工程实践与科技创新 I

单片机项目

实验报告

项目名称 简易饮水机功能模拟设计

姓名 (524031910000)

(524031910000)

任课老师 崔萌

完成时间 2025 年 4 月 25 日

目录

1　实验目的

- 了解状态机原理；
- 运用状态机思想进行程序框架的设计与编写；
- 掌握在无操作系统支持的条件下，模拟多任务切换式处理的程序设计思想和方法。

2　实验主要器材和设备

电脑、MSP-EXP430F5529 LaunchPad 实验板卡。

3　实验任务的技术解决方案说明

为实现饮水机的功能控制，采用基于状态机的设计方法，结合定时中断机制，实现对童锁与热水键的交互逻辑控制。饮水机工作状态可抽象为三个主要状态：

- **锁定状态 (LOCKED)**：红色 LED 亮，按下热水键无效；
- **解锁状态 (UNLOCKED)**：红灯灭，用户可在 3 秒内按热水键；
- **放水状态 (DISPENSING)**：绿色 LED 亮，按一次热水键后开始放水，再按一次停止放水并自动上锁。

当前状态	输入事件	状态转移结果	说明
LOCKED	童锁键长按 ≥3 秒	UNLOCKED	解锁成功，红灯熄灭
	热水键按下/松开	LOCKED	无效操作，保持锁定
UNLOCKED	热水键松开	DISPENSING	进入放水状态，绿灯亮
	3 秒内未按热水键	LOCKED	超时回锁，红灯亮
	热水键长按超 3 秒但未松开	UNLOCKED	无变化，仍保持解锁状态
DISPENSING	热水键松开	LOCKED	结束放水，自动上锁，红灯亮
	童锁键按下	DISPENSING	无效操作，维持当前状态

表 1: 状态转移表格

为实现按键长按识别和 3 秒超时逻辑，程序使用中断服务程序（每 20ms 触发一次），通过定义软件计数器判断“长按 3 秒”和“3 秒未操作”等事件。定义变量：

- `COUNTER_3S = 150`：表示 150 次 20 ms 计时中断 = 3 秒；
- `lockCounter`：用于记录童锁键的长按时间；
- `unlockTimeout`：用于检测解锁状态下是否 3 秒未操作。

所有按键使用上拉输入模式（`PULLUP`），松开检测时通过记录上一轮按键状态，并在当前轮检测到“从 `LOW` 到 `HIGH`”的跳变，从而判断“松开动作”。由此，通过设计三个状态的互相转换实现了饮水机的全部功能。

4　实验结果自测记录

按键	功能	测试操作	测试现象	结论
童锁	平时童锁按键处于锁定状态，长按童锁按键 3 秒后进入解锁状态	红色 LED 点亮、绿色 LED 熄灭时，长按 PUSH1	按下按键 3 秒后，红色 LED 熄灭，绿色 LED 保持熄灭状态	正常
	如果童锁按键处于解锁状态，3 秒内没有按下热水按键，回到锁定状态	红色 LED 熄灭、绿色 LED 熄灭时，不做操作	3 秒内，红色 LED 亮起，绿色 LED 保持熄灭状态	正常
热水	童锁按键处于锁定状态时，按下热水按键不起作用	红色 LED 点亮、绿色 LED 熄灭时，短按PUSH2	红色 LED 保持点亮状态，绿色 LED 保持熄灭状态	正常
	童锁按键处于解锁状态时，按热水按键（按键释放有效）进入放水状态，开始放水	红色 LED 熄灭、绿色 LED 熄灭时，短按 PUSH2	PUSH2 释放时，红色 LED 保持熄灭状态，绿色 LED 亮起	正常
	处于放水状态时，再次按热水按键（按键释放有效），停止放热水，童锁按键回到锁定状态	红色 LED 熄灭、绿色 LED 点亮时，短按 PUSH2	PUSH2 释放时，红色 LED 点亮，绿色 LED 熄灭	正常
	处于放水状态时，按童锁按键不起作用。	红色 LED 熄灭、绿色 LED 点亮时，短按 PUSH1	红色 LED 保持熄灭状态，绿色 LED 保持点亮状态	正常
	处于解锁状态时，长按热水按键超过 3 秒不放，不会自动回到锁定状态，按童锁按键也不起作用。	红色 LED 熄灭、绿色 LED 熄灭时，长按 PUSH2 超过 3 秒	PUSH2 未释放时，红色 LED 保持熄灭、绿色 LED 保持熄灭；PUSH2 释放时，红色 LED 保持熄灭，绿色 LED 点亮	正常

表 2: 测试操作及实验结果

5　实验核心代码清单

5.1　初始化函数

该部分导入头文件、初始化变量并设定硬件定时器。

```
#include "Timer.h"        // 导入头文件
#define LOCKED 1          // 定义状态机状态
#define UNLOCKED 2
#define DISPENSING 3

const int COUNTER_3S = 150; // 150 * 20 = 3000ms
int lockCounter = 0;        // 锁定状态计数器
int unlockTimeout = 0;      // 解锁状态计数器
```

```
bool lastPush1 = HIGH;       // 按键消抖
bool lastPush2 = HIGH;
byte state = LOCKED;         // 状态机状态，初始状态为锁定

void setup(){
    pinMode(RED_LED, OUTPUT);
    pinMode(GREEN_LED, OUTPUT);
    pinMode(PUSH1, INPUT_PULLUP);
    pinMode(PUSH2, INPUT_PULLUP);
    SetTimer(isrTimer, 20);          // 设定硬件计时器的周期为 20 ms
}
void loop(){}
```

5.2 状态机函数

该部分读取按键状态并执行状态机函数。

```
void isrTimer(){
    bool push1 = digitalRead(PUSH1); // 读取两个按键的状态
    bool push2 = digitalRead(PUSH2);

    switch(state){
        case LOCKED:
            digitalWrite(RED_LED, HIGH);
            digitalWrite(GREEN_LED, LOW);
            if(push1 == LOW){
                lockCounter++;
                // 检测 PUSH1 是否长按并执行相应函数
                if(lockCounter >= COUNTER_3S){
                    state = UNLOCKED;
                    lockCounter = 0;
                }
            }else{
                lockCounter = 0;
            }
            break;

        case UNLOCKED:
            digitalWrite(RED_LED, LOW);
            digitalWrite(GREEN_LED, LOW);
            if(push2 == HIGH){
                // 释放按键后放水
                if(lastPush2 == LOW){
```

```
                    state = DISPENSING;
                }
                unlockTimeout++;
                // 3s 内未按下热水按键
                if(unlockTimeout >= COUNTER_3S){
                    state = LOCKED;
                    unlockTimeout = 0;
                }
            }else if(push2 == LOW){
                unlockTimeout = 0;
            }
            break;

        case DISPENSING:
            // 开始放水
            digitalWrite(RED_LED, LOW);
            digitalWrite(GREEN_LED, HIGH);
            // 结束放水后回到锁定状态
            if(push2 == HIGH && lastPush2 == LOW){
                state = LOCKED;
            }
            break;
    }
    lastPush2 = push2;
    lastPush1 = push1; // 存储历史按键状态
}
```

附录　学习心得和意见建议

在本次综合实验中，我们深入学习并实践了状态机 (State Machine) 思想在嵌入式系统中的应用。本实验让我认识到，将系统行为分解为状态、事件、动作三要素，不仅有助于理清程序逻辑，还能提升程序的可维护性和可扩展性。

学到的知识中令人印象深刻的是状态机设计方法论。我们可以利用状态转移图与表格化分析，能够更加系统性地规划程序框架。例如“童锁解锁—放水—复位”的状态流程，从逻辑图出发再落地为代码，大大降低了调试难度。事实上，这一思想不仅在嵌入式开发中得到广泛应用，在软件设计与编程中也有不少体现。

同时，在代码规范性与结构思维方面，本实验进一步培养了我良好的编程习惯，如简洁的中断服务程序编写等。尽管上学期使用过 STM32F103C8T6 单片机进行过一些研究，但是系统性的教学还是让我们茅塞顿开。

在完成实验过程中，也有几点个人的思考和建议。状态机和状态转移机制是本实验的难点所在，若能够提供或者介绍一个图形化的状态机设计平台将更有利于学习与理解。同时，实验中应该增添一些给学生自由发挥的空间，让同学们自己的想法（功能设计）能够在实验作品中有所体现。另外，实验报告在

保持其规范的基础上不应该仅限于 Word 格式：一些其他的工具，如 LaTeX 等也能够生成质量较高的实验报告。

总而言之，这次综合实验让我们都学到了许多有益于日后专业学习的宝贵思想，这门课程的设置也比较完善贴心。倘若能够给与学生更多创新空间，相信能使学生受益更多。