项目名称:

电磁曲射炮设计方案

设计者：

班号：

# 设计目标

## 功能要求

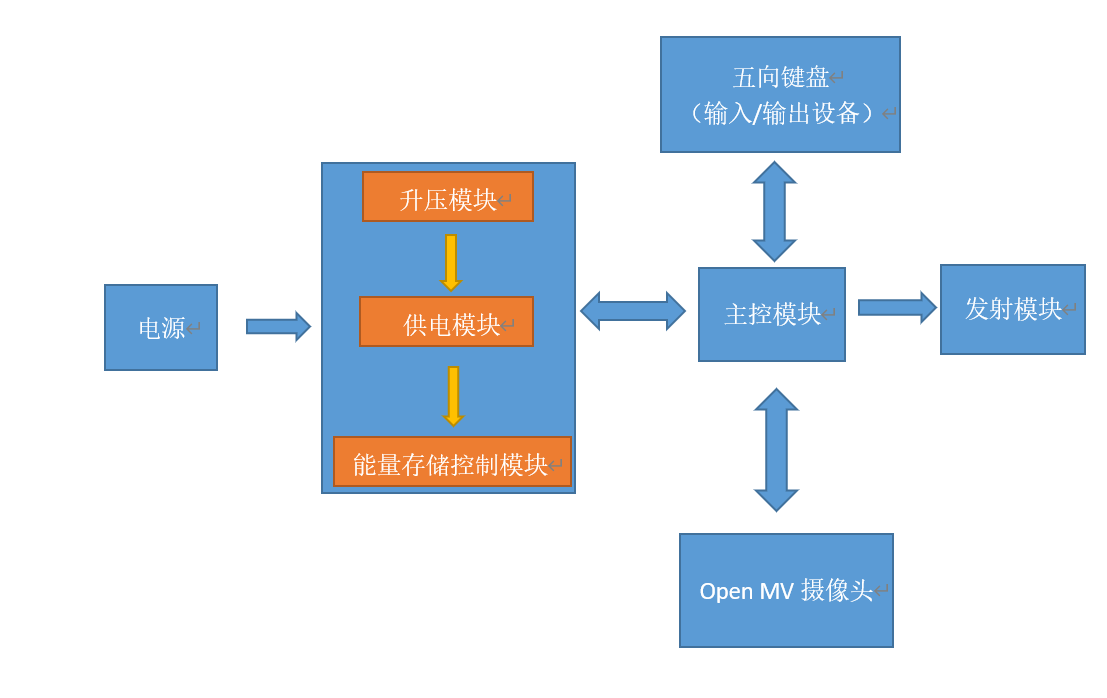
* 硬件部分通过STM32单片机对键盘输入的角度和距离或Open MV模块返回的角度和距离进行读取，正确发送指令实现对二维云台角度的控制并将角度距离等数据发送到OLED屏幕，升压模块对电容充电，延迟一段时间后，发送信号控制可控硅，实现炮击。
* 软件部分通过编写主程序代码，整合 PWM 脉宽调制、PID控制算法、颜色识别算法、单目测距算法、输入输出算法，达到整体系统协调统一。
* 控制方式：五向键盘输入或Open MV搜寻方式。

## 技术指标

* 电磁炮能够将弹丸射出炮口。
* 环形靶放置在靶心距离定标点 200~300cm 间，且在中心轴线上的位置处，键盘输入距离 d 值，电磁炮将弹丸发射至该位置，距离偏差的绝对值不大于 50cm。
* 用键盘给电磁炮输入环形靶中心与定标点的距离 d 及与中心轴线的偏离角度a，一键启动后，电磁炮自动瞄准射击，按击中环形靶环数计分；若脱靶则不计分。
* 在指定范围内任意位置放置环形靶即环形靶与引导标识一同放置在距离定标点 d=250cm 的弧线上（以靶心定位），引导标识处于最远位置。电磁炮放置在定标点，炮管水平方向与中轴线夹角a =-30°、仰角 0°。一键启动电磁炮，炮管在水平方向与中轴线夹角a从-30°至 30°、再返回-30°做往复转动，在转动过程中（中途不得停顿）电磁炮自动搜寻目标并炮击环形靶，按击中环形靶环数计分，启动至击发完成时间≤10s。

# 系统设计

## 总体设计目标



系统总体框架图

## 理论PWM与距离D关系计算

理论炮管仰角与D之间的关系（其中m为子弹的质量，g为重力加速度，D为炮台据引导标识的距离，M为磁化强度，V为子弹的体积，I为线圈的电流，N为线圈匝数）

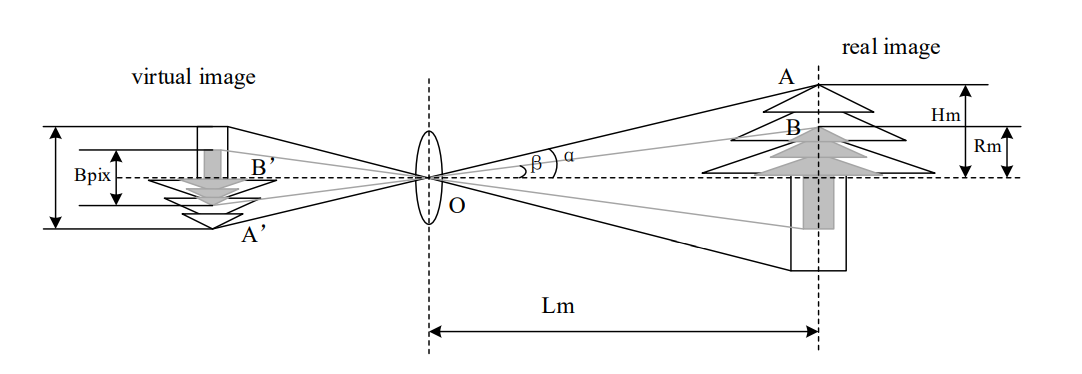
PWM参数与α关系式为：

则PWM参数与d的关系为

根据实际测量和Matlab线性拟合可得到实际情况下的关系。将关系输入程序即可。

## 理论距离D计算

小孔成像测距采用如下模型：



如图所示两次实物成像的映射关系分别用浅色和深色来标注。其中，被测目标与摄像头的距离为 D，标定摄像头参数时的物体高度和被测目标的高度、仰角、成像像素值分别用 、α、，、β、表示。通过相似三角形原理，可以得出： ； (1)

由(1)式得, (2)

在实物几何关系中，可得到目标距离D与被测目标高度 Rm 的三角函数式：

(3)

将(3)式带入(2)式可得出：

(4)

在测距前，设定摄像头的内部参数为一个常数，那么实际的距离就是一个固定的等式： (5)

最后，在测距阶段，提取摄像头内目标的成像像素值，根据（5）式即可计算出摄像头到目标的实际距离。

## 项目需要仪器设备

* 米尺
* 量角器
* 秒表
* 万用表
* STM32F103RCT6
* ST-LINK下载器/USB数据线

## 项目时间人员规划

|  |  |  |
| --- | --- | --- |
| 类别 | 时间 | 备注 |
| 方案论证 | 5工作日 | 前期准备包括器件落实、结构落实 |
| 硬件设计 | 5工作日 | 电路原理图设计PCB设计 |
| 软件设计 | 5工作日 | 软件设计可与硬件设计同步 |
| 焊接系统联调 | 5工作日 | 对电路进行焊接 |
| 软件优化和试制版本确定 | 5工作日 | 对软件部分进行调试，使软件能更好的配合硬件电路实现所有指标 |
| 文档整理 | 5工作日 | 总结在整个过程中遇到的问题及解决方方法的感想 |
| 合计 | 30工作日 |  |

整个项目过程需要3人投入约30个工作日。

# 硬件设计

## 硬件设计目标

1、硬件设计稳定可靠，节能，体积小，便于调试及维护，并满足最终产品的结构要求。

2、 硬件设计需要保证能够各部分稳定的运行，运行时各个器件的电流电压不能超过额定值，不损坏器件。

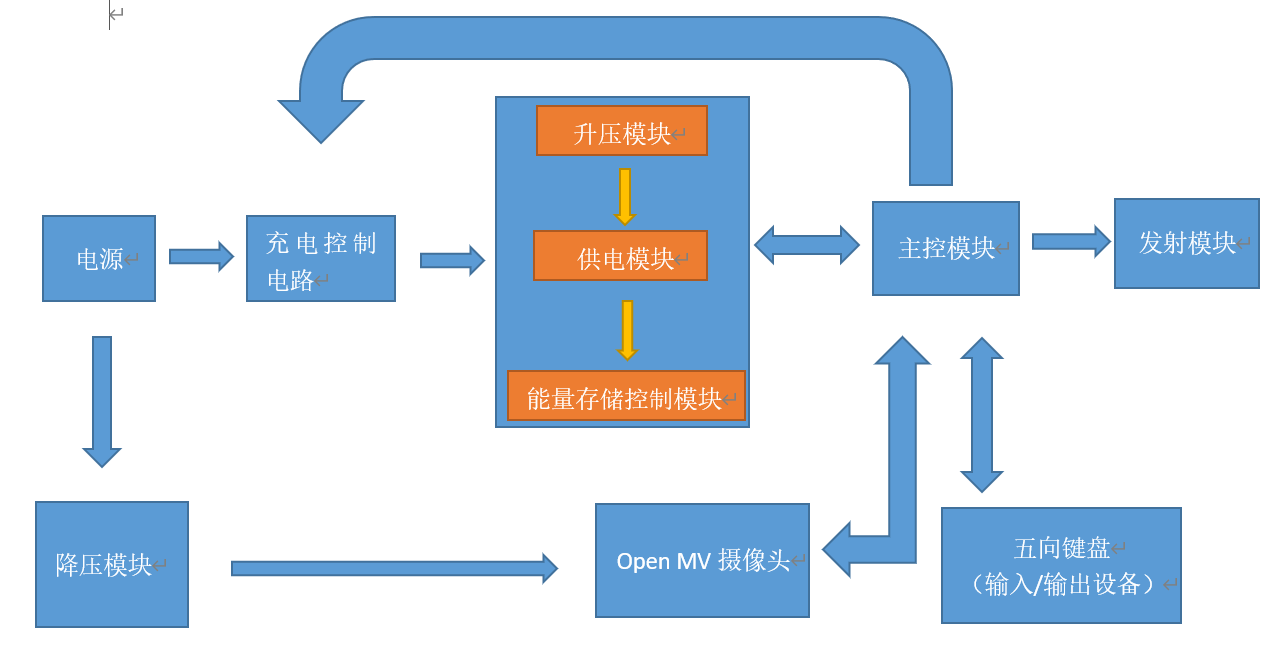
3、硬件设计需要让视觉和测距模块提供的数据信号以及人为输出的数据能够通过主控板精准控制发射装置的偏角位移等，做到误差小，准度高的目标。

4、硬件设计需要注意节能，保证电源放大的电流损耗小，效率高，信号能够完整通过芯片板。

5、硬件设计需要保证体积不能太大，最终让各个芯片板，电容，云台等器件排版有序，精密，美观。

## 硬件系统框架

### 硬件系统框图



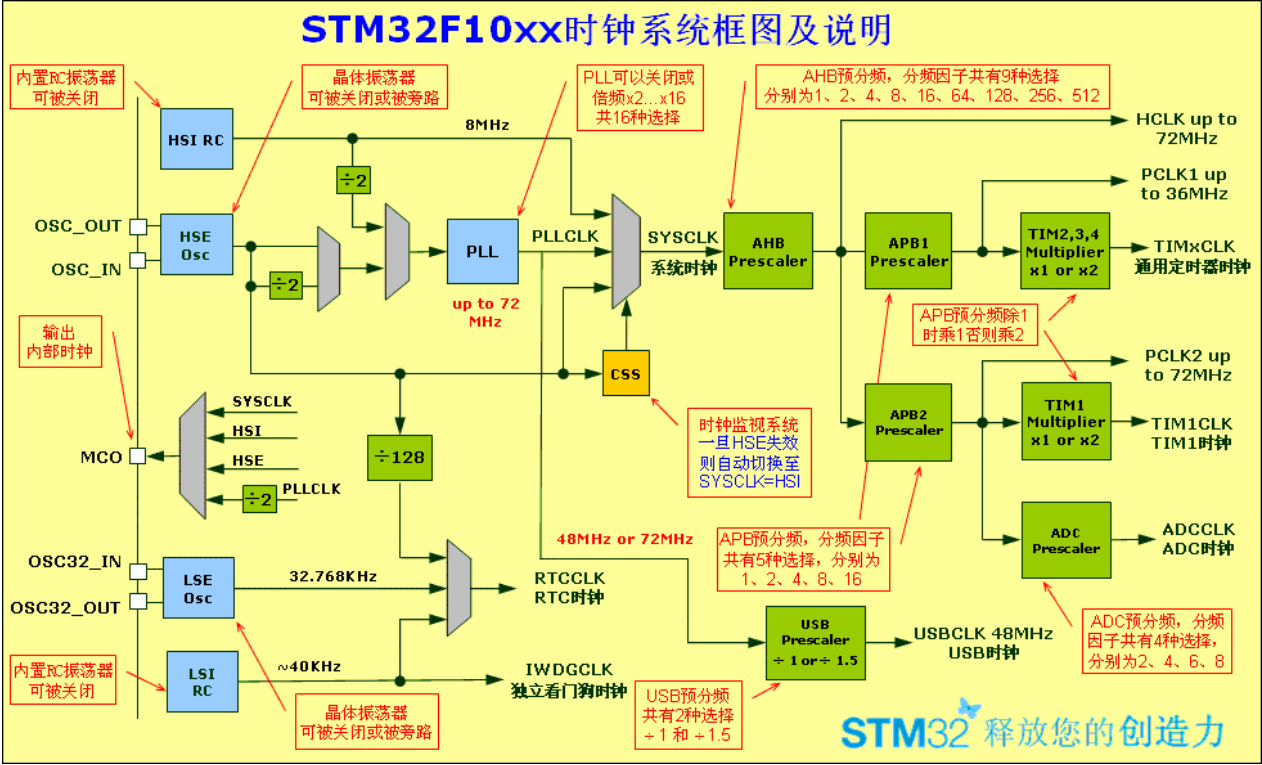
硬件框架图

### 电源网络

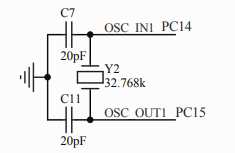
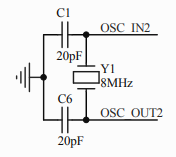
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 型号/模块 | 电压 | 最大电流 | 最小电流 | 最大功耗（功率） | 备注 |
| Open MV | 5V | 90mA | 140mA | 150mA@3.3V | STM32H7 |
| 升压模块 | 输入12V  输出120V | 输入20A  输出700mA | 输出200mA | 200W | 30%负载时效率>90% |
| 降压模块  LM2596S | 输入12V  输出5V | 3A | 1mA | ηMAX=96% |  |
| STM32F103RCT6 | 3.3V | 150mA | 1mA | 0.75W | 主控 |
| 舵机MG996R | 5V | 1450mA | 120mA | 10.44W | 主控VCC供电 |
| IIC OLED | 5V | 40mA | 320uA | 0.2W | 128\*64 |
| 电解电容 | 最大450V | 65A |  |  |  |
| FR207 | 击穿1000V  导通1.3V | 2A |  | 2000W |  |
| 继电器 | 12V | 10A |  |  | 高低电平控制 |

### 时钟网络

该系统拥有的时钟网络如图：



系统晶振电路： 外部晶振电路：

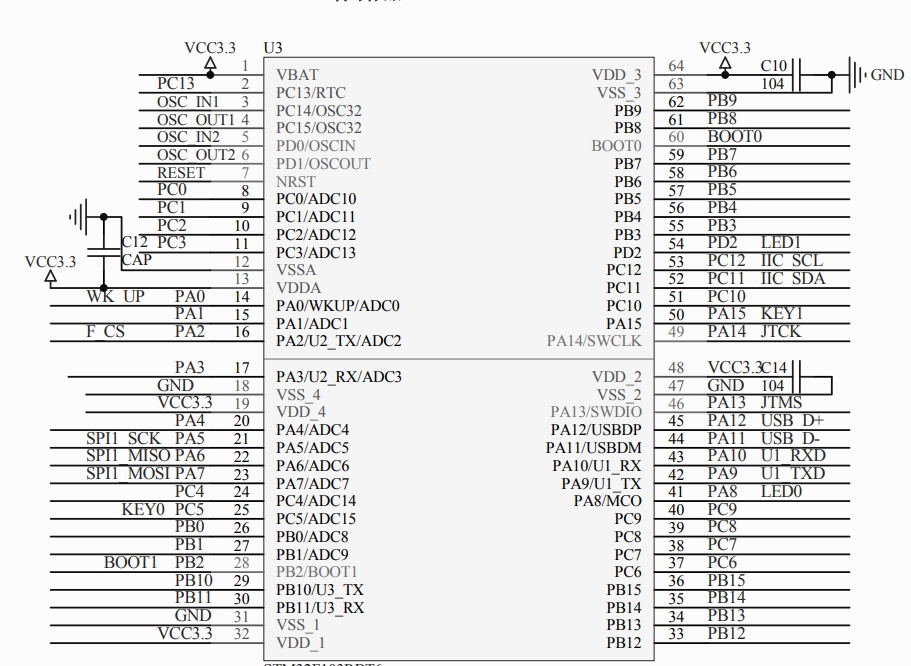


## 硬件详细设计

### 3.3.1. 主控板--（中心控制及数据处理）

主控板器，我们采用高性能、低成本、低功耗的单片机STM32F103RCT6来实现整机的信号处理与控制。该控制器功能强大，内部集成了非常丰富的外设，使用的时候只需要在程序里边进行编程设置相应的寄存器就能够使用对应的功能，相比于其他单片机来讲，其在执行效率和速度上都是他们所比不上的。在这里通过电脑载入代码到主控板来实现主控板电路的运行。主控板连接传感器，显示模块与继电器来分别处理图像的识别、距离的测算、偏角的调节与充放电功能的进行。

原理图如下：

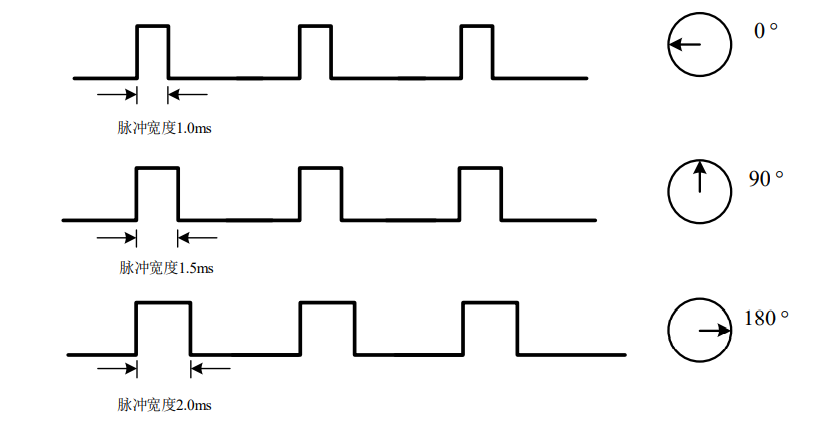


### 3.3.2.  云台装置

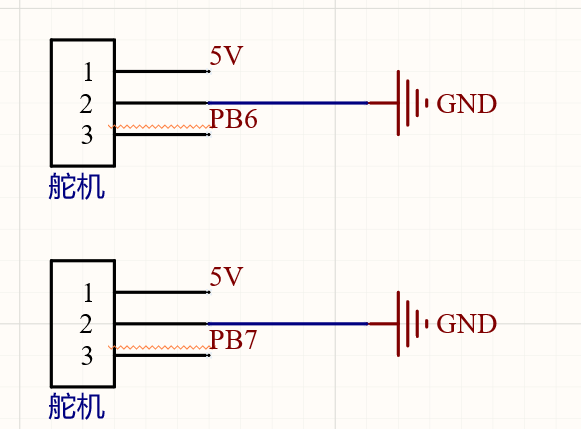
云台模块主要由舵机组成，要求能够满足水平和垂直方向受控移动，所以可以采用MG996R舵机云台。该云台能使电磁炮能够朝着水平方向与垂直方向按一定角度旋转。该模块主要由主控板来控制，当键盘输入参数信息，主控板发出信号控制舵机来实现转动。

舵机工作原理是：跟它内部的基准电压做对比，获得压差信号输出。一般需要产生一个周期为20ms的脉冲信号，这个信号的高电平可以在0.5ms~2.5ms范围内变化，并且每个脉冲对应一个角度。

舵机的供电电压在4.8-7.2V之间都是可以正常工作的，供电电压的大小影响舵机的转动速度，这里由STM32F103RCT6的VCC 5V供电。

通过信号线输入一定占空比的 PWM 脉冲确定旋转轴的位置。舵机控制的最大脉冲宽度为20毫秒，脉冲宽度将决定舵机转轴的角度，脉冲宽度与旋转角度的关系如下图所示

与STM32板连接电路图如下：

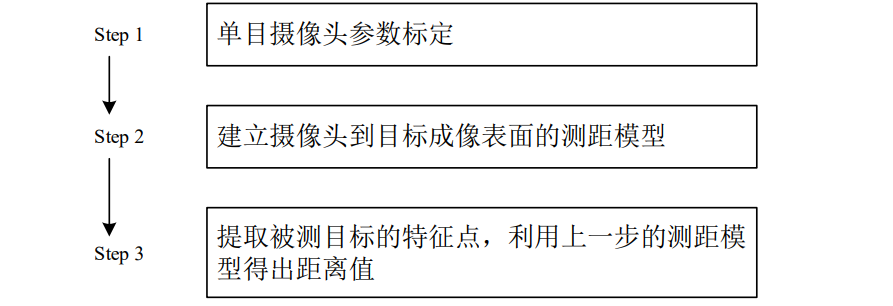


### 3.3.3. 传感器

采用了视觉传感器OpenMV4 H7 R2 Cam，主要起到目标捕捉效果和测距效果。该模块采用STM32H7主控芯片，主频为480MHz，RAM为1MB，FLASH为2MB，相比于上一代，它拥有更卓越的性能。

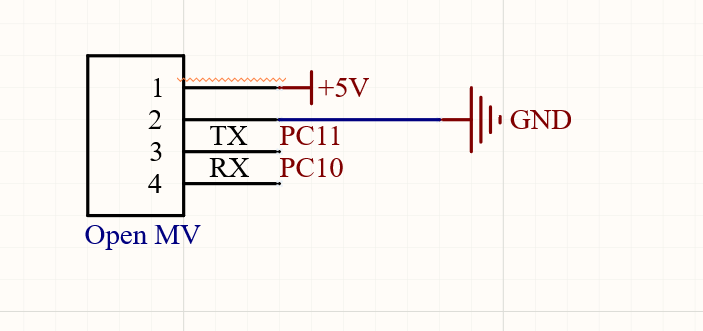
Open MV通过串口4与主控连接，实现主控与传感器之间的通信。

由于采用无畸变镜头，因此可以采用小孔成像原理实现目标测距，并分三个阶段实现。



首先，摄像机标定阶段，同时实现单目摄像头内参数的标定。其次，利用小孔成像原理，确定成像尺寸与实物尺寸的计算式(即映射关系)。

电路图如下：

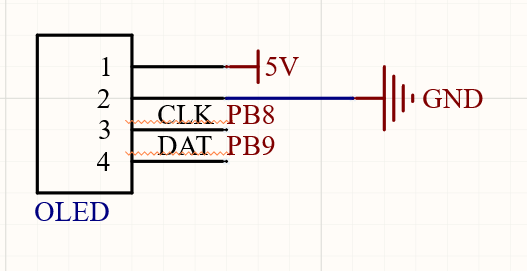
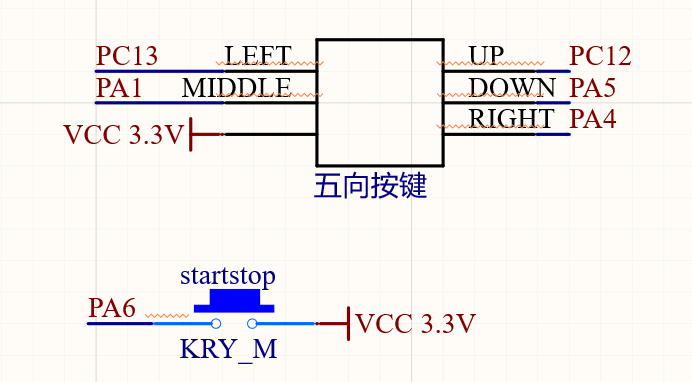


### 3.4.4. 输入、输出模块

输入模块采用五向键盘输入程序运行的模式选择、目标靶距离标定点的距离和角度、启动和结束发射的控制命令。

输出模块采用IIC OLED，其分辨率为128\*64，主要起人为操作与机器的交互功能，首先电源通过降压器使电压降到触摸屏板的5V，然后通过显示模块来得知当前电磁炮所处于的工作状态以及标靶物与定标点的角度与距离。

电路图如下：



### 3.4.5. 升压充放电模块与降压模块

直流电源通过升压模块将电压放大来达到电容所需要的充电电压，这里升压模块采用ZVS—MID模块，升压到100V。电压放大效率高、功率大、有效精度为1V，有效的提高了元器件的安全性。

稳压降模块采用LM2596S芯片，性能够用，给OLED屏、Open MV、STM32所需要的5V供电较为安全。

### 3.4.6. 发射模块

发射模块主要电容、炮管、电磁线圈和发射物构成。

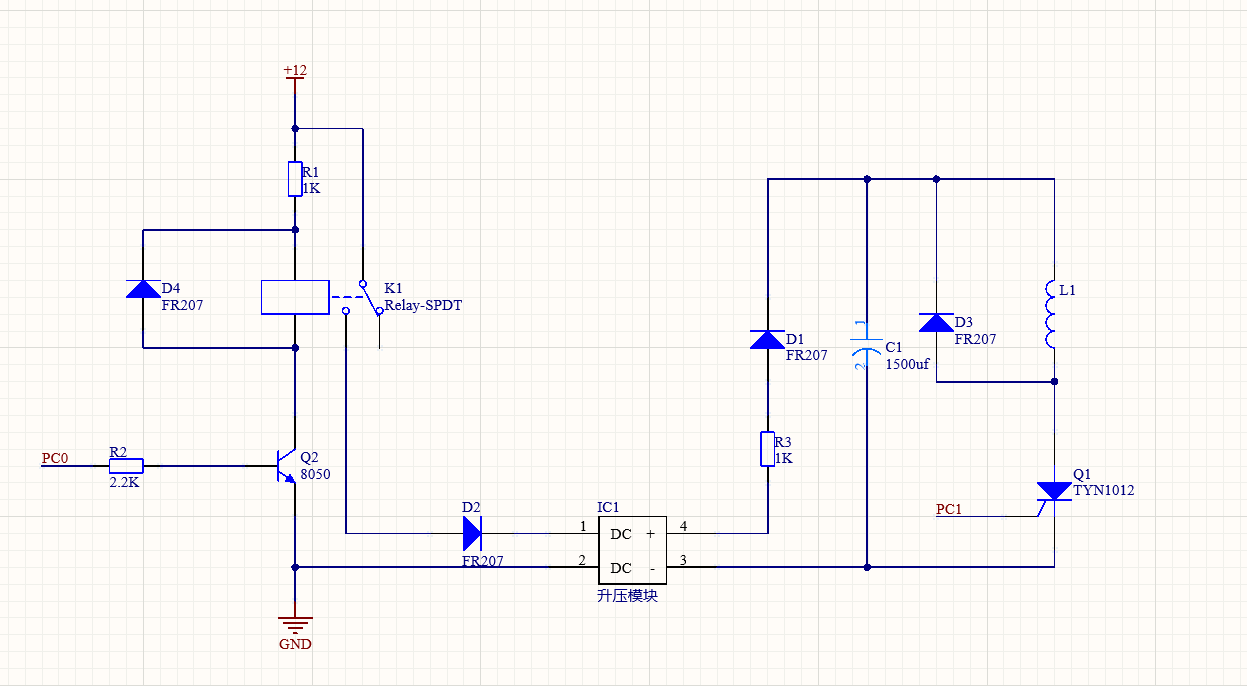
综和考虑成本以及发射能量，此次选用1500uF的电解电容，在100V电压下串联一个100Ω电阻，充电时间为150ms，另外电容在放电状态下瞬时电流可以达到900A，使得线圈产生巨大的洛伦兹力，同时在放电中后期电流逐渐减小使得线圈产生的力对发射物的反拉作用不大，保证了发射物能够获取到足够的动能。

因为炮管的环流损耗很大，因此材料需要选取非导电、弱磁性材料，所以我们准备选用PVC管。

线圈主要由我们手工制成，由于漆包线不能太细，不然电阻会太大，电流损耗就会很大，所以我们选用0.8mm漆包线。为了进一步缓解反拉作用，线圈的宽度不能够太大，所以线圈采用40匝/每层，绕8层的缠绕方式。

由动能公式E=1/2mv^2和发射能量不变得：发射物质量越小才能获得更大初速度，加上考虑空气阻力对较小物体的影响以及和管道的适配性，所以我们选取刚铁混柱形物为发射物。

具体发射电路如下图所示：



# 软件设计

## 软件设计目标

基于各硬件模块作进行开发和调试，最后整合各功能模块实现设计要求。

## 软件系统框架

### 软件流程图



图6 系统软件流程图

## 软件详细设计

### 开发工具和语言

开发平台：Keil MDK5、Open MV IDE

开发语言：STM32F103RCT6系统板使用C语言进行开发，Open MV采用python语言

### 子模块设计

子模块一：舵机驱动程序设计

舵机的角度与PWM的关系如下其中 PWMCCAP为单片机寄存器写入值。通过初始化GPIO管脚，由管脚输出PWM信号驱动舵机。

通过如下代码实现对应管脚的初始化：

GPIO\_InitTypeDef GPIO\_InitStructure;

TIM\_TimeBaseInitTypeDef TIM\_TimeBaseStructre;

TIM\_OCInitTypeDef TIM\_OCInitStructure;

RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_TIM4,ENABLE);

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOB,ENABLE);

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AF\_PP;

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_6|GPIO\_Pin\_7;//初始化引脚PB6和PB7

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOB,&GPIO\_InitStructure);//GPIO类型

TIM\_TimeBaseStructre.TIM\_Period = arr;

TIM\_TimeBaseStructre.TIM\_Prescaler = psc;

TIM\_TimeBaseStructre.TIM\_CounterMode = TIM\_CounterMode\_Up;

TIM\_TimeBaseStructre.TIM\_ClockDivision = 0;

TIM\_TimeBaseInit(TIM4,&TIM\_TimeBaseStructre);

TIM\_OCInitStructure.TIM\_OCMode = TIM\_OCMode\_PWM1;

TIM\_OCInitStructure.TIM\_OCPolarity = TIM\_OCPolarity\_High;

TIM\_OCInitStructure.TIM\_Pulse = 0;

TIM\_OCInitStructure.TIM\_OutputState = TIM\_OutputState\_Enable;

TIM\_OC1Init(TIM4,&TIM\_OCInitStructure);

TIM\_OC2Init(TIM4,&TIM\_OCInitStructure);

TIM\_OC1PreloadConfig(TIM4,TIM\_OCPreload\_Enable);

TIM\_OC2PreloadConfig(TIM4,TIM\_OCPreload\_Enable);

TIM\_ARRPreloadConfig(TIM4,ENABLE);

TIM\_Cmd(TIM4,ENABLE);

通过如下代码实现PWM信号的输出：

//左右舵机为引脚为PB6

void SetPWM1(int pwm1)

{

TIM4->CCR1 = pwm1;

}

//上下舵机引脚为PB7

void SetPWM2(int pwm2)

{

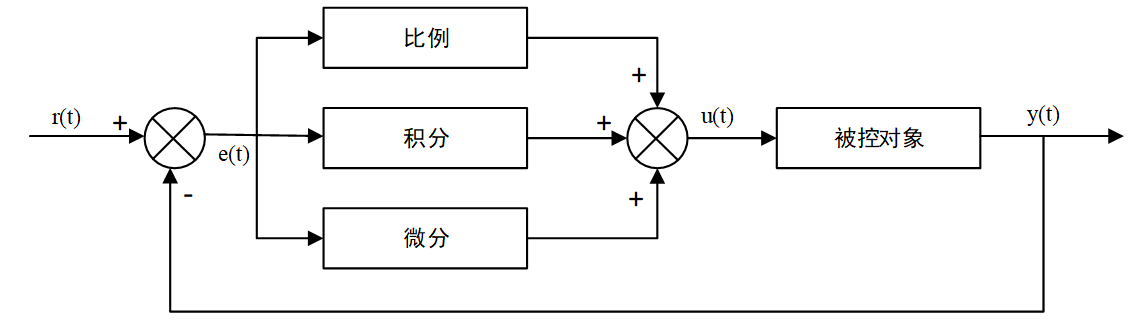
TIM4->CCR2 = pwm2;

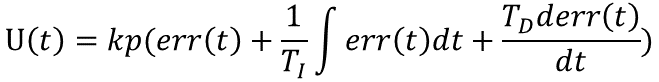
}

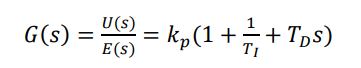
子模块二：PID控制算法设计

PID算法的使用可以让舵机转的更加迅速，同时舵机能够进行自检使得转动的位置更加准确。

PID原理如图所示：



PID公式为

或者写成传递函数形式

（kp 是比例系数，TI是积分时间常数，TD是微分时间常数）

采用位置式PID：

int M1\_PID\_Balance(void)

{

static float iError,dError;

iError = M1\_PID.Target - M1\_PID.Current; //偏差

if(Motor == 1)

M1\_PID.SumError += iError; //积分

else

M1\_PID.SumError = 0; //如果电机停止运动则清空误差和积分限幅

if(M1\_PID.SumError > SumErrorMax)

M1\_PID.SumError = SumErrorMax;

else if(M1\_PID.SumError < -SumErrorMax)

M1\_PID.SumError = -SumErrorMax;

dError = iError - M1\_PID.LastError; //当前微分

M1\_PID.LastError = iError;

return(int)( M1\_PID.Kp \* iError //比例项

+ M1\_PID.Ki \* M1\_PID.SumError //积分项

+ M1\_PID.Kd \* dError); //微分项

}

子模块三：输入、输出设计

采用五向键盘作为输入设备，不断扫描键盘来判断是否有值输入

扫描函数如下

u8 KEY\_Scan(u8 mode)

{

static u8 key\_up=1;//按键按松开标志

if(mode) key\_up=1; //支持连按

if(key\_up&&(UP==0||DOWN==0||LEFT==0||RIGHT==0||MID==0||START==0))

{

delay\_ms(2);//去抖动

key\_up=0;

if(UP==0)return UP\_PRES;

else if(DOWN==0)return DOWN\_PRES;

else if(LEFT==0)return LEFT\_PRES;

else if(RIGHT==0)return RIGHT\_PRES;

else if(MID==0)return MID\_PRES;

else if(START==0)return START\_PRES;

}else if(UP==1&&DOWN==1&&LEFT==1&&RIGHT==1&&MID==1&&START==1) key\_up=1;

return 0;// 无按键按下

}

采用2.42寸OLED屏作为输出设备

控制函数如下：

void OLED\_WR\_Byte(unsigned dat,unsigned cmd);

void OLED\_Display\_On(void);//开

void OLED\_Display\_Off(void);//关

void OLED\_Init(void);//初始化OLED

void OLED\_Clear(void);//清除OLED

void OLED\_DrawPoint(u8 x,u8 y,u8 t);

void OLED\_Fill(u8 x1,u8 y1,u8 x2,u8 y2,u8 dot);

void OLED\_ShowChar(u8 x,u8 y,u8 chr,u8 Char\_Size);//输出字符

void OLED\_ShowNum(u8 x,u8 y,u32 num,u8 len,u8 size);//输出数字

void OLED\_ShowString(u8 x,u8 y, u8 \*p,u8 Char\_Size);//输出字符串

void OLED\_Set\_Pos(unsigned char x, unsigned char y);//坐标设置

void OLED\_ShowCHinese(u8 x,u8 y,u8 no);//输出汉字

void OLED\_DrawBMP(unsigned char x0, unsigned char y0,unsigned char x1, unsigned char y1,unsigned char BMP[]);//显示图片

void Oled\_Show\_16(unsigned char x,unsigned y,int temp,unsigned char TextSize)

子模块四：Open MV识别色块及测距设计

Open MV识别红色引导标识可以迅速判别目标所在角度和距离。

red\_threshold = ( 29, 75, 38, 65, -26, 46)//红色阈值

#取像素点最大的色块

def Compare\_Pixel(blob\_a,blob\_b):

temp = blob\_a.pixels() - blob\_b.pixels()

if temp == 0:

return 0;

elif temp > 0:

return 1;

else:

return -1;

while(True):

ball\_cx = 0xfd

ball\_cy = 0xfd

Max\_Blob = None

clock.tick() # Track elapsed milliseconds between snapshots().

img = sensor.snapshot() # Take a picture and return the image.

blobs = img.find\_blobs([red\_threshold])

if len(blobs) == 1:

# Draw a rect around the blob.

b = blobs[0]

for a in blobs:

if Compare\_Pixel(Max\_Blob,b) == -1:

b = a

img.draw\_rectangle(b[0:4]) # rect

img.draw\_cross(b[5], b[6]) # cx, cy

ball\_cx = b.cx()

ball\_cy = b.cy()

Lm = (b[2]+b[3])/2

length = K/Lm

for i in range(11):

length = length + K/Lm -12

length = length / 11

#输出长度

print(length)

Ball\_Pos = bytearray([0xC8, 0xF2, int(ball\_cx), int(ball\_cx>>8)])

uart.write(Ball\_Pos)

### 调试工具及子模块测试方法

调试工具：MDK5/ST-Link/USB数据线

测试方法：

子模块1：打开KEIL5软件，主板连接电脑使用ST-link进行烧录调试，输入角度α使用量角器量取舵机转动的角度，输入PWM值测试舵机转动的角度。

子模块2：打开KEIL5软件，主板连接电脑使用ST-link进行烧录调试，观察云台舵机速度是否由快变慢且是否有自动校准动作。

子模块3：打开KEIL5软件，主板连接电脑使用ST-link进行烧录调试，使用五向键盘输入数据，观察OLED显示屏的显示是否正常，字体大小位置是否和代码有出入。

子模块4：打开Open MV IDE和相应代码，使用USB数据线连接电脑和Open MV，代码烧录完成后，连接Open MV和STM32，观察选择模式4，观察Open MV能否正确识别到红色引导标识，移动引导标识观察舵机能否做出相应的角度反应及OLED显示出来的角度。