SkControlLock控制模块PC端开发指南-C#

1. 快速上手

using SkLockApi.Device;

using SkLockApi.Device.Enum;

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.Windows.Forms;

namespace SkLockApiTest

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private SkLock skLock;

private void Form1\_Load(object sender, EventArgs e)

{

skLock = new SkLock();

skLock.IsSync = false;

skLock.Open("COM4");

skLock.OnEquipmentFeedbackHandler += EquipmentFeedbackHandler;

}

private void btnGetEdition\_Click(object sender, EventArgs e)

{

//设置同步

//skLock.IsSync = true;

//byte addr = Convert.ToByte(comboBox1.Text);

//string ver = string.Empty;

//bool flag = skLock.GetControlEdition(addr, ref ver);

//if (flag)

//{

// Console.WriteLine("设备版本号：" + ver);

//}

//skLock.IsSync = false;

//默认异步发送

byte addr = Convert.ToByte(comboBox1.Text);

skLock.GetControlEdition(addr);

}

RadioButton radioButton = null;

RadioButton rbtnOnOff = null;

private void btnGetLock\_Click(object sender, EventArgs e)

{

//默认异步发送

byte addr = Convert.ToByte(comboBox1.Text);

if (rbtnLock1.Checked)

{

radioButton = rbtnLock1;

skLock.GetLockStatus(addr, EnumDeviceSequence.Lock1);

}

else if (rbtnLock2.Checked)

{

radioButton = rbtnLock2;

skLock.GetLockStatus(addr, EnumDeviceSequence.Lock2);

}

}

private void btnSetLed\_Click(object sender, EventArgs e)

{

//默认异步发送

byte addr = Convert.ToByte(comboBox1.Text);

byte status = 0;

if (rbtnON.Checked)

{

rbtnOnOff = rbtnON;

status = 1;

}

else if (rbtnOFF.Checked)

{

rbtnOnOff = rbtnOFF;

status = 2;

}

if (rbtnLed1.Checked)

{

radioButton = rbtnLed1;

skLock.SetLedStatus(addr, EnumDeviceSequence.Led1, status);

}

if (rbtnLed2.Checked)

{

radioButton = rbtnLed2;

skLock.SetLedStatus(addr, EnumDeviceSequence.Led2, status);

}

if (rbtnLed3.Checked)

{

radioButton = rbtnLed3;

skLock.SetLedStatus(addr, EnumDeviceSequence.Led3, status);

}

if (rbtnLed4.Checked)

{

radioButton = rbtnLed4;

skLock.SetLedStatus(addr, EnumDeviceSequence.Led4, status);

}

}

private void btnSetLock\_Click(object sender, EventArgs e)

{

//默认异步发送

byte addr = Convert.ToByte(comboBox1.Text);

if (rbtnLock1.Checked)

{

radioButton = rbtnLock1;

skLock.SetLockStatus(addr, EnumDeviceSequence.Lock1);

}

else if (rbtnLock2.Checked)

{

radioButton = rbtnLock2;

skLock.SetLockStatus(addr, EnumDeviceSequence.Lock2);

}

}

private void btnSetAddr\_Click(object sender, EventArgs e)

{

//默认异步发送

byte addr = Convert.ToByte(comboBox1.Text);

byte[] newAddr = new byte[1];

newAddr[0] = Convert.ToByte(comboBox2.Text);

Console.WriteLine(newAddr[0].ToString("X2"));

skLock.SetControlAddr(addr, newAddr);

}

/// <summary>

/// 异步信息反馈事件

/// </summary>

/// <param name="cmd">命令类型</param>

/// <param name="data\_bytes">数据帧中data数据</param>

public void EquipmentFeedbackHandler(byte cmd, byte[] data\_bytes)

{

switch (cmd)

{

case 0xC1:

//设置控制板地址

textBoxShow.BeginInvoke((MethodInvoker)delegate

{

textBoxShow.AppendText("控制板地址由" + comboBox1.Text +

"更变为" + data\_bytes[0].ToString("X2") + "\r\n");

});

break;

case 0xC2:

//设置电磁锁状态

textBoxShow.BeginInvoke((MethodInvoker)delegate

{

textBoxShow.AppendText(radioButton.Text + "电磁锁" +

data\_bytes[0].ToString("X2") + "驱动成功" + "\r\n");

});

break;

case 0xC3:

//查询电磁锁状态

textBoxShow.BeginInvoke((MethodInvoker)delegate

{

if (data\_bytes[1] == 0x01)

textBoxShow.AppendText("电磁锁" + data\_bytes[0] + "处于开锁状态" + "\r\n");

else if (data\_bytes[1] == 0x02)

textBoxShow.AppendText("电磁锁" + data\_bytes[0] + "处于关锁状态" + "\r\n");

});

break;

case 0xC4:

//设置LED灯状态

textBoxShow.BeginInvoke((MethodInvoker)delegate

{

if (data\_bytes[0] == 0x01)

textBoxShow.AppendText(radioButton.Text + "灯" +

rbtnOnOff.Text + "状态设置成功" + "\r\n");

else

textBoxShow.AppendText(radioButton.Text + "灯:" +

rbtnOnOff.Text + "状态设置失败" + "\r\n");

});

break;

case 0xC5:

//查询设备版本信息

textBoxShow.BeginInvoke((MethodInvoker)delegate

{

textBoxShow.AppendText("设备版本号：V:"

+ data\_bytes[0].ToString("X2") + "." + data\_bytes[1].ToString("X2") + "\r\n");

});

break;

case 0xF1:

//控制板地址匹配错误反馈

textBoxShow.BeginInvoke((MethodInvoker)delegate

{

textBoxShow.AppendText("控制板地址匹配错误反馈" + "\r\n");

});

break;

case 0xF2:

//数据帧校验匹配失败反馈

textBoxShow.BeginInvoke((MethodInvoker)delegate

{

textBoxShow.AppendText("数据帧校验匹配失败反馈" + "\r\n");

});

break;

default:

break;

}

}

private void btnClear\_Click(object sender, EventArgs e)

{

textBoxShow.Text = "";

}

}

}

1. 接口说明
   1. 链接串口

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public bool Open(string portName) |
| 说明 | portName：需要连接的串口号，默认参数115200，8，1，none  返回值：true，链接成功；false，链接失败 |

* 1. 关闭串口

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public bool Close() |
| 说明 | 返回值：true，关闭成功；false，关闭失败 |

* 1. 设置控制板地址（异步模式）

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public void SetControlAddr(byte sourceAddr, byte[] destinationAddr) |
| 说明 | sourceAddr：设备源地址  destinationAddr：需要更变的目标地址 |

* 1. 设置控制板的电磁锁状态（异步模式）

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public void SetLockStatus(byte Addr, EnumDeviceSequence sequence) |
| 说明 | Addr：控制板地址  sequence：需要控制的Lock序号枚举 |

* 1. 获取控制板的电磁锁状态（异步模式）

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public void GetLockStatus(byte Addr, EnumDeviceSequence sequence) |
| 说明 | Addr：控制板地址  sequence：需要控制的Lock序号枚举 |

* 1. 设置控制板的 LED 灯状态（异步模式）

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public void SetLedStatus(byte Addr, EnumDeviceSequence sequence, byte status) |
| 说明 | Addr：控制板地址  sequence：需要控制的LED序号枚举  status：开关灯设置；01开灯；02关灯 |

* 1. 获取设备版本号信息（同步模式）

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public bool GetControlEdition(byte Addr,ref string ver) |
| 说明 | Addr：控制板地址  ver：回调的字符串数据 |

* 1. 获取设备版本号信息（异步模式）

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public void GetControlEdition(byte Addr) |
| 说明 | Addr：控制板地址 |

1. 反馈事件
   1. 异步反馈事件

|  |  |
| --- | --- |
| 命名空间 | SkLockApi.Device |
| 类 | SkLock |
| 方法 | public event void OnEquipmentFeedbackHandler(byte cmd, byte[] data\_bytes); |
| 说明 | cmd：命令类型  data\_bytes：反馈的数据帧中data数据 |