Operating Systems

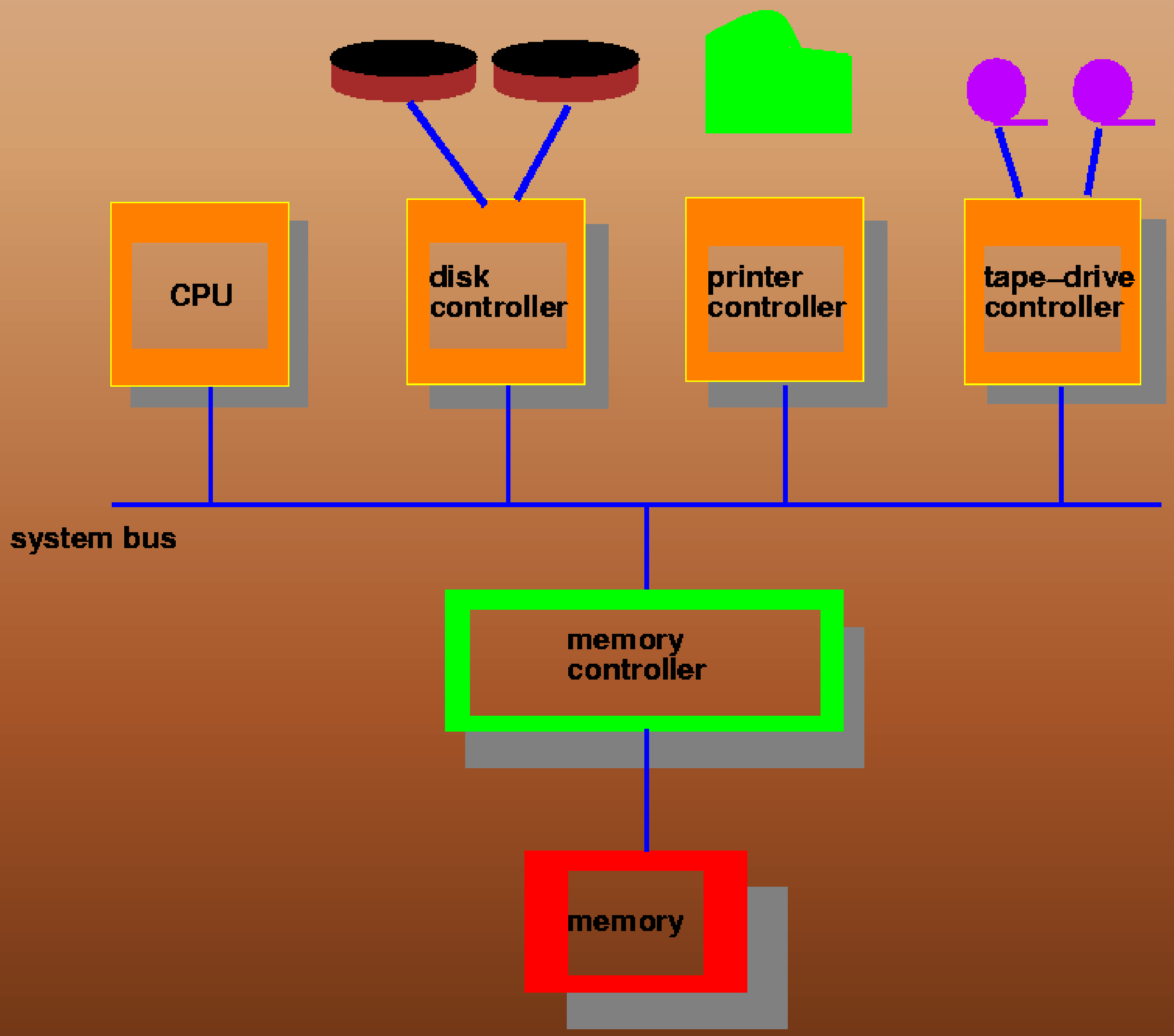
Introduction and Overview

EMMANUEL FON TATA
0540911516
tataemma99@yahoo.com

Introduction

- What is an operating system?
- Early Operating Systems
  - Simple Batch Systems
  - Multiprogrammed Batch Systems
- Time-sharing Systems
- Personal Computer Systems
- Parallel and Distributed Systems
- Real-time Systems

Computer System Architecture



What is an Operating System?

- An OS is a program that acts an intermediary between the user of a computer and computer hardware.
- Major cost of general purpose computing is software.
  - OS simplifies and manages the complexity of running application programs efficiently.

Goals of an Operating System

- Simplify the execution of user programs and make solving user problems easier.
- Use computer hardware efficiently.
  - Allow sharing of hardware and software resources.
- Make application software portable and versatile.
- Provide isolation, security and protection among user programs.
- Improve overall system reliability
  - error confinement, fault tolerance, reconfiguration.

Types of modern operating systems

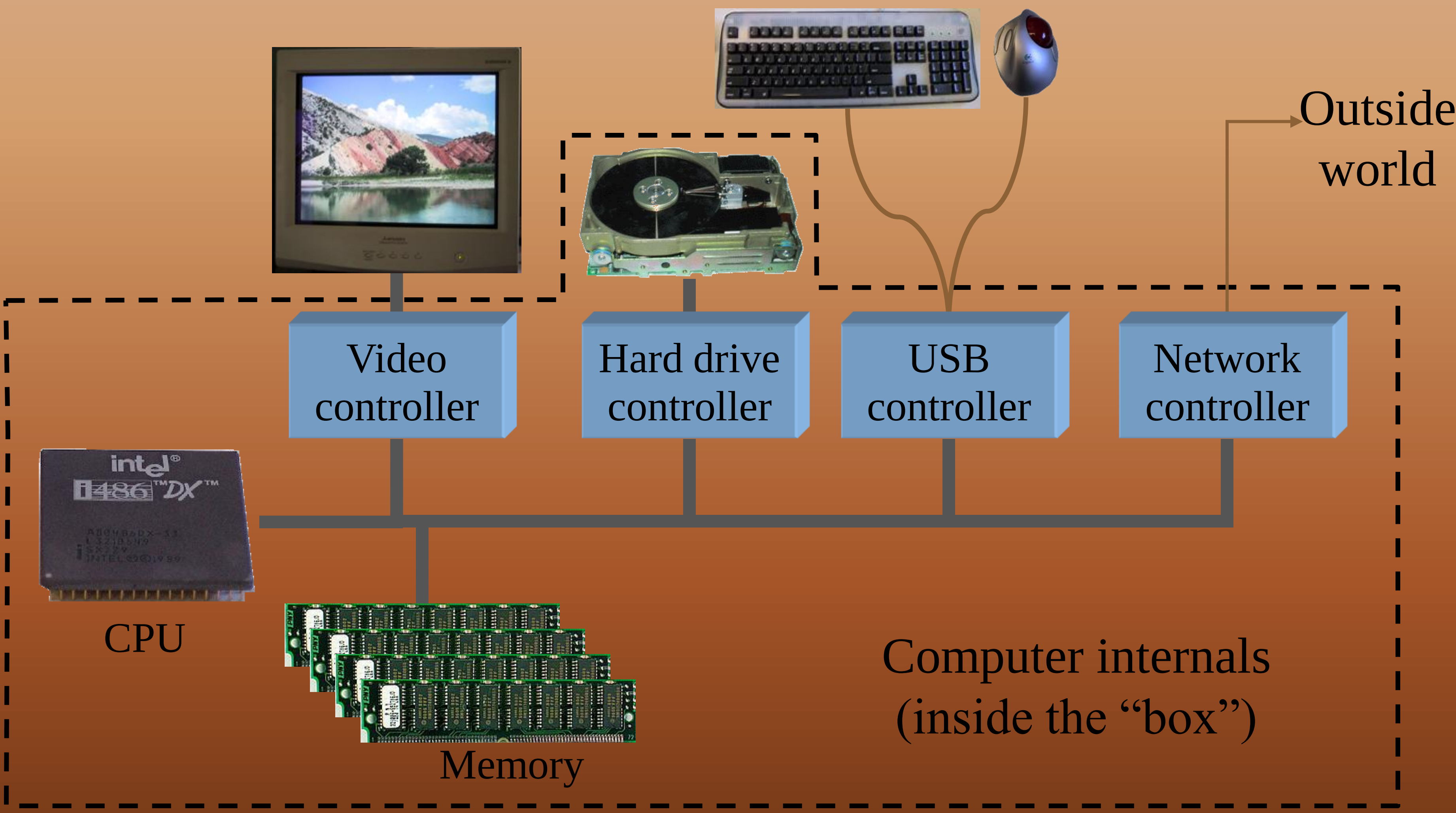
- Mainframe operating systems: MVS
- Server operating systems: FreeBSD, Solaris
- Multiprocessor operating systems: Cellular IRIX
- Personal computer operating systems: Windows, Unix
- Real-time operating systems: VxWorks
- Embedded operating systems
- Smart card operating systems

⇒Note..!! Some operating systems can fit into more than one category

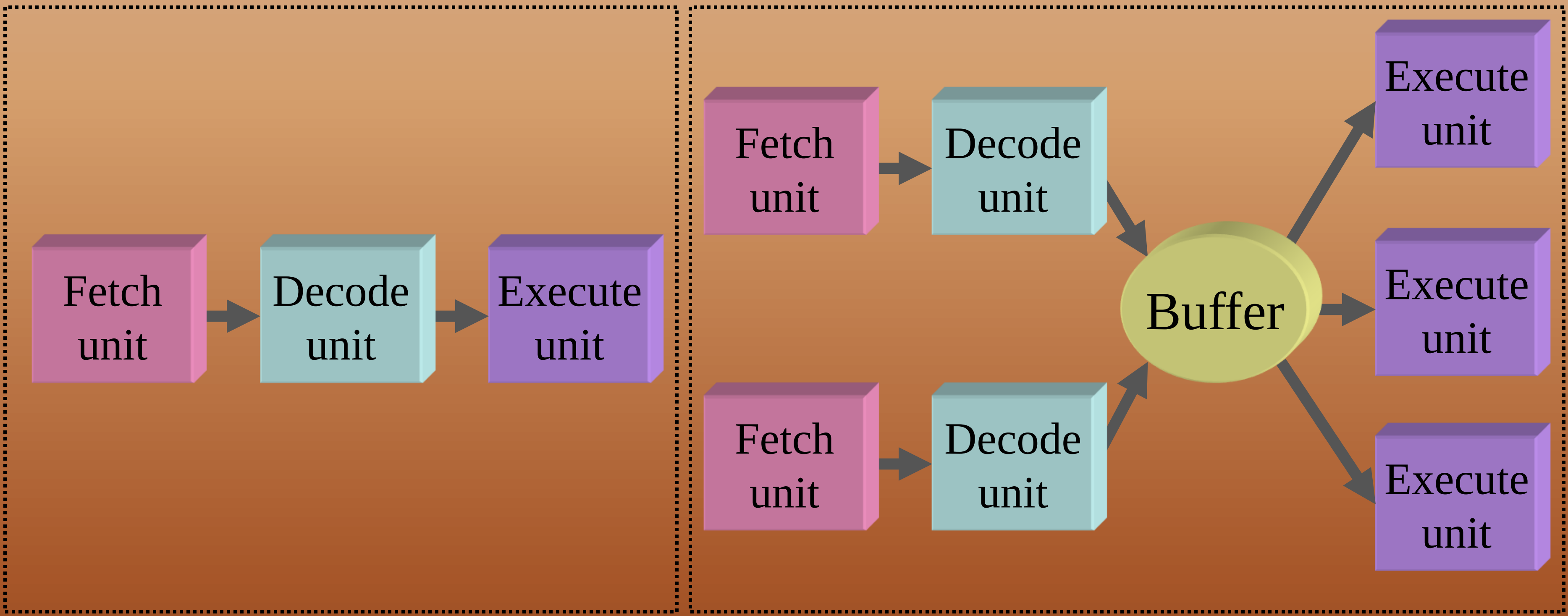
Why should I study Operating Systems?

- Need to understand interaction between the hardware and applications
  - New applications, new hardware..
  - Inherent aspect of society today
- Need to understand basic principles in the design of computer systems
  - efficient resource management, security, flexibility
- Increasing need for specialized operating systems
  - e.g. embedded operating systems for devices - cell phones, sensors and controllers
  - real-time operating systems - aircraft control, multimedia services

Components of a simple PC



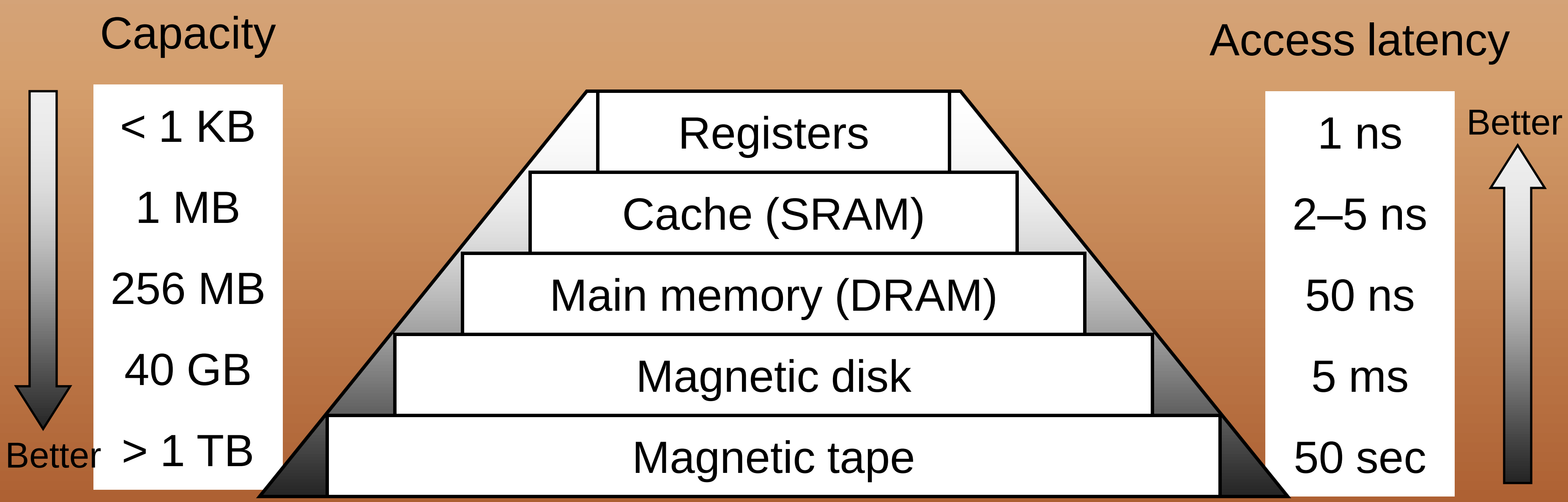
CPU internals



Pipelined CPU

Superscalar CPU

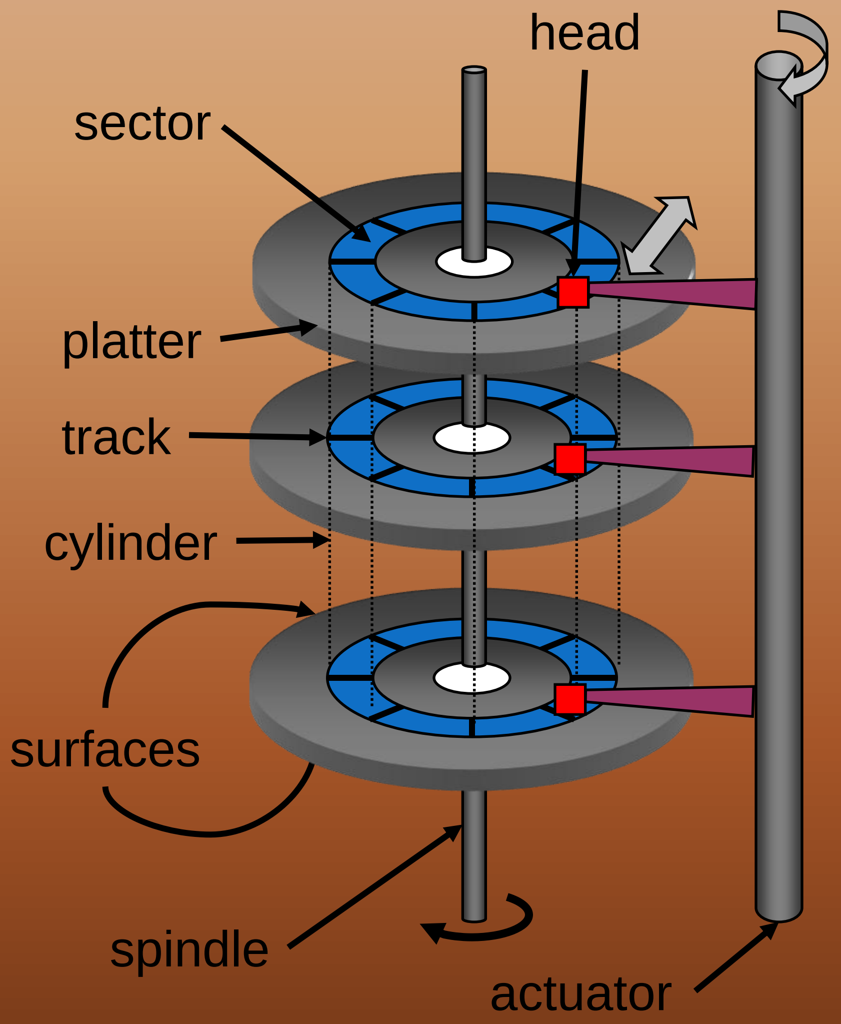
Storage pyramid



- Goal: really large memory with very low latency
  - Latencies are smaller at the top of the hierarchy
  - Capacities are larger at the bottom of the hierarchy
- Solution: move data between levels to create illusion of large memory with low latency

Disk drive structure

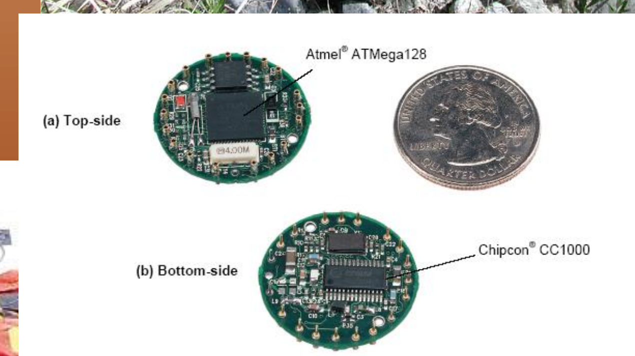
- Data stored on surfaces
  - Up to two surfaces per platter
  - One or more platters per disk
- Data in concentric tracks
  - Tracks broken into sectors
    - 256B-1KB per sector
  - Cylinder: corresponding tracks on all surfaces
- Data read and written by heads
  - Actuator moves heads
  - Heads move in unison

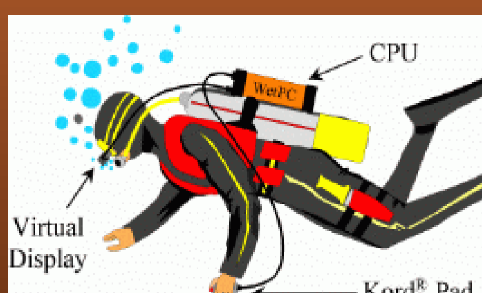


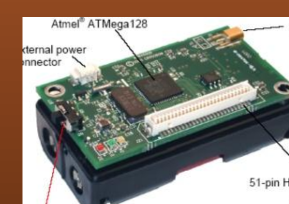
Systems Today





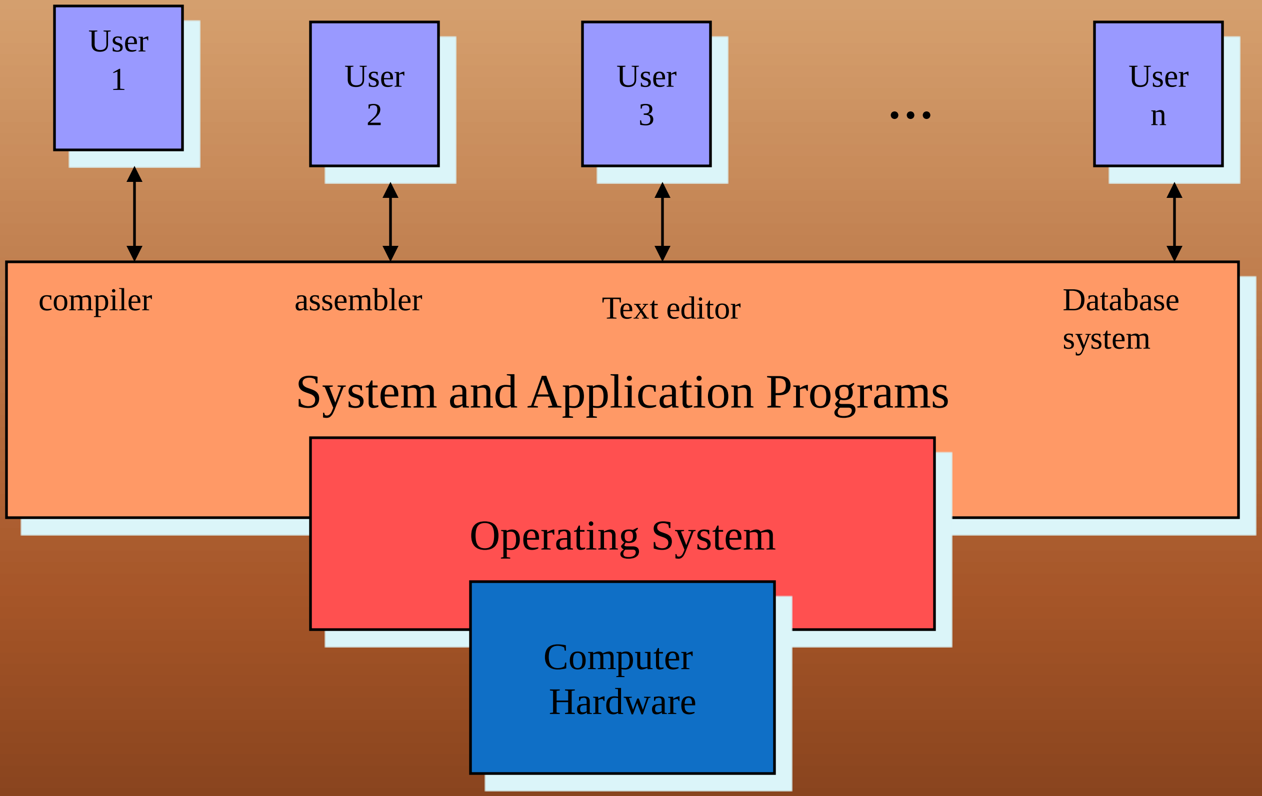



Computer System Components

- Hardware
  - Provides basic computing resources (CPU, memory, I/O devices).
- Operating System
  - Controls and coordinates the use of hardware among application programs.
- Application Programs
  - Solve computing problems of users (compilers, database systems, video games, business programs such as banking software).
- Users
  - People, machines, other computers

Abstract View of System



Operating System Views

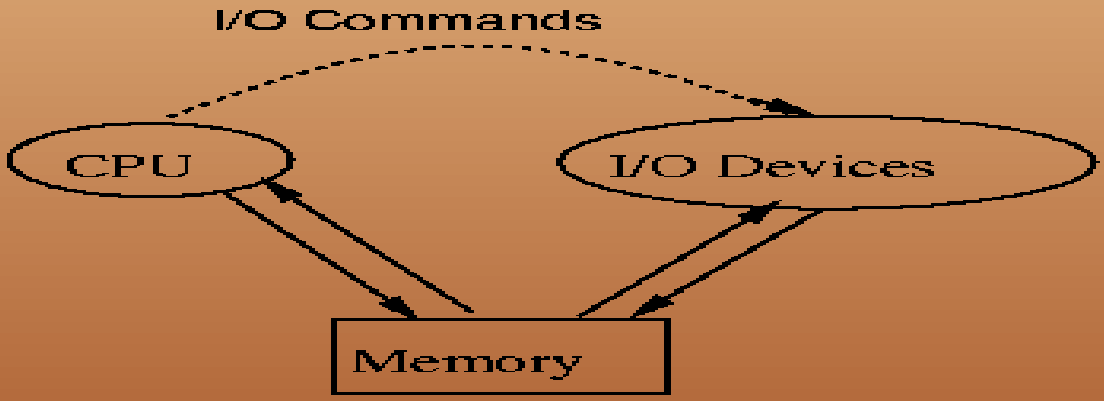
- Resource allocator
  - to allocate resources (software and hardware) of the computer system and manage them efficiently.
- Control program
  - Controls execution of user programs and operation of I/O devices.
- Kernel
  - The program that executes forever (everything else is an application with respect to the kernel).

- OS uses the **kernel mode** of the microprocessor, whereas other programs use the **user mode**.

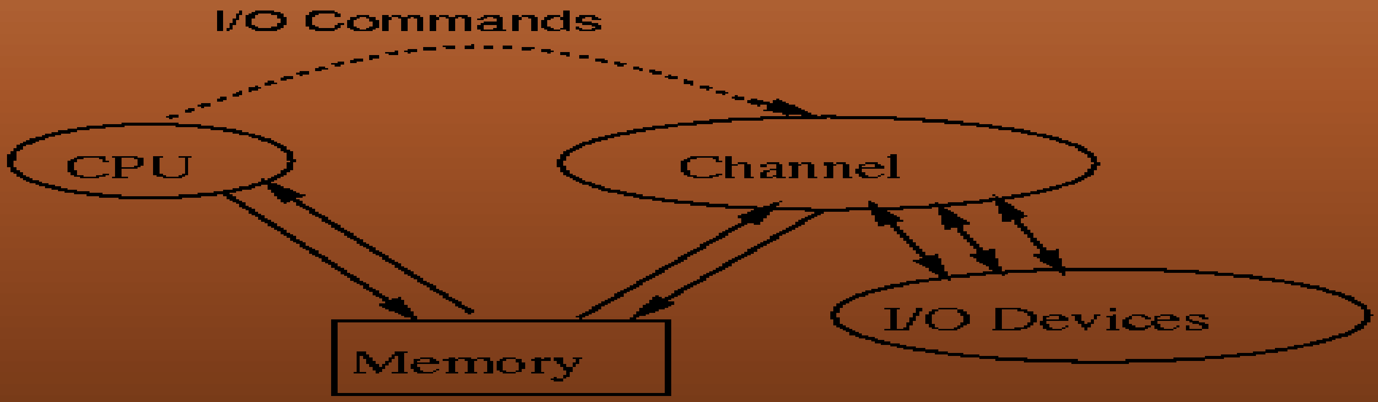
- The difference between two is that; all hardware instructions are valid in kernel mode, where some of them cannot be used in the user mode.

Speeding up I/O

- Direct Memory Access (DMA)



- Channels



Batch Systems - I/O completion

- How do we know that I/O is complete?
  - Polling:
    - Device sets a flag when it is busy.
    - Program tests the flag in a loop waiting for completion of I/O.
  - Interrupts:
    - On completion of I/O, device forces CPU to jump to a specific instruction address that contains the interrupt service routine.
    - After the interrupt has been processed, CPU returns to code it was executing prior to servicing the interrupt.

Multiprogramming

- Use interrupts to run multiple programs simultaneously
  - When a program performs I/O, instead of polling, execute another program till interrupt is received.
- Requires secure memory, I/O for each program.
- Requires intervention if program loops indefinitely.
- Requires CPU scheduling to choose the next job to run.

Timesharing

- Programs queued for execution in FIFO order.
- Like multiprogramming, but timer device interrupts after a quantum (timeslice).
  - Interrupted program is returned to end of FIFO
  - Next program is taken from head of FIFO
- Control card interpreter replaced by command language interpreter.

Timesharing (cont.)

- Interactive (action/response)
  - when OS finishes execution of one command, it seeks the next control statement from user.
- File systems
  - online filesystem is required for users to access data and code.
- Virtual memory
  - Job is swapped in and out of memory to disk.

Processor system types

- Multiprocessor systems are classified into two as tightly-coupled and loosely-coupled (distributed).
- In the tightly-coupled one, each processor is assigned a specific duty but processors work in close association, possibly sharing the same memory.
- In the loosely coupled one, each processor has its own memory and copy of the OS.

Parallel Systems

- Multiprocessor systems with more than one CPU in close communication.
- Improved Throughput, economical, increased reliability.
- Kinds:
    - Vector and pipelined
    - Symmetric and asymmetric multiprocessing
    - Distributed memory vs. shared memory
- Programming models:
    - Tightly coupled vs. loosely coupled ,message-based vs. shared variable

Distributed Systems

- Distribute computation among many processors.
- Loosely coupled -
    - no shared memory, various communication lines
- client/server architectures
- Advantages:
    - resource sharing
    - computation speed-up
    - reliability
    - communication - e.g. email
- Applications - digital libraries, digital multimedia

Real-time systems

- Correct system function depends on ti
- Feedback/control loops
- Sensors and actuators
- Hard real-time systems -
  - Failure if response time too long.
  - Secondary storage is limited
- Soft real-time systems -
  - Less accurate if response time is too long.
  - Useful in applications such as multimedia, virtual reality.