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

```
50      public byte DataLen;          //数据长度
51      public fixed byte Data[8];    //数据
52      public fixed byte Reserved[3];//保留位
53  }
54
55  //3.定义初始化CAN的数据类型
56  public struct VCI_INIT_CONFIG
57  {
58      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。
59      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。
60      public UInt32 Reserved;//保留。
61      public byte Filter;   //0或1接收所有帧。2标准帧滤波，3是扩展帧滤波。
62      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。
63      public byte Timing1;
64      /*
65     CAN波特率 Timing0(BTR0)  Timing1(BTR1)
66     10 Kbps        0x31         0x1C
67     20 Kbps        0x18         0x1C
68     40 Kbps        0x87         0xFF
69     50 Kbps        0x09         0x1C
70     80 Kbps        0x83         0xFF
71    100 Kbps        0x04         0x1C
72    125 Kbps        0x03         0x1C
73    200 Kbps        0x81         0xFA
74    250 Kbps        0x01         0x1C
75    400 Kbps        0x80         0xFA
76    500 Kbps        0x00         0x1C
77    666 Kbps        0x80         0xB6
78    800 Kbps        0x00         0x16
79   1000 Kbps        0x00         0x14
80   33.33 Kbps       0x09         0x6F
81   66.66 Kbps       0x04         0x6F
82   83.33 Kbps       0x03         0x6F
83   */
84      public byte Mode;     //模式，0表示正常模式，1表示只听模式,2自测模式
85  }
86
87  /*------------其他数据结构描述--------------------------------*/
88  //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。
89  public struct VCI_BOARD_INFO1//(第二个)
90  {
91      public UInt16 hw_Version;//申明hw_Version：硬件版本号，用16进制表示。
92      public UInt16 fw_Version;//申明fw_Version：固件版本号，用16进制表示。
93      public UInt16 dr_Version;//申明dr_Version：驱动程序版本号，用16进制表示。
94      public UInt16 in_Version;//申明dr_Version：接口库版本号，用16进制表示。
95      public UInt16 irq_Num; //申明irq_Num：保留参数。
96      public byte can_Num; //申明can_Num：表示有几路CAN通道。
97      public byte Reserved;//Reserved系统保留。
98      [MarshalAs(UnmanagedType.ByValArray, SizeConst = 8)] public byte[]
          str_Serial_Num;//二维数组转一维数组
99      [MarshalAs(UnmanagedType.ByValArray, SizeConst = 16)] public byte[]
```

```
        str_hw_Type;//二维数组转一维数组
100     [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。
101 }
102
103 /*------------数据结构描述完成--------------------------------*/
104
105 /*CHGDESIPANDPORT 结构体用于装载更改 CANET_UDP 与 CANET_TCP 的目标 IP 和
106 端口的必要信息。此结构体在 CANETE_UDP 与 CANET_TCP 中使用。
107 typedef struct _tagChgDesIPAndPort {
108 char szpwd[10];     //更改目标 IP 和端口所需要的密码
109 char szdesip[20];   //所要更改的目标 IP
110 int desport;        //所要更改的目标端口，比如为 4000。
111 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。
112 } CHGDESIPANDPORT;*/
113 public struct CHGDESIPANDPORT   //CANET 通讯结构体
114 {
115     [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”。
116     [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”。
117     public Int32 desport;
118
119     public void Init()
120     {
121         szpwd = new byte[10];
122         szdesip = new byte[20];
123     }
124 }
125
126 namespace CANalyst_II_CANOpen_方案    //方案空间(类似主函数)
127 {
128     public partial class Form1 : Form   //部分类Form1
129     {
130         /
              *-----------------------------------------------------------------
              -----------*/
131         const int DEV_USBCAN = 3;   //全局变量定义，这里USBCANalyst-I为3
132         const int DEV_USBCAN2 = 4;  //全局变量定义，这里USBCANalyst-II为4
133         /// <summary>
134         ///
135         /// </summary>
136         /// <param name="DeviceType"></param>
137         /// <param name="DeviceInd"></param>
138         /// <param name="Reserved"></param>
139         /// <returns></returns>
140         /*------------兼容ZLG的函数描述--------------------------------*/
141         [DllImport("controlcan.dll")]
142         static extern UInt32 VCI_OpenDevice(UInt32 DeviceType, UInt32
            DeviceInd, UInt32 Reserved);
143         [DllImport("controlcan.dll")]
144         static extern UInt32 VCI_CloseDevice(UInt32 DeviceType, UInt32
```

```
            DeviceInd);
145         [DllImport("controlcan.dll")]
146         static extern UInt32 VCI_InitCAN(UInt32 DeviceType, UInt32 DeviceInd,
              UInt32 CANInd, ref VCI_INIT_CONFIG pInitConfig);
147
148         [DllImport("controlcan.dll")]
149         static extern UInt32 VCI_ReadBoardInfo(UInt32 DeviceType, UInt32
              DeviceInd, ref VCI_BOARD_INFO pInfo);
150
151         [DllImport("controlcan.dll")]
152         static extern UInt32 VCI_GetReceiveNum(UInt32 DeviceType, UInt32
              DeviceInd, UInt32 CANInd);
153         [DllImport("controlcan.dll")]
154         static extern UInt32 VCI_ClearBuffer(UInt32 DeviceType, UInt32
              DeviceInd, UInt32 CANInd);
155
156         [DllImport("controlcan.dll")]
157         static extern UInt32 VCI_StartCAN(UInt32 DeviceType, UInt32 DeviceInd,
               UInt32 CANInd);
158         [DllImport("controlcan.dll")]
159         static extern UInt32 VCI_ResetCAN(UInt32 DeviceType, UInt32 DeviceInd,
               UInt32 CANInd);
160
161         [DllImport("controlcan.dll")]
162         static extern UInt32 VCI_Transmit(UInt32 DeviceType, UInt32 DeviceInd,
               UInt32 CANInd, ref VCI_CAN_OBJ pSend, UInt32 Len);
163
164         [DllImport("controlcan.dll")]
165         static extern UInt32 VCI_Receive(UInt32 DeviceType, UInt32 DeviceInd,
              UInt32 CANInd, ref VCI_CAN_OBJ pReceive, UInt32 Len, Int32
              WaitTime);
166
167         /*------------其他函数描述--------------------------------*/
168
169         [DllImport("controlcan.dll")]
170         static extern UInt32 VCI_ConnectDevice(UInt32 DevType, UInt32
              DevIndex);
171         [DllImport("controlcan.dll")]
172         static extern UInt32 VCI_UsbDeviceReset(UInt32 DevType, UInt32
              DevIndex, UInt32 Reserved);
173         [DllImport("controlcan.dll")]
174         static extern UInt32 VCI_FindUsbDevice(ref VCI_BOARD_INFO1 pInfo);
175         /*------------函数描述结束--------------------------------*/
176
177         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)
178
179         UInt32 m_bOpen = 0;     //开启按钮-状态断开
180         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.....
181         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
182
183         VCI_CAN_OBJ[] m_recobj = new VCI_CAN_OBJ[1000]; //帧结构体 VCI_CAN_OBJ
               数组
184
```

```
185         UInt32[] m_arrdevtype = new UInt32[20];
186         /
              *-------------------------------------------------------------------
              -----------*/
187
188         public Form1()
189         {
190             InitializeComponent();
191         }
192
193         private void Form1_Load(object sender, EventArgs e)
194         {
195
196         }
197     }
198 }
199
```