Amirkabir University of Technology
(Tehran Polytechnic)

Operating Systems

Introduction

Seyyed Ahmad Javadi

sajavadi@aut.ac.ir

Fall 2021

My Background and Contact Details

- Seyyed Ahmad Javadi
- PhD from New York State University at Stony Brook
- Postdoc from University of Cambridge
- Interest: Cloud computing, operating systems, performance analysis

- Office: CE department, 3rd floor
- Email: sajavadi@aut.ac.ir
  - Include CE303 in your email subject
- Home page: https://ce.aut.ac.ir/~sajavadi/

Course Introduction

- Saturday and Monday (13:30-15)
  - Attend class on time
- Course web page
  - Check the webpage on regular basis
  - Everything will be posted on CW
  - Post All your Questions on CW Forums
    - Check forum history before posting any question
- Office hours and TA classes
  - TBD

Textbook

- **Operating System Concepts**, 10th Edition, Wiley publishing
  - By A. Silberschatz, P. Galvin, & G. Gagne

- Other References:
  - Operating systems: design & implementation,
    - By A. Tanenbaum and A. Woodhull, 3rd edition, 2006.
  - Operating systems: internals and design principles,
    - By W. Stallings, 5th edition, 2005.

Grading

Section	Score	Considerations
assignments	2.5	five homework
midterm exam	4	**1400/08/22**
project	4 + 1	in three phases
final exam	8	1400/10/20
quiz	1	two quizzes
class participation	0.5	ask/answer questions be active in the course webpage

Harsh penalty for plagiarism and cheating

Project

- Adding new features to XV6 created in MIT's Operating System Engineering course; isn't this exciting ☺
  - XV6 is used in most of the well-known universities.
  - https://pdos.csail.mit.edu/6.828/2012/xv6.html
- **Three Phases:**
  - Phase 1: getting to know XV6 basics (solo work)
  - Phase 2: getting to know XV6 advanced features (solo work)
  - Phase 3: final project (teamwork)

Syllabus

- Introduction to operating systems
- Process management
  - Threads
  - Synchronization
  - Scheduling
- Memory management
- Storage management
- Protection and security

Copyright Notice

Slides are based on the slides of the main textbook.

Silberschatz

https://www.os-book.com/OS10/slide-dir/index.html

Part 1

What is an Operating System?

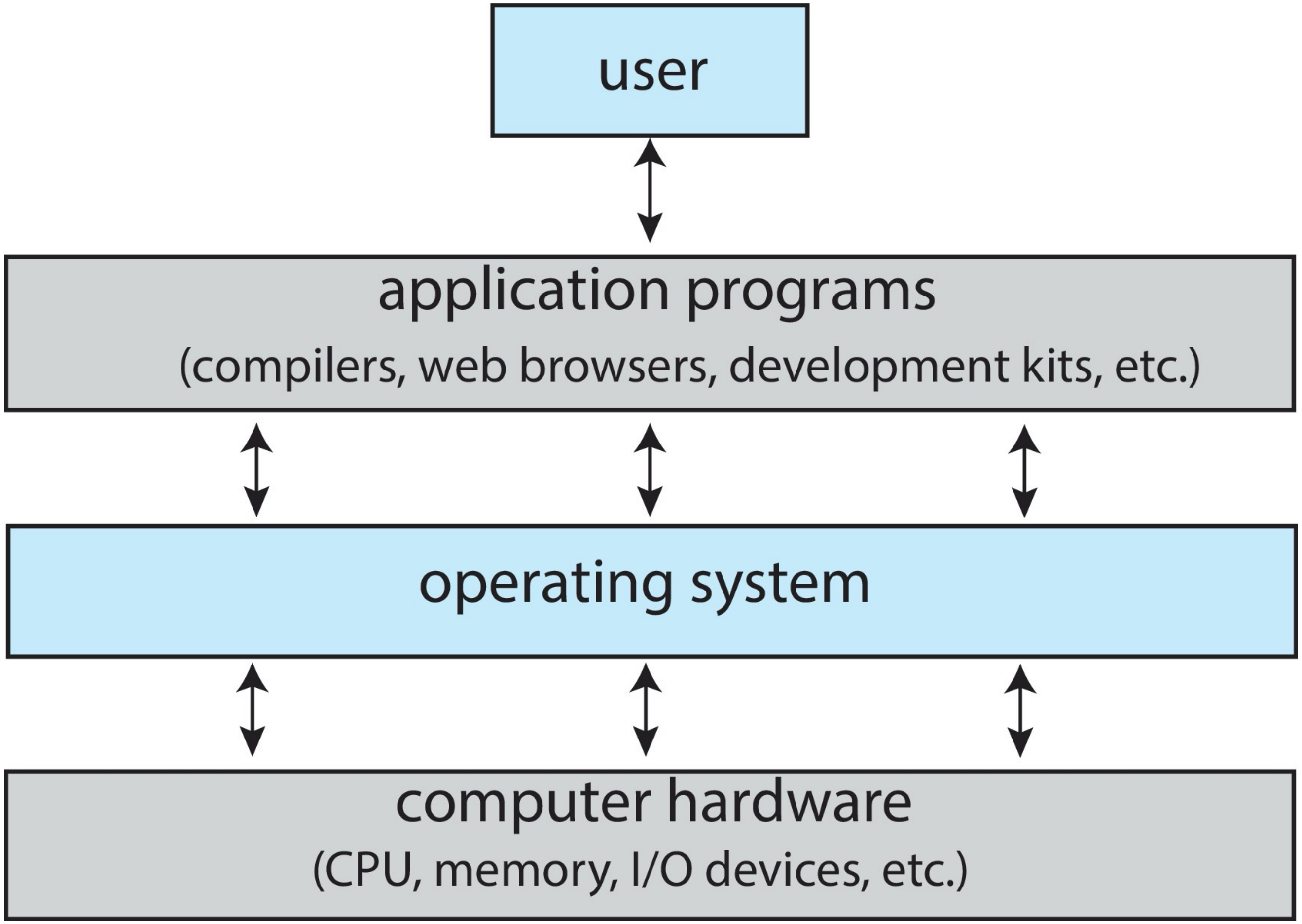
- A program that acts as an intermediary between a user of a computer and the computer hardware.
  - User can execute programs conveniently & efficiently

- 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.

OS: Mandatory or Optional?

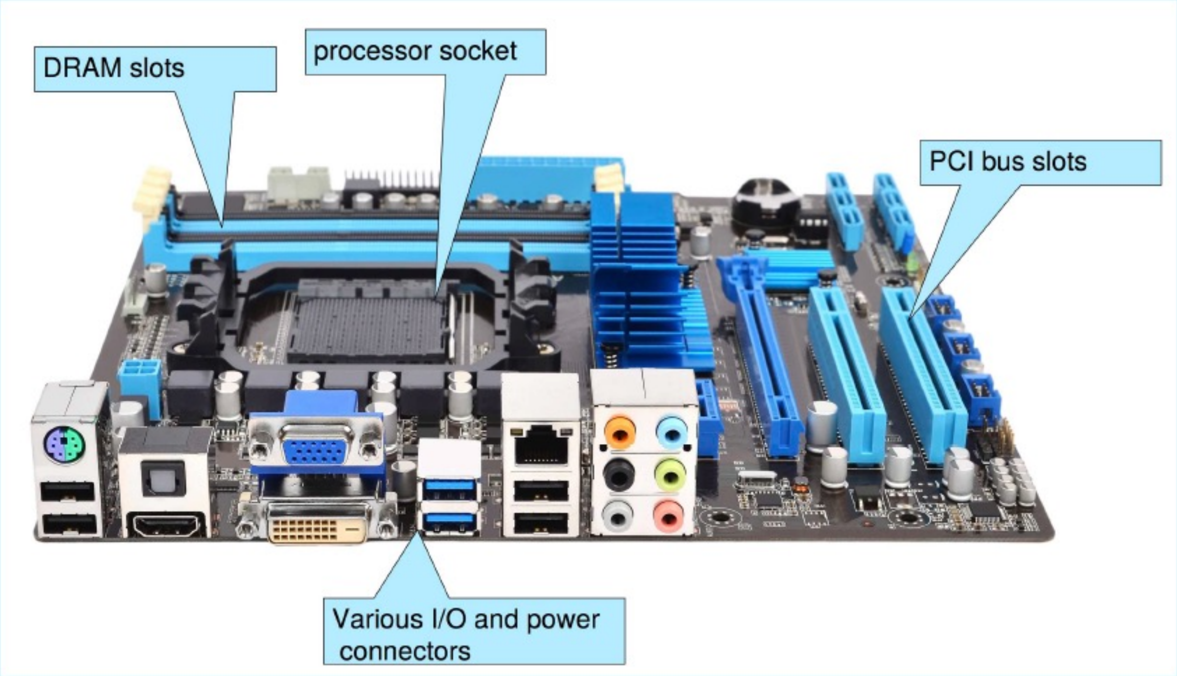
- **Can we run a computer without an operating system?**
  - Yes, earliest computers did not have OS.
- **What does a compute without an OS look like?**
  - Machines tasked with one program at a time.
    - Cannot read a pdf while listening to a music.
  - Each program has a lot of work to do.
    - Where to load a program
    - IO access

Abstract View of Components of Computer


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

PC Motherboard

Consider the desktop PC motherboard with a processor socket shown below:



This board is a fully-functioning computer, once its slots are populated. It consists of a processor socket containing a CPU, DRAM sockets, PCIe bus slots, and I/O connectors of various types. Even the lowest-cost general-purpose CPU contains multiple cores. Some motherboards contain multiple processor sockets. More advanced computers allow more than one system board, creating NUMA systems.

Operating System Story

- **Vital goal of a computer system**
  - Execute user program and make solving user problem easier.
- **Shall user program use hardware directly?**
  - Hardware alone is ***not easy to use.***
  - Application programs require certain ***common operations.***
    - Example: I/O operations

Common functions of controlling and allocating resources brought together into one piece called **OS**

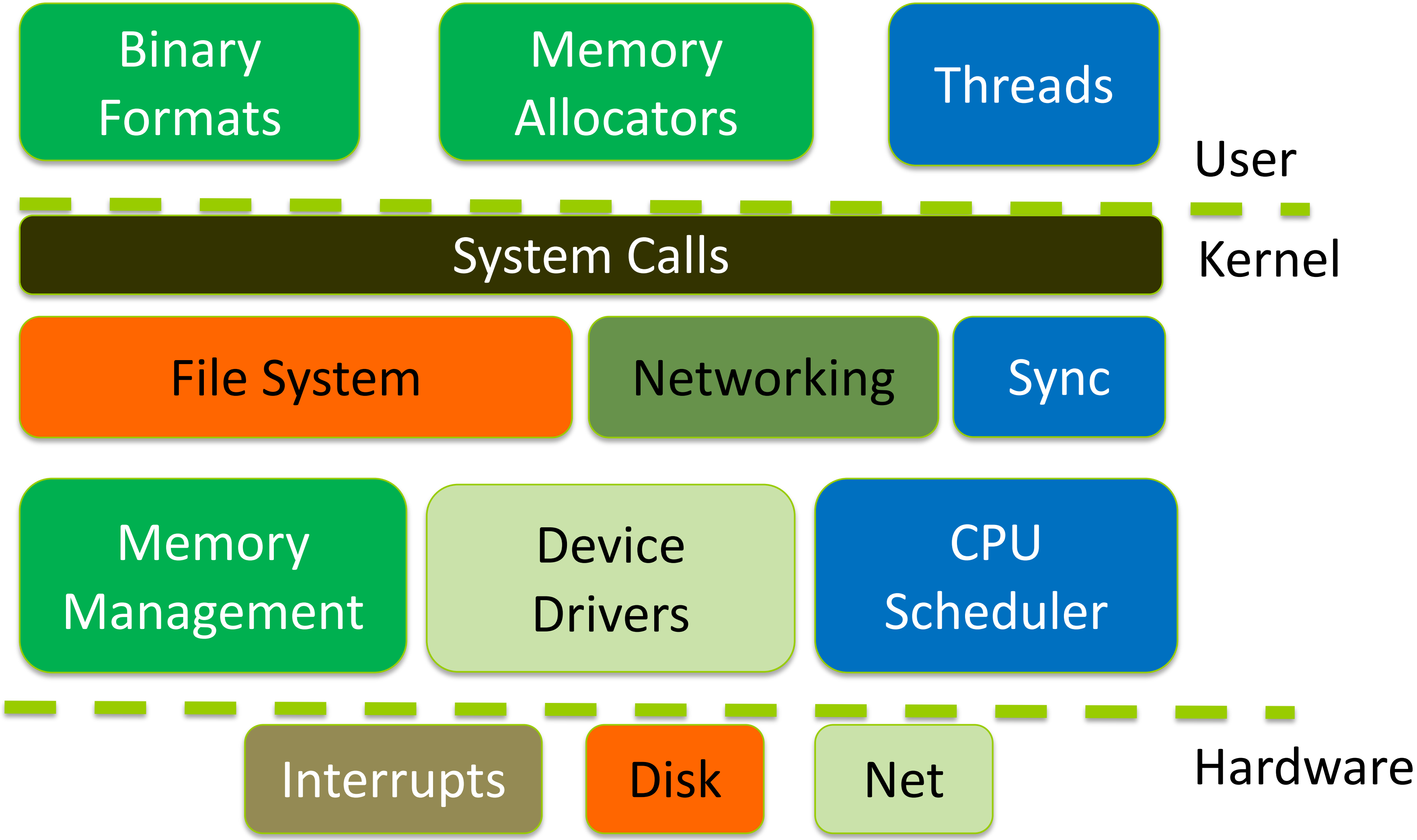
Operating System Definition (cont.)

- No universally accepted definition.

- “The one program running at all times on the computer” is the **kernel,** part of the operating system.

- Everything else is either
  - A ***system program*** (ships with the operating system, but not part of the kernel) , or
  - An ***application program***, all programs not associated with the operating system.

A logical view of the OS



Binary Formats

Memory Allocators

Threads

User

System Calls

Kernel

File System

Networking

Sync

Memory Management

Device Drivers

CPU Scheduler

Interrupts

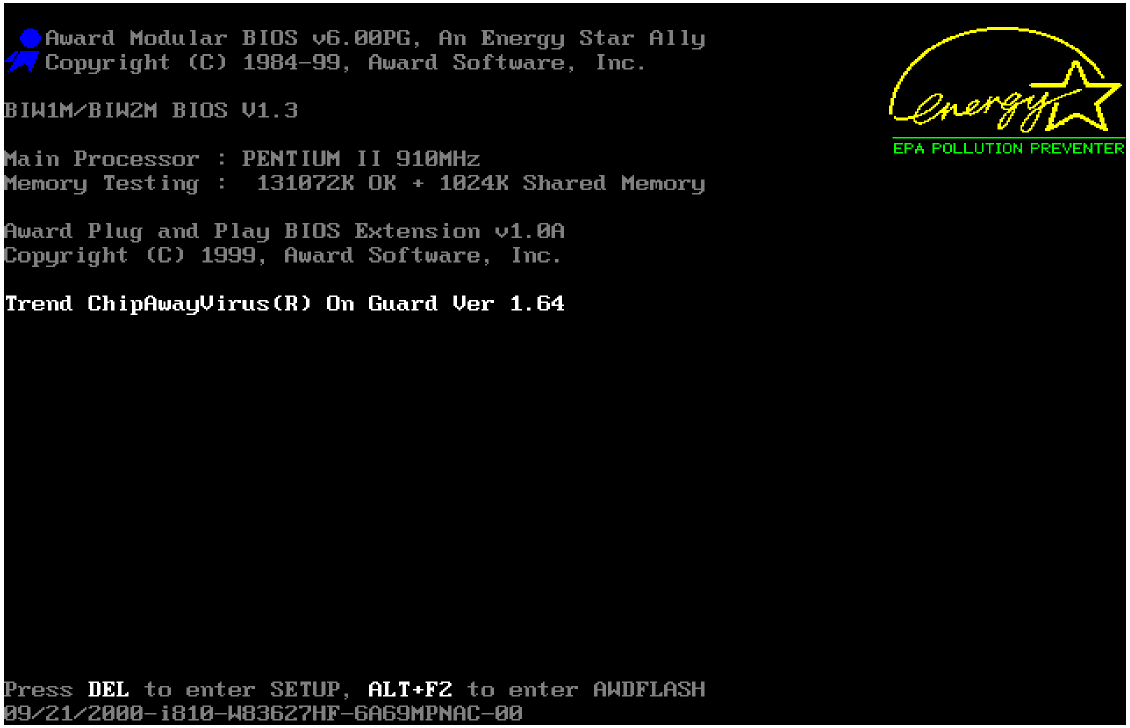
Disk

Net

Hardware

Computer Startup

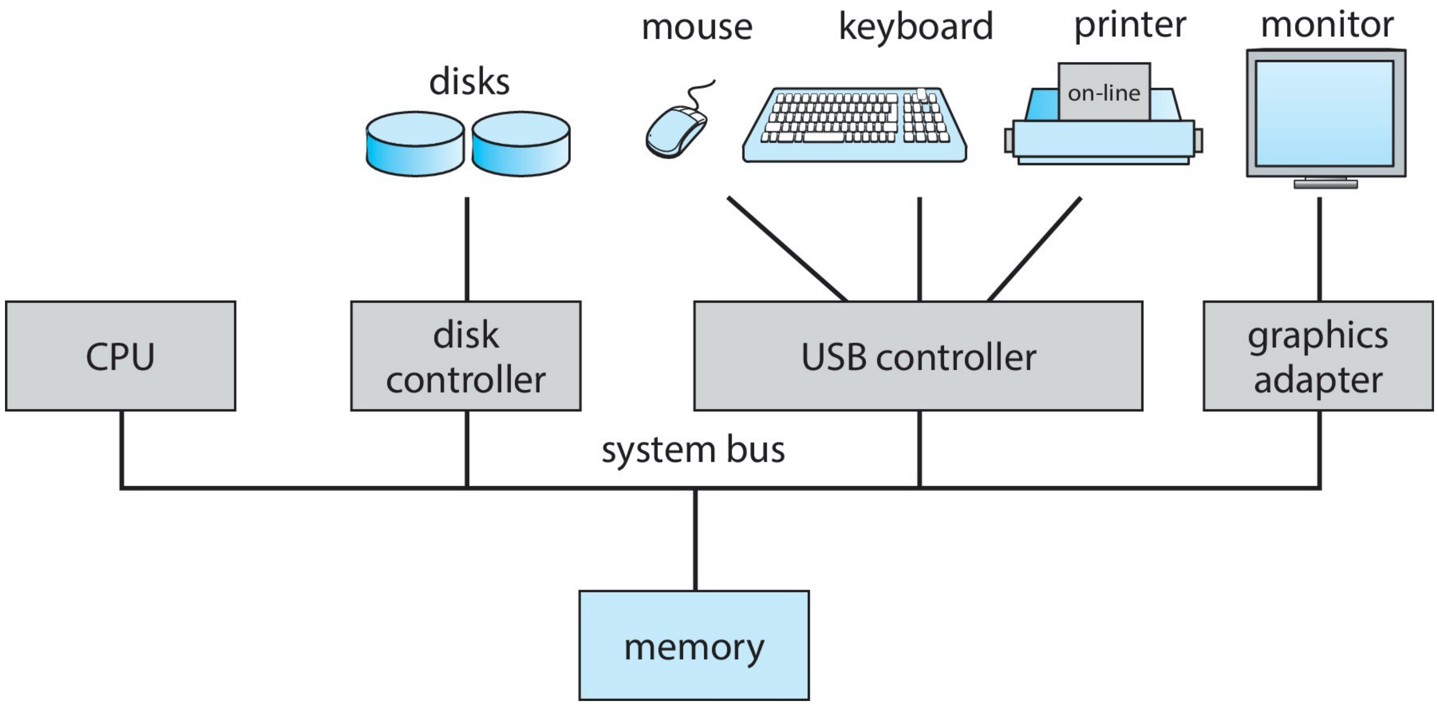
- **Bootstrap program** is loaded at power-up or reboot.
  - Typically stored in ROM or EPROM, generally known as **firmware.**
  - Initializes all aspects of system.
  - Loads operating system kernel and starts execution.



```
  Award Modular BIOS v6.00PG, An Energy Star Ally
  Copyright (C) 1984-99, Award Software, Inc.

BIW1M/BIW2M BIOS V1.3

Main Processor : PENTIUM II 910MHz
Memory Testing :  131072K OK + 1024K Shared Memory

Award Plug and Play BIOS Extension v1.0A
Copyright (C) 1999, Award Software, Inc.

Trend ChipAwayVirus(R) On Guard Ver 1.64













Press DEL to enter SETUP, ALT+F2 to enter AWDFLASH
09/21/2000-i810-W83627HF-6A69MPNAC-00
```

Computer System Organization

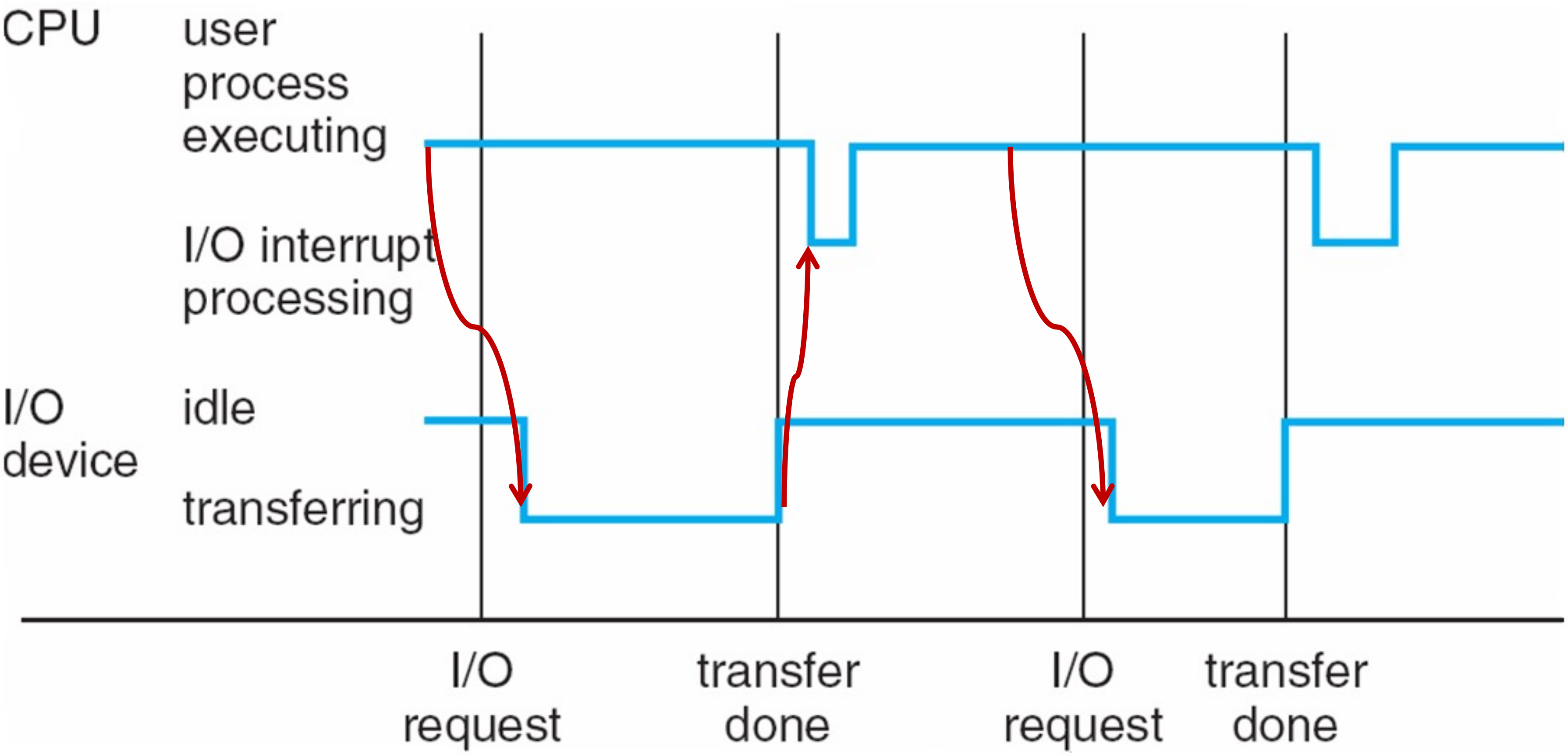
- Computer-system operation
  - One or more CPUs, device controllers connect through common **bus** providing access to shared memory.
  - Concurrent execution of CPUs and devices competing for memory cycles.



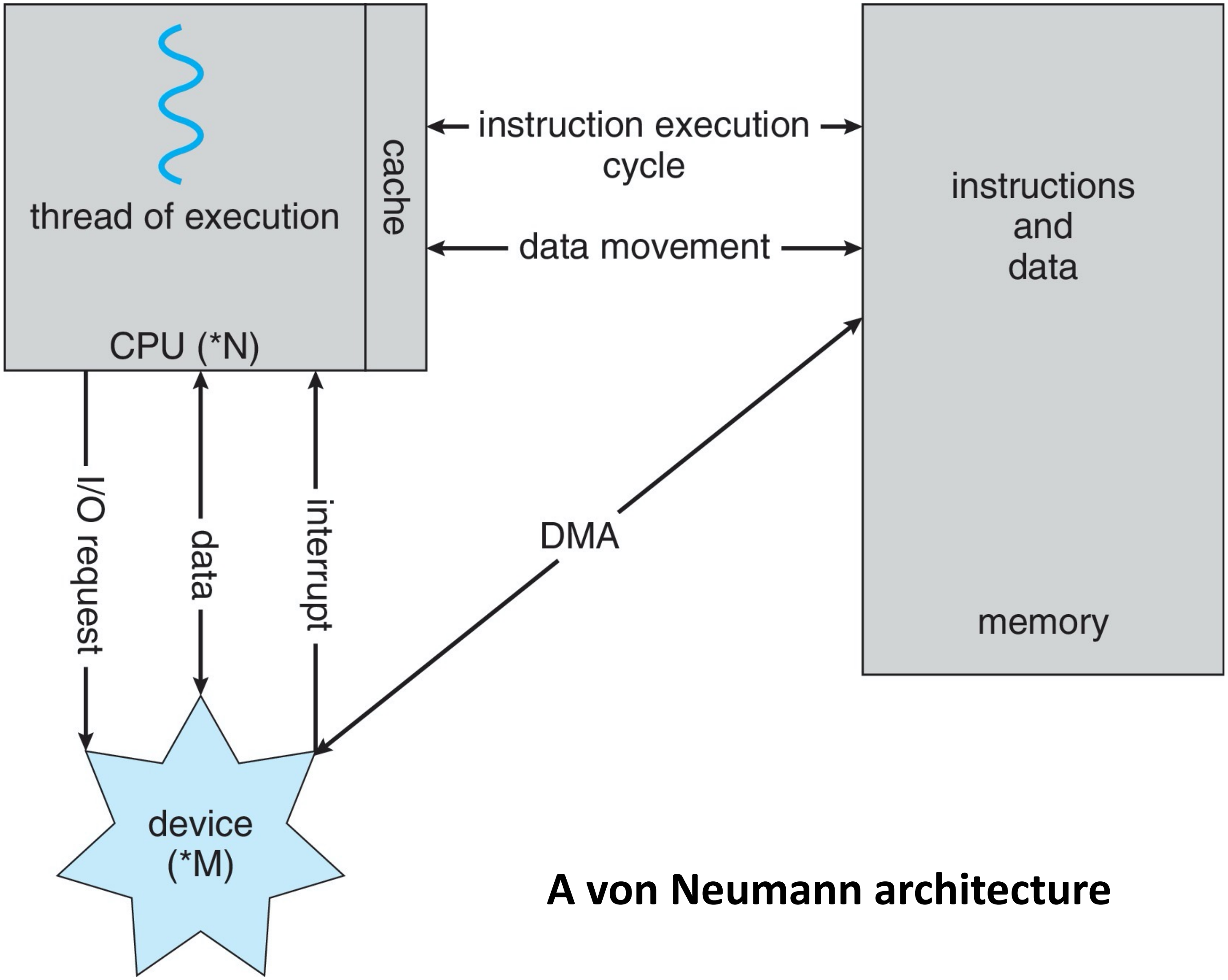
Computer-System Operation

- Each device controller is in charge of a particular device type (e.g., disk drives, audio devices).
- Each device controller has a local buffer.
- I/O devices and the CPU can execute concurrently
- I/O: device ←→ local buffer of controller.
- Device controller informs CPU that it has finished its operation by causing an **interrupt.**
- **CPU moves data**
  - Main memory ←→ local buffers

Interrupt Timeline



How a Modern Computer Works



A von Neumann architecture

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.

Multiprogramming (Batch System)

- Single user cannot always keep CPU and I/O devices busy.
- 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 job has to wait (for I/O for example), OS switches to another job.

Multitasking (Timesharing)

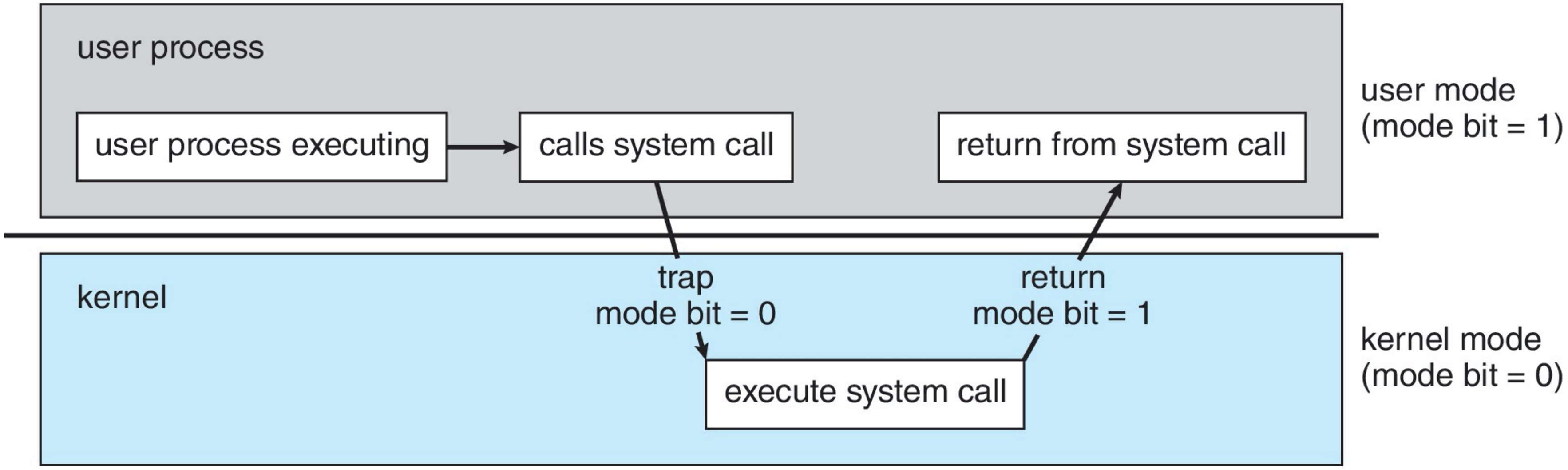
- A logical extension of Batch systems
- The CPU ***switches jobs so frequently*** that users can interact with each job while it is running, creating **interactive** computing.
  - Response time should be < 1 second.
  - Each user has at least one program executing in memory ⇨ process.
  - If several jobs ready to run at the same time ⇨ CPU scheduling.
  - If processes don’t fit in memory, swapping moves them in&out to run.
  - Virtual memory allows execution of processes not completely in memory.

Dual-mode Operation

- **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.
  - When a user is running ⇨ mode bit is “user”.
  - When kernel code is executing ⇨ mode bit is “kernel”.

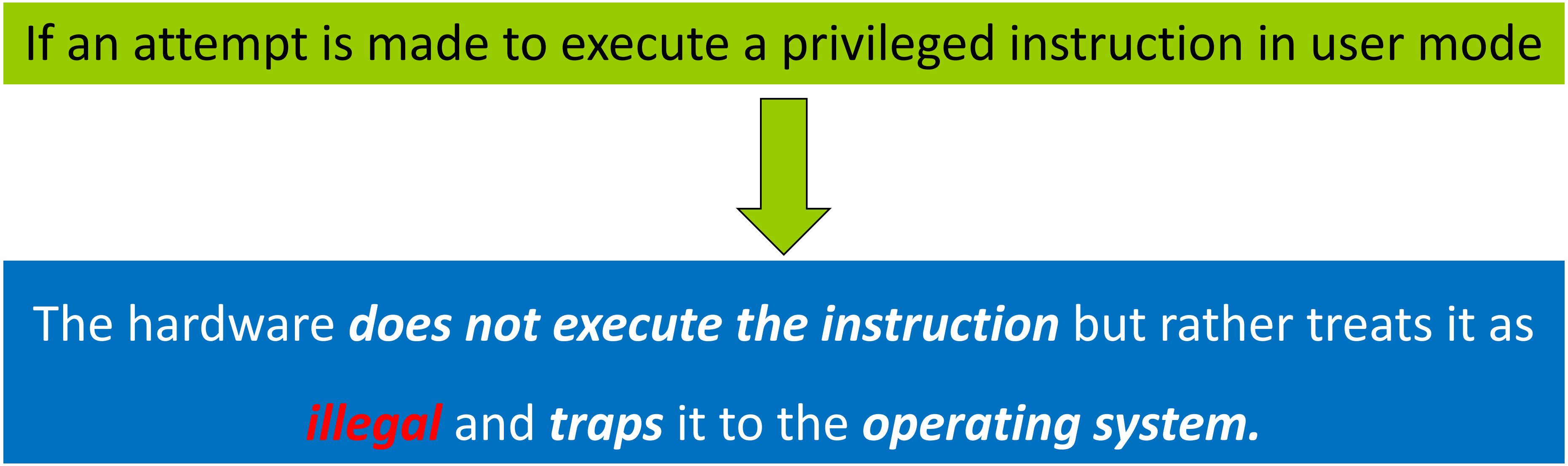
Dual-mode Operation (Cont.)

- How do we guarantee that user does not explicitly set the mode bit to “kernel”?
  - System call changes mode to kernel, return from call resets it to user.



Privileged instructions

- Some instructions designated as **privileged**, **only executable in kernel mode.**
  - Example: I/O control, timer management, and interrupt management



If an attempt is made to execute a privileged instruction in user mode

The hardware ***does not execute the instruction*** but rather treats it as ***illegal*** and ***traps*** it to the ***operating system.***

Questions?