EtherCAT 总线型运动控制系统的特点与应用（下篇）

深圳市雷赛控制技术有限公司

左力

2019 年 8 月

目　录

一．EtherCAT 总线型运动控制器 BAC332E.....1
二．SMC BASIC Studio 与总线的配置、测试.....4
三．雷赛 BASIC 与 BAC332E 编程.....16
四．#C 与 BAC332E 编程.....20
一）基本功能的实现.....20
二）执行 G 代码.....21
三）样条插值函数（PVT 函数）.....22
五．参考文献.....24
六．附录　例程代码.....25

一. EtherCAT 总线型运动控制器 BAC332E

2019 年 4 月，雷赛公司推出的 BAC332E 运动控制器最多可以控制 32 个 EtherCAT 总线型电机及扩展功能模块。可实现多轴点位运动、直线插补运动、空间圆弧插补、样条曲线插补（PVT 函数）、编码器位置检测、IO 控制、位置比较、位置锁存等功能。

和雷赛的 EtherCAT 总线型运动控制卡 DMC E3032 相比，BAC332E 运动控制器具有以下特点：

1. BAC332E 运动控制器可使用 BASIC 语言编程，独立运行（不需要 PC 机）；也可以让 BAC332E 运动控制器作下位机、PC 机作上位机并采用高级语言（VB、VC、C#、Delphi、LabVIEW 等）编程、通过 Ethernet 网线控制 BAC332E 运动控制器。
2. 可在程序中可载入 G 代码指令。该功能可用于：金属切割、金属焊接、激光切割、激光雕刻、布料裁剪等加工过程。
3. 具有多轴的样条插值功能（PVT 函数），可实现复杂的轨迹控制。

BAC332E运动控制器的运动控制函数库功能丰富、易学易用，用户开发应用软件十分方便；并提供 BASIC、C#、VB、VC、Delphi、Labview 编程语言的例程。随产品免费提供的雷赛运动控制器集成开发软件 SMC BASIC Studio，不但可以编辑、调试 BASIC 程序，还可以设置 EtherCAT 总线参数、总线节点的参数，并且用于控制器及运动控制系统的硬件测试。

BAC332E 运动控制器的外形如图 1.1 所示，其主要参数如表 1.1 所示。

BAC332E 运动控制器的 IO 接口、编码器接口及驱动软件的安装方法都和普通脉冲型运动控制器相同。详见产品手册。



图 1.1.　BAC332E 运动控制器外观

表 1.1. BAC332E 运动控制器的主要参数

项目		BAC332E
硬件部分	CPU	ARM A9 双核
	操作系统	Linux
	RAM 空间	128MBytes
	FLASH 空间	128MBytes
	掉电存储空间	64KBytes
	RTC 实时时钟	年月日时分秒
	G 代码文件	G 代码文件单一文件最大 8M，20 万行，支持 1000 个文件
通信	Ethernet	3 路；2 路百兆以太网（Eth0、Eth1），1 路千兆以太网(Eth2)
	EtherCAT	1 路；100Mbps，总线周期 500us32 轴，250us16 轴
	RS232	1 路；通讯参数可配、Modbus 协议/自由协议
	RS485	1 路；通讯参数可配、Modbus RTU 协议/自由协议
	U 盘	1 路；支持文件的上传、下载以及拷贝
轨迹控制	控制轴数	总线轴或者定位模块轴，最多 32 轴
	速度曲线规划	T 型/S 型、对称/非对称
	定长运动	相对运动、绝对运动、在线变速、在线变位
	恒速运动	恒速运动、在线变速、支持在线变方向
	回原点运动	EtherCAT 定义的总线回零模式
	直线插补	2~16 轴
	圆弧插补	2 轴平面圆弧、3 轴空间圆弧
	螺旋插补	3 轴
	插补坐标系	8 个
辅编	辅助编码器	1 个，计数长度 32 位有符号，2MHz
IO	隔离输入	本地 12 路，可接总线扩展，最多 1024 路
		最高响应频率 4KHz，导通电流>=4.2mA(15V)典型值 6.9mA
	隔离输出	本地 12 路，可接总线扩展，最多 1024 路
		500mA,(5 - 24v,吸入)
电源	主电源供电	外部 DC24V，启动电流 1A,当电压低于 15.6V 时,控制器不能正常运行,将启动低压报警
	IO 部分	外部 DC24V
工作环境	工作温度	0℃～50℃
	贮存温度	-20℃～80℃
	湿度	5～85%，非结露

与雷赛脉冲型运动控制器不同，BAC332E 运动控制器有一个小显示屏，如图 1.2 所示，可显示控制器状态、输入和输出口状态、控制器版本号、控制器通讯参数等信息。



图 1.2.　BAC332E 显示屏显示的信息

BAC332E 运动控制器上有一个 6 位拨码开关，其中 SW1 和 SW2 用来设置输出端口的初始电平。SW1 设置 OUT0 ~ OUT5 端口、SW2 设置 OUT6 ~ OUT11 端口，拨码开关设置为 ON，初始电平为低电平；设为 OFF，初始电平为高电平。SW3 ~ SW6 功能保留。

二．SMC BASIC Studio 与总线的配置、测试

如图 2.1 所示，本文介绍的 EtherCAT 总线运动控制系统实验装置的结构和“EtherCAT 总线型运动控制系统的特点与应用（上篇）”所介绍的一样。其中各产品的特点、接口电路、参数设置等内容就不再此重复。

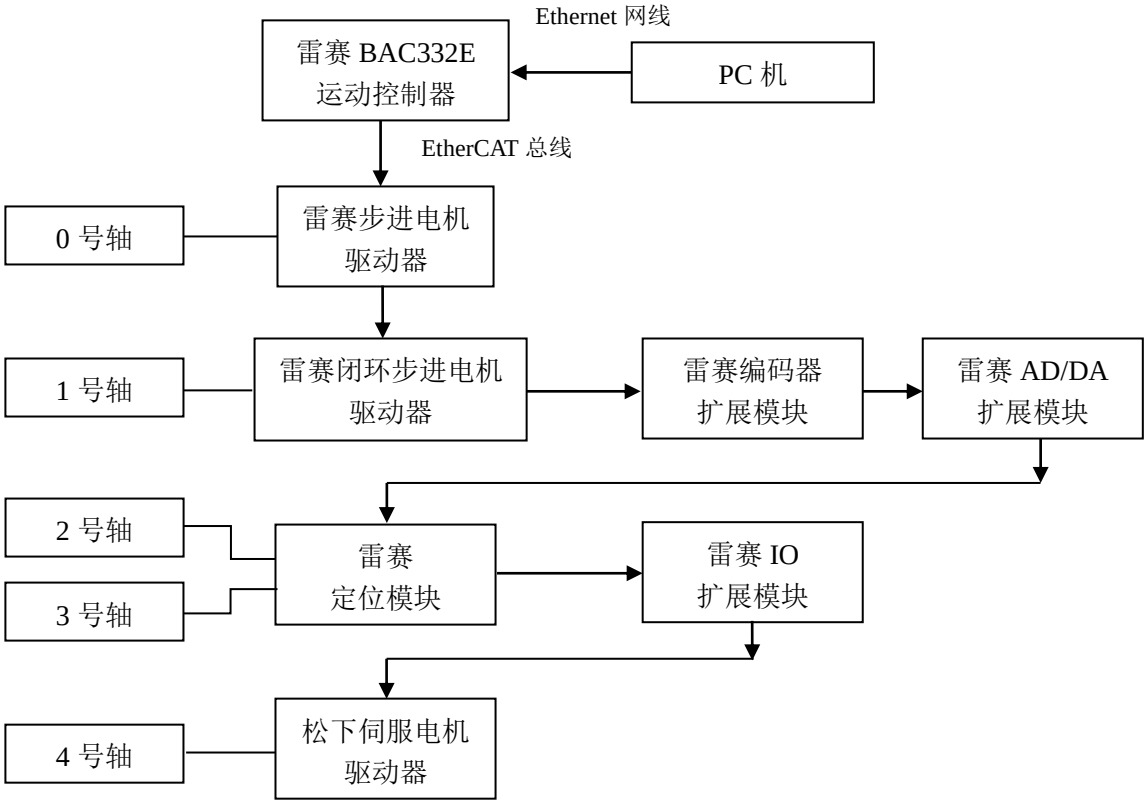


图 2.1.　EtherCAT 总线运动控制系统实验装置的结构

1. PC 机与 BAC332E 运动控制器的连接

首先，用网线将 PC 机网口与 BAC332E 运动控制器上的控制器的 Eth0 接口连接起来。

因为BAC332E的IP地址为192.168.5.11。因此,PC机的IP地址前三段也要设为192.168.5，第四段不能设为 11。

修改 PC 机 IP 地址的方法为：打开“网络和共享中心”，点击“本地连接”、“属性”，选择“Internet 协议版本 4”，然后修改 IP 地址，如图 2.2 所示。

2. 运行 SMC BASIC Studio 并连接控制器

打开 BAC332E 运动控制器的光盘中的文件夹“MOTION”，解压文件 SMCIDE_2019XXXX 后，运行其中的 Leadshine.SMC.IDE.exe 文件。然后，在主页面上，点击第 5 个图标，如图 2.3 所示，使 PC 机与 BAC332E 运动控制器连接。

3. 自动配置 EtherCAT 总线

在 SMC BASIC Studio 的菜单中打开“工程”、点击“新建”。输入文件名后，点击“保存”。

如图 2.4 所示。

在工程栏中，用鼠标右键点击“解决方案”，选择“添加设备”，点击“添加 EtherCAT 设备”，如图 2.5 所示。

在工程栏中，点击“设备”前的“+”号，然后用鼠标右键点击“EtherCAT_0”，选择“扫描设备”，如图 2.6 所示。扫描后的结果如图 2.7 所示。

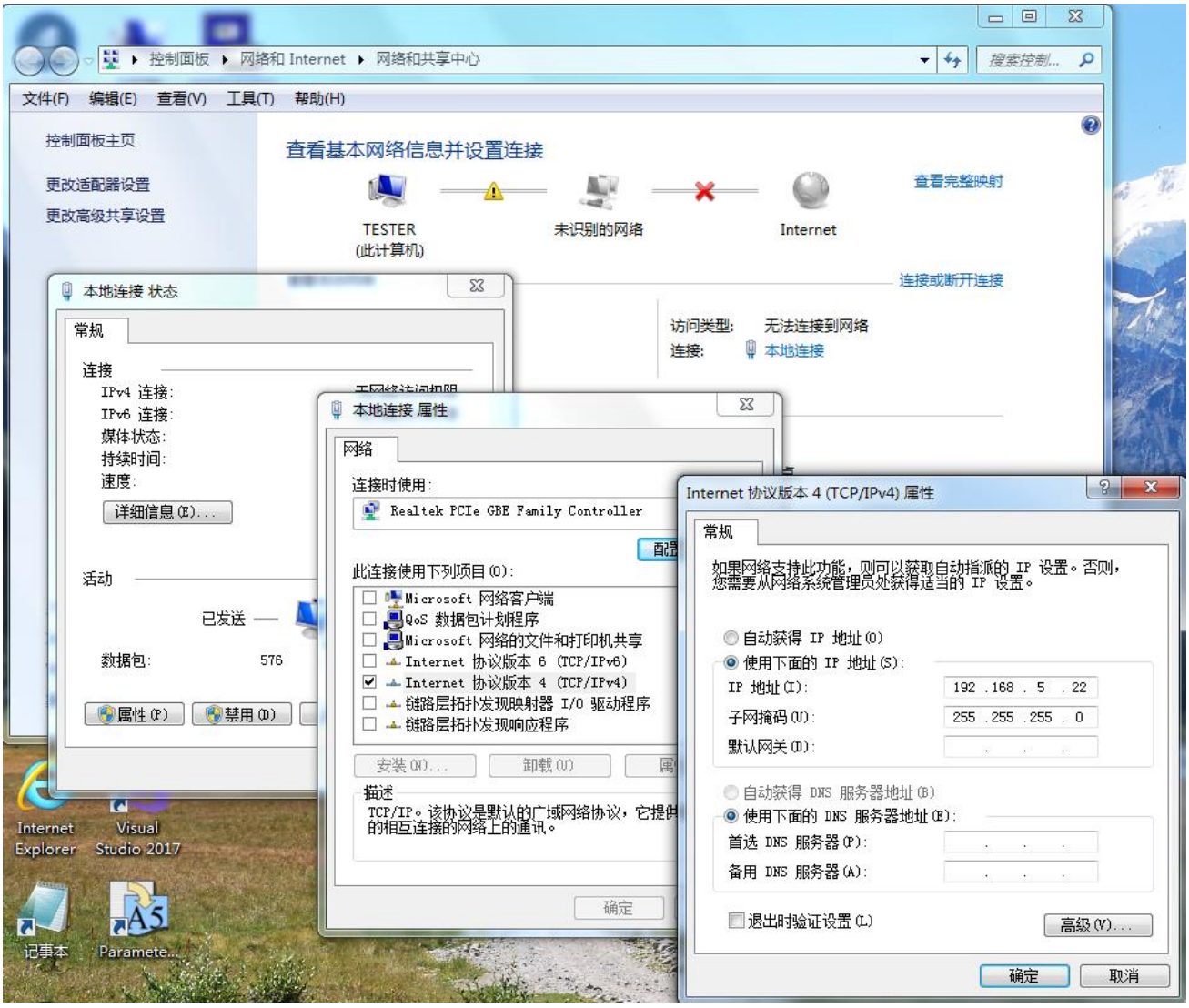


图 2.2.　PC 机修改 IP 地址的方法



图 2.3.　PC 机连接控制器

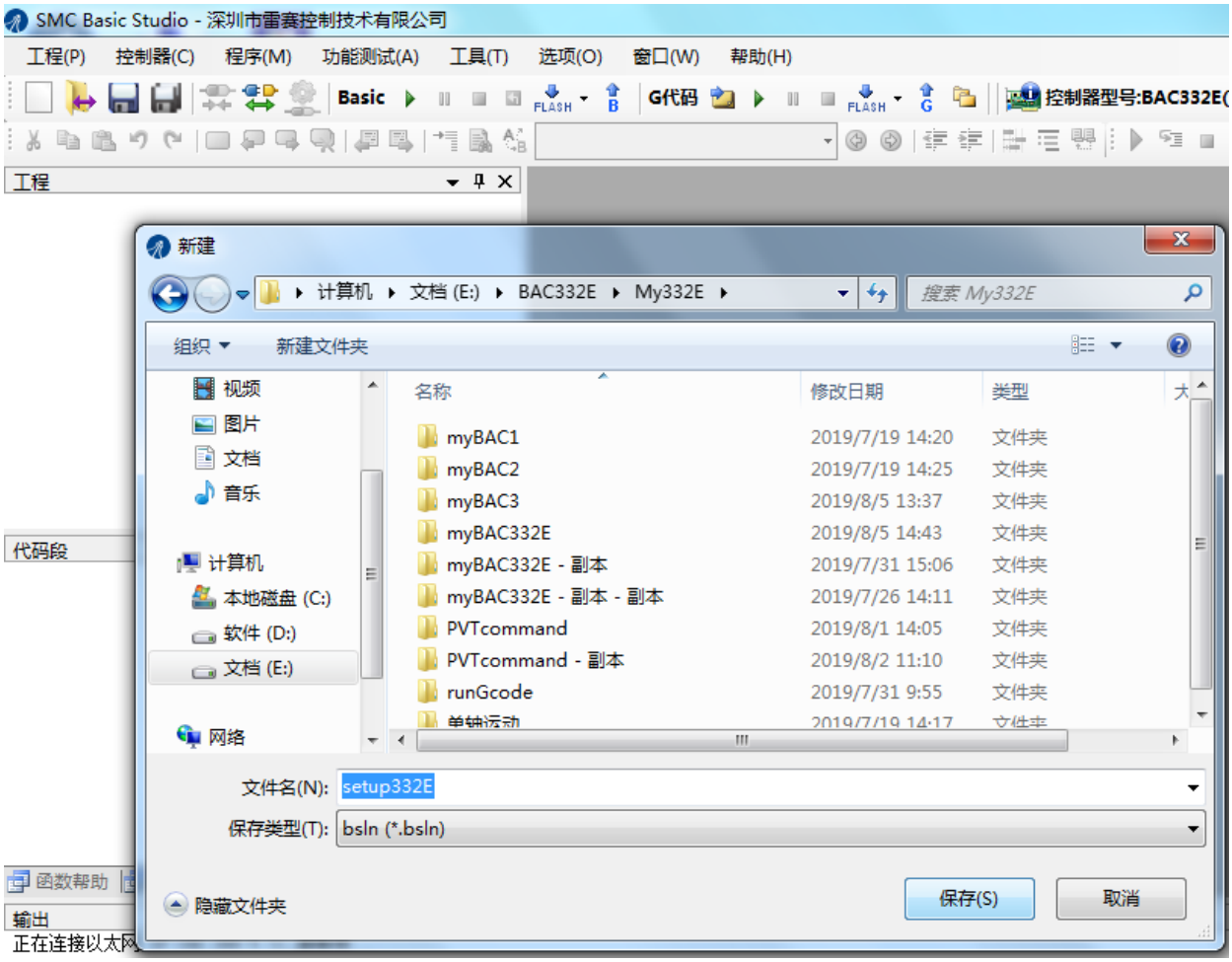


图 2.4.　新建一个项目

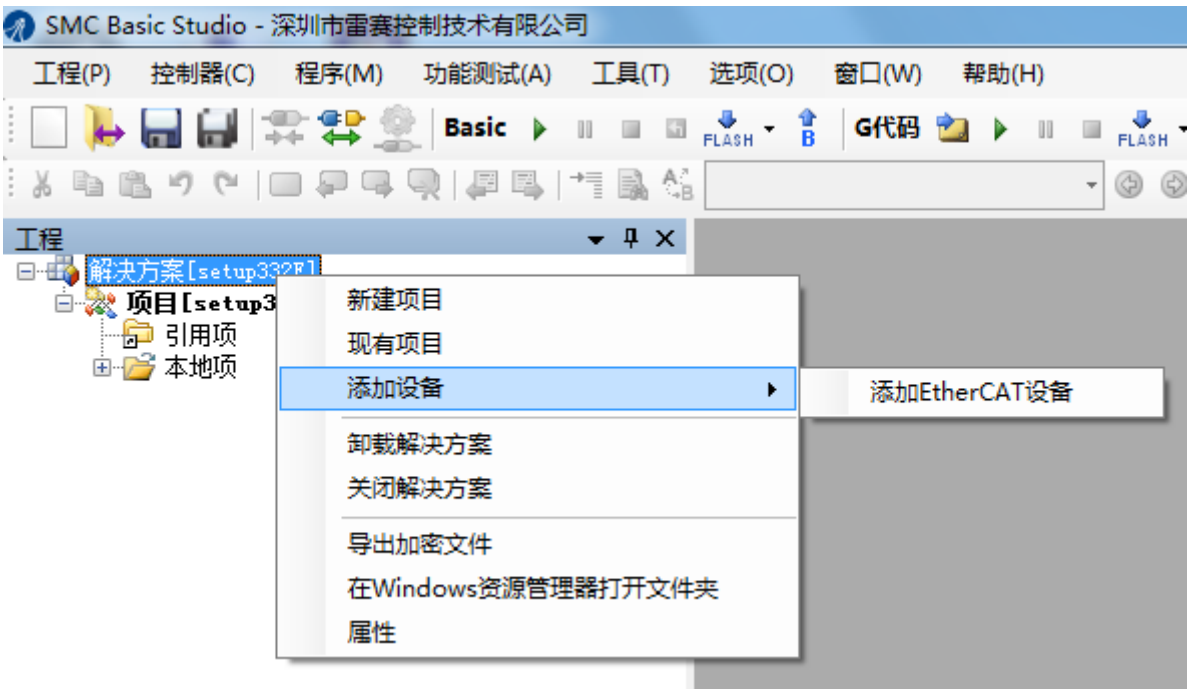


图 2.5.　添加 EtherCAT 设备

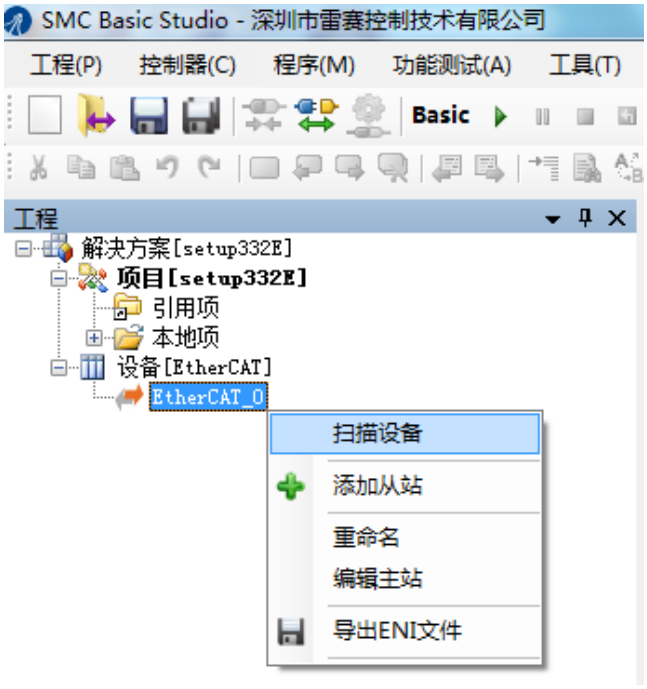


图 2.6.　扫描 EtherCAT 设备

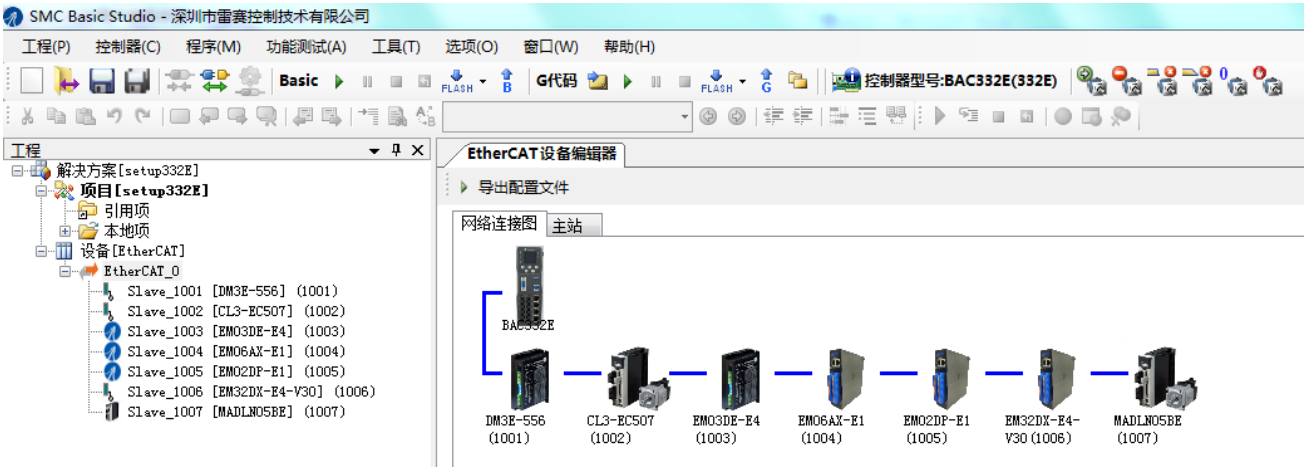


图 2.7.　扫描后的结果

这时，雷赛的步进电机驱动器 DM3E-556、AD/DA 扩展模块、IO 扩展模块自动完成了配置。其他元件还要继续手工配置。

4. 手工配置 EtherCAT 总线

1）添加轴映射

点击“EtherCAT 设备编辑器”界面中的“主站”、打开“主站”页面。在选项卡“轴映射关系”中，点击下面的“添加”按钮，然后在“映射类型”中选“EtherCAT”、在“从站类型”中选“驱动器”、“映射从站”中选“1002”，即雷赛闭环步进电机驱动器 CL3-EC507，如图 2.8 所示。最后，点击左下角的“上移”按钮，将 1002 节点上移至第 2 的位置上。即：将闭环步进电机设为轴 1。

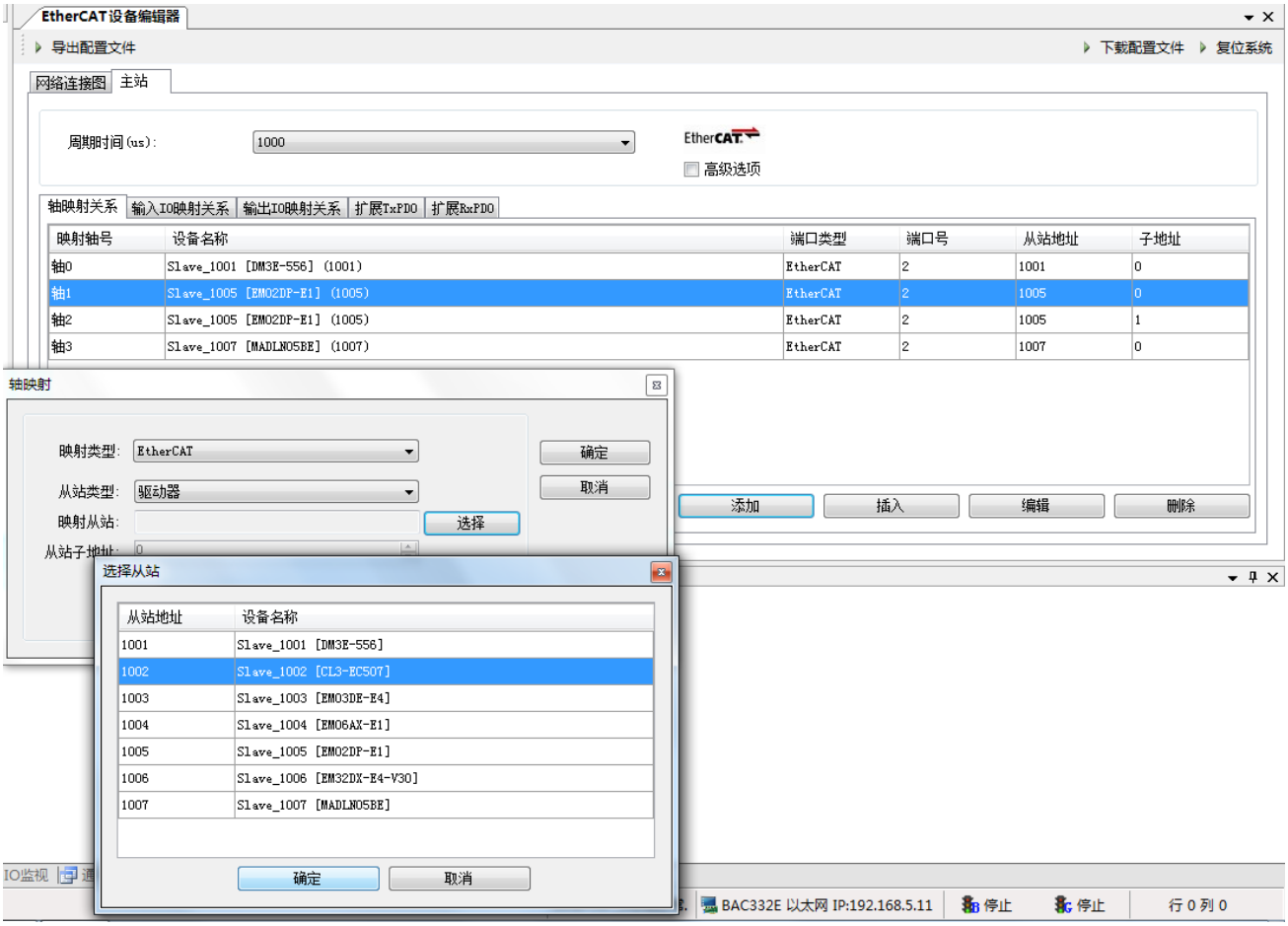


图 2.8.　手工添加电机

2）添加 IO 映射

点击选项卡“输出 IO 映射关系”，点击下面的“添加”按钮，然后在“映射类型”中选“EtherCAT”、在“从站类型”中选“数字 IO”、“映射从站”中选“1005”，即雷赛定位模块 EM02DP-E1，“映射变量”选“General Output IO”，点击“确定”，如图 2.9 所示。这样，定位模块上的 8 个数字输出接口添加在 EtherCAT 总线上。

最后，点击左下角的“上移”按钮，将定位模块上的 8 个数字输出接口上移至第 2 的位置上。即：该 8 个输出口的编号为 12~19。

3）添加 TxPDO 变量

点击选项卡“扩展 TxPDO 映射关系”，点击下面的“添加”按钮，然后在“映射类型”中选“EtherCAT”、在“映射从站”中选“1003”，即雷赛编码器扩展模块 EM03DE-E4，“映射变量”选“Encoder1_Val”，点击“确定”，如图 2.10 所示。这样，编码器扩展模块上的 1 号接口可正常使用。

4）添加从站的过程数据

在工程栏中双击“CL3-EC507”后，“EtherCAT 设备编辑器”变为图 2.11 所示界面。点击选项卡“过程数据”，选中“选择输出”栏中的“Receive PPDO 1”，然后，点击下方的“编辑”按钮。

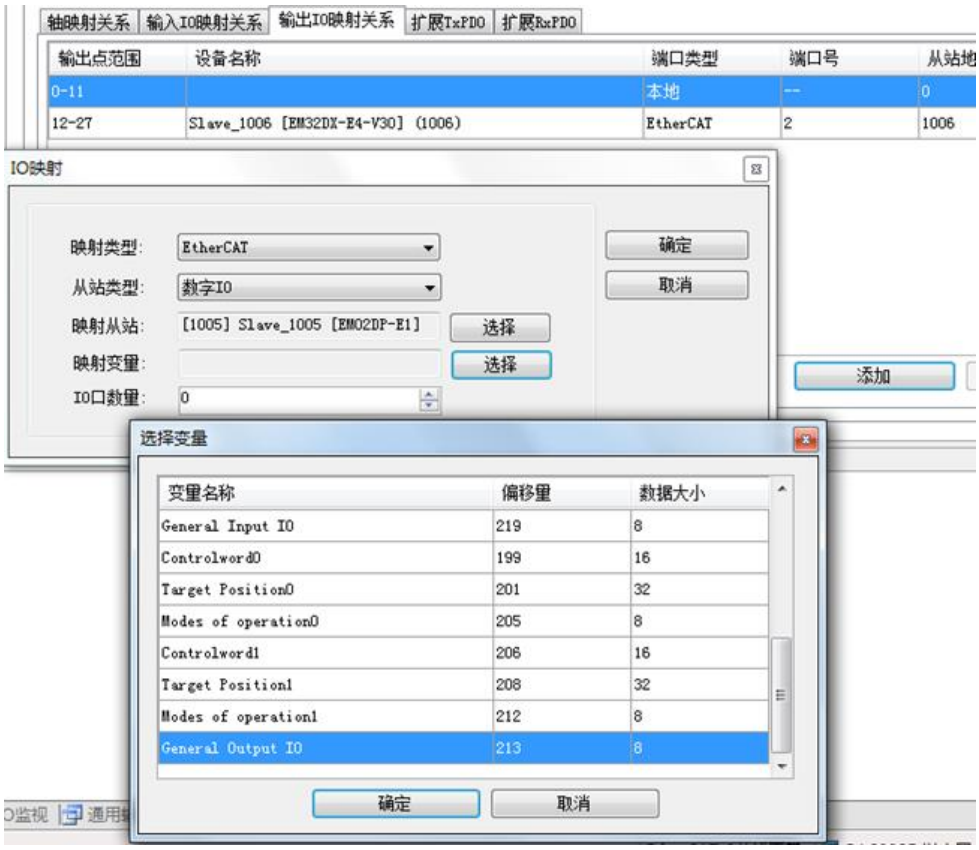


图 2.9.　手工添加数字输出接口

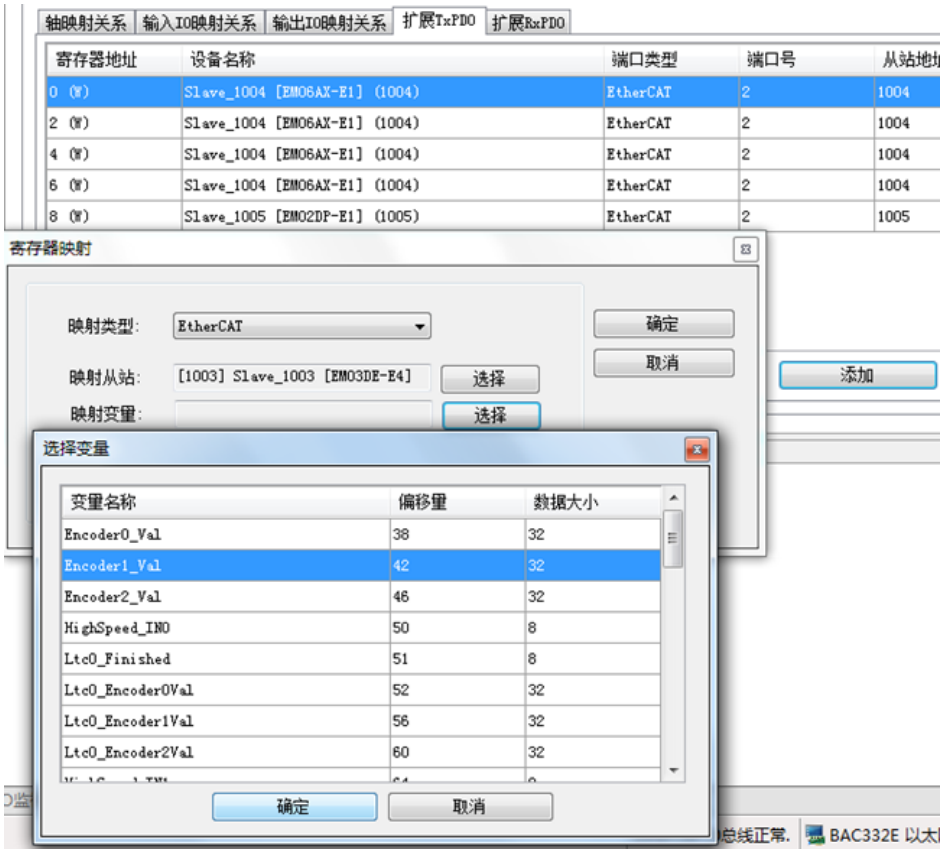


图 2.10.　手工添加 TxPDO 变量

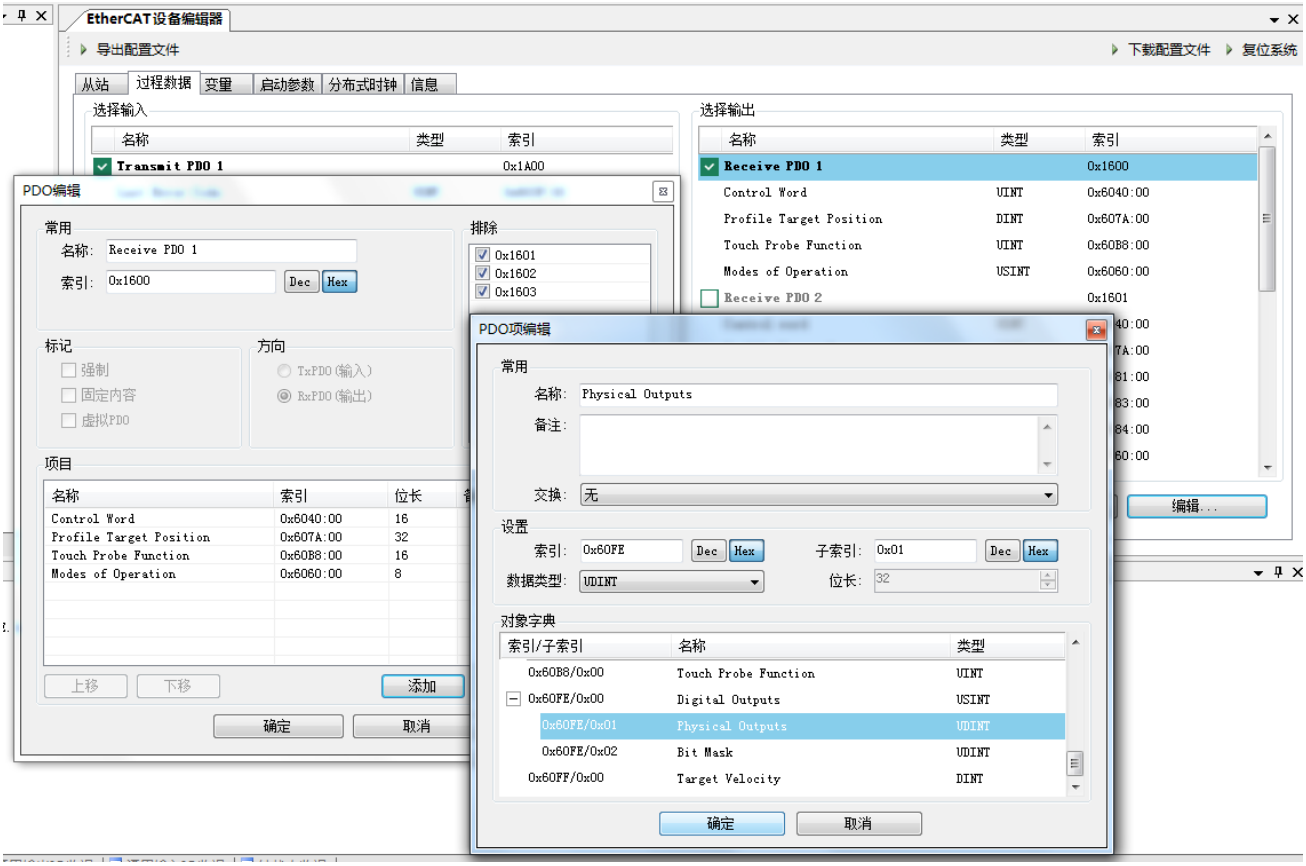


图 2.11.　手工添加从站的 PDO

在弹出的“PDO 编辑”界面中，点击下面的“添加”按钮，然后弹出“PDO 项编辑”窗口，如图 2.11 所示。在“对象字典”框中，找到“Digital Outputs”，点击其前面的“+”号，然后选择“Physical Outputs”，最后点击“确定”按钮，完成“Physical Outputs”。

再以同样的方法添加“Physical Outputs”下面的“Bit Mask”。完成后，如图 2.12 所示。添加完这 2 个 PDO 参数后，闭环步进电机驱动器 CL3-EC507 上的通用输出接口即可启用。

以同样的方法为节点 1007 松下伺服电机驱动器添加这 2 个 PDO 参数：Physical Outputs 和 Bit Mask，使其通用输出接口可控。



图 2.12.　手工添加从站的 PDO 参数：Digital Outputs 和 Bit Mask

5）设置总线周期并下载配置文件

在“EtherCAT 设备编辑器”界面中，将主站的“周期时间”设为 500 微秒，如图 2.13 所示。

之后，点击右侧的“下载配置文件”按钮，下载完成后，软件会自动复位系统。

至此，总线配置工作完成。

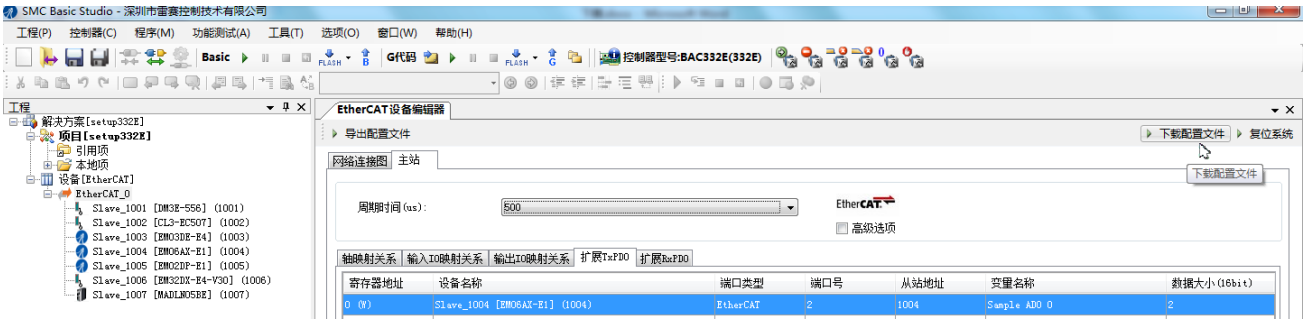


图 2.13.　设置总线周期，下载配置文件

5. 导出配置文件、加载配置文件

如图 2.14 所示，在“EtherCAT 设备编辑器”界面中，点击上方的“导出配置文件”按钮，可将总线的配置文件保存在电脑硬盘上备用。配置文件有 2 个：ethercat.eni 和 ethercat.ini。如图 2.16 所示。

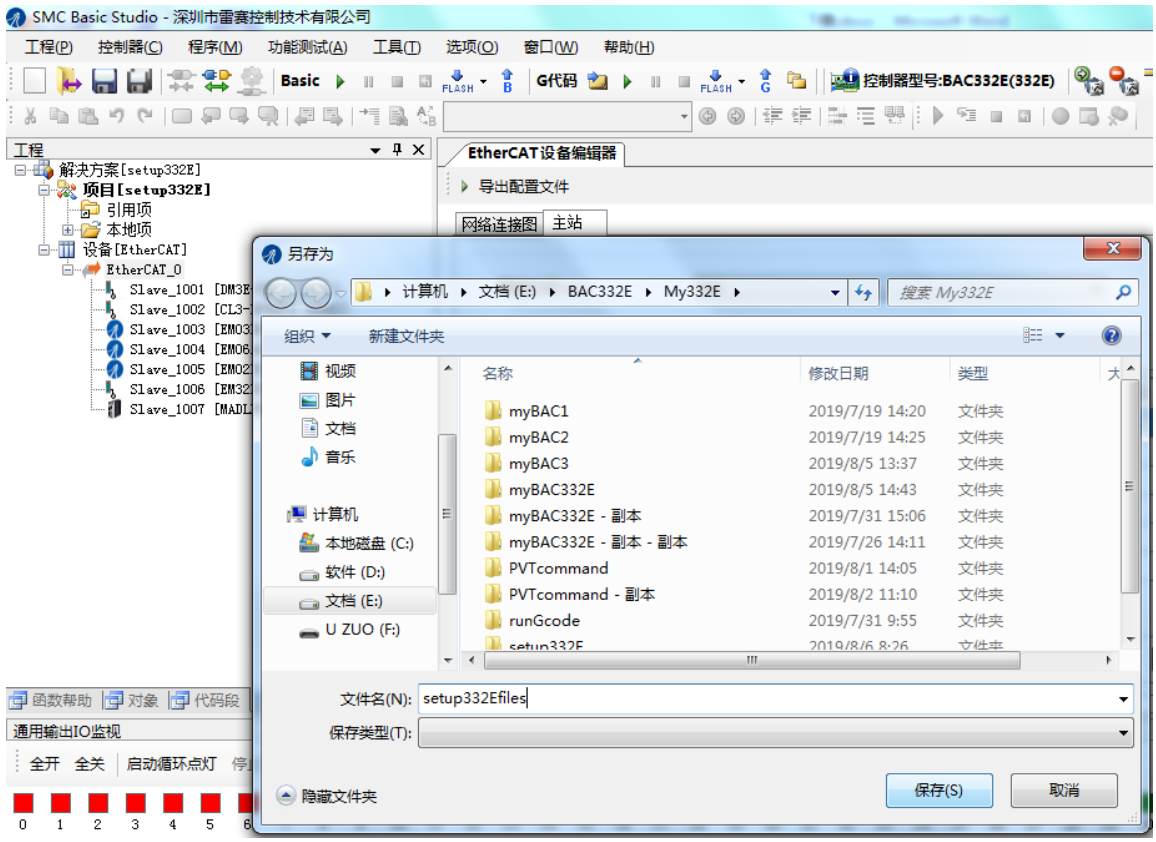


图 2.14.　导出配置文件

当使用一个新的 BAC332E 运动控制器时、或想将总线配置参数恢复到原有状态，可以将电脑上保存的配置文件下载至运动控制器。如图 2.15 所示，在 Basic Studio 菜单中，点击“工

具”、“EtherCAT 总线”、“EtherCAT 配置”之后，“EtherCAT 配置”窗口打开，如图 2.16 所示。

点击“打开配置文件”按钮，先选择 ethercat.eni 文件，打开后，点击下方的“下载配置文件”按钮；然后再点击“打开配置文件”按钮，选择 ethercat.ini 文件，打开后，点击“下载配置文件”按钮。

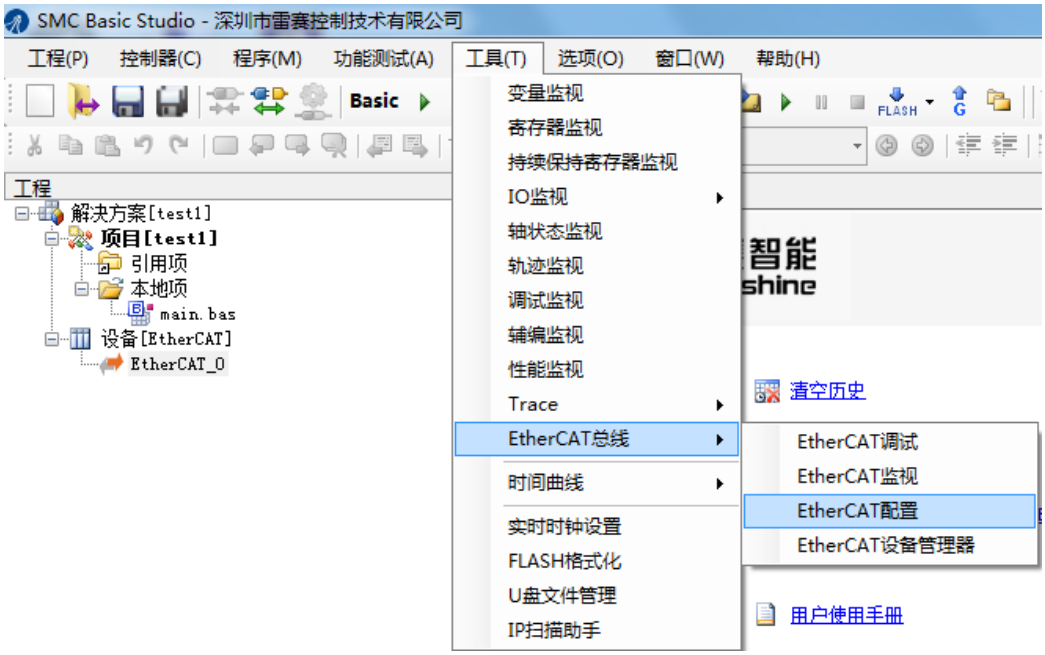


图 2.15.　打开 EtherCAT 配置窗口

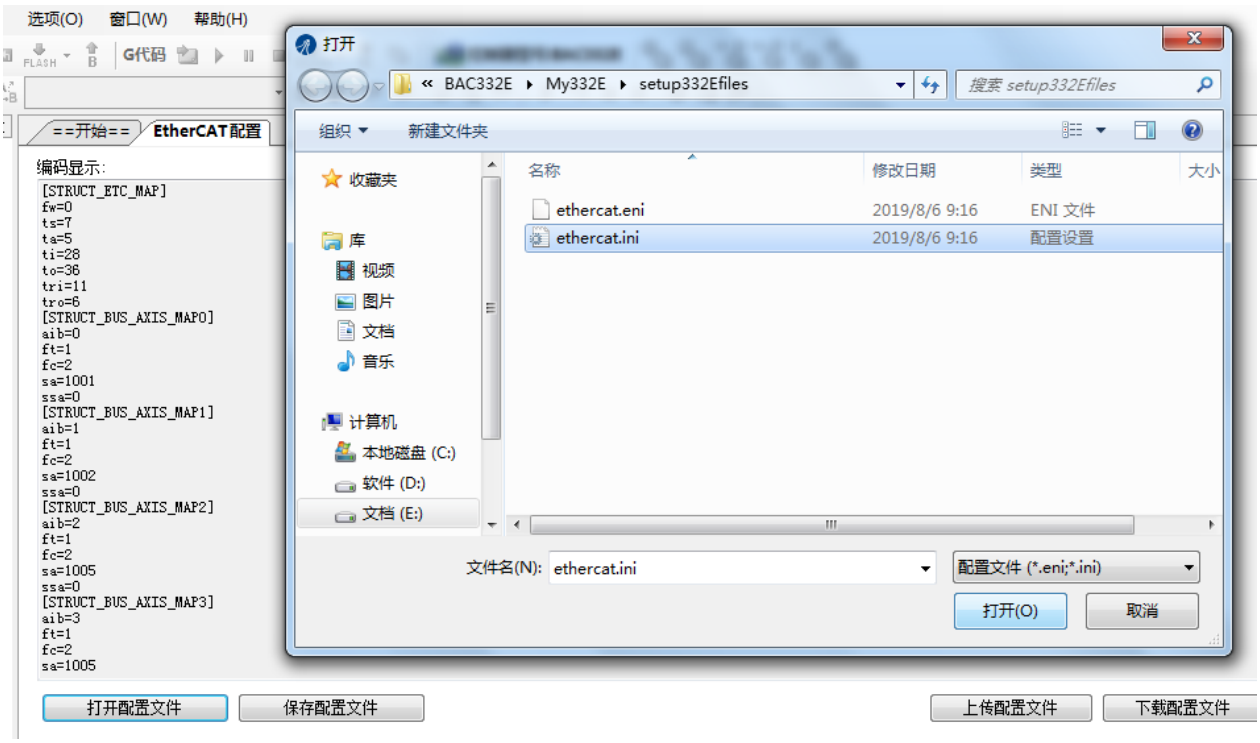


图 2.16.　打开配置文件、下载配置文件

6. 电机参数的设置

使用 SMC Basic Studio 可以读、写总线上各设备的对象字典中的参数。如图 2.15 所示，点击菜单中的“工具”、“EtherCAT 总线”、“EtherCAT 调试”之后，“EtherCAT 调试”窗口打开，如图 2.17 所示。

例如：读节点 1002 雷赛闭环步进电机的每转脉冲数。其对象字典的主索引号为 2001H、子索引号为 0，参数类型为 UINT，即 32 位无符号。将基本参数填入后，点击“读取参数”按钮，得到电机的每转脉冲为 40000。如图 2.17 所示。

将该值改为 20000，再点击“写入参数”按钮，参数即写入电机驱动器。

（注：其他参数的定义详见本文“上篇”）

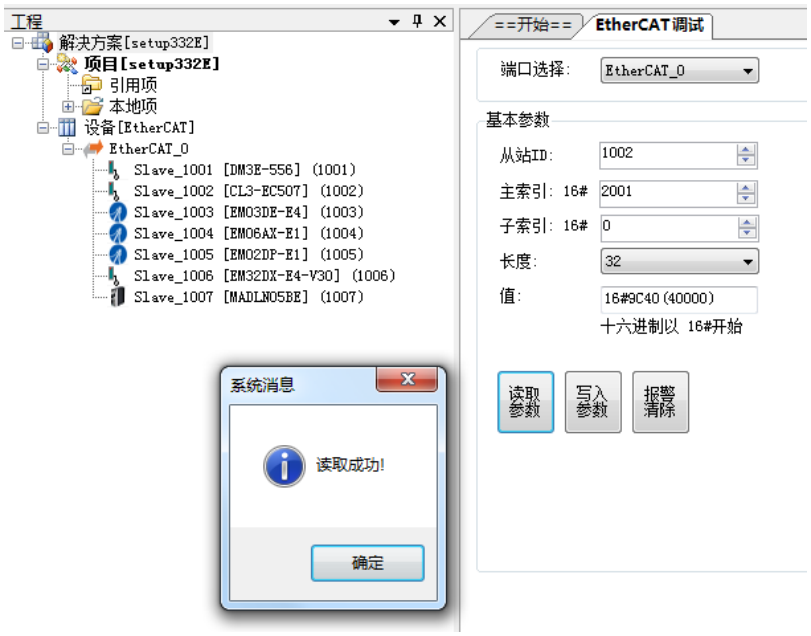


图 2.17.　读从站对象字典中的参数

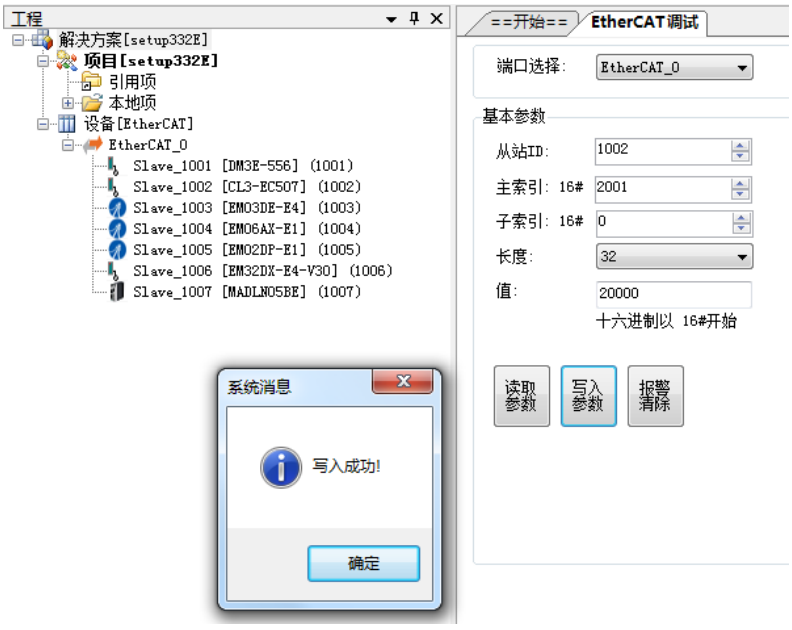


图 2.18.　改写从站对象字典中的参数

7. 电机、IO 接口的测试

完成总线配置、电机参数修改等工作后，可使用 SMC Basic Studio 进行总线上各设备及元件的测试。

1）电机测试

点击菜单中的“功能测试”，可以看到有多项测试功能。如图 2.19 所示。点击“单轴测试”，可进行电机的基本功能测试；同时，点击菜单中的“工具”、“轴状态监视”。得到的界面如图 2.19 所示，电机运动过程中，“轴状态监视”栏中的各参数随电机状态而变化。

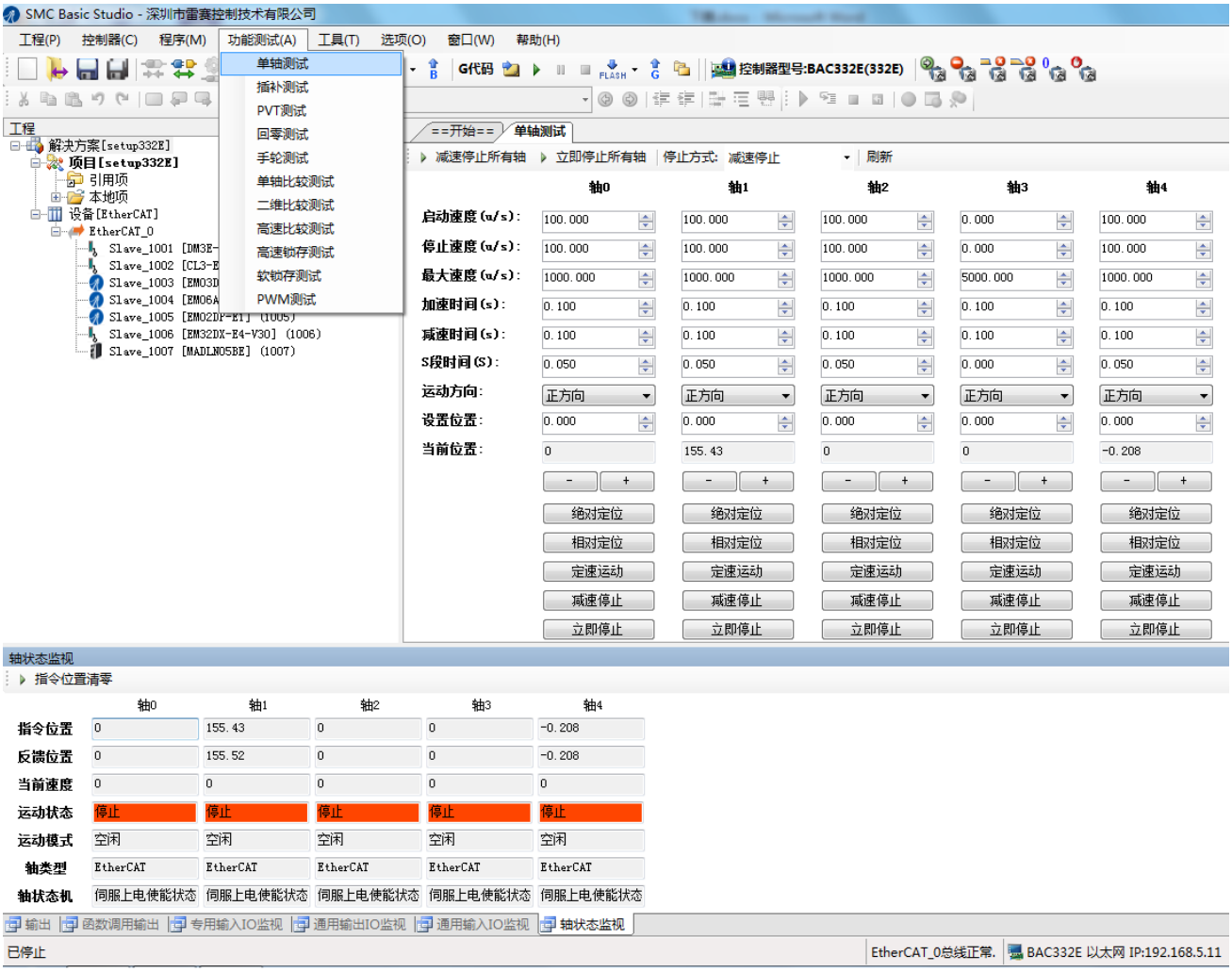


图 2.19. “单轴测试”和“轴状态监视”界面

在“单轴测试”界面中，可进行点位运动、定速运动、在线变速、在线变位等功能的测试。

注意：在进行电机功能测试之前，必须点击菜单右上角的图标，“所有轴使能开”。而且要保证电机限位信号处于未限位状态（参见图 2.20）。

2）IO 测试

点击菜单中的“工具”、“IO 监控”后，有 4 个选项：通用输入、输出 IO 监控和专用输入、输出 IO 监控。如图 2.20 所示。

当有输入信号时，界面中的输入口图标会变色。用鼠标点击输出口图标，可改变输出口的状态。

SMC Basic Studio 还有许多测试功能，详见其使用说明书。

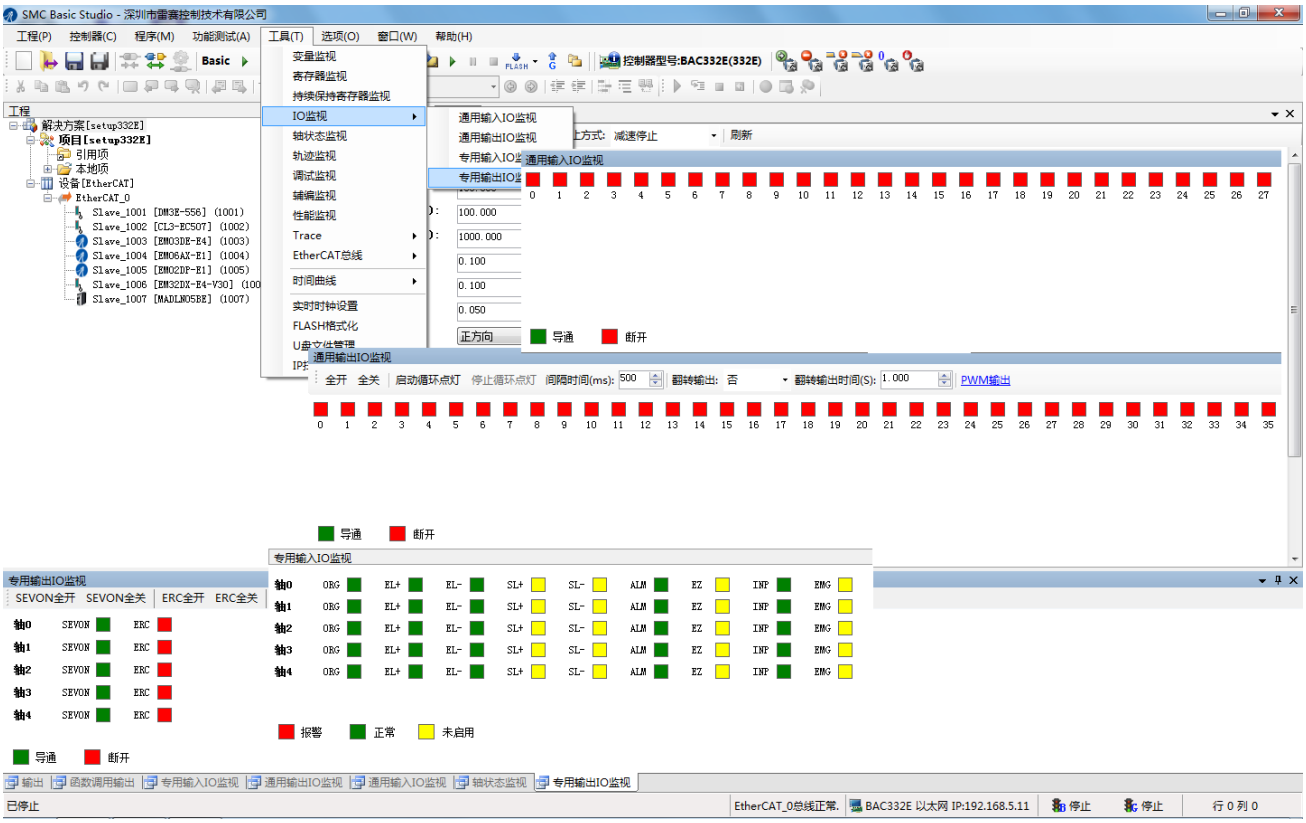


图 2.20.　通用和专用输入、输出口的监视界面

三．雷赛 BASIC 与 BAC332E 编程

2010 年,雷赛公司在国内第一个推出了用于运动控制器的 BASIC 语言,经过十年不断地改进完善，已经十分成熟，受到用户的好评。雷赛 BASIC 具有以下特点：

- 简单易学：基本指令和 Visual BASIC 兼容；运动控制指令格式与运动控制卡相同。
- 运行稳定安全：运行时不需要 PC 机，故安全可靠。可以用触摸屏作界面。
- 实时多任务：可以同时运行多个子任务。使程序简化、运行效率提高。
- 中断响应快：定时器信号、本地的通用数字输入信号、电机限位信号可作为中断源，中断处理程序使控制器的响应速度更快。
- 交互性强：程序为解释执行方式，可以实时交互控制，用户可以随时了解程序执行的状态。

在 SMC Basic Studio 软件中，点击工程栏中的“本地项”、“main.bas”，即进入到 Basic 程序编辑界面，如图 3.1 所示。

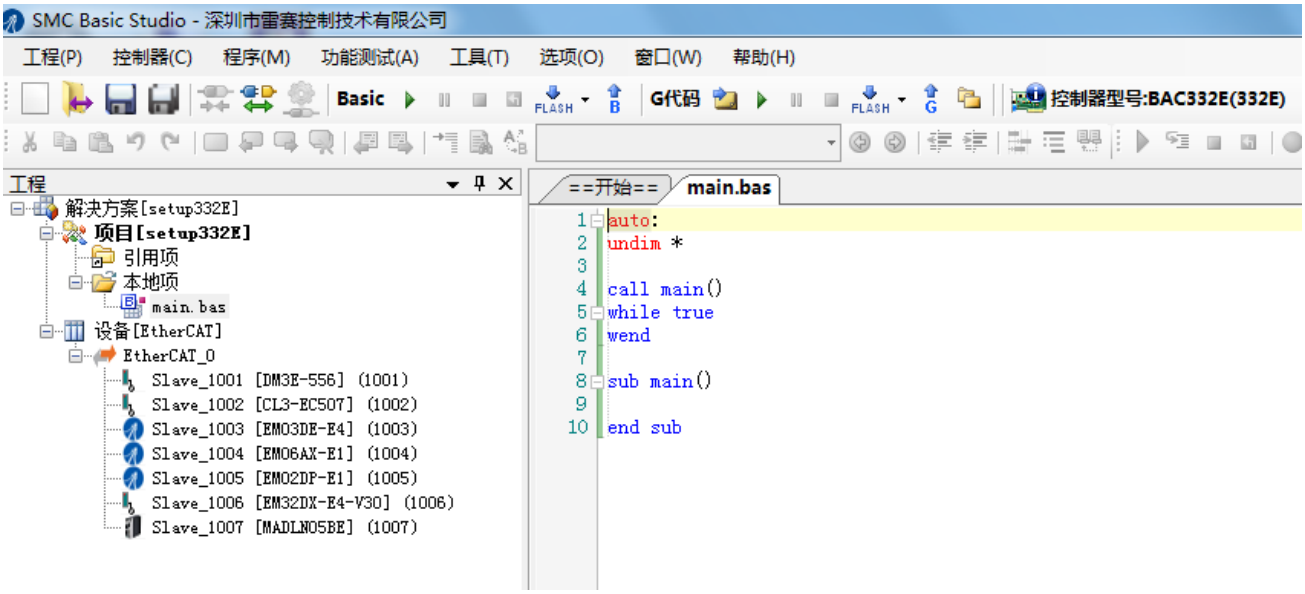


图 3.1.　Basic 程序编辑界面

例程 1：

编写如图 2.1 所示的 EtherCAT 总线运动控制系统的控制程序（其中各电机、模块的电路图详见本文“上篇”），实现以下功能：

1）各电机的参数和状态的初始化。位置单位为度。

2）轴 0、1、2、4 一个接一个地转动；正转一轮后再反转一轮，持续不断。

当 11 号输入口的开关闭合，电机运动停止；

当 0 号输入口的开关闭合，电机运动继续。

3）0~35 号输出口运行跑马灯，即每个输出口一个接一个地导通 0.2 秒、再关闭 0.2 秒。

当 3 号输入口的开关闭合，跑马灯停止；

当 1 号输入口的开关闭合，跑马灯继续。

4) 用定时器控制编码器模块读取闭环步进电机的位置；当电机绝对位置大于 3996 个脉冲时，轴 1 通用输出口 EX-OUT1 上的 LED 灯亮；否则，LED 灯灭。

5) 用定时器控制 AD/DA 模块。D/A 转换 0 号通道不断地、渐变地输出 0~10V 电压；AD1 号通道读取 DA0 的信号（DA0 输出信号已用导线接入到 AD1 通道上，电路图详见本文“上篇”），当检测到 DA0 输出的电压大于 5V 时，轴 4 的通用输出口 EX-OUT1 上的 LED 灯亮；否则，LED 灯灭。

在程序中，主任务 auto（主程序）负责进行电机参数和状态的初始化、电机运动控制、0 号输入口的检测。

跑马灯功能由子任务（子程序）LedControl 完成。

编码器模块、AD/DA 模块的控制由定时器中断任务（定时中断程序）完成。

1、3、11 号输入口的检测工作由输入口中断任务（外部信号中断程序）完成。

程序代码如下。

```
auto:        ' 以 auto 命名的主程序存入 flash，控制器上电后可自动运行
    undim *
    dim distance,i,k
    dim motor_flag,led_flag
    dim PortNum,NodeNum,Index,Zpos
    dim Vout,Vin,getOutput1,getOutput4
    dim axis(4),axisEquiv(4)
    axis(0)=0                          '定义轴号
    axis(1)=1
    axis(2)=2
    axis(3)=4
    axisEquiv(0)= 36000 / 360          ' 设置轴的位移单位为度
    axisEquiv(1)=  40000 / 360
    axisEquiv(2) = 12800 / 360
    axisEquiv(3) = 8388608.0 / 360
    led_flag=1
    Vout=0
    distance=360
    PortNum = 2             'EtherCAT 端口号
    NodeNum = 1003          '编码器模块节点号
    Index = 24577           '编码器 1 的主索引号
    nmcsSetNodeOd(PortNum, NodeNum, Index, 1, 8, 0)  '写对象字典， 0: AB 相 4 倍频模式
    nmcsSetNodeOd(PortNum, NodeNum,Index, 2, 8, 1)   ' 1: 正方向
    nmcsSetNodeOd(PortNum, NodeNum,Index,3, 32, 0)   ' 设置 Encoder1 的值为 0
    nmcsWriteRxpdoExtra( 2, 4, 1, 1)   ' 写 PDO 参数，AD/DA 使能打开

    int_enable(1)                      '中断使能：开
    run 1,LedControl
    timer_start(0, 100)                '开启 0 号定时器，定时时间为 100 毫秒
    for i=0 to 3                              '设置电机运动参数和状态
        smcSetEquiv(axis(i), axisEquiv(i))    '设置脉冲当量
        smcSetPositionUnit(axis(i), 0)        '位置清零
        smcSetSprofile( axis(i),0,0.05)       '设置 S 段时间
        smcSetProfileUnit(axis(i), 100, 2000, 0.1, 0.1, 100)
                                  '设置起始速度、运行速度、停止速度、加速时间、减速时间
        NMCSSetAxisEnable(axis(i))  '电机使能开
    next i
    motor_flag=1
    while true
```

```
        for i=0 to 3
            if motor_flag=1 then
                smcPmoveUnit(axis(i), distance, 0)        ' 电机旋转 1 圈
                while smcCheckdone(axis(i))=0             ' 等待电机停止
                wend
            endif
        next i
        if distance=360 then
            distance=-360
          else
            distance=360
        endif
        if SMCReadInBit(0)=0 then       ' 检测输入口 0 开关是否闭合
            smcSetPositionUnit(1, 0)                              ' 轴 1 位置清零
            nmcsSetNodeOd(PortNum, NodeNum,Index,3, 32, 0)    ' 设置 Encoder1 的值为 0
            motor_flag=1
        endif
    wend
end


LedControl:                 ' 跑马灯程序（子任务）
    while true
        for k=0 to 35
            if led_flag=1 then
                SMCWriteOutBit(k,0) ' 输出口 k 的指示灯亮
                delay(200)          ' 延时 200 毫秒
                SMCWriteOutBit(k,1) ' 输出口 k 的指示灯灭
                delay(200)
            endif
        next k
    wend
end


ontimer0:                                   ' 0 号定时器中断处理入口
    nmcsReadTxpdoExtra( 2, 9, 2,Zpos)           ' 读 2 号节点 PDO 参数，编码器 1 的值
    nmcsGetAxisIoOut(1, getOutput1)             ' 读 1 号轴的输出口
    if abs(Zpos)>3996 then
       getOutput1= set_bit (18,getOutput1)      '  输出口第 18 位置 1,LED 亮
      else
       getOutput1=  clear_bit (18,getOutput1)   '输出口第 18 位清零
    endif
    nmcsSetAxisIoOut(1, getOutput1)             ' 写 1 号轴的输出口
    nmcsWriteRxpdoExtra(2, 0, 2, Vout)      ' 写 2 号节点 PDO 参数，DA 的 0 号通道输出电压信号
    Vout = Vout + 2000
    if Vout > 100000 then                   ' DA 的输出电压为 0~10V
       Vout = 0
    endif
    nmcsReadTxpdoExtra(2, 2, 2, Vin)            'A/D 转换，读 AD 的 1 号通道
    print "Zpos =",Zpos,   ";    Vin =",Vin
    nmcsGetAxisIoOut(4, getOutput4)             ' 读 4 号轴的输出口
    if Vin>50000 then                           ' 如果电压高于 5V
        getOutput4= set_bit (16,getOutput4)     ' 输出口第 16 位置 1,LED 亮
      else
       getOutput4=  clear_bit (16,getOutput4)   ' 输出口第 16 位清零
    endif
    nmcsSetAxisIoOut(4, getOutput4)             ' 写 4 号轴的输出口
```

```
    timer_start(0, 100)                          ’再次开启 1 号定时器
end


oninon1:          ’输入口 IN1 触发中断处理入口
    led_flag=1                                   ’跑马灯开始
end


oninon3:          ’输入口 IN3 触发中断处理入口
    led_flag=0                                   ’跑马灯停止
end


oninon11:          ’输入口 IN11 触发中断处理入口
    SMCEmgStop                                   ’所有轴急停止
    motor_flag=0
end
```

四．#C 与 BAC332E 编程

BAC332E 运动控制器也可以象运动控制卡一样和 PC 机一起使用，即：用高级语言在 PC 机上编程，通过以太网接口向 BAC332E 控制器发送指令。

和雷赛的 DMC E3032 EtherCAT 总线型运动控制卡相比，BAC332E 功能更强：可执行 G 代码、具有样条插值功能（PVT 函数）。

一）基本功能的实现

例程 2：

本例程界面如图 4.1 所示。其主要功能有：单轴点位运动、IO 信号的控制、编码器模块检测位置、AD/DA 模块的模拟量输出和检测、控制器复位、配置文件的下载、读写控制器上的实时时钟等。

例程的原代码请见附录。

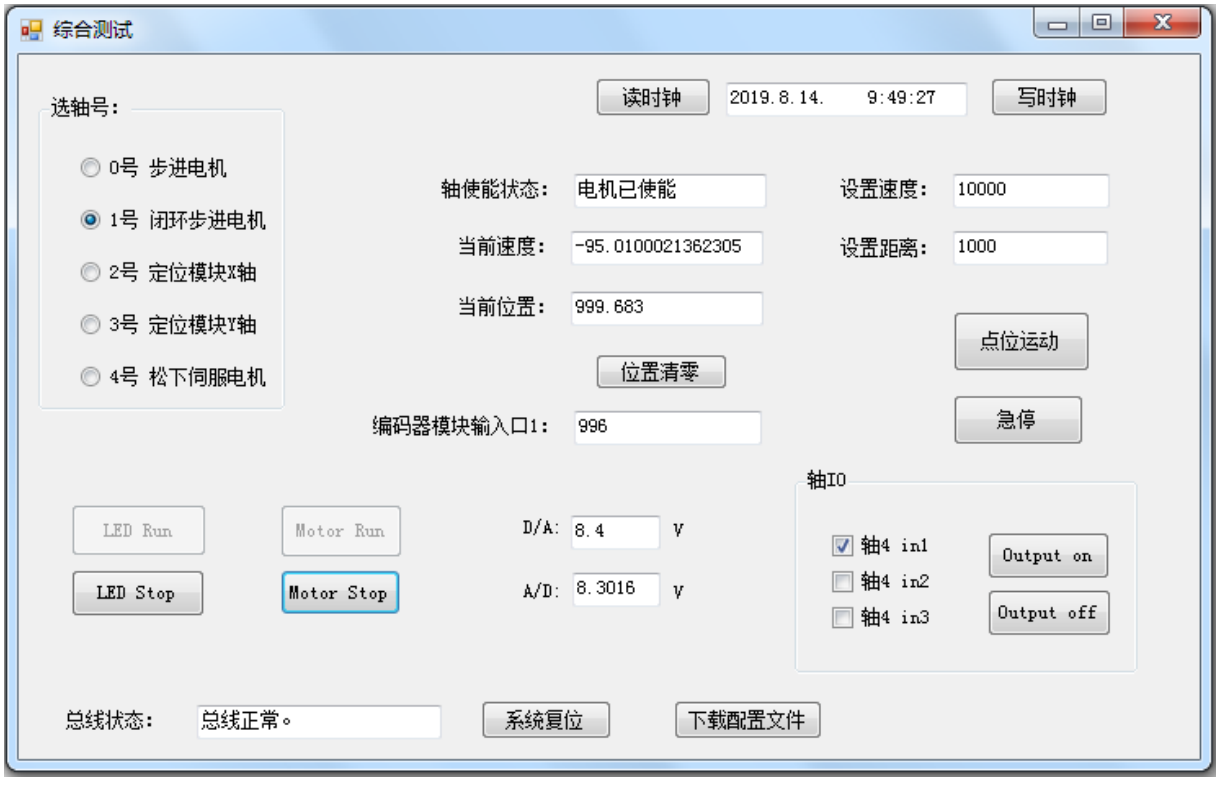


图 4.1. BAC332E 运动控制器的综合测试程序

编写 BAC332 运动控制器的程序时，需要注意以下问题：

1. 和编写运动控制卡的程序一样，在程序中要添加雷赛运动控制函数的头文件 LTSMC.cs；并要将动态链接库文件 LTSMC.dll 拷贝至项目文件夹中的 bin 文件夹下的 Debug 文件夹中。
2. 打开“电机使能”的响应速度较慢，要延时 100 毫秒后，再启动电机。
3. BAC332E 运动控制器和 IO 扩展模块上的通用数字信号输出可以使用位操作指令：smc_write_outbit(……)；也可以使用端口操作指令：smc_write_outport(……)。但电机

的通用数字信号输出口只能使用端口操作指令：nmcs_set_axis_io_out(……)。

二）执行 G 代码

G 代码是数控机床上使用的标准加工指令。使用专用软件可以很容易地将 AutoCAD 绘制的图形转换为 G 代码，所以，用 G 代码加工复杂的图形非常方便。

例程 3：

本例程采用 C#编程，在 PC 机上控制 BAC332E 运动控制器执行 G 代码指令。因为，使用 PC 机便于使用 AutoCAD 软件生成 G 代码、存储 G 代码文件、显示图形和轨迹。

该程序的界面如图 4.2 所示。

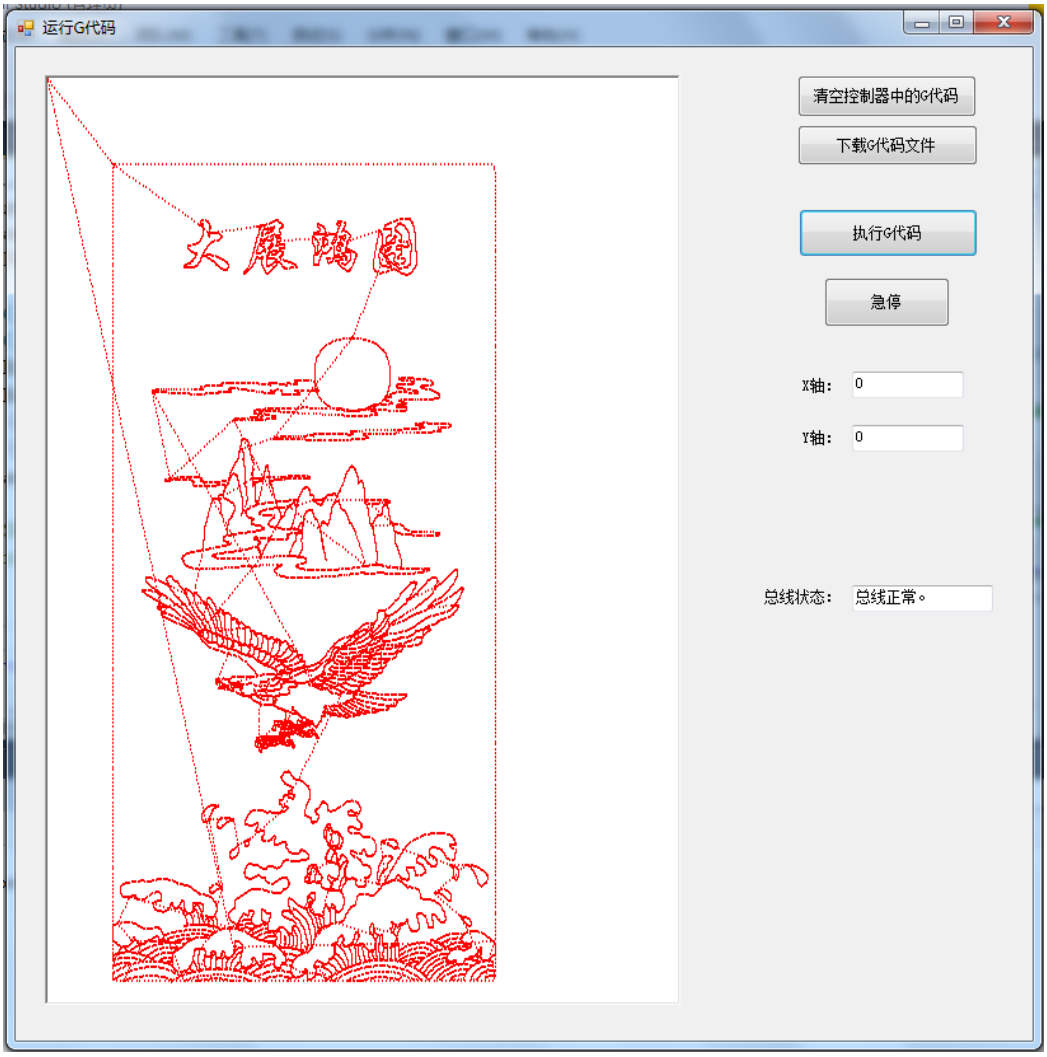


图 4.2.　G 代码加工结果

该程序调用的 G 代码文件有 9869 条指令，如下所示。

```
N1 G90                          ; 绝对坐标
N2 G00 X20.29 Y15.4673          ; 快速定位
N3 M80 S1                       ; 输出口 S1 开，打开激光器
N4 G04 P100                     ; 延时 100 毫秒
N5 G01 X20.29 Y105.6049         ; 直线插补
```

```
N6 G01 X212.9628 Y105.6049
N7 G01 X212.9638 Y15.4675
N8 G01 X20.29 Y15.4673
N9 M81 S1                          ; 输出口 S1 关,关闭激光器
N10 G04 P50
N11 G00 X34.7555 Y36.2336
N12 M80 S1                          ; 输出口 S1 开,打开激光器
N13 G04 P100
N14 G01 X34.2419 Y36.1505
N15 G01 X33.938 Y36.1293
N16 G01 X33.7541 Y36.1959
N17 G01 X33.5149 Y36.4678
N18 G01 X33.4531 Y36.6087
……
……
……
N9861 G01 X205.0388 Y45.4005
N9862 G01 X205.2675 Y44.952
N9863 G01 X205.6076 Y44.5312
N9864 G01 X206.0186 Y44.1903
N9865 G01 X206.1753 Y44.0969
N9866 M81 S1                        ; 输出口 S1 关,关闭激光器
N9867 G04 P50
N9868 G00 X0.0 Y0.0
N9869 M02                           ; 程序结束
```

使用 G 代码控制运动轨迹时，需要注意以下问题：

1. G 代码可以控制 3 个轴运动，即：X、Y 和 Z 轴。这 3 个轴只能是 0、1、2 号轴。
2. G 代码运动的速度指令和点位运动、插补运动的指令不一样。
3. BAC332E 运动控制器中可存储多个 G 代码。但存储空间有限，允许文件的最大容量为 8M，约 20 万行 G 代码。

并非所有的雷赛 BASIC 指令都有对应的 API 函数。当遇到想调用某个指令，而又没有对应的 API 函数时，可以使用“万能函数”smc_basic_command（……），间接调用 BASIC 指令。如本例程中 G 代码速度的设置即采用了该函数。

本例程的 C#程序代码见附录。

三）样条插值函数（PVT 函数）

在控制自动化设备做复杂的动作时，直线插补、圆弧插补也不能完全满足需求。为此，雷赛公司开发了样条插值函数（PVT 函数），可让多轴根据自定义的位置-时间曲线联动，实现复杂的多轴轨迹控制（允许 32 轴参与样条插值）。PVT 函数采用的是 3 次样条插值函数，所以只需要给出较少的点，就可以获得很精确的轨迹。该功能在机械手、多轴激光焊接、包装机械等设备上发挥了良好的作用。

例程 4：

采用 PVT 函数实现空间椭圆轨迹运动。

一平面上的椭圆轨迹，长轴为 a，短轴为 b，中心为 C，圆弧角为α，如图 4.3 所示。其参数方程如下。

$$\begin{cases} x = x0 + a\cos(\alpha) \\ z = z0 + b\sin(\alpha) \end{cases}$$

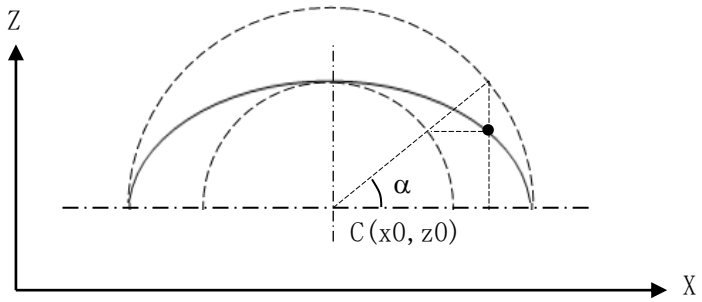
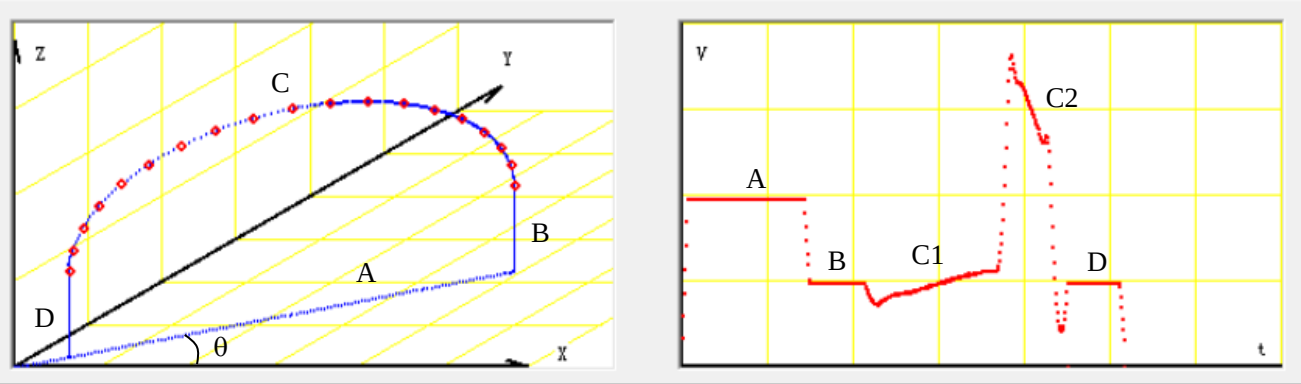


图 4.3.　平面上的椭圆

若 XZ 平面绕 Z 轴旋转θ角，如图 4.4（a）所示，该椭圆方程变为：

$$\begin{cases} x = (x0 + a\cos(\alpha))\cos(\theta) \\ y = (x0 + a\cos(\alpha))\sin(\theta) \\ z = z0 + b\sin(\alpha) \end{cases} \qquad (4.1)$$



(a)　　(b)

图 4.4.　空间椭圆轨迹及其速度曲线

如图 4.4 所示，将半个椭圆轨迹按圆弧角均分为 18 段，共有 19 个点。

用数组 tData[i]存储各点的时间，用数组 PxData[i]、PyData[i]、PzData[i]存储各点的坐标位置，并用 LTSMC.smc_pvts_table_unit(……)函数将时间和位置数据存入控制器；然后执行 LTSMC.smc_pvt_move(……)函数，即开始样条插值运动。

注意：

1. 样条曲线上各点的坐标位置为相对坐标，即：起点坐标为 0；
2. 起点的时间也为 0。
3. 通过调整各点的时间，可以控制轨迹的运动速度。
   （参见程序代码：前 9 段，每段时间为 0.3 秒；后 8 段每段时间为 0.1 秒；第 10 段时间为 0.2 秒）

图 4.4(a)中，轨迹 A、B、D 为直线插补，C 段为 PVT 函数的轨迹。C 段蓝色曲线为实测轨迹，红色点是理论值。显示，样条插值函数位置精度很高。

图 4.4(b)为轨迹运动的矢量速度曲线。C1 段为椭圆前半段速度曲线，C2 段为椭圆后半段速度曲线。后半段的速度是前半段的 3 倍，这是因为前 9 段和后 8 段的每段时间相差 3 倍。由于每段曲线的长度不一样（每段曲线对应的圆弧角相等），所以，椭圆轨迹的速度也是渐变的。除 C2 和 D 段的速度衔接不太好外，其他衔接点比较平顺。

本例程的 C#程序代码见附录。

五．参考文献

[1] 深圳市雷赛控制技术有限公司. EtherCAT总线运动控制器 BAC-332E 用户手册. 2019年4月
[2] 深圳市雷赛控制技术有限公司. 雷赛运动控制器Basic指令编程手册 V1.5. 2017年4月
[3] 深圳市雷赛控制技术有限公司. 雷赛运动控制器API函数编程手册 V1.5. 2017年5月
[4] 深圳市雷赛控制技术有限公司. 雷赛运动控制器SMC Basic Studio使用手册. 2016年12月

六. 附录　例程代码

1. 例程 2 代码

```
using System;
using System.Drawing;
using System.Text;
using System.Threading;                    // 多线程的头文件
using System.Windows.Forms;
using System.IO;                           // 文件读写的头文件
using Leadshine;                           // 雷赛指令的头文件


namespace myBAC2
{
    public partial class Form1 : Form
    {
        ushort _ConnectNo = 0;
        ushort axis = 0;
        double axisEquiv = 10;
        uint Vout = 0, Vin = 0;
        int motorRun = 0;

        ushort PortNum = 2;      // EtherCAT端口号
        ushort NodeNum3 = 1003; // 编码器模块节点号
        ushort Index;
        short getOutput;
        uint setOutput;
        int year = 0, month = 0, day = 0, hour = 0, min = 0, sec = 0;
        string realTime;

        private Thread runLED;  // 定义线程名 runLED

        public Form1()
        {
            InitializeComponent();
        }


        private void Form1_Load(object sender, EventArgs e)
        {
            short res = LTSMC.smc_board_init(_ConnectNo, 2, "192.168.5.11", 1000000);
                                    //连接控制器
            if (res != 0)
                MessageBox.Show("连接错误!", "出错");
            else
            {
                radioButton1.Checked = true;
                axis = 0;
                LTSMC.nmcs_set_axis_enable(_ConnectNo, axis);    //轴使能开
                LTSMC.smc_set_position_unit(_ConnectNo, axis, 0);//位置清零

                LTSMC.nmcs_write_rxpdo_extra(_ConnectNo, 2, 4, 1, 1);// AD/DA使能打开

                Index = 0x6001; // 设置编码器1
                LTSMC.nmcs_set_node_od(_ConnectNo, PortNum, NodeNum3, Index, 0x01, 8, 0);
                // 0: AB相4倍频模式
```

```
            LTSMC.nmcs_set_node_od(_ConnectNo, PortNum, NodeNum3, Index, 0x02, 8, 1);
            // 1: 正方向
            LTSMC.nmcs_set_node_od(_ConnectNo, PortNum, NodeNum3, Index, 0x03, 32, 0);
            // 设置Encoder1的值为0
            timer1.Start();
        }
    }


    private void button1_Click(object sender, EventArgs e) //点位运动
    {
        double start = 100;
        double speed = double.Parse(textBox4.Text);
        double stop = 100;
        double acc = 0.01;
        double dec = 0.01;
        double dis = double.Parse(textBox5.Text); ;
        double s = 0.05;
        ushort State_Machine = 0;

        LTSMC.nmcs_get_axis_state_machine(_ConnectNo, axis, ref State_Machine);
        if (State_Machine == 4)
        {
            LTSMC.smc_set_equiv(_ConnectNo, axis, axisEquiv);
            LTSMC.smc_set_s_profile(_ConnectNo, axis, 0, s);//设置S段时间
            LTSMC.smc_set_profile_unit(_ConnectNo, axis, start, speed, acc, dec, stop);
              //设置起始速度、运行速度、停止速度、加速时间、减速时间
            LTSMC.smc_set_dec_stop_time(_ConnectNo, axis, dec);
            LTSMC.smc_pmove_unit(_ConnectNo, axis, dis, 0);//定长运动
            button1.Enabled = false;
            while (LTSMC.smc_check_done(_ConnectNo, axis) == 0)
                Application.DoEvents(); // 处理当前消息队列中消息
            button1.Enabled = true;
        }
        else
        {
            MessageBox.Show("轴状态机不对  失败！状态机：" + State_Machine, "提示",
                            MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }


    private void button2_Click(object sender, EventArgs e)  //位置清零
    {
        LTSMC.smc_set_position_unit(_ConnectNo, axis, 0);// 轴的位置清零
        Index = 0x6001;
        LTSMC.nmcs_set_node_od(_ConnectNo, PortNum, NodeNum3, Index, 0x03, 32, 0);
        // 设置编码器1的值为0
    }


    private void button3_Click(object sender, EventArgs e)// 轴减速停止
    {
        LTSMC.smc_set_dec_stop_time(_ConnectNo, axis, 0.1);
        LTSMC.smc_stop(_ConnectNo, axis, 0);
    }
```

```
private void button4_Click(object sender, EventArgs e)// motor run
{                                                      // 电机连续正反转
    double start = 100;
    double speed = double.Parse(textBox4.Text);
    double stop = 100;
    double acc = 0.01;
    double dec = 0.01;
    double dis = double.Parse(textBox5.Text); ;
    double s = 0.05;
    button4.Enabled = false;
    LTSMC.smc_set_equiv(_ConnectNo, axis, axisEquiv);
    LTSMC.smc_set_s_profile(_ConnectNo, axis, 0, s);
    LTSMC.smc_set_profile_unit(_ConnectNo, axis, start, speed, acc, dec, stop);
    LTSMC.smc_set_dec_stop_time(_ConnectNo, axis, dec);
    motorRun = 1;
    while (motorRun == 1)
    {
        LTSMC.smc_pmove_unit(_ConnectNo, axis, dis, 0);//电机正转
        button6_Click(sender, e);
        while (LTSMC.smc_check_done(_ConnectNo, axis) == 0)
            Application.DoEvents();
        Thread.Sleep(100);
        LTSMC.smc_pmove_unit(_ConnectNo, axis, -1 * dis, 0);//电机反转
        button7_Click(sender, e);
        while (LTSMC.smc_check_done(_ConnectNo, axis) == 0)
            Application.DoEvents();
        Thread.Sleep(100);
    }
}


private void button5_Click(object sender, EventArgs e)//motor stop
{
    button4.Enabled = true;
    motorRun = 0;
}


private void button6_Click(object sender, EventArgs e) // Output on
{
    getOutput = LTSMC.nmcs_get_axis_io_out(_ConnectNo, 1);
    setOutput = (uint)getOutput;
    setOutput = setOutput | 0x40000;   // 按位或运算，第18位置1
    LTSMC.nmcs_set_axis_io_out(_ConnectNo, 1, setOutput);// 轴1 out18 LED亮

    getOutput = LTSMC.nmcs_get_axis_io_out(_ConnectNo, 4);
    setOutput = (uint)getOutput;
    setOutput = setOutput | 0x10000;   // 按位或运算，第16位置1
    LTSMC.nmcs_set_axis_io_out(_ConnectNo, 4, setOutput);// 轴 out16 LED亮
}


private void button7_Click(object sender, EventArgs e) // Output off
{
    getOutput = LTSMC.nmcs_get_axis_io_out(_ConnectNo, 1);
    setOutput = (uint)getOutput;
    setOutput = setOutput & 0xFFFBFFFF;   // 按位与运算，第18位清0
    LTSMC.nmcs_set_axis_io_out(_ConnectNo, 1, setOutput);// 轴 out18 LED灭
```

```
    getOutput = LTSMC.nmcs_get_axis_io_out(_ConnectNo, 4);
    setOutput = (uint)getOutput;
    setOutput = setOutput & 0xFFFEFFFF;   // 按位与运算，第16位清
    LTSMC.nmcs_set_axis_io_out(_ConnectNo, 4, setOutput);// 轴 out16 LED灭
}


private void radioButton1_CheckedChanged(object sender, EventArgs e)
{
    if (radioButton1.Checked)                          //0号轴被选中
    {
        LTSMC.nmcs_set_axis_disable(_ConnectNo, axis);    //原轴使能关
        axis = 0;
        LTSMC.nmcs_set_axis_enable(_ConnectNo, axis);     //本轴使能开
        Thread.Sleep(100);                                //线程休眠100毫秒
    }
}


private void radioButton2_CheckedChanged(object sender, EventArgs e)
{
    if (radioButton2.Checked)
    {
        LTSMC.nmcs_set_axis_disable(_ConnectNo, axis);
        axis = 1;
        LTSMC.nmcs_set_axis_enable(_ConnectNo, axis);
        Thread.Sleep(100);
    }
}


private void radioButton3_CheckedChanged(object sender, EventArgs e)
{
    if (radioButton3.Checked)
    {
        LTSMC.nmcs_set_axis_disable(_ConnectNo, axis);
        axis = 2;
        LTSMC.nmcs_set_axis_enable(_ConnectNo, axis);
        Thread.Sleep(100);
    }
}


private void radioButton4_CheckedChanged(object sender, EventArgs e)
{
    if (radioButton4.Checked)
    {
        LTSMC.nmcs_set_axis_disable(_ConnectNo, axis);
        axis = 3;
        LTSMC.nmcs_set_axis_enable(_ConnectNo, axis);
        Thread.Sleep(100);
    }
}


private void radioButton5_CheckedChanged(object sender, EventArgs e)
{
```

```
    if (radioButton5.Checked)
    {
        LTSMC.nmcs_set_axis_disable(_ConnectNo, axis);
        axis = 4;
        LTSMC.nmcs_set_axis_enable(_ConnectNo, axis);
        Thread.Sleep(100);
    }
}


private void LEDrunButton_Click(object sender, EventArgs e)  //启动LED跑马灯
{
    LEDrunButton.Enabled = false;
    runLED = new Thread(LED);                 // 创建线程
    runLED.Priority = ThreadPriority.Normal; // 该线程优先级一般
    runLED.Start();                           // 启动线程
}


private void LEDstopButton_Click(object sender, EventArgs e)  //停止LED跑马灯
{
    runLED.Abort();                // 终止线程
    LEDrunButton.Enabled = true;
}


private void LED()                        //LED跑马灯控制
{
    uint OutputVal = 0X1, OutputVal2 = 0X1, NotOutputVal;
    ushort i;
    while (true)
    {
        NotOutputVal = ~OutputVal;                                  // 非运算
        LTSMC.smc_write_outport(_ConnectNo, 0, NotOutputVal);   // OutputVal=0~31
        Thread.Sleep(200);                                          // 延时200毫秒
        OutputVal = OutputVal << 1;                                 // 左移1位
        if (OutputVal == 0)                                         // 左移了32位?
        {
            NotOutputVal = ~OutputVal;                              // 非运算
            LTSMC.smc_write_outport(_ConnectNo, 0, NotOutputVal);   // OutputVal=0~31
            for (i = 1; i < 6; i++)
            {
                NotOutputVal = ~OutputVal2;                           // 非运算
                LTSMC.smc_write_outport(_ConnectNo, 1, NotOutputVal);
                                                                    // OutputVal=32~63
                Thread.Sleep(200);                                  // 延时200毫秒
                OutputVal2 = OutputVal2 << 1;
            }
            OutputVal = 0X1;
            OutputVal2 = 0X1;
        }
    }
}


private void button8_Click(object sender, EventArgs e) // 系统复位
{
    textBox6.BackColor = Color.White;
```

```
        timer1.Stop();
        LTSMC.smc_soft_reset(_ConnectNo);        // 控制器软复位
        LTSMC.smc_board_close(_ConnectNo);

        for (int i = 100; i > 0; i--)              // 复位耗时10s左右
        {
            textBox6.Text = "控制器复位中 .... " + i.ToString();
            Application.DoEvents();
            Thread.Sleep(100);                     // 延时100毫秒
        }
        Form1_Load(sender, e);                     // 控制器初始化
        MessageBox.Show("控制器复位完成，请确认总线状态。");
        timer1.Start();
    }


    private void button9_Click(object sender, EventArgs e) // 下载配置文件
    {
        OpenFileDialog f1 = new OpenFileDialog();
        f1.Filter = "ini|*.ini";
        DialogResult res1 = f1.ShowDialog();
        if (res1 == DialogResult.OK)
        {
            FileStream fs = File.Open(f1.FileName, FileMode.Open, FileAccess.Read,
            FileShare.ReadWrite);
            StreamReader sr = new StreamReader(fs);
            string str = sr.ReadToEnd();
            sr.Close();
            fs.Close();
            byte[] buffer = Encoding.UTF8.GetBytes(str);
            byte[] fileincontrol = Encoding.UTF8.GetBytes("");
            ushort filetype = 201;
            short ret = -1;
            ret = LTSMC.nmcs_set_cycletime(_ConnectNo, 2, 500);       //设置总线周期
            ret = LTSMC.smc_download_memfile(_ConnectNo, buffer, (uint)buffer.Length,
                  fileincontrol, filetype);      //下载配置文件
            if (ret == 0)
            {
                MessageBox.Show("  ini文件下载成功！\n\n  请继续下载eni文件。");
            }
        }

        OpenFileDialog f2 = new OpenFileDialog();
        f2.Filter = "eni|*.eni";
        DialogResult res2 = f2.ShowDialog();
        if (res2 == DialogResult.OK)
        {
            FileStream fs = File.Open(f2.FileName, FileMode.Open, FileAccess.Read,
                            FileShare.ReadWrite);
            StreamReader sr = new StreamReader(fs);
            string str = sr.ReadToEnd();
            sr.Close();
            fs.Close();
            byte[] buffer = Encoding.UTF8.GetBytes(str);
            byte[] fileincontrol = Encoding.UTF8.GetBytes("");
            ushort filetype = 200;
            short ret = -1;
            ret = LTSMC.smc_download_memfile(_ConnectNo, buffer, (uint)buffer.Length,
```

```
                fileincontrol, filetype);     //下载ENI文件
            if (ret == 0)
            {
                MessageBox.Show("eni文件下载成功！");
            }
        }
    }


    private void button10_Click(object sender, EventArgs e)//读实时时钟
    {
        LTSMC.smc_rtc_get_time(_ConnectNo, ref year, ref month, ref day,
                               ref hour, ref min, ref sec);
        realTime = year.ToString()+"."+ month.ToString() + "." + day.ToString() + ".     "
                   + hour.ToString() + ":" + min.ToString() + ":" + sec.ToString();
        textBox10.Text = realTime;
    }


    private void button11_Click(object sender, EventArgs e)//写实时时钟
    {
        string temp, temp1;
        int lengthData;
        temp1 =  textBox10.Text;

        lengthData = temp1.IndexOf(".");            // 定位第一个“.”
        temp = temp1.Substring(0, lengthData);      // 截取第一个“.”之前的字符
        year = int.Parse(temp);                     // 获取year数据
        temp1 = temp1.Substring(lengthData + 1);    // 截取第一个“.”之后的字符

        lengthData = temp1.IndexOf(".");            // 定位第二个“.”
        temp = temp1.Substring(0, lengthData);
        month = int.Parse(temp);                    // 获取month数据
        temp1 = temp1.Substring(lengthData + 1);
        lengthData = temp1.IndexOf(".");            // 定位第三个“.”
        temp = temp1.Substring(0, lengthData);
        day = int.Parse(temp);                      // 获取day数据
        temp1 = temp1.Substring(lengthData + 1);
        lengthData = temp1.IndexOf(":");            // 定位第一个“:”
        temp = temp1.Substring(0, lengthData);
        hour = int.Parse(temp);                     // 获取hour数据
        temp1 = temp1.Substring(lengthData + 1);
        lengthData = temp1.IndexOf(":");             // 定位第二个“:”
        temp = temp1.Substring(0, lengthData);
        min = int.Parse(temp);                       // 获取min数据
        temp = temp1.Substring(lengthData + 1);
        sec = int.Parse(temp);                       // 获取sec数据
        LTSMC.smc_rtc_set_time(_ConnectNo, year, month, day,  hour,  min, sec);
    }


    private void Form1_FormClosed(object sender, FormClosedEventArgs e)
    {
        timer1.Stop();
        LTSMC.nmcs_set_axis_disable(_ConnectNo, axis);    //轴使能关
        LTSMC.smc_board_close(_ConnectNo);                //断开控制器
    }
```

```
        private void timer1_Tick(object sender, EventArgs e)  //定时器时间间隔50毫秒
        {
            double pos = 0, speed = 0;
            ushort errcode = 0;
            ushort axisstatemachine = 0;
            int axisIn, input1, input2, input3;
            LTSMC.nmcs_get_errcode(_ConnectNo, 2, ref errcode);
            if (errcode == 0)       //总线正常
            {
                textBox6.Text = "总线正常";
                LTSMC.smc_read_current_speed_unit(_ConnectNo, axis, ref speed);
                textBox2.Text = speed.ToString();
                LTSMC.smc_get_position_unit(_ConnectNo, axis, ref pos);
                textBox3.Text = pos.ToString();
                LTSMC.nmcs_write_rxpdo_extra(_ConnectNo, 2, 0, 2, Vout);
                // D/A转换，DA的0号通道输出电压信号
                textBox8.Text = (Vout / 10000.0).ToString();
                Vout = Vout + 1000;
                if (Vout > 100000)
                    Vout = 0;
                LTSMC.nmcs_read_txpdo_extra(_ConnectNo, 2, 2, 2, ref Vin); // 读AD的1号通道
                textBox9.Text = (Vin / 10000.0).ToString();

                LTSMC.nmcs_read_txpdo_extra(_ConnectNo, 2, 9, 2, ref Vin); // 读编码器1值
                textBox7.Text = Vin.ToString();

                axisIn = LTSMC.nmcs_get_axis_io_in(_ConnectNo, 4);    // 读取轴4输入口
                input1 = axisIn & 0x80000;
                input2 = axisIn & 0x100000;
                input3 = axisIn & 0x200000;
                if (input1 == 0)
                    checkBox1.Checked = false;
                else
                    checkBox1.Checked = true;
                if (input2 == 0)
                    checkBox2.Checked = false;
                else
                    checkBox2.Checked = true;
                if (input3 == 0)
                    checkBox3.Checked = false;
                else
                    checkBox3.Checked = true;
                LTSMC.nmcs_get_axis_state_machine(_ConnectNo, axis, ref axisstatemachine);
                if (axisstatemachine == 4)
                    textBox1.Text = "电机已使能";
                else
                    textBox1.Text = "未使能，请开使能！";
            }
            else
            {
                LTSMC.smc_emg_stop(_ConnectNo); //电机急停
                textBox6.BackColor = Color.Red;
                textBox6.Text = "总线错误！";
            }
        }
    }
}
```

2. 例程 3 代码

```
using System;
using System.Drawing;
using System.Text;
using System.Threading;
using System.Windows.Forms;
using System.IO;                          // 文件读写的头文件
using Leadshine;

namespace runGcode
{
    public partial class Form1 : Form
    {
        private ushort _ConnectNo = 0;
        ushort axisX = 0, axisY = 1;
        double equivX = 1000.0;
        double equivY = 4000.0 / 3.60;


        public Form1()
        {
            InitializeComponent();
        }


        private void Form1_Load(object sender, EventArgs e)
        {
            short res = LTSMC.smc_board_init(_ConnectNo, 2, "192.168.5.11", 1000000);
                                                          //连接控制器
            if (res != 0)
                MessageBox.Show("连接错误!", "出错");
            else
            {
                LTSMC.smc_set_equiv(_ConnectNo, axisX, equivX);    //设置脉冲当量
                LTSMC.smc_set_equiv(_ConnectNo, axisY, equivY);
                LTSMC.nmcs_set_axis_enable(_ConnectNo, axisX);     //轴使能开
                LTSMC.nmcs_set_axis_enable(_ConnectNo, axisY);
            }
        }


        private void downloadFileButton_Click(object sender, EventArgs e)
        {
            OpenFileDialog f = new OpenFileDialog();
            f.Filter = "G代码|*.nc;*.g";
            DialogResult res = f.ShowDialog();
            if (res == DialogResult.OK)
            {
                string filepath = f.FileName;
                if (string.IsNullOrEmpty(filepath))
                {
                    MessageBox.Show("文件路径不能为空!");
                    return;
                }
                FileStream fs = File.Open(filepath, FileMode.Open, FileAccess.Read,
                                          FileShare.ReadWrite); //打开硬盘上的G代码文件
                StreamReader sr = new StreamReader(fs);
```

```
            string str = sr.ReadToEnd();                    //读G代码
            sr.Close();
            fs.Close();                                     //关闭硬盘上的G代码文件
            byte[] buffer = Encoding.UTF8.GetBytes(str);
            byte[] name = Encoding.UTF8.GetBytes(Path.GetFileName(filepath));
            short result = LTSMC.smc_download_memfile(_ConnectNo, buffer,
                           (uint)buffer.Length, name, 1);  //下载G代码
            if (result == 0)
                MessageBox.Show("G代码文件下载成功!");
        }
    }

    private void startButton_Click(object sender, EventArgs e)
    {
        byte[] FileName = Encoding.UTF8.GetBytes("          ");
        short fileid = 0;
        ushort state = 0;
        uint zz = 0;
        LTSMC.smc_set_position_unit(_ConnectNo, axisX, 0);//位置清零
        LTSMC.smc_set_position_unit(_ConnectNo, axisY, 0);//位置清零
        LTSMC.smc_basic_command(_ConnectNo, Encoding.UTF8.GetBytes(
        "GCode_LinerNO=0"),FileName, zz);                   //坐标系为0
        LTSMC.smc_basic_command(_ConnectNo, Encoding.UTF8.GetBytes(
        "GCode_LocateSpeed(0) = 40"), FileName, zz);        //0号轴G代码定位速度
        LTSMC.smc_basic_command(_ConnectNo, Encoding.UTF8.GetBytes(
        "GCode_LocateSpeed(1) = 40"), FileName, zz);        //1号轴G代码定位速度
        LTSMC.smc_basic_command(_ConnectNo, Encoding.UTF8.GetBytes(
        "GCode_LocateTAcc(0) = 0.1"), FileName, zz);        //0号轴G代码定位加速减速时间
        LTSMC.smc_basic_command(_ConnectNo, Encoding.UTF8.GetBytes(
        "GCode_LocateTAcc(1) = 0.1 "), FileName, zz);       //1号轴G代码定位加速减速时间
        LTSMC.smc_basic_command(_ConnectNo, Encoding.UTF8.GetBytes(
        "GCode_PathSpeed(0) = 30"), FileName, zz);          //G代码插补速度
        LTSMC.smc_basic_command(_ConnectNo, Encoding.UTF8.GetBytes(
        "GCode_PathTAcc(0) = 0.1"), FileName, zz);          //G代码插补加减速时间
        LTSMC.smc_gcode_get_first_file(_ConnectNo, FileName, ref zz);
        LTSMC.smc_gcode_set_current_file(_ConnectNo, FileName);//将第1个G代码设为当前文件
        timer1.Start();
        LTSMC.smc_gcode_start(_ConnectNo); //运行当前文件的G代码
    }

    private void Form1_FormClosed(object sender, FormClosedEventArgs e)
    {
        LTSMC.nmcs_set_axis_disable(_ConnectNo, axisX);     //轴使能关
        LTSMC.nmcs_set_axis_disable(_ConnectNo, axisY);
        Thread.Sleep(100);                                  //线程休眠100毫秒
        timer1.Stop();
        LTSMC.smc_board_close(_ConnectNo);                  //断开控制器
    }

    private void timer1_Tick(object sender, EventArgs e)
    {
        ushort errcode = 0;
        double positionX = 0, positionY = 0;
        int x, y;
        LTSMC.nmcs_get_errcode(_ConnectNo, 2, ref errcode);
        if (errcode == 0)                   //总线正常
            textBox1.Text = "总线正常。";
        else
```

```
                textBox1.Text = "总线错误！";

            Graphics g = pictureBox1.CreateGraphics(); // 创建Graphics对象
            g.PageUnit = GraphicsUnit.Pixel;           // 以像素为单位
            g.PageScale = 1.0f;                  // 像素与坐标单位的比例，即1个单位为1个像素。
            Pen RedPen = new Pen(Color.Red, 1);
            LTSMC.smc_get_position_unit(_ConnectNo, axisY, ref positionX);// 获取当前位置
            LTSMC.smc_get_position_unit(_ConnectNo, axisX, ref positionY);
            textBox2.Text = positionX.ToString();
            textBox3.Text = positionY.ToString();
            x = (int)(positionX / 0.3);
            y = (int)(positionY / 0.3);
            g.DrawEllipse(RedPen, x, y, 1, 1);  // 画点
        }


        private void button2_Click(object sender, EventArgs e) //清空G代码
        {
            byte[] FileName = Encoding.UTF8.GetBytes("           ");
            short zz = 0;
            LTSMC.smc_gcode_clear_file(_ConnectNo);
            LTSMC.smc_gcode_get_current_file(_ConnectNo, FileName, ref zz);
            LTSMC.smc_gcode_delete_file(_ConnectNo, FileName);
            MessageBox.Show("G代码文件已经清空!");
        }


        private void button3_Click(object sender, EventArgs e) //急停
        {
            LTSMC.smc_emg_stop(_ConnectNo);
        }


    }
}
```

3. 例程 4 代码

```
using System;
using System.Drawing;
using System.Threading;
using System.Windows.Forms;
using Leadshine;

namespace PVTcommand
{
    public partial class Form1 : Form
    {
        const double PI = 3.1415926f;
        ushort ConnectNo = 0;
        ushort axisX = 0, axisY = 1, axisZ = 4;
        double equivX = 100.0;              // 一圈的距离为360
        double equivY = 400.0 / 3.60;       // 一圈的距离为360
        double equivZ = 8388608.0 / 360.0;  // 一圈的距离为360
        double t = 0;
        double alpha = PI * 30 / 180.0;

        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            double[] dtData = { 0, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.2, 0.1, 0.1, 0.1,
             0.1, 0.1, 0.1, 0.1, 0.1, 0.3 }; // 通过定义每条线段运行时间，间接定义轨迹速度。
            double[] tData = new double[19];
            double[] PxData = new double[19];
            double[] PyData = new double[19];
            double[] PzData = new double[19];
            double[] distanceXYZ = { 0, 0, 0 };
            ushort[] AxisList = { 0, 1, 4 };
            double Px0, Py0, Pz0;
            double a = 100.0, b = 70.0, theta = PI / 6.0;
            double a0 = 125.0, b0 = 50.0;
            int i, x, y;
            tData[0] = 0;
            PxData[0] = 0;
            PyData[0] = 0;
            PzData[0] = 0;
            Px0 = (a0 + a) * Math.Cos(theta);
            Py0 = (a0 + a) * Math.Sin(theta);
            Pz0 = b0;
            t = 0;
            for (i = 1; i <= 18; i++)
            {
                tData[i] = tData[i - 1] + dtData[i];                    //计算各节点的时间
                                                                        //计算各节点的位置
                PxData[i] = (a0 + a * Math.Cos(i * PI / 18.0)) * Math.Cos(theta) - Px0;
                PyData[i] = (a0 + a * Math.Cos(i * PI / 18.0)) * Math.Sin(theta) - Py0;
                PzData[i] = b0 + b * Math.Sin(i * PI / 18.0) - Pz0;
            }
            LTSMC.smc_set_position_unit(ConnectNo, axisX, 0);    //设起点位置
            LTSMC.smc_set_position_unit(ConnectNo, axisY, 0);
```

```
        LTSMC.smc_set_position_unit(ConnectNo, axisZ, 0);
        DrawXYZcoordinate();                                    //画坐标轴
        DrawVTcoordinate();
        LTSMC.smc_pvts_table_unit(ConnectNo, axisX, 19, tData, PxData, 0, 0);
        LTSMC.smc_pvts_table_unit(ConnectNo, axisY, 19, tData, PyData, 0, 0);
        LTSMC.smc_pvts_table_unit(ConnectNo, axisZ, 19, tData, PzData, 50, -50);
        timer1.Start();
        LTSMC.smc_set_vector_profile_unit(ConnectNo, 0, 10, 100, 0.1, 0.1, 50);
        //设置起始速度、运行速度、停止速度、加速时间、减速时间
        distanceXYZ[0] = Px0;
        distanceXYZ[1] = Py0;
        distanceXYZ[2] = 0;
        LTSMC.smc_line_unit(ConnectNo, 0, 3, AxisList, distanceXYZ, 0); //直线插补运动
        while (LTSMC.smc_check_done(ConnectNo, axisY) == 0)
            Application.DoEvents(); // 处理当前消息队列中的消息。
        LTSMC.smc_set_vector_profile_unit(ConnectNo, 0, 50, 50, 0.1, 0.1, 50);
        distanceXYZ[0] = 0;
        distanceXYZ[1] = 0;
        distanceXYZ[2] = b0;
        LTSMC.smc_line_unit(ConnectNo, 0, 3, AxisList, distanceXYZ, 0);//直线插补运动
        while (LTSMC.smc_check_done(ConnectNo, axisY) == 0)
            Application.DoEvents();
        LTSMC.smc_pvt_move(ConnectNo, 3, AxisList);                    //执行PVT运动
        while (LTSMC.smc_check_done(ConnectNo, axisZ) == 0 ||
               LTSMC.smc_check_done(ConnectNo, axisY) == 0 ||
               LTSMC.smc_check_done(ConnectNo, axisZ) == 0)
            Application.DoEvents();
        distanceXYZ[0] = 0;
        distanceXYZ[1] = 0;
        distanceXYZ[2] = -b0;
        LTSMC.smc_set_vector_profile_unit(ConnectNo, 0, 50, 50, 0.1, 0.1, 10);
        LTSMC.smc_line_unit(ConnectNo, 0, 3, AxisList, distanceXYZ, 0);
        while (LTSMC.smc_check_done(ConnectNo, axisZ) == 0)
            Application.DoEvents();
        timer1.Stop();
        Graphics g = pictureBox1.CreateGraphics(); // 创Graphics对象
        g.PageUnit = GraphicsUnit.Pixel;           // 以像素为单位
        g.PageScale = 1.0f;                    // 像素与坐标单位的比例，即1个单位为1个像素。
        g.TranslateTransform(0, 200);          // y轴下移400 ，方向仍朝下
        Pen RedPen = new Pen(Color.Red, 4);
        for (i = 0; i <= 18; i++)
        {
            x = (int)(Px0 + PxData[i] + (Py0 + PyData[i]) * Math.Cos(alpha));
            y = (int)(Pz0 + PzData[i] + (Py0 + PyData[i]) * Math.Sin(alpha));
            textBox1.Text = x.ToString();
            textBox2.Text = y.ToString();
            g.DrawEllipse(RedPen, x, -1 * y, 1, 1); //画理论位置
        }
    }


    private void Form1_Load(object sender, EventArgs e)
    {
        short res = LTSMC.smc_board_init(ConnectNo, 2, "192.168.5.11", 1000000);
                                                                //连接控制器
        if (res != 0)
            MessageBox.Show("连接错误!", "出错");
        else
```

```
        {
            LTSMC.smc_set_equiv(ConnectNo, axisX, equivX);    //设置脉冲当量
            LTSMC.smc_set_equiv(ConnectNo, axisY, equivY);
            LTSMC.smc_set_equiv(ConnectNo, axisZ, equivZ);
            LTSMC.nmcs_set_axis_enable(ConnectNo, axisX);    //轴使能开
            LTSMC.nmcs_set_axis_enable(ConnectNo, axisY);
            LTSMC.nmcs_set_axis_enable(ConnectNo, axisZ);
        }
    }

    private void Form1_FormClosed(object sender, FormClosedEventArgs e)
    {
        LTSMC.nmcs_set_axis_disable(ConnectNo, axisX);    //轴使能关
        LTSMC.nmcs_set_axis_disable(ConnectNo, axisY);
        LTSMC.nmcs_set_axis_disable(ConnectNo, axisZ);
        Thread.Sleep(100);                                // 线程休眠100毫秒
        timer1.Stop();
        LTSMC.smc_board_close(ConnectNo);                 // 断开控制器
    }

    private void timer1_Tick(object sender, EventArgs e)
    {
        double positionX = 0, positionY = 0, positionZ = 0;
        int x, y, z;
        double speedX = 0, speedY = 0, speedZ = 0;
        Graphics g = pictureBox1.CreateGraphics();
        g.PageUnit = GraphicsUnit.Pixel;
        g.PageScale = 1.0f;
        g.TranslateTransform(0, 200);
        Pen RedPen = new Pen(Color.Red, 2);
        Pen BluePen = new Pen(Color.Blue, 1);
        LTSMC.smc_get_position_unit(ConnectNo, axisX, ref positionX);
        LTSMC.smc_get_position_unit(ConnectNo, axisY, ref positionY);
        LTSMC.smc_get_position_unit(ConnectNo, axisZ, ref positionZ);
        x = (int)((positionX + positionY * Math.Cos(alpha)) * 1);
        y = (int)((positionZ + positionY * Math.Sin(alpha)) * 1);
        g.DrawEllipse(BluePen, x, -1 * y, 1, 1); //画位置的点
        Graphics g2 = pictureBox2.CreateGraphics();
        g2.PageUnit = GraphicsUnit.Pixel;
        g2.PageScale = 1.0f;
        g2.TranslateTransform(0, 200);
        Pen RedPen2 = new Pen(Color.Red, 2);
        LTSMC.smc_read_current_speed_unit(ConnectNo, axisX, ref speedX);
        LTSMC.smc_read_current_speed_unit(ConnectNo, axisY, ref speedY);
        LTSMC.smc_read_current_speed_unit(ConnectNo, axisZ, ref speedZ);
        z = (int)(Math.Sqrt(speedX * speedX + speedY * speedY + speedZ * speedZ));
        textBox1.Text = z.ToString();
        g2.DrawEllipse(RedPen2, (int)t, -1 * z, 1, 1); //画速度的点
        t = t + 0.5;
    }


    private void DrawXYZcoordinate()
    {
        int i, X1, Y1, X2, Y2;
        Graphics g = pictureBox1.CreateGraphics();
        g.PageUnit = GraphicsUnit.Pixel;
        g.PageScale = 1.0f;
```

```
            g.TranslateTransform(0, 200);
            g.Clear(Color.White);                       // 画布清零
            Pen YellowPen = new Pen(Color.Yellow);
            Pen BlackPen = new Pen(Color.Black, 2);
            for (i = 0; i <= 300; i = i + 50)
            {
                X1 = (int)(i * Math.Cos(alpha));
                Y1 = (int)(i * Math.Sin(alpha));
                X2 = 300 + (int)(i * Math.Cos(alpha));
                Y2 = (int)(i * Math.Sin(alpha));
                g.DrawLine(YellowPen, X1, -1 * Y1, X2, -1 * Y2);//画XY平面水平线
                X1 = i;
                Y1 = 0;
                X2 = i + (int)(300 * Math.Cos(alpha));
                Y2 = (int)(300 * Math.Sin(alpha));
                g.DrawLine(YellowPen, X1, -1 * Y1, X2, -1 * Y2);//画XY平面垂直线
                X1 = 0;
                Y1 = i;
                X2 = (int)(300 * Math.Cos(alpha));
                Y2 = i + (int)(300 * Math.Sin(alpha));
                g.DrawLine(YellowPen, X1, -1 * Y1, X2, -1 * Y2);//画YZ平面水平线
                X1 = (int)(i * Math.Cos(alpha));
                Y1 = (int)(i * Math.Sin(alpha));
                X2 = (int)(i * Math.Cos(alpha));
                Y2 = 300 + (int)(i * Math.Sin(alpha));
                g.DrawLine(YellowPen, X1, -1 * Y1, X2, -1 * Y2);//画YZ平面垂直线
            }
            g.DrawLine(BlackPen, 0, 0, 300, 0);      // 画X轴
            g.DrawLine(BlackPen, 287, -4, 300, 0);   // 画X轴
            X2 = (int)(330 * Math.Cos(alpha));
            Y2 = (int)(330 * Math.Sin(alpha));
            g.DrawLine(BlackPen, 0, 0, X2, -1 * Y2); // 画Y轴
            g.DrawLine(BlackPen, X2, -1 * Y2, X2 - 10, -1 * (Y2 - 10));//画Y轴
            g.DrawLine(BlackPen, 0, 0, 0, -190);      // 画Z轴
            g.DrawLine(BlackPen, 0, -190, 3, -177);   // 画Z轴
        }


        private void DrawVTcoordinate()
        {
            int i;
            Graphics g2 = pictureBox2.CreateGraphics();
            g2.PageUnit = GraphicsUnit.Pixel;
            g2.PageScale = 1.0f;
            g2.TranslateTransform(0, 200);
            g2.Clear(Color.White);
            Pen YellowPen = new Pen(Color.Yellow);
            Pen BlackPen = new Pen(Color.Black, 2);
            for (i = 0; i <= 400; i = i + 50)
            {
                g2.DrawLine(YellowPen, 0, -i, 400, -i);//画XY平面水平线
                g2.DrawLine(YellowPen, i, 0, i, -400); //画XY平面垂直线
            }
            g2.DrawLine(BlackPen, 0, 0, 400, 0);      // 画X轴
            g2.DrawLine(BlackPen, 0, 0, 0, -400);     // 画Y轴
        }
    }
}
```