os-lab1

181180050 孔孟荀

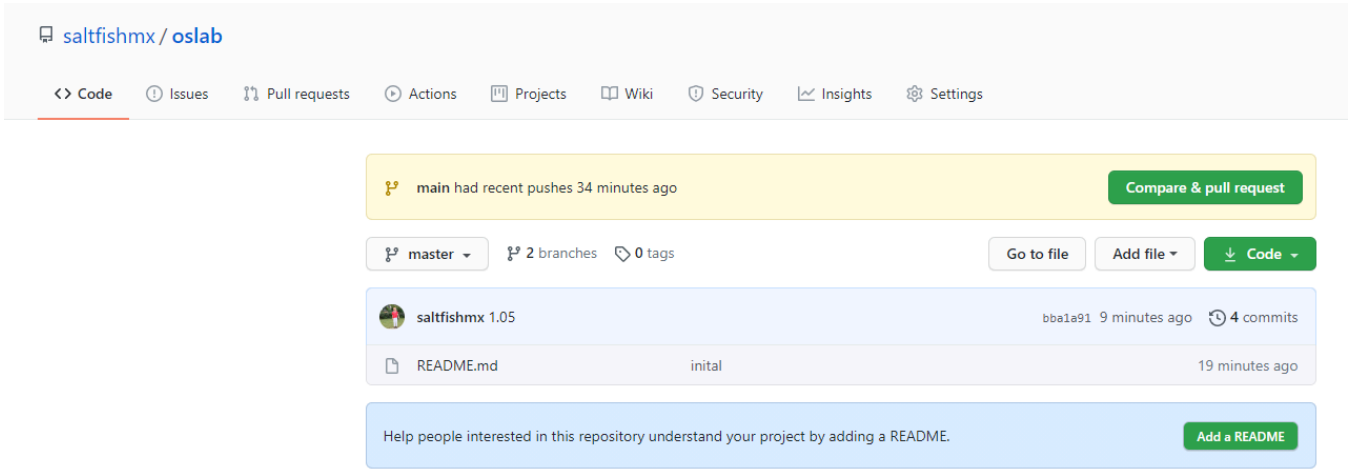
1571589383@qq.com

1：实验环境

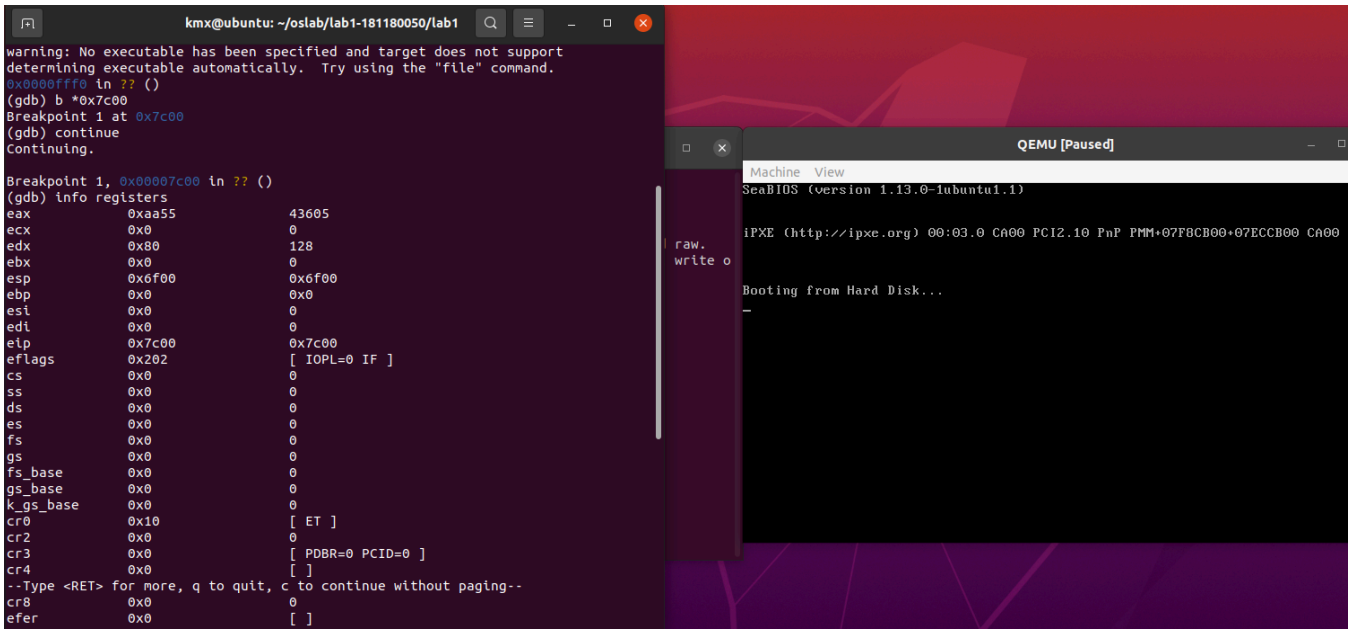
1.2 实验环境搭建

```
kmx@ubuntu:~$ dpkg --print-architecture
amd64
kmx@ubuntu:~$ $dpkg --print-foreign-architectures
--print-foreign-architectures: command not found
kmx@ubuntu:~$ dpkg --print-foreign-architectures
i386
kmx@ubuntu:~$ apt-get install gcc-multilib g++-multilib
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission d
enied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are y
ou root?
kmx@ubuntu:~$ sudo apt-get install gcc-multilib g++-multilib
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  g++ g++-9 g++-9-multilib gcc-9-multilib lib32asan5 lib32atomic1
  lib32gcc-9-dev lib32gcc-s1 lib32gomp1 lib32itm1 lib32quadmath0
  lib32stdc++-9-dev lib32stdc++6 lib32ubsan1 libc6-dev-i386 libc6-dev-x32
  libc6-i386 libc6-x32 libstdc++-9-dev libx32asan5 libx32atomic1
  libx32gcc-9-dev libx32gcc-s1 libx32gomp1 libx32itm1 libx32quadmath0
  libx32stdc++-9-dev libx32stdc++6 libx32ubsan1
```

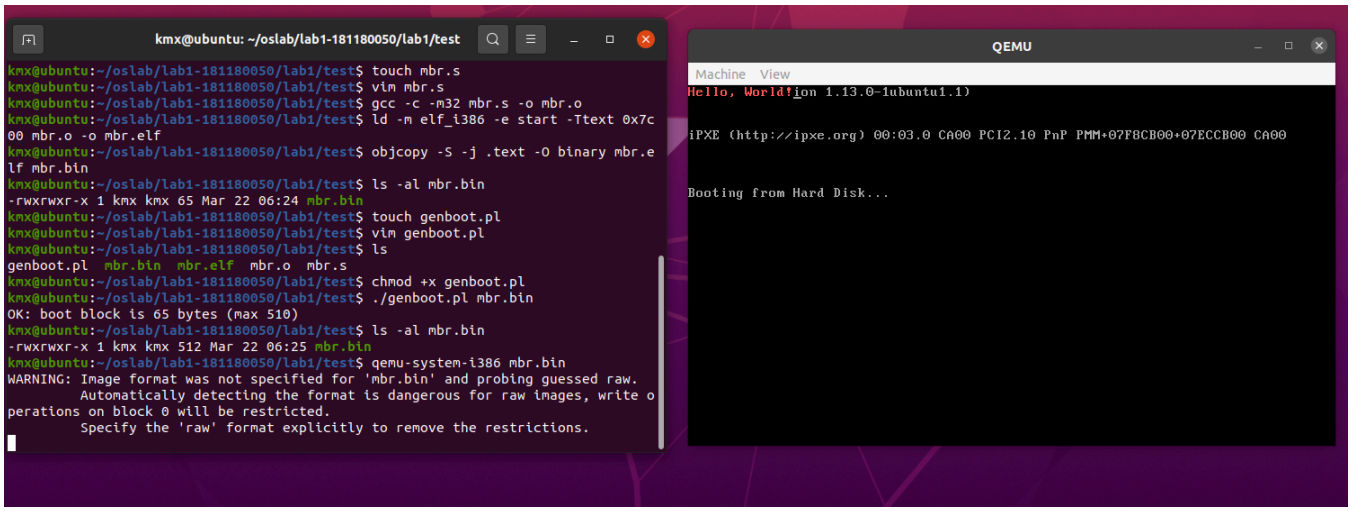
在github创建远程仓库并且关联


saltfishmx / oslab
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
main had recent pushes 34 minutes ago
Compare & pull request
master
2 branches
0 tags
Go to file
Add file
Code
saltfishmx 1.05
bba1a91 9 minutes ago
4 commits
README.md
inital
19 minutes ago
Help people interested in this repository understand your project by adding a README.
Add a README

1.3 代码运行与调试


kmx@ubuntu: ~/oslab/lab1-181180050/lab1
warning: No executable has been specified and target does not support
determining executable automatically. Try using the "file" command.
0x0000fff0 in ?? ()
(gdb) b *0x7c00
Breakpoint 1 at 0x7c00
(gdb) continue
Continuing.
Breakpoint 1, 0x00007c00 in ?? ()
(gdb) info registers
eax 0xaa55 43605
ecx 0x0 0
edx 0x80 128
ebx 0x0 0
esp 0x6f00 0x6f00
ebp 0x0 0x0
esi 0x0 0
edi 0x0 0
eip 0x7c00 0x7c00
eflags 0x202 [IOPL=0 IF]
cs 0x0 0
ss 0x0 0
ds 0x0 0
es 0x0 0
fs 0x0 0
gs 0x0 0
fs_base 0x0 0
gs_base 0x0 0
k_gs_base 0x0 0
cr0 0x10 [ET]
cr2 0x0 0
cr3 0x0 [PDBR=0 PCID=0]
cr4 0x0 [ ]
--Type <RET> for more, q to quit, c to continue without paging--
cr8 0x0 0
efer 0x0 [ ]
raw.
write o
QEMU [Paused]
Machine View
SeaBIOS (version 1.13.0-1ubuntu1.1)
iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+07F8CB00+07ECCB00 CA00
Booting from Hard Disk...

1.4简单上手



2相关资料

各种名词含义和关系：
主引导扇区，又叫主引导记录：MBR：是计算机开机以后访问硬盘时所必须要读取的第一个扇区。

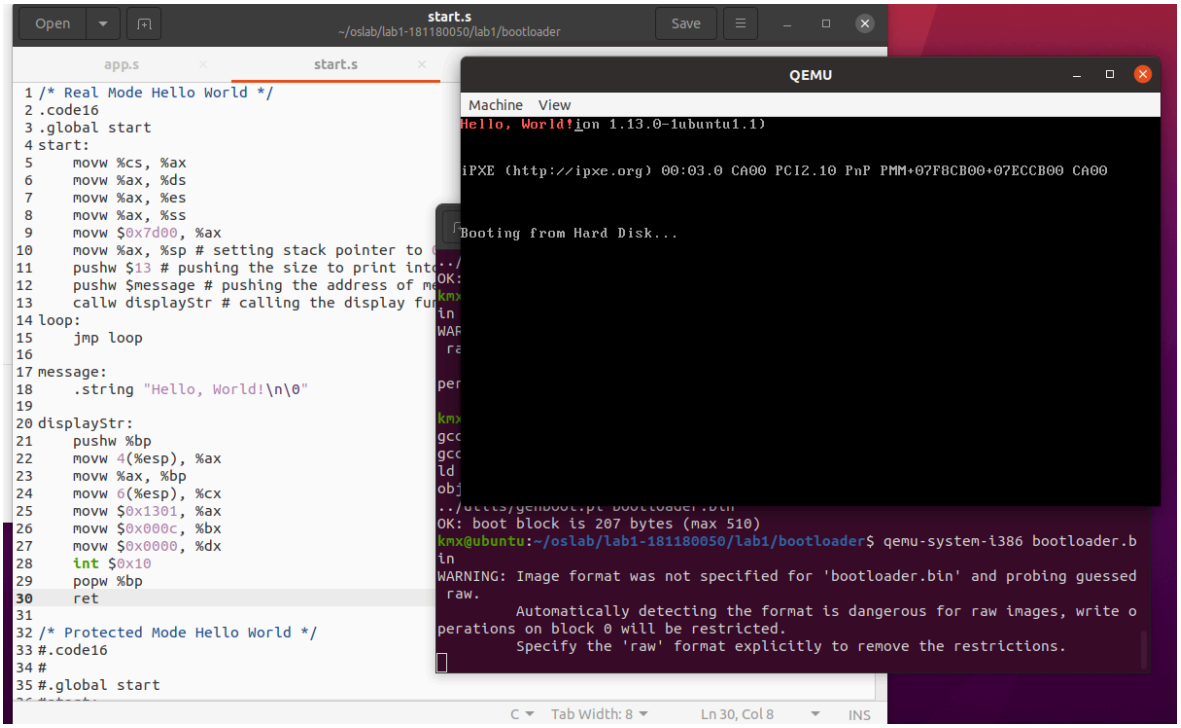
3正式实验

1.理解一下框架：

- genboot.pl功能是调整bootloader.bin的大小为512字节，让他成为一个真正的mbr
- ld -e :

> -e ENTRY'–entry=ENTRY’
> 使用符号ENTRY作为你的程序的开始执行点,而不是使用缺省的进入点.如果没有叫做ENTRY的符号,连接器
> 会企图把ENTRY作为一个数字进行分析,并使用它作为入口地址(数字会被解释为10进制的;你可以使用前
> 导的’0x’强制为16进制,或’0’作为8进制.)

2. 实模式下实现hello world



3. 保护模式下实现hello world

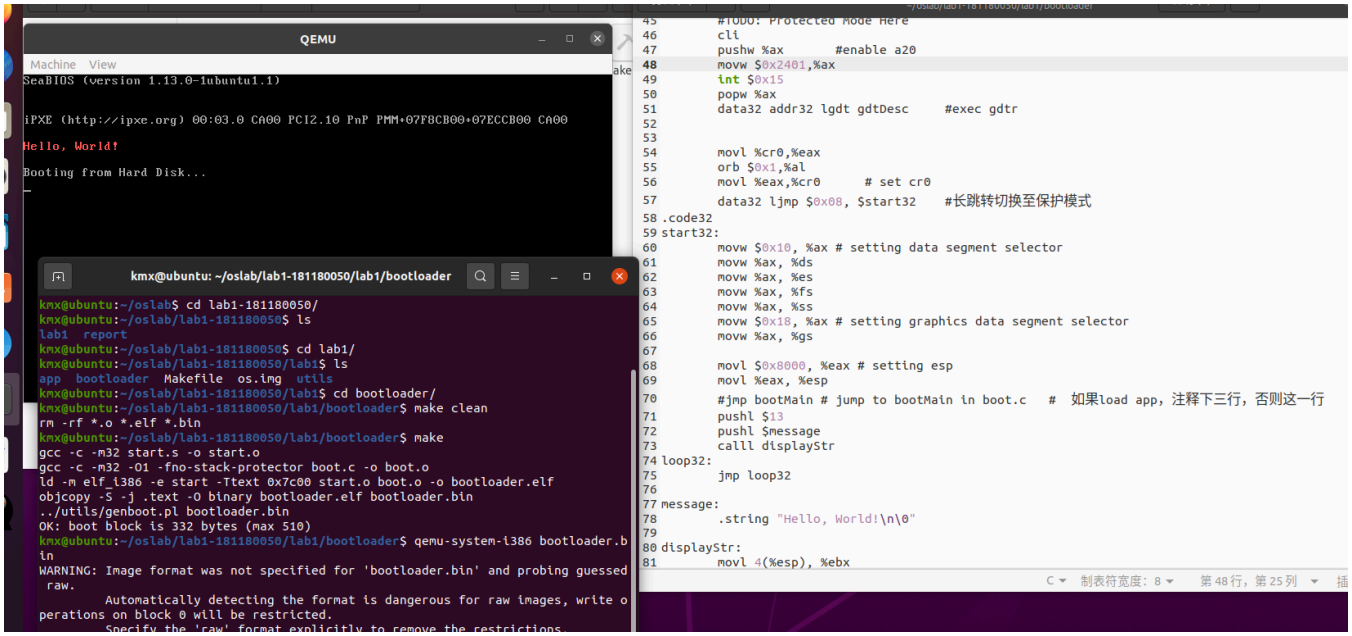
   主要仿照lab1.pdf给的框架，上网查了下怎么开启a20数据总线

```
pushw %ax       #enable a20
movw $0x2401,%ax
int $0x15
popw %ax
```

理解gdt、gdtDesc的代码：

> .byte/.word/.long
> 语法：.byte/.word/.long expressions
> 预留1个字节/字/双字，并将这个字节的内容赋值为expression，若是用逗号隔开的多个expression，则为预留多个这样的字节/字/双字，并将它们的内容依次赋值。

实验结果



4.加载磁盘中的程序并运行

完成boot.c中的bootmain()即可