计算机科学与技术学院

嵌入式系统实验报告

（七）

姓　　名：　Banban

专　　业：　计 算 机 科 学 与 技 术

班　　级：

学　　号：

指 导 教 师：

2023 年 4 月 19 日

一、任务要求

1、 跑通并理解 Timer-INT 项目

2、 模拟交通灯：使用 2 组 4 个(或 6 个)LED 灯表示十字路口的交通灯，自行设定红绿灯切换时间和切换效果，通过定时器中断加以实现。

二、实验报告要求

1、 任务 2 中自编程序的源代码（加上注释）

2、 能说明软件仿真结果的截图、反映硬件电路连接和硬件验证结果的图片或视频

三、实验过程

一. 任务一：跑通并理解 Timer-INT 项目

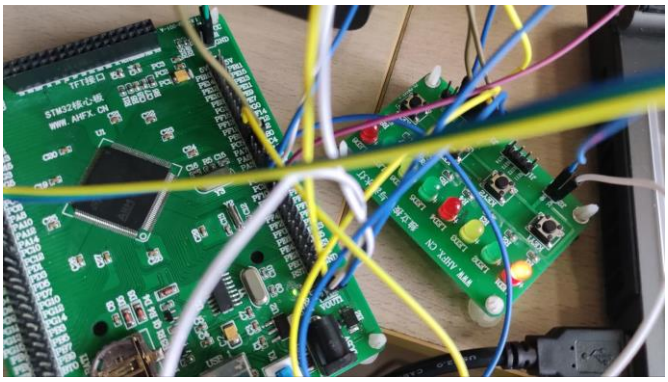
1. 代码：

```
// main.c
// -----------------------------------------------------------
volatile u32 time; // ms 计时变量
// LED 配置
void LED_GPIO_Config(void) {
    GPIO_InitTypeDef GPIO_InitStructure;
    RCC_APB2PeriphClockCmd( RCC_APB2Periph_GPIOC, ENABLE);
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_3 | GPIO_Pin_4 | GPIO_Pin_5;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_Init(GPIOC, &GPIO_InitStructure);
    GPIO_SetBits(GPIOC, GPIO_Pin_3 | GPIO_Pin_4 | GPIO_Pin_5); // 关所有灯
}

// TIM NVIC 设置
void TIM2_NVIC_Configuration(void) {
```

```
    NVIC_InitTypeDef NVIC_InitStructure;
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_0);
    NVIC_InitStructure.NVIC_IRQChannel = TIM2_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 3;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
}

// TIM_Period--1000   TIM_Prescaler--71 -->中断周期 1ms
void TIM2_Configuration(void) {
    TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure;
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);
    TIM_DeInit(TIM2);
    TIM_TimeBaseStructure.TIM_Period = 1000; // 自动重装载寄存器周期的值(计数
值)
    // 累计 TIM Period 个频率后产生一个更新或者中断
    TIM_TimeBaseStructure.TIM_Prescaler = (72 - 1);  // 时钟预分频数 72M/72
    TIM_TimeBaseStructure.TIM_ClockDivision = TIM_CKD_DIV1;  // 采样分频
    TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;  // 向上计数
模式
    TIM_TimeBaseInit(TIM2, &TIM_TimeBaseStructure);
    TIM_ClearFlag(TIM2, TIM_FLAG_Update);  // 清除溢出中断标志
    TIM_ITConfig(TIM2, TIM_IT_Update, ENABLE);
    TIM_Cmd(TIM2, ENABLE);
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, DISABLE);  // 先关闭等待使用
}
int main(void) {
    SystemInit(); // 配置系统时钟为 72M
    LED_GPIO_Config(); // led 端口配置
    TIM2_NVIC_Configuration();// TIM2 定时配置
    TIM2_Configuration();// TIM_Period-1000 TIM_Prescaler-71 -->中断周期 1ms
    START_TIME; // TIM2 开始计时
    while(1) {
    if ( time == 1000 ) {  // 1s 时间到
```

```
      time = 0;
        // LED1 取反
        GPIO_WriteBit(GPIOC, GPIO_Pin_3,
           (BitAction)((1-GPIO_ReadOutputDataBit(GPIOC, GPIO_Pin_3))));
    }
  }
}
```

2. 图片效果



二. 任务二：模拟交通灯：使用 2 组 4 个(或 6 个)LED 灯表示十字路口的交通灯，自行设定红绿灯切换时间和切换效果，通过定时器中断加以实现

1. 代码：

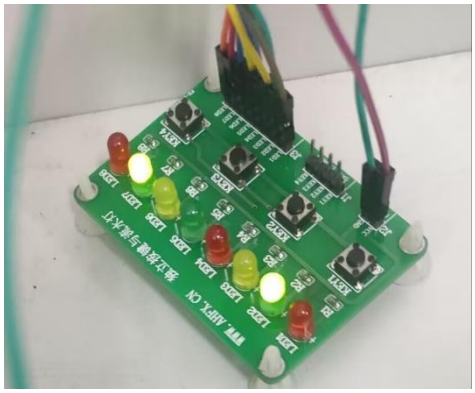
```
// main.c
// ------------------------------------------------------------
// 初始化并启动了 TIM2 定时器。该定时器可用于测量时间间隔或生成周期性中断
#define START_TIME  time=0;RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2 ,
ENABLE);TIM_Cmd(TIM2, ENABLE)

volatile u32 time; // ms 计时变量
// LED 配置
```

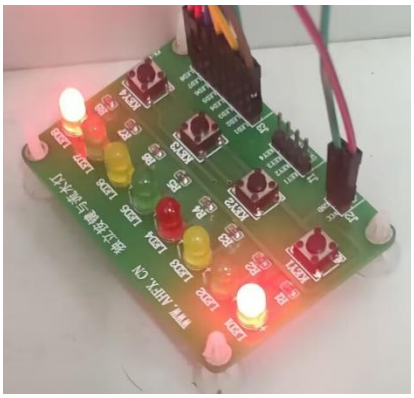
```
void LED_GPIO_Config(void) {
    GPIO_InitTypeDef GPIO_InitStructure;
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOD, ENABLE);
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_1 | GPIO_Pin_2 | GPIO_Pin_3 |
    GPIO_Pin_6 | GPIO_Pin_7 | GPIO_Pin_8;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
    GPIO_Init(GPIOD, &GPIO_InitStructure);
    GPIO_ResetBits(GPIOD, GPIO_Pin_1 | GPIO_Pin_2 | GPIO_Pin_3 | GPIO_Pin_6
    | GPIO_Pin_7 | GPIO_Pin_8;
}

// TIM NVIC 设置
void TIM2_NVIC_Configuration(void) {
    NVIC_InitTypeDef NVIC_InitStructure;
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_0);
    NVIC_InitStructure.NVIC_IRQChannel = TIM2_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 3;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
}

// TIM_Period--1000   TIM_Prescaler--71 -->中断周期 1ms
void TIM2_Configuration(void) {
    TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure;
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);
    TIM_DeInit(TIM2);
    TIM_TimeBaseStructure.TIM_Period = 1000; // 自动重装载寄存器周期的值(计数
值)
    // 累计 TIM Period 个频率后产生一个更新或者中断
    TIM_TimeBaseStructure.TIM_Prescaler = (72 - 1);  // 时钟预分频数 72M/72
    TIM_TimeBaseStructure.TIM_ClockDivision = TIM_CKD_DIV1;  // 采样分频
    TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;  // 向上计数
模式
```

```
    TIM_TimeBaseInit(TIM2, &TIM_TimeBaseStructure);
    TIM_ClearFlag(TIM2, TIM_FLAG_Update);  // 清除溢出中断标志
    TIM_ITConfig(TIM2, TIM_IT_Update, ENABLE);
    TIM_Cmd(TIM2, ENABLE);
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, DISABLE);  // 先关闭等待使用
}

int main(void) {
    SystemInit(); // 配置系统时钟为 72M
    LED_GPIO_Config(); // led 端口配置
    TIM2_NVIC_Configuration(); // TIM2 定时配置
    TIM2_Configuration();// TIM_Period-1000 TIM_Prescaler-71 -->中断周期 1ms
    START_TIME; // TIM2 开始计时
    while(1) {
        GPIOD->ODR^=GPIO_Pin_1;   // 红灯亮
        GPIOD->ODR^=GPIO_Pin_8;

        while(time!=3000); time = 0;

        GPIOD->ODR^=GPIO_Pin_1;  // 红灯灭
        GPIOD->ODR^=GPIO_Pin_8;

        GPIOD->ODR^=GPIO_Pin_2;  // 绿灯亮
        GPIOD->ODR^=GPIO_Pin_7;

        while(time!=3000); time = 0;

        GPIOD->ODR^=GPIO_Pin_2; // 绿灯灭
        GPIOD->ODR^=GPIO_Pin_7;

        for(i = 0; i < 6; i++) {
            GPIOD->ODR^=GPIO_Pin_3;  // 黄灯闪烁
            GPIOD->ODR^=GPIO_Pin_6;
            while(time!=800); time = 0;
        }
    }
}
```

2. 图片效果

描述：红灯亮 3s 后熄灭；绿灯亮起，3s 后，绿灯熄灭；黄灯闪烁，再红灯亮起。





四、总结与分析

在本次实验中，我成功地跑通了 Timer-INT 项目，并且通过模拟交通灯的方式，学习并掌握了如何使用定时器中断来实现红绿灯切换的方法。

首先，在进行实验前，我仔细阅读了相关的资料和代码，并对 STM32 芯片的开发板进行了初始化。在这个过程中，我学习到了如何配置 TIM2_NVIC 和 TIM2 等参数，以便于在定时器中断触发时切换 LED 灯的状态。

然后，我开始进行模拟交通灯的实验。我使用了两组六个 LED 灯来表示十字路口的交通灯，并自行设定了红绿灯切换时间和切换效果。通过定时器中断的设置，我成功地实现了交通灯的红绿灯切换功能，并且通过调试和测试，验证了它的正确性和稳定性。

通过这次实验，我学会了如何使用定时器中断来实现交通灯的红绿灯切换，并且深入理解了其中的原理和应用方法。此外，通过自己动手实践的过程，我也提高了自己的实验能力和代码设计水平。