Application Note

基于 ZMC600E 系列 PLC 搭建 IO 及电机控制应用

E 系列高速 IO 模块

AN01010101 1.00 Date:2025/11/18

类别	内容
关键词	集中式远程 IO 从站、集中式步进电机驱动从站、AWPLC、ZMC600E-Px PLC 运动控制器
摘要	致远主站系列产品及 E 系列高速 IO 联调示例

修订历史

版本	日期	原因
V1.00	2025/11/18	创建文档

目 录

1. 适用范围

本文介绍了致远电子开发的 PLC 运动控制器及 AWPLC，与 E 系列集中式高速 IO 从站联调，搭建 IO 及电机控制的示例。

其中 PLC 运动控制器产品包括 ZMC600E-PX、ZMC900E-Px 两个系列。

从站产品包括 E 系列集中式数字输入模块、数字输出模块以及步进电机控制模块等。

两款 PLC 控制器对应的 AWPLC 运行时版本和 AWStudio 版本如下：

- PLC 运动控制器：ZMC600E-P8/ZMC600E-P16/ZMC600E-P32
- AWPLC 运行时版本：awplc_runtime_zmc600e-2.0.0-rc.65
- AWStudio 版本：2.2.0-rc.98

以及：

- PLC 运动控制器：ZMC900E-P16/ZMC900E-P32/ZMC900E-P64
- AWPLC 运行时版本：awplc_runtime_zmc900e-1.0.0-rc.21
- AWStudio 版本：2.0.2-rc.3

2. 基于 ZMC600E-P16 系列 PLC 运动控制器的搭建

ZMC600E-Px/ZMC900E-Px 系列 PLC 运动控制器搭载了 AWPLC 运行时，用户通过致远电子基于 IEC61131-3 标准自研的 AWPLC 框架，在 AWStudio 上编写 PLC 程序并编译部署到 PLC 控制器运行。在 PLC 程序中操作 PDO 数据、请求 SDO 读写命令，就可以控制从站设备，实现电机运动。

图 2.1　AWPLC 设备开发框架图

2.1　测试环境

主站设备：

- ZMC600E-P16 PLC 运动控制器

图 2.2　ZMC600E-P16

从站设备：

- ZPT-8080 耦合器
- ZDM-E0016N 数字输出模块
- ZDM-E1600N 数字输入模块
- ZMTI-EF1200 步进电机驱动模块

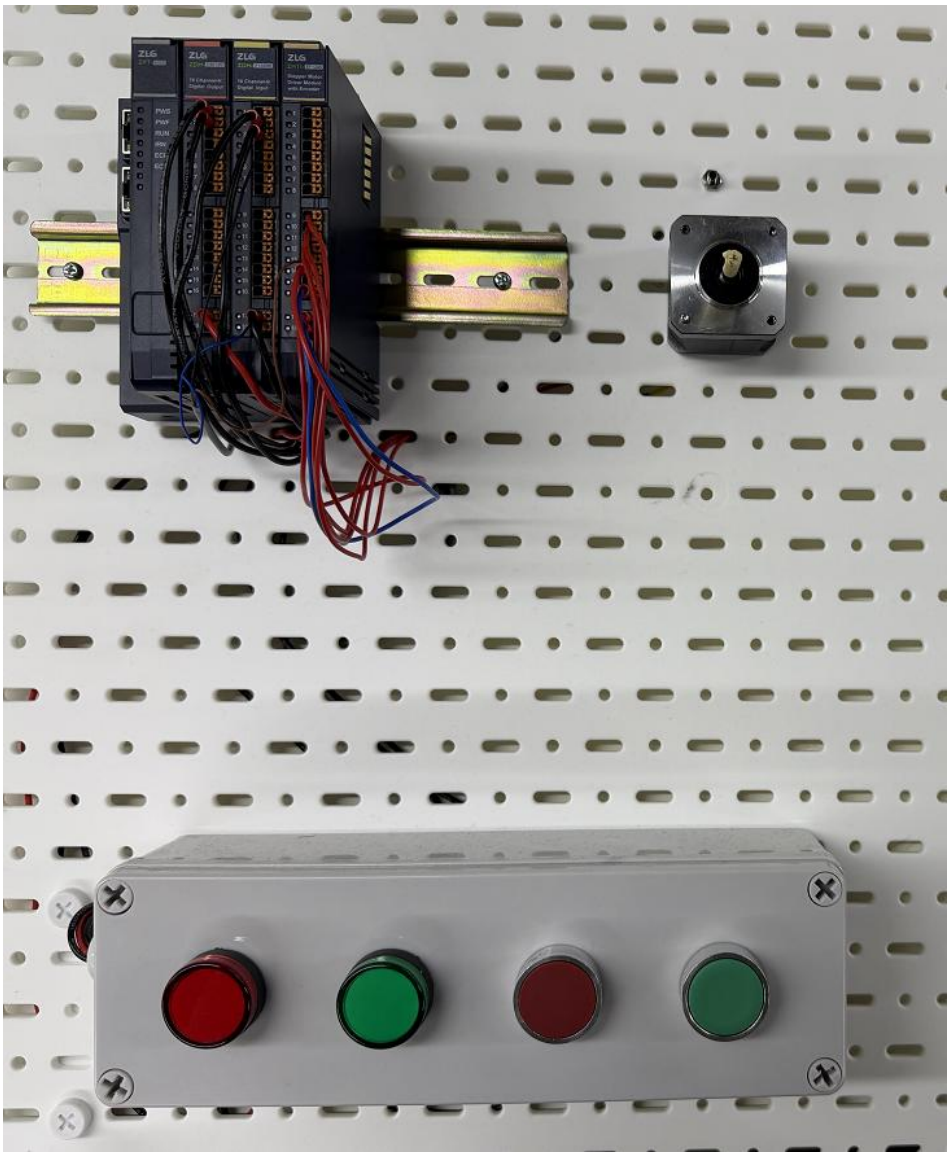

图 2.3　从站设备及其传感器执行器

其他硬件：

- 开发主机（Windows）
- 步进电机
- 指示灯（红、绿）
- 控制按钮（红、绿）
- 接近传感器
- 网线若干
- 24V 电源

软件环境：

- AWStudio 开发上位机

2.2 设备连接

用网线连接 ZMC600E-P16 控制器的 ECAT 口和 ZPT-8080 耦合器的 IN 口，将剩余模块顺次通过背板总线连接到耦合器。24V 电源分别供给 ZMC600E 主站和 ZPT-8080 耦合器。

然后还需要用网线连接 ZMC600E-P16 控制器的 NET2 网口和开发主机的网口。

指示灯的信号从 ZDM-E0016N 数字输出模块的输出引脚引出。按钮的信号接入到 ZDM-E1600N 数字输入模块的输入引脚。电机的驱动线按次序接到 ZMTI-EF1200 的 9~12 的端口，如果有编码器信号也接到 ZMTI-EF1200 上。此外，ZMTI-EF1200 还需要额外的 1 路电源输入作为电机驱动电源。

2.3 AWPLC 工程

2.3.1 新建工程

运行 AWStudio，在主工具栏中点击【新建】按钮，选择【运动控制器】-【AWPLC 编程】解决方案，下一步选择要使用的运动控制器型号 ZMC600E-P16，编程语言为 ST。

注：AWPLC 解决方案的路径不能有中文，同时也不能过深。

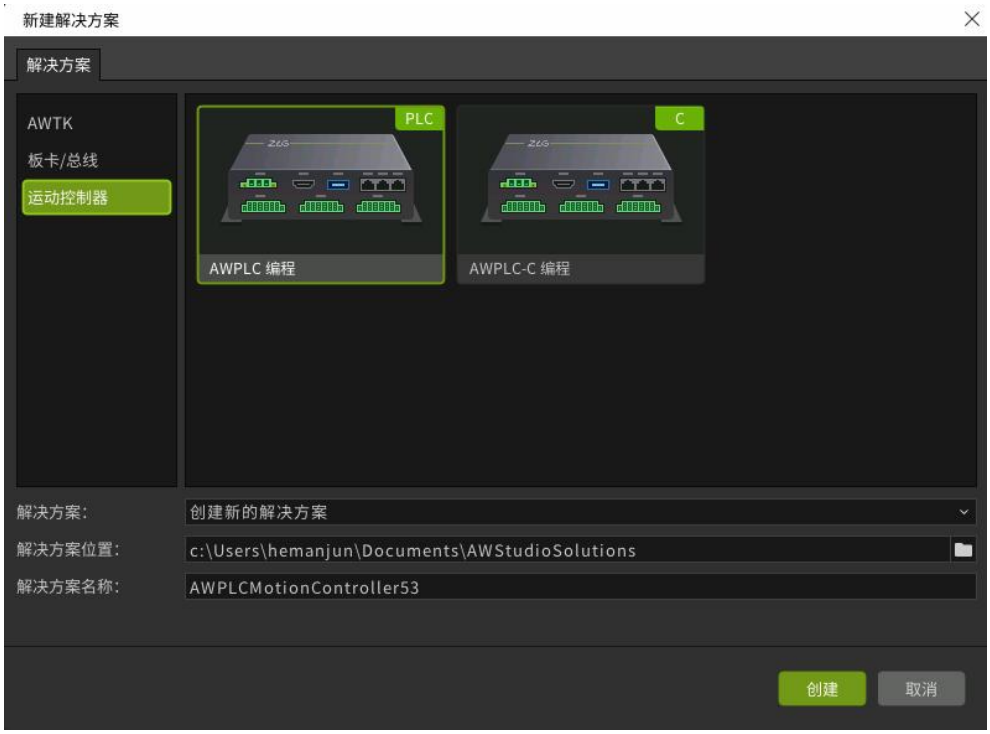

图 2.4　新建 AWPLC 解决方案

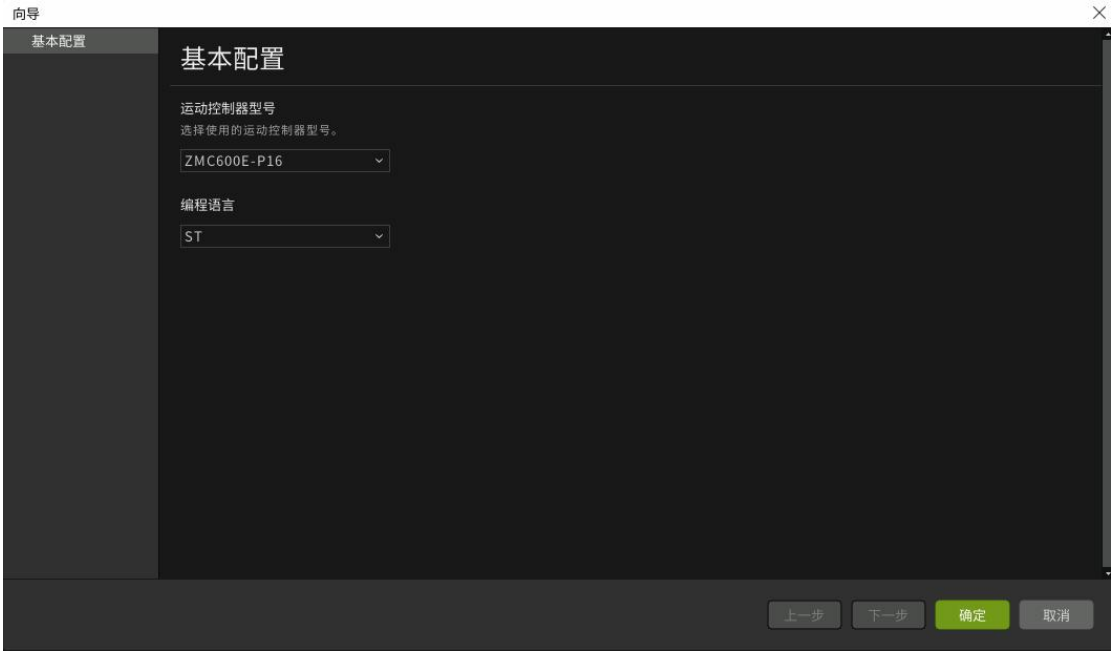

图 2.5　选择型号及语言

创建完成后，将看到 AWPLC 编程的主界面。在解决方案视图分为两大部分：Logic（逻辑组态）和 Device（设备组态）。

注:Device（设备组态）是一个整体的配置对象，修改后需要点击工具栏中的“保存全部”按钮进行保存。

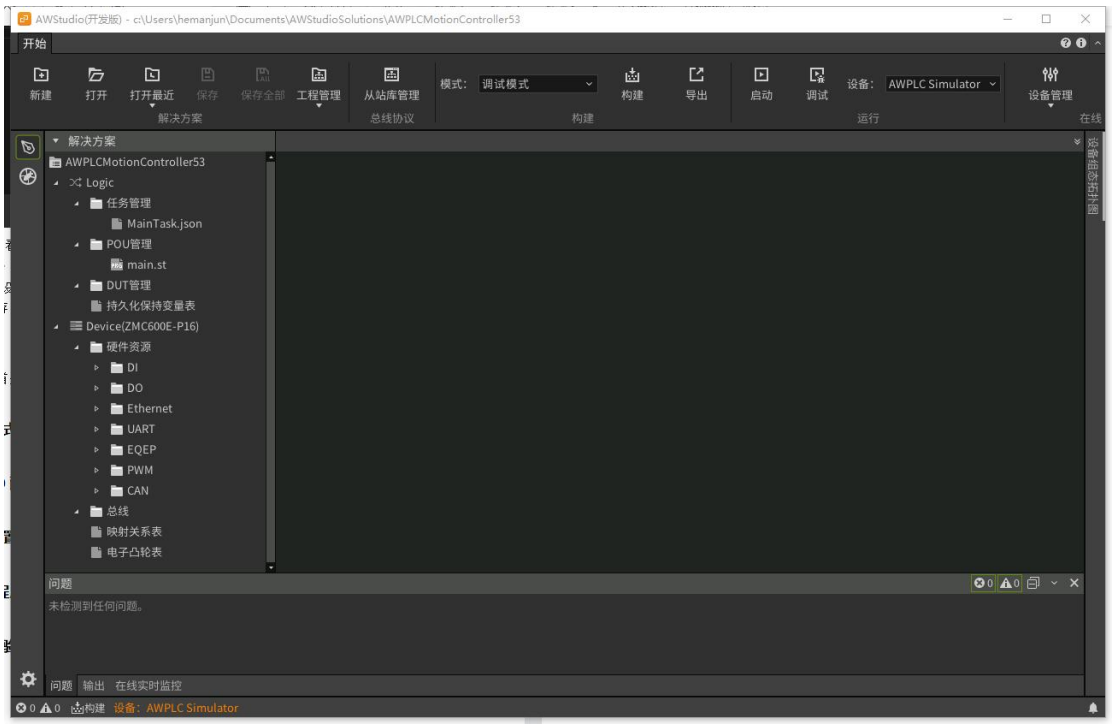

图 2.6　AWStudio AWPLC 解决方案主界面

2.3.2 添加总线

创建工程后，默认会有一个【总线协议】的节点，该节点用来给用户添加和配置总线的，新建的工程默认没有总线的，右键【总线协议】的节点，弹出右键菜单，点击【添加总线】按钮，弹出总线添加对话框。

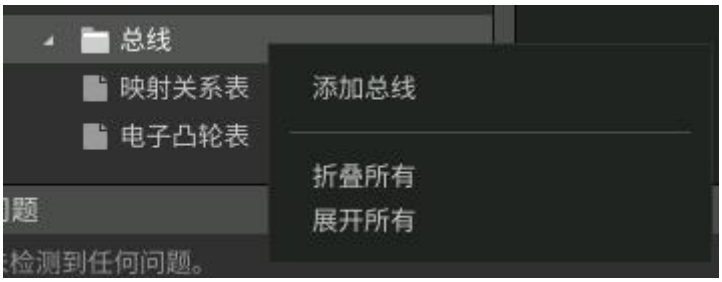

图 2.7　添加总线

选择 EtherCAT 主站添加。

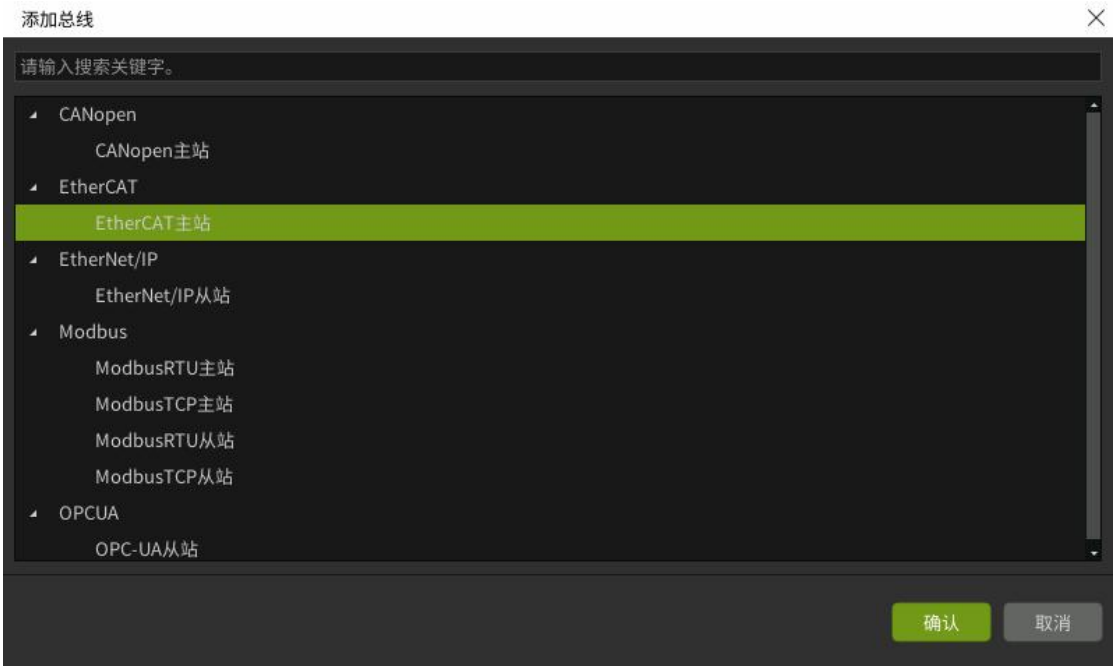

图 2.8　选择 EtherCAT 主站

EtherCAT 主站配置页面与 AWStudio 运动控制版完全一致。需要手动选择网卡，绑定到 ECAT 网口。

图 2.9　EtherCAT 主站配置界面

2.3.3 连接 PLC 控制器

在接下来无论是在线配置，还是下载启动程序，都要先连接 PLC 控制器。开发主机通过网线连接到 PLC 控制器的 NET2。在上方工具栏“运行-设备”下拉栏中打开，不断刷新以太网设备，直到扫描到 PLC 控制器设备。

图 2.10　连接 AWPLC 设备

点击设备并输入密码登陆设备。默认 PLC 连接密码为 admin。

图 2.11　密码登录，默认 admin

2.4 在线配置模式扫描 EtherCAT 从站设备

添加从站设备，有手动添加与在线扫描两种。AWPLC 支持在线配置模式，可以像其他主站控制器一样在线扫描从站设备。

点击上方工具栏“在线-在线配置”按钮进入在线配置模式。

连接成功后，右键主站节点，点击“扫描拓扑”功能就可以自动扫描当前网络中的从站设备。

图 2.12 拓扑扫描

扫描成功后，弹窗点击确定保存扫描结果。最终得到的拓扑结果如图 2.13。

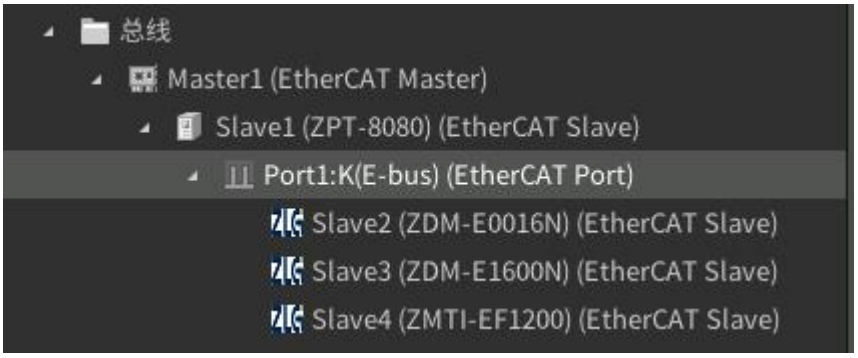

图 2.13 EtherCAT 网络拓扑

2.5 按钮及 LED 配置

按钮和 LED 灯的接法由第 2.2 节设备连接可知，按钮输入接到了数字输入模块，LED 灯接到了数字输出模块。只要控制这两个模块对应的端口即可控制按钮和 LED 灯。

打开“映射关系表-EtherCAT 主站”，会发现 AWStudio 自动将所有 EtherCAT 的 PDO 过程变量都映射到了 AWPLC 地址上。用户只要通过操作 AWPLC 对应地址就可以读写 PDO 变量。

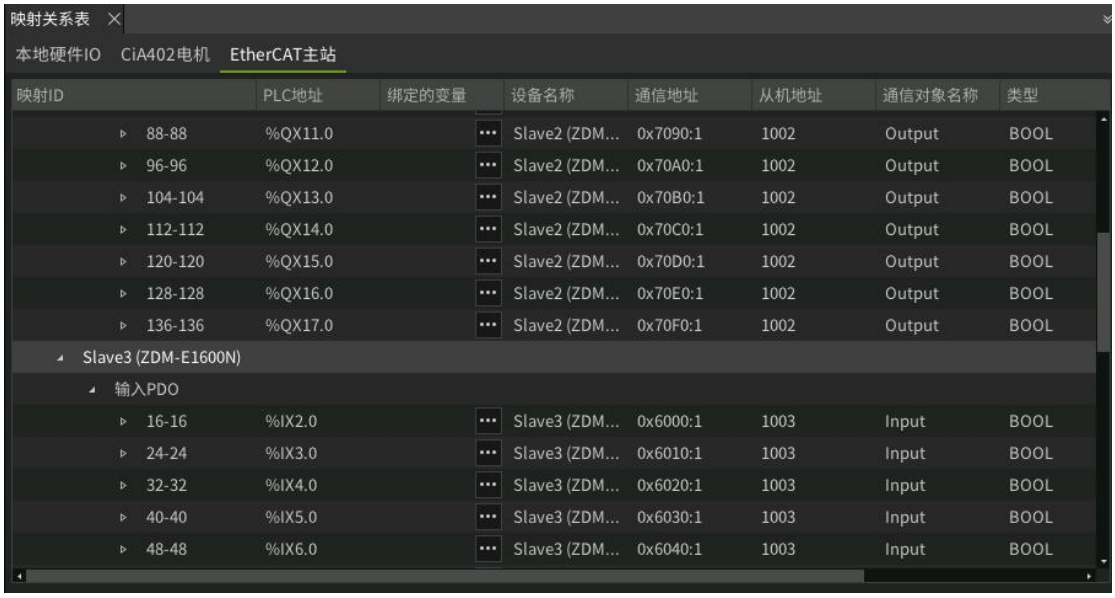

图 2.14 AWPLC 变量映射关系表

PDO 与 AWPLC 的映射关系自动实现，用户无需额外操作，只要稍后编程时变量绑定对应 PLC 地址即可。

2.6 电机参数配置

对于符合 CiA402 标准的 EtherCAT 电机控制从站，AWStudio 会自动识别并映射到 AWPLC 中。打开“映射关系表-CiA402 电机”，可以看到已经自动将 ZMTI-EF1200 识别添加为 Axis0。

图 2.15 AWPLC CiA402 电机参数表

双击该节点，可以进一步配置电机参数。目前只需要修改电机每转脉冲量即可。打开“缩放/映射”，修改脉冲量为 12800，点击确定保存。

图 2.16　AWPLC 电机参数配置

脉冲当量如何计算：

ZMTI-EF1200 为步进电机驱动从站。步进电机是用脉冲节拍控制，最常见的步进电机步距角为 1.8°，也就是每转 1 圈需要 200 个脉冲。

查阅手册，在 ZMTI-EF1200 的对象字典中，可以配置驱动器的细分精度，也就是把步进电机单步的步距角 1.8°进一步细分。在对象字典的 0x2008 和 0x2009 两项中。

0x2008	0	整步角度	R	FLOAT	---	1.8	单位度
0x2009	0	细分数	R	FLOAT	1~256	64	2 的 n 次方

默认值下，会将 1.8°细分为 64 个脉冲，也就是需要输出 64 个脉冲步进电机才会转动 1.8°。这时对于 ZMTI-EF1200 电机驱动器而言，步进电机每转 1 圈就需要

64×200=12800

因此可以直接在 AWStudio 配置脉冲量为 12800。

2.7　编写 PLC 程序

接下来编写 PLC 程序完成控制逻辑。主要完成三个任务：

1. 识别按钮输入并进入对应状态
2. 控制 LED 灯正确闪烁
3. 驱动电机运动

2.7.1 配置任务类型

新建工程后默认会自动创建一个 MainTask.json 任务，和一个主程序 main.st。

为了提高 PLC 程序响应速度，可以把任务配置到和 EtherCAT 同一个线程中。双击打开【MainTask.json】任务，然后在任务分页中修改【类型】为【注册事件】，注册事件类型选择 EtherCAT 总线，如下图：

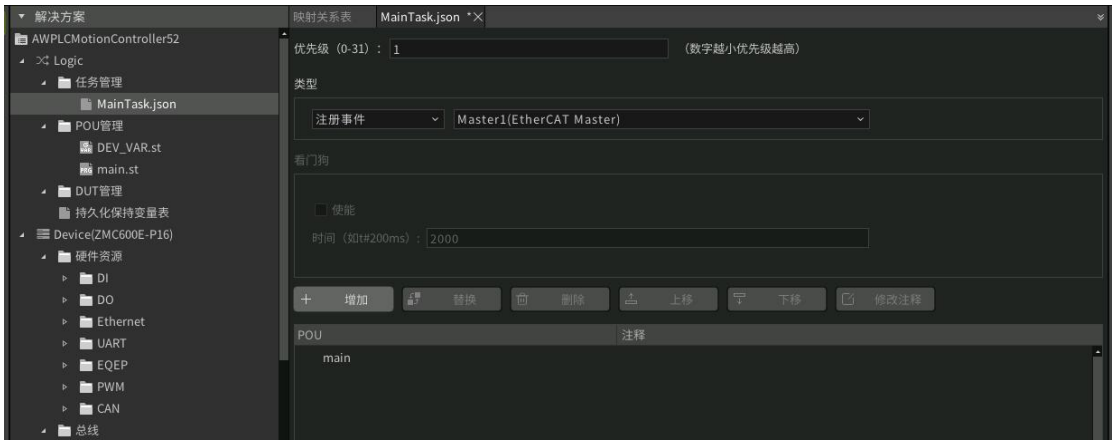

图 2.17　PLC 任务管理 task.json

2.7.2 添加 PLC 变量

双击打开 main.st，我们需要添加以下变量：

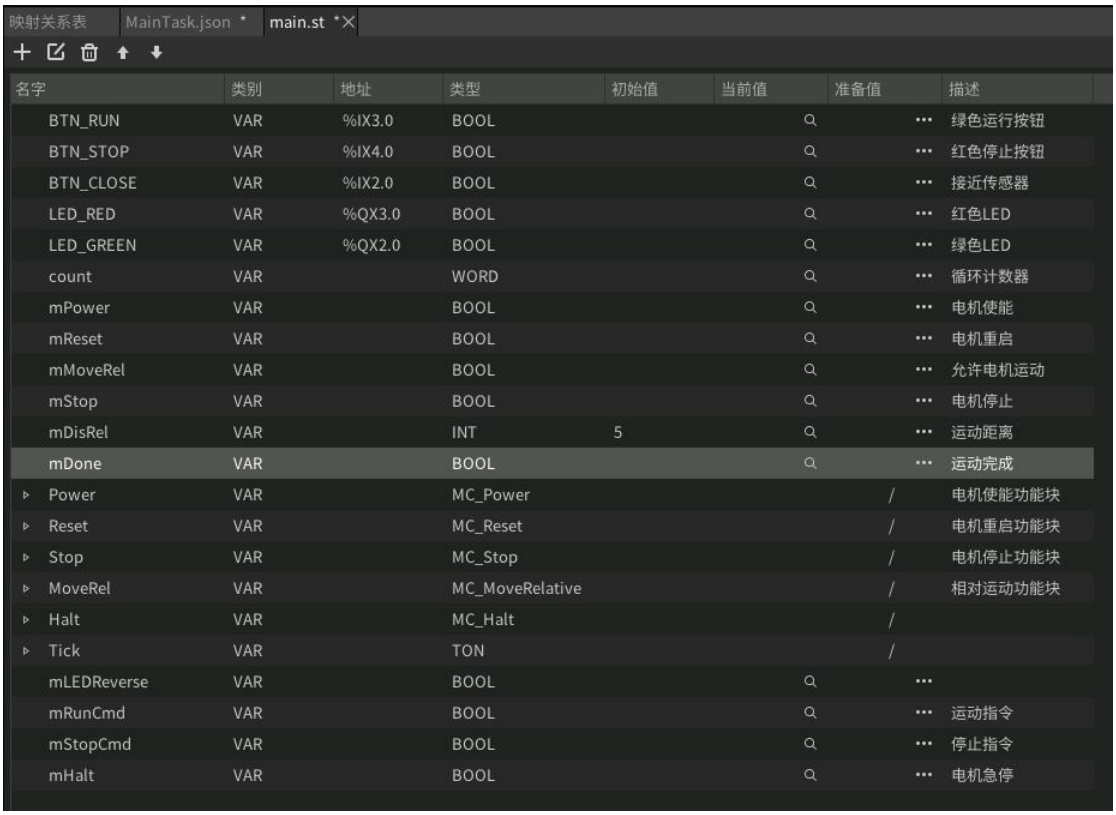

图 2.18　AWPLC ST 程序变量表

添加变量可以点击左上角的“+”按钮进行添加。输入变量名称，选择变量类型，设置初始值等。支持数据类型和实例化功能块等。填入 PLC 地址可以将变量绑定到指定对应地址，注意数据类型必须相符。

绑定 PLC 地址时可以有简便操作。在映射关系表中选中 PLC 地址，Ctrl+C 即可复制地址。

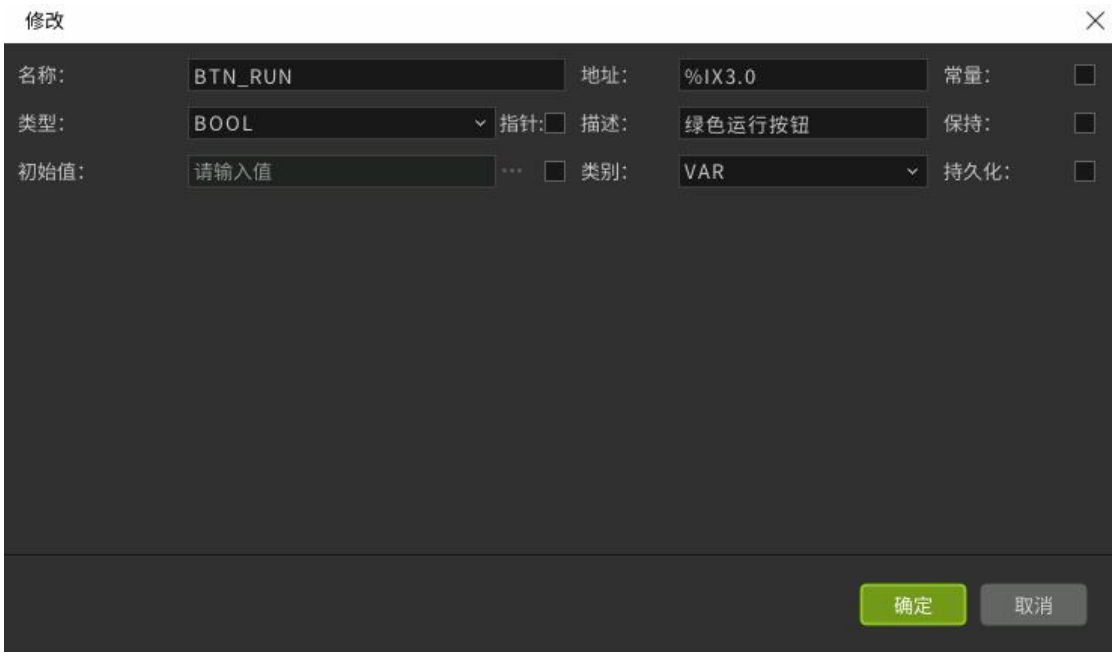

图 2.19　ST 变量绑定指定 PLC 地址

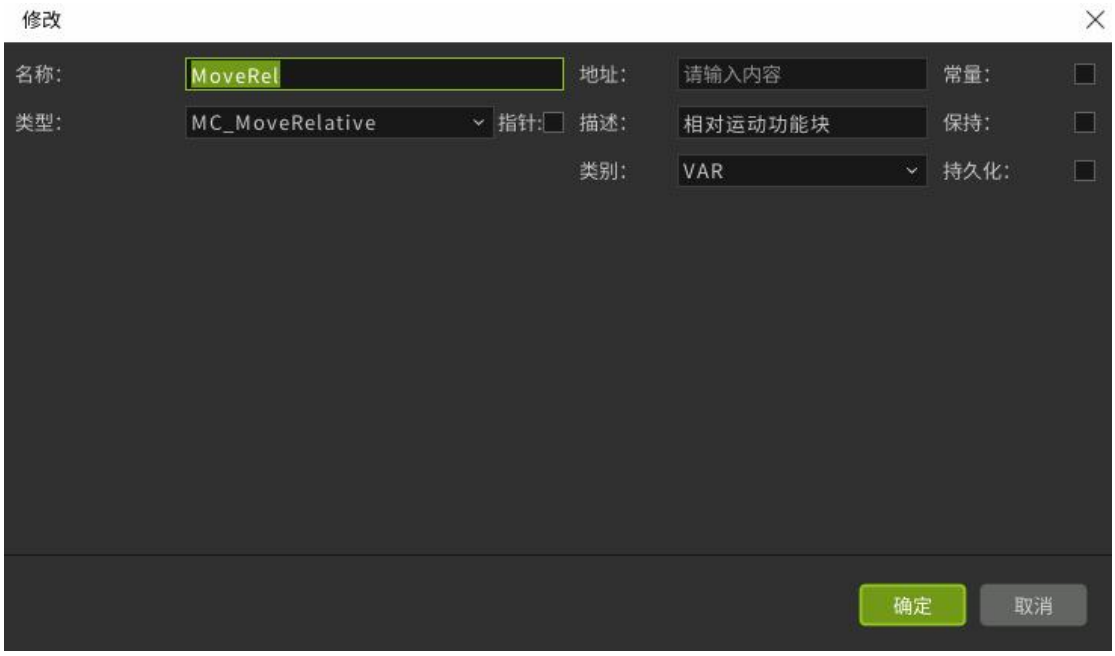

图 2.20　普通 ST 变量

也可以用文本编辑器打开 main.st 文件，用 ST 语言去创建变量和编写程序。

2.7.3 PLC 控制按钮和 LED

该应用中包括运行键（绿）、停止键（红）、运行状态灯（绿）、紧急状态灯（红）、接近传感器等外设，这些外设功能如下。

1. 运行状态灯用于指示电机运行，停止状态下为绿灯闪烁，运行状态下为绿灯常亮，紧急状态下绿灯熄灭。

2. 紧急状态灯用于指示系统进入紧急状态。正常状态下红灯熄灭。紧急状态下红灯常亮，绿灯熄灭。

3. 运行键用于启动电机，进入运行状态。停止键用于停止电机运行，进入停止状态。

4．接近传感器检测到金属物体接近时，会输出低电平。此时系统需要进入紧急状态。进入紧急状态后，必须用运行键重启电机系统。

以上控制逻辑实现在 PLC 程序中。

其中 LED 绿灯闪烁时，采用了定时器功能块 TON 定时 500ms 翻转 LED 电平，实现 LED 以 2Hz 的频率闪烁。

```
// 运行状态
IF BTN_RUN = 0 THEN // 输入低有效
  mRunCmd := TRUE;
  mStopCmd := FALSE;
  mHalt := FALSE;
END_IF;

// 停止状态
IF BTN_STOP = 0 THEN // 输入低有效
  mStopCmd := TRUE;
  mRunCmd := FALSE;
END_IF

// 急停状态
IF BTN_CLOSE = 0 THEN // 输入低有效
  mHalt := TRUE;
  mRunCmd := FALSE;
END_IF;

IF mRunCmd THEN
  LED_GREEN := 1; // 绿灯常亮
ELSE
  // 绿灯 2Hz 闪烁
  IF mLEDReverse THEN
    LED_GREEN := !LED_GREEN;
    Tick(IN := FALSE, PT := T#500ms, Q => mLEDReverse); // TON 功能块 IN 上升沿触发，因此每次定时器完成后要重新输入上升沿
```

```
ELSE
    Tick(IN := TRUE, PT := T#500ms, Q => mLEDReverse);
  END_IF;
END_IF;

// 急停状态
IF mHalt THEN
  LED_GREEN := FALSE; // 绿灯熄灭
  LED_RED := TRUE; // 红灯常亮
ELSE
  LED_RED := FALSE; //红灯熄灭
END_IF;
```

2.7.4 PLC 控制电机

运动控制轴状态机如下图：

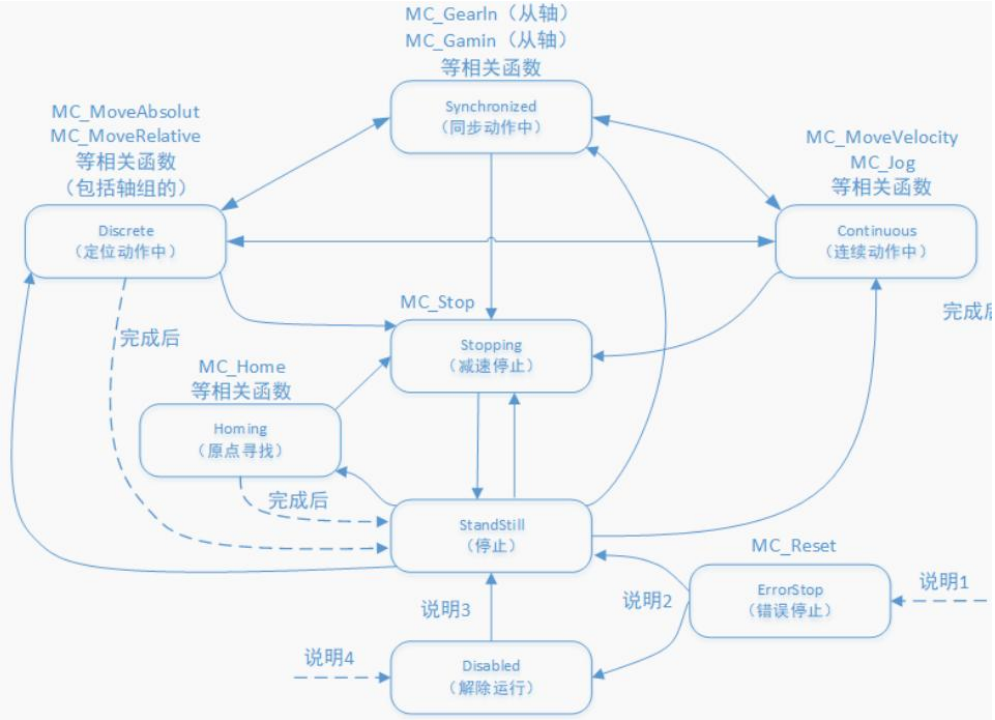

图 2.21　AWPLC 运动控制轴状态机

说明：

1. 在轴运行的过程中出现异常或者状态字的错误位为 1 的时候，轴就会进入 ErrorStop 状态（错误停止）。无法轴是否在使能状态，只要状态字的错误位为 1 的时候就会进入 ErrorStop 状态（错误停止），同时如果在轴运行的过程中，通过 MC_Power 函数解除使能状态，轴也会进入 ErrorStop 状态（错误停止）。

2. 进入 ErrorStop 状态（错误停止）后调用 MC_Reset 函数后，如果 MC_Power 的

bRegulatorOn 为 TRUE 的话，就进入 StandStill 状态（停止），如果是 MC_Power 的 bRegulatorOn 为 FALSE 的话，就进入 Disabled 状态（解除运行）。

3. MC_Power 的 bRegulatorOn 设置为 TRUE，并且 MC_Power 的 Status 返回 TRUE 的话，就会进去 StandStill 状态（停止）。

4. 任何视情况下，MC_Power 的 bRegulatorOn 设置为 FALSE，并且轴没有任何错误（没有进去 ErrorStop 状态（错误停止）），就会进去 Disabled 状态（解除运行）。

5. MC_Stop 函数调用后，当 MC_Stop 的 Done 为 TRUE，Execute 为 FALSE 时，轴状态就会从 Stopping 状态（减速停止）进入 StandStill 状态（停止）。

6. 上图“完成后”的虚线，是该对应状态下的指令执行完后，就会进入 StandStill 状态（停止）。

注: 对应的轴状态可以通过 MC_ReadStatus 函数查看。

这里我们在 PLC 程序中实现一个速度为 1 圈/s，单次旋转 5 圈的往复运动。

```
count := count + 1;

// 电机初始化时序
IF count < 10 THEN
  mPower := FALSE;
  mReset := TRUE;
  mStop := FALSE;
  mMoveRel := FALSE;
ELSIF count < 20 THEN
  mPower := TRUE;
  mReset := FALSE;
  mMoveRel := TRUE;
END_IF;

// 电机到达目标、或进入停止状态后，重走电机初始化步骤，使电机持续运行
IF mDone OR (mStop OR mStopCmd) OR mHalt THEN
  count := 0;
END_IF;

IF mDone AND mMoveRel THEN
  IF mDisRel > 0 THEN
    mDisRel := -5;
  ELSE
    mDisRel := 5;
  END_IF;
END_IF
```

```
// 如果电机无法正常使能或者旋转，重启程序后让 mReset 先变成 FALSE，然后再变成 TRUE，让电机复
```

```
位一下，再尝试启动电机。
Reset(Axis := CiA402_AXIS_0, Execute := mReset AND !mStopCmd AND !mHalt);
// 让电机使能
Power(Axis := CiA402_AXIS_0, Enable := mPower AND !mStopCmd AND !mHalt, bRegulatorOn :=
TRUE, bDriveStart :=TRUE);
// 让电机移动到 Modbus 输入的位置中
MoveRel(Axis := CiA402_AXIS_0, Execute := mMoveRel AND mRunCmd, Distance := mDisRel,
Velocity := 1, Acceleration := 10, Deceleration := 10, Done => mDone);
// 停止正在运动的电机
Stop(Axis := CiA402_AXIS_0, Execute := mStop OR mStopCmd, Deceleration := 10);
// 电机急停
Halt(Axis := CiA402_AXIS_0, Execute := mHalt, Deceleration := 100);
```

2.8 构建及运行验证

可以先用“构建”功能尝试编译，也可以直接按“启动”按钮从编译到部署、启动一步完成。

启动后，可以在 AWStudio 开发上位机实时监控 PLC 变量的数据，可以查看下方从设备抓回来的日志。

注意：程序启动后，在上位机点击“停止”只会退出上位机的监控状态，不会停止设备上正在运行的 PLC 程序。

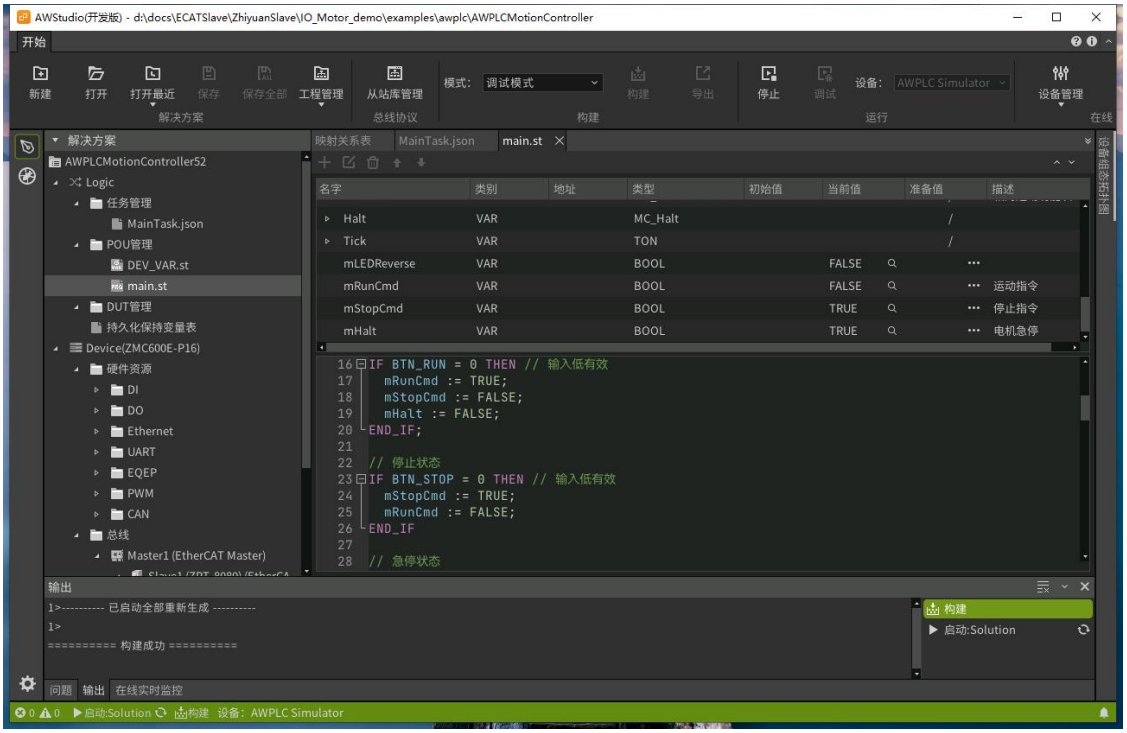

图 2.22 构建并启动 AWPLC 程序

2.9 完整 PLC 程序

```
PROGRAM main
```

```
VAR
  // 绿色运行按钮
  BTN_RUN AT %IX3.0 : BOOL;
  // 红色停止按钮
  BTN_STOP AT %IX4.0 : BOOL;
  // 接近传感器
  BTN_CLOSE AT %IX2.0 : BOOL;
  // 红色 LED
  LED_RED AT %QX3.0 : BOOL;
  // 绿色 LED
  LED_GREEN AT %QX2.0 : BOOL;
  // 循环计数器
  count : WORD;
  // 电机使能
  mPower : BOOL;
  // 电机重启
  mReset : BOOL;
  // 允许电机运动
  mMoveRel : BOOL;
  // 电机停止
  mStop : BOOL;
  // 运动距离
  mDisRel : INT := 5;
  // 运动完成
  mDone : BOOL;
  // 电机使能功能块
  Power : MC_Power;
  // 电机重启功能块
  Reset : MC_Reset;
  // 电机停止功能块
  Stop : MC_Stop;
  // 相对运动功能块
  MoveRel : MC_MoveRelative;
  Halt : MC_Halt;
  Tick : TON;
  mLEDReverse : BOOL;
  // 运动指令
  mRunCmd : BOOL;
  // 停止指令
  mStopCmd : BOOL;
  // 电机急停
  mHalt : BOOL;
END_VAR
```

```
count := count + 1;

// 电机初始化时序
IF count < 10 THEN
  mPower := FALSE;
  mReset := TRUE;
  mStop := FALSE;
  mMoveRel := FALSE;
ELSIF count < 20 THEN
  mPower := TRUE;
  mReset := FALSE;
  mMoveRel := TRUE;
END_IF;

// 运行状态
IF BTN_RUN = 0 THEN // 输入低有效
  mRunCmd := TRUE;
  mStopCmd := FALSE;
  mHalt := FALSE;
END_IF;

// 停止状态
IF BTN_STOP = 0 THEN // 输入低有效
  mStopCmd := TRUE;
  mRunCmd := FALSE;
END_IF

// 急停状态
IF BTN_CLOSE = 0 THEN // 输入低有效
  mHalt := TRUE;
  mRunCmd := FALSE;
END_IF;
IF mRunCmd THEN
  LED_GREEN := 1; // 绿灯常亮
ELSE
  // 绿灯 2Hz 闪烁
  IF mLEDReverse THEN
    LED_GREEN := !LED_GREEN;
    Tick(IN := FALSE, PT := T#500ms, Q => mLEDReverse); // TON 功能块 IN 上升沿触发，因此每次定时器完成后要重新输入上升沿
```

```
  ELSE
    Tick(IN := TRUE, PT := T#500ms, Q => mLEDReverse);
```

```
  END_IF;
END_IF;

// 急停状态
IF mHalt THEN
  LED_GREEN := FALSE; // 绿灯熄灭
  LED_RED := TRUE; // 红灯常亮
ELSE
  LED_RED := FALSE; //红灯熄灭
END_IF;

// 电机到达目标、或进入停止状态后，重走电机初始化步骤，使电机持续运行
IF mDone OR (mStop OR mStopCmd) OR mHalt THEN
  count := 0;
END_IF;

IF mDone AND mMoveRel THEN
  IF mDisRel > 0 THEN
    mDisRel := -5;
  ELSE
    mDisRel := 5;
  END_IF;
END_IF

// 如果电机无法正常使能或者旋转，重启程序后让 mReset 先变成 FALSE，然后再变成 TRUE，让电机复位一下，再尝试启动电机。
Reset(Axis := CiA402_AXIS_0, Execute := mReset AND !mStopCmd AND !mHalt);
// 让电机使能
Power(Axis := CiA402_AXIS_0, Enable := mPower AND !mStopCmd AND !mHalt, bRegulatorOn :=
TRUE, bDriveStart :=TRUE);
// 让电机移动到 Modbus 输入的位置中
MoveRel(Axis := CiA402_AXIS_0, Execute := mMoveRel AND mRunCmd, Distance := mDisRel,
Velocity := 1, Acceleration := 10, Deceleration := 10, Done => mDone);
// 停止正在运动的电机
Stop(Axis := CiA402_AXIS_0, Execute := mStop OR mStopCmd, Deceleration := 10);
// 电机急停
Halt(Axis := CiA402_AXIS_0, Execute := mHalt, Deceleration := 100);

END_PROGRAM
```

3. 免责声明

本着为用户提供更好服务的原则，广州致远电子股份有限公司（下称“致远电子”）在本手册中将尽可能地为用户呈现详实、准确的产品信息。但介于本手册的内容具有一定的时效性，致远电子不能完全保证该文档在任何时段的时效性与适用性。致远电子有权在没有通知的情况下对本手册上的内容进行更新，恕不另行通知。为了得到最新版本的信息，请尊敬的用户定时访问致远电子官方网站或者与致远电子工作人员联系。感谢您的包容与支持！

诚信共赢，持续学习，客户为先，专业专注，只做第一

广州致远电子股份有限公司

更多详情请访问 **www.zlg.cn**

欢迎拨打全国服务热线 **400-888-4005**