计算机原理与嵌入式系统实验报告

微电子学院 王蕴璇 PB21081565

一、实验一：基于 C 的 MDLK 工程建立、调试及 CMSIS-CORE 函数调用

1. 记录：变量存放格式

① 观察变量 ui_Tmp, a, b, c 保存位置

变量	寄存器
ui_Tmp	R_5
ui_a	R_2
ui_b	R_3
ui_c	R_4

② 通过对 PC 寄存器的观察，可以发现，其作为程序计数器，在代码的执行过程中不断累加增大；

③ 通过对 PSR 寄存器的标志位的观察，可以发现，其标志位变为 Negative，即表征此处运算的结果是一个负数；

④ 通过对数的表示方法（补码）的观察，有以下对应关系

数	补码表示
-1	0xFFFFFFFF
-2	0xFFFFFFFE
-2-32766=-32768	0xFFFF8000

⑤ 通过对循环体执行过程的观察，可以发现，执行过程的变化可以总结为下

S	HEX	地址
S[0]	0x80000001	0x2000000C
S[1]	0x80000002	0x20000010
S[2]	0x80000003	0x20000014
S[3]	0x80000004	0x20000018
S[4]	0x80000005	0x2000001C
S[5]	0x80000006	0x20000020
S[6]	0x80000007	0x20000024
S[7]	0x80000008	0x20000028

2. 记录：使用 CMSIS-CORE 函数实现底层操作并记录读取内容，所记录到的实验结果如下：

ui_tmp	0x00000000	ui_tmp	0x00000004
__get_FAULTMASK()	0x00000000	__get_CONTROL()	0x00000004
ui_tmp	0x00000000	ui_tmp	0x21000000
__get_BASEPRI()	0x00000000	__get_xPSR()	0x21000000
ui_tmp	0x00000000	ui_tmp	0x20000660
__get_PRIMASK()	0x00000000	__get_MSP()	0x20000660

可以发现，ui_tmp 的值和 Register 窗口观察到的数值一致。

二、实验二：基于 GPIO 的基本输入输出

记录：位带 IO 地址映射范围和实现方法

这里使用位带映射来进行 LED 灯的点亮和熄灭操作。

I/O 地址的映射范围在 sys.h 中已有定义，具体可以总结为：

GPIOA	输入	0x40020010-0x40020013	GPIOF	输入	0x40021410-0x40021413
	输出	0x40020014-0x40020017		输出	0x40021414-0x40021417
GPIOB	输入	0x40020410-0x40020413	GPIOG	输入	0x40021810-0x40021813
	输出	0x40020414-0x40020417		输出	0x40021814-0x40021817
GPIOC	输入	0x40020810-0x40020813	GPIOH	输入	0x40021C10-0x40021C13
	输出	0x40020814-0x40020817		输出	0x40021C14-0x40021C17
GPIOD	输入	0x40020C10-0x40020C13	GPIOI	输入	0x40022010-0x40022013
	输出	0x40020C14-0x40020C17		输出	0x40022014-0x40022017
GPIOE	输入	0x40021010-0x40021013			
	输出	0x40021014-0x40021017			

而实现过程利用了以下函数，这里以 GPIOG 引脚的相关操作作为示例：

```
#定义位带操作地址范围
#define GPIOG_IDR_Addr    (GPIOG_BASE+16) //0x40021810
#define GPIOG_ODR_Addr    (GPIOG_BASE+20) //0x40021814

#根据给定的 GPIO 端口和引脚编号，计算出相应的位带操作地址，从而实现对单个引脚的操作
#define BITBAND(addr, bitnum) ((addr & 0xF0000000)+0x2000000+((addr & 0xFFFFF)<<5)+(bitnum<<2))
#define MEM_ADDR(addr)  *((volatile unsigned long  *)(addr))
#define BIT_ADDR(addr, bitnum)   MEM_ADDR(BITBAND(addr, bitnum))

#位带操作定义
#define PGout(n)   BIT_ADDR(GPIOG_ODR_Addr,n)
#define PGin(n)    BIT_ADDR(GPIOG_IDR_Addr,n)

#具体操作代码
PGout(11)=1;
PGout(11)=0;
```

三、综合性实验：实验 1、2、3、4 的综合模块

1. 功能描述

综合实验为综合实验 1、2、3、4 的综合模块。利用 LCD 屏幕呈现出功能及其对应选择方式，利用三个 DIP 开关进行选择功能：

当没有 DIP 打开时，界面显示“选择功能”；

当 打开 DIP0，为呼吸灯；

当 打开 DIP1，为流水灯；

当 打开 DIP2，为外部中断；

当 打开 DIP3，为 LCD 显示字幕。

2. 功能模块划分

在选择功能时，使用到了 DIP 开关,这里采用了上拉输入。DIP-GPIO 对应关系为：

DIP0	PE4
DIP1	PE5
DIP2	PC14
DIP3	PC15

流水灯模块，用到了 8 个 LED 灯，LED-GPIO 对应关系为：

LED0	PG11
LED1	PG10
LED2	PG9
LED3	PD7
LED4	PG3
LED5	PG2
LED6	PD13
LED7	PD12

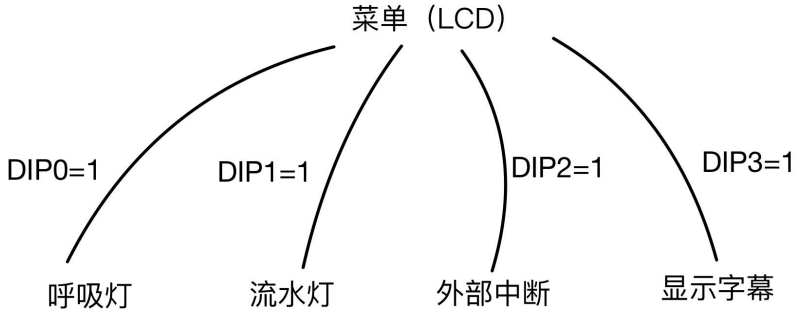
LCD 模块，采用上拉输入。LCD-GPIO 关系为：

LCD_CS	PG1
LCD_SID	PF15
LCD_CLK	PF14

呼吸灯模块，使用了 PB10 引脚，由 DIP 控制

中断模块，以 SW0（E0）做为外部中断输入引脚控制 LED 的亮灭

3. 系统结构图



4. 实现功能的核心代码

本次代码结构在之前实验的基础上进行相关改进与整合。使用到了以下几个模块：

1. main.c 主函数
2. led.c 用于初始化 LED
3. 12864.c 用于初始化 LCD
4. pwm.c 用于初始化 PWM 以实现呼吸灯模块
5. key.c 用于初始化 DIP 开关

1) main.c

```
#include "stm32f4xx.h"
#include "stm32f4xx_it.h"
#include "delay.h"//—" ±ƒØ "À˜'/µƒÕ∑Œƒº¸
#include "sys.h"
#include "usart.h"
#include "12864.h"
#include "led.h"
#include "timer.h"
#include "pwm.h"
#include "exti.h"
#include "key.h"


int mode = 6;
int pwm  = 0;
int dir  = 0;
int main(void)
{
    delay_init(168);»     //≥ı ºªØ—" ±ƒØ "
//» NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);//…Ë÷√œµÕ≥÷–∂œ"≈œ»ºƒ∑÷ôÈ2
    LED_Init();
  LCD_GPIO_Init();
  LCD_Init();
  delay_ms(20);

    KEY_Init();/*
```

```
  while(1){
    LCD_Display_Words(0,0,"—°'Òπ¶ƒ‹:1/ÙŒ£µΔ");
    LCD_Display_Words(1,0,"2 ¡˜ÀÆµΔ");
    LCD_Display_Words(1,4,"3 ÷—∂œ");
   if(DIP1==1){
        delay_ms(20);
        delay_init(168);
        NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);
        TIM2_PWM_Init(3000,0);
        while(1&(DIP1==1))
        {
        delay_ms(1);
            if(dir == 0){
        pwm += 10;
        if(pwm >= 3000) dir=1;}
            if(dir == 1){
        pwm -= 10;
        if(pwm <= 10) dir=0;}
        TIM_SetCompare3(TIM2, pwm); //
            LCD_Display_Words(2,0,"µπ°Δ ±:          ");
            LCD_ShowNum(3,2,pwm,4);
        }
      }
    if(DIP2==1){
      delay_ms(20);
      LCD_Display_Words(2,0,"¡˜ÀÆµΔ±«∞¡¡µΔ");
      PGout(11)=1;delay_ms(500);PGout(11)=0;LCD_Display_Words(3,2,"1");delay_ms(500); if(DIP2==0) break;
     PGout(10)=1;delay_ms(500);PGout(10)=0;LCD_Display_Words(3,2,"2");delay_ms(500);if(DIP2==0) break;
     PGout(9)=1;delay_ms(500);PGout(9)=0;LCD_Display_Words(3,2,"3");delay_ms(500);if(DIP2==0) break;
     PDout(7)=1;delay_ms(500);PDout(7)=0;LCD_Display_Words(3,2,"4");delay_ms(500);if(DIP2==0) break;
     PGout(3)=1;delay_ms(500);PGout(3)=0;LCD_Display_Words(3,2,"5");delay_ms(500);if(DIP2==0) break;
     PGout(2)=1;delay_ms(500);PGout(2)=0;LCD_Display_Words(3,2,"6");delay_ms(500);if(DIP2==0) break;
     PDout(13)=1;delay_ms(500);PDout(13)=0;LCD_Display_Words(3,2,"7");delay_ms(500);if(DIP2==0) break;
     PDout(12)=1;delay_ms(500);PDout(12)=0;LCD_Display_Words(3,2,"8");delay_ms(500);if(DIP2==0) break;
```

```
    }
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2); delay_init(168); LED_Init();
    if(DIP3==1){
        delay_ms(20);
//      LCD_Display_Words(2,0,"ºΔÀ„ª'≠¿Ì"Î«∂»Î ΩœμÕ≥");
//      LCD_Display_Words(3,0,"◊€ƒœ–' μ–È");
        KEY_Init();
        EXTIX_Init();
        LED0=1;
        while(1)
        {
}
        }
    if(DIP1==0&DIP2==0&DIP3==0){
        delay_ms(20);
        LCD_Display_Words(2,0,"«Î–°'Òπ¶ƒ‹");
        LCD_Display_Words(3,0,"··········");
    }
  }
}
```

2) Led.c

```
#include "led.h"

void LED_Init(void)
{    »
  GPIO_InitTypeDef  GPIO_InitStructure;   //GPIO≥ı ºªØ·ππÃÂ
  RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOG|
  »  »  »  »  »  »  »  »  RCC_AHB1Periph_GPIOD,
                          ENABLE);// πƒ‹GPIOG°¢GPIOD ±÷”

  »  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_11 | GPIO_Pin_10 |
       GPIO_Pin_9 | GPIO_Pin_2 | GPIO_Pin_3;»  »  »  »
  »  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_11 | GPIO_Pin_10 | GPIO_Pin_7 |
       GPIO_Pin_9 | GPIO_Pin_2 |
       GPIO_Pin_3);//LED1,LED2,LED3,LED4,LED5

  »  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_7 |GPIO_Pin_12 |
       GPIO_Pin_13;»  »  »  »
  »  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(GPIOD, &GPIO_InitStructure);
  »  GPIO_ResetBits(GPIOD, GPIO_Pin_7 |GPIO_Pin_12|GPIO_Pin_13);
       //LED6,LED7»

}
```

3) 12864.c

```c
#include "12864.h"
#include "delay.h"

void LCD_GPIO_Init(){
  GPIO_InitTypeDef  GPIO_InitStructure;   //GPIO≥ı ºªØ
  RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOG|
                         RCC_AHB1Periph_GPIOF,
                         ENABLE);// πƒ‹GPIOG°¢GPIOD ±÷”

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_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;
    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); //GPIO?????
  CS=1;
  SID=1;
  SCLK=1;
}

void SendByte(u8 byte)
{
u8 i;
for(i = 0;i < 8;i++)

{
if((byte << i) & 0x80) //0x80(1000 0000)
{
SID = 1;
}
else
{
SID = 0;
}
SCLK = 0;
delay_us(5);
SCLK = 1;
}
}
void Lcd_WriteCmd(u8 Cmd )
{
delay_ms(1);
SendByte(WRITE_CMD); //11111,RW(0),RS(0),0
SendByte(0xf0&Cmd);
SendByte(Cmd<<4);
}
void Lcd_WriteData(u8 Dat )
{
delay_ms(1);
SendByte(WRITE_DAT);
SendByte(0xf0&Dat);
SendByte(Dat<<4);
}
void LCD_Init(void)
{
delay_ms(50);
Lcd_WriteCmd(0x30);
delay_ms(1);
Lcd_WriteCmd(0x30);
delay_ms(1);
```

```
Lcd_WriteCmd(0x0c);
delay_ms(1);
Lcd_WriteCmd(0x01);
delay_ms(30);
Lcd_WriteCmd(0x06);
}
void LCD_Display_Words(uint8_t x,uint8_t y,uint8_t*str)
{
Lcd_WriteCmd(LCD_addr[x][y]);
while(*str>0)
{
Lcd_WriteData(*str);
str++;
}
}
void LCD_Display_Picture(uint8_t *img)
{
uint8_t x,y,i;
Lcd_WriteCmd(0x34);
Lcd_WriteCmd(0x34);
for(i = 0; i < 2; i++)
{
for(y=0;y<32;y++)
{
for(x=0;x<8;x++)
{
Lcd_WriteCmd(0x80 + y);
Lcd_WriteCmd(0x80 + x+i*0x08);
Lcd_WriteData(*img ++);
Lcd_WriteData(*img ++);
}
}
}
Lcd_WriteCmd(0x36);
Lcd WriteCmd(0x30);
}
void LCD_Clear(void)
{
Lcd_WriteCmd(0x01);
delay_ms(2);
}
int LCD_Pow(uint8_t x,uint8_t y)
{
    unsigned char i = 0;
    int result = 1;
    for(i=0;i<y;i++)
    {
        result *= x;
    }
    return result;
}
void LCD_ShowNum(uint8_t x,uint8_t y,int Num,int len)
{
    int i=0;
    Lcd_WriteCmd(LCD_addr[x][y]);
    for(i=len;i>0;i--)
    {
        Lcd_WriteData('0'+Num/LCD_Pow(10,i-1)%10); //'0' ¥"0ø™ º
    }
}
```

4) pwn.c

```c
#include "pwm.h"
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_InitStructure.GPIO_Pin = GPIO_Pin_10;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF; //????
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz; //?? 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);
}
```

5) key.c

```c
#include "stm32f4xx.h"
#include "delay.h"
#include "key.h"
void KEY_Init(void)
{
    GPIO_InitTypeDef  GPIO_InitStructure;
  RCC_AHB1PeriphClockCmd
       (RCC_AHB1Periph_GPIOE|RCC_AHB1Periph_GPIOF|RCC_AHB1Periph_GPIOC
       , ENABLE);

  GPIO_InitStructure.GPIO_Pin  = GPIO_Pin_0|GPIO_Pin_1|GPIO_Pin_2 |
       GPIO_Pin_4 | GPIO_Pin_5;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
  GPIO_Init(GPIOE, &GPIO_InitStructure);
    
  GPIO_InitStructure.GPIO_Pin  = GPIO_Pin_0 | GPIO_Pin_1 |
       GPIO_Pin_2 | GPIO_Pin_3 ;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
  GPIO_Init(GPIOF, &GPIO_InitStructure);
    
  GPIO_InitStructure.GPIO_Pin  = GPIO_Pin_14 | GPIO_Pin_15 ;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
  GPIO_Init(GPIOC, &GPIO_InitStructure);
}
```

6) timer.c

```c
#include "timer.h"
//?????3?????
//arr:?????? psc:??????
//????????***:Tout=((arr+1)*(psc+1))/Ft us. //Ft=???????,??:Mhz
//?????????3
void TIM3_Init(u16 arr, u16 psc)
{
TIM_TimeBaseInitTypeDef TIM_TimeBaseInitStructure;
NVIC_InitTypeDef NVIC_InitStructure;
RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM3,ENABLE);
//??? TIM3 ??
TIM_TimeBaseInitStructure.TIM_Period = arr;
//??????
TIM_TimeBaseInitStructure.TIM_Prescaler=psc;
//?????
TIM_TimeBaseInitStructure.TIM_CounterMode=TIM_CounterMode_Up;
//??????
TIM_TimeBaseInitStructure.TIM_ClockDivision=TIM_CKD_DIV1;
TIM_TimeBaseInit(TIM3,&TIM_TimeBaseInitStructure);
// ???????
TIM_ITConfig(TIM3,TIM_IT_Update,ENABLE);
//?????? 3 ????
NVIC_InitStructure.NVIC_IRQChannel=TIM3_IRQn;
//??? 3 ??
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority=0x01;
//????? 1
NVIC_InitStructure.NVIC_IRQChannelSubPriority=0x03;
//???? 3
NVIC_InitStructure.NVIC_IRQChannelCmd=ENABLE;
NVIC_Init(&NVIC_InitStructure);
// ???? NVIC
TIM_Cmd(TIM3,ENABLE);
//?????? 3
}
```

5. 运行结果

呼吸灯，流水灯，外部中断均可正常运行，已由助教老师检查验收

6. 总结

1) 代码进行了四次实验的综合，较好地总结了实验内容；
2) 但自我设计的实验板块较少，没有很大的创新点，对于自己的实验能力没有做到很好的考察；
3) 受到 LCD 屏幕的大小限制，无法进行较好地进行文字描述，LCD 显示功能模块较为混乱；
4) 没有考虑到多个 DIP 开关开启时的显示问题，目前仅保留第一个开启的 DIP 开关所对应的功能，若再次修改改进，可进行屏幕显示报错或其他处理方法；
5) 总的来说，综合实验有一定的综合度，对于实验内容有了较为深刻的理解，同时也巩固了理论知识。