RISC-V Single-Cycle CPU Simulation

GIK Institute Students
Department of Computer Engineering

December 3, 2025

Abstract

This document details the design, implementation, and verification of a 32-bit Single-Cycle RISC-V Processor. The simulation verifies the correct execution of arithmetic, logic, and memory instructions within a single clock cycle.

Contents

1 Introduction

This document presents the design and simulation of a **32-bit Single-Cycle RISC-V Processor**, implemented and verified by students at the **Ghulam Ishaq Khan (GIK) Institute**.

1.1 Objective

The main objective is to ensure that the CPU executes instructions correctly in a **single clock cycle per instruction**. The design focuses on:

- **Data Path:** Correct movement of data between registers, ALU, and memory.
- **Control Unit:** Generation of accurate control signals (e.g., `RegWrite`, `MemWrite`, `MemToReg`).
- **Instruction Handling:** Execution of **I-type**, **R-type**, and **S-type** instructions.

2 Test Assembly Program

The program consists of 5 instructions designed to test arithmetic operations, memory access, and register usage.

Table 1: Test Assembly Program Execution Flow

Step	Address	Instruction	Machine Code	Description
1	`0x00`	`addi x7, x0, 2`	`00200393`	Adds 2 to `x0` and stores in `x7`.
2	`0x04`	`addi x6, x0, 0x123`	`12300313`	Adds 0x123 (291) to `x0`; stores in `x6`.
3	`0x08`	`sw x7, 0(x6)`	`00732023`	Stores `x7` value into memory at address `x6`.
4	`0x0C`	`lw x8, 0(x6)`	`00032403`	Loads data from memory address `x6` into `x8`.
5	`0x10`	`add x9, x8, x7`	`007404b3`	Adds `x8 + x7` and stores in `x9` (Result: 4).

3 CPU Microarchitecture Overview

The CPU is composed of several modular Verilog components, each performing a specific role in the fetch-decode-execute cycle.

3.1 Instruction Memory

- Stores all program instructions.
- **Input:** Program Counter (PC).
- **Output:** 32-bit instruction word.

3.2 Program Counter (register_flop)

- Tracks the current instruction address.
- Updates on the rising edge of the clock.
- Resynchronous reset sets the PC to 0.

3.3 Register File (regfile)

- Contains 32 general-purpose registers (`x0`–`x31`).
- Supports dual-port read and single-port write.
- `x0` is hardwired to 0.

3.4 ALU (Arithmetic Logic Unit)

- Performs arithmetic (ADD, SUB) and logical operations.
- **Inputs:** Operand A, Operand B, ALU Control.
- **Output:** Calculation result and Zero flag.

3.5 Data Memory (data_memory)

- Simulates system RAM.
- **Store (SW):** Writes data when `mem_write` is high.
- **Load (LW):** Asynchronously outputs data based on address.

3.6 Control Unit

- Decodes the 7-bit opcode, funct3, and funct7.
- Generates control signals: `alu_src`, `mem_write`, `mem_to_reg`, etc.

4 Cycle-by-Cycle Execution

The following table details the internal state changes during the simulation.

Table 2: Cycle-by-Cycle Simulation Results

Cyc	Instruction	Stage	Activity	Result
1	`addi x7, x0, 2`	Decode	ALU adds $0 + 2$	`x7 = 2`
2	`addi x6, x0, 0x123`	Decode	ALU adds $0 + 291$	`x6 = 291`
3	`sw x7, 0(x6)`	Mem Wr	Mem write enabled at 0x123	`M[291]=2`
4	`lw x8, 0(x6)`	Mem Rd	Mem read enabled at 0x123	`x8 = 2`
5	`add x9, x8, x7`	Ex/WB	ALU adds $2 + 2$	`x9 = 4`

5 CPU Data Flow Diagram

The visual representation below illustrates the high-level connection of modules within the CPU.

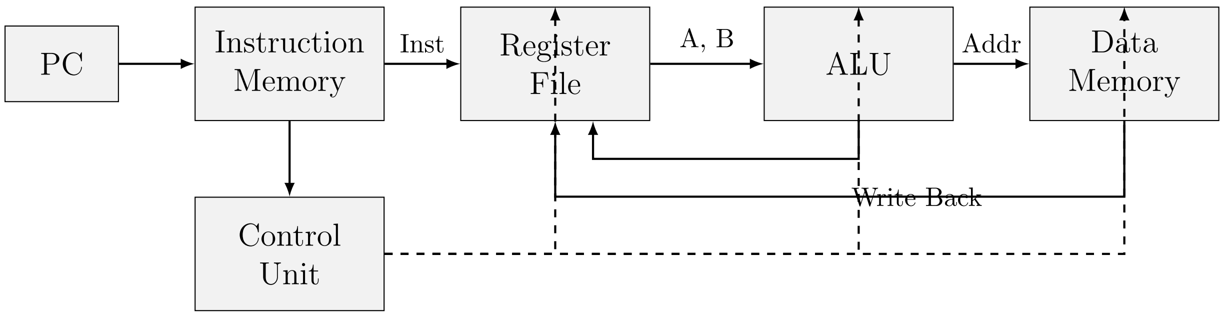

Figure 1: Abstract Data Path and Control Flow

6 Conclusion

The simulation validated that the **Single-Cycle RISC-V CPU** executes instructions correctly:

- Arithmetic and memory instructions were processed without stalls or hazards.
- The final register state **x9 = 4** confirms the correct interaction between the ALU, memory subsystem, and register file.
- The design successfully implements the core RISC-V Instruction Set Architecture subset.

Future Enhancements

To further improve the design, the following features are proposed:

1. Implementation of branch instructions (`BEQ`, `BNE`) for control flow.
2. Extension of the ALU to support logic operations (`AND`, `OR`, `XOR`).
3. Integration of a visual timing diagram for signal verification.