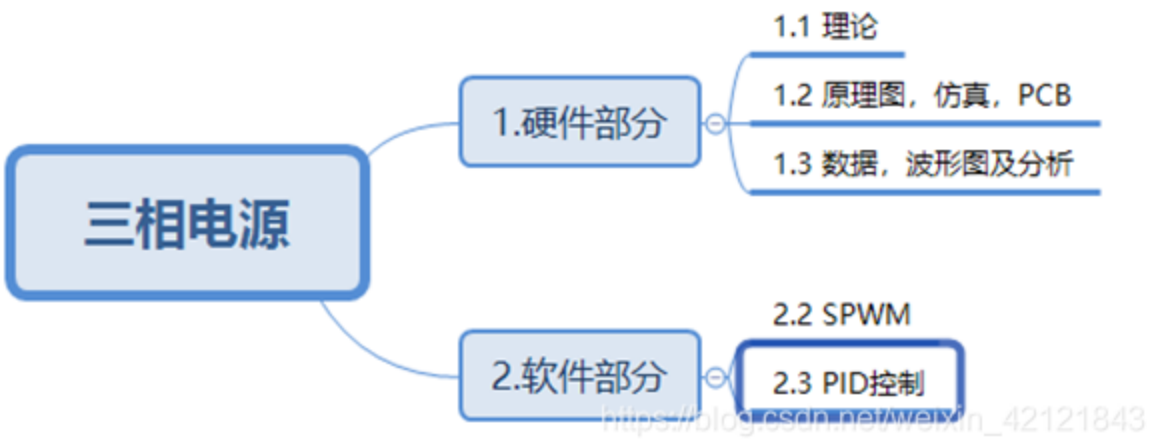
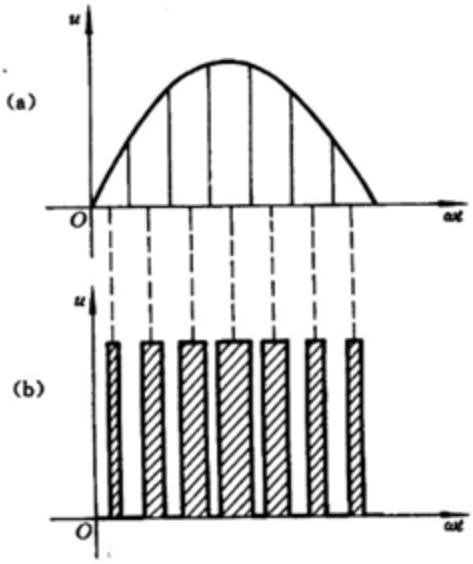
全文框架



1.硬件部分

1.1理论

三相正弦波可以由三路相位不同的正弦波组成，每一路正弦波都可以通过SPWM滤波后产生
SPWM产生原理将正弦曲线用等高的脉冲代替，用等幅不等宽的脉冲可以代替。



采用三角波和正弦波相交的方法可以确定各矩形的宽度，三角波一般用等腰三角形，因为其上下宽度与高度呈线性关系。若在交点时刻控制开关器件的通断，就可以得到宽度正比于正弦波的脉冲。
SPWM经过LC低通滤波后，就会变成正弦波。

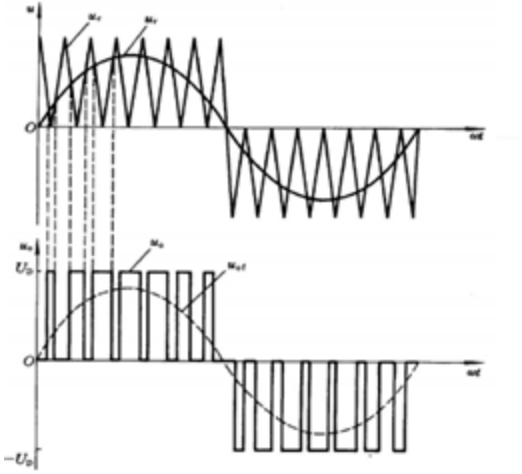


图11.8.3单极性PWM控制方式

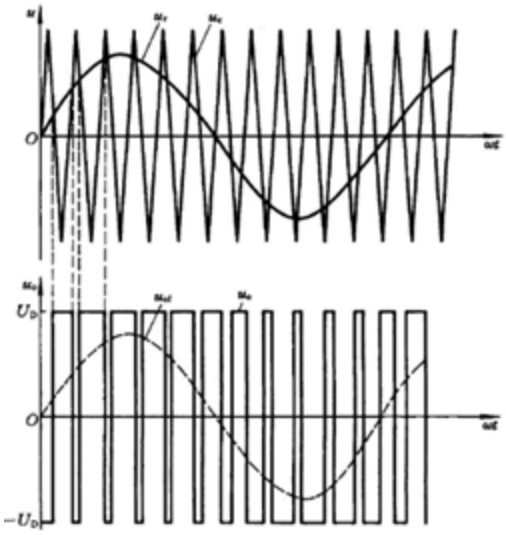
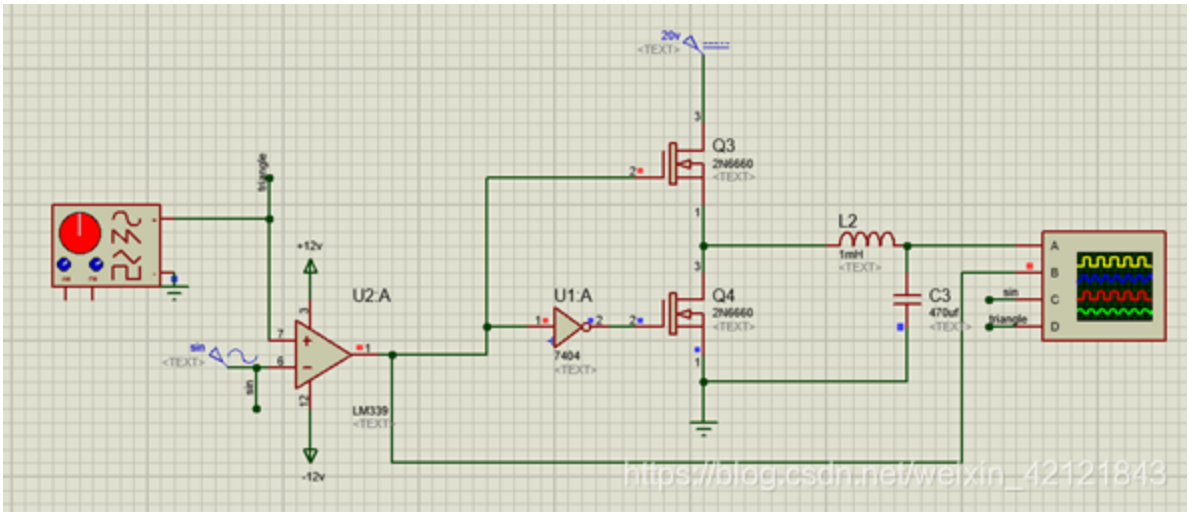
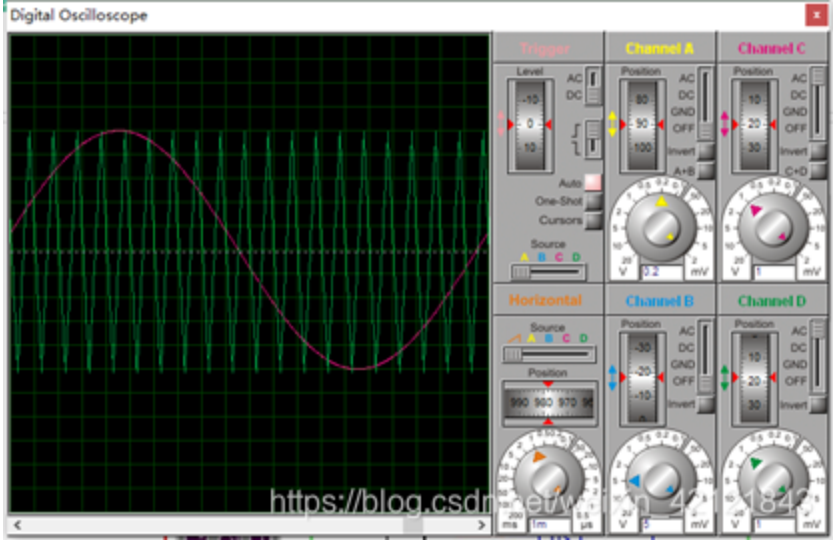
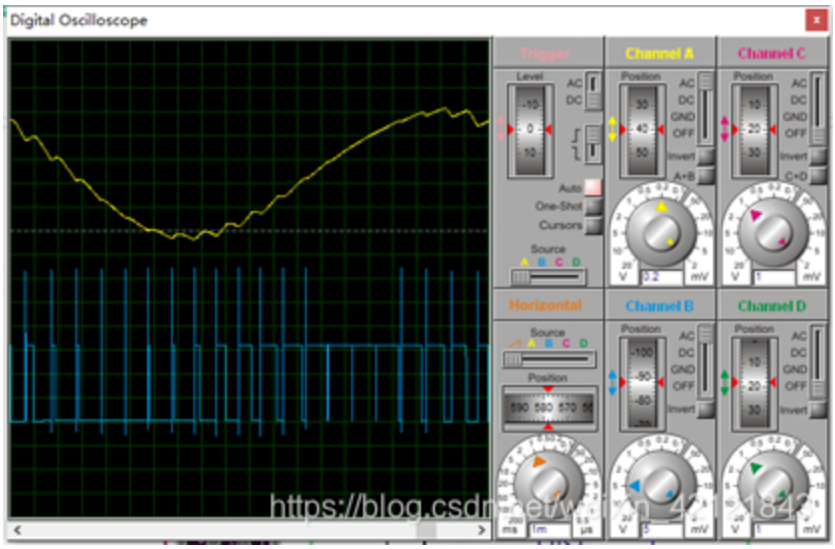


图11.8.4　双极性PWM控制方式的波形

双极性SPWM仿真

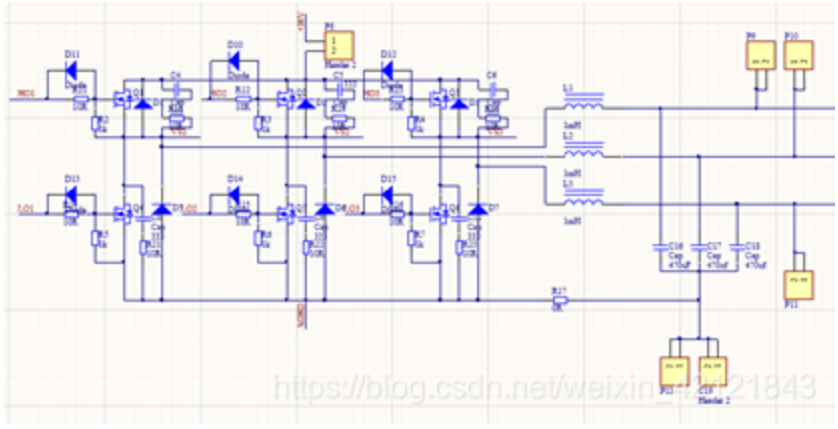


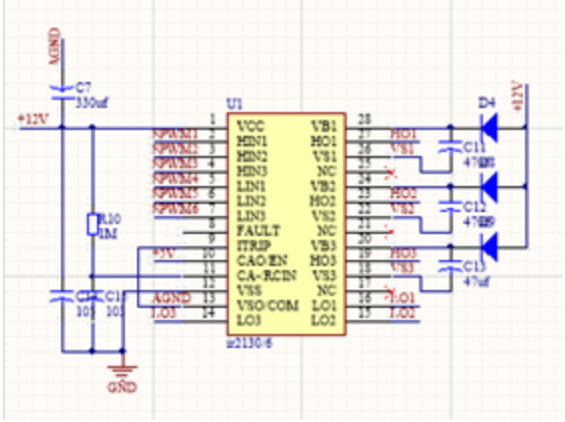




1.2 原理图，仿真，PCB

原理图：





单点接地

就是“模拟地”与“数字地”之间只有一点相连，目的是为了隔离“模拟地”与“数字地”，以免这两种一个充满着高频交流信号、一个工作于直流低频环境的电路产生相互干扰。在某些电路的原理图中我们经常会看到两种地，一种是“模拟地”（图中AGND）、一种是“数字地”（DGND）。

在制作PCB布局时，模拟器件相对集中，放置于与其他板子的接口处，减小信号衰减，数字器件相对集中，放置于板内，两种“地”分别命名，分放于两个区域，分别铺铜，最后将两片“地”用一个10uH的电感或0Ω的电阻连起来，作为共地点，即完成单点接地。

若PCB板进行整板覆铜，如果是这样，“模拟地”和“数字地”混在一起，整个叫做“地”的铜面其实就是一个活生生的电容极板，如果模拟部分工作于高频大信号环境下，由此种混合“地”产生的寄生电容不仅会使高频模拟信号衰减较大，更会使数字部分IO口电平产生起伏。而单点接地的效果便如同数字部分的电源与地之间接了一个滤波电容一般，会大大减轻相互间的干扰。

IR2130驱动电路的原理及仿真

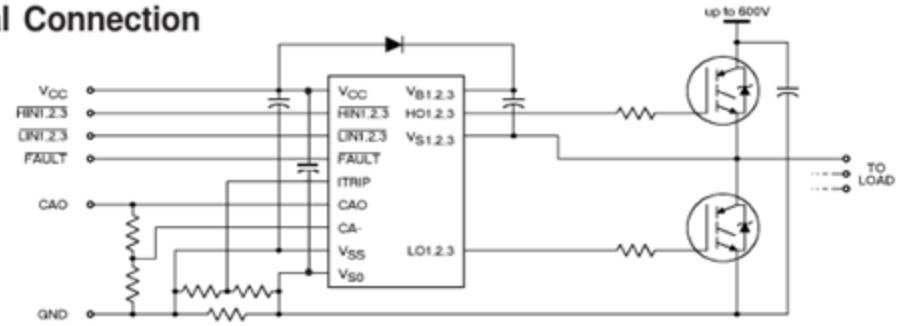
芯片资料如下，可见芯片内部设置了死区，不需要软件设置死区了

3-PHASE BRIDGE DRIVER

Product Summary

V_{OFFSET}	600V max.
I_O+/-	200 mA / 420 mA
V_{OUT}	10 - 20V
$t_{on/off}$ (typ.)	675 & 425 ns
Deadtime (typ.)	2.5 µs (IR2130) 0.8 µs (IR2132)

Typical Connection



(Refer to Lead Assignments for correct pin configuration). This/These diagram(s) show electrical connections only. Please refer to our Application Notes and DesignTips for proper circuit board layout.

VB1～VB3：是悬浮电源连接端，通过自举电容为3个上桥臂功率管的驱动器提供内部悬浮电源，VS1～

VC3是其对应的悬浮电源地端。 HIN1～HIN3、LIN1～LIN3：逆变器上桥臂和下桥臂功率管的驱动信号输入端，低电平有效。
ITRIP：过流信号检测输入端，可通过输入电流信号来完成过流或直通保护
CA-、CAO、Vso：内部放大器的反相端、输出端和同相端，可用来完成电流信号检测。
HO1～HO3、LO1～L03：逆变器上下桥臂功率开关器件驱动器信号输出端。
FAULT：过流、直通短路、过压、欠压保护输出端，该端提供一个故障保护的指示信号。它在芯片内部是漏极开路输出端，低电平有效。Vcc、Vss：芯片供电电源连接端，Vcc接正电源，而Vss接电源地。
正常工作时，输入的6路驱动信号经输入信号处理器处理后变为6路输出脉冲，驱动下桥臂功率管的信号L1～L3经输出驱动器功放后，直接送往被驱动功率器件。而驱动上桥臂功率管的信号H1～H3 先经集成于IR2130内部的3个脉冲处理器和电平移位器中的自举电路进行电位变换， 变为3路电位悬浮的驱动脉冲，再经对应的3路输出锁存器锁存并经严格的驱动脉冲与否检验之后，送到输出驱动器进行功放后才加到被驱动的功率管。一旦外电流发生过流或直通，即电流检测单元送来的信号高于0.5V时，则IR2130内部的电流比较器迅速翻转，促使故障逻辑处理单元输出低电平，一则封锁3路输入脉冲信号处理器的输出，使IR2130的输出全为低电平，保护功率管；另一方面，同时IR2130的FAULT脚给出故障指示。同样若发生IR2130的工作电源欠压，则欠压检测器迅速翻转，也会进行类似动作。发生故障后，IR2130内的故障逻辑处理单元的输出将保持故障闭锁状态。直到故障清除后，在信号输入端LIN1～LIN3同时被输入高电平，才可以解除故障闭锁状态。

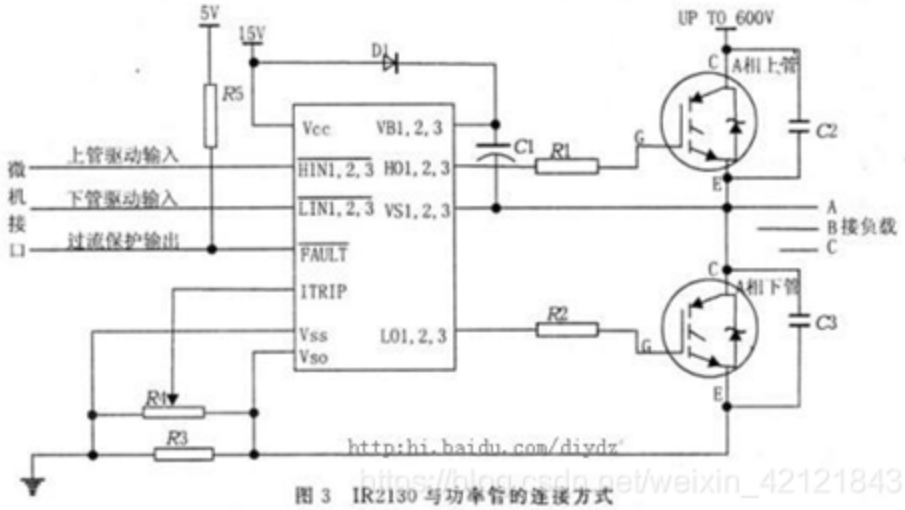
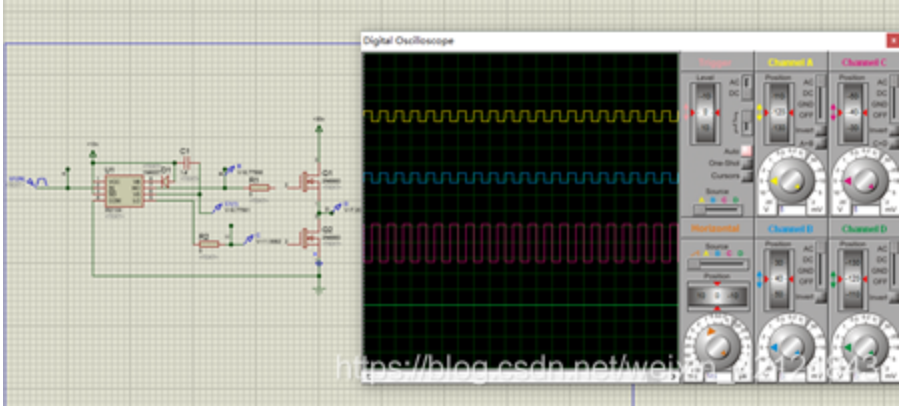


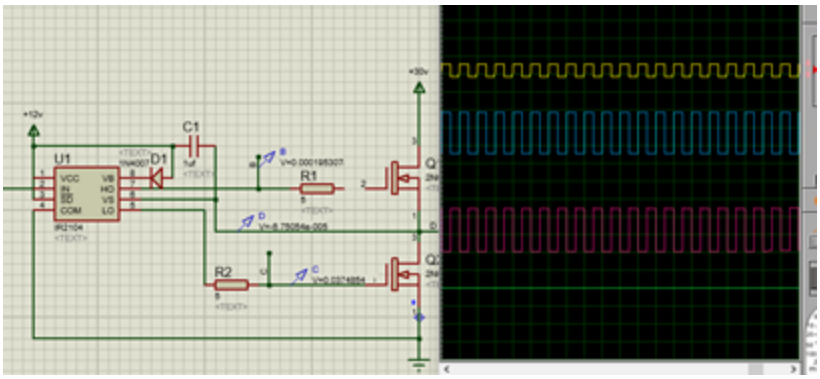
图 3 IR2130 与功率管的连接方式

图中C1是自举电容，为上桥臂功率管驱动的悬浮电源存储能量，D1的作用防止上桥臂导通时的直流电压母线电压到IR2130的电源上而使器件损坏，因此D1应有足够的反向耐压，当然由于D1与C1串联，为了满足主电路功率管开关频率的要求，D1应选快速恢复二极管。R1和R2是IGBT的门极驱动电阻，一般可采用10到几十欧。R3和R4组成过流检测电路，其中R3是过流取样电阻，R4是作为分压用的可调电阻。IR2130的HIN1～HIN3、LIN1～LIN3作为功率管的输入驱动信号与单片机连接，由单片机控制产生PWM控制信号的输入，FAULT与单片机外部中断引脚连接，由单片机中断程序来处理故障。

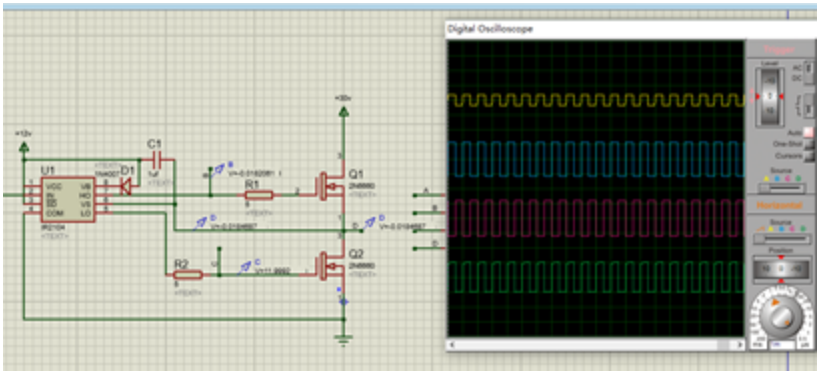
驱动仿真（用IR2104代替）：
芯片与MOS管断开时，其LO端输出正常，但HO和VS无输出



VS和LO接上后，HO不接，则HO和LO波形正常，VS无波形（HO正常是因为VS反馈给自举电容，VS无输出因为上桥臂未导通）

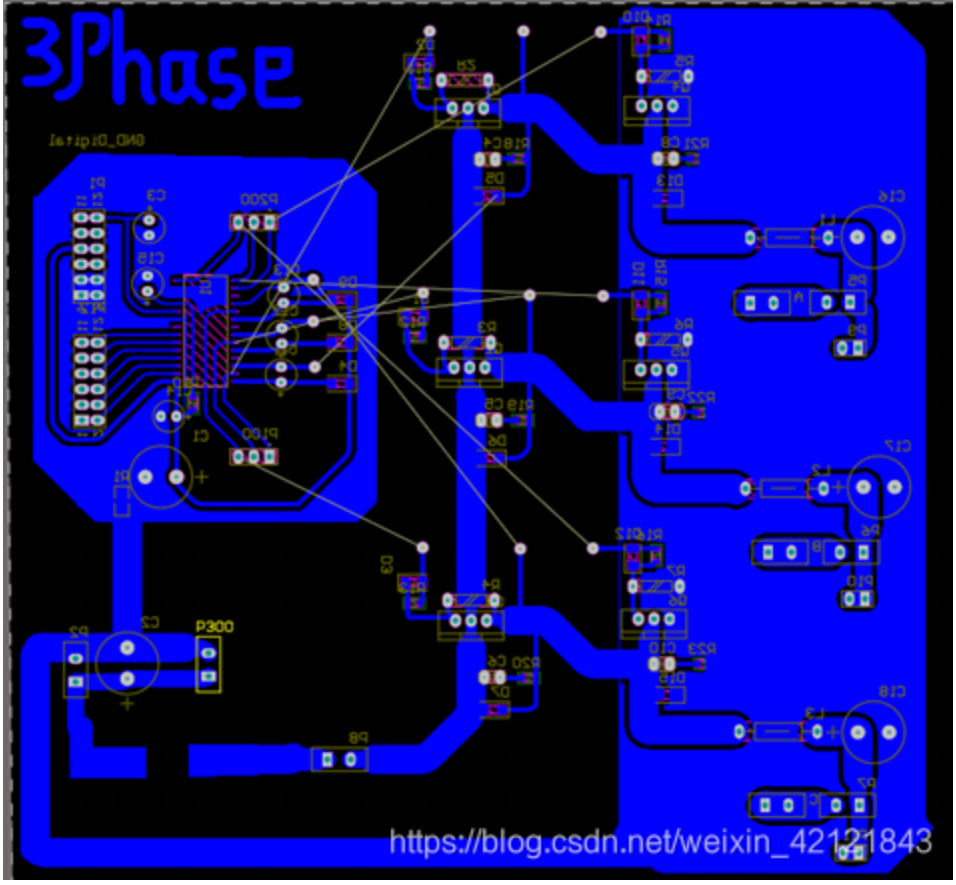


正常输出时：



1.3 数据，波形图及分析（结合PCB设计）

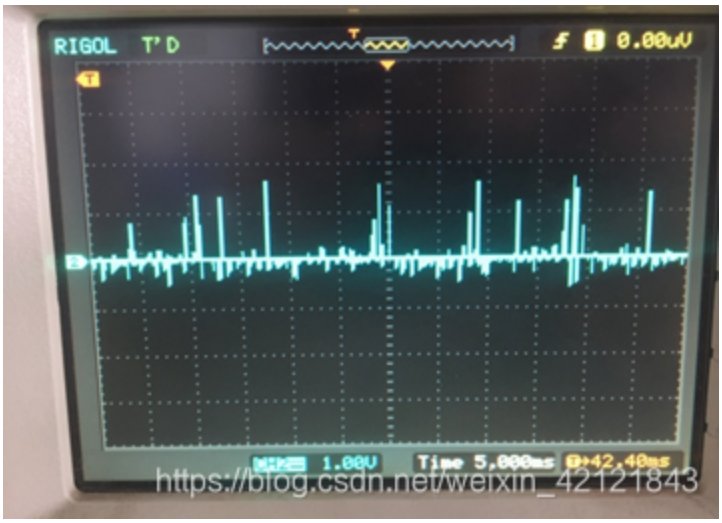
PCB板1



PCB1波形

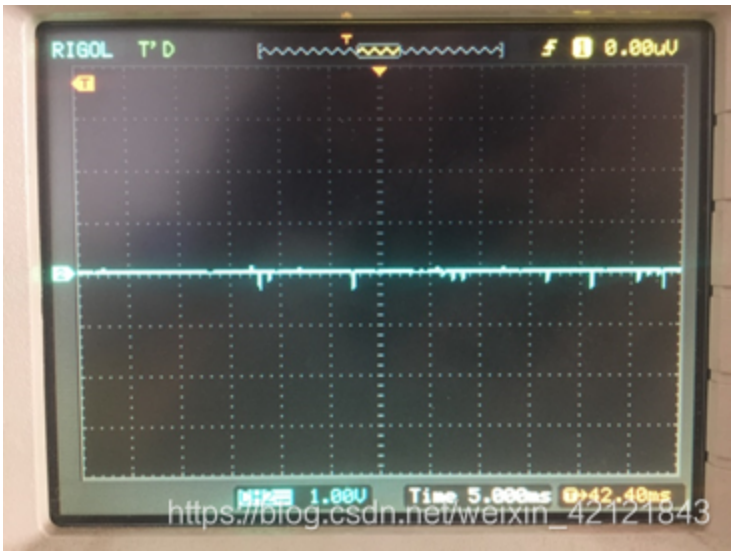
铺铜有弊端，不能整块铺，要分部分。高频部分不要走地线。模拟地和数字地要分开。

地线受到严重干扰后，导致输出有很大的干扰：下图为模拟地，正弦波上也有该噪声

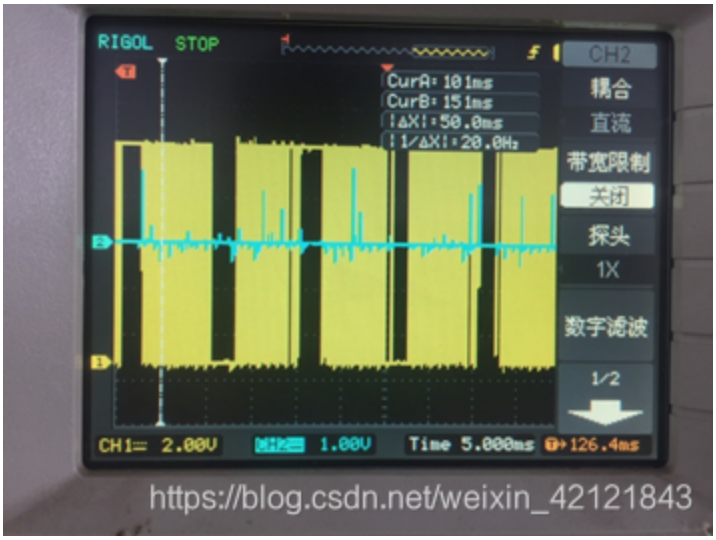


输出正弦波有很多高频分量：

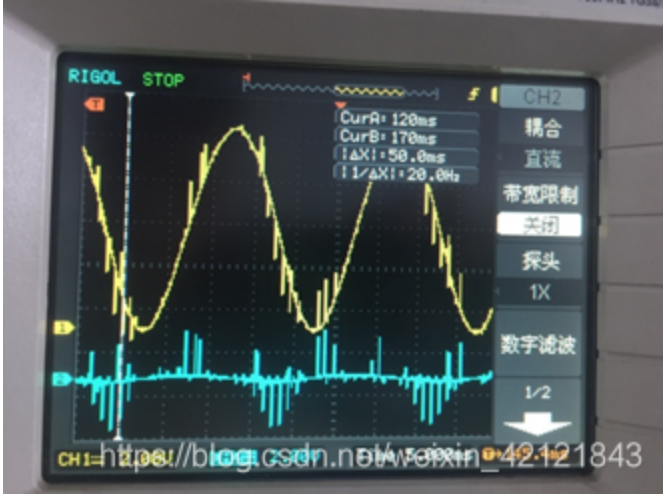
数字地正常，因与模拟地分开了，但是高频信号附近的地线还是有干扰。



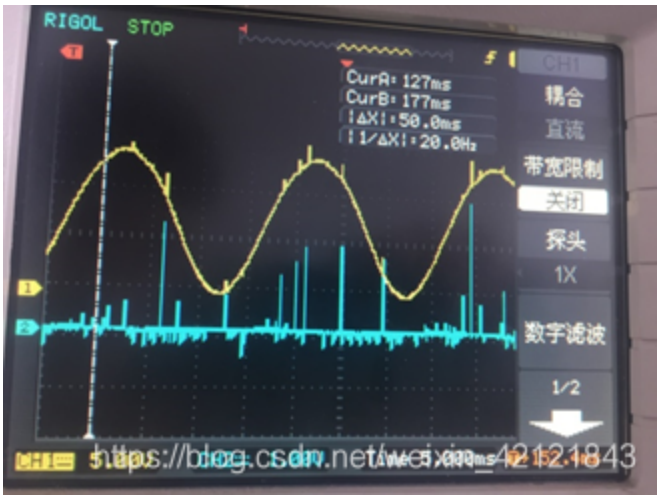
VS输出端（黄色）与与地线（蓝色）干扰无关



Ic滤波输出端和地线（蓝色）干扰有关，蓝色干扰周期约为20ms，为脉冲形状，猜测为VS的SPWM造成的干扰



划除部分敷铜后，噪声减少，但仍然存在



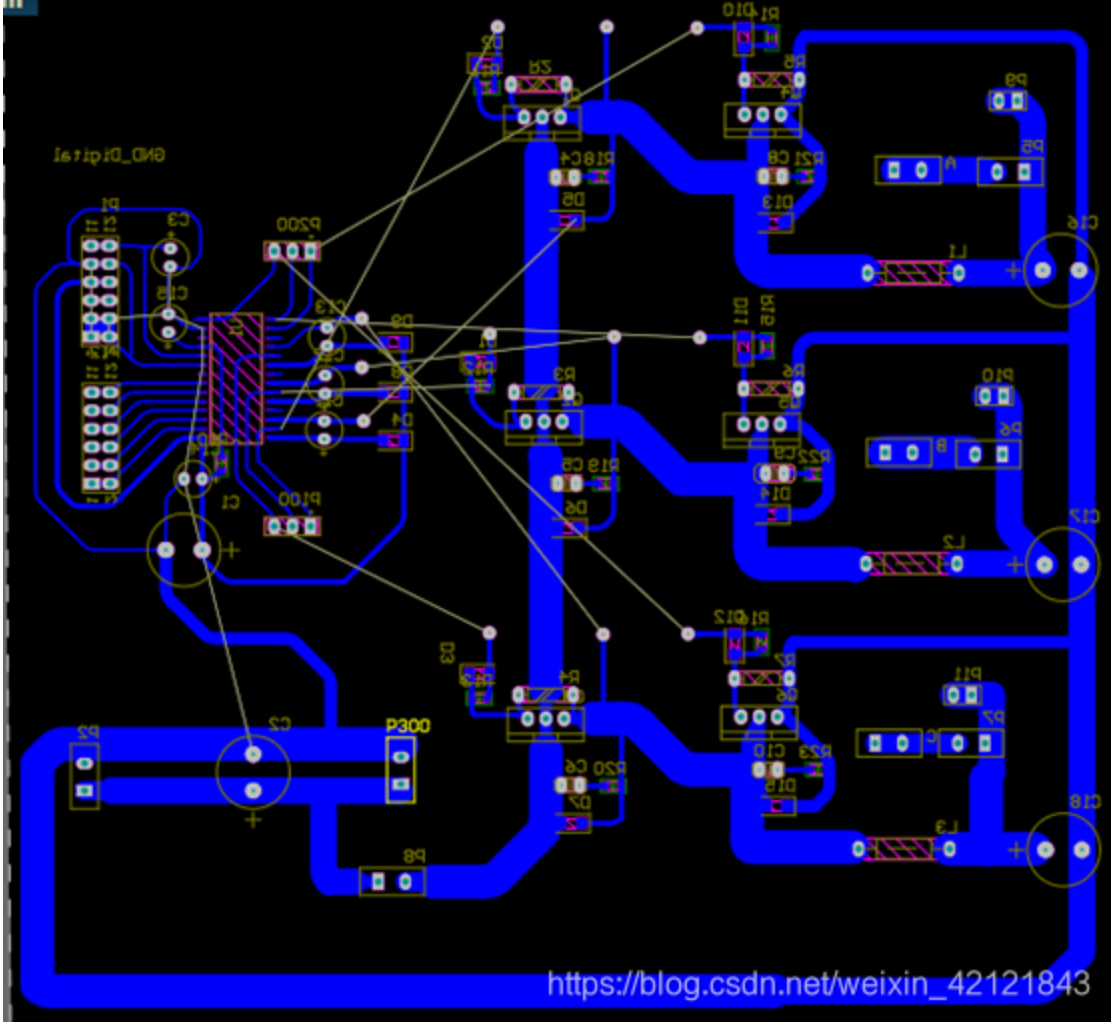
PCB1 总结分析

1.地线远离干扰，且不同部分的地线分开，防止干扰
2.注意不要随意铺铜，干扰会很大。
3.地线干扰随着远离干扰源会慢慢减小，因为走线的阻抗会降低噪声
4.滤波部分电感和电容的值不能只通过计算得到，要仿真看波形，还要实地测试。经测试，电感太大的时候，输出电流增大，效率严重降低
5.保险丝电阻约1欧，注意功耗
6.高频感应部分若是接近地线，感应会消耗很多能量，影响效率

附：布线注意事项

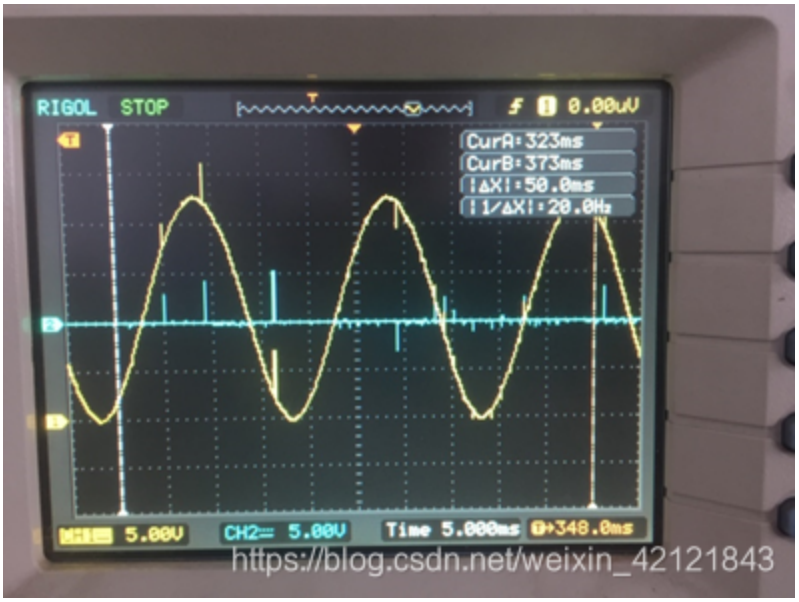
1.焊盘尽量大，2.5mm*2.5mm，或是1.75mm*2.5mm椭圆形
2.线距25mil，0.635mm左右。过细在焊接时容易短接。
3.贴片芯片引脚要引出来一点
4.大电容电感的空间位置要留够。

PCB板2



PCB板2波形：

输出波形（黄色）仍然有干扰，蓝色为地线。干扰相对于上板有所减少，但是仍然很大。

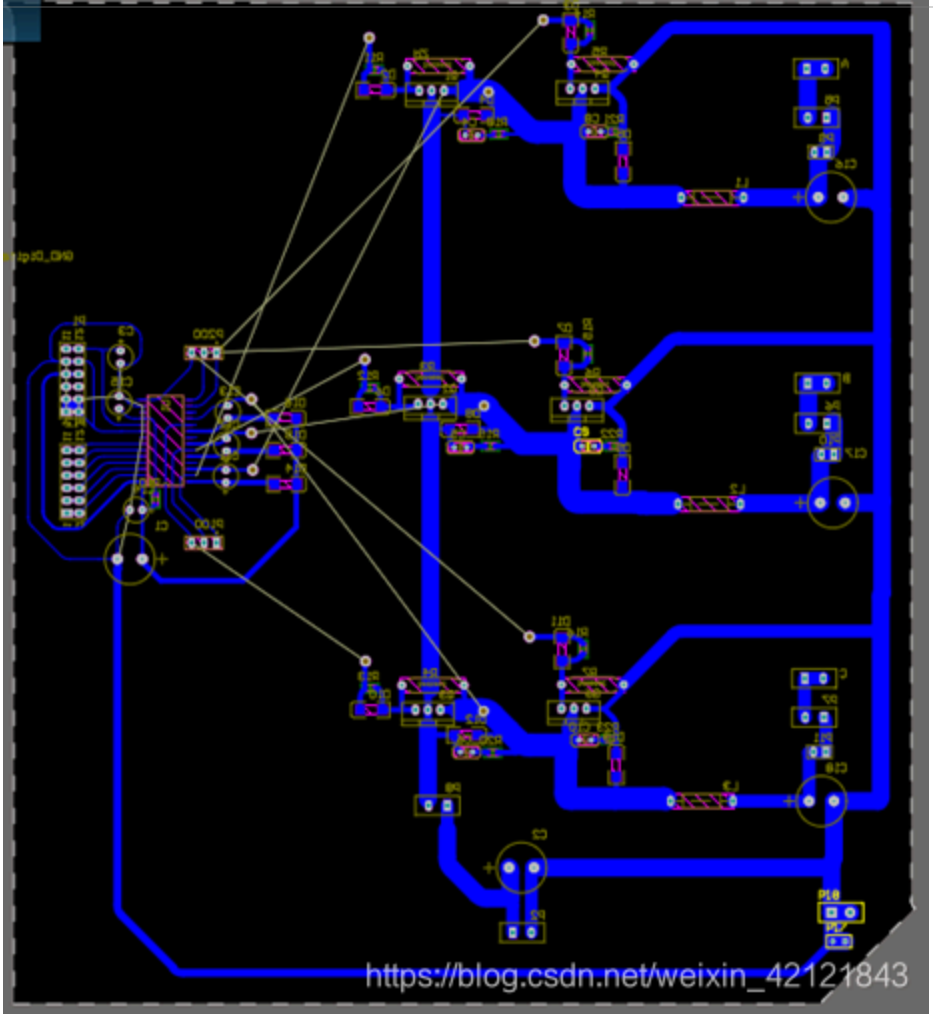


PCB板2总结分析：

1.大功率回路不要过长，这个PCB的地线过长

2.电感要远离其他回路，包括地线，输出

PCB板3



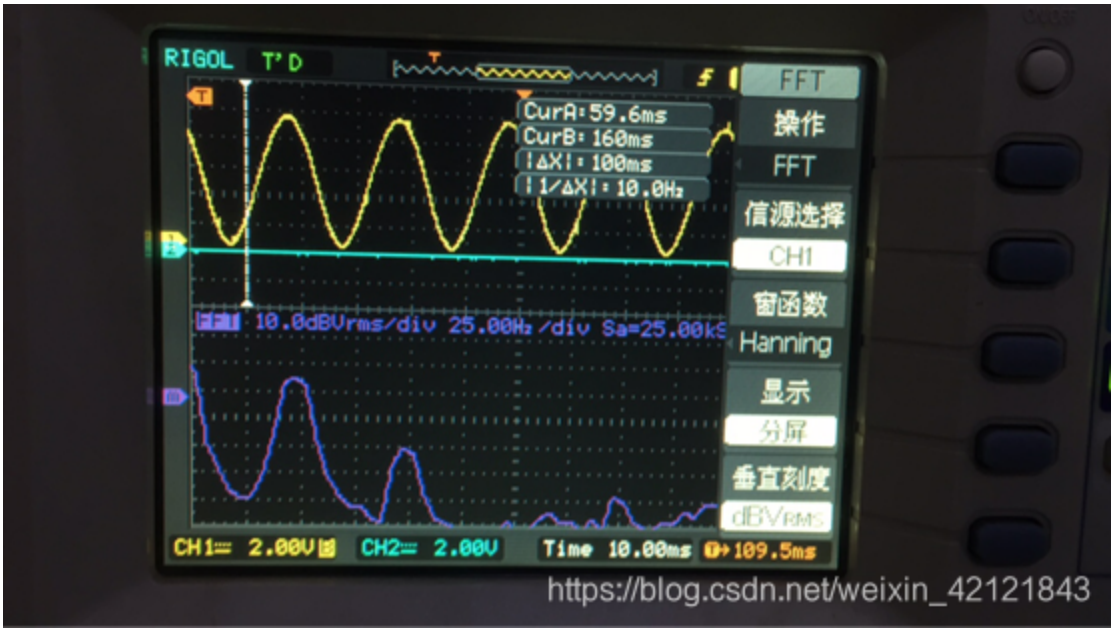
PCB板3波形

PCB改进：

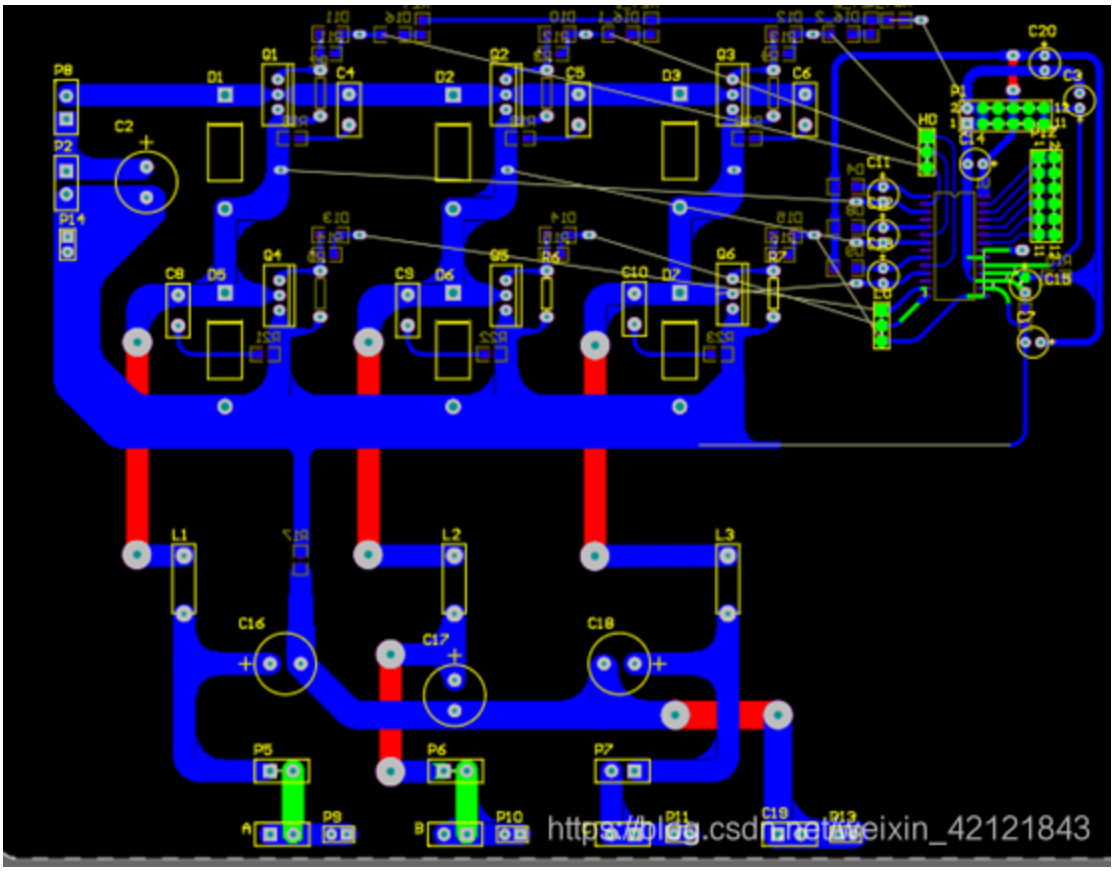
增大VS端，电感与其他器件和走线的距离，避免干扰

不足：最后一路的电感和GND总线距离太近

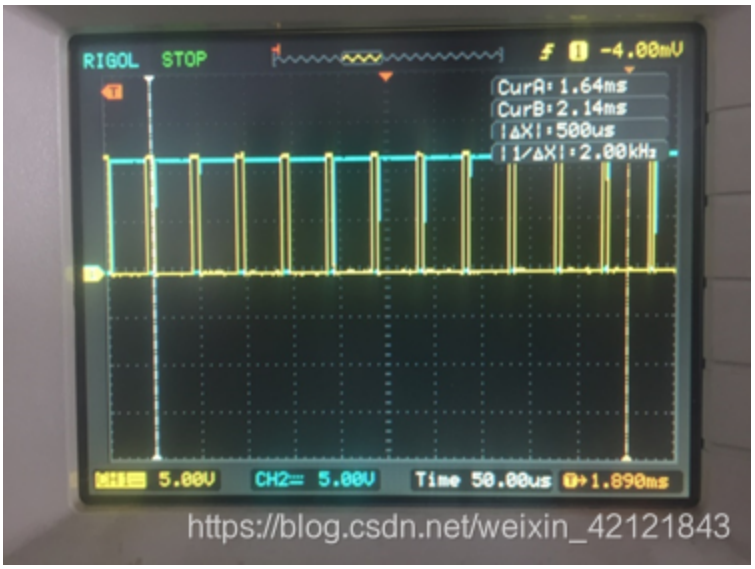
波形和FFT如下，谐波量已经减少到合理水平



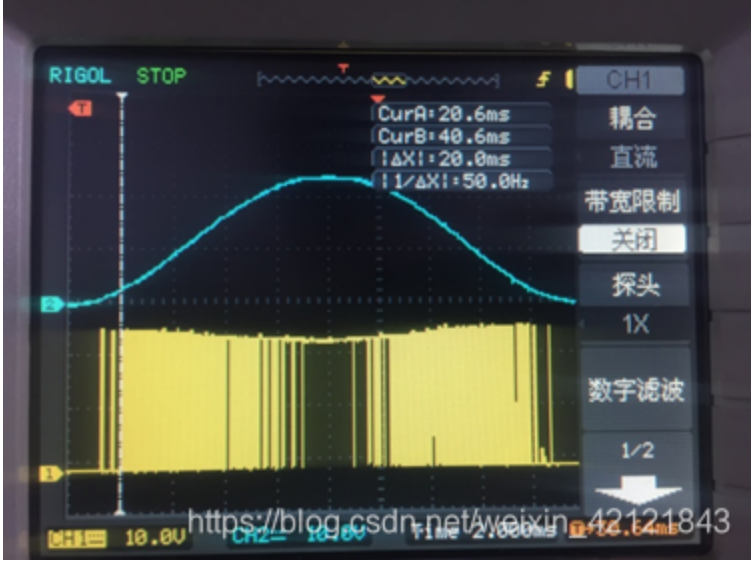
PCB板4



驱动输出的互补SPWM波：



VS输出的SPWM以及经过电感电容滤波后的50hz正弦波：



输出信号的波形以及FFT后得到的频谱，有直流分量，因为参考点为GND，没有负的电压

PCB板4总结分析：

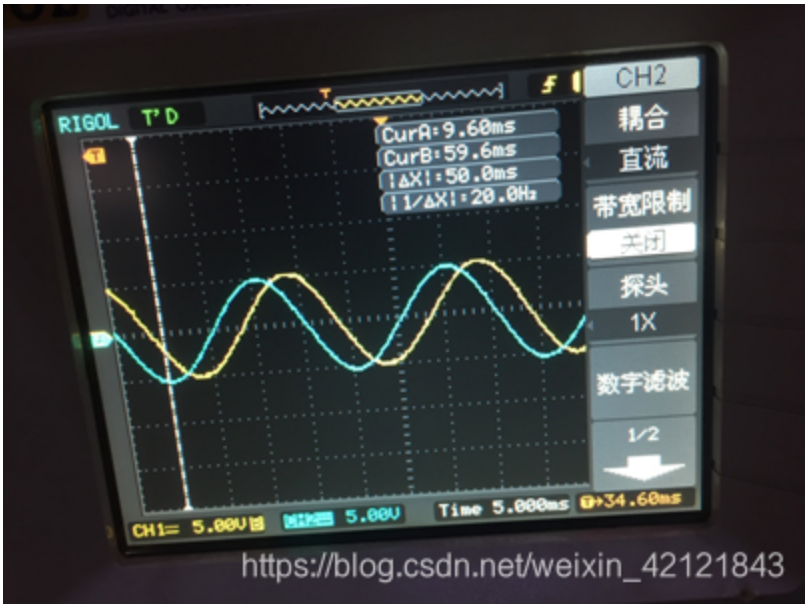
1.单片机，驱动芯片，电源注意要共地，否则可能出现：电压基准线不在原GND处，显示多了直流量，驱

动芯片HO输出随供电电压变化

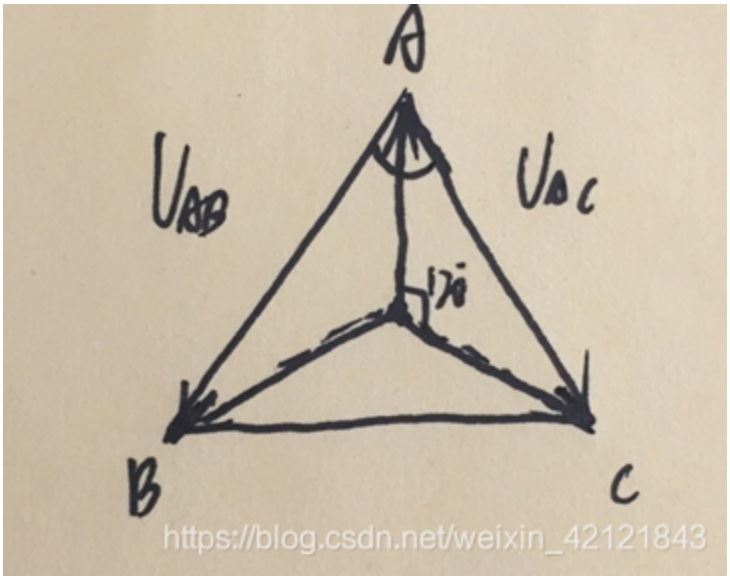
2.驱动芯片记录：不接MOS时，LO正常输出，HO高电平，但是电压缓慢下降（经测试为自举电容电压下降导致的，因下桥臂没有给自举电容充电）

PCB板4负载特性测试

经测试，线电压相差六十度（Uab和Uac）



原因分析如下：因为示波器的两探头的地线只能连在一个GND，故只能以某一相为参考点



功率测量原理：

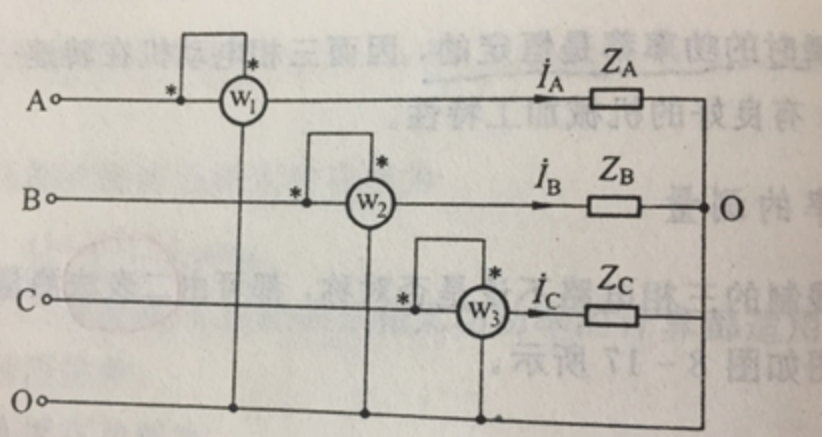


图 8-18　三相四线制三相电路功率测量

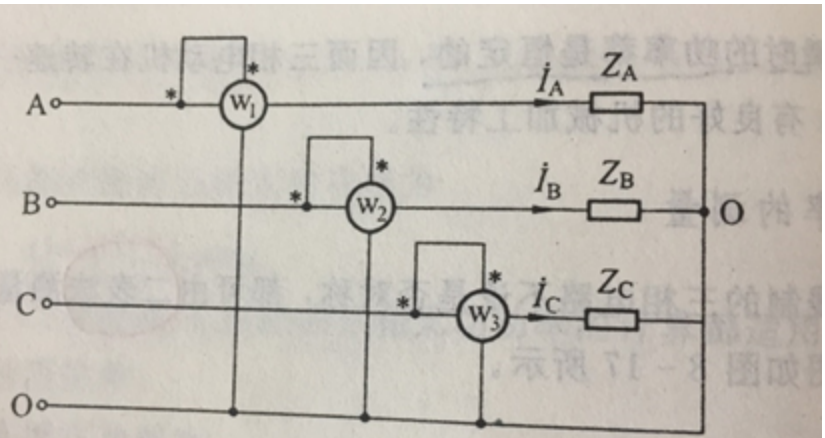
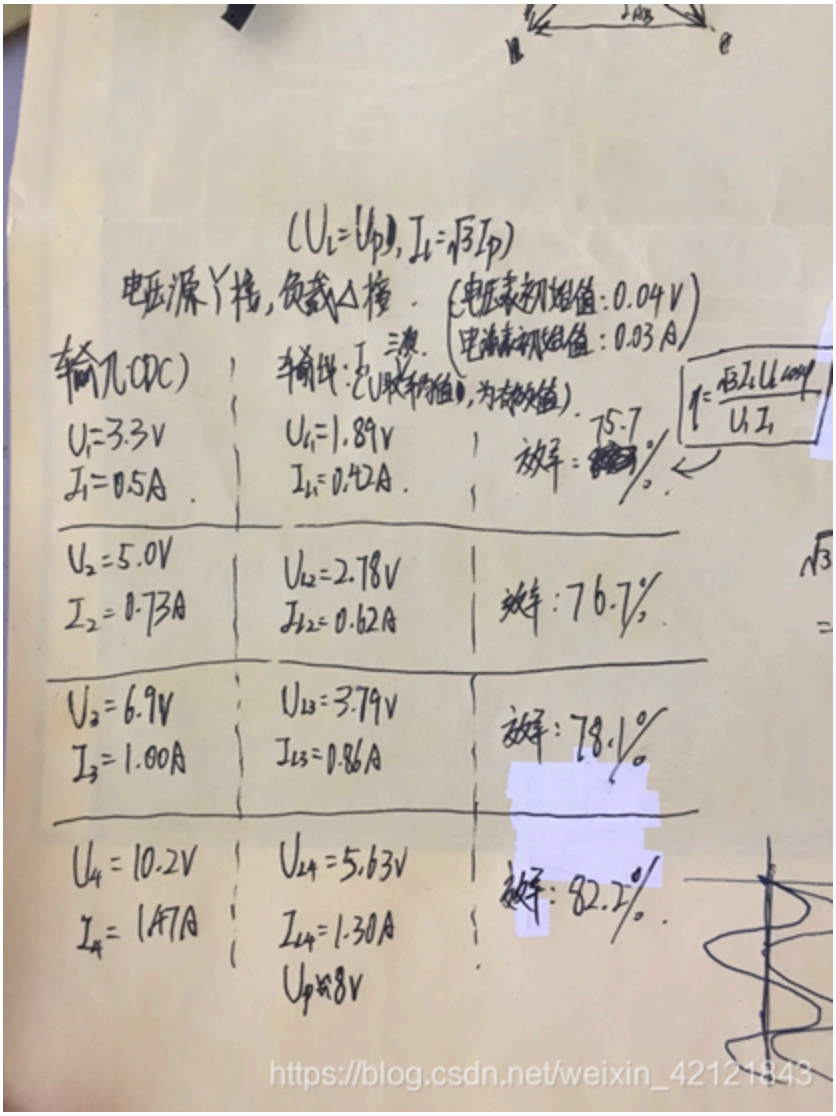


图 8-18　三相四线制三相电路功率测量

https://blog.csdn.net/weixin_42121843

PCB板4测量数据：



2.软件笔记

2.1 SPWM

原理：
如果要产生50HZ的正弦波，假设半个周期中采样100个点，则每个点的采样周期为10ms/100=0.1ms
PWM配置的输出频率则为f=10K，
如果PWM周期值配置为500即最大分辨率(计数器记到500对应频率为10K，所以这里要根据实际CPU进行分频，该定时器的频率应该设置为5M),定时器的计数器计时到500时，即输出一个f=10K的PWM波，输出PWM的占空比由产生的spwm表中的值决定，SPWM表中的值直接设置到比较寄存器中即可控制输出占空比，实际应用时，对应的100个采样点要产生100次中断，在每次中断完成后将对应的下一个点中的值设置到比较寄存器中，从而控制下一个输出PWM波时的占空比，当该100个点全部输出完时，则输出的电压经过硬件滤波就能得到50HZ的半个周期的正弦波。

代码（只截取了重要片段）

```
//标准正弦函数一个周期的值，210个，从0到400
int wave[210]={200,206,212,218,224,230,236,242,247,253,
        259,265,270,276,281,287,292,297,303,308,313,318,322,
        327,332,336,340,345,349,353,356,360,364,367,370,373,
        376,379,381,384,386,388,390,392,394,395,396,397,398,
        399,399,400,400,400,400,399,399,398,397,396,395,394,
        392,390,388,386,384,381,379,376,373,370,367,364,360,
        356,353,349,345,340,336,332,327,322,318,313,308,303,
        297,292,287,281,276,270,265,259,253,247,242,236,230,
        224,218,212,206,200,194,188,182,176,170,164,158,153,
        147,141,135,130,124,119,113,108,103,97,92,87,82,78,
        73,68,64,60,55,51,47,44,40,36,33,30,27,24,21,19,16,
        14,12,10,8,6,5,4,3,2,1,1,0,0,0,0,1,1,2,3,4,5,6,8,10,
        12,14,16,19,21,24,27,30,33,36,40,44,47,51,55,60,64,
        68,73,78,82,87,92,97,103,108,113,119,124,130,135,141,
        147,153,158,164,170,176,182,188,194};

//三相对称交流电起始位置
int i=0;
int j=70;
int k=140;

double Vol_set=1;
float PID_val=1;

//时钟设置，时钟1和8设置成PWM，时钟2设置成中断
TIM1_PWM_Init(400-1,20-1);   //自动重装载值399，分频19，输出频率约21kHZ
TIM8_PWM_Init(400-1,20-1);   //与TIM1互补
TIM2_Int_Init(400-1,20-1);

/**************************************************/主函数片段
while(1)
        {
                adcx=Get_Adc_Vpp(ADC_Channel_5,200);    //获取峰峰值
                temp=(float)adcx*(3.3/4096);
                if((temp-Vol_set)/Vol_set>=0.05||(temp-Vol_set)/Vol_set<=-0.05)
                {
                        PID_con=temp;
                        temp_p=temp;
                        while((temp_p-Vol_set)/Vol_set>0.05||(temp_p-Vol_set)/Vol_set<-0.05
                        {
                                LED0=0;
                                PID_con=PID_realize(Vol_set,temp_p);                    //位
                                PID_val=PID_con/temp;
                                adcx=Get_Adc_Vpp(ADC_Channel_5,120);            //获取峰峰值
                                if(PID_val>1)PID_val=1;
```

```
                        if(PID_val<0)PID_val=0;
                        delay_us(50);
                }
        }
        else
        {
                LED0=1;
                delay_us(50);
        }
}
/*****************************************************/求峰峰值
u16 Get_Adc_Vpp(u8 ch,u8 times)
{
        u32 temp_val=0;
        u32 max=0;
        u32 min=0;
        u8 t;
        temp_val=Get_Adc(ch);
        max=temp_val;
        min=temp_val;
        for(t=0;t<times-1;t++)
        {
                temp_val=Get_Adc(ch);
                if(temp_val>max)
                        max=temp_val;
                if(temp_val<min)
                        min=temp_val;
                delay_us(500);
        }
        return max-min;
}
/*****************************************************/定时器2中断函数
void TIM2_IRQHandler(void)
{
        if(PID_val>1)PID_val=1;
        if(PID_val<0)PID_val=0;

        //PID_val=1;
        if(TIM_GetITStatus(TIM2,TIM_IT_Update)==SET)
        {
                TIM_SetCompare1(TIM1,wave[i]*PID_val);  //修改占空比
                TIM_SetCompare2(TIM1,wave[j]*PID_val);
                TIM_SetCompare3(TIM1,wave[k]*PID_val);
                TIM_SetCompare1(TIM8,wave[i]*PID_val);
                TIM_SetCompare2(TIM8,wave[j]*PID_val);
                TIM_SetCompare3(TIM8,wave[k]*PID_val);
```

```
                i++;
                j++;
                k++;
                if(i==210)i=0;
                if(j==210)j=0;
                if(k==210)k=0;
        }
        TIM_ClearITPendingBit(TIM2,TIM_IT_Update);  //清除中断标志位
}
```

SPWM产生及控制原理

开始时，PID_val为1，若反馈的temp不处于0.95~1.05 vol_set的范围内，则启用PID调节。当输出正弦波小于设定值时，PID输出正值，PID_val>0（但是不能超过1，保证其输出SPWM不失真，因此电源电压不够的时候，PID_val恒为1，且输出达不到设定值）,定时器2中，TIM1，TIM8正常按照wave数组输出。输出正弦波大于设定值时，PID输出负值，PID_val=0，输出为0，直到降回正常值。
注意，wave数组乘了一个小于1的比例系数之后，仍然为spwm，但是幅值会减小（占空比变小）。

2.2PID算法

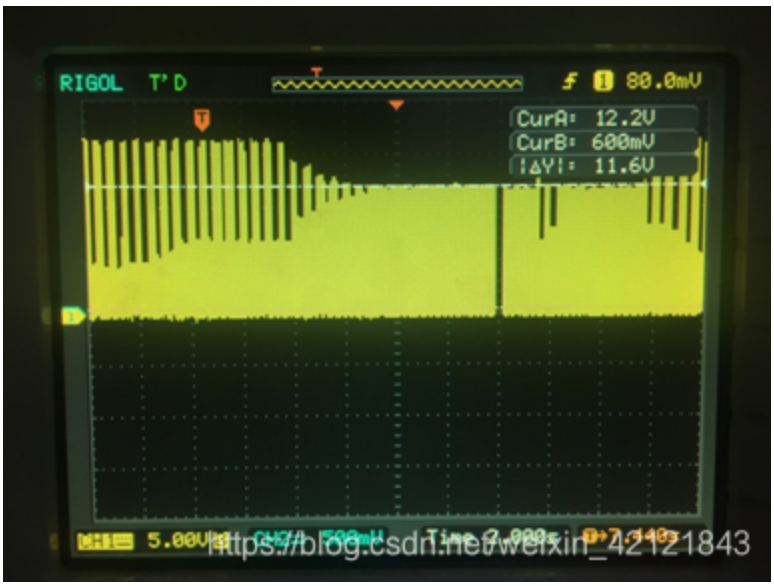
```
struct _pid{
        float SetVol;           //设定值
        float ActVol;           //实际值
        float Err;                      //误差
        float   Err_Last;                       //上一个误差
        float Kp,Ki,Kd;                 //PID参数
        float actuator;                 //控制器执行变量
        float integral;                 //积分值
}pid;

void PID_init(void)   //PID初始化函数
{
        pid.SetVol=0.0;
        pid.ActVol=0.0;
        pid.Err=0.0;
        pid.Err_Last=0.0;
        pid.actuator=0.0;
        pid.integral=0.0;
        pid.Kp=KP;
        pid.Ki=KI;
        pid.Kd=KD;
}
```

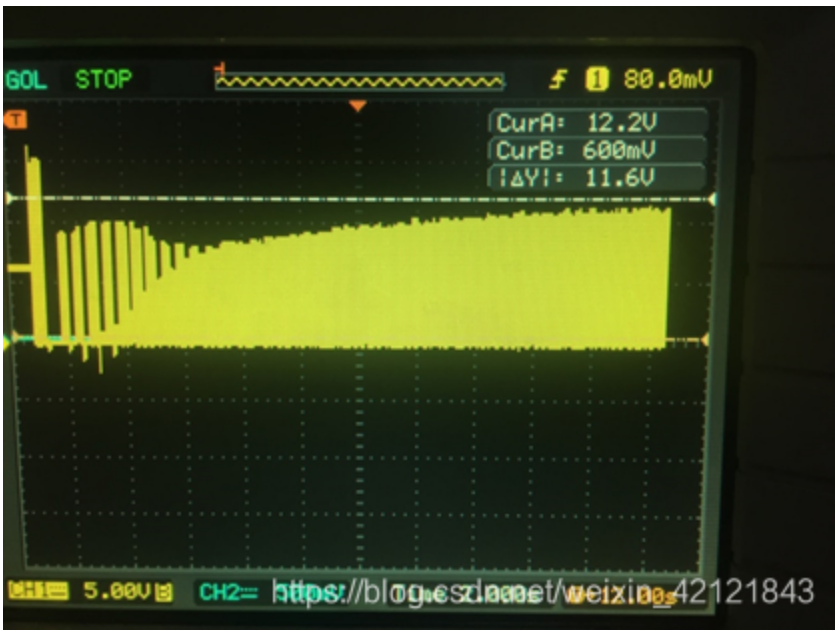
```
float PID_realize(float voltage,float adc_val)                    //位置型PID实现
{
        pid.SetVol=voltage;             //理论值
        pid.ActVol=adc_val;             //实际值
        pid.Err=pid.SetVol-pid.ActVol;  //计算误差
        pid.integral+=pid.Err;                                    //计算积分
        pid.actuator=pid.Kp*pid.Err+pid.Ki*pid.integral+pid.Kd*(pid.Err-pid.Err_Last);  //公
        pid.Err_Last=pid.Err;
        return pid.actuator;
}
```

数据及波形（调整PID参数时）

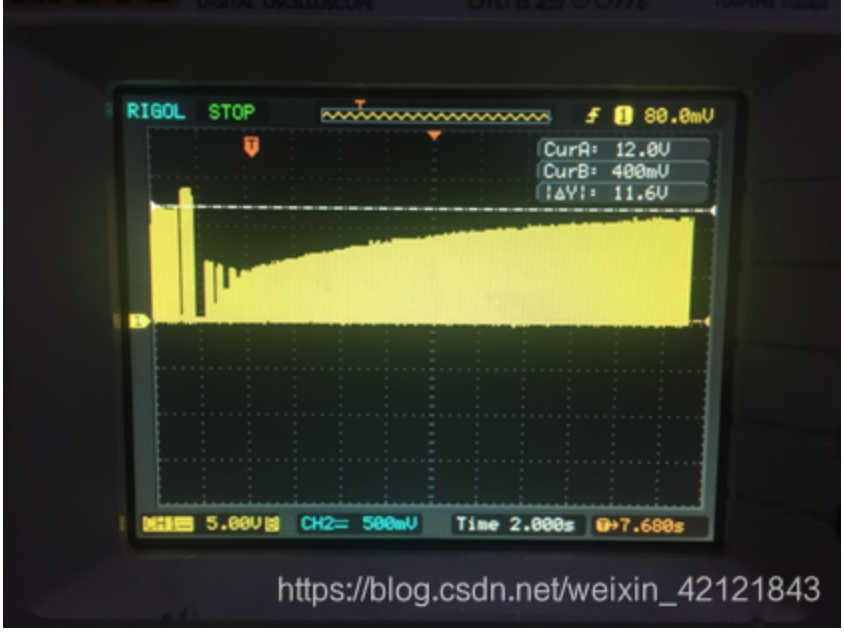
有超调量（P=0.7,I=0.1,D=0）



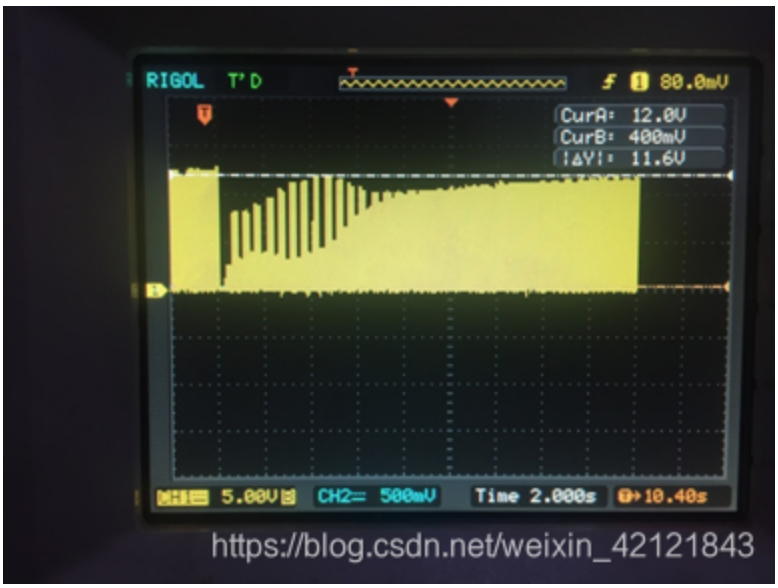
（P=0.7,I=0.1,D=0.1）



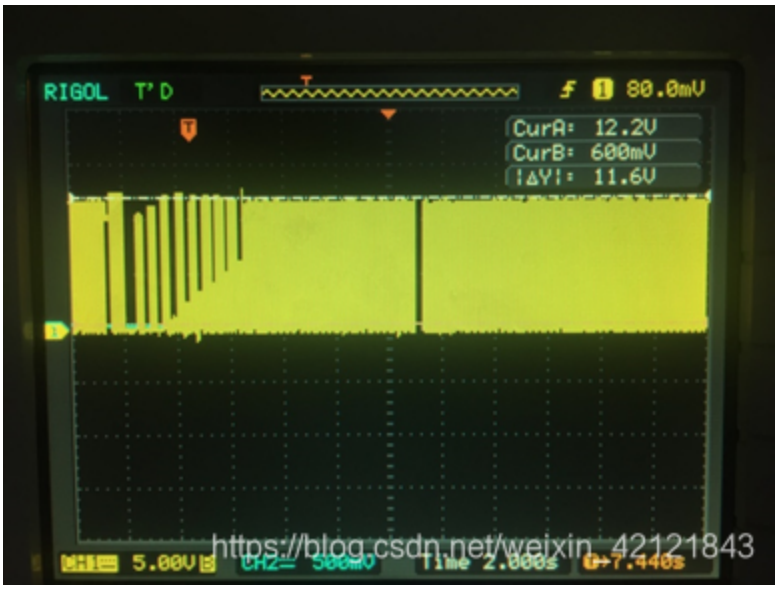
（P=0.7,I=0.05,D=0.05）



P=0.7,I=0.1,D=0.15)



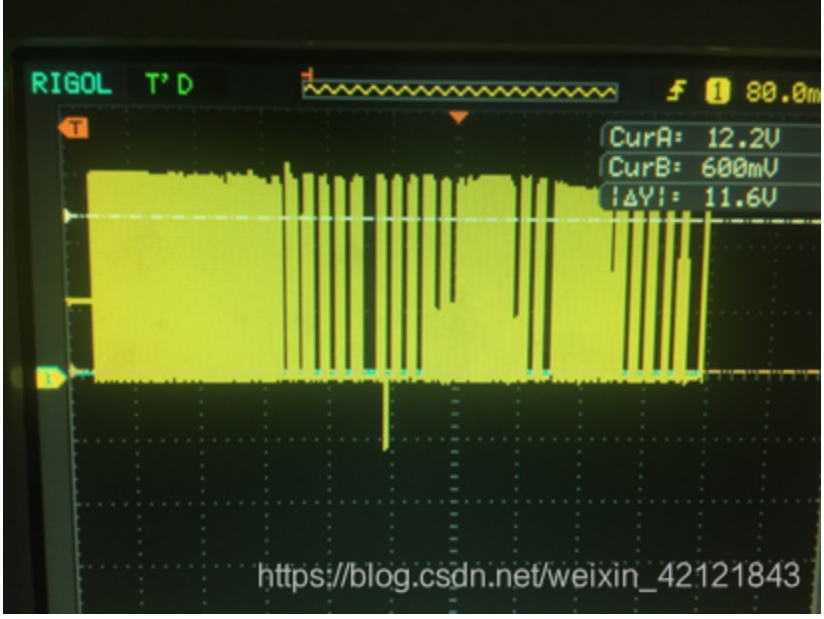
(P=0.7,I=0.15,D=1)



波形分析

1.P过大的时候，容易引起失控（震荡发散）

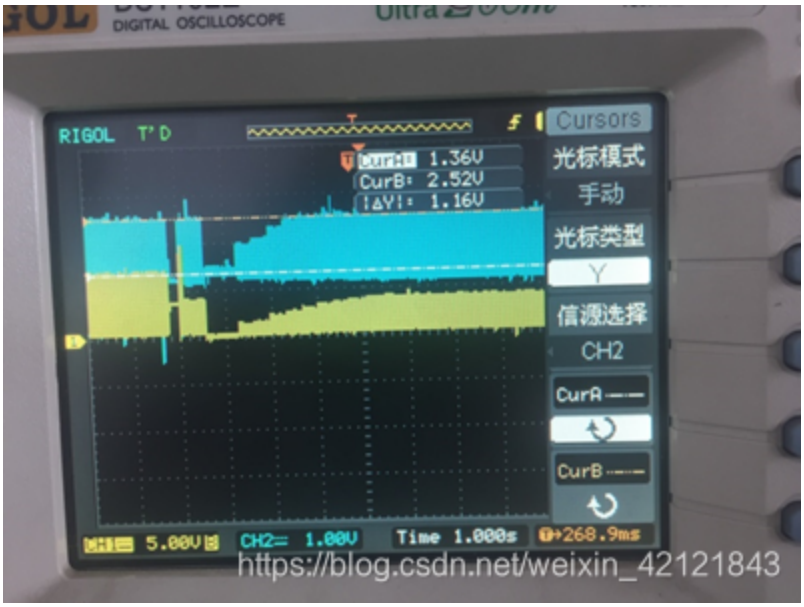
2.要保证电压和电流的平稳，宁愿慢也要稳，否则器件容易损坏。P和I都要小



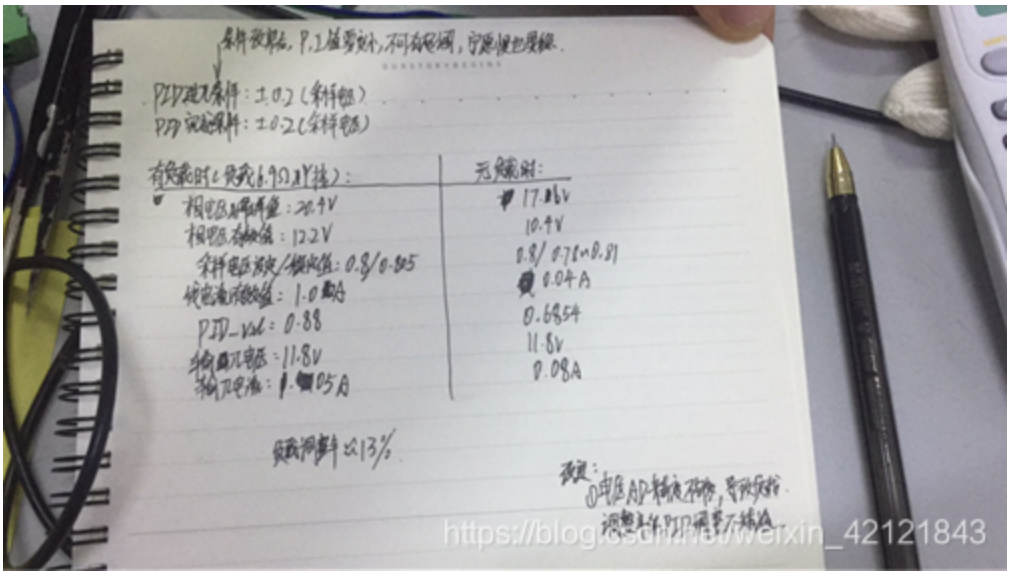
负载调整率测试

注意事项：PID调节不可有超调，要过阻尼状态，平稳缓慢上升，否则系统不稳定，容易烧板子

如下图所示，黄色为电压，蓝色为电流



负载调整率测试记录：



分析：

1.负载调整率高，原因分析为AD采样不准确，误差过大。故采用AD637模块测量之。具体见模块笔记。