UNIT-I

OVER VIEW OF OPERATING SYSTEM

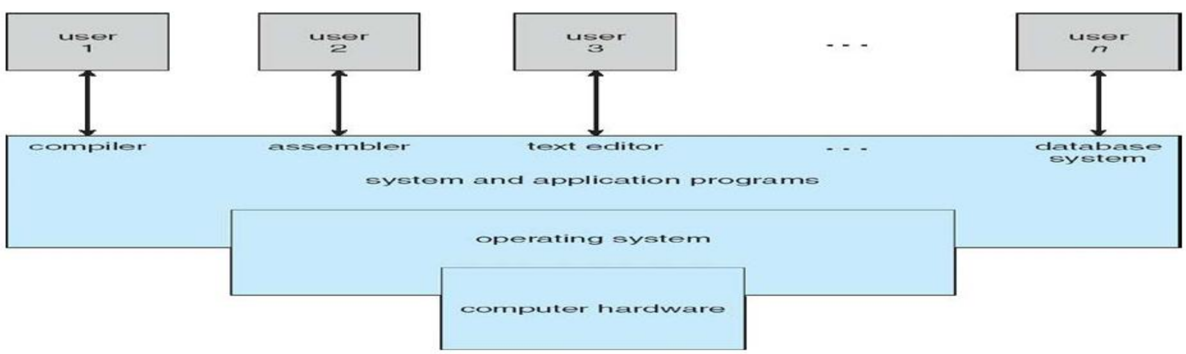
What is an Operating System?
A program that acts as an intermediary between a user of a computer and the computer hardware
Operating system goals:

- 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

- Hardware - provides basic computing resources
  - CPU, memory, I/O devices
- Operating system
  - Controls and coordinates use of hardware among various applications and users
- Application programs - define the ways in which the system resources are used to solve the computing problems of the users
  - Word processors, compilers, web browsers, database systems, video games
- Users
  - People, machines, other computers

Four Components of a Computer System



Operating System Definition

- OS is a **resource allocator**
- Manages all resources

Decides between conflicting
requests for efficient and fair

resource use

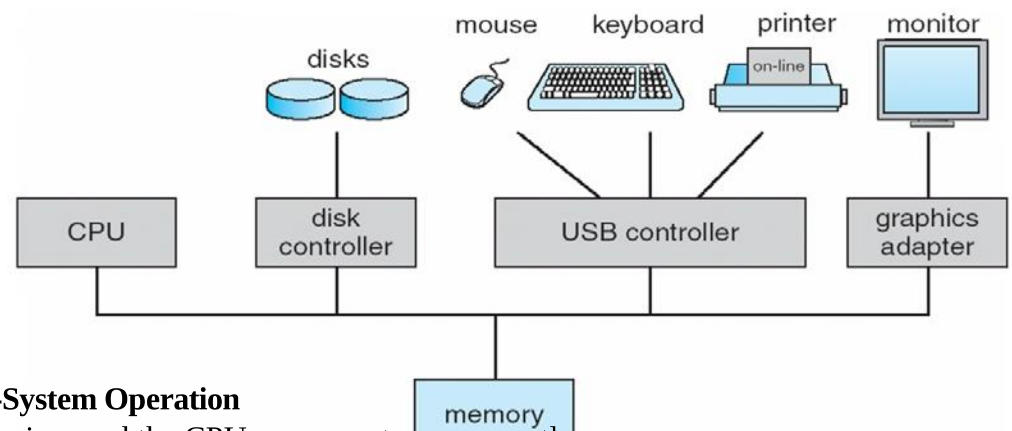
- OS is a **control program**
- Controls execution of programs to prevent errors and improper use of the computer
- No universally accepted definition
- Everything a vendor ships when you order an operating system" is good approximation
  But varies wildly
- "The one program running at all times on the computer" is the **kernel.** Everything else is either a system program (ships with the operating system) or an application program

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
- 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



Computer-System Operation

- 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
- Device controller informs CPU that it has finished its operation by causing An *interrupt*

Common Functions of Interrupts

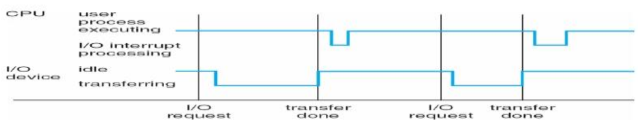
- Interrupt transfers control to the interrupt service routine generally, through the **interrupt vector**, which contains the addresses of all the service routines

- Interrupt architecture must save the address of the interrupted instruction
- Incoming interrupts are *disabled* while another interrupt is being processed to prevent a *lost interrupt*nA *trap* is a software-generated interrupt caused either by an error or a user request
- An operating system is **interrupt driven**

Interrupt Handling

- 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

Interrupt Timeline



I/O Structure

- 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
- After I/O starts, control returns to user program without waiting for I/O completion
- **System call** - request to the operating system to allow user to wait for I/O completion
- **Device-status table** contains entry for each I/O device indicating its type, address, and **state**
- Operating system indexes into I/O device table to determine device status and to modify table entry to include interrupt

Direct Memory Access Structure

- Used for high-speed I/O devices able to transmit information at close to memory speeds
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention
- Only one interrupt is generated per block, rather than the one interrupt per byte

Storage Structure

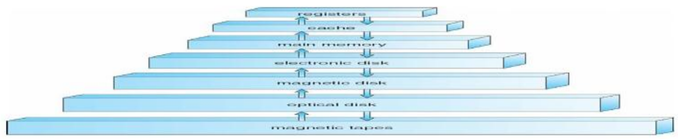
- Main memory - only large storage media that the CPU can access directly
- Secondary storage - extension of main memory that provides large nonvolatile storage capacity
- Magnetic disks - rigid metal or glass platters covered with magnetic recording material
- Disk surface is logically divided into **tracks**, which are subdivided into **sectors**

- The **disk controller** determines the logical interaction between the device and the computer

Storage Hierarchy

- Storage systems organized in hierarchy
- Speed
- Cost
- Volatility

Caching - copying information into faster storage system; main memory can be viewed as a last *cache* for secondary storage



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

Computer-System Architecture

- Most systems use a single general-purpose processor (PDAs through mainframes)
- Most systems have special-purpose processors as well
- Multiprocessors systems growing in use and importance •

Also known as parallel systems, tightly-coupled systems

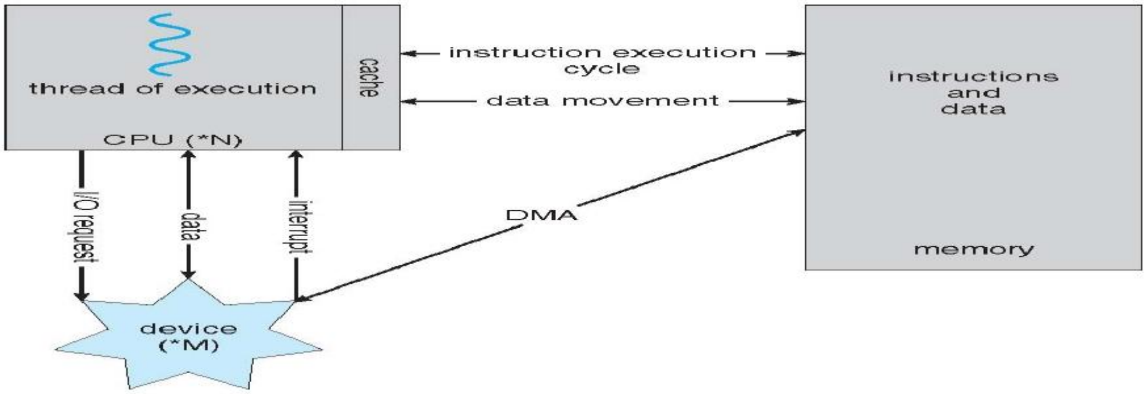
Advantages include

1.Increased throughput
2.Economy of scale
3.Increased reliability - graceful degradation or fault tolerance

Two types

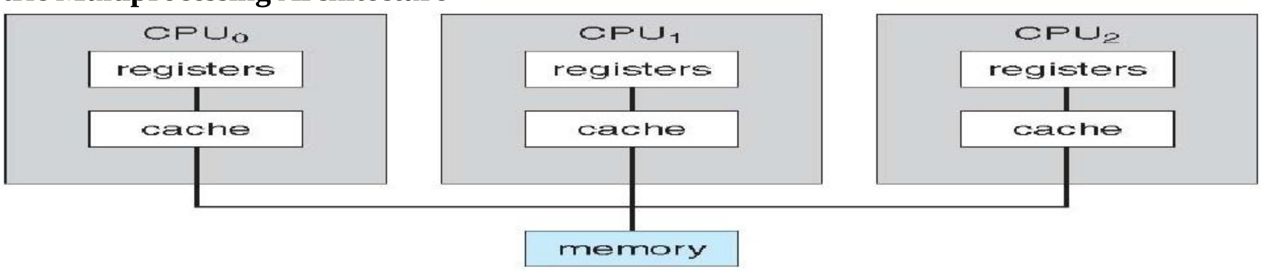
1.Asymmetric Multiprocessing
2.Symmetric Multiprocessing

•

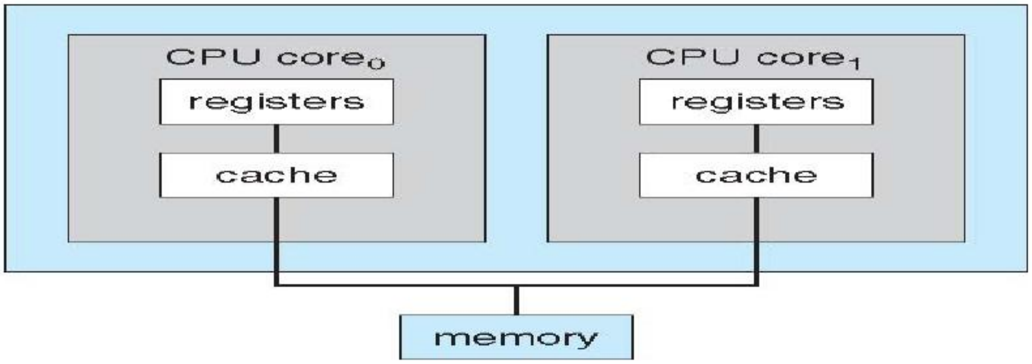


How a Modern Computer Works

Symmetric Multiprocessing Architecture



A Dual-Core Design



Clustered Systems

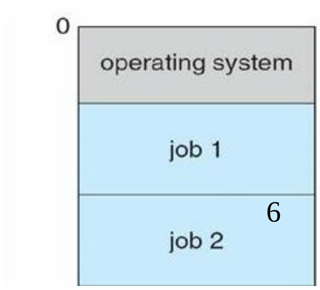
- Like multiprocessor systems, but multiple systems working together
- Usually sharing storage via a storage-area network (SAN)
- Provides a high-availability service which survives failures
  - Asymmetric clustering has one machine in hot-standby mode
  - Symmetric clustering has multiple nodes running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

Operating System Structure

- **Multiprogramming** needed for efficiency
- Single user cannot keep CPU and I/O devices busy at all times
- Multiprogramming organizes jobs (code and data) so CPU always has one to Execute
- A subset of total jobs in system is kept in memory
- One job selected and run via **job scheduling**
- When it has to wait (for I/O for example), OS switches to another job
- **Timesharing (multitasking)** is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
- **Response time** should be < 1 second
- Each user has at least one program executing in memory [**process**
- If several jobs ready to run at the same time [**CPU scheduling**
- If processes don't fit in memory, **swapping** moves them in and out to run

Virtual memory allows execution of processes not completely in memory

Memory Layout for Multiprogrammed System



Operating System Services

One set of operating-system services provides functions that are helpful to the user (Cont):lCommunications - Processes may exchange information, on the same computer or between computers over a network

 Communications may be via shared memory or through message passing (packets moved by the OS)

- Error detection - OS needs to be constantly aware of possible errors

 May occur in the CPU and memory hardware, in I/O devices, in user program

 For each type of error, OS should take the appropriate action to ensure correct and consistent computing

 Debugging facilities can greatly enhance the user's and programmer's abilities to efficiently use the system

- Another set of OS functions exists for ensuring the efficient operation of the system itself via resource sharing
- **Resource allocation -** When multiple users or multiple jobs running concurrently, resources must be allocated to each of them
- Many types of resources- Some (such as CPU cycles, main memory, and file storage) may have special allocation code, others (such as I/O devices) may have general request and release code
- **Accounting -** To keep track of which users use how much and what kinds of computer resources

- **Protection and security -** The owners of information stored in a multiuser or networked computer system may want to control use of that information, concurrent processes should not interfere with each other
- **Protection** involves ensuring that all access to system resources is controlled
- **Security** of the system from outsiders requires user authentication, extends to defending external I/O devices from invalid access attempts
- If a system is to be protected and secure, precautions must be instituted throughout it. A chain is only as strong as its weakest link.