# Lab 1: Booting a PC

## 前置条件

学习 Lab1 前应：

- 提前配置好环境，可参考环境准备
- 学习相关基础知识
  - 学习 x86汇编语言
  - 会用gdb+qemu对OS进行汇编、C级别调试，可参考gdb cheatsheet

## Part 1: PC Bootstrap

学习目标：能够更加熟悉x86汇编语言，以及PC启动的整个过程，而且也会首次学习使用QEMU软件来仿真xv6操作系统，并且配合GDB对操作系统的运行进行调试。

作业：完成Exercise1-2。

Lab 1: Booting a PC

## 模拟 x86

下载 lab 代码,按照以下步骤编译运行

```
cyl@ubuntu:~$ mkdir ~/6.828
cyl@ubuntu:~$
cyl@ubuntu:~$
cyl@ubuntu:~$ cd ~/6.828

cyl@ubuntu:~/6.828$ git clone https://pdos.csail.mit.edu/6.828/2018/jos.git lab
Cloning into 'lab'...
cyl@ubuntu:~/6.828$ cd lab/

cyl@ubuntu:~/6.828/lab$ make
+ as kern/entry.S
+ cc kern/entrypgdir.c
+ cc kern/init.c
+ cc kern/console.c
+ cc kern/monitor.c
+ cc kern/printf.c
+ cc kern/kdebug.c
+ cc lib/printfmt.c
+ cc lib/readline.c
+ cc lib/string.c
+ ld obj/kern/kernel
ld: warning: section `.bss' type changed to PROGBITS
+ as boot/boot.S
+ cc -Os boot/main.c
+ ld boot/boot
boot block is 390 bytes (max 510)
```

```
+ mk obj/kern/kernel.img


# run QEMU
cyl@ubuntu:~/6.828/lab$ make qemu

# 如果你是在服务器等其他无图形界面的环境下实验可以执行下面的命令启动
# make qemu-nox
```

若运行成功，则显示如下：

```
sed "s/localhost:1234/localhost:26000/" < .gdbinit.tmpl > .gdbinit
qemu-system-i386 -drive file=obj/kern/kernel.img,index=0,media=disk,format=raw -
serial mon:stdio -gdbtcp::26000 -D qemu.log
6828 decimal is XXX octal!
entering test_backtrace 5
entering test_backtrace 4
entering test_backtrace 3
entering test_backtrace 2
entering test_backtrace 1
entering test_backtrace 0
leaving test_backtrace 0
leaving test_backtrace 1
leaving test_backtrace 2
leaving test_backtrace 3
leaving test_backtrace 4
leaving test_backtrace 5
Welcome to the JOS kernel monitor!
Type 'help' for a list of commands.
K> help
help - Display this list of commands
kerninfo - Display information about the kernel
K> kerninfo
Special kernel symbols:
  _start                  0010000c (phys)
  entry  f010000c (virt)  0010000c (phys)
  etext  f01019e9 (virt)  001019e9 (phys)
  edata  f0113060 (virt)  00113060 (phys)
  end    f01136a0 (virt)  001136a0 (phys)
Kernel executable memory footprint: 78KB
```

若上面都能成功显示，说明配置成功

QEMU 的调试工具 gdb

```
cyl@ubuntu:~/6.828/lab$ make gdb
gdb -n -x .gdbinit
```

```
GNU gdb (Ubuntu 8.1.1-0ubuntu1) 8.1.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word".
+ target remote localhost:26000
warning: No executable has been specified and target does not support
determining executable automatically.  Try using the "file" command.
warning: A handler for the OS ABI "GNU/Linux" is not built into this configuration
of GDB.  Attempting to continue with the default i8086 settings.

The target architecture is assumed to be i8086
[f000:fff0]    0xffff0: ljmp   $0xf000,$0xe05b # GDB对要执行的第一条指令的反汇编
0x0000fff0 in ?? ()
+ symbol-file obj/kern/kernel
```

其中,`[f000:fff0] 0xffff0: ljmp $0xf000,$0xe05b` 是 GDB 对要执行的第一条指令的反汇编。从这个输出可以得出一些结论:

- IBM PC 从物理地址`0x000ffff0`开始执行,它位于为 ROM BIOS 保留的 64KB 区域的最顶部
- PC 从 `CS = 0xf000` 和 `IP = 0xfff0` 开始执行
- 第一条要执行的指令是jmp指令，跳转到分段地址CS=0xf000和IP=0xe05b。

QEMU 为什么会这样开始？这就是英特尔设计 8088 处理器的方式，IBM 在其原始 PC 中使用了该处理器,因为 PC 中的 BIOS 被“硬连线”到物理地址范围 0x000f0000-0x000fffff，这种设计确保 BIOS 在开机或任何系统重新启动后总是首先获得对机器的控制 - 这一点至关重要，因为在开机时，机器 RAM 中的任何地方都没有处理器可以执行的其他软件。在处理器复位时，（模拟的）处理器进入 Real 模式并将 CS 设置为 0xf000 并将 IP 设置为 0xfff0，以便从该 (CS:IP) 段地址开始执行。分段地址0xf000:fff0如何变成物理地址?

公式为 `physical address = 16 * segment + offset`,

## Part 2: The Boot Loader

## Part 3: The Kernel

## 参考链接

- https://pdos.csail.mit.edu/6.828/2018/labs/lab1/
- https://www.dingmos.com/index.php/archives/3/#cl-1
- https://knowledgehive.github.io/6.828/lab1-Booting a PC.html#part-1-pc-bootstrap