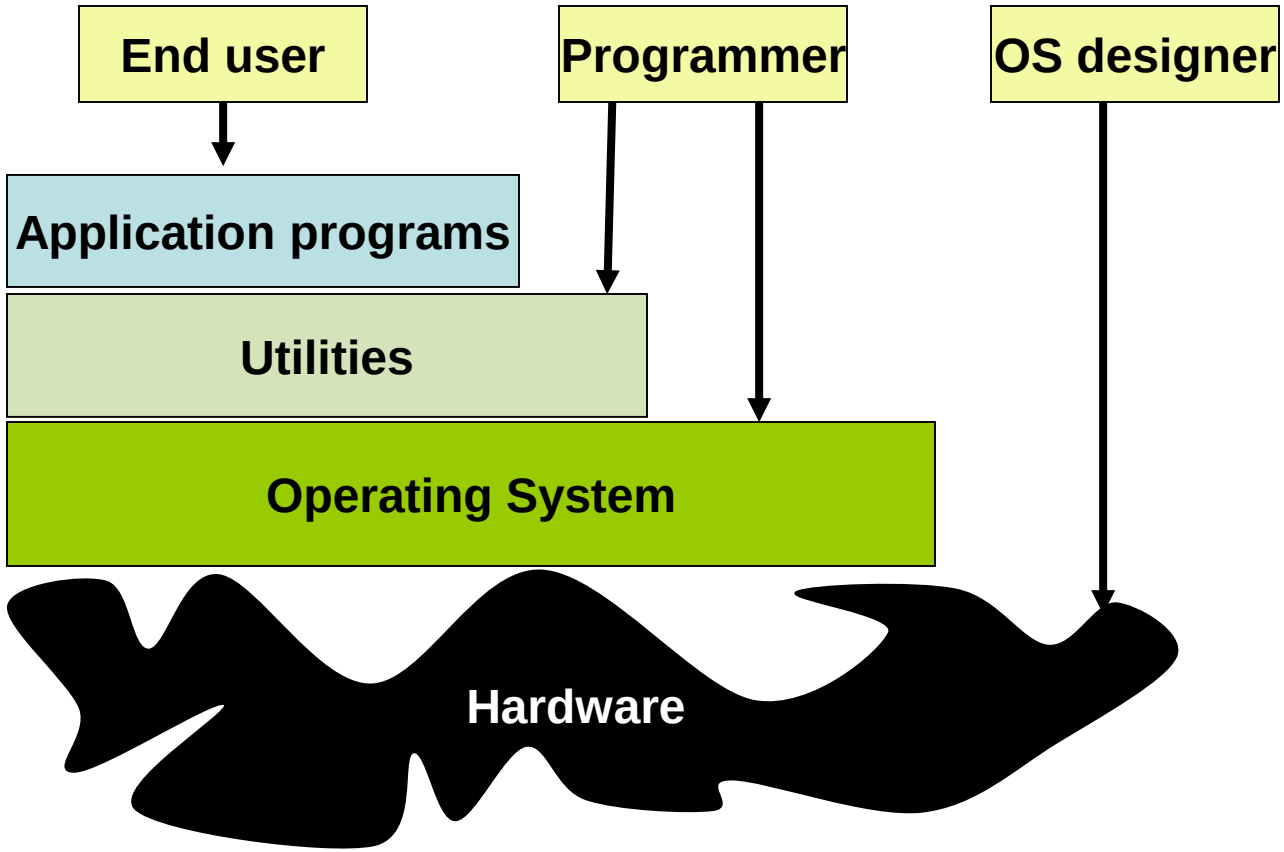
Chapter 1

Introduction to operating systems

What is an operating system?

- A program that
  - controls the execution of a program and
  - acts as an intermediary between a user of a computer and the computer hardware



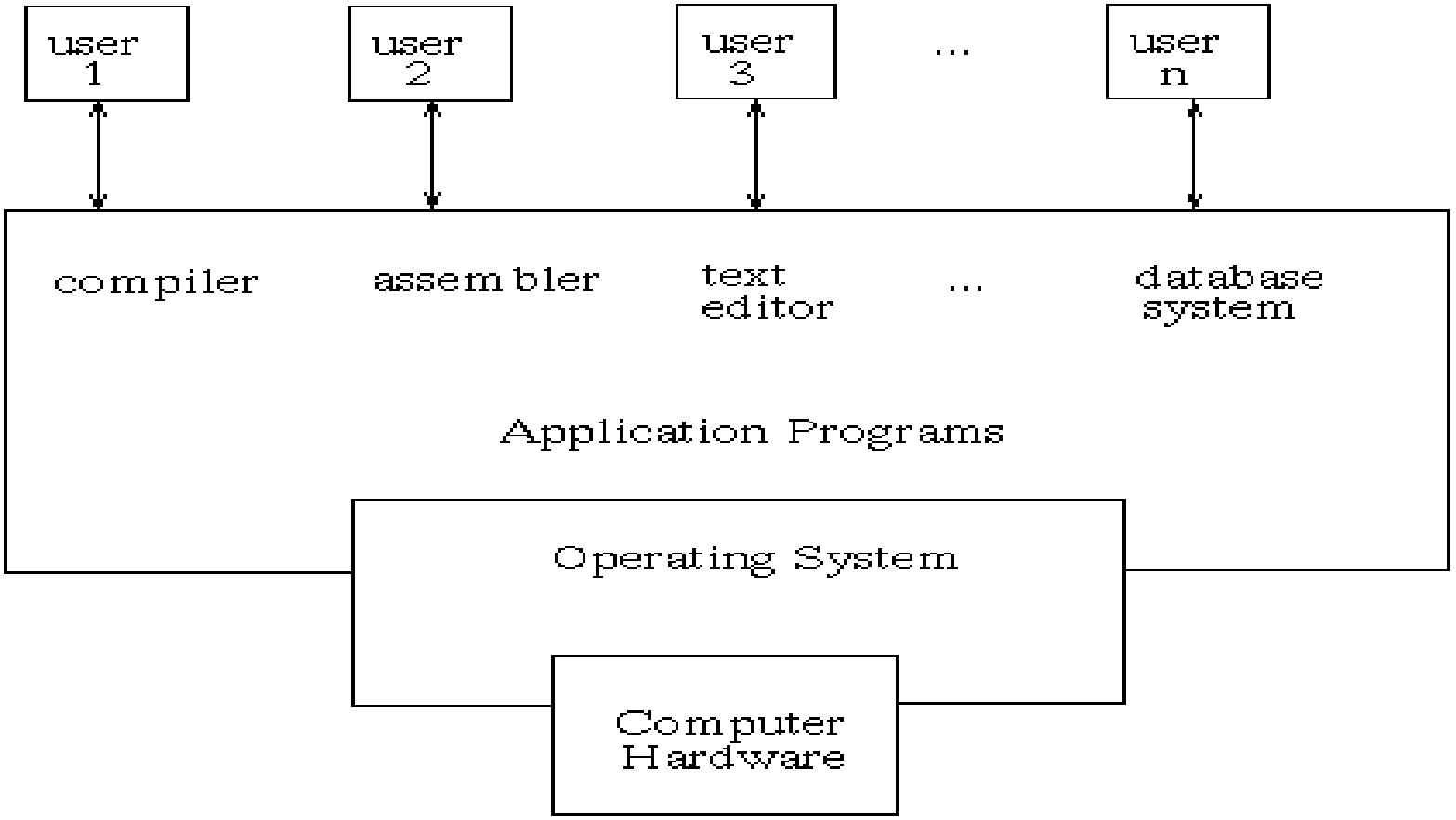
Operating System Goal

- Execute user programs and make solving user problems easier
- Make the computer system convenient to use
- Use the computer hardware in an efficient manner
- Resource allocator - manages and allocates resources.
- Control program - controls the execution of user programs and operation of I/O devices.
- Kernel - the one program running at all times (all else being application programs).

Computer System Component (external view)

- **Hardware -** provides basic computing resources (CPU, memory, I/O devices).
- **Operating system** - controls and coordinates the use of the hardware among the various application programs for the various users.
- **Applications programs** - define the ways in which the system resources are used to solve the computing problems of the users (compilers, database systems, video games, business pro grams).
- **Users** (people, machines, other computers).

Computer System ….



History of Operating Systems

- First generation 1945 - 1955
  - vacuum tubes, plug boards
  - Operating systems were unheard of
- Second generation 1955 - 1965
  - transistors, assembly and FORTRAN
  - Batch system was introduced
- Third generation 1965 – 1980
  - ICs and multiprogramming
- Fourth generation 1980 – present
  - personal computers

Early Systems - bare machine (early 1950s) - First Generation

- Structure
  - Large machines run from console
  - Single user system
  - Programmer/User as operator
  - Paper tape or punched cards
- Secure
- Inefficient use of expensive resources
  - Low CPU utilization
  - Significant amount of setup time

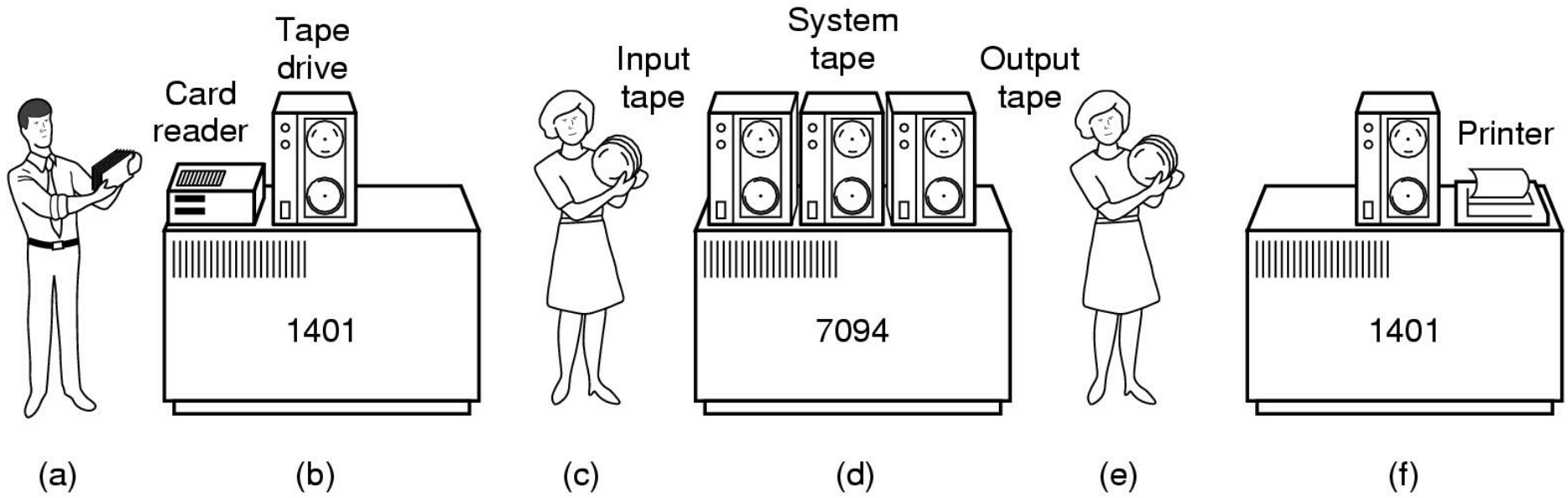
Second Generation

- Focused on cost effectiveness
- Computers were expensive
  - IBM 7094: $200,000
- Two widely adopted improvements
  - Computer operators: humans hired to facilitate machine operation
  - Concept of job scheduling: group together programs with similar requirements
- Expensive time lags between CPU and I/O devices

Simple Batch Systems - Second Generation

- Use an operator (somebody to work the machine)
- Add a card reader (a device to read programs written on punched cards)
- Reduce setup time by batching similar jobs
- Automatic job sequencing - automatically transfers control from one job to another. First rudimentary operating system
- Resident monitor
  - initial control in monitor
  - control transfers to job
  - when job completes control transfers back to monitor

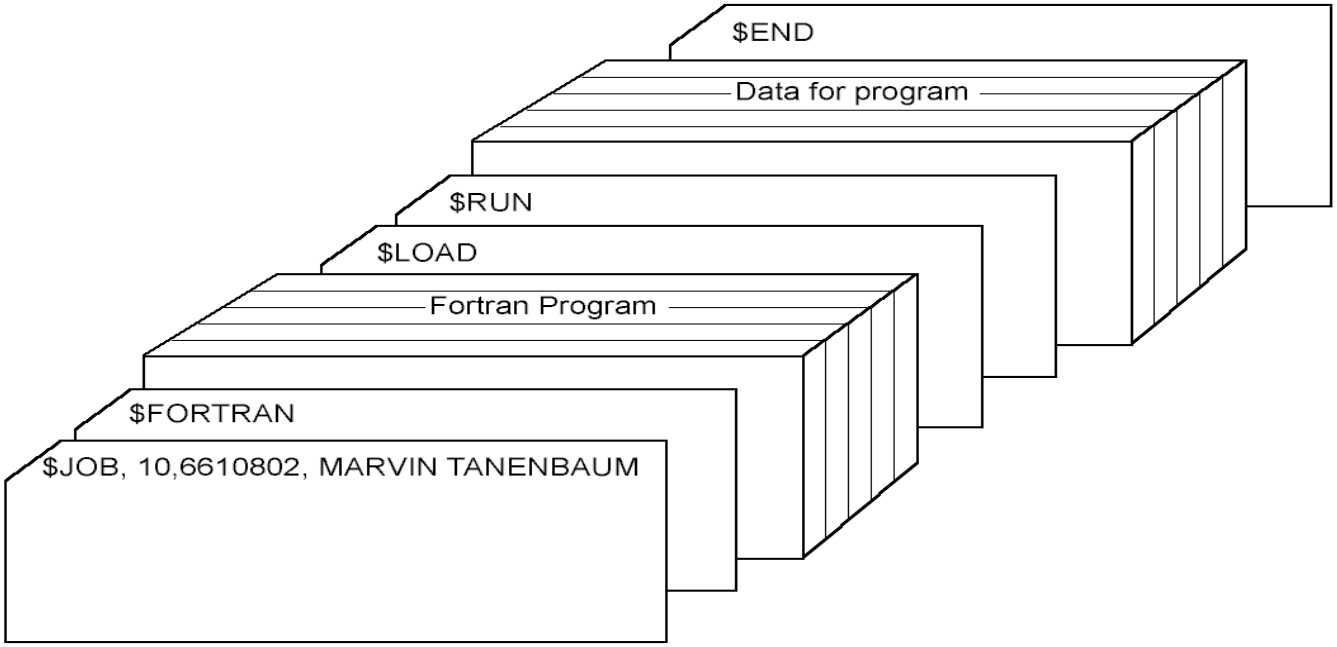
Second Generation…..



Early batch system

- bring cards to 1401
- read cards to tape
- put tape on 7094 which does computing
- put tape on 1401 which prints output

Second Generation….



- Structure of a typical FMS job Typical operating systems
  - FMS (the Fortran Monitor System)
  - IBSYS
  - IBM’s operating system for the 7094

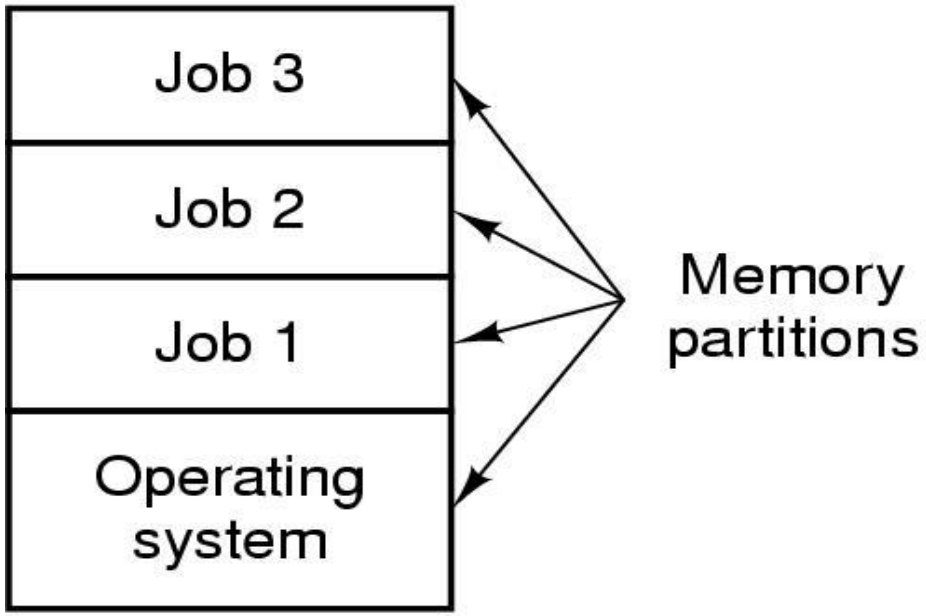
Second Generation….

- Parts of resident monitor
  - Control card interpreter - responsible for reading and carrying out instructions on the cards.
  - Loader - loads systems programs and applications programs into memory.
  - Device drivers - know special characteristics and properties for each of the system's I/O devices.
- Problem: Slow Performance - since I/O and CPU could not overlap, and card reader very slow.
- Solution: Off-line operation - speed up computation by loading jobs into memory from tapes and card reading and line printing done off-line using smaller machines.

Third Generation

- Faster CPUs
- Speed caused problems with slower I/O devices
- Multiprogramming
  - Allowed loading many programs at one time
- Program scheduling
  - Initiated with second-generation systems
  - Continues today
- Few advances in data management
- Total operating system customization
  - Suit user's needs

Third Generation….



Major achievements of third generation:

- Multiprogramming
- Timesharing
- MULTICS => UNIX

Fourth Generation

- Faster CPUs
- Speed caused problems with slower I/O devices
- Main memory physical capacity limitations
  - Multiprogramming schemes used to increase CPU
  - Virtual memory developed to solve physical limitation
- Database management software
  - Became a popular tool
- A number of query systems introduced
- Programs started using English-like words, modular structures, and standard operations

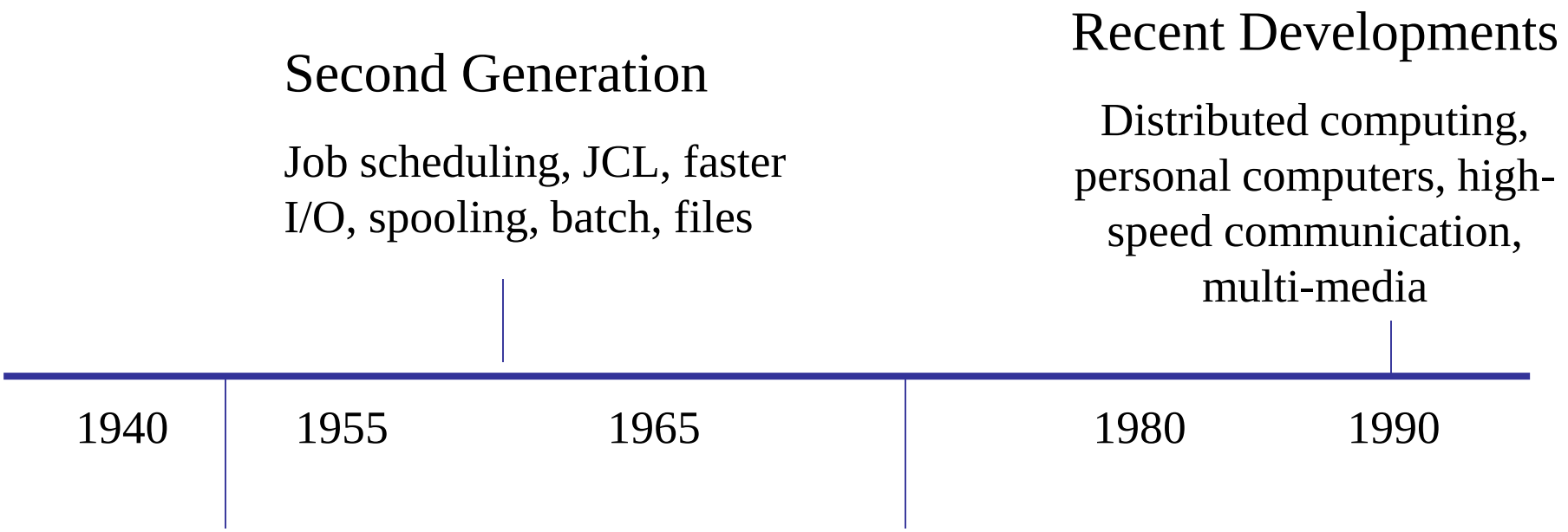
Fourth Generation….

- **Cost/performance ratio** improvement of computer components
- More flexible hardware (firmware)
- **Multiprocessing**
  - Allowed parallel program execution
- Evolution of personal computers
- Evolution of high-speed communications
- **Distributed processing** and **networked systems** introduced

Fourth Generation…..

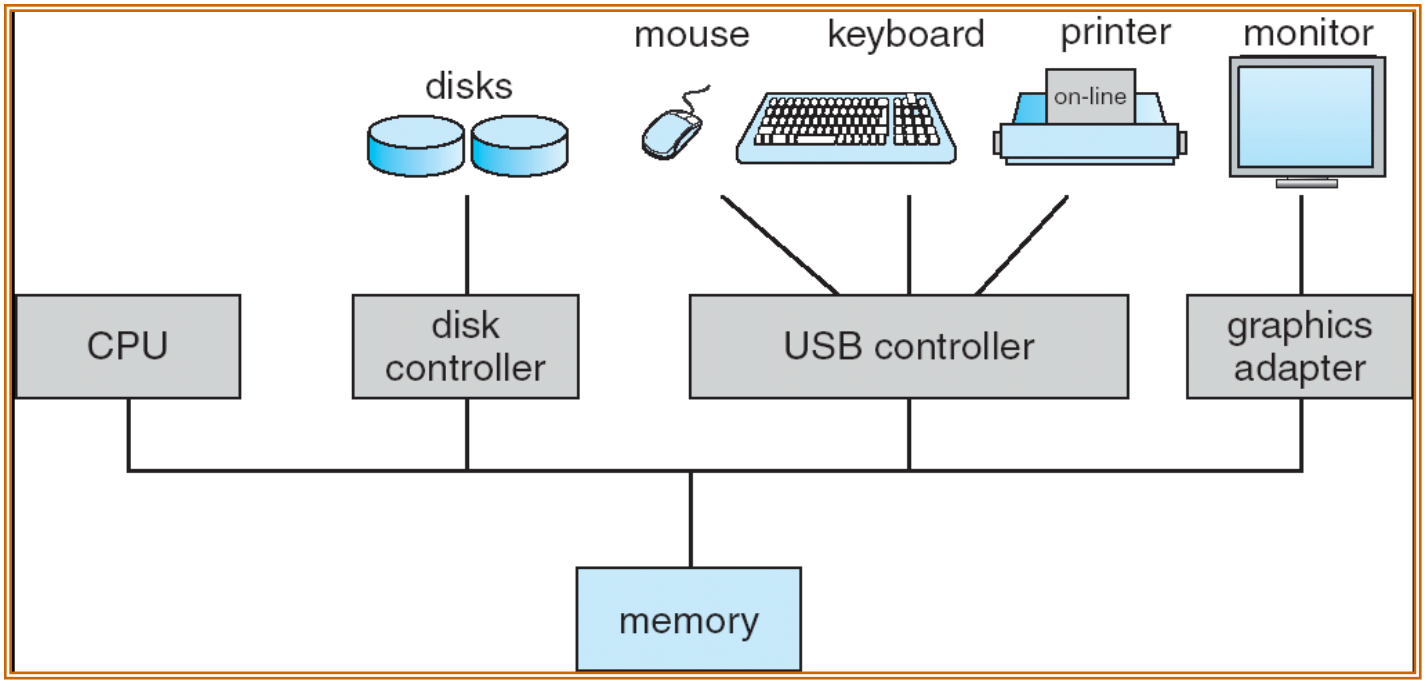
- Demand for Internet capability
  - Sparked proliferation of networking capability
  - Increased networking
  - Increased tighter security demands to protect hardware and software
- Multimedia applications
  - Demanding additional power, flexibility, and device compatibility for most operating systems

History of Operating Systems



Computer System Overview

- 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



Basic Elements

- Processor
- Main Memory
  - volatile
  - referred to as real memory or primary memory
- I/O modules
  - secondary memory devices
  - communications equipment
  - terminals
- System bus
  - communication among processors, memory, and I/O modules

Top-Level Components

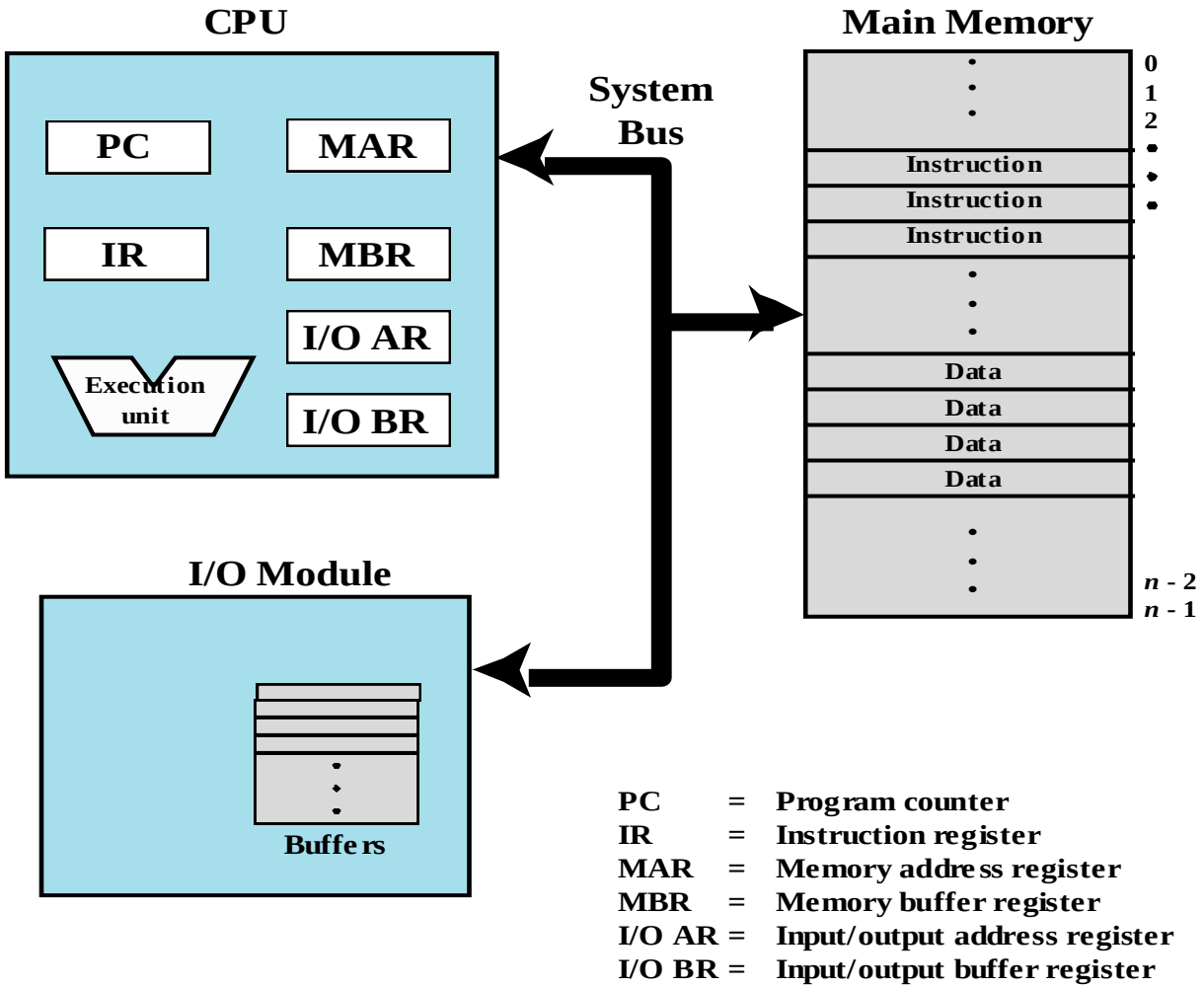


Figure 1.1 Computer Components: Top-Level View

Processor Registers

- User-visible registers
  - Enable programmer to minimize main-memory references by optimizing register use
- Control and status registers
  - Used by processor to control operating of the processor
  - Used by privileged operating-system routines to control the execution of programs

User-Visible Registers

- May be referenced by machine language
- Available to all programs - application programs and system programs
- Types of registers(Examples)
  - General Purpose Register(Ax,Bx,Cx,Dx)
  - Address
    - Index
    - Segment pointer
    - Stack pointer

Control and Status Registers

- Program Counter (PC)
  - Contains the address of an instruction to be fetched
- Instruction Register (IR)
  - Contains the instruction most recently fetched
- Program Status Word (PSW)
  - Condition codes
  - Interrupt enable/disable
  - Supervisor/user mode
- Condition Codes or Flags
  - Bits set by the processor hardware as a result of operations
  - Examples
    - Positive result
    - Negative result
    - Zero
    - Overflow

Processor

- Internal registers
  - Memory address register (MAR)
    - Specifies the address for the next read or write
  - Memory buffer register (MBR)
    - Contains data written into memory or receives data read from memory
  - I/O address register
  - I/O buffer register

Other Registers /Internal registers used by CPU

- Exchanges data with the memory and I/O
  - Uses the following registers to exchange data with the memory
    - **Memory address register (MAR)**
      - Specifies the address of memory for the next read or write
    - **Memory buffer register (MBR)**
      - Holds the data to be written into the memory
  - Holds the data read from the memory Uses the following registers to exchange data with I/O module
    - **I/O address register (I/OAR)**
      - Specifies a particular I/O device
    - **I/O buffer register (I/OBR)**
      - Holds the data to be exchanged between an I/O module and CPU

Instruction Execution

- Two steps
  - Processor reads instructions from memory
    - Fetches
  - Processor executes each instruction

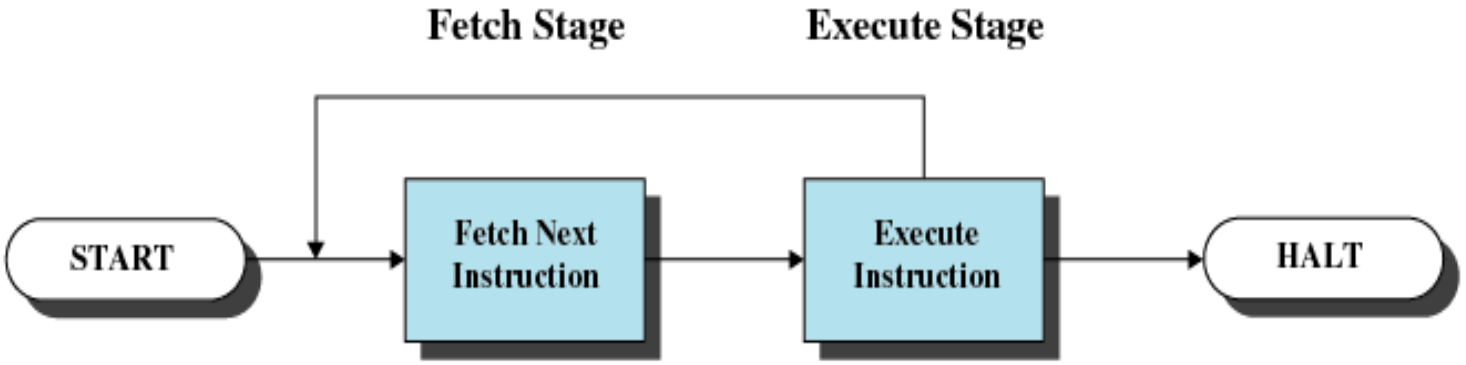


Figure 1.2 Basic Instruction Cycle

Instruction Fetch and Execute

- The processor fetches the instruction from memory
- The fetched instruction is loaded to IR
- Program counter (PC) holds address of the instruction to be fetched next
- Program counter is incremented after each fetch to point to the next instruction

Instruction Register

- Categories of instruction
  - Processor-memory
    - Transfer data between processor and memory
  - Processor-I/O
    - Data transferred to or from a peripheral device
  - Data processing
    - Arithmetic or logic operation on data
  - Control
    - Alter sequence of execution

Interrupts

- Interrupt the normal sequencing of the processor
- Most I/O devices are slower than the processor
  - Processor must pause to wait for device

Program	Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, and reference outside a user's allowed memory space.
Timer	Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.
I/O	Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions.
Hardware failure	Generated by a failure, such as power failure or memory parity error.

Fig: Classes of interrupts

Common Functions of Interrupts

- Interrupt transfers control to the interrupt service routine generally, through the *interrupt vector*, which contains the addresses of all the service routines.
- Interrupt architecture must save the address of the interrupted instruction.
- Incoming interrupts are *disabled* while another interrupt is being processed to prevent a *lost interrupt*.
- A *trap* is a software-generated interrupt caused either by an error or a user request.
- An operating system is *interrupt* driven.

Interrupt Handling

- When CPU is interrupted, it stops what it is doing and immediately transfers execution to a fixed location.
- The operating system preserves the state of the CPU by storing registers and the program counter.
- Determines which type of interrupt has occurred:
  - *polling*
  - *vectored* interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt

Interrupts

- Suspends the normal sequence of execution

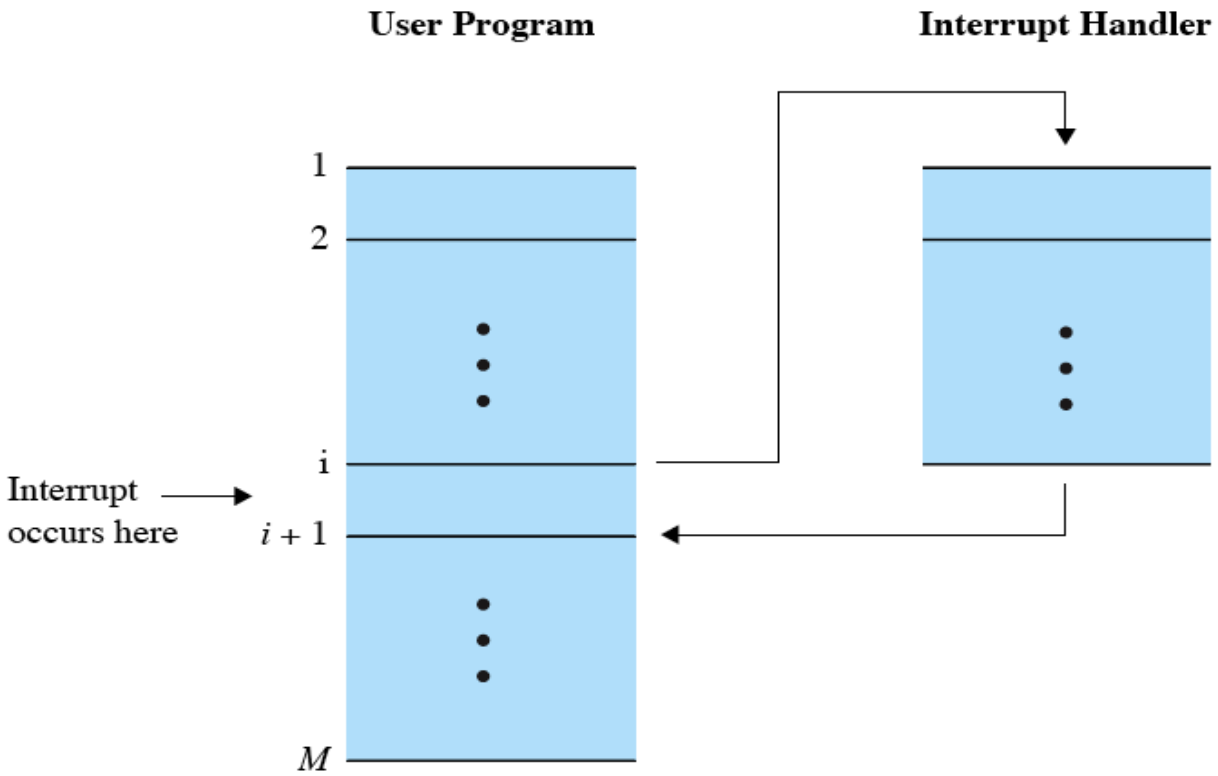
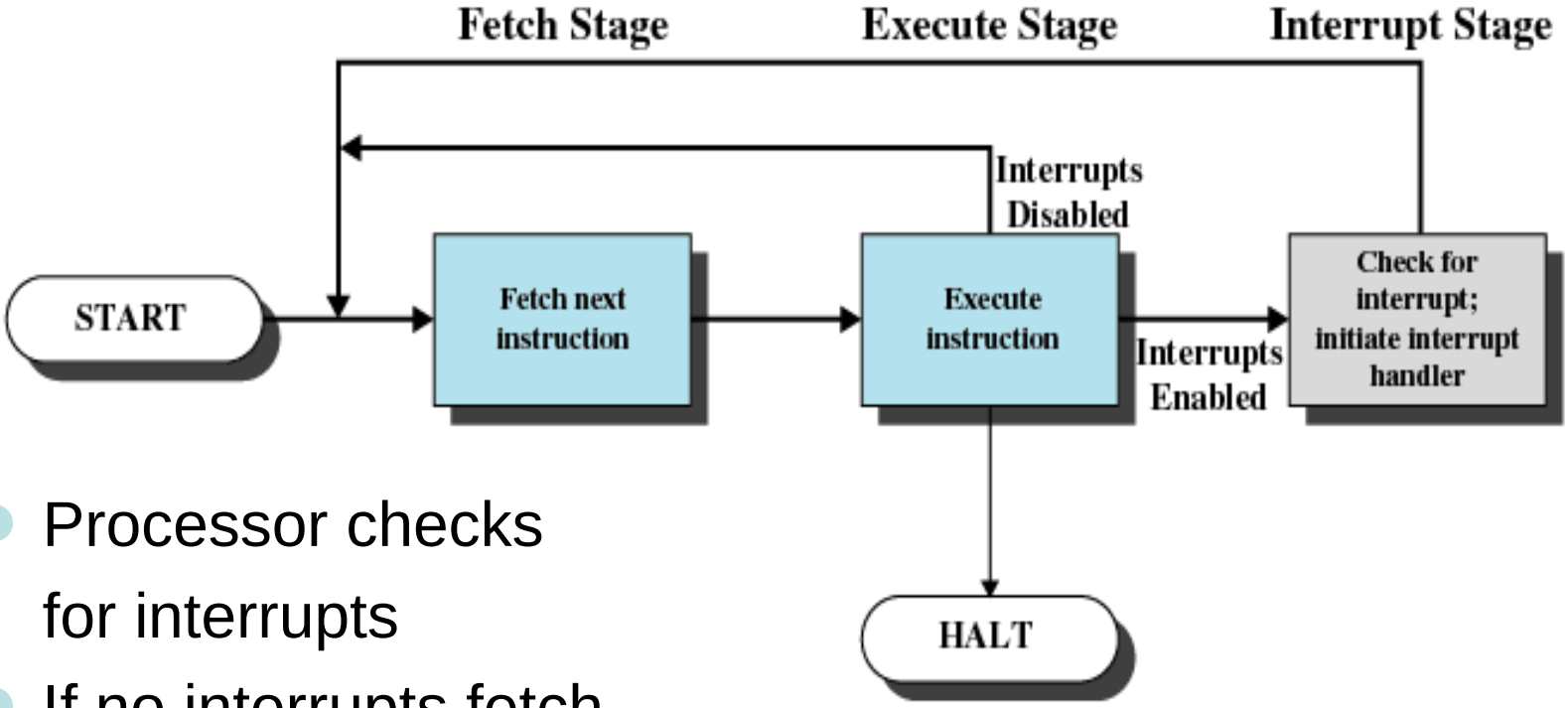


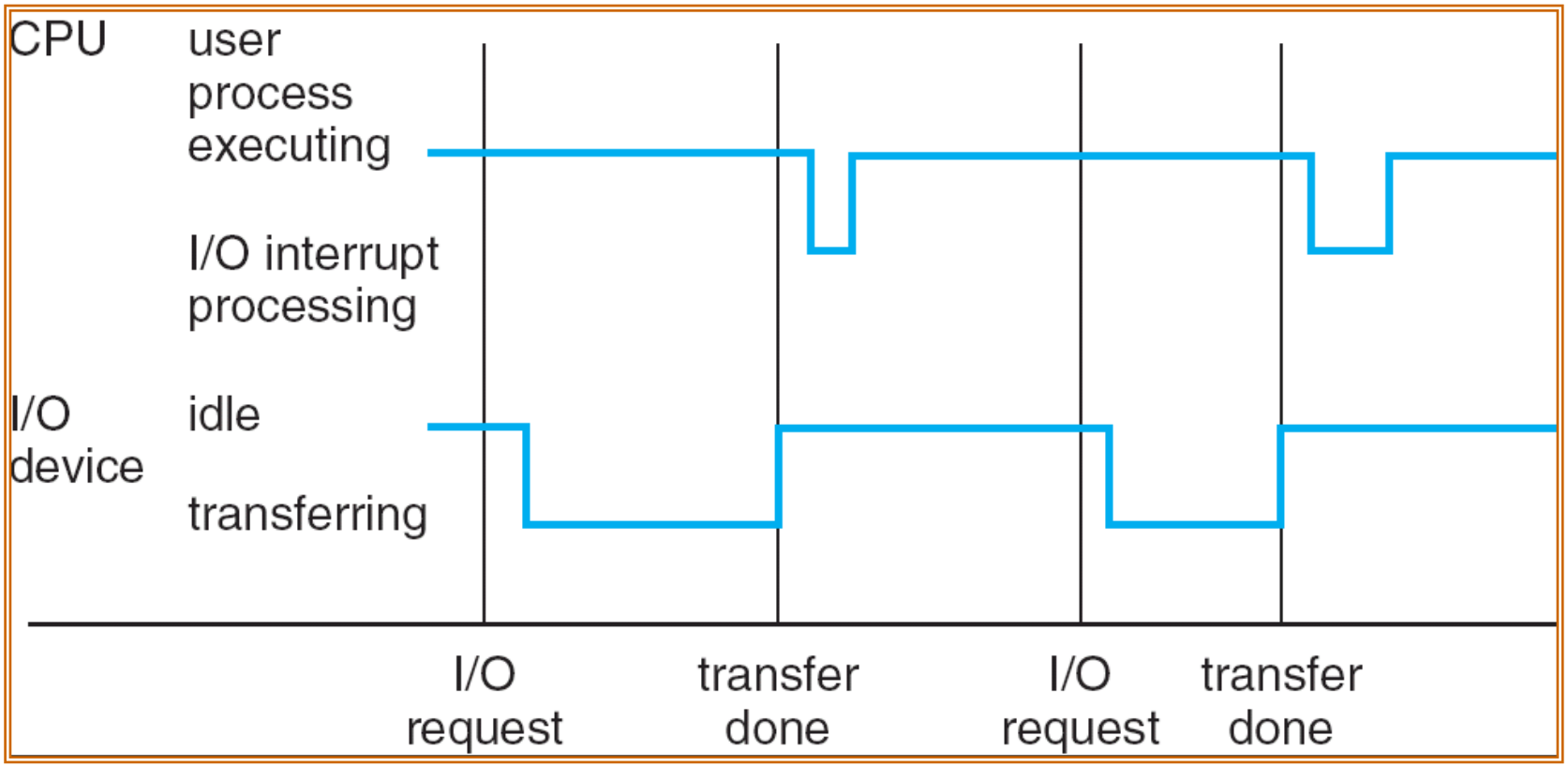
Figure 1.6 Transfer of Control via Interrupts

Interrupt Cycle



- Processor checks for interrupts
- If no interrupts fetch the next instruction for the current program
- If an interrupt is pending, suspend execution of the current program, and execute the interrupt-handler routine

Interrupt Timeline



I/O Structure

Synchronous

- After I/O starts, control returns to user program only upon I/O completion.
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access).
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing.

I/O Structure…..

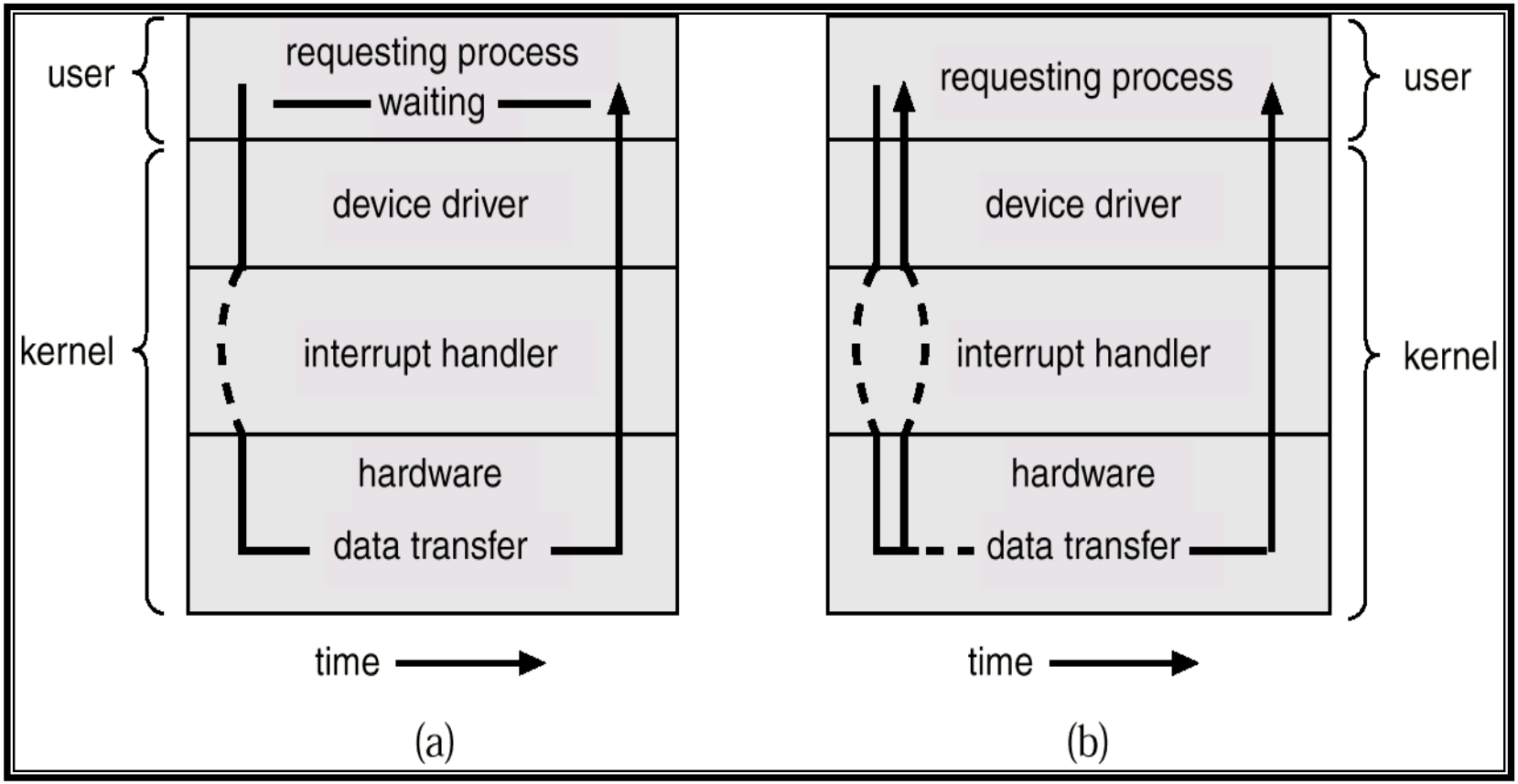
Asynchronous

- After I/O starts, control returns to user program without waiting for I/O completion.
  - *System call* – request to the operating system to allow user to wait for I/O completion.
  - *Device-status table* contains entry for each I/O device indicating its type, address, and state.
  - Operating system indexes into I/O device table to determine device status and to modify table entry to include interrupt.

Two I/O Methods

Synchronous

Asynchronous



Storage Structure

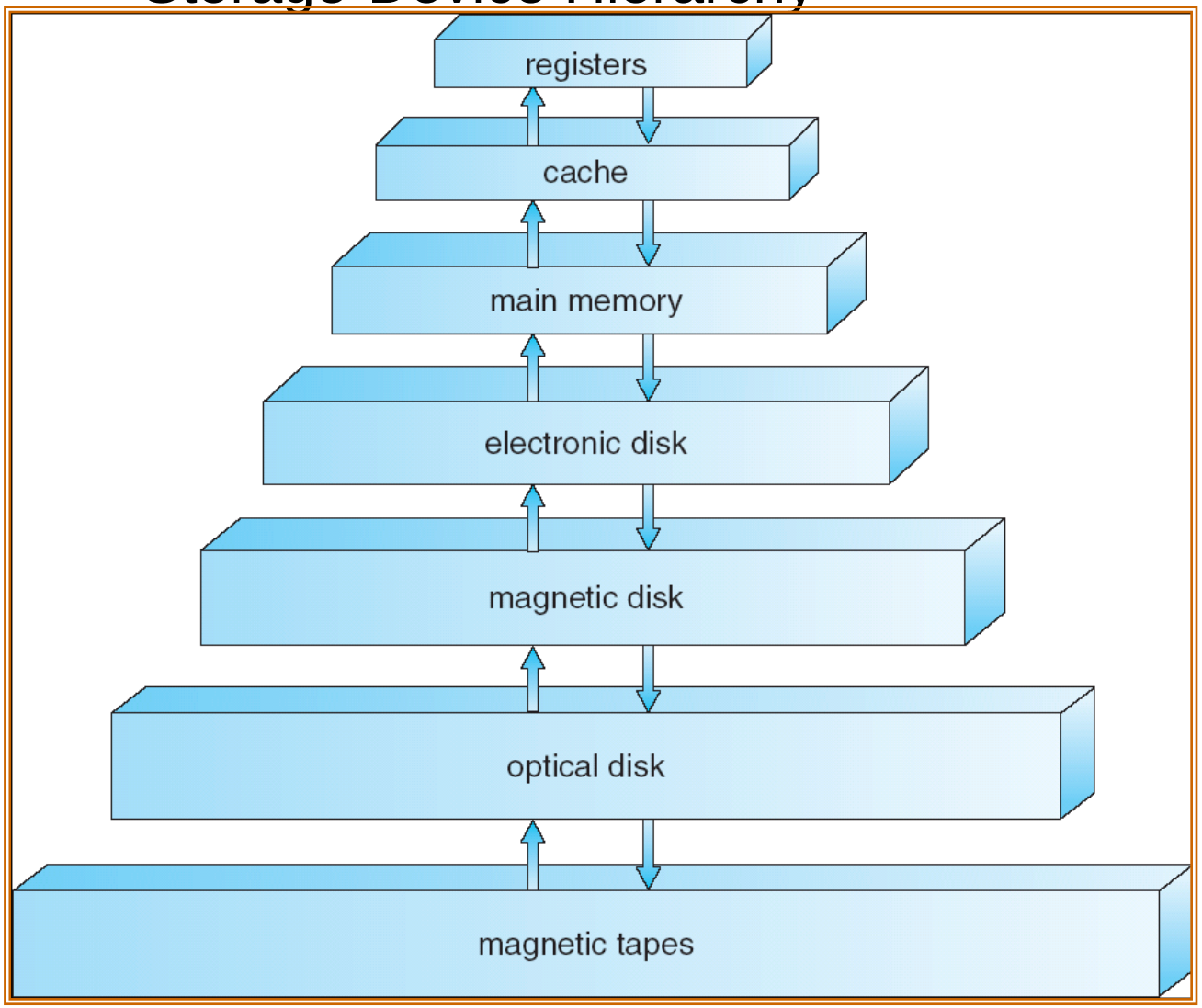
- Main memory – the only large storage media that the CPU can access directly.
- Secondary storage – extension of main memory that provides large nonvolatile storage capacity.
- Magnetic disks – rigid metal or glass platters covered with magnetic recording material
  - Disk surface is logically divided into *tracks*, which are subdivided into *sectors*.
  - The *disk controller* determines the logical interaction between the device and the computer.

Cont'd….

- Storage systems organized in hierarchy.
  - Speed
  - Cost
  - Volatility
- *Caching* – copying information into faster storage system; main memory can be viewed as a last *cache* for secondary storage.

Cont'd….

Storage-Device Hierarchy

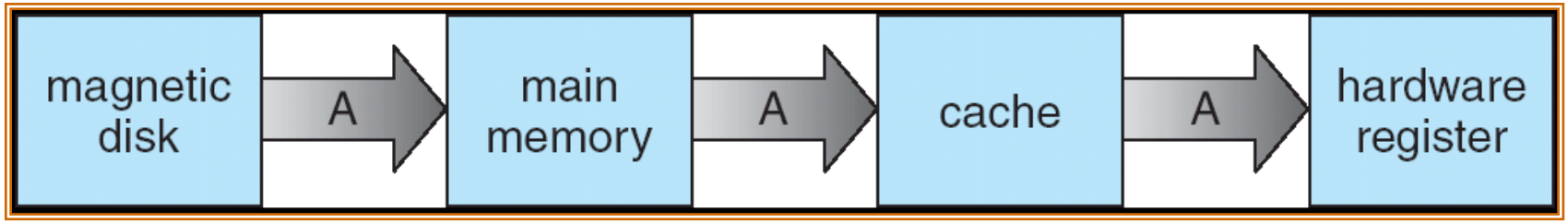


Caching

- 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

Migration of Integer A from Disk to Register

- Multitasking environments must be careful to use most recent value, not matter where it is stored in the storage hierarchy



- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
  - Several copies of a datum can exist

Operating system overview

Operating System Objectives

- Convenience
  - Makes the computer more convenient to use
- Efficiency
  - Allows computer system resources to be used in an efficient manner
- Ability to evolve
  - Permit effective development, testing, and introduction of new system functions without interfering with service

Services Provided by the OS

- Program development
  - Utility programs like editors and debuggers
- Program execution
  - Handles the scheduling duties like loading instructions and data into main memory, initializing I/O devices and files, and preparing other resources must be prepared
- Access to I/O devices
  - OS provides a uniform interface that hides details of I/O operations to the devices so that programmers can access them using simple reads and writes.

Cont'd….

- Controlled access to files
  - Reflects detailed understanding structure of the data contained in the files on the storage medium
  - In case of multiple users, the OS may provide protection mechanisms to control access to the files
- System access
  - Provides protection of resources and data from unauthorized users and must resolve conflicts for resource contention

Cont'd….

- Error detection and response
  - Errors that may occur
    - Internal and external hardware errors
      - Memory error
      - Device failure
    - Software errors
      - Arithmetic overflow
      - Access forbidden memory locations
    - Operating system cannot grant request of application
  - OS must provide a response that clears the error condition with the least impact on running applications
  - The response may include:
    - ending the program that caused the error
    - retrying the operation or
    - simply reporting the error to the application.

Cont'd….

- Accounting
  - Collect usage statistics
  - Monitor performance
  - Used to anticipate future enhancements
  - Used for billing purposes

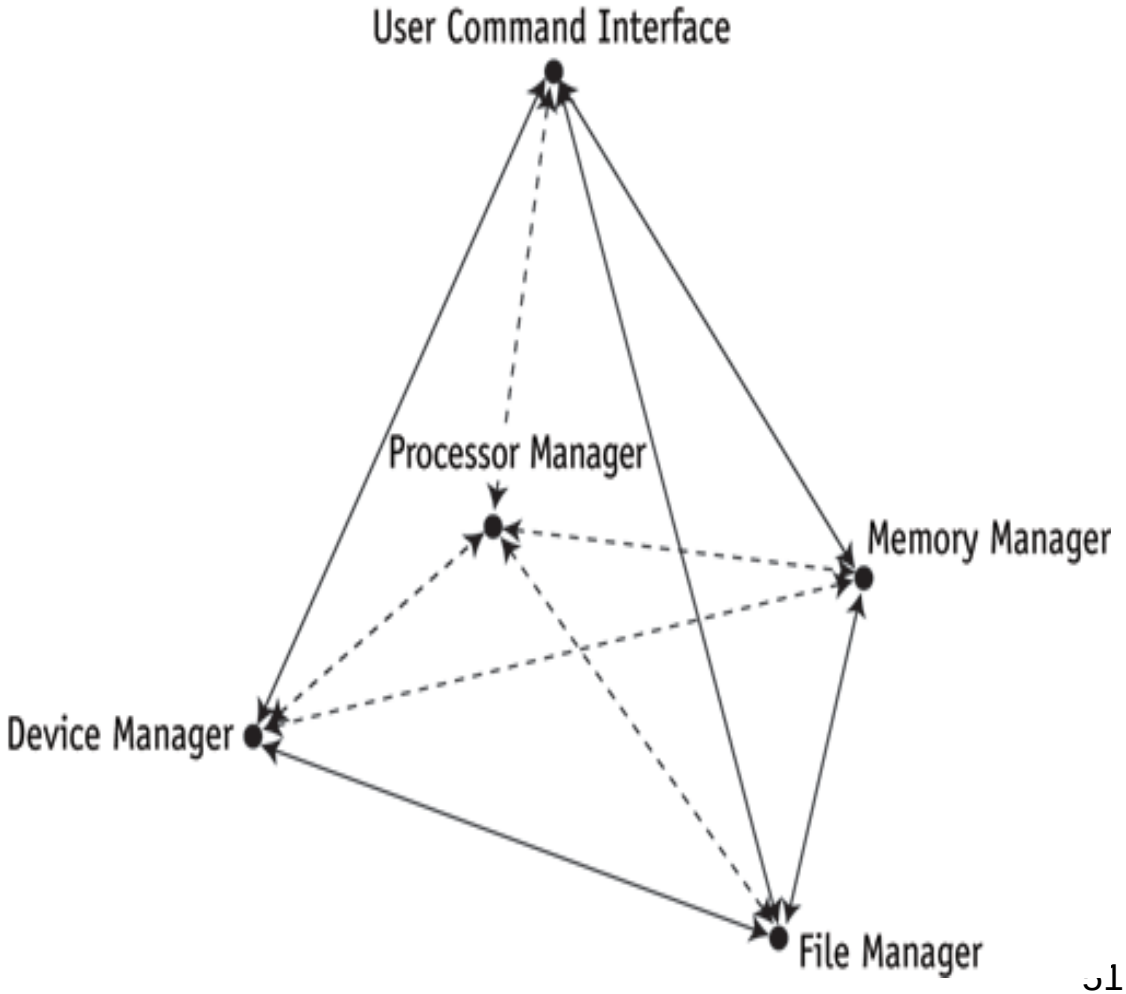
Operating System Subsystems

- OS includes five essential subsystem managers
  - Memory Manager
  - Processor Manager
  - Device Manager
  - File Manager
  - Network Manager

OS Subsystems…..

(figure 1.1)

This model of a non-networked operating system shows four subsystem managers supporting the User Command Interface.



OS Subsystems…..

- Each manager:
  - Works closely with other managers
  - Performs a unique role
- Manager tasks
  - Monitor its resources continuously
  - Enforce policies determining:
    - Who gets what, when, and how much
  - Allocate the resource (when appropriate)
  - Deallocate the resource (when appropriate)

OS Subsystems…..

- **Network Manager**
  - Operating systems with networking capability
  - Fifth essential manager
  - Convenient way for users to share resources
  - Retains user access control
  - In all modern operating systems
  - Assumes responsibility for networking tasks

OS Subsystems…..

- Memory Manager
  - In charge of main memory
    - Random Access Memory (RAM)
  - Responsibilities include:
    - Preserving space in main memory occupied by operating system
    - Checking validity and legality of memory space request
    - Setting up memory tracking table
      - Tracks usage of memory by sections
      - Needed in multiuser environment
    - Deallocating memory to reclaim it

OS Subsystems….

- Processor manager
  - In charge of allocating **Central Processing Unit** (CPU)
  - Tracks **process** status
    - An instance of program execution
  - Two levels of responsibility:
    - Handle jobs as they enter the system
      - Handled by Job Scheduler
    - Manage each process within those jobs
      - Handled by Process Scheduler

OS Subsystems…..

- Device manager
  - In charge of monitoring all resources
    - Devices, channels, and control units
  - Responsibilities include:
    - Choosing most efficient resource allocation method
      - Printers, ports, disk drives, etc.
      - Based on scheduling policy
    - Allocating the device
    - Starting device operation
    - Deallocating the device

OS Subsystems…..

- File manager
  - In charge of tracking every file in the system
    - Data files, program files, compilers, application programs
  - Responsibilities include:
    - Enforcing user/program resource access restrictions
      - Uses predetermined access policies
    - Controlling user/program modification restrictions
      - Read-only, read-write, create, delete
    - Allocating resource
      - Opening the file
      - Deallocating file (by closing it)

OS Subsystems……

- **Cooperation issue**
  - Essential manager
    - Perform individual tasks *and*
    - Harmoniously interact with other managers
      - Requires incredible precision
    - No single manager performs tasks in isolation
    - Network manager
      - Convenient way to share resources
      - Controls user access
- **Resources include**:
  - Hardware (CPUs, memory areas, printers, tape drives, modems, and disk drives)
  - Software (compilers, application programs, and data files)

Ease of Evolution of OS

- Hardware upgrade plus new types of HW
- New services
  - In response to demands of users or system managers
  - Examples:
    - Maintain good performance
    - Applications require windows
- Fixes
  - Any operating system has faults
  - A fix to a fault my introduce another fault

Process

- The concept of process is fundamental to the structure of operating systems
- **Definitions:**
  - A program in execution
  - An instance of a program running on a computer
  - The entity that can be assigned to and executed on a processor
  - A unit of activity characterized by a single sequential thread of execution, a current state, and an associated set of system resources

Process…..

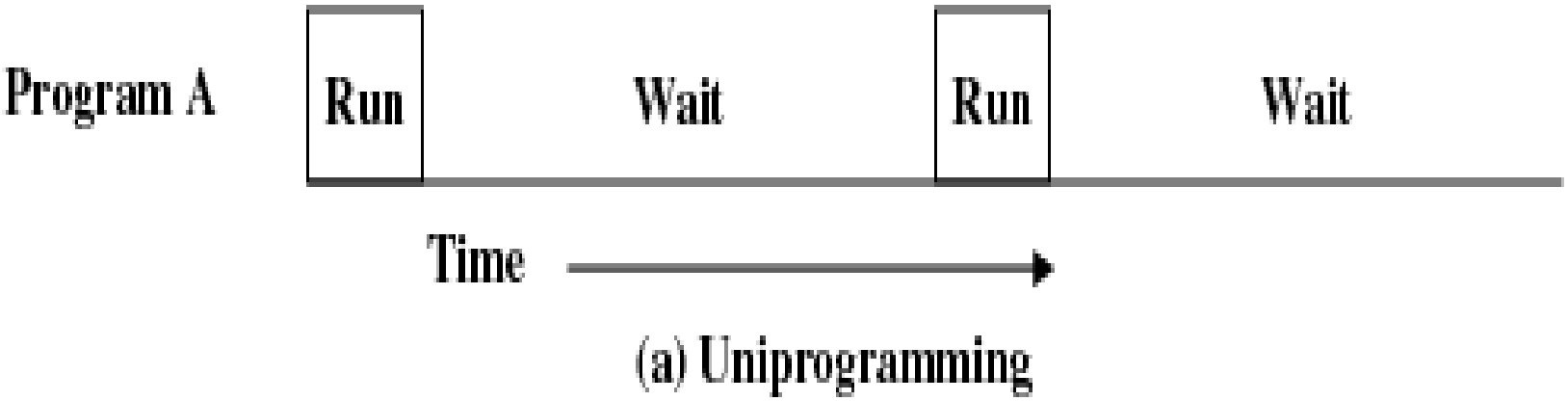
- Associated with each process is its **address space**
  - a list of memory locations from some minimum (usually 0) to some maximum
    - which the process can read and write
- The address space contains
  - the executable program
  - the program's data
  - Execution context of the program
    - Information needed by the OS to manage the process

Threads

- Multiple actions executing simultaneously
  - Heavyweight process (conventional process)
    - Owns the resources
    - Passive element
  - Lightweight process (thread)
    - Uses CPU and scheduled for execution
    - Active element
  - Multithreaded applications programs
    - Contain several threads running at one time
    - Same or different priorities
    - Examples: Web browsers and time-sharing systems

Uniprogramming

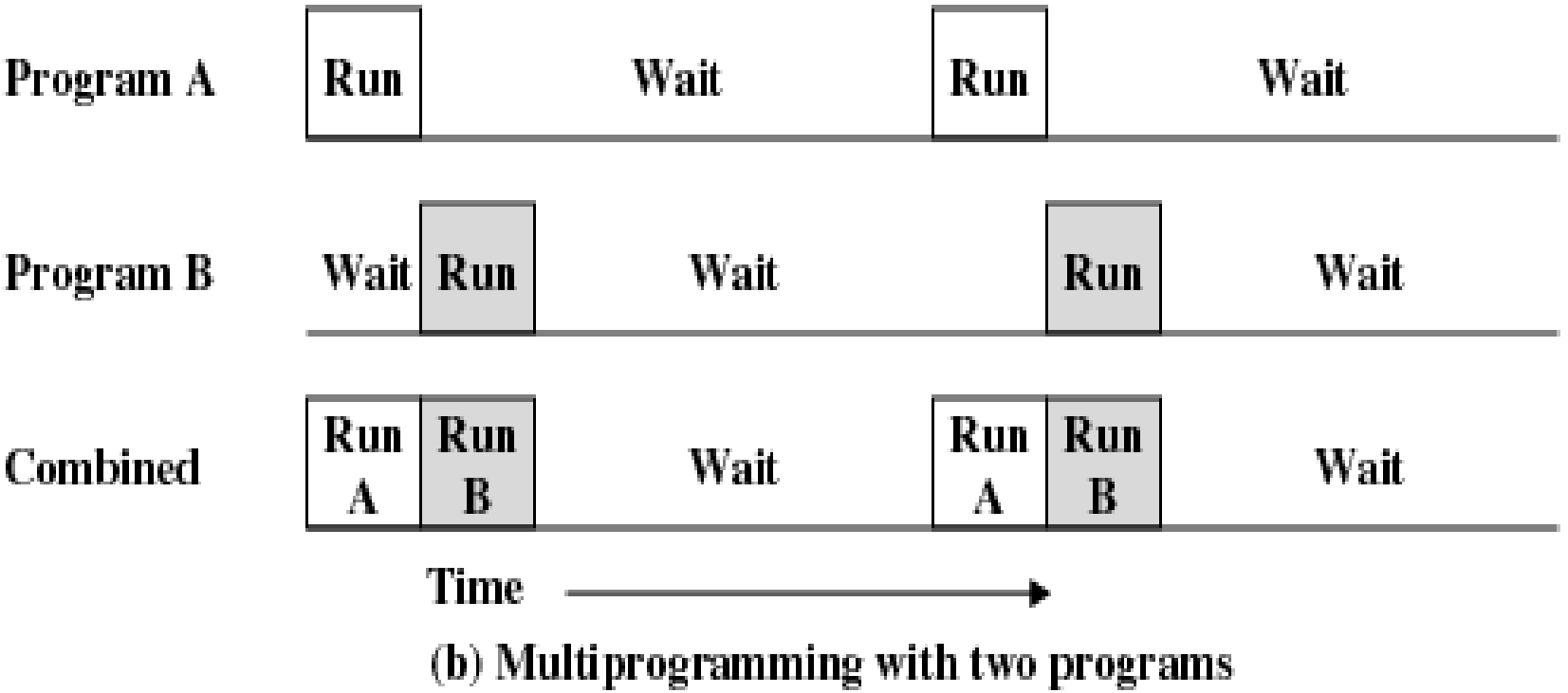
- I/O devices are slow compared to the processor
- Processor can’t switch to another process
- It must wait for I/O instruction to complete before preceding



(a) Uniprogramming

Multiprogramming

- When one job needs to wait for I/O, the processor can switch to the other job



(b) Multiprogramming with two programs

Multiprogramming… .

- **Multiprogramming** needed for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - 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 it has to wait (for I/O for example), OS switches to another job

Timesharing

- Just as multiprogramming allows the processor to handle multiple batch jobs at a time, multiprogramming can also be used to handle multiple interactive jobs.
  - In this latter case, the technique is referred to as time sharing, because processor time is shared among multiple users
- Operating system decides to stop running one process and start running another (timesharing)
  - One process has had more than its share of CPU time
- All information about the process must be explicitly saved somewhere during the suspension

Timesharing. . . .

- Process will later be restarted in exactly the same state it had when it was stopped
- In a time-sharing system, multiple users simultaneously access the system through terminals, with the OS interleaving the execution of each user program in a short burst or quantum of computation.
- Thus, if there are *n* users actively requesting service at one time, each user will only see on the average 1/*n* of the effective computer capacity, not counting OS overhead
- Given the relatively slow human reaction time, the response time on a properly designed system should be similar to that on a dedicated computer

Types of Operating Systems

- **Mainframe Operating system**: operating systems heavily oriented towards processing many jobs at once most of which need intensive input output. Typically they offer three kinds of jobs
  - Batch: routine jobs without interactive user present
    - Input relied on punched cards or tape
    - Efficiency measured in throughput
  - Transaction processing: large number of small jobs
  - Time sharing: multiple remote users run jobs at once
- **Server Operating system**: run on servers that serve multiple users over a network and allow users to share hardware and software resources.
  - Can provide print service, file service or web service

Cont'd….

- **Multi Processor Operating system:** multiple CPU in one system, a variation on server OS with some special features for communication and connectivity
- **Personal Computer Operating system**: provide a good interface for a single user
- **Real Time Operating system:** Often used as a control device in a dedicated application such as controlling, scientific experiments, medical imaging systems, industrial control systems, and some display systems

Cont'd….

- Reliability is key
- Fast and time limit sensitive
- Used in time-critical environments
  - Space flights, airport traffic control, high-speed aircraft
  - Industrial processes
  - Sophisticated medical equipment
  - Distribution of electricity
  - Telephone switching
- Must be 100% responsive, 100% of the time

- **Embedded Operating System:** run on the computers that control devices that are not generally thought of as computers such as Tv, microwave ovens and DVD

Cont'd….

- **Hybrid systems**
  - Combination of batch and interactive
  - Accept and run batch programs in the background
    - Interactive load is light
- **Interactive Systems**
  - Faster **turnaround** than batch systems
  - Slower than real-time systems
  - Introduced to provide fast turnaround when debugging programs
  - Time-sharing software developed for operating system
- **Smart Card Operating System:** handle only a single function such as electronic payments