

Lab 1 实验报告

王科 201300008 203536673@qq.com

实验进度&设计思路

根据实验手册的指导，查阅手册和网络资源，在 `bootloader/start.s` 和 `bootloader/boot.c` 中修改、增加了相关代码，完成了所有实验目标。

简答题

1. CPU、内存、BIOS、磁盘、MBR、操作系统的关系

CPU 是中央处理器，内存包括 RAM 和 ROM，分别是随机存取存储器和只读存储器，用于存储数据和代码，BIOS 是一组固化到 ROM 上的程序，包括基本输入输出的程序、开机后自检程序和系统自启动程序，磁盘是利用磁记录技术存储数据的存储器，MBR 是 0 号柱面，0 号磁头，0 号扇区对应的扇区，操作系统是管理计算机硬件和软件资源的特殊用户程序。

开机时，BIOS 在自检通过后，会从硬盘中将 MBR 整个读取到内存中，执行 MBR 的引导程序，启动操作系统，全过程由 CPU 控制。

2. 中断向量表是什么？

用于存放异常处理程序或中断服务程序的入口地址。

3. 为什么段的大小最大为 64KB？

$64KB = 2^{16}B$ ，实地址模式下每个存储单元地址由 16 位段地址左移 4 位后与 16 位偏移量相加而得到，段内偏移量最多可寻址 64KB 空间。

4. 说明 `genboot.pl` 在检查文件是否大于 510B 后做了什么，并解释为什么这么做

首先将 `mbr.bin` 扩展到 510B，不足部分补 `"\0"`，然后在文件尾加上 `"\x55\xaa"`，即正确的魔数，此时文件大小为 512B。原因是磁盘的主引导扇区是 0 号柱面，0 号磁头，0 号扇区对应的扇区，大小为 512B，末尾两字节为魔数 `0x55` 和 `0xaa`，BIOS 启动设备时会检查设备首扇区末尾的两个字节是否为 `0x55` 和 `0xaa`。最终文件类似于下图。

```
00000180  7d e8 2e 00 cd 18 eb fe 47 52 55 42 20 00 47 65 |}.....GRUB .Ge|
00000190  6f 6d 00 48 61 72 64 20 44 69 73 6b 00 52 65 61 |om.Hard Disk.Rea|
000001a0  64 00 20 45 72 72 6f 72 0d 0a 00 bb 01 00 b4 0e |d. Error.....|
000001b0  cd 10 ac 3c 00 75 f4 c3 00 00 00 00 00 00 00 00 |...<.U.....|
000001c0  02 00 ee ff ff ff 01 00 00 00 ff ff 7f 02 00 00 |.....|
000001d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
*
000001f0  00 00 00 00 00 00 00 00 00 00 00 00 00 55 aa |.....U.|
00000200
```

5. 简述从加电开始，到 OS 开始执行为止，计算机是如何运行的

加电后，CPU 读取 ROM，运行 BIOS。BIOS 首先进行硬件自检，然后按照启动顺序读取排在第一位的设备的第一个扇区 MBR，将引导程序装入内存。主引导接着从分区表中找到第一个活动分区，然后读取并执行这个活动分区的 Bootloader，然后识别硬盘文件系统，将操作系统的代码和数据从硬盘加载到内存中，并跳转到操作系统的起始地址。控制权转交给 OS，开始执行。

实验结果

1. lab 1.1

```
> vim Makefile
1
total 32K
drwxrwxr-x 2 wk wk 4.0K Mar 2 14:51 .
drwxrwxr-x 5 wk wk 4.0K Mar 2 14:32 ..
-rwxrwxr-x 1 wk wk 354 Mar 2 14:37 genboot.pl
-rw-rw-r-- 1 wk wk 229 Mar 2 14:51 Makefile
-rwxrwxr-x 1 wk wk 512 Mar 2 14:39 mbr.bin
-rwxrwxr-x 1 wk wk 3.5K Mar 2 14:35 mbr.elf
-rw-rw-r-- 1 wk wk 624 Mar 2 14:34 mbr.o
-rw-rw-r-- 1 wk wk 558 Mar 2 14:34 mbr.s
> make clean
rm mbr.o
rm mbr.bin
rm mbr.elf
> l
total 20K
drwxrwxr-x 2 wk wk 4.0K Mar 2 14:51 .
drwxrwxr-x 5 wk wk 4.0K Mar 2 14:32 ..
-rwxrwxr-x 1 wk wk 354 Mar 2 14:37 genboot.pl
-rw-rw-r-- 1 wk wk 229 Mar 2 14:51 Makefile
-rw-rw-r-- 1 wk wk 558 Mar 2 14:34 mbr.s
> make play
gcc -c -m32 mbr.s -o mbr.o
ld -m elf_i386 -e start -Ttext 0x7c00 mbr.o -o mbr.elf
objcopy -S -j .text -O binary mbr.elf mbr.bin
```

```
Machine View
SeaBIOS (version 1.14.0-2)

iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP FMM+07F8B5B0+07ECB5B0 CA00

Booting from Hard Disk...
```

2. lab 1.2

```
> make clean; make os.img; make play
cd bootloader; make clean
make[1]: Entering directory '/home/wk/os2022/lab1-201300008/bootloader'
rm -rf *.o *.elf *.bin
make[1]: Leaving directory '/home/wk/os2022/lab1-201300008/bootloader'
cd app; make clean
make[1]: Entering directory '/home/wk/os2022/lab1-201300008/app'
rm -rf *.o *.elf *.bin
make[1]: Leaving directory '/home/wk/os2022/lab1-201300008/app'
rm -f os.img
cd bootloader; make bootloader.bin
make[1]: Entering directory '/home/wk/os2022/lab1-201300008/bootloader'
gcc -c -m32 start.s -o start.o
gcc -c -m32 -O1 -fno-stack-protector boot.c -o boot.o
ld -m elf_i386 -e start -Ttext 0x7c00 start.o boot.o -o bootloader.elf
objcopy -S -j .text -O binary bootloader.elf bootloader.bin
../utils/genboot.pl bootloader.bin
OK: boot block is 317 bytes (max 510)
make[1]: Leaving directory '/home/wk/os2022/lab1-201300008/bootloader'
cd app; make app.bin
make[1]: Entering directory '/home/wk/os2022/lab1-201300008/app'
gcc -c -m32 app.s -o app.o
ld -m elf_i386 -e start -Ttext 0x8c00 app.o -o app.elf
objcopy -S -j .text -O binary app.elf app.bin
make[1]: Leaving directory '/home/wk/os2022/lab1-201300008/app'
cat bootloader/bootloader.bin app/app.bin > os.img
qemu-system-i386 os.img
WARNING: Image format was not specified for 'os.img' and probing guessed raw.
Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
Specify the 'raw' format explicitly to remove the restrictions.
```

```
Machine View
SeaBIOS (version 1.14.0-2)

iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP FMM+07F8B5B0+07ECB5B0 CA00

Hello, World!

Booting from Hard Disk...
```

3. lab 1.3

```
> make clean; make os.img; make play
cd bootloader; make clean
make[1]: Entering directory '/home/wk/os2022/lab1-201300008/bootloader'
rm -rf *.o *.elf *.bin
make[1]: Leaving directory '/home/wk/os2022/lab1-201300008/bootloader'
cd app; make clean
make[1]: Entering directory '/home/wk/os2022/lab1-201300008/app'
rm -rf *.o *.elf *.bin
make[1]: Leaving directory '/home/wk/os2022/lab1-201300008/app'
rm -f os.img
cd bootloader; make bootloader.bin
make[1]: Entering directory '/home/wk/os2022/lab1-201300008/bootloader'
gcc -c -m32 start.s -o start.o
gcc -c -m32 -O1 -fno-stack-protector boot.c -o boot.o
ld -m elf_i386 -e start -Ttext 0x7c00 start.o boot.o -o bootloader.elf
objcopy -S -j .text -O binary bootloader.elf bootloader.bin
../utils/genboot.pl bootloader.bin
OK: boot block is 264 bytes (max 510)
make[1]: Leaving directory '/home/wk/os2022/lab1-201300008/bootloader'
cd app; make app.bin
make[1]: Entering directory '/home/wk/os2022/lab1-201300008/app'
gcc -c -m32 app.s -o app.o
ld -m elf_i386 -e start -Ttext 0x8c00 app.o -o app.elf
objcopy -S -j .text -O binary app.elf app.bin
make[1]: Leaving directory '/home/wk/os2022/lab1-201300008/app'
cat bootloader/bootloader.bin app/app.bin > os.img
```

```
Machine View
SeaBIOS (version 1.14.0-2)

iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP FMM+07F8B5B0+07ECB5B0 CA00

Hello, World!

Booting from Hard Disk...
```

思考与总结

1. 三个子实验的代码放在同一个文件中，切换时手动注释略显繁琐，可以考虑分为三个文件，并修改 `Makefile`，在 `make` 时传入参数用于选择文件。
2. 因为上学期PA实验使用的是riscv系统，对段寄存器的功能不熟悉，虽然完成了Lab1，但是对某些细节仍不清楚，需要进一步RTFM。