Experiment No. 11 Date :

Roll Number:

Aim: Case study: Interrupts used in Windows X operating system

Theory:

In the Windows operating system (including Windows XP), interrupts play a crucial role in managing hardware events and system resource access. Here's a general overview of how interrupts are used in Windows XP:

1. Hardware Interrupts (IRQ)

- **Purpose**: These are signals sent by hardware devices to the CPU, requesting immediate attention to handle specific events (like a key press, mouse movement, or incoming network packet).
- **How It Works**: When a device sends an interrupt request (IRQ), the processor temporarily stops its current execution to service the interrupt. Each hardware device is assigned a specific IRQ number (e.g., keyboard IRQ 1, system timer IRQ 0).
- **Interrupt Handling**: The CPU executes an **Interrupt Service Routine (ISR)**, which is a piece of code responsible for handling the interrupt. After handling, the CPU resumes the interrupted task.

2. Software Interrupts

- **Purpose**: Software interrupts allow programs to interact with hardware without direct access to the system's hardware. They are often used for system calls, where a program requests services from the OS (like file access or memory allocation).
- **Mechanism**: In Windows, software interrupts use a method known as **interrupt vectoring**, where the interrupt number is mapped to the corresponding ISR. For instance, the INT 21h interrupt in DOS is used for services like reading a key from the keyboard.

3. Interrupt Request Levels (IRQL)

- **Purpose**: Windows uses a system called **Interrupt Request Levels (IRQLs)** to prioritize different interrupt types. Each interrupt is assigned an IRQL based on its priority, and higher IRQL interrupts can preempt lower ones.
- **Examples**:
  - **Passive Level (IRQL 0)**: Normal thread execution.
  - **APC Level (IRQL 1)**: Asynchronous Procedure Calls.
  - **DISPATCH_LEVEL (IRQL 2)**: Deferred procedure calls (DPCs).
  - **Hardware Interrupt Level**: Above dispatch level, where the actual hardware interrupts occur.

4. Deferred Procedure Calls (DPCs)

- **Purpose**: DPCs allow Windows to defer less critical tasks to be handled later, reducing interrupt latency.

- **How It Works**: When a hardware interrupt occurs, Windows handles the time-critical part in the ISR and then queues the remaining work in a DPC, which is processed later at a lower priority.

5. Advanced Programmable Interrupt Controller (APIC)

- **Purpose**: In Windows XP and later, the system can use the **APIC** to handle more advanced interrupt management, especially in multi-core processors, to allow for more interrupt lines and better scalability.
- **How It Works**: APIC allows distributing interrupts across multiple CPUs, which improves performance in multi-processor systems.

6. Plug and Play Interrupt Handling

- **Purpose**: Windows uses **Plug and Play** (PnP) for managing hardware resources, including IRQs. PnP automatically assigns IRQs to new devices during system startup, resolving conflicts without user intervention.
- **IRQ Sharing**: Modern Windows systems, including Windows XP, allow multiple devices to share the same IRQ using advanced techniques like interrupt masking and **handling.**

7. Interrupt Handling in Drivers

- **Driver Responsibility**: Device drivers in Windows XP are responsible for managing hardware interrupts. Drivers register their ISR with the operating system and handle device-specific interrupts, ensuring proper communication between the OS and hardware.

Interrupts in Windows XP are foundational for responsive and efficient system operations, providing a way to manage hardware and software tasks effectively in real-time.

Conclusion: