Introduction

CSCI 3431: Operating Systems

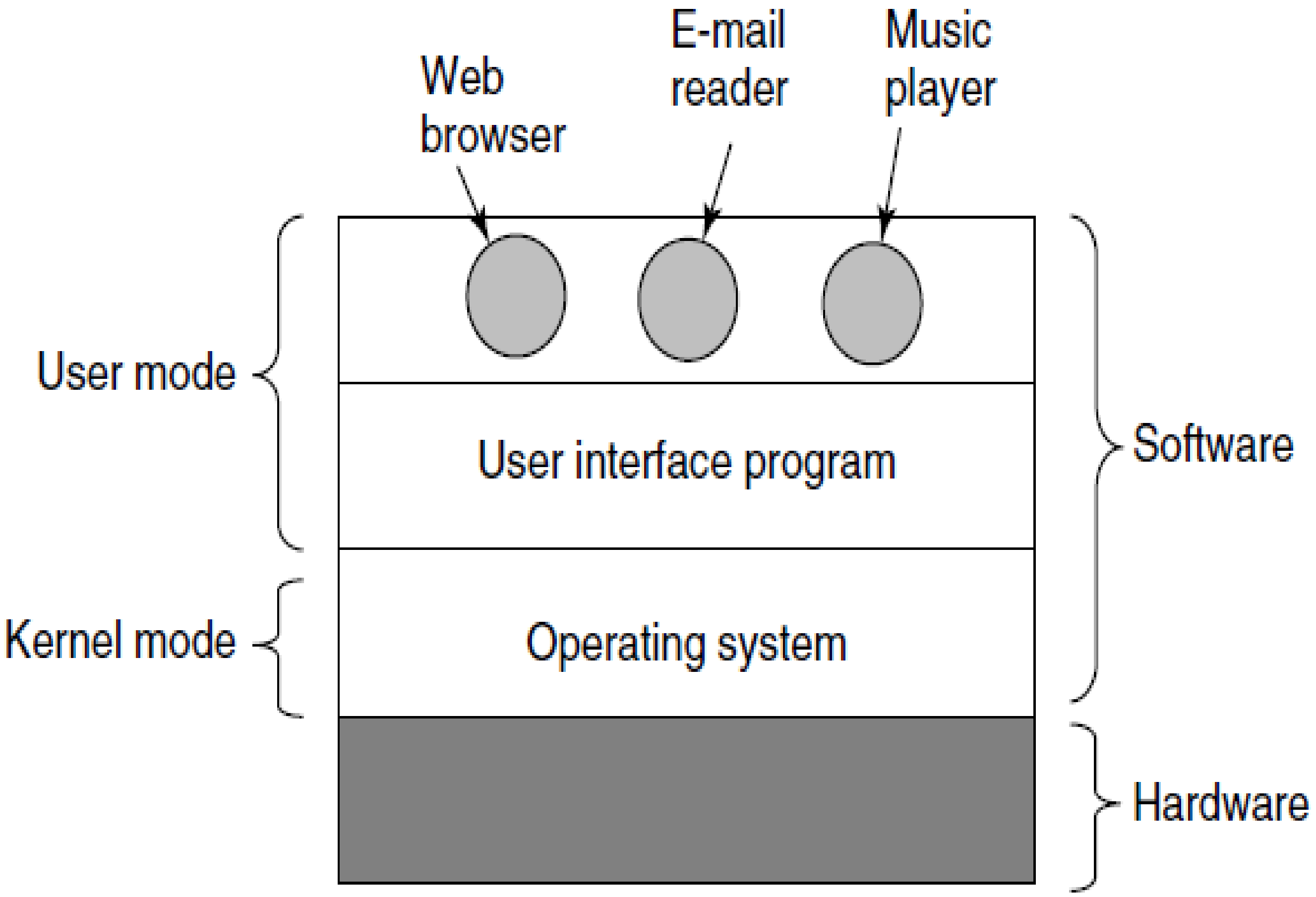
Agenda

- Assignment 1 out today
  - Due date: September 29, 2017
  - Closing date: October 1, 2017
- Today's lecture
  - OS definition
  - A little history
  - Systems components (2327 review)
  - Computing environments
  - OS concepts Zoo
  - Textbook Reading: 1.1-1.3, 1.9-1.13, 2.1-2.2

What is an OS?

- 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

Where does the OS fit?

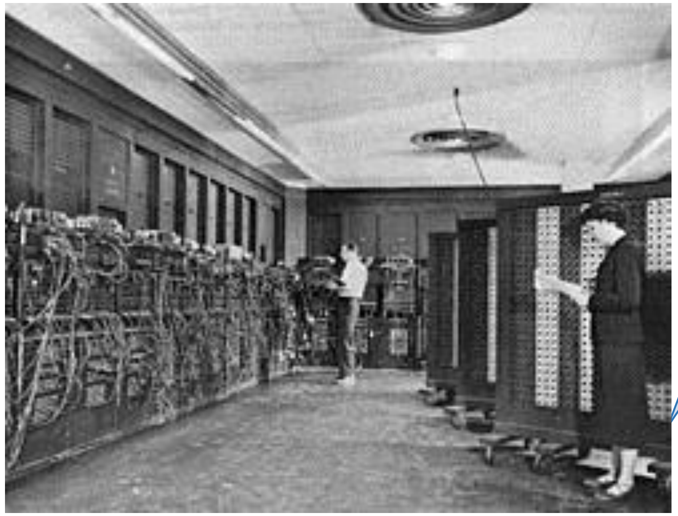


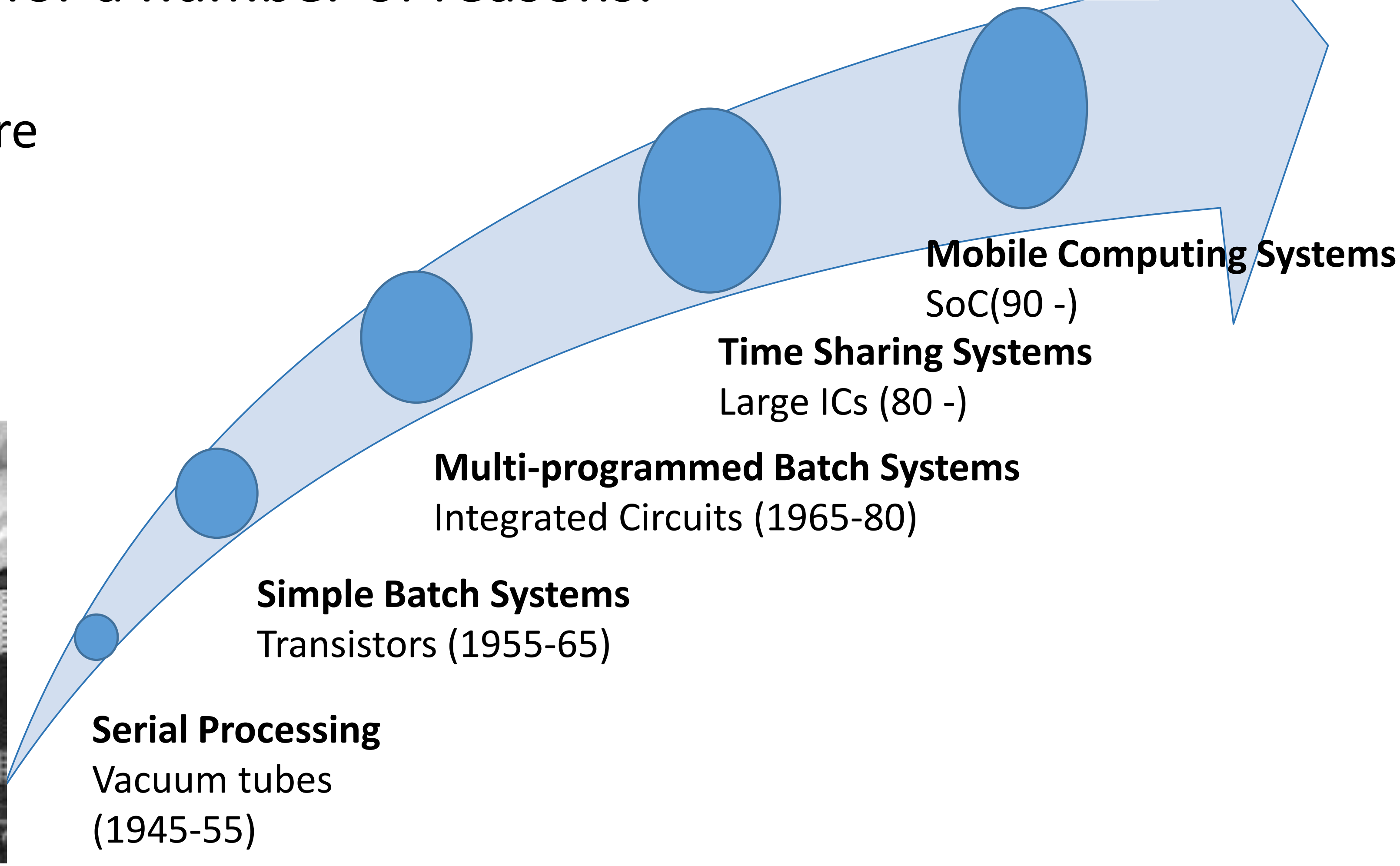
OS Definition

- 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

Evolution of Operating Systems

- OS evolved over time for a number of reasons:
  - Hardware upgrades
  - New types of hardware
  - New services
  - Fixes





In the Beginning….

- A computer ran ONE program at a time
- A program
  - Was loaded into memory
  - Executed on the computer
  - Read input
  - Performed computation
  - Generated output

- No Operating System was involved
- Each program took over all computer resources (hardware)

Computers were **Non-interactive** and very **inefficient**!

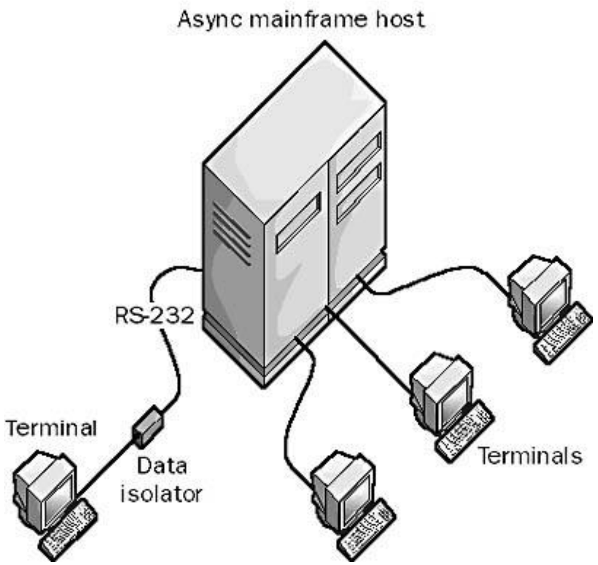
Batch Systems

- A batch of programs was load
- A program
  - Was loaded into memory
  - Executed on the computer
  - Read input
  - Performed computation
  - Generated output
  - **Unloaded for next program to load**
- Operating System was needed to ensure
  - No program could monopolize the hardware
  - Hardware was reset for each program
  - Program had the needed resources

Computers were STILL **Non-Interactive** and **inefficient**!

Time Shared Systems

- Many users each running a program
- A User
  - Logged into the system via a terminal
  - Ran an interpreter (shell) to submit commands
  - Executed programs
  - Received feedback from the computer
- Operating System was needed to ensure
  - No program could monopolize the hardware
  - Users could not interfere with one another
  - Programs accessed resources via requests



Computers became **Interactive** and still **inefficient**!

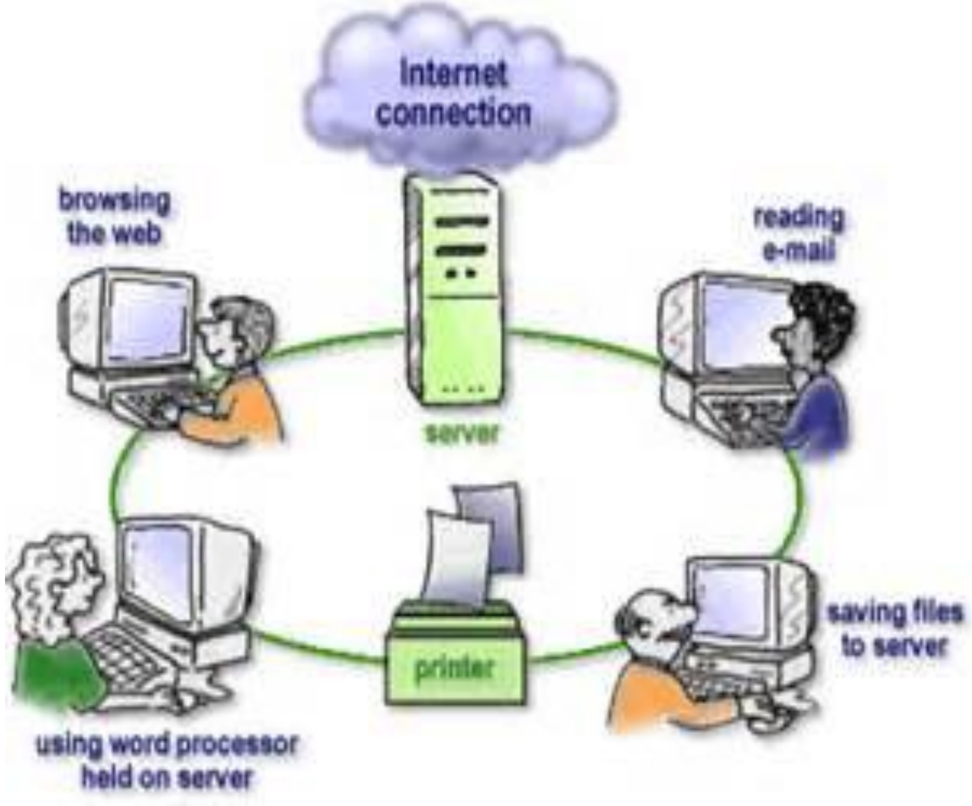
Personal Computers

- Single user running one program at a time
- A User
  - Ran an interpreter (shell) to submit commands
  - Loaded and executed a program
  - Executed programs
  - Received feedback from the computer
  - Unloaded program
  - Selected next program to run
- Operating System was needed to ensure
  - User interaction with program
  - Hardware was reset for each program
  - Program had required resources
  - Commands were received from user

Computers were **Interactive** and more **efficient**!

Modern Computers

- Multiple Users running multiple programs simultaneously
- Operating System is needed to ensure
  - Programs are loaded
  - Programs have the required resources
  - Users interaction with programs
  - Programs interaction with programs
  - Controlled access to resources
  - Defense of the system again internal and external attacks



Computers are very **Interactive** and more **efficient**!

There is Room for Better....

- OS researchers view operating systems as:
  - Massive
  - Inflexible
  - Unreliable
  - Insecure
  - Loaded with bugs

The User-System Perspective

User Perspective

- Program development
- Program execution
- Access to I/O devices
- Controlled access to files
- System access

System Perspective

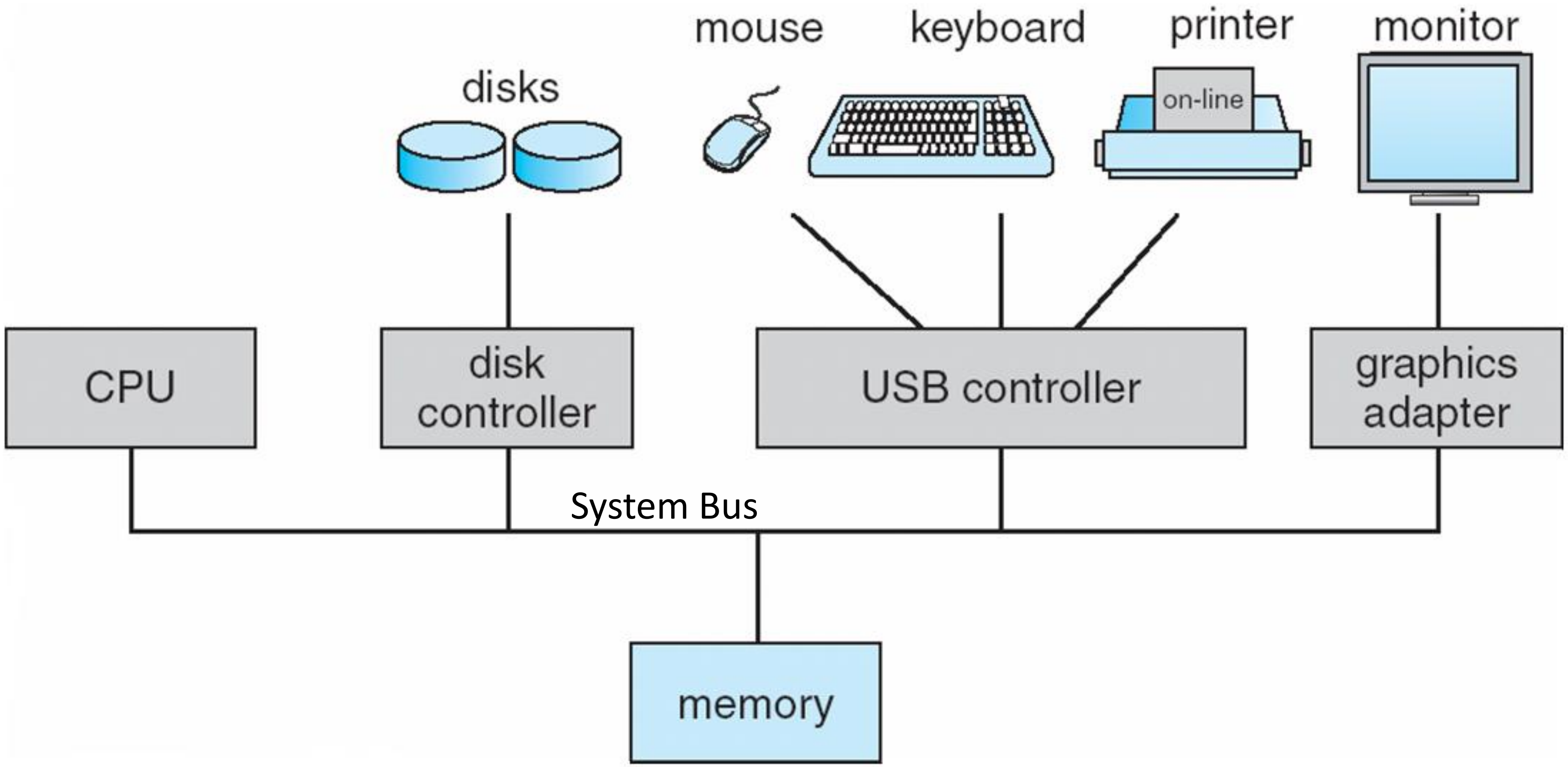
- Resource management
- Scheduling
- Interface to hardware
- Protection
- Security

Objectives of an OS

- Make Computing Efficient by
  - Allowing users to use applications → **Convenience**
  - Allowing applications to use computer resources → **Efficiency**
  - Allowing applications and users to interact → **Ability to evolve**

- What does an OS do to achieve these goals?
- Let’s go back to review systems components

So Let's Review Systems Components

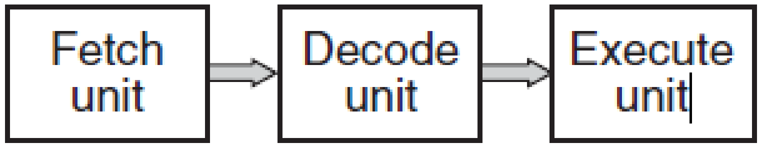


Processor

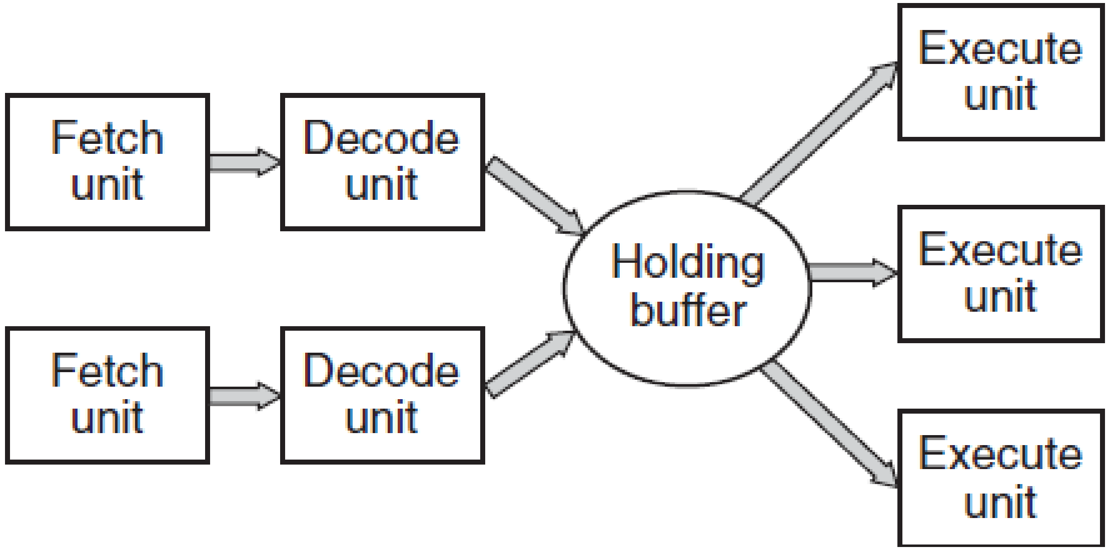
- Controls the operation of the computer
- Performs data processing functions
- Referred to as the Central Processing Unit (CPU)

Processor in Action

- A program consists of a set of instructions stored in memory
- Processor reads instructions from memory and executes each instruction



a)Three-stage pipeline



b) A superscalar CPU

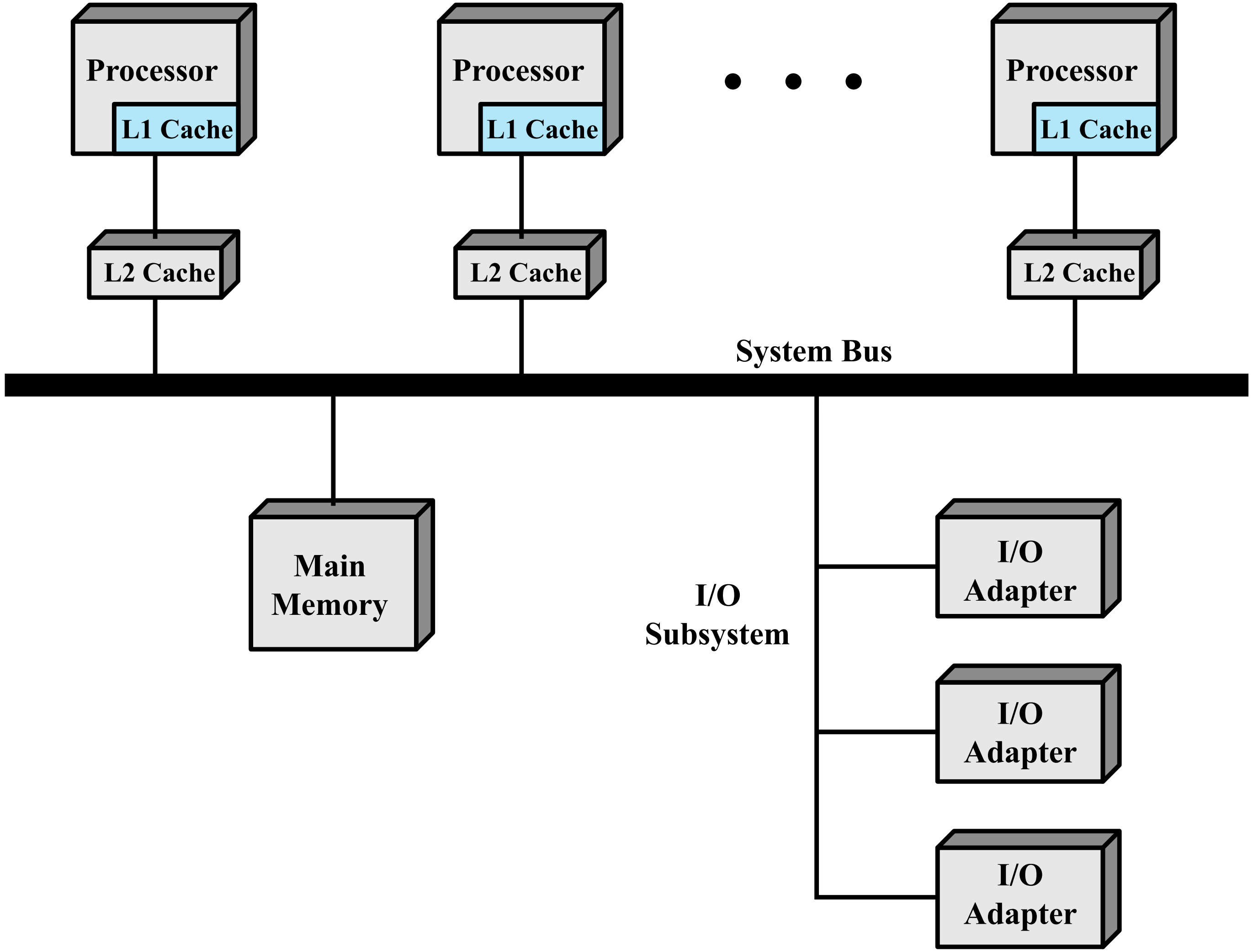
Symmetric Multiprocessors - SMP

- A stand-alone computer system with the following characteristics:
  - Two or more similar processors of comparable capability
  - Processors share the same main memory and are interconnected by a bus or other internal connection scheme
  - Processors share access to I/O devices
  - All processors can perform the same functions
  - The system is controlled by an integrated operating system that provides interaction between processors and their programs at the job, task, file, and data element levels

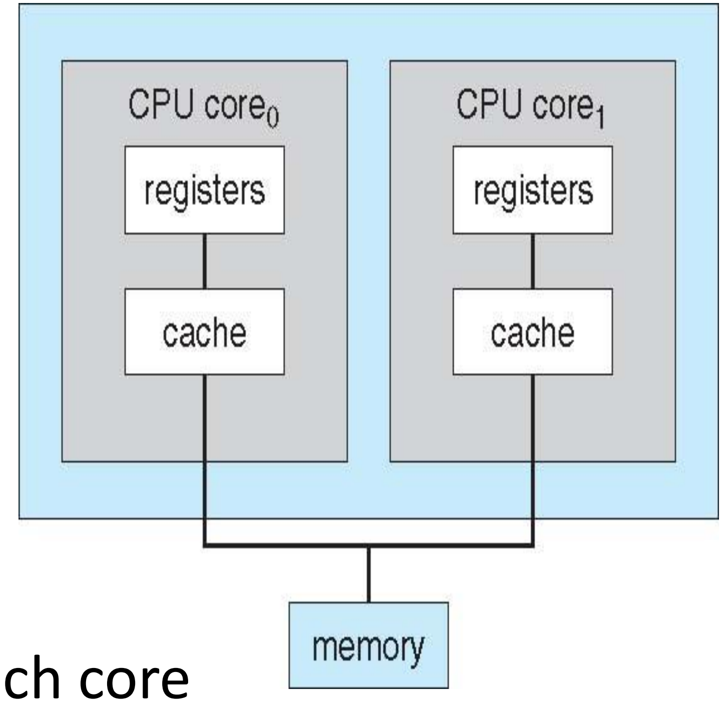
SMP Advantages

SMP vs uniprocessor

- Performance
- Availability
- Incremental growth
- Scaling



Microprocessor



- Fastest general purpose processors
- Multiprocessors
  - Each chip contains multiple processors (called cores)
  - Each with multiple levels of large memory caches
  - Multiple logical processors sharing execution units of each core
- As of 2010, it is not unusual to find even laptops with 2-4 cores
  - Each with 2 hardware threads, for a total of 4-8 logical processors
  - Multicore computers
  - Intel Xeon Phi, Tilera Tilepro support 60 cores on a single ship

Memory

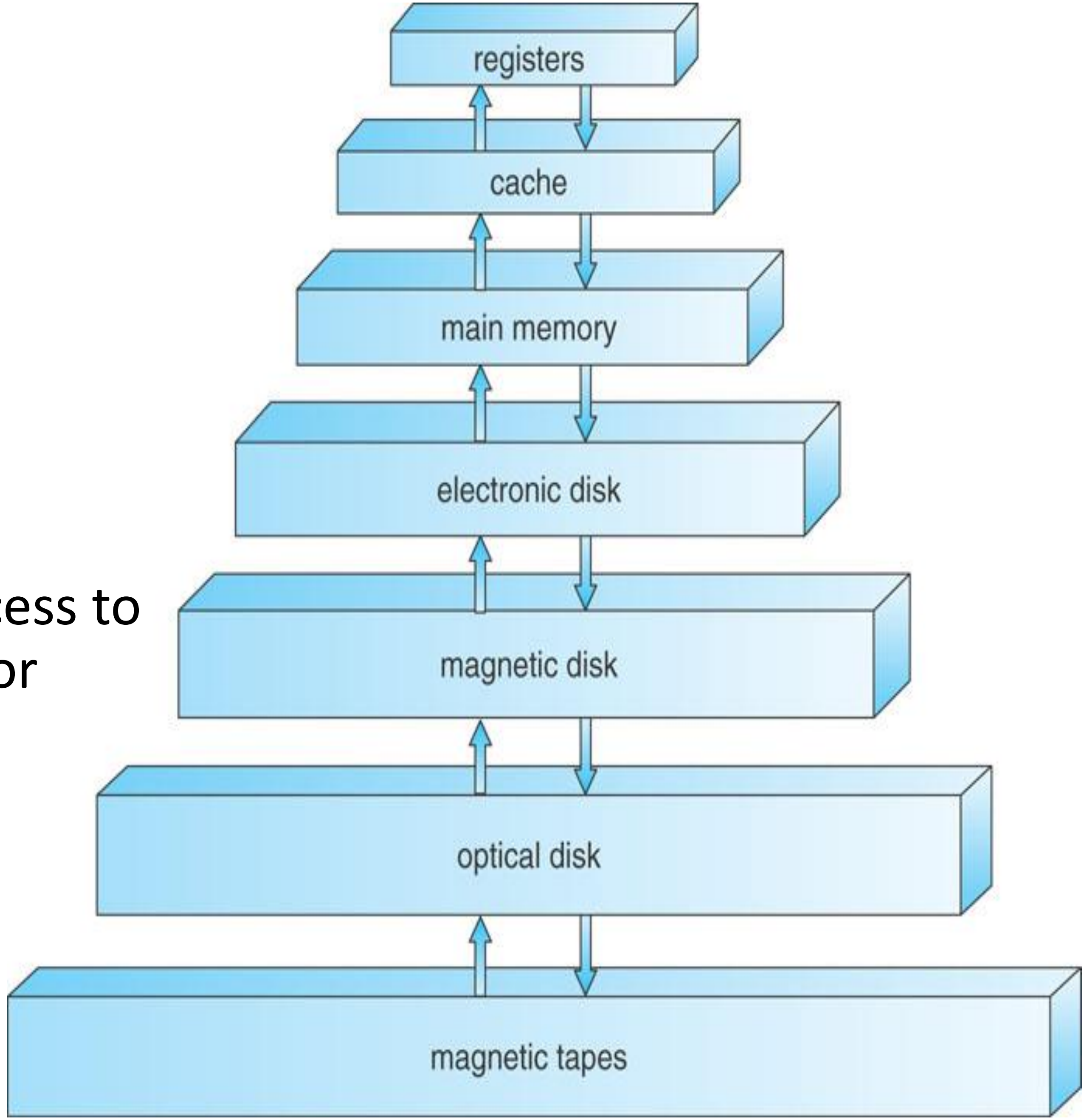
- Main memory
  - Stores data and programs
  - Typically volatile
  - Referred to as real memory or primary memory
- Auxiliary memory (optional)

Memory Hierarchy

- Design constraints on a computer’s memory
  - How much? How fast? How expensive?
- If the capacity is there, applications will likely be developed to use it
- Memory must be able to keep up with the processor
- Cost of memory must be reasonable in relationship to the other components

Memory Hierarchy

- Going down the hierarchy
  - Decreasing cost per bit
  - Increasing capacity
  - Increasing access time
  - Decreasing frequency of access to the memory by the processor



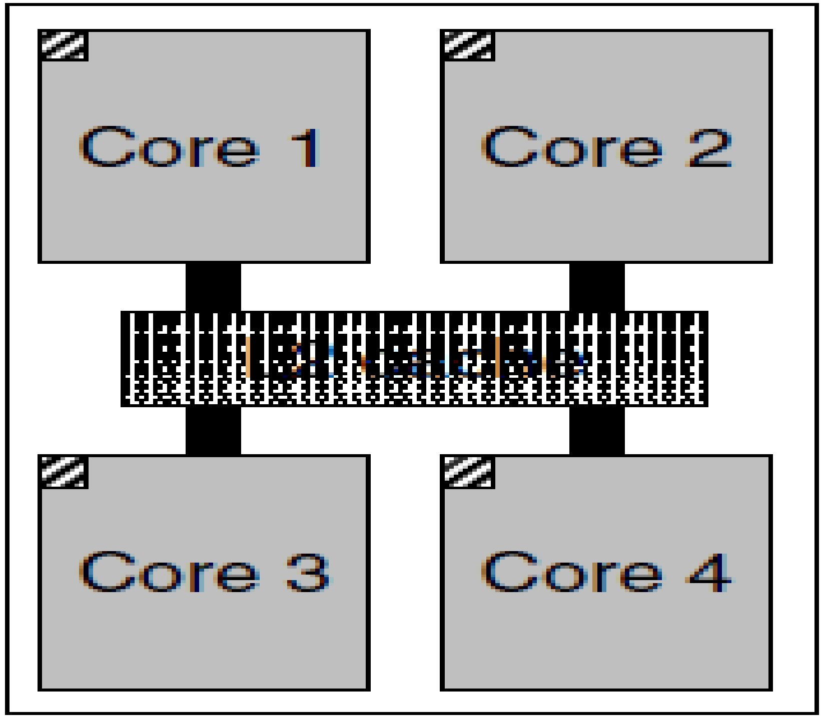
Caching

- 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

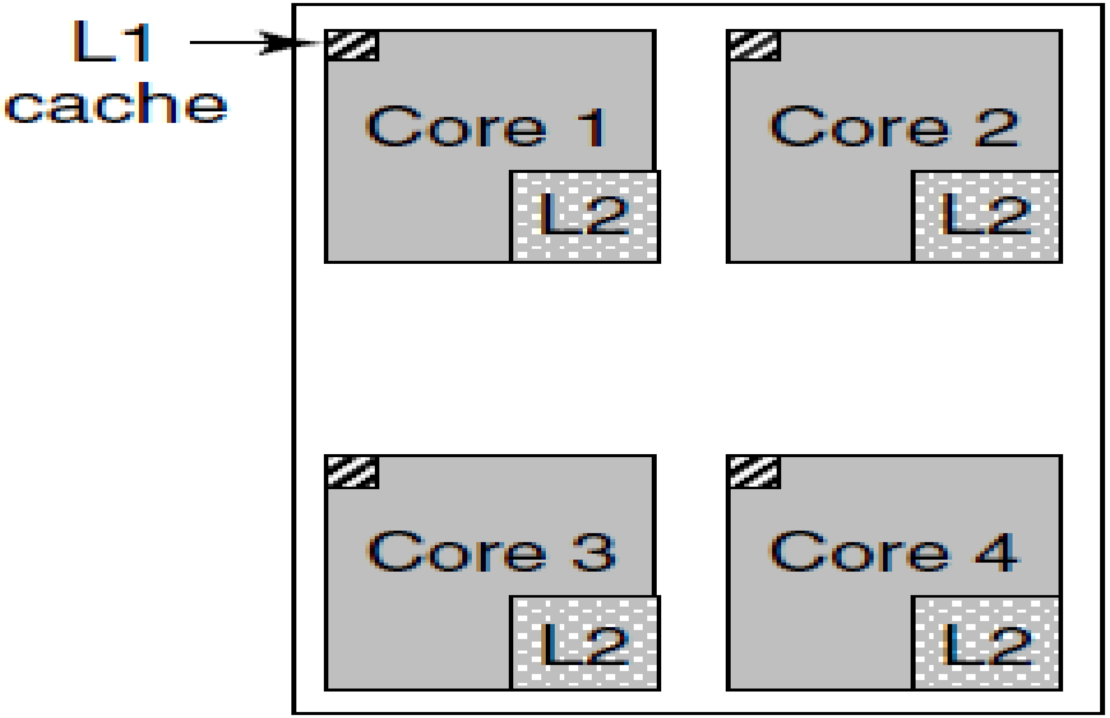
Cache Memory

- Invisible to the OS
- Interacts with other memory management hardware
- Processor must access memory at least once per instruction cycle
- Processor execution is limited by memory cycle time
- Exploit the principle of locality with a small, fast memory

Memory Organization



a) Quad-core chip with shared L2 caches



b) Quad-core chip with separate L2 caches

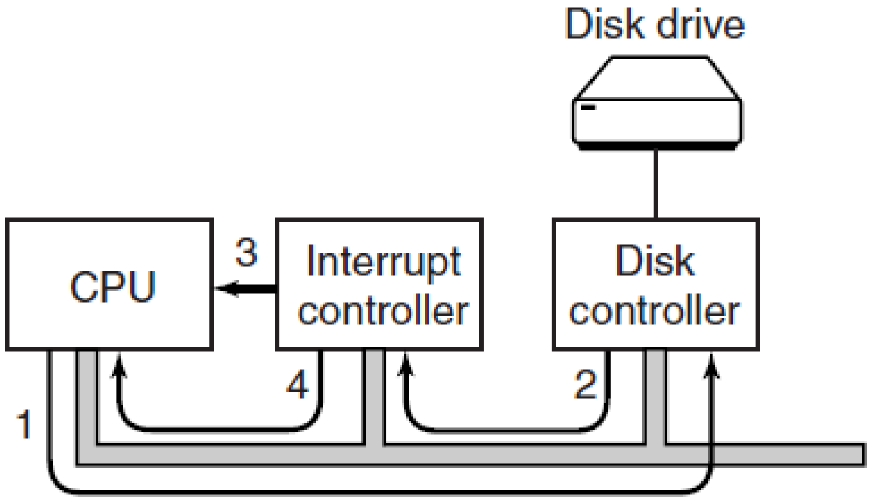
I/O Modules

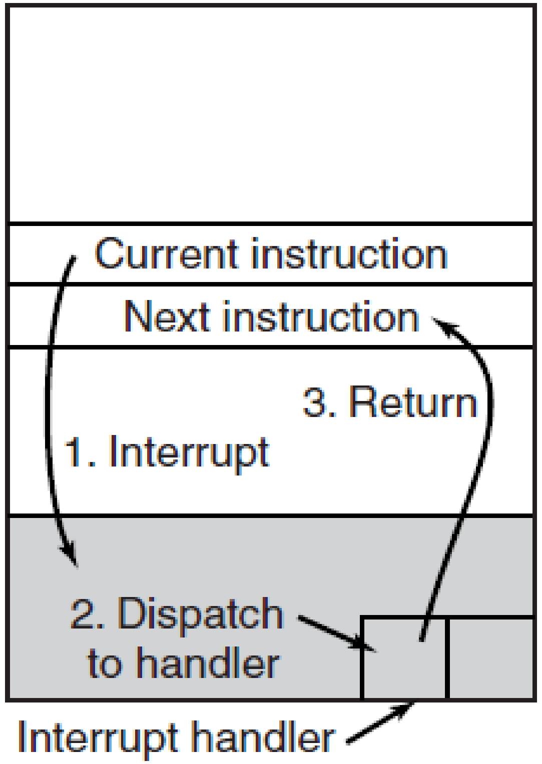
- Move data between computer and its external environment
  - Secondary devices (e.g., disks)
  - Communication equipment
  - Terminals
- When the processor encounters an instruction relating to I/O
  - It executes that instruction by issuing a command to the appropriate I/O module
- Three techniques are possible:

Programmed I/O (busy waiting)	Interrupt Driven I/O	Direct Memory Access DMA

Interrupt-Driven I/O

- Mechanism used to interrupt the normal sequencing of the processor
- Provided to improve processor utilization
    - Most I/O devices are slower than the processor
    - Processor must pause to wait for device
    - Wasteful use of the processor





Direct Memory Access - DMA

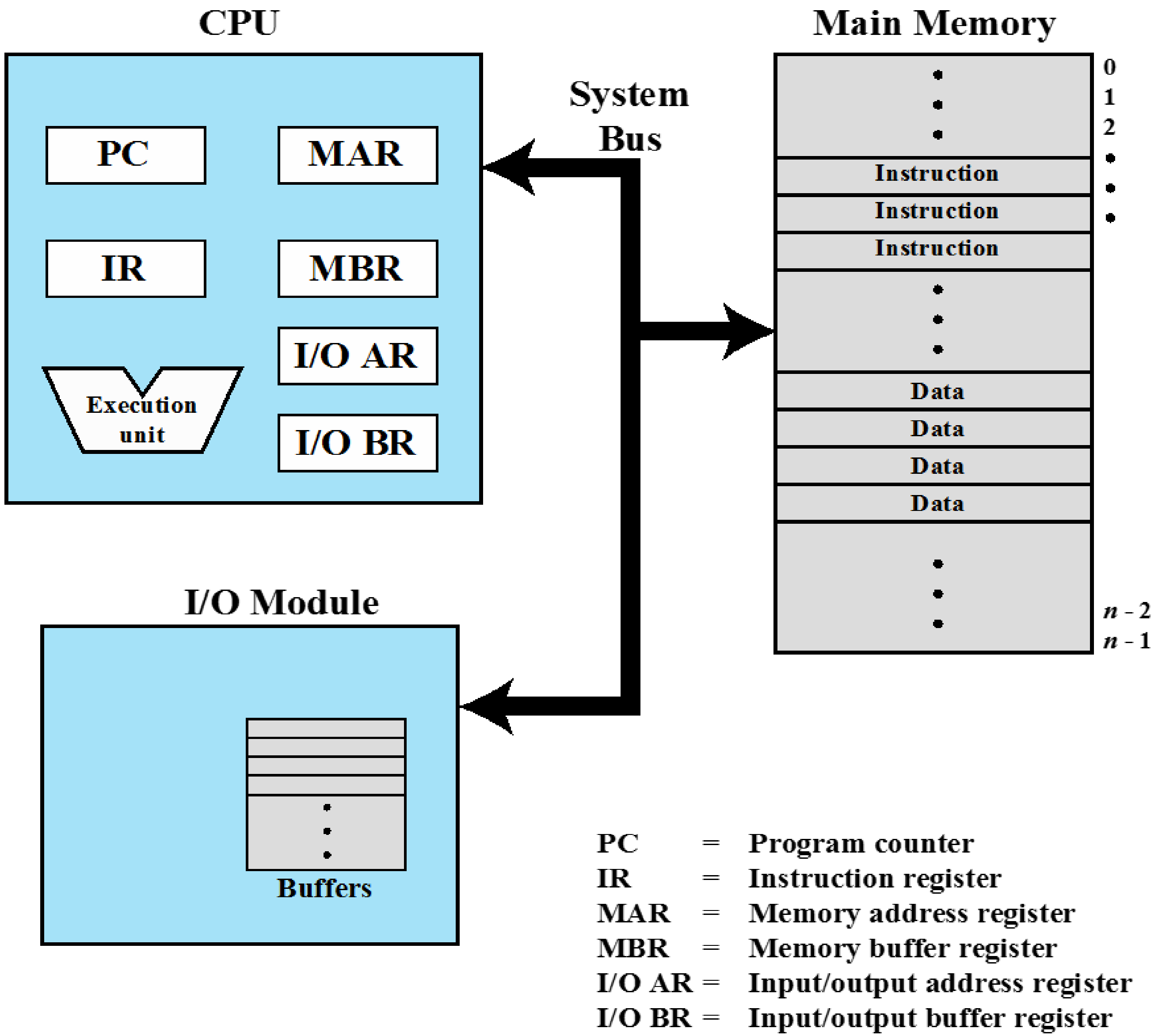
- Performed by a separate module on the system bus or incorporated into an I/O module
- When the processor wishes to read or write data, it issues a command to the DMA module containing:
  - Type of request (read/write)
  - Address of the I/O device involved
  - Starting location in memory to read/write
  - Number of words to be read/written

Direct Memory Access - DMA

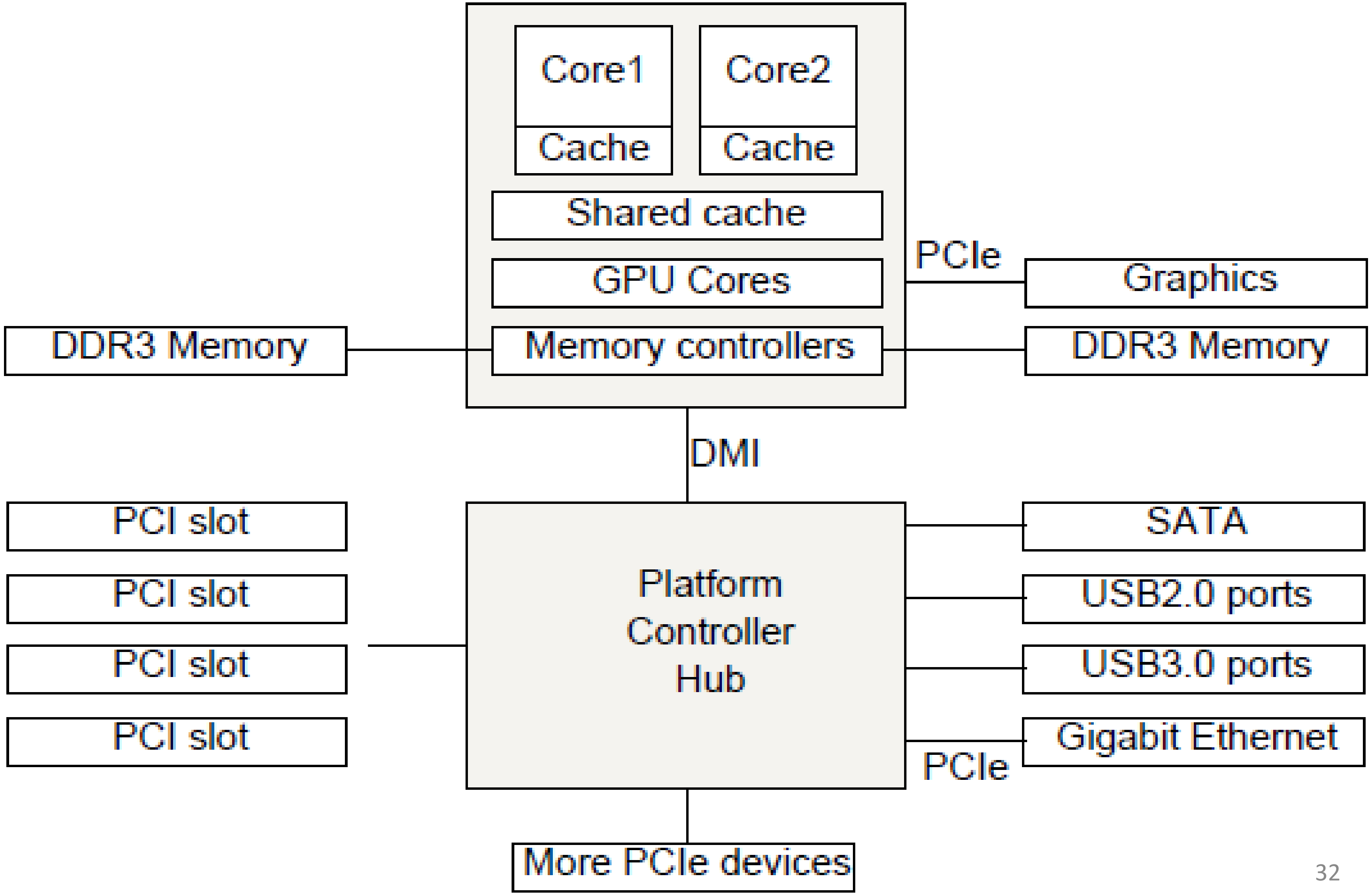
- Entire block of data is transferred directly to/from memory
  - Processor involved only at the beginning and end of the transfer
  - Processor executes more slowly during a transfer when processor access to the bus is required
- DMA is more efficient than Interrupt-driven or programmed I/O

System Bus

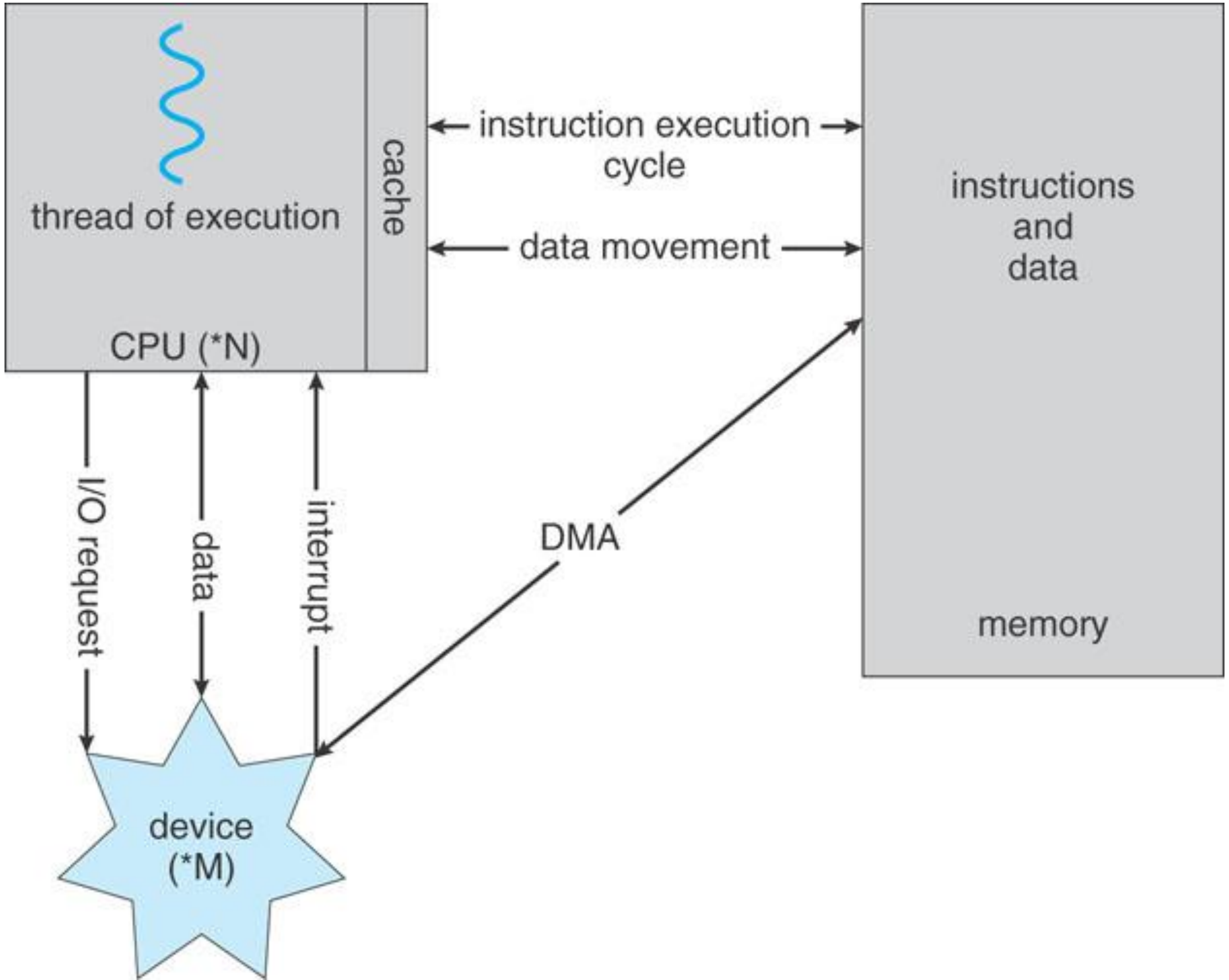
Provides for communication among processors, main memory, and I/o modules



A Large x86 System

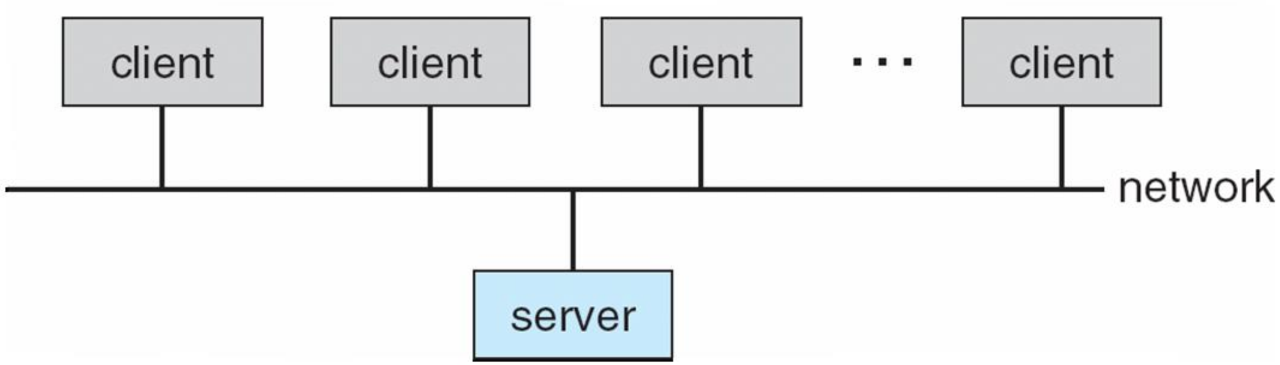


How a Modern Computer Works



Computing Environments

- Traditional computer
  - Office
  - Home
- Client-Server Computing



Computing Environments

- Peer-to-Peer Computing
- Web-Based Computing
  - Servers: load balancers
  - Use of OS has evolved into systems that can be clients and servers
- Open-Source Systems
  - OS available in source-code format rather than just binary
  - Started by Free Software Foundation (FSF), GNU Public Licence (GPL)
  - Examples: GNU/Linux, BSD Unix (including core of Mac OS X), and Sun Solaris

Operating System Zoo

- Mainframes
  - Linux
- Server OS
  - Solaris, Linux, FreeBSD, Windows server 201x
- Multiprocessor OS
  - Parallel computers
  - iOS, Windows
- Handheld
  - PDAs, smartphones
  - iOs, Android
- Embedded
  - Microwave, TV, Car (QNX)
- Sensor Nodes
  - Event driven (TinyOS)
- Real-Time OS
  - Cars, robots, avionics, military
- Smart Card OS

Next Lecture

- Next lecture
  - Key Abstractions: Process, Memory and Files
  - OS Architecture
  - System calls
  - Java Virtual Machine
  - Textbook Reading: 1.4-1.8, 2.3-2.10