

繁易自定义工艺库文件使用说明

部门：技术支持部门

版本：V1.0.0

日期：2024-11-04

作者：张金烨

保密等级：☐ 公开 ☒ 内部

摘 要：

本说明书主要介绍了繁易自定义工艺库中各个 FB 和 FC 的使用，库文件会随着 Package 的安装一并装入 Codesys 的 Library 中。所有功能块都只能下载到我司 PLC 中运行，或者在电脑中处于仿真状态也可以运行，下载到其他厂商的 PLC 中不运行。此外，工程中需要勾选允许使用 Unicode 标识符。

本文档使用硬件设备和软件工具

- 硬件：繁易 FL8、FL6 PLC
- 软件：Codesys V3.5 SP19

文档更新状态：

日期	版本	修订人	修订内容
2024/10/31	V1.0.0	张金烨	初次发版

免责声明：

我们已对本文档描述的内容做测试。但是差错在所难免，无法保证绝对正确并完全满足您的使用需求。本文档的内容可能随时更新，如有改动，恕不事先通知，也欢迎您提出改进建议。

上海繁易信息科技股份有限公司

繁易官方：<https://flexem.cn/>

繁易学院：<https://study.flexem.cn/>

技术咨询：4008-033-022 转 1

邮箱：Training@flexem.com

地址：上海市杨浦区国安路 386 号 INNO 创智 A 栋 9 楼

地址：苏州市虎丘区浒墅关镇青花路 26 号上市科创园二期 4 幢 3 楼

繁易自定义工艺库文件使用说明.....	1
IoDriveHSIO.....	4
01_PTO_PWM.....	4
PTO_Map_Virtual.....	4
FlexLibrary.....	8
01.FB.....	8
01_TEC_AxisModel.....	8
TEC_AxisModel 基础轴控功能块	8
FunNext 步序加一	15
02_插补	16
FB_ArrLineInterpolation 多轴直线插补	16
FB_CenterAngleCircleInterpolation 两轴圆弧插补（圆心圆弧）	21
FB_EndPosCircleInterpolation 两轴圆弧插补（圆心、结束位置、圈数）	25
FB_HeLixInterpolation 三轴螺旋插补.....	29
03_齿轮	33
FB_FollowGearInPos 特定位置齿轮耦合	33
04_凸轮	35
FB_FollowRotaryCutP 飞剪跟随.....	35
FB_FollowSyn 追剪跟随.....	38
05_TCP 和 UDP 通讯.....	40
TCPServer_N.....	40
TCP_Client_N.....	44
UDP_N	47
06_滤波	50
FB_KalmanFilter 卡尔曼滤波	50
10_其他	52
FB_LOG_DINT 数据变化记录	52
FB_SysTimeGet 时钟获取.....	53
TimePulse 可变占空比脉冲	55
11_光伏行业.....	56
FB_CycleCount 循环计数	56
YYX_Plane -XYR 坐标变换.....	57
丝印刮刀压力计算_FB	59
02.FC.....	60
01_ArrMaxMin 求数组最大最小值	60
ArrMax_DW/LR/LW/R/W.....	60
ArrMin_ DW/LR/LW/R/W	61
02_数据的高低位拆分和合并 Byte&Word.....	62
LHByte_To_Word.....	62
Word_To_LHByte.....	64
03_数据的高低位拆分和合并 RealWord	65
LHWord_To_Real.....	65

Real_To_LHWord.....	66
04_位移、速度、加速度计算类.....	68
FC_GatPeriodMinL.....	68
FC_GetAcc.....	69
FC_VtL.....	70
FC_两速度同步 V1 位移差.....	72
FC_求匀加速位移.....	73
FC_求匀速段位移.....	74
05_圆弧相关.....	75
FC_Circle_AngleGteEndXY	75
FC_Circle_GteR.....	77
FC_Circle_XyGteAngle.....	78
FC_w.....	80
10_其他.....	81
Mod_Real.....	81
FUN.....	82
数据类型转换—Arr.....	82
ArrBit_To_ArrWord.....	82
ArrBit_To_LWord.....	84
ArrBit_To_Word.....	86
ArrSH.....	87
ArrWord_To_ArrBit.....	89
ArrW_0.....	91
Word_To_ArrBit.....	92
数据类型转换—Str,ASCII.....	93
ArrByte_To_ASCIIStr.....	93
ASCIIStr_To_ArrByte.....	95
Byte_To_ASCIIStr.....	96
Word_To_HEXStr.....	98
FC_FlexemPLC.....	99

IoDriveHSIO

01 PTO_PWM

PTO_Map_Virtual

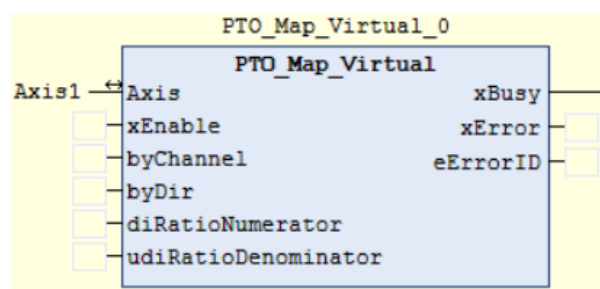
指令名称 PTO_Map_Virtual 脉冲轴实际接线和程序组态映射功能块 (IoDrvHSIO 库)

功能块说明:

本功能块适用于 FL8 用高速 PTO 带脉冲轴, 程序中用本功能块把接线和程序配置进行对应

1、指令格式

图形表现:



ST 表现:

```
PTO_Map_Virtual_0(
    Axis:= Axis1,
    xEnable:= ,
    byChannel:= ,
    byDir:= ,
    diRatioNumerator:= ,
    udiRatioDenominator:= ,
    xBusy=> ,
    xError=> ,
    eErrorID=> );
```

2、相关变量

变量名称	名称	数据类型	输入输出属性	描述
Axis	轴名称	AXIS_REF_VIRTUAL_SM3	输入输出变量	映射到虚拟轴 SM_Drive_Virtual
xEnable	功能块有效	BOOL	输入变量	电平有效
byChannel	脉冲通道	Byte	输入变量	实际接线规定哪个是脉冲通道
byDir	方向通道	Byte	输入变量	实际接线规定哪个是方向通道
diRatioNumerator	齿轮比分子	Dint	输入变量	电机转一圈, 对应工程中用户单位数值。例如电机直接接丝杠, 丝杠导程 10, 那么电机转一圈负载位移 10mm, 本参数填 10 即可
udiRatioDenominator	齿轮比分母	Dint	输入变量	电机转一圈需要多少的脉冲数。例 FV5 的 Pn0310=10000 电机转一圈, 本参数填 10000 即可
xBusy	功能块正在运行	BOOL	输出变量	功能块正在执行中

xError	功能块报错	BOOL	输出变量	功能块报错
ErrorID	报错信息	FL_Error	输出变量	报错代码（见下表）

FL_Error:

名称	类型	初始化	描述
ERR_OK	UINT	0	功能块正在执行中
HSIO_DO_CONFLICT	UINT		DO 功能配置冲突
PTO_CHANNEL_NOT_CONFIGURED	UINT		PTO 通道未配置
PTO_SET_CONFIGURATION_FAILED	UINT		PTO 配置失败
PTO_RUN_FAILED	UINT		PTO 运行失败

具体使用介绍:

1、目前 FL8 支持的 PTO 模式配置方法（任意输出点指的是 PLC 本体上的输出点）

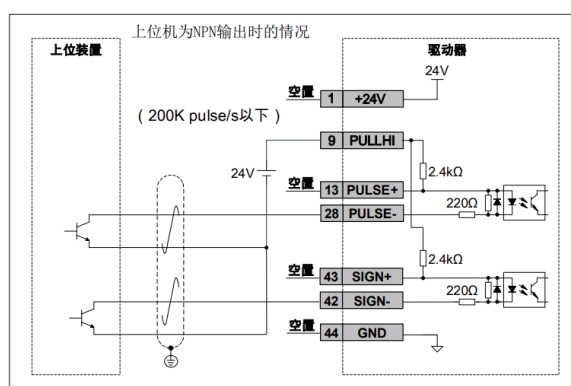
高速输出配置		
	输出地址	输出模式
		脉冲+方向
脉冲通道	QX0.0	脉冲（方向可指定任意输出点）
脉冲通道	QX0.1	脉冲（方向可指定任意输出点）
脉冲通道	QX0.2	脉冲（方向可指定任意输出点）
脉冲通道	QX0.3	脉冲（方向可指定任意输出点）
脉冲通道	QX0.4	脉冲（方向可指定任意输出点）
脉冲通道	QX0.5	脉冲（方向可指定任意输出点）

2、FL8 的 PTO 和 FV5 的典型接线和关键参数

FL8 接 FV5 驱动器: (参考伺服相关手册)

- (1) FV5: 9~~开关电源 24V+
- (2) FV5: 28~~FL8: Q0.0 脉冲输出
- (3) FV5: 42~~FL8: Q0.7 方向
- (4) FL8: Y 公共端~~开关电源 0V

● 使用外部 24V 电源和驱动器内置电阻时



驱动器参考参数设置类: (参考伺服手册)

- (1) 上电使能

☐	编号①	名称	设定值
☒	Pn0404	DI1端子功能选择	1 - 使能
☒	Pn0405	DI1端子逻辑选择	1 - 高电平

(2) 电机转一圈，需要 10000 个脉冲

☒	Pn0310	电机一圈的指令脉冲数	10000	0
☐	Pn0312	第1组电子齿数分子		1
☐	Pn0314	第1组电子齿数分母		1

(3) 脉冲来源与滤波

☒	Pn0304	脉冲滤波时间	10
☒	Pn0305	脉冲来源选择	1 - 低速脉冲口

备注：

- (1) 建议使用外部开关电源供电，比内部 24V 供电抗干扰能力高
- (2) 调高脉冲滤波时间，防止驱动器 U0005 计数偏差大
- (3) FV5 和 EAS 的引脚定义不一样，请注意区分。此外 FV5 多一个脉冲来源选择，需要设置为低速。(200kHz 以下选低速，4M 运动控制板卡选高速)
- (4) 该功能块收录在 IoDrvHSIO 库，随着 Package 安装一起打包。装 Package 的时候会自动安装。

如上，实际接线为 Q0.0 脉冲输出、Q0.7 方向。电机转一圈需要 10000 个脉冲。那么 Codesys 工程中配置如下：

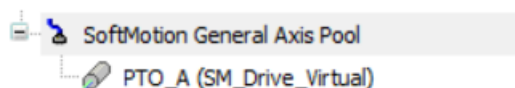
1、启用 Y0 的 HSP 模式

Flexem.HSIOIEC对象	参数	类型	值	默认值	单元
HSIO参数	Input Filtering				
	HSC Mode				
	PWM Enable	BYTE	0		
	HSP Enable	BYTE	1		
	HSP Enable [Y0]	BOOL	TRUE	FALSE	
	HSP Enable [Y1]	BOOL	FALSE	FALSE	
	HSP Enable [Y2]	BOOL	FALSE	FALSE	
	HSP Enable [Y3]	BOOL	FALSE	FALSE	
	HSP Enable [Y4]	BOOL	FALSE	FALSE	
	HSP Enable [Y5]	BOOL	FALSE	FALSE	

2、新建虚拟轴 A；SoftMotion General Axis Pool—右击---添加设备---虚拟驱动器—SM_Drive_Virtual



3、添加完成；程序中轴名称为 PTO_A



4、程序中声明与调用如下（参考功能块引脚说明）

```

1  PROGRAM POU_5
2  VAR
3
4      Pto_A_CTRL:PTO_Map_Virtual;
5
6  END_VAR
7

```

```

1  Pto_A_CTRL(
2      Axis:= PTO_A,
3      xEnable:= ,
4      byChannel:= 0,
5      byDir:= 7,
6      diRatioNumerator:= 10,
7      udiRatioDenominator:= 10000,
8      xBusy=> ,
9      xError=> ,
10     eErrorID=> );

```

5、注意，脉冲轴无法调用 402 轴的回原点模式。那么需要我们自行封装脉冲轴的回原模式，可以用 Jog 去写流程。这块 FB 还在完善中。其他运动指令和总线 402 轴的 MC 指令块相同。

FlexLibrary

01.FB

01_TEC_AxisModel

TEC_AxisModel 基础轴控功能块

指令名称 TEC_AxisModel 轴基础运动功能块

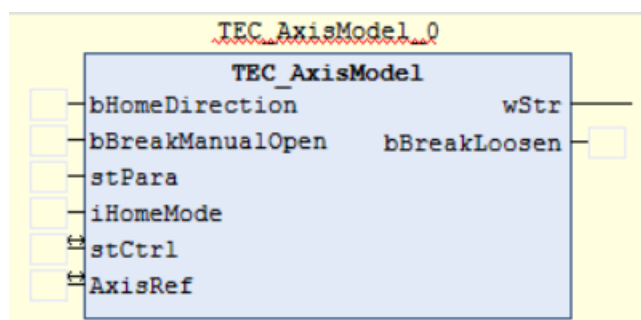
功能块说明:

本功能块将常用的轴控功能都集合在一个 FB 中，通过结构体变量对轴进行控制。本功能块包含了轴的使能、停止、复位、总线复位、JOG 运行、回原点、绝对定位，相对定位、设定轴当前位置等功能。

1、指令格式

图形表现:

ST 表现:



```
Tec_Axis1(
    bHomeDirection:= ,
    bBreakManualOpen:= ,
    stPara:= ,
    iHomeMode:= ,
    stCtrl:= ,
    AxisRef:= ,
    wStr=> ,
    bBreakLoosen=> );
```

2、相关变量

变量名	名称	数据类型	输入输出属性	描述
bHomeDirection	回原点方向	BOOL	输入变量	=True 表示先正向运行。仅在 iHomeMode=0/1/3 的时候有效
bBreakManualOpen	抱闸手动解除	BOOL	输入变量	抱闸手动打开
stPara	轴参数类	ST_AxisPara	输入变量	轴参数类变量。详见以下截图
iHomeMode	回原点模式	INT	输入变量	轴回原点模式设置。
stCtrl	轴控制类参数	ST_AxisCtrl	输入输出变量	轴控制类变量。详见以下截图
AxisREF	具体控制的轴	Axis_REF_SM3	输入输出变量	具体的 402 轴名称
wStr	运行信息	STRING	输出变量	运行状态显示
bBreakLoosen	松开抱闸	BOOL	输出变量	强制输出抱闸松开

StPara:

```

TYPE ST_AxisPara :
STRUCT
    rJogVel      :REAL:=100; //JOG速度
    rAcc         :REAL:=10000; //加速度
    rDec         :REAL:=10000; //减速度
    rJerk        :REAL:=10000; //减减速
    rStopDec     :REAL:=10000; //Stop的减速度
    rHomeVel     :REAL:=360; //回原速度
    rMaxLimit    :REAL; //最大软限位
    rMinLimit    :REAL; //最小软限位
    rHomeOffset  :REAL; //回原点偏移
END_STRUCT
END_TYPE

```

stCtrl:

```

TYPE st_AxisCtrl :
STRUCT
    bPower      :BOOL; //使能接口
    iStop       :INT; //停止
    bJogForward :BOOL; //手动正转
    bJogBackwards :BOOL; //手动反转
    iReset      :INT; //复位
    iHome       :INT; //回原点
    iSetPosition :INT; //修改轴当前位置
    iMoveAbs    :INT; //绝对定位(**)
    iMoveRel    :INT; //相对定位
    bHomeSensor :BOOL; //回原点传感器信号
    rPosition   :REAL; //定位位置值接口
    rVelocity   :REAL; //定位速度值接口
    bError      :BOOL; //报警传出接口
    udErrorId   :UDINT; //报警代码
    rActPos     :REAL; //当前位置
    rActVel     :REAL; //当前速度
    iAxisState  :INT; //轴状态
    iPauseAction :INT; //暂停轴绝对定位    88 暂停完成    -1 解除暂停    -88 解除完成
    bHomeFlag   :BOOL; //回原点标志位
END_STRUCT
END_TYPE

```

3、具体使用

(1) 声明与实例化

The screenshot shows the Siemens STEP 7 software interface. On the left, the 'Object Tree' (Object Explorer) displays the project structure. The 'P2' program is selected, and the 'EtherCAT_Task_A' is expanded, showing the 'P2' sub-task. A red box highlights the 'P2' sub-task, with a callout '4 在EtherCat Task下调用' (Call under EtherCat Task). Below it, the 'FV5_N (FV5_N_V1.3)' is expanded, showing the 'AxisMain (Axis)' object. A red box highlights this object, with a callout '1 接了一个从站轴是AxisMain' (Connected a slave axis is AxisMain). On the right, the 'Variable Declaration' (VAR) section shows the declaration of the 'Tec_Axis1' function block. A red box highlights the 'stCtrl_AxisMain' variable, with a callout '2 声明必要参数' (Declare necessary parameters). Below the declaration, the 'Tec_Axis1' function block is instantiated. A red box highlights the 'AxisRef' parameter, with a callout '3 实例化控制FB' (Instantiate control FB). The 'AxisRef' parameter is set to 'AxisMain', which is the slave axis connected to the EtherCAT task.

(2) 详细功能介绍与使用

功能 1: 使能

stCtrl.bPower:=TRUE 即可，若使能失败 stCtrl.bError=TRUE、并且输出 ErrorID。可以取 stCtrl.iAxisState=3，作为使能成功，并且没有报错的标志位。

表达式	类型	值	准备值
stCtrl.bPower	BOOL	TRUE	
stCtrl.iAxisState	INT	3	

变量	设置值	实际值
位置 [u]	0.00	0.00
速度 [u/s]	0.00	0.00
加速度 [u/s²]	0.00	0.00
扭矩	0.00	0.00

功能 2: JOG

通过 stPara.rJogVel、stPara.rAcc、stPara.rDec、stPara.rJeck 设置 Jog 运动的速度，加速度、减速度、Jeck 值。控制 JOG 运行，使用 stCtrl.bJogForward、stCtrl.bBackwards 等于 TRUE 的时候正向运动或者负向运动，等于 FALSE 的时候停止正向或者负向运动。若产生报错，stCtrl.bError=TRUE、并且输出 ErrorID。

表达式	类型	值	准备值
stCtrl.bJogForward	BOOL	TRUE	
stCtrl.iAxisState	INT	5	

变量	设置值	实际值
位置 [u]	114.30	114.20
速度 [u/s]	100.00	100.00
加速度 [u/s²]	0.00	0.00
扭矩	0.00	0.00

功能 3: 复位

当发生可复位的报错时候，通过 stCtrl.iReset 给复位命令。FB 中做了判断，如果是总线报错，那么会执行总线复位的功能；如果是 MC 指令块报错，执行 Reset 命令；如果没有产生报错，则直接返回完成。复位命令是 stCtrl.iReset:=1；当 stCtrl.iReset=88 则完成复位

bJogBackwards	BOOL	FALSE	手动反转
iReset	INT	1	复位
iHome	INT	0	回原点
iReset	INT	88	复位

功能 4：绝对定位

通过 stPara.rAcc、stPara.rDec、stPara.rJekc 设置绝对定位的加速度、减速度、Jekc 值；绝对定位的位置和速度通过 stCtrl.rPostion 和 stCtrl.rVelocity 设置。开启绝对定位需要对 stCtrl.iMoveAbs 发 1 的命令，定位完成标志位为 stCtrl.iMoveAbs=88。

stCtrl.rPostion、stCtrl.rVelocity、stCtrl.iMoveAbs 三个可以同时给值进去。在内部已处理了时序，规避由于没有参数导致绝对定位报错的问题。

给值：

iMoveAbs	INT	0	1	绝对定位(**)
iMoveRel	INT	0		相对定位
bHomeSensor	BOOL	FALSE		回原点传感器信
rPosition	REAL	0	10	定位位置值接口
rVelocity	REAL	0	360	定位速度值接口

运动中：

iMoveAbs	INT	4	运动中	绝对定位(**)
iMoveRel	INT	0		相对定位
bHomeSensor	BOOL	FALSE		回原点传感器信
rPosition	REAL	10		定位位置值接口
rVelocity	REAL	360		定位速度值接口
bError	BOOL	FALSE		报错传出接口
udErrorId	UDINT	0		报错代码
rActPos	REAL	1155.84		当前位置
rActVel	REAL	-360		当前速度
iAxisState	INT	4		轴状态

按照设定的参数在运动；轴状态=4

运动完成：

iMoveAbs	INT	88	运动完成	绝对定位(**)
iMoveRel	INT	0		相对定位
bHomeSensor	BOOL	FALSE		回原点传感器信
rPosition	REAL	10		定位位置值接口
rVelocity	REAL	360		定位速度值接口
bError	BOOL	FALSE		报错传出接口
udErrorId	UDINT	0		报错代码
rActPos	REAL	10		当前位置
rActVel	REAL	0		当前速度
iAxisState	INT	3		轴状态

到设定位置，轴状态=3

功能 5：相对定位

通过 stPara.rAcc、stPara.rDec、stPara.rJekc 设置相对定位的加速度、减速度、Jekc 值；定位的位置和速度通过 stCtrl.rPostion 和 stCtrl.rVelocity 设置。开启相对定位需要对

stCtrl.iMoveRel 发 1 的命令，定位完成标志位为 stCtrl.iMoveRel=88.

stCtrl.rPostion、stCtrl.rVelocity、 stCtrl.iMoveRel 三个可以同时给值进去。在内部已处理了时序，规避由于没有参数导致定位报错的问题。

给参数：

iMoveRel	INT	0	1	相对定位
bHomeSensor	BOOL	FALSE	给参数	回原点传感器信
rPosition	REAL	10	3600	定位位置值接口
rVelocity	REAL	360	180	定位速度值接口

运动中：

iMoveRel	INT	3	相对定位中	相对定位
bHomeSensor	BOOL	FALSE		回原点传感器信
rPosition	REAL	3600		定位位置值接口
rVelocity	REAL	180		定位速度值接口
bError	BOOL	FALSE		报错传出接口
udErrorId	UDINT	0		报错代码
rActPos	REAL	127.54	按照设定的值在运动	当前位置
rActVel	REAL	180	轴状态=4	当前速度
iAxisState	INT	4		轴状态

运动完成：

iMoveRel	INT	88	运动完成	相对定位
bHomeSensor	BOOL	FALSE		回原点传感器信
rPosition	REAL	3600		定位位置值接口
rVelocity	REAL	180		定位速度值接口
bError	BOOL	FALSE		报错传出接口
udErrorId	UDINT	0		报错代码
rActPos	REAL	3610	完成运动，轴状态切换到3	当前位置
rActVel	REAL	0		当前速度
iAxisState	INT	3		轴状态

若要产生负向的相对运动；则定位位置给负值即可

iMoveRel	INT	3		相对定位
bHomeSensor	BOOL	FALSE		回原点传感器信
rPosition	REAL	-3600		定位位置值接口
rVelocity	REAL	180		定位速度值接口
bError	BOOL	FALSE		报错传出接口
udErrorId	UDINT	0		报错代码
rActPos	REAL	3336.76		当前位置
rActVel	REAL	-180		当前速度
iAxisState	INT	4		轴状态

功能 6：设定当前位置

通过设置 stCtrl.rPostion 的值，对 stCtrl.iSetPostion 写 1，可以把轴当前位置修改成任意值

给值：

iSetPosition	INT	0	1	3	开启设定	修改轴当前位置
iMoveAbs	INT	0				绝对定位(**)
iMoveRel	INT	0				相对定位
bHomeSensor	BOOL	FALSE				回原点传感器信号
rPosition	REAL	0	100	2	设定值	定位位置值接口
rVelocity	REAL	0				定位速度值接口
bError	BOOL	FALSE				报错传出接口
udErrorId	UDINT	0				报错代码
rActPos	REAL	0		1	当前位置为0	当前位置
rActVel	REAL	0				当前速度
iAxisState	INT	3				轴状态

修改完成：

iSetPosition	INT	88		2	修改完成	修改轴当前
iMoveAbs	INT	0				绝对定位(**)
iMoveRel	INT	0				相对定位
bHomeSensor	BOOL	FALSE				回原点传感
rPosition	REAL	100				定位位置值
rVelocity	REAL	0				定位速度值
bError	BOOL	FALSE				报错传出接
udErrorId	UDINT	0				报错代码
rActPos	REAL	100		1	当前值被修改为100	当前位置
rActVel	REAL	0				当前速度
iAxisState	INT	3				轴状态

功能 7：回原点

轴回原点根据实际接线（接 PLC 还是接驱动器 DI）和现场要求，可以对 FB 的回原点方式进行设置。

- (1) iHomeMode =0 调用 SMC_Homing，寻找传感器原点。原点接在 PLC 的 DI。初始回原方向通过 bHomeDirection 设置，回原速度通过 stPara.rHomeVel 设置，触发 stCtrl.iHome=1，完成返回 88。
- (2) iHomeMode =1，先根据设定方向高速回原，然后找到原点下降沿后，返回低速找原点信号。原点接在 PLC 的 DI。初始回原方向通过 bHomeDirection 设置，回原速度通过 stPara.rHomeVel 设置，触发 stCtrl.iHome=1，完成返回 88。
- (3) iHomeMode =2，调用 MC_Home，使用 6098 规定回原方式。回原速度通过 6099 设定。要求限位和原点信号都接在驱动器。

- (4) iHomeMode =3, 先找感应器作为原点, 再找一圈内的 Z 相。即在=0 的模式上增加 402 协议内规定的 33 或者 34 回原方式。需要在 SDO 配置 6098/6099,。原点信号接在 PLC。初始回原方向通过 bHomeDirection 设置, 回原速度通过 stPara.rHomeVel 设置, 触发 stCtrl.iHome=1, 完成返回 88.

功能 8: 强制开抱闸

抱闸在使能成功后自动打开, 使能关断的时候开启抱闸。

功能 9: 轴暂停

stCtrl.iPauseAction=1, 激活暂停; =88 暂停完成。=-1 关闭暂停轴继续运动。=-88 暂停关闭完成。(仅在绝对定位中生效)

功能 10: 轴停止运动

停止运行的减速度通过 stPara.rStopDec 给定。stCtrl.iStop 给命令 1, 激活轴停止。返回 88 停止成功。

stPar_AxisMain	ST_AxisPara			AxisMain的参数类数据
rJogVel	REAL	100		JOG速度
rAcc	REAL	10000		加速度
rDec	REAL	10000		减速度
rJerk	REAL	10000		减减速
rStopDec	REAL	10000		Stop的减速度
rHomeVel	REAL	360		回原速度
rMaxLimit	REAL	0		最大软限位
rMinLimit	REAL	0		最小软限位
rHomeOffset	REAL	0		回原点偏移
stCtrl_AxisMain	ST_AxisCtrl			AxisMain的控制类数据
bPower	BOOL	TRUE		使能接口
iStop	INT	88		停止

功能 11: 轴信息传出

stCtrl.rActPos: 轴当前位置传出

stCtrl.rActVel: 轴当前速度传出

stCtrl.bError :轴产生报错, 则为 ON

stCtrl.udErrorID :报错时的代码

stCtrl.bHomeFlag :回过原点后的标志位

stCtrl.iAxisState :轴状态机 (参考 codesys 轴状态机即可)

若存在 402 轴数比较多, 可用数组和 For 循环批量实例:

```

1
2  VAR_GLOBAL
3      //=====轴控定义=====
4      g_RealAxis      : ARRAY[0..cnAxisNum-1] OF POINTER TO AXIS_REF_ETC_DS402_CS ;
5      g_stCtrl        : ARRAY[0..cnAxisNum-1] OF   st_AxisCtrl;
6      g_stPar         : ARRAY[0..cnAxisNum-1] OF   ST_AxisPara;
7  END_VAR
8      //=====常量=====
9  VAR_GLOBAL CONSTANT
10     cnAxisNum          : INT := 32 ;           //轴数量
11  END_VAR
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661

```

2、相关变量

变量名	名称	数据类型	输入输出属性	描述
byStep	需要+1 的变量	INT	输入输出变量	byStep:=byStep + 1;

3、使用实例：iA 不断的再加一

表达式	类型	值
 iA	INT	15120

1

2

FunNext (byStep:= iA 15120);

02_插补

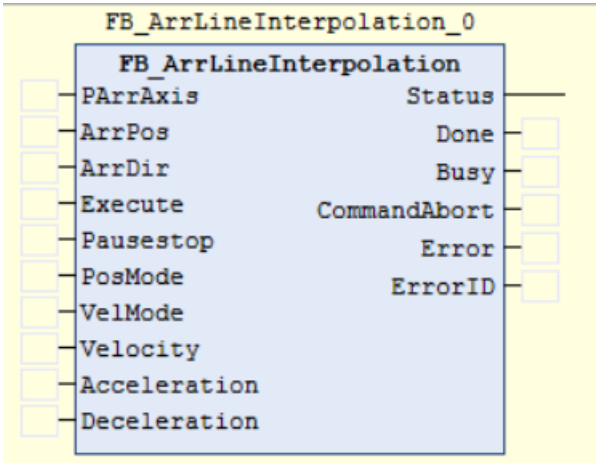
FB_ArrLineInterpolation 多轴直线插补

直线插补。最大支持 32 轴,通过指针数组给轴名字，支持线性以及模态轴。

注意：目前还不支持速度衔接，触发运动该指令时 需要确保轴速度为 0，如果不为 0 速度会从当前速度突变到 0 然后开始正常规划运动

1、指令格式

图形表现：



ST 表现：

```
FB_ArrLineInterpolation_0(  
    PArrAxis:= ,  
    ArrPos:= ,  
    ArrDir:= ,  
    Execute:= ,  
    Pausestop:= ,  
    PosMode:= ,  
    VelMode:= ,  
    Velocity:= ,  
    Acceleration:= ,  
    Deceleration:= ,  
    Status=> ,  
    Done=> ,  
    Busy=> ,  
    CommandAbort=> ,  
    Error=> ,  
    ErrorID=> );
```

2、相关变量

变量名	名称	数据类型	输入输出属性	描述
PArrAxis	轴指针数组	ARRAY[*] OF POINTER TO AXIS_REF_ETC_DS402_CS	输入输出变量	轴指针数组
ArrPos	目标位置	ARRAY[*] OF LREAL	输入输出变量	目标位置
ArrDir	方向设定	ARRAY[*] OF INT	输入输出变量	方向, -1: 反向, 0: 最短路径, 1: 正向, 。如果非 -1、0、1 则认为是最短路径
Execute	运行	BOOL	输入变量	上升沿有效
Pausestop	暂停	BOOL	输入变量	=TRUE 暂停
PosMode	坐标模式设置	BOOL	输入变量	=0 绝对坐标, =1 相对坐标
VelMode	速度模式设置	INT	输入变量	=0 梯形加速, =1 S 型加速
Velocity	速度值	LREAL	输入变量	速度
Acceleration	加速度值	LREAL	输入变量	加速度
Deceleration	减速度值	LREAL	输入变量	减速度
Status	状态输出	INT	输出变量	0:无状态, 1:加速,2:匀速, 3 减速, 4 结束, 5 暂停减速中, 6 暂停完成
Done	完成	BOOL	输出变量	=TRUE 完成
Busy	运行中	BOOL	输出变量	=TRUE 执行中
CommandAbort	中断执行	BOOL	输出变量	=TRUE 指令中断
Error	报错	BOOL	输出变量	=True 错误
ErrorID	错误代码	INT	输出变量	0 无错误, 1 轴名字、位置、方向数组长度不一致或长度超 32, 2 速度类参数错误, 3 轴未准备好,

3、具体使用

(1) 声明与调用

```

1 PROGRAM PLC_PRG
2 VAR
3   fbLineInter :FB ArrLineInterpolation;
4   VArrAxis :ARRAY[1..3] OF POINTER TO AXIS_REF_ETC_DS402_CS:= [ADR(AxisX),ADR(AxisY),ADR(AxisZ)]; //轴指针数组
5   VArrPos :ARRAY[1..3] OF LREAL:= [100,200,300]; //目标位置
6   VArrDir :ARRAY[1..3] OF INT; //0 最短路径, 1正向, -1反向
7   V: LREAL := 100;
8   ACC: LREAL := 100;
9   DEC: LREAL := 100;
10 END_VAR

```

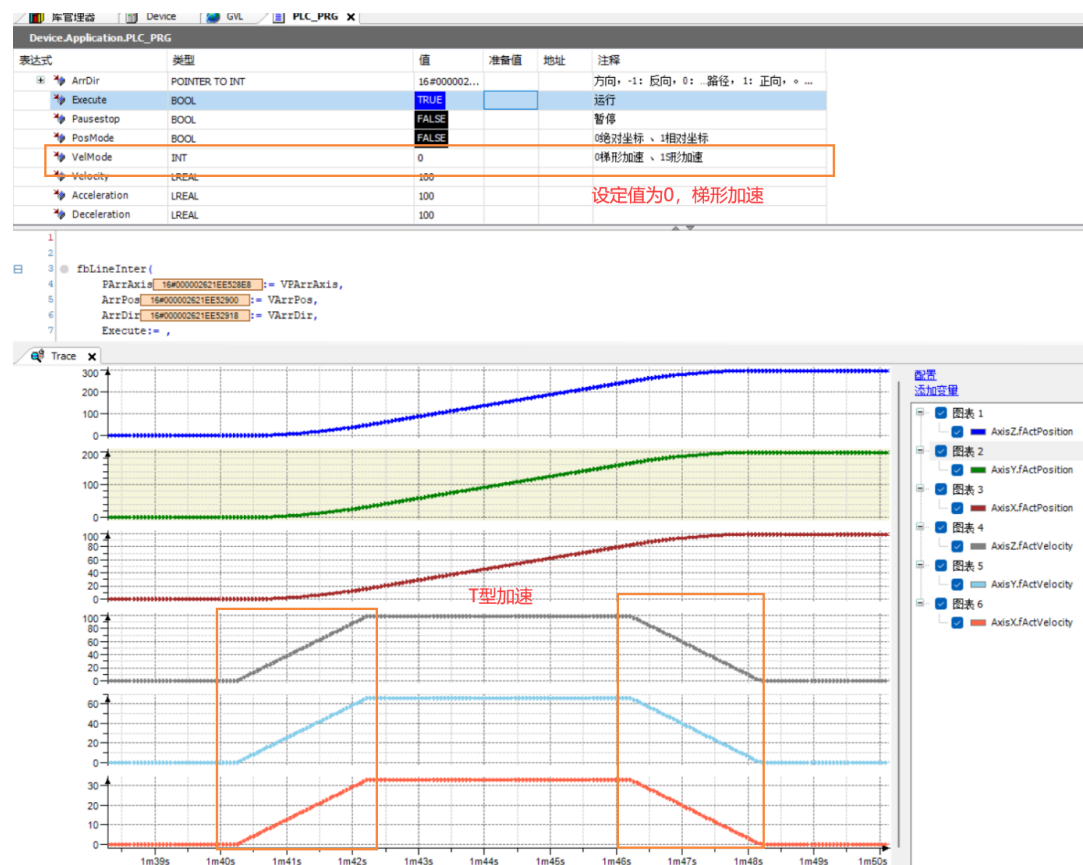
```

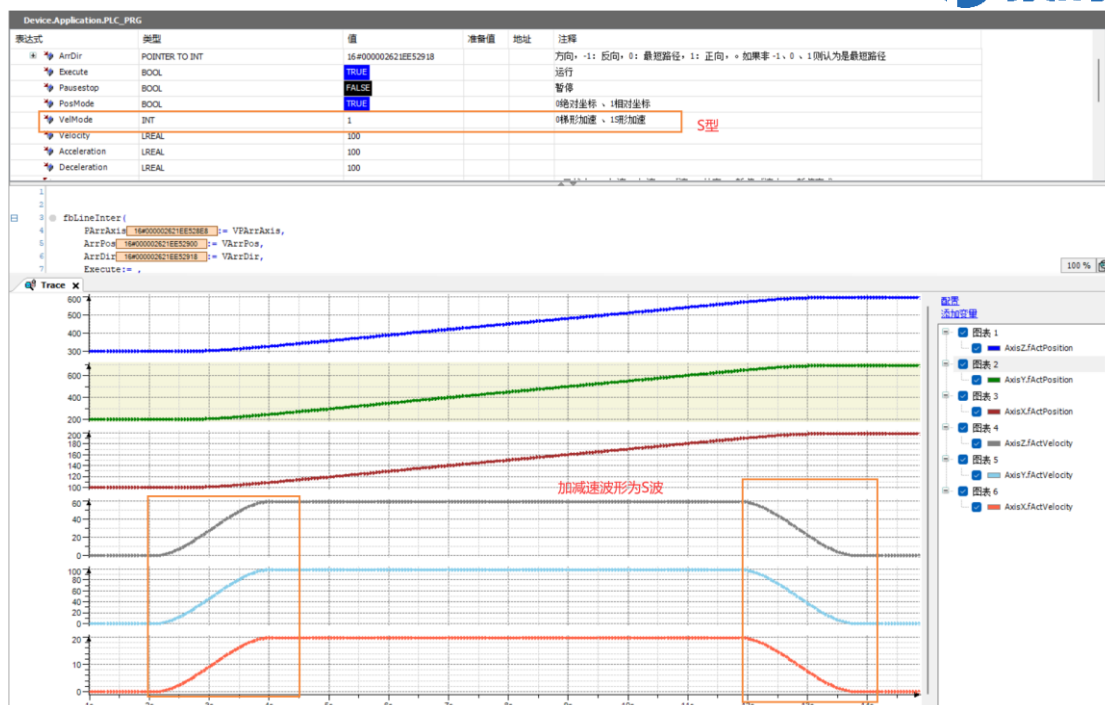
1
2
3 fbLineInter(
4   PArrAxis:= VArrAxis,
5   ArrPos:= VArrPos,
6   ArrDir:= VArrDir,
7   Execute:= ,
8   Pausestop:= ,
9   PosMode:= ,
10  VelMode:= ,
11  Velocity:= V,
12  Acceleration:= ACC,
13  Deceleration:= DEC,
14  Status=> ,
15  Done=> ,
16  Busy=> ,
17  CommandAbort=> ,
18  Error=> ,
19  ErrorID=> );

```

声明数组轴。需要多少个轴同时插补就声明数组长度对应多少
每个轴对应的位置和方向设置

(2) T型和S型加减速模式通过 VelMode 设定, =0 T型加速、=1S型加速。波形对比:



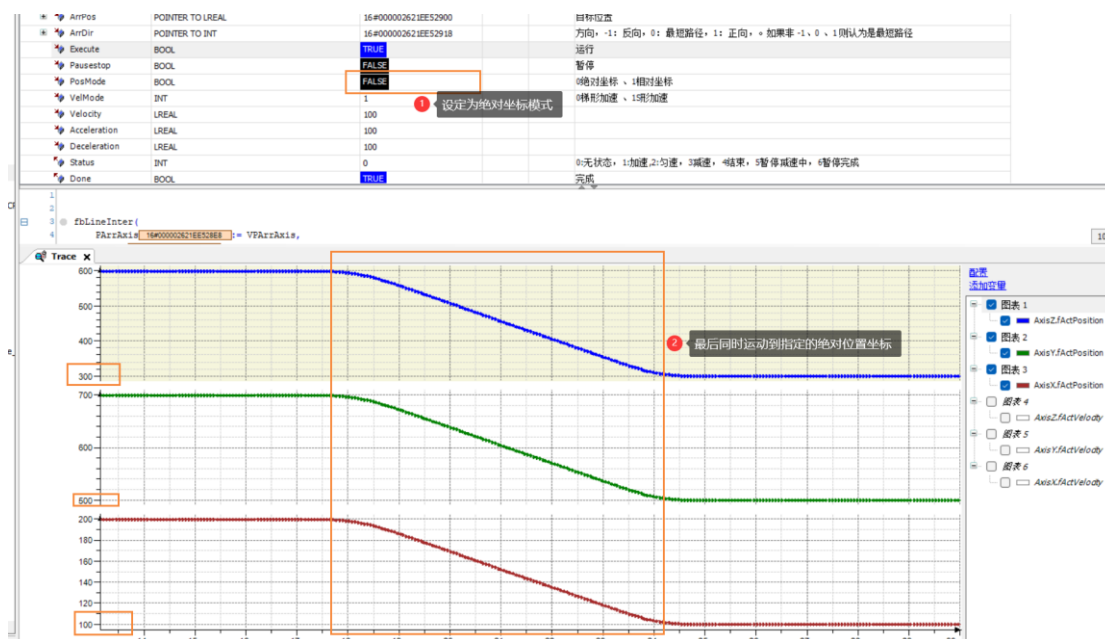


- (3) 位置值为相对坐标或绝对坐标。PosMode=0, 设定的位置值为绝对坐标; =1, 设定的位置值为相对坐标。

相同的参数值, 在绝对和相对坐标下的不同表现:

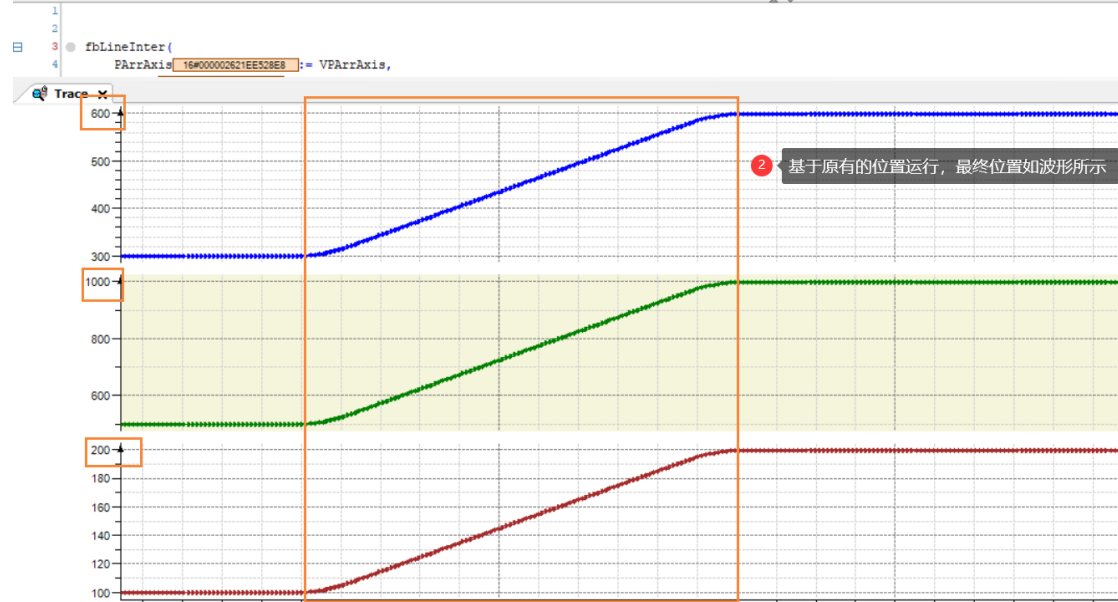
VArrPos	ARRAY [1..3] OF LREAL		
VArrPos[1]	LREAL		100
VArrPos[2]	LREAL		500
VArrPos[3]	LREAL		300

绝对模式:

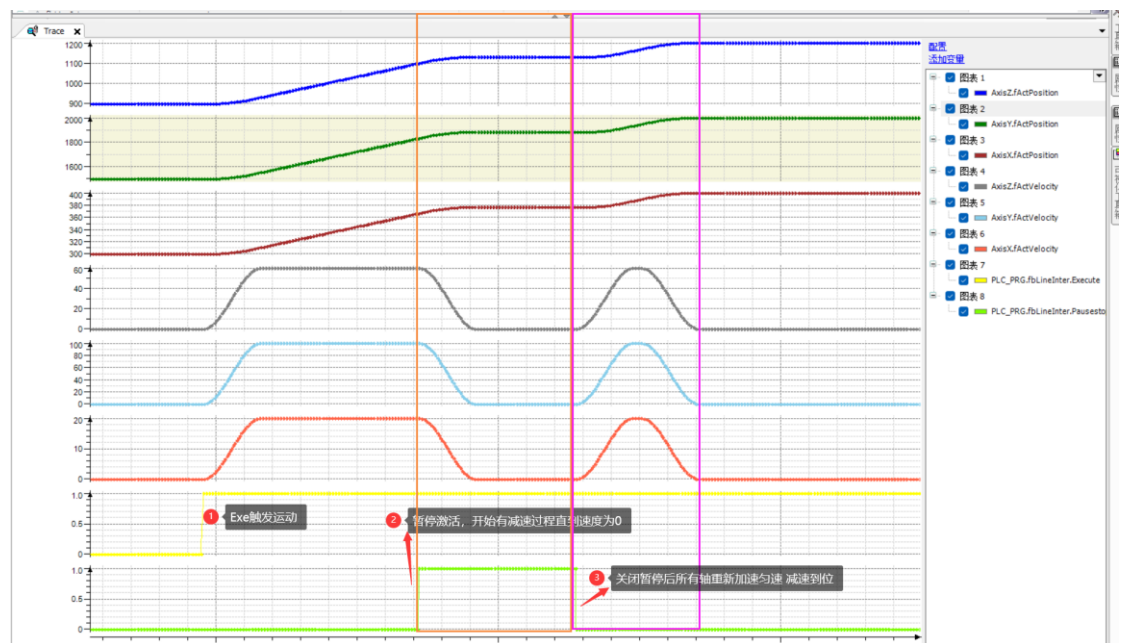


相对模式：

ArrPos	POINTER TO LREAL	16#000002621EE52900	目标位置
ArrDir	POINTER TO INT	16#000002621EE52918	方向, -1: 反向, 0: 最短路径, 1: 正向, 如果非 -1、0、1
Execute	BOOL	TRUE	运行
Paustestop	BOOL	FALSE	暂停
PosMode	BOOL	TRUE	0绝对坐标、1相对坐标
VelMode	INT	1	0梯形加速、1线性加速
Velocity	LREAL	100	
Acceleration	LREAL	100	
Deceleration	LREAL	100	
Status	INT	0	0:无状态, 1:加速, 2:匀速, 3:减速, 4:结束, 5:暂停减速中, 6:暂停完成
Done	BOOL	TRUE	



(4) 暂停和暂停后再启动



(5) 每个轴运动的路径可以选, 是用最短路径还是正反向

```

VArrPos      :ARRAY[1..3] OF LREAL:= [100,200,300]; //目标位置
VArrDir      :ARRAY[1..3] OF INT; //0 最短路径, 1正向, -1反向
V: LREAL := 100;
ACC: LREAL := 100;
DEC: LREAL := 100;
END_VAR

fbLineInter(
  PArrAxis:= VPArrAxis,
  ArrPos:= VArrPos,
  ArrDir:= VArrDir,
  Execute:= ,

```

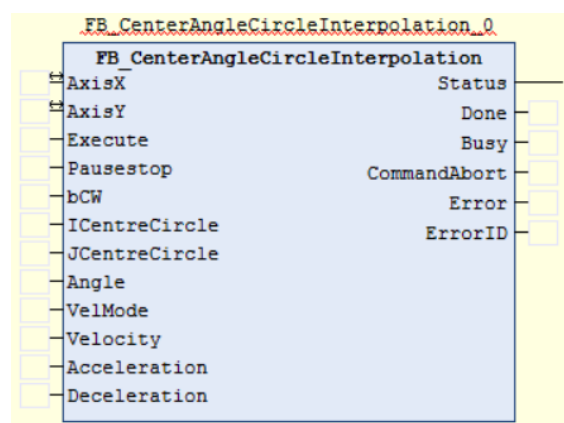
FB_CenterAngleCircleInterpolation 两轴圆弧插补（圆心圆弧）

功能块说明：圆弧插补,当前位置为起点、根据圆心、和圆弧角度，规划路径

注意：目前还不支持速度衔接，触发运动该指令时 需要确保轴速度为 0，如果不为 0 速度会从当前速度突变到 0 然后开始正常规划运动

1、指令格式

图形表现：



ST 表现：

```

FB_CenterAngleCircleInterpolation_0(
  AxisX:= ,
  AxisY:= ,
  Execute:= ,
  Pausestop:= ,
  bCW:= ,
  ICentreCircle:= ,
  JCentreCircle:= ,
  Angle:= ,
  VelMode:= ,
  Velocity:= ,
  Acceleration:= ,
  Deceleration:= ,
  Status=> ,
  Done=> ,
  Busy=> ,
  CommandAbort=> ,
  Error=> ,
  ErrorID=> );

```

2、相关变量

变量名	名称	数据类型	输入输出属性	描述
AxisX	插补轴 X	AXIS_REF_SM3	输入输出变量	插补 X 轴
AxisY	插补轴 Y	AXIS_REF_SM3	输入输出变量	插补 Y 轴
Execute	运行	BOOL	输入变量	上升沿有效

Pausestop	暂停	BOOL	输入变量	=TRUE 暂停
bCw	运动方向	BOOL	输入变量	=0 CCW, =1CW
ICentreCircle	圆心坐标 X	LREAL	输入变量	圆心的 X 坐标
JCentreCircle	圆心坐标 Y	LREAL	输入变量	圆心的 Y 坐标
Angle	圆弧角度	LREAL	输入变量	圆弧的角度, 需要设置大于 0
VelMode	速度模式设置	INT	输入变量	=0 梯形加速, =1 S 型加速
Velocity	速度值	LREAL	输入变量	速度
Acceleration	加速度值	LREAL	输入变量	加速度
Deceleration	减速度值	LREAL	输入变量	减速度
Status	状态输出	INT	输出变量	0:无状态, 1:加速, 2:匀速, 3 减速, 4 结束, 5 暂停减速中, 6 暂停完成
Done	完成	BOOL	输出变量	=TRUE 完成
Busy	运行中	BOOL	输出变量	=TRUE 执行中
CommandAbort	中断执行	BOOL	输出变量	=TRUE 指令中断
Error	报错	BOOL	输出变量	=True 错误
ErrorID	错误代码	INT	输出变量	0 无错误, 1 圆弧终点或圆心数据错误 或 角度 Angle<=0 2 速度类参数错误 3 轴未准备好

3、详细使用

(1) 声明与实例化

```

PROGRAM POU_2
VAR
    FB_CenterAngleCircleInterpolation_0: FB_CenterAngleCircleInterpolation;
    GetR: REAL;

    ANG: LREAL := 90;
    V: LREAL := 100;
    ACC: LREAL := 100;
    DEC: LREAL := 100;
END_VAR

```

```

FB_CenterAngleCircleInterpolation_0(
    AxisX:= AxisX,
    AxisY:= AxisY,
    Execute:= ,
    Pausestop:= ,
    bCW:= ,
    ICentreCircle:= ,
    JCentreCircle:= ,
    Angle:= ANG,
    VelMode:= ,
    Velocity:= V,
    Acceleration:= ACC,
    Deceleration:= DEC,
    Status=> ,
    Done=> ,
    Busy=> ,
    CommandAbort=> ,
    Error=> ,
    ErrorID=> );

```

① 设定哪两个轴进行插补运动

② 设定圆心坐标和圆弧角度

③ 运行速度设置

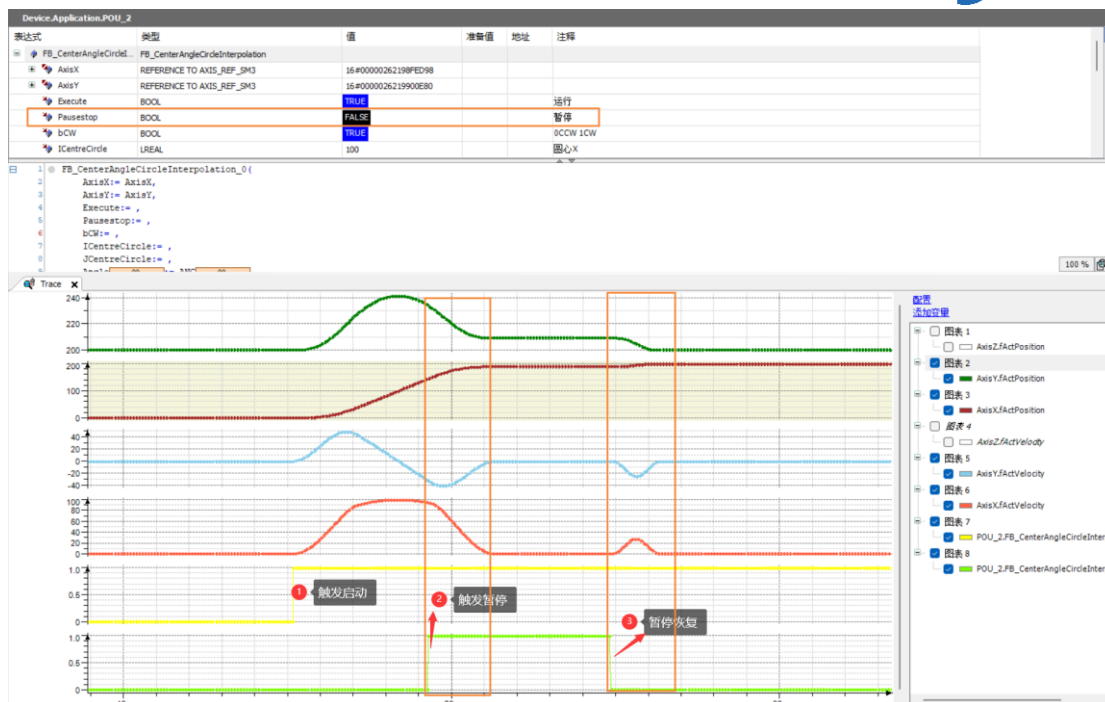
(2) 梯形加速和 S 型加速；通过 VelMode 设定。=0 为梯形加速，=1 为 S 型加速。



(3) 圆弧方向 bCW=0 为 CCW 方向，=1 为 CW 方向



(4) 暂停后再启动。使用 Pausestop 暂停和暂停后再次启动。如下所示：



FB_EndPosCircleInterpolation 两轴圆弧插补 (圆心、结束位置、圈数)

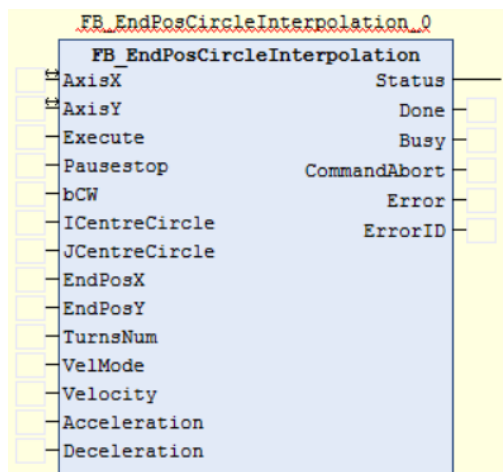
功能块说明：两轴圆弧插补 ,当前位置为起点，根据圆心、结束位置、圈数，规划路径 。如果 end 位置就在当前位置，则为一圈

注意：目前还不支持速度衔接，触发运动该指令时 需要确保轴速度为 0，如果不为 0 速度会从当前速度突变到 0 然后开始正常规划运动

1、指令格式

图形表现：

ST 表现：



```
FB_EndPosCircleInterpolation_0(
    AxisX:= ,
    AxisY:= ,
    Execute:= ,
    Pausestop:= ,
    bCW:= ,
    ICentreCircle:= ,
    JCentreCircle:= ,
    EndPosX:= ,
    EndPosY:= ,
    TurnsNum:= ,
    VelMode:= ,
    Velocity:= ,
    Acceleration:= ,
    Deceleration:= ,
    Status=> ,
    Done=> ,
    Busy=> ,
    CommandAbort=> ,
    Error=> ,
    ErrorID=> );
```

2、相关变量

变量名	名称	数据类型	输入输出属性	描述
AxisX	插补轴 X	AXIS_REF_SM3	输入输出变量	插补轴 X
AxisY	插补轴 Y	AXIS_REF_SM3	输入输出变量	插补轴 Y
Execute	运行	BOOL	输入变量	上升沿有效
Pausestop	暂停	BOOL	输入变量	=TRUE 暂停
bCw	运动方向	BOOL	输入变量	=0 CCW, =1CW
ICentreCircle	圆心坐标 X	LREAL	输入变量	圆心的 X 坐标
JCentreCircle	圆心坐标 Y	LREAL	输入变量	圆心的 Y 坐标
EndPosX	结束点 X 坐标	LREAL	输入变量	结束点 X 坐标
EndPosY	结束点 Y 坐标	LREAL	输入变量	结束点 Y 坐标
TurnsNum	圆弧圈数	UINT	输入变量	圆弧圈数
VelMode	速度模式设置	INT	输入变量	=0 梯形加速, =1 S 型加速
Velocity	速度值	LREAL	输入变量	速度
Acceleration	加速度值	LREAL	输入变量	加速度
Deceleration	加速度值	LREAL	输入变量	减速度
Status	状态输出	INT	输出变量	0:无状态, 1:加速, 2:匀速, 3 减速, 4 结束, 5 暂停减速中, 6 暂停完成
Done	完成	BOOL	输出变量	=TRUE 完成
Busy	运行中	BOOL	输出变量	=TRUE 执行中
CommandAbort	中断执行	BOOL	输出变量	=TRUE 指令中断
Error	报错	BOOL	输出变量	=TRUE 错误
ErrorID	错误代码	INT	输出变量	0 无错误,

				1 圆弧终点或圆心数据错误 2 速度类参数错误 3 轴未准备好
--	--	--	--	---------------------------------------

3、具体使用

(1) 声明与写值

```

PROGRAM POU_3
VAR
  FB_EndPosC: FB_EndPosCircleInterpolation ;

  V: LREAL := 100;
  ACC: LREAL := 100;
  DEC: LREAL := 100;

  EndX:LREAL;
  EndY:LREAL;
  Angle:LREAL;
END_VAR

```

```

FC_Circle_AngleGteEndXY(In_I:= 0, In_J:= 100, In_R:= 100, In_Angle:= Angle, Out_EndX=> EndX, Out_EndY=>EndY );
FB_EndPosC(
  AxisX:= AxisX,
  AxisY:= AxisY,
  Execute:= ,
  Pauestop:= ,
  bCW:= ,
  ICentreCircle:= 0,
  JCentreCircle:=100 ,
  EndPosX:= EndX,
  EndPosY:= EndY,
  TurnsNum:= ,
  VelMode:= ,
  Velocity:= V,
  Acceleration:= ACC,
  Deceleration:= DEC,
  Status=> ,
  Done=> ,
  Busy=> ,
  CommandAbort=> ,
  Error=> ,
  ErrorID=> );

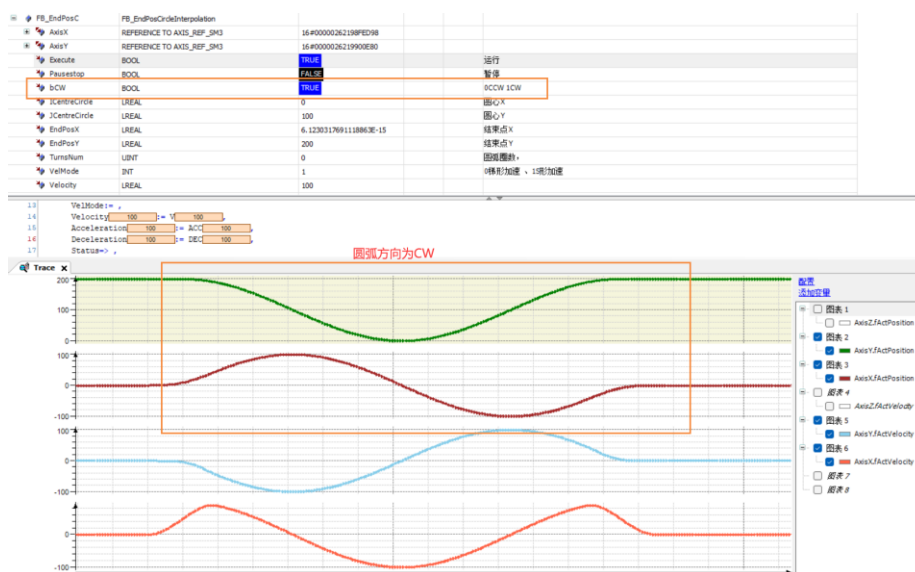
```

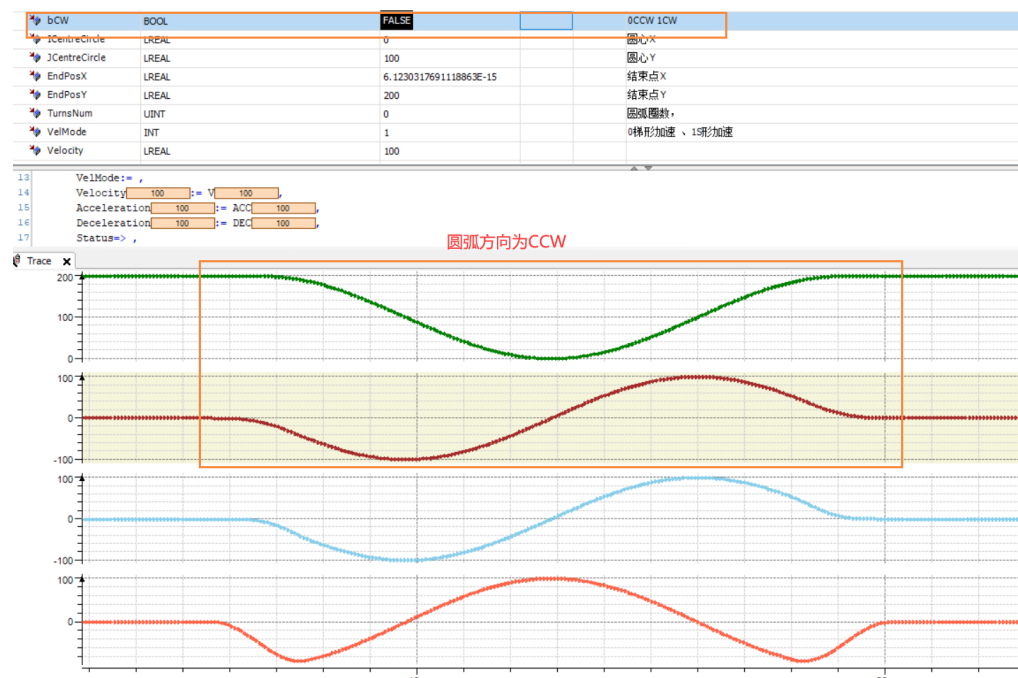
注：EndX 和 EndY 可以用通过半径用函数设定

(2) T 型和 S 型加速



(3) 圆弧方向通过 bCW 引脚设定





(4) 插补暂停后再启动

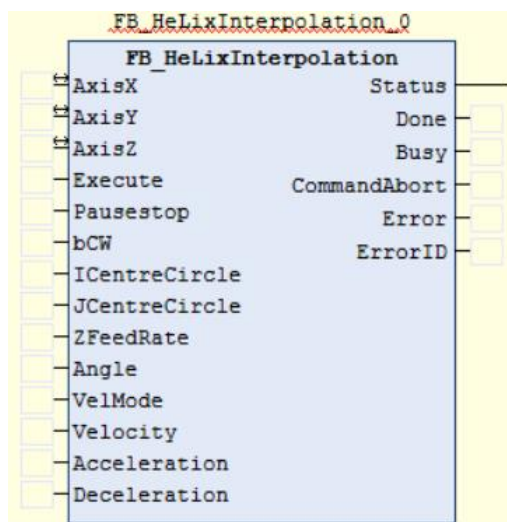


FB_HeLixInterpolation 三轴螺旋插补

功能块说明：三轴螺旋插补。当前位置为起点、根据圆心，螺旋角度 Z 增量，规划路径
 注意：目前还不支持速度衔接，触发运动该指令时 需要确保轴速度为 0，如果不为 0 速度会从当前速度突变到 0 然后开始正常规划运动

1、指令格式

图形表现：



ST 表现：

```
FB_HelixInterpolation_0(
    AxisX:= ,
    AxisY:= ,
    AxisZ:= ,
    Execute:= ,
    Pausestop:= ,
    bCW:= ,
    ICentreCircle:= ,
    JCentreCircle:= ,
    ZFeedRate:= ,
    Angle:= ,
    VelMode:= ,
    Velocity:= ,
    Acceleration:= ,
    Deceleration:= ,
    Status=> ,
    Done=> ,
    Busy=> ,
    CommandAbort=> ,
    Error=> ,
    ErrorID=> );
```

2、相关变量

变量名	名称	数据类型	输入输出属性	描述
AxisX	插补轴 X	AXIS_REF_SM3	输入输出变量	插补轴 X
AxisY	插补轴 Y	AXIS_REF_SM3	输入输出变量	插补轴 Y
AxisZ	插补轴 Z	AXIS_REF_SM3	输入输出变量	插补轴 Z
Execute	运行	BOOL	输入变量	上升沿有效
Pausestop	暂停	BOOL	输入变量	=TRUE 暂停
bCw	运动方向	BOOL	输入变量	=0 CCW, =1CW
ICentreCircle	圆心坐标 X	LREAL	输入变量	圆心的 X 坐标
JCentreCircle	圆心坐标 Y	LREAL	输入变量	圆心的 Y 坐标
ZFeedRate	Z 轴增量	LREAL	输入变量	Z 轴增量
Angle	旋转角度	LREAL	输入变量	旋转角度
VelMode	速度模式设置	INT	输入变量	=0 梯形加速, =1 S 型加速
Velocity	速度值	LREAL	输入变量	速度
Acceleration	加速度值	LREAL	输入变量	加速度
Deceleration	减速度值	LREAL	输入变量	减速度
Status	状态输出	INT	输出变量	0:无状态, 1:加速, 2:匀速, 3 减速, 4 结束, 5 暂停减速中, 6 暂停完成
Done	完成	BOOL	输出变量	=TRUE 完成
Busy	运行中	BOOL	输出变量	=TRUE 执行中
CommandAbort	中断执行	BOOL	输出变量	=TRUE 指令中断
Error	报错	BOOL	输出变量	=TRUE 错误

ErrorID	错误代码	INT	输出变量	0 无错误, 1 圆弧终点或圆心数据错误, 或者 Angle<=0 2 速度类参数错误 3 轴未准备好
---------	------	-----	------	--

3、详细使用

(1) 声明与实例化

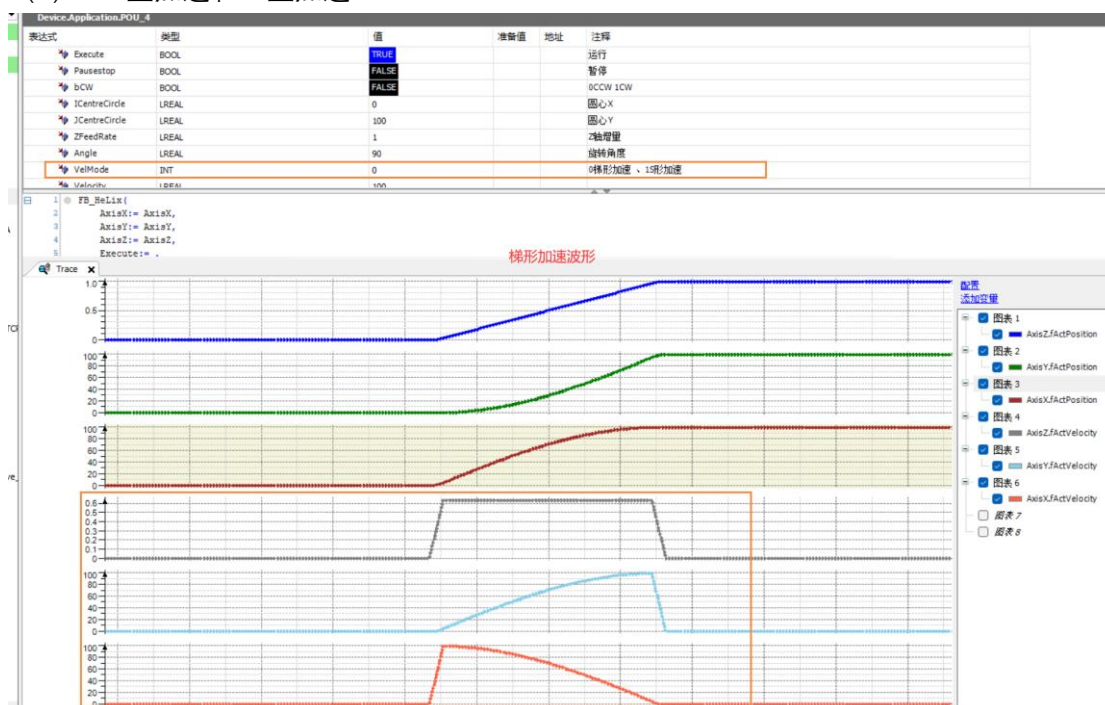
```

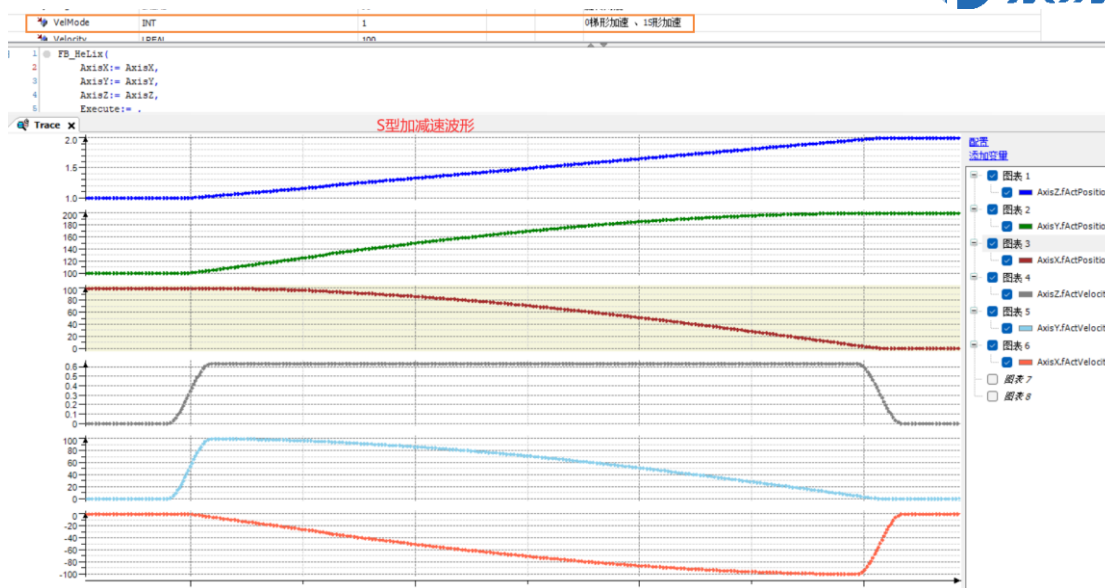
1 PROGRAM POU_4
2 VAR
3     FB_HeLix: FB_HeLixInterpolation ;
4 END_VAR

1 FB_HeLix(
2     AxisX:= AxisX,
3     AxisY:= AxisY,
4     AxisZ:= AxisZ,
5     Execute:= ,
6     Pauestop:= ,
7     bCW:= ,
8     ICentreCircle:= 0,
9     JCentreCircle:= 100,
10    ZFeedRate:= ,
11    Angle:= ,
12    VelMode:= ,
13    Velocity:= 100,
14    Acceleration:= 1000,
15    Deceleration:= 1000,
16    Status=> ,
17    Done=> ,
18    Busy=> ,
19    CommandAbort=> ,
20    Error=> ,
21    ErrorID=> );

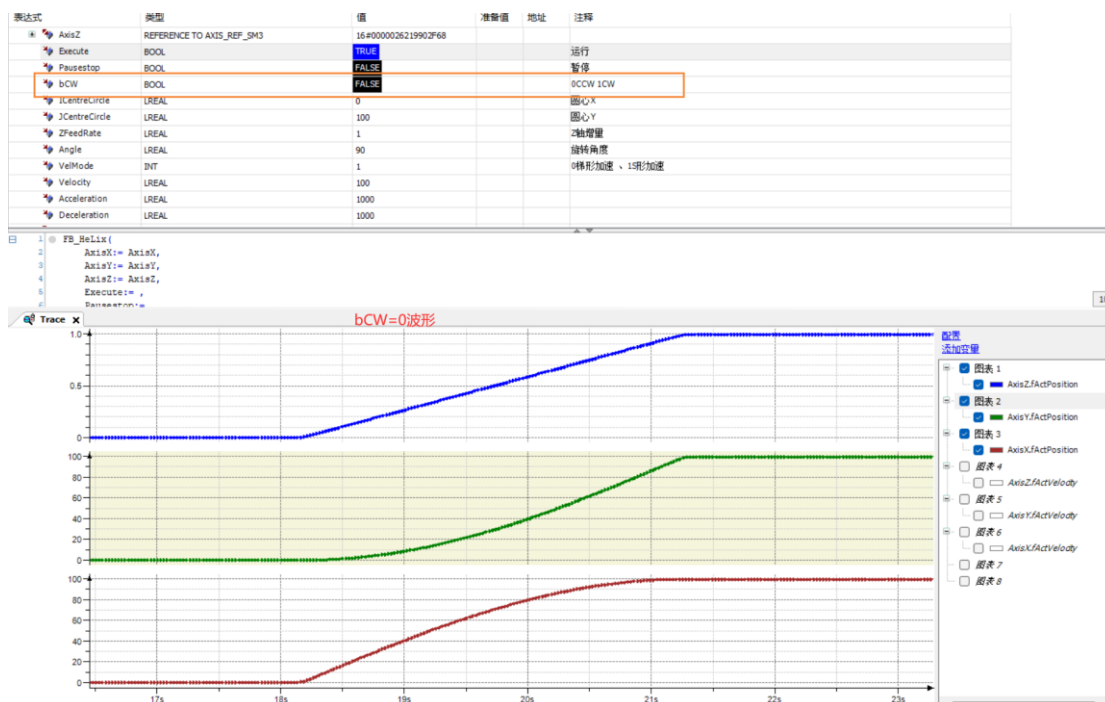
```

(2) T 型加速和 S 型加速





(3) 圆弧方向设置 bCW





(4) 暂停和暂停后再启动



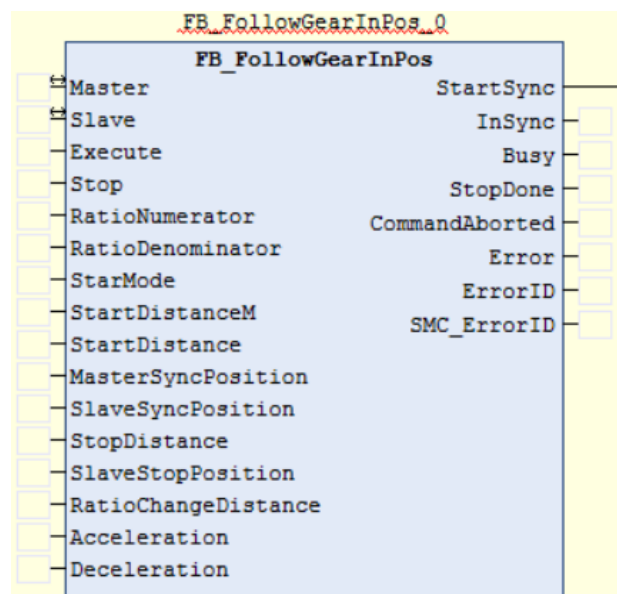
03_齿轮

FB_FollowGearInPos 特定位置齿轮耦合

功能说明：主从轴需要都是线性轴，以当前位置为起点。MasterSyncPosition, SlaveSyncPosition 为加速终点后开始以齿轮比运行主从轴耦合。位置可设定，加速的位置主从可设定，停止位置可设定，停止减速距离可设

1、指令格式

图形表现：



ST 表现：

```
FB_FollowGearInPos_0(
    Master:= ,
    Slave:= ,
    Execute:= ,
    Stop:= ,
    RatioNumerator:= ,
    RatioDenominator:= ,
    StarMode:= ,
    StartDistanceM:= ,
    StartDistance:= ,
    MasterSyncPosition:= ,
    SlaveSyncPosition:= ,
    StopDistance:= ,
    SlaveStopPosition:= ,
    RatioChangeDistance:= ,
    Acceleration:= ,
    Deceleration:= ,
    StartSync=> ,
    InSync=> ,
    Busy=> ,
    StopDone=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> ,
    SMC_ErrorID=> );
```

2、相关变量

变量名	名称	数据类型	输入输出属性	描述
Master	主轴	AXIS_REF_SM3	输入输出变量	主轴
Slave	从轴	AXIS_REF_SM3	输入输出变量	从轴
Execute	触发运行	BOOL	输入变量	触发运行
Stop	停止	BOOL	输入变量	停止
RatioNumerator	分子	LREAL	输入变量	分子
RatioDenominator	分母	LREAL	输入变量	分母
StarMode	启动模式	INT	输入变量	0:从当前位置启动，主轴以 StartDistanceM 从轴以 StartDistance 加速 1:给定位置启动，主轴以 StartDistance 从轴以 StartDistance 加速 2:从当前位置启动，按 StartDistance 齿轮比加速 3:按给定位置启动，按 StartDistance 齿轮比加速
StartDistanceM	起始位置	LREAL	输入变量	StarMode=0 有效，0 则按 ACC 加速耦合 主轴位移 StartDistanceM，从轴位置

				StartDistance
StartDistance	起始位置	LREAL	输入变量	起始位置
MasterSyncPosition	主轴开始同步位置	LREAL	输入变量	以当前位置为起点，到达该参数位置 + StartDistance 后开始同步，StarMode=1 会提前 StartDistance 开始耦合使加速更平滑
SlaveSyncPosition	从轴开始同步的位置	LREAL	输入变量	以当前位置为起点，到达该参数位置+ StartDistance 后开始同步
StopDistance	停止时的从轴运动距离	LREAL	输入变量	停止时 从轴的运动距离
SlaveStopPosition	停止时从轴位置	LREAL	输入变量	停止时从轴位置
RatioChangeDistance	齿轮比变更，加速距离	LREAL	输入变量	齿轮比变更，加速距离
Acceleration	加速度	LREAL	输入变量	加速度
Deceleration	减速度	LREAL	输入变量	减速度
StartSync	开始同步	BOOL	输出变量	开始同步
InSync	同步中	BOOL	输出变量	同步中
Busy	运行中	BOOL	输出变量	运行中
StopDone	停止完成	BOOL	输出变量	停止完成
CommandAborted	功能块被打断	BOOL	输出变量	功能块被打断
Error	产生错误	BOOL	输出变量	产生错误
ErrorID	功能块报错	WORD	输出变量	1:轴未准备好, 2: ACC DEC 错误, 3: 分子分母错误, 4 MC 错误, 5: iMovementType 主或从轴不是线性轴
SMC_ErrorID	SMC 错误信息	SM3_Basic.SMC_ERROR	输出变量	SMC 错误信息

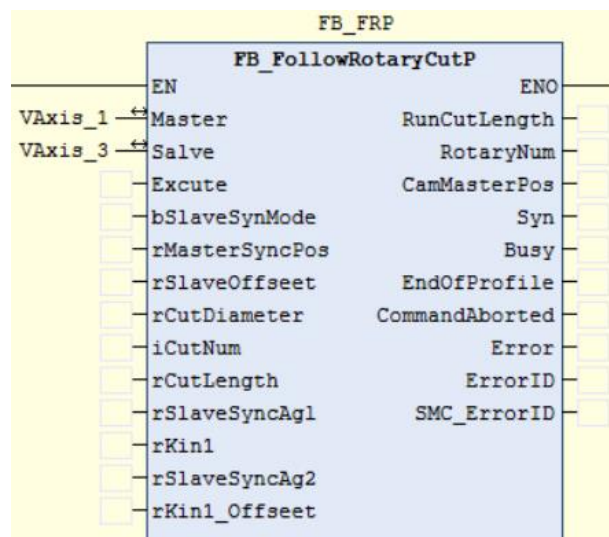
04_凸轮

FB_FollowRotaryCutP 飞剪跟随

功能块说明：本功能块适用于飞剪类型动作 可根据需求指定主轴周期（裁切长度）、从轴周期、同步比例。

1、指令格式

图形表现



ST 表现

```
FB_FollowRotaryCutP(
    Master:=VAxis_1 ,
    Salve:=VAxis_2 ,
    Excute:= ,
    bSlaveSynMode:= ,
    rMasterSyncPos:= ,
    rSlaveOffseet:= ,
    rCutDiameter:= ,
    iCutNum:= ,
    rCutLength:= ,
    rSlaveSyncAgl:= ,
    rKin1:= ,
    rSlaveSyncAg2:= ,
    rKin1_Offseet:= ,
    RunCutLength=> ,
    RotaryNum=> ,
    CamMasterPos=> ,
    Syn=> ,
    Busy=> ,
    EndOfProfile=> ,
    CommandAborted=> ,
    Error=> ,
    ErrorID=> ,
    SMC_ErrorID=> );
```

2、相关变量

输入输出变量

输入输出变量	名称	数据类型	有效范围	描述
Master	主轴	AXIS_REF_SM3		映射到轴，即 AXIS_REF_SM3 的一个实例,线性轴
Salve	从轴	AXIS_REF_SM3		映射到轴，即 AXIS_REF_SM3 的一个实例，模态轴 360°

输入变量

输入变量	名称	数据类型	有效范围	描述
Excute	旋切启动	BOOL	TRUE,FALSE	上升沿，执行旋切
bSlaveSynMode	同步模式	BOOL		0:直接以当前速比耦合， 1:当前到下一点为加速耦合
rMasterSyncPos	主轴启动位置	LREAL		Master 经过该位置后开始运行凸轮
rSlaveOffseet	从轴偏移	LREAL	>=0 <= 360	从旋切该角度切入运行
rCutDiameter	切刀直径	LREAL	>0.0	切刀的直径
iCutNum	切刀数量	Int	>0	切刀数量
rCutLength	主轴周期	LREAL	>0	旋切主轴周期
rSlaveSyncAgl	同步角度 1	LREAL	>0	切刀咬合区间主从轴线速度同步区间 (度)

rKin1	同步角度 1 的速 比	LREAL	>0	主轴与从轴同步区间线速度同步比例
rSlaveSyncAg2	同步角度 2	LREAL	>0<rSlaveSync Agl	该角度区间 rKin1_Offseet<>0 时表现为 五次曲线
rKin1_Offseet	同步速比 1 偏移	LREAL		在同步的速度下, 调整 rSlaveSyncAg2 区 间的速度

输出变量

输出变量	名称	数据类型	有效范围	描述
RunCutLength	当前裁切长度	LREAL		当前运行裁切长度
RotaryNum	当前切刀	Int		当前裁切的切刀
CamMasterPos	当前凸轮主轴位 置	LREAL	0- rCutLength	当前旋切主轴位置
Syn	同步区	BOOL	TRUE,FALSE	切刀线速度与主轴线速度同步时 该引脚置 true 当离开同步区该引脚置 false
Busy	运行	BOOL	TRUE,FALSE	Execute 输入上升沿时, 置位 TRUE, 示旋切关系耦合中, 需用 外部打断指令
EndOfProfile	凸轮曲线完成	BOOL	TRUE,FALSE	凸轮曲线执行完一次, 该引脚输出一次
CommandAborted	打断	BOOL	TRUE,FALSE	如果旋切从轴被外部打断, 该位 置 TRUE 当指令的执行条件 OFF 时, Error 位被复位。
Error	报错	BOOL	TRUE,FALSE	如果检测到有错误, Error 位被 置位; 当指令的执行条件 OFF 时, Error 位被复位。
ErrorID	功能块错误 ID	WORD		1:轴未准备好, 2: 工艺参数错误, 3: 跟随同步错误, 4:当前轴为非 模态轴
SMC_ErrorID	轴错误 ID	SMC_ERROR		当前轴控功能块错误 ID

3、使用说明

4.1、bSlaveSynMode:

- 0: 耦合时从轴不做加速处理 耦合当前周期直接根据凸轮关系到达目标速度,
- 1: 耦合时从轴按 S 形加速跟随到同步。注意:为 1 时 第一个周期不可反转 否则会突变, 加速过程规划在当前位置到下一关键点

4.2、rMasterSyncPos: 主轴到了该位置 开始跟随动作(相当于绝对位置启动凸轮, 如不需要 则直接给 Master.fsetPosition),

注意:需要是正向越过该位置才会开始执行旋切

4.3、同步区间 (rSlaveSyncAgl) 是以切刀在零点左右划分

比如设定值为 30 度 那么主从轴的同步区间在 345-15 度

4.4、从轴偏移 (rSlaveOffset)

当旋切启动切刀不是在 0 点的位置，需要给出当前切刀的位置 以保证凸轮正常运行

4.5、运行过程中参数变更

iCutNum 为 Excute 时有效，如需修改 需打断凸轮重新执行
运行过程中 rCutDiameter、rCutLength、rSlaveSyncAgl、rKin1、rSlaveSyncAg2、rKin1_Offset 参数变更后 会在下一周期有效

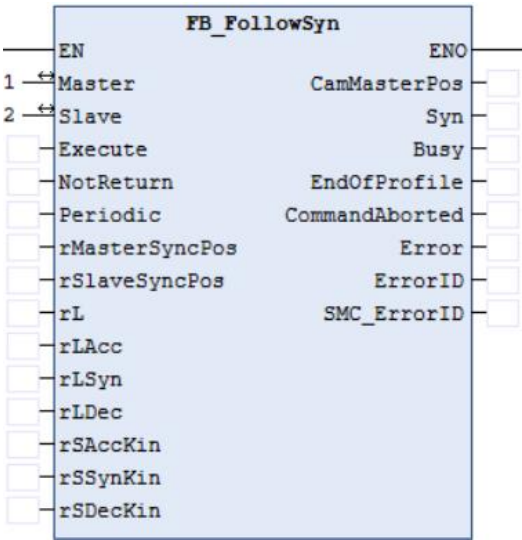
FB_FollowSyn 追剪跟随

指令名称： FB_FollowSyn 追剪功能块
功能块说明：本功能块适用于追剪类型动作 可根据设定追剪参数实现定长追剪

1、指令格式

图形表现

ST 表现



```
FB_FollowSyn(  
    Master:=Master ,  
    Salve:=Salve ,  
    Excute:= ,  
    NotReturn:= ,  
    Periodic:= ,  
    MasterSyncPos:= ,  
    SlaveSyncPos:= ,  
    rL:= ,  
    rLAcc:= ,  
    rLSyn:= ,  
    rLDec:= ,  
    rSAccKin:= ,  
    rSSynKin:= ,  
    rSDecKin:= ,  
    CamMasterPos=> ,  
    Sync=> ,  
    Busy=> ,  
    EndOfProfile=> ,  
    CommandAborted=> ,  
    Error=> ,  
    ErrorID=> ,  
    SMC_ErrorID=> );
```

2、相关变量

输入输出变量

输入输出变量	名称	数据类型	有效范围	描述
Master	主轴	AXIS_REF_SM3		映射到轴, 即 AXIS_REF_SM3 的一个实例,线性轴

Salve	从轴	AXIS_REF_SM3		映射到轴, 即 AXIS_REF_SM3 的一个实例, 模态轴 360°
-------	----	--------------	--	--------------------------------------

输入变量

输入变量	名称	数据类型	有效范围	描述
Excute	启动	BOOL	TRUE,FALSE	上升沿, 执行旋切
NotReturn	追剪模式	BOOL	TRUE,FALSE	FALSE:带返回路径, TRUE:单次凸轮减速后无返回动作
Periodic	重复模式	BOOL	TRUE,FALSE	追剪动作是否重复 TRUE:重复, FALSE:不重复
rMasterSyncPos	主轴开始跟随位置	LREAL		Master 经过该位置后开始运行凸轮
rSlaveSyncPos	从轴开始跟随位置	LREAL		从轴从该位置开始追剪
rL	追剪主轴总长	LREAL	>0.0	
rLAcc	加速主轴距离	Int	>0	
rLSyn	同步主轴距离	LREAL	>0	
rLDec	减速主轴距离	LREAL	>0	
rSAccKin	从轴加速速比	LREAL	>0	
rSSynKin	从轴同步速比	LREAL		
rSDecKin	从轴减速速比	LREAL		

输出变量

输出变量	名称	数据类型	有效范围	描述
CamMasterPos	当前凸轮主轴位置	LREAL	0- rL	当前旋切主轴位置
Syn	同步区	BOOL	TRUE,FALSE	切刀线速度与主轴线速度同步时该引脚置 true, 当离开同步区该引脚置 false
Busy	运行	BOOL	TRUE,FALSE	Execute 输入上升沿时, 置位 TRUE, 示旋切关系耦合中, 需用外部打断指令
EndOfProfile	凸轮曲线完成	BOOL	TRUE,FALSE	凸轮曲线执行完一次, 该引脚输出一次
CommandAborted	打断	BOOL	TRUE,FALSE	如果旋切从轴被外部打断, 该位置 TRUE 当指令的执行条件 OFF 时, Error 位被复位。
Error	报错	BOOL	TRUE,FALSE	如果检测到有错误, Error 位被置位; 当指令的执行条件 OFF 时, Error 位被复位。

ErrorID	功能块错误 ID	WORD		1:轴未准备好, 2: 工艺参数错误, 3: 跟随同步错误, 4:当前轴为非 模态轴
SMC_ErrorID	轴错误 ID	SMC_ERROR		当前轴控功能块错误 ID

3、使用说明

3.1、规划的轨迹为绝对路径

当 Master.fsetposition 经过 rMasterSyncPos 位置后开始按追剪轨迹运动，从轴从 rSlaveSyncPos 开始运行

3.5、运行过程中参数变更

运行过程中 rL、rLAcc、rLDec、rSAccKin、rSSynKin、rSDecKin 参数变更后 会在下一周期有效

05_TCP 和 UDP 通讯

TCPServer_N

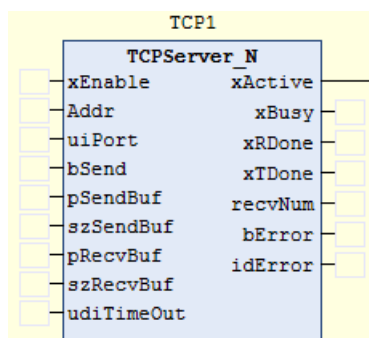
指令名称: TCPServer_N TCP 通讯 (PLC 做 Server) 基于 NBS.CAA 库

功能块说明: 本功能块用于 PLC 和其他设备之间做 TCP Socket 通讯 (PLC 做 Server), 并且进行数据的收发。

1、指令格式

图形表现:

ST 表现:



```

TCP1(
    xEnable:= ,
    Addr:= ,
    uiPort:= ,
    bSend:= ,
    pSendBuf:= ,
    szSendBuf:= ,
    pRecvBuf:= ,
    szRecvBuf:= ,
    udiTimeOut:= ,
    xActive=> ,
    xBusy=> ,
    xRDone=> ,
    xTDone=> ,
    recvNum=> ,
    bError=> ,
    idError=> );

```

2、相关变量说明

变量	名称	数据类型	输入输出属性	描述
xEnable	创建服务器	BOOL	输入变量	创建服务器
Addr	IP 地址	String	输入变量	IP
uiPort	端口号	UINT	输入变量	端口号
bSend	发送数据	BOOL	输入变量	发送
pSendBuf	发送数据的地址	POINTER TO BYTE	输入变量	发送数据的地址指针
szSendBuf	发送数据的长度	UDINT	输入变量	发送长度
pRecvBuf	接收数据的指针	POINTER TO BYTE	输入变量	接收数据的地址指针
szRecvBuf	接收数据的长度	UDINT	输入变量	接收长度
udiTimeOut	超时时间	UDINT	输入变量	超时时间 ms
xActive	侦听到客户端	BOOL	输出变量	侦听到客户端
xBusy	服务器正在运行	BOOL	输出变量	服务器正在运行中
xRDone	完成信号	BOOL	输出变量	侦听到数据的完成信号（只存在一个周期）
xTDone	发送完成	BOOL	输出变量	发送数据帧完成
RecvNum	收到的数据个数	NBS.CAA.SIZE	输出变量	接收到的数据个数
bError	报错标志	BOOL	输出变量	错误标志位
idError	Error ID	NBS.ERROR	输出变量	报错 ID；取自 NBS.Error

3.具体使用

(1) 声明与实例化

```

1  PROGRAM P2
2  VAR
3      TCP_Server_0: TCPServer_N; //功能块声明
4      RevData:DWORD; //接收的数据
5      SendData:DWORD; //发送的数据
6  END_VAR
7
8
9
10
11
12
13
14
15
16
17
18
19
20

```

```

3  TCP_Server_0(
4      xEnable:= ,
5      Addr:= '192.168.1.100', //PLC IP
6      uiPort:= 8181, //端口号
7      bSend:= ,
8      pSendBuf:= ADR(SendData), //发送数据地址指针,
9      szSendBuf:=SIZEOF(SendData), //发送到的数据大小
10     pRecvBuf:= ADR(RevData) , //接收到数据的地址指针,
11     szRecvBuf:= SIZEOF(RevData) , //接收到数据的大小
12     udiTimeOut:= 100,
13     xActive=> ,
14     xBusy=> ,
15     xRDone=> ,
16     xTDone=> ,
17     recvNum=> ,
18     bError=> ,
19     idError=> );
20

```

(2) 连接。PLC IP: 192.168.1.100; 电脑 IP : 192.168.1.19; PLC 做 Server, 那么电脑软件就做 Client, 电脑端使用 SSCOM 工具

Device.Application.P2

表达式	类型	值	准备值	地址	注释
TCP_Server_0	TCPServer N				功能块声明
xEnable	BOOL	TRUE			IP
Addr	STRING	'192.168.1.100'			端口号
uiPort	UINT	16#1FF5			发送
bSend	BOOL	FALSE			发送数据地址 指针
pSendBuf	POINTER TO ...	16#0000007F8759B66C			发送长度
szSendBuf	UDINT	16#00000004			接收收据地址 指针
pRecvBuf	POINTER TO ...	16#0000007F8759B5A4			接收长度
szRecvBuf	UDINT	16#00000004			超时时间 MS
udiTimeOut	UDINT	16#00000064			侦听到客户端
xActive	BOOL	TRUE			服务器运行中 TCP_Connecti
xBusy	BOOL	TRUE			

SSCOM V5.13.1 串口/网络数据调试器,作者:大虾丁丁,2618058@qq.com. QQ群: 52502449(最新版本)

通讯端口 串口设置 显示 发送 多字符串 小工具 帮助 联系作者 大虾电子网

清除窗口 打开文件 发送文件 停止 清除发送区 最前 English 保存

端口号 TCPClient 12345

远程 192.168.1.100 8181 连接

本地 192.168.1.19 8181 断开

接收数据到文件 接收数据到文件 接收数据到文件 接收数据到文件

HEX发送 定时发送: 1000 ms/次

超时时间: 20 ms 第1 字节 至 末尾 加校验 None

为了更好地发展SSCOM软件 请您注册嘉立创VIP客户

发 送

(3) PLC 接收数据。用调试工具发送数据，PLC 可以收到数据。数据在调试助手中显示是低位在左侧，Codesys 数据中低位在右。xRDone 只存在一个周期，可以用 Trace 抓取

Device.Application.P2

表达式	类型	值	准备值	地址	注释
TCP_Server_0	TCPServer_N				功能块声明
RevData	DWORD	16#34333235			接收的数据
SendData	DWORD	16#00000000			发送的数据

SSCOM V5.13.1 串口/网络数据调试器,作者:大虾丁丁,2618058@qq.com. QQ群: 52502449(最新版本)

通讯端口 串口设置 显示 发送 多字符串 小工具 帮助 联系作者 大虾电子网

[16:51:21.364]发→◇31 32 33 34 35 □

调试工具发数据

完成信号只存在一个周期，可以用 Trace 抓取



(4) PLC 发送数据。用调试工具接收来自 PLC 发送的数据。数据在调试助手中显示是低位在左侧，Codesys 数据中低位在右。xTDone 发送成功后始终为 1。



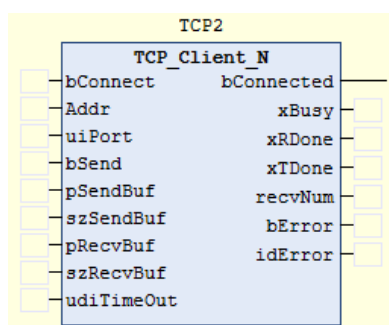
TCP_Client_N

指令名称: TCP_Client_N TCP 通讯 (PLC 做 Client) 基于 NBS.CAA 库

功能块说明: 本功能块用于 PLC 与其他设备之间做 TCP Socket 通讯 (PLC 做 Client), 并且进行数据的收发。

1、指令格式

图形表现:



ST 表现:

```
TCP2(
    bConnect:= ,
    Addr:= ,
    uiPort:= ,
    bSend:= ,
    pSendBuf:= ,
    szSendBuf:= ,
    pRecvBuf:= ,
    szRecvBuf:= ,
    udiTimeOut:= ,
    bConnected=> ,
    xBusy=> ,
    xRDone=> ,
    xTDone=> ,
    rcvNum=> ,
    bError=> ,
    idError=> );
```

2、相关变量说明

变量	名称	数据类型	输入输出属性	描述
bConnect	创建服务器	BOOL	输入变量	连接服务器
Addr	IP 地址	String	输入变量	IP
uiPort	端口号	UINT	输入变量	端口号
bSend	发送数据	BOOL	输入变量	发送
pSendBuf	发送数据的地址	POINTER TO BYTE	输入变量	发送数据的地址指针
szSendBuf	发送数据的长度	UDINT	输入变量	发送长度
pRecvBuf	接收数据的指针	POINTER TO BYTE	输入变量	接收数据的地址指针
szRecvBuf	接收数据的长度	UDINT	输入变量	接收长度
udiTimeOut	超时时间	UDINT	输入变量	超时时间 ms
bConnected	连上服务器标志	BOOL	输出变量	=TRUE 连接到服务器
xBusy	服务器正在运行	BOOL	输出变量	服务器正在运行中
xRDone	完成信号	BOOL	输出变量	侦听到数据的完成信号 (只存在一个周期)
xTDone	发送完成	BOOL	输出变量	发送数据帧完成
RecvNum	收到的数据个数	NBS.CAA.SIZE	输出变量	接收到的数据个数
bError	报错标志	BOOL	输出变量	错误标志位
idError	Error ID	NBS.ERROR	输出变量	报错 ID; 取自 NBS.Error

3.具体使用

(1) 声明与实例化

```

1 PROGRAM P2
2 VAR
3     TCP_Client_0: TCP_Client_N; //功能块声明
4     RevData:DWORD; //接收的数据
5     SendData:DWORD; //发送的数据
6 END_VAR
7
8 // 电脑网口接PLC。电脑IP: 192.168.1.19; PLC IP: 192.168.1.100
9 // 电脑使用SSCOM软件仿真
10 TCP_Client_0(
11     bConnect:= ,
12     Addr:= '192.168.1.19', //PLC IP
13     uiPort:= 8181, //端口号
14     bSend:= ,
15     pSendBuf:= ADDR(SendData), //发送数据地址指针,
16     szSendBuf:= SIZEOF(SendData), //发送到的数据大小
17     pRecvBuf:= ADDR(RevData), //接收到数据的地址指针,
18     szRecvBuf:= SIZEOF(RevData), //接收到数据的大小
19     udiTimeOut:= 100,
20     bConnected=>,
21     xBusy=>,
22     xRDone=>,
23     xTDone=>,
24     recvNum=>,
25     bError=>,
26     idError=> );

```

(2) 连接。本样例中，电脑网口接 PLC。电脑 IP：192.168.1.19；PLC IP：192.168.1.100。电脑使用 SSCOM 软件连接。

表达式	类型	值	准备值	地址	注释
TCP_Client_0	TCP_Client_N				功能块声明
bConnect	BOOL	TRUE			连接服务器
Addr	STRING	'192.168.1.19'			IP
uiPort	UINT	16#1FF5			端口号
bSend	BOOL	FALSE			发送
pSendBuf	POINTER TO ...	16#0000007F8759B66C			发送数据地址 指针
szSendBuf	UDINT	16#00000004			发送长度
pRecvBuf	POINTER TO ...	16#0000007F8759B5A4			接收数据地址 指针
szRecvBuf	UDINT	16#00000004			接收长度
udiTimeOut	UDINT	16#00000064			超时时间 MS
bConnected	BOOL	TRUE			连接到服务器
xBusy	BOOL	TRUE			客户端运行中 TCP_Connection.xBusy
xRDone	BOOL	FALSE			侦听到数据 只存在一个周期

SSCOM V5.13.1 串口/网络数据调试器,作者:大虾丁丁,2618058@qq.com. QQ群: 52502449(最新版本)

通讯端口 串口设置 显示 发送 多字符串 小工具 帮助 联系作者 大虾电子网

清除窗口 打开文件 发送文件 停止 清发送区 最前 English 保存参数 扩展

端口号 TCPServer

远程 192.168.1.100 8181 侦听

本地 192.168.1.19 8181 断开

HEX显示 保存数据 接收数据到文件 HEX发送 定时发送: 1000 ms/次 加回车换行

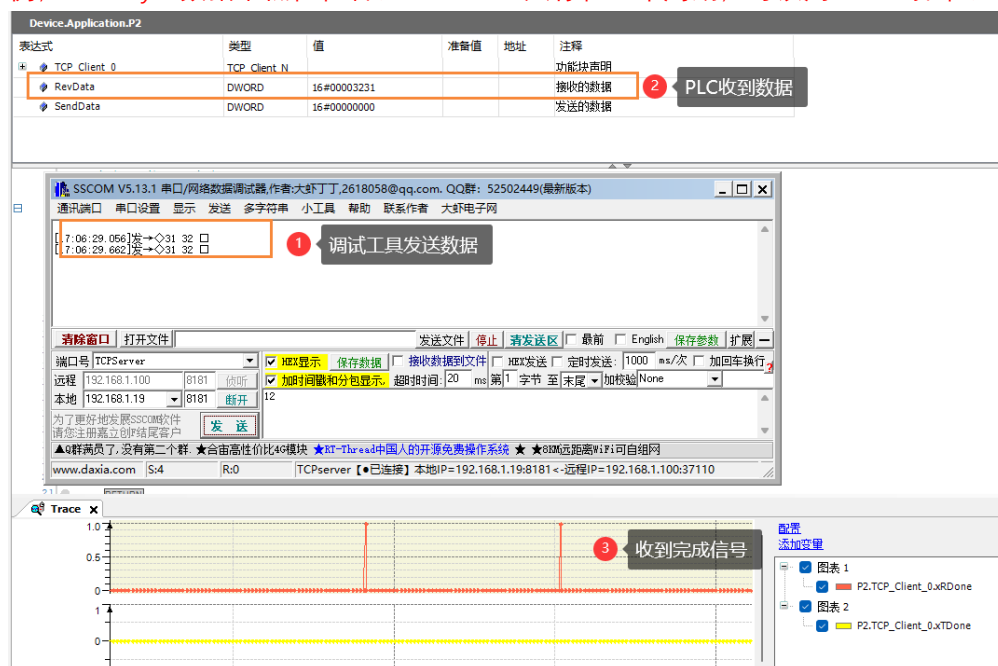
加时间戳和分包显示 超时时间: 20 ms 第1字节至末尾 加校验 None

为了更好地发展SSCOM软件 请您注册嘉立创结尾客户

▲QQ群满了,没有第二个群. ★合宙高性价比4G模块 ★RT-Thread中国人的开源免费操作系统 ★8K6远距离WiFi可自组网

www.daxia.com | S:0 | R:0 | TCPServer 【已连接】 本地IP=192.168.1.19:8181 < 远程IP=192.168.1.100:37110

(3) PLC 接收数据。用调试工具发送数据，PLC 接收。数据在调试助手中显示是低位在左侧，Codesys 数据中低位在右。xRDone 只存在一个周期，可以用 Trace 抓取。



(4) PLC 发送数据。PLC 端发送数据，调试工具接收数据。数据在调试助手中显示是低位在左侧，Codesys 数据中低位在右。xTDone 发送成功后始终为 1。



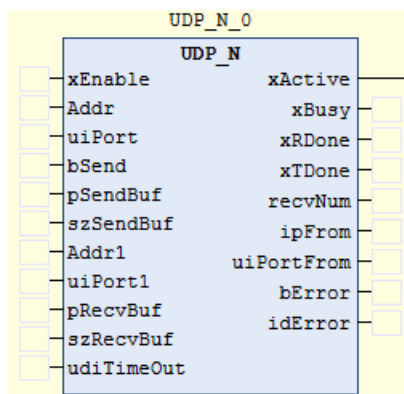
UDP_N

指令名称：UDP_N UDP 通讯，基于 NBS.CAA 库。

功能块说明：本功能块用于 PLC 与其他设备之间做 UDP 通讯，并且进行数据的收发。

1、指令格式

图形表现：



ST 表现：

```
UDP_N_0(
    xEnable:= ,
    Addr:= ,
    uiPort:= ,
    bSend:= ,
    pSendBuf:= ,
    szSendBuf:= ,
    Addr1:= ,
    uiPort1:= ,
    pRecvBuf:= ,
    szRecvBuf:= ,
    udiTimeOut:= ,
    xActive=> ,
    xBusy=> ,
    xRDone=> ,
    xTDone=> ,
    recvNum=> ,
    ipFrom=> ,
    uiPortFrom=> ,
    bError=> ,
    idError=> );
```

2、相关变量说明

变量	名称	数据类型	输入输出属性	描述
xEnable	运行	BOOL	输入变量	运行
Addr	本地 IP	String	输入变量	本地 IP
uiPort	本地端口号	UINT	输入变量	本地端口号
bSend	发送数据	BOOL	输入变量	发送
pSendBuf	发送数据的地址	POINTER TO BYTE	输入变量	发送数据的地址指针
szSendBuf	发送数据的长度	UDINT	输入变量	发送长度
Addr1	目标 IP	String	输入变量	目标 IP
uiPort1	目标端口号	UINT	输入变量	目标端口号
pRecvBuf	接收数据的指针	POINTER TO BYTE	输入变量	接收数据的地址指针
szRecvBuf	接收数据的长度	UDINT	输入变量	接收长度
udiTimeOut	超时时间	UDINT	输入变量	超时时间 ms
xActive	侦听到客户端	BOOL	输出变量	侦听到客户端
xBusy	服务器正在运行	BOOL	输出变量	服务器正在运行中
xRDone	完成信号	BOOL	输出变量	侦听到数据的完成信号（只存在一个周期）
xTDone	发送完成	BOOL	输出变量	发送数据帧完成

RecvNum	收到的数据个数	NBS.CAA.SIZE	输出变量	接收到的数据个数
ipFrom	来源 IP	String	输出变量	读取到数据包的源 IP
uiPortFrom	来源端口号	UINT	输出变量	读取到数据包的源端口号
bError	报错标志	BOOL	输出变量	错误标志位
idError	Error ID	NBS.ERROR	输出变量	报错 ID; 取自 NBS.Error

3、具体使用介绍:

(1) 声明与实例化

本样例中, 电脑网口接 PLC。电脑 IP: 192.168.1.19; PLC IP: 192.168.1.100。电脑使用 SSCOM 软件连接。声明如下所示:

```

1  PROGRAM P2
2  VAR
3      UDP_N_0: UDP_N; //UDP功能块声明
4      RevData:DWORD; //接收的数据
5      SendData:DWORD; //发送的数据
6  END_VAR
7
8  // 电脑网口接PLC。电脑IP: 192.168.1.19; PLC IP: 192.168.
9  // 电脑使用SSCOM软件仿真
10 UDP_N_0(
11     xEnable:= ,
12     Addr:= '192.168.1.100', //PLC IP
13     uiPort:= 8181, //端口号
14     bSend:= ,
15     pSendBuf:= ADR(SendData), //发送数据地址指针
16     szSendBuf:= SIZEOF(SendData), //发送到的数据大小
17     Addr1:= '192.168.1.19', //设备IP 本例程为电脑IP
18     uiPort1:= 8181, //端口号
19     pRecvBuf:=ADR(RevData), //接收到数据的地址指针
20     szRecvBuf:=SIZEOF(RevData), //接收到数据的大小
21     udiTimeOut:= 100,
22     xActive=> ,
23     xBusy=> ,
24     xRDone=> ,
25     xIDone=> ,
26     recvNum=> ,
27     ipFrom=> ,
28     uiPortFrom=> ,
29     bError=> ,
30     idError=> );

```

(2) 连接 UDP

Device.Application.P2						
表达式	类型	值	准备值	地址	注释	
UDP_N_0	UDP_N				UDP功能块声明	
xEnable	BOOL	TRUE	开启连接		运行	
Addr	STRING	'192.168.1.100'			本地IP	
uiPort	UINT	16#1FF5			本地端口号	
bSend	BOOL	TRUE			发送	
pSendBuf	POINTER TO ...	16#0000007F8759B66C			发送数据地址 指针	
szSendBuf	UDINT	16#00000004			发送长度	
Addr1	STRING	'192.168.1.19'			发送目标IP	
uiPort1	UINT	16#1FF5			发送目标端口号	
pRecvBuf	POINTER TO ...	16#0000007F8759B5A4			接收数据地址 指针	
szRecvBuf	UDINT	16#00000004			接收长度	
udiTimeOut	UDINT	16#00000064			超时时间MS	
xActive	BOOL	TRUE	连接成功		UDP通讯创建成功	
xBusy	BOOL	TRUE			Peerbusy	

调试工具的设置:

端口号	UDP		
远程	192.168.1.100	8181	连接
本地	192.168.1.19	8181	断开

(3) PLC 接收数据: 用调试工具发送数据, 能看到 RevData 变量收到数据。为了方便查看, 在显示模式中设置为数据按照 16 进制显示即可。数据在调试助手中显示是低位在左侧, Codesys 数据中低位在右。xRDone 只存在一个周期, 可以用 Trace 抓取

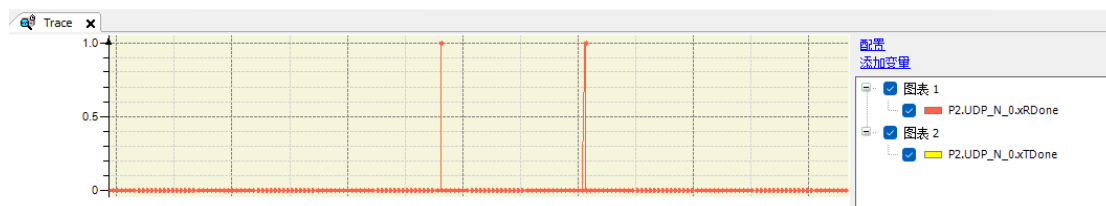
表达式	类型	值	注释
UDP_N_0	UDP_N		UDP功能块声明
RevData	DWORD	16#34333231	接收的数据
SendData	DWORD	16#000004C7	发送的数据

SSCOM V5.13.1 串口/网络数据调试器, 作者: 大虾丁, 2618058@qq.com. QQ群: 52502449(最新版本)

通讯端口 串口设置 显示 发送 多字符串 小工具 帮助 联系作者 大虾电子网

[16:24:34.199]发→◇31 32 33 34 35 □ 调试工具发数据

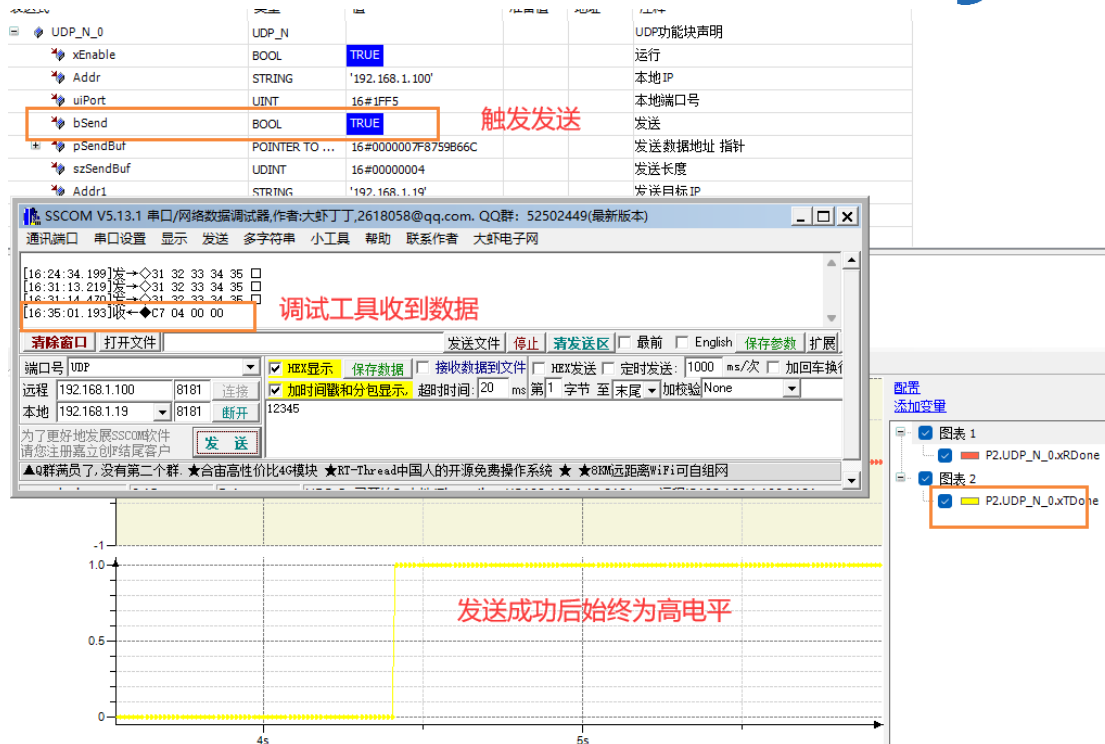
xRDone 只存在一个周期, 可以用 Trace 抓取



(4) PLC 发送数据: 用 PLC 发送数据, 能在调试工具中看到收到数据。为了方便查看, 在显示模式中设置为数据按照 16 进制显示即可。数据在调试助手中显示是低位在左侧, Codesys 数据中低位在右。xTDone 发送成功后一直为 1.

表达式	类型	值	准备值	地址	注释
UDP_N_0	UDP_N				UDP功能块声明
RevData	DWORD	16#34333231			接收的数据
SendData	DWORD	16#000004C7			发送的数据

通过 bSend 发送



06_滤波

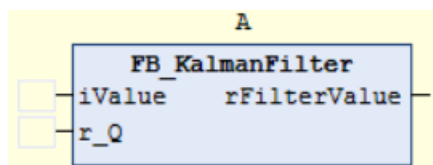
FB_KalmanFilter 卡尔曼滤波

功能块说明: 本功能块基于卡尔曼滤波原理进行实现, 可设定调控比例, 用于过滤一些杂波。例如转矩控制张力时候, 过滤一些突变的力矩值, 防止张力突变。

1、指令格式

图形表现:

ST 表现:



```

A(iValue:= ,
  r_Q:= ,
  rFilterValue=> );
  
```

2、相关变量说明

变量	名称	数据类型	输入输出属性	描述
iValue	需要过滤的数据	INT	输入变量	需要滤波的数据
r_Q	调节比例	REAL	输入变量	过滤的比例系数, 约接近 0 过滤效果越好
rFilterValue	REAL	REAL	输出变量	滤波处理后的数值

3、具体使用介绍:

如下，产生 0-10 的随机数

```

1  PROGRAM PLC_PRG
2  VAR
3      klm      :FB_KalmanFilter;
4
5      qs      :REAL;
6      FilterValue:REAL;
7
8  (* Rndi,i:UINT;
9     next:DWORD:=10;
10    iInput:UINT;*)
11
12    bintlization:BOOL;
13    aa:SM0.RndI_Range;
14    dd:BOOL;
15    Result:DINT;
16  END_VAR

```

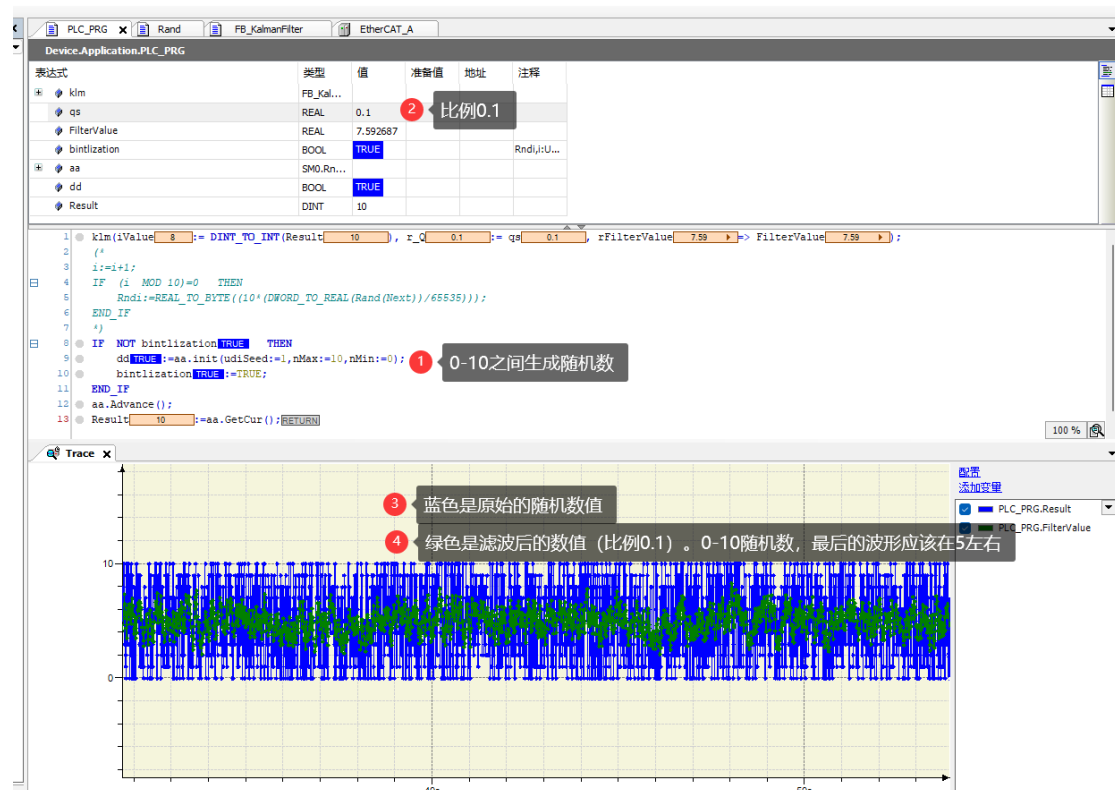
```

1  klm(iValue:= DINT_TO_INT(Result), r_Q:= qs, rFilterValue=> FilterValue);
2  (*
3  i:=i+1;
4  IF (i MOD 10)=0 THEN
5      Rndi:=REAL_TO_BYTE((10*(DWORD_TO_REAL(Rand(Next)))/65535)));
6  END_IF
7  *)
8  IF NOT bintlization THEN
9      dd:=aa.init(udiSeed:=1,nMax:=10,nMin:=0);
10     bintlization:=TRUE;
11  END_IF
12  aa.Advance();
13  Result:=aa.GetCur();

```

产生随机数

采集波形，蓝色是原始的随机数值，绿色的是滤波后的数值。当前的设定比例是 0.1



修改比例到 0.000001（越接近 0），波形产生变化，杂波过滤后越接近 5（随机数 0-10）



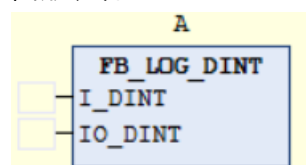
10_其他

FB_LOG_DINT 数据变化记录

功能块说明：数据记录，数据类型 DINT。当产生一个不等于数组最小下标当前值的数值的时候，把当前值存入数组。例如 Datain 当前值不等于 arData[0]的时候，存入 arData[0]；原本的数值都后移。

1、指令格式

图形表现



ST 表现

A(I_DINT:= , IO_DINT:=);

2、相关变量说明

变量名称	名称	数据类型	变量属性	描述
I_DINT	需要记录的变量	DINT	输入变量	需要记录的变量
IO_DINT	存入的数组	ARRAY[*]OF DINT	输出变量	记录存入的数组

3、具体使用：

Device.Application.P2					
表达式	类型	值	准备值	地址	注释
datain	DINT	14			
arData	ARRAY [0..10] OF DINT				
arData[0]	DINT	14			
arData[1]	DINT	13			
arData[2]	DINT	12			
arData[3]	DINT	0			
arData[4]	DINT	0			
arData[5]	DINT	0			
arData[6]	DINT	0			
arData[7]	DINT	0			
arData[8]	DINT	0			
arData[9]	DINT	0			
arData[10]	DINT	0			
logDint	FB_LOG_DINT				

1 logDint(I_DINT 14 := datain 14, IO_DINT 16#000001B0A09E0F18 := arData);RETURN

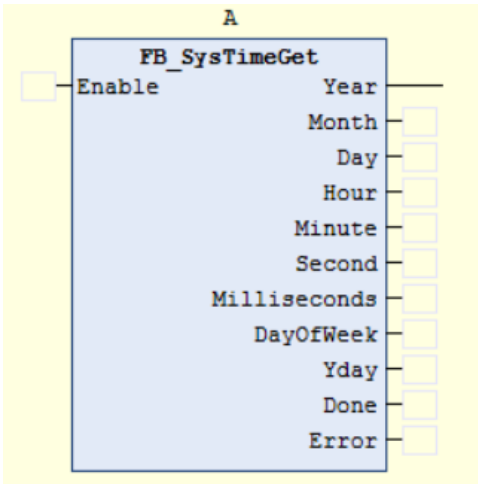
FB_SysTimeGet 时钟获取

功能块说明：获取时钟。拆成年月日时分秒毫秒，星期几，一年中的第几天。

1、指令格式

图形表现

ST 表现



```
1 PROGRAM P2
2 VAR
3
4     fbSysData:FB_SysTimeGet;
5 END_VAR
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

2、相关变量说明

变量名称	名称	数据类型	变量属性	描述
Enable	使能功能块	BOOL	输入变量	需要记录的变量
Year	年	UINT	输出变量	输出年份；例如：2020 年
Month	月	UINT	输出变量	输出月份；1-12 月
Day	日	UINT	输出变量	输出日期；例如：第 31 天
Hour	小时	UINT	输出变量	输出小时；0-23
Minute	分钟	UINT	输出变量	分钟
Second	秒	UINT	输出变量	秒
Milliseconds	毫秒	UINT	输出变量	毫秒
DayOfWeek	星期几	UINT	输出变量	星期几
YDay	一年中第几天	UINT	输出变量	一年中的第几天
Done	完成	BOOL	输出变量	功能块完成信号
Error	报错	BOOL	输出变量	功能块报错信号

3、调用举例：

Device.Application.P2				
表达式	类型	值	准备值	地址
fbSysData	FB_SysTimeGet			
Enable	BOOL	TRUE		
Year	UINT	2024		
Month	UINT	10		
Day	UINT	24		
Hour	UINT	16		
Minute	UINT	2		
Second	UINT	23		
Milliseconds	UINT	0		
DayOfWeek	UINT	4		
Yday	UINT	298		
Done	BOOL	TRUE		
Error	BOOL	FALSE		

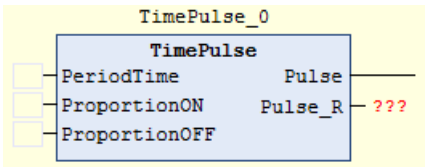
```
1
2 ● fbSysData (
3     Enable:= ,
4     Year=> ,
5     Month=> ,
6     Day=> ,
7     Hour=> ,
8     Minute=> ,
9     Second=> ,
10    Milliseconds=> ,
11    DayOfWeek=> ,
12    Yday=> ,
13    Done=> ,
14    Error=> ); RETURN
```

TimePulse 可变占空比脉冲

功能块说明：发出定时脉冲，并且可以设置占空比。

1、指令格式

图形表现



ST 表现

```
fbTimePulse(  
    PeriodTime:= ,  
    ProportionON:= ,  
    ProportionOFF:= ,  
    Pulse=> ,  
    Pulse_R=> );
```

2、相关变量说明

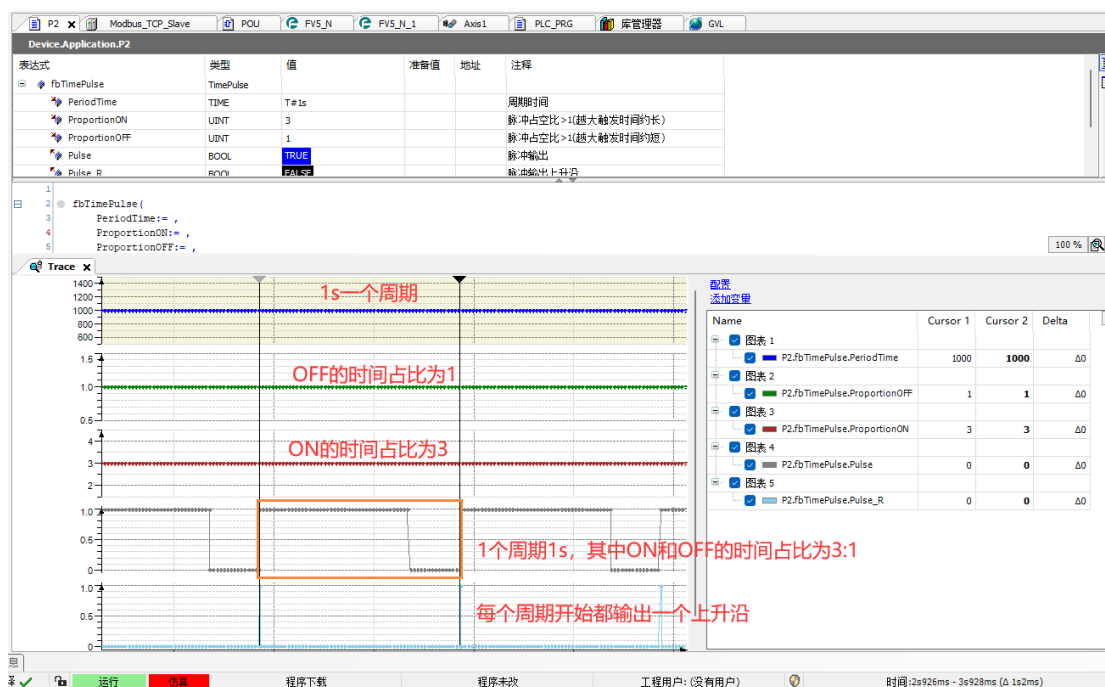
变量名称	名称	数据类型	变量属性	描述
PeriodTime	周期时间	Time	输入变量	一个周期的时间设定
ProportionON	脉冲为 ON 的占比	UINT	输入变量	脉冲为 ON 的占比
ProportionOFF	脉冲为 OFF 的占比	UINT	输入变量	脉冲为 OFF 的占比
Pulse	输出脉冲	BOOL	输出变量	脉冲输出
Pulse_R	脉冲输出的上升沿	BOOL	输出变量	脉冲输出的上升沿

3、调用举例：

(1) 声明与实例：

```
PROGRAM P2  
VAR  
    fbTimePulse:TimePulse;  
END_VAR  
  
fbTimePulse(  
    PeriodTime:= ,  
    ProportionON:= ,  
    ProportionOFF:= ,  
    Pulse=> ,  
    Pulse_R=> );
```

(2) 写值与监控



11_光伏行业

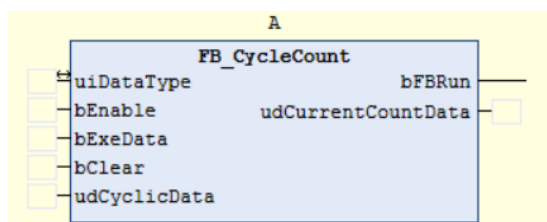
FB_CycleCount 循环计数

功能块说明：光伏行业，串焊拉丝叠片循环计数。

拉丝长度分为正常丝和串间丝，叠片位置也分为正常放片和串间放片。通过对计数和当前值的判断，输出目前是用于正常还是串间。=1 正常，=2 串间

1、指令格式

图形表现



ST 表现

```

PROGRAM P2
VAR
    fbCycleCount:FB_CycleCount;
    dataout:UINT;
END_VAR

fbCycleCount(
    uiDataType:= dataout,
    bEnable:= ,
    bExeData:= ,
    bClear:= ,
    udCyclicData:= ,
    bFRun=> ,
    udCurrentCountData=> );
    
```

2、相关变量说明

变量名称	名称	数据类型	变量属性	描述
uiDataType	输出类型	UINT	输入输出变量	=1 正常，=2 串间
bEnable	使能功能块	BOOL	输入变量	功能块开启
bExeData	计数信号	BOOL	输入变量	计数信号变量
bClear	清除	BOOL	输入变量	清除当前计数值和输出类型数据
udCyclicData	计数目标值	UDINT	输入变量	计数目标值
bFBRun	功能块运行标志	BOOL	输出变量	功能块正在运行
udCurrentCountData	当前计数值	UDINT	输出变量	bExeData 从 FALSE-TRUE，计数+1

3、调用举例：

Device.Application.P2

表达式	类型	值	准备值	地址	注释
fbCycleCount	FB_CycleCount				
uiDataType	REFERENCE TO UINT	2			
bEnable	BOOL	TRUE			功能块 ...
bExeData	BOOL	FALSE			变量触 ...
bClear	BOOL	FALSE			清空当 ...
udCyclicData	UDINT	6			目标值
bFBRun	BOOL	TRUE			FB在运 ...
udCurrentCountData	UDINT	4			当前计 ...
R_TRIG触发	R_TRIG				
dataout	UINT	2			

```

1
2  fbCycleCount(
3     uiDataType:= 2 := dataout 2 ,
4     bEnable:= ,
5     bExeData:= ,
6     bClear:= ,
7     udCyclicData:= ,
8     bFBRun=> ,
9     udCurrentCountData=> );RETURN

```

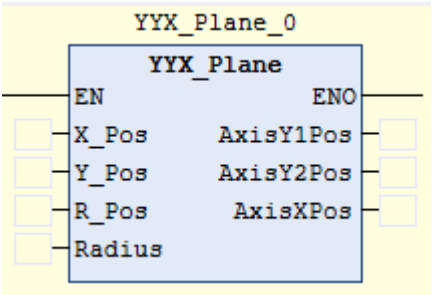
YYX_Plane -XYR 坐标变换

指令名称 YYX_Plane （XYR 坐标转换 YYX 坐标功能块）

功能块说明：本功能块适用于 YYX 轴机构和相机配合，做物料纠偏补偿等工况。将相机 XYR 坐标通过计算转换成 YYX 轴补偿值，实现相同的位置补偿。

1、指令格式

图形表现：



ST 表现：

```
YYX_Plane_0(  
    X_Pos:= ,  
    Y_Pos:= ,  
    R_Pos:= ,  
    Radius:= ,  
    AxisY1Pos=> ,  
    AxisY2Pos=> ,  
    AxisXPos=> );
```

2、相关变量

输入变量

输入变量	名称	数据类型	有效范围	描述
X_Pos	补偿 X 坐标	REAL		填写需要补偿的 X 坐标
Y_Pos	补偿 Y 坐标	REAL		填写需要补偿的 Y 坐标
R_Pos	补偿 R 坐标	REAL		填写需要补偿的 R 坐标
Radius	平台转中心半径	REAL	>0	填写实际机构平台转心半径

输出变量

输出变量	名称	数据类型	有效范围	描述
AxisY1Pos	当前 Y1 轴补偿值	REAL		将补偿值相对当前 Y1 轴位置进行定位即可
AxisY2Pos	当前 Y2 轴补偿值	REAL		将补偿值相对当前 Y2 轴位置进行定位即可
AxisXPos	当前 X 轴补偿值	REAL		将补偿值相对当前 X 轴位置进行定位即可

3、使用说明

YYX_Plane：在当前周期可计算完成，在程序上端调用，在下方读取值，进行使用。

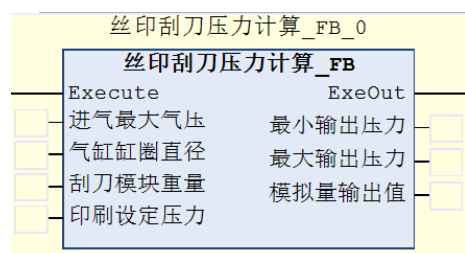
丝印刮刀压力计算_FB

指令名称：丝印刮刀压力计算_FB

功能块说明：本功能块适用于丝网印刷气缸刮刀压力计算。

1、指令格式

图形表现



ST 表现

```
丝印刮刀压力计算_FB_0(
    Execute:= ,
    进气最大气压:= ,
    气缸缸圈直径:= ,
    刮刀模块重量:= ,
    印刷设定压力:= ,
    ExeOut=> ,
    最小输出压力=> ,
    最大输出压力=> ,
    模拟量输出值=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
进气最大气压	进气最大气压	REAL	输入变量	填写刮刀连接调压阀的进气大小
气缸缸圈直径	气缸缸圈直径	REAL	输入变量	填写刮刀连接气缸的缸圈直径
刮刀模块重量	刮刀模块重量	REAL	输入变量	填写刮刀模块的重量
印刷设定压力	印刷设定压力	REAL	输入变量	填写需要设定的印刷压力
最小输出压力	最小输出压力	REAL	输出变量	输出值刮刀模块重量*9.8
最大输出压力	最大输出压力	REAL	输出变量	输出值刮刀模块重量*9.8+进气最大气压*(3.1415*气缸缸圈直径*气缸缸圈直径/4)
模拟量输出值	模拟量输出值	REAL	输出变量	输出值给到模拟量模块的调节调压阀的压力

3、使用说明

最小输出压力:=刮刀模块重量*9.8;

最大输出压力:=刮刀模块重量*9.8+进气最大气压*(3.1415*气缸缸圈直径*气缸缸圈直径/4);

当 印刷设定压力<=最小输出压力 模拟量输出值:=0;

当 印刷设定压力>=最大输出压力 模拟量输出值:=20000;

当 最小输出压力<印刷设定压力<最大输出压力 模拟量输出值:=REAL_TO_INT((印刷设定压力-最小输出压力)/(3.1415*气缸缸圈直径*气缸缸圈直径/4)*20000/进气最大气压);

02.FC

01_ArrMaxMin 求数组最大最小值

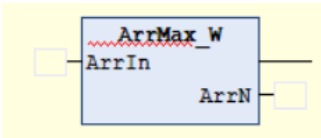
ArrMax_DW/LR/LW/R/W

函数功能：数组找最大值。输出最大值所在的数组下标和当前数组最大值是多少。若最大值相同，输出较小的数组下标。例如 arA[1]=arA[10]=arA[12]=888；ArrN 输出 1

1、指令格式

图形表现

ST 表现



```
ArrMax W(ArrIn:= , ArrN=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
ArrIn	输入数组	数组	输入变量	需要查找的数组
ArrN	最大值下标	WORD	输出变量	数组最大值所在下标

3、具体使用

```
1 PROGRAM P2
2 VAR
3     wData: ARRAY[0..100] OF WORD;
4     wMaxDataNum:WORD;
5     wMaxData:WORD;
6
7     wMiniDataNum:WORD;
8     wMiniData:WORD;
9 END_VAR
10
```

```
1 wMaxData:=ArrMax_W(ArrIn:= wData, ArrN=> wMaxDataNum);
2 wMiniData:=ArrMin_W(ArrIn:= wData, ArrN=> wMiniDataNum);
```

表达式	类型	值	准备值	地址	注释
wData	ARRAY [0..100] OF WO...				
wData[0]	WORD	100			
wData[1]	WORD	200			
wData[2]	WORD	10			
wData[3]	WORD	0			
wData[4]	WORD	0			
wData[5]	WORD	0			
wData[6]	WORD	0			
wData[7]	WORD	0			
wData[8]	WORD	0			
wData[9]	WORD	0			

```
1 wMaxData[200]:=ArrMax_W(ArrIn:= wData, ArrN=> wMaxDataNum 1);
2 wMiniData[0]:=ArrMin_W(ArrIn:= wData, ArrN=> wMiniDataNum 3);RETURN
```

4、不同的数据类型使用不同的函数。如下注解

 ArrMax_DW	_DW: DWord数组
 ArrMax_LR	_LR: LReal数组
 ArrMax_LW	_LW: LWord数组
 ArrMax_R	_R: REAL数组
 ArrMax_W	_W: Word数组

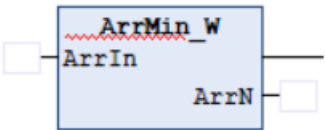
ArrMin_ DW/LR/LW/R/W

函数功能：输入任意长度的数组，找出这个数组的最小值，输出最小值所在的数组下标，返回最小值当前值。若最小值相同，输出较小的数组下标。例如 arA[1]=arA[10]=arA[12]=20, 20 是最小值；ArrN 输出 1。

1、指令格式

图形表现

ST 表现



```
ArrMin_W(ArrIn:= , ArrN=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
ArrIn	输入数组	数组	输入变量	需要查找的数组
ArrN	最大值下标	WORD	输出变量	数组最小值所在下标

3、具体使用

```

1  PROGRAM P2
2  VAR
3      wData: ARRAY[0..100] OF WORD;
4      wMaxDataNum: WORD;
5      wMaxData: WORD;
6
7      wMiniDataNum: WORD;
8      wMiniData: WORD;
9  END_VAR
10
11  wMaxData:=ArrMax W(ArrIn:= wData, ArrN=> wMaxDataNum);
12  wMiniData:=ArrMin_W(ArrIn:= wData, ArrN=> wMiniDataNum);

```

Device.Application.P2						
表达式	类型	值	准备值	地址	注释	
 wData	ARRAY [0..100] OF WO...					
 wData[0]	WORD	100				
 wData[1]	WORD	200				
 wData[2]	WORD	10				
 wData[3]	WORD	0				
 wData[4]	WORD	0				
 wData[5]	WORD	0				
 wData[6]	WORD	0				
 wData[7]	WORD	0				
 wData[8]	WORD	0				
 wData[9]	WORD	0				
1  wMaxData[200] :=ArrMax_W(ArrIn:= wData, ArrN=> wMaxDataNum[1]); 2  wMiniData[0] :=ArrMin_W(ArrIn:= wData, ArrN=> wMiniDataNum[3]); RETURN						

4、数据类型

ArrMin_DW	_DW: DWord数组
ArrMin_LR	_LR: LReal数组
ArrMin_LW	_LW: LWord数组
ArrMin_R	_R: REAL数组
ArrMin_W	_W: Word数组

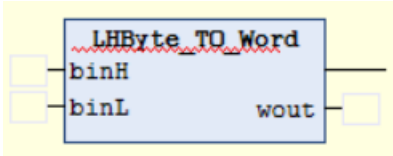
02_数据的高低位拆分和合并 Byte&Word

LHByte_To_Word

函数功能：两个 Byte 数据分别存储 Word 数据的高低位，拼成一个 WORD 输出

1、指令格式

图形表现



ST 表现

```
LHByte_TO_Word(binH:= , binL:= , wout=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
binH	高 8 位	Byte	输入变量	Byte 数据存 WORD 的高 8 位
binL	低 8 位	Byte	输入变量	Byte 数据存 WORD 的低 8 位
wout	输出的 WORD 变量	WORD	输出变量	输出 WORD 变量最终的值

3、具体使用

(1) 声明和函数调用

```
wData_Byte:WORD;  
bByte_Word_H:BYTE;  
bByte_Word_L:BYTE;  
END_VAR
```

```
LHByte_TO_Word(binH:=bByte_Word_H , binL:= bByte_Word_L, wout=> wData_Byte);
```

(2) 给值测试

wData_Byte	WORD	3073			
bByte_Word_H	BYTE	12			
bByte_Word_L	BYTE	1			

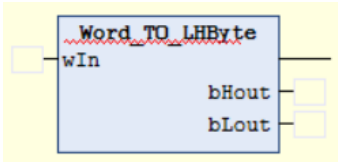
```
1  
2  
3  
4 ● LHByte_TO_Word(binH:=bByte_Word_H[12] , binL:= bByte_Word_L[1] , wout=> wData_Byte[3073] );RETURN
```

Word_To_LHByte

函数功能：输入一个 WORD 数据，按照高 8 位和低 8 位输出到两个 Byte 变量

1、指令格式

图形表现



ST 表现

```
Word_To_LHByte(wIn:= , bHout=> , bLout=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
wIn	输入 Word 的值	WORD	输入变量	输入 Word 变量
bHout	输出高 8	Byte	输出变量	把 word 变量拆分，存高 8 位
bLout	输出低 8	Byte	输出变量	把 word 变量拆分，存低 8 位

3、具体使用

(1) 声明和函数调用

```
wData_Byte:WORD;  
bByte_Word_H:BYTE;  
bByte_Word_L:BYTE;  
END_VAR  
  
Word_To_LHByte(wIn:=wData_Byte , bHout=>bByte_Word_H , bLout=>bByte_Word_L )
```

(2) 给值和输出。Word 当前值 258，输出高位是 1 低位是 2

wData_Byte	WORD	258			
bByte_Word_H	BYTE	1			
bByte_Word_L	BYTE	2			

```
1  
2  
3 Word_To_LHByte(wIn:=wData_Byte[258] , bHout=>bByte_Word_H[1] , bLout=>bByte_Word_L[2] );  
4
```

03_数据的高低位拆分和合并 RealWord

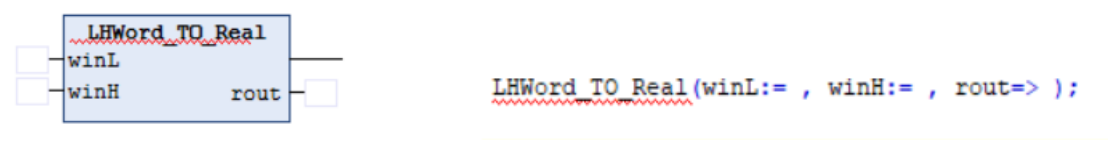
LHWord_To_Real

函数功能：按照高 16 位和低 16 位输入两个 WORD，计算后输出到 1 个 REAL 变量

1、指令格式

图形表现

ST 表现



2、相关变量

变量名称	名称	数据类型	变量属性	描述
wInL	输入低 16 位	WORD	输入变量	用 WORD 存 REAL 数据的低 16 位
wInH	输入高 16 位	WORD	输入变量	用 WORD 存 REAL 数据的高 16 位
rOut	输出 REAL	REAL	输出变量	计算后输出 REAL 当前值

3、具体使用

(1) 声明和函数调用

```
1 PROGRAM P2
2 VAR
3
4     w_H:WORD;
5     w_L:WORD;
6     rData:REAL;
7
8 END_VAR
9
10
11 LHWord_TO_Real(winL:= w_L, winH:= w_H, rout=>rData );
```

(2) 给值和输出。Word 当前值 258，输出高位是 1 低位是 2

表达式	类型	值	准备值
w_H	WORD	16105	
w_L	WORD	54757	
rData	REAL	0.45671	


```

1  LHWord_To_Real(winL:= w_L[54757], winH:= w_H[16105], rout=>rData[0.457], );RETURN

```

(3) 工具验证

二进制转十进制

输入: 0011101101001

从: 二进制

转换为: 十进制

转换结果: 16105

Float (IEEE754 Single precision 32-bit)

Decimal: 0.457

Most accurate representation = 4.569999873638153076171875E-1

Binary: 0x3EE9FBE7 = 00111110 11101001 11111011 11100111

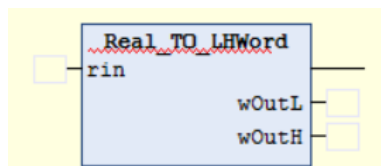
Sign: 0, Exponent: 01111101, Mantissa: 11010011111101111100111

Real_To_LHWord

函数功能：输入一个 REAL 变量，按照高 16 和低 16 位分别存入一个 WORD

1、指令格式

图形表现



ST 表现

```
Real_To_LHWord(rin:= , wOutL=> , wOutH=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
Rin	输入 REAL 值	REAL	输入变量	输入 REAL 值
wOutL	拆分出低 16 位	WORD	输出变量	用 WORD 存 REAL 的低 16 位

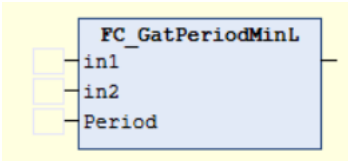
04_位移、速度、加速度计算类

FC_GatPeriodMinL

函数功能：根据模值，计算最短路径，有正负区分。例如，插补功能块里面，判断模态轴定位方向使用

1、指令格式

图形表现



ST 表现

```
FC_GatPeriodMinL(in1:= , in2:= , Period:= );|
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
In1	输入位置值 1	LREAL	输入变量	输入位置值 1
In2	输入位置值 2	LREAL	输入变量	输入位置值 2
Period	模值	LREAL	输入变量	模值
FC_GatPeriodMinL	最短距离	LREAL	输出变量	计算后的最短距离

3、具体使用

(1) 声明和函数调用

```
1 PROGRAM P2
2 VAR
3   lrData1:LREAL;
4   lrData2:LREAL;
5   lrPeriod:LREAL;
6
7   lrPeriodMinL:LREAL;
8 END_VAR
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
1
2
3
4 lrPeriodMinL:=FC_GatPeriodMinL(in1:=lrData1 , in2:= lrData2, Period:=lrPeriod );
```

(2) 给值与计算


```

1  PROGRAM P2
2  VAR
3      lrData1:LREAL;
4      lrData2:LREAL;
5      lrDistance:LREAL;
6
7      lrAcc:LREAL;
8  END_VAR
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

(2) 写值与监控

Device.Application.P2						
表达式	类型	值	准备值	地址	注释	
lrData1	LREAL	1				
lrData2	LREAL	10				
lrDistance	LREAL	20				
lrAcc	LREAL	2.475				

备注：

匀加速直线位移公式： $X = (V1 * V1 - V0 * V0) / (2 * a)$

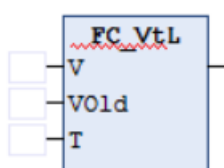
FC_VtL

函数功能：根据加速度公式，计算一段时间内的位移量

1、指令格式

图形表现

ST 表现



```
FC_VtL(V:= , V0ld:= , T:= );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
V	初始速度	LREAL	输入变量	V0 初始速度值
Vold	最终速度	LREAL	输入变量	Vt 最终速度值
T	时间	LREAL	输入变量	时间
FC_VtL	X 位移量	LREAL	输出变量	计算得到的距离

3、具体使用

(1) 声明和函数调用

```

1  PROGRAM P2
2  VAR
3      lrData1:LREAL;
4      lrData2:LREAL;
5      lrDistance:LREAL;
6
7      lrAcc:LREAL;
8  END_VAR
9
1
2  lrDistance:=FC_VtL(V:=lrData1 , Vold:= lrData2, T:= 0.1);

```

(2) 写值与监控

Device.Application.P2					
表达式	类型	值	准备值	地址	注释
lrData1	LREAL	100			
lrData2	LREAL	0			
lrDistance	LREAL	5			
lrAcc	LREAL	0			

```

1
2  lrDistance[5]:=FC_VtL(V:=lrData1[100] , Vold:= lrData2[0] , T:= 0.1);RETURN

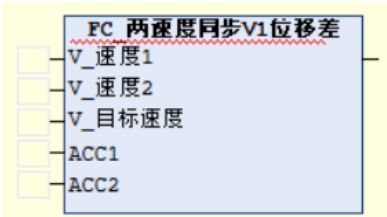
```

FC_两速度同步 V1 位移差

函数功能：例如 A 轴用 V 速度 1 和 ACC1（加速度 1）加速到目标速度，B 用 V 速度 2 和 ACC2（加速度 2）运动到目标速度，两者之间产生的位移差。可用于插补位移计算。

1、指令格式

图形表现



ST 表现

```
FC_两速度同步V1位移差(  
  V_速度1:= ,  
  V_速度2:= ,  
  V_目标速度:= ,  
  ACC1:= ,  
  ACC2:= );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
V_速度 1	速度 1	REAL	输入变量	V1
V_速度 2	速度 2	REAL	输入变量	V2
V_目标速度	时间	REAL	输入变量	目标速度
ACC1	加速度 1	REAL	输入变量	加速度 1（和 V1 一起计算）
ACC2	加速度 2	REAL	输入变量	加速度 2（和 V2 一起计算）
FC_两速度同步 V1 位移差	位移差	REAL	输出变量	计算两组 V 和 a 运动到目标速度时候产生的位移差

3、具体使用

(1) 声明和函数调用

```
1 PROGRAM P2  
2 VAR  
3   rData1:REAL;  
4   rData2:REAL;  
5  
6   rTargetV:REAL;  
7  
8   rAcc1:REAL;  
9   rAcc2:REAL;  
10  
11  rL2_L1:REAL;  
12 END_VAR  
13  
  
1 rL2_L1:=  
2 FC_两速度同步V1位移差(  
3   V_速度1:= rData1,  
4   V_速度2:= rData2,  
5   V_目标速度:= rTargetV,  
6   ACC1:=rAcc1 ,  
7   ACC2:= rAcc2);  
8  
9
```

(2) 写值与监控

Device.Application.P2

表达式	类型	值	准备值	地址	注释
rData1	REAL	10			
rData2	REAL	100			
rTargetV	REAL	10000			
rAcc1	REAL	10			
rAcc2	REAL	10			
rL2_L1	REAL	-89505			

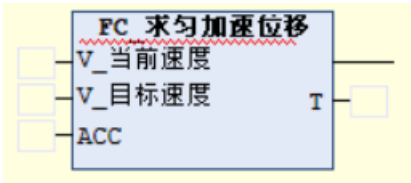
```
1 rL2_L1 := -8.95E+04
2 FC_两速度同步V1位移差(
3   V_速度1:= rData1 10,
4   V_速度2:= rData2 100,
5   V_目标速度:= rTargetV 1E+04,
6   ACC1:=rAcc1 10,
7   ACC2:= rAcc2 10);
```

FC_求匀加速位移

函数功能：根据当前速度和目标速度，以及加速度。求得匀加速段的位移，以及需要多少时间匀加速结束

1、指令格式

图形表现：



ST 表现：

FC_求匀加速位移(V_当前速度:= , V_目标速度:= , ACC:= , T=>);

2、相关变量

变量名称	名称	数据类型	变量属性	描述
V_当前速度	当前速度值	REAL	输入变量	当前速度值 u/s
V_目标速度	目标速度值	REAL	输入变量	需要加速到的目标速度值 u/s
ACC	加速度	REAL	输入变量	加速度 u/s²
T	时间	REAL	输出变量	时间 s
FC_求匀加速位移	返回匀加速段的位移	REAL	输出变量	匀加速过程中的位移量 u

3、具体使用

(1) 声明和函数调用

```
PROGRAM P2
VAR
    rData1 :REAL;
    rTargetV:REAL;
    rAcc1   :REAL;
    rS      :REAL;
    tTime   :REAL;
END_VAR

rS:=FC_求匀加速位移(V_当前速度:= rData1, V_目标速度:= rTargetV, ACC:= rAcc1, T=> tTime);
```

(2) 写值与监控（从 10 用 100000 的加速度加速到 1000，计算得需要 0.0099s，匀加速段位移是 4.9995）

Device.Application.P2

表达式	类型	值	准备值	地址	注释
rData1	REAL	10			
rTargetV	REAL	1000			
rAcc1	REAL	100000			
rS	REAL	4.9995			
tTime	REAL	0.0099			

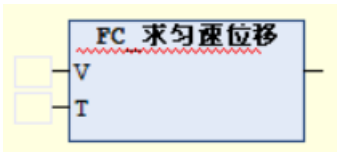
1
2
3 rS := FC_求匀加速位移(V_当前速度:= rData1, V_目标速度:= rTargetV, ACC:= rAcc1, T=> tTime); RETURN

FC_求匀加速段位移

函数功能：根据匀加速段的速度值和时间，计算匀加速段的位移量

1、指令格式

图形表现：



ST 表现：

FC_求匀加速位移(V:= , T:=);

2、相关变量

变量名称	名称	数据类型	变量属性	描述
V	匀速速度值	REAL	输入变量	匀速速度值
T	匀速运动的时间	REAL	输入变量	匀速运动的时间
FC_求匀速位移	匀速运动的位移	REAL	输出变量	计算后的位移量

3、具体使用

(1) 声明和函数调用

```
1 PROGRAM P2
2 VAR
3
4     rV:REAL;
5     rT:REAL;
6     rMove:REAL;
7 END_VAR
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

(2) 写值与监控：匀速段的 $S=V \times T$

rV	REAL	10.1
rT	REAL	0.5
rMove	REAL	5.05

```
1
2
3 rMove := FC_求匀速位移 (V:= rV, T:= rT);
4
5
```

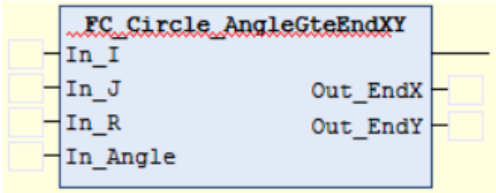
05_圆弧相关

FC_Circle_AngleGteEndXY

函数功能：已知圆心， 角度， 半径； 计算圆弧结束点坐标。 水平第一象限为 0 度

1、指令格式

图形表现：



ST 表现：

```
FC_Circle_AngleGteEndXY(In_I:= ,
                        In_J:= ,
                        In_R:= ,
                        In_Angle:= ,
                        Out_EndX=> ,
                        Out_EndY=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
In_I	圆心的 X 坐标	LREAL	输入变量	圆心 X 坐标
In_J	圆心的 Y 坐标	LREAL	输入变量	圆心 Y 坐标
In_R	半径	LREAL	输入变量	半径
In_Angle	夹角	LREAL	输入变量	夹角
Out_EndX	输出圆弧终点 X 坐标	LREAL	输出变量	计算后的圆弧终点 X 坐标
Out_EndY	输出圆弧终点 Y 坐标	LREAL	输出变量	计算后的圆弧终点 Y 坐标

3、具体使用

(1) 声明和函数调用

```
1 PROGRAM P2
2 VAR
3     lrR :LREAL;
4     lrX :LREAL;
5     lrY :LREAL;
6     lrAngle :LREAL;
7
8     lr_EndX :LREAL;
9     lr_EndY :LREAL;
10 END_VAR
11
12
13 FC_Circle_AngleGteEndXY(In_I:=lrX ,
14                        In_J:= lrY,
15                        In_R:=lrR ,
16                        In_Angle:=lrAngle ,
17                        Out_EndX=>lr_EndX ,
18                        Out_EndY=>lr_EndY );
```

(2) 写值与监控

表达式	类型	值	准备值
lrR	LREAL	30	
lrX	LREAL	10	
lrY	LREAL	10	
lrAngle	LREAL	180	
lr_EndX	LREAL	-20	
lr_EndY	LREAL	10.000000000000004	

1

2

3

4

5

6

FC_Circle_AngleGteEndXY(In_I:=lrX10, In_J:= lrY10, In_R:=lrR30, In_Angle:=lrAngle180, Out_EndX=>lr_EndX-20, Out_EndY=>lr_EndY10);RETURN

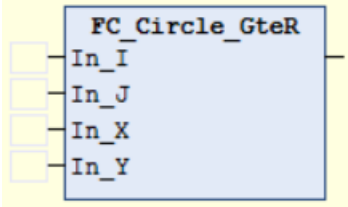
FC_Circle_GteR

函数功能：已知圆心，圆弧的坐标；计算圆的半径

1、指令格式

图形表现：

ST 表现：



```
FC_Circle_GteR(In_I:= , In_J:= , In_X:= , In_Y:= );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
In_I	圆心的 X 坐标	LREAL	输入变量	圆心的 X 坐标
In_J	圆心的 Y 坐标	LREAL	输入变量	圆心的 Y 坐标
In_X	圆弧的 X 坐标	LREAL	输入变量	圆弧的 X 坐标
In_Y	圆弧的 Y 坐标	LREAL	输入变量	圆弧的 Y 坐标
FC_Circle_GteR	输出 R	LREAL	输出变量	计算得到的圆的半径

3、具体使用

(1) 声明和函数调用

```

1 PROGRAM P2
2 VAR
3   lrR :LREAL;
4   lrX :LREAL;
5   lrY :LREAL;
6   lrAngle :LREAL;
7
8   lr_EndX :LREAL;
9   lr_EndY :LREAL;
10 END_VAR
11
12
13 lrR:=FC_Circle_GteR(In_I:= lrX, In_J:=lrY , In_X:=lr_EndX , In_Y:=lr_EndY );

```

(2) 写值与监控

表达式	类型	值	准备值	地址	注释
lrR	LREAL	56.568542494923804			
lrX	LREAL	10			
lrY	LREAL	10			
lrAngle	LREAL	0			
lr_EndX	LREAL	50			
lr_EndY	LREAL	50			

```

1
2
3 lrR[56.6] :=FC_Circle_GteR(In_I:= lrX[10] , In_J:=lrY[10] , In_X:=lr_EndX[50] , In_Y:=lr_EndY[50] );RETURN

```

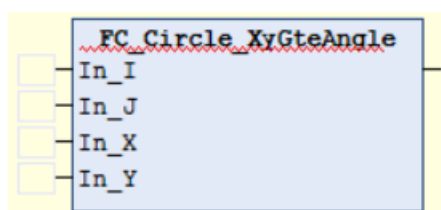
FC_Circle_XyGteAngle

函数功能：已知，起始点、终点、求圆弧夹角。水平第一象限为 0 度

1、指令格式

图形表现：

ST 表现：



```

FC_Circle_XyGteAngle(In_I:= ,
                      In_J:= ,
                      In_X:= ,
                      In_Y:= );

```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
In_I	圆心的 X 坐标	LREAL	输入变量	圆心的 X 坐标
In_J	圆心的 Y 坐标	LREAL	输入变量	圆心的 Y 坐标
In_X	圆弧的 X 坐标	LREAL	输入变量	圆弧的 X 坐标
In_Y	圆弧的 Y 坐标	LREAL	输入变量	圆弧的 Y 坐标
FC_Circle_XyGteAngle	输出夹角	LREAL	输出变量	计算得到的圆的夹角

3、具体使用

(1) 声明和函数调用

GVL

FV5_N

Axis1

PLC_PRG

FV5_N_1

LW1016

LocalBus

P2 x

1

PROGRAM P2

2

VAR

3

lrR :LREAL;

4

lrX :LREAL;

5

lrY :LREAL;

6

lrAngle :LREAL;

7

8

lr_EndX :LREAL;

9

lr_EndY :LREAL;

10

END_VAR

11

1

2

lrAngle:=FC_Circle_XyGteAngle(In_I:= lrX, In_J:=lrY , In_X:=lr_EndX , In_Y:=lr_EndY);

(2) 写值与监控

Device:Application:P2

表达式	类型	值	准备值	地址	注释
lrR	LREAL	0			
lrX	LREAL	10			
lrY	LREAL	10			
lrAngle	LREAL	45			
lr_EndX	LREAL	50			
lr_EndY	LREAL	50			

1

2

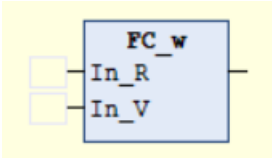
lrAngle 45 :=FC_Circle_XyGteAngle(In_I:= lrX 10 , In_J:=lrY 10 , In_X:=lr_EndX 50 , In_Y:=lr_EndY 50);RETURN

FC_w

函数功能：已知线速度和半径， 计算角速度

1、指令格式

图形表现：



ST 表现：

```
FC Circle_XyGteAngle(In_I:= ,  
                    In_J:= ,  
                    In_X:= ,  
                    In_Y:= );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
In_R	半径	LREAL	输入变量	输入半径
In_V	线速度	LREAL	输入变量	输入线速度
FC_w	角速度	LREAL	输出变量	计算后的角速度

3、具体使用

(1) 声明和函数调用

```
PROGRAM P2  
VAR  
  
    lrR:LREAL;  
    lrV:LREAL ;  
    lrW:LREAL;  
END_VAR  
  
lrW:= FC_w(In_R:= lrR, In_V:= lrV);
```

(2) 写值与调用（半径 30，线速度 2u/s，角速度为 0.066667）

表达式	类型	值
lrR	LREAL	30
lrV	LREAL	2
lrW	LREAL	0.066666666666666666

1

2

lrW 0.0667 := FC_w(In_R:= lrR 30, In_V:= lrV 2);

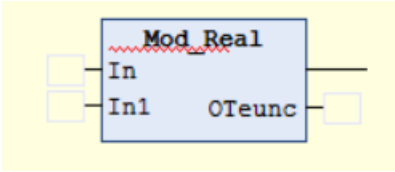
10_其他

Mod_Real

函数功能：对 Real 数据进行取余数和取整运算

1、指令格式

图形表现：ST 表现：



```
Mod_Real(In:= , In1:= , OTeunc=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
In	被除数	LREAL	输入变量	被除数值
In1	除数	LREAL	输入变量	除数值
OTeunc	商取整	DINT	输出变量	商的整数部分
Mod_Real	余数	REAL	输出变量	余数值

3、具体使用

(1) 声明和函数调用

GVL

FV5_N

Axis1

PLC_PRG

FV5_N_1

LW1016

```
1 PROGRAM P2
2 VAR
3     lrR :REAL;
4     lrX :LREAL;
5     lrY :LREAL;
6     TruncData:DINT;
7
8 END_VAR
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

(2) 写值与调用

Device.Application.P2

表达式	类型	值	准备值	地址	注释
lrR	REAL	1			
lrX	LREAL	33			
lrY	LREAL	2			
TruncData	DINT	16			

```
1
2 lrR 1 := Mod_Real(In:= lrX 33, Inl:= lrY 2, OTeunc=> TruncData 16);RETURN
```

FUN

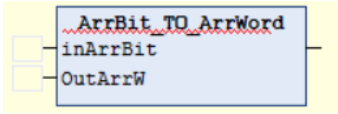
数据类型转换—Arr

ArrBit_To_ArrWord

函数功能：数组形式输入 bool，16 个 BOOL 组成一个 Word，存入一个 Word 数组

1、指令格式

图形表现：



ST 表现：

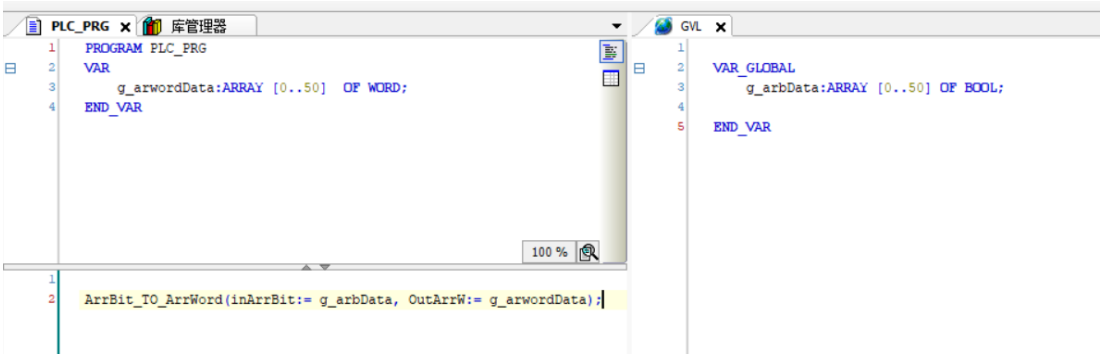
```
ArrBit_TO_ArrWord(inArrBit:= , OutArrW:= );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
inArrBit	Bit 数组	ARRAY[*] OF BOOL	输入变量	Bit 数组，一个 Word 对应 16 个 bit
OutArrW	Word 数组	ARRAY[*] OF WORD	输出变量	Word 数据数组

3、具体使用

(1) 声明和函数调用



(2) 写值与监控（BOOL 数组的 0/1/2 和 16/17/18 均写值为 TRUE，转换成 Word 为 7）

g_arwordData	ARRAY [0..50] OF WORD	
g_arwordData[0]	WORD	7
g_arwordData[1]	WORD	7
g_arwordData[2]	WORD	0
g_arwordData[3]	WORD	0

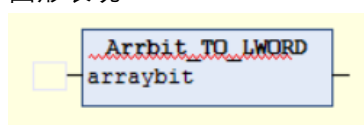
Device.Application.GVL		
表达式	类型	值
g_arbData	ARRAY [0..50] OF BOOL	
g_arbData[0]	BOOL	TRUE
g_arbData[1]	BOOL	TRUE
g_arbData[2]	BOOL	TRUE
g_arbData[3]	BOOL	FALSE
g_arbData[4]	BOOL	FALSE
g_arbData[5]	BOOL	FALSE
g_arbData[6]	BOOL	FALSE
g_arbData[7]	BOOL	FALSE
g_arbData[8]	BOOL	FALSE
g_arbData[9]	BOOL	FALSE
g_arbData[10]	BOOL	FALSE
g_arbData[11]	BOOL	FALSE
g_arbData[12]	BOOL	FALSE
g_arbData[13]	BOOL	FALSE
g_arbData[14]	BOOL	FALSE
g_arbData[15]	BOOL	FALSE
g_arbData[16]	BOOL	TRUE
g_arbData[17]	BOOL	TRUE
g_arbData[18]	BOOL	TRUE
g_arbData[19]	BOOL	FALSE
g_arbData[20]	BOOL	FALSE
g_arbData[21]	BOOL	FALSE
g_arbData[22]	BOOL	FALSE
g_arbData[23]	BOOL	FALSE
g_arbData[24]	BOOL	FALSE
g_arbData[25]	BOOL	FALSE
g_arbData[26]	BOOL	FALSE
g_arbData[27]	BOOL	FALSE
g_arbData[28]	BOOL	FALSE
g_arbData[29]	BOOL	FALSE
g_arbData[30]	BOOL	FALSE
g_arbData[31]	BOOL	FALSE
g_arbData[32]	BOOL	FALSE
g_arbData[33]	BOOL	FALSE
g_arbData[34]	BOOL	FALSE
g_arbData[35]	BOOL	FALSE
g_arbData[36]	BOOL	FALSE
g_arbData[37]	BOOL	FALSE
g_arbData[38]	BOOL	FALSE
g_arbData[39]	BOOL	FALSE

ArrBit_To_LWord

函数功能：输入不定长的 BOOL 数组，根据每一位的值，存入一个 LWord 类型数据

1、指令格式

图形表现：



ST 表现：

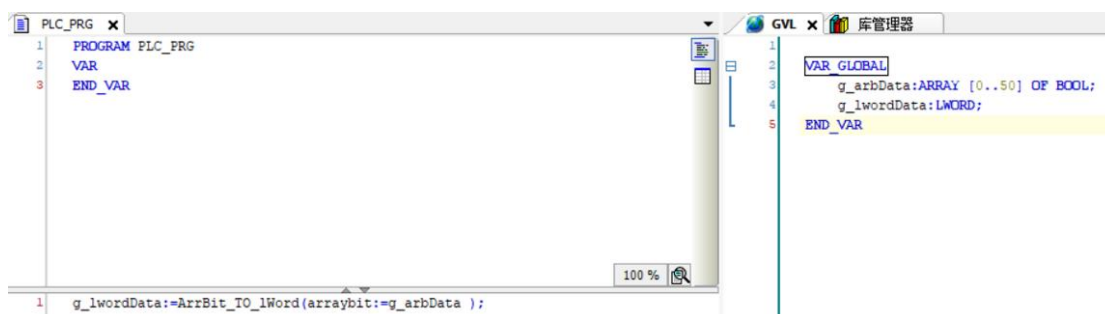
```
Arrbit_To_LWord(arraybit:= );
```

2、相关变量

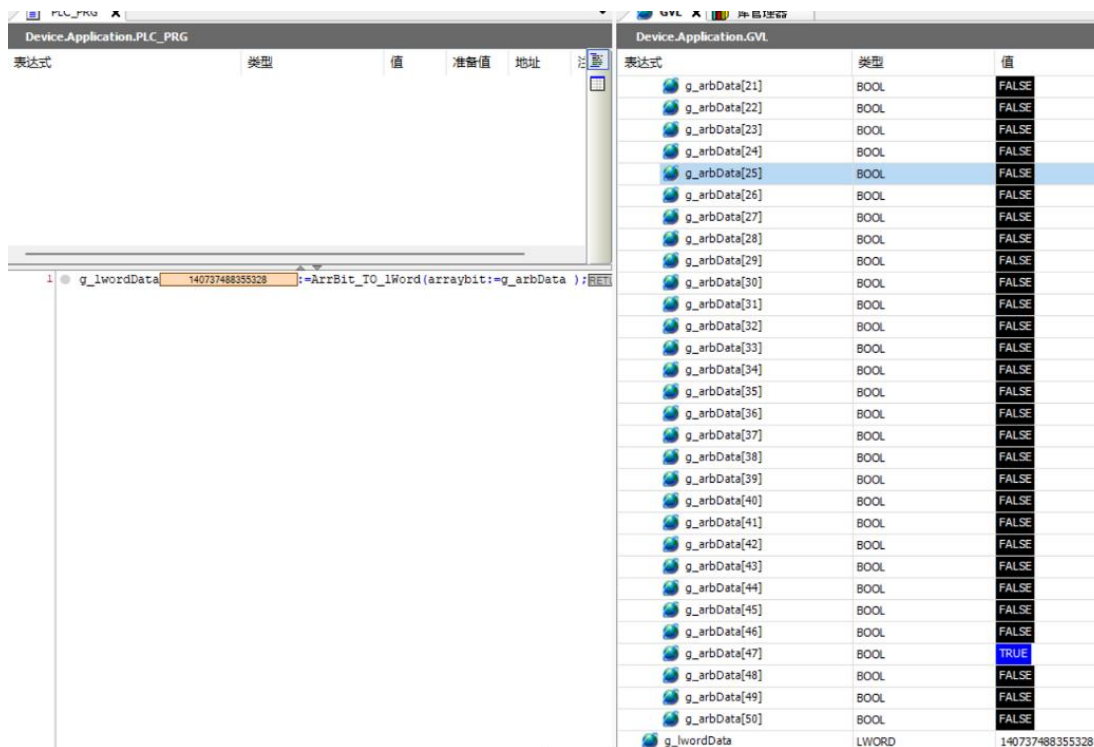
变量名称	名称	数据类型	变量属性	描述
inArrBit	Bit 数组	ARRAY[*] OF BOOL	输入变量	用 Bit 数组来描述一个 LWord 的每一位
Arrbit_To_LWORD	LWORD	LWORD	输出变量	LWord 数据

3、具体使用

(1) 声明和函数调用



(2) 写值与监控

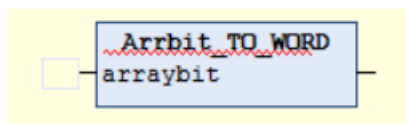


Arrbit_To_Word

函数功能：输入不定长的 BOOL 数组，根据每一位的值，存入一个 Word 类型数据

1、指令格式

图形表现：



ST 表现：

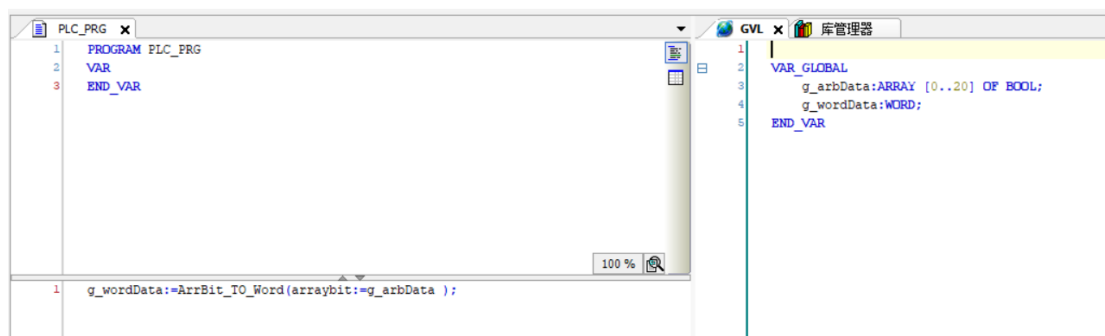
```
Arrbit_To_Word(arraybit:= );
```

2、相关变量

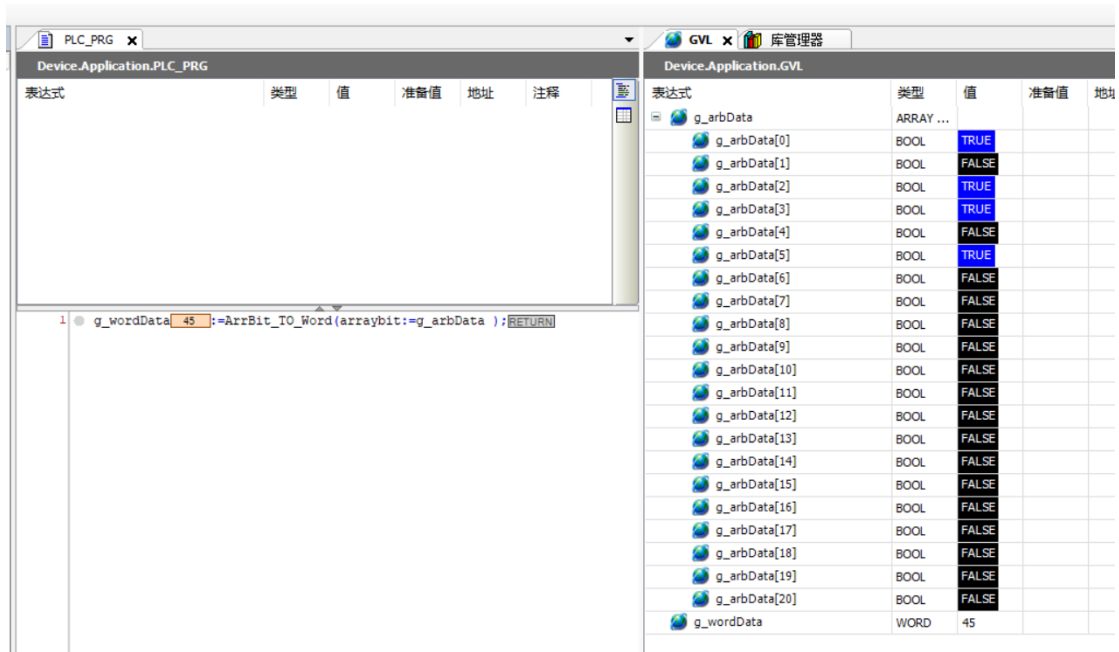
变量名称	名称	数据类型	变量属性	描述
inArrBit	Bit 数组	ARRAY[*] OF BOOL	输入变量	用 Bit 数组来描述一个 LWord 的每一位
Arrbit_To_Word	WORD 数据	WORD	输出变量	Word 数据

3、具体使用

(1) 声明和函数调用



(2) 写值与调用 (45=2#0010 1101)

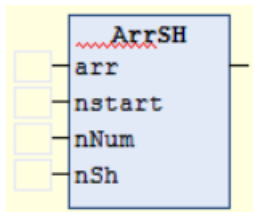


ArrSH

函数功能：数组数据的左移或者右移；设定从哪个位置开始移动 一共要移动多少个；左右移动，移动间距都通过 D_nSh 调整

1、指令格式

图形表现：



ST 表现：

```
ArrSH(  
  arr:= ,  
  nstart:= ,  
  nNum:= ,  
  nSh:= );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
Arr	Word 数组数据	ARRAY[*] OF Word	输入输出变量	Word 数组数据
nStart	移动开始的位置	UINT	输入变量	移动开始的位置
nNum	移动个数	UINT	输入变量	移动的个数
nSh	移动的步距	INT	输入变量	移动步距，大于 0 正向移动，小于 0 反向移动

3、具体使用

(1) 声明和函数调用

```

1 PROGRAM PLC_PRG
2 VAR
3     D_ArrSH:ARRAY[0..99] OF WORD;
4     D_nstart:UINT:=0;    //移动开始位置
5     D_nNum:UINT:=100;    //移动个数
6     D_nSh:INT:=1;        //移动步距 ,>0正向移动 小于0反向移动
7
8 END_VAR

1
2 ArrSH(arr:=D_ArrSH , nstart:=D_nstart , nNum:=D_nNum , nSh:=D_nSh );

```

(2) 写值与调用

表达式	类型	值	准备值	地址	注释
D_ArrSH	ARRAY [0..99] OF WORD				
D_ArrSH[0]	WORD	2097			
D_ArrSH[1]	WORD	2096			
D_ArrSH[2]	WORD	2095			
D_ArrSH[3]	WORD	2094			
D_ArrSH[4]	WORD	2093			
D_ArrSH[5]	WORD	2092			
D_ArrSH[6]	WORD	2091			
D_ArrSH[7]	WORD	2090			
D_ArrSH[8]	WORD	2089			
D_ArrSH[9]	WORD	2088			
D_ArrSH[10]	WORD	2087			
D_ArrSH[11]	WORD	2086			
D_ArrSH[12]	WORD	2085			
D_ArrSH[13]	WORD	2084			
D_ArrSH[14]	WORD	2083			
D_ArrSH[15]	WORD	2082			
D_ArrSH[16]	WORD	2081			
D_ArrSH[17]	WORD	2080			
D_ArrSH[18]	WORD	2079			
D_ArrSH[19]	WORD	2078			
D_ArrSH[20]	WORD	2077			

数据在数组中产生移动

```

1
2
3 D_T(IN TRUE:=NOT D_T.QFALSE , PT T#500ms :=T#500MS , Q=> , ET=> );
4 D_RT(CLKFALSE:= D_T.QFALSE , Q=> );
5 IF D_ISB0FALSE THEN
6     D_ISB0FALSE:=FALSE;
7     FOR D_I[2097]:=0 TO 99 BY 1 DO
8         D_ArrSH[D_I[2097]] [ ??? ]:=D_I[2097];
9     END_FOR
10  ELSIF D_ISB1FALSE OR D_RT.QFALSE THEN//右移
11      D_ISB1FALSE:=FALSE; 设置移动范围。从0开始, 移动100个
12      ArrSH(arr:=D_ArrSH , nstart:=D_nstart 0 , nNum:=D_nNum 100 , nSh:=D_nSh 1 );
13      D_I[2097]:=D_I[2097]+1;
14      D_ArrSH[0][2097]:=D_I[2097];
15      //D_ArrSH[1]:=D_I MOD 2;
16  ELSIF D_ISB2FALSE THEN//左移
17      D_ISB2FALSE:=FALSE;
18      ArrSH(arr:=D_ArrSH , nstart:=D_nstart 0 , nNum:=D_nNum 100 , nSh:=D_nSh 1 *-1 );

```

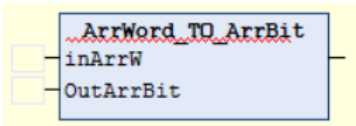
调节是左移还是右移

ArrWord_To_ArrBit

函数功能：Word 数组数据，按位处理转换成 BOOL 数组数据

1、指令格式

图形表现：ST 表现：



```
ArrWord_To_ArrBit(inArrW:= , OutArrBit:= );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
inArrW	Word 数组数据	ARRAY[*] OF Word	输入输出变量	Word 数组数据
OutArrBit	转换后的 BOOL 数组	ARRAY[*] OF BOOL	输入输出变量	BOOL 数组数据

3、具体使用

(1) 声明和函数调用

```
PROGRAM P2
VAR
    arW :ARRAY[0..3] OF WORD;
    arB :ARRAY[0..100] OF BOOL;
END_VAR
```

```
ArrWord_To_ArrBit(inArrW:= arW, OutArrBit:= arB);
```

(2) 写值与监控

A_inword4	ARRAY [0..3] OF WORD	
A_inword4[0]	WORD	2
A_inword4[1]	WORD	255
A_inword4[2]	WORD	65535
A_inword4[3]	WORD	0

A_outarrbit64	ARRAY [0..63] OF BOOL		
A_outarrbit64[0]	BOOL	FALSE	
A_outarrbit64[1]	BOOL	TRUE	
A_outarrbit64[2]	BOOL	FALSE	
A_outarrbit64[3]	BOOL	FALSE	
A_outarrbit64[4]	BOOL	FALSE	
A_outarrbit64[5]	BOOL	FALSE	
A_outarrbit64[6]	BOOL	FALSE	
A_outarrbit64[7]	BOOL	FALSE	
A_outarrbit64[8]	BOOL	FALSE	
A_outarrbit64[9]	BOOL	FALSE	
A_outarrbit64[10]	BOOL	FALSE	
A_outarrbit64[11]	BOOL	FALSE	
A_outarrbit64[12]	BOOL	FALSE	
A_outarrbit64[13]	BOOL	FALSE	
A_outarrbit64[14]	BOOL	FALSE	
A_outarrbit64[15]	BOOL	FALSE	
A_outarrbit64[16]	BOOL	TRUE	
A_outarrbit64[17]	BOOL	TRUE	
A_outarrbit64[18]	BOOL	TRUE	
A_outarrbit64[19]	BOOL	TRUE	
A_outarrbit64[20]	BOOL	TRUE	
A_outarrbit64[21]	BOOL	TRUE	
A_outarrbit64[22]	BOOL	TRUE	
A_outarrbit64[23]	BOOL	TRUE	
A_outarrbit64[24]	BOOL	FALSE	
A_outarrbit64[25]	BOOL	FALSE	
A_outarrbit64[26]	BOOL	FALSE	
A_outarrbit64[27]	BOOL	FALSE	
A_outarrbit64[28]	BOOL	FALSE	
A_outarrbit64[29]	BOOL	FALSE	
A_outarrbit64[30]	BOOL	FALSE	
A_outarrbit64[31]	BOOL	FALSE	

ArrW_0

函数功能：数组清零

1、指令格式

图形表现：ST 表现：



2、相关变量

变量名称	名称	数据类型	变量属性	描述
ArrIn	数组	ARRAY[*] OF Word	输入输出变量	需要被清零的 Word 数组数据

3、具体使用

(1) 声明和函数调用

```
1 PROGRAM P2
2 VAR
3
4     arW :ARRAY[0..3] OF WORD;
5
6 END_VAR
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

(2) 写值与监控

表达式	类型	值
arW	ARRAY [0..3] ...	
arW[0]	WORD	0
arW[1]	WORD	0
arW[2]	WORD	0
arW[3]	WORD	0

```
1
2 ● ArrW_0(ArrIn:= arW);
```

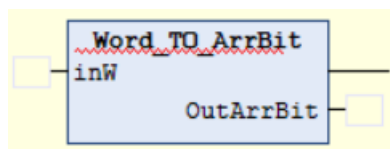
Word_To_ArrBit

函数功能：Word 变量转换成 16 个 BOOL 存入数组

1、指令格式

图形表现：

ST 表现：



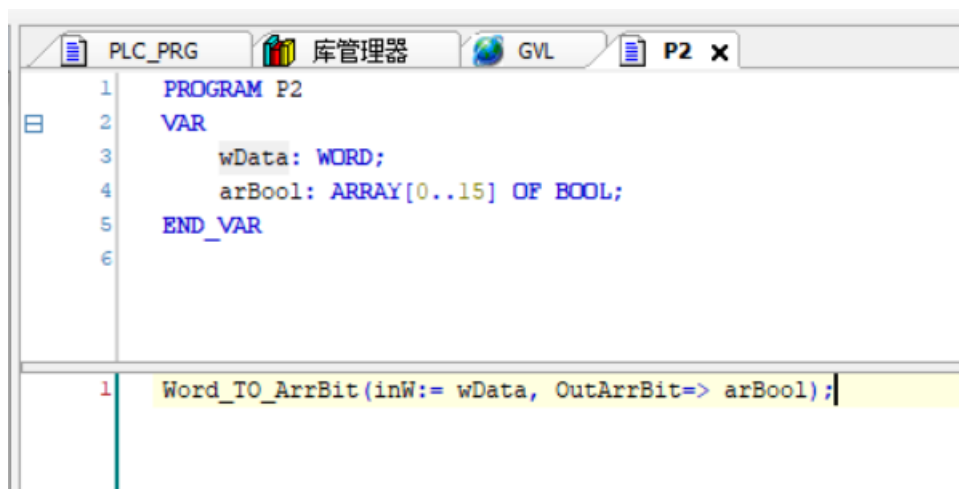
```
Word_To_ArrBit(inW:= , OutArrBit=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
inW	Word 数据	Word	输入输出变量	Word 数组数据
OutArrBit	转换后的 BOOL 数组	ARRAY[0..15] OF BOOL	输入输出变量	BOOL 数组数据

3、具体使用

(1) 声明和函数调用



(2) 写值与监控

表达式	类型	值	准备值	地址	注释
wData	WORD	15			
arBool	ARRAY [0..15] OF BOOL				
arBool[0]	BOOL	TRUE			
arBool[1]	BOOL	TRUE			
arBool[2]	BOOL	TRUE			
arBool[3]	BOOL	TRUE			
arBool[4]	BOOL	FALSE			
arBool[5]	BOOL	FALSE			
arBool[6]	BOOL	FALSE			
arBool[7]	BOOL	FALSE			
arBool[8]	BOOL	FALSE			
arBool[9]	BOOL	FALSE			
arBool[10]	BOOL	FALSE			
arBool[11]	BOOL	FALSE			
arBool[12]	BOOL	FALSE			
arBool[13]	BOOL	FALSE			
arBool[14]	BOOL	FALSE			
arBool[15]	BOOL	FALSE			

1 Word_TO_ArrBit(inW:= wData 15, OutArrBit=> arBool);RETURN

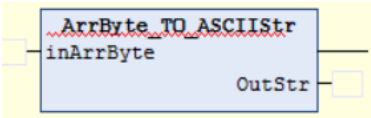
数据类型转换—Str,ASCII

ArrByte_To_ASCIIStr

函数功能：Byte 数组输入 28 个，转成 ASCII 输出字符串

1、指令格式

图形表现：



ST 表现：

```
ArrByte_To_ASCIIStr(inArrByte:= , OutStr=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
inArrByte	输入 Byte 数组	ARRAY [0..27] OF BYTE	输入变量	Byte 数组数据
OutStr	输出 String	STRING(28)	输出变量	String 数据

3、具体使用
(1) 声明和函数调用

PLC_PRG 库管理器 GVL P2 x Device Axis1 FV!

1
2
3
4
5
6
7
8
9

```
PROGRAM P2
VAR
    sData: STRING;
    arByte: ARRAY[0..27] OF BYTE;

END_VAR
```

1
2
3

```
ArrByte_TO_ASCIIStr(inArrByte:= arByte, OutStr=> sData);
```

(2) 写值与调用

sData	STRING	'\$01\$03\$04'		
arByte	ARRAY [0..27] OF BYTE			
arByte[0]	BYTE	1		
arByte[1]	BYTE	3		
arByte[2]	BYTE	4		
arByte[3]	BYTE	0		
arByte[4]	BYTE	0		
arByte[5]	BYTE	0		
arByte[6]	BYTE	0		
arByte[7]	BYTE	0		
arByte[8]	BYTE	0		
arByte[9]	BYTE	0		
arByte[10]	BYTE	0		
arByte[11]	BYTE	0		
arByte[12]	BYTE	0		

1
2
3
4

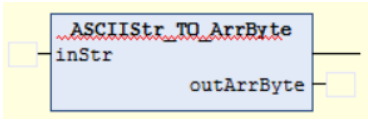
```
ArrByte_TO_ASCIIStr(inArrByte:= arByte, OutStr=> sData '$01$03$04' );
RETURN
```

ASCIIStr_To_ArrByte

函数功能：一个 ASCII 码字符串转换成多个 Byte 存入数组

1、指令格式

图形表现：



ST 表现：

```
ASCIIStr_To_ArrByte(inStr:= , outArrByte=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
inStr	输入 String	String(28)	输入变量	String 数据
OutArrByte	输出 Byte	ARRAY [0..27] OF BYTE	输出变量	Byte 数组数据

3、具体使用

(1) 声明和函数调用

```
1 PROGRAM P2
2 VAR
3     sData: STRING;
4     arByte: ARRAY[0..27] OF BYTE;
5
6
7
8 END_VAR
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

(2) 写值与监控

Device.Application.P2					
表达式	类型	值	准备值	地址	注释
sData	STRING	'ABCS'			
arByte	ARRAY [0..27] OF BYTE				
arByte[0]	BYTE	65			
arByte[1]	BYTE	66			
arByte[2]	BYTE	67			
arByte[3]	BYTE	83			
arByte[4]	BYTE	0			
arByte[5]	BYTE	0			
arByte[6]	BYTE	0			
arByte[7]	BYTE	0			
arByte[8]	BYTE	0			
arByte[9]	BYTE	0			

1

2

3

4

```
ASCIIStr_TO_ArrByte(inStr:= sData, outArrByte=>arByte );RETURN
```

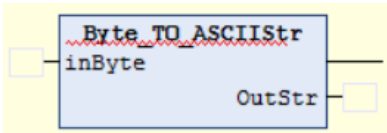
Byte_To_ASIIStr

函数功能：单个 byte 转换成 ASCII 码输出字符

1、指令格式

图形表现：

ST 表现：



```
Byte_TO_ASIIStr(inByte:= , OutStr=> );
```

2、相关变量

变量名称	名称	数据类型	变量属性	描述
inByte	输入 Byte 数据	Byte	输入变量	Byte 数据
OutStr	输出 String 数据	String(1)	输出变量	String 数据

3、具体使用

(1) 声明和函数调用

```

1  PROGRAM P2
2  VAR
3      sData: STRING;
4
5      bByte: BYTE;
6
7  END_VAR
8

```

```

1
2
3  Byte_TO_ASCIIStr(inByte:=bByte , OutStr=>sData );

```

(2) 写值与监控

Device.Application.P2

表达式	类型	值	准备值	地址	注释
sData	STRING	'#'			
bByte	BYTE	35			

```

1
2
3  ● Byte_TO_ASCIIStr(inByte:=bByte[35] , OutStr=>sData[ '#'] );RETURN

```

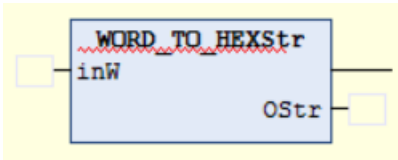
备注：ASC 码表：<https://www.hunanit.net/asciicode/>

Word_To_HEXStr

函数功能：十进制的数，以 16 进制用字符显示

1、指令格式

图形表现：



ST 表现：

```
WORD_TO_HEXStr(inW:= , OStr=> );
```

2、相关变量

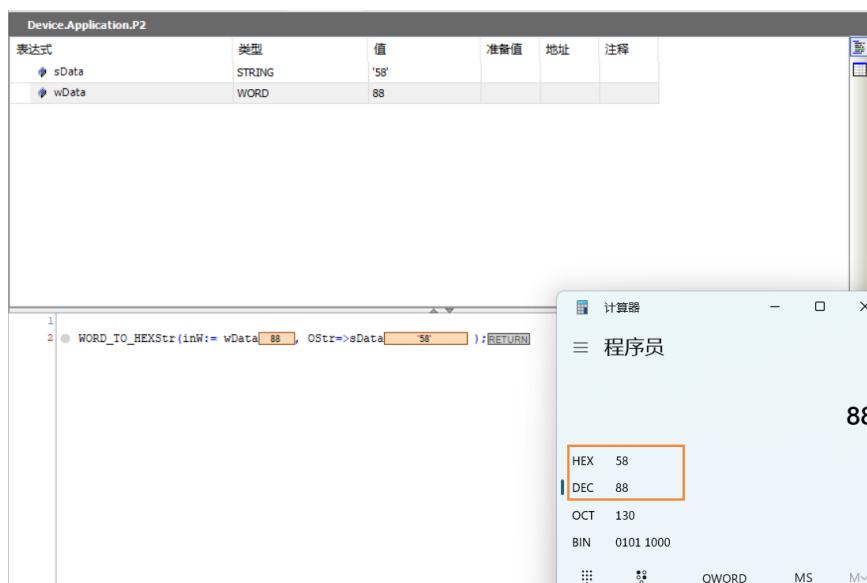
变量名称	名称	数据类型	变量属性	描述
inW	输入 Word 数据	WORD	输入变量	Byte 数据
OStr	输出 String 数据	String	输出变量	String 数据显示 16 进制值

3、具体使用

(1) 声明和函数调用

```
1 PROGRAM P2
2 VAR
3     sData: STRING;
4
5     wData: WORD;
6
7 END_VAR
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

(2) 写值与监控



FC_FlexemPLC

读取产品 ID， TRUE: 仿真 或者是 FlexemID； FALSE: 非 FlexemID

可加在程序的最上方，判断若不是繁易的总线 PLC，则直接 Return 不执行其他程序。

```

1 IF NOT FC_FlexmPLC() THEN
2     RETURN;
3 END_IF
4

```