**ECEN 449 - Lab Report**

**Lab Number:** 4

**Lab Title:** Linux boot-up on ZYBO Z7-10 board via SD Card

**Section Number:** 508

**Student's Name:** Samuel Fafel

**Date Due:** 03-09-2023

**TA:** Prajwal Holla

#### Purpose/Introduction:

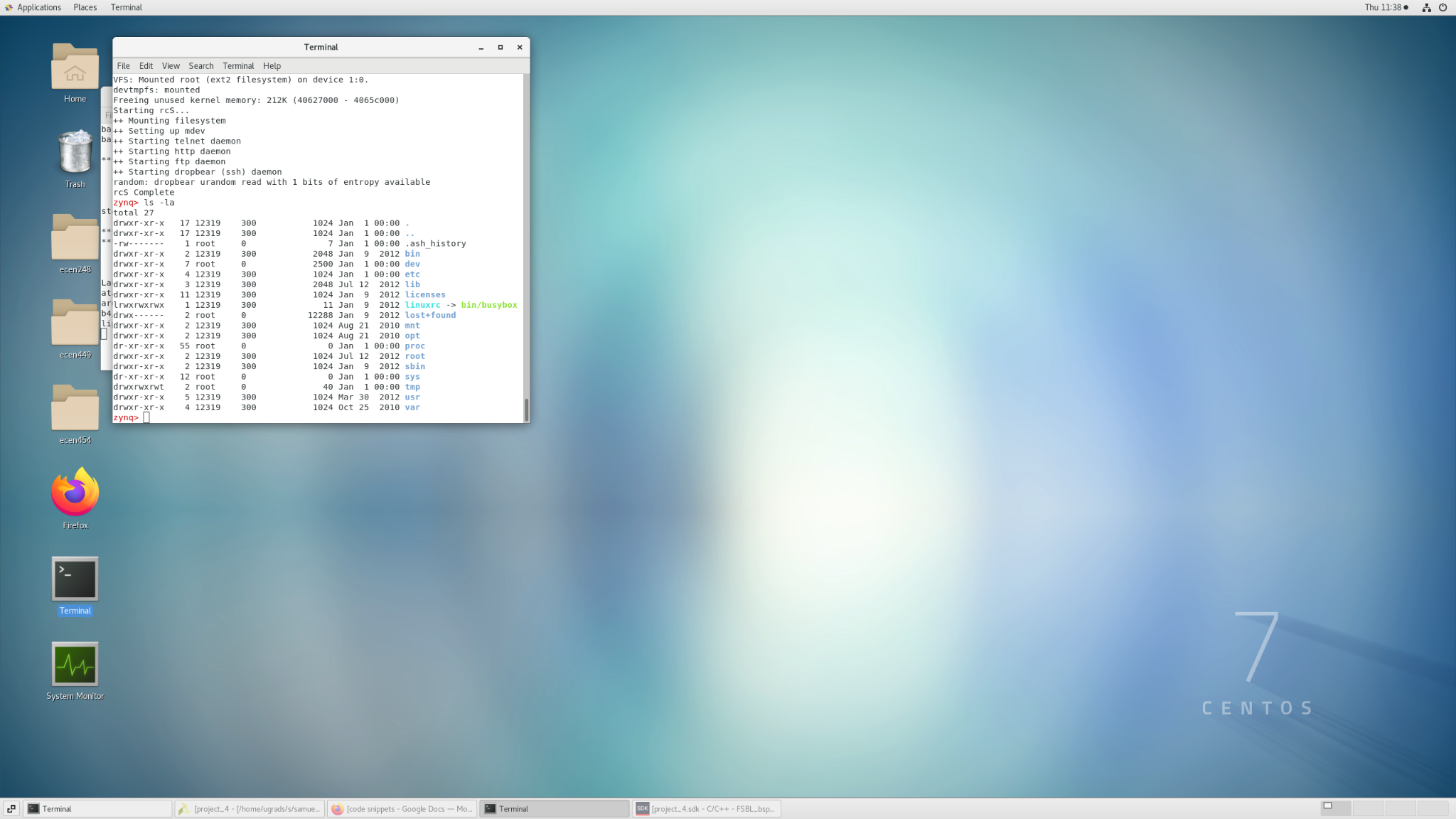
This lab resulted in the ability to run Linux on the ZYBO Z7-10 board, using "First Stage Boot Loader" (FSBL) and "Universal Boot Loader" (U-Boot) to create a Boot Image.

#### Procedure:

First, as in Lab 3, we created a block design and added our custom Multiply IP to it. Then we generated a bitstream for the block design and exported it to SDK. Parallel to this, we compiled U-Boot and generated an ELF file to be read by SDK. After it was generated, we generated the BOOT.bin file using the FSBL.elf, Bitstream, and U-Boot.elf files. Next, we sourced and untarred the Linux kernel and configured it to run on our ZYBO Z7-10 board. After configuration, we compiled it and generated a uImage file. Next, we edited the default device tree and added our custom Multiply code, then compiled it into the device tree blob (.dtb) file. Finally, we copied the ramdisk file from the shared directory and used it to create the uramdisk image.

Once all of these steps were accomplished, we placed the BOOT.bin, uImage, devicetree.dtb, and uramdisk image files all onto an SD card, inserted that SD Card into the board, booted up Picocom to monitor the output, and ran Linux on the board.

#### Results:



The result of this lab is a functional Linux kernel running on the ZYBO Z7-10 Board.

#### Conclusion:

This lab implements the first highly complex system we have encountered on the Zybo board. It shows the extreme flexibility of the FPGA and gives us the ability to do much more complex tasks in the OS itself.

#### Questions:

1. The local memory stores the cache of the CPU. This exists on a standard motherboard in the CPU in order to hold the BIOS (basic initialization operating system).
2. Using the command "ls -la" we can see a list of all files and subdirectories located in our current working directory. The first column shows the type and permissions of the listed item. A "d" at the very front indicates a directory while a "-" indicates a file. After the "d/-", there is an "r" and "w/-". The "r" indicates read permissions. The "w" indicates write permissions. If there is no "w" and only a "-", then write permissions are not granted. If we create a file in one of the writable directories, then restart the board, the file is lost. This is because the kernel we are running does not save updates on the board, but only possesses a set filesystem.
3. If we were to add another peripheral to our system, we would need to redo the steps in Part 1 (to reconfigure the block design) and recreate our BOOT.bin file with the updated bitstream.