运动控制产品

应用说明

AC500 - EtherCAT®寻零方法

AN00220

版本E（中文）

ABB AC500 PLC提供了根据应用要求在PLC或ABB运动交流伺服驱动器本身上配置寻零轨迹的灵活性。

引言

可以使用AC500 PLC（PM585和PM59x）来执行支持EtherCAT的ABB伺服驱动器的实时运动控制。本应用说明接续了AN00205（AC500 – EtherCAT入门指南）的话题，并详细介绍了如何使用Automation Builder定义适合各种EtherCAT轴寻零方法的硬件和软件配置。

前提条件

你需要配备以下项目以完成本应用说明：

- 5860或更高版本的Mint Workbench（访问www.abbmotion.com 以获取最新下载和支持信息）
- 带有5868.7或更高版本固件的MicroFlex e190驱动器（在对附带文件进行简单修改后，也能用于5868.7或更高版本的MotiFlex e180）
- 运行Automation Builder 2.1.1或更高版本的PC或笔记本电脑
- 已安装经许可的最新版本的ABB PLCopen运动控制库（3.2.0或更高版本的PS552-MC-E）的副本
- 带有CM579-ECAT通信模块的AC500 PM585或PM59x-ETH PLC（CM579-ECAT模块必须运行4.3.0.2或更高的版本的固件 – 有关如何检查该模块的详细信息，请与您当地的ABB PLC支持团队联系，必要时进行更新），或带集成EtherCAT连接器的PM595 PLC
- 用于将CM579-ECAT模块连接到驱动器的直通以太网电缆（插线电缆）（符合EtherCAT标准的规定，但这些产品使用跨接以太网电缆也可以正常工作）
- 了解通过以太网进行的AC500 PLC和ABB运动驱动器的基本操作-如有必要，请参阅应用说明AN00205了解更多详细信息

理想情况下，您还需要：

- 某种类型的数字输入模块，用于AC500 PLC（我们在项目示例中使用了DC523模块）

但是这并不重要，因为可以使用变量来模拟数字输入。本文稍后将对此进行说明。

驱动器设置

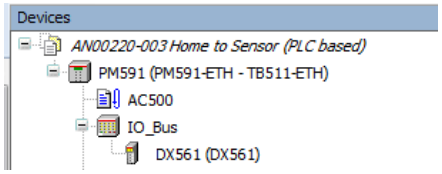
本应用说明假设您已经调试了ABB运动驱动器，并且它已加载了适当的固件。也即是说，您已经通过调试向导定义了电机和应用设置，并为驱动器自动调节了（如有必要，进行微调）控制回路。可在驱动器安装手册中找到关于驱动器调试的详细信息。本文还假设您至少已阅读并理解应用说明AN00205的内容。

为了方便起见，本应用说明中包含了示例项目（所有项目的命名格式均为“AN00220[homing type].project”）。这些项目使用MicroFlex e190驱动器，以及适用于5868.7固件的以太网ESI/XML文件。所有ABB运动驱动器的5868.7固件的ESI/XML文件都包含在本应用说明中。如果需要，可以把它们安装到Automation Builder的设备库中。本应用说明提供的示例项目包含一个可视化功能。可以通过它更轻松地测试示例函数。

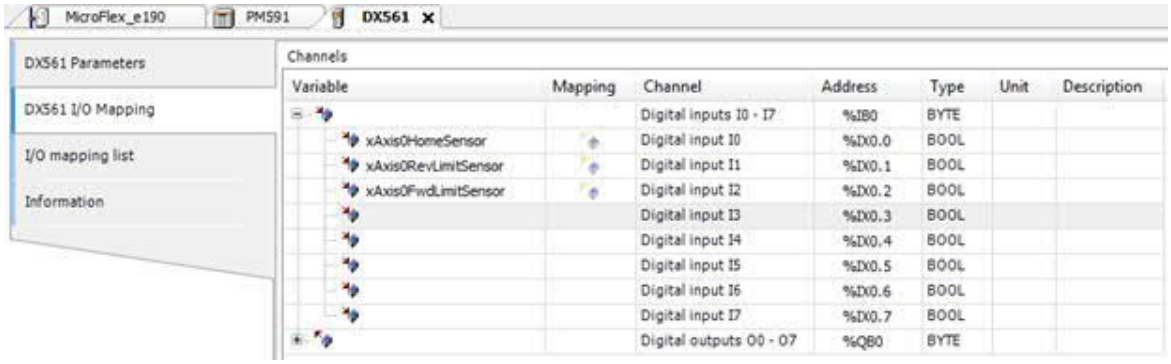
由PLC配置的寻零方法

如果PLC正在分析寻零运动，则需要由PLC自己检测零点传感器的状态。可以将传感器连接到驱动器，然后通过远程对象访问（SDO）或将PDO映射添加到驱动器输入来读取该输入的状态，但最简单的方法是把零点传感器直接连接到PLC机架的数字输入。

在我们的“寻零到传感器”项目示例中，我们为AC500 PLC系统添加了一个DX561IO模块来接受零点传感器的输入。



然后，我们为模块上用作EtherCAT轴的寻零输入（输入0）的输入命名。由于PLCopen寻零方法能够自动反转限位传感器，我们还为正向和反向限位传感器添加了变量名，以便在需要时测试此功能。

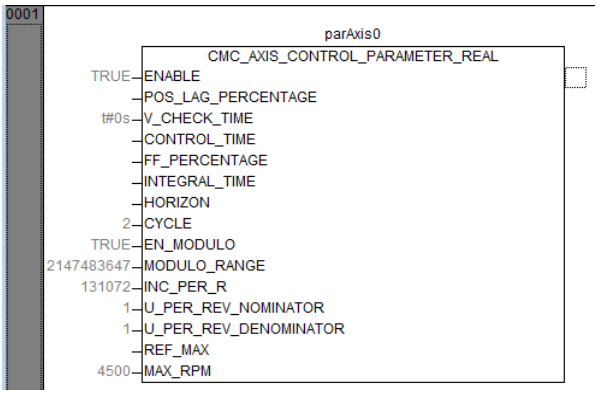


要查看示例代码，请双击Automation Builder设备树（标记为AC500）中的程序图标来启动Codesys。单击“更新”按钮，确认硬件配置已更改，并且需要重新生成（如果需要）。

在Codesys打开后，如有必要，导航到POU选项卡并打开PLC程序（prgMotion）。以下是简单的示例代码，这个程序由EtherCAT相关任务调用。现在，我们将详细介绍这个示例代码的一些主要组成部分。

换算

我们所有的示例项目都通过程序中的CMC_AXIS_CONTROL_PARAMETER_REAL函数块把轴换算成“电机转数”单位。



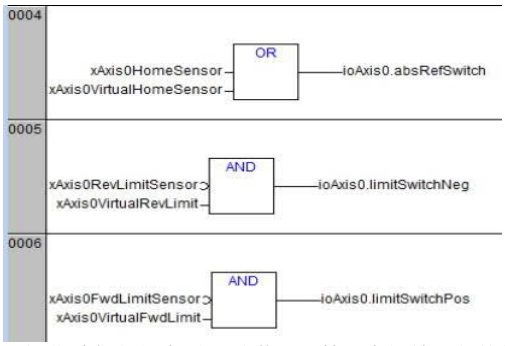
我们的项目是围绕一个MicroFlex e190演示套件设计的。演示套件使用了一个伺服电机和一个每转131072个计数的Smartabs编码器。

通过把两个U_PER_REV参数设置为1，可以把轴换算成电机的旋转次数。

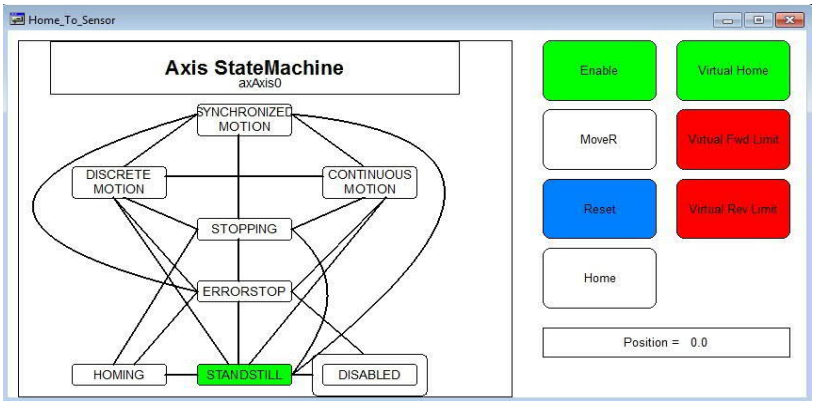
零点传感器和限位传感器

代码必须包含在prgMotion中，以使用轴的零位和限位开关的当前状态不断更新轴的IO结构。输入状态将在每个EtherCAT循环中更新。因此，最终的零位精度将取决于配置的EtherCAT循环时间和其他因素，例如PLC输入本身的反应时间、在寻零期间达到的速度和配置的减速。为了尽量提高传感器的寻零精度，必须将传感器直接与驱动器连接，并使用如后文所述的基于驱动器的寻零方法。但是，对部分应用来说，使用PLC来配置寻零轨迹可能是有利的（例如，在驱动器输入的数量有限，而可用的输入需要用于其它功能时）。

示例代码带有一个可视化功能。在无法把输入连接到当前用于测试的PLC时，可以通过本功能“虚拟”激活零点传感器和限位传感器。这些“虚拟输入”与我们示例程序中的实际物理输入并行编码。

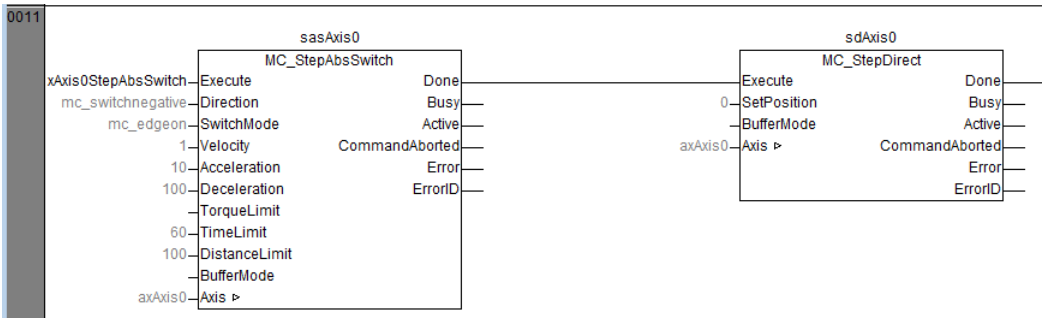


可视化功能包括在需要时激活和禁用虚拟输入的按钮（如果相应输入关闭，则按钮为红色；如果相应输入打开，则按钮为绿色）。



寻零到传感器

“寻零到传感器”项目有一个用于寻零到传感器操作的 MC_StepAbsSwitch功能块。
可视化功能中的“寻零”按钮链接到我们的MC_StepAbsSwitch实例（sasAxis0）的执行输入。



建议读者下载PLCopen运动控制标准的第5部分（寻零扩展），以供进一步参考。可以通过以下网址获取上述文档以及有关运动控制的PLCopen运动标准的其它部分：

http://www.plcopen.org/pages/tc2_motion_control/.

上面的示例代码给出了类似于Mint Home命令HOME(axis) = _hmNEGATIVE_SWITCH的结果，但是对于熟悉ABB的基于Mint的运动控制产品系列的读者来说，它没有后退操作。

请注意，当前不支持TorqueLimit输入参数，应将其留空。

如果用户在寻零过程中操作正向和反向限位开关，则PLC将根据PLCopen运动标准的要求自动改变方向。

在完成MC_StepAbsSwitch功能块（和许多其他寻零功能块）时，轴保持在寻零状态。因此，在退出该状态之前，无法进行进一步的运动（例如MC_MoveRelative）。可以通过多种方式实现这一目的：

1. 通过禁用轴的MC_Power 功能块来禁用轴
2. 通过发出MC_Halt命令
3. 通过发出MC_FinishHoming命令（目前ABB运动库不支持）
4. 通过发出MC_StepDirect命令

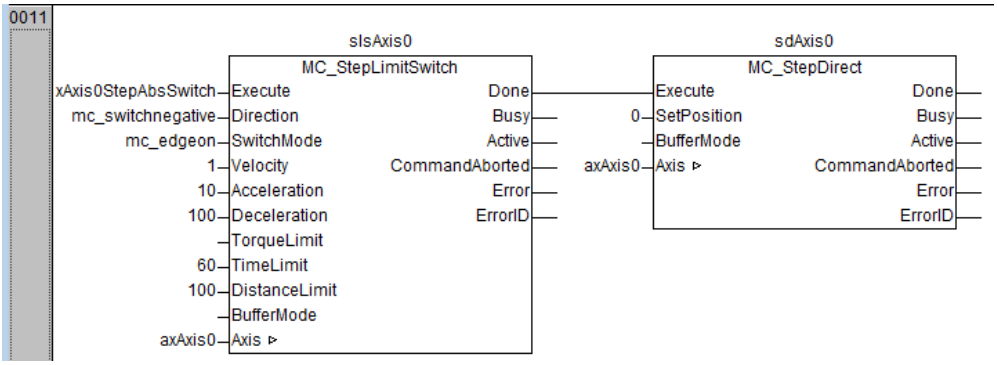
由于在寻零序列结束时设置轴位置（例如零位）是典型的操作方式，所以使用MC_StepDirect功能块最为常见。从上一个屏幕截图可以看出，该功能块已经被添加到同一梯级，并由MC_StepAbsSwitch功能块的DONE输出触发。

如果轴已经处于寻零状态（即与发送MC_StepAbsSwitch命令后的状态相同），则只能使用MC_StepDirect。在用户只想正常设置轴的位置（即在寻零过程之外）时，如果需要完成的所有操作是设置PLC的轴位置的版本，则使用MC_SetPosition。如果应用程序同时要求设置驱动器本身的位置（例如，驱动器上可能正在执行一些基于驱动器实际位置的SENTINEL式操作），则应执行基于驱动器的“在当前位置上的寻零”，而不是使用基于 PLC的一种寻零方法 – 本文稍后会对这种方法进行说明。

请注意，更详细的解决方案要求程序检查功能块的错误输出，以检测寻零过程中出现的故障（它允许在之后调用MC_Reset功能块来清除故障），但本示例至少显示了实现寻零到传感器所需的基本逻辑。在任何时候都不能同时激活两个限位开关，这将导致驱动器进入错误状态 - 以通过执行MC_Reset功能块来清除错误（如果需要，可以使用示例中的可视化工具提供的重置按钮来执行此操作）。

寻零到限位

正如轴可以寻零到一个专用的零点传感器一样，也可以把轴寻零到一个定义的限位开关输入。ABB PS552-MC库提供了一种用于MC_StepLimitSwitch的PLCopen运动寻零方法。本应用说明中的“寻零到限位”项目复制了“寻零到传感器”示例的许多特点（例如，轴仍然被换算成电机转数，并且可视化工具具有相同的特点）。但下图显示了“寻零到限位”项目代码是如何使用MC_StepLimitSwitch来代替MC_StepAbsSwitch的。



该示例被硬编码为在负方向上运行轴（功能块的方向输入被设置为mc_switchnegative）。因此，当负限位开关被激活时，表示寻零完成。

再次建议读者进一步参考PLCopen运动控制标准的第5部分（寻零扩展）。

寻零到驱动器探针（快速输入或z脉冲）

上文描述的方法对于某些应用来说可能是足够的，但是最终零位的精度在一定程度上取决于EtherCAT循环时间、PLC数字输入的反应时间、限定的减速速度和在检测到传感器水平或传感器边缘时轴的速度。为了实现非常精确的寻零，提供了使用驱动器上的探针对象（即快速中断）来使轴寻零的方法。

在这种情况下，寻零传感器必须连接到ABB运动驱动器的一个快速输入（输入1或输入2）。如果电机编码器有一个z脉冲，也可以使用它（在这种情况下，无需连接任何一个快速输入 – 驱动器将在检测到z脉冲时自动锁存轴位置）。

同样，本应用说明提供了一个示例项目来说明这种寻零方法。因为本方法使用了驱动器上的探针对象，所以现在Automation Builder的轴配置必须包括探针功能的PDO映射、状态和其中一个探针位置对象。

- DS402_TouchProbePositionPos1_I32（索引60BA）
- DS402_TouchProbePositionNeg1_I32（索引60BB）
- DS402_TouchProbePositionPos2_I32（索引60BC）
- DS402_TouchProbePositionNeg2_I32（索引60BD）

与Pos1和Neg1相关的对象与驱动器上的数字输入1捕获的快速锁存值直接相关，并且与探针1相关。
与Pos2和Neg2相关的对象与驱动器上的数字输入2捕获的快速锁存值直接相关，并且与探针2相关。
（Pos与上升沿锁存数据有关，Neg与下降沿锁存数据有关）。

如果使用来自电机编码器的Z脉冲来锁存轴位置，那么使用哪一个探针并不重要，因为Z通道始终在驱动器中被设置为通道0。

根据寻零传感器的性质（即，它是在上升沿还是下降沿激活）和驱动器上使用的输入（即，输入1或输入2）选择适当的对象（例如，我们将在驱动器上使用输入1，并假设它是在上升沿触发的，所以我们将选择DS402_TouchProbePositionPos1_I32）。

在我们寻零到探针的示例项目中，您将在Automation Builder中看到所需的驱动器PDO映射。

PDO Content (16#1600):

Index	Size	Offs	Name	Type
16#6040:00	2.0	0.0	DS402_ControlWord_U16	UINT
16#607A:00	4.0	2.0	DS402_TargetPosition_I32	DINT
16#60B8:00	2.0	6.0	DS402_TouchProbeFunction_U16	UINT
		8.0		

PDO Content (16#1A00):

Index	Size	Offs	Name	Type
16#6041:00	2.0	0.0	DS402_StatusWord_U16	UINT
16#6064:00	4.0	2.0	DS402_ActualPosition_I32	DINT
16#60B9:00	2.0	6.0	DS402_TouchProbeStatus_U16	UINT
16#60BA:00	4.0	8.0	DS402_TouchProbePositionPos1_I32	DINT
		12.0		

Automation Builder的驱动器配置页面中的EtherCAT I/O映射选项卡显示已分配给这些PDO映射的变量名。

Channels

Variable	Mapping	Channel	Address	Type	Unit	Description
wAxis0ControlWord		DS402_ControlWord_U16	%QW1.0	UINT		DS402_ControlWord_U16
diAxis0TargetPos		DS402_TargetPosition_I32	%QD1.1	DINT		DS402_TargetPosition_I32
wAxis0TPFunction		DS402_TouchProbeFunction_U16	%QW1.4	UINT		DS402_TouchProbeFunction_U16
wAxis0StatusWord		DS402_StatusWord_U16	%IW1.0	UINT		DS402_StatusWord_U16
diAxis0ActualPos		DS402_ActualPosition_I32	%ID1.1	DINT		DS402_ActualPosition_I32
wAxis0TPStatus		DS402_TouchProbeStatus_U16	%IW1.4	UINT		DS402_TouchProbeStatus_U16
diAxis0TPPos1		DS402_TouchProbePositionPos1_I32	%ID1.3	DINT		DS402_TouchProbePositionPos1_I32

应用程序代码使用一个名为ECAT_HomingOnTouchProbe_APP的ABB特定运动功能块来与名为ECAT_CiA402_TouchProbe_APP的ABB特定探针功能块一起工作。使用这些模块的两个PLC网络如下所示：

重要说明：探针和寻零功能块应始终位于EtherCAT相关任务调用的程序元素中。

在我们的示例中，探针功能块被配置为使用探针1（SEL_TP=1）。因为EN_TP_z_pulse被设置为FALSE，这会把驱动器配置为使用数字输入1而不是反馈编码器的z脉冲作为锁存机构（因为我们使用的是具有Smartabs编码器的BSM60R-240MT电机，因此没有z脉冲可用，所以我们只能使用数字输入来锁存轴位置）。

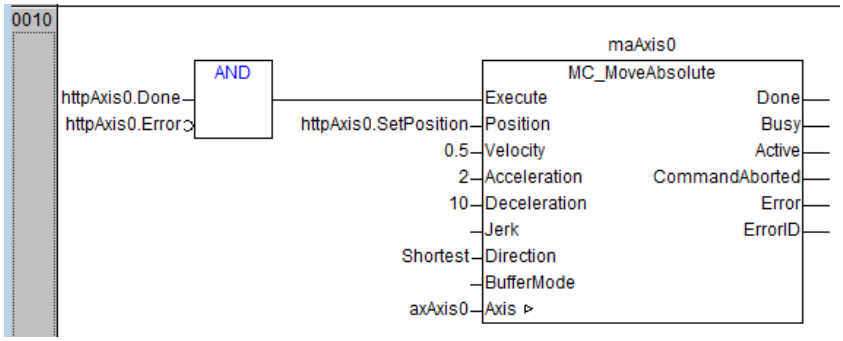
当探针功能块启用时，会触发寻零功能块（在我们示例的可视化工具中，寻零功能块又是通过激活“寻零”按钮触发的）。寻零功能块的速度、加速度、减速度和方向参数决定了PLC如何对轴上的初始运动进行轨迹配置。当寻零功能块上的TP_DONE输入激活时，此运动停止。此输入连接到探针功能块的TP_pos_Edge输出（因为我们将探针功能块配置为通过设置为TRUE的EN_TP_pos_edge输入参数从上升边沿操作，因此我们在这种情况下使用TP_pos_Edge）。

寻零功能块上的SetPosition输入参数告诉PLC当探针触发时轴应该处于什么位置。例如，如果SetPosition为零，在探针触发时，轴比记录的TP_VALUE值延迟两个单位停止（连接到探针功能块的TP_position_pos输出），则新的轴位置被设置为两个单位。

这就是在完成寻零功能块时，轴位置始终与轴已经走过的锁存位置保持的距离。

在我们的示例程序中，我们使用MC_MoveAbsolute功能块将轴移动到我们的SetPosition值（在本例中为零）。实际上，这会使轴回到锁存发生的精确位置。在此过程中，唯一的不准确是由伺服驱动器的锁存延迟引起的（最坏情况下是1微秒），因此本操作的准确度非常高。

MC_MoveAbsolute因寻零功能块被成功完成而触发（寻零功能块包括时间和距离限值输入参数。因此，如果超过其中一个限值，寻零功能块将产生错误）。



注意，在我们的例子中，SetPosition的值是零。由于我们使用了一个模数轴（因为在理论上我们的轴可以在一个方向上永远运行，并最后在32位的位置边界上回绕），那么在零位的启用轴上的“抖动”将导致测量的轴位置在零和最大回绕位置之间波动。因此，我们使用“Shortest”作为MC_MoveAbsolute的Direction输入，以确保轴总是以最短的距离移动到所需的位置。如果我们没有指定这个Direction的类型，PLC可能会把运动轨迹配置为从最大位置一直到零。

对需要在开始寻零到传感器，然后从需要由PLC配置的电机编码器中搜索索引脉冲的应用，PLC程序应结合使用我们在本应用说明中提到的MC_StepAbsSwitch和ECAT_HomingOnTouchProbe_APP。

由驱动器配置的寻零方法

如果驱动器正在分析寻零运动，则需要由驱动器自己检测零点传感器的状态。如果在应用程序中使用正向和反向限位，则还需要将这些限位连接到驱动器（例如，这样寻零程序可以根据需要自动退出这些限位）。应用程序还可能要求PLC检测正向和反向限位输入的状态（例如，中断非寻零类型的运动）。在这些情况下，可以将限位传感器连接到驱动器，然后通过远程对象访问（SDO）或将PDO映射添加到DIP_InState_AU32（对象16#4020子索引01）来读取这些输入的状态。然后，如上面的例子所示相同，PLC代码可以将从驱动器读取的值分配给它的本地轴IO结构。

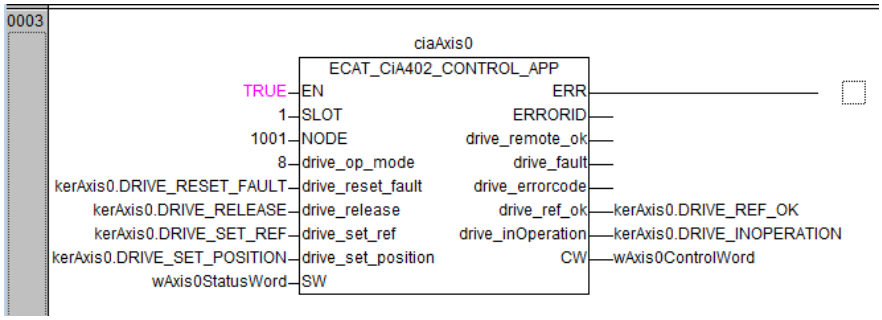
在我们的示例项目中，我们使用Mint Workbench把输入3配置为驱动器上的主输入（HOMEINPUT = 3）、输入4作为正向限位输入（LIMITFORWARDINPUT = 4）和输入5作为反向输入（LIMITREVERSEINPUT = 5）。也可以使用PLC的SDO访问来配置这些参数 - 有关更多详细信息，请参阅应用说明AN00242。

为执行基于驱动器的寻零操作，必须包括从驱动器回到PLC的附加PDO映射。AX0_ModesOfOperationDisplay_I8对象（16#6061子索引00）作为附加输出PDO包含在我们的示例项目中。

PDO Content (16#1A00):

Index	Size	Offs	Name	Type
16#6041:00	2.0	0.0	AX0_StatusWord_U16	UINT
16#6064:00	4.0	2.0	AX0_ActualPosition_I32	DINT
16#6061:00	1.0	6.0	AX0_ModesOfOperationDisplay_I8	SINT

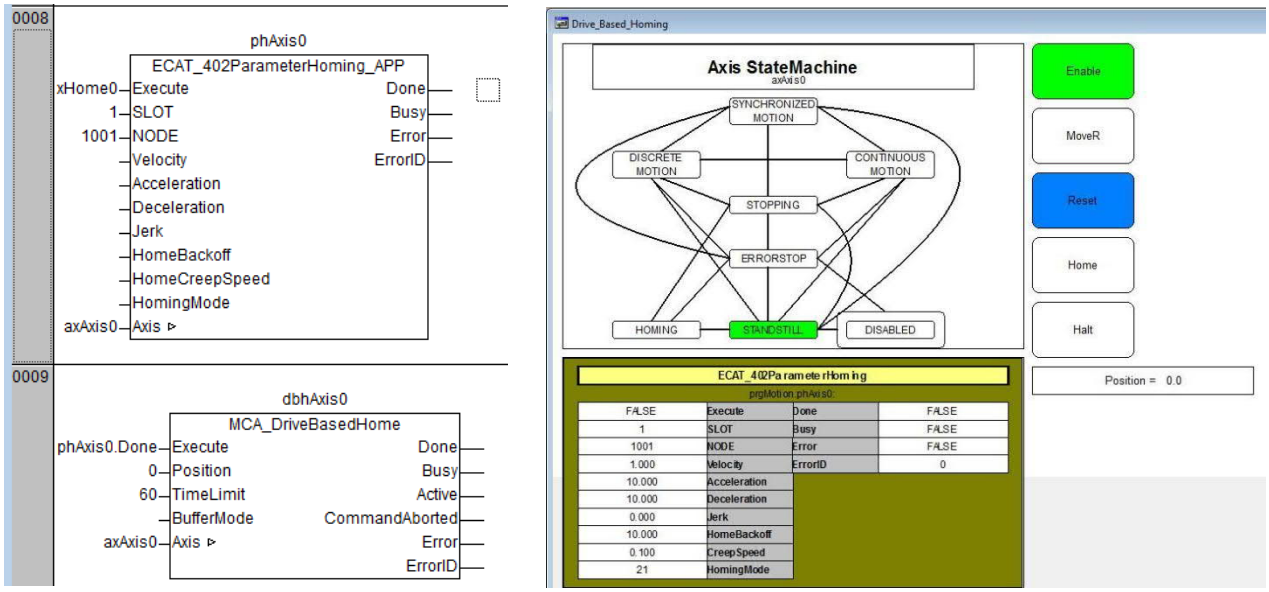
由于PLC需要能够在其启动操作模式（如循环同步位置，即模式8）和寻零模式之间切换驱动器，因此有必要把启动模式作为drive_op_mode输入参数输入到ECAT_CIA40_CONTROL_APP功能块，并将drive_set_ref和drive_set_position输入连接到关联的内核输出，如下文所示。



3.2.0版的PS552-MC PLCopen运动库提供了两个新功能块：

- ECAT_402ParameterHoming_APP：本功能块允许PLC（通过SDO）写入驱动器的寻零参数，并允许驱动器使用寻零速度、寻零加速度和减速度等参数，以及DS402寻零方法。注意，驱动器的本地PROFILEMODE参数必须设置为s-ramp，才能使Jerk参数生效，否则寻零将使用梯形轨迹配置
- MCA_DriveBasedHome：该功能块启动驱动器上的本地序列，并设置驱动器在寻零结束时所需的位置（之后，PLC也将其视为是轴的位置）。还可以指定寻零程序完成的时间限制（以秒为单位）。如果驱动器未在这段时间内寻零，则中止运动，并且功能块报告错误。

本应用说明中包含的示例项目说明了这两个模块的使用方法。



示例中的可视化工具允许用户设置寻零参数。不需要在可视化工具中设置Execute输入，单击“Home”按钮将把寻零参数写入驱动器。在成功完成此操作时，将触发寻零序列（驱动器的七段显示屏将变为“h”，表示正在进行寻零。在寻零完成时，它将返回指示启动操作模式 – 例如，“P”表示循环同步位置模式）。

注意：

1. 要使任何基于驱动器的寻零方法生效，驱动器/轴必须处于启用状态。
2. 所有的寻零参数由PLC根据轴位置换算系数来换算（即使驱动器本身为位置、速度和加速度提供了单独的换算系数）。如果驱动器被配置为对这些属性使用不同的换算系数（例如，位置为mm，速度为rpm，而不是mm/s），则在设置寻零参数时必须考虑到这一点

有关支持的基于驱动器的寻零方法的详细信息，请参阅以下部分。

可用寻零方法的总结

基于PLC的寻零方法		
零点类型	*说明*	*注释*
MC_StepAbsRefSwitch	寻零到寻零传感器	不支持转矩限制参数或开关模式MC_Off、MC_EdgeOff、MC_edgeswitch positive或MC_EdgeSwitchNegative
MC_StepLimitSwitch	寻零到正向或反向限位开关	同上
MC_StepDirect	设置轴位置的PLC版本，退出寻零状态	结合MC_StepAbsRefSwitch或MC_StepLimitSwitch使用
ECAT_HomingOnTouchProbe_AP	寻零到驱动器上的快速输入或z脉冲锁存的轴位置	结合ECAT_CiA402_TouchProbe_APP使用

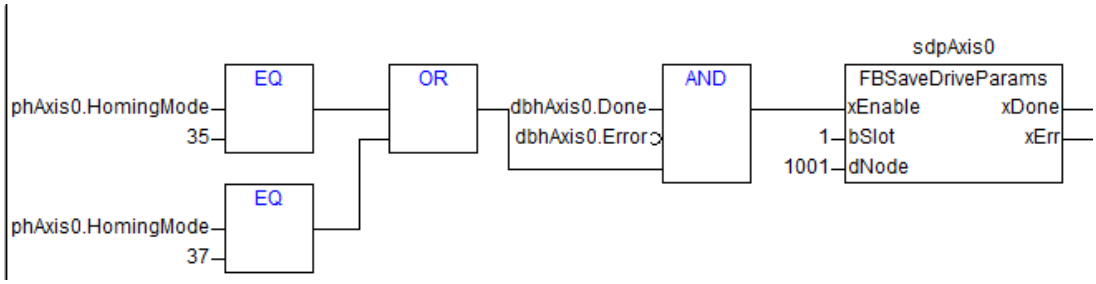
基于驱动器（DS402）的寻零方法		
零点类型	*说明*	*注释*
1	负限位开关和索引脉冲	确保反馈设备支持索引脉冲
2	正限位开关和索引脉冲	确保反馈设备支持索引脉冲
3	正零点开关未激活，且使用索引脉冲	确保反馈设备支持索引脉冲
4	正零点开关激活，且使用索引脉冲	确保反馈设备支持索引脉冲
5	负零点开关未激活，且使用索引脉冲	确保反馈设备支持索引脉冲
6	负零点开关激活，且使用索引脉冲	确保反馈设备支持索引脉冲
7	通过寻零开关寻零（未激活），索引脉冲指示正初始方向。如果达到正限位开关，则反转方向	确保反馈设备支持索引脉冲
8	通过寻零开关寻零（激活），索引脉冲指示正初始方向。如果达到正限位开关，则反转方向	确保反馈设备支持索引脉冲
9	通过反向寻零开关寻零（激活），索引脉冲指示正初始方向。如果达到正限位开关，则反转方向	确保反馈设备支持索引脉冲
10	通过反向寻零开关寻零（未激活），索引脉冲指示正初始方向。如果达到正限位开关，则反转方向	确保反馈设备支持索引脉冲
11	通过寻零开关寻零（未激活），索引脉冲指示负初始方向。如果达到负限位开关，则反转方向	确保反馈设备支持索引脉冲
12	通过寻零开关寻零（激活），索引脉冲指示负初始方向。如果达到负限位开关，则反转方向	确保反馈设备支持索引脉冲
13	通过反向寻零开关寻零（激活），索引脉冲指示负初始方向。如果达到负限位开关，则反转方向	确保反馈设备支持索引脉冲
14	通过反向寻零开关寻零（未激活），索引脉冲指示负初始方向。如果达到负限位开关，则反转方向	确保反馈设备支持索引脉冲
17	通过负限位开关寻零	
18	通过正限位开关寻零	
19	通过正寻零开关寻零（未激活）	
20	通过正寻零开关寻零（激活）	
21	通过负寻零开关寻零（未激活）	
22	通过负寻零开关寻零（激活）	
23	通过寻零开关寻零（未激活），初始方向为正。如果达到正限位开关，则反转方向	
24	通过寻零开关寻零（激活），初始方向为正。如果达到正限位开关，则反转方向	
25	通过反向寻零开关寻零（激活），初始方向为正。如果达到正限位开关，则反转方向	
26	通过反向寻零开关寻零（未激活），初始方向为正。如果达到正限位开关，则反转方向	
27	通过寻零开关寻零（未激活），初始方向为负。如果达到负限位开关，则反转方向	
28	通过寻零开关寻零（激活），初始方向为负。如果达到负限位开关，则反转方向	
29	通过反向寻零开关寻零（激活），初始方向为负。如果达到负限位开关，则反转方向	
30	通过反向寻零开关寻零（未激活），初始方向为负。如果达到负限位开关，则反转方向	
33	通过索引脉冲寻零，初始方向为负	
34	通过索引脉冲寻零，初始方向为正	
35	通过当前位置寻零	支持，但这是一个传统方法，现在已被寻零方法37所取代
36	通过探针寻零	当前不支持 - 改为使用基于PLC的ECAT_HomingOnTouchProbe_APP功能
37	通过当前位置寻零	

保存绝对位置

寻零方法37（或传统方法35）用于将当前驱动器位置设置为定义的初始位置值（功能块MCA_DriveBasedHome的Position输入参数的值）。如果轴使用绝对编码器（例如Biss、Smartabs、SSI、Endat、旋转变压器、Hiperface、Hiperface DSL），则某些应用可能要求在某个点“传递”轴位置（即为轴设置新的参考位置）。

可以通过使用方法37（或35）执行基于驱动器的寻零，然后将“evas”（之后写入“save”）写入驱动器上的对象16#1010来实现。这将强制驱动器保存产生的位置偏移，从而永久存储新的绝对位置。

本应用程序说明中包含的示例项目有演示这一操作的代码。可视化工具包括一个名为“Set Home Pos”的按钮，当它被激活（按下状态）并初始化寻零序列（通过可视化中的主按钮）时，如果定义的寻零方法是35或37，并且已经成功完成寻零序列，则将调用预先写入的功能块（FBSaveDriveParameters）。



联系我们

要了解更多信息，请联系您
当地的ABB代表，或使用以下一种方式：

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

© ABB公司，2016年，版权所有。保留所有权利。技术规格如有变更，恕不另行通知。

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。