Design Document

1. Introduction

The UVSim is a simple virtual machine, but powerful. The UVSim can only interpret a machine language called BasicML.

2. User Stories

2.1 User Story 1: Learning Machine Language

As a computer science student, I want to learn and understand machine language, so I can execute BasicML programs on the UVSim simulator and enhance my skills.

2.2 User Story 2: Experimenting with Code

As a computer science student, I want to experiment with code in a safe and controlled environment, so that I can make mistakes, learn from them, and improve my coding skills.

3. Use Cases

3.1 Use Case 1: Load Program into Memory

Description: User loads a BasicML program into the UVSim memory starting at location 00.

3.2 Use Case 2: Input/Output Operations

Description: User utilizes READ & WRITE operations for input and output to and from specific memory locations.

3.3 Use Case 3: Load and Store Operations

Description: User uses LOAD & STORE operations to manage data within the memory and accumulator.

3.4 Use Case 4: Perform Arithmetic Operations

Description: User performs arithmetic operations (ADD, SUBTRACT, DIVIDE, and MULTIPLY) on words in the memory and accumulator.

3.5 Use Case 5: Control Operations

Description: User uses control operations (BRANCH, BRANCHNEG, BRANCHZERO, and HALT) to control the flow of the program.

3.6 Use Case 6: Debugging

Description: Program displays specific errors based on user input (ex. division by zero, invalid operation, invalid memory address)

3.7 Use Case 7: Launch Simulator from Command Line

Description: User launches UVSim from the command line.

3.8 Use Case 8: Interpret BasicML Instructions

Description: UVSim interprets BasicML instructions and executes them accordingly.

3.9 Use Case 9: Fetch Instructions From Memory

Description: UVSim will fetch the next instructions from memory to be executed.

3.10 Use Case 10: Pause & Resume Execution

Description: The user can halt and resume the program execution.

4. Architecture and Components

The UVSim will contain the following main components:

- **CPU**: This will be responsible for interpreting and executing BasicML instructions.
- **Register**: This will hold information before it is used in calculations or examined in various ways.
- **Main Memory**: Equipped with a 100-word memory, and these words are referenced by their location numbers 00, 01, ..., 99. The BasicML program must be loaded into the main memory starting at location 00 before executing. Each instruction written in BasicML occupies one word of the UVSim memory (instruction are signed four-digit decimal number). We shall assume that the sign of a BasicML instruction is always plus, but the sign of a data word may be either plus or minus. Each location in the UVSim memory may contain an instruction, a data value used by a program or an unused area of memory. The first two digits of each BasicML instruction are the operation code specifying the operation to be performed.
- **Accumulator**: A register into which information is put before the UVSim uses it in calculations or examines it in various ways.

5. Technologies and Tools

- **Programming Language**: Python
- **Version Control System**: Git