UVSim Design Document

# **UVSim**

UVSim is a simulator used by computer science students to learn Basic Machine Language. It provides a virtual environment to learn about memory, registers, and machine-level operations. It uses four digit numbers to interpret and execute these operations.

# **User Stories**

* **Student:** As a student, I want to load and execute a BasicML program in the simulator so I can learn and understand how machine language instructions work.
* **Instructor:** As an instructor, I want to track errors and executions so I can explain the behavior of a program and debug for my students.

# **Use Cases**

* **UC1: Read word into memory**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Store a four-digit number at a specified memory location
  + **Steps:**
    1. User inputs a signed four-digit number
    2. System validates the number
    3. System checks if memory location specified is free
    4. If valid, system stores the number at the memory location
* **UC2: Write word to screen**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Retrieve and display a word from a specified memory location
  + **Steps:**
    1. User specifies a memory location
    2. System validates the memory location
    3. System retrieves data from the location
    4. System displays the retrieved data
* **UC3: Perform Arithmetic operation**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Execute arithmetic operations using the accumulator
  + **Steps:**
    1. System fetches the operation code
    2. System retrieves the operand from the memory address
    3. System validates the memory address and checks if occupied
    4. System retrieves the word from the memory address
    5. System performs arithmetic operation using the accumulator and the retrieve word
    6. System updates the accumulator with the result
* **UC4: Execute Branch Instruction**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Update the program counter based on a branch instruction
  + **Steps:**
    1. System fetches the branch instruction and operand
    2. System validates the memory address
    3. System updates the program counter based on the instruction
* **UC5: Handle Erros**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Detect and handle errors appropriately
  + **Steps:**
    1. System detects an error during execution
    2. System displays an error message
    3. System halts execution or prompts user for input correction
* **UC6: Dump Memory**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Display all memory contents for inspection
  + **Steps:**
    1. User issues a command to dump memory
    2. System displays memory values in rows with their address
* **UC7: Inspect Current Instruction**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Display the current instruction, accumulator, and program counter
  + **Steps:**
    1. User issues a command to inspect current state
    2. System retrieves and displays:
       - Decoded operation and operand of current instruction
       - Current value of the accumulator
       - Current program counter value
* **UC8: Load program**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Load a program from a text file into memory
  + **Steps:**
    1. User specifies the input file containing BasicML program
    2. System validates the file format
    3. System loads the instructions into memory address
* **UC9: Execute program**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Execute the BasicML program sequentially
  + **Steps:**
    1. System sets the program counter to 00
    2. System fetches the instruction at current program counter
    3. System decodes and executes the instruction
    4. System updates the program counter
    5. Steps repeat until HALT instruction is reached or error occurs
* **UC10: Reset program**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Reset simulator state
  + **Steps:**
    1. User issues a reset command
    2. System clears all memory contents
    3. System resets the accumulator to zero
    4. System resets the program counter to 00
* **UC11: Save program**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Save the current program state
  + **Steps:**
    1. User issues a save command
    2. System writes memory, accumulator, and program counter values to a file
* **UC12: Log execution**
  + **Actor:** User
  + **System:** UVSim
  + **Goal:** Maintain an execution log for debugging and analysis
  + **Steps:**
    1. System maintains a log during program execution
    2. For each instruction, system records:
       - Executed instruction and address
       - Updated state of the accumulator and memory
    3. System displays or saves the log upon program termination