

学号：221220113

姓名：仲瑞泉

邮箱：1448853658@qq.com

Lab1.1

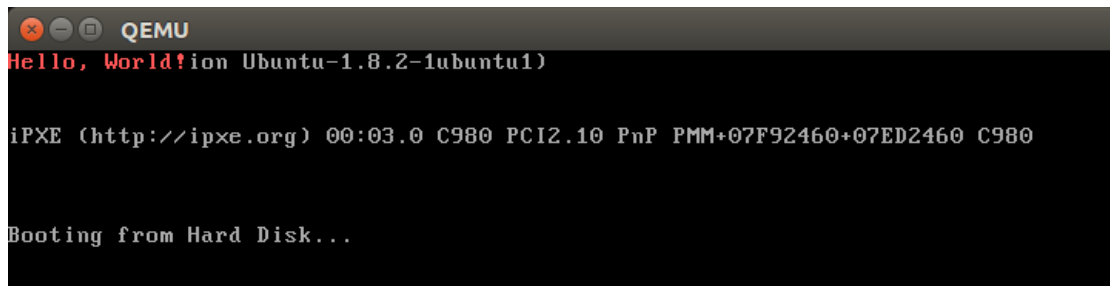
这一小节的实验流程：BIOS 自检，TODO：显示 HELLOWORLD。  
实模式下可以利用 BIOS 中断 INT \$0x10(%ah = 0x13) 显示字符串。用到的参数如下：

10	13	显示字符串(适用AT)	ES:BP=串地址
			CX=串长度
			DH,DL=起始行,列
			BH=页号
			AL=0,BL=属性
			串:char,char,...
			AL=1,BL=属性
			串:char,char,...
			AL=2
			串:char,attr,char,attr,...
			AL=3
			串:char,attr,char,attr,...

在中断前，设置好参数：

- message 地址 -> %bp
- message 长度 -> %cx
- 中断功能 0x13 -> %ah
- 仅显示字符，光标跟随移动 0x01 -> %al
- 视频页码 0x0 -> %bh
- 显示属性 0x0c -> %bl
- (关于显示属性，没有找到详细资料，尝试更改为 0x0d,0x00，发现与字符颜色有关)
- 显示位置 0 行 0 列 0x0000 -> %ex

设置好参数后，通过 INT \$0x10 即可显示字符串，1.1 节实验完成。



## Lab1.2

这一小节的实验流程：BIOS 自检，TODO：引导 CPU 进入保护模式，并在保护模式下显示 HELLOPOWORLD。

(1) 要进入保护模式，主要需要分配好 GDT，以进行保护模式下的寻址。

根据 GDT 初始化的具体格式：

```
# .word limit[15:0],base[15:0]
# .byte base[23:16],(0x90|(type)),(0xc0|(limit[19:16])),base[31:24]
```

(其中 type 含义见参考资料 lab1.pdf)

代码段如下拼凑：

Base:0x0

Limit:0xffffffff

Type:可执行、可读,代码段, 1010

数据段如下拼凑：

Base:0x0

Limit:0xffffffff

Type:可读写,数据段, 0010

还有一个段用于显存映射，下面再进行解析。

到此，GDT 分配完成，其余细节比较简单，跟着注释做即可。

(2) 要在保护模式下显示 HELLOWORLD，需要进行显存映射。

一般显存起点在 0xb8000，要在保护模式下对 0xb8000 处读写，可以为显存分配一个段基址为 0xb8000 的段，将其索引存入段选择子 %gs (意义上和 graphic segment 可以搭个边)，通过 %gs : OFFSET 进行索引。

其 GDT 初始化如下拼凑：

Base:0xb8000

Limit:0xffffffff

Type:可读写,数据段, 0010

另外，VGA 显存中，某字符的具体位置如下：

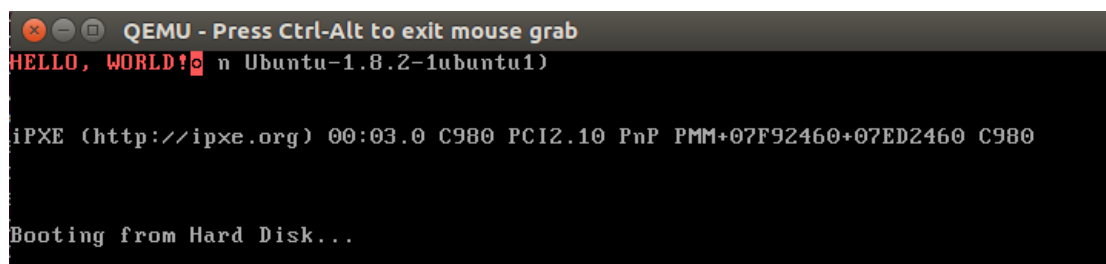
$$\text{offset} = (\text{row} * 80 + \text{col}) * \text{CHARACTER\_SIZE}$$

其中 CHARACTER\_SIZE=2，即一个字符占两个字节（颜色码+字符码）。

例如，HELLOWORLD!总计占据一行，假设在第 0 行，则第 i 个字符的偏移量为 2\*i。

对于 HELLO, WORLD!\n\0, 共 13 个字符，利用 eax 中的 ah 作为颜色码 0x0c，利用 al 存放 ascii 码，依次放入显存即可。

1.2 节实验完成。



## Lab1.3

这一小节的实验流程：BIOS 自检，TODO：进入保护模式，加载引导程序到指定位置，执行引导程序。

在进入保护模式上与 lab1.2 并无区别。

为了加载引导程序到指定位置，可以利用给出的接口 readSect(void\* dst, int offset)，其作用为将 offset 扇区的内容读入 dst 位置处。

在 app/Makefile 中可以找到规定的程序入口地址 0x8c00。

也知道第一扇区中装载的就是引导程序 app.s，于是将第一扇区内容读入 0x8c00，然后跳转 0x8c00 执行程序显示 HELLOWORLD。

1.3 节实验完成。

