CS 4348.501 Project 1 Summary

Abbas Khawaja

1. The purpose of this project is to implement a CPU and Memory system using a pipe to facilitate communication between the two components. The Memory should have 2000 entries with the first 1000 for user mode and the next 1000 for kernel mode. Additionally, there should be a stack pointer for both user and kernel mode. The CPU should be able to implement a set of 31 instructions provided by the assignment. The overall purpose of the project is to demonstrate an understanding of a CPU and Memory system working together as well as demonstrating an understanding of multiprocessing through fork command and inter-process communication through pipes.
2. The project was implemented in C Programming Language. My program uses a fork command which creates a child process. The child process is the Memory system and the parent process is the CPU system. Along with this, there are two pipes being used by the processes. The Memory to CPU Pipe is used by the Memory to write data to the CPU and used by the CPU to read data from Memory. The CPU to Memory Pipe is used by the CPU to write data to Memory and used by Memory to read data from the CPU. When the program executes, the sample file from command line is loaded into memory and places each instruction in its proper register. After loading the sample file, the CPU executes the instructions from the sample file. This is done by having the CPU write the PC using the pipe to memory and the Memory returning the value at the PC. This value is the instruction the CPU executes using a switch statement to determine what lines of code need to be executed. The CPU may do additionally writes to the pipe which the memory will always read for. If the CPU wanted to write data inside memory, the CPU sends -1 which is essentially a Write Flag that allows the memory to know that the CPU wants to write to it. The Memory will read the address and data the CPU wants to write to and complete those actions. For the stack, the implementation is a decrement after write implementation so the Stack Pointer will be set to 999 on start. The CPU and Memory exit once the instruction 50 is run.
3. This project was a fun and tiring project to do. The CPU and Memory system is easy to understand but the additional components such as the Stack Pointer and the jump instructions were the hardest part to think through. Many of the issues that came were simply accessing the adjacent memory address instead of the actual memory address. The source of the issue came from incrementing or decrementing the PC or SP at the wrong time. Overall, though, this project was fun to do and is satisfying to have it working at the end.