
# 嵌入式系统设计实验报告

| | |
|---|---|
| **实验：** | 实验 8 PWM 控制实验 |
| **专业：** | 计算机科学与技术 |
| **班级：** | 1 班 |
| **姓名：** | 姚怀聿 |
| **学号：** | 22920202204632 |

2022 年 12 月 26 日

22920202204632

# 目　　录

一、　实验目的 ..... 3

二、　实验方案设计 ..... 3

1. 基础实验 ..... 3

2. 进阶实验 ..... 3

三、　实验过程 ..... 4

1. 基础实验 ..... 5

2. 进阶实验 ..... 6

四、　实验结果 ..... 8

五、　实验分析 ..... 10

六、　心得体会 ..... 10

2292022204632

# 一、实验目的

1、 掌握 PWM 的控制原理，掌握 PWM 模块相关寄存器及其相关参数的计算与设置方法。

2、 了解直流电机的基本工作原理，学会根据电机特性结合 PWM 模块工作原理进行应用程序编写。

# 二、实验方案设计

## 1. 基础实验

实验要求:

根据实验箱提供的电机特性设置 PWM 模块相关参数，实现电机的转动。

## 2. 进阶实验

实验要求:

根据实验箱提供的电机特性设置 PWM 模块相关参数，使得电机的位置与转速能通过改变 PWM 的脉冲个数与频率而被控制，实现电机的加速减速，并在 OLED 上显示电机转动信息，当拨码开关 4 拨为 RESET 时通过按动按键开关设置电机速度，转向，当拨码开关 4 拨为 SET 时启动电机。

2292020220432

# 三、实验过程

首先，对于所有的实验，我们都需要进行一些初始化的设置，这个函数在 main.c 中调用:

```
16
17  int main(void)
18  {
19    system_init();
20    while(1){
21      test2();
22    }
23  }
24
```

在该函数中初始化电机 GPIO:

```
23  //初始化 时钟、xll、电机gpio、oled、按键
24  void system_init(void)
25  {
26    usart0Init(EVAL_COM0, 115200U);
27    USART2_Init(115200U);
28    systick_config();
29    gd_XII_systeminit();
30    motor_gpio_config();
31    TIM3_Init(500, 500);
32    OLED_Gpio_Init(); // OLED初始化
33    OLED_Init();
34    gd_eval_key_init(USER1_KEY, KEY_MODE_EXTI);
35    gd_eval_key_init(USER2_KEY, KEY_MODE_EXTI);
36  }
37
```

这里要注意一点，OLED_Gpio_Init()和 OLED_Init()顺序不能颠倒，否则 OLED 会倒置。

电机 GPIO 配置如下，首先要开启时钟，将 PC6、PC7、PC8、PC9 的模式配置好:

2920202204632

```
void motor_gpio_config(void)
{
  rcu_periph_clock_enable(RCU_GPIOC);
  gpio_mode_set(GPIOC, GPIO_MODE_OUTPUT, GPIO_PUPD_PULLUP, GPIO_PIN_6);
  gpio_output_options_set(GPIOC, GPIO_OTYPE_PP, GPIO_OSPEED_50MHZ, GPIO_PIN_6);

  gpio_mode_set(GPIOC, GPIO_MODE_OUTPUT, GPIO_PUPD_PULLUP, GPIO_PIN_7);
  gpio_output_options_set(GPIOC, GPIO_OTYPE_PP, GPIO_OSPEED_50MHZ, GPIO_PIN_7);

  gpio_mode_set(GPIOC, GPIO_MODE_OUTPUT, GPIO_PUPD_PULLUP, GPIO_PIN_8);
  gpio_output_options_set(GPIOC, GPIO_OTYPE_PP, GPIO_OSPEED_50MHZ, GPIO_PIN_8);

  gpio_mode_set(GPIOC, GPIO_MODE_OUTPUT, GPIO_PUPD_PULLUP, GPIO_PIN_9);
  gpio_output_options_set(GPIOC, GPIO_OTYPE_PP, GPIO_OSPEED_50MHZ, GPIO_PIN_9);
}
```

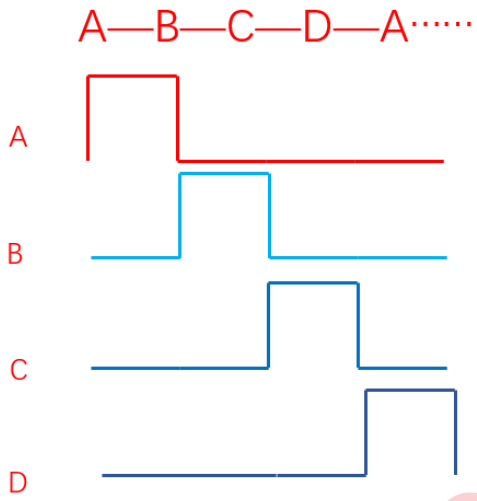
## 1. 基础实验

在 test.c 文件中编写 test1()函数:

基础实验比较简单，按照如下编写:

```
//基础 电机转动
void test1(void)
{
  uint8_t speed = 5;
  gpio_bit_reset(GPIOC, GPIO_PIN_6);
  gpio_bit_set(GPIOC, GPIO_PIN_7);
  gpio_bit_set(GPIOC, GPIO_PIN_8);
  gpio_bit_set(GPIOC, GPIO_PIN_9);
  delay_1ms(speed);
  gpio_bit_set(GPIOC, GPIO_PIN_6);
  gpio_bit_reset(GPIOC, GPIO_PIN_7);
  gpio_bit_set(GPIOC, GPIO_PIN_8);
  gpio_bit_set(GPIOC, GPIO_PIN_9);
  delay_1ms(speed);
  gpio_bit_set(GPIOC, GPIO_PIN_6);
  gpio_bit_set(GPIOC, GPIO_PIN_7);
  gpio_bit_reset(GPIOC, GPIO_PIN_8);
  gpio_bit_set(GPIOC, GPIO_PIN_9);
  delay_1ms(speed);
  gpio_bit_set(GPIOC, GPIO_PIN_6);
  gpio_bit_set(GPIOC, GPIO_PIN_7);
  gpio_bit_set(GPIOC, GPIO_PIN_8);
  gpio_bit_reset(GPIOC, GPIO_PIN_9);
  delay_1ms(speed);
}
```

2292202204632




便可实现电机的转动。

## 2. 进阶实验

在 test.c 文件中编写 test2()函数实现进阶实验的要求。

定义全局变量用于在中断函数中改变变量的值:

```
volatile uint16_t tim3_count;
volatile uint8_t motor_speed = 3;
volatile uint8_t mode_select;
char dir[] = "xxxxxxxxxx"; // 电机旋转方向
char pos[] = "Positive";
char rev[] = "Reverse";
char sto[] = "stop";
```

接下来是两个中断函数:

2250220204632

```
// 按下按键1，改变转速
void EXTI2_IRQHandler(void) {
  if(exti_interrupt_flag_get(USER1_KEY_EXTI_LINE)) {
    if(motor_speed++ >= 10) {
      motor_speed = 3;
    }
    exti_interrupt_flag_clear(USER1_KEY_EXTI_LINE);  // 清空标志
  }
}
// 按下按键2，改变转动的方向
void EXTI3_IRQHandler(void) {
  if(exti_interrupt_flag_get(USER2_KEY_EXTI_LINE)) {
    if(mode_select++ >= 2) {
      mode_select = 0;
    }
    exti_interrupt_flag_clear(USER2_KEY_EXTI_LINE);  // 清空标志
  }
}
```

三种模式:

```
typedef enum {
  forward = 0,
  reverse = 1,
  stop = 2
}motor_mode;
```

test2():

```
//进阶 按键控制电机转速、转向
void test2() {
  uint16_t i;
  if(gpio_input_bit_get(DIP4_GPIO_PORT, DIP4_PIN) == RESET) { // 拨码开关关闭 设置电机的转速和方向
    char str[4];
    sprintf(str, "%d.00", motor_speed); // 将全局motor_speed转换为字符串
    OLED_ShowString(0, 0, "speed:");
    OLED_ShowString(64, 0, str);

    if(mode_select == forward) { // 如果是正转 OLED上显示Positive
      OLED_ShowString(0, 2, "Positive  ");
      for(i = 0; i < 15; i++) {
        dir[i] = pos[i];
      }
    } else if(mode_select == reverse){ // 反转 显示Reverse
      OLED_ShowString(0, 2, "Reverse   ");
      for(i = 0; i < 15; i++) {
        dir[i] = rev[i];
      }
    } else {
      OLED_ShowString(0, 2, "Stop      "); // 停止 显示Stop
      for(i = 0; i < 15; i++) {
        dir[i] = sto[i];
      }
    }
  } else if(gpio_input_bit_get(DIP4_GPIO_PORT, DIP4_PIN) == SET) { // 拨码开关开启
    if(mode_select == forward) { // 选择不同的模式运行
      motor_fanz(motor_speed);
    } else if(mode_select == reverse) {
      motor_zhez(motor_speed);
    } else if(mode_select == stop) {
      motor_stop();
    }
  }
}
```

2292022204632

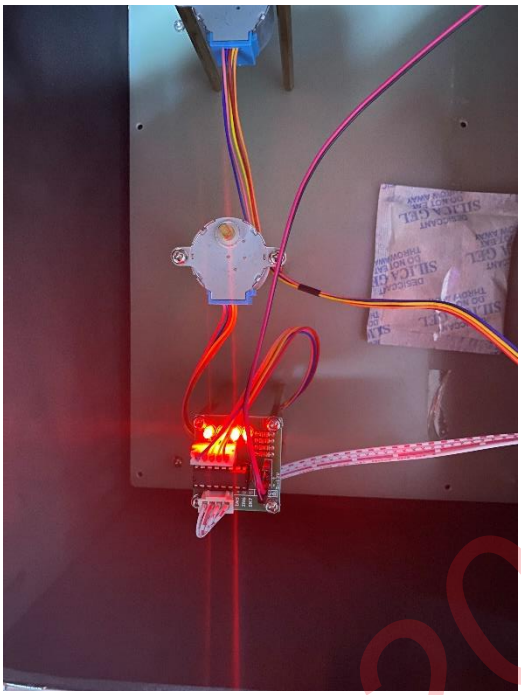
```
87  void motor_zhez(uint8_t speed) {
88      gpio_bit_set(GPIOC, GPIO_PIN_6);
89      gpio_bit_set(GPIOC, GPIO_PIN_7);
90      gpio_bit_set(GPIOC, GPIO_PIN_8);
91      gpio_bit_reset(GPIOC, GPIO_PIN_9);
92      delay_1ms(speed);
93      gpio_bit_set(GPIOC, GPIO_PIN_6);
94      gpio_bit_set(GPIOC, GPIO_PIN_7);
95      gpio_bit_reset(GPIOC, GPIO_PIN_8);
96      gpio_bit_set(GPIOC, GPIO_PIN_9);
97      delay_1ms(speed);
98      gpio_bit_set(GPIOC, GPIO_PIN_6);
99      gpio_bit_reset(GPIOC, GPIO_PIN_7);
100     gpio_bit_set(GPIOC, GPIO_PIN_8);
101     gpio_bit_set(GPIOC, GPIO_PIN_9);
102     delay_1ms(speed);
103     gpio_bit_reset(GPIOC, GPIO_PIN_6);
104     gpio_bit_set(GPIOC, GPIO_PIN_7);
105     gpio_bit_set(GPIOC, GPIO_PIN_8);
106     gpio_bit_set(GPIOC, GPIO_PIN_9);
107     delay_1ms(speed);
108 }
109
```

```
63
64  void motor_fanz(uint8_t speed) {
65      gpio_bit_reset(GPIOC, GPIO_PIN_6);
66      gpio_bit_set(GPIOC, GPIO_PIN_7);
67      gpio_bit_set(GPIOC, GPIO_PIN_8);
68      gpio_bit_set(GPIOC, GPIO_PIN_9);
69      delay_1ms(speed);
70      gpio_bit_set(GPIOC, GPIO_PIN_6);
71      gpio_bit_reset(GPIOC, GPIO_PIN_7);
72      gpio_bit_set(GPIOC, GPIO_PIN_8);
73      gpio_bit_set(GPIOC, GPIO_PIN_9);
74      delay_1ms(speed);
75      gpio_bit_set(GPIOC, GPIO_PIN_6);
76      gpio_bit_set(GPIOC, GPIO_PIN_7);
77      gpio_bit_reset(GPIOC, GPIO_PIN_8);
78      gpio_bit_set(GPIOC, GPIO_PIN_9);
79      delay_1ms(speed);
80      gpio_bit_set(GPIOC, GPIO_PIN_6);
81      gpio_bit_set(GPIOC, GPIO_PIN_7);
82      gpio_bit_set(GPIOC, GPIO_PIN_8);
83      gpio_bit_reset(GPIOC, GPIO_PIN_9);
84      delay_1ms(speed);
85  }
```

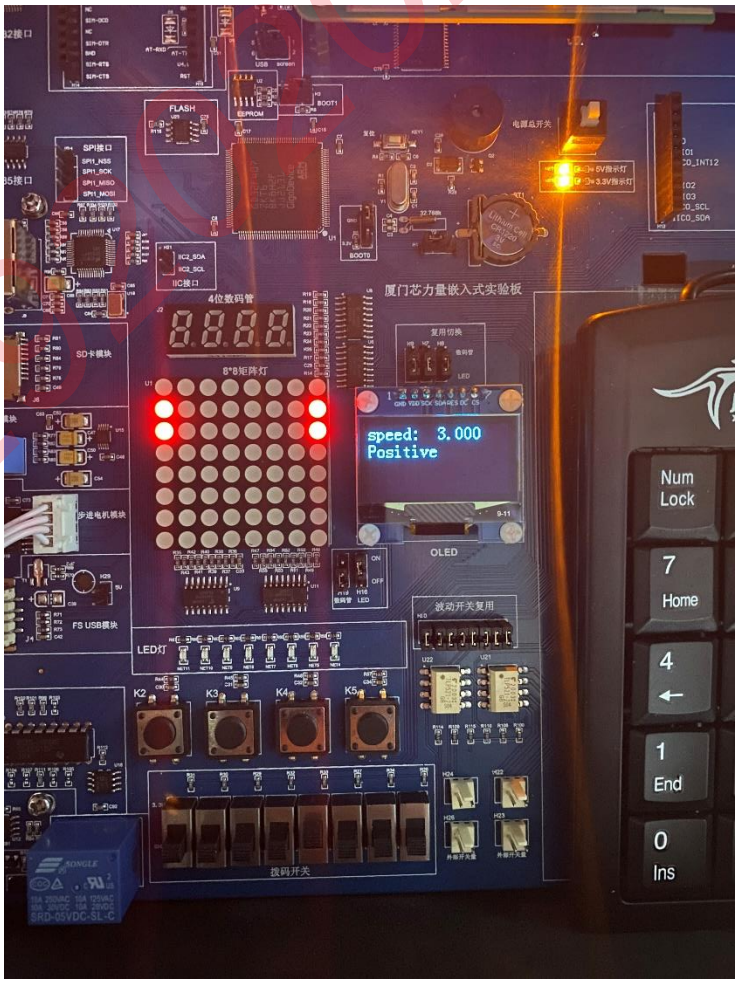
# 四、实验结果

1. 基础实验

2290702204632



2. 进阶实验




22[illegible]022204632

## 五、实验分析

本次实验较为简单，上课时老师已经讲述得比较清晰了。对于基础实验，基本只要把课上的 PPT 看一遍，就能很快写出来。对于进阶实验，实现起来会比较麻烦，主要是一开始没有弄明白实验的要求，看了很久。弄清楚实验要求后，就比较简单了。做实验时，不仅用到了 PWM，还与 OLED 以及中断函数相结合，本次实验也算是一个比较有综合性的实验了。

## 六、心得体会

本次实验并不复杂，通过本次实验，我学会了步进电机的工作原理，以及如何通过编程实现步进电机的转动，掌握了 PWM 模块的相关寄存器以及相关参数的计算与设置方法。

2292020220463232