Brandon Angeles

CS 4348.002

Dr. Greg Ozbirn

7 March 2022

**CS 4348.002 Project 1 Summary**

**1. Project Purpose**

The purpose of this project was to use two different processes that would communicate through an IPC, in this case a pipe. One process would act as a CPU and the other would act as a Memory for the CPU. We would need to be able to run the fetch and execute instruction cycle from the CPU based on the program saved into the Memory before execution. This will simulate a simple computer system as the CPU will be able to communicate with the Memory and get instructions and data with ease.

**2. Implementation**

I used Java because it is the language I am the most familiar with. First to implement was the pipe between the two processes. I used the CPU as the “parent” of the Memory. So, I ran the Memory inside of the CPU and piped it to memory. I had to first put the program into the memory before I could start the instruction cycle. To do this I piped the filename from the CPU to the memory and ran through the file and added it to the memory. Next, I ran through the program with a fetch and execute cycle from CPU, Memory, back to CPU and then ran the instruction given. I set up all the registers and as many instructions as we could without the implementation of interrupts yet. Next, I added the ability of the write function with a function added to the memory that can be flagged by the CPU by sending a “w”. This gave the memory its read and write functionality that it needed. I then set up the stack’s push and pop functions of them. This is important for a lot of reasons as it helped with the implementation of the interrupts. This was next, I set up the interrupts of timers and system calls making sure to disable interrupts, change from user to kernel mode, and to make sure that there was an easy return to the original PC and Stack Pointer. Lastly, I added checks to make sure that if the program tried to access data it wasn’t allowed to access then it would return an error message and end the program. That was everything that needed to be implemented and I just made sure both programs ended at the same time with no errors.

**3. My Personal Experience**

While doing this project my main obstacle was creating the pipe between the two processes. I kept trying to make pipes in each process and that was not the right answer. Once I got that it was very easy to implement the rest of project. I will say the other obstacle that stopped we was the timer and system call interrupts. This was difficult to implement but I finally figured it out once I understood what an OS should really do during a real system call. Making sure that it went to a system mode and afterwards went back to a user mode helped everything work in the end. It was a difficult project but in the end, I really understand the basic concept of pipes and how the basic instruction cycle works in the operating system.