Bronze100系列PLC指令手册

v1.0

天津滨海新区三格电子科技有限公司

版本信息

日期	版本号	修改内容	备注
2026/5/14	v1.0	建立	

第一章 概述

1.1 产品简介

Bronze100 是天津滨海新区三格电子科技有限公司研发的一款高性能、紧凑型可编程逻辑控制器。它基于工业级的STM32 32位ARM Cortex-M7系列微控制器设计，内置高性能实时运行内核。

该产品采用全球广泛使用的CODESYS V3软件平台作为编程环境，支持IEC 61131-3标准的六种编程语言（LD、FBD、ST、IL、SFC、CFC）。系统提供了强大的逻辑控制、数据处理及通信组网能力，旨在为各种工业自动化应用提供稳定、高效的解决方案。

CPU

Bronze100_R_AD_E
20路数字量输入（包含4路高速脉冲计数）
14路继电器输出
2路模拟量输入（0-10V，0-20mA），无隔离
1路模拟量输出（0-10V，0-20mA），无隔离
Bronze100_R_D_E
24路数字量输入（包含4路高速脉冲计数）
16路继电器输出
Bronze100_T_AD_E
20路数字量输入（包含4路高速脉冲计数）
14路晶体管输出（包含3路高速脉冲输出）
2路模拟量输入（0-10V，0-20mA），无隔离
1路模拟量输出（0-10V，0-20mA），无隔离
Bronze100_T_D_E
24路数字量输入（包含4路高速脉冲计数）
16路晶体管输出（包含3路高速脉冲输出）

数字量

DE08　8路数字量输入
DE16　16路数字量输入
DT08　8路晶体管输出
DR08　8路继电器输出
QT16　16路晶体管输出
QR16　16路继电器输出
DT16　8路数字量输入+8路晶体管输出
DR16　8路数字量输入+8路继电器输出
DT32　16路数字量输入+16路晶体管输出
DR32　16路数字量输入+16路继电器输出

模拟量

AE04　4路模拟量输入（±10V、0-20mA）
AE08　8路模拟量输入（±10V、0-20mA）
AQ02　2路模拟量输出（±10V、0-20mA）
AQ04　4路模拟量输出（±10V、0-20mA）
AM03　2路模拟量输入（±10V、0-20mA）+
1路模拟量输出（±10V、0-20mA）
AM06　4路模拟量输入（±10V、0-20mA）+
2路模拟量输出（±10V、0-20mA）

温度

AR02　2路PT100/PT1000
AR04　4路PT100/PT1000
AT04　4路热电偶输入，
支持K、J、E、T、S、R、B、N、C型热电偶

扩展电源

POWER　扩展电源

第二章 存储区与变量

了解 Bronze100 系列 PLC 的存储区划分和变量使用方法，是编写用户程序的基础。本章介绍以下内容：

五类存储区（I、Q、M、D、R）的容量与访问规则

直接地址的格式与寻址范围

常量与变量的声明方法

数组与结构体的使用

2.1 存储区总览

Bronze100 系列 PLC 内部共划分了 5 类存储区域，各自承担不同的数据角色：

区域代号	名称	容量	访问方式	掉电保持	主要用途
I 区	输入映象区	2KB	可寻址（位/字节/字/双字）	否	映射数字量/模拟量输入点
Q 区	输出映象区	2KB	可寻址（位/字节/字/双字）	否	映射数字量/模拟量输出点
M 区	中间寄存器区	16KB	可寻址（位/字节/字/双字）	否	存放中间变量、HMI交互数据
D 区	变量存储区	100KB	仅通过变量名访问（不可寻址）	否	存放数组、结构体、功能块实例
R 区	掉电保持区	4064 字节	仅通过变量名访问（不可寻址）	是	存放需断电保存的关键参数

各区域相互独立，容量互不挤占。

2.1.1 I 区（输入映象区）

I 区用于映射扩展模块的数字量输入和模拟量输入通道。

容量上限：2 KB

访问粒度：位（X）、字节（B）、字（W）、双字（D）

访问属性：只读

断电保持：不支持

仿真调试时，I 区地址可以被赋值或强制；在线运行时，其值来自外部信号（传感器、按钮、扩展模块等）。

2.1.2 Q 区（输出映象区）

Q 区用于控制扩展模块的数字量输出和模拟量输出通道。

容量上限：2 KB

访问粒度：位、字节、字、双字

访问属性：可读可写

断电保持：不支持

仿真和在线调试时，均可直接对 Q 区地址赋值或强制。

2.1.3 M 区（中间寄存器区）

M 区用于存放程序中间状态，以及与人机界面（HMI）、触摸屏、上位机交互的过程数据。

容量上限：16 KB

地址范围：MB0 ~ MB16383（按字节编址）

访问粒度：位、字节、字、双字

访问属性：可读可写

断电保持：不支持

M 区全部地址均可自由分配使用，无保留或自诊断占用。

2.1.4 D 区（变量存储区）

D 区用于存放数组、结构体、字符串、浮点数以及功能块实例。

容量上限：100 KB

访问方式：仅通过变量名（不可直接寻址）

访问属性：可读可写

断电保持：不支持

变量声明时不指定地址且不勾选保持属性，系统自动分配至 D 区。

2.1.5 R 区（掉电保持区）

R 区用于存放断电后需要保留的关键数据（如累计量、配方参数、设备 ID）。

容量上限：4064 字节

访问方式：仅通过变量名

访问属性：可读可写

断电保持：支持（永久保持）

变量声明时勾选保持功能，或定义在 VAR_RETAIN 区块中，系统自动分配至 R 区。

2.2 直接地址访问格式

遵循 IEC 61131-3 标准，直接地址以 % 开头：

% 区域前缀 数据格式 地址编号

区域	前缀	数据格式符号	说明
I 区	I	X(位) / B(字节) / W(字) / D(双字)	输入映象
Q 区	Q	X(位) / B(字节) / W(字) / D(双字)	输出映象
M 区	M	X(位) / B(字节) / W(字) / D(双字)	中间寄存器

地址示例（M 区）：

寻址方式	格式	示例	数据类型
位	%MXm.n	%MX100.0	BOOL
字节	%MBm	%MB100	BYTE
字	%MWm	%MW50	WORD
双字	%MDm	%MD25	DWORD

其中 m 为编号，n 为字节内的位编号（0~7）。I 区和 Q 区用法相同，替换前缀即可。

地址有效范围：

存储区	位范围	字节范围	字范围	双字范围
I 区	%IX0.0 ~ %IX2047.7	%IB0 ~ %IB2047	%IW0 ~ %IW1023	视模块而定
Q 区	%QX0.0 ~ %QX2047.7	%QB0 ~ %QB2047	%QW0 ~ %QW1023	视模块而定
M 区	%MX0.0 ~ %MX16383.7	%MB0 ~ %MB16383	%MW0 ~ %MW8191	%MD0 ~ %MD4095

超出上表范围的地址视为无效。

存储映射关系：

以 M 区为例，地址之间存在重叠关系：

双字	字	字节	位
%MD25	%MW51	%MB103	%MX103.7～%MX103.0
		%MB102	%MX102.7～%MX102.0
	%MW50	%MB101	%MX101.7～%MX101.0
		%MB100	%MX100.7～%MX100.0

映射规则：

%MX100.0 是 %MB100 的第 0 位

%MB100 由 %MX100.0～%MX100.7 组成

%MW50 由 %MB100 和 %MB101 组成

%MD25 由 %MW50 和 %MW51 组成

注意事项：

不同数据格式访问同一地址区域时会发生重叠。例如 %MW50 的值改变会同时影响 %MB100、%MB101 及对应的位地址。

直接地址访问仅支持 BOOL、BYTE、WORD、DWORD 类型。如需在特定地址上使用 INT、REAL 等类型，须采用"变量+地址"方式声明（见 2.8 节）。

2.3 变量

变量是程序运行时数值可以改变的数据载体。每个变量需声明名称、数据类型，并可选择指定存储位置或由系统自动分配。

2.3.1 命名规则

以字母或单下划线开头，后跟字母、数字或下划线

不允许连续两个下划线，不允许空格

不区分大小写（ABC 与 abc 视为同一变量）

不能使用关键词

支持中文变量名：在"工程设置"中勾选"允许标识符使用 unicode 字符"

2.3.2 数据类型

类型	名称	下限	上限	存储空间
BOOL	布尔	0	1	1 bit

BYTE	字节	0	255	8 bit
WORD	字	0	65535	16 bit
DWORD	双字	0	4294967295	32 bit
SINT	短整型	-128	127	8 bit
USINT	无符号短整型	0	255	8 bit
INT	整型	-32768	32767	16 bit
UINT	无符号整型	0	65535	16 bit
DINT	长整型	-2147483648	2147483647	32 bit
UDINT	无符号长整型	0	4294967295	32 bit
REAL	单精度浮点	-3.402823E+38	3.402823E+38	32 bit
TIME	时间	—	—	32 bit
STRING	字符串	—	—	可变

2.3.3 变量类型（作用域标识符）

标识符	说明
VAR	局部变量，仅在本 POU 内有效
VAR_INPUT	输入参数，用于程序调用时传入值
VAR_OUTPUT	输出参数，用于程序调用时返回值
VAR_IN_OUT	输入输出参数
VAR_TEMP	临时变量
VAR_GLOBAL	全局变量，工程内任意 POU 均可访问
VAR_EXTERNAL	外部变量，引用其他 POU 中的全局变量

VAR_CONSTANT	常量，只读
VAR_RETAIN	保持变量（掉电保存）

2.3.4 变量声明方式

自动声明：在编辑器中输入新变量名时，系统自动弹出对话框，填写范围、类型、初始值、地址（可选）即可。

手动声明：在变量声明区按以下格式书写：

<变量名> {AT <地址>} : <数据类型> {:= <初始值>};

示例：

```
VAR
    startBtn AT %IX0.0 : BOOL := FALSE;
    motorSpeed : INT := 0;
    pidParam AT %MD100 : REAL := 0.0;
END_VAR
```

2.3.5 变量存储位置的分配规则

声明方式	分配的存储区
使用 AT 指定 I 地址	I 区
使用 AT 指定 Q 地址	Q 区
使用 AT 指定 M 地址	M 区
不指定地址，且未声明 VAR_RETAIN	D 区（系统分配）
不指定地址，声明为 VAR_RETAIN	R 区（系统分配）
使用 AT 指定地址且声明 VAR_RETAIN	不支持（编译报错）

类型适配说明：直接地址访问仅支持 BOOL/BYTE/WORD/DWORD。要在某个地址上存放 REAL 或 INT 类型数据，必须使用 AT 方式声明变量。

2.4 常量

常量是程序运行期间数值固定不变的量。

类型	格式	示例

布尔	TRUE / FALSE	TRUE、FALSE
整数	十进制 / 二进制 / 八进制 / 十六进制	10、2#101010、8#52、16#AF
实数	十进制小数或科学计数法	3.14、1.23e-4
类型化常量	<类型>#<常量>	BOOL#1、DINT#123
时间	T# + 时间值 (d/h/m/s/ms 顺序)	T#5s、T#1h30m
时刻	TOD# + 时:分:秒	TOD#12:30:00
日期	D# + 年-月-日	D#2024-01-01
日期时刻	DT# + 年-月-日-时:分:秒	DT#2024-01-01-12:00:00
字符串	单引号括起	'Hello'、'$41'（字符 'A'）

2.5 数组

数组由相同数据类型的多个元素组成，支持一维、二维、三维。

语法：

<数组名> : ARRAY [<L1>..<U1> {,<L2>..<U2>, <L3>..<U3>}] OF <数据类型>;

示例：

```
// 一维数组，10个元素
counters : ARRAY [0..9] OF INT := [0,0,0,0,0,0,0,0,0,0];
// 二维数组，3行4列
matrix : ARRAY [1..3, 1..4] OF REAL;
// 使用缩写初始化（3(5) 表示三个5）
data : ARRAY [1..6] OF BYTE := [1,2,3,3(5)];  // 结果为 1,2,3,5,5,5
```

2.6 结构体（自定义数据类型）

将多种不同类型的数据组合成一个新类型。

定义：

```
TYPE <类型名> :
STRUCT
```

<成员1> : <数据类型1>;

<成员2> : <数据类型2>;

...

END_STRUCT

END_TYPE

示例：

```
TYPE MotorParams :
STRUCT
    Speed   : REAL;
    Current : INT;
    Temp    : INT;
    Alarm   : BOOL;
END_STRUCT
END_TYPE
```

变量声明与初始化：

```
m1 : MotorParams := (Speed := 1500.0, Current := 50, Temp := 25, Alarm := FALSE);
```

成员访问：

```
m1.Speed := 2000.0;
IF m1.Temp > 80 THEN
    m1.Alarm := TRUE;
END_IF
```

2.7 指针

指针用于存储数据的绝对地址。

定义：

<指针名> : POINTER TO <数据类型>;

常用操作：

ADR(变量)：取变量地址

指针名^：访问指针所指地址的内容

示例：批量复制数据

```
VAR
```

```
    i : INT;
    src : POINTER TO BYTE;
    dst : POINTER TO BYTE;
END_VAR
src := ADR(%MB100);     // 源地址：M区100号字节
dst := ADR(%MB300);     // 目标地址：M区300号字节
FOR i := 1 TO 100 DO
    dst^ := src^;
    src := src + 1;
    dst := dst + 1;
END_FOR
```

2.8 常见陷阱与工程注意事项

2.8.1 地址重叠问题

现象：不同数据格式访问同一地址区域时相互覆盖，且编译器不会报警。

示例：

```
// 假设声明了以下两个变量
temp1 AT %MB100 : BYTE;
temp2 AT %MW50 : WORD;     // %MW50 由 %MB100 和 %MB101 组成
```

执行 temp2 := 1000; 后，temp1 的值会被覆盖为 1000 的低 8 位（即 232）。

建议：

同一物理地址只使用一种粒度访问

使用 M 区时建立地址分配表，避免无意重叠

不同 POU 中的 AT 声明不会相互警告，需人工管理

2.8.2 D/R 区"不可寻址"的真正含义

含义：无法像 M 区那样直接使用 %DB100 或 %RB100 这样的地址。必须在变量声明时绑定变量名，由系统自动分配地址

正确做法：

```
VAR
    myArray : ARRAY[1..100] OF REAL;     // 系统自动分配至 D 区
END_VAR
```

如果需要绝对地址操作：

```
VAR
    pData : POINTER TO REAL;
    value : REAL;
END_VAR
pData := ADR(myArray[1]);    // 获取 D 区变量的地址
value := pData^;
```

2.8.3 R 区容量限制

容量：4064 字节

风险示例：

```
VAR_RETAIN
    param1 : STRING(255);    // 占用 256 字节
    param2 : STRING(255);    // 又占 256 字节
    // 16 个这样的字符串就会耗尽 R 区
END_VAR
```

建议：

编译时关注输出窗口的变量分配信息

大数组、长字符串尽量放在 D 区

R 区仅存放真正需要掉电保持的关键参数（累计量、设备 ID、配方索引等）

2.8.4 AT 与 VAR_RETAIN 冲突

规则：使用 AT 指定地址的变量不能声明为 VAR_RETAIN。

```
VAR_RETAIN
    // 错误：编译报错
    keepParam AT %MD100 : DINT := 0;
END_VAR
```

替代方案：将需要固定地址的保持变量放在 R 区，通过变量名访问（不指定地址）。

2.9 综合示例：电机控制参数管理

以下示例串联了 I 区、Q 区、M 区、D 区、R 区、数组、结构体、指针的典型用法。

```
// ========== 全局变量声明 ==========
VAR_GLOBAL
    // ---- I 区：物理输入（只读）----
    startBtn AT %IX0.0 : BOOL;
    stopBtn AT %IX0.1 : BOOL;
    speedPot AT %IW100 : UINT;          // 0-65535 对应 0-10V

    // ---- Q 区：物理输出（读写）----
    motorRun AT %QX0.0 : BOOL;
    motorFault AT %QX0.1 : BOOL;
    analogOut AT %QW200 : WORD;

    // ---- M 区：HMI 交互（读写）----
    hmiSetSpeed AT %MW100 : INT;        // HMI 设定转速
    hmiCurrentSpeed AT %MW102 : INT; // 上传当前转速
    hmiErrorCode AT %MB200 : BYTE;

    // ---- R 区：掉电保持参数 ----
    VAR_RETAIN
        totalRuntime : DINT := 0;           // 累计运行时间（扫描周期计数）
        maxCurrent : REAL := 15.5;          // 最大允许电流
        deviceID : DINT := 10001;           // 设备编号
        alarmHistory : ARRAY[1..10] OF DINT;   // 历史报警码
    END_VAR

    // ---- D 区：复杂数据结构 ----
    motorParams : MotorParams;              // 结构体实例
    speedCurve : ARRAY[1..100] OF INT;     // 速度曲线数组
END_VAR

// ========== 自定义结构体 ==========
```

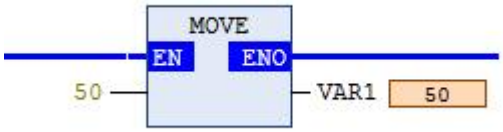
```
TYPE MotorParams :
STRUCT
    ratedSpeed     : INT;          // 额定转速
    ratedCurrent : REAL;         // 额定电流
    accelTime      : TIME;         // 加速时间
    decelTime      : TIME;         // 减速时间
    enable         : BOOL;         // 使能标志
END_STRUCT
END_TYPE

// ========== 主程序示例（POU） ==========
PROGRAM Main
VAR
    currentSpeed : INT := 0;
    pSpeed : POINTER TO INT;
    i : INT;
END_VAR

// 初始化结构体
motorParams := (ratedSpeed := 1500, ratedCurrent := 12.5,
                accelTime := T#2s, decelTime := T#3s, enable := TRUE);

// 启动控制
IF startBtn AND NOT motorRun AND motorParams.enable THEN
    motorRun := TRUE;
    motorFault := FALSE;
END_IF

// 停止控制
IF stopBtn OR (currentSpeed > motorParams.ratedSpeed + 100) THEN
    motorRun := FALSE;
END_IF
```

```
// 模拟量输入转换（0~65535 -> 0~1500）
IF motorRun THEN
    currentSpeed := speedPot * 1500 / 65535;
END_IF

// 同步到 HMI 显示
hmiCurrentSpeed := currentSpeed;

// 使用指针遍历速度曲线
pSpeed := ADR(speedCurve[1]);
FOR i := 1 TO 100 DO
    IF pSpeed^ > motorParams.ratedSpeed THEN
        hmiErrorCode := 1;  // 超速报警
    END_IF
    pSpeed := pSpeed + 1;
END_FOR

// 累计运行时间（配合定时器使用，简化示意）
IF motorRun THEN
    totalRuntime := totalRuntime + 1;
END_IF
```

第三章 指令系统

3.1 赋值指令

赋值指令 MOVE 用于将一个常量或者变量的值赋值给另外一个变量。

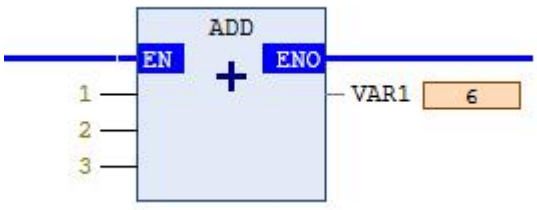
指令对应的输入和输出数据类型	
输入和输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING、ARRAY。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR **VAR1** INT	
编程语言	**程序**
梯形图（LD）	 结果 VAR1 为 50
结构化文本（ST）	VAR1:=50; 结果 VAR1 为 50

3.2 算术运算指令

3.2.1 ADD——加法指令

加法指令用于对两个及两个以上的常量或变量，进行相加运算。

指令对应的输入和输出数据类型
输入和输出的数据类型为 BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD。
变量定义
类别 名称 地址 数据类型 初值 注释 属性 VAR **VAR1** INT

编程语言	程序
梯形图（LD）	 结果 VAR1 为 6
结构化文本（ST）	VAR1:=1+2+3; 结果 VAR1 为 6

1、ADD 加法指令默认有两个输入端填写输入数据，可以在输入端添加输入。

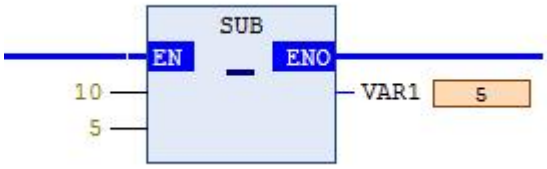
2、TIME 数据类型的输入数据与其它输入数据相加时，TIME+TIME = TIME、TOD+TIME = TOD、DT+TIME = DT。

3.2.2 SUB——减法指令

减法指令用于对两个常量或变量，进行相减运算。

指令对应的输入和输出数据类型
输入和输出的数据类型为 BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD。

变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		INT			

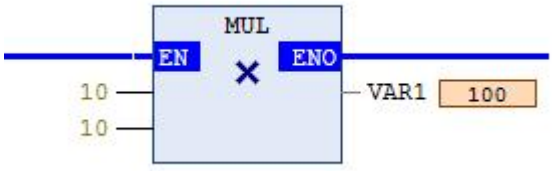
编程语言	程序
梯形图（LD）	 结果 VAR1 为5
结构化文本（ST）	VAR1:=10-5; 结果 VAR1 为5

TIME 数据类型的输入数据与其它输入数据相减时，TIME-TIME = TIME、DATE-DATE

= TIME、TOD-TIME = TOD、TOD -TOD = TIME、DT -TIME= DT、DT -DT = TIME。

3.2.3 MUL——乘法指令

乘法指令用于对两个及两个以上的常量或变量，进行相乘运算。

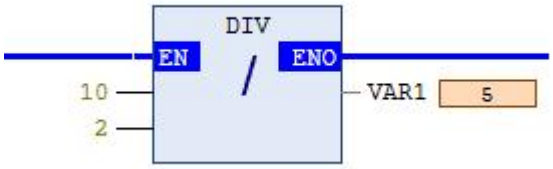
指令对应的输入和输出数据类型	
输入和输出的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 INT	
编程语言	程序
梯形图（LD）	 结果 VAR1 为 100
结构化文本（ST）	VAR1:=10*10; 结果 VAR1 为 100

MUL 乘法指令默认有两个输入端填写输入数据，可以在输入端处添加输入。

3.2.4 DIV——除法指令

除法指令用于对两个常量或变量，进行相除运算。

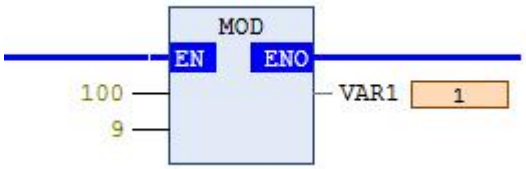
指令对应的输入和输出数据类型	
输入和输出的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 INT	
编程语言	程序

梯形图 （LD）	 结果 VAR1 为 5
结构化文本 （ST）	Var1:=10/2; 结果 VAR1 为 5

使用除法指令时需要注意不要出现除数为 0 的情况，除数为 0 会根据目标系统产生意料之外的结果。

3.2.5 MOD——取余指令

取余指令用于对一个变量或常量进行相除取余，其结果是一个整数。

指令对应的输入和输出数据类型						
输入和输出的数据类型为 BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。						
变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		INT			
编程语言	**程序**					
梯形图 （LD）	 结果 VAR1 为 1					
结构化文本 （ST）	VAR1:=100 MOD 9; 结果 VAR1 为1					

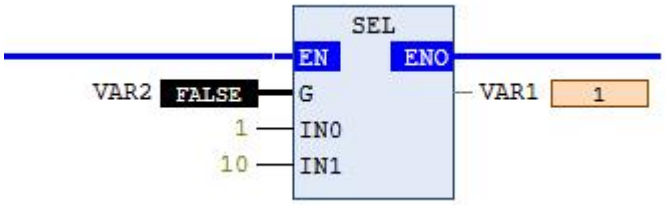
对零取余后，结果为 0。

3.3 选择指令

所有选择指令均可接受变量或常量输入。需要注意，输出数据的数据类型存储长度必须大于等于输入数据的数据类型类型存储长度。

3.3.1 SEL——二选一指令

二选一指令的格式为：OUT:=SEL(G, IN0, IN1)，其中 G 为控制位，IN0 和 IN1 分别为两个输入数据，OUT 为输出数据。通过控制位的值，选择两个输入数据中的其中一个作为输出数据，控制位的值为 0 时选择第一个输入数据，控制位的值为 1 时选择第二个输入数据。

<table>
<tr><th colspan="2">指令对应的输入和输出数据类型</th></tr>
<tr><td colspan="2">该指令中控制位为 BOOL。
IN0 、 IN1 和 OUT 的数据类型为 BOOL 、 BYTE 、 DATE 、 DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING、ARRAY。</td></tr>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2">
<table>
<tr><th></th><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>1</td><td>VAR</td><td>VAR1</td><td></td><td>INT</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>VAR</td><td>VAR2</td><td></td><td>BOOL</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td>梯形图
（LD）</td><td> 

VAR2 为 FALSE，选择对应 IN0，VAR1 结果为 1</td></tr>
<tr><td>结构化文本
（ST）</td><td>VAR1:=SEL(VAR2,1,10);
Var1 结果为 1</td></tr>
</table>

1.输入参数引脚 IN0, IN1 和输出参数引脚 OUT 需要使用相同的数据类型，特别是使用用户定义的数据类型时。

2.当控制位的值 G 为 1 时，软件不计算 IN0 的表达式。当控制位的值 G 为 0 时，软件不会计算 IN1 的表达式。

3.3.2 MUX——多选一指令

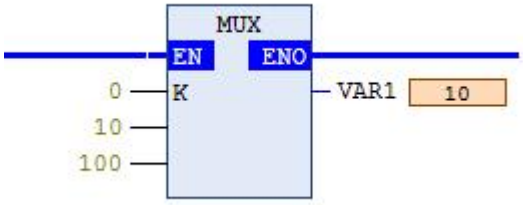
多选一指令 MUX 的功能是根据控制数的不同，选择两个及两个以上输入数据中的一个值作为输出。该指令格式为：OUT:=MUX(K,IN0,…,INn)，其中 K 为

控制数，输入数据为 IN0~INn，OUT 为输出数据，输出的值等于输入数据 INk。

指令对应的输入和输出数据类型
输入和输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING、ARRAY。 控制数 K 必须是下面类型中的一种：BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		INT			

编程语言	程序
梯形图（LD）	 VAR1 结果为 10
结构化文本（ST）	VAR1:=MUX(0,10,100,); Var1 结果为 10

1、MUX 多选一指令默认有两个输入端填写输入数据，可以在输入端处添加输入。

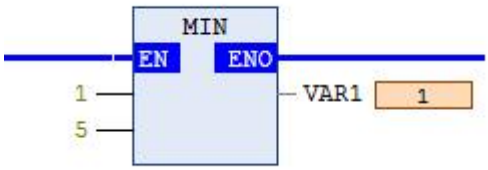
2、输入参数引脚 IN0, ..., INn 和输出参数引脚 OUT 需要使用相同的数据类型，特别是使用用户定义的数据类型时。

3、MUX 从一组输入参数引脚 IN0, ..., INn 中选择一个作为输出数据，控制数为 K，若选择第一个输入数据作为输出时，K=0。若 K 比 n 值大，那么软件传递输入数据的最后一个值(INn)。

4、出于对运行优化的考虑，软件只计算输入中的第 K 个表达式。

3.3.3 MIN——取最小值指令

MIN 指令的功能为取出两个及两个以上输入数据中的最小值作为输出结果。其格式为： OUT:=MIN(IN0,…,INn)，IN0,…,INn 为输入数据，OUT 为输出数据。

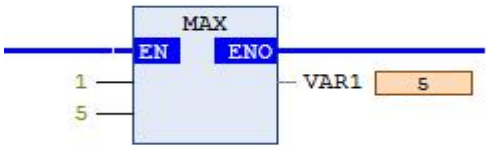
指令对应的输入和输出数据类型	
输入和输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 INT	
编程语言	程序
梯形图（LD）	 VAR1 结果为 1
结构化文本（ST）	VAR1:=MIN(1,5); Var1 结果为 1

MIN 取最小值指令默认有两个输入端填写输入数据，可以在输入端处添加输入。

3.3.4 MAX——取最大值指令

MAX 指令的功能为取出两个及两个以上输入数据中的最大值作为输出结果。其格式为：OUT:=MAX (IN0,…,INn)，IN0,…,INn 为输入数据，OUT 为输出数据。

指令对应的输入和输出数据类型	
输入和输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 INT	
编程语言	程序

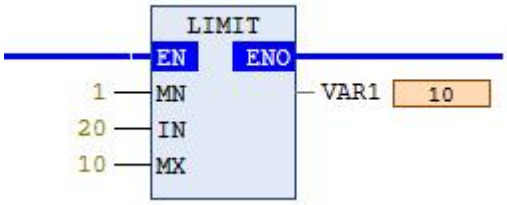
梯形图 （LD）	 Var1 结果为 6
结构化文本 （ST）	Var1:=MAX(1,5); Var1 结果为 5

MAX 取最大值指令默认有两个输入端填写输入数据，可以在输入端处添加输入。

3.3.5 LIMIT——极限值指令

极限值指令 LIMIT 用于控制输入数据在一定范围内，其格式为 OUT:=LIMIT(Min,IN,Max)。若输入数据介于最小值和最大值之间，输出数据等于输入数据；若输入数据小于最小值，输出数据等于最小值；若输入数据大于最大值，输出数据等于最大值。

指令对应的输入和输出数据类型						
输入和输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。						
变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		INT			

编程语言	程序
梯形图 （LD）	 VAR1 结果为 10
结构化文本 （ST）	VAR1:=LIMIT(1,20,10)； VAR1 结果为 10

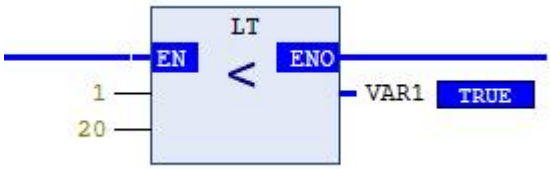
若最小值大于最大值，输出数据等于最大值。

3.4 比较指令

所有选择指令均可接受变量或常量输入。

3.4.1 LT——小于指令

小于指令 LT 用于比较两个输入数据的大小。当第一个输入数据小于第二个输入数据时，指令输出数据为TRUE，否则输出数据为 FALSE。

<table>
<tr><th colspan="2">指令对应的输入和输出数据类型</th></tr>
<tr><td colspan="2">输入的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。
输出的数据类型为 BOOL。</td></tr>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2"><table>
<tr><th></th><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>1</td><td>VAR</td><td>VAR1</td><td></td><td>BOOL</td><td></td><td></td><td></td></tr>
</table></td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td>梯形图
（LD）</td><td> 

VAR1 结果为 TRUE</td></tr>
<tr><td>结构化文本
（ST）</td><td>VAR1:=1<20；
VAR1 结果为 TRUE</td></tr>
</table>

3.4.2 LE——小于等于指令

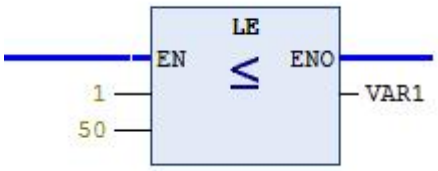
小于等于指令 LE 用于比较两个输入数据的大小。当第一个输入数据小于等于第二个输入数据时，指令输出数据为 TRUE，否则输出数据为 FALSE。

<table>
<tr><th>指令对应的输入和输出数据类型</th></tr>
<tr><td>输入的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。</td></tr>
</table>

输出的数据类型为 BOOL。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
L	VAR	VAR1		BOOL			

编程语言	程序
梯形图（LD）	 VAR1 结果为 TRUE
结构化文本（ST）	VAR1:=1<=50； VAR1 结果为 TRUE

3.4.3 GT——大于指令

大于指令 GT 用于比较两个输入数据的大小。当第一个输入数据大于第二个输入数据时，指令输出数据为TRUE，否则输出数据为 FALSE。

指令对应的输入和输出数据类型

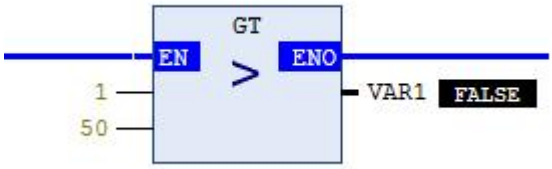
输入的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。

输出的数据类型为 BOOL。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
L	VAR	VAR1		BOOL			

编程语言	程序

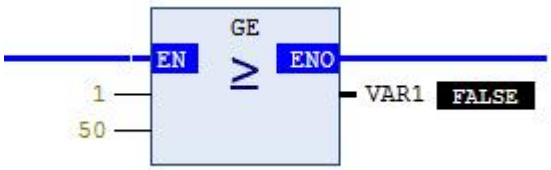
梯形图 （LD）	 VAR1 结果为 FALSE
结构化文本 （ST）	VAR1:=1>50； VAR1结果为 FALSE

3.4.4 GE——大于等于指令

大于等于指令 GE 用于比较两个输入数据的大小。当第一个输入数据大于等于第二个输入数据时，指令输出数据为 TRUE，否则输出数据为 FALSE。

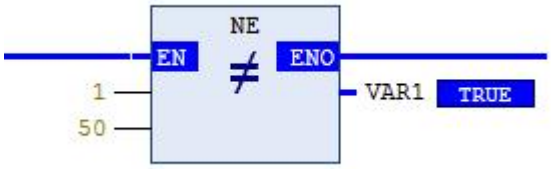
指令对应的输入和输出数据类型
输入的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。 输出的数据类型为 BOOL。
变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		BOOL			

编程语言	程序
梯形图 （LD）	 VAR1 结果为 FALSE
结构化文本 （ST）	VAR1:=1>=50； VAR1 结果为 FALSE

3.4.5 NE——不等于指令

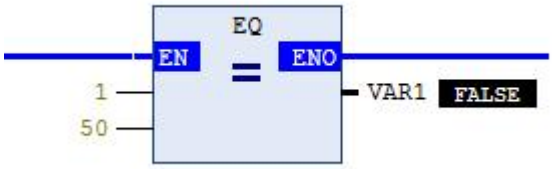
不等于指令 NE 用于比较两个输入数据的大小。当第一个输入数据不等于第二个输入数据时，指令输出数据为TRUE，否则输出数据为 FALSE。

指令对应的输入和输出数据类型	
输入的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。 输出的数据类型为 BOOL。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 BOOL	
编程语言	程序
梯形图（LD）	 VAR1 结果为 TRUE
结构化文本（ST）	VAR1:=1<>50; VAR1 结果为 TRUE

3.4.6 EQ——等于指令

等于指令 EQ 用于比较两个输入数据的大小。当第一个输入数据等于第二个输入数据时，指令输出数据为TRUE，否则输出数据为 FALSE。

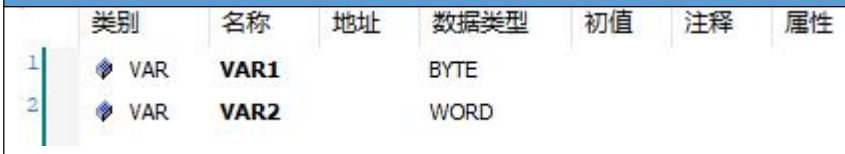
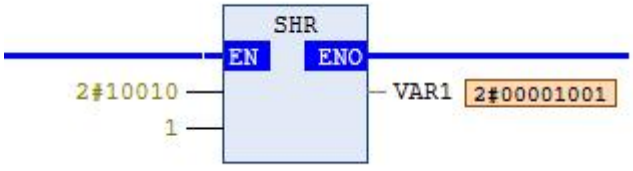
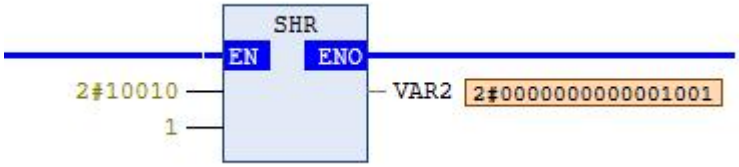
指令对应的输入和输出数据类型	
输入的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。 输出的数据类型为 BOOL。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 BOOL	
编程语言	程序

梯形图 （LD）	 VAR1 结果为 FALSE
结构化文本 （ST）	VAR1:=1=50; VAR1 结果为 FALSE

3.5 移位指令

3.5.1 SHR——右移指令

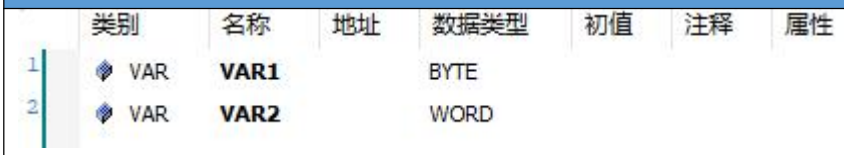
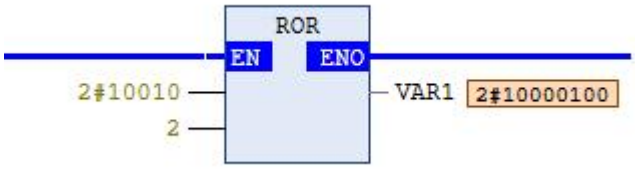
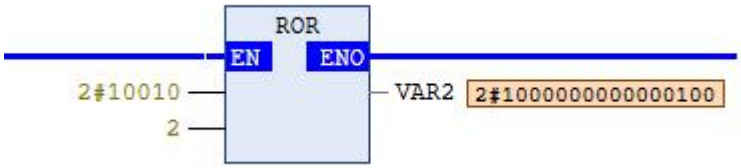
右移指令是对输入数据进行按位右移，左边空缺位自动补 0，不需要处理右边移出的位。

指令对应的输入和输出数据类型	
输入和输出的数据类型为 BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。	
变量定义	
	
编程语言	程序
梯形图 （LD）	 VAR1 结果为 2#00001001  VAR2 结果为 2#0000000000001001

结构化文本（ST）	VAR1:=SHR(2#10010,1); VAR1 结果为 2#00001001 VAR2:=SHR(2#10010,1); VAR2 结果为 2#0000000000001001

3.5.2 ROR——循环右移指令

循环右移指令是对输入数据进行按位循环右移，右边移出的最低位循环送到左边的最高位处。

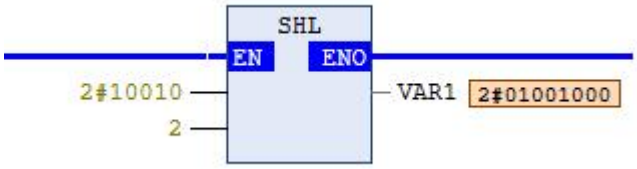
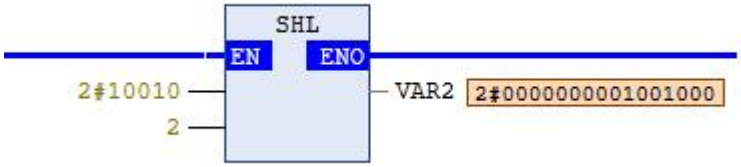
指令对应的输入和输出数据类型	
输入和输出的数据类型为 BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。	
变量定义	
	
编程语言	**程序**
梯形图（LD）	 VAR1 结果为 2#10000100  VAR2 结果为 2#1000000000000100
结构化文本（ST）	VAR1:=ROR(2#10010,2); VAR1 结果为2#10000100 VAR2:=ROR(2#10010,2); VAR2 结果为 2#1000000000000100

在循环右移过程中，若输出数据的数据类型的长度不一样，则输入数据相同时输出的数

据结果不一样。

3.5.3 SHL——左移指令

左移指令是对输入数据进行按位左移，右边空缺位自动补 0，不需要处理左边移出的位。

指令对应的输入和输出数据类型	
输入和输出的数据类型为 BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 1 VAR VAR1 BYTE 2 VAR VAR2 WORD	
编程语言	程序
梯形图（LD）	 VAR1 结果为 2#01001000  VAR2 结果为 2#0000000001001000
结构化文本（ST）	VAR1:=SHL(2#10010,2); VAR1 结果为 2#01001000 VAR2:=SHL(2#10010,2); VAR2 结果为 2#0000000001001000

3.5.4 ROL——循环左移指令

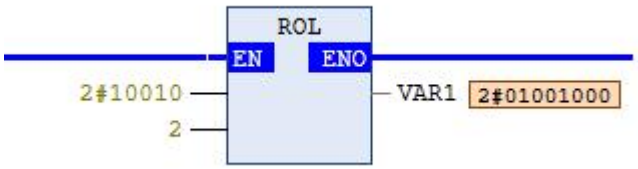
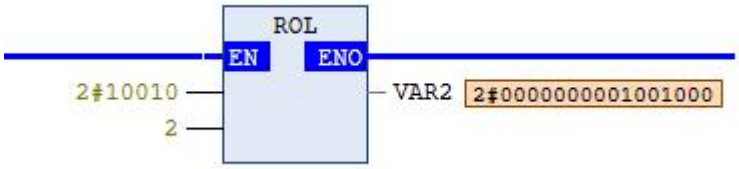
循环左移指令是对输入数据进行按位循环左移，左边移出的最高位循环送到右边的最低位处。

指令对应的输入和输出数据类型

输入和输出的数据类型为 BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		BYTE			
2	VAR	VAR2		WORD			

编程语言	程序
梯形图（LD）	 VAR1 结果为 2#01001000  VAR2 结果为 2#0000000001001000
结构化文本（ST）	VAR1:=ROL(2#10010,2); Var1 结果为 2#01001000 VAR2:=ROL(2#10010,2); Var2 结果为 2#0000000001001000

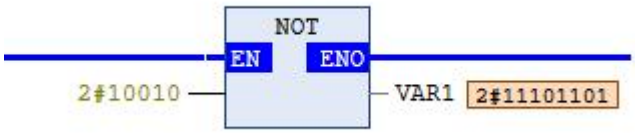
3.6 逻辑运算指令

3.6.1 NOT——取非指令

取非指令用于对变量或常量的所有位进行取非运算。

指令对应的输入和输出数据类型
输入和输出的数据类型为 BOOL、BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		BYTE			

编程语言	程序
梯形图（LD）	 VAR1 结果为 2#11101101
结构化文本（ST）	VAR1:=NOT 2#10010; VAR1 结果为 2#011101101

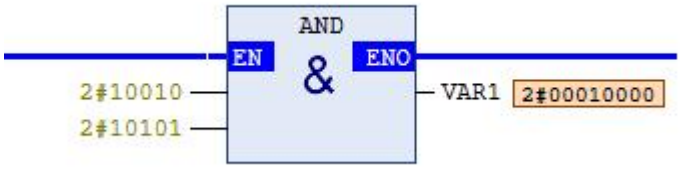
3.6.2 AND——与指令

与指令用于对两个及两个以上的变量或常量的所有位，按照由低位到高位的顺序进行与运算。

指令对应的输入和输出数据类型
输入和输出的数据类型为 BOOL、BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		BYTE			

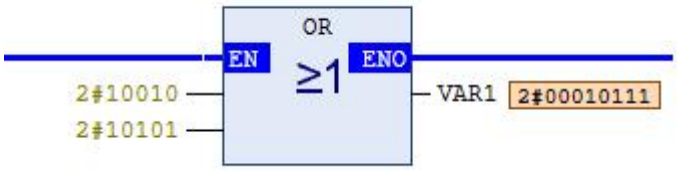
编程语言	程序
梯形图（LD）	 VAR1 结果为 2#00010000
结构化文本（ST）	VAR1:=2#10010 AND 2#10101; VAR1 结果为 2#00010000

1.AND 与指令默认有两个输入端填写输入数据，可以在输入端处添加输入。

2.输出数据的数据类型长度要大于或等于所有输入数据的数据类型长度。

3.6.3 OR——或指令

或指令用于对两个及两个以上的变量或常量的所有位，按照由低位到高位的顺序进行或运算。

指令对应的输入和输出数据类型	
输入和输出的数据类型为 BOOL、BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 1 VAR **VAR1** BYTE	
编程语言	**程序**
梯形图（LD）	 VAR1 结果为 2#00010111
结构化文本（ST）	VAR1:=2#10010 OR 2#00010111; Var1 结果为 2#11101111

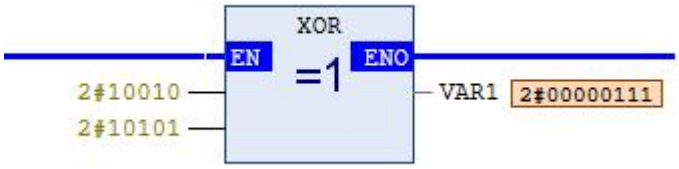
1.OR 或指令默认有两个输入端填写输入数据，可以在输入端处添加输入。

2.输出数据的数据类型长度要大于或等于所有输入数据的数据类型长度。

3.6.4 XOR——异或指令

异或指令用于对两个及两个以上的变量或常量的所有位，按照由低位到高位的顺序进行异或运算。

指令对应的输入和输出数据类型	
输入和输出的数据类型为 BOOL、BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 1 VAR **VAR1** BYTE	
编程语言	**程序**

梯形图 （LD）	 VAR1 结果为 2#00000111
结构化文本 （ST）	VAR1:=2#10010 XOR 2#1010101; VAR1 结果为 2#00000111

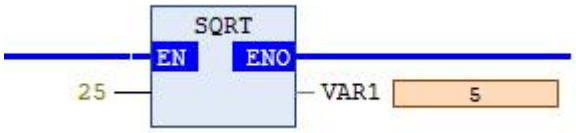
1.XOR 异或指令默认有两个输入端填写输入数据，可以在输入端处添加输入。

2.输出数据的数据类型长度要大于或等于所有输入数据的数据类型长度。

3.7 初等数学运算指令

3.7.1 SQRT——平方根指令

平方根指令用于求解输入数据的平方根，输出数据为运算结果。

指令对应的输入和输出数据类型	
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型是 REAL。	
变量定义	
	
编程语言	**程序**
梯形图 （LD）	 VAR1 结果为 5
结构化文本 （ST）	VAR1:=SQRT(25); VAR1 结果为 5

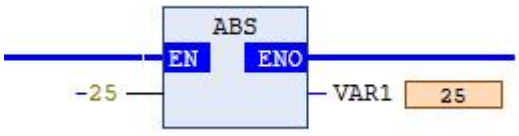
当输入数据为负数时，输出数据的值为 0。

3.7.2 ABS——绝对值指令

绝对值指令用于求解输入数据的绝对值，输出数据为运算结果。

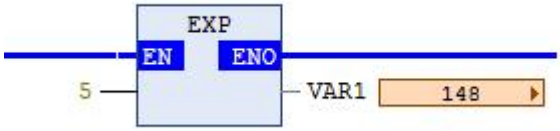
输入数据类型	输出数据类型
BYTE	BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD
DINT	DINT、DWORD、REAL、UDINT
DWORD	DINT、DWORD、REAL、UDINT
INT	DINT、DWORD、INT、REAL、UDINT、UINT、WORD
REAL	REAL
SINT	BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD
UDINT	DINT、DWORD、REAL、UDINT
UINT	DINT、DWORD、INT、REAL、UDINT、UINT、WORD
USINT	BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD
WORD	DINT、DWORD、INT、REAL、UINT、WORD

变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		INT			

编程语言	程序
梯形图（LD）	 VAR1 结果为 25
结构化文本（ST）	VAR1:=ABS(-25); VAR1 结果为 25

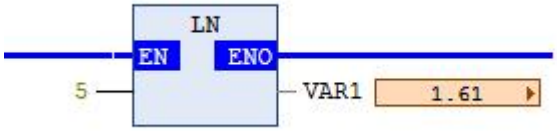
3.7.3 EXP——指数指令

指数指令用于求解底数为 e、指数为输入数据的幂运算，即 ex，输出数据为运算结果。

指令对应的输入和输出数据类型	
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 REAL	
编程语言	程序
梯形图（LD）	 VAR1 结果为 148
结构化文本（ST）	VAR1:=EXP(5); VAR1 结果为 148

3.7.4 LN——自然对数指令

自然对数指令用于求解输入数据的自然对数，输出数据为运算结果。

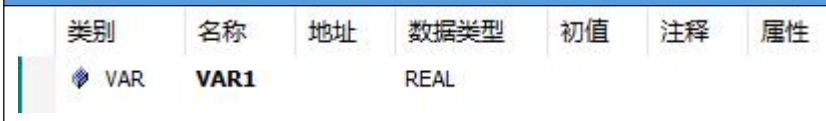
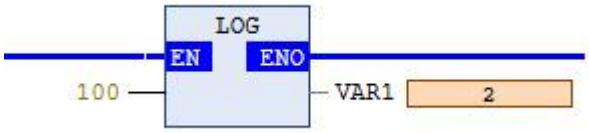
指令对应的输入和输出数据类型	
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 REAL	
编程语言	程序
梯形图（LD）	

	VAR1 结果为 1.61
结构化文本（ST）	VAR1:=LN(5); VAR1 结果为 1.61

当输入数据为负数时，输出数据的值为 0。

3.7.5 LOG——常用对数指令

常用对数指令用于求解输入数据以 10 为底的对数，输出数据为运算结果。

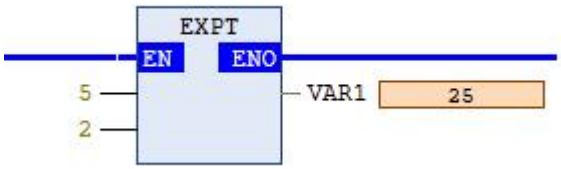
指令对应的输入和输出数据类型	
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。	
变量定义	
	
编程语言	程序
梯形图（LD）	 VAR1 结果为 2
结构化文本（ST）	VAR1:=LOG(100); VAR1 结果为 2

当输入数据为负数时，输出数据的值为 0。

3.7.6 EXPT——幂指令

幂指令用于对两个输入数据进行幂运算，第一个数据为幂运算的底数，第二个数据为幂运算的指数，输出数　据为运算结果。

指令对应的输入和输出数据类型

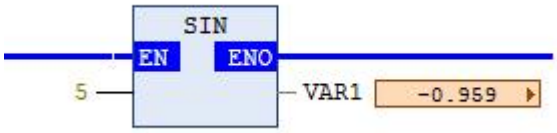
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 REAL	
编程语言	**程序**
梯形图 （LD）	 VAR1 结果为 25
结构化文本 （ST）	VAR1:=EXPT(5,2); VAR1 结果为 25

3.7.8 SIN——正弦指令

在软件中，所有的三角函数均采用弧度进行运算，包括正弦、余弦、正切、余切。若已知角度值，需换算成　　弧度值后再参与计算。应用上述三角函数的反函数进行运算时，其结果为弧度值。若要得到角度值，可对弧度值进行换算，弧度=角度/180*3.14。

正弦指令用于求解输入数据的正弦值，输出数据为运算结果。

输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 REAL	
编程语言	**程序**

梯形图 （LD）	 VAR1 结果为-0.959
结构化文本 （ST）	VAR1:=SIN(5); VAR1 结果为-0.959

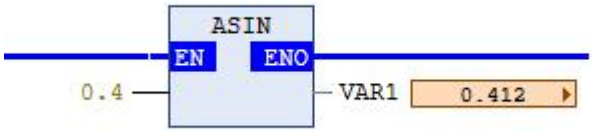
3.7.9 ASIN——反正弦指令

反正弦指令用于求解输入数据的反正弦值，输出数据为运算结果。

指令对应的输入和输出数据类型
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		REAL			

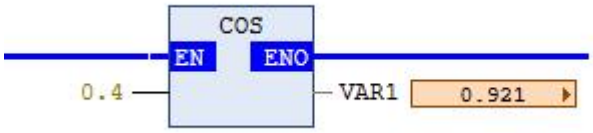
编程语言	程序
梯形图 （LD）	 VAR1 结果为 0.412
结构化文本 （ST）	VAR1:=ASIN(0.4); VAR1 结果为 0.412

3.7.10 COS——余弦指令

余弦指令用于求解输入数据的与弦值，输出数据为运算结果。

指令对应的输入和输出数据类型
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。

变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		REAL			

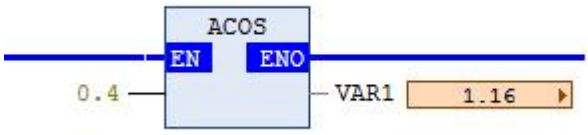
编程语言	程序
梯形图（LD）	 VAR1 结果为 0.921
结构化文本（ST）	VAR1:=COS(0.4); VAR1 结果为 0.921

3.7.11 ACOS——反余弦指令

反余弦指令用于求解输入数据的反余弦值，输出数据为运算结果。

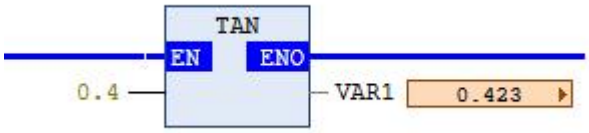
指令对应的输入和输出数据类型
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。

变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		REAL			

编程语言	程序
梯形图（LD）	 VAR1 结果为 1.16
结构化文本（ST）	VAR1:=ACOS(0.4); VAR1 结果为 1.16

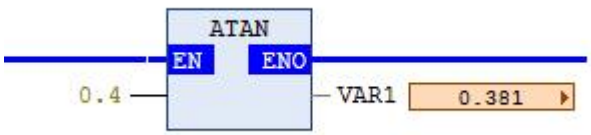
3.7.12 TAN——正切指令

正切指令用于求解输入数据的正切值，输出数据为运算结果。

指令对应的输入和输出数据类型	
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 REAL	
编程语言	程序
梯形图（LD）	 VAR1 结果为 0.423
结构化文本（ST）	VAR1:=TAN(0.4); VAR1 结果为 0.423

3.7.13 ATAN——反正切指令

反正切指令用于求解输入数据的反正切值，输出数据为运算结果。

指令对应的输入和输出数据类型	
输入的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、USINT、WORD。 输出的数据类型为 REAL。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 VAR VAR1 REAL	
编程语言	程序
梯形图（LD）	 VAR1 结果为 0.381

结构化文本（ST）	VAR1:=ATAN(0.4); VAR1 结果为 0.381

3.8 地址运算指令

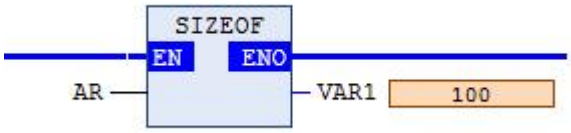
3.8.1 SIZEOF——数据类型大小指令

通过调用数据类型大小指令可获取各种数据类型的变量所占用的存储空间字节数。SIZEOF 数据类型大小指令的输入数据为需要计算所占用存储空间的变量，输出数据为计算得出的存储空间字节数，是一个无符号的值。

指令对应的输入和输出数据类型
输入的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING、ARRAY。 输出的数据类型为 BYTE、DINT、DWORD、INT、REAL、SINT、UDINT、UINT、WORD。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		REAL			
2	VAR	AR		ARRAY[1..50]OF INT			

编程语言	程序
梯形图（LD）	 VAR1 结果为 100
结构化文本（ST）	VAR1:=SIZEOF(AR); VAR1 结果为 100

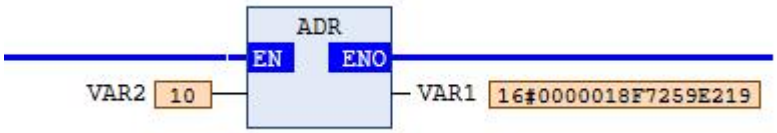
1、DATE、DATE_AND_TIME(DT)、TIME、TIME_OF_DAY(TOD)等数据类型的输入数据所占用的地址空间均为 4 个字节，即双字。

2、STRING 数据类型的输入数据所占用的地址空间为 81 个字节。

3、WSTRING 数据类型的输入数据所占用的地址空间为 162 个字节。

3.8.2 ADR——取地址指令

在 PLC 应用中，经常会遇到需要获取变量绝对地址的情况。取地址指令可获取变量的绝对地址，获取的地址可当作指针使用，既可参加指针运算，也可作为输入参数传输给函数。ADR 取地址指令的输入数据为需要获取绝对地址的变量，输出数据为获取到的变量绝对地址。

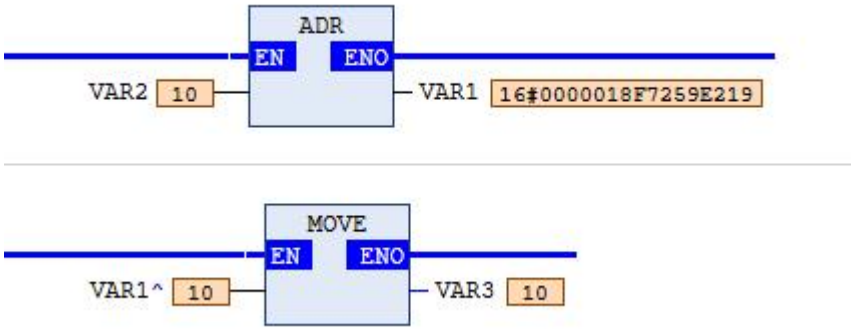
<table>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2">
<table>
<tr><th></th><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>1</td><td>VAR</td><td>VAR1</td><td></td><td>POINTER TO BYTE</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>VAR</td><td>VAR2</td><td></td><td>BYTE</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td>梯形图
（LD）</td><td> 

VAR1 结果为 16#0000018F7259E219</td></tr>
<tr><td>结构化文本
（ST）</td><td>VAR1:=ADR(VAR2);
VAR1结果为 16#0000018F7259E219</td></tr>
</table>

3.8.3 ^——取地址内容指令

取地址指令对应的指令为取地址内容指令。通过该指令取出绝对地址中的内容，其语法格式为：在指针变量后增加“^”符号。

<table>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2">
<table>
<tr><th></th><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>1</td><td>VAR</td><td>VAR1</td><td></td><td>POINTER TO BYTE</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>VAR</td><td>VAR2</td><td></td><td>BYTE</td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>VAR</td><td>VAR3</td><td></td><td>BYTE</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
</table>

梯形图 （LD）	 VAR3 结果为 10
结构化文本 （ST）	VAR1:= ADR(VAR2); VAR3:=Var1^; VAR3 结果为10

3.9 转换指令

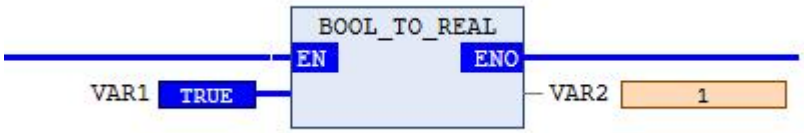
3.9.1 BOOL_TO_<TYPE>——布尔类型转换指令

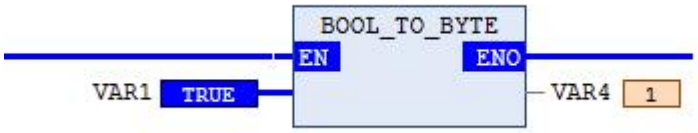
布尔类型转换指令用于把布尔数据类型的输入数据转换为其它数据类型输出。

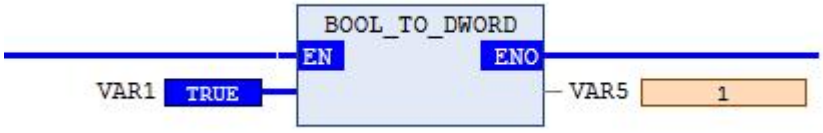
指令对应的输入和输出数据类型
输入的数据类型为 BOOL。 输出的数据类型为 BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。
变量定义

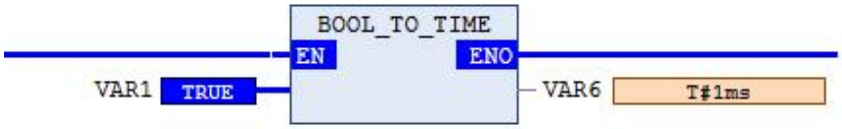
	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		BOOL			
2	VAR	VAR2		REAL			
3	VAR	VAR3		DATE			
4	VAR	VAR4		BYTE			
5	VAR	VAR5		DWORD			
6	VAR	VAR6		TIME			

编程语言	程序

<table>
<tr><td>梯形图
（LD）</td><td> 

VAR2 结果为 1
 

VAR4 结果为 1
 

VAR5 结果为 1
 

VAR6 结果为 T#1ms</td></tr>
<tr><td>结构化文本
（ST）</td><td>VAR2:=BOOL_TO_REAL(VAR);
VAR2 结果为 1
VAR4:=BOOL_TO_BYTE(VAR1);
VAR4 结果为 1
VAR5:=BOOL_TO_DWORD(VAR1);
VAR5 结果为 1
VAR6:=BOOL_TO_TIME(VAR1);
VAR6 结果为 T#1ms</td></tr>
</table>

3.9.2 BYTE_TO_<TYPE>——字节型转换指令

字节型转换指令用于把字节型的输入数据转换为其它数据类型输出。

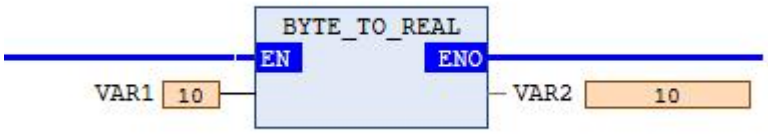
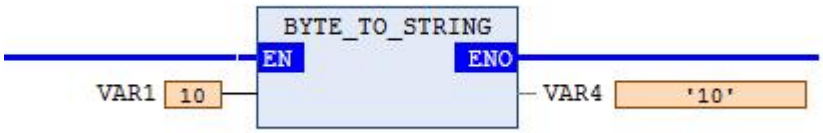
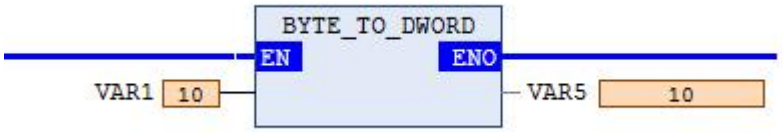
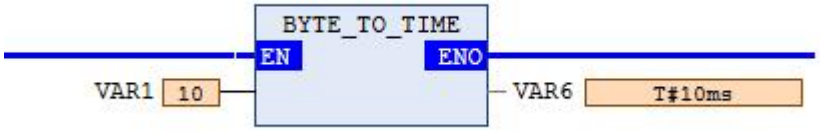
指令对应的输入和输出数据类型

输入的数据类型为 BYTE。

输出的数据类型为BOOL、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**VAR1**		BYTE			
2	VAR	**VAR2**		REAL			
3	VAR	**VAR3**		DATE			
4	VAR	**VAR4**		STRING			
5	VAR	**VAR5**		DWORD			
6	VAR	**VAR6**		TIME			

编程语言	程序
梯形图（LD）	 VAR2 结果为 10  VAR4 结果为'10'  VAR5 结果为 10  VAR6 结果为 T#10ms

结构化文本（ST）	VAR2:=BYTE_TO_REAL(VAR1); VAR2 结果为 10 VAR4:=BYTE _TO_STRING(VAR1); VAR4 结果为'10' VAR5:=BYTE _TO_DWORD(VAR1); VAR5 结果为 10 VAR6:=BYTE _TO_TIME(VAR1); VAR6 结果为 T#10ms

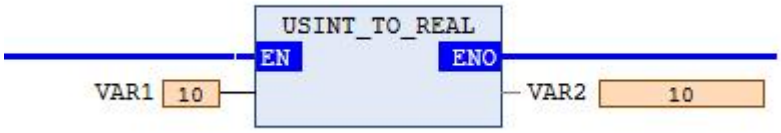
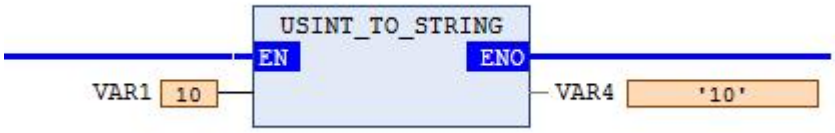
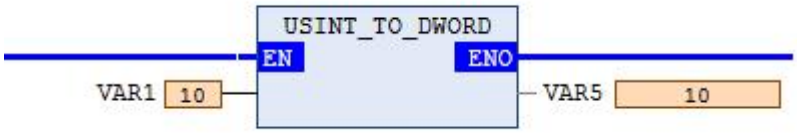
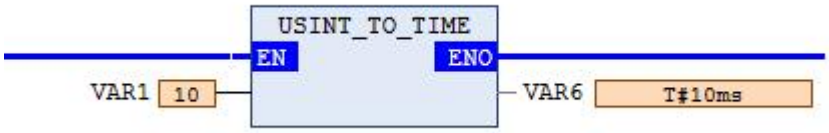
3.9.3 USINT_TO_<TYPE>——无符号短整型转换指令

无符号短整型转换指令用于把无符号短整型的输入数据转换为其它数据类型输出。

指令对应的输入和输出数据类型
输入的数据类型为 USINT。 输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、WORD、WSTRING。
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**VAR1**		USINT			
2	VAR	**VAR2**		REAL			
3	VAR	**VAR3**		DATE			
4	VAR	**VAR4**		STRING			
5	VAR	**VAR5**		DWORD			
6	VAR	**VAR6**		TIME			

编程语言	程序

梯形图（LD）	 VAR2 结果为 10  VAR4 结果为'10'  VAR5 结果为 10  VAR6 结果为 T#10ms
结构化文本（ST）	VAR2:=USINT_TO_REAL(VAR1); VAR2 结果为 10 VAR4:=USINT _TO_STRING(VAR1); VAR4 结果为'10' VAR5:=USINT _TO_DWORD(VAR1); VAR5 结果为 10 VAR6:=USINT _TO_TIME(VAR1); VAR6 结果为 T#10ms

3.9.4 SINT_TO_<TYPE>——短整型转换指令

短整型转换指令用于把短整型的输入数据转换为其它数据类型输出。

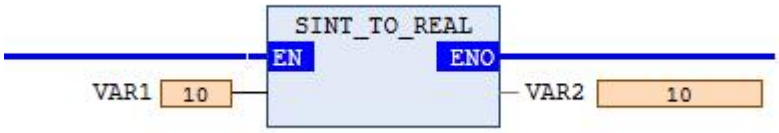
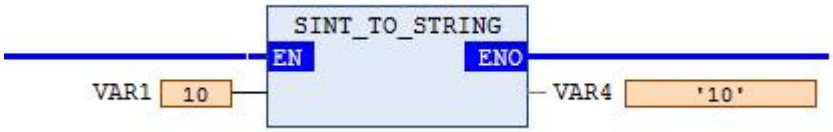
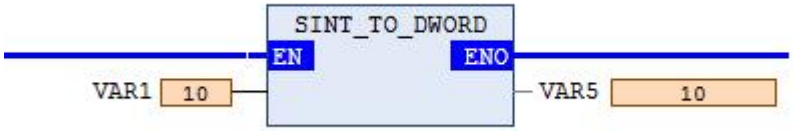
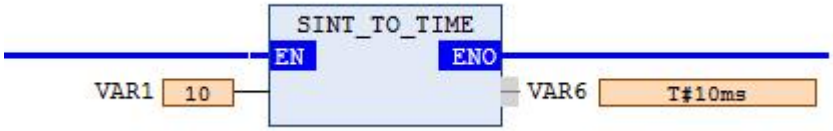
指令对应的输入和输出数据类型

输入的数据类型为 SINT。

输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**VAR1**		SINT			
2	VAR	**VAR2**		REAL			
3	VAR	**VAR3**		DATE			
4	VAR	**VAR4**		STRING			
5	VAR	**VAR5**		DWORD			
6	VAR	**VAR6**		TIME			

编程语言	程序
梯形图（LD）	 VAR2 结果为 10  VAR4 结果为'10'  VAR5 结果为 10  VAR6 结果为 T#10ms

结构化文本（ST）	VAR2:=SINT_TO_REAL(VAR1); VAR2 结果为 10 VAR4:=SINT _TO_STRING(VAR1); VAR4 结果为'10' VAR5:=SINT _TO_DWORD(VAR1); VAR5 结果为 10 VAR6:=SINT _TO_TIME(VAR1); VAR6 结果为 T#10ms

3.9.5 WORD_TO_<TYPE>——字类型转换指令

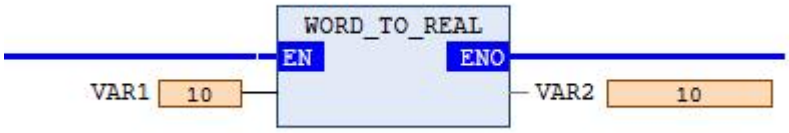
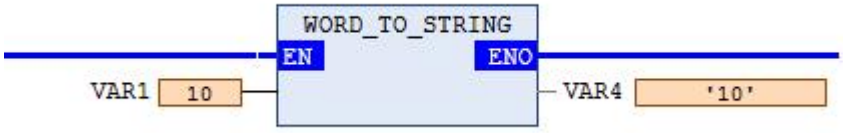
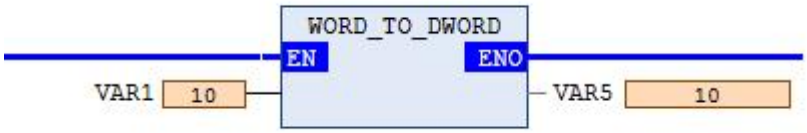
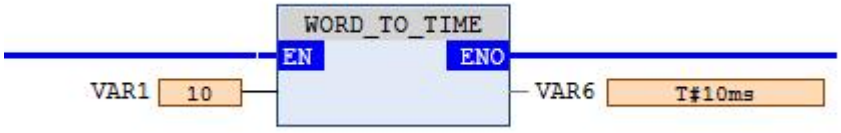
字类型转换指令用于把字类型的输入数据转换为其它数据类型输出。

指令对应的输入和输出数据类型
输入的数据类型为 WORD。 输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WSTRING。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		WORD			
2	VAR	VAR2		REAL			
3	VAR	VAR3		DATE			
4	VAR	VAR4		STRING			
5	VAR	VAR5		DWORD			
6	VAR	VAR6		TIME			

编程语言	程序

梯形图（LD）	 VAR2 结果为 10  VAR4 结果为'10'  VAR5 结果为 10  VAR6 结果为 T#10ms
结构化文本（ST）	VAR2:=WORD_TO_REAL(VAR1); VAR2 结果为 10 VAR4:=WORD _TO_STRING(VAR1); VAR4 结果为'10' VAR5:=WORD _TO_DWORD(VAR1); VAR5 结果为 10 VAR6:=WORD _TO_TIME(VAR1); VAR6 结果为 T#10ms

3.9.6 DWORD_TO_<TYPE>——双字类型转换指令

双字类型转换指令用于把双字类型的输入数据转换为其它数据类型输出。

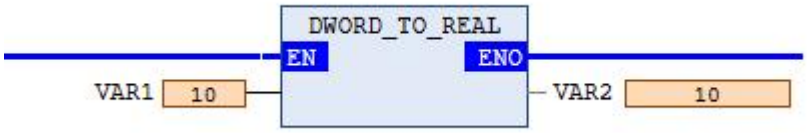
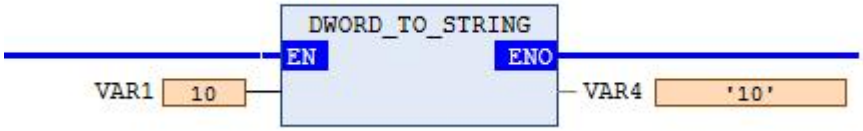
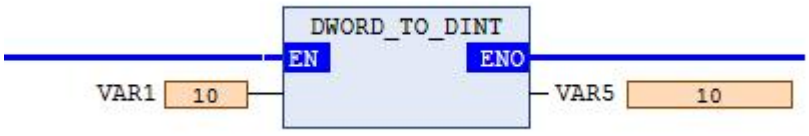
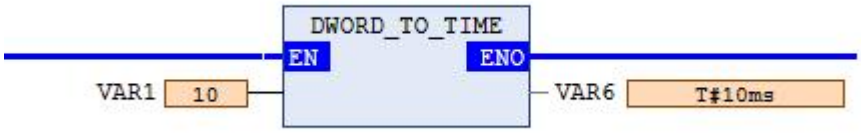
指令对应的输入和输出数据类型

输入的数据类型为 DWORD。

输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**VAR1**		WORD			
2	VAR	**VAR2**		REAL			
3	VAR	**VAR3**		DATE			
4	VAR	**VAR4**		STRING			
5	VAR	**VAR5**		DINT			
6	VAR	**VAR6**		TIME			

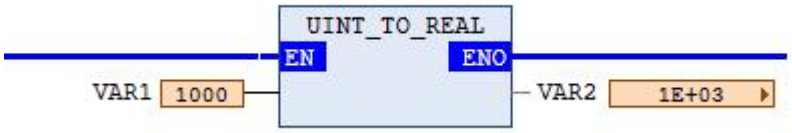
编程语言	程序
梯形图（LD）	 VAR2 结果为 10  VAR4 结果为'10'  VAR5 结果为 10  VAR6 结果为 T#10ms

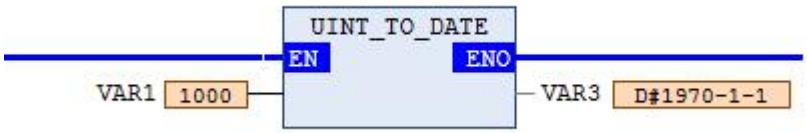
结构化文本（ST）	VAR2:=DWORD_TO_REAL(VAR1); VAR2 结果为 10 VAR4:=DWORD _TO_STRING(VAR1); VAR4 结果为'10' VAR5:=DWORD _TO_DINT(VAR1); VAR5 结果为 10 VAR6:=DWORD _TO_TIME(VAR1); VAR6 结果为 T#10ms

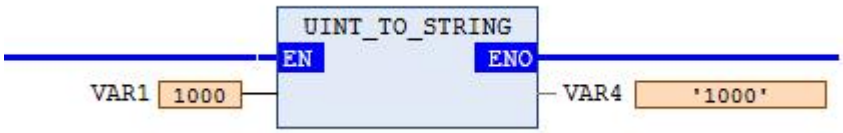
3.9.7 UINT_TO_<TYPE>——无符号整数类型转换指令

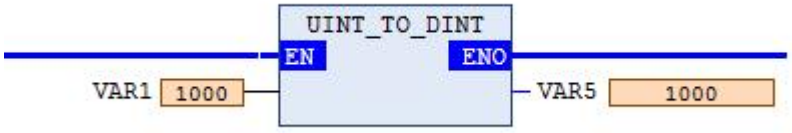
无符号整数类型转换指令用于把无符号整数类型的输入数据转换为其它数据类型输出。

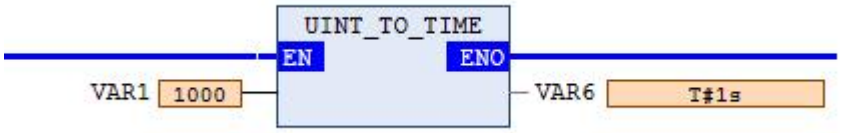
指令对应的输入和输出数据类型	
输入的数据类型为 UINT。 输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、USINT、WORD、WSTRING。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 1 VAR VAR1 UINT 2 VAR VAR2 REAL 3 VAR VAR3 DATE 4 VAR VAR4 STRING 5 VAR VAR5 DINT 6 VAR VAR6 TIME	
编程语言	程序

<table>
<tr>
<td>梯形图
（LD）</td>
<td>


VAR2 结果为 1000


VAR3 结果为 D#1970-1-1


VAR4 结果为'1000'


VAR5 结果为 1000


VAR6 结果为 T#1s
</td>
</tr>
<tr>
<td>结构化文本
（ST）</td>
<td>VAR2:=UINT_TO_REAL(VAR1);
VAR2 结 果 为 1000
VAR3:=UINT_TO_DATE(VAR1);
VAR3 结果 为D#1970-1-1
VAR4:=UINT_TO_STRING(Var1);
VAR4 结 果 为 '1000'
VAR5:=UINT_TO_DINT(Var1);
VAR5 结 果 为 1000</td>
</tr>
</table>

	VAR6:=UINT_TO_TIME(VAR1); VAR6 结果为 T#1s

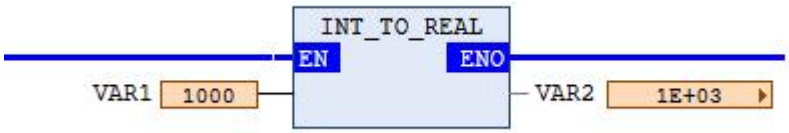
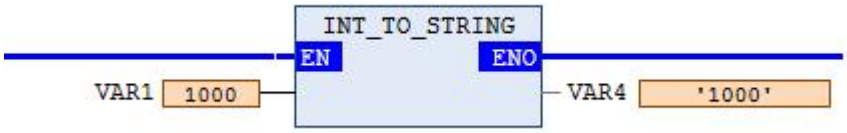
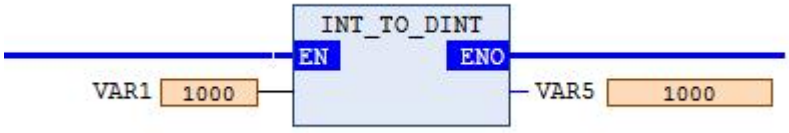
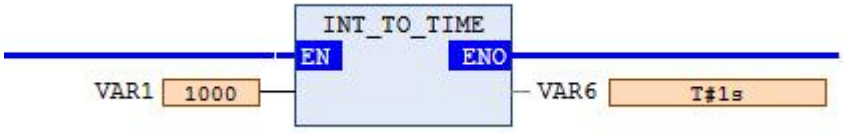
3.9.8 INT_TO_<TYPE>——整数类型转换指令

整数类型转换指令用于把整数类型的输入数据转换为其它数据类型输出。

指令对应的输入和输出数据类型	
输入的数据类型为 INT。 输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。	
变量定义	

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**VAR1**		INT			
2	VAR	**VAR2**		REAL			
3	VAR	**VAR3**		DATE			
4	VAR	**VAR4**		STRING			
5	VAR	**VAR5**		DINT			
6	VAR	**VAR6**		TIME			

编程语言	程序

梯形图（LD）	 VAR2 结果为 1000  VAR4 结果为'1000'  VAR5 结果为 1000  VAR6 结果为 T#1s
结构化文本（ST）	VAR2:=INT_TO_REAL(VAR1); VAR2 结果为 1000 VAR4:=INT _TO_STRING(VAR1); VAR4 结果为'1000' VAR5:=INT _TO_DINT(VAR1); VAR5 结果为 1000 VAR6:=INT _TO_TIME(VAR1); VAR6 结果为 T#1s

3.9.9 UDINT_TO_<TYPE>——无符号双整数类型转换指令

无符号双整数类型转换指令用于把无符号双整数类型的输入数据转换为其它数据类型输出。

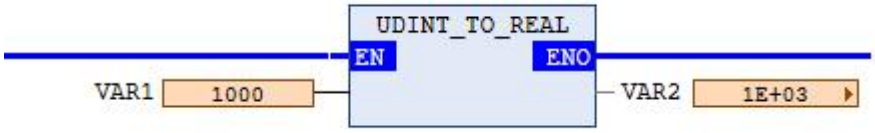
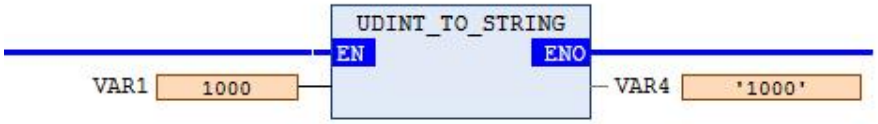
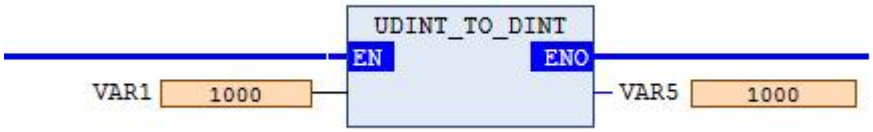
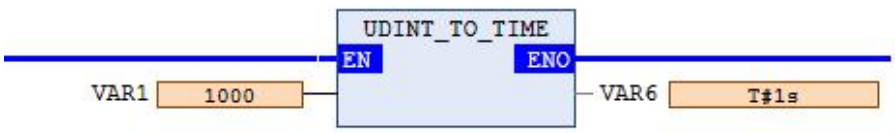
指令对应的输入和输出数据类型

输入的数据类型为 UDINT。

输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UINT、USINT、WORD、WSTRING。

变量定义

	类别	名称	地址	数据类型	初值	注释	特性
1	VAR	Var1		UDINT			
2	VAR	Var2		REAL			
3	VAR	Var3		DATE			
4	VAR	Var4		STRING			
5	VAR	Var5		DINT			
6	VAR	Var6		TIME			

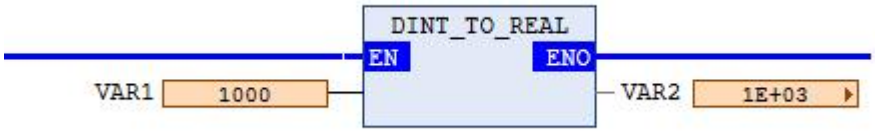
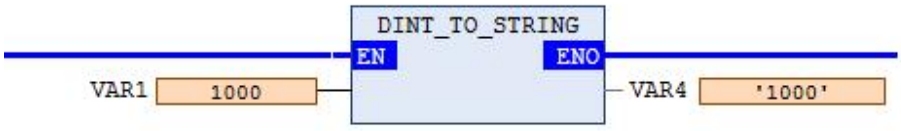
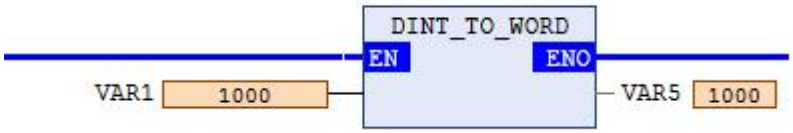
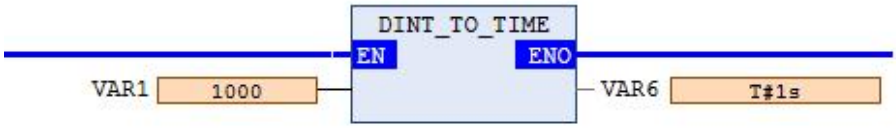
编程语言	程序
梯形图（LD）	 VAR2 结果为 1E+03  VAR4 结果为 '1000'  VAR5 结果为 1000  VAR6 结果为 T#1s
结构化文本	VAR2:=UDINT_TO_REAL(VAR1);

（ST）	VAR2 结果为 1000 VAR4:=UDINT _TO_STRING(VAR1); VAR4 结果为 '1000' VAR5:=UDINT _TO_DINT(VAR1); VAR5 结果为 1000 VAR6:=UDINT _TO_TIME(VAR1); VAR6 结果为 T#1s

3.9.10 DINT_TO_<TYPE>——双整数类型转换指令

双整数类型转换指令用于把双整数类型的输入数据转换为其它数据类型输出。

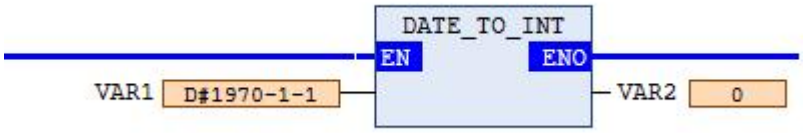
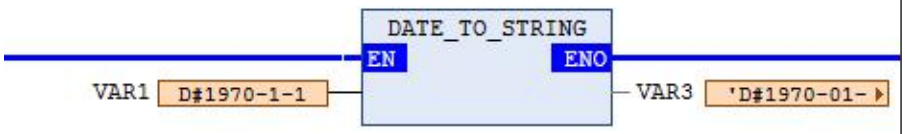
指令对应的输入和输出数据类型	
输入的数据类型为 DINT。 Z输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。	
变量定义	
	
编程语言	程序

梯形图（LD）	 VAR2 结果为 1E+03  VAR4 结果为'1000'  VAR5 结果为 1000  VAR6 结果为 T#1s
结构化文本（ST）	VAR2:=DINT_TO_REAL(VAR1); VAR2 结果为 1000 VAR4:=DINT _TO_STRING(VAR1); VAR4 结果为'1000' VAR5:=DINT _TO_WORD(VAR1); VAR5 结果为 1000 VAR6:=DINT _TO_TIME(VAR1); VAR6 结果为 T#1s

3.9.11 DATE_TO_<TYPE>——日期类型转换指令

日期类型的数据起始时间为 1970 年1 月1 日。日期类型转换指令用于把日期类型的输入数据转换为其它数据类型输出。

指令对应的输入和输出数据类型

输入的数据类型为 DATE。 输出的数据类型为 BOOL、BYTE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。	
变量定义	
类别 名称 地址 数据类型 初值 注释 属性 1 VAR VAR1 DATE 2 VAR VAR2 INT 3 VAR VAR3 STRING	
编程语言	**程序**
梯形图（LD）	 VAR2 结果为 0  VAR3 结果为'D#1970-01-01'
结构化文本（ST）	VAR2:=DATE_TO_INT(VAR1); VAR2 结果为 0 VAR3:=DATE _TO_STRING(VAR1); VAR3 结果为'D#1970-01-01'

当转换指令为 DATE_TO_BOOL 时，输入为 D#1970-01-01 时输出为 FALSE，否则输出为 TRUE。

3.9.12 TIME_TO_<TYPE>——时间类型转换指令

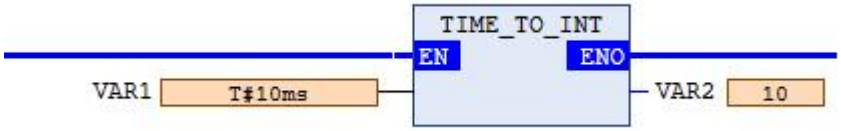
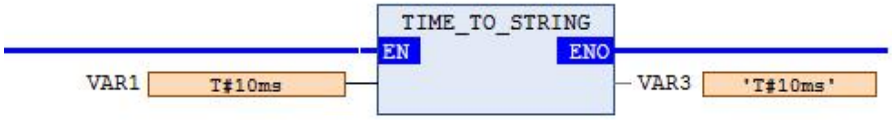
时间类型转换指令用于把时间类型的输入数据转换为其它数据类型输出。

指令对应的输入和输出数据类型
输入的数据类型为 TIME。 输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、

DWORD、INT、REAL、SINT、STRING、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		TIME			
2	VAR	VAR2		INT			
3	VAR	VAR3		STRING			

编程语言	程序
梯形图（LD）	 VAR2 结果为 10  VAR3 结果为'T#10ms'
结构化文本（ST）	VAR2:=TIME_TO_INT(VAR1); VAR2 结果为 10 VAR3:=TIME _TO_STRING(VAR1); VAR3 结果为'T#10ms'

当转换指令为 TIME_TO_BOOL 时，输入为 T#0ms 时输出为 FALSE，否则输出为 TRUE。

3.9.13 DT_TO_<TYPE>——日期时间类型转换指令

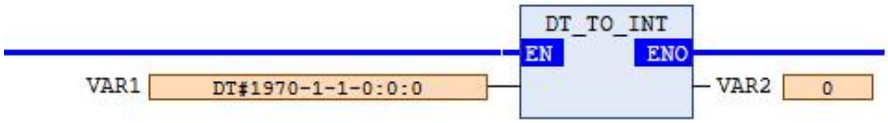
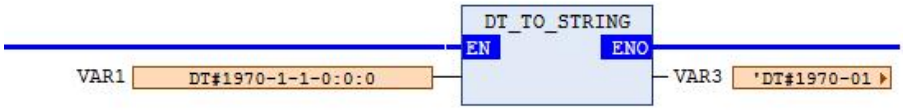
日期时间类型转换指令用于把日期时间类型的输入数据转换为其它数据类型输出。

指令对应的输入和输出数据类型
输入的数据类型为 DT。 输出的数据类型为 BOOL、BYTE、DATE、DINT、DWORD、INT、REAL、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、

WSTRING。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		DT			
2	VAR	VAR2		INT			
3	VAR	VAR3		STRING			

编程语言	程序
梯形图（LD）	 VAR2 结果为 0  VAR3 结果为'DT#1970-1-1-0:0:0'
结构化文本（ST）	VAR2:=DT_TO_INT(VAR1); VAR2 结果为 0 VAR3:=DT _TO_STRING(VAR1); VAR3 结果为'DT#1970-1-1-0:0:0'

当转换指令为 DT_TO_BOOL 时，输入为 DT#1970-1-1-0:0:0 时输出为 FALSE，否则输出为 TRUE。

3.9.14 TOD_TO_<TYPE>——时间日期类型转换指令

时间日期类型转换指令用于把时间日期类型的输入数据转换为其它数据类型输出。

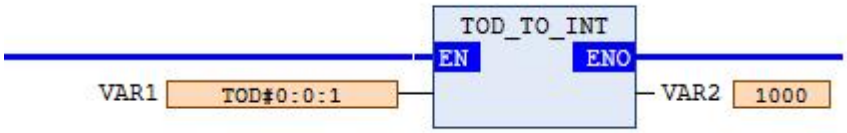
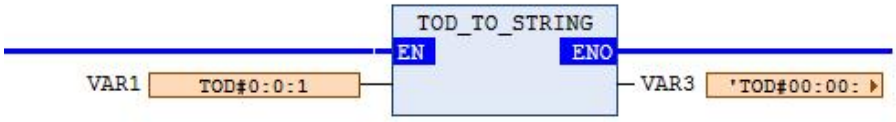
指令对应的输入和输出数据类型

输入的数据类型为 TOD。

输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、STRING、TIME、UDINT、UINT、USINT、WORD、WSTRING。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		TOD			
2	VAR	VAR2		INT			
3	VAR	VAR3		STRING			

编程语言	程序
梯形图（LD）	 VAR2 结果为1000  VAR3 结果为'TOD#00:00:01'
结构化文本（ST）	VAR2:=TOD_TO_INT(VAR1); VAR2 结果为1000 VAR3:=TOD _TO_STRING(VAR1); VAR3 结果为'TOD#00:00:01'

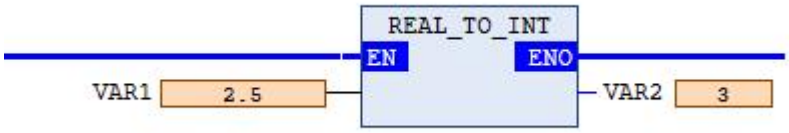
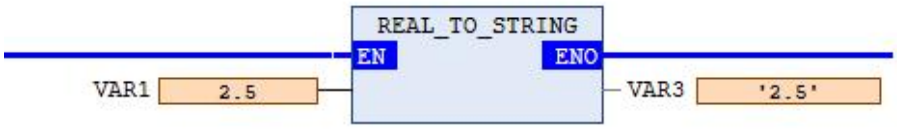
当转换指令为 TOD_TO_BOOL 时，输入为 TOD#00:00:00 时输出为 FALSE，否则输出为 TRUE。

3.9.15 REAL_TO_<TYPE>——实数类型转换指令

实数类型转换指令用于把实数类型的输入数据转换为其它数据类型输出。把实数转换为其它类型数据(字符型数据类型除外)前，先要对实数进行四舍五入，变成整数后再转成其它数据类型。

指令对应的输入和输出数据类型
输入的数据类型为 REAL。 输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、SINT、STRING、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		REAL			
2	VAR	VAR2		INT			
3	VAR	VAR3		STRING			

编程语言	程序
梯形图（LD）	 VAR2 结果为 3  VAR3 结果为'2.5'
结构化文本（ST）	VAR2:=REAL_TO_INT(VAR1); VAR2 结果为 3 VAR3:=REAL _TO_STRING(VAR1); VAR3 结果为'2.5'

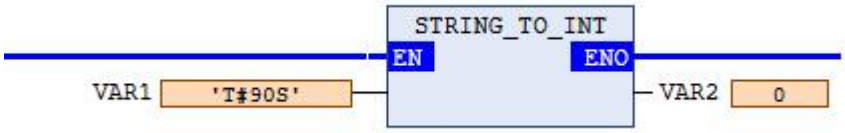
1、将数据类型从 REAL 转换成 BYTE、DINT、DWORD、INT、SINT、UDINT、UINT、USINT、WORD 时，如果 REAL 数的值超出了转换整数的范围， 将收到根据目标系统产生的一个不确定的结果。

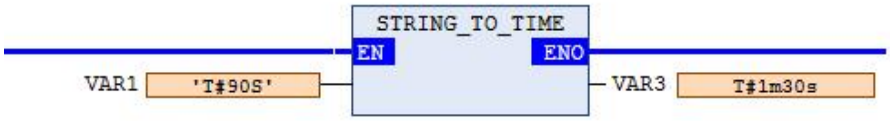
2、当转换指令为 REAL_TO_BOOL 时，输入为 0 时输出为 FALSE，否则输出为 TRUE。

3.9.16 STRING_TO_<TYPE>——字符类型转换指令

字符类型转换指令用于把字符类型的输入数据转换为其它数据类型输出。若被转换的字符串含有数值，转换结果取其数值。若被转换字符串不含数值，则转换结果为 0。

定义 STRING 操作数必须是目标类型的有效常量，不能包含无穷值、前缀、分组字符（_）和逗号。允许在一个数字后附加字符，例如：23xy，在一个数字之前添加字符是不允许的。输入数据必须是目标数据类型的有效值。

<table>
<tr><th colspan="2">指令对应的输入和输出数据类型</th></tr>
<tr><td colspan="2">输入的数据类型为 STRING。
输出的数据类型为 BOOL、BYTE、DATE、DATE_AND_TIME(DT)、DINT、DWORD、INT、REAL、SINT、TIME、TIME_OF_DAY(TOD)、UDINT、UINT、USINT、WORD、WSTRING。</td></tr>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2">
<table>
<tr><th></th><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>1</td><td>VAR</td><td>VAR1</td><td></td><td>STRING</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>VAR</td><td>VAR2</td><td></td><td>INT</td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>VAR</td><td>VAR3</td><td></td><td>TIME</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td>梯形图
（LD）</td><td> 

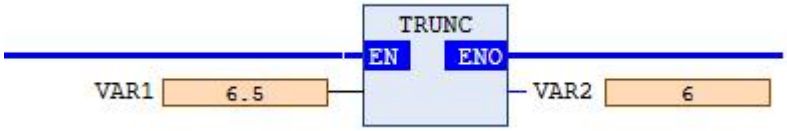
VAR2 结果为 0
 

VAR3 结果为'T#1m30s'</td></tr>
<tr><td>结构化文本
（ST）</td><td>VAR2:=STRING_TO_INT(VAR1);
VAR2 结果为 0
VAR3:=STRING _TO_TIME(VAR1);
VAR3 结果为'T#1m30s'</td></tr>
</table>

将 STRING 数据类型的字符正确的转换为 TIME 数据类型，需要把区分时间的 d、h、m、s、ms 等字符修改为小写。例如'T#100s'可以转换为'T#1m40s'， 而'T#100S'则转换为'T#0s'。

3.9.17 TRUNC——截短转换指令

截断转换指令用于截取输入数据的实数整数部分，忽略实数的小数部分，转换为其它数据类型输出。

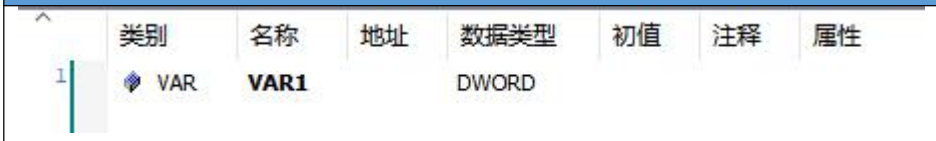
指令对应的输入和输出数据类型	
输入的数据类型为 REAL。 输出的数据类型为 DINT 、DWORD 、UDINT。	
变量定义	
	
编程语言	程序
梯形图 （LD）	 VAR2 结果为 6
结构化文本 （ST）	VAR2:=TRUNC(VAR1); VAR2 结果为 6

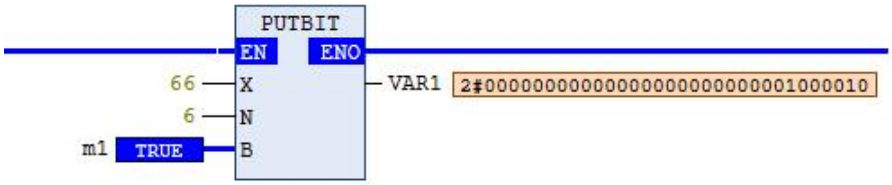
该指令的功能与四舍五入不同，若要通过四舍五入取整，请参照指令 REAL_TO_INT。

3.10 位/字节操作指令（Util.compiled-library）

3.10.1 PUTBIT——位赋值指令

若要对输入数据的某一位进行赋值，可使用位赋值指令 PUTBIT。

指令对应的输入和输出数据类型	
输入 X 的数据类型为 DWORD，输入 N 的数据类型为 BYTE，输入 B 的数据类型为 BOOL。 输出的数据类型为 DWORD。	
变量定义	
	
编程语言	程序

梯形图（LD）	
结构化文本（ST）	VAR1:=PUTBIT(66,6,TRUE); 结　果　VAR1　为 2#00000000000000000000000001000010

3.10.2 UNPACK——位拆分指令

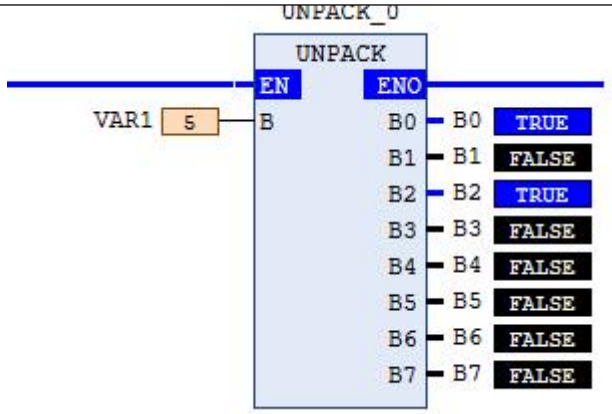
若要将字节型输入数据拆分成 8 个独立的位，可使用位拆分指令 UNPACK。位拆分指令 UNPACK 是位整合指令 PACK 的逆运算指令。

指令对应的输入和输出数据类型
输入的数据类型为 BYTE。 输出 B0-B7 的数据类型为 BOOL。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**VAR1**		BYTE			
2	VAR	**UNPACK_0**		UNPACK			
3	VAR	**B0**		BOOL			
4	VAR	**B1**		BOOL			
5	VAR	**B2**		BOOL			
6	VAR	**B3**		BOOL			
7	VAR	**B4**		BOOL			
8	VAR	**B5**		BOOL			
9	VAR	**B6**		BOOL			
10	VAR	**B7**		BOOL			

编程语言	程序

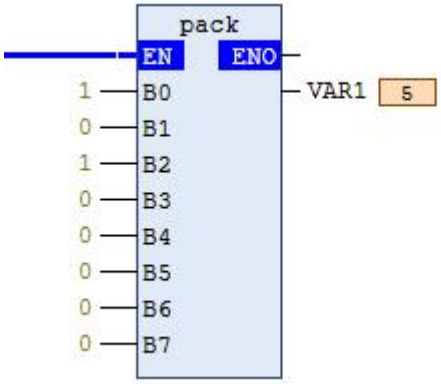
梯形图 （LD）	
结构化文本 （ST）	UNPACK_0(B:=VAR1); B0:=UNPACK_0.B0; B1:=UNPACK_0.B1; B2:=UNPACK_0.B2; B3:=UNPACK_0.B3; B4:=UNPACK_0.B4; B5:=UNPACK_0.B5; B6:=UNPACK_0.B6; B7:=UNPACK_0.B7; VAR1=2#00000101 结果 VarBOOL0 为 TRUE 结果 VarBOOL1 为 FALSE 结果 VarBOOL2 为 TRUE 结果 VarBOOL3 为 FALSE 结果 VarBOOL4 为 FALSE 结果 VarBOOL5 为 FALSE 结果 VarBOOL6 为 FALSE 结果 VarBOOL7 为 FALSE

3.10.3 PACK——位整合指令

若要将多个布尔型输入数据合成为字节型数据，可使用位整合指令 PACK。

指令对应的输入和输出数据类型
输入 B0-B7 的数据类型为 BOOL。 输出的数据类型为 BYTE。

变量定义						
	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		BYTE			

编程语言	程序
梯形图（LD）	 结果 VAR1 为 2，即 2#00000101
结构化文本（ST）	. VAR1:=PACK(1,0,1,0,0,0,0,0); 结果 VAR1 为 5，即2#00000101

在应用该指令时需注意，在 LD 语言中，字节的存储是由 B7 作为高位、B0 作为低位进行存储的。但在 IL 语言和 ST 语言中，先声明的部分是低位， 结尾部分是高位，切勿弄错顺序。

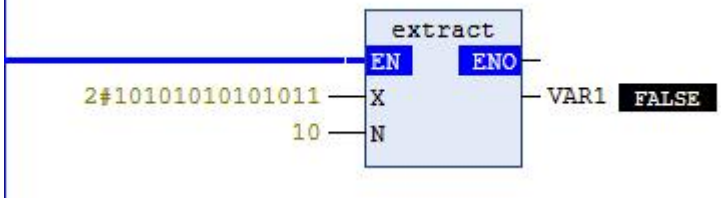
3.10.4 EXTRACT——位提取指令

若要从输入数据中提取对应位的值，可使用位提取指令 EXTRACT。

指令对应的输入和输出数据类型
输入 X 的数据类型为 DWORD，输入 N 的数据类型为 BYTE。 输出的数据类型为 BOOL。

变量定义						
	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		Bool			

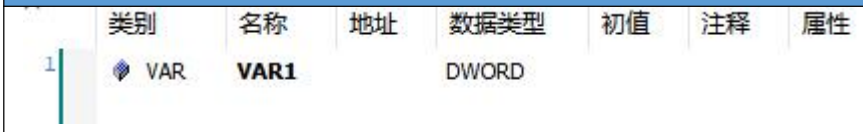
编程语言	程序

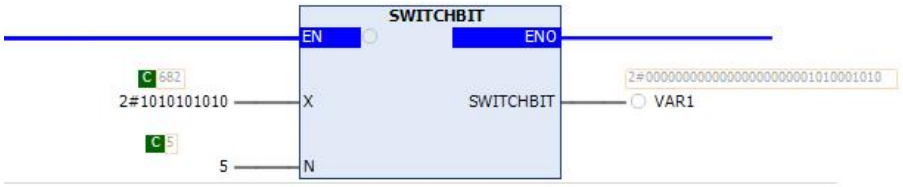
梯形图 （LD）	 结果 VAR1 为 FALSE
结构化文本 （ST）	VAR1:=PUTBIT(2#10101010101011,10); 结果 VAR1 为 FALSE

该指令的功能是提取输入变量 X 的第 N 位输出，N 从零开始计算。

3.10.5 SWITCHBIT——位取反指令

若要从输入数据中取反对应位的值，可使用位取反指令 SWITCHBIT。

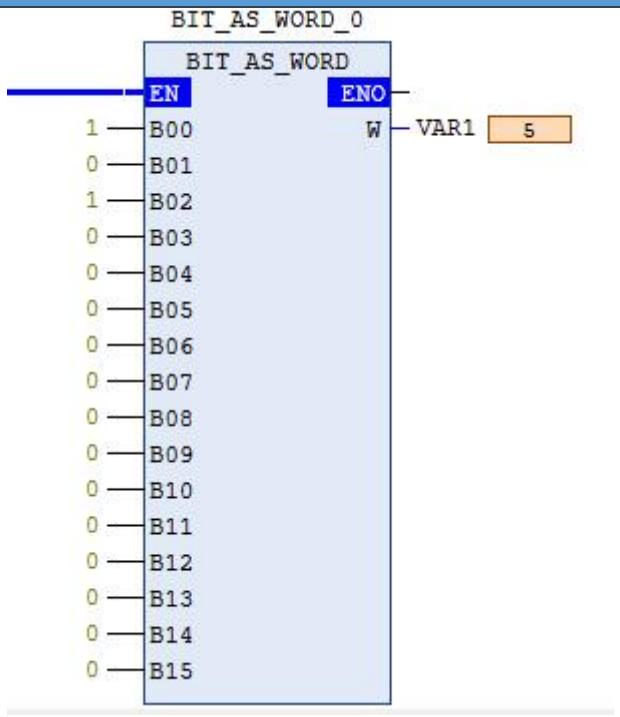
指令对应的输入和输出数据类型
输入 X 的数据类型为 DWORD，输入 N 的数据类型为 BYTE。 输出的数据类型为 DWORD。
变量定义


编程语言	程序
梯形图 （LD）	 结果 VAR1 为 2#00000000000000000000001010001010
结构化文本 （ST）	VAR1:=SWITCHBIT(X:=2#1010101010, N:=5); 结果VAR1 为2#00000000000000000000001010001010

该指令的功能是取反输入变量 X 的第 N 位输出，N 从零开始计算。

3.10.6 BIT_AS_WORD——位整合字指令

若要将多个布尔型输入数据合成为字类型数据，可使用位整合字指令 BIT_AS_WORD。

<table>
<tr><th colspan="2">指令对应的输入和输出数据类型</th></tr>
<tr><td colspan="2">输入 B00-B15 的数据类型为 BOOL。输出的数据类型为 WORD。</td></tr>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2">
<table>
<tr><th></th><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>1</td><td>VAR</td><td>VAR1</td><td></td><td>WORD</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>VAR</td><td>BIT_AS_WORD_0</td><td></td><td>BIT_AS_WORD</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td>梯形图
（LD）</td><td> 

结果 VAR1 为 5，即 2#0000000000000101</td></tr>
<tr><td>结构化文本
（ST）</td><td>BIT_AS_WORD0(
B00:=1,
B01:=0,
B02:=1,
B03:=0,
B04:=0,
B05:=0,
B06:=0,
B07:=0,
B08:=0,
B09:=0,</td></tr>
</table>

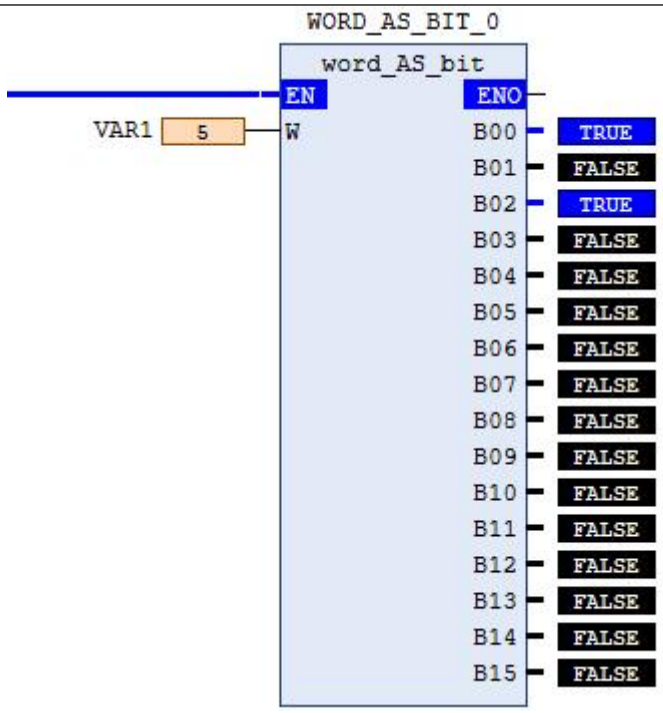
	B10:=0, B11:=0, B12:=0, B13:=0, B14:=0, B15:=0, W=>VAR1); 结果 VAR1 为 5，即 2#0000000000000101

在应用该指令时需注意，在 LD 语言中，字节的存储是由 B15 作为高位、B00 作为低位进行存储的。但在 IL 语言和 ST 语言中，先声明的部分是低位， 结尾部分是高位，切勿弄错顺序。

3.10.7 WORD_AS_BIT——字提取位指令

若要将输入字类型的数据中提取对应位的值，可使用字提取位指令 WORD_AS_BIT。

指令对应的输入和输出数据类型						
输入 W 的数据类型为 WORD。 输出的数据类型为 BOOL。						
变量定义						
	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**VAR1**		WORD			
2	VAR	**WORD_AS_BIT_0**		word_AS_bit			
编程语言	**程序**						

梯形图 （LD）	 输入 VAR1 为 5，结果为 2#0000000000000101
结构化文本 （ST）	WORD_AS_BIT_0(W:=VAR1 , B00=> , B01=> , B02=> , B03=> , B04=> , B05=> , B06=> , B07=> , B08=> , B09=> , B10=> , B11=> , B12=> , B13=> , B14=> , B15=>);

	输入 VAR1 为 5，结果为 2#0000000000000101

该指令的功能是提取输入变量 W，输出为位。

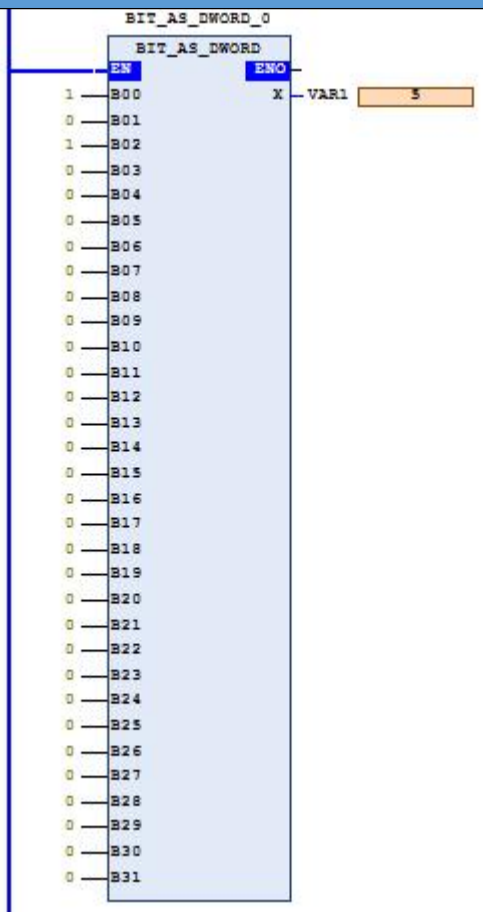
3.10.8 BIT_AS_DWORD——位整合双字指令

若要将多个布尔型输入数据合成为双字型数据，可使用位整合双字指令 BIT_AS_DWORD。

指令对应的输入和输出数据类型
输入 B00-B31 的数据类型为 BOOL。 输出的数据类型为 DWORD。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	VAR1		DWORD			
2	VAR	BIT_AS_DWORD_0		BIT_AS_DWORD			

编程语言	程序
梯形图（LD）	 结果 VAR1 为 5，即 2#00000000000000000000000000000101

结构化文本（ST）	BIT_AS_WORD0(B00:=1, B01:=0, B02:=1, B03:=0, B04:=0, B05:=0, B06:=0, B07:=0, B08:=0, B09:=0, B10:=0, B11:=0, B12:=0, B13:=0, B14:=0, B15:=0, B16:=0, B17:=0, B18:=0, B19:=0, B20:=0, B21:=0, B22:=0, B23:=0, B24:=0, B25:=0, B26:=0, B27:=0, B28:=0, B29:=0, B30:=0, B31:=0,

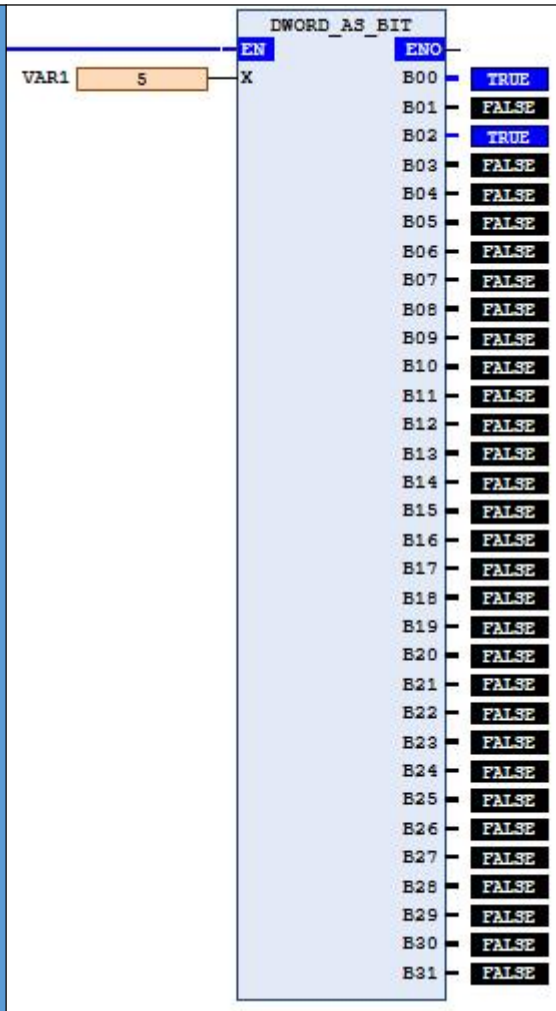
	W=>VAR 1); 结果 VAR1 为 5，即 2#00000000000000000000000000000101

在应用该指令时需注意，在 LD 语言中，字节的存储是由 B31 作为高位、B00 作为低位进行存储的。但在 IL 语言和 ST 语言中，先声明的部分是低位， 结尾部分是高位，切勿弄错顺序。

3.10.9 DWORD_AS_BIT——双字提取位指令

若要将输入双字型的数据中提取对应位的值，可使用双字提取位指令 DWORD_AS_BIT。

指令对应的输入和输出数据类型	
输入 X 的数据类型为 DWORD。 输出的数据类型为 BOOL。	
变量定义	
	
编程语言	程序

梯形图 （LD）	 输入 VAR1 为5，结果为 2#00000000000000000000000000000101
结构化文本 （ST）	DWORD_AS_BIT_0(W:=Var1 , B00=> , B01=> , B02=> , B03=> , B04=> , B05=> , B06=> , B07=> , B08=> , B09=> , B10=> ,

	B11=> , B12=> , B13=> , B14=> , B15=> , B16=> , B17=> , B18=> , B19=> , B20=> , B21=> , B22=> , B23=> , B24=> , B25=> , B26=> , B27=> , B28=> , B29=> , B30=> , B31=>); 输入 VAR1 为5，结果为 2#00000000000000000000000000000101

该指令的功能是提取输入变量 W，输出为位。

3.11 BCD 码转换指令（Util）

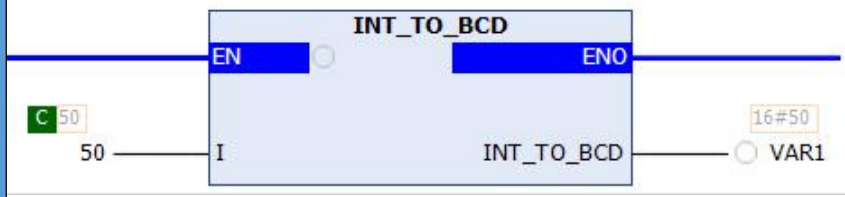
工程应用中大量采用 BCD 码，其数据存储格式为：每一字节 BCD 码代表 0 至 99 之间的一个两位整数，该整数的个位数存储在字节的第 3 位至第 0 位，该整数的十位数存储在字节的第 7 位至第 4 位。尽管 BCD 码的格式与 16 进制的格式比较相似，但它们的范围不同，BCD 码的范围为 0-99，而 16 进制的范围为 0-FF。

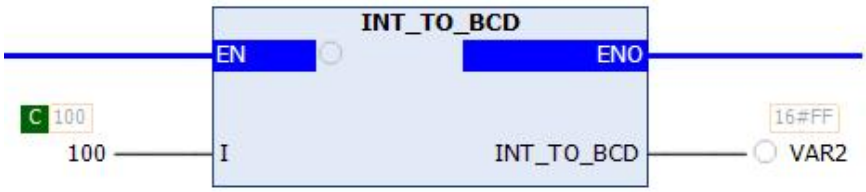
十进制 39 转换成 BCD 码，3 的二进制是 0011，9 的二进制是 1001，那么 39 转换 BCD 码为 00111001。十进制数 88 表示成 BCD 码为 10001000，表示

成二进制为 2#0101 1000，表示成十六进制为 16#58。

3.11.1 INT_TO_BCD——整数型转 BCD 码指令

整数型转 BCD 码指令用于把整数类型的输入数据转换为 BCD 码输出。

<table>
<tr><th colspan="2">指令对应的输入和输出数据类型</th></tr>
<tr><td colspan="2">输入的数据类型为 INT。
输出的数据类型为 BYTE。</td></tr>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2">
<table>
<tr><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>VAR</td><td>VAR1</td><td></td><td>INT</td><td></td><td></td><td></td></tr>
<tr><td>VAR</td><td>VAR2</td><td></td><td>INT</td><td></td><td></td><td></td></tr>
<tr><td>VAR</td><td>VAR3</td><td></td><td>INT</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td rowspan="2">梯形图
（LD2）</td><td> 

结果 VAR1 为16#50
 

错误！结果 VAR2 为 16#FF</td></tr>
<tr><td>结构化文本
（ST）</td><td>VAR1:=INT_TO_BCD(50);
结果 VAR1 为 16#50
VAR2:=INT_TO_BCD(100);
错误！结果 VAR2 为 16#FF</td></tr>
</table>

该指令对应的输入为 INT 型，输出为 BYTE 型，是转换后的 BCD 码值。若输入的数据不能转换成 BCD 码时，输出结果为 255，即 16#FF。

3.11.2 BCD_TO_INT——BCD 码转整数型指令

BCD 码转整数型指令用于把 BCD 码的输入数据转换为整数类型输出。

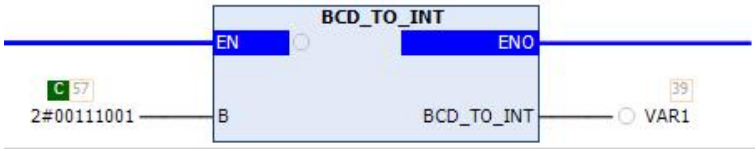
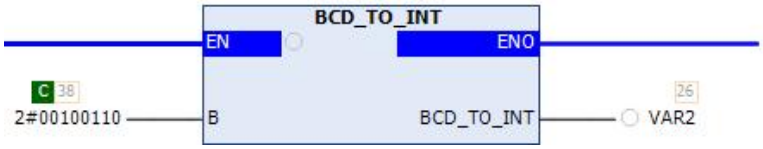
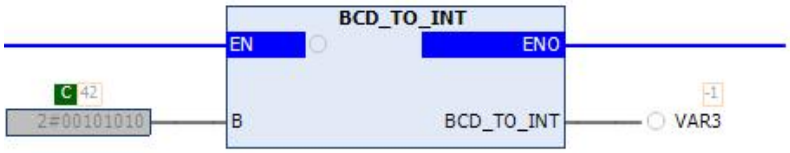
指令对应的输入和输出数据类型

输入的数据类型为 BYTE。

输出的数据类型为 INT。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		INT			
VAR	VAR2		INT			
VAR	VAR3		INT			

编程语言	程序
梯形图（LD2）	 结果 VAR1 为39  结果 VAR2 为26  输出-1，因为后四位 1010 不是 BCD 码格式
结构化文本（ST）	VAR1:=BCD_TO_INT(2#00111001); 结果 VAR1 为 39 VAR2:= BCD_TO_INT(2#00100110); 结果 VAR2 为 26 VAR3:= BCD_TO_INT(2#00101010); 输出-1，因为后四位 1010 不是 BCD 码格式

该函数用于将一个 BCD 位转化为一个 INT 值，如果要转换的字节不是 BCD 格式，那么结果为-1。

3.11.3 WORD_TO_BCD——字类型转 BCD 码指令

字类型转 BCD 码指令用于把字类型的输入数据转换为 BCD 码输出。

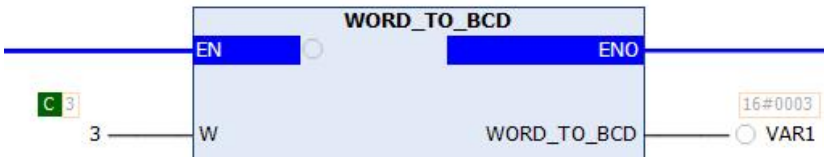
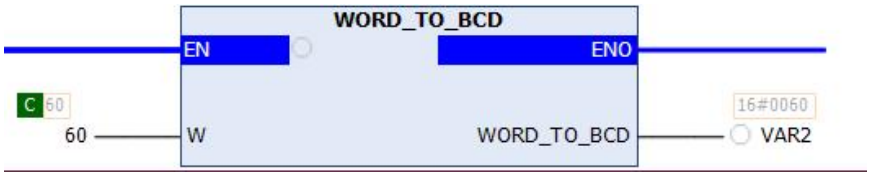
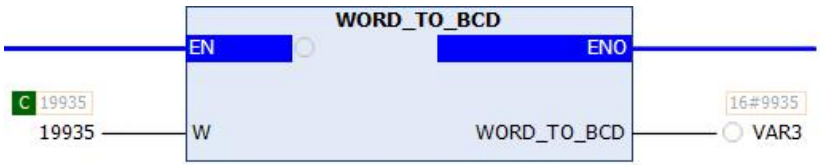
指令对应的输入和输出数据类型

输入的数据类型为 WORD。

输出的数据类型为 WORD。

变量定义

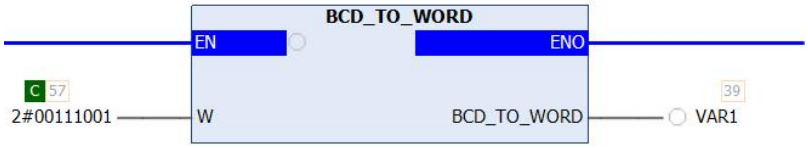
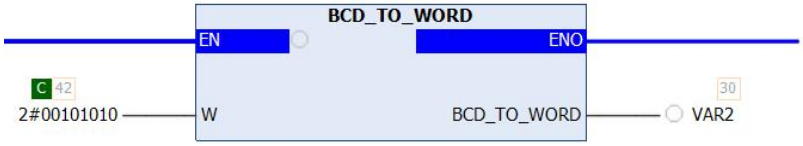
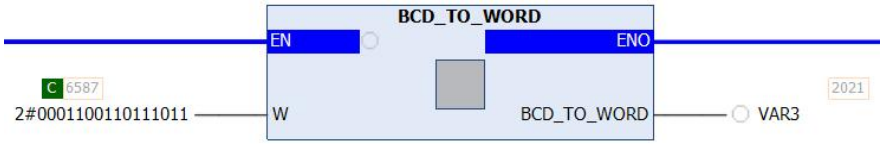
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		WORD			
VAR	**VAR2**		WORD			
VAR	**VAR3**		WORD			

编程语言	程序
梯形图（LD2）	 结果 VAR1 为16#0003  结果 VAR2 为16#0060  错误！结果 VAR3 为 16#9935
结构化文本（ST）	VAR1:=WORD_TO_BCD(3); 结果 VAR1 为 16#0003 VAR2:=WORD_TO_BCD(60); 结果 VAR2 为 16#0060 VAR3:=WORD_TO_BCD(19935); 错误！结果 VAR3 为 16#9935

该指令对应的输入为 WORD 型，输出为 WORD 型，是转换后的 BCD 码值。输入数据的范围为 0~9999，若输入的数据超过 9999 不能转换成 BCD码时，输出结果为后四位。

3.11.4 BCD_TO_WORD——BCD 码转字类型指令

BCD 码转字类型指令用于把 BCD 码的输入数据转换为字类型输出。

指令对应的输入和输出数据类型	
输入的数据类型为 WORD。 输出的数据类型为 WORD。	
变量定义	
类别 / 名称 / 地址 / 数据类型 / 初值 / 注释 / 属性 VAR　VAR1　WORD VAR　VAR2　WORD VAR　VAR3　WORD	
编程语言	**程序**
梯形图 （LD2）	 结果 VAR1 为39  结果 VAR2 为30  结果 VAR3 为 2021
结构化文本 （ST）	VAR1:=BCD_TO_WORD(2#00111001); 结果 VAR1 为 39 VAR2:= BCD_TO_WORD(2#00101010); 结果 Var2 为 30 VAR3:= BCD_TO_WORD(2#0001100110111011); 结果 VAR3 为 2021

3.11.5 BYTE_TO_BCD——字节型转 BCD 码指令

字节型转 BCD 码指令用于把字节型的输入数据转换为 BCD 码输出。

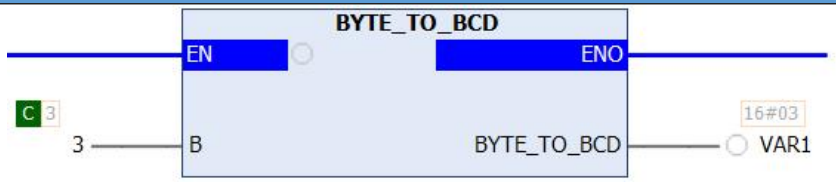
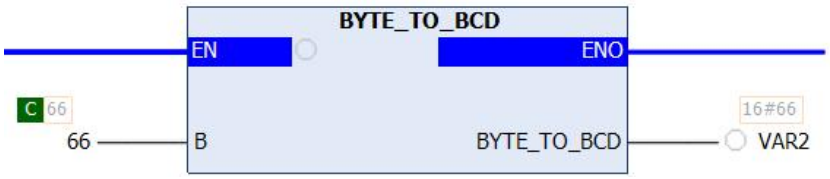
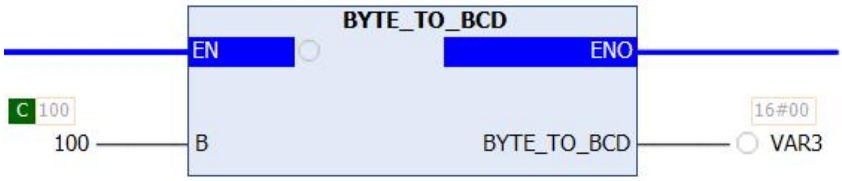
指令对应的输入和输出数据类型

输入的数据类型为 BYTE。

输出的数据类型为 BYTE。

变量定义

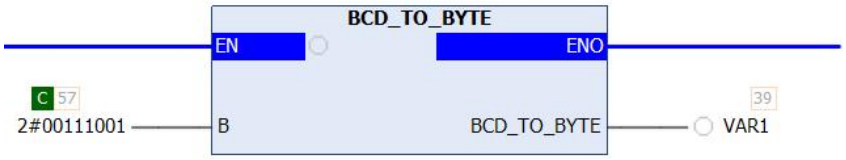
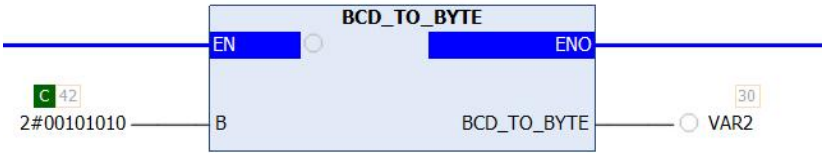
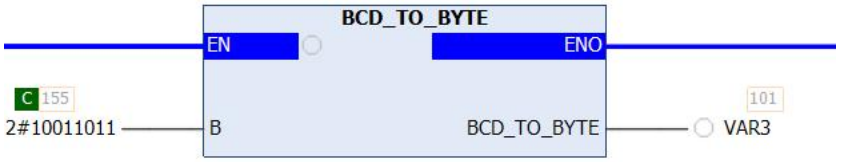
类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		BYTE			
VAR	VAR2		BYTE			
VAR	VAR3		BYTE			

编程语言	程序
梯形图（LD2）	 结果 VAR1 为 16#03  结果 VAR2 为 16#66  错误！结果 VAR3 为 16#00
结构化文本（ST）	VAR1:=BYTE_TO_BCD(3); 结果 VAR1 为 16#03 VAR2:=BYTE_TO_BCD(66); 结果 VAR2 为 16#66 VAR3:=BYTE_TO_BCD(100); 错误！结果 VAR3 为 16#00

该指令对应的输入为 BYTE 型，输出为 BYTE 型，是转换后的 BCD 码值。输入数据的范围为 0~99，若输入的数据超过 99 不能转换成 BCD 码时， 输出结果为后两位。

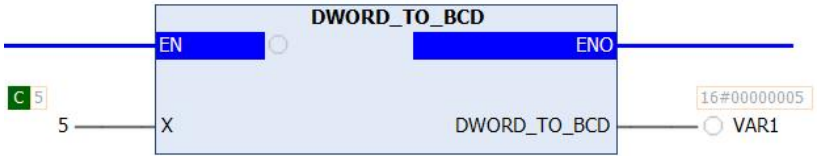
3.11.6 BCD_TO_BYTE——BCD 码转字节型指令

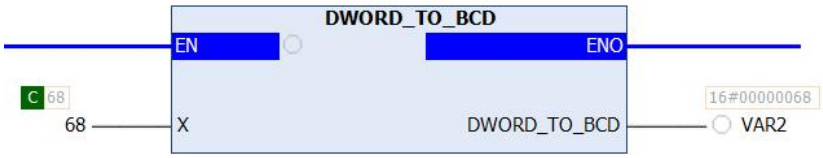
BCD 码转字类型指令用于把 BCD 码的输入数据转换为字节型输出。

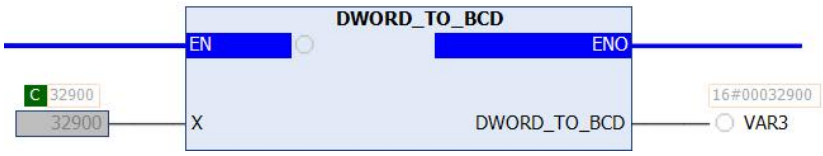
指令对应的输入和输出数据类型	
输入的数据类型为 BYTE。 输出的数据类型为 BYTE。	
变量定义	
类别 / 名称 / 地址 / 数据类型 / 初值 / 注释 / 属性 VAR　VAR1　BYTE VAR　VAR2　BYTE VAR　VAR3　BYTE	
编程语言	**程序**
梯形图 （LD2）	 结果 VAR1 为39  结果 VAR2 为30  结果 VAR3 为 101
结构化文本 （ST）	VAR1:=BCD_TO_BYTE(2#00111001); 结果 VAR1 为 39 VAR2:= BCD_TO_BYTE(2#00101010); 结果 VAR2 为 30 VAR3:= BCD_TO_BYTE(2#10011011); 结果 VAR3 为 101

3.11.7 DWORD_TO_BCD——双字型转 BCD 码指令

双字型转 BCD 码指令用于把双字型的输入数据转换为 BCD 码输出。

<table>
<tr><th colspan="2">指令对应的输入和输出数据类型</th></tr>
<tr><td colspan="2">输入的数据类型为 DWORD。
输出的数据类型为 DWORD。</td></tr>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2">
<table>
<tr><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>VAR</td><td>VAR1</td><td></td><td>DWORD</td><td></td><td></td><td></td></tr>
<tr><td>VAR</td><td>VAR2</td><td></td><td>DWORD</td><td></td><td></td><td></td></tr>
<tr><td>VAR</td><td>VAR3</td><td></td><td>DWORD</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td>梯形图
（LD2）</td><td>


结果 VAR1 为16#00000005


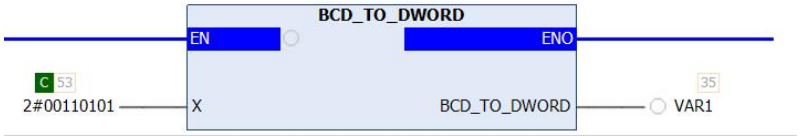
结果 VAR2 为16#00000068


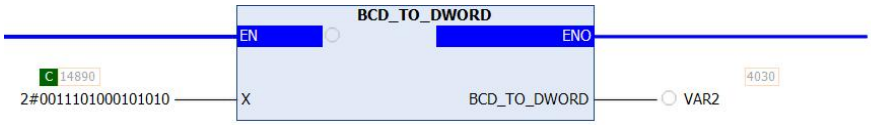
结果 VAR3 为16#00032900
</td></tr>
<tr><td>结构化文本
（ST）</td><td>VAR1:=WORD_TO_BCD(3);
结果 VAR1 为 16#0003
VAR2:=WORD_TO_BCD(60);
结果 VAR2 为 16#0060
VAR3:=WORD_TO_BCD(19935);
结果 VAR3 为 16#00032900</td></tr>
</table>

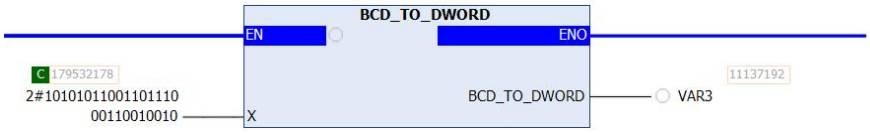
该指令对应的输入为 WORD 型，输出为 WORD 型，是转换后的 BCD 码值。输入数据的范围为 0~99999999，若输入的数据超过 99999999 不能转换成 BCD 码时，输出结果为后八位。

3.11.8 BCD_TO_DWORD——BCD 码转双字型指令

BCD 码转双字型指令用于把 BCD 码的输入数据转换为双字型输出。

<table>
<tr><th colspan="2">指令对应的输入和输出数据类型</th></tr>
<tr><td colspan="2">输入的数据类型为 DWORD。
输出的数据类型为 DWORD。</td></tr>
<tr><th colspan="2">变量定义</th></tr>
<tr><td colspan="2">
<table>
<tr><th>类别</th><th>名称</th><th>地址</th><th>数据类型</th><th>初值</th><th>注释</th><th>属性</th></tr>
<tr><td>VAR</td><td>VAR1</td><td></td><td>DWORD</td><td></td><td></td><td></td></tr>
<tr><td>VAR</td><td>VAR2</td><td></td><td>DWORD</td><td></td><td></td><td></td></tr>
<tr><td>VAR</td><td>VAR3</td><td></td><td>DWORD</td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td>梯形图
（LD2）</td><td>


结果 VAR1 为 35


结果 VAR2 为 4030


结果 VAR3 为 11137192
</td></tr>
<tr><td>结构化文本
（ST）</td><td>VAR1:=BCD_TO_DWORD(2#00110101);
结果 VAR1 为 35
VAR2:= BCD_TO_DWORD(2#0011101000101010);
结果 VAR2 为 4030
VAR3:= BCD_TO_DWORD(2#10101011001101110000110010010);
结果 VAR3 为 11137192</td></tr>
</table>

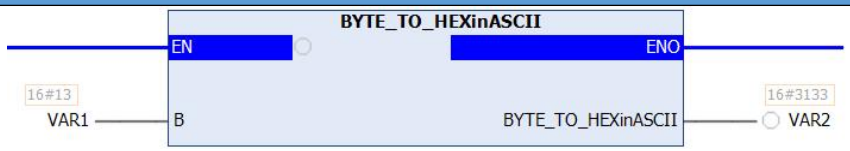
3.12 ASCII 码转换指令（Util）

3.12.1 BYTE_TO_HEXinASCII——字节型转换为 ASCII 码指令

将 1 个字节的高 4 位与低 4 位分别转化为 ASCII 码并存储在 1 个字的高 8 位与低 8 位中。

指令对应的输入和输出数据类型	
输入的数据类型为 BYTE。 输出的数据类型为 WORD。	
变量定义	

表达式	类型	值	准备值	地址	注释
VAR1	BYTE	16#13			
VAR2	WORD	16#3133			

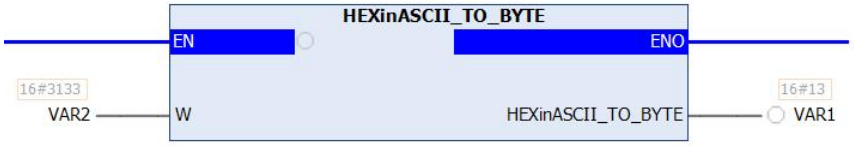
编程语言	程序
梯形图 （LD2）	 VAR2 结果为 16#3133
结构化文本 （ST）	VAR2:=BYTE_TO_HEXinASCII(B:=VAR1); VAR2 结果为 16#3133

3.12.2 HEXinASCII_TO_BYTE——ASCII 码转换为字节型指令

将 1 个字的高 8 位与低 8 位 ASCII 码分别转化为 1 个字节的高 4 位与低 4 位。

指令对应的输入和输出数据类型	
输入的数据类型为 WORD。输出的数据类型为 BYTE。	
变量定义	

表达式	类型	值	准备值	地址	注释
VAR1	BYTE	16#13			
VAR2	WORD	16#3133			

编程语言	程序
梯形图 （LD2）	 VAR1 结果为 16#13
结构化文本 （ST）	VAR1:=HEXinASCII_TO_BYTE(W:=VAR2); VAR1 结果为 16#13

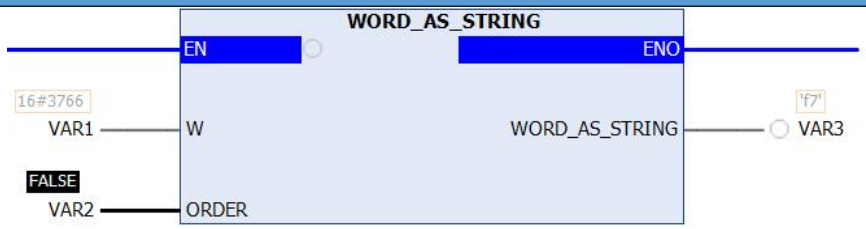
3.12.3 WORD_AS_STRING——字类型转 ASCII 码指令

将 1 个字的高 8 位与低 8 位分别转化为 ASCII 码并存储在字符类型中。

指令对应的输入和输出数据类型
输入的数据类型为 WORD，高 8 位与低 8 位的 16 进制数范围分别为（20H-7FH）。 输出的数据类型为 STRING(2)。 该指令中控制位 ORDER 数据类型为 BOOL，将转换的高低字符对调。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		WORD			
VAR	**VAR2**		BOOL			
VAR	**VAR3**		STRING(2)			

编程语言	程序
梯形图（LD2）	 VAR1 结果为’f7’
结构化文本（ST）	VAR3:=WORD_AS_STRING(W:=VAR1, ORDER:=VAR2); VAR1 结果为’f7’

3.13 信号发生器指令（Util）

3.13.1 GEN——典型周期信号发生器

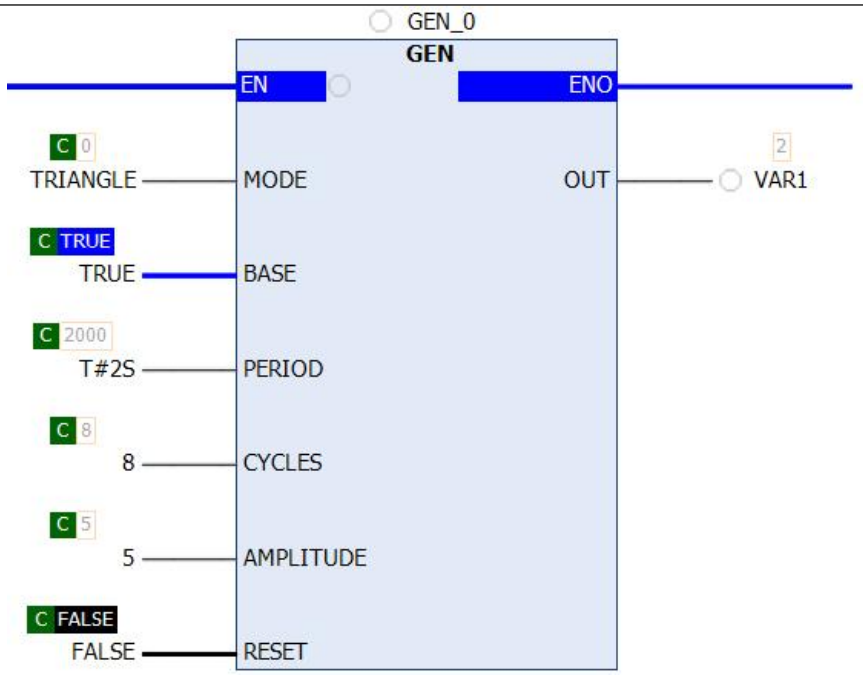
典型周期信号发生器指令用于生成典型的周期信号，包括三角波信号、零起点三角波信号、上升锯齿波信号、　下降锯齿波信号、方波信号、正弦波信号和余弦波信号。

输入参数	数据类型	功能描述	参数值说明

MODE	GEN_MODE	指定要产生的信号类型	引脚输入名称及对应信号类型如下： TRIANGLE，三角波 TRIANGLE_POS，零起点三角波 SAWTOOTH_RISE，上升锯齿波 SAWTOOTH_FALL，下降锯齿波RECTANGLE，方波SINUS，正弦波COSINUS，余弦波
BASE	BOOL	循环方式选择	当BASE 为TRUE 时，信号发生器与定义的循环周期有关。当 BASE 为FALSE 时，信号发生器与特定的发生的个数有关。
PERIOD	TIME	循环周期	
CYCLES	INT	发生的个数	
AMPLITUDE	INT	信号的振幅	
RESET	BOOL	初始化	当 RESET=TRUE 时，信号发生器被重新设置为 0。
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	INT	波形信号输出值	
变量定义			

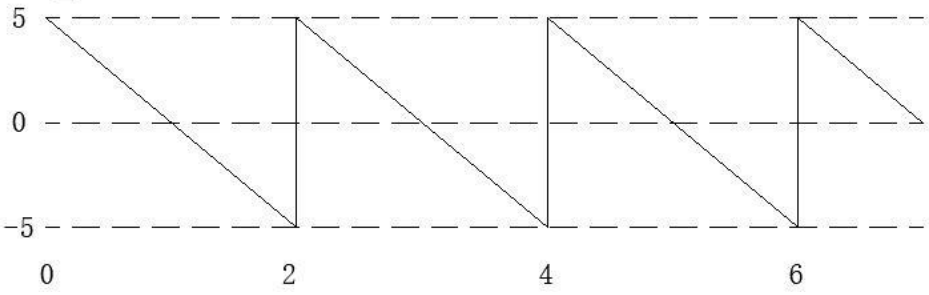
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		INT			
VAR	**GEN_0**		GEN			

编程语言	**程序**

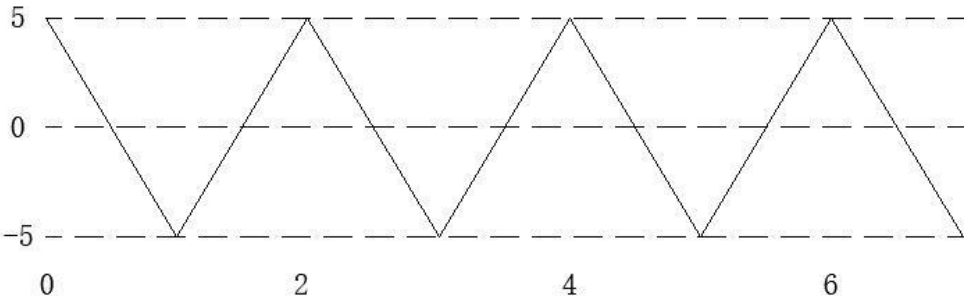
梯形图 （LD2）	
结构化文本 （ST）	GEN_0(MODE:=TRIANGLE, BASE:=TRUE, PERIOD:=T#2s, CYCLES:=8, AMPLITUDE:=5, RESET:=FALSE); Var1:= GEN_0.OUT;

在 MODE 处输入不同的信号类型，得到不同的波形，如下所示。

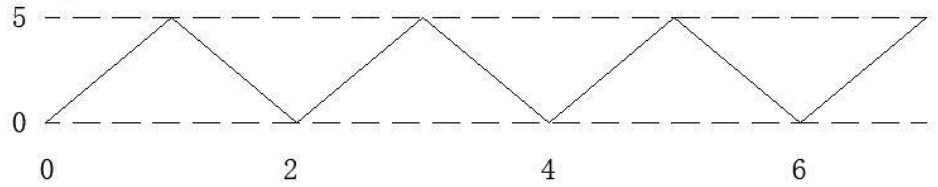
SAWTOOTH_FALL



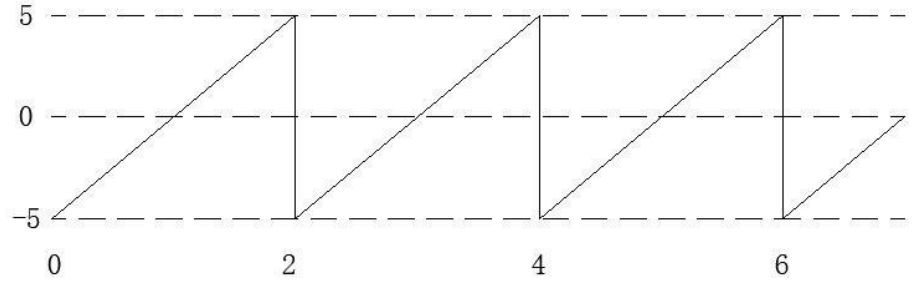
TRIANGLE



TRIANGLE_POS



SAWTOOTH_RISE



3.13.2 BLINK——脉冲信号发生器

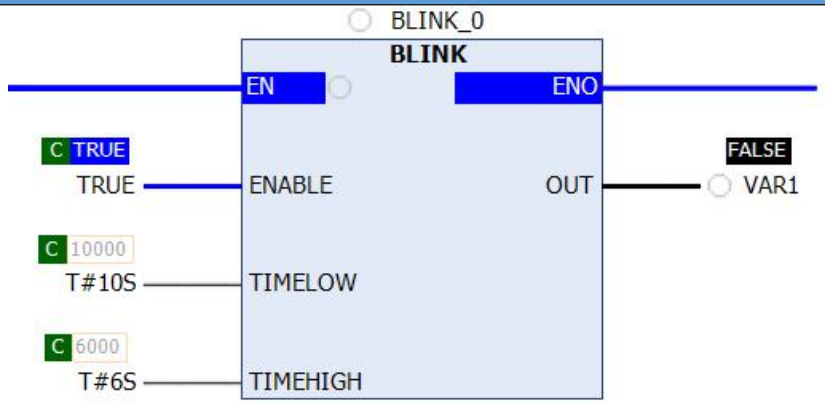
信号发生指令 BLINK 用于产生脉冲信号。

输入参数	数据类型	功能描述	参数值说明
ENABLE	BOOL	使能	TRUE 时开始工作
TIMELOW	TIME	输出低电平时间	
TIMEHIGH	TIME	输出高电平时间	
输出参数	**数据类型**	**功能描述**	**参数值说明**

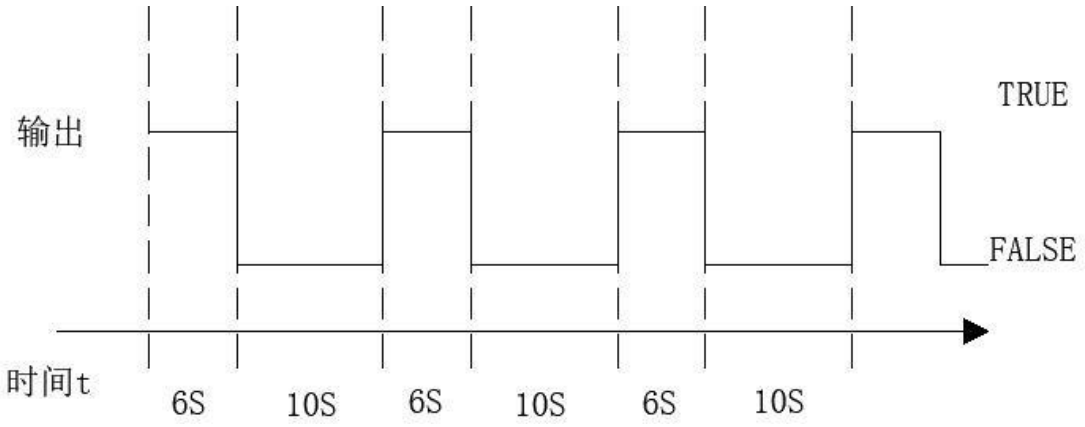
OUT	BOOL	脉冲信号输出值	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		BOOL			
VAR	**BLINK_0**		BLINK			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	BLINK_0 (TIMELOW:=T#10s, TIMEHIGH:=T#6s); VAR1:=BLINK_0.OUT;

指令执行时，输出下图所示的波形。



1、系统设计时已确定，该指令输出的第一个电平一定是高电平。

2、当使能端断开时输出保持不变。

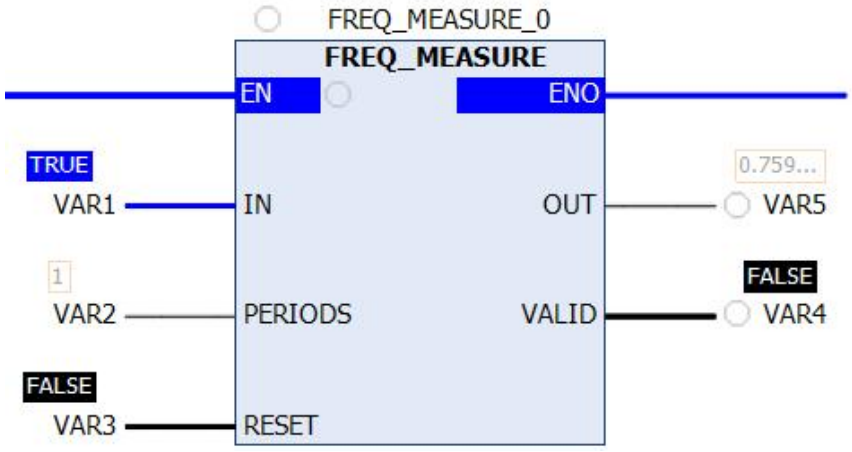
3.13.3 FREQ_MEASURE——频率测量指令

测量输入的布尔类型信号的频率值。

输入参数	数据类型	功能描述	参数值说明
IN	BOOL	输入信号	TRUE 时开始工作
PERIODS	INT	测量周期	范围 1 到10，默认值为 1，输入信号两个上升沿的时间间隔为一个周期，将 N 个周期的频率值求平均值就得到输入信号的频率，此参数即为求平均值运算的周 期次数
RESET	BOOL	复位	TRUE 时重新测量，正常执行时为 FALSE
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	REAL	频率输出	输入信号的频率值，单位 Hz
VALID	BOOL	运算标志	第一次运算完成后或者当运算错误置 TRUE，其余时刻为 FALSE

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**FREQ_MEASURE_0**		FREQ_MEASURE			
VAR	**VAR1**		BOOL			
VAR	**VAR2**		INT (1..10)			
VAR	**VAR3**		BOOL			
VAR	**VAR4**		BOOL			
VAR	**VAR5**		REAL			

编程语言	程序
梯形图（LD2）	
结构化文本	FREQ_MEASURE_0(

（ST）	IN:=VAR1, PERIODS:=VAR2, RESET:=VAR3, OUT=>VAR5 , VALID=>VAR4);

3.14 函数操纵器指令（Util）

3.14.1 RAMP_INT——整型限速

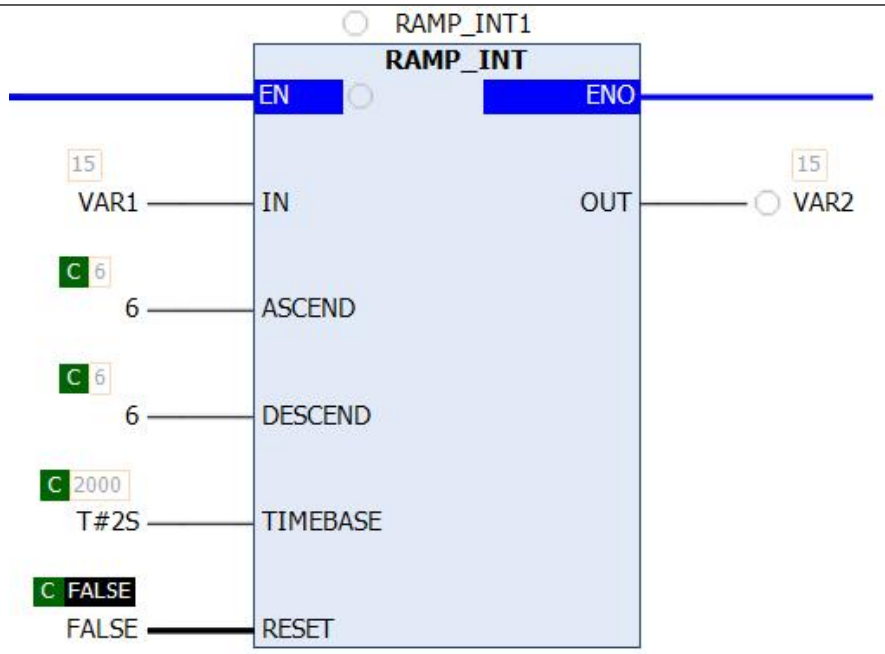
整型限速指令 RAMP_INT 用于对整型输入数据升降速度进行限制，防止输入数据变化过快。

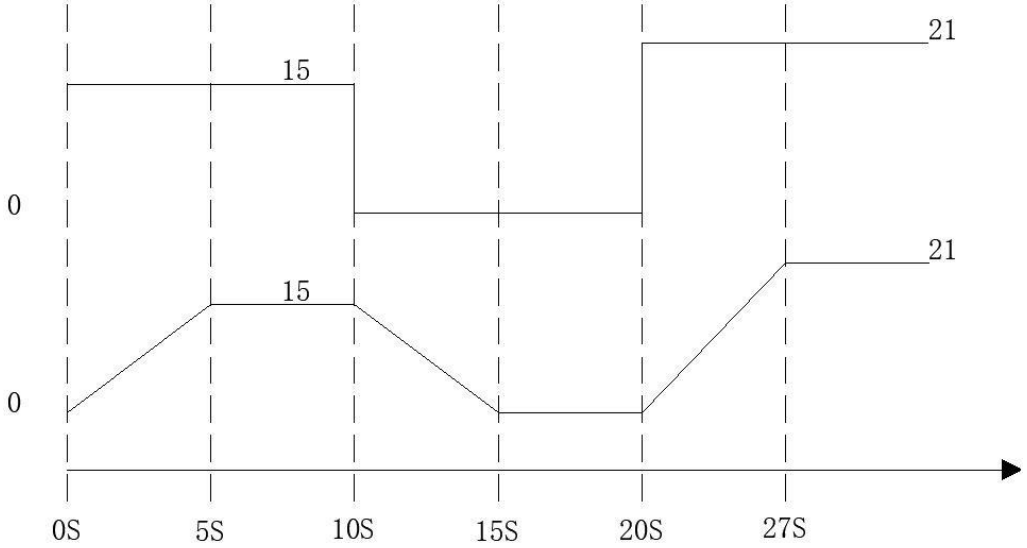
输入参数	数据类型	功能描述	参数值说明
IN	BOOL	输入数据	若输入值大于上一周期的数值，按照时间基数和上升速率进行加法运算。若输入值小于上一周期的数值，按照时间基数和下降速率进行减法运算。
ASCEND	INT	上升速率	在时间基数内上升的数值
DESCEND	INT	下升速率	在时间基数内下降的数值
TIMEBASE	TIME	时间基数	按上升或者下降速率计算变化量的时间基数
RESET	BOOL	初始化	TRUE 时 RAMP_INT 被重新初始化
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	INT	数据输出	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		INT			
VAR	VAR2		INT			
VAR	RAMP_INT1		RAMP_INT			

编程语言	程序

梯形图（LD2）	
结构化文本（ST）	RAMP_INT1(IN:=Var1, ASCEND := 6, DESCEND:=6, TIMEBASE:=T#2s, RESET := FALSE); Var2:=RAMP_INT1.OUT;



指令执行时，输出下图所示的波形。

3.14.2 RAMP_REAL——实型限速

实型限速指令 RAMP_REAL 用于对实型输入数据升降速度进行限制，防止

输入数据变化过快。

输入参数	数据类型	功能描述	参数值说明
IN	REAL	输入数据	若输入值大于上一周期的数值，按照时间基数和上升速 率进行加法运算。若输入值小于上一周期的数值，按照 时间基数和下降速率进行减法运算。
ASCEND	REAL	上升速率	在时间基数内上升的数值
DESCEND	REAL	下升速率	在时间基数内下降的数值
TIMEBASE	TIME	时间基数	按上升或者下降速率计算变化量的时间基数
RESET	BOOL	初始化	TRUE 时 RAMP_REAL 被重新初始化
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	REAL	数据输出	

该指令的应用举例可参见整数限速指令 RAMP_INT。

输入参数	数据类型	功能描述	参数值说明
IN	INT	输入 X 轴的数值	
N	BYTE	定义曲线所使用数组中的点数	（2≤N≤11） N=2 代表使用 P[0]. X、P[0]. Y， P[0]. X、P[0]. Y，2 个点。 N=11 代表使用 P[0]. X、P[0]. Y… P[10].X、P[10]. Y，11 个点。
P	ARRAY[0..10] OF POINT		XY 平面上特征曲线的特征点
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	INT	输出值	输入 X 值对应的输出 Y 值

ERR	BYTE	显示错误类型	ERR=1：数组中的点 P[0]..P[N-1]中的 X 值有错误。 ERR=2：输入值 IN 不在 P[0].X 和 P[N-1].X之间。 若输入值小于 P[0].X，此时 OUT 的输出为P[0].Y。 若输入值大于 P[N-1].X，此时OUT 的输出为P[N-1].Y。 ERR=4：输入 N 小于 2，或者大于 11。

3.14.3 CHARCURVE——特征曲线

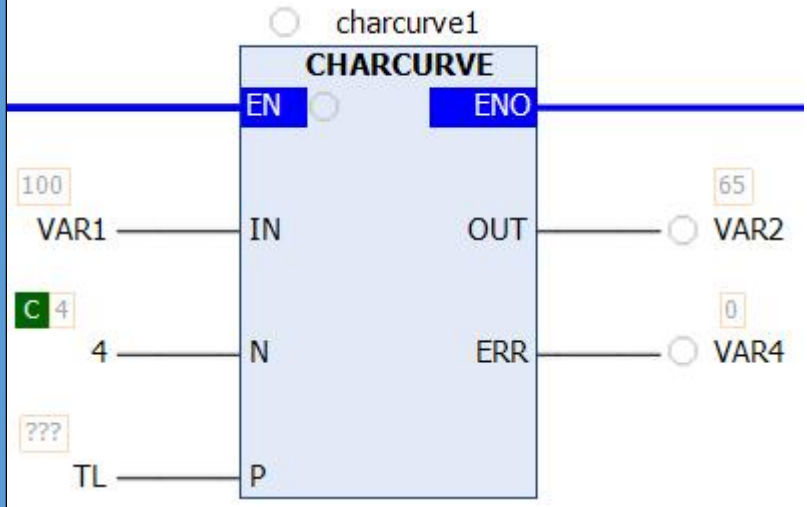
特征曲线指令 CHARCURVE 用于描述特征曲线上任意一点的位置。根据给定的二位数组，在 XY 平面上可绘制出一条特征曲线，该二位数组共有 N 个数据，分别为(x0，y0)、(x1，y1)、(x2，y2)…(XN-1，y N-1)。在此基础上，任意输入一个数值在 X0…XN-1 之间，可求取相应的 y 的数值。

应用举例

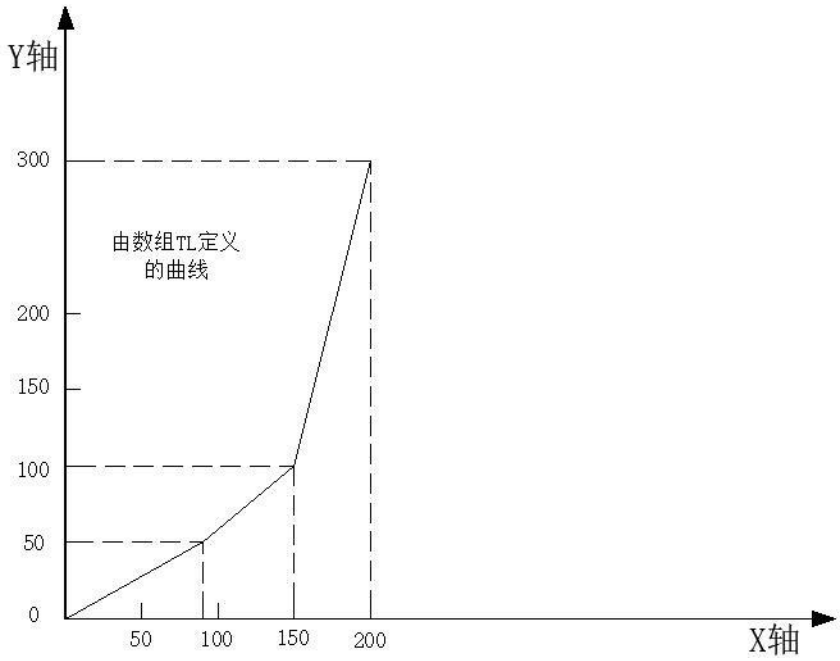
变量定义

类别	名称	地址	数据类型	初值
VAR	VAR1		INT	
VAR	VAR2		INT	
VAR	RAMP_INT1		RAMP_INT	
VAR	charcurve1		CHARCURVE	
VAR	VAR4		BYTE	
VAR	TL		ARRAY [0..10] OF POINT	[(x:=0,y:=0),(x:=80,y:=50),(x:=150,y:=100),(x:=200,y:=300)]

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	charcurve1 (IN:=VAR1, N := 4, P := TL); ERR:= VAR4; OUT:= VAR2;

上述程序运行时，可得到下图所示的特征曲线，输入 IN 值为 100，输出 OUT 的值为 65。



3.15 高等数学运算指令（Util）

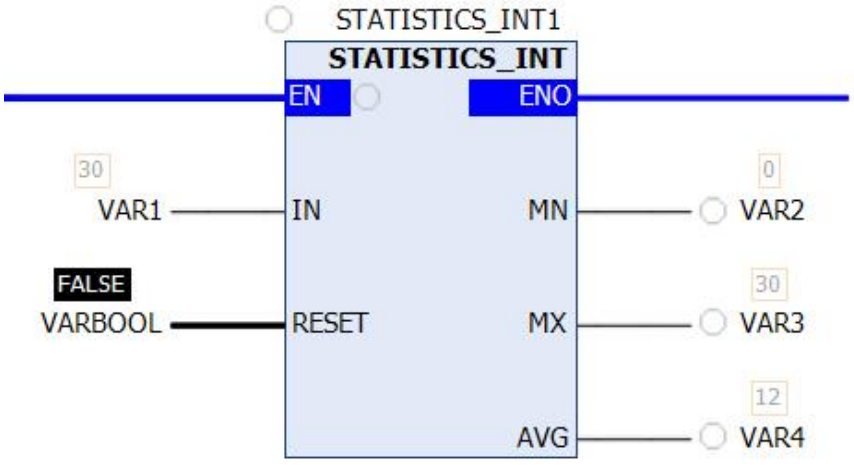
3.15.1 STATISTICS_INT——整型统计指令

整型统计指令 STATISTIC_INT 用于统计输入整型数据的最大值、最小值和平均值。

输入参数	数据类型	功能描述	参数值说明
IN	INT	输入	
RESET	BOOL	初始化	为 TRUE 时，重新初始化
输出参数	**数据类型**	**功能描述**	**参数值说明**
MN	INT	最小值	
MX	INT	最大值	
AVG	INT	平均值	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**STATISTICS_INT1**		STATISTICS_INT			
VAR	**VAR1**		INT			
VAR	**VAR2**		INT			
VAR	**VAR3**		INT			
VAR	**VAR4**		INT			
VAR	**VARBOOL**		BOOL			

编程语言	程序
梯形图（LD2）	 整型统计指令是每改变一次 VAR1 的值，计算一次
结构化文本（ST）	STATISTICS_INT1(IN := VAR1, RESET := VARBOOL);

	VAR2:=STATISTICS_INT 1.MN; VAR3:=STATISTICS_INT 1.MX; VAR4:=STATISTICS_INT1.AVG;

当布尔型输入 RESET 为 TRUE 时，所有的变量值都将会被初始化。

3.15.2 STATISTICS_REAL——实型统计指令

实型统计指令 STATISTIC_REAL 用于统计输入实型数据的最大值、最小值和平均值。

输入参数	数据类型	功能描述	参数值说明
IN	REAL	输入	
RESET	BOOL	初始化	为 TRUE 时，重新初始化
输出参数	**数据类型**	**功能描述**	**参数值说明**
MN	REAL	最小值	
MX	REAL	最大值	
AVG	REAL	平均值	

该指令的应用举例可参见整型统计指令 STATISTICS_INT。

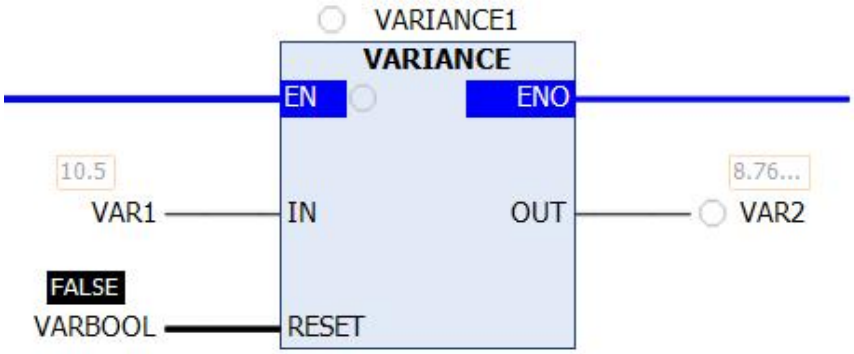
3.15.3 VARIANCE——平方偏差指令

平方偏差指令用于求解输入数据的反正弦值，输出数据为运算结果。

输入参数	数据类型	功能描述	参数值说明
IN	REAL	输入	
RESET	BOOL	复位	为 TRUE 时，指令复位
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	REAL	平方偏差	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		REAL			
VAR	**VAR2**		REAL			
VAR	**VARBOOL**		BOOL			
VAR	**VARIANCE1**		VARIANCE			

编程语言	程序
梯形图（LD2）	 输出结果：8.76...
结构化文本（ST）	VARIANCE1(IN:=VAR1, RESET:=VARBOOL); VAR2:=VARIANCE. OUT; 输出结果：8.76...

得到一组数的平方偏差后，可对平方偏差值求平方根，得到该组数的标准偏差。

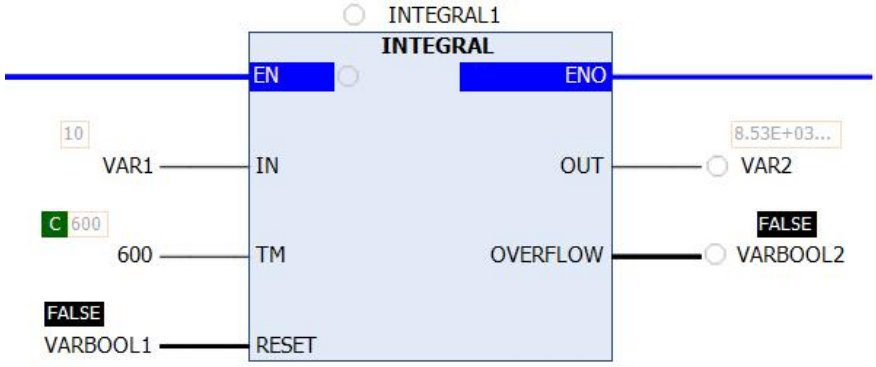
3.15.4 INTEGRAL——积分指令

积分指令用于求解输入数据的积分值，输出数据为运算结果。

输入参数	数据类型	功能描述	参数值说明
IN	REAL	连续输入的变量	
TM	DWORD	积分时间	单位秒
RESET	BOOL	复位信号	为 TRUE 时，重新启动指令
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	REAL	积分运算结果	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		INT			
VAR	**VAR2**		REAL			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**INTEGRAL1**		INTEGRAL			

编程语言	程序
梯形图（LD2）	 输出结果：8.53E+03
结构化文本（ST）	INTEGRAL1(IN := VAR1, TM := 600, RESET:=VARBOOL1); VAR2:=INTEGRAL.OUT; VARBOOL2:=INTEGRAL.OVERFLOW; 输出结果：8.53E+03

在 PLC 编程应用中，所有连续变化的输入量均需离散化后再计算、存储，故积分运算也是采用离散方式进行的，当积分周期足够小时可认为该计算接近实际连续过程。

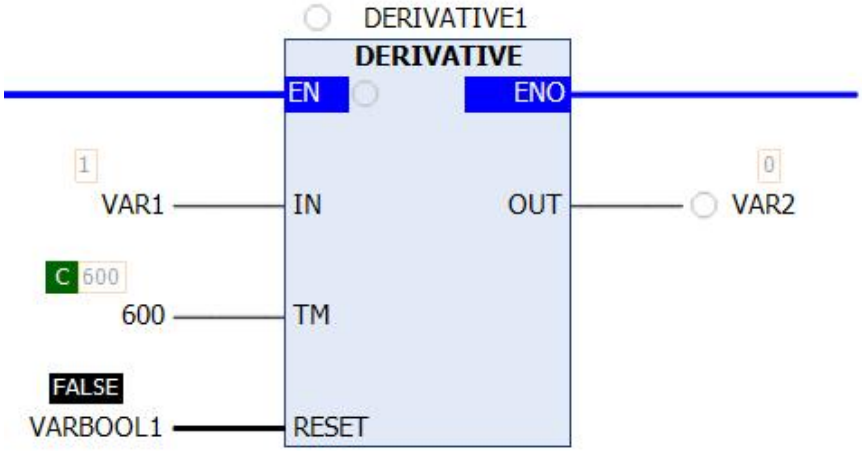
3.15.5 DERIVATIVE——微分指令

微分指令用于求解输入数据的微分值，输出数据为运算结果。

输入参数	数据类型	功能描述	参数值说明
IN	REAL	连续输入的变量	
TM	DWORD	微分时间	单位秒
RESET	BOOL	复位信号	为 TRUE 时，重新启动指令
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	REAL	微分运算结果	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		INT			
VAR	**VAR2**		REAL			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**DERIVATIVE1**		DERIVATIVE			

编程语言	程序
梯形图（LD2）	 输出结果：0
结构化文本（ST）	DERIVATIVE1(IN:=VAR1, TM:=600, RESET:=VARBOOL1); VAR2:=DERIVATIVE1. OUT; 输出结果：0

微分指令对当前时刻的前 3 个运算周期的数据进行运算，以提高数据处理的准确性和平滑性，3 个周期所占权重分别为 30%、40%和 30%。

3.15.6 LIN_TRAFO——线性变换指令

通过使用输入的最小值和最大值来实现输出值的线性近似值。

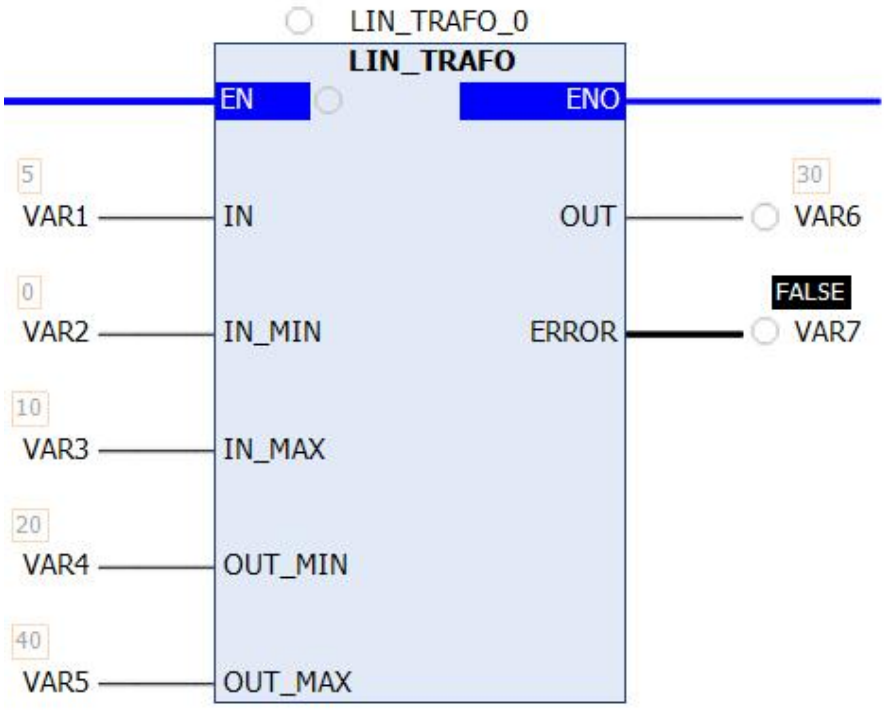
输入参数	数据类型	功能描述	参数值说明
IN	REAL	输入值	
IN_MIN	REAL	最小输入值	

IN_MAX	REAL	最大输入值	
OUT_MIN	REAL	最小输出值	
OUT_MAX	REAL	最大输出值	
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	REAL	输出值	
ERROR	BOOL	错误代码	0：无错误 1：IN_MIN = IN_MAX 或 IN 超出输入范围

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		REAL			
VAR	**VAR2**		REAL			
VAR	**VAR3**		REAL			
VAR	**VAR4**		REAL			
VAR	**VAR5**		REAL			
VAR	**VAR6**		REAL			
VAR	**VAR7**		BOOL			
VAR	**LIN_TRAFO_0**		LIN_TRAFO			

编程语言	程序

梯形图 （LD2）	 输出结果：30
结构化文本 （ST）	LIN_TRAFO_0(IN:=VAR1, IN_MIN:=VAR2, IN_MAX:=VAR3, OUT_MIN:=VAR4, OUT_MAX:=VAR5, OUT=>VAR6, ERROR=>VAR7); 输出结果：30

3.16 控制器指令（Util）

在对连续变化的过程参数进行调节时，目前应用最多的控制策略仍是经典的 PID 调节，即比例、积分、微分调节。

3.16.1 PD——比例微分控制器

比例微分控制器 PD 指令实现比例微分调节功能，该指令由测量值和设定值得到二者的差值，再根据差值求比例部分和微分部分。

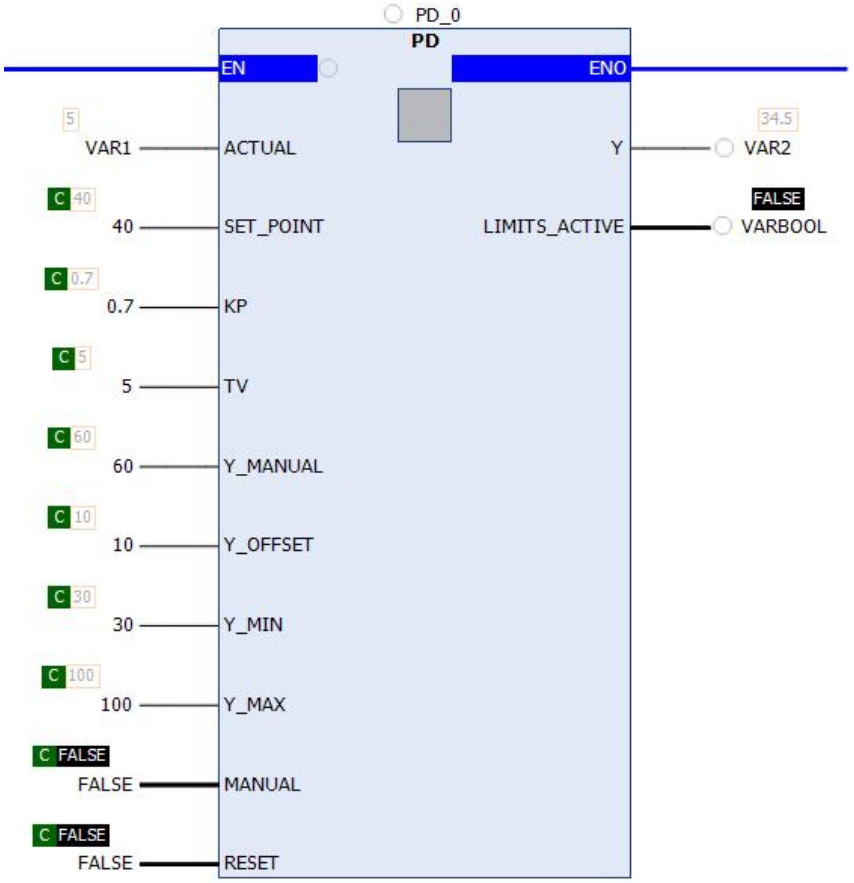
输入参数	数据类型	功能描述	参数值说明

ACTUAL	REAL	测量值	
SET_POINT	REAL	设定值	
KP	REAL	比例系数	
TV	REAL	微分时间	单位秒
Y_MANUAL	REAL	手动值	MANUAL=TRUE 时，Y=Y_MANUAL
Y_OFFSET	REAL	输出偏移量	
Y_MIN	REAL	输出最小值	
Y_MAX	REAL	输出最大值	
MANUAL	BOOL	手自动选择	TRUE 为手动调节，FALSE 为自动调节
RESET	BOOL	重置	TRUE 时重置控制器，正常时应置 FALSE
输出参数	**数据类型**	**功能描述**	**参数值说明**
Y	REAL	输出值	
LIMITS_ACTIVE	BOOL	输出超限标志	超限时为 TRUE，不超限时为 FALSE

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		REAL			
VAR	**VAR2**		REAL			
VAR	**VARBOOL**		BOOL			
VAR	**PD_0**		PD			

编程语言	**程序**

梯形图 （LD2）	
结构化文本 （ST）	PD_0(ACTUAL:=VAR1, SET_POINT:=40, KP:=0.7, TV:=5, Y_MANUAL:=60, Y_OFFSET:=10, Y_MIN:=30, Y_MAX:=100, MANUAL:=FALSE, RESET:=FALSE, Y=>VAR2, LIMITS_ACTIVE=>VARBOOL);

3.16.2 PID——比例积分微分控制器

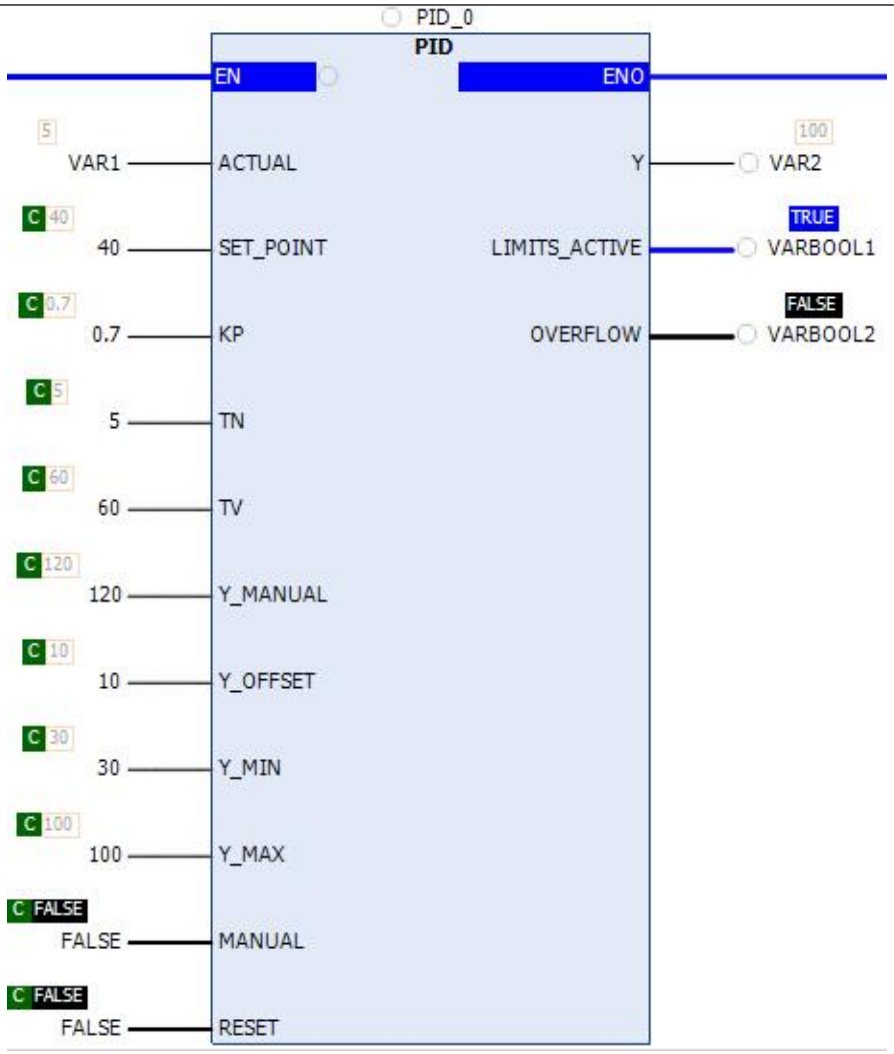
比例积分微分控制器 PID 指令实现比例积分微分调节功能，该指令由测量值和设定值得到二者的差值，再根据差值求比例部分、积分部分和微分部分。比例积分微分控制器中 TV=0 时，PID 控制器就变成了 PI 控制器。

输入参数	数据类型	功能描述	参数值说明
ACTUAL	REAL	测量值	
SET_POINT	REAL	设定值	
KP	REAL	比例系数	
TN	REAL	积分时间	单位为秒
TV	REAL	微分时间	单位为秒
Y_MANUAL	REAL	手动值	MANUAL=TRUE 时， Y= Y_MANUAL
Y_OFFSET	REAL	输出偏移量	
Y_MIN	REAL	输出最小值	
Y_MAX	REAL	输出最大值	
MANUAL	BOOL	手自动选择	TRUE 为手动调节，FALSE 为自动调节
RESET	BOOL	重置	TRUE 重置该控制器，正常时置 FALSE
输出参数	**数据类型**	**功能描述**	**参数值说明**
Y	REAL	输出值	
LIMITS_ACTIVE	BOOL	输出超限标志	输出值超限时为 TRUE
OVERFLOW	BOOL	输出溢出标志	积分溢出时为 TRUE

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		REAL			
VAR	**VAR2**		REAL			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**PID_0**		PID			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	PID_0(ACTUAL:=VAR1, SET_POINT:=40, KP:=0.7, TN:=5, TV:=60, Y_MANUAL:=120, Y_OFFSET:=10, Y_MIN:=30, Y_MAX:=100, MANUAL:= FALSE, RESET:= FALSE, Y=>VAR2, LIMITS_ACTIVE=>VARBOOL1,

	OVERFLOW=>VARBOOL2);

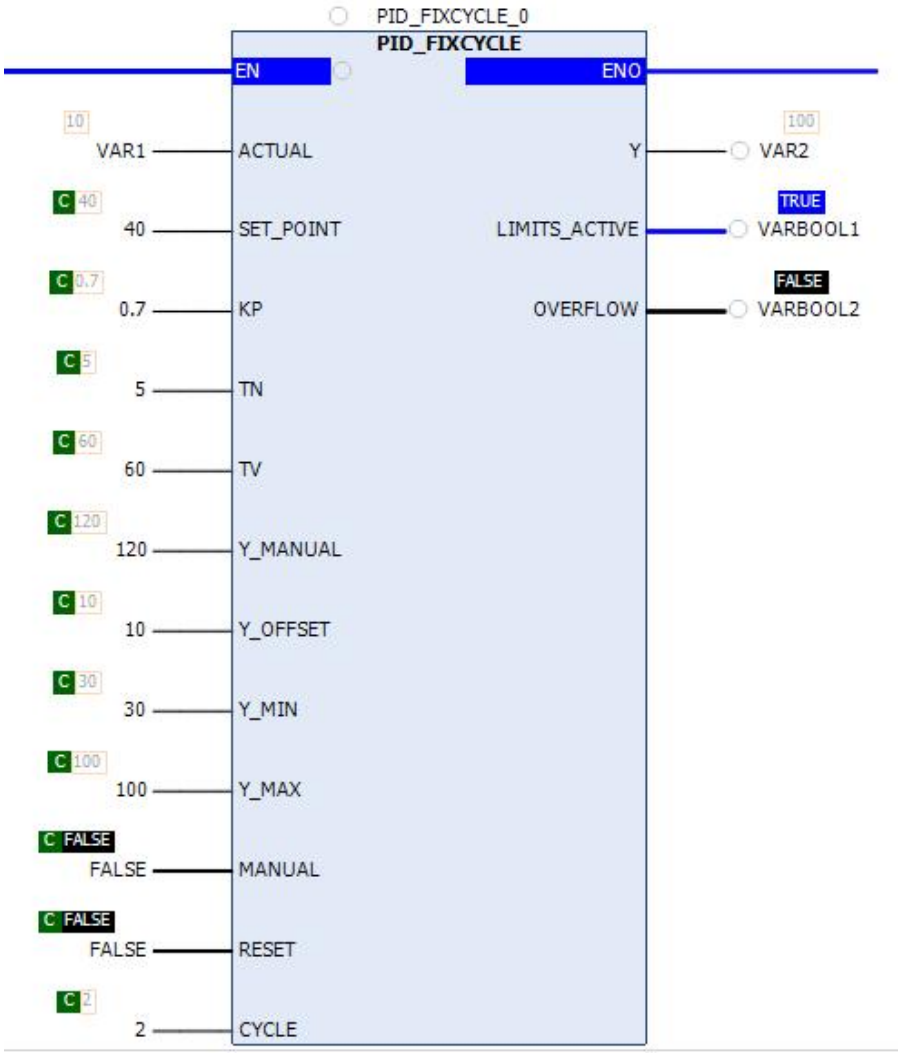
3.16.3 PID_FIXCYCLE——比例积分微分控制器

该指令与前述的比例积分微分控制器功能基本相同，唯一的区别是，PID 调节周期是通过参数 CYCLE 传输进来的，该数据用来设定积分和微分的时间长短。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		REAL			
VAR	**VAR2**		REAL			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**PID_FIXCYCLE_0**		PID_FIXCYCLE			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	PID_FIXCYCLE_0(ACTUAL:=VAR1, SET_POINT:=40, KP:= 0.7, TN:=5, TV:=60, Y_MANUAL:=120, Y_OFFSET:=10, Y_MIN:=30, Y_MAX:=100, MANUAL:=FALSE, RESET:=FALSE, CYCLE:=2,

	Y=>VAR2, LIMITS_ACTIVE=>VARBOOL1, OVERFLOW=>VARBOOL2);

3.17 数据异常处理指令（Util）

3.17.1 LIMITALARM——上下限报警

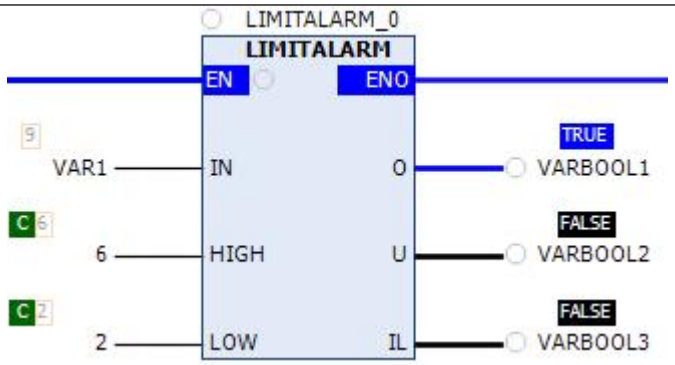
上下限报警指令 LIMITALARM 用于对输入数据的上下限值进行监视，若输入值超过上限值则发出上限报警信号，若输入值超过下限值则发出下限报警信号。

输入参数	数据类型	功能描述	参数值说明
IN	INT	输入值	
HIGH	INT	上限值	
LOW	INT	下限值	
输出参数	**数据类型**	**功能描述**	**参数值说明**
O	BOOL	输出值	输入超过上限时为 TRUE
U	BOOL	输出值	输入低于下限时为 TRUE
IL	BOOL	输出值	输入介于上下限之间时为 TRUE

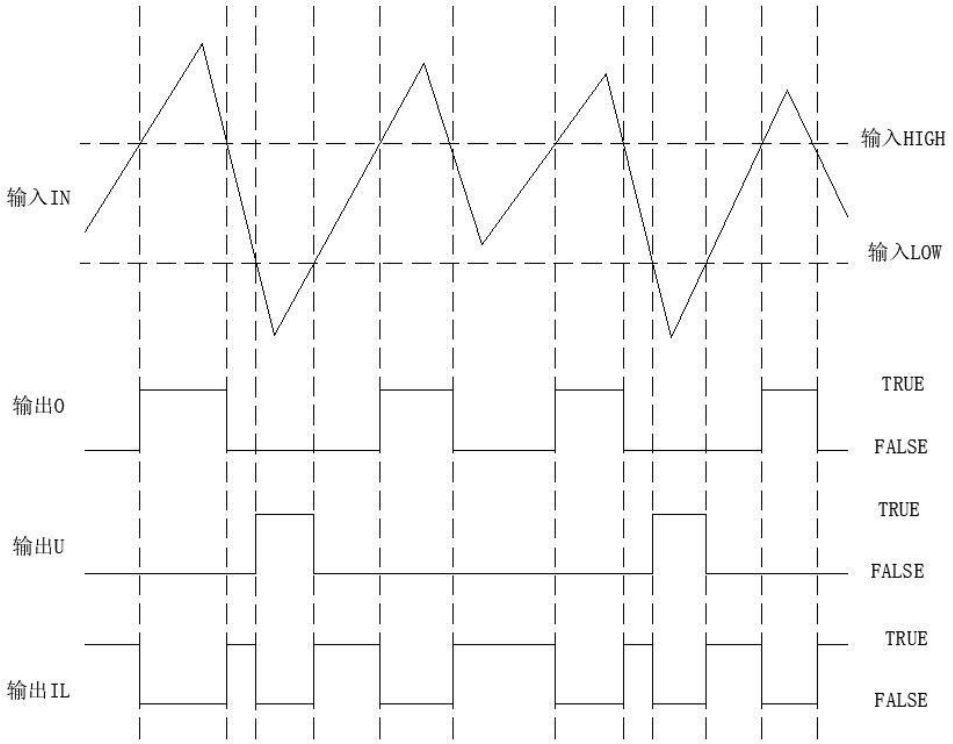
变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		INT			
VAR	VARBOOL1		BOOL			
VAR	VARBOOL2		BOOL			
VAR	VARBOOL3		BOOL			
VAR	LIMITALARM_0		LIMITALARM			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	LIMITALARM_0(IN:=VAR1, HIGH:=6, LOW:=2, O=>VARBOOL1, U=>VARBOOL2, IL=>VARBOOL3);

上述程序运行时，输入、输出对应关系如下图所示。



3.17.2 HYSTERESIS——滞后

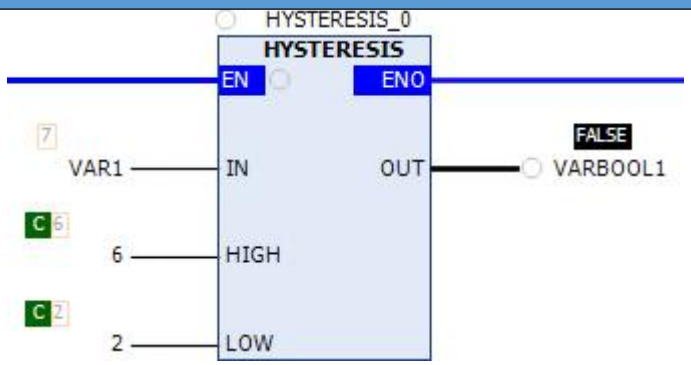
滞后指令 HYSTERESIS 相当于在变量变化过程中设置了一个死区，当输入信号由低向高变化和由高向低变化时，输出状态发生变化的阈值不同。应

用该指令后，一方面可以减少各类工业现场执行机构的动作频率，另一方面还能在一定程度上克服现场干扰信号的影响。

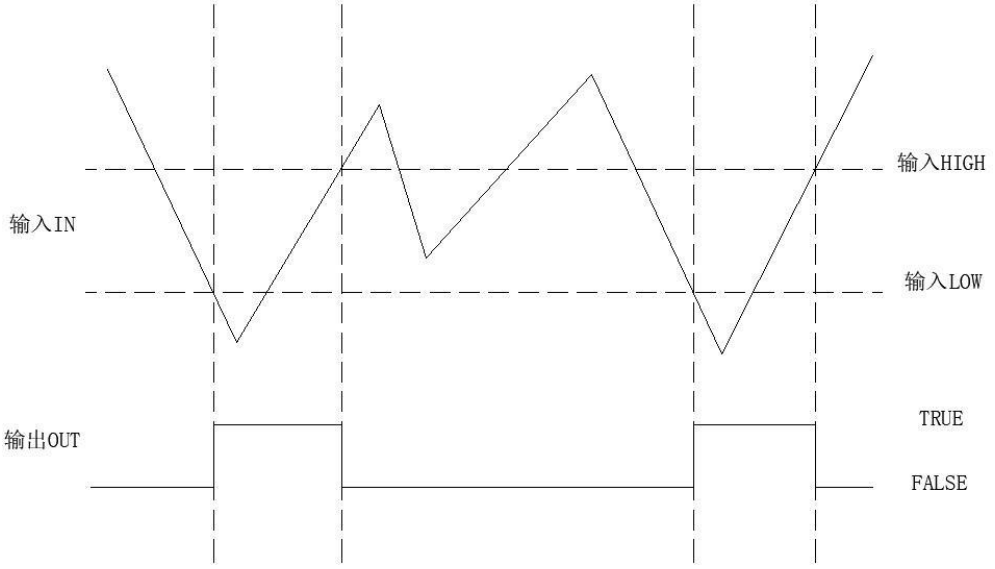
输入参数	数据类型	功能描述	参数值说明
IN	INT	输入值	
HIGH	INT	上限值	
LOW	INT	下限值	
输出参数	**数据类型**	**功能描述**	**参数值说明**
OUT	BOOL	输出值	输入低于下限值后为 TRUE，输入高于上限值后为 FALSE

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		INT			
VAR	VARBOOL1		BOOL			
VAR	HYSTERESIS_0		HYSTERESIS			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	HYSTERESIS_0(IN :=VAR1, HIGH:=6, LOW:=2, OUT=>VARBOOL1);

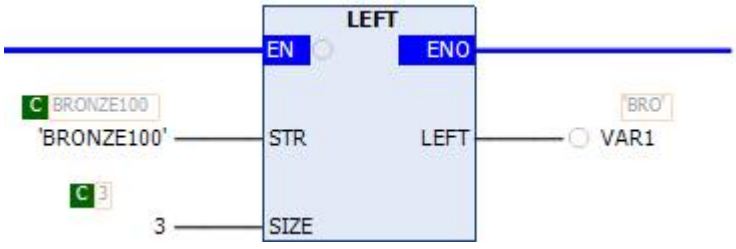
上述程序运行时，得到如下图所示结果。



3.18 字符串处理指令（Util.Standard）

3.18.1 LEFT——左边取字符串指令

左边取字符串指令 LEFT 用于从字符串左边取一部分或全部字符串，该指令的语法格式为 LEFT(STR,SIZE)。

指令对应的输入和输出数据类型	
输入 STR 的数据类型为 STRING，输入 SIZE 的数据类型为 BYTE、INT、SINT、UINT、USINT、WORD。 输出的数据类型为 STRING。	
变量定义	
类别 / 名称 / 地址 / 数据类型 / 初值 / 注释 / 属性 VAR / VAR1 / / STRING / / /	
编程语言	**程序**
梯形图 （LD2）	 输出结果：'BRO'

结构化文本（ST）	VAR1:=LEFT('BRONZE100',3); 输出结果：'BRO'

3.18.2 MID——中间取字符串指令

中间取字符串指令 MID 用于从字符串中间某一位置开始由左向右取一部分或全部字符串，该指令的语法格式为 MID(STR,LEN,POS)。

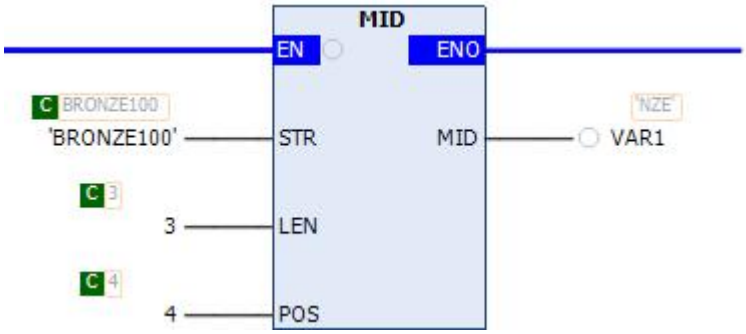
指令对应的输入和输出数据类型

输入 STR 的数据类型为 STRING，输入 LEN 和POS 的数据类型为 BYTE、INT、SINT、UINT、USINT、WORD。

输出的数据类型为 STRING。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		STRING			

编程语言	程序
梯形图（LD2）	 输出结果：'NZE'
结构化文本（ST）	VAR1:=MID('BRONZE100',3,4); 输出结果：'NZE'

3.18.3 RIGHT——右边取字符串指令

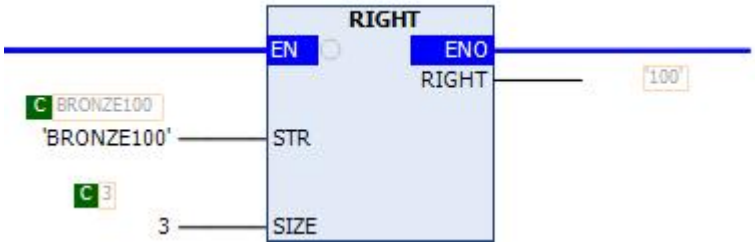
右边取字符串指令RIGHT用于从字符串右边取一部分或全部字符串，该指令的语法格式为RIGHT(STR,SIZE)。

指令对应的输入和输出数据类型

输入 STR 的数据类型为 STRING，输入 SIZE 的数据类型为 BYTE、INT、SINT、UINT、USINT、WORD。

输出的数据类型为 STRING。

变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		STRING			

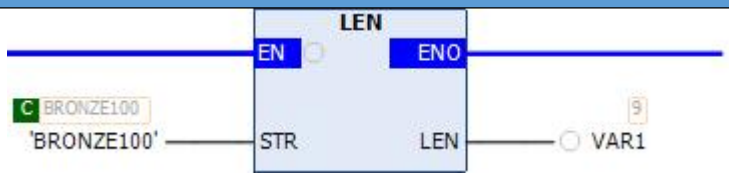
编程语言	程序
梯形图（LD2）	 输出结果：'100'
结构化文本（ST）	VAR1:=RIGHT('BRONZE100',3); 输出结果：'100'

3.18.4 LEN——取字符串长度指令

取字符串长度指令 LEN 用于计算输入字符串的长度。

指令对应的输入和输出数据类型
输入的数据类型为 STRING，输出的数据类型为 DINT、DWORD、INT、REAL、UDINT、UINT、USINT、WORD。

变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		INT			

编程语言	程序
梯形图（LD2）	 输出结果：9
结构化文本（ST）	VAR1:=LEN('BRONZE100'); 输出结果：9

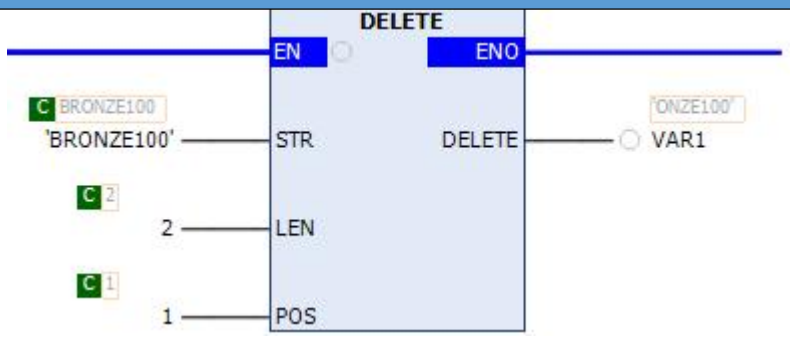
3.18.5 DELETE——删除字符指令

删除字符串指令 DELETE 用于从字符串的某一位置开始删除部分字符串或

全部字符串，该指令格式为：DELETE(STR,LEN,POS)，其功能为从输入字符串从前往后数的位置 POS 处开始，连续删除 LEN 个字符。

指令对应的输入和输出数据类型
输入 STR 的数据类型为 STRING，输入 LEN 和POS 的数据类型为 BYTE、INT、SINT、UINT、USINT、WORD。 输出的数据类型为 STRING。

变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		STRING			

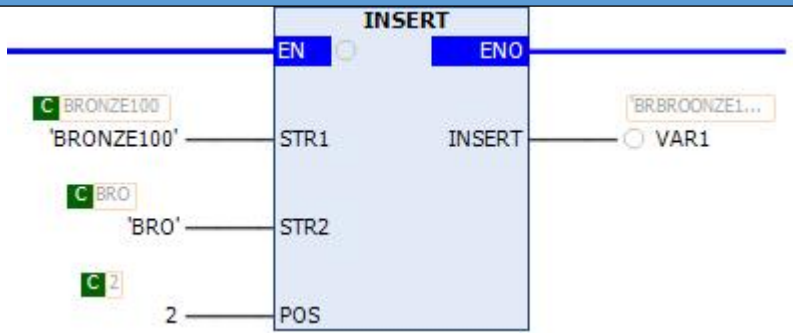
编程语言	程序
梯形图 （LD2）	 输出结果：'ONZE100'
结构化文本 （ST）	VAR1:= DELETE('BRONZE100',1,2); 输出结果：''ONZE100'

3.18.6 INSERT——插入字符串指令

插入字符串指令用于把一个字符串插入到另一个字符串内，该指令的语法格式为：INSERT(STR1,STR2,POS)，其中 STR1 为需插入字符串的原始字符串，STR2 为插入的字符串，POS+1 为原始字符串从左向右开始计算的插入起始位置，即把 STR2 插入到 STR1 的POS 位置之后。

指令对应的输入和输出数据类型
输入 STR1 和STR2 的数据类型为 STRING，输入 POS 的数据类型为 BYTE、INT、SINT、UINT、USINT、WORD。 输出的数据类型为 STRING。
变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		STRING			

编程语言	程序
梯形图（LD2）	 输出结果：'BRBROONZE100'
结构化文本（ST）	VAR1:=INSERT(' BRONZE100', ' BRO',2); 输出结果：'BRBROONZE100'

3.18.7 REPLACE——替换字符串指令

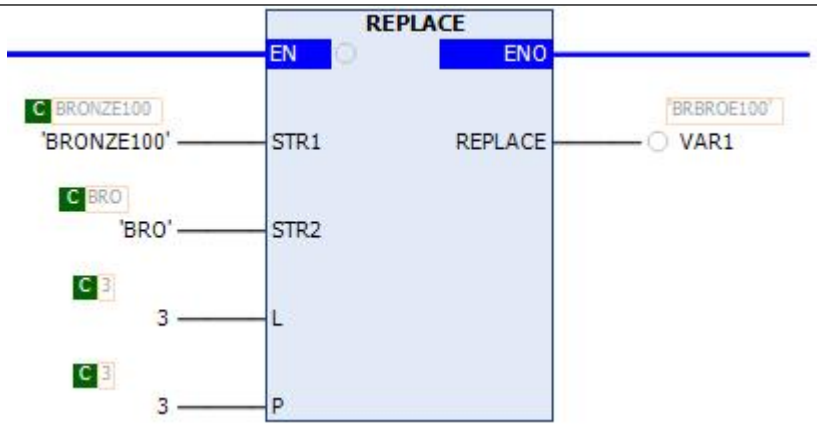
替换字符串指令 REPLACE 用于将一个字符串代替另一个字符串中的部分内容，该指令的语法格式为：

REPLACE(STR1,STR2,L,P)，其功能为用字符串 STR2 代替字符串 STR1 中从位置 P 起始的 L 个字符。

指令对应的输入和输出数据类型
输入 STR1 和STR2 的数据类型为 STRING，输入 L 和P 的数据类型为 BYTE、INT、SINT、UINT、USINT、WORD。 输出的数据类型为 STRING。

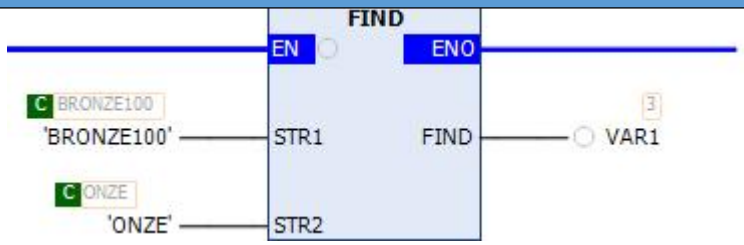
变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		STRING			

编程语言	程序

梯形图 （LD2）	 输出结果：'BRBROE100'
结构化文本 （ST）	VAR1:=REPLACE('BRONZE100','BRO',3,3); 输出结果：'BRBROE100'

3.18.8 FIND——查找字符串指令

查找字符串指令 FIND 用于在一个字符串中查找既定内容，该内容由另一字符串确定，该指令的语法格式为： FIND(STR1,STR2)，输出结果为既定查找内容在第一个字符串中的起始位置。若第一个字符串中没有与第二个字符串完全相同的内容，输出结果为 0。

指令对应的输入和输出数据类型						
输入 STR1 和 STR2 的数据类型为 STRING。 输出的数据类型为 BYTE、INT、REAL、SINT、UINT、USINT、WORD。						
变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		INT			
编程语言	**程序**					
梯形图 （LD2）	 输出结果：3					
结构化文本	VAR1:=FIND ('BRONZE100', 'ONZE'); 输出结果：3					

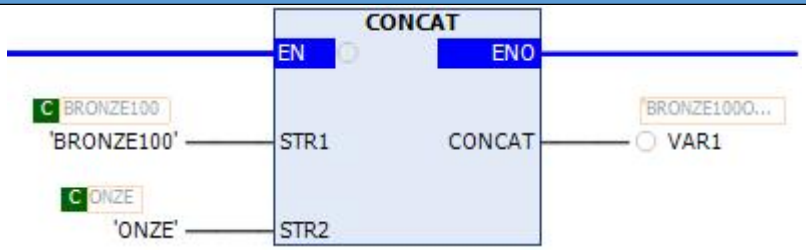
（ST）	

3.18.9 CONCAT——合并字符串指令

合并字符串指令 CONCAT 用于把两个字符串合并成一个字符串。

指令对应的输入和输出数据类型
输入和输出的数据类型均为 STRING。

变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		STRING			

编程语言	程序
梯形图（LD2）	 输出结果：'BRONZE100ONZE'
结构化文本（ST）	VAR1:=CONCAT ('BRONZE100', 'ONZE'); 输出结果：'BRONZE100ONZE'

3.19 计数器（Util.Standard）

系统支持三类计数器，分别为递减计数器 CTD、递增计数器 CTU 和递增递减计数器 CTUD。

3.19.1 CTD——递减计数器

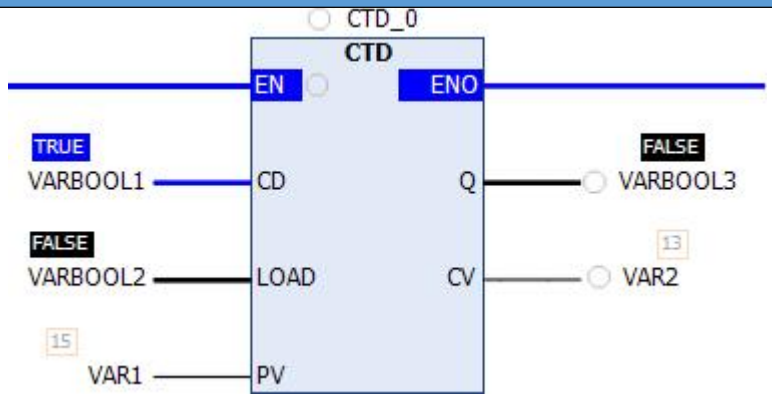
递减计数器指令 CTD 各参数的含义及相关说明见下表。

输入参数	数据类型	功能描述	参数值说明
CD	BOOL	计数输入	CV 大于 0 时，CD 每检测到 1 个上升沿 CV 减 1
LOAD	BOOL	初始化	为 TRUE 时，CV 被初始化为 PV
PV	WORD	计数器设定值	0-65535

输出参数	数据类型	功能描述	参数值说明
Q	BOOL	计数标志输出	当 CV 等于 0 时，Q 为 TRUE
CV	WORD	当前计数值	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VAR1		WORD			
VAR	VAR2		WORD			
VAR	VARBOOL1		BOOL			
VAR	VARBOOL2		BOOL			
VAR	VARBOOL3		BOOL			
VAR	CTD_0		CTD			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	CTD_0(CD:=VARBOOL1, LOAD:=VARBOOL2, PV:=VAR1, Q=>VARBOOL3, CV=>VAR2);

3.19.2 CTU——递增计数器

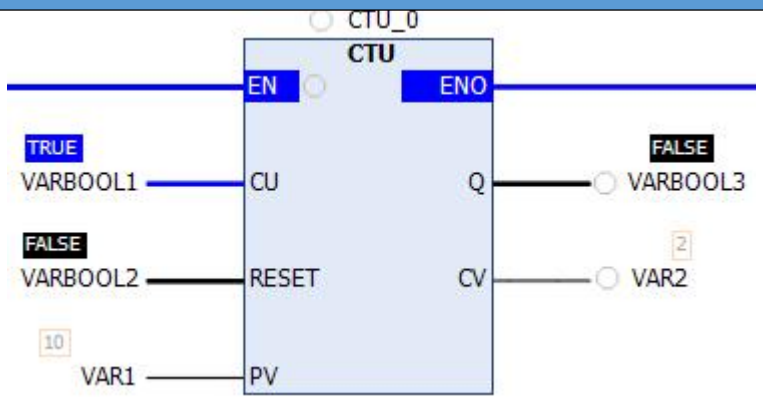
递增计数器指令 CTU 各参数的含义及相关说明见下表。

输入参数	数据类型	功能描述	参数值说明
CU	BOOL	计数输入	CU 每检测到 1 个上升沿，CV 加 1

RESET	BOOL	初始化	为 TRUE 时 CTU 被重新初始化
PV	WORD	计数器设定值	0-65535
输出参数	**数据类型**	**功能描述**	**参数值说明**
Q	BOOL	计数标志输出	CV 大于或等于 PV 时，Q 为 TRUE
CV	WORD	当前计数值	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		WORD			
VAR	**VAR2**		WORD			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			
VAR	**CTU_0**		CTU			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	CTU_0(CU:=VARBOOL1, LOAD:=VARBOO L2,PV:=VAR1, Q=>VARBOOL3, CV=>VAR2);

3.19.3 CTUD——递增递减计数器

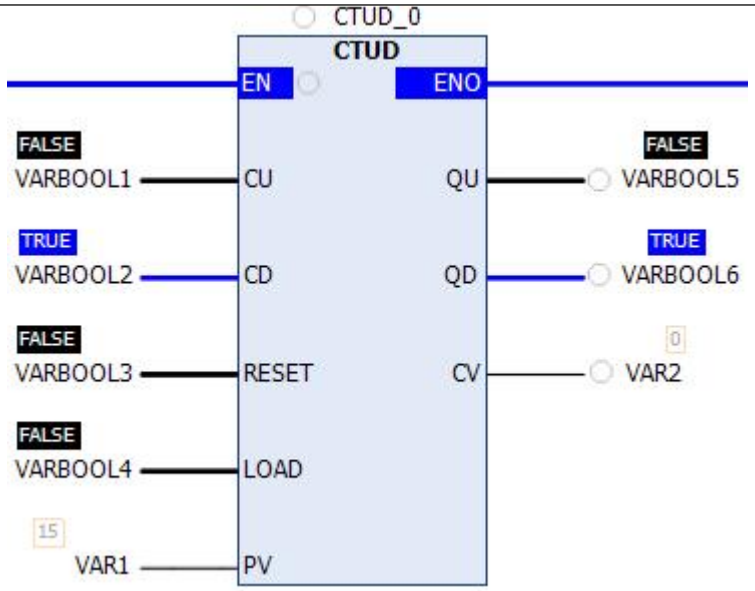
递增递减计数器指令 CTUD 各参数的含义及相关说明见下表。

输入参数	数据类型	功能描述	参数值说明
CU	BOOL	计数输入	CU 每检测到 1 个上升沿，CV 加 1
CD	BOOL	计数输入	CV 大于0 时CD 每检测到1 个上升沿，CV 减 1
RESET	BOOL	复位输入	为 TRUE 时，CV 初始化为 0
LOAD	BOOL	初始化	为 TRUE 时，CV 被初始化为 PV
PV	WORD	计数器设定值	0-65535
输出参数	**数据类型**	**功能描述**	**参数值说明**
QU	BOOL	计数标志输出	当 CV 等于 PV 时，QU 为 TRUE
QD	BOOL	计数标志输出	当 CV 等于 0 时，QD 为 TRUE
CV	WORD	当前计数值	

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		WORD			
VAR	**VAR2**		WORD			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			
VAR	**VARBOOL4**		BOOL			
VAR	**VARBOOL5**		BOOL			
VAR	**VARBOOL6**		BOOL			
VAR	**CTUD_0**		CTUD			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	CTUD_0(CU:=VARBOOL1, CD:=VARBOOL2, RESET:=VARBOOL3, LOAD:=VARBOOL4, PV:=VAR1, QU=>VARBOOL5, QD=>VARBOOL6, CV=>VAR2);

3.20 定时器（Util.Standard）

在软件中，有实时时钟指令 RTC，以及普通定时器 TP、通电延时定时器 TON、断电延时定时器 TOF 三类定时器指令。

3.20.1 RTC——实时时钟

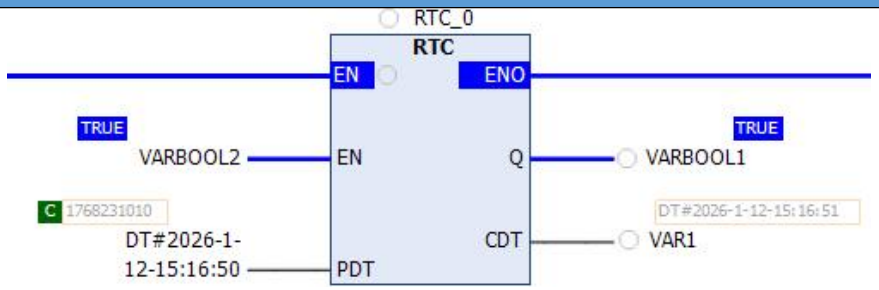
实时时钟指令 RTC 各参数的含义及相关说明见下表。

输入参数	数据类型	功能描述	参数值说明
EN	BOOL	使能信号	
PDT	DT	时间基值	
输出参数	**数据类型**	**功能描述**	**参数值说明**
Q	BOOL	定时器输出	EN 为“0”时，Q 为“0”；EN 为“1”时，Q 为“1”。

CDT	DT	当前时间值	EN 为“0”时，CDT 为1970-01-01-00:00:00；EN 为“1”时，CDT 从 PDT 提供的时间开始计时。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	RTC_0		RTC			
VAR	VAR1		DT			
VAR	VARBOOL1		BOOL			
VAR	VARBOOL2		BOOL			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	RTC_0(EN:=VARBOOL2, PDT:=DT#2026-1-12-15:16:50, Q=>VARBOOL1, CDT=>VAR1);

1.EN 为 TRUE (上升沿)，PDT 将接收时间给定值，并以秒为单位开始计时。

2.EN 为 TRUE 时，实时变化的计时值将传递给 CDT。EN 被复位为 FALSE 时，CDT 将被复位为系统初始值。

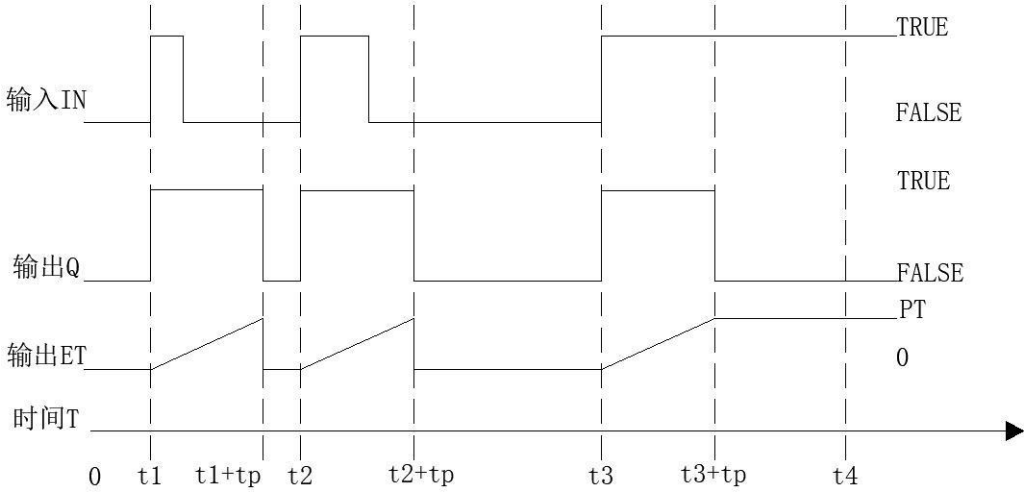
3.20.2 TP——普通定时器

普通定时器指令 TP 各参数的含义及相关说明见下表。

输入参数	数据类型	功能描述	参数值说明
IN	BOOL	定时器初始化信号	当 IN 由“0”变为“1”时，ET 以 ms 为单位计时，直至 ET 等于 PT 后 ET 保持不变
PT	TIME	定时时间值	

输出参数	数据类型	功能描述	参数值说明
Q	BOOL	定时器输出	IN 为“0”时，Q 为“0”，ET 为 0。IN 为“1”时，TP 开始计时，Q 为“1”。开始计时后，在 ET 小于等于 PT 前，IN 变化对计时过程无影响。ET 计时至等于PT 时，Q 变为“0”。计时结束后，当 IN 变为“0”，ET 等于 0。
ET	TIME	当前时间值	

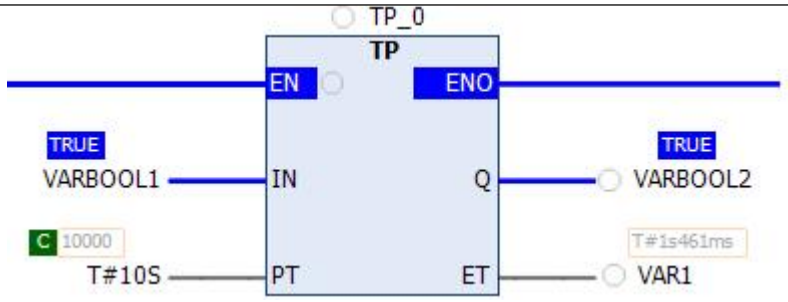
普通定时器 TP 的输入、输出关系如下图所示。



变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	TP_0		TP			
VAR	VAR1		TIME			
VAR	VARBOOL2		BOOL			
VAR	VARBOOL1		BOOL			

编程语言	程序

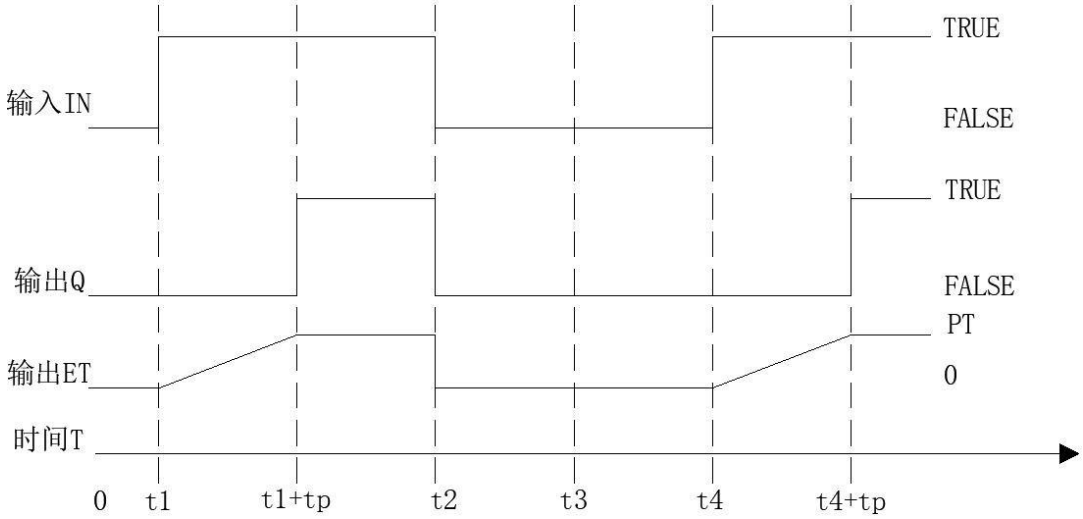
梯形图（LD2）	
结构化文本（ST）	TP_0(IN:=VARBOOL1, PT:=T#10s, Q=>VARBOOL2, ET=>VAR1);

3.20.3 TON——通电延时定时器

通电延时定时器指令 TON 各参数的含义及相关说明见下表。

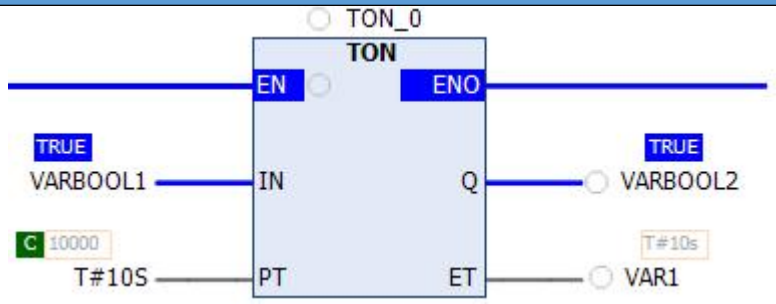
输入参数	数据类型	功能描述	参数值说明
IN	BOOL	定时器初始化信号	当 IN 由“0”变为“1”时，ET 以 ms 为单位计时，直至 ET 等于 PT 后 ET 保持不变。
PT	TIME	定时时间值	
输出参数	**数据类型**	**功能描述**	**参数值说明**
Q	BOOL	定时器输出	IN 为“0”时，Q 为“0”，ET 为 0。IN 为“1”时，TP 开始计时，ET 等于 PT 后 Q 为“1”开始计时后，在 ET 等于 PT 前，若 IN 由“1”变为“0”，则计时过程终止，ET 等于 0。
ET	TIME	当前时间值	

通电延时定时器 TON 的输入、输出关系如下图所示。



变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**TON_0**		TON			
VAR	**VAR1**		TIME			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL1**		BOOL			

编程语言	程序
梯形图（LD2）	 VARBOOL1 触点为 TRUE，延时 10s 后 VARBOOL2 输出为 TRUE
结构化文本（ST）	TON_0(IN:=VARBOOL1, PT:=T#10s, Q=>VARBOOL2, ET=>VAR1);

如果 IN 为 FALSE，Q 将会为 FALSE，且 ET 为 0。一旦 IN 为 TRUE，在 ET 端将会以毫秒为单位进行计时，直到 ET 等于 PT 值，其后将保持该常量值。当 IN 为

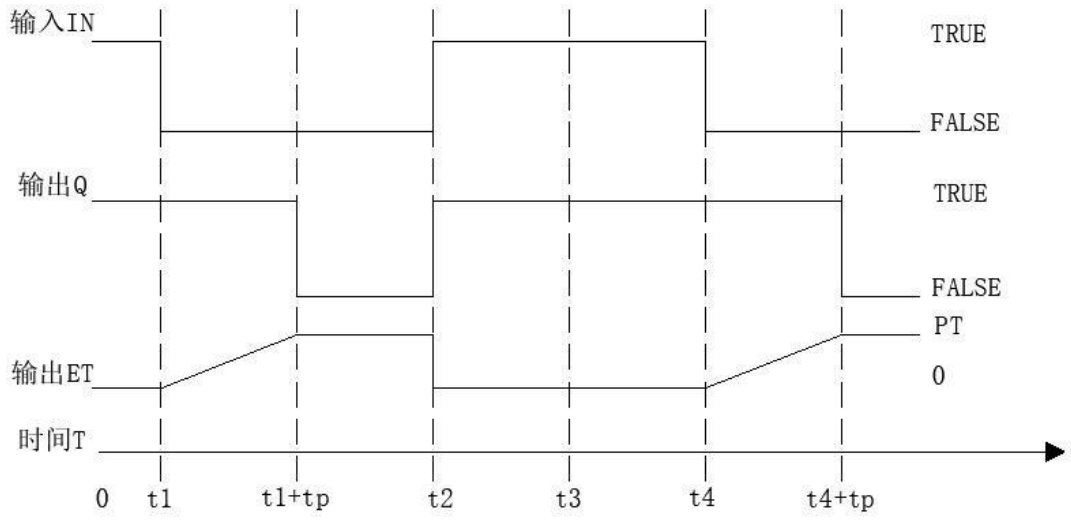
TRUE 并且 ET 等于 PT 时，Q 为 TRUE，否则 Q 为 FALSE。因此，以毫秒为单位计时，达到 PT 的输入值后，Q 会产生一个上升沿。

3.20.4 TOF——断电延时定时器

断电延时定时器指令 TOF 各参数的含义及相关说明见下表。

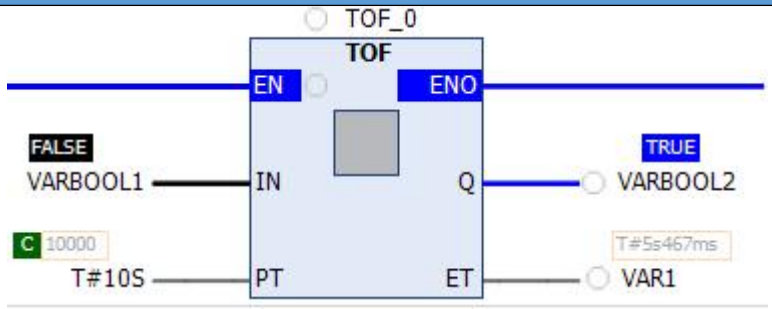
输入参数	数据类型	功能描述	参数值说明
IN	BOOL	定时器初始化信号	当 IN 由“1”变为“0”时，ET 以 ms 为单位计时，直至 ET 等于 PT 后 ET 保持不变。
PT	TIME	定时时间值	
输出参数	**数据类型**	**功能描述**	**参数值说明**
Q	BOOL	定时器输出	IN 为“1”时，Q 为“1”，ET 为 0。IN 为“0” 时，TP 开始计时，ET 等于 PT 后 Q 为“0” 开始计时后，在 ET 等于 PT 前若 IN 由“0” 变为“1”，则计时过程终止，ET 等于 0。
ET	TIME	当前时间值	

断电延时定时器 TOF 的输入、输出关系如下图所示。



变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	TOF_0		TOF			
VAR	VAR1		TIME			
VAR	VARBOOL2		BOOL			
VAR	VARBOOL1		BOOL			

编程语言	程序
梯形图（LD2）	 VARBOOL1 由“TRUE”变为“FALSE”，延时 10s 后 VARBOOL2 断开
结构化文本（ST）	TOF_0(IN:=VARBOOL1, PT:=T#10s, Q=>VARBOOL2, ET=>VAR1);

3.21 双稳态指令（Util.Standard）

3.21.1 SR——置位优先双稳态触发器

置位优先双稳态触发器指令 SR 的逻辑关系为：Q1:= SET1 OR (NOT RESET AND Q1)，其中 SET1 为置位信号，RESET 为复位信号，Q1 为输出信号。SR 触发器的真值表如下所示：

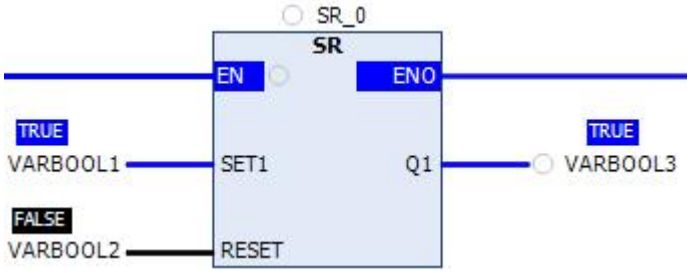
SET1	RESET	输出
0	0	保持原状态
1	0	1
0	1	0
1	1	1

指令对应的输入和输出数据类型

输入和输出的数据类型均为 BOOL 型。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	SR_0		SR			
VAR	VARBOOL1		BOOL			
VAR	VARBOOL2		BOOL			
VAR	VARBOOL3		BOOL			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	SR_0(SET1:=VARBOOL1, RESET:=VARBOOL2, Q1=>VARBOOL3);

3.21.2 RS——复位优先双稳态触发器

复位优先双稳态触发器指令 RS 的逻辑关系为：Q1:= NOT RESET1 AND (Q1 OR SET)，其中 SET1 为置位信号，RESET 为复位信号，Q1 为输出信号。RS 触发器的真值表如下所示：

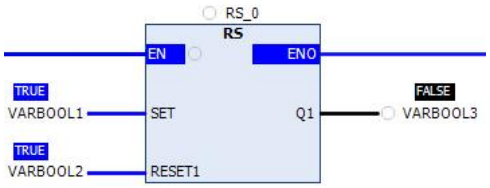
SET	RESET1	输出
0	0	保持原状态
1	0	1
0	1	0
1	1	0

指令对应的输入和输出数据类型

输入和输出的数据类型为 BOOL 型。

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			
VAR	**RS_0**		RS			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	RS_0(SET:=VARBOOL1, RESET1:=VARBBOL2, Q1=>VARBOOL3);

3.22 触发器指令（Util.Standard）

触发器分为两类，下降沿检测触发器 F_TRIG 用于检测下降沿，上升沿检测触发器 R_TRIG 用于检测上升沿。

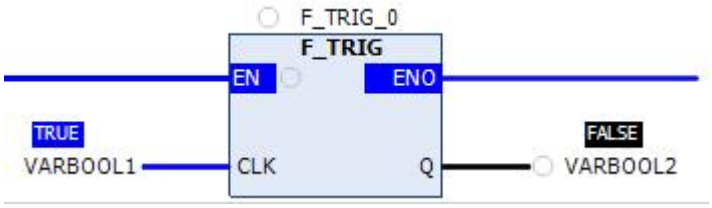
3.22.1 F_TRIG——下降沿检测触发器

下降沿检测触发器 F_TRIG 检测到输入信号的下降沿时，将向输出端发送触发信号。该指令的语法格式为：Q := NOT CLK AND NOT M; M := NOT CLK; 在该指令的编程语言中看不到 M，M 是BOOL 型中间变量，其初始值为“0”。当 CLK 是由“1”变为“0”时，Q 维持一个周期的高电平，下一周期 Q 又被 M 置为低电平。

指令对应的输入和输出数据类型
输入和输出的数据类型为 BOOL 型。

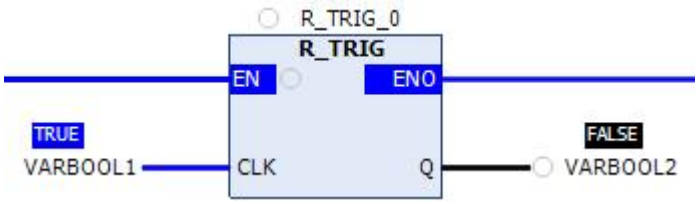
变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**F_TRIG_0**		F_TRIG			

编程语言	程序

梯形图（LD2）	
结构化文本（ST）	F_TRIG_0(CLK:=VARBOOL1, Q=>VARBOOL2);

3.22.2 R_TRIG——上升沿检测触发器

上升沿检测触发器 R_TRIG 检测到输入信号的上升沿时，将向输出端发送触发信号。该指令的语法格式为：Q :=CLK AND NOT M; M :=CLK;在该指令的编程语言中看不到 M，M 是BOOL 型中间变量，其初始值为“1”。当CLK 是由“0”变为“1”时，Q 维持一个周期的高电平，下一周期 Q 又被 M 置为低电平。

指令对应的输入和输出数据类型						
输入和输出的数据类型为 BOOL 型。						
变量定义						
类别	名称	地址	数据类型	初值	注释	属性
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**R_TRIG_0**		R_TRIG			
编程语言	**程序**					
梯形图（LD2）						
结构化文本（ST）	R_TRIG_0(CLK:=VARBOOL1, Q=>VARBOOL2);					

3.23 数学指令指令（Tianjin Sange Electr. Tech. Bronze100）

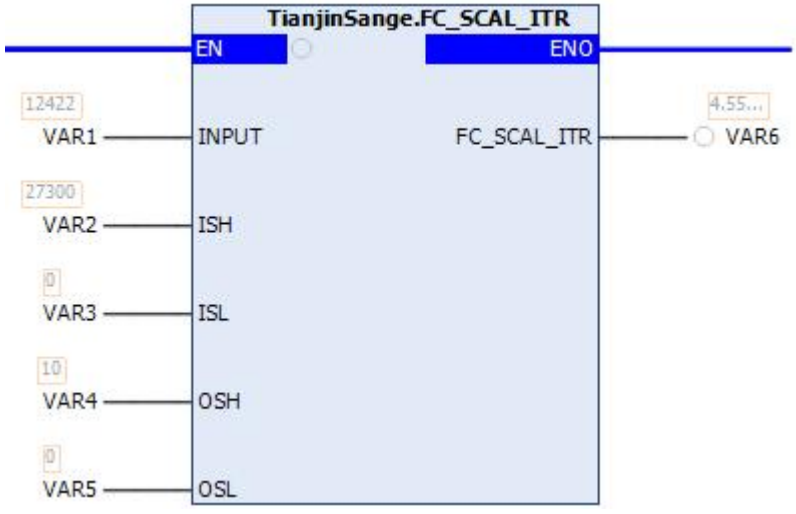
3.23.1 FC_SCAL_ITR——模拟量转工程量

模拟量转工程量指令 FC_SCAL_ITR 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
INPUT	INT	输入值	模拟量输入值
ISH	INT	输入值上限	模拟量输入上限值
ISL	INT	输入值下限	模拟量输入下限值
OSH	REAL	输出值上限	工程量输出上限值
OSL	REAL	输出值下限	工程量输出下限值
输出参数	**数据类型**	**功能描述**	**参数值说明**
FC_SCAL_ITR	REAL	输出值	工程量输出值

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VAR1**		DINT			
VAR	**VAR2**		DINT			
VAR	**VAR3**		DINT			
VAR	**VAR4**		REAL			
VAR	**VAR5**		REAL			
VAR	**VAR6**		REAL			

编程语言	程序
梯形图（LD2）	

结构化文本（ST）	TianjinSange.FC_SCAL_ITR(INPUT:=VAR1, ISH:=VAR2, ISL:=VAR3, OSH:=VAR4, OSL:=VAR5, FC_SCAL_ITR=>VAR6);

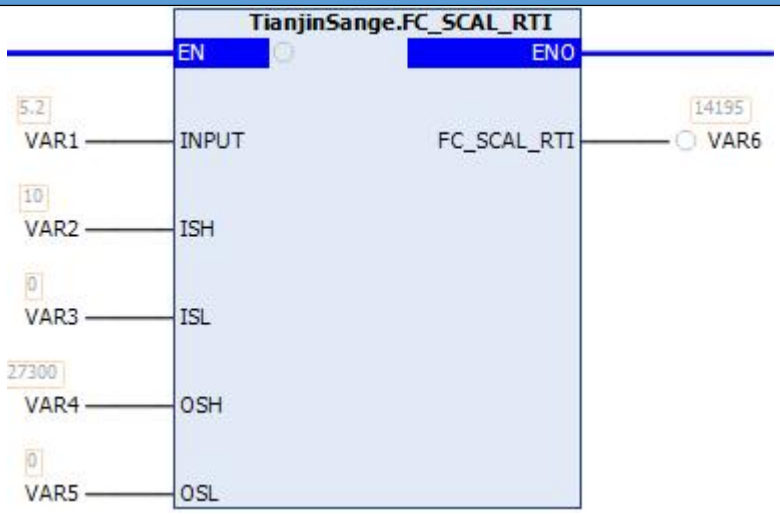
3.23.2 FC_SCAL_RTI——工程量转模拟量

工程量转模拟量指令 FC_SCAL_RTI 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
INPUT	REAL	输入值	工程量输入值
ISH	REAL	输入值上限	工程量输入上限值
ISL	REAL	输入值下限	工程量输入下限值
OSH	INT	输出值上限	模拟量输出上限值
OSL	INT	输出值下限	模拟量输出下限值
输出参数	**数据类型**	**功能描述**	**参数值说明**
FC_SCAL_RTI	INT	输出值	模拟量输出值

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
2	VAR	PROTOCOL		BYTE	0		
3	VAR	PORT		BYTE	1		
4	VAR	MODE		BYTE	0		
5	VAR	SLAVE		BYTE	51		
6	VAR	DATABITS		BYTE	3		
7	VAR	PARITY		BYTE	0		
8	VAR	BAUDRATE		BYTE	5		
9	VAR	ERROR		BYTE			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	TianjinSange.FC_SCAL_RTI(INPUT:=VAR1, ISH:=VAR2, ISL:=VAR3, OSH:=VAR4, OSL:=VAR5, FC_SCAL_RTI=>VAR6);

3.23.3 FB_BlockMove——批量数据传送指令

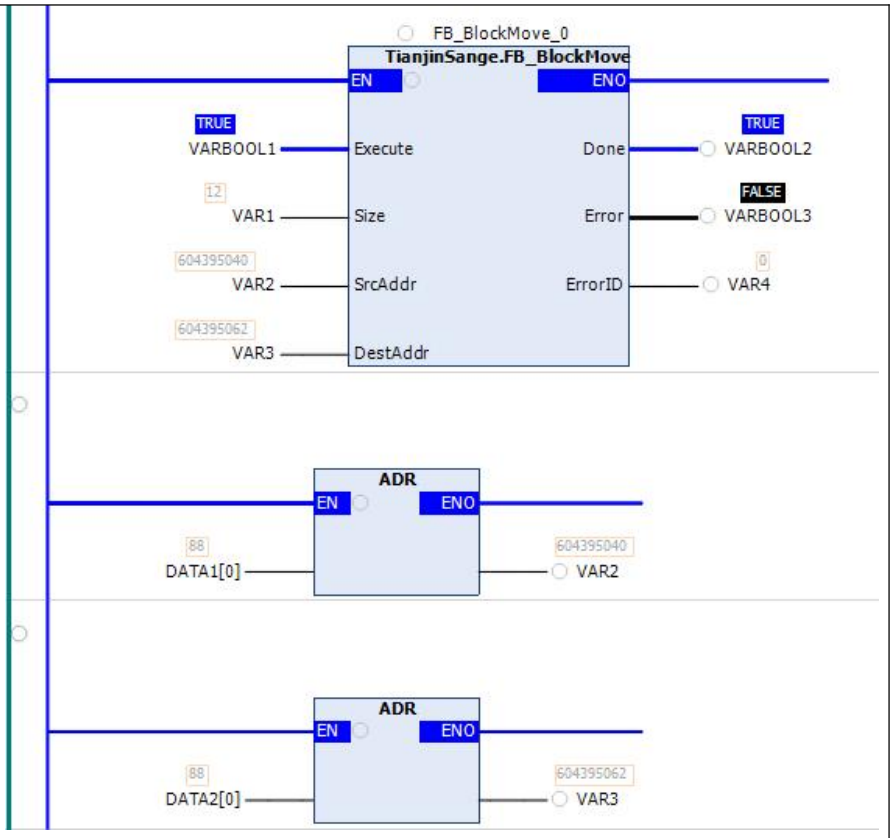
批量数据传送指令 FB_BlockMove 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Execute	BOOL	触发	上升沿有效
Size	UINT	要传送的数据量	1-65535
SrcAddr	DWORD	数据源地址	数据源地址
DestAddr	DWORD	数据目的地址	数据目的地址
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	传送完成标志	0：未传送完成 1：传送完成
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	BYTE	错误ID	0x01：内存地址错误

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**FB_BlockMove_0**		TianjinSange.FB_BlockMove			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			
VAR	**VAR1**		UINT			
VAR	**VAR2**		DWORD			
VAR	**VAR3**		DWORD			
VAR	**VAR4**		BYTE			
VAR	**DATA1**		ARRAY[0..10] OF INT			
VAR	**DATA2**		ARRAY[0..10] OF INT			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	FB_BlockMove_0(Execute:=VARBOOL1, Size:=VAR1, SrcAddr:=VAR2, DestAddr:=VAR3, Done=>VARBOOL2, Error=>VARBOOL3, ErrorID=>VAR4); VAR2:=ADR (DATA1[0]); VAR3:=ADR(DATA2[0]);

3.23.4 FB_ModbusCRC——计算Modbus CRC指令

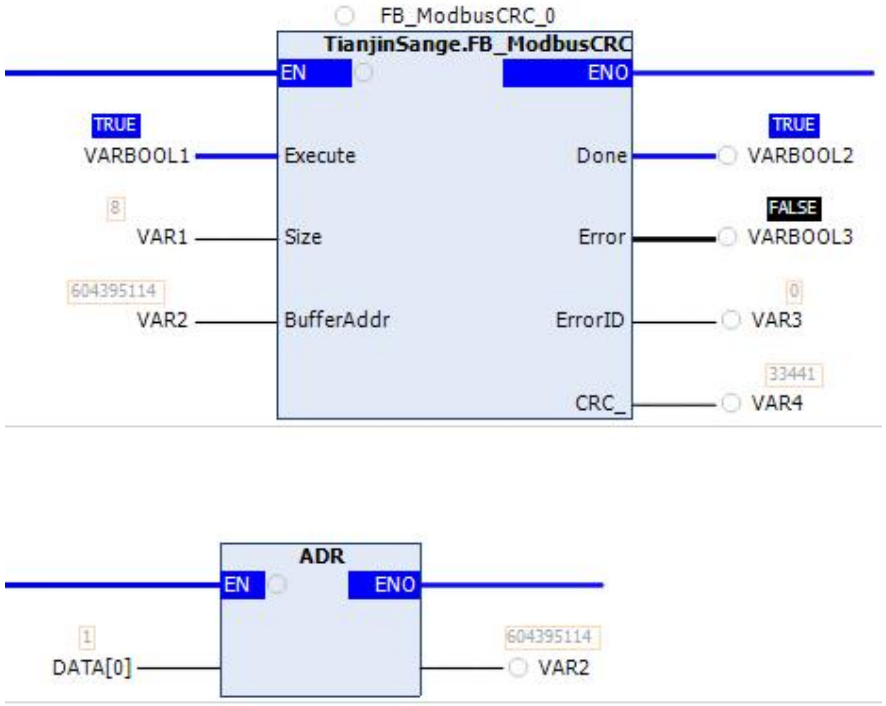
计算Modbus CRC指令 FB_ModbusCRC 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Eexcute	BOOL	触发	上升沿有效
Size	UINT	字节长度	要计算ModbusCRC的数据长度 1-65535
BufferAddr	BufferAddr	数据地址	要计算ModbusCRC的数据地址
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：未计算完成 1：计算完成
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	ErrorID	错误ID	0x01：内存地址错误
CRC_	CRC_	CRC值	CRC输出值

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**FB_ModbusCRC_0**		TianjinSange.FB_ModbusCRC			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			
VAR	**VAR1**		UINT			
VAR	**VAR2**		DWORD			
VAR	**VAR3**		BYTE			
VAR	**VAR4**		UINT			
VAR	**DATA**		ARRAY[0..10] OF INT			

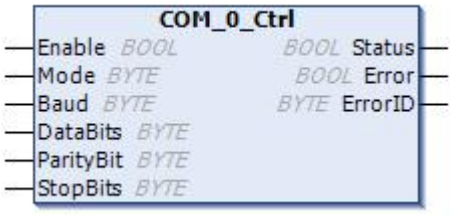
编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	FB_ModbusCRC_0(Execute:=VARBOOL1, Size:=VAR1, BufferAddr:=VAR2, Done=>VARBOOL2, Error=>VARBOOL3, ErrorID=>VAR3, CRC_=>VAR4); VAR2:=ADR (DATA[0]);

3.24 串口通讯指令（Tianjin Sange Electr. Tech. Bronze100）

3.24.1 COM_0_Ctrl——设置串口0通讯参数

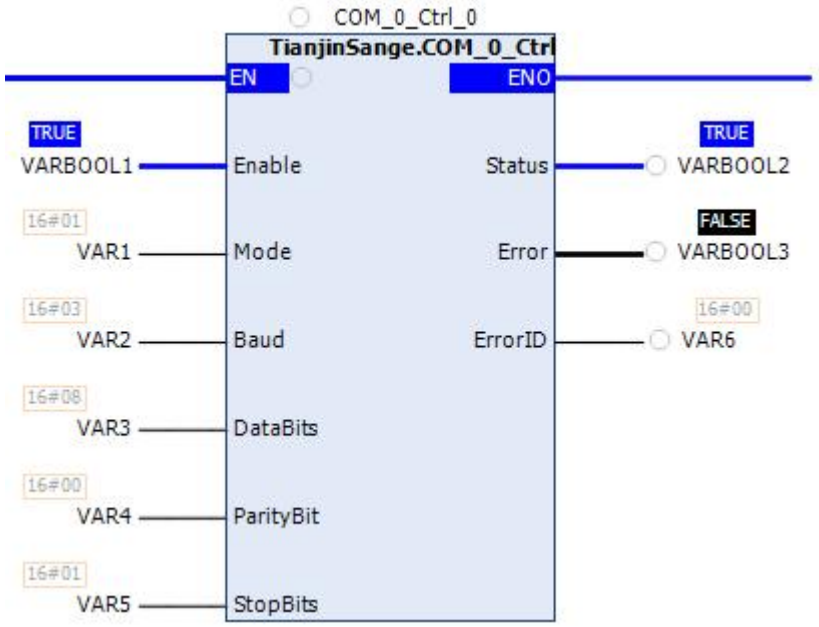
设置串口0通讯参数指令 COM_0_Ctrl 在库文件中的图示表达和相关参数说明如下：



输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：失能 1：使能（上升沿触发，后续操作保持高电平）
Mode	BYTE	协议类型	0：Modbus RTU从站 1：Modbus RTU主站 2：自由口
Baud	BYTE	波特率	1：2400 2：4800 3：9600 4：19200 5：38400 6：57600 7：115200
DataBits	BYTE	数据位	固定为8，不可更改
ParityBit	BYTE	校验位	0：无校验 1：奇校验 2：偶校验
StopBits	BYTE	停止位	1：1位 2：2位
输出参数	**数据类型**	**功能描述**	**参数值说明**
Status	BOOL	串口使能标志	0：串口未使能 1：串口已使能
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x01：模式错误 0x02：波特率错误 0x08：校验位错误 0x10：停止位错误

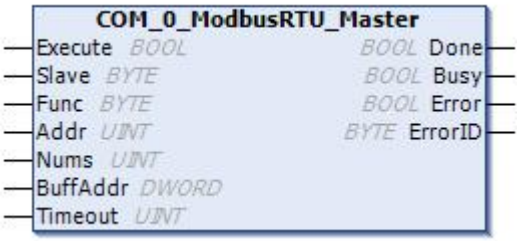
变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	COM_0_Ctrl_0		TianjinSange.COM_0_Ctrl			
VAR	VARBOOL1		BOOL			
VAR	VARBOOL2		BOOL			
VAR	VARBOOL3		BOOL			
VAR	VAR1		BYTE			
VAR	VAR2		BYTE			
VAR	VAR3		BYTE			
VAR	VAR4		BYTE			
VAR	VAR5		BYTE			
VAR	VAR6		BYTE			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	COM_0_Ctrl_0(Enable:=VARBOOL1, Mode:=VAR1, Baud:=VAR2, DataBits:=VAR3, ParityBit:=VAR4, StopBits:=VAR5, Status=>VARBOOL2, Error=>VARBOOL3, ErrorID=>VAR6);

3.24.2 COM_0_ModbusRTU_Master——Modbus RTU主站

Modbus RTU主站指令 COM_0_ModbusRTU_Master 在库文件中的图示表达和相关参数说明如下：

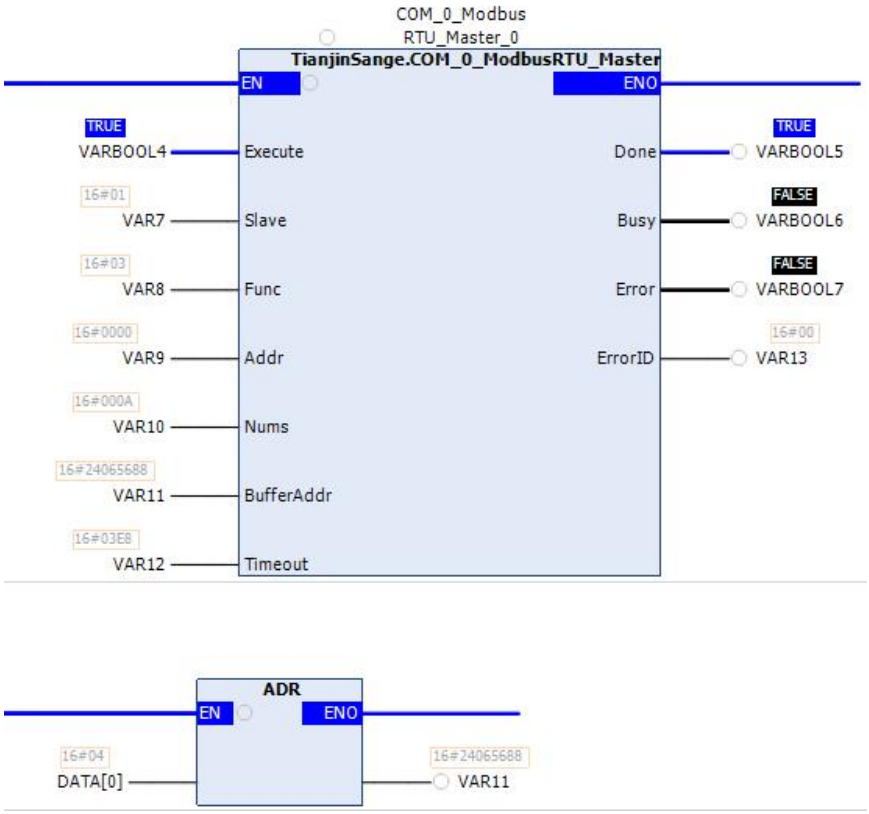


输入参数	数据类型	功能描述	参数值说明
Execute	BOOL	触发	上升沿有效
Slave	BYTE	从站号	
Func	BYTE	Modbus功能码	支持1、2、3、4、5、6、15、16
Addr	UINT	寄存器地址	
Nums	UINT	寄存器个数	寄存器类型1-120 离散输入/线圈1-1920
BufferAddr	DWORD	内存地址	读取或写入Modbus寄存器的内容的地址
Timeout	UINT	应答超时时间	1-65535，单位ms
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	指令完成标志	0：未完成 1：完成
Busy	BOOL	指令忙标志	0：指令未指令 1：指令正在执行
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x01：功能码错误 0x02：寄存器个数错误 0x04：内存地址错误 0x81：模式错误，串口不是Modbus RTU主站模式

			0x82：应答超时 0xC1：应答CRC错误 0xC2：应答错误

变量定义

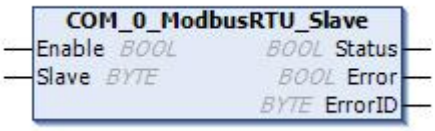
类别	名称	地址	数据类型	初值	注释	属性
VAR	COM_0_ModbusRTU_Master_0		TianjinSange.COM_0_ModbusRTU_Master			
VAR	VARBOOL4		BOOL			
VAR	VARBOOL5		BOOL			
VAR	VARBOOL6		BOOL			
VAR	VARBOOL7		BOOL			
VAR	VAR7		BYTE			
VAR	VAR8		BYTE			
VAR	VAR9		UINT			
VAR	VAR10		UINT			
VAR	VAR11		DWORD			
VAR	VAR12		UINT			
VAR	VAR13		BYTE			
VAR	DATA		ARRAY[0..20] OF byte			

编程语言	程序
梯形图 （LD2）	 0 3功能码读取保持寄存器40001-40010，放到DATA[0]-DATA[19]

结构化文本（ST）	COM_0_ModbusRTU_Master_0(Execute:=VARBOOL4, Slave:=VAR7, Func:=VAR8, Addr:=VAR9, Nums:=VAR10, BufferAddr:=VAR11, Timeout:=VAR12, Done=>VARBOOL5, Busy=>VARBOOL6, Error=>VARBOOL7, ErrorID=>VAR13); VAR11:=ADR (DATA[0]);

3.24.3 COM_0_ModbusRTU_Slave——Modbus RTU从站

Modbus RTU从站功能块指令 COM_0_ModbusRTU_Slave 在库文件中的图示表达和相关参数说明如下：

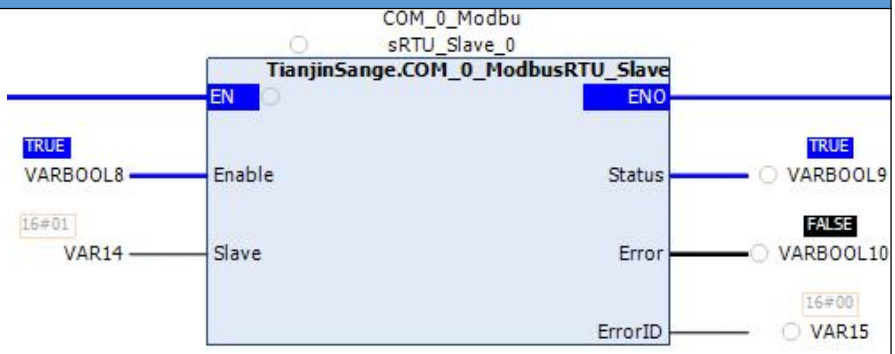


输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：失能 1：使能（上升沿触发，后续操作保持高电平）
Slave	BYTE	本机从站号	1-247
输出参数	**数据类型**	**功能描述**	**参数值说明**
Status	BOOL	从站使能标志	0：从站未使能 1：从站使能

Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x01：从站号错误 0x81：模式错误，串口不是Modbus RTU从站模式

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	VARBOOL8		BOOL			
VAR	VAR14		BYTE			
VAR	VARBOOL9		BOOL			
VAR	VARBOOL10		BOOL			
VAR	VAR15		BYTE			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	COM_0_ModbusRTU_Slave_0(Enable:=VARBOOL8, Slave:=VAR14, Status=>VARBOOL9, Error=>VARBOOL10, ErrorID=>VAR15);

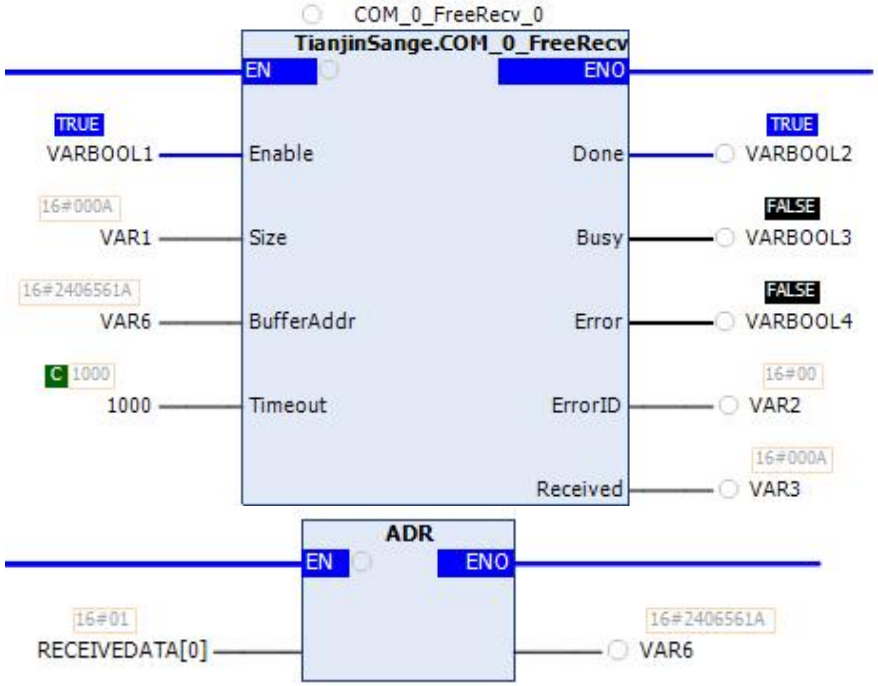
3.24.4 COM_0_FreeRecv——串口自由口接收

串口自由口接收指令 COM_0_FreeRecv 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：失能 1：使能（上升沿触发，后续操作保持高电平）
Size	UINT	要接收数据的长度	1-1536字节
BufferAddr	DWORD	内存地址	要接收数据的存放地址
Timeout	UINT	接收超时时间	1-65535，单位ms
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	指令完成标志	0：未完成 1：完成
Busy	BOOL	指令忙标志	0：指令未执行 1：指令正在执行
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x02：接收数据长度错误 0x04：内存地址错误 0x81：模式错误，串口不是自由口 0x82：接收超时
Received	UINT	接收到的字节数	接收到的字节数

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			
VAR	**VARBOOL4**		BOOL			
VAR	**VAR1**		UINT			
VAR	**VAR2**		BYTE			
VAR	**VAR3**		UINT			
VAR	**VAR6**		DWORD			
VAR	**RECEIVEDATA**		ARRAY[0..100] OF BYTE			
VAR	**COM_0_FreeRecv_0**		TianjinSange.COM_0_FreeRecv			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	COM_0_FreeRecv_0(Enable:=VARBOOL1, Size:=VAR1, BufferAddr:=VAR6, Timeout:=1000, Done=>VARBOOL2, Busy=>VARBOOL3, Error=>VARBOOL4, ErrorID=>VAR2, Received=>VAR3);

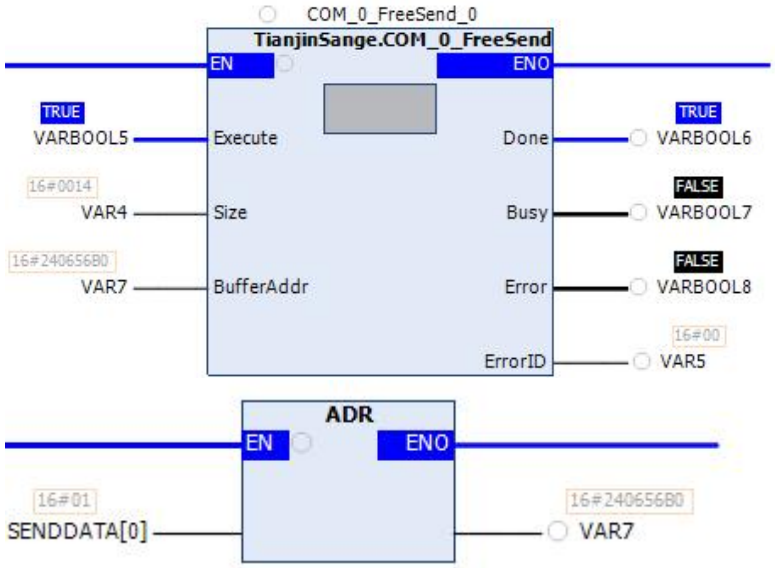
3.24.5 COM_0_FreeSend——串口自由口发送

串口自由口发送指令 COM_0_FreeSend 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Execute	BOOL	触发	上升沿有效
Size	UINT	要发送数据的长度	1-1536字节
BufferAddr	DWORD	内存地址	要发送数据的地址
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	指令完成标志	0：未完成 1：完成
Busy	BOOL	指令忙标志	0：指令未指令 1：指令正在执行
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x02：发送数据长度错误 0x04：内存地址错误 0x81：模式错误，串口不是自由口

变量定义

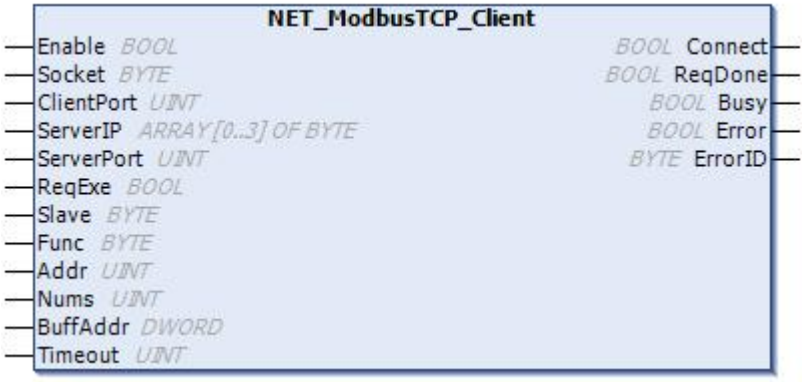
类别	名称	地址	数据类型	初值	注释	属性
VAR	**SENDDATA**		ARRAY[0..100] OF BYTE			
VAR	**COM_0_FreeSend_0**		TianjinSange.COM_0_FreeSend			
VAR	**VARBOOL5**		BOOL			
VAR	**VARBOOL6**		BOOL			
VAR	**VARBOOL7**		BOOL			
VAR	**VARBOOL8**		BOOL			
VAR	**VAR4**		UINT			
VAR	**VAR5**		BYTE			
VAR	**VAR7**		DWORD			

编程语言	程序
梯形图（LD2）	
结构化文本（ST）	COM_0_FreeSend_0(Execute:=VARBOOL5, Size:=VAR4, BufferAddr:=VAR7, Done=>VARBOOL6, Busy=>VARBOOL7, Error=>VARBOOL8, ErrorID=>VAR5,);

3.25 以太网口通讯指令（Tianjin Sange Electr. Tech. Bronze100）

3.25.1 NET_Modbus TCP_Client——Modbus TCP Client

Modbus TCP Client指令 NET_ModbusTCP_Client 在库文件中的图示表达和相关参数说明如下：

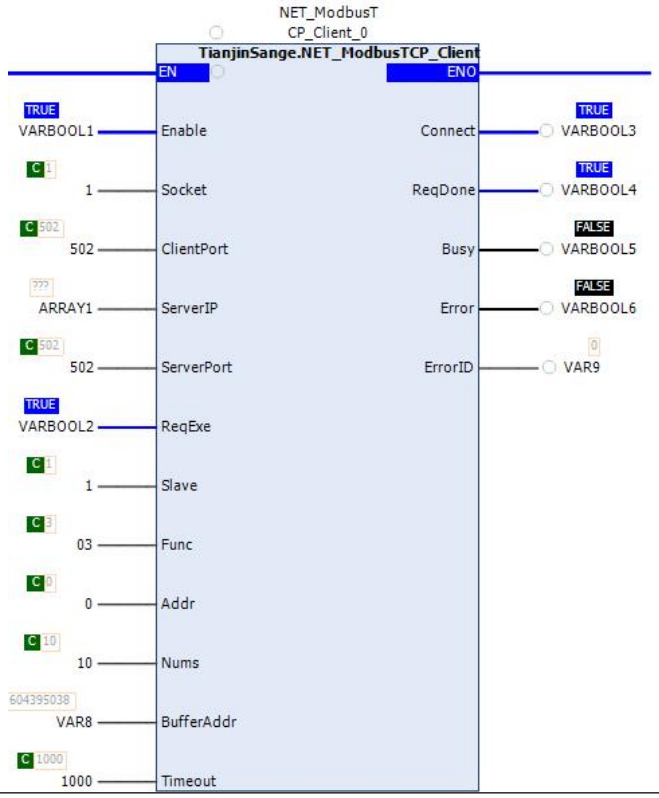


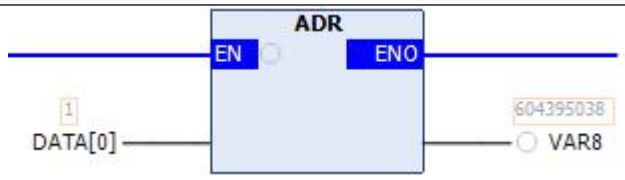
输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：关闭 socket 1：打开 socket连接目标IP（上升沿触发，后续操作保持高电平）
Socket	BYTE	Socket ID	0-5
ClientPort	UINT	客户端端口号	0-65535 如果为0则随机分配
ServerIP	ARRAY [0..3] OF BYTE	服务器IP地址	数组变量定义，如[192.168.1.10] 中的'192.168.1.10'为IP 地址
ServerPort	UINT	服务器端口号	1-65535，一般为502
ReqExe	BOOL	触发	上升沿有效
Slave	BYTE	从站号	
Func	BYTE	Modbus功能码	支持1、2、3、4、5、6、15、16
Addr	UINT	寄存器地址	
Nums	UINT	寄存器个数	寄存器类型1-120 离散输入/线圈1-1920
BufferAddr	DWORD	内存地址	读取或写入Modbus寄存器的内容的地址

Timeout	UINT	应答超时时间	1-65535，单位ms
输出参数	**数据类型**	**功能描述**	**参数值说明**
Connect	BOOL	连接服务器成功标志	0：未连接服务器 1：连接服务器成功
ReqDone	BOOL	指令完成标志	0：未完成 1：完成
Busy	BOOL	指令忙标志	0：指令未指令 1：指令正在执行
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x01：功能码错误 0x02：寄存器个数错误 0x04：内存地址错误 0x81：Socket错误 0x84：应答超时 0xC2：应答错误

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**NET_ModbusTCP_Client_0**		TianjinSange.NET_ModbusTCP_Client			
VAR	**VARBOOL1**		BOOL			
VAR	**VAR1**		BYTE			
VAR	**VAR2**		UINT			
VAR	**ARRAY1**		ARRAY [0..3] OF BYTE			
VAR	**VAR3**		UINT			
VAR	**VARBOOL2**		BOOL			
VAR	**VAR4**		BYTE			
VAR	**VAR5**		BYTE			
VAR	**VAR6**		UINT			
VAR	**VAR7**		UINT			
VAR	**VAR8**		DWORD			
VAR	**VAR9**		UINT			
VAR	**VARBOOL3**		BOOL			
VAR	**VARBOOL4**		BOOL			
VAR	**VARBOOL5**		BOOL			
VAR	**VARBOOL6**		BOOL			
VAR	**DATA**		ARRAY [0..100] OF BYTE			

编程语言	程序
梯形图（LD2）	

	
结构化文本（ST）	NET_ModbusTCP_Client_0(Enable:=VARBOOL1, Socket:=1, ClientPort:=502, ServerIP:=ARRAY1, ServerPort:=502, ReqExe:=VARBOOL2, Slave:=1, Func:=3, Addr:=0, Nums:=10, BufferAddr:=VAR8, Timeout:=1000, Connect=>VARBOOL3, ReqDone=>VARBOOL4, Busy=>VARBOOL5, Error=>VARBOOL6, ErrorID=>VAR9); VAR8:=DATA[0];

在 1 个程序中最多可以有 6 个实例。

3.25.2 NET_ModbusTCP_Server——Modbus TCP Server

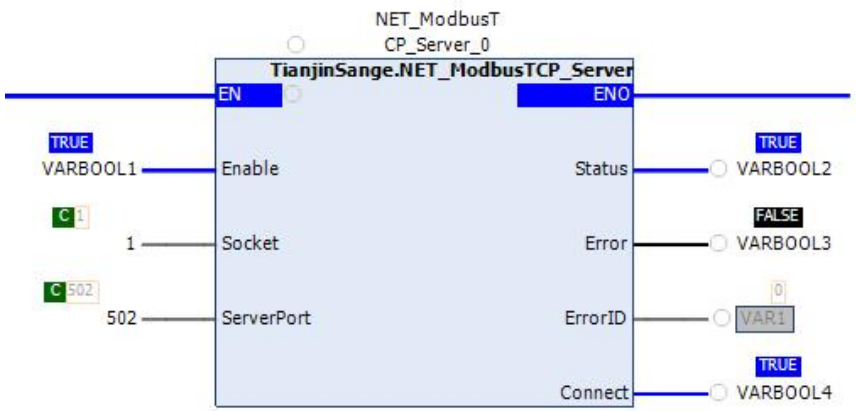
Modbus TCP Server指令 NET_ModbusTCP_Server 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：关闭 socket 1：打开 socket 监听端口（上升沿触发，后续操作保持高电平）
Socket	BYTE	Socket ID	0-5
ServerPort	UINT	服务器端口号	1-65535，一般为502
输出参数	**数据类型**	**功能描述**	**参数值说明**
Status	BOOL	从站使能标志	0：从站未使能 1：从站使能
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x81：Socket错误 0xC2：指令错误
Connect	BOOL	连接状态	0：没有客户端连接 1：有客户端连接

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**NET_ModbusTCP_Server_0**		TianjinSange.NET_ModbusTCP_Server			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			
VAR	**VAR1**		BYTE			
VAR	**VARBOOL4**		BOOL			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	NET_ModbusTCP_Server_0(Enable:=VARBOOL1, Socket:=1, ServerPort:=502, Status=>VARBOOL2, Error=>VARBOOL3, ErrorID=>VAR1, Connect=>VARBOOL4,);

在 1 个程序中最多可以有 6 个实例。

3.25.3 NET_FreeTCP_Client——TCP Client自由口

TCP Client自由口指令 NET_FreeTCP_Client 在库文件中的图示表达和相关参数说明如下：

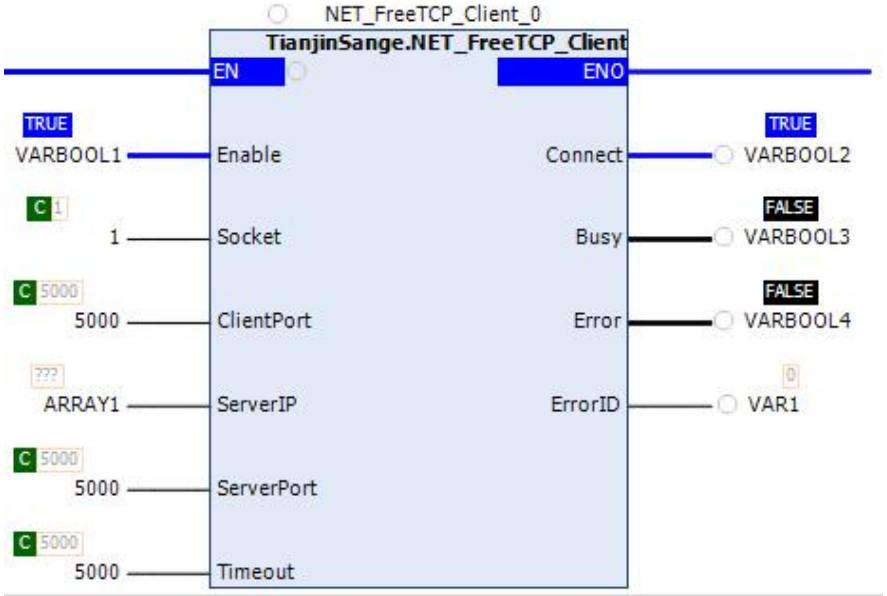
输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：关闭 socket 1：打开 socket连接目标IP（上升沿触发，后续操作保持高电平）

Socket	BYTE	Socket ID	0-5
ClientPort	UINT	客户端端口号	0-65535 如果为0则随机分配
ServerIP	ARRAY [0..3] OF BYTE	服务器IP地址	数组变量定义，如[192.168.1.10]中的'192.168.1.10'为IP 地址
ServerPort	UINT	服务器端口号	1-65535
Timeout	UINT	连接超时时间	1-65535，单位ms
输出参数	**数据类型**	**功能描述**	**参数值说明**
Connect	BOOL	连接状态	0：没有连接服务器 1：连接服务器成功
Busy	BOOL	忙标志	0：在尝试连接服务器 1：不在尝试连接服务器
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x81：Socket错误 0x82：连接服务器超时

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	NET_FreeTCP_Client_0		TianjinSange.NET_FreeTCP_Client			
VAR	ARRAY1		ARRAY [0..3] OF BYTE			
VAR	VARBOOL1		BOOL			
VAR	VARBOOL2		BOOL			
VAR	VARBOOL3		BOOL			
VAR	VARBOOL4		BOOL			
VAR	VAR1		BYTE			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	NET_FreeTCP_Client_0(Enable:=VARBOOL1, Socket:=1, ClientPort:=5000, ServerIP:=ARRAY1, ServerPort:=5000, Timeout:=5000, Connect=>VARBOOL2, Busy=>VARBOOL3, Error=>VARBOOL4, ErrorID=>VAR1,);

在 1 个程序中最多可以有 6 个实例。

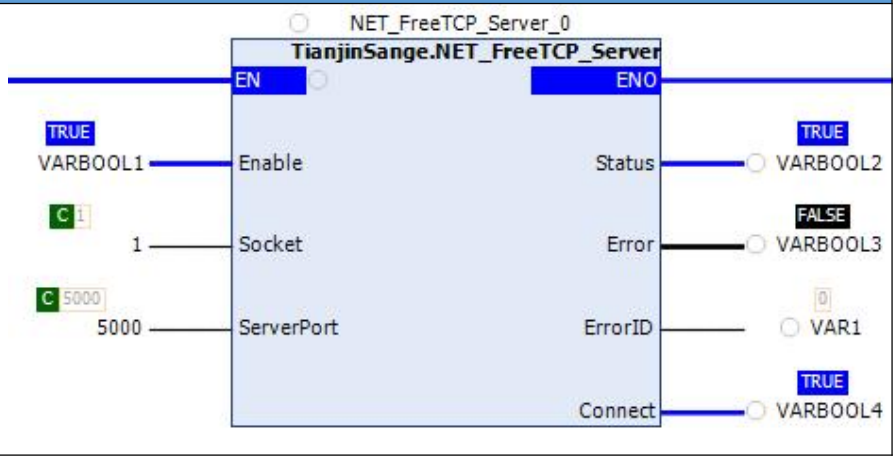
3.25.4 NET_FreeTCP_Server——TCP Server自由口

TCP Server自由口指令NET_FreeTCP_Server 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：关闭 socket 1：打开 socket 监听端口（上升沿触发，后续操作保持高电平）
Socket	BYTE	Socket ID	0-5
ServerPort	UINT	服务器端口号	1-65535
输出参数	**数据类型**	**功能描述**	**参数值说明**
Status	BOOL	状态标志	0：失能 1：使能
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x81：Socket错误
Connect	BOOL	连接状态	0：没有客户端连接 1：有客户端连接

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**NET_FreeTCP_Server_0**		TianjinSange.NET_FreeTCP_Server			
VAR	**VAR1**		BYTE			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			
VAR	**VARBOOL4**		BOOL			

编程语言	程序
梯形图（LD2）	

结构化文本（ST）	NET_FreeTCP_Server_0(Enable:=VARBOOL1, Socket:=1, ServerPort:=502, Status=>VARBOOL2, Error=>VARBOOL3, ErrorID=>VAR1, Connect=>VARBOOL4,);

在 1 个程序中最多可以有 6 个实例。

3.25.5 NET_FreeTCP_Recv——TCP自由口接收

TCP自由口接收指令 NET_FreeTCP_Recv 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：失能接收 1：使能接收
Socket	BYTE	Socket ID	0-5
Size	UINT	要接收数据的长度	1-1536字节
BufferAddr	DWORD	内存地址	要接收数据的存放地址
Timeout	UINT	接收超时时间	1-65535，单位ms
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	指令完成标志	0：未完成 1：完成
Busy	BOOL	指令忙标志	0：指令未执行 1：指令正在执行
Error	BOOL	错误	0：无错误

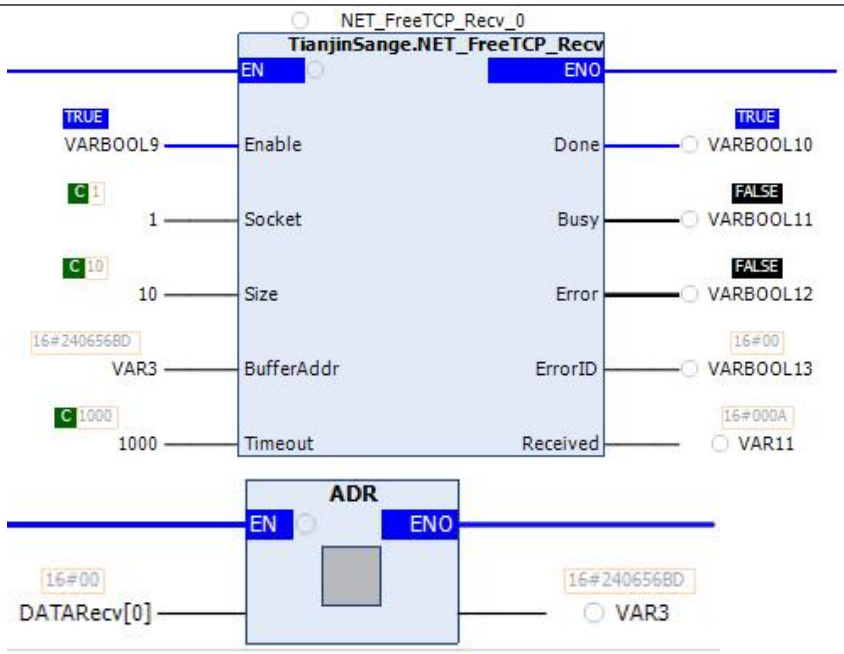
			1：有错误
ErrorID	BYTE	错误代码	0x02：接收数据长度错误 0x04：内存地址错误 0x81：Socket错误 0x83：TCP未连接 0x84：接收超时
Received	UINT	接收到的字节数	接收到的字节数

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	NET_FreeTCP_Recv_0		TianjinSange.NET_FreeTCP_Recv			
VAR	VARBOOL9		BOOL			
VAR	VARBOOL10		BOOL			
VAR	VARBOOL11		BOOL			
VAR	VARBOOL12		BOOL			
VAR	VARBOOL13		BYTE			
VAR	VAR3		DWORD			
VAR	VAR11		UINT			
VAR	DATARecv		ARRAY [0..100] OF BYTE			

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	ModbusTCP_Master_Ex_0		ModbusTCP_Master_Ex			
2	VAR	INDEX		BYTE	0		
3	VAR	IPADD		ARRAY [0..3] OF WORD	[192, 168, 1, 130]		
4	VAR	PORT		UINT	502		
5	VAR	REQ		BYTE	1		
6	VAR	RW		BYTE	0		
7	VAR	STATION		BYTE	0		
8	VAR	ADDRESS		DWORD	400000		
9	VAR	NUMBER		BYTE	20		
10	VAR	TBL		ARRAY[1..20] OF WORD			
11	VAR	TIMEOUT		DWORD	500		
12	VAR	VALID		BOOL			
13	VAR	STATUS		BYTE			
14	VAR	ERROR		BYTE			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	NET_FreeTCP_Recv_0(Enable:=VARBOOL9, Socket:=1, Size:=10, BufferAddr:=VAR3, Done=>VARBOOL10, Busy=>VARBOOL11, Error=>VARBOOL12, ErrorID=>VAR13, Received=>VAR11); VAR3:=ADR(DATARECV[0]);

在 1 个程序中最多可以有 6 个实例。

3.25.6 NET_FreeTCP_Send——TCP自由口发送

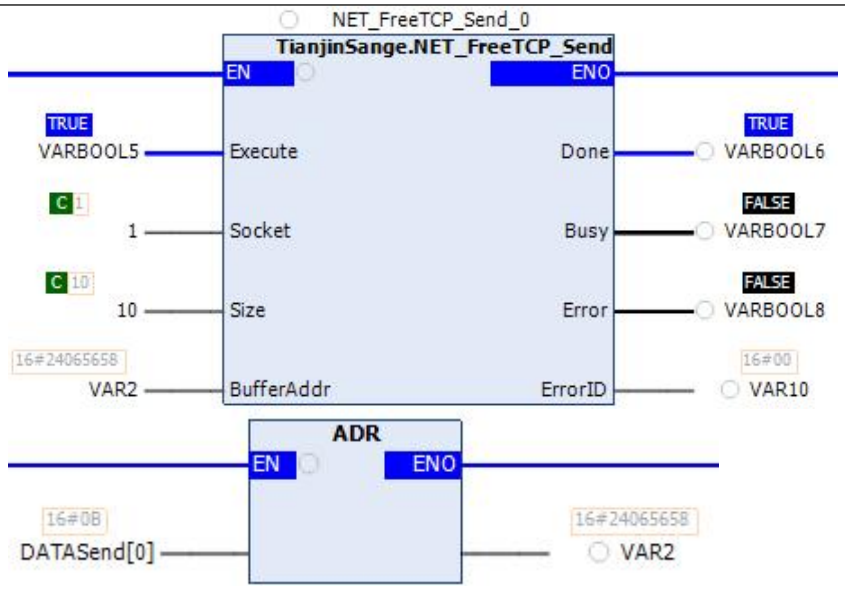
TCP自由口发送指令 NET_FreeTCP_Send 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Execute	BOOL	触发发送	上升沿有效
Socket	BYTE	Socket ID	0-5
Size	UINT	要发送数据的长度	1-1536字节
BufferAddr	DWORD	内存地址	要发送数据的地址
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	指令完成标志	0：未完成 1：完成
Busy	BOOL	指令忙标志	0：指令未指令 1：指令正在执行
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x02：发送数据长度错误 0x04：内存地址错误 0x81：Socket错误 0x83：TCP未连接

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	NET_FreeTCP_Send_0		TianjinSange.NET_FreeTCP_Send			
VAR	VARBOOL5		BOOL			
VAR	VARBOOL6		BOOL			
VAR	VARBOOL7		BOOL			
VAR	VARBOOL8		BOOL			
VAR	VAR2		DWORD			
VAR	VAR10		BYTE			
VAR	DATASend		ARRAY [0..100] OF BYTE			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	NET_FreeTCP_Send_0(Execute:=VARBOOL5, Socket:=1, Size:=10, BufferAddr:=VAR2, Done=>VARBOOL6, Busy=>VARBOOL7, Error=>VARBOOL8, ErrorID=>VAR10,); VAR2:=ADR(DATASend[0]);

在 1 个程序中最多可以有 6 个实例。

3.25.7 NET_FreeUDP——UDP自由口

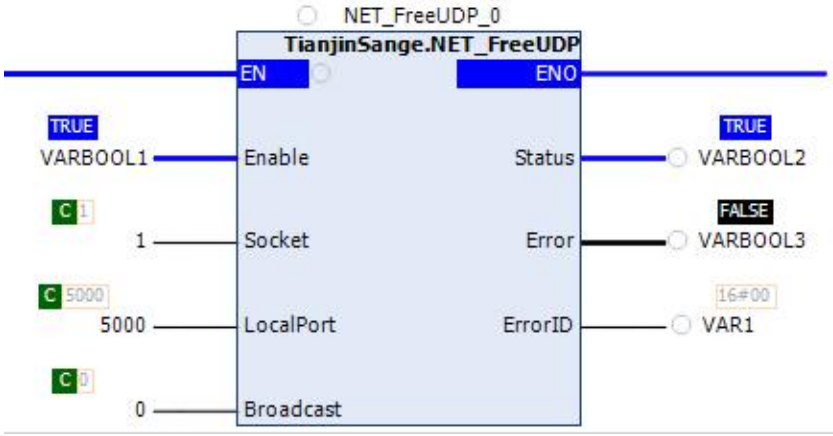
UDP自由口指令 NET_FreeUDP 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：关闭 socket

			1：打开 socke（上升沿触发，后续操作保持高电平）
Socket	BYTE	Socket ID	0-5
LocalPort	UINT	本机端口号	1-65535
Broadcast	BOOL	使能广播接收	0：不使能广播接收 1：使能广播接收
输出参数	**数据类型**	**功能描述**	**参数值说明**
Status	BOOL	状态标志	0：失能 1：使能
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x81：Socket错误

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	NET_FreeUDP_0		TianjinSange.NET_FreeUDP			
VAR	VARBOOL1		BOOL			
VAR	VARBOOL2		BOOL			
VAR	VARBOOL3		BOOL			
VAR	VAR1		BYTE			

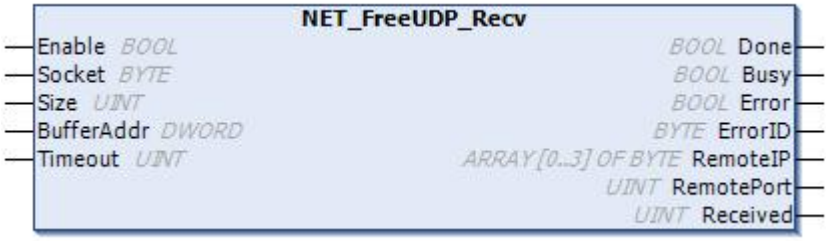
编程语言	程序
梯形图（LD2）	

结构化文本（ST）	NET_FreeUDP_0(Enable:=VARBOOL1, Socket:=1, LocalPort:=5000, Broadcast:=0, Status=>VARBOOL2, Error=>VARBOOL3, ErrorID=>VAR1,);

在 1 个程序中最多可以有 6 个实例。

3.25.8 NET_FreeUDP_Recv——UDP自由口接收

网口UDP自由口接收指令 NET_FreeUDP_Recv 在库文件中的图示表达和相关参数说明如下：



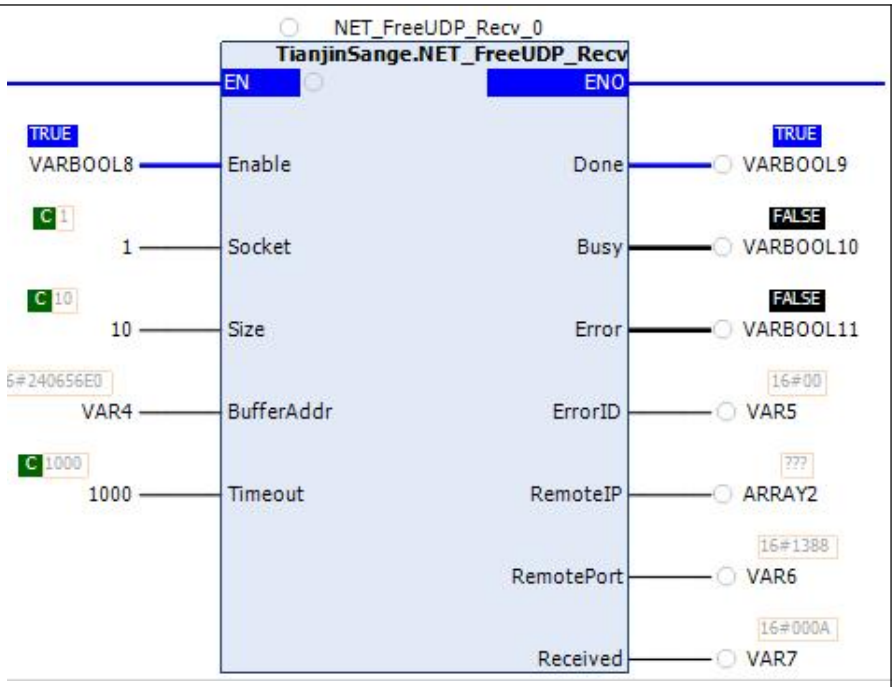
输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：失能接收 1：使能接收
Socket	BYTE	Socket ID	0-5
Size	UINT	要接收数据的长度	1-1472字节
BufferAddr	DWORD	内存地址	要接收数据的存放地址
Timeout	UINT	接收超时时间	1-65535，单位ms
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	指令完成标志	0：未完成 1：完成

Busy	BOOL	指令忙标志	0：指令未执行 1：指令正在执行
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x02：接收数据长度错误 0x04：内存地址错误 0x81：Socket错误 0x83：UDP未创建 0x84：接收超时
RemoteIP	ARRAY [0..3] OF BYTE	发送方IP地址	数组变量定义，如[192.168.1.10] 中的'192.168.1.10'为IP 地址
RemotePort	UINT	发送方端口号	1-65535
Received	UINT	接收到的字节数	接收到的字节数

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	NET_FreeUDP_Recv_0		TianjinSange.NET_FreeUDP_Recv			
VAR	VARBOOL8		BOOL			
VAR	VAR4		DWORD			
VAR	VARBOOL9		BOOL			
VAR	VARBOOL10		BOOL			
VAR	VARBOOL11		BOOL			
VAR	VAR5		BYTE			
VAR	ARRAY2		ARRAY [0..3] OF BYTE			
VAR	VAR6		UINT			
VAR	VAR7		UINT			
VAR	DATARecv		ARRAY [0..100] OF BYTE			

编程语言	程序

梯形图 （LD2）	
结构化文本 （ST）	NET_FreeUDP_Recv_0(Enable:=VARBOOL8, Socket:=1, Size:=10, BufferAddr:=VAR4, Timeout:=1000, Done=>VARBOOL9, Busy=>VARBOOL10, Error=>VARBOOL11, ErrorID=>VAR5, RemoteIP=>ARRAY2, RemotePort=>var6, Received=>var7); VAR4:=ADR(DATARecv[0]);

在 1 个程序中最多可以有 6 个实例。

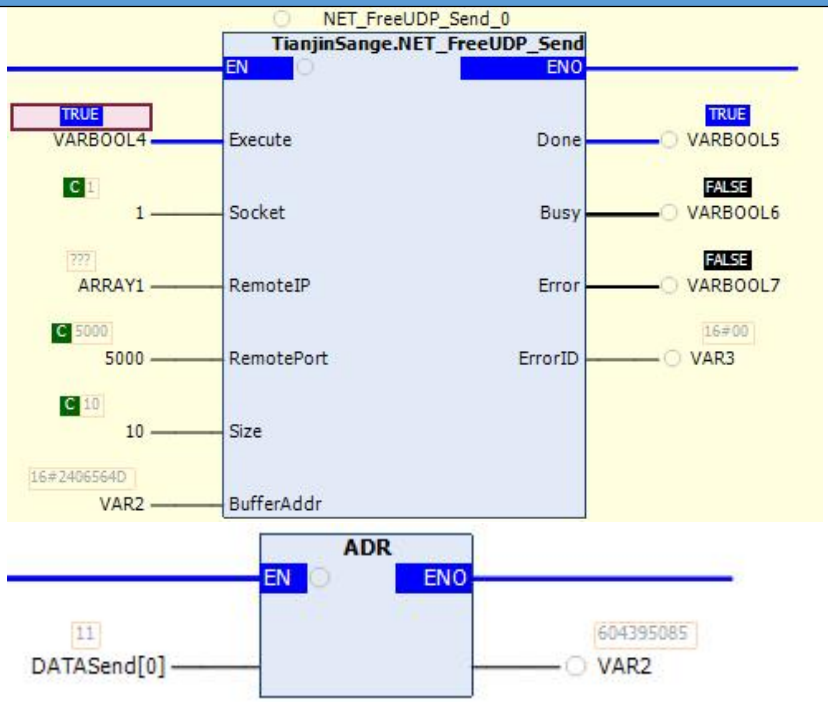
3.25.9 NET_FreeUDP_Send——网口UDP自由口发送

网口UDP自由口发送指令 NET_FreeUDP_Send 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Execute	BOOL	触发发送	上升沿有效
Socket	BYTE	Socket ID	0-5
RemoteIP	ARRAY [0..3] OF BYTE	目标IP地址	数组变量定义，如[192.168.1.10] 中的'192.168.1.10'为IP 地址
RemotePort	UINT	目标端口号	1-65535
Size	UINT	要发送数据的长度	1-1472字节
BufferAddr	DWORD	内存地址	要发送数据的地址
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	指令完成标志	0：未完成 1：完成
Busy	BOOL	指令忙标志	0：指令未指令 1：指令正在执行
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x02：发送数据长度错误 0x04：内存地址错误 0x81：Socket错误 0x83：UDP未创建

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**NET_FreeUDP_Send_0**		TianjinSange.NET_FreeUDP_Send			
VAR	**VARBOOL4**		BOOL			
VAR	**ARRAY1**		ARRAY [0..3] OF BYTE			
VAR	**VAR2**		DWORD			
VAR	**VARBOOL5**		BOOL			
VAR	**VARBOOL6**		BOOL			
VAR	**VARBOOL7**		BOOL			
VAR	**VAR3**		BYTE			
VAR	**DATASend**		ARRAY [0..100] OF BYTE			
VAR	**NET_FreeUDP_0**		TianjinSange.NET_FreeUDP			
VAR	**VARBOOL1**		BOOL			
VAR	**VARBOOL2**		BOOL			
VAR	**VARBOOL3**		BOOL			

编程语言	程序
梯形图（LD2）	

结构化文本（ST）	NET_FreeUDP_Send_0(Execute:=VARBOOL4, Socket:=1, RemoteIP:=ARRAY1, RemotePort:=5000, Size:=10, BufferAddr:=VAR2, Done=>VARBOOL5, Busy=>VARBOOL6, Error=>VARBOOL7, ErrorID=>VAR3); VAR2:=ADR(DATASend[0]);

在 1 个程序中最多可以有 6 个实例。

3.25.10 NET_FreeUDP_Multicast——网口UDP组播

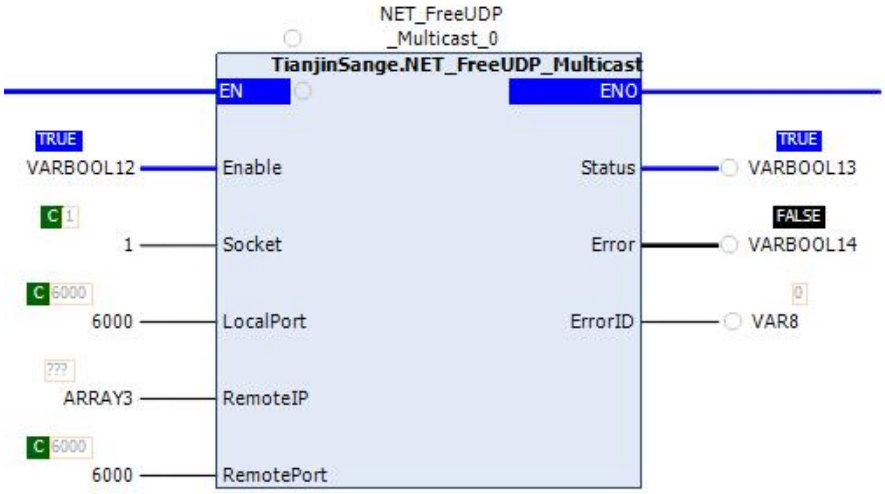
网口UDP组播指令 NET_FreeUDP_Multicast 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：关闭 socket 1：打开 socket（上升沿触发，后续操作保持高电平）
Socket	BYTE	Socket ID	0-5
LocalPort	UINT	本机端口号	1-65535
RemoteIP	ARRAY [0..3] OF BYTE	组播IP地址	数组变量定义，如[192.168.1.10] 中的'192.168.1.10'为IP 地址
RemotePort	UINT	组播口号	1-65535

输出参数	数据类型	功能描述	参数值说明
Status	BOOL	状态标志	0：失能 1：使能
Error	BOOL	错误	0：无错误 1：有错误
ErrorID	BYTE	错误代码	0x81：Socket错误

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**NET_FreeUDP_Multicast_0**		TianjinSange.NET_FreeUDP_Multicast			
VAR	**VARBOOL12**		BOOL			
VAR	**ARRAY3**		ARRAY [0..3] OF BYTE			
VAR	**VARBOOL13**		BOOL			
VAR	**VARBOOL14**		BOOL			
VAR	**VAR8**		BYTE			

编程语言	程序
梯形图 （LD2）	

结构化文本（ST）	NET_FreeUDP_Multicast_0(Enable:=VARBOOL12, Socket:=1, LocalPort:=6000, RemoteIP:=ARRAY3, RemotePort:=6000, Status=>VARBOOL13, Error=>VARBOOL14, ErrorID=>VAR8,);

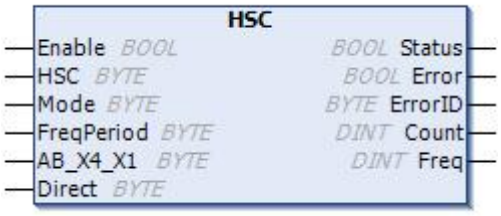
在 1 个程序中最多可以有 6 个实例。

3.26 高速计数器HSC（Tianjin Sange Electr. Tech. Bronze100）

计数模式及引脚分配见《应用手册》。

3.26.1 HSC——高速计数器

高速计数器指令 HSC 在库文件中的图示表达和相关参数说明如下：



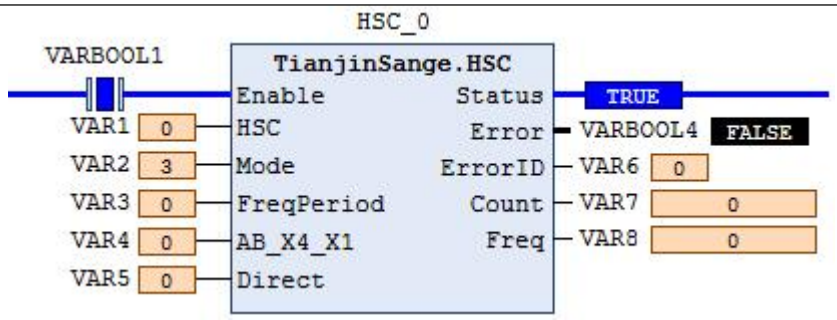
输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：失能计数器 1：使能计数器
HSC	BYTE	HSC ID	0-3
Mode	BYTE	计数器模式	0、1、3、4、6、7、9、10，见以上模式截图
FreqPeriod	BYTE	频率测量周期	0：1000ms 1：100ms 2：10ms
AB_X4_X1	BYTE	AB相计数×4	0：×4

		或×1	1：×1
Direct	BYTE	计数方向	0：向下计数 1：向上计数
输出参数	**数据类型**	**功能描述**	**参数值说明**
Status	BOOL	计数器状态	0：计数器未使能 1：计数器使能
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	BYTE	错误ID	0x01：HSC ID错误 0x02：模式错误 0x04：频率测量周期错误 0x08：AB相计数×4/×1错误 0x10：内部复位错误 0x20：计数方向错误
Count	DINT	计数值	计数值
Freq	DINT	当前频率	当前频率

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**HSC_0**		TianjinSange.HSC			
VAR	**VARBOOL1**		BOOL			
VAR	**VAR1**		BYTE			
VAR	**VAR2**		BYTE			
VAR	**VAR3**		BYTE			
VAR	**VAR4**		BYTE			
VAR	**VARBOOL2**		BOOL			
VAR	**VAR5**		BYTE			
VAR	**VARBOOL3**		BOOL			
VAR	**VARBOOL4**		BOOL			
VAR	**VAR6**		BYTE			
VAR	**VAR7**		DINT			
VAR	**VAR8**		DINT			

编程语言	程序

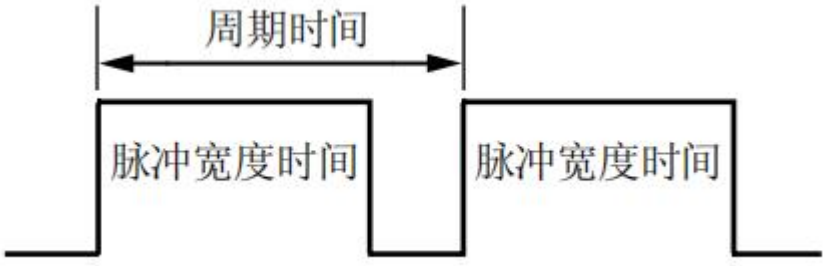
梯形图（LD）	
结构化文本（ST）	HSC_0(Enable:=VARBOOL1, HSC:=VAR1, Mode:=VAR2, FreqPeriod:=VAR3, AB_X4_X1:=VAR4, Direct:=VAR5, Status=>VARBOOL3, Error=>VARBOOL4, ErrorID=>VAR6, Count=>var7, Freq=>var8);

3.27 高速输出（Tianjin Sange Electr. Tech. Bronze100）

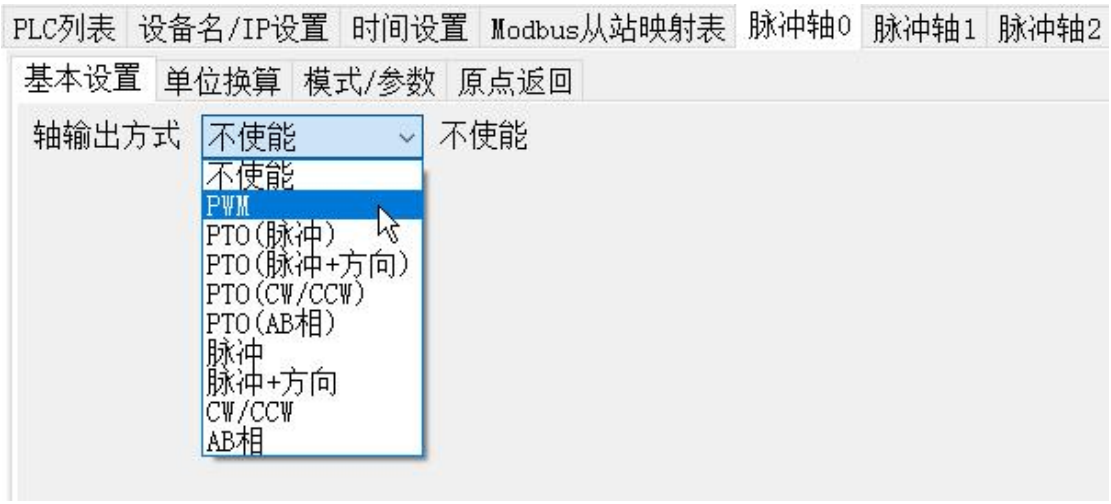
输出模式及引脚分配见《应用手册》。

3.27.1 PWM——PWM输出

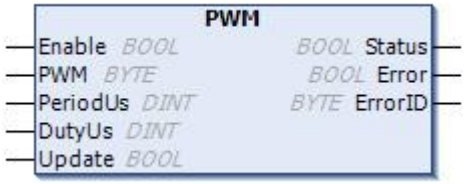
PWM 功能实现周期时间固定、占空比可变的输出，周期时间和脉冲宽度以微秒或毫秒为增量进行指定。当脉冲持续时间等于循环时间，负载循环为 100%，该输出持续打开。当脉冲持续时间为 0，负载循环为 0%，该输出关闭。



使用PWM输出之前需要设置脉冲轴为PWM模式：



PWM输出指令 PWM 在库文件中的图示表达和相关参数说明如下：

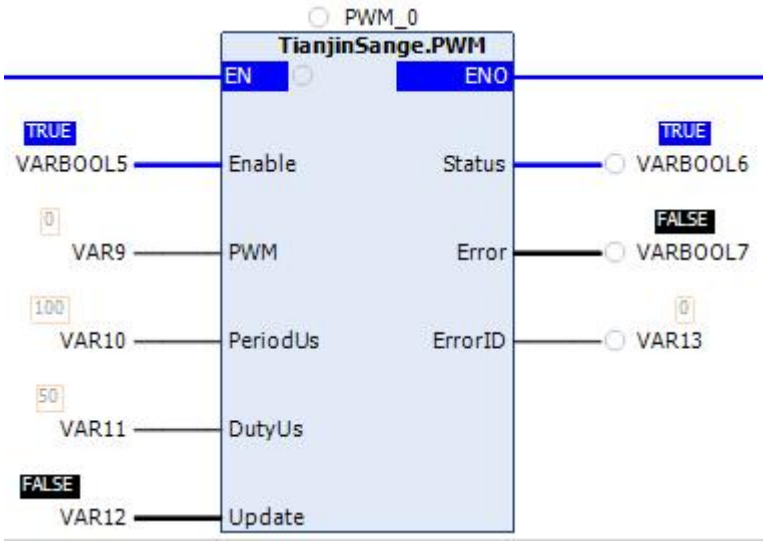


输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	PWM使能	0：停止PWM输出 1：使能PWM输出
PWM	BYTE	PWM ID	0-2
PeriodUs	DINE	PWM周期	10-10000000，单位us
DutyUs	DINE	PWM高电平时间	10-10000000，单位us
Update	BOOL	PWM触发更新	上升沿有效，更新PWM周期和高电平时间
输出参数	**数据类型**	**功能描述**	**参数值说明**
Status	BOOL	PWM输出状态	0：PWM停止输出 1：PWM正在输出
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	BYTE	错误ID	0x01：PWM ID错误

			0x02：高速输出没配置为PWM模式 0x04：PWM周期错误 0x08：PWM高电平时间错误 0x10：PWM高电平时间大于周期时间

变量定义

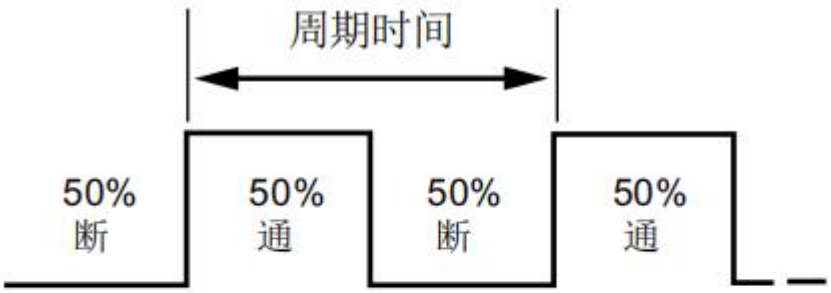
类别	名称	地址	数据类型	初值	注释	属性
VAR	**PWM_0**		TianjinSange.PWM			
VAR	**VARBOOL5**		BOOL			
VAR	**VAR9**		BYTE			
VAR	**VAR10**		DINT			
VAR	**VAR11**		DINT			
VAR	**VAR12**		BOOL			
VAR	**VARBOOL6**		BOOL			
VAR	**VARBOOL7**		BOOL			
VAR	**VAR13**		BYTE			

编程语言	程序
梯形图（LD2）	

结构化文本（ST）	PWM_0(Enable:=VARBOOL5, PWM:=VAR9, PeriodUs:=VAR10, DutyUs:=VAR11, Update:=VAR12, Status=>VARBOOL6, Error=>VARBOOL7, ErrorID=>VAR13);

3.27.2 PTO_SingleSegment——单段脉冲串输出

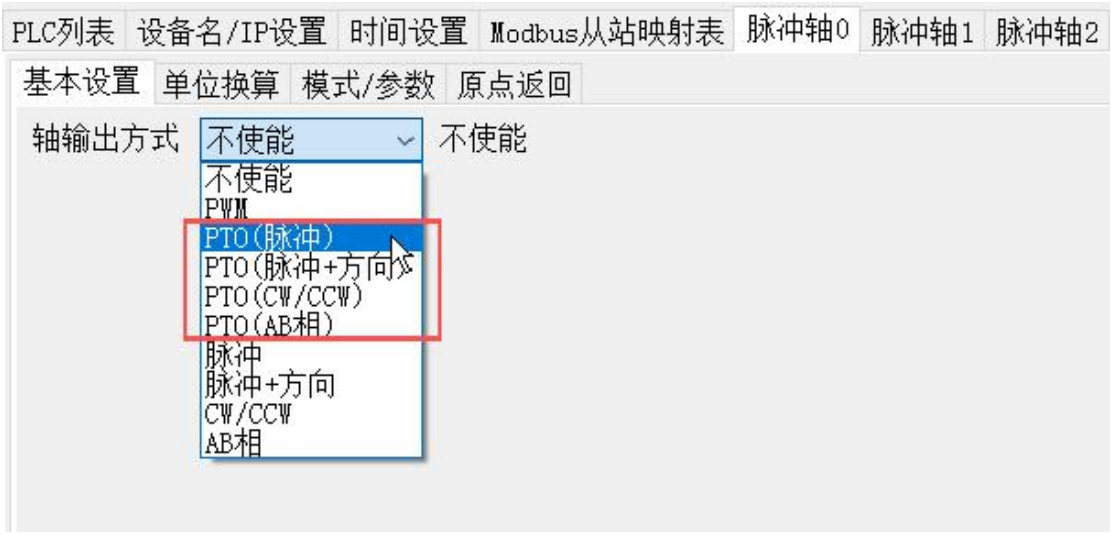
PTO 函数以指定脉冲数（从-2,147,483,648 到 2,147,483,647 个脉冲）和指定频率 (Hz) 提供一个方波 （50% 负载循环）输出。可编写 PTO 函数以产生一个脉冲串或包含多个脉冲串的一个脉冲包络。例如，可使用一个脉冲包络通过一个简单的斜升、运行和斜降顺序或更复杂的顺序控制步进电机。



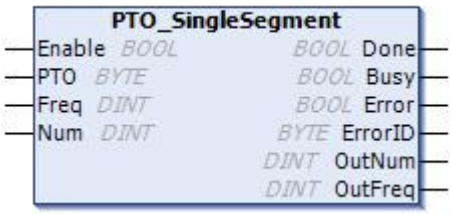
使用以下公式将周期时间转换为频率：

$$F = 1 / CT$$

使用PTO输出之前需要设置脉冲轴为PTO模式：



单段脉冲串输出指令 PTO_SingleSegment 在库文件中的图示表达和相关参数说明如下：

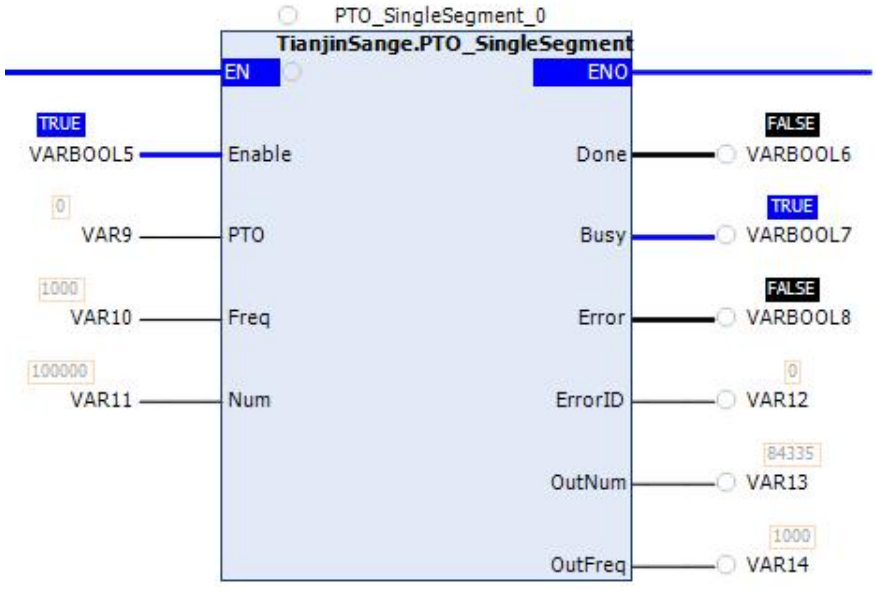


输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能PTO输出	0：停止PTO输出 1：使能PTO输出
PTO	BYTE	PTO ID	0-2
Freq	DINT	输出脉冲频率	1-100000
Num	DINT	输出脉冲个数	-2,147,483,648 到 2,147,483,647
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	输出完成标志	0：未输出完成 1：输出完成
Busy	BOOL	正在输出标志	0：没在输出 1：正在输出
Error	BOOL	错误标志	0：没有错误 1：有错误

ErrorID	BYTE	错误ID	0x01：PTO ID错误 0x02：高速输出没配置为PTO模式 0x04：脉冲频率输入错误 0x08：脉冲个数输入错误
OutNum	DINT	输出个数	已输出脉冲个数
OutFreq	DINT	当前输出频率	当前输出频率

变量定义

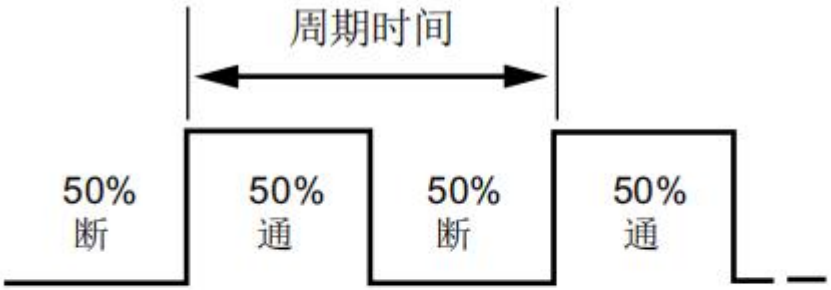
类别	名称	地址	数据类型	初值	注释	属性
VAR	**PTO_SingleSegment_0**		TianjinSange.PTO_SingleSegment			
VAR	**VARBOOL5**		BOOL			
VAR	**VAR9**		BYTE			
VAR	**VAR10**		DINT			
VAR	**VAR11**		DINT			
VAR	**VARBOOL6**		BOOL			
VAR	**VARBOOL7**		BOOL			
VAR	**VARBOOL8**		BOOL			
VAR	**VAR12**		BYTE			
VAR	**VAR13**		DINT			
VAR	**VAR14**		DINT			

编程语言	程序
梯形图（LD2）	

结构化文本（ST）	PTO_SingleSegment_0(Enable:=VARBOOL5, PTO:=VAR9, Freq:=VAR10, Num:=VAR11, Done=>VARBOOL6, Busy=>VARBOOL7, Error=>VARBOOL8, ErrorID=>VAR12, OutNum=>VAR13, OutFreq=>VAR14);

3.27.3 PTO_MultiSegment——多段脉冲串输出

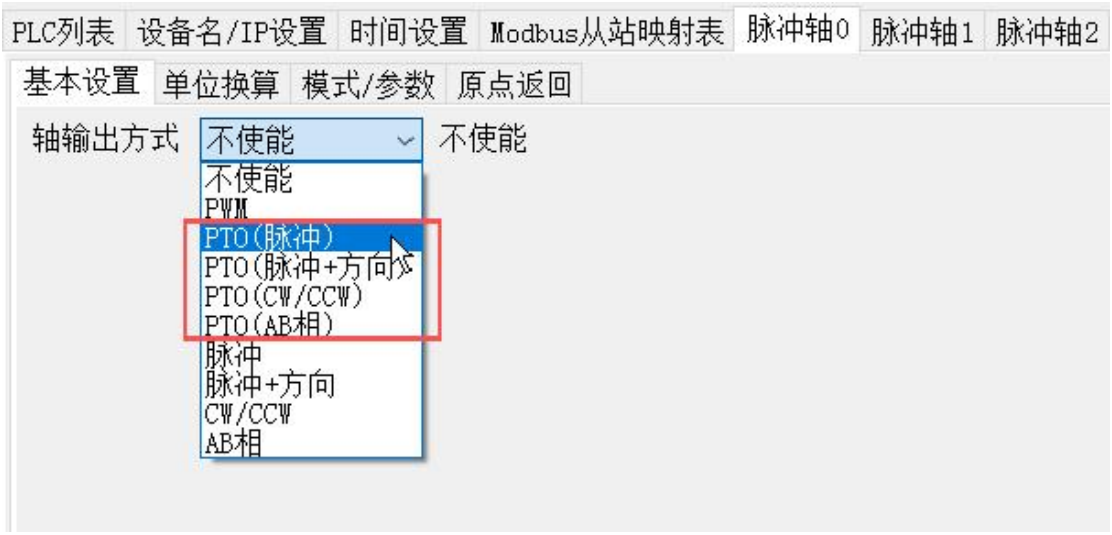
PTO 函数以指定脉冲数（从-2,147,483,648 到 2,147,483,647 个脉冲）和指定频率 (Hz) 提供一个方波 （50% 负载循环）输出。可编写 PTO 函数以产生一个脉冲串或包含多个脉冲串的一个脉冲包络。例如，可使用一个脉冲包络通过一个简单的斜升、运行和斜降顺序或更复杂的顺序控制步进电机。



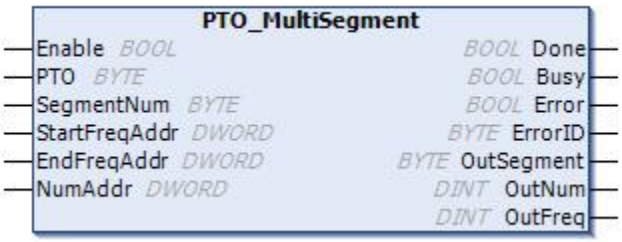
使用以下公式将周期时间转换为频率：

$$F = 1 / CT$$

使用PTO输出之前需要设置脉冲轴为PTO模式：



多段脉冲串输出指令 PTO_MultiSegment 在库文件中的图示表达和相关参数说明如下：

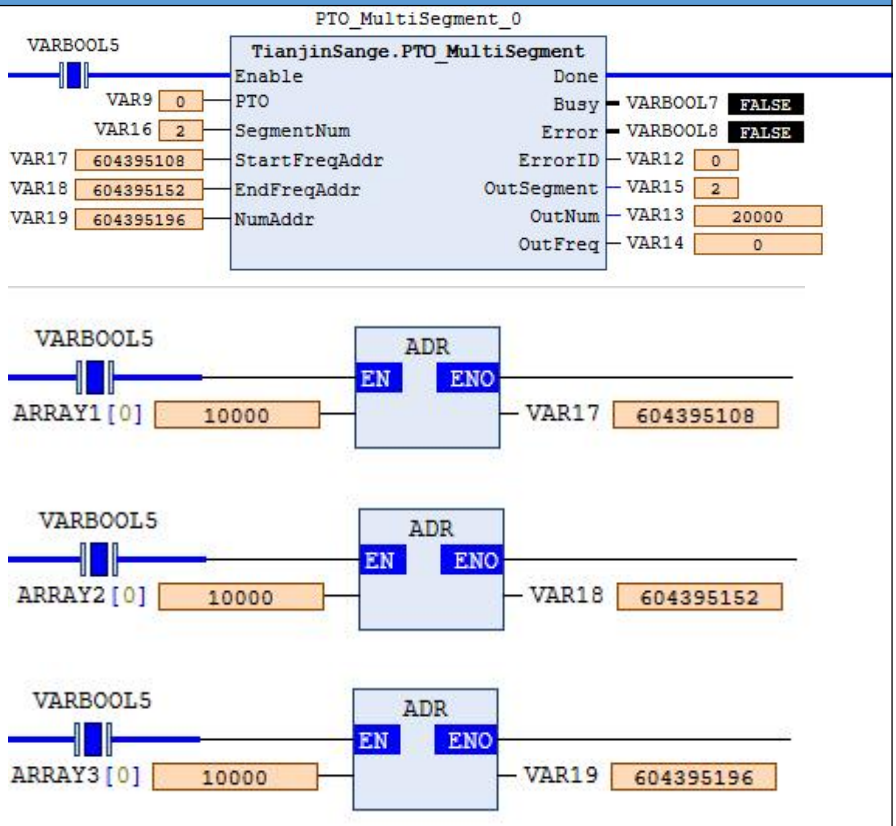


输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能PTO输出	0：停止PTO输出 1：使能PTO输出
PTO	BYTE	PTO ID	0-2
SegmentNum	BYTE	脉冲串段数	1-255
StartFreqAddr	DWORD	起始频率列表地址 ARRAY[..] OF DINT	起始频率列表地址 频率1-100000
EndFreqAddr	DWORD	结束频率列表地址 ARRAY[..] OF DINT	结束频率列表地址 频率1-100000
NumAddr	DWORD	脉冲个数列表地址 ARRAY[..] OF DINT	脉冲个数列表地址 个数 -2,147,483,648 到 2,147,483,647
输出参数	**数据类型**	**功能描述**	**参数值说明**

Done	BOOL	输出完成标志	0：未输出完成 1：输出完成
Busy	BOOL	正在输出标志	0：没在输出 1：正在输出
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	BYTE	错误ID	0x01：PTO ID错误 0x02：高速输出没配置为PTO模式 0x04：脉冲频率输入错误 0x08：脉冲个数输入错误 0x10：段个数错误 0x20：内存地址错误
OutSegment	BYTE	输出段序号	正在输出的脉冲段序号
OutNum	DINT	输出个数	已输出脉冲个数
OutFreq	DINT	当前输出频率	当前输出频率

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**VARBOOL5**		BOOL			
VAR	**VAR9**		BYTE			
VAR	**VAR10**		DINT			
VAR	**VAR11**		DINT			
VAR	**VARBOOL6**		BOOL			
VAR	**VARBOOL7**		BOOL			
VAR	**VARBOOL8**		BOOL			
VAR	**VAR12**		BYTE			
VAR	**VAR13**		DINT			
VAR	**VAR14**		DINT			
VAR	**PTO_MultiSegment_0**		TianjinSange.PTO_MultiSegment			
VAR	**VAR15**		BYTE			
VAR	**VAR16**		BYTE			
VAR	**ARRAY1**		ARRAY[0..9] OF DINT			
VAR	**ARRAY2**		ARRAY[0..9] OF DINT			
VAR	**ARRAY3**		ARRAY[0..9] OF DINT			
VAR	**VAR17**		DWORD			
VAR	**VAR18**		DWORD			
VAR	**VAR19**		DWORD			

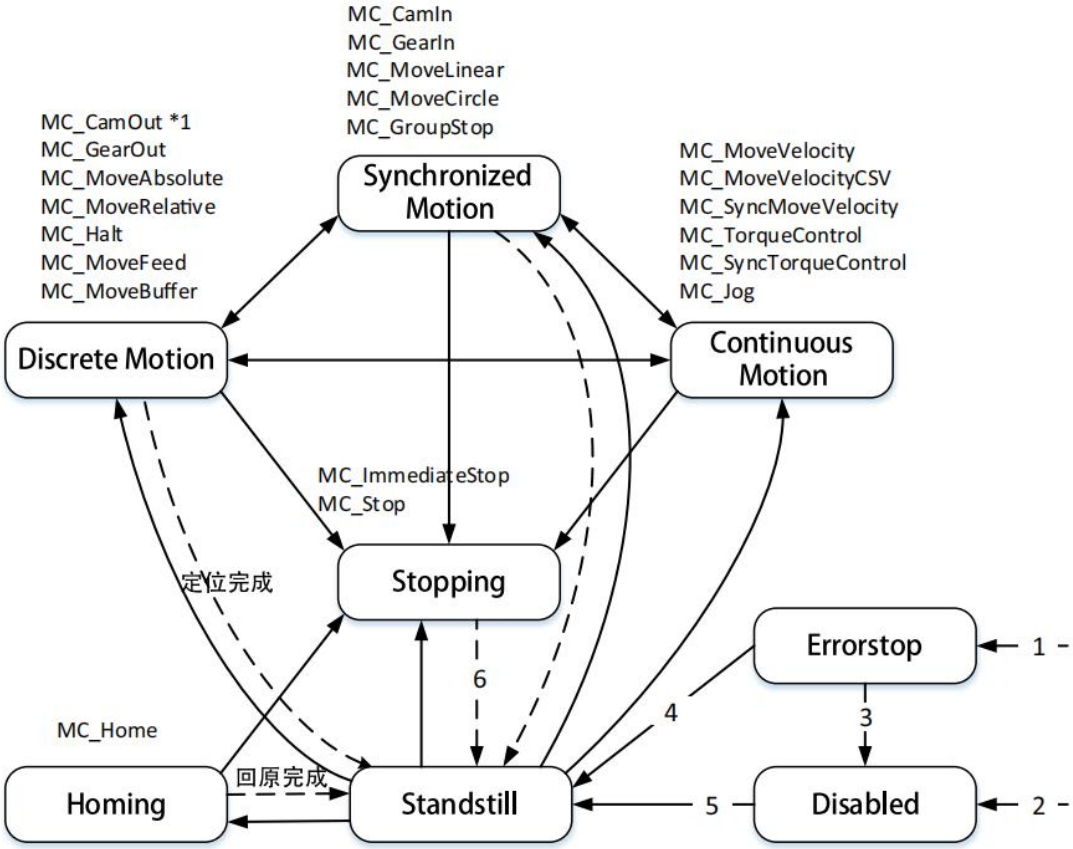
编程语言	程序
梯形图（LD）	

结构化文本（ST）	PTO_MultiSegment_0(Enable:=VARBOOL5, PTO:=VAR9, SegmentNum:=VAR16, StartFreqAddr:=VAR17, EndFreqAddr:=VAR18, NumAddr:=VAR19, Done=>VARBOOL6, Busy=>VARBOOL7, Error=>VARBOOL8, ErrorID=>VAR12, OutSegment=>VAR15, OutNum=>VAR13, OutFreq=>); VAR17:=ADR(ARRAY1[0]); VAR18:=ADR(ARRAY2[0]); VAR19:=ADR(ARRAY3[0]);

3.28 脉冲轴（Tianjin Sange Electr. Tech. Bronze100）

运动控制指令基于 PLCopen® 实现的标准化运动控制功能块。PLCopen®是总部位于欧洲的 IEC 61131-3 推广团体，它是一个全球会员制组织。PLCopen®对运动控制功能块进行了标准化并定义了 IEC 61131-3 (JISB 3503)语言程序接口规范。

基于PLCOpen状态机对轴的状态和运动进行管理。



状态定义

状态值	状态	功能描述
0	Disabled	未使能状态
1	ErrorStop	故障停机状态
2	Stopping	停止状态
3	Standstill	使能状态
4	Discrete Motion	离散运动状态
5	Continuous Motion	连续运行状态
6	Homing	原点回归状态
7	Synchronized Motion	同步运行状态

状态迁移条件

转换	转换条件
1	当轴的故障检测逻辑检测到故障时立即进入该状态
2	当轴无故障且MC_Power.Enable=OFF时

3	当调用MC_Reset复位轴故障且MC_Power.Status=OFF时
4	当调用MC_Reset复位轴故障且MC_Power.Status=ON时
5	当MC_Power.Enable=ON且MC_Power.Status=ON时
6	当MC_Stop(MC_ImmediateStop).Done=ON且 MC_Stop(MC_ImmediateStop).Execute=OFF时

指令中目标位置和目标速度等浮点数采用的是单精度浮点型，因此在PLC程序中处理时指令的值要符合单精度浮点型的范围和精度，即其数值范围为-3.4E38～3.4E38，精度位7位有效数字，如果某个数的有效数字位数超过7位，超出的部分会自动四舍五入。

脉冲轴及引脚分配见《应用手册》。

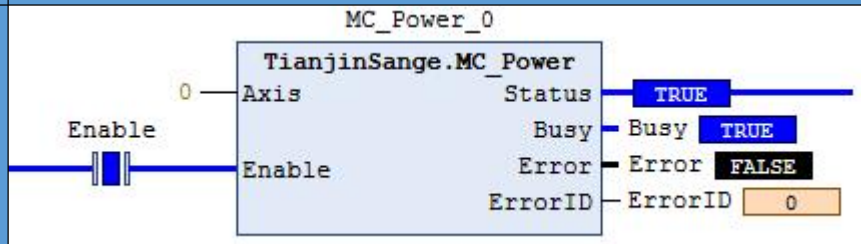
3.28.1 MC_Power——轴使能控制

轴使能控制指令 MC_Power 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Enable	BOOL	使能	0：失能轴 1：使能轴
输出参数	**数据类型**	**功能描述**	**参数值说明**
Status	BOOL	轴使能标志	0：轴失能 1：轴使能
Busy	BOOL	忙标志	0：轴不在工作 1：轴在工作
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**MC_Power_0**		TianjinSange.MC_Power			
2	VAR	**Enable**		BOOL			
3	VAR	**Busy**		BOOL			
4	VAR	**Error**		BOOL			
5	VAR	**ErrorID**		INT			

编程语言	程序
梯形图（LD）	
结构化文本（ST）	MC_Power_0(Axis:=0, Enable:=TRUE, Busy=>Busy, Error=>Error, ErrorID=>ErrorID, Status=>);

指令时序图：

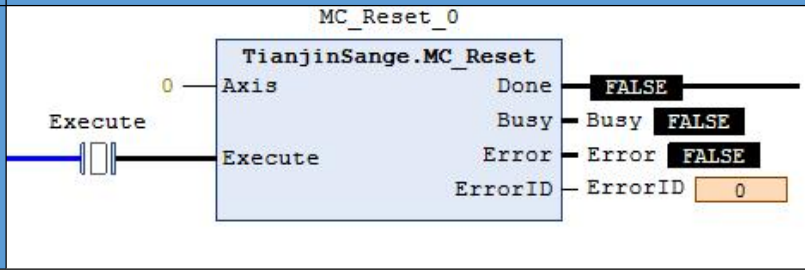
3.28.2 MC_Reset——轴复位故障

轴复位故障指令 MC_Reset 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	触发	上升沿触发
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：未完成复位故障 1：完成复位故障
Busy	BOOL	忙标志	0：不在复位故障中 1：复位故障中
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

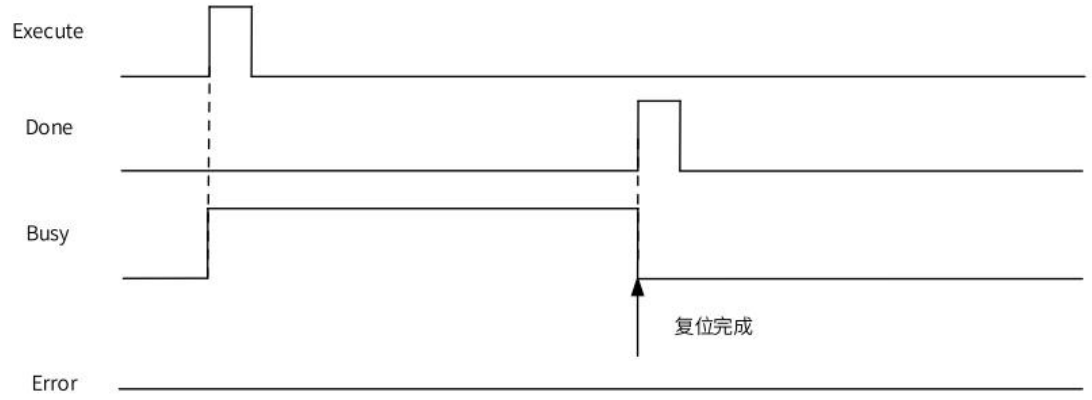
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_Reset_0		TianjinSange.MC_Reset			
2	VAR	Execute		BOOL			
3	VAR	Busy		BOOL			
4	VAR	Error		BOOL			
5	VAR	ErrorID		INT			

编程语言	程序
梯形图（LD）	

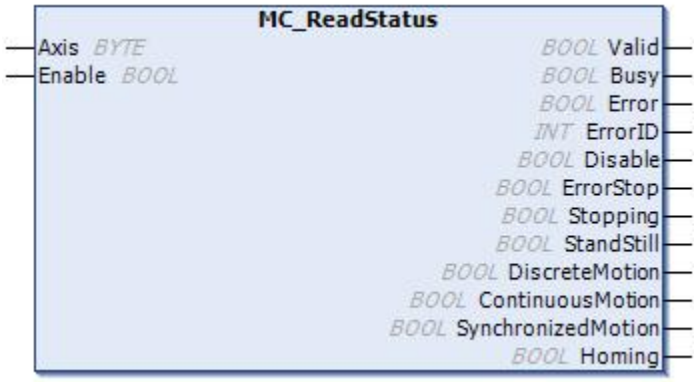
结构化文本（ST）	MC_Reset_0(Axis:=0， Execute:=TRUE, Busy=>Busy, Error=>Error, ErrorID=>ErrorID, Done=>);

指令时序图：



3.28.3 MC_ReadStatus——读取轴状态

读取轴状态指令 MC_ReadStatus 在库文件中的图示表达和相关参数说明如下：



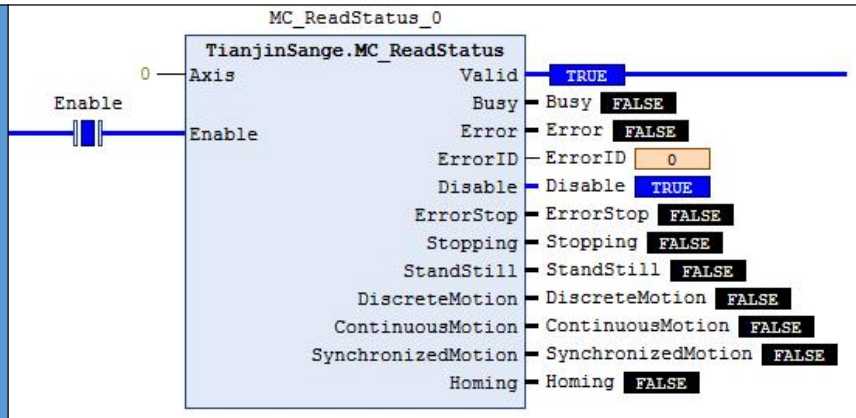
输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Enable	BOOL	使能标志	0：失能读轴状态 1：使能读轴状态

输出参数	数据类型	功能描述	参数值说明
Valid	BOOL	有效标志	0：输出无效 1：输出有效
Busy	BOOL	忙标志	0：不在读取轴状态中 1：读取轴状态中
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表
Disable	BOOL	轴失能	0：轴不处于 该状态 1：轴处于该状态
ErrorStop	BOOL	轴故障	
Stopping	BOOL	轴停止运行	
StandStill	BOOL	轴使能且非运动状态	
DiscreteMotion	BOOL	轴离散运动	
ContinuousMotion	BOOL	轴连续运动	
SynchronizedMotion	BOOL	轴同步运动	
Homing	BOOL	轴原点回归	

变量定义

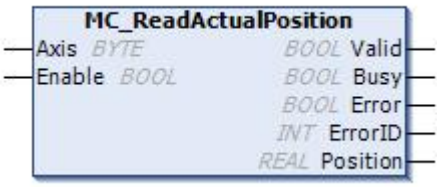
	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_ReadStatus_0		TianjinSange.MC_ReadStatus			
2	VAR	Enable		BOOL			
3	VAR	Busy		BOOL			
4	VAR	Error		BOOL			
5	VAR	ErrorID		INT			
6	VAR	Disable		BOOL			
7	VAR	ErrorStop		BOOL			
8	VAR	Stopping		BOOL			
9	VAR	StandStill		BOOL			
10	VAR	DiscreteMotion		BOOL			
11	VAR	ContinuousMotion		BOOL			
12	VAR	SynchronizedMotion		BOOL			
13	VAR	Homing		BOOL			

编程语言	程序

梯形图（LD）	
结构化文本（ST）	MC_ReadStatus_0(Axis:=0, Enable:=TRUE, Busy=>Busy, Error=>Error, ErrorID=>ErrorID, Disable=>Disable, ErrorStop=> ErrorStop, Stopping=>Stopping, StandStill=>StandStill, DiscreteMotion=>DiscreteMotion, ContinuousMotion=>ContinuousMotion, SynchronizedMotion=>SynchronizedMotion, Homing=>Homing);

3.28.4 MC_ReadActualPosition——读取实际位置

读取实际位置指令 MC_ReadActualPosition 在库文件中的图示表达和相关参数说明如下：

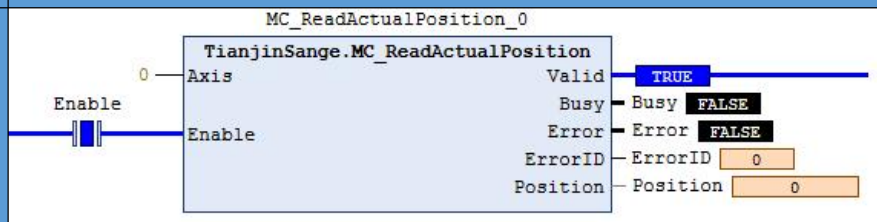


输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2

Enable	BOOL	使能标志	0：失能读实际位置 1：使能读实际位置
输出参数	**数据类型**	**功能描述**	**参数值说明**
Valid	BOOL	有效标志	0：输出无效 1：输出有效
Busy	BOOL	忙标志	0：不在读实际位置中 1：读取实际位置中
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表
Position	REAL	当前位置	轴当前位置

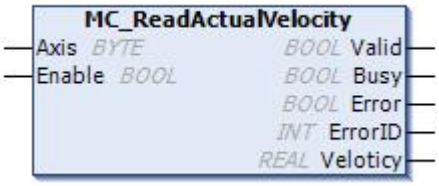
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_ReadActualPosition_0		TianjinSange.MC_ReadActualPosition			
2	VAR	Enable		BOOL			
3	VAR	Busy		BOOL			
4	VAR	Error		BOOL			
5	VAR	ErrorID		INT			
6	VAR	Position		REAL			

编程语言	**程序**
梯形图 （LD）	
结构化文本 （ST）	MC_ReadActualPosition_0(Axis:=0, Enable:=TRUE, Busy=>Busy, Error=>Error, ErrorID=>ErrorID, Position=>Position);

3.28.5 MC_ReadActualVelocity——读取实际速度

读取实际速度指令 MC_ReadActualVelocity 在库文件中的图示表达和相关参数说明如下：

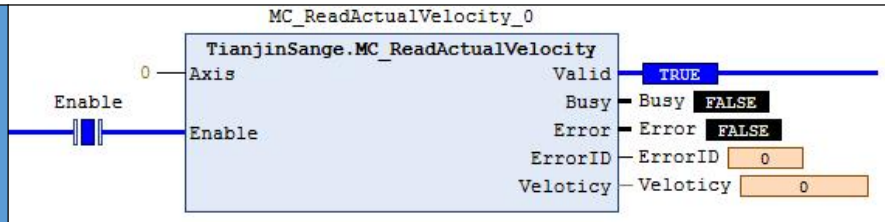


输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Enable	BOOL	使能标志	0：失能读实际速度 1：使能读实际速度
输出参数	**数据类型**	**功能描述**	**参数值说明**
Valid	BOOL	有效标志	0：输出无效 1：输出有效
Busy	BOOL	忙标志	0：不在读实际速度中 1：读取实际速度中
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表
Velocity	REAL	当前位置	轴当前速度

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_ReadActualVelocity_0		TianjinSange.MC_ReadActualVelocity			
2	VAR	Enable		BOOL			
3	VAR	Busy		BOOL			
4	VAR	Error		BOOL			
5	VAR	ErrorID		INT			
6	VAR	Velotiy		REAL			

编程语言	程序

梯形图（LD）	
结构化文本（ST）	MC_ReadActualVelocity_0(Axis:=0, Enable:=TRUE, Busy=>Busy, Error=>Error, ErrorID=>ErrorID, Velocity=>Velocity);

3.28.6 MC_SetPosition——设置当前位置

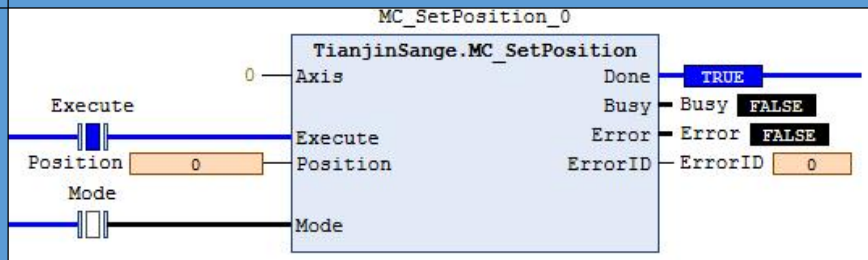
设置当前位置指令 MC_SetPosition 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	使能标志	0：失能读实际速度 1：使能读实际速度
Position	REAL	目标位置	正数/负数/0
Mode	BOOL	绝对/相对位置	0：绝对位置 1：相对位置
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：设置位置未完成 1：设置位置完成
Busy	BOOL	忙标志	0：不忙 1：在忙

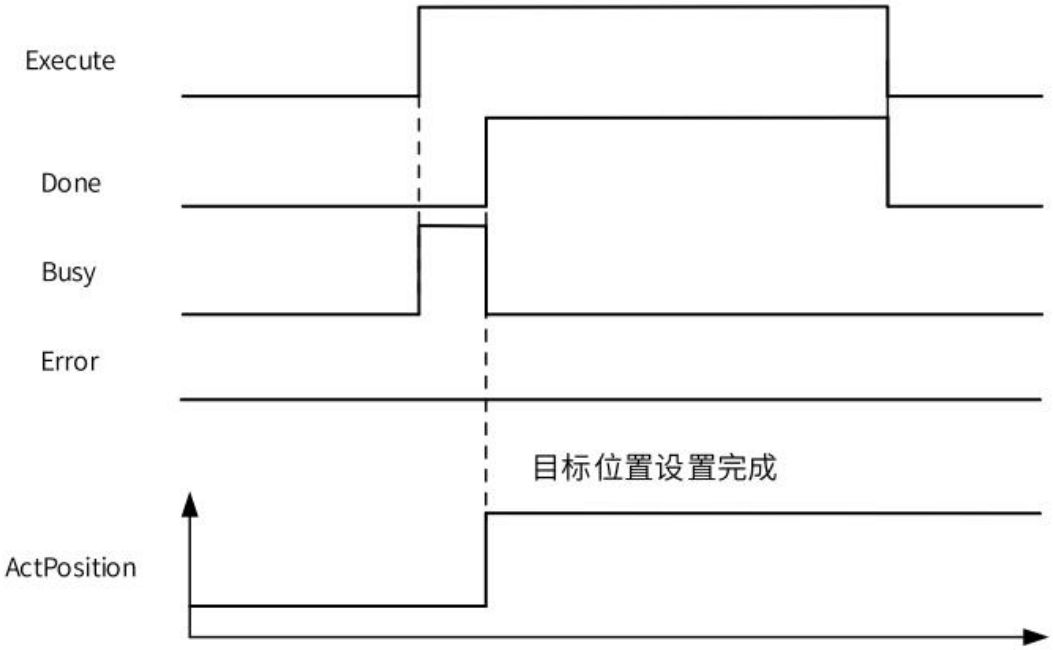
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_SetPosition_0		TianjinSange.MC_SetPosition			
2	VAR	Execute		BOOL			
3	VAR	Position		REAL			
4	VAR	Mode		BOOL			
5	VAR	Busy		BOOL			
6	VAR	Error		BOOL			
7	VAR	ErrorID		INT			

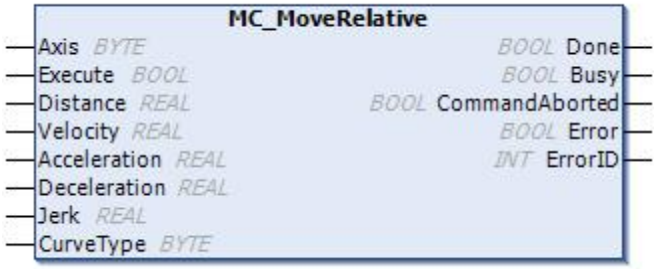
编程语言	程序
梯形图（LD）	
结构化文本（ST）	MC_SetPosition_0(Axis:=0, Execute:=TRUE, Position:=Position, Mode:=Mode, Busy=>Busy, Error=>Error, ErrorID=>ErrorID,);

指令时序图：



3.28.7 MC_MoveRelative——相对定位

相对定位指令 MC_MoveRelative 在库文件中的图示表达和相关参数说明如下：

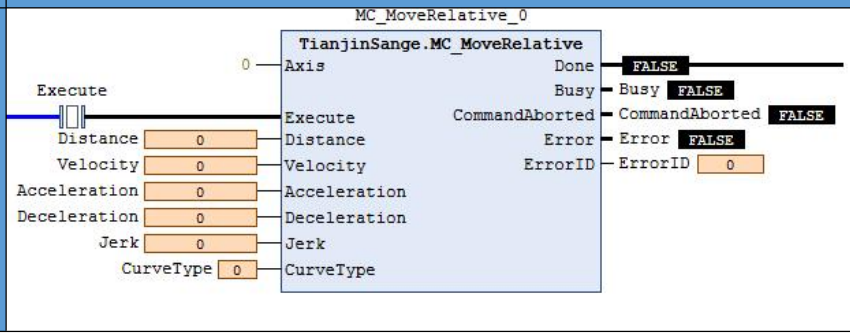


输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	触发	上升沿有效
Distance	REAL	目标位置	正数/负数/0
Velocity	REAL	目标速度	正数
Acceleration	REAL	加速度	正数
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数
CurveType	BYTE	速度曲线类型	0：T型速度曲线

			1：S型速度曲线
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：相对定位未完成 1：相对定位完成
Busy	BOOL	忙标志	0：不忙 1：在忙
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

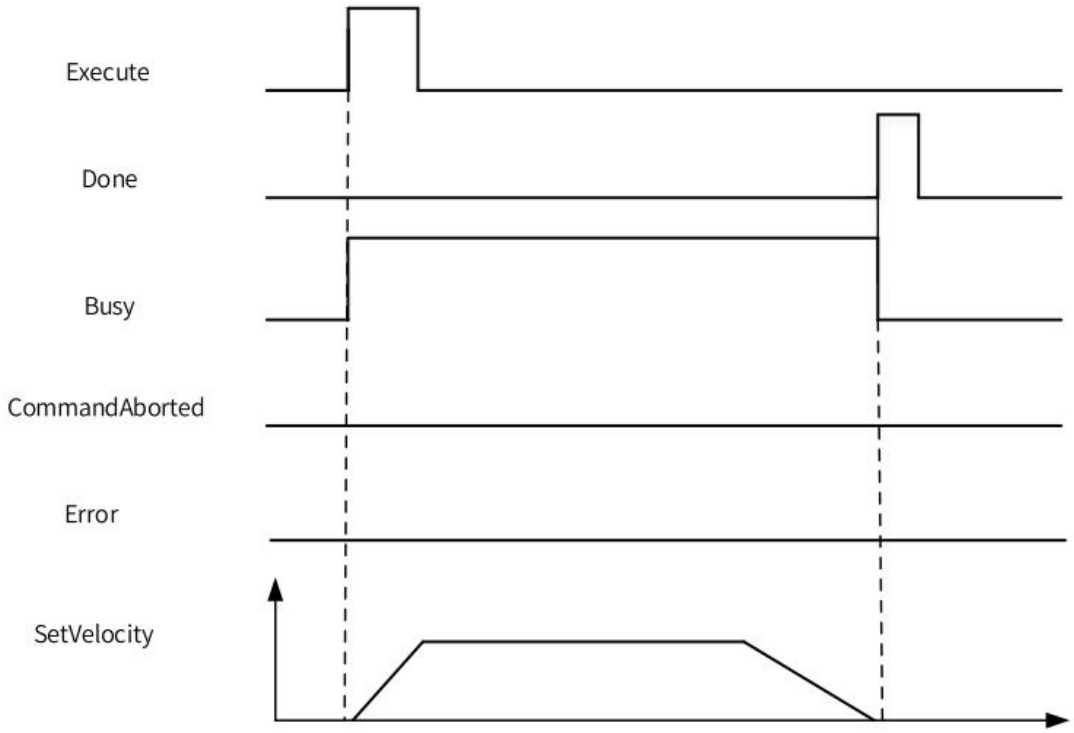
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_MoveRelative_0		TianjinSange.MC_MoveRelative			
2	VAR	Execute		BOOL			
3	VAR	Distance		REAL			
4	VAR	Velocity		REAL			
5	VAR	Acceleration		REAL			
6	VAR	Deceleration		REAL			
7	VAR	Jerk		REAL			
8	VAR	CurveType		BYTE			
9	VAR	Busy		BOOL			
10	VAR	CommandAborted		BOOL			
11	VAR	Error		BOOL			
12	VAR	ErrorID		INT			

编程语言	**程序**
梯形图（LD）	

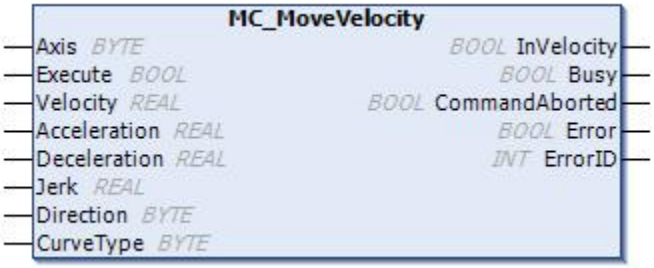
结构化文本（ST）	MC_MoveRelative_0(Axis:=0, Execute:=TRUE, Distance:=Distance, Velocity:=Velocity, Acceleration:=Acceleration, Deceleration:=Deceleration, Jerk:=Jerk, CurveType:=CurveType, Busy=>Busy, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID,);

指令时序图：



3.28.8 MC_MoveVelocity——速度指令

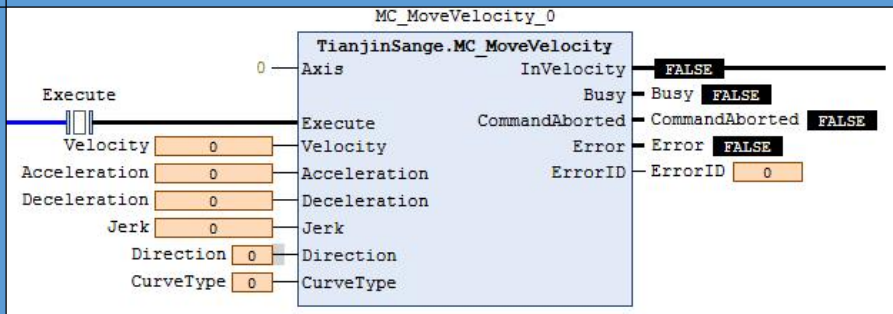
速度指令 MC_MoveVelocity 在库文件中的图示表达和相关参数说明如下：



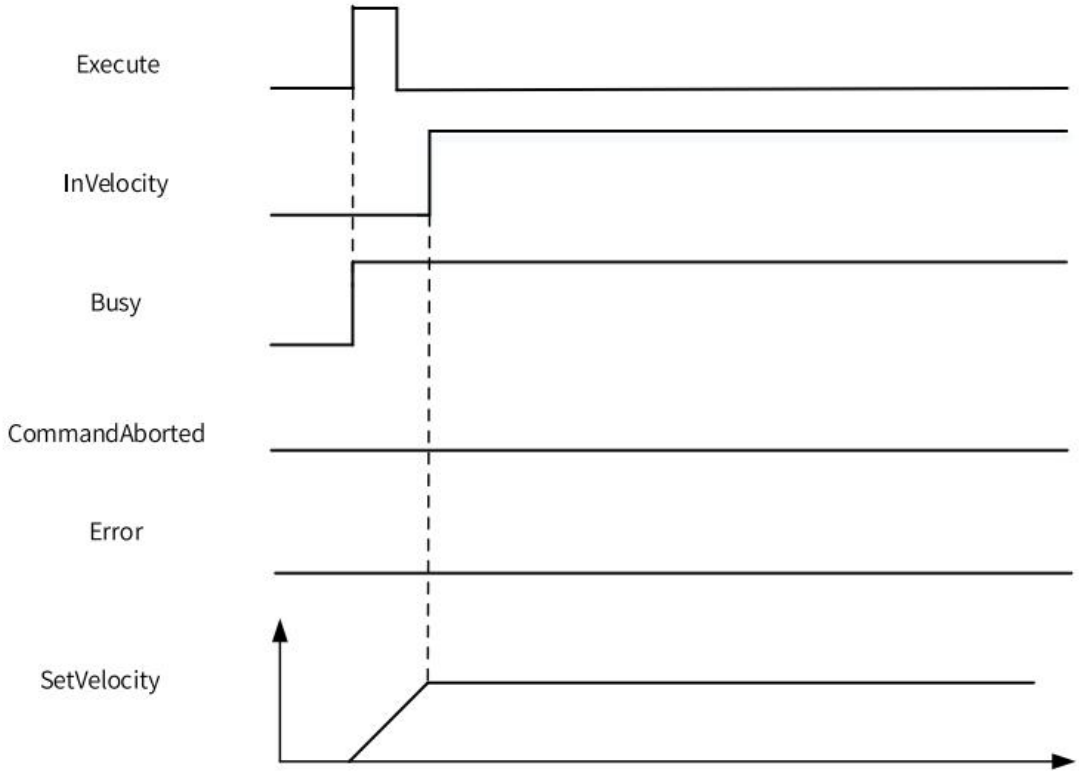
输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	触发	上升沿有效
Velocity	REAL	目标速度	正数
Acceleration	REAL	加速度	正数
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数
Direction	BOOL	方向	0：负向 1：正向 2：当前方向
CurveType	BYTE	速度曲线类型	0：T型速度曲线 1：S型速度曲线
输出参数	**数据类型**	**功能描述**	**参数值说明**
InVelocity	BOOL	速度到达	0：速度未到达 1：速度到达
Busy	BOOL	忙标志	0：不忙 1：在忙
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**MC_MoveVelocity_0**		TianjinSange.MC_MoveVelocity			
2	VAR	**Execute**		BOOL			
3	VAR	**Velocity**		REAL			
4	VAR	**Acceleration**		REAL			
5	VAR	**Deceleration**		REAL			
6	VAR	**Jerk**		REAL			
7	VAR	**Direction**		BYTE			
8	VAR	**CurveType**		BYTE			
9	VAR	**Busy**		BOOL			
10	VAR	**CommandAborted**		BOOL			
11	VAR	**Error**		BOOL			
12	VAR	**ErrorID**		INT			

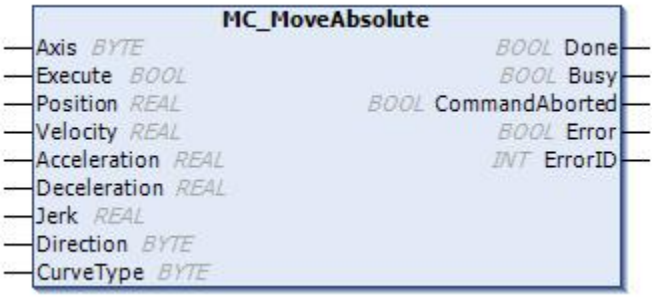
编程语言	程序
梯形图（LD）	
结构化文本（ST）	MC_MoveVelocity_0(Axis:=0, Execute:=Execute, Velocity:=Velocity, Acceleration:=Acceleration, Deceleration:=Deceleration, Jerk:=Jerk, Direction:=Direction, CurveType:=CurveType, Busy=>Busy, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID,);

指令时序图：



3.28.9 MC_MoveAbsolute——绝对定位

绝对定位指令 MC_MoveAbsolute 在库文件中的图示表达和相关参数说明如下：

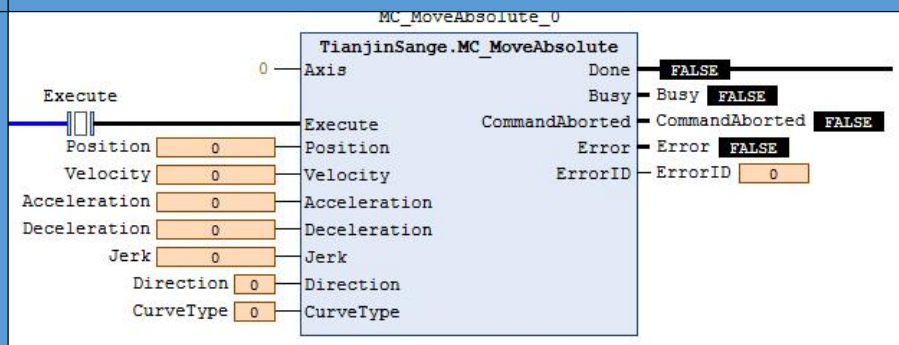


输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	触发	上升沿有效
Position	REAL	目标位置	正数/负数/0
Velocity	REAL	目标速度	正数
Acceleration	REAL	加速度	正数
Deceleration	REAL	减速度	正数

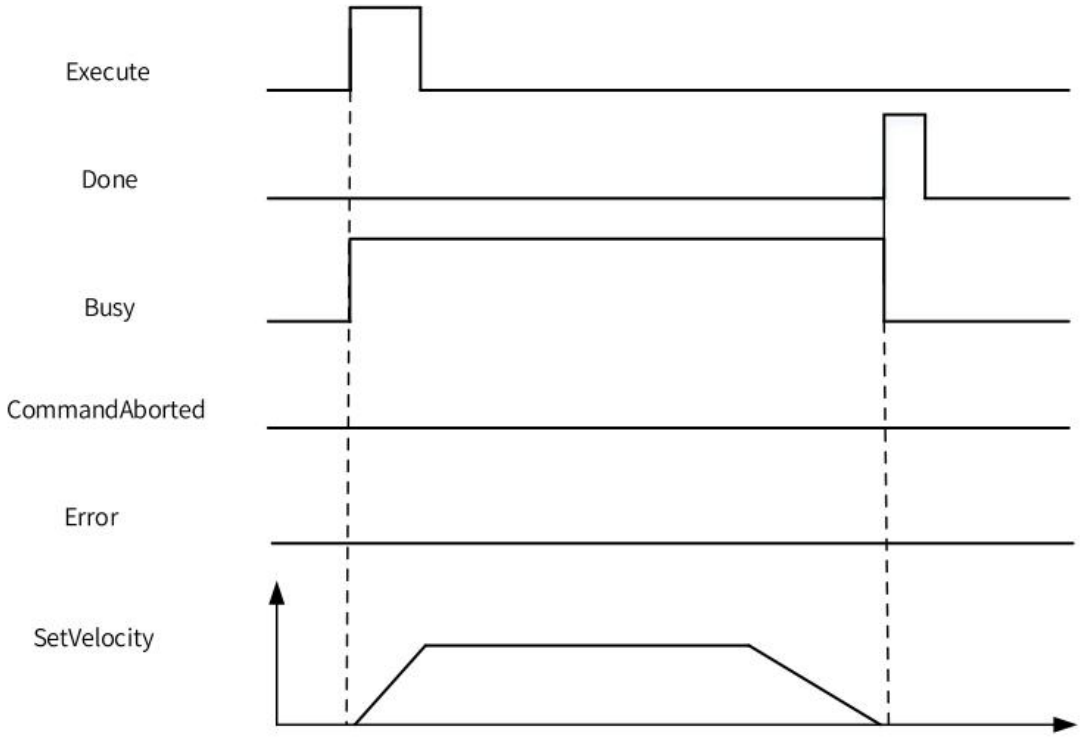
Jerk	REAL	加加速度	正数
Direction	BYTE	方向	仅旋转轴有效 0：负向 1：正向 2：当前方向 3：最短距离
CurveType	BYTE	速度曲线类型	0：T型速度曲线 1：S型速度曲线
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：绝对定位未完成 1：绝对定位完成
Busy	BOOL	忙标志	0：不忙 1：在忙
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_MoveAbsolute_0		TianjinSange.MC_MoveAbsolute			
2	VAR	Execute		BOOL			
3	VAR	Acceleration		REAL			
4	VAR	Deceleration		REAL			
5	VAR	Position		REAL			
6	VAR	Velocity		REAL			
7	VAR	Jerk		REAL			
8	VAR	Direction		BYTE			
9	VAR	CurveType		BYTE			
10	VAR	Busy		BOOL			
11	VAR	CommandAborted		BOOL			
12	VAR	Error		BOOL			
13	VAR	ErrorID		INT			

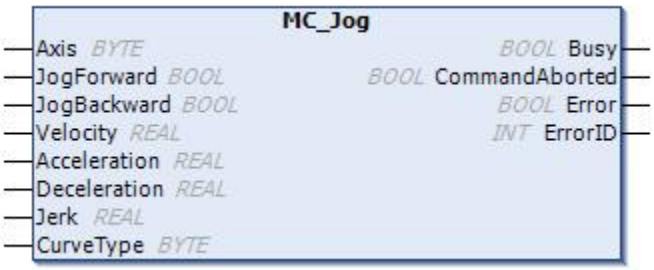
<table>
<tr><th>编程语言</th><th>程序</th></tr>
<tr><td>梯形图
（LD）</td><td> 
</td></tr>
<tr><td>结构化文本
（ST）</td><td>MC_MoveAbsolute_0(
Axis:=0,
Execute:=Execute,
Position:=Position,
Velocity:=Velocity,
Acceleration:=Acceleration,
Deceleration:=Deceleration,
Jerk:=Jerk,
Direction:=Direction,
CurveType:=CurveType,
Busy=>Busy,
CommandAborted=>CommandAborted,
Error=>Error,
ErrorID=>ErrorID,);</td></tr>
</table>

指令时序图：



3.28.10 MC_Jog——点动运动

点动运动指令 MC_Jog 在库文件中的图示表达和相关参数说明如下：

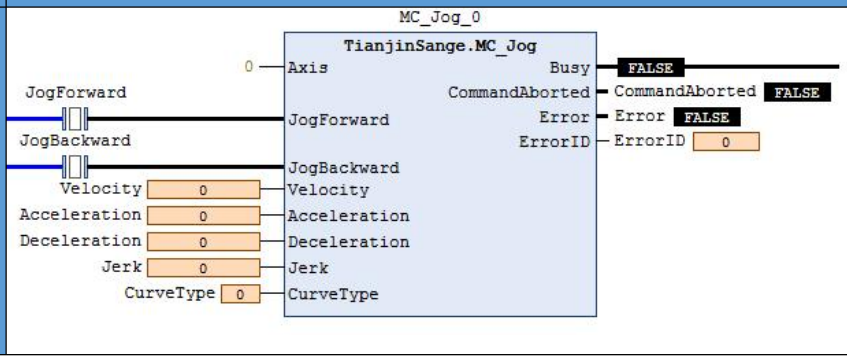


输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
JogForward	BOOL	正向运动	上升沿触发，下降沿减速停止
JogBackward	REAL	负向运动	上升沿触发，下降沿减速停止
Velocity	REAL	目标速度	正数
Acceleration	REAL	加速度	正数
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数
CurveType	BYTE	速度曲线类型	0：T型速度曲线

			1：S型速度曲线
输出参数	**数据类型**	**功能描述**	**参数值说明**
Busy	BOOL	忙标志	0：不忙 1：在忙
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

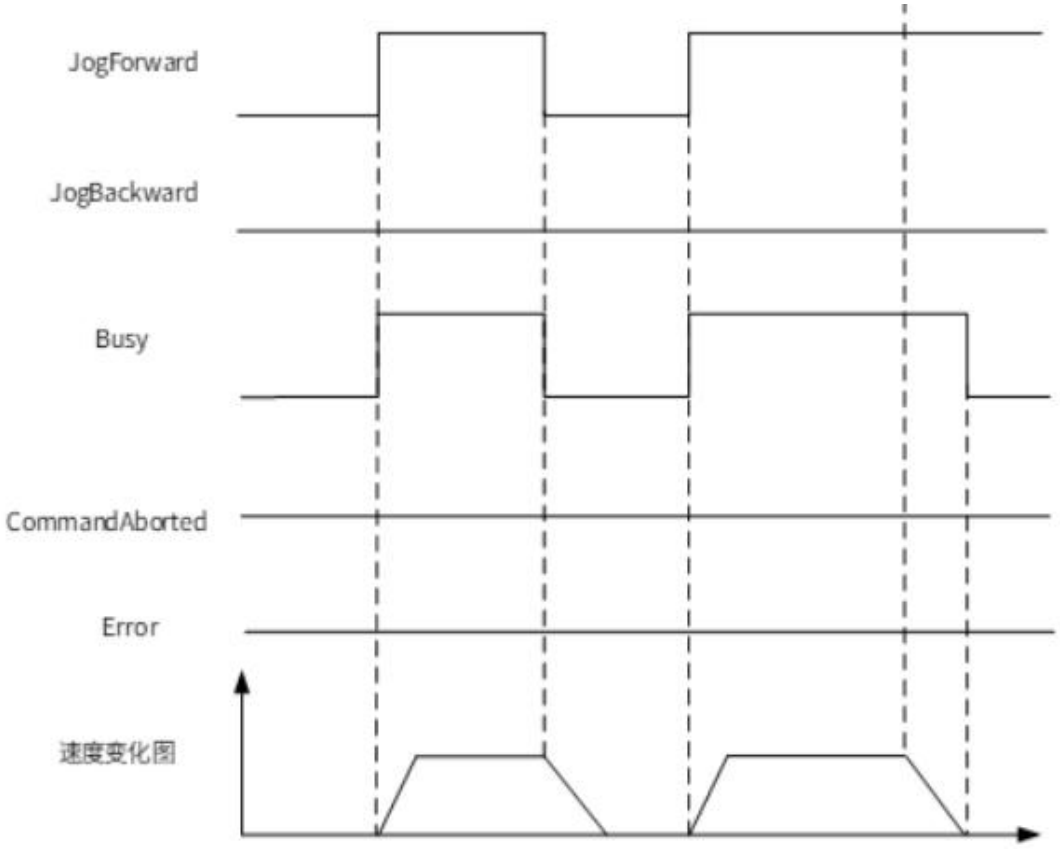
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_Jog_0		TianjinSange.MC_Jog			
2	VAR	JogForward		BOOL			
3	VAR	JogBackward		BOOL			
4	VAR	Velocity		REAL			
5	VAR	Acceleration		REAL			
6	VAR	Deceleration		REAL			
7	VAR	Jerk		REAL			
8	VAR	CurveType		BYTE			
9	VAR	Busy		BOOL			
10	VAR	CommandAborted		BOOL			
11	VAR	Error		BOOL			
12	VAR	ErrorID		INT			

编程语言	**程序**
梯形图（LD）	

结构化文本（ST）	MC_Jog_0(Axis:=0, JogForward:=JogForward, JogBackward:=JogBackward, Velocity:=Velocity, Acceleration:=Acceleration, Deceleration:=Deceleration, Jerk:=Jerk, CurveType:=CurveType, Busy=>Busy, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID,);

指令时序图：



3.28.11 MC_Home——原点回归

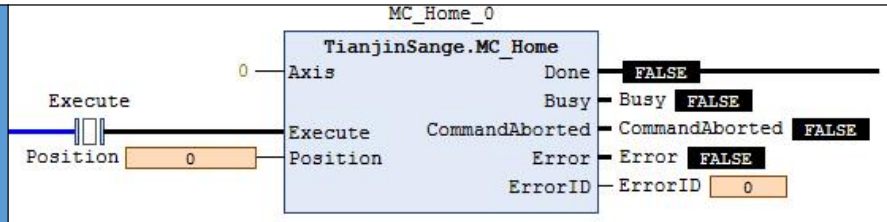
原点回归指令 MC_Home 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	触发	上升沿触发
Position	REAL	原点偏移	正数/负数/0
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：原点回归未完成 1：原点回归完成
Busy	BOOL	忙标志	0：不忙 1：在忙
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

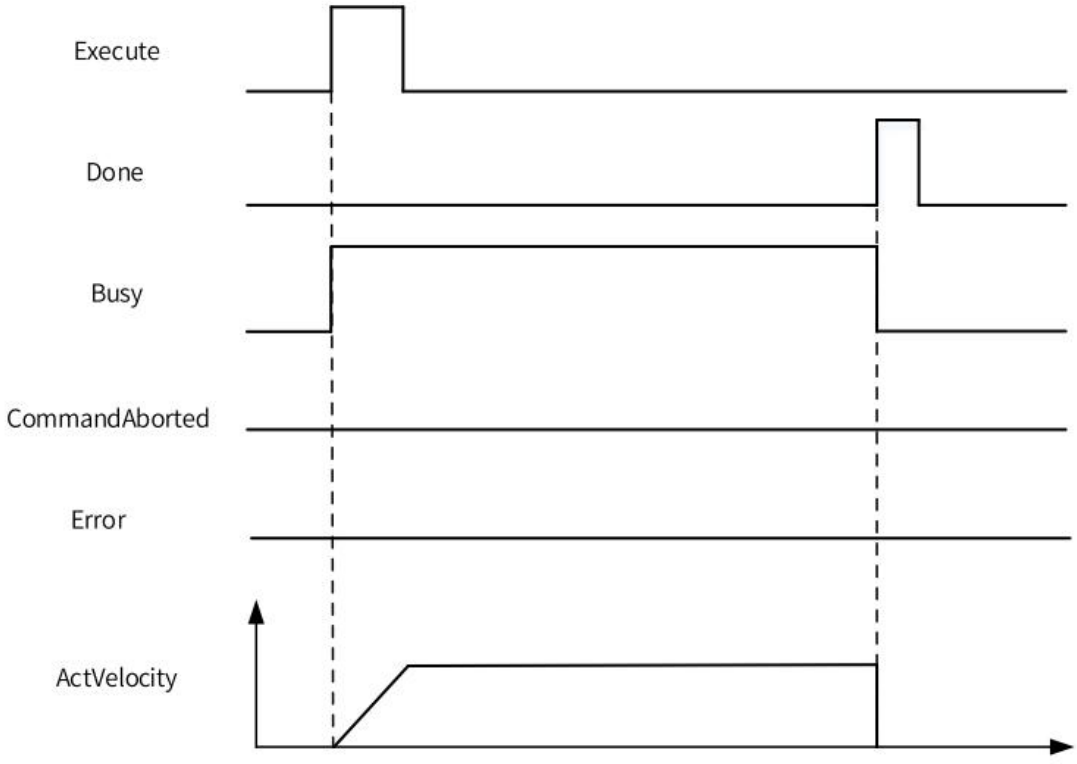
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**MC_Home_0**		TianjinSange.MC_Home			
2	VAR	**Execute**		BOOL			
3	VAR	**Position**		REAL			
4	VAR	**Busy**		BOOL			
5	VAR	**CommandAborted**		BOOL			
6	VAR	**Error**		BOOL			
7	VAR	**ErrorID**		INT			

编程语言	程序

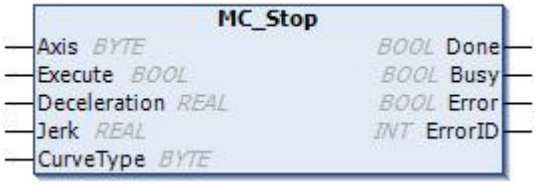
梯形图（LD）	
结构化文本（ST）	MC_Home_0(Axis:=0, Execute:=Execute, Position:=Position, Busy=>Busy, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID,);

指令时序图：



3.28.12 MC_Stop——停止指令

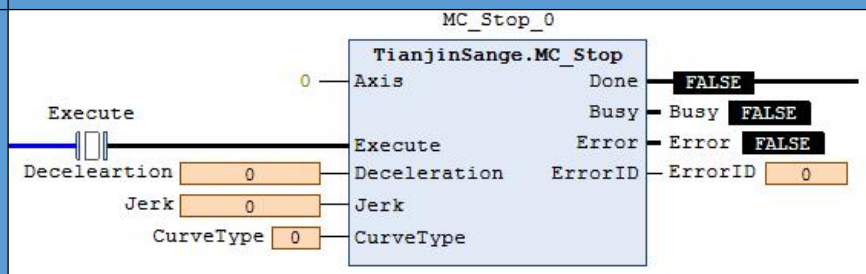
停止指令 MC_Stop 在库文件中的图示表达和相关参数说明如下：



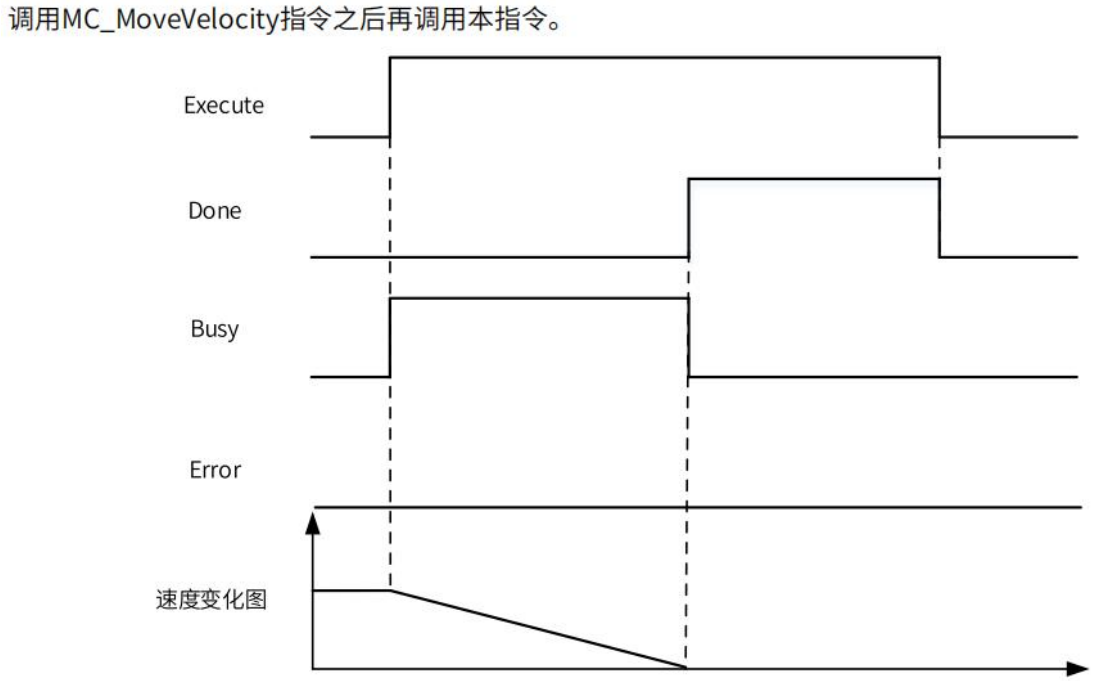
输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	触发	上升沿触发
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数
CurveType	BYTE	速度曲线类型	0：T型速度曲线 1：S型速度曲线
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：停止未完成 1：停止完成
Busy	BOOL	忙标志	0：不忙 1：在忙
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**MC_Stop_0**		TianjinSange.MC_Stop			
2	VAR	**Execute**		BOOL			
3	VAR	**Deceleartion**		REAL			
4	VAR	**Jerk**		REAL			
5	VAR	**CurveType**		BYTE			
6	VAR	**Busy**		BOOL			
7	VAR	**Error**		BOOL			
8	VAR	**ErrorID**		INT			

编程语言	程序
梯形图（LD）	
结构化文本（ST）	MC_Stop_0(Axis:=0, Execute:=Execute, Deceleration:=Deceleration, Jerk:=Jerk, CurveType:=CurveType, Busy=>Busy, Error=>Error, ErrorID=>ErrorID,);

指令时序图：



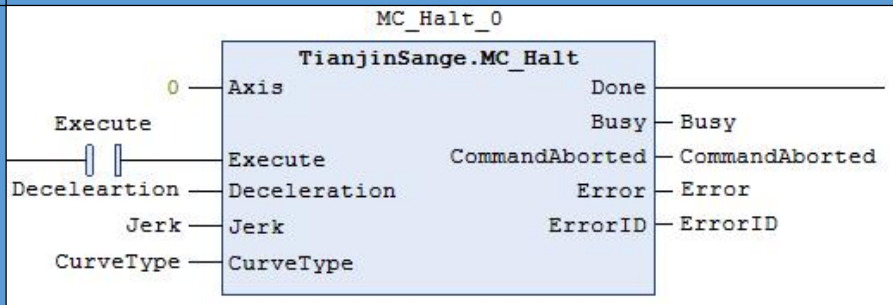
3.28.13 MC_Halt——暂停指令

暂停指令 MC_Halt 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	触发	上升沿触发
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数
CurveType	BYTE	速度曲线类型	0：T型速度曲线 1：S型速度曲线
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：暂停未完成 1：暂停完成
Busy	BOOL	忙标志	0：不忙 1：在忙
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴错误列表

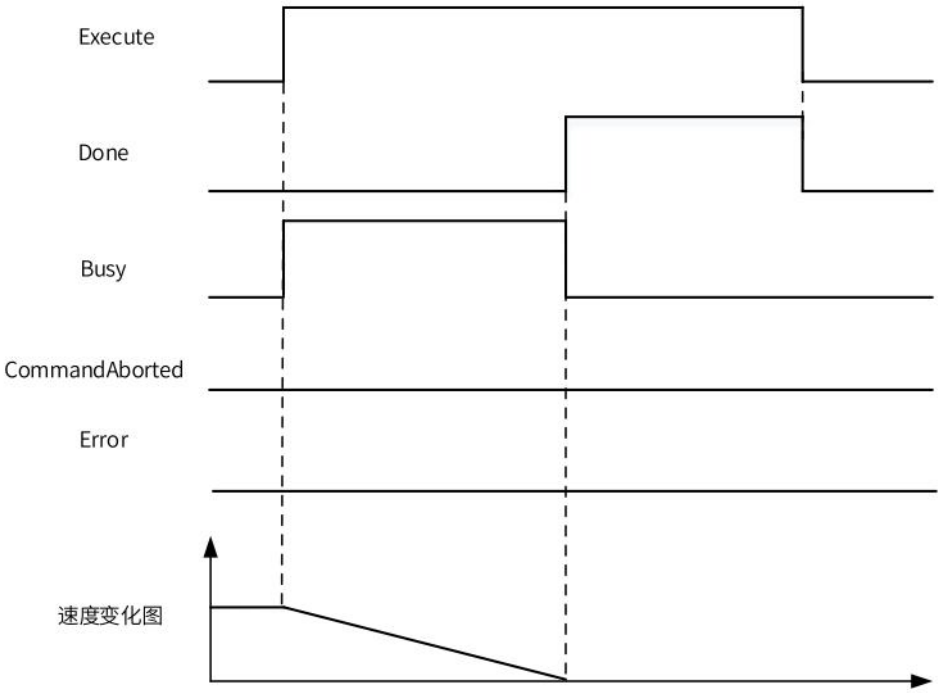
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**MC_Halt_0**		TianjinSange.MC_Halt			
2	VAR	**Execute**		BOOL			
3	VAR	**Deceleartion**		REAL			
4	VAR	**Jerk**		REAL			
5	VAR	**CurveType**		BYTE			
6	VAR	**Busy**		BOOL			
7	VAR	**CommabdAborted**		BOOL			
8	VAR	**Error**		BOOL			
9	VAR	**ErrorID**		INT			

编程语言	程序
梯形图（LD）	
结构化文本（ST）	MC_Halt_0(Axis:=0, Execute:=Execute, Deceleration:=Deceleration, Jerk:=Jerk, CurveType:=CurveType, Busy=>Busy, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID,);

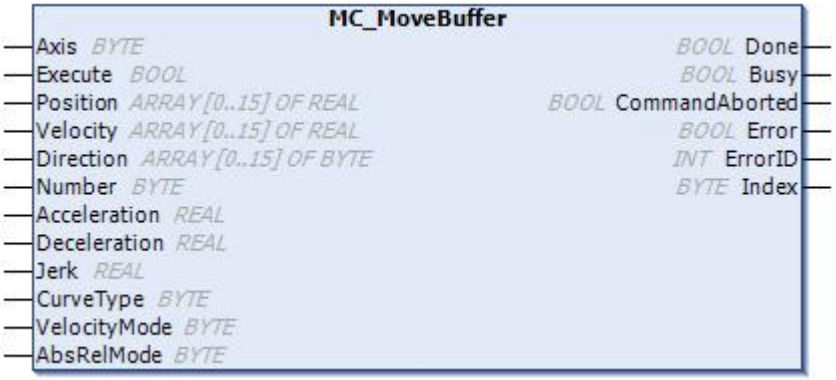
指令时序图：

调用定位之后，触发MC_Halt指令。



3.28.14 MC_MoveBuffer——多段位置定位

多段位置定位指令 MC_MoveBuffer 在库文件中的图示表达和相关参数说明如下：



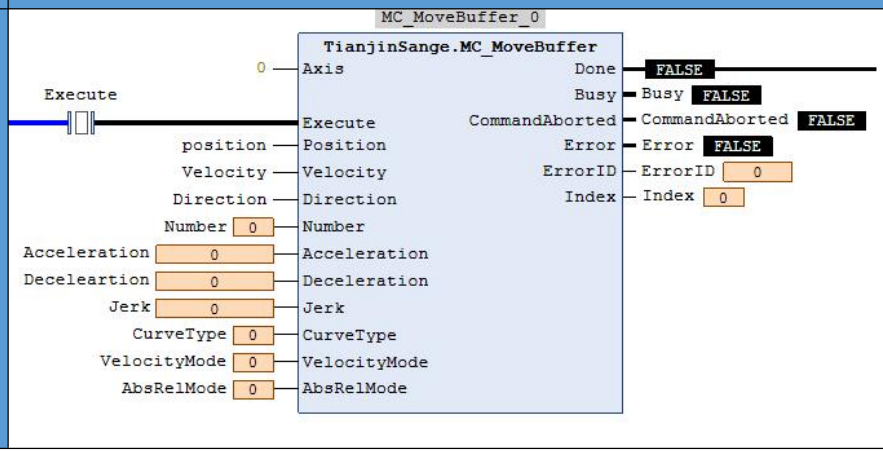
输入参数	数据类型	功能描述	参数值说明
Axis	BYTE	轴ID	0-2
Execute	BOOL	触发	上升沿有效
Position	ARRAY[0..15] OF REAL	目标位置列表	正数/负数/0

Velocity	ARRAY[0..15] OF REAL	目标速度列表	正数
Direction	ARRAY[0..15] OF BYTE	方向列表	仅旋转轴有效 0：负向 1：正向 2：当前方向 3：最短距离
Number	BYTE	段数	1-16
Acceleration	REAL	加速度	正数
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数
CurveType	BYTE	速度曲线类型	0：T型速度曲线 1：S型速度曲线
VelocityMode	BYTE	速度切换模式	0：减速到0然后启动下一段 1：保持当前速度启动下一段
AbsRelMode	BYTE	定位模式	0：绝对定位 1：相对定位
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：绝对定位未完成 1：绝对定位完成
Busy	BOOL	忙标志	0：不忙 1：在忙
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误

ErrorID	INT	错误ID	见后面脉冲轴错误列表
Index	BYTE	当前正在执行的段	当前正在执行的段

变量定义

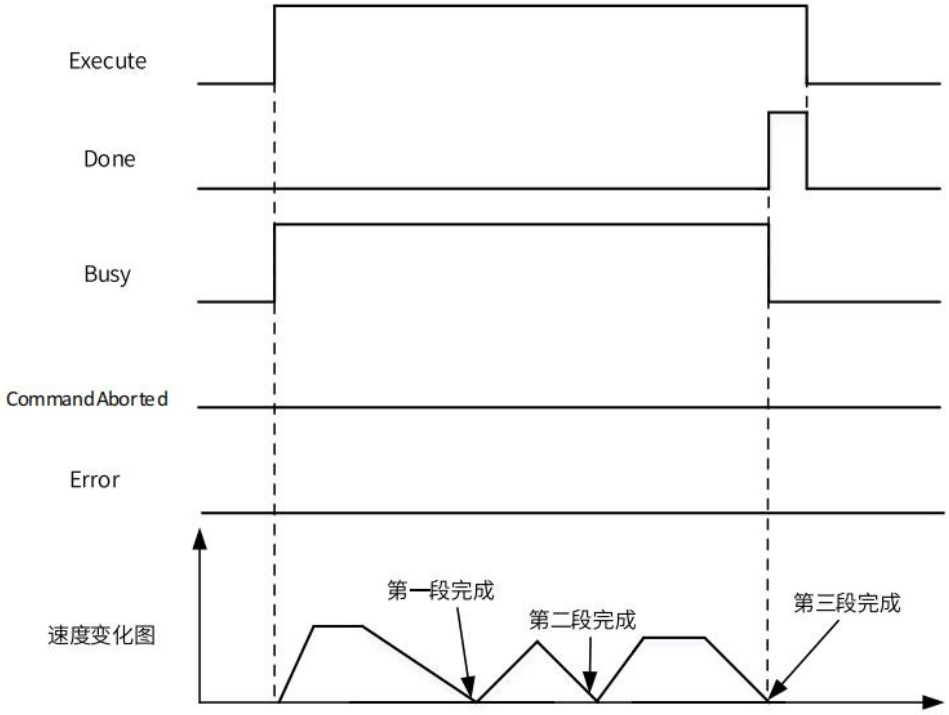
	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_MoveBuffer_0		TianjinSange.MC_MoveBuffer			
2	VAR	Execute		BOOL			
3	VAR	Position		ARRAY[0..15] OF REAL			
4	VAR	Velocity		ARRAY[0..15] OF REAL			
5	VAR	Direction		ARRAY[0..15] OF BYTE			
6	VAR	Number		BYTE			
7	VAR	Acceleration		REAL			
8	VAR	Deceleartion		REAL			
9	VAR	Jerk		REAL			
10	VAR	CurveType		BYTE			
11	VAR	VelocityMode		BYTE			
12	VAR	AbsRelMode		BYTE			
13	VAR	Busy		BOOL			
14	VAR	CommandAborted		BOOL			
15	VAR	Error		BOOL			
16	VAR	ErrorID		INT			
17	VAR	Index		BYTE			

编程语言	程序
梯形图（LD）	

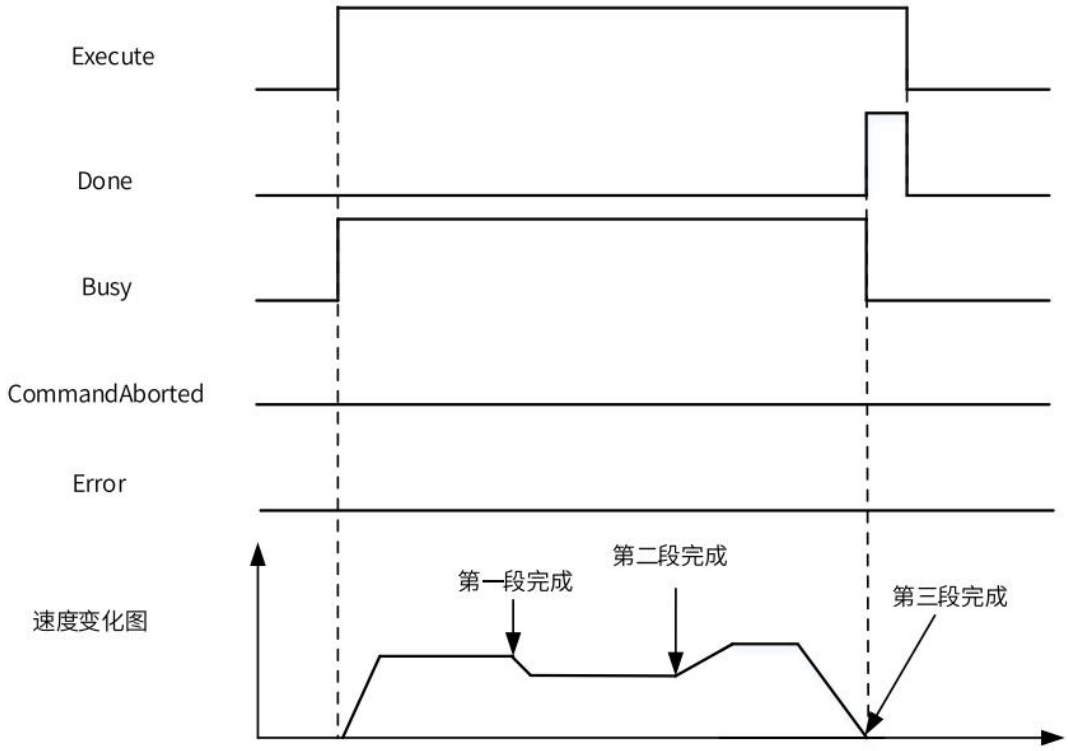
结构化文本（ST）	MC_MoveBuffer_0(Axis:=0, Execute:=Execute, Position:=Position, Velocity:=Velocity, Direction:=Direction, Number:=Number, Acceleration:=Acceleration, Deceleration:=Deceleration, Jerk:=Jerk, CurveType:=CurveType, VelocityMode:=VelocityMode, AbsRelMode:=AbsRelMode, Busy=>Busy, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID, Index=>Index);

指令时序图：

设置为3段缓冲，当VelocityMode = 1时；



设置为3段缓冲，当VelocityMode = 1时；



3.28.15 单轴错误码

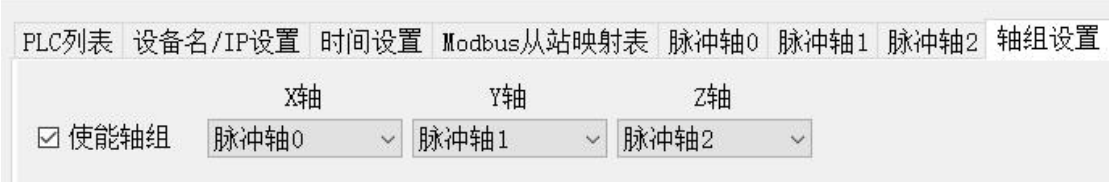
错误ID	错误信息	解决方法
1001	轴号错误	轴号是0-2

1002	脉冲轴配置错误	使用配置软件配置脉冲轴模式
1003	MC_Halt输入参数错误	检查MC_Halt输入参数
1004	MC_Home输入参数错误	检查MC_Home输入参数
1005	MC_Jog输入参数错误	检查MC_Jog输入参数
1006	MC_MoveAbsolute输入参数错误	检查MC_MoveAbsolute输入参数
1007	MC_MoveBuffer输入参数错误	检查MC_MoveBuffer输入参数
1008	MC_MoveRelative输入参数错误	检查MC_MoveRelative输入参数
1009	MC_MoveVelocity输入参数错误	检查MC_MoveVelocity输入参数
1010	MC_SetPosition输入参数错误	检查MC_SetPosition输入参数
1011	MC_Stop输入参数错误	检查MC_Stop输入参数
1012	速度设置过小	脉冲输出频率最低1Hz
1013	速度设置过大	脉冲输出频率最高100K
1014	加速度输入错误	加速度为正数
1015	减速度输入错误	加速度为正数
1016	加加速度输入错误	加加速度为正数
1017	方向输入错误	检查方向
1018	曲线类型输入错误	检查曲线类型
1019	MC_MoveBuffer Num错误	检查Num参数
1020	MC_MoveBufferr VelocityMode错误	检查VelocityMode参数
1021	MC_MoveBufferr AbsRelMode错误	检查AbsRelMode参数
1022	MC_Jog JogForward和JogBackward同时为高	JogForward和JogBackward不能同时为高
1023	MC_Home模式错误	使用配置软件配置回零模式
1024	MC_Home引脚错误	使用配置软件配置回零引脚
1025	MC_Home超时	回零超时
1026	MC_SetPosition Mode错误	检查Mode参数
1027	软限位错误	发生了软限位

1028	硬限位错误	发生了硬限位
2000	轴处于Disable状态	在合法状态触发指令
2001	轴处于ErrorStop状态	
2002	轴处于Stopping状态	
2003	轴处于StandStill状态	
2004	轴处于Discrete Motion状态	
2005	轴处于Continuous Motion状态	
2006	轴处于Homing状态	
2007	轴处于Synchronized Motion状态	

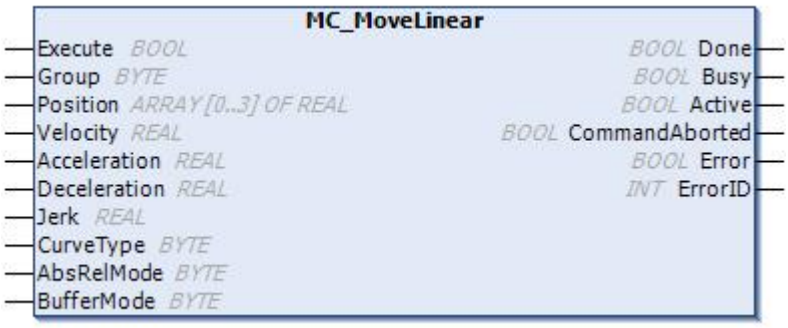
3.29 脉冲轴组（Tianjin Sange Electr. Tech. Bronze100）

Bronze100支持1个轴组，使用之前必须配置：



3.29.1 MC_MoveLinear——直线插补

直线插补指令 MC_MoveLinear 在库文件中的图示表达和相关参数说明如下：



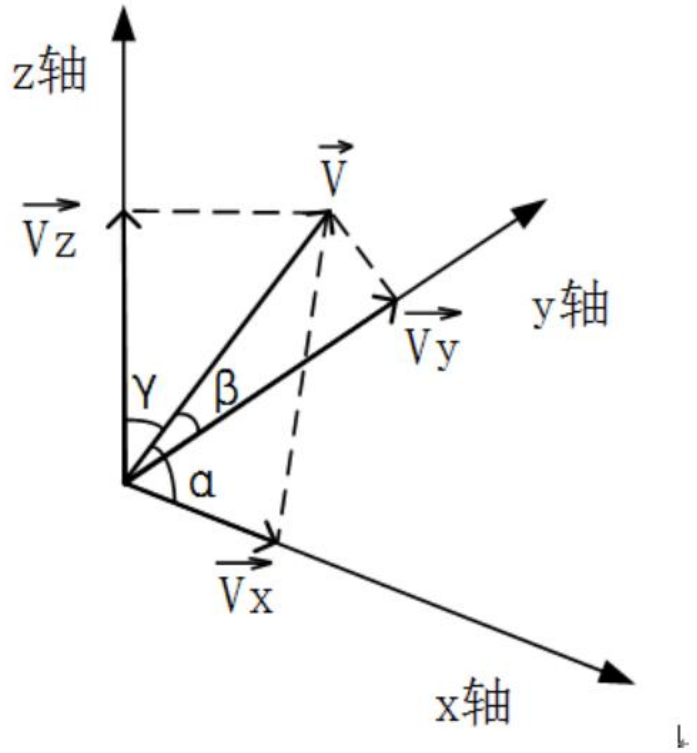
输入参数	数据类型	功能描述	参数值说明
Execute	BOOL	触发	上升沿有效
Group	BYTE	轴组ID	固定为0
Position	ARRAY[0..	目标位置	[0]：X轴目标位置

	3] OF REAL		[1]：Y轴目标位置 [2]：Z轴目标位置 [3]：保留 正数/负数/0
Velocity	REAL	目标速度	正数
Acceleration	REAL	加速度	正数
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数
CurveType	BYTE	速度曲线类型	0：T型速度曲线 1：S型速度曲线
AbsRelMode	BYTE	定位模式	0：绝对定位 1：相对定位
BufferMode	BYTE	缓冲模式	0：打断 + 无过渡 1：缓冲 + 无过渡 2：前一个速度 + 无过渡
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：插补未完成 1：插补完成
Busy	BOOL	忙标志	0：不忙 1：在忙
Active	BOOL	正在执行标志	0：不在执行本段曲线 1：正在执行本段曲线
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴组错误列表

合速度分解：

Velocity表示插补器的目标速率，其中坐标轴的标速度按照公式（1）、（2）、

（3）分解。



$$Vx = V \times \cos\alpha \quad (1)$$

$$Vy = V \times \cos\beta \quad (2)$$

$$Vz = V \times \cos\gamma \quad (3)$$

$$V = \sqrt[2]{Vx^2 + Vy^2 + Vz^2} \quad (4)$$

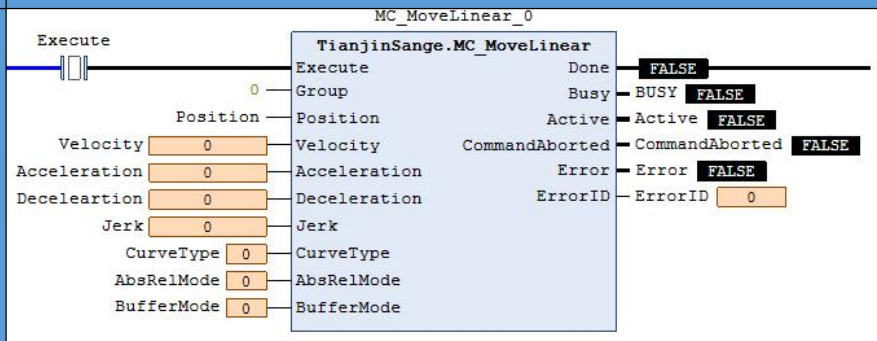
缓冲与过渡模式：

BufferMode	缓冲模式	描述
0	打断 + 无过渡	立即切换到下一个功能块动作，无过渡曲线
1	缓冲 + 无过渡	第一段减速完成开始执行缓冲的功能块，无过渡曲线
2	前一个速度 + 无过渡	以当前速度走到第一段结束并按照第一段的速率开始执行第二段

在Busy输出为ON时重复触发本指令，轴将报故障，同时所有轴立即停止运动，进入

ErrorStop状态。

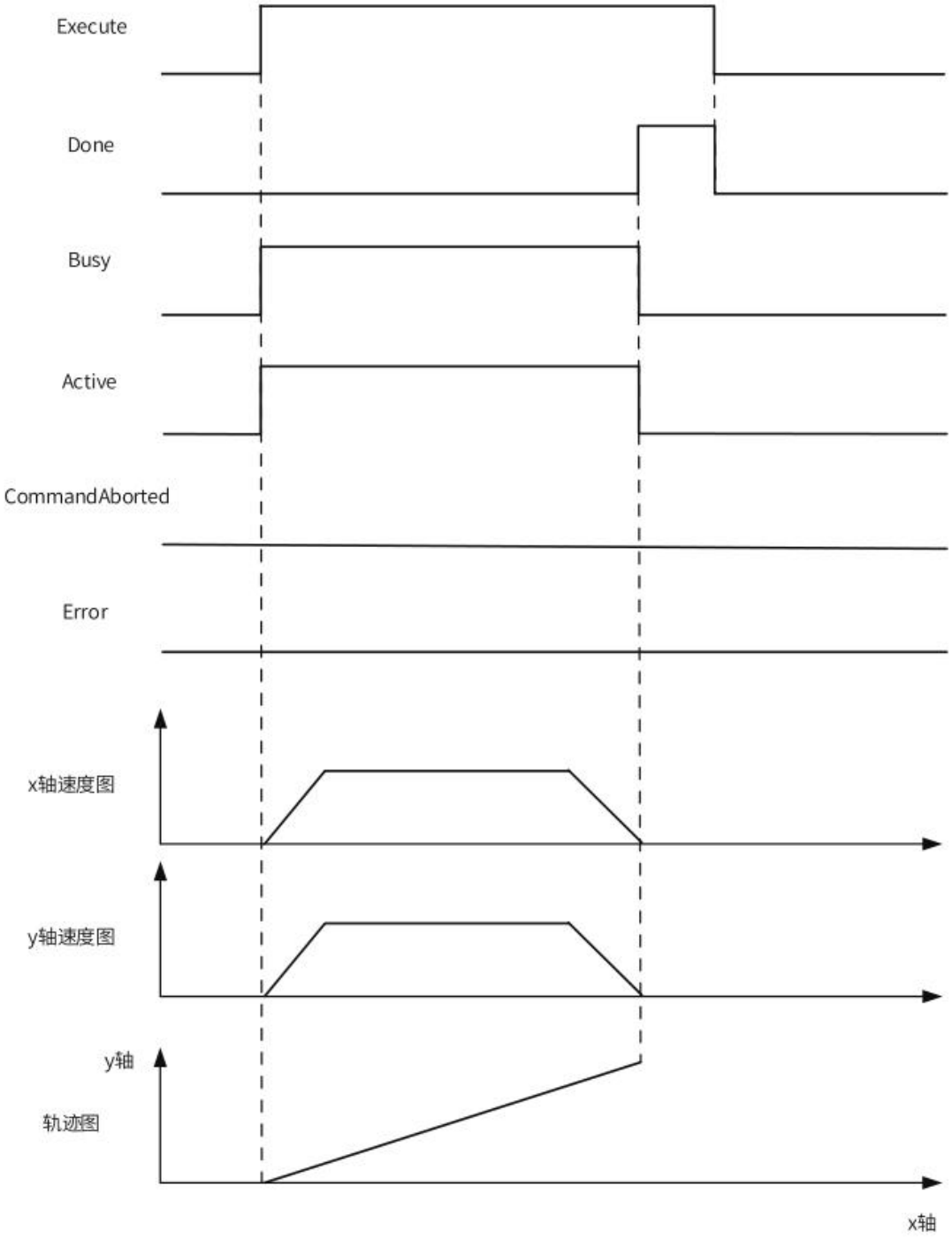
变量定义						
	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**MC_MoveLinear_0**		TianjinSange.MC_MoveLinear			
2	VAR	**Execute**		BOOL			
3	VAR	**Position**		ARRAY[0..3] OF REAL			
4	VAR	**Velocity**		REAL			
5	VAR	**Acceleration**		REAL			
6	VAR	**Deceleartion**		REAL			
7	VAR	**Jerk**		REAL			
8	VAR	**CurveType**		BYTE			
9	VAR	**AbsRelMode**		BYTE			
10	VAR	**BufferMode**		BYTE			
11	VAR	**Busy**		BOOL			
12	VAR	**Active**		BOOL			
13	VAR	**CommandAborted**		BOOL			
14	VAR	**Error**		BOOL			
15	VAR	**ErrorID**		INT			

编程语言	程序
梯形图（LD）	
结构化文本（ST）	MC_MoveLinear_0(Group:=0, Execute:=Execute, Position:=Position, Velocity:=Velocity, Acceleration:=Acceleration, Deceleration:=Deceleration, Jerk:=Jerk, CurveType:=CurveType, AbsRelMode:=AbsRelMode, BufferMode:=BufferMode,

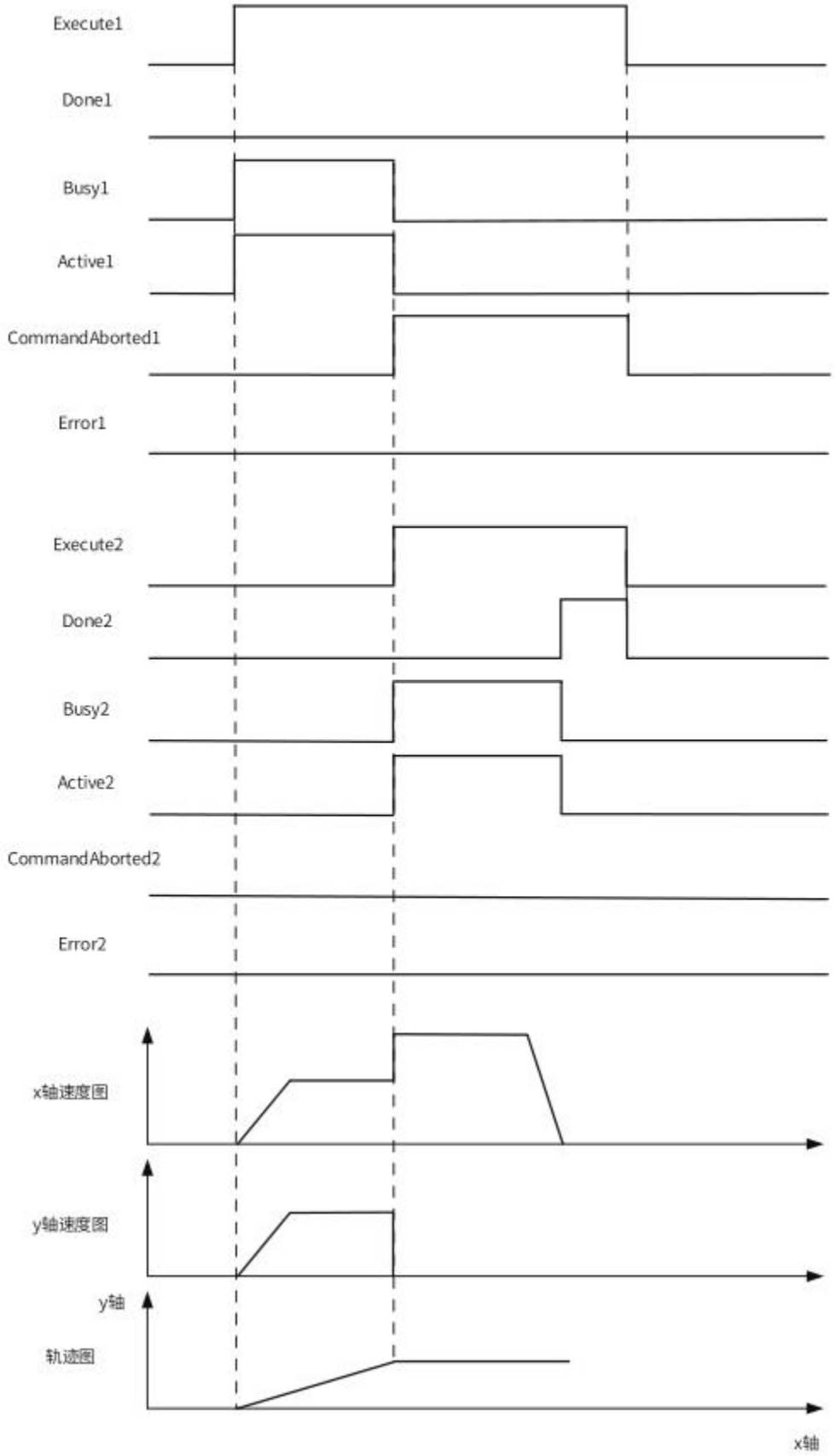
	Busy=>Busy, Active=>Active, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID);

指令时序图：

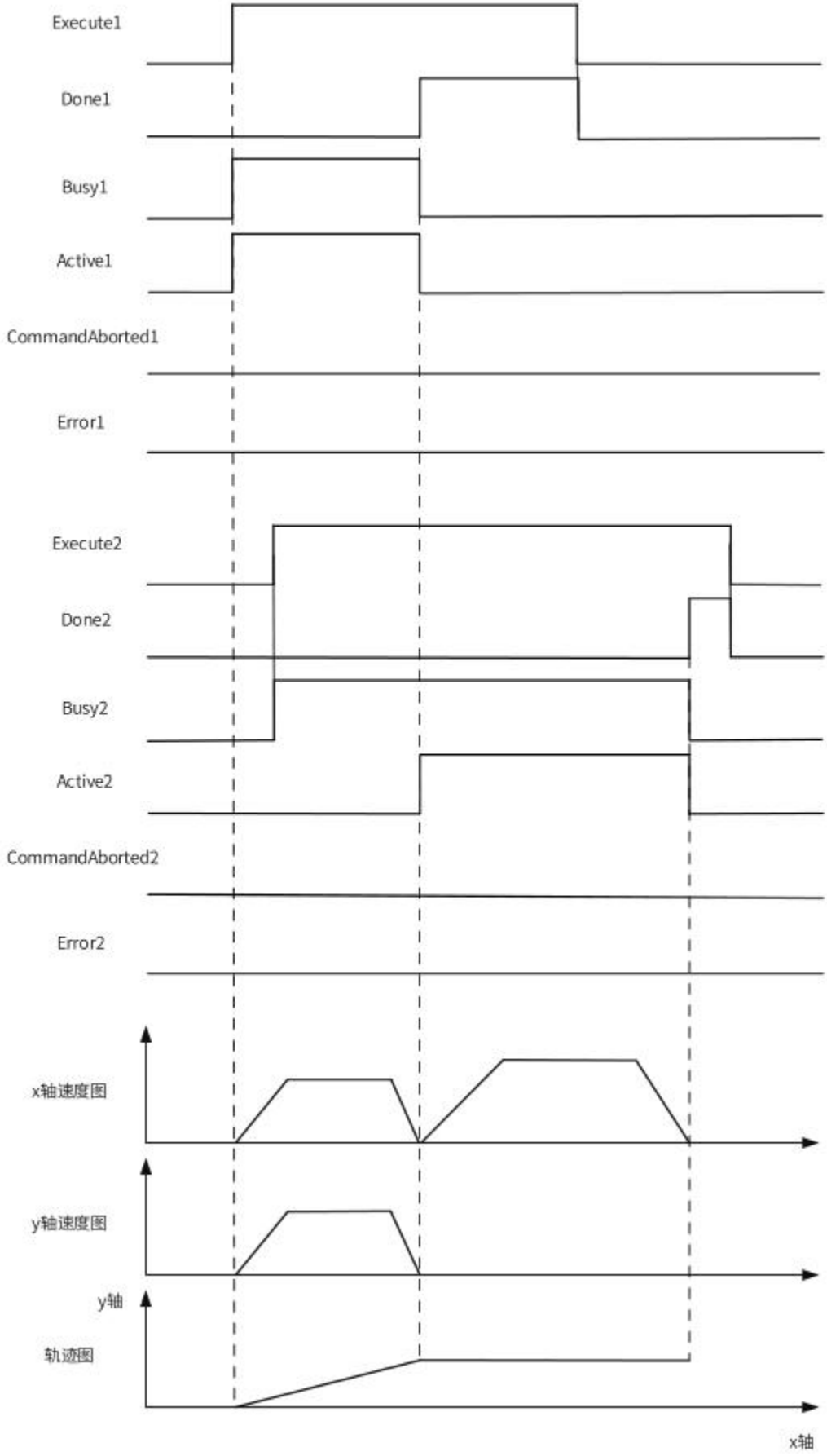
调用一个直线插补指令，在X轴、Y轴方向做插补：



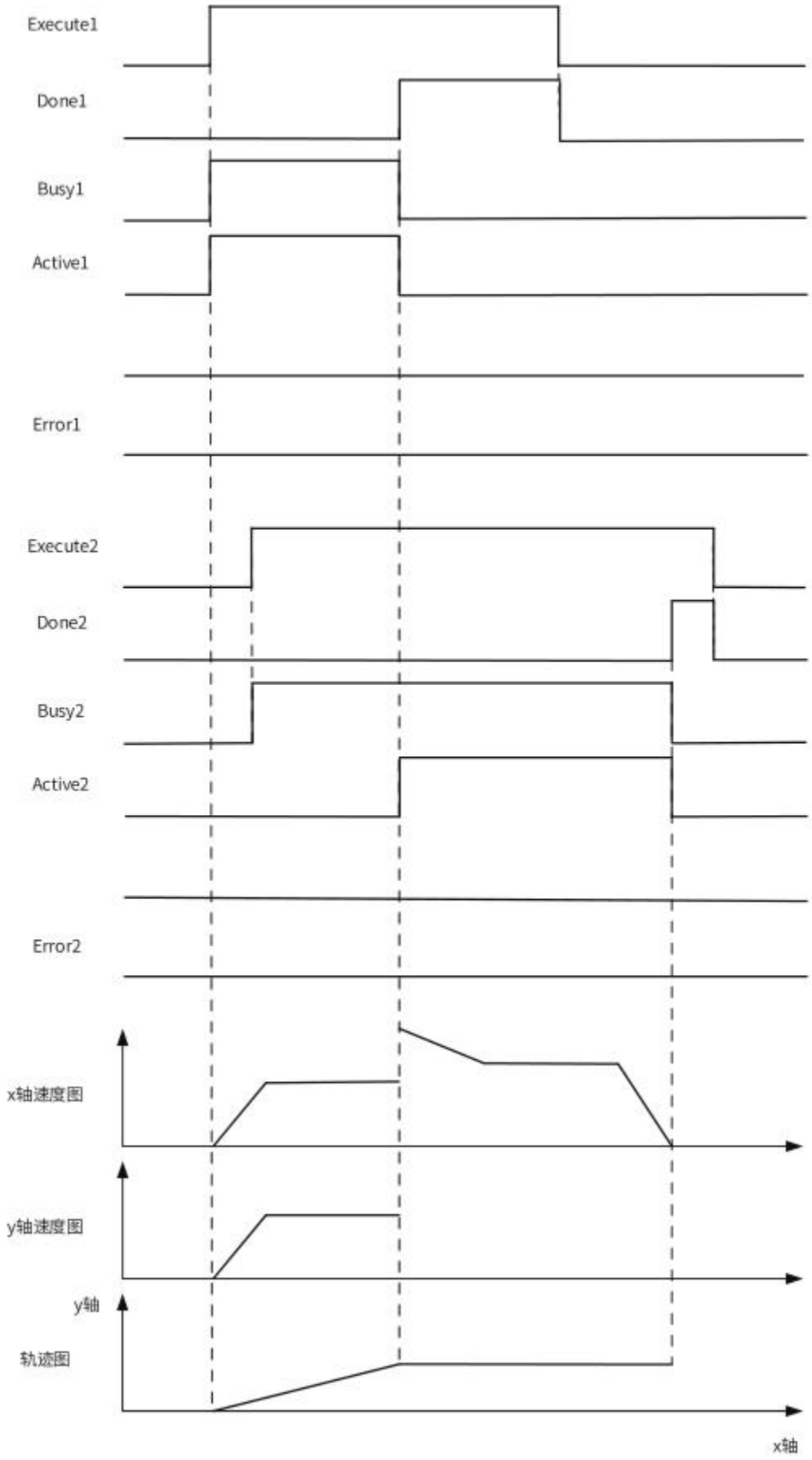
调用两个直线插补指令，在第一个插补指令运行过程中触发第二个插补指令，打断第一个插补指令：



调用两个直线插补指令，第二条指令采用“缓冲+无过渡”模式执行：

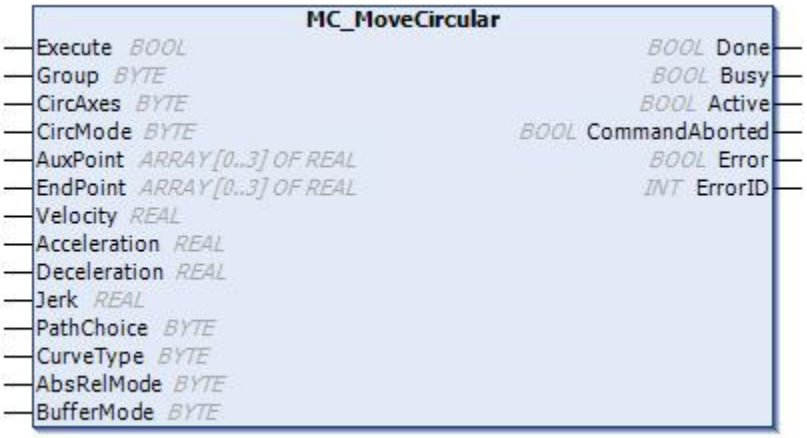

Execute1
Done1
Busy1
Active1
CommandAborted1
Error1
Execute2
Done2
Busy2
Active2
CommandAborted2
Error2
x轴速度图
y轴速度图
y轴
轨迹图
x轴

调用两个直线插补指令，第二条指令采用“前一个速度+无过渡”模式执行：


Execute1
Done1
Busy1
Active1
Error1
Execute2
Done2
Busy2
Active2
Error2
x轴速度图
y轴速度图
y轴
轨迹图
x轴

3.29.2 MC_MoveCircular——圆弧插补

圆弧插补指令 MC_MoveCircular 在库文件中的图示表达和相关参数说明如下：

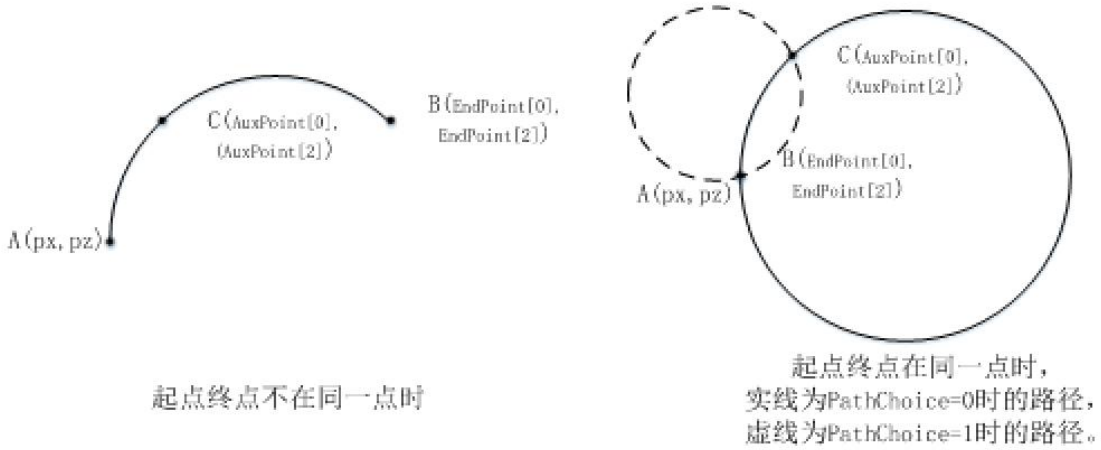


输入参数	数据类型	功能描述	参数值说明
Execute	BOOL	触发	上升沿有效
Group	BYTE	轴组ID	固定为0
CircAxes	BYTE	圆弧轴指定	0：x- y轴平面 1：y- z轴平面 2：x- z轴平面
CircMode	BYTE	圆弧插补模式	0：指定为通过点 1：指定为中心点 2：指定为半径
AuxPoint	ARRAY[0..3] OF REAL	辅助点坐标	
EndPoint	ARRAY[0..3] OF REAl	终点坐标	
Velocity	REAL	目标速度	正数
Acceleration	REAL	加速度	正数
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数
PathChoice	BYTE	路径选择	0：CW

			1：CCW
CurveType	BYTE	速度曲线类型	0：T型速度曲线 1：S型速度曲线
AbsRelMode	BYTE	定位模式	0：绝对定位 1：相对定位
BufferMode	BYTE	缓冲模式	0：打断 + 无过渡 1：缓冲 + 无过渡 2：前一个速度 + 无过渡
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：插补未完成 1：插补完成
Busy	BOOL	忙标志	0：不忙 1：在忙
Active	BOOL	正在执行标志	0：不在执行本段曲线 1：正在执行本段曲线
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴组错误列表

插补模式选择：

CircMode = 0代表根据通过点进行圆弧插补：



选择x- y平面时通过点为(AuxPoint[0], (AuxPoint[1])，终点为(EndPoint[0], EndPoint[1]); 选择y- z平面时通过点为(AuxPoint[1], (AuxPoint[2])，终点为(EndPoint[1], EndPoint[2])；选择x- z平面时通过点为(AuxPoint[0], (AuxPoint[2])，终点为(EndPoint[0], EndPoint[2])。

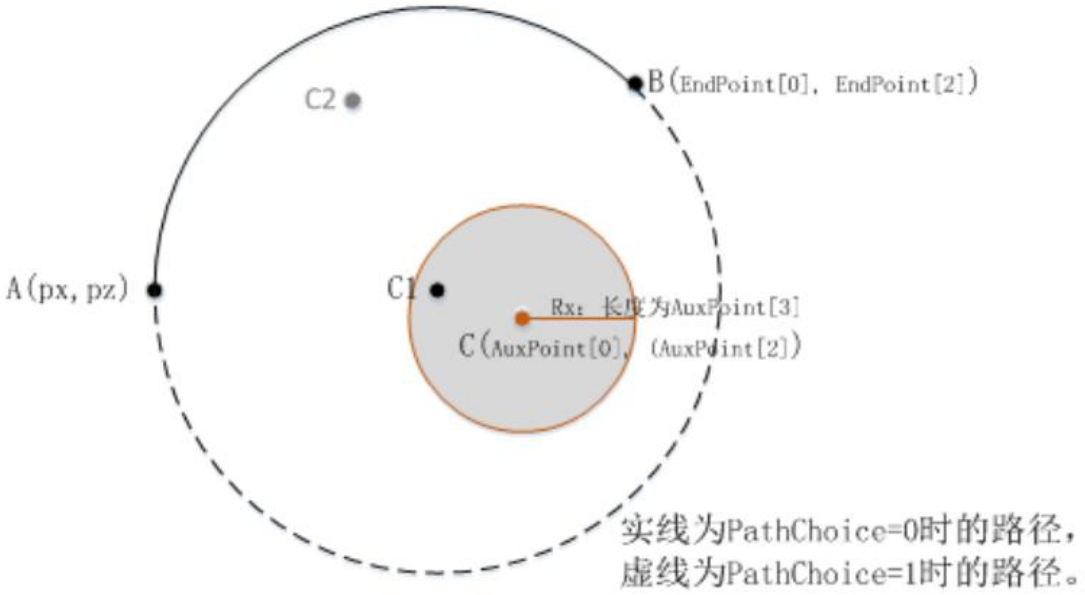
以x- y平面为例，x轴的起始位置为Px，y轴的起始位置为Py，触发指令后将执行以(Px, Py)为起点，以(EndPoint[0], EndPoint[1])为终点并通过点(AuxPoint[0], (AuxPoint[1])的圆弧插补。

当起点和终点为同一点时，以起点(Px, Py)和通过点(AuxPoint[0], (AuxPoint[1])为直径绘制正圆。这种情况下，通PathChoice (路径选择) 指定圆弧的旋转方向。

当起点、通过点与终点在同一条直线上时不能构成圆，指令报错，停止插补指令的执行。

当通过点与终点为同一点或者起点和通过点位置为同一点时，指令报错，停止插补指令的执行。

CircMode = 1代表根据中心点进行圆弧插补：

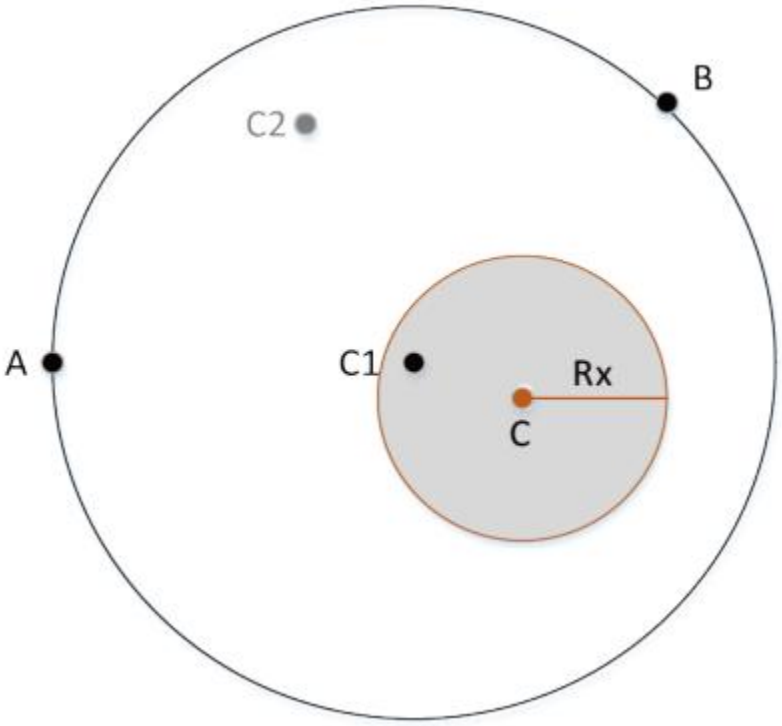


选择x- y平面时中心点为(AuxPoint[0], (AuxPoint[1])，终点为(EndPoint[0], EndPoint[1])；择y- z平面时中心点为(AuxPoint[1], (AuxPoint[2])，终点为(EndPoint[1], EndPoint[2])；择x- z平面时中心点为(AuxPoint[0], (AuxPoint[2])，终点为(EndPoint[0], EndPoint[2])；

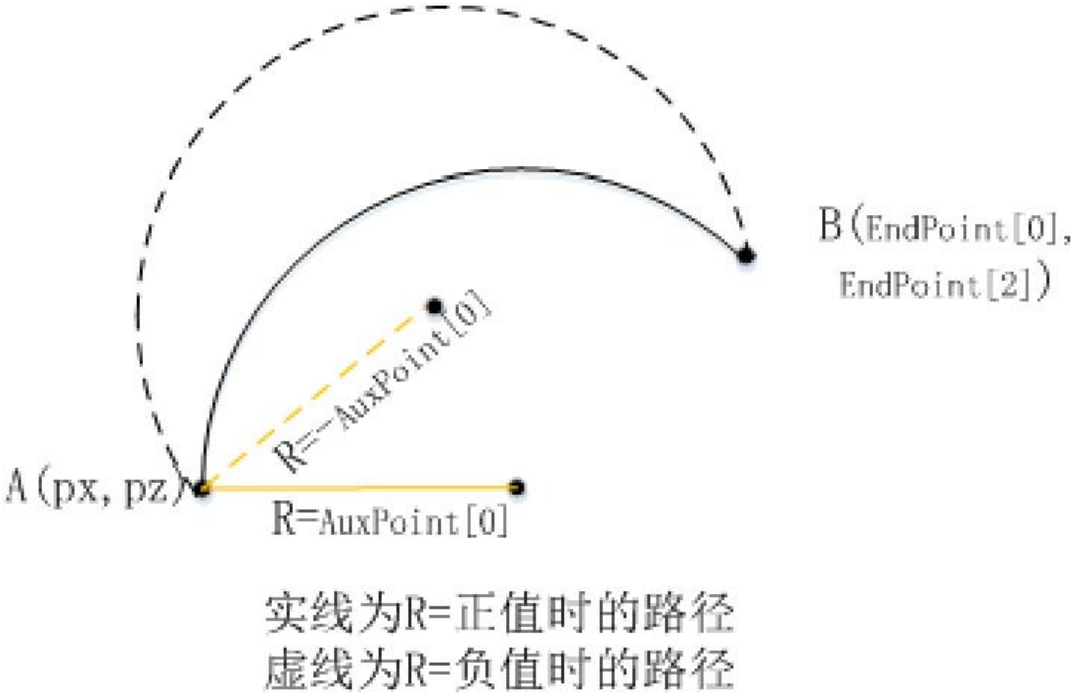
以x轴和z轴为例，x轴的起始位置为Px，z轴的起始位置为Pz，触发指令后将执行以(Px, Pz)为起点，以 (AuxPoint[0], (AuxPoint[2]) 为圆心，以 (EndPoint[0], EndPoint[2])为终点的圆弧插补，圆弧的绘制旋转方向PathChoice决定。

当从指定的中心位置（AuxPoint[0],AuxPoint[2]）到起点(Px, Pz)的距离R1与到终点(EndPoint[0],EndPoint[2])的距离R2不同时(R1与R2的差值大于1)，将用R1和R2计算出平均值R，并按照与半径指定同样的方法计算中心位置（Cx,Cy），使用该半径和中心位置画圆弧。

需要注意的是，在重新调整中心位置时，如果算出两个圆心，首先计算出两个圆心到设定中心点的距离，选出距离短的那个且该点必须位于以（AuxPoint[0],AuxPoint[2]）为圆心以AuxPoint[3]为半径的圆的内部。如下图所示，重新调整时以C1点为新的圆心。



CircMode = 2代表根据指定半径进行圆弧插补：



不管选择哪个平面，圆弧的半径大小始终由|AuxPoint[0]|决定。选择x- y平面时终点为(EndPoint[0],EndPoint[1])；择y- z平面时终点为(EndPoint[1], EndPoint[2])；择x- z平面时终点为(EndPoint[0],EndPoint[2])；

以y轴和z轴为例，y轴的起始位置为Py，z轴的起始位置为Pz，y轴和z轴执行以(Py, Pz)为起点，以|AuxPoint[0]|为半径，以 (EndPoint[1], EndPoint[2])为终点的圆弧插补。

半径符号为负时，绘制出较长的圆弧；半径符号为正时，绘制出较短的圆弧。圆弧的旋转方向通过PathChoice(路径选择) 指定。

缓冲与过渡模式：

参考MC_MoveLinear。

辅助轴的处理：

选择xy平面时，z轴为辅助轴；yz平面时，x轴为辅助轴；选择xz平面时，y轴为辅助轴。

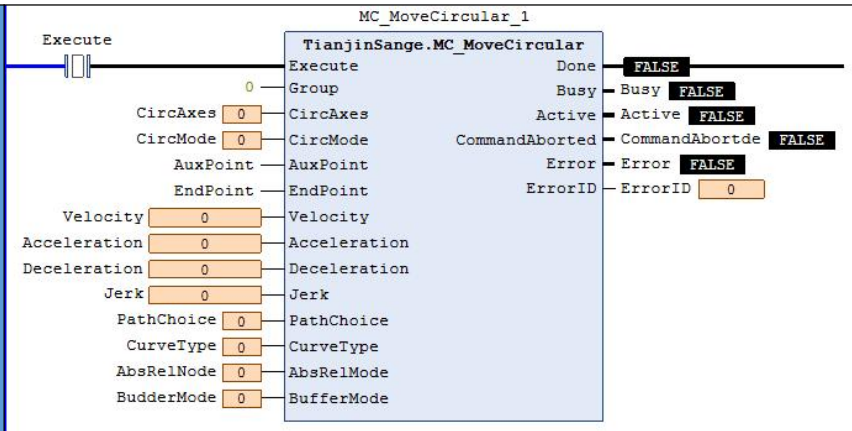
在圆弧插补中，辅助轴将同圆弧坐标轴一起开始运行，并一起结束动作。在圆弧坐标轴运行过程中，辅助轴执行从起点到终点的直线插补，辅助轴在终点的坐标由EndPoint指定。例如选择xy平面时，z轴的终点坐标为EndPoint[2]。

在Busy输出为ON时重复触发本指令，轴将报故障，同时所有轴立即停止运动，进入ErrorStop状态。

变量定义

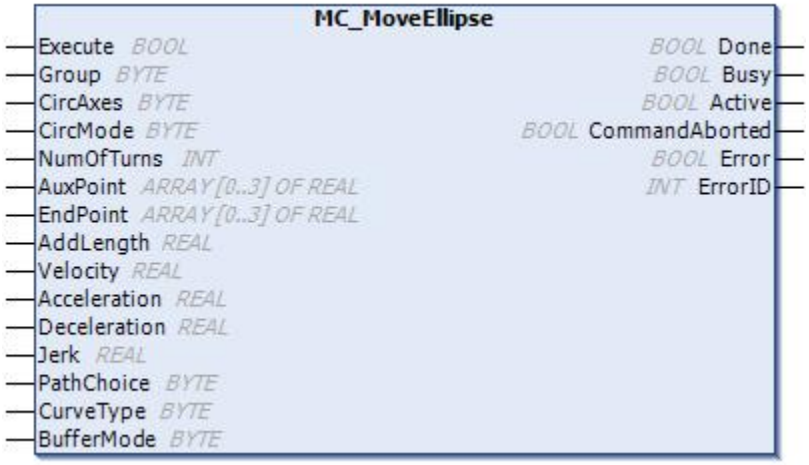
	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	CircAxes		BYTE			
2	VAR	CircMode		BYTE			
3	VAR	AuxPoint		ARRAY [0..3] OF REAL			
4	VAR	EndPoint		ARRAY [0..3] OF REAL			
5	VAR	Velocity		REAL			
6	VAR	Acceleration		REAL			
7	VAR	Deceleration		REAL			
8	VAR	Jerk		REAL			
9	VAR	PathChoice		BYTE			
10	VAR	CurveType		BYTE			
11	VAR	AbsRelNode		BYTE			
12	VAR	BudderMode		BYTE			
13	VAR	Busy		BOOL			
14	VAR	Active		BOOL			
15	VAR	CommandAbortde		BOOL			
16	VAR	Error		BOOL			
17	VAR	ErrorID		INT			
18	VAR	MC_MoveCircular_1		TianjinSange.MC_MoveCircular			
19	VAR	Execute		BOOL			

编程语言	程序

梯形图（LD）	
结构化文本（ST）	MC_MoveCircular_1(Group:=0, Execute:=Execute, CircAxes:=CircAxes, CircMode:=CircMode, AuxPoint:=AuxPoint, EndPoint:=EndPoint, Velocity:=Velocity, Acceleration:=Acceleration, Deceleration:=Deceleration, Jerk:=Jerk, PathChoice:=PathChoice, CurveType:=CurveType, AbsRelMode:=AbsRelMode, BufferMode:=BufferMode, Busy=>Busy, Active=>Active, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID);

3.29.3 MC_MoveEllipse——椭圆插补

椭圆插补指令 MC_MoveEllipse 在库文件中的图示表达和相关参数说明如下：



输入参数	数据类型	功能描述	参数值说明
Execute	BOOL	触发	上升沿有效
Group	BYTE	轴组ID	固定为0
CircAxes	BYTE	圆弧轴指定	0：x- y轴平面 1：y- z轴平面 2：x- z轴平面
CircMode	BYTE	椭圆插补模式	0：完整的椭圆 1：指定弧长
NumOfTurns	INT	圈数	1-100
AuxPoint	ARRAY[0..3] OF REAL	中心点坐标	
EndPoint	ARRAY[0..3] OF REAl	终点坐标	
AddLength	REAL	弧长	CircMode = 1时表示运行的弧长 正数/负数/0
Velocity	REAL	目标速度	正数
Acceleration	REAL	加速度	正数
Deceleration	REAL	减速度	正数
Jerk	REAL	加加速度	正数

PathChoice	BYTE	路径选择	0：CW 1：CCW
CurveType	BYTE	速度曲线类型	0：T型速度曲线 1：S型速度曲线
AbsRelMode	BYTE	定位模式	0：绝对定位 1：相对定位
BufferMode	BYTE	缓冲模式	0：打断 + 无过渡 1：缓冲 + 无过渡 2：前一个速度 + 无过渡
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：插补未完成 1：插补完成
Busy	BOOL	忙标志	0：不忙 1：在忙
Active	BOOL	正在执行标志	0：不在执行本段曲线 1：正在执行本段曲线
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴组错误列表

椭圆的起点与中心点：

椭圆的起点为调用指令时坐标轴的设置位置，中心点为指令中的(AuxPoint[0],AuxPoint[1])。

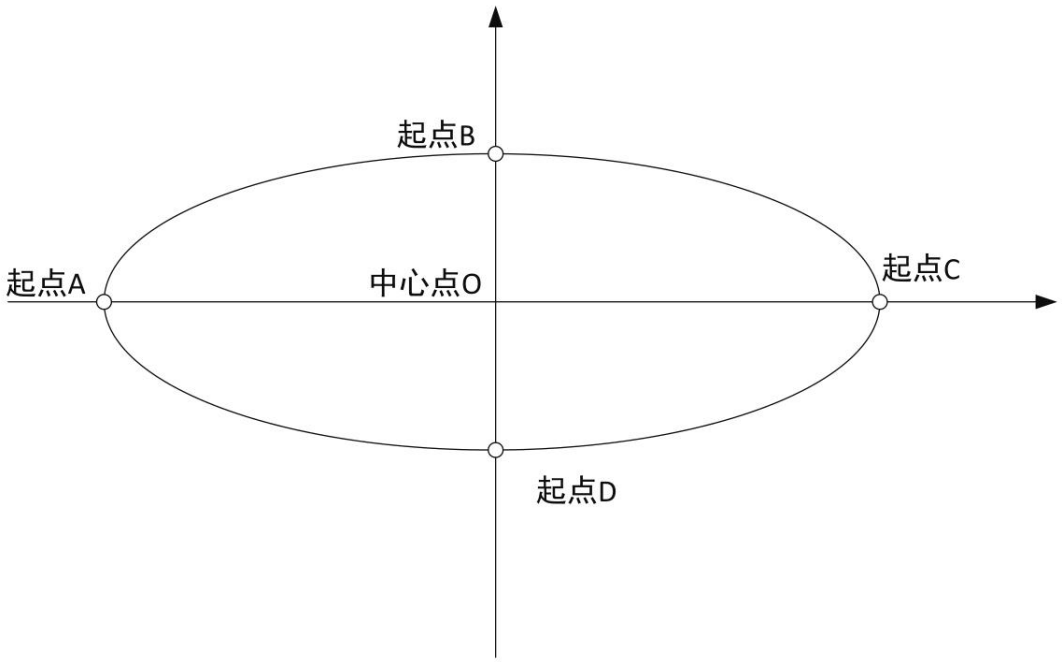
当选择xy平面时，AuxPoint[0]为x轴坐标，AuxPoint[1]为y轴坐标。

当选择xz平面时，AuxPoint[0]为x轴坐标，AuxPoint[1]为z轴坐标。

当选择yz平面时，AuxPoint[0]为y轴坐标，AuxPoint[1]为z轴坐标。

要求起点和中心点的连线必须平行于坐标轴。以xy平面为例，设起点坐标为（x0,y0），中心点坐标为（x1,y1）。当x0=x1时两点的连线平行于y坐标轴，此

时如果y0 > y1，起点为下图中的C点，如果y0 < y1，此时起点为下图中的A点。

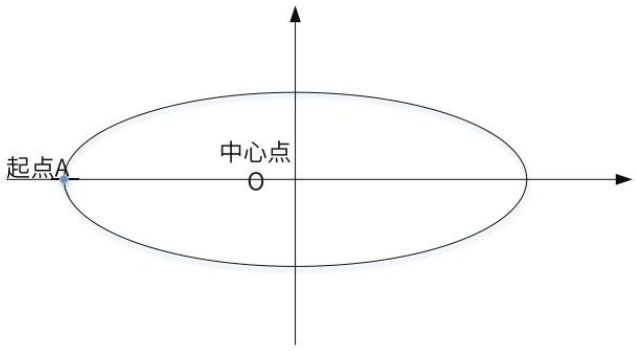


长轴与短轴的确定：

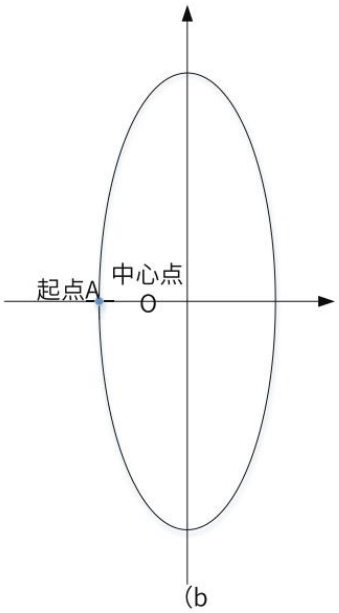
指令中参数AuxPoint[2]、AuxPoint[3]用于设置椭圆长半轴和短半轴的长度，当AuxPoint[2] > AuxPoint[3]时，AuxPoint[2]为长半轴，否则AuxPoint[3]为长半轴。

假如AuxPoint[2] = 12，AuxPoint[3] = 5，如果|AO|的距离等于12，则最终选择的是左侧的椭圆，如果|AO|的距离等于5，则最终选择的右侧的椭圆。

当长轴等于短轴时，椭圆将变成正圆。



（a
图）



（b
图）

终点的确认：

由参数CircMode确定终点的位置。

当CircMode = 0时，运行完整的椭圆，此时终点等于起点。

当CircMode = 1时，需要指定从起点开始运行的弧长。当NumOfTurns = 0时，PathChoice指定方向为CW（CCW），AddLength为正，则顺时针（逆时针）运行AddLength指定的弧长，如果AddLength为负，则逆时针（顺时针）运行AddLength指定的弧长；如则NumOfTurns大于0，此时首先按照PathChoice指定的方向运行NumOfTurns － 1圈，然后在最后一圈按照AddLength指定的弧长运行。

缓冲与过渡模式：

参考MC_MoveLinear。

辅助轴的处理：

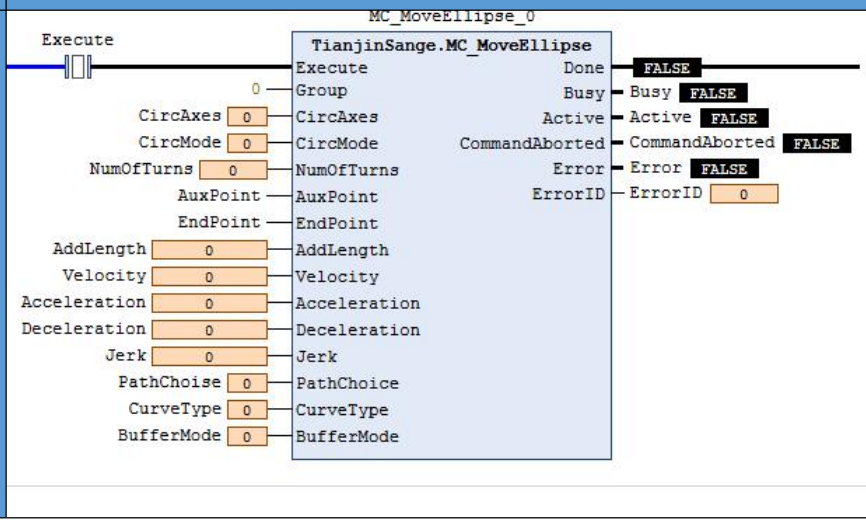
选择xy平面时，z轴为辅助轴；yz平面时，x轴为辅助轴；选择xz平面时，y轴为辅助轴。

在椭圆插补中，辅助轴将同椭圆坐标轴一起开始运行，并一起结束动作。在椭圆坐标轴运行过程中，辅助轴执行从起点到终点的直线插补，辅助轴在终点的坐标由EndPoint指定。例如选择xy平面时，z轴的终点坐标为EndPoint[2]。

在Busy输出为ON时重复触发本指令，轴将报故障，同时所有轴立即停止运动，进入ErrorStop状态。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_MoveEllipse_0		TianjinSange.MC_MoveEllipse			
2	VAR	Execute		BOOL			
3	VAR	CircAxes		BYTE			
4	VAR	CircMode		BYTE			
5	VAR	NumOfTurns		INT			
6	VAR	AuxPoint		ARRAY [0..3] OF REAL			
7	VAR	EndPoint		ARRAY [0..3] OF REAL			
8	VAR	AddLength		REAL			
9	VAR	Velocity		REAL			
10	VAR	Acceleration		REAL			
11	VAR	Deceleration		REAL			
12	VAR	Jerk		REAL			
13	VAR	PathChoise		BYTE			
14	VAR	CurveType		BYTE			
15	VAR	BufferMode		BYTE			
16	VAR	Busy		BOOL			
17	VAR	Active		BOOL			
18	VAR	CommandAborted		BOOL			
19	VAR	Error		BOOL			
20	VAR	ErrorID		INT			

编程语言	程序
梯形图（LD）	

结构化文本（ST）	MC_MoveEllipse_0(Group:=0, Execute:=Execute, CircAxes:=CircAxes, CircMode:=CircMode, NumOfTurns:=NumOfTurns, AuxPoint:=AuxPoint, EndPoint:=EndPoint, AddLength:=AddLength, Velocity:=Velocity, Acceleration:=Acceleration, Deceleration:=Deceleration, Jerk:=Jerk, PathChoice:=PathChoise, CurveType:=CurveType, BufferMode:=BufferMode, Busy=>Busy, Active=>Active, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID);

3.29.4 MC_GroupStop——轴组停止

轴组停止指令 MC_GroupStop 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：不使能停止 1：使能停止

Group	BYTE	轴组ID	固定为0
StopMode	BYTE	停止模式	0：减速停止 1：立即停止
Deceleration	REAL	减速度	正数
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：停止未完成 1：停止完成
Busy	BOOL	忙标志	0：不忙 1：在忙
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴组错误列表

在Execute的上升沿，插补器根据StopMode设置的停机方式执行停机，并打断所有处于缓冲状态的插补指令，停机完成后Done信号输出有效，单轴的PLCOpen状态机仍处于Synchronized Motion状态。

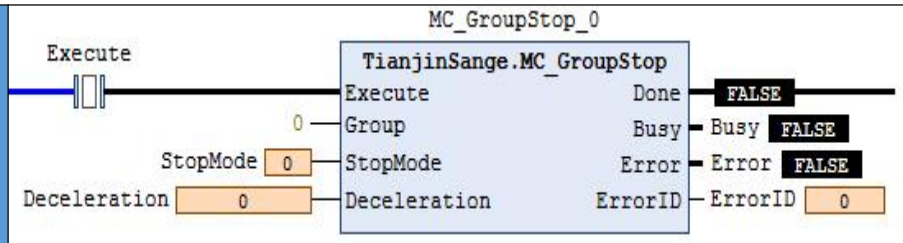
在Execute=ON期间，插补器一直处于停止状态，此时触发新的插补指令无效。

在Execute的下降沿，插补器将切换到非停止状态，单轴进入StandStill状态，此时可以触发新的插补指令。

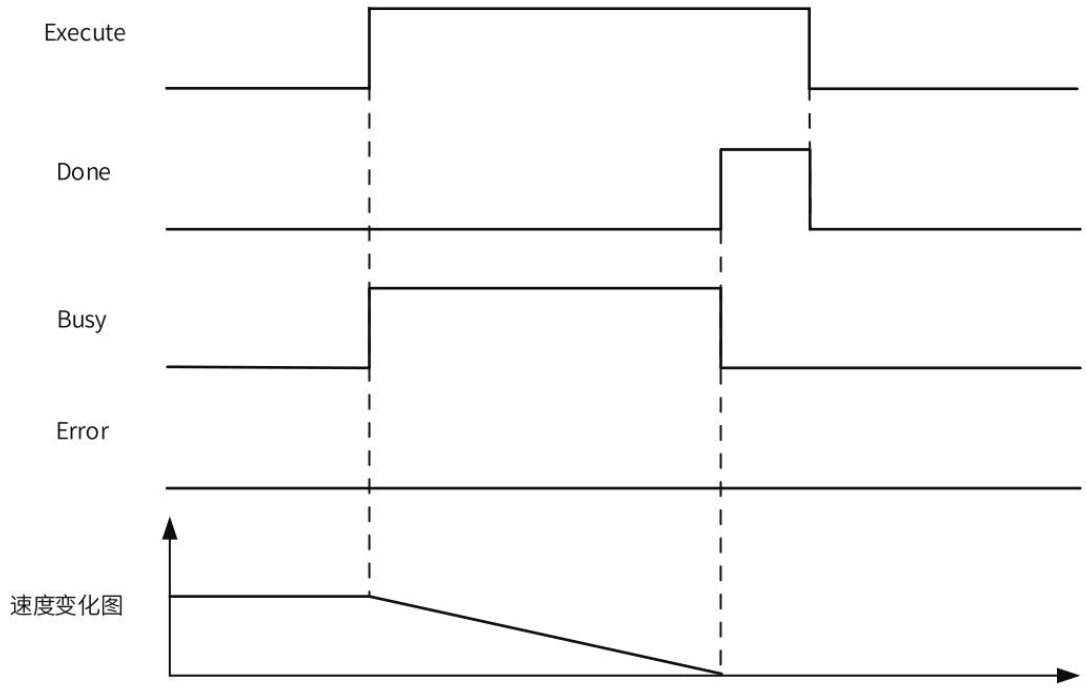
变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	**MC_GroupStop_0**		TianjinSange.MC_GroupStop			
2	VAR	**Execute**		BOOL			
3	VAR	**StopMode**		BYTE			
4	VAR	**Deceleration**		REAL			
5	VAR	**Busy**		BOOL			
6	VAR	**Error**		BOOL			
7	VAR	**ErrorID**		INT			

编程语言	**程序**

梯形图（LD）	
结构化文本（ST）	MC_GroupStop_0(Group:=0, Execute:=Execute, StopMode:=StopMode, Deceleration:=Deceleration, Busy=>Busy, Error=>Error, ErrorID=>ErrorID);

指令时序图：



3.29.5 MC_GroupPause——轴组暂停

轴组暂停指令 MC_GroupPause 在库文件中的图示表达和相关参数说明如下：

输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	0：不使能暂停 1：使能暂停
Group	BYTE	轴组ID	固定为0
Deceleration	REAL	减速度	正数
输出参数	**数据类型**	**功能描述**	**参数值说明**
Done	BOOL	完成标志	0：暂停未完成 1：暂停完成
Busy	BOOL	忙标志	0：不忙 1：在忙
CommandAborted	BOOL	中止执行	0：未中止 1：中止执行
Error	BOOL	错误标志	0：没有错误 1：有错误
ErrorID	INT	错误ID	见后面脉冲轴组错误列表

当轴组内的轴都处于StandStill状态时：

将Enable设置为ON，此时轴组内的轴仍处于StandStill状态，如果此时触发直线插补指令或者圆弧插补指令，轴组内的轴将切换到Synchronized Motion状态，但是处于暂停状态，不执行插补算法。只有MC_GroupPause指令的Enable信号设置为OFF时才开始执行插补算法。

当轴组内的轴都处于Synchronized Motion状态时：

在Enable的上升沿，插补器根据Deceleration设置的减速度执行减速过程，减速完成后Done信号输出有效，单轴的PLCOpen状态机仍处于Synchronized Motion状态，暂停期间正在执行的插补指令的Busy信号和Valid信号保持输出有效。

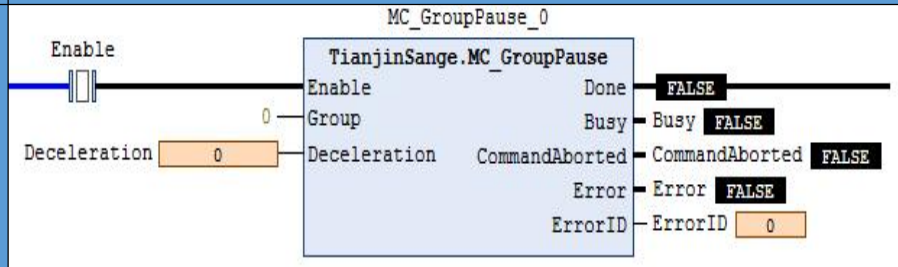
在Enable=ON期间，插补器一直处于暂停状态，此时触发新的插补指令新的插补

指令处于缓冲状态。

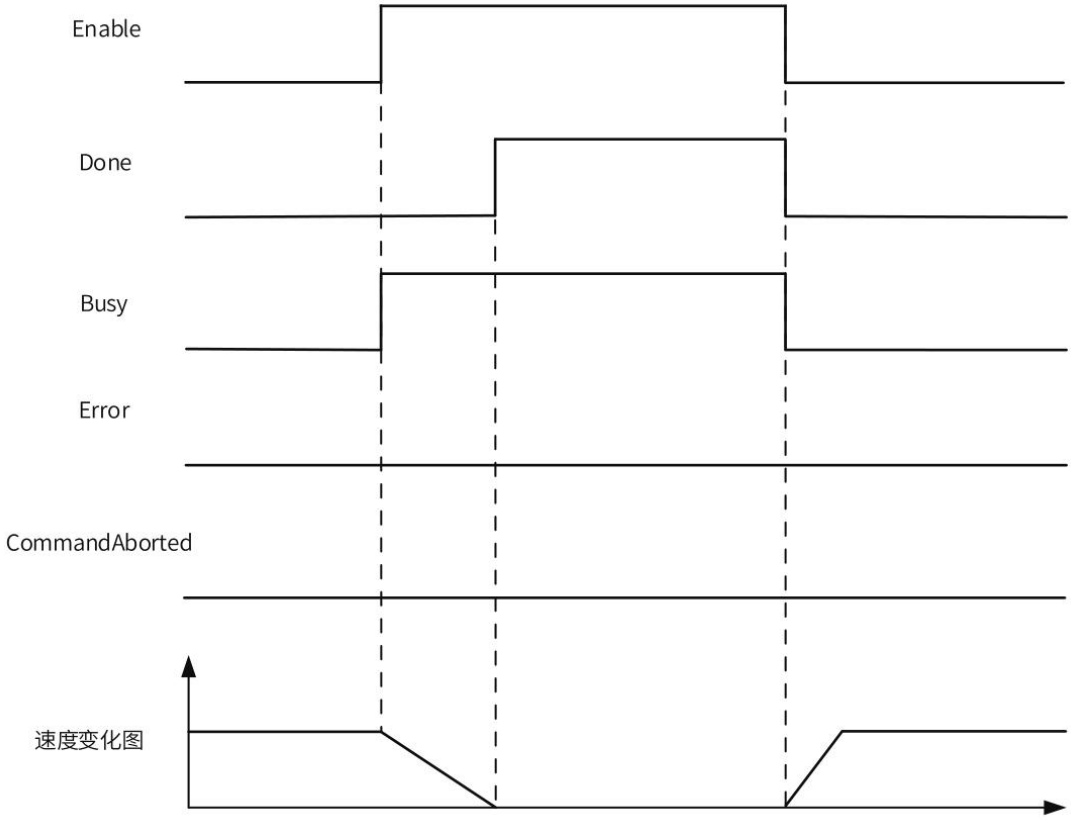
在Enable的下降沿，插补器重新开始执行之前被暂停的插补指令。

变量定义

	类别	名称	地址	数据类型	初值	注释	属性
1	VAR	MC_GroupPause_0		TianjinSange.MC_GroupPause			
2	VAR	Enable		BOOL			
3	VAR	Deceleration		REAL			
4	VAR	Busy		BOOL			
5	VAR	CommandAborted		BOOL			
6	VAR	Error		BOOL			
7	VAR	ErrorID		INT			

编程语言	程序
梯形图（LD）	
结构化文本（ST）	MC_GroupPause_0(Group:=0, Enable:=Enable, Deceleration:=Deceleration, Busy=>Busy, CommandAborted=>CommandAborted, Error=>Error, ErrorID=>ErrorID);

指令时序图：



3.29.6 轴组错误码

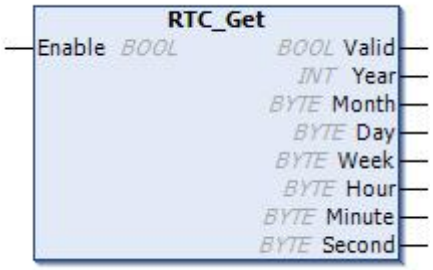
错误ID	错误信息	解决方法
1501	轴组号错误	轴组号是0
1502	轴组未配置	使用配置软件配置轴组
1503	轴组内轴数小于2个	使用配置软件配置轴组
1504	轴组内有轴相同	使用配置软件配置轴组
1505	轴组内有轴配置错误	使用配置软件配置轴
1506	轴组内有轴单位不同	使用配置软件配置轴
1507	轴组内有轴配置为环形模式	使用配置软件配置轴
1508	MC_GroupPause输入参数错误	检查MC_GroupPause输入参数
1509	MC_GroupStop输入参数错误	检查MC_GroupStop输入参数
1510	MC_MoveLinear输入参数错误	检查MC_MoveLinear输入参数
1511	MC_MoveCircular输入参数错误	检查MC_MoveCircular输入参数
1512	MC_MoveEllipse输入参数错误	检查MC_MoveEllipse输入参数

1513	插补功能块Busy时重复触发	插补功能块Busy时不能重复触发
1514	轴组处于Stop状态	Stop时不能触发插补和暂停
1515	MC_GroupStop Mode错误	检查Mode参数
1516	插补指令缓存区	只能缓存8条插补曲线
1517	插补绝对/相对模式错误	检查AbsRelMode参数
1518	插补指令缓冲模式错误	检查BufferMode
1519	圆弧/椭圆插补轴平面错误	检查CircAxes参数
1520	圆弧/椭圆插补轴缺失	检查CircAxes参数，使用配置软件配置轴和轴组
1521		
1522	圆弧/椭圆插补模式错误	检查CircMode参数
1523	满插补路径错误	检查PathChoice参数
1524	椭圆插补圈数错误	检查NumberOfTurns，0-100
1525	圆弧插补构不成圆弧	检查圆弧插补输入参数
1526	椭圆插补构不成椭圆	检查椭圆插补输入参数
1527	软限位错误	发生了软限位
1528	硬限位错误	发生了硬限位
2000	轴组内有轴处于Disable状态	在合法状态触发指令
2001	轴组内有轴处于ErrorStop状态	
2002	轴组内有轴处于Stopping状态	
2003	轴组内有轴处于StandStill状态	
2004	轴组内有轴处于Discrete Motion状态	
2005	轴组内有轴处于Continuous Motion状态	
2006	轴组内有轴处于Homing状态	
2007	轴组内有轴处于Synchronized Motion状态	
1001-1500错误ID见单轴错误码		

3.30 RTC（Tianjin Sange Electr. Tech. Bronze100）

3.30.1 RTC_Get——获取RTC日期时间

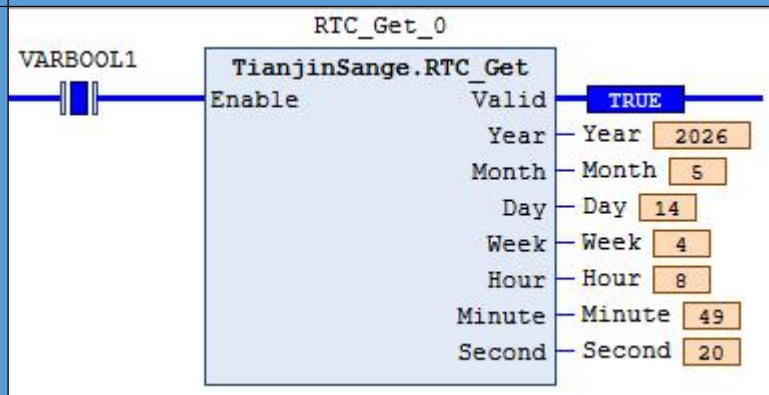
获取RTC日期时间指令 RTC_Get 在库文件中的图示表达和相关参数说明如下：



输入参数	数据类型	功能描述	参数值说明
Enable	BOOL	使能	
输出参数	**数据类型**	**功能描述**	**参数值说明**
Valid	BOOL	数据有效标志	0：数据无效 1：数据有效
Year	INT	年	年
Month	BYTE	月	月
Day	BYTE	日	日
Week	BYTE	0：周日 1：周一 。。。 6：周六	0：周日 1：周一 。。。 6：周六
Hour	BYTE	时	时
Minute	BYTE	分	分
Second	BYTE	秒	秒

变量定义

类别	名称	地址	数据类型	初值	注释	属性
VAR	**RTC_Get_0**		TianjinSange.RTC_Get			
VAR	**Year**		INT			
VAR	**Month**		BYTE			
VAR	**Day**		BYTE			
VAR	**Week**		BYTE			
VAR	**Hour**		BYTE			
VAR	**Minute**		BYTE			
VAR	**Second**		BYTE			
VAR	**VARBOOL1**		BOOL			
VAR	**X**		BOOL			

编程语言	程序
梯形图（LD）	
结构化文本（ST）	TianjinSange.RTC_Get(Enable:= VARBOOL1, Valid=> X, Year=> Year, Month=> Month, Day=> Day, Week=> Week, Hour=> Hour, Minute=> Minute, Second=> Second);

联系方式及售后

公司网址：www.tj-sange.com　www.sange-cbm.com

售后技术电话：022-2210-6681　130-7220-8083（微信）

售前购买咨询：176-0260-2061（同微信）

公众账号：获取产品使用视频和更多资讯。