

西门子 S7-1500T 通过 Profinet 控制 CMMT 插补

LkinCtrl 库（G_Code）篇



王金亮
Festo 技术支持
2020 年 4 月 23 日

关键词:

插补, LkinCtrl 库, CMMT-PN, Profinet, 1500T, Kinematics, G_Code

摘要:

本文介绍了使用西门子 S7-1500T PLC 通过 Kinematics 工艺对象控制 Festo CMMT 控制器实现插补的实例, 通讯协议为 Profinet, PLC 编程软件为 TIA Portal。文档主要内容包括 CMMT 在 FAS 中的基本配置、CMMT 在 TIA Portal 中的组态、工艺对象中的调试, G 代码导入和 DB 块编辑, LkinCtrl 库使用等。

目标群体:

本文仅针对有一定自动化设备调试基础的工程师, 需要对 Festo CMMT 伺服以及西门子运动控制器 1500T 有一定了解。

声明:

本文档为技术工程师根据官方资料和测试结果编写, 旨在指导用户快速上手使用 Festo 产品, 如果发现描述与官方正式出版物冲突, 请以正式出版物为准。

我们尽量罗列了实验室测试的软、硬件环境, 但现场设备型号可能不同, 软件/固件版本可能有差异, 请务必在理解文档内容和确保安全的前提下执行测试。

我们会持续更正和更新文档内容, 恕不另行通知。

目录

1	概述	4
2	运动机构工艺对象的基本工作原理	4
3	LkinCtrl 库简介及应用原理	5
3.1	简介	5
3.2	LkinCtrl 库的应用流程和原理	5
4	软/硬件环境	6
5	FAS 中的关键设置	7
5.1	报文设置	7
5.2	AC4 相关参数	7
5.3	软限位设置	8
5.4	FAS 基本测试	8
6	TIA Portal 中的配置	8
6.1	释放样例程序	8
6.2	修改硬件组态	8
6.3	修改组态工艺对象	11
6.3.1	修改 X,Y,Z 定位轴工艺对象组态	11
6.3.2	修改运动学工艺对象组态	16
6.4	设置 OB91 的循环时间	20
7	工艺面板调试	20
7.1	单轴调试面板	20
7.2	单轴状态及故障面板	21
7.3	运动机构调试面板	21
7.4	运动机构状态及故障面板	22
7.5	运动机构轨迹监控	22
8	实例演示	23
8.1	dxg 文件转成 db 文件	23
8.2	将 db 文件导入博途并生成 DB 数据块	26
8.3	PathData 参数修改	27
8.4	重新链接新的 PathData	28
8.5	添加涂胶控制程序（样例程序中已添加）	29
8.6	控制测试	30
8.7	运动机构轨迹监控中监控轨迹	31
9	功能说明	32
9.1	G 代码转换为路径 DB 块的流程	32
9.2	PathData 结构体	32
9.3	LkinCtrl 库的运行	34
9.3.1	常用功能块介绍	34
9.3.2	常用功能	37
9.3.3	运行模式	38
9.4	Flags	39
9.4.1	工作原理	39
9.4.2	Flag 模式	40
9.5	关于对象（工件）坐标系（OCS）和运动机构坐标系（KCS）偏移补偿	43
9.6	等待指令	44
9.7	FlagOnly 指令	45
9.8	路径示教	45
10	附录 A 错误处理	47

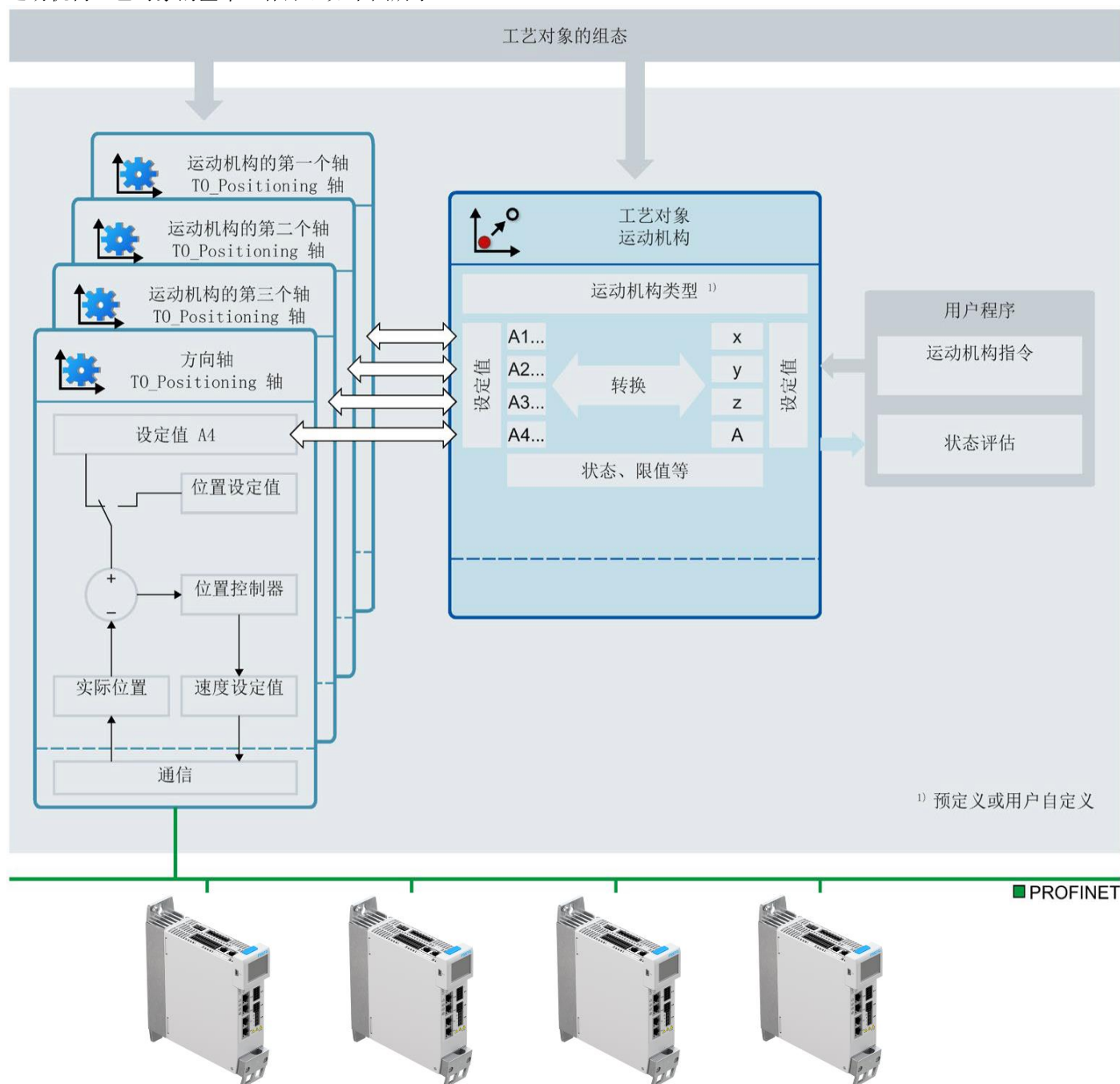
1 概述

本测试使用西门子 S7-1500T PLC 通过 Kinematics 工艺对象控制 Festo CMMT 控制器实现运动控制。基于 PROFINET，通过 105 报文使用 LKinCtrl 库控制 CMMT 实现 AC4 的插补应用。

2 运动机构工艺对象的基本工作原理

运动机构工艺对象通过预定义的类型结构，按用户指定的机械尺寸提供运动机构的正逆变换，即运动机构工艺对象根据程序命令中设置的指定目标位置，计算运动机构工具中心点（TCP）的运动设定值及运动机构各个轴的运动设定值。在博途中，可以创建“定位轴”或“同步轴”工艺对象用于与运动机构工艺对象互连，运动机构将计算出的各个轴的运动设定值传递给相应的定位轴或同步轴，

运动机构工艺对象的基本工作原理如下图所示：

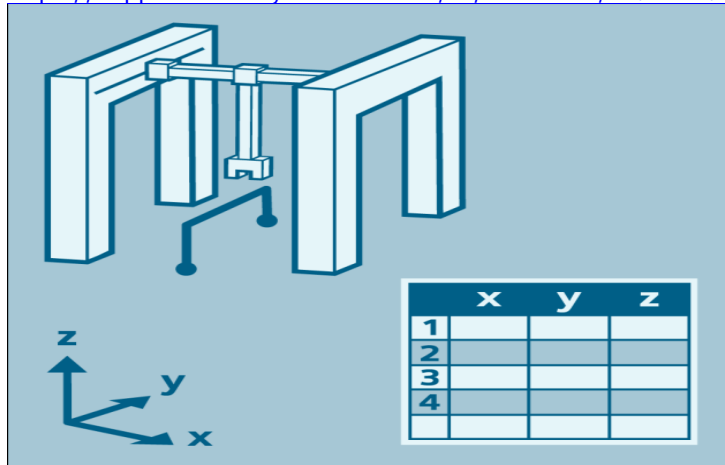


3 LKinCtrl 库简介及应用原理

3.1 简介

LKinCtrl 库为运动机构工艺对象提供了简单的编程和控制方式，它用指令列表的形式合成了多个单独的路径运动。它是由西门子提供，下载链接：

<https://support.industry.siemens.com/cs/document/109755891/simatic-s7-1500t-kinematics-control?dti=0&lc=en-WW>



MC_MovePath 指令

运动机构控制库使得用户能够轻易地通过运动机构工艺对象控制运动机构运行一个预定义好的路径运动。相比之前需要分别运行和处理单条运动指令的用户程序，现在仅需要一个核心功能块来控制运动机构工艺对象。路径轨迹信息由一个指令列表提供。支持的指令类型包括直线、圆弧的绝对或者相对插补运动等。通过输入参数，可中断、停止和恢复路径运动。该块还能提供路径运行过程中的状态和错误诊断等详细信息。

MC_JogFrame 指令

MC_JogFrame 功能块能够通过连续的、增量式的或者直接设定目标位置的方式，在笛卡尔坐标系中的 X, Y, Z 和旋转方向上点动运动机构。

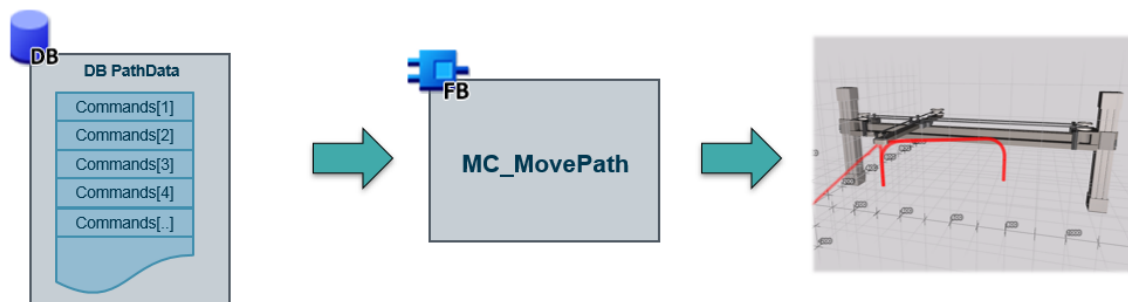
GCode2MovePath 工具

另外 Windows 环境下的工具 GCode2MovePath 能够将已有的 G 代码转换成 MC_MovePath 直接运行的指令列表。

该库可以在 2.5 及以上固件版本的所有 1500T CPU 上运行。

3.2 LKinCtrl 库的应用流程和原理

下图展示了应用的流程和原理。



user defined path motion in
commandList

PLCopen conform FB
controls TO_Kinematics

easily run a kinematics with a
minimum of programming

路径运动通过一个可参数化的指令列表来定义，这个指令列表就叫做 PathData，它有几种不同等级的内部数据，已经作为库的标准数据类型供直接使用。PathData 结构的参数化遵循系统功能包括运动机构工艺对象支持的系统运控控制指令。PathData 里的指令数量可以根据需要通过库的内部用户常量来做调整。

路径数据是核心功能块 LKinCtrl_MC_MovePath 的 InOut 参数。该功能块负责控制运动机构工艺对象并在内部执行 PathData 中定义的运动指令。而且，该功能块能够通过接口参数中断，恢复和停止路径运动，还能给出详细的诊断信息。

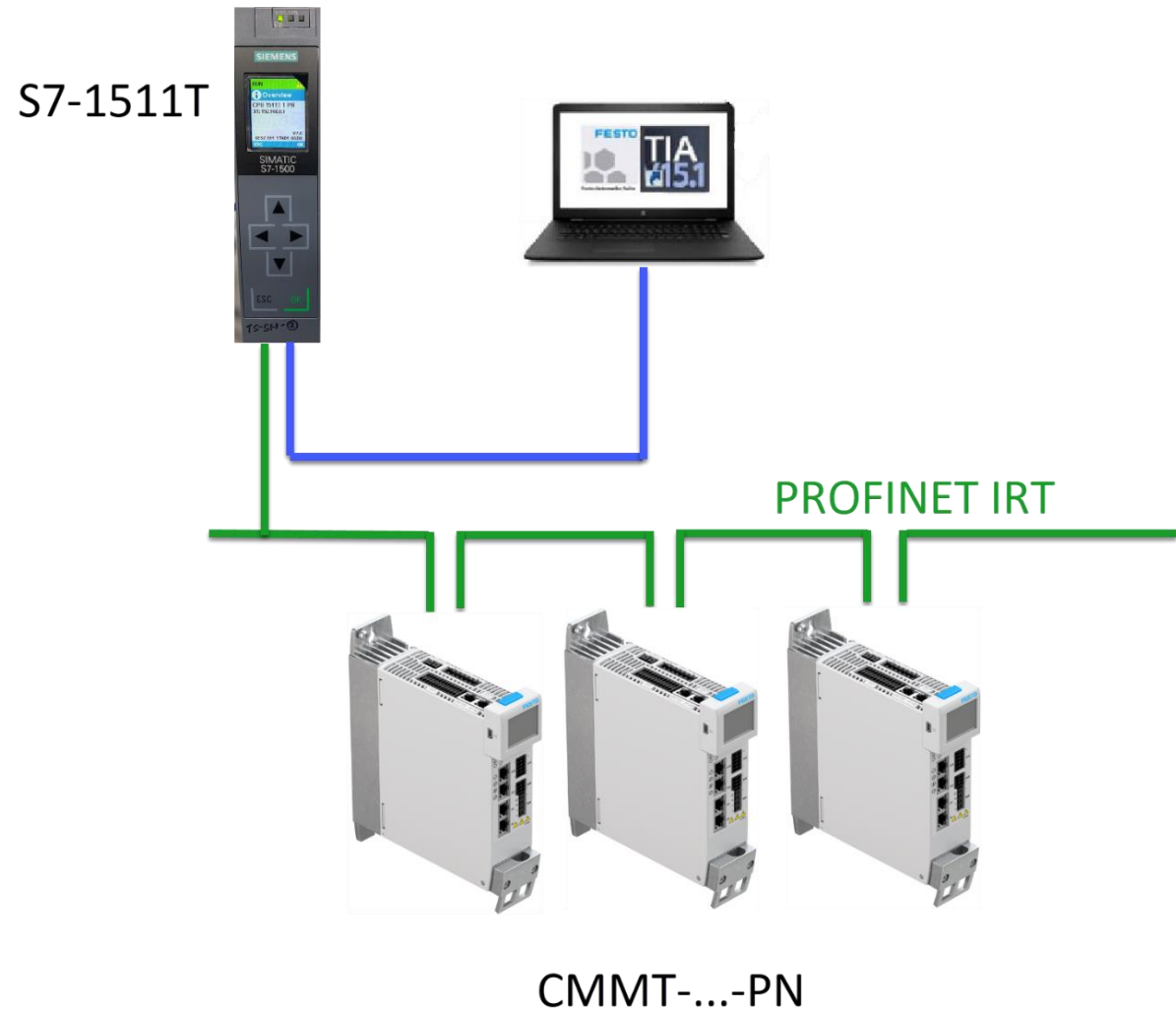
该功能块还能控制下发给工艺对象的指令数量和队列，保证它们有序运行并且不会溢出。因此，能最大化的进行路径动态规划。

4 软/硬件环境

软/硬件	版本
CMMT-AS-C4-3A-PN-S1	V17.0.8.48
S7-1511T-1PN	V2.6
Automation Suite	V1.3.2.4
CMMT-AS Plug-in	V1.3.1.9
TIA Portal	V15.1
CMMT_GSDML	GSDML-V2.34-Festo-CMMT-AS-20191201
LKinCtrl 库	109755891_LKinCtrl_LIB_V2_1_4_V15.1.zip***
LKinCtrl 原样例项目	109755891_LKinCtrl_PRJ_v2_1_4_V15.1.zip
LKinCtrl 修改样例项目	CMMT_LKinCtrl_G_Code_ExampleProject_V15.1.zap15_1
LKinCtrl 手册	109755891_LKinCtrl_DOC_v2_1_0_en.pdf

***建议用博途 V15.1 和 V16 版本，V15 版本下编译报错。

实验拓扑结构：



5 FAS 中的关键设置

三个轴都可参考以下设置：

5.1 报文设置

将报文设置为 105

X CMMT-AS-C4-3A-PN-S1
Path: 192.168.0.102
Connected

Disconnect

Plug-in PLC Control

Enabled Disabled Powerstage

Stop

Acknowledge all

Store on Load fact

Parameter pages < Fieldbus

- Drive configuration
- Device settings
- Fieldbus** 1
- Digital I/O
- Analogue I/O
- Encoder interface

Connection properties

PZD telegram selection	Telegram (105) ▼	■
Current application class	Application class 4 (4)	■

5.2 AC4 相关参数

下图参数（可以保持默认）需要和工艺对象中组态保持一致



X
CMMT-AS-C4-3A-PN-S1
Path: 192.168.0.102
Connected

Disconnect



Plug-in
PLC
Control



Enabled
Disabled
Powerstage



Stop



Acknowledge all



Store on device



Load factory settings



Reinitialize



Restart device

Parameter pages

<
Fieldbus

Drive configuration

Device settings

▶
Fieldbus
1

Digital I/O

Analogue I/O

Encoder interface

▼
Axis 1
12

Motor

Gearbox

▶
Axis
10

Record list

Monitoring functions

▶
Closed loop
2

Position

Velocity

Acceleration

Jerk

Reference values

Base value velocity (user unit)

0.50

m/s

■

Base value speed (controller)

3000.00

rpm

■

Base value acceleration

0.10

m/s²

■

Base value deceleration

0.10

m/s²

■

Parameter pages

<
Fieldbus

Drive configuration

Device settings

▶
Fieldbus
1

Digital I/O

Analogue I/O

Encoder interface

▼
Axis 1
12

Motor

Gearbox

▶
Axis
10

Record list

Monitoring functions

▶
Closed loop
2

Auto tuning

Vibration compensation

Feed forward control

▶
Cam controller

AC4

Maximum

10000.00

rpm

■

Maximum motor or servo drive torque

3.235

Nm

■

Bus cycle time (KPC)

0.00

s

■

Amplification factor Position control (without DSC)

15.10752

■

Amplification factor Position control (with DSC)

15.10752

■

Resolution per revolution for Gn_XIST

Resolution per revol

■

Number of revolution

16384

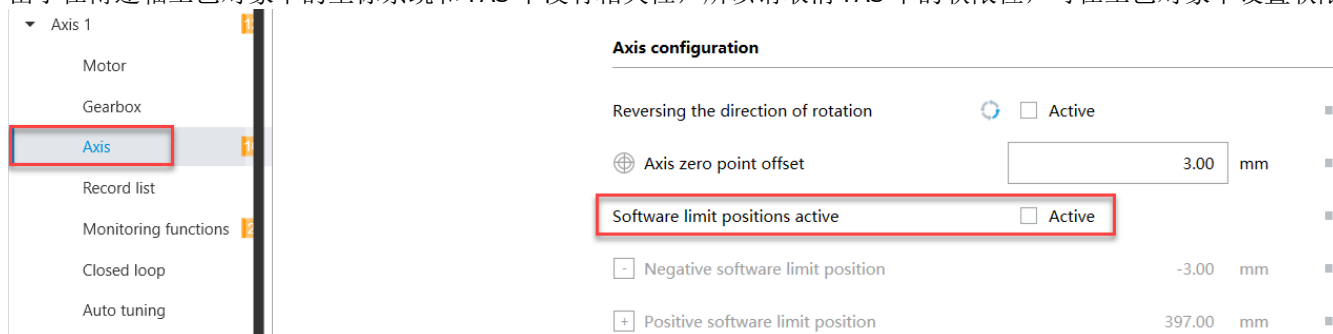
■

Bits in Gn_XIST1

0

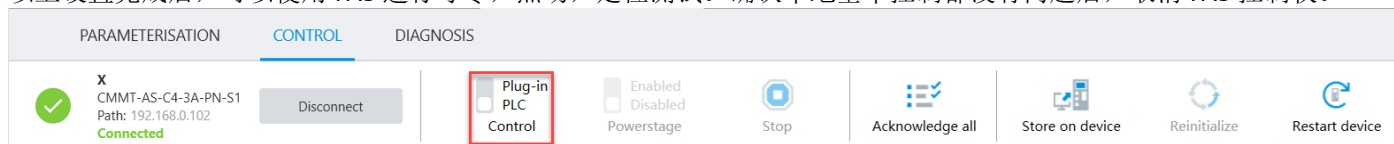
5.3 软限位设置

由于在博途轴工艺对象中的坐标系统和 FAS 中没有相关性，所以请取消 FAS 中的软限位，可在工艺对象中设置软限位。



5.4 FAS 基本测试

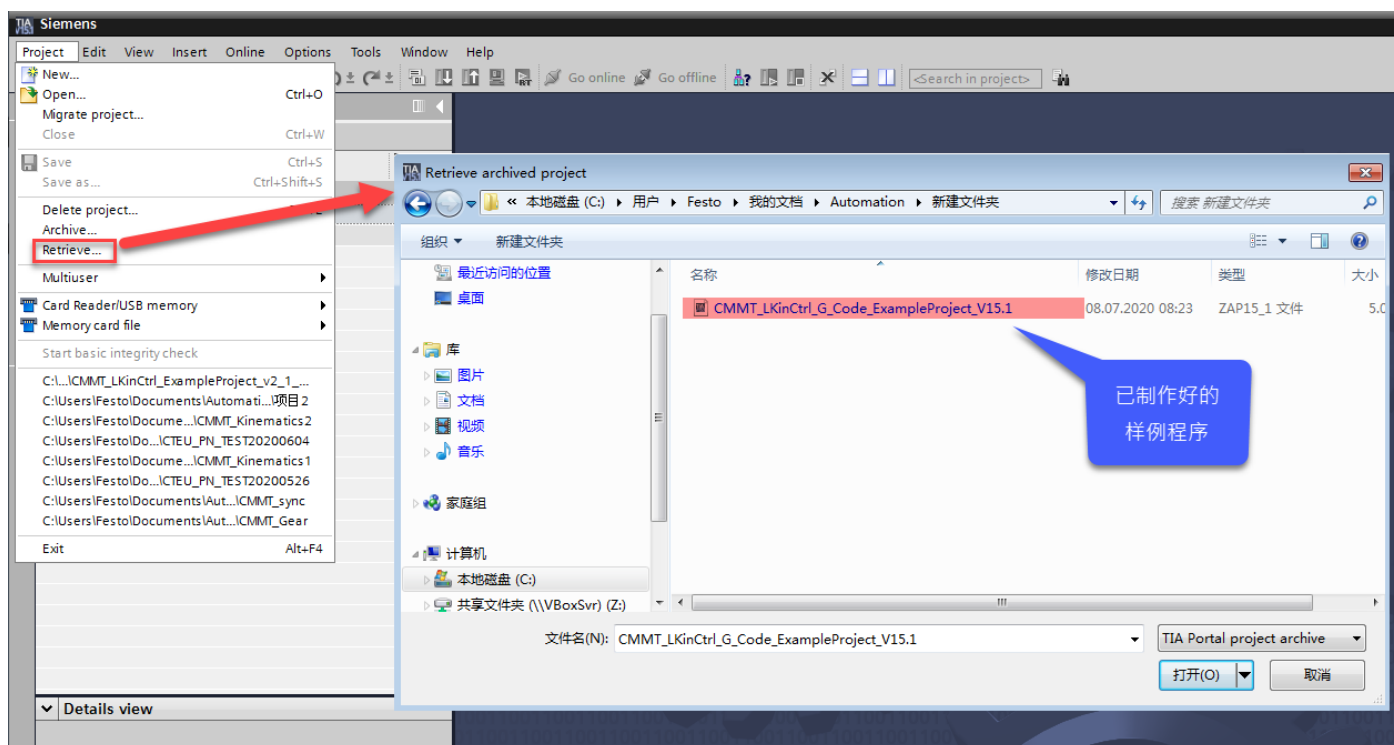
以上设置完成后，可以使用 FAS 进行寻零，点动，定位测试。确认本地基本控制都没有问题后，取消 FAS 控制权。



6 TIA Portal 中的配置

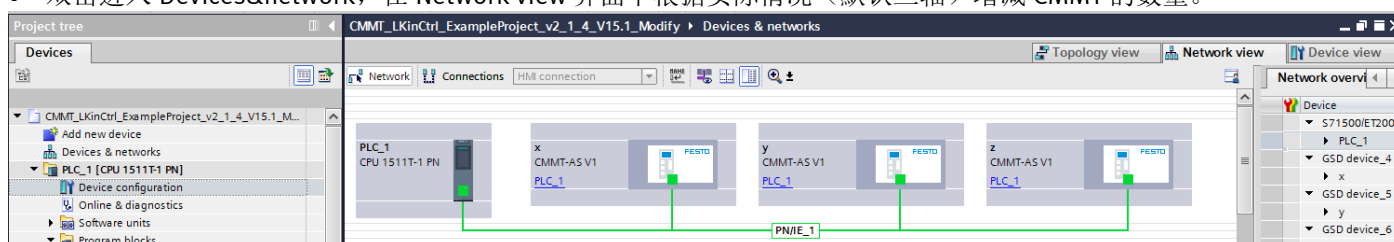
为了方便大家调试，针对于 CMMT，我们在西门子样例程序的基础上已对其进行了修改并归档，可向 Festo 技术支持获取该样例程序。

6.1 释放样例程序

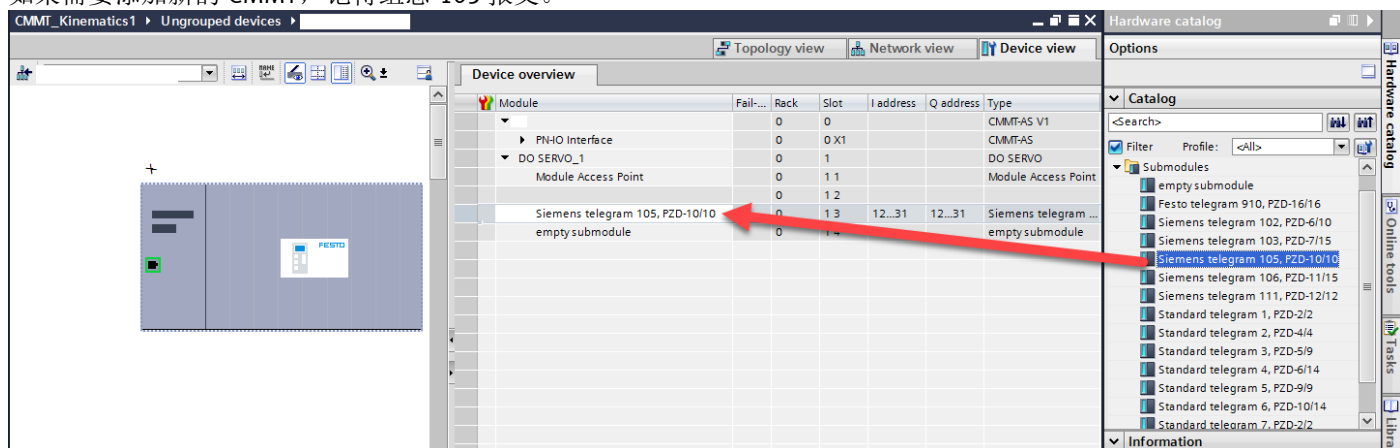


6.2 修改硬件组态

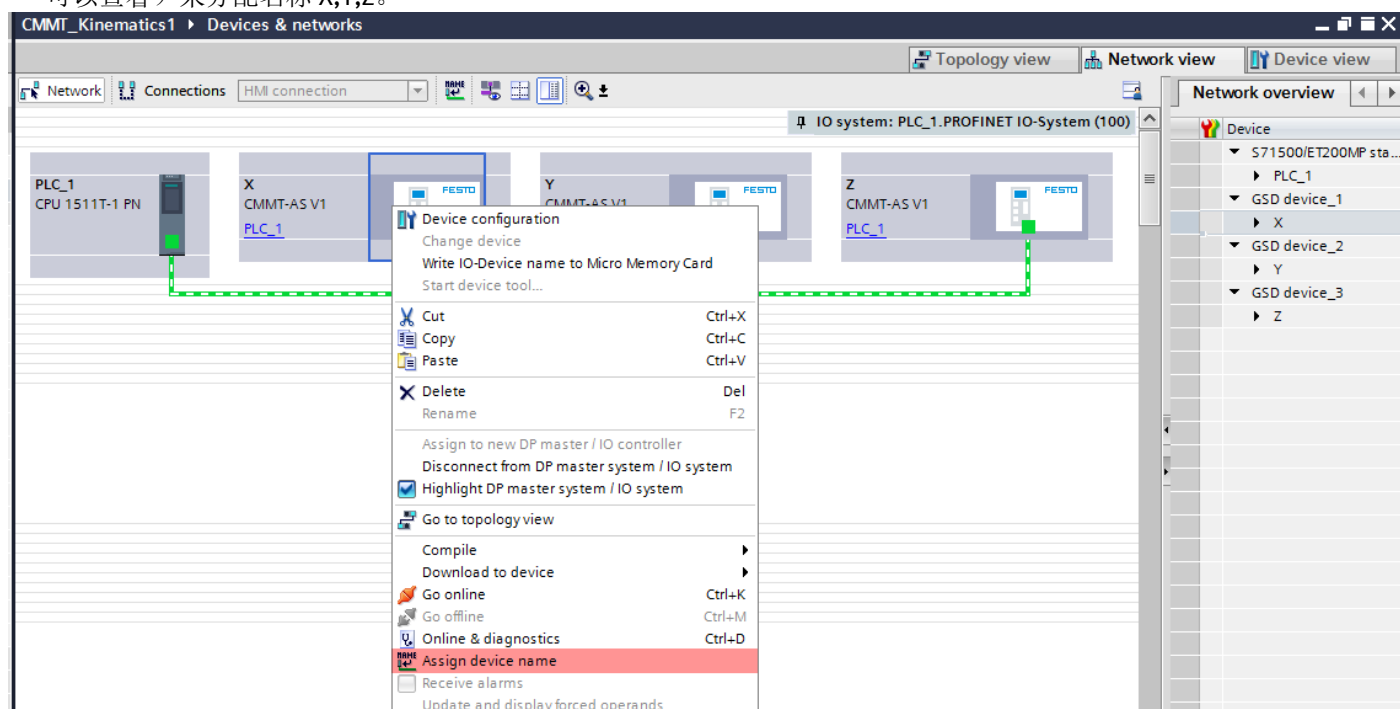
- 双击进入 Devices&network，在 Network view 界面下根据实际情况（默认三轴）增减 CMMT 的数量。



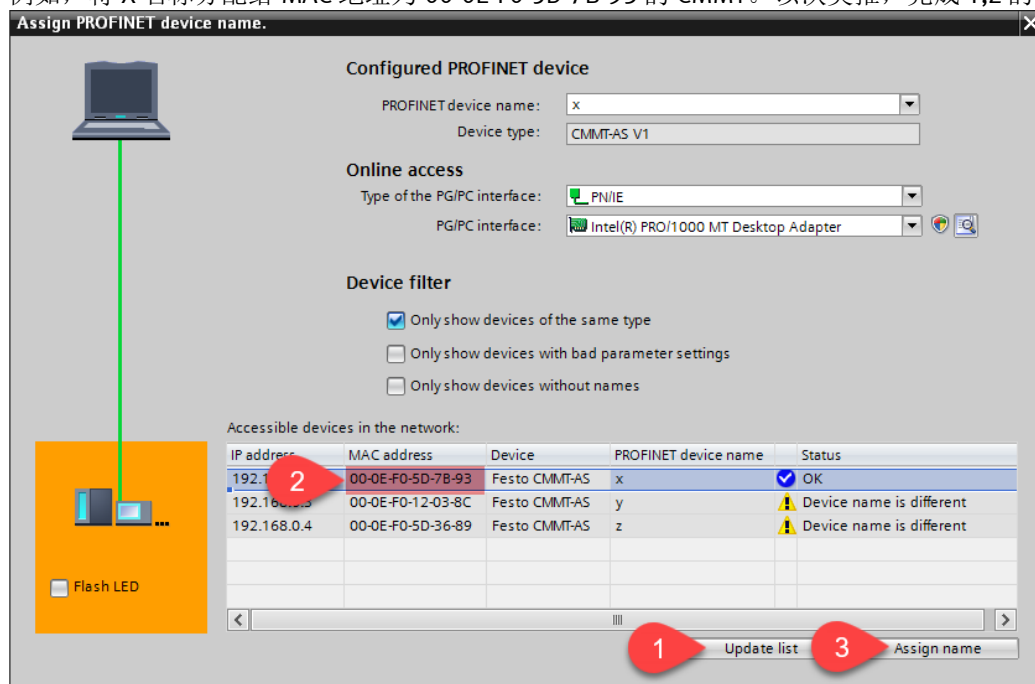
如果需要添加新的 CMMT，记得组态 105 报文。



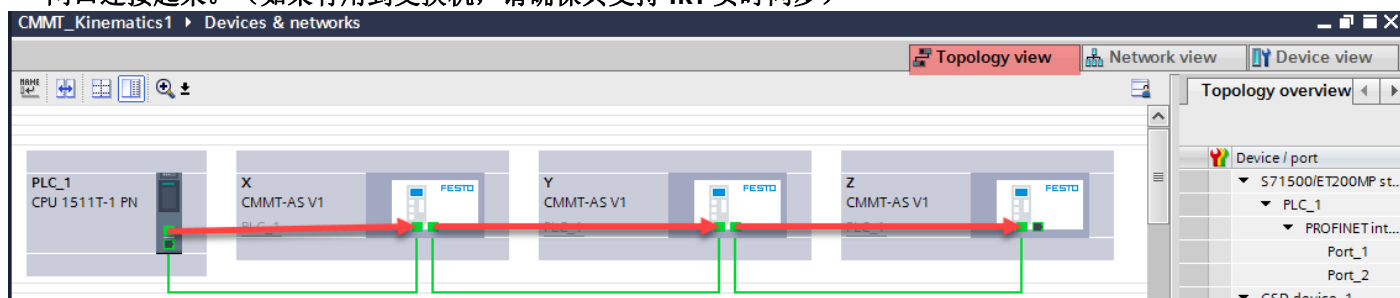
- 在 Network view 界面下右击 CMMT-AS，选择 Assign Device Name，根据对应 CMMT 的 MAC 地址（CMMT 实物标签上可以查看）来分配名称 X,Y,Z。



例如，将 X 名称分配给 MAC 地址为 00-0E-F0-5D-7B-93 的 CMMT。以次类推，完成 Y,Z 的名称分配。



- 再进入 **Topology View** 界面，根据网口实际的物理连接来修改拓扑连接，将 **PLC** 的网口和 3 台 **CMMT** 的 **PROFINET** 网口连接起来。（如果有用到交换机，请确保其支持 **IRT** 实时同步）



- 设置等时同步（程序中已设置好，可以修改）

闭环运动控制必须有很好的实时性，所以必须设置等时同步。

在 **Network view** 中用鼠标双击连接线“**PN/IE_1**”在下边的“属性”、“常规”页面内选择“同步域”、“**Sync-Domain_1**”，设置“**PLC_1.PROFINET** 接口_1”的“同步角色”为“同步主站”，设置 **X,Y,Z** 轴的“**RT 等级**”为“**IRT**”，“同步角色”为“同步从站”。

其中“发送时钟”越小，采样周期越小，系统动态特性越好，但带来的负面影响是系统通讯负荷增大。这里选择 **1ms**。

Sync-Domain_1

Sync domain: Sync-Domain_1
 Converted name: sync-domainxb19998
 Send clock: 1.000 ms

☒ Default domain
☐ Make 'high performance' possible
☐ Allows the use of 'fast forwarding'

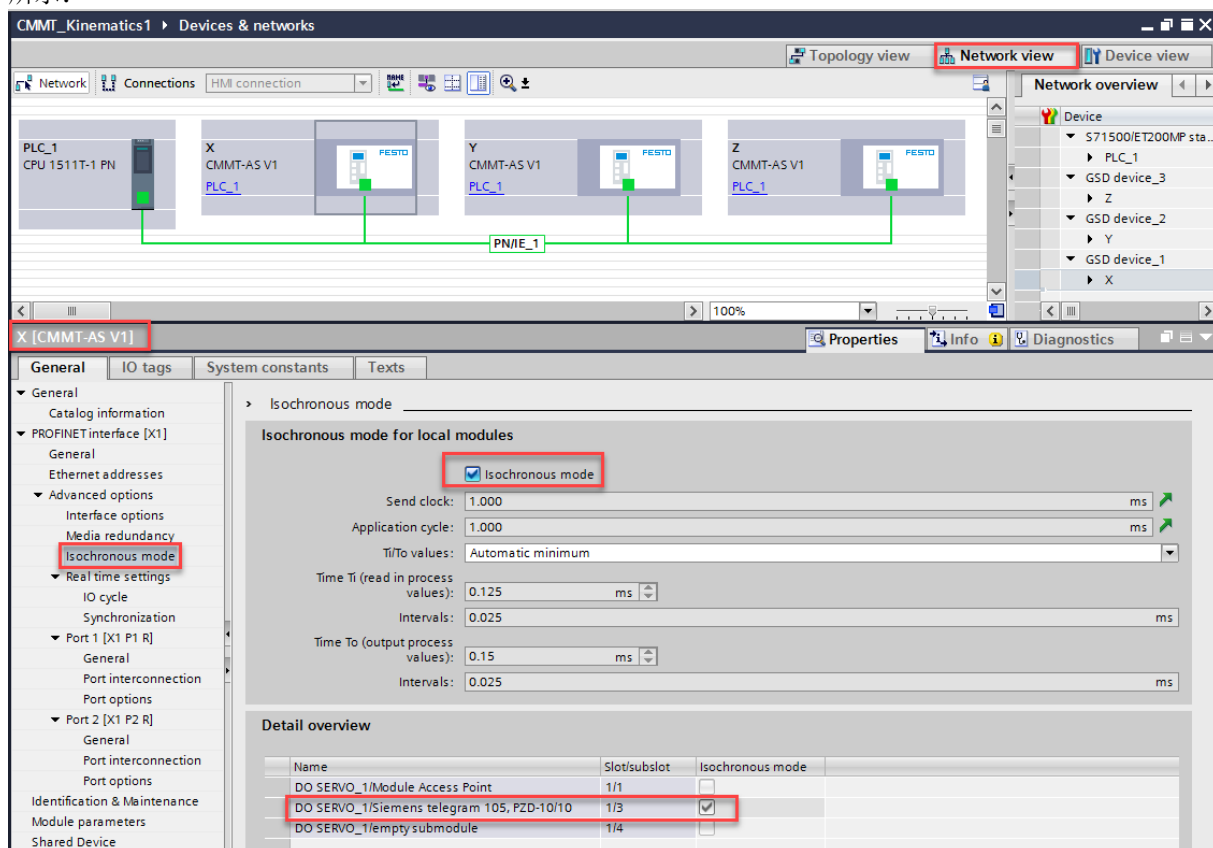
IO system

IO system	Sync master
PLC_1.PROFINET IO-System (100)	PLC_1

IO devices

PRO..	RT class	Synchronization role	Redundancy level	DFP group
plc_1	IRT...	Sync master		
x	IRT	Sync slave	No redundancy	1
y	IRT	Sync slave	No redundancy	1
z	IRT	Sync slave	No redundancy	1

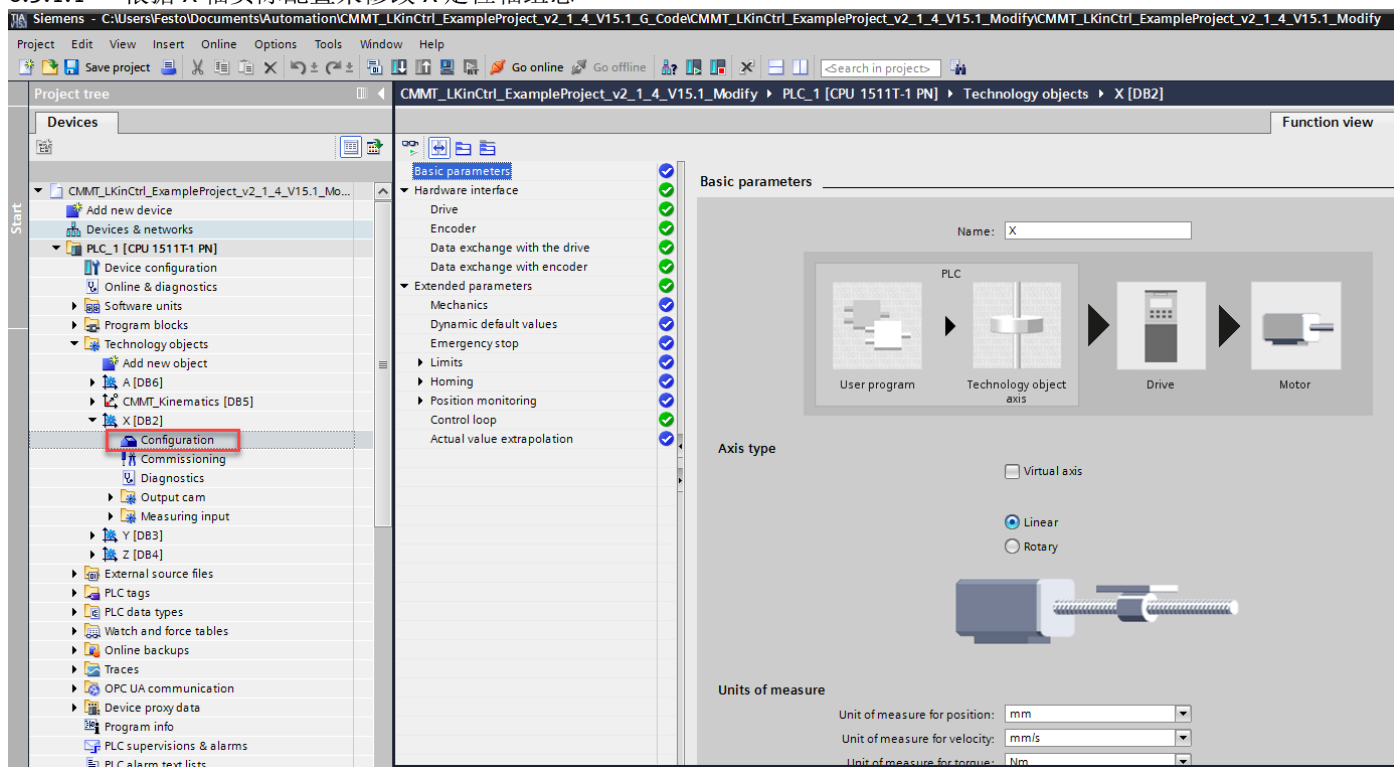
分别在 X,Y,Z 轴的设备视图选择“属性”、“常规”、“等时同步模式”页面，然后勾选“等时同步模式”，如下图所示：



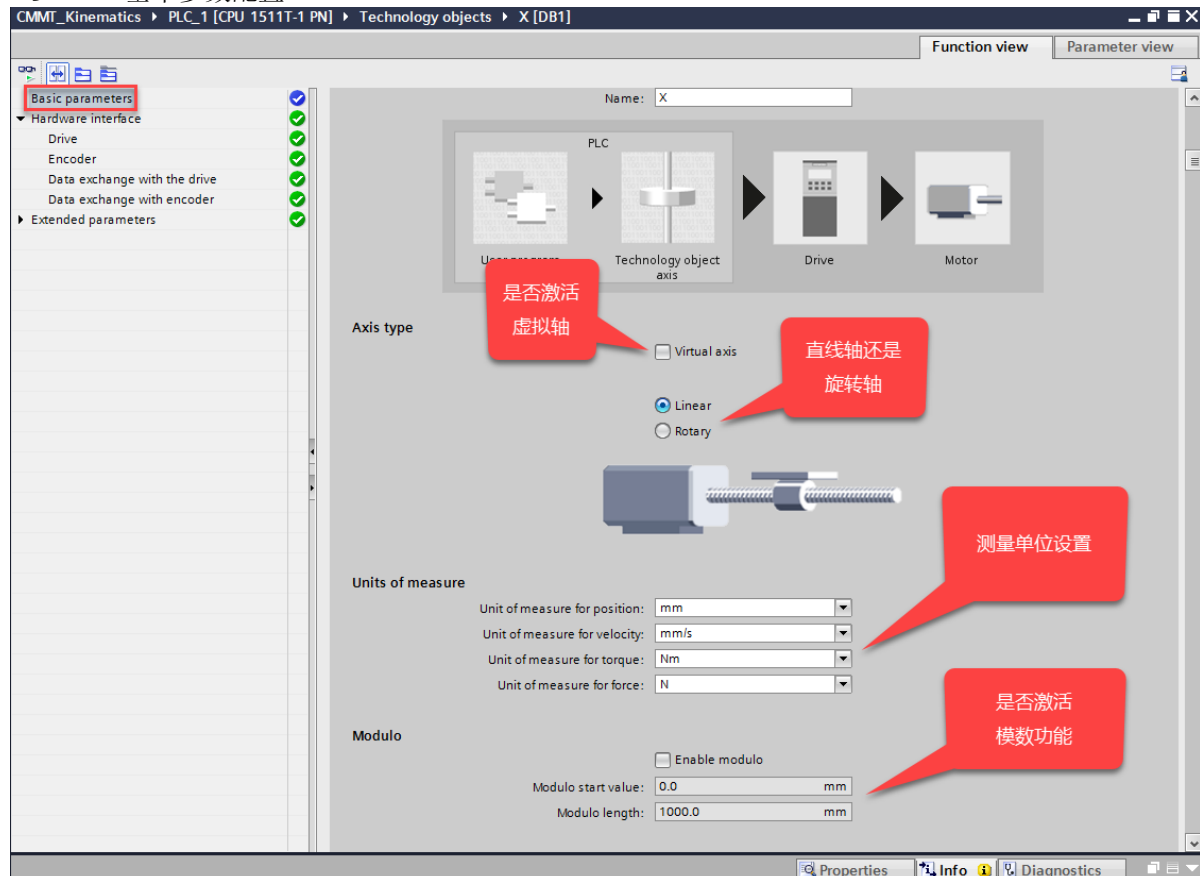
6.3 修改组态工艺对象

6.3.1 修改 X,Y,Z 定位轴工艺对象组态

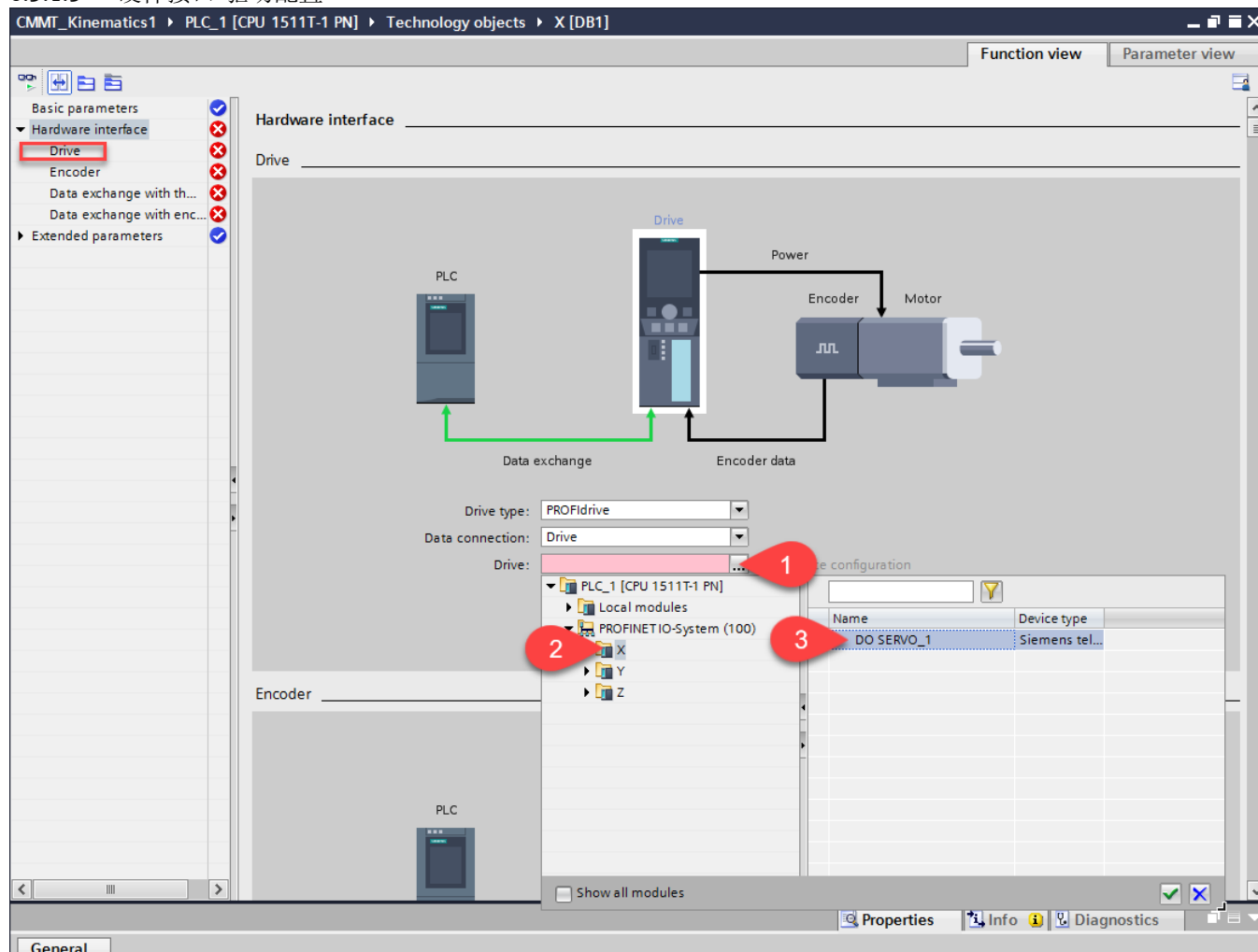
6.3.1.1 根据 X 轴实际配置来修改 X 定位轴组态



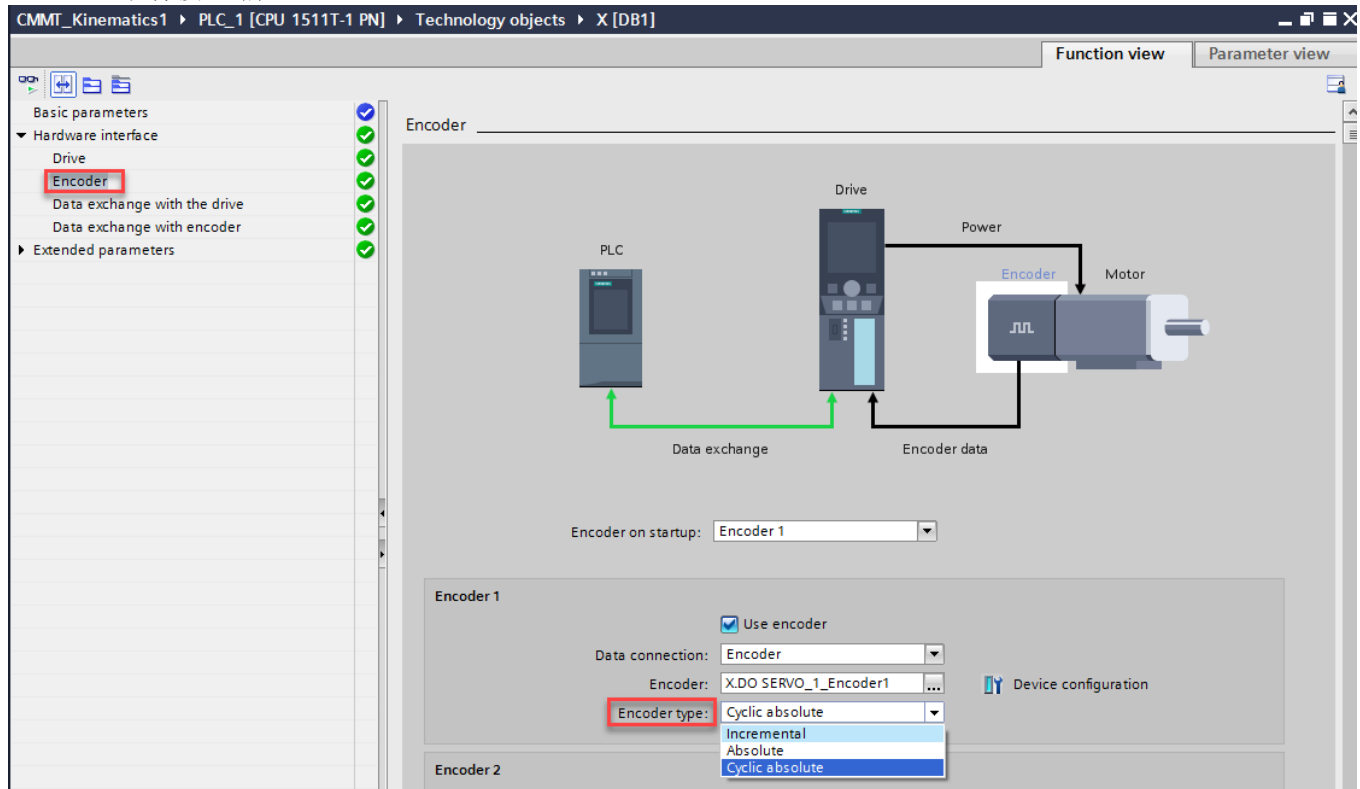
6.3.1.2 基本参数配置



6.3.1.3 硬件接口-驱动配置

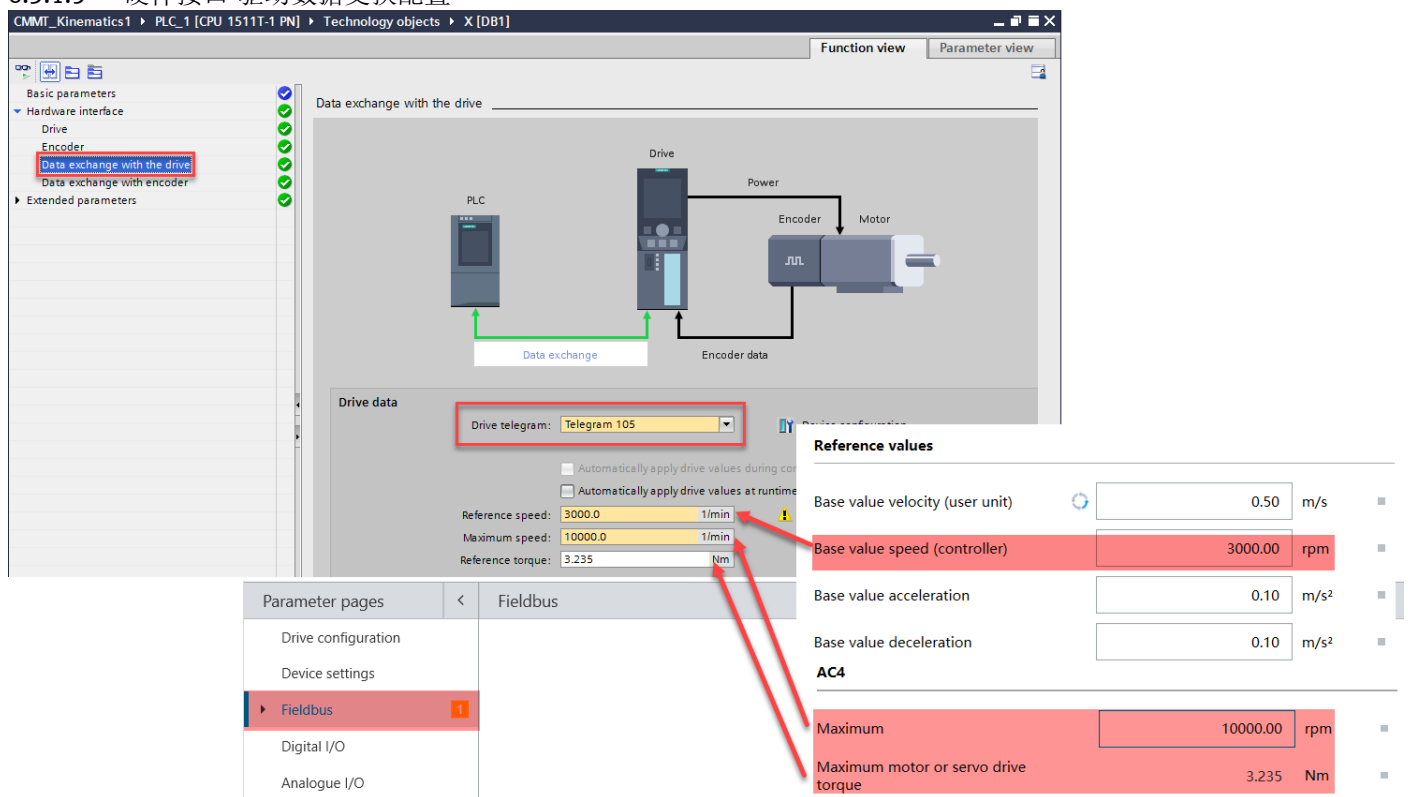


6.3.1.4 硬件接口-编码器配置



- Incremental: 增量型编码器
- Absolute: 绝对值单圈编码器 (例如: EMMT/S/E/B-AS- x-x-xx-xSx)
- Cyclic Absolute: 绝对值多圈编码器 (例如: EMMT/S/E/B-AS- x-x-xx-xMx)

6.3.1.5 硬件接口-驱动数据交换配置



6.3.1.6 硬件接口-编码器数据交换配置

勾选后，当软件在线时，CMMT 自动上传编码器数据到工艺对象

Encoder telegram: Telegram 105

☒ Automatic data exchange for encoder values (online)

6.3.1.7 扩展参数-机械设置

编码器安装位置

正反转设置

减速机比例设置

电缸的进给常量/螺距

Feed constant: 10.00 mm/r

Encoder mounting type: On motor shaft

Invert encoder direction

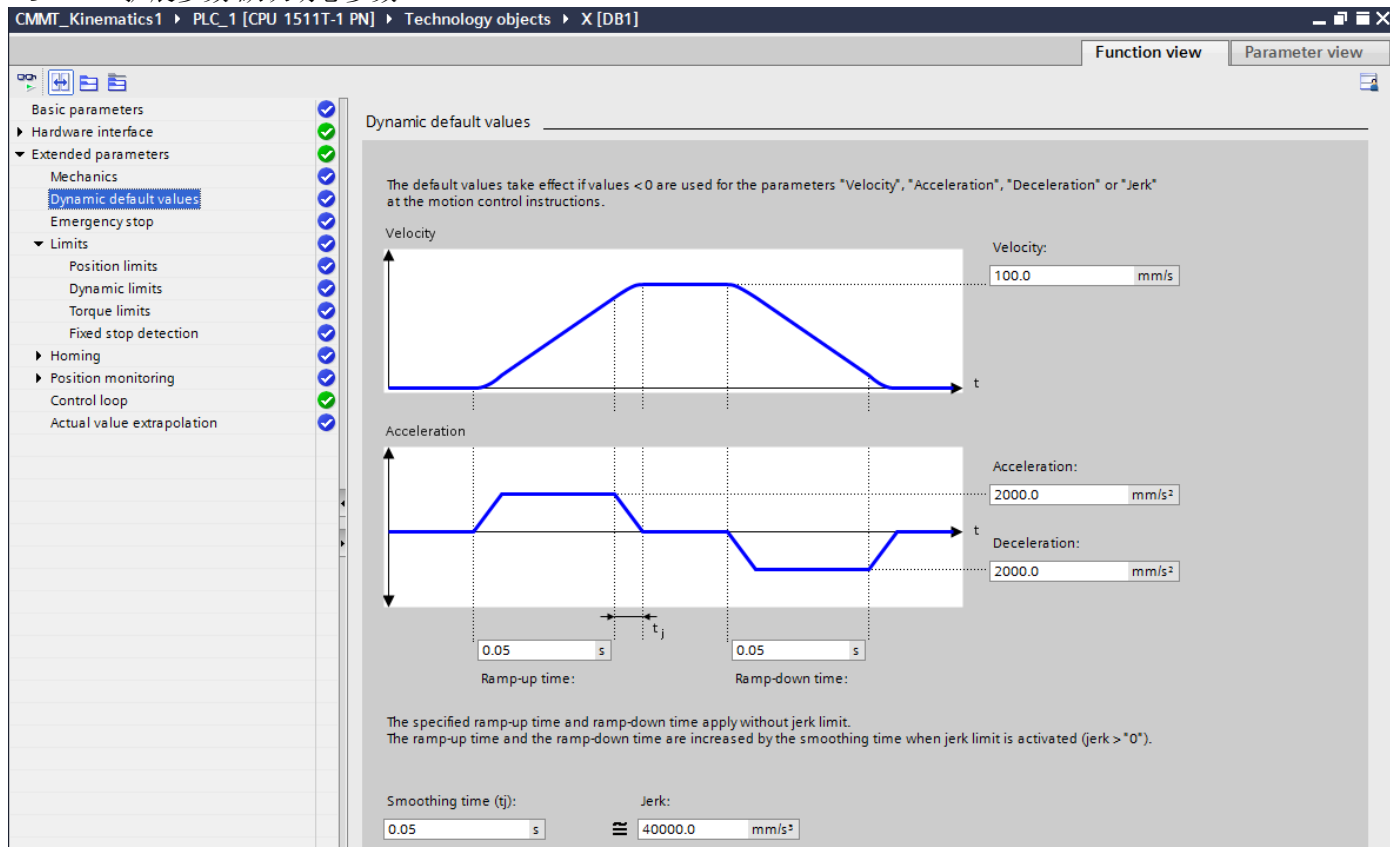
Invert drive direction

Number of motor revolutions: 1

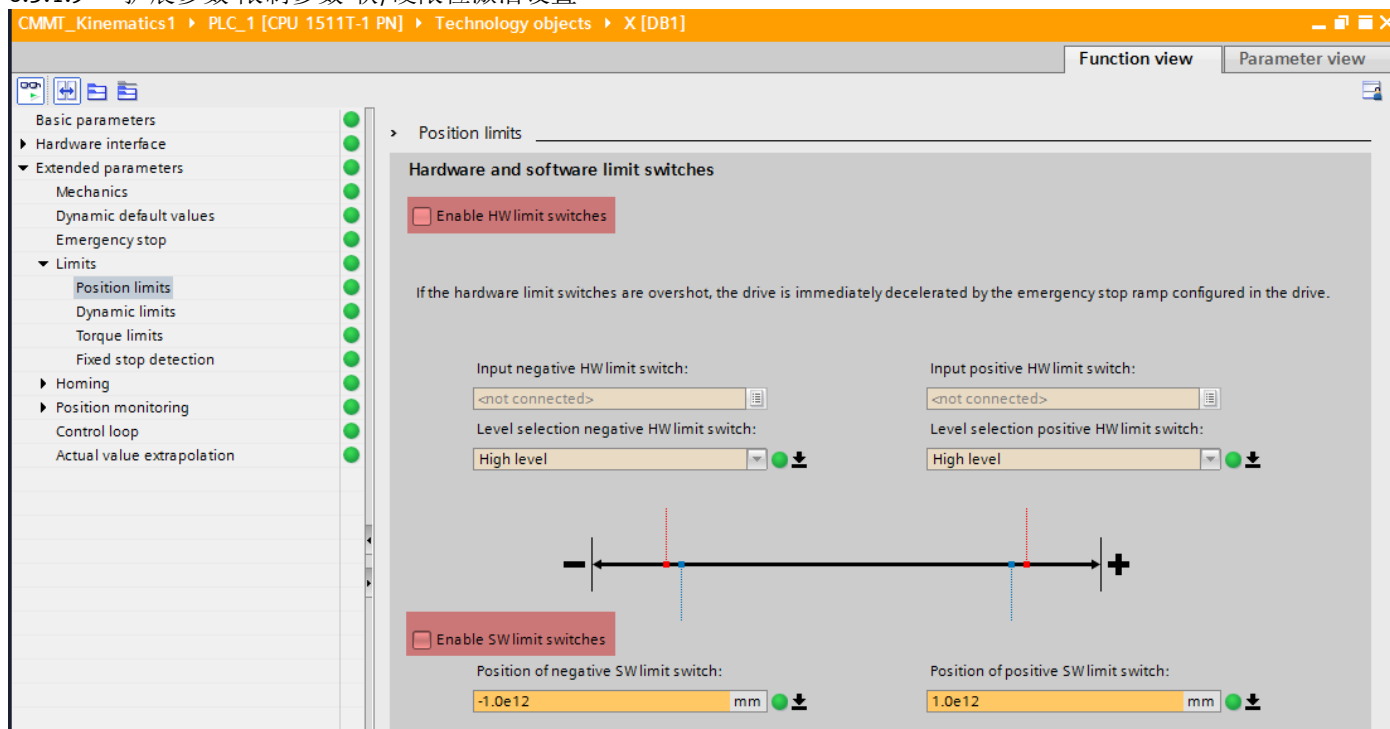
Number of load revolutions: 1

Leadscrew pitch: 10.0 mm/rot

6.3.1.8 扩展参数-默认动态参数



6.3.1.9 扩展参数-限制参数-软/硬限位激活设置



6.3.1.10 其他设置

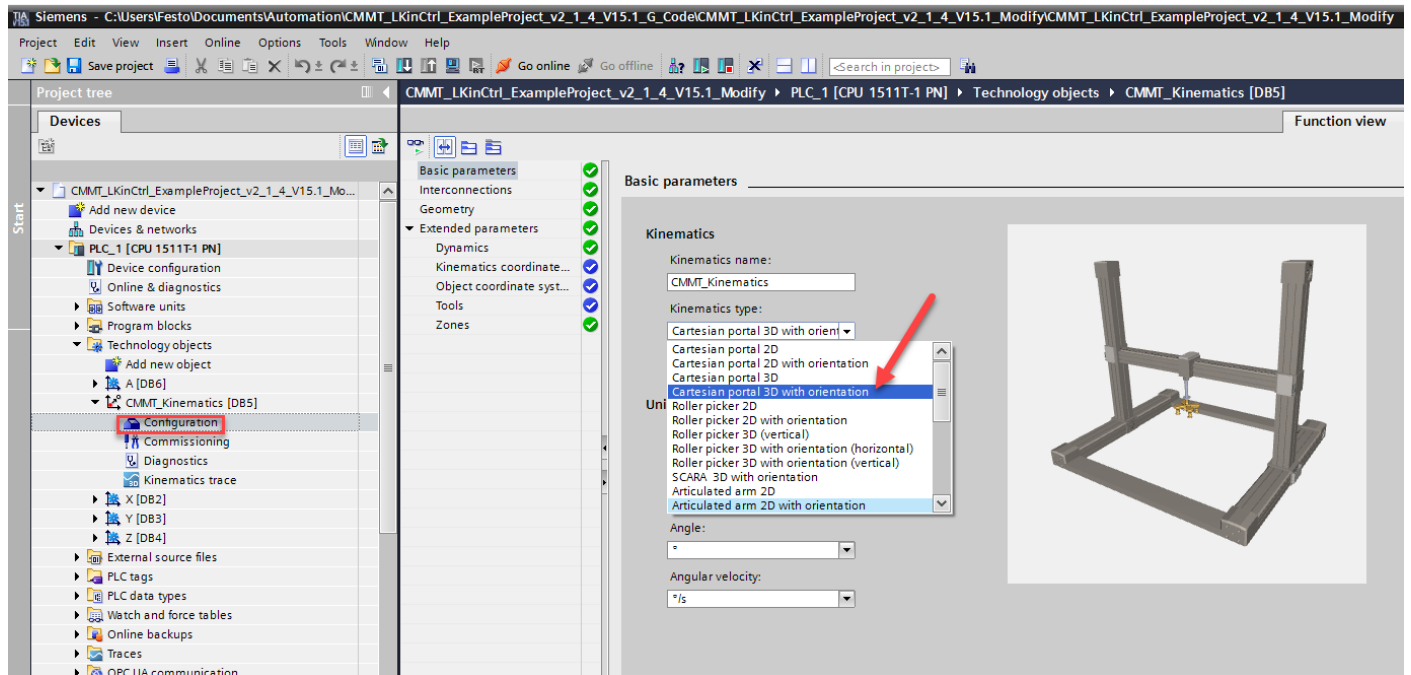
关于扩展参数里的其他参数，例如：急停参数，寻零参数（绝对型编码器不支持主动寻零，无需设置），位置监控参数等，建议客户根据实际的应用来进行设置，这里就不一一展开说明了。

6.3.1.11 以上 X 轴的工艺对象组态完成。Y 和 Z 轴根据实际情况，参考上面步骤也完成配置。

6.3.2 修改运动学工艺对象组态

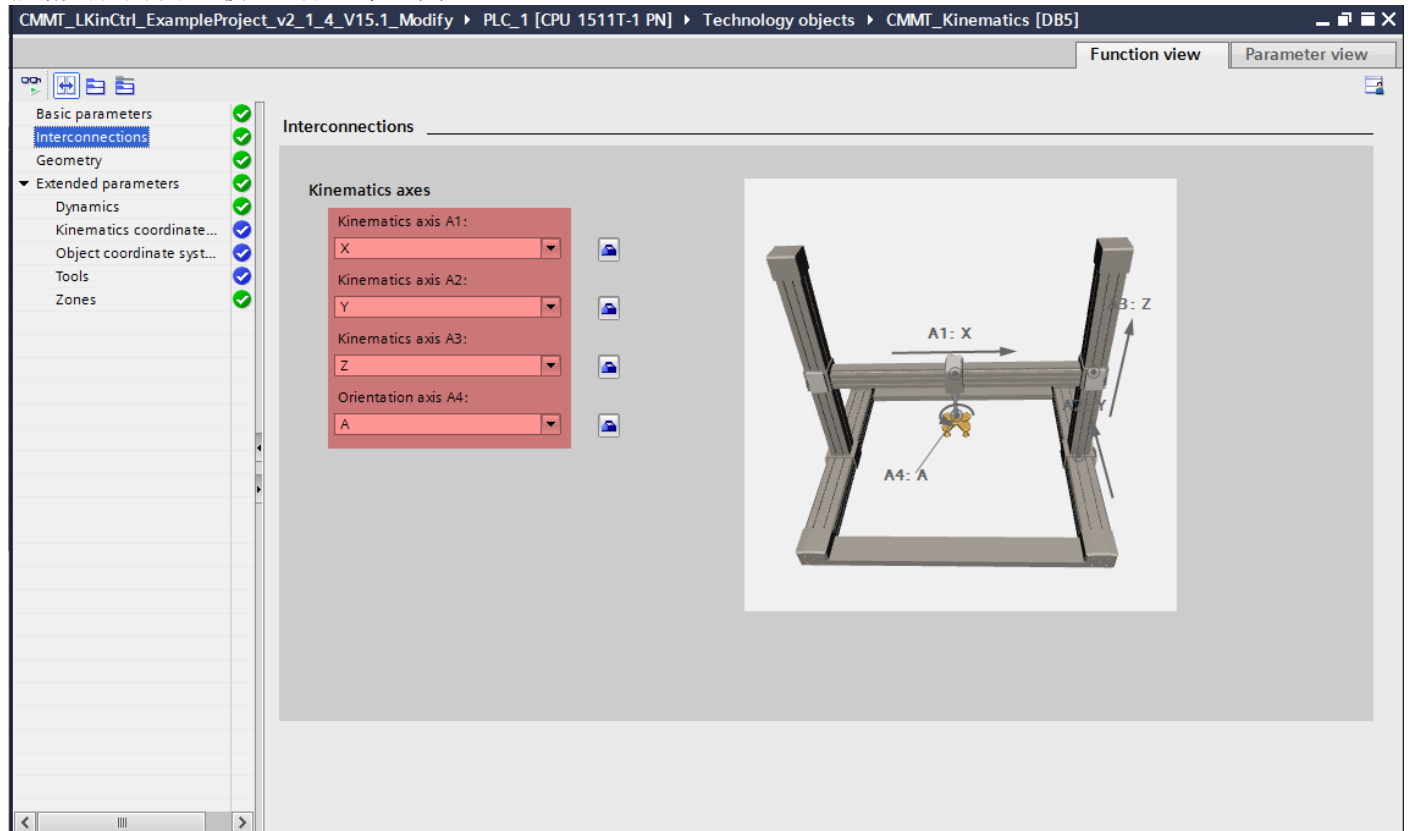
6.3.2.1 基本参数

根据运动机构的类型选择相应的运动模型，本次测试用的是直角坐标 3D 系统带旋转（旋转为虚轴 A）。



6.3.2.2 互联

根据实际结构结合模型，分配每一个轴。



6.3.2.3 几何参数

CMMT_Kinematics1 ▸ PLC_1 [CPU 1511T-1 PN] ▸ Technology objects ▸ CMMT_Kinematics [DB4]

Function view | Parameter view

Basic parameters
Interconnections
Geometry
Extended parameters
Dynamics
Kinematics coordinate...
Object coordinate syst...
Tools
Zones

Geometry

Transformation parameters

Length L1: 0.0 mm
Length L2: 0.0 mm
Length L3: 0.0 mm
Flange length LF: 0.0 mm

Representation in kinematics trace

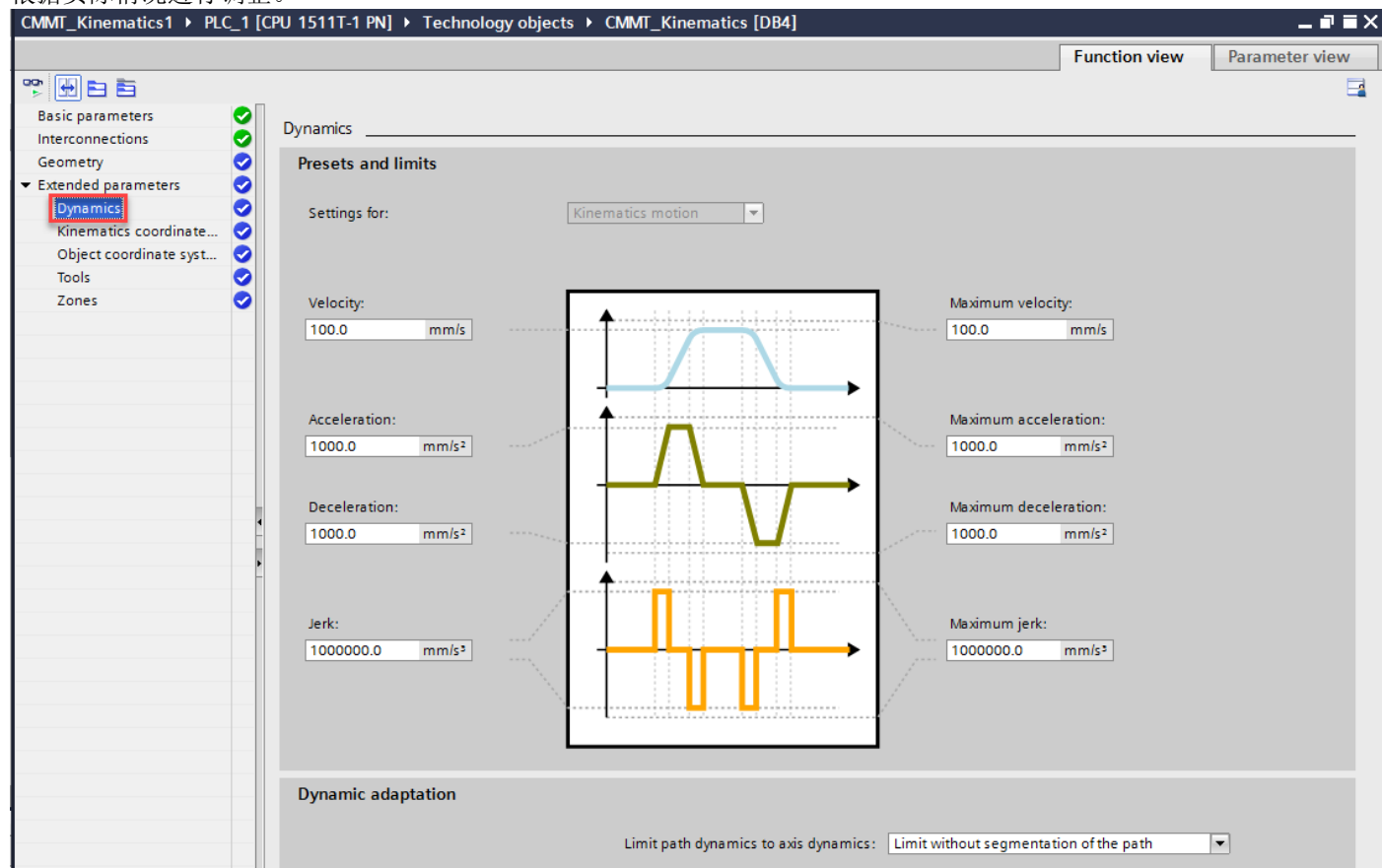
x minimum: 0.0 mm
x maximum: 400.0 mm
y minimum: 0.0 mm
y maximum: 300.0 mm
z minimum: 0.0 mm
z maximum: 100.0 mm

定义每个独立驱动轴的零位
相对于运动机构零位的距离

定义运动系统轨迹中显示
的运动机构的坐标范围

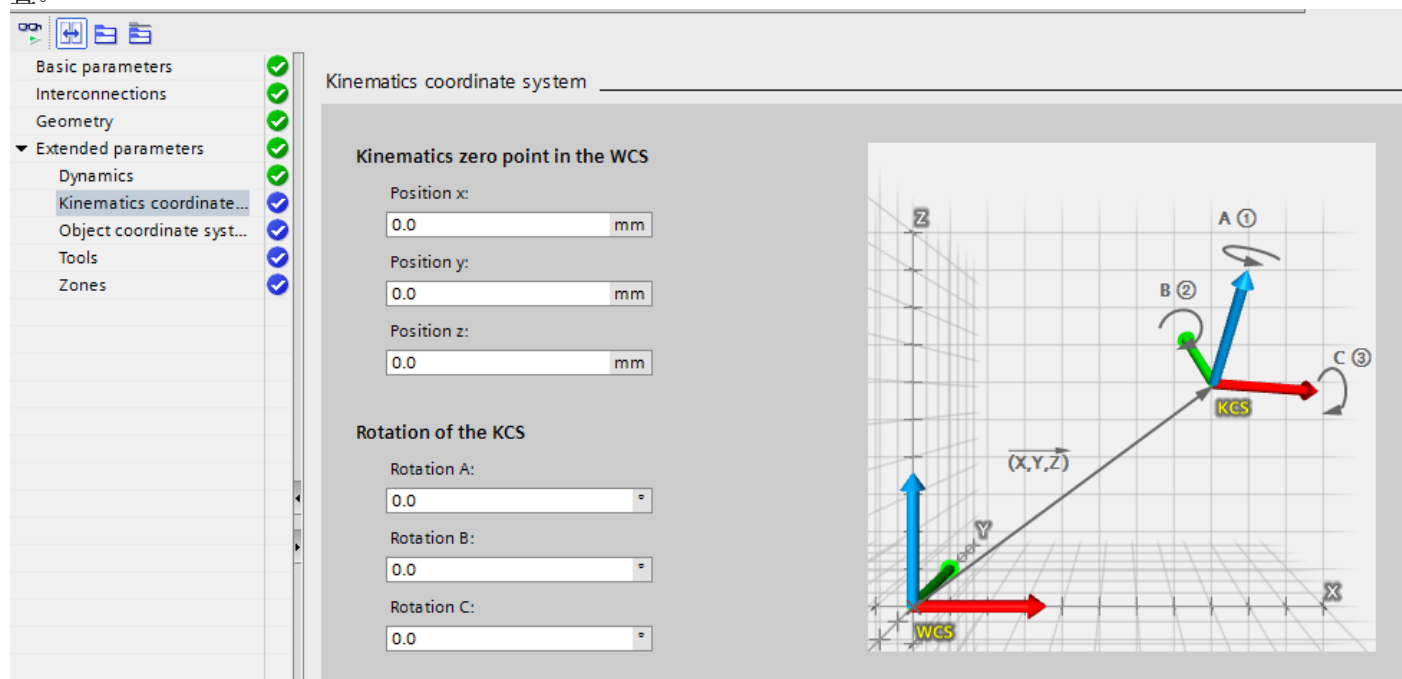
6.3.2.4 扩展参数-默认动态参数

根据实际情况进行调整。



6.3.2.5 扩展参数-运动系统坐标系

在“运动系统坐标系” (Kinematics coordinate system) 组态窗口中组态运动系统坐标系 (KCS) 在世界坐标系 (WCS) 中的位置。



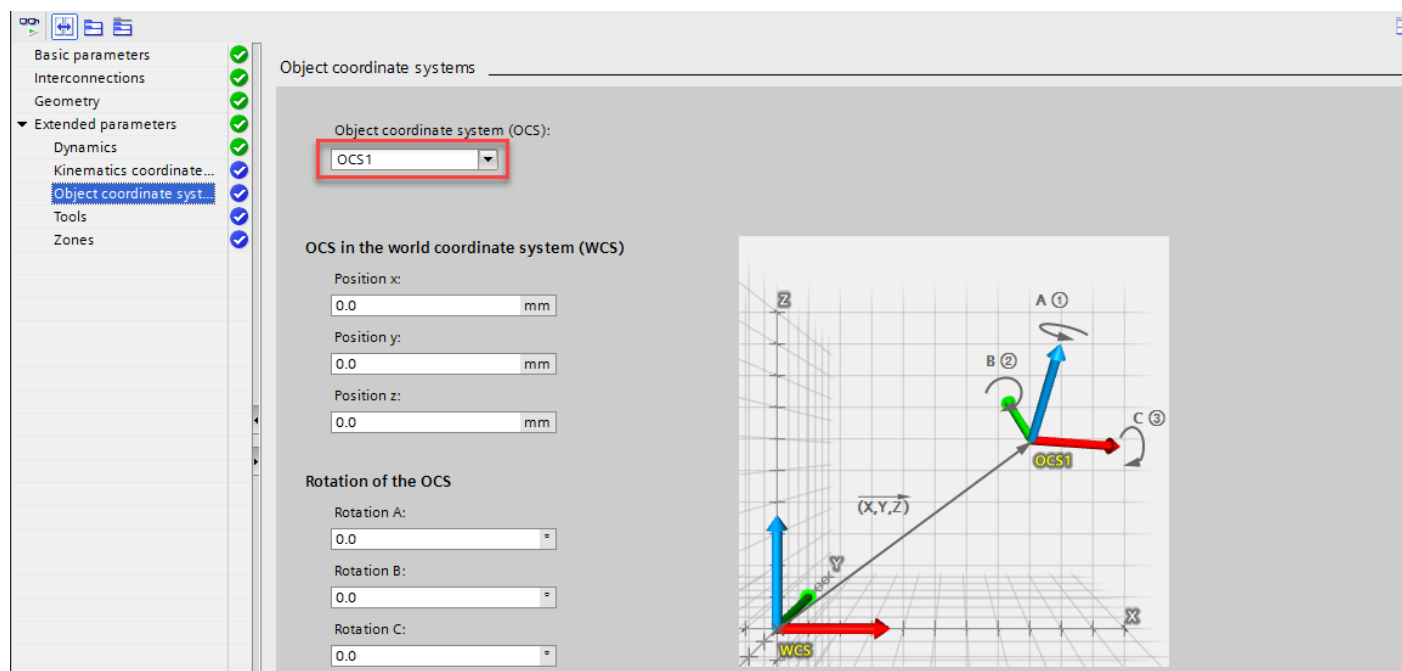
6.3.2.6 扩展参数-对象坐标系

在“对象坐标系”(Object coordinate system) 组态窗口中组态对象坐标系 (OCS) 在世界坐标系 (WCS) 中的位置。

可以预置三个 OCS，用于对于三个不同的对象。

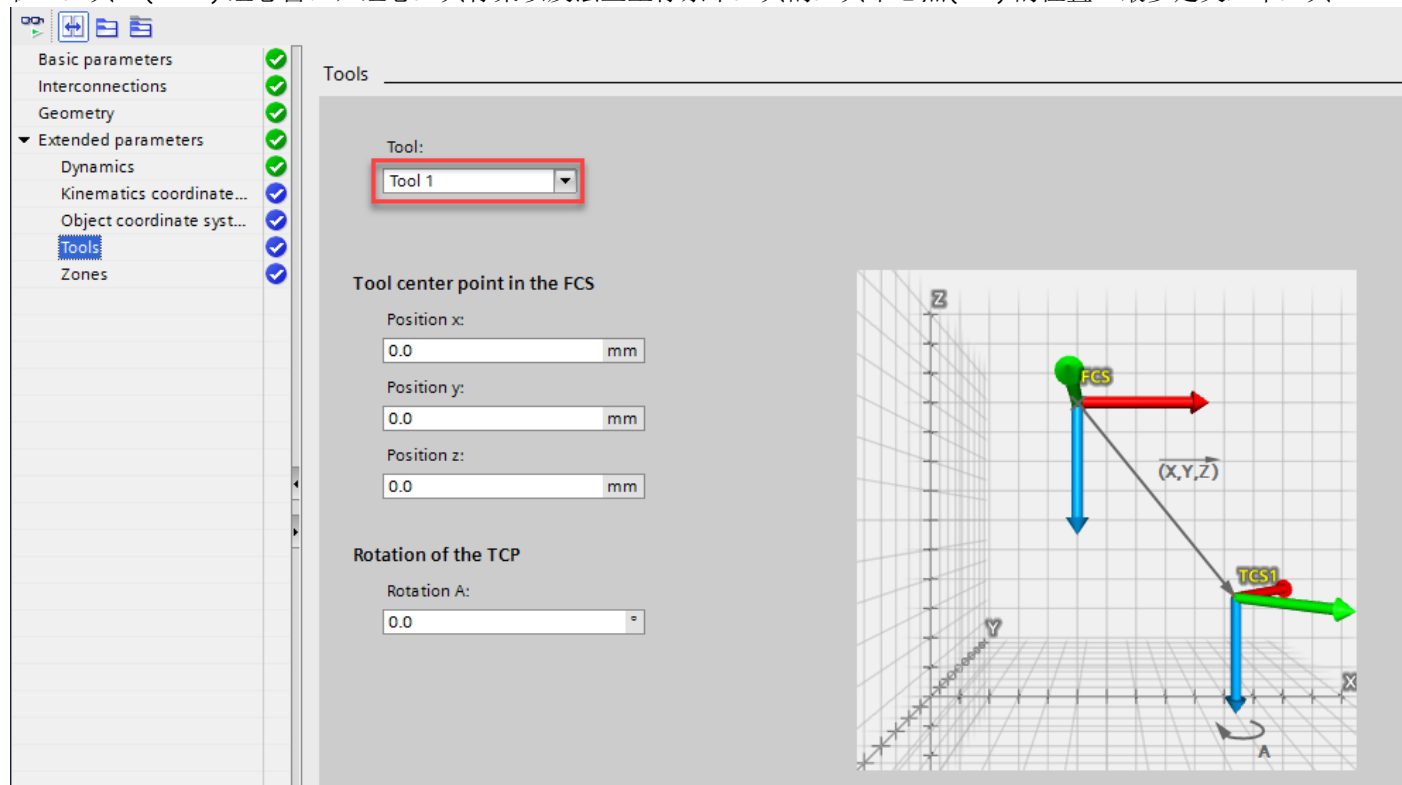
当现场对象（工件）坐标系和 WCS 坐标系出现偏移时，也可以用来进行修正。

也可通过指令 MC_SetOcsFrame：灵活地重新定义对象坐标系。详见后续 9.5 章节。



6.3.2.7 扩展参数-工具

在“工具”(Tools) 组态窗口，组态工具标架以及法兰坐标系中工具的工具中心点(TCP) 的位置。最多定义三个工具。

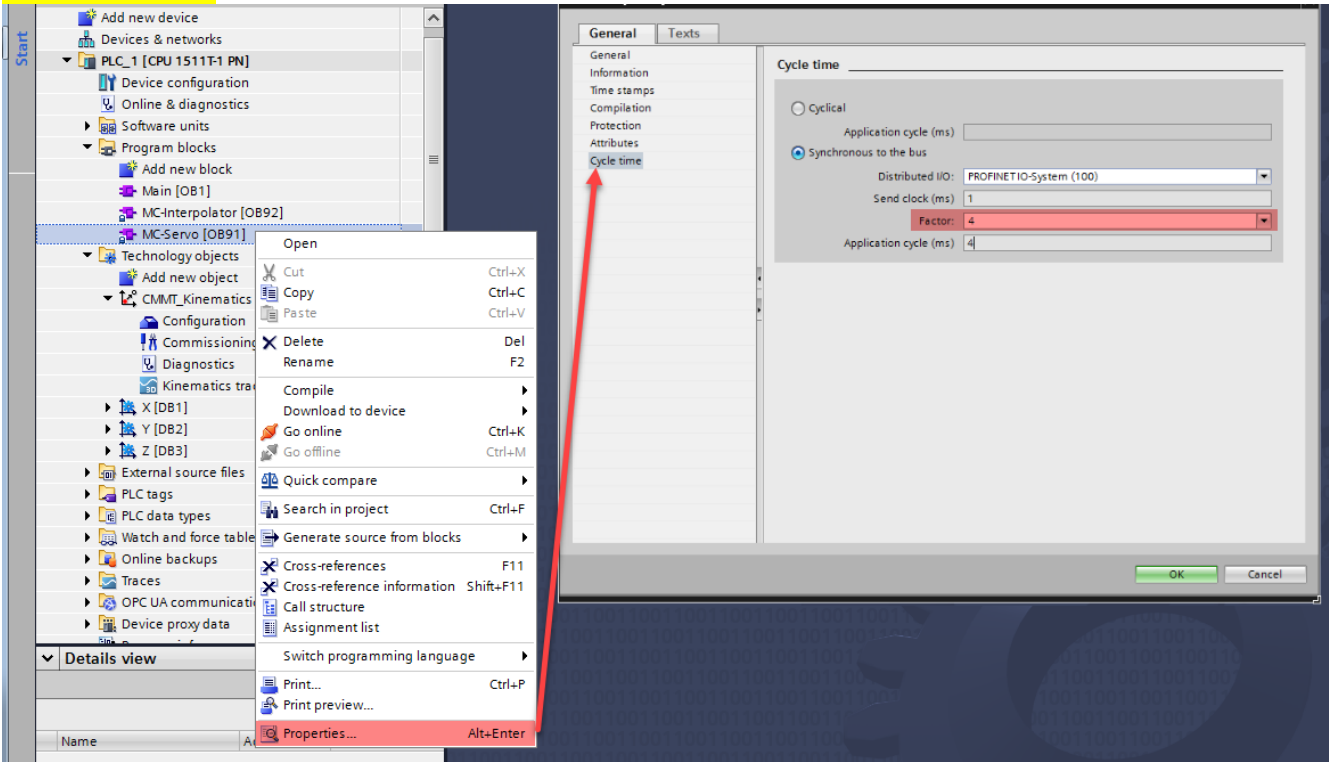


6.3.2.8 扩展参数-区域

在“区域”(Zones) 组态窗口中组态工艺对象的工作空间区域和运动系统区域。这里不做详细介绍，详见西门子 S7-1500T 运动系统功能手册。

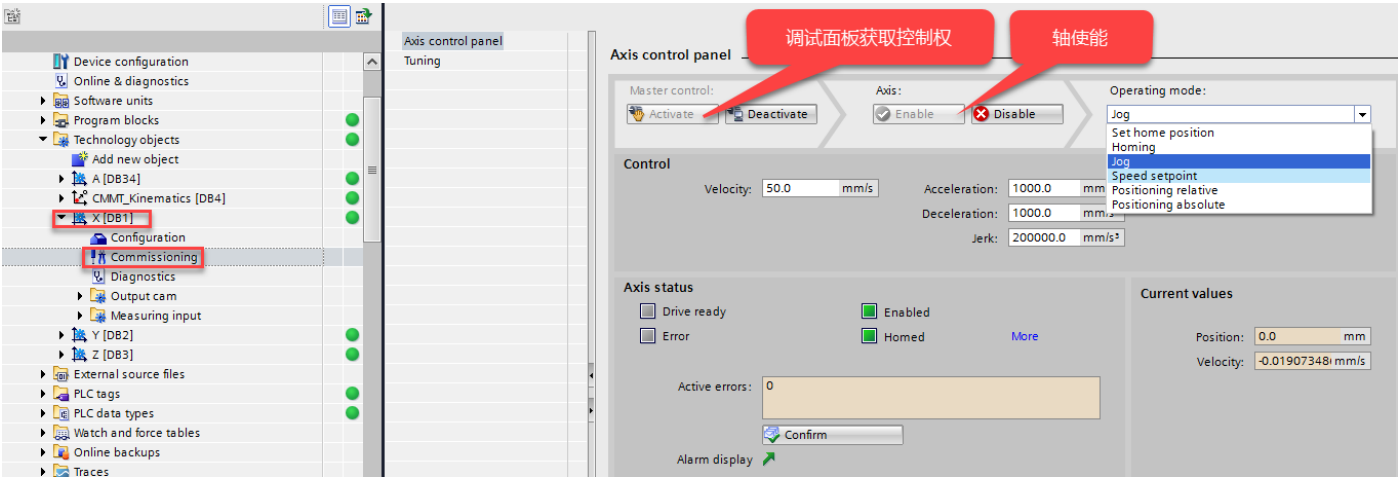
6.4 设置 OB91 的循环时间

OB91 是 S7-1500T 运动控制的核心组织块，为了更好的实现运动控制功能，需要把 OB91 的循环时间同步到总线。另外可根据应用情况选择“因子”，增加“因子”可减轻 CPU 负荷，但会影响控制效果。如果“因子”设置过小，可能会导致 PLC 无法运行。本测试设置为 4，应用的循环时间为 4ms。



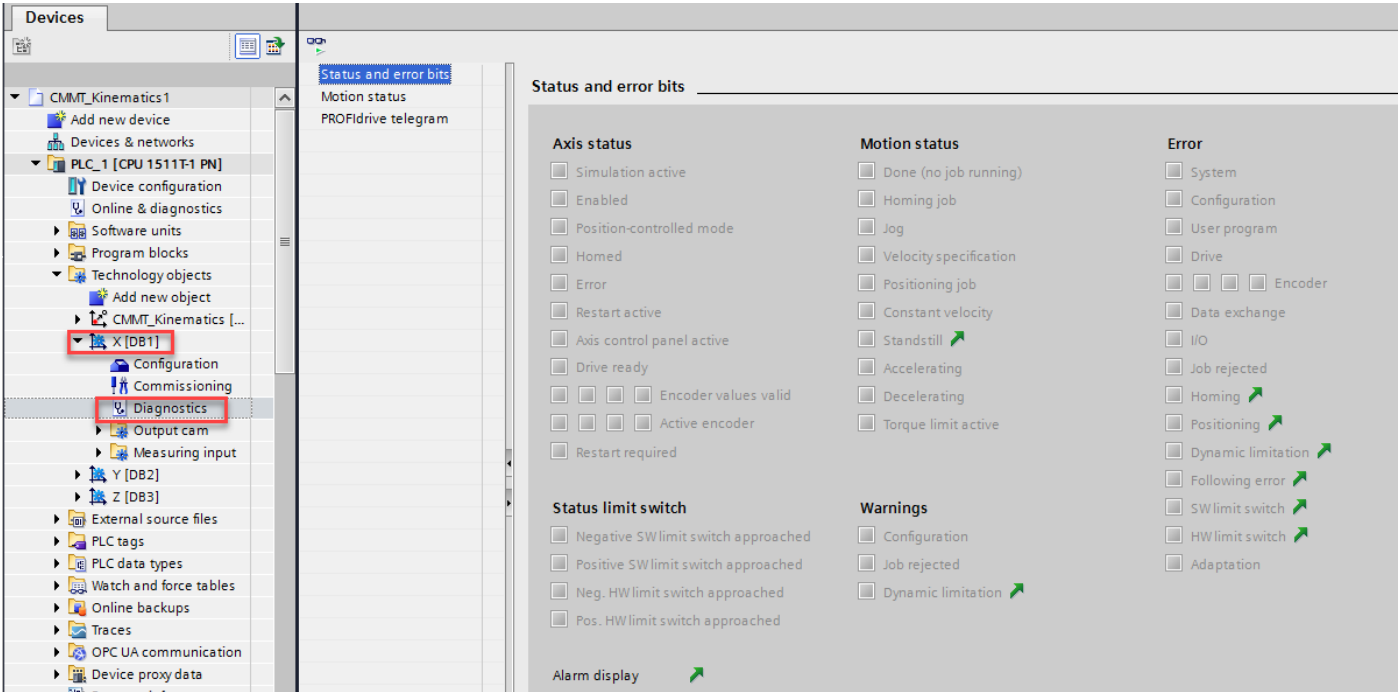
7 工艺面板调试

7.1 单轴调试面板

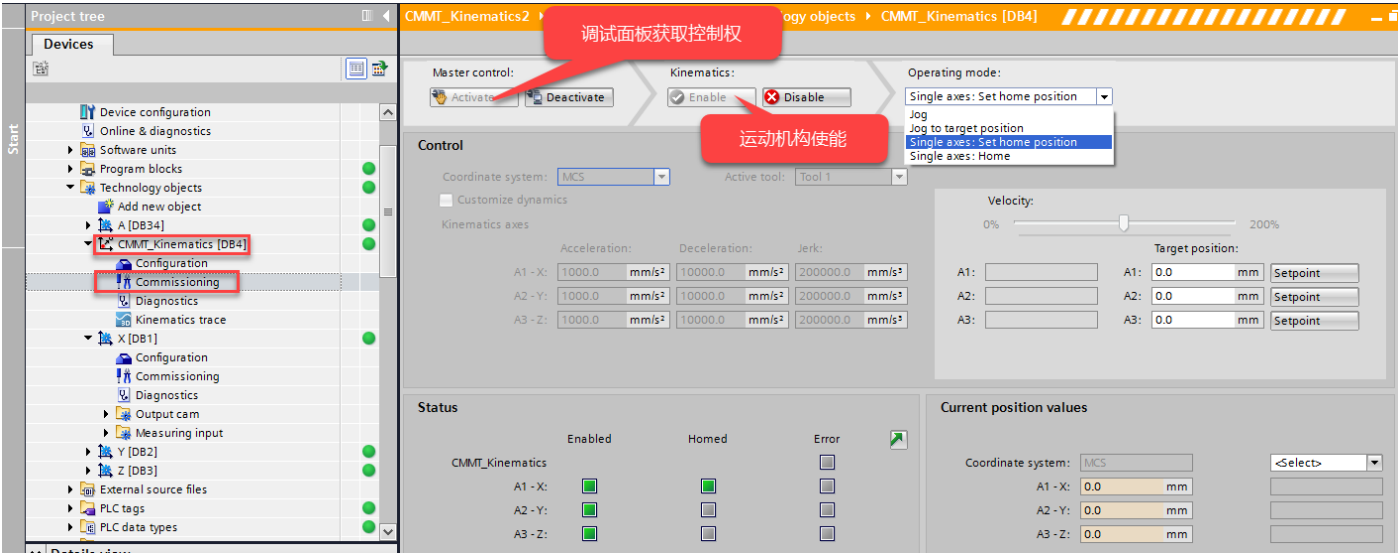


操作模式	
回原点	此功能相当于主动回原点。必须组态回原点参数。 对于绝对编码器不能使用回原点。将该模式与绝对编码器配合使用时，无法引用工艺对象。
设置回参考点位置	此功能相当于直接回原点（绝对）。 使用“开始” (Start) 按钮，可将实际位置设置为“位置” (Position) 中指定的值，并设置“已回原点” (Homed) 状态。
点动	通过点动方式执行运动指令。 使用“向前” (Forward) 或“向后” (Backward) 按钮可开始朝正方向或负方向运动。只要按住鼠标左键不放，运动就会继续进行。
速度参数/速度设定值	轴将按指定的速度移动，直到将其停止。
相对定位	根据“Control”下指定的默认值，以相对行进运动方式进行定位。
绝对定位	根据“Control”下指定的默认值，以绝对行进运动方式进行定位。

7.2 单轴状态及故障面板

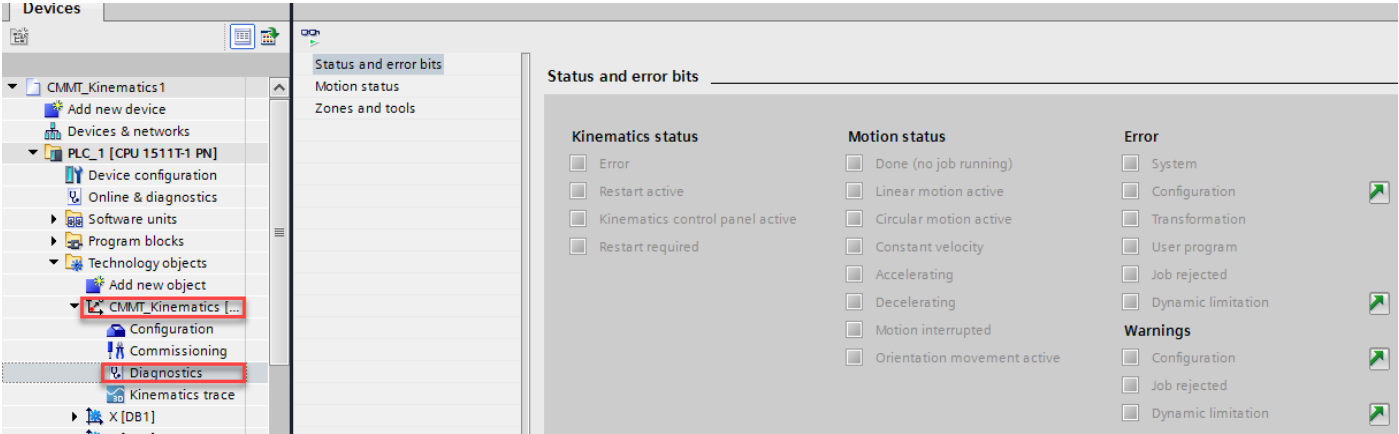


7.3 运动机构调试面板

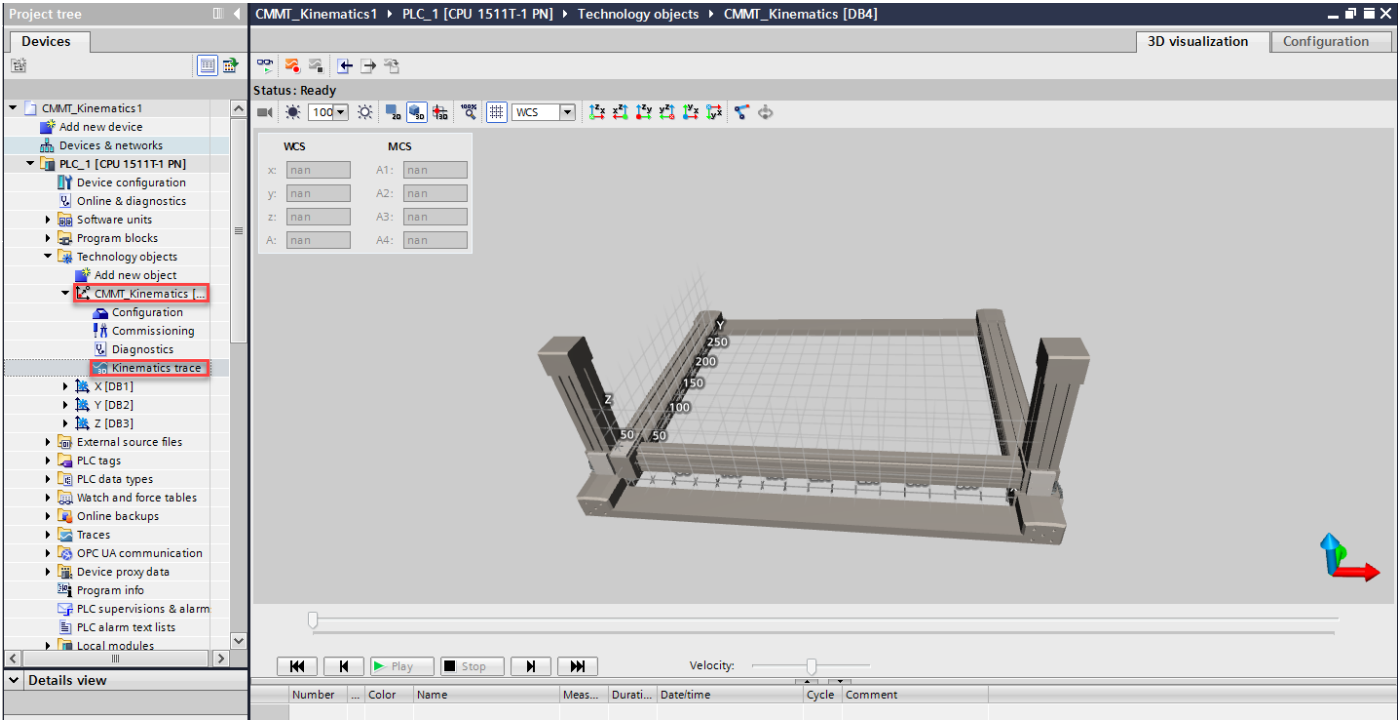


操作模式	说明
点动	使用“向前”(Forward) 按钮，将以正方向点动移动轴。使用“向后”(Backward) 按钮，将以负方向点动移动轴。
点动到目标位置	使用“向前”(Forward) 按钮，运动系统或轴将点动移动到“目标位置”(Target position)中指定的位置处。按住“向前”(Forward) 按钮时，运动系统将进行移动。到达目标位置时，运动系统将自动停止移动。 所指定的位置与“坐标系”(Coordinate system) 下拉列表中选择坐标系相关。
单轴：设置原点位置	使用“设置”(Set) 按钮，可将“目标位置”(Target Position) 值设置为相应轴的原点位置。将相应轴将设置为“已回原点”(Homed) 状态。 指定位置与此操作模式下“坐标系”(Coordinate system) 下拉列表中预设的机床坐标系(MCS) 相关。 该功能相当于直接回原点（绝对）。 绝对编码器不支持主动回原点功能。将该模式与绝对编码器配合使用时，无法引用工艺对象。
单轴：回原点	使用“启动”(Start) 按钮，轴将点动移动到预定义的原点位置处。按住“启动”(Start) 按钮时，相应轴将进行移动。到达原点位置时，轴将自动停止移动。

7.4 运动机构状态及故障面板

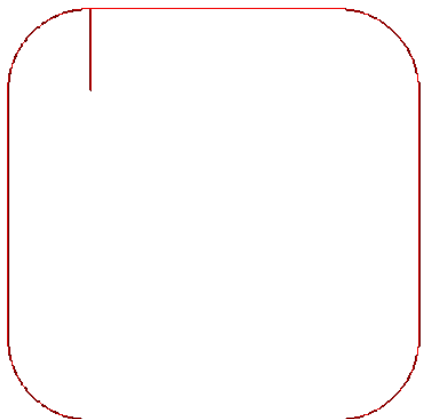


7.5 运动机构轨迹监控



8 实例演示

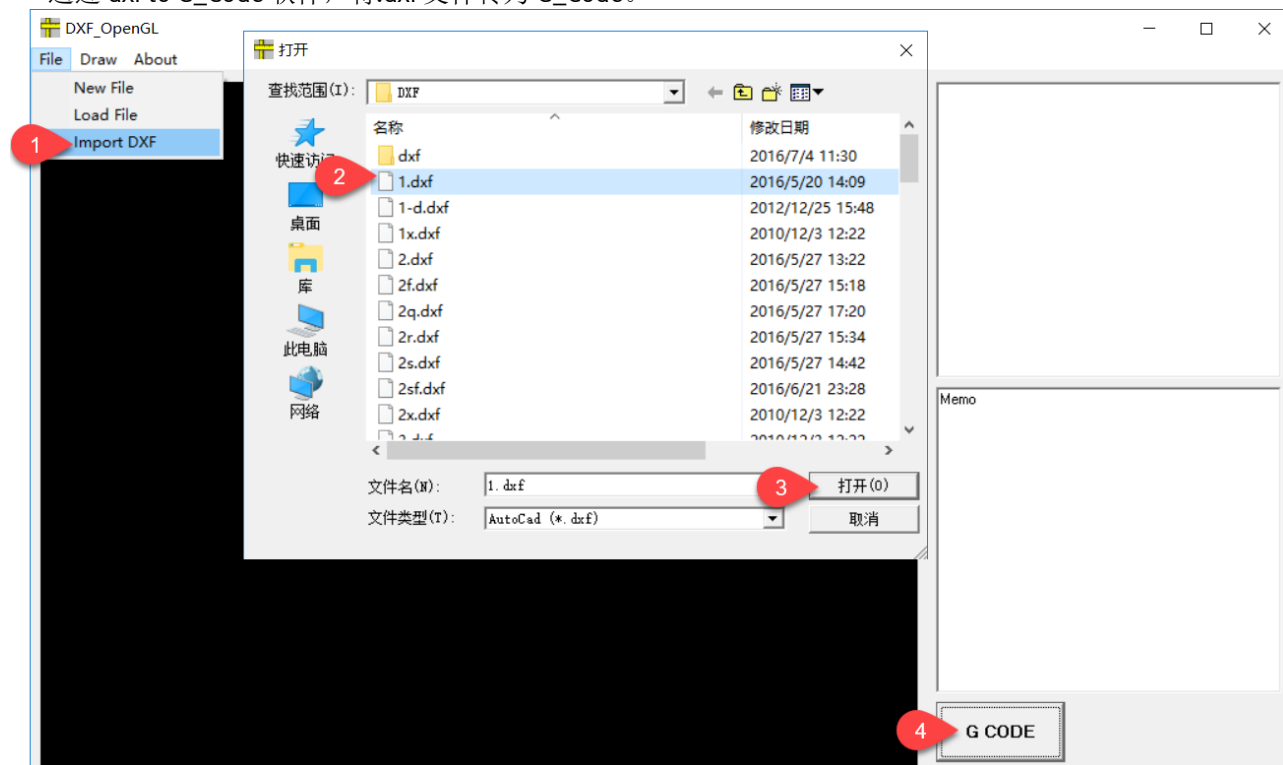
实例：模拟客户一个涂胶工艺，轨迹如下图所示：



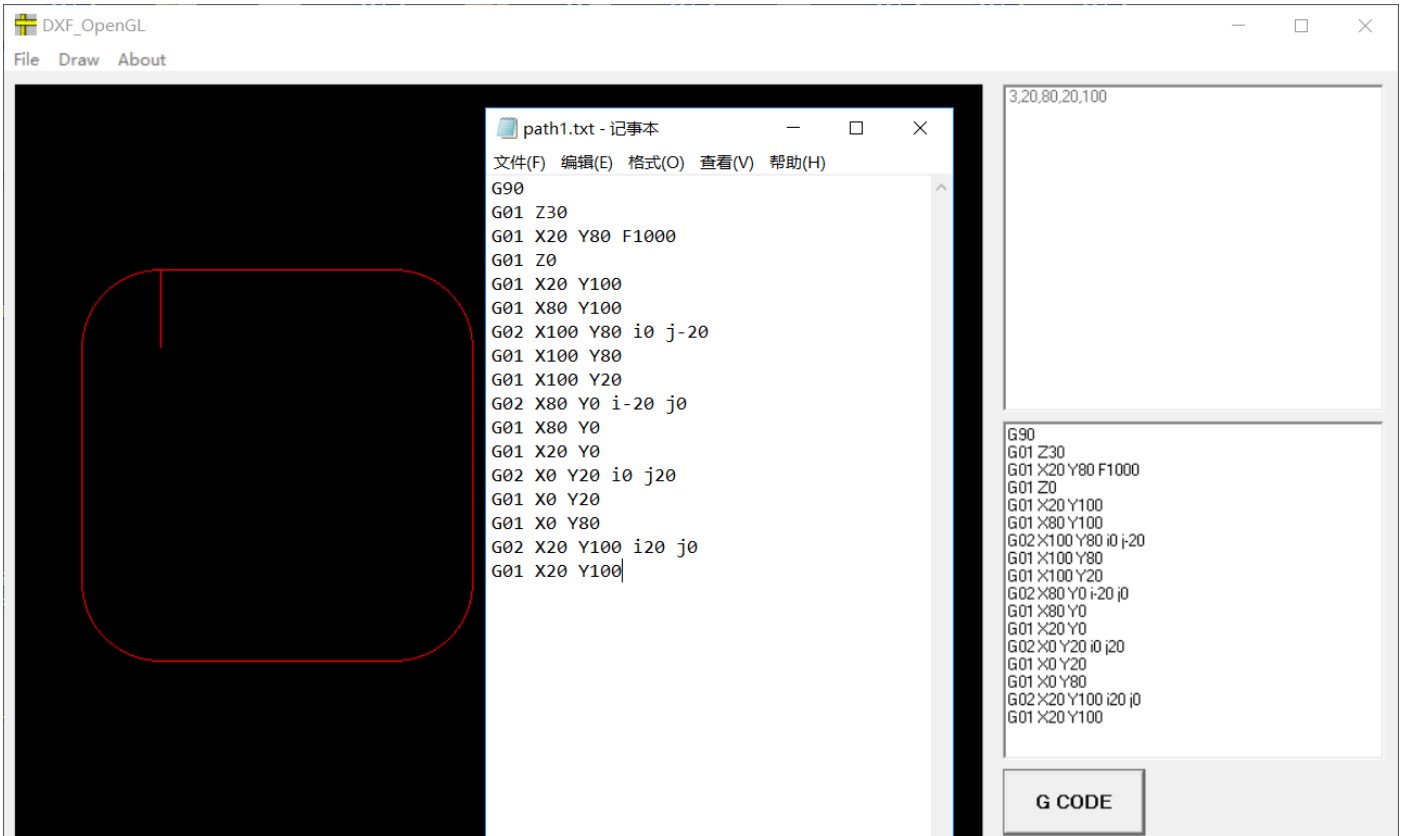
要求：在第 3 条 G 代码开始之前打开胶枪，在第 16 条 G 代码开始之前关闭胶枪，并在第 16 条 G 代码之后暂停，待确认信号后，继续走完剩下轨迹。

8.1 dxf 文件转成 db 文件

- 使用 CAD 软件画好轨迹，并保存为.dxf 文件。
- 通过 dxf to G_Code 软件，将.dxf 文件转为 G_Code。

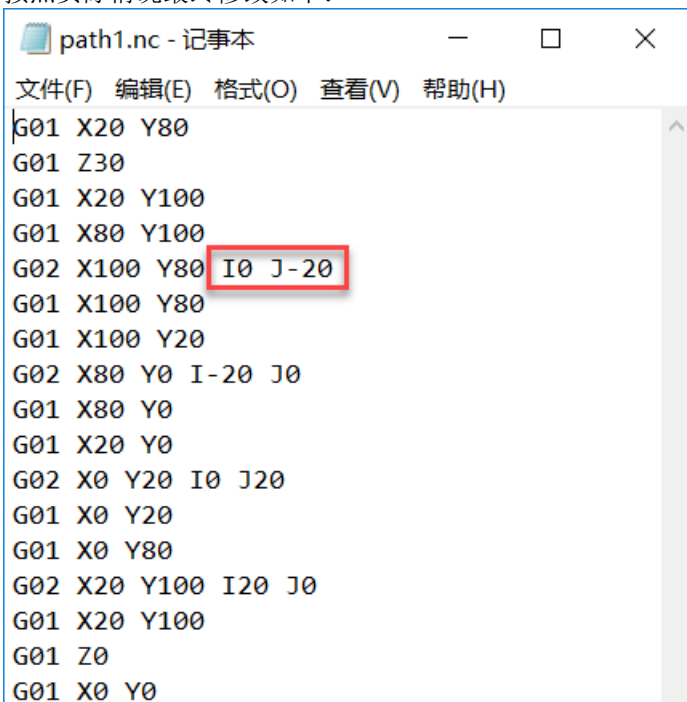


- 新建一个 txt 文本文档，将转换好的 G_Code 复制在里面，按照实际情况可以适当修改 G 代码并保存。然后将该 txt 文本后缀名改成.nc，如果后续还想修改 G 代码，还是可以用记事本打开.nc 文件修改的。



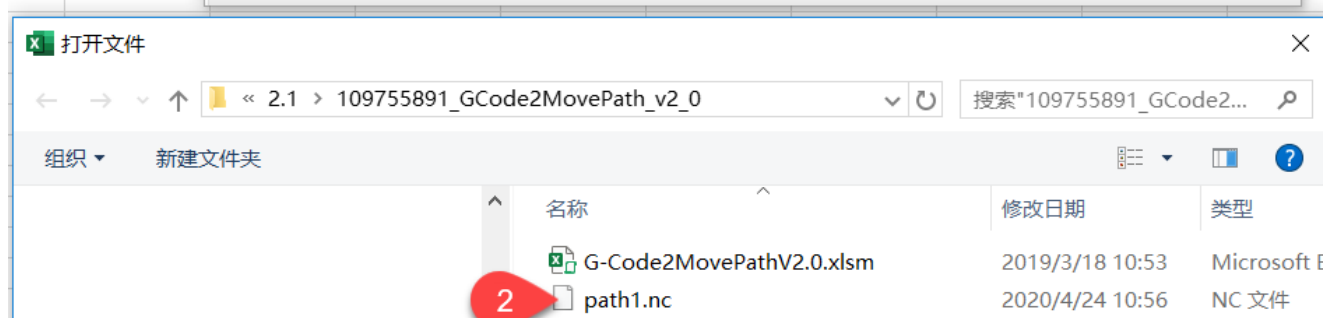
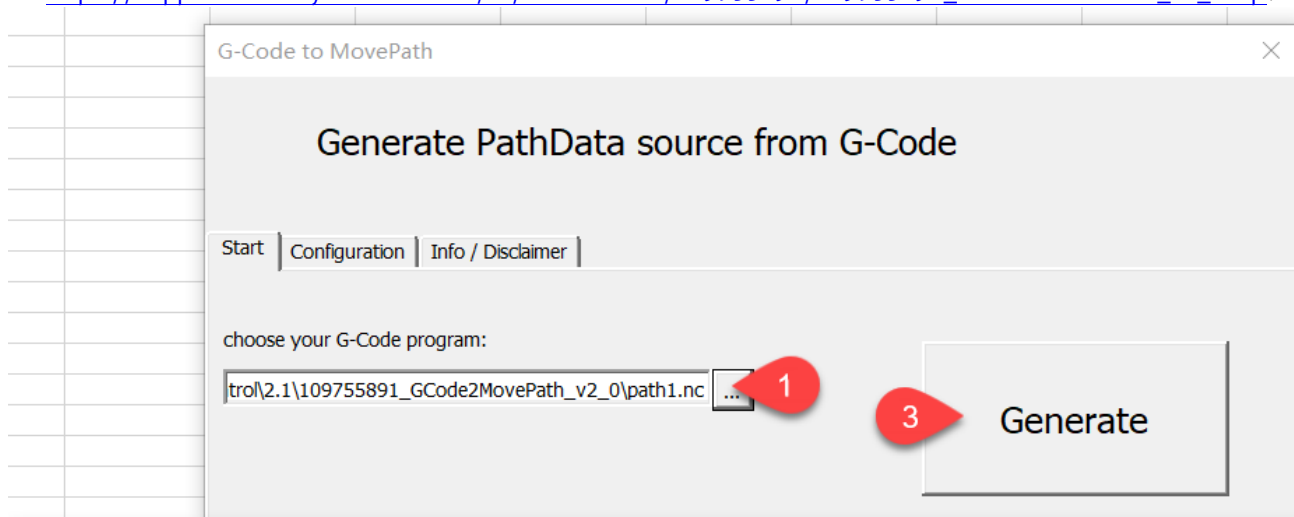
有些转换软件会将 G 代码转成小写，例如 i0 j-20，此时需要将其改为大写，否则在下一步转换中，无法被博途识别。

按照实际情况最终修改如下：

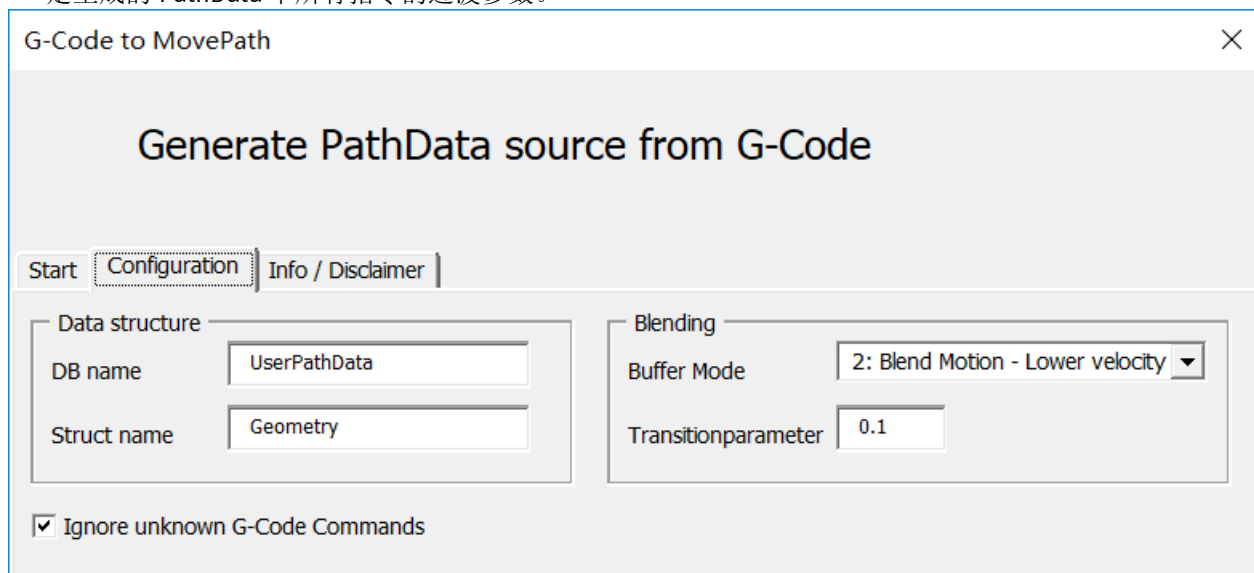


- 双击运行 G-Code2MovePathVxx.xlsm。如下图所示，在 Start 窗口下选择之前已经创建好的后缀名为“.nc”的 G 代码文件。（G-Code2MovePathVxx.xlsm 由西门子提供，下载地址：

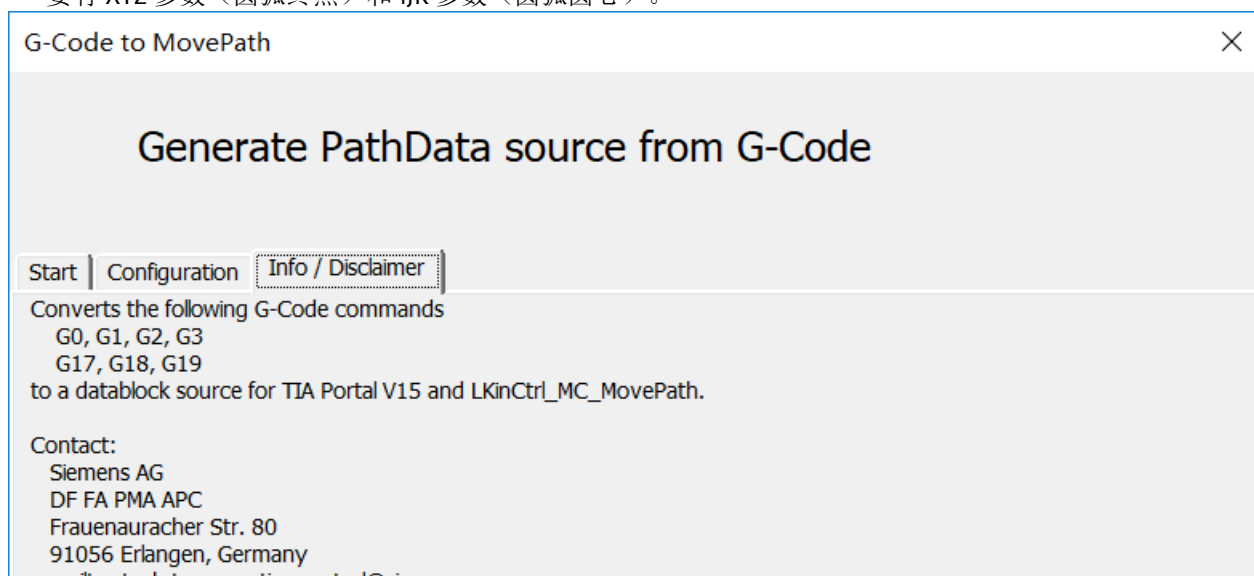
https://support.industry.siemens.com/cs/attachments/109755891/109755891_GCode2MovePath_v2_0.zip）



- 选择好 G 代码文件以后就可以开始生成了。顺利完成后，会在 G 代码源文件相同目录下会生成一个后缀名为“.db”的同名的 PathData 文件。上图例子的输出文件是“Path1.db”。
- 在工具的 Configuration 选项卡中还有一些生成文件的可选配置项。下图演示了如何更改导出的 DB 文件和 PathData 数据结构的名称。输入框中显示的是默认名称。此外，我们还可以通过设定 Buffer mode 和 Transition parameter 来设定生成的 PathData 中所有指令的过渡参数。

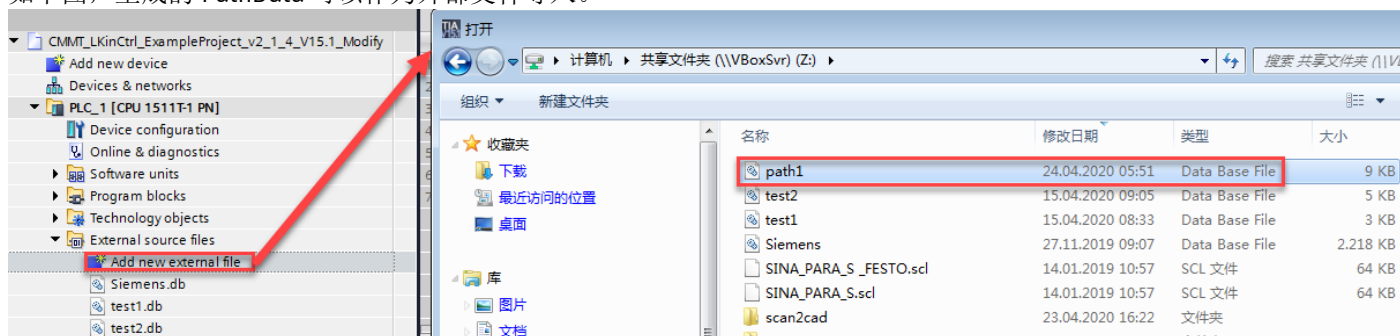


- 下图中展示的 Info / Disclaimer 选项卡列出了支持转换的 G 代码指令。圆弧 G 代码格式必须遵循 G2 或者 G3 格式并且要有 XYZ 参数（圆弧终点）和 IJK 参数（圆弧圆心）。

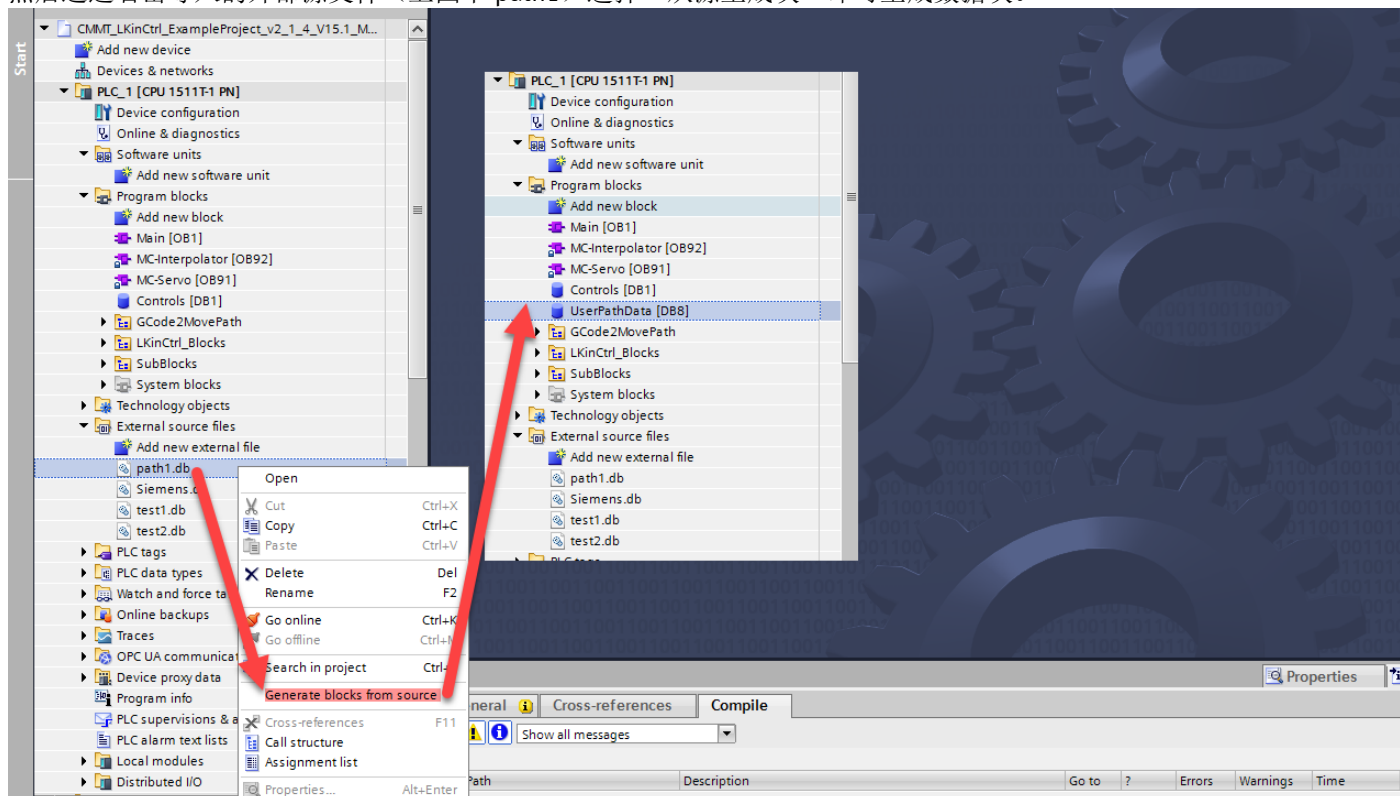


8.2 将 db 文件导入博途并生成 DB 数据块

如下图，生成的 PathData 可以作为外部文件导入。



然后通过右击导入的外部源文件（上图中 path1）选择“从源生成块”即可生成数据块。



默认生成的名称为 **UserPathData**，本例将 UserPathData[DB8]重命名为 **path1[DB8]**备用。

path1		
	Name	Data type
1	Static	
2	Geometry	"LKinCtrl_typePathData_advanced"
3	pathDataName	String["LKINCTRL_LENGTH_OF_PATHDATA_NAME"]
4	pointers	"LKinCtrl_typeAdvPointers"
5	commands	Array[1.."LKINCTRL_NO_OF_PATHDATA_ELEMENTS_..." of "LKinCtrl_typePathDataElement_advanced"
6	commands[1]	"LKinCtrl_typePathDataElement_advanced"
7	cmdType	Int
8	cmdActivated	Bool
9	cmdName	String["LKINCTRL_LENGTH_OF_CMDPOINT_NAME"]
10	cmdCoordinates	"LKinCtrl_typePointCoordinates"
11	cmdParameters	"LKinCtrl_typePointParameter"
12	setFlags	Array[1.."LKINCTRL_NO_OF_CMD_SETFLAGS"] of "LKinCtrl_typeSetFlagElement"
13	parameterValid	Array[1..4] of Bool
14	commands[2]	"LKinCtrl_typePathDataElement_advanced"
15	commands[3]	"LKinCtrl_typePathDataElement_advanced"
16	commands[4]	"LKinCtrl_typePathDataElement_advanced"
17	commands[5]	"LKinCtrl_typePathDataElement_advanced"

8.3 PathData 参数修改

所生成 PathData 为 LKinCtrl_typePathData_advanced 类型。根据实际应用需求，调整路径数据参数。

path1			
	Name	Data type	Start value
1	Static		
2	Geometry	"LKinCtrl_typePathData_advanced"	
3	pathDataName	String["LKINCTRL_LENGTH_OF_PATHDATA_NAME"]	'Glue_Path'
4	pointers	"LKinCtrl_typeAdvPointers"	
5	commands	Array[1.."LKINCTRL_NO_OF_PATHDATA_ELEMENTS_ADVANCED"] of "LKinCtrl_typePathDataElement_advanced"	
6	commands[1]	"LKinCtrl_typePathDataElement_advanced"	
7	cmdType	Int	1
8	cmdActivated	Bool	TRUE
9	cmdName	String["LKINCTRL_LENGTH_OF_CMDPOINT_NAME"]	
10	cmdCoordinates	"LKinCtrl_typePointCoordinates"	
11	cartesianPosition	"LKinCtrl_typeCartesianPosition"	
12	x	LReal	20.0
13	y	LReal	80.0
14	z	LReal	0.0
15	a	LReal	0.0
16	coordSystem	DInt	0
17	cmdParameters	"LKinCtrl_typePointParameter"	
18	pathDynamics	"LKinCtrl_typePathDynamics"	
19	velocity	LReal	-1.0
20	acceleration	LReal	-1.0
21	deceleration	LReal	-1.0
22	jerk	LReal	-1.0
23	orientationDirection	DInt	3
24	bufferMode	DInt	2
25	transitionParameters	Array[1..5] of LReal	
26	transitionParameters[1]	LReal	0.1
27	transitionParameters[2]	LReal	0.0
28	transitionParameters[3]	LReal	0.0
29	transitionParameters[4]	LReal	0.0
30	transitionParameters[5]	LReal	0.0
31	dynamicAdaption	DInt	-1
32	offsetParameters	"LKinCtrl_typeToolParameters"	
33	circleParameters	"LKinCtrl_typeCircleParameters"	
34	setFlags	Array[1.."LKINCTRL_NO_OF_CMD_SETFLAGS"] of "LKinCtrl_typeSetFlagElement"	
35	setFlags[1]	"LKinCtrl_typeSetFlagElement"	
36	flag	Int	-1
37	flagMode	USInt	0
38	remainingDistance	LReal	-1.0
39	setFlags[2]	"LKinCtrl_typeSetFlagElement"	
40	setFlags[3]	"LKinCtrl_typeSetFlagElement"	
41	parameterValid	Array[1..4] of Bool	
42	commands[2]	"LKinCtrl_typePathDataElement_advanced"	

上图展示了数据块视图中 PathData 结构的第一个指令的组态。PathData 最上层的数据结构包含了 PathData 的字符串名称和指令数组。名称的字符串长度默认是 10 个字符，也可以通过库常量 LKinCTRL_LENGTH_OF_PATHDATA_NAME 进行调整。

每一个 Commands 数组成员代表一条指令。通过变量 cmdType 来选择指令类型（见下表）。

TO Kinematics 运动指令	cmdType
MC_MoveLinearAbsolute	1
MC_MoveLinearRelative	2
MC_MoveCircularAbsolute	3
MC_MoveCircularRelative	4
激活左侧工具尺寸补偿	41
激活右侧工具尺寸补偿	42
关闭工具尺寸补偿	40
等待	100

注意：默认值 cmdType = -1 代表不激活该条指令并且 PathData 指令列表到达结束点。
通过布尔量 cmdActivated 每一条指令都可以激活或者取消激活。未激活的指令将不执行。
cmdActivated = FALSE 将会不激活 PathData 中的这条指令，该指令也不会被 LKinCtrl_MC_MovePath 执行。

cmdCoordinates 和 cmdParameters 包含了配置系统运动指令的基本参数。这里可以调整目标坐标位置、速度、加减速速度、过度模式等。

另外，子结构体 ‘flags’ 还提供了通过功能块 LKinCtrl_MovePath 置位数字量输出的功能，flag 功能的详细信息见 9.4 章节。

8.4 重新链接新的 PathData

将生成的数据块（path1）链接到 LKinCtrl_MC_MovePath 的 pathData 管脚上

▼ CMMT_LKinCtrl_ExampleProject_v2_1_4_V15.1_M...

▼ PLC_1 [CPU 1511T-1 PN]

▼ Software units

▼ Program blocks

▼ GCode2MovePath

▼ GCode2MovePath_Call [OB125]

▼ GCode2MovePath_FB1

▼ GCode2MovePath_DB [DB16]

▼ path1 [DB8]

▼ test2 [DB14]

▼ LKinCtrl_Blocks

▼ SubBlocks

▼ System blocks

▼ Technology objects

▼ External source files

▼ PLC tags

▼ Show all tags

▼ Add new tag table

▼ Standard-Variablentabelle [86]

▼ MC_Constants [28]

▼ LKinCtrl_Tags

▼ LKinCtrl_PLC_Tags [41]

GCode2MovePath

	Name	Data type	Default value	Retain	Acc
1	▼ Input				
2	▼ <Add new>				
3	▼ Output				
4	▼ <Add new>				
5	▼ InOut				

&

>=1

??

←

→

↔

[-]

#instLKInCtrl_

MC_MovePath

%FB35000

"LKInCtrl_MC_MovePath"

... EN

false execute

false stop

false interrupt

false reset

0 sequenceMode

0 pathTransition

1 bufferMode

... transitionParameter

#statConfiguration

n configuration

#toKinematics

axesGroup

"path1".Geometry

#statFlags

flags

done false

busy false

active false

commandAborted false

error false

status 16#0

diagnostics ...

activePathData ""

activeCmdNo 0

remainingDistanceActCmd 0.0

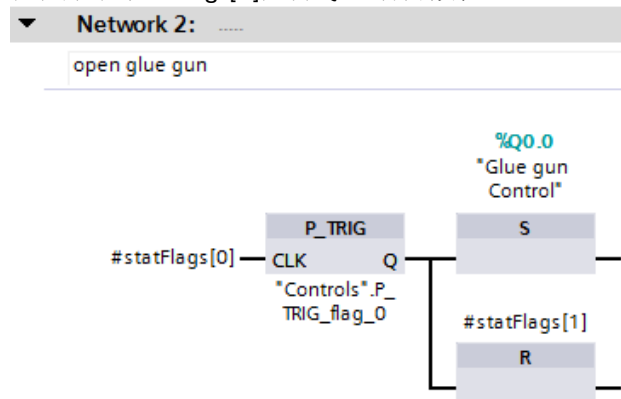
readyForNextPath true

ENO

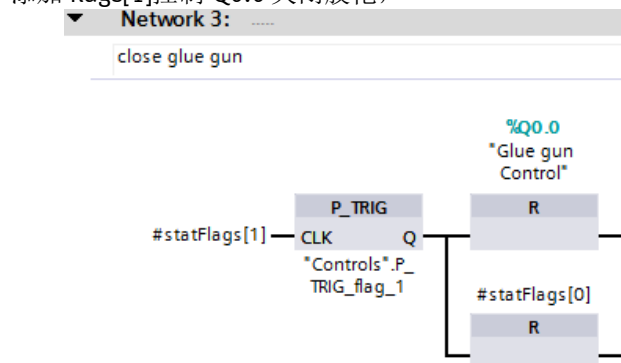
将用户的Pathdata链接到这里

8.5 添加涂胶控制程序（样例程序中已添加）

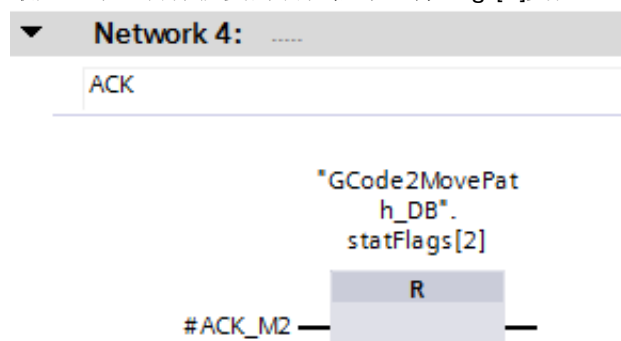
在程序中添加 flags[0]控制 Q0.0 打开胶枪，



添加 flags[1]控制 Q0.0 关闭胶枪，



最后，添加暂停恢复的确认信号，将 flags[2]复位。



完成后，下载程序。

8.6 控制测试

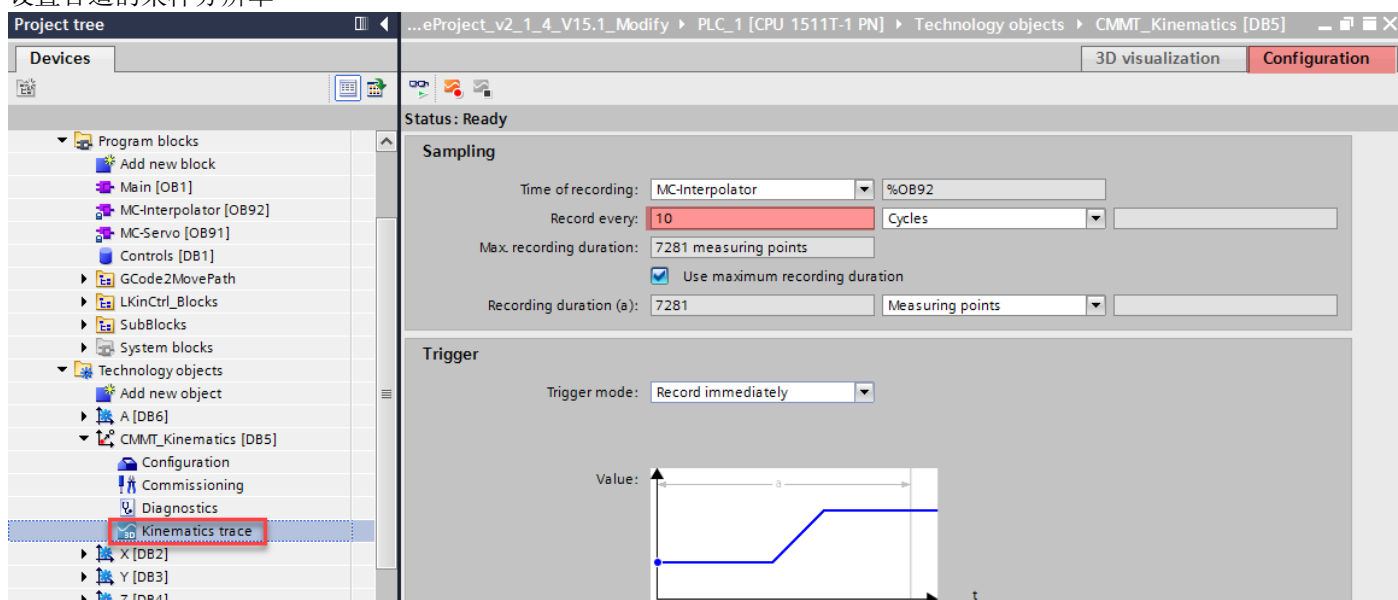
打开监控表 GCode2MovePath，如下图所示：

CMMT_LKinCtrl_ExampleProject_v2_1_4_V15.1_Modify ▶ PLC_1 [CPU 1511T-1 PN] ▶ Watch and force tables ▶ GCode2MovePath					
	Name	Address	Display format	Monitor value	Modify value
1	// Basic Control				
2	"Controls".EnableGCode2MovePathDemo		Bool	☒ TRUE	TRUE
3	"Controls".EnableAxes		Bool	☒ TRUE	TRUE
4	"Controls".ResetAxes		Bool	☐ FALSE	FALSE
5	"Controls".HomeAxes		Bool	☐ FALSE	FALSE
6	"LKinCtrl_MC_JogFrame_DB".jogForward		Bool	☐ FALSE	
7	"LKinCtrl_MC_JogFrame_DB".jogBackward		Bool	☐ FALSE	FALSE
8	"LKinCtrl_MC_JogFrame_DB".jogToPosition		Bool	☐ FALSE	FALSE
9	"LKinCtrl_MC_JogFrame_DB".jogDirection		DEC	2	
10	"LKinCtrl_MC_JogFrame_DB".jogModeInc		Bool	☐ FALSE	
11	"LKinCtrl_MC_JogFrame_DB".jogIncrement		Floating-point nu...	1.0	
12	"Controls".TargetPosition_Jog.x		Floating-point nu...	0.0	
13	"Controls".TargetPosition_Jog.y		Floating-point nu...	0.0	
14	"Controls".TargetPosition_Jog.z		Floating-point nu...	0.0	
15	"Controls".TargetPosition_Jog.a		Floating-point nu...	0.0	
16	// G_Code Path Control				
17	"CMMT_Kinematics".Override.Velocity		Floating-point nu...	20.0	20.0
18	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.execute		Bool	☐ FALSE	FALSE
19	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.stop		Bool	☐ FALSE	FALSE
20	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.interrupt		Bool	☐ FALSE	FALSE
21	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.reset		Bool	☐ FALSE	FALSE
22	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.sequenceMode		DEC+/-	0	
23	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.pathTransition		DEC+/-	0	
24	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.bufferMode		DEC+/-	1	
25	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.transitionParameter[1]		Floating-point nu...	-1.0	
26	// TCP coordinate and velocity				
27	"CMMT_Kinematics".TcpInWcs.x.Position		Floating-point nu...	0.0	
28	"CMMT_Kinematics".TcpInWcs.y.Position		Floating-point nu...	0.0	
29	"CMMT_Kinematics".TcpInWcs.z.Position		Floating-point nu...	0.0	
30	"CMMT_Kinematics".TcpInWcs.a.Position		Floating-point nu...	0.0	
31	"CMMT_Kinematics".StatusPath.Velocity		Floating-point nu...	0.0	
32	"CMMT_Kinematics".StatusMotionQueue.NumberOfCommands		DEC+/-	0	
33	// MovePath status				
34	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.done		Bool	☐ FALSE	
35	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.busy		Bool	☐ FALSE	
36	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.active		Bool	☐ FALSE	
37	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.commandAborted		Bool	☐ FALSE	
38	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.error		Bool	☐ FALSE	
39	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.status		Hex	16#7000	
40	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.activeCmdNo		DEC+/-	17	
41	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.remainingDistanceActCmd		Floating-point nu...	0.0	
42	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.readyForNextPath		Bool	☒ TRUE	
43	// Diag status				
44	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.diagnostics.errorID		Hex	16#0000	
45	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.diagnostics.cmdNumber		DEC+/-	0	
46	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.diagnostics.stateMovePath		DEC+/-	0	
47	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.diagnostics.statusSubFunction		Hex	16#0000	
48	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.diagnostics.motionFBNumber		DEC+/-	0	
49	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.diagnostics.motionFBStatus		Hex	16#0000_0000	
50	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.diagnostics.kinematicsStatusWord		Hex	16#0000_0000	
51	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.diagnostics.kinematicsErrorWord		Hex	16#0000_0000	
52	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.instLKinCtrl_MC_ExecuteKinMotionCmd.errorID		Hex	16#0000_0000	
53	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.instLKinCtrl_MC_ExecuteKinMotionCmd.statError		Bool	☐ FALSE	
54	// Flags setting				
55	"path1".Geometry.commands[3].setFlags[1].flag		DEC+/-	0	0
56	"path1".Geometry.commands[3].setFlags[1].flagMode		DEC	1	1
57	"path1".Geometry.commands[3].setFlags[1].remainingDistance		Floating-point nu...	-1.0	
58	"path1".Geometry.commands[16].setFlags[1].flag		DEC+/-	1	1
59	"path1".Geometry.commands[16].setFlags[1].flagMode		DEC	1	1
60	"path1".Geometry.commands[16].setFlags[1].remainingDistance		Floating-point nu...	-1.0	
61	"path1".Geometry.commands[16].setFlags[2].flag		DEC+/-	2	2
62	"path1".Geometry.commands[16].setFlags[2].flagMode		DEC	15	15
63	// Flags Control				
64	"GCode2MovePath_DB".statFlags[0]		Bool	☐ FALSE	
65	"GCode2MovePath_DB".statFlags[1]		Bool	☐ FALSE	
66	"GCode2MovePath_DB".statFlags[2]		Bool	☐ FALSE	
67	"GCode2MovePath_DB".statFlags[3]		Bool	☐ FALSE	
68	"GCode2MovePath_DB".statFlags[4]		Bool	☐ FALSE	
69	"GCode2MovePath_DB".statFlags[5]		Bool	☐ FALSE	
70	"Glue gun Control"	%Q0.0	Bool	☐ FALSE	
71	"GCode2MovePath_DB".ACK_M2		Bool	☐ FALSE	FALSE
72	"GCode2MovePath_DB".instLKinCtrl_MC_MovePath.statActivePathDataNameOld		String	'Glue_Path'	
73	"LKinCtrl_MC_GroupPower_DB".diagnostics.errorDetail		DEC	0	
74	// Path offset				
75	"MC_SETOCSFRAME_DB".Execute		Bool	☐ FALSE	FALSE
76	"Controls".Offset.x		Floating-point nu...	0.0	0.0
77	"Controls".Offset.y		Floating-point nu...	0.0	0.0
78	"Controls".Offset.z		Floating-point nu...	0.0	0.0
79	"Controls".Offset.a		Floating-point nu...	0.0	
80	"Controls".Offset.b		Floating-point nu...	0.0	
81	"Controls".Offset.c		Floating-point nu...	0.0	
82	"switchtoOCS1"	%M300.0	Bool	☐ FALSE	FALSE

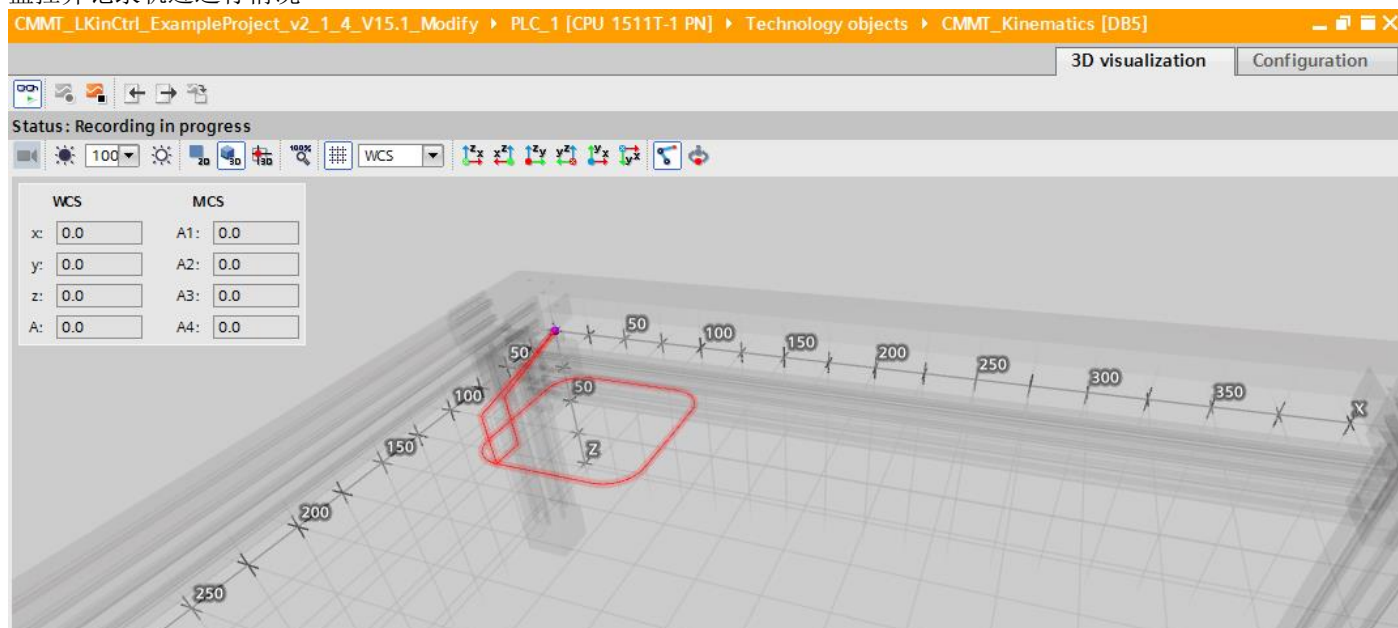
1. 使能 GCode2MovePath FB1 和三轴
 2. LKinCtrl_MC_JogFrame 三轴点动功能，注意点动速度（详见 9.3.1.2 章节）
 3. 激活三轴寻零，默认寻零模式为 7，将当前位置置零并将偏移量永久保存在 PLC 中（建议在用功能块寻零前，先用 FAS 软件对每个轴进行一次寻零）
 4. 上升沿触发轨迹运行指令，轨迹运行开始
 5. 当前 XYZ 坐标位置，速度以及运动序列中已排入队列的指令数量
 6. LKinCtrl_MC_MovePath 指令的状态输出管脚(详见 9.3.1.1 章节)
 7. LKinCtrl_MC_MovePath 指令的相关诊断信息（详见 10 章节）
 8. 设置标记位及其模式（详见 9.4 章节）
- 第 3 条 G 代码中激活 flag，编号设置为 0，模式为 1；在第 16 条 G 代码中激活 flag，编号为 1，模式为 1；在第 16 条代码中激活 flag，编号为 2，模式为 15。
9. 标记位的输出状态，胶枪的输出状态，以及暂停的确认信号。
- 在第 3 条 G 代码开始的同时，statFlag[0] 输出 TRUE，%Q0.0 打开胶枪；在第 16 条 G 代码开始的同时 statFlag[1] 输出 TRUE，同时它会将 statFlag[0] 复位，%Q0.0 复位胶枪，在本条路径运行结束后，statFlag[1] 也会自动复位。
- 在第 16 条 G 代码结束的同时，statFlag[2] 输出 TRUE，指令停止执行，运动停止，可使用 ACK_M2 将其复位，会继续执行剩下的指令。
10. Override.velocity 实时调速（0-200%）。
11. 轨迹偏移补偿（详见 9.5 章节）

8.7 运动机构轨迹监控中监控轨迹

设置合适的采样分辨率

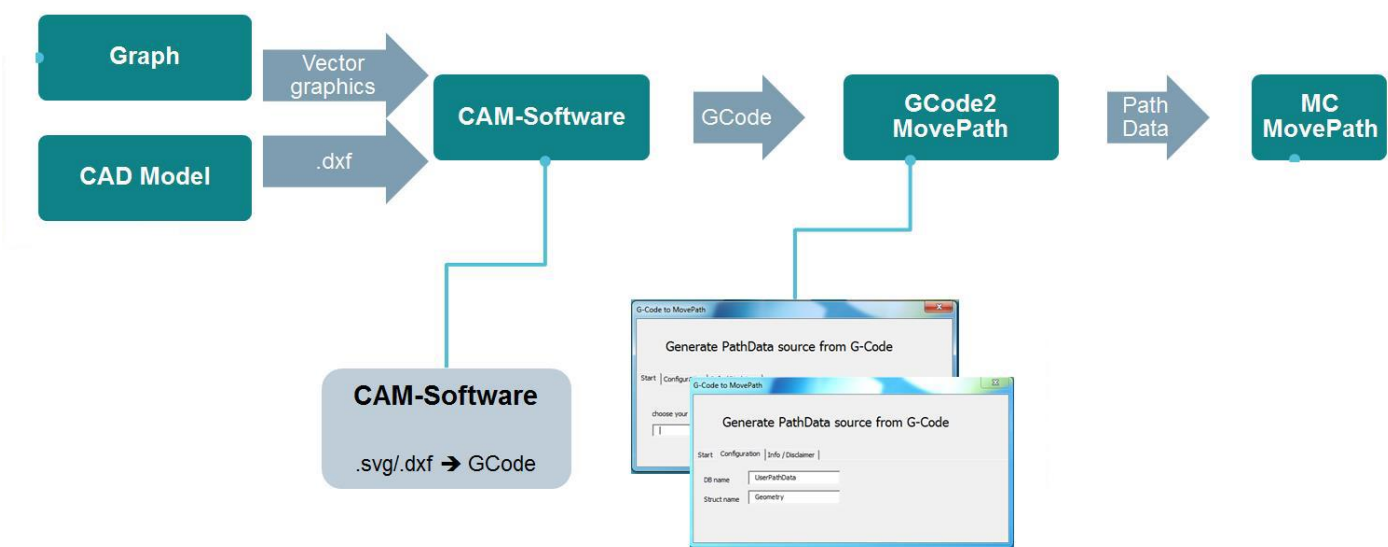


监控并记录轨迹运行情况



9 功能说明

9.1 G 代码转换为路径 DB 块的流程



- 1. 路径轮廓由矢量图而来。通过 CAM（computer aided manufacturing）软件（很多免费工具可用）从源文件生成 G 代码。
- 2. 通过 GCode2MovePath 工具将 G 代码转换为一个包含路径信息的 PathData 源。
- 3. 将 PathData 源转化为 PathData 数据块。
- 4. 使用功能块 LKinCtrl_MC_MovePath 调用 PathData 数据块。

9.2 PathData 结构体

PathData 结构体含有路径定义的指令列表。该数据类型作为库的标准数据类型，在一个全局数据块中进行初始化。为了适应不同的应用场景，一共有三种不同的 PathData 数据类型，它们之间的区别在于内部数据的等级和指令支持。每一种类型的长度都可以通过常量 LKINCTRL_NO_OF_PATHDATA_ELEMENTS_TYPE 根据需要进行调整以节省 PLC 存储空间。不同的 PathData 类型说明如下：

PathData_reduced	PathData	PathData_advanced
用于简单路径定义（例如运行到等待位置） • 仅支持直线插补指令 • 缩减的路径长度（可调整）	用于常规路径定义（例如简单路径，抓取） • 支持所有路径指令 • 中等的路径长度（可调整）	用于运行中能够连续插入指令的场合 • 支持所有路径指令 • 包含添加/提取指令到 PathData 的指针 • 较长的路径长度（可调整）

根据客户的 G 代码行数，如下图所示，在 PLC tags-LkinCtrl_Tags 中调整合适的值。



PLC_1 的资源

对象	装载存储器	代码工作存储器	数据工作存储器	保持性存储器	运动控制资源	
1	7%	38 %	154 %	0 %	30 %	
2						
3	总计:	未指定	230400 个字节	1048576 个字节	90784 个字节	800
4	已使用:	3839939 个字节	86588 个字节	1615788 个字节	0 个字节	240
5	详细信息					
6	OB	31501 个字节	3099 个字节			

常规 | **交叉引用** | **编译**

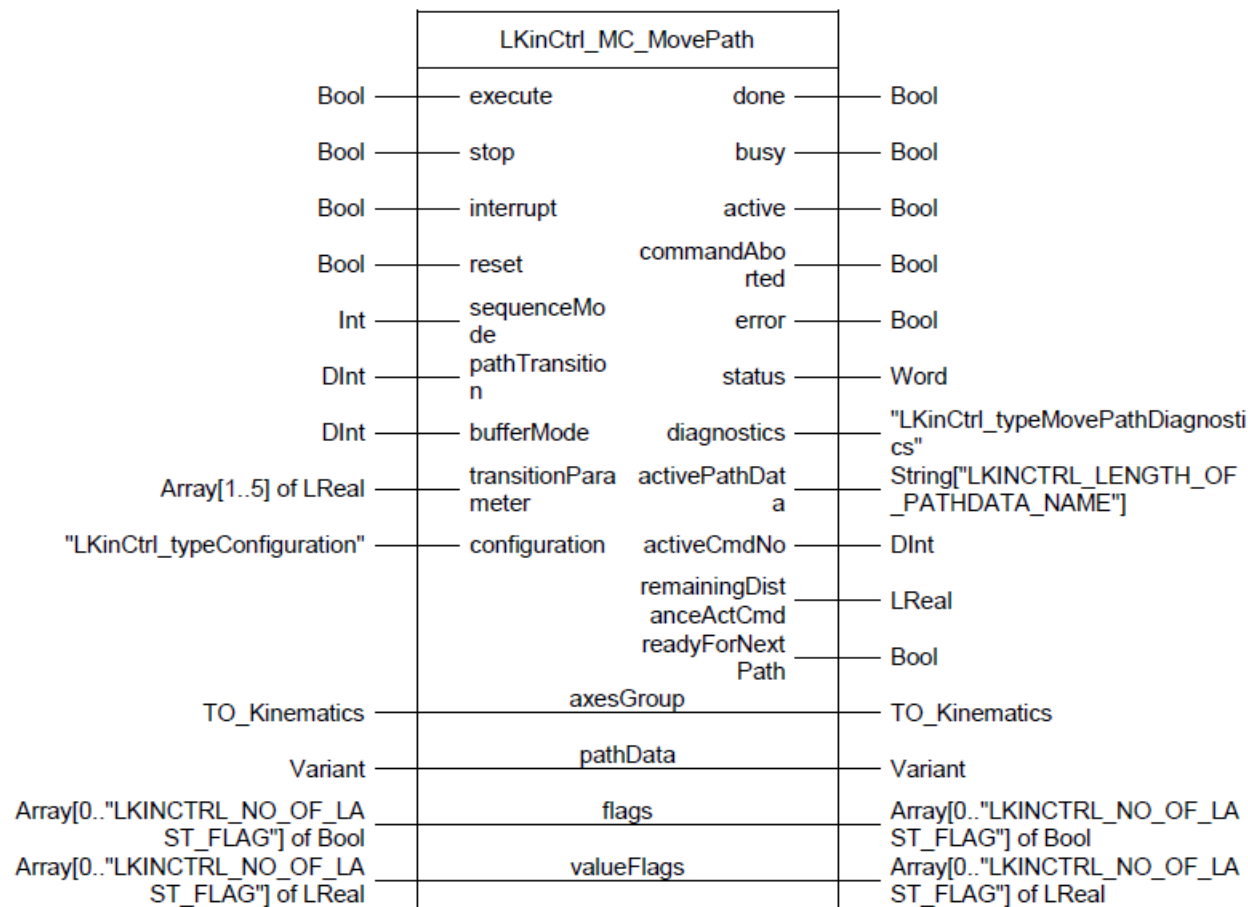
显示所有消息

消息	转至	日期	时间
下载前取消动作。		2020/6/22	13:55:55
下载前取消动作。		2020/6/22	14:30:51
下载前取消动作。		2020/6/22	14:36:41
开始下载到设备。		2020/6/22	16:10:58
PLC_1		2020/6/22	16:10:59
CPU上没有足够的代码空间或数据存储空间存储指定数量的数据。		2020/6/22	16:11:39
装裁中止（错误：1：警告：0）。		2020/6/22	16:11:44
后期下载已中止。		2020/6/22	16:11:44

9.3 LKinCtrl 库的运行

9.3.1 常用功能块介绍

9.3.1.1 功能块 LKinCtrl_MC_MovePath (FB 35000)



工作原理

LKinCtrl_MC_MovePath 功能块负责组织和运行此应用的完整功能。它负责控制运动机构工艺对象运行定义好的路径指令并给出不同状态下诊断信息。

支持的常用功能

- 启动/恢复路径运动执行
- 停止/中断路径运动执行
- 复位功能块错误（非工艺对象错误）
- 选择运行模式（自动/顺序模式）
- 通过 flags 控制执行器

支持的诊断信息

和 PLCOpen 类似的运动状态信息

错误和状态信息

- 功能块内部
- 运动指令错误
- 工艺对象错误

路径运动状态

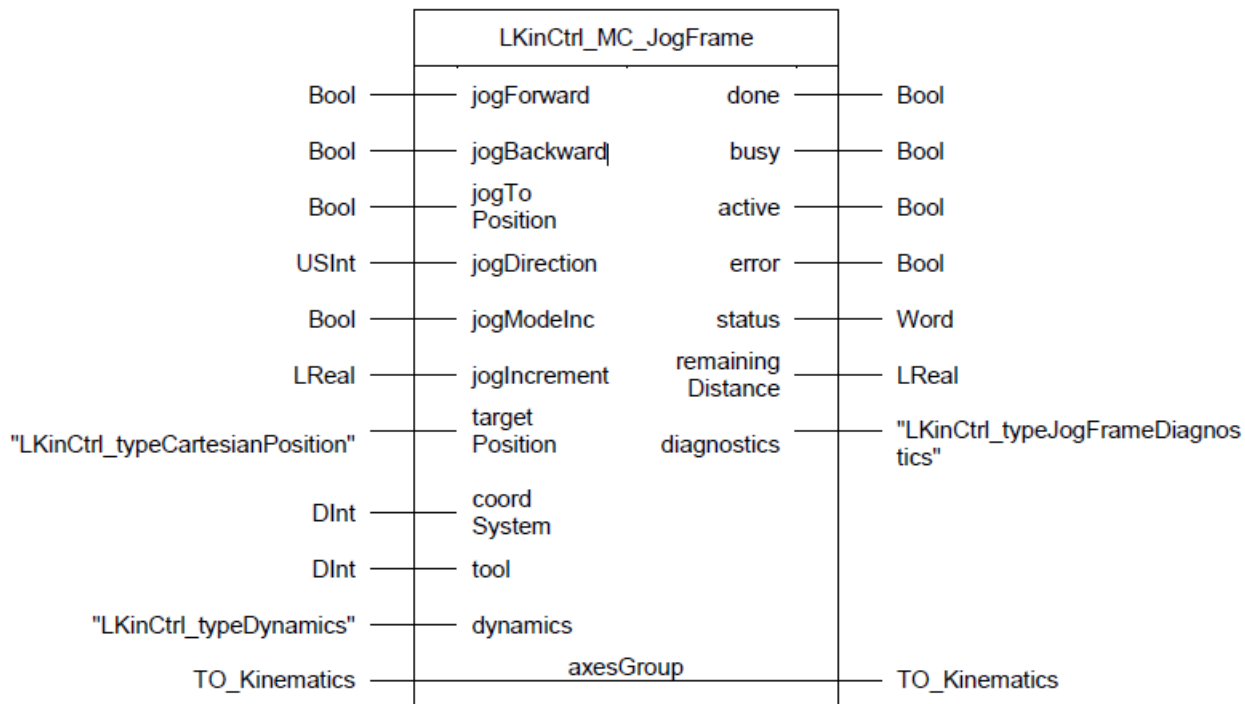
- 激活的 PathData（名称）
- 激活的 PathData 指令（序号）
- 激活的指令剩余运动距离

LKinCtrl_MC_MovePath接口参数

名称	参数类型	数据类型	注释
execute	IN	Bool	上升沿触发一次
stop	IN	Bool	终止路径运动
interrupt	IN	Bool	中断路径运动
reset	IN	Bool	复位 FB / 确认错误
sequenceMode	IN	Int	激活顺序模式(0) / 单步模式(1)

pathTransition	IN	DInt	新路径的初始过渡参数选择 (0: 使用PathData 中第一个Command自己规划的参数1: 使用功能块 bufferMode 和transitionParameter参数)
bufferMode	IN	DInt	新路径的初始bufferMode
transitionParameter	IN	Array[1..5] of LReal	新路径的初始transitionParameter
configuration	IN	"LKInCtrl_typeConfiguration"	配置数据结构
done	OUT	Bool	TRUE: 指令顺利执行完成
busy	OUT	Bool	TRUE: 功能块运行中, 等待输出值
active	OUT	Bool	功能块控制轴组中/运动机构运行中
commandAborted	OUT	Bool	TRUE:运行中的指令被另一个指令终止
error	OUT	Bool	TRUE: 功能块运行中有错误发生
status	OUT	Word	功能块状态
diagnostics	OUT	"LKInCtrl_typeMovePathDiagnostics"	功能块诊断信息
activePathData	OUT	String["LKINCTRL_LENGTH_OF_PATHDATA_NAME"]	当前PathData名称
activeCmdNo	OUT	DInt	当前指令序号
remainingDistanceActCmd	OUT	LReal	当前指令剩余距离
readyForNextPath	OUT	Bool	TRUE:下一个路径可以开始运行
axesGroup	IN_OUT	TO_Kinematics	运动机构工艺对象
pathData	IN_OUT	Variant	路径数据 PathData
flags	IN_OUT	Array[0.."LKINCTRL_NO_OF_LAST_FLAG"] of Bool	路径指令里配合控制执行器的布尔数组
valueFlags	IN_OUT	Array[0.."LKINCTRL_NO_OF_LAST_FLAG"] of LReal	路径指令里配合控制执行器的LREAL数组

9.3.1.2 功能块 LKinCtrl_MC_JogFrame (FB 35010)



工作原理

LKinCtrl_MC_JogFrame 功能块能够在笛卡尔坐标系中的 X, Y, Z 和旋转方向上点动运动机构。而且,还可以在笛卡尔空间内直接定位到某一目标位置。该功能块通过“jogForward”, “jogBackward”和“jogToPosition”的上升沿启动点动操作,下降沿停止运动。运动机构还可以通过配置"coordSystem"参数选择在不同的坐标系下进行点动操作。也可以通过配置"tool"参数选择不同工具进行点动。点动的动态参数通过"dynamics"参数输入,如果未配置此参数,则采用运动机构工艺对象中的默认动态参数。

当功能块被调用时，输出"busy"会被置位。当有运动时，输出“active”会被置位。当点动操作成功完成时，输出"done"会被置位。当点动命令置位并保持至少一个程序周期时，如果有错误发生，输出"error"也会随命令置位，详细的错误信息可以在输出"diagnostics"中查询。功能块当前的状态通过输出"status"显示。当使用增量式点动或者设定目标位置的方式点动时，当前点动剩余距离可以在输出“remainingDistanceActCmd”中显示。

连续点动模式

当“jogForward”或“jogBackward”有上升沿时，运动机构将进入连续点动模式。只有当相同命令有下降沿或者工艺对象出现错误时，点动才会停止。

增量式点动模式

增量式点动模式通过置位输入"jogModeInc"选择，点动的距离通过输入"jogModeInc"设置，点动通过输入"jogForward"或者"jogBackward"上升沿触发，在到达点动的距离后，运动会自行停止，如果运动过程中，点动命令出现下降沿，运动将直接停止。

点动到目标位置

如果要将运动机构点动到设定目标位置，必须置位输入"jogToPosition"，不过首先要定义目标位置"targetPosition"。

"jogToPosition"在运动过程中要保持，下降沿将会马上停止运动。

支持的功能：

在 X, Y, Z 和 A 方向点动运动机构（增量式/连续式）

点动运动机构到目标位置

指定工具 (tool1...3)

指定点动坐标系 (WCS, OCS1...3)

指定点动速度 (路径/旋转)

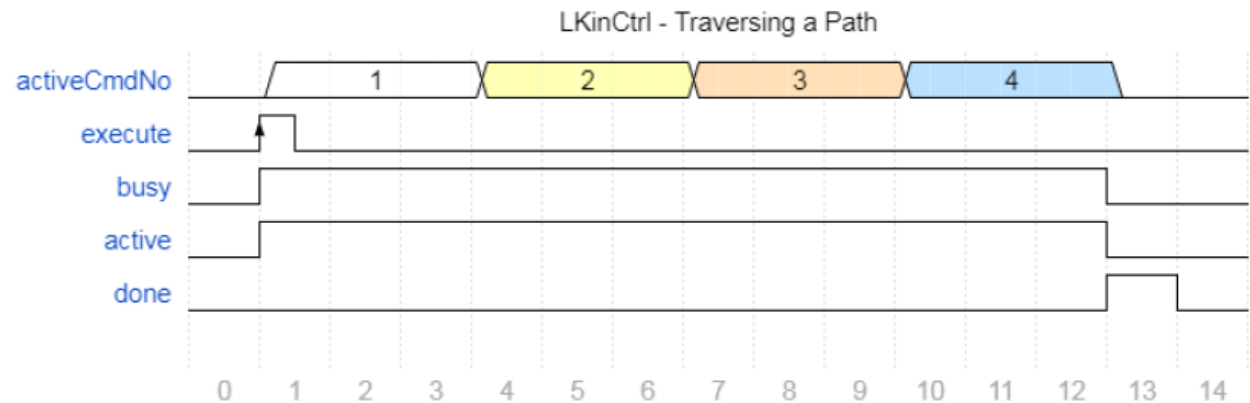
LKinCtrl_MC_JogFrame 接口参数

名称	参数类型	数据类型	注释
jogForward	IN	Bool	选择的坐标方向上正向点动
jogBackward	IN	Bool	选择的坐标方向上反向点动
jogToPosition	IN	Bool	点动到指定目标位置
jogDirection	IN	USInt	1=x-方向, 2=y-方向, 3=z-方向, 4=a-方向.
jogModeInc	IN	Bool	1=增量式点动模式 0=连续式点动模式
jogIncrement	IN	LReal	增量距离
targetPosition	IN	LKinCtrl_type CartesianPosition	目标位置坐标
coordSystem	IN	DInt	0=WCS, 1= OCS1, 2=OCS2, 3=OCS3.
tool	IN	DInt	1= tool1, 2=tool2, 3=tool3.
dynamics	IN	LKinCtrl_typeDynamics	动态参数设定
done	OUT	Bool	TRUE: 运动指令顺利完成
busy	OUT	Bool	TRUE: 功能块运行中，等待输出值
active	OUT	Bool	TRUE: 计算设定值
error	OUT	Bool	TRUE: 功能块运行中出错
status	OUT	Word	16#0000 - 16#7FFF: 功能块状态, 16#8000 - 16#FFFF: 错误代码
remainingDistance	OUT	LReal	当前运动剩余距离
diagnostics	OUT	"LKinCtrl_typeJog FrameDiagnostics"	诊断信息
axesGroup	TO_Kinematics	Kinematics technology object	运动机构工艺对象

9.3.2 常用功能

• 接口控制

基于 PLCOpen Part4 行为标准, 功能块 LKinCtrl_MC_MovePath 的 execute 输入在检测到上升沿时启动。下图呈现了功能块的常用行为和时序。



stop 输入会在内部触发系统函数 MC_GroupStop。因此, stop 输入检测到上升沿时, 路径运动会终止, MotionQueue 中的所有指令都会被移除。

根据 MC_GroupInterrupt 指令, 输入 interrupt 检测到上升沿时, 只是中断路径运动。当前指令状态和 MotionQueue 中的指令以及运动都可以被继续。

中断后, execute 输入再次有上升沿时, 会恢复中断的路径运动。

如果 reset 检测到上升沿, 会执行错误复位或者一次普通的功能块复位。

注意: 复位功能块和确认错误仅在 execute = FALSE 并且运动机构没用运动时才可行。

通过 sequenceMode 输入可以选择功能块在自动还是单步模式下运行。运行模式详细信息见下面的 [9.3.3 章节](#)。

通过 configuration 输入可以配置内部 MC_GroupStop 的停止模式 (例如紧急停止)。

• 诊断和状态

instLKinCtrl_MC_MovePath.done
instLKinCtrl_MC_MovePath.busy
instLKinCtrl_MC_MovePath.active
instLKinCtrl_MC_MovePath.commandAborted
instLKinCtrl_MC_MovePath.error
instLKinCtrl_MC_MovePath.status
instLKinCtrl_MC_MovePath.activeCmdNo
instLKinCtrl_MC_MovePath.remainingDistanceActCmd
instLKinCtrl_MC_MovePath.readyForNextPath

诊断输出信息分为两个部分。第一部分是基于 PLCOpen 标准的输出 done, busy, active, commandAborted, error 和 status。它们的时序行为遵照 PLCOpen 标准。状态字 status 能够概括地给出功能块的状态和错误信息。第一部分中还提供了更进一步的信息用于分析路径运动状态和输出诊断信息里的错误信息。

路径运动状态

在执行路径运动的过程中, activePathData 和 activeCmdNo 输出能给出运行中的 PathData 名称和指令编号。remainingDistanceActCmd 还能显示单个指令的剩余距离。

布尔量 readyForNextPath 代表 PathData 在内部都已经下发到 MotionQueue。它不意味着所有指令都已经执行完成, 而是都已经在 MotionQueue 中等待执行。一旦 readyForNextPath 变为 TRUE, 功能块可以再次启动执行下一个 PathData。因此 MotionQueue 始终保持饱和和最大化的前瞻以便于动态规划。

诊断数据结构

instLKinCtrl_MC_MovePath.diagnostics.errorID
instLKinCtrl_MC_MovePath.diagnostics.cmdNumber
instLKinCtrl_MC_MovePath.diagnostics.stateMovePath
instLKinCtrl_MC_MovePath.diagnostics.statusSubFunction
instLKinCtrl_MC_MovePath.diagnostics.motionFBNumber
instLKinCtrl_MC_MovePath.diagnostics.motionFBStatus
instLKinCtrl_MC_MovePath.diagnostics.kinematicsStatusWord
instLKinCtrl_MC_MovePath.diagnostics.kinematicsErrorWord
instLKinCtrl_MC_MovePath.instLKinCtrl_MC_ExecuteKinMotionCmd.errorID
instLKinCtrl_MC_MovePath.instLKinCtrl_MC_ExecuteKinMotionCmd.statError

第二部分，除了功能块状态信息以外，诊断数据结构还包含了错误发生时写入的详细信息。
有了 PathData 名称和具体指令编号还有 LKinCtrl_MC_MovePath 和 LKinCtrl_ExecuteKinMotionCmd 功能块的状态机状态，就能够定位到有可能导致错误的指令，并且，运动机构工艺对象发生错误时的状态字和错误字也可以提供诊断信息。

9.3.3 运行模式

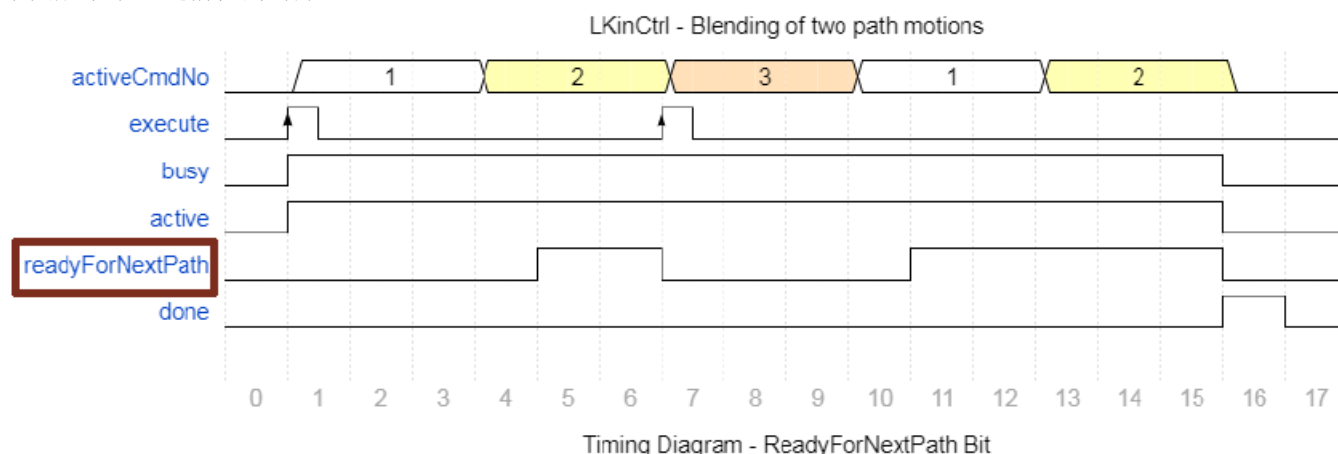
• 自动模式

要想功能块运行在自动模式，sequenceMode 输入必须设置为 0，同时这也是功能块的默认设置。
在自动模式下，功能块会运行完整的 PathData，在完成 PathData 中定义的最后一个指令以后，返回 done 信号。
在路径运动彻底结束以后（done = TRUE）或者 PathData 中的所有指令均已下发到 MotionQueue 中（readyForNextPath = TRUE）以后都可以再次触发功能块。

运动机构运行 PathData 过程中再次触发 MovePath

输出位信号 readyForNextPath 代表传输到 LKinCtrl_MC_MovePath 功能块的所有 PathData 中的指令均已注入到 MotionQueue。在此状态下可以再次触发功能块开始新的后续 PathData 路径运动。这种情况下，MotionQueue 仍然完全饱和并且在第一个 PathData 完成后会立即继续下一个。

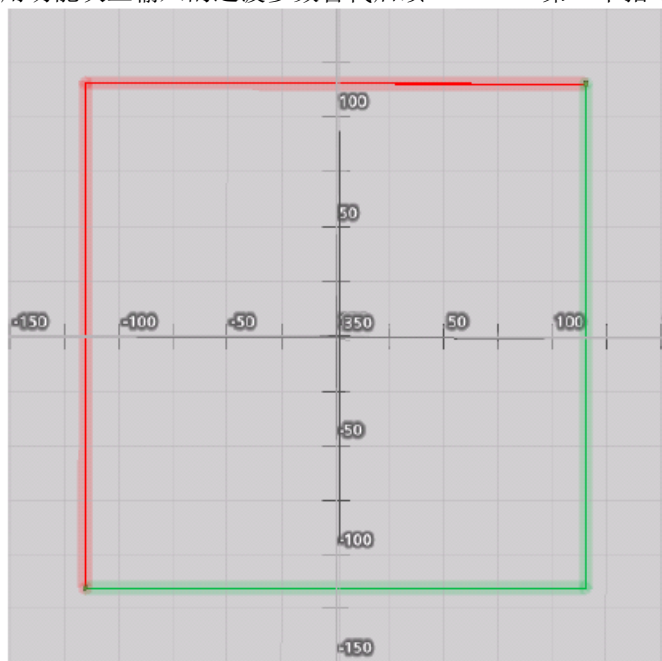
下图展示了上述情况的时序。



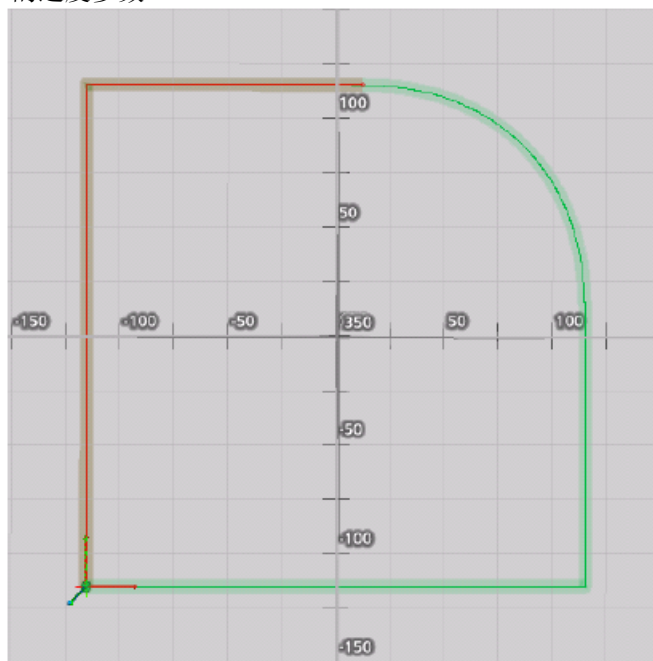
路径与路径之间的过渡

此功能类似圆角功能，可以让路径与路径之间的运动更加平滑。

需要的过渡方式既可以在后续 PathData 的第一个指令中组态也可以在功能块的 bufferMode 和 transitionParameter[1..5] 输入参数进行设置，然后结合功能块输入 pathTransition 选择新路径的初始过渡参数(0: 使用 PathData 中第一个 Command 自己规划的参数 1: 使用功能块 bufferMode 和 transitionParameter 参数)。将 pathTransition 设置成 1 即选择用功能块上输入的过渡参数替代后续 PathData 第一个指令中的过渡参数。



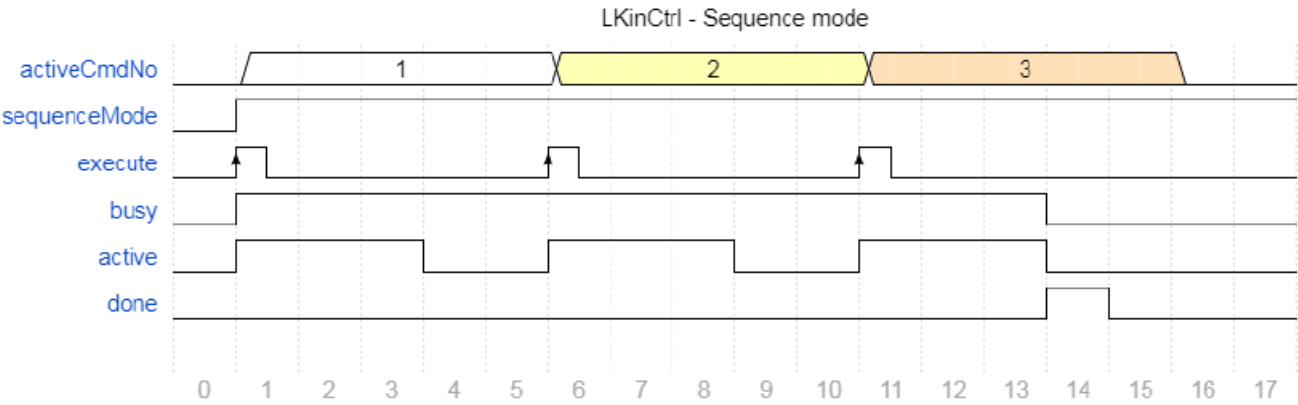
两个路径运动之间没有过渡衔接



两个路径运动之间有过渡衔接

• 单步模式

要想功能块运行在单步模式，sequenceMode 输入必须设置为 1。
在单步模式下，功能块会单独地执行每一条指令。execute 输入每一次上升沿只触发 PathData 中的一条指令。下图演示了单步模式的使用。



当功能块运行在单步模式时，PathData 中的每一条指令都会单独执行。因此，路径过渡无效，指令之间没有过渡。因为缺少过渡段，最后得出的路径轨迹和自动模式下运行结果可能不一样。指令中的每一个目标位置都会严格执行并抵达。

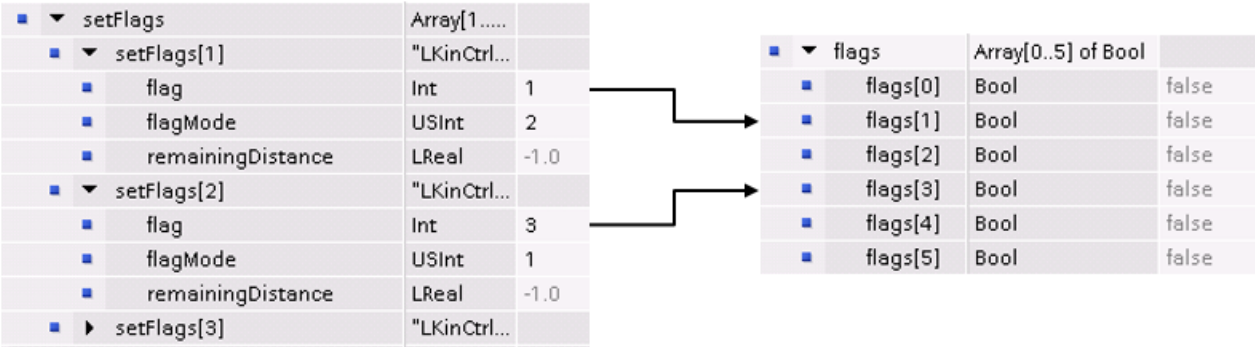
9.4 Flags

9.4.1 工作原理

Flags 为 PathData 中激活的指令控制执行器提供了可能。我们可以在 PathData 数据结构中配置 flags。PathData 中的每一条指令都可以单独配置 flags。因此，LKInCtrl_MC_MovePath 接口上布尔量 flags 和 LReal 的 valueFlags 输入输出信号可以根据运动机构当前激活的指令中的 setflags 和 valueFlags 来设定。通过选择特定的 flag 模式，可以指定置位和复位 flags 的时效。同样也可以编程实现路径运动过程中等待外部对 flag 置位的确认。以下将仅以 setFlags 举例进行解释，因为 setFlags 和 valueFlags 基本原理相同。

• 组态 Flags

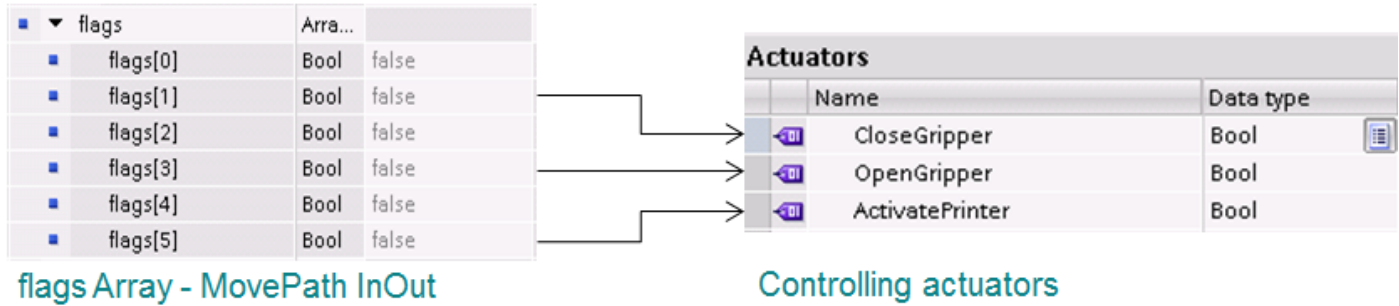
PathData 中的 setFlags 定义了 MovePath 接口上要被置位的 flags



PathData 中的每一条指令中都包含上图左侧展示的 setFlags 数据结构。setFlags 数组中的 flag 指向 LKInCtrl_MC_MovePath 功能块输入输出参数 flags 的数组索引。flags 数组对应索引的数组元素会根据组态的 flagMode 被复位或置位。
setFlags 数组中的 flag 默认值都是-1。该默认值不会触发 flags 的任何改变，也不会复位任何 flags。
同样默认情况下，在每一条指令中可以定义两种类型的 flags 数量是 3 个。也就是说当一条指令运行时，每种类型可以同时设置 3 个 flags。根据不同应用场景，一条指令中两种类型的 flags 数量可以通过库常量 LKINCTRL_NO_OF_CMD_SETFLAGS 和 LKINCTRL_NO_OF_CMD_VALUEFLAGS 进行调整。
如果 LKInCtrl_MC_MovePath 的输入输出参数 flags 数组长度需要大于默认值 6，可以通过同样的方式调整库常量 LKINCTRL_NO_OF_LAST_FLAG 来增加。

• 控制

flags 可以链接到输出用于执行器控制。

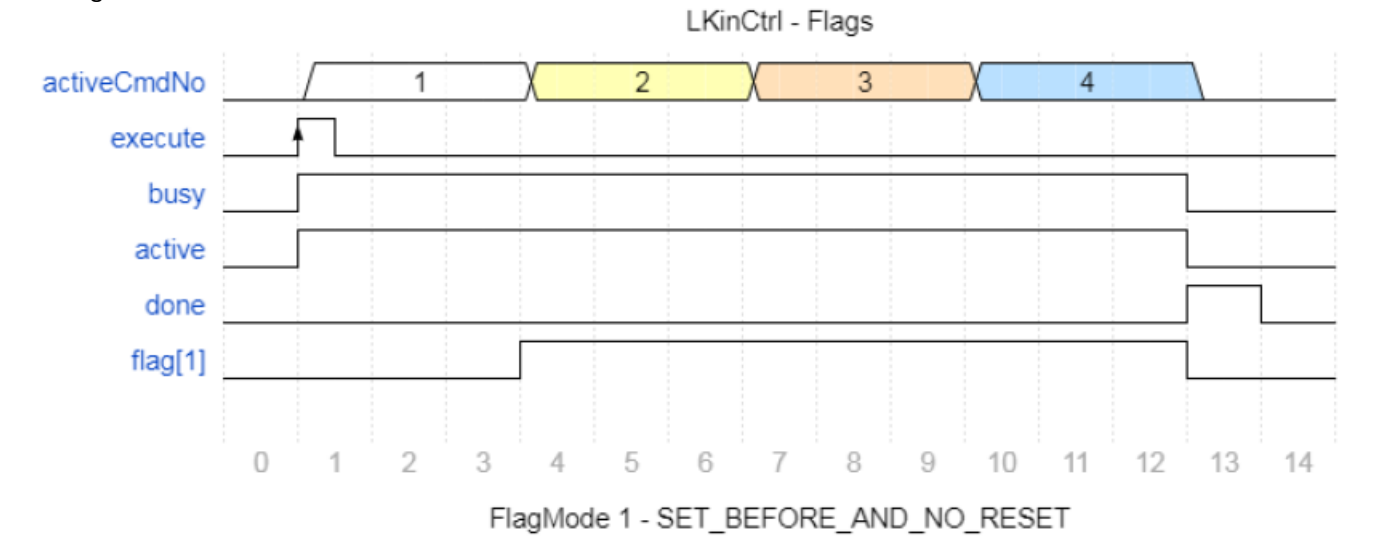


9.4.2 Flag 模式

Flag 模式	Flag 功能
0	Flag deactivated
1	SET_BEFORE_AND_NO_RESET
2	SET_BEFORE_AND_RESET_AFTER
3	SET_BEFORE_AND_RESET_AFTER_ONE_CYCLE
5	SET_BEFORE_AND_NO_RESET_AND_WAIT_FOR_ACKNOWLEDGE
10	SET_IN_REMAINING_DISTANCE_TO_TARGET
11	SET_AFTER_AND_NO_RESET
13	SET_AFTER_AND_RESET_AFTER_ONE_CYCLE
15	SET_AFTER_AND_NO_RESET_AND_WAIT_FOR_ACKNOWLEDGE

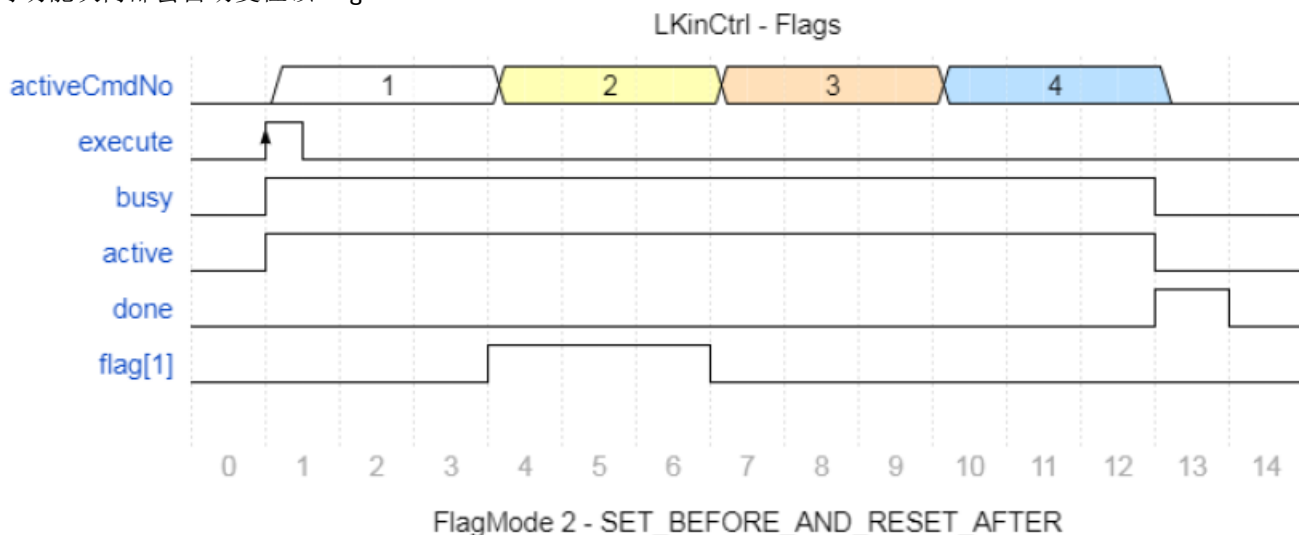
Flag 模式 1: SET_BEFORE_AND_NO_RESET

Flag 模式 1 下，PathData 中定义该 flag 的指令激活的瞬间立即就会置位该 flag。在路径运动完成之前功能块内部不会复位该 flag，除非期间它被用户程序在外部复位。



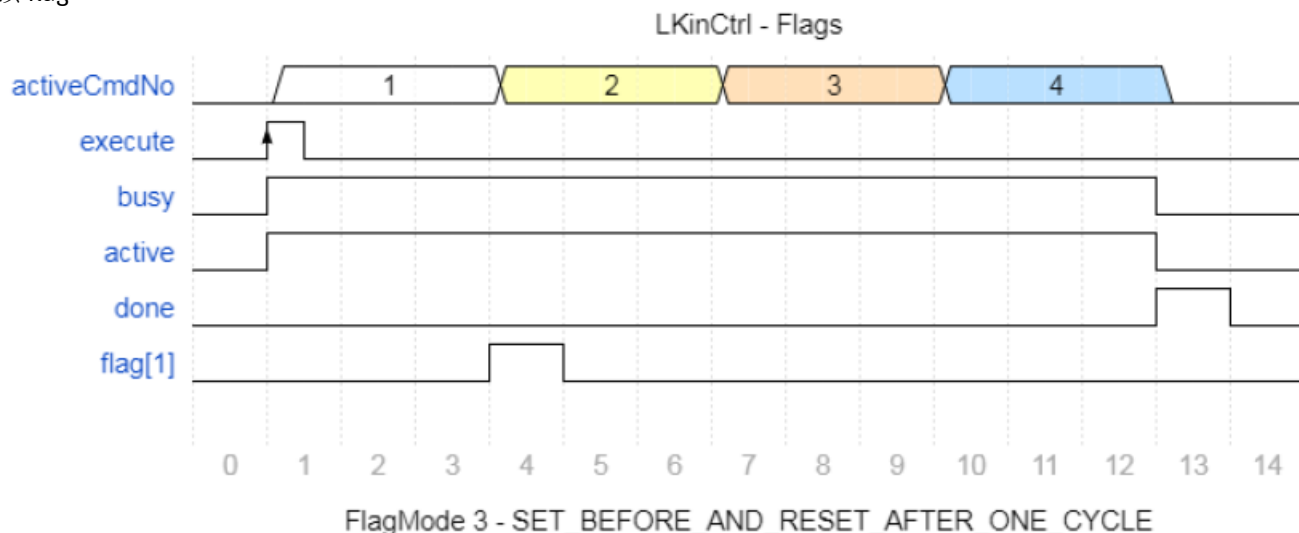
Flag 模式 2: SET_BEFORE_AND_RESET_AFTER

Flag 模式 2 下，PathData 中定义该 flag 的指令激活的瞬间立即就会置位该 flag。在这一条指令完成之后和下一条指令开始时功能块内部会自动复位该 flag



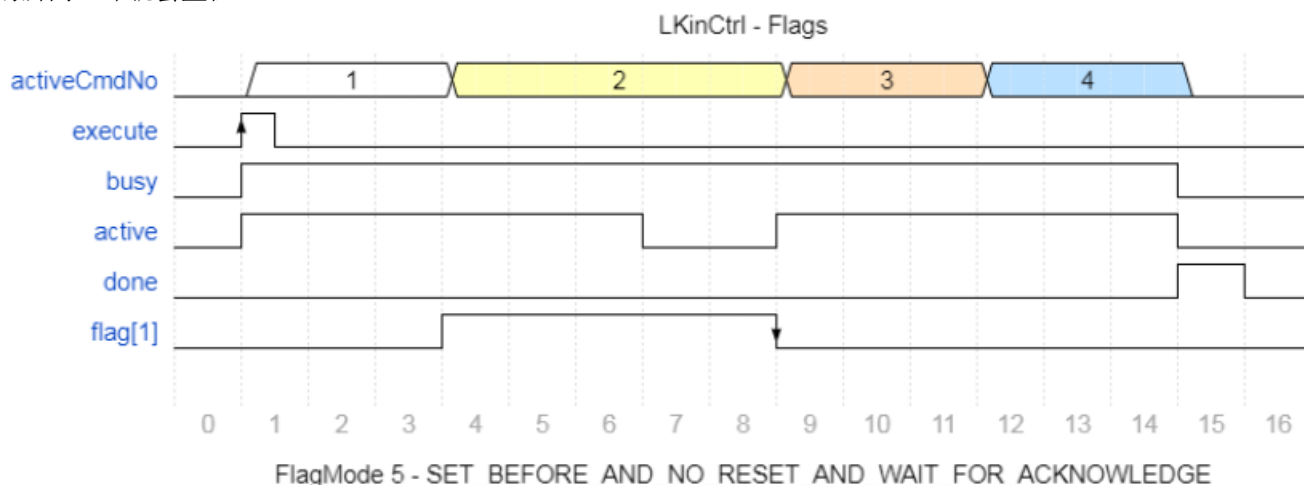
Flag 模式 3: SET_BEFORE_AND_RESET_AFTER_ONE_CYCLE

Flag 模式 3 下，PathData 中定义该 flag 的指令激活的瞬间立即就会置位该 flag。在一个扫描周期后功能块内部会自动复位该 flag



Flag 模式 5: SET_BEFORE_AND_NO_RESET_AND_WAIT_FOR_ACKNOWLEDGE

Flag 模式 5 下，PathData 中定义该 flag 的指令激活的瞬间立即就会置位该 flag。功能块内部不会自动复位该 flag，除非它在外部被用户程序复位。后面的路径执行在完成置位该 flag 以后会进入等待。一旦该 flag 被复位（确认）后，路径运动就会继续。该模式可以用于创建需要外部确认的等待点。该模式类似于 G 代码中的 M 点，只不过它是在该条指令激活的瞬间立即就会置位。



valueFlags 的确认只需要将对应的 statFlags[x]复位为 False 即可。

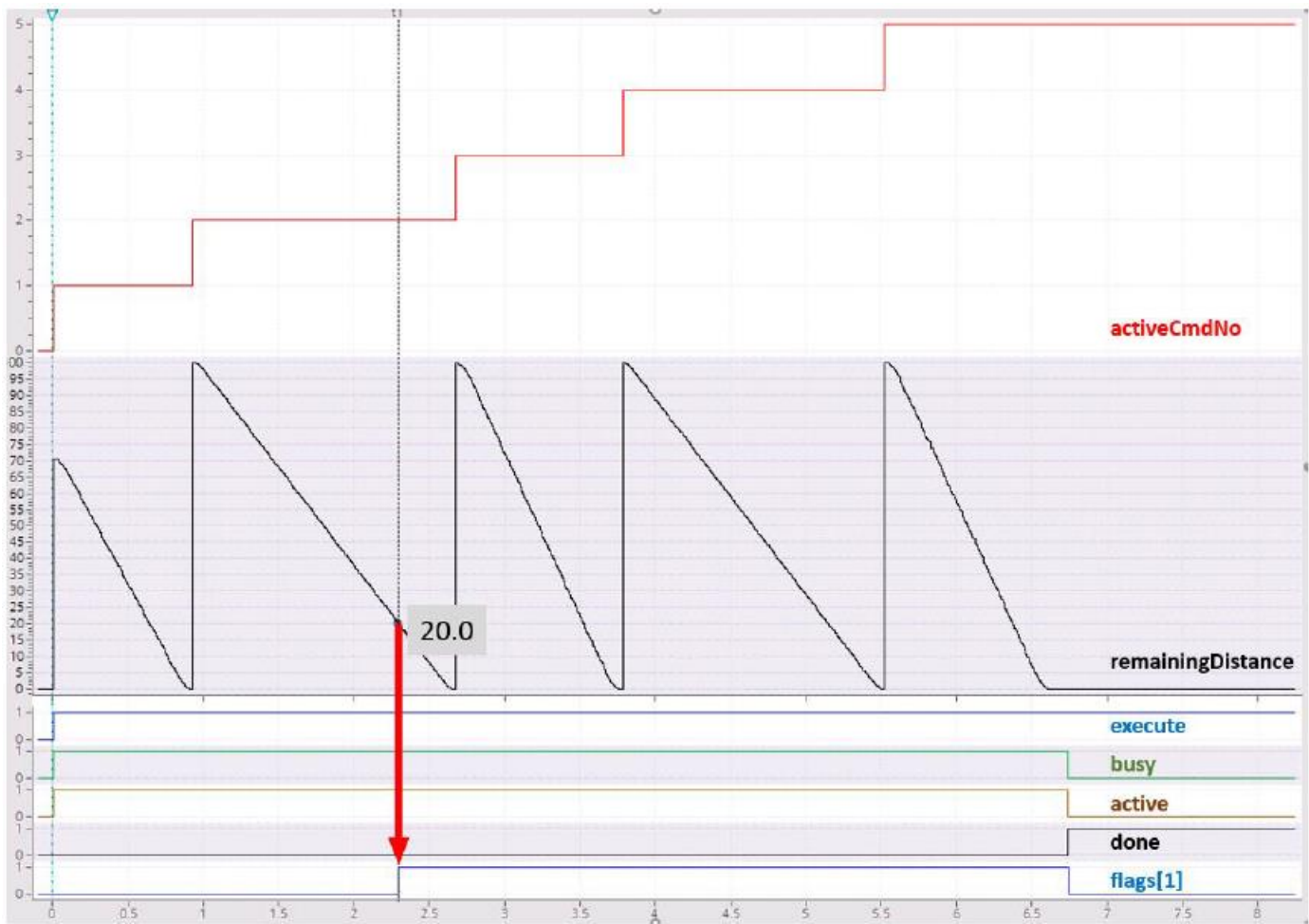
Flag 模式 10: SET_IN_REMAINING_DISTANCE_TO_TARGET

Flag 模式 10 下，可以在距离指令目标位置的一定距离处置位一个 flag。在路径运动完成之前功能块内部不会复位该 flag，除非期间它被用户程序在外部复位。该模式类似于 G 代码中的 H 代码。

例如：

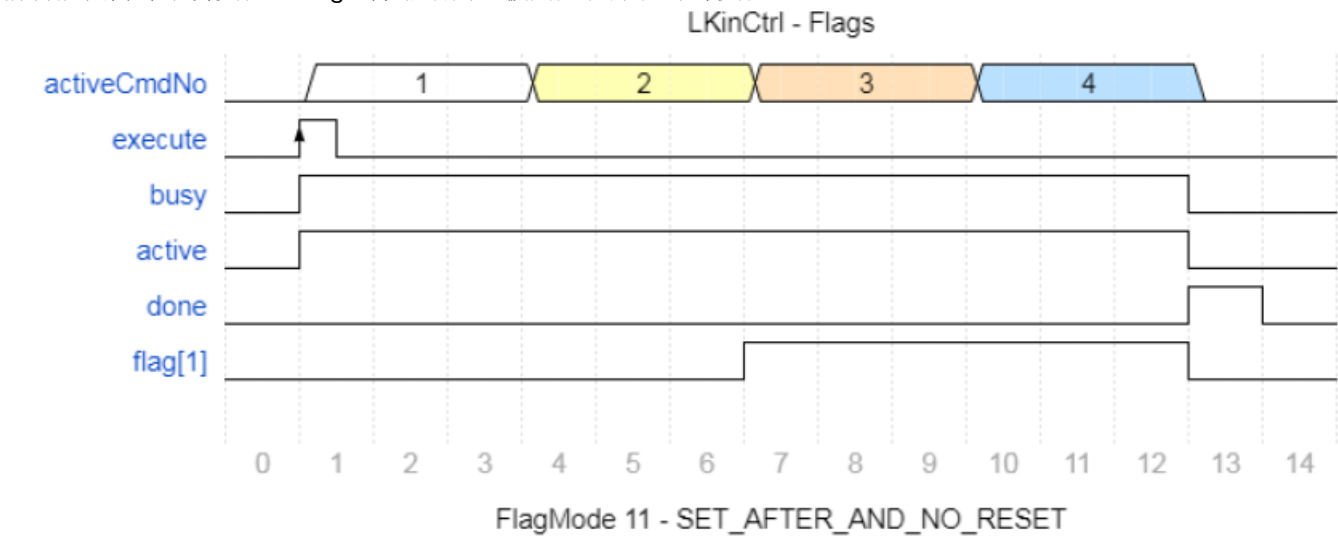
path1.Geometry.commands[2].setFlags[1].flag	DEC+/-	1
path1.Geometry.commands[2].setFlags[1].flagMode	DEC	10
path1.Geometry.commands[2].setFlags[1].remainingDistance	Floating-point nu...	20.0

时序如下：



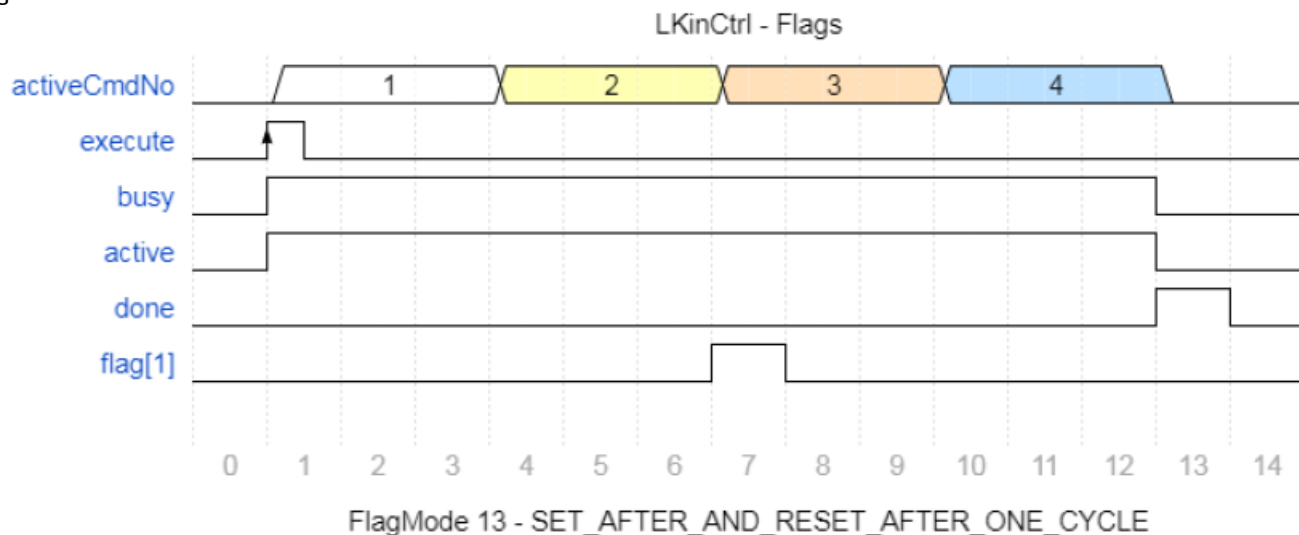
Flag 模式 11: SET_AFTER_AND_NO_RESET

Flag 模式 11 下，PathData 中定义该 flag 的指令完成后就会置位该 flag。随后下一条指令开始执行。在路径运动完成之前功能块内部不会复位该 flag，除非期间它被用户程序在外部复位。



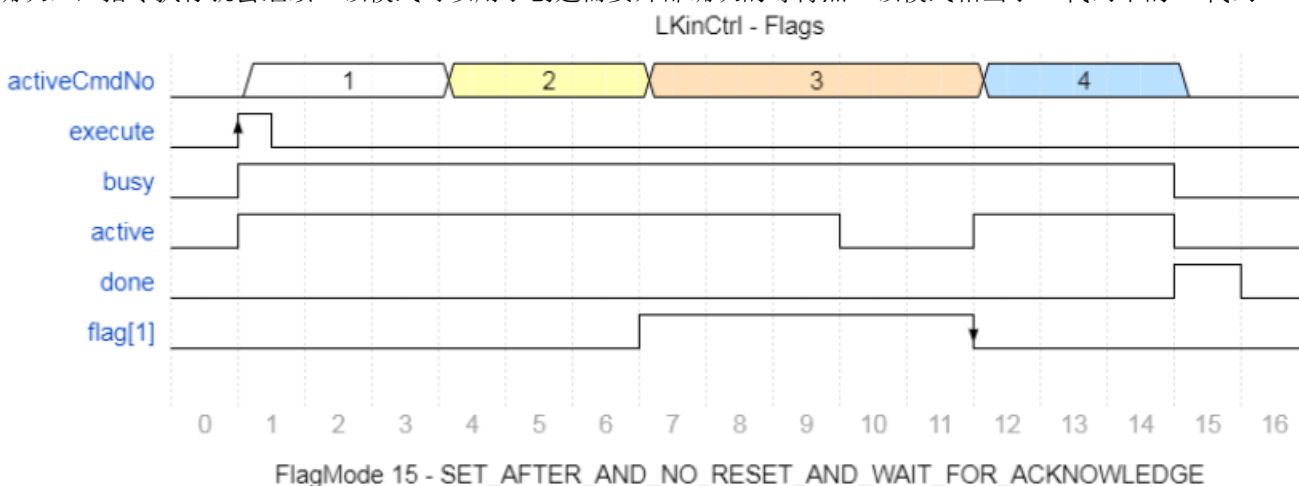
Flag mode 13: SET_AFTER_AND_RESET_AFTER_ONE_CYCLE

Flag 模式 13 下，PathData 中定义该 flag 的指令完成后就会置位该 flag。在一个扫描周期后功能块内部自动复位该 flag。



Flag mode 15: SET_AFTER_AND_NO_RESET_AND_WAIT_FOR_ACKNOWLEDGE

Flag 模式 15 下，PathData 中定义该 flag 的指令完成后就会立即置位该 flag。功能块内部不会自动复位该 flag，除非它在外用户程序中被复位。置位该 flag 的指令的下一条指令执行完成以后后续的指令会进入等待。一旦该 flag 被复位（确认），指令执行就会继续。该模式可以用于创建需要外部确认的等待点。该模式相当于 G 代码中的 M 代码。



valueFlags 的确认只需要将对应的 statFlags[x]复位为 False 即可。

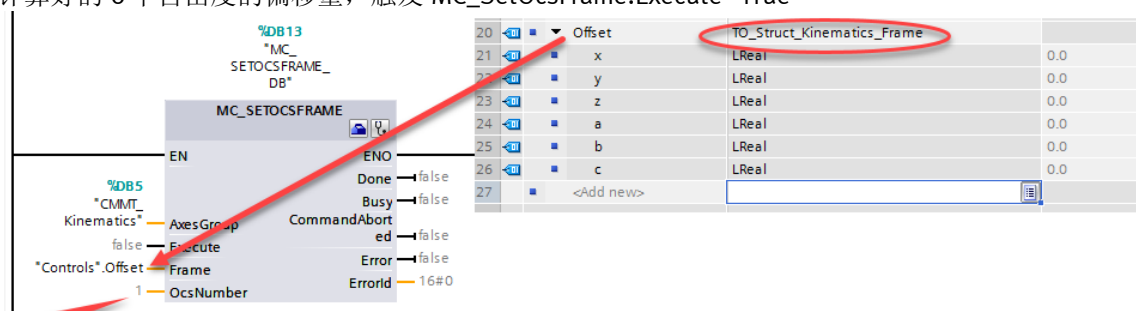
9.5 关于对象（工件）坐标系（OCS）和运动机构坐标系（KCS）偏移补偿

在现实的应用中，由于设备的搬运，震动等等原因，造成原本工件（轨迹）坐标在运动系统坐标系中出现了偏差。此时，我们可以通过 [6.3.2.6 章节](#) 里介绍的直接在工艺对象组态里修改这个偏移值，但是这样有个问题，就是无法实时修改，每次改完都要重新下载程序。

此时，可以通过功能块 MC_SetOcsFrame 来重新定义对象坐标系，来修改 6 个自由度的偏移量。

举例如下：

首先写入事先自行计算好的 6 个自由度的偏移量，触发 MC_SetOcsFrame.Execute=True



1: 表示修改的是OCS1

然后将轨迹中所有 G 代码的坐标系转换为 OCS1.



将轨迹中每一条G代码中的坐标系切换到OCS1

最后，执行 G 代码 LKinCtrl_MC_MovePath. Execute.

9.6 等待指令

在 PathData 中可以通过特定指令类型编写一个等待指令。在 PathData 的指令中组态 cmdType = 100 就激活了上一条指令完成之后，下一条指令开始之前的一条等待指令，等待的时长在 cmdparameters.pathDynamics.velocity 参数中配置（单位：毫秒）

如下图所示，在第 8 条命令中添加 2000ms 的等待。这意味着，在第 7 条 G 代码执行完成后，会等待 2S，才会执行第 9 条 G 代码。

test2 (snapshot created: 4/15/2020 9:49:28 AM)			
	Name	Data type	Start value
1	Static		
2	Geometry	"LKInCtrl_typePath...	
3	pathDataName	String["LKINCTRL_L...	"
4	pointers	"LKInCtrl_typeAdvP...	
5	commands	Array[1.."LKINCTRL...	
6	commands[1]	"LKInCtrl_typePath...	
7	commands[2]	"LKInCtrl_typePath...	
8	commands[3]	"LKInCtrl_typePath...	
9	commands[4]	"LKInCtrl_typePath...	
10	commands[5]	"LKInCtrl_typePath...	
11	commands[6]	"LKInCtrl_typePath...	
12	commands[7]	"LKInCtrl_typePath...	
13	commands[8]	"LKInCtrl_typePath...	
14	cmdType	Int	100
15	cmdActivated	Bool	TRUE
16	cmdName	String["LKINCTRL_L...	"
17	cmdCoordinates	"LKInCtrl_typePoint...	
18	cmdParameters	"LKInCtrl_typePoint...	
19	pathDynamics	"LKInCtrl_typePath...	
20	velocity	LReal	2000.0
21	acceleration	LReal	-1.0
22	deceleration	LReal	-1.0
23	jerk	LReal	-1.0

9.7 FlagOnly 指令

不定义一个运动指令同样可以使用 12.4 章节中描述的 flag 功能（类似于某行 G 代码只有 M1 这一个指令，无运动指令），这个时候需要将指令类型定义为 FlagOnly。

如下图所示，要使用一个 FlagOnly 指令，需要在 PathData 指令列表中将 cmdType 设置为 0。

commands[10]	"LKInCtrl_typePathDataElement"	
cmdType	Int	0
cmdActivated	Bool	TRUE
cmdName	String["LKINCTRL_LENGTH_OF_CMD...	"
cmdCoordinates	"LKInCtrl_typePointCoordinates"	
cmdParameters	"LKInCtrl_typePointParameter"	
setFlags	Array[1.."LKINCTRL_NO_OF_CMD_S...	
setFlags[1]	"LKInCtrl_typeSetFlagElement_Bool"	
flag	Int	1
flagMode	USInt	5
remainingDistance	LReal	-1.0
setFlags[2]	"LKInCtrl_typeSetFlagElement_Bool"	
setFlags[3]	"LKInCtrl_typeSetFlagElement_Bool"	
valueFlags	Array[1.."LKINCTRL_NO_OF_CMD_V...	
valueFlags[1]	"LKInCtrl_typeSetFlagElement_LRE...	
flag	Int	1
value	LReal	12.5
flagMode	USInt	1
remainingDistance	LReal	-1.0
valueFlags[2]	"LKInCtrl_typeSetFlagElement_LRE...	

FlagOnly 指令也支持所有的 flag 模式，虽然很多模式工作起来很相似，因为不需要等待任何运动完成。需要外部复位（确认）的模式会中断后续指令执行。因此，可以通过这种方式编写一些间歇点。

9.8 路径示教

当需要将运动机构当前位置坐标存储到一个路径中，库中提供了几种不同的能够存储点的数据类型。

示教到 PathData 数据结构

使用工程系统示教路径点以后，坐标值可以直接存储到 PathData 中。过后，可以在 PathData 中手动设置 cmdType 和其他指令参数来创建一个完整的路径信息。

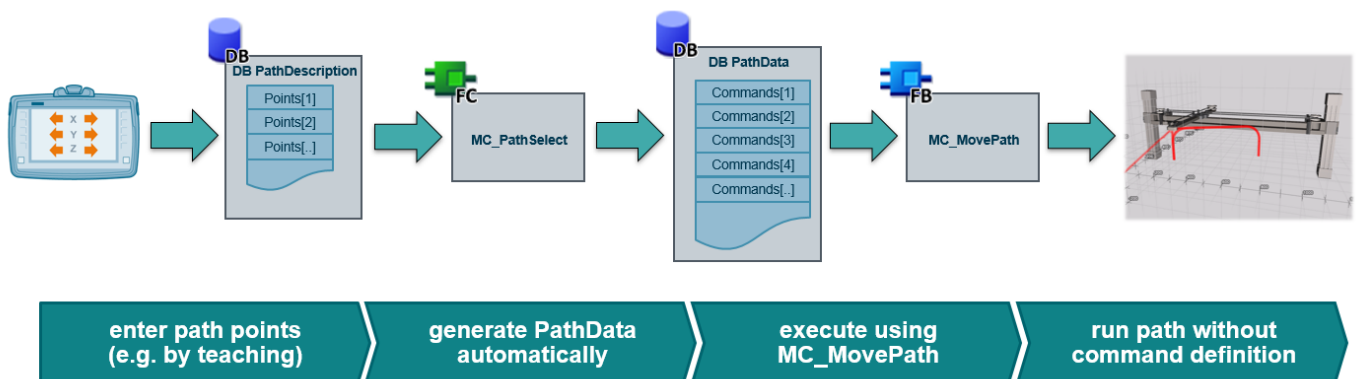
示教到 PathDescription 数据结构

PathDescription 数据结构提供了包括坐标，过渡信息和 flags 在内的一个点信息列表。这个列表可以用作用户示教点的存储位置。下图是在数据块中创建的 PathDescription 数据结构。

PathDescription				
	Name	Data type	Start value	Comment
1	Static			
2	pointTable	"LKinCtrl_typePath..."		
3	pathDescriptionName	String[10]	'teaching'	
4	points	Array[1.."LKINCTRL_..."		
5	points[1]	"LKinCtrl_typePath..."		
6	pointName	String["LKINCTRL_L..."	'pick'	name of point
7	targetPosition	"LKinCtrl_typeCart..."		
8	x	LReal	50.0	X
9	y	LReal	0.0	Y
10	z	LReal	10.0	Z
11	a	LReal	0.0	A
12	coordSystem	DInt	0	0: WCS; 1: OCS[1]; 2: OCS[2]; 3: OCS[3]
13	blendingDistance	LReal	0.0	blending distance ->0.0: configured value user
14	transitionParameters	Array[1..5] of LReal		blending parameters
15	flags	Array[1.."LKINCTRL_..."		flags to control actuators
16	flags[1]	"LKinCtrl_typeSetFl..."		flags to control actuators
17	flag	Int	2	flag to be set
18	flagMode	USInt	5	(see LKinCtrl_PLC_Tags) 0: deactivated; 1: SET_
19	remainingDistance	LReal	-1.0	>0.0: remainingDistance
20	flags[2]	"LKinCtrl_typeSetFl..."		flags to control actuators
21	flags[3]	"LKinCtrl_typeSetFl..."		flags to control actuators
22	points[2]	"LKinCtrl_typePath..."		
23	points[3]	"LKinCtrl_typePath..."		
24	points[4]	"LKinCtrl_typePath..."		
25	points[5]	"LKinCtrl_typePath..."		

在将点信息填充到 PathDescription 以后，可以通过 LKinCtrl_MC_PathSelect 函数从 PathDescription 生成一个 PathData。因此，可以不通过工程系统在用户程序里生成一个路径，如下图流程。

使用 PathDescription 和 MC_PathSelect 生成路径的简易流程



点动运动机构到指定位置来完成示教的操作可以使用运动机构手动控制标准应用（LKinMCtrl），下载链接：

<https://support.industry.siemens.com/cs/ww/de/view/109755892>

10 附录 A 错误处理

• 主功能块错误

LKinCtrl 块的错误在布尔量接口 `error` 上输出。`diagnostics.errorID` 输出还包含并存储了详细的错误信息直到被复位。`status` 输出包含了功能块调用循环中的状态和错误信息。下表列出了 `MC_MovePath` 的一些错误常量。

LKinCtrl_MC_MovePath 错误常量

名称	数据类型	值	注释
ERR_UNDEFINED_STATE	Word	16#8600	状态机未定义的状态
ERR_EXECUTE_KINMOTIONCMD	Word	16#8201	LKinCtrl_ExecuteKinMotionCmd 执行出错
ERR_WRONG_SEQUENCEMODE	Word	16#8203	sequenceMode 输入参数无效
ERR_PATHDATA_PARAMETERS	Word	16#8204	运动指令参数组态错误
ERR_OFFSET_CALCULATION	Word	16#8205	工具补偿计算错误
ERR_ERROR_STILL_PENDING	Word	16#8207	前道错误尚未被复位 (复位 MovePath 以确认)

LKinCtrl_typeMovePathDiagnostics

LKinCtrl_typeMovePathDiagnostics 参数

名称	类型	值	注释
errorID	Word	16#0	功能块错误ID
pathDataName	String[4]	"	发生错误的PathData名称
cmdNumber	DInt	0	PathData中发生错误的指令编号
stateMovePath	DInt	0	功能块状态机状态
statusSubFunction	Word	16#0	LKinCtrl_MC_ExecuteKinMotionCmd 状态
motionFBNumber	DInt	0	发生错误的内部功能块号
motionFBStatus	DWord	16#0	发生错误的内部功能块状态
kinematicsStatusWord	DWord	16#0	运动机构工艺对象状态字
kinematicsErrorWord	DWord	16#0	运动机构工艺对象错误字

功能块运行的每个周期，包含功能块错误和一般信息的状态参数都会刷新。其他仅作为错误诊断的参数只有在有错误发生时才刷新。

举个例子，当功能块在执行一个路径运动时发生错误，诊断数据结构就会提供包括发生错误的路径名称和当前指令编号，还有内部运动控制功能块实例的错误信息和说明以及运动机构工艺对象的状态字和错误字。这些能够帮助分析和找到错误。

`motionFBstatus` 参数提供了运动机构系统运动控制指令的 `errorID` 输出。当有错误发生时，也可以通过 TIA 的帮助系统查询错误代码。

如果 `PathData` 中输入了无效的参数（系统 MC 指令不支持的参数），`motionFBstatus`, `pathDataName` 和 `cmdNumber` 诊断参数就会列出配置不正确的地方。功能块的状态也会提供相应信息。

下面列出了主功能块的子块的错误常量并在注释中进行了注解。

LKinCtrl_ExecuteKinMotionCmd 错误常量

名称	数据类型	值	注释
ERR_UNDEFINED_STATE	Word	16#8600	未定义的功能块状态
ERR_GROUPINTERRUPT_FAILED	Word	16#8242	MC_GroupInterrupt 出错
ERR_GROUPCONTINUE_FAILED	Word	16#8243	MC_GroupContinue 出错
ERR_GROUPSTOP_FAILED	Word	16#8244	MC_GroupStop 出错
ERR_TO_KINEMATIC_ERROR	Word	16#8245	TO_Kinematic 出错
ERR_LINEAR_CMD	Word	16#8246	MC_MoveLinear 出错
ERR_CIRCULAR_CMD	Word	16#8247	MC_MoveCircular 出错
ERR_CMD_PARAMETER	Word	16#8248	指令参数有错
ERR_CALC_CIRC_ARC	Word	16#8249	指令参数有错

LKinCtrl_PreBuffer 错误常量

名称	数据类型	值	注释
ERR_UNDEFINED_STATE	Word	16#8600	未定义的功能块状态
ERR_OFFSET_COMPENSATION	Word	16#8261	补偿参数计算错误
ERR_LASTPOS_CALCULATION	Word	16#8262	上一个目标位置计算出错
ERR_INVALID_PATHDATA_TYPE	Word	16#8263	不支持的pathdata类型

LKinCtrl_CheckPathDataCmd 的错误常量用于识别 `PathData` 组态的错误。如果 `PathData` 中输入无效，错误信息会传递到诊断数据中。如果 `PathData` 中有错误的参数 `diagnostics.motionFBstatus` 会输出一个下表中的值来识别这个无效参数。另外，`diagnostics.motionFBNumber` 还会包含 `PathData` 的错误指令编号。`diagnostics.statusSubFunction` 也会预先输出一个如下错误。

LKinCtrl_CheckPathDataCmd 错误常量

名称	数据类型	值	注释
ERR_UNDEFINED_STATE	Word	16#8600	未定义的功能块状态
ERR_PATHDATA_CMDTYPE	Word	16#80A0	cmdType错误
ERR_PATHDATA_COORDSYSTEM	Word	16#80B1	坐标系错误
ERR_PATHDATA_ORIENTATIONDIRECTION	Word	16#8007	旋转轴方向错误
ERR_PATHDATA_BUFFERMODE	Word	16#80B2	缓存模式错误
ERR_PATHDATA_DYNAMICADAPTION	Word	16#80B5	动态调整错误
ERR_PATHDATA_CIRCMODE	Word	16#80B6	圆弧插补模式错误
ERR_PATHDATA_PATHCHOICE	Word	16#80B9	路径选择错误
ERR_PATHDATA_CIRCLEPLANE	Word	16#80BA	圆弧平面错误
ERR_PATHDATA_RADIUS	Word	16#80BB	圆弧半径错误
ERR_PATHDATA_ARC	Word	16#80BC	圆弧弧度错误
ERR_PATHDATA_SETFLAGS_FLAG	Word	16#80D1	flag定义错误
ERR_PATHDATA_SETFLAGS_FLAGMODE	Word	16#80D2	flag模式定义错误
ERR_PATHDATA_OFFSET_PARA_NO	Word	16#80A1	补偿参数选择编号错误
ERR_PATHDATA_OFFSET_MAINPATHPLANE	Word	16#80A2	主路径平面错误

LKinCtrl_MC_PathSelect 错误常量

名称	数据类型	值	注释
ERR_UNDEFINED_STATE	Word	16#8600	未定义的功能块状态
ERR_WRONG_CS	Word	16#8280	坐标系定义错误
ERR_WRONG_FLAG	Word	16#8281	flag定义错误
ERR_WRONG_FLAGMODE	Word	16#8282	flagMode定义错误
ERR_PATHDATA_OVERFLOW	Word	16#8283	pathData 点数超限

• 辅助功能块错误

LKinCtrl_MC_GroupPower 错误常量

名称	数据类型	值	注释
ERR_UNDEFINED_STATE	Word	16#8600	未定义的状态机状态
ERR_AXIS_ERROR	Word	16#8370	轴当前有错误
ERR_KIN_ERROR	Word	16#8371	运动机构当前有错误
ERR_MC_POWER	Word	16#8372	MC_POWER执行出错

LKinCtrl_MC_GroupReset 错误常量

名称	数据类型	值	注释
ERR_UNDEFINED_STATE	Word	16#8600	未定义的状态机状态
ERR_AXIS_ERROR	Word	16#8360	轴当前有错误
ERR_KIN_ERROR	Word	16#8361	运动机构当前有错误
ERR_MC_RESET_AXES	Word	16#8362	轴在运行MC_RESET出错
ERR_MC_RESET_KIN	Word	16#8363	运动机构在运行MC_RESET出错

LKinCtrl_MC_GroupHome 错误常量

名称	数据类型	值	注释
ERR_UNDEFINED_STATE	Word	16#8600	未定义的状态机状态
ERR_AXIS_ERROR	Word	16#8350	轴当前有错误
ERR_MC_HOME	Word	16#8351	轴在执行MC_HOME出错

LKinCtrl_typeAuxFCDiagnostics 参数

名称	类型	值	注释
status	Word	DEF_VAL	功能块状态或者出错时的ID
subfunctionStatus	Word	DEF_VAL	调用的内部功能块，函数以及系统功函数状态
axisNumber	Int	DEF_VAL	受影响或者出错的轴号
errorDetail	UDInt	DEF_VAL	发生错误的工艺对象详细错误信息
stateNumber	DInt	DEF_VAL	发生错误时功能块状态机的状态号

- 轮廓补偿功能块错误

LKinCtrl_OffsetContour 错误常量

名称	数据类型	值	注释
ERR_COFF_INVALID_TOOL_LENGTH	Word	16#8301	工具长度无效
ERR_COFF_INVALID_TOOL_RADIUS	Word	16#8302	工具半径无效
ERR_COFF_INVALID_PATH_PLANE	Word	16#8303	路径平面无效
ERR_COFF_INVALID_COMPENSATION_DIRECTION	Word	16#8304	补偿方向无效
ERR_COFF_FUNCTION_NOT_SUPPORTED	Word	16#8305	不支持的功能
ERR_COFF_INVALID_COMMAND_TYPE	Word	16#8306	指令类型无效
ERR_COFF_INVALID_CIRCLE_MODE	Word	16#8307	圆弧插补模式无效
ERR_COFF_INVALID_PATH_PLANE_OF_CIRCULAR_CMD	Word	16#8308	圆弧插补指令中路径平面无效
ERR_COFF_INVALID_PATH_CHOICE_OF_CIRCULAR_CMD	Word	16#8309	圆弧插补指令中的路径选择无效
ERR_COFF_INTERNAL	Word	16#830F	内部错误
ERR_COFF_VECTOR_LENGTH_ZERO	Word	16#8310	矢量长度为0
ERR_COFF_INVALID_ACOS_ARGUMENT	Word	16#8311	ACOS函数参数无效
ERR_COFF_LINES_DO_NOT_INTERSECT	Word	16#8312	线条不相交
ERR_COFF_ZERO_VECTOR	Word	16#8313	0矢量
ERR_COFF_LINE_CIRCLE_DO_NOT_INTERSECT	Word	16#8315	直线和圆弧不相交
ERR_COFF_CIRCLES_DO_NOT_INTERSECT	Word	16#8318	圆弧之间不相交