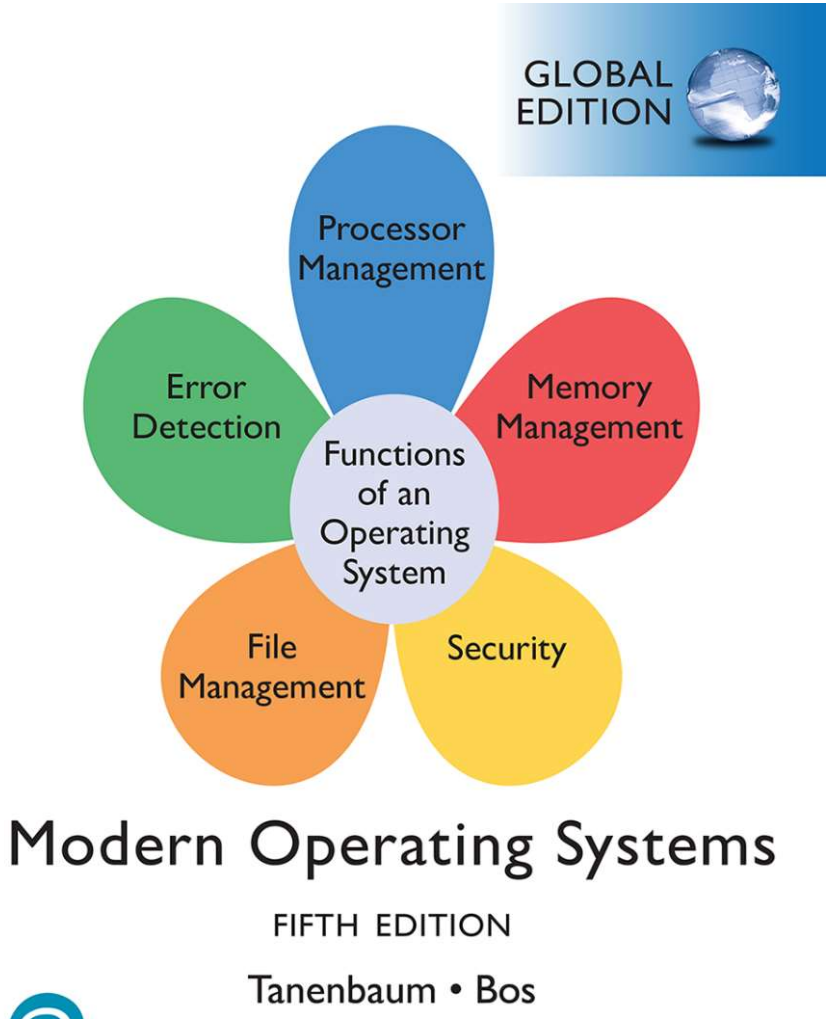
Modern Operating Systems

Fifth Edition, Global Edition



Lecture 1

Introduction - Definitions

What is an Operating System

The Operating System (OS) is a program that serves as an interface between a computer system's hardware and its users (applications). It performs:

- Controlled access to the system's resources (security)
- Sharing system resources (performance and fairness)
- Services for easy access to hardware
  - File systems
  - Network protocols
- A common hardware interface for programmers (POSIX)
  - hiding the messy details (abstraction, standardization)

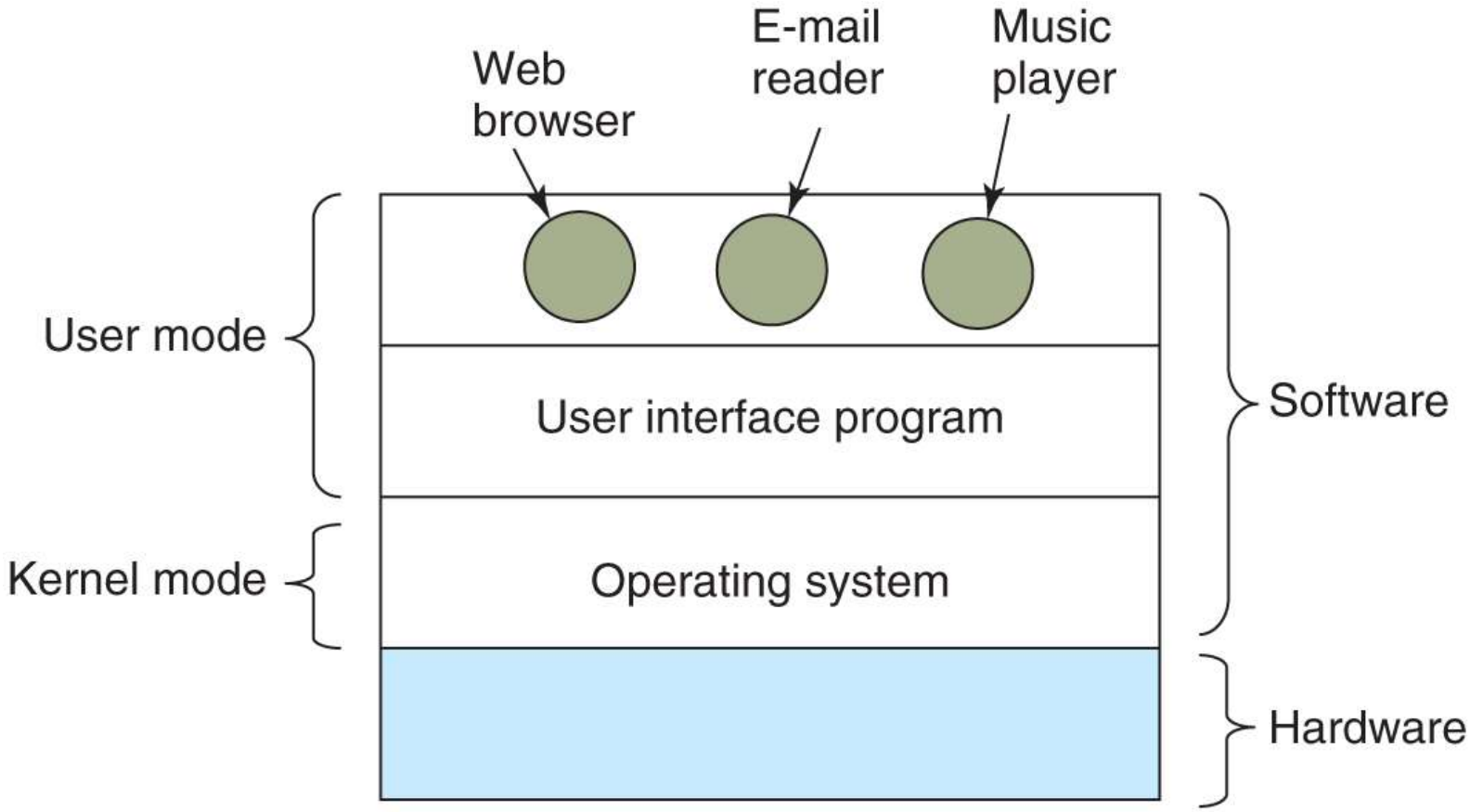
What is an Operating System

Top-down and bottom-up definitions

- Provides programs with easy and efficient access to system resources
- Provides a structured and controlled allocation of processors, memory, and other I/O devices among the various client programs that compete with each other to use them." (Tanenbaum, 2001)

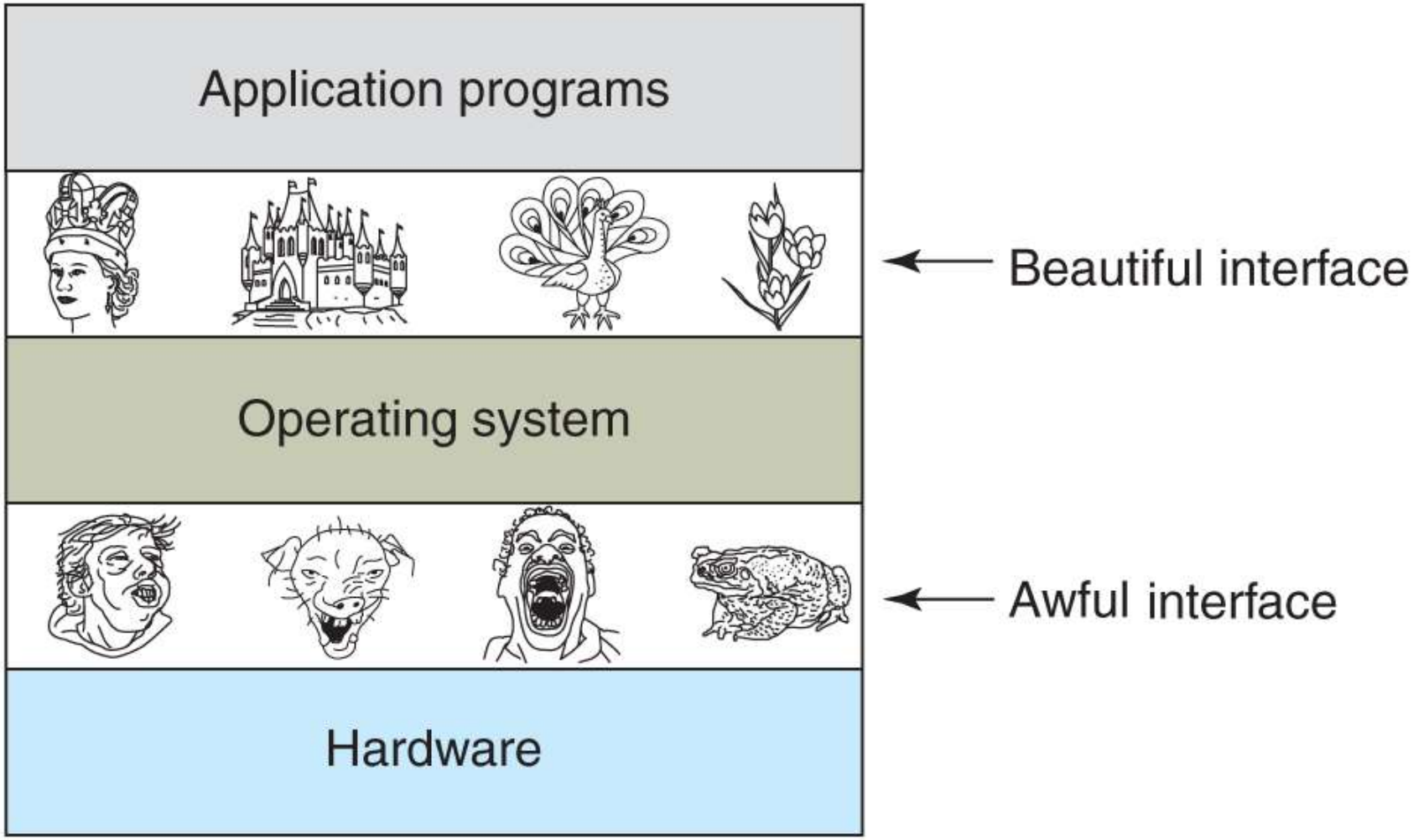
OS Layers: User vs Supervisor

Computers are equipped with an OS layer, whose job is to provide user programs with a better, simpler, cleaner, model of the computer and handle resource managing.



Operating Systems Layers: Why

Operating systems turn awful hardware into beautiful abstractions.

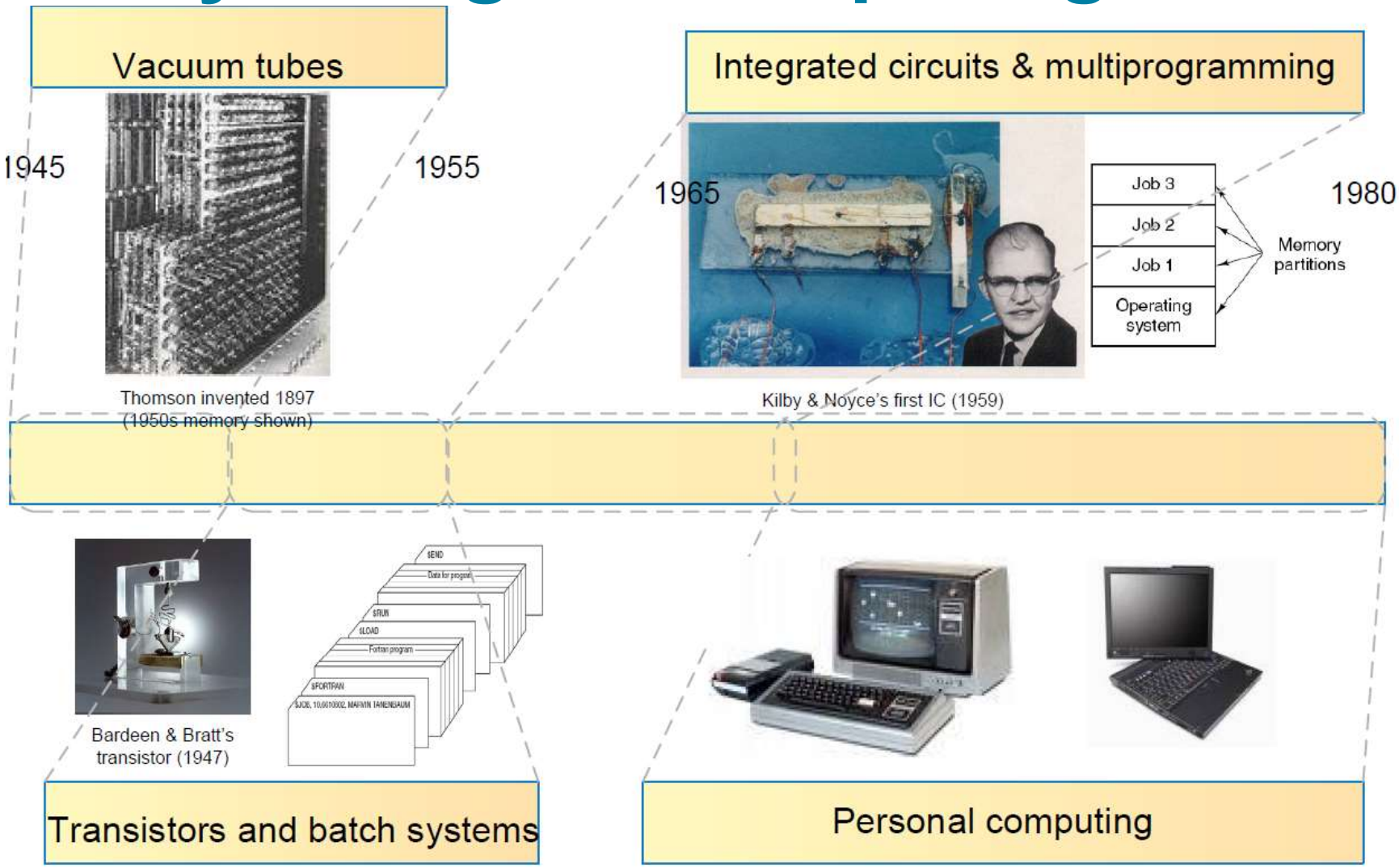


Operating System Layers

Structure of the operating system.

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

History of Digital Computing

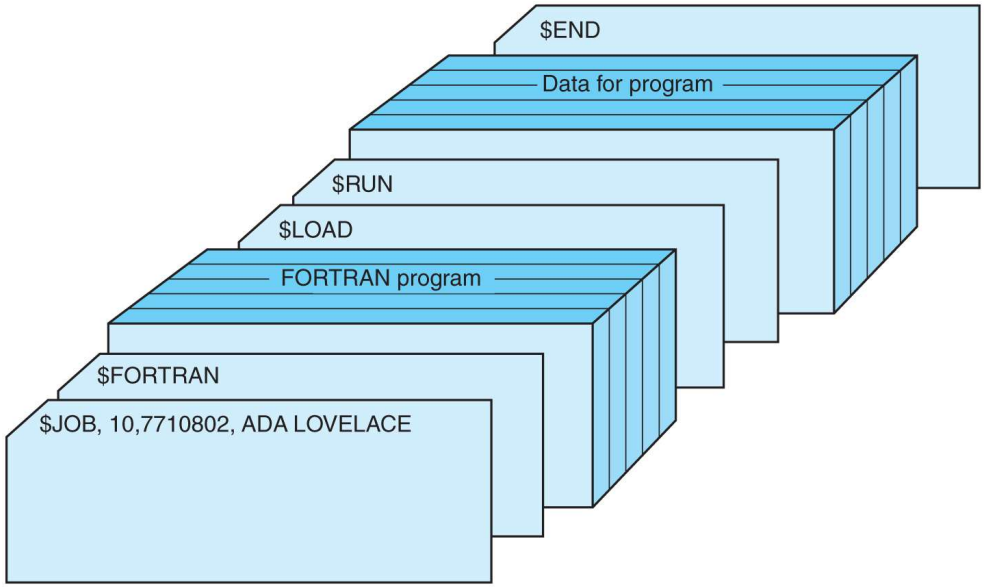


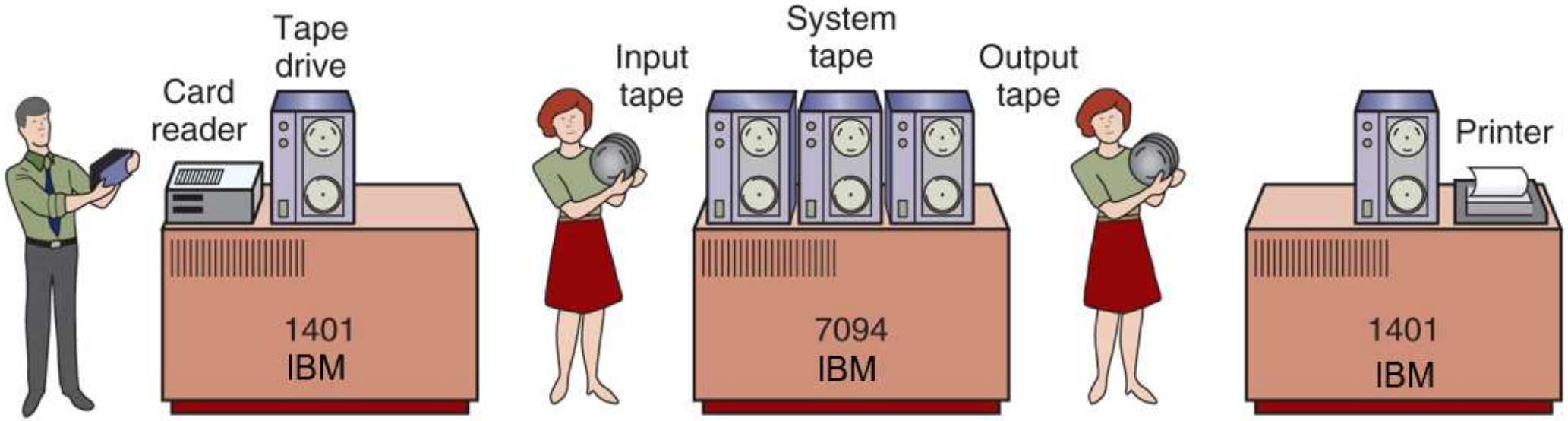
- 1955 1965 1980 2008s

Then, Mobile Computers, Edge Computing, Cloud, …

Fortran Monitoring System (FMS)

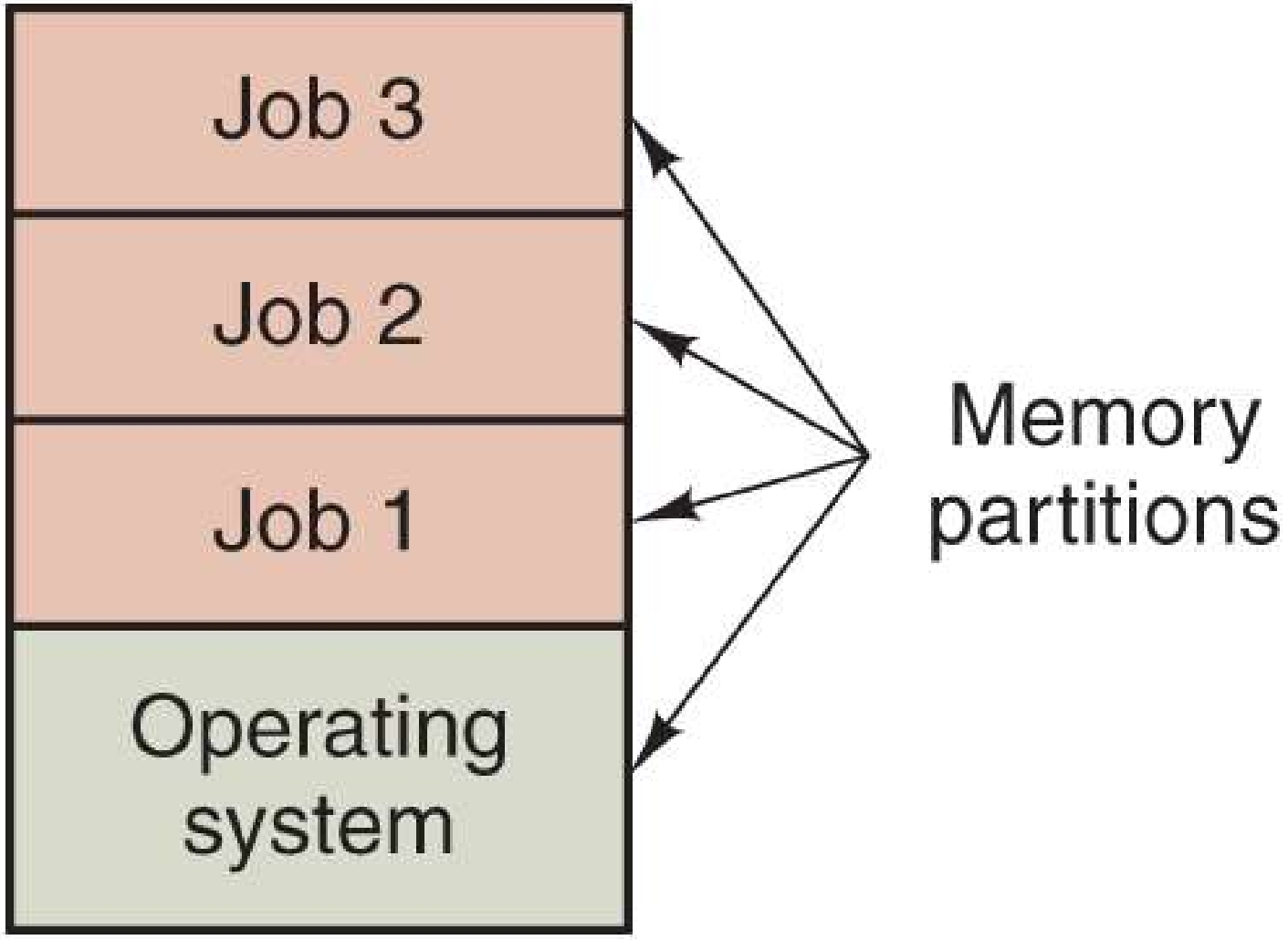
Structure of a typical FMS job.





Multiprogramming on Uniprocessors

A multiprogramming system executing three jobs “in parallel”

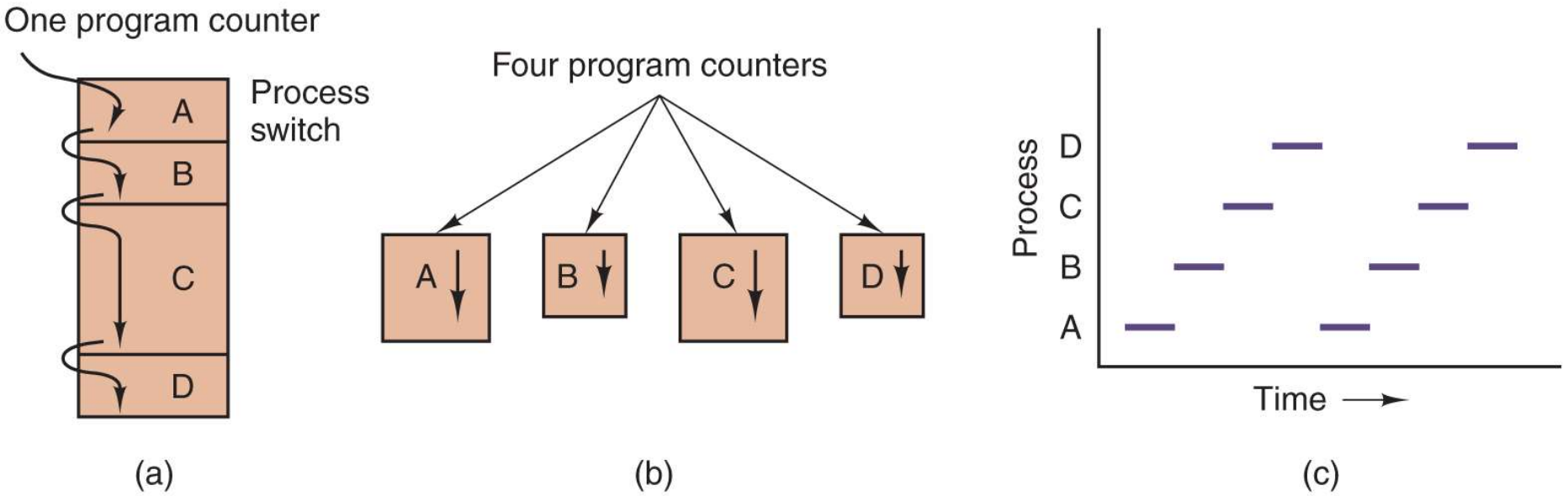


Multiprogramming

How do we transition from one process to another?

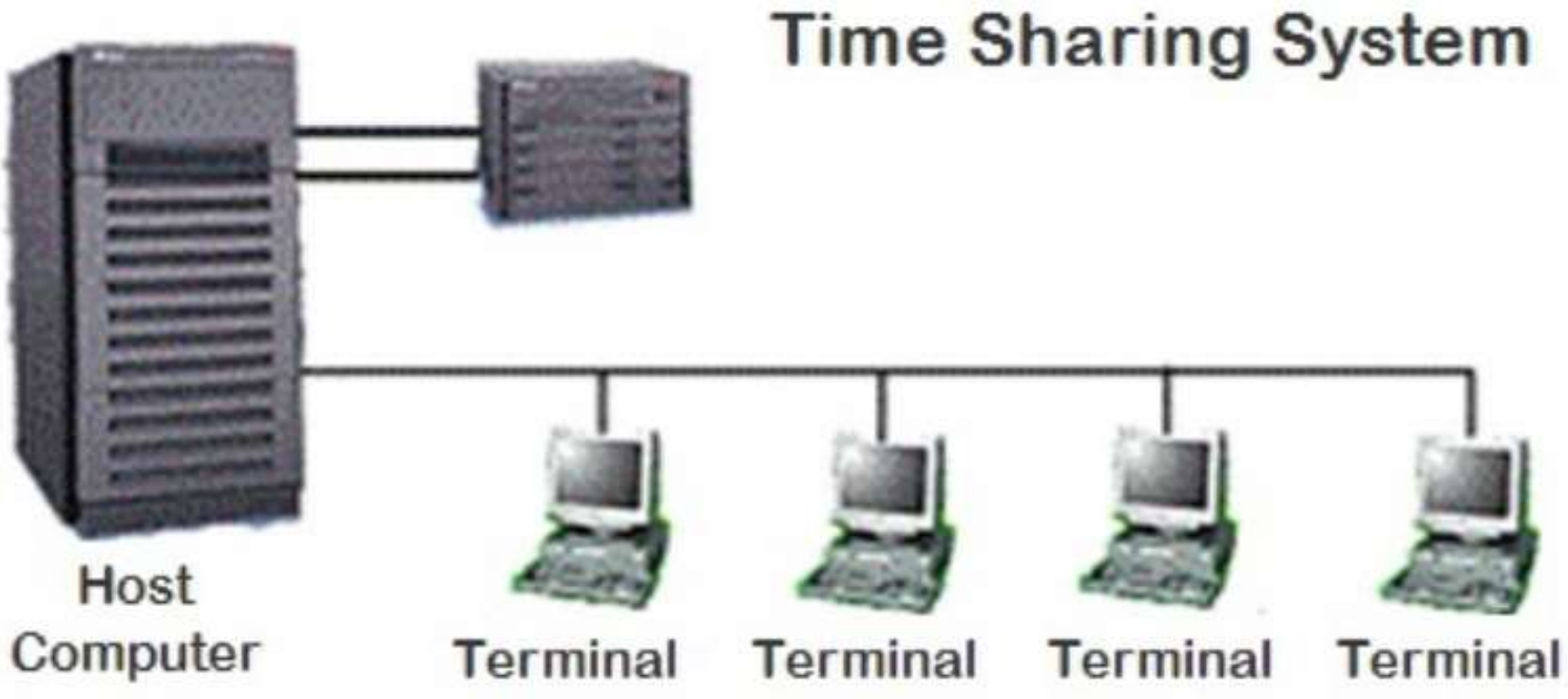
- preemptive multitasking: forced release of processor resource based on external conditions (e.g., wait)
- cooperative multitasking: processes explicitly “yield”

(a) Multiprogramming four programs
(b) Four independent, sequential processes
(c) Only one program is active at once.

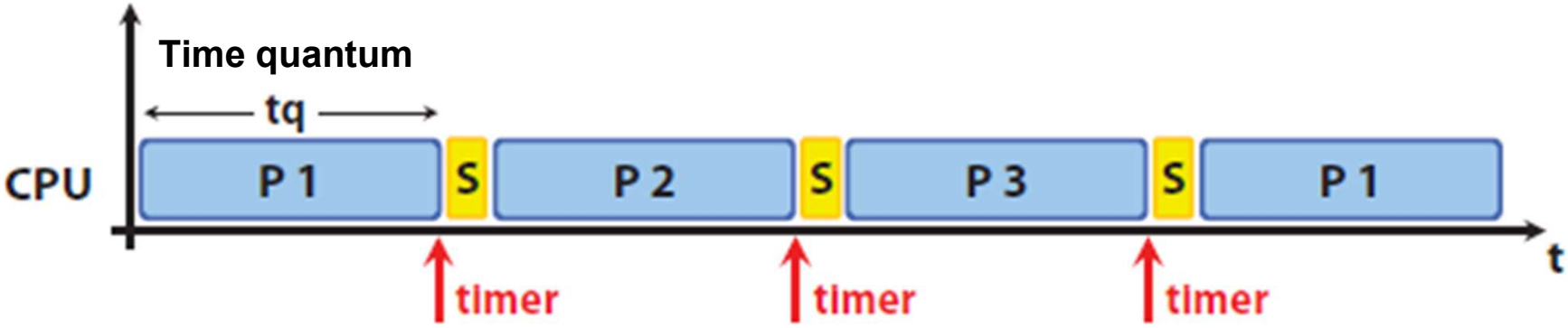


Multitasking (Time Sharing/Co-op Sched)

Time sharing allows users to connect in round robin fashion to share the mainframe resources (CPU, etc)



Context Switch



- The CPU switches from one process to another
- Save the state of the old process, e.g., PCB P1 & MMU data
- Flush & clear cache* and processor pipeline
- Restore the new process, e.g., PCB P2 & MMU data
- Return from interrupt to new process
- How long does it take? (Lost time for the CPU)
- How often does it occur?

*OS/CPU dependent

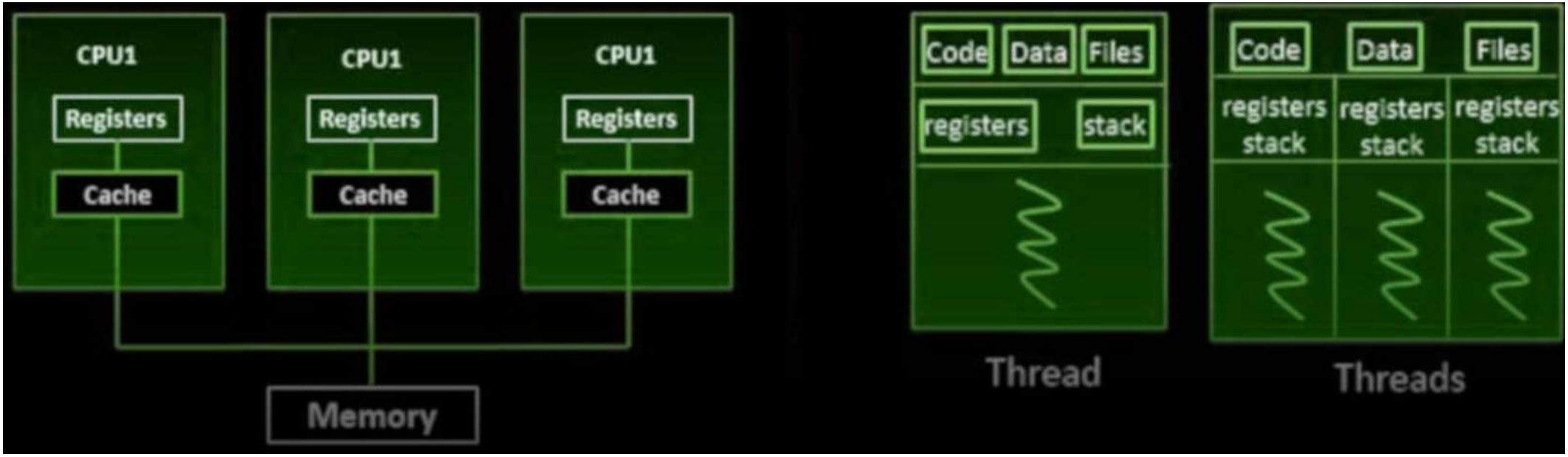
Scheduler: Dispatching

Responsible for context switches between processes

- Interrupt request occurs and dispatch service routine invoked
- Save memory management unit (MMU) data
- Save process state (e.g.registers)
- Flush and clear cache* and processor pipeline
- Load new MMU data
- Load new process state
- Return from interrupt into new process

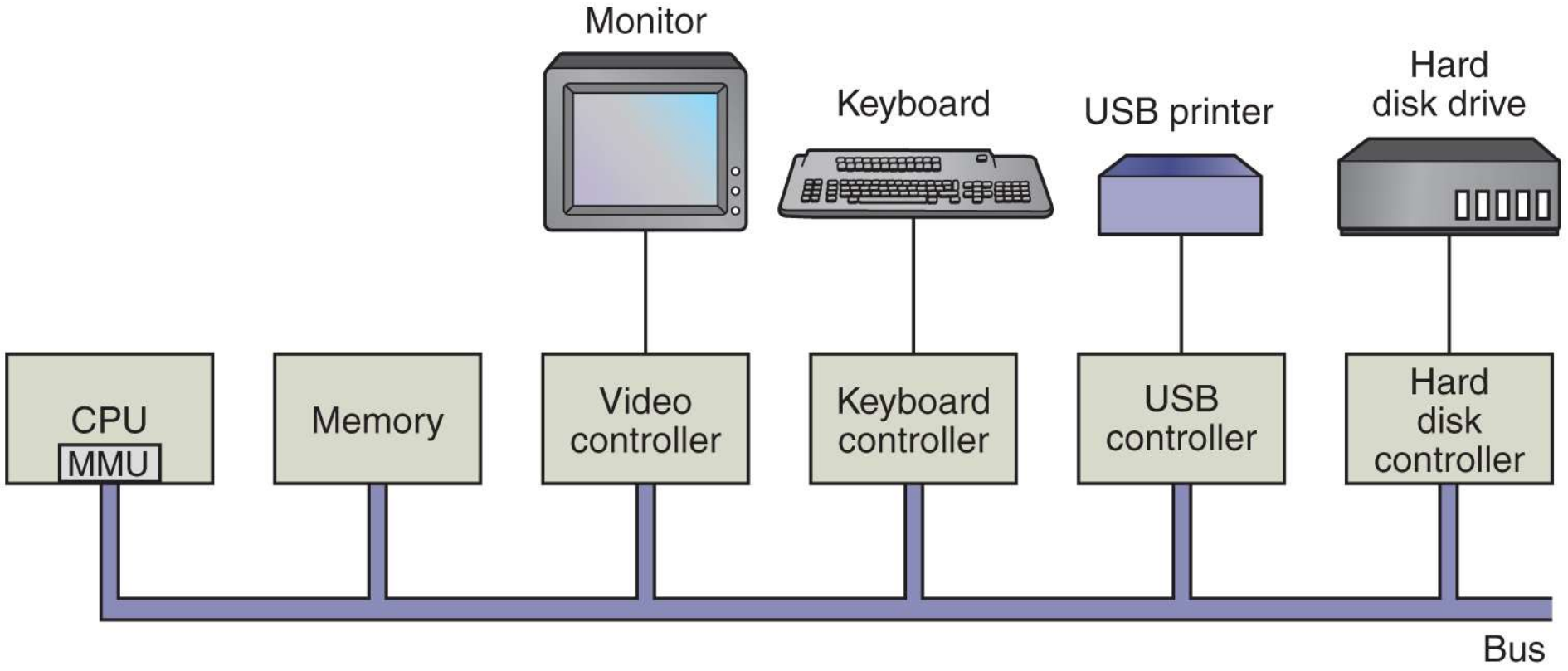
Multiprocessing vs Multithreading

Processes or threads run on multicores (SMP)



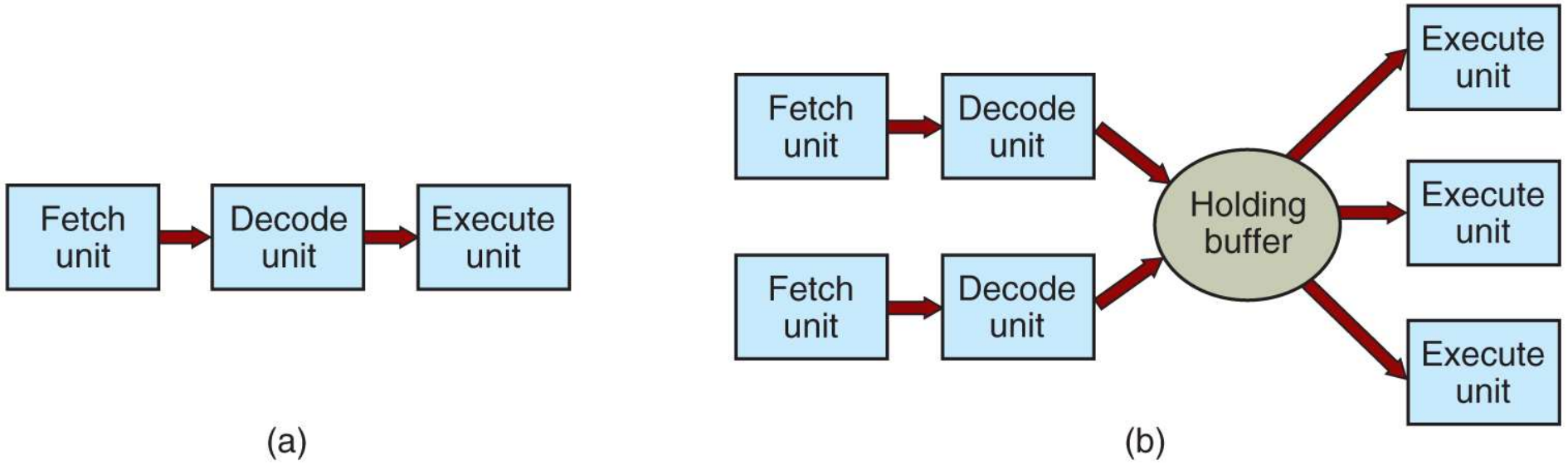
Modern PC - Hardware Components

Some of the components of a simple personal computer.



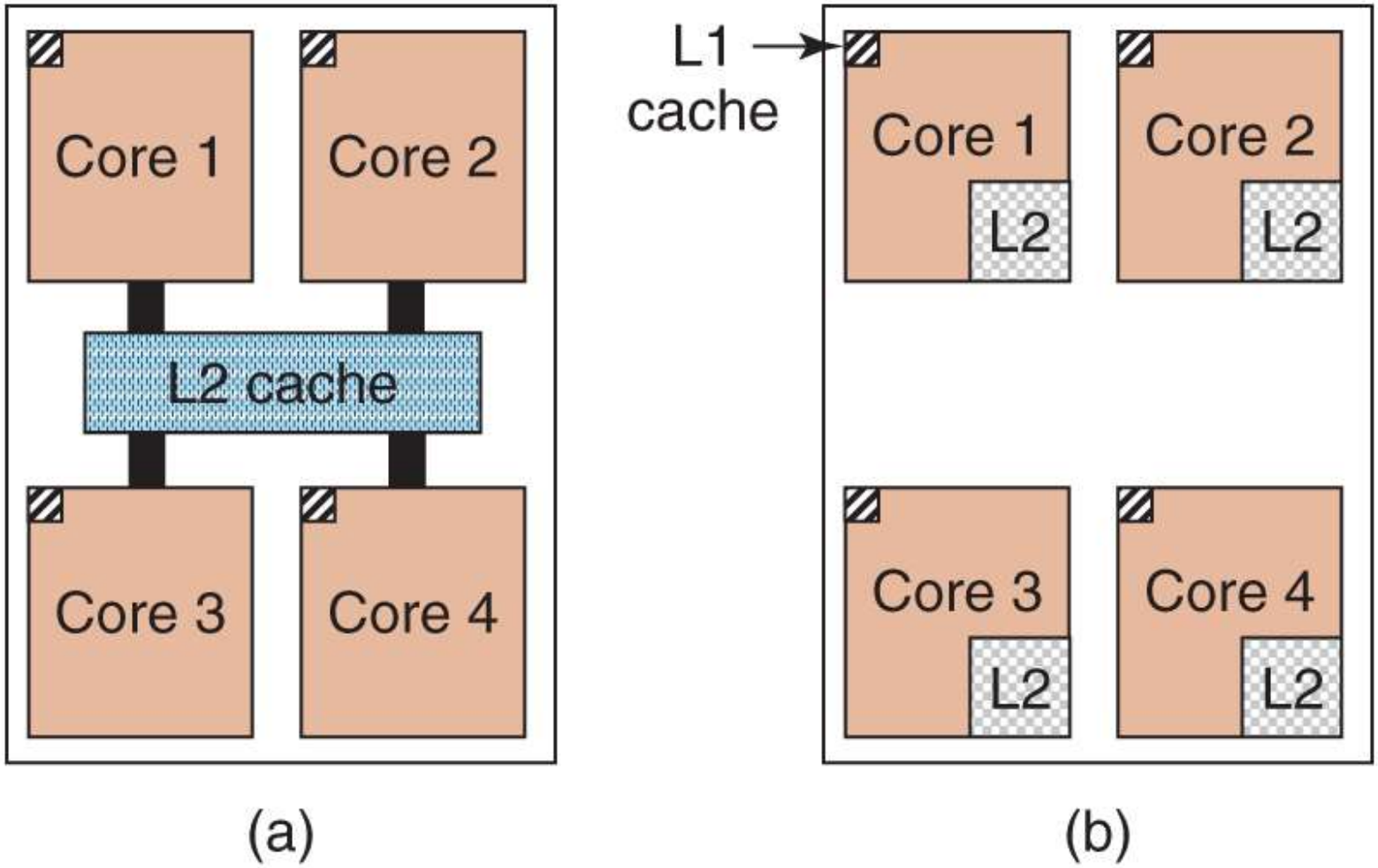
System Architecture - Exec Pipelines

a) A three-stage pipeline. (b) A superscalar CPU.

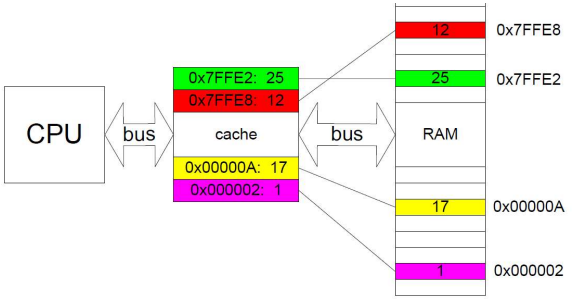


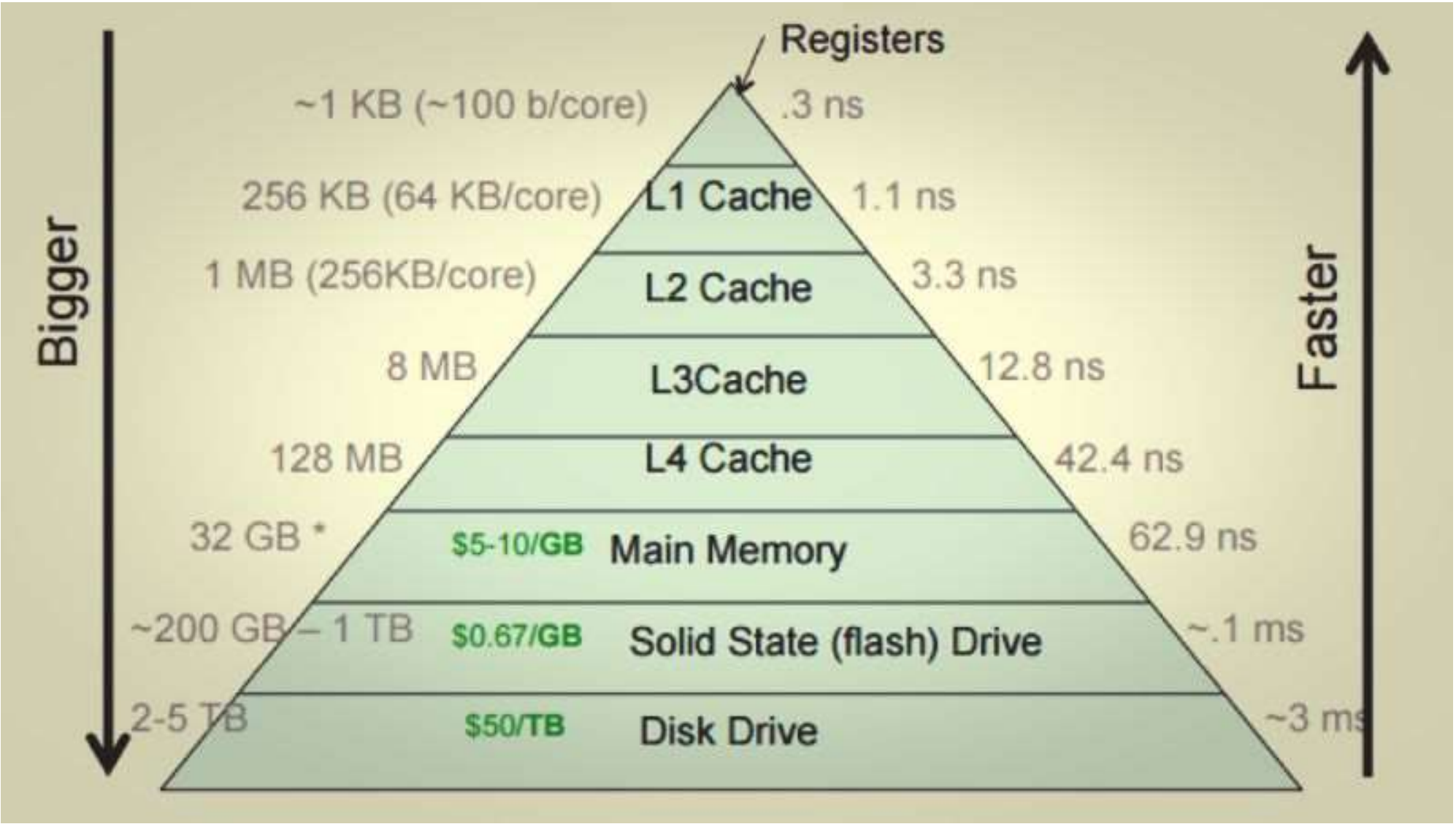
Multicore SoC (NoC)

(a) A quad-core chip with a shared L2 cache. (b) A quad-core chip with separate L2 caches.

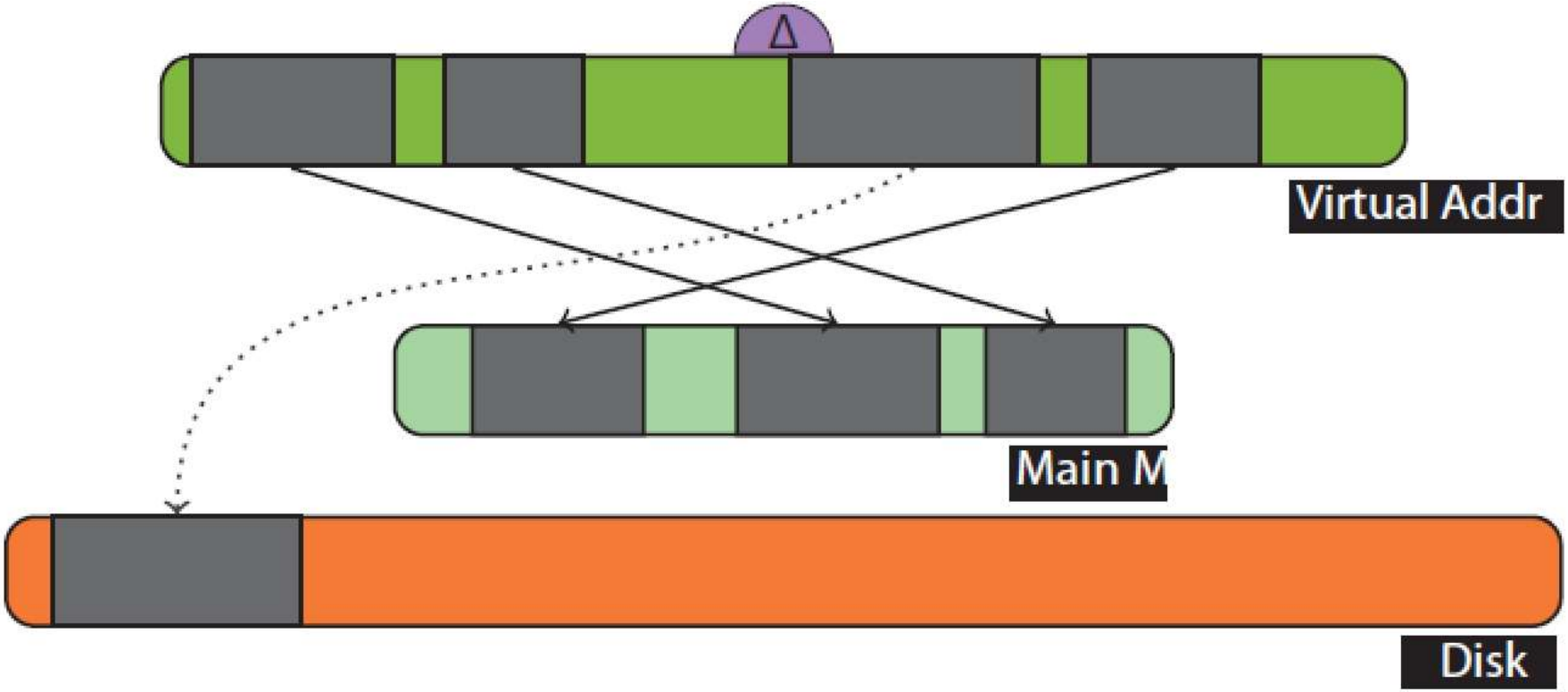


Memory Hierarchy





Memory Management



- Virtual to Physical Address Translation

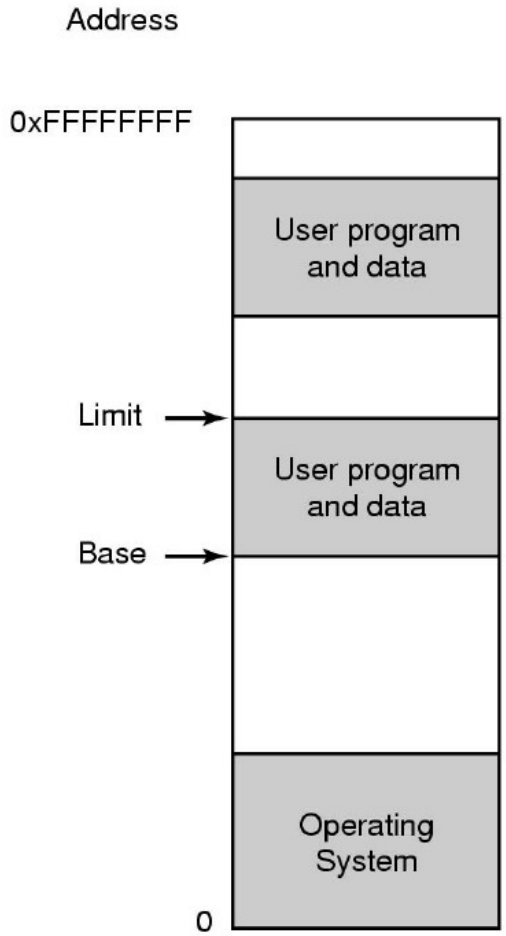
Memory Protection

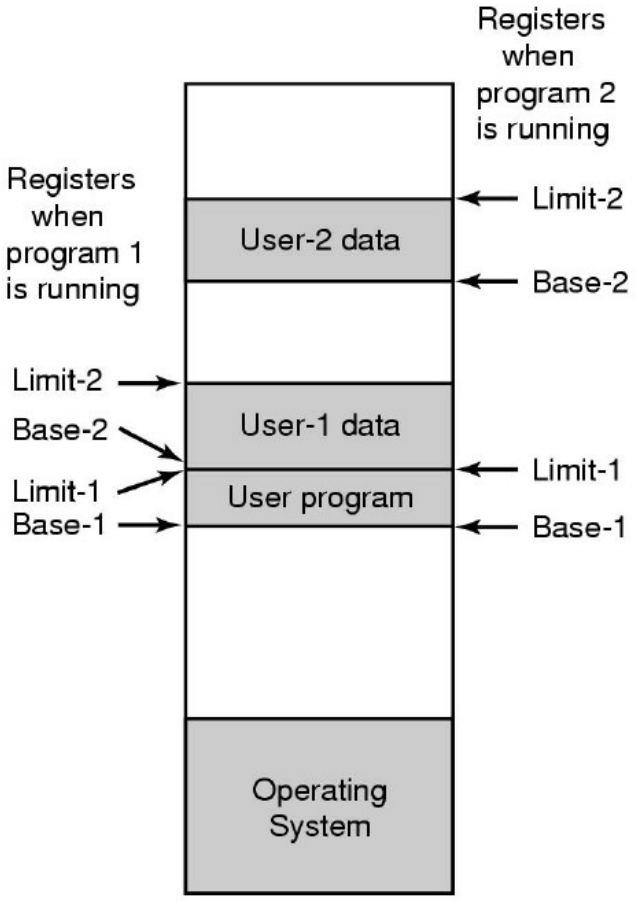
- Isolation (User- vs Kernel-space)
  - Trusted Execution Environments (Rich vs Secure world)
- Some instructions are privileged
- Interrupt service routines (ISR) runs in supervisor mode

Other concepts

- Authentication/Access Control (Hardware Protection, Shadow Stack)
- Protection/Sharing/Data Exec (Segmentation, Paging, VM/Containers)
- Base-Limit registers (Stack), Control-Flow Integrity
- Address Space Layout Randomization
- Encryption and Cryptography
- TPM, Secure Boot, Secure Firmware/Software over the Air Updates

Memory Protection (Base-Limit Pairs)





I/O Architecture

All Input/Output (I/O) is mediated by a controller (driver).

Two types of I/O

- Programmed: CPU polls device
- Interrupt driven: Device signals upon completion

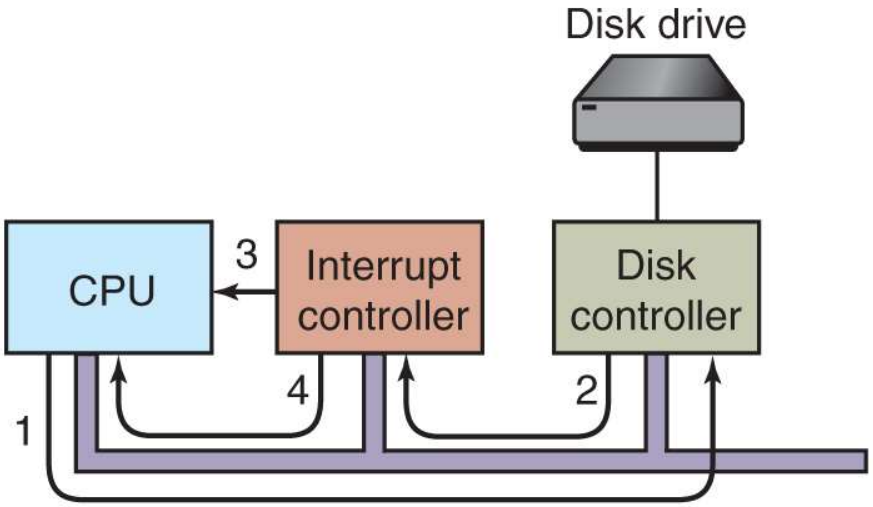
I/O can be

- Synchronous
- Asynchronous

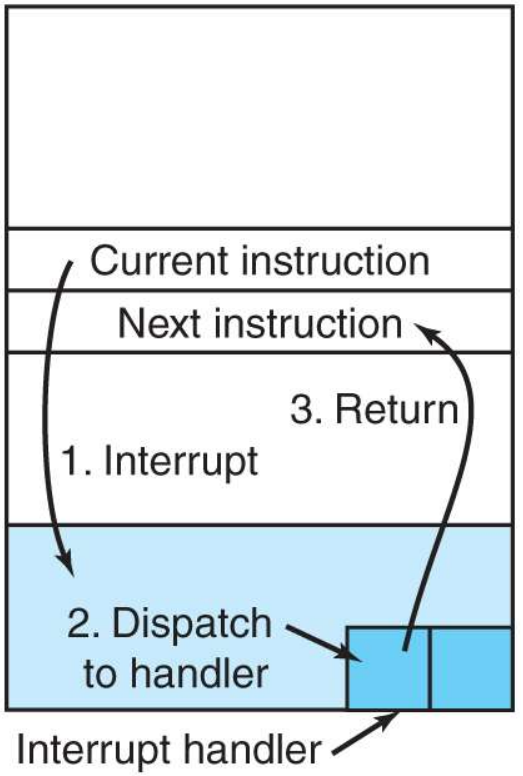
System Architecture: I/O Interrupts

(a) Operating an I/O device, which requested an interrupt

(b) (b) Interrupt processing involves requesting the interrupt, running the handler, and returning to the user program



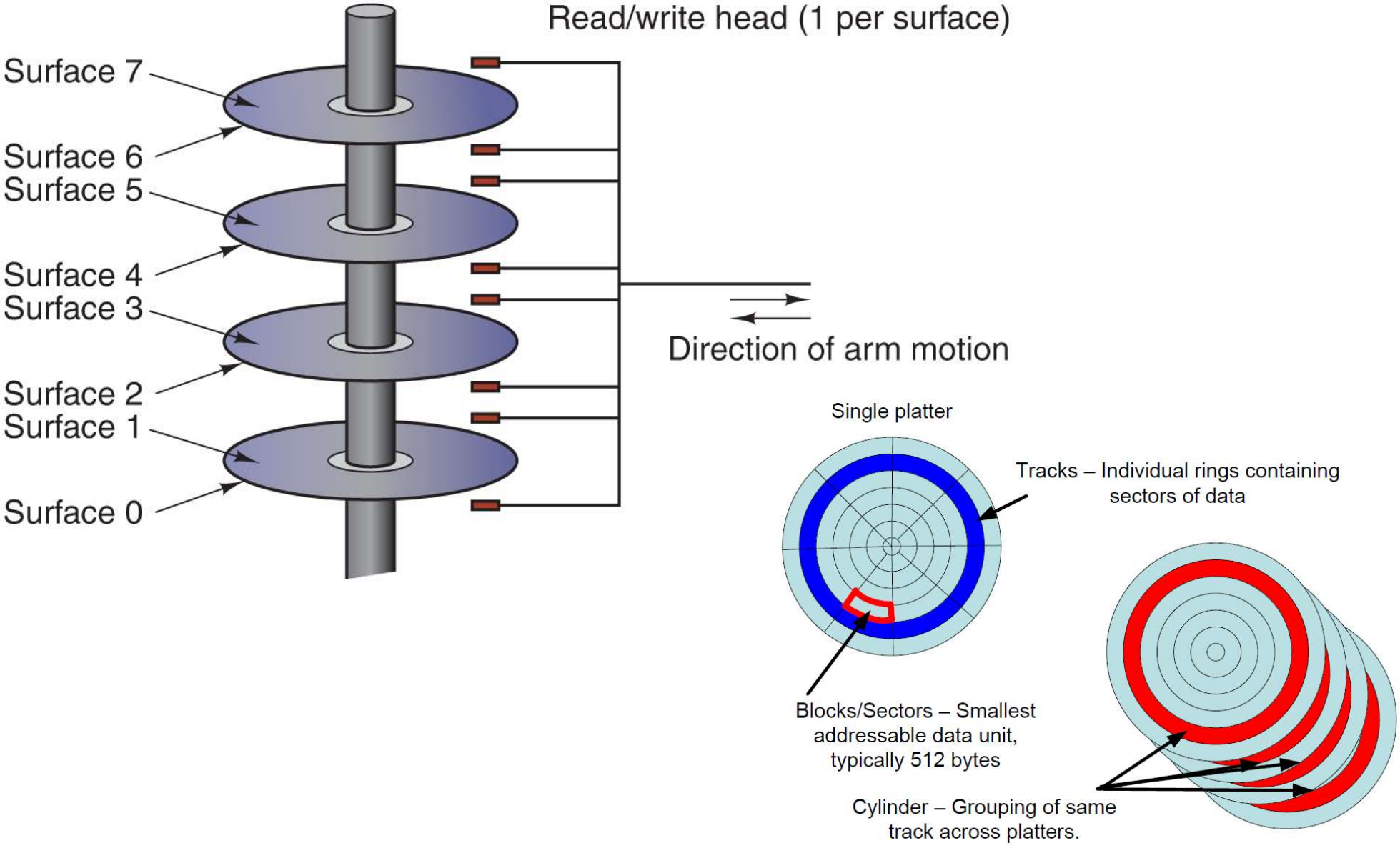
(a)



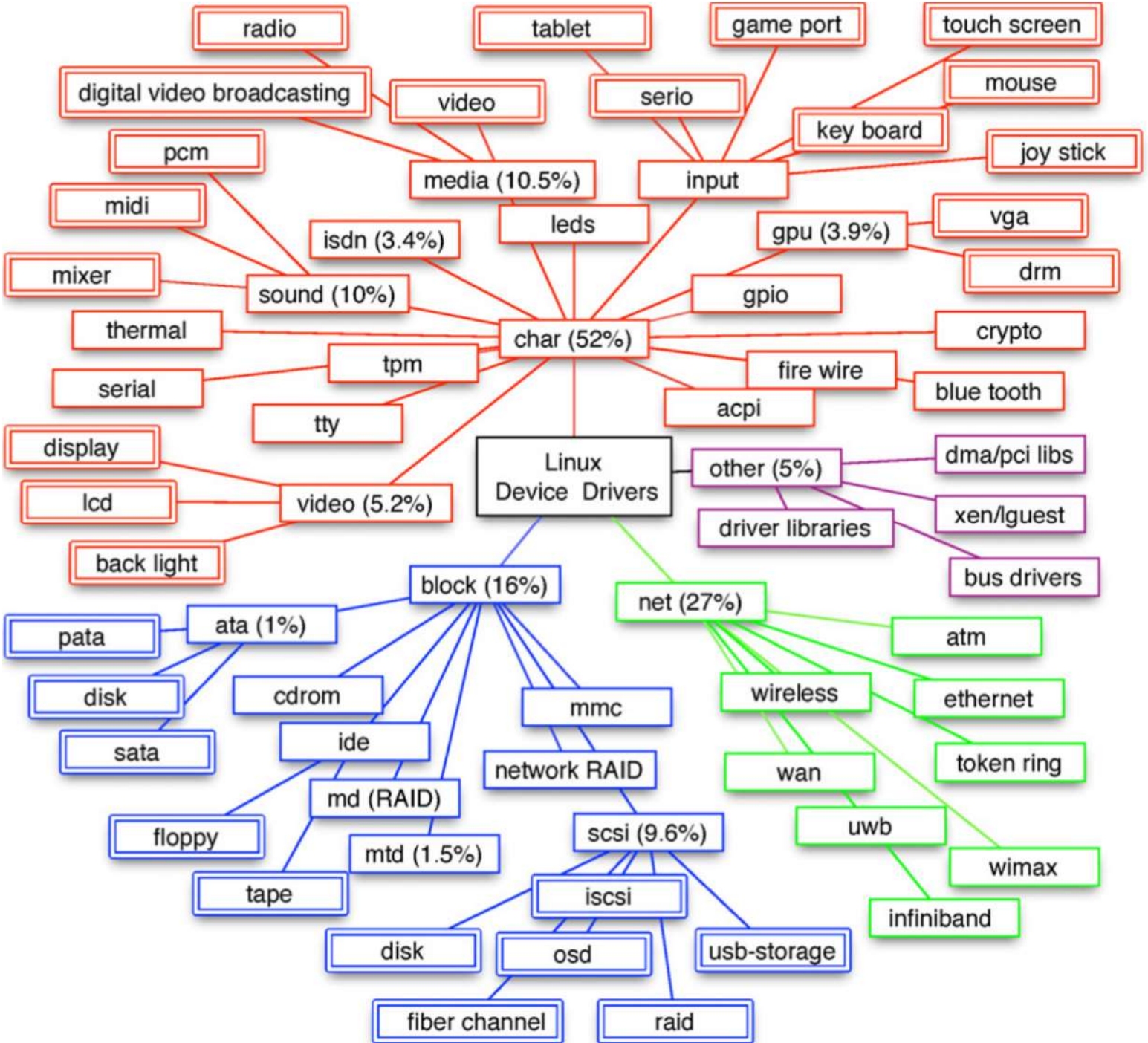
(b)

I/O Architecture

Structure of a disk drive.

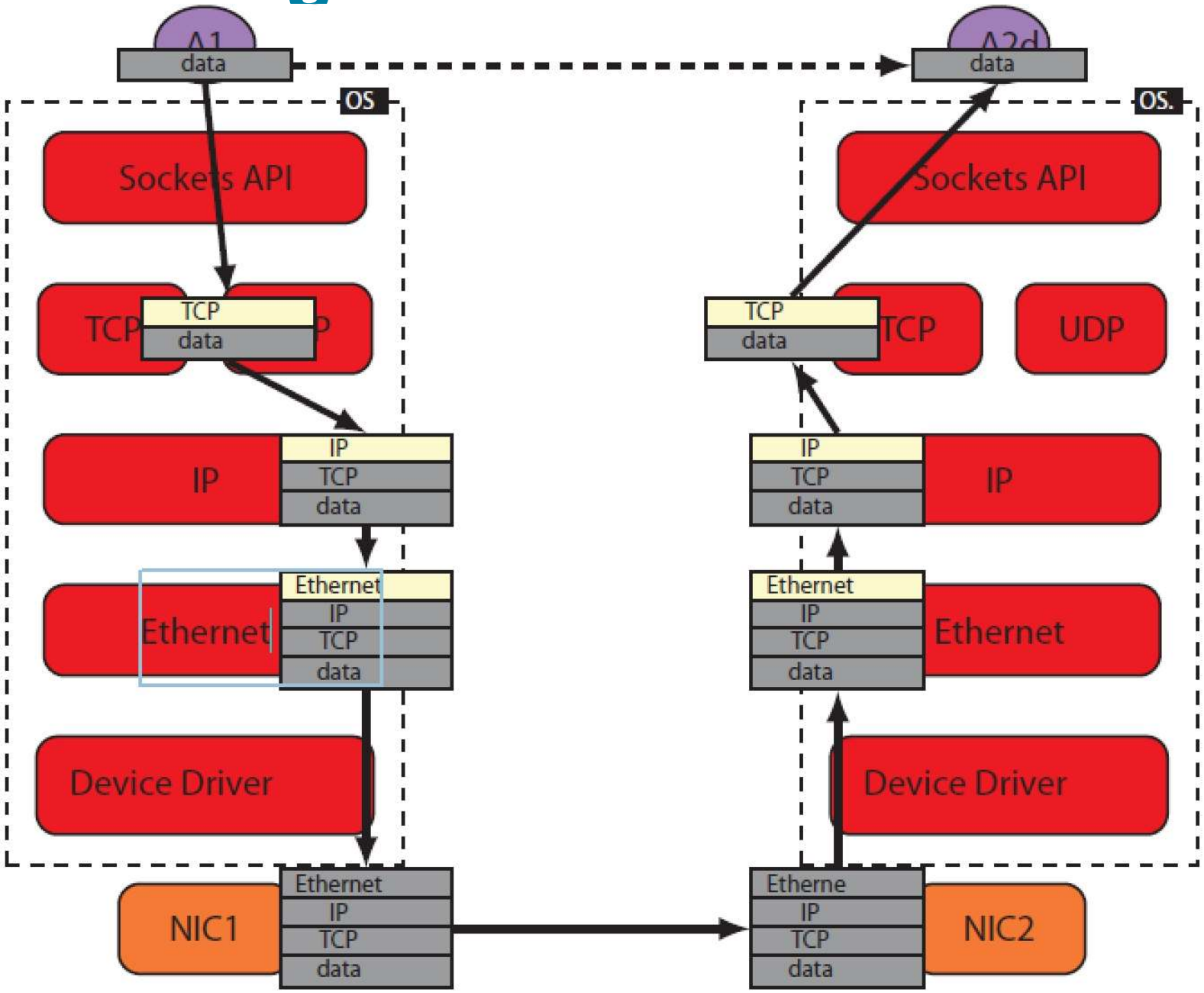


I/O Architecture: Device Drivers (Types)

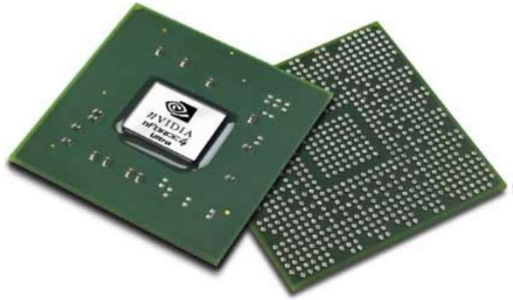


I/O Architecture: Device Drivers

Networking

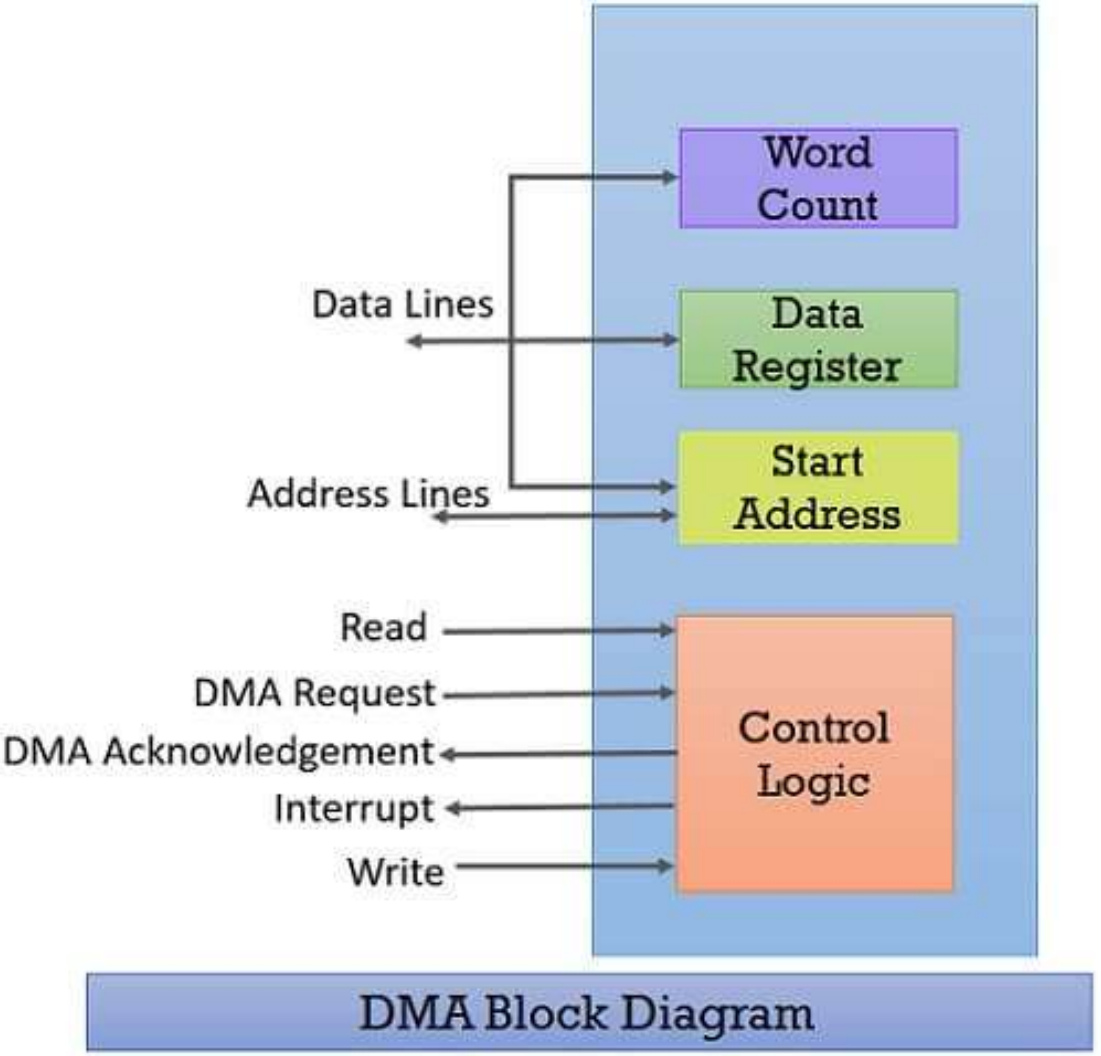

A1
data
OS
Sockets API
TCP
TCP
data
IP
IP
TCP
data
Ethernet
Ethernet
IP
TCP
data
Device Driver
NIC1
Ethernet
IP
TCP
data
A2d
data
OS.
Sockets API
TCP
TCP
data
UDP
IP
IP
TCP
data
Ethernet
Ethernet
IP
TCP
data
Device Driver
Etherne
IP
TCP
data
NIC2

Direct Memory Access (DMA)



DMA releases the CPU from managing a data transfer between peripheral and RAM

- CPU indicates source/dest address and instructs the DMA controller what to do
- CPU continues executing
- DMA manages transfer between device and RAM
- Controller informs CPU when done via interrupt



x86 Architecture

The structure of a large x86 system.