



Operating System

Lecture 4

Dr. Ghada Fathy



Table of contents



01 Computer-System
Architecture

02 A Dual-Core Design

03 Operating System
Structure

04 Clustered Systems

Computer System Architecture



- **Computer-System Architecture**, focusing on how **processors** (CPUs) are organized inside a computer system.

1- Single Processor System

- Traditional computers have one general-purpose CPU.
- This CPU executes the operating system instructions and user programs.
- Example: Old desktop or simple embedded system.
- Limitations: Only one instruction stream at a time → performance limited.

2- Special-Purpose Processors

- Besides the main CPU, systems may include special-purpose processors that handle specific tasks.

Examples:

- GPU (Graphics Processing Unit): handles graphics/calculations.
- I/O processor: manages input/output devices.
- Network processor: handles network communication.
- These work under the control of the main CPU but relieve it from heavy, repetitive work.

Computer System Architecture

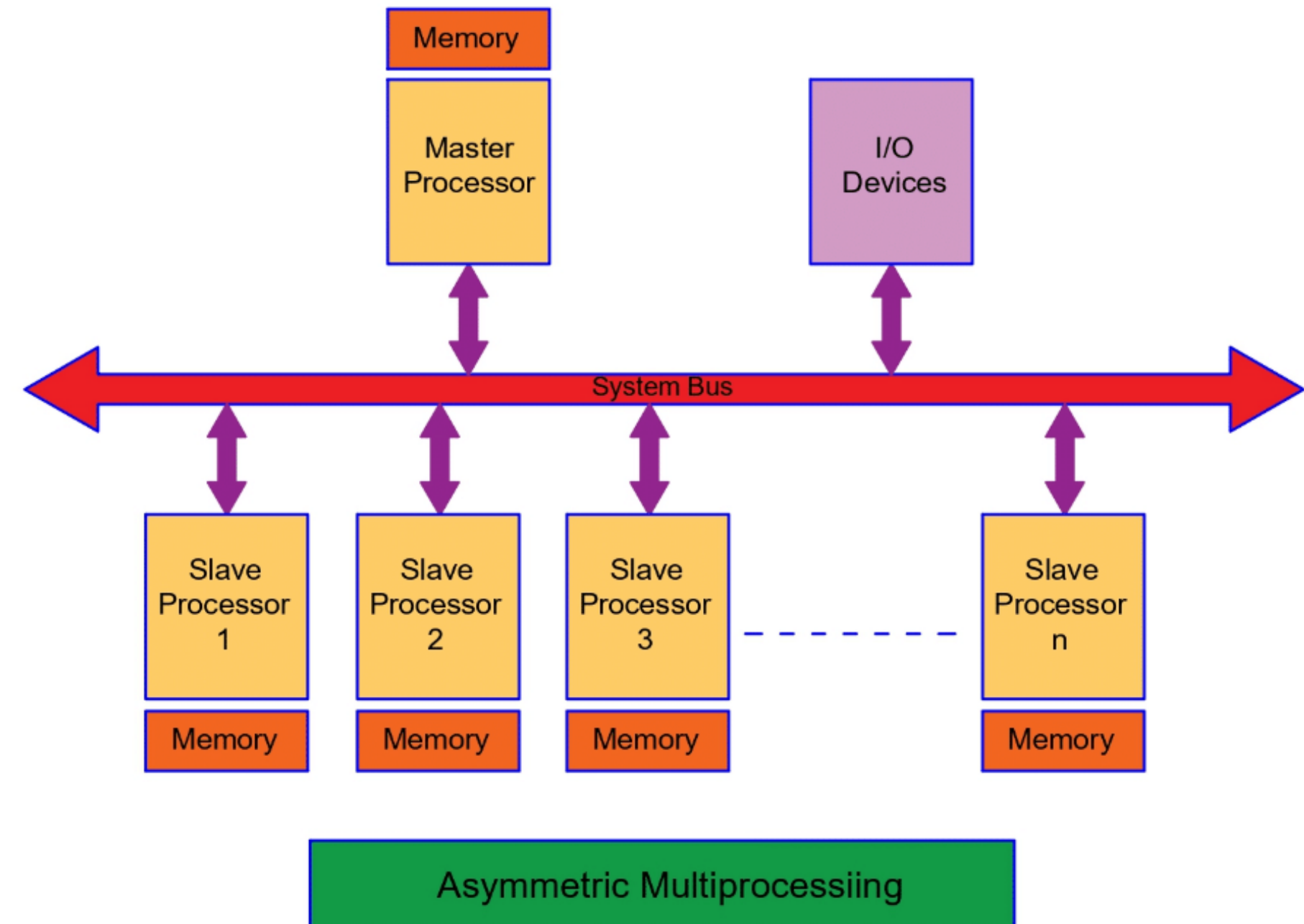


3- Multiprocessor Systems (Parallel Systems)

- Systems with two or more CPUs that share main memory and work together. Also called: **Parallel systems**, and **Tightly-coupled systems** (because processors share memory and communicate closely)

Advantages:

- **Increased Throughput**
More CPUs = more instructions executed per second → better performance.
- **Economy of Scale**
It's cheaper to have several processors sharing memory and devices than building several separate computers.
- **Increased Reliability (Fault Tolerance)**
If one CPU fails, others can continue — the system degrades gracefully instead of crashing completely.

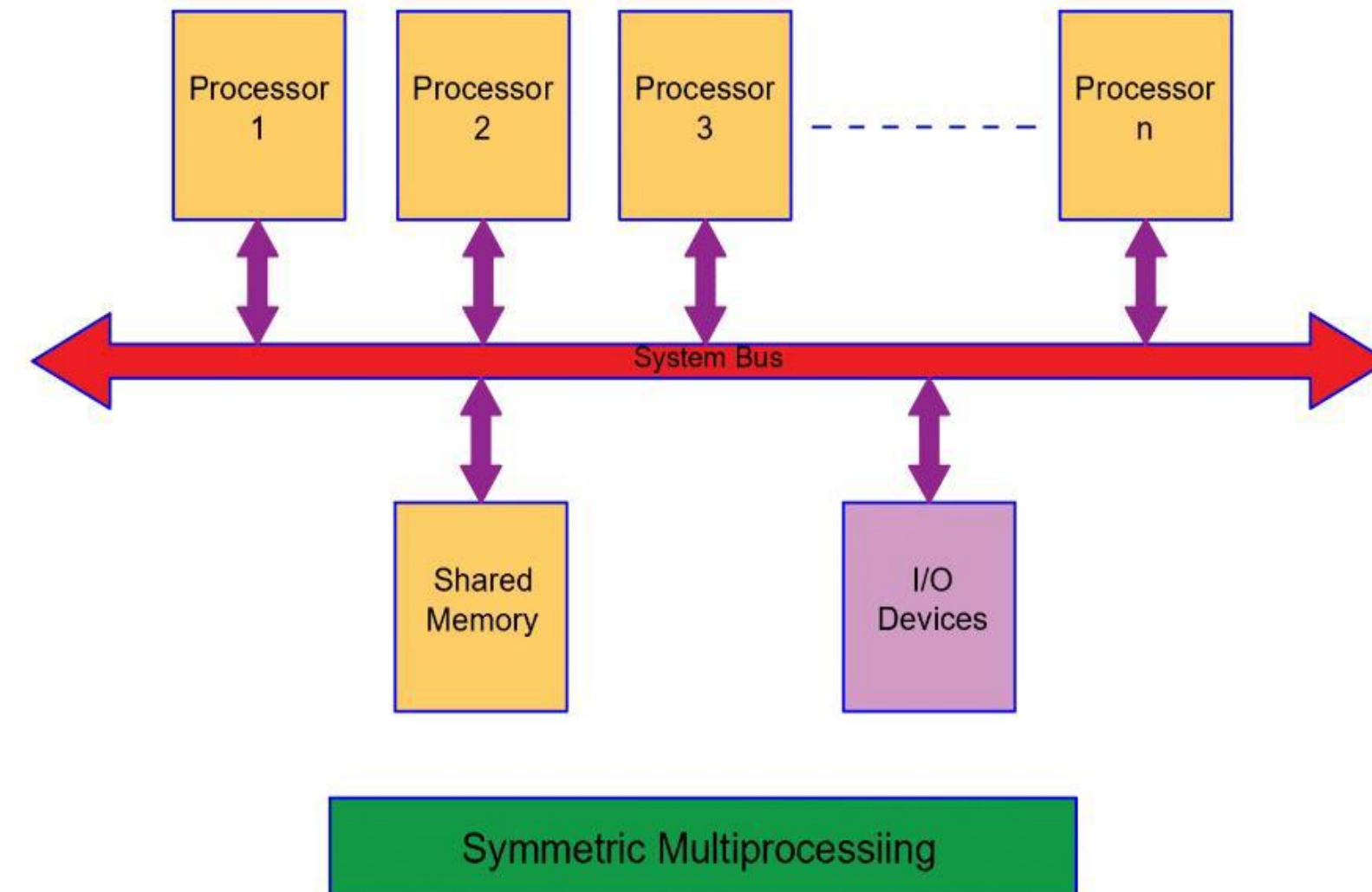


Types of Multiprocessing Systems

- **Symmetric Multiprocessing (SMP)**
 - All processors are peers — each can perform any task.
 - They share memory and operate under a single OS.
 - The OS decides which process runs on which CPU dynamically.

Advantages of SMP:

- Balanced workload.
- Better performance scalability.
- If one CPU is busy, another can take over its tasks.

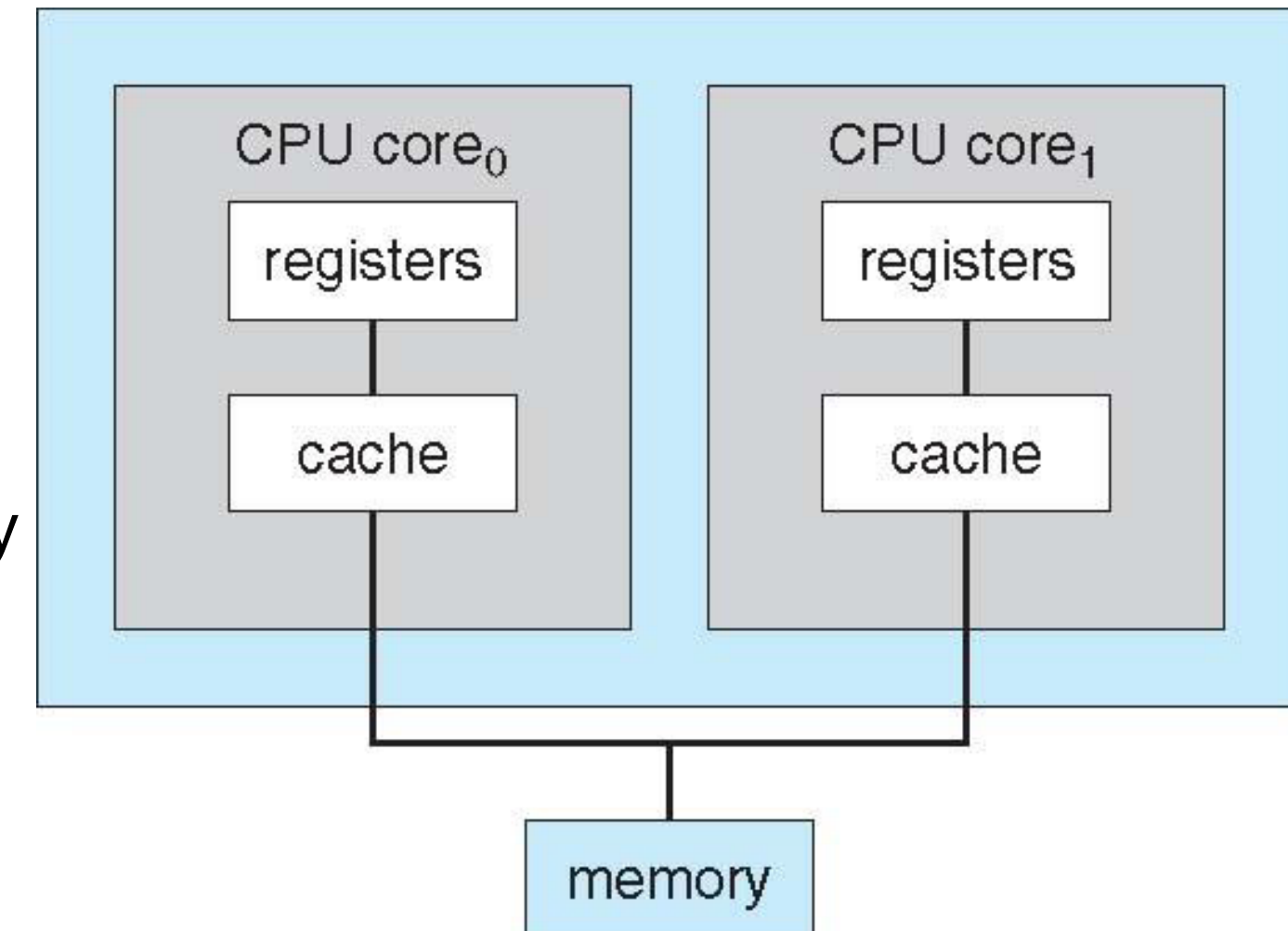


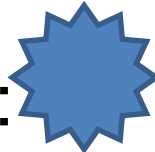
A Dual-Core Design



A **dual-core processor** means there are **two CPU cores** on a **single physical chip**.

- Each core can execute its own instructions independently, which makes the computer faster and more efficient.
- Both cores share some hardware, such as cache and memory bus, but they can process two tasks at the same time.



Note :  cache shown in the diagram is typically the **L1 (Level 1)** cache — the fastest and smallest cache, dedicated per core not the CPU cache.

Operating System Structure



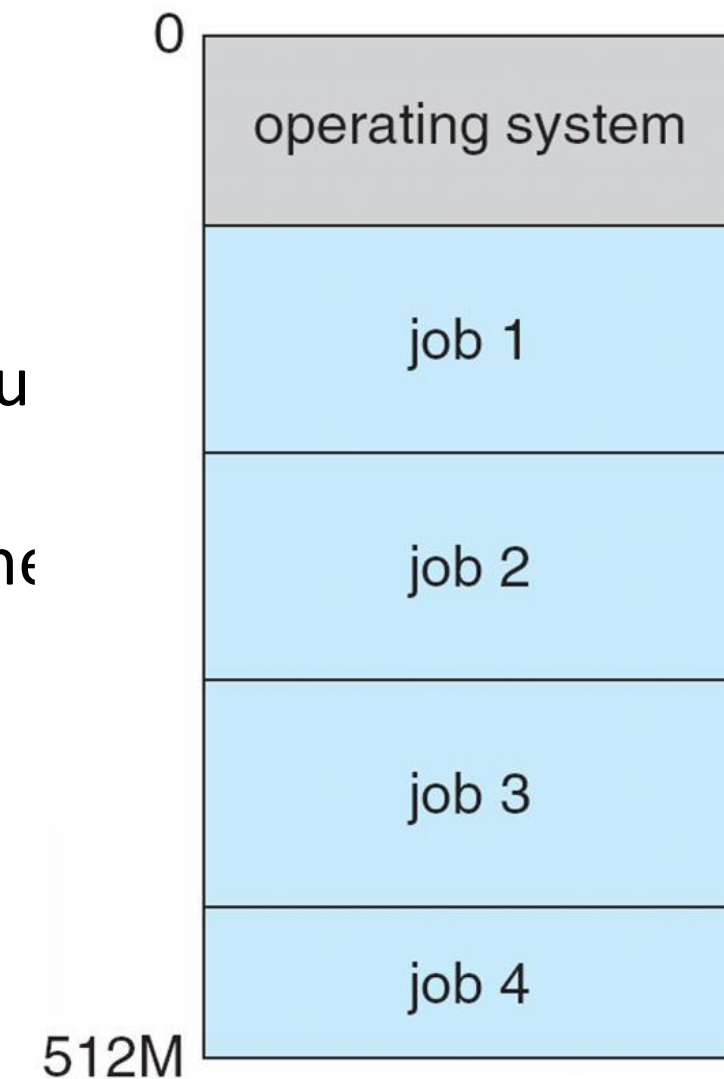
- **The Operating System (OS)** manages CPU and memory efficiently using **multiprogramming** and **timesharing** concepts.

Multiprogramming (Batch system) needed for efficiency:

- A single user (or a single job) cannot keep the CPU and I/O devices busy all the time.
- When one program waits for I/O (for example, reading from disk), the CPU would normally be idle.
- To avoid wasting time, multiprogramming keeps multiple jobs in memory, so the CPU always has something to execute.

How Multiprogramming Works:

- Several jobs (programs + data) are loaded into main memory.
- The job scheduler selects one job to run.
- When that job needs to wait for I/O, the OS switches to another job.
- This way, the CPU never stays idle — it's always executing something.

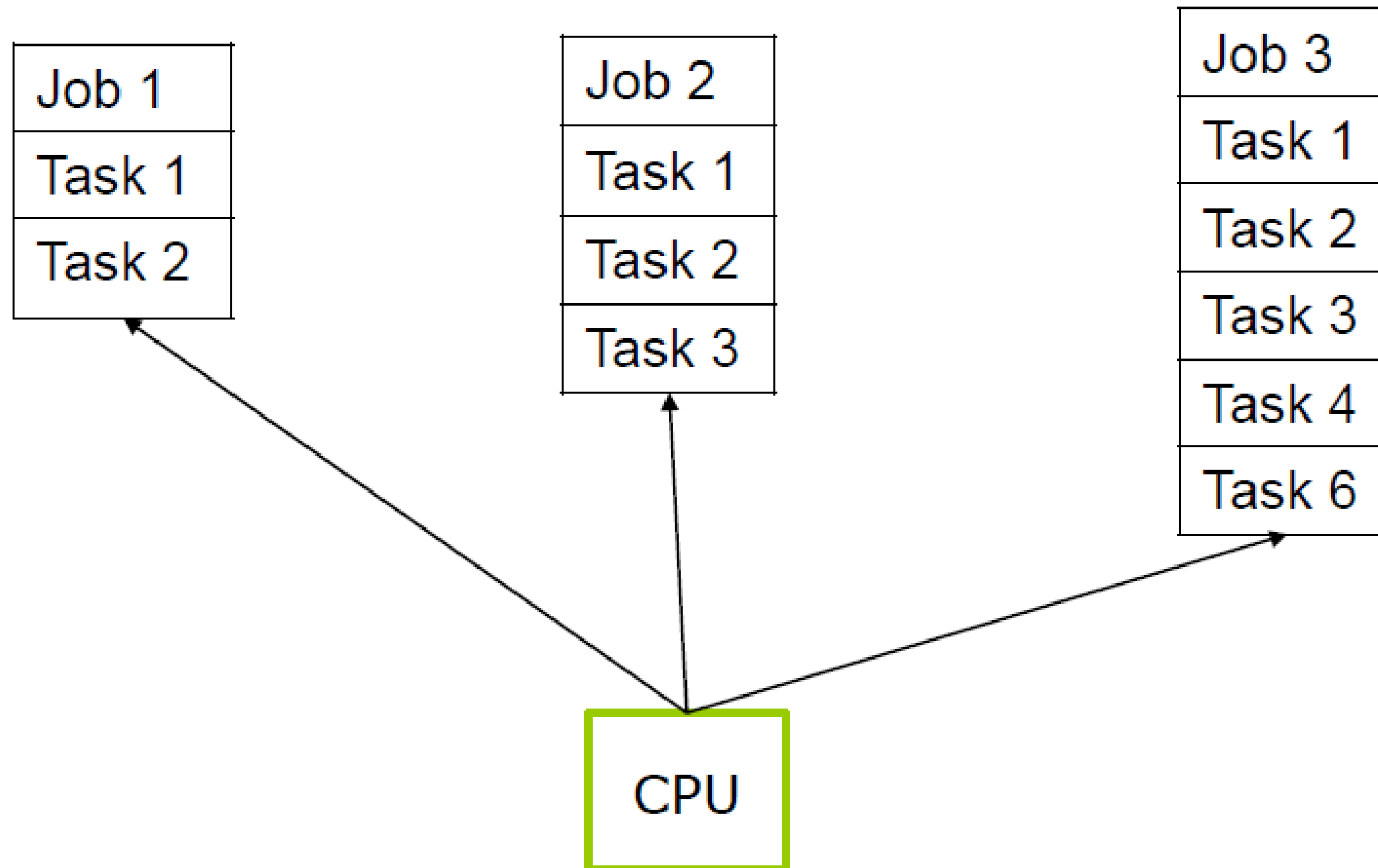


Operating System Structure



- Timesharing (Multitasking)
 - Timesharing is an extension of multiprogramming — it allows many users to use the computer at the same time.
 - The CPU switches between users' programs very quickly, giving the illusion that everyone's program is running simultaneously.
 - Response time should be less than 1 second so that users feel the system is interactive.
-

How a Modern Computer Works

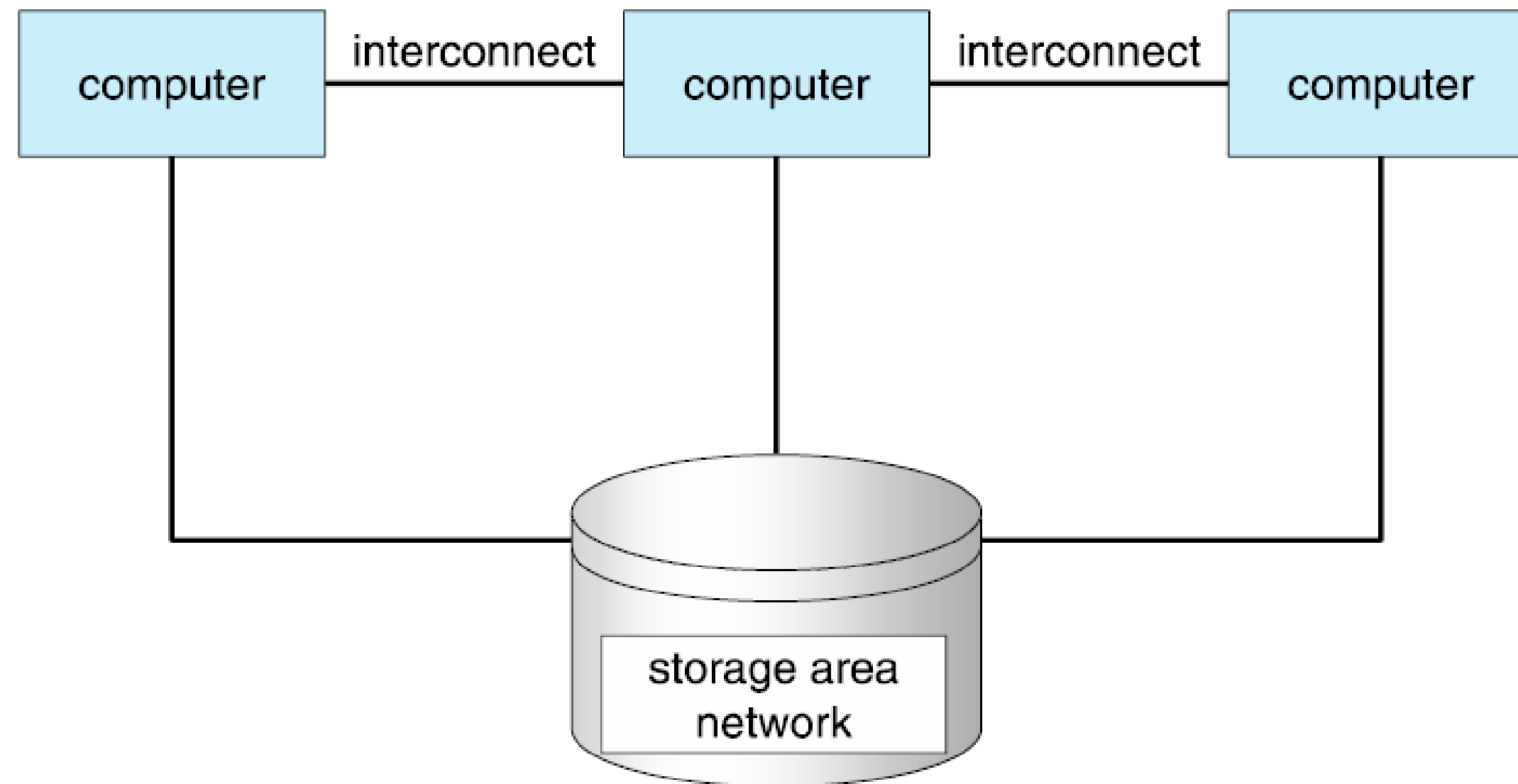


Clustered Systems



- Like multiprocessor systems, but multiple systems working together.
 - Usually sharing storage via a **storage-area network (SAN)**.
 - Provides a high-availability service that survives failures.
 - ❑ **Asymmetric clustering**
 - One computer is active; another is on standby (hot backup).
 - The standby only becomes active if the main one fails.
 - ❑ **Symmetric clustering**
 - All nodes are active simultaneously.
 - They share the workload and monitor each other for failures.
 - More efficient but also more complex to manage.
- Some clusters are for **high-performance computing (HPC)** Applications must be written to use **parallelization**
- Some have **distributed lock managers (DLM)** to avoid conflicting operations.
 - Ensures two nodes don't modify the same file/data at the same time

Clustered Systems



*Thank
you*

