19CSB201 – Operating Systems

Unit 1

Computer System Organization, Architecture, Operation , Process Management – Memory Management, Storage Management ,Operating System – Process concept - Process scheduling

Operations on processes, Cooperating processes, Inter process communication, Threads Basic Models

Multi-threading Models- Threading issues

1.1 Computer System Organization, Architecture, Operation

Operating System

- A program that acts as an intermediary between a user of a computer and the computer hardware
- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer
- Operating system goals:
  - Execute user programs and make solving user problems **easier**
  - Make the computer system **convenient** to use
  - Use the computer hardware in an **efficient** manner

Computer System Structure

Computer system can be divided into four components

the hardware, the operating system the application programs, and a user

- Hardware – provides basic computing resources
  - CPU, memory, I/O devices, file storage space
- Operating system
  - Controls and coordinates use of hardware among various applications and users

• **The application programs—**such as word processors, spreadsheets, compilers, and web browsers—define the ways in which these resources are used to solve users' computing problems

•Users

People, machines, other computers

Computer Startup

- **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM, generally known as **firmware**
  - Initializes all aspects of system (CPU registers, device controllers, memory contents, etc.)
  - Loads operating system kernel and starts execution

Computer System Organization

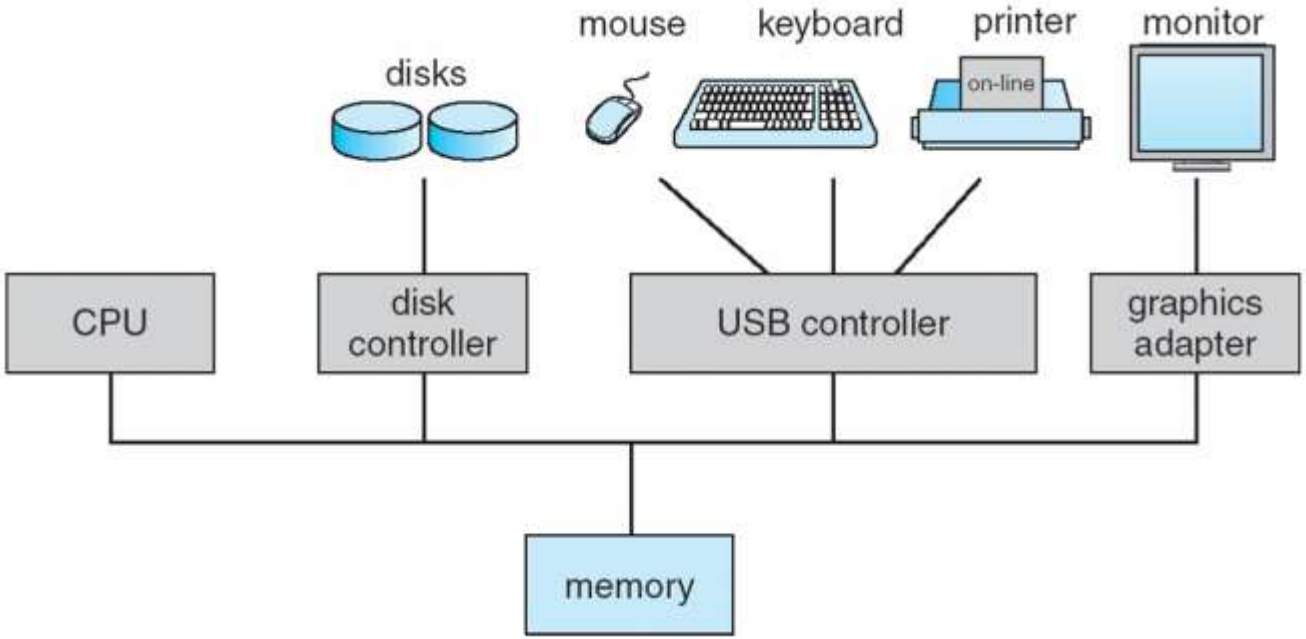
- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles
  - I/O devices and the CPU can execute concurrently
  - Each device controller is in charge of a particular device type
  - Each device controller has a local buffer
  - CPU moves data from/to main memory to/from local buffers ,I/O is from the device to local buffer of controller

Computer System Organization

- Typically, operating systems have a device driver for each device controller.
- This device driver understands the device controller and provides the rest of the operating system with a uniform interface to the device.
- The CPU and the device controllers can execute in parallel, competing for memory cycles.

Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory



Interrupts

- Device controller informs CPU that it has finished its operation by causing an interrupt
- Interrupt transfers control to the interrupt service routine generally, through the interrupt vector, which contains the addresses of all the service routines
- Interrupt architecture must save the address of the interrupted instruction
- Incoming interrupts are disabled while another interrupt is being processed to prevent a lost interrupts

- When the CPU is interrupted, it stops what it is doing and immediately transfers execution to a fixed location. The fixed location usually contains the starting address where the service routine for the interrupt is located. The interrupt service routine executes; on completion, the CPU resumes the interrupted computation

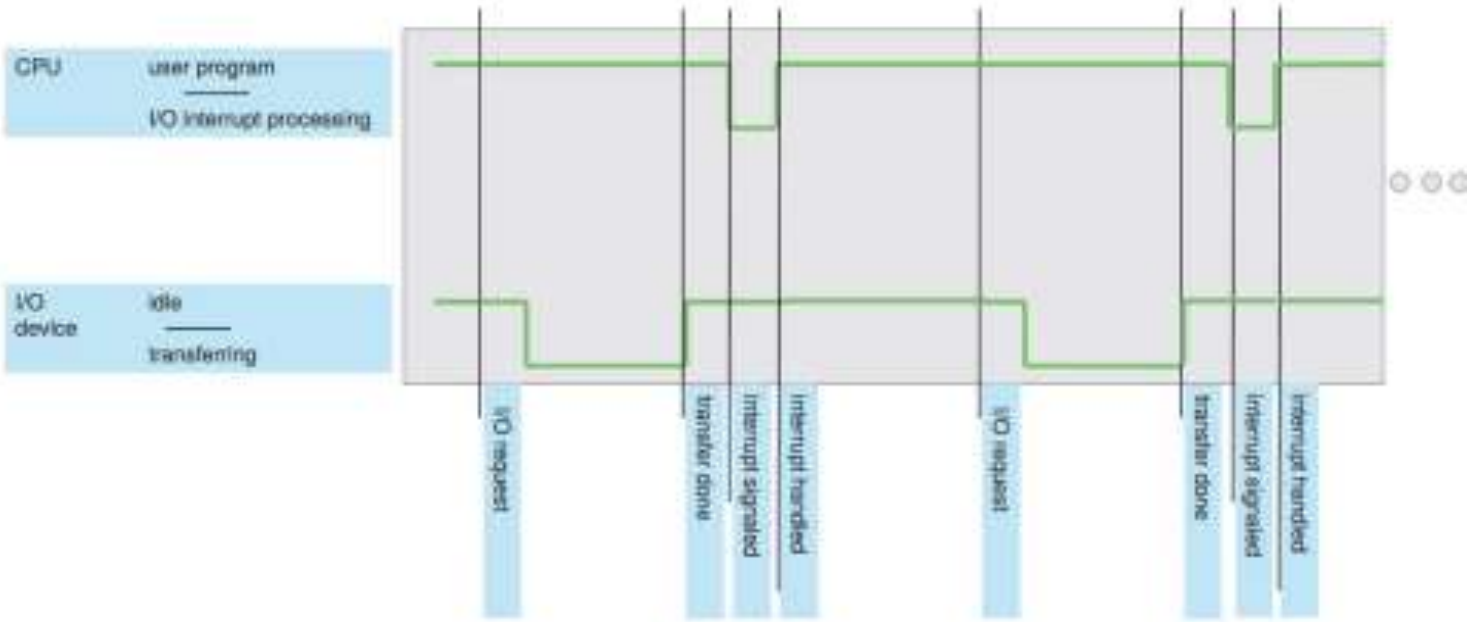


Figure 1.3 Interrupt timeline for a single program doing output.

- A trap is a software-generated interrupt caused either by an error or a user request
- An operating system is interrupt driven
- The operating system preserves the state of the CPU by storing registers and the program counter
- Determines which type of interrupt has occurred:
    - polling
    - vectored interrupt system
    - Separate segments of code determine what action should be taken for each type of interrupt

Storage Structure

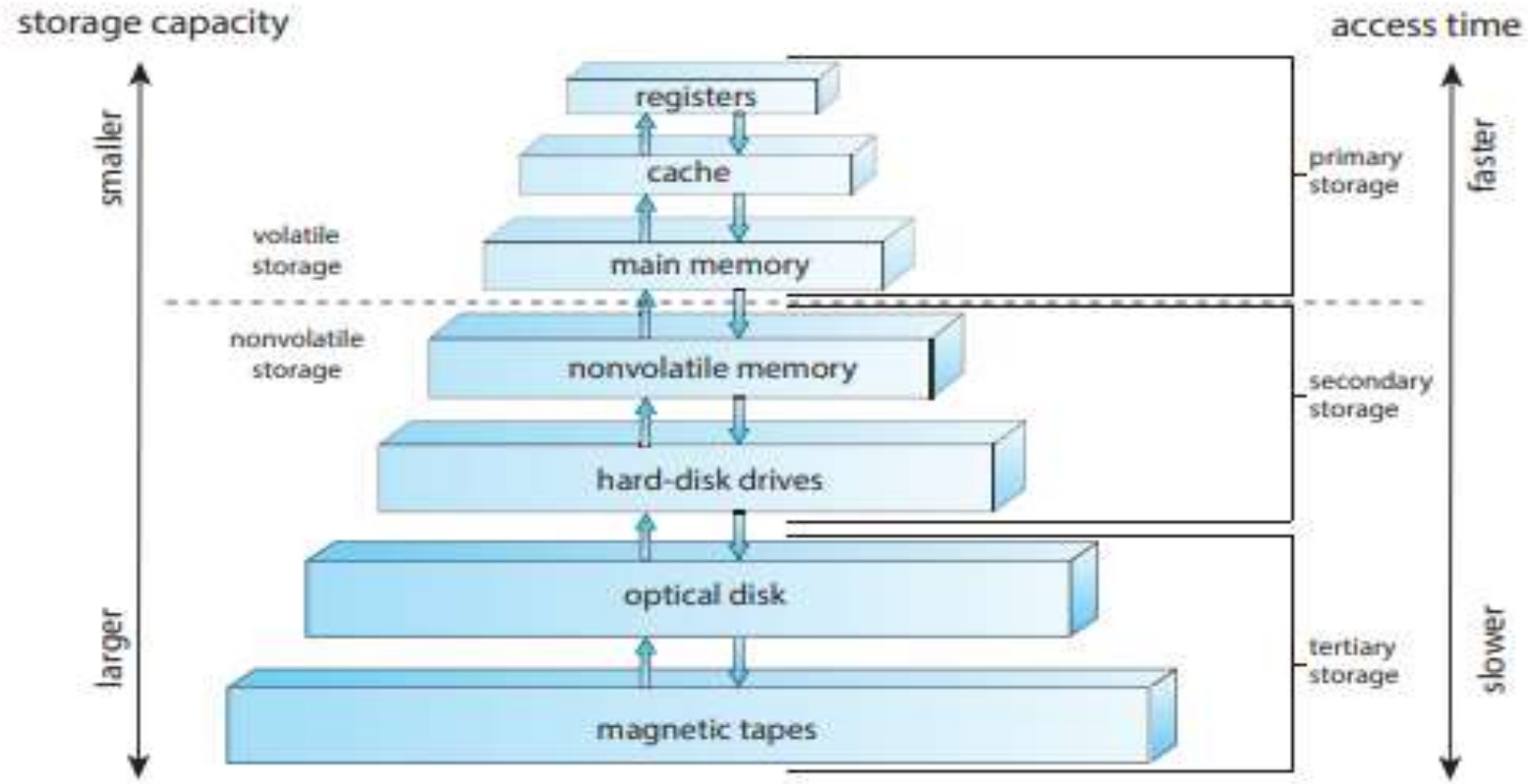


Figure 1.6 Storage-device hierarchy.

1. Main memory is usually too small to store all needed programs and data permanently.

2. Main memory, as mentioned, is volatile—it loses its contents when power is turned off or otherwise lost

The most common secondary-storage devices are hard-disk drives (HDDs) and nonvolatile memory (NVM) devices, which provide storage for both programs and data.

Secondary storage is also much slower than main memory

- tertiary storage. Each storage system provides the basic functions of storing a datum and holding that datum until it is retrieved at a later time.
- The main differences among the various storage systems lie in **speed, size, and volatility.**

Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
    - If it is, information used directly from the cache (fast)
    - If not, data copied to cache and used there
- Cache smaller than storage being cached
    - Cache management important design problem
    - Cache size and replacement policy
    - Disk surface is logically divided into tracks, which are subdivided into sectors
    - The disk controller determines the logical interaction between the device and the computer

I/O Structure

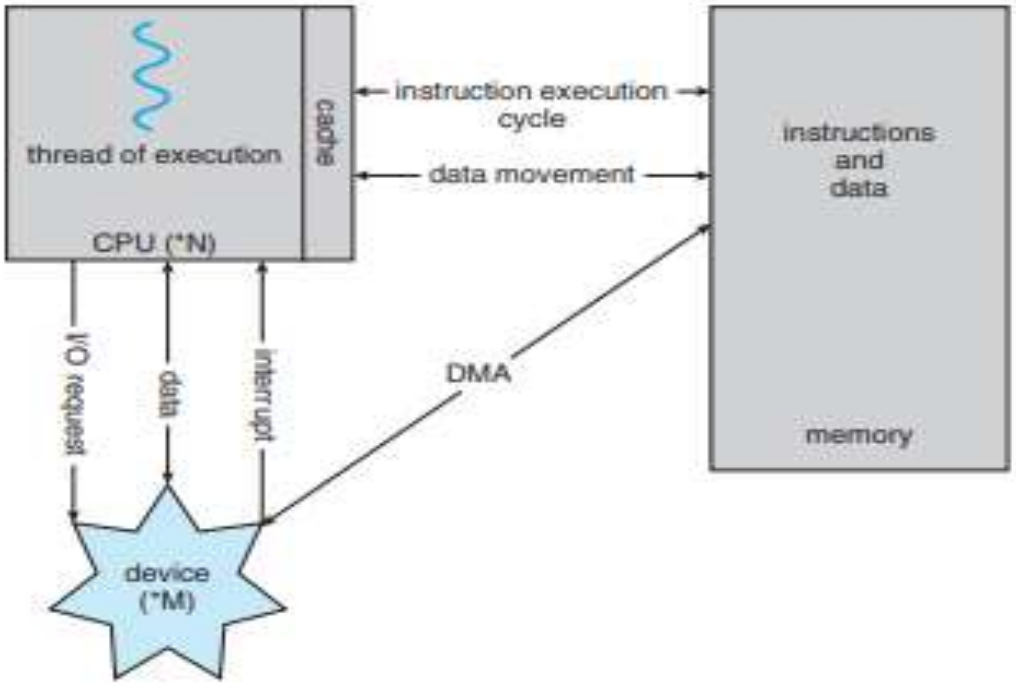


Figure 1.7 How a modern computer system works.

- After I/O starts, control returns to user program only upon I/O completion
    - Wait instruction idles the CPU until the next interrupt
    - Wait loop (contention for memory access)
    - At most one I/O request is outstanding at a time, no simultaneous I/O processing

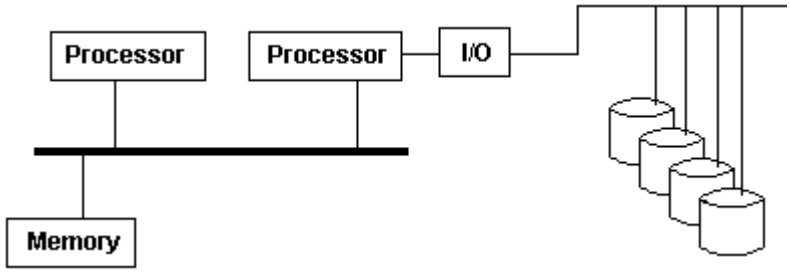
Computer-System Architecture

- Most systems use a single general-purpose processor
  - Most systems have special-purpose processors as well
- Multiprocessors systems (two or more processors in close communication, sharing bus and sometimes clock and memory) growing in use and importance
  - Also known as parallel systems, tightly-coupled systems
  - Advantages include
    1. Increased throughput
    2. Economy of scale
    3. Increased reliability – graceful degradation or fault tolerance

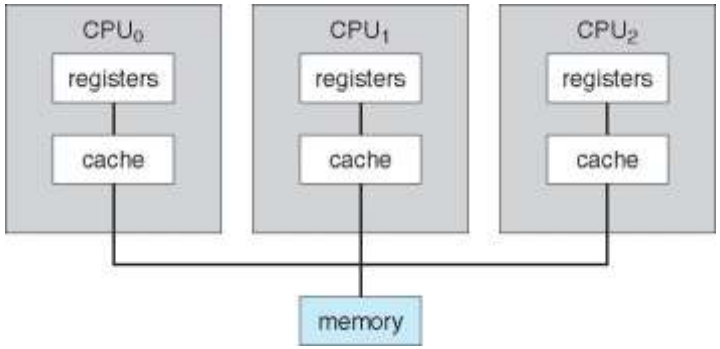
Multiprocessors systems

Key role – the scheduler

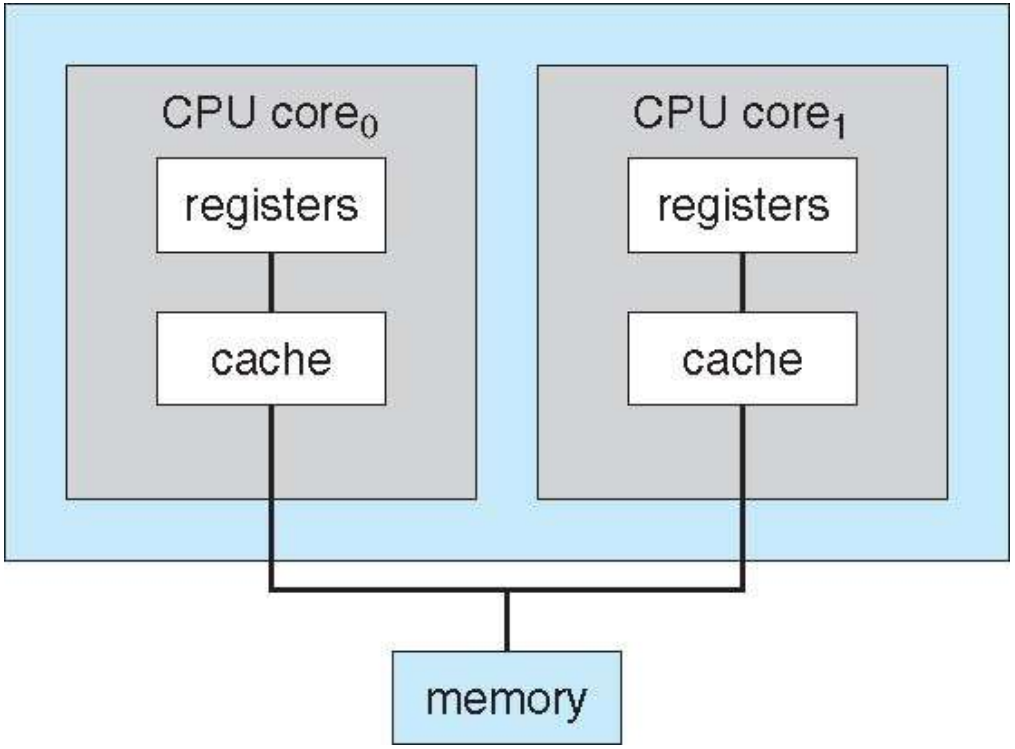
- Two types of Multiprocessing:
  1. Asymmetric Multiprocessing - assigns certain tasks only to certain processors. In particular, only one processor may be responsible for handling all of the interrupts in the system or perhaps even performing all of the I/O in the system
  2. Symmetric Multiprocessing - treats all of the processing elements in the system identically



multicore systems, in which multiple computing cores reside on a single chip. Multicore systems can be more efficient than multiple chips with single cores because on-chip communication is faster than between-chip communication



Symmetric multiprocessing architecture.



Clustered Systems

- Like multiprocessor systems, but **multiple systems working together**
  - Usually sharing storage via a storage-area network (SAN)
  - Provides a high-availability service which survives failures
    - Asymmetric clustering has one machine in hot-standby mode
    - Symmetric clustering has multiple nodes running applications, monitoring each other
  - Some clusters are for high-performance computing (HPC)
    - Applications must be written to use parallelization

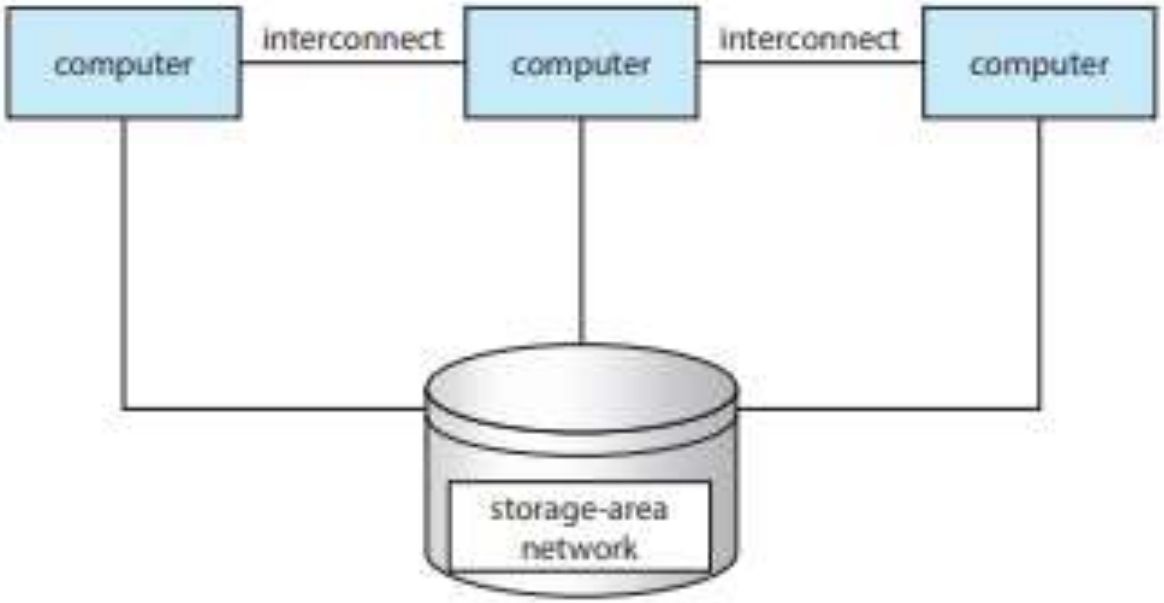


Figure 1.11 General structure of a clustered system.

Operating-System Operations

- computer to start running—when it is powered up or rebooted—it needs to have an initial program to run. This initial program, or bootstrap program, tends to be simple.
- Typically, it is stored within the computer hardware in firmware.
- It initializes all aspects of the system, from CPU registers to device controllers to memory contents. The bootstrap program must know how to load the operating system and how to
- Events are almost always signaled by the occurrence of an interrupt.
- Another form of interrupt is a trap (or an exception), which is a software-generated interrupt caused either by an error
- (for example, division by zero or invalid memory access) or by a specific request from a user program
- An operating-system service be performed by executing a special operation called a system call

Operating-System Operations
1. Multiprogramming and Multitasking

- Users typically want to run more than one program at a time.
- Multiprogramming increases CPU utilization
- In a multiprogrammed system, a program in execution is termed a **process**

Multitasking is a logical extension of multiprogramming.

In multitasking systems, the CPU executes multiple processes by switching among them,

but the switches occur frequently, providing the user with a fast response time

Timer: A timer can be set to interrupt the computer after a specified period.

The period may be fixed (for example, 1/60 second) or variable (for example, from 1 millisecond to 1 second).

A variable timer is generally implemented by a fixed-rate clock and a counter.

Operating-System Operations
2. Dual-Mode and Multimode Operation

- Dual-mode operation allows OS to protect itself and other system component
- User mode and kernel mode
- Mode bit provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as privileged, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user

Timer to prevent infinite loop / process hogging resources

A timer can be set to interrupt the computer after a specified period