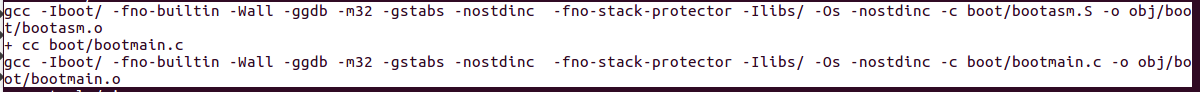
练习1：操作系统镜像文件ucore.img是如何一步一步生成的？

从代码来看，ucore.img是依赖于kernel和bootblock的。

Bootblock的编译部分分析：



上面的两段和下面的两端是一一对应的。

而且后面的这个编译是依赖于前面的那部分的，第一个是将 .S文件编译为.o文件，也就是将汇编语言文件直接生成可执行文件。第二个是将.c文件转换为.o文件，也就是用c语言文件生成可执行文件。

Kernel的编译部分分析：

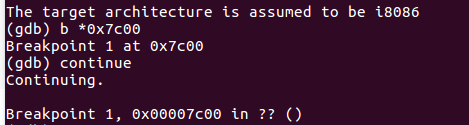


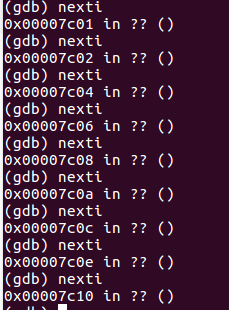
这里我只找到这个，这个是将.c文件转换成可执行文件 .o.

练习2：使用qemu执行并调试lab1中的软件。

1. 从cpu加电后执行第一步指令，单步跟踪BIOS的执行。
2. 在初始位置0x7c00设置实地址断点测试断点正常。

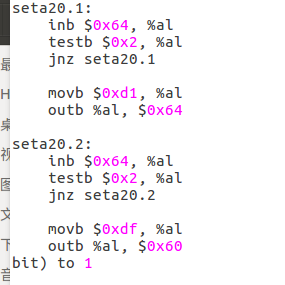
这里第一步和第二步是一起完成的。

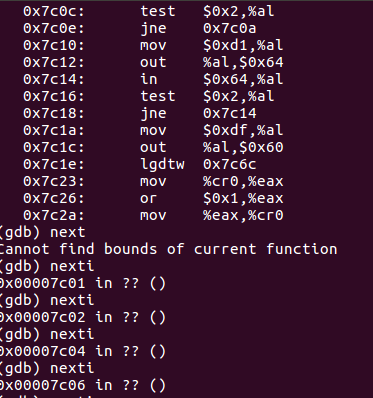
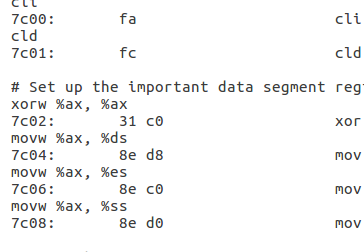




1. 从0x7c00开始跟踪代码运行。

比较，看下图中额可以知道：bootasm.S是汇编文件，bootloader.asm是汇编语言源程序文件。前者是由后者编译而来的。因为源文件就是汇编语言，所以看上去差距不大。





练习3：分析bootloader进入保护模式的过程。

1. 为何开启A20，以及如何开启A20？

Intel早期的8086 CPU提供了20根地址线,可寻址空间范围即0~2^20(00000H~FFFFFH)的 1MB内存空间。但8086的数据处理位宽位16位，，无法直接寻址1MB内存空间，所以8086提供 了段地址加偏移地址的地址转换机制。为了保持完全的向下兼 容性，IBM决定在PC AT计算机系统上加个硬件逻辑，来模仿以上的回绕特征，于是出现了 A20 Gate。他们的方法就是把A20地址线控制和键盘控制器的一个输出进行AND操作，这样 来控制A20地址线的打开（使能）和关闭（屏蔽\禁止）。在保护模式下，为了使能所有地址位的寻址能力，需要打开A20地址线控制，即需要通过向键 盘控制器8042发送一个命令来完成。

1. 如何初始化GDT表？

由于GDT 不能有GDT本身之内的描述符进行描述定义，所以处理器采用GDTR为GDT这一特殊的系统 段。

实模式：在bootloader接手BIOS的工作后，当前的PC系统处于实模式（16位模式）运行状态，在这种 状态下软件可访问的物理内存空间不能超过1MB，且无法发挥Intel80386以上级别的32位 CPU的4GB内存管理能力。

保护模式：只有在保护模式下，80386的全部32根地址线有效，可寻址高达4G字节的线性地址空间和物 理地址空间，可访问64TB的模式地址。

3.进入保护模式就是让cr0寄存器的pe为1

