**河北地质大学**

**《操作系统原理实验》**

实验名称：

学 号：

姓 名：

指导教师：

**项目1.1 引导扇区**

1.实验目的

(1) 了解计算机启动基本过程；

(2) 掌握引导扇区的特点及编码实现。

(3) 了解计算机运行过程中不同阶段内存的分布。

(4) 能够使用Bochs模拟器加载运行并进行调试。

2.实验内容

实现引导扇区完成以下任务：

在显示器上显示“自己的名字的拼音或英文名字”

利用Bochs调试跟踪引导程序执行过程。

为了熟悉使用Bochs的调试工作，进行如下练习：

从CPU加电后执行的第一条指令开始，单步跟踪BIOS。

在初始化位置0x7c00设置实地址断点，测试断点正常。

从0x7c00开始跟踪代码执行，将单步跟踪反汇编得到的代码与源代码进行比较。

3. 实验原理

计算加电后，CPU一开始会到固化区ROM读取BIOS，保证BIOS获得控制权。

之后，BIOS首先会检查硬件设备是否正常，同时并把检查结果写到1K 到 2K 的 RAM,这个区域因此也被称为 BIOS Data Area。同时还将中断向量以及 BIOS 程序运行所需要的Stack 设置置于 0 到 1K 的 RAM。

之后，BIOS从找到响应的启动设备，BIOS 将会查找引导扇区以开始 OS 的启动过程

BIOS 读取设备的第一个扇区也就是读取最前面的 512 个字节,如果发现它以 0xaa55结束,则 BIOS 认为它是一个引导扇区,将其装载到指定的内存位置(CS 设置为 0x0000,IP设置为 0x7c00,即物理内存 0x07c00)内存中,然后跳转到 0x07c00 处将控制权彻底交给这段引导代码。到此,计算机不再由 BIOS 中固有的程序来控制,而是变成由操作系统的一部分来控制。

4. 实验设计

//实验的过程

首先创建引导源程序bootloader.S

之后利用as汇编器对bootloader.S文件进行编译链接，生成引导扇区程序bootloader.com

接下来，使用Ld链接器对目标文件执行链接操作,生成可执行文件 bootloader.elf

之后,调用 objcopy 生成纯二进制文件,-j .text 选项表示只需要代码段,-O binary 选项指定输出格式为纯二进制文件,输出文件为 bootloader.com

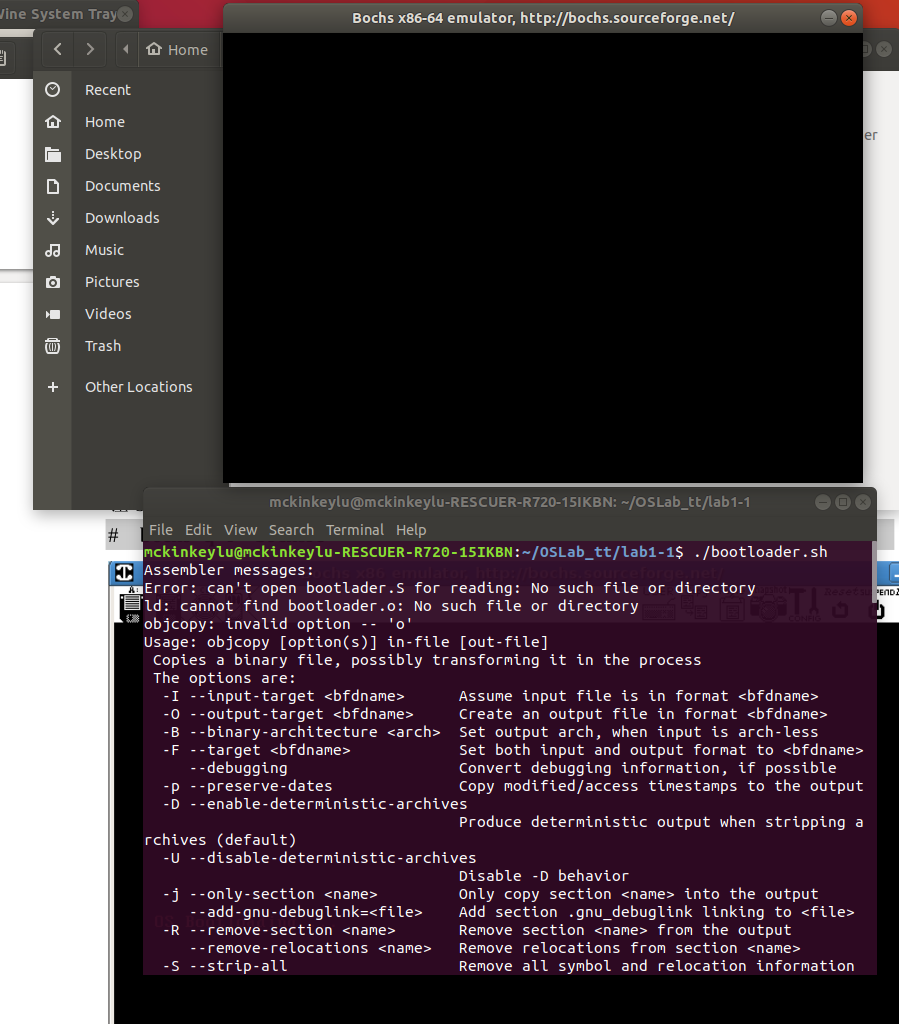
再利用 linux 系统的 dd 命令,生成镜像文件 bootloader.img

最后，使用让 bootloader.img 在 bochs 模拟的 x86 硬件环境中执行。

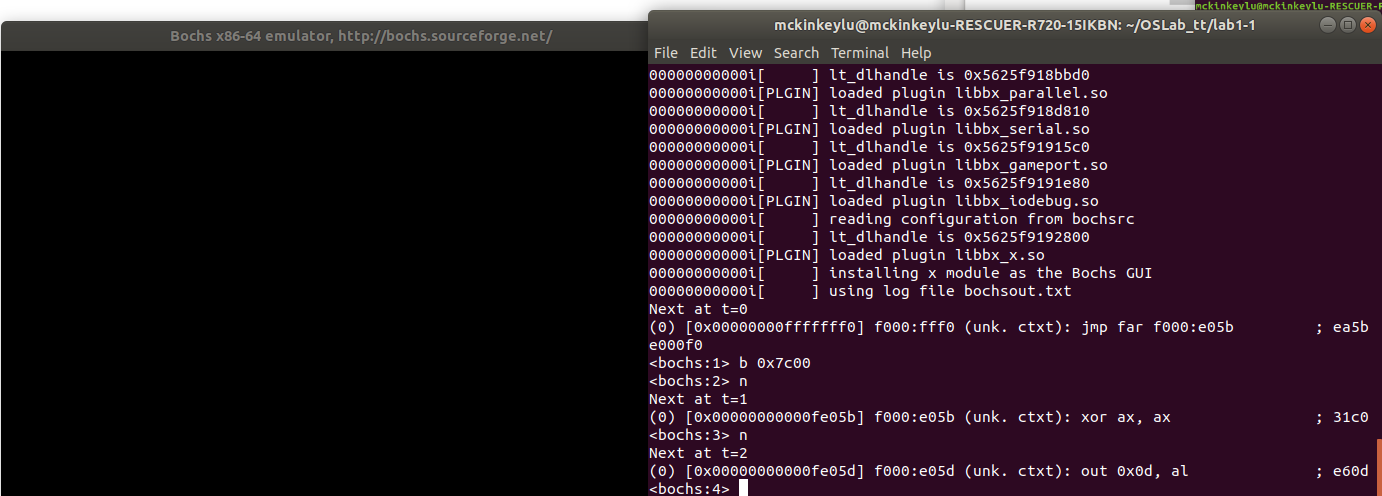
5. 运行结果及分析

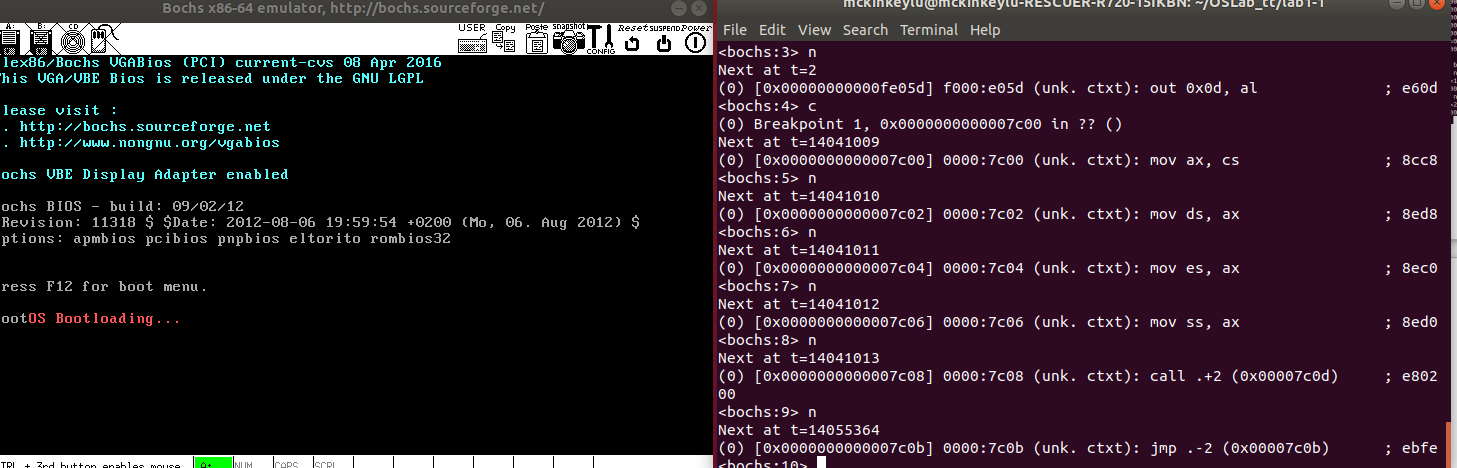
//运行结果及截图说明

首先进入x86环境进行执行

 解释：bootloader.sh为自定义shell文件。用于生成镜像文件以及运行boch

之后设置断点至0x7c00处，进行单步调试，观察指令执行情况

 接下来对引导扇区指令执行进行观察



我们会发现引导扇区汇编代码，程序调用子程序打印出 OS Bootloading....，最后跳转(子程序返回)进入死循环。

6. 总结

//对本部分实验的整体说明，做了什么，涉及到哪些知识点；哪些技术…

在本实验我明白OS如何被写入内存的，以及BISO功能：

开机后，计算机首先从固化区(ROM)，读取BIOS内容至内存，BIOS检查硬件设备，并将检测保存内存指定位置，之后，磁盘第一个扇区(以.0xaa55结尾）的引导扇区，进入子程序打印字符，之后进入死循环。

在本实验掌握了编写、编译、生成引导程序方法以及制作镜像文件的方法。学到了让引导程序在boch上运行以及调试程序的方法

//实验中遇到哪些问题，如何解决的

在本实验中，遇到键盘配置错误导致Bochs不能正常，解决的办法是将keymap语句注释掉。

//实验的收获,疑问，建议等

在本次实验中，学到了计算机BIOS固化区都有那些，以及编写、编译、生成、转换引导扇区为二进制文件的方法、bochs使用、调试代码、解决镜像加载出错的解决方法。