中山大学移动信息工程学院本科生实验报告

（2016 学年秋季学期）

课程名称：Operating System　　任课教师：饶洋辉　　批改人(此处为 TA 填写)：

年级+班级	1403	专业（方向）	移动信息工程
学号	14353435	姓名	朱恩东
电话	15919154005	Email	562870140@qq.com
开始日期	2016.3.21	完成日期	2016.3.21

1. 实验目的

安装今后实验必需的操作系统 linux，虚拟机软件 bochs，以及一个简单的操作系统 pintos 的环境配置。

2. 实验过程

（1）安装虚拟机 VMware，然后读入 Ubuntu 的镜像文件，完成 Ubuntu 的安装。

（2）为了后续环境配置的顺利进行，需要先设置 Ubuntu 的网络连接，我使用的方法是把在 windows 下的本地连接共享给虚拟机系统的虚拟网卡使用，这样设置起来比较方便，而且只要 windows 下可以上网，即可保证 linux 下网络可以使用。

（3）连上网后，我们需要安装编译所需的一些软件和库，例如：g++，gcc，perl 等等，这里可以直接安装 build-essential 包，即可安装编译必需的软件和库。然后我们还需要单独安装一个 autoconf 软件，以保证后续的指令可以使用。

（4）完成必要的前期准备之后，我们就可以开始配置 bochs，我们只需在终端下先进入 bochs 的文件夹，然后初始化配置文件，再用 sudo make install 指令即可完成编译安装。

（5）完成 bochs 的配置之后，我们就可以开始配置 pintos。首先我们需要修改 pintos 文件夹下的 .profile 文件，在其后加上一句 export path =...语句以修改路径。然后用 source .profile 语句使 profile 文件生效。

（6）接着，我们进入 pintos 下的 utils 文件夹，先手动地修改其中的 makefile 文件，更改 LDFLAGS = -lm 为 LDLIBS = -lm。然后使用 make 指令编译 utils 文件夹下的工程文件。至此，我们就完成了 pintos 的配置工作。

（7）测试 pintos 内核，先进入 utils 编译产生的 build 文件夹，分别执行 make check 和 pintos run alarm-multiple 指令完成两个测试。

3. 配置结果

```
zed@14353435_zed: ~/pintos/src/threads/build
pass tests/threads/alarm-negative
FAIL tests/threads/priority-change
FAIL tests/threads/priority-donate-one
FAIL tests/threads/priority-donate-multiple
FAIL tests/threads/priority-donate-multiple2
FAIL tests/threads/priority-donate-nest
FAIL tests/threads/priority-donate-sema
FAIL tests/threads/priority-donate-lower
FAIL tests/threads/priority-fifo
FAIL tests/threads/priority-preempt
FAIL tests/threads/priority-sema
FAIL tests/threads/priority-condvar
FAIL tests/threads/priority-donate-chain
FAIL tests/threads/mlfqs-load-1
FAIL tests/threads/mlfqs-load-60
FAIL tests/threads/mlfqs-load-avg
FAIL tests/threads/mlfqs-recent-1
pass tests/threads/mlfqs-fair-2
pass tests/threads/mlfqs-fair-20
FAIL tests/threads/mlfqs-nice-2
FAIL tests/threads/mlfqs-nice-10
FAIL tests/threads/mlfqs-block
20 of 27 tests failed.
make: *** [check] Error 1
```

以上是测试一结果，可以看出我们 fail 了 27 个测试中的 20 个，与预期结果相同。

```
zed@14353435_zed: ~/pintos/src/threads/build
(alarm-multiple) thread 3: duration=40, iteration=2, product=80
(alarm-multiple) thread 1: duration=20, iteration=4, product=80
(alarm-multiple) thread 2: duration=30, iteration=3, product=90
(alarm-multiple) thread 1: duration=20, iteration=5, product=100
(alarm-multiple) thread 4: duration=50, iteration=2, product=100
(alarm-multiple) thread 1: duration=20, iteration=6, product=120
(alarm-multiple) thread 2: duration=30, iteration=4, product=120
(alarm-multiple) thread 3: duration=40, iteration=3, product=120
(alarm-multiple) thread 1: duration=20, iteration=7, product=140
(alarm-multiple) thread 2: duration=30, iteration=5, product=150
(alarm-multiple) thread 4: duration=50, iteration=3, product=150
(alarm-multiple) thread 3: duration=40, iteration=4, product=160
(alarm-multiple) thread 2: duration=30, iteration=6, product=180
(alarm-multiple) thread 4: duration=50, iteration=4, product=200
(alarm-multiple) thread 3: duration=40, iteration=5, product=200
(alarm-multiple) thread 2: duration=30, iteration=7, product=210
(alarm-multiple) thread 3: duration=40, iteration=6, product=240
(alarm-multiple) thread 4: duration=50, iteration=5, product=250
(alarm-multiple) thread 3: duration=40, iteration=7, product=280
(alarm-multiple) thread 4: duration=50, iteration=6, product=300
(alarm-multiple) thread 4: duration=50, iteration=7, product=350
(alarm-multiple) end
Execution of 'alarm-multiple' complete.
```

以上是测试二结果，与标准结果完全相同，说明通过测试二。综上可知，我们环境配置成功。

4. 实验感想

本次实验遇到的第一个问题是在安装 Ubuntu 的时候，在载入系统镜像后就被提示无法开启 Intel VT-x 而导致安装失败。百度之后了解了解决方法，即进入 BIOS 系统，将 intel virtual 选项开启即可。Intel virtual 是 intel 芯片用于支持虚拟机的技术，这种技术让可以让一个 CPU 工作起来就像多个 CPU 并行运行，从而使得在一部电脑内同时运行多个操作系统成为可能。

接下来遇到了换源上的问题，在 Ubuntu 下默认的源是美国的，导致我一开始下载 build-essential 等包的时候都失败了。后来我用了设置中的查找最合适的源，这次找的是英国的源，然而还是下载失败。。。最后只有换回了国内的 aliyun 的源才解决了问题。

换完源后，安装 build-essential 包依旧失败，根据报错信息发现是由于包之间的依赖性问题。因为觉得一个个去下太麻烦了。所以使用了 aptitude 包来解决这个问题，aptitude 与 apt-get 功能相类似，但是 aptitude 更加智能，当遇到安装报错时，它会尝试提供解决方案。所以我使用 aptitude 来安装 build-essential 时，它提供解决这种库之间依赖性问题的解决方案，按照他提供的解决方案，才得以成功按照本次实验中所必需的包。

在用命令行进入文件夹编译时，犯了一个很低级的错误，因为解压出来的文件夹有两层，而我就直接把其复制到 Ubuntu 下了。所以直接使用教程里的命令是不可以的，这也说明了平时一定要细心一点，很有可能就因为这样的一个小细节而浪费掉很多时间。