

---

# 综合实验—展馆灯光控制

朱雪秦

实验与创新实践教育中心

实验内容：3个功能验证实验，1个综合设计

实验形式：学生动手实践为主，老师讲解为辅

实验报告：报告模板，不需包含验证实验，

只需要写综合设计如何实现，**个人报告**

# 目录 CONTENTS

- 1 实验须知
- 2 总体设计
- 3 详细设计——功能验证实验
- 4 C程序编码规范
- 5 软件设计说明书

## 第一部分

# 实验须知

# 一、实验须知

## ◆ 实验目的

1. 了解光敏电阻等光线传感器的原理；
2. 了解ADC数据采集软件滤波算法及其运用；
3. 了解单片机PWM波形产生的原理及其在LED调光的运用；
4. 了解单片机系统设计流程；
5. 了解单片机软件设计文件编写及程序编写规范。

## ◆ 实验设备

1. MSP430F5529LP+MSP430F5529 POCKET KIT开发板一套；
2. 光敏电阻传感器模块一套；
3. PC机操作系统Windows 10，CCS集成开发环境。

# 一、实验须知

## ◆ 安全事项

1. 用电安全
2. 仪器操作使用安全
3. 调试安全

## ◆ 实验要求

1. 课堂实验要求按照规定步骤完成实验程序编写，调试程序、完成功能演示。
2. 了解课后作业要求。
3. 考核：出勤、课堂实验、**作业难度**、程序、说明书

## 第二部分

# 总体设计

## 二、总体设计—功能描述

该系统主要用于展览馆等需要解说员解说，且需要调节光线以达到最佳演示效果的场合。系统检测到外界声音后打开灯光，系统根据周边环境光的情况自动调整**LED**灯的亮度，以达到最佳展示效果。如果一定时间内都没有检测到声音信号，则自动关闭**LED**灯，以达到节能目的。

从中提取关键信息：

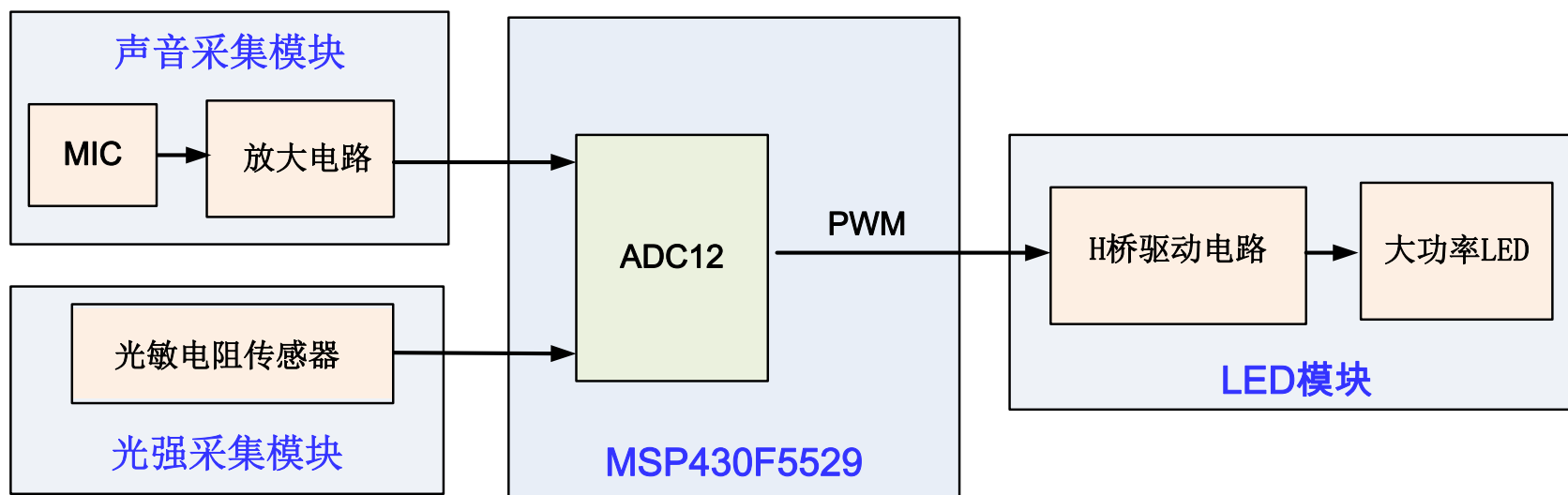
- 1.能感知声音信号
- 2.能感知光强信号
- 3.能调节**LED**灯亮度
- 4.能自动关闭**LED**灯

声控灯+自动调节亮度



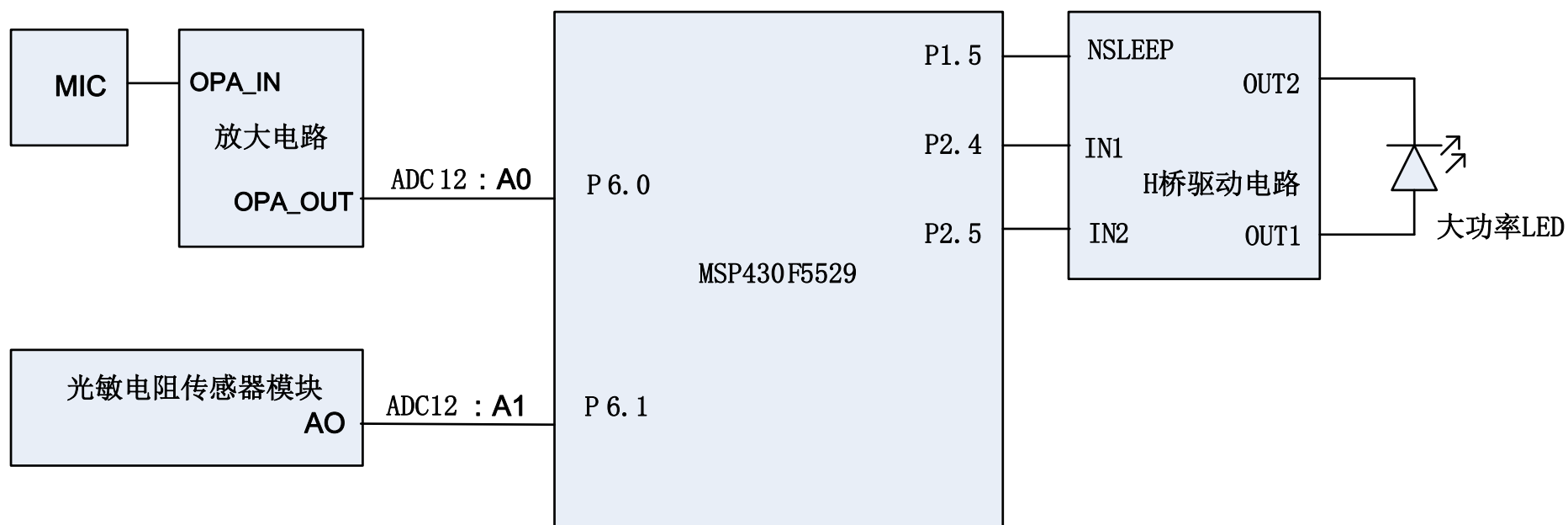
## 二、总体设计—功能描述

原理框图



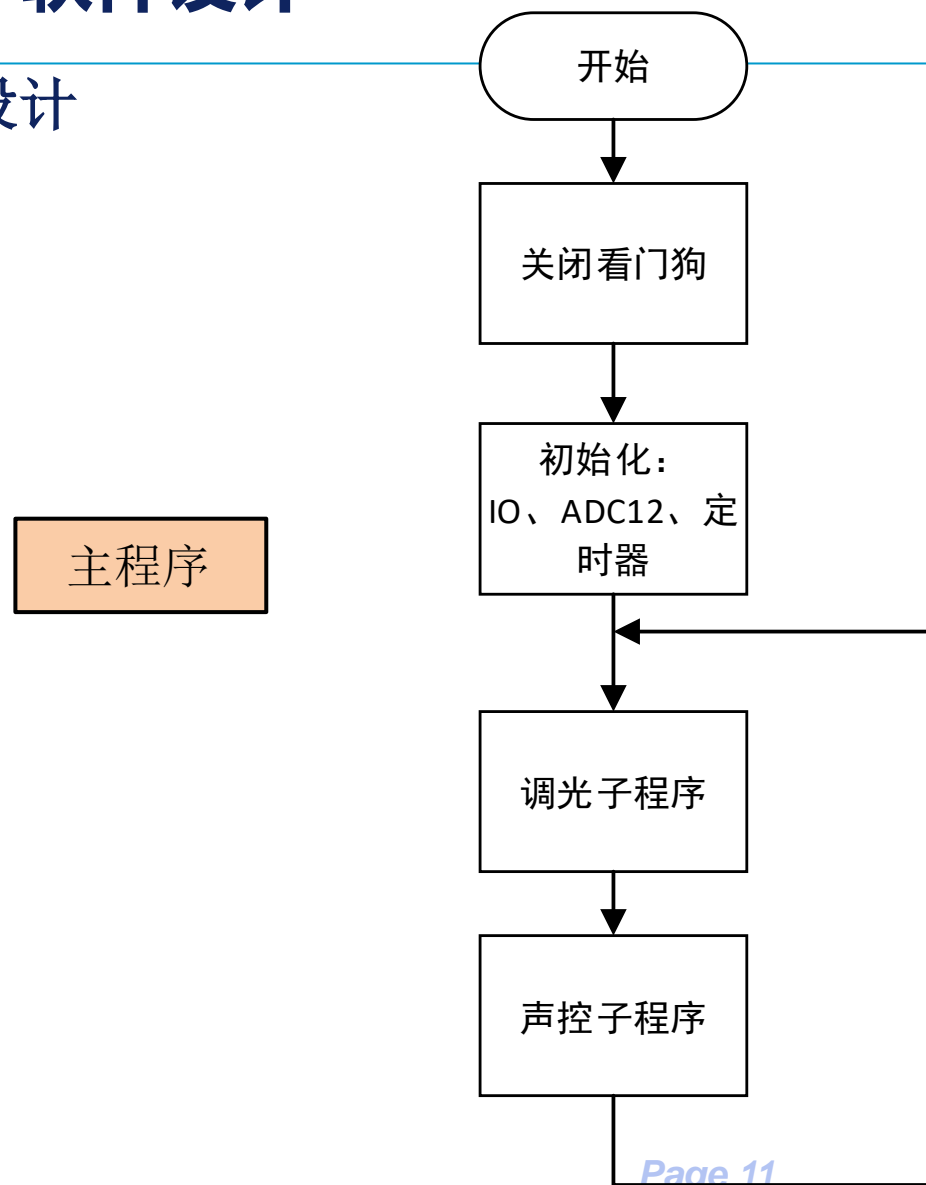
## 二、总体设计—硬件设计

### 硬件设计



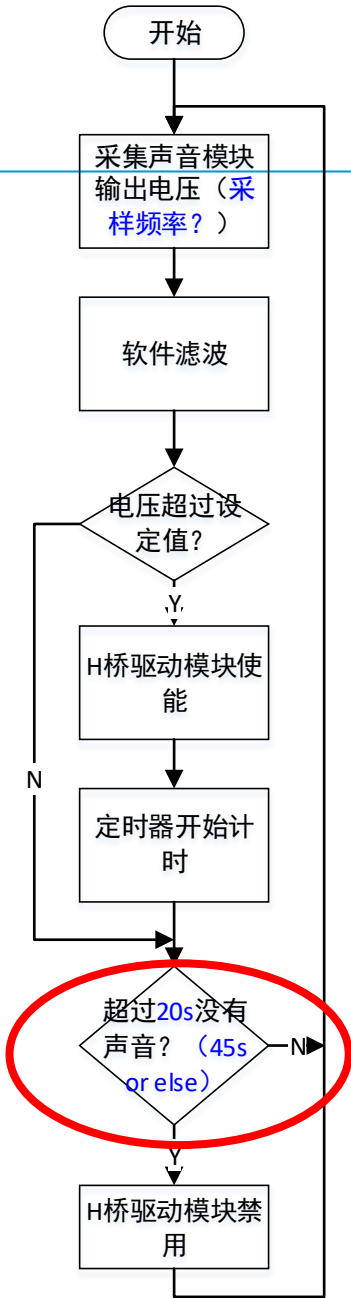
## 二、总体设计—软件设计

### 软件设计

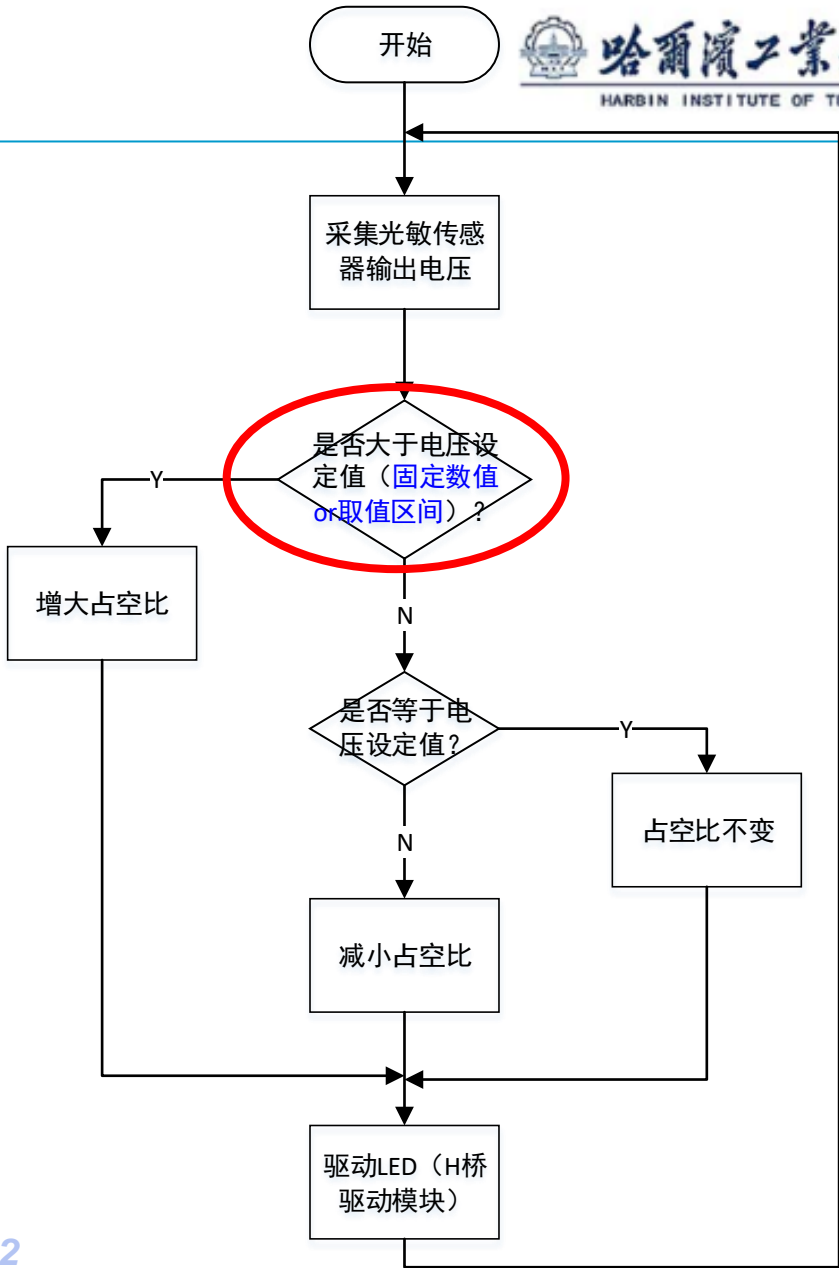


# 二、总体设计—软件设计

声控子程序



调光子程序



## 第三部分

# 详细设计——功能验证实验

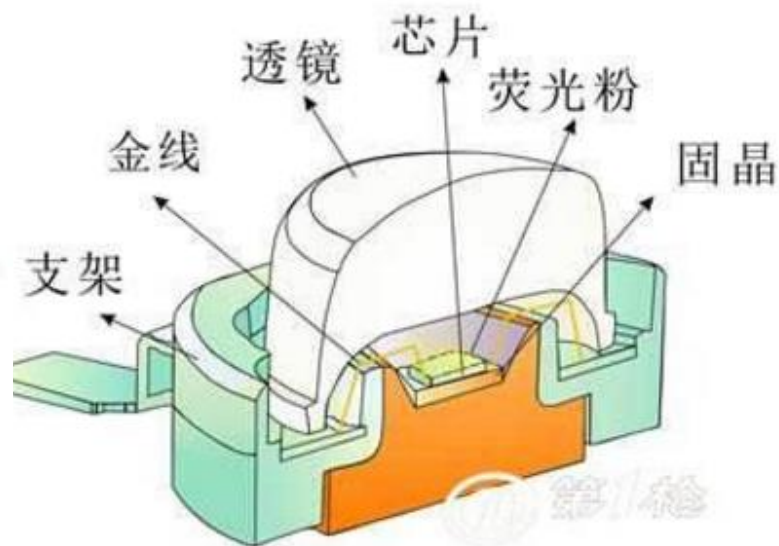
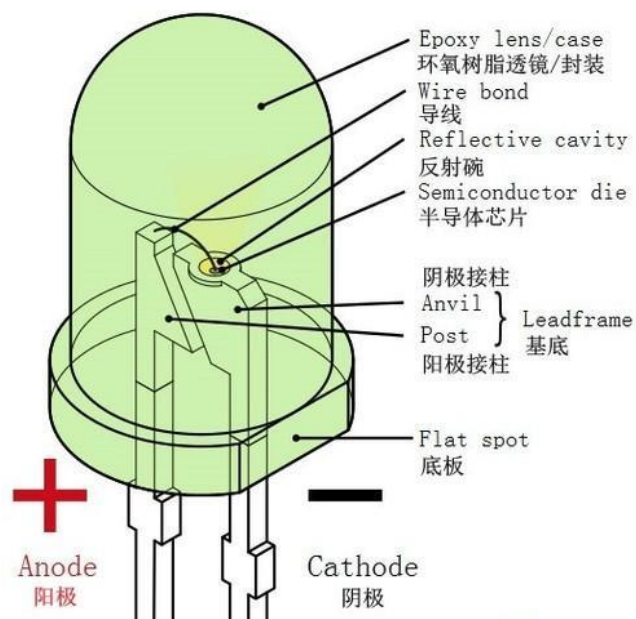
### 三、详细设计—LED驱动控制实验

#### 从受控对象——大功率LED开始

LED (Light Emitting Diode), 发光二极管, 是一种能够将电能转化为可见光的固态的半导体器件。

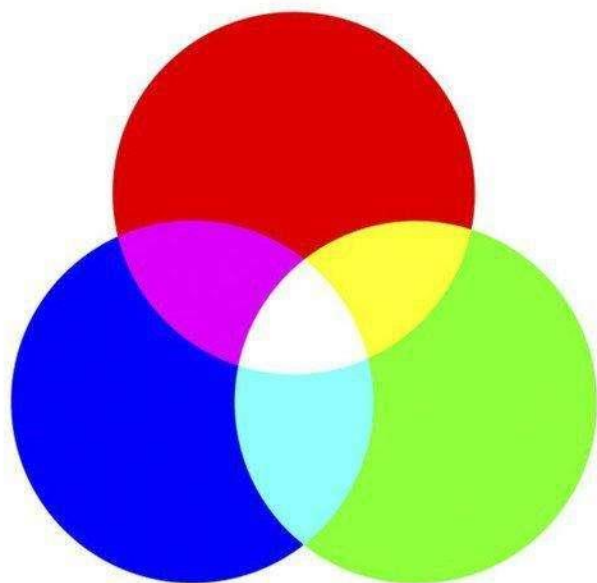
◆ 在一定范围内, 电流越大, 亮度越大。

◆ 单片机的输出电流不足以驱动大功率LED



### 三、详细设计—LED驱动控制实验

LED通常由含镓(Ga)、砷(As)、磷(P)、氮(N)等的化合物制成，砷化镓二极管发红光(Red)，磷化镓二极管发绿光(Green)，碳化硅二极管发黄光，氮化镓二极管发蓝光(Blue)。



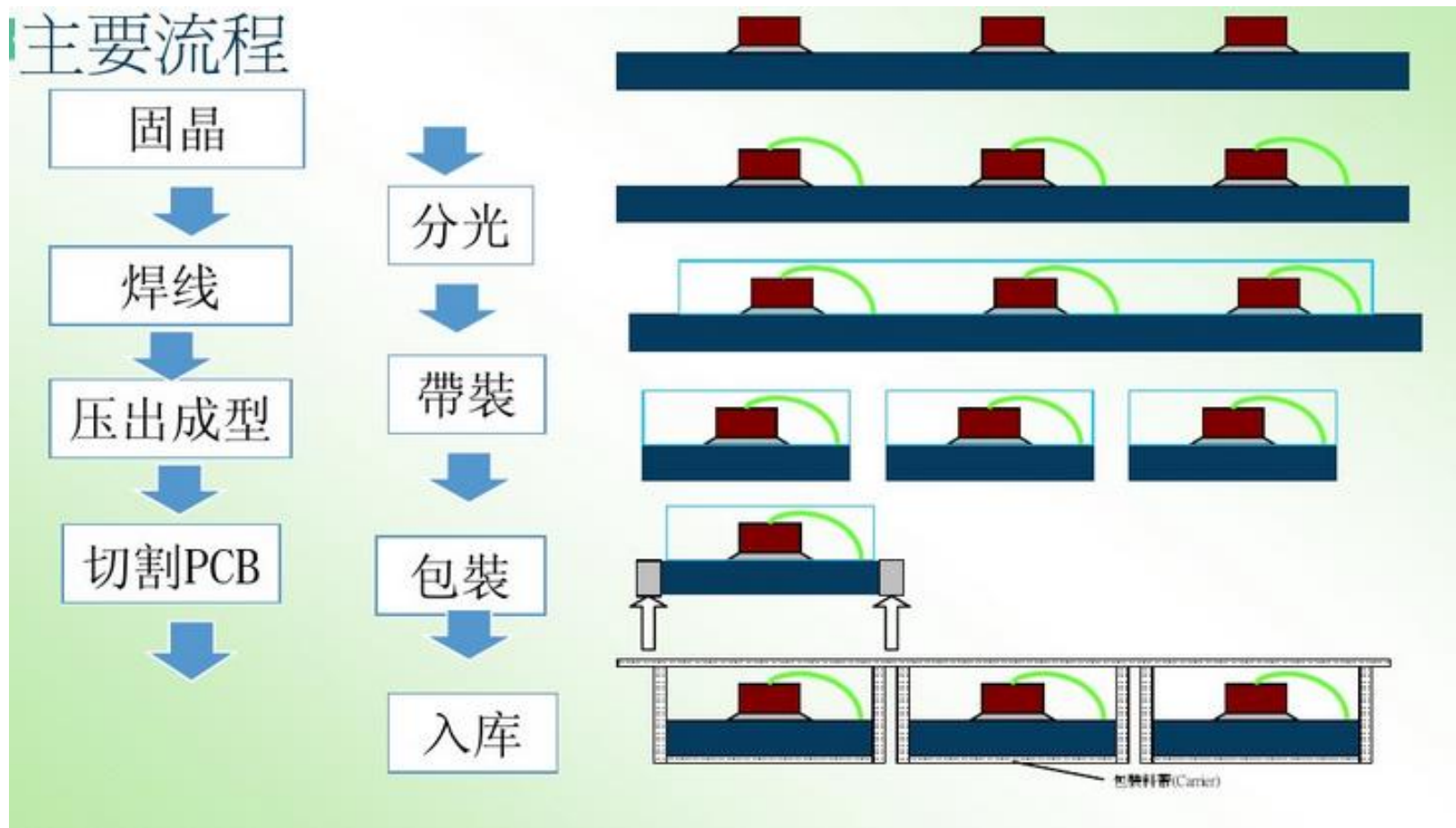
荧光粉的作用



RGBW技术就是在原有的RGB三原色上增加了W白色子像素，成为四色型像素设计。液晶面板的透光率高，在显示相同亮度的画面时，其耗电量更低;而相同功耗的情况下，亮度大幅提高，这使得画面层次更加分明，画面更通透。

### 三、详细设计—LED驱动控制实验

#### LED 制造流程

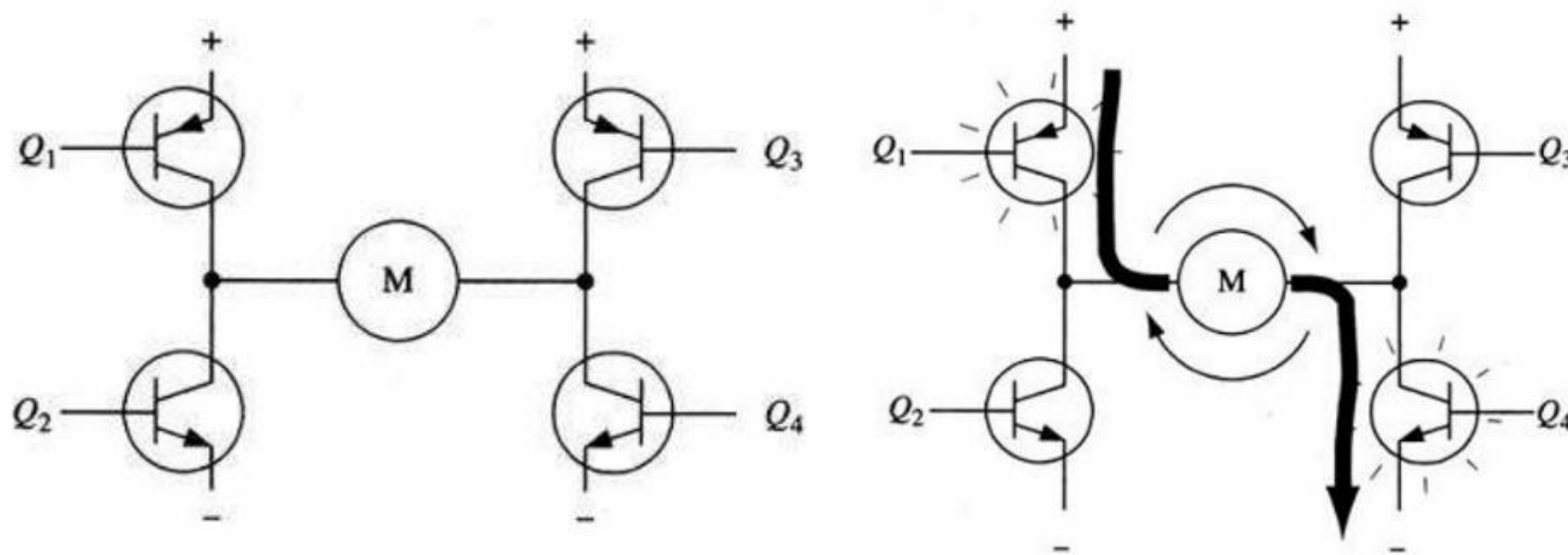




### 三、详细设计—LED驱动控制实验

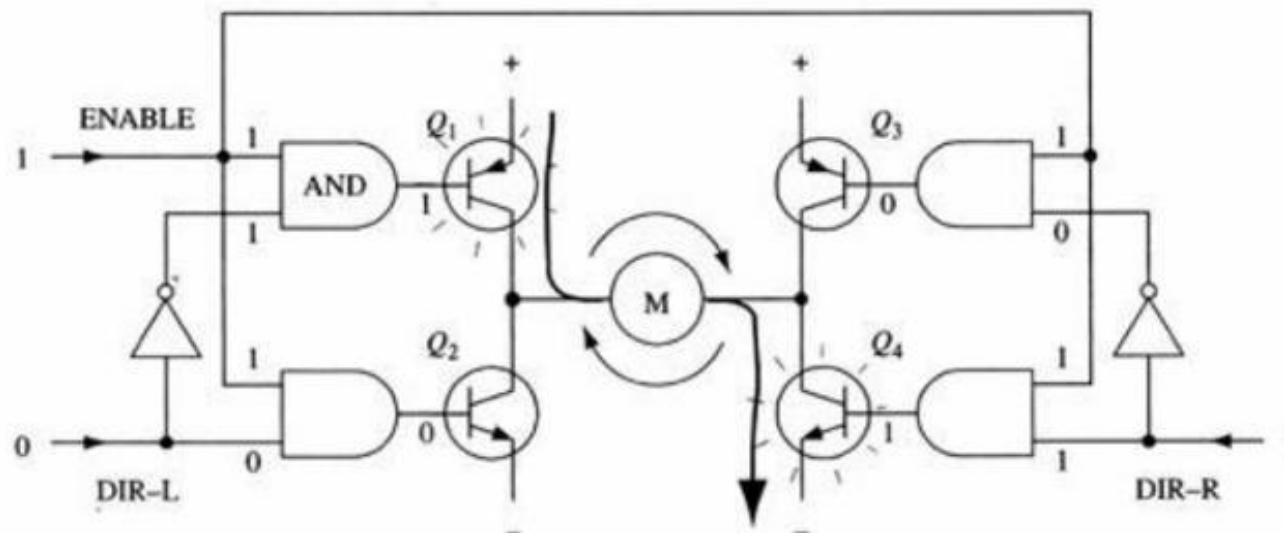
在单片机运用中，对于需要较大电流的元件，往往使用三极管来驱动，如数码管、蜂鸣器等。

在MSP430F5529 POCKET KIT中，典型的驱动电路最就是 H 桥驱动电路。



### 三、详细设计—LED驱动控制实验

保证 H 桥上两个同侧的三极管不会同时导通非常重要。如果三极管 Q1、Q2 同时导通，那么电流就会从正极穿过两个三极管直接回到负极。此时，电路中除了三极管外没有其他任何负载，因此电路上的电流就可能达到最大值（该电流仅受电源性能限制），甚至烧坏三极管。



### 三、详细设计—LED驱动控制实验

口袋板上使用了TI DRV8837低电压电机驱动芯片。

芯片工作电压范围VCC: 1.8 V—7 V

独立的电机电源VM: 0—11V



drv8837.pdf

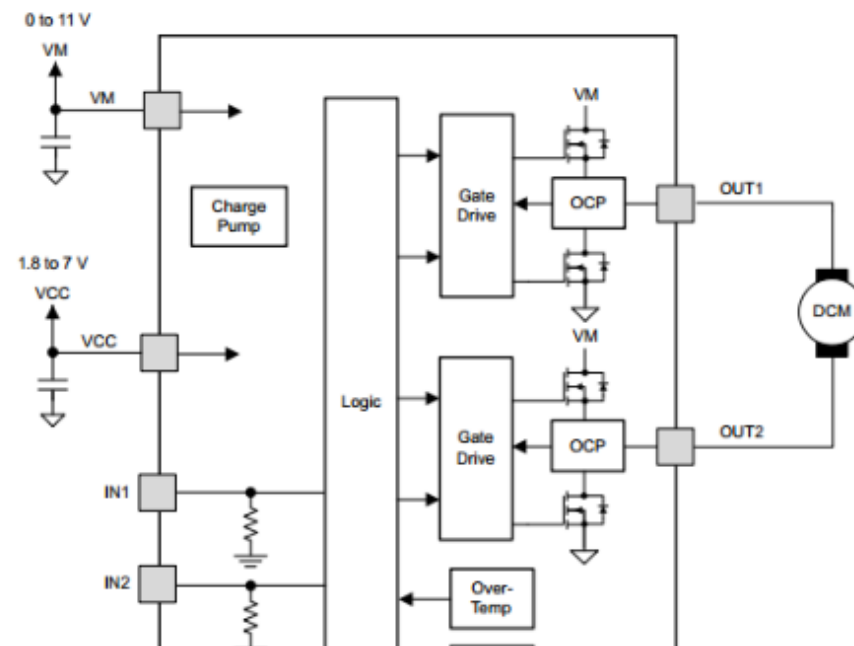


Table 1. DRV8837 Device Logic

nSLEEP	IN1	IN2	OUT1	OUT2	FUNCTION (DC MOTOR)
0	X	X	Z	Z	Coast
1	0	0	Z	Z	Coast
1	0	1	L	H	Reverse
1	1	0	H	L	Forward
1	1	1	L	L	Brake

### 三、详细设计—LED驱动控制实验

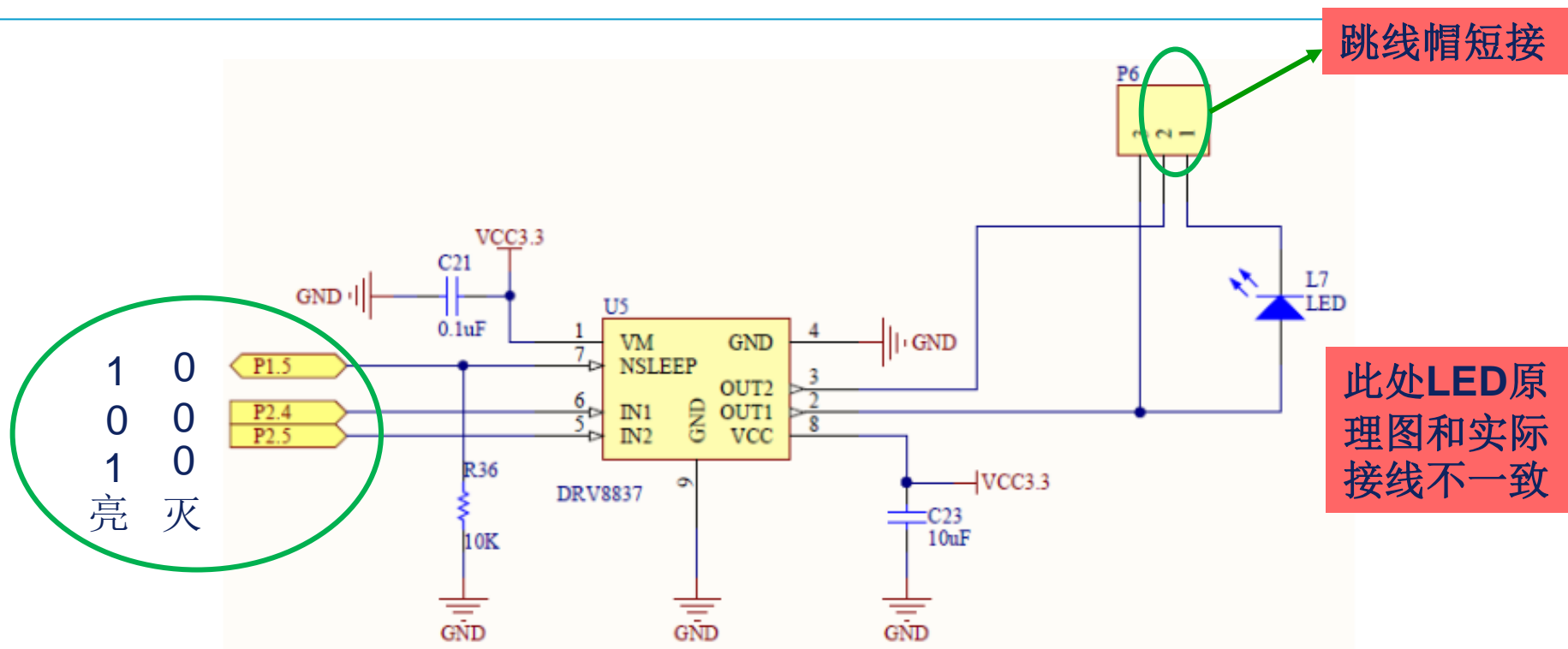
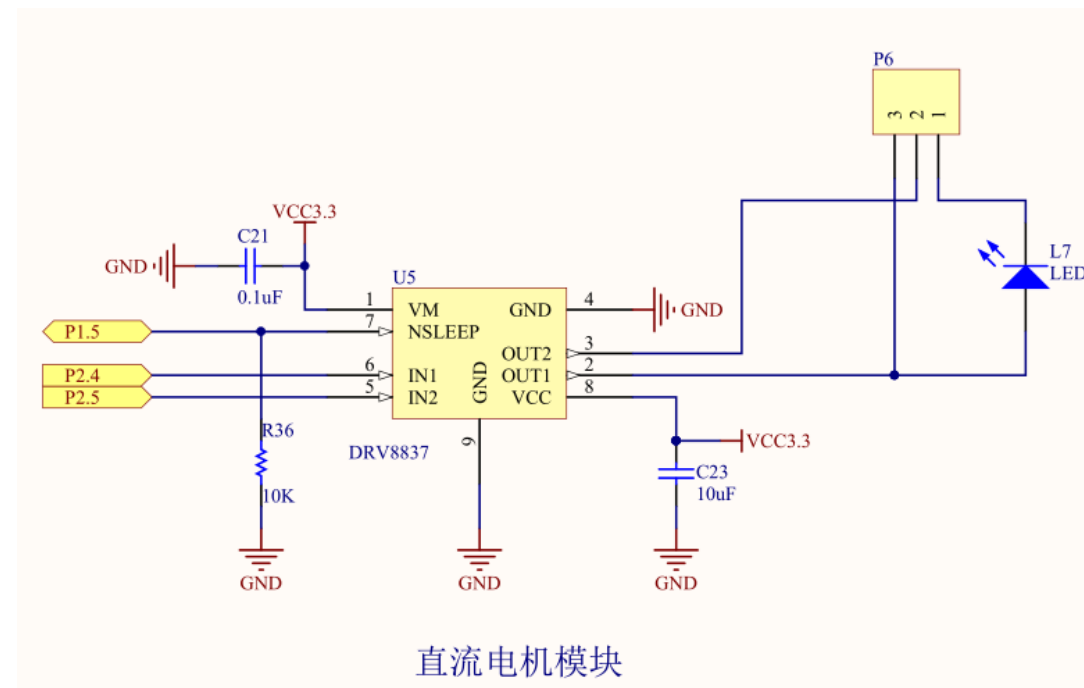
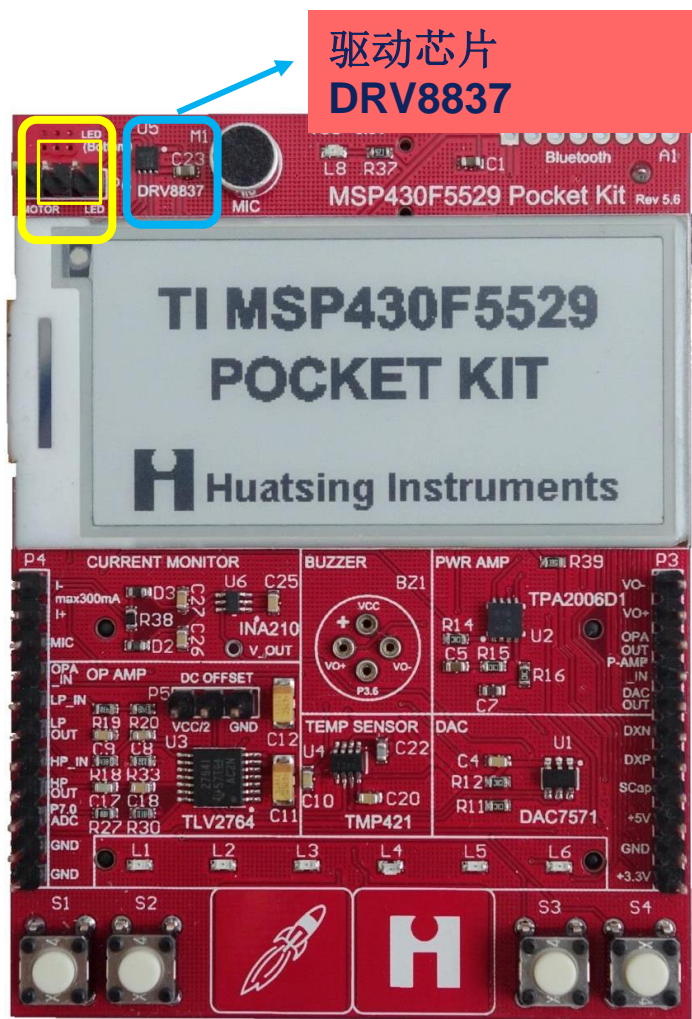


Table 1. DRV8837 Device Logic

nSLEEP	IN1	IN2	OUT1	OUT2	FUNCTION (DC MOTOR)
0	X	X	Z	Z	Coast
1	0	0	Z	Z	Coast
1	0	1	L	H	Reverse
1	1	0	H	L	Forward
1	1	1	L	L	Brake

### 三、详细设计—LED驱动控制实验

电路连接：用跳线帽短接图上插针



直流电机模块



# 课上实验1

通过按键S1、S2控制大功率LED灯的亮灭：按下S1，LED灯点亮；按下S2，LED灯熄灭。



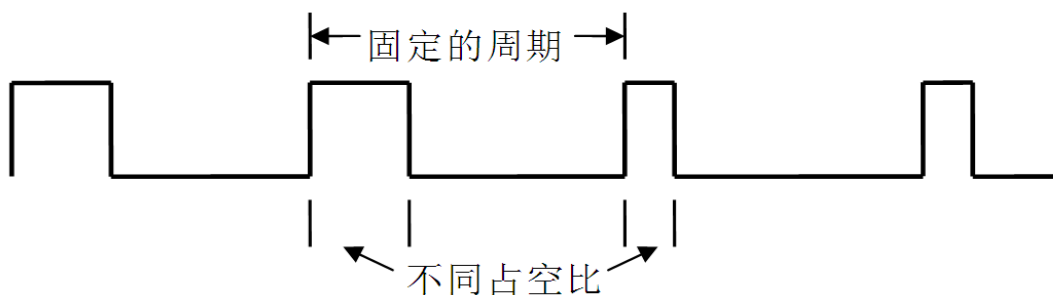
伪代码

如何调节LED灯的亮度?

### 三、详细设计—LED驱动控制实验

PWM (Pulse-Width Modulation): 脉宽调制的缩写

PWM信号是一种具有**固定周期****不定占空比**的数字信号。

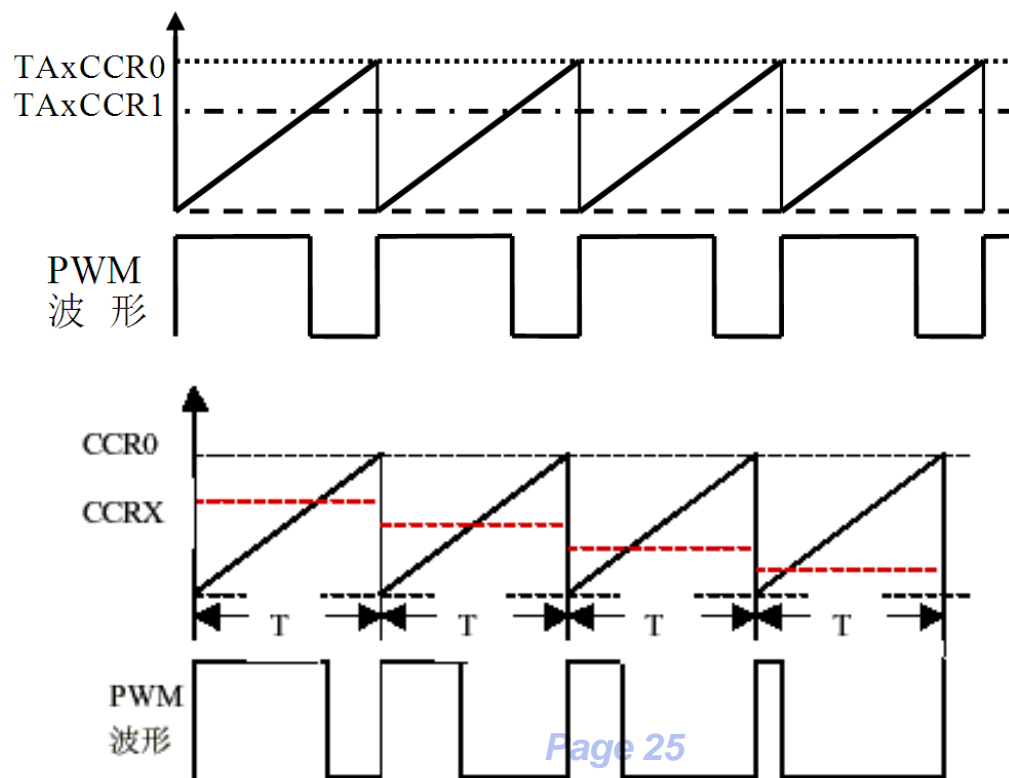


用水阀的反复开关动作模拟 PWM 波对电流的控制作用。水池中水面的上升速度表示了“平均水流量”的大小。当开启与关闭的总时间不变时，两者的时间之比决定了“平均水流”的值



### 三、详细设计—LED驱动控制实验

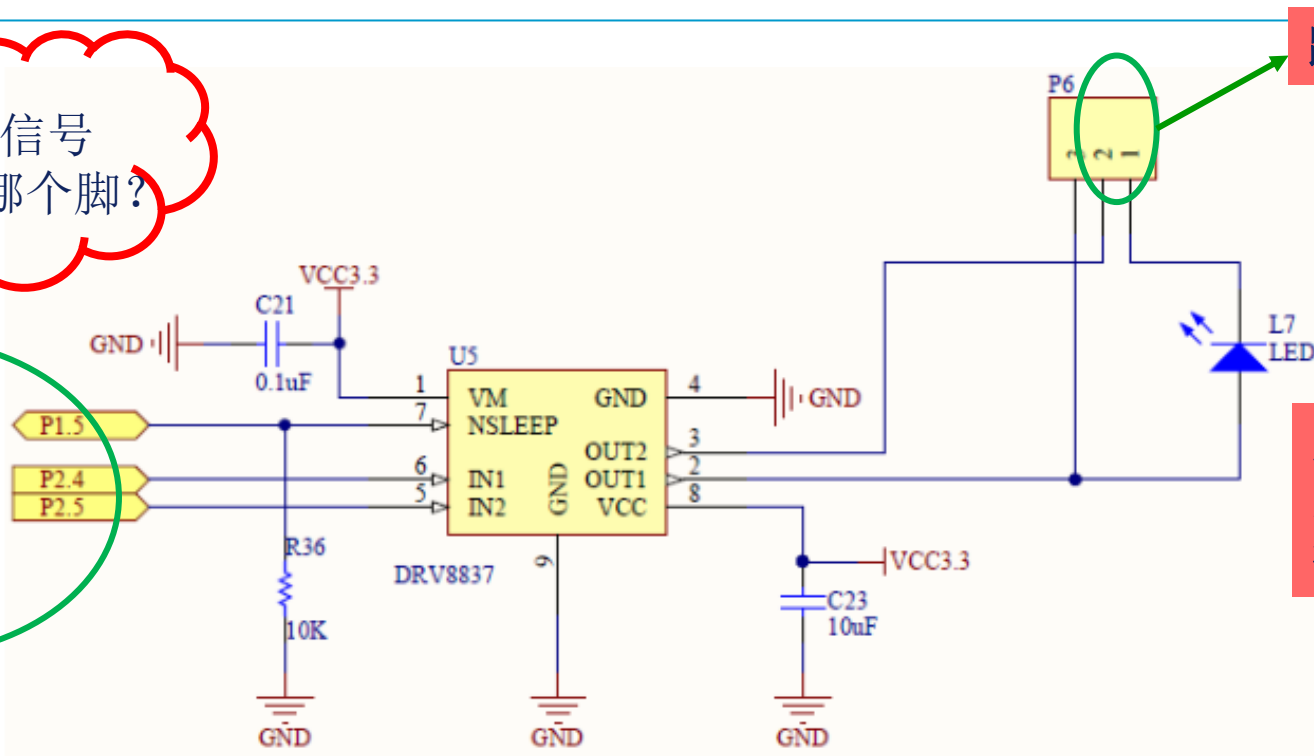
Timer\_A定时器的计数器工作在增计数方式或者增减计数方式，  
输出采用输出模式7（复位/置位模式）或者输出模式6（翻转/置位模式），  
则可利用寄存器TAXCCR0控制PWM波形的周期，用某个寄存器TAXCCRx控制占空比。  
这样Timer\_A就可以产生出任意占空比的PWM波形。



### 三、详细设计—LED驱动控制实验

PWM信号  
接至哪个脚?

1 1 1  
0 0 1  
0 1 1  
灭 亮 灭



跳线帽短接

此处LED原理图和实际接线不一致

Table 1. DRV8837 Device Logic

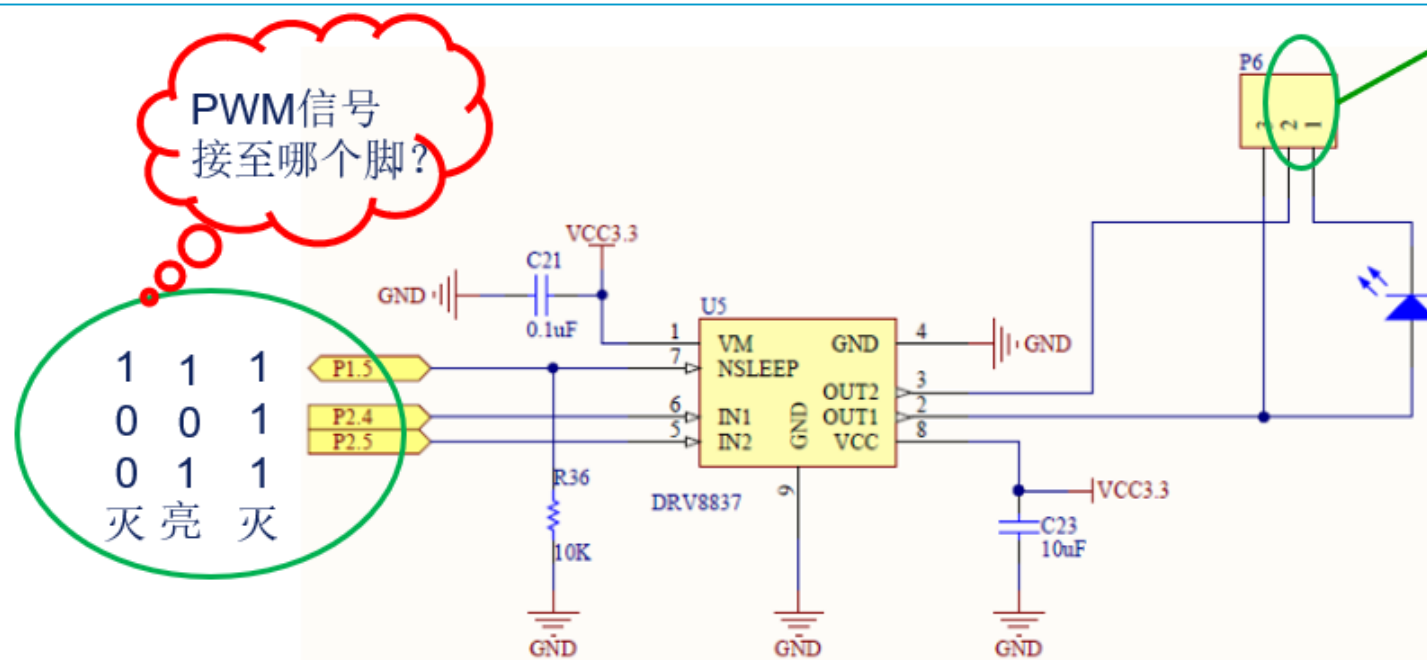
nSLEEP	IN1	IN2	OUT1	OUT2	FUNCTION (DC MOTOR)
0	X	X	Z	Z	Coast
1	0	0	Z	Z	Coast
1	0	1	L	H	Reverse
1	1	0	H	L	Forward
1	1	1	L	L	Brake

在上述H桥驱动大功率LED的电路中，为实现亮度调节，PWM信号可以接至驱动芯片8837的哪个管脚？

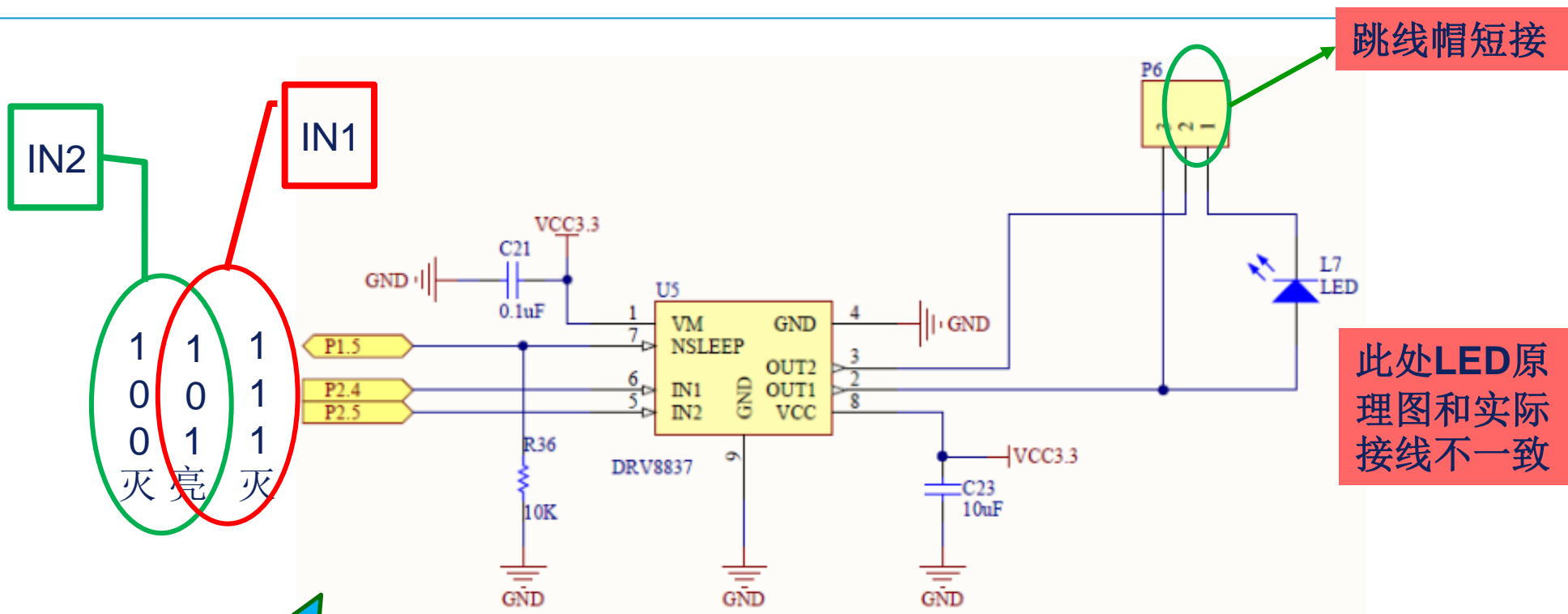
A. nSLEEP

B. IN1

C. IN2



### 三、详细设计—LED驱动控制实验

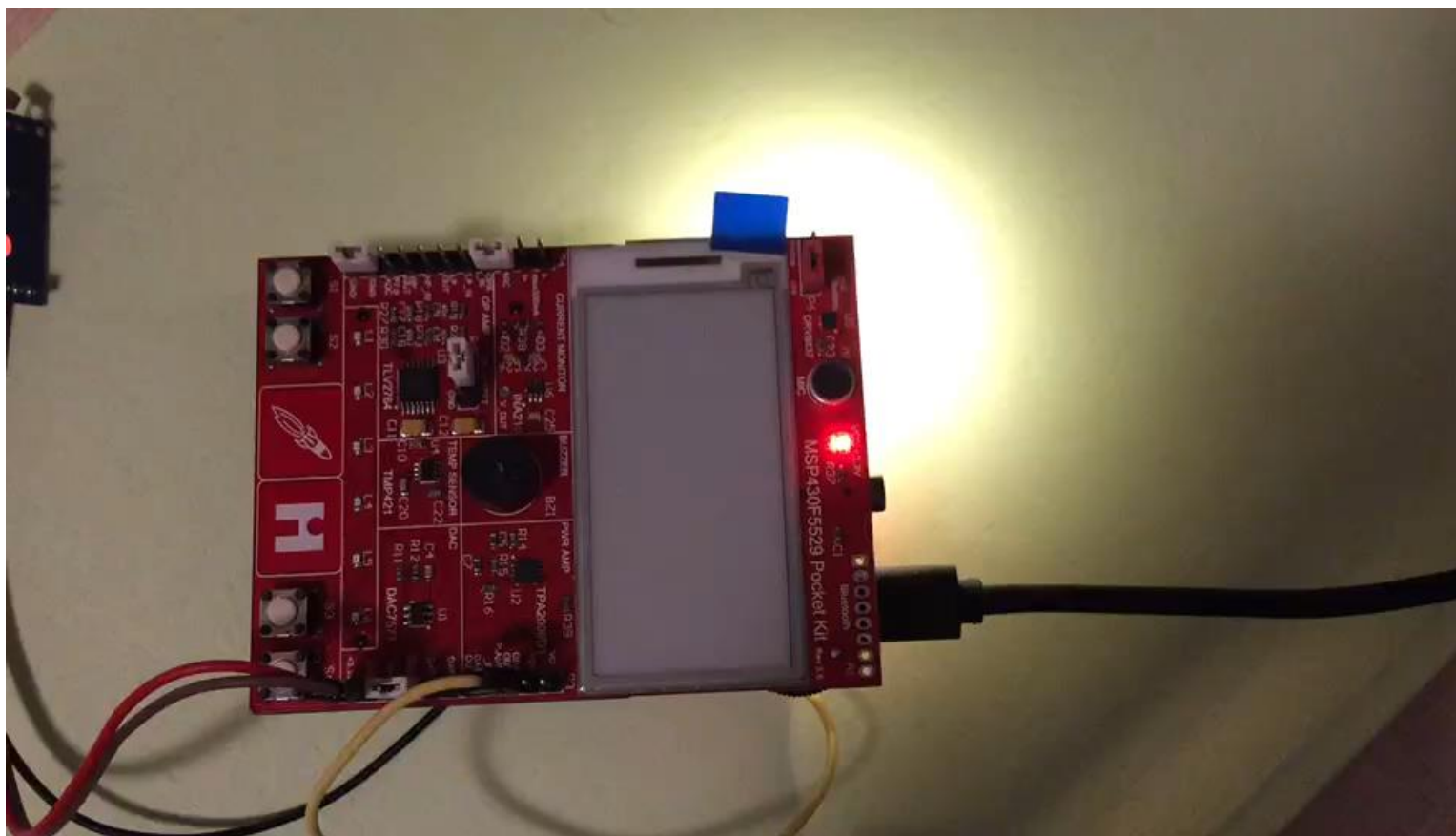


实现方法  
不唯一！

Table 1. DRV8837 Device Logic

nSLEEP	IN1	IN2	OUT1	OUT2	FUNCTION (DC MOTOR)
0	X	X	Z	Z	Coast
1	0	0	Z	Z	Coast
1	0	1	L	H	Reverse
1	1	0	H	L	Forward
1	1	1	L	L	Brake

通过PWM控制大功率LED灯，使之产生呼吸灯的效果。



伪代码

### 三、详细设计—LED驱动控制实验

延时接口

```
void Delay (unsigned int delaytime)
{
    int i,j;
    for( i = 0; i < delaytime; i++ )
        for( j = 0; j < 10; j++ );
}
```

或直接使用 **delay\_cycles();**

时钟设置请使用老师给的代码初始化时钟，并且将TA时钟源  
选择为SMCLK(8MHz)

```
/******
位操作定义
*****/
#define SET(Reg,n)  Reg |= n    //置位为1
#define CLR(Reg,n)  Reg &= ~n    //置位为0
#define NOT(Reg,n)  Reg ^= n    //取反操作
```

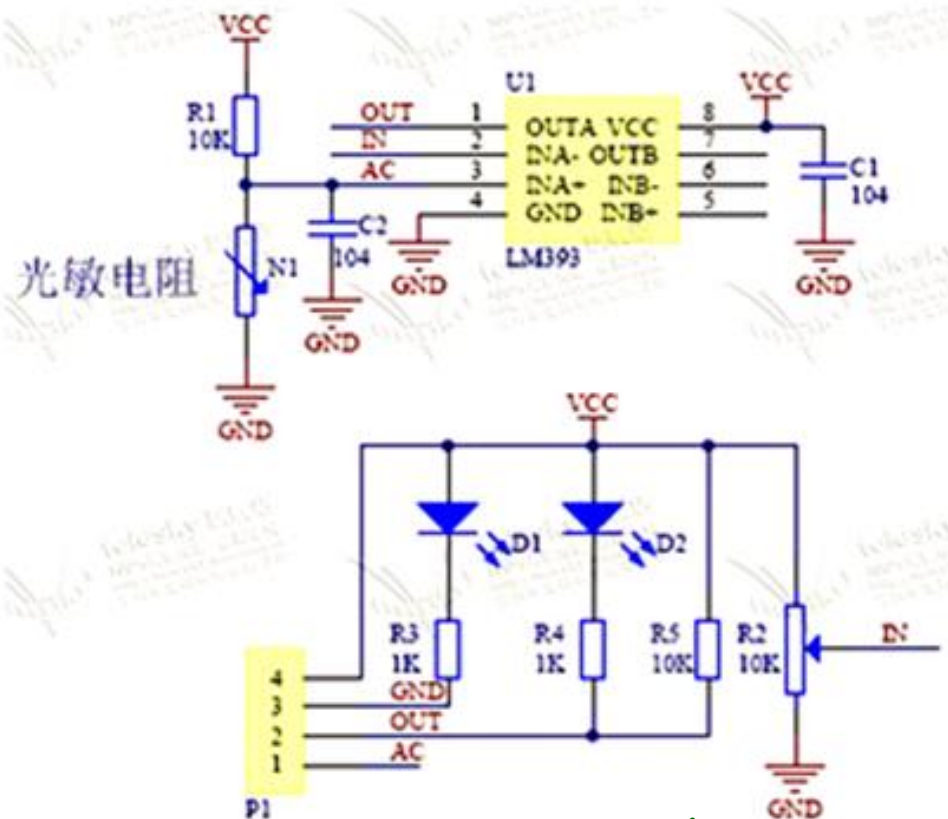
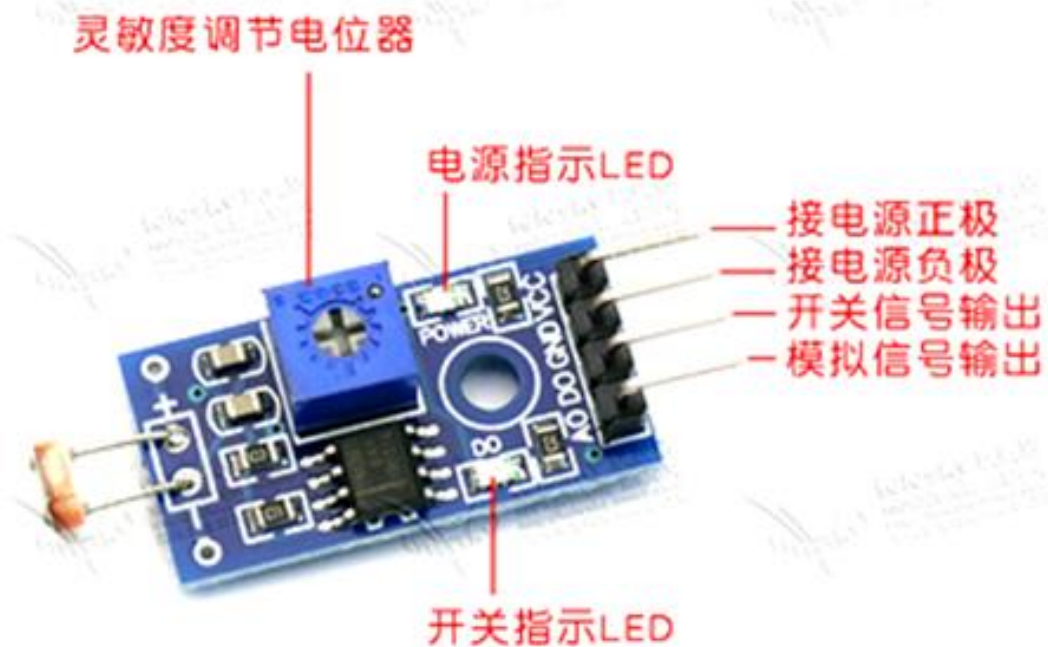
### 三、详细设计—LED驱动控制实验

#### **void initClock()**

```
{//最终效果: MCLK:16MHZ,SMCLK:8MHZ,ACLK:32KHZ
    UCSCTL6 &= ~XT1OFF;      //启动XT1
    P5SEL |= BIT2 + BIT3;    //XT2引脚功能选择
    UCSCTL6 &= ~XT2OFF;      //打开XT2
    __bis_SR_register(SCG0);
    //手动选择DCO频率阶梯(8种阶梯), 确定DCO频率大致范围。
    UCSCTL0 = DCO0+DCO1+DCO2+DCO3+DCO4;
    UCSCTL1 = DCORSEL_4;
    //fDCOCLK/32, 锁频环分频器
    UCSCTL2 = FLLD_5;
    //n=8,FLLREFCLK时钟源为XT2CLK, DCOCLK=D*(N+1)*(FLLREFCLK/n)
    //DCOCLKDIV=(N+1)*(FLLREFCLK/n)
    UCSCTL3 = SELREF_5 + FLLREFDIV_3;
    //ACLK的时钟源为DCOCLKDIV,MCLK\SMCLK的时钟源为DCOCLK
    UCSCTL4 = SELA_4 + SELS_3 + SELM_3;
    //ACLK由DCOCLKDIV的32分频得到, SMCLK由DCOCLK的2分频得到
    UCSCTL5 = DIVA_5 + DIVS_1;
}
```



### 三、详细设计—光敏电阻传感器



怎么测？



使用光敏传感器测量环境光强的变化，并用L1~L6灯来指示当前的光强。

### 硬件接线:

光敏电阻传感器模块VCC-->MSP430F5529口袋版3.3V


光敏电阻传感器模块GND -->MSP430F5529口袋版GND

光敏电阻传感器模块 AO-->MSP430F5529开发板 P6.1

### 编程参考:

ADC实验内容（修改？）

观察效果



光强与输出模拟电压值的关系？

### 三、详细设计—光敏电阻传感器

如何根据光强调节**LED**亮度:

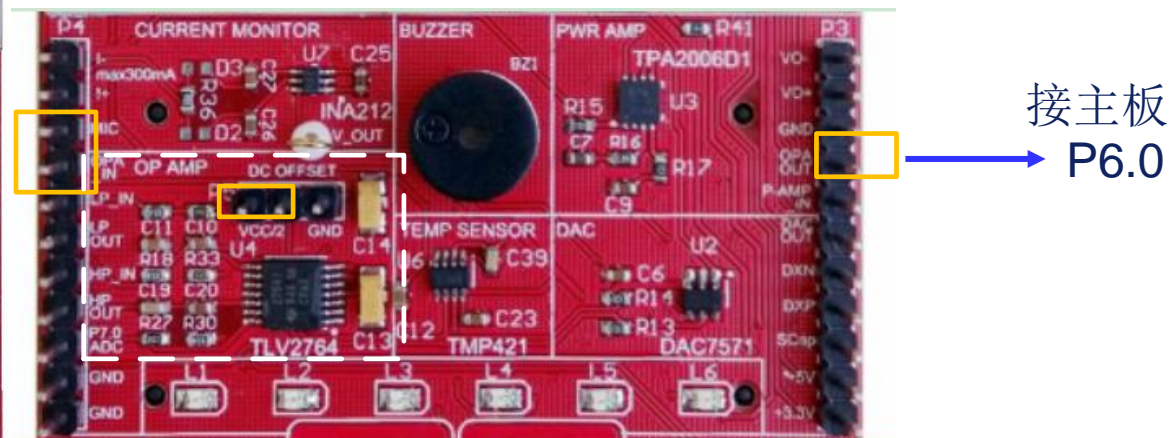
$$\frac{\text{采集到光强的电压值}}{3300} = \frac{\text{一个周期内点亮LED的时间}}{\text{周期}}$$

### 三、详细设计—声音采集模块

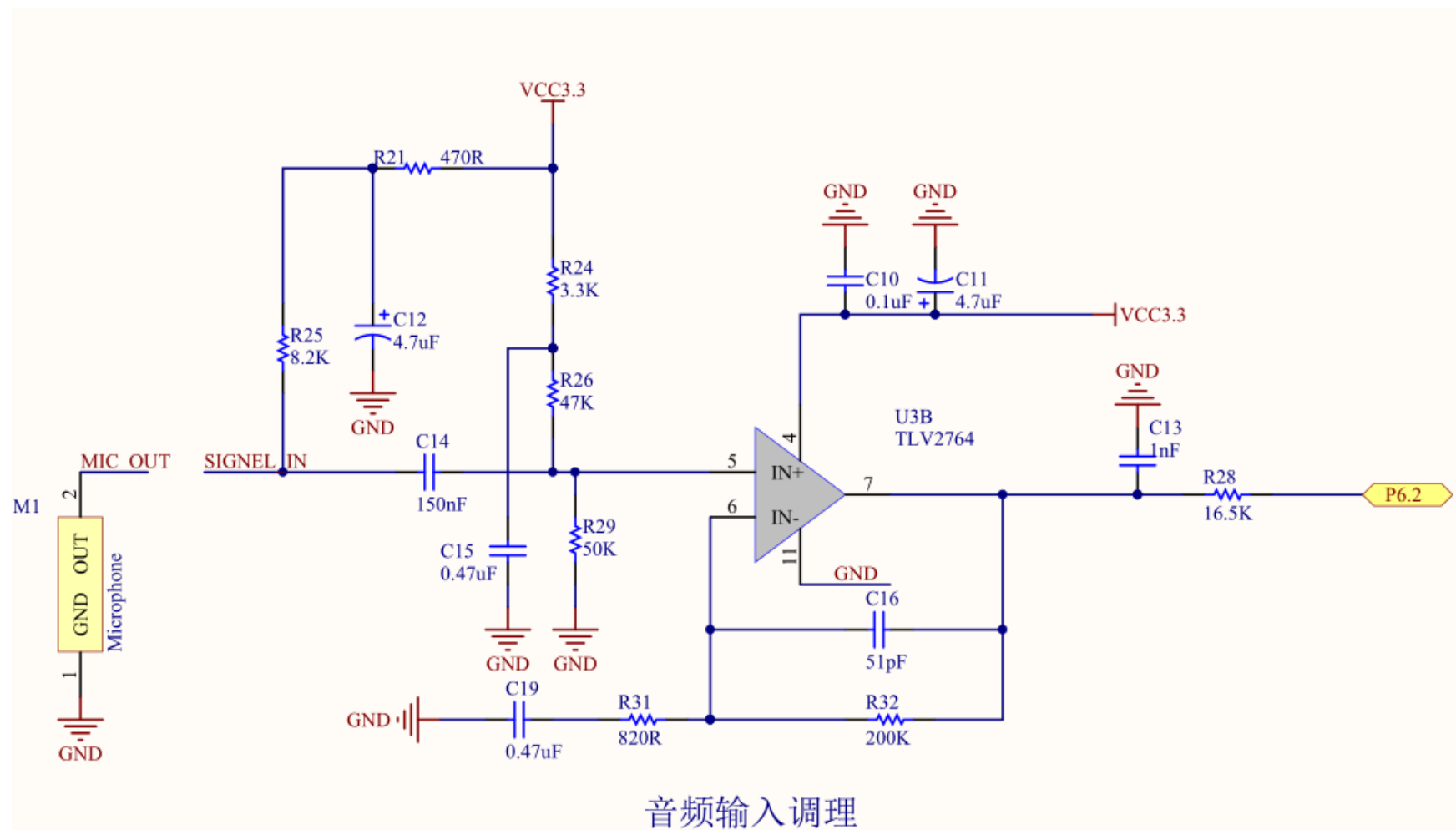


麦克风的输出信号与 TLV2764 运放的输入端连接起来，即将F5529 口袋板左侧MIC与OPA\_IN 用跳线帽短接。

F5529口袋板右侧OPA\_OUT与MSP430F5529LP 端口P6.0用杜邦线连接。



### 三、详细设计—声音采集模块



### 三、详细设计—声音采集模块

```
void InitAD()
{
    ADC12CTL0 |= ADC12MSC;//自动循环采样转换
    ADC12CTL0 |= ADC12ON;//启动ADC12模块
    ADC12CTL1 |= ADC12CONSEQ_3 ;//选择序列通道多次循环采样转换
    ADC12CTL1 |= ADC12SHP; //采样保持模式
    ADC12CTL1 |= ADC12CSTARTADD_0;
    ADC12MCTL0 |=ADC12INCH_0;//通道选择
    ADC12MCTL1 |=ADC12INCH_1;
    ADC12MCTL2 |=ADC12INCH_2;
    ADC12MCTL3 |=ADC12INCH_3;
    ADC12MCTL4 |=ADC12INCH_4;
    ADC12MCTL5 |=ADC12INCH_5+ADC12EOS;
    ADC12CTL0 |= ADC12ENC;
}
```

## 三、详细设计—声音采集模块

### 软件滤波:

ADC转换时, 由于干扰的存在, 转换数据往往出现大幅波动, 影响其转换数据的准确性, 处理在外部增加滤波电路外, 软件滤波也经常用到, 这里以**中位值平均滤波法**为例:

1) **方法:** 相当于“中位值滤波法”+“算术平均滤波法”

连续采样 $N$ 个数据, 去掉一个最大值和一个最小值, 然后计算 $N-2$ 个数据的算术平均值

$N$ 值的选取: 3~14

1) **优点:** 算法简单, 融合了两种滤波法的优点; 对于偶然出现的脉冲性干扰, 可消除由于脉冲干扰所引起的采样值偏差

2) **缺点:** 测量速度较慢, 和算术平均滤波法一样; 比较浪费RAM

### 三、详细设计—声音采集模块

#### 软件滤波:

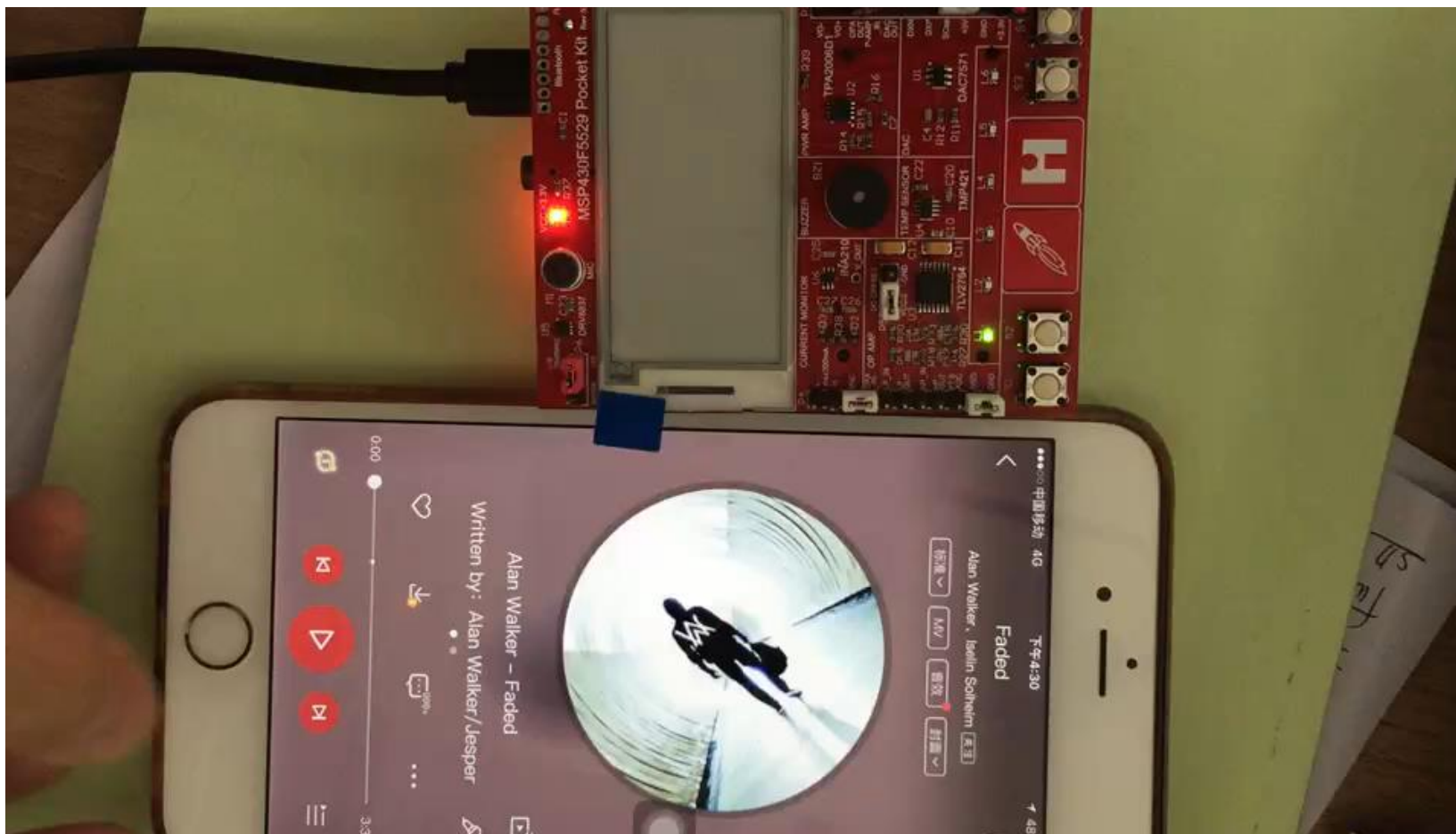
```
#define N 12
char Filter()
{
    char count, i, j;
    char value_buf[N];
    int sum = 0;
    for (count = 0; count < N; count++)
    {
        value_buf[count] = get_ad();
        delay();
    }
    for (j=0;j<N-1;j++)
    {
        for (i=0;i<N-j-1;i++)
        {
            if ( value_buf[i]>value_buf[i+1] )
            {
                temp = value_buf[i];
                value_buf[i] = value_buf[i+1];
                value_buf[i+1] = temp;
            }
        }
    }
    for(count=1;count<N-1;count++)
    sum += value[count];
    return (char)(sum/(N-2));
}
```

对比：有无滤波时的区别

代码中的  
错误?



### 三、详细设计—声音采集模块实验





## 第四部分

# C语言程序编码规范

### 1. 命名基本原则

(1) 命名清晰明了，有明确含义，使用完整单词或约定俗成的缩写。

- 尽量采用采用英文单词或全部中文全拼表示，
- 若出现英文单词和中文混合定义时，使用连字符“\_”将英文与中文割开。
- 较短的单词可通过去掉“元音”形成缩写；较长的单词可取单词的头几个字母形成缩写；一些单词有大家公认的缩写。例如：**temp->tmp、flag->flg、message->msg、statistic->stat、increment->inc**等缩写能够被大家基本认可。

## 四、编程规范—命名规则

- (2) 命名风格要[自始至终保持一致](#)。
- (3) 命名中若使用特殊约定或缩写，要有[注释说明](#)。
- (4) 为了代码复用，命名中应避免使用与具体项目相关的前缀。
- (5) 应尽量使用英语命名。

### 2.预定义(#define)

只使用大写字母，下划线和数字。

例如： `#define MAX_LENGTH 1`

### 3.宏和常量命名

只使用大写字母，下划线和数字。

宏和常量用全部大写字母来命名，词与词之间用下划线分隔。

对程序中用到的数字均应用有意义的枚举或宏来代替。

## 四、编程规范—命名规则

### 4. 变量命名

类型	前缀
指针 ( <b>Pointer</b> )	<b>p</b>
枚举 (Enumeration)	<b>e</b>
结构 (Structure)	<b>st</b>
布尔 ( <b>Boolean</b> )	<b>b</b>
浮点 ( <b>Float</b> )	<b>f</b>
双精度 ( <b>Double</b> )	<b>d</b>
字符 ( <b>Char</b> )	<b>c</b>
其他数字类型, <b>e.g.</b> byte, (unsigned) int, (unsigned)long int, (unsigned)short, (unsigned) short int, (unsigned) long long	<b>n</b>

## 四、编程规范—命名规则

类型	前缀
结构体的成员变量	m_
静态变量	s_
全局变量	g_
局部变量	无前缀

//全局变量

```
Int g_nMaxCount;
```

//局部变量

局部循环体控制变量优先使用**i、j、k**等；

局部长度变量优先使用**len、num**等；

临时中间变量优先使用**temp、tmp**等。

### 5. 函数命名

遵循动宾结构的命名法则，函数名中动词在前,并在命名前加入函数的前缀，函数名的长度不得少于8个字母。

**unsigned char** getNumber(.....);

**unsigned char** get\_number (.....);

### 6.文件命名

一个文件包含一类功能或一个模块的所有函数，文件名称应清楚表明其功能或性质。

每个**.c**文件应该有一个同名的**.h**文件作为头文件。



### 注释基本原则

有助于对程序的阅读理解，说明程序在“做什么”，解释代码的目的、功能和采用的方法。

- 一般情况源程序有效注释量在30%左右。
- 注释语言必须准确、易懂、简洁。
- 边写代码边注释，修改代码同时修改相应的注释，无用的注释要删除。
- C语言编程中用“//”，尽量不使用段注释“/\* \*/”（调试时可用）。

### 1. 文件注释

文件注释必须说明文件名、项目名称、函数功能、创建人、创建日期、版本信息等相关信息。修改文件代码时，应在文件注释中记录修改日期、修改人员，并简要说明此次修改的目的。所有修改记录必须保持完整。文件注释放在文件顶端，用"/\*.....\*/"格式包含。注释文本每行缩进4个空格；每个注释文本分项名称应对齐。

```
/*  
*****  
文件名称:  
项目名称:  
平台信息:  
功能说明:  
作者:  
版本:  
说明:  
修改记录:  
*****  
*/
```

### 2.函数注释

函数头部注释应包括函数名称、函数功能、入口参数、出口参数等内容。如有必要还可增加作者、创建日期、修改记录（备注）等相关项目。函数头部注释放在每个函数的顶端，用"/\*.....\*/"的格式包含。其中函数名称应简写为FunctionName()，不加入、出口参数等信息。

```
/******  
函数名称:  
函数功能:  
入口参数:  
出口参数:  
备 注:  
*****/
```

### 3.代码注释

代码注释应与被注释的代码紧邻，放在其上方或右方，不可放在下面。如放于上方则需与其上面的代码用空行隔开。

一般少量注释应该添加在被注释语句的行尾，一个函数内的多个注释左对齐；较多注释则应加在上方且注释行与被注释的语句左对齐。

通常，分支语句（条件分支、循环语句等）必须编写注释。其程序块结束行“}”的右方应加表明该程序块结束的标记“end of .....”，尤其在多重嵌套时。示例如下：

### 3.代码注释

```
int  nNum = 7;    // 注释。。。
```

```
// 注释。。。
```

```
for(int i = 0 ; i < nNum; i++)
```

```
{
```

```
    if( i == 0)
```

```
    {
```

```
        // code.....
```

```
    }
```

```
    else
```

```
    {
```

```
        //code
```

```
    } //end of if(i == 0)
```

```
} // end of for (i)
```

## 四、编程规范—代码格式

- ◆ 代码的每一级均往右缩进4个空格的位置
- ◆ 不使用Tab键
- ◆ 相对独立的程序块之间要加空行
- ◆ 括号内侧（即左括号后面和右括号前面）不加空格，多重括号间不加空格。如：SetName(GetFunc());
- ◆ 函数形参之间应该有且只有一个空格（形参逗号后面加空格），如：callFunction(para1, para2, para3),  
而 callFunction(para1,para2,para3) 不符合要求。

## 四、编程规范—代码编写要求

- ◆ 操作符前后均加一个空格，如： `nSum = nNum1 + nNum2` 。而`nSum=nNum1+nNum2` 则不符合要求。
- ◆ 单目操作符，如`!"`、`"~"`、`"++"`、`"-"`、`"&"`（地址运算符）等，后面不加空格，如：`i++` ，
- ◆ `if`、`else if`、`else`、`for`、`while`语句无论其执行体是一条语句还是多条语句都必须加花括号，且左右花括号各独占一行。

## 四、编程规范—代码编写要求

### ◆ 条件判断语句中的操作符

当if、else if、else、for、while中的条件判断，  
操作符只可能是 ==、!=、&&、||、<、>、<=、>=。

如：

```
int nNum = nValue1 & nValue2;  
if(nNum == 0)  
{  
    // TODO  
}
```



## 四、编程规范—代码编写要求

### ◆ Switch 语句

Switch 语句必须包含default 分支。如：

```
Switch(nNum)
{
    Case 1:
        Break;
    Case 2:
        Break;
    Default:
        Break;
}
```

一个函数最好不要超过80行代码。

- ◆ 可通过加入临时变量去观察代码执行的中间值或中间状态等

建议调试用的临时变量加上**关键字volatile**，确保不会被编译器优化。

```
volatile unsigned int ledFlag;
```

## 第五部分

# 软件设计说明书

设计说明书内不  
需要粘贴源代码

## 任务：

结合课上实验内容，实现展馆灯光控制系统：该系统主要用于展览馆等需要解说员解说，且需要调节光线以达到最佳演示效果的场合。系统检测到外界声音后打开灯光，系统根据周边环境光的情况自动调整**LED**灯的亮度，以达到最佳展示效果。如果一定时间内都没有检测到声音信号，则自动关闭**LED**灯，以达到节能目的。

## 演示效果：

以小功率**LED**灯**L1~L6**亮灯的数量指示环境光强值。有声音时，打开大功率**LED**灯并进行光线调节，使**小功率LED灯在各种光线条件下维持在恒定的亮灯数量。一定时间没检测到声音**，则关闭大功率**LED**灯。要求大功率**LED**光强平滑改变，不可突变，开环或闭环控制均可。

提交： 演示视频，源代码工程，实验报告

# EXP.5 模拟展厅灯

Sheeper\_Xu

**任务形式：**以小组（一般三人，自由组队）为单位，完成程序编写（按照编程规范）、任务演示（总结课）、编写说明书（按照模板）

**作业提交：**完整代码工程、软件设计说明书（电子版+纸质版）、视频演示

命名形式：19电信1班1组——源程序

19电信1班1组——软件设计说明书

19电信1班1组——视频演示

（注：请学委收集整理，**总结课**时一并提交纸质版和三个电子版文件）

**评分标准：**

- (1) 程序难度：功能完成度、程序代码优化程度
- (2) 表现度：表现形式的多样性、新颖度、演示表现
- (3) 规范性：程序编码规范、说明书编写规范



# Q&A

