

# 实验报告

PB21511897      李霄奕

## 实验 1

变量名	数据类型	变量值	寄存器	内容
ui_a	unsigned int	1	r5	0x01
ui_b	unsigned int	2	r6	0x02
ui_c	unsigned int	0xFF	r7	0xFF
寄存器标志位			x-PSR^N	1
i_tmp	static int	-1	r9	0xFFFFFFFF
s16_tmp	static float	-32768	r0	0xFFFF8000
f_tmp	static float	-0.5	s0	0xBF000000
		0	FAULT	0
		0x00	BASEPRI	0x00
		0	PRIMASK	0
		0x04	CONTORL	0x04
			xPSR	0x61000000
			MSP	0x20000674

## 实验 2

位带映射公式：

```
#define BITBAND(addr,bitnum)
((addr&0xF0000000)+0x20000000+((addr&0xFFFF)<<5)+(bitnum<<2))
```

内存中关于 GPIO 的地址分配如下：

```
#define PERIPH_BASE ((uint32_t)0x40000000)
#define AHB1PERIPH_BASE (PERIPH_BASE + 0x00020000)
#define GPIOA_BASE (AHB1PERIPH_BASE + 0x0000)
#define GPIOA_ODR_Addr (GPIOA_BASE+20)
#define GPIOA_IDR_Addr (GPIOA_BASE+16)
#define PAout(n) BIT_ADDR(GPIOA_ODR_Addr,n)
```

由此可知，GPIOA\_ODR\_Addr==0x40000000+0x00020000+0x0000+20==0x40020014

则有：BITBAND(GPIOA\_ODR\_Addr,0)==0x42400280;

BITBAND(GPIOA\_ODR\_Addr,1)==0x42400284;

BITBAND(GPIOA\_ODR\_Addr,7)==0x4240029C;

因此能够得出，位带操作扩充了 32 倍

## 综合实验

### 1. 功能描述

利用定时器生成 PWM 波，并在 LCD 显示占空比。

### 2. 功能模块划分

定时器模块、中断服务模块、PWM 波模块、LCD 显示模块

### 3. 系统结构图



### 4. 实现功能的核心代码

#### A) main 函数：实现主要逻辑

```
int main(){
    uint32_t pwm=0;
    uint8_t data[7]={0};
    delay_init(168);
    LCD_GPIO_Init();
    LCD_Init();
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);
    TIM2_PWM_Init(3000,0);
    LCD_Clear();//初始化，启动相关时钟、优先级中断
    while(1)
    {
        //循环实现 PWM 波的呼吸灯效果
        for(pwm=0;pwm<3000;pwm+=5){
            delay_ms(1);
            TIM_SetCompare3(TIM2, pwm);
            update(data,pwm);//根据 pwm 的值计算要显示的占空比内容
            LCD_Display_Words(0,0,data);
        }
        for(pwm=3000;pwm>0;pwm-=5){
            delay_ms(1);
            TIM_SetCompare3(TIM2, pwm);
            update(data,pwm);
            LCD_Display_Words(0,0,data);
        }
    }
}
```

#### B) update 函数：输入 pwm 值，计算占空比，将结果以百分比的形式输出到字符串上

```
void update(uint8_t* data,uint32_t pwmm){//利用整型变量乘除的性质，逐位计算显示值
    data[0]=(pwmm*10/3000)%10+48;
```

```

    data[1]=(pwmm*100/3000)%10+48;
    data[2]='.';
    data[3]=(pwmm*1000/3000)%10+48;
    data[4]=(pwmm*10000/3000)%10+48;
    data[5]='%';
    data[6]=0;
}

```

#### C) 初始化 LCD 的 GPIO 接口

```

void LCD_GPIO_Init(){
    GPIO_InitTypeDef GPIO_InitStructure;    //GPIO 初始化结构体
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOG|RCC_AHB1Periph_GPIOF, ENABLE);// 使
    能 GPIOG、GPIOF 时钟

```

```

        GPIO_InitStructure.GPIO_Pin = GPIO_Pin_1;//初始化 G 引脚 1
        GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
        GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
        GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
        GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
        GPIO_Init(GPIOG, &GPIO_InitStructure);
        GPIO_ResetBits(GPIOG,GPIO_Pin_1);

```

```

        GPIO_InitStructure.GPIO_Pin = GPIO_Pin_14 | GPIO_Pin_15;//初始化 F 引脚 14、15
        GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
        GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
        GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
        GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
        GPIO_Init(GPIOF, &GPIO_InitStructure);
        GPIO_ResetBits(GPIOF,GPIO_Pin_14 | GPIO_Pin_15);
        CS=1;
        SID=1;
        SCLK=1;
}

```

#### D) 初始化中断函数 TIM2

```

void TIM2_PWM_Init(u32 arr, u32 psc)
{
    GPIO_InitTypeDef GPIO_InitStructure;
    TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure;
    TIM_OCInitTypeDef TIM_OCInitStructure;
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2,ENABLE);
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);
    GPIO_PinAFConfig(GPIOB,GPIO_PinSource10,GPIO_AF_TIM2);
    GPIO_PinAFConfig(GPIOB,GPIO_PinSource10,GPIO_AF_TIM2);
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;

```

```
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
GPIO_Init(GPIOB,&GPIO_InitStructure);
TIM_TimeBaseStructure.TIM_Prescaler=psc;
TIM_TimeBaseStructure.TIM_CounterMode=TIM_CounterMode_Up;
TIM_TimeBaseStructure.TIM_Period=arr;
TIM_TimeBaseStructure.TIM_ClockDivision=TIM_CKD_DIV1;
TIM_TimeBaseInit(TIM2,&TIM_TimeBaseStructure);
TIM_OCInitStructure.TIM_OCMode = TIM_OCMode_PWM1;
TIM_OCInitStructure.TIM_OutputState = TIM_OutputState_Enable;
TIM_OCInitStructure.TIM_OCPolarity = TIM_OCPolarity_High ;
TIM_OC3Init(TIM2, &TIM_OCInitStructure);
TIM_Cmd(TIM2, ENABLE);
}
```