# 南 通 大 学 实 验 报 告

院系 信息科学技术学院 年级专业 计171 姓名 刘乾 学号 1714021057

课程名称 嵌入式技术基础与实践 成绩

指导教师 李跃华 同组实验者 邹宸 实验日期 20191128

实验名称： 串口通信（SCI）实验

# 一.实验目的

（1）熟悉定时中断计时的工作及编程方法

（2）理解定时器模块的输入捕捉、输出比较、脉宽调制（PWM）功能的基本原理。

（3）掌握定时器模块的输入捕捉、输出比较、脉宽调制（PWM）编程方法。

（4）理解PWM占空比的含义。

（5）进一步深入理解MCU和C#串口通信的编程方法。

# 二.实验内容

## 实验一：利用TMP模块控制PWM波占空比继而控制小灯的亮暗程度；

### 实验目的：

1、利用TPM模块用来控制PWM波的占空比从而控制小灯的亮度 ；

2、通过UART模块实现输入1-100之间的数字来控制小灯亮暗

3、通过C#编程，实现窗口内滑动滚动条控制小灯的亮度

### 实验环境：

1）PC机一台，KL25开发板一片，导线若干

2) KDS开发环境，VS2017或更高平台

### 实验内容：

#### 1）设计串口接收模块UART

##### #1目的：

通过设计串口接收模块来接受输入的亮度值，根据对用的亮度值，改变duty——占空比

##### #2问题：

###### 1）如何正确识别每次输入的亮度值？

@方法：

1）考虑到接受中断一次只能接受一个字节，所以无法很好的区分输入的亮度；所以决定通过前面章节组帧的知识，来进行组帧传输；从符合条件的帧中提取亮度值；（只有满足亮度值范围的帧才会被提取亮度值）

2）通过设置一个静态全局变量conDuty来向TPM2模块中的函数pwm\_update(TPM2\_CH0, conDuty); 进行duty传值，每次TPM2模块中断都会传入更新后对应的值来更新占空比；

@源码：

**void** **UART1\_IRQHandler**(**void**) {

uint\_8 ch;

uint\_8 flag;

**static** uint\_8 buffer[64];

DISABLE\_INTERRUPTS;

//获取一个字节的输入

**if** (uart\_get\_re\_int(UART\_1)) {

ch = uart\_re1(UART\_1, &flag);

//调用组帧函数；

**if** (CreateFrame(ch, buffer) != 0) {

//进行条件选择，看看是几位数；

**if** (buffer[1] == 1) {

conDuty = buffer[2]-48; //进行值传递;

} **else** **if** (buffer[1] == 2) {

conDuty=0;//通过设置for循环来将conDuty递增到满足条件的值

uint\_8 conDuty=(buffer[2]-48)\*10+(buffer[3]-48);

} **else** **if** (buffer[1] == 3) {

**if** (buffer[2] == '1' && buffer[3] == '0' && buffer[4] == '0') {

conDuty = 100; //进行值传递;

}

} **else** {

}

}

}

**if** (flag) {

uart\_send1(UART\_1, ch); //向原串口发回一个字节

}

ENABLE\_INTERRUPTS;

}

#### 2）设计TPM1模块

##### #目的：

控制UART模块提取的亮度值来设置占空比duty进而控制小灯的亮暗程度；

##### #问题：

###### 1）如何将提取的亮度值传入更新占空比的函数中

@方法：

通过设置一个静态全局变量conDuty来连接TPM2模块和UART模块；在两个模块之间进行占空比的值传递

@源码：

**void TPM1\_IRQHandler(void) {**//深入了解中断的内涵

//默认值为100,则表示1S闪一次

**static** **float** duty = 90.0; //静态变量duty（占空比），数值越高，灯越暗

**static** uint\_8 Up\_Down = 1; //占空比增减标志

**if** (tpm\_get\_int(1) == 1) //若有TPM1的溢出中断

{

tpm\_clear\_int(1); //清TPM1的溢出中断标志

pwm\_update(TPM1\_CH0, conDuty); //PWN更新，注意参数！指明了哪个模块的那个通道！

}

}

#### 3）进行C#窗口编程，

##### #目的：实现滑动滚动条控制小灯的亮暗

##### #问题：

###### 1）如何或取滚动条的位置的取值然后进行组帧发送给MCU板进行亮度控制

@方法：1、定义HScrollBar 控件，定义一个字符串进行组成组帧；最后通过发送函数，将字符串发送给MCU板子，来控制小灯；

@源码：//滑动条控制小灯亮度

private void hScrollBar1\_Scroll(object sender, ScrollEventArgs e)

{

if (!sci.IsOpen)

{

//状态条进行提示

this.TSSLState.Text += "请先打开串口!";

return;

}

else

{

string str1 = "P";

string str2 = "";

int ex = 0;//存放滑动条的value

ex=hScrollBar1.Value;

if (ex <= 9)

{

str1 += "1";

}

else if (ex < 100)

{

str1 += "2";

}

else

{

str += "3";

}

str2 = ex.ToString();//整型转换为字符串进行组帧

str1 += str2;

str1 += "C";

//组帧成功；

//发送给板子；

PublicVar.g\_SendByteArray = new byte[10];

PublicVar.g\_SendByteArray =

System.Text.Encoding.Default.GetBytes(str1);//发送消息；

sci.SCISendData(ref PublicVar.g\_SendByteArray);

}

}



### 实验总结：

1、这次试验初步理解了TPM模块的使用；学习了占空比和小灯亮暗成都之间的关系

2、进一步学习了C#窗口编程里的组件的使用；

3、硬件这方面真的需要花很多时间去学习，我现在学的都只是皮毛而已；

## 实验二：发送系统时间至MCU实验板，并发送时间至C#窗口显示，统计时差；

### 实验目的：

1、利用MCU中的TPM模块用来计时；

2、通过UART模块实现输入发送系统时间至MCU中并且计时

3、通过C#编程，实现窗口内显示MCU返回的时间

### 实验环境：

1）PC机一台，KL25开发板一片，导线若干

2) KDS开发环境，VS2017或更高平台

### 实验内容：

#### 1）设计串口接收模块UART

##### #1目的：通过设计串口接收模块来接受系统发送的时间，通过组帧函数进行判断从而将系统发送过来的时间存放到g\_time[ ]数组中；

##### #2问题：

###### 1）如何正确将发送过来的时间存放到g\_time[ ]数组里

@方法：

1）考虑到C#输入的数字都是字符型，每个数字字符都对应一个（码）值；buffer缓存中的数字部分每位字节运算都是用对应的值计算；但是发送给c#的时候，必须发送数字相应的字符码；通过减去字符码和实际数值的差值进行存放；

@源码：

**void** **UART1\_IRQHandler**(**void**) {

uint\_8 ch;

uint\_8 flag;

**static** uint\_8 buffer[64];

DISABLE\_INTERRUPTS;

**if** (uart\_get\_re\_int(UART\_1)) {

ch = uart\_re1(UART\_1, &flag);

**if** (CreateFrame(ch, buffer) != 0) {

**if** (buffer[1] == 8) { //C#输入的数字都是字符型，每个数字字符都对应一个（码）值

//buffer缓存中的数字部分每位字节运算都是用对应的ASCII码值计算

//但是发送给c#的时候，必须发送数字相应的字符ASCII码值；

//计算hh:mm:ss

uint\_8 g\_time0 = (buffer[2]);

g\_time0 -= 48;

g\_time0 \*= 10;

buffer[3] -= 48;

g\_time0 += (buffer[3]);

uint\_8 g\_time1 = (buffer[5]);

g\_time1 -= 48;

g\_time1 \*= 10;

buffer[6] -= 48;

g\_time1 += (buffer[6]);

uint\_8 g\_time2 = (buffer[8]);

g\_time2 -= 48;

g\_time2 \*= 10;

buffer[9] -= 48;

g\_time2 += (buffer[9]);

g\_time[0] = g\_time0;

g\_time[1] = g\_time1;

g\_time[2] = g\_time2;

}

}

}

**if** (flag) {

uart\_send1(UART\_1, ch);

}

ENABLE\_INTERRUPTS;

}

#### 2）设计TPM0模块

##### #目的：控制TPM0模块进行计时，改变g\_time[ ]的数值

##### #问题：

###### 1）如何将MCU内部的g\_time[ ]数组内部的时间用字符串形式发送给C#界面；

@方法：

通过将g\_time[ ]内部的三位时分秒的数值进行位拆分，拆分出时分秒每个数值的十位，个位；加入字符‘ ：’，组合成时间格式发送给C#通信窗口

@源码：

**void** **TPM0\_IRQHandler**(**void**) {

**static** uint\_32 cnt; //中断次数

**if** (tpm\_get\_int(0) == 1) //若有TPM0的溢出中断

{

tpm\_clear\_int(0); //清TPM0的溢出中断标志

cnt++; //中断次数+1

**if** (cnt >= 100) //若达到100次（即1秒）

{

//实现数值向数字字符ASCII码值的转换,组成时间字符串

uart\_send1(UART\_1, (g\_time[0] / 10) + 48);

uart\_send1(UART\_1, (g\_time[0] % 10) + 48);

uart\_send1(UART\_1, ':');

uart\_send1(UART\_1, (g\_time[1] / 10) + 48);

uart\_send1(UART\_1, (g\_time[1] % 10) + 48);

uart\_send1(UART\_1, ':');

uart\_send1(UART\_1, (g\_time[2] / 10) + 48);

uart\_send1(UART\_1, (g\_time[2] % 10) + 48);

cnt = 0; //中断次数清0

//调用“秒+1计时子函数”，给全局变量数组g\_time赋值（时、分、秒）

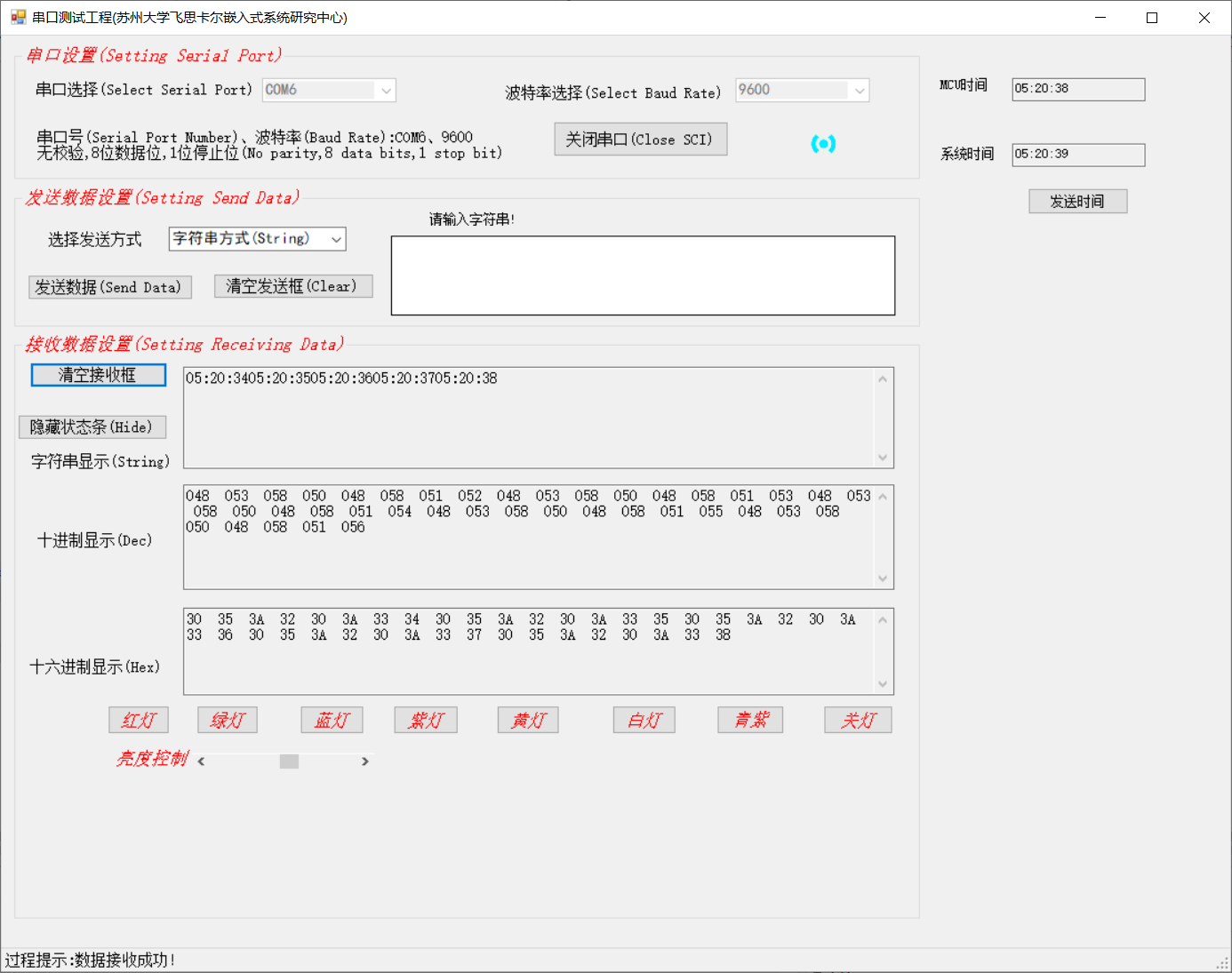
SecAdd1(g\_time);

}

}

}

#### 3）进行C#窗口编程



##### #目的：实现动态显示系统时间

##### #问题：

###### 1）如何每秒动态更新时间

@方法：1、定义Timer控件，利用方法private void timer1\_Tick\_1(object sender, EventArgs e)

用于每秒更新时间；

@源码：private void timer1\_Tick\_1(object sender, EventArgs e)

{

String date = DateTime.Now.ToString("hh:mm:ss");

systemTime.Text = date;

}

###### 2）发送系统时间至MCU板子

@方法：定义一个按钮控件，进行点击控件事件编程，实现系统时间发送；

@源码：

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

{

//更改全局变量值

change\_flag = 1;

String date = "P8" + DateTime.Now.ToString("hh:mm:ss") + "C";

//方法二

//date2用于分割时间字符串

String date2 = DateTime.Now.ToString("hh:mm:ss");

String[] date\_split = date2.Split(':');

for (int i = 0; i < 3; i++)

{

g\_time[i] = Int32.Parse(date\_split[i]);

}

//发送日期数据给板子；

PublicVar.g\_SendByteArray = new byte[10];

PublicVar.g\_SendByteArray =

System.Text.Encoding.Default.GetBytes(date);//发送消息；

sci.SCISendData(ref PublicVar.g\_SendByteArray);

}

###### 3）如何接收MCU发送的时间到Text控件里

@方法：定义private void SCIUpdateRevtxtbox\_2(Object textbox, string text) 方法，实现发送数据到TextBox mcu\_time 控件当中；在private void SCIPort\_DataReceived(object sender,

System.IO.Ports.SerialDataReceivedEventArgs e) 方法中调用SCIUpdateRevtxtbox\_2（）方法实现每次发送新时间时清空原来的旧时间；

@源码：

private void SCIPort\_DataReceived(object sender,

System.IO.Ports.SerialDataReceivedEventArgs e)

{

String str = String.Empty;

bool Flag;//标记串口接收数据是否成功

int len;//标记接收的数据的长度

byte[] ch2 = new byte[2];

//ComDevice.Encoding = System.Text.Encoding.GetEncoding("GB2312");

//调用串口接收函数,并返回结果

Flag = sci.SCIReceiveData(ref PublicVar.g\_ReceiveByteArray);

if (Flag == true)//串口接收数据成功

{

len = PublicVar.g\_ReceiveByteArray.Length;

//对于字符串形式,考虑到可能有汉字,

//直接调用系统定义的函数,处理整个字符串

str = Encoding.GetEncoding("GB2312").GetString(PublicVar.g\_ReceiveByteArray);

//

SCIUpdateRevtxtbox(TbShowString, str);

//发送MCU时间到C#窗口上

if (change\_flag == 1)

{

SecAdd1(g\_time);

String date = g\_time[0].ToString() +":"+ g\_time[1].ToString() + ":"

+ g\_time[2].ToString();

date = String.Format("{0:hh:mm:ss}", date);

//这是第一种最好的方法；通过MCU发送MCU里面的g\_time[]数组的时间显示到文本框里面

SCIUpdateRevtxtbox\_2(mcuTime, str);

//第二种办法是通过在C#内设置一个int g\_time[]数组；

//将时间存放到数组中，通过MCU发送秒加一信号，实现MCU同样的计时功能；

//缺点：时间显示格式不能统一

//SCIUpdateRevtxtbox\_2(mcuTime, date);

}

//十进制和十六进制形式按字节进行处理

for (int i = 0; i < len; i++)

{

//十进制都是按照三位来显示,字节之间有空格表示区分

SCIUpdateRevtxtbox(TbShowDec,

PublicVar.g\_ReceiveByteArray[i].ToString("D3") + " ");

//十六进制都是按照两位来显示,字节之间有空格表示区分

SCIUpdateRevtxtbox(TbShowHex,

PublicVar.g\_ReceiveByteArray[i].ToString("X2") + " ");

}

// sci.SCIReceInt(SCIPort, 1);//设置产生接收中断的字节数【2014-5-5 注释，否则会导致程序无响应】

this.TSSLState.Text = "过程提示:数据接收成功!";

}

//接收数据失败

else

{

//sci.SCIReceInt(SCIPort, 1);//设置产生接收中断的字节数【2014-5-5 注释，否则会导致程序无响应】

this.TSSLState.Text = "过程提示:数据接收失败!";

}

}

private void SCIUpdateRevtxtbox\_2(Object textbox, string text)

{

//textbox显示文本与串口执行不在同一个线程中

if (((TextBox)textbox).InvokeRequired)

{

handleinterfaceupdatedelegate InterFaceUpdate = new

handleinterfaceupdatedelegate(SCIUpdateRevtxtbox\_2);

this.Invoke(InterFaceUpdate, new object[] { textbox, text });

}

else

{

((TextBox)textbox).Text = text;

//把光标放在第一行

((TextBox)textbox).SelectionStart = 0;

//将文本框中的内容调整到当前插入符号位置

((TextBox)textbox).ScrollToCaret();

}

}

4）如何在点击发送按钮后才发送MCU时间到C#通信窗口；

@方法：设置一个显示MCU系统时间标志private int change\_flag = 0——传送时间开始标志

@源码：private void sendButton\_Click(object sender, EventArgs e)

{

//更改全局变量值

change\_flag = 1;

String date = "P8" + DateTime.Now.ToString("hh:mm:ss") + "C";

//date2用于分割时间字符串

String date2 = DateTime.Now.ToString("hh:mm:ss");

String[] date\_split = date2.Split(':');

for (int i = 0; i < 3; i++)

{

g\_time[i] = Int32.Parse(date\_split[i]);

}

//发送日期数据给板子；

PublicVar.g\_SendByteArray = new byte[10];

PublicVar.g\_SendByteArray =

System.Text.Encoding.Default.GetBytes(date);//发送消息；

sci.SCISendData(ref PublicVar.g\_SendByteArray);

}

### 实验总结：

1、这次试验初步理解了TPM模块的使用；学习了C#中的时间组件使用，能利用系统内部方法实现动态地更新时间了；

2、还有一点就是花了很长的时间弄明白了C#与MCU进行传输的字符具体使用方法，每一个字符字节的值是对应他的ASCII码的十进制数值；一开始在编写MCU中的UART\_1控件的内容的时候给我带来了很大的麻烦；花了很长时间才弄懂了其中地转换方法；

3、还有就是明白了静态全局变量的生存周期和时间数组的时间保存的关系，从而能够通过buffer[ ]传输时间到MCU中的g\_time[ ] 中

3、对C#的语法学习也在编程的过程中得到一定的锻炼，对程序结构的设计也得到了训练，学习嵌入式可以训练自己很多方面的能力！全面提高！

4、嵌入式还是需要花很大很大的气力去学习，要提高自己的学习效率，积极向同学老师请教才行，要培养自己的兴趣！才能有源源不断学习的乐趣！