DESIGN SPEC DOCUMENT

ECE-593: Fundamentals of Pre-Silicon Validation
Maseeh College of Engineering and Computer Science
Winter, 2026

Project Name: Design, Implementation and Verification of a MIPS-Lite 5-Stage In-Order CPU.

Members:
Mathumitha Rajan - 985876262
Srinivasan Venkatavardhan - 986982335

Date: 27th Jan 2026

Project Name	Design, Implementation and Verification of a MIPS-Lite 5-Stage In-Order CPU.
Location	PSU, ECE 593
Start Date	Jan 15, 2026
Estimated Finish Date	March 14, 2026
Completed Date	

Prepared by: FinalProject_Group 5	
Prepared for: Prof. Venkatesh Patil	
Team Member Name	**Email**
Mathumitha Rajan	mrajan@pdx.edu
Srinivasan Venkatavardhan	srinilnu@ pdx.edu

Design Features:
• In-order, single-issue MIPS-Lite processor
• Classic 5-stage pipeline: IF, ID, EX, MEM, WB
• Fixed 32-bit instruction format
• Supports arithmetic, logical, immediate, load/store, and branch instructions
• Register file with two read ports and one write port
• ALU supporting basic arithmetic and logical operations
• Pipeline registers between all stages
• Data hazard handling using forwarding and pipeline stalling
• Control hazard handling by resolving branches in EX stage
• No out-of-order execution or speculative execution
• Single clock domain design

Project Description:
This project involves the design, implementation, and functional verification of a simplified MIPS-like in-order processor using SystemVerilog. The processor follows a classic 5-stage pipelined architecture consisting of Instruction Fetch (IF), Instruction Decode (ID), Execute (EX), Memory Access (MEM), and Write Back (WB). The primary goal of the design is to understand pipelined processor microarchitecture, including Datapath design, pipeline control, and hazard handling. Verification will be performed using directed tests and randomized instruction sequences to ensure correct architectural behavior.

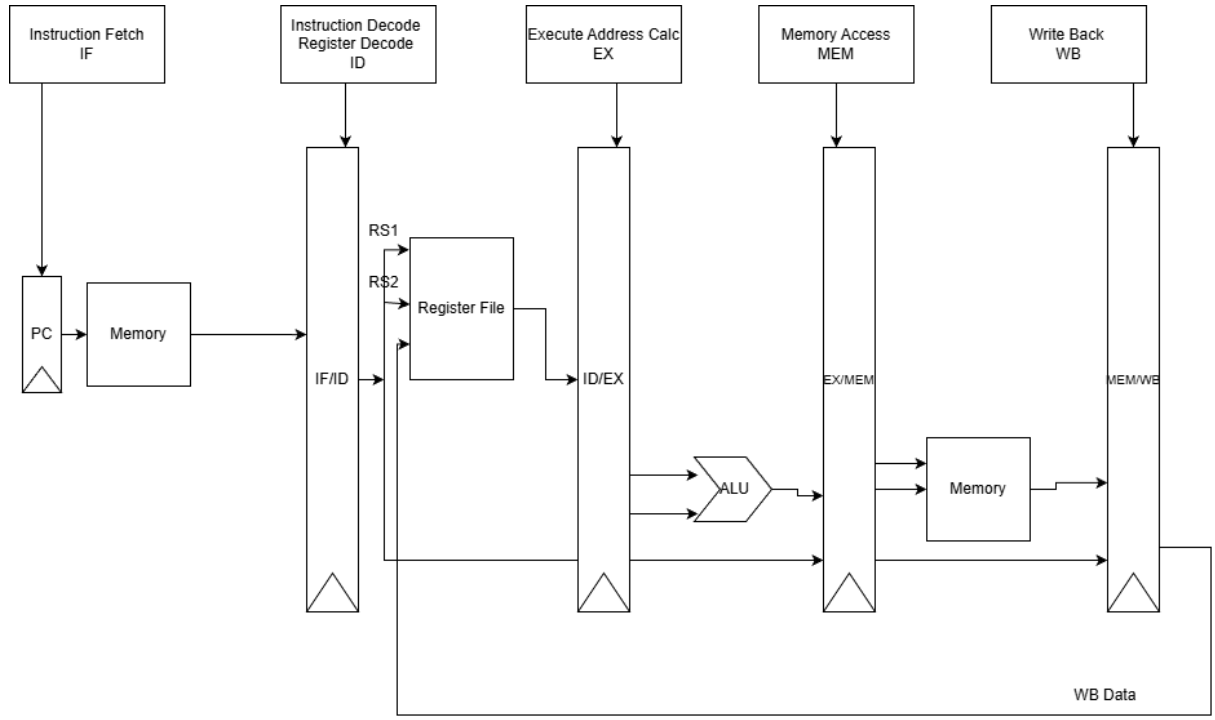
Important Signals/Flags

- `clk` – System clock
- `reset` – Global synchronous reset
- `pc` – Program counter
- `instr` – Instruction fetched from instruction memory
- `reg_rs1_data` – Source register 1 data
- `reg_rs2_data` – Source register 2 data
- `alu_result` – Output of ALU
- `mem_read` – Data memory read enable
- `mem_write` – Data memory write enable
- `reg_write` – Register file write enable
- `branch_taken` – Indicates taken branch
- `stall` – Pipeline stall signal
- `flush` – Pipeline flush signal

Design Signals

- Control signals for ALU operation selection
- Register destination selection signals
- Memory read and write control signals
- Write-back control signals
- Forwarding control signals
- Pipeline stall and flush control signals

Block Diagram



References/Citations

• Patterson, D. A., & Hennessy, J. L., Computer Organization and Design: The Hardware/Software Interface

• MIPS32 Architecture Overview

• Course lecture slides (ECE 593 – Pre-Silicon Validation)