```
30        double    m_fT_wet;
31        CString   m_szCh0_id;
32        CString   m_szCh1_id;
33        int       m_nAmp_range;
34        //}}AFX_DATA
35
36    // Overrides
37        // ClassWizard generated virtual function overrides
38        //{{AFX_VIRTUAL(CPara)
39        protected:
40        virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
41        //}}AFX_VIRTUAL
42
43    // Implementation
44    protected:
45
46        // Generated message map functions
47        //{{AFX_MSG(CPara)
48            // NOTE: the ClassWizard will add member functions here
49        //}}AFX_MSG
50        DECLARE_MESSAGE_MAP()
51    };
52
53    //{{AFX_INSERT_LOCATION}}
54    // Microsoft Visual C++ will insert additional declarations immediately before the previous line.
55
56    #endif // !defined(AFX_PARA_H__CC5DE021_541F_11D3_B89F_0080C875FA7E__INCLUDED_)
```

4、文件 Para.cpp

```
1     // Para.cpp : implementation file
2     //
3
4     #include "stdafx.h"
5     #include "solartester.h"
6     #include "Para.h"
7     #include "stdio.h"
8
9     #ifdef _DEBUG
10    #define new DEBUG_NEW
11    #undef THIS_FILE
12    static char THIS_FILE[] = __FILE__;
13    #endif
14
15    /////////////////////////////////////////////////////////////////////////////
16    // CPara dialog
17
18    CPara::CPara(CWnd* pParent /*=NULL*/)
19        : CDialog(CPara::IDD, pParent)
20    {
21        //{{AFX_DATA_INIT(CPara)
22        char str0[25];
23        char str1[25];
24
```

```
25        FILE* fp1;
26        fp1=fopen("PARAS.DAP","r");
27
28        if(fp1 != NULL)
29        {
30             fscanf(fp1,"m_szCh0_id = %s\n\r",str0);
31             fscanf(fp1,"m_szCh1_id = %s\n\r",str1);
32             fscanf(fp1,"m_nAverage = %d\n",&m_nAverage);
33             fscanf(fp1,"m_fVmax = %lf\n",&m_fVmax);
34             fscanf(fp1,"m_nSample = %d\n",&m_nSample);
35             fscanf(fp1,"m_nDelay = %d\n",&m_nDelay);
36             fscanf(fp1,"m_fSun = %lf\n",&m_fSun);
37             fscanf(fp1,"m_fSize = %lf\n",&m_fSize);
38             fscanf(fp1,"m_fT_dry = %lf\n",&m_fT_dry);
39             fscanf(fp1,"m_fT_wet = %lf\n",&m_fT_wet);
40             fscanf(fp1,"m_fHum = %lf\n",&m_fHum);
41             fscanf(fp1,"m_nAmp_range = %d\n",&m_nAmp_range);
42             fclose(fp1);
43
44             m_szCh0_id = str0;
45             m_szCh1_id = str1;
46        }
47        else
48        {
49             m_szCh0_id = "ch0";
50             m_szCh1_id = "ch1";
51             m_nAverage = 100;
52             m_fVmax = 10.0;
53             m_nSample =100 ;
54             m_nDelay = 0;
55             m_fSun = 0.5;
56             m_fSize = 7.0;
57             m_fT_dry = 20;
58             m_fT_wet = 20;
59             m_fHum = 0.0;
60             m_nAmp_range = 0;
61        }
62        //}}AFX_DATA_INIT
63   }
64
65   void CPara::DoDataExchange(CDataExchange* pDX)
66   {
67        CDialog::DoDataExchange(pDX);
68        //{{AFX_DATA_MAP(CPara)
69        DDX_Text(pDX, IDC_AVERAGE, m_nAverage);
70        DDV_MinMaxUInt(pDX, m_nAverage, 1, 10000);
71        DDX_Text(pDX, IDC_VMAX, m_fVmax);
72        DDV_MinMaxFloat(pDX, m_fVmax, 0.f, 10.0f);
73        DDX_Text(pDX, IDC_SAMPLE, m_nSample);
74        DDV_MinMaxUInt(pDX, m_nSample, 2, 1000);
75        DDX_Text(pDX, IDC_DELAY, m_nDelay);
76        DDV_MinMaxUInt(pDX, m_nDelay, 0, 60000);
77        DDX_Text(pDX, IDC_SUN, m_fSun);
78        DDV_MinMaxFloat(pDX, m_fSun, 0.001f, 2.f);
79        DDX_Text(pDX, IDC_BATTSIZE, m_fSize);
```

```
80        DDV_MinMaxFloat(pDX, m_fSize, 0.001f, 1.e+006f);
81        DDX_Text(pDX, IDC_HUM, m_fHum);
82        DDV_MinMaxDouble(pDX, m_fHum, 0., 100.);
83        DDX_Text(pDX, IDC_T_DRY, m_fT_dry);
84        DDV_MinMaxDouble(pDX, m_fT_dry, -55., 150.);
85        DDX_Text(pDX, IDC_T_WET, m_fT_wet);
86        DDV_MinMaxDouble(pDX, m_fT_wet, -55., 150.);
87        DDX_Text(pDX, IDC_CH0_ID, m_szCh0_id);
88        DDV_MaxChars(pDX, m_szCh0_id, 20);
89        DDX_Text(pDX, IDC_CH1_ID, m_szCh1_id);
90        DDV_MaxChars(pDX, m_szCh1_id, 20);
91        DDX_CBIndex(pDX, IDC_AMP_RANGE, m_nAmp_range);
92
93        //}}AFX_DATA_MAP
94    }
95
96
97    BEGIN_MESSAGE_MAP(CPara, CDialog)
98        //{{AFX_MSG_MAP(CPara)
99            // NOTE: the ClassWizard will add message map macros here
100       //}}AFX_MSG_MAP
101   END_MESSAGE_MAP()
102
103   /////////////////////////////////////////////////////////////////////////////
104   // CPara message handlers
```

5、文件 Rotate.h

```
1   #if !defined(AFX_ROTATE_H__5B559321_FB1B_11D3_AE66_0080C875FA7E__INCLUDED_)
2   #define AFX_ROTATE_H__5B559321_FB1B_11D3_AE66_0080C875FA7E__INCLUDED_
3
4   #if _MSC_VER > 1000
5   #pragma once
6   #endif // _MSC_VER > 1000
7   // Rotate.h : header file
8   //
9
10  # include "conio.h"
11
12  # define uchar unsigned char
13  # define uint  unsigned int
14  # define ulong unsigned long int
15
16  # define RESULT_L 2
17    /* RESULT_L is length for sampling result. */
18
19  /* Displacement for result stored in resultbuffer. */
20  # define CNT_DIS  0  /* counts/cycle       */
21
22  # define CNTBUFSIZE 2
23
24  const double ANGLE_UNIT = 1.5/40; //angle for one rotating unit
25
26  const double ANGLE_MAX = 160.0;   //max angle for rotating
```

```
27
28  void  comsend(uchar charsend);
29  uchar comrcv(void);
30  void  initcom(void);
31
32
33  /////////////////////////////////////////////////////////////////////////////
34  // CRotate dialog
35
36  class CRotate : public CDialog
37  {
38  // Construction
39  public:
40      CRotate(CWnd* pParent = NULL);   // standard constructor
41
42  // Dialog Data
43      //{{AFX_DATA(CRotate)
44      enum { IDD = IDD_DIALOG2 };
45      int       m_nDir;
46      double    m_fAngle;
47      //}}AFX_DATA
48
49
50  // Overrides
51      // ClassWizard generated virtual function overrides
52      //{{AFX_VIRTUAL(CRotate)
53      protected:
54      virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
55      //}}AFX_VIRTUAL
56
57  // Implementation
58  protected:
59
60      // Generated message map functions
61      //{{AFX_MSG(CRotate)
62      afx_msg void OnRotate();
63      //}}AFX_MSG
64      DECLARE_MESSAGE_MAP()
65  };
66
67  //{{AFX_INSERT_LOCATION}}
68  // Microsoft Visual C++ will insert additional declarations immediately before the previous line.
69
70  #endif // !defined(AFX_ROTATE_H__5B559321_FB1B_11D3_AE66_0080C875FA7E__INCLUDED_)
```

6、文件 Rotate.cpp

```
1  // Rotate.cpp : implementation file
2  //
3
4  #include "stdafx.h"
5  #include "solartester.h"
6  #include "Rotate.h"
7
```

```
8   #ifdef _DEBUG
9   #define new DEBUG_NEW
10  #undef THIS_FILE
11  static char THIS_FILE[] = __FILE__;
12  #endif
13
14  /////////////////////////////////////////////////////////////////////////////
15  // CRotate dialog
16
17
18  CRotate::CRotate(CWnd* pParent /*=NULL*/)
19      : CDialog(CRotate::IDD, pParent)
20  {
21      //{{AFX_DATA_INIT(CRotate)
22      m_nDir = -1;
23      m_fAngle = 0.0;
24      //}}AFX_DATA_INIT
25  }
26
27
28  void CRotate::DoDataExchange(CDataExchange* pDX)
29  {
30      CDialog::DoDataExchange(pDX);
31      //{{AFX_DATA_MAP(CRotate)
32      DDX_Radio(pDX, IDC_DIR, m_nDir);
33      DDX_Text(pDX, IDC_ANGLE, m_fAngle);
34      DDV_MinMaxDouble(pDX, m_fAngle, 0., ANGLE_MAX);
35      //}}AFX_DATA_MAP
36  }
37
38
39  BEGIN_MESSAGE_MAP(CRotate, CDialog)
40      //{{AFX_MSG_MAP(CRotate)
41      ON_BN_CLICKED(IDROTATE, OnRotate)
42      //}}AFX_MSG_MAP
43  END_MESSAGE_MAP()
44
45  /////////////////////////////////////////////////////////////////////////////
46  // CRotate message handlers
47
48
49
50  void CRotate::OnRotate()
51  {
52      UpdateData(TRUE);
53
54      if(m_fAngle>=0 && m_fAngle<=ANGLE_MAX)
55      {
56        Beep(1000,1000);
57
58          uchar status,dataread;
59
60          initcom();
61
62          do
```

```
63      {
64          status=_inp(0x3fd);
65              status &= 0x01;
66              if( status == 0x01)
67              {
68                  dataread=_inp(0x3f8);
69              }
70          } while( status == 0x01 );
71          Sleep(1);
72
73          uchar mcucntbuf[2*CNTBUFSIZE];
74          uint  count0,count1;
75
76          CWnd::SetDlgItemText(IDC_STATIC1,"正在与微控制器通讯,请稍候...");
77
78          comsend( int(m_fAngle/ANGLE_UNIT)%256 );  /* unit_x lowwer byte */
79          comsend( int(m_fAngle/ANGLE_UNIT)/256 );  /* unit_x higher byte */
80
81          comsend(0);       /* unit_y lowwer byte */
82          comsend(0);       /* unit_y higher byte */
83
84          comsend(0);       /* unit_z lowwer byte */
85          comsend(0);       /* unit_z higher byte */
86
87          if(m_nDir == 0)     //0 = clockwise,1 = counter clockwise
88          {
89            comsend(0);       /* dir_x */
90          }
91          else
92          {
93              comsend(1);       /* dir_x */
94          }
95
96          comsend(0);       /* dir_y */
97          comsend(0);       /* dir_z */
98
99          comsend(1);      /* speed */
100         comsend(0);      /* smp_time */
101         comsend(0);       /* mode */
102
103         CWnd::SetDlgItemText(IDC_STATIC1,"正在跟踪,请稍候...");
104
105         for(uint i=0;i<2*CNTBUFSIZE;i++)
106         {
107             mcucntbuf[i]=comrcv();
108         }
109
110         count0=mcucntbuf[0]+mcucntbuf[1]*256;
111         count1=mcucntbuf[2]+mcucntbuf[3]*256;
112
113         CWnd::SetDlgItemText(IDC_STATIC1,"已经跟踪到位");
114     }
115 }
116
117
```

```
118  /* This is the root module for Solar tracer in PC side.
119     Version 1.0.
120     Written by CHUAN-DONG XIA,Fri.,Oct. 15th,1999.
121  */
122  void initcom(void)
123  {
124    _outp(0x3fb,0x80);
125    _outp(0x3f8,0x60);
126    _outp(0x3f9,0x00);
127
128    _outp(0x3fb,0x03);
129    _outp(0x3fc,0x03);
130    _outp(0x3f9,0x00);
131  }
132
133  /* This function sends sample time to serial port.*/
134  void comsend(uchar charsend)
135  {
136    uchar pf;
137    uchar status;
138    uchar dataread;
139
140    do
141    {
142      _outp(0x3f8,0xaa);
143      do
144      {
145        status=_inp(0x3fd);
146        status &= 0x20;
147      } while( status != 0x20 );
148
149          Sleep(1);
150
151      do
152      {
153        status=_inp(0x3fd);
154        status &= 0x01;
155      } while( status != 0x01 );
156
157      dataread=_inp(0x3f8);
158    } while( (dataread^0xbb) != 0 );
159
160    do
161    {
162      pf=0;
163      _outp(0x3f8,charsend);
164      pf+=charsend;
165      do
166      {
167        status=_inp(0x3fd);
168        status &= 0x20;
169      } while( status != 0x20 );
170
171      _outp(0x3f8,pf);
172      do
```

```
173       {
174         status=_inp(0x3fd);
175         status &= 0x20;
176       } while( status != 0x20 );
177
178       do
179       {
180         status=_inp(0x3fd);
181         status &= 0x01;
182       } while( status != 0x01 );
183
184       dataread=_inp(0x3f8);
185     } while( dataread != 0 );
186   }
187
188   /* This function receives the count from serial port.*/
189   uchar comrcv(void)
190   {
191     uchar pf;
192     uchar charrcv;
193     uchar status;
194
195     do
196     {
197           Sleep(1);
198       do
199       {
200         status=_inp(0x3fd);
201         status &= 0x01;
202       } while( status != 0x01 );
203
204       charrcv=_inp(0x3f8);
205     } while( (charrcv^0xaa) !=0 );
206
207     _outp(0x3f8,0xbb);
208     do
209     {
210       status=_inp(0x3fd);
211       status &= 0x20;
212     } while( status != 0x20 );
213
214     while(1)
215     {
216       pf=0;
217
218       do
219       {
220         status=_inp(0x3fd);
221         status &= 0x01;
222       } while( status != 0x01 );
223       charrcv=_inp(0x3f8);
224       pf+=charrcv;
225
226       do
227       {
```

```
228         status=_inp(0x3fd);
229         status &= 0x01;
230       } while( status != 0x01 );
231       charrcv=_inp(0x3f8);
232       if((charrcv^pf)==0)
233       {
234         _outp(0x3f8,0x00);
235         do
236         {
237       status=_inp(0x3fd);
238       status &= 0x20;
239         } while( status != 0x20 );
240         break;
241       }
242       else
243       {
244         _outp(0x3f8,0xff);
245         do
246         {
247       status=_inp(0x3fd);
248       status &= 0x20;
249         } while( status != 0x20 );
250       }
251     }
252
253     return(charrcv);
254   }
```

7、文件 solartester.h

```
1   // solartester.h : main header file for the SOLARTESTER application
2   //
3
4   #if !defined(AFX_SOLARTESTER_H__49A03BC8_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_)
5   #define AFX_SOLARTESTER_H__49A03BC8_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_
6
7   #if _MSC_VER > 1000
8   #pragma once
9   #endif // _MSC_VER > 1000
10
11  #ifndef __AFXWIN_H__
12      #error include 'stdafx.h' before including this file for PCH
13  #endif
14
15  #include "resource.h"       // main symbols
16
17  /////////////////////////////////////////////////////////////////////////////
18  // CSolartesterApp:
19  // See solartester.cpp for the implementation of this class
20  //
21
22  class CSolartesterApp : public CWinApp
23  {
24  public:
```

```
25      CSolartesterApp();
26
27  // Overrides
28      // ClassWizard generated virtual function overrides
29      //{{AFX_VIRTUAL(CSolartesterApp)
30      public:
31      virtual BOOL InitInstance();
32      //}}AFX_VIRTUAL
33
34  // Implementation
35      COleTemplateServer m_server;
36          // Server object for document creation
37      //{{AFX_MSG(CSolartesterApp)
38      afx_msg void OnAppAbout();
39          // NOTE - the ClassWizard will add and remove member functions here.
40          //    DO NOT EDIT what you see in these blocks of generated code !
41      //}}AFX_MSG
42      DECLARE_MESSAGE_MAP()
43  };
44
45
46  /////////////////////////////////////////////////////////////////////////////
47
48  //{{AFX_INSERT_LOCATION}}
49  // Microsoft Visual C++ will insert additional declarations immediately before the previous line.
50
51  #endif // !defined(AFX_SOLARTESTER_H__49A03BC8_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_)
```

8、文件 solartester.cpp

```
1   // solartester.cpp : Defines the class behaviors for the application.
2
3
4   #include "stdafx.h"
5   #include "solartester.h"
6
7   #include "MainFrm.h"
8   #include "solartesterDoc.h"
9   #include "solartesterView.h"
10
11  #ifdef _DEBUG
12  #define new DEBUG_NEW
13  #undef THIS_FILE
14  static char THIS_FILE[] = __FILE__;
15  #endif
16
17  /////////////////////////////////////////////////////////////////////////////
18  // CSolartesterApp
19
20  BEGIN_MESSAGE_MAP(CSolartesterApp, CWinApp)
21      //{{AFX_MSG_MAP(CSolartesterApp)
22      ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
23          // NOTE - the ClassWizard will add and remove mapping macros here.
24          //    DO NOT EDIT what you see in these blocks of generated code!
```

```
25      //}}AFX_MSG_MAP
26      // Standard file based document commands
27      ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
28      ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
29      // Standard print setup command
30      ON_COMMAND(ID_FILE_PRINT_SETUP, CWinApp::OnFilePrintSetup)
31  END_MESSAGE_MAP()
32
33  /////////////////////////////////////////////////////////////////////////////
34  // CSolartesterApp construction
35
36  CSolartesterApp::CSolartesterApp()
37  {
38      // TODO: add construction code here,
39      // Place all significant initialization in InitInstance
40  }
41
42  /////////////////////////////////////////////////////////////////////////////
43  // The one and only CSolartesterApp object
44
45  CSolartesterApp theApp;
46
47  // This identifier was generated to be statistically unique for your app.
48  // You may change it if you prefer to choose a specific identifier.
49
50  // {49A03BC3-3FD8-11D3-B89D-0080C875FA7E}
51  static const CLSID clsid =
52  { 0x49a03bc3, 0x3fd8, 0x11d3, { 0xb8, 0x9d, 0x0, 0x80, 0xc8, 0x75, 0xfa, 0x7e } };
53
54  /////////////////////////////////////////////////////////////////////////////
55  // CSolartesterApp initialization
56
57  BOOL CSolartesterApp::InitInstance()
58  {
59      // Initialize OLE libraries
60      if (!AfxOleInit())
61      {
62          AfxMessageBox(IDP_OLE_INIT_FAILED);
63          return FALSE;
64      }
65
66      AfxEnableControlContainer();
67
68      // Standard initialization
69      // If you are not using these features and wish to reduce the size
70      //  of your final executable, you should remove from the following
71      //  the specific initialization routines you do not need.
72
73  #ifdef _AFXDLL
74      Enable3dControls();            // Call this when using MFC in a shared DLL
75  #else
76      Enable3dControlsStatic();   // Call this when linking to MFC statically
77  #endif
78
79      // Change the registry key under which our settings are stored.
```

```
80      // TODO: You should modify this string to be something appropriate
81      // such as the name of your company or organization.
82      SetRegistryKey(_T("SOLAR CELL RESEARCH TEAM"));
83
84      LoadStdProfileSettings(10);  // Load standard INI file options (including MRU)
85
86      // Register the application's document templates.  Document templates
87      //  serve as the connection between documents, frame windows and views.
88
89      CSingleDocTemplate* pDocTemplate;
90      pDocTemplate = new CSingleDocTemplate(
91          IDR_MAINFRAME,
92          RUNTIME_CLASS(CSolartesterDoc),
93          RUNTIME_CLASS(CMainFrame),       // main SDI frame window
94          RUNTIME_CLASS(CSolartesterView));
95      AddDocTemplate(pDocTemplate);
96
97      // Connect the COleTemplateServer to the document template.
98      //  The COleTemplateServer creates new documents on behalf
99      //  of requesting OLE containers by using information
100     //  specified in the document template.
101     m_server.ConnectTemplate(clsid, pDocTemplate, TRUE);
102         // Note: SDI applications register server objects only if /Embedding
103         //   or /Automation is present on the command line.
104
105     // Parse command line for standard shell commands, DDE, file open
106     CCommandLineInfo cmdInfo;
107     ParseCommandLine(cmdInfo);
108
109     // Check to see if launched as OLE server
110     if (cmdInfo.m_bRunEmbedded || cmdInfo.m_bRunAutomated)
111     {
112         // Register all OLE server (factories) as running.  This enables the
113         //  OLE libraries to create objects from other applications.
114         COleTemplateServer::RegisterAll();
115
116         // Application was run with /Embedding or /Automation.  Don't show the
117         //  main window in this case.
118         return TRUE;
119     }
120
121     // When a server application is launched stand-alone, it is a good idea
122     //  to update the system registry in case it has been damaged.
123     m_server.UpdateRegistry(OAT_DISPATCH_OBJECT);
124     COleObjectFactory::UpdateRegistryAll();
125
126     // Dispatch commands specified on the command line
127     if (!ProcessShellCommand(cmdInfo))
128         return FALSE;
129
130     // The one and only window has been initialized, so show and update it.
131     m_pMainWnd->ShowWindow(SW_SHOW);
132     m_pMainWnd->UpdateWindow();
133
134     return TRUE;
```

```
135  }
136
137
138  /////////////////////////////////////////////////////////////////////////////
139  // CAboutDlg dialog used for App About
140
141  class CAboutDlg : public CDialog
142  {
143  public:
144      CAboutDlg();
145
146  // Dialog Data
147      //{{AFX_DATA(CAboutDlg)
148      enum { IDD = IDD_ABOUTBOX };
149      //}}AFX_DATA
150
151      // ClassWizard generated virtual function overrides
152      //{{AFX_VIRTUAL(CAboutDlg)
153      protected:
154      virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
155      //}}AFX_VIRTUAL
156
157  // Implementation
158  protected:
159      //{{AFX_MSG(CAboutDlg)
160          // No message handlers
161      //}}AFX_MSG
162      DECLARE_MESSAGE_MAP()
163  };
164
165  CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
166  {
167      //{{AFX_DATA_INIT(CAboutDlg)
168      //}}AFX_DATA_INIT
169  }
170
171  void CAboutDlg::DoDataExchange(CDataExchange* pDX)
172  {
173      CDialog::DoDataExchange(pDX);
174      //{{AFX_DATA_MAP(CAboutDlg)
175      //}}AFX_DATA_MAP
176  }
177
178  BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
179      //{{AFX_MSG_MAP(CAboutDlg)
180          // No message handlers
181      //}}AFX_MSG_MAP
182  END_MESSAGE_MAP()
183
184  // App command to run the dialog
185  void CSolartesterApp::OnAppAbout()
186  {
187      CAboutDlg aboutDlg;
188      aboutDlg.DoModal();
189  }
```

```
190
191 ////////////////////////////////////////////////////////////////////////////
192 // CSolartesterApp message handlers
```

9、文件 solartesterDoc.h

```
1  // solartesterDoc.h : interface of the CSolartesterDoc class
2  //
3  ////////////////////////////////////////////////////////////////////////////
4
5  #if !defined(AFX_SOLARTESTERDOC_H__49A03BCF_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_)
6  #define AFX_SOLARTESTERDOC_H__49A03BCF_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_
7
8  #if _MSC_VER > 1000
9  #pragma once
10 #endif // _MSC_VER > 1000
11
12
13 class CSolartesterDoc : public CDocument
14 {
15 protected: // create from serialization only
16     CSolartesterDoc();
17     DECLARE_DYNCREATE(CSolartesterDoc)
18
19 // Attributes
20 public:
21
22 // Operations
23 public:
24
25 // Overrides
26     // ClassWizard generated virtual function overrides
27     //{{AFX_VIRTUAL(CSolartesterDoc)
28     public:
29     virtual BOOL OnNewDocument();
30     virtual void Serialize(CArchive& ar);
31     //}}AFX_VIRTUAL
32
33 // Implementation
34 public:
35     virtual ~CSolartesterDoc();
36 #ifdef _DEBUG
37     virtual void AssertValid() const;
38     virtual void Dump(CDumpContext& dc) const;
39 #endif
40
41 protected:
42
43 // Generated message map functions
44 protected:
45     //{{AFX_MSG(CSolartesterDoc)
46         // NOTE - the ClassWizard will add and remove member functions here.
47         //    DO NOT EDIT what you see in these blocks of generated code !
48     //}}AFX_MSG
```

```
49      DECLARE_MESSAGE_MAP()
50
51      // Generated OLE dispatch map functions
52      //{{AFX_DISPATCH(CSolartesterDoc)
53          // NOTE - the ClassWizard will add and remove member functions here.
54          //    DO NOT EDIT what you see in these blocks of generated code !
55      //}}AFX_DISPATCH
56      DECLARE_DISPATCH_MAP()
57      DECLARE_INTERFACE_MAP()
58  };
59
60  /////////////////////////////////////////////////////////////////////////////
61
62  //{{AFX_INSERT_LOCATION}}
63  // Microsoft Visual C++ will insert additional declarations immediately before the previous line.
64
65  #endif // !defined(AFX_SOLARTESTERDOC_H__49A03BCF_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_)
```

10、文件 solartesterDoc.cpp

```
1   // solartesterDoc.cpp : implementation of the CSolartesterDoc class
2   //
3
4   #include "stdafx.h"
5   #include "solartester.h"
6
7   #include "solartesterDoc.h"
8
9   #ifdef _DEBUG
10  #define new DEBUG_NEW
11  #undef THIS_FILE
12  static char THIS_FILE[] = __FILE__;
13  #endif
14
15  /////////////////////////////////////////////////////////////////////////////
16  // CSolartesterDoc
17
18  IMPLEMENT_DYNCREATE(CSolartesterDoc, CDocument)
19
20  BEGIN_MESSAGE_MAP(CSolartesterDoc, CDocument)
21      //{{AFX_MSG_MAP(CSolartesterDoc)
22          // NOTE - the ClassWizard will add and remove mapping macros here.
23          //    DO NOT EDIT what you see in these blocks of generated code!
24      //}}AFX_MSG_MAP
25  END_MESSAGE_MAP()
26
27  BEGIN_DISPATCH_MAP(CSolartesterDoc, CDocument)
28      //{{AFX_DISPATCH_MAP(CSolartesterDoc)
29          // NOTE - the ClassWizard will add and remove mapping macros here.
30          //      DO NOT EDIT what you see in these blocks of generated code!
31      //}}AFX_DISPATCH_MAP
32  END_DISPATCH_MAP()
33
34  // Note: we add support for IID_ISolartester to support typesafe binding
```

```
35  //  from VBA.  This IID must match the GUID that is attached to the
36  //  dispinterface in the .ODL file.
37
38  // {49A03BC5-3FD8-11D3-B89D-0080C875FA7E}
39  static const IID IID_ISolartester =
40  { 0x49a03bc5, 0x3fd8, 0x11d3, { 0xb8, 0x9d, 0x0, 0x80, 0xc8, 0x75, 0xfa, 0x7e } };
41
42  BEGIN_INTERFACE_MAP(CSolartesterDoc, CDocument)
43      INTERFACE_PART(CSolartesterDoc, IID_ISolartester, Dispatch)
44  END_INTERFACE_MAP()
45
46  /////////////////////////////////////////////////////////////////////////////
47  // CSolartesterDoc construction/destruction
48
49  CSolartesterDoc::CSolartesterDoc()
50  {
51      // TODO: add one-time construction code here
52
53      EnableAutomation();
54
55      AfxOleLockApp();
56  }
57
58  CSolartesterDoc::~CSolartesterDoc()
59  {
60      AfxOleUnlockApp();
61  }
62
63  BOOL CSolartesterDoc::OnNewDocument()
64  {
65      if (!CDocument::OnNewDocument())
66          return FALSE;
67
68      // TODO: add reinitialization code here
69      // (SDI documents will reuse this document)
70
71      return TRUE;
72  }
73
74
75
76  /////////////////////////////////////////////////////////////////////////////
77  // CSolartesterDoc serialization
78
79  void CSolartesterDoc::Serialize(CArchive& ar)
80  {
81      if (ar.IsStoring())
82      {
83          // TODO: add storing code here
84      }
85      else
86      {
87          // TODO: add loading code here
88      }
89  }
```

```
90
91  /////////////////////////////////////////////////////////////////////////////
92  // CSolartesterDoc diagnostics
93
94  #ifdef _DEBUG
95  void CSolartesterDoc::AssertValid() const
96  {
97      CDocument::AssertValid();
98  }
99
100 void CSolartesterDoc::Dump(CDumpContext& dc) const
101 {
102     CDocument::Dump(dc);
103 }
104 #endif //_DEBUG
105
106 /////////////////////////////////////////////////////////////////////////////
107 // CSolartesterDoc commands
```

11、文件 solartesterView.h

```
1   // solartesterView.h : interface of the CSolartesterView class
2   //
3   /////////////////////////////////////////////////////////////////////////////
4
5   #if !defined(AFX_SOLARTESTERVIEW_H__49A03BD1_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_)
6   #define AFX_SOLARTESTERVIEW_H__49A03BD1_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_
7
8   #if _MSC_VER > 1000
9   #pragma once
10  #endif // _MSC_VER > 1000
11
12
13
14
15  # include "para.h"
16  # include "conio.h"
17  # include "stdio.h"
18  # include "process.h"
19  # include "rotate.h"
20
21  //parameters for screen drawing
22  const int X_ORG = 700;          //X_AXIS origin
23  const int X_LEN = 26000;        //X_AXIS length
24  const int X_END = X_ORG+X_LEN; //X_AXIS endding
25
26  const int Y_ORG = -700;         //Y_AXIS origin
27  const int Y_LEN = -17000;       //Y_AXIS length
28  const int Y_END = Y_ORG+Y_LEN; //Y_AXIS endding
29
30
31  const double DA_REF = 5.0;      // D/A reference voltage
32
33  const double AD_RANGE = 10.0;  // A/D input range(+/-10V)
```

```
34
35  const double AMP_RES_CH0 = 5.0; // current sensor resistance for ch0
36  const double AMP_RES_CH1 = 5.0; // current sensor resistance for ch1
37
38  const long   METER_CH_AVERAGE = 50000; // average times for radiation meter
39  const double METER_AMP_FACTOR = 500; // gain factor for radiation meter
40  const double METER_CONST = 10.377;     // constant of radiation meter (mV/SUN)
41
42  const long   TEMP_CH_AVERAGE = 50000;  // average times for temperature meter
43  const double T_DRY_RES = 11500;      // resistor value for AD590: dry
44  const double T_WET_RES = 11500;      // resistor value for AD590: wet
45
46  /* A/D channel settings for PCL812PG multifunction card                    */
47  # define A_0_CH  0              /* Define A/D channel for current:cell 0  */
48  # define V_0_CH  1              /* Define A/D channel for voltage:cell 0  */
49
50  # define A_1_CH  2              /* Define A/D channel for current:cell 1  */
51  # define V_1_CH  3              /* Define A/D channel for voltage:cell 1  */
52
53  # define METER_CH  4            /* Define A/D channel for radiation meter  */
54
55  # define REF_CH  5              /* Define A/D channel for reference cell  */
56
57  # define T_DRY_CH  6            /* Define A/D channel for AD590:dry*/
58  # define T_WET_CH  7            /* Define A/D channel for AD590:wet*/
59
60  /* I/O address for PCL812PG multifunction card                          */
61  # define BASE      0x320            /* base I/O address                */
62
63  # define CNT0      (BASE+0)         /* unit 0 of 8254 counter/timer */
64  # define CNT1      (BASE+1)         /* unit 1 of 8254 counter/timer */
65  # define CNT2      (BASE+2)         /* unit 2 of 8254 counter/timer */
66  # define CNT_CTRL  (BASE+3)         /* control byte for 8254        */
67  # define AD_LOW    (BASE+4)         /* result of A/D, low byte      */
68  # define AD_HIGH   (BASE+5)         /* result of A/D, high byte     */
69  # define DI_LOW    (BASE+6)         /* result of digital input:low  */
70  # define DI_HIGH   (BASE+7)         /* result of digital input:high */
71  # define DA_CH0    (BASE+4)         /* D/A conversion CH0, low byte  */
72  # define DA_CH1    (BASE+6)         /* D/A conversion CH1, low byte  */
73  # define CLR_IRQ   (BASE+8)         /* clear interrupt request      */
74  # define AD_GAIN   (BASE+9)         /* A/D gain control             */
75  # define AD_CH     (BASE+10)        /* A/D channel MUX              */
76  # define AD_MODE   (BASE+11)        /* A/D conversion mode control  */
77  # define AD_TRIG   (BASE+12)        /* A/D software trigger         */
78  # define DO_LOW    (BASE+13)        /* digital output:low           */
79  # define DO_HIGH   (BASE+14)        /* digital output:high          */
80
81  # define DO_LOW_BYTE  0                          /* digital output lowwer byte   */
82  # define DO_HIGH_BYTE 1                          /* digital output upper byte    */
83
84
85  class CSolartesterView : public CScrollView
86  {
87  protected: // create from serialization only
88      CSolartesterView();
```

```
89          DECLARE_DYNCREATE(CSolartesterView)
90
91      // Attributes
92      public:
93          CSolartesterDoc* GetDocument();
94
95      // Operations
96      public:
97
98      // Overrides
99          // ClassWizard generated virtual function overrides
100         //{{AFX_VIRTUAL(CSolartesterView)
101         public:
102         virtual void OnDraw(CDC* pDC);  // overridden to draw this view
103         virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
104         virtual void OnPrepareDC(CDC* pDC, CPrintInfo* pInfo = NULL);
105         protected:
106         virtual BOOL OnPreparePrinting(CPrintInfo* pInfo);
107         virtual void OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo);
108         virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo);
109         virtual void OnInitialUpdate();
110         //}}AFX_VIRTUAL
111
112     // Implementation
113     public:
114         BOOL m_bEnable;
115         virtual ~CSolartesterView();
116     #ifdef _DEBUG
117         virtual void AssertValid() const;
118         virtual void Dump(CDumpContext& dc) const;
119     #endif
120
121     protected:
122
123     // Generated message map functions
124     protected:
125         //{{AFX_MSG(CSolartesterView)
126         afx_msg void OnBegin();
127         afx_msg void OnTimer(UINT nIDEvent);
128         afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
129         afx_msg void OnDestroy();
130         afx_msg void OnUpdateBegin(CCmdUI* pCmdUI);
131         afx_msg void OnVmax();
132         afx_msg void OnEnd();
133         afx_msg void OnUpdateEnd(CCmdUI* pCmdUI);
134         afx_msg void OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags);
135         afx_msg void OnSun();
136         afx_msg void OnTemp();
137         afx_msg void OnTrace();
138         //}}AFX_MSG
139         DECLARE_MESSAGE_MAP()
140
141     protected:
142         void BlkGraph(CDC* pDC);
143
```

```
144  public:
145      int ao(int ch,UINT data);
146    int ad_set_channel(int ad_ch_no);
147      int ad_set_gain(int ad_gain);
148      int ad_set_mode(int ad_mode);
149      int ad_soft_trig(void);
150      int ad_acquire(int *ad_data);
151      int Do(int port_number,unsigned char data);
152
153  protected:
154      int ad_a;
155      int ad_v;
156
157      int  x1,y1,x2,y2;
158    int  i;
159    int  j;
160      int  jj;
161    int  ad;
162
163    int    v_count_ch0[10001];
164    int    a_count_ch0[10001];
165
166      int    x_ch0[1001];
167    int    y_ch0[1001];
168
169      double volt_ch0[1001];
170    double amp_ch0[1001];
171    double pwr_ch0[1001];
172
173    long   v_sum_ch0;
174    long   a_sum_ch0;
175
176      double FF_ch0;
177    double Eff_ch0;
178    double Jsc_ch0;
179    double Isc_ch0;
180    double Voc_ch0;
181    double Imp_ch0;
182    double Vmp_ch0;
183    double Pmax_ch0;
184
185      int    v_count_ch1[10001];
186    int    a_count_ch1[10001];
187
188      int    x_ch1[1001];
189    int    y_ch1[1001];
190
191      double volt_ch1[1001];
192    double amp_ch1[1001];
193    double pwr_ch1[1001];
194
195      long   v_sum_ch1;
196    long   a_sum_ch1;
197
198      double FF_ch1;
```

```
199     double Eff_ch1;
200     double Jsc_ch1;
201     double Isc_ch1;
202     double Voc_ch1;
203     double Imp_ch1;
204     double Vmp_ch1;
205     double Pmax_ch1;
206
207        CString Ch0_id;
208        CString Ch1_id;
209        double Vmax;
210     double Size;
211     int    Sample;
212     int    Average;
213     int    Delay;
214     double Sun;
215
216        double T_dry;
217        double T_wet;
218        double Hum;
219
220        int    Amp_range;
221
222     double amp_factor;
223        double volt_factor;
224     double da_factor;
225
226     double a_factor;
227     double v_factor;
228
229     int amp_card_gain;
230        int amp_board_gain;
231        int volt_card_gain;
232        int da_gain;
233
234        int step;
235     double step_len;
236     int count;
237
238     int  a_count_min;
239
240     char ch;
241
242        double meter;
243
244        double t_dry;
245        double t_wet;
246        double hum;
247
248        time_t t_begin;
249        time_t t_end;
250
251        double a_range;
252        double v_range;
253  };
```

```
254
255  #ifndef _DEBUG  // debug version in solartesterView.cpp
256  inline CSolartesterDoc* CSolartesterView::GetDocument()
257     { return (CSolartesterDoc*)m_pDocument; }
258  #endif
259
260  /////////////////////////////////////////////////////////////////////////////
261
262  //{{AFX_INSERT_LOCATION}}
263  // Microsoft Visual C++ will insert additional declarations immediately before the previous line.
264
265  #endif // !defined(AFX_SOLARTESTERVIEW_H__49A03BD1_3FD8_11D3_B89D_0080C875FA7E__INCLUDED_)
```

12、文件solartesterView.cpp

```
1   // solartesterView.cpp : implementation of the CSolartesterView class
2   //
3
4   #include "stdafx.h"
5   #include "afxext.h"
6
7   #include "winuser.h"
8
9   #include "solartester.h"
10
11  #include "solartesterDoc.h"
12  #include "solartesterView.h"
13
14  #ifdef _DEBUG
15  #define new DEBUG_NEW
16  #undef THIS_FILE
17  static char THIS_FILE[] = __FILE__;
18  #endif
19
20  /////////////////////////////////////////////////////////////////////////////
21  // CSolartesterView
22
23  IMPLEMENT_DYNCREATE(CSolartesterView, CView)
24
25  BEGIN_MESSAGE_MAP(CSolartesterView, CScrollView)
26      //{{AFX_MSG_MAP(CSolartesterView)
27      ON_COMMAND(ID_BEGIN, OnBegin)
28      ON_WM_TIMER()
29      ON_WM_CREATE()
30      ON_WM_DESTROY()
31      ON_UPDATE_COMMAND_UI(ID_BEGIN, OnUpdateBegin)
32      ON_COMMAND(ID_VMAX, OnVmax)
33      ON_COMMAND(ID_END, OnEnd)
34      ON_UPDATE_COMMAND_UI(ID_END, OnUpdateEnd)
35      ON_WM_KEYDOWN()
36      ON_COMMAND(ID_SUN, OnSun)
37      ON_COMMAND(ID_TEMP, OnTemp)
38      ON_COMMAND(ID_TRACE, OnTrace)
39      //}}AFX_MSG_MAP
```

```
40      // Standard printing commands
41      ON_COMMAND(ID_FILE_PRINT, CView::OnFilePrint)
42      ON_COMMAND(ID_FILE_PRINT_DIRECT, CScrollView::OnFilePrint)
43      ON_COMMAND(ID_FILE_PRINT_PREVIEW, CView::OnFilePrintPreview)
44  END_MESSAGE_MAP()
45
46  /////////////////////////////////////////////////////////////////////////////
47  // CSolartesterView construction/destruction
48
49  CSolartesterView::CSolartesterView()
50  {
51      char str2[] =
52          "请确认太阳电池测试仪已经开机\n通讯电缆接至PC机串行口COM2\n\n继续吗?";
53      int msg = MessageBox( str2,"确认开机",MB_YESNO );
54      if(msg == IDNO )
55      {
56          exit(1);
57      }
58
59
60      m_bEnable=TRUE;
61
62      Ch0_id.Empty();
63      Ch1_id.Empty();
64      char str0[25];
65      char str1[25];
66
67      FILE* fp3;
68      fp3=fopen("PARAS.DAP","r");
69
70      if(fp3 != NULL)
71      {
72          fscanf(fp3,"m_szCh0_id = %s\n\r",str0);
73          fscanf(fp3,"m_szCh1_id = %s\n\r",str1);
74          fscanf(fp3,"m_nAverage = %d\n",&Average);
75          fscanf(fp3,"m_fVmax = %lf\n",&Vmax);
76          fscanf(fp3,"m_nSample = %d\n",&Sample);
77          fscanf(fp3,"m_nDelay = %d\n",&Delay);
78          fscanf(fp3,"m_fSun = %lf\n",&Sun);
79          fscanf(fp3,"m_fSize = %lf\n",&Size);
80          fscanf(fp3,"m_fT_dry = %lf\n",&T_dry);
81          fscanf(fp3,"m_fT_wet = %lf\n",&T_wet);
82          fscanf(fp3,"m_fHum = %lf\n",&Hum);
83          fscanf(fp3,"m_nAmp_range = %d\n",&Amp_range);
84          fclose(fp3);
85
86          Ch0_id = str0;
87          Ch1_id = str1;
88      }
89      else
90      {
91          Ch0_id = "ch0";
92          Ch1_id = "ch1";
93          Average = 100;
94          Vmax = 10.0;
```

```
95            Sample =100 ;
96            Delay = 0;
97            Sun = 0.5;
98            Size = 7.0;
99            T_dry = 20;
100           T_wet = 20;
101           Hum = 0.0;
102           Amp_range = 0;
103       }
104
105       v_sum_ch0=0;
106   a_sum_ch0=0;
107
108       FF_ch0=0;
109   Eff_ch0=0;
110   Jsc_ch0=0;
111       Isc_ch0=0;
112   Voc_ch0=0;
113   Imp_ch0=0;
114   Vmp_ch0=0;
115   Pmax_ch0=0;
116
117       v_sum_ch1=0;
118   a_sum_ch1=0;
119
120       FF_ch1=0;
121   Eff_ch1=0;
122       Jsc_ch1 = 0;
123   Isc_ch1=0;
124   Voc_ch1=0;
125   Imp_ch1=0;
126   Vmp_ch1=0;
127   Pmax_ch1=0;
128
129       amp_card_gain=1;
130       amp_board_gain=1;
131       if( Vmax <= AD_RANGE/4 )
132       {
133           volt_card_gain=4;
134       }
135       else
136       {
137           if( Vmax <= AD_RANGE/2 )
138           {
139               volt_card_gain=2;
140           }
141           else
142           {
143               volt_card_gain=1;
144           }
145       }
146
147       ad_set_gain(amp_card_gain);
148   ad_set_mode(1);
149
```

```
150        da_gain=1;
151        da_factor=DA_REF/4095*da_gain;
152
153     amp_factor=AD_RANGE/(2048*amp_card_gain*amp_board_gain);
154        volt_factor=AD_RANGE/(2048*volt_card_gain);
155
156     a_factor=X_LEN/4095.0;
157     v_factor=Y_LEN/2048.0;
158
159     step=(int)(Vmax/da_factor);
160     step_len=Vmax/da_factor/(Sample-1);
161
162     if(Sample>step)
163     {
164       count=step;
165     }
166     else
167     {
168       count=Sample;
169     }
170
171     if(step_len<1.0)
172     {
173       step_len=1;
174     }
175
176        switch(Amp_range)
177        {
178             case(0)://2A
179             {
180                  amp_card_gain=1;
181                  amp_board_gain=1;
182                  break;
183             }
184             case(1)://1A
185             {
186                  amp_card_gain=2;
187                  amp_board_gain=1;
188                  break;
189             }
190             case(2)://500mA
191             {
192                  amp_card_gain=4;
193                  amp_board_gain=1;
194                  break;
195             }
196             case(3)://200mA
197             {
198                  amp_card_gain=1;
199                  amp_board_gain=10;
200                  break;
201             }
202             case(4)://100mA
203             {
204                  amp_card_gain=2;
```

```
205                 amp_board_gain=10;
206                 break;
207             }
208             case(5)://50mA
209             {
210                 amp_card_gain=4;
211                 amp_board_gain=10;
212                 break;
213             }
214             case(6)://20mA
215             {
216                 amp_card_gain=1;
217                 amp_board_gain=100;
218                 break;
219             }
220             case(7)://10mA
221             {
222                 amp_card_gain=2;
223                 amp_board_gain=100;
224                 break;
225             }
226             case(8)://5mA
227             {
228                 amp_card_gain=4;
229                 amp_board_gain=100;
230                 break;
231             }
232             case(9)://AUTO
233             {
234                 amp_card_gain=1;
235                 amp_board_gain=1;
236                 break;
237             }
238             default:
239             {
240                 amp_card_gain=1;
241                 amp_board_gain=1;
242                 break;
243             }
244         }
245     }
246
247     CSolartesterView::~CSolartesterView()
248     {
249         ao(0,0);
250         ao(1,0);
251     }
252
253     BOOL CSolartesterView::PreCreateWindow(CREATESTRUCT& cs)
254     {
255         // TODO: Modify the Window class or styles here by modifying
256         //  the CREATESTRUCT cs
257
258         return CScrollView::PreCreateWindow(cs);
259     }
```

```
260
261
262  void CSolartesterView::BlkGraph(CDC* pDC).
263  {
264      int x1,y1,x2,y2,x_step,y_step;
265
266      a_range = AD_RANGE/AMP_RES_CH0/amp_card_gain/amp_board_gain*1000;
267      v_range = AD_RANGE/volt_card_gain;
268
269      char str1[50];
270
271      COLORREF crGreen = RGB(0,255,0);
272      COLORREF crBlue = RGB(0,0,220);
273
274      COLORREF  colorRef;
275      colorRef = pDC->SetTextColor( crBlue );
276
277      CPen newPen;
278      CPen newPen2;
279      CPen* oldPen;
280
281      newPen.CreatePen(PS_SOLID,50,crBlue);
282
283      oldPen = pDC -> SelectObject(&newPen);
284
285      pDC->MoveTo(X_ORG,Y_END);                          // draw the horizontal axis
286      pDC->LineTo(X_END,Y_END);
287
288      pDC->MoveTo(X_ORG+X_LEN/2,Y_ORG);     // draw the vertical axis
289      pDC->LineTo(X_ORG+X_LEN/2,Y_END);
290
291      newPen2.CreatePen(PS_SOLID,1,crGreen);
292      pDC -> SelectObject(&newPen2);
293
294      x1=X_ORG,x2=X_END,y_step=Y_LEN/10;        //draw the horizontal coordinate line
295      for(y1=Y_ORG;y1>Y_END;y1+=y_step)
296      {
297          pDC->MoveTo(x1,y1);
298          pDC->LineTo(x2,y1);
299      }
300
301      y1=Y_ORG,y2=Y_END,x_step=X_LEN/20;        //draw the vertical coordinate line
302      for(x1=X_ORG;x1<=X_END;x1+=x_step)
303      {
304          pDC->MoveTo(x1,y1);
305          pDC->LineTo(x1,y2);
306      }
307
308      x1=X_ORG-X_ORG*0.99,x2=X_END+100,y1=Y_ORG*0.75,y_step=Y_LEN/10;
309      for(int n=0;n<=10;n++)
310      {
311          if(v_range<=2.5)
312          {
313       sprintf(str1,"%4.2lf",v_range*(10-n)/10.0);
314          }
```

```
315             else
316             {
317                 sprintf(str1,"%4.1lf",v_range*(10-n)/10.0);
318             }
319
320           pDC->TextOut(x1,y1+n*y_step,str1);
321             pDC->TextOut(x2,y1+n*y_step,str1);
322         }
323
324         x1=X_ORG-350;y1=Y_ORG*0.35;y2=Y_END-50;x_step=X_LEN/20;
325         for(n=0;n<=20;n++)
326         {
327             if(a_range<10)
328             {
329          sprintf(str1,"%4.1lf",a_range*(n-10)/10.0);
330             }
331             else
332             {
333                 sprintf(str1,"%4.0lf",a_range*(n-10)/10.0);
334             }
335
336           pDC->TextOut(x1+n*x_step,y1,str1);
337             pDC->TextOut(x1+n*x_step,y2,str1);
338         }
339
340         pDC -> SelectObject(oldPen);
341         pDC->SetTextColor(colorRef);
342     }
343
344     /////////////////////////////////////////////////////////////////////////////
345     // CSolartesterView printing
346
347     BOOL CSolartesterView::OnPreparePrinting(CPrintInfo* pInfo)
348     {
349         return DoPreparePrinting(pInfo);
350     }
351
352     void CSolartesterView::OnBeginPrinting(CDC* pDC, CPrintInfo* /*pInfo*/)
353     {
354
355     }
356
357     void CSolartesterView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
358     {
359         // TODO: add cleanup after printing
360     }
361
362     /////////////////////////////////////////////////////////////////////////////
363     // CSolartesterView diagnostics
364
365     #ifdef _DEBUG
366     void CSolartesterView::AssertValid() const
367     {
368         CScrollView::AssertValid();
369     }
```

```
370
371 void CSolartesterView::Dump(CDumpContext& dc) const
372 {
373     CView::Dump(dc);
374 }
375
376 CSolartesterDoc* CSolartesterView::GetDocument() // non-debug version is inline
377 {
378     ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CSolartesterDoc)));
379     return (CSolartesterDoc*)m_pDocument;
380 }
381 #endif //_DEBUG
382
383 /////////////////////////////////////////////////////////////////////////////
384 // CSolartesterView message handlers
385
386 void CSolartesterView::OnBegin()
387 {
388   int msg = MessageBox( "准备开始测试吗?","开始测试",MB_YESNO );
389
390     if(msg == IDNO )
391     {
392         return;
393     }
394     Beep(1000,500);
395
396     m_bEnable=FALSE;
397
398     FF_ch0 = 0;
399   Eff_ch0 = 0;
400   Jsc_ch0 = 0;
401   Isc_ch0 = 0;
402   Voc_ch0 = 0;
403   Imp_ch0 = 0;
404   Vmp_ch0 = 0;
405   Pmax_ch0 = 0;
406
407     FF_ch1 = 0;
408   Eff_ch1 = 0;
409   Jsc_ch1 = 0;
410   Isc_ch1 = 0;
411   Voc_ch1 = 0;
412   Imp_ch1 = 0;
413   Vmp_ch1 = 0;
414   Pmax_ch1 = 0;
415
416     switch(Amp_range)
417     {
418         case(0)://2A
419         {
420             amp_card_gain=1;
421             amp_board_gain=1;
422             break;
423         }
424         case(1)://1A
```

```
425             {
426                 amp_card_gain=2;
427                 amp_board_gain=1;
428                 break;
429             }
430             case(2)://500mA
431             {
432                 amp_card_gain=4;
433                 amp_board_gain=1;
434                 break;
435             }
436             case(3)://200mA
437             {
438                 amp_card_gain=1;
439                 amp_board_gain=10;
440                 break;
441             }
442             case(4)://100mA
443             {
444                 amp_card_gain=2;
445                 amp_board_gain=10;
446                 break;
447             }
448             case(5)://50mA
449             {
450                 amp_card_gain=4;
451                 amp_board_gain=10;
452                 break;
453             }
454             case(6)://20mA
455             {
456                 amp_card_gain=1;
457                 amp_board_gain=100;
458                 break;
459             }
460             case(7)://10mA
461             {
462                 amp_card_gain=2;
463                 amp_board_gain=100;
464                 break;
465             }
466             case(8)://5mA
467             {
468                 amp_card_gain=4;
469                 amp_board_gain=100;
470                 break;
471             }
472             case(9)://AUTO
473             {
474                 amp_card_gain=1;
475                 amp_board_gain=1;
476                 break;
477             }
478             default:
479             {
```

```
480                 amp_card_gain=1;
481                 amp_board_gain=1;
482                 break;
483             }
484         }
485
486         if( Vmax <= DA_REF )
487         {
488             da_gain=1;
489         }
490         else
491         {
492             if( Vmax <= 2*DA_REF )
493             {
494                 da_gain=2;
495             }
496             else
497             {
498                 if( Vmax <= 4*DA_REF )
499                 {
500                     da_gain=4;
501                 }
502                 else
503                 {
504                     if( Vmax <= 8*DA_REF )
505                     {
506                         da_gain=8;
507                     }
508                 }
509             }
510         }
511
512         Do(DO_LOW_BYTE,0x00);   //00 00  00 00
513
514         if( (amp_board_gain==1) && (da_gain==1) )
515         {
516             Do(DO_LOW_BYTE,0x00);   //00 00  00 00
517         }
518
519         if( (amp_board_gain==1) && (da_gain==2) )
520         {
521             Do(DO_LOW_BYTE,0x11);   //00 01 00 01
522         }
523
524         if( (amp_board_gain==1) && (da_gain==4) )
525         {
526             Do(DO_LOW_BYTE,0x22);   //00 10  00 10
527         }
528
529         if( (amp_board_gain==1) && (da_gain==8) )
530         {
531             Do(DO_LOW_BYTE,0x33);   //00 11  00 11
532         }
533
534         if( (amp_board_gain==10) && (da_gain==1) )
```

```
535         {
536             Do(DO_LOW_BYTE,0x44);   //01 00  01 00
537         }
538
539         if( (amp_board_gain==10) && (da_gain==2) )
540         {
541             Do(DO_LOW_BYTE,0x55);   //01 01 01 01
542         }
543
544         if( (amp_board_gain==10) && (da_gain==4) )
545         {
546             Do(DO_LOW_BYTE,0x66);   //01 10  01 10
547         }
548
549         if( (amp_board_gain==10) && (da_gain==8) )
550         {
551             Do(DO_LOW_BYTE,0x77);   //01 11  01 11
552         }
553
554         if( (amp_board_gain==100) && (da_gain==1) )
555         {
556             Do(DO_LOW_BYTE,0x88);   //10 00  10 00
557         }
558
559         if( (amp_board_gain==100) && (da_gain==2) )
560         {
561             Do(DO_LOW_BYTE,0x99);   //10 01 10 01
562         }
563
564         if( (amp_board_gain==100) && (da_gain==4) )
565         {
566             Do(DO_LOW_BYTE,0xaa);   //10 10  10 10
567         }
568
569         if( (amp_board_gain==100) && (da_gain==8) )
570         {
571             Do(DO_LOW_BYTE,0xbb);   //10 11  10 11
572         }
573
574         if( (amp_board_gain==1000) && (da_gain==1) )
575         {
576             Do(DO_LOW_BYTE,0xcc);   //11 00  11 00
577         }
578
579         if( (amp_board_gain==1000) && (da_gain==2) )
580         {
581             Do(DO_LOW_BYTE,0xdd);   //11 01 11 01
582         }
583
584         if( (amp_board_gain==1000) && (da_gain==4) )
585         {
586             Do(DO_LOW_BYTE,0xee);   //11 10  11 10
587         }
588
589         if( (amp_board_gain==1000) && (da_gain==8) )
```

```
590         {
591             Do(DO_LOW_BYTE,0xff);   //11 11  11 11
592         }
593
594         if( Vmax <= AD_RANGE/4 )
595         {
596             volt_card_gain=4;
597         }
598         else
599         {
600             if( Vmax <= AD_RANGE/2 )
601             {
602                 volt_card_gain=2;
603             }
604             else
605             {
606                 volt_card_gain=1;
607             }
608         }
609
610         ad_set_mode(1);
611
612         da_factor=DA_REF/4095*da_gain;
613
614     amp_factor=AD_RANGE/(2048*amp_card_gain*amp_board_gain);
615         volt_factor=AD_RANGE/(2048*volt_card_gain);
616
617     a_factor=X_LEN/4095.0;
618     v_factor=Y_LEN/2048.0;
619
620     step=(int)(Vmax/da_factor);
621     step_len=Vmax/da_factor/(Sample-1);
622
623     if(Sample>step)
624     {
625       count=step;
626     }
627     else
628     {
629       count=Sample;
630     }
631
632     if(step_len<1.0)
633     {
634       step_len=1.0;
635     }
636
637         time(&t_begin);
638
639         ao(0,0);
640         ao(1,0);
641         Sleep(1);
642
643         Invalidate();
644
```