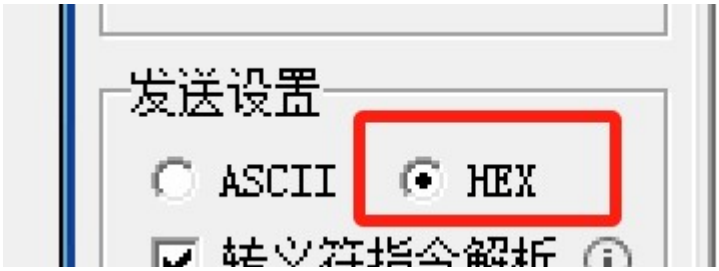
安亚麦 USB 串口控制三色报警灯

通讯说明 V2.2

开发案例代码及资料见:
阿里云盘:
https://www.aliyundrive.com/s/M6xVqtot1PM
Gitbub:
AmAYM001/USB-Alarm-Light: USB 控制三色灯的说明文件及驱动 (github.com)
https://github.com/AmAYM001/USB-Alarm-Light

1. 将灯 usb 插入设备(电脑/工控/平板等等)的 usb 口
2. 安装驱动<CH341_Drive.zip>
3. 针对系统分配 com 名称进行编程控制， 通过软件向对应的串口以 16 进制形式发送指令即可。
4. 

5. 提供 C++/c#/vb/ Python/java/node.js/html(web page)代码,末尾
6. 控制协议如下:

A.波特率:

随意设置，例如 9600

B.指令结构:

起始	地址	操作码	校验(sum)
0xA0	0x01	0x01	0xA2

地址名称	地址
黄灯	0x01
绿灯	0x02
红灯	0x03
蜂鸣	0x04
黄灯+蜂鸣	0x05
绿灯+蜂鸣	0x06
红灯+蜂鸣	0x07
全局	0x08

操作功能	操作码
关闭	0x01
打开	0x02
闪烁	0x03

校验:为前三码之和(sum)

C.操作码一览表:

关闭黄灯：A0 01 00 A1
打开黄灯：A0 01 01 A2
黄灯闪烁：A0 01 02 A3

关闭绿灯：A0 02 00 A2
打开绿灯：A0 02 01 A3
绿灯闪烁：A0 02 02 A4

关闭红灯：A0 03 00 A3
打开红灯：A0 03 01 A4
红灯闪烁：A0 03 02 A5

关闭蜂鸣：A0 04 00 A4
打开蜂鸣：A0 04 01 A5
间断蜂鸣：A0 04 02 A6

关闭红灯+蜂鸣：A0 07 00 A7
打开红灯+蜂鸣：A0 07 01 A8
红灯闪烁+蜂鸣：A0 07 02 A9

关闭全部：A0 00 00 A0
打开全部：A0 00 01 A1
全部闪烁：A0 00 02 A2

D.代码 demo

C++:

```
#include <windows.h>
#include <iostream>
using namespace std;
int main()
{
    HANDLE hCom;
    hCom = CreateFile(L"COM1", GENERIC_READ | GENERIC_WRITE, 0, NULL,
OPEN_EXISTING, 0, NULL);//打开串口 COM0
```

```
    if (hCom == INVALID_HANDLE_VALUE)
    {
        cout << "Error in opening serial port\n";
        return 0;
    }

    //串口参数设置
    DCB dcbSerialParams = { 0 };
    dcbSerialParams.DCBlength = sizeof(dcbSerialParams);
    if (!GetCommState(hCom, &dcbSerialParams))
    {
        cout << "Error in getting serial port state\n";
        CloseHandle(hCom);
        return 0;
    }
    dcbSerialParams.BaudRate = CBR_9600;
    dcbSerialParams.ByteSize = 8;
    dcbSerialParams.StopBits = ONESTOPBIT;
    dcbSerialParams.Parity = NOPARITY;
    if (!SetCommState(hCom, &dcbSerialParams))
    {
        cout << "Error in setting serial port state\n";
        CloseHandle(hCom);
        return 0;
    }

    // 配置串口超时时间
    COMMTIMEOUTS timeouts = { 0 };
    timeouts.ReadIntervalTimeout = 50;
    timeouts.ReadTotalTimeoutConstant = 50;
    timeouts.ReadTotalTimeoutMultiplier = 10;
    timeouts.WriteTotalTimeoutConstant = 50;
    timeouts.WriteTotalTimeoutMultiplier = 10;

    if (!SetCommTimeouts(hCom, &timeouts))
    {
        std::cerr << "Error setting serial port timeouts." << std::endl;
        CloseHandle(hCom);
        return 0;
    }

    //开始发送指令
    DWORD dwBytesWritten;
    BYTE byte[] = { 0xA0, 0x02, 0x02, 0xA4 };//控灯指令,绿灯
```

```
53.     if (!WriteFile(hCom, byte, sizeof(byte), &dwBytesWritten, NULL))
54.     {
55.         cout << "Error in writing data to serial port\n";
56.         CloseHandle(hCom);
57.         return 0;
58.     }
59.
83.     PurgeComm(hCom, PURGE_RXCLEAR);
84.     PurgeComm(hCom, PURGE_TXCLEAR);
85.     CloseHandle(hCom);87.
88. }
89.
```

C#:

```
1. this.sp = new SerialPort("COM4", 9600);
2. this.sp.DataBits = 8;
3. this.sp.Parity = Parity.None;
4. this.sp.StopBits = StopBits.One;
5. this.sp.Open();
6. byte[] buffer = new byte[] { 0xA0, 0x1, 0x1, 0xA2 };
7. this.sp.Write(buffer, 0x0, 0x4);
```

Python:

```
1. import serial #导入模块 2.
3. # 打开串口连接
4. ser = serial.Serial('COM2', 9600, timeout=1)
5. # 要发送的 16 进制数据
6. hex_data = [0xa0,0x01,0x01,0xa2]
7. # 发送数据
8. ser.write(hex_data)
9. # 关闭串口连接
10. ser.close()
```

Java:

```
import com.fazecast.jSerialComm.SerialPort;

public class SerialCommExample {

    public static void main(String[] args) {
        // 查找并打开串口
        SerialPort serialPort = SerialPort.getCommPort("COM3"); // 替换为你
的串口名称
        if (serialPort.openPort()) {
```

```
            System.out.println("串口已打开");

            // 16 进制数据，例如 "A0 01 01 A2"
            String hexData = " A00101A2";

            // 将 16 进制字符串转换为字节数组
            byte[] dataBytes = new byte[hexData.length() / 2];
            for (int i = 0; i < dataBytes.length; i++) {
                dataBytes[i] = (byte) ((Character.digit(hexData.charAt(2 *
i), 16) << 4)
                        + Character.digit(hexData.charAt(2 * i + 1), 16));
            }

            // 发送数据
            if (serialPort.writeBytes(dataBytes, dataBytes.length) ==
dataBytes.length) {
                System.out.println("数据已发送");
            } else {
                System.out.println("发送数据失败");
            }

            // 关闭串口
            serialPort.closePort();
            System.out.println("串口已关闭");
        } else {
            System.out.println("无法打开串口");
        }
    }
}
```

Node.js

SerialPort 版本： 12.x.x

文档:https://serialport.io/docs/guide-usage

```
const { SerialPort } = require('serialport')

//打开 com1
const port = new SerialPort({ path: 'com1', baudRate: 57600 })

// 亮灯 16 进制数据，比如 H [0xA0, 0x01, 0x01, 0xA2],
//这里改为需要控制的指令
```

```
const hexData = [0xA0, 0x01, 0x01, 0xA2];
// 将 16 进制数据转换为 Buffer 对象
const bufferData = Buffer.from(hexData);
port.write(bufferData, function (err)
{
      if (err)
      {
            return console.log('Error on write: ', err.message)
      }
      console.log('发送的数据: ',bufferData)
})

// Open errors will be emitted as an error event
port.on('error', function (err)
{
      console.log('Error: ', err.message)
})

// Switches the port into "flowing mode"
port.on('data', function (receivedData) {
      console.log('接收到的数据:', receivedData)
      // 在这里可以对 receivedData 进行进一步处理，例如解析数据等
})
```

Web API:
需浏览器支持清单:

	Chrome	Edge	Firefox	Opera
Serial	✓ 89	✓ 89	✗ No	✓ 75
getPorts	✓ 89	✓ 89	✗ No	✓ 75
requestPort	✓ 89	✓ 89	✗ No	✓ 75

```
<!DOCTYPE html>
<html>

<head>
	<title>Web Serial Example</title>
</head>

<body>
	<button id="connectButton">Connect</button>
	<button id="sendButton">Open Light</button>
	<textarea id="outputData" rows="4" cols="50" placeholder="接收到的数据" readonly></textarea>

	<script>
		let port;
		let keepReading = true;
		const connectButton = document.getElementById('connectButton');
		const sendButton = document.getElementById('sendButton');

		// 请求浏览器授权访问串口
		async function connectSerial() {
			if ('serial' in navigator) {
				port = await navigator.serial.requestPort();
				await port.open({ baudRate: 9600 }); // 设置波特率
				console.log('Serial port connected');
				Resposeloop();
			} else {
```

```
                        const outputDataElement =
document.getElementById('outputData');
                        const decodedData = new TextDecoder().decode("不支持
的浏览器");
                        outputDataElement.value = decodedData;
                  }
            }

            // 发送数据
            async function sendData() {
                  if (!port) {
                        console.error('Serial port not connected');
                        return;
                  }
                  const writer = port.writable.getWriter();
                  const data = new Uint8Array([0x01, 0x02, 0x03, 0x04]); //
要发送的指令
                  await writer.write(data);
                  writer.releaseLock();
                  console.log('Data sent to Light');
            }


            //接收
            async function Resposeloop() {
                  const reader = port.readable.getReader();
                  try {
                        while (keepReading) {
                              const { value, done } = await reader.read();
                              if (done) {
                                    reader.releaseLock();
                                    break;
                              }
                              if (value) {
                                    /*** TODO: deal with the data value ***/
                                    dealWithData(value);
                              }
                        }
                  } catch (error) {
                        console.error(error);
                  } finally {
                        console.log(port.readable, keepReading);
                  }
            }
```

```
            //关闭
            async function ClosePort() {
                  await port.close();
                  console.log("port closed");

            }
            /*** function dealWithData below ***/
            function dealWithData(data) {
                  const outputDataElement =
document.getElementById('outputData');
                  // const decodedData = new
TextDecoder().decode(data);
                  const hexData = Array.from(data, byte =>
byte.toString(16).padStart(2, '0')).join(' ');
                  outputDataElement.value = hexData;
            }
            connectButton.addEventListener('click', connectSerial);
            sendButton.addEventListener('click', sendData);
      </script>
</body>

</html>
```

商品路径:
https://item.taobao.com/item.htm?ft=t&id=761863586110
或者使用淘宝扫码: