World University of Bangladesh (WUB)

Corse Title : **Advanced Computer Organization & Architecture**
Course Code : **CSE 1101 (OLD)**
Submission Date : **19-08-2020**

40/B

From
Md. Imran Hossain
Roll No : **2400**
Batch No : **40(B)**
Semester : **8th**
Department of Computer Science & Engineering(CSE)

To
Md. Ashiqur Rahman
Asst. Professor
Department of CSE
World University of Bangladesh (WUB)

Tittle: The organization and architecture of computer-based systems and how a range of design choices are influenced by applications.

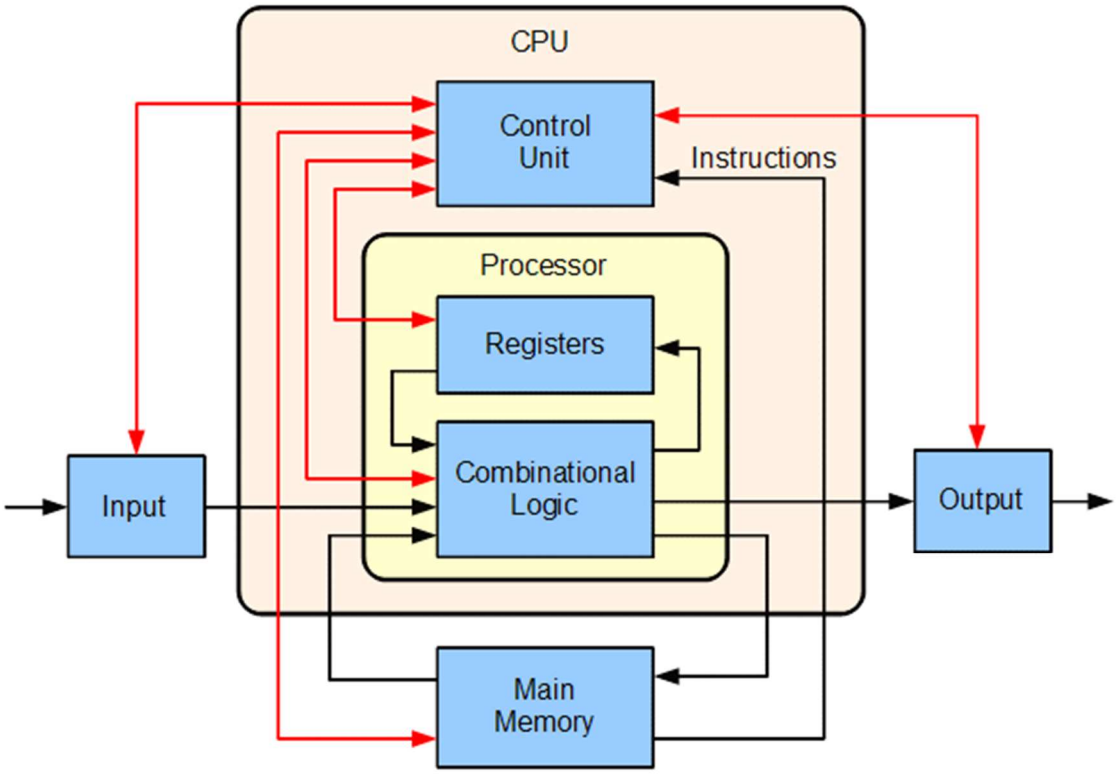
Abstract:
Computer Organization and Architecture is a key component of the computer engineering and the practicing computer engineer should have a practical understanding of this topic. It is concerned with all aspects of the design and organization of the central processing unit and the integration of the CPU into the computer system itself. In describing computers a distinction is often made between computer architecture and computer organization. Although it is difficult to give precise definitions for these terms a consensus exists about the general areas covered.

Introduction:
Computer Organization and Architecture is the study of internal working, structuring and implementation of a computer system. Architecture in computer system, same as anywhere else, refers to the externally visual attributes of the system. Externally visual attributes, here in computer science, mean the way a system is visible to the logic of programs (not the human eyes!). Organization of computer system is the way of practical implementation which results in realization of architectural specifications of a computer system. In more general language, Architecture of computer system can be considered as a catalog of tools available for any operator using the system, while Organization will be the way the system is structured so that all those cataloged tools can be used, and that in an efficient fashion.

For example, it is an architectural design issue whether a computer will have a multiply instruction. It is an organizational issue whether that instruction will be implemented by a special multiply unit or by a mechanism that makes repeated use of the add unit of the system. The organizational decision may be based on the anticipated frequency of use of the multiply instruction, the relative speed of the two approaches, and the cost and physical size of a special multiply unit

Computer Architecture:

In computer engineering, computer architecture is a set of rules and methods that describe the functionality, organization, and implementation of computer systems. Some definitions of architecture define it as describing the capabilities and programming model of a computer but not a particular implementation. In other definitions computer architecture involves instruction set architecture design, microarchitecture design, logic design, and implementation.



Block diagram of a basic computer with uniprocessor CPU. Black lines indicate data flow, whereas red lines indicate control flow. Arrows indicate the direction of flow.

History:

The first documented computer architecture was in the correspondence between Charles Babbage and Ada Lovelace, describing the analytical engine. When building the computer Z1 in 1936, Konrad Zuse described in two patent applications for his future projects that machine instructions could be stored in the same storage used for data, i.e., the stored-program concept. Two other early and important examples are:

- John von Neumann's 1945 paper, First Draft of a Report on the EDVAC, which described an organization of logical elements; and
- Alan Turing's more 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 traced to the work of Lyle R. Johnson and Frederick P. Brooks, Jr., members of the Machine Organization department in IBM's main research center in 1959. Johnson had the opportunity to write a proprietary research communication about the Stretch, an IBM-developed supercomputer for Los Alamos National Laboratory (at the time known as Los Alamos Scientific Laboratory). To describe the level of detail for discussing the luxuriously embellished computer, he noted that his description of formats, instruction types, hardware parameters, and speed enhancements were at the level of “system architecture”, a term that seemed more useful than “machine organization”.

Subsequently, Brooks, a Stretch designer, opened Chapter 2 of a book called Planning a Computer System: Project Stretch by stating, “Computer architecture, like other architecture, is the art of determining the needs of the user of a structure and then designing to meet those needs as effectively as possible within economic and technological constraints.”

Brooks went on to help develop the IBM System/360 (now called the IBM zSeries) line of computers, in which “architecture” became a noun defining “what the user needs to know”.[9] Later, computer users came to use the term in many less explicit ways.[10]

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

Subcategories:

The discipline of computer architecture has three main subcategories:

- Instruction set architecture (ISA): defines the machine code that a processor reads and acts upon as well as the word size, memory address modes, processor registers, and data type.
- Microarchitecture: also known as "computer organization", this describes how a particular processor will implement the ISA. The size of a computer's CPU cache for instance, is an issue that generally has nothing to do with the ISA.
- Systems design: includes all of the other hardware components within a computing system, such as data processing other than the CPU (e.g., direct memory access), virtualization, and multiprocessing
- There are other technologies in computer architecture. The following technologies are used in bigger companies like Intel, and were estimated in 2002 to count for 1% of all of computer architecture:
- Macro architecture: architectural layers more abstract than microarchitecture
- Assembly instruction set architecture: A smart assembler may convert an abstract assembly language common to a group of machines into slightly different machine language for different implementations.
- Programmer-visible macro architecture: higher-level language tools such as compilers may define a consistent interface or contract to programmers using them, abstracting differences between underlying ISA, UISA, and microarchitectures. For example, the C, C++, or Java standards define different programmer-visible macro architectures.
- Microcode: microcode is software that translates instructions to run on a chip. It acts like a wrapper around the hardware, presenting a preferred version of the hardware's instruction set interface. This instruction translation facility gives chip designers flexible options: Eg 1. A new improved version of the chip can use microcode to present the exact same instruction set as the old chip version, so all software targeting that instruction set will run on the new chip without needing changes. Eg 2. Microcode can present a variety of instruction sets for the same underlying chip, allowing it to run a wider variety of software.
- UISA: User Instruction Set Architecture, refers to one of three subsets of the RISC CPU instructions provided by PowerPC RISC Processors. The UISA subset, are those RISC instructions of interest to application developers. The other two subsets are VEA (Virtual Environment Architecture) instructions used by virtualization system developers, and OEA (Operating Environment Architecture) used by Operation System developers.
- Pin architecture: The hardware functions that a microprocessor should provide to a hardware platform, e.g., the x86 pins A20M, FERR/IGNNE or FLUSH. Also, messages that the processor should emit so that external caches can be invalidated (emptied). Pin architecture functions are more flexible than ISA functions because external hardware can adapt to new encodings, or change from a pin to a message. The term "architecture" fits, because the functions must be provided for compatible systems, even if the detailed method changes.

Instruction set architecture:
An instruction set architecture (ISA) is the interface between the computer's software and hardware and also can be viewed as the programmer's view of the machine. Computers do not understand high-level programming languages such as Java, C++, or most programming languages used. A processor only understands instructions encoded in some numerical fashion, usually as binary numbers. Software tools, such as compilers, translate those high level languages into instructions that the processor can understand.

Besides instructions, the ISA defines items in the computer that are available to a program—e.g., data types, registers, addressing modes, and memory. Instructions locate these available items with register indexes (or names) and memory addressing modes.

The ISA of a computer is usually described in a small instruction manual, which describes how the instructions are encoded. Also, it may define short (vaguely) mnemonic names for the instructions. The names can be recognized by a software development tool called an assembler. An assembler is a computer program that translates a human-readable form of the ISA into a computer-readable form. Disassemblers are also widely available, usually in debuggers and software programs to isolate and correct malfunctions in binary computer programs.

ISAs vary in quality and completeness. A good ISA compromises between programmer convenience (how easy the code is to understand), size of the code (how much code is required to do a specific action), cost of the computer to interpret the instructions (more complexity means more hardware needed to decode and execute the instructions), and speed of the computer (with more complex decoding hardware comes longer decode time). Memory organization defines how instructions interact with the memory, and how memory interacts with itself.

During design emulation, emulators can run programs written in a proposed instruction set. Modern emulators can measure size, cost, and speed to determine whether a particular ISA is meeting its goals

Computer organization:
Computer organization helps optimize performance-based products. For example, software engineers need to know the processing power of processors. They may need to optimize software in order to gain the most performance for the lowest price. This can require quite detailed analysis of the computer's organization. For example, in a SD card, the designers might need to arrange the card so that the most data can be processed in the fastest possible way.

Computer organization also helps plan the selection of a processor for a particular project. Multimedia projects may need very rapid data access, while virtual machines may need fast interrupts. Sometimes certain tasks need additional components as well. For example, a computer

capable of running a virtual machine needs virtual memory hardware so that the memory of different virtual computers can be kept separated. Computer organization and features also affect power consumption and processor cost.

Implementation:
Once an instruction set and micro-architecture have been designed, a practical machine must be developed. This design process is called the implementation. Implementation is usually not considered architectural design, but rather hardware design engineering. Implementation can be further broken down into several steps:

- Logic implementation designs the circuits required at a logic-gate level.
- Circuit implementation does transistor-level designs of basic elements (e.g., gates, multiplexers, latches) as well as of some larger blocks (ALUs, caches etc.) that may be implemented at the logic-gate level, or even at the physical level if the design calls for it.
- Physical implementation draws physical circuits. The different circuit components are placed in a chip floor plan or on a board and the wires connecting them are created.
- Design validation tests the computer as a whole to see if it works in all situations and all timings. Once the design validation process starts, the design at the logic level are tested using logic emulators. However, this is usually too slow to run a realistic 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 stage. The final step is to test prototype integrated circuits, which may require several redesigns.
- For CPUs, the entire implementation process is organized differently and is often referred to as CPU design.

Design goals
The exact form of a computer system depends on the constraints and goals. Computer architectures usually trade off standards, power versus performance, cost, memory capacity, latency (latency is the amount of time that it takes for information from one node to travel to the source) and throughput. Sometimes other considerations, such as features, size, weight, reliability, and expandability are also factors.

The most common scheme does an in-depth power analysis and figures out how to keep power consumption low while maintaining adequate performance.

Performance:
Modern computer performance is often described in instructions per cycle (IPC), which measures the efficiency of the architecture at any clock frequency; a faster IPC rate means the computer is faster. Older computers had IPC counts as low as 0.1 while modern processors easily reach near

Superscalar processors may reach three to five IPC by executing several instructions per clock cycle

Counting machine-language instructions would be misleading because they can do varying amounts of work in different ISAs. The "instruction" in the standard measurements is not a count of the ISA's machine-language instructions, but a unit of measurement, usually based on the speed of the VAX computer architecture.

Many people used to measure a computer's speed by the clock rate (usually in MHz or GHz). This refers to the cycles per second of the main clock of the CPU. However, this metric is somewhat misleading, as a machine with a higher clock rate may not necessarily have greater performance. As a result, manufacturers have moved away from clock speed as a measure of performance.

Other factors influence speed, such as the mix of functional units, bus speeds, available memory, and the type and order of instructions in the programs.

There are two main types of speed: latency and throughput. Latency is the time between the start of a process and its completion. Throughput is the amount of work done per unit time. Interrupt latency is the guaranteed maximum response time of the system to an electronic event (like when the disk drive finishes moving some data).

Performance is affected by a very wide range of design choices — for example, pipelining a processor usually makes latency worse, but makes throughput better. Computers that control machinery usually need low interrupt latencies. These computers operate in a real-time environment and fail if an operation is not completed in a specified amount of time. For example, computer-controlled anti-lock brakes must begin braking within a predictable and limited time period after the brake pedal is sensed or else failure of the brake will occur.

Benchmarking takes all these factors into account by measuring the time a computer takes to run through a series of test programs. Although benchmarking shows strengths, it shouldn't be how you choose a computer. Often the measured machines split on different measures. For example, one system might handle scientific applications quickly, while another might render video games more smoothly. Furthermore, designers may target and add special features to their products, through hardware or software, That permit a specific benchmark to execute quickly but don't offer similar advantages to general tasks.

Power efficiency:

Power efficiency is another important measurement in modern computers. A higher power efficiency can often be traded for lower speed or higher cost. The typical measurement when

referring to power consumption in computer architecture is MIPS/W (millions of instructions per second per watt).

Modern circuits have less power required per transistor as the number of transistors per chip grows. This is because each transistor that is put in a new chip requires its own power supply and requires new pathways to be built to power it. However the number of transistors per chip is starting to increase at a slower rate. Therefore, power efficiency is starting to become as important, if not more important than fitting more and more transistors into a single chip. Recent processor designs have shown this emphasis as they put more focus on power efficiency rather than cramming as many transistors into a single chip as possible. In the world of embedded computers, power efficiency has long been an important goal next to throughput and latency.

Shifts in market demand:
Increases in clock frequency have grown more slowly over the past few years, compared to power reduction improvements. This has been driven by the end of Moore's Law and demand for longer battery life and reductions in size for mobile technology. This change in focus from higher clock rates to power consumption and miniaturization can be shown by the significant reductions in power consumption, as much as 50%, that were reported by Intel in their release of the Haswell microarchitecture; where they dropped their power consumption benchmark from 30-40 watts down to 10-20 watts. Comparing this to the processing speed increase of 3 GHz to 4 GHz (2002 to 2006) it can be seen that the focus in research and development are shifting away from clock frequency and moving towards consuming less power and taking up less space

Structure and Function:
A computer is a complex system; contemporary computers contain millions of elementary electronic components. How, then, can one clearly describe them? The key is to recognize the hierarchical nature of most complex system. A hierarchical system is a set of interrelated subsystem, each of the later, in turn, hierarchical in structure until we reach some lowest level of elementary subsystem. The behavior at each level depends only on a simplified, abstracted characterization of the system at the next lower level. At each level, the designer is concerned with structure and function:

- Structure: The way in which the components are interrelated.
- Function: The operation of each individual component as part of the structure.

Function:

In general terms, there are four main functions of a computer:

- Data processing
- Data storage
- Data movement
- Control

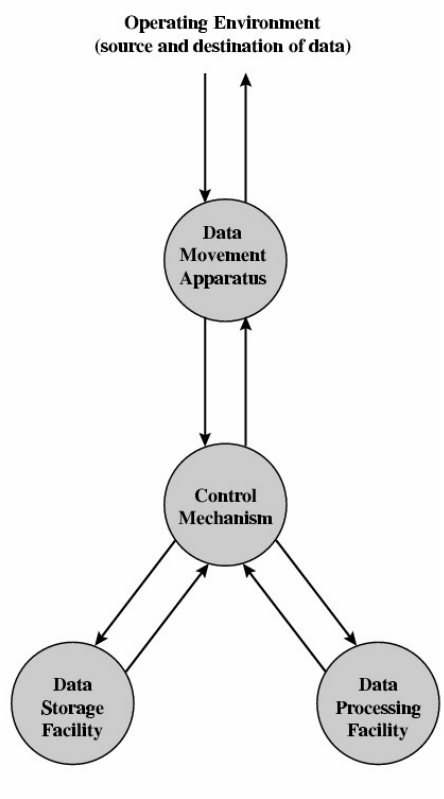


Figure: A functional view of the computer

The computer, of course, must be able to process data. The data may take a wide variety of forms, and the range of processing requirements is broad. However, we shall see that there are only a few fundamental methods or types of data processing.

It is also essential that a computer store data. Even if the computer is processing data on the fly (i.e., data come in and get processed, and the results go out immediately), the computer must temporarily store at least those pieces of data that are being worked on at any given moment. Thus, there is at least a short-term data storage function. Files of data are stored on the computer for subsequent retrieval and update.

The computer must be able to move data between itself and the outside world. The computer's operating environment consists of devices that serve as either sources or destinations of data.

When data are received from or delivered to a device that is directly connected to the computer, the process is known as input-output (I/O), and the device is referred to as a peripheral. When data are moved over longer distances, to or from a remote device, the process is known as data communications. Finally, there must be control of their three functions. Ultimately, this control is exercised by the individual who provides the computer with instructions. Within the computer system, a control unit manages the computer's resources and orchestrates the performance of its functional parts in response to those instructions.

At this general level of discussion, the number of possible operations that can be performed is few. The figure 1.2 depicts the four possible types of operations.

The computer can function as a data movement device simply transferring data from one peripheral or communications line to another. It can also function as a data storage device with data transferred from the external environment to computer storage (read) and vice versa (write). The final two diagrams show operations involving data processing, on data either in storage or in route between storage and the external environment.

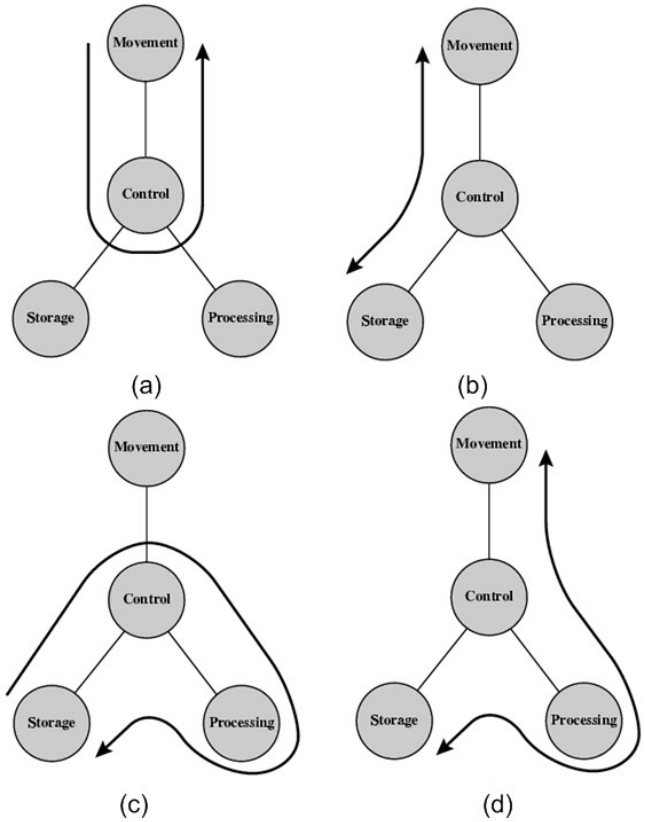


Figure: Possible computer operations

Structure:

Figure is the simplest possible depiction of a computer. The computer is an entity that interacts in some fashion with its external environment. In general, all of its linkages to the external environment can be classified as peripheral devices or communication lines. We will have something to say about both types of linkages.

- Central Processing Unit (CPU): Controls the operation of the computer and performs its data processing functions. Often simply referred to as processor.
- Main Memory: Stores data.
- I/O: Moves data between the computer and its external environment.
- System Interconnection: Some mechanism that provides for communication among CPU, main memory, and I/O.

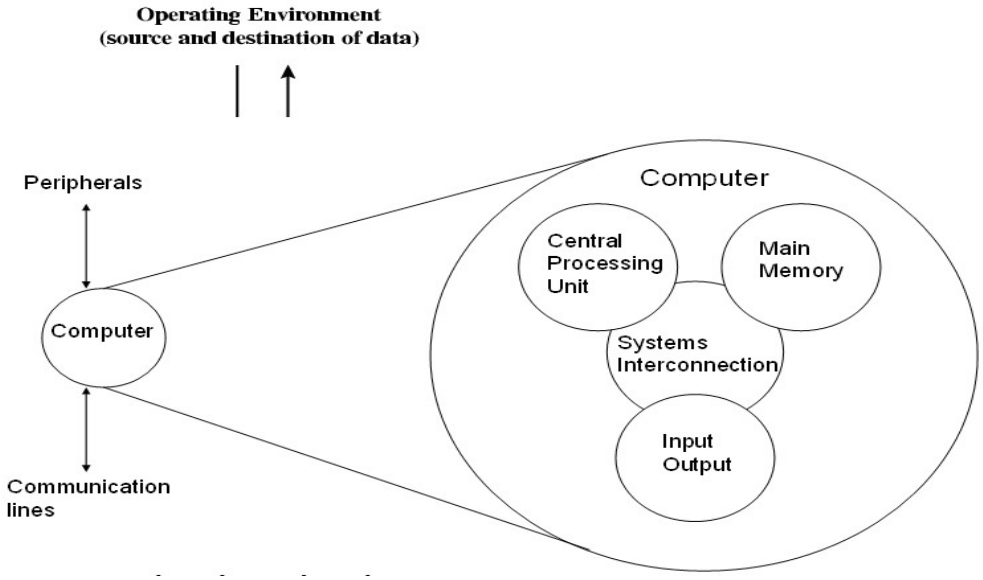


Figure: The computer: top-level structure

There may be one or more of each of the above components. Traditionally, there has been just a single CPU. In recent years, there has been increasing use of multiple processors, in a single system. Each of these components will be examined in some detail in later lectures. However, for our purpose, the most interesting and in some ways the most complex component is the CPU; its structure is depicted in Figure its major structural components are:

- Control Unit (CU): Controls the operation of the CPU and hence the computer.
- Arithmetic and Logic Unit (ALU): Performs computer's data processing functions.
- Register: Provides storage internal to the CPU.

- CPU Interconnection: Some mechanism that provides for communication among the control unit, ALU, and register.

Each of these components will be examined in some detail in next lectures.

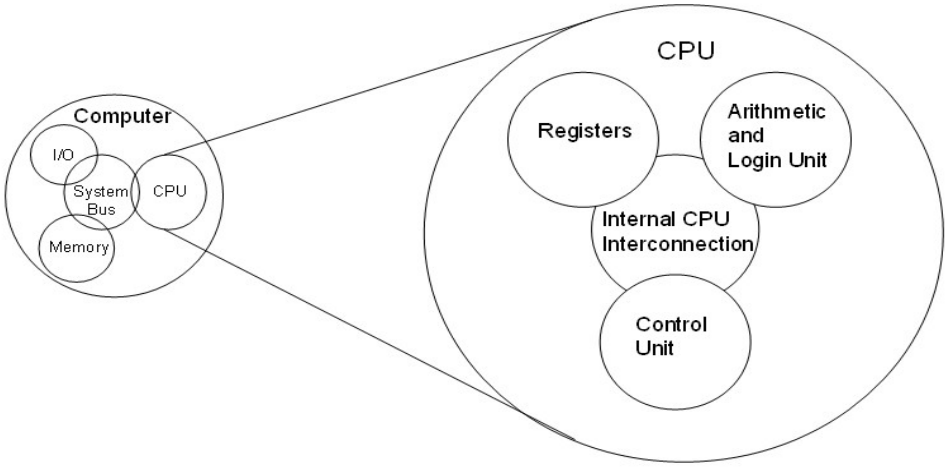


Figure of The CPU

Designing for performance:

Some of the driving factors behind the need to design for performance:

- Microprocessor Speed
- Pipelining
- On board cache, on board L1 & L2 cache
- Branch prediction: The processor looks ahead in the instruction code fetched from memory and predicts which branches, or group of instructions are likely to be processed next.
- Data flow analysis: The processor analyzes which instructions are dependent on each other's results, or data, to create an optimized schedule of instructions to prevent delay.
- Speculative execution: Using branch prediction and data flow analysis, some processors speculatively execute instructions ahead of their actual appearance in the program execution, holding the results in temporary locations.
- Performance Mismatch
- Processor speed increased
- Memory capacity increased
- Memory speed lags behind processor speed.

Brief History of Computers:

In this section, we provide a brief overview of the history of the development of computers. This history is interesting in itself, but more importantly, provides a basic introduction to many important concepts that we deal with throughout the book.

The First Generation: Vacuum Tubes:

The first generation of computers used vacuum tubes for digital logic elements and memory. A number of research and then commercial computers were built using vacuum tubes. For our purposes, it will be instructive to examine perhaps the most famous first-generation computer, known as the IAS computer. A fundamental design approach first implemented in the IAS computer is known as the stored-Program concept. This idea is usually attributed to the mathematician John von Neumann. Alan Turing developed the idea at about the same time. The first publication of the idea was in a 1945 proposal by von Neumann for a new computer, the EDVAC (Electronic Discrete Variable Computer). In 1946, von Neumann and his colleagues began the design of a new stored- program computer, referred to as the IAS computer, at the Princeton Institute for Advanced Studies. The IAS computer, although not completed until 1952, is the prototype of all subsequent general-purpose computers.

Figure shows the structure of the IAS computer (compare with Figure).

It consists of

- A **main memory**, which stores both data and instructions5
- An **arithmetic and logic unit (ALU)** capable of operating on binary data

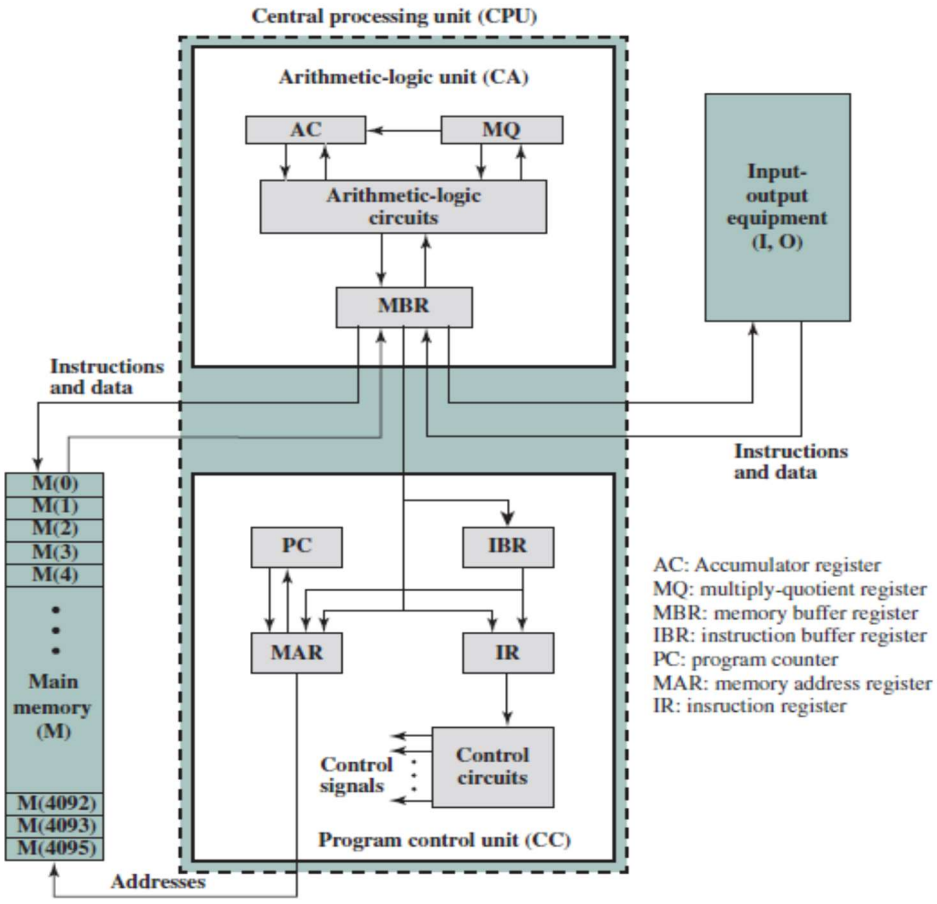


Figure of IAS Structure

- A control unit, which interprets the instructions in memory and causes them to be executed
- Input–output (I/O) equipment operated by the control unit This structure was outlined in von Neumann's earlier proposal, which is worth quoting in part at this point [VONN45]:

ENIAC:
The ENIAC (Electronic Numerical Integrator And Computer), designed by and constructed under the supervision of Jonh Mauchly and John Presper Eckert at the University of Pennsylvania, was the world's first general-purpose electronic digital computer. The project was a response to U.S. wartime needs. Mauchly, a professor of electrical engineering at the University of Pennsylvania and Eckert, one of his graduate students, proposed to build a general-purpose computer using vacuum tubes. In 1943, this proposal was accepted by the Army, and work began on the ENIAC. The resulting machine was enormous, weighting 30 tons, occupying 15,000 square feet of floor space, and containing more than 18,000 vacuum tubes. When operating, it consumed 140 kilowatts of power. It was aloes substantially faster than any electronic-mechanical computer, being capable of 5000 additions per second.

The ENIAC was completed in 1946, too late to be used in the war effort. Instead, its first task was to perform a series of complex calculations that were used to help determine the feasibility of the H-bomb. The ENIAC continued to be used until 1955.

The von Neumann Machine:
The programming process could be facilitated if the program could be represented in a form suitable for storing in memory alongside the data. Then, a computer could get its instructions by reading them from memory, and a program could be set of altered by setting the values of a portion of memory.

This idea, known as the Stored-program concept, is usually attributed to the ENIAC designers, most notably the mathematician John von Neumann, who was a consultant on the ENIAC project. The idea was also developed at about the same time by Turing. The first publication of the idea was in a 1945 proposal by von Neumann for a new computer, the EDVAC (Electronic Discrete Variable Computer).

In 1946, von Neumann and his colleagues began the design of a new stored-program computer, referred to as the IAS computer, at the Princeton Institute for Advanced Studies. The IAS computer, although not completed until 1952, is the prototype of all subsequent general-purpose computers. Figure 1.5 shows the general structure of the IAS computer. It consists of:

- A main memory, which stores both data and instructions.
- An arithmetic-logical unit (ALU) capable of operating on binary data.
- A control unit, which interprets the instructions in memory and causes them to be executed.
- Input and output (I/O) equipment operated by the control unit.

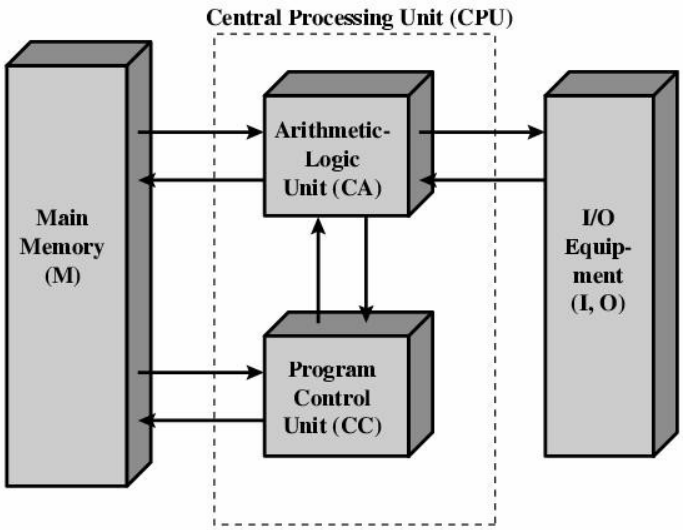


Figure Structure of the IAS computer

Commercial Computers:

The 1950s saw the birth of the computer industry with two companies, Sperry and IBM, dominating the marketplace.

In 1947, Eckert and Mauchly formed the Eckert-Maunchly computer Corporation to manufacture computers commercially. Their first successful machine was the UNIVAC I (Universal Automatic Computer), which was commissioned by the Bureau of the Census for the 1950 calculations. The Eckert-Maunchly Computer Corporation became part of the UNIVAC division of Sperry-Rand Corporation, which went on to build a series of successor machines.

The UNIVAC II, which had greater memory capacity and higher performance than the UNIVAC I, was delivered in the late 1950s and illustrates several trends that have remained characteristic of the computer industry. First, advances in technology allow companies to continue to build larger, more powerful computers. Second, each company tries to make its new machines upward compatible with the older machines. This means that the programs written for the older machines can be executed on the new machine. This strategy is adopted in the hopes of retaining the customer base; that is, when a customer decides to buy a newer machine, he is likely to get it from the same company to avoid losing the investment in programs.

The UNIVAC division also began development of the 1100 series of computers, which was to be its bread and butler. This series illustrates a distinction that existed at one time. The first model, the UNIVAC 1103, and its successors for many years were primarily intended for scientific applications, involving long and complex calculations. Other companies concentrated on business applications, which involved processing large amounts of text data. This split has largely disappeared but it was evident for a number of years.

The Second Generation: Transistors:

The first major change in the electronic computer came with the replacement of the vacuum tube by the transistor. The transistor is smaller, cheaper, and dissipates less heal than a vacuum tube but can be used in the same way as a vacuum tube to construct computers. Unlike the vacuum tube, which requires wires, metal plates, a glass capsule, and a vacuum, the transistor is a solid-state device, made from silicon.

The transistor was invented at Bell Labs in 1947 and by the 1950s had launched an electronic revolution. It was not until the late 1950s, however, that fully transistorized computers were commercially available. IBM again was not the first company to deliver the new technology. NCR and. more successfully. RCA were the front-runners with some small transistor machines. IBM followed shortly with the 7000 series.

The use of the transistor defines the second generation of computers. It has become widely accepted to classify computers into generations based on the fundamental hardware technology employed. Each new generation is characterized by greater processing performance, larger memory capacity, and smaller size than the previous one.

The Third Generation: Integrated Circuits:

A single, self-contained transistor is called a discrete component. Throughout the 1950s and early 1960s, electronic equipment was composed largely of discrete components—transistors, resistors, capacitors, and so on. Discrete components were manufactured separately, packaged in their own containers, and soldered or wired together onto circuit boards, which were then installed in computers, oscilloscopes, and other electronic equipment. Whenever an electronic device called for a transistor, a little lube of metal containing a pinhead-sized piece of silicon had to be soldered to a circuit hoard. The entire manufacturing process, from transistor to circuit board, was expensive and cumbersome.

These facts of life were beginning to create problems in the computer industry. Early second-generation computers contained about 10,000 transistors. This figure grew to the hundreds of thousands, making the manufacture of newer, more powerful machines increasingly difficult.

In 1958 came the achievement that revolutionized electronics and started the era of microelectronics: the invention of the integrated circuit. It is the integrated circuit that defines the third generation of computers. Perhaps the two most important members of the third generation are the IBM System/360 and the DEC PDP-8.