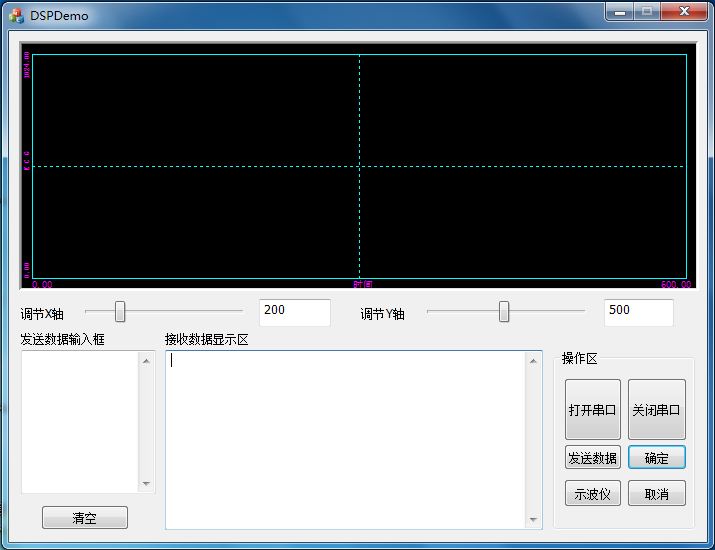
**简易数字示波器的设计**

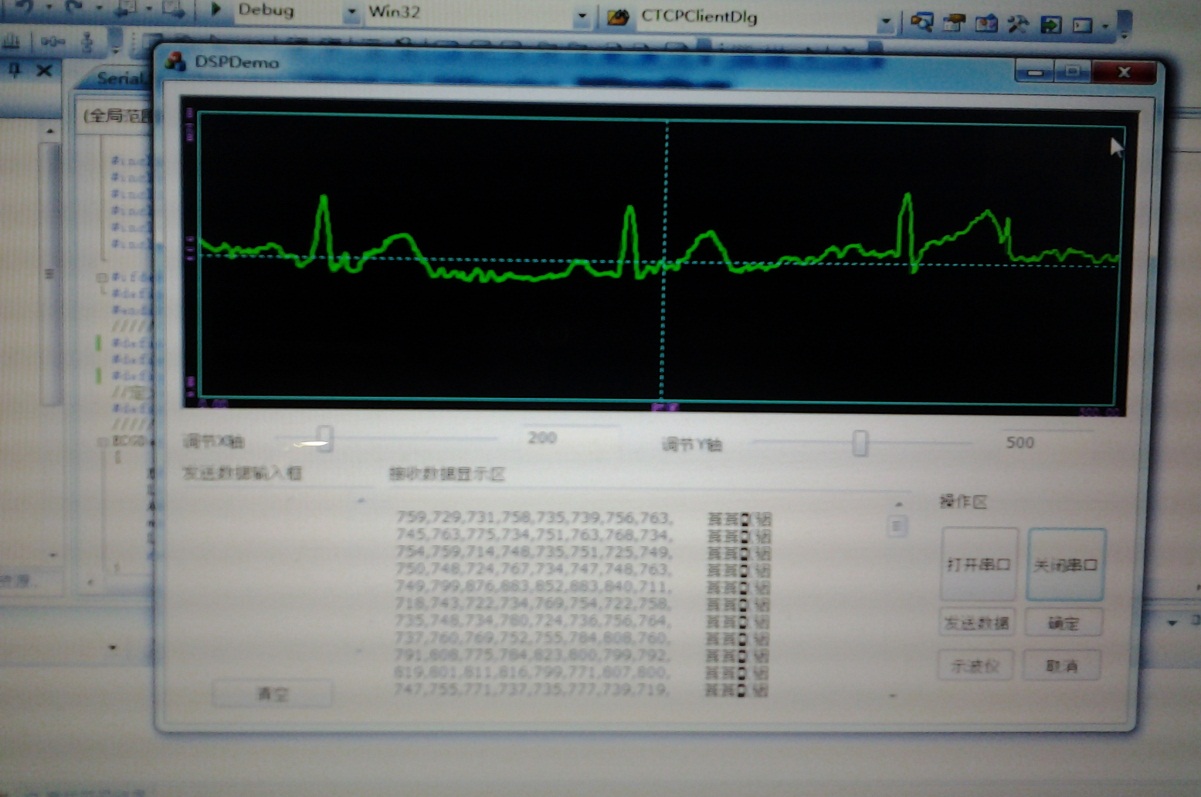
**（软件部分）**

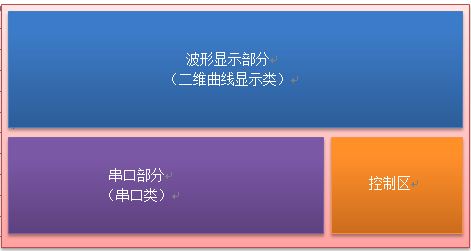
**说明：**

此数字示波器设计的初衷是：因为周末时实验室不开放而且老师不肯借示波器给我们，而周末也正是我们时间充裕的时候，于是，我就想出了利用电脑结合自己所掌握的知识自己开发软件进行简单示波，固然可以用一些虚拟仪器的现成软件，但是学习使用他们也是一种开销，在时间这个成本还有经历这个成本上划不来。

下面就简单介绍一下其编写过程。

**一．软件运行效果图**



**二．软件基本构架**

**说明：**

本软件主要应用了两个类

* 二维曲线显示类
* 串口类

**三．程序源代码**

* /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
* //C2DGraph.h
* //二维曲线显示类
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
* #pragma once
* #include <Afxtempl.h>
* // C2DGraph
* //////////////////////////////////////////
* #define X\_MAX 600 //X轴最大值设定
* #define Y\_MAX 1024 //Y轴最大值设定
* #define PIXEL 2 //曲线像素设定
* //////////////////////////////////////////
* class C2DGraph : public CWnd
* {
* DECLARE\_DYNAMIC(C2DGraph)
* public:
* C2DGraph();
* virtual ~C2DGraph();
* public:
* //控件创建的虚方法覆盖
* virtual BOOL Create(LPCTSTR lpszClassName,
* LPCTSTR lpszWindowName,
* DWORD dwStyle,
* const RECT& rect,
* CWnd\* pParentWnd,
* UINT nID,
* CCreateContext\* pContext) ;
* protected:
* afx\_msg void OnPaint();
* DECLARE\_MESSAGE\_MAP()
* private:
* //将用户添加的点值转换成屏幕坐标
* CPoint GpToSp(CPoint& point);
* //绘制屏幕点
* void DrawPoints();
* public:
* //用于刷新控件显示
* void InvalidateCtrl();
* //在曲线末尾添加一个点
* void AppendPoint(double dwPointY);
* //删除曲线第一点
* void DeleteFirstPoint();
* private:
* //存储线中的点Y坐标
* CList<double,double> m\_lstPointsY;
* //画表格边框设备环境
* CDC m\_dcGrid;
* //表格边框设备环境缓冲位图
* CBitmap m\_bitmapGrid ;
* CBitmap \*m\_pOldBitmapGrid;
* //用来画线的设备环境
* CDC m\_dcLine;
* //画线设备环境缓冲位图
* CBitmap m\_bitmapLine;
* CBitmap \*m\_pOldBitmapLine;
* public:
* COLORREF m\_crBackColor; //控件背景色
* COLORREF m\_crGridColor; //表格边框颜色
* COLORREF m\_crLineColor; //曲线颜色
* COLORREF m\_crTextColor; //输出文本颜色
* double m\_dXMaxValue ; //X轴最大值
* double m\_dXMinValue ; //X轴最小值
* double m\_dYMaxValue ; //Y轴最大值
* double m\_dYMinValue ; //Y轴最小值
* LPCTSTR m\_strXCaption ; //X轴标题
* LPCTSTR m\_strYCaption ; //Y轴标题
* };
* /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
* // 2DGraph.cpp
* //二维曲线显示类
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
* #include "stdafx.h"
* #include "C2DGraph.h"
* // C2DGraph
* IMPLEMENT\_DYNAMIC(C2DGraph, CWnd)
* //构造函数
* C2DGraph::C2DGraph()
* {
* //初始化控件属性
* m\_crBackColor = RGB(0,0,0); //背景色设定
* m\_crGridColor = RGB(0,255,255); //栅格线颜色设定
* m\_crLineColor = RGB(0,255,0); //曲线颜色设定
* m\_crTextColor = RGB(255,0,255); //轴示文本设定
* m\_strXCaption = \_T("时间"); //X轴示文字设定
* m\_strYCaption = \_T("Y X S"); //Y轴示文字设定
* m\_dXMaxValue = X\_MAX; //X轴最大值
* m\_dXMinValue = 0; //X轴最小值
* m\_dYMaxValue = Y\_MAX; //Y轴最大值
* m\_dYMinValue = 0; //Y轴最小值
* m\_pOldBitmapGrid = NULL;
* m\_pOldBitmapLine = NULL;
* }
* //析构函数
* C2DGraph::~C2DGraph()
* {
* //还原GDI对象
* if (m\_dcGrid.GetSafeHdc() != NULL)
* {
* m\_dcGrid.SelectObject(m\_pOldBitmapGrid);
* }
* if (m\_dcLine.GetSafeHdc() != NULL)
* {
* m\_dcLine.SelectObject(m\_pOldBitmapLine);
* }
* }
* BEGIN\_MESSAGE\_MAP(C2DGraph, CWnd)
* ON\_WM\_PAINT()
* END\_MESSAGE\_MAP()
* // C2DGraph 消息处理程序
* BOOL C2DGraph::Create(LPCTSTR lpszClassName, LPCTSTR lpszWindowName, DWORD dwStyle, const RECT& rect, CWnd\* pParentWnd, UINT nID, CCreateContext\* pContext)
* {
* BOOL result ;
* //注册窗体类
* static CString className = AfxRegisterWndClass(CS\_HREDRAW | CS\_VREDRAW) ;
* //创建窗体类
* result = CWnd::CreateEx(WS\_EX\_CLIENTEDGE | WS\_EX\_STATICEDGE,
* className, NULL, dwStyle,
* rect.left, rect.top, rect.right-rect.left, rect.bottom-rect.top,
* pParentWnd->GetSafeHwnd(), (HMENU)nID) ;
* return TRUE;
* }
* //将用户添加的点值转换成屏幕坐标
* CPoint C2DGraph::GpToSp(CPoint& point)
* {
* CPoint rPoint;
* CRect rectClient;
* CRect rectLine;
* GetClientRect(rectClient) ;
* rectLine.left = rectClient.left + 10;
* rectLine.right = rectClient.right - 10;
* rectLine.top = rectClient.top + 10;
* rectLine.bottom = rectClient.bottom - 10;
* rPoint.x = rectLine.left + (point.x / (m\_dXMaxValue - m\_dXMinValue))\*rectLine.Width();
* rPoint.y = rectLine.top + (1 - point.y / (m\_dYMaxValue - m\_dYMinValue))\*rectLine.Height();
* return rPoint;
* }
* //用于刷新控件显示
* void C2DGraph::InvalidateCtrl()
* {
* CPen \*oldPen;
* CPen solidPen(PS\_SOLID,0,m\_crGridColor);
* CPen newPen(PS\_DOT,0,m\_crGridColor);
* CFont xFont,yFont,\*oldFont;
* CBrush brushBack;
* brushBack.CreateSolidBrush(m\_crBackColor) ;
* CRect rectClient;
* GetClientRect(rectClient);
* CClientDC dc(this);
* //创建表格设备环境以及创建相应缓冲区
* if (m\_dcGrid.GetSafeHdc() == NULL)
* {
* m\_dcGrid.CreateCompatibleDC(&dc);
* m\_bitmapGrid.CreateCompatibleBitmap(&dc,rectClient.Width(),rectClient.Height());
* m\_pOldBitmapGrid = m\_dcGrid.SelectObject(&m\_bitmapGrid);
* }
* //设置背景颜色
* m\_dcGrid.SetBkColor(m\_crBackColor);
* m\_dcGrid.FillRect(rectClient,&brushBack);
* //画边框
* oldPen = m\_dcGrid.SelectObject(&solidPen);
* m\_dcGrid.MoveTo(rectClient.left + 10,rectClient.top + 10);
* m\_dcGrid.LineTo(rectClient.right-10,rectClient.top + 10);
* m\_dcGrid.LineTo (rectClient.right-10,rectClient.bottom - 10) ;
* m\_dcGrid.LineTo (rectClient.left + 10 , rectClient.bottom - 10) ;
* m\_dcGrid.LineTo (rectClient.left + 10 , rectClient.top + 10) ;
* //画中央分割线
* oldPen = m\_dcGrid.SelectObject(&newPen);
* m\_dcGrid.MoveTo(rectClient.CenterPoint().x,rectClient.bottom - 10);
* m\_dcGrid.LineTo(rectClient.CenterPoint().x,rectClient.top + 10);
* m\_dcGrid.MoveTo(rectClient.left + 10,rectClient.CenterPoint().y);
* m\_dcGrid.LineTo(rectClient.right-10,rectClient.CenterPoint().y);
* //还原GDI对象
* m\_dcGrid.SelectObject(oldPen);
* //创建Y轴字体
* yFont.CreateFont (8, 0, 900, 0, 400,
* FALSE, FALSE, 0, ANSI\_CHARSET,
* OUT\_DEFAULT\_PRECIS,
* CLIP\_DEFAULT\_PRECIS,
* DEFAULT\_QUALITY,
* DEFAULT\_PITCH|FF\_SWISS, \_T("宋体")) ;
* //创建X轴字体
* xFont.CreateFont (10, 0, 0, 0, 300,
* FALSE, FALSE, 0, ANSI\_CHARSET,
* OUT\_DEFAULT\_PRECIS,
* CLIP\_DEFAULT\_PRECIS,
* DEFAULT\_QUALITY,
* DEFAULT\_PITCH|FF\_SWISS, \_T("宋体")) ;
* //画Y坐标标题
* m\_dcGrid.SetTextColor(m\_crTextColor);
* oldFont = m\_dcGrid.SelectObject(&yFont);
* m\_dcGrid.ExtTextOut (rectClient.left,
* (rectClient.top + rectClient.bottom ) / 2 + 5, ETO\_CLIPPED,NULL,m\_strYCaption,wcslen(m\_strYCaption),NULL) ;
* //画Y坐标最大值
* CString strTmp;
* strTmp.Format(\_T("%.2f"),m\_dYMaxValue);
* m\_dcGrid.ExtTextOut (rectClient.left ,
* rectClient.top + 35 , ETO\_CLIPPED,NULL
* ,strTmp,strTmp.GetLength(),NULL) ;
* //画Y坐标最小值
* strTmp.Format(\_T("%.2f"),m\_dYMinValue);
* m\_dcGrid.ExtTextOut (rectClient.left ,
* rectClient.bottom - 10 , ETO\_CLIPPED,NULL
* ,strTmp,strTmp.GetLength(),NULL) ;
* //还原GDI对象
* m\_dcGrid.SelectObject(oldPen);
* //画X坐标标题
* oldFont = m\_dcGrid.SelectObject(&xFont);
* m\_dcGrid.ExtTextOut ((rectClient.left+ rectClient.right)/2 - 6,
* rectClient.bottom - 9 , ETO\_CLIPPED,NULL,m\_strXCaption,wcslen(m\_strXCaption) ,NULL) ;
* //画X坐标最大值
* strTmp.Format(\_T("%.2f"),m\_dXMaxValue);
* m\_dcGrid.ExtTextOut (rectClient.right - 35,
* rectClient.bottom - 9 , ETO\_CLIPPED,NULL
* ,strTmp,strTmp.GetLength(),NULL) ;
* //画X坐标最小值
* strTmp.Format(\_T("%.2f"),m\_dXMinValue);
* m\_dcGrid.ExtTextOut (rectClient.left + 10,
* rectClient.bottom - 9 , ETO\_CLIPPED,NULL
* ,strTmp,strTmp.GetLength(),NULL) ;
* //还原GDI对象
* m\_dcGrid.SelectObject(oldPen);
* //创建画线设备环境以及创建相应缓冲区
* if (m\_dcLine.GetSafeHdc() == NULL)
* {
* m\_dcLine.CreateCompatibleDC(&dc) ;
* m\_bitmapLine.CreateCompatibleBitmap(&dc, rectClient.Width(), rectClient.Height()) ;
* m\_pOldBitmapLine = m\_dcLine.SelectObject(&m\_bitmapLine) ;
* }
* m\_dcLine.SetBkColor (m\_crBackColor) ;
* m\_dcLine.FillRect(rectClient, &brushBack) ;
* //删除创建的GDI对象
* solidPen.DeleteObject();
* xFont.DeleteObject();
* yFont.DeleteObject();
* brushBack.DeleteObject();
* }
* //绘制屏幕点
* void C2DGraph::DrawPoints()
* {
* CPen \*oldPen;
* CRect rectCleanUp;
* CPoint points[2];
* CPoint point\_old;
* CPen penLine;
* //创建画笔
* penLine.CreatePen(PS\_SOLID, PIXEL, m\_crLineColor) ;
* //得到绘图区域
* CRect rectClient;
* GetClientRect(rectClient);
* m\_dcLine.FillSolidRect(rectClient,m\_crBackColor);
* int i = 0;
* oldPen = m\_dcLine.SelectObject(&penLine) ;
* if (m\_dcLine.GetSafeHdc() == NULL)
* {
* return;
* }
* //绘制曲线
* if (m\_lstPointsY.GetCount() == 0 )
* {
* }
* //如果是第个点
* else if (m\_lstPointsY.GetCount() == 1)
* {
* POSITION pos = m\_lstPointsY.GetHeadPosition();
* points[0].y = m\_lstPointsY.GetAt(pos);
* points[0].x = 0; //表示第个点
* points[0] = GpToSp(points[0]); //转换成屏幕坐标
* m\_dcLine.SetPixel(points[0],m\_crLineColor);
* }
* //如果多于或等于个点
* else
* {
* POSITION pos = m\_lstPointsY.GetHeadPosition();
* points[0].y = m\_lstPointsY.GetNext(pos);
* points[0].x = 0; //表示第个点
* point\_old = points[0];
* i = 0;
* while (TRUE)
* {
* if (pos == NULL)
* {
* break;
* }
* points[1] = point\_old;
* points[0].y = m\_lstPointsY.GetNext(pos);
* points[0].x = i+1;
* point\_old = points[0];
* points[0] = GpToSp(points[0]);
* points[1] = GpToSp(points[1]);
* //绘制曲线
* m\_dcLine.MoveTo (points[1].x, points[1].y) ;
* m\_dcLine.LineTo (points[0].x,points[0].y) ;
* i++;
* }
* }
* m\_dcLine.SelectObject(oldPen) ;
* //删除画笔GDI对象
* penLine.DeleteObject();
* }
* //覆盖WM\_PAINT消息方法
* void C2DGraph::OnPaint()
* {
* CPaintDC dc(this); // device context for painting
* CRect rectClient;
* GetClientRect(rectClient) ;
* CDC memDC ;
* CBitmap memBitmap ;
* CBitmap\* oldBitmap ;
* memDC.CreateCompatibleDC(&dc) ;
* memBitmap.CreateCompatibleBitmap(&dc, rectClient.Width(), rectClient.Height()) ;
* oldBitmap = (CBitmap \*)memDC.SelectObject(&memBitmap) ;
* //更新背景显示
* InvalidateCtrl() ;
* //画点
* DrawPoints();
* //将m\_dcGrid和m\_dcLine绘制到控件上
* if (memDC.GetSafeHdc() != NULL)
* {
* memDC.BitBlt(0, 0, rectClient.Width(), rectClient.Height(),
* &m\_dcGrid, 0, 0, SRCCOPY) ;
* memDC.BitBlt(0, 0, rectClient.Width(), rectClient.Height(),
* &m\_dcLine, 0, 0, SRCPAINT) ;
* dc.BitBlt(0, 0, rectClient.Width(), rectClient.Height(),
* &memDC, 0, 0, SRCCOPY) ;
* }
* memDC.SelectObject(oldBitmap) ;
* //删除内存位图GDI对象
* memBitmap.DeleteObject();
* //删除内存绘图环境
* memDC.DeleteDC();
* }
* //在曲线末尾添加一个点
* void C2DGraph::AppendPoint(double dwPointY)
* {
* //在队列中添加个点
* m\_lstPointsY.AddTail(dwPointY);
* //更新显示
* Invalidate();
* }
* //删除曲线第一点
* void C2DGraph::DeleteFirstPoint()
* {
* //删除个节点
* if (m\_lstPointsY.GetCount() > 0)
* {
* m\_lstPointsY.RemoveHead();
* }
* //更新显示
* Invalidate();
* }
* /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
* // DSPDemoDlg.h
* //窗体
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
* #pragma once
* #include "C2DGraph.h"
* #include "Serial.h"
* #include "afxcmn.h"
* //每个链表节点所能容纳的最大数据量
* #define ECG\_SIZE 400
* ////////////////////////////////////////
* //串口数据暂存数据结构
* typedef struct ECGDataNode
* {
* int ECG[ECG\_SIZE];
* struct ECGDataNode \*next;
* }ECGDataNode;
* ///////////////////////////////////////
* ECGDataNode \*InitECGDataNode();
* // CDSPDemoDlg 对话框
* class CDSPDemoDlg : public CDialog
* {
* // 构造
* public:
* CDSPDemoDlg(CWnd\* pParent = NULL); // 标准构造函数
* // 对话框数据
* enum { IDD = IDD\_DSPDEMO\_DIALOG };
* protected:
* virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持
* // 实现
* protected:
* HICON m\_hIcon;
* // 生成的消息映射函数
* virtual BOOL OnInitDialog();
* #if defined(\_DEVICE\_RESOLUTION\_AWARE) && !defined(WIN32\_PLATFORM\_WFSP)
* afx\_msg void OnSize(UINT /\*nType\*/, int /\*cx\*/, int /\*cy\*/);
* #endif
* afx\_msg void OnSysCommand(UINT nID, LPARAM lParam);
* afx\_msg void OnPaint();
* afx\_msg HCURSOR OnQueryDragIcon();
* // 串口接收数据处理函数
* afx\_msg LONG OnRecvSerialData(WPARAM wParam,LPARAM lParam);
* DECLARE\_MESSAGE\_MAP()
* /\*信号显示\*/
* private:
* C2DGraph m\_2DGraph; //二维曲线对象
* int m\_pointCount; //曲线点计数
* int ecgpoint;
* double y[501],y2[501];
* /////////////////////////////////////////////////////
* CSerial\* pCSerial;
* static void CALLBACK OnSerialRead(DWORD UserParam,
* char\* buf,
* DWORD bufLen);
* static DWORD WINAPI ThreadProc(LPVOID lpParameter);
* public:
* afx\_msg void OnTimer(UINT\_PTR nIDEvent);//定时器消息处理函数
* afx\_msg void OnBnClickedButton1();
* afx\_msg void OnBnClickedButton2();
* afx\_msg void OnBnClickedButton3();
* afx\_msg void OnBnClickedOk();
* afx\_msg void OnBnClickedCancel();
* afx\_msg void OnBnClickedButton4();
* afx\_msg void OnBnClickedButton5();
* CSliderCtrl m\_ctrlSliderX;
* CSliderCtrl m\_ctrlSliderY;
* afx\_msg void OnNMCustomdrawSliderx(NMHDR \*pNMHDR, LRESULT \*pResult);
* afx\_msg void OnNMCustomdrawSlidery(NMHDR \*pNMHDR, LRESULT \*pResult);
* };
* // DSPDemoDlg.cpp : 实现文件
* //
* #include "stdafx.h"
* #include "DSPDemo.h"
* #include "DSPDemoDlg.h"
* #include "DlgParams.h"
* #include "DSP.h"
* #include "math.h"
* #ifdef \_DEBUG
* #define new DEBUG\_NEW
* #endif
* ////////////////////////////////////////////////////
* #define MAX\_SIZE 1000
* #define MIN\_SIZE 100
* #define N 7
* static int ecg;//要显示的点的Y坐标
* //定义串口数据接收消息常量
* #define WM\_RECV\_SERIAL\_DATA WM\_USER + 101
* ////////////////////////////////////////////////////
* ECGDataNode \*InitECGDataNode()
* {
* ECGDataNode \*L;
* L=(ECGDataNode\*)malloc(sizeof(ECGDataNode));
* ASSERT(L!=NULL);
* memset(L->ECG,0,ECG\_SIZE\*sizeof(int));
* L->next = NULL;
* return L;
* }
* ///////////////////////////////////////////////////
* ECGDataNode \*ECG = InitECGDataNode();
* ECGDataNode \*Ptr\_ECG = ECG;
* ECGDataNode \*Ptr\_Next\_ECG = ECG;
* //////////////////////////////////////////////////
* // 用于应用程序“关于”菜单项的CAboutDlg 对话框
* class CAboutDlg : public CDialog
* {
* public:
* CAboutDlg();
* // 对话框数据
* enum { IDD = IDD\_ABOUTBOX };
* protected:
* virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持
* // 实现
* protected:
* DECLARE\_MESSAGE\_MAP()
* };
* CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
* {
* }
* void CAboutDlg::DoDataExchange(CDataExchange\* pDX)
* {
* CDialog::DoDataExchange(pDX);
* }
* BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialog)
* END\_MESSAGE\_MAP()
* // CDSPDemoDlg 对话框
* CDSPDemoDlg::CDSPDemoDlg(CWnd\* pParent /\*=NULL\*/)
* : CDialog(CDSPDemoDlg::IDD, pParent)
* {
* m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);
* pCSerial = NULL;
* }
* void CDSPDemoDlg::DoDataExchange(CDataExchange\* pDX)
* {
* CDialog::DoDataExchange(pDX);
* DDX\_Control(pDX, IDC\_SLIDERX, m\_ctrlSliderX);
* DDX\_Control(pDX, IDC\_SLIDERY, m\_ctrlSliderY);
* }
* BEGIN\_MESSAGE\_MAP(CDSPDemoDlg, CDialog)
* ON\_WM\_TIMER()
* ON\_WM\_SYSCOMMAND()
* ON\_WM\_PAINT()
* ON\_WM\_QUERYDRAGICON()
* ON\_MESSAGE(WM\_RECV\_SERIAL\_DATA,OnRecvSerialData)
* //}}AFX\_MSG\_MAP
* ON\_BN\_CLICKED(IDC\_BUTTON1, &CDSPDemoDlg::OnBnClickedButton1)
* ON\_BN\_CLICKED(IDC\_BUTTON2, &CDSPDemoDlg::OnBnClickedButton2)
* ON\_BN\_CLICKED(IDC\_BUTTON3, &CDSPDemoDlg::OnBnClickedButton3)
* ON\_BN\_CLICKED(IDOK, &CDSPDemoDlg::OnBnClickedOk)
* ON\_BN\_CLICKED(IDCANCEL, &CDSPDemoDlg::OnBnClickedCancel)
* ON\_BN\_CLICKED(IDC\_BUTTON4, &CDSPDemoDlg::OnBnClickedButton4)
* ON\_BN\_CLICKED(IDC\_BUTTON5, &CDSPDemoDlg::OnBnClickedButton5)
* ON\_NOTIFY(NM\_CUSTOMDRAW, IDC\_SLIDERX, &CDSPDemoDlg::OnNMCustomdrawSliderx)
* ON\_NOTIFY(NM\_CUSTOMDRAW, IDC\_SLIDERY, &CDSPDemoDlg::OnNMCustomdrawSlidery)
* END\_MESSAGE\_MAP()
* // CDSPDemoDlg 消息处理程序
* BOOL CDSPDemoDlg::OnInitDialog()
* {
* CDialog::OnInitDialog();
* // 将“关于...”菜单项添加到系统菜单中。
* // IDM\_ABOUTBOX 必须在系统命令范围内。
* ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);
* ASSERT(IDM\_ABOUTBOX < 0xF000);
* CMenu\* pSysMenu = GetSystemMenu(FALSE);
* if (pSysMenu != NULL)
* {
* BOOL bNameValid;
* CString strAboutMenu;
* bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);
* ASSERT(bNameValid);
* if (!strAboutMenu.IsEmpty())
* {
* pSysMenu->AppendMenu(MF\_SEPARATOR);
* pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);
* }
* }
* // 设置此对话框的图标。当应用程序主窗口不是对话框时，框架将自动
* // 执行此操作
* SetIcon(m\_hIcon, TRUE); // 设置大图标
* SetIcon(m\_hIcon, FALSE); // 设置小图标
* ShowWindow(SW\_MINIMIZE);
* // TODO: 在此添加额外的初始化代码
* //初始化显示控件矩形区域大小
* CRect rect;
* rect.left = 10;
* rect.top = 10;
* rect.right = 690;
* rect.bottom = 260;
* //创建曲线控件实例
* m\_2DGraph.Create(\_T(""),\_T(""),WS\_VISIBLE | WS\_CHILD, rect, this,0,NULL) ;
* ////////////////////////////////////////////
* m\_pointCount = 0;
* ecgpoint = 0;
* ecg = 0;
* //启动添加点计时器
* // SetTimer(1,20,NULL);
* /////////////////////////////////////////////////////
* m\_ctrlSliderX.SetRange(0,1000);
* m\_ctrlSliderX.SetPos(200);
* m\_ctrlSliderY.SetRange(0,1024);
* m\_ctrlSliderY.SetPos(500);
* return TRUE; // 除非将焦点设置到控件，否则返回TRUE
* }
* void CDSPDemoDlg::OnSysCommand(UINT nID, LPARAM lParam)
* {
* if ((nID & 0xFFF0) == IDM\_ABOUTBOX)
* {
* CAboutDlg dlgAbout;
* dlgAbout.DoModal();
* }
* else
* {
* CDialog::OnSysCommand(nID, lParam);
* }
* }
* // 如果向对话框添加最小化按钮，则需要下面的代码
* // 来绘制该图标。对于使用文档/视图模型的MFC 应用程序，
* // 这将由框架自动完成。
* void CDSPDemoDlg::OnPaint()
* {
* if (IsIconic())
* {
* CPaintDC dc(this); // 用于绘制的设备上下文
* SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);
* // 使图标在工作区矩形中居中
* int cxIcon = GetSystemMetrics(SM\_CXICON);
* int cyIcon = GetSystemMetrics(SM\_CYICON);
* CRect rect;
* GetClientRect(&rect);
* int x = (rect.Width() - cxIcon + 1) / 2;
* int y = (rect.Height() - cyIcon + 1) / 2;
* // 绘制图标
* dc.DrawIcon(x, y, m\_hIcon);
* }
* else
* {
* CDialog::OnPaint();
* }
* }
* //当用户拖动最小化窗口时系统调用此函数取得光标
* //显示。
* HCURSOR CDSPDemoDlg::OnQueryDragIcon()
* {
* return static\_cast<HCURSOR>(m\_hIcon);
* }
* DWORD CDSPDemoDlg::ThreadProc(LPVOID lpParameter)
* {
* CDSPDemoDlg\* args = NULL;
* args = (CDSPDemoDlg\*)lpParameter;
* // FILE\* fp;
* // fp = fopen("ECG.txt","r");
* return 1;
* }
* //定时器消息处理函数
* void CDSPDemoDlg::OnTimer(UINT\_PTR nIDEvent)
* {
* // TODO: 在此添加消息处理程序代码和/或调用默认值
* int nRandomY;//要显示的点的Y坐标
* // //坐标的赋值和滤波函数
* {//平滑滤波
* int ECGtemp = 0;
* for(int i=0;i<N;i++)
* {
* ECGtemp += Ptr\_ECG->ECG[(ecgpoint+i)%ECG\_SIZE];
* }
* Ptr\_ECG->ECG[ecgpoint] = (int)(ECGtemp/N);
* }
* Ptr\_ECG->ECG[ecgpoint] = (Ptr\_ECG->ECG[ecgpoint]-700)\*5;
* // nRandomY = (Ptr\_ECG->ECG[ecgpoint]-700)\*5;
* // Ptr\_ECG->ECG[ecgpoint] = ((Ptr\_ECG->ECG[ecgpoint])\*5/1024)\*1000;
* nRandomY = Ptr\_ECG->ECG[ecgpoint];
* if(nRandomY<100)
* {
* nRandomY = 100;
* }
* /////////////////////////
* ecgpoint++;
* if(ecgpoint == ECG\_SIZE)
* {
* if(Ptr\_ECG->next != NULL)
* {
* ecgpoint=0;
* Ptr\_ECG = Ptr\_ECG->next;
* }
* else
* {
* KillTimer(1);
* }
* }
* //如果曲线点数大于X\_MAX个点，则删除第个点。
* if (m\_pointCount > X\_MAX )
* {
* m\_2DGraph.DeleteFirstPoint();
* m\_pointCount--;
* }
* //给曲线添加点
* m\_2DGraph.AppendPoint(nRandomY);
* TRACE(L" y = %d \n",nRandomY);
* m\_pointCount++;
* /////////////////////////
* CDialog::OnTimer(nIDEvent);
* }
* void CALLBACK CDSPDemoDlg::OnSerialRead(DWORD UserParamr,char\* buf,DWORD bufLen)
* {
* char \*pRecvBuf; //接收缓冲区
* //得到父对象指针
* CDSPDemoDlg\* pThis = (CDSPDemoDlg\*)UserParamr;
* //将接收的缓冲区拷贝到pRecvBuf种
* pRecvBuf = new char[bufLen];
* ZeroMemory(pRecvBuf,bufLen);
* MoveMemory(pRecvBuf,buf,bufLen);
* //////////////////////////////////////////////////////
* CString strRecv;
* strRecv = CString(buf);
* DWORD id;
* HANDLE handle;
* handle = CreateThread(NULL,0,ThreadProc,NULL,0,&id);
* if(strRecv.Find(\_T(","),0)>=0)
* {
* //////////////////////////////////////
* //处理接收到的数据
* CString num = L"";
* int pos = 0;
* while (!strRecv.IsEmpty())
* {
* int arg = 0;
* pos = strRecv.Find(\_T(","),0);
* if(pos == 0)
* {//","
* strRecv.Delete(0,pos+1);
* continue;
* }
* if(pos<0)
* {//只剩一个数
* num = strRecv;
* strRecv.Empty();
* Ptr\_Next\_ECG->ECG[ecg] = \_ttoi(num);
* if(Ptr\_Next\_ECG->ECG[ecg]<MIN\_SIZE||Ptr\_Next\_ECG->ECG[ecg]>MAX\_SIZE)
* Ptr\_Next\_ECG->ECG[ecg] = Ptr\_Next\_ECG->ECG[ecg-1];
* ecg++;
* if(ecg ==ECG\_SIZE)
* {
* ecg = 0;
* Ptr\_Next\_ECG->next = InitECGDataNode();
* Ptr\_Next\_ECG = Ptr\_Next\_ECG->next;
* }
* }
* else
* {
* num = strRecv.Left(pos);
* int len = num.GetLength();
* strRecv.Delete(0,pos+1);
* Ptr\_Next\_ECG->ECG[ecg] = \_ttoi(num);
* if(Ptr\_Next\_ECG->ECG[ecg]<MIN\_SIZE||Ptr\_Next\_ECG->ECG[ecg]>MAX\_SIZE)
* Ptr\_Next\_ECG->ECG[ecg] = Ptr\_Next\_ECG->ECG[ecg-1];
* ecg++;
* if(ecg == ECG\_SIZE)
* {
* ecg = 0;
* Ptr\_Next\_ECG->next = InitECGDataNode();
* Ptr\_Next\_ECG = Ptr\_Next\_ECG->next;
* }
* }
* }
* //////////////////////////////////////
* }
* //发送异步消息，表示收到串口数据，消息处理完，应释放内存
* pThis->PostMessage(WM\_RECV\_SERIAL\_DATA,WPARAM(pRecvBuf),bufLen);
* }
* // 串口接收数据处理函数
* LONG CDSPDemoDlg::OnRecvSerialData(WPARAM wParam,LPARAM lParam)
* {
* CString strOldRecv = L"";
* CString strRecv = L"";
* //串口接收到的BUF
* CHAR \*pBuf = (CHAR\*)wParam;
* //串口接收到的BUF长度
* DWORD dwBufLen = lParam;
* //接收框
* CEdit \*pEdtRecvMsg = (CEdit\*)GetDlgItem(IDC\_EDT\_RECV);
* ASSERT(pEdtRecvMsg != NULL);
* //得到接收框中的历史文本
* pEdtRecvMsg->GetWindowTextW(strOldRecv);
* strRecv = CString(pBuf);
* //将新接收到的文本添加到接收框中
* strOldRecv = strOldRecv+\_T("\r\n")+ strRecv;
* pEdtRecvMsg->SetWindowTextW(strOldRecv);
* if(strOldRecv.GetLength()>=2048)
* {
* pEdtRecvMsg->SetWindowTextW(\_T(""));
* }
* /\* if(strRecv.Find(\_T(","),0)>=0)
* {
* //////////////////////////////////////
* //处理接收到的数据
* CString num = L"";
* int pos = 0;
* while (!strRecv.IsEmpty())
* {
* int arg = 0;
* pos = strRecv.Find(\_T(","),0);
* if(pos == 0)
* {//","
* strRecv.Delete(0,pos+1);
* continue;
* }
* if(pos<0)
* {//只剩一个数
* num = strRecv;
* strRecv.Empty();
* Ptr\_Next\_ECG->ECG[ecg] = \_ttoi(num);
* if(Ptr\_Next\_ECG->ECG[ecg]<MIN\_SIZE||Ptr\_Next\_ECG->ECG[ecg]>MAX\_SIZE)
* Ptr\_Next\_ECG->ECG[ecg] = Ptr\_Next\_ECG->ECG[ecg-1];
* ecg++;
* if(ecg ==60)
* {
* ecg = 0;
* Ptr\_Next\_ECG->next = InitECGDataNode();
* Ptr\_Next\_ECG = Ptr\_Next\_ECG->next;
* }
* }
* else
* {
* num = strRecv.Left(pos);
* int len = num.GetLength();
* strRecv.Delete(0,pos+1);
* Ptr\_Next\_ECG->ECG[ecg] = \_ttoi(num);
* if(Ptr\_Next\_ECG->ECG[ecg]<MIN\_SIZE||Ptr\_Next\_ECG->ECG[ecg]>MAX\_SIZE)
* Ptr\_Next\_ECG->ECG[ecg] = Ptr\_Next\_ECG->ECG[ecg-1];
* ecg++;
* if(ecg ==60)
* {
* ecg = 0;
* Ptr\_Next\_ECG->next = InitECGDataNode();
* Ptr\_Next\_ECG = Ptr\_Next\_ECG->next;
* }
* }
* }
* //////////////////////////////////////
* }\*/
* //释放内存
* delete[] pBuf;
* pBuf = NULL;
* return 0;
* }
* void CDSPDemoDlg::OnBnClickedButton1()
* {
* // TODO: 在此添加控件通知处理程序代码
* //判断串口是否已经打开
* if (pCSerial != NULL)
* {
* pCSerial->Close();
* delete pCSerial;
* pCSerial = NULL;
* }
* //串口参数输入对话框
* CDlgParams dlgParams;
* if (dlgParams.DoModal() == IDOK)
* {
* TCHAR szPort[15];
* wsprintf(szPort, L"COM%d:",dlgParams.m\_portNo);
* pCSerial = new CSerial;
* pCSerial->m\_OnSeriesRead = OnSerialRead;//串口接收成功回调函数
* BOOL ret;
* ret = pCSerial->Open((DWORD)this,
* szPort,
* dlgParams.m\_baud,
* 8,
* 0,
* 1);
* if (ret == FALSE) //打开串口, 数据位为,停止位为,无校验位
* {
* AfxMessageBox(L"串口打开失败");
* }
* else
* {
* AfxMessageBox(L"串口打开成功");
* }
* }
* }
* void CDSPDemoDlg::OnBnClickedButton2()
* {
* // TODO: 在此添加控件通知处理程序代码
* if (pCSerial != NULL)
* {
* pCSerial->Close();
* delete pCSerial;
* pCSerial = NULL;
* }
* }
* void CDSPDemoDlg::OnBnClickedButton3()
* {
* // TODO: 在此添加控件通知处理程序代码
* char \* buf =NULL; //定义发送缓冲区
* DWORD dwBufLen = 0; //定义发送缓冲区长度
* CString strSend = L"";
* //得到发送输入框
* CEdit \*pEdtSendMsg = (CEdit\*)GetDlgItem(IDC\_EDT\_SEND);
* ASSERT(pEdtSendMsg != NULL);
* //串口如果没有打开，直接返回
* if (pCSerial == NULL)
* {
* AfxMessageBox(L"串口未打开");
* return;
* }
* //得到待发送的字符串
* pEdtSendMsg->GetWindowTextW(strSend);
* //将待发送的字符串转换成单字节，进行发送
* buf = new char[strSend.GetLength()\*2+1];
* ZeroMemory(buf,strSend.GetLength()\*2+1);
* //转换成单字节进行发送
* WideCharToMultiByte(CP\_ACP,WC\_COMPOSITECHECK,
* strSend.GetBuffer(strSend.GetLength()),
* strSend.GetLength(),
* buf,
* strSend.GetLength()\*2,
* NULL,NULL);
* dwBufLen = strlen(buf)+1;
* //发送字符串
* pCSerial->SendData(buf,dwBufLen);
* //释放内存
* delete[] buf;
* buf = NULL;
* }
* void CDSPDemoDlg::OnBnClickedOk()
* {
* // TODO: 在此添加控件通知处理程序代码
* OnOK();
* }
* void CDSPDemoDlg::OnBnClickedCancel()
* {
* // TODO: 在此添加控件通知处理程序代码
* OnCancel();
* }
* void CDSPDemoDlg::OnBnClickedButton4()
* {
* // TODO: 在此添加控件通知处理程序代码
* //输入框清空
* CEdit \*pEdtSendMsg = (CEdit\*)GetDlgItem(IDC\_EDT\_SEND);
* ASSERT(pEdtSendMsg != NULL);
* pEdtSendMsg->SetWindowTextW(\_T(""));
* //接收框清空
* CEdit \*pEdtRecvMsg = (CEdit\*)GetDlgItem(IDC\_EDT\_RECV);
* ASSERT(pEdtRecvMsg != NULL);
* //得到接收框中的历史文本
* pEdtRecvMsg->SetWindowTextW(\_T(""));
* }
* void CDSPDemoDlg::OnBnClickedButton5()
* {
* //启动添加点计时器
* SetTimer(1,20,NULL);
* }
* void CDSPDemoDlg::OnNMCustomdrawSliderx(NMHDR \*pNMHDR, LRESULT \*pResult)
* {
* LPNMCUSTOMDRAW pNMCD = reinterpret\_cast<LPNMCUSTOMDRAW>(pNMHDR);
* // TODO: 在此添加控件通知处理程序代码
* \*pResult = 0;
* int nPos = m\_ctrlSliderX.GetPos(); //获得滑块的当前位置
* CString str=L"";
* str.Format(L"%d",nPos);
* SetDlgItemText(IDC\_EDITX,str);
* }
* void CDSPDemoDlg::OnNMCustomdrawSlidery(NMHDR \*pNMHDR, LRESULT \*pResult)
* {
* LPNMCUSTOMDRAW pNMCD = reinterpret\_cast<LPNMCUSTOMDRAW>(pNMHDR);
* // TODO: 在此添加控件通知处理程序代码
* \*pResult = 0;
* int nPos = m\_ctrlSliderY.GetPos(); //获得滑块的当前位置
* CString str=L"";
* str.Format(L"%d",nPos);
* SetDlgItemText(IDC\_EDITY,str);
* }