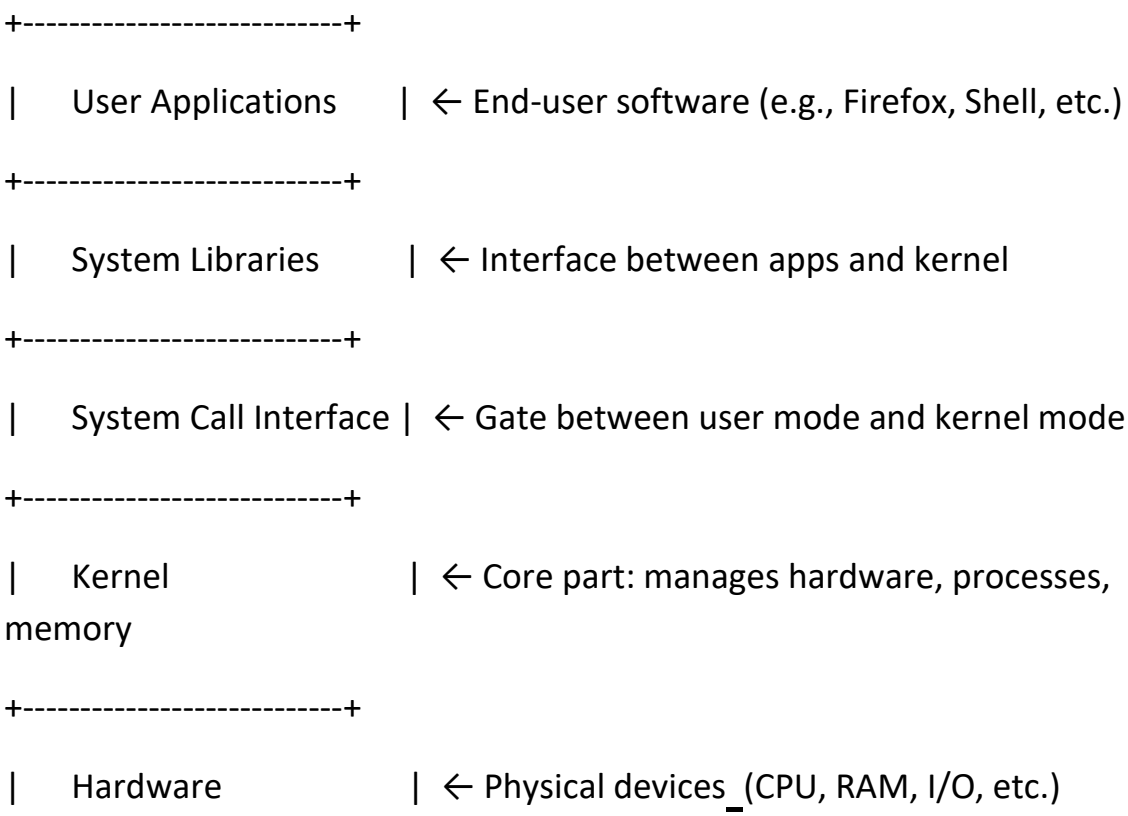
Session - 4

Linux Architecture

Overview of Linux Architecture

Linux architecture is structured into four main layers:



1. User Space

This is the **topmost layer**, where applications run.

Components:

- **User Applications**: Programs like text editors, media players, command-line tools.
- **Shells**: bash, sh, zsh — interfaces between user and OS.
- **Utilities**: BusyBox in embedded Linux provides lightweight commands.

🟢 *Runs in "User Mode" with limited privileges* — cannot directly access hardware.

2. System Libraries

These are **shared libraries** that provide **APIs for applications** to use OS functionalities.

Example Libraries:

- **glibc**: GNU C Library – provides standard C functions.
- **libm**: Math library.
- **libpthread**: For multi-threading.

🟢 *They make system calls to the kernel, abstracting hardware complexities from applications.*

3. System Call Interface (SCI)

Acts as a **bridge between user space and kernel space**.

Purpose:

- Allows user programs to **request services** (file access, memory allocation, etc.).
- Every action like read(), write(), or fork() goes through a system call.

🟢 *Implemented via software interrupts or processor traps.*

4. Kernel Space

This is the **core of Linux**, running in **privileged (kernel) mode**. It has full control over hardware.

The Kernel is divided into key components:

a) Process Management

- Schedules and manages processes and threads.
- Handles creation, execution, and termination.

b) Memory Management

- Manages RAM.
- Virtual memory, swapping, memory allocation, page tables.

- ### c) Device Drivers
- Interfaces for communicating with hardware.
- Each hardware has a driver (keyboard, USB, LCD, etc.).

- ### d) File System
- Supports various file systems (ext4, FAT, NFS, etc.).
- Provides file access, permissions, I/O buffering.

- ### e) Network Stack
- ımplements networking protocols (TCP/IP).
- Manages sockets, routing, firewall (iptables).

- ### f) Inter-Process Communication (IPC)
- Semaphores, shared memory, message queues, pipes.
- 🟢 *The kernel is **monolithic** (all parts in one large binary), but modular (supports loadable modules).*

5. Hardware Layer

- This is the actual **physical hardware** like:
- CPU (ARM, x86, etc.)
- RAM
- I/O devices (sensors, USB, serial)
- Network interface

Linux in Embedded Systems

In **Embedded Linux**, the architecture is the same, but:

- Applications are usually minimal and optimized.
- Filesystem may use BusyBox instead of full GNU tools.
- Kernel may be customized (real-time patches, fewer modules).
- Devices often boot from flash instead of a hard disk.