**ECEN 449 - Lab Report**

**Lab Number:** 6

**Lab Title:** Intro to Character Device Driver Development

**Section Number:** 508

**Student's Name:** Samuel Fafel

**Date Due:** 04-13-2023

**TA:** Prajwal Holla

#### Purpose/Introduction:

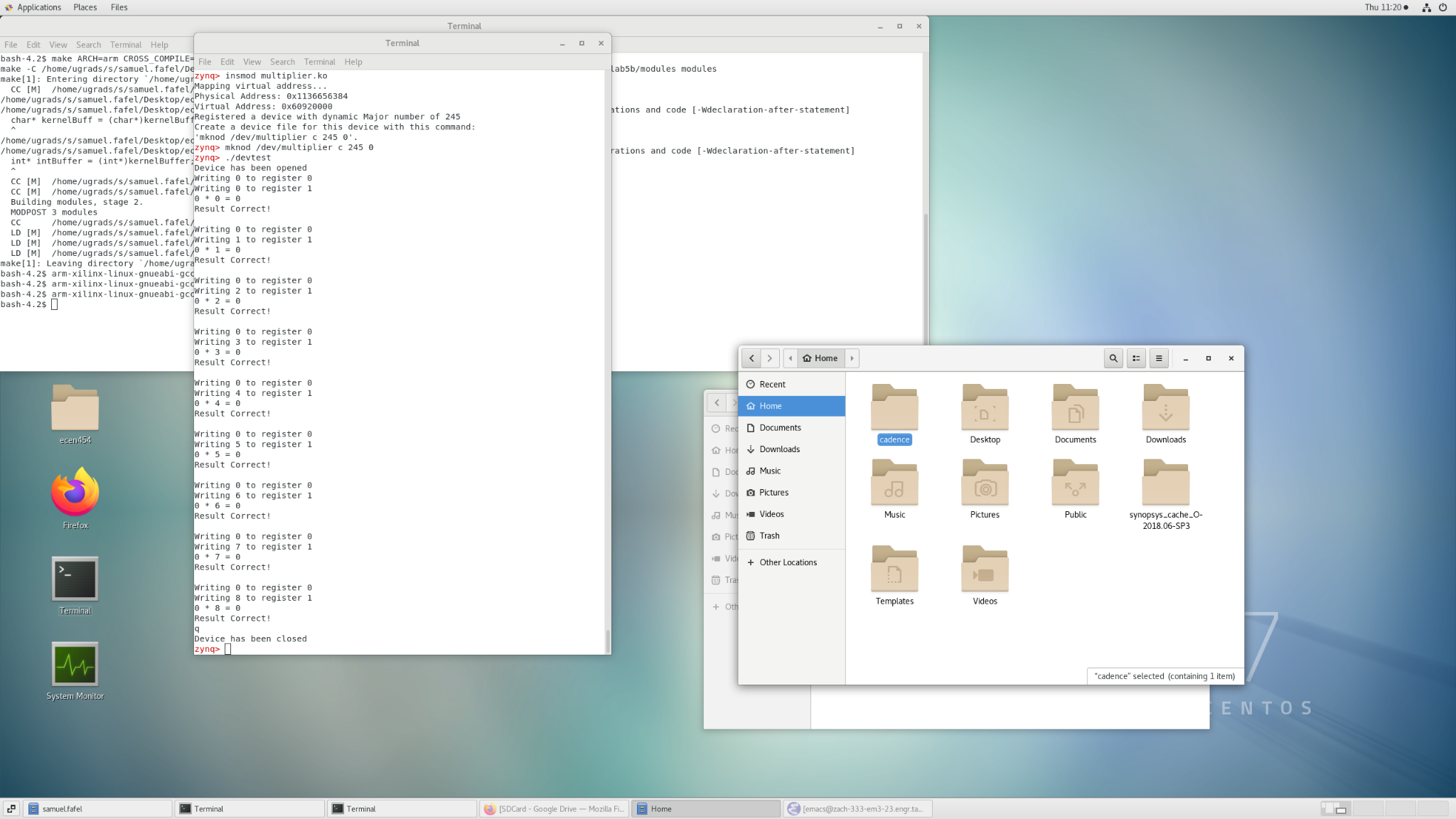
The goal of this lab is to create device drivers in and for our linux environment. This will result in the capabilities of the kernel being increased.

#### Procedure:

* Create multiplier.c according to the specifications in the lab manual, in reference to my\_chardev.c and my\_chardev\_mem.c and compile it to create multiply.ko.
* Create devtest.c to write to the registers of the multiply module. Compile it and move to the Zynq system.
* Start multiplier.ko and make the node for devtest to use. Run devtest and record output.

#### Results:

The longest part of this assignment was writing the multiplier.c file based on the others referenced in the lab manual, as doing so required time to parse and understand the reference code. However, after completing the multiplier.c, it now operates by defining multiple functions which are called on the open/close/read/write of the device/file itself. Screenshots of the commented code can be found in the appendix. After writing this file, the rest of the lab was relatively straightforward.



Picocom output of ‘devtest’ on multiplier

#### Conclusion:

This lab expands even further upon the previous labs in that we have created a functional character device driver, which allows the sharing of hardware devices used by the operating system.

#### Questions:

1. The ioremap command is required because we have to map the physical address of the module to the virtual address (used for convenience since we can predict the placement in virtual memory).
2. Since Lab 3 was not working with/through an operating system (linux), I expect the Lab 3 implementation to take less overall time.
3. Lab 3's implementation was directly for our multiplication module, but was not flexible in any way. By implementing it through Linux, we slow down the process (by a tiny bit) but also have an easier time reading/writing to the device.
4. First, we want to initialize everything we need for the device prior to registering it. In the same way, we want to unregister the device prior to changing or removing any configuration on it, since it may still be using resources that would interrupt the cleaning process if done out of order.

#### Appendix:

