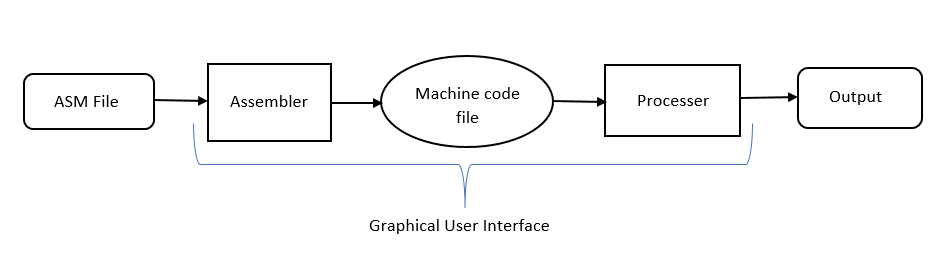
RISC-V ISA Simulator:

Design Documentation

**Features:**

* **Simulator backend:** C++
* **ISA:** RISC-V RV32IM
* **Size of instruction:** 32 bits
* **Number of registers:** 32
* **Size of registers:** 32 bits

**Overview:**

**Phase 1: Five step instruction execution**

## Input:

Input to the simulator is a machine code file that contains the encoded instruction and the corresponding address at which instruction is supposed to be stored, separated by space.

For example:

0x0 0x00500513

0x4 0x008000EF

0x8 0x04000463

Functional behaviour:

All the instructions given in the input .mc file is executed as per the functional behaviour of the instructions. Each Instruction will go through the following steps:

* Step 1: **Fetch:** It gets the instruction and increment the PC, and prints that the instruction is fetched and PC is incremented.
* Step 2: **Decode:** Identify the instruction and registers (source and destination), and prints the operation, rs1, rs2, rd.
* Step 3: **Execute:** ALU operations is done or effective address calculation for load, store instructions. The control circuitry select the ALU operation and either Read data from the registers or a sign-extended immediate value as inputs to the ALU, then prints the instruction statement.
* Step 4: **Memory Access:** Read/write data from/to the memory. The control lines set in this stage are Branch, Memory Read, and Memory Write. Then print the retrieved memory at the memory address.
* Step 5: **Register Update or Writeback:** update the destination register, and print the result updated in the destination register.

# Test plan:

We test the simulator with the following assembly programs:

* Bubble sort
* Sum