# A Comparative Study on Computer Architecture

Yash Jain [#1],
Ritik Kumar [#2],
Shakti Kundu [#3]
[#] *College of Computing Sciences and Information Technology,*
*Teerthanker Mahaveer University, Moradabad, UP, India.*
[1] yashlovingly@gmail.com
[2] rajr32200@gmail.com
[3] shaktikundu@gmail.com

***Abstract*** --- modern results in parallel computing validate that highly parallel, general-purpose shared memory computers can be built in principle. The architectural development tracks follow no simple predictable path. Current computer architecture research continues to have a preconceived notion for the past in that it focuses on desktop and server applications. In our view, a different computing domain personal mobile computing will play a significant role in dynamic technology in the next decade. This domain will create a different set of requirements for microprocessors and could redirect the prominence of computer architecture research. This paper gives a brief overview and comparative study about some architecture that has been recently emerged.

***Keywords:*** Instruction Set Architecture Design, Micro-Architecture Design, Logic Design, Implementation.

## 1. INTRODUCTION

Although the term computer architecture sounds very complex, its definition is easier than one might think. Computer architecture is a science or a set of rules stating how computer software and hardware are joined together and relate to make a computer work. It not only determines how the computer works but also of which technologies the computer is capable.

Computers continue to be a major part of our lives, and computer architects continue to develop new and enhanced programs and technologies.

Computer architecture is a arrangement describing how hardware and software technologies interact to create a computer platform or structure. When we think of the word architecture, we think of building a house or a building. Keeping that same principle in mind, computer architecture involves building a computer and all that goes into a computer system. Computer architecture consists of three main categories.

- *System Design* – it includes all the hardware parts, such as data processors, CPU, memory controllers ,multiprocessors and (DMA) direct memory access. This part is the actual computer system.
- *Instruction Set Architecture* – it includes the CPU's functions and abilities, the CPU's programming language, data formats, processor register types and instructions used by computer programmers. This part is the software that makes it runs, such as Windows or Photoshop or like programs.
- *Micro-Architecture* – it describes the data processing and storage element or data paths and how they should be enforced into the instruction set architecture. These may comprise of DVD storage devices or similar devices.

All these parts go together in a definite order and must be developed in a pattern so they will function correctly.

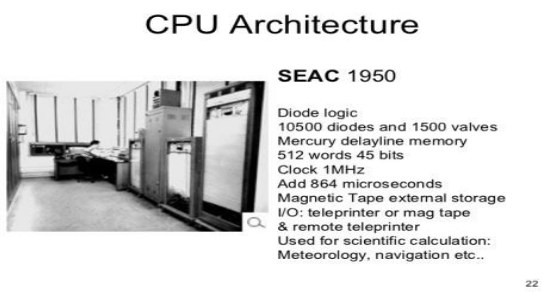
## 1. REVIEW LITERATURE

The first documented computer architecture was in the association between Charles Babbage and Ada

Lovelace, unfolding the analytical engine. When constructing the computer Z1 in 1936, Konrad Zuse described in two patent applications for his upcoming projects that machine instructions could be stored in the same storage used for data, i.e. the stored-program concept. Two other near the beginning and important examples are:

- John Von Neumann's 1945 paper, first draft of a report on the EDVAC, which gave a detail of an organization of logical elements; and
- Alan Turing's further detailed *proposed electronic calculator* for the automatic computing engine, also 1945 and which cited john von Neumann's paper.

The term "architecture" in computer literature can be sketched to the work of Lyle r. Johnson and Frederick p. Brooks, Jr., members of the machine organization department in IBM's major research center in 1959. Johnson had the chance to write a proprietary research communication about the stretch, an ibmdeveloped supercomputer for Los Alamos national laboratory (at the time known as Los Alamos scientific laboratory). To define the level of detail for discussing the splendidly blown up computer, he noted that his description of formats, instruction types, hardware parameters, and speed enhancements were at the level of "system architecture" – an expression that seemed more useful than "machine organization."



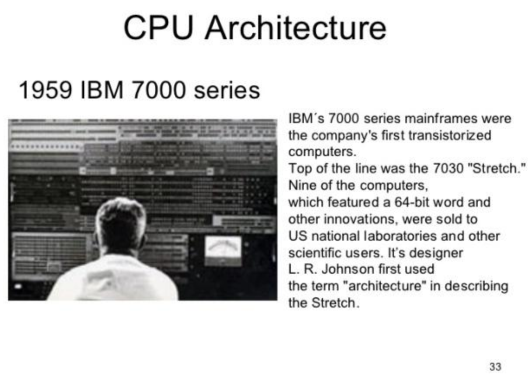




Figure 1: CPU Architecture

consequently, brooks, a stretch designer, started chapter 2 of a book (planning a computer system: project stretch, ed. W. Buchholz, 1962) by writing, Computer architecture, like other architecture, is the art of formatting the needs of the user of a structure and then designing to meet those needs as efficiently as possible within economic and technological constraints.

Brooks went on to help develop the IBM system/360 (now called the IBM z series) line of computers, in which "architecture" became a noun central "what the user needs to know". Later, computer users came to use the term in many less-certain ways.

The initial computer architectures were designed on paper and then straightforwardly built into the final hardware form. Later, computer architecture prototypes were manually built in the form of a transistor–transistor logic (TTL) computer such as the prototypes of the 6800 and the PA-RISC tested, and twisted, before committing to the final hardware form. As of the 1990s, new computer architectures are classically "built", tested, and twisted—inside some other computer architecture in a computer architecture simulator; or inside a FPGA as a soft microprocessor; or both before executing to the final hardware form.

## 2. INSTRUCTION SET ARCHITECTURE

An instruction set architecture (ISA) is an theoretical model of a computer. It is also referred to as architecture or computer architecture. A realization of an ISA is called an *implementation*. An ISA allows multiple implementations that may vary in performance, physical size, and financial cost (among other things); because the ISA serves as the interface between software and hardware. Software that has been written for an ISA can run on different implementations of the same ISA. This has facilitated binary compatibility between different generations of computers to be easily accomplished, and the development of computer families. Both of these developments have helped to lower the cost of computers and to enlarge their applicability. For these reasons, the ISA is one of the most important abstractions in computing today. An ISA defines everything about a machine language programmer needs to know in order to program a computer. What an ISA defines differs between ISAs; in universal, ISAs define the supported data types, what state there is (such as the main memory and registers) and their semantics (such as the memory consistency and addressing modes), the *instruction set* (the set of machine instructions that constitutes a computer's machine language), and the input/output model.

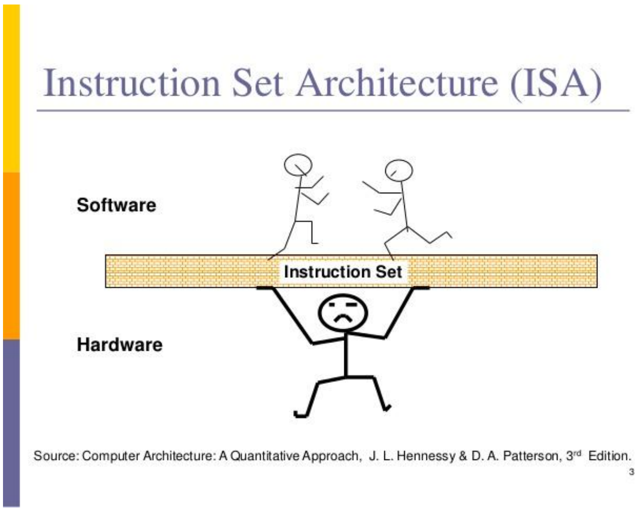



Figure 2: Instruction Set Architecture

## 3. COMPUTER ORGANIATION

In computer engineering, micro architecture, also called computer organization and at times condensed as *µarch* , is the way a given instruction set architecture (ISA)is implemented in a particular processor. A given ISA may be implemented with different micro architectures; implementations may differ due to different goals of a given design or due to changes in technology .

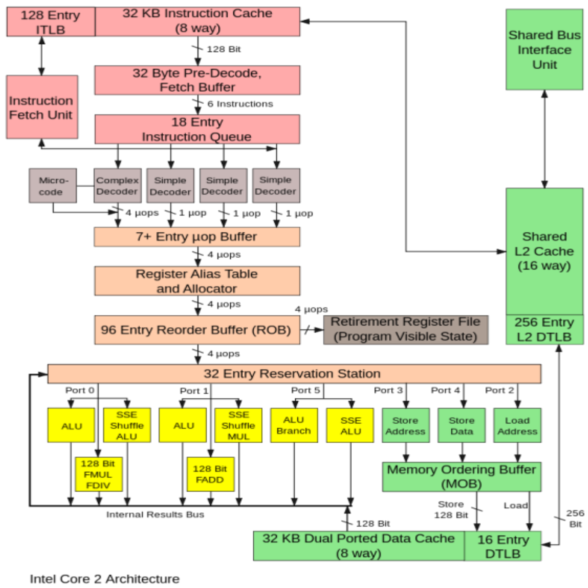



Figure 3: Intel Core 2 Architecture

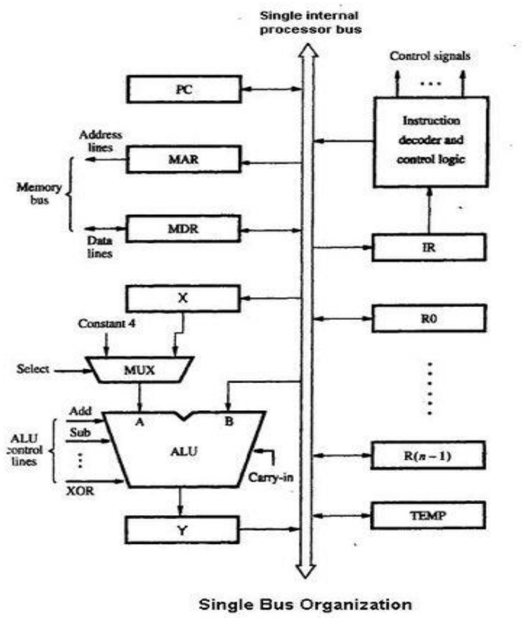



Figure 4: Single Bus Organization

## 4. IMPLEMENTATION

In computer science, an implementation is a insight of a technical specification or algorithm as a program, software component, or other computer system through computer programming and deployment.

Many implementations may exist for a given specification or standard. For example, web browsers enclose implementations of World Wide Web consortium-recommended specifications, and software development tools include implementations of programming languages. A special case occurs in object-oriented programming, when a concrete class implements an interface; in this case the concrete class is an *implementation* of the interface and it includes methods which are *implementations* of those methods particular by the interface**.** Once an instruction set and micro-architecture are designed, a practical machine must be developed. This design process is called the *implementation*. Implementation is usually not measured architectural design, but rather hardware design engineering. Implementation can be further broken down into fairly a few steps:

- Logic Implementation designs the circuits required at a logic gate level
- Circuit Implementation does transistor-level designs of basic elements (gates, multiplexers, latches etc.) As well as of some bigger blocks (alus, caches etc.) That may be implemented at the log gate level or even at the physical level if the design calls for it.
- Physical Implementation draws physical circuits. The various circuit components are positioned in a chip floor plan or on a board and the wires connecting them are formed.
- Design Validation tests the computer as a whole to see if it works in all conditions and all timings. Once the design validation process starts, the design at the logic level are tested using logic emulators. However, this is generally too slow to run practical test. So, after making corrections based on the first test, prototypes are constructed using Field-Programmable Gate-Arrays (fpgas). Most hobby projects stop at this phase. The final step is to test prototype integrated circuits. Integrated circuits may call for several redesigns to fix problems.

For CPUs, the complete implementation process is organized differently and is frequently referred to as CPU design.

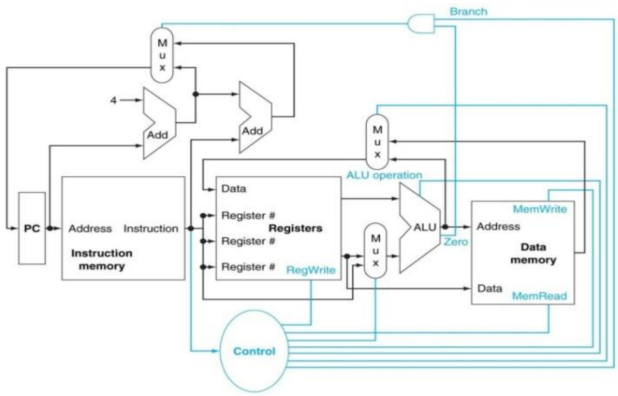



Figure 5: CPU Design Implementation

## 5. CASE STUDY- BILLION TRANSISTOR ARCHITECTURE

Presenting a Billion Transistors in the first two architectures in Computer's study—the advanced superscalar and super speculative have very similar characteristics. The fundamental idea is a wide superscalar organization with multiple execution units or functional cores. These architectures use multilevel caching and aggressive prediction of data, control, and even sequences of instructions (traces) to use all the existing instruction level parallelism (ILP). Due to their resemblance, we group them together and call them wide superscalar processors. The trace processor consists of multiple superscalar processing cores, each implementing a trace issued by a mutual instruction issue unit. It also employs trace and data prediction, and shared caches. The simultaneous multithreading (SMT) processor uses multithreading at the granularity of instruction issue slot to maximize the use of a wide-issue, out-of-order superscalar processor. It does so at the cost of extra complexity in the issue and control logic. The chip multiprocessor (CMP) uses the transistor budget by inserting a symmetric multiprocessor on a single die. There will be eight uniprocessors on the chip, all similar to present out-of-order processors. Each uniprocessor will have its own separate first level caches but share a large second-level cache and the main memory interface. IA-64 can be measured a recent commercial reincarnation of architectures based on the very long instruction word (VLIW), now renamed clearly parallel instruction computing. Based on the information announced thus far, its major innovations are the instruction dependence information attached to each long instruction and the support for bundling multiple long instructions. These changes attack the problem of scaling and low code density that repeatedly accompanied older VLIW machines. IA-64 also includes hardware checks for hazards and interlocks, which helps to preserve binary compatibility across chip generations. Finally, it supports predicated execution through general function predication registers, which reduces control hazards. The Raw machine is probably the most world-shattering architecture proposed, as it incorporates reconfigurable logic for general-purpose computing. The processor consists of 128 tiles, each of which has a processing core, small first-level caches backed by a superior amount of dynamic memory (128 Kbytes) used as main memory, and a reconfigurable functional unit. The tiles interrelate in a matrix approach via a reconfigurable network. This design emphasizes the software infrastructure, compiler, and dynamic event support. This infrastructure controls the partitioning and mapping of programs on the tiles, plus the configuration selection, data routing, and scheduling.

## 6. CONCLUSIONS

Today, it is broadly recognized computer is really necessary for the whole world. Throughout the 21th century, many people were realizing the use of computer could help them successfully in a lot of things with easier or either way that's for calculation or management. So, as we know computer is a very helpful digital machine, but not everyone actually know how it is created by. Therefore in this research paper, we would discuss all the things about computer architecture.

As a conclusion, this research paper is about the function and structure of computer. The idea of this paper is to present as visibly and completely as possible, the characteristics and nature of modern-day computer system. Although most of the resources of this assignment are taken from internet and reference book, the objective is to present the material in a manner that keeps new material in a clear context to those readers.

## REFERENCES

[1] Clements, Alan. Principles of Computer Hardware (Fourth ed.)

[2] Hennessy, John; Patterson, David. Computer Architecture: A Quantitative Approach (Fifth Ed.)

"Electronic Digital Computers", *Nature, 162: 487, 25 September 1948.*

[3] Susanne Faber, "Konrad Zuses Bemuehungen um die Patentanmeldung der Z3", 2000.

[4] Neumann, John (1945). First Draft of a Report on the EDVAC.

[5] Reproduced in B. J. Copeland (Ed.), "Alan Turing's Automatic Computing Engine", OUP, 2005.

[6] Buchholz, Werner (1962). Planning a Computer System.

[7] John L. Hennessy and David A. Patterson. Computer Architecture: A Quantitative Approach (Third ed.). Morgan Kaufmann Publishers.

[8] A New Direction for Computer Architecture Research by Christoforos E. Kozyrakis David A. Patterson University of California, Berkeley.