Homework 3

Academic Honesty

Aside from the narrow exception for collaboration on homework, all work submitted in this course must be your own. Cheating and plagiarism will not be tolerated. If you have any questions about a specific case, please ask me. We will be checking for this!

NYU Poly's Policy on Academic Misconduct: http://engineering.nyu.edu/academics/code-of-conduct/academic-misconduct

Homework Notes :

General Notes:

- Read the assignment carefully, including what files to include.
- Don't assume limitations unless they are explicitly stated.
- Treat provided examples as just that, not exhaustive list of cases that should work.
- **TEST** your solutions, make sure they work. It's obvious when you didn't test the code.

Prerequisites:

Before starting on this assignment, make sure you have gone through the guide to setting your system up for development (available in the Resources section in NYU classes). You will need the following for this assignment:

1. git clone a **fresh** xv6 repository

```
$ git clone https://github.com/gussand/xv6-public.git
Cloning into 'xv6-public'...
remote: Counting objects: 4475, done.
remote: Compressing objects: 100% (2679/2679), done.
remote: Total 4475 (delta 1792), reused 4475 (delta 1792), pack-reused 0
Receiving objects: 100% (4475/4475), 11.66 MiB | 954.00 KiB/s, done.
Resolving deltas: 100% (1792/1792), done.
Checking connectivity... done.
```

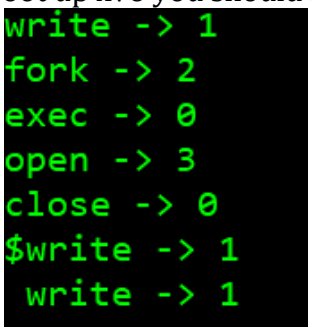
You can follow the same instructions as from **Homework 1** for setting up your system.

Homework 3: xv6 System Calls

Part 1: Call Tracing

Modify the xv6 kernel to print out the name of the system call and its return value for each system call invocation. You do not have to print the system call arguments.

Upon completion, when you first boot up xv6 you should see output similar to:



Note that the output of the shell and the system call trace are intermixed (e.g. the $write -> 1). This is due to the fact that the shell utilized the write system call to print its output.

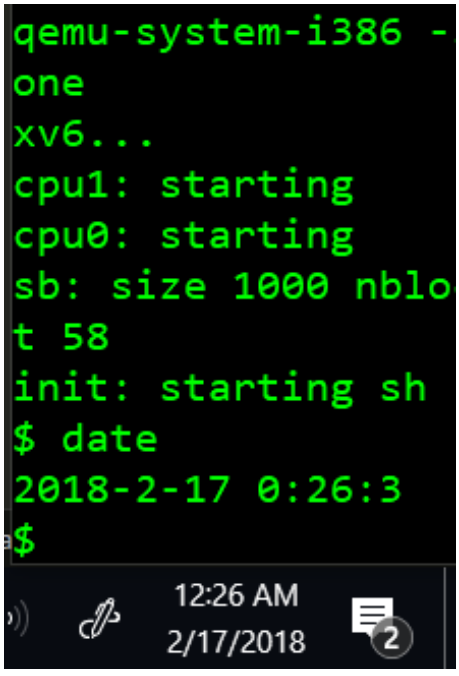
Hints:

1. You will primarily be working in syscall.c
2. Think about who is responsible for keep track of the return codes for the system calls.

Part 2: Date System Call

Now add a new system call to xv6 that will get the current UTC time and return the **EST (EASTERN STANDARD TIME)** to the user program. You may use the cmosttime( ) help function to read the real time clock. The *struct rctdate* struct is defined in the date.h file.

You will modify the give date.c file to create a user-level program call for the new data system call. When done, typing *date* to the xv6 prompt should print the current EST time.



You may either have the system call return the UTC time, then convert to EST time or you may have the system call return the EST time.

Hints:

1. When making your date system call, you should focus on mimicking all the pieces of code that are specific for some existing system call, for example the "uptime" system call. Or you can refer to the slides for the steps of how to add a system call.
2. Use grep for the uptime in all the source files using:
   grep -n uptime *.[chS]

Submit

You will use git to create a patch that contains your changes. First, tell git who you are:

```
$ git config --global user.email "you@example.com"
$ git config --global user.name "Your Name"
```

Then, commit your changes:

```
$ git commit --all --message="Implement data system call"
    [hw3 94b0cf7] Implement date system call
     7 files changed, 60 insertions(+), 7 deletions(-)
```

(Note: if you added any new files, you will also have to use

```
`git add <filename>` before you run `git commit`.)
```

Finally, create a patch file. The command takes an argument that specifies what code we're comparing against to make the patch; in this case I have created a git *tag* that refers to the unmodified Homework 3 called `hw3.unmodified`, so you should run:

```
$ git format-patch hw3.unmodified
    0001- Implement-date-system-call.patch
```

The command creates a file, `0001-Implement-date-system-call.patch`, containing the changes you've made. Submit this file on NYU Classes.

Submission Note

Don't try to edit the patch file after creating it. Doing so will most likely corrupt the patch file and make it impossible to apply. Instead, change the original file, commit your changes, and run `git format-patch` again:

```
$ git commit --all --message="Description of your change here"
    [hw3 7cc4977] Description of your change here
     1 file changed, 1 insertion(+), 1 deletion(-)
$ git format-patch hw3.unmodified
    0001-Implement-date-system-call.patch
    0002-Description-of-your-change-here.patch
```

Then submit *both* patch files.