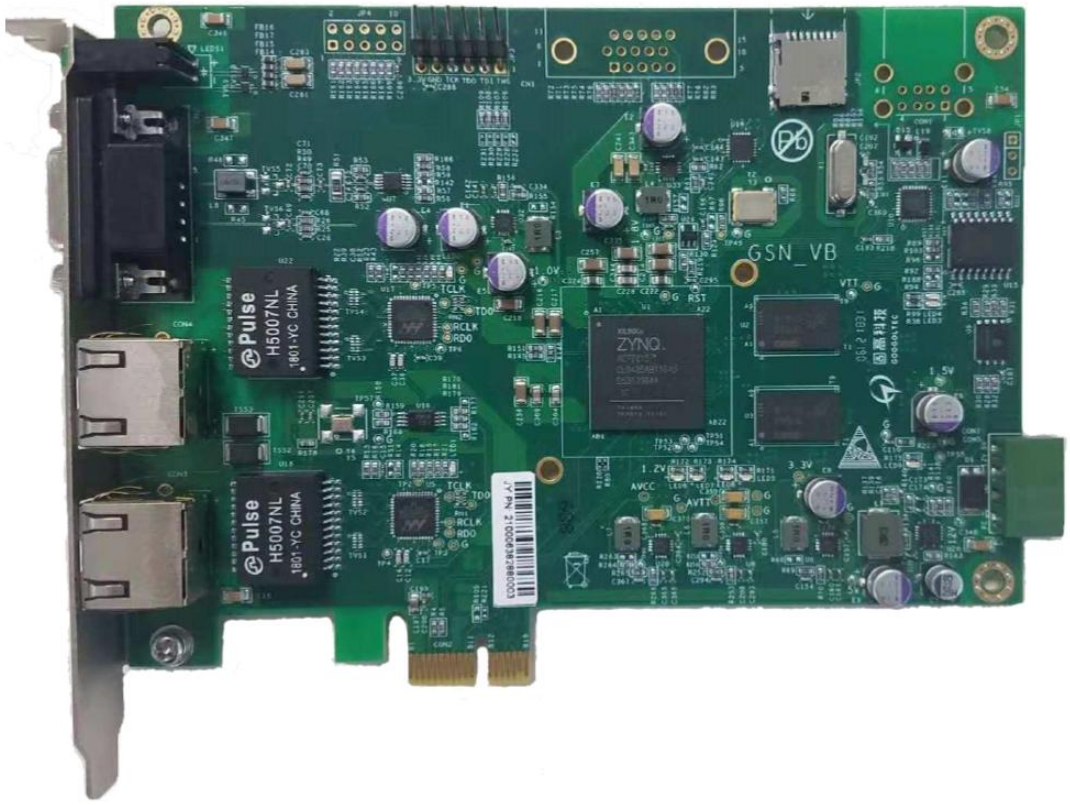
固高科技
GOOGOL TECHNOLOGY



基于C++语言的GEN控制卡

初级入门手册

（点位运动示例）

目录

一、如何识别控制卡端子板型号

方法1、包装盒,如图1

图 1

方法 2、控制卡标签，如图 2

图 2

二、安装控制卡和驱动

1、打开主机后盖，将卡插到PCI-E插槽上，然后右击我的电脑，点击属性，通过查看系统类型查看电脑的操作系统是32为还是64位；打开设备管理器，会看到PCI设备上有个感叹号，然后右击，弹出更新驱动程序软件，选择对应的驱动程序如图4，如图5，如图6，如图7，如图8，如图9，如图10所示

图4

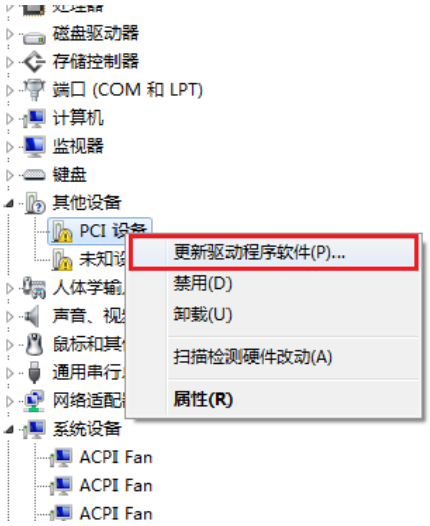

磁盘驱动器
存储控制器
端口 (COM 和 LPT)
计算机
监视器
键盘
其他设备
PCI 设备
未知设
更新驱动程序软件(P)...
禁用(D)
卸载(U)
扫描检测硬件改动(A)
属性(R)
人体学输
声音、视
鼠标和其
通用串行
网络适配
系统设备
ACPI Fan
ACPI Fan
ACPI Fan

图5

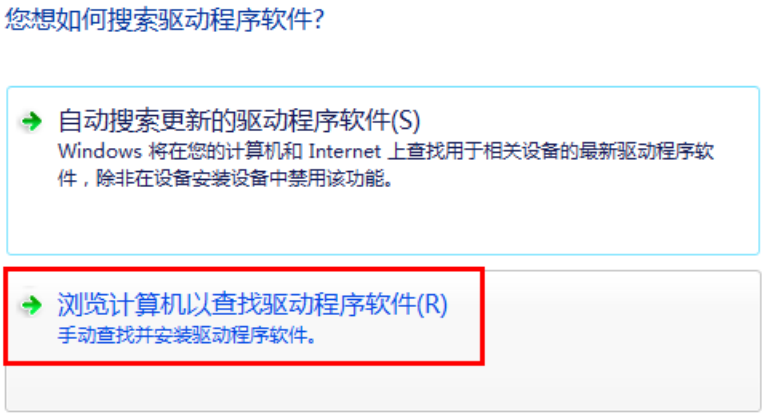

您想如何搜索驱动程序软件?
自动搜索更新的驱动程序软件(S)
Windows 将在您的计算机和 Internet 上查找用于相关设备的最新驱动程序软件，除非在设备安装设备中禁用该功能。
浏览计算机以查找驱动程序软件(R)
手动查找并安装驱动程序软件。

图 6

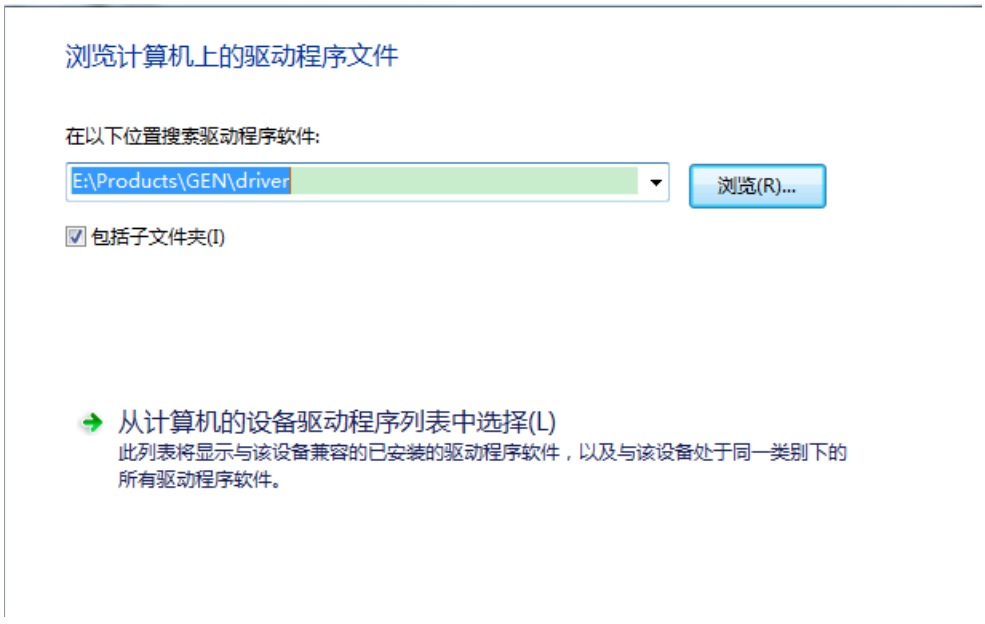

浏览计算机上的驱动程序文件
在以下位置搜索驱动程序软件:
E:\Products\GEN\driver
浏览(R)...
包括子文件夹(I)
从计算机的设备驱动程序列表中选择(L)
此列表将显示与该设备兼容的已安装的驱动程序软件，以及与该设备处于同一类别下的所有驱动程序软件。

图 7

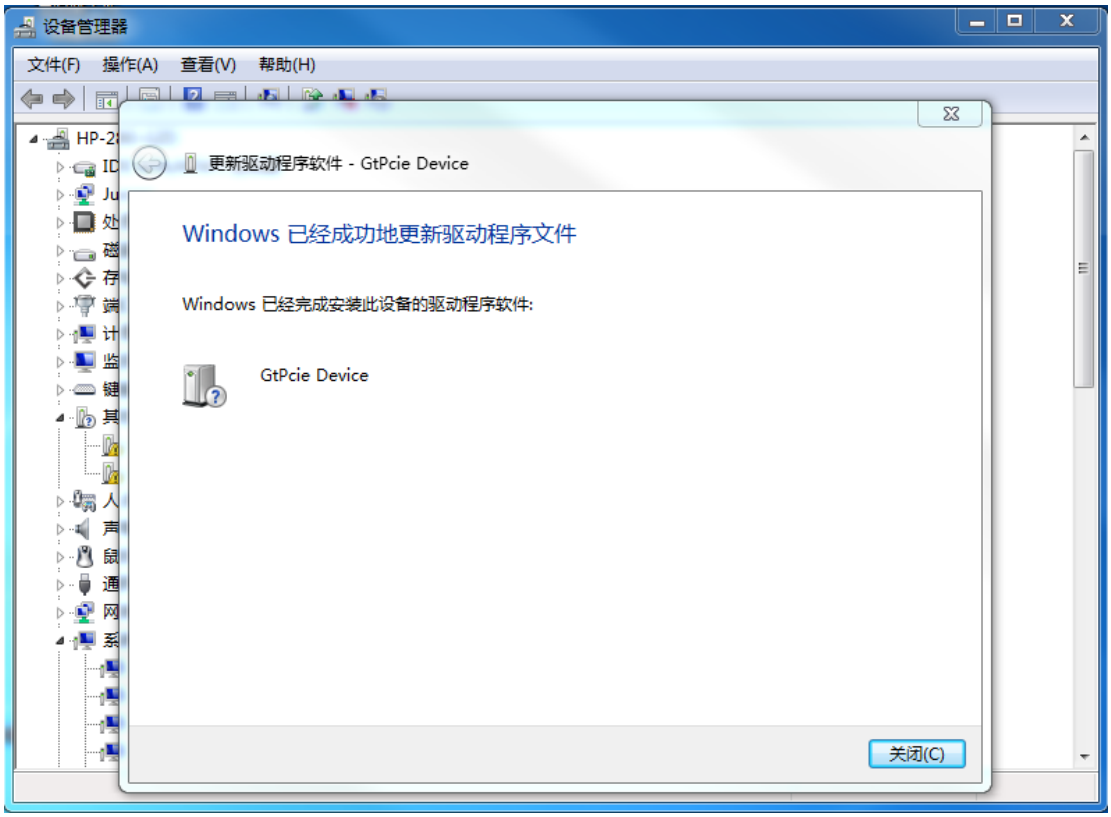

设备管理器
文件(F) 操作(A) 查看(V) 帮助(H)
更新驱动程序软件 - GtPcie Device
Windows 已经成功地更新驱动程序文件
Windows 已经完成安装此设备的驱动程序软件:
GtPcie Device
关闭(C)

图 8

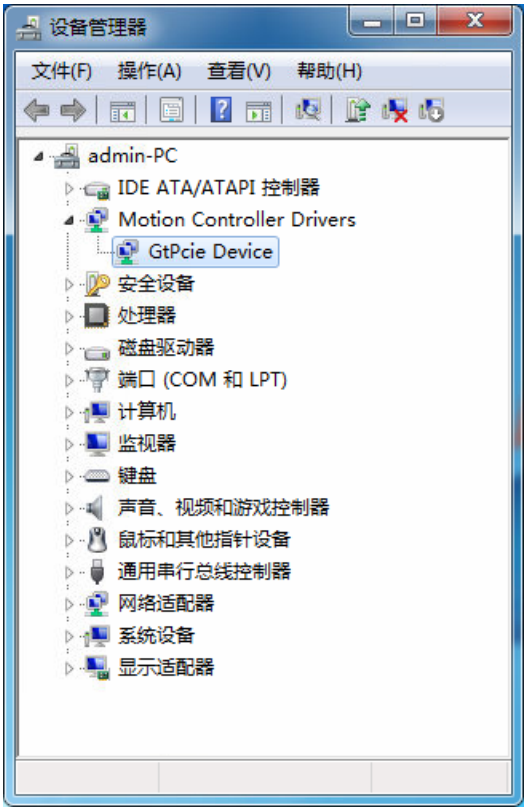

设备管理器
文件(F) 操作(A) 查看(V) 帮助(H)
admin-PC
IDE ATA/ATAPI 控制器
Motion Controller Drivers
GtPcie Device
安全设备
处理器
磁盘驱动器
端口 (COM 和 LPT)
计算机
监视器
键盘
声音、视频和游戏控制器
鼠标和其他指针设备
通用串行总线控制器
网络适配器
系统设备
显示适配器

图9

注：如果图 8 步骤之后显示图 10 所示黄色叹号，则需要安装相应的系统补丁。系统补丁的安装方法较简单，在驱动路径下找到正确的补丁，直接双击运行“msu”文件即可。

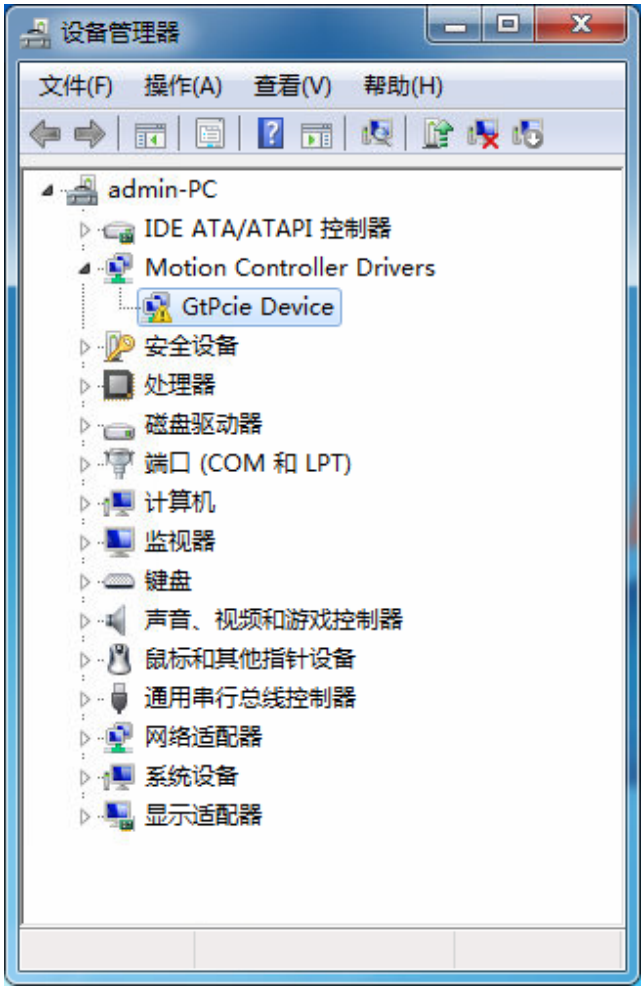


图10

三、使用EthercatConfig软件，导入XML文件生成相应的硬件描述信息文件Sii，然后配置EtherCAT从站。（xml -> sii -> eni）

1. **安装设备文件（xml -> sii）**，菜单栏“Options”，选择“Import Xml...”，打开Slave Install Toolkit，选择xml文件，执行install，即可转换出对应的 sii 文件，这里我们使用松下 A5 驱动器的 XML 文件作为例子， 如图 11

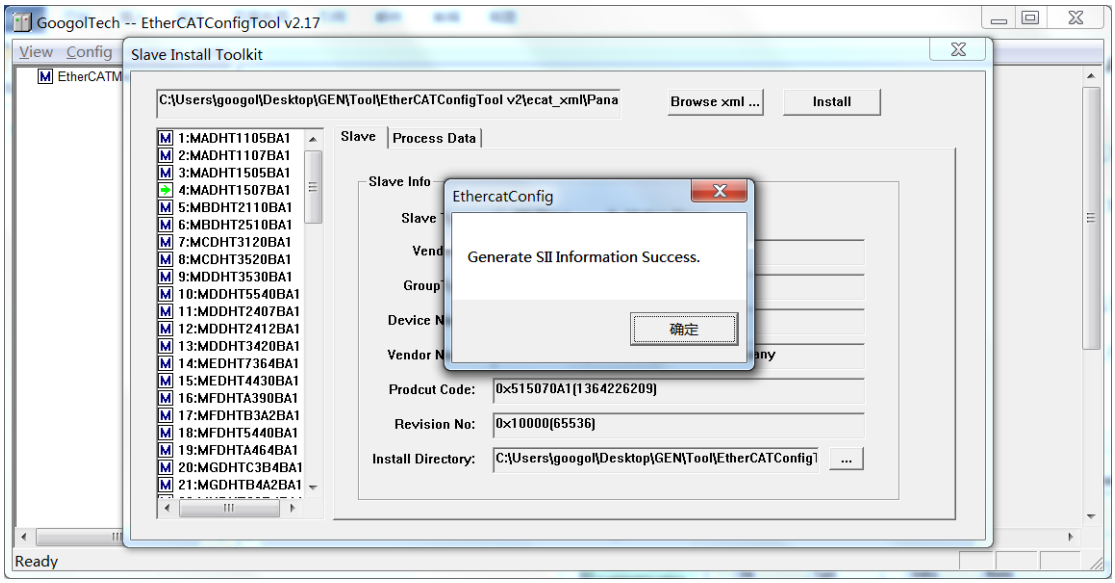


图11

接着可以再 Install Directory（安装目录）中找到生成的 sii 文件即 MADHT1507BA1.sii。

2. 配置 EtherCAT 总线（sii -> eni）左侧右击设备列表取添加从站（此处我们仅添加一个轴），在右侧 Slave 的 Slave Type 选择 1：Motion Slave, 并在右侧“PDO”选项卡中配置 PDO 映射，随后在菜单栏“Config”中选择“Save ECAT Config”，将上一步生成的 MADHT1507BA1.sii 转换成 eni 文件，如图 12

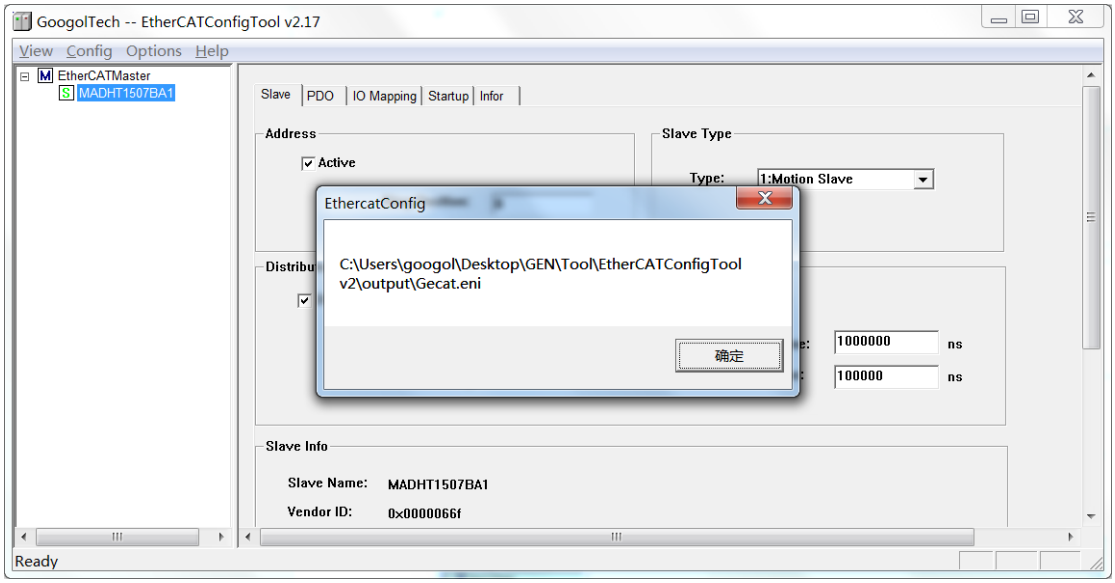


图12

3. 部署EtherCAT总线配置文件，最后，将“...\EtherCATConfigTool v*\output\Gecat.eni”文件拷贝到目标可执行程序目录中

例如，MotionStudio软件目录，如图13

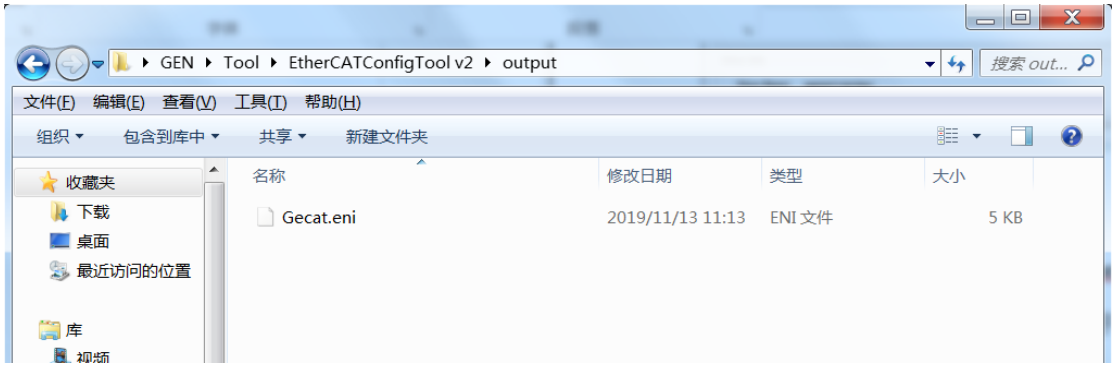


图13

四、使用 MotionStudio 软件初步测试

Motionstudio 电气调试界面，EtherCAT 配置过程提示信息，

如图 14，图 15

图14

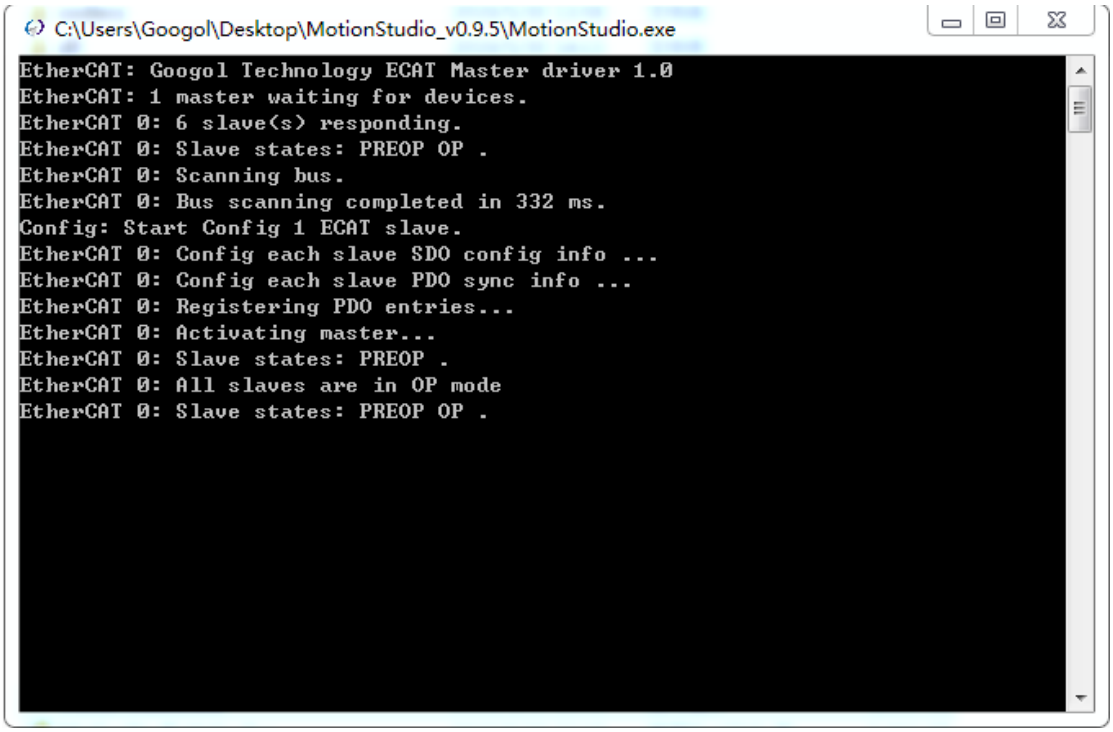


图15

五、驱动安装完以后，开始编程。

1、首先打开 VS2008，如图 16

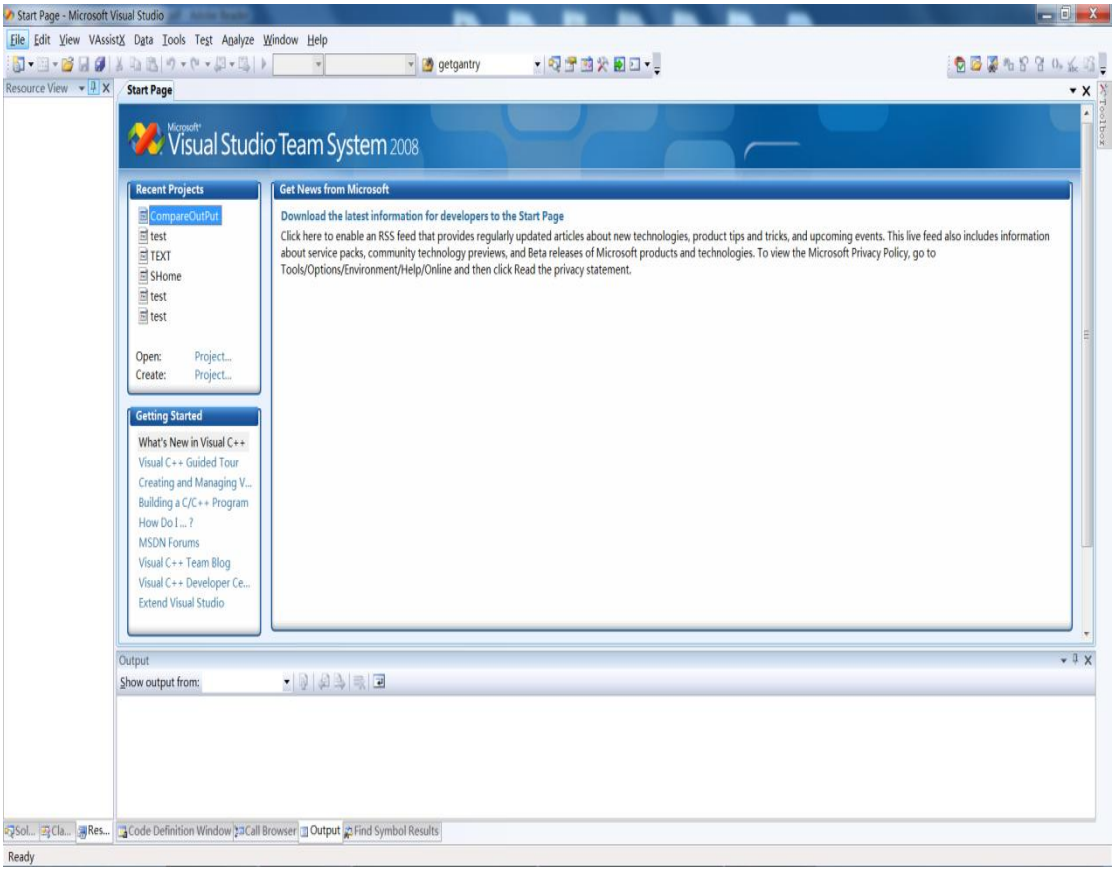


图 16

2、然后点击新建项目，选择 Visual C++，然后选择创建 MFC Application(MFC 应用程序)，将名称改为点位运动，点击 Next，然后选择 Dialog base(基于对话框)，最后点击 Finish 如图 16，如图 17

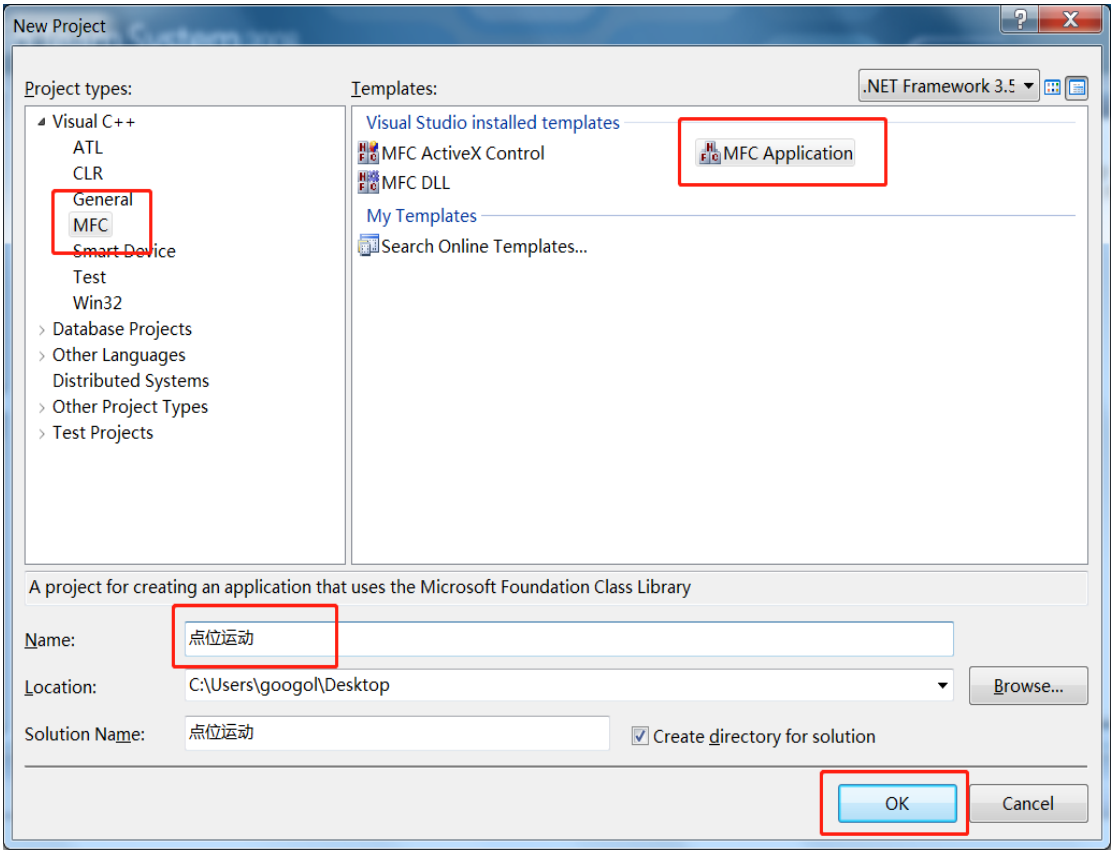


图 17

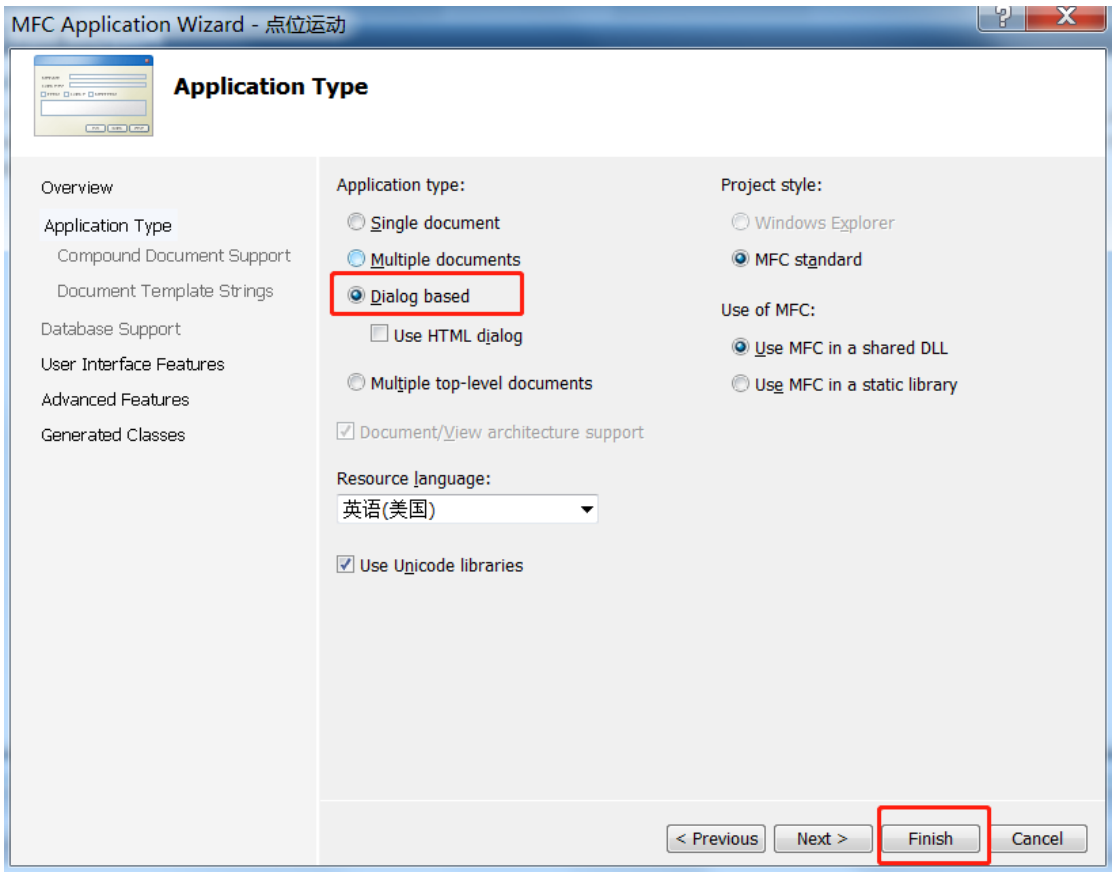

MFC Application Wizard - 点位运动
Application Type
Overview
Application Type
Compound Document Support
Document Template Strings
Database Support
User Interface Features
Advanced Features
Generated Classes
Application type:
Single document
Multiple documents
Dialog based
Use HTML dialog
Multiple top-level documents
Document/View architecture support
Resource language:
英语(美国)
Use Unicode libraries
Project style:
Windows Explorer
MFC standard
Use of MFC:
Use MFC in a shared DLL
Use MFC in a static library
< Previous
Next >
Finish
Cancel

图 18

3、将产品配套光盘Library\Win32\C.C++文件夹中的动态链接库(例如：gts.dll)、头文件(例如：gts.h)、lib文件(gts.lib)和步骤三生成的Gecat.eni复制到工程文件夹中，如图18

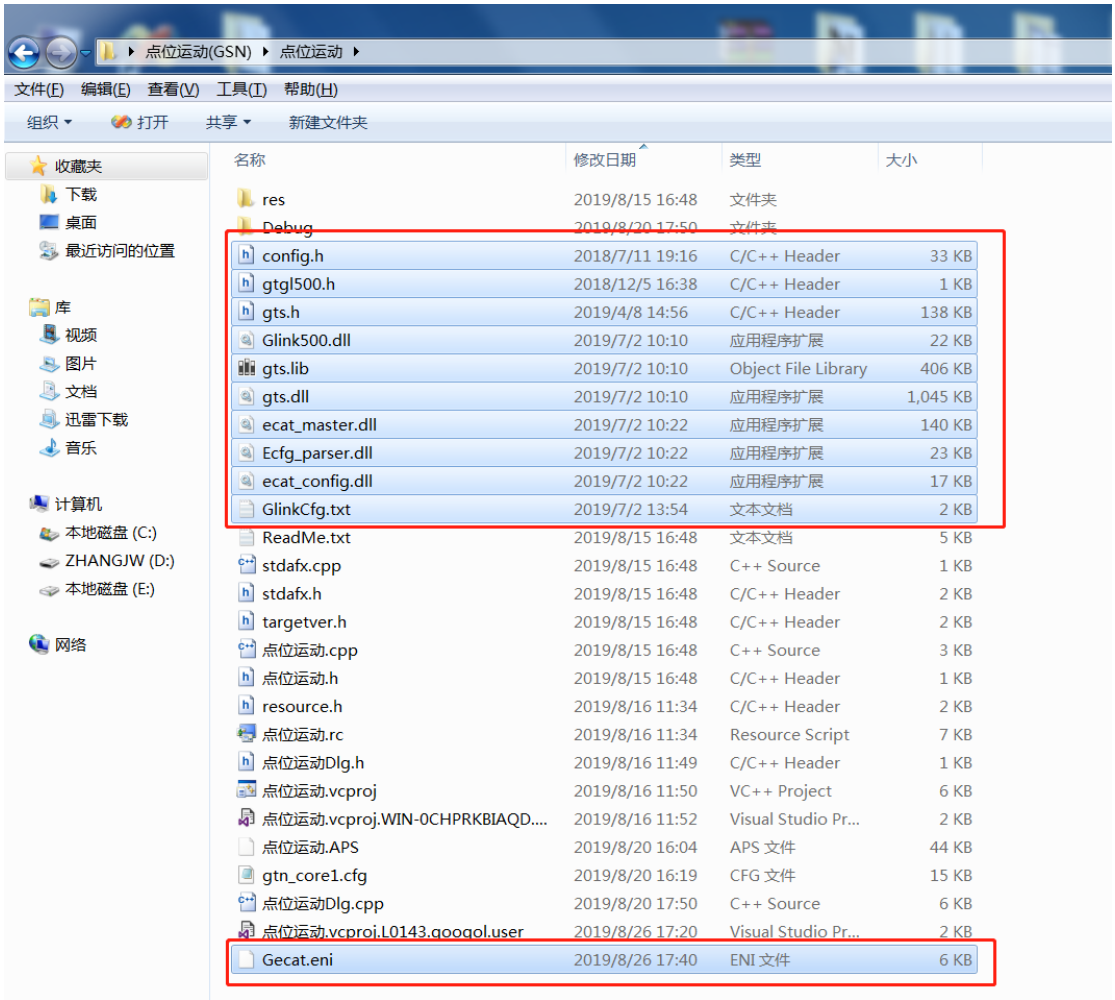


图 19

4、打开MotionStudio，选择工具→控制器配置，正、负限位选择none，点击写入到控制器，查看轴状态，确认驱动、限位都无报警后，点击运动，如图20，21

图 20

图 21

5、然后点击写入到控制器，再点击写入到文件，文件名默认为 gtn_core1，保存到桌面，如图 22

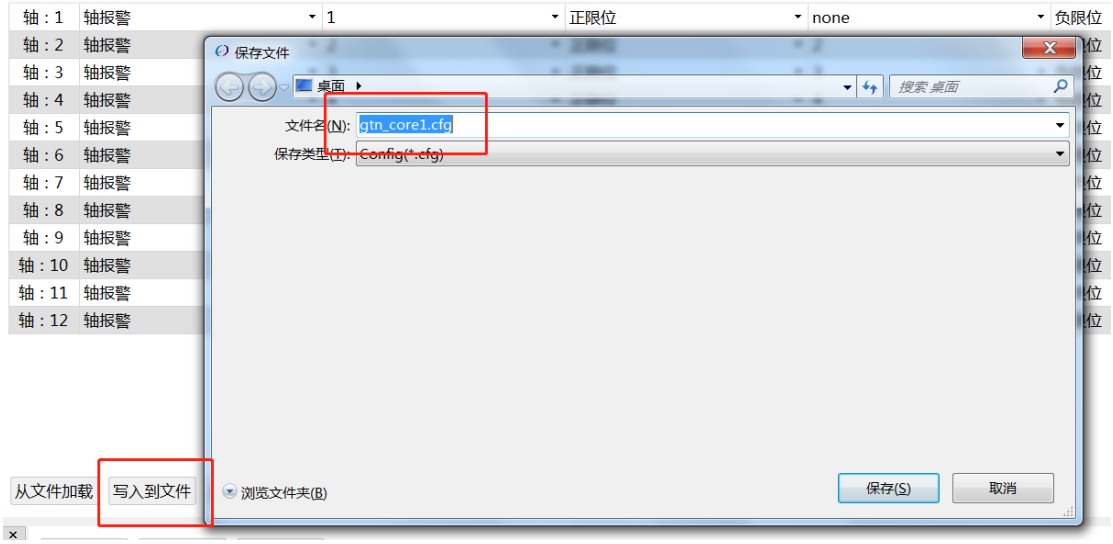


图 22

6、把桌面生成的 gtn_core1 文件复制到（C:\Users\googol\Desktop\点位运动\点位运动）文件下，如图 23

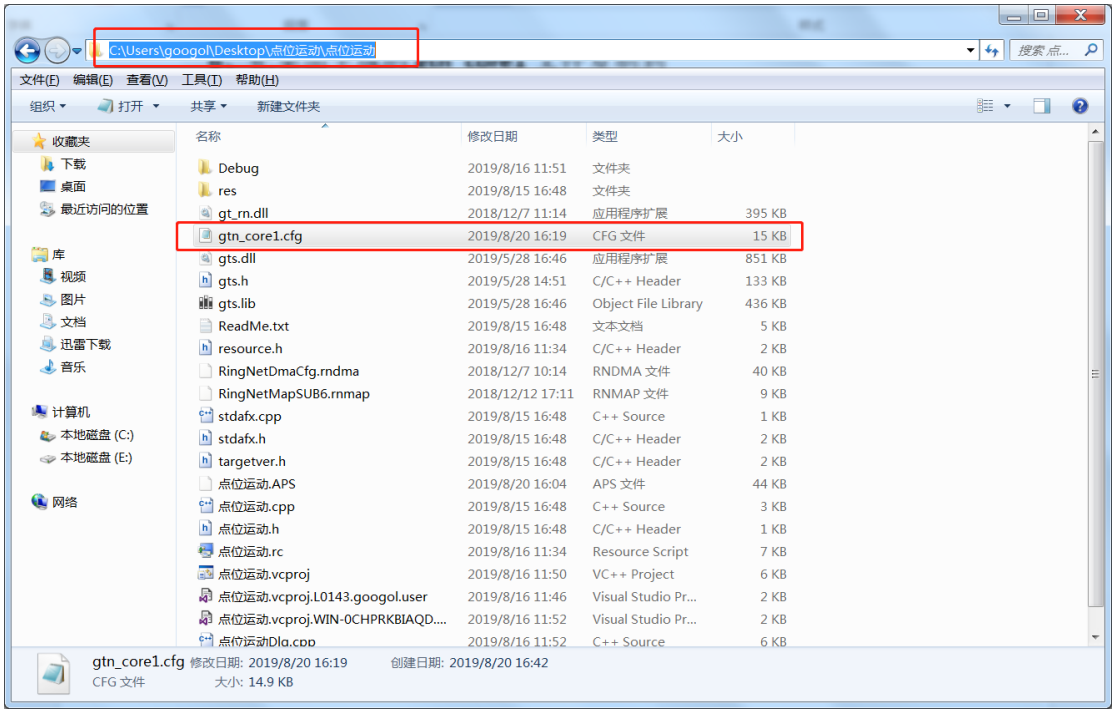


图 23

7、打开刚才创建的vs2008程序，在左侧的解决方案管理器中右击点位运动→属性→配置属性→链接器→输入→附加依赖项栏中输入lib文件名，例如gts.lib，如图24，如图25

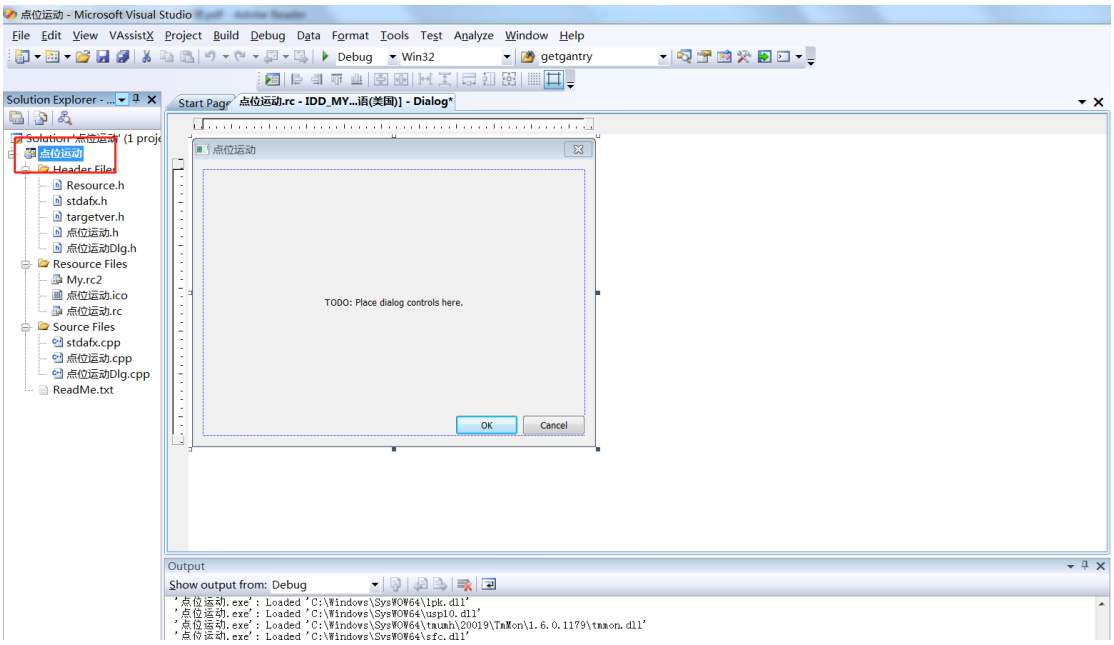


图 24

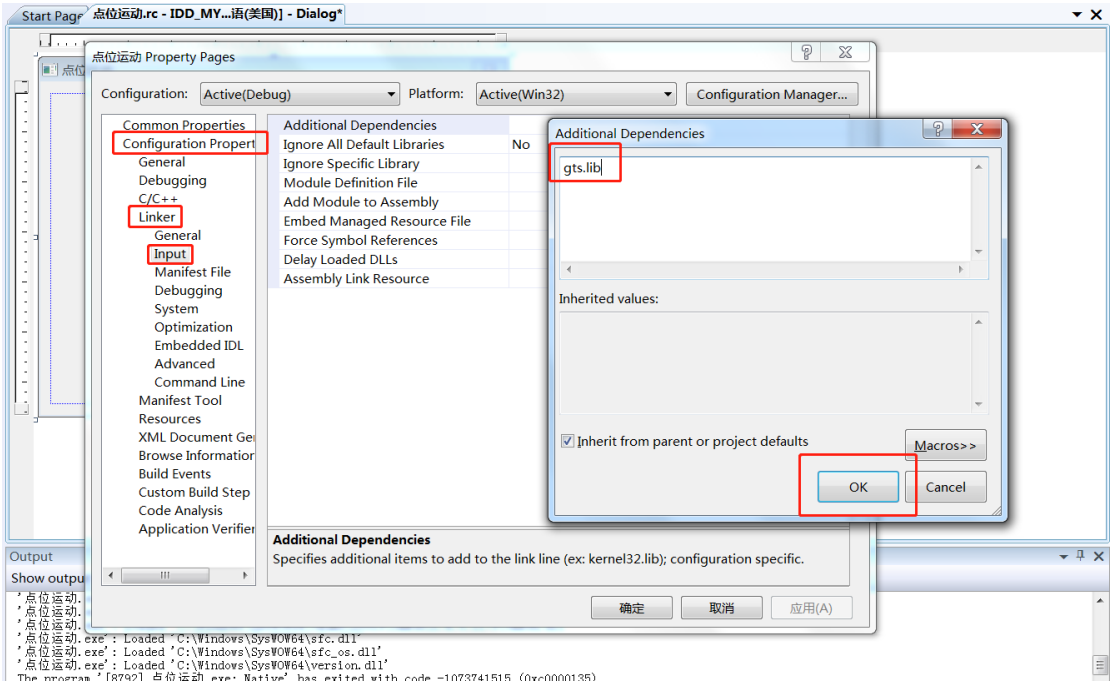


图 25

8、在应用程序文件中加入函数库头文件的声明，例如：#include “gts.h”； 至此，用户就可以在Visual C++中调用函数库中的任何函数，开始编写应用程序。 对于步骤 7，还有一种比较简便的方法，那就是在应用程序文件里面添加包含链接文件的声明，例如：#pragma comment (lib,"gts.lib") ，如图26

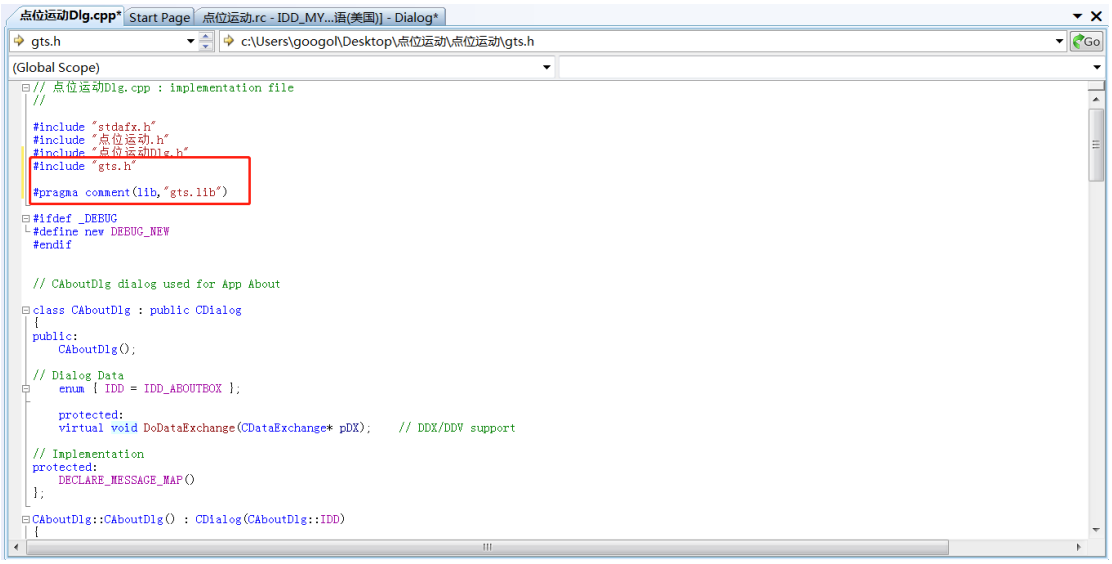


图 26

9、下一步开始编程，先将对话框上多余的控件删除，在右侧工具箱中选中 button 控件，然后添加到点位运动窗体中，如图 27

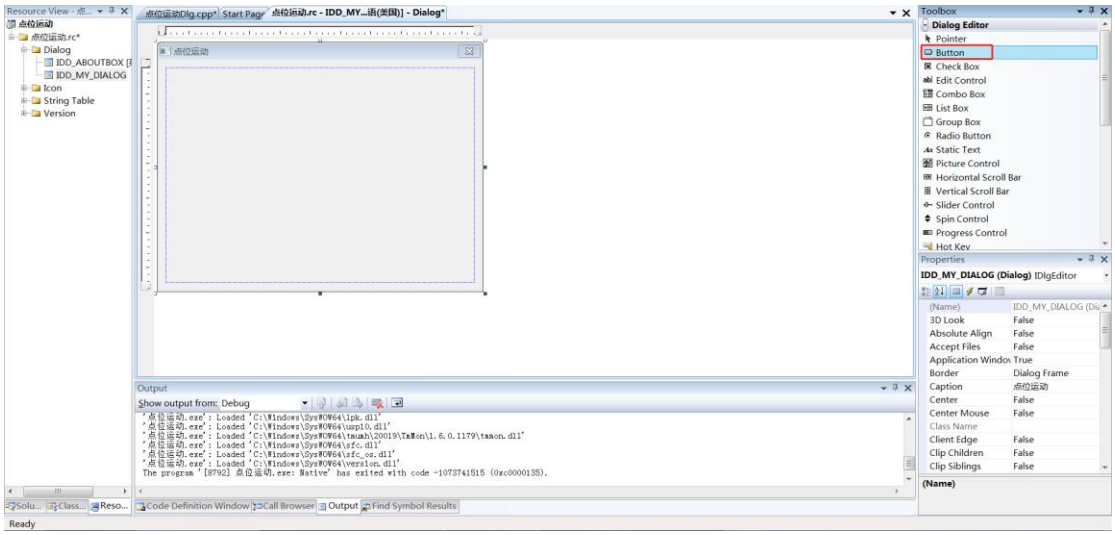


图 27

10、然后选中 button1，在右下角的属性窗口中，选择 text 属性，将其改为“初始化”，如图 28

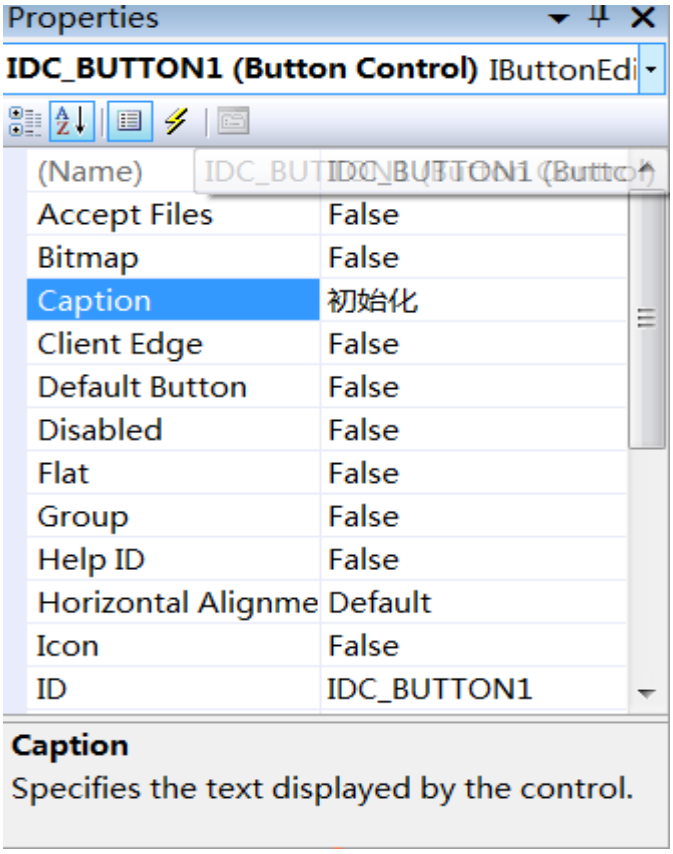


图 28

11、然后再在工具箱中选择 Static Text，Edit Control 控件放到窗体中，并将其控件 Caption 属性分别改成图中所示名称，如图 29

图29

接着，为点位距离和点位速度的编辑框IDC _EDIT1和IDC _EDIT2添加变量。

（1）在编辑框上点右键，在右键菜单中选择“Add Variable”。弹出添加成员变量的向导对话框。

（2）我们想为其添加值变量而不是控件变量，所以对话框中“Category”下的组合框中选择 Value。

（3）“Variable type”下的组合框此时默认选中的是“CString”，CString 是字符串类，显然不能进行数据交换。我们可以选择 double、float、int 等。这里我们选择 double，即编辑框关联一个 double 类型的变量。

（4）在“Variable name”中写入自定义的变量名。我为其取名 m_editPos 和 m_editVel。

12、下一步开始编程，首先，声明全局变量，如图30

图30

13、在窗体的属性视图中添加消息处理函数，这里我们双击红框，加入定时器，以便获取规划位置、规划速度等变量的实时变化，如图31

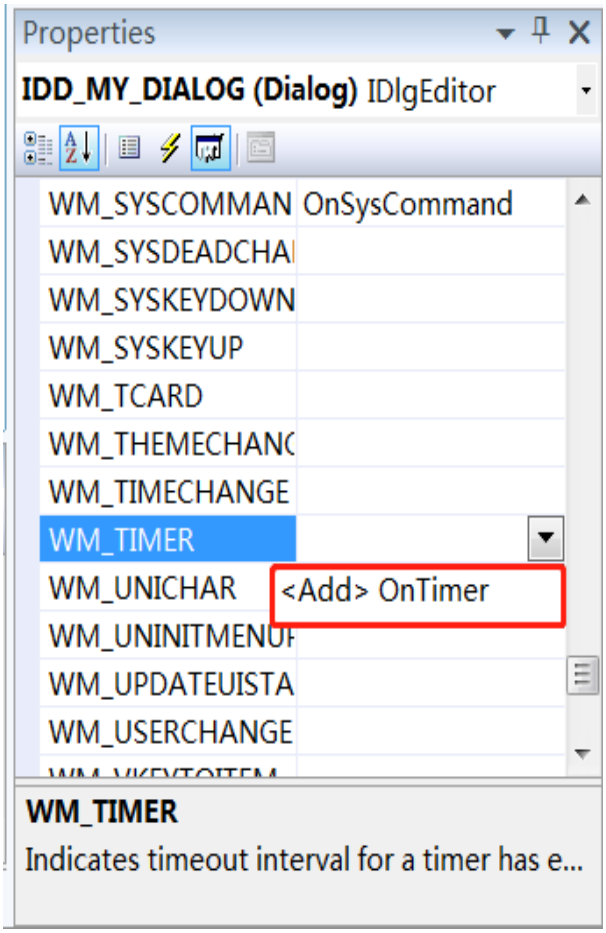


图31

接着会自动进入定时器代码编辑页面，

在void C点位运动Dlg:: OnTimer(UINT_PTR nIDEvent)

{

    // TODO: Add your control notification handler code here

}中输入以下代码，如图32所示

```
void C点位运动Dlg::OnTimer(UINT_PTR nIDEvent)
{
    GTN_GetPrfPos(core,1,&prfPos,1,NULL);            // 读1轴的规划位置
    strTemp.Format(_T("%.3f"),prfPos);
    SetDlgItemText(IDC_STATIC1,strTemp);          //IDC_EDIT1为规划位置右侧静态文本的ID

    GTN_GetPrfVel(core,1,&prfVel,1,NULL);            // 读1轴的规划速度
    strTemp.Format(_T("%.3f"),prfPos);
    SetDlgItemText(IDC_STATIC2,strTemp);

    GTN_GetEncPos(core,1,&encPos,1,NULL);            // 读1轴的实际位置
    strTemp.Format(_T("%.3f"),prfPos);
    SetDlgItemText(IDC_STATIC3,strTemp);

    GTN_GetEncVel(core,1,&encVel,1,NULL);            // 读1轴的实际速度
    strTemp.Format(_T("%.3f"),prfPos);
    SetDlgItemText(IDC_STATIC4,strTemp);

    CDialog::OnTimer(nIDEvent);
}
```

图32

14、然后双击初始化按钮，会自动进入按钮的代码编辑页面，如图33

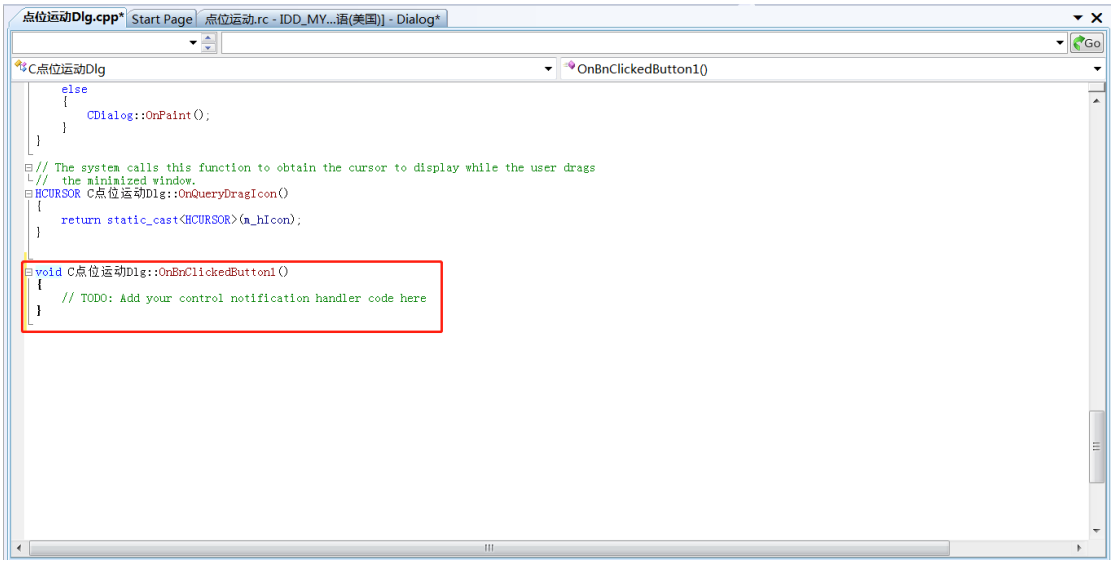


图33

在void C点位运动Dlg::OnBnClickedButton1()

{

// TODO: Add your control notification handler code here

}中输入以下代码，如图34所示

```
void C点位运动Dlg::OnBnClickedButton1()
{
    // TODO: Add your control notification handler code here
    short sEcatSts;                                    // Ether CAT通讯状态变量
    sRtn = GTN_Open();                          // 打开运动控制器
    sRtn = GTN_InitEcatComm(core);                     //初始化EtherCAT通讯配置
    do {                                               // 查询EtherCAT通讯是否准备好了
        sRtn = GTN_IsEcatReady(core,&sEcatSts);
    } while (sEcatSts != 1 || sRtn != 0);
    sRtn = GTN_StartEcatComm(core);                    //开始EtherCAT通讯
    sRtn = GTN_Reset(core);                            // 复位运动控制器
    sRtn = GTN_LoadConfig(core,"gtn_core1.cfg");       // 下载控制器配置文件
    sRtn = GTN_ClrSts(core,1,1);                      // 清除1轴的轴状态
    SetTimer(1,10,NULL);                               //设置定时器
}
```

图34

15、然后依次双击所有按钮，按照步骤 14 的方法分别在每个按钮的代码编辑页面添加代码，如图 35

```
    sRtn = GTN_InitEcatComm(core);                          //初始化EtherCAT通讯配置
    do {                                                    // 查询EtherCAT通讯是否准备好了
        sRtn = GTN_IsEcatReady(core,&sEcatSts);
    } while (sEcatSts != 1 || sRtn != 0);
    sRtn = GTN_StartEcatComm(core);                         //开始EtherCAT通讯
    sRtn = GTN_Reset(core);                                 // 复位运动控制器
    sRtn = GTN_LoadConfig(core,"gtn_core1.cfg");          // 下载控制器配置文件
    sRtn = GTN_ClrSts(core,1,1);                          // 清除1轴的轴状态
    SetTimer(1,10,NULL);                                  //设置定时器
}
void C点位运动Dlg::OnBnClickedButton2()
{
    sRtn = GTN_ClrSts(core,1,1);                    // 清除1轴的轴状态
}
void C点位运动Dlg::OnBnClickedButton3()
{
    sRtn = GTN_AxisOn(core,1);                       // 使能运动轴
}
void C点位运动Dlg::OnBnClickedButton4()
{
    sRtn = GTN_ZeroPos(core,1,1);                       //位置清零
}
void C点位运动Dlg::OnBnClickedButton5()
{
    UpdateData(TRUE);                       // 将各控件中的数据保存到相应的变量
    sRtn = GTN_SetPrfPos(core,1,0);          // 将AXIS轴设为点位模式
    sRtn = GTN_PrfTrap(core,1);              // 读取点位运动参数
    TTrapPrm trap;
    sRtn = GTN_GetTrapPrm(core,1, &trap);
    trap.acc = 0.1;
    trap.dec = 0.1;
    trap.smoothTime = 1;                    // 设置点位运动参数
    sRtn = GTN_SetTrapPrm(core,1, &trap);       // 设置AXIS轴的目标位置
    sRtn = GTN_SetPos(core,1, m_editPos);       // 设置AXIS轴的目标速度
    sRtn = GTN_SetVel(core,1, m_editVel);       // 启动AXIS轴的运动
    sRtn = GTN_Update(core,1<<0);
}
void C点位运动Dlg::OnBnClickedButton6()
{
    sRtn = GTN_Stop(core,1,1);              //停止运动
    sRtn = GTN_TerminateEcatComm(core);     // 关闭EtherCAT通讯
    sRtn = GTN_Close();                     // 关闭运动控制器
}
```

图35

16、检查代码没有错误后，开始调试运行按F5，或者点击如图所示按钮，如图36

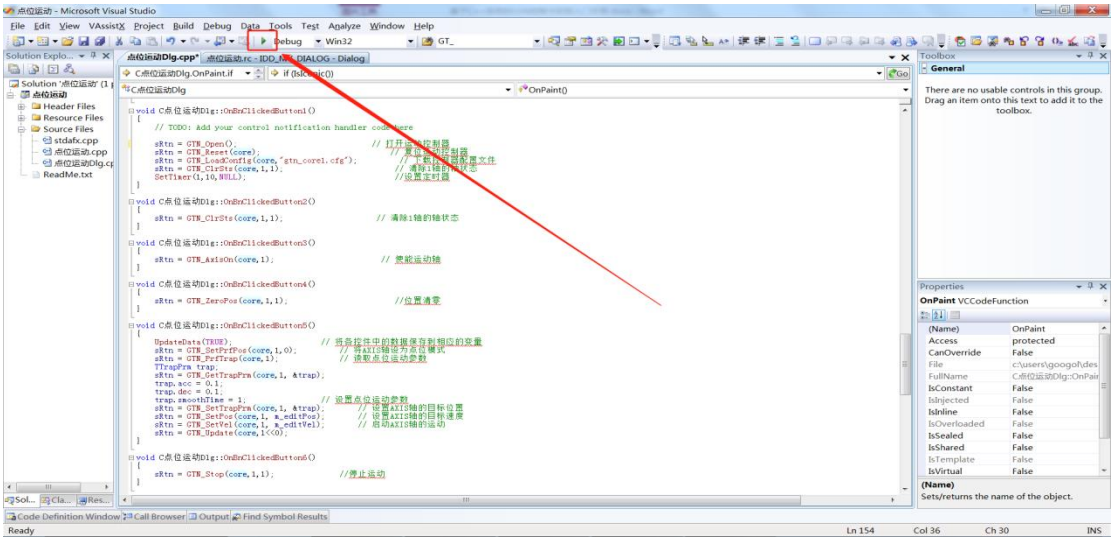


图36

17、输入点位距离和速度后点击开始运动按钮，如图37

图37

18、运动过程中，规划位置和实际位置会实时变化，点击停止按钮可停止点位运动。点击×号可以关闭整个调试程序。