```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Text;
7  using System.Windows.Forms;
8  using System.Runtime.InteropServices;//需要用到这个命名空间
9
10 /*------------兼容ZLG的数据类型--------------------------------*/
11
12 //1.ZLGCAN系列接口卡信息的数据类型。
13 public struct VCI_BOARD_INFO//(第一个)
14 {
15     public UInt16 hw_Version;   //申明hw_Version：硬件版本号，用16进制表示。比
       如0x0100表示V1.00。
16     public UInt16 fw_Version;   //申明fw_Version：固件版本号，用16进制表示。比
       如0x0100表示V1.00。
17     public UInt16 dr_Version;   //申明dr_Version：驱动程序版本号，用16进制表
       示。比如0x0100表示V1.00。
18     public UInt16 in_Version;   //申明dr_Version：接口库版本号，用16进制表示。
       比如0x0100表示V1.00。
19     public UInt16 irq_Num;      //申明irq_Num：保留参数。
20     public byte can_Num;        //申明can_Num：表示有几路CAN通道。
21     /*
22      str_Serial_Num此板卡的序列号。
23      str_hw_Type硬件类型，比如“USBCAN V1.00”（注意：包括字符串结束符’\0’）
24      Reserved系统保留。
25      */
26     [MarshalAs(UnmanagedType.ByValArray, SizeConst = 20)] public byte[]
       str_Serial_Num;//二维数组转一维数组
27     [MarshalAs(UnmanagedType.ByValArray, SizeConst = 40)] public byte[]
       str_hw_Type;//二维数组转一维数组
28     [MarshalAs(UnmanagedType.ByValArray, SizeConst = 4)] public byte[]
       Reserved;//二维数组转一维数组
29     /*
30     作用：
31     MarshalAs属性指示如何在托管代码和非托管代码之间封送数据。
32     使用方法：
33     [MarshalAs(UnmanagedType unmanagedType, 命名参数)]
34     实际上相当于构造一个MarshalAsAttribute类的对象
35      */
36 }
37
38 ////////////////////////////////////////////////////
39 //2.定义CAN信息帧的数据类型。
40 unsafe public struct VCI_CAN_OBJ  //使用不安全代码
41 {
42     public uint ID;
43     public uint TimeStamp;        //时间标识
44     public byte TimeFlag;         //是否使用时间标识
45     public byte SendType;         //发送标志。保留，未用
46     public byte RemoteFlag;       //是否是远程帧
47     public byte ExternFlag;       //是否是扩展帧
48     public byte DataLen;          //数据长度
49     public fixed byte Data[8];    //数据
```

```
50      public fixed byte Reserved[3];//保留位
51  }
52
53  //3.定义初始化CAN的数据类型
54  public struct VCI_INIT_CONFIG
55  {
56      public UInt32 AccCode;//验收码。SJA1000的帧过滤验收码。对经过屏蔽码过滤为“有关位”进行匹配，全部匹配成功后，此帧可以被接收。否则不接收。详见VCI_InitCAN。
57      public UInt32 AccMask;//屏蔽码。SJA1000的帧过滤屏蔽码。对接收的CAN帧ID进行过滤，对应位为0的是“有关位”，对应位为1的是“无关位”。屏蔽码推荐设置为0xFFFFFFFF，即全部接收。
58      public UInt32 Reserved;//保留。
59      public byte Filter;   //0或1接收所有帧。2标准帧滤波，3是扩展帧滤波。
60      public byte Timing0;  //波特率参数，具体配置，请查看二次开发库函数说明书。
61      public byte Timing1;
62      /*
63     CAN波特率 Timing0(BTR0)  Timing1(BTR1)
64     10 Kbps         0x31          0x1C
65     20 Kbps         0x18          0x1C
66     40 Kbps         0x87          0xFF
67     50 Kbps         0x09          0x1C
68     80 Kbps         0x83          0xFF
69    100 Kbps         0x04          0x1C
70    125 Kbps         0x03          0x1C
71    200 Kbps         0x81          0xFA
72    250 Kbps         0x01          0x1C
73    400 Kbps         0x80          0xFA
74    500 Kbps         0x00          0x1C
75    666 Kbps         0x80          0xB6
76    800 Kbps         0x00          0x16
77   1000 Kbps         0x00          0x14
78   33.33 Kbps        0x09          0x6F
79   66.66 Kbps        0x04          0x6F
80   83.33 Kbps        0x03          0x6F
81   */
82      public byte Mode;     //模式，0表示正常模式，1表示只听模式,2自测模式
83  }
84
85  /*------------其他数据结构描述--------------------------------*/
86  //4.USB-CAN总线适配器板卡信息的数据类型1，该类型为VCI_FindUsbDevice函数的返回参数。
87  public struct VCI_BOARD_INFO1//(第二个)
88  {
89      public UInt16 hw_Version;//申明hw_Version：硬件版本号，用16进制表示。
90      public UInt16 fw_Version;//申明fw_Version：固件版本号，用16进制表示。
91      public UInt16 dr_Version;//申明dr_Version：驱动程序版本号，用16进制表示。
92      public UInt16 in_Version;//申明dr_Version：接口库版本号，用16进制表示。
93      public UInt16 irq_Num; //申明irq_Num：保留参数。
94      public byte can_Num; //申明can_Num：表示有几路CAN通道。
95      public byte Reserved;//Reserved系统保留。
96      [MarshalAs(UnmanagedType.ByValArray, SizeConst = 8)] public byte[] str_Serial_Num;//二维数组转一维数组
97      [MarshalAs(UnmanagedType.ByValArray, SizeConst = 16)] public byte[] str_hw_Type;//二维数组转一维数组
98      [MarshalAs(UnmanagedType.ByValArray, SizeConst = 16)] public byte[]
```

```
        str_Usb_Serial;//两个CAN卡同一台电脑使用，调用VCI_FindUsbDevice(ref
        VCI_BOARD_INFO1 pInfo)函数，函数返回值为2。结构体中，str_Usb_Serial前4字
        节为卡1的序列号，后4字节为卡2的序列号。
 99 }
100
101 /*------------数据结构描述完成--------------------------------*/
102
103 /*CHGDESIPANDPORT 结构体用于装载更改 CANET_UDP 与 CANET_TCP 的目标 IP 和
104 端口的必要信息。此结构体在 CANETE_UDP 与 CANET_TCP 中使用。
105 typedef struct _tagChgDesIPAndPort {
106 char szpwd[10];      //更改目标 IP 和端口所需要的密码
107 char szdesip[20];    //所要更改的目标 IP
108 int desport;         //所要更改的目标端口，比如为 4000。
109 BYTE blisten;        //所要更改的工作模式，0 表示正常模式，1 表示只听模式。
110 } CHGDESIPANDPORT;*/
111 public struct CHGDESIPANDPORT   //CANET 通讯结构体
112 {
113     [MarshalAs(UnmanagedType.ByValArray, SizeConst = 10)] public byte[]
        szpwd;//转一维数组（10位），更改目标 IP 和端口所需要的密码，长度小于
        10，比如为“11223344”。
114     [MarshalAs(UnmanagedType.ByValArray, SizeConst = 20)] public byte[]
        szdesip;//转一维数组（20位），所要更改的目标 IP，比如
        为“192.168.0.111”。
115     public Int32 desport;
116
117     public void Init()
118     {
119         szpwd = new byte[10];
120         szdesip = new byte[20];
121     }
122 }
123
124 namespace CANalyst_II_CANOpen_方案    //方案空间(类似主函数)
125 {
126     public partial class Form1 : Form   //部分类Form1
127     {
128         /
            *-----------------------------------------------------------------
            -----------*/
129         const int DEV_USBCAN = 3;   //全局变量定义，这里USBCANalyst-I为3
130         const int DEV_USBCAN2 = 4;  //全局变量定义，这里USBCANalyst-II为4
131         int flag_can = 1;//定义一个标志，1的时候接收2通道，2的时候接收1通道(全
            局变量属性)
132         /// <summary>
133         ///
134         /// </summary>
135         /// <param name="DeviceType"></param>
136         /// <param name="DeviceInd"></param>
137         /// <param name="Reserved"></param>
138         /// <returns></returns>
139         /*------------兼容ZLG的函数描述--------------------------------*/
140         [DllImport("controlcan.dll")]
141         static extern UInt32 VCI_OpenDevice(UInt32 DeviceType, UInt32
            DeviceInd, UInt32 Reserved);
142         [DllImport("controlcan.dll")]
143         static extern UInt32 VCI_CloseDevice(UInt32 DeviceType, UInt32
```

```
            DeviceInd);
144         [DllImport("controlcan.dll")]
145         static extern UInt32 VCI_InitCAN(UInt32 DeviceType, UInt32 DeviceInd,
              UInt32 CANInd, ref VCI_INIT_CONFIG pInitConfig);
146
147         [DllImport("controlcan.dll")]
148         static extern UInt32 VCI_ReadBoardInfo(UInt32 DeviceType, UInt32
              DeviceInd, ref VCI_BOARD_INFO pInfo);
149
150         [DllImport("controlcan.dll")]
151         static extern UInt32 VCI_GetReceiveNum(UInt32 DeviceType, UInt32
              DeviceInd, UInt32 CANInd);
152         [DllImport("controlcan.dll")]
153         static extern UInt32 VCI_ClearBuffer(UInt32 DeviceType, UInt32
              DeviceInd, UInt32 CANInd);
154
155         [DllImport("controlcan.dll")]
156         static extern UInt32 VCI_StartCAN(UInt32 DeviceType, UInt32 DeviceInd,
               UInt32 CANInd);
157         [DllImport("controlcan.dll")]
158         static extern UInt32 VCI_ResetCAN(UInt32 DeviceType, UInt32 DeviceInd,
               UInt32 CANInd);
159
160         [DllImport("controlcan.dll")]
161         static extern UInt32 VCI_Transmit(UInt32 DeviceType, UInt32 DeviceInd,
               UInt32 CANInd, ref VCI_CAN_OBJ pSend, UInt32 Len);
162
163         [DllImport("controlcan.dll")]
164         static extern UInt32 VCI_Receive(UInt32 DeviceType, UInt32 DeviceInd,
              UInt32 CANInd, ref VCI_CAN_OBJ pReceive, UInt32 Len, Int32
              WaitTime);
165
166         /*------------其他函数描述--------------------------------*/
167
168         [DllImport("controlcan.dll")]
169         static extern UInt32 VCI_ConnectDevice(UInt32 DevType, UInt32
              DevIndex);
170         [DllImport("controlcan.dll")]
171         static extern UInt32 VCI_UsbDeviceReset(UInt32 DevType, UInt32
              DevIndex, UInt32 Reserved);
172         [DllImport("controlcan.dll")]
173         static extern UInt32 VCI_FindUsbDevice(ref VCI_BOARD_INFO1 pInfo);
174         /*------------函数描述结束--------------------------------*/
175
176         static UInt32 m_devtype = 4;//USBCAN2，默认初始自动选择CANalyst-II(在
              后面下拉菜单的时候会用到)
177
178         UInt32 m_bOpen = 0;     //开启按钮-状态断开
179         UInt32 m_devind = 0;    //索引号：默认状态第0个，PC端只插入一个CAN设备
              时均为0索引，插入多个同型号的CAN时索引号自动分配0，1,2.....
180         //UInt32 m_canind = 0;    //第几路CAN：一个CAN盒子中有多路CAN时选择，0
              表示第一路，1表示第二路，以此类推
181         UInt32 m_canind1 = 0, m_canind2 = 1;
182
183         VCI_CAN_OBJ[] m_recobj1 = new VCI_CAN_OBJ[1000]; //帧结构体
              VCI_CAN_OBJ 数组
```

```
184         VCI_CAN_OBJ[] m_recobj2 = new VCI_CAN_OBJ[1000]; //帧结构体
              VCI_CAN_OBJ2 数组
185
186         UInt32[] m_arrdevtype = new UInt32[20];//定义字段m_arrdevtype,20位一维
              数组，用于存放设备类型comboBox_devtype菜单中的项目的值
187         /
              *-----------------------------------------------------------------
              -----------*/
188
189         public Form1()
190         {
191             InitializeComponent();
192         }
193
194         //窗口开启时进行的操作（各种参数初始化）
195         private void Form1_Load(object sender, EventArgs e)
196         {
197             //窗口中的各种控件的初始值设定
198             //-----------------1、2CAN公用参数部分--------------
199             comboBox_DevIndex.SelectedIndex = 0;//索引号选择
200             //-------------------------------------------------
201             //第一路CAN
202             textBox_AccCode1.Text = "00000000";
203             textBox_AccMask1.Text = "FFFFFFFF";
204             textBox1_Time0.Text = "00";
205             textBox1_Time1.Text = "1C";
206             comboBox_Filter1.SelectedIndex = 0;              //接收所有类型
207             comboBox_Mode1.SelectedIndex = 2;                //还回测试模式
208             comboBox_FrameFormat1.SelectedIndex = 0;
209             comboBox_FrameType1.SelectedIndex = 0;
210             textBox_ID1.Text = "00000123";
211             textBox_Data1.Text = "00 01 02 03 04 05 06 07 ";
212             //第二路CAN
213             textBox_AccCode2.Text = "00000000";
214             textBox_AccMask2.Text = "FFFFFFFF";
215             textBox2_Time0.Text = "00";
216             textBox2_Time1.Text = "1C";
217             comboBox_Filter2.SelectedIndex = 0;              //接收所有类型
218             comboBox_Mode2.SelectedIndex = 2;                //还回测试模式
219             comboBox_FrameFormat2.SelectedIndex = 0;
220             comboBox_FrameType2.SelectedIndex = 0;
221             textBox_ID2.Text = "00000123";
222             textBox_Data2.Text = "00 01 02 03 04 05 06 07 ";
223
224             //设备类型选择菜单comboBox_devtype的设置部分
225             Int32 curindex = 0; //定义变量
226             comboBox_devtype.Items.Clear();//清空comboBox所有填充的items。
227             /*listview.clear()与listview.item.clear()的区别就是使用了
              listview.item.clear()后，listview控件中仍然保存着listviewitem项
              的结构，即listview有多个列，每列可能对应的列标题数据等。而当你使
              用了listview.clear()后，整个listview内保存数据的结构就没了。*/
228
229             curindex = comboBox_devtype.Items.Add("CANalyst-I");//添加项目
              DEV_USBCAN
230             m_arrdevtype[curindex] = DEV_USBCAN;
231             //comboBox_devtype.Items[2] = "VCI_USBCAN1";
```

```
232             //m_arrdevtype[2]=  VCI_USBCAN1 ;
233
234             curindex = comboBox_devtype.Items.Add("CANalyst-II");//添加项目
                  DEV_USBCAN2
235             m_arrdevtype[curindex] = DEV_USBCAN2;
236             //comboBox_devtype.Items[3] = "VCI_USBCAN2";
237             //m_arrdevtype[3]=  VCI_USBCAN2 ;
238
239             comboBox_devtype.SelectedIndex = 1; //选择菜单默认选择第二项
240             comboBox_devtype.MaxDropDownItems =
                  comboBox_devtype.Items.Count;//获取或设置要在 ComboBox 的下拉部
                  分中显示的最大项数。
241         }
242
243         //窗口关闭时进行的操作（进行CAN盒子的关闭操作）
244         private void Form1_FormClosed(object sender, FormClosedEventArgs e)
245         {
246             if (m_bOpen == 1)//如果开启状态值显示1表示开启，则进行下面的关闭操
                  作
247             {
248                 VCI_CloseDevice(m_devtype, m_devind);//源于controlcan.dll，VCI
                      关闭操作
249             }
250         }
251
252         //“连接/初始化”按钮按下执行的操作
253         private void buttonConnect_Click(object sender, EventArgs e)
254         {
255             if (m_bOpen == 1)//如果设备开启的，先执行关闭
256             {
257                 VCI_CloseDevice(m_devtype, m_devind);
258                 m_bOpen = 0;
259             }
260             else//如果设备处于关闭状态
261             {
262                 m_devtype = m_arrdevtype[comboBox_devtype.SelectedIndex];//选
                      择类型
263
264                 m_devind = (UInt32)comboBox_DevIndex.SelectedIndex;//选择索引
                      号
265                 if (VCI_OpenDevice(m_devtype, m_devind, 0) == 0)//判定连接函数
266                 {
267                     MessageBox.Show("打开设备失败,请检查设备类型和设备索引号是
                      否正确", "错误",
268                             MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
269                     return;
270                 }
271
272                 m_bOpen = 1;//状态设为1表示开启设备
273                 VCI_INIT_CONFIG config = new VCI_INIT_CONFIG();
274                 VCI_INIT_CONFIG config2 = new VCI_INIT_CONFIG();
275                 //以下装入初始化的初值(第1路CAN)
276                 config.AccCode = System.Convert.ToUInt32("0x" +
                      textBox_AccCode1.Text, 16);//验收码
277                 config.AccMask = System.Convert.ToUInt32("0x" +
                      textBox_AccMask1.Text, 16);//屏蔽码
```

```
278                 config.Timing0 = System.Convert.ToByte("0x" +
                      textBox1_Time0.Text, 16);//定时器0（设置波特率）
279                 config.Timing1 = System.Convert.ToByte("0x" +
                      textBox1_Time1.Text, 16);//定时器1（设置波特率）
280                 config.Filter = (Byte)(comboBox_Filter1.SelectedIndex + 1);//
                      滤波方式设置
281                 config.Mode = (Byte)comboBox_Mode1.SelectedIndex;//工作模式设
                      置
282                 VCI_InitCAN(m_devtype, m_devind, m_canind1, ref config);//初始
                      化第1路CAN
283
284                 //以下装入初始化的初值(第2路CAN)
285                 config2.AccCode = System.Convert.ToUInt32("0x" +
                      textBox_AccCode2.Text, 16);//验收码
286                 config2.AccMask = System.Convert.ToUInt32("0x" +
                      textBox_AccMask2.Text, 16);//屏蔽码
287                 config2.Timing0 = System.Convert.ToByte("0x" +
                      textBox2_Time0.Text, 16);//定时器0（设置波特率）
288                 config2.Timing1 = System.Convert.ToByte("0x" +
                      textBox2_Time1.Text, 16);//定时器1（设置波特率）
289                 config2.Filter = (Byte)(comboBox_Filter2.SelectedIndex + 1);//
                      滤波方式设置
290                 config2.Mode = (Byte)comboBox_Mode2.SelectedIndex;//工作模式设
                      置
291                 VCI_InitCAN(m_devtype, m_devind, m_canind2, ref config2);//初
                      始化第2路CAN
292             }
293             buttonConnect.Text = m_bOpen == 1 ? "断开" : "连接/初始化参数";//
                  改变连接按钮显示
294             timer1.Enabled = m_bOpen == 1 ? true : false;//定时器使能操作
295         }
296
297         //启动CAN按钮操作
298         private void button_StartCAN_Click(object sender, EventArgs e)
299         {
300             if (m_bOpen == 0)
301                 return;
302             VCI_StartCAN(m_devtype, m_devind, m_canind1);
303             VCI_StartCAN(m_devtype, m_devind, m_canind2);
304         }
305
306         //复位CAN按钮操作
307         private void button_StopCAN_Click(object sender, EventArgs e)
308         {
309             if (m_bOpen == 0)
310                 return;
311             VCI_ResetCAN(m_devtype, m_devind, m_canind1);
312             VCI_ResetCAN(m_devtype, m_devind, m_canind2);
313         }
314
315         //CAN1发送按钮操作,注意！允许不安全代码
316         unsafe private void button_Send1_Click(object sender, EventArgs e)
317         {
318             if (m_bOpen == 0)
319                 return;
320
```

```
321             VCI_CAN_OBJ sendobj1 = new VCI_CAN_OBJ();//定义结构体
322             //sendobj.Init();
323             sendobj1.RemoteFlag = (byte)comboBox_FrameFormat1.SelectedIndex;
324             sendobj1.ExternFlag = (byte)comboBox_FrameType1.SelectedIndex;
325             sendobj1.ID = System.Convert.ToUInt32("0x" + textBox_ID1.Text,
                  16);
326             int len = (textBox_Data1.Text.Length + 1) / 3;
327             sendobj1.DataLen = System.Convert.ToByte(len);
328             String strdata = textBox_Data1.Text;
329             int i = -1;
330             if (i++ < len - 1)
331                 sendobj1.Data[0] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
332             if (i++ < len - 1)
333                 sendobj1.Data[1] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
334             if (i++ < len - 1)
335                 sendobj1.Data[2] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
336             if (i++ < len - 1)
337                 sendobj1.Data[3] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
338             if (i++ < len - 1)
339                 sendobj1.Data[4] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
340             if (i++ < len - 1)
341                 sendobj1.Data[5] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
342             if (i++ < len - 1)
343                 sendobj1.Data[6] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
344             if (i++ < len - 1)
345                 sendobj1.Data[7] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
346
347             if (VCI_Transmit(m_devtype, m_devind, m_canind1, ref sendobj1, 1)
                  == 0)
348             {
349                 MessageBox.Show("发送失败", "错误",
350                         MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
351             }
352         }
353
354         //CAN2发送按钮操作,注意！允许不安全代码
355         unsafe private void button_Send2_Click(object sender, EventArgs e)
356         {
357             if (m_bOpen == 0)
358                 return;
359
360             VCI_CAN_OBJ sendobj2 = new VCI_CAN_OBJ();
361             //sendobj.Init();
362             sendobj2.RemoteFlag = (byte)comboBox_FrameFormat2.SelectedIndex;
363             sendobj2.ExternFlag = (byte)comboBox_FrameType2.SelectedIndex;
364             sendobj2.ID = System.Convert.ToUInt32("0x" + textBox_ID2.Text,
                  16);
365             int len = (textBox_Data2.Text.Length + 1) / 3;
```

```
366             sendobj2.DataLen = System.Convert.ToByte(len);
367             String strdata = textBox_Data2.Text;
368             int i = -1;
369             if (i++ < len - 1)
370                 sendobj2.Data[0] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
371             if (i++ < len - 1)
372                 sendobj2.Data[1] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
373             if (i++ < len - 1)
374                 sendobj2.Data[2] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
375             if (i++ < len - 1)
376                 sendobj2.Data[3] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
377             if (i++ < len - 1)
378                 sendobj2.Data[4] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
379             if (i++ < len - 1)
380                 sendobj2.Data[5] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
381             if (i++ < len - 1)
382                 sendobj2.Data[6] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
383             if (i++ < len - 1)
384                 sendobj2.Data[7] = System.Convert.ToByte("0x" +
                      strdata.Substring(i * 3, 2), 16);
385
386             if (VCI_Transmit(m_devtype, m_devind, m_canind2, ref sendobj2, 1)
                  == 0)
387             {
388                 MessageBox.Show("发送失败", "错误",
389                         MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
390             }
391         }
392
393         //信息窗口1清屏
394         private void button_Clear1_Click(object sender, EventArgs e)
395         {
396             listBox_Info1.Items.Clear();
397         }
398         //信息窗口2清屏
399         private void button_Clear2_Click(object sender, EventArgs e)
400         {
401             listBox_Info2.Items.Clear();
402         }
403
404         //定时器操作部分，触发周期：50ms，注意！！！使用不安全代码允许unsafe
405         unsafe private void timer1_Tick(object sender, EventArgs e)
406         {
407             UInt32 res2 = new UInt32();
408             UInt32 res1 = new UInt32();
409
410             String str2 = "";
411             String str = "";
412
```

```
413             //------------------------------------------------------------------
                  ----------------------
414             if (flag_can == 1)
415             {
416                 res2 = VCI_Receive(m_devtype, m_devind, m_canind2, ref
                      m_recobj2[0], 1000, 100);//第2路CAN的接收函数
417                 for (UInt32 i = 0; i < res2; i++)
418                 {
419                     //VCI_CAN_OBJ obj = (VCI_CAN_OBJ)Marshal.PtrToStructure
                      ((IntPtr)((UInt32)pt + i * Marshal.SizeOf(typeof
                      (VCI_CAN_OBJ))), typeof(VCI_CAN_OBJ));
420
421                     str2 = "接收到数据: ";
422                     str2 += "  帧ID:0x" + System.Convert.ToString(m_recobj2
                      [i].ID, 16);
423                     str2 += "  帧格式:";
424                     if (m_recobj1[i].RemoteFlag == 0)
425                         str2 += "数据帧 ";
426                     else
427                         str2 += "远程帧 ";
428                     if (m_recobj1[i].ExternFlag == 0)
429                         str2 += "标准帧 ";
430                     else
431                         str2 += "扩展帧 ";
432
433                     ////////////////////////////////////////
434                     if (m_recobj1[i].RemoteFlag == 0)
435                     {
436                         str2 += "数据: ";
437                         byte len = (byte)(m_recobj1[i].DataLen % 9);
438                         byte j = 0;
439                         fixed (VCI_CAN_OBJ* m_recobj_2 = &m_recobj2[i])
440                         {
441                             if (j++ < len)
442                                 str2 += " " + System.Convert.ToString
                                  (m_recobj_2->Data[0], 16);
443                             if (j++ < len)
444                                 str2 += " " + System.Convert.ToString
                                  (m_recobj_2->Data[1], 16);
445                             if (j++ < len)
446                                 str2 += " " + System.Convert.ToString
                                  (m_recobj_2->Data[2], 16);
447                             if (j++ < len)
448                                 str2 += " " + System.Convert.ToString
                                  (m_recobj_2->Data[3], 16);
449                             if (j++ < len)
450                                 str2 += " " + System.Convert.ToString
                                  (m_recobj_2->Data[4], 16);
451                             if (j++ < len)
452                                 str2 += " " + System.Convert.ToString
                                  (m_recobj_2->Data[5], 16);
453                             if (j++ < len)
454                                 str2 += " " + System.Convert.ToString
                                  (m_recobj_2->Data[6], 16);
455                             if (j++ < len)
456                                 str2 += " " + System.Convert.ToString
```

```
                    (m_recobj_2->Data[7], 16);
457                     }
458                 }
459                 listBox_Info2.Items.Add(str2);//显示信息框中显示收到的字符串
460                 listBox_Info2.SelectedIndex = listBox_Info2.Items.Count - 1;//选择最新出现的一行，selectedIndex是document.form.site的当前选择项的索引值，从0开始从上往下依次递增，没选中是-1
461
462             }
463             flag_can = 2;//标志置位
464         }
465         else if (flag_can == 2)
466         {
467             res1 = VCI_Receive(m_devtype, m_devind, m_canind1, ref m_recobj1[0], 1000, 100);//第1路CAN的接收函数
468             for (UInt32 i = 0; i < res1; i++)
469             {
470                 //VCI_CAN_OBJ obj = (VCI_CAN_OBJ)Marshal.PtrToStructure((IntPtr)((UInt32)pt + i * Marshal.SizeOf(typeof(VCI_CAN_OBJ))), typeof(VCI_CAN_OBJ));
471
472                 str = "接收到数据: ";
473                 str += "  帧ID:0x" + System.Convert.ToString(m_recobj1[i].ID, 16);
474                 str += "  帧格式:";
475                 if (m_recobj1[i].RemoteFlag == 0)
476                     str += "数据帧 ";
477                 else
478                     str += "远程帧 ";
479                 if (m_recobj1[i].ExternFlag == 0)
480                     str += "标准帧 ";
481                 else
482                     str += "扩展帧 ";
483
484                 ////////////////////////////////////////
485                 if (m_recobj1[i].RemoteFlag == 0)
486                 {
487                     str += "数据: ";
488                     byte len = (byte)(m_recobj1[i].DataLen % 9);
489                     byte j = 0;
490                     fixed (VCI_CAN_OBJ* m_recobj_1 = &m_recobj1[i])
491                     {
492                         if (j++ < len)
493                             str += " " + System.Convert.ToString(m_recobj_1->Data[0], 16);
494                         if (j++ < len)
495                             str += " " + System.Convert.ToString(m_recobj_1->Data[1], 16);
496                         if (j++ < len)
497                             str += " " + System.Convert.ToString(m_recobj_1->Data[2], 16);
498                         if (j++ < len)
499                             str += " " + System.Convert.ToString(m_recobj_1->Data[3], 16);
500                         if (j++ < len)
```

```
501                                 str += " " + System.Convert.ToString
                        (m_recobj_1->Data[4], 16);
502                                 if (j++ < len)
503                                     str += " " + System.Convert.ToString
                        (m_recobj_1->Data[5], 16);
504                                 if (j++ < len)
505                                     str += " " + System.Convert.ToString
                        (m_recobj_1->Data[6], 16);
506                                 if (j++ < len)
507                                     str += " " + System.Convert.ToString
                        (m_recobj_1->Data[7], 16);
508                                 }
509                             }
510
511                             listBox_Info1.Items.Add(str);//显示信息框中显示收到的字符
                            串
512                             listBox_Info1.SelectedIndex = listBox_Info1.Items.Count -
                            1;//选择最新出现的一行，selectedIndex是document.form.site的
                            当前选择项的索引值，从0开始从上往下依次递增，没选中是-1
513                         }
514                         flag_can = 1;//标志置位
515                     }
516                 }
517             }
518 }
519
```