COMP604 – Operating Systems

Unofficial course manual

Written by Zach Barrett Last updated: 2024-12-07

Table of contents

1. Introduction

This document is a collection of my notes and knowledge related to the class "Operating Systems" at AUT, written for someone taking this class. While largely intended as a way of writting down everything I know for my own benefit, my hope is that it will be of use for you as well.

1.1. Additional resources

This class recommends the use of the OSTEP (Operating Systems, Three Easy Pieces) textbook. It can be found here: OSTEP This is an excellent resource, and I highly recommend using this.

2. Class Overview

2.1. Assessment Overview
There are three assignments in this class, with assignment two being the longest and most difficult.

Assessment Item	Weighting
Assignment One	30%
Assignment Two	40%
Assignment Three	30%

2.1.1. Assignment One
This assignment covered shell scripting, C programming, and a simple xv6 system call. Easiest by far, do not use your extension here. When I took it, this assignment had 3 questions.

2.1.2. Assignment Two
This assignment covered scheduling algorithms, processes, and more advanced system calls. This is the longest assignment, using your extension here is fine. When I took it, this assignment had 5 questions. All programming questions were done in xv6.

2.1.3. Assignment Three
This assignment covered semaphores, inodes, RAID, storage in xv6. This assignment was harder than assignment one but easier than assignment two. If you can save your extension for this assignment then great, but dont feel bad about spending it on assignment two.

2.2. Software tools
The later portion of this course makes use of the XV6 operating system. Because XV6 runs on a different architecture than most computers, it can only be run using a virtual machine. Thankfully, a free and easy to use virtual machine exists called QEMU. The next sections will show you how to install this software stack.

If you are able to get access to a linux or mac computer, doing so will make your life much easier as you will be able to run QEMU directly and not through another virtual machine (Unlike windows, where you will need to use WSL).

If you have to use WSL, I highly recommend installing Visual Studio Code and the WSL extension for it, allowing you to edit the xv6 files directly through the editor instead of having to use either copy them in/out or having to use a command line editor like nano or vim. (Although I also recommend you learn how to use vim keybindings.)

This information is a combination of the "Software setup" section on canvas and my own experiences.

2.2.1. Mac setup

1. To compile the software tools necessary to run XV6, you must first install xcode.

   ```
   xcode-select --install
   ```

   This will take some time.

2. Next, ensure the Homebrew package manager is installed. If it is not, install with the following command:

   ```
   /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
   ```

   (If you dont already use homebrew, you should. Its extremely useful.)

3. Then, use Homebrew to install the risc-v compiler toolchain:

   ```
   brew tap riscv/riscv
   brew install riscv-tools
   ```

   With this step, sometimes people (me, I had this issue) have issues with compiling the riscv toolchain. If that occurs try changing your OSX version to OSX Sonoma, as there is a precompiled binary available for that version. Once thats done try running those commands again, Homebrew will handle it for you.

4. Next you must add the toolchain directory to your "path", which is where your computer looks for programs if they are not in the current directory. This can be done using any text editor. To do so, you must open a file called .zshrc (This is the configuration file for the default shell on mac, if you have changed the shell I assume you know which **other** file to change instead.)

   Then append one of the following lines to the bottom of the file depending on your situation

   ```
   # Use the below line on M series macs
   export PATH=$PATH:/usr/local/opt/homebrew/Cellar/riscv-gnu-toolchain/master/bin

   # Use this one for intel series macs
   export PATH=$PATH:/usr/local/Cellar/riscv-gnu-toolchain/master/bin

   # If you are on an M series mac and you get an error message at the end of
   # setup saying the computer cant find the compiler toolchain or something,
   # replace the line you added with this one:
   export PATH=$PATH:/opt/homebrew/Cellar/riscv-gnu-toolchain/main.reinstall/bin
   ```

5. Finally, install QEMU.

   ```
   brew install qemu
   ```

   If you have issues with this step ensure that the path was set correctly.

2.2.2. Ubuntu setup

Getting this running is very easy on ubuntu. Note that the university computers with WSL enabled also have ubuntu installed.

To setup QEMU on ubuntu, simply run the following commands:

```
sudo apt update && sudo apt upgrade
sudo apt install build-essential gdb-multiarch qemu-system-misc gcc-riscv64-linux-gnu binutils-riscv64-linux-gnu
```

If that throws an error, try separating all of the packages into seperate lines in a text file (Lets call it packs.txt), and running the following command:

```
cat packs.txt | xargs sudo apt install
```

(Learn about the xargs command by the way, very useful!)

2.2.3. Windows setup

You will need to use WSL (Windows Subsystem for Linux) as windows cannot run QEMU directly.

1. Start by opening a command prompt and running the following command:

   ```
   wsl --install -d ubuntu-22.04
   ```

   This will install WSL and ubuntu.

2. Reboot your computer, open bios and ensure that hardware virtualisation is enabled. (look up how to do this, it varies by motherboard manufacturer)

3. Once your computer has rebooted, open a command prompt and run:

   ```
   ubuntu
   ```

   This will open an ubuntu command prompt. From here setup proceeds the same as in the ubuntu setup section.

   Once you have completed that, I recommend setting up an editor capable of opening and editing files in WSL. This can be done using Visual Studio Code and an extension. Here is the link to that extension

3. What you need to know about C programming

4. XV6 for Dummies

4.1. General structure

4.2. System Calls

4.2.1. How to add a system call