Interface Description of HWUG Multifunctional Electronic Sign and Batch Dynamic Library

theme	Interface Description of HWUG Multifunctional Electronic Sign and Batch Dynamic Library
version	V1.0.0.2
content	Multi-functional original handwriting signing and approval toolkit and development interface description
creation time	February 21, 2020
founder	Leon
turnover time	February 21, 2020

Document change record

Changed by	date	Change content
Leon		

Main review opinions of documents

Product group

reviewers	date	idea

QA group

reviewers	date	idea

development platform

Development environment and tools

tool	function
C++	Description of dynamic library file interface

Dependency library description

Document name	explain
UgeeSignFIDCam.dll	Multifunctional dynamic library (signature, fingerprint, ID card, camera)

UgeeSignFIDCam dynamic library environment support

system	Windows XP、Windows 7、Windows 8、Windows 10

UgeeSignFIDCam dynamic library interface description

----------1----------

● Turn on the device

```
/*********************************
* Parameter: None
* Return value: 0: Open successfully; Other values: open failed * * * * * * * * * * * * * * * *
* * * * * * * * * * * * * * * * *
int UgeeOpenDevice();
```

----------2----------

● Register button positions (up to 32 buttons can be registered)

```
/*********************************
* Parameters: button position information, top@left@bottom@right@btn_id@. It is recommended
  that btn_id start at 10.
* Return value: 0: success; Other values: failed
**********************************/
int UgeeRegisterBtnPosInfo(char* szBtnPos);
```

----------3----------

● Register button response callback function

```
/*********************************
* Parameters: Callback function pointer
* Return value: 0: success; Other values: failed
**********************************/
int UgeeRegisterBtnCallBack(RESPONSE_MSG_FUNC func);
```

----------4----------

● The logout button responds to the callback function.

```
/*********************************
* Parameters: Callback function pointer
* Return value: 0: success; Other values: failed
**********************************/
```

```
int UgeeUnregisterBtnCallBack(RESPONSE_MSG_FUNC func);

//RESPONSE_MSG_FUNC
Typedef int( stdcall* RESPONSE_MSG_FUNC)(int status); Status :0
device available -1 device unavailable
1 signature completed, 2 signature cancelled, 3 screen cleared, 4 click OK, but there is no handwriting.
5 fingerprint completion other values: btn_id
```

---5---

● Start signing

```
/*********************************
* Parameters:
     Parameter 1: left coordinate of signature box,
     based on client coordinate parameter 2: top
     coordinate of signature box.
     Parameter          3:
     Signature  Box  Width
     Parameter          4:
     Signature Box Height
     Parameter          5:
     Signature Pen Width
     Parameter 6: Signature image path (never put it in the system packing path) * Return value: 0:
     success; Other values: failed
**********************************/

int UgeeStartSign(int left, int top, int width, int height, int penWidth, const char* szSignPath);
```

---6---

● Began to hold back fingerprints

```
/*********************************
* Parameters:
     Parameter 1: Fingerprint quality
     Parameter 2: Signature Image Path
     Parameter 3: Signature+Fingerprint Image Path
* Return value: 0: success; Other values: failed
**********************************/
int UgeeStartFinger(int quality, const char* szFingerPath, const char* szSignFPath);
```

---7---

● Close the window

```
/*********************************
* Return value: 0: success; Other values: failed
**********************************/
```

```
int UgeeCloseWindow();
```

----------8----------

● Turn off the equipment

```
/*********************************
* Parameter: None
* Return value: 0: success; Other values: failed
*********************************/
int UgeeCloseDevice();
```

----------9----------

● Turn on the camera

```
/*********************************
* Return value: 0: success; Other values: failed
*********************************/
int UgeeOpenCamera();
```

----------10----------

● Get the resolution supported by the camera.

```
/*********************************
* Return value: resolution, such as "640*480@800*600"
*********************************/
char* UgeeGetResolution();
```

----------11----------

● Set the camera resolution

```
/*********************************
* Parameters: Resolution
* Return value: 0: success; Other values: failed
*********************************/
int UgeeSetResolution(char* szResolution);
```

----------12----------

● Left and right mirror image

```
/*********************************
* Return value: 0: success; Other values: failed
*********************************/
int UgeeSetLRMirror();
```

----------13----------

● Acquire an image frame

```
/*********************************
* Return value: 0: success; Other values: failed
* Parameter: image frame path
*********************************/
int UgeeGetImgFrame(char* szResolution);
```

----------14----------

● take a picture

```
/*********************************
* Return value: 0: success; Other values: failed
* Parameter: photo path
*********************************/
int UgeeTakePhoto(const char* szPhotoPath);
```

----------15----------

● Turn off the camera

```
/*********************************
* Return value: 0: success; Other values: failed
*********************************/
int UgeeCloseCamera();
```

Example of C# calling c++Dll interface

1, the declaration of the function

```
/// <summary>
///Turn on the device
/// </summary>
/// <returns > return value: 0: successful opening; Other values: failed to open < /returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeOpenDevice")]
public extern static int UgeeOpenDevice();

/// <summary>
///Register the button position, up to 32 buttons can be registered.
/// </summary>
//< param name = "szbtnpos" > parameters: button position information, top@left@bottom@right@btn_id@. It is recommended that btn_id start at 10 < /param >
/// <returns > return value: 0: success; Other values: failed < /returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeRegisterBtnPosInfo")]
public extern static int UgeeRegisterBtnPosInfo(string szBtnPos);

/// <summary>
///Register the button to respond to the callback function
/// </summary>
//< param name = "status" > parameters: callback function pointer < /param >
/// <returns > return value: 0: success; Other values: failed < /returns >
public delegate int UgeeRegisterBtnCallBack_delegate(int status);

[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, CharSet = CharSet.Ansi, EntryPoint = "UgeeRegisterBtnCallBack")]
public extern static int UgeeRegisterBtnCallBack(UgeeRegisterBtnCallBack_delegate callback);

/// <summary>
///The logout button responds to the callback function.
/// </summary>
```

```
//< param name = "status" > return value: 0: success; Other values: failed < /param >

/// <returns > parameters: callback function pointer < /returns >

public delegate int UgeeUnregisterBtnCallBack_delegate(int status);


[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, CharSet = CharSet.Ansi, EntryPoint = "UgeeUnregisterBtnCallBack")]

public extern static int UgeeUnregisterBtnCallBack(UgeeUnregisterBtnCallBack_delegate callback);


/// <summary>

///Start signing

/// </summary>

//< param name = "left" > parameter 1: the left coordinate of the signature box, based on the client coordinate < /param >

//< param name = "top" > Parameter 2: Top coordinate of signature box < /param >

//< param name = "width" > parameter 3: width of signature box < /param >

//< param name = "height" > parameter 4: height of signature box < /param >

//< param name = "penwidth" > parameter 5: pen width < /param >

//< param name = "szsignpath" > Parameter 6: Signature image path (never put it in the system packing path) < /param >

/// <returns > return value: 0: success; Other values: failed < /returns >

[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeStartSign")]

public extern static int UgeeStartSign(int left, int top, int width, int height, int penWidth, string szSignPath);


/// <summary>

///began to suppress fingerprints

/// </summary>

//< param name = "szfingerpath" > parameter 1: signature image path < /param >

//< param name = "szsignfpath" > parameter 2: signature+fingerprint image path < /param >

/// <returns > return value: 0: success; Other values: failed < /returns >

[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeStartFinger")]

public extern static int UgeeStartFinger(int quality, string szFingerPath, string szSignFPath);


/// <summary>

///Close the window

/// </summary>
```

```
/// <returns > return value: 0: success; Other values: failed < /returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeCloseWindow")]
public extern static int UgeeCloseWindow();

/// <summary>
///Turn off the device
/// </summary>
/// <returns > return value: 0: success; Other values: failed < /returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeCloseDevice")]
public extern static int UgeeCloseDevice();

/// <summary>
///Turn on the camera
/// </summary>
/// <returns > return value: 0: success; Other values: failed < /returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeOpenCamera")]
public extern static int UgeeOpenCamera();

/// <summary>
///Get the resolution supported by the camera.
/// </summary>
/// <returns > return value: resolution, such as "640 * 480 @ 800 * 600" </returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeGetResolution")]
public extern static IntPtr UgeeGetResolution();

/// <summary>
///Set the camera resolution
/// </summary>
//< param name = "width" > parameters: resolution < /param >
//< param name = "height" > parameters: resolution < /param >
/// <returns > return value: 0: success; Other values: failed < /returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeSetResolution")]
```

```
public extern static int UgeeSetResolution(string Resolution);

/// <summary>
///Left and right mirror images
/// </summary>
/// <returns > return value: 0: success; Other values: failed < /returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeSetLRMirror")]
public extern static int UgeeSetLRMirror();

/// <summary>
///Get an image frame
/// </summary>
/// <returns > return value: image frame base64</returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeGetImgFrame")]
public extern static int UgeeGetImgFrame(string framePath);

/// <summary>
///Take pictures
/// </summary>
/// <returns > return value: photo base64</returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeTakePhoto")]
public extern static int UgeeTakePhoto(string photoPath);

/// <summary>
///Turn off the camera
/// </summary>
/// <returns > return value: 0: success; Other values: failed < /returns >
[DllImport("UgeeSignFIDCam.dll", CallingConvention = CallingConvention.StdCall, EntryPoint = "UgeeCloseCamera")]
public extern static int UgeeCloseCamera();
```

2. Register the callback function

```
//Register the callback function
UgeeRegisterBtnCallBack_delegate Callback = new
```

```
Ugee.UgeeRegisterBtnCallBack_delegate(RESPONSE_MSG_FUNC);
Ugee.UgeeRegisterBtnCallBack(Callback);
```

3. The implementation of callback function.

```
/// <summary>
///response function
/// </summary>
/// <param name="status"></param>
/// <returns></returns>
private int RESPONSE_MSG_FUNC(int status) {
    switch (status)
    {
        Case 0:// Device available
        break; Case -1:// The
        device is unavailable
        break; Case 1:// Signature
        completed
            Ugee.UgeeStartFinger(150,Ugee.FingerImgFullPath,
            Ugee.SignFingerImgFullPath); break;
        Case 2:// Cancel signature Ugee.HasSign1 = false;
            ReleaseDevice(); break; Case 3:// Clear the screen
            break;
        Case 4:// Click OK but there is no handwriting
            ReleaseDevice (); break;
        Case 5:// Fingerprint complete
            SignFingerOK(); break;
        case 10:
            EvaluateOK(10); break;
        case 11:
            EvaluateOK(11); break;
        case 12:
            EvaluateOK(12); break;
        case 13:
            EvaluateOK(13); break;
        case 14:
            EvaluateOK(14); break;
        case 15:
            EvaluateOK(15); break;
        default:
            break;
```

```
        }
            return 0;
            }
```

4. Example of Registration Button Information

```
/// <summary>
/// 按钮位置信息
/// </summary>
/// <param name="btn"></param>
/// <returns></returns>
private string BtnPostionInfo(PictureBox btn)
{
    return btn.Top + "@" + btn.Left + "@" + btn.Bottom + "@" + btn.Right + "@" + btn.Tag.ToString() + "@";
}
UgeeRegisterBtnPosInfo(BtnPostionInfo(pb_EV1) + BtnPostionInfo(pb_EV2) + BtnPostionInfo(pb_EV3) + BtnPostionInfo(pb_E1) + BtnPostionInfo(pb_E2) + BtnPostionInfo(pb_E3));
```

5、打开摄像头

```
private void btnOpenCamera_Click(object sender, EventArgs e)
{
      int ret = Ugee.UgeeOpenCamera();
      if (ret == 0)
      {
          CameraOpen = true;
          TimerCamera =新系统。计时器。计时器(100)；//实例化计时器类,设置间隔时间为100 毫秒；
          定时器摄像机。消逝+=新系统。timers . ElapsedEventHandler(the out)；//到达时间的时候执行事件；
          定时器摄像机。AutoReset = true//设置是执行一次(错误)还是一直执行(真)；
          定时器摄像机。Enabled = true//是否执行系统。计时器计时器流逝事件；
      }
    }
```

6. Get the image frame in real time and display it.

```
public void theout(object source, System.Timers.ElapsedEventArgs e)
{
    try
    {
        Ugee.UgeeGetImgFrame("");
        if (File.Exists(FramePath))
        {
            FileStream fileStream = new FileStream(FramePath, FileMode.Open, FileAccess.Read);
            int byteLength = (int)fileStream.Length;
            byte[] fileBytes = new byte[byteLength];
            fileStream.Read(fileBytes, 0, byteLength);
            //The file stream is closed and the file is unlocked.
            fileStream.Close();
            pbCamera.BackgroundImage = Image.FromStream(new MemoryStream(fileBytes));
        }
    }
    catch (Exception)
    {}
}
```

7, get the camera resolution

```
private void btnGetResolution_Click(object sender, EventArgs e)
{
  if (CameraOpen)
  {
      string res = Marshal.PtrToStringAnsi(Ugee.UgeeGetResolution());
      string[] ResolutionList = res.Split('@');
      cbResolution.Items.Clear();
      foreach (var item in ResolutionList)
      {
          cbResolution.Items.Add(item);
```

```
        }
        if (ResolutionList.Length > 0)
        {
            cbResolution.SelectedIndex = 0;
        }
    }
}
```

7, set the camera resolution

```
private void btnSetResolution_Click(object sender, EventArgs e)

{
  if (CameraOpen)
    {
        if (cbResolution.Items.Count > 0)
        {
  //"640*480@800*600@1024*768@1280*720@1600*1200@1920*1080@2048*1536@2560*1440@2592*194
4";
            Ugee.UgeeSetResolution(cbResolution.SelectedItem.ToString());
        }
        else
        {
            Ugee.UgeeSetResolution("1920*1080");
        }
    }
}
```

8. Left and right mirror images

```
private void btnSetLRMirror_Click(object sender, EventArgs e)
  {
        if (CameraOpen)
        {
            Ugee.UgeeSetLRMirror();
        }
  }
```

9. Take pictures

```
private void btnPhoto_Click(object sender, EventArgs e)
{
  try
  {
      if (CameraOpen)
      {
          Ugee.UgeeTakePhoto("");
          if (File.Exists(PhotoPath))
          {
              FileStream fileStream = new FileStream(PhotoPath, FileMode.Open,
FileAccess.Read);
              int byteLength = (int)fileStream.Length;
              byte[] fileBytes = new byte[byteLength];
              fileStream.Read(fileBytes, 0, byteLength);
              //The file stream is closed and the file is unlocked.
              fileStream.Close();
              pbPhoto.BackgroundImage = Image.FromStream(new
MemoryStream(fileBytes));
                  }
              }
          }
          catch (Exception)
          {
              //throw;
          }
      }
```

C# Call ID Interface Example

1, the declaration of the function

```
[DllImport("termb.dll")]
public static extern int InitComm(int port); //Connect the ID card reader

[DllImport("termb.dll")]
public static extern int InitCommExt(); //Automatically search the ID card reader and connect it.

[DllImport("termb.dll")]
public static extern int CloseComm(); //Disconnect from ID card reader

[DllImport("termb.dll")]
公共静态extern int Authenticate(); //判断是否有放卡,且是否身份证

[DllImport("termb.dll")]
公共静态extern int Read _ Content(int index); //读卡操作,信息文件存储在动态链接库所在下

[DllImport("termb.dll")]
公共静态extern int read content(int index); //读卡操作,信息文件存储在动态链接库所在下

[DllImport("termb.dll")]
public static extern int gets amid(StringBuilder SAMID); //获取萨摩亚模块编号

[DllImport("termb.dll")]
public static extern int GetSAMIDEx(StringBuilder SAMID); //获取萨摩亚模块编号(10 位编号)

[DllImport("termb.dll")]
public static extern int GetBmpPhoto(string PhotoPath);//解析身份证照片

[DllImport("termb.dll")]
public static extern int getbmppphototomem(byte[]imageData, int cbImageData); //解析身份证照片

[DllImport("termb.dll")]
```

```
public static extern int GetBmpPhotoExt();//解析身份证照片

[DllImport("termb.dll")]
公共静态extern int Reset _ SAM(); //重置萨姆（男子名）模块

[DllImport("termb.dll")]
public static extern int GetSAMStatus(); //获取萨摩亚模块状态

[DllImport("termb.dll")]
public static extern int GetCardInfo(int index, StringBuilder value);//解析身份证信息

[DllImport("termb.dll")]
public static extern int ExportCardImageV();//生成竖版身份证正反两面图片(输出目录：dll 所在目录的cardv.jpg 和 SetCardJPGPathNameV 指定路径)

[DllImport("termb.dll")]
public static extern int ExportCardImageH();//生成横版身份证正反两面图片(输出目录：dll 所在目录的cardh.jpg 和 SetCardJPGPathNameH 指定路径)

[DllImport("termb.dll")]
public static extern int SetTempDir(string DirPath);//设置生成文件临时目录

[DllImport("termb.dll")]
public static extern int GetTempDir(StringBuilder path, int cbPath);//获取文件生成临时目录

[DllImport("termb.dll")]
public static extern void GetPhotoJPGPathName(StringBuilder path, int CB path); //获取使用jpeg文件交换格式存储的编码图像文件扩展名头像全路径名

[DllImport("termb.dll")]
public static extern int SetPhotoJPGPathName(字符串路径); //设置使用jpeg文件交换格式存储的编码图像文件扩展名头像全路径名

[DllImport("termb.dll")]
public static extern int setcardjgpathnamev(字符串路径); //设置竖版身份证正反两面图片全路径
```

```
[DllImport("termb.dll")]
public static extern int GetCardJPGPathNameV(StringBuilder path, int cbPath);//获取竖版Full path of front and back pictures of ID card

[DllImport("termb.dll")]
public static extern int SetCardJPGPathNameH(字符串路径); //设置横版身份证正反两面图片全 路径

[DllImport("termb.dll")]
public static extern int GetCardJPGPathNameH(StringBuilder path, int cbPath);//获取横版Full path of front and back pictures of ID card

[DllImport("termb.dll")]
public static extern int getName(StringBuilder data, int cbData);//获取姓名

[DllImport("termb.dll")]
public static extern int getSex(StringBuilder data, int cbData);//获取性别

[DllImport("termb.dll")]
public static extern int getNation(StringBuilder data, int cbData);//获取民族

[DllImport("termb.dll")]
public static extern int getBirthdate(StringBuilder data, int cbData);//获取生日(YYYYMMDD)

[DllImport("termb.dll")]
public static extern int getAddress(StringBuilder data, int cbData);//获取地址

[DllImport("termb.dll")]
public static extern int getIDNum(StringBuilder data, int cbData);//获取身份证号

[DllImport("termb.dll")]
public static extern int getIssue(StringBuilder data, int cbData);//获取签发机关

[DllImport("termb.dll")]
public static extern int getEffectedDate(StringBuilder data, int cbData);//获取有效期起始日期(YYYYMMDD)
```

```
[DllImport("termb.dll")]
        public static extern int getExpiredDate(StringBuilder data, int cbData);//获取有效期截止日期(YYYYMMDD)

[DllImport("termb.dll")]
public static extern int getbmpphotobase 64(StringBuilder data, int CB data); //获取位图文件的扩展名(Bitmap)头 像Base64编码

[DllImport("termb.dll")]
public static extern int getjpgphotobase 64(StringBuilder data, int CB data); //获取使用jpeg文件交换格式存储的编码图像文件扩展名头 像Base64编码

[DllImport("termb.dll")]
public static extern int getjpgcardbase 64v(StringBuilder data, int CB data); //获取竖版身 份证正反两面使用jpeg文件交换格式存储的编码图像文件扩展名图像base64编码字符串

[DllImport("termb.dll")]
public static extern int getjpgcardbase 64h(StringBuilder data, int CB data); //获取横版身 份证正反两面使用jpeg文件交换格式存储的编码图像文件扩展名图像base64编码字符串

[DllImport("termb.dll")]
public static extern int hid voice(int nVoice); //语音提示..仅适用于与带(hide的过去式)隐藏¡遮蔽语音设备的身 份证阅读器（如ID200)

[DllImport("termb.dll")]
public static extern int IC_SetDevNum(int iPort, StringBuilder data, int cbdata);//设置发卡器序列号

[DllImport("termb.dll")]
public static extern int IC_GetDevNum(int iPort, StringBuilder data, int cbdata);//获取发卡器序列号

[DllImport("termb.dll")]
public static extern int IC_GetDevVersion(int iPort, StringBuilder data, int cbdata);//设置发卡器序列号

[DllImport("termb.dll")]
public static extern int IC_WriteData(int iPort, int keyMode, int sector, int idx,
```

```
StringBuilder key, StringBuilder data, int cbdata, ref uint snr); //Write
data
```

```
[DllImport("termb.dll")]
public static extern int IC_ReadData(int iPort, int keyMode, int sector, int idx,
StringBuilder key, StringBuilder data, int cbdata, ref uint snr); //du data
```

```
[DllImport("termb.dll")]
public static extern int IC_GetICSnr(int iPort, ref uint snr); //Read the physical card
number of IC card
```

```
[DllImport("termb.dll")]
public static extern int IC_GetIDSnr(int iPort, StringBuilder data, int cbdata); //Read
the physical card number of the ID card
```

```
[DllImport("termb.dll")]
public static extern int getEnName(StringBuilder data, int cbdata); //Get English name
```

```
[DllImport("termb.dll")]
public static extern int getCnName(StringBuilder data, int cbdata); //Get the Chinese
name
```

```
[DllImport("termb.dll")]
public static extern int getPassNum(StringBuilder data, int cbdata); //Get the Hong
Kong, Macao and Taiwan Residence Pass Number
```

```
[DllImport("termb.dll")]
public static extern int getVisaTimes(); //Get the issuance times
```

```
[DllImport("termb.dll")]
public static extern int IC_ChangeSectorKey(int iPort, int keyMode, int nSector,
StringBuilder oldKey, StringBuilder newKey);
```

2. Read ID card information

```
private void btnGetIDCard_Click(object sender, EventArgs e)
{
    int AutoSearchReader = IDCard.InitCommExt();
    if (AutoSearchReader > 0)
    {
```

```
        //Port = AutoSearchReader;
        //IsConnected = true;
        //lblName.Text = AutoSearchReader.ToString();
        StringBuilder sb = new StringBuilder(cbDataSize);
        IDCard.GetSAMID(sb);
        //MessageBox.Show ("ID card reader connected successfully, SAM module number:"+sb);

        //Card authentication
        int FindCard = IDCard.Authenticate();

        //read the card
        int rs = IDCard.Read_Content(1);
        //MessageBox.Show("Read_Content:" + rs);
        if (rs != 1 && rs != 2 && rs != 3)
        {
            ClearData();
            return;
        }

        //Card reading succeeded.
        //Name
        //StringBuilder sb = new StringBuilder(cbDataSize);
        IDCard.getName(sb, cbDataSize);
        lblName.Text = sb.ToString();

        //Nation/country
        IDCard.getNation(sb, cbDataSize);
        lblNation.Text = sb.ToString();

        //Gender
        IDCard.getSex(sb, cbDataSize);
        lblSex.Text = sb.ToString();

        //Born
        IDCard.getBirthdate(sb, cbDataSize);
```

```
        string birthdate = sb.ToString();
        DateTime dt = Convert.ToDateTime(birthdate.Substring(0, 4) + "-" + birthdate.Substring(4, 2) + "-" + birthdate.Substring(6, 2));
        lblBirthYear.Text = dt.Year.ToString();
        lblBirthMonth.Text = dt.Month.ToString();
        lblBirthDay.Text = dt.Day.ToString();

        //address
        IDCard.getAddress(sb, cbDataSize);
        string ad = sb.ToString();
        lblAdd.Text = ad;

        //number
        IDCard.getIDNum(sb, cbDataSize);
        lblIDNum.Text = sb.ToString();

        //organ
        IDCard.getIssue(sb, cbDataSize);
        lblIssue.Text = sb.ToString();

        //Validity period
        IDCard.getEffectedDate(sb, cbDataSize);
        DateTime dt1 = Convert.ToDateTime(sb.ToString().Substring(0, 4) + "-" + sb.ToString().Substring(4, 2) + "-" + sb.ToString().Substring(6, 2));
        lblEffectedDate.Text = string.Format("{0:yyyy.MM.dd}", dt1);
        IDCard.getExpiredDate(sb, cbDataSize);
        DateTime dt2 = Convert.ToDateTime(sb.ToString().Substring(0, 4) + "-" + sb.ToString().Substring(4, 2) + "-" + sb.ToString().Substring(6, 2));
        lblExpiredDate.Text = string.Format("{0:yyyy.MM.dd}", dt2);

        //Show avatar
        IDCard.GetBmpPhotoExt();
        int cbPhoto = 256 * 1024;
        StringBuilder sbPhoto = new StringBuilder(cbPhoto);
        int nRet = IDCard.getBMPPhotoBase64(sbPhoto, cbPhoto);
        byte[] byPhoto = Convert.FromBase64String(sbPhoto.ToString());
```

```
        if (nRet == 1)
        {
            MemoryStream ms = new MemoryStream(byPhoto);
            Image a = Image.FromStream(ms);
            Bitmap srcbmp1 = new Bitmap(a);
            a.Dispose();
            srcbmp1.MakeTransparent(Color.FromArgb(255, 255, 255));
            pbIDPhoto.BackgroundImage = srcbmp1;
        }

        int retc = IDCard.CloseComm();
    }
    else
    {
        MessageBox.Show ("Check whether the equipment is connected correctly", "Ugee
Hyss001demo", MessageboxButtons.ok, Messageboxicon.information);
    }
}
```