C6461 Simulator - Project Overview

Part 3

The simulator component has been updated from part 2 to include a card reader DEVID functionality in the I/O instructions.

In addition, data/program_two.txt has been included which follows the program two project description.

Part 2

The simulator component has been updated from part 1 to include several new components:

1. All instructions have been implemented except CHK and part 4 instructions.

2. A cache of size 16 words has been implemented between the computer and memory. This cache utilizes a simple FIFO replacement policy: as soon as a cache miss occurs, the least recently used word is replaced.

3. In addition, we have developed the program located at data/program.txt to test the functionality of the simulator. This program first prompts the user to enter 20 numbers. Once they have done that, the program prompts the user to enter a guess number. Finally, the program prints out the closest of the first 20 numbers to the guess number.

Part 1

Part 1 implements the ComputerSimulatorGUI, a Java Swing interface that connects to the Computer class to control and observe the CSCI 6461 machine simulator. It displays CPU registers (R0–R3, X1–X3, PC, MAR, MBR, IR) in octal format and allows manual register loading using octal input. The GUI supports program loading through IPL, which assembles a source file and initializes memory, and provides execution control via Step and Run functions. The IR is also shown in binary for instruction inspection.

Part 0 (Assembler)

The assembler component translates human-readable assembly language into machine code that can be executed by the C6461 simulator.

For part 0, the main class being run as a jar is src.assembler.Assembler. This file reads source files and outputs listing and load files via the src.simulator.FileIO class. The Assembler class is essentially a simplified 2-pass assembler, first collecting labels as a map, then generating listing and load file output through system directives and instructions. The

src.assembler.Encoder class is responsible for all instruction encodings, while the src.assembler.Assembler class directly handles directives.