PCI8616带ICP功能高精度并行数据采集卡

使 用 说 明 书

目　录

第一章　概　　述

PCI8616 板卡介绍：

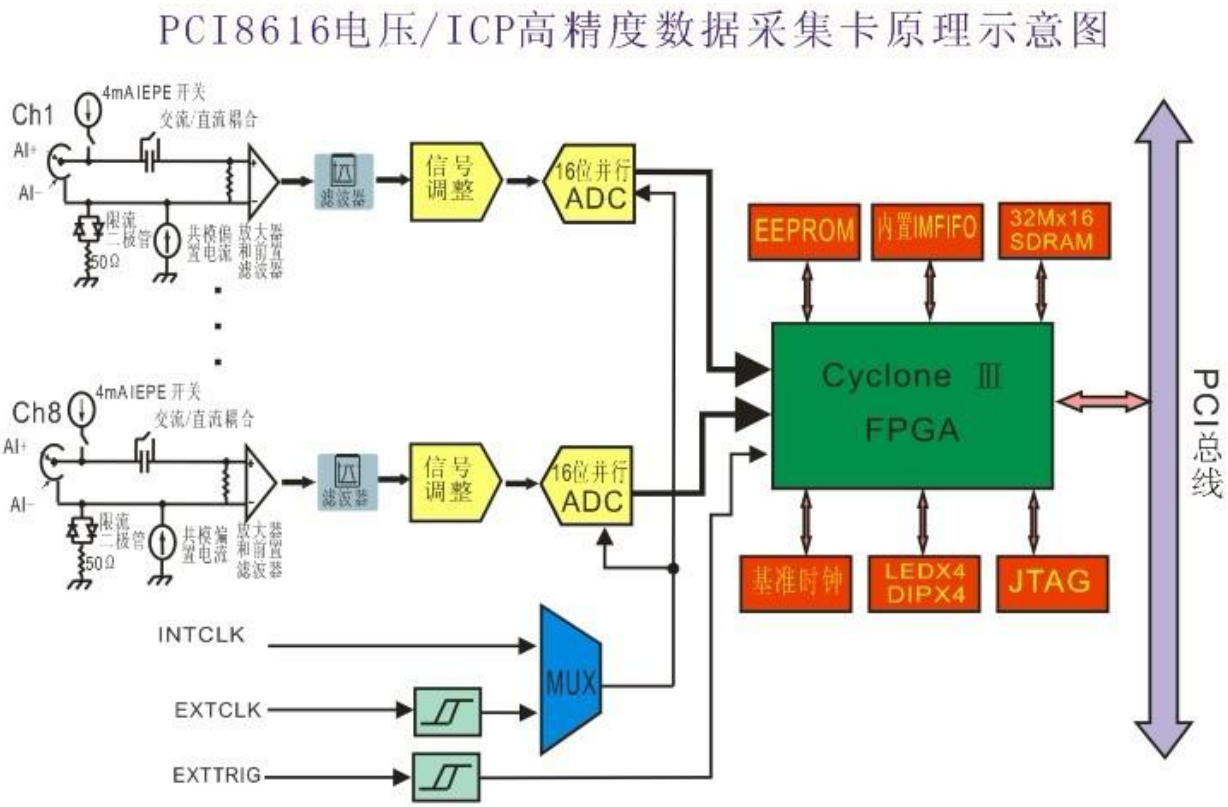
PCI8616 是一款集数据采集、信号产生、扫频于一体的产品，将它插入计算机 PCI 槽上，再运行 PCI8616 虚拟示波器软件，便可组成一台价格便宜、人机界面友好、性能优良的数字存储示波器。它具有数据采集、测量信号、过程监测、多种触发等功能，因此大量应用于高速的数据采集系统、自动测试系统、自动控制系统。

型 号	通道数	采样率	AD精度	ICP	采样长度	DI	DO	外时钟	外触发	1KHz
PCI8616VOL_2CH	2CH	1MSPS	16		连续采样	4	4	√	√	√
PCI8616VOL_4CH	4CH	1MSPS	16		连续采样	4	4	√	√	√
PCI8616VOL_8CH	8CH	1MSPS	16		连续采样	4	4	√	√	√
PCI8616ICP_2CH	2CH	1MSPS	16	√	连续采样	4	4	√	√	√
PCI8616ICP_4CH	4CH	1MSPS	16	√	连续采样	4	4	√	√	√
PCI8616ICP_8CH	8CH	1MSPS	16	√	连续采样	4	4	√	√	√

一、PCI8616 主要性能指标

最大采样率：	1Msps
单台通道数：	并行 8CH 采集+4DI+4DO+1KHz5V TTL 方波输出+8 路 DO+外时钟+外触发
ADC 分辨率：	16Bits，直流精度≤±0.1% 系统精度:≤±0.3%
存储容量：	连续采集，FIFO 32ksa/ch
量程：	±1V、±2V、±5V、±10V
输入方式：	BNC 单端双极性伪差分电压输入
输入阻抗：	正端 1MΩ；负端 50 欧
输入信号带宽：	0Hz～100KHz（-3dB）
通道间相位差：	≤0.01°（正弦波 1KHz）
带内波动：	≤±0.1dB(0Hz～300KHz)
时基范围：	1MSPS~1KSPS 分 10 挡
耦合方式：	AC/DC
触发模式：	正常、自动、单次
触发边沿：	上升、下降
触发模式：	正常、自动、单次
触发通道：	CH1~CH8、EXT(外触发)
通道间隔离度：	≥80Db
尺寸：	192mm×112mm
重量：	0.2Kg

二、PCI8616 高精度数据采集卡原理示意图



第二章　安装

一、最低配置：PⅡ及其兼容机，1024X768 显示器，2G 内存、WindowsXP 操作系统。

二、PCI8616 板卡外型

PCI8616 板卡实物图所示：

板卡外观示意图

其中

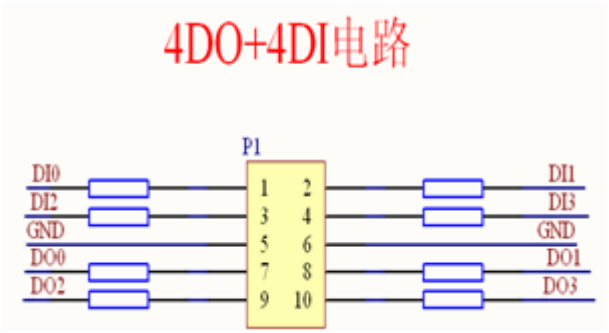
CH0=1 通道输入

CH1=2 通道输入

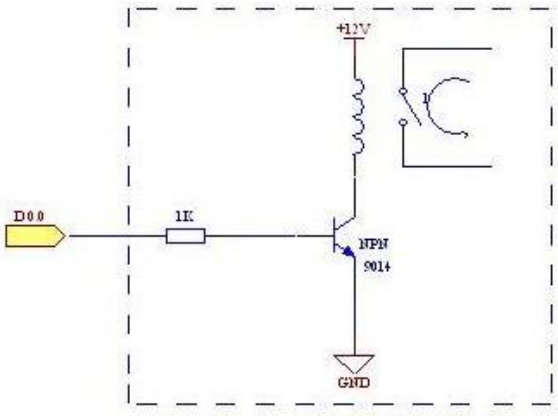
CH2=3 通道输入

CH3=4 通道输入

三、PCI8616 的 DI DO 管脚定义



PCI8616 DO口驱动外部继电器示意图



虚线框内为用户电路

四、PCI8616 卡安装步骤

1) 关闭计算机电源。
2) 在一空闲 PCI 槽插入本板卡。
3) 启动计算机，安装设备驱动程序，为光盘\driver\PCI8616.inf
4) 安装 PCI8616 软件，为光盘 Setup\Setep.exe，按提示操作即可。
5) 在“控制面板”\“系统”下可以看到：

第三章　PCI8616 软件

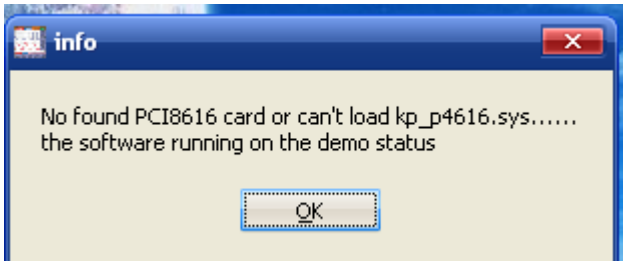
一、运行环境

WindowsXP 操作系统，2G 内存，1024x768 分辨率。

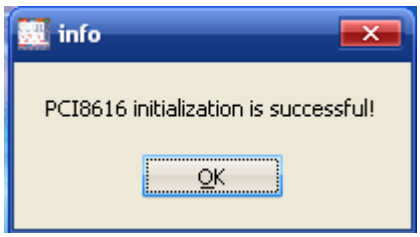
二、软件运行

运行桌面 

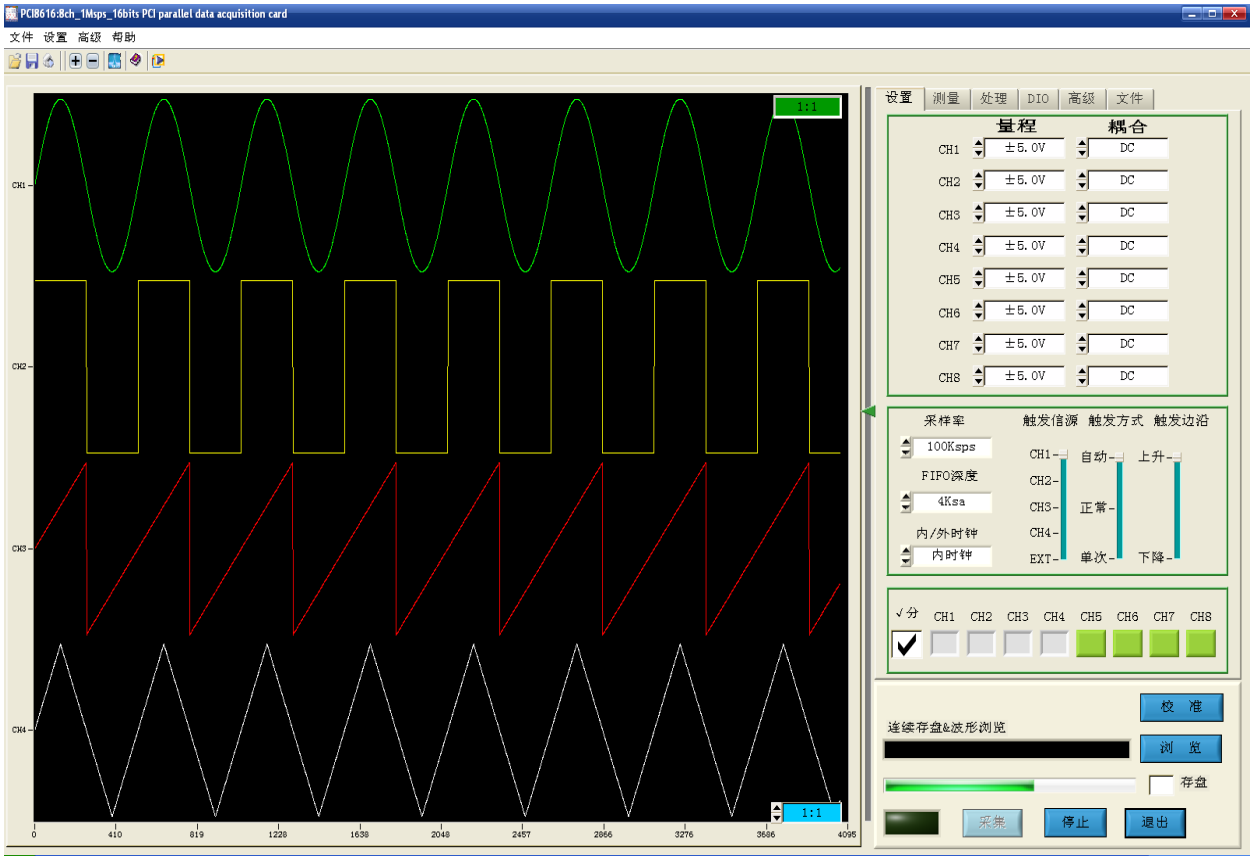
PCI8616 程序，程序进入自检。若没有 PCI8616，会弹出对话框



PCI8616 软件会运行于演示状态。
如 PCI8616 卡及设备驱动程序正确安装，弹出

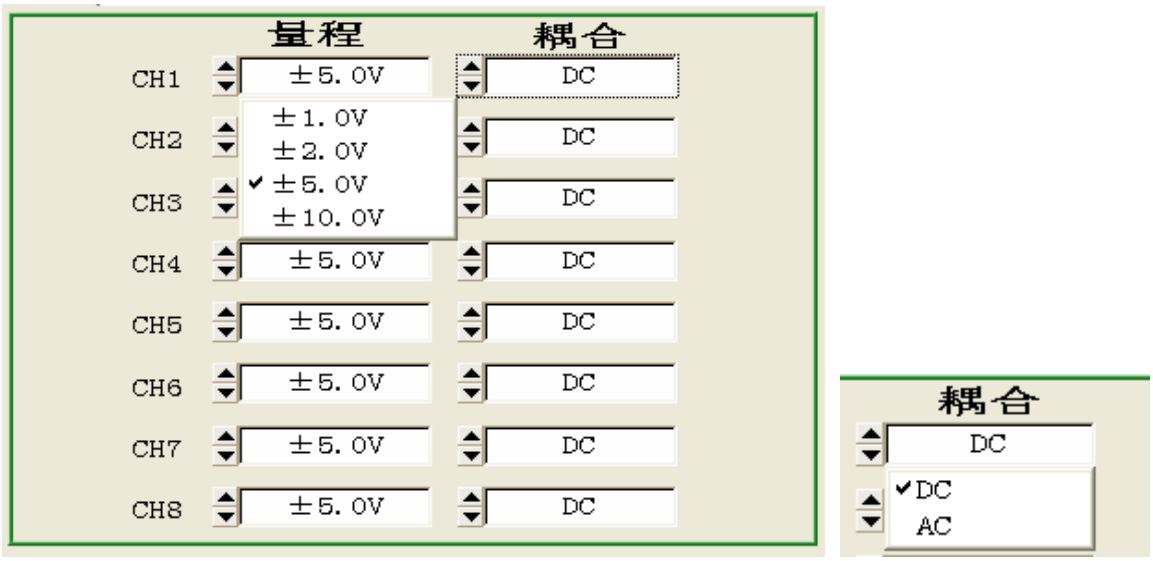


按 OK 进入主程序界面



三、软件功能

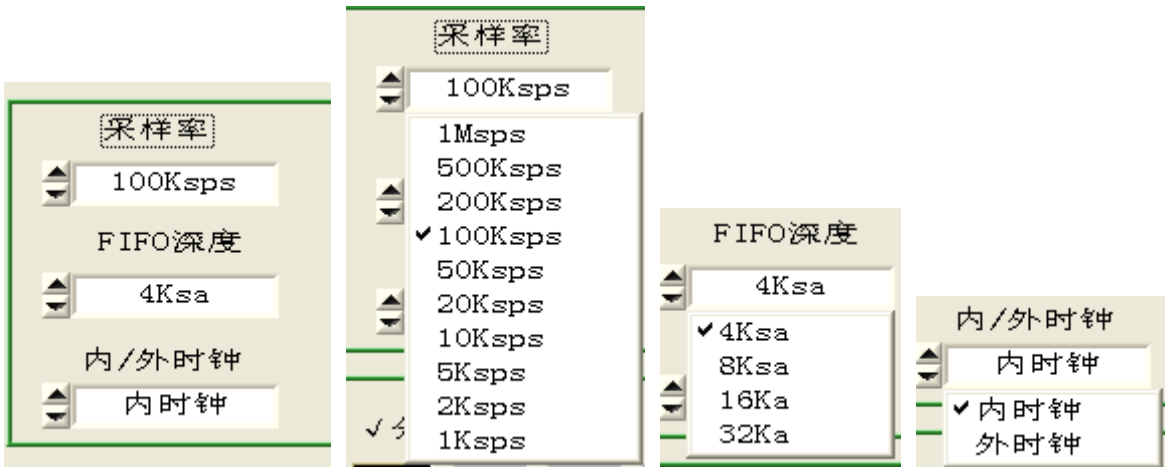
3.1 量程和耦合方式



1、每个通道有±1V、 ±2V、±5.0 V、±10V 挡量程

2、每个通道有 DC AC 两种耦合模式，对于 ICP 型传感器，设置为 AC 模式，PCI8616 卡向传感器提供 24V4mA 恒流源驱动传感器工作。

3.2 采样设置

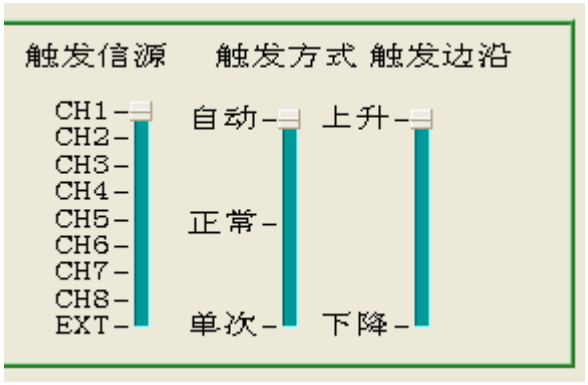


1、采样率：从 1Msps 到 1Ksps 分 10 档

2、FIFO 深度：PCI8616 只有采集完一个 FIFO 深度的数据后，才会把波形 DMA 到内存，所以采样率设置 4Ksa,中采样率设置为 8Ksa,高采样率设置为 32Ksa。

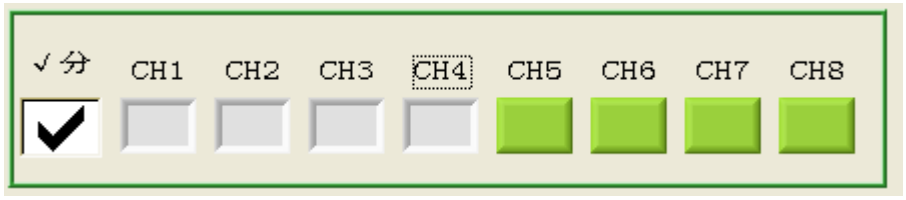
3、内外时钟：PCI8616 卡支持外时钟，范围 1KHz~200KHz,TTL 方波。

3.3 触发设置



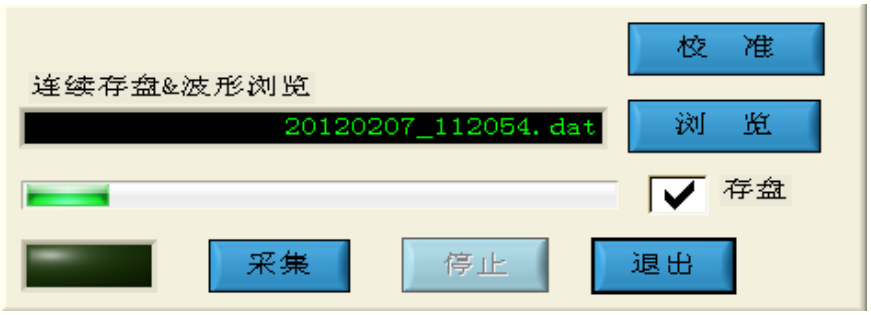
1、触发信源：CH1~CH8 EXT 9 个触发源。
2、触发方式：自动、正常、单次。
自动：软件启动采集。
正常：满足触发条件后，启动采集，一直到人工干预后停止。
单次：满足触发条件后，启动采集，采集一个 FIFO 深度，停止采集。
3、触发边沿：上升、下降。
4、触发电平：满量程的-100%~100%内调整。

3.4 波形显示



1、分/合：分现，屏幕按设置显示的通道数分割。合显，各通道共用一个屏幕。
2、显示通道：选择要显示波形的通道

3.5 采集控制



1、 

启动采集时，自动按系统时间创建一个文件。

2、 

启动采集

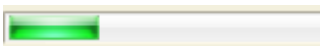
3、 

停止采集

4、 

退出程序

5、 
工作指示灯

6、 
采集进度。

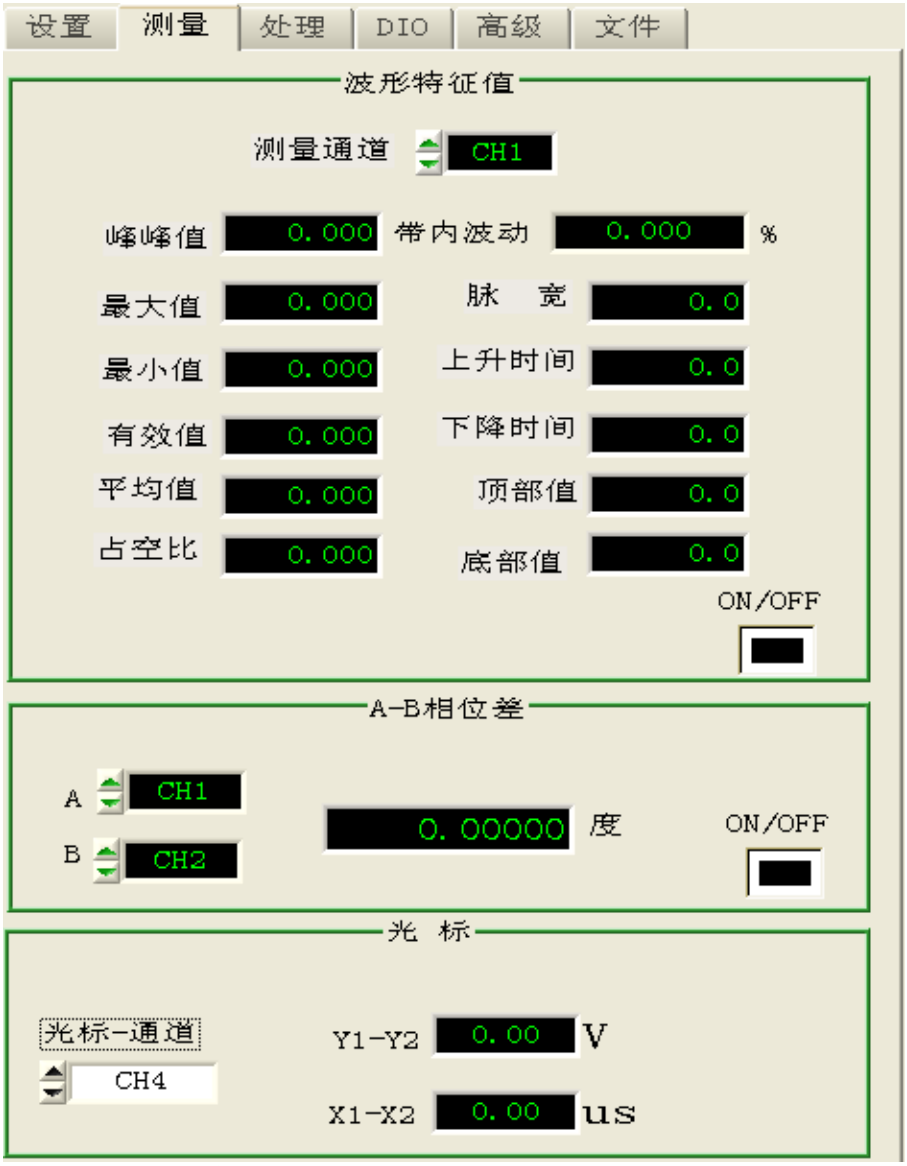
7、 

详见视频文件 “PCI8616 波形的浏览.exe”

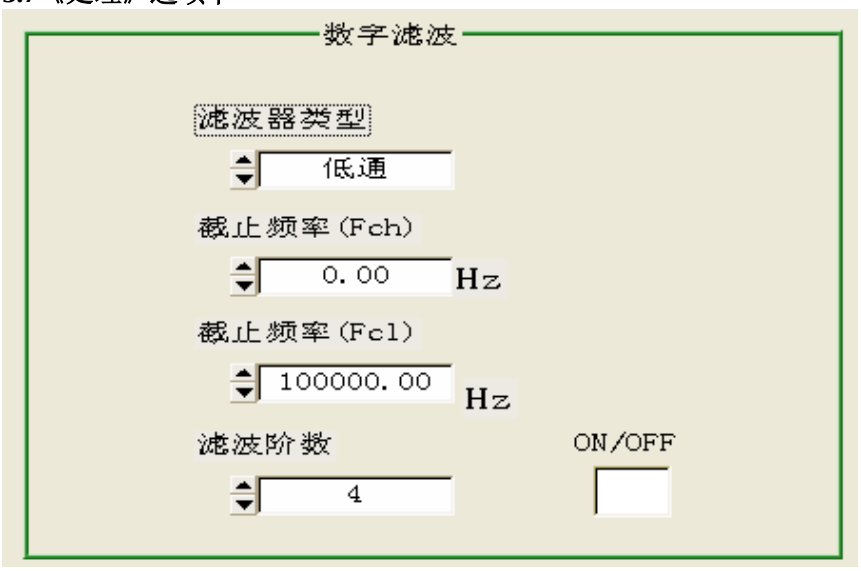
8、 

详见视频文件 “PCI8616 的校准.exe”

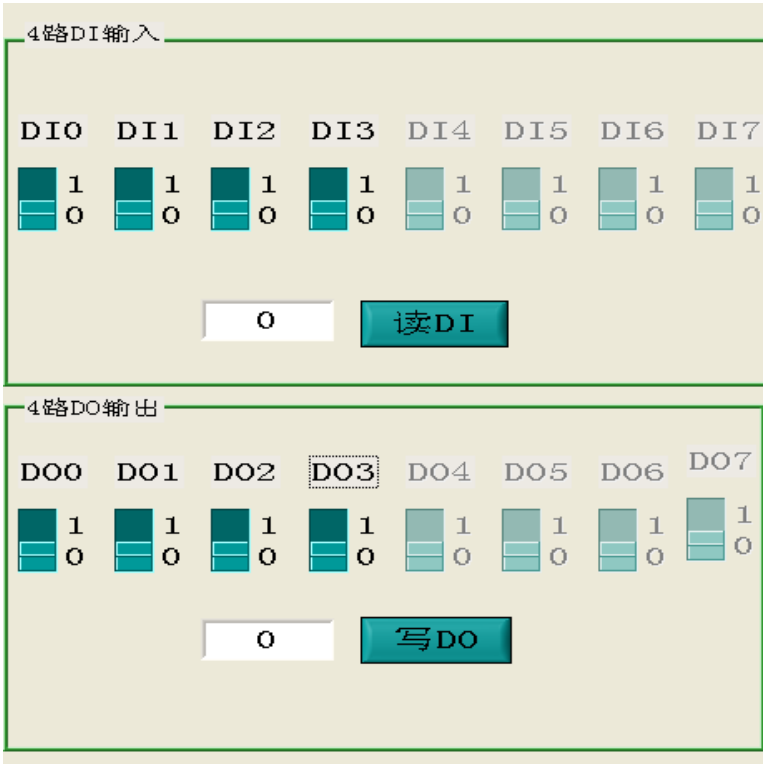
3.6《测量》选项卡



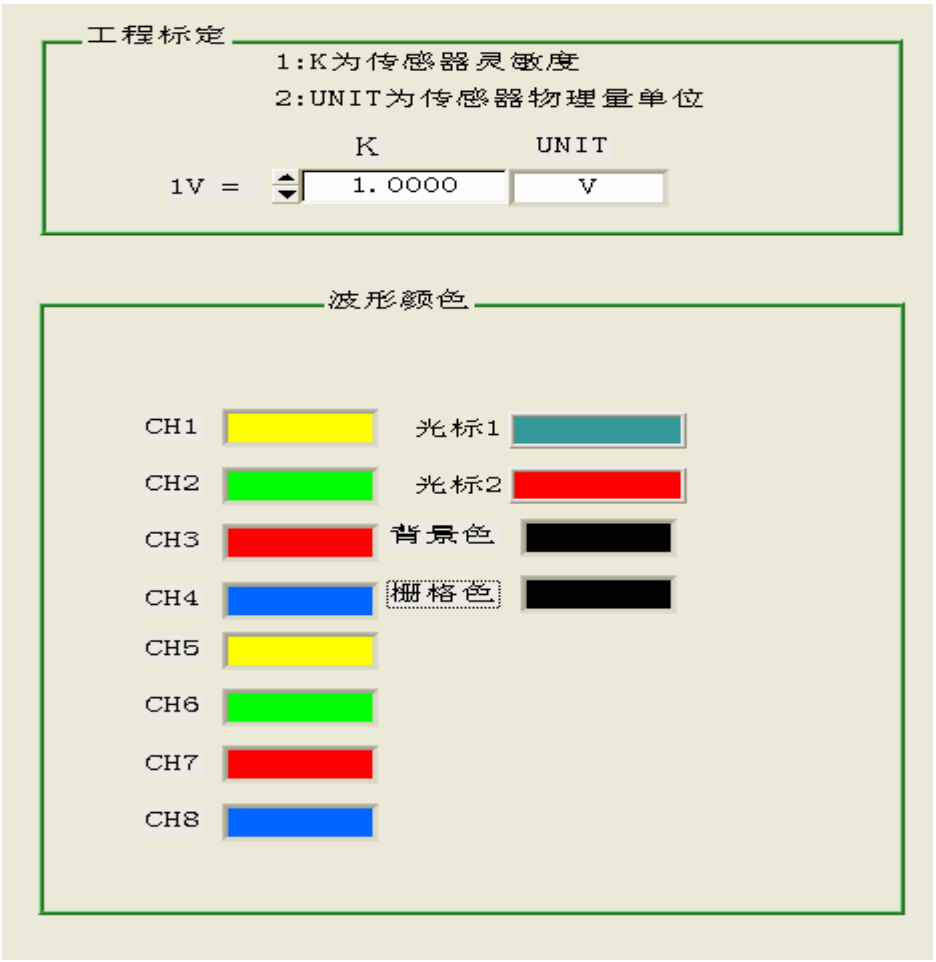
3.7《处理》选项卡



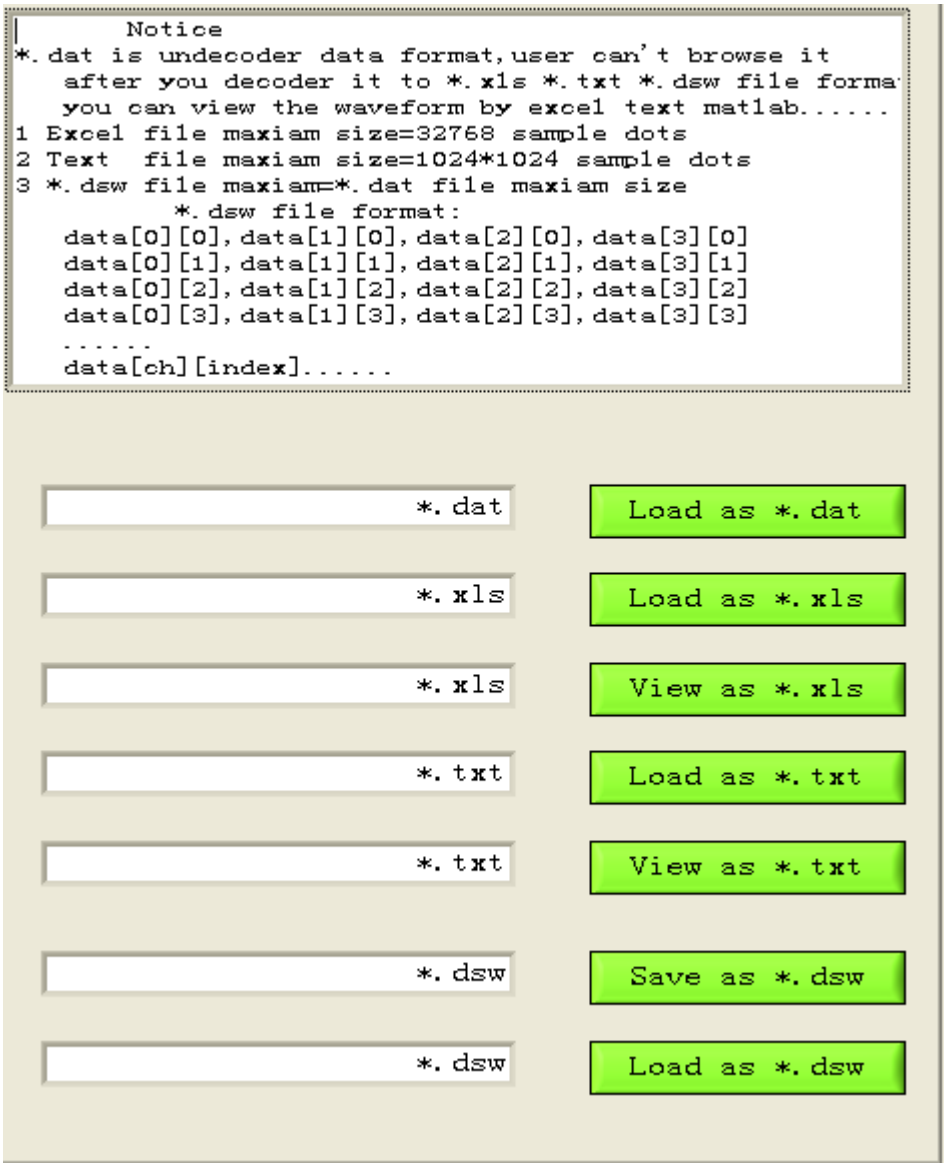
3.8《DIO》选项卡



3.9《高级》选项卡

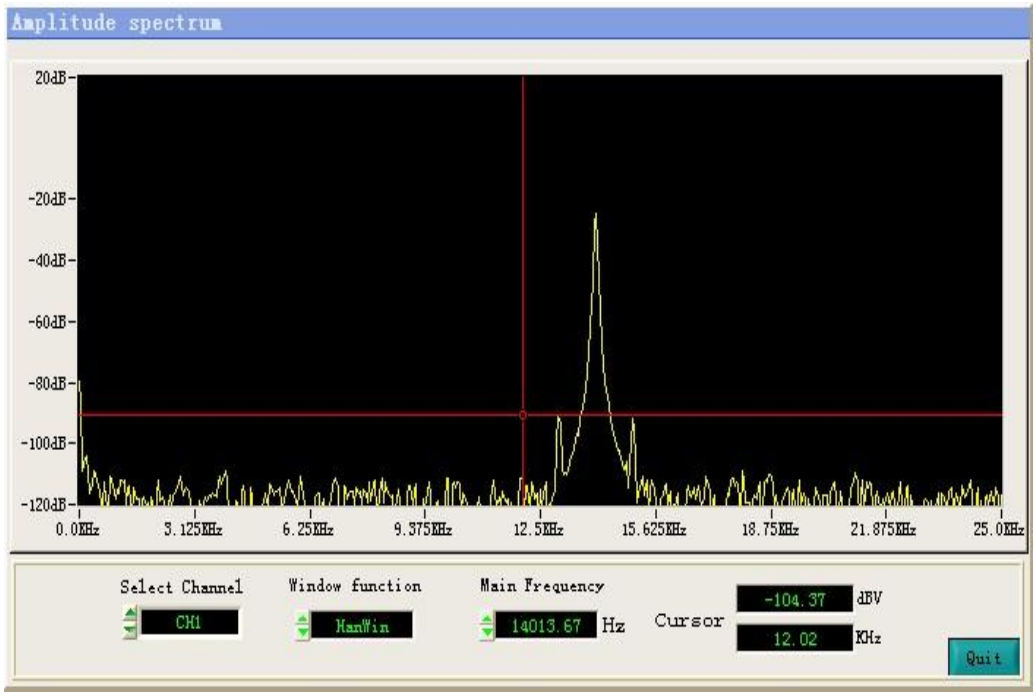


3.10《文件》选项卡



1、详见视频文件 “PCI8616 的文件的转换.exe”

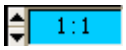
《FFT》功能



3.11 波形的放大、缩小、和左右移动

 X 轴扩展波形

 X 轴压缩波形



Y 轴放大波形

拉动水平滑动条  可左右移动波形。

3.12 其它功能

a)拖动波形显示区下的滑块，水平移动波形，可观测波形其它部分。

b)在暂停时，波形显示区内，按CTRL+鼠标右键，无级放大波形。

c)在暂停时，波形显示区内，按CTRL+鼠标左键，无级缩小波形。

四、 菜单功能

1、文件

打开：调用一个波形文件到显示区

保存：保存当前波形

打印：打印当前波形

退出：退出 PCI8616 软件

2、设置

采集设置：设置采集参数

刷新率： 设置波形显示间隔

波形显示：设置波形显示颜色

3、高级

FFT： 调 FFT 进行频率谱分析。

低通滤波：调用低通滤波对波形进行处理。

4、关于

关于 PCI8616

第四章　PCI8616 卡二次开发手册

一、二次开发概述

PCI8616 提供标准的动态连接库，用户可通过调用动态连接库里的函数，完成对 PCI8616 卡的控制，库文件包括 PCI8616DLL.DLL、PCI8616DLL.LIB、PCI8616DLL.H 三个文件。

所用到的数据结构：

PCI8616 采集卡系统信息

```
struct TSysInfo
{
        unsigned int Idnumber;          //卡 ID 号
        double  GainTable[8][4];        //各通道增益
        double  BaseLine[8][4];         //各通道零点
        double  AD_ClkTable[20];        //本卡采样率列表
        unsigned int SampleRateIdx;     //设置采样率序号
        unsigned int RangeIdx[8];       //设置各通道量程
        unsigned int CoupleIdx[8];      //设置各通道耦合
        unsigned int TrigMode;          //设置触发模式
        unsigned int TrigEdge;          //设置触发边沿
        unsigned int TrigSource;        //设置触发源
        unsigned int TrigLevel;         //设置触发电平
        unsigned int clkmode;           //设置时钟模式
        unsigned int FifoSizeIdx;       //设置 FIFO 深度
        double mUnitK;                  //工程标定之系数
        unsigned char mUnitStr[20];     //工程标定之单位

};
```

其中定义：

SampleRateIdx:采样率索引号

索引号	采样率
0	1Msps
1	500Ksps
2	200Ksps
3	100Ksps
4	50Ksps
5	20Ksps
6	10Ksps
7	5Ksps
8	2Ksps
9	1Ksps

RangeIdx[n]：CHn 的量程设置

索引号	量 程
0	±1V
1	±2V
2	±5V
3	±10V

CoupleIdx[n]：CHn 的耦合设置

索引号	耦 合
0	DC
1	AC

TrigMode:触发模式

索引号	模 式
0	自动
1	正常

TrigEdge:触发边沿

索引号	模 式
0	上升
1	下降

TrigSource：触发通道

索引号	触发通道
0	CH1
1	CH2
2	CH3
3	CH4
4	CH5
5	CH6
6	CH7
7	CH8
8	EXT

TrigLevel:触发电平

0~255 对应当前量程幅度的-100%~+100%

二、PCI8616DLL.DLL 函数简介:

2.1 初始化函数

```
PCI8616_Init(unsigned int *deviceid,
             double BaseLine[8][2],
             double GainTable[8][2]
             );
```

功能描述：PCI8616 初始化

入口参数：无。

出口参数：deviceid：设备 ID 号

BaseLine ：PCI8616 卡的零点补偿。

GainTable：PCI8616 的增益配置参数。

函数返回：0，无卡。

1，有卡。

2.2 设置采集控制参数

```
PCI8616_VBSetHardWare(
                        unsigned int ClkMode,
                        unsigned int TrigMode,
                        unsigned int TrigEdge,
                        unsigned int TrigSource,
                        unsigned int TrigLevel,
                        unsigned int SampleIdx,
                        unsigned int FifoSizeIdx,

                        unsigned int RangeIdxch0,
                       unsigned int RangeIdxch1,
                       unsigned int RangeIdxch2,
                       unsigned int RangeIdxch3,
                       unsigned int RangeIdxch4,
                       unsigned int RangeIdxch5,
                       unsigned int RangeIdxch6,
                       unsigned int RangeIdxch7,
```

```
                    unsigned int CoupleIdxch0,
                    unsigned int CoupleIdxch1,
                    unsigned int CoupleIdxch2,
                    unsigned int CoupleIdxch3,
                    unsigned int CoupleIdxch4,
                    unsigned int CoupleIdxch5,
                    unsigned int CoupleIdxch6,
                    unsigned int CoupleIdxch7
                                        );
        unsigned int SampleRateIdx;     //设置采样率序号
        unsigned int RangeIdx[8];       //设置各通道量程
        unsigned int CoupleIdx[8];      //设置各通道耦合
        unsigned int TrigMode;          //设置触发模式
        unsigned int TrigEdge;          //设置触发边沿
        unsigned int TrigSource;        //设置触发源
        unsigned int TrigLevel;         //设置触发电平
        unsigned int clkmode;           //设置时钟模式
        unsigned int FifoSizeIdx;       //设置 FIFO 深度
        double mUnitK;                  //工程标定之系数
```

功能描述：设置采集控制参数
入口参数：ClkMode：时钟模式
TrigMode：触发模式
TrigEdge：触发边沿
TrigSource：触发源
TrigLevel：触发电平
SampleIdx：采样率序号
FifoSizeIdx：FIFO 深度

RangeIdxch0：CH1 量程
RangeIdxch1：CH2 量程
RangeIdxch2：CH3 量程
RangeIdxch3：CH4 量程
RangeIdxch4：CH5 量程
RangeIdxch5：CH6 量程
RangeIdxch6：CH7 量程
RangeIdxch7：CH8 量程

CoupleIdxch0:CH1 耦合
CoupleIdxch1:CH2 耦合
CoupleIdxch2:CH3 耦合
CoupleIdxch3:CH4 耦合
CoupleIdxch4:CH5 耦合
CoupleIdxch5:CH6 耦合
CoupleIdxch6:CH7 耦合
CoupleIdxch7:CH8 耦合
函数返回：无。

2.3 启动采集

void PCI8616_Acq(void);
功能描述：启动 PCI8616 卡采集数据
入口参数：无。
函数返回：无。

2.4 暂停采集

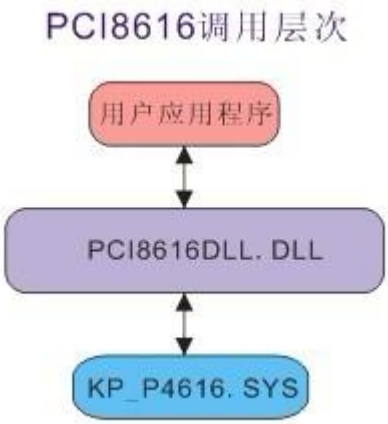
void PCI8616_Stop(void);
功能描述：暂停 PCI8616 卡采集数据
入口参数：无。
函数返回：无。
2.5 退出采集
void PCI8616_Exit(void);
功能描述：释放驱动程序，退出采集
入口参数：无。
函数返回：无

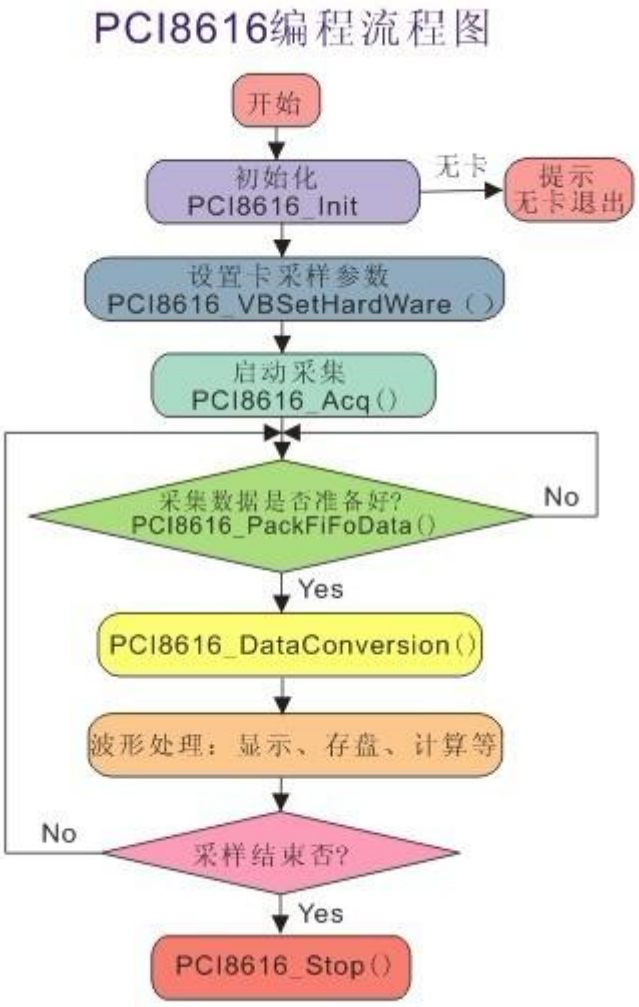
2.6 读取 FIFO 数据
PCI8616_PackFifoData(int Dots,
*double *WaveData1,*
*double *WaveData2,*
*double *WaveData3,*
*double *WaveData4,*
*double *WaveData5,*
*double *WaveData6,*
*double *WaveData7,*
*double *WaveData8*
);
入口参数：Dots:FIFO 深度，如 4096
出口参数：WaveData1: CH0 数据 -1.0~+1.0
WaveData2: CH1 数据 -1.0~+1.0
WaveData3: CH2 数据 -1.0~+1.0
WaveData4: CH3 数据 -1.0~+1.0
WaveData5: CH4 数据 -1.0~+1.0
WaveData6: CH5 数据 -1.0~+1.0
WaveData7: CH6 数据 -1.0~+1.0
WaveData8: CH7 数据 -1.0~+1.0

函数返回：1，数据已刷新
0， 数据喂刷新，无效

2.7 读 4 路数字量输入 DI
int PCI8616_ReadDI(void);
功能描述：读 DI 状态
入口参数：无。
函数返回：数字量
如 DATA= PCI8616_ReadDI();
DATA.D0-----DI0
DATA.D1-----DI1
DATA.D2-----DI2
DATA.D3-----DI3
2.8 写 4 路数字量输入 DO
void PCI8616_WriteDO(int data);
功能描述：写 4 路 DO 状态
入口参数：无
函数返回：数字量
如 PCI8616_WriteDO(DATA);
DATA.D0-----DO0
DATA.D1-----DO1
DATA.D2-----DO2
DATA.D3-----DO3

三、函数调用步骤


PCI8616调用层次
用户应用程序
PCI8616DLL. DLL
KP_P4616. SYS


PCI8616编程流程图
开始
初始化
PCI8616_Init
无卡
提示
无卡退出
设置卡采样参数
PCI8616_VBSetHardWare ()
启动采集
PCI8616_Acq()
采集数据是否准备好?
PCI8616_PackFiFoData()
No
Yes
PCI8616_DataConversion()
波形处理：显示、存盘、计算等
No
采样结束否?
Yes
PCI8616_Stop()

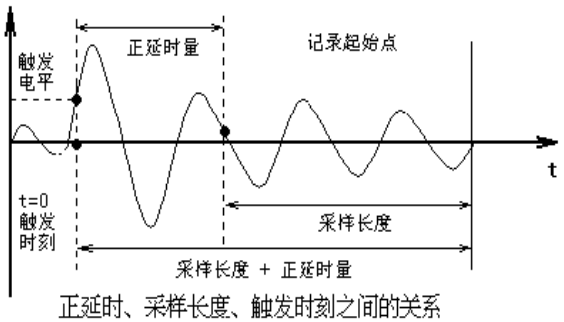
附件一、触发的基本说明

a)触发模式：包括自动触发、正常触发和单次触发。区别是：正常触发时，只有触发事件存在，并满足触发条件，才能触发采样并回送状态，否则不回送状态；而自动触发时，如果在一段时间内（这段时间可以通过调节自动触发的存储深度来调整）有触发事件，则按照触发事件进行触发，反之则强制进行触发采样并回送状态；单次触发，触发条件满足后，采样一次便停止。

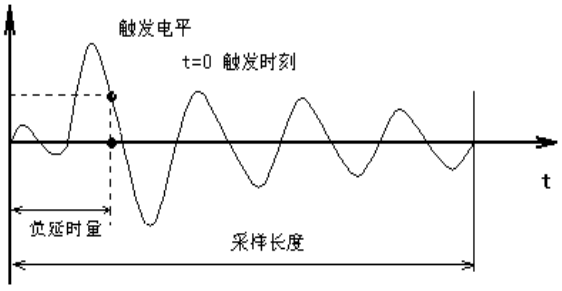
b)预触发：就是触发事件来到之前，所采集的数据量。本卡无预先触发。

c)触发边沿：包括上升沿触发和下降沿触发。

d)触发信源：即产生触发事件的信号源，包括CH1~CH8 、EXT。



正延时、采样长度、触发时刻之间的关系



负延时、采样长度、触发时刻之间的关系

1-1．触发源选择与触发电平设置

关于延时长度、采样长度及触发时刻的关系请见图。

触发：触发功能代表着对信号的捕捉能力，采集分析产品设有 五种触发方式，这些方式根据多种不相同的条件来触发和采集数据。既有手动触发（软件触发），外触发，上升沿内触发，下降沿内触发等。

内触发（又称沿触发）：由被捕捉信号本身使仪器开始采集和记录。如图所示：仪器设定的触发电平为 0.5V，所采集到幅值为 4V 的正弦波波形。在使用内触发时，若采用连续采集功能，则该仪器可以当一台大容量的数字示波器使用。

延时触发：指仪器采集的信号，记录的起始点位置较触发电平前或后（时间轴上，以触发信号到达为 0 时刻），延时触发分为正延时触发和负延时触发，如图所示。

正延时触发：无须观察信号波形的前沿部分或触发后一段时间才会有波形出现。

负延时触发：主要观察上升，下降前沿的波形或波形以前的信号（如：触发事件之前的有效信号

附件二、采样率、采样长度的选择与设置

采样是对模拟信号的时间量化的过程，经过采样以后的信号是“离散时间信号”，它只表达时间轴上一些离散点 0，T，2T，…nT，上的信号值。这里的 T 是采样的间隔时间，因此，采样率就为 1/T。

根据奈奎斯特(Naquist)定律，只要采样率大于信号的最高频率的 2 倍，就能够不失真的把离散时间数字信号恢复到连续的模拟信号，否则就会造成采样信号的频谱混迭，产生失真的“伪波形”。下面几点，用户在设置的时候必须注意一下。

1、但采样率和采样时间是紧密相连的，在同样的采样长度下，采样率越高，则采样时间越短。采样率，采样时间，采样长度的关系为：采样时间=采样长度/采样率。在采样长度有限的情况下，如果信号的时间长，而所关心的信号频率不是很高的话，那么采样率不必设得过高，但采样率必须大于所关心的信号频率的 2 倍以上。

2、采样时间应该大于信号的持续时间。

3、一般情况下，先满足采样率，根据采样时间，再设采样长度。

附件三、配置文件

每一张PCI8616卡都只有唯一的ID号，如PCI8616XXXX。其对应的配置文件为PCI8616XXXX.CFG 用户在二次开发的时候，请将PCI8616XXXX.CFG拷贝到应用程序目录。

附件四、校准

PCI8616卡在使用较长时间后，测量误差可能会超过出厂测定值，用户可以寄回我公司免费重新校准，也可按以下流程自行校准。

附件五、补充函数说明

myAdvanalysis.dll 中提供了一些求波形特征值的函数

1） 求一组波形的最大值、最小值、有效值、平均值,占空比。

```
int Pack_SignalWaveformFeature
                    (int dots,
                     double data[],
                     double *Vpp,
                     double *Vmax,
                     double *Vmin,
                     double *Vrms,
                     double *Vmean,
                     double *duty,
                     double *stdev);
```

功能描述：求波形的最大、最小、有效值、平均值。

入口参数：Dots: 参与计算的点数。

入口参数：data: 波形数据。

出口参数：vpp: 峰峰值

vmax: 最大值

vmin: 最小值

vrms: 有效值

vmean: 平均值

duty: 占空比

stdev：均方根值

2)、谱分析函数

功能描述：FFT 谱分析。

```
// 幅度谱分析
int   Pack_SignalAmplitudeSpectrum(int Dots,                    // 输入波形数据数组长度
                                   int FFTWindowIdx,          // 窗函数索引 Index
                                   double InputData[],        // 输入的波形数据数组
                                   double SampleFrequency, // 采样时钟频率
                                   double *MainFrequency,   // 波形数据数组主频输出
                                   double *AmpSpectrum// 幅度谱分析数据数组输出
                                   );
```

入口参数：Dots：取 4096。

FFTWindowIdx：窗函数，取 3。

InputData：输入波形数组。

SampleFrequency：采样频率。

出口参数：*MainFrequency ：主频。

*OutputData：谱数。

3）功率谱分析

```
int Pack_SignalPowerSpectrum(int Dots,                      输入波形数据数组长度
                             int FFTWindowIdx,   //函数索引 Index
                             double InputData[],// 输入的波形数据数组
                             double SampleFrequency, // 采样时钟频率
                             double *MainFrequency, // 波形数据数组主频输出
                             double *AmpSpectrum // 功率谱分析数据数组输出
                             );
```

4）获取脉冲波形之特征参数

```
int   Pack_SignalPluseWaveformFeature(
            double *wavedata,    //波形数据
            int dots,            //波形点数
            double samplerate    //采样率
            double *topvale,     //顶部值
            double *basevalue,   //底部值
            double *overshoot,   //超调
            double *undershoot, //欠调
            double *width,       //脉宽
            double *Risetime,    //上升时间
            double *Falltime     //下降时间
            );
```

附件六、PCI8616 在 VC++6.0 下采集软件开发说明书

通过演示应用程序及其源代码，凡购买采集产品卡的用户能很快地掌握采集卡的原理、功能和工作流程，清楚地了解采集动态驱动库接口函数和调用方法，使用户能有效地自主开发应用程序。

并行数据采集卡提供对多路模拟信号的同步高速化，为避免多通道总线宽度不够而造成数据丢失，在采集卡上设计了一定容量的数据缓存池，便于高速采集数据实时存入。用户只需要操作采集动态库就可以实现采集卡的全部功能。

系列采集卡使用相同的采集动态库，继承了以前的产品特色，也为产品升级和功能的扩展做好了准备。演示程序可以驱动已生产出的各种类型的采集卡，并对采集卡上各个通道的属性参数分别设置，一次显示四个通道的波形。

本文主要介绍采集卡的工作流程、采集动态驱动库接口函数采集演示程序的使用方法。为适应用户需要，采集演示程序有 VCDEMO 版本，实现了连续采集和分组采集和其他的一些采集基本功能，参数设置方便简洁，一目了然，便于初次使用者进行开发。

因 PCI8616 是多功能卡，分成 2 个部分。

ARB+DDS+DO+5V 部分

1、卡自检初始化


```
void CPCI8616_ARBDlg::OnInit()
{
 CString    string;

if(PCI8616_DSO_VBSysInit(CardAddr,pSysInfo.Idnumber,pSysInfo.NewOffset,pSysInfo.CHA_GainTable,pSysInfo.CHB_GainTable))
    {       if(CardAddr[0])
            {
                MessageBox("PCI8616 卡初始化成功!");
                string.Format("卡序列号:""%s",pSysInfo.Idnumber);
                SetDlgItemText(IDC_EDIT1,string);
            }
        }
    else
        {

            MessageBox("没找到 PCI8616 卡\n 程序处于演示状态!!");
            string.Format("%s","无卡");
            SetDlgItemText(IDC_EDIT1,string);
        }
    }
```

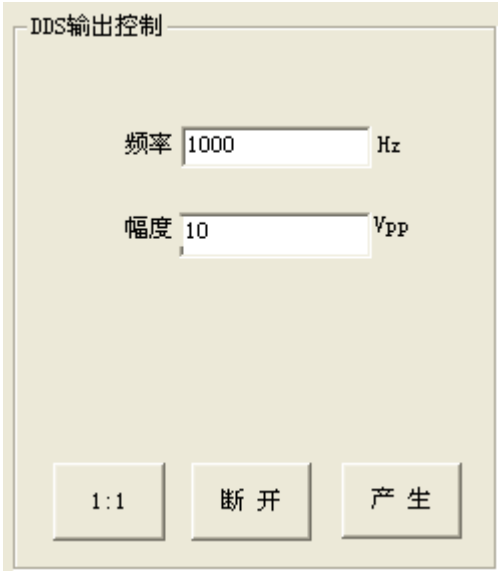
2、产生任意波形

设置好参数，按产生按钮，执行以下代码

```
.
void CPCI8616_ARBDlg::OnArbgen()
{
unsigned int tmp;
UpdateData(TRUE);
tmp=m_Type.GetCurSel();
PCI8616_ARB_GenAwgWave(tmp, m_freq,m_Amp, m_Offset, m_Duty,&mDots, mWaveData,
                                        &mDacFreqency );
}
void CPCI8616_ARBDlg::OnArbon()
{
      arbonflag=!arbonflag;
      if(arbonflag)
```

```
        {     m_Arbon.SetWindowText("输出");      }
        else
        {     m_Arbon.SetWindowText("断开");      }
    PCI8616_ARB_Kswitch(arbonflag,arbattrflag);
    }
    void CPCI8616_ARBDlg::OnArbattr()
    {
        arbattrflag=!arbattrflag;
        if(arbattrflag)
        {     m_ArbAttr.SetWindowText("10:1");     }
        else
        {     m_ArbAttr.SetWindowText("1:1");      }
    PCI8616_ARB_Kswitch(arbonflag,arbattrflag);
    }
```

3、产生 DDS 正玄波信号



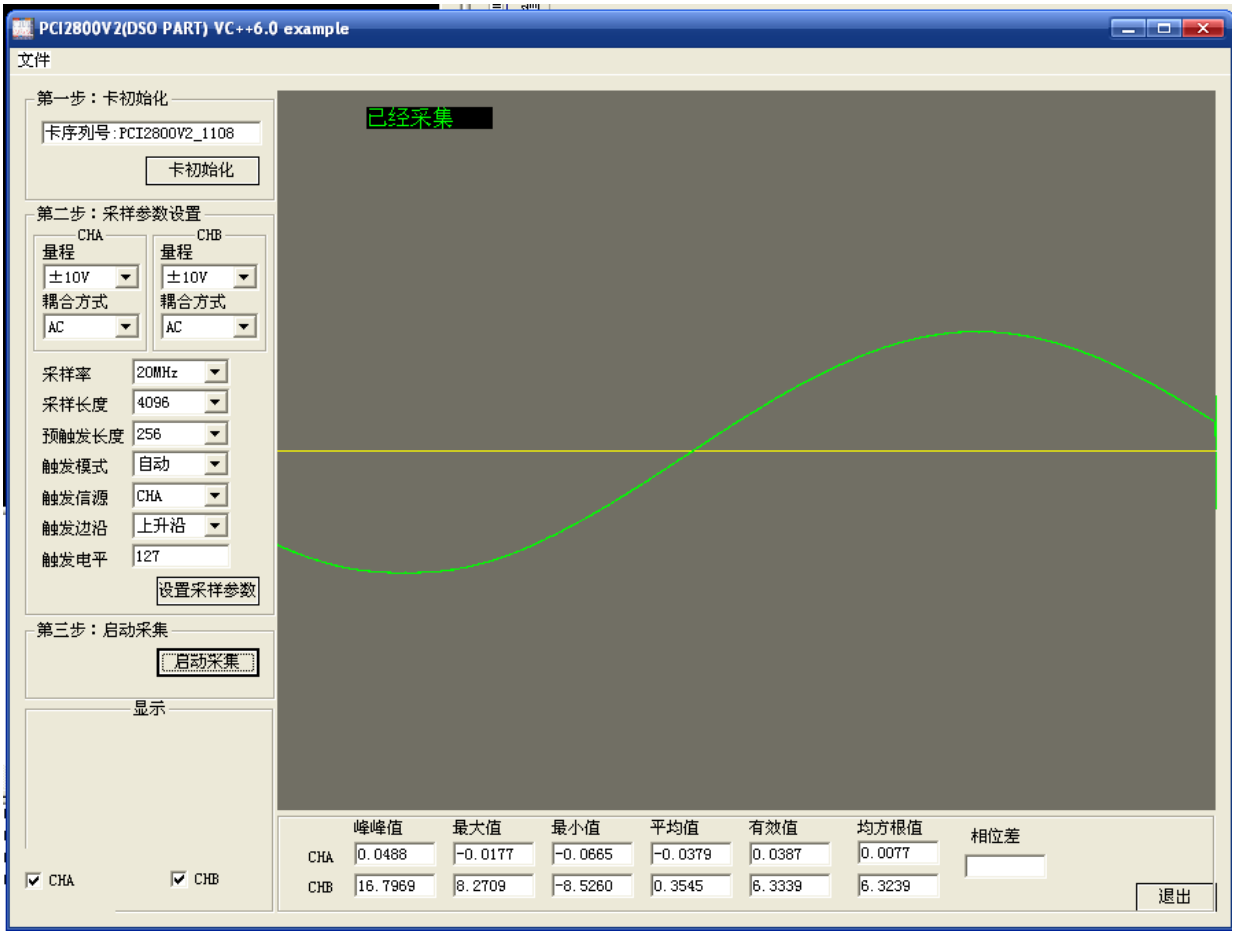
设置好参数，按产生按钮，执行以下代码

```
    void CPCI8616_ARBDlg::OnSetdo()
    {
        UpdateData(TRUE);
        PCI8616_WriteDO(CardAddr[0],m_doval);
    }
    void CPCI8616_ARBDlg::On5v()
    {
    m5vflag=!m5vflag;
        if(m5vflag)
        {     m_5v.SetWindowText("悬空");     }
        else
        {     m_5v.SetWindowText("+5V");      }
    PCI8616_DC5V_Kswitch(m5vflag);
    }

    void CPCI8616_ARBDlg::OnDdsgen()
    {
    UpdateData(TRUE);
    PCI8616_DDS_SingleToneMode(m_ddsamp,m_ddsfreq);
    }
```

4、8 路 DO　1 路 5V 控制


```
void CPCI8616_ARBDlg::OnSetdo()
{
      UpdateData(TRUE);
      PCI8616_WriteDO(CardAddr[0],m_doval);
}
void CPCI8616_ARBDlg::On5v()
{
m5vflag=!m5vflag;
      if(m5vflag)
      {     m_5v.SetWindowText("悬空");     }
      else
      {     m_5v.SetWindowText("+5V");      }
PCI8616_DC5V_Kswitch(m5vflag);
}
```

DAQ 数据采集部分



卡初始化

```
void CExampleDlg::OnInitialize()
{
CString    string;
if(PCI8616_DSO_VBSysInit(CardAddr,pSysInfo.Idnumber,pSysInfo.NewOffset,pSysInfo.CHA_GainTable,p
SysInfo.CHB_GainTable))
   {        if(CardAddr[0])
            {
                  MessageBox("PCI8200V2 卡初始化成功!");
                  string.Format("卡序列号:""%s",pSysInfo.Idnumber);
```

```
                SetDlgItemText(IDC_EDIT1,string);          }
    }
else
    {   MessageBox("没找到 PCI8200V2 卡\n 程序处于演示状态!!");
        string.Format("%s","无卡");
        SetDlgItemText(IDC_EDIT1,string);
    }
    PowerStatus=1;
}
```

设置采样参数

```
void CExampleDlg::OnInstallSample()
{
    // TODO: Add your control notification handler code here
    UpdateData();//更新数据
    unsigned int tmp,tmp1,tmp2,tmp3;
    //注：具体的参数请参见二次开发手册
    //设置采样率
    pSysInfo.SampleRateIdx=m_SampleIdx.GetCurSel();     //采样率
    tmp=pSysInfo.SampleRateIdx;
    //设置量程
    pSysInfo.gaina=m_GainCHA.GetCurSel();               //CHA 量程
    pSysInfo.gainb=m_GainCHB.GetCurSel();               //CHB 量程
    //设置耦合方式
    pSysInfo.couplecha=m_CoupleCHA.GetCurSel();         //CHA 耦合方式
    pSysInfo.couplechb=m_CoupleCHB.GetCurSel();         //CHB 耦合方式
    //设置触发参数
    pSysInfo.TrigMode=m_TrigMode.GetCurSel();             //触发方式
    pSysInfo.TrigSource=m_TrigSource.GetCurSel();       //触发源
    pSysInfo.TrigEdge=m_TrigEdge.GetCurSel();             //触发边沿

    tmp2=m_SampleLength.GetCurSel(); //采样长度
    pSysInfo.SampleLength=1024*pow(2,tmp2);
    tmp3=m_TrigPreIdx.GetCurSel();
    if(tmp3==0)
    {   pSysInfo.PreSampleLength=0;}
    else
    {
    pSysInfo.PreSampleLength=1024*pow(2,(tmp3-1));        //预触发长度
    }
    pSysInfo.TrigLevel=0xff-m_TrigLevel;                  //触发电平
   PCI8616_DSO_VBSetHardWare(CardAddr[0],
                                pSysInfo.SampleRateIdx,         //采样率序号
                                pSysInfo.SampleLength,        //采样长度
                                pSysInfo.PreSampleLength,    //预触发

                                        pSysInfo.gaina,
                                        pSysInfo.gainb,
                                        pSysInfo.couplecha,
                                        pSysInfo.couplechb,
                                        pSysInfo.TrigMode,
                                        pSysInfo.TrigEdge,          //触发边沿
                                        pSysInfo.TrigSource,        //触发源
                                        pSysInfo.TrigLevel      //触发电平
                                        );
}
```

启动采集

```
void CExampleDlg::OnMotivateCollect()
```

```
{
    // TODO: Add your control notification handler code here
    UpdateData();//更新数据
    //启动卡开始采样
    PCI8616_DSO_Acq(CardAddr[0]);
    //设置定时器
    SetTimer(1,500,NULL);
}
```

获取波形并显示

```
void CExampleDlg::OnTimer(UINT nIDEvent)
{
    int i,mDots;
    CString string,str1,str2,str;
    double Vpp[2],Vmax[2],Vmin[2],Vrms[2],Vmean[2],stdev[2];
    unsigned int FreqValue;
    double duty1,duty2,phase;

    //获取采样长度
    m_SampleLength.GetLBText(pSysInfo.SampleLength,string);
    mDots=atoi(string);
    if(PCI8616_DSO_PackData(CardAddr[0],mDots,addata1,addata2))
    {
        str2.Format("%s","已经采集          ");
        //取真实电压值,the last 128 dots is invalid data
        for(i=0;i<mDots-128;i++)
        {
            WaveData1[i]=addata1[i] * (1/pSysInfo.CHA_GainTable[pSysInfo.gaina]);
            WaveData2[i]=addata2[i] * (1/pSysInfo.CHB_GainTable[pSysInfo.gainb]);
        }
        //用户可增加自己的处理.....
Pack_SignalWaveformFeature(mDots-128,WaveData1,&Vpp[0],&Vmax[0],&Vmin[0],&Vrms[0],&Vmean[0]
,&duty1,&stdev[0])Pack_SignalWaveformFeature(mDots-128,WaveData2,&Vpp[1],&Vmax[1],&Vmin[1],&
Vrms[1],&Vmean[1],&duty2,&stdev[1])
        //显示结果
        m_MaxCHA = Vmax[0];      string.Format("%.4f",m_MaxCHA);
    SetDlgItemText(IDC_MaxValueCHA,string);
        m_MaxCHB = Vmax[1];      string.Format("%.4f",m_MaxCHB);
    SetDlgItemText(IDC_MaxValueCHB,string);
        m_MinCHA = Vmin[0]; string.Format("%.4f",m_MinCHA);
    SetDlgItemText(IDC_MinValueCHA,string);
        m_MinCHB = Vmin[1]; string.Format("%.4f",m_MinCHB);
    SetDlgItemText(IDC_MinValueCHB,string);
        m_PeakCHA = Vpp[0];  string.Format("%.4f",m_PeakCHA);
    SetDlgItemText(IDC_PeakValueCHA,string);
        m_PeakCHB = Vpp[1];  string.Format("%.4f",m_PeakCHB);
    SetDlgItemText(IDC_PeakValueCHB,string);
        m_MeanCHA = Vmean[0];    string.Format("%.4f",m_MeanCHA);
    SetDlgItemText(IDC_MeanValueCHA,string);
        m_MeanCHB = Vmean[1];    string.Format("%.4f",m_MeanCHB);
    SetDlgItemText(IDC_MeanValueCHB,string);
        m_RMSCHA = Vrms[0];      string.Format("%.4f",m_RMSCHA);
    SetDlgItemText(IDC_RMSValueCHA,string);
        m_RMSCHB = Vrms[1];      string.Format("%.4f",m_RMSCHB);
    SetDlgItemText(IDC_RMSValueCHB,string);
        m_stdevcha = stdev[0];  string.Format("%.4f",m_stdevcha);
    SetDlgItemText(IDC_STDEVCHA,string);
        m_stdevchb = stdev[1];  string.Format("%.4f",m_stdevchb);
    SetDlgItemText(IDC_STDEVCHB,string);
```

```
    }
    else
    {
        str2.Format("%s","正在采集...      ");
    }
    //刷新
    Invalidate(NULL);
    UpdateWindow();
    OnCheckCha();
    OnCheckChb();
    str.Format("%s%s",str2,str1);
    //str=str2+str1;
    SetText(str,RGB(0,0,0),RGB(0,255,0),IDC_STATIC_Sign);

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

2.1 显示波形数据转化

在 VC++6.0 中，在屏幕上画点是按照屏幕分辨率来作为坐标表示的，因此从采集卡读到的波形数据不能直接用于显示，必须先通过算法把波形数据转换成适合显示的数组（DrawBuffer），这个二维数组第一个参数（行）表示通道，第二个参数的点数有 x，y 坐标，就是在屏幕上画点的坐标，所以 DrawBuffer 中的数据跟真实数据没关系，DrawBuffer 仅仅是为了最后的画图，程序就是通过画 DrawBuffer 来实现波形显示功能的。

2.2 启动采集

一旦启动采集，由于硬件特性，采集卡上的所有通道都开始采集，而不是只有选择或者是设置了的那个通道进行采集。

2.3 采样设置

每个通道单独设置量程、耦合方式；

采样率、采样长度与通道无关，一旦设定好每个通道这三个属性都一样。

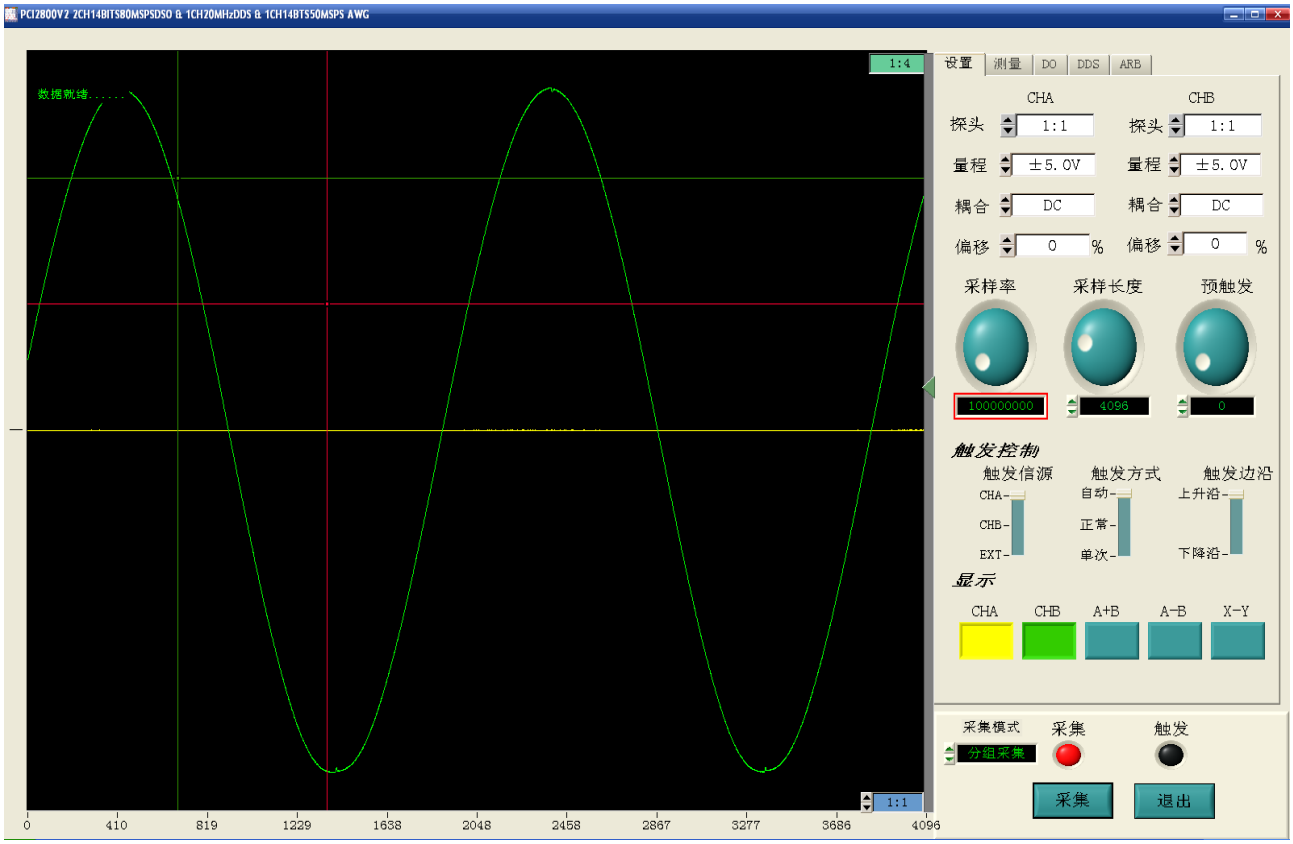
附件七、PCI8616 在 CVI 2010 下采集软件开发说明书

通过演示应用程序及其源代码，凡购买采集产品卡的用户能很快地掌握采集卡的原理、功能和工作流程，清楚地了解采集动态驱动库接口函数和调用方法，使用户能有效地自主开发应用程序。

并行数据采集卡提供对多路模拟信号的同步高速化，为避免多通道总线宽度不够而造成数据丢失，在采集卡上设计了一定容量的数据缓存池，便于高速采集数据实时存入。用户只需要操作采集动态库就可以实现采集卡的全部功能。

系列采集卡使用相同的采集动态库，继承了以前的产品特色，也为产品升级和功能的扩展做好了准备。演示程序可以驱动已生产出的各种类型的采集卡，并对采集卡上各个通道的属性参数分别设置，一次显示四个通道的波形。

本文主要介绍采集卡的工作流程、采集动态驱动库接口函数采集演示程序的使用方法。为适应用户需要，采集演示程序版本，实现了连续采集和分组采集和其他的一些采集基本功能，参数设置方便简洁，一目了然，便于初次使用者进行开发。

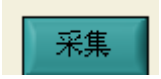


1．卡自检初始化

```
void Login_PCI2800(void)
{
   int i,listcount;
   AutoScreen(panel_Handle);
   cha_color=VAL_GREEN;
   chb_color=VAL_YELLOW;

if(!PCI8616_DSO_VBSysInit(CardAddress,pSysInfo.Idnumber,pSysInfo.NewOffset,pSysInfo.CHA_GainTable,pSysInfo.CHB_GainTable))
        {       MessagePopup("info","No found PCI8616!");
            AutoCheckStatus=0;
            SetPanelAttribute (panel_Handle, ATTR_TITLE, "PCI8616 2CH14BITS80MSPSDSO & 1CH20MHzDDS & 1CH14BTS50MSPS AWG");
            pSysInfo.SampleLength=4096;
            hDevice=CardAddress[0];
        }
   else{AutoCheckStatus=1;}
     SetPanelAttribute (panel_Handle, ATTR_TITLE, "PCI8616 2CH14BITS80MSPSDSO & 1CH20MHzDDS & 1CH14BTS50MSPS AWG");
   AutoCheckStatus=1;
       hDevice=CardAddress[0];
   PCI8616_DDS_Powerdown();
   PCI8616_DDS_Kswitch(0,OFF);
   PCI8616_ARB_Kswitch(OFF,OFF);
   PCI8616_DC5V_Kswitch(OFF);
        if(AutoCheckStatus)
   {
        //删除已采样率例表
        GetNumListItems (SetHandle,SETPANEL_SampleRate,&listcount);
        DeleteListItem (SetHandle,SETPANEL_SampleRate,0,-1);
        for(i=0;i<15;i++) {InsertListItem (SetHandle,SETPANEL_SampleRate, i,"",0);}
```

```
            GetCtrlIndex(SetHandle,SETPANEL_SamLength,&pSysInfo.SampleLength);
            //给量程列表重新赋值
        GetNumListItems (SetHandle,SETPANEL_GAINCHA,&listcount);
        DeleteListItem (SetHandle,SETPANEL_GAINCHA,0,-1);
        GetNumListItems (SetHandle,SETPANEL_GAINCHB,&listcount);
        DeleteListItem (SetHandle,SETPANEL_GAINCHB,0,-1);
           for(i=0;i<8;i++)
                                    {
                                    InsertListItem (SetHandle,SETPANEL_GAINCHA,
i,PCI8616_RangeText[i],0);
                                    InsertListItem (SetHandle,SETPANEL_GAINCHB,
i,PCI8616_RangeText[i],0);
                                    }

                                    SetCtrlIndex(SetHandle,SETPANEL_GAINCHA,5);
                                    SetCtrlIndex(SetHandle,SETPANEL_GAINCHB,5);
                                    SetCtrlIndex(SetHandle,SETPANEL_SamLength,2);
                                    SetCtrlIndex(SetHandle,SETPANEL_PreLength,0);
                                    SetCtrlIndex(SetHandle,SETPANEL_SampleRate,6);
                                    Kaxis=4;

                                    pSysInfo.AD_ClkNum=16;
                                    for(i=0;i<16;i++)    {pSysInfo.AD_ClkTable[i]=
PCI8616_SampleRateClk[i];}

                                     mWAVEMIN=-1.0;
                                     mWAVEMAX=1.0;

                                    //初始化面板结束

                                   DeltaTime=0.5;

          }

        DisplayPanel(panel_Handle);
    }
```

2.设置采集参数，启动采集



执行以下代码

```
    int CVICALLBACK Sample (int panel, int control, int event,
            void *callbackData, int eventData1, int eventData2)
    {

        unsigned int sampleidx;
        unsigned char tmp,tmpmax;
        int sub,mode;
        double mpretrigtime;
        switch (event)
            {
            case EVENT_COMMIT:
                RunStatus=!RunStatus;
                if(PlayThreadStatus == RUNING)
                                {
                                 PlayCounter=0;
                                 PlayThreadStatus = STOP;
```

```
                                    Delay(0.5);
                                    CmtWaitForThreadPoolFunctionCompletion
(DEFAULT_THREAD_POOL_HANDLE,threadFunctionId2,OPT_TP_PROCESS_EVENTS_WHILE_WAIT
ING);
                                    CmtReleaseThreadPoolFunctionID
(DEFAULT_THREAD_POOL_HANDLE, threadFunctionId2);
                                  }



                    if(RunStatus)
                                  {

                                    Kaxis=4;
                                   SetAxisRange (panel_Handle, PANEL_WAVE, VAL_MANUAL,
0,4096,VAL_MANUAL,(-1.0)*myscale,myscale);
                                   SetCtrlVal(panel_Handle,PANEL_Scale,Kaxis);
                                  // Change_Wave_XAxis_Item();

                                   SetCtrlAttribute (panel_Handle, PANEL_START, ATTR_CTRL_VAL,1);
                                   SetCtrlAttribute (panel_Handle, PANEL_START, ATTR_LABEL_TEXT,"采集");
                                   SetCtrlAttribute (panel_Handle, PANEL_STATUS2, ATTR_VISIBLE, 1);
                                   SetCtrlVal(panel_Handle,PANEL_LED0,1);

                                   GetCtrlIndex(SetHandle,SETPANEL_SampleRate,&pSysInfo.SampleRateIdx);
                                  GetCtrlIndex(SetHandle,SETPANEL_SamLength,&pSysInfo.SampleLength);
                                   SetCtrlIndex(SetHandle,SETPANEL_TSamLength,pSysInfo.SampleLength);
                                   GetCtrlVal(SetHandle,SETPANEL_PreLength,&pSysInfo.PreSampleLength);
                                   GetCtrlIndex(SetHandle,SETPANEL_GAINCHA,&pSysInfo.gaina);
                                   GetCtrlIndex(SetHandle,SETPANEL_GAINCHB,&pSysInfo.gainb);
                                    //Couple
                                   GetCtrlVal(SetHandle,SETPANEL_COUPLECHA,&pSysInfo.couplecha);
                                   GetCtrlVal(SetHandle,SETPANEL_COUPLECHB,&pSysInfo.couplechb);

                                   //Trig Para
                                   GetCtrlIndex(SetHandle,SETPANEL_TrigMode,&pSysInfo.TrigMode);
                                         if(pSysInfo.TrigMode==2)
                                                       {pSysInfo.TrigMode=1;mTRIGMODE=ONCETRIG;}
                                      else
                                                       {mTRIGMODE=LOOPTRIG;}
                                   GetCtrlIndex(SetHandle,SETPANEL_TrigSource,&pSysInfo.TrigSource);
                                   GetCtrlIndex(SetHandle,SETPANEL_TrigEdge,&pSysInfo.TrigEdge);
                                   GetCtrlVal(SetHandle,SETPANEL_PreLength,&pSysInfo.PreSampleLength);
                                   GetCtrlVal(panel_Handle,PANEL_TRIGLEVEL,&pSysInfo.TrigLevel);
                                   pSysInfo.TrigLevel=0xff-pSysInfo.TrigLevel;
                                   StartAcq();
                                  if(DaqThreadStatus == STOP)
                                         {
                                           CmtScheduleThreadPoolFunction
(DEFAULT_THREAD_POOL_HANDLE,DaqThreadFunction, NULL,&DaqThreadFunctionId);
                                           DaqThreadStatus=RUNING;
                                         }

                                         }
                                   else
                                   {
                                   SetCtrlAttribute (panel_Handle, PANEL_START, ATTR_CTRL_VAL,0);
                                   SetCtrlAttribute (panel_Handle, PANEL_START, ATTR_LABEL_TEXT,"暂停");
                                   SetCtrlVal(panel_Handle,PANEL_LED0,0);
                                   SetCtrlAttribute (panel_Handle, PANEL_STATUS2, ATTR_VISIBLE, 0);
```

```
                        outpd(CardAddress[0]+0x44,0x00);
                if(DaqThreadStatus == RUNING)
                        {
                        DaqThreadStatus = STOP;
                        Delay(0.5);
                        CmtWaitForThreadPoolFunctionCompletion
(DEFAULT_THREAD_POOL_HANDLE,DaqThreadFunctionId,OPT_TP_PROCESS_EVENTS_WHILE_W
AITING);
                        CmtReleaseThreadPoolFunctionID (DEFAULT_THREAD_POOL_HANDLE,
DaqThreadFunctionId);
                        }

                                }


                break;
                }
        return 0;
    }
```

3.数据采集线程

```
    /* First thread function */
    static int CVICALLBACK DaqThreadFunction (void *functionData)
    {


        while (DaqThreadStatus==RUNING) {
                if(RunStatus==1)
                   {
                   CmtGetLock(lock);

                   pSysInfo.SampleLength=pSysInfo.SampleLength;
                   status=0;
                   if(samplemode==0)
                                {

status=PCI8616_DSO_PackData2(CardAddress[0],pSysInfo.SampleLength,addata1,addata2);
                            //     status=PCI8616_DSO_PackData(0,pSysInfo.SampleLength,addata1,addata2);
                                }
                   else
                   {
                   pSysInfo.SampleLength=2048;
                   status=PCI8616_DSO_PackFIFOData(CardAddress[0],0,addata1,addata2);
                   }
                   if(status==0) {SetCtrlVal(panel_Handle, PANEL_STATUS2,"等待数据中.....");}
                     else
                                {
                                 SetCtrlVal(panel_Handle, PANEL_STATUS2,"数据就绪.....");
                                 ledflag=!ledflag;
                                 SetCtrlVal(panel_Handle,PANEL_LED1,ledflag);
                                 WaveChanged_f=1;
                                 ReDraw_WaveForm();
                                 WaveChanged_f=0;
                                }
                   CmtReleaseLock(lock);
                   }

                   if(samplemode==0)    { Delay(0.2);}
                   else                 {Delay(0.001);    }
                                }
        return 0;
    }
```

二、ARB + DDS +8DO

DDS 按钮 


```
int CVICALLBACK SetSingleTonePara (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{
    unsigned int kstatus;
    double freq,amp;
    switch (event)
        {
        case EVENT_COMMIT:
            GetCtrlVal(ddsHandle,DDSPANEL_AMP,&amp);
            GetCtrlVal(ddsHandle,DDSPANEL_FREQ,&freq);
            PCI8616_DDS_SingleToneMode(amp,freq);
        break;
        }
    return 0;
}
```

ARB 模式产生波形

```
int CVICALLBACK SendWave (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{
    int ch,mLoopMode,mfcidx,WaveNumbers,mTrigSource,mWaveNumbers;
    int offset[2];
    switch (event)
        {
        case EVENT_COMMIT:
            arbRunStatus=!arbRunStatus;
            GetCtrlVal(arbHandle,ARBPANEL_ARBATTR,&mArbAttr);
            if(arbRunStatus)
               {
               SetCtrlAttribute (arbHandle, ARBPANEL_RUN, ATTR_LABEL_TEXT,"输出");
               PCI8616_ARB_Kswitch(1,mArbAttr);
              }
               else
               {
                SetCtrlAttribute (arbHandle, ARBPANEL_RUN, ATTR_LABEL_TEXT,"断开");
                PCI8616_ARB_Kswitch(0,mArbAttr);

                }
            break;
        }
    return 0;
}
```

8 路 DO 控制

```
int CVICALLBACK diowrite (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{
    unsigned int i,j;
    unsigned int D[8],Data1;

    switch (event)
        {
        case EVENT_COMMIT:
```

```
                    GetCtrlVal(Dioda_Handle,DIODAPANEL_DO0,&D[0]);
                    GetCtrlVal(Dioda_Handle,DIODAPANEL_DO1,&D[1]);
                    GetCtrlVal(Dioda_Handle,DIODAPANEL_DO2,&D[2]);
                    GetCtrlVal(Dioda_Handle,DIODAPANEL_DO3,&D[3]);
                    GetCtrlVal(Dioda_Handle,DIODAPANEL_DO4,&D[4]);
                    GetCtrlVal(Dioda_Handle,DIODAPANEL_DO5,&D[5]);
                    GetCtrlVal(Dioda_Handle,DIODAPANEL_DO6,&D[6]);
                    GetCtrlVal(Dioda_Handle,DIODAPANEL_DO7,&D[7]);
                    Data1=0;
                    for(i=0;i<8;i++)
                    {
                     j=D[i]<<i;
                     Data1=Data1 | j;
                    }

                   PCI8616_WriteDO(CardAddress[0],Data1);
                   break;
                }
          return 0;

     }
```

卡初始化

```
void CExampleDlg::OnInitialize()
{
CString    string;
if(PCI8616_DSO_VBSysInit(CardAddr,pSysInfo.Idnumber,pSysInfo.NewOffset,pSysInfo.CHA_GainTable,p
SysInfo.CHB_GainTable))
   {       if(CardAddr[0])
           {
                MessageBox("PCI8200V2 卡初始化成功!");
                string.Format("卡序列号:""%s",pSysInfo.Idnumber);
                SetDlgItemText(IDC_EDIT1,string);          }
     }
else
     {     MessageBox("没找到 PCI8200V2 卡\n 程序处于演示状态!!");
           string.Format("%s","无卡");
           SetDlgItemText(IDC_EDIT1,string);
     }
     PowerStatus=1;
}
```

设置采样参数

```
void CExampleDlg::OnInstallSample()
{
     // TODO: Add your control notification handler code here
     UpdateData();//更新数据
     unsigned int tmp,tmp1,tmp2,tmp3;
     //注：具体的参数请参见二次开发手册
     //设置采样率
     pSysInfo.SampleRateIdx=m_SampleIdx.GetCurSel();     //采样率
     tmp=pSysInfo.SampleRateIdx;
     //设置量程
     pSysInfo.gaina=m_GainCHA.GetCurSel();               //CHA 量程
     pSysInfo.gainb=m_GainCHB.GetCurSel();               //CHB 量程
     //设置耦合方式
     pSysInfo.couplecha=m_CoupleCHA.GetCurSel();         //CHA 耦合方式
     pSysInfo.couplechb=m_CoupleCHB.GetCurSel();         //CHB 耦合方式
     //设置触发参数
```

```
    pSysInfo.TrigMode=m_TrigMode.GetCurSel();              //触发方式
    pSysInfo.TrigSource=m_TrigSource.GetCurSel();        //触发源
    pSysInfo.TrigEdge=m_TrigEdge.GetCurSel();             //触发边沿

    tmp2=m_SampleLength.GetCurSel(); //采样长度
    pSysInfo.SampleLength=1024*pow(2,tmp2);
    tmp3=m_TrigPreIdx.GetCurSel();
    if(tmp3==0)
    {    pSysInfo.PreSampleLength=0;}
    else
    {
    pSysInfo.PreSampleLength=1024*pow(2,(tmp3-1));        //预触发长度
    }
    pSysInfo.TrigLevel=0xff-m_TrigLevel;                  //触发电平
   PCI8616_DSO_VBSetHardWare(CardAddr[0],
                                    pSysInfo.SampleRateIdx,        //采样率序号
                                    pSysInfo.SampleLength,       //采样长度
                                    pSysInfo.PreSampleLength,    //预触发

                                        pSysInfo.gaina,
                                        pSysInfo.gainb,
                                        pSysInfo.couplecha,
                                        pSysInfo.couplechb,
                                        pSysInfo.TrigMode,
                                        pSysInfo.TrigEdge,          //触发边沿
                                        pSysInfo.TrigSource,        //触发源
                                        pSysInfo.TrigLevel     //触发电平
                                        );
}
```

启动采集

```
void CExampleDlg::OnMotivateCollect()
{
    // TODO: Add your control notification handler code here
    UpdateData();//更新数据
    //启动卡开始采样
    PCI8616_DSO_Acq(CardAddr[0]);
    //设置定时器
    SetTimer(1,500,NULL);
}
```

获取波形并显示

```
void CExampleDlg::OnTimer(UINT nIDEvent)
{
    int i,mDots;
    CString string,str1,str2,str;
    double Vpp[2],Vmax[2],Vmin[2],Vrms[2],Vmean[2],stdev[2];
    unsigned int FreqValue;
    double duty1,duty2,phase;

    //获取采样长度
    m_SampleLength.GetLBText(pSysInfo.SampleLength,string);
    mDots=atoi(string);
    if(PCI8616_DSO_PackData(CardAddr[0],mDots,addata1,addata2))
    {
        str2.Format("%s","已经采集          ");
        //取真实电压值,the last 128 dots is invalid data
        for(i=0;i<mDots-128;i++)
        {
```

```
                WaveData1[i]=addata1[i] * (1/pSysInfo.CHA_GainTable[pSysInfo.gaina]);
                WaveData2[i]=addata2[i] * (1/pSysInfo.CHB_GainTable[pSysInfo.gainb]);
            }
            //用户可增加自己的处理.....
Pack_SignalWaveformFeature(mDots-128,WaveData1,&Vpp[0],&Vmax[0],&Vmin[0],&Vrms[0],&Vmean[0]
,&duty1,&stdev[0])Pack_SignalWaveformFeature(mDots-128,WaveData2,&Vpp[1],&Vmax[1],&Vmin[1],&
Vrms[1],&Vmean[1],&duty2,&stdev[1])
            //显示结果
            m_MaxCHA = Vmax[0];       string.Format("%.4f",m_MaxCHA);
        SetDlgItemText(IDC_MaxValueCHA,string);
            m_MaxCHB = Vmax[1];       string.Format("%.4f",m_MaxCHB);
        SetDlgItemText(IDC_MaxValueCHB,string);
            m_MinCHA = Vmin[0]; string.Format("%.4f",m_MinCHA);
        SetDlgItemText(IDC_MinValueCHA,string);
            m_MinCHB = Vmin[1]; string.Format("%.4f",m_MinCHB);
        SetDlgItemText(IDC_MinValueCHB,string);
            m_PeakCHA = Vpp[0];  string.Format("%.4f",m_PeakCHA);
        SetDlgItemText(IDC_PeakValueCHA,string);
            m_PeakCHB = Vpp[1];  string.Format("%.4f",m_PeakCHB);
        SetDlgItemText(IDC_PeakValueCHB,string);
            m_MeanCHA = Vmean[0];    string.Format("%.4f",m_MeanCHA);
        SetDlgItemText(IDC_MeanValueCHA,string);
            m_MeanCHB = Vmean[1];    string.Format("%.4f",m_MeanCHB);
        SetDlgItemText(IDC_MeanValueCHB,string);
            m_RMSCHA = Vrms[0];       string.Format("%.4f",m_RMSCHA);
        SetDlgItemText(IDC_RMSValueCHA,string);
            m_RMSCHB = Vrms[1];       string.Format("%.4f",m_RMSCHB);
        SetDlgItemText(IDC_RMSValueCHB,string);
            m_stdevcha = stdev[0];  string.Format("%.4f",m_stdevcha);
        SetDlgItemText(IDC_STDEVCHA,string);
            m_stdevchb = stdev[1];  string.Format("%.4f",m_stdevchb);
        SetDlgItemText(IDC_STDEVCHB,string);

        }
        else
        {
            str2.Format("%s","正在采集...       ");
        }
        //刷新
        Invalidate(NULL);
        UpdateWindow();
        OnCheckCha();
        OnCheckChb();
        str.Format("%s%s",str2,str1);
        //str=str2+str1;
        SetText(str,RGB(0,0,0),RGB(0,255,0),IDC_STATIC_Sign);

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

2.1 显示波形数据转化

2.2 启动采集

一旦启动采集，由于硬件特性，采集卡上的所有通道都开始采集，而不是只有选择或者是设置了的那个通道进行采集。

2.3 采样设置

每个通道单独设置量程、耦合方式；

采样率、采样长度与通道无关，一旦设定好每个通道这三个属性都一样。

附件八、PCI8616 在 LABVIEW2009 下采集软件开发说明书

通过演示应用程序及其源代码，凡购买采集产品卡的用户能很快地掌握采集卡的原理、功能和工作流程，清楚地了解采集动态驱动库接口函数和调用方法，使用户能有效地自主开发应用程序。

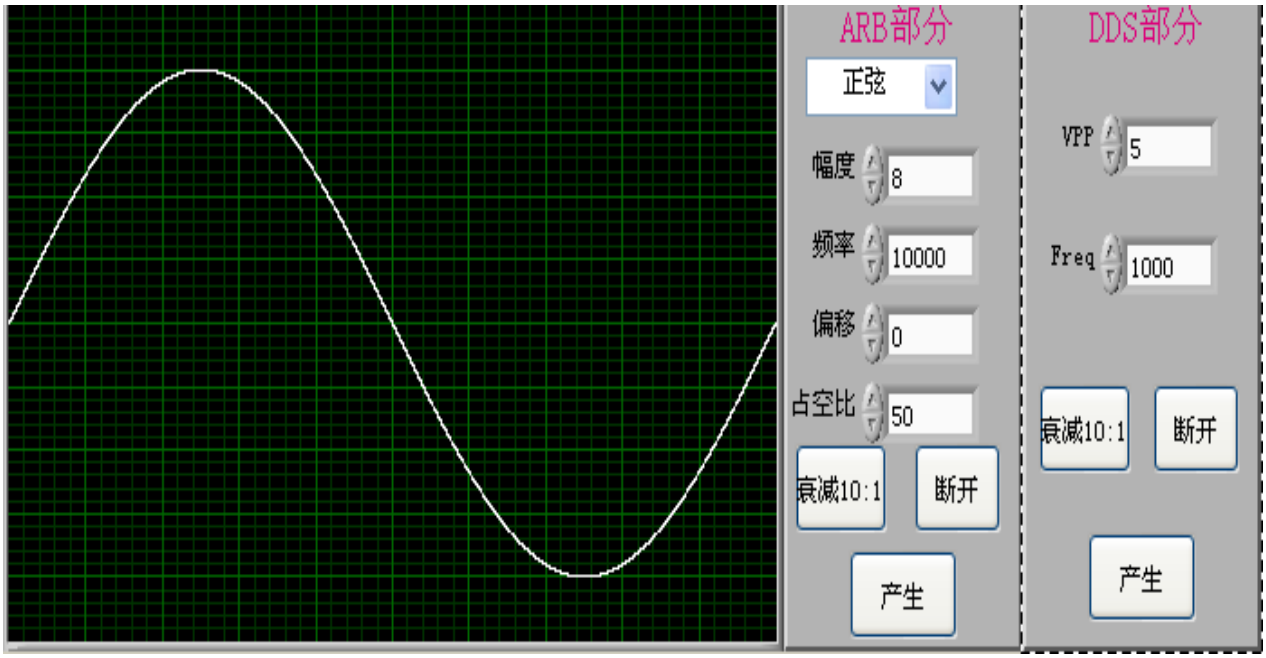
并行数据采集卡提供对多路模拟信号的同步高速化，为避免多通道总线宽度不够而造成数据丢失，在采集卡上设计了一定容量的数据缓存池，便于高速采集数据实时存入。用户只需要操作采集动态库就可以实现采集卡的全部功能。

系列采集卡使用相同的采集动态库，继承了以前的产品特色，也为产品升级和功能的扩展做好了准备。演示程序可以驱动已生产出的各种类型的采集卡，并对采集卡上各个通道的属性参数分别设置，一次显示 2 个通道的波形。

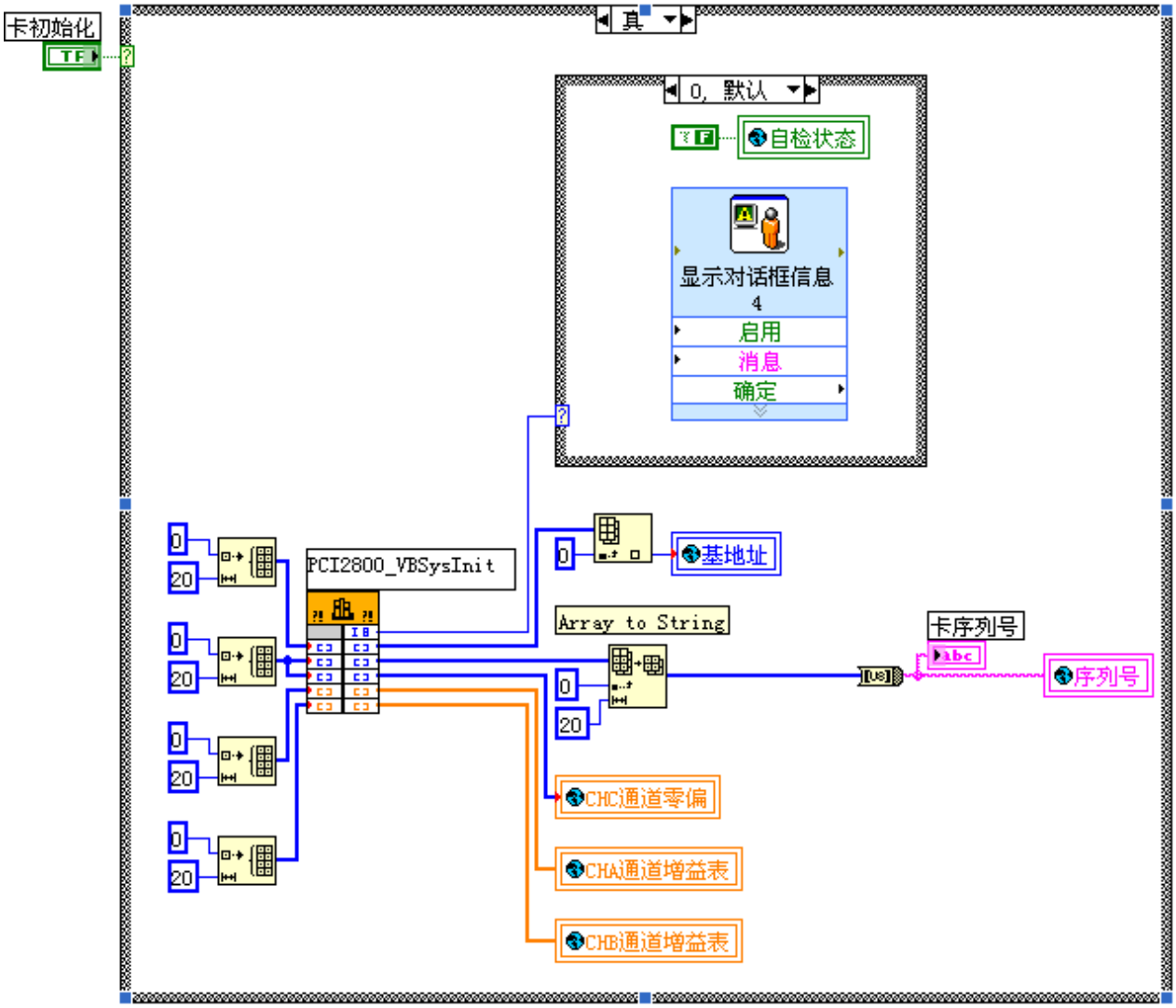
本文主要介绍采集卡的工作流程、采集动态驱动库接口函数采集演示程序的使用方法。为适应用户需要，采集演示程序有版本，实现了连续采集和分组采集和其他的一些采集基本功能，参数设置方便简洁，一目了然，便于初次使用者进行开发。

因 PCI8616 是多功能卡，分成 2 个部分。

ARB+DDS 部分

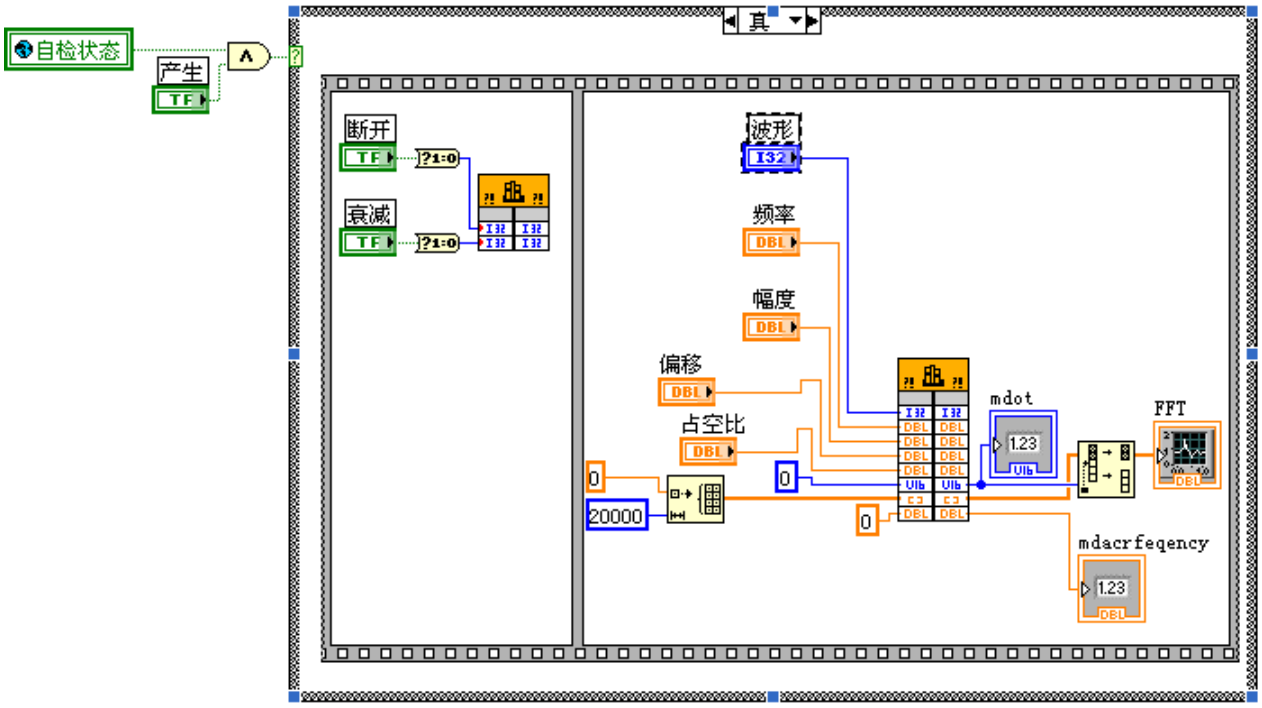


1、卡自检初始化



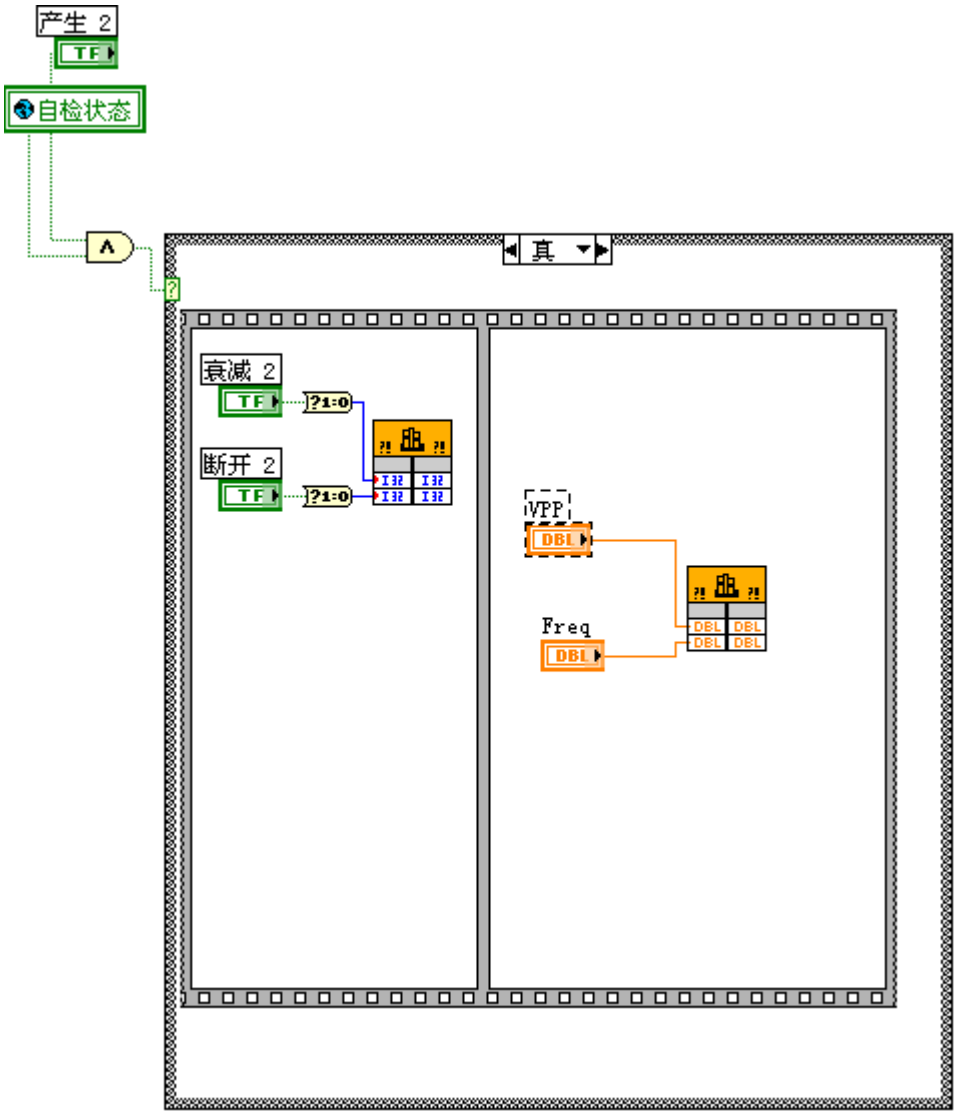
2、产生任意波形

3、设置好参数，按产生按钮，执行以下代码

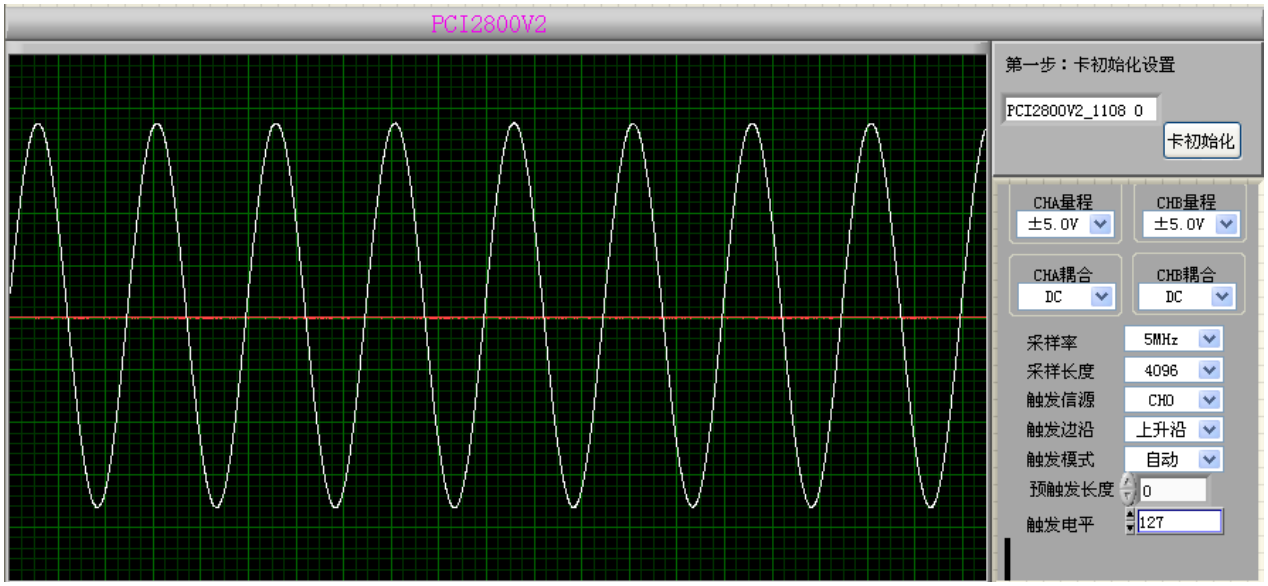


4、产生 DDS 正玄波信号

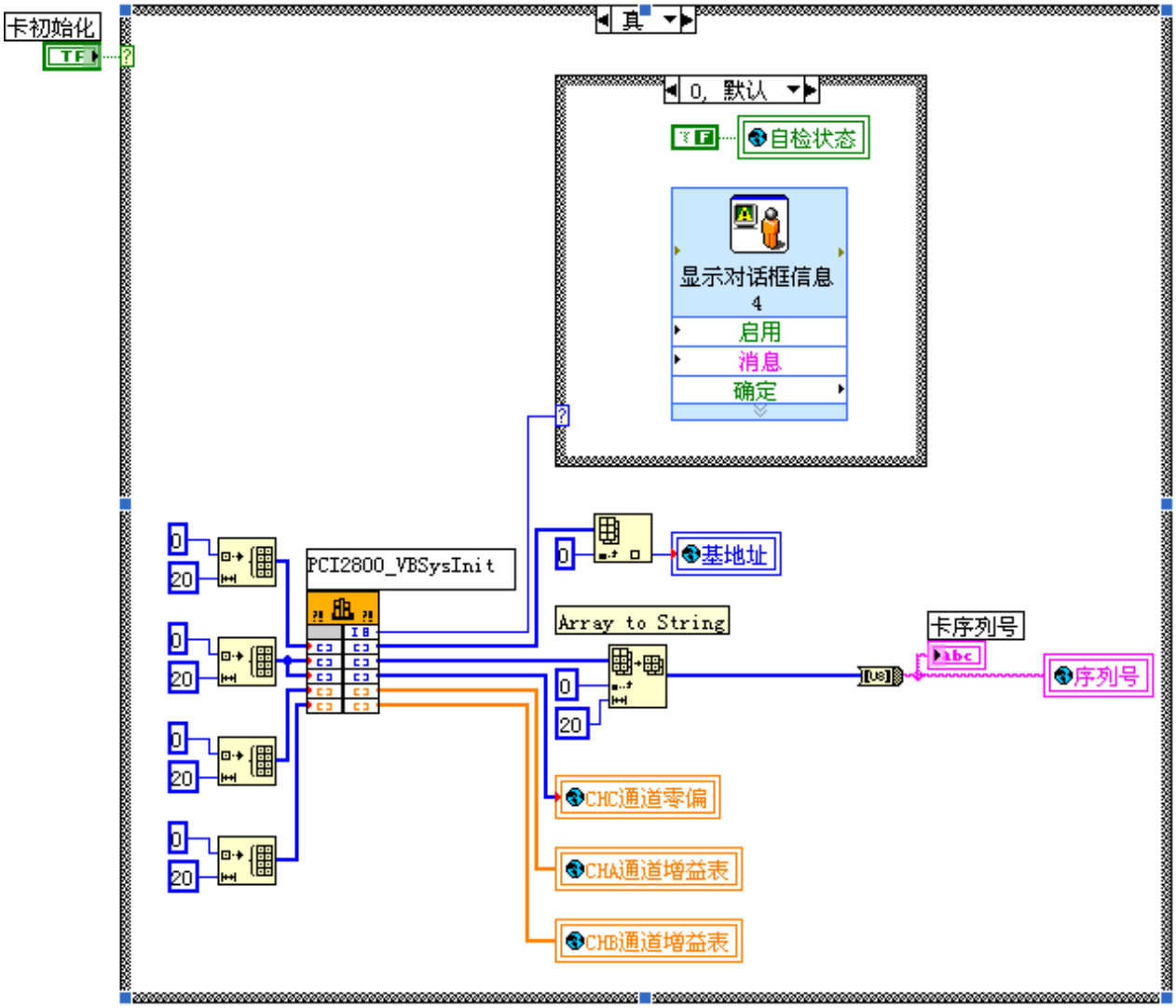
设置好参数，按产生按钮，执行以下代码



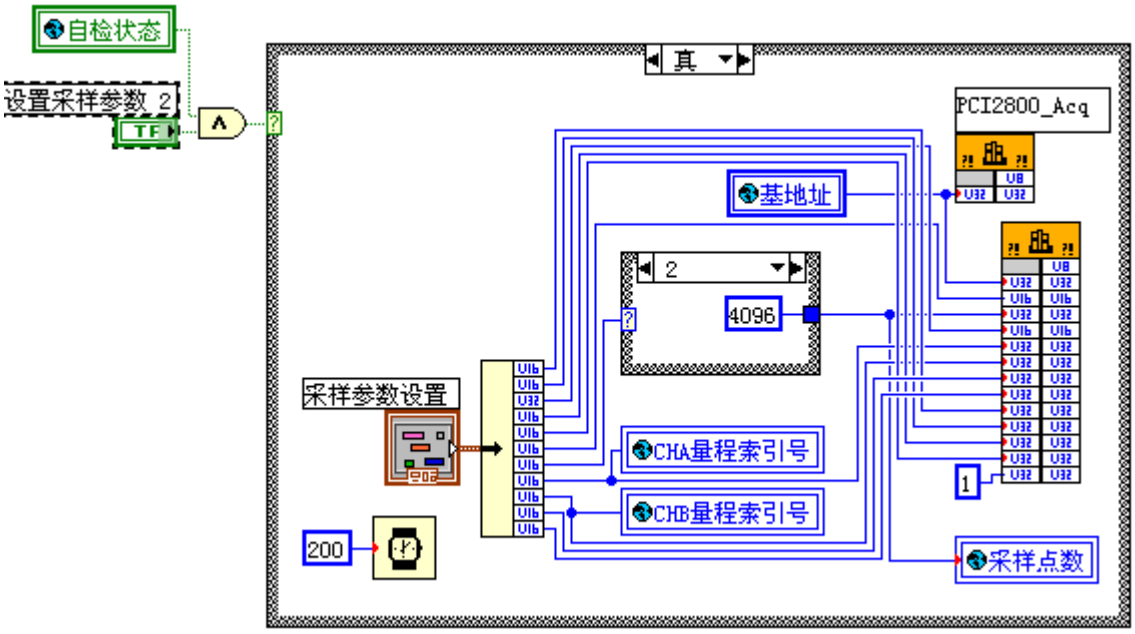
DAQ 数据采集部分



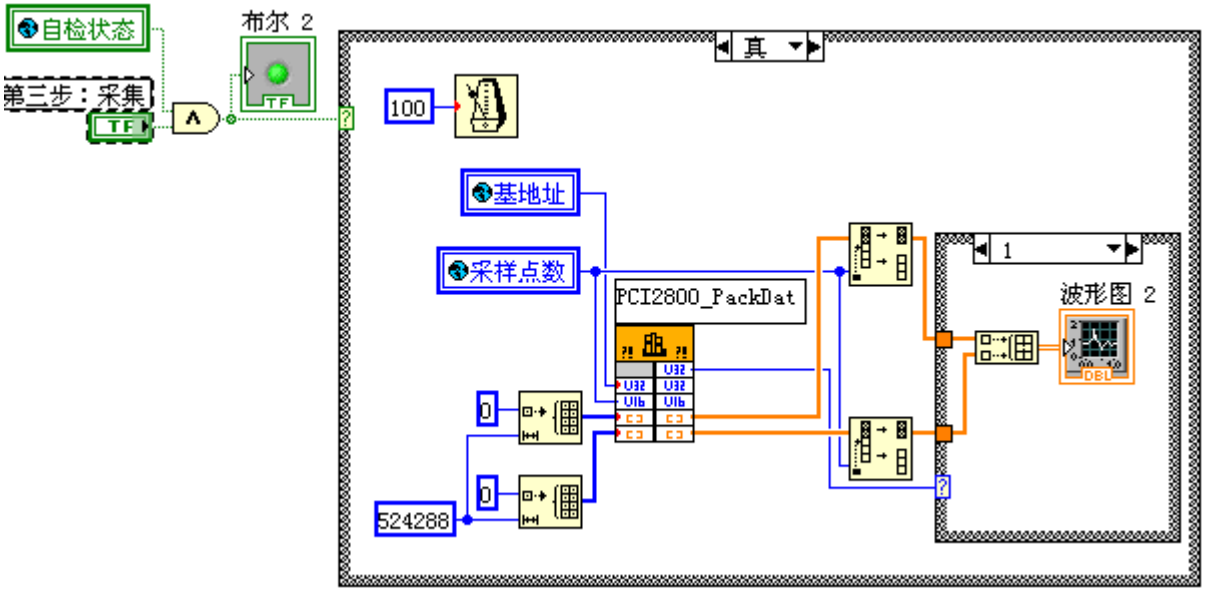
5、卡初始化


卡初始化
TF
真
0, 默认
自检状态
显示对话框信息
4
启用
消息
确定
PCI2800_VBSysInit
基地址
Array to String
卡序列号
abc
序列号
CHC通道零偏
CHA通道增益表
CHB通道增益表

6、设置采样参数


自检状态
设置采样参数 2
TF
A
真
PCI2800_Acq
基地址
2
4096
采样参数设置
CHA量程索引号
CHB量程索引号
200
1
采样点数

7、启动采集



2.1 显示波形数据转化

2.2 启动采集

一旦启动采集，由于硬件特性，采集卡上的所有通道都开始采集，而不是只有选择或者是设置了的那个通道进行采集。

2.3 采样设置

每个通道单独设置量程、耦合方式；

采样率、采样长度与通道无关，一旦设定好每个通道这三个属性都一样。

附件九　PCI8616 在 VB6.0 下采集软件开发说明书

通过演示应用程序及其源代码，凡购买采集产品卡的用户能很快地掌握采集卡的原理、功能和工作流程，清楚地了解采集动态驱动库接口函数和调用方法，使用户能有效地自主开发应用程序。

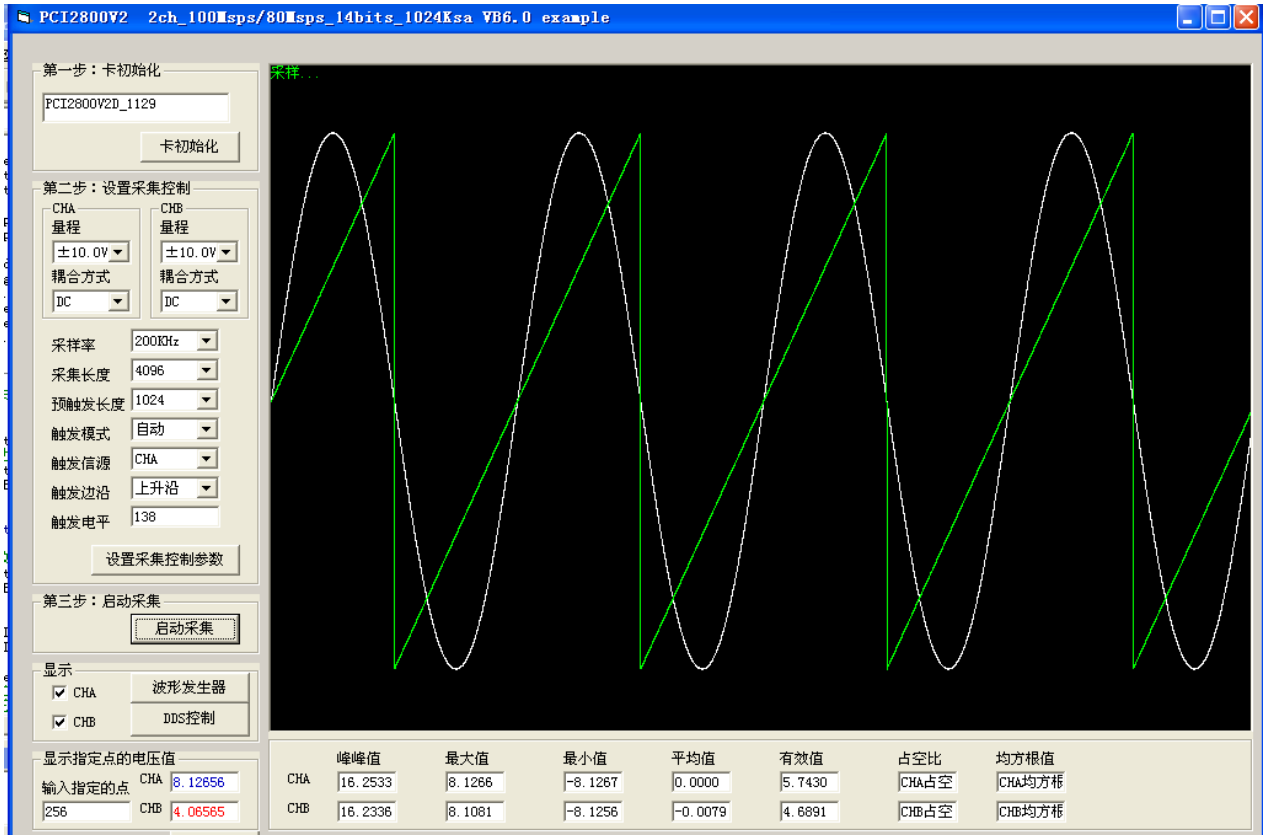
并行数据采集卡提供对多路模拟信号的同步高速化，为避免多通道总线宽度不够而造成数据丢失，在采集卡上设计了一定容量的数据缓存池，便于高速采集数据实时存入。用户只需要操作采集动态库就可以实现采集卡的全部功能。

系列采集卡使用相同的采集动态库，继承了以前的产品特色，也为产品升级和功能的扩展做好了准备。演示程序可以驱动已生产出的各种类型的采集卡，并对采集卡上各个通道的属性参数分别设置，一次显示四个通道的波形。

本文主要介绍采集卡的工作流程、采集动态驱动库接口函数采集演示程序的使用方法。为适应用户需要，采集演示程序有 VB DEMO 版本，实现了连续采集和分组采集和其他的一些采集基本功能，参数设

置方便简洁，一目了然，便于初次使用者进行开发。

DSO 数据采集部分



第 1 步、卡自检初始化

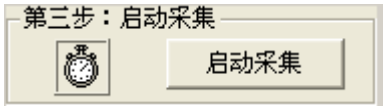

```
'PCI8616 卡初始化
 Private Sub InitCard_Click()
    Dim i As Integer, str As String, Numbers As String
      autoStatus = PCI8616_DSO_VBSysInit(mCardAddr(0), pSysInfo.Idnumber(0),
pSysInfo.NewOffset(0), pSysInfo.CHA_GainTable(0), pSysInfo.CHB_GainTable(0))
    '自检成功
    If autoStatus = 1 Then
        MsgBox "找到 PCI8616 卡并自检通过！"
        For i = 0 To 16
        str = str & ChrW(pSysInfo.Idnumber(i))
        Next
    End If
    '自检失败
    If autoStatus = 0 Then
        MsgBox "没找到 PCI8616 卡！"
        str = "无　卡"
    End If
    卡 1 地址.Text = str
    CardAddress = mCardAddr(0)     '本例程只对第一张卡进行操作
    '如对第二张卡进行操作  CardAddress = mCardAddr(1)
    '如对第三张卡进行操作  CardAddress = mCardAddr(2)

    End Sub
```

第 2 步、设置卡采样参数


```
'PCI8616 卡设置采样参数
Private Sub OutputWave_Click()
    Dim tmp As Integer
    Dim t1 As Integer
    Dim t2 As Integer

    '设置采集控制参数
    pCtrlInfo.SampleIdx = SetSamRate.ListIndex


    '设置量程
    pCtrlInfo.gaina = AGain.ListIndex    '量程
    pCtrlInfo.gainb = BGain.ListIndex    '量程

    '设置藕合方式  DC
    pCtrlInfo.couplecha = SetChaCoup.ListIndex
    pCtrlInfo.couplechb = SetChbCoup.ListIndex

    '设置触发控制
    pCtrlInfo.TrigMode = SetTrigMode.ListIndex       '触发模式
    pCtrlInfo.TrigSource = TrigSource.ListIndex      '触发信号源
    pCtrlInfo.TrigEdge = SetTrigEdge.ListIndex       '触发边沿
    pCtrlInfo.TrigLevel = Val(TrigLevel.Text)        '触发电平
    t1 = SamLength.ListIndex     '采样长度
    t2 = SetTrigPreIdx.ListIndex '预触发长度
    pCtrlInfo.SampleLength = 1024 * (2 ^ t1)
    pCtrlInfo.PreSampleLength = 1024 * (2 ^ t2)


    Ka = 1 / pSysInfo.CHA_GainTable(pCtrlInfo.gaina)
    Kb = 1 / pSysInfo.CHB_GainTable(pCtrlInfo.gainb)
    If (autoStatus = 1) Then
        status = PCI8616_DSO_VBSetHardWare(CardAddress, pCtrlInfo.SampleIdx, pCtrlInfo.gaina,
pCtrlInfo.gainb, pCtrlInfo.couplecha, pCtrlInfo.couplechb, pCtrlInfo.TrigMode, pCtrlInfo.TrigEdge,
pCtrlInfo.TrigSource, pCtrlInfo.SampleLength, pCtrlInfo.PreSampleLength, pCtrlInfo.TrigLevel)
    Else
        MsgBox "卡没有初始化！"
```

```
        End If
    End Sub
```

第 3 步、启动采样




```
    'PCI8616 卡启动采集
    Private Sub mAcq_Click()
        status = PCI8616_DSO_Acq(CardAddress)
        Timer1.Enabled = True
    End Sub
```

第 4 步、读取采集数据

```
Private Sub Timer1_Timer()
  Dim i As Long, bool As Boolean, temp As Double, FreqValue As Long, str As String, str1 As String, str2 As String
  Dim mDots As Long, casual As Double
  Dim Vpp(1) As Double, Vmax(1) As Double, Vmin(1) As Double, Vrms(1) As Double, Vmean(1) As Double, Vduty(1) As Double, stdev(1) As Double

  mDots = Val(SamLength.Text)
  casual = mDots / picDraw.Width

  '查询数据准备好了没有
    status = PCI8616_DSO_PackData(CardAddress, mDots, addata1(0), addata2(0))
    If status = 1 Then
    '获取真实电压值 ADC 值-1.0V~+1.0V 乘上增益系数就是真实的电压值
      For i = 0 To mDots - 1 Step 1
          mWavedata1(i) = addata1(i) * Ka
          mWavedata2(i) = addata2(i) * Kb
      Next i


      Vmax(0) = Max(mDots, mWavedata1)
      Vmin(0) = Min(mDots, mWavedata1)
      Vpp(0) = Vmax(0) - Vmin(0)
      Vrms(0) = RMS(mDots, mWavedata1)
      Vmean(0) = Mean(mDots, mWavedata1)
      CHA 最大值.Text = Format(Vmax(0), "###0.0000")
      CHA 最小值.Text = Format(Vmin(0), "###0.0000")
      CHA 峰峰值.Text = Format(Vpp(0), "###0.0000")
      CHA 平均值.Text = Format(Vmean(0), "###0.0000")
      CHA 有效值.Text = Format(Vrms(0), "####0.0000")


      Vmax(1) = Max(mDots, mWavedata2)
      Vmin(1) = Min(mDots, mWavedata2)
      Vpp(1) = Vmax(1) - Vmin(1)
      Vrms(1) = RMS(mDots, mWavedata2)
      Vmean(1) = Mean(mDots, mWavedata2)
      CHB 最大值.Text = Format(Vmax(1), "###0.0000")
      CHB 最小值.Text = Format(Vmin(1), "###0.0000")
      CHB 峰峰值.Text = Format(Vpp(1), "###0.0000")
      CHB 平均值.Text = Format(Vmean(1), "###0.0000")
      CHB 有效值.Text = Format(Vrms(1), "####0.0000")
```

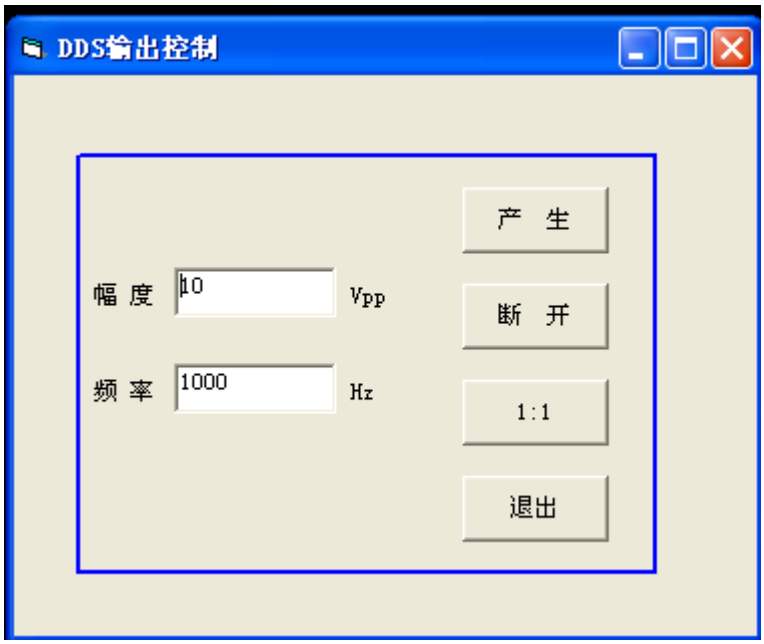
```
        picDraw.AutoRedraw = True
        'RunStatus.Caption = "已触发      "
        str2 = "已触发"
        '采集波形数据并显示
        picDraw.Cls
        If CheckCHA.Value = 1 Then
            bool = DrawLine(casual, addata1, RGB(255, 255, 255))
        End If
        If CheckCHB.Value = 1 Then
            bool = DrawLine(casual, addata2, RGB(0, 255, 0))
        End If
        'RunStatus.Caption = "采样...    "
        str2 = "采样..."
        '显示指定点的电压值
        Dim DataDot As Long
        DataDot = Val(指定点.Text)
        CHA 显示指定点.Text = Format(mWavedata1(DataDot), "###0.00000")
        CHB 显示指定点.Text = Format(mWavedata2(DataDot), "###0.00000")
    Else
        str2 = "正在采集... "
    End If
    str = str2 & str1
    RunStatus.Caption = str
End Sub
```

任意波形部分+DIO 部分


```
        Private Sub AWGATTR_Click()
        If (Kattr = 0) Then
                Kattr = 1
                        AWGATTR.Caption = "10:1"
                Else
                        Kattr = 0
```

```
                AWGATTR.Caption = "1:1"
    End If
    status = PCI8616_ARB_Kswitch(Kstatus, Kattr)
    End Sub

    Private Sub AWGGEN_Click()
    Dim status As Integer
    Dim WAVETYPE As Integer
    Dim freq As Double
    Dim amp As Double
    Dim offset As Double
    Dim duty As Double
    Dim mDots(16) As Integer
    Dim mDacFreqency(16) As Double
    Dim mWaveData(16384) As Double

    amp = Val(AWGAMP.Text)
    freq = Val(AWGFREQ.Text)
    offset = Val(AWGOFFSET.Text)
    duty = Val(AWGDUTY.Text)

    status = PCI8616_ARB_GenAwgWave(wavetpe, freq, amp, offset, duty, mDots(0), mWaveData(0),
mDacFreqency(0))

    End Sub
```

DDS 部分




```
Dim Kattr As Integer
Dim Kstatus As Integer
Private Sub DDSATTR_Click()
Dim status As Integer
If (Kattr = 0) Then
                Kattr = 1
                DDSATTR.Caption = "10:1"
Else
                Kattr = 0
                DDSATTR.Caption = "1:1"
End If
status = PCI8616_DDS_Kswitch(Kattr, Kstatus)
End Sub
```

```
Private Sub DDSGEN_Click()
Dim amp As Double
Dim freq As Double
Dim status As Integer
amp = Val(DDSAMP.Text)
freq = Val(DDSFREQ.Text)
status = PCI8616_DDS_SingleToneMode(amp, freq)
End Sub
Private Sub DDSON_Click()
Dim status As Integer
If (Kstatus = 0) Then
        Kstatus = 1
        DDSON.Caption = "断  开"
Else
        Kstatus = 0
        DDSON.Caption = "输  出"
End If
status = PCI8616_DDS_Kswitch(Kattr, Kstatus)
End Sub
Private Sub exit_Click()
    Unload Form2
End Sub
Private Sub Form_Load()
    DDSAMP.Text = 10#
    DDSFREQ = 1000#
 End Sub



}
```

2.1 显示波形数据转化

在 VB6.0 中，在屏幕上画点是按照屏幕分辨率来作为坐标表示的，因此从采集卡读到的波形数据不能直接用于显示，必须先通过算法把波形数据转换成适合显示的数组（DrawBuffer），这个二维数组第一个参数（行）表示通道，第二个参数的点数有 x，y 坐标，就是在屏幕上画点的坐标，所以 DrawBuffer 中的数据跟真实数据没关系，DrawBuffer 仅仅是为了最后的画图，程序就是通过画 DrawBuffer 来实现波形显示功能的。

2.2 启动采集

一旦启动采集，由于硬件特性，采集卡上的所有通道都开始采集，而不是只有选择或者是设置了的那个通道进行采集。

2.3 采样设置

每个通道单独设置量程、耦合方式；

采样率、采样长度与通道无关，一旦设定好每个通道这三个属性都一样。