Arduino 电位器调光程序详解

本实验目标：电位器控制LED灯光亮度

目标分析：为了完成这个实验，我们首先对于它进行架构的分析

传感器：电位器
控制器：UNO/ESP32
执行器：LED灯

接着是效果分析：

有一个小灯和一个旋钮电位器，用手转动电位器的不同角度，灯光会有相对的，不一样的亮度

明确目标系统与效果，开始正式学习---

代码全貌

```
void setup(){
  pinMode(3, OUTPUT); // 设置3号引脚为输出模式
}

void loop(){
  analogWrite(3, map(analogRead(A0), 0, 1023, 0, 255));
}
```

代码逐行解析

1. `void setup() { ... }`

```
pinMode(3, OUTPUT);
```

- `pinMode`：引脚模式配置函数
  - **参数1**：`3` → 目标引脚编号（支持PWM的引脚）
  - **参数2**：`OUTPUT` → 输出模式
- **硬件要求**：必须使用标有"~"的PWM引脚（UNO的3,5,6,9,10,11）

2. `void loop() { ... }`

```
analogWrite(3, map(analogRead(A0), 0, 1023, 0, 255));
```

分解执行流程：

1. `analogRead(A0)`：
   - 功能：读取A0引脚的模拟电压值
   - 返回值：0（0V）到1023（5V）的整数
   - 技术参数：

输入电压	返回值	分辨率
0V	0	10位
5V	1023	

2. `map()` **函数**：

```
map(value, fromLow, fromHigh, toLow, toHigh)
```

   - 功能：线性映射数值范围
   - 计算过程：

```
输出值 = (value - fromLow) * (toHigh - toLow) / (fromHigh - fromLow) + toLow
```

   - 本例映射：

```
analogRead值：0~1023 → 映射为：0~255
```

3. `analogWrite(3, ...)`：
   - 功能：输出PWM信号控制LED亮度
   - 参数范围：0（全关）到255（全亮）
   - PWM特性：

参数值	占空比	等效电压（5V系统）	亮度等级
0	0%	0V	熄灭
127	50%	2.5V	中等亮度
255	100%	5V	最亮

核心概念详解

1. 模拟信号与数字信号

类型	特征	典型应用	Arduino函数
模拟信号	连续变化值（0-5V）	电位器、光敏电阻	analogRead()
数字信号	离散状态（HIGH/LOW）	开关、LED	digitalWrite()
PWM信号	脉冲宽度调制的数字信号	LED调光、电机调速	analogWrite()

2. 电位器工作原理

```
电位器结构：
      VCC（5V）
       ▲
       |
       └——中间引脚（信号输出）→ A0
       |
       ▼
      GND（0V）

工作特性：
- 旋钮转动改变电阻值
- 输出电压在0-5V之间线性变化
- 对应analogRead()返回0-1023
```

3. PWM调光原理

- **脉宽调制**：通过快速开关产生等效电压
- **频率特性**：Arduino UNO默认PWM频率≈490Hz
- **视觉暂留**：当频率>24Hz时，人眼感知为连续亮度变化

硬件连接规范

标准电路连接

```
电位器：
  左侧引脚 → 5V
  中间引脚 → A0
  右侧引脚 → GND

LED电路：
  引脚3 → 220Ω电阻 → LED正极
  LED负极 → GND
```

元件参数选择

元件	参数要求	推荐型号
电位器	10kΩ线性电位器	B10K
限流电阻	220Ω（红色LED）	1/4W碳膜电阻
LED	通用5mm发光二极管	红/黄/绿LED

关键问题解析

Q1：LED亮度变化不线性

- **原因分析**：
  1. 电位器非线性（应选用线性电位器B型）
  2. 人眼对亮度的对数感知特性
- **解决方案**：

```
// 伽马校正公式
float gamma = 2.8;//这里定义了伽马值，为2.8，伽马值是伽马校正中的一个重要参数，不同的伽马值会产生不同的校正效果。通常，伽马值大于 1 时会使图像或灯光的亮部更亮，暗部更暗；伽马值小于 1 时则相反。
int corrected = pow((value/255.0), gamma) * 255;//执行伽马校正
analogWrite(3, corrected);//输出校正后的值
```

Q2：数值抖动问题

- **现象**：LED亮度轻微波动
- **解决方法**：

```
// 添加软件滤波
int filter(int pin) {
  int sum = 0;
  for(int i=0; i<10; i++){
    sum += analogRead(pin);
    delay(1);
  }
  return sum/10;
}
```

Q3：如何扩展多级控制

```
void loop(){
  int val = map(analogRead(A0), 0, 1023, 0, 255);

  // 分级控制
  if(val < 50)       analogWrite(3, 0);
  else if(val < 150) analogWrite(3, 80);
  else if(val < 200) analogWrite(3, 160);
  else               analogWrite(3, 255);
}
```

扩展实验

实验1：串口亮度监测//串口在后面学习（第九文）

```
void setup(){
  Serial.begin(9600);
  pinMode(3, OUTPUT);
}

void loop(){
  int raw = analogRead(A0);
  int pwm = map(raw, 0, 1023, 0, 255);
  analogWrite(3, pwm);

  Serial.print("Raw: ");
  Serial.print(raw);
  Serial.print(" → PWM: ");
  Serial.println(pwm);
  delay(100);
}
```

实验2：光控-手控双模式//需光敏电阻

```
void loop(){
  int potValue = analogRead(A0);
  int lightValue = analogRead(A1); // 光敏电阻接A1

  // 模式切换开关接D2
  if(digitalRead(2) == HIGH){
    analogWrite(3, map(potValue, 0, 1023, 0, 255));
  } else {
    analogWrite(3, map(lightValue, 0, 1023, 255, 0));
  }
}
```

实验3：渐变呼吸灯改进

```
void loop(){
  int target = map(analogRead(A0), 0, 1023, 0, 255);
  int current = analogRead(3); // 实际需通过其他方式获取当前亮度

  // 平滑过渡
  if(current < target) current++;
  else if(current > target) current--;

  analogWrite(3, current);
  delay(10);
}
```

总结

本程序通过电位器实现LED无级调光，核心知识点包括：

1. **模拟信号采集**：掌握analogRead()的使用方法
2. **数值映射**：理解map()函数的线性转换原理
3. **PWM控制**：熟悉analogWrite()的占空比调节机制
4. **电路设计**：正确连接电位器与LED电路

建议使用示波器观察PWM波形变化，直观理解占空比与亮度的关系。

有问题找柯萌ʕ•ᴥ•ʔ
mingyoufhh@outlook.com