 EtherCAT 上关键设备的次序）
- 应用说明 AN00205（AC500 和运动驱动器 - EtherCAT 入门指南）和随附的 Automation Builder PLC 项目的副本
- 应用说明 AN00242 的副本及其附带的导出文件
- RS422（5v 差分线路驱动器）增量式编码器

要按照基本步骤创建硬件配置并编写一些使用驱动器编码器值的 PLC 代码，只需要运行 Automation Builder 2.1.1 或更高版本的 PC 或笔记本电脑，已安装的 PS552-MC-E 运动控制库和应用说明 AN00242 中包含的导出文件。本文假设读者具有 Mint Workbench、Automation Builder、CoDeSys 和 AC500 PLC 方面的基本工作知识。如果打算执行操作，读者应该已

阅读并理解文档 AN00205 的内容（也可以从 new.abb.com/motion 下载该说明），并且已经对基于 EtherCAT 的伺服驱动器（例如 MicroFlex e190 或 MotiFlex e180）进行过调试，做好与 AC500 PLC 连接的准备。

连接主编码器

MicroFlex e190 驱动器有 3 个编码器通道。

- 通道 0 - 通用编码器输入（支持编码器，带霍尔传感器的编码器，SSI，Biss，Smartabs，Endat 2.1，Endat 2.2，Sin/Cos 和经由外部适配器的旋转变压器）。
- 通道 1 - 由快速输入 1 和 2 组成的增量编码器。在设置 ENCODERMODE（1）后，支持对这些输入的编码器型操作。
- 通道 2 - 在编码器通道 0 设置为任何数字编码器类型（SSI，Biss，Smartabs，Endat 2.2）或外部旋转变压器的适配器时，通过通用编码器输入上未使用的霍尔传感器输入实现的增量编码器

MotiFlex e180 驱动器也配有 3 个编码器通道。

- 通道 0 - 电机反馈输入（X13）。根据安装在驱动器上的特定反馈模块，编码器支持会有所不同。
- 通道 1 - 由快速输入 1 和 2 组成的增量编码器。在设置 ENCODERMODE（1）后，支持对这些输入的编码器型操作。
- 通道 2 - 增量式编码器输入（X11）

这些编码器通道中的任何一个都可以用作主编码器输入，只要它可用于连接/使用（例如，没有被用于电机反馈），并且来自要使用的编码器的信号电平与所选择的编码器通道兼容。如有必要，请参阅相应的驱动器安装手册或更多信息。

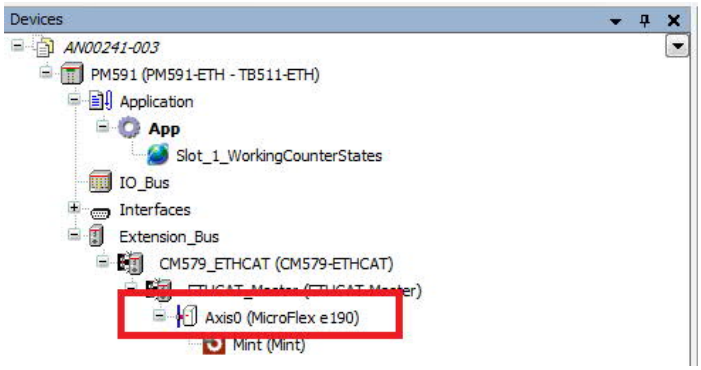
在本应用说明示例中，假设 MicroFlex e190 驱动器用于控制配备 Smartabs 反馈设备的 ESM 系列电机，并且 RS422 差分线驱动编码器通过驱动器上 X7 连接到编码器通道 2。对 MotiFlex e180，其原理是相同的，不重复叙述。

Automation Builder - 添加其他编码器映射

在本应用说明中，我们假设读者已打开 AN00205 附带的 Automation Builder 工程。我们只说明在驱动器上添加和配置/使用编码器输入所需的其他步骤。
或者，也可以打开本文档随附的示例工程。

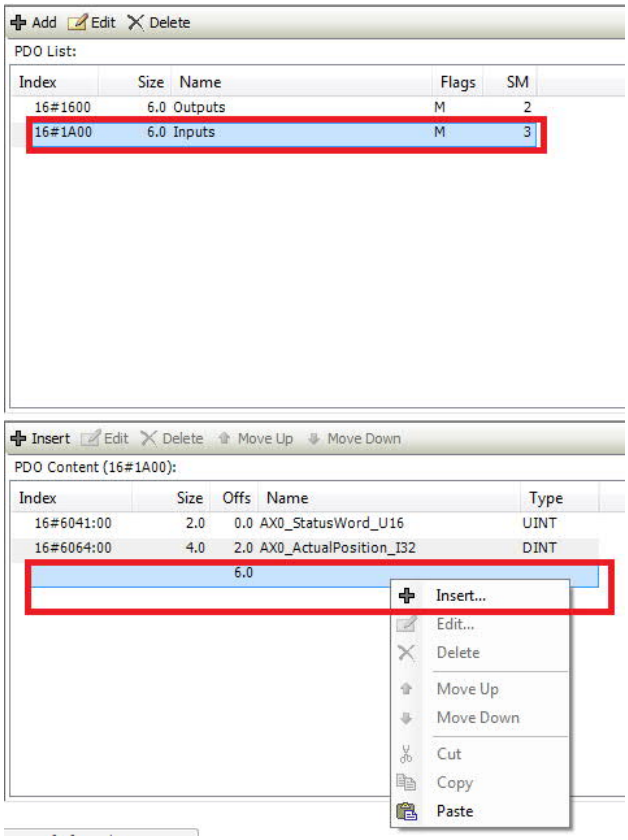
从 Automation Builder 的“File>Open Project”菜单中打开作为 AN00205 的一部分提供的 PLC 项目，打开后选择“File>Save Project As…”，为项目指定一个新名称（以避免破坏原始项目）。

如有必要，展开“设备”树，然后展开 Extension_Bus 图标及其所有子元素，直至找到 MicroFlex e190 驱动器的图标。

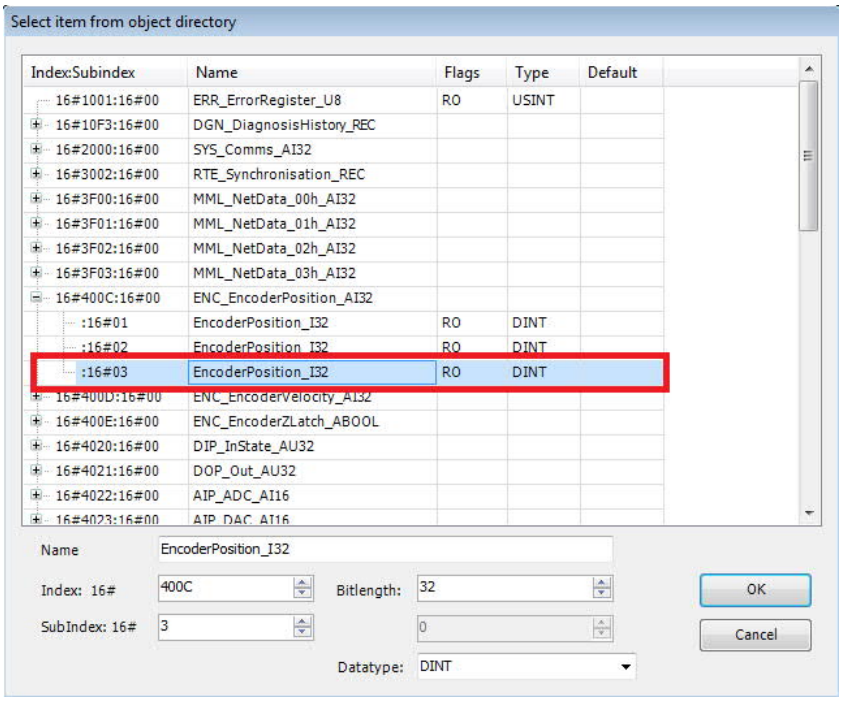


双击此图标可在右侧窗格中显示设置。需要在 General 选项卡上选择“Enable Expert Settings”（如果您从一开始就使用 AN00205 中的项目，或已打开本应用说明中包含的示例，则当前应该就是这种情况）。

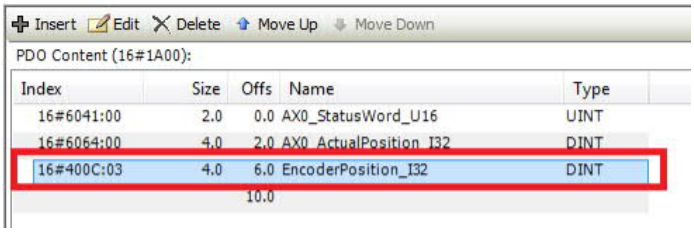
需要为编码器通道 2 添加一个新的 PDO 映射（由 PLC 从驱动器读取），因此在右侧窗格中选择“Expert Process Data”选项卡，单击右上方窗口中的“Inputs”，然后右键单击右下方窗口中“AX0_ActualPosition_I32”的 PDO 映射下方的空行。



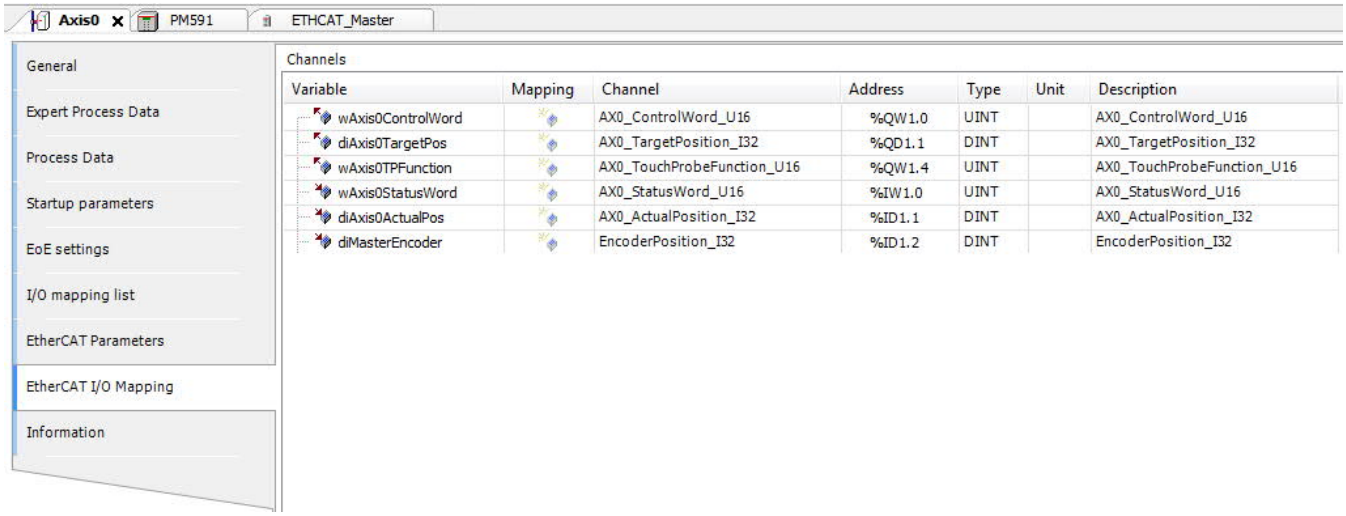
现在点击“Insert…”，并从可用对象列表中选择对象 16#400C 子索引 16#03（对象 16#400C 是“Encoder position”，子索引与每个编码器通道关联。子索引 16#01 是通道 0，子索引 16#02 是通道 1，子索引 16#03 是通道 2）。



单击“OK”后，此对象将添加到从驱动器到 PLC 的 PDO 映射列表中。



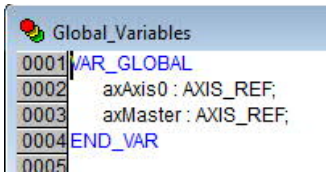
现在，需为此映射指定一个名称。因此，单击右侧窗格中的“EtherCAT I/O Mapping”选项卡，并为编码器位置映射指定一个名称。如下图所示，我们把它称为 diMasterEncoder。



保存项目并启动 CoDeSys（接受确认以更新配置）。现在，我们可以创建主轴并使用此映射编码器来设置主轴的位置。

PLC 应用程序 - 添加主轴的代码

我们需要做的第一件事是在主轴的代码中添加一个新的 AXIS_REF。转到“Resources”选项卡，打开 Global_Variables 窗口。现在我们可以为主轴添加新的 AXIS_REF 定义。如下所示，我们将命名为“axMaster”。



在计算任何齿轮轴的目标位置之前，必然需要主轴位置/轨迹，因此我们将主轴所需的代码添加到 EtherCAT_Control 程序的开头。

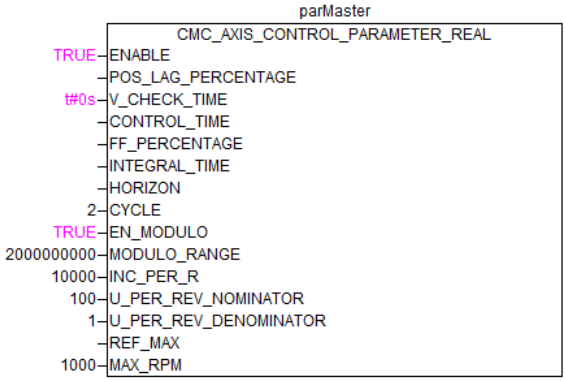
打开 EtherCAT_Control 程序（或者 EtherCAT 任务调用的任何名称的程序），选择 network 0001 并右键单击左侧的灰色区域。选择“Network before”以在程序开头插入新网络。

添加新网络 0001，并在此网络中包含 CMC_AXIS_CONTROL_PARAMETER_REAL 功能块（我们将其命名为 parMaster）。该块将为我们的主轴设置各种控制参数（比如，换算和模数）。下表详细说明了必须设置的输入参数 - 所有其他输入参数可以留空。

输入参数	描述	值/赋值	数据类型
ENABLE	启用功能块处理	True	BOOL
V_CHECK_TIME	速度监测的延迟时间	T#0s	TIME
CYCLE	轴更新的周期时间。这应该与我们在 Automation Builder 设备树中为 EtherCAT 主设备设置的 EtherCAT 周期时间一致	2	LREAL
EN_MODULO	如果轴需要被作为旋转轴（即轴位置在预先定义的范围内有环绕），或者轴位置将在环绕后最终超过 32 位位置边界（比如连续旋转的单向轴），本参数应该设置为 TRUE。如果轴只在 32 位位置范围内向前/后移动，并且没有模数功能的要求，则将本参数设置为 FALSE。对于本应用说明，我们将其设置为 TRUE，以创建一个连续向前运行的主轴（例如，如果主编码器	TRUE	BOOL

	与单向输送机一起使用）		
MODULO_RANGE	结合 EN_MODULO 使用。本参数设置一个模数周期内有多少次编码器计数。如果 EN_MODULO 设置为 FALSE 或者没有要求为主轴定义模数/周期大小，则建议将此值设置为接近但不等于最大值（2147483647）...例如，2000000000.对于主轴具有重复周期的应用（例如，用于切割或印刷机的虚拟主轴），将该值设置为主轴循环中的编码器计数（这取决于主编码器的分辨率）和机械设备与这个编码器的联动情况）。在我们的例子中，我们假设没有主循环，因此我们将此参数设置为 2000000000。	2000000000	DINT
INC_PER_R	主编码器一次旋转时的编码器计数（正交）。在我们的例子中，我们假设有一个 2500 线编码器（所以正交计数为 10000）	10000	DWORD
U_PER_REV_NOMINATOR	与下面的 DENOMINATOR 参数一起，这两个值定义主轴在主编码器的一次旋转内移动的用户单位数。在我们的例子中，我们假设主编码器的一次旋转等于主轴的 100mm 行程，因此我们可以使用 100 和 1 来表示这两个参数。	100	DINT
U_PER_REV_DENOMINATOR	见上文	1	DINT
MAX_RPM	定义主轴将行进的最大速度（以每分钟的转数为单位）。在我们的示例中，最大主速度为 60 米/分钟（1000 毫米/秒）。因为我们之前的换算设置为编码器一次旋转等于的 100mm 行程，因此最大速度为每秒 10 转（600rpm）。我们将设置 1000rpm，以留出一些余量（例如，如果我们要在主轴上执行一些相位调整运动，则可能需要这样的余量）	1000	WORD

下面的截图显示了我们已完成的功能块。



现在，我们以与实轴相同的方式使用 kernel 功能块。我们必须将 CMC_MOTION_KERNEL_REAL 功能块添加到程序中（即主轴的控制器），因此在控制参数块之后添加新网络并加入内核功能块（将其命名为 parMaster）。

下表详细说明了此块所需的输入参数。

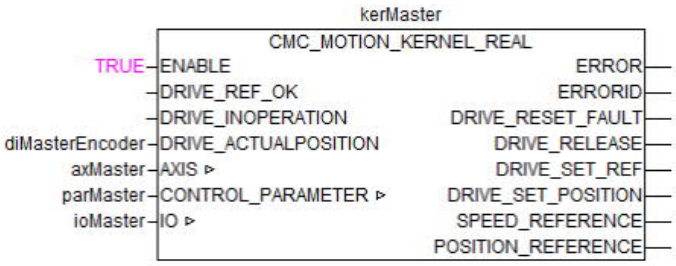
输入参数	描述	值/赋值	数据类型
ENABLE	启用功能块处理	TRUE	布尔
DRIVE_ACTUALPOSITION	主轴的位置。在本例中，它来自连接到远程驱动器的 PDO 映射编码器。	diMasterEncoder	DINT
AXIS	与 AXIS_REF 关联的内核块	axMaster	AXIS_REF
CONTROL_PARAMETER	与功能块关联的 CMC_AXIS_CONTROL_PARAMETER_REAL 功能块	parMaster	CMC_AXIS_CONTROL_PARAMETER_REAL
IO	定义轴的 IO 结构（有关详细信息，请参见 AN00205）。在本示	ioMaster	CMC_AXIS_IO

	例中，输入文字“Master_IO”并将此变量声明为 CMC_AXIS_IO 类型（与 AN00205 中的实轴一样）		

在使用实轴时，必须将内核功能块的至少一个输出参数（即 SPEED_REFERENCE 或 POSITION_REFERENCE）分配给与驱动器关联的相关 EtherCAT PDO 映射变量。

在这种情况下，没有与主轴相关的实际硬件（除连接到远程驱动器的编码器外），因此不需要为任何内核输出参数分配变量。如果用户希望检测主轴处理时发生的故障，则唯一的例外可能是 ERROR 输出（和 ERRORID）。在当前这个简单的示例中，我们决定忽略 ERROR 输出，但在实际应用中，建议将其合并到系统的错误处理逻辑中。

下面的截图显示了我们功能块。



现在，创建主轴需要的所有代码都已完成，axMaster AXIS_REF 现在可用作 PS552-MC-E 运动库提供的所有多轴 PLCopen 运动功能的主轴（例如 MC_GearIn、MC_GearInPos、MC_CamIn 等）。

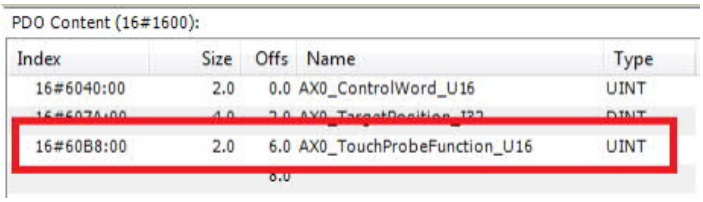
请注意，在所有情况下，在使用从远程编码器生成的主轴时，在使用多轴功能块时必须使用“mcActualValue”作为“MasterValueSource”。

PLC 应用程序 - 配置和使用探针捕获主编码器位置

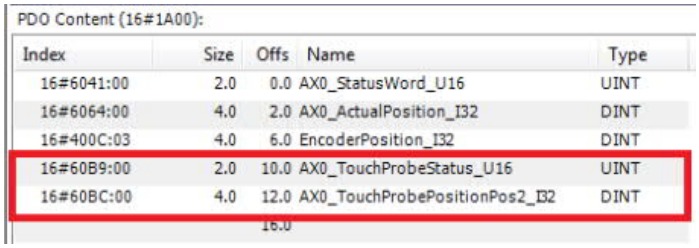
MicroFlex e190 和 MotiFlex e180 驱动器配有两个探针（快速位置捕获）对象。它们与驱动器上的两个快速数字输入相关联。探针 1 始终与数字输入 1 相关联，而探针 2 始终与数字输入 2 相关联。默认情况下，PLC 探针功能块使用驱动器捕获轴位置，但从固件版本 5860 开始，还可以配置这些驱动器通过 EtherCAT 向 PLC 提供捕获的编码器值。 有关使用探针功能块和可用过程数据对象的更多信息，请参阅应用说明 AN00221。

在本应用说明中，我们将在驱动器上使用数字输入 2（以及探针 2）来捕获我们的主编码器值。我们采用输入 2 上升边沿捕获的值。因此，下面的屏幕截图显示了我们需要在 Automation Builder 中添加到驱动器配置中的输出和输入 PDO 映射。

输出：

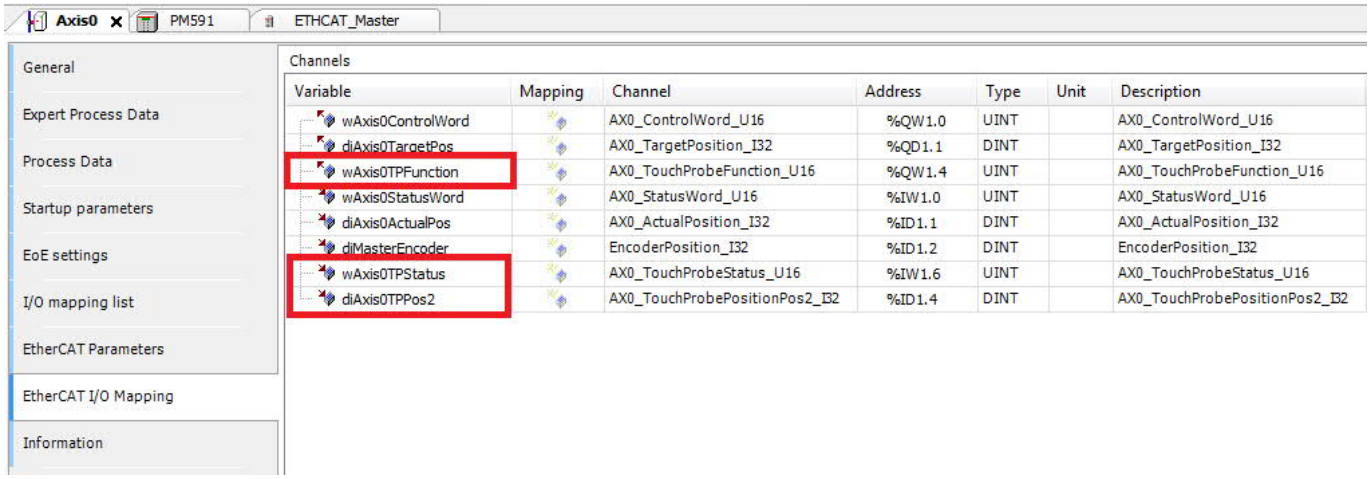


输入：



如果需要使用其中一个快速输入的数字输入 1 或负边沿（或两个边沿），只需添加必要的 PDO 映射即可访问此数据（如果需要，请再次参考 AN00221 以获取更多信息）。

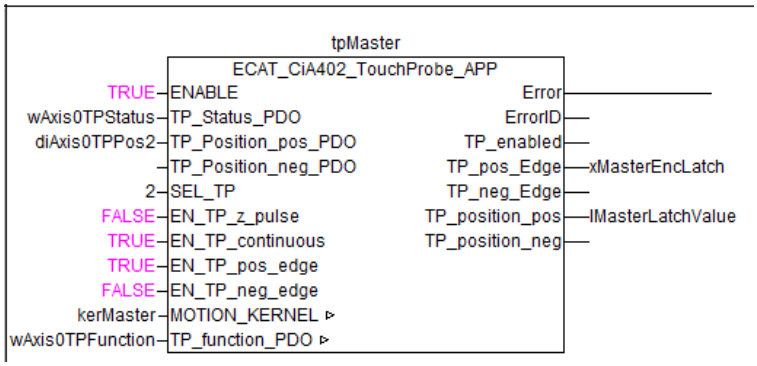
现在，我们需要通过驱动器的“EtherCAT I/O Mapping”选项卡为我们的其他 PDO 映射提供合适的名称。下面的截图显示了如何为变量命名。



确保您已从 PLC 注销。然后，如上文所示，将所需的 PDO 映射添加到 Automation Builder 配置中，然后右键单击 PLC 应用程序图标并选择“Create configuration data”，以强制更新 CoDeSys 全局变量，使其包含我们的新 PDO 映射的定义。

切换到 CoDeSys 编辑器，现在我们可以开始将探针功能块添加到 PLC 代码中。如 AN00221 所强调的那样，必须将探针功能块放在由 EtherCAT 同步调用的程序单元中。在本示例中，可以使用名为“prgEtherCATControl”的程序单元。将新网络插入 PLC 程序中包含主轴 KERNEL 功能块的网络之后，并将 ECAT_CiA402_TouchProbe_APP 块添加到此新网络。

需要配置此功能块，以使用数字输入 2 的上升沿连续查找驱动器上探针 2 的探针值。下面的截图显示了我们已经完成的网络。



请注意，我们已经使用了我们之前为需要 PDO 引用的功能块输入参数的探针 PDO 映射分配的名称。我们还添加了一个 Boolean 变量（xMasterEncLatch），用来指示是否存在新的锁存值（它只在一个程序周期内设置为 TRUE。因此，您的应用程序代码可能需要考虑在必要时设置/锁存此变量）和 LREAL 变量（lMasterLatchValue），用于存储实际的锁存数据。请注意，与主轴 KERNEL 的链接可确保此锁存数据按换算后的用户单位显示，而不是按原始编码器计数显示。

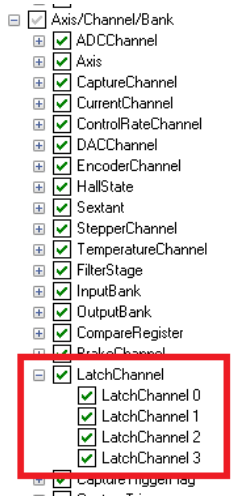
如上文所述，在默认情况下，驱动器被配置在轴的锁存**位置**。因此，为了使探针功能块显示**编码器**数据，必须配置我们的 e190 或 e180 驱动器来执行此操作。有两种方法可以实现此目标。

1. 我们可以使用驱动器参数来配置驱动器的探针/锁存通道的操作（例如，使用 Mint Workbench 配置这些设置，然后永久存储这些驱动器参数）
2. 我们可以使用来自 PLC 程序的 EtherCAT 对象写入（即 SDO 访问）功能来修改驱动器的配置 - 使用此方法时，配置通常是“易失性”的（因此，只要 PLC 程序停止、重置或者如果驱动器重新上电，驱动器就会默认返回其原始配置），并且由 PLC 应用程序代码确定何时需要调整驱动器配置。

在实践中，经常结合使用上述两种方法。用户可能希望使用驱动器的参数来永久性地配置探针 2，以操作锁存禁止值来锁存编码器通道 2（这个过滤器允许忽略后面的快速输入，直到编码器自上次锁存以来移动了指定的一段距离）。然后，可以使用来自 PLC 的 SDO 访问，使机器操作员能够在运行时调整该锁存禁止值以适应不同的操作条件。

使用 Mint Workbench 配置探针设置

使用 Mint Workbench 连接到 MicroFlex e190 或 MotiFlex e180 驱动器，在左侧屏幕的选择栏中单击“Parameters”以查看驱动器的参数表。在参数树中， Axis/Channel/Bank 部分，展开参数树的“Latch channel”部分。



如下所示，驱动器有分配给探针对象的四个锁存通道：
LatchChannel 0 = 探针 1 上升沿操作
LatchChannel 1 = 探针 1 下降沿操作
LatchChannel 2 = 探针 2 上升沿操作
LatchChannel 3 = 探针 2 下降沿操作

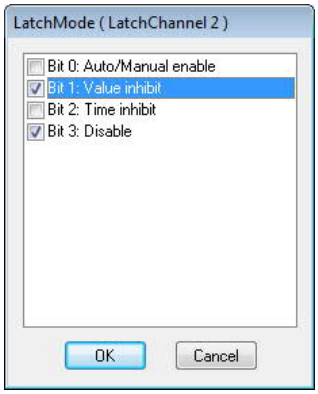
对我们在应用说明中的示例来说，我们在数字输入 2（即探针 2）上使用上升沿来捕获主编码器值。为此，单击树中的 LatchChannel 2 图标，以在参数窗口的右侧窗格中显示与此相关的参数。

此时， Latch Channel 2 的参数窗口如下图所示：

Parameter	Active
LatchEnable (LatchChannel 2)	FD 0
LatchInhibitTime (LatchChannel 2)	FD 0 ms
LatchInhibitValue (LatchChannel 2)	FD 0.0000
LatchMode (LatchChannel 2)	FD 0x0008
LatchSource (LatchChannel 2)	FD Axis position
LatchSourceChannel (LatchChannel 2)	FD 0
LatchTriggerChannel (LatchChannel 2)	C 2
LatchTriggerEdge (LatchChannel 2)	FD Positive edge
LatchTriggerMode (LatchChannel 2)	FD Digital input
LatchValue (LatchChannel 2)	RO 0.0000

您可以看到，在默认情况下，此通道被配置为从数字输入 2 的正/上升沿捕获轴位置（轴 0 的位置，由 LatchSourceChannel 设置）（其中 2 由 LatchTriggerChannel 设置）。
在 PLC 程序正在运行并且 Touchprobe 功能块被启用时，PLC 使用（我们之前映射的）Touchprobe 功能 PDO 将 LatchEnable 设置为 1，将 LatchMode 的第 0 位设置为 1 并清除 LatchMode 的第 3 位。通过所有这些操作，将启用 LatchChannel。同时，数字输入 2 上的每个上升沿将使驱动器通过我们之前映射的 TP2Pos PDO 把通道 2 的“LatchValue”传递回 PLC 的 Touchprobe 功能块。

我们需要锁存编码器通道 2（我们的主编码器），因此单击当前的 LatchSource 设置（轴位置），并将其更改为“Encoder value”。现在将 LatchSourceChannel 的值从 0 改为 2（表示编码器通道 2）。
如果要使用驱动器的内置功能来过滤锁存器，请单击现有的 LatchMode 设置，并根据需要选择“值禁止”或“时间禁止”位（保留“禁用”位，因为它由 PLC 自动控制）... 。



详细信息，请参阅 Mint 帮助系统中的 LATCHMODE。

现在输入 LatchInhibitValue 的值。因为我们正在锁存编码器，因此该值采用编码器单位（在本例中，由驱动器上的 ENCODERSCALE（2）换算）。在本例中，我们假设此编码器通道的 ENCODERSCALE 为 1（因此，它将在编码器计数时运行）。因此我们可能输入值 10000 来设置 10000 次计数的过滤器距离。

在完成所有这些操作后，您的参数看起来应该是这样的。

Parameter	Active
LatchEnable (LatchChannel 2)	FD 0
LatchInhibitTime (LatchChannel 2)	FD 0 ms
LatchInhibitValue (LatchChannel 2)	C 10000.0000
LatchMode (LatchChannel 2)	C 0x000A
LatchSource (LatchChannel 2)	C Encoder value
LatchSourceChannel (LatchChannel 2)	C 2
LatchTriggerChannel (LatchChannel 2)	C 2
LatchTriggerEdge (LatchChannel 2)	FD Positive edge
LatchTriggerMode (LatchChannel 2)	FD Digital input
LatchValue (LatchChannel 2)	RO 0.0000

现在，从 Workbench 顶部菜单中选择 Tools> Store Drive Parameters，将这些设置保存在驱动器的非易失性参数区域内。

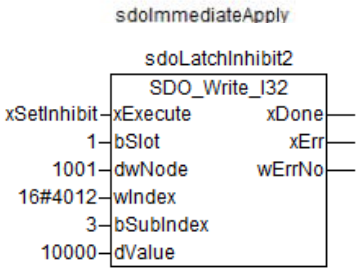
使用 PLC 代码配置探针设置

如上一节所述，不是（或除了）永久性地设置驱动器参数，还可以使用 PLC 的 SDO 对象写入在运行时修改驱动器参数。

应用说明 AN00242 详细描述了如何使用 PLC 执行 SDO 对象写入（和读取。它还提供了一个导出文件。导入该文件后，可以为 PLC 应用程序提供现成的功能块，用于访问连接的伺服驱动器中的各种对象。

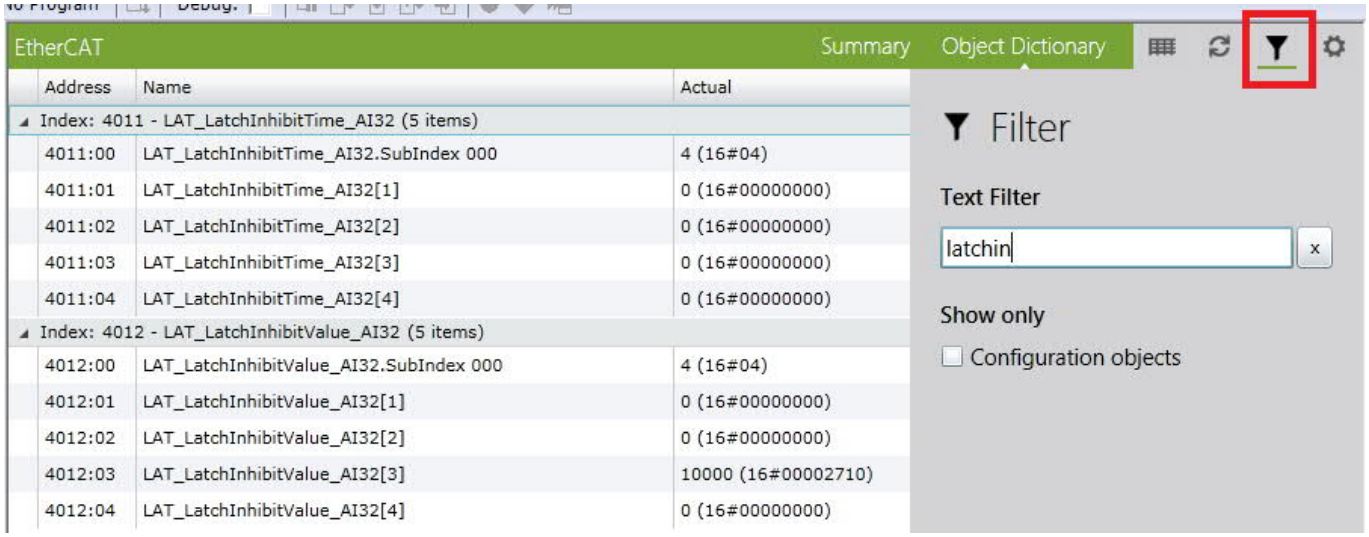
如果您还没有这些功能块，请确保已将应用笔记 AN00242 中包含的“SDO Access.exp”文件复制到硬盘驱动器上，然后在 CoDeSys 菜单中选择“Project > Import…”。如 AN00242 所述，必须将 Immediate Apply 对象（16#3004 子索引 0）设置为 TRUE，以便将写入 EtherCAT 对象的数据传递到驱动器参数表。下面的屏幕截图显示了在 FBD 中编写的 PLC 程序中的样子（注意，输入 xExecute 是上升沿触发。因此，把这些输入设置为 TRUE 将使 SDO 仅在程序首次启动时被调用一次）。

一旦立即应用设置为 TRUE，PLC 代码就可以根据需要更新 touchprobe 设置。例如，下面的屏幕截图显示了 PLC 代码如何为



LATCHINHIBITVALUE（2）设置值 10000。

请注意，此参数由 EtherCAT 对象 16#4012 子索引 03（LAT_LatchInhibitValue_AI32）控制。如果您不确定要使用哪个对象，可以使用 Workbench Object Dictionary 查看器中的 Filter 选项来搜索条目。例如，在下面的屏幕截图中，我们开始输入“latchin”来查找包含此文本的对象。



正如我们前面提到的，如果您把驱动器用于主编码器，则不太可能更改 LATCHSOURCE 和 LATCHSOURCECHANNEL 的设置。此外，硬件连接可能会确定 LATCHTRIGGERCHANNEL。因此，您最有可能使用参数查看器来配置“固定设置”，并为操作员可能想要调整以适应特定机器配置的设置配置 SDO 访问。

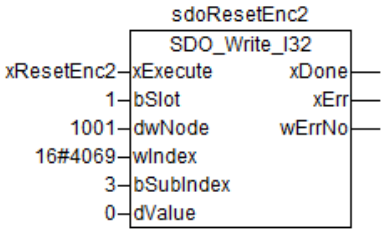
设置新的主编码器值

映射的编码器值（在使用编码器通道 2 时，为对象 0x400C 子索引 3）是只读的，因此 PLC 不可能向该对象写入值。但是，可能希望在应用的某个点重置主编码器值（比如，如果驱动器本身正在使用编码器来触发可配置的各种 SENTINEL 通道）。

为了允许这一操作，引入了额外的编码器相关对象（0x4069 ENC_ForcePosition_AI32） - 该对象的子索引与编码器位置对象（0x400C）的子索引匹配，比如，写入 0x4069 子索引 3 将导致 0x400C 子索引 3（即编码器 2）被修改。

对象 0x4069 不可进行 PDO 映射，只能通过 SDO 写入来访问。由于对象采用 I32 数据类型，因此应用说明 AN00242 中的 SDO_Write_I32 功能块可用来写入新的编码器值。

下文给出了如何将编码器 2 设置为零的例子。



请注意，如果要保留主编码器值与主轴位置之间的关系，必须在修改主编码器值的同时重置主轴位置（使用 MC_SetPosition）。

联系我们

欲了解更多信息，请联系当地的 ABB 代表，
或以下一种方式：

new.abb.com/motion
new.abb.com/drives
new.abb.com/drivespartners
new.abb.com/PLC

EtherCAT®是由德国倍福自动化有限公司许可的注册商标和专利技术。