Introduction to Computer Interfacing

CSE360 – Computer Interfacing

Contents

- What is computer interfacing?
- How does a computer work?
- Software hierarchy of a computer system
- Buses and Ports: Bus architectures, Tri-state logic, Bus conflicts, Glitch, Different types of ports
- Microprocessor vs Microcontroller
- Embedded systems vs Systems on a chip (SoC)
- BIOS, Operating system vs Driver vs Firmware

What is computer interfacing?

- An interface is a shared boundary where two or more separate components (I/O devices, Memory, etc.) of a computer system exchange information.

- Types of Computer Interfaces:
  - Software Interface: Interaction between programs or modules (e.g., API).
  - Hardware Interface: Communication between hardware components (e.g., CPU, I/O devices and memory).
  - User Interface (UI): Where humans interact with computers (e.g., GUI or command line).
  - Network Interface: Where systems communicate over networks (e.g., network cards).

- Examples:
  - Human-Machine Interface (HMI): Like touch screens or keyboards.
  - Peripheral Interfaces: Communication with devices like printers, mice, or storage.
  - Software Interfaces: APIs allowing programs to interact (e.g., REST, SOAP).

Peripheral Devices & Communication Links

- **Peripheral Communication Links:**
  - **Peripherals** connected to a computer require special communication links to interface with the **CPU**.
  - **Input-output interfaces** are hardware components that manage data transfers between the **CPU** and **peripherals**.

- **Major Differences:**
  - **Peripheral devices** are electromagnetic/electro-mechanical, while the **CPU** is electronic, often requiring **signal conversion**.
  - **Peripheral data transfer rates** are slower, requiring **synchronization** with the CPU.
  - **Data formats** differ between peripherals and the CPU, requiring **format conversion**.
  - Each peripheral device has its own operating mode, which must be managed to avoid **disturbing** other devices.

Types of Peripheral Devices

- A peripheral is a device connected to a host computer that expands its capabilities but is not part of the core architecture.
- It can be partially or completely dependent on the host for operation.

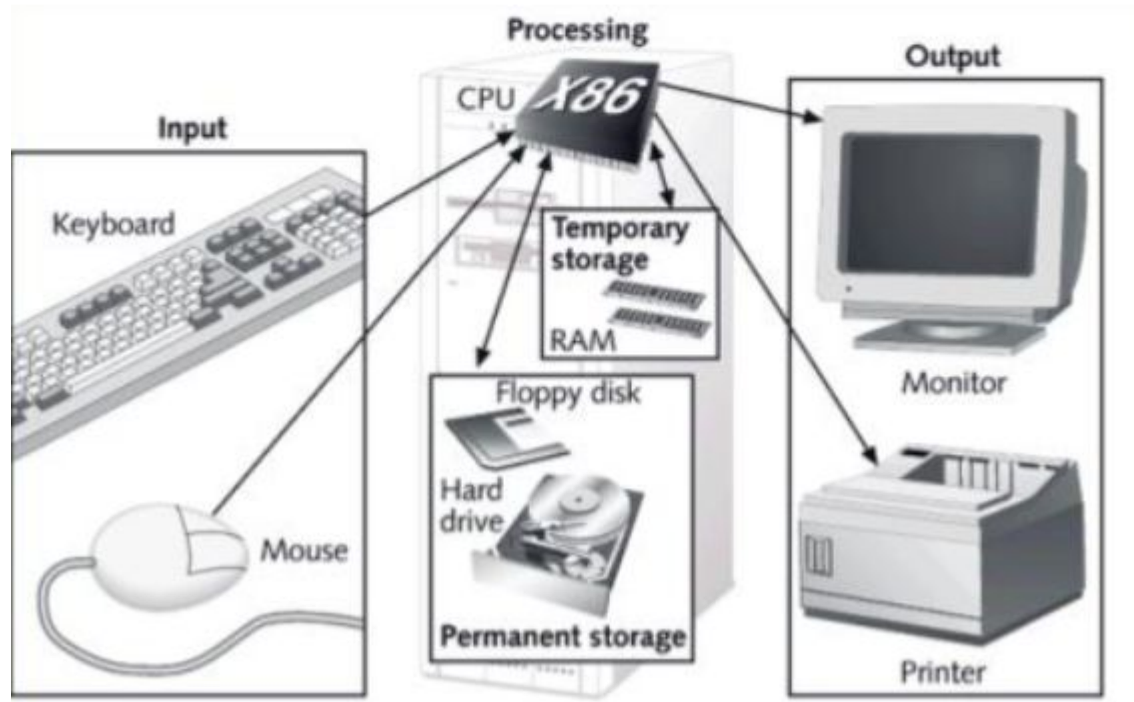
Types of Peripheral Devices:

- **Input Devices:** Devices that send data to the computer (e.g., keyboard, mouse).
- **Output Devices:** Devices that receive data from the computer (e.g., monitor, printer).
- **Input/Output Devices:** Devices that both send and receive data (e.g., external hard drives).

How does a computer work?

Functions of a Computer-

- Input
- Processing
- Output
- Storage



How does a computer work?

Input

- Data entered through devices like Keyboard, Mouse, Scanner.
- Can be analog (e.g., microphone) or digital (e.g., keyboard).

Processing

- Handled by the CPU (Central Processing Unit), the computer's brain.
- Performs calculations and controls tasks.
- RAM (Memory) provides temporary storage for fast access.

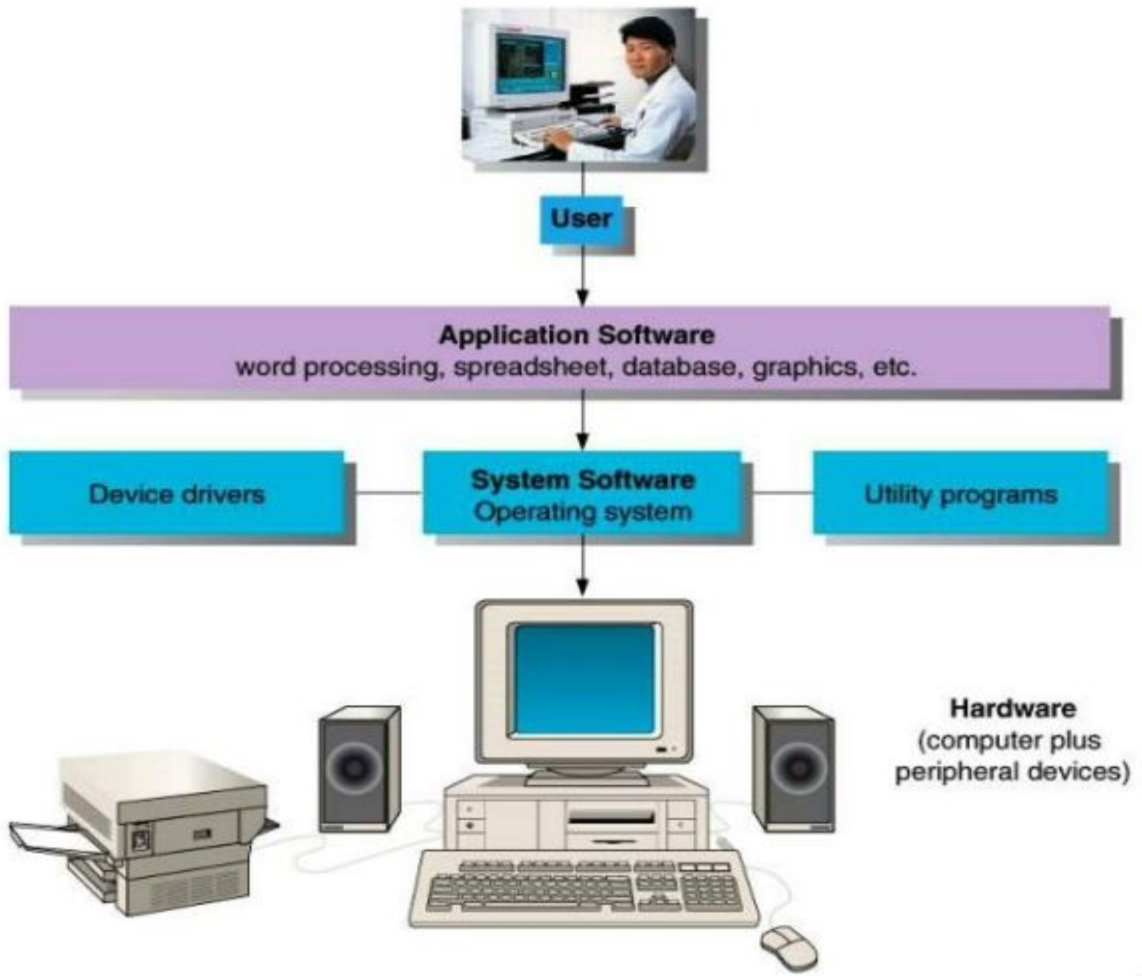
Output

- Results displayed on output devices such as Monitor, Printer, Speakers
- Forms of output: visual, printed, or audio.

Storage

- Processed information saved in HDD/SSD, Cloud Storage.
- Stored as files, images, or databases.

Software hierarchy of a computer system



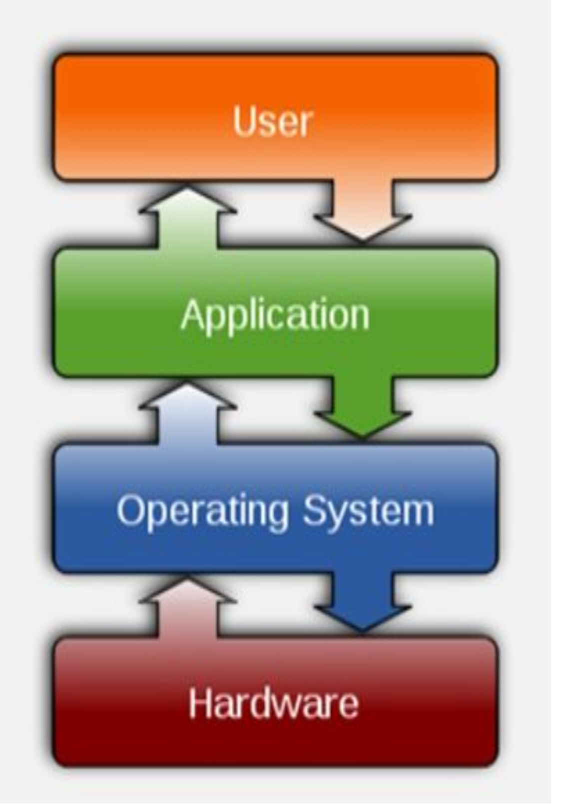
Software hierarchy of a computer system

- **User**
  - The person interacting with the computer, using various applications for tasks such as word processing, spreadsheets, or graphics.

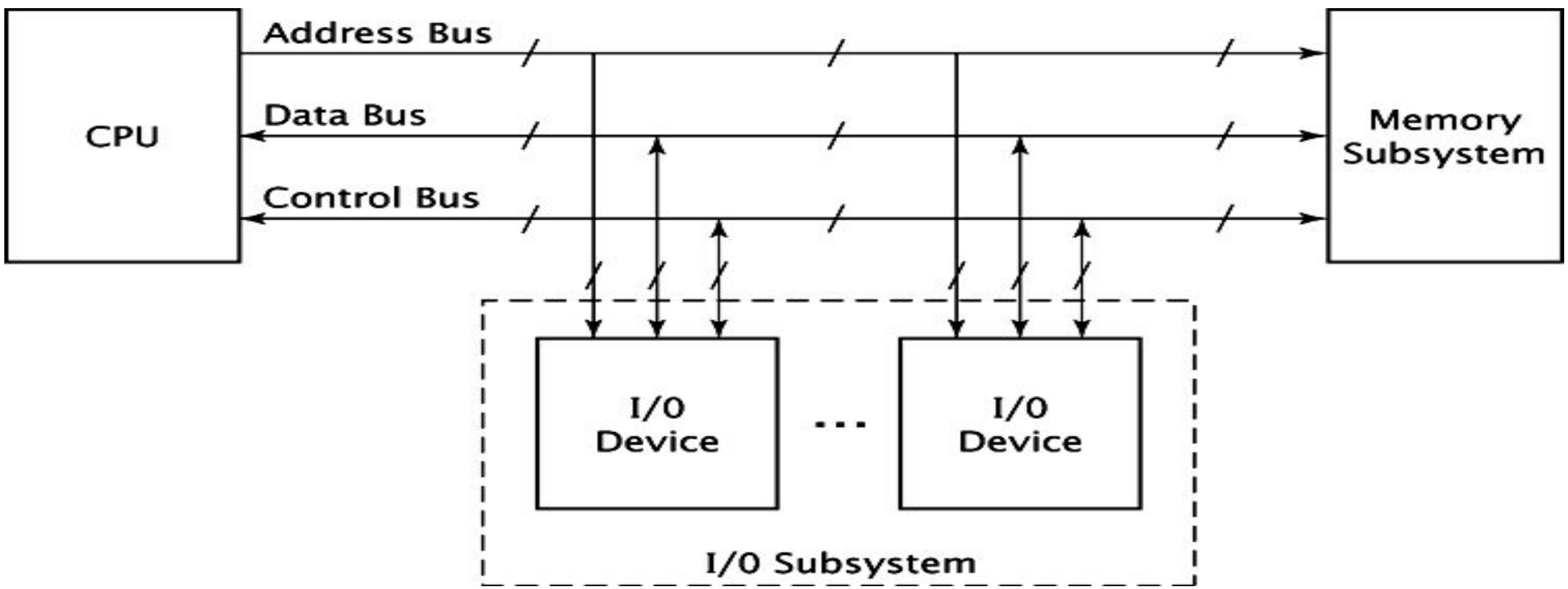
- **Application Software**
  - Programs used by the user to perform specific tasks, including:
  - Word Processing, Database Management, Graphics.

- **System Software**
  - Manages the overall functioning of the computer:
    - **Operating System (OS):** Controls hardware and provides services to application software.
    - **Device Drivers:** Facilitate communication between the OS and hardware devices.
    - **Utility Programs:** Perform maintenance tasks like virus scanning and disk cleanup.

- **Hardware**
  - The physical components of the computer (e.g., monitor, keyboard, CPU, speakers), along with peripheral devices like printers.



BUS Architecture



- A bus architecture is a system of pathways used for communication between components inside a computer. It transfers data, addresses, and control signals.

- **Key Components**:
  - **CPU**: The central processing unit that sends and receives data.
  - **Memory**: Where data is stored for quick access.
  - **I/O Devices**: Devices like keyboards, printers, etc., that interact with the CPU.

System Bus

- A set of wires that interconnects all the components (subsystems) of a computer system.

- **Data Flow:**
  - Source Component: Outputs data onto the bus.
  - Destination Component: Inputs data from the bus.

- **Bus Hierarchy:**
  - Address, Data, and Control Buses: Used to access memory and I/O controllers.
  - I/O Controller: Second set of buses connects the I/O controller to attached devices (e.g., peripherals).

- **Example:** PCI Bus: A common local bus example used in many computer systems.

.

Address Bus

- The CPU reads/writes data by addressing a **unique location**. The location is output onto the address bus, and the memory uses it to access the proper data.

- **Unique Addresses**:
  - Each I/O device (e.g., monitor, keyboard) has its own unique address.
  - CPU places the address on the address bus when accessing an I/O device, and the correct device acts accordingly.

- **Uni-Directional**:
  - Data flows **only from the CPU** to the address bus.
  - The CPU never reads data from the address bus.

Data Bus

- Transfers actual data between the CPU, memory, and I/O devices.

- **Data Transfer Process**:
  - **CPU Fetches**:
    - CPU outputs address on the **address bus**.
    - Memory outputs data onto the **data bus**.
    - CPU reads the data from the data bus.
  - **CPU Writes**:
    - CPU outputs address on the **address bus**.
    - CPU outputs data onto the **data bus**.
    - Memory reads the data and stores it.
- **Bidirectional**:

  - Data can flow to and from the CPU.

Control Bus

- The control bus is a collection of individual control signals, unlike address and data buses that transmit n-bit values.

- **Signals**:
  - **Read/Write Signals**: Whether data is being read into or written out of the CPU.
  - **Memory/I/O Signals**: Whether the CPU is accessing memory or an I/O device.
  - **Ready Signals**: Whether the I/O device or memory is ready for data transfer.
- **Unidirectional**:

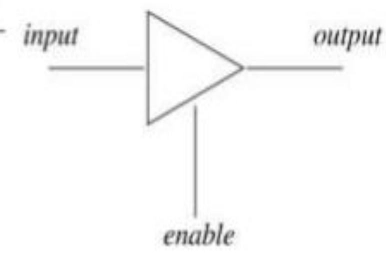
  - Control signals can flow to and from the CPU for synchronization between the CPU and other components like memory and I/O devices.

Tri- state Logic

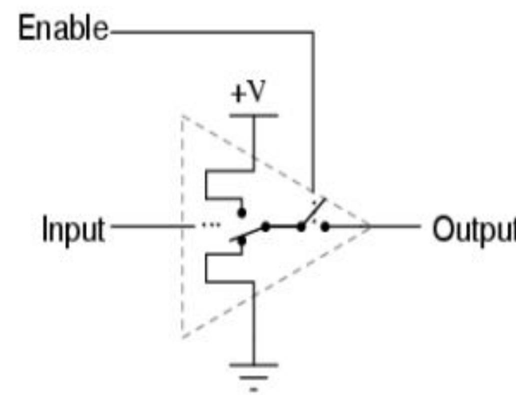
- Tri-state logic allows a bus to be shared by multiple devices but ensures that only one device can access the bus at any given time.

- **Three States**:
  - **Logic 0:** Represents a low voltage signal (e.g., 0V).
  - **Logic 1:** Represents a high voltage signal (e.g., 5V).
  - **High Impedance (Hi-Z):** This is the third state where the device is effectively disconnected from the circuit and does not influence the bus.

enable	*input*	*output*
0	X	Z
1	0	0
1	1	1



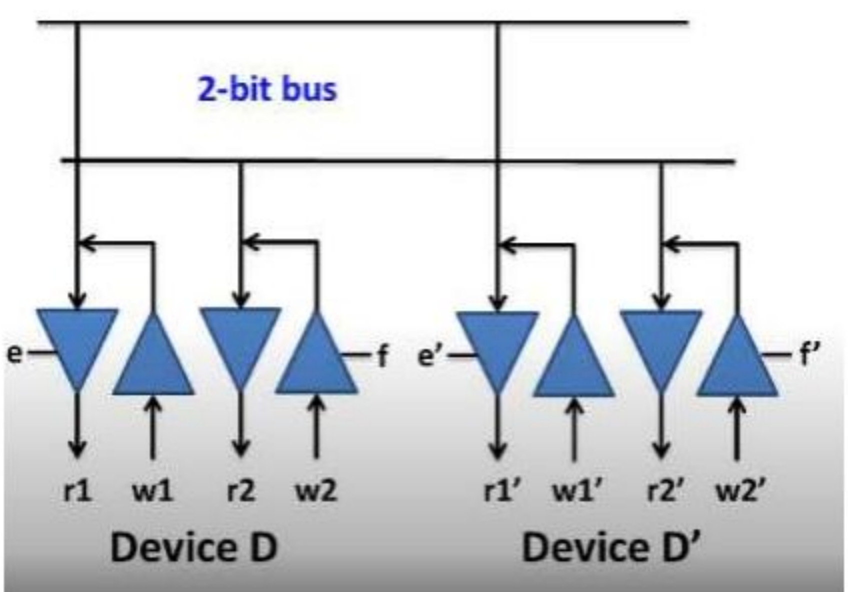
Tristate buffer gate



How Tri-Stating works

- When a device is in the Hi-Z state, it acts as though it is disconnected, preventing interference with other devices on the bus.

- This enables multiple devices to be connected to the same bus but ensures that only one device is active at a time.



- D is writing to D'
  - e=0, f=1, e'=1, f'=0
- D' is writing to D
  - e=1, f=0, e'=0, f'=1
- e=f=0 ➔ D is tri-stated
  - D is disconnected from bus

Advantages of Tri-State Bus

- To resolve conflict between multiple sources driving the shared bus line.
- To use the same port as both for input and output operation.

    Note: not all devices need to drive all bus lines.

- It higher the bus fan out.

Bus Interfacing

- **Bus Master:** The device that controls communication on the bus, determining who will speak and who will listen.
  - The **CPU** is typically the bus master.
  - **Advanced buses** can allow other devices to become bus masters when necessary.

- **Input/Output Operations:**
  - READ: Data moves into the bus master (input).
  - WRITE: Data moves out from the bus master (output).

- **Bus Lines:**
  - Each bus has several lines dedicated to three main functions:
    - Addressing
    - Data Transfer
    - Control

Bus Errors and Conflicts

- **Bus Conflict:** Occurs when multiple I/O devices try to use the bus simultaneously as output.

- **No Conflict:** If one device is used as output and others as input, no bus error occurs.

- **Causes of Bus Errors:**
  - Multiple devices using the bus as output at the same time.
  - Wait states not being properly maintained (mismatch between speeds).
  - An I/O device causing a bus pin to get stuck at 1 or 0.
  - Glitches: Short-lived faults that may cause transient errors.

Wait States & Glitches

- **Wait State:** A time-out period when the CPU or bus lies idle to account for components working at different speeds.
  - Necessary when the CPU is faster than the memory or other components.
  - Without proper wait states, synchronization issues can lead to bus errors.

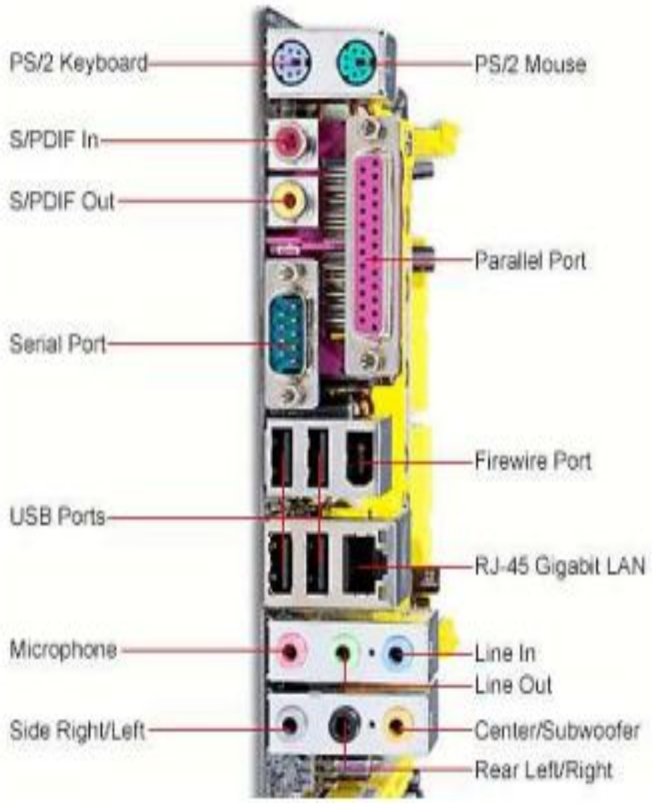
- **Glitches:** A temporary fault that is self-correcting but can cause errors like signal transitions at the wrong time.
  - Example: In I2C bus systems, glitches can cause incorrect start commands to be detected, disrupting communication.

Computer Ports

- Computer ports are connection points or interfaces with other peripheral devices.

- There are two main type of computer ports:
    - Parallel Ports
    - Serial Ports

- Others types are also there • PS2, SCSI etc.



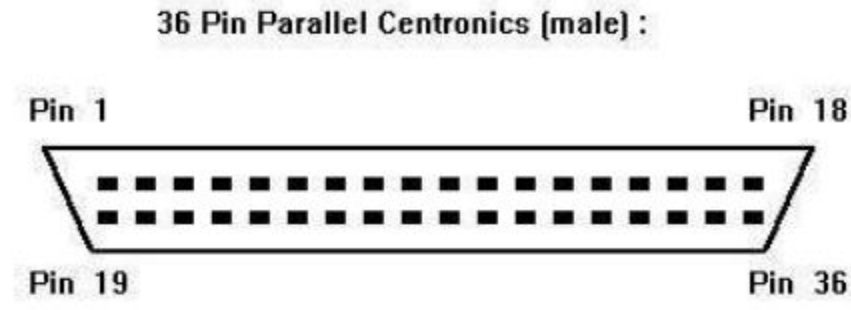
Parallel Port

- Used for data transfer between a computer and peripheral device through a 25 or 36 pin connector.

- Transfers multiple bits at a time in **parallel communication**.

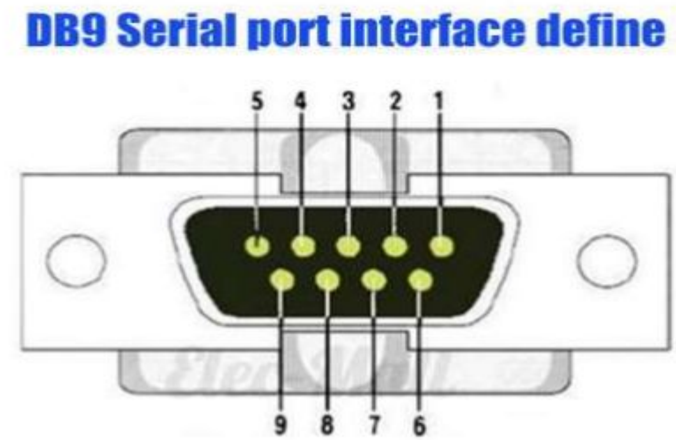
- With IEEE 1284 standard, it supports **bidirectional** data transfer.

- Speed ranges from 50 KBPS to 150 KBPS, and up to 2 MBPS with EPP and ECP modes.



Serial Port

- **Serial port** transfers data one bit at a time.
- Along with parallel port, it was commonly used for **data transfer** between computers and peripherals.
- Nowadays, it has been replaced by ports like USB, VGA, and Ethernet for more efficient communication.



VGA and HDMI Ports

- **VGA (Video Graphics Array)** and **HDMI (High-Definition Multimedia Interface)** are interface standards used to connect devices (laptops, DVD players) to displays (TVs, monitors, projectors).

- **VGA** carries only video signals.

- **HDMI** is the standard for newer devices like Blu-Ray players and LED TVs, carrying both video and audio.

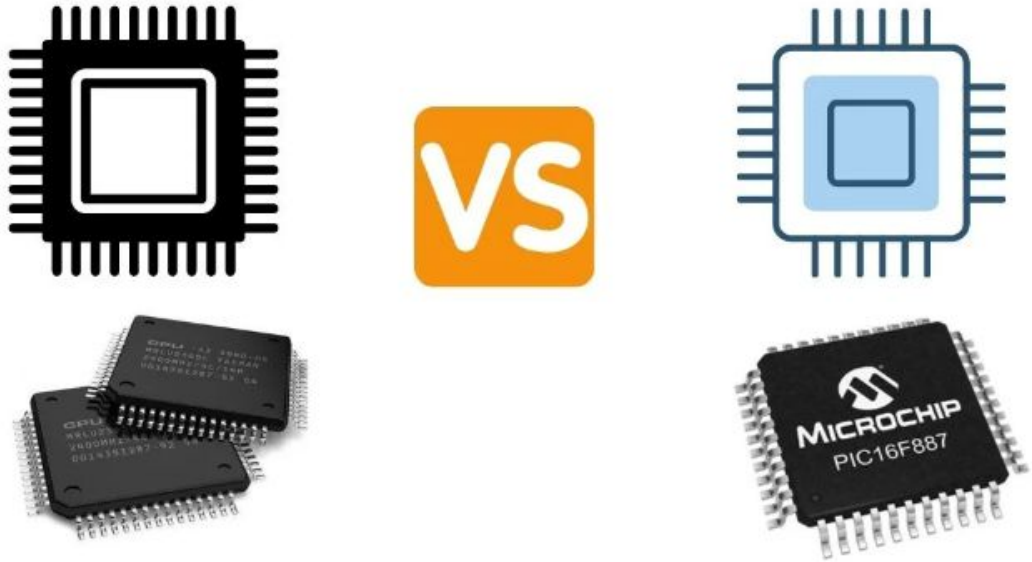
VGA Port

- **VGA** uses a 15-pin connector, usually marked in blue.
- It is an **analog video standard**, developed for older computer displays.
- VGA can only carry a **single video signal**.
- Maximum resolution: **2048×1536**, though at higher resolutions, image quality may degrade with ghosting and other issues.

HDMI Port

- **HDMI** is a **digital** interface that supports **both video and audio**.
- It uses a **19-pin connector** (HDMI A is the most common type).
- Supports high-definition resolutions without interference or signal loss.
- HDMI is more versatile, also carrying audio and, in newer versions, **internet**.
- HDMI includes encryption technology like **HDCP** for protecting data.

MICROPROCESSOR VS MICROCONTROLLER



- **Microcontroller:** A compact integrated circuit designed to govern a specific operation in an embedded system. It typically contains a processor, memory, and I/O peripherals on a single chip.

- **Microprocessor:** A general-purpose processing unit that only includes a CPU. It requires external components like memory, I/O ports, etc., to function.

Microprocessor VS Microcontroller

Architecture:

- Microcontroller: Combines a CPU, memory (RAM, ROM), and I/O peripherals all on a single chip.
- Microprocessor: Contains only the CPU; requires external components like RAM, ROM, and I/O interfaces for functioning.

Purpose:

- Microcontroller: Designed for specific control-oriented applications, often used in embedded systems.
- Microprocessor: Used in general-purpose computing systems like personal computers and servers for high-performance tasks.

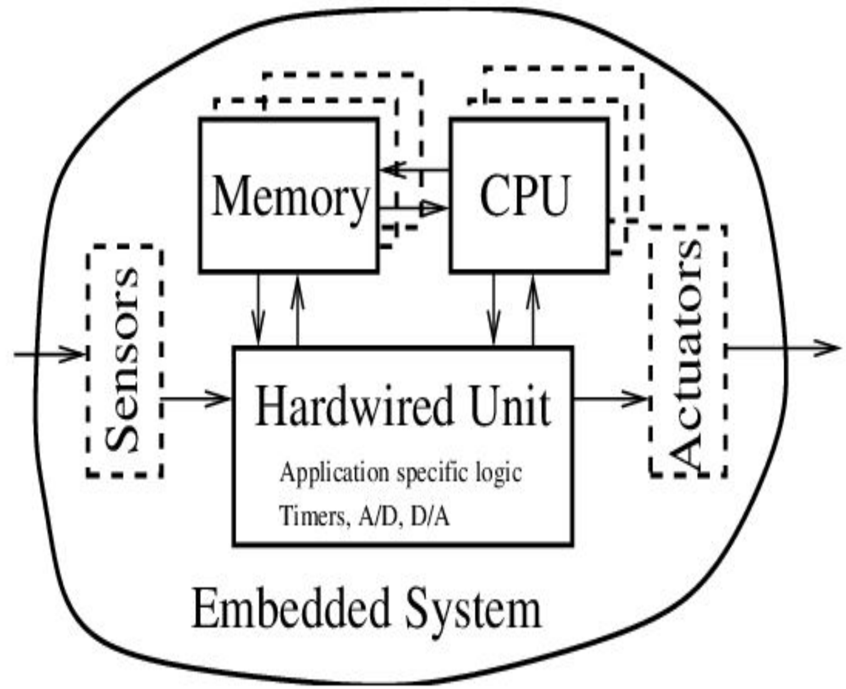
Power Consumption:

- Microcontroller: Consumes less power and is ideal for battery-powered or energy-efficient devices.
- Microprocessor: Consumes more power as it handles complex computations and tasks.

Embedded System

- A specialized computing system designed to perform dedicated tasks within a larger system.

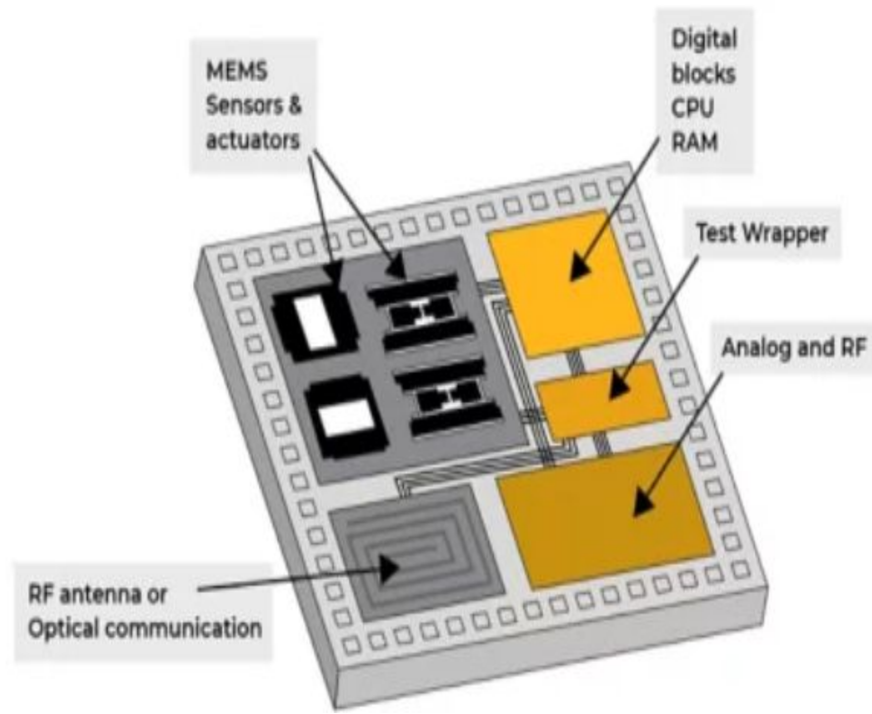
- **Components:**
  - CPU
  - Memory
  - Hardwired Unit: Performs application-specific tasks like timing, A/D or D/A conversions.
  - Sensors & Actuators: Sensors collect data from the environment. Actuators take actions based on processed data.



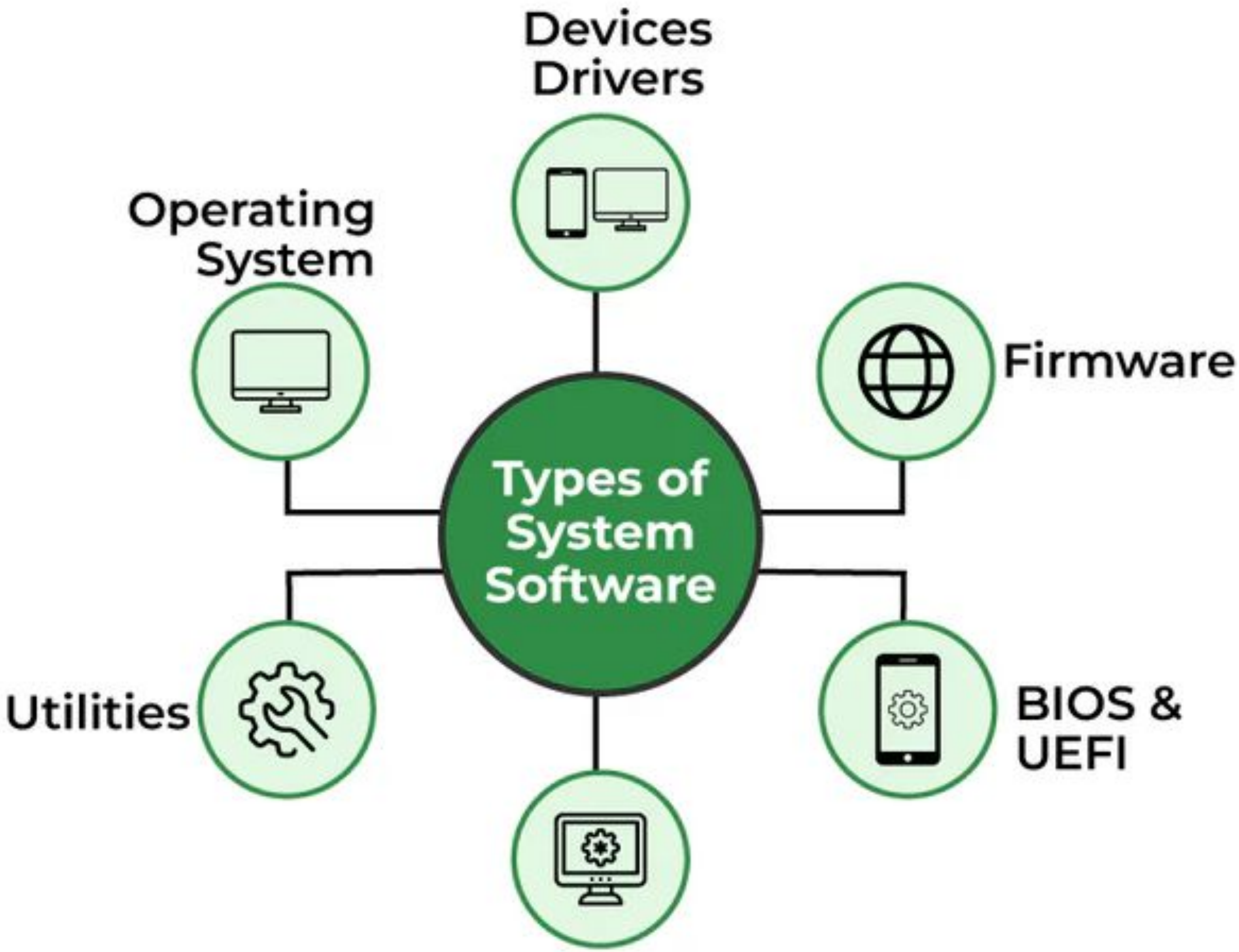
- **Key Features:**
  - Designed for specific tasks.
  - Real-time performance.
  - Compact and power-efficient.

Systems on a chip (SoC)

- A highly integrated circuit that combines all essential components of a computing system (CPU, GPU, memory, I/O, etc.) on a single chip.
- **Components Integrated:**
  - Sensors & Actuators:
  - Digital Blocks: Contains CPU and RAM for processing tasks.
  - Analog and RF Blocks: Handles communication and signal conversion for wireless and analog communication.
- **Key Features:**
  - Combines multiple functionalities (processing, communication, sensing) in one chip.
  - Power-efficient, compact, and cost-effective design for modern devices.
- Widely used in smartphones, IoT devices, smart wearables, and embedded systems.



System Softwares



System Softwares

- **BIOS (Basic Input/Output System)**
  - Initializes and tests hardware during startup (POST).
  - Loads the bootloader/OS from disk.
  - Located on motherboard ROM/Flash memory.

- **Operating System (OS)**
  - Manages hardware resources and runs applications.
  - Provides an interface between users and hardware.
  - Examples: Windows, Linux, macOS.

System Softwares

- **Driver**
  - Software that helps the OS communicate with hardware.
  - Enables devices like printers, GPUs, and keyboards to function.
  - Examples: Graphics card, network adapter drivers.

- **Firmware**
  - Low-level software stored in hardware (ROM).
  - Controls device functions (e.g., hard drives, keyboards).
  - Semi-permanent, rarely updated compared to drivers.

References

- "Computer Systems Organization & Architecture", John D. Carpinelli, ISBN: 0-201-61253-4
- "Operating Systems – A modern perspective", Garry Nutt, ISBN 0-8053-1295-1