CS4375-13948 Fall 2023
Geoff Waller
Gwaller@miners.utep.edu

Homework Report
Submitted on 9/16/2023

HW 1: Introduction to xv6

Task 1. Boot xv6 and explore utilities

My Linux virtual machine setup is : VMware Fusion 12.1.2 running on macOS 11.7.8.

Result of xv6 build:

```
gwaller@gwaller-virtual-machine:~/CS4375.git/CS4375$ ls
fs.img  kernel  LICENSE  Makefile  mkfs  README  user
gwaller@gwaller-virtual-machine:~/CS4375.git/CS4375$ make qemu
qemu-system-riscv64 -machine virt -bios none -kernel kernel/kernel -m 128M -smp 3 -nographic -drive file=fs.img,if=none,format=raw,i
d=x0 -device virtio-blk-device,drive=x0,bus=virtio-mmio-bus.0

xv6 kernel is booting

hart 1 starting
hart 2 starting
init: starting sh
$
```

Result of ls command:

```
Ubuntu 64-bit 22.04.1
Activities   Terminal                    Sep 16 13:59
gwaller@gwaller-virtual-machine: ~/CS4375.git/CS4375
stressfs        2 14 23576
usertests       2 15 150312
grind           2 16 37104
wc              2 17 24696
zombie          2 18 21856
sleep           2 19 22280
ps              2 20 23400
pstree          2 21 24368
pstest          2 22 23456
console         3 23 0
$ ls
.               1 1 1024
..              1 1 1024
README          2 2 2226
cat             2 3 23720
echo            2 4 22568
forktest        2 5 13288
grep            2 6 26872
init            2 7 23368
kill            2 8 22480
ln              2 9 22352
ls              2 10 25904
mkdir           2 11 22616
rm              2 12 22600
sh              2 13 40624
stressfs        2 14 23576
usertests       2 15 150312
grind           2 16 37104
wc              2 17 24696
zombie          2 18 21856
sleep           2 19 22280
ps              2 20 23400
pstree          2 21 24368
pstest          2 22 23456
console         3 23 0
$
```

Result of Cat:

```
$ cat README
xv6 is a re-implementation of Dennis Ritchie's and Ken Thompson's Unix
Version 6 (v6).  xv6 loosely follows the structure and style of v6,
but is implemented for a modern RISC-V multiprocessor using ANSI C.

ACKNOWLEDGMENTS

xv6 is inspired by John Lions's Commentary on UNIX 6th Edition (Peer
to Peer Communications; ISBN: 1-57398-013-7; 1st edition (June 14,
2000)). See also https://pdos.csail.mit.edu/6.828/, which
provides pointers to on-line resources for v6.

The following people have made contributions: Russ Cox (context switching,
locking), Cliff Frey (MP), Xiao Yu (MP), Nickolai Zeldovich, and Austin
Clements.
```

The cat command does a Boolean check for one argument, iterates through file names and writes to the terminal.

Result of Echo:

```
$ echo "Hello World"
"Hello World"
$
```

The echo command iterates through the argument provided as char input and writes to the terminal the corresponding characters.

Result of mkdir:

```
$ mkdir temp
$ ls
ps             2 20 23400
pstree         2 21 24368
pstest         2 22 23456
console        3 23 0
temp           1 24 32
$
```

The mkdir command takes user input for the name of a new directory and then outputs the name of the directory, provided the argument is not empty, meaning the array of characters entered is not zero.

Difficulties I had during this process involved the setup. I have issues primarily with the setup of GitHub. I have to mirror the repository several times in order for it to work properly. I also had to look through out YouTube in order to find instructions that allowed me to install Qemu correctly, as the original ones provided did not work for me correctly.

Task 2. Implement the uptime utility

In this assignment, I learned how to implement created files and run them on Qemu. I learned I could create code that called upon different system calls. In this assignment, I created code that called systems time ticks and printed those ticks as output.

The difficulties I ran into can with how to approach this task, as I have done this before. To overcome this I had to spend extra time navigating through the directories, particularly the kernel and user directory to understand what the code was doing and where is should be placed. The code itself was not difficult to implement.

Example of code:

```
gwaller@gwaller-virtual-machine:~/CS4375.git/CS4375/user$ cat uptime.c
#include "kernel/types.h"
#include "kernel/stat.h"
#include "user/user.h"

int main(){
        int ticks = sys_uptime();

        if (ticks < 0){

                exit(0);
        }

        printf("up %d clock ticks\n", ticks);

        exit(0);
}
```

Code running in Qemu:

```
echo            2 4 22568
forktest        2 5 13288
grep            2 6 26872
init            2 7 23368
kill            2 8 22480
ln              2 9 22352
ls              2 10 25904
mkdir           2 11 22616
rm              2 12 22600
sh              2 13 40624
stressfs        2 14 23576
usertests       2 15 150312
grind           2 16 37104
wc              2 17 24696
zombie          2 18 21856
sleep           2 19 22280
ps              2 20 23400
pstree          2 21 24368
pstest          2 22 23456
uptime          2 23 22072
console         3 24 0
$ uptime
up 72 clock ticks
$
```

Git url:
https://github.com/Gwaller915/CS4375