VC 编程实例操作说明

瑞博华公司的数据采集卡提供简单、灵活的接口方式，通过 DLL 调用就可以在 WINDOWS 下高速、连续的采集。根据 DLL 的接口函数在编程指南已经有非常详细的介绍，这里主要是从具体操作的方面进行介绍，便于用户可以快速进行二次开发。

为了介绍方便，这里以本公司的产品 RBH8255 为例，对于其它产品，原理和方法完全一样。

一、VC 下 DLL 调用原理

采集卡提供驱动程序，该驱动程序在光盘的产品目录下对应产品额 Drivers 下，用户只要按照提示就可以快捷地安装上。驱动程序有两部分内容，一部分是以 SYS 为扩展名的驱动程序 USB8255.SYS，另外一部分是动态链接库包括 Adcard.dll，USB8255.DLL，其中 Adcard.dll 和 USB8255.DLL 这两个程序完全一样，为了便于用户程序具有通用性，应用程序就调用相同的 Adcard.dll，可以让程序更加有通用性。驱动程序 SYS 和动态链接库在用安装驱动程序的时候，自动安装到系统中，用户不用对其操作。用户对采集卡操作只需调用动态链接库，而由动态链接库自动调用 SYS 程序。

VC 下调用 DLL 时，采用 Adcard.LIB 的方式进行调用，再通过 Adcard.LIB 调用 Adcard.DLL，然后由 Adcard.DLL 调用 USB8255.SYS，就可以实现对采集卡的操作。因此，要求用户必须把 Adcard.lib 放到当前目录。

二、VC 下 DLL 调用的步骤

如图 1 所示，本软件的实现的功能是实时采集 AD 结果并在屏幕上打印出来，实现开关量采集功能，实现开关量输出功能，基本上实现了全部的硬件功能。在定时器程序中实现全部通道的信号采集与解包，实现连续采集的功能。

图 1 VC 软件运行界面

如图 2 所示，右点击工程的文件，添加 Adcard.lib 和 Adcard.h 文件到工程中。
如图 3 所示，将 Adcard.lib 和 Adcard.h 添加到工程中。

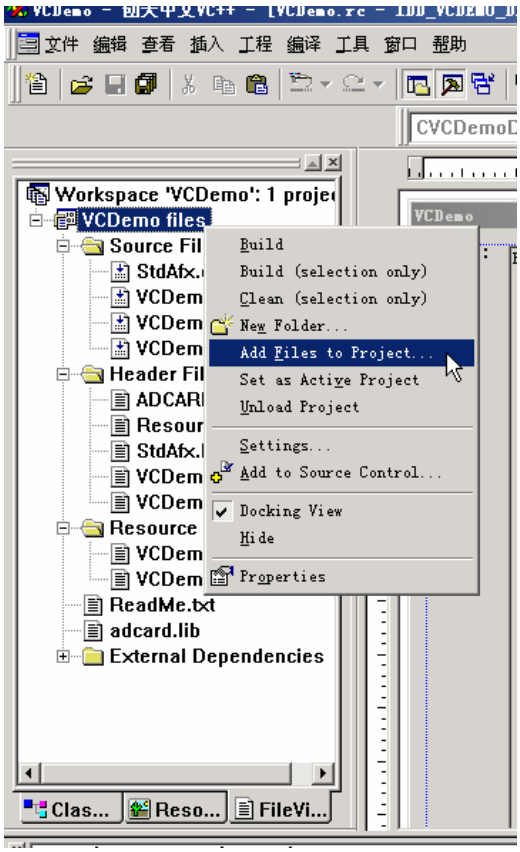

图 2 在工程中添加 Adcard.lib 和 Adcard.h 的方法

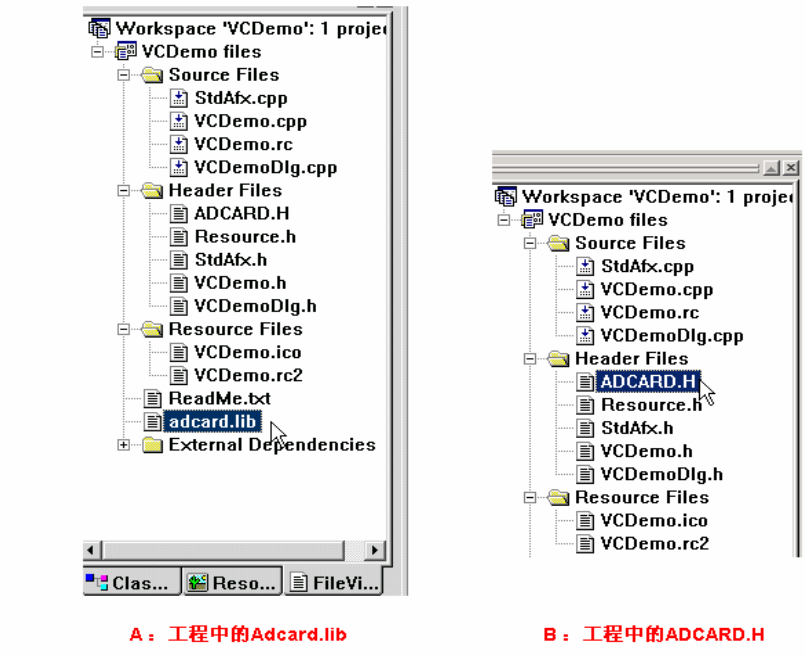

图 3 在工程中添加.lib 和.h 文件

```
BOOL CVCDemoDlg::OnInitDialog()
{
    CDialog::OnInitDialog();

    // Set the icon for this dialog.  The framework does this automatically
    //  when the application's main window is not a dialog
    SetIcon(m_hIcon, TRUE);         // Set big icon
    SetIcon(m_hIcon, FALSE);        // Set small icon

    // TODO: Add extra initialization here
    char DrvName[32];
    int i;
    m_DOChn.SetCurSel(0);
    m_DOVal.SetWindowText("0");
    //从下位机中读取板卡的名称，AD的最大值和零点数据
    ConfigInfo(DrvName,&i,&i,&i,&i,&VZero,&VMax,&i);
    m_strADResult.Format("本机驱动为: %s",DrvName);
    NumBuf = 10;
    NumSamp = 300;
    begchn = 0;
    NumChn = 33;
    FrqSamp = 20000;
    FrqFilter = 0;
    AmpGain = 1;
    ADBuf = (WORD*) malloc( (NumChn*NumSamp+1)*sizeof(WORD) );

    UpdateData(FALSE);
    return TRUE;  // return TRUE  unless you set the focus to a control
}
```

初始化程序

读取采集卡的配置功能

设置基本参数，在启动函数中用于设定采集卡

请注意：NumChn是通道数，需要多少通道就设置多少，有些采集卡的通道设置有特殊要求，请按照硬件说明书上要求进行设置

申请一个数据缓冲区，用于存放来自采集卡的数据

这几个参数非常重要，FrqSamp是采样频率，NumChn是通道数，NumSamp是每个通道采样点数，NumBuf是在内存中开辟多少个ADBuf的缓冲区

图 4 初始化程序

点击屏幕上“启动”命令按钮就执行以下代码

```
void CVCDemoDlg::OnStart()
{
    // TODO: Add your control notification handler code here
    StopIntr();
    if(ADBuf == NULL) {
        MessageBox("Buffer malloc failed!","Warning",MB_OK);
        return ;
    }
    if(Initial(IOBase,IRQNum,PhysAddr,DMAChn) == ADCard_Error) {
        MessageBox("Initial error!","Warning",MB_OK);
        return ;
    }

    if( StartIntr(NumBuf,NumSamp,begchn,NumChn,FrqSamp,FrqFilter,AmpGain) == ADCard_Error) {
        MessageBox("Start Intr error!","Warning",MB_OK);
        return ;
    }
    SetTimer(1,200,NULL);
    GetDlgItem(IDC_START)->EnableWindow(FALSE);
}
```

启动前先停止采集，目的是防止重复启动采集

初始化的方法，也可以放在软件起始的地方初始化

启动采集

开启定时器，准备在定时器中读取采集结果

启动采集的方法

图 5 启动采集的程序

```
void CVCDemoDlg::OnStop()
{
    // TODO: Add your control notification handler code here
    KillTimer(1);
    StopIntr();

    GetDlgItem(IDC_START)->EnableWindow(TRUE);
}
```

停止定时器

停止采集功能

停止采集的例程

图 6 停止采集的程序

```
void CVCDemoDlg::OnTimer(UINT nIDEvent)
{
    // TODO: Add your message handler code here and/or call default
    WORD RChn;
    RChn=NumChn;
    if(NumChn>MAXCHN)RChn=MAXCHN;//纪录到数组中的通道数，一方面纪录通道数不能大于采集通道数，另一方
    WORD NumFill = QueryBuf();  //填充满的缓冲区个数
    for (int i=0;i<NumFill;i++)//表明本次采集的数据包数，每包数据存放在ADBuf数组中
    {
        if( ADResult((struct structADResult *)ADBuf) == ADCard_Success) {//读出一包数据到ADBuf中
            //下面演示如何从ADBuf数组中将通道解包，实现连续采集的功能，将数据保存到数据缓存中
            for(WORD j=0;j<NunSamp;j++){//每包内的数据点数（注意，这个点是指时刻点，在该深刻点每个
                for(int k=0;k<RChn;k++){//将每个点内的数据分配到各个通道内
                    ChnRecordData[k][RecordSeq]=ADBuf[1+j*NumChn+k];
                }
                RecordSeq++;//下一个时刻点的数据存放指针
                if(RecordSeq==MAXREC)RecordSeq=0;//数据在数组中循环存放，根据RecordSeq就可以判断当
            }
            //在此处可以增加存盘、绘图等功能
            //DisplayResult();    //直接从ADBuf中读取数据的显示方法
            DisplayResult_Chn();  //从数据缓冲区ChnRecordData中显示数据的方法
        }
    }
    //读取开关量输入
    WORD IOResult[32];
    Rbh_DI(2,IOResult);
    m_strIOResult.Format("%X   %X",IOResult[0],IOResult[1]);

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

确定要记录到数组中的通道数

首先查询已经采集了多少数据包

一次读取一包数据

将一包内的数据进行解包，然后保留到数组中

两种数据显示方法，用户可以通过注解一个而选择另一个来检验

演示开关量输入函数RBH_DI的使用方法

数据读取的方法，关键是数据如何解包

图 7 读取采集结果的程序

```
//直接从ADBuf中读取数据的显示方法
void CVCDemoDlg::DisplayResult(void)
{
    CString strTmp;
    m_strADResult.Format("块 %d\r\n",(int)ADBuf[0]);
    for (int i=1;i<NumChn;i++) //由于第一个通道是开关量，所以跳过
    {
        double fVal = (ADBuf[1+i]-VZero)/(VMax-VZero)*5.;
        strTmp.Format("Chn%d = %7.4f\r\n",i,fVal);
        m_strADResult += strTmp;
    }
}
//下面的程序演示从大缓冲区ChnRecordData中显示数据的方法
void CVCDemoDlg::DisplayResult_Chn(void)
{
    CString strTmp;
    WORD RChn;
    RChn=NumChn;
    if(NumChn>MAXCHN)RChn=MAXCHN;//纪录到数组中的通道数，一方面纪录通道数不能大于采集通道数，另一

    int LastPoint;//当前最新的数据点
    if(RecordSeq==0)LastPoint=MAXREC;
    else LastPoint=RecordSeq-1;

    m_strADResult.Format("块:记录缓冲区中显示数据 \r\n");
    for (int i=1;i<RChn;i++) //由于第一个通道是开关量，所以跳过
    {
        double fVal = (ChnRecordData[i][LastPoint]-VZero)/(VMax-VZero)*5.;

        strTmp.Format("Chn%d = %7.4f\r\n",i,fVal);
        m_strADResult += strTmp;
    }
}
```

从ADBUF中读取最新数据的方法

AD数据的读取与电压的转换

onTimer程序中选择

从连续缓冲区中得到最新的数据点

从连续缓冲区中读取数据并转换为电压

在屏幕上打印结果

这是演示两种数据读取的方法：上面一种是仅仅读取当前包的第一个数据点，下面是在连续记录中读取最新的数据

图 8 显示采集结果的两种程序