实验1 系统软件启动过程

班级：硬件2班 姓名：刘思雨

学号：171491324 日期：2019年10月14日

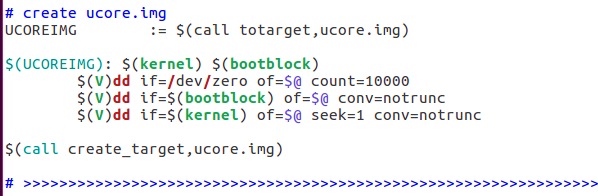
1. 理解通过make生成执行文件的过程。

列出本实验各练习中对应的OS原理的知识点，并说明本实验中的实现部分如何对应和体现了 原理中的基本概念和关键知识点。

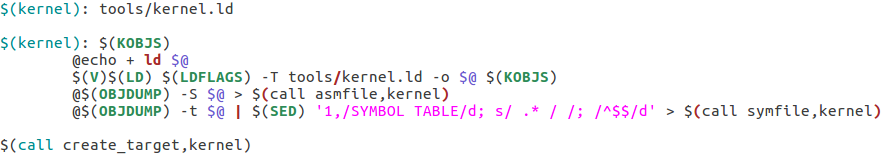
需要通过静态分析代码来了解：

操作系统镜像文件ucore.img是如何一步一步生成的？(需要比较详细地解释Makefile中每 一条相关命令和命令参数的含义，以及说明命令导致的结果)

1. 要生成ucore.img需要kernel和bootblock



1. 那如何生成kernel呢？



我们通过make V=来看一下执行了那些命令：

它调用了gcc，gcc来把一些c的源代码编译成了.o文件，也就是目标文件

然后通过ld，ld把这些目标文件会转换成一个执行程序，比如说其中的bootblock.out，

最终生成了两个代码，第一个是Bootloader,第二个就是Kernel Kernel事实上是Ucore的组成部分

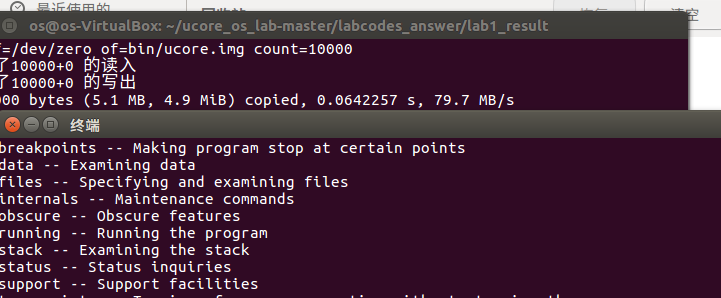
【-m 模拟指定的连接器  
-nostdlib 不使用标准库  
-T 指定命令文件  
-o 指定输出文件的名称

dd，把bootloader放到一个虚拟的硬盘中去，生成一个硬盘叫ucore.img count,硬件模拟器就会基于这个虚拟硬盘中的数据来执行相应的这个代码】

二．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）

为了熟悉使用qemu和gdb进行的调试工作，我们进行如下的小练习：

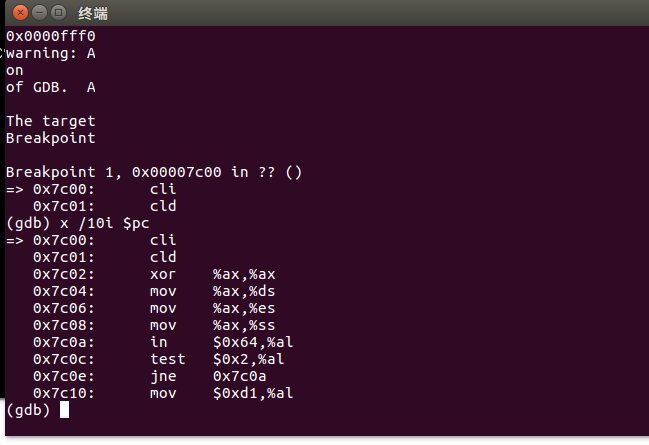
1. 从CPU加电后执行的第一条指令开始，单步跟踪BIOS的执行。



2. 在初始化位置0x7c00设置实地址断点,测试断点正常。

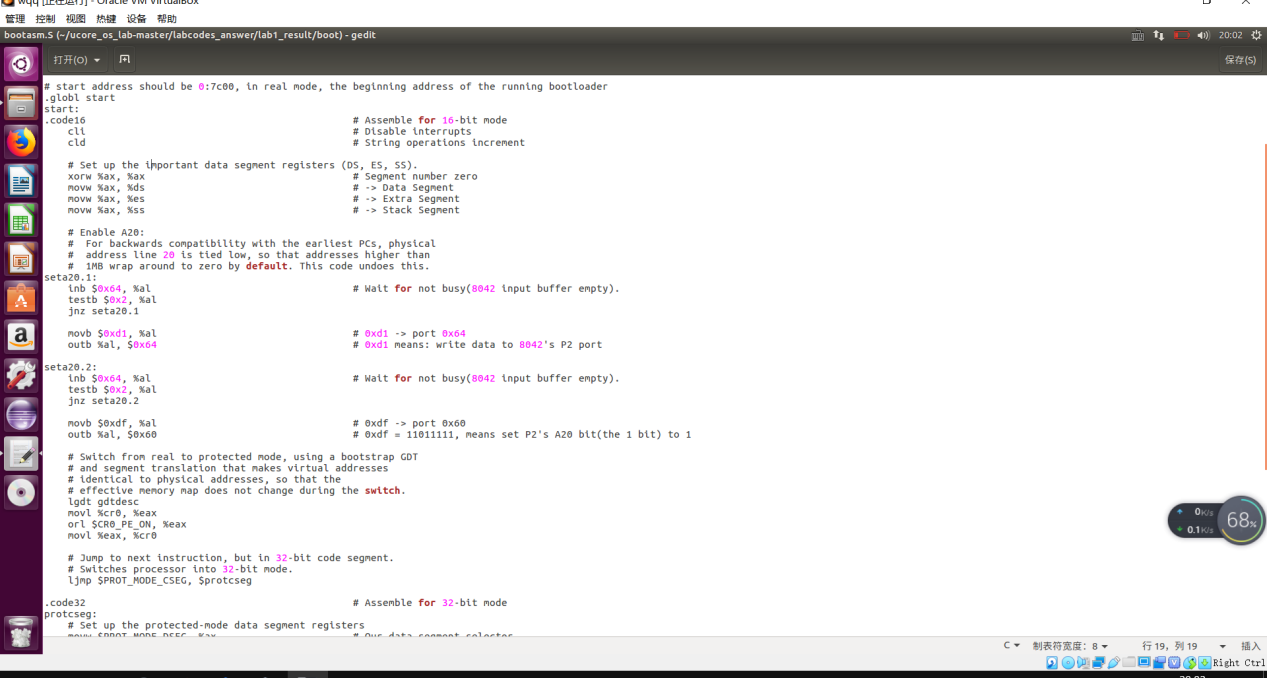
3. 从0x7c00开始跟踪代码运行,将单步跟踪反汇编得到的代码与bootasm.S和 bootblock.asm进行比较。

4. 自己找一个bootloader或内核中的代码位置，设置断点并进行测试。

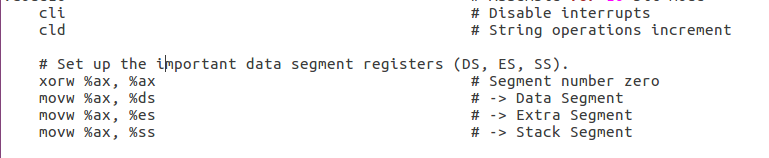


三．分析bootloader进入保护模式的过程。（要求在报告中 写出分析）

BIOS将通过读取硬盘主引导扇区到内存，并转跳到对应内存中的位置执行bootloader。请分 析bootloader是如何完成从实模式进入保护模式的。

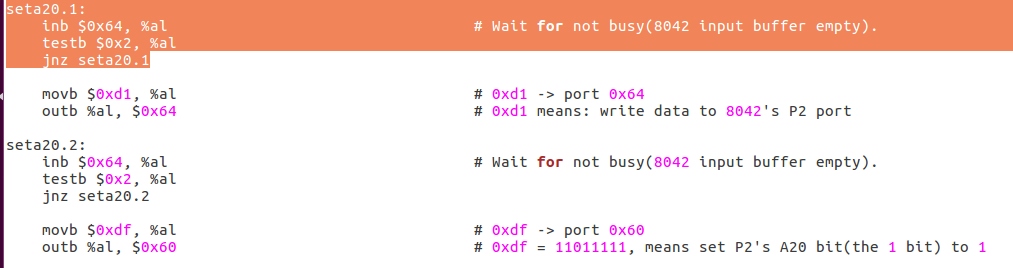


1. 关闭中断，将各个寄存器置为0



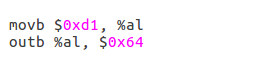
1. 开启A20

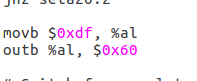
A20的地址线控制禁止时，则程序就像在8086中运行，1mb以上的地是不可访问的。而在保护模式下，A20地址线控制是要打开的，所以要通过键盘控制器上的A20线置于高电位，使得全部32位地址线可用。



#读取状态寄存器，等待8042键盘控制器闲置

 #判断输入缓存是否为空

# 0xd1表示写输出端口命令，参数随后通过0x60端口写入

# 通过0x60写入数据11011111 即将A20置1

1. 加载GDT 表



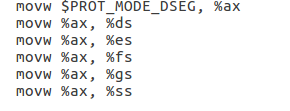
1. 将CR0的第0位置1



1. 长跳转到32位代码段，重装cs和EIP



1. 重装DS,ES 等段寄存器等



1. 转到保护模式完成，进入boot主方法

