Mansoura University
Faculty of Computers and Information
(Fall 2020)

# Operating Systems

## Introduction

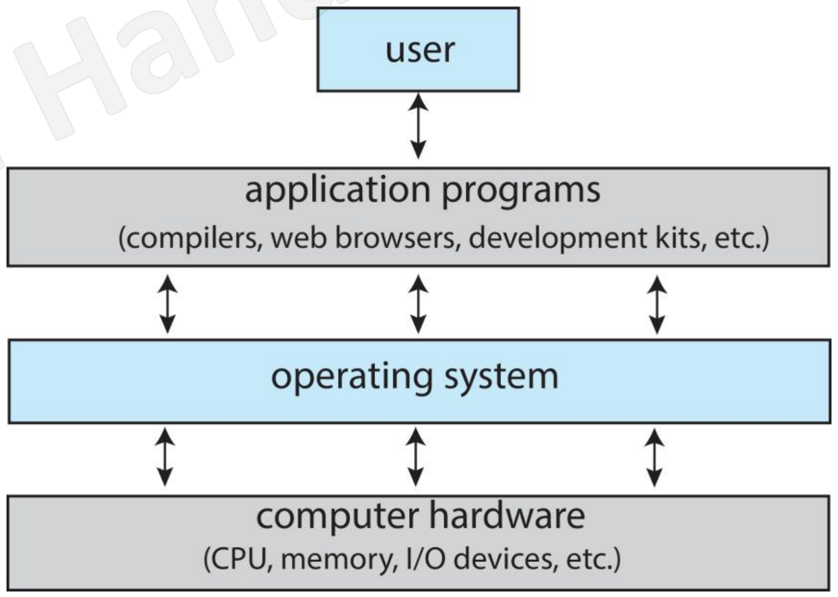
Dr. Mohamed Handosa
Computer Science Department
handosa@mans.edu.eg

## Topics

## Computer System Components

## Computer System Components

- Hardware
  - Provides basic computing resources.
- Application programs
  - Define the ways in which resources are used to solve user problems.
- Operating system (OS)
  - Controls the hardware and coordinates its use among the application programs.


user
application programs
(compilers, web browsers, development kits, etc.)
operating system
computer hardware
(CPU, memory, I/O devices, etc.)

## Operating System Views

- Can be viewed as a **resource allocator**
  - It acts as the manager of resources and decides how to allocate them to specific programs and users.
- Can be viewed as a **control program**
  - It controls the I/O devices and manages the execution of user programs to prevent errors and improper use of the computer.

## Kernel and System Programs

- Operating system = kernel + system programs
- The **Kernel** is the one program running at all times.
- **System programs** are associated with the operating system.
- **Application Programs** are not associated with the operating system.

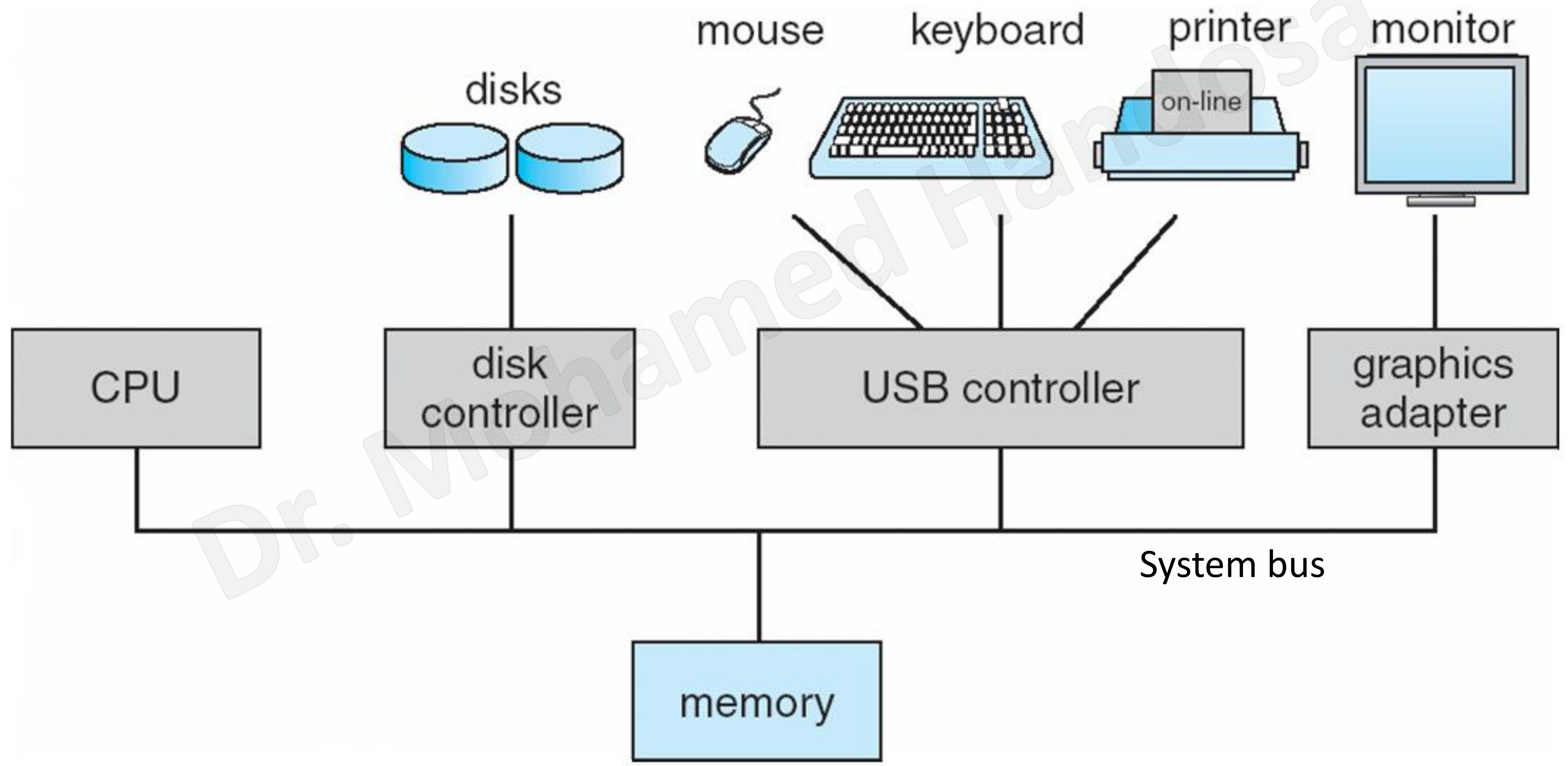
## Computer System Organization

Interrupts

Storage Structure

I/O Structure

## Computer System Organization



System bus

## Computer System Organization

- Each device controller controls a specific type of devices.
- The CPU and the device controllers can execute in parallel.
- The memory controller synchronizes access to the shared memory.

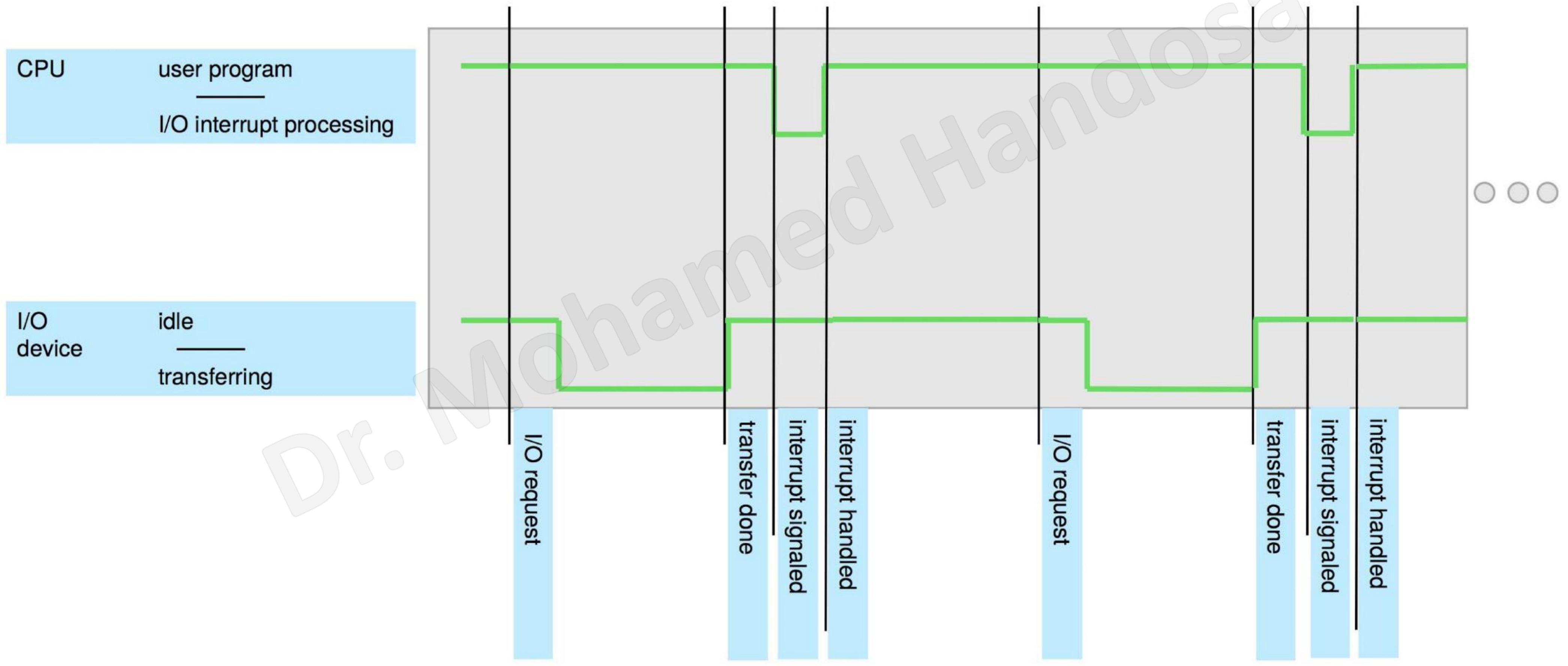
## Bootstrap Program

- A simple program stored in ROM or EEPROM.
- It is also known by the general term **firmware**.
- It runs when the computer is powered up or rebooted to:
  - Initialize all aspects of the system (e.g. CPU, memory).
  - Locate the kernel, load it into memory, and start its execution.

## Interrupts

- Once the system is fully booted, it waits for some event to occur.
- An **interrupt** is a signal that is generated when some event occurs.
  - A hardware-generated interrupt occurs by sending a signal to the CPU.
  - A software-generated interrupt (or **trap**) occurs by executing a **system call**.
- Each computer architecture has a predefined set of interrupts.
- Each interrupt has a corresponding service routine (or handler).
- The interrupt handler must be executed when the interrupt occurs.
- The **interrupt vector** is stored in low memory and holds the addresses of the **interrupt service routines** (or **interrupt handlers**).

# Interrupt Timeline Example



## Storage Structure

- ROM cannot be changed.
- EEPROM can be changed but not frequently.
- The main memory (or RAM) is implemented as DRAM.
- The CPU can load instructions only from the main memory.
- The CPU interacts with RAM using load or store instructions.
- The load instruction moves a word from RAM to a CPU register.
- The store instruction moves the content of a CPU register to RAM.

## Instruction Execution Cycle

- **Fetch** the instruction from memory and store it in the CPU's instruction register.

- **Decode** the instruction, which may require fetching operands from memory and store them in some CPU internal registers.

- **Execute** the instruction, which may store the result back in memory.

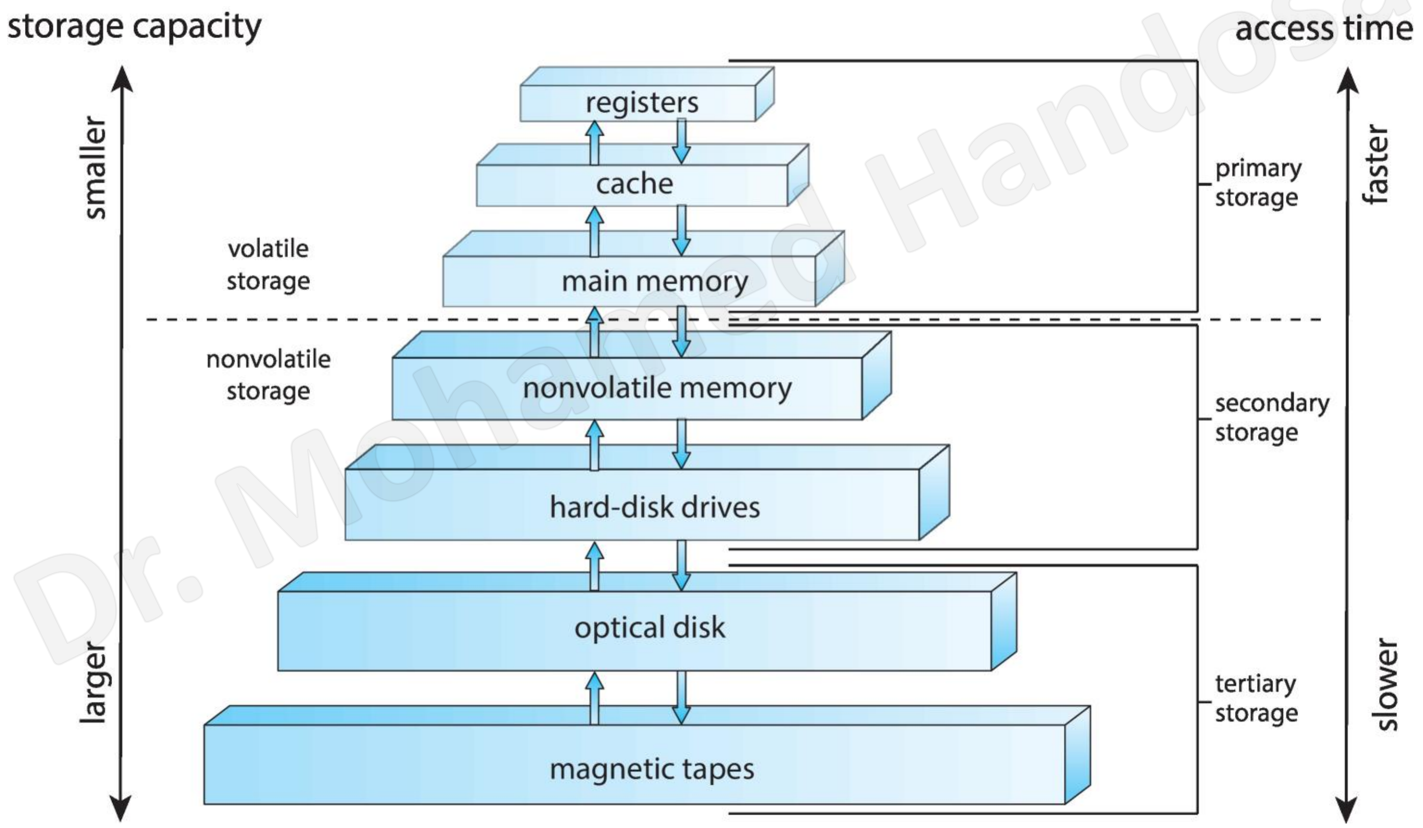
## Secondary Storage

- Main memory cannot store all programs and data permanently:
  - RAM is usually too small to store all needed programs and data.
  - RAM is a volatile storage device that loses its contents when power is off.

- Thus, most computer systems provide secondary storage as an extension of main memory.

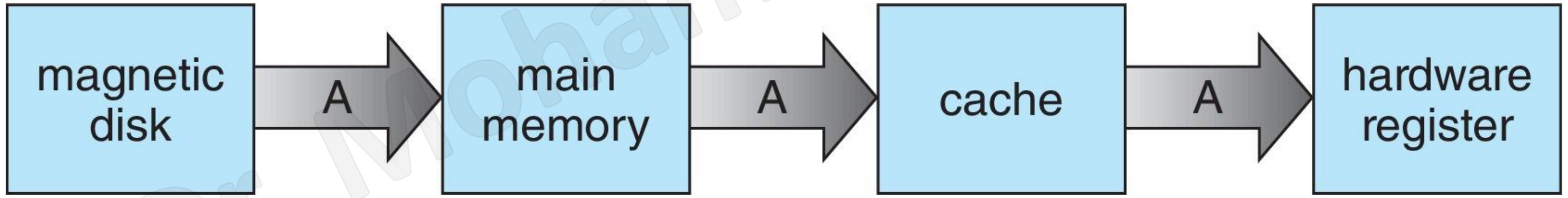
- The main requirement for secondary storage is to be able to hold large quantities of data permanently (e.g. hard disk drive).

# Storage Hierarchy



## Caching

- **Caches** can be installed to improve access time or transfer rate.
- **Caching** refers to the use of high-speed memory to hold a copy of recently-accessed data assuming that it will be needed again soon.



- **Cache coherency** ensures that an update to some data in a cache is reflected immediately in other caches containing a copy of that data.

## I/O Structure

- A general-purpose computer system consists of CPUs and multiple device controllers connected through a common bus.
- Each device controller controls a specific type of device (e.g. USB).
- A device controller has a local buffer storage and a set of registers.
- The device controller is responsible for moving the data between the peripheral devices that it controls and its local buffer storage.
- The operating system provides a device driver that understands the device controller and provides a uniform interface to the device.

## I/O Operation

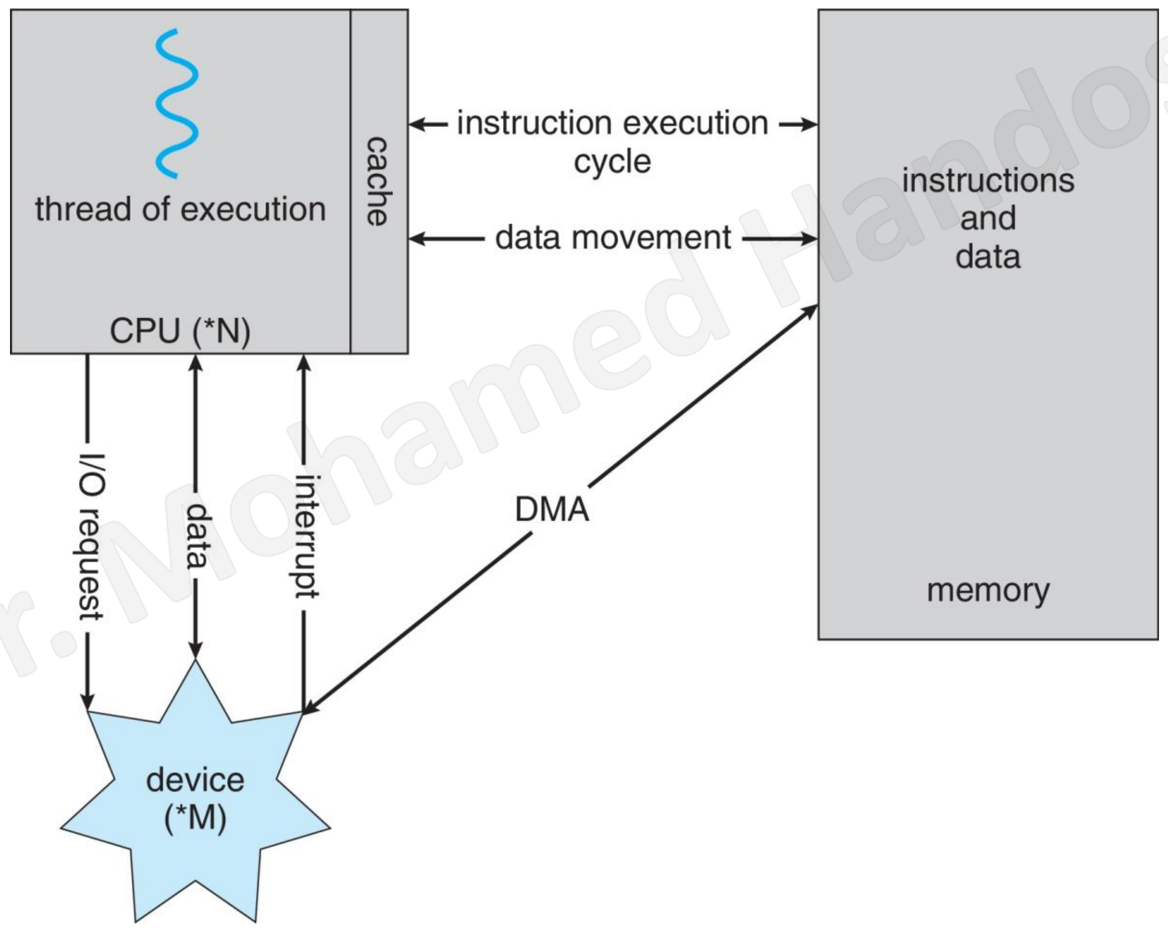
1. Device driver loads the registers within the device controller.
2. Device controller determines action to take based on registers.
3. Device controller transfers data between device and its local buffer.
4. Device controller generates an interrupt to inform the device driver.
5. Device driver returns control to the operating system
   - Possibly returning the data if the operation was a read.
   - For other operations, the device driver returns status information.

- This is fine for moving small amounts of data but can produce high overhead on the CPU when used for bulk data movement.

## Direct Memory Access

- Device controller transfers an entire block of data directly between its local buffer and main memory, with no intervention by the CPU.
- Device controller generates only one interrupt per block, to inform the device driver, rather than one interrupt per byte.
- While the device controller is transferring a whole data block, the CPU is available to accomplish other work.

# Modern Computer System



## Computer System Architecture

Single Processor Systems

Multiprocessor Systems

Clustered Systems

## Single Processor Systems

- A general-purpose processor supports a complete instruction set.
  - Therefore, it can execute user processes.
- A special-purpose processor supports a limited instruction set.
  - Therefore, it cannot execute user processes.
- Special-purpose processors include device-specific processors, such as disk, keyboard, and graphics controllers.
- A single processor system has only one general-purpose processor.
- The use of special-purpose microprocessors is common and does not turn a single-processor system into a multiprocessor.

## Multiprocessor Systems

- Also known as **parallel systems** or **tightly-coupled systems**.
- Two or more general-purpose processors in close communication, sharing bus, clock, memory, and peripheral devices.
- First appeared in servers but recently appeared on mobile devices such as smartphones and tablet computers.

# Advantages of Multiprocessor Systems

- Increased throughput.
  - More processors means more work in less time.
  - The speed-up ratio with $N$ processors is less than $N$.
- Economy of scale.
  - Sharing peripherals, mass storage, and power supplies means less cost.
- Increased reliability.
  - Failure of one processor will not halt the system, but only slows it down.

## System Reliability

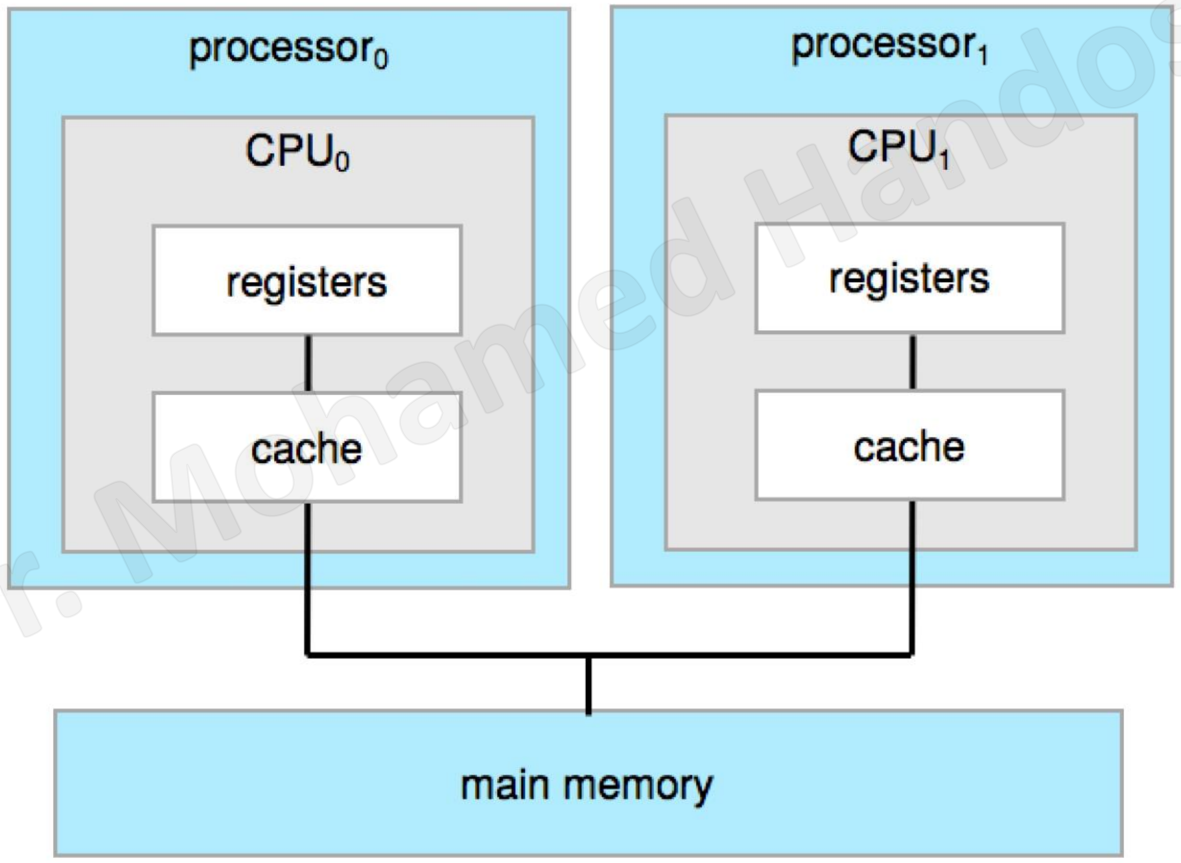
- Increased reliability is crucial in many applications.
- **Graceful degradation** is the ability to continue providing service proportional to the level of surviving hardware.
- **Fault tolerant systems** can continue operation despite of failures.
- **Fault tolerance** requires a mechanism to allow the failure to be detected, diagnosed, and, if possible, corrected.

## Types of Multiprocessing

- Asymmetric multiprocessing
  - This scheme defines a boss-worker relationship.
  - The boss processor schedules and allocates work to the worker processors.
  - Worker processors look to the boss for instruction or have predefined tasks.

- Symmetric multiprocessing (SMP)
  - Used by most systems.
  - No boss-worker relationship.
  - All processors are peers, where each processor can perform any task.

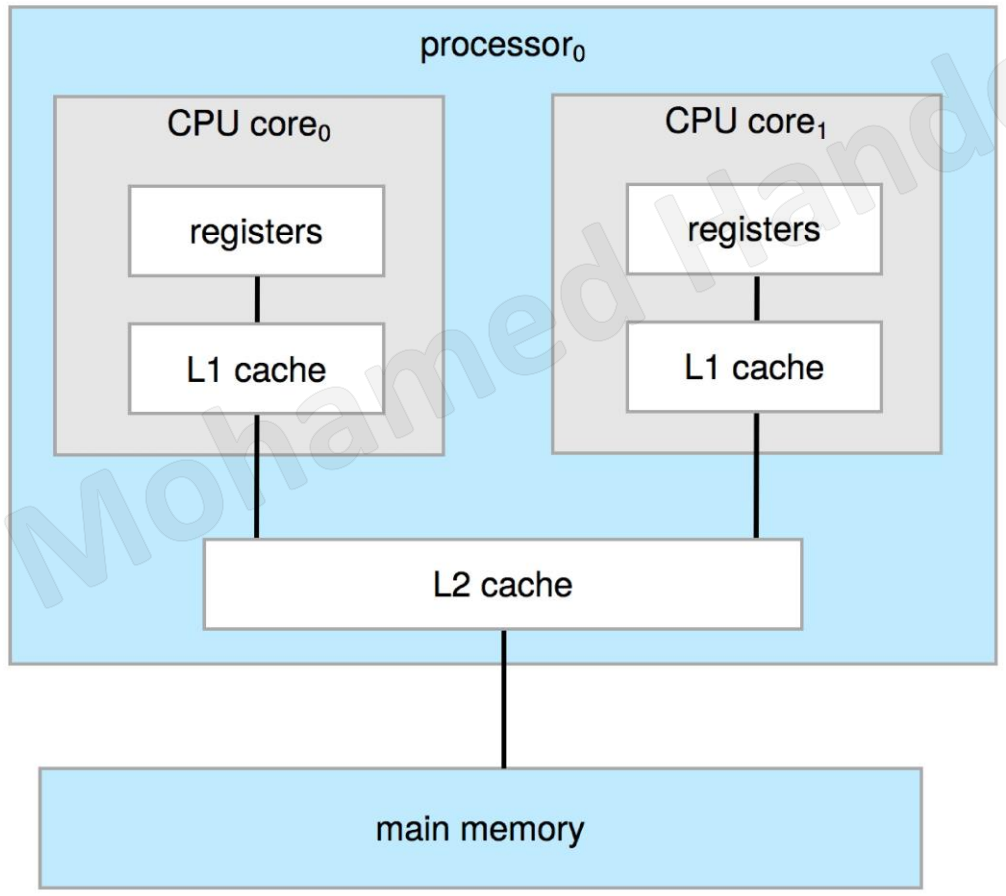
# Symmetric Multiprocessing Architecture



## Multicore Systems

- Multiple computing cores on a single chip.
- More efficient than multiple chips with single cores
    - On-chip communication is faster than between-chip communication.
    - One chip with multiple cores uses less power than multiple single-core chips.
- Multicore systems are multiprocessor systems.
- Not all multiprocessor systems are multicore systems.

# Dual-Core Architecture



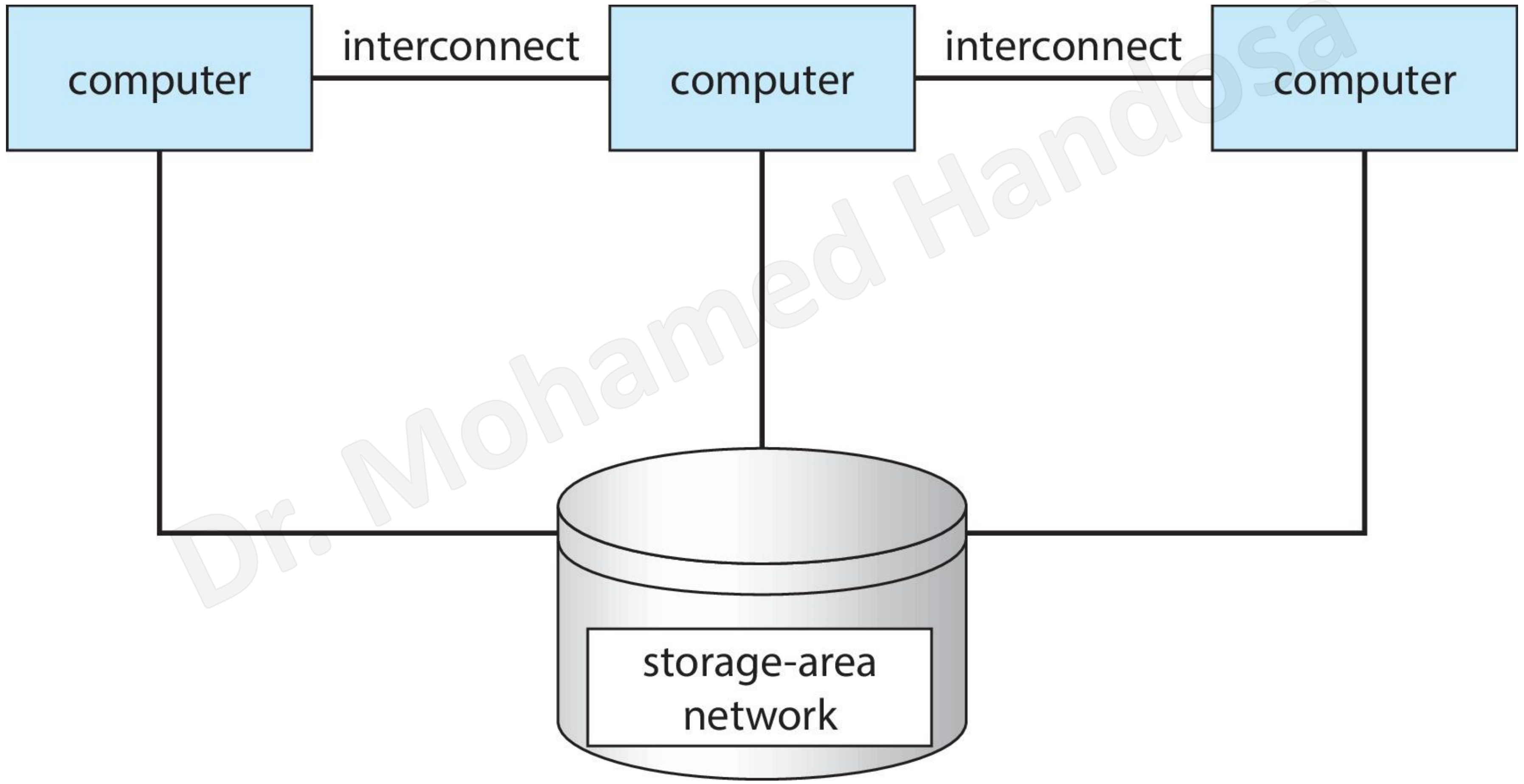
## Clustered Systems

- Also known as **loosely-coupled systems**.
- Composed of two or more individual systems (or nodes).
- Each node may be a single processor system or a multicore system.
- Clustered computers share storage and are closely linked via a LAN.

## Types of Clustering

- Asymmetric clustering
  - One node is in **hot-standby mode** monitoring the active server.
  - If the active server fails, the hot-standby node becomes the active server.
  - Supports **high-availability service**, where a node failure does not stop service.
- Symmetric clustering
  - Two or more nodes are running applications and are monitoring each other.
  - Supports **high performance computing** better than multiprocessor systems.
  - An application can run concurrently on all cluster nodes using **parallelization**.
  - **Parallelization** divides a program into separate components to run in parallel.

## General Structure of a clustered Systems



## Operating System Operations

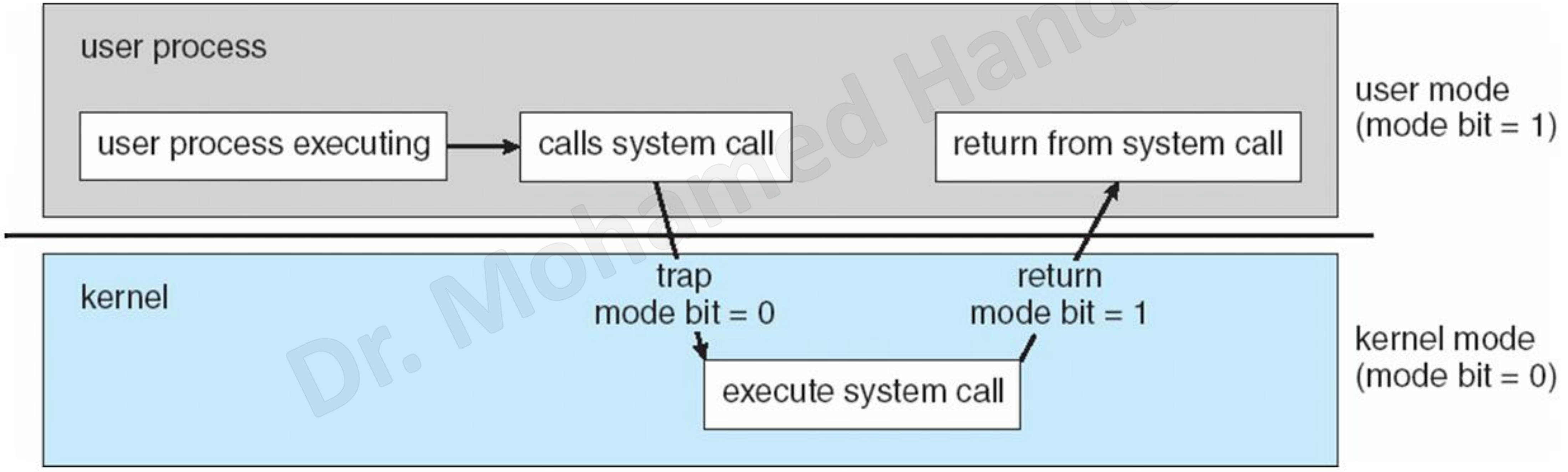
Dual Mode and Multimode Operation

Timer

## Dual-Mode Operation

- Protects the operating system from errant users.
- Two modes of operation: **user mode** and **kernel mode**.
- **Kernel mode = supervisor mode = system mode = privileged mode**.
- User defined code must execute in user mode.
- Only the operating system can execute in kernel mode.
- A **mode bit** indicates the current mode: kernel (0) or user (1).
- All instructions that may cause harm are deigned as **privileged**.
- **Privileged instructions** can execute only in kernel mode (i.e. by OS).

## Dual-Mode Operation



## Multi-Mode Operation

- The concept of modes can be extended beyond two modes.
- A CPU that supports virtualization can have a third mode for VMM.
  - VMM mode provide more privileges than user but fewer than kernel.
  - VMM needs these privileges so it can create and manage virtual machines.
- Sometimes, different modes are used by various kernel components.

VMM = Virtual Machine Manager

## Timer

- The OS must prevent a user program from running too long.
- A timer can be set to interrupt the system after a specified period.
- A timer is generally implemented by a fixed-rate clock and a counter.
- The OS sets the counter before turning the control to a user program.
  - Every time the clock ticks, the counter is decremented.
  - When the counter reaches 0, an interrupt occurs and control return to the OS.
  - The OS may treat the interrupt as a fatal error or give the program more time.
- Clearly, instructions to modify the content of the timer are privileged.

## Computing Environment

Traditional Computing

Mobile Computing

Distributed Computing

Virtualization

Cloud Computing

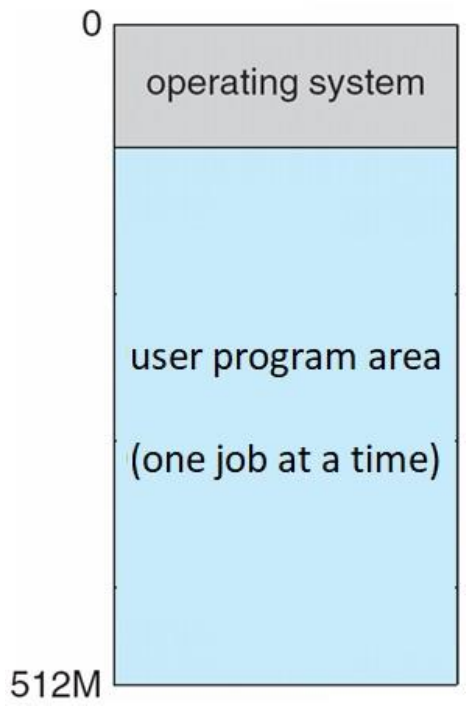
Real-Time Embedded Systems

## Traditional Computing

- Computer systems started as mainframe computers.
- Mainframes were used primarily by large organizations.
- Mainframe computers have evolved from batch systems to multiprogramming systems, and then time-sharing systems.
- Later, desktop computers were introduced and gained popularity.

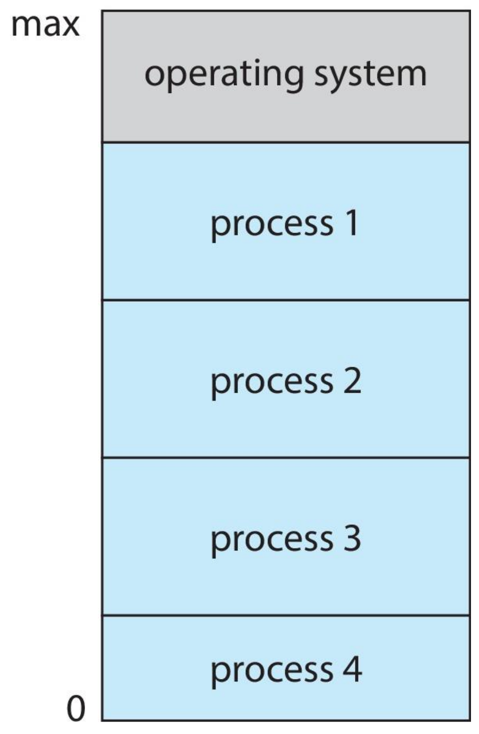
## Batch Systems

- Processed jobs in bulk, one job after another.
- Input devices were card readers and tape drives.
- Line printers output results and a memory dump.
- OS = **resident monitor** for **automatic job sequencing**.
- If the running job needs to wait for an I/O operation, the CPU remains idle waiting for the job.
- Disadvantages
  - Low CPU utilization as the CPU is often idle.
  - There is no direct interaction between user and system.



## Multiprogramming Systems

- Several jobs are kept in main memory.
- The CPU is always executing a job (no idle time).
- If the running job needs to wait for an I/O operation, the OS picks another job to start/continue execution.
- This requires the OS to provide several features:
  - **I/O routines:** only the OS can perform I/O.
  - **CPU scheduling:** to selecting a job for execution.
  - **Memory management:** to allocate memory to several jobs.
  - **Resource allocation:** no job can use resources of other jobs.
- Increased CPU utilization but still no direct interaction.



## Timesharing Systems

- An **interactive computer system**
  - Provides direct communication between the user and the system.
  - Should be responsive (i.e. response time to user input should be minimal).
- A **time-sharing** (or **multitasking**) system is
  - An interactive computer system.
  - An extension of multiprogramming systems.
- The CPU switches between jobs so frequently that the user can interact with each program while it is running.

## Desktop Computers

- A personal computer dedicated to a single user.
- I/O devices: keyboards, mice, display screens, small printers.
- OS focuses on achieving user convenience and responsiveness.
- Less focus on advanced CPU utilization and protection features.
- Desktop computer OSs include Windows, Mac OS, UNIX, and Linux.

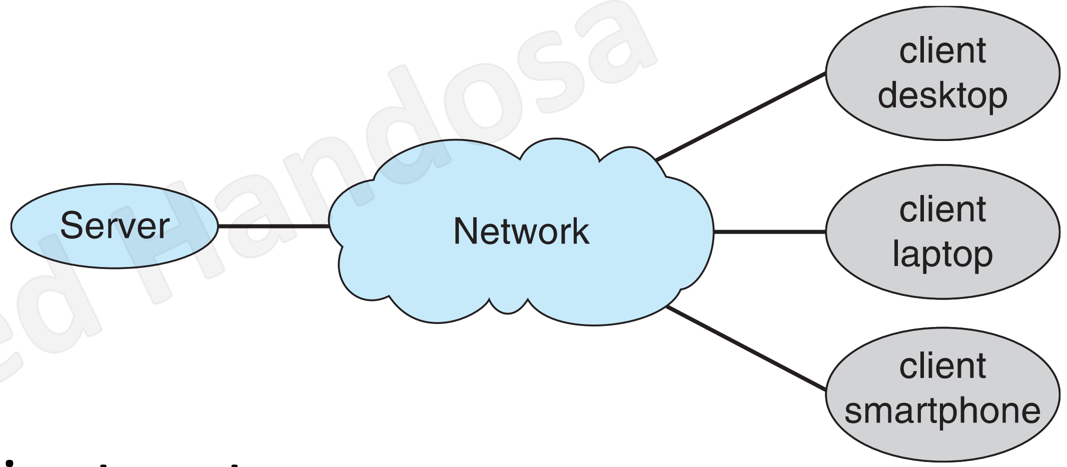
## Mobile Computing

- Computing on handheld smartphones and tablet computers.
- These devices are portable, lightweight, and battery-powered.
- A mobile device can provide features that are either unavailable or impractical on a desktop or laptop computer, such as:
  - GPS: to determine precise location of the device on Earth.
  - Accelerometers: to measure linear acceleration in different directions.
  - Gyroscopes: to detect orientation of the device with respect to the ground.
- Two OSs dominate mobile computing: Apple iOS and Google Android.

## Distributed Computing

- **Computer networks** are characterized by distances between nodes:
  - **Personal-area network (PAN):** wireless links over a short distance.
  - **Local-area network (LAN):** links nodes within a room, building, or campus.
  - **Metropolitan-area network (MAN):** connects computer nodes within a city.
  - **Wide-area network (WAN):** links nodes within buildings, cities, or countries.
- A **distributed system** is a collection of physically separate, possibly heterogeneous, computer systems that are networked together.
- Distributed computing
  - is the use of a distributed systems to solve single large problems.
  - may take the form of **client-server computing** or **peer-to-peer computing**.

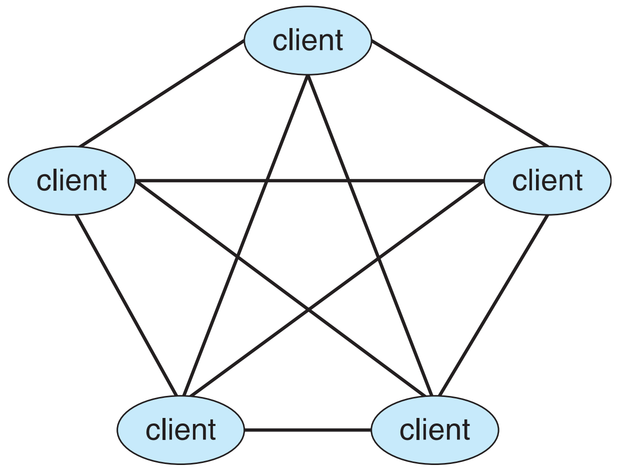
## Client-Server Computing



- A model of distributed systems.
- Server systems satisfy requests of client systems.
- A server can be categorized as a compute-server or a file-server.
    - **File-servers** allows clients to create, update, read, and delete files.
    - **Compute-servers** receive client requests, execute actions, and return results.

## Peer-to-Peer Computing

- Another model of distributed computing.
- A node must first join the peer-to-peer system.
- All nodes are peer, where each node may act as
  - A server, when providing services to other nodes.
  - A client, when requesting services from other nodes.

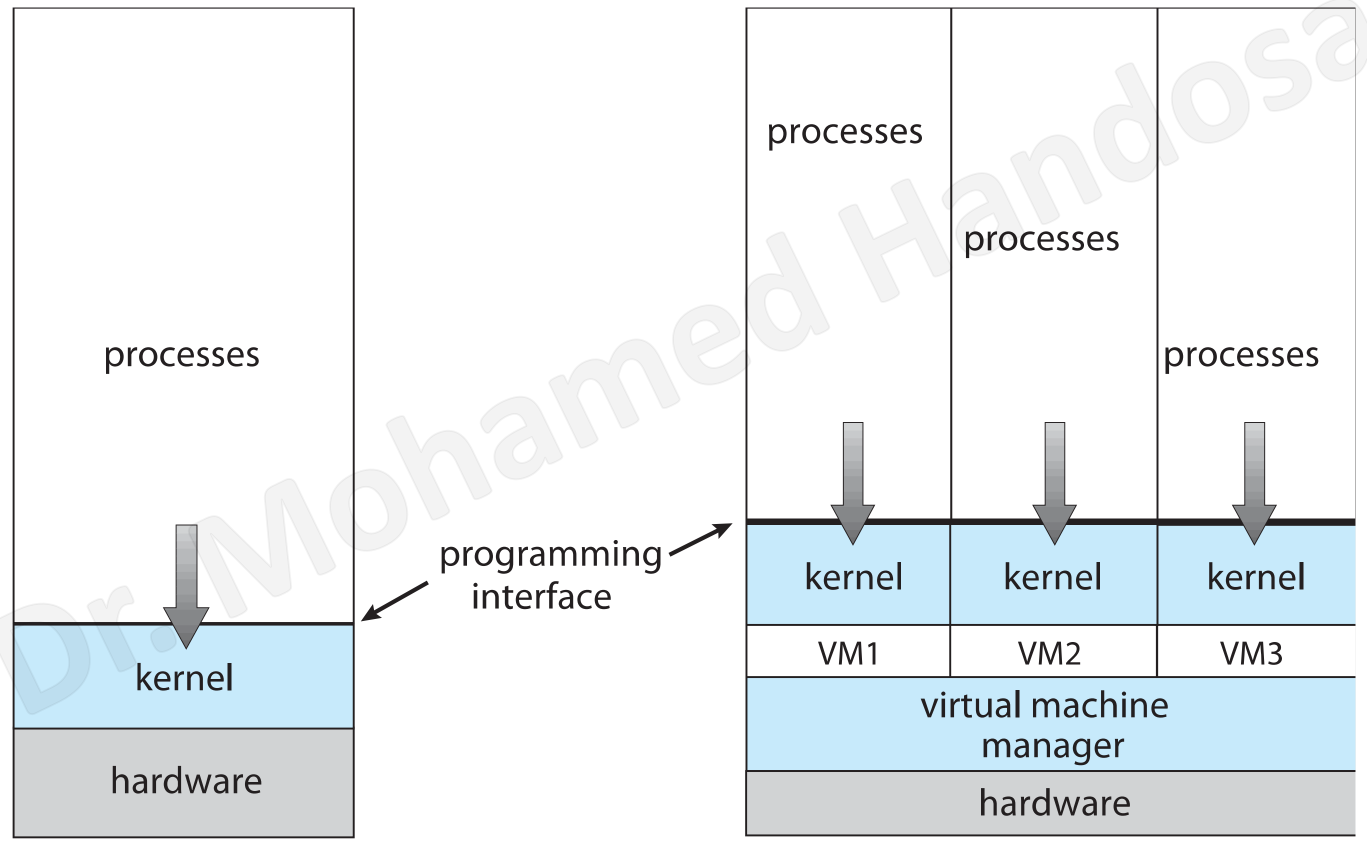


- In client-server systems, a server is a bottleneck.
- In P2P systems, several nodes can provide the service.

## Virtualization

- Allows for creating a virtual machine that acts like a real computer.
- A single computer (host) may run several virtual machines (guests).
- The host and the guests share the hardware, but each has its own OS.
- The virtual machine manager runs the guest machines, manages their resource use, and protects each guest from the others.

# Virtualization



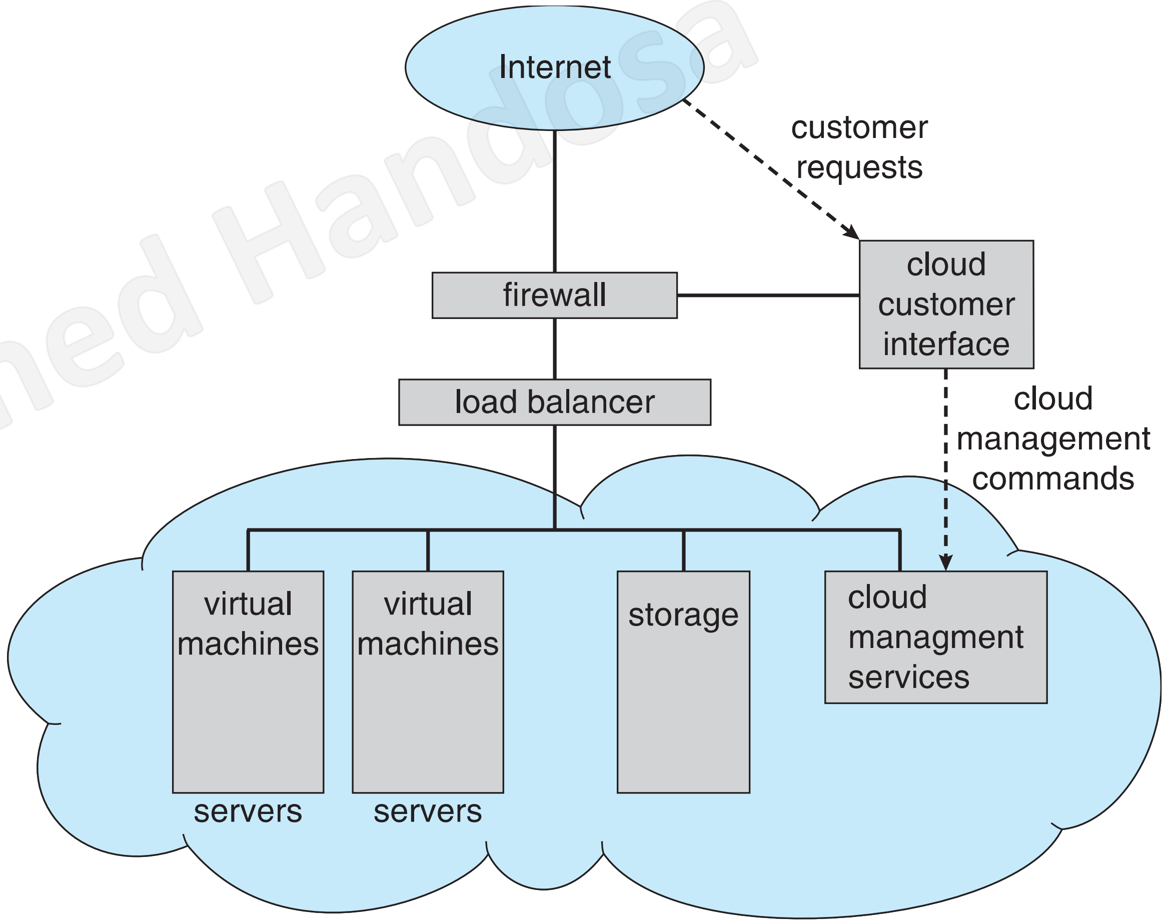
(a) (b)

## Cloud Computing

- Delivers computing as a service and users pay based on usage.
- Uses virtualization to run millions of VMS on thousands of servers.
- A cloud service can be categorized as public, private, or hybrid cloud.
  - A **public cloud** is available to anyone via the Internet.
  - A **private cloud** is used only by the company owning it.
  - A **hybrid cloud** includes both public and private cloud components.
- It may provide infrastructure, platform, or application as a service.
  - **Software as a service (SaaS)** provides applications (e.g. Google Docs).
  - **Platform as a service (PaaS)** provides a software stack (e.g. Google Firebase).
  - **Infrastructure as a service (IaaS)** provides servers or storage over the internet

# Cloud Computing

- Internet connectivity requires security like firewalls.
- Load balancers spread traffic across multiple applications.



## Real-Time Embedded Systems

- Embedded computers are primitive with very specific tasks.
- Their OSs provide limited features with little or no user interface.
- Embedded systems almost always run real-time operating systems.
  - A real-time system has well-defined, fixed time constraints.
  - Processing must be done within the defined constraints, or the system fails.
- Examples: weapons systems, nuclear plant systems, medical imaging systems, and industrial control systems.

# Thank You