Lecture 5: Operating System Operations, Process Management, and Deadlock Handling

---

1. Explain the difference between hardware interrupts and software interrupts.

- **Hardware Interrupts:**
  Happen by external devices such as keyboards, disks, or network cards to signal events.
- **Software Interrupts (Traps or Exceptions):**
  Generated by software due to an error (like division by zero) or a system request

---

2. Define the Dual-Mode Operation in an Operating System.

The **Dual-Mode Operation** allows the CPU to operate in two modes to protect the system:

1. **User Mode:**
   - Normal applications run here.
   - Limited access to system resources.
   - Prevents users from harming the system.
2. **Kernel Mode:**
   - The operating system runs here.
   - Has full access to hardware and memory.
   - Can execute privileged instructions.

---

3. What is the purpose of the Mode Bit?

The **Mode Bit** is a special hardware signal that indicates the current mode of the CPU:

- `1` → **User Mode**
- `0` → **Kernel Mode**

It helps the OS differentiate between user-level and system-level instructions..

---

4. What happens during a transition from User Mode to Kernel Mode?

The CPU switches to **Kernel Mode** when a program:

- Makes a **system call** (e.g., file access, printing).
- Encounters an **exception or interrupt**.

The OS takes control to execute the requested operation safely, then switches back to **User Mode** when done.

---

5. Define a Process. How does it differ from a Program?

- A **Program** is a **passive entity** — a file stored on disk containing code.
- A **Process** is an **active entity** — a program **in execution**, with its own CPU state, memory, and resources.

---

6. List the resources that a process may need to execute.

A process typically needs:

- **CPU time**
- **Memory space**
- **I/O devices**
- **Initialization data**

---

7. Differentiate between a single-threaded and a multi-threaded process.

Aspect	Single-threaded Process	Multi-threaded Process
Definition	One sequence of execution (one program counter).	Multiple sequences (threads) within the same process.
Performance	Executes one instruction at a time.	Executes multiple parts of the program concurrently.

---

9. Explain how the OS manages multiple processes running concurrently.

- The OS gives each process a **small time slice** of CPU.
- When time expires or the process waits for I/O, the OS switches to another process.
- This switching happens quickly (context switching), making it **appear as if all processes run simultaneously**.

---

10. What are the main activities of Process Management in an Operating System?

1. **Process Creation and Deletion:**
    - OS creates a new process when a user opens an app and deletes it when it finishes.
2. **Process Suspension and Resumption:**
    - The OS can pause a process and later resume it (e.g., pausing a video).
3. **Process Synchronization:**
    - Ensures multiple processes do not interfere with each other (e.g., printing jobs).
4. **Process Communication:**
    - Allows safe data exchange between processes (e.g., browser ↔ network service).
5. **Deadlock Handling:**
    - Prevents or resolves situations where processes block each other indefinitely.

---

11. Define Deadlock.

A **Deadlock** occurs when two or more processes are waiting resources held by each other.

---

12. Describe the four methods the OS uses to handle deadlocks.

1. **Deadlock Prevention:**
    - The system is designed to prevent circular waiting.
    - Ensures processes don’t hold one resource while waiting for another.
2. **Deadlock Avoidance:**
    - The OS checks system states before granting resources to avoid unsafe conditions.
3. **Deadlock Detection:**

- Allows deadlocks to occur but uses algorithms to detect them.
- The OS terminates or restarts processes to fix the issue.

4. **Deadlock Ignorance:**
   - The OS simply ignores deadlocks (common in small systems).
   - If one occurs, the system may need to restart.