Abed Mir

CS 4348 Project 1

Feb 22, 2020

IPC in Java

The purpose of an operating system is to manage resources and programs on a computing machine. It is simply a software designed to manage other software. This project gave a window into some of the logic that operating systems use when dealing with parallel executing programs and how information can be transferred between each pipeline. The project’s purpose was to allow students to design a miniature version of a basic Instruction Set Architecture that contains a CPU unit and the main memory it uses during execution, both running in parallel reading and writing information from each other. In Java, this is done using the Runtime library, which can create processes that communicate to each other through the System IO stream.

For implementation, two java files were used for the CPU and memory. For memory, the stack was an array of 2000 integers, split in half for user and kernel mode. The memory class, when ran, would read the instructions from the sample text and store it in the stack depending on User or Kernel mode. Then, the processor would begin fetching executions. Based on the program counter’s value, the memory would be read at the counter’s index in the array, and the instruction would be set in the IR variable. Then, a switch could occur that contained all the instructions that could be in IR and is executed. Lastly, there is a timer that increments for interrupts that is procced every X amount of cycles (X specified in command line).

Before the project I had knowledge of what processes were from CS 3377, but never extensively used them at all. This project showed me how the CPU can interact with multiple memory devices through IPC, which is a completely different experience from normal coding where variables and data is assumed to be within the same scope as the file being used or an imported class. This is crucial when it comes to higher levels of computing, cluster computing, and developing algorithms in the fields of AI or Data Science that rely on processor-memory interaction.