**VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY**

**UNIVERSITY OF TECHNOLOGY FACULTY OF ELECTRICAL AND ELECTRONICS ENGINEERING**

**DEPARTMENT OF ELECTRONICS**

🙞∙∙∙☼∙∙∙🙜



**Capstone Project 1 – Progress Report**

**RISC-V Processor with branch comparator**

**another model of ADDER**

**HK242**

**Date: 14/4/2025**

**Supervisor: Dr.Tran Hoang Linh**

|  |  |
| --- | --- |
| **Full Name** | **Student’s ID** |
| Đào Nguyên Khôi | 2151215 |

**Hồ Chí Minh City – 2025**

Table of Contents

[**Introduction** 3](#_Toc195533861)

[**I.)** **Overview:** 4](#_Toc195533862)

[**II.)** **Instruction Implementation:** 5](#_Toc195533863)

[**III.)** **Top-Level Architecture Diagram:** 6](#_Toc195533864)

[**IV.)** **Module Highlights:** 7](#_Toc195533865)

[**V.)** **Simulation and Verification:** 8](#_Toc195533866)

# **Introduction**

In recent years, the RISC-V instruction set architecture (ISA) has emerged as a promising open standard for processor design. Unlike proprietary architectures, RISC-V is open-source, allowing researchers, students, and engineers to explore CPU design without licensing constraints. Its simplicity, modular design, and support for customization make it an ideal choice for academic projects and educational purposes.

This project focuses on re-implementing a single-cycle RISC-V CPU using Verilog on the Quartus development environment. The main goal is to understand and apply core computer architecture concepts, including the design of the datapath, control unit, and ALU. By constructing these components from the ground up, the project reinforces theoretical knowledge through practical implementation.

The design includes essential features of the RISC-V base integer instruction set (RV32I), supporting operations such as arithmetic computation, logic operations, data transfer, and branching. Each instruction is fetched, decoded, and executed within a single clock cycle, reflecting a simplified yet instructive architecture model.

To verify the functionality of the CPU, testbenches are developed for simulation, allowing step-by-step observation of instruction execution. This ensures correctness in logic and timing before synthesis on FPGA. The Quartus platform is used to write, simulate, and potentially deploy the design on actual hardware, although the focus remains on correctness and simulation-level validation.

Overall, the project offers a valuable opportunity to explore digital system design, improve HDL programming skills, and gain insights into the internal workings of modern processors through the lens of the RISC-V architecture.

1. **Overview:**

This project focuses on the design and implementation of a Single-Cycle CPU following the RISC-V RV32I instruction set architecture. The primary goal is to support most of the core integer instructions defined in the RV32I base ISA, including arithmetic, logical, memory access, and control-flow operations. By targeting a wide coverage of instructions, the CPU aims to behave similarly to a simplified RISC-V processor model suitable for educational and experimental use.

Beyond achieving instruction-level functionality, the project also aims to explore improved architectural components, with special attention given to performance and efficiency. Traditional designs often rely on basic structures such as the Ripple Carry Adder, which, while simple, are not optimal in terms of speed. To address this, the Arithmetic Logic Unit (ALU) in this project is designed using a **Kogge-Stone Adder**, a parallel-prefix adder architecture that significantly reduces the delay in addition operations. This change is intended to improve the performance of arithmetic calculations, especially in time-critical instructions like ADD, ADDI, and related operations.

Furthermore, updated module designs are considered for other critical components such as the branch control unit, multiplexer structures, and the overall datapath. These improvements aim to make the CPU not only functional but also optimized in terms of internal signal propagation and execution timing. The development process involves modeling in Verilog HDL and simulation using the Quartus environment to verify correctness and behavior.

At this stage of the project, the CPU is partially implemented and tested. This report outlines the progress made so far, including the number of completed instructions, key architectural components, and testing results, as well as the next steps toward completing the processor.

1. **Instruction Implementation:**

Gfg

A screenshot of a computer

AI-generated content may be incorrect.

j

1. **Top-Level Architecture Diagram:**

The top-level architecture of this Single-Cycle RISC-V CPU is primarily inspired by the structure presented in the book *"Digital Design and Computer Architecture – RISC-V Edition"*. The original diagram from the book provides a foundational overview, illustrating the essential modules and their basic interconnections. However, while useful for learning purposes, the book's design abstracts many low-level details and omits certain control paths and data flows that are crucial for an actual implementation.

To build a working and more detailed CPU, the architecture has been **modified and extended**. Several new components have been added to better reflect real-world processor behavior and to prepare the design for future enhancements. Notably, additional **multiplexers (MUXes)** were introduced to improve control over data routing, especially in instruction types involving immediate values, write-back selection, and program counter updates.

A diagram of a computer

AI-generated content may be incorrect.

fgfgfg

1. **Module Highlights:**

fgffg

1. **Simulation and Verification:**

* Test instruction code:

A screenshot of a computer code

AI-generated content may be incorrect.

* A screenshot of a computer

  AI-generated content may be incorrect.Waveform result:

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.