lab1实验报告

刘国涛 181860055 计算机科学与技术系

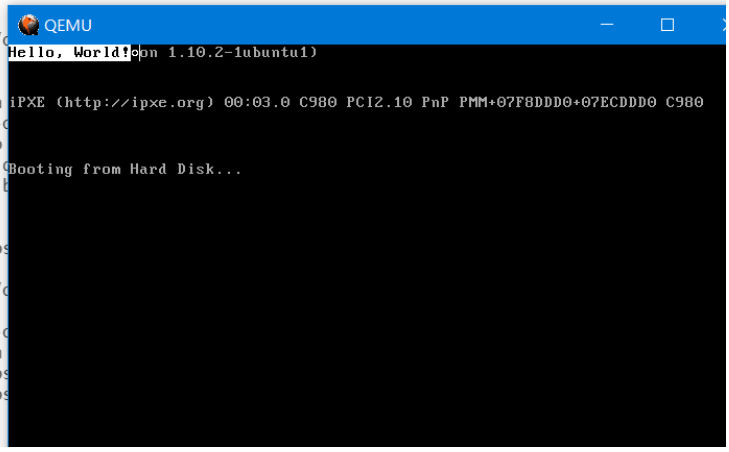
181860055@smail.nju.edu.cn

实验描述

实验进度:完成了所有实验内容

实验结果:

```
liuguotao@HP-PC:~/os2020/lab1$ make
cd bootloader; make bootloader.bin
make[1]: Entering directory '/home/liuguotao/o
gcc -c -m32 start.s -o start.o
start.s: Assembler messages:
start.s: Warning: end of file not at end of a
gcc -c -m32 -O1 -fno-stack-protector boot.c -o
ld -m elf_i386 -e start -Ttext 0x7c00 start.o
ld: warning: cannot find entry symbol start; D
objcopy -S -j .text -O binary bootloader.elf b
../utils/genboot.pl bootloader.bin
OK: boot block is 262 bytes (max 510)
make[1]: Leaving directory '/home/liuguotao/os
cd app; make app.bin
make[1]: Entering directory '/home/liuguotao/o
gcc -c -m32 app.s -o app.o
ld -m elf_i386 -e start -Ttext 0x8c00 app.o -o
objcopy -S -j .text -O binary app.elf app.bin
make[1]: Leaving directory '/home/liuguotao/os
cat bootloader/bootloader.bin app/app.bin > os
liuguotao@HP-PC:~/os2020/lab1$ make play
qemu-system-i386 os.img
```



代码修改:

1. **start.s:**
    1. 将实模式启动改为实模式启动后切换到保护模式
    2. 对DS ES FS GS SS进行初始化
        - 查看gdt,ds在第二项,因此初始化为(2<<3)
        - gs在第三项,初始化为(3<<3)
    3. 对栈顶指针ESP进行了初始化
    4. 切换完成后跳转到bootmain
2. **boot.c:**
    1. 在bootmain中改为通过 `readSect` 完成一号扇区的加载,并通过elf的调用进入到位于app.s的 `Hello World` 程序

3. **app.s:** 无修改

CPU 内存 BIOS 磁盘 主引导扇区 加载程序 操作系统

从通电到操作系统启动的流程如下:

电源稳定后, `CPU` 初始化内部寄存器,然后跳转到 `BIOS` 固件进行开机自检,将磁盘的 `主引导扇区` 加载到 `内存` 的0x7c00. `CPU` 执行从0x7c00开始执行 `加载程序` ,将 `操作系统` 装载入 `内存` 中,装载完成后跳转到 `操作系统` 起始处启动 `操作系统` .

他们之间的关系是:

1. CPU总是优先从内存获取指令
2. 内存里的数据来自于BIOS和磁盘
3. 主引导扇区包含了加载程序
4. 操作系统在装载前都存在于磁盘内,在加载程序执行完以后被装载到内存里