19MCE304- DESIGN OF EMBEDDED SYSTEMS

Supervisor mode, exceptions, and traps are fundamental concepts in computer architecture and operating systems, crucial for managing hardware resources and ensuring system stability and security. Here's an overview of these concepts:

1. Supervisor Mode

Definition

- **Supervisor Mode**: Also known as kernel mode or privileged mode, is a mode of operation where the CPU has access to all system resources and can execute all instructions, including privileged ones that are not available in user mode.

Characteristics

- **Full Access**: The CPU can access hardware directly, modify control registers, and execute privileged instructions.
- **OS Kernel Operations**: Typically, the operating system kernel operates in supervisor mode.
- **Protection Mechanism**: Prevents user applications from performing operations that could disrupt system stability or security.

Transition

- **User Mode to Supervisor Mode**: Transition occurs through system calls, interrupts, or exceptions.
- **Supervisor Mode to User Mode**: Transition occurs when the operating system completes a privileged operation and returns control to a user application.

2. Exceptions

Definition

- **Exceptions**: Events that disrupt the normal flow of execution in a program, typically triggered by errors or unusual conditions.

Types of Exceptions

- **Hardware Exceptions**: Triggered by hardware faults, such as divide-by-zero errors, invalid memory access, or hardware malfunctions.
- **Software Exceptions**: Triggered by software errors or specific instructions, such as invalid opcode or breakpoint instructions.

Handling Exceptions

- **Exception Handler**: A special routine within the operating system that is invoked when an exception occurs. The handler determines the cause of the exception and takes appropriate action, such as terminating the offending process, logging an error, or attempting to recover.

3. Traps

Definition

- **Traps**: A specific type of exception that is intentionally triggered by a software instruction to request a service from the operating system.

Characteristics

- **Intentional**: Unlike hardware exceptions, traps are deliberately invoked by executing a special instruction, often called a "trap instruction."
- **System Calls**: Commonly used to implement system calls, where a user application requests services like file operations, memory allocation, or process control from the OS.

Handling Traps

- **Trap Handler**: Similar to an exception handler, but specifically designed to process system calls or software interrupts. The trap handler executes the requested service and returns control to the application.

4. Supervisor Mode, Exceptions, and Traps in Practice

Transition and Execution Flow

- **User Mode Execution**: User applications run in user mode with restricted access to system resources.
- **System Call (Trap Instruction)**: When an application needs a privileged service, it executes a trap instruction to request the service.
- **Trap Handler Invocation**: The trap instruction triggers a switch to supervisor mode, and the OS trap handler is invoked to process the request.
- **Return to User Mode**: After completing the service, the OS returns control to the application, switching back to user mode.
- **Exception Handling**: If an exception occurs (e.g., divide-by-zero), the CPU switches to supervisor mode and invokes the appropriate exception handler.

Example Code

- **Trap Instruction Example**: In many architectures, a specific instruction is used to trigger a trap.

  assembly
  Copy code
  int 0x80   ; x86 assembly instruction to trigger a system call in Linux

- **Handling a System Call in C**:

  c
  Copy code
  // User space code
  #include <unistd.h>

```
#include <sys/syscall.h>

int main() {
    syscall(SYS_write, STDOUT_FILENO, "Hello, World!\n", 14);
    return 0;
}
```