

BILKENT UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING
CS342 OPERATING SYSTEMS

# LECTURE 1
# INTRODUCTION
# (CHAPTER 1 OF TEXTBOOK)

Dr. İbrahim Körpeoğlu
http://www.cs.bilkent.edu.tr/~korpe



## References

- *The slides here are adapted/modified from the textbook and its slides: Operating System Concepts, Silberschatz et al., 7th & 8th editions, Wiley.*

**REFERENCES**

- Operating System Concepts, 7th and 8th editions, Silberschatz et al. Wiley.
- Modern Operating Systems, Andrew S. Tanenbaum, 3rd edition, 2009.

## Chapter 1: Introduction Outline

- What Operating Systems Do
- Computer-System Organization and Archietcture
- Operating-System Structure and Operations
- Major Operating Systems Concepts/Components/Functionalities
  - Process Management
  - Memory Management
  - Storage Management
  - Protection and Security
- Computing Environments



## Objectives

- To provide a grand tour of the major operating systems components
- To provide coverage of basic computer system organization

## What is an operating system?

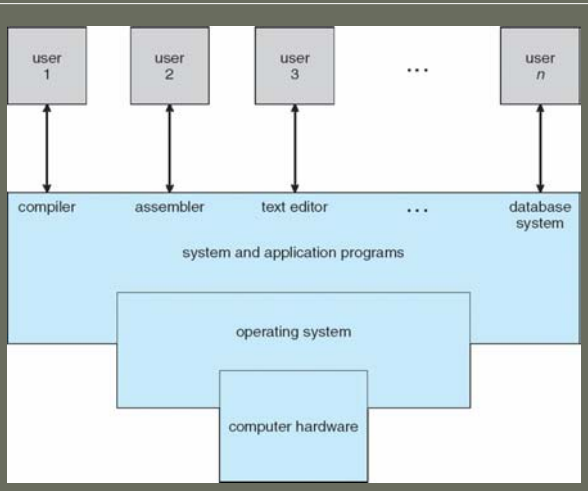
- A program that acts as an intermediary between a user of a computer and the computer hardware
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
  - Use the computer hardware in an efficient manner



## 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



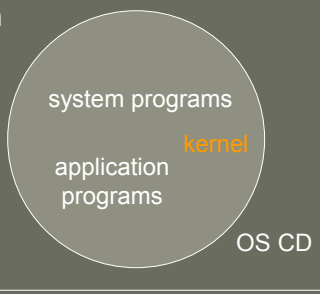



## Operating System Definition

- 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
  - I/O is accessed via the operating system

## Operating System Definition (cont.)

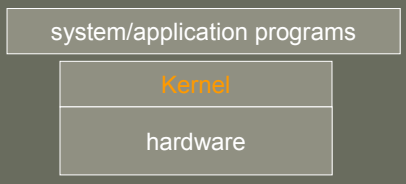
- No universally accepted definition
- "Everything a vendor ships when you order an operating system" is good approximation
  - But varies wildly
- "The one program running at all times on the computer" is the **kernel**. Everything else is either a system program (ships with the operating system) or an application program






## Computer Startup

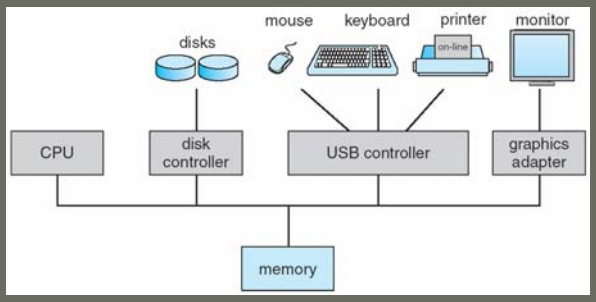
- **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM,
  - generally known as **firmware**
  - Initializes all aspects of the system
  - Loads operating system kernel and starts execution



# Computer System Organization

CS224 Knowledge

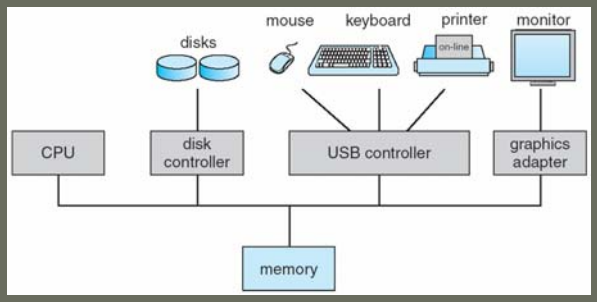
- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles






# 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*

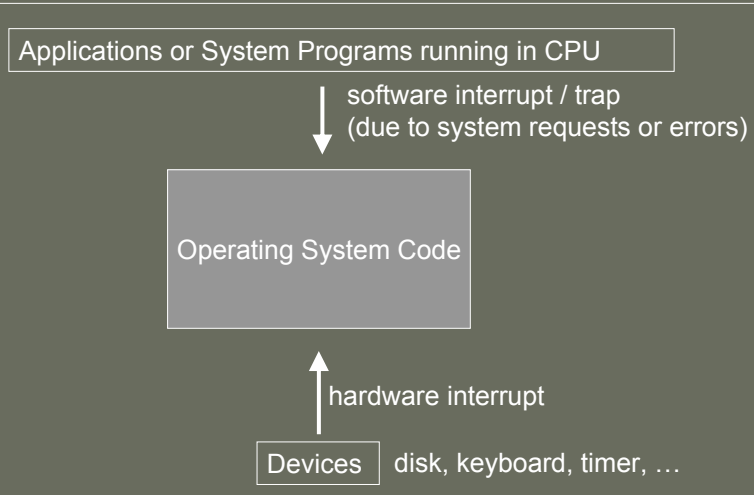


## Common Functions of Interrupts

- An 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*
- A *trap* is a software-generated interrupt caused either by an error or a user request
- An operating system is **interrupt driven**

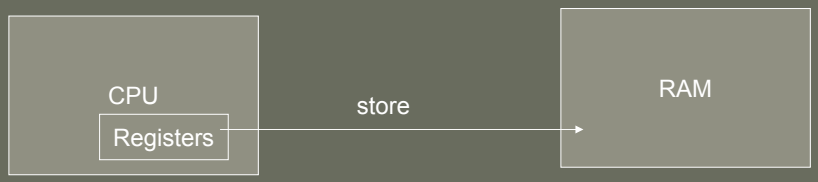


## Interrupt-Driven OS



# Interrupt Handling

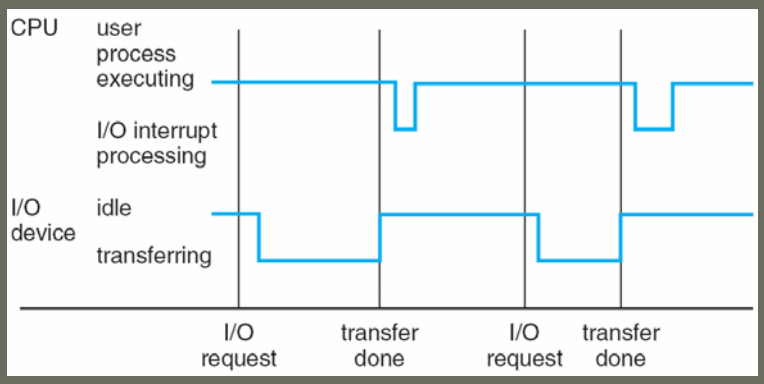
- The operating system preserves the state of the CPU by storing registers and the program counter
- Determines which type of interrupt has occurred:
  - **polling**
  - **vectored** interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt




[Application in CPU] – Interrupt – [Service Routine in CPU]

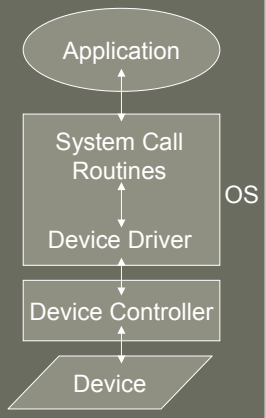


# Interrupt Timeline



# I/O Structure

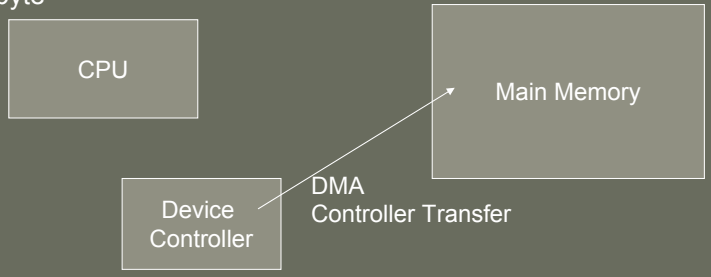
- Application programs can request I/O (read from a device or write to a device) via the help of OS
  - The request is done by calling a System Call (OS routine)
  - System call routine in OS performs the I/O via the help of device driver routines in OS.
  - OS maintains device status table: one entry per device. The entry keeps the state of the device, etc.
  - After issuing a system call, an application may wait for the call to finish (blocking call) or may continue to do something else (non-blocking call)



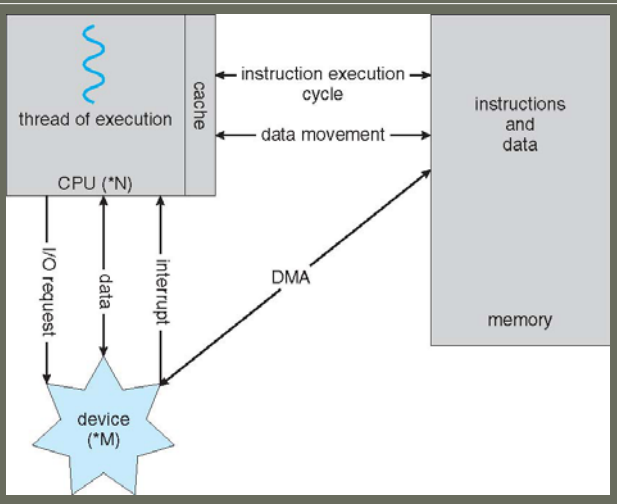



# Direct Memory Access Structure

- 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



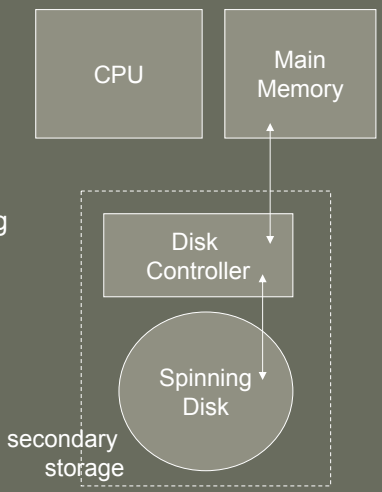
# How a Modern Computer Works





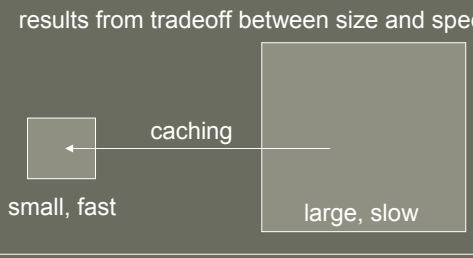

# Storage Structure

- 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



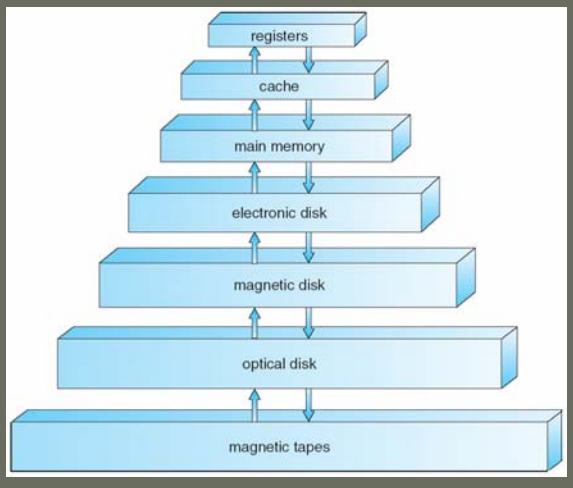
# Storage Hierarchy

- 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



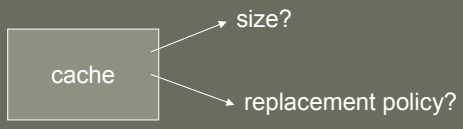



# Storage-Device Hierarchy



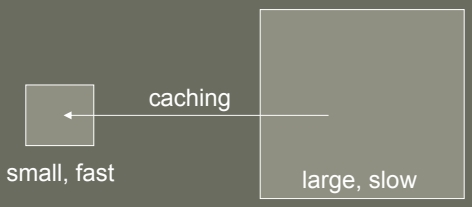
# Caching

- 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
- The *cache* is smaller than the *storage* being cached
  - Cache management important design problem
  - Cache size and replacement policy






# Caching



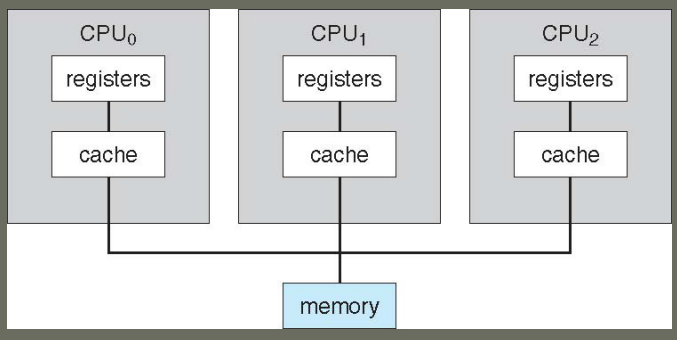

| | |
|---|---|
| Registers | Main Memory |
| Hardware cache L1, L2, etc | Main Memory |
| Main Memory | Hard Disk |
| Hard Disk | Tape |

## Computer System Architecture

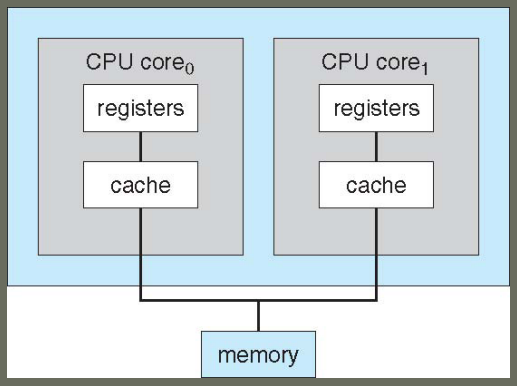
- Most systems use a single general-purpose processor (PDAs through mainframes)
  - Most systems have special-purpose processors as well
- Multiprocessor systems growing in use and importance
  - Also known as parallel systems, tightly-coupled systems
  - Advantages include
    1. Increased throughput
    2. Economy of scale (cheaper than using multiple computers)
    3. Increased reliability – graceful degradation or fault tolerance
  - Two types
    1. Asymmetric Multiprocessing
    2. Symmetric Multiprocessing



## Symmetric Multiprocessing Architecture



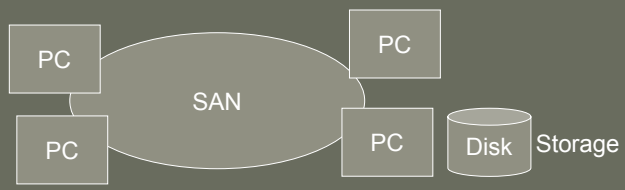
# A Dual Core Design





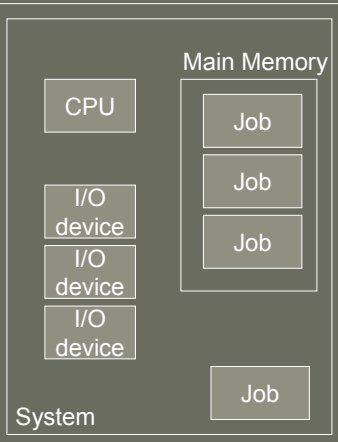

# Clustered Systems

- 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 Systems 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**
    - OS selects which job
  - When it has to wait (for I/O for example), OS switches to another job

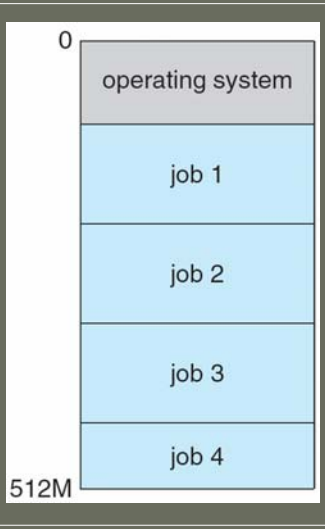





## Operating Systems Structure

- **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 Operations

- Interrupt driven by hardware
- Software error or request creates **exception** or **trap**
  - Division by zero, for example (exception)
  - request for operating system service (trap)
- Other process problems include:
  - processes modifying each other or the operating system
    - Handle by dual mode
  - infinite loop
    - Handle infinite loop by use of timer interrupt

# Operating System Operations

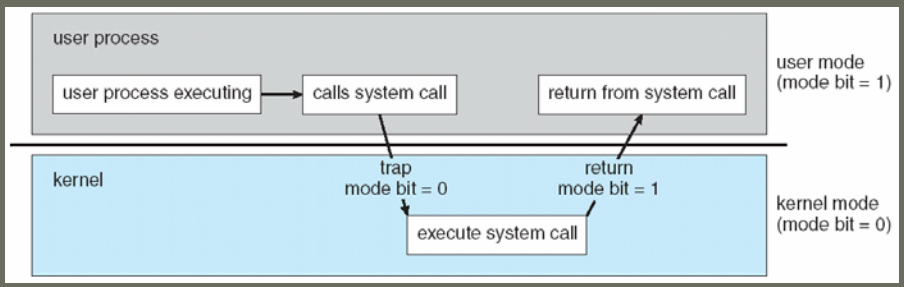
- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only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



# Operating System Operations

**Dual mode system operation**

**Transition from User to Kernel Mode and Vice Versa**

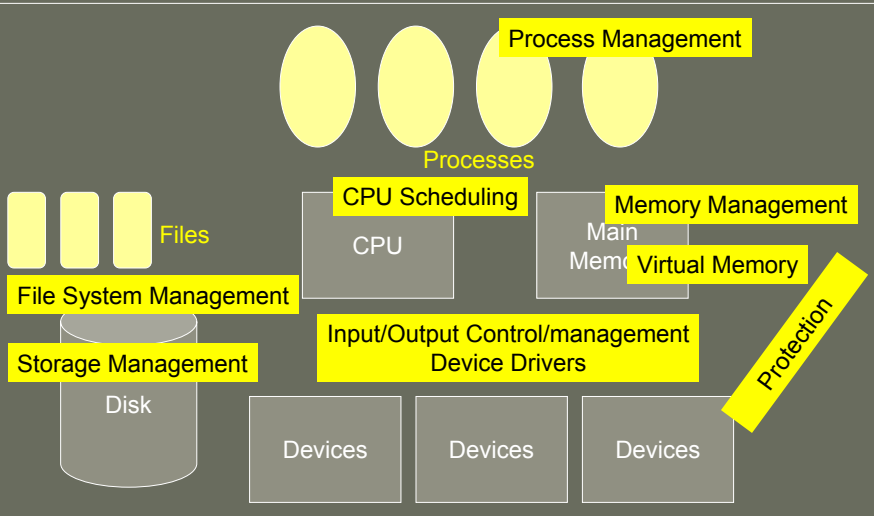


## Preventing a Process Hogging Resources

- Timer to prevent infinite loop / process hogging resources
  - 1) Set the timer device to interrupt after a while
    - Can be a fixed or variable time period
  - 2) CPU executes a program (a process)
  - 3) Timer device sends an interrupt after that period
  - 4) CPU starts executing timer handler: OS gains control
  - 5) OS can schedule the same process or other process
  - 6) OS sets the timer again before giving the CPU to the scheduled process



## Major OS Concepts/Components/Functions or



## Major OS Concepts/Components/Functions

- Process Management
  - Providing process abstraction and managing processes
- Memory management
  - Sharing memory among many processes
- Storage (disk) management
  - Providing file abstraction, managing files
  - Mapping files to disk blocks, disk scheduling
- I/O control and management
  - Device derivers, buffering, providing uniform access interface
- Protection and security
  - Controlled access to resources, preventing processes interfering with each other and OS



## Process Management

- A process is a program in execution. It is a unit of work within the system. Program is a *passive entity*, process is an *active entity*.
- Process executes instructions sequentially, one at a time, until completion
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
- Typically system has many processes (some user, some operating system) running concurrently on one or more CPUs
  - Concurrency by multiplexing the CPUs among the processes / threads
- Upon process termination, resources are released

## Process Management Activities

The operating system is responsible for the following activities in connection with process management:

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling



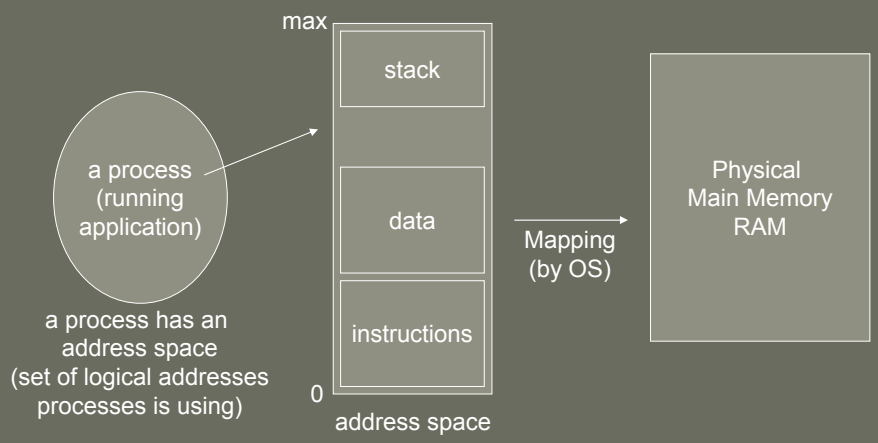
## Memory Management

- All data in memory before and after processing
- All instructions in memory in order to execute

- Memory management determines what is in memory and when

- Memory management activities
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts of a process) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

## Process Address Space






## Storage Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
    - Each medium is controlled by device (i.e., disk drive, tape drive)
      - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories; Primitives to manipulate files/dirs; Mapping files onto secondary storage

## Mass-Storage Management

- Mass Storage: disk (secondary); tapes, CDs, etc. (tertiary)
- Disk store data that does not fit into memory and to be stored for long time; Proper management is of central importance
- Entire speed of computer operation may depend on the disk subsystem and its algorithms
- OS activities
  - Free-space management; Storage allocation
  - Disk scheduling
- Some storage need not be fast
  - Tertiary storage includes optical storage, magnetic tape
  - Still must be managed



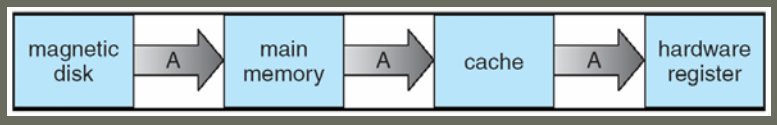
## Performance of Various Levels of Storage

- Movement between levels of storage hierarchy can be explicit or implicit

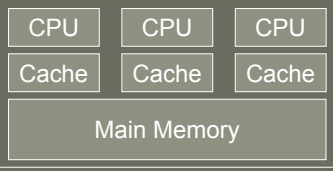
| Level | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Name | registers | cache | main memory | disk storage |
| Typical size | < 1 KB | > 16 MB | > 16 GB | > 100 GB |
| Implementation technology | custom memory with multiple ports, CMOS | on-chip or off-chip CMOS SRAM | CMOS DRAM | magnetic disk |
| Access time (ns) | 0.25 – 0.5 | 0.5 – 25 | 80 – 250 | 5,000.000 |
| Bandwidth (MB/sec) | 20,000 – 100,000 | 5000 – 10,000 | 1000 – 5000 | 20 – 150 |
| Managed by | compiler | hardware | operating system | operating system |
| Backed by | cache | main memory | disk | CD or tape |

## Migration of Integer A from Disk to Register

- Multitasking environments must be careful to use most recent value, no matter where it is stored in the storage hierarchy




- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache

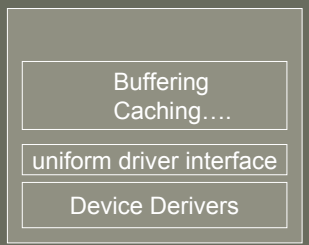





## Input/Output Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), ..
  - General device-driver interface
  - Drivers for specific hardware devices

I/Os sub-system of Kernel



## Protection and Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control

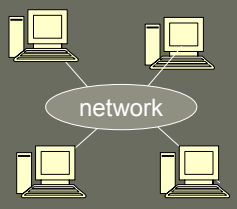


## Distributing Computing

- Earlier systems executed tasks on a single system



- Now we have systems interconnected (networked) together
  - Enabling distributed computing, resource sharing, etc.
- Operating systems have support now for networking multiple systems, distributing file storage, accessing remote resources, etc.
- Hence the computing environment is no longer a single system.
  - It can consist of many systems used in different ways

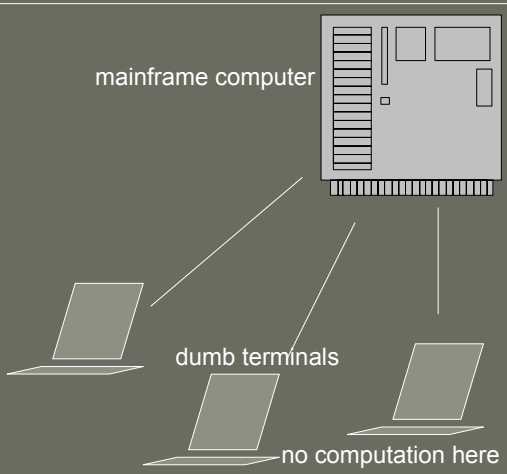


# Computing Environments

- Traditionally



a single system with a user

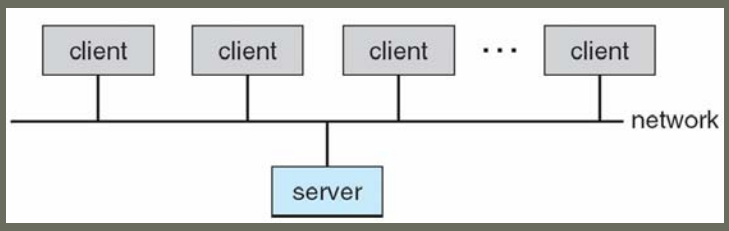



Computing and OS
in a single machine



# Computing Environments

- Client-Server Computing
  - Dumb terminals replaced by smart PCs
  - Many systems now **servers**, responding to requests generated by **clients**
    - **Compute-server** provides an interface to client to request services (i.e. database)
    - **File-server** provides interface for clients to store and retrieve files



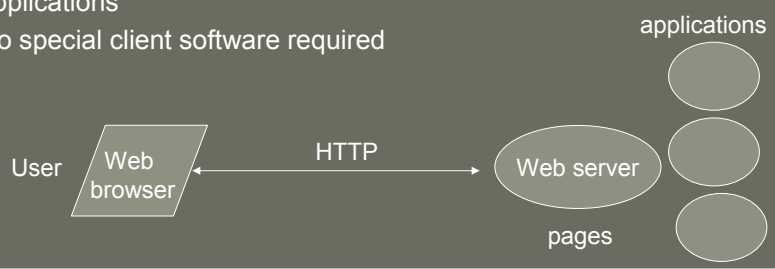
## Peer-To-Peer Computing

- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - Each may act as a client, a server or both
  - A node must join P2P network
    - Registers its service with central lookup service on network, or
    - Broadcast request for service and respond to requests for service via **resource discovery/lookup protocol**
  - Examples include *Napster* and *Gnutella*



## Web Based Computing

- Web has become ubiquitous
- More devices becoming networked to allow web access
- OSs run web servers and web clients
- Web based applications can be developed to run over web servers and clients.
  - Having a browser at the client is enough to run most of the applications
  - No special client software required



## Open-Source Operating Systems

- Operating systems made available in source-code format rather than just binary closed-source
- Counter to the copy protection movement
- Examples include
  - GNU/Linux,
  - BSD UNIX (FreeBSD, etc.)
  - Sun Solaris



## End of Chapter 1