

实验报告：安装 pintos

课程名称：操作系统实践

年级：2023 级

上机实践成绩：

指导教师：张民

姓名：张建夫

上机实践名称：安装 pintos

学号 10235101477

上机实践日期 2024/9/25

上机实践编号：

组号：

上机实践时间：18~19 点

1. 硬件环境：

调用 dxdiag 工具：

Operating System: Windows 11 家庭中文版 64-bit (10.0, Build 22621)
(22621.ni_release.220506-1250)

Language: Chinese (Simplified) (Regional Setting: Chinese (Simplified))

System Manufacturer: HP

System Model: HP Pavilion Aero Laptop 13-be2xxx

BIOS: F.13 (type: UEFI)

Processor: AMD Ryzen 5 7535U with Radeon Graphics (12 CPUs), ~2.9GHz

Memory: 16384MB RAM

Available OS Memory: 15574MB RAM

Page File: 27604MB used, 5685MB available

Windows Dir: C:\WINDOWS

DirectX Version: DirectX 12

DX Setup Parameters: Not found

User DPI Setting: 144 DPI (150 percent)

System DPI Setting: 192 DPI (200 percent)

DWM DPI Scaling: UnKnown

Miracast: Available, with HDCP

Microsoft Graphics Hybrid: Not Supported

2. 安装过程：

1. 先安装 docker



安装完成后，运行如下命令：

`docker run -it pkuflyingpig/pintos bash`

2. 进入容器的 bash 希尔程序

在运行上述的命令后，docker 会创建一个容器（第一次进入），该镜像会运行在该容器内，且自动进入该容器的 bash shell 中，为查看容器是否创建，打开另一个 powershell，利用 `docker ps -a` 查看所有的容器：

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
c83d539f9fb3	pkuflyingpig/pintos	"bash"	11 days ago	Up 8 minutes		sleepy_visvesvaraya
69d2794469c5	cs162/pintospace:latest	"/entrypoint.sh"	2 weeks ago	Up 3 hours	0.0.0.0:16222->22/tcp	cs162-workspace-cs162-1

其中第一个为我们的镜像，可以看到它的 status 是 up 的。

3. 拉取骨架代码并启动 pintos

回到镜像的 bash shell，由于该镜像自带 git，可以直接从 pku 的远程仓库进行拉取，可以用该命令：`git clone https://github.com/PKU-OS/pintos.git`（原本我想用 ssh 进行拉取的，但是这个镜像下不了 openssh /(ㄟ o ㄟ)/呜呜~~）

接着利用 ls 发现只有 toolchain 一个文件夹，一直向下进入 pintos 源码的 threads 文件夹，键入 make 命令构建 pintos，pintos 根据 makefile 文件在 build 中建立，进入 build 文件夹，运行 `pintos --`，运行截图如下：（由于之前已经 make 过，make 这里显示的是 nothing to be done）

```

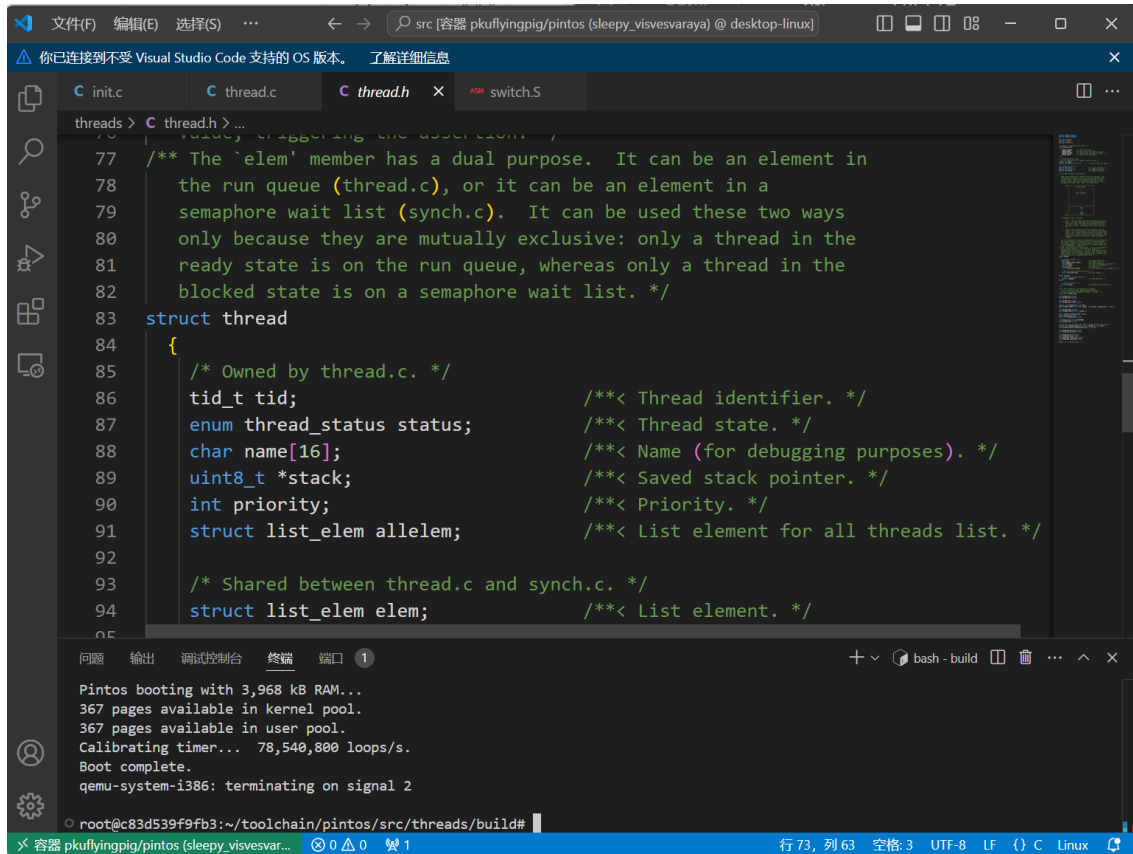
root@c83d539f9fb3:~/toolchain/pintos/src/threads# make
cd build && make all
make[1]: Entering directory '/home/PKUOS/toolchain/pintos/src/threads/build'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/home/PKUOS/toolchain/pintos/src/threads/build'
root@c83d539f9fb3:~/toolchain/pintos/src/threads# cd build
root@c83d539f9fb3:~/toolchain/pintos/src/threads/build# pintos --
perl: warning: Setting locale failed.
perl: warning: Please check that your locale settings:
    LANGUAGE = (unset),
    LC_ALL = (unset),
    LANG = "en_US.UTF-8"
are supported and installed on your system.
perl: warning: Falling back to the standard locale ("C").
qemu-system-i386 -device isa-debug-exit -drive format=raw,media=disk,index=0,file=/tmp/3VirZnsx26.dsk -m 4 -net none -nographic -monitor null
Pintos hda1
Loading.....
Kernel command line:
Pintos booting with 3,968 kB RAM...
367 pages available in kernel pool.
367 pages available in user pool.
Calibrating timer... 78,540,800 loops/s.
Boot complete.

```

至此，pintos 环境安装完毕，利用 Ctrl+c 即可退出 pintos。

3. 遇到的问题：

由于之前用 git 和 docker 建立过类似环境，环境安装过程中没有遇到任何问题，唯一的一个不算是问题的问题是如何利用 vscode 编辑 pintos 代码，由于之前我都是用 ssh 连接远程环境进行 linux 环境下的编程，而这个镜像安装不了 openssh，这次要用到 vscode 的 docker 插件实现 vscode 能够附加到运行中的容器，这样来连接该镜像编辑代码，实现结果：



```

threads > C thread.h ...
77 /** The 'elem' member has a dual purpose. It can be an element in
78 the run queue (thread.c), or it can be an element in a
79 semaphore wait list (synch.c). It can be used these two ways
80 only because they are mutually exclusive: only a thread in the
81 ready state is on the run queue, whereas only a thread in the
82 blocked state is on a semaphore wait list. */
83 struct thread
84 {
85     /* Owned by thread.c. */
86     tid_t tid;                /**< Thread identifier. */
87     enum thread_status status; /**< Thread state. */
88     char name[16];            /**< Name (for debugging purposes). */
89     uint8_t *stack;           /**< Saved stack pointer. */
90     int priority;              /**< Priority. */
91     struct list_elem allelem;  /**< List element for all threads list. */
92
93     /* Shared between thread.c and synch.c. */
94     struct list_elem elem;     /**< List element. */
95 }

```

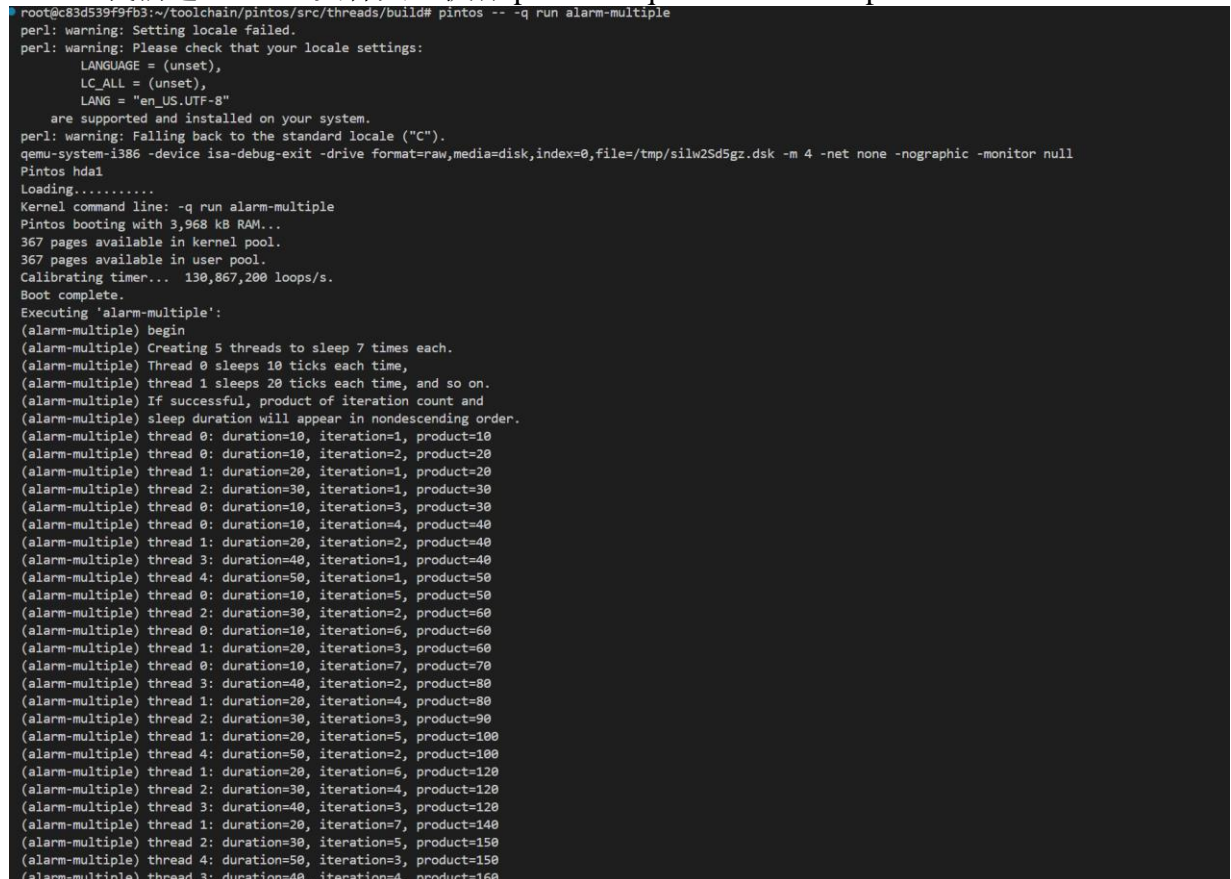
```

Pintos booting with 3,968 kB RAM...
367 pages available in kernel pool.
367 pages available in user pool.
Calibrating timer... 78,540,800 loops/s.
Boot complete.
qemu-system-i386: terminating on signal 2

```

4. 完成结果:

我们进入 build 文件夹，执行 `pintos -- -q run alarm-multiple`:



```

root@cs3d539f9fb3:~/toolchain/pintos/src/threads/build# pintos -- -q run alarm-multiple
perl: warning: Setting locale failed.
perl: warning: Please check that your locale settings:
    LANGUAGE = (unset),
    LC_ALL = (unset),
    LANG = "en_US.UTF-8"
are supported and installed on your system.
perl: warning: Falling back to the standard locale ("C").
qemu-system-i386 -device isa-debug-exit -drive format=raw,media=disk,index=0,file=/tmp/silw2Sd5gz.dsk -m 4 -net none -nographic -monitor null
Pintos hda1
Loading.....
Kernel command line: -q run alarm-multiple
Pintos booting with 3,968 kB RAM...
367 pages available in kernel pool.
367 pages available in user pool.
Calibrating timer... 130,867,200 loops/s.
Boot complete.
Executing 'alarm-multiple':
(alarm-multiple) begin
(alarm-multiple) Creating 5 threads to sleep 7 times each.
(alarm-multiple) Thread 0 sleeps 10 ticks each time,
(alarm-multiple) thread 1 sleeps 20 ticks each time, and so on.
(alarm-multiple) If successful, product of iteration count and
(alarm-multiple) sleep duration will appear in nondescending order.
(alarm-multiple) thread 0: duration=10, iteration=1, product=10
(alarm-multiple) thread 0: duration=10, iteration=2, product=20
(alarm-multiple) thread 1: duration=20, iteration=1, product=20
(alarm-multiple) thread 2: duration=30, iteration=1, product=30
(alarm-multiple) thread 0: duration=10, iteration=3, product=30
(alarm-multiple) thread 0: duration=10, iteration=4, product=40
(alarm-multiple) thread 1: duration=20, iteration=2, product=40
(alarm-multiple) thread 3: duration=40, iteration=1, product=40
(alarm-multiple) thread 4: duration=50, iteration=1, product=50
(alarm-multiple) thread 0: duration=10, iteration=5, product=50
(alarm-multiple) thread 2: duration=30, iteration=2, product=60
(alarm-multiple) thread 0: duration=10, iteration=6, product=60
(alarm-multiple) thread 1: duration=20, iteration=3, product=60
(alarm-multiple) thread 0: duration=10, iteration=7, product=70
(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 3: duration=40, iteration=5, product=200
(alarm-multiple) thread 4: duration=50, iteration=4, 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.
Timer: 585 ticks
Thread: 0 idle ticks, 585 kernel ticks, 0 user ticks
Console: 2954 characters output
Keyboard: 0 keys pressed
Powering off...
```

以上为执行结果。

5. 报告总结:

由于之前配过相近环境，这次配环境没有遇到太大困难，还有一些意外收获，如对 vscode docker 插件的一些功能的了解和使用（我发现我之前配的远程环境也可以用附加容器的方式打开），同时，也加深了对 docker 的使用经验。总而言之，这是一次很好的提升配环境能力的实验。