UNIT – 1
BASIC CONCEPTS OF COMPUTERS

FYBTECH

Subject : **Programming and Problem Solving**

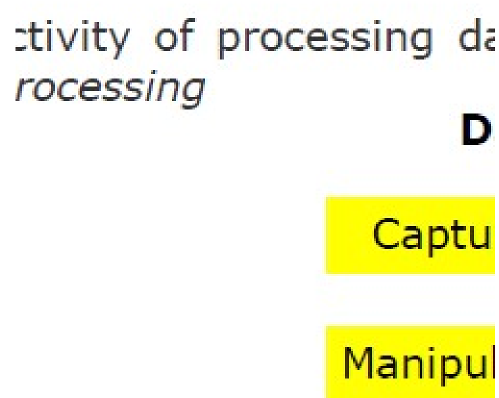
Unit 1 Contents

- **Basic Concepts of computers:** Architecture of computer, Types of Processor, Primary Storage, Secondary Storage, Number Systems, Data Representation-Signed, Unsigned, one’s complement, two’s complement, Floating point, char, String. Introduction to system software: Operating system, Editor, Compiler, Assembler,
  Linker, Loader.

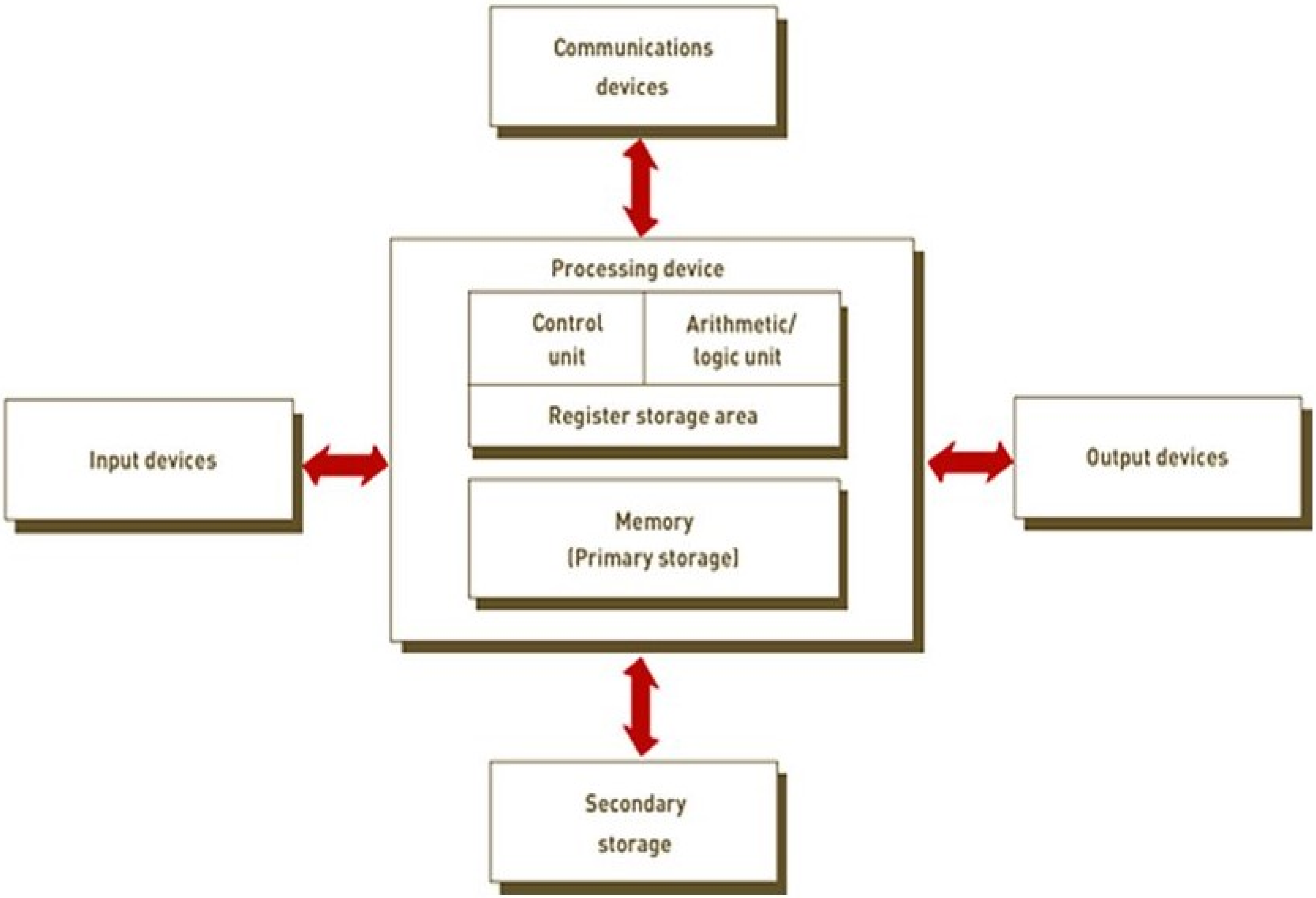
Introduction to Computer

Computer:

- The word computer comes from the word "compute", which means, "to calculate"
- Computer is an electronic device that can perform arithmetic and logical operations at high speed
- A computer is also called a data processor because it can store, process, and retrieve data whenever desired
- Data is raw material used as input and information is processed data obtained as output of data processing



Architecture of Computer system



Generation of Computers

- "Generation" in computer talk is a step in technology
- It provides a framework for the growth of computer industry
- Originally it was used to distinguish between various hardware technologies, but now it has been extended to include both hardware and software
- Till today, there are five computer generations

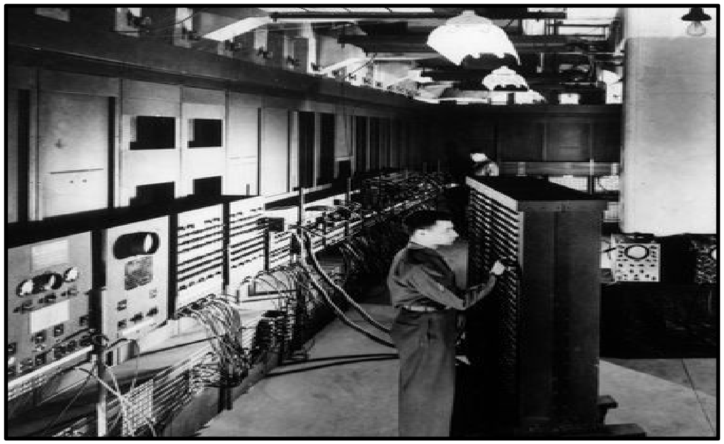
Generation of Computers

(Continued from previous slide..)

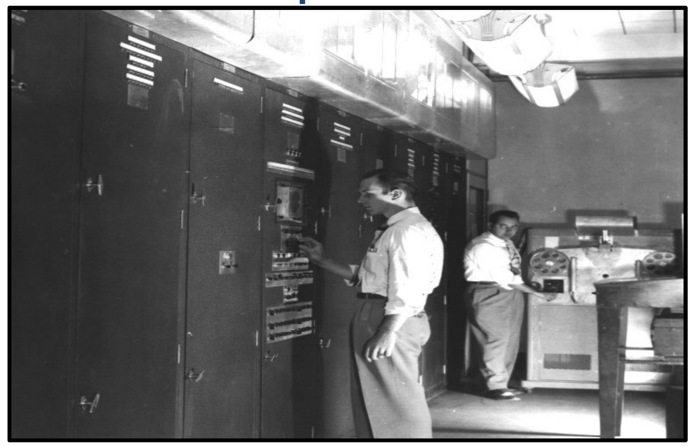
Generation (Period)	Key hardware technologies	Key software technologies	Key characteristi
First (1942-1955)	▪ Vacuum tubes ▪ Electromagnetic relay memory ▪ Punched cards secondary storage	▪ Machine and assembly languages ▪ Stored program concept ▪ Mostly scientific applications	▪ Bulky in size ▪ Highly unreliable ▪ Limited comr use and costly ▪ Difficult comr production ▪ Difficult to use
Second (1955-1964)	▪ Transistors ▪ Magnetic cores memory	▪ Batch operating system ▪ High-level	▪ Faster, smaller, reliable and eas program than pr

Some First Generation Computers

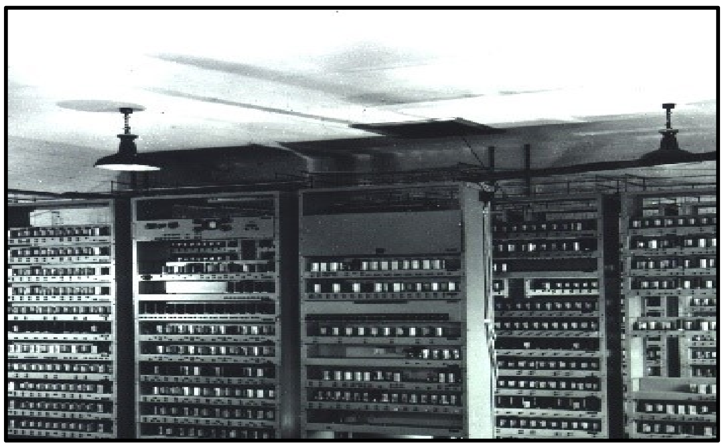
ENIAC: Electronic Numerical Integrator and Calculator



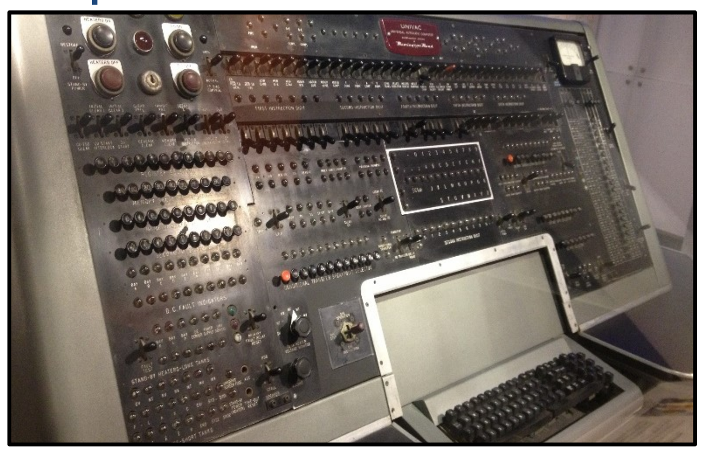
EDVAC: Electronic Discrete Variable Automatic Computer



EDSAC: Electronic Delay Storage Automatic Calculator



UNIVAC: Universal Automatic Computer



Generation of Computers

(Continued from previous slide..)

Generation (Period)	Key hardware technologies	Key software technologies	Key characteristic
Third (1964-1975)	▪ ICs with SSI and MSI technologies ▪ Larger magnetic cores memory ▪ Larger capacity disks and magnetic tapes secondary storage	▪ Timesharing operating system ▪ Standardization of high-level programming languages ▪ Unbundling of software from	▪ Faster, smaller, r reliable, easier ar cheaper to produ ▪ Commercially, ea to use, and easie upgrade than previous generati systems ▪ Scientific, comme

Note:

- SSI: Small Scale Integration
- MSI: Medium Scale Integration
- PDP: Personal Data Processors
- CDC: Control Data Corporation

Generation of Computers

(Continued from previous slide..)

Generation (Period)	Key hardware Technologies	Key software technologies	Key character
Fourth (1975-1989)	▪ ICs with VLSI technology ▪ Microprocessors; semiconductor memory ▪ Larger capacity hard disks as in-built secondary storage ▪ Magnetic tapes and floppy disks as portable storage media ▪ Personal computers ▪ Supercomputers based on parallel vector processing and	▪ Operating systems for PCs with GUI and multiple windows on a single terminal screen ▪ Multiprocessing OS with concurrent programming languages ▪ UNIX operating system with C programming language ▪ Object-oriented design and programming ▪ PC Network-based	▪ Small, aff reliable, a to use PCs ▪ More and mainframe systems supercomp ▪ Totally purpose m ▪ Easier to commercia ▪ Easier to u Rapid

Note:

- VLSI: Very Large Scale Integration
- VAX: *Virtual Address Extension*

Generation of Computers

(Continued from previous slide..)

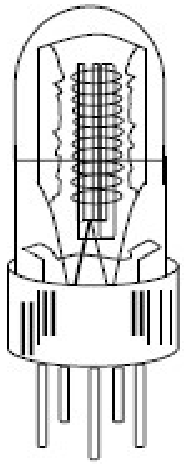
Generation (Period)	Key hardware technologies	Key software technologies	Key characteristics
Fifth (1989-Present)	▪ ICs with ULSI technology ▪ Larger capacity main memory, hard disks with RAID support ▪ Optical disks as portable read-only storage media ▪ Notebooks, powerful desktop PCs and workstations ▪ Powerful servers	▪ Micro-kernel based, multithreading, distributed OS ▪ Parallel programming libraries like MPI & PVM ▪ JAVA ▪ World Wide Web ▪ Multimedia, Internet applications ▪ More complex	▪ Portable compute ▪ Powerful, cheape reliable, and easi to use deskt machines ▪ Powerful supercomputers ▪ High uptime due hot-pluggable components ▪ Totally gene purpose machines ▪ Easier to produ

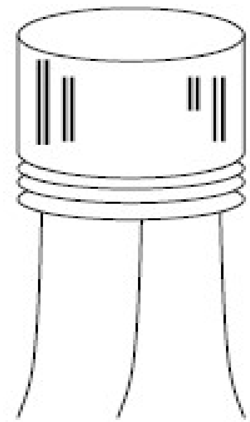
Note:

- ULSI: Ultra Large Scale Integration
- PVM: Parallel Virtual Machine
- SGI: Silicon Graphics Inc.
- PARAM: Parallel Machine

Generation of Computers

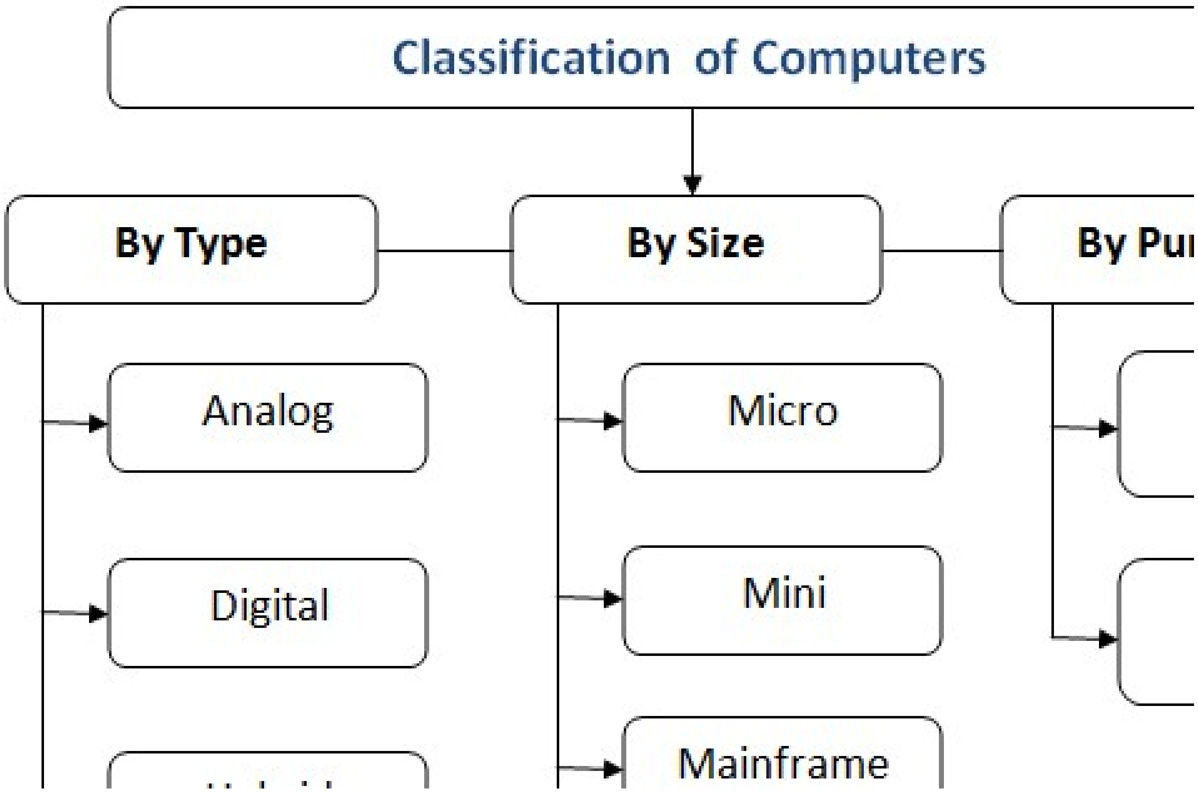
- Electronic Devices Used in Computers of Different Generations





Classification of Computer

- Computers can be classified as follows:



Classification of Computer

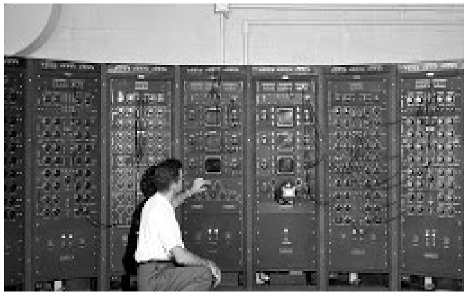
1. According to Type

i. Digital Computer

A computer that performs calculations and logical operations with quantities represented as digits, usually in the binary number system

ii. Analog Computer

An analog computer is a form of computer that uses continuous physical phenomena such as electrical, mechanical, or hydraulic quantities to model the problem being solved



iii. Hybrid Computer

Exhibit features of analog and digital computers

Classification of Computer

2. According to Size (capabilities)

i. Mini computer

A midsized computer. In size and power, minicomputers lie between workstations and mainframes.

ii. Micro computer

- **Desktop Computer:** A personal or micro-mini computer sufficient to fit on a desk
- **Laptop Computer:** A portable computer complete with an integrated screen and keyboard. It is generally smaller in size than a desktop computer and larger than a notebook computer
- **Palmtop Computer/Digital Diary /Notebook /PDAs:** A hand-sized computer. Palmtops have no keyboard but the screen serves both as an input and output device

iii. Mainframes

A very large and expensive computers capable of supporting hundreds, or even thousands, of users simultaneously

Classification of Computer

2. According to Size (capabilities)

iv. Super Computer

- The fastest and most powerful type of computer
- Supercomputers are very expensive and are employed for specialized applications that require immense amounts of mathematical calculations
- For example, weather forecasting requires a supercomputer

v. Workstations

- A terminal or desktop computer in a network
- In this context, workstation is just a generic term for a user's machine (client machine) in contrast to a "server" or "mainframe"

Classification of Computer

3. According to Purpose

i. General Purpose

General purpose computers are designed to perform a range of tasks

e.g. Personal Computer

ii. Special Purpose

Specific purpose computers are designed to handle a specific problem or to perform a specific task

e.g. Computers that control aircraft and satellites

Types of Processor

- A processor, or "microprocessor," is a small chip that resides in computers and other electronic devices
- A processor performs arithmetical, logical, input/output (I/O) and other basic instructions that are passed from an operating system (OS)
- Types of processor depends on the architecture of processor

Type of Architecture	Features	
CISC (Complex Instruction Set Computer)	▪ Large instruction set ▪ Variable-length instructions ▪ Variety of addressing modes ▪ Complex & expensive to produce	M p c
[illegible]	▪ Small instruction set	

Types of Processor

(Continued from previous slide..)

Type of Architecture	Features	
EPIC (Explicitly Parallel Instruction Computing)	▪ Allows software to communicate explicitly to the processor when operations are parallel ▪ Uses tighter coupling between the compiler and the processor	M hi ar

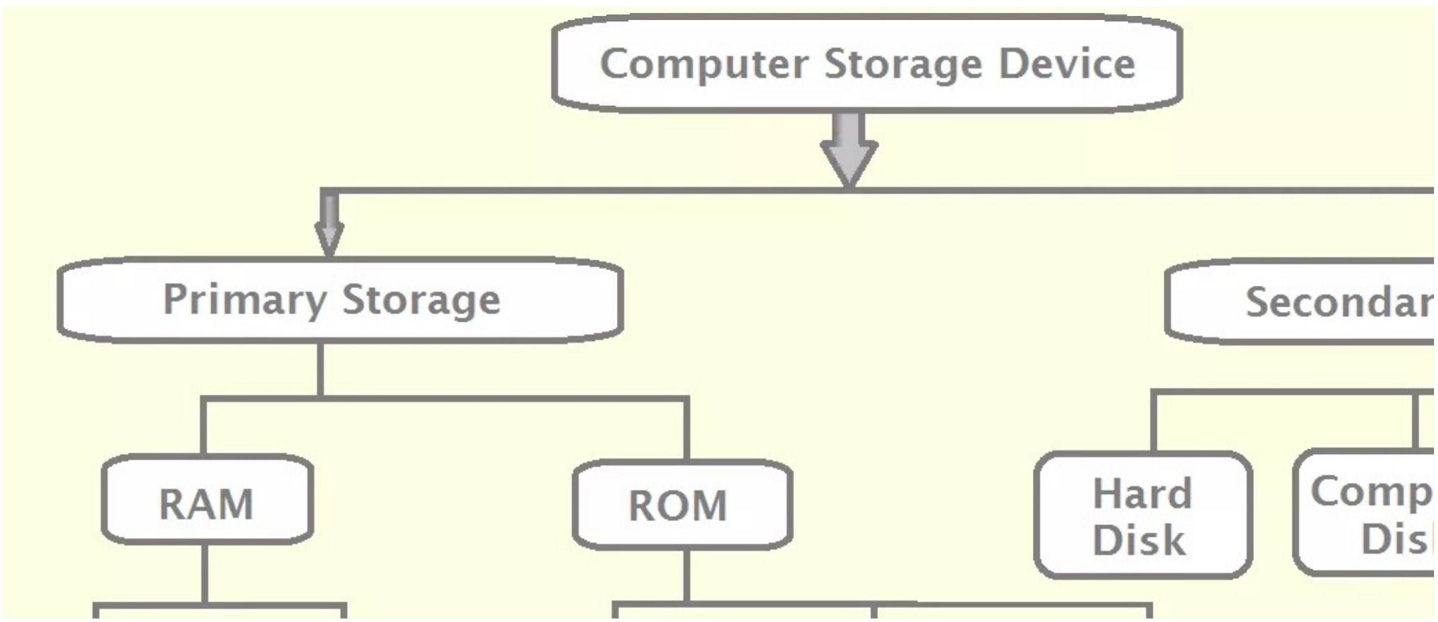
Types of Processor

(Continued from previous slide..)

Type of Architecture	Features	
Multi-Core Processor	▪ Processor chip has multiple cooler-running, more energy-efficient processing cores ▪ Improve overall performance by handling more work in parallel ▪ can share architectural	hi ar

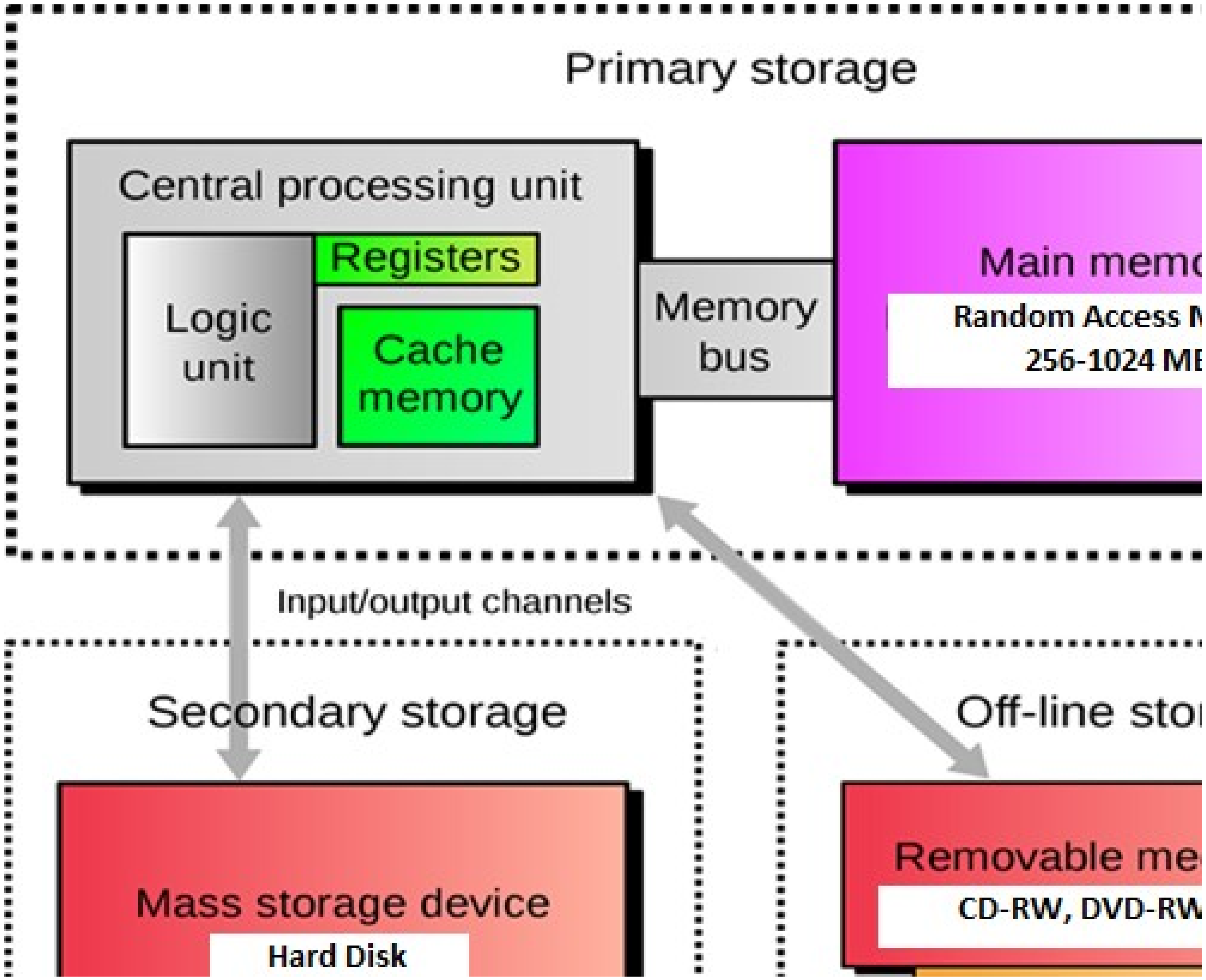
Computer Storage

- A storage device is any **computing hardware** that is used for storing and extracting data
- It can hold and store information both temporarily and permanently, and can be internal or external to a computer, server or any similar computing device
- There are two main types of storage devices: Primary and Secondary



Types of Computer Storage

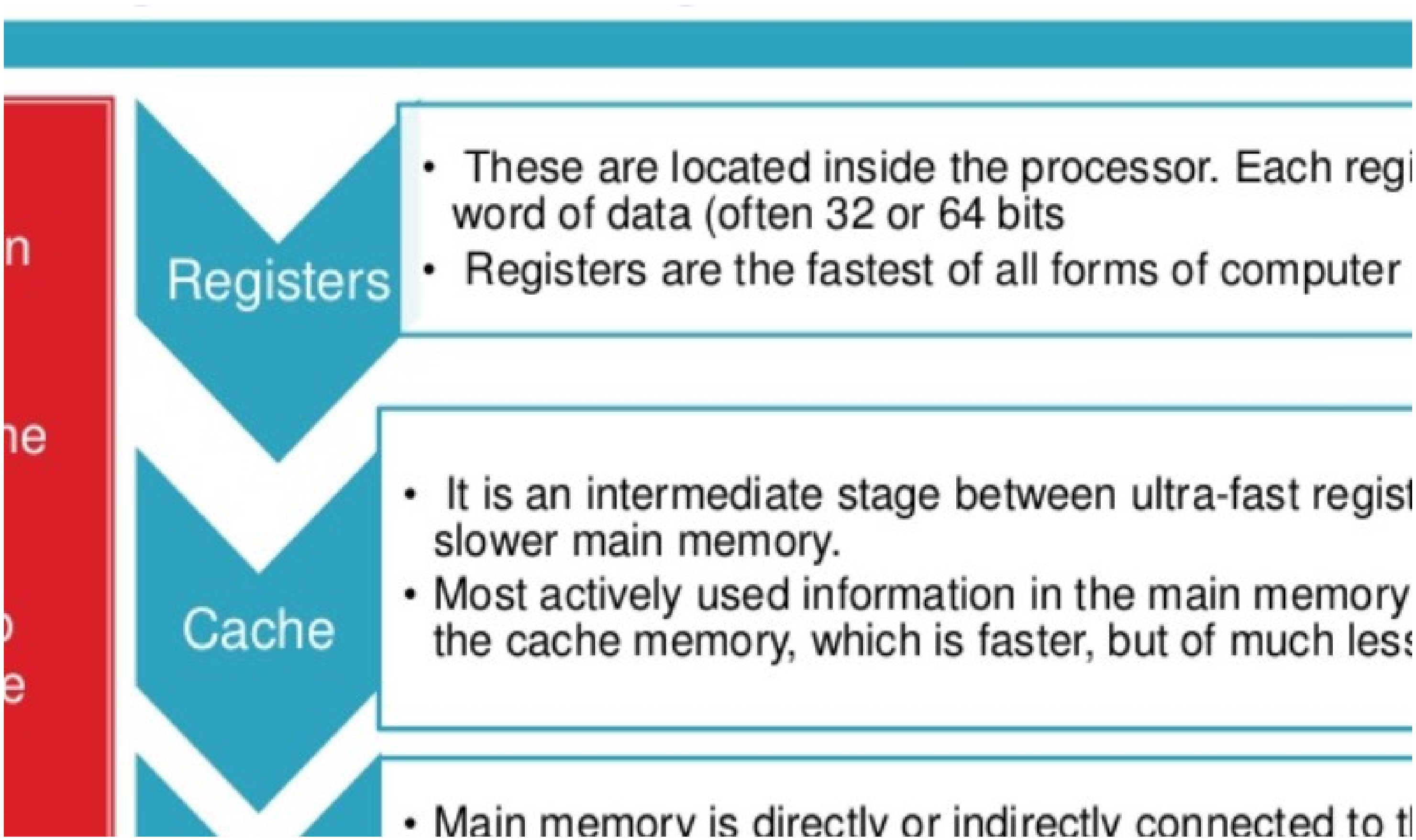
Computer Storage



Primary Storage and Secondary Storage

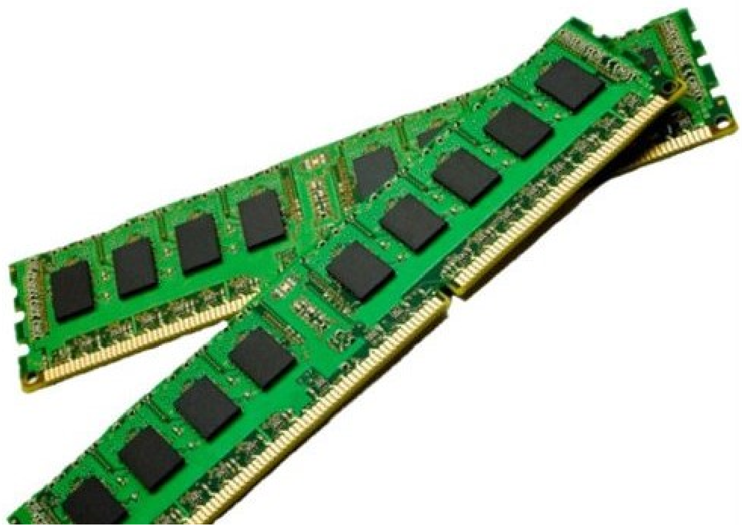
Primary Storage

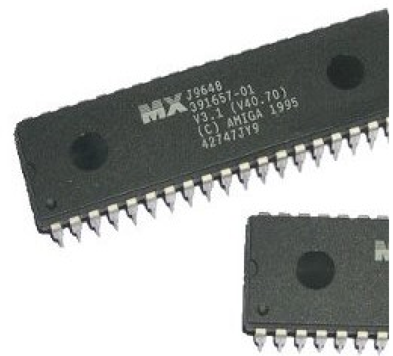
- **Primary storage** (also known as main **storage** or **memory**) is the area in a computer in which data is stored for quick access by the computer's processor. It consists of Registers, Cache and Main Memory



Primary Storage

- Main Memory consists of **Random Access Memory (RAM)** and **Read Only Memory (ROM)**





Random Access Memory (RAM)

- It is a read/write (R/W) memory which is volatile
- This means when power is turned off, all the contents are destroyed
- It can be accessed randomly: that is, any byte of memory can be accessed without touching the preceding bytes
- RAM is the most common type of memory found in computers and other devices such as printers
- There are two basic types of RAM:
  - Dynamic RAM (DRAM)
  - Static RAM(SRAM)

Types of RAM

- **DRAM:**
  - DRAM needs to be **refreshed thousands of times per second**
  - DRAM stores a bit of data using a transistor and capacitor pair, which together comprise a memory cell
  - The capacitor holds a high or low charge (1 or 0, respectively), and the transistor acts as a switch that lets the control circuitry on the chip read the capacitor's state of charge or change it
  - As this form of **memory is less expensive** to produce than SRAM, it is the predominant form of computer memory used in modern computers.

- **SRAM:**
  - SRAM does not need to be refreshed, which makes it faster, but it is more **expensive than DRAM**
  - In SRAM, a bit of data is stored using the state of a flip-flop
  - This form of RAM is more expensive to produce, but is generally faster and requires less power than DRAM and, in modern computers, is often used as cache memory for the CPU

Read Only Memory (ROM)

- ROM is **non-volatile** which means it retains the stored information even if power is turned off
- It is used to **store programs that boot the computer** and perform diagnostics
- Different types of ROM as follows:
  - Programmable ROM (PROM)
  - Erasable Programmable ROM (EPROM)
  - Electrically Erasable Programmable ROM (EEPROM)

Types of ROM

- **PROM** (Programmable ROM):

  - A PROM is a memory chip on which data can be **written onto only once**. Once a program is written onto a PROM chip, it remains there forever
  - Unlike RAM, PROM retains its contents when the computer is turned off
  - The difference between a PROM and a ROM is that a PROM is manufactured as blank memory and programmed later with a special device called PROM programmer or the PROM burner, whereas the ROM is programmed during manufacturing process.
  - The process of programming a PROM is sometimes called burning a PROM

- **EPROM** (Erasable Programmable ROM) :

  - An EPROM is a special type of PROM that can be erased by exposing it to ultraviolet light
  - Once erased, it can be reprogrammed. An EPROM is similar to a PROM except that it requires ultraviolet radiation to be erased

Types of ROM

- **EEPROM** (Electrically Erasable Programmable ROM):
  - EEPROM **is a special type of PROM** that can be erased by exposing it to an electrical charge
  - Like other types of PROM, EEPROM retains its contents even when the power is turned off
  - Also, like other types of ROM, EEPROM is not as fast as RAM
  - EEPROM is similar to Flash Memory (sometimes called flash EEPROM)
  - The principal difference is that EEPROM requires **data to be written or erased one byte at a time** whereas flash memory allows data to be written or erased in blocks

Secondary Storage

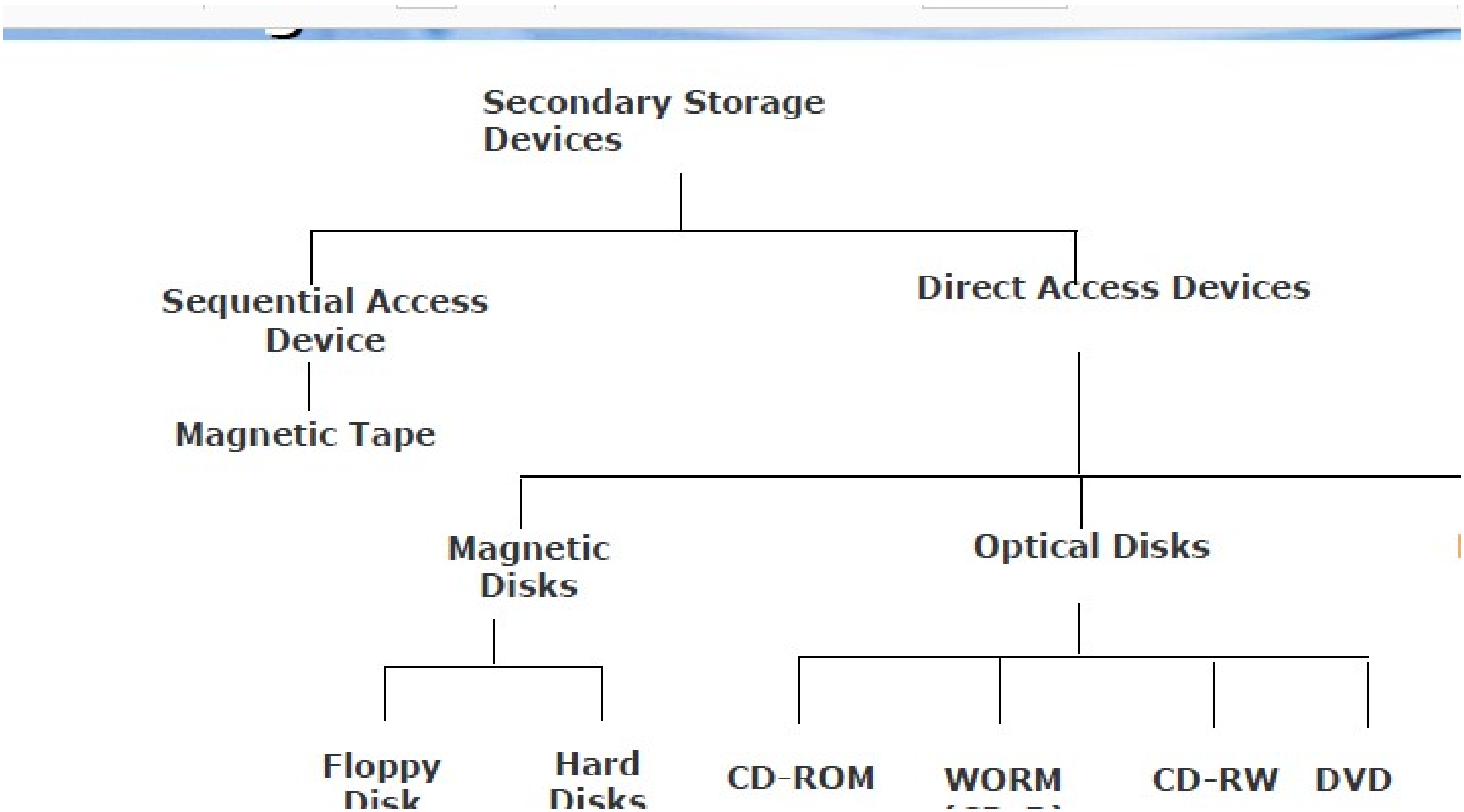
- Secondary Storage is alternatively referred to as external memory, secondary memory or auxiliary storage
- Secondary storage device is a **non-volatile device** that holds data until it is deleted or overwritten

- **Primary storage**
  - Volatile
  - Temporary
- It loses all of its contents when power to the system unit is shut off

- **Secondary storage**
  - Nonvolatile
  - Permanent
  - **Writing** : is the process of saving information
  - **Reading:** is the process of accessing information

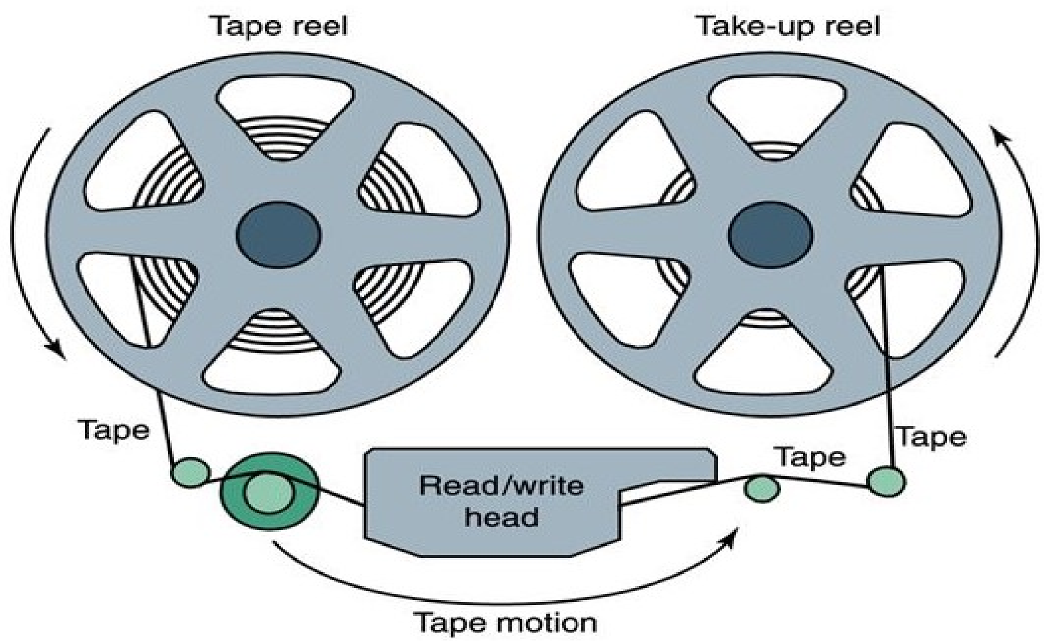
Secondary Storage

- Secondary Storage devices are classified as follows:



Secondary Storage

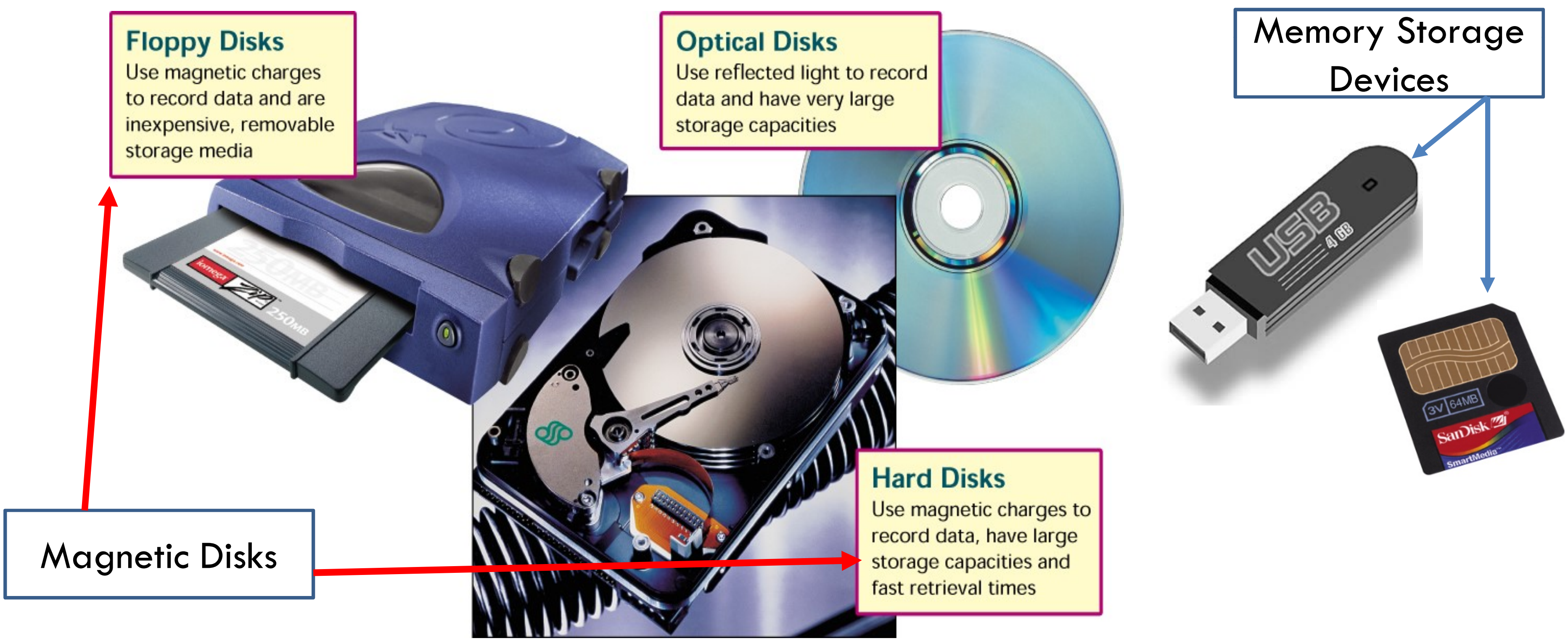
- Sequential-access Storage Devices
  - Arrival at the desired storage location **may be preceded by sequencing through other locations**
  - Data can only be **retrieved in the same sequence** in which it is stored
  - Magnetic tape is a typical example of such a storage device



A magnetic tape storage mechanism

Secondary Storage

- Direct-access Storage Devices
  - Devices where any storage location may be selected and **accessed at random**
  - Permits access to individual information in a more direct or immediate manner
  - **Magnetic and optical disks** are typical examples of such a storage device



Number System

- A **number system** is a collection of various symbols which are called digits

- **Two types of number systems are:**
  - Non-positional Number Systems
  - Positional Number Systems

Non-positional Number System

- **Characteristics**
  - Use symbols such as I for 1, II for 2, III for 3 etc
  - Each symbol represents the same value regardless of its position in the number
  - The symbols are simply added to find out the value of a particular number

- **Difficulty**
  - It is difficult to perform arithmetic with such a number system

Positional Number System

- **Characteristics**
  - Use only a few symbols called digits
  - These symbols represent different values depending on the position they occupy in the number
  - The value of each digit is determined by:
    1. The digit itself
    2. The position of the digit in the number
    3. **The base of the number system** (Base = total number of digits in the number system)
  - The maximum value of a single digit is always equal to one less than the value of the base

Positional Number System

- There are four Positional Number Systems: Binary, Decimal, Octal and Hexadecimal

Binary	Decimal	Octal
0000	00	0
0001	01	1
0010	02	2
0011	03	3
0100	04	4
0101	05	5
0110	06	6
0111	07	7
1000	08	10
1001	09	11
1010	10	12
1011	11	13
1100	12	14
1101	13	15

Binary Number System

- **Characteristics:**

  - Has only 2 symbols or digits (0 and 1). Hence its base = 2
  - The maximum value of a single digit is 1 (one less than the value of the base)
  - **Each position of a digit represents a specific power of the base (2)**
  - This number system is used in computers

(Continued from previous slide..)

Example

$10101_2 = (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) +$

Decimal Number System

- **Characteristics:**
  - Has 10 symbols or digits (0, 1, 2, 3, 4, 5, 6, 7,8, 9). Hence, its base = 10
  - The maximum value of a single digit is 9 (one less than the value of the base)
  - Each position of a digit represents a specific power of the base (10)
  - We use this number system in our day-to-day life

(Continued from previous slide..)

Example

$2586_{10} = (2 \times 10^3) + (5 \times 10^2) + (8 \times$

Octal Number System

- **Characteristics:**
  - Has total 8 symbols or digits (0, 1, 2, 3, 4, 5, 6, 7). Hence, its base = 8
  - The maximum value of a single digit is 7 (one less than the value of the base)
  - Each position of a digit represents a specific power of the base (8)
  - Since there are only 8 digits, 3 bits ($2^3 = 8$) are sufficient to represent any octal number in binary

- Since there are only 8 digits, 3 b
sufficient to represent any octal numb

Example

$2057 = (2 \times 8^3) + (0 \times 8^2) + (5$

Hexadecimal Number System

- **Characteristics:**
  - Has total 16 symbols or digits (0, 1, 2, 3, 4, 5, 6, 7,8, 9, A, B, C, D, E, F). Hence its base = 16
  - The symbols A, B, C, D, E and F represent the decimal values 10, 11, 12, 13, 14 and 15 respectively
  - The maximum value of a single digit is 15 (one less than the value of the base)
  - Each position of a digit represents a specific power of the base (16)
  - Since there are only 16 digits, 4 bits ($2^4 = 16$) are sufficient to represent any

sufficient to represent any he
binary

Example

$1AF_{16} = (1 \times 16^2) + (A \times 16^1)$

Conversions

- **Decimal to Binary**
- Here is an example of using repeated division to convert 1792 decimal to binary:

Here is an example of using repeated division tc

Decimal Number	Operation	Quotie
1792	÷ 2 =	8
896	÷ 2 =	4
448	÷ 2 =	2
224	÷ 2 =	1
112	÷ 2 =	
56	÷ 2 =	
28	÷ 2 =	
14	÷ 2 =	

- Reverse the remainders, we get 11100000000
- $(1792)_{10} = (11100000000)_2$

Conversions

- **Decimal to Octal**
- Here is an example of using repeated division to convert 1792 decimal to octal:

Decimal to Octal

Here is an example of using repeated division tc

Decimal Number	Operation	Quotie
1792	÷ 8 =	2:

- Reverse the remainders, we get 3400
- $(1792)_{10} = (3400)_8$

Conversions

- **Decimal to Hexadecimal**
- Here is an example of using repeated division to convert 1792 decimal to hexadecimal:

Here is an example of using repeated division tc

Decimal Number	Operation	Quotie
1792	÷ 16 =	1
112	÷ 16 =	

- Reverse the remainders, we get 700
- $(1792)_{10} = (700)_{16}$

Conversions

- The only addition to the algorithm when converting from decimal to hexadecimal is that a table must be used to obtain the hexadecimal digit if the remainder is greater than decimal 9.

be used to obtain the hexadecimal digit if the remainder is g

Decimal:	0	1	2	3	4	5
Hexadecimal:	0	1	2	3	4	5

- For example, 590 decimal converted to hex is:

Decimal Number	Operation	Quotient	Remainder	Hexadecimal Result
590	÷ 16 =	36	14	E
36	÷ 16 =	2	4	4
2	÷ 16 =	0	2	2
0	done.			

- Reverse the remainders, we get 24E
- $(590)_{10} = (24E)_{16}$

Conversion from other to decimal number system

Binary Number System

(Continued from previous slide..)

Example

- $10101_2 = (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) +$

Since there are only 8 digits, 3 b

sufficient to represent any octal numb

Example

$2057_8 = (2 \times 8^3) + (0 \times 8^2) + (5$

sufficient to represent any he

binary

Example

$1AF_{16} = (1 \times 16^2) + (A \times 16^1)$

Conversion from Binary to octal

- **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
1.	10101_2	010 101
2.	10101_2	2_8 5_8
3.	10101_2	25_8

Binary Number – 10101_2 = Octal Number – 25_8

Conversion from octal to Binary

- **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

Binary Number − 10101_2

Calculating Octal Equivalent −

Step	Octal Number	Binary Number
1.	25_8	$2_{10}\ 5_{10}$
2.	25_8	$010_2\ 101_2$
3.	25_8	010101_2

Octal Number − 25_8 = Binary Number − 10101_2

Conversion from binary to Hexadecimal

- **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 – 10101_2

Calculating Octal Equivalent –

Step	Binary Number	Hexadecimal Number
1.	10101_2	0001 0101
2.	10101_2	$1_{10}\ 5_{10}$
3.	10101_2	15_{16}

Binary Number – 10101_2 = Hexadecimal Number – 15_{16}

Conversion from Hexadecimal to binary

- **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

Step	Hexadecimal Number	Binary Number
1.	15_{16}	51_{1010}
2.	15_{16}	$0001_2\ 0101_2$
3.	15_{16}	00010101_2

Hexadecimal Number – 15_{16} = Binary Number – 10101_2

Octal to Hexadecimal

- When converting from octal to hexadecimal, it is often easier to first convert the octal number into binary and then from binary into hexadecimal.
- For example, to convert 345 octal into hex:*(from the previous example)*
- Octal =345

  Binary =011 100 101

Drop any leading zeros or pad with leading zeros to get groups of four binary digits (bits):
Binary 011100101 = 1110 0101

Then, look up the groups in a table to convert to hexadecimal digits.

- Binary =1110 0101 Hexadecimal =E5= E5 hex

Binary	Decimal	Octal
0000	00	0
0001	01	1
0010	02	2
0011	03	3
0100	04	4
0101	05	5
0110	06	6
0111	07	7
1000	08	10
1001	09	11
1010	10	12
1011	11	13
1100	12	14
1101	13	15

Hexadecimal to Octal

- When converting from hexadecimal to octal, it is often easier to first convert the hexadecimal number into binary and then from binary into octal.
- For example, to convert A2DE hex into octal:
- Hexadecimal =**A** **2** **D** **E**
- **Binary** =1010 0010 1101 1110 = 1010001011011110
- **Add leading zeros or remove leading zeros to group into sets of three binary digits.**
- Binary: 1010001011011110 = 001 010 001 011 011 110
- Then, look up each group in a table:

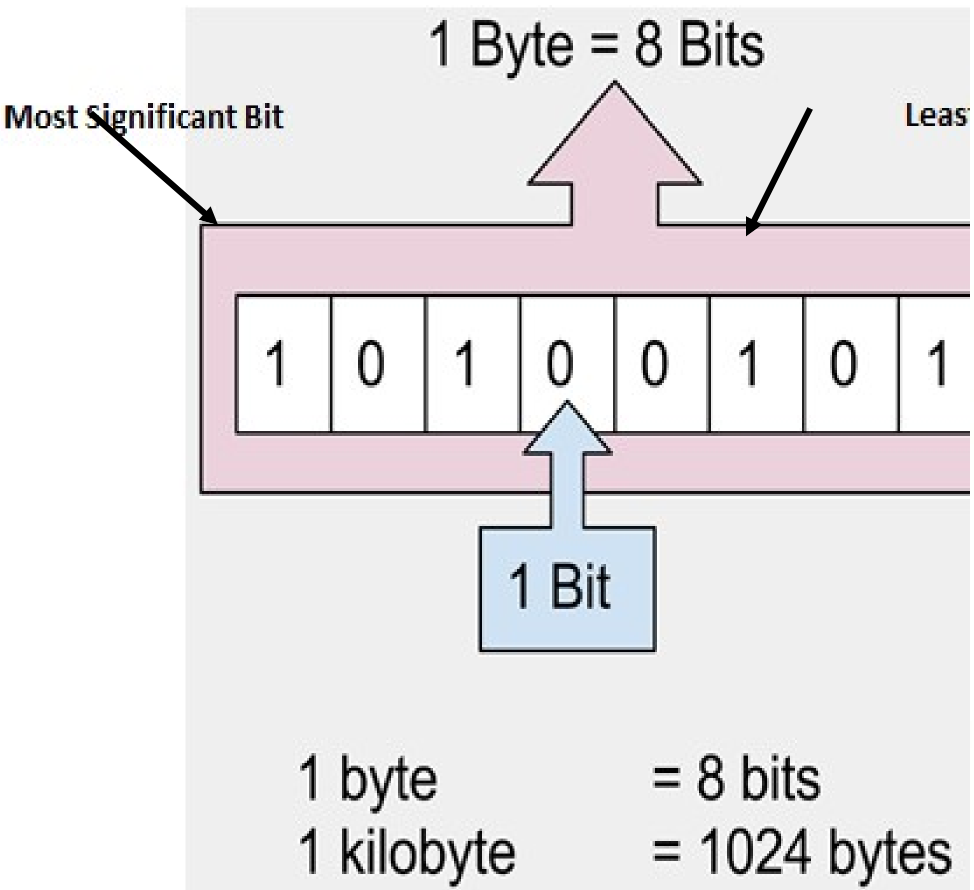
Binary	000	001	010	011	100	101	110	111
Octal	0	1	2	3	4	5	6	7

Binary =001010001011011110 Octal =121336

Data Representation

- Computer uses a fixed number of bits to represent a piece of data, which could be a number, a character, or others
- A n-bit storage location can represent up to 2^n distinct entities
- For example, a 3-bit memory location can hold one of these eight binary patterns: 000, 001, 010, 011, 100, 101, 110, or 111
- Hence, it can represent at most 8 distinct entities. You could use them to represent:
  - Numbers 0 to 7
  - Characters 'A' to 'H'
  - 8 kinds of fruits like apple, orange, banana
  - or 8 kinds of animals like lion, tiger, etc.

Data Representation



Multiples of Bytes		
Unit (Symbol)	**Value (SI)**	**Value (Binary)**
Kilobyte (kB)	10^3	2^{10}
Megabyte (MB)	10^6	2^{20}
Gigabyte (GB)	10^9	2^{30}
Terabyte (TB)	10^{12}	2^{40}
Petabyte (PB)	10^{15}	2^{50}
Exabyte (EB)	10^{18}	2^{60}
Zettabyte (ZB)	10^{21}	2^{70}
Yottabyte (YB)	10^{24}	2^{80}

Memory Units

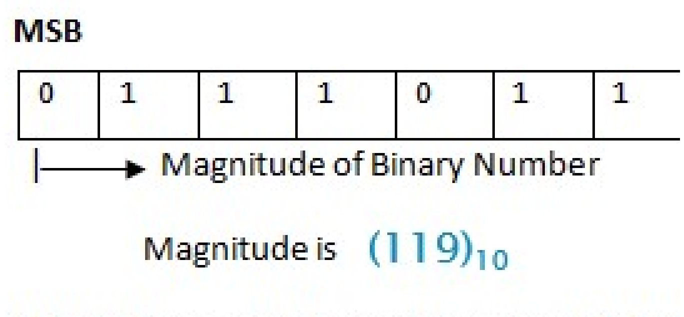
Data Representation: Signed and Unsigned

- Integers, for example, can be represented in 8-bit, 16-bit, 32-bit or 64-bit

- Besides bit-lengths, there are two representation schemes for integers:
  1. ***Unsigned Integers:*** can represent zero and positive integers
  2. ***Signed Integers:*** can represent zero, positive and negative integers

Data Representation: Unsigned

- Unsigned integers can represent zero and positive integers, but not negative integers.
- The value of an unsigned integer is interpreted as "the magnitude of its underlying binary pattern".



Example 1:	**Example 2:**
Suppose n=8 and	Suppose n=16 and
binary pattern is **0100 0001B**	binary pattern is **0001 0000 0000 1000B**
Value of this unsigned integer is:	Value of this unsigned integer is:
1×2^0 + 1×2^4 = 17D	1×2^0+ 1×2^3= 9D

Data Representation: Unsigned

- An n-bit pattern can represent 2^n distinct integers. An n-bit unsigned integer can represent integers from 0 to (2^n)-1, as tabulated below:

n	Minimum	Maximum
8	0	(2^8)-1 (=255)
16	0	(2^16)-1 (=65,535)
32	0	(2^32)-1 (=4,294,967,295)
64	0	(2^64)-1 (=18,446,744,073,709,551,615)

- An 8-bit unsigned integer has a range of 0 to 255

- Maximum value = 255

128+64+32+16+8+4+2+1 = 255

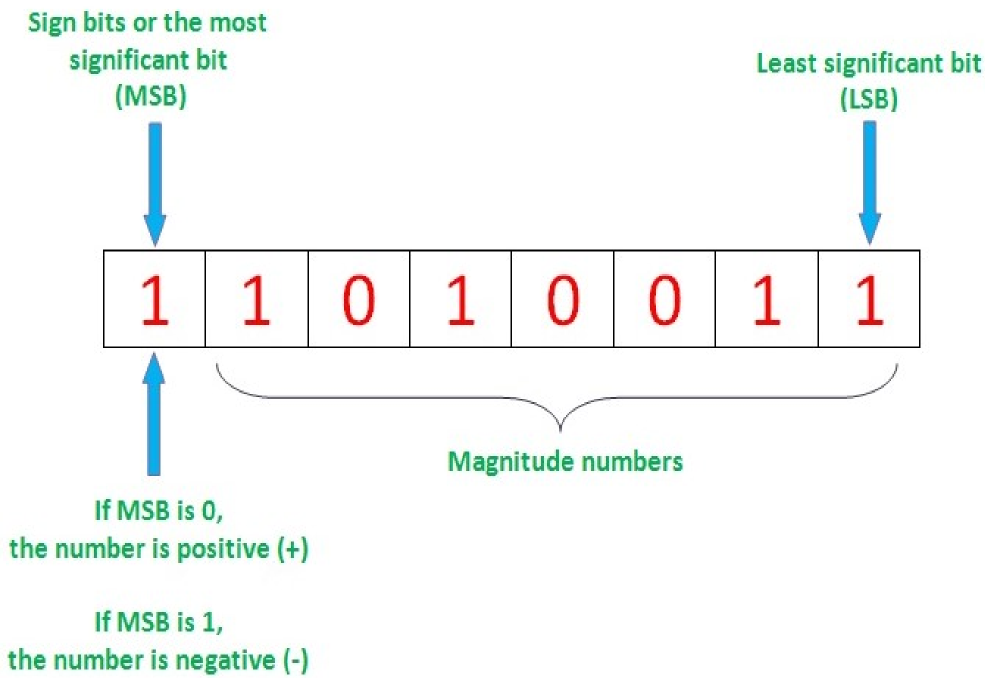
128	64	32	16	8	4	2	1
1	1	1	1	1	1	1	1

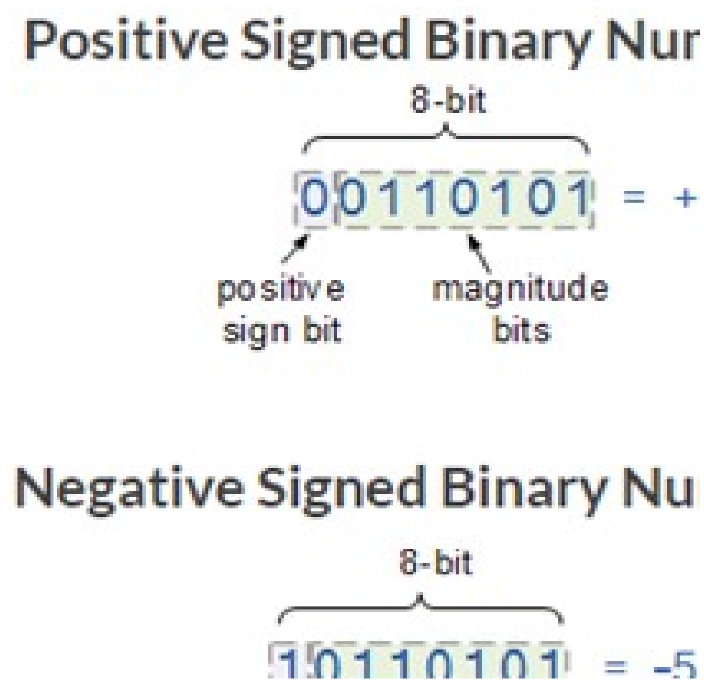
- Minimum value = 0

128	64	32	16	8	4	2	1
0	0	0	0	0	0	0	0

Data Representation: Signed

- The Most-significant Bit (MSB) is the sign bit, with value of 0 representing positive integer and 1 representing negative integer.
- The remaining n-1 bits represents the magnitude (absolute value) of the integer.
- The absolute value of the integer is interpreted as "the magnitude of the (n-1)-bit binary pattern".





Data Representation: Signed

Example 1:

Suppose n=8 and binary representation is **0 100 0001B**

Sign bit is 0 ⇒ positive

Absolute value is 100 0001B = 65D

Hence, the integer is +65D

Example 2:

Suppose n=8 and binary representation is **1 000 0001B**

Sign bit is 1 ⇒ negative

Absolute value is 000 0001B = 1D

Hence, the integer is -1D

Example 3:

Suppose n=8 and binary representation is **0 000 0000B**

Sign bit is 0 ⇒ positive

Absolute value is 000 0000B = 0D

Hence, the integer is +0D

Example 4:

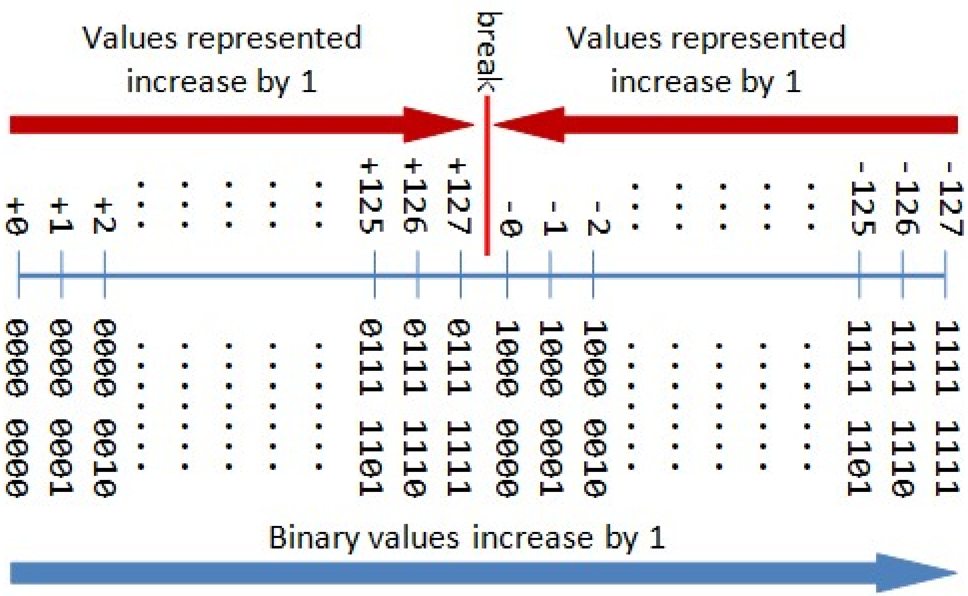
Suppose n=8 and binary representation is **1 000 0000B**

Sign bit is 1 ⇒ negative

Absolute value is 000 0000B = 0D

Hence, the integer is -0D

Data Representation: Signed



Sign-Magnitude Representation

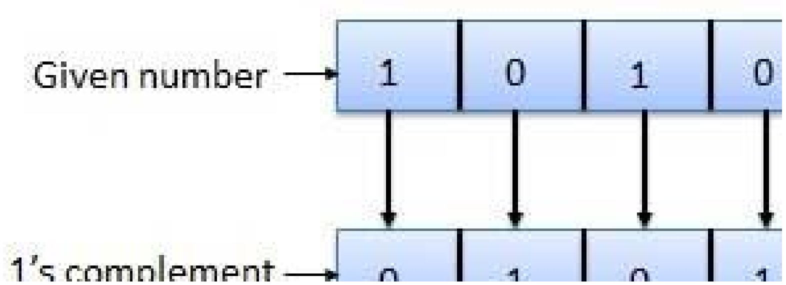
The drawbacks of sign-magnitude representation are:

1. There are two representations (0000 0000B and 1000 0000B) for the number zero, which could lead to inefficiency and confusion.

2. Positive and negative integers need to be processed separately.

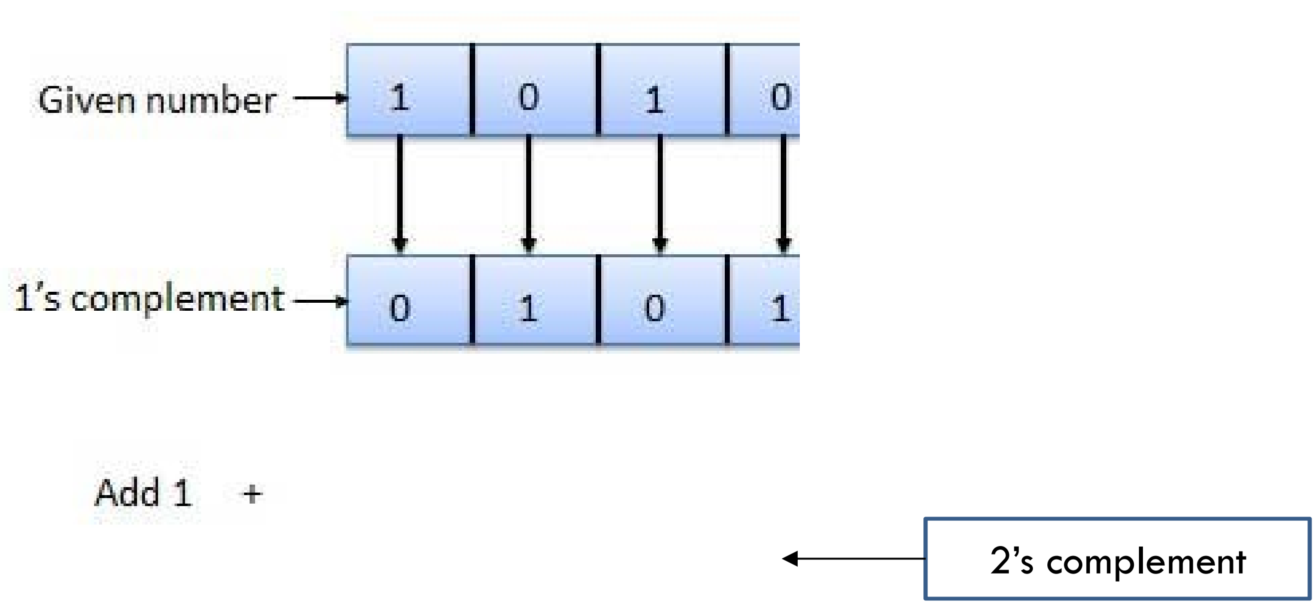
Complement of a number

- Complements are used in the digital computers in order to simplify the subtraction operation and for the logical manipulations
- **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. Example of 1's Complement is as follows:



Complement of a number

- **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:

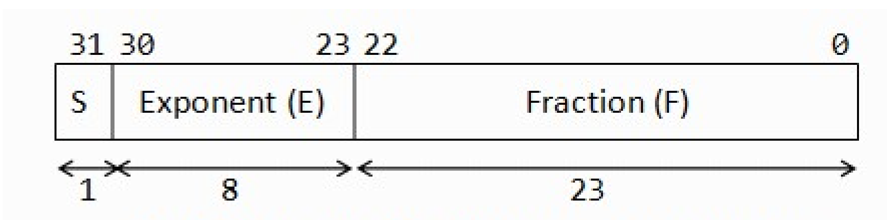


Data Representation: Floating point

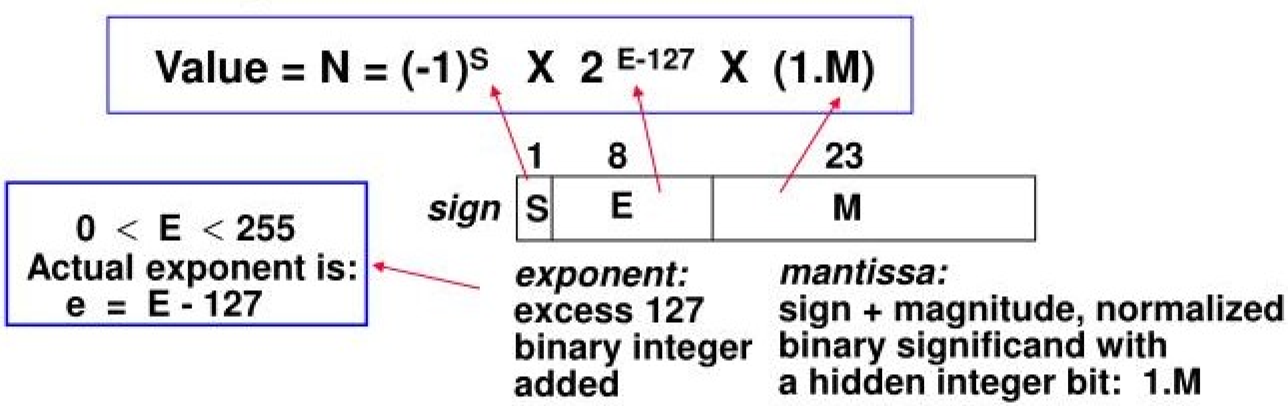
- In computers, floating-point numbers are represented in scientific notation of fraction (F) and exponent (E) with a radix of 2, in the form of $F \times 2^E$
- Both E and F can be positive as well as negative
- Modern computers adopt IEEE 754 standard for representing floating-point numbers
- There are two representation schemes:
  - 32-bit single-precision
  - 64-bit double-precision

Data Representation: Floating point

- **32-bit Single-Precision Floating-Point Numbers**
  - The most significant bit is the *sign bit* (S), with 0 for positive numbers and 1 for negative numbers
  - The following 8 bits represent *exponent* (E)
  - The remaining 23 bits represents *fraction* (F) / Mantissa (M)



32-bit Single-Precision Floating-point Number



Examples of Single-Precision Floating-Point Numbers

- What is the decimal value of this Single Precisio

1	0	1	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- Solution:
  - Sign = 1 is negative
  - Exponent = $(01111100)_2$ = 124, E – bias = 124 – 127
  - Significand = $(1.0100 \ldots 0)_2 = 1 + 2^{-2} = 1.25$ (1. is imp
  - Value in decimal = $-1.25 \times 2^{-3} = -0.15625$
- What is the decimal value of?

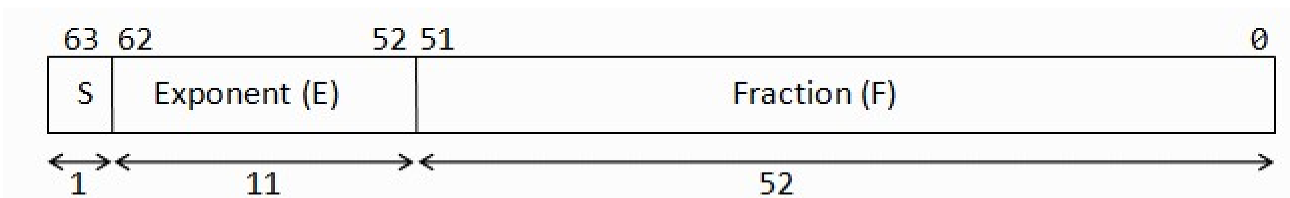
0	1	0	0	0	0	0	1	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- Solution:

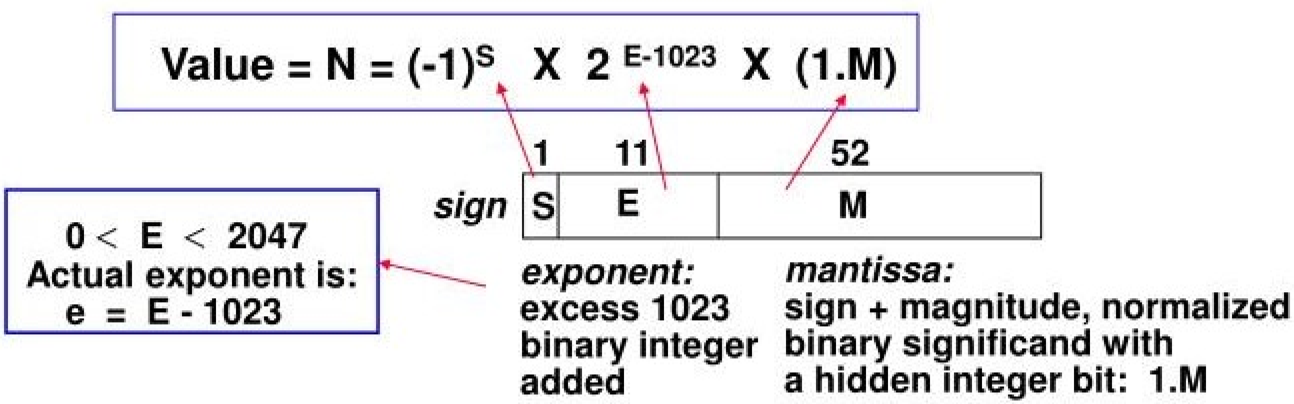
implicit –

Data Representation: Floating point

- **In 64-bit Double-Precision Floating-Point Numbers**
  - The most significant bit is the sign bit (S), with 0 for positive numbers and 1 for negative numbers
  - The following 11 bits represent exponent (E)
  - The remaining 52 bits represents fraction (F) / Mantissa (M)



64-bit Double-Precision Floating-point Number



Examples of Double-Precision Floating-Point Numbers

- What is the decimal value of this Double Precis

0	1	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- Solution:
  - Value of exponent = $(10000000101)_2$ – Bias = 1029
  - Value of double float = $(1.00101010 \ldots 0)_2 \times 2^6$ (1. i

    $(1001010.10 \ldots 0)_2 = 74.5$
- What is the decimal value of ?

1	0	1	1	1	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Data Representation: char

- In computer memory, character are "encoded" (or "represented") using a chosen "character encoding schemes"

- It is important to note that the representation scheme must be known before a binary pattern can be interpreted

- e.g. the 8-bit pattern "0100 0010B" could represent anything under the sun known only to the person encoded it

- The most commonly-used character encoding schemes are:
  - 7-bit ASCII
  - Unicode

ASCII

- It is an acronym for the **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange

- It is a standard **seven-bit code** that was first proposed by the American National Standards Institute or ANSI in 1963, and finalized in 1968 as ANSI Standard X3.4

- In the **ASCII character set,** each binary value between 0 and 127 represents a specific character

- Most computers extend the ASCII character set to use the full range of 256 characters available in a byte

ASCII

- An uppercase "A," for example, is represented by the decimal number 65."
- By looking at the ASCII table, you can clearly see a one-to-one correspondence between each character and the ASCII code used.

Name	Hex	Dec
. (period)	2E	046
0	30	048
1	31	049
2	32	050
3	33	051
4	34	052
5	35	053
6	36	054
7	37	055
8	38	056
9	39	057

Name	Hex	Dec
A	41	065
B	42	066
C	43	067
D	44	068
E	45	069
F	46	070
G	47	071
H	48	072
I	49	073
J	4A	074
K	4B	075

Name	Hex	Dec
L	4C	076
M	4D	077
N	4E	078
O	4F	079
P	50	080
Q	51	081
R	52	802
S	53	083
T	54	084
U	55	085
V	56	086

Name	Hex	Dec
W	57	087
X	58	088
Y	59	089
Z	5A	090

- For example, 32 is the ASCII code for a space

- Code numbers
- 65 to 90 represents 'A' to 'Z'
- 97 to 122 represents 'a' to 'z'
- 48 to 57 represents '0' to '9'

ASCII - Binary Character Table (Sample)

Letter	ASCII Code	Binary	Letter	ASCII
a	097	01100001	A	06
b	098	01100010	B	06

Unicode

- In the pre-Unicode environment, we had 8-bit characters, which limited us to a maximum limit of 256 characters
- No single encoding could contain enough characters to cover all the languages
- Many encoding schemes are in conflict of each other, i.e., the same code number is assigned to different characters

- Unicode aims to provide a standard character encoding scheme, which is universal, efficient, uniform and unambiguous.
- Unicode is a computing industry standard for the consistent encoding, representation and handling of text expressed in most of the world's writing systems

- Unicode provides a unique number for every character:
  - no matter what the platform
  - no matter what the program
  - no matter what the language

Unicode

- Where is Unicode Used?
  - The Unicode standards has been adopted by many software and hardware vendors
    - Most OSs support Unicode
    - Unicode is required for international document and data interchange
    - Programming languages such as Java, C#, Perl, Python
    - Markup Languages such as XML, HTML, XHTML, JavaScript
- Unicode Transformation Format (UTF)
  - An algorithmic mapping from virtually every Unicode code point to a unique byte sequence
  - UTF Encodings Types
    - UTF-8
    - UTF-16
    - UTF-32

Data Representation: String

- Being able to represent individual characters in specified memory locations is very useful, but it is not very convenient for the way we normally want to work with text information.
- Typically, when we work with text, we work with "strings" of characters.
- The obvious way to represent a string in memory is as a sequence of ASCII codes - and this is exactly what is done.

- **Endianess (or byte-order):**
  - For a multi-byte character, you need to take care of the order of the bytes in storage.
  - In *big endian*, the most significant byte is stored at the memory location with the lowest address (big byte first).
  - In *little endian*, the most significant byte is stored at the memory location with the highest address (little byte first).

Software

- Software refers to a collection of programs
- There are two types of software:

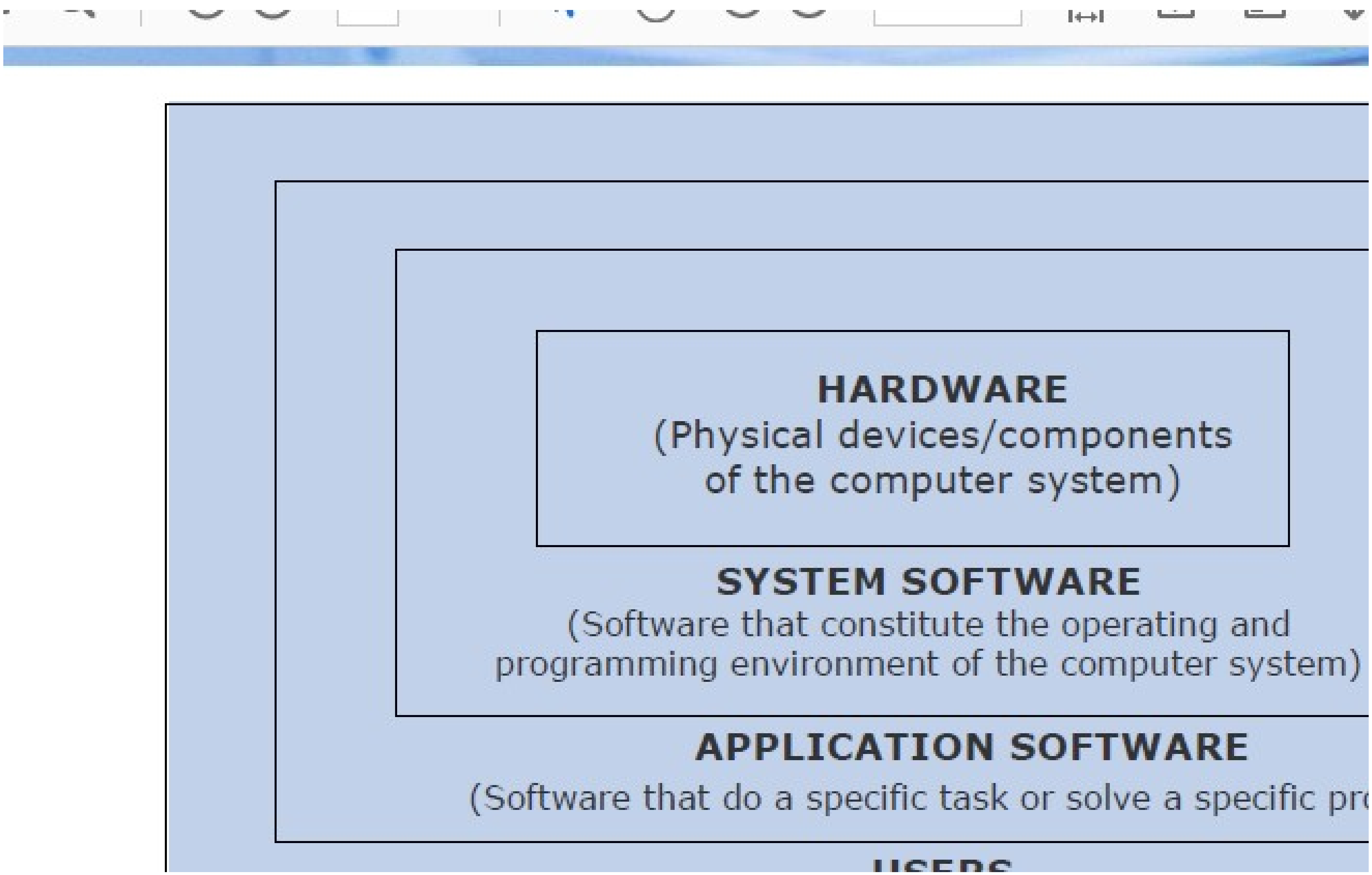
  1. System Software
  2. Application Software

- System software are designed to control the operation and extend the processing capability of a computer system
  - e.g. Operating System, Compiler, Assembler, Linker, Loader
- Application software are designed to solve a specific problem or to do a specific task
  - e.g. MS Word, Paint, Web Browsers (like Chrome, Mozilla Firefox etc.)

Introduction to System Software

- System Software are programs that are designed specifically for running the hardware on a personal computer
- This means that system software is designed to communicate with the internal parts of your computer such as the hard drive, RAM, ROM, cache, microprocessors, etc. so that the user doesn't have to.
- System software can be separated into two different categories, **Utility Programs and Operating Systems**
  - The OS boots up the computer and makes sure everything is operational.
  - Utility programs perform a very specific task, to either enhance or manage your computer, for example your virus protection program

Introduction to System Software



- **Editor:**

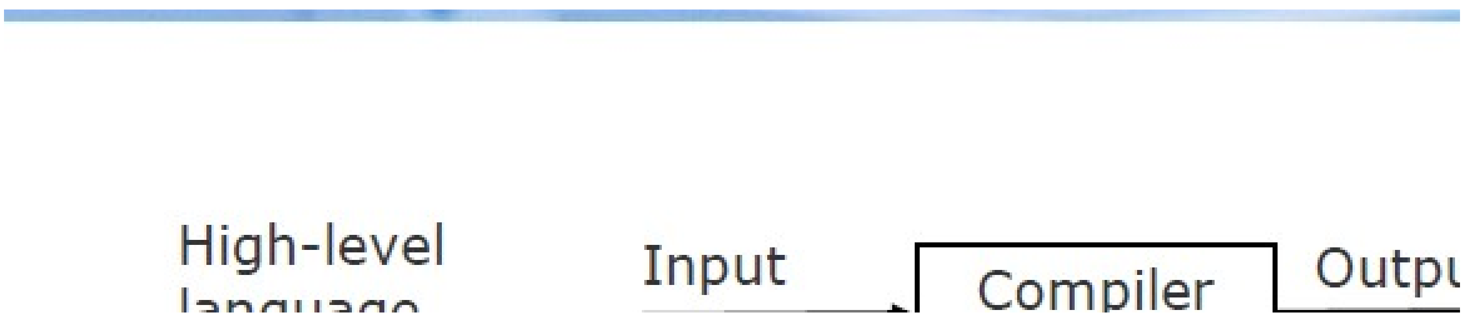
  - A program that enables you to create and edit text files

  - **Source Code Editor** is an standalone application required for writing or editing the source code. A source code editor checks for syntaxes while user writes a code and immediately warns of syntax errors.

  - **Text Editor** is a simple application required for editing plain text such as notepad.

Introduction to System Software

- **Compiler:**
  - It is a program which translates a high level language program into a machine language program.
  - A compiler is more intelligent than an assembler. It checks all kinds of limits, ranges, errors etc.



Introduction to System Software

- **Assembler:**

  - A computer will not understand any program written in a language, other than its machine language
  - The programs written in other languages must be translated into the machine language
  - Such translation is performed with the help of software
  - A program which translates an assembly language program into a machine language program is called an assembler

program into an equivalent machine
of a computer

- **Linker:**

  - In high level languages, some built in header files or libraries are stored
  - These libraries are predefined and these contain basic functions which are essential for executing the program
  - These functions are linked to the libraries by a program called Linker
  - If linker does not find a library of a function then it informs to compiler and then compiler generates an error
  - The compiler automatically invokes the linker as the last step in compiling a program

Introduction to System Software

- **Loader:**
  - Loader is a program that loads machine codes of a program into the system memory
  - In Computing, a **loader** is the part of an Operating System that is responsible for loading programs
  - It is one of the essential stages in the process of starting a program, because it places programs into memory and prepares them for execution
  - Loading a program involves reading the contents of executable file into memory
  - Once loading is complete, the operating system starts the program by passing control to the loaded program code

Steps to execute a High Level Program

Example: C Program Execution