Project 1: CSCI 6461 Machine Simulator

Team 17:

Abhi Bhardwaj, Esha Arun Angadi

Note: This Programs was built and tested on OpenJDK 22.0.2 version

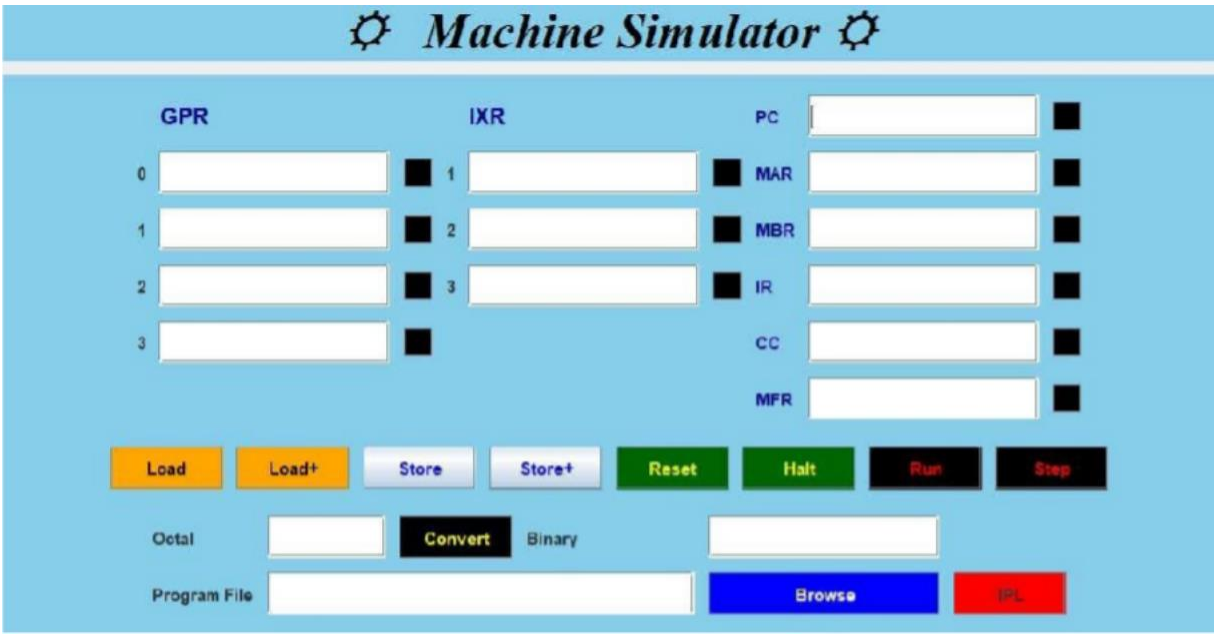
1. Introduction

The Java application known as the CSA Simulator emulates the basic components of computer system architecture. It emulates input/output devices, memory, the central processing unit, and number conversion tools. The system's Graphical User Interface (GUI), developed with Java Swing, allows for easy management of RAM and CPU interaction by the user.

This documentation provides an overview of the project structure, functionality, and how to use Machine Simulator.

2. System Components

GUI (Graphical User Interface)



- The Java interface uses Swing components such as JLabel and JButton to display memory, control buttons, and system registers.
- Key Components-
  - The primary components are GPRs and IXRs, which are both 16-bit registers. When you press the black button, it puts the value into the register.

- Load, Load+, Store, Store+:

  - **Load:** Reads the contents of the memory location addressed by the MAR into the MBR. The user must enter a binary value into the MAR to trigger the load operation.
  - **Load+:** Loads data and adds one to the MAR.
  - **Store:** Moves the value from the MBR to the memory location pointed to by the MAR.
  - **Store+:** Performs a store operation and raises the MAR by one.

- Reset and Halt: **Reset** will reset the Simulator and initialize all values to zero. Pressing **Halt** will stop the execution.

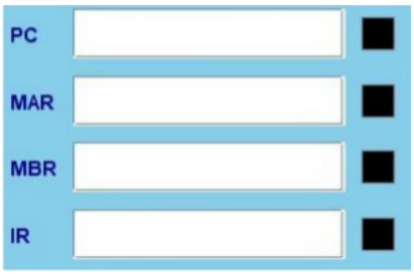
- Run and Step: **Run** will run the input Load file, whereas S**tep** will execute each instruction one by one.

- Octal and Binary: Pressing the convert button will convert any octal number into a 16-bit binary number.

- Program File: This will select an input file (.txt) from the system using the Browse button.

- **IPL**: It will initialize the memory with the contents of the input load file.

- **PC** (12 bits): Contains the address of the next instruction.
- **MAR** (12 bits): Holds the memory location.
- **MBR** (16 bits): Stores the content at the memory location specified by the MAR.
- **IR** (16 bits): Displays the instruction pointed to by the PC.



3. Memory Management:

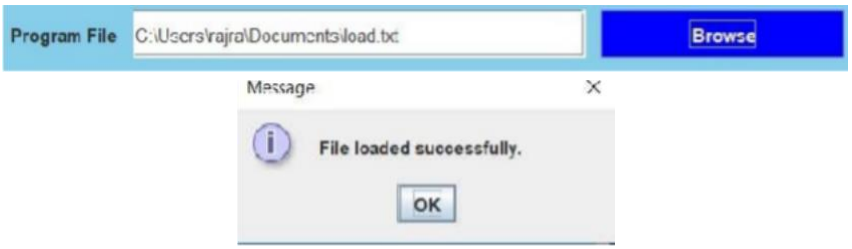
- **2KB Memory Array:** The memory system of the simulator is a short array of 2048 cells (2KB) starting at zero. Each cell has the capacity to store 16 bits of information.
- **Fixed Memory Size:** Dynamic memory allocation may be allowed in future releases despite the current 2KB capacity limit.

4. Number Conversion Utility:

- The Converter Class: This tool includes useful routines for converting between binary, decimal, and hexadecimal forms to make CPU tasks easier.
  - The BinaryToDecimal() method converts binary data to a decimal representation.
  - The DecimalToBinary() function converts decimal values to binary representations.
  - The system uses these conversion techniques frequently, especially when managing memory addresses and CPU instructions.

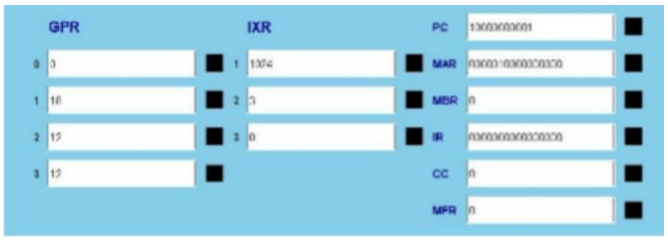
5. Working with the Program File:

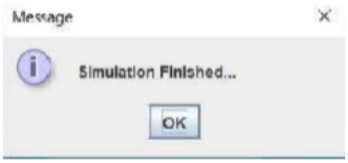
i. Loading the input file(load.txt) from local system. Click the Browse button to load the into the machine.



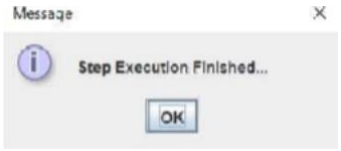
ii. On pressing IPL button will initialize the memory with content present in input file which is **load.txt** in above case.

iii. Now user has two options at this point: select **Run** to run the entire program continuously until it is finished. User will be able to see its final values getting loaded into required registers as present in given code of execution.





iv. If user selects **Step** execution, it allows commands to be carried out one at a time for thorough monitoring and user can see values getting populated in required fields of registers.



6. Run the panel:

- Click the "Run" button to begin the process. The memory address register (MAR) stores the current address, the memory buffer register (MBR) displays the necessary data, and the program counter (PC) advances to the next address.
- Until an operation with opcode 0000 (the HLT instruction) is encountered, the machine will continue to execute each instruction and store data in the Instruction Register (IR).

***Note**: The load.txt file is used to load all addresses and values with help of clicking IPL button. 0000 is the default address if it is not present in the load.txt. The machine is able to access addresses between 0 and 2048.*

7. Operating the Assembler

- Open the terminal on your system.
- Navigate to the directory where your ***CSCI6461 Team2.jar.***

8. Conclusion

Finally, we have successfully created the Machine simulator front panel that has all the components and functionality required, such as a simple graphical user interface and the basic memory and CPU controls. Serving as the primary user interface, this front panel increases usability as it allows users to interact with the simulator in an intuitive way.