|  |
| --- |
|  |

Jordan National Semiconductor Design Competition (JOSDC’2023)

MIPS single cycle 32 bit

By

Omar AL-khasawneh & Omar Salah & Moyad Abu Mallouh

Amman, Jordan

Month Year

Acknowledgments

Recognition or favorable notice for people.

Abstract

Describe your project briefly in few paragraphs. The abstract should not exceed one page.

**Table of Contents**

[1 Introduction 7](#_Toc146783944)

[1.1 Section 7](#_Toc146783945)

[2 Design 8](#_Toc146783946)

[2.1 Hardware Design and Implementation 8](#_Toc146783947)

[2.2 Coding and Software Development 8](#_Toc146783948)

[3 Results 9](#_Toc146783949)

[3.1 Experiment/Simulation Results Discussion: 9](#_Toc146783950)

[3.2 Prototype Setup 9](#_Toc146783951)

[3.3 Validation of requirements 9](#_Toc146783952)

[4 Conclusion 10](#_Toc146783953)

**List of Figures**

Figure 1.1 Figure one in chapter one ..……………………………….……………..page number

Figure 2.1 Figure one in chapter two ..……………………………….……………..page number

Figure 2.2 Figure two in chapter two ..……………………………….……………..page number

**List of Tables**

Table 1.1 Table one in chapter one ..……………………………….……………..page number

Table 2.1 Table one in chapter two ..……………………………….……………..page number

Table 2.2 Table two in chapter two ..……………………………….……………..page number

# Introduction

## Introduction

In the ever-evolving landscape of computer science and engineering, the design and implementation of central processing units (CPUs) continue to be a focal point of innovation and exploration. Among the various CPU architectures, the development of a single-cycle CPU, akin to the revered MIPS (Microprocessor without Interlocked Pipeline Stages), stands as a compelling and vital undertaking. This project represents our endeavor to grasp and engineer a fundamental yet pivotal component of modern computing systems.

## The Importance of the Design

The importance of designing and understanding single-cycle CPUs is multifaceted and integral to the field of computer science and engineering. In a world where processing power and efficiency are at the forefront of technological advancement, the design of CPUs becomes an arena where every microarchitectural decision counts. The single-cycle CPU, characterized by its simplicity and transparency, holds particular significance. It offers a clear and unambiguous model for comprehending the inner workings of a CPU. By its nature, it encapsulates the essence of instruction execution in a single clock cycle, thereby facilitating a straightforward understanding of processor operations. In this project, we dive into this simplicity to harness its educational and practical merits.

## Motivation

The motivation behind this project stems from a shared passion for learning and a profound curiosity about the architecture of modern computers. Understanding the CPU, often referred to as the "brain" of a computer, is a fundamental step in comprehending how computers execute instructions, process data, and perform complex operations. As students in the field of computer science and engineering, our motivation is twofold. Firstly, we aim to deepen our knowledge of CPU design, enabling us to demystify the underlying mechanisms of computing systems. Secondly, we aspire to create a resource for students and enthusiasts alike to embark on a journey of discovery, employing our project as an educational tool.

## Why This Topic is Important for Students

For students, the significance of exploring the design and implementation of a single-cycle CPU lies in the educational value it holds. It serves as a comprehensive learning experience, bridging theory and practice in the field of computer architecture. As an educational endeavor, our project offers students the opportunity to delve into the intricacies of CPU design, to comprehend the essence of instruction execution, and to witness the tangible results of their efforts. This not only enhances their academic knowledge but also fosters a deeper appreciation for the technologies that underpin our modern world. By providing a practical and accessible entry point into CPU architecture, our project empowers students to become proficient problem solvers, critical thinkers, and innovative engineers.

## Objectives of the Project:

1. **Single-Cycle CPU Design:** The primary objective of this project is to design and implement a single-cycle CPU, heavily inspired by the MIPS architecture. This CPU will be capable of executing a set of fundamental instructions in a single clock cycle.
2. **Educational Resource:** We aim to create an educational resource that not only serves our learning goals but also benefits other students and enthusiasts in the field of computer science and engineering. The project aims to offer a clear, step-by-step insight into CPU design and facilitate a deeper understanding of the architectural choices involved.
3. **Comprehensive Understanding:** To achieve a comprehensive understanding of computer architecture, we will delve into the intricacies of various CPU components, including the control unit, ALU, registers, and memory hierarchy. This understanding will enable us to articulate the rationale behind design decisions.
4. **Performance Analysis:** The project seeks to analyze the performance of the single-cycle CPU in terms of execution speed, resource utilization, and comparison with other CPU architectures. This analysis will help in assessing the practical implications and limitations of the design.

## Description of Design Achieved:

In pursuit of these objectives, we have successfully designed a single-cycle CPU model with the following characteristics:

* **MIPS-Inspired Architecture:** Our CPU design is heavily influenced by the MIPS architecture, renowned for its simplicity and elegance. This architecture served as a valuable reference point in shaping our CPU's instruction set, control unit, and data path.
* **RISC (Reduced Instruction Set Computer) Principles:** Our CPU follows the RISC principles by focusing on a limited set of simple and frequently used instructions. This design choice enhances the CPU's efficiency and ease of understanding.
* **Single-Cycle Execution:** Our CPU is designed to execute instructions in a single clock cycle. This efficient one-cycle process involves fetching instructions, decoding them, executing operations, and writing results back to registers.
* **Control Unit:** We have implemented a control unit that generates control signals to direct the CPU's operations based on the current instruction. This control unit is responsible for managing the flow of data within the CPU.
* **ALU and Registers:** The Arithmetic Logic Unit (ALU) performs arithmetic and logical operations, while the registers store data. Our CPU features a specific number of registers, and the ALU is capable of executing a variety of operations.
* **Memory Hierarchy:** We have incorporated a memory hierarchy with separate instruction and data memory. This design choice aligns with modern CPU architectures and offers an accurate representation of how instructions and data are stored and accessed.

## Design Requirements:

Our CPU design adheres to the following key requirements:

* **Simplicity:** The design should prioritize simplicity and clarity to serve as an educational tool. It should be comprehensible to students and enthusiasts with a basic understanding of digital logic and computer architecture.
* **One-Cycle Execution:** All instructions should be executed in a single clock cycle, reflecting the core concept of a single-cycle CPU.
* **Instruction Set:** The CPU should support a defined instruction set, inspired by the MIPS architecture. This instruction set should include fundamental operations such as arithmetic, logic, and data movement instructions.
* **Efficiency:** The design should strive for efficiency by optimizing the use of resources and minimizing redundancy.

## The teams member responsibility

### Omar AL-khasawneh (Leader)

* ALU design
* ALU control design
* PC design

### Omar AL-salah

* Register file
* Control unit

### Moayyad Abu Mallouh

* Instruction memory
* Data memory
* Mux between component

## Organization of the rest of the documentation.

# Design

## Hardware Design and Implementation

* Describe the hardware design and components used.
* Explain the physical setup of the hardware.
* Discuss any challenges faced during hardware implementation.
* Include photographs, diagrams, or schematics of the hardware setup.

## Coding and Software Development

* Describe the coding and software development aspects of your project.
* **Explain the programming languages and tools used.**
* Provide code flow charts or algorithms related to your project.
* **Discuss the coding challenges and solutions.**

### The tools we are used

Verilog is used for various purposes in the field of digital design and electronic engineering for several reasons:

Digital Circuit Design: Verilog is primarily used to design digital circuits. It allows engineers to model and simulate the behavior of these circuits before they are physically built, which helps in verifying functionality and fixing issues in the design phase.

FPGA and ASIC Development: Verilog is essential for programming Field-Programmable Gate Arrays (FPGAs) and designing Application-Specific Integrated Circuits (ASICs). It provides a hardware description that can be synthesized into the physical logic of these devices.

Simulation: Verilog supports simulation at various levels of abstraction, from high-level system simulations down to gate-level simulations. This helps in testing the functionality and performance of digital designs without the need for physical prototypes.

Verification: Engineers use Verilog to verify the correctness of their designs. It allows for creating testbenches that can apply stimulus to the design and check if it behaves as expected, identifying and resolving any design flaws or errors.

Rapid Prototyping: Verilog accelerates the development of digital systems by allowing engineers to rapidly prototype and iterate on designs. This reduces development time and costs.

Reusability: Verilog promotes the reusability of modules and components. Engineers can create well-defined modules that can be reused in different projects, enhancing productivity.

Parallelism: Verilog's inherent support for parallelism and concurrency makes it suitable for modeling and simulating systems with multiple components running in parallel.

Industry Standard: Verilog is widely accepted and used in the electronics industry. Many Electronic Design Automation (EDA) tools support Verilog, and there's a vast community and resources available for Verilog users.

### Challenging

We are challenged with learning resource constraints due to limited availability on the internet, and optimization presents a significant challenge.

How to make our design faster is the most challenging.

### Omar hon

### Omar hon

# Results

• Present the results of testing and validation procedures for both the simulation and hardware.

• Include data, graphs, and tables to support your findings.

• Discuss the performance and functionality of the integrated system.

• Use a table to summarize that requirements were met

## Experiment/Simulation Results Discussion:

* + Use
    - Tables
    - Graphs
    - Waveform
    - figures

## Prototype Setup

* + Hardware
  + Software

## Validation of requirements

* + Discuss and analyze whether the requirements are met

# Conclusion

The journey to design a single-cycle CPU, inspired by the venerable MIPS architecture, has led us to a comprehensive exploration of computer architecture, instruction execution, and the intricacies of CPU design. This endeavor has not only enriched our understanding of these fundamental concepts but has also yielded insights and a valuable educational resource for students and enthusiasts alike.

### Key Findings and Implications

Through this project, we have made several key findings and drawn implications:

1. Educational Resource: We have successfully created an educational resource that unravels the intricacies of CPU design. The project offers a detailed, step-by-step account of how a single-cycle CPU operates, emphasizing simplicity and clarity, making it accessible to a broad audience.
2. Simplicity and Efficiency: The MIPS-inspired single-cycle design demonstrates the power of simplicity and efficiency in CPU architecture. We have observed that by adhering to a reduced instruction set and single-cycle execution, it is possible to achieve transparency in the execution of instructions.
3. Performance Analysis: Our performance analysis revealed that the single-cycle CPU's execution time is generally quicker compared to other designs. However, this speed comes at the expense of resource utilization, and it may not be the most efficient choice for all applications.

### Project’s objectives achieved.

While our project has achieved its primary objectives, there are avenues for future work and exploration:

1. Pipeline CPU Design: Future projects could delve into the design of pipeline CPUs, which allow for a more balanced trade-off between execution speed and resource utilization. A pipeline design can be more efficient and is widely used in modern processors..
2. Advanced Instructions: Expanding the instruction set to include more complex operations, such as floating-point arithmetic or SIMD instructions, would further enrich the educational value of the CPU model.
3. Out of order processor: Out-of-order processors are often found in high-performance computing environments and modern microprocessors where speed and efficiency are paramount.

References

Follow a format consistent with IEEE guidelines and utilize conference and journal papers for your reference list

Computer Organization and Design MIPS Edition: The Hardware/Software Interface

Mohammed Hammori channel

Chip design training from ruba AL0khasawneh

APPENDICES

These are detailed documentation of points mentioned in the report (e.g. technical data, questionnaires, chart …. etc.) which are considered supplementary information but too long or not quite relevant enough to include in the main body of the report.

Appendices may be labeled with letters as Appendix A, Appendix B, and so on.

Example,

Appendix A: CODE