INTRODUCTION TO INFORMATION AND COMMUNICATION TECHNOLOGY

Complete Notes

Name	Muhammad Taha
Roll#	25F-0755
Dept.	BSCS
Section	1B

Introduction to Information and Communication Technology

What is a Computer and What Does It Do?

A **computer** is a programmable electronic device that:

1. Accepts data
2. Performs operations on data
3. Presents results
4. Stores data/results as needed

Four Primary Operations of a Computer:

1. Input
    - Entering data into the computer (e.g., keyboard, mouse, scanner).
2. Processing
    - Performing operations on the input data (e.g., calculations, comparisons, decision-making by CPU)
3. Output
    - Presenting the results of processing (e.g., monitor, printer, speakers).
4. Storage
    - Saving data, programs, or output for future use (e.g., hard drive, SSD, cloud storage).

Data vs. Information:

A user **inputs data** into a computer, and then the computer processes it. When data is processed into a **meaningful form**, it becomes **information**.

1. History / Evolution of Computers

Earliest Devices:

- **Abacus** → **(**First manual calculator with beads on rods.**)**
- **Pascal (France)** → invented the **Pascaline**. (first Mechanical calculator using interlocked gears , **automatic gear movement**.) → Pascaline → Addition & Subtraction
- Leibniz (Germany) → improved Pascal's work → Stepped Reckoner (name of Leibniz Calculator) → All four operations (i.e. addition, subtraction, multiplication and division)
- **Charles Babbage (England)** → **Father of the Computer**. → Analytical engine, used **punch cards** (algorithms on paper, Punch Cards Help in Algorithms changes , like we can change the input and get another result).

Electromechanical Stage-Electronically Controlled Mechanical Relays:

- **Stibitz (1940, Bell Labs)** → made the first Small electromechanical calculator using relays.
- Harvard Mark-I (1944) → first big (large) electromechanical computer, automatic, built at Harvard.

ENIAC (Electronic Numerical Integrator and Computer):

Facts of ENIAC :

- Area: 1800 sq. ft
- Components: 20,000 vacuum tubes, 1500 relays, 10,000 capacitors, 70,000 registers
- Power: 200 KW
- Weight: 30 tons
- Cost: $487,000 (~62.5M PKR)

Rapid Advancements:

- **Transistor** → Smaller, faster, efficient.
- **Integrated Circuits (ICs)** → Miniaturization.
- **Moore's Law** → Processing doubles ~2 years.
- **1976**: Apple desktop (Steve Jobs & Stephen Wozniak).
- **1981**: IBM PC.
- **Modern Era**: Web, smartphones, AI.

Miniaturization:
is the process or act of making something significantly smaller in size

Moore's Law:
says transistors double every 2 years **not automatically**, but because that was the **practical speed engineers and companies could sustain** — faster (like 6 months) would be too hard and too expensive.

2.Generations of Computers

Generation	Years	Technology	Features
1st	1944–59	Used Vacuum tubes	Huge, hot, unreliable
2nd	1959–64	Used Transistors	Smaller, faster
3rd	1964–75	ICs (LSI)	Multiprogramming
4th	1975–…	VLSI(Very Large Scale Integrated Circuits) (microprocessor)	PCs, networking
5th	1980–Now	ULSI (Ultra Large Scale Integration) + AI based computers	Robotics, quantum

3. Modern Applications of Computer Science

- **AI in Healthcare** → Diagnosis, drug discovery.
- **AI in Vehicles/Drones** → Self-driving, delivery.
- **Smartphones** → Mobile computing.
- **Social Media** → Personalized feeds, ads.
- **Video Games (VR/AR)** → Immersive experiences.
- **Smart Homes** → IoT devices, Alexa/Google.
- **Surveillance** → CCTV, AI monitoring.

4. Basic Components of a Computer System

1. **Input** (keyboard, mouse)
2. **Output** (monitor, printer)
3. **Storage** (RAM, hard disk)
4. **CPU** = CU + ALU + Registers (brain of computer)

- **Control Unit (CU):** Directs flow of data and instructions.
- **Arithmetic & Logic Unit (ALU):** Performs calculations and logical decisions.
- **Registers:** Small, very fast memory inside CPU.

5. Storage Devices

1. **Primary Memory**
   a. Directly accessible by CPU.
   b. Includes **RAM + ROM + Cache + Registers**.
   c. Stores data + instructions needed by CPU for quick access.
   d. **Fast**, limited in size.
   e. Partly volatile (RAM), partly permanent (ROM).
2. **Main Memory (RAM)**
   a. Part of **Primary Memory**.
   b. Stores **currently running programs & data**. (currently in use)
   c. **Volatile** → erased (lost) data when power is off.
   d. **Fast,** as compared to secondary memory
   e. **ROM** is also main memory in some books but main working memory is only RAM
   f. Examples: DRAM, SRAM.

☞ **Shortcut to remember:**

- **Primary = Broad category (RAM + ROM + Cache + Registers).**
- **Main Memory = Just RAM (working memory).**
- **Volatile = Temporary memory (data lost without power) →** e.g., RAM.
- **Non-Volatile = Permanent memory (data saved even without power) →** e.g., ROM, Hard disk.

Types of RAM:

1. **DRAM** (Dynamic RAM) → Needs refresh, slower, cheaper.
2. **SRAM** (Static RAM) → No refresh, faster, costly (used in cache)

- **Secondary Memory = Permanent storage**.
- **Slower but larger** than primary memory.
- **Non-volatile** (data stays even without power).
- Examples: HDD, SSD, CD, DVD, USB.

SRAM (Static RAM):

- Uses **flip-flop circuits** to store data.
- **Very fast** and reliable (Utilize less power and it is more expensive.).
- **Does not need refreshing** as long as power is on.
- **Expensive** and small in size.
- Used in **Cache Memory (L1, L2, L3)** and CPU registers.

☞ **In short:** Faster, costlier, used in cache.

DRAM (Dynamic RAM):

- Uses **capacitors** to store data (charges = 1, no charge = 0).
- Needs **continuous refreshing** (thousands of times per second).
- **Slower** than SRAM but **much cheaper**.
- Processor cannot access data of DRAM when it is being refreshed.
- **Higher capacity** → used as main system memory (RAM sticks in PCs/laptops).

☞ **In short:** Cheaper, larger, slower, used as main memory.

Types of ROM:

1. PROM (Programmable ROM):

a. Initially empty.
b. User can **program it once** using a special device.
c. Cannot be erased or re-written.

☞ **In short:** Write once, permanent.

2. EPROM (Erasable Programmable ROM):

a. Can be **erased with UV light** and reprogrammed.
b. Used in development/testing phases
c. Has a transparent quartz window to expose it to UV light.

☞ **In short:** Erasable with UV light, reusable.

3. EEPROM (Electrically Erasable Programmable ROM):

a. Can be **erased and re-written electrically**.
b. Slower and smaller storage than modern flash.
c. Used for storing small data like BIOS settings.

☞ **In short:** Erasable electrically, limited re-writes.

Registers:

- Registers = **Fastest memory inside CPU**.
- Store **temporary data/instructions**.
- Act as CPU's **working desk** for execution. (Data is transferred from **main memory (RAM) into registers** for execution)
- CPU contains **several registers**, each with a **predefined function**.

- Types → **Special Purpose** & **General Purpose**.

Cache Memory:

- Pronounced as **"cash"**.
- **Very fast memory**, faster than RAM.
- Stores **most recently or frequently used data** for the CPU.
- On **first use**, CPU retrieves data from RAM.
- A **copy of the data** is stored in cache.
- On **next use**, CPU checks cache first:
  - If data is found → retrieved from cache (faster).
  - If not → retrieved from RAM again.

Relative Fast (speediness):

Registers (Fastest, smallest, inside CPU) → Cache (stores most recent/frequent data) → RAM → Secondary → Tertiary

Trick:
R → C → R → S → T
(RCRST)

Direct Memory Access (DMA):

Definition: A technique that allows **peripheral devices** (like hard drives, SSDs, or network cards) to **transfer/access data directly to/from main memory (RAM) without involving the CPU**.
Purpose: To **speed up data transfer/access** and **free the CPU** to perform other tasks.

How it Works:

1. CPU sets up **DMA controller** with:
   - Source address (device or memory)
   - Destination address (memory or device)
   - Amount of data to transfer
2. **DMA controller** handles the actual data transfer **directly between device and RAM**.
3. CPU is **free during the transfer**, only gets interrupted when the transfer is complete.

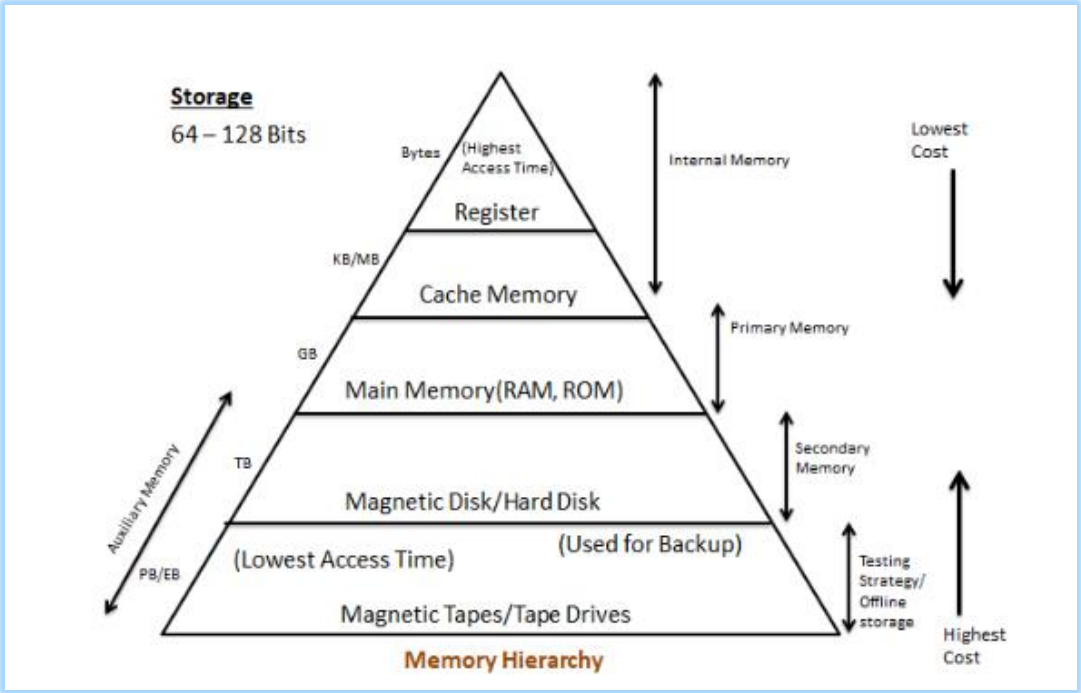
Advantages of DMA:

- Faster data transfer than CPU-managed transfers.
- Reduces **CPU overhead**.
- Efficient for **large data blocks** (like audio, video, disk I/O).

Quick Notes (Quiz-Ready):

- **DMA = Direct Memory Access.**
- **Transfers data directly between device ↔ RAM** without CPU involvement.
- **CPU only sets it up** and is free during transfer.
- **Faster & more efficient**, used in disks, network cards, audio/video devices.

Storage Memory Heirarchy:



Processing Device / CPU (Central Processing Unit):

- **Definition:** The **brain of the computer** that **processes instructions and data**.
- CPU is located on motherboard.
- Performs **all calculations, logic decisions, and controls other components**.

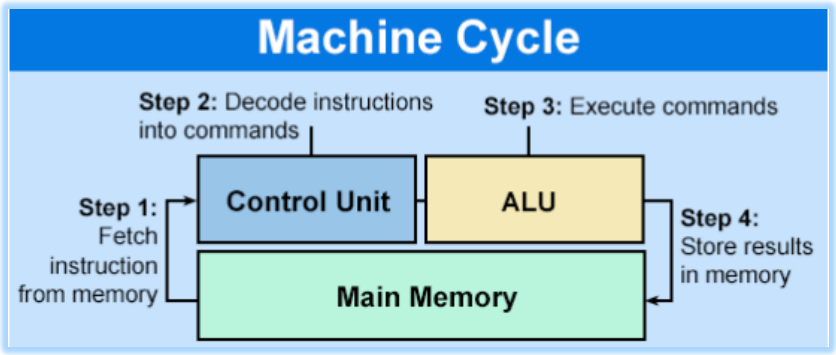
Main Components of CPU:

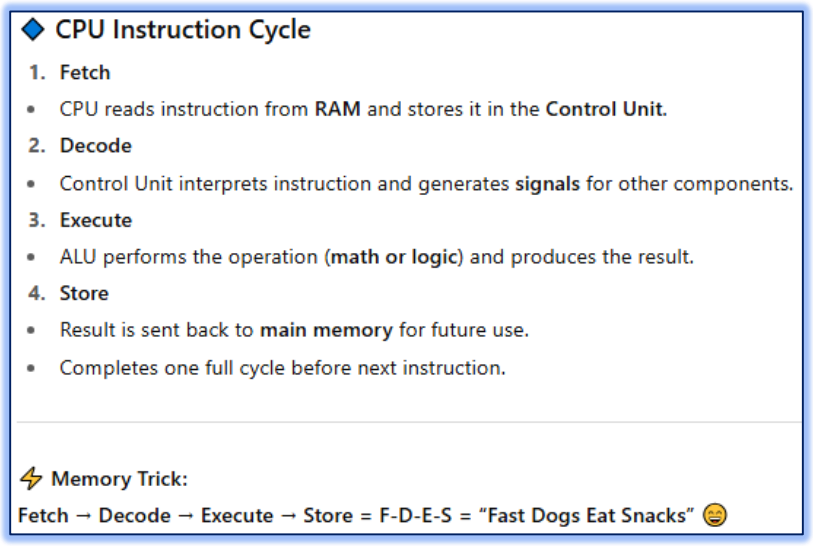
1. **Control Unit (CU)**
   - Directs **flow of data and instructions** between memory, input, output, and ALU.
   - **Manages execution** of programs.
2. **Arithmetic & Logic Unit (ALU)**
   - Performs **arithmetic operations** (add, subtract, multiply, divide).
   - Performs **logical operations** (AND, OR, NOT, comparisons).
3. **Registers**
   - Small, **high-speed storage inside CPU**.
   - Temporarily holds **data, instructions, or addresses** being processed.

In short:

- CPU = **Processing Device / Brain of Computer**.
- CU = controls data flow & program execution (manager)
- ALU = calculator (performs arithmetic & logic operations)
- Registers → tiny high-speed storage inside CPU (for immediate work)
- Works in **fetch → decode → execute → store** cycle

The Machine Cycle





Motherboard (Main Circuit Board):

- **Definition:** The main printed circuit board (**PCB**) of a computer that **connects all components together**.
- Acts as the **backbone of the computer**, allowing communication between CPU, memory, storage, and peripherals.

Key Components on Motherboard:

1. **CPU Socket** → Holds the **processor (CPU)**.
2. **Memory Slots** → For **RAM / Main Memory modules**.
3. **Chipset** → Manages **data flow between CPU, RAM, and peripherals**.
4. **Expansion Slots** → For **graphics cards, network cards, sound cards** (PCI, PCIe).
5. **Storage Connectors** → SATA, NVMe slots for **HDD, SSD, Optical drives**.
6. **Power Connectors** → Supply electricity to motherboard and components.
7. **Input/Output Ports** → USB, HDMI, Ethernet, Audio, etc.
8. **BIOS/UEFI Chip** → Stores **firmware** for booting the computer.

Functions of Motherboard:

- Provides **physical platform** for components.
- Enables **communication** between CPU, memory, storage, and peripherals.
- Supplies **power** to components via connectors.
- Houses **BIOS/UEFI** for system initialization.

Quick Notes (Quiz-Ready):

- Motherboard = **main circuit board connecting all components**. (Holds CPU and connects all subsystems.)
- Contains: **CPU socket, memory slots, chipset, expansion slots, storage connectors, I/O ports**.

- **Function:** Provides power, communication, and platform for all hardware.

System Bus:

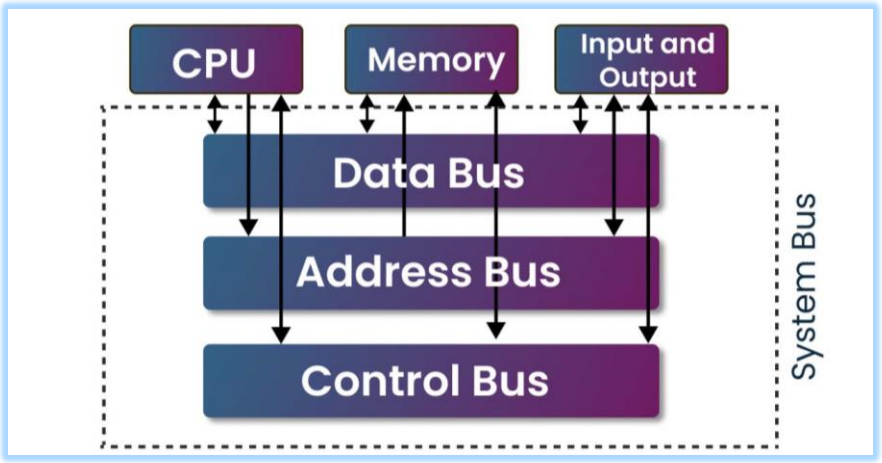
- **Definition:** A **set of parallel wires (pathways)** that allows **data, instructions, and control signals** to flow between **CPU, memory, and peripherals**.
- Think of it as the **communication highway of the computer**.

Types of System Buses:

1. **Data Bus**
   - Carries **actual data** between CPU, memory, and I/O devices.
   - **Bidirectional** (data can go both ways).
   - Width (number of wires) affects **how much data can be transferred at once**.
2. **Address Bus**
   - Carries **memory or device addresses** where data should be read/written.
   - **Unidirectional** (CPU → Memory/Device).
   - Width determines **maximum addressable memory**.
3. **Control Bus**
   - Carries **control signals** to coordinate operations.
   - Signals include **Read, Write, Clock, Interrupts**.
   - Ensures devices act at the **right time**.

Quick Notes (Quiz-Ready):

- **System Bus = Communication highway** of CPU, memory, and I/O.
- **Data Bus:** Transfers actual data.
- **Address Bus:** Transfers memory/device addresses.
- **Control Bus:** Transfers control signals to synchronize operations.
- **Width & speed of buses** affect overall computer performance.



Interrupts:

- **Definition:** A signal sent to the **CPU** by a device or software to **get the CPU's attention**.
- It temporarily **pauses the current program** so the CPU can handle an **urgent task**.
- After handling, CPU **resumes the original program**.

Types of Interrupts:

1. **Hardware Interrupts**
   - Generated by **hardware devices** like keyboard, mouse, printer, or disk.
   - Example: Pressing a key → keyboard sends interrupt to CPU.
2. **Software Interrupts**

- Generated by **programs or software** to request CPU services.
- Example: System call in an OS.

Working of Interrupts:

1. Device or software sends an **interrupt signal** to CPU.
2. CPU **saves the current state** of execution.
3. CPU executes the **Interrupt Service Routine (ISR)** to handle the interrupt.
4. After completion, CPU **resumes the previous task**.

Advantages of Interrupts:

- Efficient CPU usage → CPU doesn't waste time waiting for devices.
- **Immediate attention** to urgent tasks.
- Enables **multitasking and real-time processing**.

Quick Notes (Quiz-Ready):

- **Interrupt = Signal to CPU to pause current task** and handle urgent event.
- **Types:** Hardware (keyboard, mouse) and Software (program requests).
- **CPU executes ISR**, then resumes previous program.
- **Purpose:** Efficient CPU usage & faster response to devices.

Software:

A set of **programs, instructions, and data** that tells the computer **what to do**.
Software = Intangible → cannot be touched physically like hardware.

Types:

- **System Software:** OS & utilities → manages computer.
- **Application Software:** User tasks → Word, Excel, games.
- **Programming Software:** Helps create programs → compiler, IDE.

In short, system software makes the hardware usable while application software handles specific tasks.

Hardware:

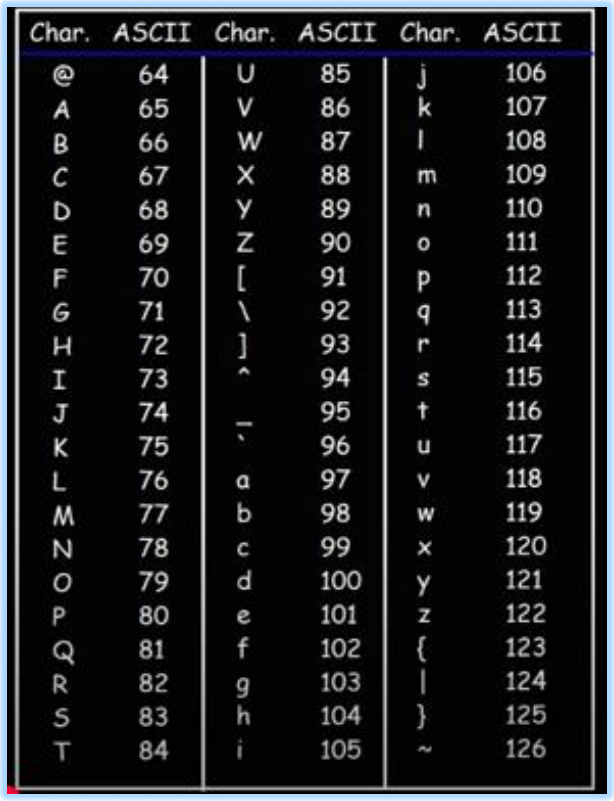
The **physical components of a computer** that you can **see and touch**.

- Hardware **executes instructions** provided by software.

- Hardware = **Physical components of computer**.
- Works according to **software instructions**.
- **Types:** Input(keyboard,etc), Output(monitor,printer), Storage(Hdd,sdd), Processing(CPU, ALU), Communication(Modem, Network Card, Router). Devices.
- **Tangible** → can touch physically.

Language of a computer:

- Computers use **digital signals (0s & 1s)** → machine language.
- **Digital** → 0 = low voltage,
  → 1 = high voltage.
- **Bit = 0 or 1**, sequences of bits form binary code.
- From a shortening of the words "**b**inary dig**it**"
- **Bit** = 1 binary digit.
- **Binary code =** sequences of bits form binary code .
- **ASCII:** 7-bit code, 128 characters, e.g., `A = 1000001`.
- **Assembly Language:** Human-friendly, needs **assembler** to convert to machine code.
- **High-Level Languages:** English-like instructions, need **compiler + linker + loader** to run.



Char.	ASCII	Char.	ASCII	Char.	ASCII
@	64	U	85	j	106
A	65	V	86	k	107
B	66	W	87	l	108
C	67	X	88	m	109
D	68	Y	89	n	110
E	69	Z	90	o	111
F	70	[	91	p	112
G	71	\	92	q	113
H	72	]	93	r	114
I	73	^	94	s	115
J	74	_	95	t	116
K	75	`	96	u	117
L	76	a	97	v	118
M	77	b	98	w	119
N	78	c	99	x	120
O	79	d	100	y	121
P	80	e	101	z	122
Q	81	f	102	{	123
R	82	g	103	|	124
S	83	h	104	}	125
T	84	i	105	~	126

Character	ASCII
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57

Programming Levels:

- **Machine Language** → Binary only.
- **Assembly Language** → Mnemonics, needs **assembler**.
- **High-Level Languages (HLLs)** → C, C++, Java, closer to English.

Tools:

- **Assembler** → Assembly → Machine code.
- **Compiler** → HLL → Machine code.
- **Linker** → Combines object + libraries.
- **Loader** → Loads executable into RAM.

ASCII (American Standard Code for Information

Interchange):

ASCII = decimal number
then we convert decimal number into binary code(number) by two ways:

Method 1: Division by 2 (Repeated Division):

Steps:

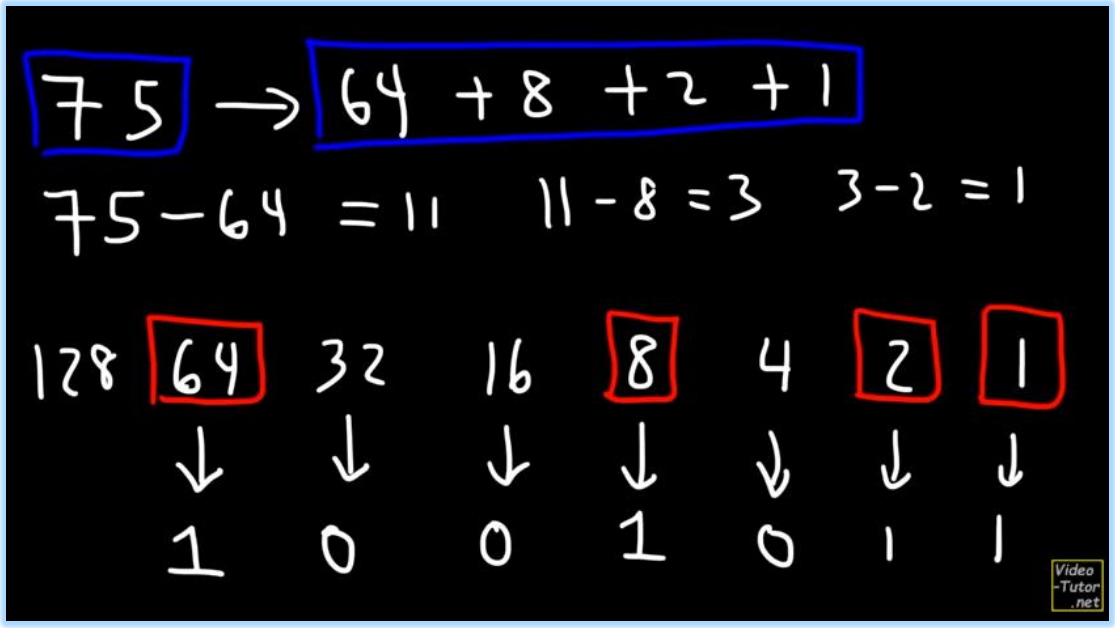
1. Divide the decimal number by **2**.
2. Write down the **remainder** (0 or 1).
3. Divide the **quotient** by 2 again.
4. Repeat until the quotient becomes **0**.
5. **Binary number = remainders read from bottom to top**. (write remainders bottom to top.)

Method 2: Subtraction (Place Value):

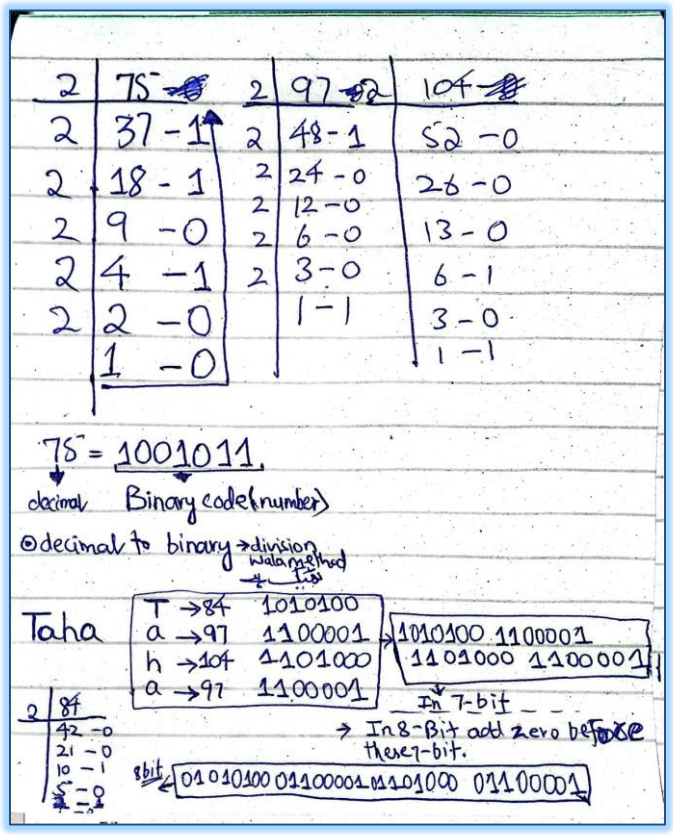
Steps:

1. Write powers of 2 ≤ decimal number: 8, 4, 2, 1
2. Subtract the largest possible power of 2 from the number and mark **1**.
3. Mark **0** for powers not used.
4. Binary = sequence of 1s and 0s from highest to lowest power.

Example: 13 → 8 + 4 + 0 + 1 → Binary = **1101**



EXAMPLE:

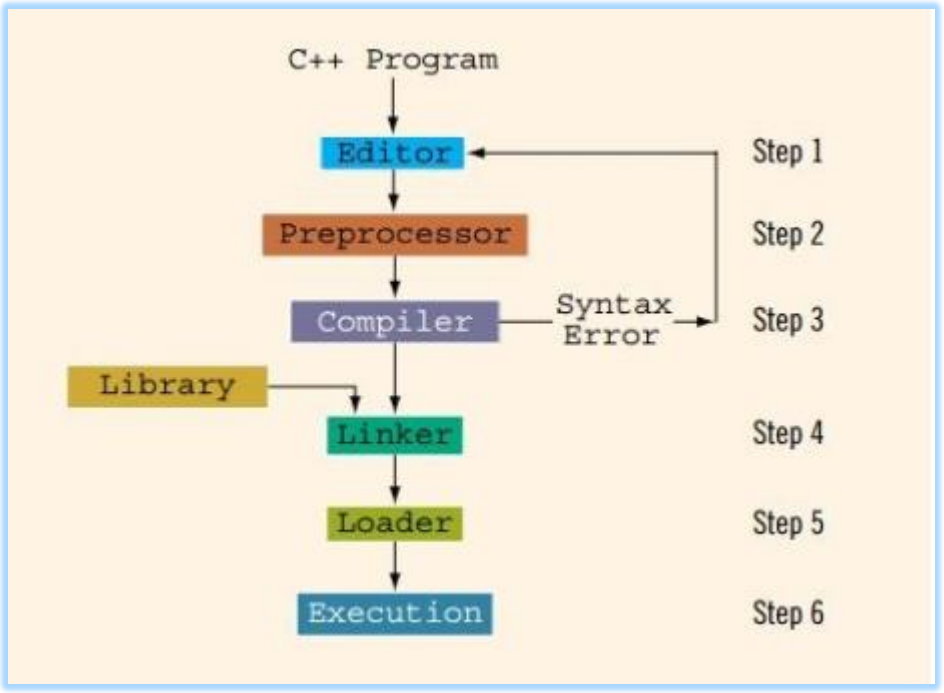


- 7-bit mai ans aye ga so inko agr 8 bits mai krna tu each decimal ka binary ka start mai 0 laga do simple.

Data Units in Computers:

- **Bit = smallest data unit (0 or 1)**
- **1 Byte = 8 bits**
- Each keyboard character is stored as a Byte (Example A = 01000001)
- **Data increases by 1024 (2^{10}) times** at each level (KB → MB → GB → TB → PB)
- Common usage:
  - KB → text files
  - MB → images, documents
  - GB → videos, software
  - TB → hard disks, cloud storage
  - This can vary file quality or quantity wise. (that's just the

Bits → Bytes → Kilo → Mega → Giga → Tera → Peta



Information System

- An **Information System** is a combination of **people, technology, processes, and data** that work together to **collect, process, store, and distribute information**.
- Purpose: To **support decision-making, coordination, control, analysis, and visualization** in an organization.

Components of an Information System:

1. **Hardware** – Physical devices (CPU, memory, storage, input/output devices).
2. **Software** – Programs and applications that process data.
3. **Data** – Raw facts that are processed into meaningful information.
4. **People** – Users who interact with the system (managers, IT staff, employees).
5. **Processes / Procedures** – Rules and methods to operate the system efficiently.

Hardware:

The **physical components of a computer** that you can **see and touch**.

- Hardware **executes instructions** provided by software.

- Hardware = **Physical components of computer**.
- Works according to **software instructions**.
- **Types:** Input(keyboard,etc), Output(monitor,printer), Storage(Hdd,sdd), Processing(CPU, ALU), Communication(Modem, Network Card, Router). Devices.
- **Tangible** → can touch physically.

Software:

A set of **programs, instructions, and data** that tells the computer **what to do**.
Software = Intangible → cannot be touched physically like hardware.

Types:

- **System Software:** OS & utilities → manages computer.
- **Application Software:** User tasks → Word, Excel, games.
- **Programming Software:** Helps create programs → compiler, IDE.

In short, system software makes the hardware usable while application software handles specific tasks.

- Nowadays, storage services are offered from the cloud, which can be accessed from telecommunications networks.

Data Storage Technologies

Data Storage Technology = Method & device to store digital data.

- Provide **structures and models** to **store, retrieve, and analyze data efficiently**.
- **Main Categories:**
  1. Relational Databases
  2. NoSQL Databases
  3. Data Warehouses

Relational Databases (RDBMS):

- Store **structured data** in tables (rows & columns).
- **Reliable** with **strong data integrity**.
- Used for **transactional applications**.
- **Examples:** MySQL, Oracle, SQL Server

NoSQL Databases:

- Stands for **"Not Only SQL"**.
- Designed for **unstructured and semi-structured data**.
- **Types:**
  - Document (e.g., MongoDB)
  - Key-Value (e.g., Redis)
  - Column-family (e.g., Cassandra)
  - Graph
- **Features:** Flexible schema, highly scalable
- **Examples:** MongoDB, Cassandra, Redis

Data Warehouses:

- **Central repository** for **large volumes of structured data** from multiple sources (databases, files, apps).
- Stores **historical data** for **business intelligence and analytics**.
- **Examples:** Amazon Redshift, Snowflake, Google BigQuery
- **Use Case:**
    - Amazon collects sales data from all countries → managers analyze to decide which products to promote in which region.

Comparison

Type	Features / Usage	Examples
Relational Databases	• Store **structured data** in tables (rows & columns). • **Reliable** with **strong data integrity**. • Used for **transactional applications**.	MySQL, Oracle, SQL Server
NoSQL Databases	• Stands for **"Not Only SQL"**. • Designed for **unstructured and semi-structured data**. • **Features:** Flexible schema, highly scalable	MongoDB, Cassandra, Redis
Data Warehouses	Stores **historical data** from multiple sources for **business intelligence and analytics** Analytical, historical, supports BI tools	Amazon Redshift, Snowflake, BigQuery

Information System (Networks/Computer Networks)

Quick Notes (Quiz-Ready):

- **Network = Connected computers/devices sharing resources.**
- **Purpose:** Share data, communication, collaboration, access resources.
- **Types:** LAN, WAN, MAN, PAN, CAN
- **Components:** Router, Switch, Hub, Modem, Cables/Wi-Fi

Types of Networks:

Sr.	Type	Description	Example
1.	**LAN (Local Area Network)**	Small area network, e.g., home, office	Home Wi-Fi, office network
2.	**WAN (Wide Area Network)**	Covers large areas, countries, or continents	Internet, corporate networks
3.	**MAN (Metropolitan Area Network)**	Covers a city or town	City government networks
4.	**PAN (Personal Area Network)**	Very small network, personal devices connected	Bluetooth devices, smartphone + laptop
5.	**CAN (Campus Area Network)**	Covers university or company campus	University campus network

Transmission Modes:

1. **Wired Networks:** Use physical cables to transmit data.
    - Examples: Fiber optic cables, Coaxial cables
2. **Wireless Networks:** Use electromagnetic signals to transmit data.
    - Examples: Radio waves, Microwaves, Wi-Fi

Information System (People)

1. Role of People in IS

- Users and stakeholders interact with the **information system**.
- Provide **input, operate the system, and make decisions** based on information.

2. Feedback Mechanism

- **Information systems often include mechanisms** to collect feedback from:
    - Users
    - Other sources (e.g., sensors, logs)
- **Purpose of Feedback:**
    - Improve system **performance**
    - Enhance **effectiveness**
    - Identify **errors or inefficiencies**

3. Quick Notes (Quiz-Ready)

- **People = Users / Operators / Decision-makers** in IS
- **Feedback = Input from users & sources**
- **Goal:** Improve **performance and effectiveness** of the system

Need for Information System (IS):

1. **Fast and Accurate Information**
2. **Fast Communication Network**
3. **Support for Decision-Making**
    - Helps managers and decision-makers to:
        - **a)** Analyze data
        - **b)** Make informed decisions
        - **c)** Plan and control operations effectively
4. **Reduction in Information Load**
    - Filters and organizes data to prevent **information overload**.
    - Ensures **relevant information** is delivered to the right person at the right time.

Common Number Systems

Number system can be categorized as:

- Decimal number system
- Binary number system
- Octal number system
- Hexadecimal Number System

- Each number system is associated with a base or radix
  - The decimal number system is said to be of base or radix 10
- A number in *base r* contains r digits 0,1,2,...,r-1
  - Decimal (Base 10): 0,1,2,3,4,5,6,7,8,9

System	Base	Symbols	Used by humans ?	Used in computer s?
Decimal	10	0, 1, … 9	Yes	No
Binary	2	0, 1	No	Yes
Octal	8	0, 1, … 7	No	yes
Hexa-decimal	16	0, 1, … 9, A, B, … F	No	yes

Number Conversions [(binary, decimal, octa decimal, Hexadecimal)

Number System Conversion:

1. Decimal (Base 10)
   - Uses digits 0–9.
   - Example: 245_{10}
2. Binary (Base 2)
   - Uses digits 0 and 1.
   - Example: 1011_2
3. Octal (Base 8)
   - Uses digits 0–7.
   - Example: 17_8
4. Hexadecimal (Base 16)
   - Uses digits 0–9 and A–F (where A=10 … F=15).
   - Example: $1A_{16}$

Conversion:

Conversion:

Decimal to Binary: $(75)_{10} = (1001011)_2$

2	75
2	37 – 1 ↑
2	18 – 1
2	9 – 0
2	4 – 1
2	2 – 0
	1 – 0

Decimal to Octal: $(75)_{10} = (113)_8$

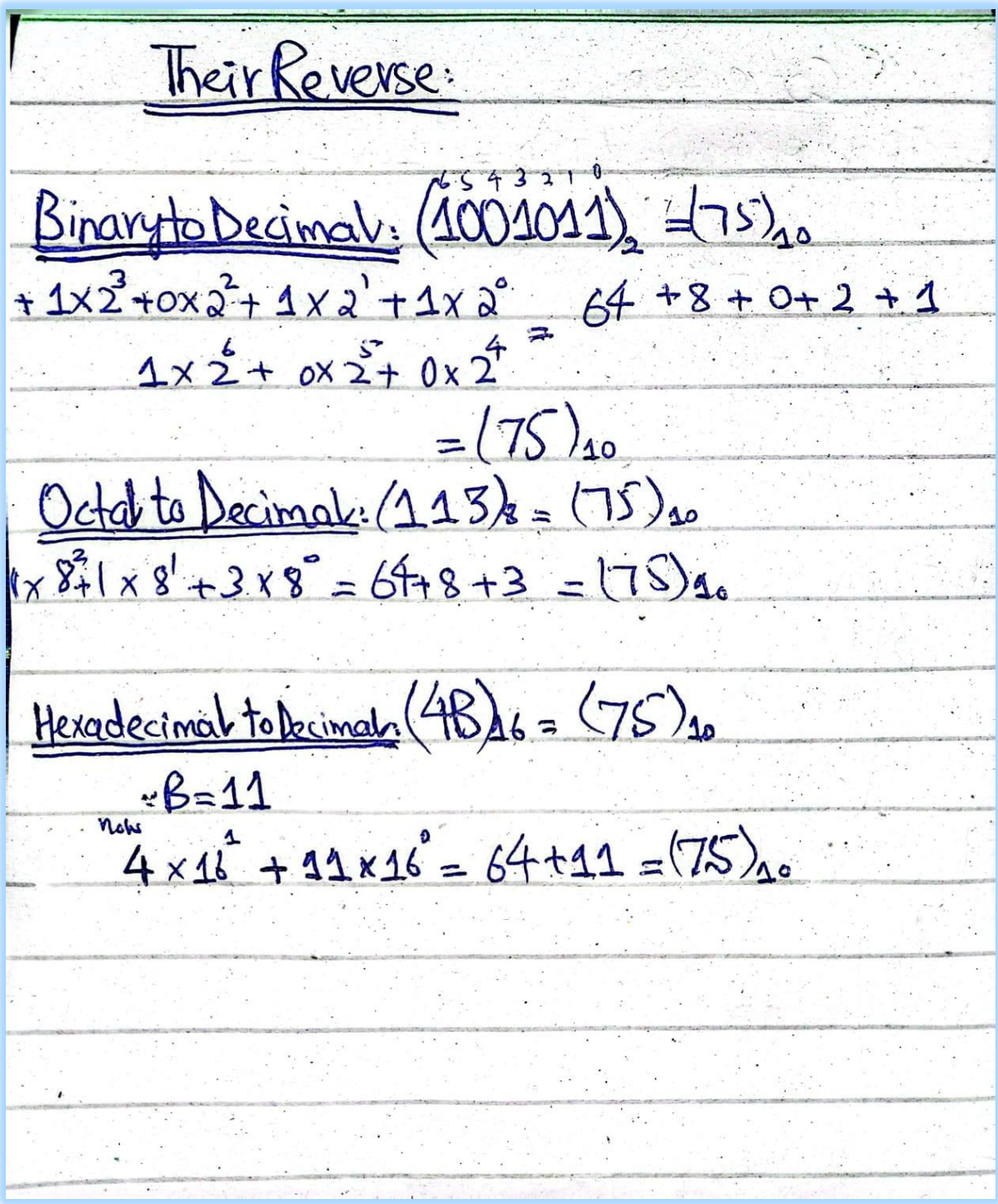
8	75
8	9 – 3 ↑
	1 – 1

Decimal to Hexadecimal: $(75)_{10} = (4B)_{16}$

16	75
	4 – 11 ↑

Hexa, decimals

$6 + 10 = 16$ → 10 → 0–9; → 6 → A–F



- Convert (from decimal into binary, octal, hexadecimal, etc.) using **division by base**.

Addition:

if:

- sum < base → then write sum
- sum ≥ base → write sum-base & carry 1

Subtraction:

while borrowing:

- jis se le raha us se -1 kr
  or jis mai se raha usma base add kr

example

①
$$\begin{array}{r} 572_8 \\ +245_8 \\ \hline 1037_8 \end{array}$$

R.W
11 - 8 = 3
8 - 8 = 0

②
$$3C_{16} - 1A_{16} = ?$$

$$\begin{array}{r} 3\ \ 12 \\ -1\ \ 10 \\ \hline 2\ \ 2_{16} \end{array}$$

③
$$2D_{16} - 1E_{16} = ?$$

$$\begin{array}{r} 1 \\ 2\ \ 13 \\ -1\ \ 14 \\ \hline 0\ \ 15_{16} \end{array}$$

13 + 16 = 29 - 14 = 15

For Multiplication and division → convert the number in the decimal then do multiplication / division and then convert back into original number system.

1. One's Complement:

One's complement of a binary number is obtained by **inverting all bits**:

- $0 \rightarrow 1$
- $1 \rightarrow 0$

How to Find 1's Complement

Just **flip every bit**.

Example

Binary number:

```
101100
```

One's complement:

```
010011
```

2. Two's Complement:

Two's complement is obtained by:

1. Taking **1's complement**
2. **Adding 1** to the result (i.e. add 1 to the least significant bit i.e. to the right one)

How to Find 2's Complement (Steps)

Given:

```
101100
```

Step 1: One's complement

```
010011
```

Step 2: Add 1

```markdown
010011
+     1
-------
010100
```

So, **2's complement = 010100**

Shortcut Method (Very Important)

> **From right, copy all bits up to the first 1, then invert the remaining left bits.**

Example:

```
101100
```

- Copy from right: `100`
- Invert remaining `101 → 010`

Result:

```
010100
```

Like from the right when the first 1 (pahla 1) aye tu stop coping then remaining part ko invert kr do

Fractions in Number System (point mai)

There are **two parts** to handle:

1. **Integer part (**whole number to the left of the decimal point**)**
2. **Fractional part (**decimal portion to the right of the decimal point**)**

Steps of conversion:

1. Integer Part Conversion:

Convert (from decimal into binary, octal, hexadecimal, etc.) using **division by base (**uski base jis mai convert kr raha hain**)** (and then arranging remainders from bottom to top).

2. Fractional Part Conversion:

Convert using **multiplication by base (**uski base jis mai convert kr raha hain**)**. (and then arranging the integers from top to bottom)

3. Combine Both

IEEE (Institute of Electrical and Electronics Engineers)

IEEE is a professional organization that sets **standards** for electronics, electrical, and computer engineering technologies, including **floating-point representation** in computers.

1. Common Conversion Steps (Same for ALL):

No matter whether it is **half, single, or double precision**, you always:

1. Convert decimal → binary
2. Normalize → 1.x×2 ki power e (exponent)
3. Determine **sign bit**
4. Calculate **biased exponent → e + bias**
5. Write **mantissa** (fraction only)

Format Comparison (MEMORIZE THIS TABLE)					
Precision	**Total Bits**	**Sign**	**Exponent**	**Mantissa (Fraction)**	**Bias**
Half (IEEE-754)	16	1	5	10	**15**
Single (IEEE-754)	32	1	8	23	**127**
Double (IEEE-754)	64	1	11	52	**1023**
Quadruple (IEEE-754)	128	1	15	112	**16383**

Bias formula = [2^(k-1)] – 1

Where k = number of exponent bits

2. Decoding (Reverse Conversion)

Same idea:

1. Read sign
2. Exponent − bias
3. Add implicit **1.** before mantissa
4. Multiply by 2^e

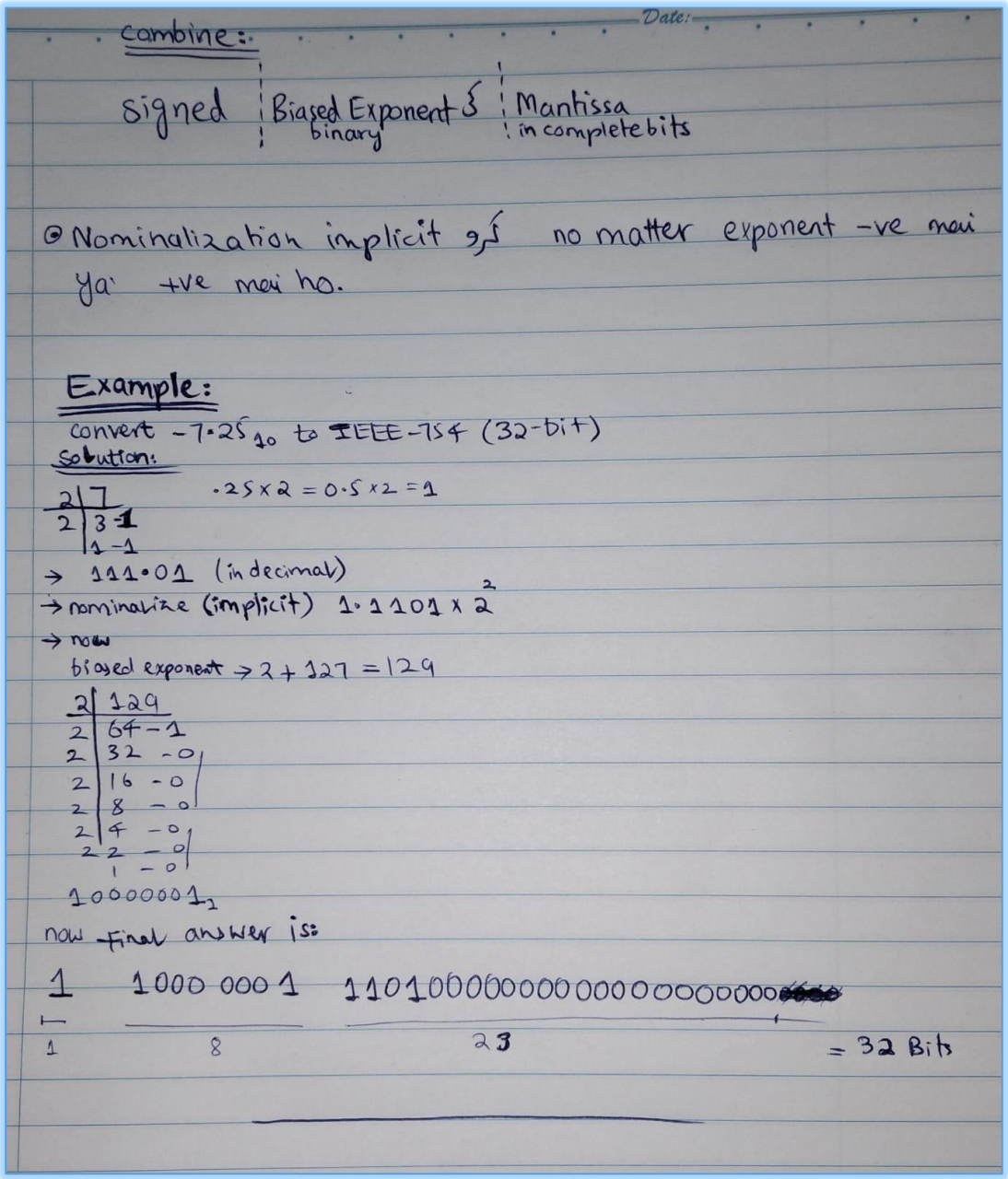
✓ Bias depends on precision

✓ Mantissa length depends on precision

FAST-NUCES Exam Tip

If the question says:

- If a question says **“IEEE”**, it almost always refers to **floating-point representation**.
- **“IEEE-754” only** → assume **single precision**
- **“Half / Single / Double” mentioned** → use correct **bias + bits**
- **Always write bias value** to avoid losing marks



ASCII

ASCII = **American Standard Code for Information Interchange**

It is a **standard encoding** that represents **characters (letters, digits, symbols)** as **numbers**, which can then be stored in binary.

HOW TO FIND ANY WORD BECOMING BY BINARY:

(agr koi binary di ho long tu un mai 8 , 8 bits ka group bana laina or prh usko decimal mai convert krna then vo ascii hai so us decimal sa jo banta ho vo bana daina)

Each character in original ASCII = 7 bits

Each character in extended ASCII = 8 bits → use 8 bit extended ascii

Key ASCII Ranges	
Decimal	**Character Type**
0–31	Control characters (e.g., newline, tab)

32–47	Symbols (space, ! " # …)
48–57	Digits 0–9
65–90	Uppercase A–Z
97–122	Lowercase a–z
128–255	Extended symbols (extended ASCII)

3. How to Convert Binary → Text Using ASCII

Step 1: Split the binary number

- If 8-bit ASCII, split into **8-bit groups**
- Example: `01001000 01101001`

Step 2: Convert each binary group to decimal

- `01001000` → 72
- `01101001` → 105

Step 3: Look up the ASCII table

- 72 → `H`
- 105 → `i`

Step 4: Combine characters

01001000 01101001 → Hi

Quick Binary-to-Text Example

Binary:
`01000001 01000010 01000011`
Step 1: Split → 8-bit groups
Step 2: Convert to decimal: 65, 66, 67
Step 3: Look up ASCII table → A, B, C
Answer: `ABC`

FAST Exam Tips

1. Always **split binary correctly** (7 or 8 bits)
2. Use **decimal conversion first**, then ASCII
3. Remember key ranges:
    - 48–57 → 0–9
    - 65–90 → A–Z
    - 97–122 → a–z