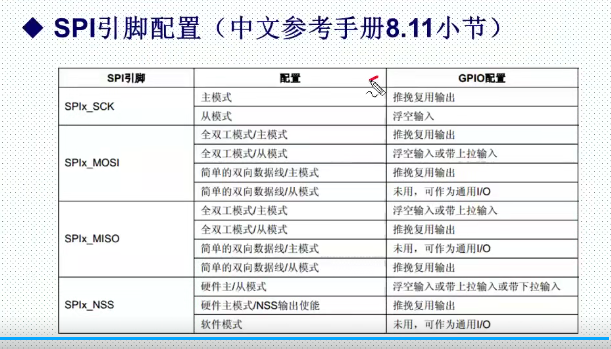
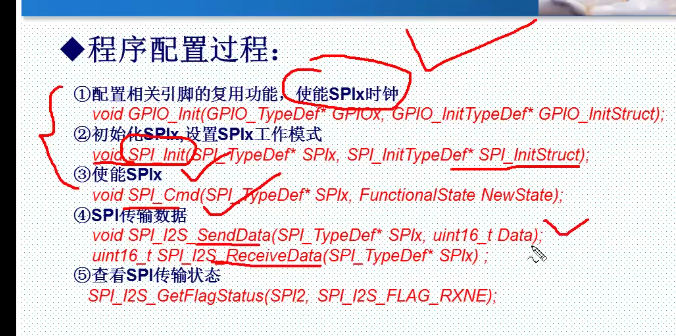
8.3

1. Bootloader升级程序，长文件，短文件
2. 漏电流阈值500uA，其他的等标定的时候再继续
3. **抓包工具**
4. 测试SPI通讯
5. 设计实验——SPI+FLASH+OLED
6. 示波器怎么使用，怎么看波形

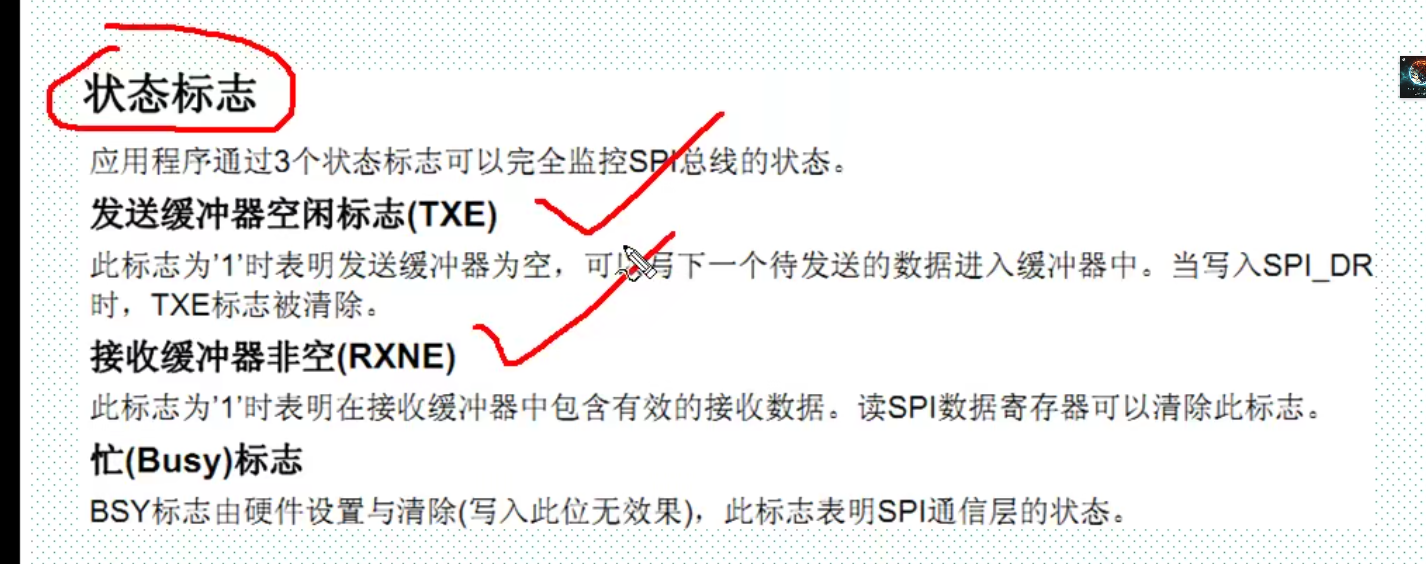
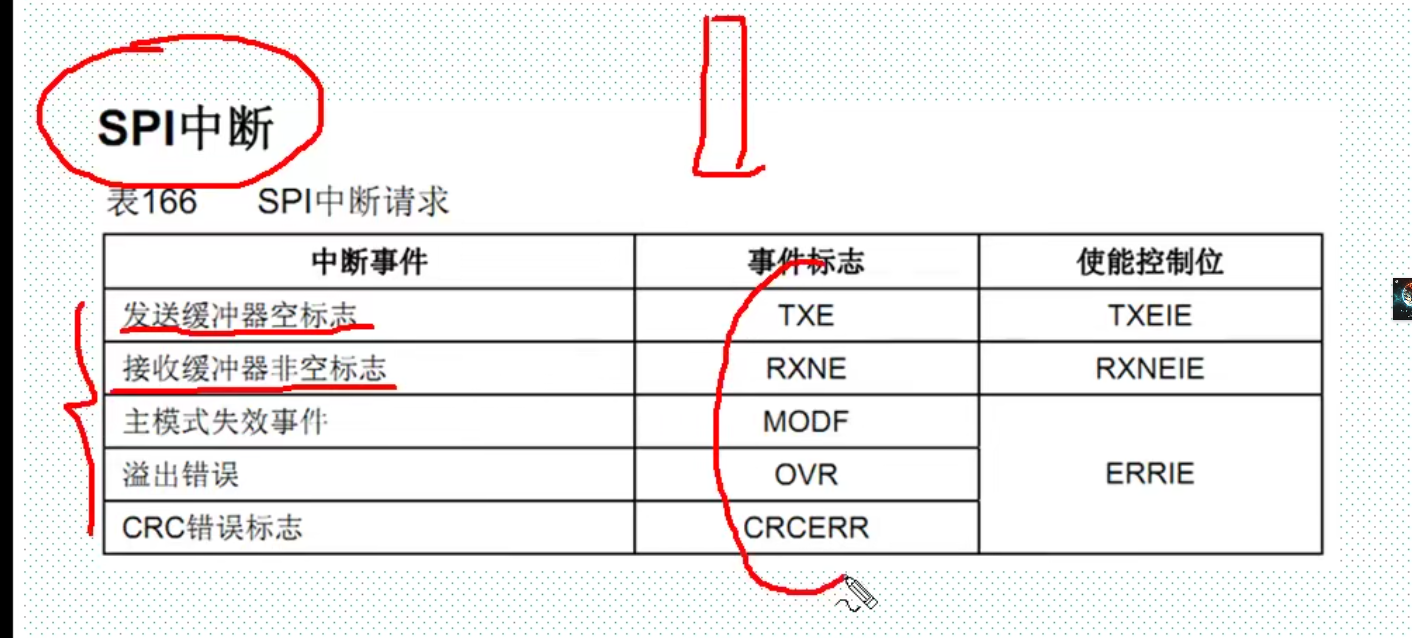
8.4

1. 

SPI的引脚配置



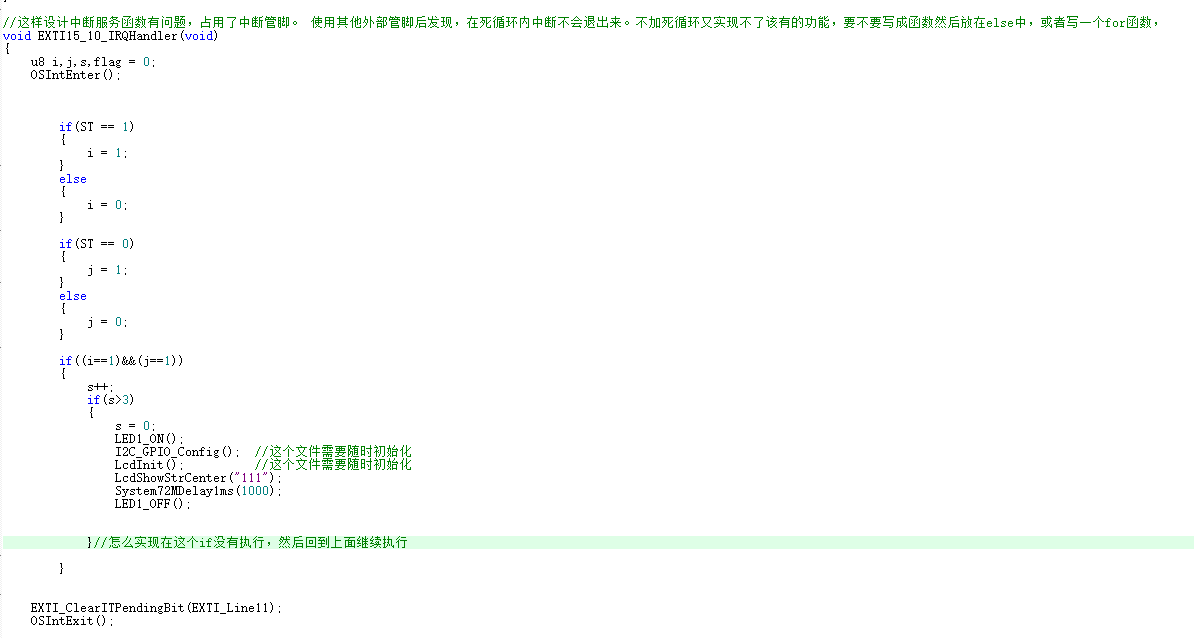
循环发送试试

1. CH374上的24M的晶振是在哪个部分起作用的
2. SPI，如果想从从机中只读出一个字节而不写入，就向从机发送一个空字节，比如0x00H,0Xffh
3. 
4. 
5. 
6. 今天解决了SPI和CH374无法通讯的问题，问题出在晶振电路出错

8.5

1. 自己查，自己测信号
2. Ucosii的中断函数怎么写。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 外部输入高(1) | 外部输入低(0) |  |  |
| 配置上拉输入 | 1 | 0 |  |  |
| 配置下拉输入 | 1 | 0 |  |  |
| 配置浮空输入 | 1 | 0 |  |  |

1. 
2. 目前的问题：上电以后INT管脚为3.3V，但是刷完程序以后这个管脚直接就变低，并不是插上USB以后才变低
3. **INT这个中断源是怎么产生的**
4. 中断输出引脚是可选连接，低电平有效，可以通过查询寄存器中的中断标志位代替
5. 为什么一上电INT管脚就变低，之后就不再变高
6. 硬件复位是什么过程
7. 问题：

描述：烧写好程序后，INT管脚直接变低

重启单片机，INT管脚为高，插入USB没有变化

硬件复位复位哪些

尝试查询方式去读取寄存器，查询USB状态。

1. 明天在另一块板子上面焊USB芯片

8.6

1. 十通道程序问题描述：会不断枚举

调试过程结论：在进入设备模式初始化，在启动USB那一步后电平确实会变低。但是会跳转到终端服务函数中，然后电平会恢复

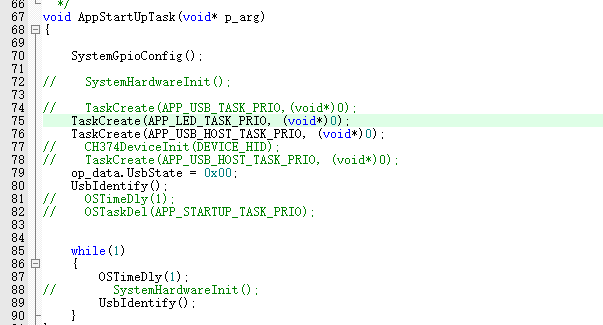
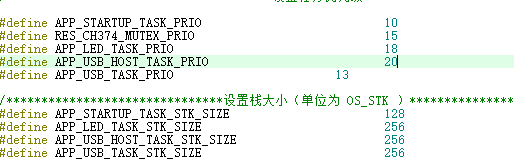
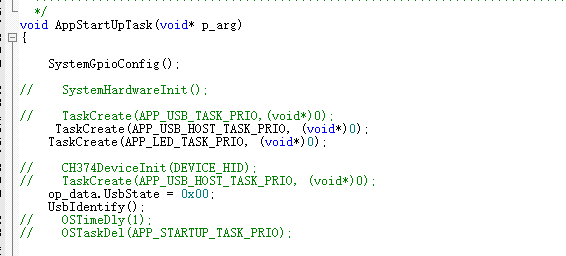
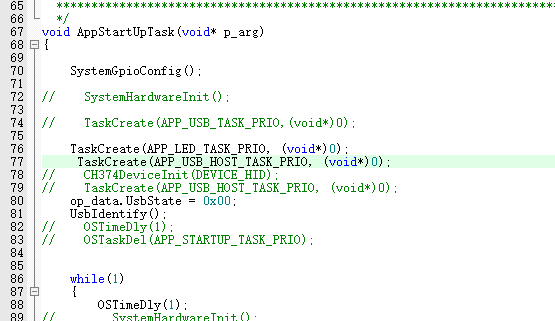
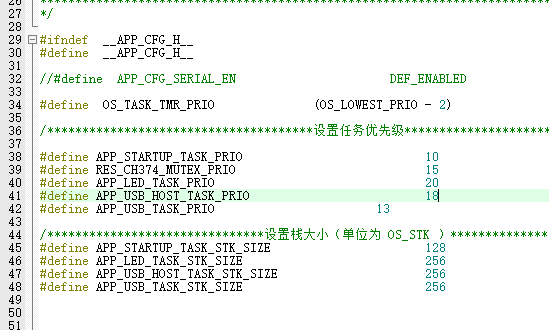
十通道程序中会不断检测电平状态，从而选择是初始化为从机模式还是主机模式，检测到电平变化后会初始化，反之在插入USB线后是不会初始化的。

1. 设备模式都有哪些描述符，主机发送的命令，从机怎么动作，还是跳入到中断中么
2. 问题：下载到单片机的代码每次重新上电以后从哪里开始执行
3. 想知道，下载到单片机里的代码的启动任务会不会执行。宏定义预编译就被写在了固定的地址上，那么上次断电前的的数据重新上电以后会不会会还是上一次断电前的数据。
4. **任务的运行顺序问题，优先级怎么放才比较合适**

起始任务用while好还是把任务挂起好

1. **有新的想法，把自己验证过的实验，都统计起来，写成一个文档，有时候当时记住了，过几天就又忘记实验的结论了**

**归好类**

1. **经过试验验证ucosii的起始任务在重新上电以后不运行while循环以外的内容，只运行while循环以内的内容**
2.  
3. 图片中这种状态下两个任务均可以正常运行
4. 
5. 这样状态下，优先级没改，还可以正常运行
6. 
7. 
8. 但是在这种状态下，就只能有USB\_HOST任务进行
9. TaskCreate(APP\_LED\_TASK\_PRIO, (void\*)0);

TaskCreate(APP\_USB\_HOST\_TASK\_PRIO, (void\*)0);

#define APP\_LED\_TASK\_PRIO 20

#define APP\_USB\_HOST\_TASK\_PRIO 18

**这样就不能正常运行LED任务**

1. **思考**：在十通道的程序中，如果当前是设备模式（作为从机），然后关机，重启以后作为从机还能连上电脑么
2. 今天十通道检测仪器在只有USB供电的情况下去调试代码，发现USB不断复位，找不到原因。后来佳哥指出是没有接上电源，只有USB供电的情况下，晶振电流增加，晶振频率增加，单片机工作不正常，不断复位。
3. 电流用示波器怎么看

8.7

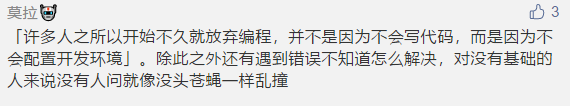
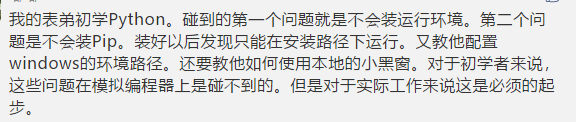
1. 实现昨天的想法。
2. 测试只在USB供电的情况下，晶振的频率，以及单片机的工作状态
3. 今天做adc测试
4. 测一测十通道的VCC\_SCAN端的电压大小
5. OLED初始化放在什么位置比较好，能够正常显示，放在任务中的while循环外面好像也不行，貌似每次任务跳转过来后就直接进入while循环里面了。测试一下
6. 建立多个任务，用调试模式去调试，判断任务执行的先后顺序（**是否和优先级有关系**）
7. 单片机断电再上电以后程序从哪里开始执行，断电前的位置。如果想从头开始执行就需要复位。**复位怎么设置**
8. 下班再研究一下十通道那个废板子为什么供电以后还是不能工作
9. **任务截止日，两周后（8月21日）**，能够用，程序没问题，不用再修改，硬件飞线不用管，后期再改
10. 任务待完成项：**bootloader升级，AD模块（漏电流，短路），校准**
11. **写一个详细的框架**
12. **提前想的清楚一点**，把各种可能性都想到
13. **写自己在实操中学到的东西，输出出来**
14. 延时函数，delay中的，OS中的
15. 在程序设计中时间的概念，毫秒，分解任务
16. 把IO口配置放在一起好还是分成模块放在不同的驱动单元内。分成模块是方便移植，写在一起是为了方便修改，不用去多个文件内去修改
17. **程序端口起名字要起好，不能奇奇怪怪的**
18. 文件设立的还是太杂，要自己定一个规则，在项目建立的时候建立几个文件夹，分别是什么文件夹，都有什么功能（比如说要是会的功能就怎么样，不会的功能，需要重新学习的功能就需要自己建立例程文件，跑一跑例程，详细分类）
19. **工作的时候把任何会干扰到工作状态的程序关掉（自己容易被打断的几类信息《新闻，百度侧面的新闻，微信公众号的文章，各种吸引人的标题》）坚决不看浪费时间没营养的消息、文章**

8.10

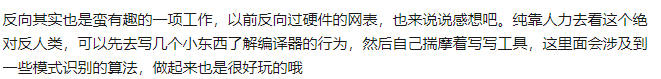
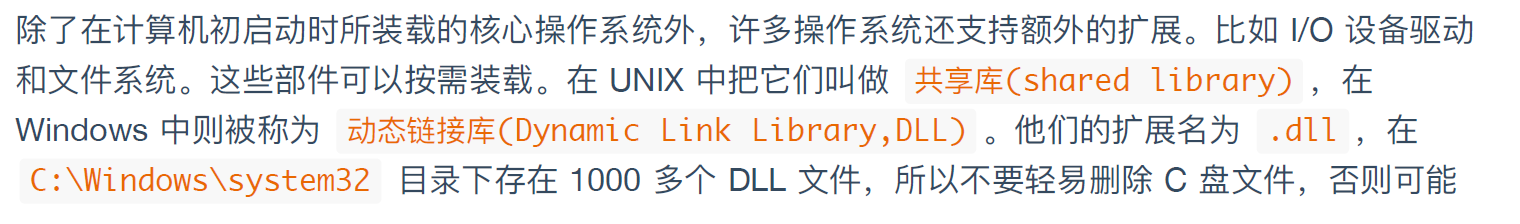
1. 看放大器的芯片手册，确定放大倍数是多大
2. 解决之前的K1SPDT无法正常转换电压的问题，更换1613芯片，可以正常转换，但是又发现MOS管作为开关控制的5V无法正常切换，于是换了NMOS和PMOS发现NMOS能够正常拉低，但是PMOS的G,S均为同一电平，这就很不正常，换了10R的电阻，恢复
3. AD在ucosii中采样的时候，为什么要加申请资源函数和释放资源函数
4. 互斥信号量也是需要占用优先级的，他是通过临时赋予需要访问资源的任务的优先级，把优先级高于所有任务的前面，这样他就能够访问资源。
5. 共享资源，但是共享的是什么资源，怎么去理解共享资源
6. **做事情不能走一步看一步，答应别人的事情要提前完成。不能拖拖拉拉，推脱。要提前把自己要做的事情规划好，详细到每一个细节都要完善到位。一颗雄壮的大树是需要茂密的枝叶和粗壮的根系。每一件事情也一样，需要详细的细节去丰富它。**
7. ADC采集中，使用UCOSII去调度任务，不同的优先级的任务在访问同一个资源的时候是需要的分配的

8.11

8.12

1. 今日目标，改写函数——ADC模块+声音模块
2. 上位机的事情先不考虑。先把其他的做完，校准放在下周
3. 先写框架——**详细的框架**写出来
4. 复位问题——是不是硬件的原因，在上电的时候，理应是从头开始执行
5. 定时器——配置以及作用，用来做什么
6. 若想使用文件系统是不是必须**先将实时时钟加载进去**
7. 学习四重奏：知识输入，知识处理，知识储存，知识输出
8. 
9. **汇编语言**
10. 
11. 

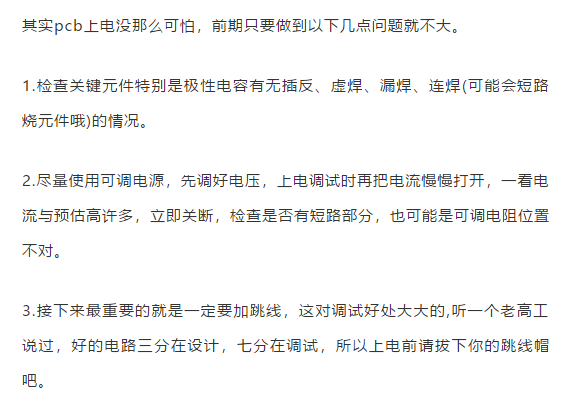
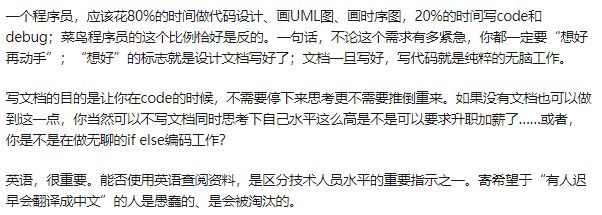
8.13

1. 
2. 
3. 
4. 
5. 
6. **访问指令所需要的的时间**
7. 

8.14

1. 怎么去理解串口，串口不是用UART通信么
2. 今日任务：先把AD写出来，先不去调试。然后再把SOUND写出来，不去调试。就这两个基本任务。
3. 使用串口和USB口有什么区别，他们该怎么通信。我硬件是USB口，而给我的用到的是UART。
4. 把每一个源文件对应的函数都写出来
5. **把工程可视化，系统的框架到源文件中函数的框架都要写出来还有程序的运行流程**
6. 电池电量需要校准不，**怎么去提醒电池没电**。
7. **DAC去改动输出电压**
8. 去追溯源头，程序用在哪里，怎么用的。
9. **发一个书单过去，如果要建立书屋的话。**
10. **先建立框架再写函数**
11. 校准的话，0点固定为了好选择截距，6.7V固定，再选一个5V，正好三个点。校准的话其实是为了测得更准，数学里面的校准，放在程序里面去实现。
12. 应用程序为什么要分成两个去下载进去
13. 学习如何使用QT做上位机

8.17

1. 移植控制器的ADC程序，尤其是校准的程序
2. USB和上位机通讯，写好程序
3. 声音模块
4. 循环显示电流和电压
5. 使用DAC去修改输出电压
6. 有一个问题，索引和定位怎么写。或者说建立好一个数据库以后怎么去匹配数据，这个就是索引的作用
7. 专注
8. 
9. 

8.18

1. 在6.7V输出电压这块，R1和R2两个电阻参数配比是49.9和6.65，计算出来的电压应该是6.8V，但是实际输出的电压是6.7V，这里需要再研究一下。
2. 
3. 还有一个就是运放的放大倍数。实际上应该是0.79\*（R1/R2+1）
4. 
5. 有一个问题，电流校准的时候不同的量程该怎么校准

8.19

1. abs()是绝对值函数
2. 先把函数的框架写出来，然后再写函数的内容
3. **把开发文档写出来**，尤其是不同的电平对应的意义，不然去写程序的时候就GG了，啥都想不起来，还得回头去查资料。

8.20

1. **短路的时候电流太大怎么办**
2. 开发文档中还需要加一个管脚高低电平对应的函数
3. 发现一个问题，就是在5/6.7V切换前端5V使能那里，经常NMOS管就不起作用了，换一个NMOS，就可以了
4. 短路检测和漏电流检测同时夺取COM\_OUT的使用权
5. 今日任务：检查为什么给声音模块通电以后灯的闪烁有问题，疑似电流太小
6. 测量一下声音输出波形的周期，周期为300MS
7. 工作总结：在设计硬件的时候要标注好不同的电平代表的含义是什么，方便编程
8. 在调试测量总线电流的时候，选择重复测四次就会出现度数大幅度变化。而且采集到的数据变化幅度也大。百思不得其解。如果说重复测10次的话就可以解决

8.21

1. 自动关机怎么写（伪造一个假象，停止任何活动）
2. 注意哪些资源应用，在函数中加好优先级改变。
3. 今天任务，把ADC移植到ucosii中。并且写USB校准的程序。
4. 滚动显示怎么操作
5. OLED中使用两个变量去传递参数，然后分别显示。比如使用Temp1和Temp2，但是会出现问题。**问题就是Temp1中的数据会和Temp2的数据连在一起**。怎么连接的不清楚。猜测可能是sprintf的原因。解决方案就是使用一个变量，然后分别赋值。这样就不会出现这种情况。
6. **驱动就是驱动，建立一个中间函数的源文件。专门写驱动层上面应用层的函数封装。留好最后的API。**
7. 在测量COM\_OUT的AD值的时候，突然发现为0.检查了一下发现是由小金属屑掉进去了。使和地接在一起，从而使采集到的值为0
8. **0.9寸OLED显示十二个字符（英文，数字）**
9. 静态变量static不能乱加，在本源文件中使用的话没问题。一旦出现其它文件调用它，就会出问题。所以在**外部调用的时候不能使用static**。
10. 这里看不懂，这是要返回校准电压值么

8.24

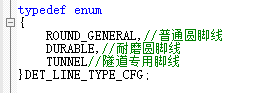
1. 今天的任务，就是写USB传输代码
2. 把所有分压电阻中的20K改成20.3K，DEMO就部分改部分不改了，其他的都要改
3. 测量电流用的运放OP262，连续运行了2天以后就出问题了。排除了单片机问题后怀疑是运放的问题
4. 在测试短路的时候发现短路的时候的电压到不了20mA，最多就是9mA。后来一直想不明白。佳哥指出是“满量程”了。就是说运放达到了最大输出。重新设计以后就行。
5. **芯片手册是一定要看的**，很多问题就出在不了解芯片。比如这次的运放会有最大输出。上次的开关切换电路没有接电源。

8.25

1. 有一个问题：在往ucosii中移植ADC代码，由于乱用互斥信号量而导致程序卡死。问题的解决方法是：明确互斥信号量的共享资源是谁，只在使用该资源的函数中使用互斥信号量

8.26

1. 睡眠模式设置是和定时器有关的，具体的使用例程见十通道检测仪中sleep.c源文件中
2. 从USB主机发送过来的数据主要经过解码和解包程序的处理，然后得到的数据就是主机发送过来的数据了。
3. 上位机中的ID是指什么，可不可以去掉
4. 和电线有关的参数在十通道检测仪里的程序里面，在system.h文件中

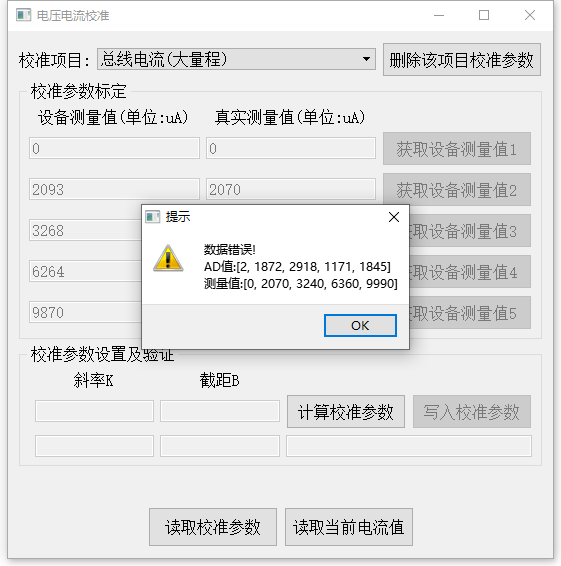


1. 重新建立一个USB传输工程，删减掉没有必要的东西。
2. 下传数据没问题，上传数据有问题，**不知道主机会不会给一个回应标志**。先写其他内容
3. **写程序一定要细**，即使是在其他的工程里面运转正常的程序，移植过来的时候都首先要测试一下，该函数是否工作正常。然后再使用这个函数去搭建封装更高的函数。**其次是**发现故障的时候就需要把每一个小的函数都去测试一下是否工作正常，不要宏观地，盲目地去找故障原因。

8.27

1. **每次都犯同样的错误，不初始化驱动，初始化配置函数很重要！很重要！**
2. MOS管容易坏么，NMOS
3. 问题：下位机给上位机传输数据，上位机收到数据以后小数点的位置有变化。例如下位机发送930，上位机显示9.3。考虑是上位机的问题。
4. Flash地址从0x08000000-0x0803FFFF
5. 从上位机发过来的数据是怎样打包的
6. 发送设置校准值的数据是怎样打包的 数据
7. 查一下PMOS管的参数

8.28

1. 调试的时候要去看传递的数据是什么，格式是什么，这样才叫调试。而不是只看表面上的程序调试的通不通。
2. 
3. **Flash读写操作**，把一些参数存在flash中。但是flash中存储的地址是需要去计算的。
4. 中断向量表的位置是不固定的，可以是在SRAM区，可以是FLASH区，也可以是内置BootLoader中
5. 1、 通过boot引脚设置可以将中断向量表定位于SRAM区，即起始地址为0x2000000，同时复位后PC指针位于0x2000000处；

2、 通过boot引脚设置可以将中断向量表定位于FLASH区，即起始地址为0x8000000，同时复位后PC指针位于0x8000000处；

3、 通过boot引脚设置可以将中断向量表定位于内置Bootloader区，本文不对这种情况做论述；

1. NVIC\_SetVectorTable(NVIC\_VectTab\_FLASH, 0x2000); //重映射中断向量表APP\_START\_ADDRESS
2. Hex文件的大小要注意。
3. 如果说用ucosii文件太大的话那就不用ucosii，使用无操作系统的程序。

8.31

1. 今日任务：修改BootLoader程序。并把USB通讯移植在BootLoader中。存储数据到flash的程序则等有时间再写。先把说明书写出来。
2. 函数指针
3. **\*(\_\_IO uint32\_t\*)APP\_START\_ADDRESS**——含义就是取APP\_START\_ADDRESS中的值，为什么要这么写呢而不是\*(\_\_IO uint32\_t)APP\_START\_ADDRESS呢。**答案就是**后者的写法是错误的，因为要想使用指针之前是需要先定义指针的，（\_\_IO uint32t\*）的作用就是**先定义**一个强制转换为\_\_IO unit32\_t类型的**指针**。最左边加一个\*的含义就是去取值。
4. **在bootloarder中还缺少USB数据传输程序，以及USB主机传输信号标志和USB从机接收到信号应答标志**
5. 还是需要自己去写一个USB驱动，剔除掉主机模式和从机MSC模式，只保留USB从机HID模式。
6. 验证一下USB枚举中，回调函数去掉能不能正常枚举
7. 先能够正常通讯，然后连接上位机，再去掉主机模式，最后去掉从机模式中的MSC模式。