**ISO14443权限的分析与生成**

**姓名:马俊 学号:165013105 班级:1650442**

一、实验目的：

（1）进一步掌握用户控件的使用。

（2）进一步掌握窗体与用户控件间的传值方法。

（3）进一步学习ListView控件的使用。

（4）掌握数据转换方法。

（5）理解存取控制权限的分析与生成过程。

二、实验内容：

（1）创建存取控制权限分析与生成用户控件。

（2）窗体与用户控件间的传值方法。

（3）ListView控件的使用及数据转换方法。

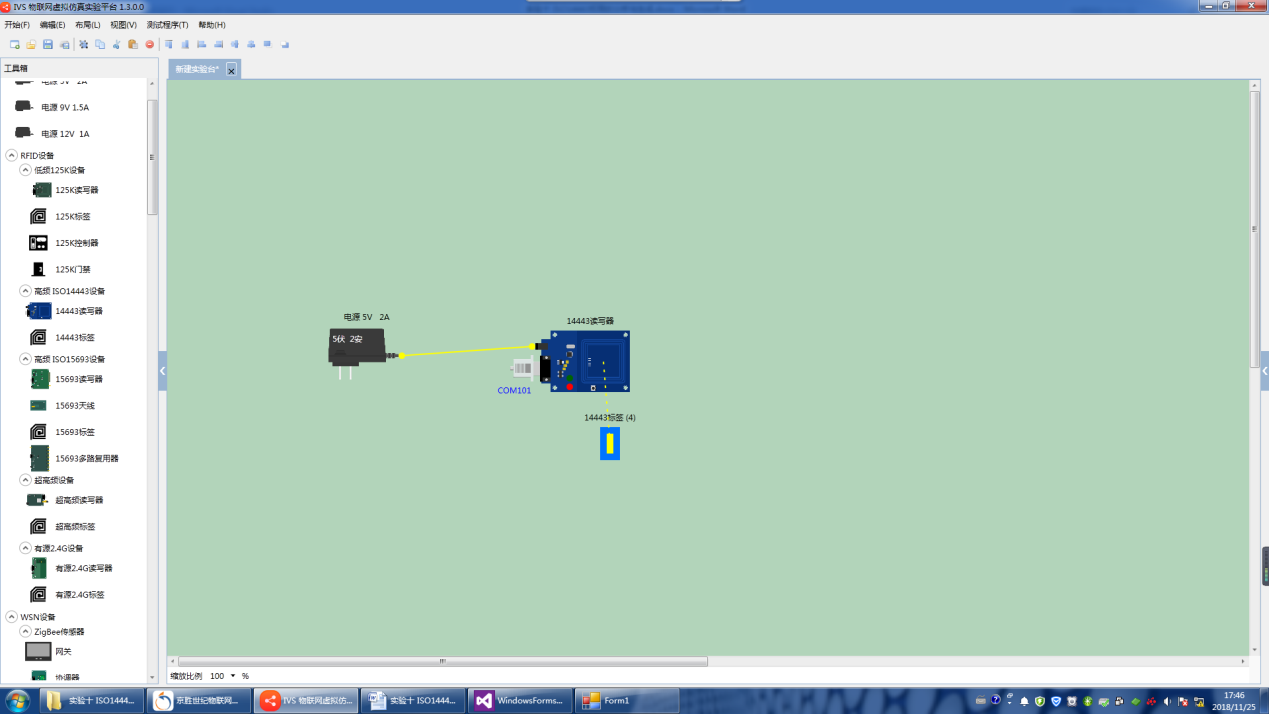
（4）完成存取控制权限的分析。

三、实验设备：

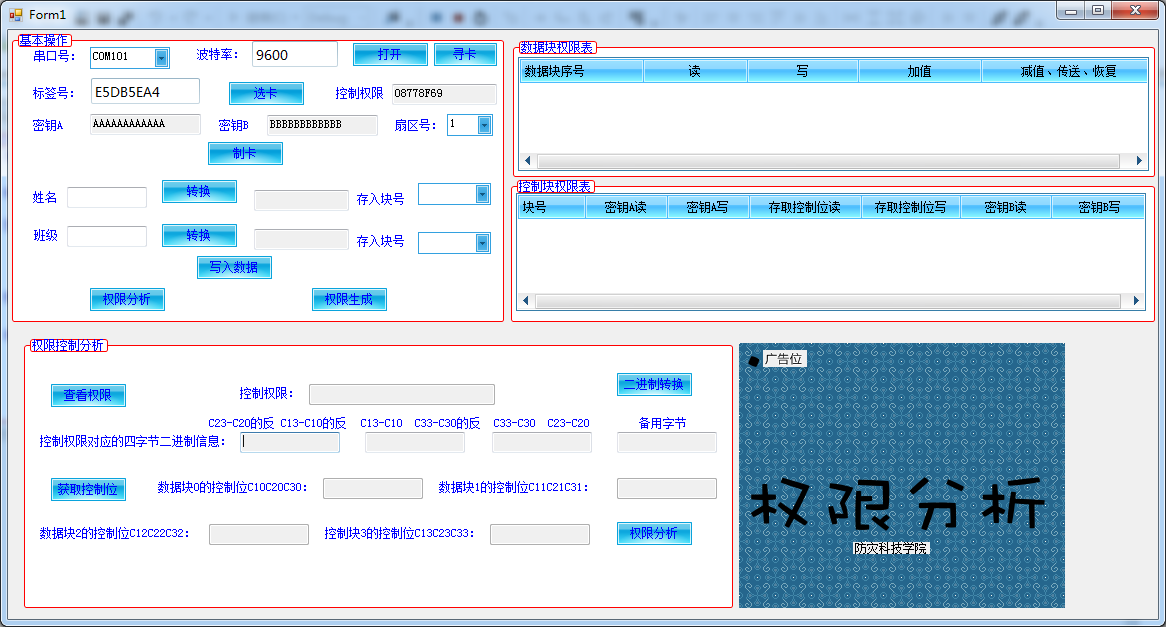
ISO14443读写器、串口线、5V，2A电源、ISO14443卡片

四、实验步骤：

1.在虚拟仿真实验平台上，搭建ISO14443硬件原理图，并截图放入实验报告中。



**硬件原理图**



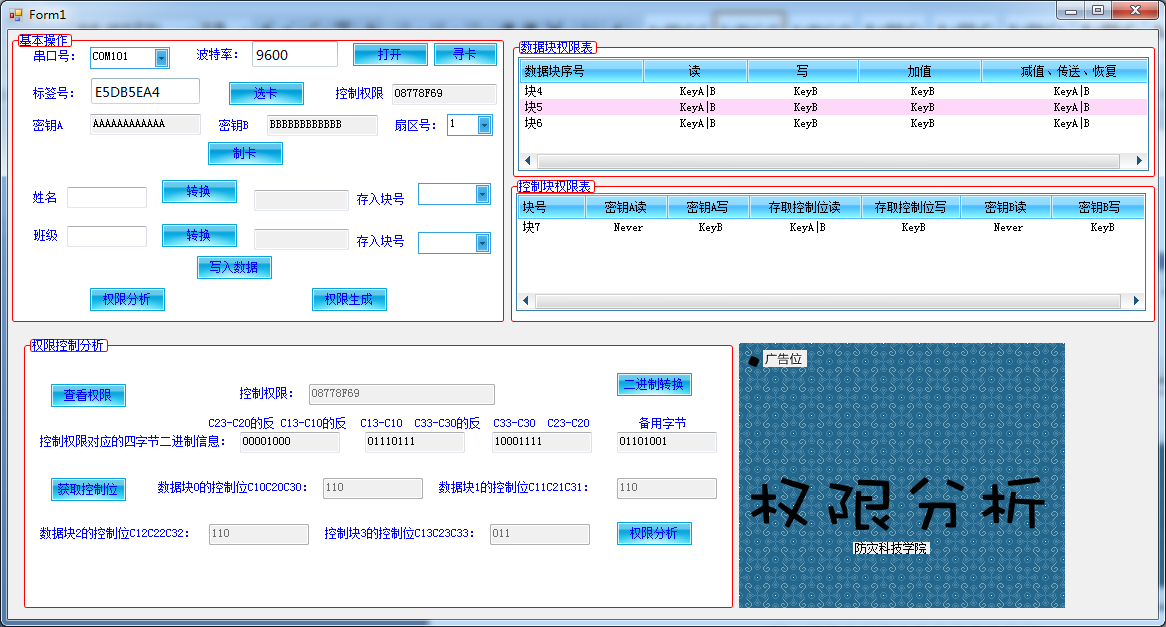
**图1 主窗体的运行效果**

一、主窗体应用包括打开串口，寻卡，选卡，制卡操作，选择扇区号，将预制好的密钥A、B,控制权限写进卡里

二、写卡操作，可以将姓名和班级转为二进制字符串存入对应的块内，块号由所选扇区号不同而变。



**图2卡片初始时的数据**

**图3权限分析用户控件的设计界面**

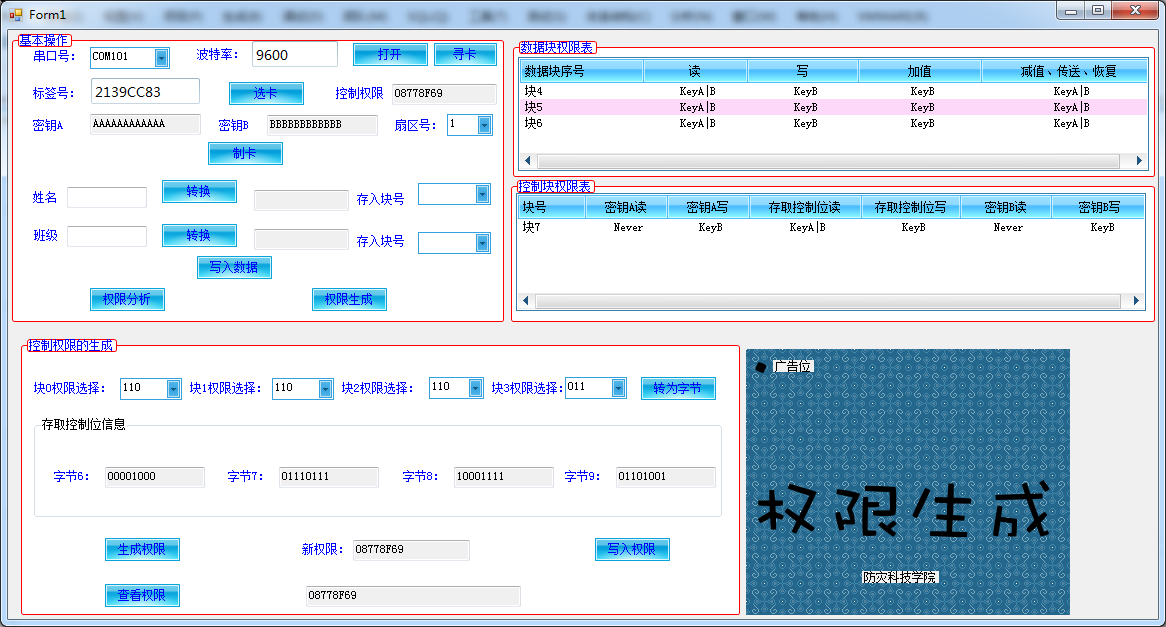


**图4制卡后的数据**

1、点击权限分析后，将用户窗体加载到主窗体。

2、点击查看权限，通过验证密钥B并从第8位截取8位，得到控制权限，将控制权限二进制转换，得到四个字节二进制信息，送入四个文本框。通过截取，得到块0-块3的控制位送到对应文本框内。

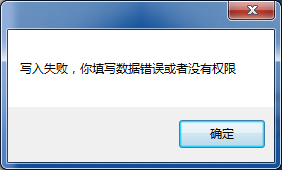
3、点击权限分析，将类RightDisp中对应的字符串数据传递进来，将卡权限信息和控制块权限表显示在两个文本框中。

****

**图5权限生成用户控件的设计界面**

****

**图6权限写入后的数据**

****

**图7 密钥B不可写时写入失败**

****

**图8成功写入块4块5数据**

1、将类RightDisp中对应的字符串数据传递进来，则可以选择块0-块3，点击转为字节后，对照表格进行截取和取反连结操作，得到字节6-字节9。

2、点击生成权限，现将四个文本框的字符串连接成32位二进制字符串，转为十六进制传入到文本框。

3、点击写入权限，在验证密钥B后，利用writekey写入。点击查看权限，利用readdata并截取得到新权限进行验证。

|  |
| --- |
| **实验完成的程序及抓图效果：**  **一、**主窗体程序  using IES\_ISO14443\_Share;  using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.IO.Ports;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  namespace WindowsFormsApplication5  {  public partial class Form1 : Form  {  public Form1()  {  InitializeComponent();  }  private void splitContainer1\_Panel2\_Paint(object sender, PaintEventArgs e)  {  }  /\*  \* 打开串口操作  \*/  private void btnOpenPort\_Click(object sender, EventArgs e)  {  if (!ISO14443\_Tag.Is\_Port)  {  try  {  if (ISO14443\_Tag.OpenPorts(cmbPorts.Text.Trim()) == "")  {  ISO14443\_Tag.Is\_Port = true;  MessageBox.Show("串口打开成功！");  }  else  {  ISO14443\_Tag.Is\_Port = false;  MessageBox.Show("串口打开失败！");  }  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }  }  else  {  MessageBox.Show("串口已处于打开状态，无需再次打卡！");  }  }  /\*  \* 主窗体加载给窗体送值  \*/  private void Form1\_Load(object sender, EventArgs e)  {  string[] ports = SerialPort.GetPortNames();//从系统得到串口号  if (ports.Length > 00)  {  cmbPorts.Items.AddRange(ports);//将得到的串口号送给cmbPort的项目里  cmbPorts.SelectedIndex = 0;//默认选择第一个  txtBaud.Text = "9600";  }  else  {  MessageBox.Show("系统中没有可用串口！");  }  for (int i = 0; i < 16; i++)  {  cmbSectionID.Items.Add(i.ToString());  }  cmbSectionID.SelectedIndex = 0;  }  /\*  \* 寻卡操作  \*/  private void btnFindCard\_Click(object sender, EventArgs e)  {  if (!ISO14443\_Tag.Is\_Port)  {  MessageBox.Show("寻卡前必须打开串口！");  return;  }  try  {  string strCardID = ISO14443\_Tag.ReadTag();//得到卡的标签号  if (strCardID != "")  {  txtCardID.Text = strCardID;  //ISO14443\_Tag.Tag = strCardID;  ISO14443\_Tag.Is\_Find = true;  }  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }  }  /\*  \* 制卡操作，密钥A为AAAAAAAAAAAA，密钥B为BBBBBBBBBBBB，控制权限为08778F69  \* \*/  private void btnSelectCard\_Click(object sender, EventArgs e)  {  if (!ISO14443\_Tag.Is\_Find)  {  MessageBox.Show("选卡前必须先寻卡！");  return;  }  try  {  if (ISO14443\_Tag.TagSelect(txtCardID.Text.Trim()))  {  ISO14443\_Tag.Is\_Select = true;  txtAKey.Text = "AAAAAAAAAAAA";  txtBKey.Text = "BBBBBBBBBBBB";  txtControl.Text = "08778F69";  MessageBox.Show("选卡成功！");  }  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }  }  /\*  \* 定义listviewadd，三个参数为（listview名字，块号，定义包含字符串数组）  \*/  private void ListViewAdd(ListView lst, int blockID, string[] items)  {  int i;  ListViewItem lv = new ListViewItem();  lv.Text = "块" + blockID.ToString();  for (i = 0; i < items.Length; i++)  {  lv.SubItems.Add(items[i]);  }  lst.Items.Add(lv);  }  private void uc\_EventPowerAnalasys(string[] BlockControl)  {  //根据控制位信息在listview控件中显示  int k;  int blockID;  string[] str = { };  for (k = 0; k < 3; k++)  {  blockID = Convert.ToInt32(cmbSectionID.Text.Trim()) \* 4 + k;  str = RightDisp.DispDataRight(BlockControl[k]);//自定义类RightDisp中的静态方法，用于显示数据块控制位的具体信息  ListViewAdd(listView1, blockID, str);//用于将获取到的卡权限信息显示到列表框1中  }  blockID = Convert.ToInt32(cmbSectionID.Text) \* 4 + 3;  str = RightDisp.DispControlRight(BlockControl[3]); //自定义类RightDisp中的静态方法，用于显示控制块控制位的具体信息  ListViewAdd(listView2, blockID, str);//将获取到的控制信息显示到列表框2中  }  private void btnPowerAnalasys\_Click(object sender, EventArgs e)  {  listView1.Items.Clear();//清空两个数据框的项目  listView2.Items.Clear();  UserControl2 uc = new UserControl2(txtCardID.Text.Trim(), cmbSectionID.Text.Trim());//将卡号，选择的扇区号传到用户控件  splitContainer1.Panel2.Controls.Clear();  splitContainer1.Panel2.Controls.Add(uc);//显示用户控件  uc.EventPowerAnalasys += uc\_EventPowerAnalasys;  }  private void uc\_EventListViewUpdate(int blockID, string[] items)  {  bool flag = true;//判别当前列表中是否存在该块号的数据，如果没有则为true,如果存在则为false  string key = "块¨¦" + blockID.ToString();//块1、块2....的显示  for (int i = 0; i < listView1.Items.Count; i++)  {  if (listView1.Items[i].Text == key)  {  for (int j = 1; j <= items.Length; j++)  {  listView1.Items[i].SubItems[j].Text = items[j - 1];//更新当前项数据  }  flag = false;  break;  }  }  if (flag)  {  int BlockID = Convert.ToInt32(cmbSectionID.Text) \* 4;  if (blockID == BlockID || blockID == BlockID+1 || blockID == BlockID+2)//判断如果是前三个块，则为卡数据块，添加进列表框1，否则为控制块，添加到列表2  {  ListViewAdd(listView1, blockID, items);  }  else  {  ListViewAdd(listView2,blockID,items);  }  }  }  private void btnPowerCreate\_Click(object sender, EventArgs e)  {  listView1.Items.Clear();  listView2.Items.Clear();  UserControl1 uc = new UserControl1(txtAKey.Text.Trim(),txtBKey.Text.Trim(),txtCardID.Text.Trim(), cmbSectionID.Text.Trim());//将制卡的密钥AB，卡号，扇区号传入用户控件  splitContainer1.Panel2.Controls.Clear();  splitContainer1.Panel2.Controls.Add(uc);  uc.EventListViewUpdate += uc\_EventListViewUpdate;  }  private void textBox3\_TextChanged(object sender, EventArgs e)  {  }  private void button1\_Click(object sender, EventArgs e)  {  try  {  if (ISO14443\_Tag.Is\_Select)  {  if (ISO14443\_Tag.KeyB((Convert.ToInt32(cmbSectionID.Text) \* 4 + 3).ToString(), "FFFFFFFFFFFF", txtCardID.Text.Trim()) == "")//验证密钥B  {  // if (ISO14443\_Tag.WriteData((Convert.ToInt32(cmbSectionID.Text) \* 4 + 3).ToString(), Carry.ChsToHex(textBox1.Text.Trim())) == "")  // {  if (ISO14443\_Tag.WriteKey(txtAKey.Text.Trim(), txtBKey.Text.Trim(), txtControl.Text.Trim(), (Convert.ToInt32(cmbSectionID.Text) \* 4 + 3).ToString()) == "")//写入控制块数据  {  MessageBox.Show("制卡成功！");  }  // }  }  }  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }    }    private void groupBox1\_Enter(object sender, EventArgs e)  {  }  private void button2\_Click(object sender, EventArgs e)  {  textBox1.Text = Carry.ChsToHex(textBox3.Text);//将输入的姓名转为十六进制  }  private void grpCreateCard\_Enter(object sender, EventArgs e)  {  }  private void cmbSectionID\_SelectedIndexChanged(object sender, EventArgs e)  {  cmbBlock1.Items.Clear();  cmbBlock2.Items.Clear();  for (int i = Convert.ToInt32(cmbSectionID.Text)\*4; i < Convert.ToInt32(cmbSectionID.Text)\*4 + 3; i++)//选择想写入的块区号，由所选扇区号计算，前三个块可以写数据  {      cmbBlock1.Items.Add(i.ToString());  cmbBlock2.Items.Add(i.ToString());  }  cmbBlock1.SelectedIndex = 0;  cmbBlock2.SelectedIndex = 0;  }  private void button4\_Click(object sender, EventArgs e)  {  try  {  if (ISO14443\_Tag.Is\_Select)  {  if (ISO14443\_Tag.KeyB((Convert.ToInt32(cmbSectionID.Text) \* 4 + 3).ToString(), "BBBBBBBBBBBB", txtCardID.Text.Trim()) == "")//验证密钥B  {  if (ISO14443\_Tag.WriteData( cmbBlock1.Text.Trim(),textBox1.Text.Trim()) == "")//写数据操作  {  if (ISO14443\_Tag.WriteData(cmbBlock2.Text.Trim(),textBox4.Text.Trim() ) == "")  {  MessageBox.Show("写入数据成功！");  }  }  }  }  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }  }  private void button3\_Click(object sender, EventArgs e)  {  string str = string.Format("{0:X2}",Convert.ToInt32(textBox2.Text.Trim()));//将班级号转为二进制再转为字符串类型  if (str.Length <= 32)//补全32位  {  int result = str.Length;  for (int i = 0; i < 32 - result; i++)  {  str = "0" + str;  }  }  textBox4.Text = str;  }  }  }  **二、权限生成用户控件程序**  using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Drawing;  using System.Data;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using IES\_ISO14443\_Share;  namespace WindowsFormsApplication5  {  public partial class UserControl2 : UserControl  {  public delegate void DelegatePowerAnalasys(string[] BlockControl);//定义委托，参数为块号字符串组  public event DelegatePowerAnalasys EventPowerAnalasys;//定义事件  private string CardID;  private string SectionID;  public UserControl2(string cardID,string sectionID)//从主体获取卡标签号，所选扇区号  {  InitializeComponent();  this.CardID = cardID;  this.SectionID = sectionID;    }  private void UserControl2\_Load(object sender, EventArgs e)  {  }  private void btnViewControl\_Click(object sender, EventArgs e)  {  string strBlockID=(Convert.ToInt32(SectionID) \* 4 + 3).ToString();//通过扇区号求得块号  try  {  if(ISO14443\_Tag.KeyB(strBlockID,"BBBBBBBBBBBB",CardID)=="")//验证秘钥B  {    string strPower = ISO14443\_Tag.ReadData(strBlockID);//读取控制权限所在的块数据  txtPower.Text = strPower.Substring(12, 8);//截取控制权限  }  }  catch(Exception ex)  {  MessageBox.Show(ex.Message);  }  }  /\*private string[] HexToBin(string HexStr)  {  string[] Binstrs = new string[4]; //用于存放转换后的四字节二进制串  byte[] bytes = new byte[4];  for (int i = 0; i < 4; i++)  {  bytes[i] = Convert.ToByte(HexStr.Substring(i \* 2, 2), 16);  }  for (int i = bytes.Length-1; i>=0; i--)  {  Binstrs[3-i] = Convert.ToString(bytes[i], 2);  int len = Binstrs[3-i].Length;  if (len< 8)  {  for (int j = 0; j < 8 - len; j++)  {  Binstrs[3-i] = "0" + Binstrs[3-i];  }  }  }  return Binstrs;  }  \*/  /\*  \* 定义一个函数，参数为字符串，可将字符串转为二进制  \*/  private string[] HexToBin(string HexStr) //ff078069  {  string[] Binstrs = new string[4]; //用于存放转换后的四字节二进制串  Int32 Decdata = Convert.ToInt32(HexStr, 16);//将16进制串转为32位的整数  byte[] bytes = BitConverter.GetBytes(Decdata);//将整型数据转换为字节数组,0下标存数据的低字节  for (int i = 0; i <bytes.Length; i++)  {  Binstrs[i] = Convert.ToString(bytes[i], 2);  int len = Binstrs[i].Length;  if (len< 8)  {  for (int j = 0; j < 8 - len; j++)  {  Binstrs[i] = "0" + Binstrs[i];  }  }  }  return Binstrs;  }    private void btnHexToBin\_Click(object sender, EventArgs e)  {    string[] Binstrs=HexToBin(txtPower.Text);//调用函数，送入txtpower文本框的值，再将得到的值送入四个文本框  txtBinary9.Text = Binstrs[0];  txtBinary8.Text = Binstrs[1];  txtBinary7.Text = Binstrs[2];  txtBinary6.Text = Binstrs[3];  }  /\*  \* 参照表截取相应的值，送入到块0-块3的控制位  \*/  private void btnGetControlBit\_Click(object sender, EventArgs e)  {  txtData0.Text = txtBinary7.Text.Substring(3, 1) + txtBinary8.Text.Substring(7, 1) + txtBinary8.Text.Substring(3, 1);  txtData1.Text = txtBinary7.Text.Substring(2, 1) + txtBinary8.Text.Substring(6, 1) + txtBinary8.Text.Substring(2, 1);  txtData2.Text = txtBinary7.Text.Substring(1, 1) + txtBinary8.Text.Substring(5, 1) + txtBinary8.Text.Substring(1, 1);  txtData3.Text = txtBinary7.Text.Substring(0, 1) + txtBinary8.Text.Substring(4, 1) + txtBinary8.Text.Substring(0, 1);  }  /\*  \* 分析权限  \*/  private void btnAnalasys\_Click(object sender, EventArgs e)  {  string[] blockControl=new string[4];//定义一个字符串数组  blockControl[0] = txtData0.Text.Trim();//将四个文本框的字符串存入数组  blockControl[1] = txtData1.Text.Trim();  blockControl[2] = txtData2.Text.Trim();  blockControl[3] = txtData3.Text.Trim();  if (EventPowerAnalasys != null)  {    // lstData.Items.Clear();  // lstPower.Items.Clear();  EventPowerAnalasys(blockControl);//将用户窗体的字符串数组传送到主窗体  }    }  }  }  **三、权限生成用户控件程序**  using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Drawing;  using System.Data;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  using IES\_ISO14443\_Share;  namespace WindowsFormsApplication5  {  public partial class UserControl1 : UserControl  {  public delegate void DelegateListViewUpdate(int blockID, string[] items);//委托，参数为块号和字符串组  public event DelegateListViewUpdate EventListViewUpdate;//定义委托时间  private string KeyA;  private string KeyB;  private string CardID;  private string SectionID;  /\*  \* 将主窗体的数据keyA，keyB，卡号和扇区号传递进来  \*/  public UserControl1(string keyA,string keyB,string cardID, string sectionID)  {  InitializeComponent();  this.KeyA = keyA;  this.KeyB = keyB;  this.CardID = cardID;  this.SectionID = sectionID;  }  private void UserControl1\_Load(object sender, EventArgs e)  {  }  private void cmbBlock0\_SelectedIndexChanged(object sender, EventArgs e)  {  string[] strBlock0Items = RightDisp.DispDataRight(cmbBlock0.Text);//将类RightDisp中对应的字符串数据传递进来  int blockID = Convert.ToInt32(SectionID) \* 4 + 0;//计算该扇区第0块地址    if (EventListViewUpdate != null)  {  EventListViewUpdate(blockID, strBlock0Items);//将用户控件里的块号和类里的字符串数组传递到主窗体  }  }    private void cmbBlock1\_SelectedIndexChanged(object sender, EventArgs e)  {  string[] strBlock0Items = RightDisp.DispDataRight(cmbBlock1.Text);  int blockID = Convert.ToInt32(SectionID) \* 4 + 1;//计算该扇区第1块地址    if (EventListViewUpdate != null)  {  EventListViewUpdate( blockID, strBlock0Items);  }  }  private void cmbBlock2\_SelectedIndexChanged(object sender, EventArgs e)  {  string[] strBlock0Items = RightDisp.DispDataRight(cmbBlock2.Text);  int blockID = Convert.ToInt32(SectionID) \* 4 + 2;//计算该扇区第2块地址    if (EventListViewUpdate != null)  {  EventListViewUpdate( blockID, strBlock0Items);  }  }  private void cmbBlock3\_SelectedIndexChanged(object sender, EventArgs e)  {  string[] strBlock0Items = RightDisp.DispControlRight(cmbBlock3.Text);  int blockID = Convert.ToInt32(SectionID) \* 4 + 3;//计算该扇区第3块地址    if (EventListViewUpdate != null)  {  EventListViewUpdate( blockID, strBlock0Items);  }  }  private void btnTransByte\_Click(object sender, EventArgs e)  {  /\* string[] strTemp = { cmbBlock3.Text.Substring(0, 1) + cmbBlock2.Text.Substring(0, 1)+ cmbBlock1.Text.Substring(0, 1) + cmbBlock0.Text.Substring(0, 1),  cmbBlock3.Text.Substring(1, 1) + cmbBlock2.Text.Substring(1, 1) + cmbBlock1.Text.Substring(1, 1) + cmbBlock0.Text.Substring(1, 1),  cmbBlock3.Text.Substring(2, 1) + cmbBlock2.Text.Substring(2, 1) + cmbBlock1.Text.Substring(2, 1) + cmbBlock0.Text.Substring(2, 1) };  string[] strByte = { strTemp[1] + strTemp[0], strTemp[0] + strTemp[2], strTemp[2] + strTemp[1] }; \*/  string[] strRight=new string[4];  string[] strTemp = new string[3];  string[] strByte = new string[3];  strRight[0] = cmbBlock0.Text;  strRight[1] = cmbBlock1.Text;  strRight[2] = cmbBlock2.Text;  strRight[3] = cmbBlock3.Text;  strTemp[0] = cmbBlock3.Text.Substring(0, 1) + cmbBlock2.Text.Substring(0, 1) + cmbBlock1.Text.Substring(0, 1) + cmbBlock0.Text.Substring(0, 1);//对照图分析，将需要的字符截取出来并连接  strTemp[1] = cmbBlock3.Text.Substring(1, 1) + cmbBlock2.Text.Substring(1, 1) + cmbBlock1.Text.Substring(1, 1) + cmbBlock0.Text.Substring(1, 1);  strTemp[2] = cmbBlock3.Text.Substring(2, 1) + cmbBlock2.Text.Substring(2, 1) + cmbBlock1.Text.Substring(2, 1) + cmbBlock0.Text.Substring(2, 1);  /\*  \* 对照图，得到字节6、7、8  \*/  strByte[0] = strTemp[1] + strTemp[0];  strByte[1] = strTemp[0] + strTemp[2];  strByte[2] = strTemp[2] + strTemp[1];    byte[] bytes = new byte[3];//定义byte数组  for (int i = 0; i < 3; i++)  {  bytes[i] = Convert.ToByte(strByte[i], 2);//将二进制字符串转为byte  }  bytes[0] = (byte)(bytes[0] ^ 0xff);//根据权限对照表，取反  bytes[1] = (byte)(bytes[1] ^ 0x0f);//根据权限对照表，后四位取反    for (int i = 0; i < 3; i++)  {  strByte[i] = Convert.ToString(bytes[i], 2);//byte转字符串  int len = strByte[i].Length;  if (len < 8)//如果小于8位，前面加“0”  {  for (int j = 0; j < 8 - len; j++)  {  strByte[i] = "0" + strByte[i];  }  }  }  txtByte6.Text = strByte[0];  txtByte7.Text = strByte[1];  txtByte8.Text = strByte[2];  txtByte9.Text = "01101001";//最后备用字节永远为69（01101001）        }    private void btnCreateRight\_Click(object sender, EventArgs e)  {    string str1=txtByte6.Text.Trim()+txtByte7.Text.Trim()+txtByte8.Text.Trim()+txtByte9.Text.Trim();//将四个框里的二进制字符串连成一个  string str= string.Format("{0:X}", System.Convert.ToInt32(str1, 2));//字符串转为十六进制    if (str.Length <= 8)//如果不够8位，补全8位  {  int result = str.Length;  for (int i = 0; i < 8 - result; i++)  {  str = "0" + str;  }  }  txtNewRight.Text = str.ToUpper();//把str转为大写送入文本框  }  private void btnCreateCard\_Click(object sender, EventArgs e)  {  string strBlockID = (Convert.ToInt32(SectionID) \* 4 + 3).ToString();//由扇区号得到块号  try  {  if (ISO14443\_Tag.Is\_Select)  {  if (ISO14443\_Tag.KeyB(strBlockID , "BBBBBBBBBBBB", CardID) == "")//验证B密钥  {  if (ISO14443\_Tag.WriteData(strBlockID, KeyA + txtNewRight.Text.Trim() + KeyB) == "")//写入权限  {  MessageBox.Show("写入成功！");  }  }  }  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }  }  private void groupBox1\_Enter(object sender, EventArgs e)  {  }  private void button1\_Click(object sender, EventArgs e)  {  string strBlockID = (Convert.ToInt32(SectionID) \* 4 + 3).ToString();//通过扇区号计算块号  try  {  if (ISO14443\_Tag.KeyB(strBlockID, "BBBBBBBBBBBB", CardID) == "")//验证秘钥B  {  string strPower = ISO14443\_Tag.ReadData(strBlockID).Substring(12,8);//读取控制权限  textBox1.Text = strPower;  }  }  catch (Exception ex)  {  MessageBox.Show(ex.Message);  }  }  }  }  **四、定义类RightDisp**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace WindowsFormsApplication5  {  class RightDisp  {  public static string[] DispDataRight(string strData)  {  string[] str = { };  switch (strData)  {  case"000": str = new string[] { "KeyA|B", "KeyA|B", "KeyA|B", "KeyA|B" }; break;  case"010": str = new string[] { "KeyA|B", "Never", "Never", "Never" }; break;  case"100": str = new string[] { "KeyA|B", "KeyB", "Never", "Never" }; break;  case"110": str = new string[] { "KeyA|B", "KeyB", "KeyB", "KeyA|B" }; break;  case"001": str = new string[] { "KeyA|B", "Never", "Never", "KeyA|B" }; break;  case"011": str = new string[] { "KeyB", "KeyB", "Never", "Never" }; break;  case"101": str = new string[] { "KeyB", "Never", "Never", "Never" }; break;  case"111": str = new string[] { "Never", "Never", "Never", "Never" }; break;  }  return str;  }  public static string[] DispControlRight(string strControl)  {  string[] str = { };  switch (strControl)  {  case"000": str = new string[] { "Never", "KeyA|B", "KeyA|B", "Never", "KeyA|B", "KeyA|B" }; break;  case"010": str = new string[] { "Never", "Never", "KeyA|B", "Never", "KeyA|B", "Never" }; break;  case"100": str = new string[] { "Never", "KeyB", "KeyA|B", "Never", "Never", "KeyB" }; break;  case"110": str = new string[] { "Never", "Never", "KeyA|B", "Never", "Never", "Never" }; break;  case"001": str = new string[] { "Never", "KeyA|B", "KeyA|B", "KeyA|B", "KeyA|B", "KeyA|B" }; break;  case"011": str = new string[] { "Never", "KeyB", "KeyA|B", "KeyB", "Never", "KeyB" }; break;  case"101": str = new string[] { "Never", "Never", "KeyA|B", "KeyB", "Never", "Never" }; break;  case"111": str = new string[] { "Never", "Never", "KeyA|B", "Never", "Never", "Never" }; break;  }  return str;  }  }  } |
| **实验心得体会：** |