

# OS Lab1 Report

## 1. 个人信息

姓名：余帅杰

学号：181860077

邮箱：[3121416933@qq.com](mailto:3121416933@qq.com)

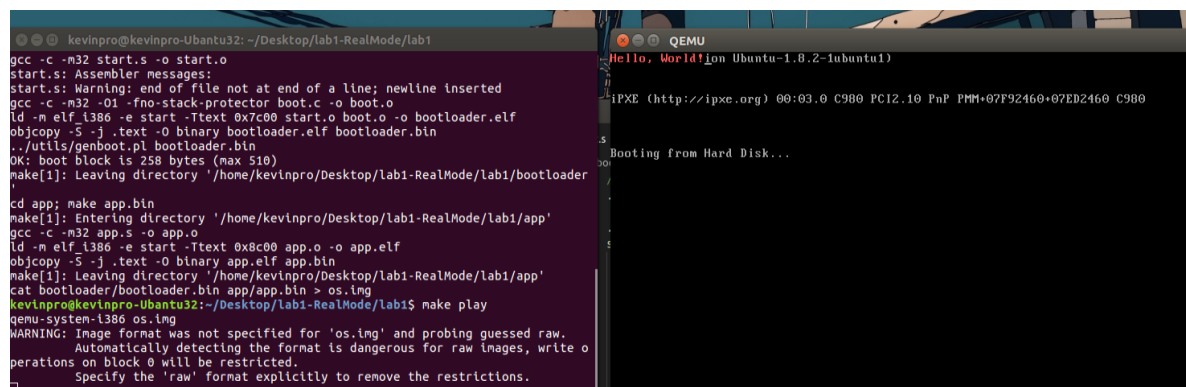
## 2. 实验进度

我完成了所有的内容

成功的分别在实模式，保护模式以及保护模式调用磁盘的条件下打印了Hello World（3个任务）

## 3. 实验结果

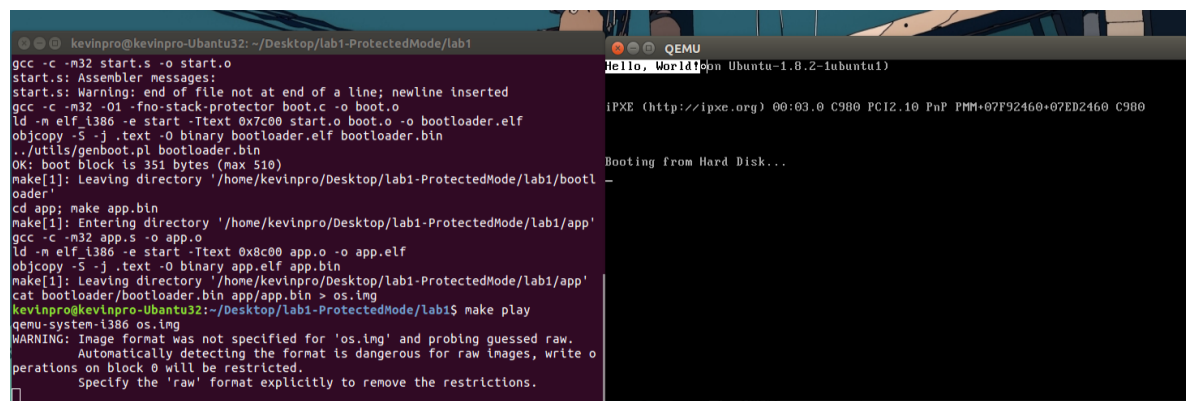
先是保护模式的



```
kevinpro@kevinpro-Ubuntu32: ~/Desktop/lab1-RealMode/lab1
gcc -c -m32 start.s -o start.o
start.s: Assembler messages:
start.s: Warning: end of file not at end of a line; newline inserted
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 258 bytes (max 510)
make[1]: Leaving directory '/home/kevinpro/Desktop/lab1-RealMode/lab1/bootloader'

cd app; make app.bin
make[1]: Entering directory '/home/kevinpro/Desktop/lab1-RealMode/lab1/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/kevinpro/Desktop/lab1-RealMode/lab1/app'
cat bootloader/bootloader.bin app/app.bin > os.img
kevinpro@kevinpro-Ubuntu32: ~/Desktop/lab1-RealMode/lab1$ make play
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 o
perations on block 0 will be restricted.
Specify the 'raw' format explicitly to remove the restrictions.
```

然后是保护模式下实现Hello world的



```
kevinpro@kevinpro-Ubuntu32: ~/Desktop/lab1-ProtectedMode/lab1
gcc -c -m32 start.s -o start.o
start.s: Assembler messages:
start.s: Warning: end of file not at end of a line; newline inserted
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 351 bytes (max 510)
make[1]: Leaving directory '/home/kevinpro/Desktop/lab1-ProtectedMode/lab1/bootl
oader'

cd app; make app.bin
make[1]: Entering directory '/home/kevinpro/Desktop/lab1-ProtectedMode/lab1/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/kevinpro/Desktop/lab1-ProtectedMode/lab1/app'
cat bootloader/bootloader.bin app/app.bin > os.img
kevinpro@kevinpro-Ubuntu32: ~/Desktop/lab1-ProtectedMode/lab1$ make play
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 o
perations on block 0 will be restricted.
Specify the 'raw' format explicitly to remove the restrictions.
```

最后是提交版本的

```
kevinpro@kevinpro-Ubuntu32: ~/workspace/lab1/lab1-181860077/lab1
start.s: Warning: end of file not at end of a line; newline inserted
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 299 bytes (max 510)
make[1]: Leaving directory '/home/kevinpro/workspace/lab1/lab1-181860077/lab1/bootloader'
cd app; make app.bin
make[1]: Entering directory '/home/kevinpro/workspace/lab1/lab1-181860077/lab1/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/kevinpro/workspace/lab1/lab1-181860077/lab1/app'
cat bootloader/bootloader.bin app/app.bin > os.img
kevinpro@kevinpro-Ubuntu32:~/workspace/lab1/lab1-181860077/lab1$ make play
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.
```

(对应的文件夹名字也可以看到)

## 4. 实验修改的代码位置

实验课上已经给了实验的实模式代码。

对于保护模式下实现的版本我修改的是start.s的文件。

最后的保护模式调用磁盘版本的，我修改了start.s以及boot.c的内容（bootmain）。

## 5. 自由选择的报告内容

下面的问题是我在解决问题的过程中产生的问题和部分的思考题，通过搜索解决，在此记录总结一下（可能会有点多）

### 问题1：code16和code32是啥意思，什么区别

CODE16是thumb指令, CODE32是arm指令。CODE16，CODE32作用是告诉编译器编译成相应指令。

1、使用了 code16 是告诉编译器：下面的代码要编译成 16 位代码。更易于理解地说：是告诉编译器，下面的代码要运行在 16 位模式（实模式）下，你要帮我编译成实模式的代码。在这种情况下，要注意的是：16 位代码的寻址模式比 32 位代码的寻址少很多。

2、使用了 code32 是告诉编译器，下面的代码要编译成 32 位代码。更易于理解地说：是告诉编译器，下面的代码要运行在 32 保护模式下，因此，你要帮我编译成 32 位的代码。

16 位代码与 32 位代码，除了使用的数据宽度区别外，另一个重要的区别就是：寻址模式。

16 位下：只支持 [si]、[di]、[bp]、[bx] 这些基址寄存器与变址寄存器的间接寻址以及：[bx+si]、[bx+di]、[bp+si]、[bp+di] 基址 + 变址的间接寻址。32 位下寻址模式就丰富得多了，并不局限于由基址与变址组合成的寻址。扩大到所有的 GPRs 寄存器。codd 16 与 code 32 同时出现在一个汇编语言文件里，主要是 OS 的实模式引导之类的程序。用来从实模式切换到保护模式时使用。绝大多数的汇编源文件，不大可能同时出现这两个指示命令字

查询得到的信息来源：<http://bbs.chinaunix.net/thread-1975547-1-1.html>

### 问题2：为什么gdt第一个表项是空的？

有一个特殊的选择子称为空(Null)选择子，它的Index=0，TI=0，而RPL字段可以为任意值。空选择子有特定的用途，当用空选择子进行存储访问时会引起异常。空选择子是特别定义的，它不对应于全局描述符表GDT中的第0个描述符，因此处理器中的第0个描述符总不被处理器访问，一般把它置成全0

### 问题3：为什么要打开A20开关？

保护模式是需要将地址位从20位切换到32位（如果是64位系统，则是切换到64位）。所以先打开这个开关，表明将32位地址总线打开。（兼容性）

#### 问题4：加载GDT的一些细节

引导加载器执行 lgdt指令来把指向 gdt 的指针 gdtDesc加载到全局描述符表（GDT）寄存器中 同时 LGDT 要先载入GDT的大小, 然后才是gdt的地址，这也是为什么start.s里结构是如下的

```
1 gdtDesc:
2     .word (gdtDesc - gdt - 1)
3     .long gdt
```