

STRUCTURE OF COMPUTERS: Computer types, functional units, basic operational concepts, Von-Neumann architecture, bus structures, software, performance, multiprocessors and multicomputer

Book: Carl Hamacher, Zvonks Vranesic, SafeaZaky (2002), Computer Organization, 5th edition, McGraw Hill: Unit-1 Pages: 1-23

Data representation, fixed and floating point and error detecting codes.

Book: M. Moris Mano (2006), Computer System Architecture, 3rd edition, Pearson/PHI, India: Unit-3 Pages: 67-91

REGISTER TRANSFER AND MICRO-OPERATIONS: Register transfer language, register transfer, bus and memory transfers, arithmetic micro-operations, logic micro-operations, shift micro-operations, arithmetic logic shift unit.

Book: M. Moris Mano (2006), Computer System Architecture, 3rd edition, Pearson/PHI, India: Unit-3 Pages: 93-118

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).

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.

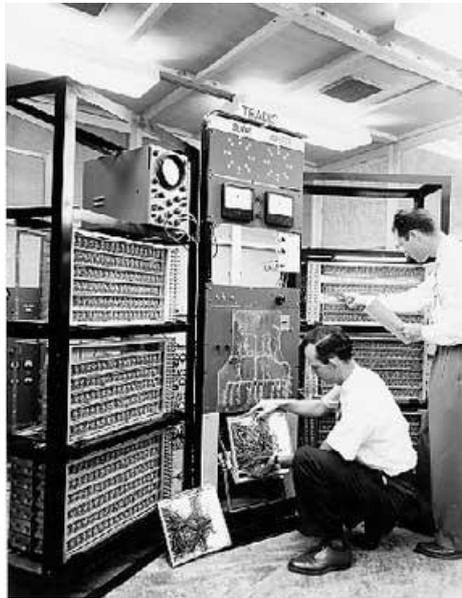
There are totally five computer generations known till date. Each generation has been discussed in detail along with their time period and characteristics. Here approximate dates against each generations have been mentioned which are normally accepted.

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.
2	Second Generation The period of second generation: 1959-1965. Transistor based.
3	Third Generation The period of third generation: 1965-1971. Integrated Circuit based.
4	Fourth Generation The period of fourth generation: 1971-1980. VLSI microprocessor based.
5	Fifth Generation The period of fifth generation: 1980-onwards. ULSI microprocessor based

First generation

The period of first generation was 1946-1959. The computers of first generation used vacuum tubes as the basic components for memory and circuitry for CPU (Central Processing Unit). These tubes, like electric bulbs, produced a lot of heat and were prone to frequent fusing of the installations, therefore, were very expensive and could be afforded only by very large organizations. In this generation mainly batch processing operating system were used. Punched cards, paper tape, and magnetic tape were used as input and output devices. The computers in this generation used machine code as programming language.



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

Some computers of this generation were:

- ENIAC
- EDVAC
- UNIVAC
- IBM-701
- IBM-650

Second generation

The period of second generation was 1959-1965. In this generation transistors were used that were cheaper, consumed less power, more compact in size, more reliable and faster than the first generation machines made of vacuum tubes. In this generation, magnetic cores were used as primary memory and magnetic tape and magnetic disks as secondary storage devices. In this generation assembly language and high-level programming languages like FORTRAN, COBOL were used. The computers used batch processing and multiprogramming operating system.



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

Some computers of this generation were:

- IBM 1620
- IBM 7094
- CDC 1604
- CDC 3600
- UNIVAC 1108

Third generation

The period of third generation was 1965-1971. The computers of third generation used integrated circuits (IC's) in place of transistors. A single IC has many transistors, resistors and capacitors along with the associated circuitry. The IC was invented by Jack Kilby. This development made computers smaller in size, reliable and efficient. In this generation remote processing, time-sharing, multi-programming operating system were used. High-level languages (FORTRAN-II TO IV, COBOL, PASCAL PL/1, BASIC, ALGOL-68 etc.) were used during this generation.



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

Some computers of this generation were:

- IBM-360 series
- Honeywell-6000 series
- PDP(Personal Data Processor)
- IBM-370/168
- TDC-316

Fourth generation

The period of fourth generation was 1971-1980. The computers of fourth generation used Very Large Scale Integrated (VLSI) circuits. VLSI circuits having about 5000 transistors and other circuit elements and their associated circuits on a single chip made it possible to have microcomputers of fourth generation. Fourth generation computers became more powerful, compact, reliable, and affordable. As a result, it gave rise to personal computer (PC) revolution. In this generation time sharing, real time, networks, distributed operating system were used. All the high-level languages like C, C++, DBASE etc., were used in this generation.



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

Some computers of this generation were:

- DEC 10
- STAR 1000
- PDP 11
- CRAY-1(Super Computer)
- CRAY-X-MP(Super Computer)

Fifth generation

The period of fifth generation is 1980-till date. In the fifth generation, the VLSI technology became ULSI (Ultra Large Scale Integration) technology, resulting in the production of microprocessor chips having ten million electronic components. This generation is based on parallel processing hardware and AI (Artificial Intelligence) software. AI is an emerging branch in computer science, which interprets means and method of making computers think like human beings. All the high-level languages like C and C++, Java, .Net etc., are used in this generation.

AI includes:

- Robotics
- Neural Networks
- Game Playing
- Development of expert systems to make decisions in real life situations.
- Natural language understanding and generation.



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

Some computer types of this generation are:

- Desktop
- Laptop
- NoteBook
- UltraBook

- ChromeBook

COMPUTER TYPES

Classification based on Operating Principles

Based on the operating principles, computers can be classified into one of the following types:

-

- 1) Digital Computers
- 2) Analog Computers
- 3) Hybrid Computers

Digital Computers: - Operate essentially by counting. All quantities are expressed as discrete or numbers. Digital computers are useful for evaluating arithmetic expressions and manipulations of data (such as preparation of bills, ledgers, solution of simultaneous equations etc).

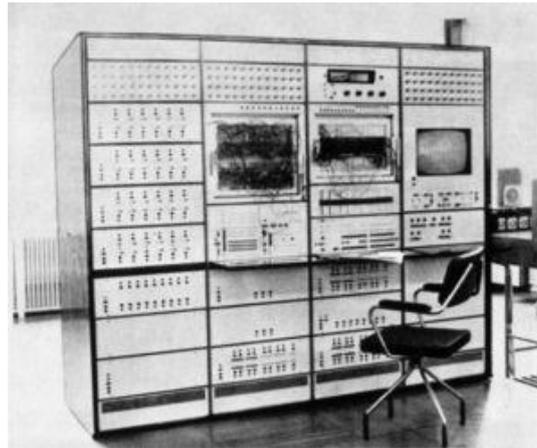


Analog Computers:- An **analog computer** is a form of **computer** that uses the continuously changeable aspects of physical phenomena such as **electrical, mechanical,** or **hydraulic** quantities to **model** the problem being solved. In contrast, **digital computers** represent varying quantities symbolically, as their numerical values change.



Hybrid Computers:- are computers that exhibit features of **analog computers** and **digital computers**. The digital component normally serves as the controller

and provides **logical operations**, while the analog component normally serves as a solver of **differential equations**.



Classification digital Computer based on size and Capability

Based on size and capability, computers are broadly classified into

Micro Computers(Personal Computer)

A microcomputer is the smallest general purpose processing system. The older pc started 8 bit processor with speed of 3.7MB and current pc 64 bit processor with speed of 4.66 GB.

Examples: - **IBM PCs, APPLE** computers

Microcomputer can be classified into 2 types:

1. Desktops
2. Portables

The difference is portables can be used while travelling whereas desktops computers cannot be carried around.

The different portable computers are: -

- 1) Laptop
- 2) Notebooks
- 3) Palmtop (hand held)
- 4) Wearable computers

Laptop: - this computer is similar to a desktop computers but the size is smaller. They are expensive than desktop. The weight of laptop is around 3 to 5 kg.



Notebook: - These computers are as powerful as desktop but size of these computers are comparatively smaller than laptop and desktop. They weigh 2 to 3 kg. They are more costly than laptop.



Palmtop (Hand held): - They are also called as personal Digital Assistant (PDA). These computers are small in size. They can be held in hands. It is capable of doing word processing, spreadsheets and hand writing recognition, game playing, faxing and paging. These computers are not as powerful as desktop computers. Ex: - 3com palmV.



Wearable computer: - The size of this computer is very small so that it can be worn on the body. It has smaller processing power. It is used in the field of medicine. For example pace maker to correct the heart beats. Insulin meter to find the levels of insulin in the blood.



Workstations:- It is used in large, high-resolution graphics screen built in network support, Engineering applications(CAD/CAM), software development desktop publishing

Ex: Unix and windows NT.

b) Minicomputer: - A minicomputer is a medium-sized computer. That is more powerful than a microcomputer. These computers are usually designed to serve multiple users simultaneously (Parallel Processing). They are more expensive than microcomputers.

Examples: Digital Alpha, Sun Ultra.



c) Mainframe (Enterprise) computers: - Computers with large storage capacities and very high speed of processing (compared to mini- or microcomputers) are known as mainframe computers. They support a large number of terminals for simultaneous use by a number of users like ATM transactions. They are also used as central host computers in distributed data processing system.

Examples: - **IBM 370, S/390.**



d) Supercomputer: - Supercomputers have extremely large storage capacity and computing speeds which are many times faster than other computers. A supercomputer is measured in terms of tens of millions Instructions per second (mips), an operation is made up of numerous instructions. The supercomputer is mainly used for large scale numerical problems in scientific and engineering disciplines such as Weather analysis.

Examples: - **IBM Deep Blue**



Classification based on number of microprocessors

Based on the number of microprocessors, computers can be classified into

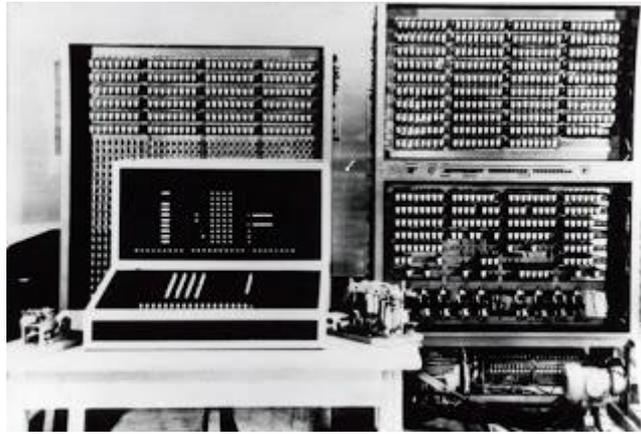
- a) Sequential computers and
- b) Parallel computers

a) Sequential computers: - Any task complete in sequential computers is with one microcomputer only. Most of the computers (today) we see are sequential computers where in any task is completed sequentially instruction after instruction from the beginning to the end.

b) Parallel computers: - The parallel computer is relatively fast. New types of computers that use a large number of processors. The processors perform different tasks independently and simultaneously thus improving the speed of execution of complex programs dramatically. Parallel computers match the speed of supercomputers at a fraction of the cost.

Classification based on word-length

A binary digit is called “**BIT**”. A word is a group of bits which is fixed for a computer. The number of bits in a word (or word length) determines the representation of all characters in these many bits. Word length leis in the range from 16-bit to 64-bitsf or most computers of today.



Classification based on number of users

Based on number of users, computers are classified into: -

Single User: - Only one user can use the resource at any time.



Multi User: - A single computer shared by a number of users at any time.



Network: - A number of interconnected autonomous computers shared by a number of users at any time.



COMPUTER TYPES

A computer can be defined as a fast electronic calculating machine that accepts the (data) digitized input information process it as per the list of internally stored instructions and produces the resulting information. List of instructions are called programs & internal storage is called computer memory.

The different types of computers are

- 1. Personal computers:** - This is the most common type found in homes, schools, Business offices etc., It is the most common type of desk top computers with processing and storage units along with various input and output devices.
- 2. Note book computers:** - These are compact and portable versions of PC
- 3. Work stations:** - These have high resolution input/output (I/O) graphics capability, but with same dimensions as that of desktop computer. These are used in engineering applications of interactive design work.
- 4. Enterprise systems:** - These are used for business data processing in medium to large corporations that require much more computing power and storage capacity than work stations. Internet associated with servers have become a dominant worldwide source of all types of information.
- 5. Super computers:** - These are used for large scale numerical calculations required in the applications like weather forecasting etc.,

BASIC TERMINOLOGY

- Input: Whatever is put into a computer system.
- Data: Refers to the symbols that represent facts, objects, or ideas.
- Information: The results of the computer storing data as bits and bytes; the words, numbers, sounds, and graphics.
- Output: Consists of the processing results produced by a computer.
- Processing: Manipulation of the data in many ways.
- Memory: Area of the computer that temporarily holds data waiting to be processed, stored, or output.
- Storage: Area of the computer that holds data on a permanent basis when it is not immediately needed for processing.
- Assembly language program (ALP) –Programs are written using mnemonics
- Mnemonic –Instruction will be in the form of English like form
- Assembler –is a software which converts ALP to MLL (Machine Level Language)
- HLL (High Level Language) –Programs are written using English like statements
- Compiler –Convert HLL to MLL, does this job by reading source program at once
- Interpreter –Converts HLL to MLL, does this job statement by statement
- System software –Program routines which aid the user in the execution of programs eg: Assemblers, Compilers
- Operating system –Collection of routines responsible for controlling and coordinating all the activities in a computer system

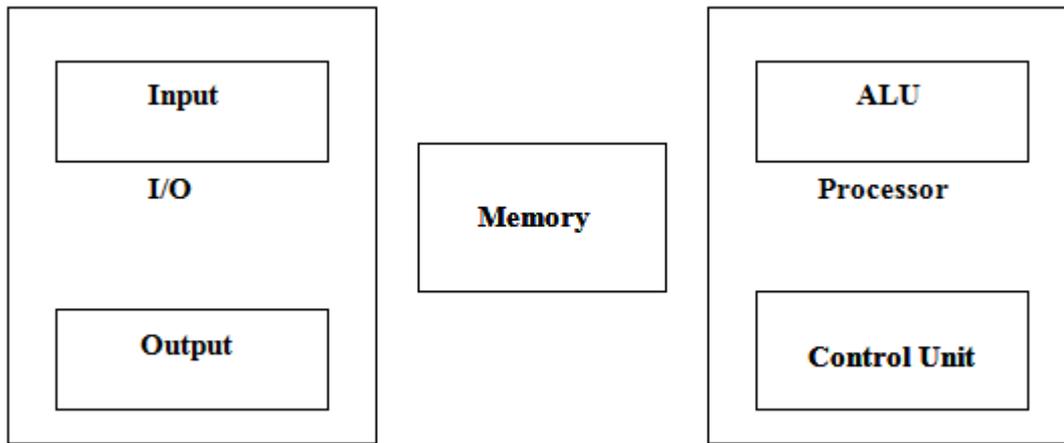
Computers has two kinds of components:

Hardware, consisting of its physical devices (CPU, memory, bus, storage devices, ...)

Software, consisting of the programs it has (Operating system, applications, utilities, ...)

FUNCTIONAL UNIT

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

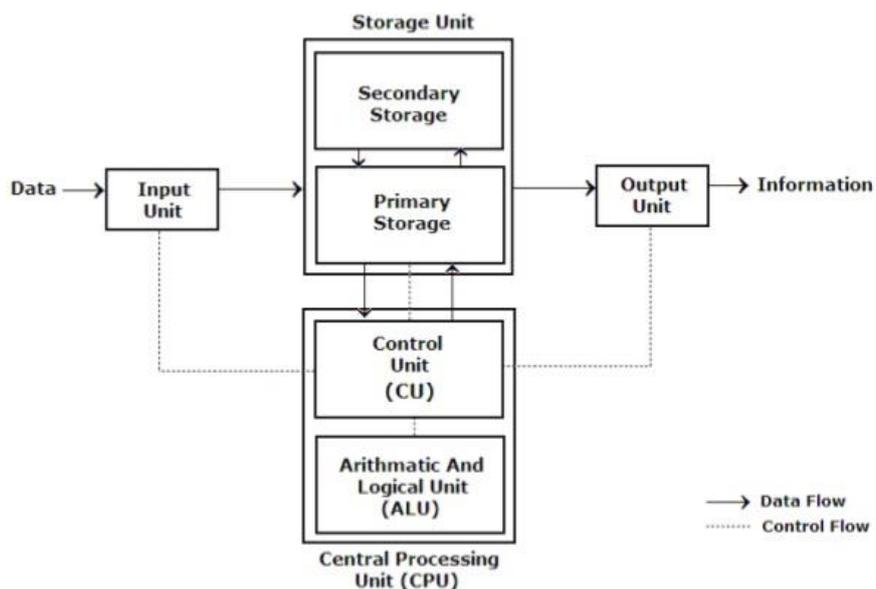


Functional units of computer

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.

Block diagram of computer



Input 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

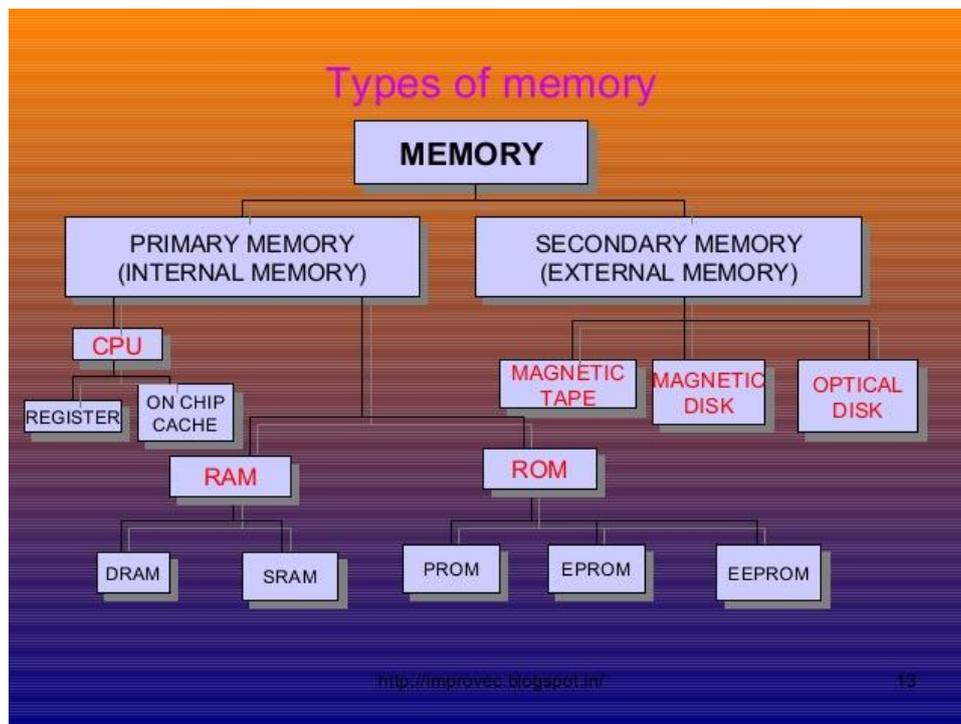
1. **Primary memory**
2. **Secondary memory**

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 size 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 is 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 many 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, R₀

This instruction adds the operand at memory location LOCA, to operand in register R₀ & 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 R₀
3. Finally the resulting sum is stored in the register R₀

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.

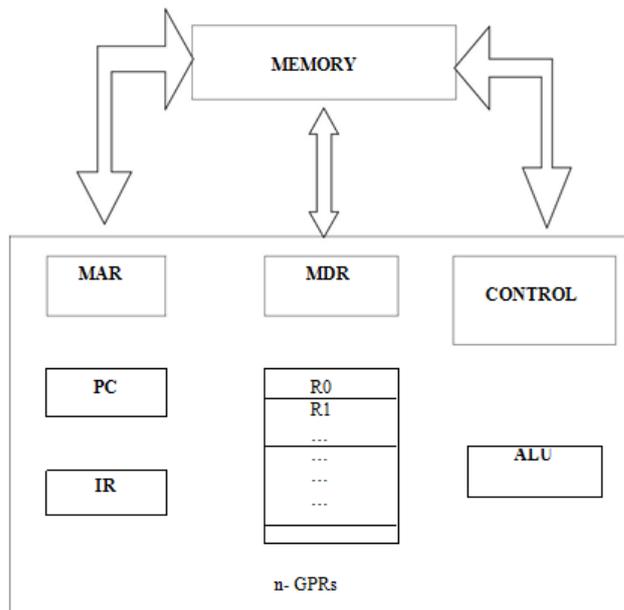


Fig b : Connections between the processor and the memory

The fig

shows how memory &

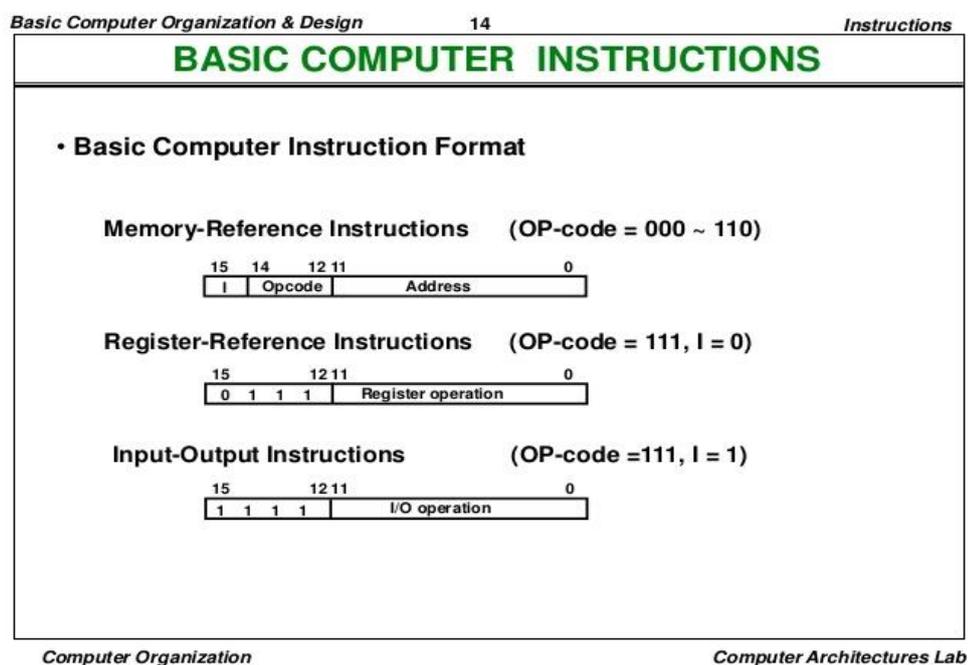
the processor can be connected. In addition to the ALU & the control circuitry, the processor contains a number of registers used for several different purposes.

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:

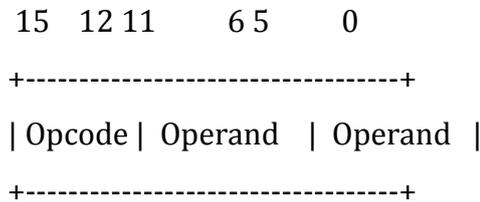


Computer instructions are the basic components of a machine language program. They are also known as *macro operations*, since each one is comprised of sequences of micro operations.

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.

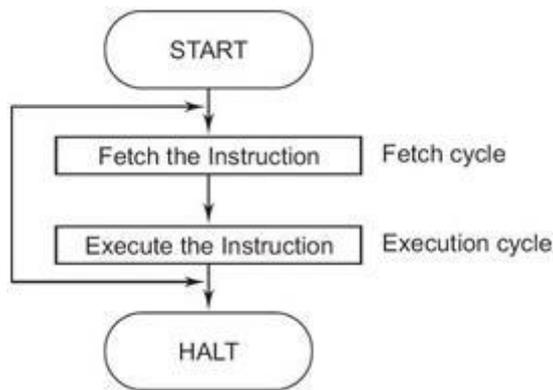
In addition to the opcode, many instructions also contain one or more *operands*, which indicate where in registers or memory the data required for the operation is located. For example, and add instruction requires two operands, and a not instruction requires one.

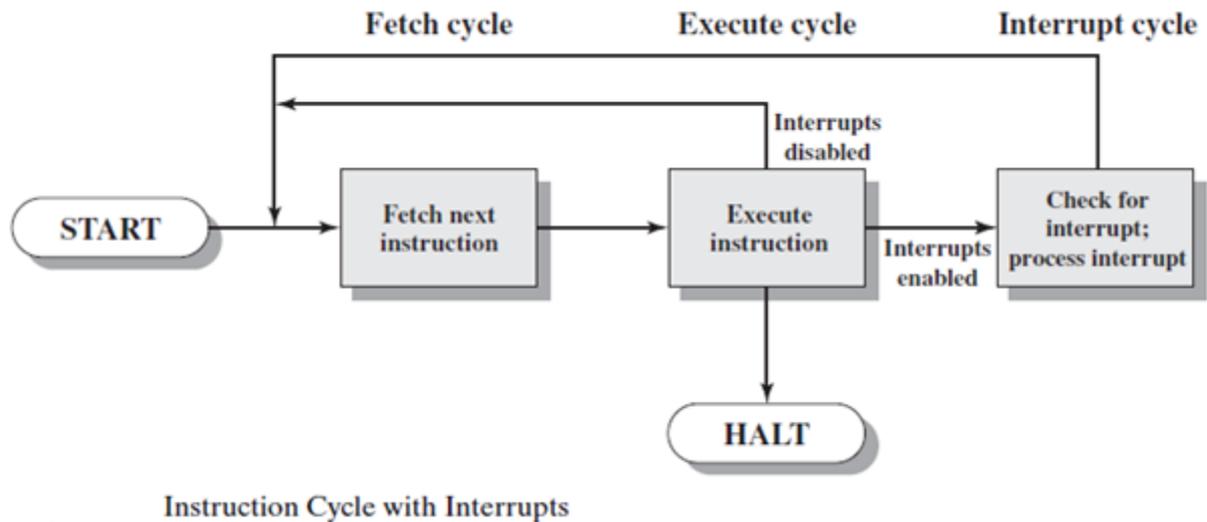


The opcode and operands are most often encoded as unsigned binary numbers in order to minimize the number of bits used to store them. For example, a 4-bit opcode encoded as a binary number could represent up to 16 different operations.

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:





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 R_0 through R_{n-1} .

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

1. **MAR - (Memory Address Register):-** It holds the address of the location to be accessed.
2. **MDR - (Memory Data Register):-** It contains the data to be written into or read out of the address location.

Operating steps are

1. Programs reside in the memory & usually get these through the I/P unit.
2. Execution of the program starts when the PC is set to point at the first instruction of the program.
3. Contents of PC are transferred to MAR and a Read Control Signal is sent to the memory.

4. After the time required to access the memory elapses, the address word is read out of the memory and loaded into the MDR.
5. Now contents of MDR are transferred to the IR & now the instruction is ready to be decoded and executed.
6. If the instruction involves an operation by the ALU, it is necessary to obtain the required operands.
7. An operand in the memory is fetched by sending its address to MAR & Initiating a read cycle.
8. When the operand has been read from the memory to the MDR, it is transferred from MDR to the ALU.
9. After one or two such repeated cycles, the ALU can perform the desired operation.
10. If the result of this operation is to be stored in the memory, the result is sent to MDR.
11. Address of location where the result is stored is sent to MAR & a write cycle is initiated.
12. 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 as 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.

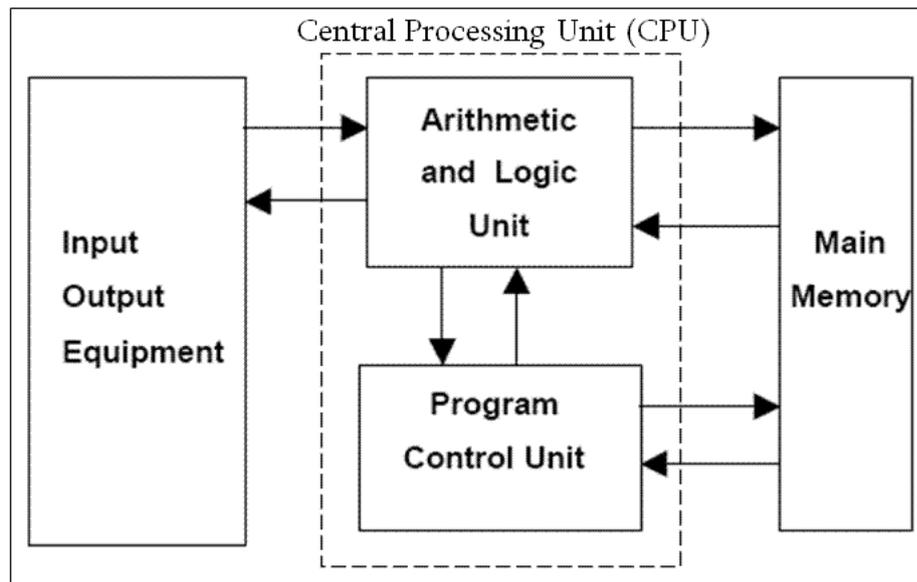


Figure : General structure of Von Neumann Architecture

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

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

II) 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.

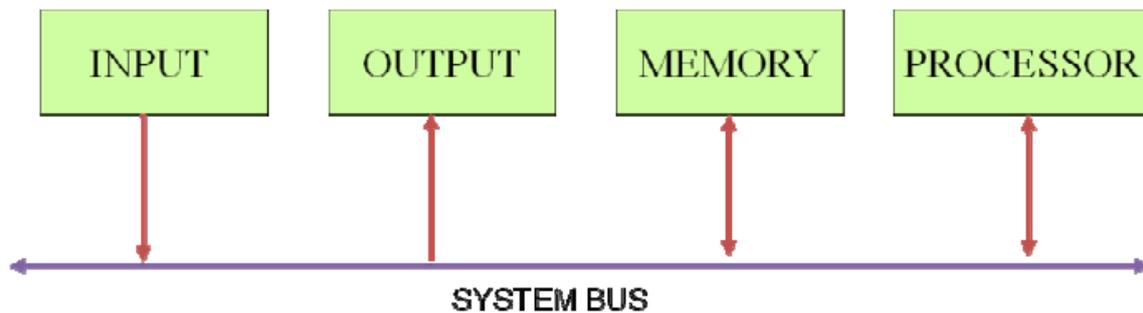


Figure 5: Single bus structure

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

Types of Buses:

1. 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

2. 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. If 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.

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

- 1 Timing information: It specifies the time for which a device can use data and address bus.
- 2 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.

SOFTWARE

If a user wants to enter and run an application program, he/she needs a System Software. System Software is a collection of programs that are executed as needed to perform functions such as:

- Receiving and interpreting user commands
- Entering and editing application programs and storing them as files in secondary storage devices
- Running standard application programs such as word processors, spread sheets, games etc...

Operating system - is key system software component which helps the user to exploit the below underlying hardware with the programs.

Types of software

A layer structure showing where Operating System is located on generally used software systems on desktops

System software

System software helps run the computer hardware and computer system. It includes a combination of the following:

- device drivers
- operating systems
- servers
- utilities
- windowing systems
- compilers
- debuggers
- interpreters
- linkers

The purpose of systems software is to unburden the applications programmer from the often complex details of the particular computer being used, including such accessories as communications devices, printers, device readers, displays and keyboards, and also to partition the computer's resources such as memory and processor time in a safe and stable manner. Examples are- Windows XP, Linux and Mac.

Application software

Application software allows end users to accomplish one or more specific (not directly computer development related) tasks. Typical applications include:

- ☒ business software
- ☒ computer games
- ☒ quantum chemistry and solid state physics software
- ☒ telecommunications (i.e., the internet and everything that flows on it)
- ☒ databases
- ☒ educational software
- ☒ medical software
- ☒ military software

- ☒ molecular modeling software
- ☒ image editing
- ☒ spreadsheet
- ☒ simulation software
- ☒ Word processing
- ☒ Decision making software

Application software exists for and has impacted a wide variety of topics.

PERFORMANCE

The most important measure of the performance of a computer is how quickly it can execute programs. The speed with which a computer executes program is affected by the design of its hardware. For best performance, it is necessary to design the compiles, the machine instruction set, and the hardware in a coordinated way.

The total time required to execute the program is elapsed time is a measure of the performance of the entire computer system. It is affected by the speed of the processor, the disk and the printer. The time needed to execute a instruction is called the processor time.

Just as the elapsed time for the execution of a program depends on all units in a computer system, the processor time depends on the hardware involved in the execution of individual machine instructions. This hardware comprises the processor and the memory which are usually connected by the bus as shown in the fig c.

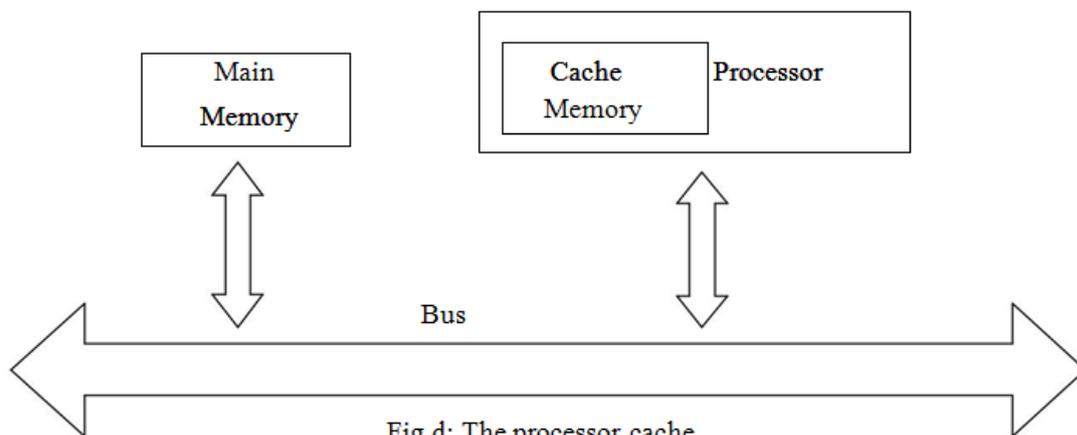


Fig d: The processor cache

The pertinent parts of the fig. c are repeated in fig. d which includes the cache memory as part of the processor unit.

Let us examine the flow of program instructions and data between the memory and the processor. At the start of execution, all program instructions and the required data are stored in the main memory. As the execution proceeds, instructions are fetched one by one over the bus into the processor, and a copy is placed in the cache later if the same instruction or data item is needed a second time, it is read directly from the cache.

The processor and relatively small cache memory can be fabricated on a single IC chip. The internal speed of performing the basic steps of instruction processing on chip is very high and is considerably faster than the speed at which the instruction and data can be fetched from the main memory. A program will be executed faster if the movement of instructions and data between the main memory and the processor is minimized, which is achieved by using the cache.

For example:- Suppose a number of instructions are executed repeatedly over a short period of time as happens in a program loop. If these instructions are available in the cache, they can be fetched quickly during the period of repeated use. The same applies to the data that are used repeatedly.

Processor clock: -

Processor circuits are controlled by a timing signal called clock. The clock designer the regular time intervals called clock cycles. To execute a machine instruction the processor divides the action to be performed into a sequence of basic steps that each step can be completed in one clock cycle. The length P of one clock cycle is an important parameter that affects the processor performance.

Processor used in today's personal computer and work station have a clock rates that range from a few hundred million to over a billion cycles per second.

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 .

Pipelining and super scalar operation: -

We assume that instructions are executed one after the other. Hence the value of S is the total number of basic steps, or clock cycles, required to execute one instruction. A substantial improvement in performance can be achieved by overlapping the execution of successive instructions using a technique called pipelining.

Consider Add R_1 R_2 R_3

This adds the contents of R_1 & R_2 and places the sum into R_3 .

The contents of R_1 & R_2 are first transferred to the inputs of ALU. After the addition operation is performed, the sum is transferred to R_3 . The processor can read the next instruction from the memory, while the addition operation is being performed. Then of that instruction also uses, the ALU, its operand can be transferred to the ALU inputs at the same time that the add instructions is being transferred to R_3 .

In the ideal case if all instructions are overlapped to the maximum degree possible the execution proceeds at the rate of one instruction completed in each clock cycle.

Individual instructions still require several clock cycles to complete. But for the purpose of computing T , effective value of S is 1.

A higher degree of concurrency can be achieved if multiple instructions pipelines are implemented in the processor. This means that multiple functional units are used creating parallel paths through which different instructions can be executed in parallel with such an arrangement, it becomes possible to start the execution of several instructions in every clock cycle. This mode of operation is called superscalar execution. If it can be sustained for a long time during program execution the effective value of S can be reduced to less than one. But the parallel execution must preserve logical correctness of programs that is the results produced must be same as those produced by the serial execution of program instructions. Now days many processors are designed in this manner.

Clock rate

These are two possibilities for increasing the clock rate 'R'.

1. Improving the IC technology makes logical circuit faster, which reduces the time of execution of basic steps. This allows the clock period P , to be reduced and the clock rate R to be increased.
2. Reducing the amount of processing done in one basic step also makes it possible to reduce the clock period P . however if the actions that have to be performed by an instructions remain the same, the number of basic steps needed may increase.

Increase in the value 'R' that are entirely caused by improvements in IC technology affects all aspects of the processor's operation equally with the exception of the time it takes to access the main memory. In the presence of cache the percentage of accesses to the main memory is small. Hence much of the performance gain excepted from the use of faster technology can be realized.

Instruction set CISC & RISC:-

Simple instructions require a small number of basic steps to execute. Complex instructions involve a large number of steps. For a processor that has only simple instruction a large number of instructions may be needed to perform a given programming task. This could lead to a large value of 'N' and a small value of 'S' on the other hand if individual instructions perform more complex operations, a fewer instructions will be needed, leading

to a lower value of N and a larger value of S. It is not obvious if one choice is better than the other.

But complex instructions combined with pipelining (effective value of $S \approx 1$) would achieve one best performance. However, it is much easier to implement efficient pipelining in processors with simple instruction sets.

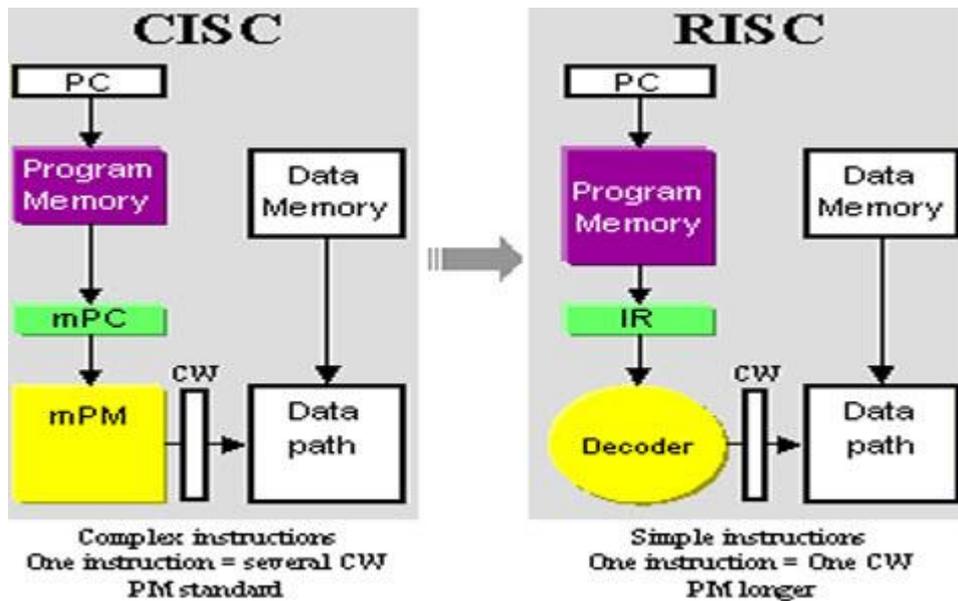
RISC and CISC are computing systems developed for computers. Instruction set or instruction set architecture is the structure of the computer that provides commands to the computer to guide the computer for processing data manipulation. Instruction set consists of instructions, addressing modes, native data types, registers, interrupt, exception handling and memory architecture. Instruction set can be emulated in software by using an interpreter or built into hardware of the processor. Instruction Set Architecture can be considered as a boundary between the software and hardware. [Classification of microcontrollers](#) and microprocessors can be done based on the RISC and CISC instruction set architecture.

Comparison between RISC and CISC:

	RISC	CISC
Acronym	It stands for 'Reduced Instruction Set Computer'.	It stands for 'Complex Instruction Set Computer'.
Definition	The RISC processors have a smaller set of instructions with few addressing nodes.	The CISC processors have a larger set of instructions with many addressing nodes.
Memory unit	It has no memory unit and uses a separate hardware to implement instructions.	It has a memory unit to implement complex instructions.
Program	It has a hard-wired unit of programming.	It has a micro-programming unit.
Design	It is a complex compiler design.	It is an easy compiler design.
Calculations	The calculations are faster and precise.	The calculations are slow and precise.
Decoding	Decoding of instructions is simple.	Decoding of instructions is complex.
Time	Execution time is very less.	Execution time is very high.
External	It does not require external	It requires external memory

memory	memory for calculations.	for calculations.
Pipelining	Pipelining does function correctly.	Pipelining does not function correctly.
Stalling	Stalling is mostly reduced in processors.	The processors often stall.
Code expansion	Code expansion can be a problem.	Code expansion is not a problem.
Disc space	The space is saved.	The space is wasted.
Applications	Used in high end applications such as video processing, telecommunications and image processing.	Used in low end applications such as security systems, home automations, etc.

CISC	RISC
Emphasis on hardware	Emphasis on software
Multiple instruction sizes and formats	Instructions of same set with few formats
Less registers	Uses more registers
More addressing modes	Fewer addressing modes
Extensive use of microprogramming	Complexity in compiler
Instructions take a varying amount of cycle time	Instructions take one cycle time
Pipelining is difficult	Pipelining is easy



1.8 Performance measurements

The performance measure is the time taken by the computer to execute a given bench mark. Initially some attempts were made to create artificial programs that could be used as bench mark programs. But synthetic programs do not properly predict the performance obtained when real application programs are run.

A non-profit organization called SPEC- system performance Evaluation Corporation selects and publishes bench marks.

The program selected range from game playing, compiler, and data base applications to numerically intensive programs in astrophysics and quantum chemistry. In each case, the program is compiled under test, and the running time on a real computer is measured. The same program is also compiled and run on one computer selected as reference.

The 'SPEC' rating is computed as follows.

SPEC rating = Running time on the reference computer / Running time on the computer under test

MULTIPROCESSORS AND MULTICOMPUTER

Multiprocessors and Multicomputers



- Multiprocessor computer
 - Execute a number of different application tasks in parallel
 - Execute subtasks of a single large task in parallel
 - All processors have access to all of the memory – shared-memory multiprocessor
 - Cost – processors, memory units, complex interconnection networks
- Multicomputers
 - Each computer only have access to its own memory
 - Exchange message via a communication network – message-passing multicomputers

multicomputer	multiprocessors
1. A computer made up of several computers. 2. Distributed computing deals with hardware and software systems containing more than one processing element, multiple programs 3. It can run faster 4. A multi-computer is multiple computers, each of which can have multiple processors. 5. Used for true parallel processing. 6. Processor can not share the memory. 7. Called as message passing multi computers 8. Cost is more	1. A computer that has more than one CPU on its motherboard. 2. Multiprocessing is the use of two or more central processing units (CPUs) within a single computer system. 3. Speed depends on the all processors speed 4. Single Computer with multiple processors 5. Used for true parallel processing. 6. Processors can share the memory. 7. Called as shared memory multi processors 8. Cost is low

Registers are made up of flip-flops and flip-flops are two-state devices that can store only 1's and 0's.

Numbering Systems		
System	Base	Digits
Binary	2	0 1
Octal	8	0 1 2 3 4 5 6 7
Decimal	10	0 1 2 3 4 5 6 7 8 9
Hexadecimal	16	0 1 2 3 4 5 6 7 8 9 A B C D E F

There are many methods or techniques which can be used to convert numbers from one base to another. We'll demonstrate here the following –

- Decimal to Other Base System
- Other Base System to Decimal
- Other Base System to Non-Decimal
- Shortcut method – Binary to Octal
- Shortcut method – Octal to Binary
- Shortcut method – Binary to Hexadecimal
- Shortcut method – Hexadecimal to Binary

Decimal to Other Base System

Steps

- **Step 1** – Divide the decimal number to be converted by the value of the new base.
- **Step 2** – Get the remainder from Step 1 as the rightmost digit (least significant digit) of new base number.
- **Step 3** – Divide the quotient of the previous divide by the new base.
- **Step 4** – Record the remainder from Step 3 as the next digit (to the left) of the new base number.

Repeat Steps 3 and 4, getting remainders from right to left, until the quotient becomes zero in Step 3.

The last remainder thus obtained will be the Most Significant Digit (MSD) of the new base number.

Example –

Decimal Number: 29_{10}

Calculating Binary Equivalent –

Step	Operation	Result	Remainder
Step 1	29 / 2	14	1
Step 2	14 / 2	7	0
Step 3	7 / 2	3	1
Step 4	3 / 2	1	1
Step 5	1 / 2	0	1

As mentioned in Steps 2 and 4, the remainders have to be arranged in the reverse order so that the first remainder becomes the Least Significant Digit (LSD) and the last remainder becomes the Most Significant Digit (MSD).

Decimal Number – 29_{10} = Binary Number – 11101_2 .

Other Base System to Decimal System

Steps

- **Step 1** – Determine the column (positional) value of each digit (this depends on the position of the digit and the base of the number system).
- **Step 2** – Multiply the obtained column values (in Step 1) by the digits in the corresponding columns.
- **Step 3** – Sum the products calculated in Step 2. The total is the equivalent value in decimal.

Step	Binary Number	Decimal Number
Step 1	11101_2	$((1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0))_{10}$
Step 2	11101_2	$(16 + 8 + 4 + 0 + 1)_{10}$
Step 3	11101_2	29_{10}

Example

Binary Number – 11101_2

Calculating Decimal Equivalent –

Binary Number – 11101_2 = Decimal Number – 29_{10}

Other Base System to Non-Decimal System

Steps

- **Step 1** – Convert the original number to a decimal number (base 10).
- **Step 2** – Convert the decimal number so obtained to the new base number.

Example

Octal Number – 25_8

Calculating Binary Equivalent –

Step 1 – Convert to Decimal

Step	Octal Number	Decimal Number
Step 1	25_8	$((2 \times 8^1) + (5 \times 8^0))_{10}$
Step 2	25_8	$(16 + 5)_{10}$
Step 3	25_8	21_{10}

Octal Number – $25_8 =$ Decimal Number – 21_{10}

Step 2 – Convert Decimal to Binary

Step	Operation	Result	Remainder
Step 1	$21 / 2$	10	1
Step 2	$10 / 2$	5	0
Step 3	$5 / 2$	2	1
Step 4	$2 / 2$	1	0
Step 5	$1 / 2$	0	1

Decimal Number – $21_{10} =$ Binary Number – 10101_2

Octal Number – $25_8 =$ Binary Number – 10101_2

Shortcut method - Binary to Octal

Steps

- **Step 1** – Divide the binary digits into groups of three (starting from the right).
- **Step 2** – Convert each group of three binary digits to one octal digit.

Example

Binary Number – 10101_2

Calculating Octal Equivalent –

Step	Binary Number	Octal Number
Step 1	10101 ₂	010 101
Step 2	10101 ₂	2 ₈ 5 ₈
Step 3	10101 ₂	25 ₈

Binary Number – 10101₂ = Octal Number – 25₈

Shortcut method - Octal to Binary

Steps

- **Step 1** – Convert each octal digit to a 3 digit binary number (the octal digits may be treated as decimal for this conversion).
- **Step 2** – Combine all the resulting binary groups (of 3 digits each) into a single binary number.

Example

Octal Number – 25₈

Calculating Binary Equivalent –

Step	Octal Number	Binary Number
Step 1	25 ₈	2 ₁₀ 5 ₁₀
Step 2	25 ₈	010 ₂ 101 ₂
Step 3	25 ₈	010101 ₂

Octal Number – 25₈ = Binary Number – 10101₂

Shortcut method - Binary to Hexadecimal

Steps

- **Step 1** – Divide the binary digits into groups of four (starting from the right).
- **Step 2** – Convert each group of four binary digits to one hexadecimal symbol.

Example

Binary Number – 10101₂

Calculating hexadecimal Equivalent –

Step	Binary Number	Hexadecimal Number
Step 1	10101 ₂	0001 0101

Step 2	10101 ₂	1 ₁₀ 5 ₁₀
Step 3	10101 ₂	15 ₁₆

Binary Number – 10101₂ = Hexadecimal Number – 15₁₆

Shortcut method - Hexadecimal to Binary

Steps

- **Step 1** – Convert each hexadecimal digit to a 4 digit binary number (the hexadecimal digits may be treated as decimal for this conversion).
- **Step 2** – Combine all the resulting binary groups (of 4 digits each) into a single binary number.

Example

Hexadecimal Number – 15₁₆

Calculating Binary Equivalent –

Step	Hexadecimal Number	Binary Number
Step 1	15 ₁₆	1 ₁₀ 5 ₁₀
Step 2	15 ₁₆	0001 ₂ 0101 ₂
Step 3	15 ₁₆	00010101 ₂

Hexadecimal Number – 15₁₆ = Binary Number – 10101₂

Binary Coded Decimal (BCD) code

In this code each decimal digit is represented by a 4-bit binary number. BCD is a way to express each of the decimal digits with a binary code. In the BCD, with four bits we can represent sixteen numbers (0000 to 1111). But in BCD code only first ten of these are used (0000 to 1001). The remaining six code combinations i.e. 1010 to 1111 are invalid in BCD.

Decimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Advantages of BCD Codes

- It is very similar to decimal system.

- We need to remember binary equivalent of decimal numbers 0 to 9 only.

Disadvantages of BCD Codes

- The addition and subtraction of BCD have different rules.
- The BCD arithmetic is little more complicated.
- BCD needs more number of bits than binary to represent the decimal number. So BCD is less efficient than binary.

Alphanumeric codes

A binary digit or bit can represent only two symbols as it has only two states '0' or '1'. But this is not enough for communication between two computers because there we need many more symbols for communication. These symbols are required to represent 26 alphabets with capital and small letters, numbers from 0 to 9, punctuation marks and other symbols.

The alphanumeric codes are the codes that represent numbers and alphabetic characters. Mostly such codes also represent other characters such as symbol and various instructions necessary for conveying information. An alphanumeric code should at least represent 10 digits and 26 letters of alphabet i.e. total 36 items. The following three alphanumeric codes are very commonly used for the data representation.

- American Standard Code for Information Interchange (ASCII).
- Extended Binary Coded Decimal Interchange Code (EBCDIC).
- Five bit Baudot Code.

ASCII code is a 7-bit code whereas EBCDIC is an 8-bit code. ASCII code is more commonly used worldwide while EBCDIC is used primarily in large IBM computers.

Complement Arithmetic

Complements are used in the digital computers in order to simplify the subtraction operation and for the logical manipulations. For each radix-r system (radix r represents base of number system) there are two types of complements.

S.N.	Complement	Description
1	Radix Complement	The radix complement is referred to as the r's complement
2	Diminished Radix Complement	The diminished radix complement is referred

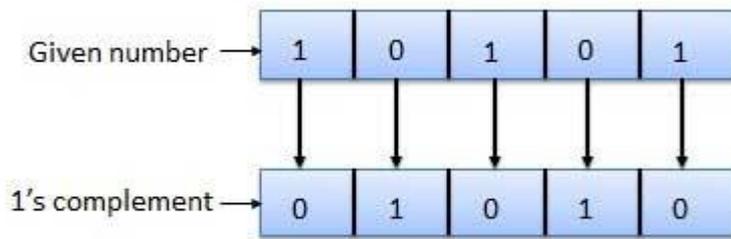
	to as the (r-1)'s complement
--	------------------------------

Binary system complements

As the binary system has base $r = 2$. So the two types of complements for the binary system are 2's complement and 1's complement.

1's complement

The 1's complement of a number is found by changing all 1's to 0's and all 0's to 1's. This is called as taking complement or 1's complement. Example of 1's Complement is as follows.

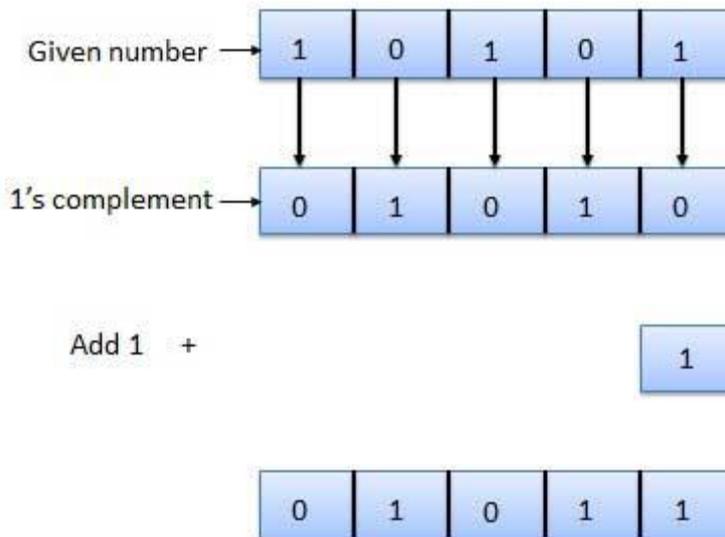


2's complement

The 2's complement of binary number is obtained by adding 1 to the Least Significant Bit (LSB) of 1's complement of the number.

2's complement = 1's complement + 1

Example of 2's Complement is as follows.



Binary Arithmetic

Binary arithmetic is essential part of all the digital computers and many other digital system.

Binary Addition

It is a key for binary subtraction, multiplication, division. There are four rules of binary addition.

Case	A + B	Sum	Carry
1	0 + 0	0	0
2	0 + 1	1	0
3	1 + 0	1	0
4	1 + 1	0	1

In fourth case, a binary addition is creating a sum of (1 + 1 = 10) i.e. 0 is written in the given column and a carry of 1 over to the next column.

Example – Addition

$$0011010 + 0011100 = 00100110$$

1 1	carry	
0 0 1 1 0 1 0	=	26 ₁₀
+ 0 0 0 1 1 0 0	=	12 ₁₀
0 1 0 0 1 1 0 = 38 ₁₀		

Binary Subtraction

Subtraction and Borrow, these two words will be used very frequently for the binary subtraction. There are four rules of binary subtraction.

Case	A - B	Subtract	Borrow
1	0 - 0	0	0
2	1 - 0	1	0
3	1 - 1	0	0
4	0 - 1	0	1

Example – Subtraction

$$0011010 - 0011100 = 00001110$$

1 1	borrow	
0 0 1 0 1 0	=	26 ₁₀
- 0 0 0 1 1 0 0	=	12 ₁₀
0 0 0 1 1 1 0 = 14 ₁₀		

Binary Multiplication

Binary multiplication is similar to decimal multiplication. It is simpler than decimal multiplication because only 0s and 1s are involved. There are four rules of binary multiplication.

Case	A	x	B	Multiplication
1	0	x	0	0
2	0	x	1	0
3	1	x	0	0
4	1	x	1	1

Example – Multiplication

Example:

$$0011010 \times 001100 = 100111000$$

$$\begin{array}{r}
 0011010 = 26_{10} \\
 \times 0001100 = 12_{10} \\
 \hline
 0000000 \\
 0000000 \\
 0011010 \\
 0011010 \\
 \hline
 0100111000 = 312_{10}
 \end{array}$$

Binary Division

Binary division is similar to decimal division. It is called as the long division procedure.

Example – Division

$$101010 / 000110 = 000111$$

$$\begin{array}{r}
 111 = 7_{10} \\
 000110 \overline{) 101010} = 42_{10} \\
 \underline{-110} = 6_{10} \\
 1001 \\
 \underline{-110} \\
 110 \\
 \underline{-110} \\
 0
 \end{array}$$

Subtraction by 1's Complement

In subtraction by 1's complement we subtract two binary numbers using carried by 1's complement.

$$\begin{array}{r} \underline{\hspace{1.5cm}} \quad 1 \\ 0100.101 \end{array}$$

Hence the required difference is **100.101**

(iv) 10110.01 – 11010.10

Solution:

1's complement of 11010.10 is 00101.01

$$\begin{array}{r} 10110.01 \\ \underline{00101.01} \\ 11011.10 \end{array}$$

Hence the required difference is – **00100.01** i.e. – **100.01**

Subtraction by 2's Complement

With the help of subtraction by 2's complement method we can easily subtract two binary numbers.

The operation is carried out by means of the following steps:

- (i) At first, 2's complement of the subtrahend is found.
- (ii) Then it is added to the minuend.
- (iii) If the final carry over of the sum is 1, it is dropped and the result is positive.
- (iv) If there is no carry over, the two's complement of the sum will be the result and it is negative.

The following examples on subtraction by 2's complement will make the procedure clear:

Evaluate:

(i) 110110 - 10110

Solution:

The numbers of bits in the subtrahend is 5 while that of minuend is 6. We make the number of bits in the subtrahend equal to that of minuend by taking a '0' in the sixth place of the subtrahend.

Now, 2's complement of 010110 is (101101 + 1) i.e.101010. Adding this with the minuend.

$$\begin{array}{r} 1 \quad 10110 \quad \text{Minuend} \\ \underline{1 \quad 01010} \quad \text{2's complement of subtrahend} \end{array}$$

Carry over 1 1 0 0 0 0 Result of addition

After dropping the carry over we get the result of subtraction to be 100000.

(ii) 10110 – 11010

Solution:

2's complement of 11010 is (00101 + 1) i.e. 00110. Hence

$$\begin{array}{r}
 \text{Minued -} \quad 10110 \\
 \text{2's complement of subtrahend -} \quad \underline{00110} \\
 \text{Result of addition -} \quad 11100
 \end{array}$$

As there is no carry over, the result of subtraction is negative and is obtained by writing the 2's complement of 11100 i.e.(00011 + 1) or 00100.

Hence the difference is – 100.

(iii) 1010.11 – 1001.01

Solution:

2's complement of 1001.01 is 0110.11. Hence

$$\begin{array}{r}
 \text{Minued -} \quad 1010.11 \\
 \text{2's complement of subtrahend -} \quad \underline{0110.11} \\
 \text{Carry over} \quad 1 \quad 0001.10
 \end{array}$$

After dropping the carry over we get the result of subtraction as 1.10.

(iv) 10100.01 – 11011.10

Solution:

2's complement of 11011.10 is 00100.10. Hence

$$\begin{array}{r}
 \text{Minued -} \quad 10100.01 \\
 \text{2's complement of subtrahend -} \quad \underline{01100.10} \\
 \text{Result of addition -} \quad 11000.11
 \end{array}$$

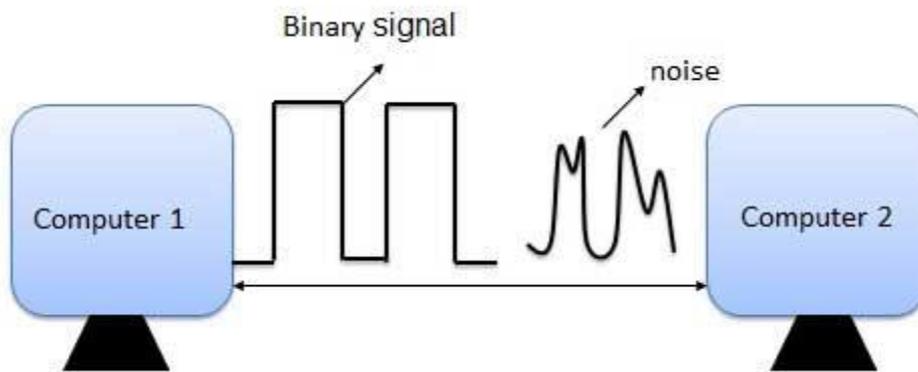
As there is no carry over the result of subtraction is negative and is obtained by writing the 2's complement of 11000.11.

Hence the required result is – 00111.01.

Error Detection & Correction

What is Error?

Error is a condition when the output information does not match with the input information. During transmission, digital signals suffer from noise that can introduce errors in the binary bits travelling from one system to other. That means a 0 bit may change to 1 or a 1 bit may change to 0.



Error-Detecting codes

Whenever a message is transmitted, it may get scrambled by noise or data may get corrupted. To avoid this, we use error-detecting codes which are additional data added to a given digital message to help us detect if an error occurred during transmission of the message. A simple example of error-detecting code is **parity check**.

Error-Correcting codes

Along with error-detecting code, we can also pass some data to figure out the original message from the corrupt message that we received. This type of code is called an error-correcting code. Error-correcting codes also deploy the same strategy as error-detecting codes but additionally, such codes also detect the exact location of the corrupt bit.

In error-correcting codes, parity check has a simple way to detect errors along with a sophisticated mechanism to determine the corrupt bit location. Once the corrupt bit is located, its value is reverted (from 0 to 1 or 1 to 0) to get the original message.

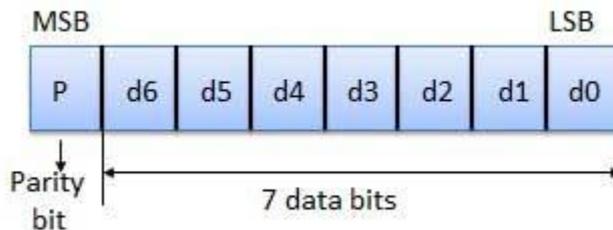
How to Detect and Correct Errors?

To detect and correct the errors, additional bits are added to the data bits at the time of transmission.

- The additional bits are called **parity bits**. They allow detection or correction of the errors.
- The data bits along with the parity bits form a **code word**.

Parity Checking of Error Detection

It is the simplest technique for detecting and correcting errors. The MSB of an 8-bits word is used as the parity bit and the remaining 7 bits are used as data or message bits. The parity of 8-bits transmitted word can be either even parity or odd parity.



Even parity -- Even parity means the number of 1's in the given word including the parity bit should be even (2,4,6,...).

Odd parity -- Odd parity means the number of 1's in the given word including the parity bit should be odd (1,3,5,...).

Use of Parity Bit

The parity bit can be set to 0 and 1 depending on the type of the parity required.

- For even parity, this bit is set to 1 or 0 such that the no. of "1 bits" in the entire word is even. Shown in fig. (a).
- For odd parity, this bit is set to 1 or 0 such that the no. of "1 bits" in the entire word is odd. Shown in fig. (b).

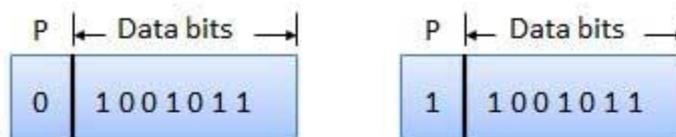


Fig. (a)

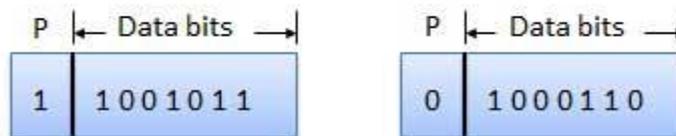


Fig. (b)

How Does Error Detection Take Place?

Parity checking at the receiver can detect the presence of an error if the parity of the receiver signal is different from the expected parity. That means, if it is known that the parity of the transmitted signal is always going to be "even" and if the received signal has an odd parity, then the receiver can conclude that the received signal is not correct. If an error is detected,

then the receiver will ignore the received byte and request for retransmission of the same byte to the transmitter.

