INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 4, Issue 6)
Available online at: www.ijariit.com

The survey of concepts of architecture in RISC and CISC computers

Muskan Saxena
muskan.saxena2017@vitbhopal.ac.in
VIT Bhopal University, Bhopal, Madhya Pradesh

Ojaswini Nimbalkar
ojaswini.nimbalkar2017@vitbhopal.ac.in
VIT Bhopal University, Bhopal, Madhya Pradesh

Vidhi Jaiswal
vidhi.jaiswal2017@vitbhopal.ac.in
VIT Bhopal University, Bhopal, Madhya Pradesh

Vishakha Pandey
vishakha.pandey2017@vitbhopal.ac.in
VIT Bhopal University, Bhopal, Madhya Pradesh

P. Sanjeevi
sanjeevi.p@vitbhopal.ac.in
VIT Bhopal University, Bhopal, Madhya Pradesh

ABSTRACT

In the ever-growing world of computer architecture, Instruction Set Architecture (ISA) is one of the major components of a computer system, as it provides the information about the instructions present in the system prior to the programmer. Having various implementations and uses it is classified into various categories, RISC (Reduced Instruction Set Computer) and CISC (Complex Instruction Set Computer) being two, both being a bit different in their basic architecture and working. Both CISC and RISC architectures continue to be widely used. The research topic on CISC and RISC has been a well-known research area for many years. Since modern processors have to address both power consumption and performance, it is important to compare these architectures to support future project decisions. Our paper also includes basic details regarding the various addressing modes, instruction formats along with the instruction execution cycle giving detailed information about RISC and CISC processors simultaneously.

Keywords— *RISC (Reduced Instruction Set Computer), CISC (Complex Instruction Set Computer), Instruction execution cycle, ISA (Instruction Set Architecture)*

1. INTRODUCTION

Instruction Set Architecture (ISA) separates the hardware and software components of a computer system. An ISA is considered similar to human language, it is the language used by processors. An ISA lists all the instructions user can give. Registers, addressing modes, data-types, memory management, devices and exception handling, power management, and multi-threading support serve as components of ISA. Implementation of an ISA comes under micro-architecture. Multiple implementations can be made for a single ISA, and hence it can have various micro-architectures. ISAs are classified into RISC (Reduced Instruction Set Computer) and CISC (Complex Instruction Set Computer) [1].

A basic set of operations is specified by the instruction set. Any instruction from the instruction set commands an operation that paves way for altering the machine state, data, or performing an input-output operation.

Arithmetic and logical operations are performed by the instructions from the first category. Operations valid can be string, logical, arithmetic, or floating-point. Appropriate functional units perform the operations of the particular implementation of the architecture. Branches, calls and returns, and loop control instructions affect the flow of the program and machine state. Instructions performing data movement across different functional units of the machine belong to the third category.

Examples are LOAD instruction that helps to load the content of a memory location to a register in particular or STORE instruction that functions exactly the opposite. Then there's an instruction that moves a block of data from a memory location to another, it is called MOVE [2].

A short vivid description of an ISA can be seen in the following figure 1 below:

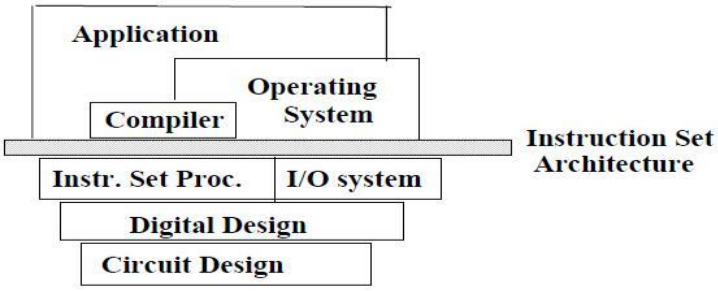

Fig. 1: Instruction Set Architecture [3]

No standard computer architecture is there which accepts different types like CISC, RISC, etc [3]

While talking about the operation achieved, we can broadly categorize the instructions of the instruction set in the four general types:

1. Instructions performing transformation of data
2. Instructions altering the program flow
3. Instruction performing data movement
4. System instructions [2]

1.1 Reduced Instruction Set Computer (RISC)

Nowadays, modern CPUs are mostly of the GPR (General Purpose Register) type. For example IBM 360, DEC VAX, Intel 80x86 and Motorola 68xxx. But while these CPUs were clearly better than previous stack and accumulator-based CPUs they were still lacking in several areas, one out of many examples is: Instruction length varies from 1 byte to 6-8 bytes. This caused problems with the management of instructions such as pipelining and pre-fetching of instructions. Thus, the idea of RISC was introduced [4]. A reduced instruction set computer architecture (RISC) termed as MIPS is considered as the original RISC Instruction Set Architecture.

A phrase MIPS is for "Microprocessor without interlocked pipeline stages". At Stanford University John Hennessey led a team that developed this architecture. Implementations of MIPS are fundamentally used in embedded systems, for example, residential gateways, video games consoles (such as PlayStation Portable, Sony PlayStation 2) routers, Windows CE devices. Also, in many SGI's computer products they were being used till late 2006. Considering MIPS as RISC computer there is a decrease in the transistor count along with a lesser number of transistors employment. Thus, here we have a heavily employed pipelining to produce an extra use of available space on the chip in order to enhance the code execution performance. MIPS was elucidated as a 32-bit architecture termed as MIPS32 **[**5**].**

1.2 Characteristics of RISC

(a) RISC processor per instruction's (CPI) of a single sequence is clocked which is available for optimization of each instruction on the microprocessor.
(b) To avoid dealings with memory, a large number of registers usually join.
(c) Instructions which change the movement of controls such as branch's instruction which is executed well since they compress about 20 to 30% of distinctive programs.
(d) infrequently used instructions are not attempted to implement by RISC [6]**.**

Why this architecture is called RISC? What is reduced about it? The number of bits that are used for the opcode is reduced to make all instructions the same length. Hence, fewer instructions are provided. The instructions that were thrown out are the least significant ones. Since memory access is restricted therefore there aren't several kinds of ADD or MOV instructions. Hence, the older architecture is called CISC (Complete Instruction Set Computer). Code size is the only disadvantage of RISC. LOAD/STORE architectures are same as RISC architectures. Usually, RISC has 32 or number of registers for use. The MIPS 2000 (The first RISC CPU) has 32 GPRs as compared to 8 in the 80x86 architecture and 16 in the 68xxx architecture. The simplest reason why CISC CPUs are being developed is that they have backward compatibility [7].

1.3 Comparison of RISC and CISC

Table 1: Comparison of RISC CISC

Properties of CISC	Properties of RISC
1. CISC instructions take more than 1 clock per cycle to get executed. 2. It has complex instructions. 3. Instructions are of variable length. 4. The emphasis of Hardware. 5. Uses very few registers. 6. Not used as a load and store machine. 7. Not so good in term of speed for compilation. 8. No pipelining. 9. For storing complex instructions, transistors are used.	1. RISC Instructions are executed in one clock cycle per instruction. 2. It has simple instructions. 3. All instructions have the same length. 4. Emphasis on software 5. Uses more registers than CISC. 6. Load and Store architecture implemented. 7. Optimal compilation speed as compared to CISC. 8. It has Pipelining 9. As compared to CISC a RISC spends more transistors on memory registers [7].

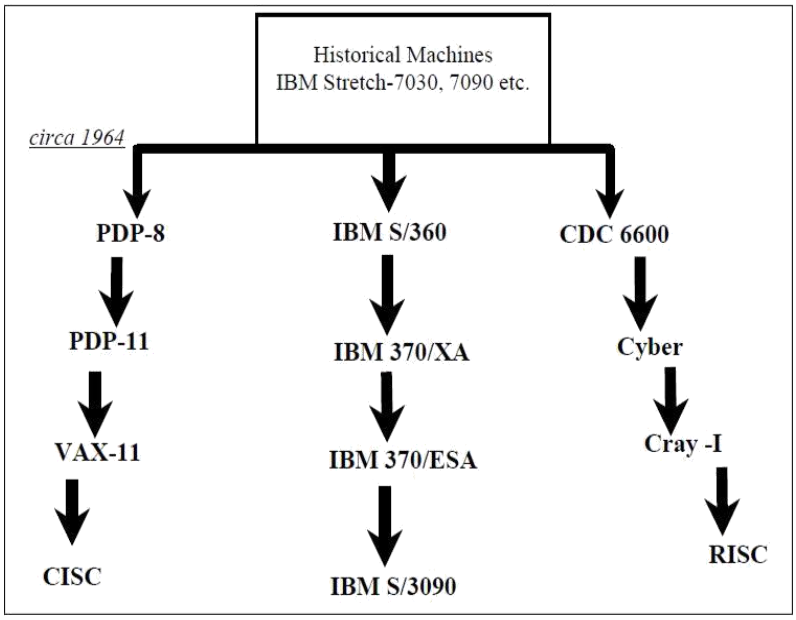

Fig. 2: Main Branches in Development of Computer Architecture [8]

The hierarchy of the various branches in the development history of computer architecture as shown in the above figure describes how and after which IBM machine does the RISC and CISC architecture evolved.

Other types of ISAs include Very Long Instruction Word (VLIW) architectures, and the *Long Instruction Word* (LIW) and explicitly parallel instruction computing (EPIC) architectures. Instruction-level parallelism is looked to exploit by these architectures with less hardware than RISC and CISC by making the compiler responsible for scheduling and instruction issue. Examples of VLIW architectures are Intel's Itanium processor [1] and STMicroelectronics' ST1001 and ST200 processor families [22].

High performance is made easy with simpler VLIW implementations. RISC architectures permit simpler, cheaper high-performance implementations than CISC architectures, also VLIW architectures are simpler and cheaper than RISCs because of further hardware simplifications. However, more compiler support is required by VLIW architectures [21].

Some CPUs have been specifically designed to have a very small set of instructions – but since these designs are very different from classic RISC designs, they have been given other names such as minimal instruction set computer (MISC) [9].

1.4 Modern ISAs

Intel in its x86 chips has retained CISC architecture, but changes in hardware implementations over the years have made the line between RISC and CISC less clear and has various impact on cloud data center [23-32]. Features of both approaches are exhibited by many chips. MIPS (Microprocessor without Interlocked Pipeline Stages) processors are RISC chips often used in embedded systems like video game consoles and ARM (Advanced RISC Machine) processors are ever-present in mobile devices [10].

2. ADDRESSING MODES

Addressing modes can be seen as the way in which in an instruction operand are quantified [20].

The location of the operand is an important parameter on which the way of formation of the operand address depends along with the choices present in the instruction architecture. In the case of accumulator architecture or stack, there is no need of citing the operand address as it is already entailed. If it belongs to the General-Purpose Register (GPR) then the address of that particular register is already intact in the operand field in the instruction. So, this mode is known as register direct addressing and is among the easiest ways of citing the location of the operand. When the operand is being comprised within the instruction then the addressing of the operand becomes even simpler and is known as immediate addressing mode. The pointed location by the addressed formed comprises in the operand field of the instruction that is capable of containing the address of the operand or the operand itself [2].

2.1 Instruction format

Instruction formats can be categorized into five formats that are:

- Byte-oriented operations
- Byte-to-Byte operations
- Bit-oriented file register operations
- Literal operation Control operations
- Byte-oriented operations [20]

Byte-oriented operations: A 16-bit length instruction is divided in such a way that 8 bits are utilized to cite address of the operand, one bit to cite which bank to be operated, another bit to cite destination and 6 bits that are remaining are used to cite the Opcode of the instruction.

Byte-to-Byte operations: movff f1, f2 is the only instruction that uses this format. Being 32 bit long it comprises of two 12 bit address and 4 bit lengthy Opcode.

Bit-oriented file register operation: This format utilizes 8 bits to cite the file register address, 3 bits to cite the bit position, one bit to cite the bank to be employed, and enduring 4 bits to cite the Opcode.

Literal Operations: These instructions basically cites the value of operand as it is instead of their address. So, these kinds of instructions comprise 8-bit for operand value and 8-bit for Opcode.

Control operations: They are the offshoot instructions which contain BC (Conditional Short jump), Bra (Unconditional short jump), Call and Goto (Unconditional long jump) [20]

3. INSTRUCTION EXECUTION

Every computer performs a very basic function i.e., the execution of a program, where every executable program is basically a vast collection of instructions which are present in the memory. In order to complete a specific task, the instructions of the program are executed by the CPU (Central Processing Unit). The CPU has the prime responsibility of executing every instruction and every execution takes place in the CPU registers [11]. It executes the programs that are hoarded in the main memory by fetching their instructions, thereby investigating them, and executing them in a sequential manner [12].

The following are the registers that help the CPU in executing the instructions:

- MAR (Memory Address Register): The addresses of the memory locations that are being catalogued by this register specifies the instruction or data to be retrieved from the memory or to be stored in the memory.
- PC (Program Counter): It keeps the course of the next executable instruction, after the accomplishment of the current instruction.
- IR (Instruction Register): Before the implementation all the instructions are being loaded in here.

The most elementary model of instruction processing can be seen in the form of two-step process i.e., fetch (where the CPU one at a time reads the instructions from the memory) and perform or execute the described operation. Instruction fetch requires the reading of an instruction from the specified memory location described in the register, whereas the execution of the same instructions involves various operations depending on the complexion of the instruction **[**11].

Instruction fetch has basically evolved from collecting one instruction per cycle to few cycles, to collecting several basic blocks per cycle, to a full basic block per cycle: the advancement of the mechanism surrounding the instruction cache, along with that different compiler optimizations are used to utilize these mechanisms in a better way **[**13].

The required processing for a single instruction (fetch and execute) is broached as an instruction cycle, consisting of two further divisions i.e., fetch cycle and execute cycle.

Only in the cases like discontinued electric supply or some sort of unrecoverable error, the program execution gets terminated or sometimes because of the program itself [11].

Fetch -decode- execute cycle, in simple words known as the instruction cycle is the elementary operational process of a computer. Starting right from boot up to shut down of the computer the following process of fetch-decode-execute is repeated continuously [14]

There five stages that fragment the processor i.e., instruction fetch, instruction decode, execution, data memory and write back. The operations performed at various stages are being controlled by the control unit [12].

Instruction cycle's each phase can be further fragmented into a series of fundamental micro-operations. The instruction cycle occurs with the help of the following steps:

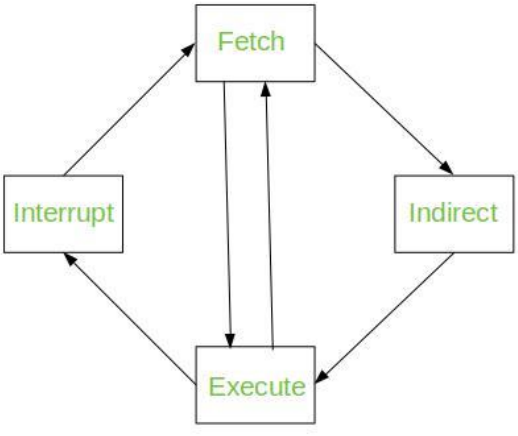

Fig. 3: Instruction execution cycle [15]

- **Fetching of the instruction:** With the help of the memory address stored in the Program Counter (PC) is used to fetch the instruction and then store the same in the instruction register (IR). When the fetch operation gets finished, an increment of 1 is done in the program counter which as a result points to the next executable instruction.
- **Decode the Instruction:** The decoder executes the instructions present in the IR (Instruction Register).
- **Read the Effective Address:** In case of any indirect address for an instruction, the memory provides the effective address to be read. Otherwise, in case of immediate operand instruction, operands are directly read.
- **Execute the Instruction:** The information is being passed by the control unit in the form of control signals to the functional unit of the CPU. The generated result is further stored in the main memory or is being sent to the respective output devices.

The interrupt cycle (figure 3) is followed by the fetch cycle whereas the indirect cycle is being followed by the execute cycle. The next cycle for both fetch and execute cycle depends on the state of the system.

3.1 Fetch Cycle

There are three simple steps and four micro-operations that a fetch cycle consists.
MAR← PC
MBR← MEMORY
PC← (PC)+1
IR← (MBR)

3.2 Indirect Cycle

With the help of indirect addressing the source operand is being fetched. There are three micro-operations present in it.
MAR← (IR(ADDRESS))
MBR← MEMORY
IR(ADDRESS) ← (MBR(ADDRESS))

3.3 Execute Cycle

Execute cycle is different from fetch, indirect and interrupt, as each requires a small, simple and fixed sequence of operations whereas for a machine having N different opcodes there will be present N different sequence of micro-operations that can occur for the execute cycle.

3.4 Interrupt Cycle

In order to determine whether any enabled interrupt has occurred or not a test after the execute cycle is being done. If an interrupt comes across in the test then the following cycle comes into the act, which varies from machine to machine. There are three micro-operations present.
MBR← PC
MAR← SAVE_ADDRESS,
PC← ROUTINE_ADDRESS
MEMORY← (MBR) [15]

In RISC architecture instructions must be executed as one instruction per cycle so that the 1st instruction is as fast as it can and not more complicated than the microinstructions in the current machines [16].

4. CONCLUSION

The result of our survey specifies that ISAs can highly affect performance and measure the effect, that varies based on the microarchitecture. Phases of execution are commonly exhibited by the programs, which can be more affine to one ISA than the other. RISC has been inspected regarding its design and in addition, to give details about how it operates. In the future, ISAs will have a major impact on diverse types of workloads such as the ones related to server and cloud computing, cryptography, artificial intelligence, multimedia, and ultra-low power systems.

Hence, keeping in mind, today's paced-up technological world, Instruction Set Architecture plays a very important role

5. REFERENCES

[1] Ayaz Akram, A Study on the Impact of Instruction Set Architectures on Processor's Performance
[2] Vojin G. Oklobdzija: Advanced Computer System Engineering Laboratory Electrical and Computer Engineering Department, University of California, Davis, CA 95616
[3] Difference between RISC and CISC Architecturehttp://firmcode.blogspot.com/2015/01/difference-between-risc-and cisc.html
[4] Sagar Bhavsar, Akhil Rao, Abhishek Sen, Rohan Joshi, International Journal of Scientific and Research Publications, Volume 3, Issue 4, April 2013
[5] AndrewObidaYoila, "A 64-Bit RISC Processor Design and Implementation Using VHDL"The International Journal Of Science & Technoledge.
[6] Shahzeb1, Naveed Hussain1, Amanulllah1, Furqan Ahmad1, Salman Khan, Department of Computer Science, Abdul Wali Khan University MardanInternational Journal of Computer Applications Technology and ResearchVolume 5 issue 7, 500 – 503, 2016
[7] https://www.semanticscholar.org/paper/Reduced-Instruction-Set-Computers-Prof-Oklobdzija/ce73fe0372f5477c228c58c4dc1b93232ca0bb2f

[8] International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Volume: 4 Issue: 2, Feb - 2017 www.irjet.net p-ISSN: 2395-0072, 2017, IRJET, "32-Bit RISC Processor With Floating Point Arithmetic" Mrudul S. Ghaturle, Prof. R. D. Kadam, E&T Department, BDCOE, Sevagram,Maharashtra,India1 Assistant Professor, E&T Department, BDCOE, Sevagram,Maharashtra,India2
[9] https://study.com/academy/lesson/examples-of-instruction-set-architectures.html
[10] https://webservices.ignou.ac.in/virtualcampus/adit/course/cst101/block1/unit1/cst101-bl1-u1-18.htm
[11] Control Unit Design of a 16-bit Processor Using VHDL Alpesh Kumar Dauda* Nalinikanta Barpanda Nilamani Bhoi Deptt. Of El&TC Engg. Deptt. Of AEl&IE Engg. Deptt. Of El&TC Engg. Place, Bargarh, Odisha G.I.E.T, Gunupur Vssut, Burla, Odisha India India India Manoranjan Pradhan Deptt. Of El&TC Engg Vssut, Burla, Odisha India
[12] A. Ramirez ; J.L. Larriba-Pey ; M. Valero; Instruction fetch architectures and code layout optimizations
[13] https://www.studytonight.com/computer-architecture/addressingmodes-instruction cycle
[14] https://www.geeksforgeeks.org/different-instruction-cycles/
[15] David A. Patterson And Carlo H. Sequin, Computer Science Division University of California Berkeley, California, RISCI: A Reduced Instruction Set VLSI Computer
[16] https://www.thefreedictionary.com/instruction+execution
[17] Priyavrat Bhardwaj1, Siddharth Murugesan: Design & Simulation Of A 32-Bit RISC Based MIPS Processor Using Verilog ; IJRET: International Journal of Research in Engineering and Technology, eISSN: 2319-1163 | pISSN: 2321-7308 ; Volume: 05 Issue: 11 | Nov-2016, Available @ https://ijret.org 166 ,
[18] Dr. N. Dulay, Imperial College London Department of Computing, CPU Organisation, and Operation PIC 18 Peripheral Interface Controller Pramathesh Trivedi Third Year Electronics Engineering, Atharva College of Engineering, Mumbai, Maharashtra, India
[19] An Introduction to Very-Long Instruction Word (VLIW) Computer Architecture VLIW – A Case Study of Parallelism Verification; Allon Adir, Yaron Arbetman, Bella Dubrov, Yossi Lichtenstein, Michal Rimon, Michael Vinov {adir, Arbetman, bella, yossil, Michael, vinov@il.ibm.com} IBM Research Laboratory Haifa, Israel Massimo
[20] P. Sanjeevi and P. Viswanathan, 'NUTS scheduling approach for energy cloud data centers to optimize consumption', *Computing (Springer)*, Vol. 99, No. 12, pp. 1179-1205, 2017.
[21] P. Sanjeevi and P. Viswanathan, 'Workload Consolidation Techniques to Optimize Energy in *Internet Protocol Technology*, Vol. 10, No. 2, pp. 115–125, 2017.
[22] P. Sanjeevi and P. Viswanathan, 'DTCF: Deadline Task Consolidation First for energy minimization in cloud data centers', *International Journal of Networking and Virtual Organizations*, Inderscience, Vol. 19, No. 3, pp. 209–233, 2017.
[23] P. Sanjeevi and P. Viswanathan, 'A survey on various problems and techniques for optimizing energy efficiency in cloud architecture', *Walailak Journal of Science and Technology*, Vol. 14, No. 10, 2017.
[24] P. Sanjeevi and P. Viswanathan, 'Employing Smart Homes IoT Techniques for Dynamic Provision of Cloud Benefactors', *Int. J. of Critical Computer-Based Systems*, Inderscience, Vol. 7, No. 3, pp. 209–224, 2017.
[25] P. Sanjeevi, P. Viswanathan, M. R. Babu, and P. V. Krishna, 'Study and Analysis of Energy Issues in Cloud Computing', *International Journal of Applied Engineering Research*, Vol. 10, No. 7, pp. 16961-16969, 2015.
[26] . Sanjeevi, G. Balamurugan and P. Viswanathan, 'The Improved DROP Security based on Hard AI Problem in Cloud', *Int. J. of Internet Protocol Technology*, Vol. 9, No. 4, pp. 207-217, 2017.
[27] P. Sanjeevi and P. Viswanathan, 'A green energy optimized scheduling algorithm for cloud data centers', *IEEE International Conference on Computing and Network Communications, Trivandrum*, pp. 941-945, 2015.
[28] [P. Sanjeevi and P. Viswanathan, 'Towards energy-aware job consolidation scheduling in cloud', *International Conference on Inventive Computation Technologies (ICICT 2016), IEEE Xplore*, pp. 361- 366, 2016.
[29] G. Kesavan, P. Sanjeevi and P. Viswanathan, 'A 24 hour IoT framework for monitoring and managing home automation'.

APPENDIX

ISA	: In order to program a computer, a machine language programmer should have prior knowledge, which is given by Instruction Set Architecture.
Instruction Set	: Set of all instructions recognized by the processor.
CISC	: An architectural design that has the ability to execute addressing modes and multi-step operations within a single instruction set.
RISC	: It is a microprocessor that performs a small number of types of computer instructions.
Register	: It is a data holding place.
Addressing mode:	: Architectures specify the address using addressing modes.
Stack	: It serves as a collection of elements.
Accumulator	: It is a short-term register that stores arithmetic and logical data.
Instruction Execution	: Instructions are executed by getting decoded by the CPU. [17]
MIPS	: Microprocessor without Interlocked Pipeline Stages. [18]
GPR	: General purpose register. [19]
Instruction Pipelining:	: Pipelining is fetching of next instruction while the current instruction is still being executed[20]

