

# Design of Double-channel Differential Input Oscilloscope Based on JKU12DAS (CVI)

Shu Ding

Chengdu Airlines Electronic Technology Co., Ltd., Chengdu, Sichuan, 611430, China

## Abstract

Virtual instrument is based on computer hardware and software test platform, which consists of computer, application software and hardware, the computer hardware resources and instrument hardware organically integrated by the virtual instrument software development platform, the powerful data processing ability combine with computer hardware instrument, realization of the data display, storage and analysis by the software.

This paper gives a way of using virtual instrument of virtual oscilloscope design scheme. the major hardware is LabjackU12 data acquisition card, to realize the two-way differential input; the major software is LabWindows/CVI development environment, to realize the PC virtual oscilloscope instrument panel control and corresponding program design. The final debugging results is achieve correct data acquisition, can be achieved right waveform display, channel selection, horizontal time-base, sensitivity, vertical displacement and two channel wave add and subtract, and other functions by button on the control panel. the design can complete the general situations instead of traditional oscilloscope, such as simple waveform signal detection.

## Keywords

virtual instrument; DAQ Card(Data Acquisition Card); oscilloscope; LabWindows/CVI

---

## 基于 JKU12DAS 双路差动输入示波器设计 (CVI)

丁舒

民航成都电子技术有限责任公司, 中国·四川·成都 611430

## 摘要

虚拟仪器是在计算机的软件和硬件基础上搭建的测试平台, 它由计算机、应用软件和硬件三部分组成, 通过虚拟仪器软件开发平台将计算机硬件资源与仪器硬件有机地融为一体, 把计算机强大的数据处理能力和仪器硬件的测量、控制能力结合在一起, 通过软件实现对数据的显示、存储以及分析处理。

本文给出了一种用虚拟仪器方式实现的虚拟示波器的设计方案。硬件采用 LabjackU12 数据采集卡, 实现双路差动输入; 软件采用 LabWindows/CVI 开发环境, 实现在 PC 机上的虚拟示波器仪器面板控制及相应程序设计。最终调试结果为可以实现正确的数据采集, 可通过控制面板上的按钮正确实现波形的显示、通道选择、水平时基、垂直灵敏度、位移、两通道波形相加减等功能。完成的设计可在一般场合代替传统的示波器, 用于进行简单的波形信号检测。

## 关键词

虚拟仪器; 数据采集卡; 示波器; LabWindows/CVI

---

## 1 引言

网络技术的多样化和智能化, 使得虚拟仪器越来越受各领域喜爱, 传统的仪器必将由于自身功能的不足而逐步被虚拟仪器所取代。虚拟示波器不仅可以像传统示波器一样测试波形, 调节增益或水平时基, 还可以根据用户自身需求自己定义或添加新的功能; 比起传统仪器的功能固定化, 设计灵活已成为虚拟仪器独特的优势<sup>[1]</sup>。

## 2 设计说明

### 2.1 设计思想

采用数据采集卡和微型计算机实现双路差动输入示波器信号采集。其中硬件使用 LabJackU12 采集卡, 软件采用 LabWindows/CVI 开发环境。

本次设计主要分为三个步骤进行:

- (1) 硬件: 数据采集卡安装于调试, 整个硬件平台搭建。
- (2) 软件: 示波器主面板设计, 后台所有程序编写。

(2) 调试: 整体设计完成, 检查并修改后台程序, 调试前面板各个控件功能是否可以正常使用。

## 2. 2 技术指标

- (1) 模拟输入: 差动输入, 双通道。
- (2) 每路输入范围:  $-10V \sim +10V$ 。
- (3) 分辨率: 12bit。
- (4) 设置水平时基调节(根据硬件特性)。
- (5) 设置垂直灵敏度调节(根据硬件特性)。
- (6) 显示模式: 双通道、A+B 及 A-B。

## 3 硬件设计

### 3. 1 数据采集系统

数据采集, 即运用数字化处理技术, 对被测对象进行分析, 将被测对象的各种参量通过采集系统内部的传感模块转换为电信号, 然后对转换的电信号进行常规的信号处理, 如采样、量化、编码等, 最后传送到计算机进行显示、存储或者分析的过程, 数据采集系统就是可以完成完整数据采集过程的系统, 完整的数据采集系统由多路开关、采样/保持器、模/数转换器及存储缓存器等构成<sup>[2-3]</sup>, 如图 3.1 所示:

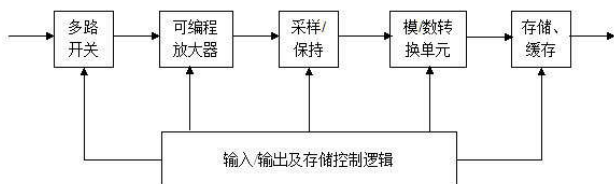


图 3.1 DAQ 工作原理图

### 3. 2 LabJackU1简介

数据采集卡 LabJackU12 与整体系统连接效果如图 3.2 所示:

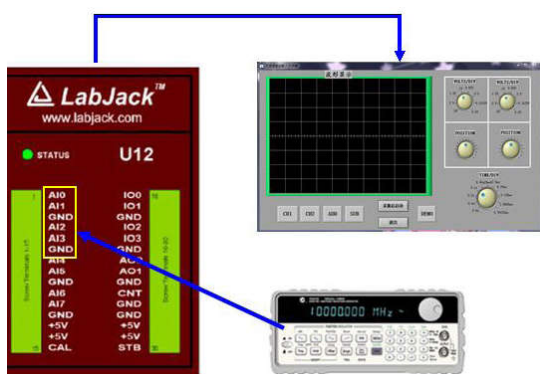


图 3.2 LabJackU12 的正面图

其是由 LabJack 公司生产的具有 USB 接口的采集卡, 拥有 12 位的分辨率, 在差动模式下甚至可以达到 16 位, 而且拥有 8 个单端输入或者 4 个差动输入端<sup>[4]</sup>, USB 接口既是供电口, 亦是数据传输口, 所有的数据传输都通过数据线完成, 也就是说, 用户只需一根数据线就可以实现计算机与采集卡之间的通信, 本次设计是实现双路差动, 所用引脚口为 AI0-AI3 以及两个接地端, LabJackU12 数据采集卡输入双通道与单通道的原理相同, 单通道原理框图如图 3.3 所示:

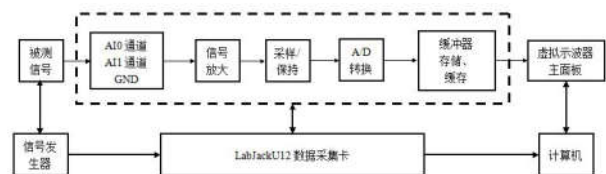


图 3.3 LabJackU12 数据采集卡输入通道原理图

### 3. 与设计相关的采集卡驱动函数

为了顺利完成数据采集, 使用数据采集卡的 PC 机必须装有相应的驱动, 本设计使用到的数据采集卡对应驱动函数是 AIBurst, 它以指定的扫描率(可高到 8192 赫兹)读取指定扫描数的电压, 然后才被传送到计算机上, AIBurst 在 LabWindows/CVI 中各个参数主要的设置形式如下<sup>[4]</sup>:

```
long error;  
long id=-1,*idnum=&id;  
long localID=-1;  
long demo=0,numChannels=2,disableCal=0;  
long transferMode=0;  
long channels[4]={8,9,10,11};  
long gains[4]={0,0,0,0};  
float sr=1024,*scanRate=&sr;  
long numScans=1024,timeout=30;  
float voltages[4096][4]={0};  
long stateIOout[4096]={0};  
long ov,*overVoltage=&ov;  
long stateIOin=0;  
long updateIO=0;  
long ledOn=1;  
long triggerIO=0;  
long triggerState=0;
```

具体的参数说明:

- \*idnum - 设备号, 系统所能识别出的第一个设备
  - demo - 正常操作时为 0, 大于 0 时为演示模式。
  - \*stateIOin - 要输出的 IO0-IO3 的状态。
  - updateIO - 大于 0, 则输出状态值, 否则只进行读操作。
  - ledOn - 大于 0 时, LabJack 的发光二极管被点亮。
  - numChannels - 要读取的模拟通道数 (1,2, 或 4)。
  - \*channels - 这是一个数组的指针, 该数组至少要有 numChannels 个单元。单端输入用 0-7, 差动输入用 8-11。
  - \*gains - 这是一个数组的指针, 该数组至少要有 numChannels 个单元。各单元值是响应通道的增益指令。0= $\rightarrow$  1, 1= $\rightarrow$  2, ..., 7= $\rightarrow$  20。只在差动通道采集时有效。
  - \*scanRate - 每秒的扫描数。一个扫描意味着对所有指定通道的一次采集。采样频率 (等于 scanRate \* numChannels) 必须在 400 和 8192 之间。
  - disableCal - 大于 0 时, 返回的电压是没有用校正系数进行校正计算的值。(叫做原始值)。
  - triggerIO - 出发信号的 IO 口号。(0=不用触发, 1=IO0, 2=IO1)。
  - triggerState - 大于 0 时, 采集将在所选的 IO 口变高电平时开始。
  - numScans - 指定多少个扫描将被返回。最小值为 1。numSamples 最大值为 4096, 其中 numSamples = numScans \* numChannels。
  - timeout - 超时时间 (单位为秒)。该函数执行时间超过规定的超时时间时, 即使还没有采到一个扫描, 也会立即退出, 同时返回超时错误码。
  - \*voltages - 这是一个指向一个 4096 乘 4 数组的指针。该数组用于放置采集的电压值。将该数组元素置零输出。
  - transferMode - 0=自动, 1=正常, 2=快速。如果是自动, 当 timeout  $\geq$  4 或 numScans/scanRate  $\geq$  4 时, 实际使用模式是正常, 否则是快速。
- 对以上各个参数逐一设计调试, 调用以后可以正常运行。调用格式如下:
- ```
error=AIBurst(idnum,demo,stateIOin,updateIO,ledOn,numChannels,channels,gains,scanRate,disableCal,triggerIO,triggerState,numScans,timeout,voltages,stateIOout,overV
```

age,transferMode);

## 4 软件设计

### 4.1 Labwindows/CVI 简介

本设计使用的软件平台 Labwindows/CVI, 它是一款基于 C 语言的软件开发平台, 使用简单, 设计灵活, 由 NI 公司生产, 主要用于虚拟仪器仿真, 用户可根据自身需求合理选择所需控件, 设计出的界面美观逼真, 而且只需对每个控件进行回调函数定义就可以很方便控制该控件, 软件本身后台程序调试方法多样且有效, 断点调试让用户不必担心编写程序时整体与局部之间的矛盾, 设计出低成本、高效率的虚拟仪器<sup>[5]</sup>。

### 4.2 示波器主面板设计

使用软件设计虚拟示波器主面板的过程主要包括控件放置及属性设置, 此次设计的主面板界面如图 4.1 所示:

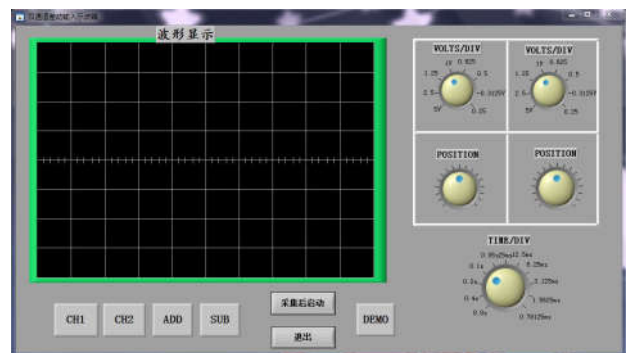


图 4.1 示波器前面板

控件放置时, 主要考虑到整个主面板的美观性及实用性, 避免太花哨或太单一; 属性设置时, 主要是回调函数和控件外观的设计, 各个控件的回调函数直接影响到后台程序的编写以及整体仪器是否可以正常运行, 具体的控件回调函数和后台程序参见附录 A。

### 4.3 面板各旋钮设计

前面板控件放置完成, 还需对各控件的属性进行设置, 其中最重要的是水平时基旋钮、垂直增益旋钮、垂直位移旋钮的档位设计。

#### 4.3.1 水平时基旋钮设计

水平时基, 又称作水平扫描时基, 即水平轴上采集一个单元格内样本数所需要的时间, 单位为 s/div, ms/div 等等<sup>[6]</sup>, 本设计水平时基旋钮如图 4.2 所示:

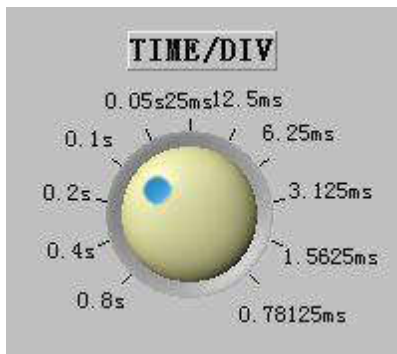


图 4.2 水平时基旋钮

本次设计使用 LabJackU12 采集卡，参考采集卡的详细参数说明，并结合采集卡驱动函数 AIBurst 中的相关规定，可以得到如下公式：

$$\left(\frac{N_{(Sa)}}{R}\right)/10 = time / div$$

其中，N 代表采集的总样本数，采集卡规定采集样本数范围为 64 至 4096；R 代表采样速率，即单位时间能采集的样本数；数字 10 代表将 X 轴平均分为 10 份，据此，所有档位均满足此公式，具体设置如表 4.1 所示：

表 4.1 水平时基档位设置

| 序号           | 1    | 2    | 3    | 4    | 5    |
|--------------|------|------|------|------|------|
| 时基因数 (t/div) | 0.8s | 0.4s | 0.2s | 0.1s | 50ms |
| 取样点数 (N)     | 4096 | 4096 | 4096 | 4096 | 4096 |
| 采样速率 (R)     | 256  | 512  | 1024 | 2048 | 4096 |

| 序号           | 6    | 7      | 8      | 9       | 10       | 11        |
|--------------|------|--------|--------|---------|----------|-----------|
| 时基因数 (t/div) | 25ms | 12.5ms | 6.25ms | 3.125ms | 1.5625ms | 0.78125ms |
| 取样点数 (N)     | 2048 | 1024   | 512    | 256     | 128      | 64        |
| 采样速率 (R)     | 4096 | 4096   | 4096   | 4096    | 4096     | 4096      |

#### 4.3.2 垂直灵敏度旋钮设计

垂直灵敏度，又称作垂直增益，即 Y 轴上每个单元格的偏转电压大小，单位为 V/DIV, mV/DIV 等等<sup>[6]</sup>，本设计垂直灵敏度旋钮如图 4.3 所示：

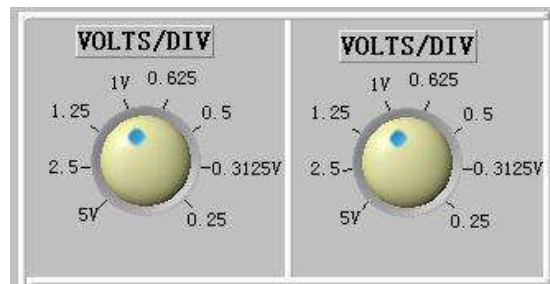


图 4.3 垂直灵敏度旋钮

LabJackU12 采集卡在差分模式下，同样满足如下公式：

$$\frac{V}{G}/8 = v / div$$

其中，V 代表信号之间的电压差，规定为 ±20V, G 代表采集卡放大倍数，数字 8 代表 Y 轴方向均分为 8 份，结合采集卡的放大倍数相关规范，垂直灵敏度旋钮档位设置如表 4.2 所示：

表 4.2 增益与电压的关系

| 序号 | 放大倍数 | 电压 (伏) | 序号 | 放大倍数 | 电压 (伏) |
|----|------|--------|----|------|--------|
| 1  | G=1  | ±20    | 5  | G=8  | ±2.5   |
| 2  | G=2  | ±10    | 6  | G=10 | ±2     |
| 3  | G=4  | ±5     | 7  | G=16 | ±1.25  |
| 4  | G=5  | ±4     | 8  | G=20 | ±1     |

#### 4.3.3 垂直位移旋钮设计

垂直位移用于调整波形的垂直位置，便于观察采集到的数据，Y 轴方向均分为 8 份，为使设计更加细致，本设计将垂直位移设计为 17 档，即每个单元格占用 2 个档位，其中，往左旋转旋钮波形将向下移动，往右旋转旋钮波形将向上移动，垂直偏移旋钮如图 4.4 所示：

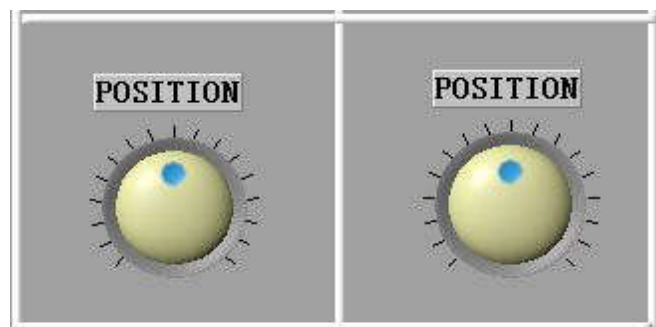


图 4.4 垂直偏移旋钮

### 4. 整体运行流程

软件与硬件设计完成后，便可对整个系统进行全面调试，调试流程分为两个模块，分别是波形显示模块和波形控制模块，具体流程图如图 4.5，图 4.6 所示：

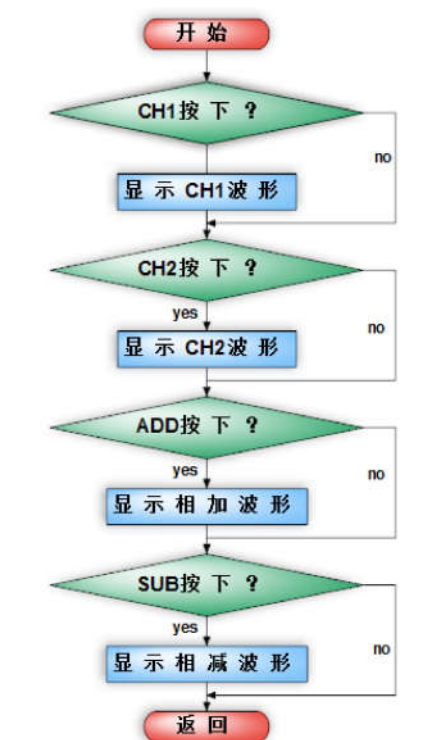


图 4.5 波形显示模块程序流程图

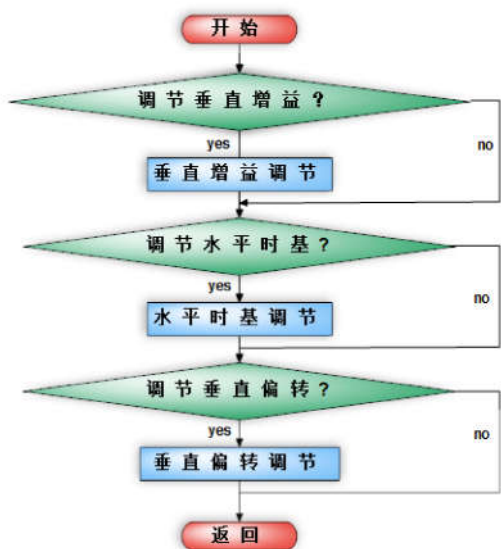


图 4.6 波形控制模块程序流程图

## 5 系统整体运行调试

### 5.1 演示模式调试

本设计配有标准波形演示模式，当未连接数据采集卡或没有任何外界输入信号时，按下 demo 按键，面板右下角出现一系列控件，可根据改变控件作用来改变需要显示的波形，

再次点击 demo 按键，按键被关闭，演示的一系列控件被隐藏，如图 5.1 所示为当标准波形为锯齿波时的情况：

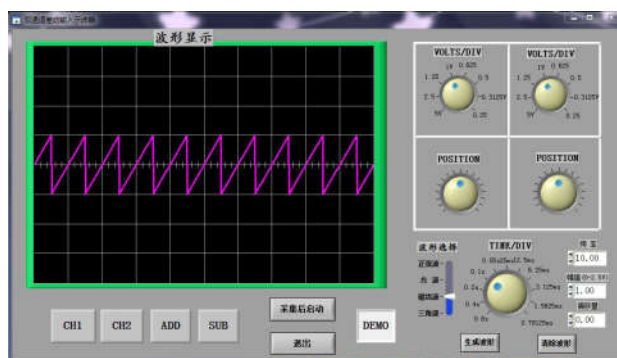


图 5.1 demo 调试

### 5.2 数据采集调试

本次设计的整个硬件平台搭建如图 5.2 所示：

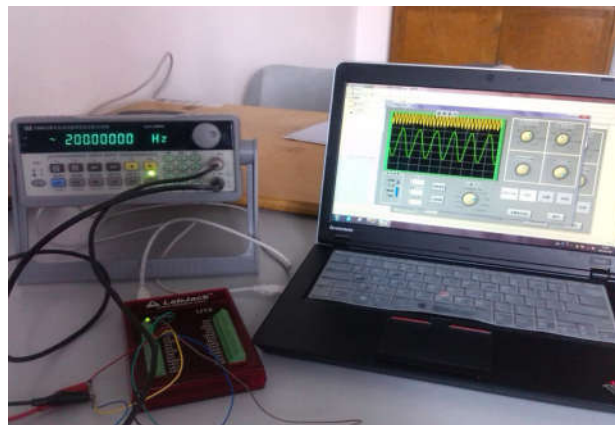


图 5.2 系统调试硬件连接图

#### 5.2.1 最高信号频率确定

LabJackU12 数据采集卡的最高扫描频率为 8192Hz，由采样定理可知：

$$f(S) \geq 2f(H) \text{ (一般取 } 5 \sim 10 \text{)}$$

即采样频率至少为 2 倍信号频率<sup>[7]</sup>，但实际情况中，采样是为了恢复原始波形，以正弦波为例，恢复原始波形至少需要 5 个点，一般情况采样频率取 5 至 10 倍的最高信号频率，为减小误差，此处取 10 倍，则每个通道最高信号频率为 400Hz，此处调试取 200Hz。

#### 5.2.2 通道功能调试

##### (1) 单通道显示

整个系统搭建完成并上电后，点击 CH1 按键，点击采集后启动，稍待片刻，波形显示区域将显示采集到的波形，若波形显示紊乱，调节水平时基旋钮可获得理想波形，最终效

果如图 5.3、5.4 所示:

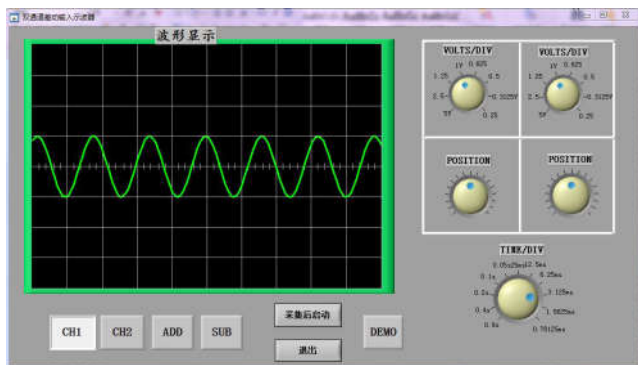


图 5.3 通道 1 的调试

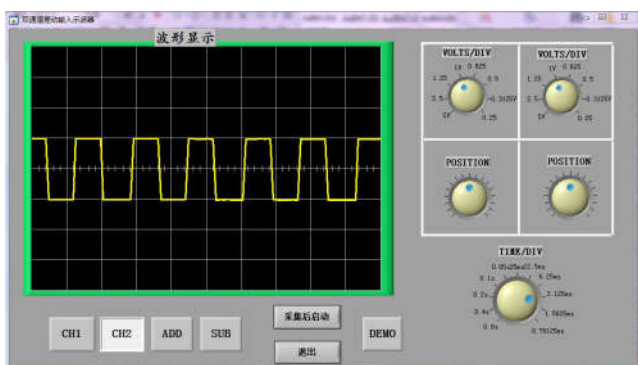


图 5.4 通道 2 的调试

### (2) 双通道显示

相继按下 CH1、CH2 按键, 调整到合适的水平时基, 点击采集后启动, 稍待片刻, 波形显示区域显示采集到的双通道波形, 如图 5.5 所示:

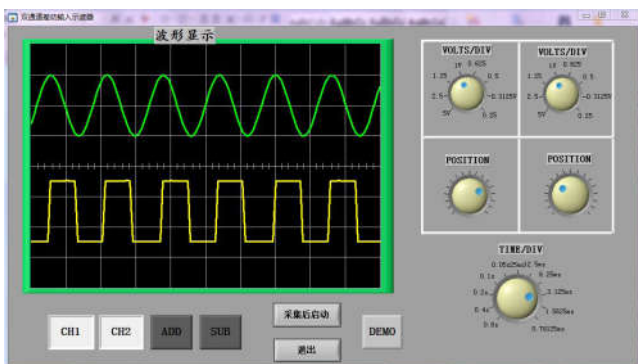


图 5.5 双通道显示的调试

### (3) 波形相加与相减

双通道都可以正常显示的情况下, 按下 ADD 按键或是 SUB 按键, 可以观察到相应波形叠加与相减的结果, 图 5.6 所示的红色不规则曲线即为波形叠加效果, 图 5.7 所示的浅蓝色不规则曲线即为波形相减效果。

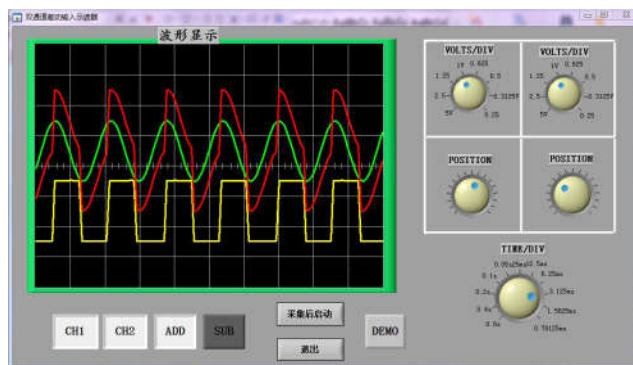


图 5.6 ADD 的调试

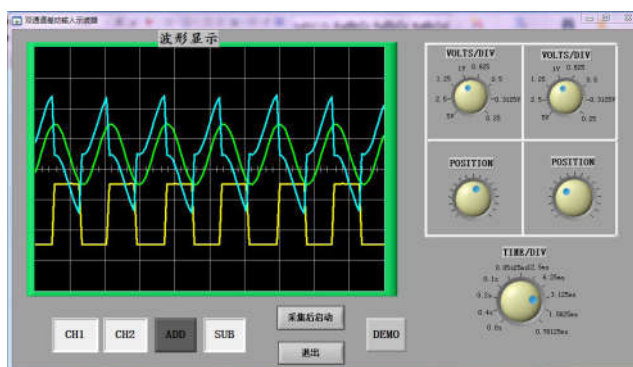


图 5.7 SUB 的调试

## 5.2.3 旋钮功能调试

### (1) 水平时基旋钮调试

波形正常显示情况下, 通过选择水平时基旋钮的档位, 可以改变波形的水平时基, 具体效果如图 5.8 所示:

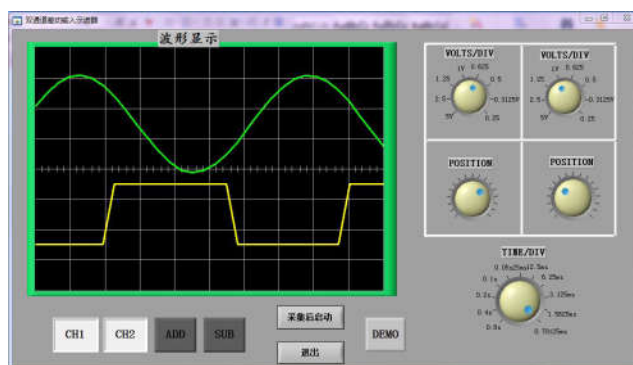


图 5.8 水平时基的调试

### (2) 垂直灵敏度旋钮调试

波形正常显示情况下, 通过选择垂直灵敏度旋钮的档位, 可以改变波形的垂直偏转电压大小, 如图 5.9 所示为当双通道分别选择不同垂直灵敏度的情况:

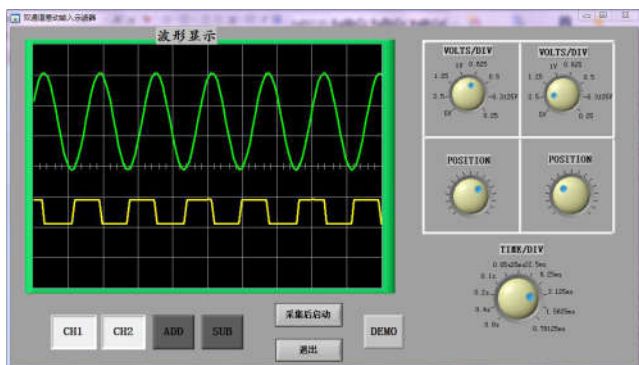


图 5.9 垂直灵敏度的调试

综上所述，结合以上调试结果，明显看出：本次设计调试基本符合要求，即可以单通道单独显示（如只显示 CH1 或 CH2），又可以双通道显示（如同时显示 CH1 通道和 CH2 通道的波形），垂直灵敏度和水平时基的调节也可以完整实现，而垂直位移在整个波形显示过程都在使用，在此不做过多介绍。

## 6 总结

本次设计基本满足指标要求，不仅可以正常采集数据，还可以对采集波形进行简单调试，但是，由于采集卡自身条件限制，采样频率无法提的更高，相信随着科技的不断进步，更加高频率的采集卡或者采集系统会满足所有领域的要求。

## 参考文献

- [1] 李江全等编著. 案例解说虚拟仪器典型控制应用. 北京: 电子工业出版社, 2011.5
- [2] 白云, 高育鹏, 胡小江. 基于 LabVIEW 的数据采集与处理技术. 西安: 西安电子科技大学出版社, 2009
- [3] 孙晓云, 郭立伟, 孙会琴. 基于 LabWindows/CVI 的虚拟仪器设计与应用. 北京: 电子工业出版社, 011.9
- [4] LabJack U12 多功能数据采集控制器用户说明书版本 1.08. 北京迪阳世纪科技有限公司, 2004.4
- [5] 刘君华. 虚拟仪器编程语言 LabWindows/CVI 教程. 北京: 电子工业出版社, 2009.8
- [6] 孙灯亮编著. 数字示波器原理和应用. 上海: 上海交通大学出版社, 2012.02
- [7] 苗长云主编. 现代通信原理及应用 (第二版), 北京: 电子工业出版社, 2009.8

## 附录 A

### 虚拟示波器面板的后台程序

```
#include "toolbox.h"
#include "asynctmr.h"
#include <analysis.h>
#include <cvirte.h>
#include <userint.h>
#include "ljackuw.h"
#include "主面板.h"
static int panelHandle;
void CVICALLBACK huatu(void);
void initial(void);
void GenerateWave(int shape);
double wave[4096];
double wave1[4096];
double wave11[4096];
double wave2[4096];
double wave22[4096];
double temp[4096];
int goe=0;
int status=0;
int statusforoutput;
int count;
int ao;
int acnt=0;
long dm;
double x0,y0,xn,yn;
int statusofchannel1;
int statusofchannel2;
int statusofadd;
int statusofsub;
int statusofdemo;
int handleofchannel1;
int handleofchannel2;
unsigned int zengyi,zengyi2,yinzi1,yinzi2;
double pianyi,pianyi2;
unsigned int shuiping;
```

```
float sr=1024,*scanRate=&sr;
long numScans=1024,timeout=30;
long gains[4]={0,0,0,0};
long id=-1,*idnum=&id; // 等于-1 代表能找到的第一个
设备
/*
程序主函数部分
*/
int main (int argc, char *argv[])
{
if (InitCVIRTE (0, argv, 0) == 0)
return -1; /* out of memory */
if ((panelHandle = LoadPanel (0, “主面板.uir” , PANEL))
< 0)
return -1;
DisplayPanel (panelHandle);
SetCtrlAttribute(panelHandle,PANEL_WAVECHOOSE,ATTR_
VISIBLE,0) ;
SetCtrlAttribute(panelHandle,PANEL_FREQ,ATTR_VISI-
BLE,0) ;
SetCtrlAttribute(panelHandle,PANEL_VAL,ATTR_VISI-
BLE,0) ;
SetCtrlAttribute(panelHandle,PANEL_GWAVE,ATTR_
VISIBLE,0) ;
SetCtrlAttribute (panelHandle , P A N E L _
CLEARPLOT,ATTR_VISIBLE,0) ;
SetCtrlAttribute(panelHandle,PANEL_PHASE,ATTR_
VISIBLE,0) ;
RunUserInterface ();
DiscardPanel (panelHandle);
return 0;
}
int CVICALLBACK Quit (int panel, int control, int event,
void *callbackData, int eventData1, int eventData2)
{ int confirm;
switch (event)
{
```

```
case EVENT_COMMIT:
confirm=ConfirmPopup( “Confirm” ,” Are you sure you
want to quit system?” );
if(confirm==1)
QuitUserInterface(0);
break;
case EVENT_RIGHT_CLICK:
MessagePopup ( “Quit Button Help” ,
“quit the program and close the pannel ”);
break;
}
return 0;
}
int CVICALLBACK GWave (int panel, int control, int
event,
void *callbackData, int eventData1, int eventData2)
{ int shape;
int i;
if(event==EVENT_COMMIT)
{
GetCtrlVal(panelHandle,PANEL_DEMO,&statusofdemo);
G e t C t r l V a l ( p a n e l H a n d l e , P A N E L _
CHANNEL_1,&statusofchannel1);
GetCtrlVal(panelHandle,PANEL_WAVECHOOSE,&shape);
GetCtrlVal(panelHandle,PANEL_RINGKNOB,&pianyi);
GetCtrlVal(panelHandle,PANEL_RINGKNOB_2,&zengyi);
G e t C t r l V a l ( p a n e l H a n d l e , P A N E L _
CHANNEL_2,&statusofchannel2);
GenerateWave(shape);
if (statusofdemo==statusofchannel1)
{
MessagePopup( “警告” ,” 请选择一个通道!!! ”);
}
if (statusofdemo==1&statusofchannel1==0)
{
//SetCtrlVal
if (shape==4)
```



```

    {}
    else
    {
        DeleteGraphPlot(panelHandle,PANEL_GRAPH,-1,VAL_
IMMEDIATE_DRAW);
        for(i=0;i<numScans;i++)
        {
            wave1[i]=wave[i]*zengyi+pianyi;
        }
        huatu();
    }
    if (statusofchannel1 == 1 & statusofdemo == 0)
    {
        if (shape == 4)
        {}
        else
        {
            DeleteGraphPlot(panelHandle,PANEL_GRAPH,-1,VAL_
IMMEDIATE_DRAW);
            for(i=0;i<numScans;i++)
            {
                wave2[i]=wave[i]*zengyi+pianyi;
            }
        }
    }
    return 0;
}

void GenerateWave(int shape)
{
    double valtage,phase;
    int n;
    double freq;
    GetCtrlVal(panelHandle,PANEL_PHASE,&phase);
    GetCtrlVal(panelHandle,PANEL_FREQ,&freq);
    GetCtrlVal(panelHandle,PANEL_VAL,&valtage);

```

```

    if(valtage>2.5 || valtage<0)
    {
        MessagePopup(“错误信息”,“幅值必须在 0 到 2.5V”);
    }
    else
    switch(shape)
    { case 0:
        SineWave(1024,valtage/5,freq/1024,&phase,wave); //
        break;
        case 1:
        SquareWave(1024,valtage/5,freq/1024,&phase,50.0,wave);
        break;
        case 2:
        SawtoothWave(1024,valtage/5,freq/1024,&phase,wave);
        break;
        case 3:
        TriangleWave(1024,valtage/5,freq/1024,&phase,wave);
        break;
    }
}

int CVICALLBACK updatecallback (int panel, int control,
int event,
void *callbackData, int eventData1, int eventData2)
{ /*long error;
long id=-1,*idnum=&id;
long localID=-1;
long demo=0,numChannels=2,disableCal=0;
long transferMode=0;
long channels[4]={ 8,9,10,11 };
long gains[4]={ 0,0,0,0 };
float voltages[4096][4]={ 0 };
long stateIOout[4096]={ 0 };
long ljb=-1,*ljbScanBacklog=&ljb;
long ov,*overVoltage=&ov;
long numIts = 5;
long junk;

```

```
long stateIOin=0 ;
long updateIO=0 ;
long ledOn=1;
long triggerIO=0;
long triggerState=0;
switch (event)
{
case EVENT_TIMER_TICK:
GetCtrlVal (panelHandle, PANEL_CHANNEL_1,
&statusofchannel1);
GetCtrlVal (panelHandle, PANEL_CHANNEL_2,
&statusofchannel2);
huatu();
break;
}
return 0;
}
int CVICALLBACK ClearPlot (int panel, int control, int
event,
void *callbackData, int eventData1, int eventData2)
{
switch (event)
{
case EVENT_COMMIT:
DeleteGraphPlot(panelHandle,PANEL_GRAPH,-1,VAL_
IMMEDIATE_DRAW);
break;
}
return 0;
}
int CVICALLBACK PianYi(int panel, int control, int event,
void *callbackData, int eventData1, int eventData2)
{
if (event == EVENT_VAL_CHANGED) {
GetCtrlVal (panelHandle, PANEL_RINGKNOB, &pianyi);
// SetCtrlVal (panelHandle, PANEL_NUMERIC_VERTI-
CAL, pianyi);
huatu(); // 执行 huatu ( ) 函数
}
return (0);
}
int CVICALLBACK ZengYi(int panel, int control, int event,
void *callbackData, int eventData1, int eventData2)
{ int i,j,k;
long error;
long id=-1,*idnum=&id;
long localID=-1;
long demo=0,numChannels=2,disableCal=0;
long transferMode=0;
long channels[4]={8,9,10,11};
long gains[4]={0,0,0,0};
float voltages[4096][4]={0};
long stateIOout[4096]={0};
long ljb=-1,*ljbScanBacklog=&ljb;
long ov,*overVoltage=&ov;
long numIts = 5;
long junk;
long stateIOin=0 ;
long updateIO=0 ;
long ledOn=1;
long triggerIO=0;
long triggerState=0;
//stateIOout=0;
if (event == EVENT_VAL_CHANGED)
{
GetCtrlVal (panelHandle, PANEL_RINGKNOB_2,
&zengyi);
switch(zengyi)
{
case 0: gains[0]=0;yinzi1=1; break;
case 1: gains[0]=1;yinzi1=2; break;
case 2: gains[0]=2;yinzi1=4; break;
case 3: gains[0]=3;yinzi1=5; break;
```

```

case 4: gains[0]=4;yinzi1=8; break;
case 5: gains[0]=5;yinzi1=10; break;
case 6: gains[0]=6;yinzi1=16; break;
case 7: gains[0]=7;yinzi1=20; break;
}

AIBurst(idnum,demo,stateIOin,updateIO,ledOn,numChannels,channels,gains,scanRate,disableCal,triggerIO,triggerState,numScans,timeout,voltages,stateIOout,overVoltage,transferMode);

for(i=0;i<numScans;i++) wave1[i]=voltages[i][0];
for(i=0;i<numScans;i++) wave2[i]=voltages[i][1];
for(i=0;i<numScans;i++)
{
wave11[i]=wave1[i];
wave22[i]=wave2[i];
}
huatu();
}
return (0);
}

int CVICALLBACK PianYi2 (int panel, int control, int event,
void *callbackData, int eventData1, int eventData2)
{
switch (event)
{
case EVENT_COMMIT:
GetCtrlVal (panelHandle, PANEL_RINGKNOB_4,
&pianyi2);//huatu();
break;
}
return 0;
}

int CVICALLBACK ZengYi2 (int panel, int control, int event,
void *callbackData, int eventData1, int eventData2)
{ int i,j,k;
long error;
long id=-1,*idnum=&id;
long localID=-1;
long demo=0,numChannels=2,disableCal=0;
long transferMode=0;
long channels[4]={8,9,10,11};
long gains[4]={0,0,0,0};
float voltages[4096][4]={0};
long stateIOout[4096]={0};
long ljb=-1,*ljbScanBacklog=&ljb;
long ov,*overVoltage=&ov;
long numIts = 5;
long junk;
long stateIOin=0 ;
long updateIO=0 ;
long ledOn=1;
long triggerIO=0;
long triggerState=0;
//stateIOout=0;
if (event == EVENT_VAL_CHANGED)
{
GetCtrlVal (panelHandle, PANEL_RINGKNOB_5,
&zengyi2);
switch(zengyi2)
{
case 0: gains[1]=0;yinzi2=1; break;
case 1: gains[1]=1;yinzi2=2;break;
case 2: gains[1]=2;yinzi2=4;break;
case 3: gains[1]=3;yinzi2=5;break;
case 4: gains[1]=4;yinzi2=8;break;
case 5: gains[1]=5;yinzi2=10;break;
case 6: gains[1]=6;yinzi2=16;break;
case 7: gains[1]=7;yinzi2=20;break;
}
AIBurst(idnum,demo,stateIOin,updateIO,ledOn,numChannels,channels,gains,scanRate,disableCal,triggerIO,triggerState,numScans,timeout,voltages,stateIOout,overVoltage,transferMode);
for(i=0;i<numScans;i++) wave1[i]=voltages[i][0];

```

```
for(i=0;i<numScans;i++) wave2[i]=voltages[i][11];  
for(i=0;i<numScans;i++)  
{wave11[i]=wave1[i];  
wave22[i]=wave2[i];  
}  
huatu();  
}  
return 0;  
}  
int CVICALLBACK ShuiPing (int panel, int control, int  
event,  
void *callbackData, int eventData1, int eventData2)  
{  
int i,j,k;  
long error;  
long id=-1,*idnum=&id;  
long localID=-1;  
long demo=0,numChannels=2,disableCal=0;  
long transferMode=0;  
long channels[4]={8,9,10,11};  
long gains[4]={0,0,0,0};  
float voltages[4096][4]={0};  
long stateIOout[4096]={0};  
long ljb=-1,*ljbScanBacklog=&ljb;  
long ov,*overVoltage=&ov;  
long numIts = 5;  
long junk;  
long stateIOin=0 ;  
long updateIO=0 ;  
long ledOn=1;  
long triggerIO=0;  
long triggerState=0;  
//stateIOout=0;  
if (event == EVENT_VAL_CHANGED) {  
GetCtrlVal (panelHandle, PANEL_RINGKNOB_3,  
&shuiping);  
switch(shuiping)  
{  
case 0: sr=256;numScans=2048; break;  
case 1:sr=256;numScans=1024;break;  
case 2:sr=512;numScans=1024;break;  
case 3:sr=1024;numScans=1024;break;  
case 4:sr=2048;numScans=1024;break;  
case 5:sr=4096;numScans=1024;break;  
case 6:sr=4096;numScans=512;break;  
case 7:sr=4096;numScans=256;break;  
case 8:sr=4096;numScans=128;break;  
case 9:sr=4096;numScans=64;break;  
case 10:sr=4096;numScans=32;break;  
}  
AIBurst(idnum,demo,stateIOin,updateIO,ledOn,numChannels,  
s,channels,gains,scanRate,disableCal,triggerIO,triggerState,numSc  
ans,timeout,voltages,stateIOout,overVoltage,transferMode);  
for(i=0;i<numScans;i++) wave1[i]=voltages[i][0];  
for(i=0;i<numScans;i++) wave2[i]=voltages[i][11];  
for(i=0;i<numScans;i++)  
{wave11[i]=wave1[i];  
wave22[i]=wave2[i];  
}  
huatu();  
}  
return (0);  
return 0;  
}  
int CVICALLBACK qidong (int panel, int control, int event,  
void *callbackData, int eventData1, int eventData2)  
{  
int i,j,k;  
long error;  
long id=-1,*idnum=&id;  
long localID=-1;  
long demo=0,numChannels=2,disableCal=0;  
long transferMode=0;// 做触发模式,0为自动,1为正常,  
2为快速。
```

```

long channels[4]={8,9,10,11};
long gains[4]={0,0,0,0};
float voltages[4096][4]={0};
long stateIOout[4096]={0};
long ljb=-1,*IjScanBacklog=&ljb;
long ov,*overVoltage=&ov;
long numIts = 5;
long junk;
long stateIOin=0 ;
long updateIO=0 ;
long ledOn=1;
long triggerIO=0;// 触发信号选择
long triggerState=0;// 高低电平选择
switch (event)
{
case EVENT_COMMIT:
GetCtrlVal(panelHandle,PANEL_CMD_ADD,&statusofadd);
GetCtrlVal(panelHandle,PANEL_CMD_SUB,&statusofsub);
initial();
error=AIBurst(idnum,demo,stateIOin,updateIO,ledOn,numChannels,channels,gains,scanRate,disableCal,triggerIO,triggerState,numScans,timeout,voltages,stateIOout,overVoltage,transferMode);
for(i=0;i<numScans;i++) wave1[i]=voltages[i][0];
for(i=0;i<numScans;i++) wave2[i]=voltages[i][1];
for(i=0;i<numScans;i++)
{wave11[i]=wave1[i];
wave22[i]=wave2[i];
}
huatu();
break;
}
return 0;
}
void initial()
{

```

```

GetCtrlVal (panelHandle, PANEL_RINGKNOB, &pianyi);
GetCtrlVal (panelHandle, PANEL_RINGKNOB_2,
&zengyi);
GetCtrlVal (panelHandle, PANEL_RINGKNOB_4,
&pianyi2);
GetCtrlVal (panelHandle, PANEL_RINGKNOB_5,
&zengyi2);
GetCtrlVal (panelHandle, PANEL_RINGKNOB_3,
&shuiping);
}
int CVICALLBACK channel1 (int panel, int control, int
event,
void *callbackData, int eventData1, int eventData2)
{
switch (event)
{
case EVENT_COMMIT:
huatu();
break;
}
return 0;
}
int CVICALLBACK channel2 (int panel, int control, int
event,
void *callbackData, int eventData1, int eventData2)
{
switch (event)
{
case EVENT_COMMIT:
huatu();
break;
}
return 0;
}
int CVICALLBACK add (int panel, int control, int event,
void *callbackData, int eventData1, int eventData2)
{

```

```
switch (event)
{
case EVENT_COMMIT:
huatu();
break;
}
return 0;
}
int CVICALLBACK sub (int panel, int control, int event,
void *callbackData, int eventData1, int eventData2)
{
switch (event)
{
case EVENT_COMMIT:
huatu();
break;
}
return 0;
}
void CVICALLBACK huatu()
{
int i,j,k;
/* long error;
long id=-1,*idnum=&id;
long localID=-1;
long demo=0,numChannels=2,disableCal=0;
long transferMode=0;
long channels[4]={8,9,10,11};
long gains[4]={0,0,0,0};
//float sr=1024,*scanRate=&sr;
//long numScans=1024,timeout=30;
float voltages[4096][4]={0};
//long overVoltage[4096][4]={1.0F};
long stateIOout[4096]={0};
long ljb=-1,*ljbScanBacklog=&ljb;
long ov,*overVoltage=&ov;
long numIts = 5;
long junk;
long stateIOin=0;
long updateIO=0;
long ledOn=1;
long triggerIO=0;
long triggerState=0;
//stateIOout=0; */
double currentX_wave1[4096];
double currentY_wave1[4096];
double currentX_wave2[4096];
double currentY_wave2[4096];
double currentX_waveadd[4096];
double currentY_waveadd[4096];
double currentX_wavesub[4096];
double currentY_wavesub[4096];
initial();
//DeleteGraphPlot(panelHandle,PANEL_GRAPH,-1,VAL_
IMMEDIATE_DRAW);
GetCtrlVal(panelHandle,PANEL_CHANNEL_1,&statusofchannel1);
GetCtrlVal(panelHandle,PANEL_CHANNEL_2,&statusofchannel2);
GetCtrlVal(panelHandle,PANEL_CMD_ADD,&statusofadd);
GetCtrlVal(panelHandle,PANEL_CMD_SUB,&statusofsub);
GetCtrlVal(panelHandle,PANEL_DEMO,&statusofdemo);
for(i=0;i<numScans;i++) wave1[i]=voltages[i][0];
for(i=0;i<numScans;i++) wave2[i]=voltages[i][1];
for(i=0;i<numScans;i++)
{wave11[i]=wave1[i];
wave22[i]=wave2[i];
}*/
DeleteGraphPlot(panelHandle,PANEL_GRAPH,-1,VAL_
IMMEDIATE_DRAW);
if(statusofdemo==1)
{
SetCtrlAttribute(panelHandle,PANEL_
```

```

WAVECHOOSE,ATTR_VISIBLE,1);
    SetCtrlAttribute(panelHandle,PANEL_FREQ,ATTR_
VISIBLE,1);
    SetCtrlAttribute(panelHandle,PANEL_VAL,ATTR_
VISIBLE,1);
    SetCtrlAttribute(panelHandle,PANEL_GWAVE,ATTR_
VISIBLE,1);
    SetCtrlAttribute(panelHandle,PANEL_
CLEARPLOT,ATTR_VISIBLE,1);
    SetCtrlAttribute(panelHandle,PANEL_PHASE,ATTR_
VISIBLE,1);
    for(i=0;i<numScans;i++){
        currentX_wave1[i]=i+i*(1/sr);
        currentY_wave1[i]=wave1[i];
        currentY_wave1[i]=currentY_wave1[i]*zengyi+pianyi;
    }
    PlotXY (panelHandle, PANEL_GRAPH, currentX_wave1,
currentY_wave1, numScans, VAL_DOUBLE, VAL_DOUBLE,
VAL_FAT_LINE, VAL_SOLID_SQUARE, VAL_SOLID_
SQUARE, VAL_DASH, VAL_MAGENTA);
    }
    if(statusofdemo==0)
    {
        SetCtrlAttribute(panelHandle,PANEL_
WAVECHOOSE,ATTR_VISIBLE,0);
        SetCtrlAttribute(panelHandle,PANEL_FREQ,ATTR_VISI-
BLE,0);
        SetCtrlAttribute(panelHandle,PANEL_VAL,ATTR_VISI-
BLE,0);
        SetCtrlAttribute(panelHandle,PANEL_GWAVE,ATTR_VISI-
BLE,0);
        SetCtrlAttribute(panelHandle,PANEL_CLEARPLOT,AT-
TR_VISIBLE,0);
        SetCtrlAttribute(panelHandle,PANEL_PHASE,ATTR_VISI-
BLE,0);
        DeleteGraphPlot(panelHandle,PANEL_GRAPH,-1,VAL_
IMMEDIATE_DRAW);
    }
}
if(statusofchannel1==1)// 并且触发源选择了1, 则通道1
开启。
{
    for(i=0;i<numScans;i++)
    {
        currentX_wave1[i]=i+i*(1/sr);
        currentY_wave1[i]=wave1[i];
        currentY_wave1[i]=currentY_wave1[i]*zengyi+pianyi;
    }
    PlotXY (panelHandle, PANEL_GRAPH, currentX_wave1,
currentY_wave1, numScans, VAL_DOUBLE, VAL_DOUBLE,
VAL_FAT_LINE, VAL_SOLID_SQUARE, VAL_SOLID_
SQUARE, VAL_DASH, VAL_GREEN);
    }
    if(statusofchannel2==1)
    {
        for(i=0;i<numScans;i++)
        {
            currentX_wave2[i]=i+i*(1/sr);
            currentY_wave2[i]=wave2[i];
            currentY_wave2[i]=currentY_wave2[i]*zengyi2+pianyi2;
        }
        PlotXY (panelHandle, PANEL_GRAPH, currentX_wave2,
currentY_wave2, numScans, VAL_DOUBLE, VAL_DOUBLE,
VAL_FAT_LINE, VAL_SOLID_SQUARE, VAL_SOLID_
SQUARE, VAL_DASH, VAL_YELLOW);
    }
    if(statusofadd==1)
    {
        for(i=0;i<numScans;i++)
        {
            currentX_waveadd[i]=i+i*(1/sr);
            currentY_waveadd[i]=wave1[i]+wave2[i];
            currentY_waveadd[i]=currentY_waveadd[i]*zengyi+pianyi;
        }
        PlotXY (panelHandle, PANEL_GRAPH, currentX_waveadd,

```

```
currentY_waveadd, numScans, VAL_DOUBLE, VAL_DOUBLE, VAL_FAT_LINE, VAL_SOLID_SQUARE, VAL_SOLID_
VAL_FAT_LINE, VAL_SOLID_SQUARE, VAL_SOLID_ SQUARE, VAL_DASH, VAL_CYAN);
SQUARE, VAL_DASH, VAL_RED);
}
}
if(statusofsub==1)
{
for(i=0;i<numScans;i++)
{
currentX_wavesub[i]=i+i*(1/sr);
currentY_wavesub[i]=wave1[i]-wave2[i];
currentY_wavesub[i]=currentY_wavesub[i]*zengyi+pianyi;
}
PlotXY (panelHandle, PANEL_GRAPH, currentX_wavesub,
currentY_wavesub, numScans, VAL_DOUBLE, VAL_DOUBLE,
VAL_FAT_LINE, VAL_SOLID_SQUARE, VAL_SOLID_
SQUARE, VAL_DASH, VAL_CYAN);
}
}
int CVICALLBACK demo (int panel, int control, int event,
void *callbackData, int eventData1, int eventData2)
{ switch (event)
{
case EVENT_COMMIT:
huatu();
break;
}
return 0;
}
```