Arduino 按键控制LED状态切换教程

本实验目标：完成一个按钮控制灯的亮灭程序,小小的升级

目标分析：为了完成这个实验，我们首先对于它进行系统化的分析

传感器：按钮
控制器：UNO/ESP32
执行器：LED灯

接着是效果分析

有一个按钮，有一个小灯，当按钮按一下时灯亮，再按一下时灯灭，也就是说按钮可以切换小灯工作状态

明确目标系统，开始正式学习---

代码全文

```
void setup() {                // 初始化函数
  pinMode(12, INPUT_PULLUP); // 设置12号引脚为带上拉电阻的输入模式
  pinMode(13, OUTPUT);        // 设置13号引脚为输出模式
}

void loop() {                      // 重复执行函数
  if (digitalRead(12) == LOW) {  // 检测按钮是否被按下
    digitalWrite(13, !digitalRead(13)); // 翻转LED状态
    delay(300);                    // 防抖动延迟
  }
}
```

代码逐行解析

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

- **功能**：Arduino启动时执行一次的初始化设置
- **详细说明**：
  - `pinMode(12, INPUT_PULLUP)`：
    - `pinMode`：设置引脚模式的函数
    - `12`：目标引脚编号（物理接口位置）
    - `INPUT_PULLUP`：输入模式并启用内部上拉电阻
  - `pinMode(13, OUTPUT)`：
    - `OUTPUT`：输出模式（用于控制LED等设备）

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

- **功能**：循环执行的主程序逻辑

关键代码解析：

行1： `if (digitalRead(12) == LOW) {`

- `digitalRead(12)`：读取12号引脚的电压状态
  - 返回值为`HIGH`（5V）或`LOW`（0V）
- `== LOW`：判断条件是否成立
  - 当按钮按下时（正确接线情况下），引脚电压会被拉低至0V

行2： `digitalWrite(13, !digitalRead(13));`

- `digitalRead(13)`：读取当前LED引脚状态
- `!`**运算符**：逻辑非，将`HIGH`转为`LOW`，`LOW`转为`HIGH`
- **综合作用**：每次执行时翻转LED的亮灭状态

行3： `delay(300);`

- **作用**：产生300ms延时，防止按钮抖动导致多次状态切换
- **单位**：毫秒（1秒=1000毫秒）

核心概念详解

1. 上拉输入模式（INPUT_PULLUP）

特性	说明
内部电阻值	约20kΩ（不同开发板可能略有差异）
默认状态	未按下按钮时引脚电压为HIGH（5V）
按钮按下状态	引脚通过按钮接地，电压变为LOW（0V）
优势	无需外接电阻，简化电路设计

2. 状态翻转逻辑

```
digitalWrite(13, !digitalRead(13));
```

- **执行过程**：
  1. 读取13号引脚当前状态（HIGH/LOW）
  2. 使用`!`运算符取反// ！的意思是非，比如说现在是高，取！后变为低，现在是低取！后变为高
  3. 将新状态写回引脚 //当你高时给你低，当你低时给你高，这就是控制原理
- **等效代码**：

```
if(digitalRead(13) == HIGH) {//判断是否为高
  digitalWrite(13, LOW);     //是高则给你低
} else {                     //否则
  digitalWrite(13, HIGH);    //给你高
}
```

3. 按钮抖动现象

现象	解决方法
机械触点振动	硬件：并联0.1μF电容
产生多次电平变化	软件：延时检测（本代码使用300ms）
误触发	状态锁定：等待按钮释放

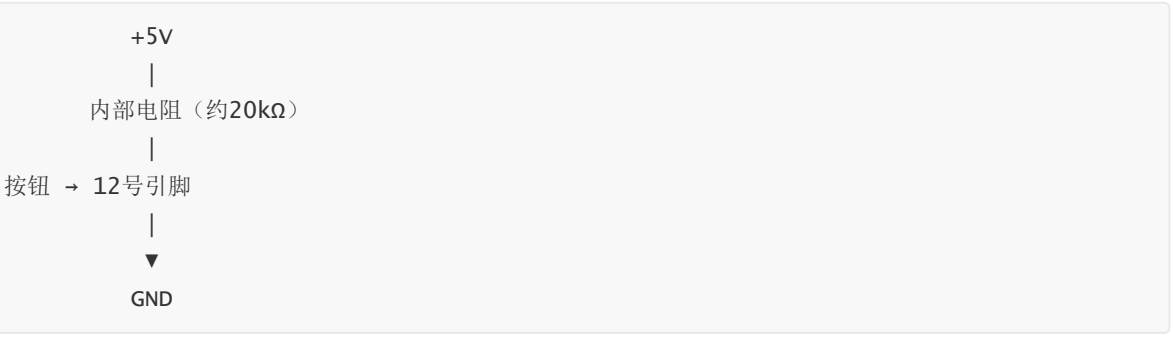
硬件连接说明

1. 推荐电路

```
12号引脚 → 按钮 → GND
13号引脚 → 220Ω电阻 → LED正极 → LED负极 → GND
```

- **注意**：无需外接上拉电阻，使用 `INPUT_PULLUP` 模式时Arduino内部已集成

2. 内部上拉等效电路



关键问题解析

Q1：为什么按下按钮LED会快速闪烁?

- **原因**：未正确处理按钮抖动和状态锁定
- **改进方案**：

```
void loop() {
  if (digitalRead(12) == LOW) {
    digitalWrite(13, !digitalRead(13));
    while(digitalRead(12) == LOW); // 等待按钮释放
    delay(50); // 释放抖动处理
  }
}
```

Q2：`!digitalRead(13)`如何工作？

- **真值表**：

当前状态	!运算结果	执行动作
HIGH	LOW	熄灭LED
LOW	HIGH	点亮LED

Q3：如何调整状态切换速度？

- **修改延时参数**：

```
delay(100);  // 缩短至100ms（更灵敏但可能误触发）
delay(500);  // 延长至500ms（更稳定但响应较慢）
```

扩展实验

实验1：三态切换控制

```
int state = 0;

void loop() {
  if (digitalRead(12) == LOW) {
    state = (state + 1) % 3;
    switch(state) {
      case 0: digitalWrite(13, LOW); break;
      case 1: digitalWrite(13, HIGH); break;
      case 2: // 闪烁模式
        for(int i=0; i<5; i++){
          digitalWrite(13, !digitalRead(13));
          delay(200);
        }
    }
    delay(300); // 模式切换防抖
  }
}
```

实验2：多LED控制

```
void setup() {
  pinMode(12, INPUT_PULLUP);
  for(int i=8; i<=13; i++) { // 控制8-13号引脚
    pinMode(i, OUTPUT);
  }
}

void loop() {
  if (digitalRead(12) == LOW) {
    static int current = 8;
    digitalWrite(current, LOW);  // 关闭当前LED
    current = (current < 13) ? current+1 : 8;
    digitalWrite(current, HIGH); // 点亮下一个LED
    delay(300);
  }
}
```

总结

本程序实现了通过带内部上拉的按钮控制LED状态切换，涉及以下核心知识点：

1. **输入上拉模式**：理解内部电阻的作用和接线方式
2. **状态翻转逻辑**：掌握`!`运算符的实际应用
3. **防抖动处理**：认识按钮机械特性对电路的影响
4. **数字信号处理**：学习数字输入/输出的基本控制方法

通过修改延时参数和扩展逻辑，可以实现更复杂的控制功能。

有问题找柯萌_:(´□`」 ∠):_
mingyoufhh@outlook.com