CSC 212: COMPUTER ORGANIZATION AND ARCHITECTURE 1

Lecturer: Barasa Mamati Godliphas

godliphas@gmail.com

Contents

INTRODUCTION

Computer Architecture:

Computer Architecture deals with giving operational attributes of the computer or Processor to be specific. It deals with details like physical memory, ISA (Instruction Set Architecture) of the processor, the number of bits used to represent the data types, Input Output mechanism and technique for addressing memories.

Computer Organization:

Computer Organization is realization of what is specified by the computer architecture .It deals with how operational attributes are linked together to meet the requirements specified by computer architecture. Some organizational attributes are hardware details, control signals, peripherals.

EXAMPLE:

Say you are in a company that manufactures cars, design and all low-level details of the car come under computer architecture (abstract, programmers view), while making it's parts piece by piece and connecting together the different components of that car by keeping the basic design in mind comes under computer organization (physical and visible).

Differences between Computer Organization and Computer Architecture

Computer Organization	**Computer Architecture**
Often called microarchitecture (low level)	Computer architecture (a bit higher level)
Transparent from programmer (ex. a programmer does not worry much how addition is implemented in hardware)	Programmer view (i.e. Programmer has to be aware of which instruction set used)
Physical components (Circuit design, Adders, Signals, Peripherals)	Logic (Instruction set, Addressing modes, Data types, Cache optimization)
How to do ? (implementation of the architecture)	What to do ? (Instruction set)

GENERATIONS OF A COMPUTER

Generation in computer terminology is a change in technology a computer is/was being used. Initially, the generation term was used to distinguish between varying hardware technologies. But nowadays, generation includes both hardware and software, which together make up an entire computer system. There are totally five computer generations known till date. Each generation has been discussed in detail along with their time period and characteristics. Here approximate dates against each generations have been mentioned which are normally accepted.

Following are the main five generations of computers

S.N.	Generation & Description
1	**First Generation** The period of first generation: 1946-1959. Vacuum tube based. The main features of first generation are: Vacuum tube technology Unreliable Supported machine language only Very costly Generated lot of heat Slow input and output devices Huge size Need of A.C. Non-portable Consumed lot of electricity
2	**Second Generation** The period of second generation: 1959-1965. Transistor based. The main features of second generation are: Use of transistors Reliable in comparison to first generation computers Smaller size as compared to first generation computers Generated less heat as compared to first generation computers Consumed less electricity as compared to first generation computers Faster than first generation computers Still very costly A.C. needed Supported machine and assembly languages

3	**Third Generation** The period of third generation: 1965-1971. Integrated Circuit based. The main features of third generation are: IC used More reliable in comparison to previous two generations Smaller size Generated less heat Faster Lesser maintenance Still costly A.C needed Consumed lesser electricity Supported high-level language
4	**Fourth Generation** The period of fourth generation: 1971-1980. VLSI microprocessor based. The main features of fourth generation are: VLSI technology used Very cheap Portable and reliable Use of PC's Very small size Pipeline processing No A.C. needed Concept of internet was introduced Great developments in the fields of networks Computers became easily available
5	**Fifth Generation** The period of fifth generation: 1980-onwards. ULSI microprocessor based The main features of fifth generation are: ULSI technology Development of true artificial intelligence Development of Natural language processing Advancement in Parallel Processing Advancement in Superconductor technology More user friendly interfaces with multimedia features Availability of very powerful and compact computers at cheaper rates

DATA REPRESENTATION

Data representation

Computers represent data using a binary number system, which means that all data is stored as a sequence of 0s and 1s. This is because computers are electronic devices, and they can only store two different voltage levels: high and low. The high voltage level is represented by the number 1, and the low voltage level is represented by the number 0.

There are different ways to represent different types of data in binary. For example, numbers can be represented using the two's complement notation, while characters can be represented using the ASCII character encoding scheme.

Here are some examples of how different types of data are represented in binary:

- Number: The number 10 in decimal is represented as 1010 in binary.
- Character: The letter "A" is represented as 01000001 in ASCII.
- Image: An image can be represented as a grid of pixels, where each pixel is represented by a single byte. The brightness of each pixel is determined by the value of the byte.
- Audio: An audio clip can be represented as a series of samples, where each sample is represented by a single byte. The amplitude of each sample represents the volume of the sound at that point in time.

Computers use a variety of different hardware components to store and process data in binary. The main memory of a computer is used to store data that the computer is currently using. The central processing unit (CPU) is used to process data and perform calculations. The hard disk drive (HDD) or solid-state drive (SSD) is used to store data that is not currently being used by the computer.

Number systems

A number system is a way of representing numbers. The most common number system is the decimal system, which uses ten digits (0, 1, 2, 3, 4, 5, 6, 7, 8, 9). However, there are many other number systems, such as the binary system (which uses two digits, 0 and 1), the octal system (which uses eight digits, 0, 1, 2, 3, 4, 5, 6, 7), and the hexadecimal system (which uses sixteen digits, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F).

The base of a number system is the number of digits that are used to represent numbers. For example, the decimal system has a base of 10, the binary system has a base of 2, and the hexadecimal system has a base of 16.

To convert a number from one number system to another, you need to know the base of both number systems. For example, to convert the decimal number 10 to binary, you would divide 10 by 2 and then write down the remainder (0). Then, you would divide the quotient (5) by 2 and then write down the remainder (1). You would continue this process until the quotient is 0. The remainders, read from bottom to top, give you the binary representation of 10, which is 1010.

Here is a table that shows the equivalent representations of the number 10 in different number systems:

Number system	Base	Representation
Decimal	10	10

Binary	2	1010
Octal	8	12
Hexadecimal	16	A

DIGITAL LOGIC

Digital logic is the study of how to represent and manipulate information using binary digits, or bits. Bits can be either 0 or 1, and they are used to represent all sorts of information, from numbers and letters to images and sounds.

Digital logic is the foundation of all modern digital electronics, including computers, smartphones, and smart TVs. It is used to design and implement the circuits that perform the computations and operations that these devices need to function.

Digital logic is based on a set of fundamental Boolean operators: AND, OR, NOT, and XOR. These operators can be used to create more complex logic circuits, such as adders, subtractors, multipliers, and memory elements.

Digital logic circuits can be divided into two main categories: combinational logic and sequential logic.

Combinational logic circuits are those whose outputs depend only on their current inputs. Sequential logic circuits, on the other hand, have memory elements that store previous inputs and outputs, and their outputs depend on both their current inputs and their previous state.

Combinational logic circuits are used to implement basic logical operations, such as AND, OR, NOT, and XOR. They are also used to implement more complex functions, such as adders, subtractors, multipliers, and comparators.

Sequential logic circuits are used to implement more complex functions, such as memory, registers, and counters. They are also used to implement state machines, which are circuits that can perform a sequence of actions based on their inputs and their current state.

Digital logic is a powerful tool that can be used to design and implement a wide variety of electronic devices. It is an essential part of the modern world, and it is used in everything from computers to smartphones to cars.

Here are some examples of digital logic circuits:

- Adder: An adder is a circuit that adds two numbers together.
- Subtractor: A subtractor is a circuit that subtracts one number from another.
- Multiplier: A multiplier is a circuit that multiplies two numbers together.
- Divider: A divider is a circuit that divides one number by another.
- Memory: Memory is a circuit that stores data.
- Register: A register is a circuit that stores a small amount of data, such as the current instruction being executed by a CPU.
- Counter: A counter is a circuit that counts the number of times a particular event has occurred.
- State machine: A state machine is a circuit that can perform a sequence of actions based on its inputs and its current state.

LOGIC GATES

Logic gates are the basic building blocks of digital logic circuits. They are used to perform simple logical operations, such as AND, OR, NOT, and XOR. Logic gates can be combined to create more complex logic circuits, such as adders, subtractors, multipliers, and memory elements.

There are seven basic logic gates:

- AND: The AND gate outputs a 1 only if both inputs are 1.
- OR: The OR gate outputs a 1 if either input is 1.
- NOT: The NOT gate inverts the input signal.
- NAND: The NAND gate is the inverse of the AND gate. It outputs a 0 if both inputs are 1, and a 1 otherwise.
- NOR: The NOR gate is the inverse of the OR gate. It outputs a 0 if either input is 1, and a 1 otherwise.
- XOR: The XOR gate outputs a 1 if the inputs are different, and a 0 if the inputs are the same.
- XNOR: The XNOR gate is the inverse of the XOR gate. It outputs a 1 if the inputs are the same, and a 0 if the inputs are different.

Logic gates can be implemented using a variety of technologies, such as transistors, diodes, and integrated circuits. Logic gates are used in all sorts of digital electronic devices, including computers, smartphones, and smart TVs.

Here is an example of a truth table involving logic gates:

AND gate:

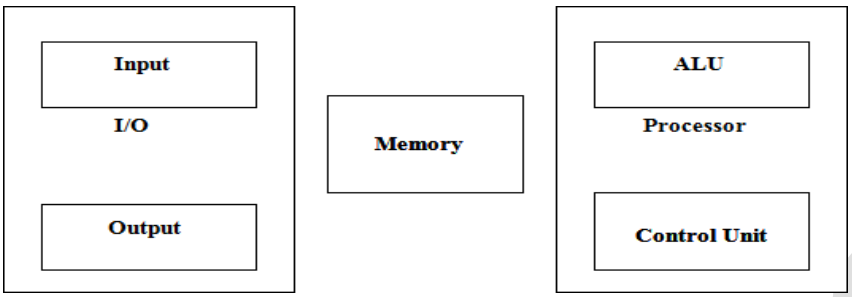
Input A	Input B	Output
0	0	0
0	1	0
1	0	0
1	1	1

OR gate:

Input A	Input B	Output
0	0	0
0	1	1
1	0	1
1	1	1

FUNCTIONAL UNIT OF A COMPUTER

A computer consists of five functionally independent main parts input, memory, arithmetic logic unit (ALU), output and control unit.

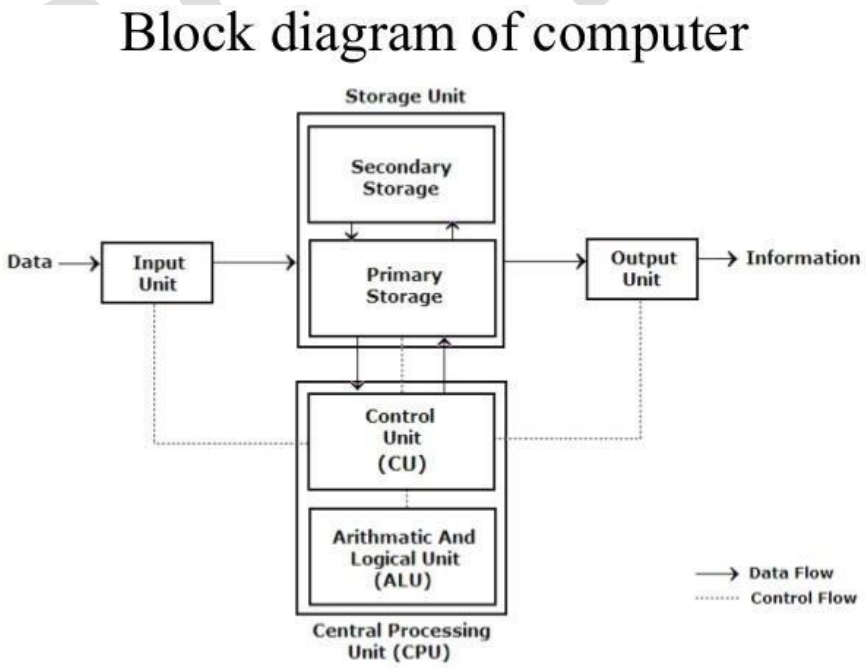


Functional units of computer

Input devices

Input device accepts the coded information as source program i.e. high level language. This is either stored in the memory or immediately used by the processor to perform the desired operations. The program stored in the memory determines the processing steps. Basically the computer converts one source program to an object program.i.e. into machine language.

Finally the results are sent to the outside world through output device. All of these actions are coordinated by the control unit.



The source program/high level language program/coded information/simply data is fed to a computer through input devices keyboard is a most common type. Whenever a key is pressed, one corresponding word or number is translated into its equivalent binary code over a cable & fed either to memory or processor.
Joysticks, trackballs, mouse, scanners etc are other input devices.

Memory unit: -

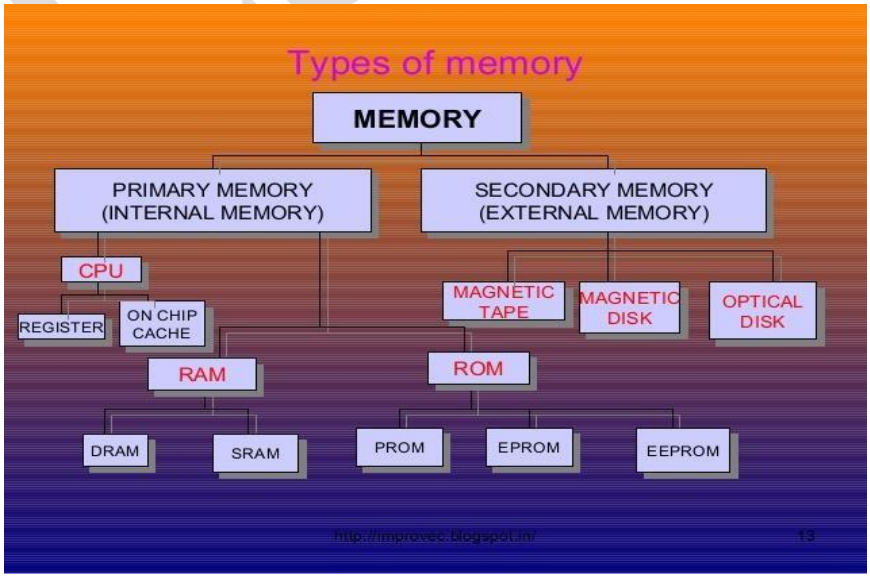
Its function into store programs and data. It is basically to two types

- **Primary memory**
- **Secondary memory**

Word:

In computer architecture, a word is a unit of data of a defined bit length that can be addressed and moved between storage and the computer processor. Usually, the defined bit length of a word is equivalent to the width of the computer's data bus so that a word can be moved in a single operation from storage to a processor register. For any computer architecture with an eight-bit byte, the word will be some multiple of eight bits. In IBM's evolutionary System/360 architecture, a word is 32 bits, or four contiguous eight-bit bytes. In Intel's PC processor architecture, a word is 16 bits, or two contiguous eight-bit bytes. A word can contain a computer instruction, a storage address, or application data that is to be manipulated (for example, added to the data in another word space).

The number of bits in each word is known as word length. Word length refers to the number of bits processed by the CPU in one go. With modern general purpose computers, word size can be 16 **bits** to **64 bits**. The time required to access one word is called the memory access time. The small, fast, RAM units are called caches. They are tightly coupled with the processor and are often contained on the same IC chip to achieve high performance.



1. **Primary memory: -** Is the one exclusively associated with the processor and operates at the electronics speeds programs must be stored in this memory while they are being executed. The memory contains a large number of semiconductors storage cells. Each capable of storing one bit of information. These are processed in a group of fixed site called word.

To provide easy access to a word in memory, a distinct address is associated with each word location.

Addresses are numbers that identify memory location.

Number of bits in each word is called word length of the computer. Programs must reside in the memory during execution. Instructions and data can be written into the memory or read out under the control of processor. Memory in which any location can be reached in a short and fixed amount of time after specifying its address is called random- access memory (RAM).

The time required to access one word in called memory access time. Memory which is only readable by the user and contents of which can't be altered is called read only memory (ROM) it contains operating system.

Caches are the small fast RAM units, which are coupled with the processor and are often contained on the same IC chip to achieve high performance. Although primary storage is essential it tends to be expensive.

2 Secondary memory: - Is used where large amounts of data & programs have to be stored, particularly information that is accessed infrequently.

Examples: - Magnetic disks & tapes, optical disks (ie CD-ROM's), floppies etc.,

Arithmetic logic unit (ALU):-

Most of the computer operators are executed in ALU of the processor like addition, subtraction, division, multiplication, etc. the operands are brought into the ALU from memory and stored in high speed storage elements called register. Then according to the instructions the operation is performed in the required sequence. The control and the ALU are may times faster than other devices connected to a computer system. This enables a single processor to control a number of external devices such as key boards, displays, magnetic and optical disks, sensors and other mechanical controllers.

Output unit:-

These actually are the counterparts of input unit. Its basic function is to send the processed results to the outside world.

Examples:- Printer, speakers, monitor etc.

Control unit:-

It effectively is the nerve center that sends signals to other units and senses their states. The actual timing signals that govern the transfer of data between input unit, processor, memory and output unit are generated by the control unit.

To perform a given task an appropriate program consisting of a list of instructions is stored in the memory. Individual instructions are brought from the memory into the processor, which executes the specified operations. Data to be stored are also stored in the memory.

Examples: - Add LOCA, R_0

This instruction adds the operand at memory location LOCA, to operand in register R_0 & places the sum into register. This instruction requires the performance of several steps,

1. First the instruction is fetched from the memory into the processor.
2. The operand at LOCA is fetched and added to the contents of R_0
3. Finally the resulting sum is stored in the register R_0

The preceding add instruction combines a memory access operation with an ALU Operations. In some other type of computers, these two types of operations are performed by separate instructions for performance reasons.

Load LOCA, R1 Add R1, R0

Transfers between the memory and the processor are started by sending the address of the memory location to be accessed to the memory unit and issuing the appropriate control signals. The data are then transferred to or from the memory.

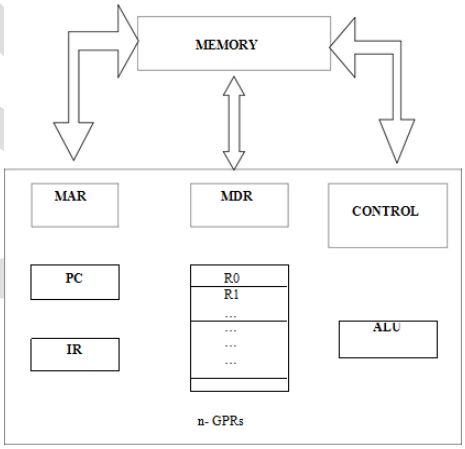


Fig b : Connections between the processor and the memory

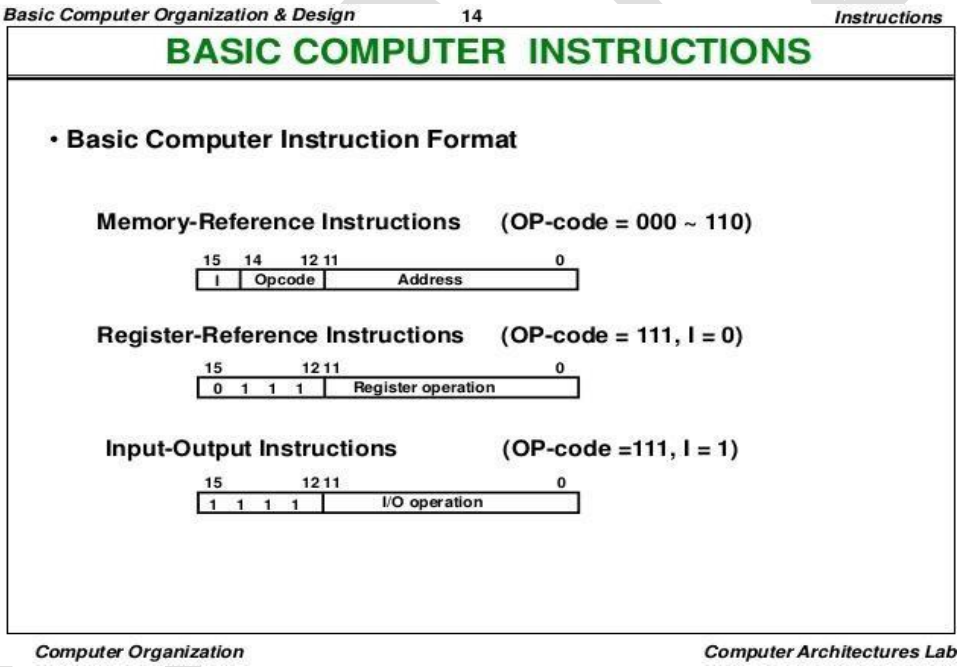
the processor can be connected. In addition to the ALU & the control circuitry, the processor contains a number of registers used for several different purposes.

Register:

It is a special, high-speed storage area within the CPU. All data must be represented in a register before it can be processed. For example, if two numbers are to be multiplied, both numbers must be in registers, and the result is also placed in a register. (The register can contain the address of a memory location where data is stored rather than the actual data itself.)

The number of registers that a CPU has and the size of each (number of bits) help determine the power and speed of a CPU. For example a 32-bit CPU is one in which each register is 32 bits wide. Therefore, each CPU instruction can manipulate 32 bits of data. In high-level languages, the compiler is responsible for translating high-level operations into low-level operations that access registers.

Instruction Format:



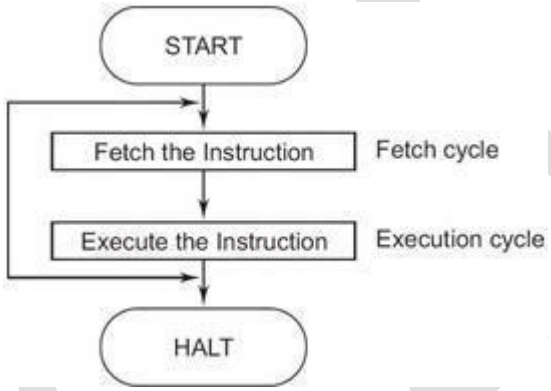
Computer instructions are the basic components of a machine language program. They are also known as *macro operations*, since each one is comprised of sequences of micro operations.

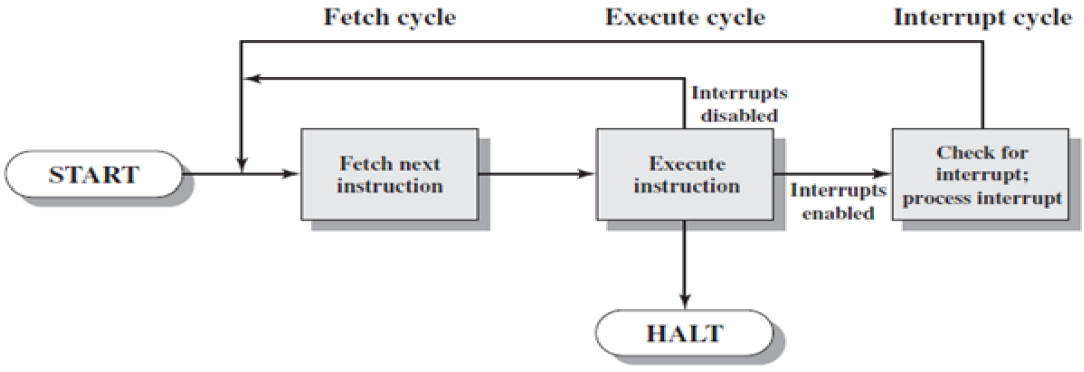
Each instruction initiates a sequence of micro operations that fetch operands from registers or memory, possibly perform arithmetic, logic, or shift operations, and store results in registers or memory.

Instructions are encoded as binary *instruction codes*. Each instruction code contains of a *operation code*, or *opcode*, which designates the overall purpose of the instruction (e.g. add, subtract, move, input, etc.). The number of bits allocated for the opcode determined how many different instructions the architecture supports.

The *control unit* is responsible for decoding the opcode and operand bits in the instruction register, and then generating the control signals necessary to drive all other hardware in the CPU to perform the sequence of micro operations that comprise the instruction.

INSTRUCTION CYCLE:





Instruction Cycle with Interrupts

The instruction register (IR):- Holds the instructions that are currently being executed. Its output is available for the control circuits which generates the timing signals that control the various processing elements in one execution of instruction.

The program counter PC:-

This is another specialized register that keeps track of execution of a program. It contains the memory address of the next instruction to be fetched and executed.

Besides IR and PC, there are n-general purpose registers R0 through R_{n-1}.

The other two registers which facilitate communication with memory are: -

MAR – (Memory Address Register):- It holds the address of the location to be accessed.

MDR – (Memory Data Register):- It contains the data to be written into or read out of the address location.

Operating steps are

Programs reside in the memory & usually get these through the I/P unit.

Execution of the program starts when the PC is set to point at the first instruction of the program.

Contents of PC are transferred to MAR and a Read Control Signal is sent to the memory.

After the time required to access the memory elapses, the address word is read out of the memory and loaded into the MDR.

Now contents of MDR are transferred to the IR & now the instruction is ready to be decoded and executed.

If the instruction involves an operation by the ALU, it is necessary to obtain the required operands.

An operand in the memory is fetched by sending its address to MAR & Initiating a read cycle.

When the operand has been read from the memory to the MDR, it is transferred from MDR to the ALU.

After one or two such repeated cycles, the ALU can perform the desired operation.

If the result of this operation is to be stored in the memory, the result is sent to MDR.

Address of location where the result is stored is sent to MAR & a write cycle is initiated. The contents of PC are incremented so that PC points to the next instruction that is to be executed.

Normal execution of a program may be preempted (temporarily interrupted) if some devices require urgent servicing, to do this one device raises an Interrupt signal. An interrupt is a request signal from an I/O device for service by the processor. The processor provides the requested service by executing an appropriate interrupt service routine.

The Diversion may change the internal stage of the processor its state must be saved in the memory location before interruption. When the interrupt-routine service is completed the state of the processor is restored so that the interrupted program may continue

THE VON NEUMANN ARCHITECTURE

The task of entering and altering programs for the ENIAC was extremely tedious. The programming process can be easy 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 or altered by setting the values of a portion of memory. This idea is known a the stored-program concept. 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.

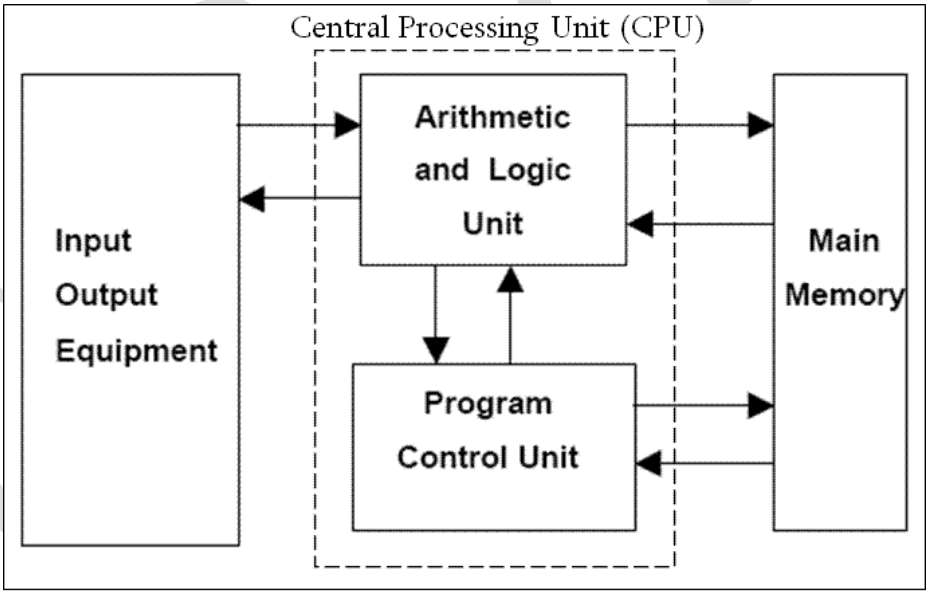


Figure : General structure of Von Neumann Architecture

It consists of

- A main memory, which stores both data and instruction
- An arithmetic and logic 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

BUS STRUCTURES:

Bus structure and multiple bus structures are types of bus or computing. A bus is basically a subsystem which transfers data between the components of Computer components either within a computer or between two computers. It connects peripheral devices at the same time.

A multiple Bus Structure has multiple inter connected service integration buses and for each bus the other buses are its foreign buses. A Single bus structure is very simple and consists of a single server.

A bus cannot span multiple cells. And each cell can have more than one buses. - Published messages are printed on it. There is no messaging engine on Single bus structure

In single bus structure all units are connected in the same bus than connecting different buses as multiple bus structure.

Multiple bus structure's performance is better than single bus structure. Iii)single bus structure's cost is cheap than multiple bus structure.

Group of lines that serve as connecting path for several devices is called a bus (one bit per line).

Individual parts must communicate over a communication line or path for exchanging data, address and control information as shown in the diagram below. Printer example – processor to printer. A common approach is to use the concept of buffer registers to hold the content during the transfer.

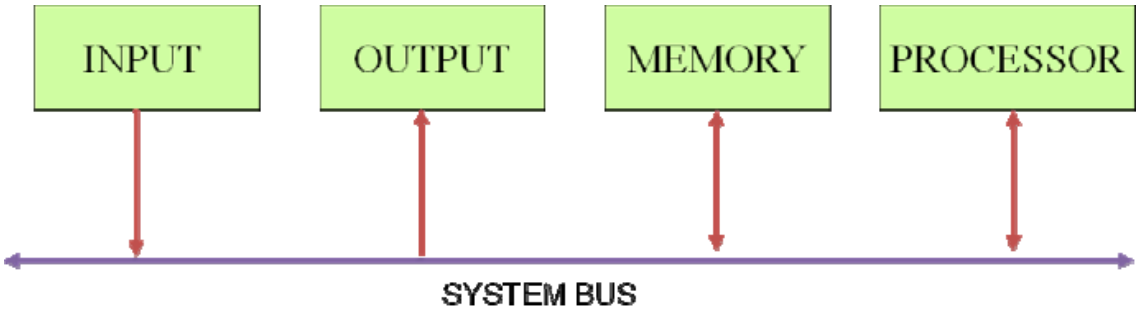


Figure 5: Single bus structure

Buffer registers hold the data during the data transfer temporarily. Ex: printing

Types of Buses:

Data Bus:

Data bus is the most common type of bus. It is used to transfer data between different components of computer. The number of lines in data bus affects the speed of data transfer between different components. The data bus consists of 8, 16, 32, or 64 lines. A 64-line data bus can transfer 64 bits of data at one time.

The data bus lines are bi-directional. It means that:

CPU can read data from memory using these lines CPU can write data to memory locations using these lines

Address Bus:

Many components are connected to one another through buses. Each component is assigned a unique ID. This ID is called the address of that component. It a component wants to communicate with another component, it uses address bus to specify the address of that component. The address bus is a unidirectional bus. It can carry information only in one direction. It carries address of memory location from microprocessor to the main memory.

Control Bus:

Control bus is used to transmit different commands or control signals from one component to another component. Suppose CPU wants to read data from main memory. It will use control is

also used to transmit control signals like ASKS (Acknowledgement signals). A control signal contains the following:

Timing information: It specifies the time for which a device can use data and address bus.

Command Signal: It specifies the type of operation to be performed. Suppose that CPU gives a command to the main memory to write data. The memory sends acknowledgement signal to CPU after writing the data successfully. CPU receives the signal and then moves to perform some other action.

PC PERFORMANCE

What is PC performance measurement?

PC performance measurement is the process of evaluating the performance of a computer system. This can be done for a variety of reasons, such as to compare different systems, to identify bottlenecks, or to track the performance of a system over time.

Performance metrics

There are many different types of PC performance metrics, but some of the most common include:

- CPU performance: This metric measures how well the CPU can execute instructions. It is typically measured in terms of cycles per instruction (CPI) or instructions per second (IPS).
- Memory performance: This metric measures how well the memory system can access data. It is typically measured in terms of bandwidth or latency.
- Disk performance: This metric measures how well the disk system can read and write data. It is typically measured in terms of IOPS (input/output operations per second) or throughput.
- Graphics performance: This metric measures how well the graphics card can render images. It is typically measured in terms of frames per second (FPS).
- Network performance: This metric measures how well the network adapter can send and receive data. It is typically measured in terms of bandwidth or latency.

Basic performance equation

We now focus our attention on the processor time component of the total elapsed time. Let 'T' be the processor time required to execute a program that has been prepared in some high-level language. The compiler generates a machine language object program that corresponds to the source program. Assume that complete execution of the program requires the execution of N machine cycle language instructions. The number N is the actual number of instruction execution and is not necessarily equal to the number of machine cycle instructions in the object program. Some instruction may be executed more than once, which in the case for instructions inside a program loop others may not be executed all, depending on the input data used.

Suppose that the average number of basic steps needed to execute one machine cycle instruction is S, where each basic step is completed in one clock cycle. If clock rate is 'R' cycles per second, the program execution time is given by

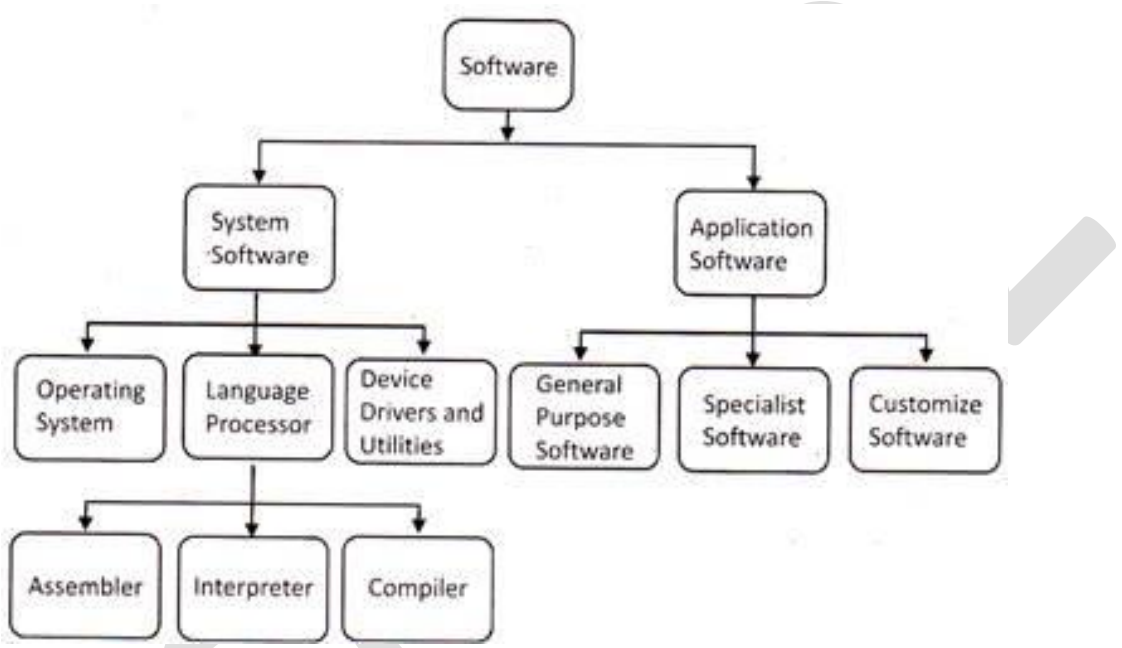
T=N*S/R this is often referred to as the basic performance equation.

We must emphasize that N, S & R are not independent parameters changing one may affect another. Introducing a new feature in the design of a processor will lead to improved performance only if the overall result is to reduce the value of T.

SOFTWARE

Introduction

Software is the set of instructions, data, or programs used to run the computers and enables the computers to perform specific tasks. Software is the opposite of hardware, that is, it cannot be touched. These are essential in running the computers.



Computer Software Classification

Software can be broadly classified into two types:

- System Software
- Application Software.

System Software:

System software is the basic and the most important requirement of computers. System software makes the computer usable as they run the computer hardware, and the computer itself. System software is responsible for controlling and integrating the computer system. System software is also responsible for running application softwares in the computers.

System software consists of utilities, operating system, compilers, and debuggers. The main function of the system software is to provide a platform or working arena for the application software.

Operating system

An operating system (OS) is a software that manages computer hardware and software resources and provides common services for computer programs. The operating system is an essential component of the system software in a computer system. Application programs usually require an operating system to function.

The operating system performs basic tasks, such as:

Booting the system: When you turn on your computer, the operating system is the first software to load. It initializes the hardware and loads the necessary drivers.

- **Process management**: The operating system manages the execution of programs. It keeps track of which programs are running and allocates resources to them.
- **Memory management**: The operating system manages the computer's memory. It allocates memory to programs and keeps track of which parts of memory are being used.
- **File management**: The operating system manages the computer's files. It creates, deletes, and moves files. It also provides access to files for programs.
- Input/output (I/O) management: The operating system manages the computer's I/O devices. It allows programs to read and write data to devices such as disks, printers, and keyboards.
- **Security:** The operating system protects the computer from unauthorized access and malicious software.
- Operating systems are typically classified into two types:
- **Single-user operating systems**: Single-user operating systems are designed for use by a single user at a time. Examples of single-user operating systems include Windows, macOS, and Linux.
- **Multi-user operating systems**: Multi-user operating systems are designed for use by multiple users at the same time. Examples of multi-user operating systems include Unix, Linux, and Windows Server.

Here are some examples of popular operating systems:

Windows: Windows is a family of proprietary graphical operating systems developed by Microsoft. It is the most popular operating system in the world, used by billions of people on personal computers, tablets, and smartphones.

macOS: macOS is a proprietary graphical operating system developed by Apple for its Macintosh computers. It is the second most popular operating system in the world.

Linux: Linux is a family of open-source operating systems based on the Linux kernel. It is used on a wide variety of devices, including personal computers, servers, and embedded systems.

Android: Android is a mobile operating system developed by Google. It is the most popular mobile operating system in the world, used by billions of people on smartphones and tablets.

iOS: iOS is a mobile operating system developed by Apple for its iPhone and iPad devices. It is the second most popular mobile operating system in the world.

Operating systems are complex and sophisticated pieces of software. They are essential for the efficient and reliable operation of modern computers.

Language Translators (language processors).

Language translators are system programs that convert assembly language and high-level language to machine language for the program to execute because Computers work in machine code only,

Programs must be translated into machine/Object codes before execution

There are three types of translators:

- Assemblers
- Compilers
- Interpreters

Programming Languages

A Programming language is the vocabulary and set of grammatical rules for use by people to write instructions for the computer to perform specific tasks.

There are many different programming

Languages each having a unique set of keywords (words that it understands) and a special syntax (grammar) for organizing program instructions.

Examples of common programming languages

Java C#

PHP Object-C

Python

C

C++

Visual basic

Javasript

Ruby

SQL

Classification of Programming Languages

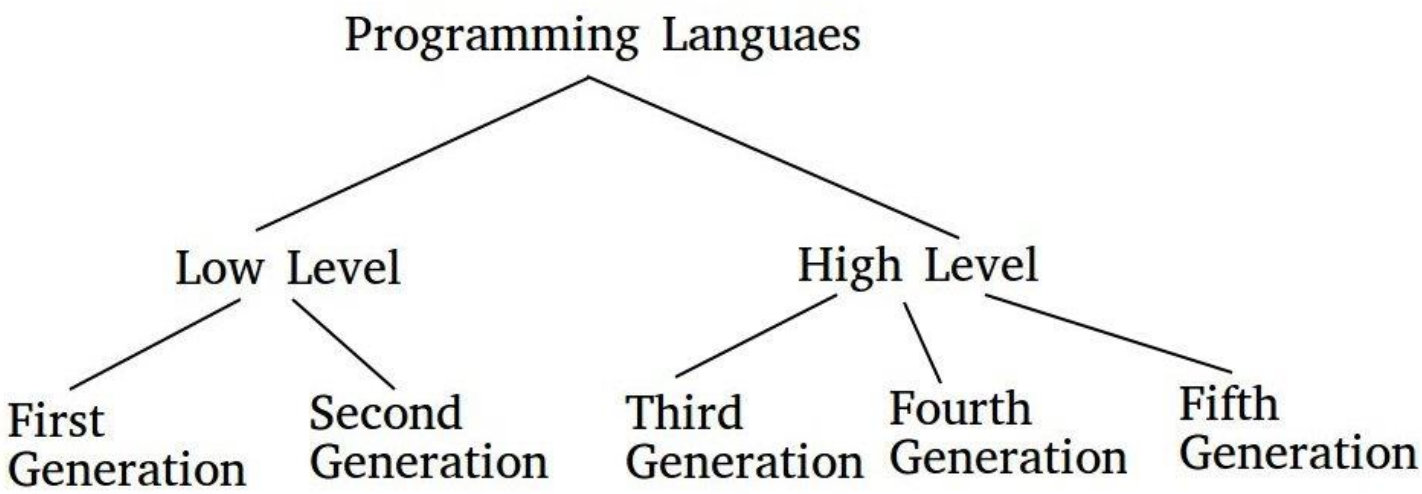
Programming languages are classified into two levels: 1)The low level language is machine language or very close to machine language.

Examples of low level languages are:

Machine languages (first generation languages),

Assembly languages (second generation languages)

Generations of Programming Languages



. Examples of fifth-generation languages include Mercury, OPS5, and Prolog.

Machine language – First Generation Language (1GL): The machine language is low level language that writes programs using the machine code of 1s and 0s, which is directly understood by the computer.

Assembly language – Second Generation Language (2GL): Assembly language is low level symbolic language written using mnemonics (abbreviated sets of letters) or short codes that suggest their meaning and are therefore easier to remember. But must be converted to machine language before the computer can interpret it. An example of a program code to add and store two numbers would be:

LDA A, 20: load accumulator A with the value 20

ADD A,10 : add the value 10 to accumulator A

STO B, A: store contents of accumulator A into storage register B

NOP : no operation (stop here)

Characteristics of 2GL

- Assembly language, being machine dependent, is faster and more efficient in the use of hardware than high-level programming languages.
- Assembly languages have to be translated into machine language by language translators known as assemblers for the processor to understand.
- Easier to write than machine language
- Assembly language is machine dependent. The code is not very easy to understand, hence the introduction of high level programming languages.

Advantages of low level languages

- Very fast to execute because it is already in the language that the computer can understand.
- Require little memory resources takes up less storage space.
- No need of language translator for machine language.
- useful for writing system programs where accuracy is required

Disadvantages of low level languages

- Difficult to interpret by the programmer (requires the aid of a reference manual to interpret the meaning of each code)
- Easy to make mistakes in the sequence of 1s and 0s; replacing a 1 for a 0 can result in the wrong command/instruction being executed
- It is difficult to identify mistakes made
- Time-consuming, slow and tedious to write

- Machine dependent, e.g. one written for an IBM machine cannot work on an Apple machine.
- Makes writing of complex programs difficult.

High-level programming languages

High level programming language is a language that is near to natural language, therefore it is machine independent and uses variables and objects, Boolean expressions, functions, loops, threads, locks which are similar to their meaning (abstraction).

High-level languages have evolved over the years and can be grouped into five categories: Third Generation Languages (3GL), Fourth Generation Languages (4GL), Object Oriented Programming Languages (OOP), Fifth Generation Languages (5GL), Scripting

Third-Generation Languages: These are high-level languages like C, C++, Java, Visual Basic, and JavaScript.

Fourth Generation Languages: These are languages that consist of statements that are similar to statements in the human language. These are used mainly in database programming and scripting. Examples of these languages include Perl, Python, Ruby, SQL, and Mat Lab(MatrixLaboratory).

Fifth Generation Languages: These are the programming languages that have visual tools to develop a program

Advantages of high level programming languages

- High-level language programs are easy to debug
- They are machine independent. Provide programs that can be used on more than one computer.
- They are user friendly and easy to learn because they are near to natural language.
- They are flexible hence they enhance the creativity of the programmer, increasing productivity
- Allows the programmer to focus on understanding the user's needs and design the required software.
- They permit faster development of large programs.

Disadvantages of high level programming languages

- They are executed much slower than low- level programming languages
- They have to be translated into machine code before execution

- Require more memory than the low level languages

Application Software:

Application software is the set of instructions or programs that run on the system software to perform a specific task. It also helps the users to solve various computing problems that one might face in its day to day life. Also, all the works done on the computers include some or the other type of the application software. Some common examples of the application software are (i) MS Word, (ii) Tally, and (iii) VLC Media Player.

Virtual Machines

A virtual machine (VM) is a software computer that, like a physical computer, runs an operating system and applications. But unlike a physical computer, a VM is created within a software layer on top of an existing physical computer. This means that multiple VMs can run on a single physical computer, each with its own operating system and applications.

VMs are created and managed by a virtualization platform, such as VMware ESXi or Microsoft Hyper-V. The virtualization platform allocates the necessary resources, such as CPU, memory, and storage, to each VM. The VM then runs as if it were a physical computer.

VMs have a number of advantages over physical computers, including:

- Efficiency: VMs can help to improve the efficiency of physical computers by allowing multiple VMs to run on a single physical computer. This can help to reduce the number of physical computers that are needed, which can save money and energy.
- Flexibility: VMs are more flexible than physical computers. VMs can be easily created, deleted, and moved between physical computers. This can make it easier to manage and deploy applications.
- Isolation: VMs are isolated from each other. This means that a problem with one VM will not affect the other VMs. This can improve the reliability and security of the overall system.

VMs are used in a variety of applications, including:

- Server consolidation: VMs can be used to consolidate multiple servers onto a single physical server. This can help to reduce the number of physical servers that are needed, which can save money and energy.

- Application development and testing: VMs can be used to develop and test applications in a safe and isolated environment. This can help to prevent problems with applications from affecting other applications or the overall system.
- Desktop virtualization: VMs can be used to provide virtual desktops to users. This can allow users to access their desktops from anywhere, and it can also help to improve the security of the overall system

Human Computer Interaction

Human-Computer Interaction is a multidisciplinary field that focuses on designing and evaluating computer systems and technologies that people interact with. It is concerned with understanding and improving the interaction between humans and computers to make technology more user-friendly, efficient, and enjoyable.

HCI specialists consider how to develop and deploy computer systems that satisfy human users. The majority of this research focuses on enhancing human-computer interaction by enhancing how people utilize and comprehend an interface.

Components of Human-Computer Interaction

HCI is primarily composed of four essential elements:

The User

An individual or a group of individuals who work together on a project is referred to as the user component. HCI researches the needs, objectives, and interaction styles of users.

The Goal-Oriented Task

When using a computer, a user always has a purpose or aim in mind. To achieve this, the computer presents a digital representation of things.

The Interface

An essential HCI element that can improve the quality of user interaction is the interface. Many interface-related factors need to be taken into account, including the type of interaction, screen resolution, display size, and even color contrast.

The Context

HCI is not only about providing better communication between users and computers but also about factoring in the context and environment in which the system is accessed.

Principles of HCI

Researchers and designers in the field of human-computer interaction have established numerous concepts. These regulations range from general norms and design guidelines to abstract design principles. Let's look at the most crucial HCI guidelines.

- Design for familiarity and learnability
- Make the elements readable and approachable.
- Tolerance for errors
- Flexibility

FUTURE TRENDS IN COMPUTER ORGANIZATION AND ARCHITECTURE

The future of computer organization and architecture is likely to be driven by a number of trends, including:

- The increasing demand for performance and efficiency: As the world becomes increasingly digitalized, there is a growing demand for computers that can perform complex tasks quickly and efficiently. This is driving research into new ways to design and build computer systems that can deliver the required performance while also being energy efficient.
- The rise of new technologies: New technologies, such as quantum computing, neuromorphic computing, and photonic computing, have the potential to revolutionize the way that computers are designed and built. These technologies are still in their early stages of development, but they have the potential to significantly improve the performance and efficiency of computer systems in the future.
- The increasing use of cloud computing and artificial intelligence: The increasing use of cloud computing and artificial intelligence is also driving changes in computer organization and architecture. Cloud computing requires computers that can be scaled up or down quickly and easily to meet the changing needs of users. Artificial intelligence requires computers that can process large amounts of data quickly and efficiently. These requirements are driving research into new ways to design and build computer systems that are optimized for cloud computing and artificial intelligence workloads.

Specific examples of future trends in computer organization and architecture:

- **Manycore processors**: Manycore processors are processors with a large number of cores, typically hundreds or thousands. These processors are designed to deliver high performance for parallel workloads.
- Heterogeneous computing: Heterogeneous computing involves using different types of processors, such as CPUs and GPUs, to work together on the same task. This can be used to improve the performance and efficiency of computing for a wide range of applications.
- In-memory computing: In-memory computing involves moving data processing closer to the memory where the data is stored. This can reduce the time it takes to access data and improve the performance of computing for data-intensive workloads.
- Quantum computing: Quantum computers are a new type of computer that uses the principles of quantum mechanics to perform calculations. Quantum computers have the potential to solve problems that are intractable for classical computers.

- Neuromorphic computing: Neuromorphic computers are a new type of computer that is inspired by the structure and function of the human brain. Neuromorphic computers are designed to be very efficient at processing data in parallel.
- Photonic computing: Photonic computers are a new type of computer that uses light to perform calculations. Photonic computers have the potential to be much faster than classical computers