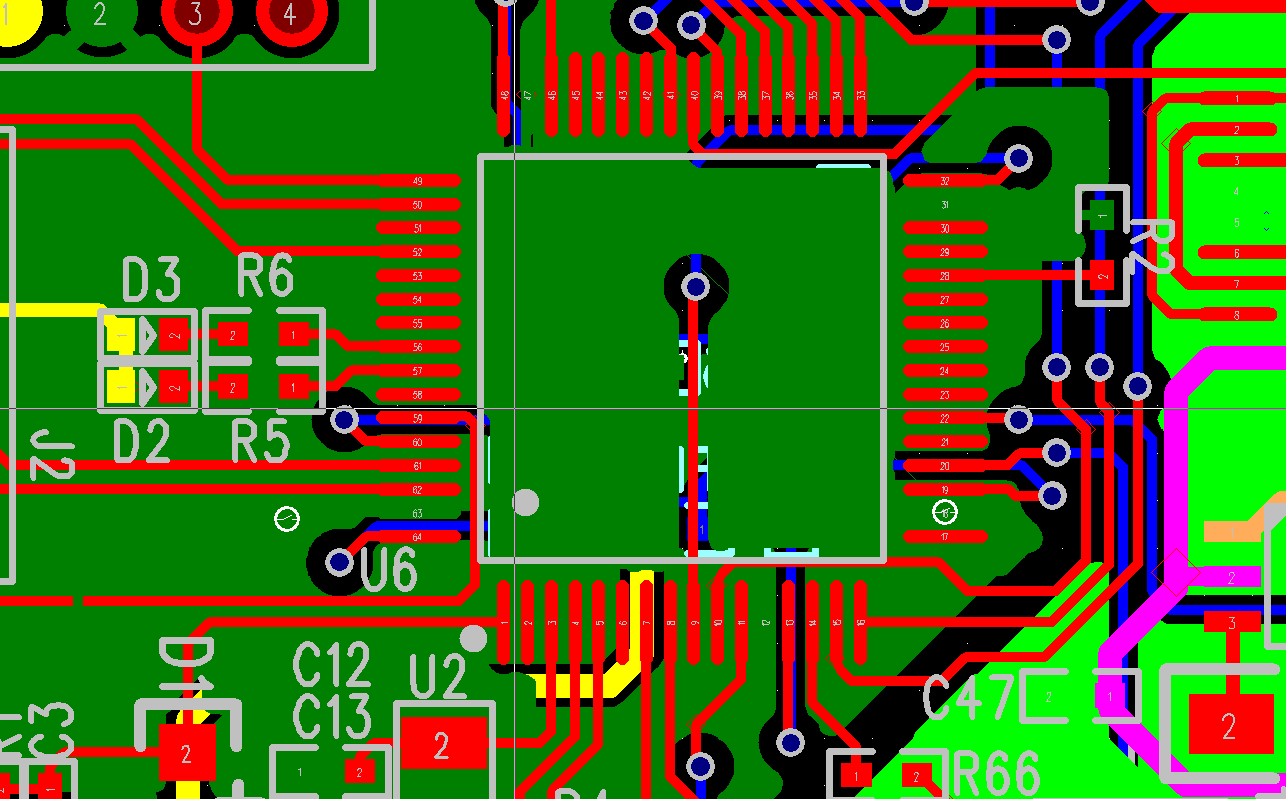
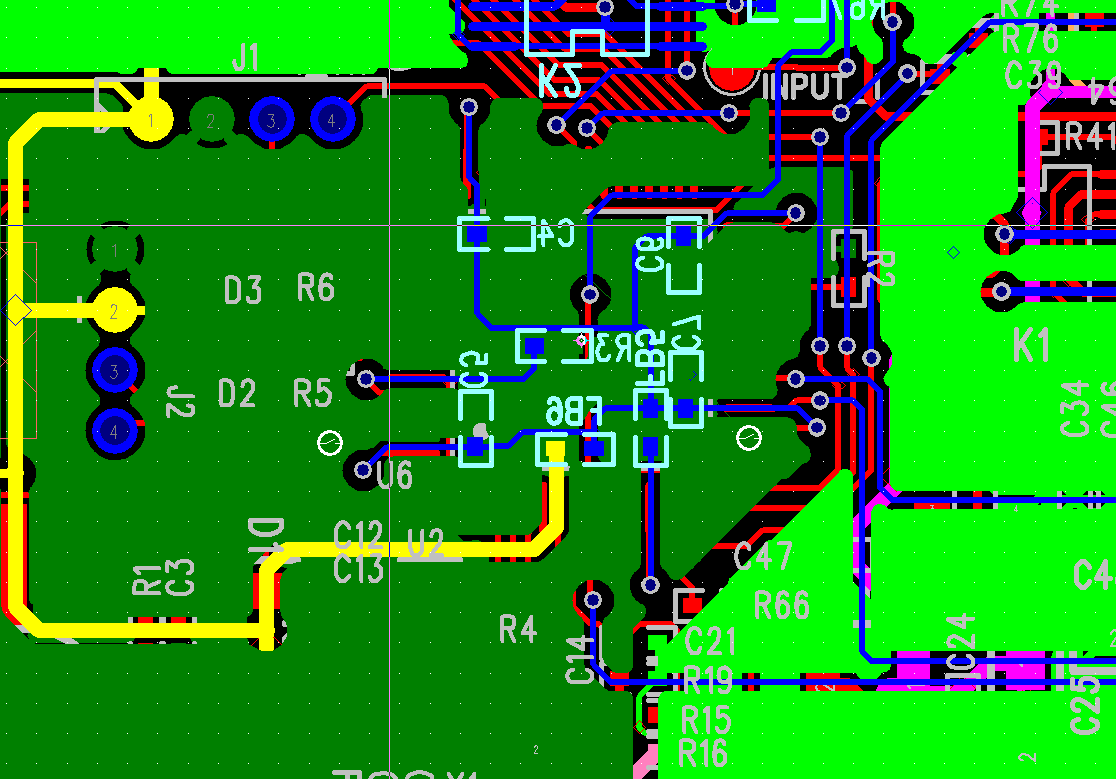
6.1

1. 今日工作日程：把CPU的去偶电容改为四个；修改电源线的宽度；修改布局的美观度；把一个LED换成硬件层面控制的类型
2. 铜皮之间未连接

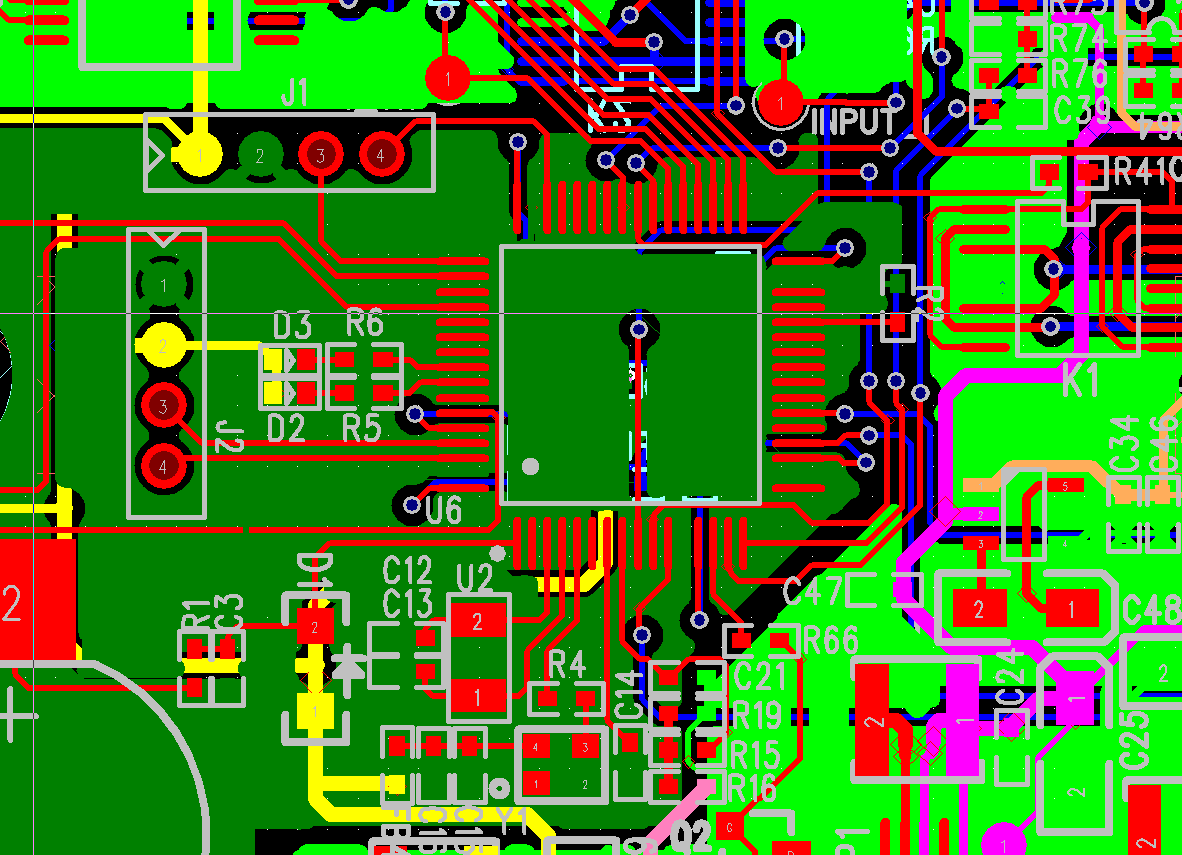


TOP

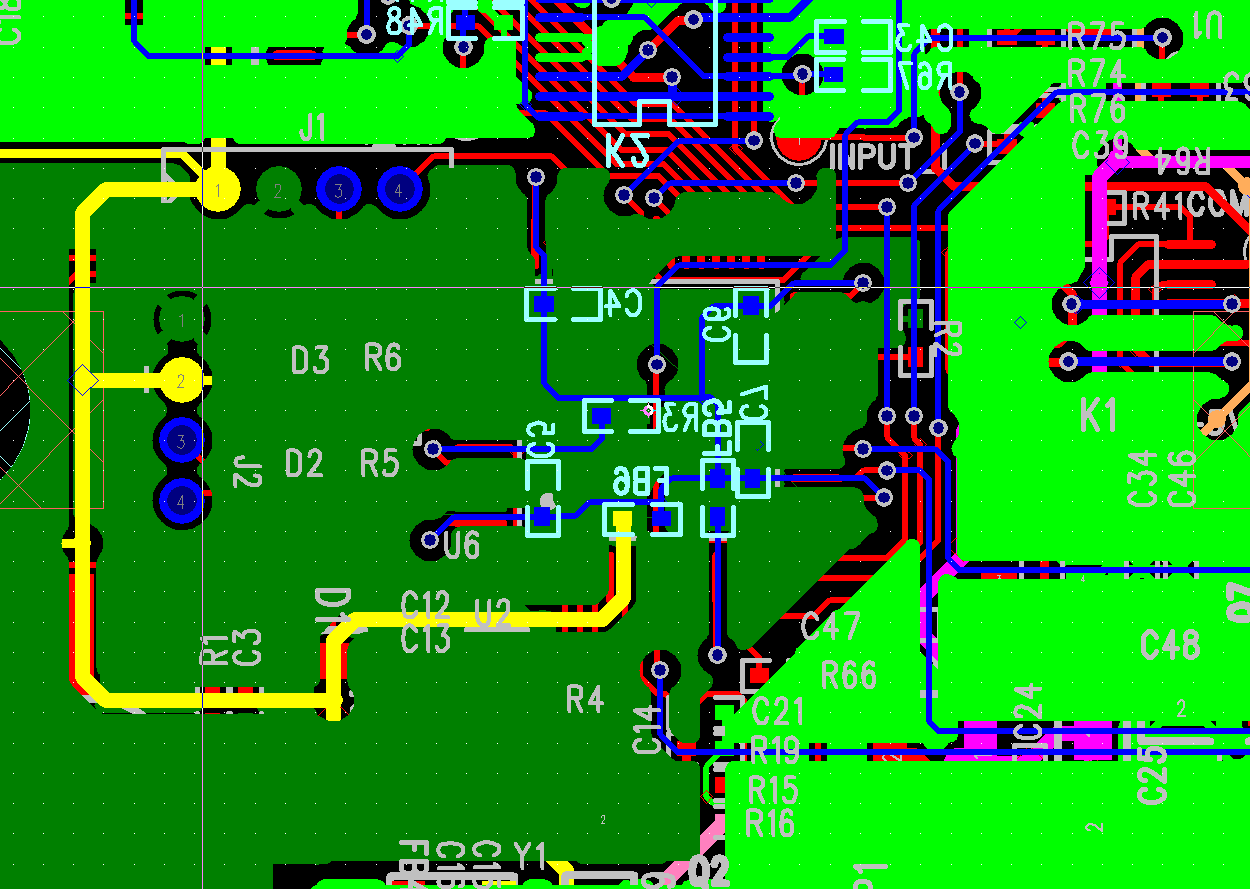


BOTTOM

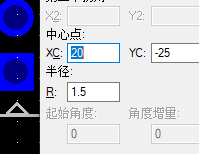
增加过孔后



TOP



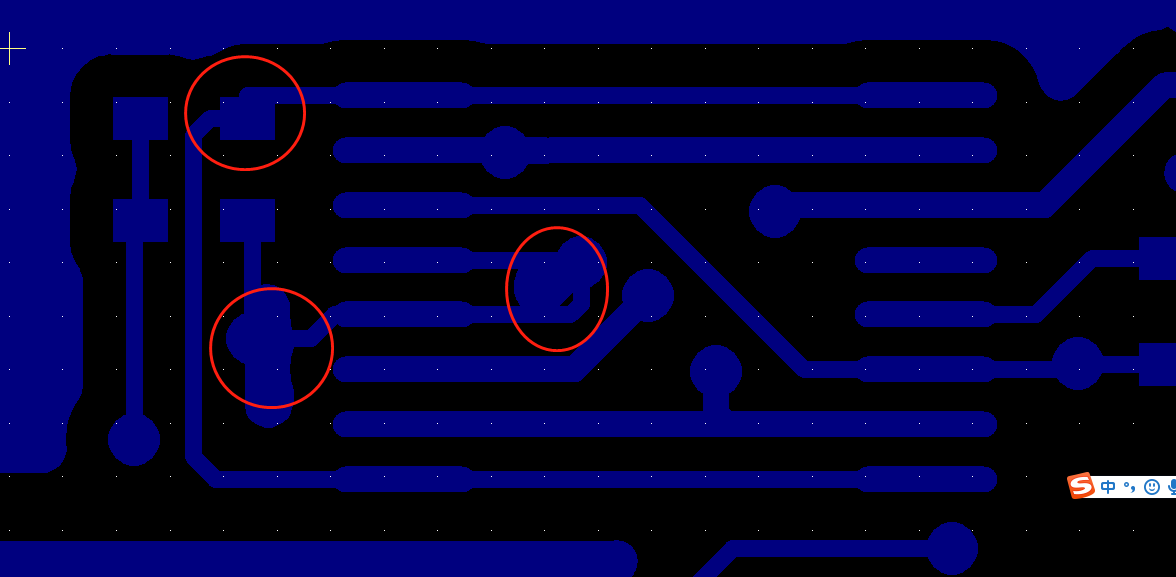
结论：虽然明面上看不出来有什么区别。但是顶层和地层之间的连接性断了

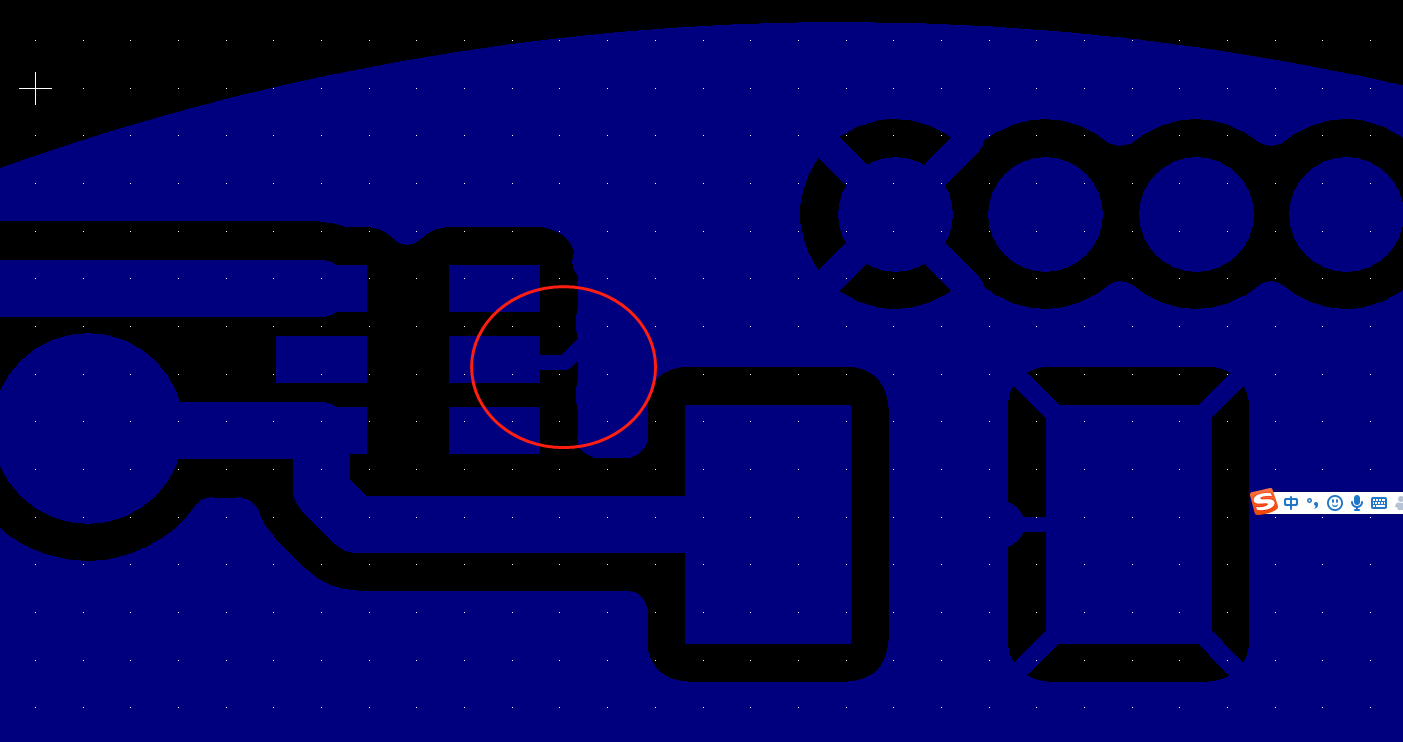
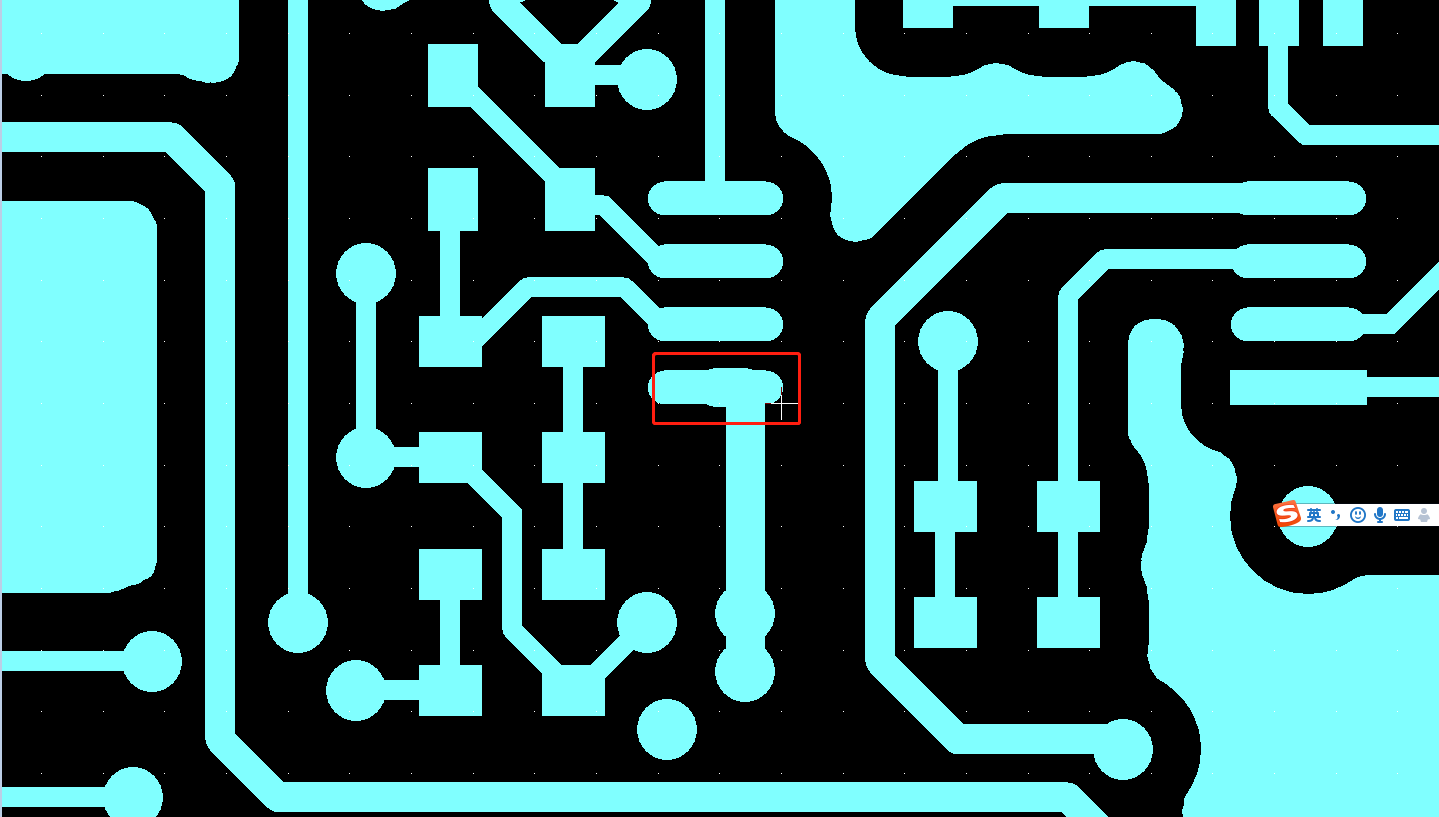


1. 改两个地方，第一个是改原理图电流测试部分，第二个是贴片输出改为通孔输出，第三个是将电源线有通孔的增加几个通孔
2. 能走电源层就走电源层，走线最短，地层不要绕的太大，不要把地层切的支离破碎
3. 板子看不出来电流走向，把原理图再捋一捋

6.2

1. 地层布线层的问题：线不规整，带有过孔的线会出现糊成一团的铜皮



1. 
2. 
3. 看PCB布线规则手册，然后就可以打样了
4. 共模电感怎么选取——怎么根据频率去选取阻抗，共模电感中增加阻抗的目的是什么，为什么阻抗成为了选取的一个重要参数？而不是说亨利是重要的参数呢
5. 什么是弱上拉电阻
6. 识别的原理是不是就是比较器

6.3

1. 电阻阻值的设定规则
2. 为什么一直在提示未移除回路
3. PCB已经在打样的路上了，下一步计划是完全熟悉原理图中的每个参数，并将他们串成一张网。
4. 计划初步定下：具体细则详细展开——
5. 手机通讯基本原理
6. BOM表

6.4

1. 今日总结：在画完原理图和PCB后要去核对封装对不对，元器件的封装是不是统一。要把BOM表导出来核对
2. 焊接材料的特性，松香是助焊剂，焊接紧密引脚的元器件怎么焊
3. 不同大小的电容的封装是怎么样的 比如33uF/16V的电容和其不同的封装大小的电容之间的关系
4. 下载电容和电阻的管脚手册

6.5

1. 熟悉元器件以及封装的命名方式，了解所有的元器件以及外形和封装
2. UCOSII是在C++器上面编译的

6.8

**6.11**

|  |  |  |  |
| --- | --- | --- | --- |
| R1 | R2 | 参考电压(V) | 实际电压(V) |
| 49.9K | 9.09K | 5.19 | 5.185 |
|  |  |  |  |

6.12

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 坏（漏极） | | | 好（漏极） | | |
|  | 前 | 中 | 后 | 前 | 中 | 后 |
| Q24 | 6.7 | 0 | 6.7 | 6.7 | 0 | 6.7 |
| Q21 | 6.7 | 0.4-2.1 | 6.7 | 6.7 | 0 | 6.7 |
| Q20 | 6.7 | 0.4-4 | 6.7 | 6.7 | 0 | 6.7 |
| Q19 |  | 0.4-4 |  | 6.7 | 0 | 6.7 |
| Q16 |  | 0.4-4 |  |  | 0 |  |
| Q15 |  | 0 |  |  | 0 |  |
| Q14 |  | 不定 |  |  | 0 |  |
| Q13 |  | 不定 |  |  | 0 |  |
| Q11 |  | 不定 |  |  | 0 |  |
| Q10 |  | 不定 |  |  | 0 |  |
| Q9 |  | 6.7 |  |  | 0 |  |
| Q8 |  | 不定 |  |  | 0 |  |
| Q7 |  | 不定 |  |  | 0 |  |
| Q6 |  | .. |  |  | 0 |  |
| Q5 |  | .. |  |  | 0 |  |
| Q2 |  | .. |  |  | 0 |  |
| Q26 |  |  |  | 6.7 | 3.5 | 6.7 |
| Q17 |  |  |  |  | 0 |  |
| Q22 |  |  |  | 6.7 | 0 | 6.7 |
| Q25 |  | 6.7 |  | 6.7 | 3.5 | 6.7 |
| Q23 |  |  |  | 6.7 | 0 | 6.7 |

1. 有极性电容和无极性电容的应用场合有什么不同。
2. Alarm电源是6.6V不是6.7V。这个原理是什么，改怎么调整

6.15

6.17

6.18

1. IF 与 #IF的区别：前者是条件判断，后者是条件编译。前者可以跟变量后者不可以跟变量
2. ->是结构体中的指向符号，指针传递变量，后面的变量值传递给前者
3. >>是二进制右移符号，正补零，负补一
4. &是与 |是或
5. 布尔运算：真1 假0 或| 与& 异或^

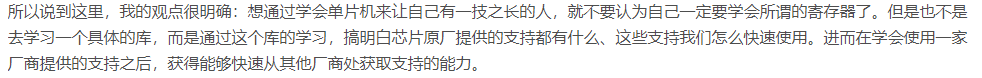
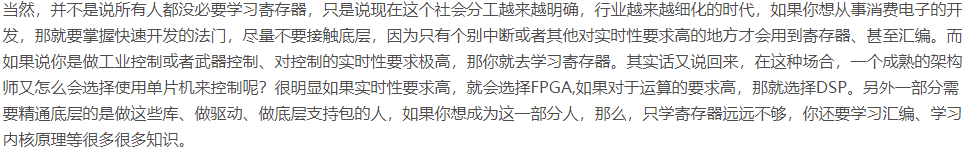
运算规则：或为同假为假 0|0=0 0|1=1

与为同真为真

相同为假

1. 数组怎么理解。它有多少位
2. CPU的寄存器都有哪种类型，哪些是必要的
3. 堆栈的方向和存储器的地址有没有必然的关系呢
4. 保存当前的CPU中的处理器值后，下一个阶段返回来读取回数据后他还需要重新运行这个任务吗，还是说从离开的位置开始运行

6.19

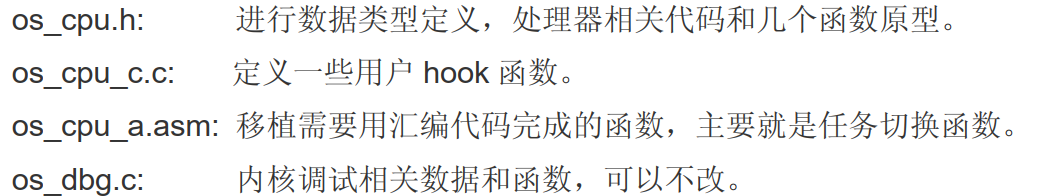
1. 中断服务子程序包含的部分：关中断，保护现场，中断服务，开中断，恢复现场，中断返回
2. USB的驱动和OLED的驱动程序
3. UCOSII包含哪些内容，那些不同的文件夹下面的文件都是用来做什么的。他们之间有什么关联？
4. 0L是把0定义成长整型的意思
5. 0X0E07中0X的含义就是代表使用的是十六进制的意思
6. 什么是双向列表
7. 在FOR循环里面写成for (;;)这样的意思是死循环，不跳出循环的意思。
8. C语言中的Static关键字的用法，用于变量和用于函数。用于函数则是说明这个函数只能在该文件中被调用，其他文件中即使是同名的函数也是无法调用的。而用于变量则是成为静态变量，在该程序中只能被执行一次，直到程序执行完毕。用于全局变量，则只能被该文件中使用，不能被文件使用。
9. 头文件和C的源文件都是怎么建立的
10. 先用多任务系统建立几个简单的任务
11. 把框架写出来
12. Traceanlyzer这个是软件还是什么。针对STM32开发的？？
13. 
14. 

**6.22**

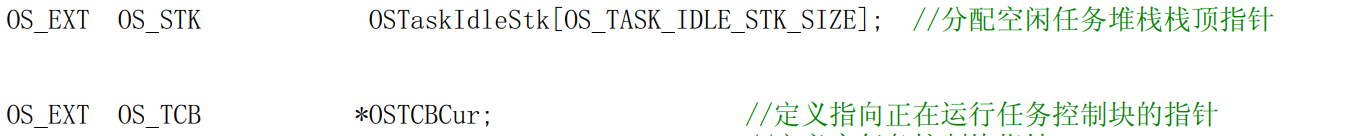
1. 软件编译的时候出现错误，错误代码如下

.\STM32F103\_ucos ii.sct(7): error: L6236E: No section matches selector - **no section to be FIRST/LAST.**

错误原因是没有加入启动文件，从库函数中找到对应芯片信号的启动文件加到工程中，这个错误就没事情了

1. 还有一个问题就是，建立工程后，把代码文件明明都放在那些文件夹中了，可是为什么打开工程所在的文件夹中堆满了我自己代码文件。
2. 错误代码为L6218E，别人给出的解决方案
3. **STM32工程建立必备的几个文件**：启动文件 系统文件（.c和.h）还需要设置配置
4. M3核只使用thumb2指令集
5. M3核需要关注的地方：寄存器，操作模式和特权级别，中断
6. 任务调度，任务同步，内存管理这三个块是UCOSII系统的主要部分
7. 一个定时器的三个部分：定时时间，回调函数和属性（哪种定时方式，是一次还是周期的）
8. 移植需要的四个文件
9. 三种编程方式一种是寄存器 一种是库 一种是HAL库三者有什么区别。
10. 定时器和计数器的差别在哪里
11. Extern
12. C语言编译器有哪些：GCC是编译器
13. 编译原理
14. 整理库存

6.23

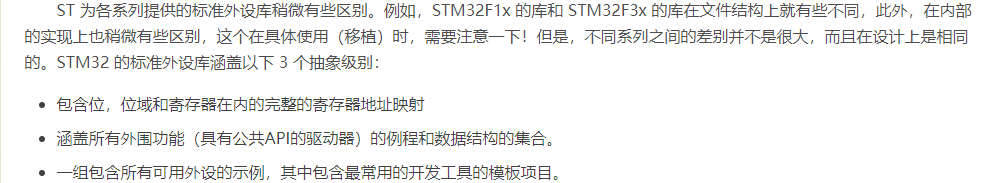
1. C语言的关键字有32+11个
2. 关键字不等于函数，比如printf是一个函数，虽然它在C语言中很常见，但是他并不是关键字
3. 变量 整数(int short long) 实数也就是浮点类型(float, double, long double) 字符型ASC II （一个字节，八位）
4. 整型常量的类型由常量本事数值决定，32700为int型，33000为long型
5. 常量无unsigned型（整数）
6. OS\_EXT的含义是什么
7. 
8. 硬件底层知识,包括平台移植,驱动开发,以及android APP开发
9. **存储器和总线架构、电源控制、备份寄存器、复位和时钟控制，通用和复用功能I/O，中断和时间等等前几章一定要花时间阅读**
10. **阅读《STM32固件库使用手册》的时候，前面几章也是必须阅读的。比如第一章文档和库规范中的命名规则，编码规则，这些都是需要注意的。第二章是最关键的，希望大家熟读。第二章描述了固件库的架构，我们如何去使用固件库的步骤等。有了第二章的基础，我们就可以借助固件库写出自己的代码了.**
11. 建议对GPIO库函数、中断部分库函数、复位和时钟设置的库函数要比较熟悉
12. GPIO本身的编程实际上很简单：

1、设置GPIO口的引脚为输入或者输出模式。我们在进行点灯代码的时候，一般设置为推挽输出模式。

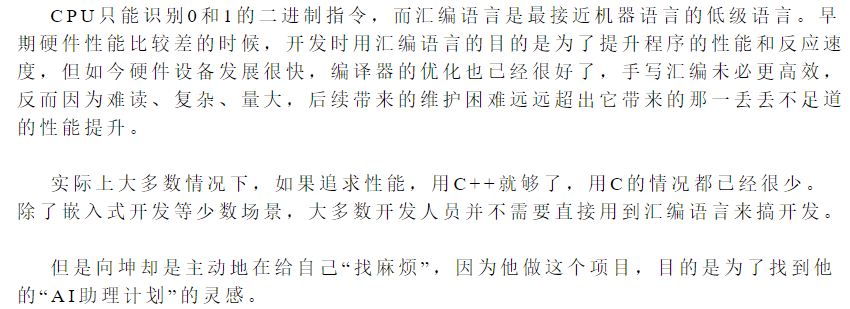
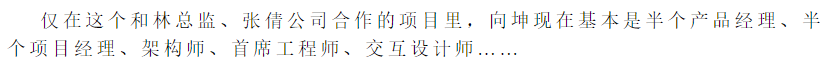
2、操作寄存器，往寄存器里置1或者清零操作——这个步骤，固件库已经提供了专门的GPIO\_SetBits函数和GPIO\_ResetBits函数，我们只要去调用即可实现对IO口的置1和清零。

3、实现多种花样的LED闪动，使得自己熟悉GPIO的编程过程。

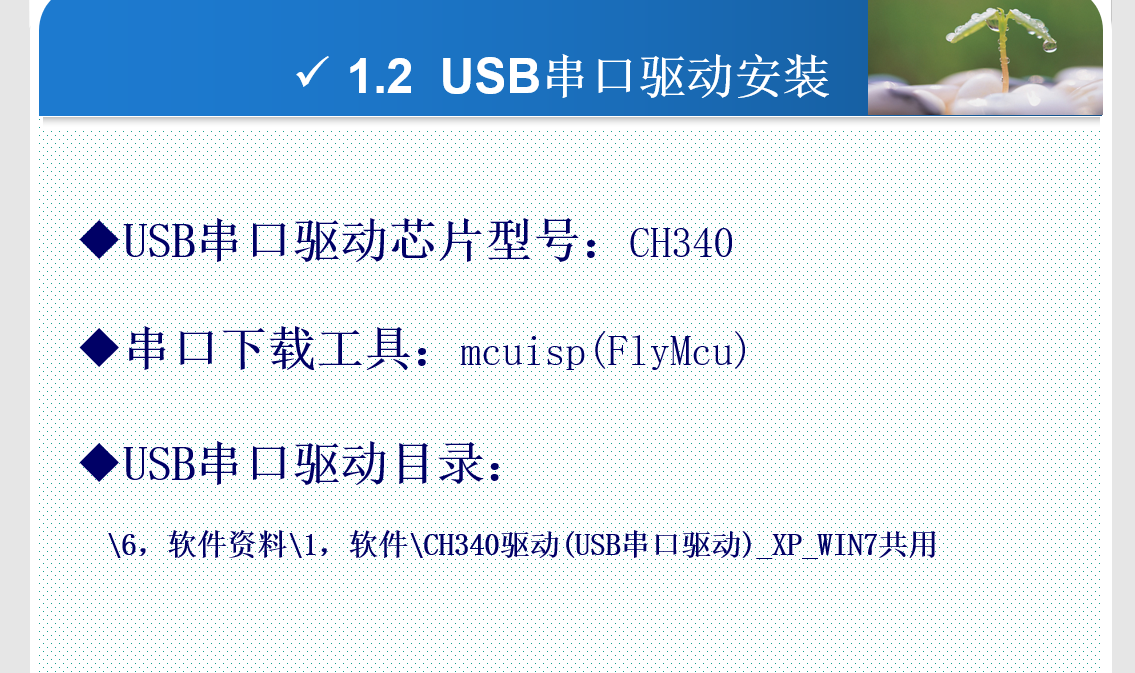
该步骤要达到的目标：熟悉调试软件，如烧写HEX出现问题，可简单判别问题所在，并独立解决。

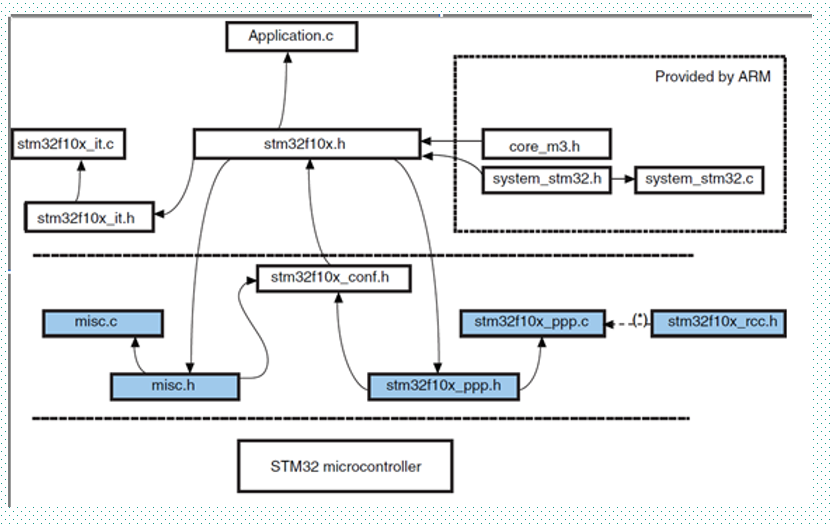
1. 将要接触到串口编程、TFT液晶屏驱动编程、定时器编程、串行外设接口SPI编程、存储器编程、SD卡与文件系统移植、USB读写、UCOS移植等，有精力还可以研究其他外设。
2. 学习嵌入式重要的是整体的工程思想和项目经验积累
3. 无非就是对内资源（寄存器、中断、定时器、串口等等），和对各种外设的操作而已，挂载的外设越多，系统会越复杂，这时可能会用上操作系统，同时，如果工程中涉及到各种算法、自动控制、采样、滤波等实际问题的时候，不仅会上操作系统，而且还对数学、物理、电路等方面知识要有较强的功底，对编程的能力的要求会更高，此时就要求对STM32的各种资源有比较明晰的了解了。
4. 

6.24

1. 伪代码，伪代码的产生是因为在不同的编程语言移植比较麻烦，而伪代码直接写清楚了代码的逻辑，在不同的编程语言上直接替换对应的关键字就好了
2. FACS（facial action coding system）面部动作编码系统
3. 
4. 

6.28





关键词：GPIO

预处理——怎么看预处理后的结果。