



北京航空航天大学
BEIHANG UNIVERSITY

《单片机系统实验》报告二 ——传感器或执行器实验

北京航空航天大学研究生院

二〇二五年十月

目 录

第一章 实验目的	1
第二章 实验设备	1
第三章 实验原理	1
3.1 角度伺服电机工作原理	1
3.2 接口电路分析	2
第四章 实验内容与步骤	3
4.1 实验内容	3
4.2 实验步骤	3

第一章 实验目的

通过本实验了解相关传感器或执行器的工作原理，能通过编程实现传感器或执行器的数据传输。

第二章 实验设备

STM32实验系统一套，个人笔记本（Windows操作系统，带USB接口）一台。

第三章 实验原理

了解温湿度传感器、空气质量传感器、光敏传感器、步进电机驱动、舵机驱动（选其中一个）的使用方法，编程实现相关信号的采集和控制。

3.1 角度伺服电机工作原理

伺服电机是一种特殊类型的电机，与大多数其他电机不同，它设计用于精确定位而不是可控速度。伺服电机是服从控制信号指挥的电机，在信号来之前，转子停止不动；信号来到之后，转子立即运动。其功能是将电信号转换为转轴的角位移或角速度。伺服电机主要靠脉冲来定位，伺服电机接收到一个脉冲，就会旋转一个脉冲对应的角度，从而实现位移。

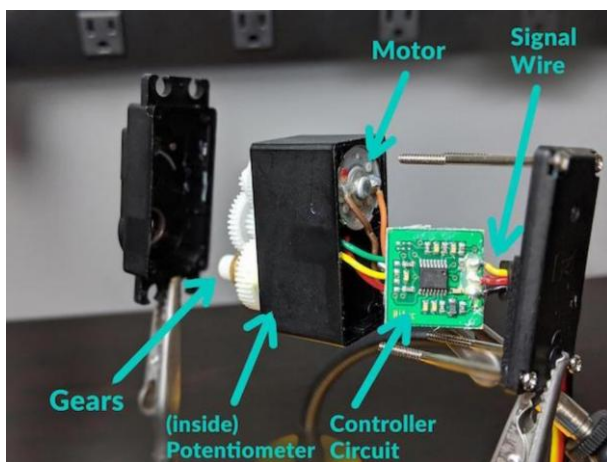


图1 伺服电机主要工作部件

伺服电机本身具备发射脉冲的功能，所以伺服电机每旋转一个角度，都会发出对应数量的脉冲，与伺服电机所接收的脉冲形成了呼应，也称为闭环。如此，系统在发送脉冲给电机的同时，能够知悉收回的脉冲数量，从而实现对电机的精确控制和定位，精度可达0.001mm。伺服电机内部的转子是永磁铁，驱动器控制的U/V/W三相电形成电磁场，转子在此磁场的作用下转动，同时电机自带的编码器反馈信号给驱动器，驱动器根据反馈值与目标值进行比较，调整转子转动的角度。伺服电机的

精度决定于编码器的精度（线数）。

3.2 接口电路分析

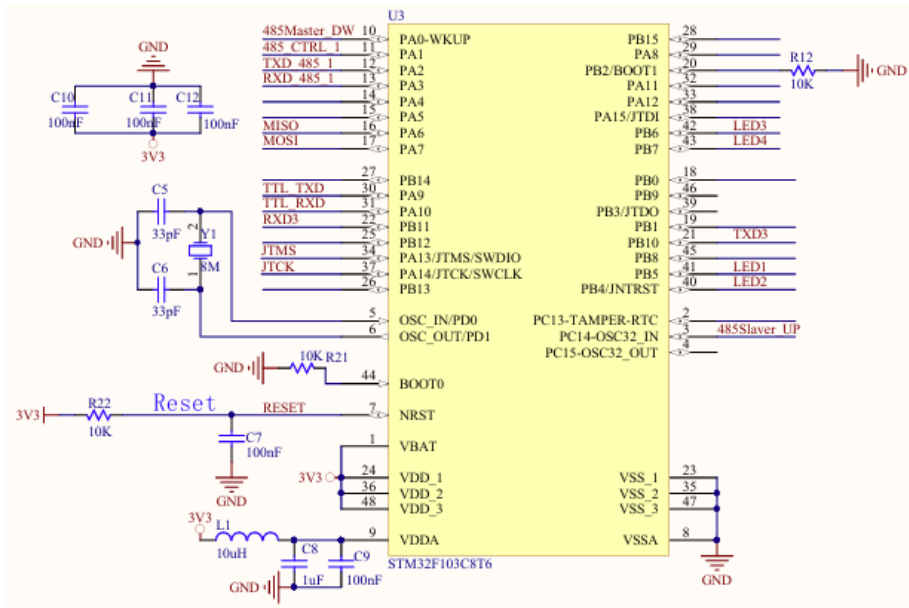


图2 角度伺服电机单片机部分底板

角度伺服电机，即舵机，有三根线，分别是地线GND，电源线和信号线。舵机通过信号线接收的是PWM信号，当信号进入内部电路产生一个偏置电压，触发电机通过减速齿轮带动电位器移动，使电压差为零时，电机停转，从而达到伺服的效果。简单来说就是给舵机一个特定的PWM信号，舵机就可以旋转到指定的位置。本次实验的角度伺服电机只能通过IN4信号线对舵机进行控制，该信号线连接的是STM32上的PA6口。即STM32通过PA6输出PWM信号控制舵机转动的角度。

舵机的转动角度与PWM（Pulse Width Modulation，脉冲宽度调制）信号的占空比有关。本实验所使用的舵机，PWM信号的周期固定为20ms（50Hz），脉宽可设置为0.5m2.5ms之间，脉宽和舵机的转角0°～180°相对应，如表1所示。

表1 脉宽与舵机转角对应关系

脉宽（ms）	转角（°）
0.5	0
1.0	45
1.5	90
2.0	145
2.5	180

STM32的8个通用定时器中除了TIM6和TIM7两个基本定时器，其它定时器都可以产生PWM输出。其中高级定时器TIM1和TIM8可以同时产生多达7路的PWM输出。而普通定时器也能同时产生多达4路的PWM输出。这样，STM32最多可以同时产生30路PWM输出。本实验利用TIM3的CH1产生一路PWM输出来控制舵机，定时器的对应引脚为PA6。实物连接图如图3所示。



图3 实物连接图

第四章 实验内容与步骤

4.1 实验内容

选择一个传感器或控制器进行相关信号的采集或控制，能在调试环境中观察到相关实验数据的变化或在实际中观察到控制变化。（4学时）

4.2 实验步骤

(1) 引用头文件。

```
#include"stm32f10x_conf.h"
#include"gpio.h"
#include <stdint.h>
```

(2) 在源文件main.c中编写主函数，在无限循环中，以45°为步进角度，从0°逐步增加到180°，再反向递减回0°，实现舵机周期性摆动效果。

```
int main(void)
{
    u16 i=0;
    Servo_TIM3_PWM_Init();
    while(1)
    {
        for(;i<180;i+=45)
        {
            Servo_Move(i);
            Delay(9000000);
        }
        for(;i>0;i-=45)
        {
            Servo_Move(i);
            Delay(9000000);
        }
    }
}
```

```
}
}
}
```

(3) 在源文件main.c中编写定时器初始化函数，完成TIM3的PWM配置，设置计数周期。




```
void Servo_TIM3_PWM_Init(void)
{
    //初始化 TIM3
    GPIO_InitTypeDef GPIO_InitStructure;
    TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure;
    TIM_OCInitTypeDef TIM_OCInitStructure;
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM3, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
    //配置 GPIO.6 为输出 TIM 3 CH1 的 PWM 输出
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_6; // CH1
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_Init(GPIOA, &GPIO_InitStructure);
    //72MHz/((199+1)(7199+1))=50Hz 对应 T=20ms
    TIM_TimeBaseStructure.TIM_Period = 199;
    TIM_TimeBaseStructure.TIM_Prescaler = 7199;
    TIM_TimeBaseStructure.TIM_ClockDivision = 0;
    TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;
    TIM_TimeBaseInit(TIM3, &TIM_TimeBaseStructure);
    TIM_OCInitStructure.TIM_OCMode = TIM_OCMode_PWM1;
    //PWM1 计数器小于 CCR 值输出高电平，大于 CCR 值输出低电平
    TIM_OCInitStructure.TIM_OutputState = TIM_OutputState_Enable;
    TIM_OCInitStructure.TIM_OCPolarity = TIM_OCPolarity_High;
    TIM_OC1Init(TIM3, &TIM_OCInitStructure);
    TIM_OC1PreloadConfig(TIM3, TIM_OCPreload_Enable);
    TIM_Cmd(TIM3, ENABLE);
}
```

(4) 在源文件main.c中编写延时函数。

```
void Delay(u32 count)
{
    u32 i=0;
    for(;i<count;i++);
}
```

(5) 在源文件main.c中编写舵机转角函数，通过改变比较寄存器的值实现占空比调节，当角度变化时，对应PWM脉宽线性变化。

```
//angle/90+0.5=ms
void Servo_Move(u8 angle)
{
    TIM_SetCompare1(TIM3, 5-1 + angle/9);
}
```

(6) 编译与下载程序：点击工具栏依次执行Translate 、Build 、Download ，烧录完成后，舵机开始以固定间隔从0° 转至180°，再逐步返回至0°，循环往复平稳运动。