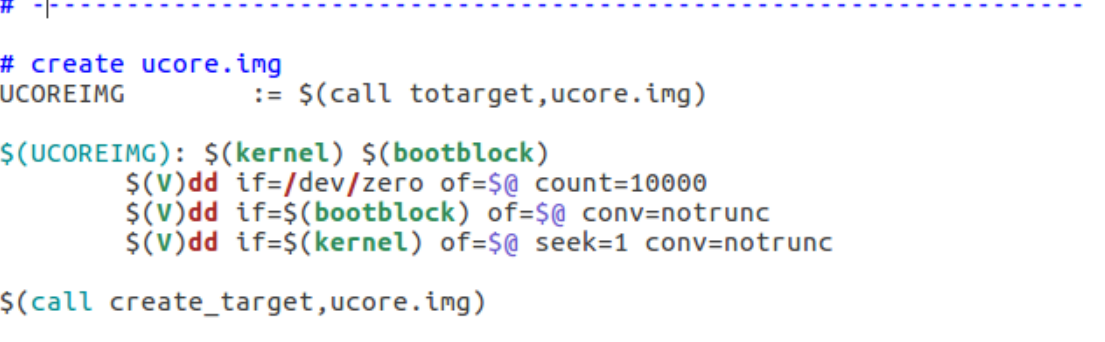
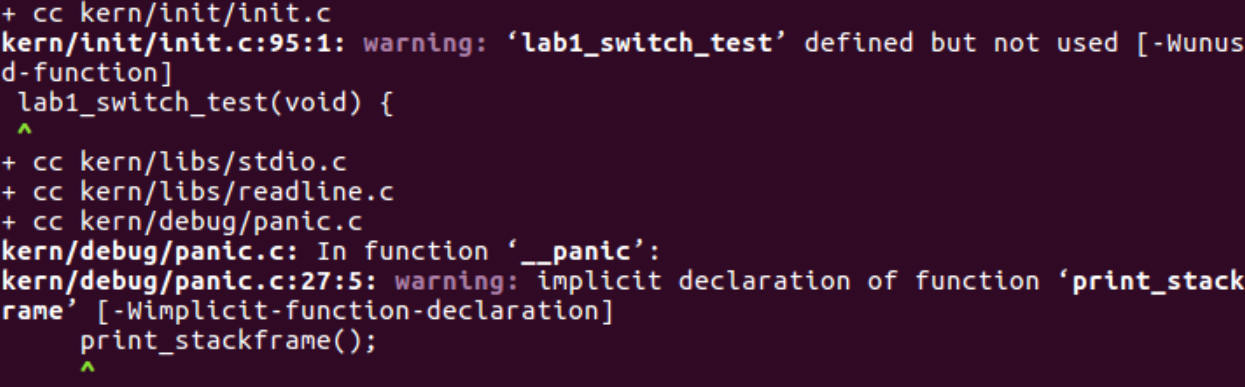
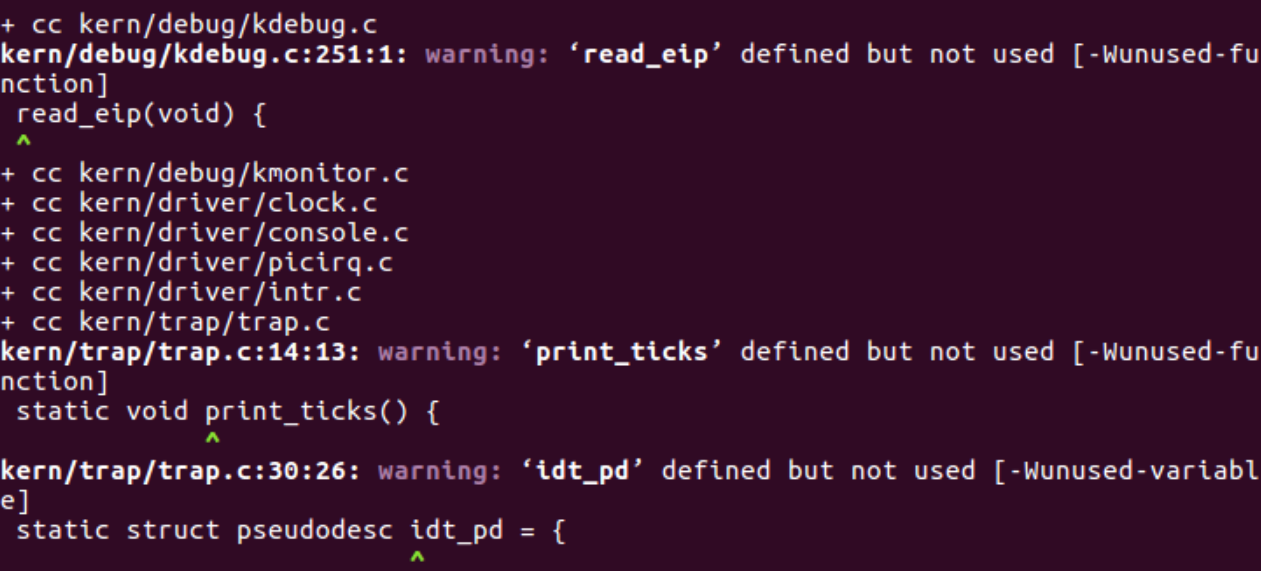
练习1：理解通过make生成执行文件的过程。（要求在报告中写 出对下述问题的回答）

操作系统镜像文件ucore.img是如何一步一步生成的



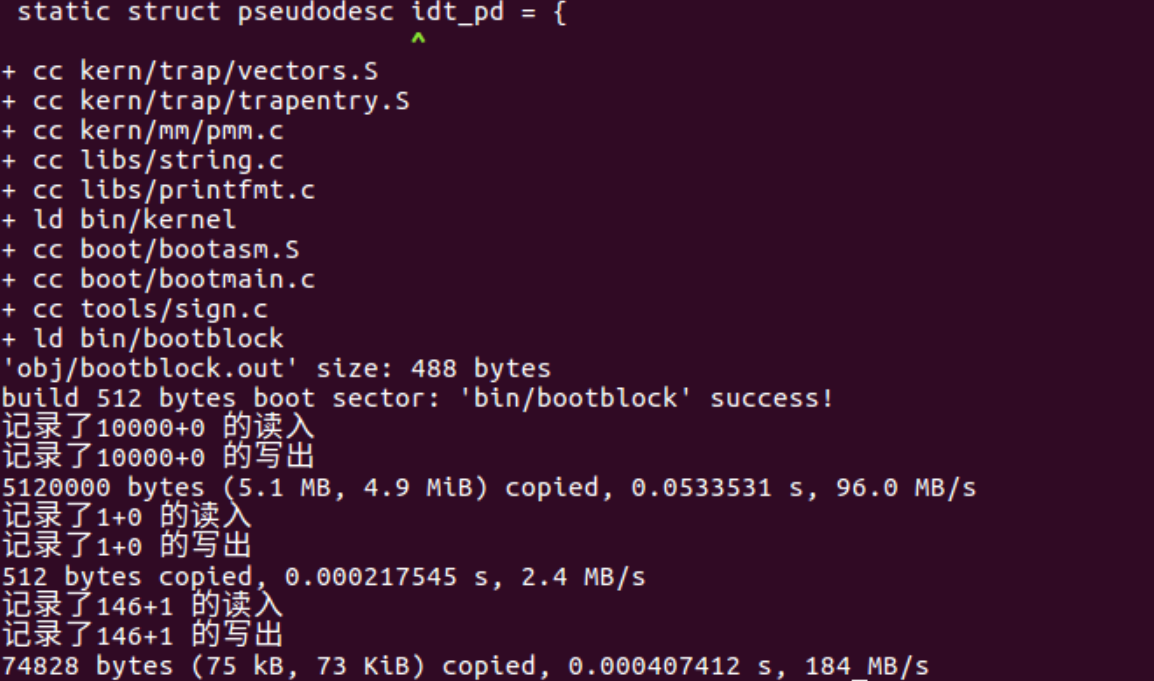
#构建bin/kernel





#构建sign工具与bin/bootblock

#构建ucore.img

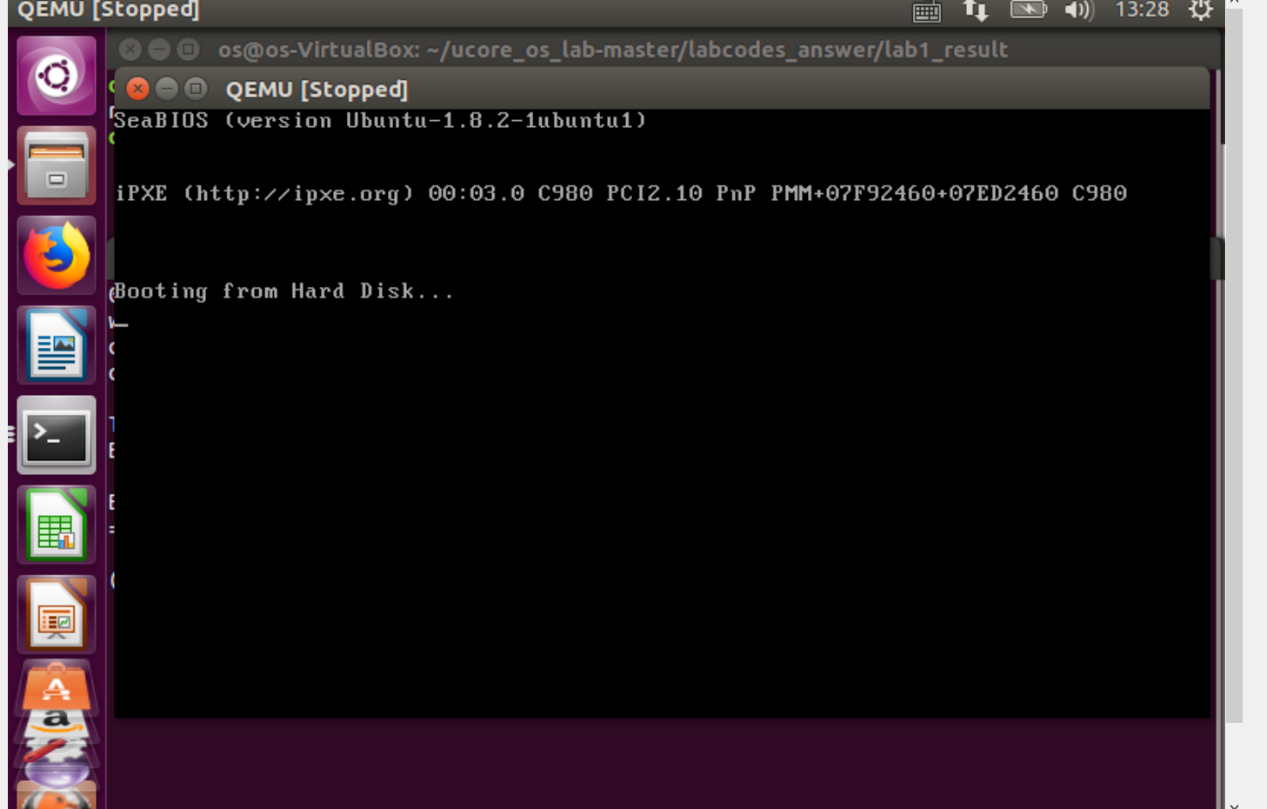


1. 通过编译内核文件，生成内核bin/kernel
2. 生成bin/bootblock的引导程序
3. 编译bootasm.s和bootmain.c，生成obj/bootblock.o
4. 编译sign.c生成sign.o工具

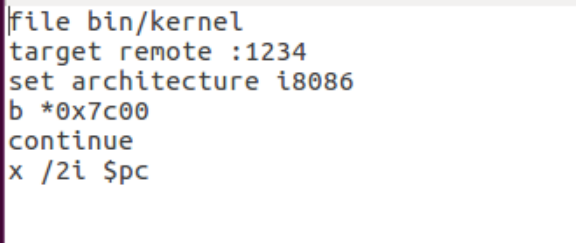
（3）使用sign.o工具规范化bootblock.o，生成bin/bootblock的引导程序

1. 生成ucore.img虚拟磁盘
2. dd初始化ucore.img为5120000byes
3. dd拷贝bin/bootblock到ucore.img的第一个扇区
4. dd拷贝bin/kernel到ucore.img的第二个扇区往后的空间

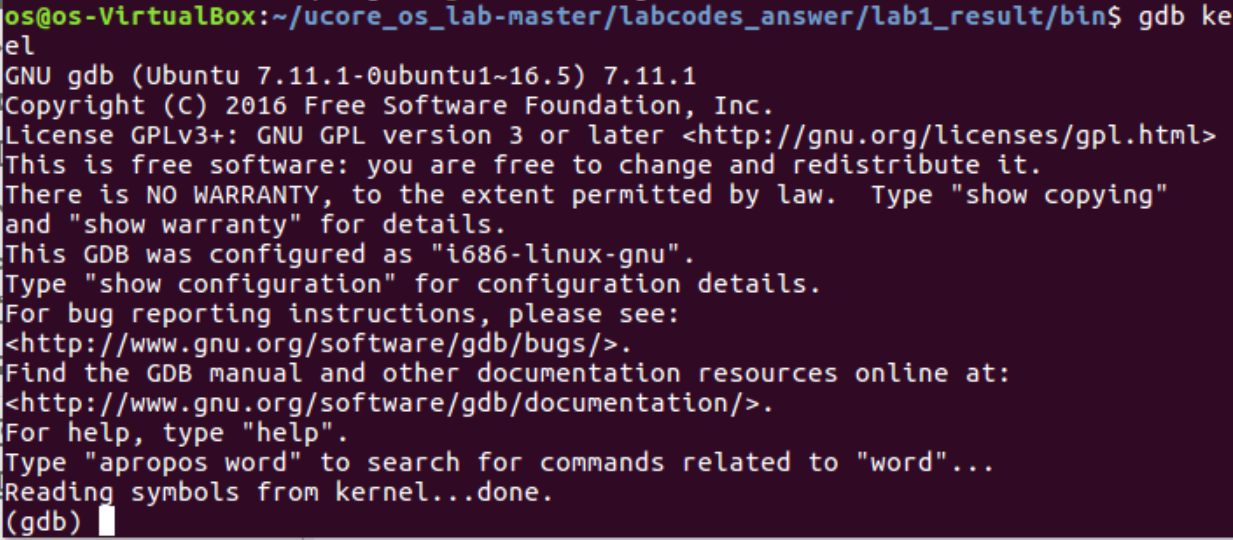
练习2：使用qemu执行并调试lab1中的软件。（要求在报告中简 要写出练习过程）

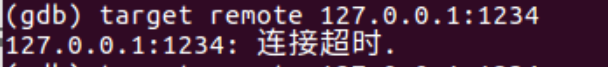


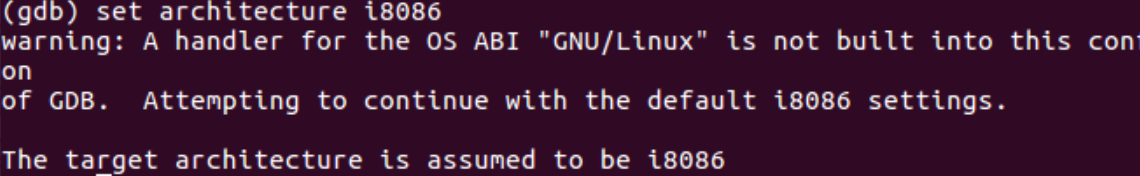
死机了…

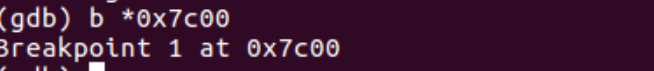


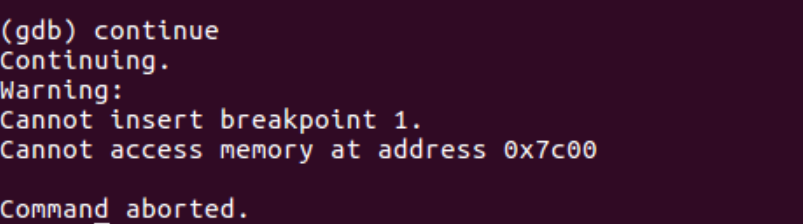
依次执行这几条代码

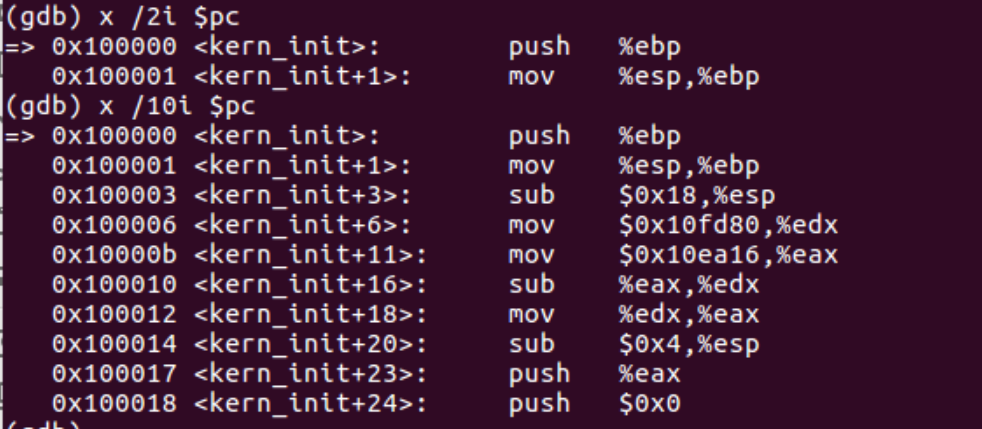


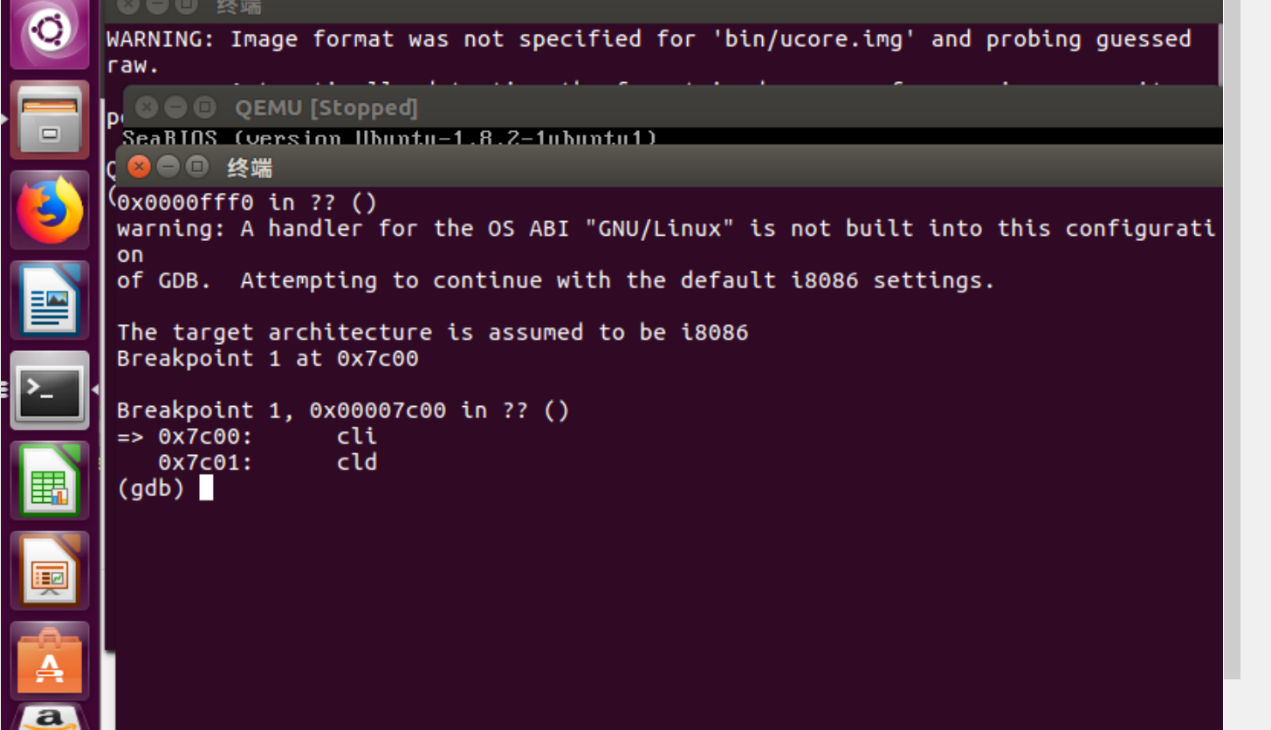
没好使…





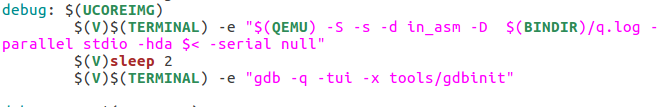


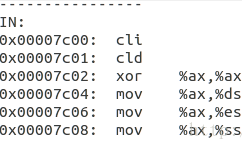


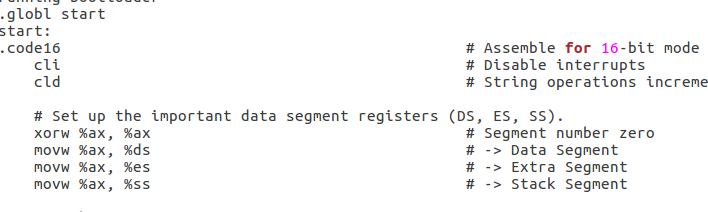
又好了

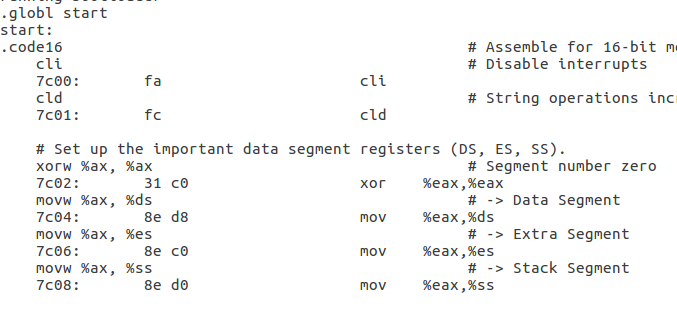
从0x7c00开始跟踪代码运行,将单步跟踪反汇编得到的代码与bootasm.S和 bootblock.asm进行比较。

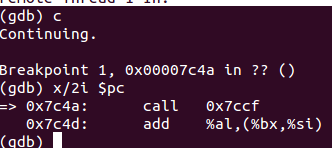
修改makefile:









断点设置正常。

练习3：分析bootloader进入保护模式的过程。（要求在报告中 写出分析）

**1为何开启A20，以及如何开启A20**

初始时A20为0，访问超过1MB的地址时，就会从0循环计数，将A20地址线置为1之后，才可以访问4G内存。A20地址位由8042控制，8042有2个有两个I/O端口：0x60和0x64。

打开流程：

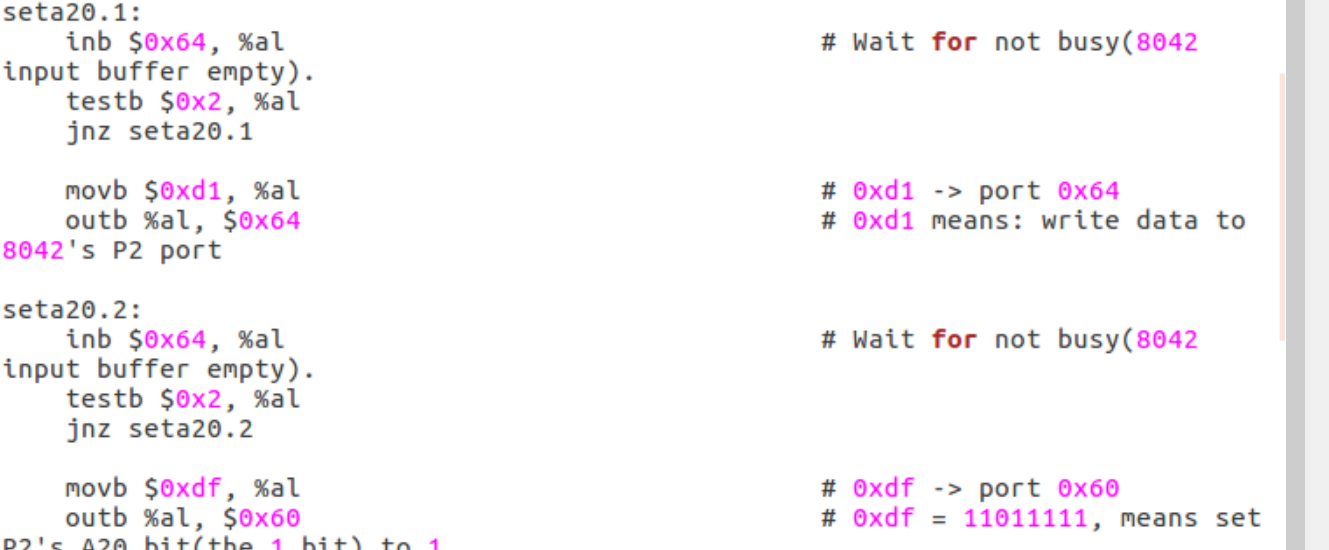
1.等待8042 Input buffer为空；

2.发送Write 8042 Output Port （P2）命令到8042 Input buffer；

3.等待8042 Input buffer为空；

4.将8042 Output Port（P2）得到字节的第2位置1，然后写入8042 Input buffer；

通过这段代码将关闭的a20地址线开启



**2如何初始化GDT表**

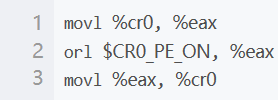
**1载入GDT表**1

****

2 进入保护模式：

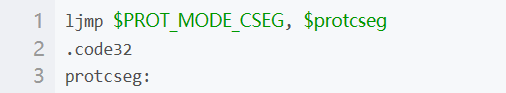
通过将cr0寄存器PE位置1便开启了保护模式

cro的第0位为1表示处于保护模式



### 3 通过长跳转更新cs的基地址:

​ 上面已经打开了保护模式，所以这里需要用到逻辑地址。$PROT\_MODE\_CSEG的值为0x80



4 设置段寄存器，并建立堆栈



5 转到保护模式完成，进入boot主方法



进入保护模式



将cr0寄存器置1