



# **Operating Systems**

## **Computer System Organization**

Seyyed Ahmad Javadi

[sajavadi@aut.ac.ir](mailto:sajavadi@aut.ac.ir)

Fall 2023

# Copyright Notice

---

Slides are based on the slides of the main **textbook**.

**Silberschatz**

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



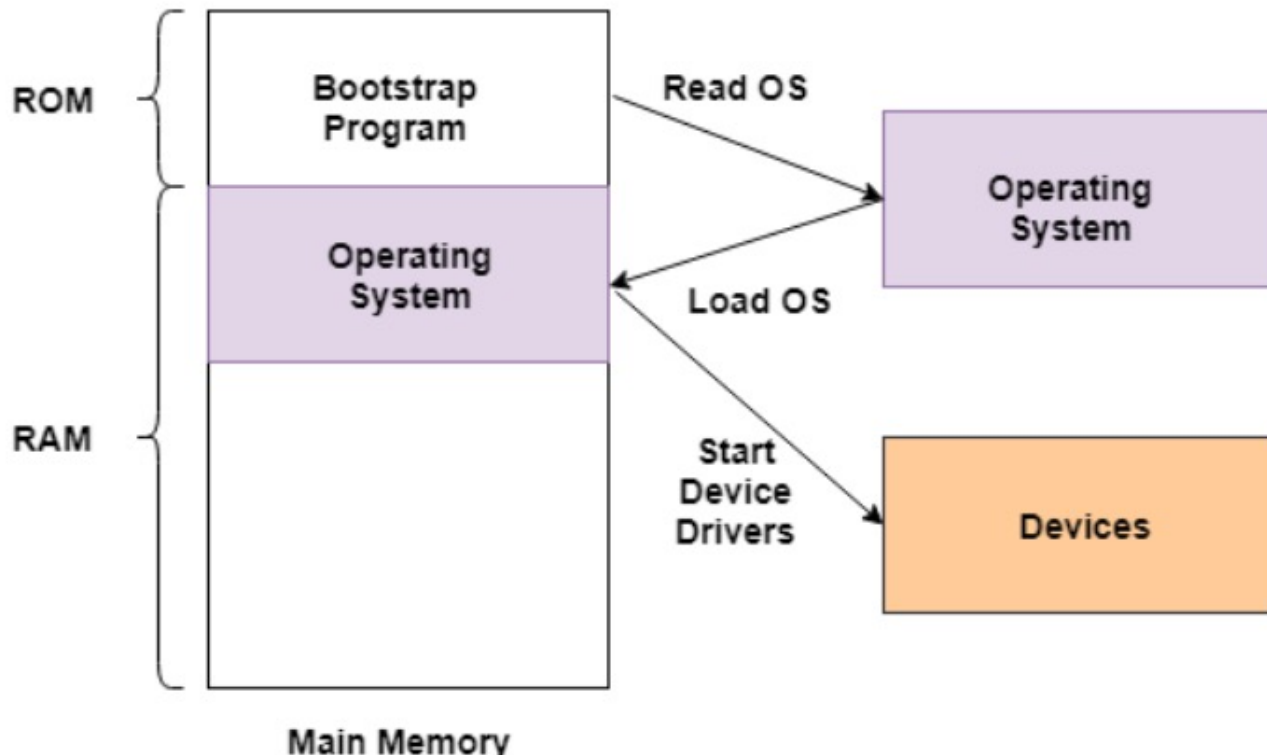
---

# COMPUTER STARTUP



# 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.



<https://www.tutorialspoint.com/what-is-a-bootstrap-program>

# Computer Startup (cont.)

● 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.

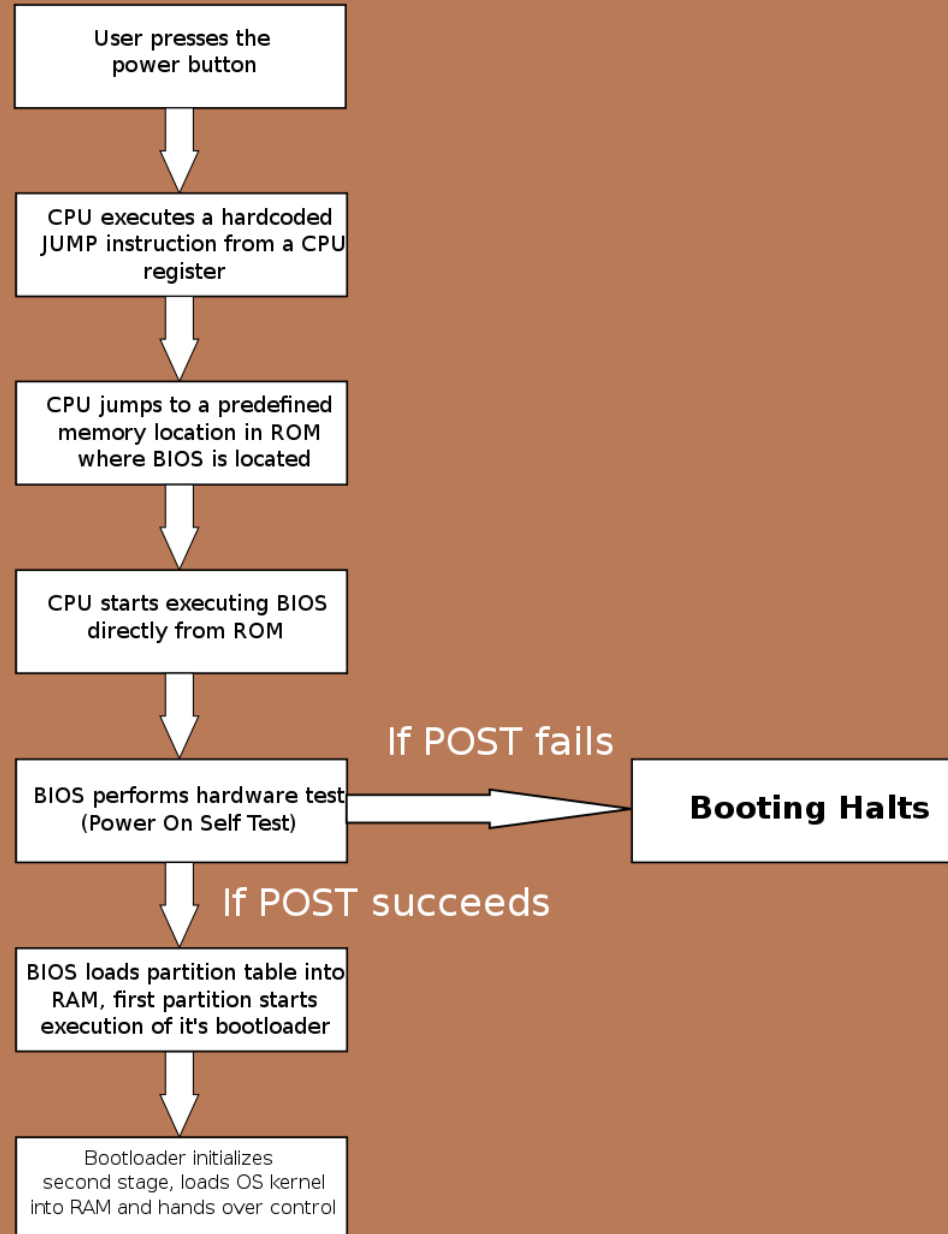
Trend ChipAwayVirus(R) On Guard Ver 1.64



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



# Computer booting sequence



---

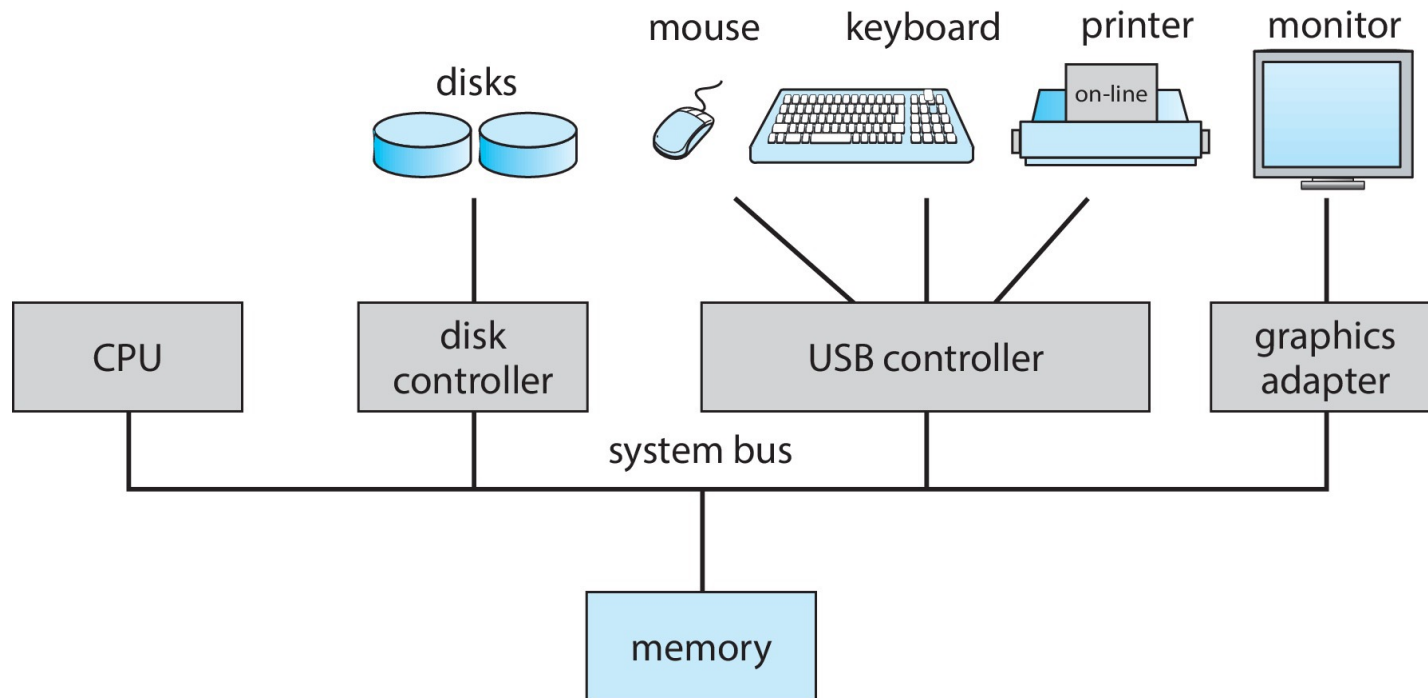
# COMPUTER SYSTEM ORGANIZATION



# Computer System Organization

## ■ Computer-system operation

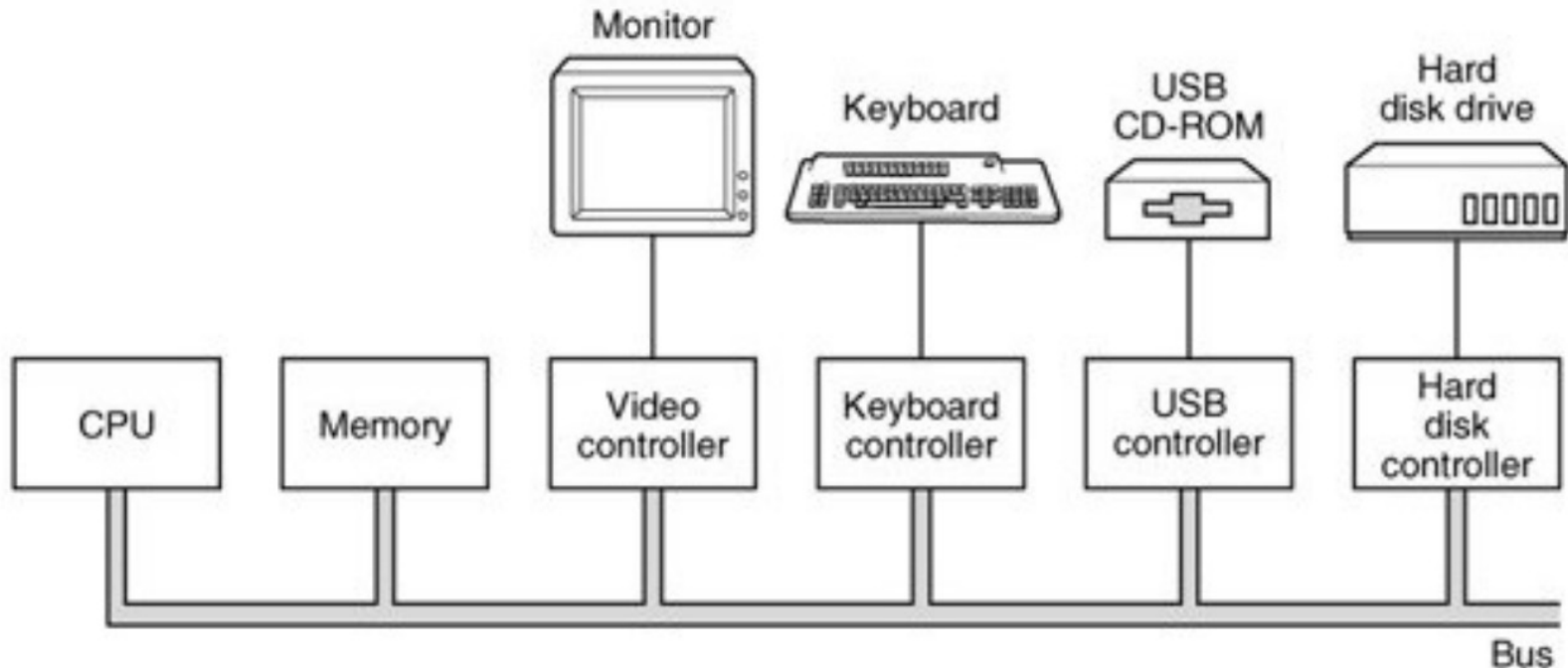
- One or more CPUs, device controllers connect through common **bus** providing access to shared memory.
- Parallel execution of CPUs and devices competing for memory cycles.





# Computer-System Operation

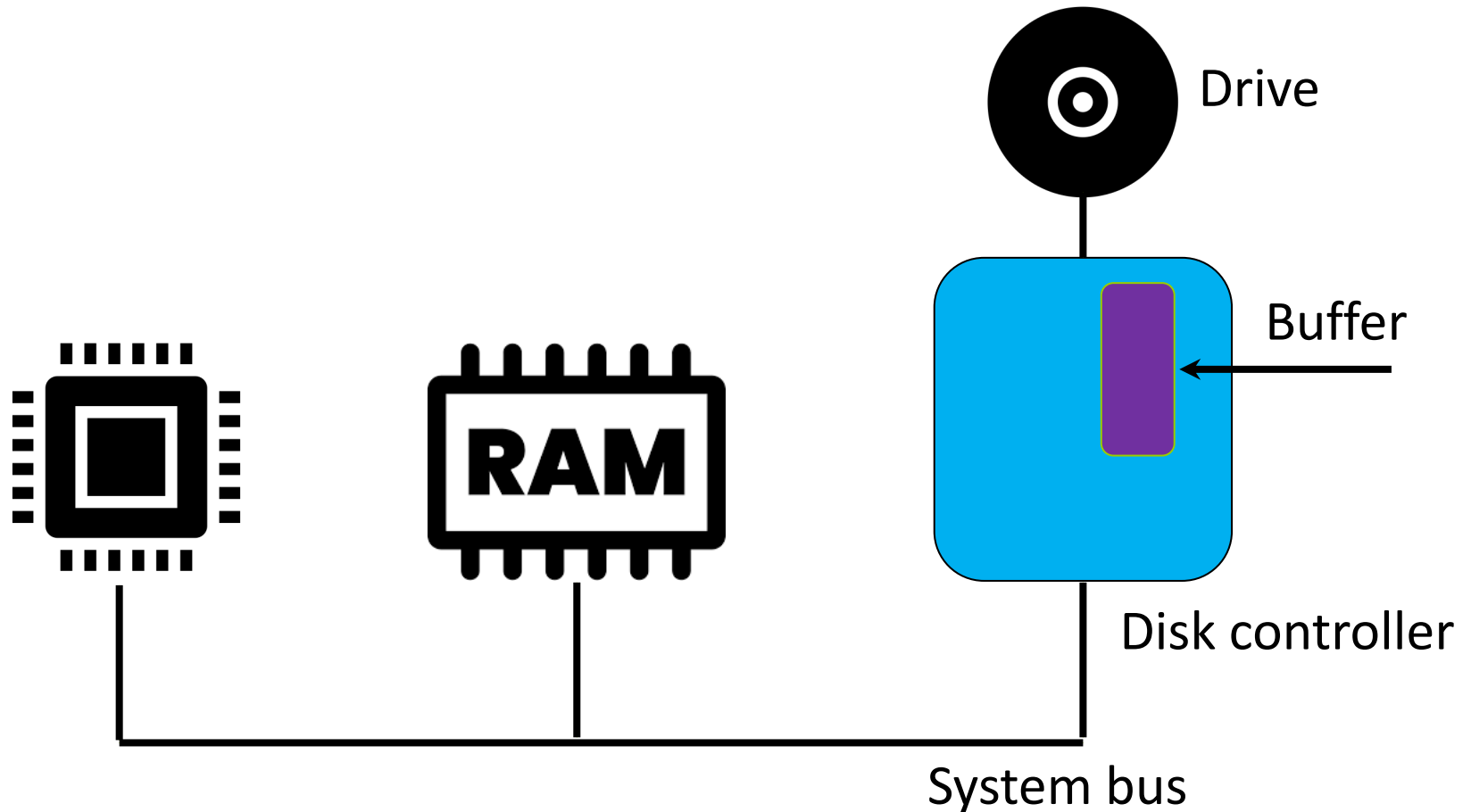
- Each device controller is in charge of a particular device typ.
  - Such as disk drives, audio devices, etc.



[http://www.idc-online.com/technical\\_references/pdfs/information\\_technology/Device\\_Controllers\\_Memory\\_Mapped\\_and\\_Port\\_Mapped.pdf](http://www.idc-online.com/technical_references/pdfs/information_technology/Device_Controllers_Memory_Mapped_and_Port_Mapped.pdf)

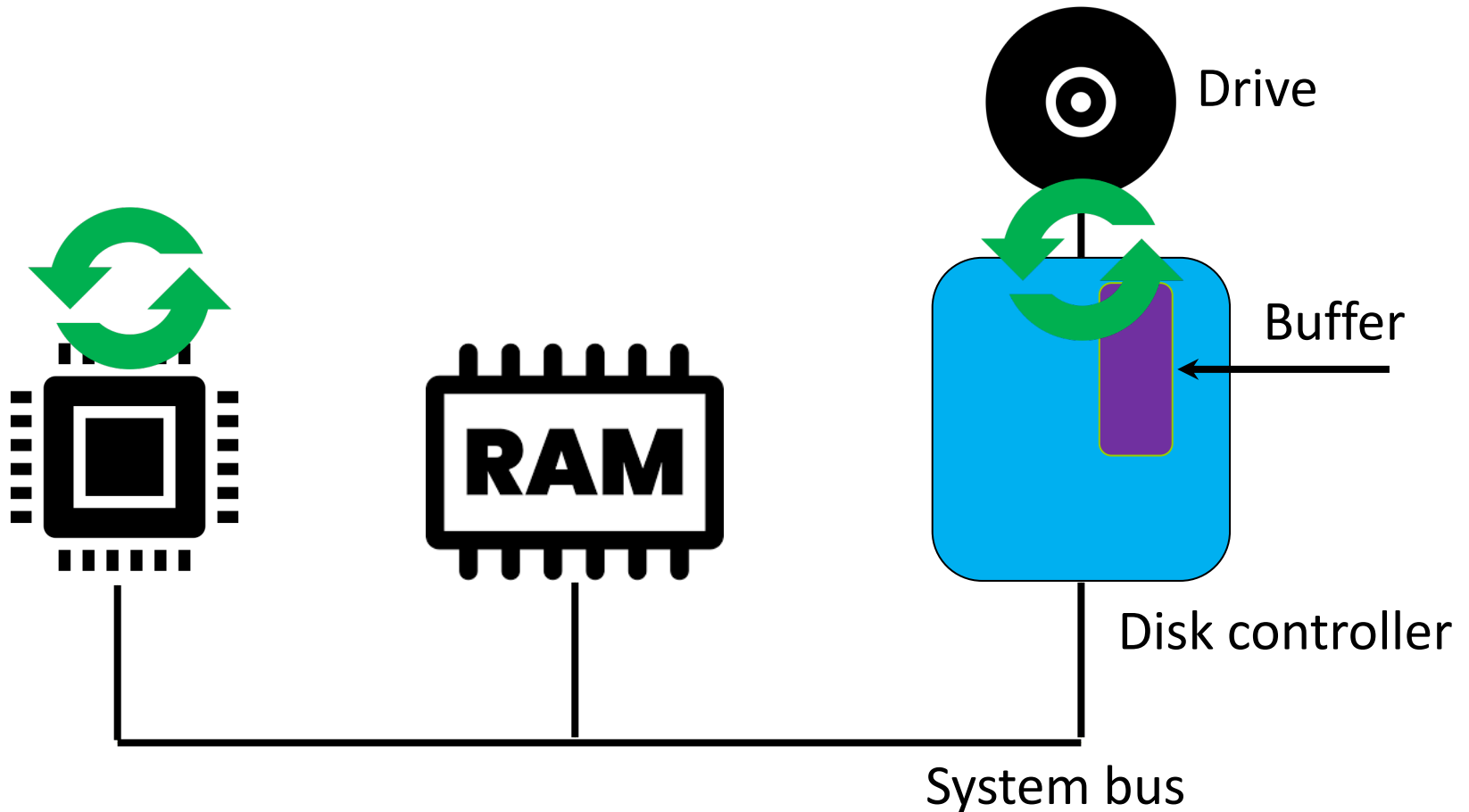
# Computer-System Operation (cont.)

- Each device controller has a local buffer.



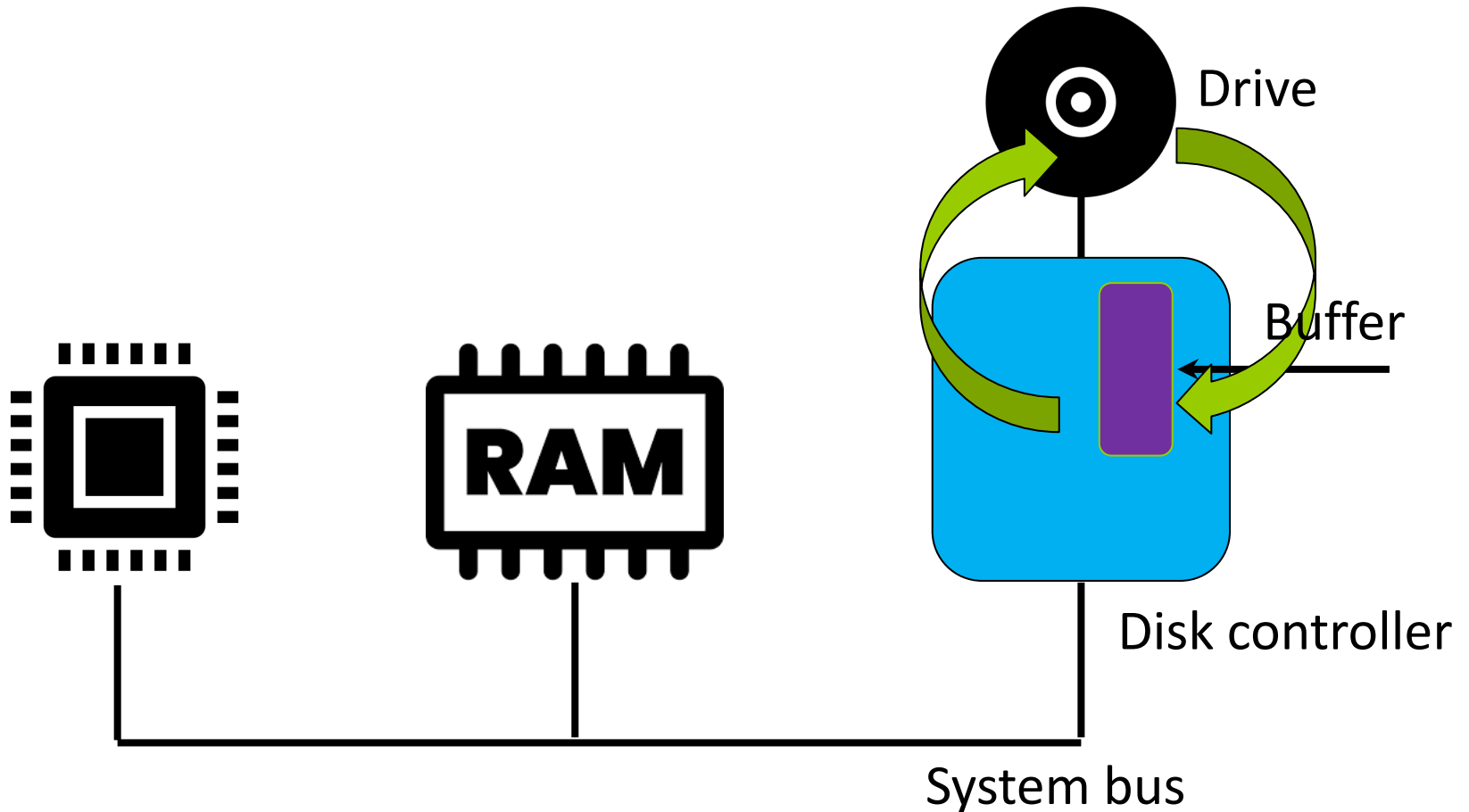
# Computer-System Operation (cont.)

- I/O devices and the CPU can execute in parallel



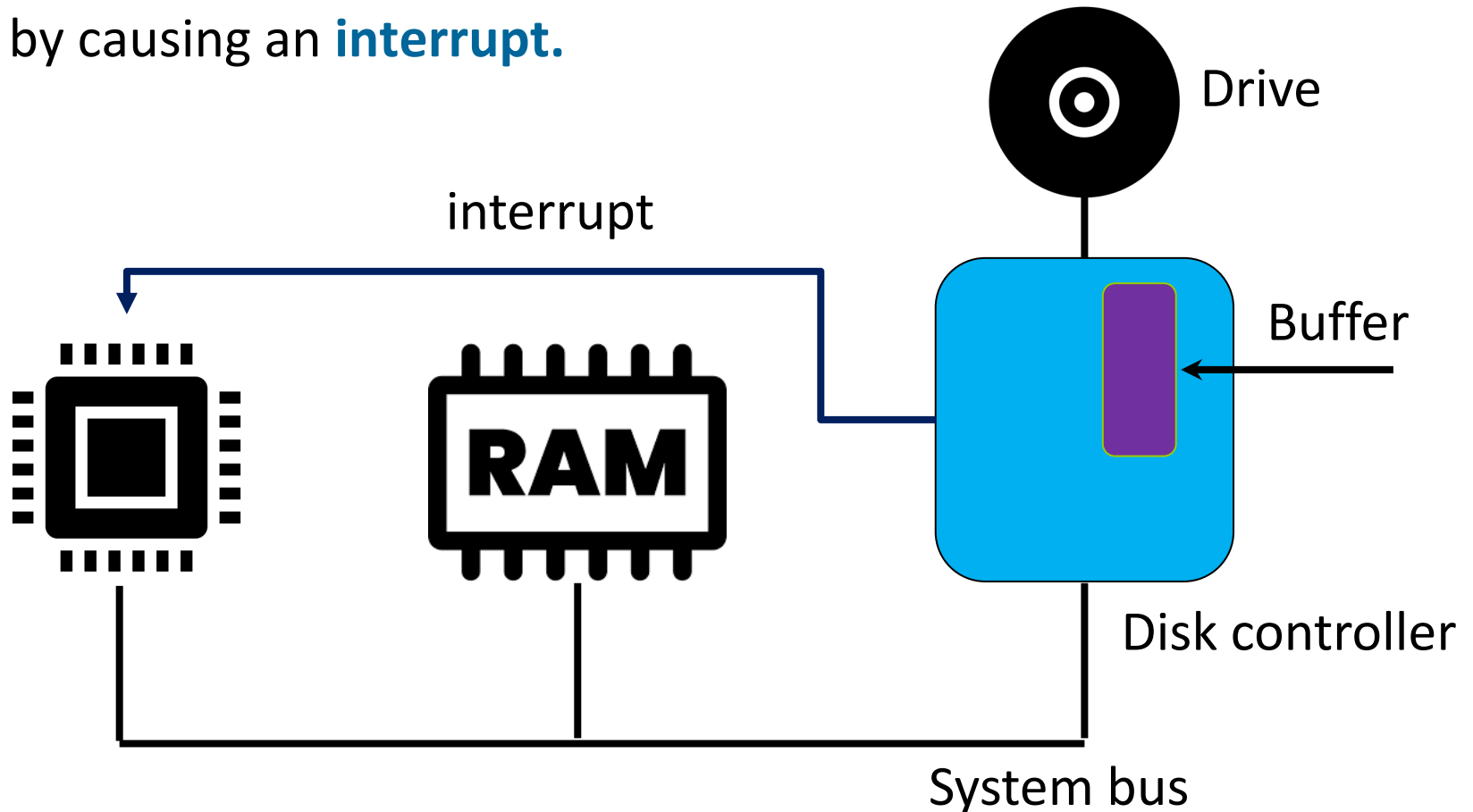
# Computer-System Operation (cont.)

- I/O: device  $\leftrightarrow$  local buffer of controller.



# Computer-System Operation (cont.)

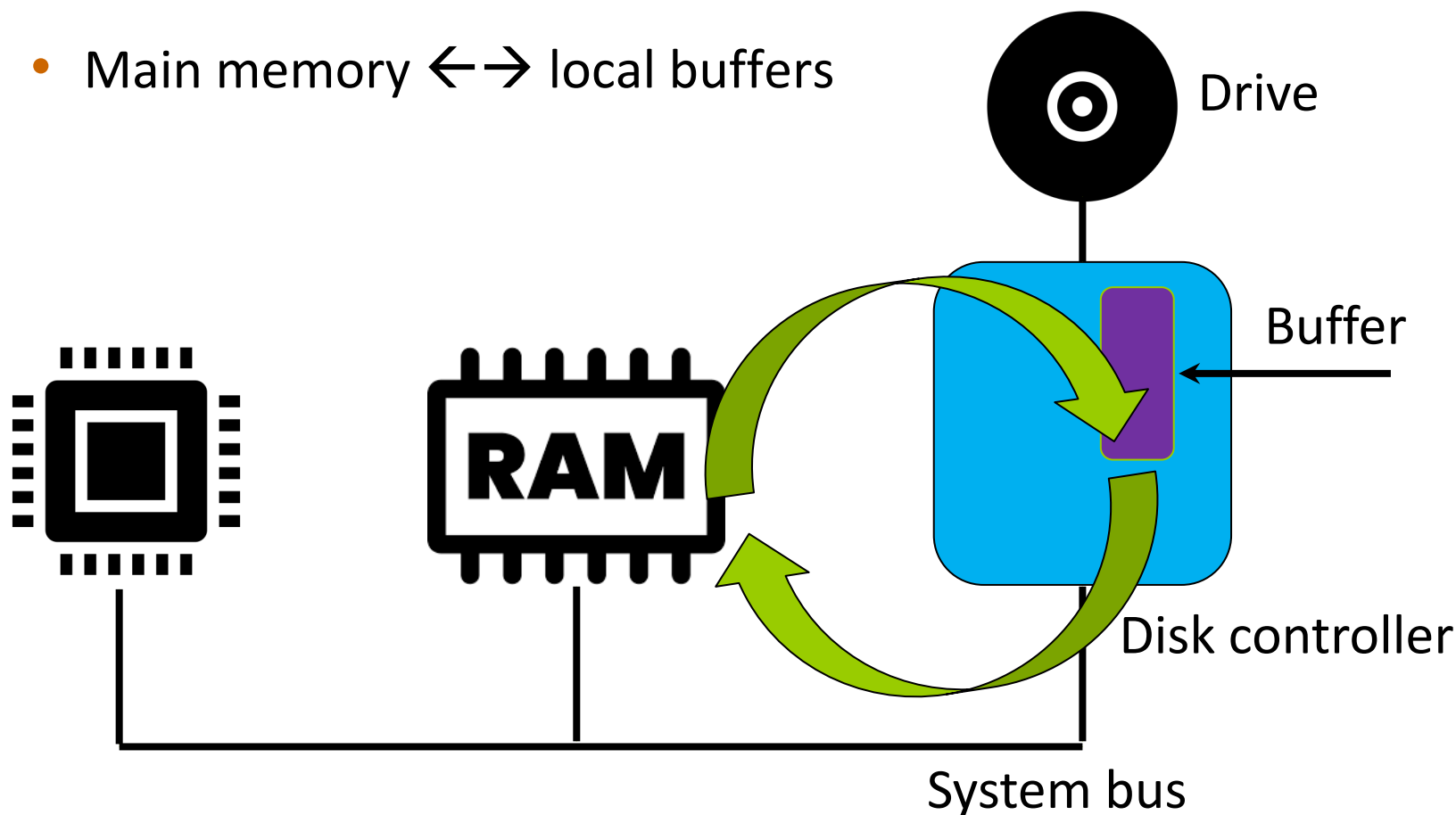
- Device controller informs CPU that it has finished its operation by causing an **interrupt**.



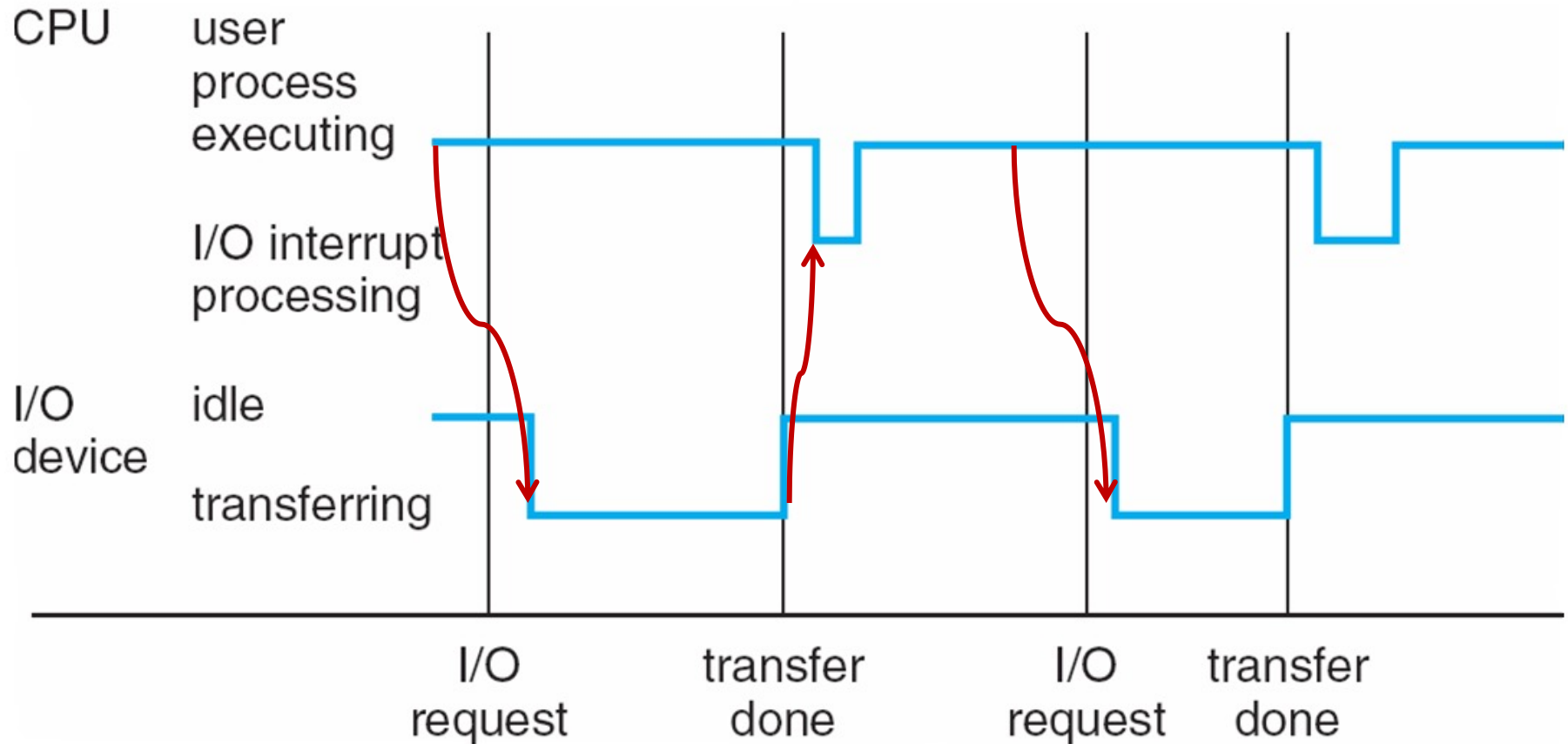
# Computer-System Operation (cont.)

## ■ CPU moves data

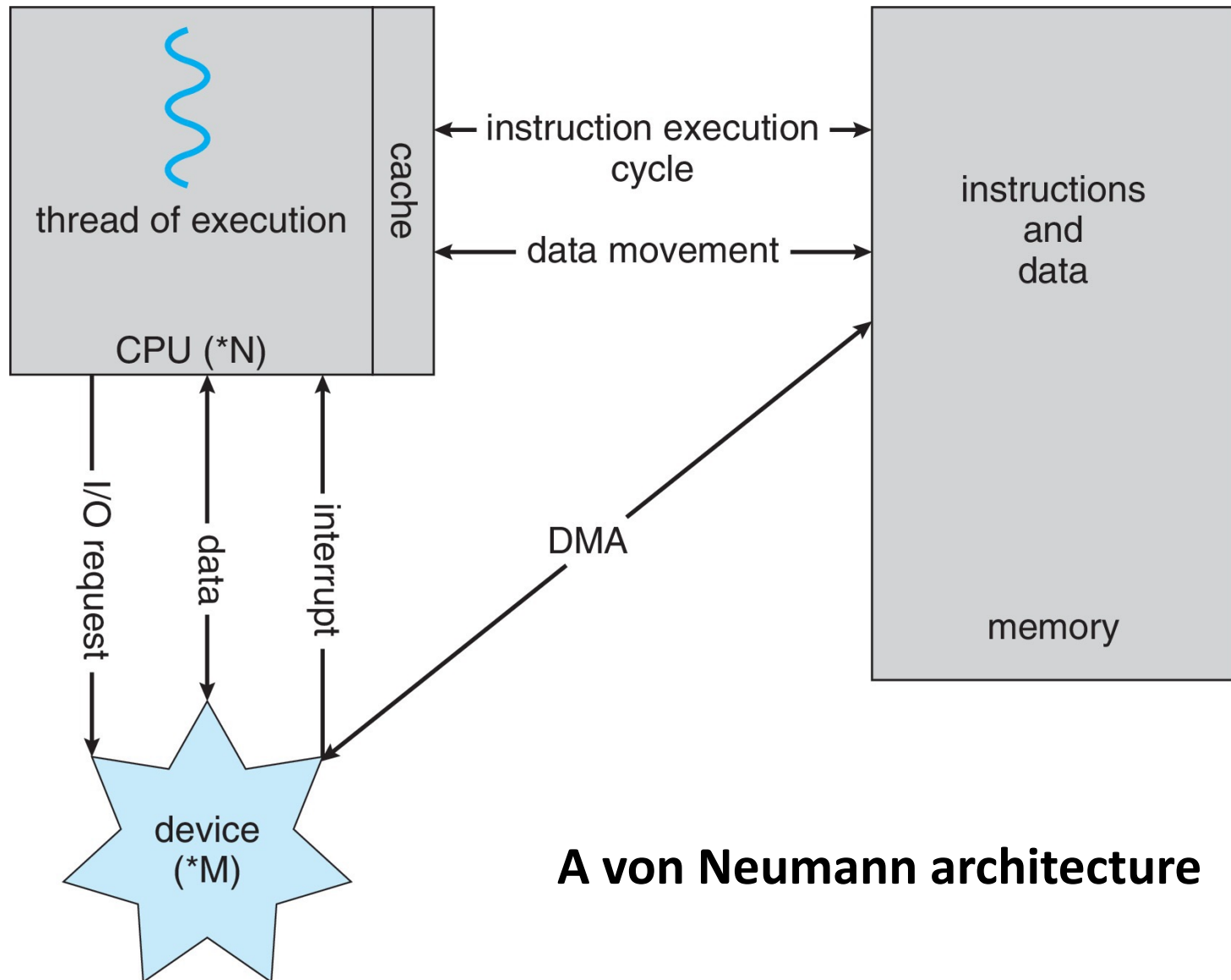
- Main memory  $\leftrightarrow$  local buffers



# 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.



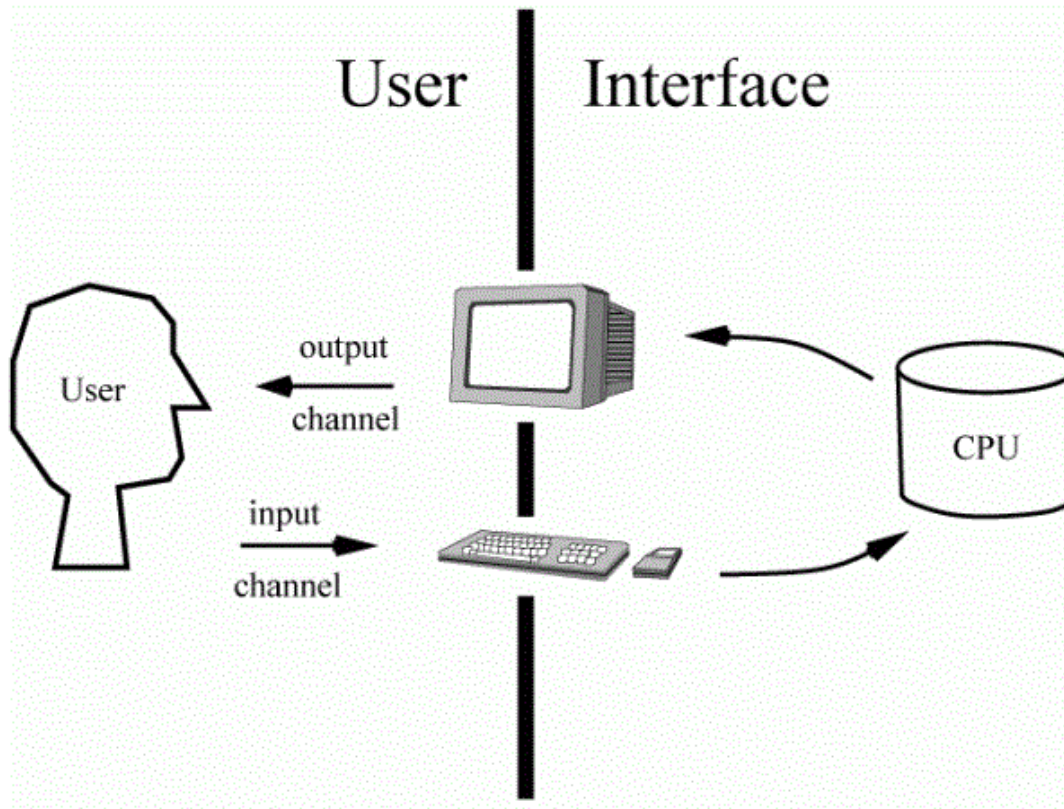
---

# MULTIPROGRAMMING VS MULTITASKING



# Multiprogramming (Batch System)

- Single user/program cannot always keep CPU and I/O devices busy.



# Multiprogramming (Batch System) (cont.)

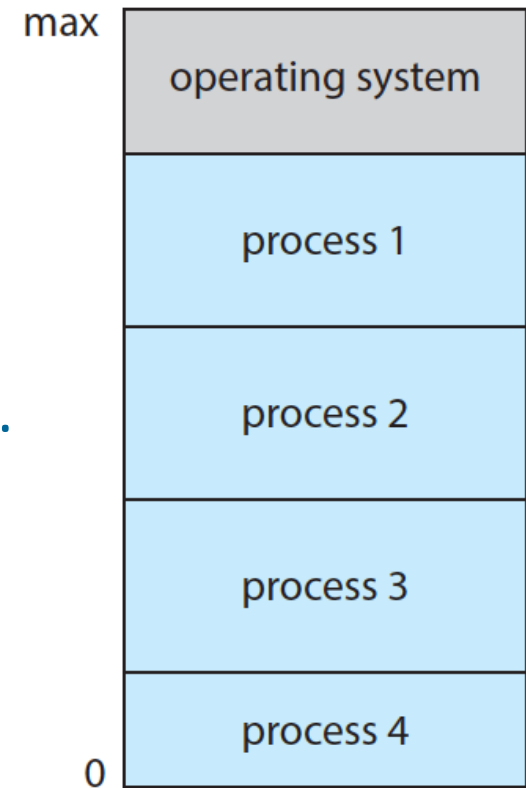
- Single user/program cannot always keep CPU and I/O devices busy.
- Examples

Program	CPU-intensive	Memory-intensive	I/O-intensive
Random Number Generator	?	?	?
Microsoft word	?	?	?
QuickTime Player (a long 4K video)	?	?	?



# Multiprogramming (Batch System) (cont.)

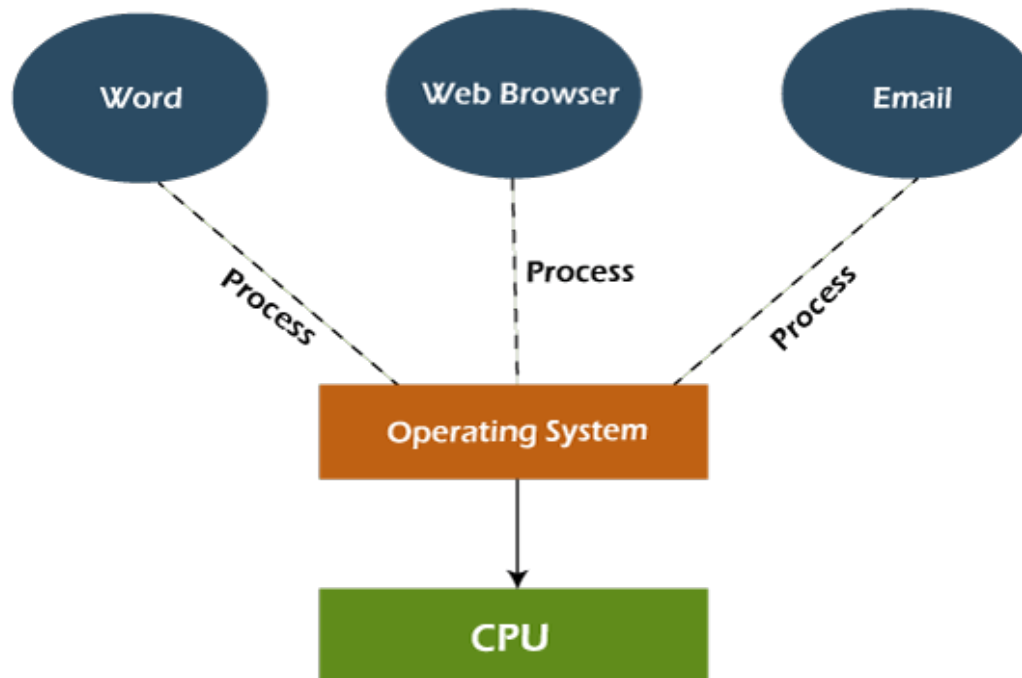
- Multiprogramming organizes multiple jobs (code and data) -->
  - CPU always has one to execute.
- A subset of total jobs in system is kept in memory.
- One job selected and run via **job scheduling**.
- When job has to wait (I/O for example), OS switches to another job.



Memory layout for a multiprogramming system

# Multitasking (Timesharing) (cont.)

- 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.



# Multitasking (Timesharing) (cont.)

---

- 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.

<https://www.geeksforgeeks.org/difference-between-job-task-and-process/>



---

# DUAL-MODE OPERATION

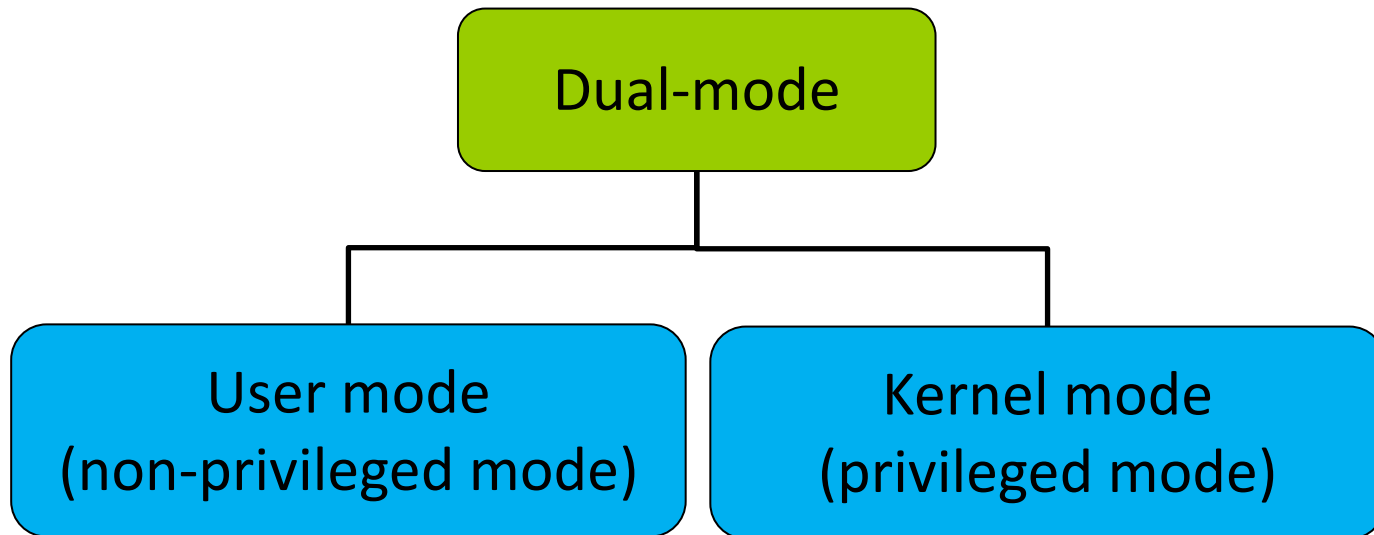




# Dual-mode Operation

---

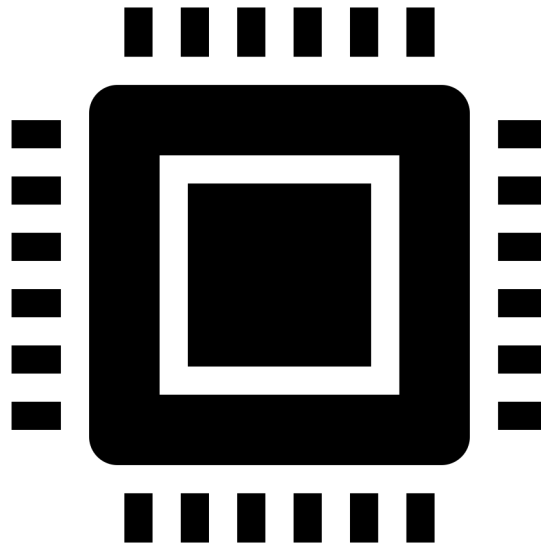
- **Dual-mode** operation allows OS to protect itself and other system components.
  - **User mode** and **kernel mode**



<https://allaboutse.com/what-are-the-dual-modes-advantages/>

# Dual-mode Operation (cont.)

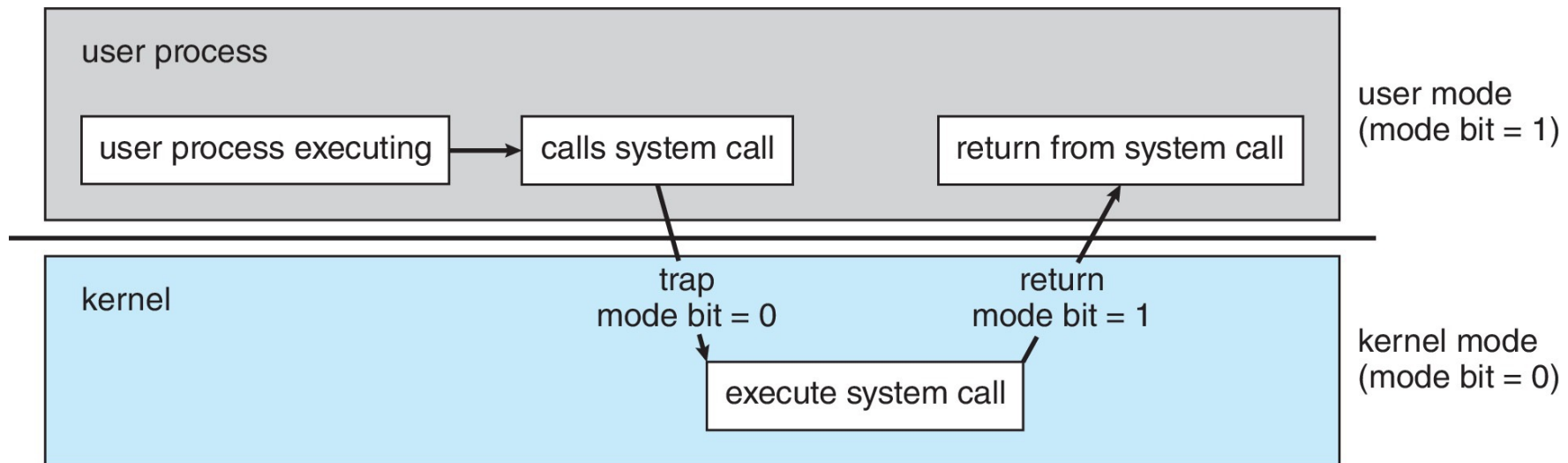
- **Mode bit** provided by hardware
  - To distinguish when system is running *user code* or *kernel code*.
  - When a user is running  $\Rightarrow$  mode bit is “user”.
  - When kernel code is executing  $\Rightarrow$  mode bit is “kernel”.



Mode bit

# 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.



# Types of Instructions

---

- Instructions are divided into two categories:
  - The ***non-privileged instruction*** instruction is an instruction that ***any application or user can execute***.
  - The ***privileged instruction*** is an instruction that ***can only be executed in kernel mode***.
- Instructions are divided in this manner because privileged instructions ***could harm the kernel***.

<http://web.cs.ucla.edu/classes/winter13/cs111/scribe/4a/>



# Examples of instructions

---

Instruction	Type
Reading the status of Processor	?
Set the Timer	?
Sending the final printout of Printer	?
Remove a process from the memory	?



# Examples of non-privileged instructions

---

- Reading the status of Processor
- Reading the System Time
- Sending the final printout of Printer

<https://www.geeksforgeeks.org/privileged-and-non-privileged-instructions-in-operating-system/>



# Examples of privileged instructions

---

- I/O instructions and halt instructions
- Turn off all Interrupts
- Set the timer
- Context switching
- Clear the memory or remove a process from the memory
- Modify entries in the device-status table

<https://www.geeksforgeeks.org/privileged-and-non-privileged-instructions-in-operating-system/>



# Privileged instructions

---

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*.