Report on

RISC and CISC architectures with details about both the architectures. Conclude with a comparative analysis.

Department of Computer Science and Engineering

By

Suvadeep Bhattacharjee – 12200122130

Third Semester

PCC-CS302

Computer Organization

St. Thomas' College of Engineering and Technology, Kolkata

Affiliated to

Maulana Abul Kalam Azad University of Technology, West Bengal

Session:2023-2024

CONTENTS

Abstract:

Comparison between RISC and CISC in the language of computer architecture for research is not very simple because a lot of researchers worked on RISC and CISC Architectures. Both these architectures differ substantially in terms of their underlying platforms and hardware architectures. The type of chips used differs a lot and there exists too many variants as well. This paper gives us the architectural comparison between RISC and CISC architectures. Also, we provide their advantages performance point of view and share our idea to the new researchers.

INTRODUCTION:

The Microprocessor chips are divided into two categories. In both architectures the main purpose is too optimal the performance of the system. Our Research on these two architectures which is complex, but we do our best [1-8]. CISC is stand for complex instruction set computer. Nowadays the PCs mostly uses CISC architecture Like AMD and Intel etc. CISC chips have large and complex instructions [9-13]. We know that hardware is faster than software so therefore one should make a powerful instruction set which provides programmers with assembly instructions to do a lot with short program. In common CISC chips are relatively slow (as compared to RISC). RISC is stand for Reduced Instruction Set Computer. Nowadays mostly Mobile Phones Based on RISC architecture Like MIPS and ARM etc. RISC has simple and small Instruction. RISC chips Comes around the mid 80's because the reaction of CISC chips. The philosophy behind that almost no one use complex instructions and mostly people use compilers which never use complex instructions. So, for Apple uses RISC chips [14-20]. So therefore, simple and faster instructions are better than large complex and slower (CISC) instructions. However, RISC required more instruction to complete a task than CISC. An advantage of RISC is that because it more simple instructions. RISC chips require less transistors which makes easier to design and cheaper to produce. So now it easier to write powerful optimal compilers since fewer instructions exists.

This report aims to provide a comprehensive analysis of two prominent computer architectures, namely Reduced Instruction Set Computer (RISC) and Complex Instruction Set Computer (CISC). Both architectures play a crucial role in shaping the performance, efficiency, and design considerations of modern processors. This report presents an overview of RISC and CISC architectures, highlighting their key characteristics, advantages, and disadvantages, followed by a comparative analysis.

COMPARISON OF RISC AND CISC:

First, we explain the properties of CISC architectures and then we explain the properties of RISC architecture.

- **CISC Architecture (Complex Instruction Set Computer):**

  CISC architecture, in contrast, focuses on providing a rich set of complex instructions that can perform multi-step operations in a single instruction. The key features of CISC architecture are:

Properties of CISC:

Certainly, let's dive deeper into some additional properties and considerations of Complex Instruction Set Computer (CISC) architectures:

1.Versatility and Expressiveness:

CISC architectures are known for their extensive and diverse instruction sets. This characteristic allows programmers to express complex operations in a single instruction, which can lead to more concise and intuitive code. The ability to perform intricate calculations, memory manipulation, and data transformations in a single instruction simplifies programming for certain applications.

2.Reduced Memory Access:

The presence of complex instructions in CISC architectures can lead to reduced memory access requirements compared to RISC architectures. This is because a single complex instruction might replace multiple simpler instructions, reducing the need to load multiple instructions from memory. As a result, CISC architectures can sometimes achieve higher execution efficiency for certain types of programs that heavily rely on complex operations.

3.Efficient High-Level Language Compilation:

CISC architectures are often considered more amenable to high-level language compilation due to their wide range of instructions. High-level language constructs can be more directly mapped to CISC instructions, potentially leading to optimized code generation. This can simplify the work of compilers and result in more efficient executable code.

4.Legacy Compatibility:

CISC architectures have been around for a long time and have accumulated a substantial base of software written for them. This legacy software can be valuable, and CISC architectures are designed to support a wide range of older software without requiring significant modifications or

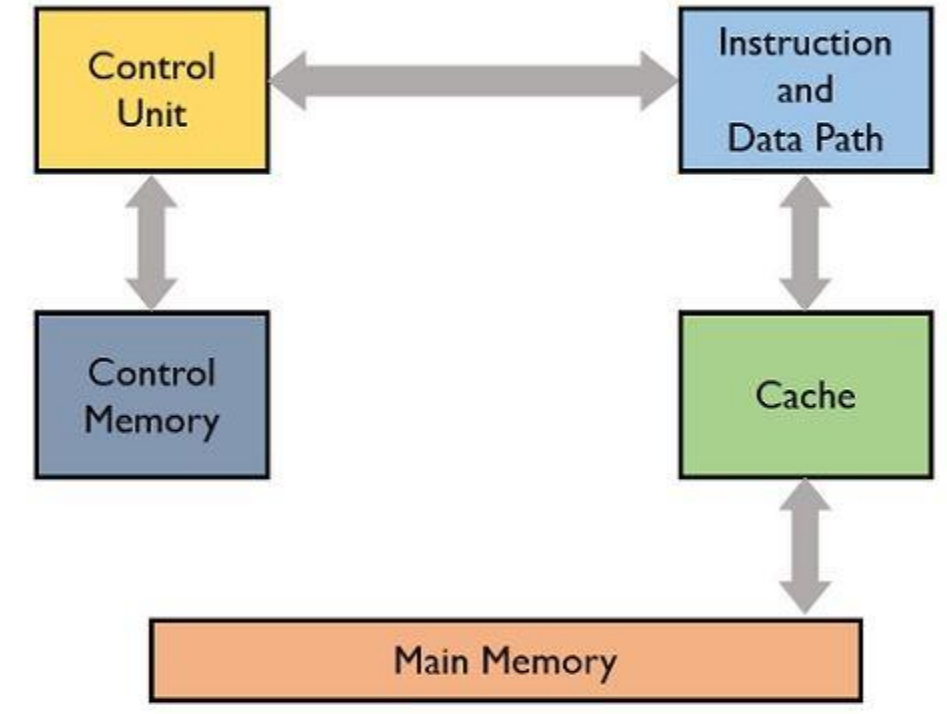
adaptations. This compatibility can be advantageous in situations where maintaining compatibility with existing software is critical.

5.Microcode and Hardware Abstraction:

CISC architectures often employ microcode, a layer of hardware abstraction that translates complex instructions into a sequence of simpler microoperations. This allows the processor's internal architecture to be more flexible and abstract, enabling manufacturers to make changes to the microcode to improve performance or fix bugs without altering the external instruction set.

6.Trade-Offs:

While CISC architectures offer several advantages, they are not without trade-offs. The complexity of the instruction set can lead to longer instruction decoding times and potentially slower overall execution for certain types of programs. Additionally, the larger number of possible instructions can complicate the design of control circuitry within the processor, which might impact clock speeds and overall efficiency.



CISC Architecture

RISC Architecture (Reduced Instruction Set Computer):

RISC architecture is a type of computer architecture that emphasizes simplicity and efficiency in the design of the instruction set. The key characteristics of RISC architecture include:

Properties of RISC:

Here are some additional properties and considerations related to Reduced Instruction Set Computer (RISC) architectures:

1.Simplicity and Predictability:

RISC architectures are designed with simplicity in mind. The reduced and straightforward instruction set leads to simpler hardware design, easier debugging, and better predictability of instruction execution time. This predictability is crucial for real-time systems and applications where consistent performance is required.

2.Compiler Optimizations:

RISC architectures often rely on optimizing compilers to take advantage of their simple instruction set. Compiler optimizations can rearrange instructions, eliminate redundant operations, and utilize the available registers effectively, leading to more efficient code execution.

3.Faster Clock Speeds:

The simplified instruction set of RISC architectures allows for faster clock speeds compared to CISC architectures. Since each instruction is designed to execute in a single clock cycle, the processor can potentially achieve higher clock frequencies, resulting in faster overall execution of programs.

4.Memory Efficiency:

RISC architectures tend to promote memory efficiency due to their load-store architecture. Data must be loaded from memory into registers before operations, reducing the need for memory access during computation. This design can lead to reduced memory traffic and faster execution.

5.Compiler-Driven Operations:

In RISC architectures, complex operations are often achieved through sequences of simple instructions. While this may seem less intuitive for programmers, it allows compilers to have greater control over optimizing instruction sequences, which can lead to more efficient execution.

6.Parallelism and Pipelining:

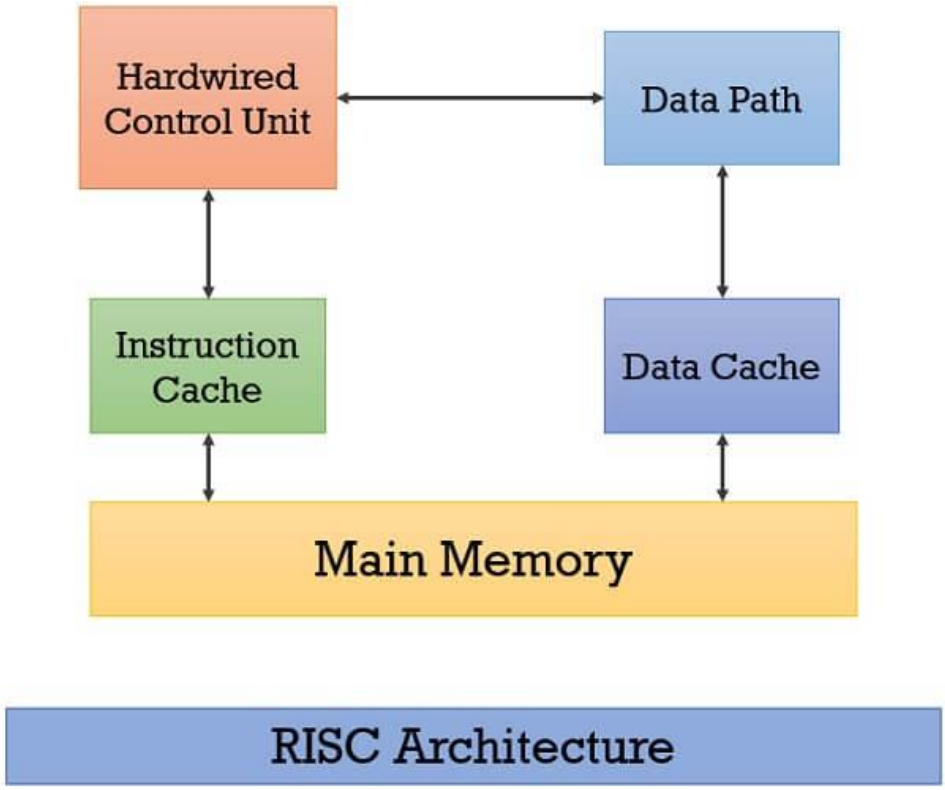
RISC architectures are well-suited for pipelining, a technique that divides the instruction execution process into stages. Each stage can handle a different instruction, enabling multiple instructions to be executed simultaneously. This parallelism enhances throughput and performance.

7.Code Portability:

Due to their simpler design and instruction set, RISC architectures are generally more portable across different processor implementations and architectures. This portability can be advantageous when designing software for a variety of platforms.

8.Energy Efficiency:

RISC architectures' focus on executing simple instructions efficiently and minimizing memory access contributes to their energy efficiency. Reduced memory access translates to lower power consumption, making RISC architectures suitable for battery-powered devices and energy-conscious applications.



Detailed comparison between RISC (Reduced Instruction Set Computer) and CISC (Complex Instruction Set Computer) architectures:

1. Instruction Set:

- RISC: RISC architectures have a smaller and simpler instruction set, consisting of basic instructions that perform simple operations. Each instruction typically takes one clock cycle to execute.
- CISC: CISC architectures offer a larger and more diverse instruction set, including complex instructions that can perform multi-step operations in a single instruction.

2. Instruction Length:

- RISC: RISC instructions are usually of fixed length, simplifying the decoding process and enabling faster instruction fetching.
- CISC: CISC instructions can vary in length due to the complexity of the operations they perform.

3. Memory Access:

- RISC: RISC architectures often follow a load-store architecture, where data must be loaded from memory into registers before operations can be performed. This reduces memory access requirements.
- CISC: CISC architectures allow instructions to directly access memory, potentially combining memory access and operations in a single instruction.

4. Execution Time:

- RISC: RISC architectures tend to have shorter execution times for individual instructions due to their simplicity and focus on executing basic operations quickly.
- CISC: CISC architectures can have longer execution times for complex instructions that perform multiple operations in a single instruction.

5. Pipelining:

- RISC: RISC architectures are well-suited for pipelining, as their simple instructions allow for smooth division of instruction execution into pipeline stages.
- CISC: While pipelining is possible in CISC architectures, the presence of complex instructions can make it more challenging to maintain a balanced pipeline.

6. Compiler Dependency:

- RISC: RISC architectures rely on intelligent compilers to optimize code, as optimization centers around instruction sequencing and efficient use of registers.
- CISC: CISC architectures offer more opportunities for compiler optimizations within individual instructions due to their complex nature.

7. Code Size:

- RISC: RISC instructions are generally shorter, resulting in more compact code.
- CISC: CISC instructions can be longer due to their complexity, potentially leading to larger code size.

8. Energy Efficiency:

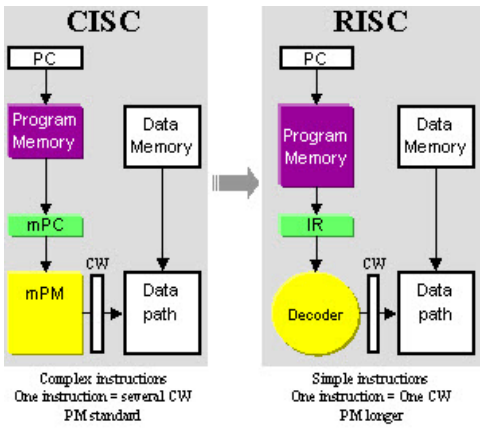
- RISC: RISC architectures tend to be more energy-efficient due to their emphasis on executing simple instructions quickly and minimizing memory access.
- CISC: CISC architectures might consume more energy due to the execution of complex instructions and potential memory access.

9. Compatibility:

- RISC: RISC architectures are often designed with a focus on forward compatibility, making them more adaptable to future changes in technology.
- CISC: CISC architectures may prioritize backward compatibility to maintain support for legacy software and instruction sets.

10. Application Focus:

- RISC: RISC architectures are suitable for applications that require fast and efficient execution of simple instructions, such as embedded systems, real-time processing, and high-performance computing.
- CISC: CISC architectures are well-suited for applications that require complex operations within a single instruction, such as multimedia processing, scientific simulations, and legacy software support.



Conclusion:

In this paper we briefly explain RISC and CISC architectures on the bases of their properties and explain their advantages. Now Due to this exploration both architectures RISC and CISC have continuously developed. The RISC architecture had advantages that the results to a machines excellent performance and adopted for commercial products. We also talk little bit about the performance problem provide idea for Researchers to further explain the Risc and Cisc and to Clarify the new Processor based on Risc and Cisc. The choice between RISC and CISC architectures depends on the specific requirements of the application and the balance between simplicity, performance, and energy efficiency. RISC architectures excel in streamlined execution and energy conservation, while CISC architectures offer versatility and the ability to perform complex operations in fewer instructions. Both architectures continue to shape the design and development of modern processors, impacting various aspects of computer engineering and software development.

References:

- https://ijcat.com/archives/volume5/issue7/ijcatr05071015.pdf
- Hennessy, J. L., & Patterson, D. A. (2017). Computer Architecture: A Quantitative Approach. Morgan Kaufmann.
- Tanenbaum, A. S., & Goodman, J. (2014). Structured Computer Organization. Pearson.
- Stallings, W. (2015). Computer Organization and Architecture. Pearson.
- Patterson, D. A., & Hennessy, J. L. (2004). Computer Organization and Design: The Hardware/Software Interface. Morgan Kaufmann.
- Flynn, M. J. (1995). Computer Architecture: Pipelined and Parallel Processor Design. Jones & Bartlett Learning.
- https://binaryterms.com/risc-processor.html
- https://electronicsdesk.com/cisc-processor.html