

# LC1500系列 PLC快速应用手册



## 公司简介



苏州市凌臣采集计算机有限公司成立于2006年，是一家本着与客户共赢、为客户创造方案价值的经营理念的企业。为客户提供各种测试测量、运动控制、机器视觉、机器人等自动化设备的核心零部件和系统解决方案。凌臣科技于2017年开始创立凌臣采集LCT品牌，研发了包括工控机、PLC控制器、EtherCAT步进驱动器、

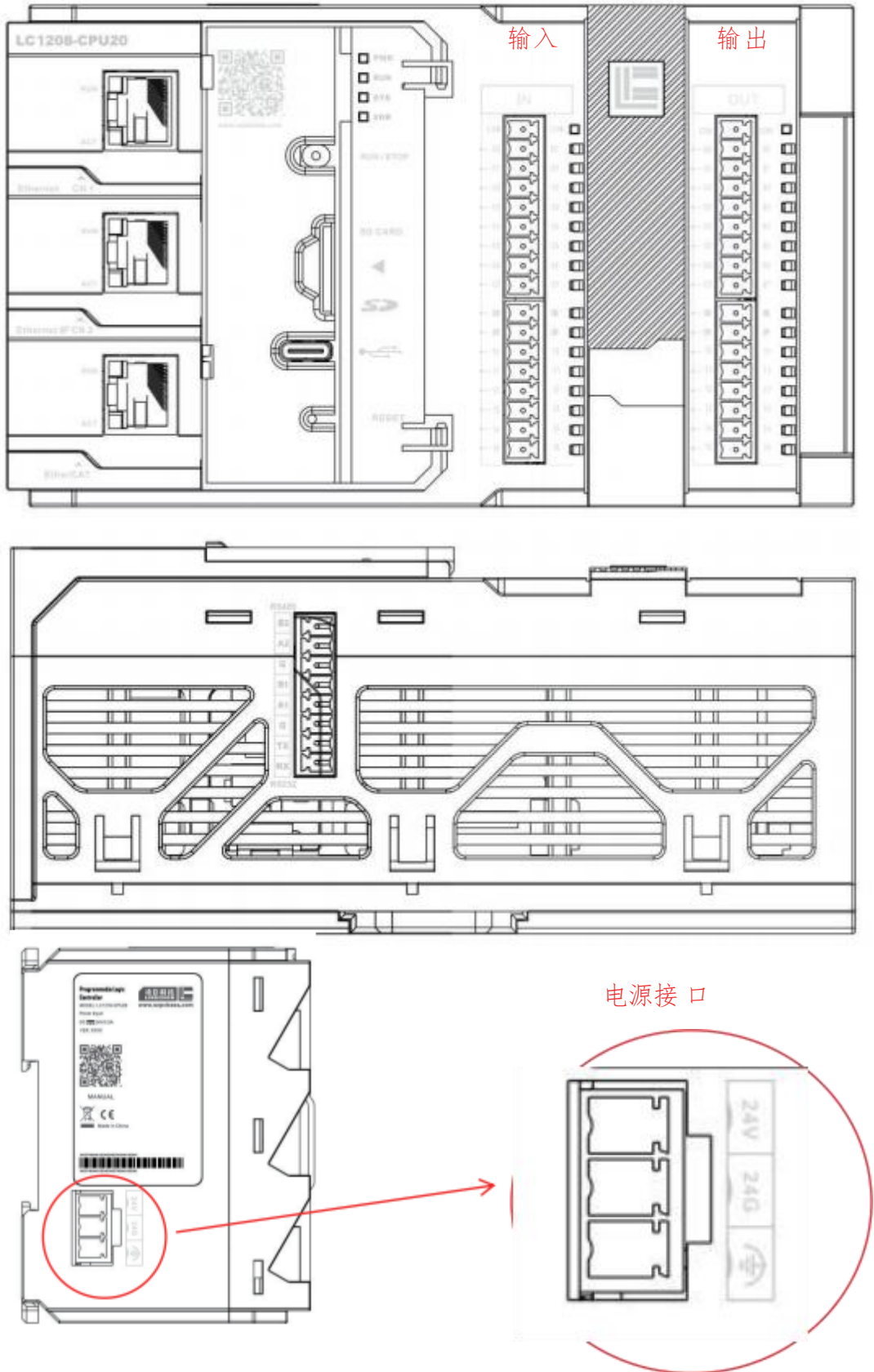
PLC/PCIe运动控制卡、远程IO模块、气动阀岛等产品，为我司的客户提供了更具性价比的方案解决。



## 目 录

|  |    |
|--|----|
| 硬件接口展示 .....                           | 1  |
| 参数简介 .....                             | 2  |
| 指示灯说明 .....                            | 2  |
| 1. 设备存储库的使用 .....                      | 3  |
| 2. 库文件的使用 .....                        | 5  |
| 3. 建立一个工程并下载调试 .....                   | 7  |
| 4. 单轴简单用户控制程序编写 .....                  | 20 |
| 5. 基于 EtherCAT 通信的电子凸轮程序例程 .....       | 23 |
| 6. 添加本地IO .....                        | 27 |
| 7. 添加拓展IO（以LC1100和1488、2488模块为例） ..... | 30 |
| 8. Modbus .....                        | 34 |
| 8.1 Modbus TCP Master .....            | 34 |
| 8.2 Modbus TCP Slave .....             | 41 |
| 8.3 Modbus RTU .....                   | 47 |
| 9. 断电保持功能的使用 .....                     | 50 |
| 10. OPC UA例程 .....                     | 52 |

硬件接口展示



## 参数简介

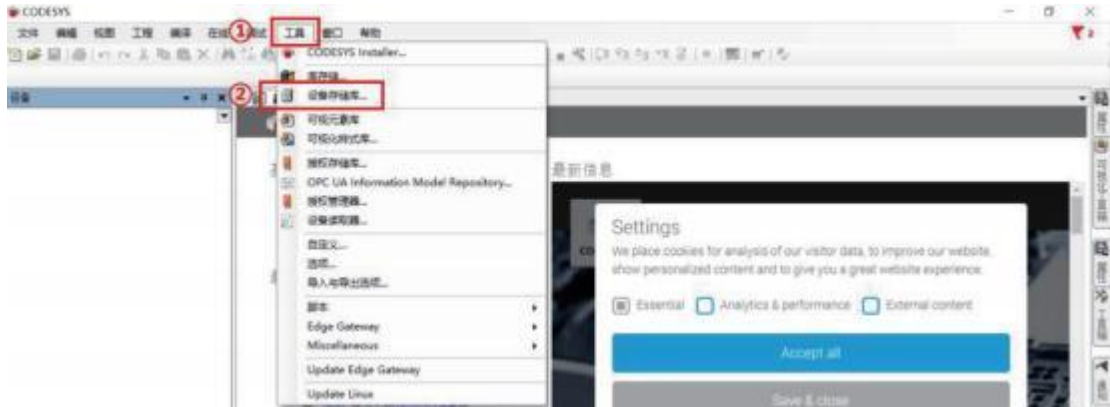
|        |                            |
|--------|----------------------------|
| Memory | DDR3L 1GB                  |
| EMMC   | 8GB                        |
| USB    | 1个USB插口 (TYPE-C)           |
| TF     | 1个SD卡插口                    |
| NET    | 2个EtherNET网口, 1个EtherCAT网口 |
| COM    | 2路RS485, 1路RS232           |
| INPUT  | GPIO* 16 输入高低电平检测          |
| OUTPUT | GPIO* 16 开漏控制输出            |
| RESET  | 1个复位按键                     |
| POWER  | 24V 1A                     |

## 指示灯说明

|     | 常亮     | 常灭     | 闪烁       | 颜色 |
|-----|--------|--------|----------|----|
| PWR | 有供电    | 无供电    |          | 绿灯 |
| RUN | run    | stop   |          | 绿灯 |
| SYS | 启动中/死机 | 启动中/死机 | PLC进程运行中 | 绿灯 |
| ERR | 运行错误   | 无错     | 总线错误     | 红灯 |

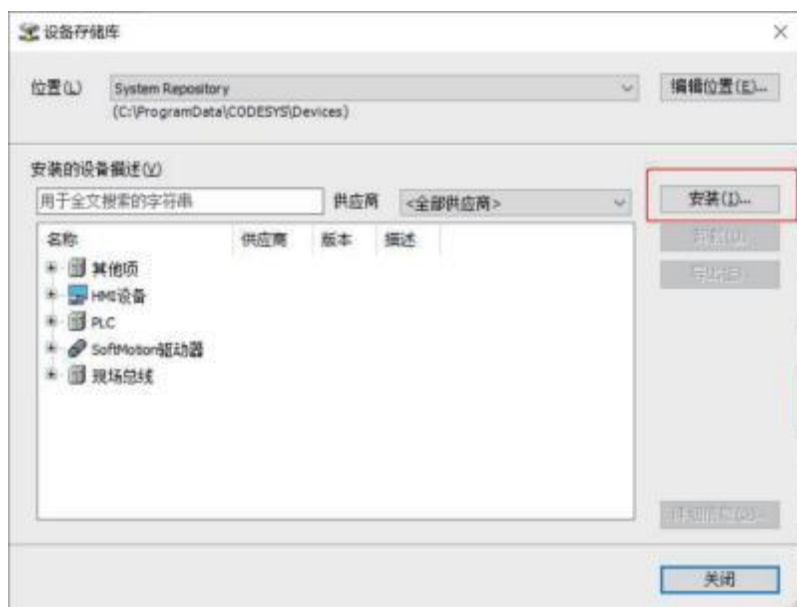
## 1. 设备存储库的使用

1.1 打开CODESYS V3.5 SP18软件，在界面最上方的菜单栏找到“工具”，点击“工具—设备存储库”；



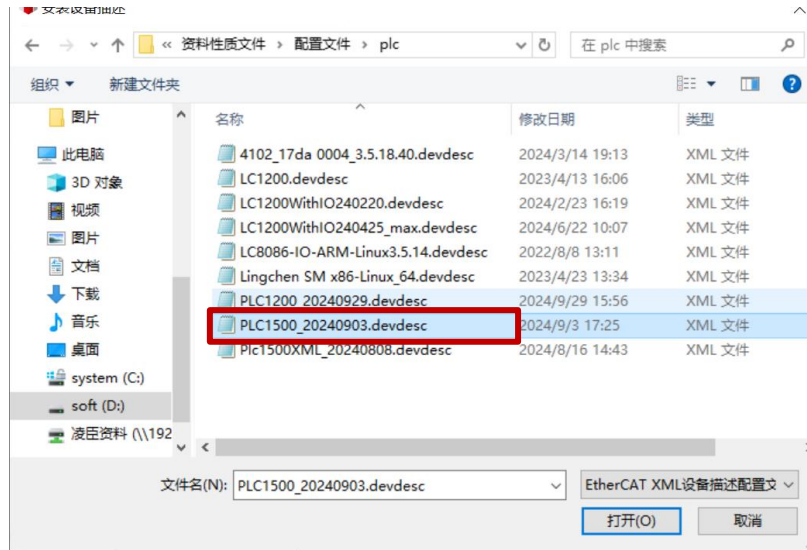
图\_1.1

1.2 点击“安装”；



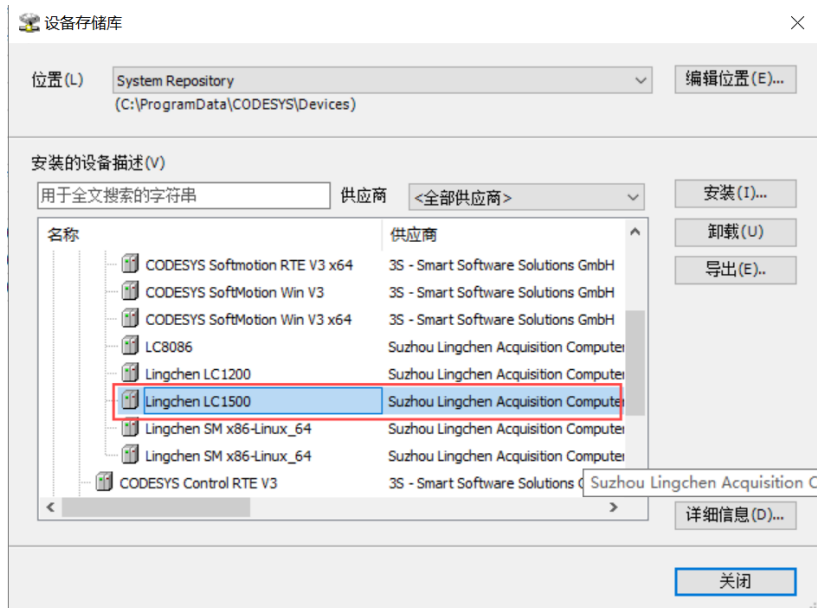
1.2

1.3 找到设备描述文件 (\*.xml) 所存放的位置，选择相应设备描述文件，点击“确定”（或者直接双击相应设备描述文件）；



### 1.3

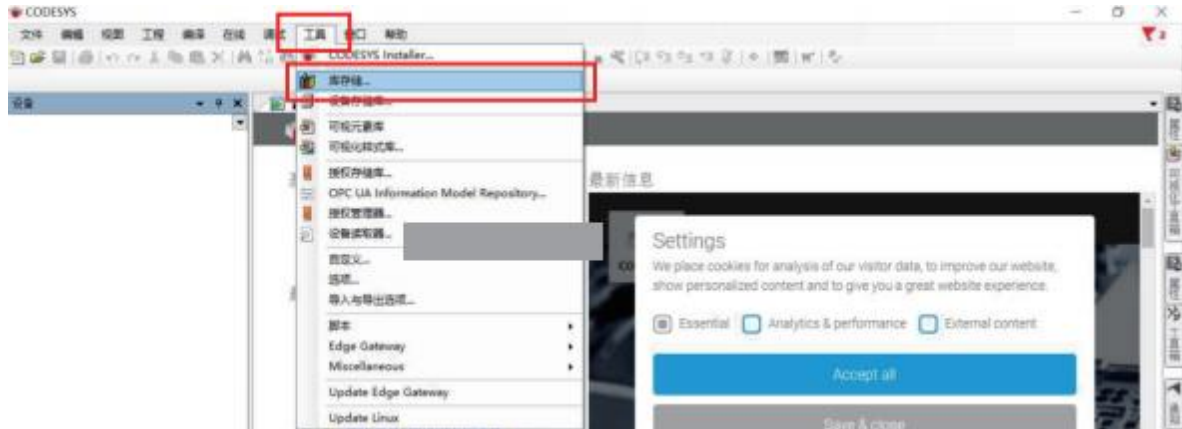
1.4 安装成功，点击“关闭”；



### 1.4

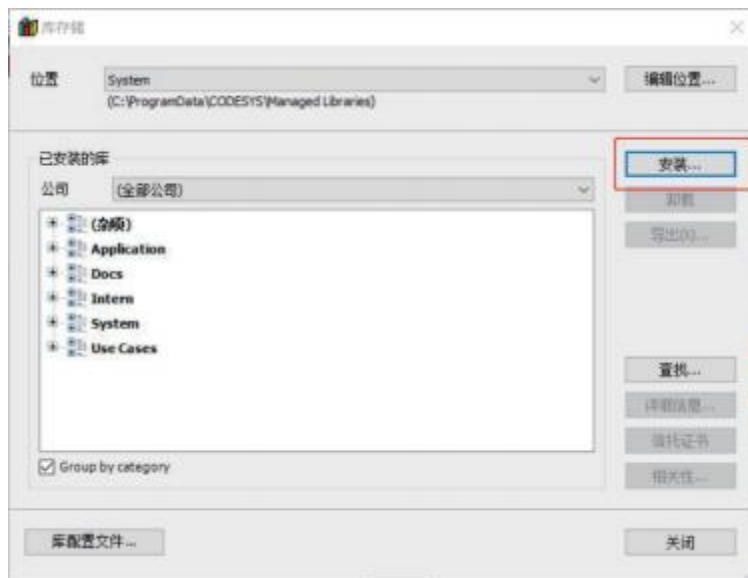
## 2. 库文件的使用

2.1 同样在菜单栏中找到“工具”，点击“工具—库存储”；



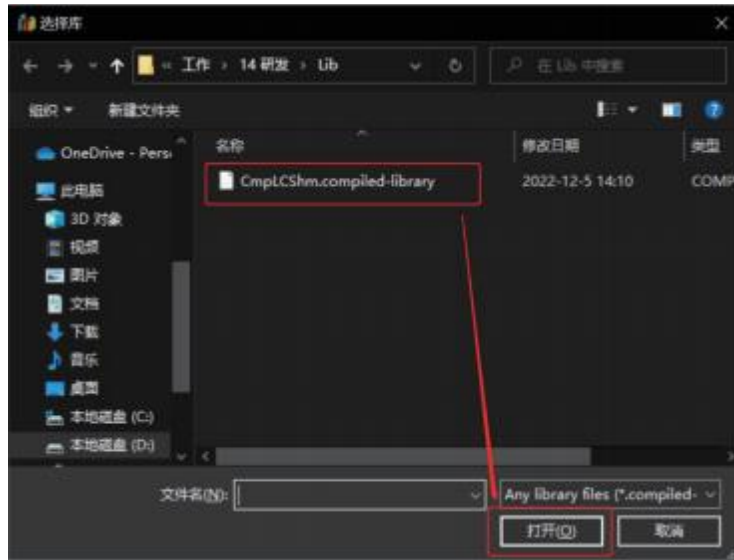
2.1

2.2 点击“安装”；



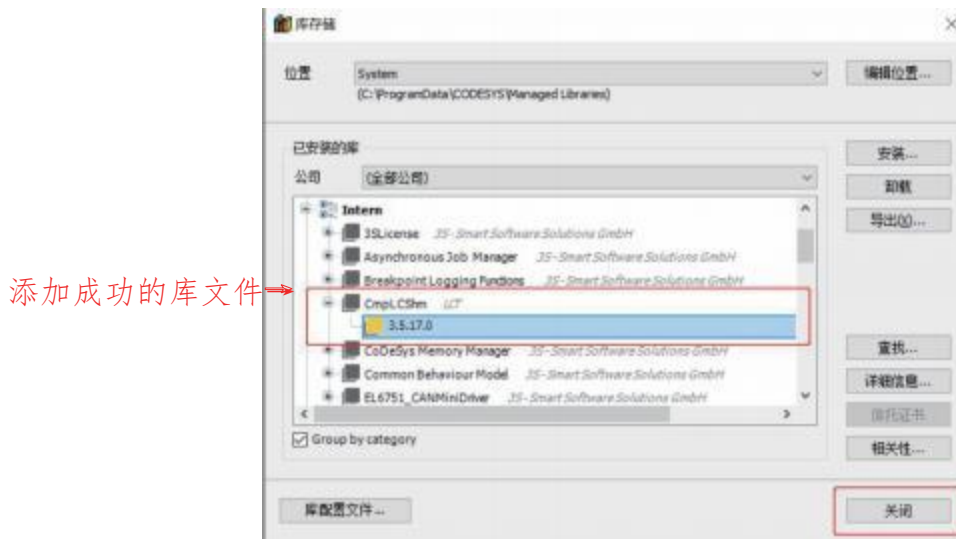
2.2

2.3 找到存放库文件的路径，选择要添加到存储的库文件，点击“打开”（或者直接双击要添加到存储的库文件）；



2.3

2.4 添加成功，点击“关闭”。



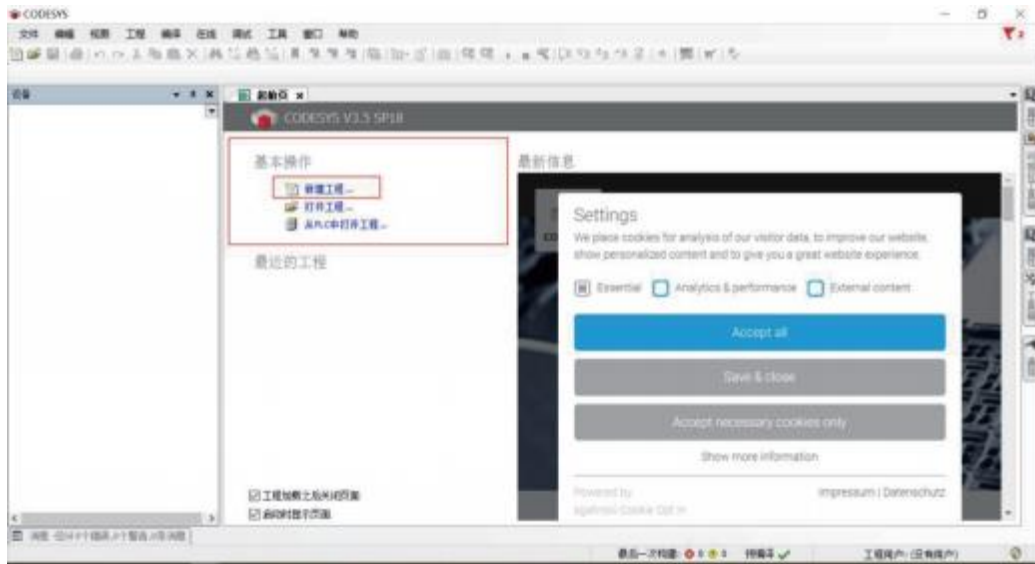
2.4

2.5 添加完成后，还需要安装到工程文件中，才能调用（详情见“3. 建立一个工程并下载调试”）。

### 3. 建立一个工程并下载调试

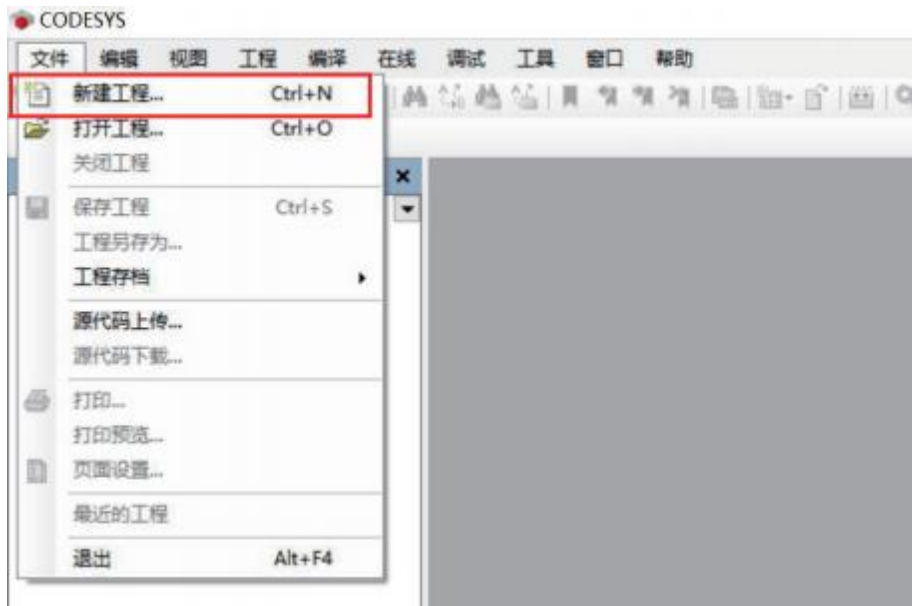
#### 3.1 创建一个新的PLC程序

方法1: 在“起始页”的“基本操作”区域, 点击“新建工程...”;



3.1

方法2: 若没有“起始页”, 可以在菜单栏点击“文件—新建工程...”;



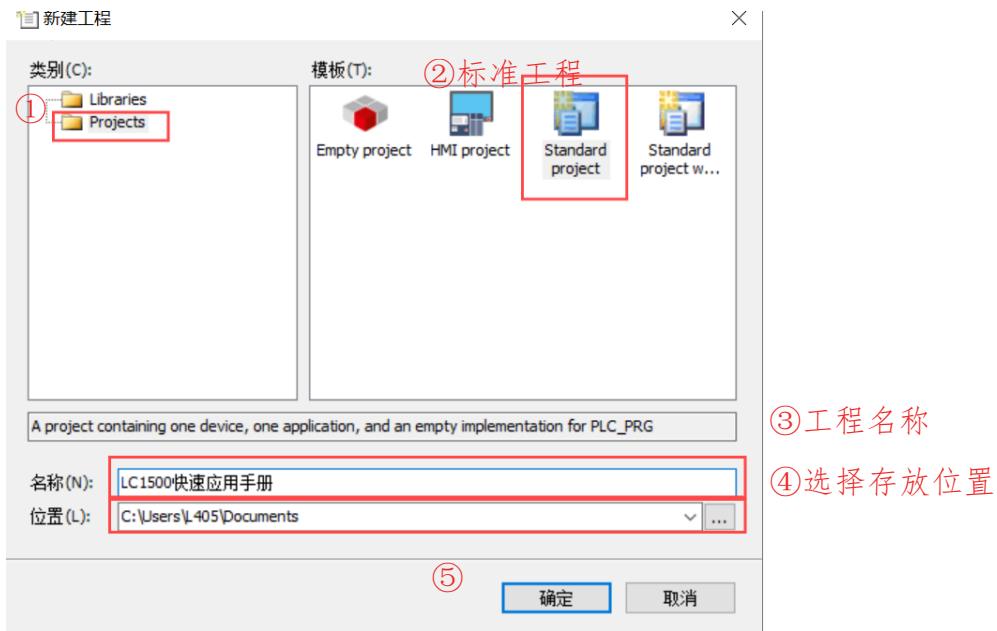
3.2

【注1: 若想显示“起始页”, 可以点击菜单栏“视图—起始页”。】

方法3: 使用快捷键Ctrl+N;

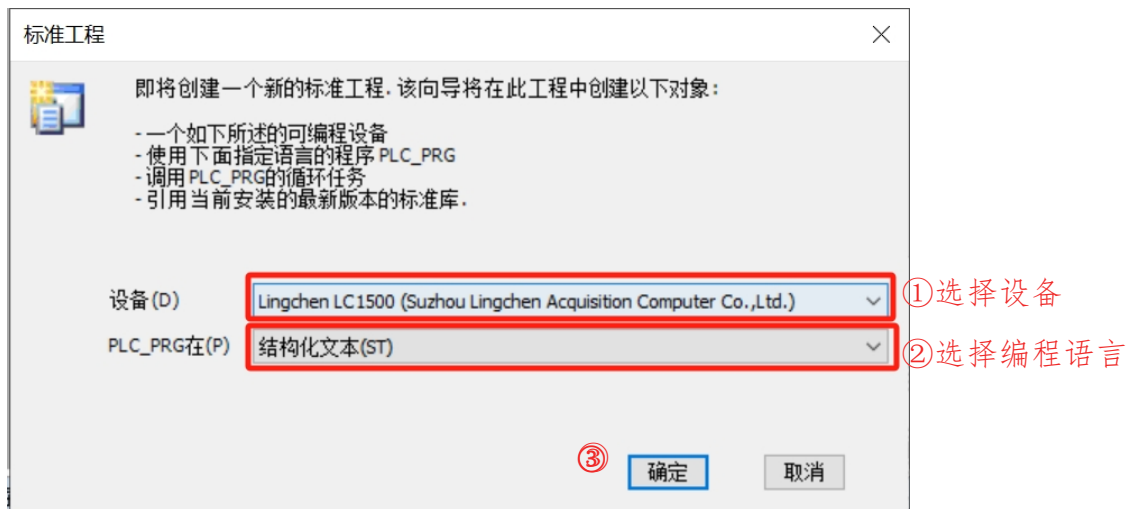
3.2 类别选择“工程”, 模板选择“标准工程”, 输入用户工程的名称, 选择存放位置, 点击“确定”;

LC1



### 3.3

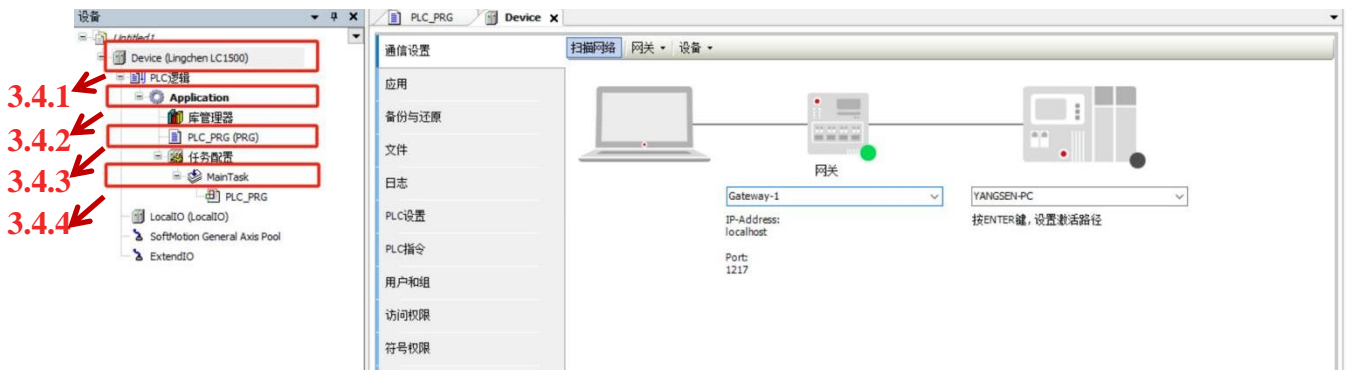
3.3 选择设备（Lingchen LC1216）与编程语言（后续新建POU可以更改编程语言）后，点击“确定”；



### 3.4

【注2：所有第一次使用的设备，都必须要在“设备存储库”中安装该设备（具体步骤参见本文档 1. 设备存储库 的使用）。】

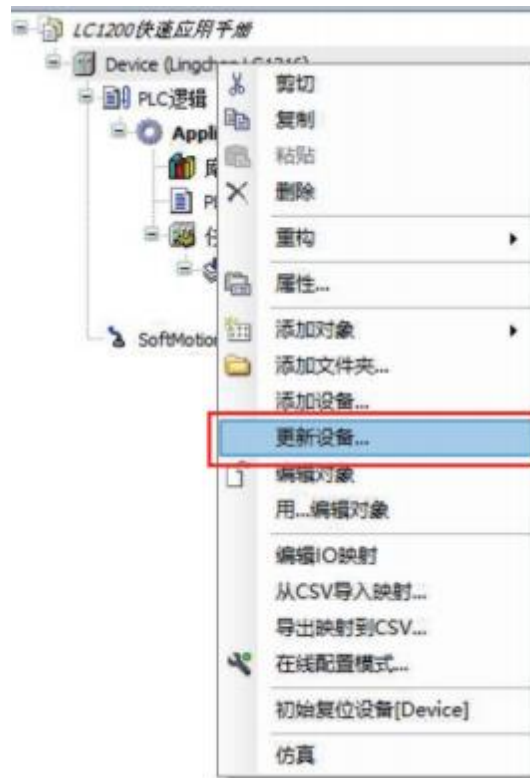
3.4 新建工程完成；



### 3.5

#### 3.4.1 Device 设备。

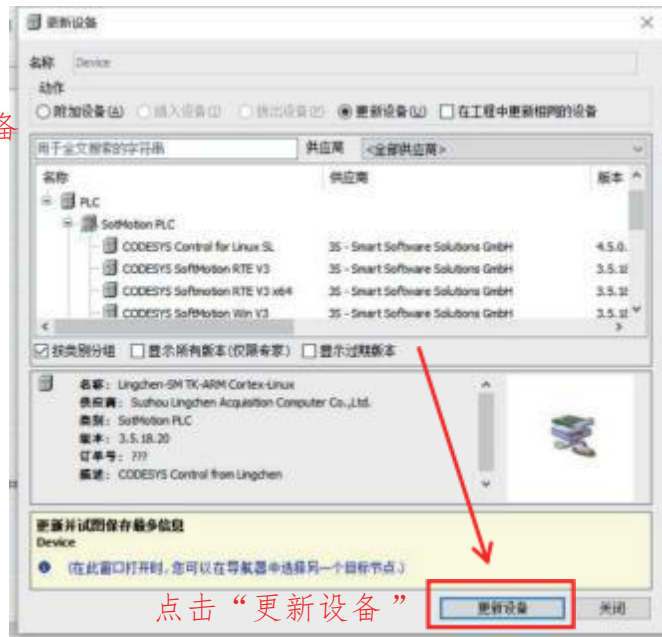
此处是新建工程时所选择的设备，可以通过“右键（Device）—更新设备”来更换用户工程中的设备；



### 3.6

选择要更换的设备后，点击“更新设备”（或者直接双击要更换的设备）。

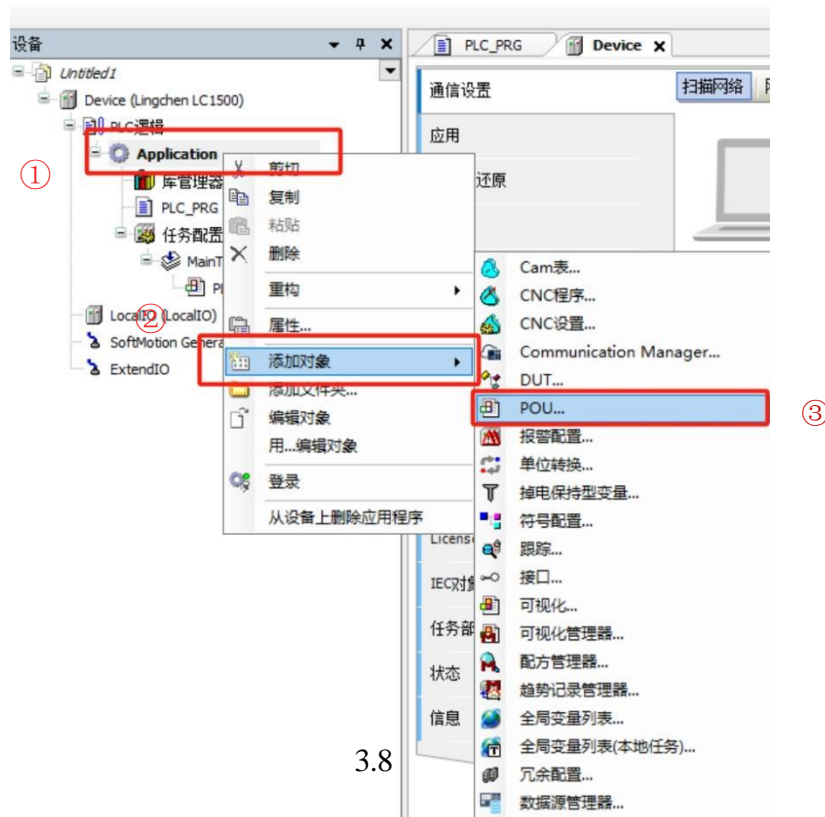
选择要更换的设备



3.7

### 3.4.2 Application 应用。

在此处添加用户程序单元（POU）等。具体操作方法为：“右键Application—添加对象—POU”，输入POU名称，选择“程序”，选择该用户程序单元所要使用的语言。

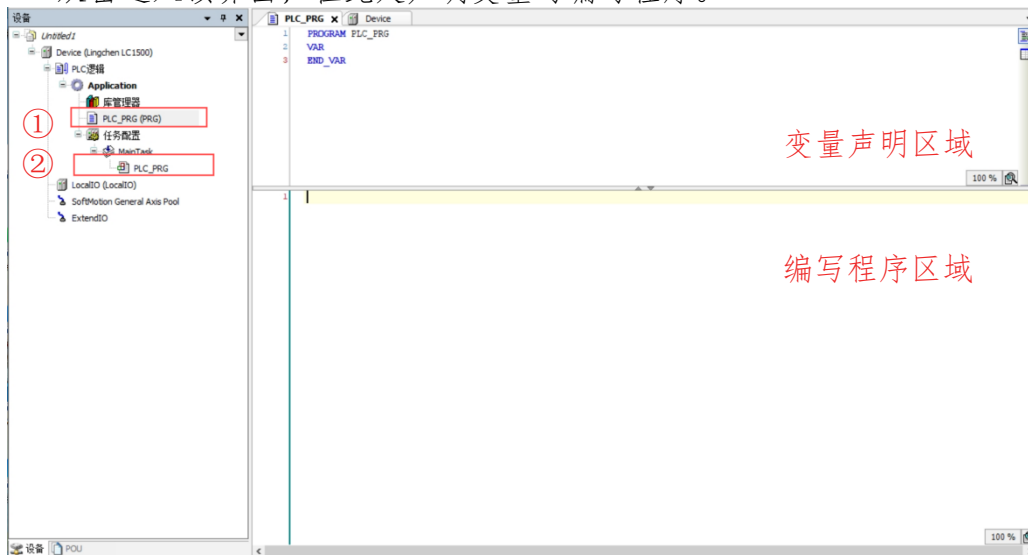




### 3.9

#### 3.4.3 PLC\_PRG(PRG) 用户程序。

双击进入该界面，在此处声明变量与编写程序。



### 3.10

【注3：要将写好的程序①拖入任务配置②下，方可编译调用。】

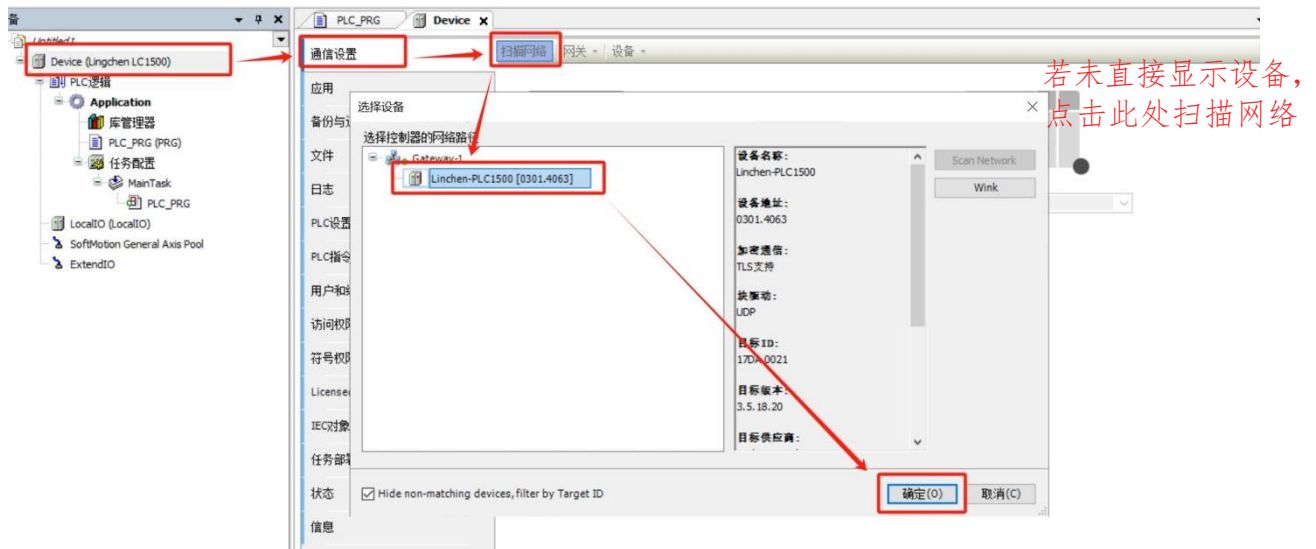
#### 3.4.5 任务配置及程序调用选择。

#### 3.5 编写程序；

### 3.6 系统配置及参数设定;

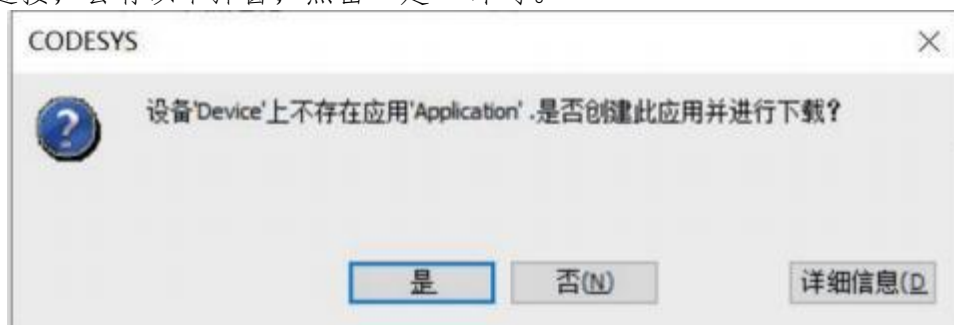
#### 3.6.1 连接设备

双击设备栏“Device”，在“通信设置”子页面内，点击“扫描网络”，选择设备“Lingchen-LC1500”后，点击“确定”。



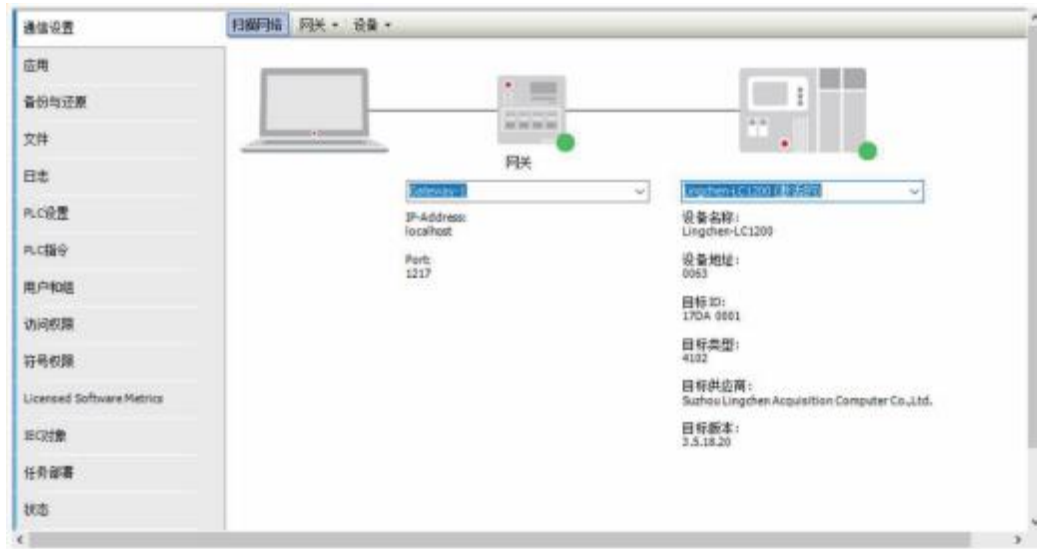
#### 3.11

初次连接，会有以下弹窗，点击“是”即可。



#### 3.

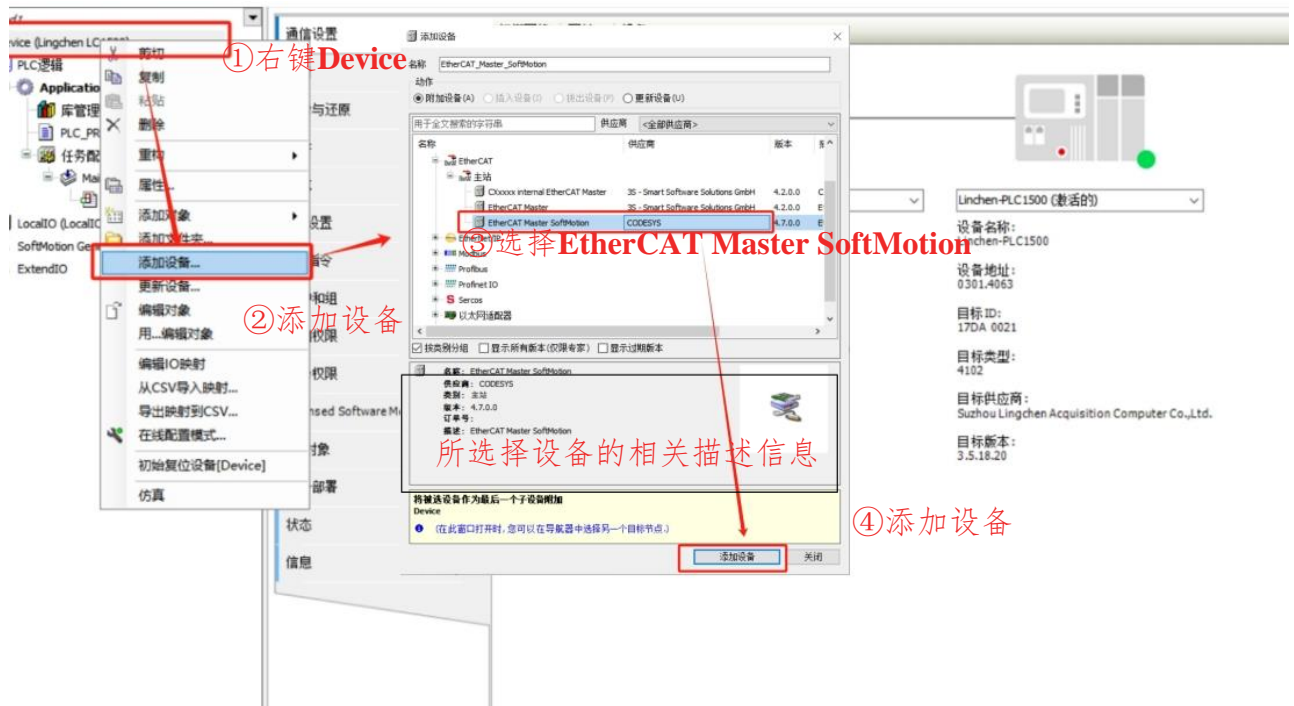
连接成功，如3.12。



3.12

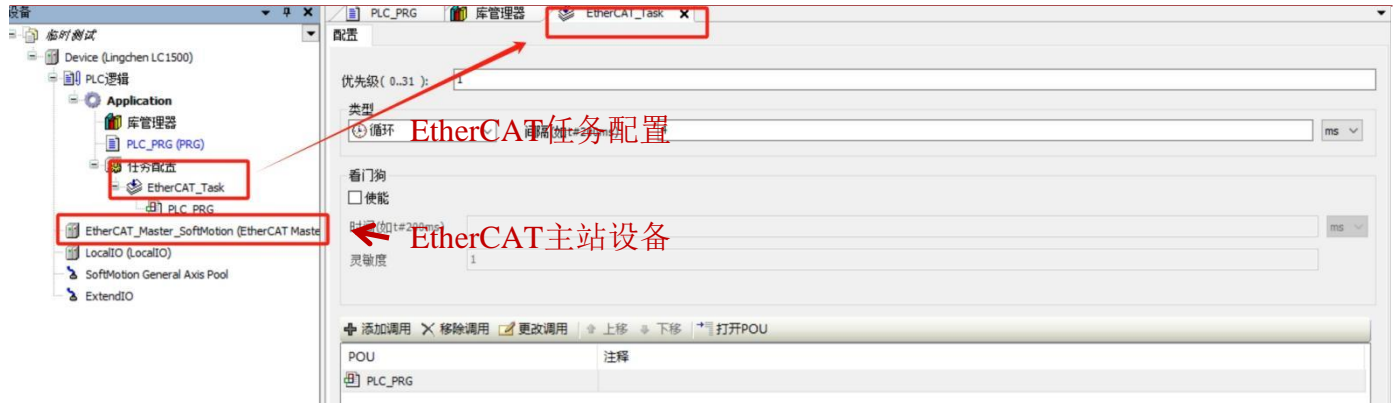
### 3.6.2 添加 EtherCAT\_Master\_SoftMotion

EtherCAT Master SoftMotion是带实时运动控制的EtherCAT主站模块。具体添加方法为：“右键Device—添加设备—现场总线—EtherCAT—主站—EtherCAT Master SoftMotion—添加设备”，添加 EtherCAT 主站。



3.13

添加后如3.14，同时系统会分配一个任务EtherCAT\_Task，可配置EtherCAT任务相关参数。



3.14

在EtherCAT\_Task配置界面，可以设定挂在该任务配置下的程序优先级、类型与扫描周期时间间隔等。

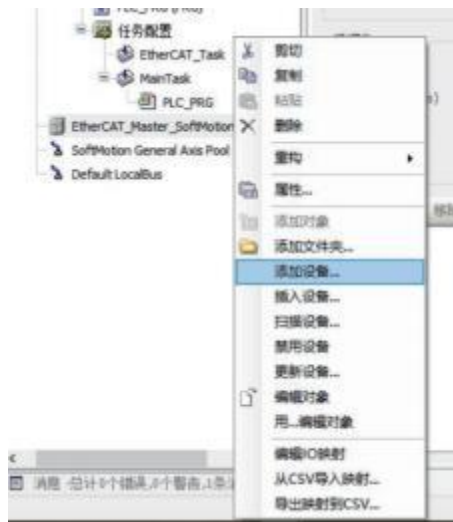
### 3.6.3 添加总线伺服

添加主站设备后，在该主站下方添加从站设备，此处添加设备为“LC\_SDE\_DC03A1（凌臣总线步进\_单轴）”。添加前必须先安装设备描述文件（.XML），安装流程参考本文档1.设备存储库的使用。

具体添加方式有两种：

第一种：手动添加

未连接主站的离线状态下，通过“右键EtherCAT Master Softmotion — 添加设备”，在“添加设备”弹窗中筛选对应供应商及设备型号“LC SDE-DC03A1”，选中该设备，点击“添加设备”（或直接双击要添加的设备）。



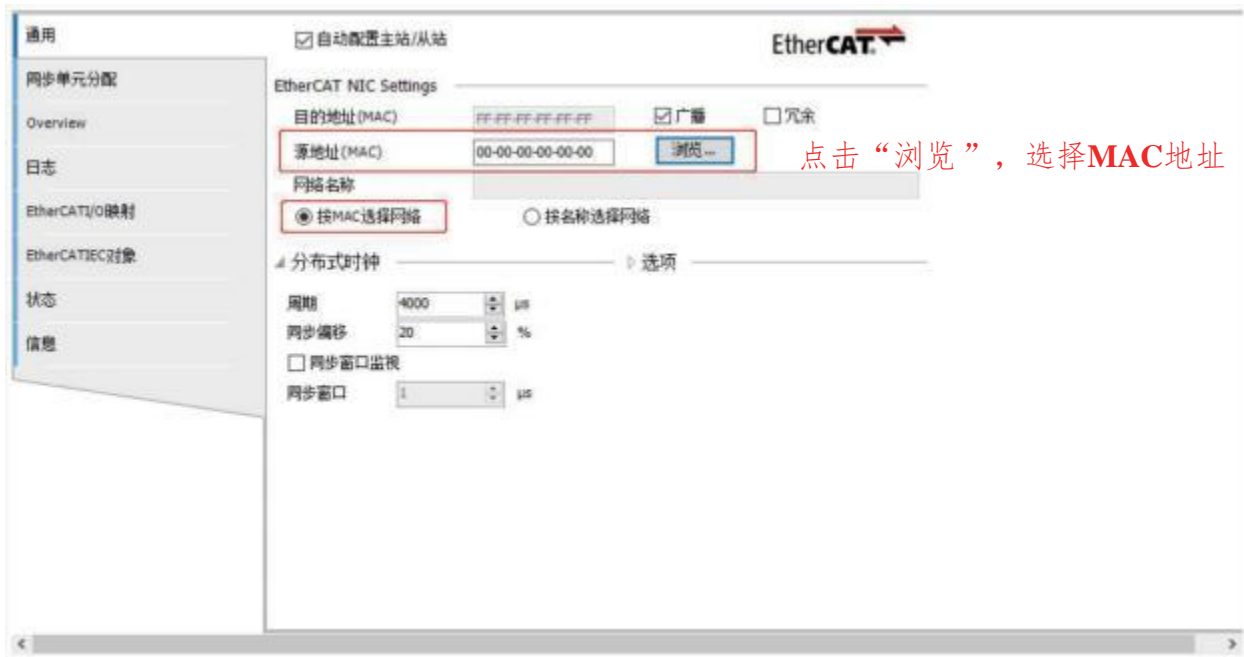
3.15



3.16

第二种：扫描添加

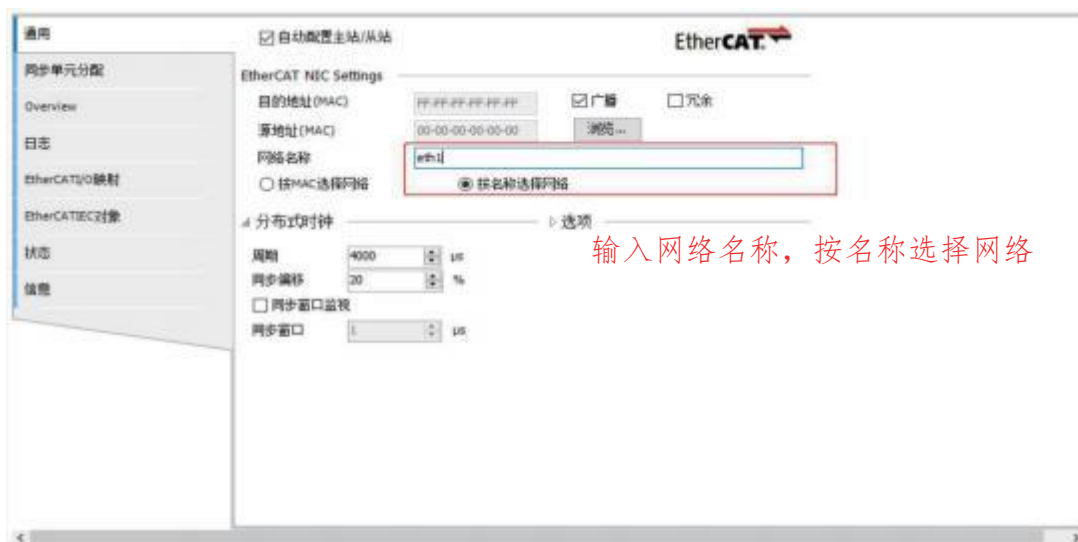
(1) “双击EtherCAT Master SoftMotion”打开主站设置界面，在“通用”子页面内，设置EtherCAT网口源地址，默认选项为“按MAC选择网络”时，可以通过点击“浏览”，选择EtherCAT网口MAC地址（3.17，3.18）；也可以更换为“按名称选择网络”选项，在“网络名称”输入框内，输入网口名称“eth1”（3.19）。



3.17



3.18



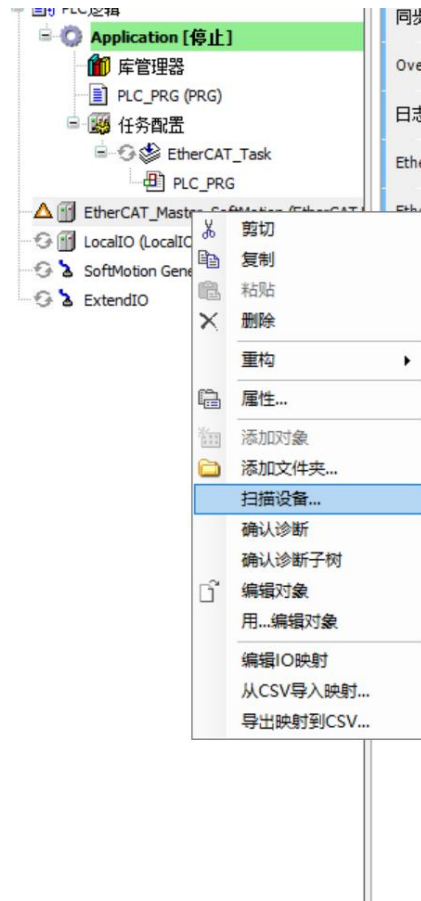
3.19

(2) 选定EtherCAT网口后，点击“登录”或“在线配置模式”。



3.20

(3) 登录成功后，“右键EtherCAT\_Master\_SoftMotion—扫描设备”。



3.21

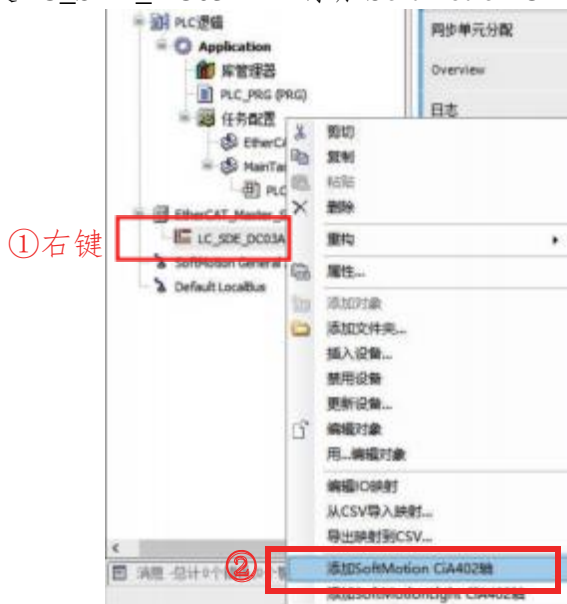


### 3.22

自动扫描到连接的设备后，点击“复制所有设备到工程”。

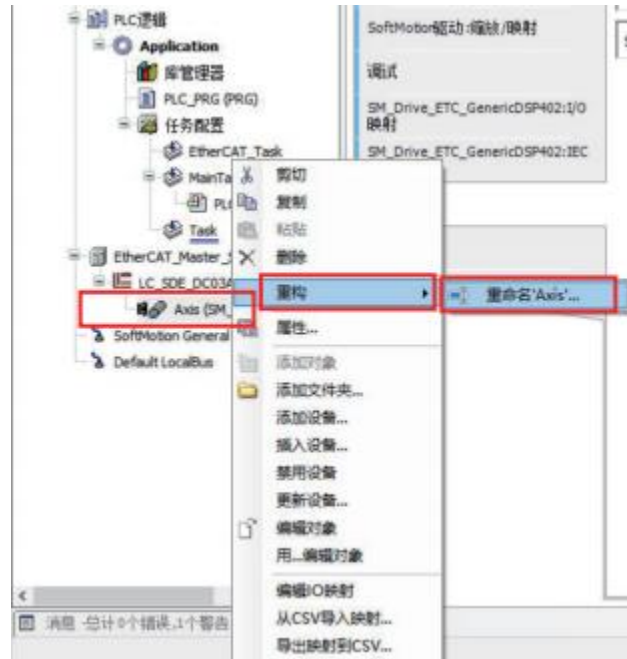
【注：设备的节点地址，默认是自动分配，即按照距主机距离从近到远来分配节点地址，本例不做修改，按默认设置。】

(4) 完成之后，退出登录，在新添加的设备处，添加运动控制轴，具体操作步骤为“右键LC\_SDE\_DC03A1—添加SoftMotion CiA402轴”。



### 3.23

(5) 添加完成，为方便编程，可以对新添加的轴重命名。



3.24

(6) 双击新添加的SoftMotion CiA402轴，设置控制相关参数，在“SoftMotion 驱动：缩放/映射”子页面内调整电机类型、比例缩放与映射等。



3.24

## 4. 单轴简单用户控制程序编写

此处编写一个单轴的简单程序，使控制器控制伺服电机执行绝对位置指令，做往返运动。

(1) 首先建立对象（POU）（详情参考3.4.2）。

(2) 打开编程界面POU\_Abs（详情参照3.4.3，3.10），在变量声明区域添加变量，变量声明代码如下：

```
PROGRAM POU_Abs
```

```
VAR
```

```
  iStatus: INT; // 执行步骤
```

```
  Power: MC_Power; // 使能模块
```

```
  ActPos: LREAL; // 实际位置值
```

```
  MoveAbsolute: MC_MoveAbsolute; // 绝对位移模块
```

```
  p: REAL:=30; // 位移值
```

```
END_VAR
```

【注：声明变量时一般都是默认添

加库文件“SM3\_Basic”，如若未添加则需要手动通过双击“库管理器—添加库”，找到库“SM3\_Basic”然后选择添加，也可以通过此方法添加更多的库。】



图3-

(3) 在编程区域添加程序如下。（程序功能：程序执行时，立即使能伺服，等伺服使能成功后，控制电机在位置P与起点O之间做往返运动。）

```
CASE iStatus OF
```

```
  0: // 启动后，轴Axis上使能
```

```
    Power(Axis:=Axis, Enable:=TRUE, bRegulatorOn:=TRUE,
```

```
    bDriveStart:=TRUE );
```

```

IF Power.Status THEN
    iStatus:=iStatus+1;

END_IF

1: //走绝对位移, 运行到P 处
    MoveAbsolute(Axis:=Axis, Execute:=TRUE, Position:= 0, Velocity:=100 ,
Acceleration:= 100, Deceleration:=100 );

    IF MoveAbsolute.Done THEN
        MoveAbsolute(Axis:=Axis, Execute:= FALSE); iStatus:=iStatus+1;
    END_IF

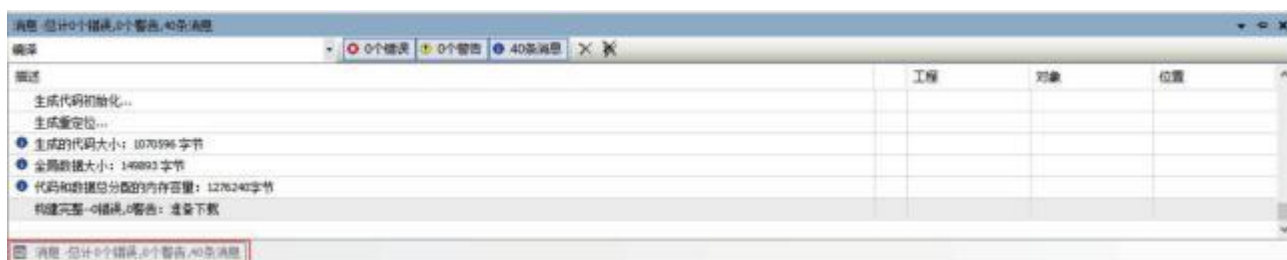
2: //走绝对位移, 运行回到O 处
    MoveAbsolute(Axis:=Axis, Execute:=TRUE, Position:= p, Velocity:=100 ,
Acceleration:= 100, Deceleration:=100 );

    IF MoveAbsolute.Done THEN
        MoveAbsolute(Axis:=Axis, Execute:= FALSE);
        iStatus:=1;
    END_IF

END_CASE

ActPos:= Axis.fActPosition;//获取轴实际位置值
    
```

(4) 程序编写完成后，点击“编译”，确认编写没有错误。



### 3.

在左下角的“消息”提示框内会显示编译过程描述，如果有错误，也会显示在该区域。

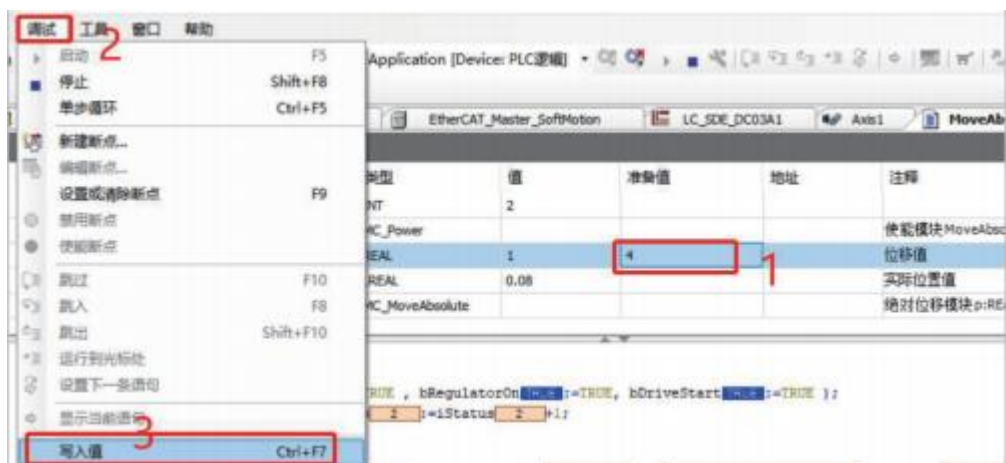
(5) 将编写完成的对象（POU\_Abs）拖进任务节点下。

(6) 连接设备（参照3.6.1），无误后，点击“登录—启动”，打开“POU\_Abs”，程序运行如下图所示，程序执行伺服使能后，可以看到电机在位置 p与起点位置 O之间做往返运动。



3.

在线修改位置 P 的值：单击变量“P”的预设值“准备值”，使其为然后输入数值“10”，然后选择“调试 写入值”或者快捷键“Ctrl+F7”将值写入到“值”中，即可在线修改变量“P”的值。



3.

## 5. 基于 EtherCAT 通信的电子凸轮程序例程

参照上面的操作步骤，新建工程，添加EtherCAT Master SoftMotion主站，更改EtherCAT网口后，“登录—右键EtherCAT Master SoftMotion—扫描设备”，扫描到设备如下（本例程以两套“LC\_SDE\_DC03A1（凌臣总线步进\_单轴）”为例），“复制所有设备到工程”，添加完毕并为每个步进添加与一个402轴，为方便编程示范分别重命名为x，y，如下图（3.25）所示（详情请参考3.6.4 单轴简单用户控制程序编写）。

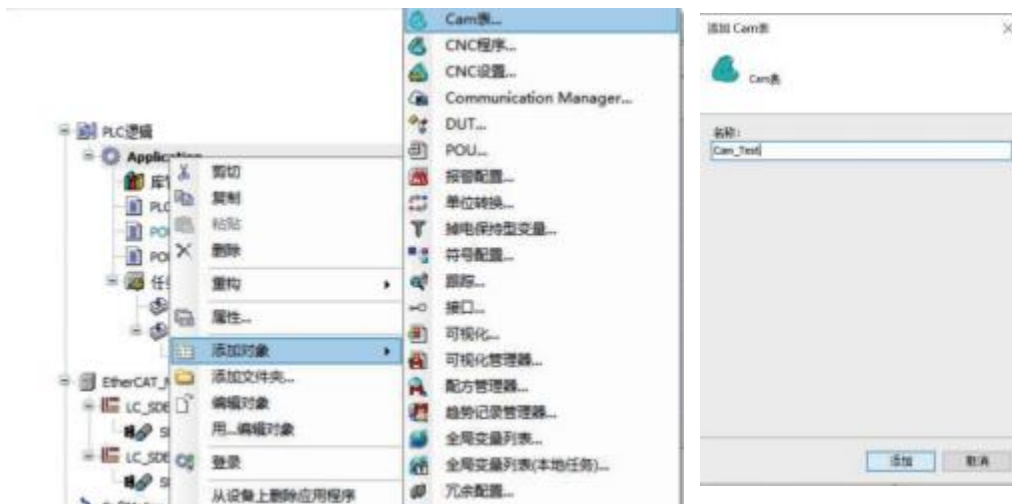


3.

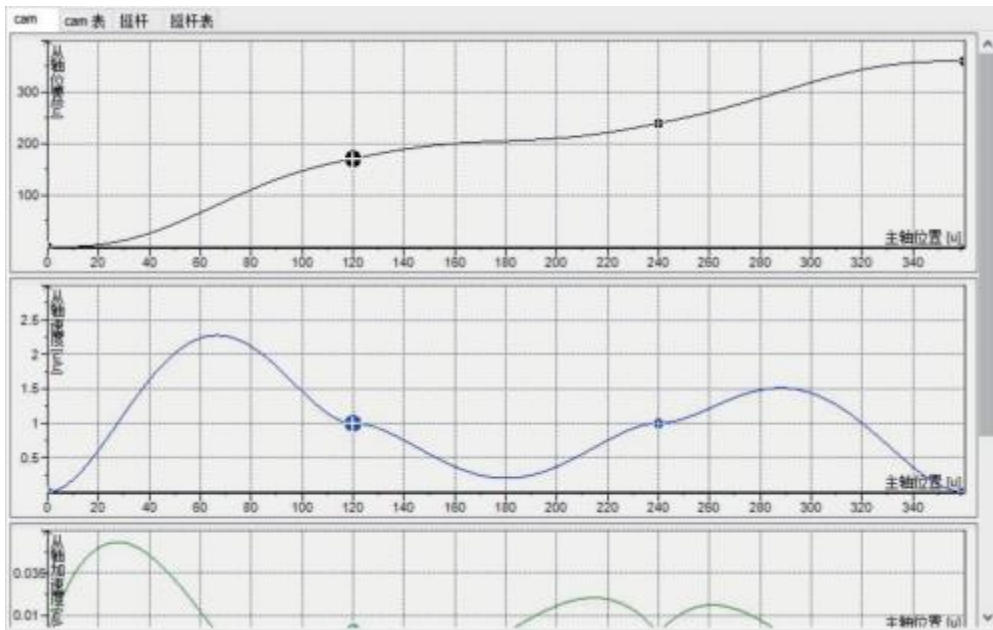


3.

在Application下添加一个对象（POU\_Cam），再添加一个对象（Cam表），具体操作步骤为“右键Application—添加对象—Cam表...”，自定义Cam表的名称，



点击“添加”。添加完成后，Cam表界面如图所示。



可以对相关项进行自定义设置（本文档仅做示范作用，因此使用默认设置）。

打开编程界面POU\_Cam，在变量声明区域添加变量，变量声明代码如下：

**PROGRAM POU\_Cam**

**VAR**

```

xstart      :BOOL;//启动开关
power_x     :MC_Power;//轴x_使能模块
power_y     :MC_Power;//轴y_使能模块
mov         :MC_MoveRelative;//相对运动模块
SEL1        :MC_CamTableSelect;//Cam表选择
camin       :MC_CamIn;
camout      :MC_CamOut;
t           :TON;
step        :INT;//执行步骤
    
```

**END\_VAR**

在编程声明区域添加程序，程序如下：（程序功能：程序执行时，通过xstart开启运动，等伺服使能成功后，控制轴x与轴y进行电子凸轮运动。）

```
power_x( Axis:= x , Enable:= xstart , bRegulatorOn:= 1 , bDriveStart:= 1 );
```

```
power_y( Axis:= y , Enable:= xstart , bRegulatorOn:= 1 , bDriveStart:= 1 );
```

```
mov( Axis:= x , Execute:= , Distance:=10 , Velocity:= 10, Acceleration:= 100,
Deceleration:=100 , Jerk:= 1000 );
```

```
SEL1( Master:= x , Slave:= y , CamTable:=Cam_Test , Execute:= , Periodic:= 1,
MasterAbsolute:=0 , SlaveAbsolute:= 0 );
```

```
camin( Master:= x, Slave:= y, Execute:= , MasterScaling:=1 , SlaveScaling:= 1,
StartMode:= absolute,
```

```
CamTableID:=SEL1.CamTableID , VelocityDiff:= , Acceleration:=100 ,
Deceleration:=100 , Jerk:=100 );
```

```
camout( Slave:= y );
```

```
t(IN:= , PT:= T#1S, Q=> , ET=> );//设定延时时间为1s
```

```
CASE step OF
```

```
0://等待轴使能
```

```
    IF power_x.Status THEN
```

```
        step:= 100;
```

```
    END_IF
```

```
100://选择Cam表, 开启Cam表: Cam_test
```

```
    SEL1.Execute:= 1;
```

```
    step:=200;
```

```
200:
```

```
    camin.Execute:= 1;//进入
```

```
    camout.Execute:=0;
```

```
    step:=500;
```

```
500:
```

```
    mov.Execute:= 1;//开启相对运动
```

```
    camin.Execute:=0;
```

```
    step:=550;
```

```
550://运动到位时跳出
```

```
    IF mov.Done THEN
```

```
        camout.Execute:= 1;
```

```
        step:=600;
```

```

END_IF

600:
mov.Execute:=0;
t.IN:=1;
IF t.Q THEN
    t.IN:=0;
    step:=200;
END_IF

END_CASE

```

程序编写结束后，点击“编译”，确认无误，将POU\_Cam放到“EtherCAT\_Task”下。点击“登录”，登录成功后，点击“启动”，等驱动和轴前的图标变为绿色后，将“xstart”的值改为“true”，程序运行如下图所示。



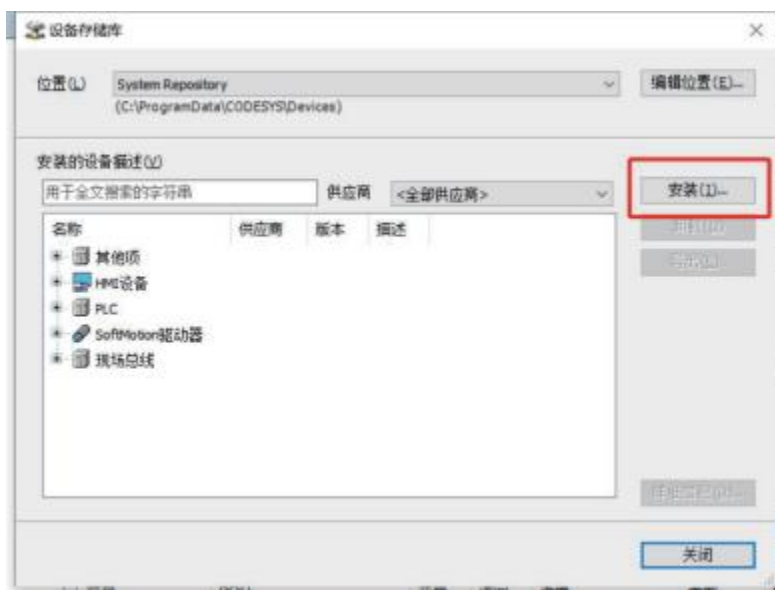
3.

## 6. 添加本地IO

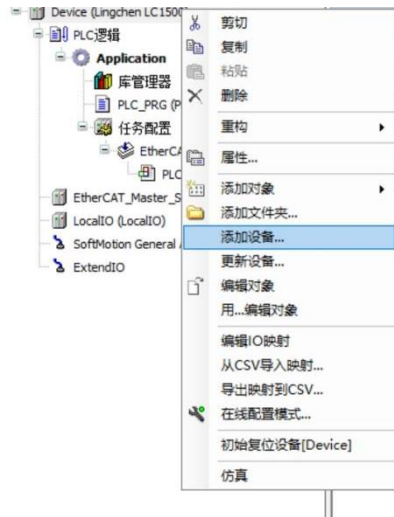
1. 创建一个标准工程，设备选择Lingchen LC1500;
2. 在菜单工具栏，点击“工具”，选择“设备存储库”，如下图所示;



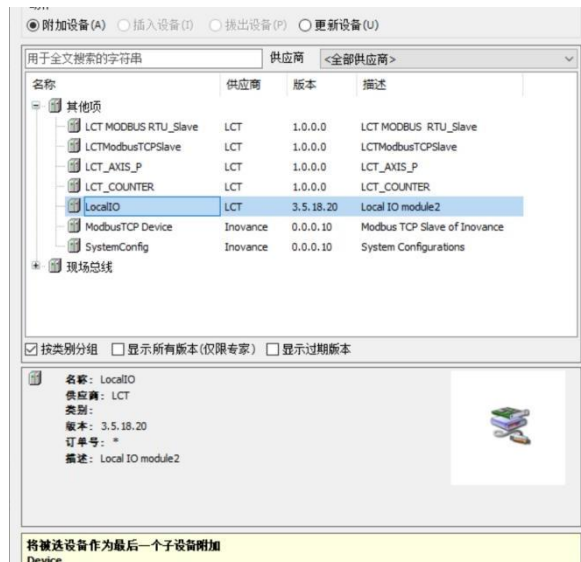
3. 在弹出的“设备存储库”对话框内，选择“安装”，选择已经存好的对应设备xml文件;



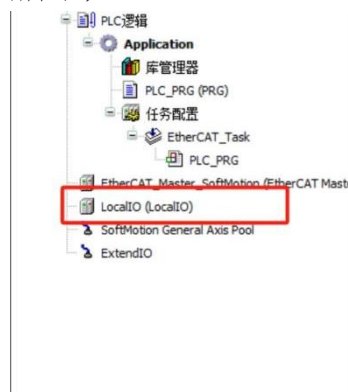
4. 安装完成后，在设备栏中右键“Device”，选择“添加设备”，如下图所示:



5. 在这里就可以看到前面已经安装完成的设备“LocalIO”，双击或选中后点右下角的“添加设备”，即可完成添加；



6. 添加完成后，如下图所示；



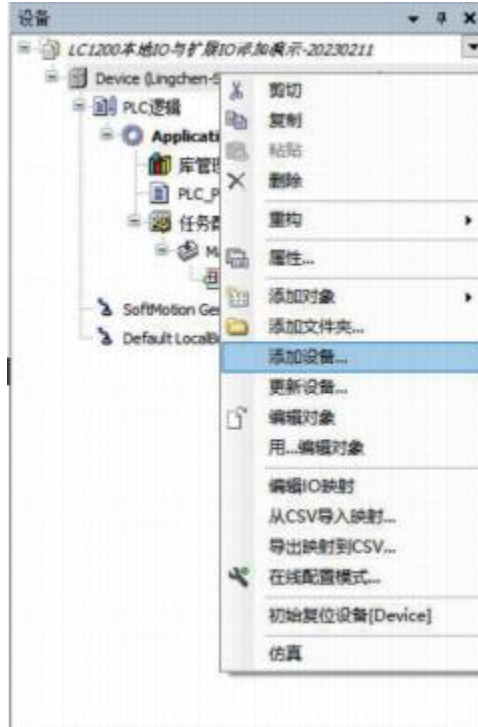
7. 登录后即可在LocalIO的“I/O映射”子界面进行查看。

The screenshot shows the 'LocalIO (LocalIO)' configuration window. On the left is a tree view with 'Application (运行)' selected. The main area is divided into 'I/O映射' (I/O Mapping), '状态' (Status), and '信息' (Information) tabs. The 'I/O Mapping' tab displays a table of digital I/O points.

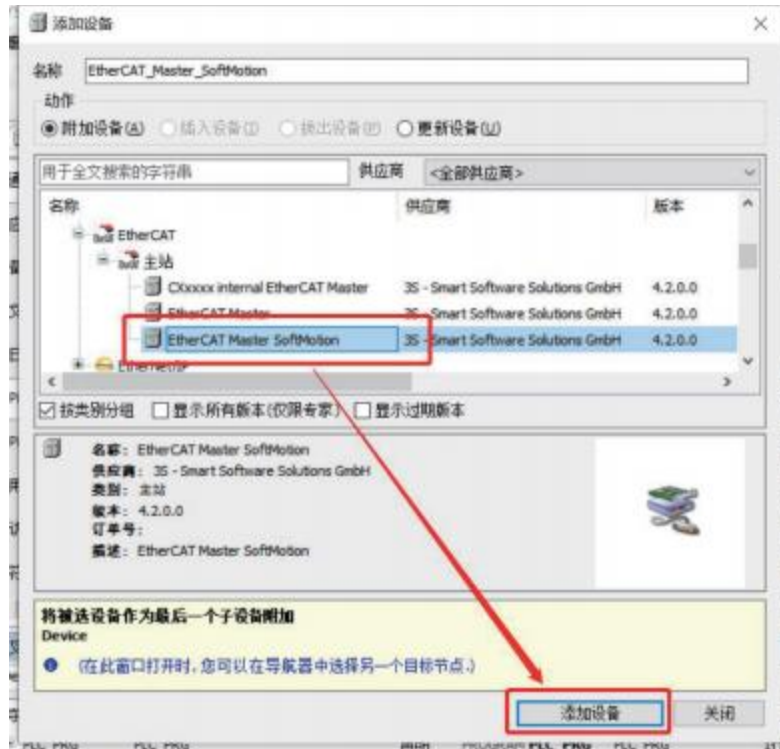
| 变量 | 映射 | 通道            | 地址     | 类型    | 当前值   | 预备值 | 单元 | 描述            |
|----|----|---------------|--------|-------|-------|-----|----|---------------|
|    |    | DigitalInput  | %IW0   | UBINT | 0     |     |    | DigitalInput  |
|    |    | DigitalOutput | %QW0   | UBINT | 65535 |     |    | DigitalOutput |
|    |    | Bit0          | %QX0.0 | BOOL  | TRUE  |     |    |               |
|    |    | Bit1          | %QX0.1 | BOOL  | TRUE  |     |    |               |
|    |    | Bit2          | %QX0.2 | BOOL  | TRUE  |     |    |               |
|    |    | Bit3          | %QX0.3 | BOOL  | TRUE  |     |    |               |
|    |    | Bit4          | %QX0.4 | BOOL  | TRUE  |     |    |               |
|    |    | Bit5          | %QX0.5 | BOOL  | TRUE  |     |    |               |
|    |    | Bit6          | %QX0.6 | BOOL  | TRUE  |     |    |               |
|    |    | Bit7          | %QX0.7 | BOOL  | TRUE  |     |    |               |
|    |    | Bit8          | %QX1.0 | BOOL  | TRUE  |     |    |               |
|    |    | Bit9          | %QX1.1 | BOOL  | TRUE  |     |    |               |
|    |    | Bit10         | %QX1.2 | BOOL  | TRUE  |     |    |               |
|    |    | Bit11         | %QX1.3 | BOOL  | TRUE  |     |    |               |
|    |    | Bit12         | %QX1.4 | BOOL  | TRUE  |     |    |               |
|    |    | Bit13         | %QX1.5 | BOOL  | TRUE  |     |    |               |
|    |    | Bit14         | %QX1.6 | BOOL  | TRUE  |     |    |               |
|    |    | Bit15         | %QX1.7 | BOOL  | TRUE  |     |    |               |

## 7. 添加拓展IO（以LC1100和1488、2488模块为例）

1. 同样创建标准工程后，在工具-设备存储库中导入设备文件（一次导入后，下次不用重复操作）；
2. 在设备栏，右键“Device”，选择“添加设备”；



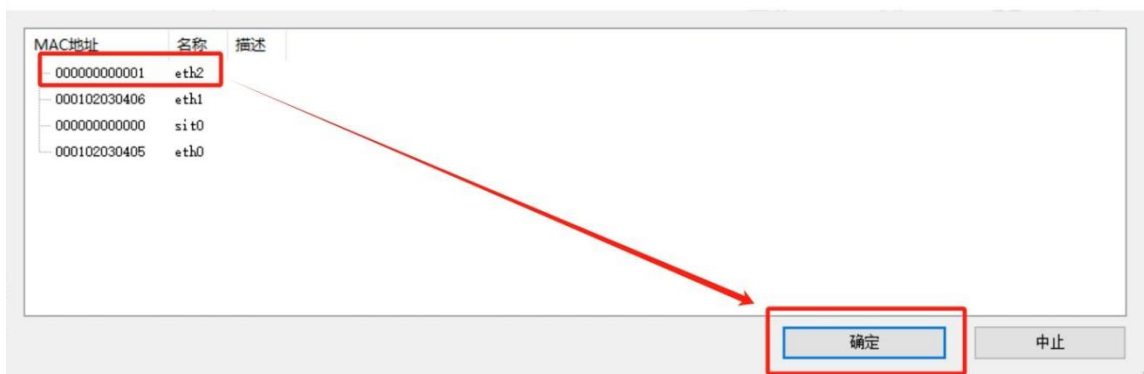
3. 添加主站；



4. 选中刚刚添加的设备，在“通用”界面选择“源地址”，选择准备连接的网口源地址；



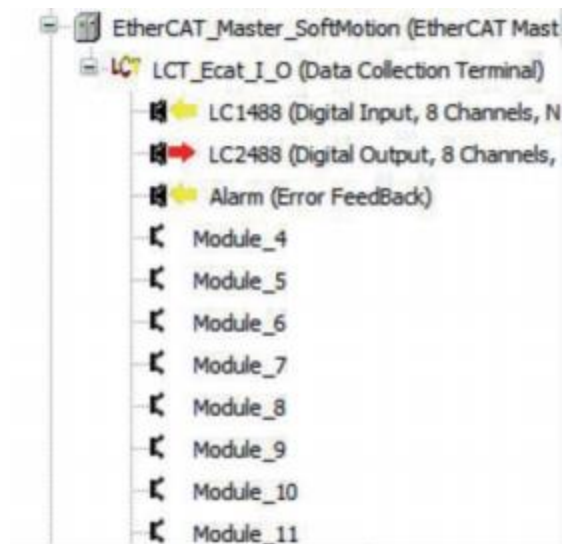
选择网络适配器



5. 选好后，“登录”或“在线”，右键主站（EtherCAT\_Master\_SoftMotion），“扫描设备”，在弹出的对话框内确认设备是否全部正常后，“复制所有设备到工程”；



复制完成后，在设备栏展示如下：



6. “登录—启动”，在“设备栏”双击LC2488（输出模块）打开界面，点击“ModuleI/O映射”子界面，可以在此为输出赋值，同理，在LC1488（输入模块）界面的“ModuleI/O映射”子界面，监测输入当前值。



## 8. Modbus

### 8.1 Modbus TCP Master

配合使用软件：Modbus Slave

具体操作步骤：

关闭电脑防火墙。

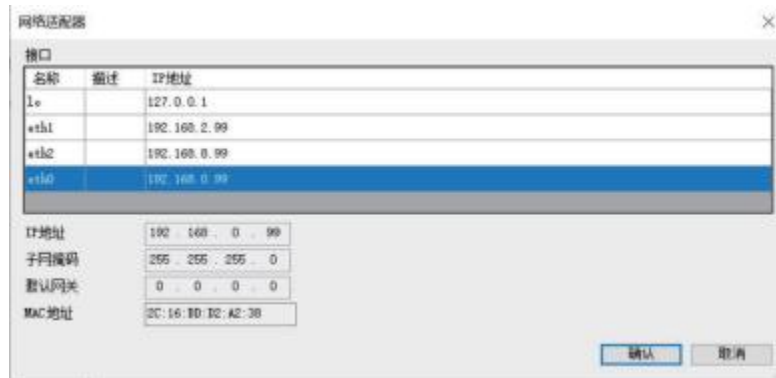
- (1) 新建标准工程文件；
- (2) 右键“Device”—添加设备；
- (3) 点开“以太网适配器”子选项，选中“Ethernet”，点击“添加设备”；



- (4) 添加完成后，双击打开Ethernet设备界面，在“通用”子界面，点击“Browse...”选择网络接口（选择网络接口前需要先连接设备Device）；



(5) 选中“eth0”（也可以自定义），点击确认；



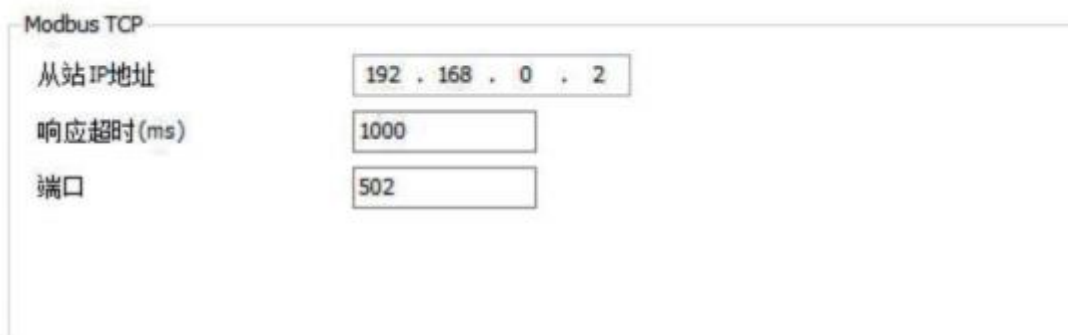
(6) “右键Ethernet—添加设备—按照图中所示位置，找到Modbus TCP Master选中—添加设备—关闭”；



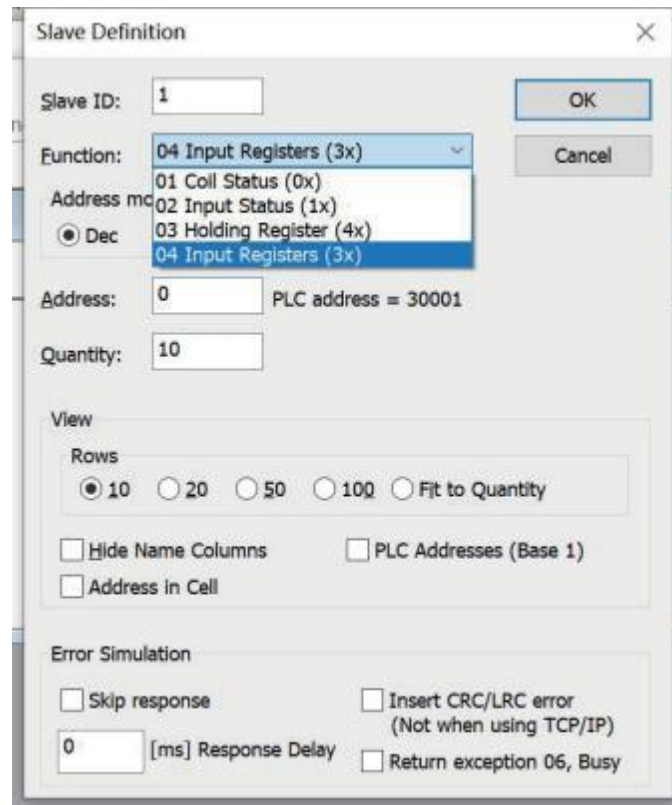
(7) “右键Modbus TCP Master—添加设备—按照图中所示位置，找到Modbus TCP slave选中—添加设备—关闭”；



(8) Master设备界面不需要做修改，双击打开slave设备界面，设置要连接的从站ip（也即测试笔记本使用的ip，这里设定的是192.168.0.2，因此如下图填入）；

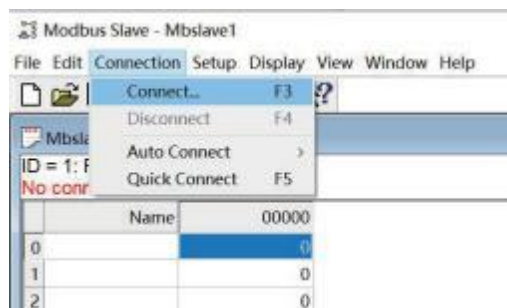


(9) 设置完成后，打开Modbus Slave程序，点击“Setup - Slave Definition...”；



功能码选择04，可以改变长度（Quantity），点击“OK”。

(10) 点击“Connection — Connect...”



在弹出的界面内，先选择“connection”为“Modbus TCP/IP”，然后去掉“Any Address”前的勾选，将测试电脑的ip地址（192.168.0.2）输入至“IP Address”框内，点击“OK”。

1. 下拉框选择Modbus TCP/IP



4. 点击“OK”

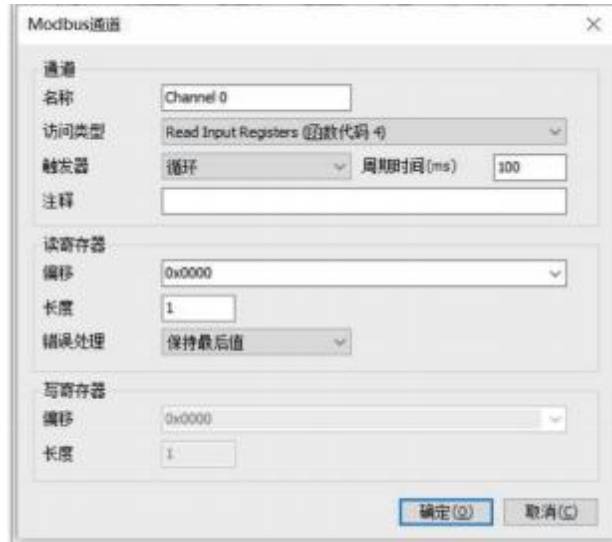
2. 去掉“Any Address”前的勾选

3. 输入IP Address

(11) 此时已经连接上，返回codesys界面，在Modbus\_TCP\_Slave界面的“Modbus从站通道”子界面内，点击“添加通道”。



添加Read Input Register通道（访问类型选“Read Input Register通道（函数代码4）”），可对从站寄存器进行读操作；



3 Channel 3 Read Input Registers (函数代码 04) 循环, t#100ms 16#0000 1 保持最后值

按照上图所示，读寄存器的地址偏移为0x0000，因此在Modbus Slave中双击地址为0的地方，在弹出的窗口内，输入数值（如15），点击“OK”写入。



在Modbus\_Slave\_TCP的“ModbusTCPSlaveI/O映射”子界面内可以看到数值已经同步过去。

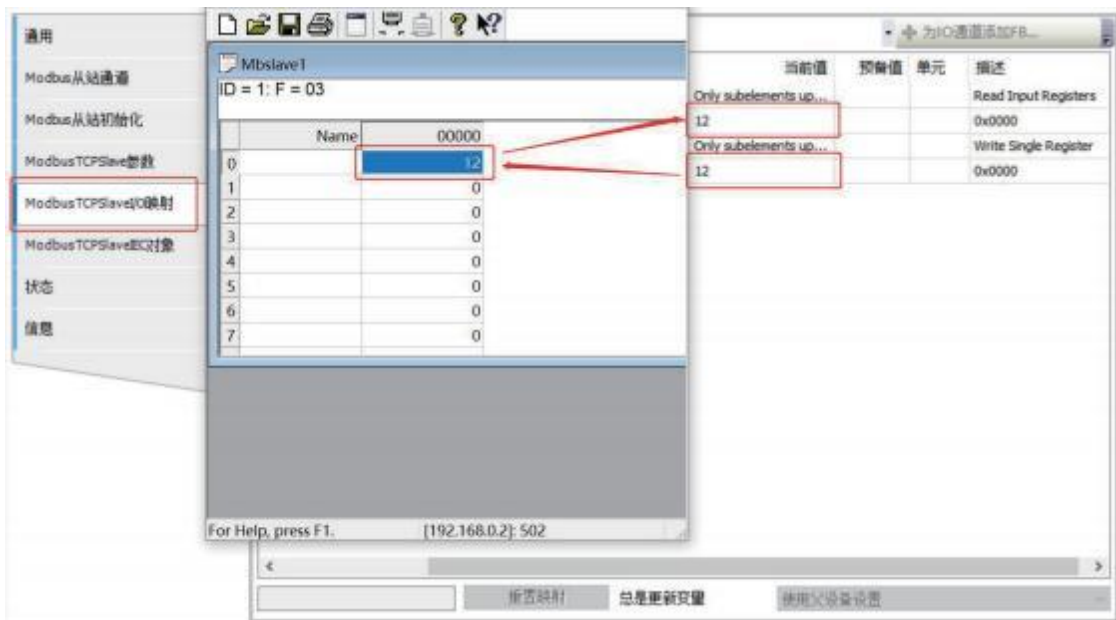


(12) 在Modbus Slave程序中，选功能码03（03 Holding Register (4x)）



在添加Read Holding Register和Write Single Register通道（访问类型分别选“Read Holding Register（函数代码3）”和“Write Single Register（函数代码6）”），可对从站寄存器进行读写操作。

|   |           |                                  |     |         |   |       |         |   |
|---|-----------|----------------------------------|-----|---------|---|-------|---------|---|
| 1 | Channel 1 | Write Single Register (函数代码 06)  | 上升沿 |         |   |       | 16#0000 | 1 |
| 2 | Channel 2 | Read Holding Registers (函数代码 03) | 上升沿 | 16#0000 | 1 | 保持最后值 |         |   |



如上图所示，读写地址都为0x0000，因此在“ModbusTCPSlaveI/O映射”子页面内，写入“12”时，ModbusSlave程序内和I/O映射界面的读输入寄存器位置都可以看到数值“12”。

## 8.2 Modbus TCP Slave

使用软件：Modbus Poll

具体操作步骤：

- (1) 新建标准工程文件；
- (2) 右键“Device”—添加设备；
- (3) 点开“以太网适配器”子选项，选中“Ethernet”，点击“添加设备”；



- (4) 添加完成后，双击打开Ethernet设备界面，在“通用”子界面，点击“Browse...”选择网络接口（选择网络接口前需要先连接设备Device）；



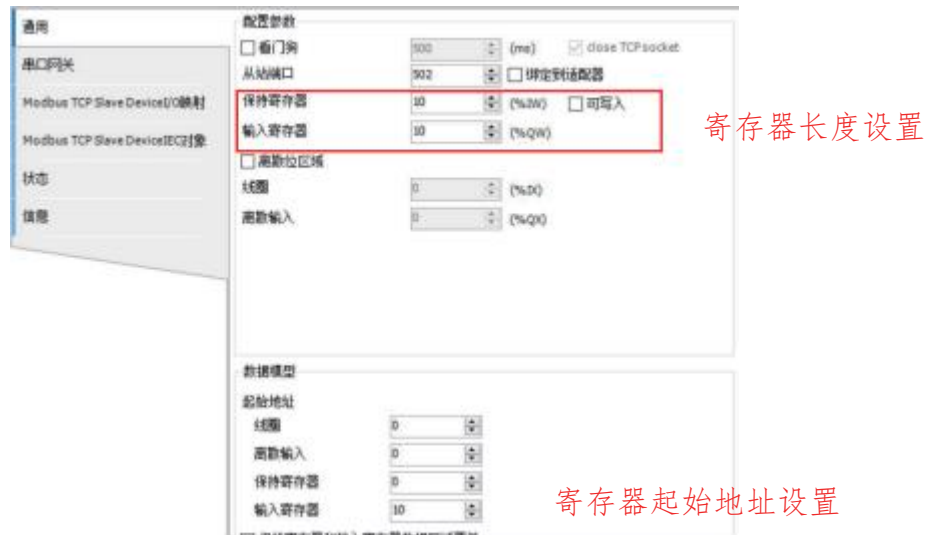
- (5) 选中“eth0”（也可以自定义），点击确认；



(6) “右键Ethernet—添加设备—按照图中所示位置，找到Modbus TCP Slave选中—添加设备—关闭”；

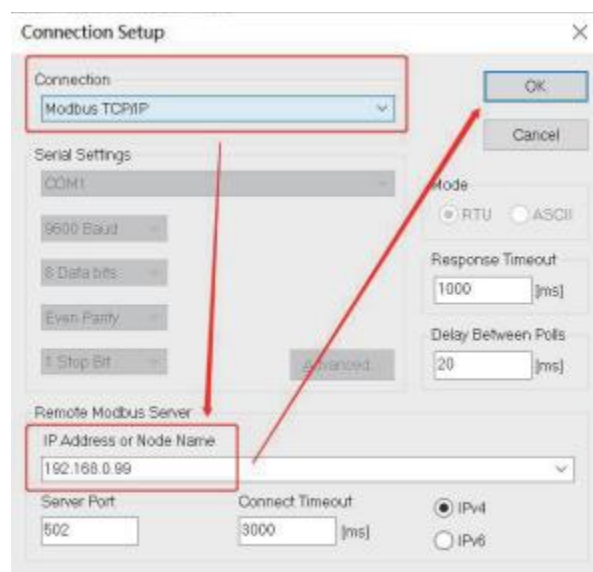
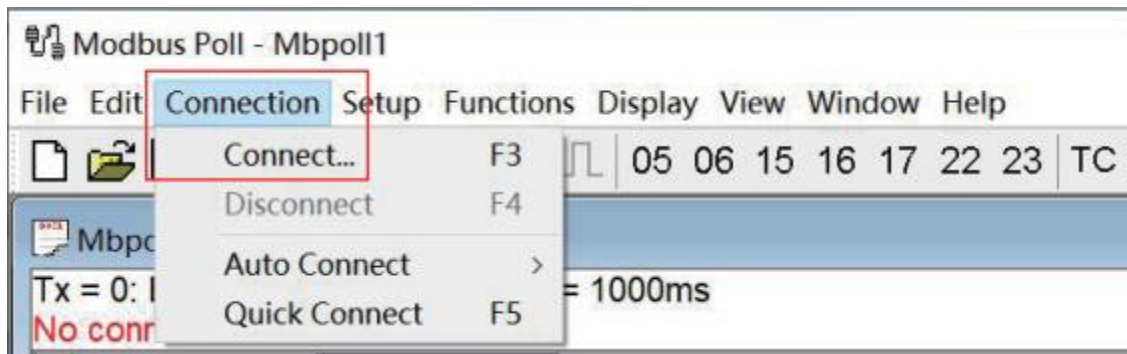


(7) 双击打开刚添加的“Slave”设备—“通用”界面，可以设定保持寄存器和输入寄存器的长度及起始地址，若需要用到线圈和离散输入，请勾选“离散位区域”，并对其长度进行设置；

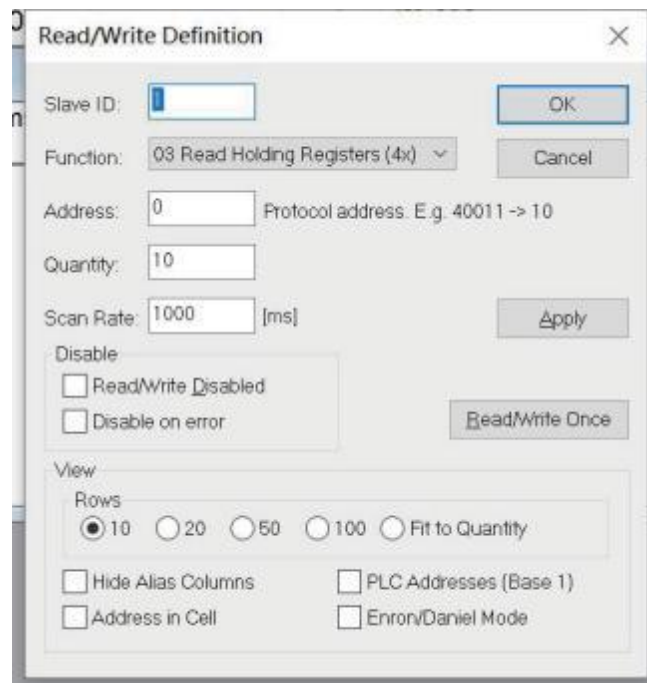
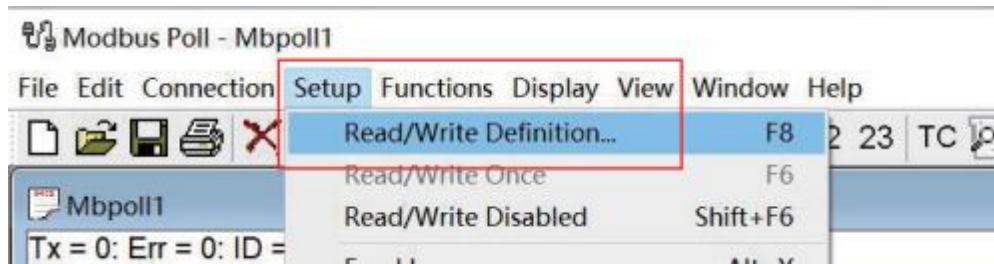


【注：功能码03，06，05对应的是保持寄存器，写线圈偏移0对应的是bit8。】

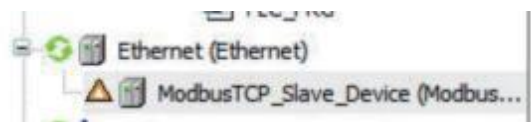
(8) 打开Modbus Poll程序，点击“Connection—Connect...”，输入PLC的ip地址；



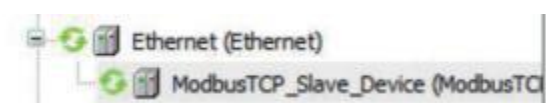
(9) 在“Setup—Read/Write Definition...”中，可以设置读写功能码（本例针对保持寄存器或输入寄存器）、读写地址以及长度等；

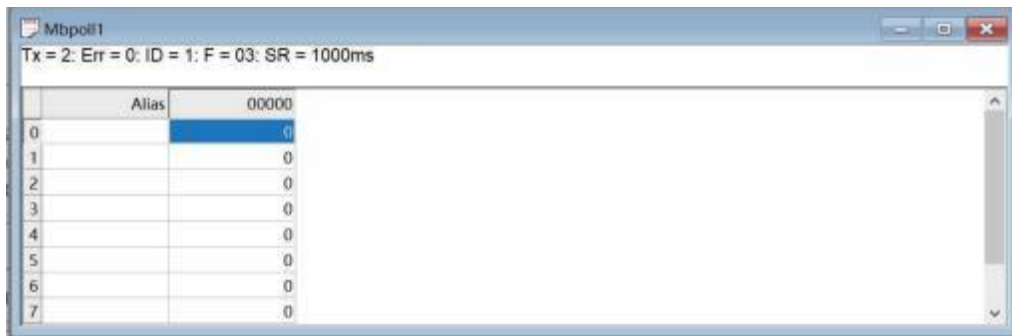


(10) 对保持寄存器操作：在Codesys软件中，登录设备，启动，会发现如下状态，这是因为Modbus Poll软件未启动导致，在上图中的Function（功能）处选择功能码03，确定之后，点击“Connection—Connect...”；

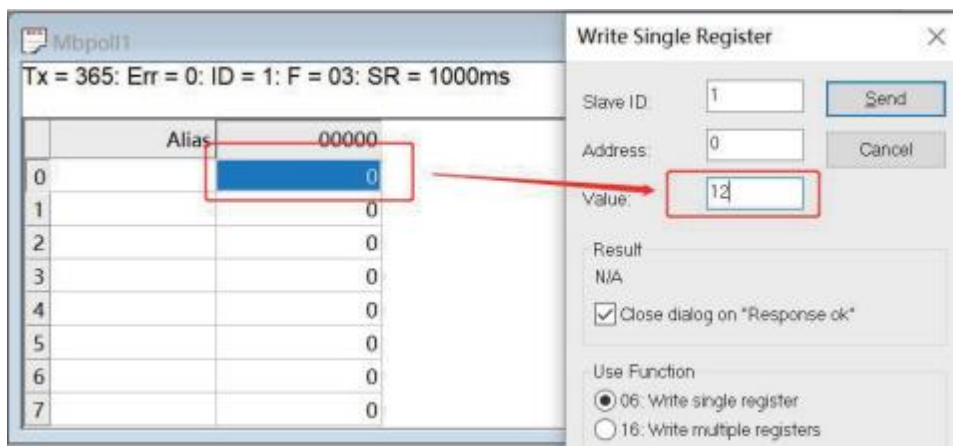


连接成功，如下图所示，

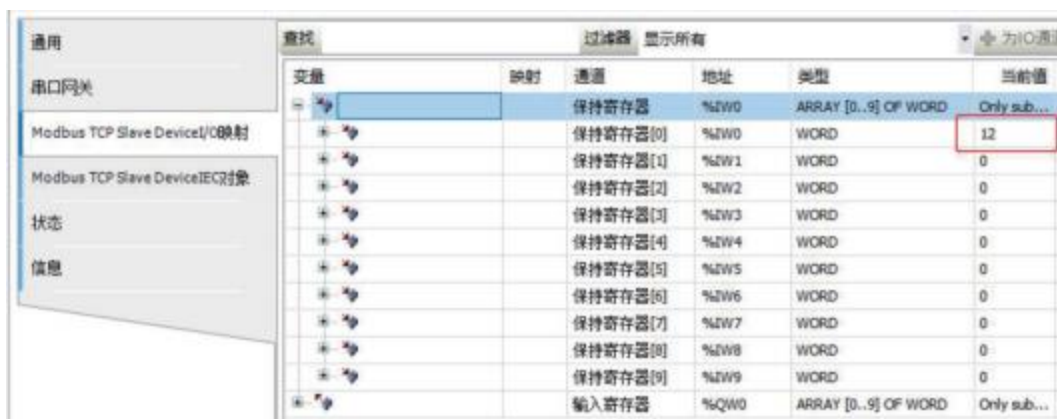




此时可以在Modbus Poll中可写对应寄存器值，



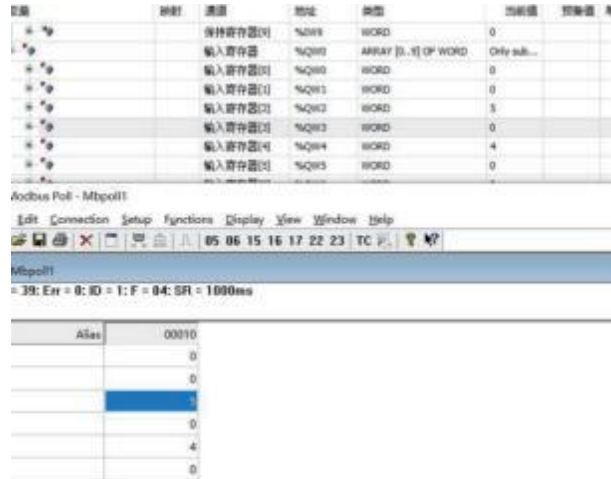
输入值12，点击“Send”发送成功后，可以在相应位置看到结果，点开“Modbus TCP Slave Device I/O”界面，发现保持寄存器[0]已经有数据收到。



(11) 对输入寄存器操作：在Codesys软件中，登录设备，启动，在下图中的

Function（功能）处选择功能码04，地址改为10，确定之后，点击“Connection-Connect...”；

同样连接成功后，在PLC的“Modbus TCP Slave DeviceI/O”界面的输入寄存器写入值，在Modbus Poll中可读对应寄存器值。

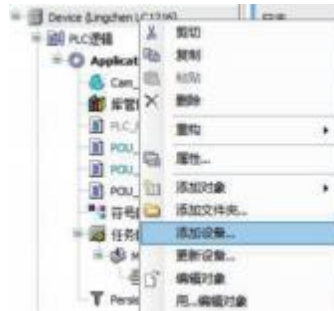


### 8.3 Modbus RTU

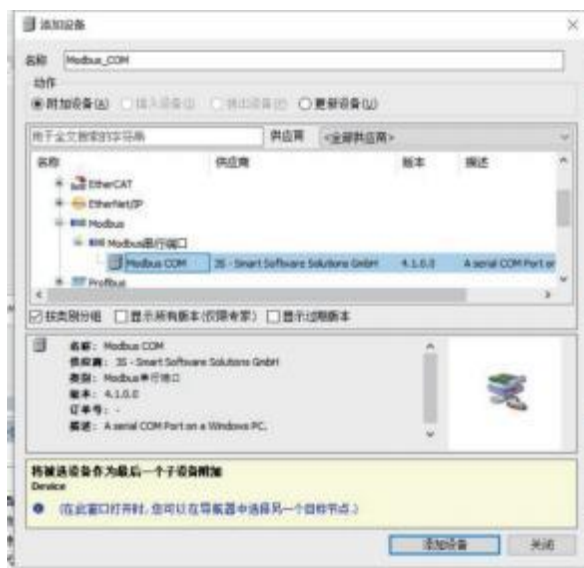
注意：请先关闭电脑防火墙。

LC1500只具有Modbus Serial Master功能，因此将其设为主站，操作步骤如下：

1. 右键“Device” — “添加设备”。



2. 依次展开“现场总线—Modbus—Modbus串行端口—Modbus COM”，选中后，点击“添加设备”。



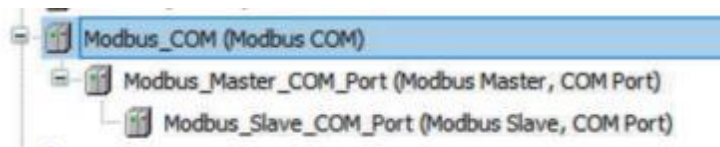
3. 添加完成后，可以不用关闭“添加设备”窗口，点击设备栏刚刚添加的“Modbus\_COM”，“添加设备”的窗口会自行跳转，点击串行主站，选中添加。



添加完串行主站，同样不用关闭“添加设备”窗口，点击设备栏刚刚添加的 Modbus\_Master\_Com\_Port，“添加设备”窗口跳转，点击串行设备，选中添加。



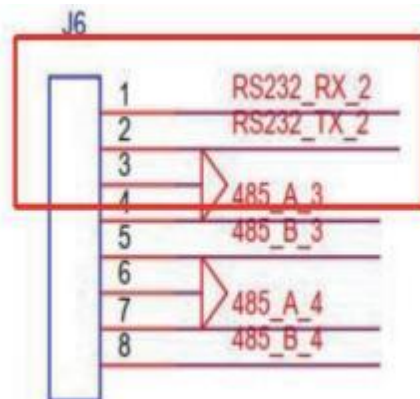
4. 添加完成后，设备栏如下图所示，双击“Modbus\_COM”打开界面，选择“通



用”子界面，修改COM端口及其他串口配置参数。



注意，端口号要和硬件连接的端口对应，如选232，则端口号为2，选485\_3，端口号为3，选485\_4,端口号为4。

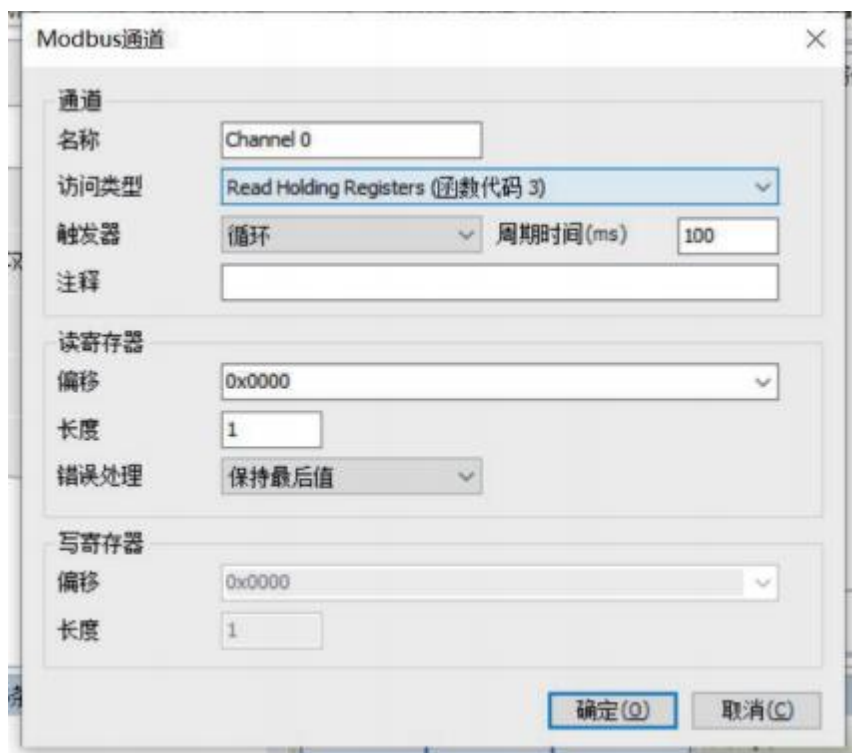


5. 双击“Modbus\_Slave\_Com\_Port”打开界面，在“通用”界面可以选择从站地

址（1~247），在“Modbus从站通道”子界面点击“添加通道”。



6. 在弹出的界面选择访问类型及地址偏移。



7. 添加完成后，如下图所示。可以配合其他串行从站进行使用。

| 名称          | 访问类型                             | 触发器         | 读偏移     | 长度 | 错误处理  | 写偏移     | 长度 | 注 |
|-------------|----------------------------------|-------------|---------|----|-------|---------|----|---|
| 0 Channel 0 | Read Holding Registers (函数代码 03) | 循环, t#100ms | 16#0000 | 1  | 保持最后值 |         |    |   |
| 1 Channel 1 | Write Single Register (函数代码 06)  | 循环, t#100ms |         |    |       | 16#0000 | 1  |   |

## 9. 断电保持功能的使用

1. 添加“掉电保持型变量”对象。右键“Application” — 添加对象 — 掉电保持型变量。



设定列表名称（这里使用默认名称），点击“添加”。

2. 添加完成后，会自动跳转该页面，注释掉首行，声明一个数组变量。

```

1  //(attribute 'qualified_only')
2  VAR_GLOBAL PERSISTENT RETAIN
3  test:ARRAY [1..20] OF USINT;
4  END_VAR
    
```

3. 登录一启动，打开掉电保持界面，展开变量，随意输入准备数字，“Ctrl+F7”写入。

| 表达式     | 类型        | 值 | 准备值 | 地址 | 注释 |
|---------|-----------|---|-----|----|----|
| test    | ARRAY ... |   |     |    |    |
| test[1] | USINT     | 1 |     |    |    |
| test[2] | USINT     | 0 |     |    |    |
| test[3] | USINT     | 2 |     |    |    |
| test[4] | USINT     | 0 |     |    |    |
| test[5] | USINT     | 3 |     |    |    |
| test[6] | USINT     | 0 |     |    |    |
| test[7] | USINT     | 0 |     |    |    |

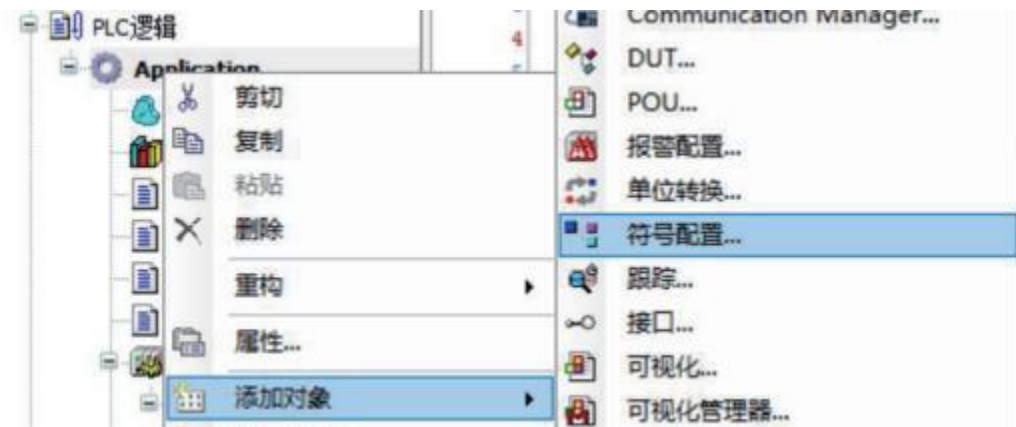
4. 断掉LC1500电源，重新上电，待PLC重新运行起来后，登录查看。

| 表达式     | 类型        | 值 | 准备值 |
|---------|-----------|---|-----|
| test    | ARRAY ... |   |     |
| test[1] | USINT     | 1 |     |
| test[2] | USINT     | 0 |     |
| test[3] | USINT     | 2 |     |
| test[4] | USINT     | 0 |     |
| test[5] | USINT     | 3 |     |
| test[6] | USINT     | 0 |     |

因此，可将相关变量声明为掉电保持型变量，掉电后再次上电，依旧可以保存下来。

## 10. OPC UA例程

1. 编写好程序之后，右键“Application”—添加对象—符号配置…。



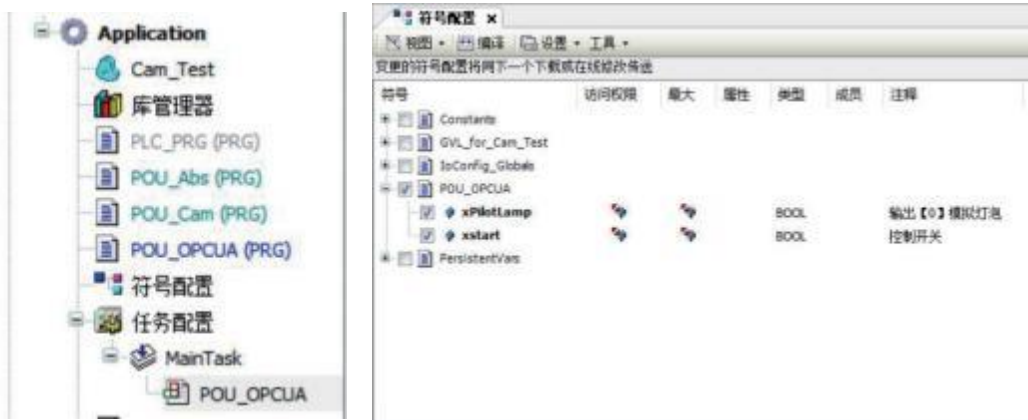
2. 在弹出的“添加 符号配置”界面，注意是否勾选支持OPC UA特征，点击“添加”。



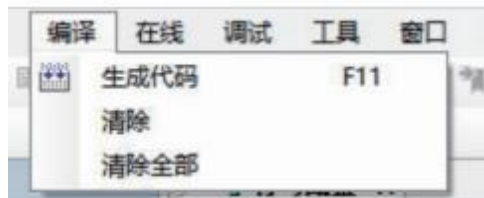
3. 添加完成后，可以看到“符号配置界面”，注意，需要先执行“编译”命令，



才能够选择需要的变量，点击“编译”。在此之前，请确认需要的变量所在POU已拖入任务配置，分配Task。勾选变量。



4. 勾选完后，在菜单栏点击“编译——生成代码”。



5. 生成代码结束，可以在消息栏看到相关信息。



6. 此时，可以在该工程文件的存储目录下，找到生成的xx.Device.Application.xml文件。

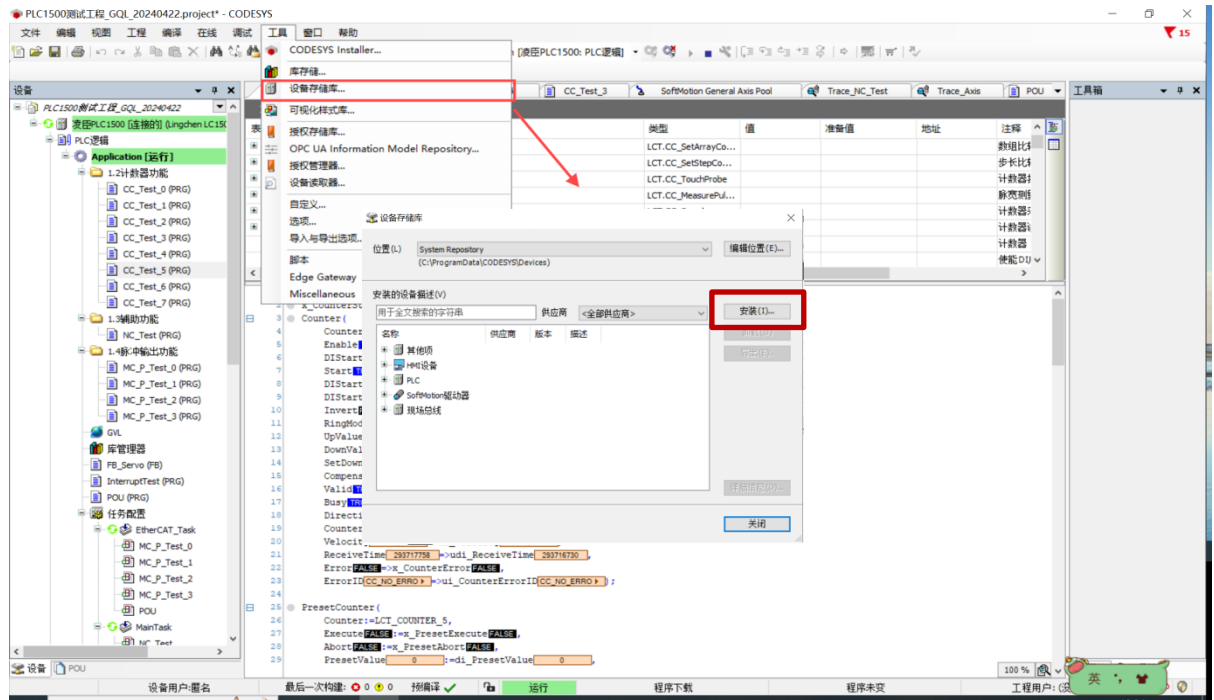


7. 之后使用人机界面的组态程序，扫描标签即可。

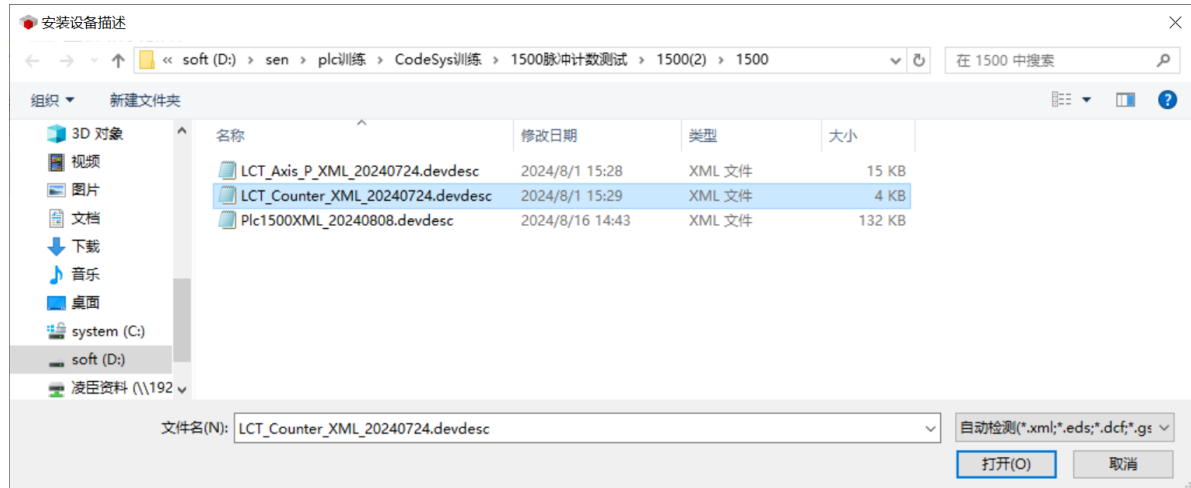
## 8.4 高速脉冲计数

### 1. 库文件安装及应用

点击工具栏“工具”选项，选择“设备存储库”，然后点击“安装”。

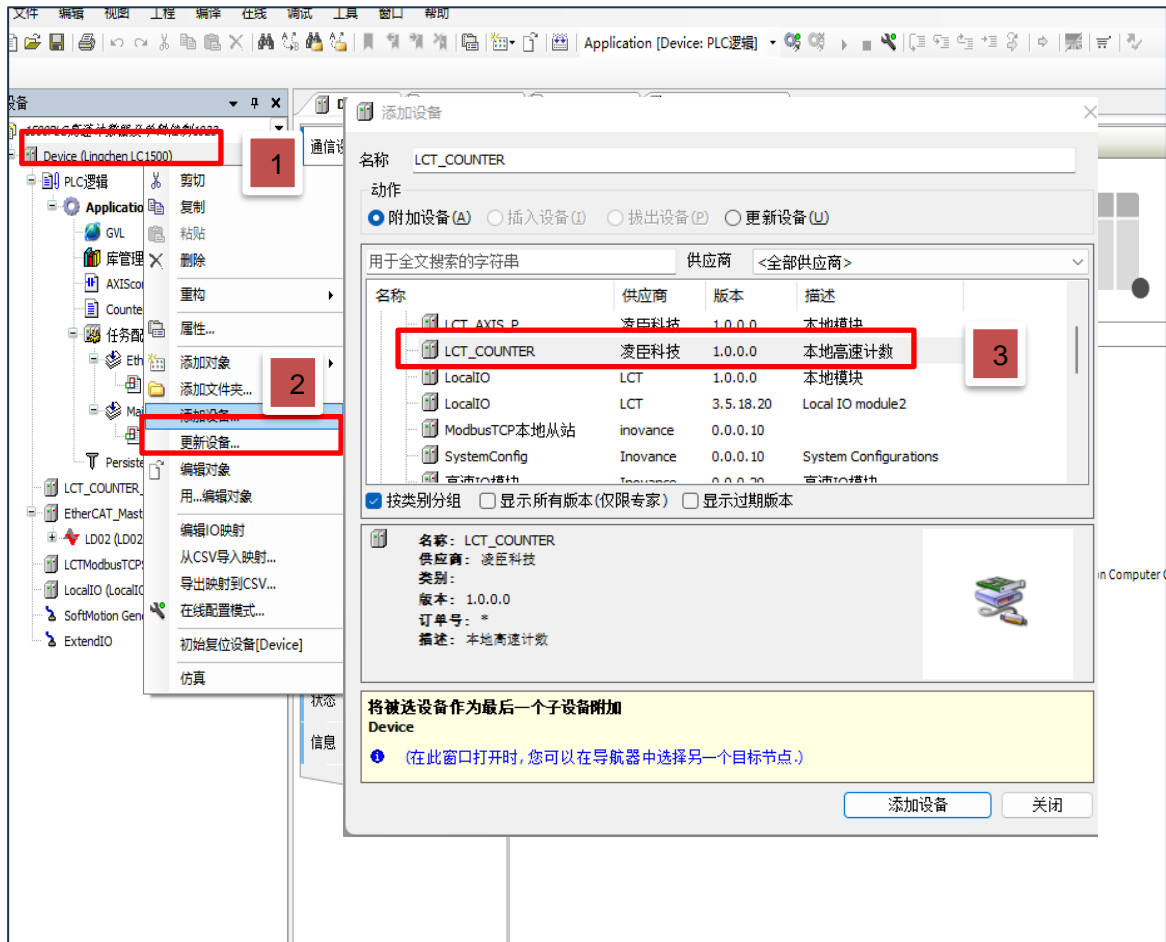


选择“LCT\_Counter\_XML” XML文件，点击打开进行安装。

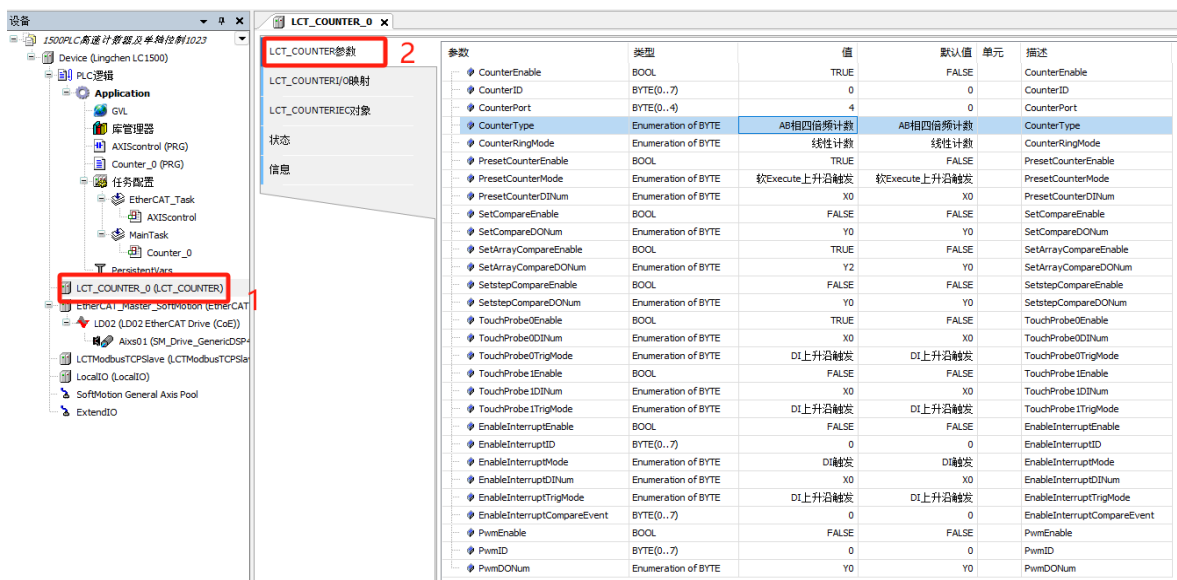


## 2.设备添加

右击“plc1500”，选择“添加设备”，选择“其他项”，选择“LCT\_COUNTER”，点击确定。左图为添加后的计数器设备



点击计数器组件，即可进入计数器参数配置界面如下：



### 3. 计数器参数配置

#### 计数器ID

数值范围[0..7]，每一个计数器对应一个ID，每个计数器有且只有一个ID。

#### 计数器输入硬件编号

数值范围[0..4]，编号1-3为常规脉冲单端计数器，编号4为编码器差分计数器；

#### 常规脉冲单端计数器：

硬件编号0对应硬件输入X0-X1两个输入口，硬件编号1对应硬件输入X2-X3两个输入口，硬件编号2对应硬件输入X4-X5两个输入口，硬件编号3对应硬件输入X6-X7两个输入口。

#### 编码器脉冲差分计数器：

硬件编号4，对应硬件输入X10-X17八个输入口；端口X10、X11分别为EA+、EA-；端口X12、X13分别为EB+、EB-；端口X14、X15分别为EZ+、EZ-；端口X16、X17分别为编码器电源5V输出、编码器电源0V输出。

#### 计数类型

一输入单相计数：单相脉冲单输入口计数，无方向。

二输入单相计数：单相脉冲双输入口正反向计数，输入口一脉冲输入加计数，输入口二脉冲输入减计数。

AB相四倍频计数：A、B相双输入正交计数，若A相早于B相则加计数，若B相早于A相则减计数。

### 4. 计数器指令块应用

声明计数器“Counter:Lct.CC\_Counter;”实例化

```

1 PROGRAM CC_Test_5
2 VAR
3 // 计数器[5]测试功能块
4 Counter : LCT_CC_Counter; // 计数器
5 PresetCounter : LCT_CC_PresetCounter; // 计数器预置
6 SetCompare : LCT_CC_SetCompare; // 比较器
7 SetArrayCompare : LCT_CC_SetArrayCompare; // 数组比较器
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
246
```

```
Counter(  
  Counter:=LCT_COUNTER_5, //指定使用的计数器名称  
  Enable[TRUE]:=x_CounterEnable[TRUE], //使能计数器, TRUE:初始化并生效功能块, FALSE:功能块不生效  
  DIStartValid[FALSE]:=x_DIStartValid[FALSE], //启动模式选择, TRUE:DI启动, FALSE:软件启动  
  Start[TRUE]:=x_CounterStart[TRUE], //软件启动信号(上升沿)  
  DIStartMode[0]:=usi_DIStartMode[0], //DI启动模式选择, 0: 上升沿启动, 1:下降沿启动  
  DIStartNum[0]:=usi_DIStartNum[0], //DI端口号选择[0..15]  
  Invert[FALSE]:=x_Invert[FALSE], //计数值取反  
  RingMode[FALSE]:=x_RingMode[FALSE], //计数模式, 0:线性模式, 1:循环模式  
  UpValue[1000]:=di_UpValue[1000], //计数上限值设置  
  DownValue[-1000]:=di_DownValue[-1000], //计数下限值设置  
  SetDown[FALSE]:=x_SetDown[FALSE], //计数器启动默认初始数值为计数下限值, TRUE:生效, FALSE:不生效  
  Compensation[FALSE]:=x_Compensation[FALSE], //补偿方式, TRUE, 手动补偿或不补偿, FALSE:系统自动补偿  
  Valid[TRUE]:=x_Valid[TRUE], //计数器计数标志, TRUE:有效, FALSE:无效  
  Busy[TRUE]:=x_CounterBusy[TRUE], //执行标志, FALSE: 计数器未执行, TRUE: 计数器已执行  
  Direction[FALSE]:=x_Direction[FALSE], //计数方向, FALSE: 正方向(加法计数), TRUE: 反方向(减法计数)  
  CounterValue[40000]:=di_CounterValue[40000], //计数器值, 计数器当前计数值, 单位: pulse  
  Velocity[0]:=di_Velocity[0], //计数器速度, 计数器当前速度, 单位: pulse/s  
  ReceiveTime[1469865941]:=udi_ReceiveTime[1469865941], //读取计数值时刻, 读取CounterValue的系统时刻  
  Error[FALSE]:=x_CounterError[FALSE], //错误标志, TRUE: 功能块内部发生错误  
  ErrorID[CC_NO_ERROR]:=ui_CounterErrorID[CC_NO_ERROR]; //错误ID, 错误码, 具体参考CC_ERROR
```

### 功能块操作说明

- 1, “Counter” 引脚绑定添加的计数器名称, 该功能块使用前必须使能计数器 “Enable” 引脚, 否则计数器不生效。
- 2, “Start”, 0→1跳变: 启动计数器。
- 3, “Invert” 引脚决定计数器计数方向, 默认FALSE, 随接收脉冲量累加; 置TRUE, 随接收脉冲量递减。
- 4, “RingMode” 引脚决定计数器计数模式, 默认FALSE, 计数器数值可以逐渐增大, 持续累加; 置TRUE, 计数器计数数值在引脚 “UpValue”、“DownValue” 之间循环计数。
- 5, “SetDown” 引脚决定计数器初始值, 默认FALSE, 计数器计数初始值为0; 置TRUE, 计数器初始值为设定的 “DownValue” 引脚值。
- 6, “DIStartValid” 引脚置 “TRUE” 时, 计数器则不受 “Start” 引脚控制, “DIStartNum” 引脚选择的输入点信号置 “TRUE” 时, 计数器启动, 可通过 “DIStartMode” 引脚选择输入点信号上升沿或下降沿启动计数器。
- 7, “RingMode” 引脚激活时, 计数器会在 “UpValue” 上限引脚和 “DownValue” 下限引脚设定的范围内循环计数, 当数值到达上限, 计数会从下限重新计数; 若 “RingMode” 引脚不激活则计数一直累加。

## 声明计数器预置“PresetCounter: LCT\_CC\_PresetCounter;”实例化

```

1  PROGRAM CC_Test_5
2  VAR
3  // 计数器(5)测试功能块
4  Counter : LCT_CC_Counter; // 计数器
5  PresetCounter : LCT_CC_PresetCounter; // 计数器预置
6  SetCompare : LCT_CC_SetCompare; // 比较器
7  SetArrayCompare : LCT_CC_SetArrayCompare; // 数组比较器
8  SetStepCompare : LCT_CC_SetStepCompare; // 步长比较器
9  TouchProbe : LCT_CC_TouchProbe; // 计数器探针
10
11 PresetCounter(
12 Counter:=LCT_COUNTER_5, //指定使用的计数器名称
13 Execute:=x_PresetExecute, //上升沿触发, TRUE: 使能计数器预置
14 Abort:=x_PresetAbort, //上升沿触发, TRUE: 终止计数器预置
15 PresetValue:=di_PresetValue, //预置值, 单位: pulse
16 PresetMode:=usi_PresetMode, //预置模式, 0: 软Execute上升沿触发1: 比较一致输出时触发2: DI上升沿/下降沿触发3: DI上升沿触发4: DI下降沿触发
17 DINumPreset:=usi_DINumPreset, //PresetMode=0,3,4时, 指定预置的DI口
18 DIPresetLenValid:=x_DIPresetLenValid, //PresetMode=2,3,4时, TRUE: 启动预置信号长度滤波
19 DIPresetLenMin:=di_DIPresetLenMin, //预置信号最小长度设置, 单位: pulse
20 DIPresetLenMax:=di_DIPresetLenMax, //预置信号最大长度设置, 单位: pulse
21 Done:=x_PresetDone, //TRUE: 功能块执行完成
22 Busy:=x_PresetBusy, //FALSE: 功能块未执行, TRUE: 功能块执行中
23 CommandAborted:=x_PresetCommandAborted, //TRUE: 功能块被终止
24 Error:=x_PresetError, //TRUE: 功能块内部发生错误
25 ErrorID:=ui_PresetErrorID); //错误码, 具体参考CC_ERROR
26

```

### 功能块操作说明

```

PresetCounter(
Counter:=LCT_COUNTER_5, //指定使用的计数器名称
Execute TRUE :=x_PresetExecute TRUE, //上升沿触发, TRUE: 使能计数器预置
Abort FALSE :=x_PresetAbort FALSE, //上升沿触发, TRUE: 终止计数器预置
PresetValue 0 :=di_PresetValue 0, //预置值, 单位: pulse
PresetMode 0 :=usi_PresetMode 0, //预置模式, 0: 软Execute上升沿触发1: 比较一致输出时触发2: DI上升沿/下降沿触发3: DI上升沿触发4: DI下降沿触发
DINumPreset 0 :=usi_DINumPreset 0, //PresetMode=2,3,4时, 指定预置的DI口
DIPresetLenValid FALSE :=x_DIPresetLenValid FALSE, //PresetMode=2,3,4时, TRUE: 启动预置信号长度滤波
DIPresetLenMin 0 :=di_DIPresetLenMin 0, //预置信号最小长度设置, 单位: pulse
DIPresetLenMax 0 :=di_DIPresetLenMax 0, //预置信号最大长度设置, 单位: pulse
Done TRUE =>x_PresetDone TRUE, //TRUE: 功能块执行完成
Busy FALSE :=x_PresetBusy FALSE, //FALSE: 功能块未执行, TRUE: 功能块执行中
CommandAborted FALSE =>x_PresetCommandAborted FALSE, //TRUE: 功能块被终止
Error FALSE =>x_PresetError FALSE, //TRUE: 功能块内部发生错误
ErrorID[CC_NO_ERROR] =>ui_PresetErrorID[CC_NO_ERROR]; //错误码, 具体参考CC_ERROR

```

- 1, “Counter” 引脚绑定添加的计数器名称;计数器预置功能块能够实现设定计数器预置PresetCounter的“PresetValue” 引脚的值赋值到计数器Counter的“CounterValue” 引脚的计数值。
- 2, “Execute” 引脚, 默认FALSE, 置TRUE上升沿, “PresetValue” 引脚当前值赋值到计数器Counter “CounterValue” 引脚的值; 该功能需在“PresetMode” 引脚值为0时实现。
- 3, “PresetMode” 引脚决定计数器预置的模式, 0: “Execute” 引脚触发, 1: 比较器PresetCounter比较完成后触发, 2: “DINumPreset” 引脚设定的端口信号置TRUE上升沿和下降沿各触发一次, 3: “DINumPreset” 引脚设定的端口信号置TRUE上升沿触发一次, 4: “DINumPreset” 引脚设定的端口信号置TRUE上升沿触发一次。
- 4, “DINumPreset” 引脚决定预置模式选定DI触发的端口。
- 5, “DIPresetLenValid” 引脚默认FALSE; 置TRUE, 功能块会过滤掉“DIPresetLenMin” 引脚和“DIPresetLenMax” 引脚分别设定的预置最大值和最小值外的长度不符高低电平。
- 6, 若“Done” 引脚输出表示功能块预置完成, “Execute” 引脚需要重新触发上升沿, 预置器重启。

声明比较器“SetCompare: LCT.CC\_SetCompare;”实例化

```

1  PROGRAM CC_Test_5
2  VAR
3  // 计数器[5]测试功能块
4  Counter                               : LCT.CC_Counter;           // 计数器
5  PresetCounter                         : LCT.CC_PresetCounter;     // 计数器预置
6  SetCompare                            : LCT.CC_SetCompare;        // 比较器
7  SetArrayCompare                       : LCT.CC_SetArrayCompare;   // 数组比较器
8  SetStepCompare                        : LCT.CC_SetStepCompare;    // 步长比较器
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44  SetCompare (
45  Counter:=LCT_COUNTER_5, //指定使用的计数器名称
46  Execute:=x_CompareExecute, //上升沿触发, TRUE: 使能功能块
47  Abort:=x_CompareAbort, //上升沿触发, TRUE: 终止功能块
48  CompareValue:=di_CompareValue, //比较值, 单位: pulse
49  Mode:=usi_CompareMode, //FALSE: 单次模式, TRUE: 连续模式
50  OutputDONum:=usi_CompDONum, //硬件输出DO端口编号
51  OutputType:=x_CompOutputType, //FALSE: 时间方式, TRUE: 脉冲方式
52  OutputValue:=di_CompOutputValue, //如为时间方式, 单位为100us;如为脉冲方式, 单位为pulse.
53  Done:=x_CompDone, //如为单次模式, 执行一次输出后, Done, 功能自动终止. 连续模式, 此信号无用
54  Busy:=x_CompBusy, //FALSE: 功能块未执行, TRUE: 功能块执行中
55  CommandAborted:=x_CompCommandAborted, //TRUE: 功能块终止
56  Output:=x_CompOutput, //比较一致输出, 与DO一致
57  CmpCount:=ui_CompCount, //比较输出次数
58  Position:=di_CompPositon, //当前需要被比较的值
59  Error:=x_CompareError, //TRUE: 功能块内部发生错误
60  ErrorID:=ui_CompareErrorID); //错误码, 具体参考CC_ERROR

```

功能块操作说明

```

SetCompare (
Counter:=LCT_COUNTER_5, //指定使用的计数器名称
Execute TRUE:=x_CompareExecute TRUE, //上升沿触发, TRUE: 使能功能块
Abort FALSE:=x_CompareAbort FALSE, //上升沿触发, TRUE: 终止功能块
CompareValue 10000:=di_CompareValue 10000, //比较值, 单位: pulse
Mode 1:=usi_CompareMode 1, //FALSE: 单次模式, TRUE: 连续模式
OutputDONum 5:=usi_CompDONum 5, //硬件输出DO端口编号
OutputType FALSE:=x_CompOutputType FALSE, //FALSE: 时间方式, TRUE: 脉冲方式
OutputValue 8:=di_CompOutputValue 8, //如为时间方式, 单位为100us;如为脉冲方式, 单位为pulse.
Done FALSE:=x_CompDone FALSE, //如为单次模式, 执行一次输出后, Done, 功能自动终止. 连续模式, 此信号无用
Busy TRUE:=x_CompBusy TRUE, //FALSE: 功能块未执行, TRUE: 功能块执行中
CommandAborted FALSE:=x_CompCommandAborted FALSE, //TRUE: 功能块终止
Output FALSE:=x_CompOutput FALSE, //比较一致输出, 与DO一致
CmpCount 0:=ui_CompCount 0, //比较输出次数
Position 33564432:=di_CompPositon 33564432, //当前需要被比较的值
Error FALSE:=x_CompareError FALSE, //TRUE: 功能块内部发生错误
ErrorID CC_NO_ERRO:=ui_CompareErrorID CC_NO_ERRO); //错误码, 具体参考CC_ERROR

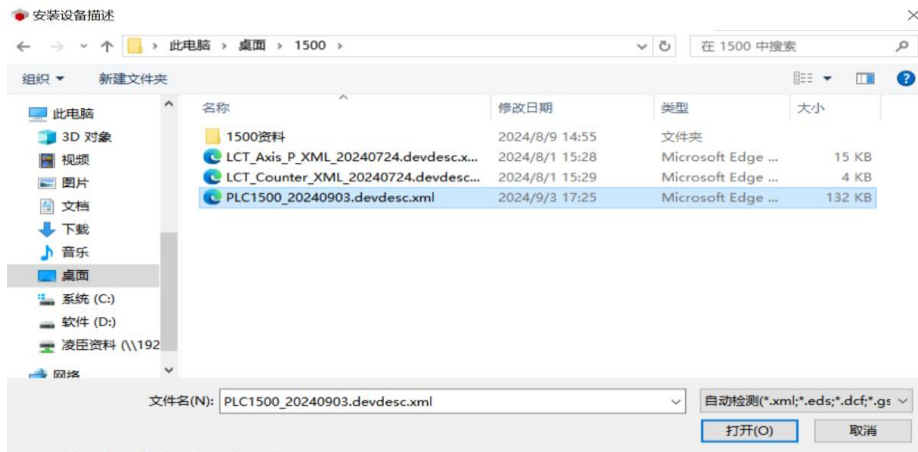
```

- 1, “Counter” 引脚绑定添加的计数器名称;比较器功能块能够实现设定比较器SetCompare的“CompareValue”引脚的设定值与计数器Counter的“CounterValue”引脚的计数值比较, 若相等则完成比较。
- 2, “Mode” 引脚决定比较器比较模式; 0: 单次模式, 比较器在引脚“Execute”上升沿触发后, 执行一次比较, 完成后功能块结束, “OutputDONum” 引脚设定的端口和“Output” 引脚输出, “Done” 引脚置TRUE。1: 循环模式, 比较器比较器在引脚“Execute”上升沿触发后一直执行比较, 完成后“OutputDONum” 引脚设定的端口和“Output” 引脚输出, “Done” 不变, “Abort” 引脚置TRUE后功能块比较停止。
- 3, “CmpCount” 引脚显示比较完成次数。
- 4, “Position” 引脚显示当前需要比较的设定值。
- 5, “OutputType” 引脚可以设定输出。

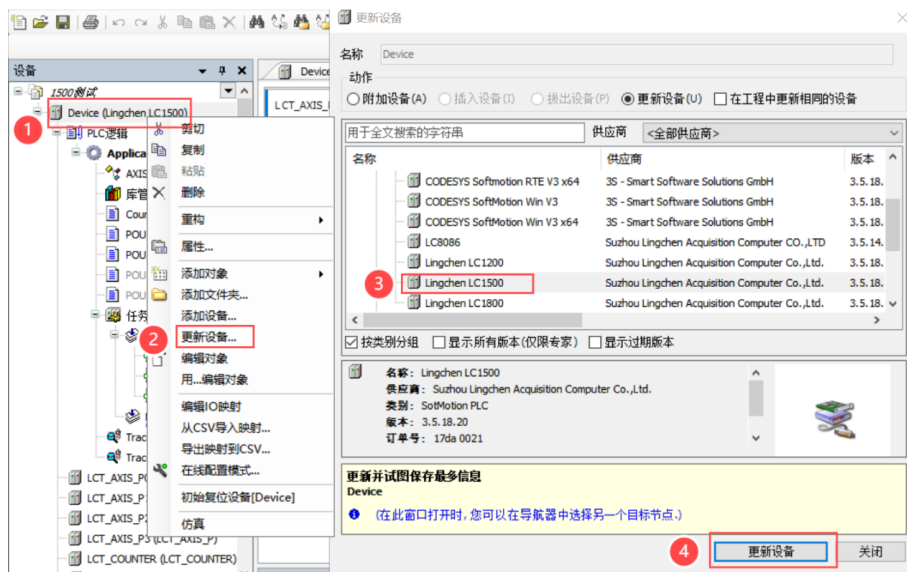
以上为个别功能块示例, 其他计数器功能块还请查阅《高速IO硬件功能说明书》



选择“PLC1500\_20240903.devdesc” XML文件，点击打开进行安装。

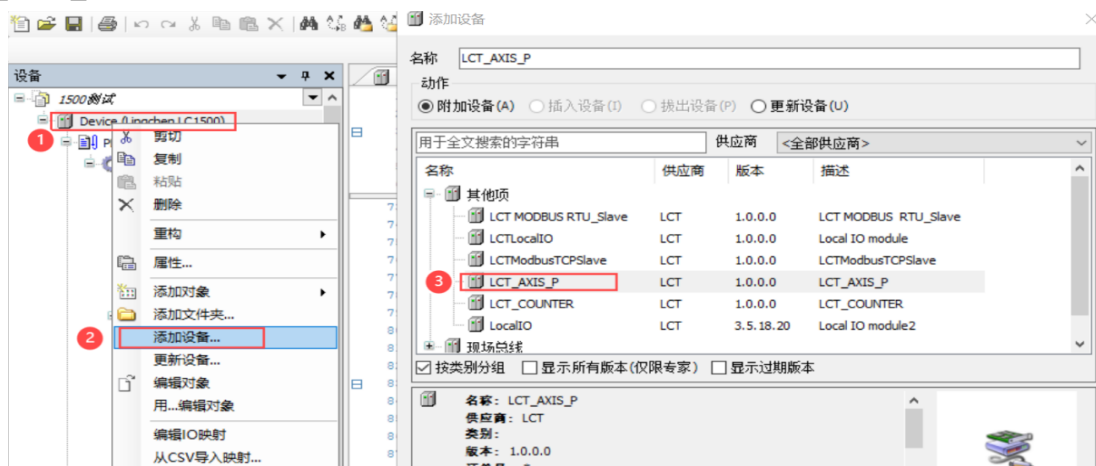


安装完成后，右击Device，点击更新设备，选择Lingchen LC1500，更新设备即可。



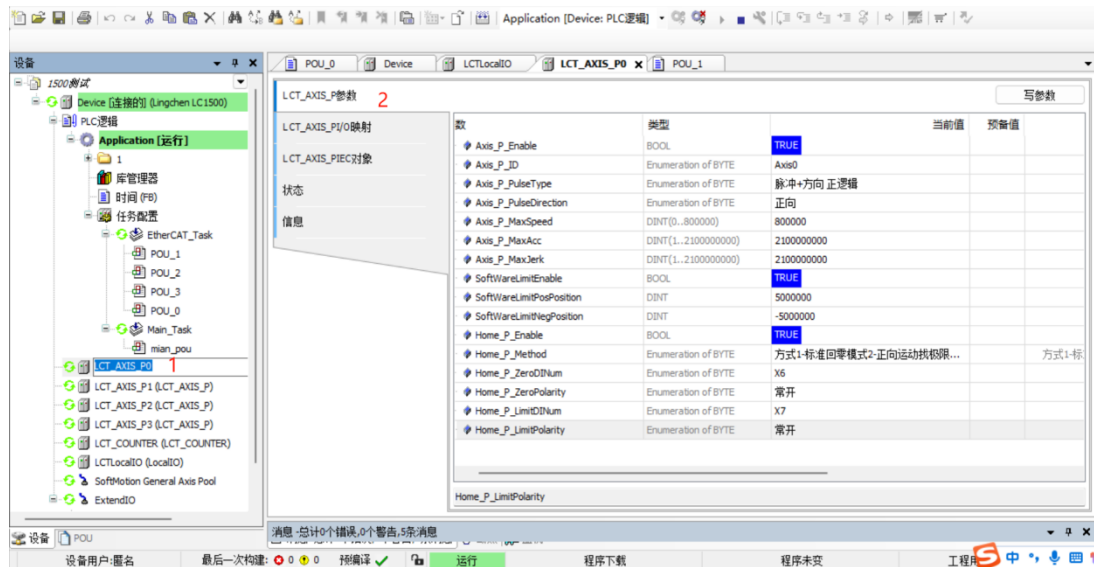
然后以同样的方式安装“LCT\_Axis\_P\_XML” XML文件。

安装完成后，添加高速脉冲轴，右击Device，单击添加设备，在设备列表中选择 LCT\_AXIS\_P。



添加完成后，双击已添加的轴进入参数设置

轴ID根据接线选择Axis0、Axis1、Axis2、Axis3，PLC设置脉冲类型选择与驱动器一致，这里选用脉冲+方向控制正逻辑，回零方式按需求选择，原点DI、极限DI按接线选择、软限位设置。



### 8.5.2 参数说明

打开轴参数配置页面，可对轴参数进行配置。

| 参数                       | 类型                  | 值                    | 默认值         | 单元 | 描述                       |
|--------------------------|---------------------|----------------------|-------------|----|--------------------------|
| Axis_P_Enable            | BOOL                | TRUE                 | FALSE       |    | Axis_P_Enable            |
| Axis_P_ID                | Enumeration of BYTE | Axis0                | Axis0       |    | Axis_P_ID                |
| Axis_P_PulseType         | Enumeration of BYTE | 脉冲+方向 正逻辑            | AB相四倍频脉冲    |    | Axis_P_PulseType         |
| Axis_P_PulseDirection    | Enumeration of BYTE | 正向                   | 正向          |    | Axis_P_PulseDirection    |
| Axis_P_MaxSpeed          | DINT(0..800000)     | 1000000              | 1000000     |    | Axis_P_MaxSpeed          |
| Axis_P_MaxAcc            | DINT(1..2100000000) | 1000000              | 1000000     |    | Axis_P_MaxAcc            |
| Axis_P_MaxJerk           | DINT(1..2100000000) | 10000000             | 10000000    |    | Axis_P_MaxJerk           |
| SoftWareLimitEnable      | BOOL                | TRUE                 | FALSE       |    | SoftWareLimitEnable      |
| SoftWareLimitPosPosition | DINT                | 2147483647           | 2147483647  |    | SoftWareLimitPosPosition |
| SoftWareLimitNegPosition | DINT                | -2147483648          | -2147483648 |    | SoftWareLimitNegPosition |
| Home_P_Enable            | BOOL                | TRUE                 | FALSE       |    | Home_P_Enable            |
| Home_P_Method            | Enumeration of BYTE | 方式3-标准回零模式18-正向运动找极限 | 方式8-标准回...  |    | Home_P_Method            |
| Home_P_ZeroDINum         | Enumeration of BYTE | X6                   | X0          |    | Home_P_ZeroDINum         |
| Home_P_ZeroPolarity      | Enumeration of BYTE | 常开                   | 常开          |    | Home_P_ZeroPolarity      |
| Home_P_LimitDINum        | Enumeration of BYTE | X7                   | X0          |    | Home_P_LimitDINum        |
| Home_P_LimitPolarity     | Enumeration of BYTE | 常开                   | 常开          |    | Home_P_LimitPolarity     |

1. 轴使能：Axis\_P\_Enable设为TRUE。
2. 轴ID：根据接线可选：AXIS0：D00为脉冲信号，D01为方向信号；AXIS1：D02为脉冲信号，D03为方向信号；AXIS2：D04为脉冲信号，D05为方向信号；AXIS3：D06为脉冲信号，D07为方向信号，脉冲+方向控制支持正逻辑和负逻辑；AB相四倍频脉冲：AXIS0：D00为A相，D01为B相；AXIS1：D02为A相，D03为B相；AXIS2：D04为A相，D05为B相；AXIS3：D06为A相，D07为B相。
3. 脉冲类型：脉冲+方向（正逻辑或负逻辑）；A/B相正交脉冲4倍频。脉冲类型选择与驱动器设置一致。

| 脉冲形态                  | 信号                    | 正转脉冲示意图 | 反转脉冲示意图 |
|-----------------------|-----------------------|---------|---------|
| 脉冲 + 方向<br>正逻辑        | PULSE<br>SIGN         |         |         |
| 脉冲 + 方向<br>负逻辑        | PULSE<br>SIGN         |         |         |
| A相 +B相<br>正交脉冲<br>4倍频 | PULSE(A相)<br>SIGN(B相) |         |         |

4. 方向: TRUE: 方向取反, FALSE: 正常方向。
5. 最大速度: 速度允许设置的最大值, 最大1000000, 单位pulse。
6. 最大加速度: 加速度允许设置的最大值, 最大2100000000, 单位pulse。
7. 最大加加速度: 加加速度允许设置的最大值, 最大2100000000, 单位pulse。
8. 使能正向软限位: TRUE: 启用软限位, FALSE: 不启用软限位。
9. 使能负向软限位: TRUE: 启用软限位, FALSE: 不启用软限位。
10. 回零方式: 见MC\_Home\_P功能块说明。
11. 原点DI: 根据原点开关接线选择X0~X7。
12. 原点DI极性: 根据原点开关极性选择常开常闭。
13. 回零极限DI: 根据极限开关接线选择X0~X7。
14. 回零极限DI极性: 根据极限开关极性选择常开常闭。

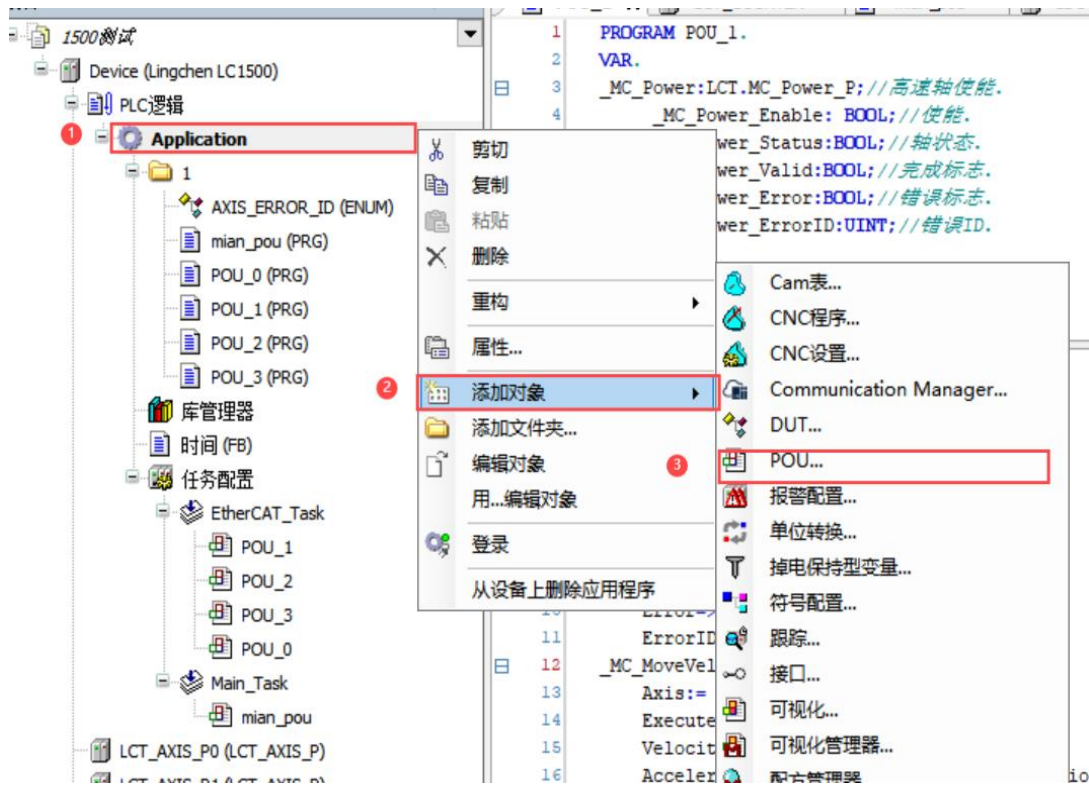
### 功能块指令列表:

| 序号       | 指令名称                  | FB/FC | 功能        |
|----------|-----------------------|-------|-----------|
| 1. 4. 1  | MC_Power_P            | FB    | 高速轴使能     |
| 1. 4. 2  | MC_Reset_P            | FB    | 高速轴复位     |
| 1. 4. 3  | MC_MoveAbsolute_P     | FB    | 高速轴绝对定位控制 |
| 1. 4. 4  | MC_MoveRelative_P     | FB    | 高速轴相对定位控制 |
| 1. 4. 5  | MC_MoveVelocity_P     | FB    | 高速轴速度控制   |
| 1. 4. 6  | MC_Home_P             | FB    | 高速轴回零     |
| 1. 4. 7  | MC_Stop_P             | FB    | 高速轴停止     |
| 1. 4. 8  | MC_Jog_P              | FB    | 高速轴点动     |
| 1. 4. 9  | MC_SetPosition_P      | FB    | 高速轴位置设定   |
| 1. 4. 10 | MC_ReadParameter_P    | FB    | 高速轴读参数    |
| 1. 4. 11 | MC_WriteParameter_P   | FB    | 高速轴写参数    |
| 1. 4. 12 | MC_ReadStatus_P       | FB    | 高速轴读轴状态   |
| 1. 4. 13 | MC_ReadMotionStatus_P | FB    | 高速轴读轴运动状态 |

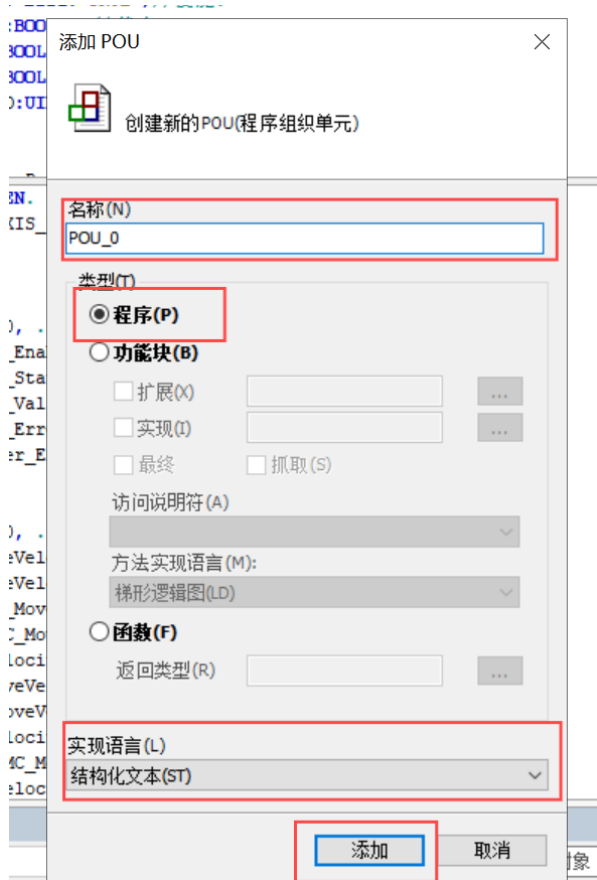
以上脉冲输出功能块是常用功能块, 具体功能说明请参考“高速IO说明书”

### 8.5.3 创建POU程序

右击“Application”，点击“添加对象”，选择“POU”后会跳出添加页面。

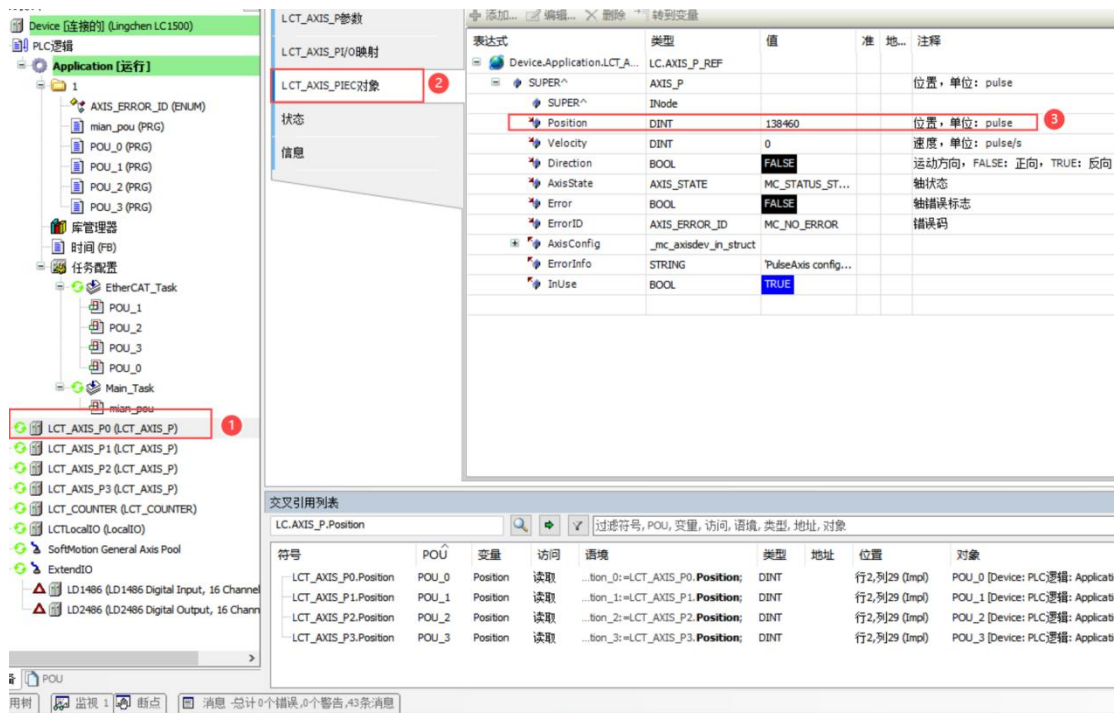


起定名称后选择“程序”，这里先选择“结构化文本”语言，最后点击“添加”即可。



### 8.5.4 获取脉冲轴位置

获取脉冲轴实际运行位置数据映射到程序中，方便本次测试观察，以轴“LCT\_AXIS\_P0”为例：如下图，程序登录进PLC，点击“LCT\_AXIS\_P0”脉冲轴，选择“LCT\_AXIS\_PIEC对象”，打开表达式列表，其中的“Position”便是此轴的实际轴位置，此外还有轴状态、轴错误码等轴信息，可根据需求应用到程序POU中。



在刚建立的POU中定义个轴位置变量如下图“1”步，将上面所找到的轴位置赋值到定义的变量上，如下图“2”步，其中“LCT\_AXIS\_P0”是轴名称，“Position”是指次轴的位置。

```

164     X8 AT %QX1.0:BOOL; //原点DI.
165     X9 AT %QX1.1:BOOL; //限位DI.
166     Axisposition_0:DINT; //轴位置.
167 END_VAR.

1 IF_MC Power_Valid THEN.
2   Axisposition_0:=LCT_AXIS_P0.Position;.
3 END_IF.
4 //轴使能.
5 _MC_Power( .
6   Axis:= LCT_AXIS_P0, .
7   Enable:=_MC_Power_Enable, .
8   Status=>_MC_Power_Status, .
9   Valid=>_MC_Power_Valid, .
10  Error=>_MC_Power_Error, .
11  ErrorID=>_MC_Power_ErrorID);.
    
```

### 8.5.5 功能块的POU应用

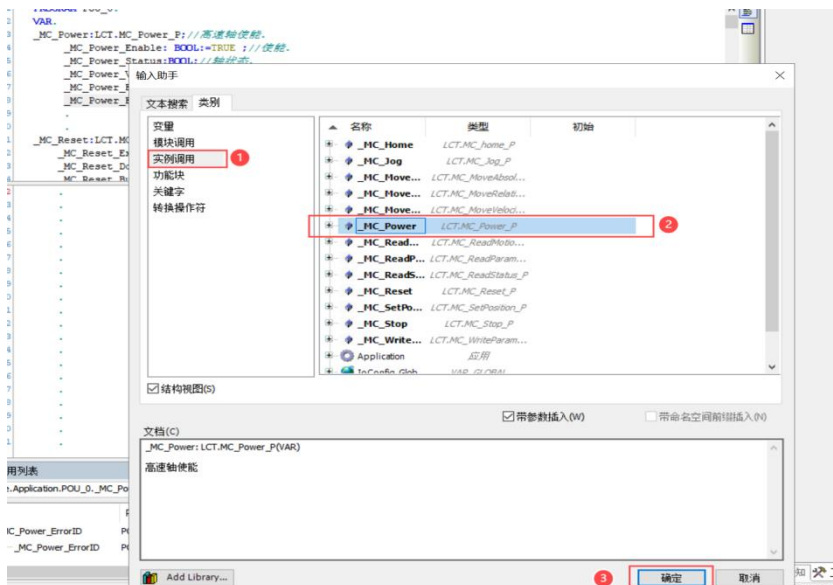
先在POU中声明功能块使其实例化，以“MC\_Power\_P”为例，并声明出功能块所用引角的变量，如下图：

```

Device POU_0 x LCT_AXIS_P0
1 PROGRAM POU_0
2 VAR
3   _MC_Power:LCT.MC_Power_P; //高速轴使能
4   _MC_Power_Enable: BOOL; //使能
5   _MC_Power_Status:BOOL; //轴状态
6   _MC_Power_Valid:BOOL; //完成标志
7   _MC_Power_Error:BOOL; //错误标志
8   _MC_Power_ErrorID:UINT; //错误ID

```

将实例化好的模块调用到程序中，在下面程序编辑中按“F2”按钮打开输入助手页面，点击实例调用，选择刚声明好的“\_MC\_Power\_P”，点击“确定”即可。



调用好后如下，然后绑定上面定义好的各功能变量。功能块操作说明：1.Axis引脚绑定添加的轴名称；2.Enable使能轴功能块，高电平有效，Status和Valid置TRUE，功能块使能完成；3.此功能块为使能高速脉冲输出指令，如果此功能块轴使能没有完成，其他所以功能块都无法运行。

```

_MC_Power(
  Axis:= LCT_AXIS_P0, //轴名称
  Enable:= _MC_Power_Enable, //使能, 电平触发, TRUE: 使能功能块
  Status=> _MC_Power_Status, //轴状态, TRUE: 高速轴使能
  Valid=> _MC_Power_Valid, //完成标志, TRUE: 功能块有效
  Error=> _MC_Power_Error, //错误标志, TRUE: 功能块内部发生错误
  ErrorID=> _MC_Power_ErrorID); //错误码, 参考轴错误ID说明

```

按此方法可声明调用所需的其他功能块。

### 8.5.6 MC\_ReadMotionStatus\_P的应用

本功能块主要实现高速轴读取轴运动状态功能，便于测试观察轴的运动。

以轴“LCT\_AXIS\_P”为例，声明高速轴功能块“  
\_MC\_ReadMotionStatus:LCT.MC\_ReadMotionStatus\_P”实例化，并定义出用于功能引角的变量。

```

Device | POU_0 x | LCT_AXIS_P0
152  _MC_ReadMotionStatus:LCT.MC_ReadMotionStatus_P; //高速轴读取轴运动状态
153  _MC_ReadMotionStatus_Enable:BOOL; //使能
154  _MC_ReadMotionStatus_Valid:BOOL; //有效标志
155  _MC_ReadMotionStatus_Busy:BOOL; //执行标志
156  _MC_ReadMotionStatus_Error:BOOL; //错误标志
157  _MC_ReadMotionStatus_ErrorID:UINT; //错误ID
158  _MC_ReadMotionStatus_ConstanVelocity:BOOL; //匀速运动
159  _MC_ReadMotionStatus_Accelerating:BOOL; //加速运动
160  _MC_ReadMotionStatus_Decelerating:BOOL; //减速运动
161  _MC_ReadMotionStatus_DirectionPositive:BOOL; //正向运动
162  _MC_ReadMotionStatus_DirectionNegative:BOOL; //反向运动
    
```

此功能块操作说明：1. Axis引脚绑定添加的轴名称；2. Enable使能轴功能块，高电平有效，Valid置TRUE，功能块使能完成；3. Busy引脚，执行标志，置TRUE，功能块正在执行；4. ConstanVelocity引脚，匀速运动，轴匀速运动时置TRUE；5. Accelerating引脚，加速运动，轴加速运动时置TRUE；6. Decelerating引脚，减速运动，轴减速运动时置TRUE；7. DirectionPositive引脚，正向运动，轴正向运动时置TRUE；8. DirectionNegative引脚，反向运动，轴反向运动时置TRUE。

```

135  _MC_ReadMotionStatus(
136  Axis:=LCT_AXIS_P0 , //轴名称
137  Enable:=_MC_ReadMotionStatus_Enable, //使能, 电平触发, TRUE: 使能功能块
138  Valid=> _MC_ReadMotionStatus_Valid, //有效标志, TRUE: 功能块有效
139  Busy=> _MC_ReadMotionStatus_Busy, //执行标志, TRUE: 功能块正在执行
140  Error=> _MC_ReadMotionStatus_Error, //错误标志, TRUE: 功能块内部发生错误
141  ErrorID=> _MC_ReadMotionStatus_ErrorID , //错误码, 参考轴错误ID说明
142  ConstanVelocity=> _MC_ReadMotionStatus_ConstanVelocity, //匀速运动
143  Accelerating=> _MC_ReadMotionStatus_Accelerating , //加速运动
144  Decelerating=> _MC_ReadMotionStatus_Decelerating , //减速运动
145  DirectionPositive=> _MC_ReadMotionStatus_DirectionPositive , //正向运动
146  DirectionNegative=> _MC_ReadMotionStatus_DirectionNegative ); //反向运动
    
```

### 8.5.7 MC\_Reset\_P的应用

本功能块主要实现高速轴错误复位操作。

声明高速轴功能块 “\_MC\_Reset:LCT.MC\_Reset\_P;” 实例化并定义出用于功能引角的变量。

```

Device | POU_0 x | LCT_AXIS_P0
10
11  _MC_Reset:LCT.MC_Reset_P; //高速轴复位
12  _MC_Reset_Execute:BOOL; //复位
13  _MC_Reset_Done:BOOL; //完成标志
14  _MC_Reset_Busy:BOOL; //执行标志
15  _MC_Reset_Error:BOOL; //错误标志
16  _MC_Reset_ErrorID:UINT; //错误ID
    
```

此模块功能块操作说明：1. Axis引脚绑定添加的轴名称；2. Execute引脚，默认FALSE，置TRUE上升沿使能功能块；3. Done引脚，完成标志，置TRUE，高速轴复位完成；4. Busy引脚，执行标志，置TRUE，高速轴正在复位。

```

23  _MC_Reset (
24  Axis:=LCT_AXIS_P0 , //轴名称
25  Execute:=_MC_Reset_Execute, //使能, 上升沿触发, TRUE: 使能功能块
26  Done=> _MC_Reset_Done, //完成标志, TRUE: 高速轴复位完成
27  Busy=>_MC_Reset_Busy , //执行标志, TRUE: 高速轴正在复位
28  Error=> _MC_Reset_Error, //错误标志, TRUE: 功能块内部发生错误
29  ErrorID=>_MC_Reset_ErrorID); //错误码, 参考轴错误ID说明
    
```

### 8.5.8 MC\_MoveAbsolute\_P的应用

本功能块主要实现高速轴绝对定位控制，Position 数据为轴的绝对位置。加减速过程中不允许改变速度。

声明高速轴功能块 “\_MC\_MoveAbsolute:LCT.MC\_MoveAbsolute\_P;” 实例化，并定义出用于功能引角的变量。

```

18
19  _MC_MoveAbsolute:LCT.MC_MoveAbsolute_P; //高速轴绝对位置控制
20  _MC_MoveAbsolute_Execute:BOOL; //使能
21  _MC_MoveAbsolute_Position:DINT:=100000; //绝对位置
22  _MC_MoveAbsolute_Velocity:UDINT:=10000; //速度
23  _MC_MoveAbsolute_Acceleration:UDINT:=100000; //加速度
24  _MC_MoveAbsolute_Deceleration:UDINT:=100000; //减速度
25  _MC_MoveAbsolute_Jerk:UDINT:=2100000000; //加加速度
26  _MC_MoveAbsolute_Done:BOOL; //完成标志
27  _MC_MoveAbsolute_Busy:BOOL; //执行标志
28  _MC_MoveAbsolute_CommandAborted:BOOL; //终止标志
29  _MC_MoveAbsolute_Error:BOOL; //错误标志
30  _MC_MoveAbsolute_ErrorID:UINT; //错误ID

```

此功能块操作说明：1.Axis引脚绑定添加的轴名称；2.Execute引脚，默认FALSE，置TRUE上升沿使能功能块；3.Position引脚，轴运动目标绝对位置；4.Done引脚，完成标志，置TRUE，功能块执行完成；5.Busy引脚，执行标志，置TRUE，功能块正在执行；6.CommandAborted引脚，中止标志，功能块执行过程中被打断时置TRUE。

```

97  _MC_MoveAbsolute(
98  Axis:= LCT_AXIS_P0, //轴名称
99  Execute:=_MC_MoveAbsolute_Execute, //使能, 上升沿触发, TRUE: 使能功能块
100 Position:=_MC_MoveAbsolute_Position, //绝对位置, 单位: pulse
101 Velocity:=_MC_MoveAbsolute_Velocity, //定位速度, 单位: pulse/s
102 Acceleration:=_MC_MoveAbsolute_Acceleration, //加速度, 单位: pulse/s²
103 Deceleration:=_MC_MoveAbsolute_Deceleration, //减速度, 单位: pulse/s²
104 Jerk:=_MC_MoveAbsolute_Jerk, //加加速度, 单位: pulse/s³
105 Done=> _MC_MoveAbsolute_Done, //完成标志, TRUE: 功能块执行完成
106 Busy=> _MC_MoveAbsolute_Busy, //执行标志, TRUE: 功能块正在执行
107 CommandAborted=>_MC_MoveAbsolute_CommandAborted, //终止标志, TRUE: 功能块被终止
108 Error=> _MC_MoveAbsolute_Error, //错误标志, TRUE: 功能块内部发生错误
109 ErrorID=>_MC_MoveAbsolute_ErrorID ); //错误码, 参考轴错误ID说明

```

#### 【绝对定位调试实例】

以轴“LCT\_AXIS\_P0”的为实例如下所示：当轴使能模块使能完成后，此时轴实际位置“Axisposition\_0”为0，绝对位置“MC\_MoveAbsolute\_Posit”设为100000，设置好速度值后，点击绝对位置触发“\_MC\_MoveAbsolute\_Execute”预值<TRUE>，后右击选择写入其所有值，步“3”或者按“Ctrl+F7”。

|                |      |       |        |      |
|----------------|------|-------|--------|------|
| X9             | BIT  | FALSE | %QX1.1 | 限位DI |
| Axisposition_0 | DINT | 0     |        | 轴位置  |

```

100 CommandAbortedFALSE=>_MC_MoveRelative_CommandAbortedFALSE, .
101 ErrorFALSE=>_MC_MoveRelative_ErrorFALSE, .
102 ErrorID[MC_NO_ERROR]>=>_MC_MoveRelative_ErrorID(0));.
103 //轴绝对位置控制.
104 ● _MC_MoveAbsolute(.
105   Axis:= LCT_AXIS_P0, .
106   ExecuteFALSE := _MC_MoveAbsolute_Execute FALSE<TRUE>, .
107   Position100000 := _MC_MoveAbsolute_Position 100000, .
108   Velocity10000 := _MC_MoveAbsolute_Velocity 10000, .
109   Acceleration100000 := _MC_MoveAbsolute_Acceleration 100000, .
110   Deceleration100000 := _MC_MoveAbsolute_Deceleration 100000, .
111   Jerk2100000000 := _MC_MoveAbsolute_Jerk 2100000000, .
112   DoneFALSE=>_MC_MoveAbsolute_DoneFALSE, .
113   BusyFALSE=>_MC_MoveAbsolute_BusyFALSE, .
114   CommandAbortedFALSE=>_MC_MoveAbsolute_CommandAbortedFALSE, .
115   ErrorFALSE=>_MC_MoveAbsolute_ErrorFALSE, .
116   ErrorID[MC_NO_ERROR]>=>_MC_MoveAbsolute_ErrorID(0));.
117
118 //高速轴速度参数.
119 ● _MC_ReadParameter(.
120   Axis:= LCT_AXIS_P0, .
121   EnableFALSE := _MC_ReadParameter_Enable FALSE, .
122   ParameterNumber220 := _MC_ReadParameter_ParameterNumber 220, .
123   ValidFALSE=>_MC_ReadParameter_ValidFALSE, .

```

绝对位置触发后，电机以1000pul/s的速度开始运行至\_MC\_MoveAbsolute\_Done为TRUE时停止，此时测试轴位置Axisposition\_0为100000，正好等于设置值。

|                |      |        |  |     |
|----------------|------|--------|--|-----|
| Axisposition_0 | DINT | 100000 |  | 轴位置 |
|----------------|------|--------|--|-----|

```

100 CommandAbortedFALSE=>_MC_MoveRelative_CommandAbortedFALSE, .
101 ErrorFALSE=>_MC_MoveRelative_ErrorFALSE, .
102 ErrorID[MC_NO_ERROR]>=>_MC_MoveRelative_ErrorID(0));.
103 //轴绝对位置控制.
104 ● _MC_MoveAbsolute(.
105   Axis:= LCT_AXIS_P0, .
106   ExecuteTRUE := _MC_MoveAbsolute_Execute TRUE, .
107   Position100000 := _MC_MoveAbsolute_Position 100000, .
108   Velocity10000 := _MC_MoveAbsolute_Velocity 10000, .
109   Acceleration100000 := _MC_MoveAbsolute_Acceleration 100000, .
110   Deceleration100000 := _MC_MoveAbsolute_Deceleration 100000, .
111   Jerk2100000000 := _MC_MoveAbsolute_Jerk 2100000000, .
112   DoneTRUE=>_MC_MoveAbsolute_DoneTRUE, .
113   BusyFALSE=>_MC_MoveAbsolute_BusyFALSE, .
114   CommandAbortedFALSE=>_MC_MoveAbsolute_CommandAbortedFALSE, .
115   ErrorFALSE=>_MC_MoveAbsolute_ErrorFALSE, .
116   ErrorID[MC_NO_ERROR]>=>_MC_MoveAbsolute_ErrorID(0));.
117

```

在设置值为100000不变情况下，再次触发“\_MC\_MoveAbsolute\_Execute”，给其上值TRUE后，轴位置仍然为100000等于设置值，电机也处在当前位置不会运动。

|                |      |        |  |     |
|----------------|------|--------|--|-----|
| Axisposition_0 | DINT | 100000 |  | 轴位置 |
|----------------|------|--------|--|-----|

```

103 //轴绝对位置控制.
104 ● _MC_MoveAbsolute(.
105   Axis:= LCT_AXIS_P0, .
106   ExecuteTRUE := _MC_MoveAbsolute_Execute TRUE, .
107   Position100000 := _MC_MoveAbsolute_Position 100000, .
108   Velocity10000 := _MC_MoveAbsolute_Velocity 10000, .
109   Acceleration100000 := _MC_MoveAbsolute_Acceleration 100000, .
110   Deceleration100000 := _MC_MoveAbsolute_Deceleration 100000, .
111   Jerk2100000000 := _MC_MoveAbsolute_Jerk 2100000000, .
112   DoneTRUE=>_MC_MoveAbsolute_DoneTRUE, .
113   BusyFALSE=>_MC_MoveAbsolute_BusyFALSE, .

```

### 8.5.9 MC\_MoveRelative\_P 的应用

本功能块主要实现高速轴轴相对定位控制，Distance数据为轴的相对位置。加减速过程中不允许改变速度。

声明高速轴功能块“\_MC\_MoveRelative:LCT.MC\_MoveRelative\_P;”实例化，并定义出用于功能引角的变量。

```

Device | POU_0 x | LCT_AXIS_P0
32
33  _MC_MoveRelative:LCT.MC_MoveRelative_P; //高速轴相对位置控制
34  _MC_MoveRelative_Execute:BOOL; //使能
35  _MC_MoveRelative_Distance:DINT:=100000; //绝对位置
36  _MC_MoveRelative_Velocity:UDINT:=10000; //速度
37  _MC_MoveRelative_Acceleration:UDINT:=100000; //加速度
38  _MC_MoveRelative_Deceleration:UDINT:=100000; //减速度
39  _MC_MoveRelative_Jerk:UDINT:=2100000000; //加加速度
40  _MC_MoveRelative_Done:BOOL; //完成标志
41  _MC_MoveRelative_Busy:BOOL; //执行标志
42  _MC_MoveRelative_CommandAborted:BOOL; //终止标志
43  _MC_MoveRelative_Error:BOOL; //错误标志
44  _MC_MoveRelative_ErrorID:UINT; //错误ID
    
```

此功能块操作说明：1.Axis引脚绑定添加的轴名称；2.Execute引脚，默认FALSE，置TRUE上升沿使能功能块；3.Distance引脚，轴运动相对位置，即相对于当前位置轴运行位置；4.Done引脚，完成标志，置TRUE，功能块执行完成；5.Busy引脚，执行标志，置TRUE，功能块正在执行；6.CommandAborted引脚，中止标志，功能块执行过程中被打断时置TRUE。

```

85  _MC_MoveRelative(
86  Axis:= LCT_AXIS_P0, //轴名称
87  Execute:= _MC_MoveRelative_Execute , //使能, 上升沿触发, TRUE: 使能功能块
88  Distance:= _MC_MoveRelative_Distance, //相对位置, 单位: pulse
89  Velocity:= _MC_MoveRelative_Velocity, //定位速度, 单位: pulse/s
90  Acceleration:= _MC_MoveRelative_Acceleration, //加速度, 单位: pulse/s²
91  Deceleration:= _MC_MoveRelative_Deceleration, //减速度, 单位: pulse/s²
92  Jerk:= _MC_MoveRelative_Jerk, //加加速度, 单位: pulse/s³
93  Done=> _MC_MoveRelative_Done , //完成标志, TRUE: 功能块执行完成
94  Busy=> _MC_MoveRelative_Busy , //执行标志, TRUE: 功能块正在执行
95  CommandAborted=> _MC_MoveRelative_CommandAborted , //终止标志, TRUE: 功能块被终止
96  Error=> _MC_MoveRelative_Error , //错误标志, TRUE: 功能块内部发生错误
97  ErrorID=> _MC_MoveRelative_ErrorID ); //错误码, 参考轴错误ID说明
    
```

#### 【相对定位调试实例】

以轴“LCT\_AXIS\_P0”的为实例如下所示：当轴使能模块使能完成后，轴实际位置“Axisposition\_0”为0时，相对位置值\_MC\_MoveRelative\_Distance设为100000，触发\_MC\_MoveRelative\_Execute启动模块运行，同时所接的电机开始运动。

```

Axisposition_0      DINT      0      轴位置
//轴相对位置控制.
MC_MoveRelative(.
  Axis:= LCT_AXIS_P0, .
  ExecuteFALSE:= _MC_MoveRelative_Execute FALSE<TRUE>, .
  Distance_100000 := _MC_MoveRelative_Distance_100000, .
  Velocity_10000 := _MC_MoveRelative_Velocity_10000, .
  Acceleration_100000 := _MC_MoveRelative_Acceleration_100000, .
  Deceleration_100000 := _MC_MoveRelative_Deceleration_100000, .
  Jerk_2100000000 := _MC_MoveRelative_Jerk_2100000000, .
  DoneFALSE=> _MC_MoveRelative_DoneFALSE, .
  BusyFALSE=> _MC_MoveRelative_BusyFALSE, .
  CommandAbortedFALSE=> _MC_MoveRelative_CommandAbortedFALSE, .
  ErrorFALSE=> _MC_MoveRelative_ErrorFALSE, .
  ErrorID[MC_NO_ERROR]>=> _MC_MoveRelative_ErrorID_0 );.
//轴绝对位置控制.

```

\_MC\_MoveRelative\_Done亮时相对运动结束，电机也停止运动，此时轴位置等于设置值100000。

```

Axisposition_0      DINT      100000      轴位置
//轴相对位置控制.
MC_MoveRelative(.
  Axis:= LCT_AXIS_P0, .
  ExecuteTRUE:= _MC_MoveRelative_Execute TRUE, .
  Distance_100000 := _MC_MoveRelative_Distance_100000, .
  Velocity_10000 := _MC_MoveRelative_Velocity_10000, .
  Acceleration_100000 := _MC_MoveRelative_Acceleration_100000, .
  Deceleration_100000 := _MC_MoveRelative_Deceleration_100000, .
  Jerk_2100000000 := _MC_MoveRelative_Jerk_2100000000, .
  DoneTRUE=> _MC_MoveRelative_DoneTRUE, .
  BusyFALSE=> _MC_MoveRelative_BusyFALSE, .
  CommandAbortedFALSE=> _MC_MoveRelative_CommandAbortedFALSE, .
  ErrorFALSE=> _MC_MoveRelative_ErrorFALSE, .
  ErrorID[MC_NO_ERROR]>=> _MC_MoveRelative_ErrorID_0 );.
//轴绝对位置控制.

```

保持\_MC\_MoveRelative\_Distance值为100000不变，再次触发MC\_MoveRelative\_Execute,模块程序和电机再次运行至轴位置为200000时停止，可见此功能块不同与绝对运动，而是可以再当前位置上不断做相对运动。

```

Axisposition_0      DINT      200000      轴位置
//轴相对位置控制.
MC_MoveRelative(.
  Axis:= LCT_AXIS_P0, .
  ExecuteTRUE:= _MC_MoveRelative_Execute TRUE, .
  Distance_100000 := _MC_MoveRelative_Distance_100000, .
  Velocity_10000 := _MC_MoveRelative_Velocity_10000, .
  Acceleration_100000 := _MC_MoveRelative_Acceleration_100000, .
  Deceleration_100000 := _MC_MoveRelative_Deceleration_100000, .
  Jerk_2100000000 := _MC_MoveRelative_Jerk_2100000000, .
  DoneTRUE=> _MC_MoveRelative_DoneTRUE, .
  BusyFALSE=> _MC_MoveRelative_BusyFALSE, .
  CommandAbortedFALSE=> _MC_MoveRelative_CommandAbortedFALSE, .
  ErrorFALSE=> _MC_MoveRelative_ErrorFALSE, .
  ErrorID[MC_NO_ERROR]>=> _MC_MoveRelative_ErrorID_0 );.
//轴绝对位置控制.

```

### 8.5.10 MC\_SetPosition\_P 的应用

本功能块主要实现高速轴位置设定功能。当Relative=FALSE时设置的位置为绝对位置，当Relative=TRUE时设置的位置为相对位置，即当前位置=当前位置+Position。

声明计数器 “\_MC\_SetPosition:LCT.MC\_SetPosition\_P;” 实例化，并定义出用于功能引角的变量：

```

Device | POU_0 x | LCT_AXIS_P0
102  _MC_SetPosition:LCT.MC_SetPosition_P; //高速轴设置位置
103  _MC_SetPosition_Execute:BOOL; //使能
104  _MC_SetPosition_Position:DINT:=100000; //位置
105  _MC_SetPosition_Relative:BOOL; //相对位置
106  _MC_SetPosition_Done:BOOL; //完成标志
107  _MC_SetPosition_Busy:BOOL; //执行标志
108  _MC_SetPosition_Error:BOOL; //错误标志
109  _MC_SetPosition_ErrorID:UINT; //错误ID
    
```

此功能块操作说明：1. Axis引脚绑定添加的轴名称；2. Execute引脚，默认FALSE，置TRUE上升沿使能功能块；3. Position引脚，设定位置；4. Relative引脚，默认FALSE，绝对位置，触发时将轴位置设置为Position所设值，置TRUE，相对位置，触发时将轴位置设置为当前位置+Position所设值；5. Done引脚，完成标志，置TRUE，功能块执行完成；6. Busy引脚，执行标志，置TRUE，功能块正在执行。

```

59  _MC_SetPosition(
60  Axis:=LCT_AXIS_P0, //轴名称
61  Execute:=_MC_SetPosition_Execute, //使能, 上升沿触发, TRUE: 使能功能块
62  Position:=_MC_SetPosition_Position, //位置, 单位: pulse
63  Relative:=_MC_SetPosition_Relative, //相对位置, FALSE: 绝对位置, TRUE: 相对位置
64  Done=>_MC_SetPosition_Done, //完成标志, TRUE: 功能块执行完成
65  Busy=>_MC_SetPosition_Busy, //执行标志, TRUE: 功能块正在执行
66  Error=>_MC_SetPosition_Error, //错误标志, TRUE: 功能块内部发生错误
67  ErrorID=>_MC_SetPosition_ErrorID ); //错误码, 参考轴错误ID说明
    
```

#### 【高速轴设置位置调试实例】

以轴“LCT\_AXIS\_P0”的为实例如下所示：当轴使能模块使能完成后，此时轴实际位置Axisposition\_0为0，设置位值\_MC\_SetPosition\_Position设为100000，触发\_MC\_SetPosition\_Execute后功能模块运行，但电机不会运动。

| Name           | Data Type | Value | Description |
|----------------|-----------|-------|-------------|
| Axisposition_0 | DINT      | 0     | 轴位置         |

```

62  //高速轴设置位置.
63  _MC_SetPosition(
64  Axis:=LCT_AXIS_P0,
65  ExecuteFALSE:=_MC_SetPosition_Execute FALSE<TRUE>, // 2
66  Position_100000:=_MC_SetPosition_Position_100000, // 3
67  RelativeFALSE:=_MC_SetPosition_RelativeFALSE,
68  DoneFALSE=>_MC_SetPosition_DoneFALSE,
69  BusyFALSE=>_MC_SetPosition_BusyFALSE,
70  ErrorFALSE=>_MC_SetPosition_ErrorFALSE,
71  ErrorID[MC_NO_ERROR]>=>_MC_SetPosition_ErrorID_0 );
72
73
74  //轴回原.
75  _MC_Home(
    
```

\_MC\_SetPosition\_Done完成标志亮起，模块运行结束，此时实际位置为100000。

|                |      |        |     |
|----------------|------|--------|-----|
| Axisposition_0 | DINT | 100000 | 轴位置 |
|----------------|------|--------|-----|

```

62 .
63 //高速轴设置位置.
64 ● _MC_SetPosition(.
65     Axis:=LCT_AXIS_P0, .
66     Execute TRUE := _MC_SetPosition_Execute TRUE , .
67     Position 100000 := _MC_SetPosition_Position 100000 , .
68     Relative FALSE := _MC_SetPosition_Relative FALSE , .
69     Done TRUE => _MC_SetPosition_Done TRUE , .
70     Busy FALSE => _MC_SetPosition_Busy FALSE , .
71     Error FALSE => _MC_SetPosition_Error FALSE , .
72     ErrorID [MC_NO_ERROR] => _MC_SetPosition_ErrorID 0 ); .
73 .
74 //轴回原.
75 ● _MC_Home(.
    
```

再把设置位置改为50000，重新触发\_MC\_SetPosition\_Execute后，轴实际位置Axisposition\_0变为设置值50000，由此可见，此功能可设置改变标定轴位置值，但不需要电机运动。

|                |      |       |     |
|----------------|------|-------|-----|
| Axisposition_0 | DINT | 50000 | 轴位置 |
|----------------|------|-------|-----|

```

62 .
63 //高速轴设置位置.
64 ● _MC_SetPosition(.
65     Axis:=LCT_AXIS_P0, .
66     Execute TRUE := _MC_SetPosition_Execute TRUE , .
67     Position 50000 := _MC_SetPosition_Position 50000 , .
68     Relative FALSE := _MC_SetPosition_Relative FALSE , .
69     Done TRUE => _MC_SetPosition_Done TRUE , .
70     Busy FALSE => _MC_SetPosition_Busy FALSE , .
71     Error FALSE => _MC_SetPosition_Error FALSE , .
72     ErrorID [MC_NO_ERROR] => _MC_SetPosition_ErrorID 0 ); .
73 .
74 //轴回原.
    
```

### 8.5.11 MC\_Home\_P的应用

本功能块为使轴回零点功能块，LC1500目前配有9种回零方式，具体功能描述请参考另一份“高速IO”说明书。

声明高速轴功能块“\_MC\_Home:LCT.MC\_home\_P;”实例化，并定义出用于功能引角的变量：

```

Device POU_0 x LCT_AXIS_P0
61  _MC_Home:LCT.MC_home_P; //高速轴回零
62  _MC_Home_Execute:BOOL; //使能
63  _MC_Home_Position:DINT:=100000; //回零位置
64  _MC_Home_FirstVelocity:UDINT:=100000; //第一段速度
65  _MC_Home_SecondVelocity:UDINT:=100000; //第二段速度
66  _MC_Home_Acceleration:UDINT:=100000; //加速度
67  _MC_Home_Deceleration:UDINT:=100000; //减速度
68  _MC_Home_Jerk:UDINT:=2100000000; //加加速度
69  _MC_Home_Done:BOOL; //完成标志
70  _MC_Home_Busy:BOOL; //执行标志
71  _MC_Home_CommandAborted:BOOL; //终止标志
72  _MC_Home_Error:BOOL; //错误标志
73  _MC_Home_ErrorID:UINT; //错误ID

```

此功能块操作说明：1.Axis引脚绑定添加的轴名称；2.Execute引脚，默认FALSE，置TRUE上升沿使能功能块；3.Position引脚，轴回零位置；4.FirstVelocity引脚，第一段速度；5.SecondVelocity引脚，第二段速度；6.Done引脚，完成标志，置TRUE，功能块执行完成；Busy引脚，执行标志，置TRUE，功能块正在执行；7.CommandAborted引脚，中止标志，功能块执行过程中被打断时置TRUE。

```

71  _MC_Home (
72  Axis:= LCT_AXIS_P0, //轴名称
73  Execute:= _MC_Home_Execute, //使能, 上升沿触发, TRUE: 使能功能块
74  Position:= _MC_Home_Position, //回零位置, 单位: pulse
75  FirstVelocity:= _MC_Home_FirstVelocity, //第一段速度, 单位: pulse/s
76  SecondVelocity:= _MC_Home_SecondVelocity, //第二段速度, 单位: pulse/s
77  Acceleration:= _MC_Home_Acceleration, //加速度, 单位: pulse/s²
78  Deceleration:= _MC_Home_Deceleration, //减速度, 单位: pulse/s²
79  Jerk:= _MC_Home_Jerk, //加加速度, 单位: pulse/s³
80  Done=> _MC_Home_Done, //完成标志, TRUE: 功能块执行完成
81  Busy=> _MC_Home_Busy, //执行标志, TRUE: 功能块正在执行
82  CommandAborted=> _MC_Home_CommandAborted, //终止标志, TRUE: 功能块被终止
83  Error=> _MC_Home_Error, //错误标志, TRUE: 功能块内部发生错误
84  ErrorID=> _MC_Home_ErrorID ); //错误码, 参考轴错误ID说明

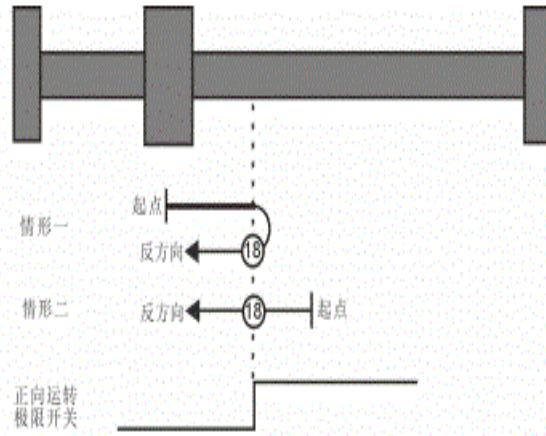
```

#### 【高速轴轴回零调试实例】

以回零方式3（DS402协议标注回零方式18）：正向运动找极限，即正向运动找极限的下降沿为例子，此回原方式功能说明：

情形一：当执行回零时，若正向极限开关状态处于低电平时，那么轴开始以第一段速度正向运动，当遇正向极限开关处于高电平时，轴运动方向改变且以第二段速度开始运动；在正向极限开关状态处于低电平时的位置就是原点位置。

情形二：当执行回零时，若正向极限开关处于高电平时，那么轴直接会以第二段速度开始反向运动，在正向极限开关状态处于低电平时的位置就是原点位置。



取决于正向运转极限开关的原点回零，上图中的表示18原点回零模式18

应用调试操作：以轴“LCT\_AXIS\_P0”的为例如下所示：打开轴参数配置，回原使能设TRUE，选择回原方式18，选X7为极限位。

|                          |                     |                      |
|--------------------------|---------------------|----------------------|
| SoftwareLimitNegPosition | DINT                | -5000000             |
| Home_P_Enable            | BOOL                | TRUE                 |
| Home_P_Method            | Enumeration of BYTE | 方式3-标准回零模式18-正向运动找极限 |
| Home_P_ZeroDINum         | Enumeration of BYTE | X6                   |
| Home_P_ZeroPolarity      | Enumeration of BYTE | 常开                   |
| Home_P_LimitDINum        | Enumeration of BYTE | X7                   |
| Home_P_LimitPolarity     | Enumeration of BYTE | 常开                   |

情形一：在限位处于低电平“1”步、轴位置为0“2”步时，分别设回原第一速度“3”步和第二速度“4”步的值，再触发\_MC\_Home\_Execute开启回原运动。

|                | DI1  | DI2   | DI3 | DI4 | 说明          |
|----------------|------|-------|-----|-----|-------------|
| X6             | BIT  | FALSE |     |     | %QX1.0 原点DI |
| X7             | BIT  | FALSE |     |     | %QX1.1 限位DI |
| Axisposition_0 | DINT | 0     |     |     | 轴位置         |

```

62 //轴回原.
63 ● _MC_Home(.
64     Axis:= LCT_AXIS_P0, .
65     ExecuteFALSE := _MC_Home_Execute FALSE <TRUE> , .
66     Position 0 := _MC_Home_Position 0 , .
67     FirstVelocity 100000 := _MC_Home_FirstVelocity 100000 3 .
68     SecondVelocity 10000 := _MC_Home_SecondVelocity 10000 4 .
69     Acceleration 100000 := _MC_Home_Acceleration 100000 , .
70     Deceleration 100000 := _MC_Home_Deceleration 100000 , .
71     Jerk 2100000000 := _MC_Home_Jerk 2100000000 , .
72     Done FALSE => _MC_Home_Done FALSE , .
73     Busy FALSE => _MC_Home_Busy FALSE , .
74     CommandAborted FALSE => _MC_Home_CommandAborted FALSE , .
75     Error FALSE => _MC_Home_Error FALSE , .
76     ErrorID [MC_NO_ERROR] => _MC_Home_ErrorID 0 );.
77 .
78 //高速轴设置位置.
79 ● _MC_SetPosition(.
80     Axis:=LCT_AXIS_P0
    
```

如“1”和“2”步所示，电机以第一速度100000开始做正向回原运动。

|  |      |        |        |      |
|--|------|--------|--------|------|
| ◆ _MC_ReadMotionStatus_Decelerating        | BOOL | FALSE  |        | 减速运动 |
| ◆ _MC_ReadMotionStatus_DirectionPositive ① | BOOL | TRUE   |        | 正向运动 |
| ◆ _MC_ReadMotionStatus_DirectionNegative   | BOOL | FALSE  |        | 反向运动 |
| ◆ X6                                       | BIT  | FALSE  | %QX1.0 | 原点DI |
| ◆ X7                                       | BIT  | FALSE  | %QX1.1 | 限位DI |
| ◆ Axisposition_0 ②                         | DINT | 781045 |        | 轴位置  |

```

62 //轴回原.
63 ● _MC_Home(.
64     Axis:= LCT_AXIS_P0, .
65     Execute TRUE := _MC_Home_Execute TRUE, .
66     Position 0 := _MC_Home_Position 0, .
67     FirstVelocity 100000 := _MC_Home_FirstVelocity 100000, .
68     SecondVelocity 10000 := _MC_Home_SecondVelocity 10000, .
69     Acceleration 100000 := _MC_Home_Acceleration 100000, .
70     Deceleration 100000 := _MC_Home_Deceleration 100000, .
71     Jerk 2100000000 := _MC_Home_Jerk 2100000000, .
72     Done FALSE => _MC_Home_Done FALSE, .
73     Busy TRUE => _MC_Home_Busy TRUE, ③
74     CommandAborted FALSE => _MC_Home_CommandAborted FALSE, .
75     Error FALSE => _MC_Home_Error FALSE, .
76     ErrorID [MC_NO_ERROR] => _MC_Home_ErrorID 0 );.
77 .
78 //高速轴设置位置.

```

如“2”步所示，手动触发变量X7为TRUE模拟正限位感应后，限位获得高电平且电机以第二速度开始做反向运动，“1”步所示。

|  |      |         |        |      |
|--|------|---------|--------|------|
| ◆ _MC_ReadMotionStatus_Decelerating        | BOOL | FALSE   |        | 减速运动 |
| ◆ _MC_ReadMotionStatus_DirectionPositive   | BOOL | FALSE   |        | 正向运动 |
| ◆ _MC_ReadMotionStatus_DirectionNegative ① | BOOL | TRUE    |        | 反向运动 |
| ◆ X6                                       | BIT  | FALSE   | %QX1.0 | 原点DI |
| ◆ X7 ②                                     | BIT  | TRUE    | %QX1.1 | 限位DI |
| ◆ Axisposition_0 ③                         | DINT | 3230755 |        | 轴位置  |

```

62 //轴回原.
63 ● _MC_Home(.
64     Axis:= LCT_AXIS_P0, .
65     Execute TRUE := _MC_Home_Execute TRUE, .
66     Position 0 := _MC_Home_Position 0, .
67     FirstVelocity 100000 := _MC_Home_FirstVelocity 100000, .
68     SecondVelocity 10000 := _MC_Home_SecondVelocity 10000, .
69     Acceleration 100000 := _MC_Home_Acceleration 100000, .
70     Deceleration 100000 := _MC_Home_Deceleration 100000, .
71     Jerk 2100000000 := _MC_Home_Jerk 2100000000, .
72     Done FALSE => _MC_Home_Done FALSE, .
73     Busy TRUE => _MC_Home_Busy TRUE, ④
74     CommandAborted FALSE => _MC_Home_CommandAborted FALSE, .
75     Error FALSE => _MC_Home_Error FALSE, .
76     ErrorID [MC_NO_ERROR] => _MC_Home_ErrorID 0 );.
77 .
78 //高速轴设置位置.
79 ● _MC_SetPosition(.

```

如“1”步，手动给限位FALSE模拟正限位下降沿后，回原完成标志亮起“3”步，轴位置为0，本次回原结束。

|  |      |       |        |      |
|--|------|-------|--------|------|
| ◆ _MC_ReadMotionStatus_Decelerating      | BOOL | FALSE |        | 减速运动 |
| ◆ _MC_ReadMotionStatus_DirectionPositive | BOOL | FALSE |        | 正向运动 |
| ◆ _MC_ReadMotionStatus_DirectionNegative | BOOL | FALSE |        | 反向运动 |
| ◆ X6                                     | BIT  | FALSE | %QX1.0 | 原点DI |
| ◆ X7                                     | BIT  | FALSE | %QX1.1 | 限位DI |
| ◆ Axisposition_0                         | DINT | 0     |        | 轴位置  |

```

62 //轴回原.
63 ● _MC_Home (.
64     Axis:= LCT_AXIS_P0, .
65     Execute TRUE := _MC_Home_Execute TRUE, .
66     Position 0 := _MC_Home_Position 0, .
67     FirstVelocity 100000 := _MC_Home_FirstVelocity 100000, .
68     SecondVelocity 10000 := _MC_Home_SecondVelocity 10000, .
69     Acceleration 100000 := _MC_Home_Acceleration 100000, .
70     Deceleration 100000 := _MC_Home_Deceleration 100000, .
71     Jerk 2100000000 := _MC_Home_Jerk 2100000000, .
72     Done TRUE => _MC_Home_Done TRUE, ③
73     Busy FALSE => _MC_Home_Busy FALSE, .
74     CommandAborted FALSE => _MC_Home_CommandAborted FALSE, .
75     Error FALSE => _MC_Home_Error FALSE, .
76     ErrorID [MC_NO_ERRO] => _MC_Home_ErrorID 0 );.
77 .
78 //高速轴设置位置.
79 ● _MC_SetPosition (.
80     Axis:=LCT_AXIS_P0, .

```

情形二：在回原触发前就给限位TRUE信号，限位保持高电平，然后再触发回原“2”。

|  |      |       |        |      |
|--|------|-------|--------|------|
| ◆ _MC_ReadMotionStatus_DirectionNegative | BOOL | FALSE |        | 反向运动 |
| ◆ X6                                     | BIT  | FALSE | %QX1.0 | 原点DI |
| ◆ X7                                     | BIT  | TRUE  | %QX1.1 | 限位DI |
| ◆ Axisposition_0                         | DINT | 0     |        | 轴位置  |

```

62 //轴回原.
63 ● _MC_Home (.
64     Axis:= LCT_AXIS_P0, .
65     Execute FALSE := _MC_Home_Execute FALSE < TRUE >, ②
66     Position 0 := _MC_Home_Position 0, .
67     FirstVelocity 100000 := _MC_Home_FirstVelocity 100000, .
68     SecondVelocity 10000 := _MC_Home_SecondVelocity 10000, .
69     Acceleration 100000 := _MC_Home_Acceleration 100000, .
70     Deceleration 100000 := _MC_Home_Deceleration 100000, .
71     Jerk 2100000000 := _MC_Home_Jerk 2100000000, .
72     Done FALSE => _MC_Home_Done FALSE, .
73     Busy FALSE => _MC_Home_Busy FALSE, .
74     CommandAborted FALSE => _MC_Home_CommandAborted FALSE, .
75     Error FALSE => _MC_Home_Error FALSE, .
76     ErrorID [MC_NO_ERRO] => _MC_Home_ErrorID 0 );.
77 .
78 //高速轴设置位置.

```

如“1”、“2”所示，回原触发后电机就立即以第二速度做反向回原运动。

|  |      |       |        |      |
|--|------|-------|--------|------|
| ◆ _MC_ReadMotionStatus_Decelerating      | BOOL | FALSE |        | 减速运动 |
| ◆ _MC_ReadMotionStatus_DirectionPositive | BOOL | FALSE |        | 正向运动 |
| ◆ _MC_ReadMotionStatus_DirectionNegative | 1    | BOOL  | TRUE   | 反向运动 |
| ◆ X6                                     | BIT  | FALSE | %QX1.0 | 原点DI |
| ◆ X7                                     | BIT  | TRUE  | %QX1.1 | 限位DI |
| ◆ Axisposition_0                         | 2    | DINT  | -47615 | 轴位置  |

```

62 //轴回原.
63 ● _MC_Home (.
64   Axis:= LCT_AXIS_P0, .
65   Execute TRUE := _MC_Home_Execute TRUE 3
66   Position 0 := _MC_Home_Position 0, .
67   FirstVelocity 100000 := _MC_Home_FirstVelocity 100000, .
68   SecondVelocity 10000 := _MC_Home_SecondVelocity 10000, .
69   Acceleration 100000 := _MC_Home_Acceleration 100000, .
70   Deceleration 100000 := _MC_Home_Deceleration 100000, .
71   Jerk 2100000000 := _MC_Home_Jerk 2100000000, .
72   Done FALSE => _MC_Home_Done FALSE, .
73   Busy TRUE := _MC_Home_Busy TRUE 4
74   CommandAborted FALSE => _MC_Home_CommandAborted FALSE, .
75   Error FALSE => _MC_Home_Error FALSE, .
76   ErrorID [MC_NO_ERROR] => _MC_Home_ErrorID 0 );.
77 .
78 //高速轴设置位置.

```

手动给限位FALSE，限位获得下降沿的同时回原完成标志亮起“3”步，轴位置为0，回原完成。

|  |      |       |        |             |
|--|------|-------|--------|-------------|
| ◆ _MC_ReadMotionStatus_Decelerating      | BOOL | FALSE |        | 减速运动        |
| ◆ _MC_ReadMotionStatus_DirectionPositive | BOOL | FALSE |        | 正向运动        |
| ◆ _MC_ReadMotionStatus_DirectionNegative | BOOL | FALSE |        | 反向运动        |
| ◆ X6                                     | BIT  | FALSE | %QX1.0 | 原点DI        |
| ◆ X7                                     | 1    | BIT   | FALSE  | %QX1.1 限位DI |
| ◆ Axisposition_0                         | 2    | DINT  | 0      | 轴位置         |

```

62 //轴回原.
63 ● _MC_Home (.
64   Axis:= LCT_AXIS_P0, .
65   Execute TRUE := _MC_Home_Execute TRUE, .
66   Position 0 := _MC_Home_Position 0, .
67   FirstVelocity 100000 := _MC_Home_FirstVelocity 100000, .
68   SecondVelocity 10000 := _MC_Home_SecondVelocity 10000, .
69   Acceleration 100000 := _MC_Home_Acceleration 100000, .
70   Deceleration 100000 := _MC_Home_Deceleration 100000, .
71   Jerk 2100000000 := _MC_Home_Jerk 2100000000, .
72   Done TRUE := _MC_Home_Done TRUE 3
73   Busy FALSE => _MC_Home_Busy FALSE, .
74   CommandAborted FALSE => _MC_Home_CommandAborted FALSE, .
75   Error FALSE => _MC_Home_Error FALSE, .
76   ErrorID [MC_NO_ERROR] => _MC_Home_ErrorID 0 );.
77 .
78 //高速轴设置位置.

```