Computer Fundamentals

Pradeep K. Sinha
Priti Sinha

Chapter 14

Operating Systems

Learning Objectives

In this chapter you will learn about:

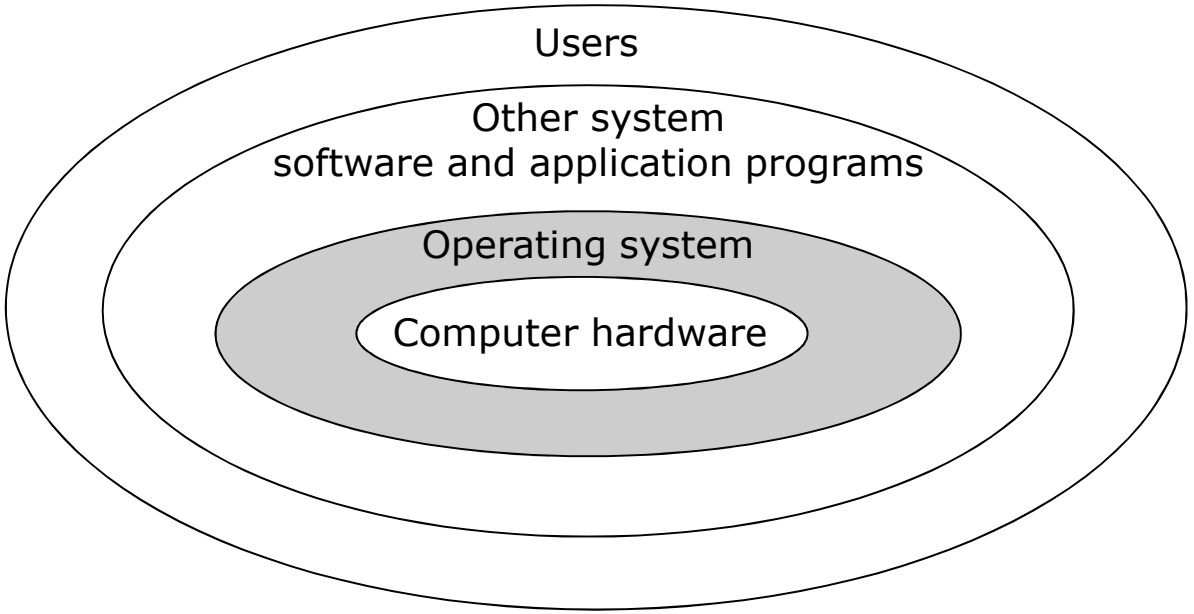
- Definition and need for operating system
- Main functions of an operating system
- Commonly used mechanisms for:
  - Process management
  - Memory management
  - File management
  - Device management
  - Security
  - Command interpretation
- Some commonly used OS capability enhancement software
- Some popular operating systems

Definition, Need and Functions of an OS

Definition and Need for OS

- Integrated set of programs that controls the resources (the CPU, memory, I/O devices, etc.) of a computer system
- Provides its users with an interface or virtual machine that is more convenient to use than the bare machine
- Two primary objectives of an OS are:
  - Making a computer system convenient to use
  - Managing the resources of a computer system

Logical Architecture of a Computer System



Operating system layer hides details of hardware from programmers and other users and provides them with a convenient interface for using the system

Main Functions of an OS

- Process management
- Memory management
- File management
- Device management
- Security
- Command interpretation

Parameters for Measuring System Performance

- **Throughput:** Amount of work that the system is able to do per unit time
- **Turnaround time:** Interval from the time of submission of a job to the system for processing to the time of completion of the job
- **Response time:** Interval from the time of submission of a job to the system for processing to the time the first response for the job is produced by the system

Process Management

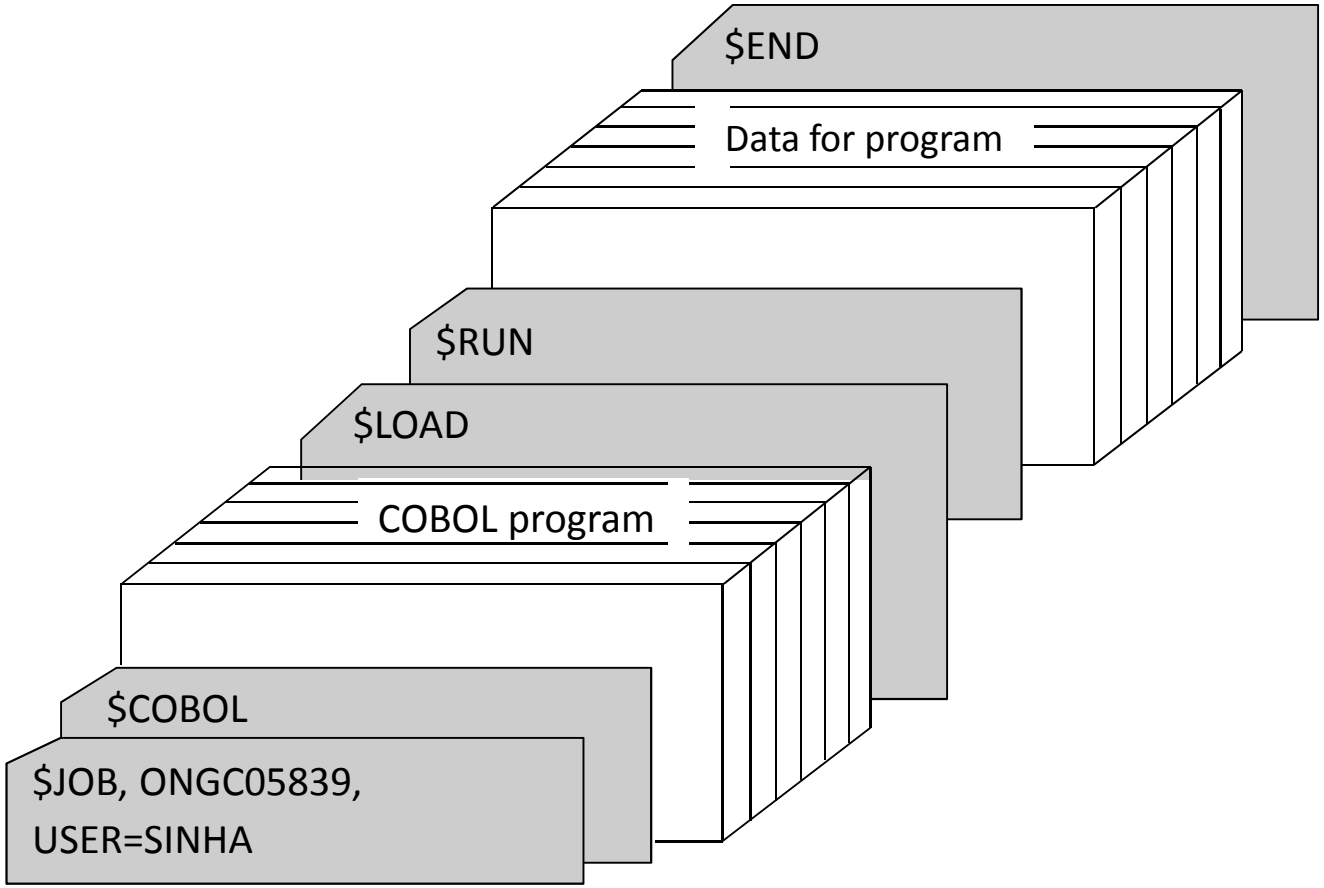
Process Management

- A **process** (also called **job**) is a program in execution
- **Process management** module of an operating system manages the processes submitted to a system in a manner to minimize *idle time* of processors (CPUs, I/O processors, etc.) of the system

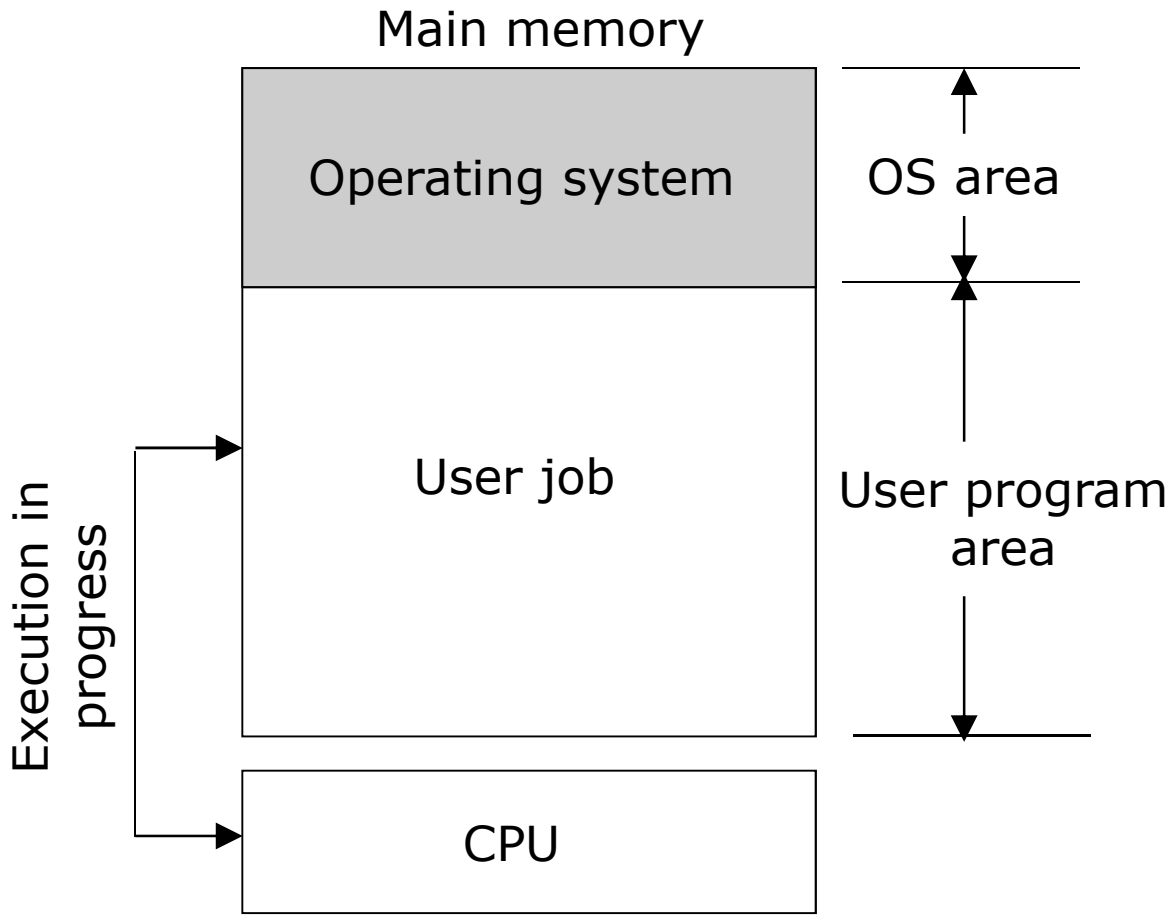
Process Management Mechanisms in Early Systems

- **Manual loading mechanism:** Jobs were manually loaded one after another in a computer by the computer operator
- **Batch processing mechanism:** Batch of jobs was submitted together to the computer and job-to-job transition was done automatically by the operating system
- **Job Control Language (JCL):** Control statements were used to identify a new job in a batch of jobs and to determine its resource requirements

Use of Job Control Statements in Batch Processing (An Example)



Uniprogramming



(Continued on next slide…)

Uniprogramming

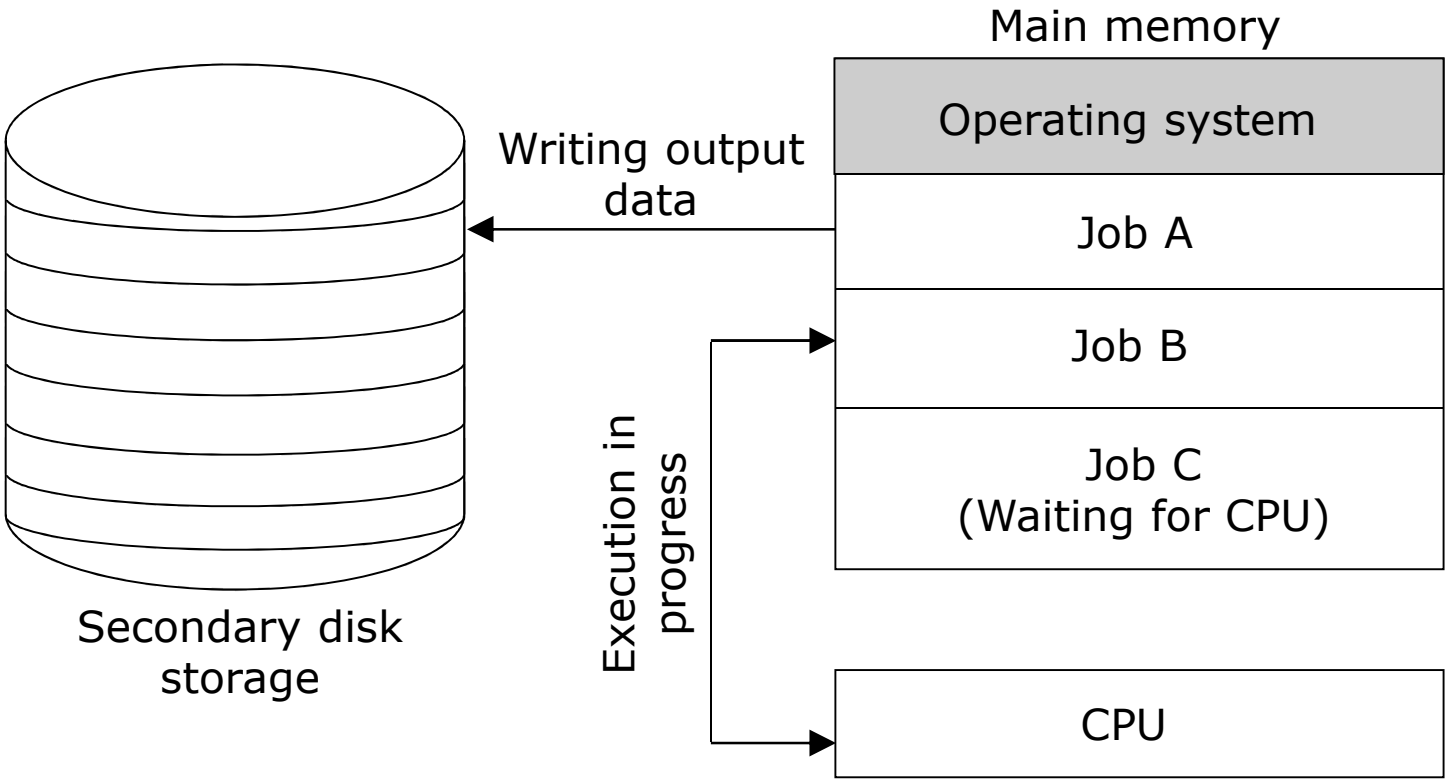
- A job does not need CPU for entire duration of its processing
- Depending on CPU utilization during the course of processing, jobs are of two types
  - *CPU-bound jobs*, which mostly perform computations with little I/O operations
  - *I/O-bound jobs*, which mostly perform I/O operations with little computation
- In a *uniprogramming system*, CPU is idle whenever the currently executing job performs I/O operations

Multiprogramming

- *Multiprogramming* is interleaved execution of two or more different and independent programs by a computer
- Multiprogramming enables two or more user programs to reside simultaneously in main memory and carries out their interleaved execution
- With multiple user programs residing simultaneously in main memory, whenever a user program that was executing goes to perform I/O operations, the operating system allocates CPU to another user program in main memory
- In multiprogramming, several user programs share CPU time to keep it busy
- Note that multiprogramming does not mean execution of instructions from several programs simultaneously

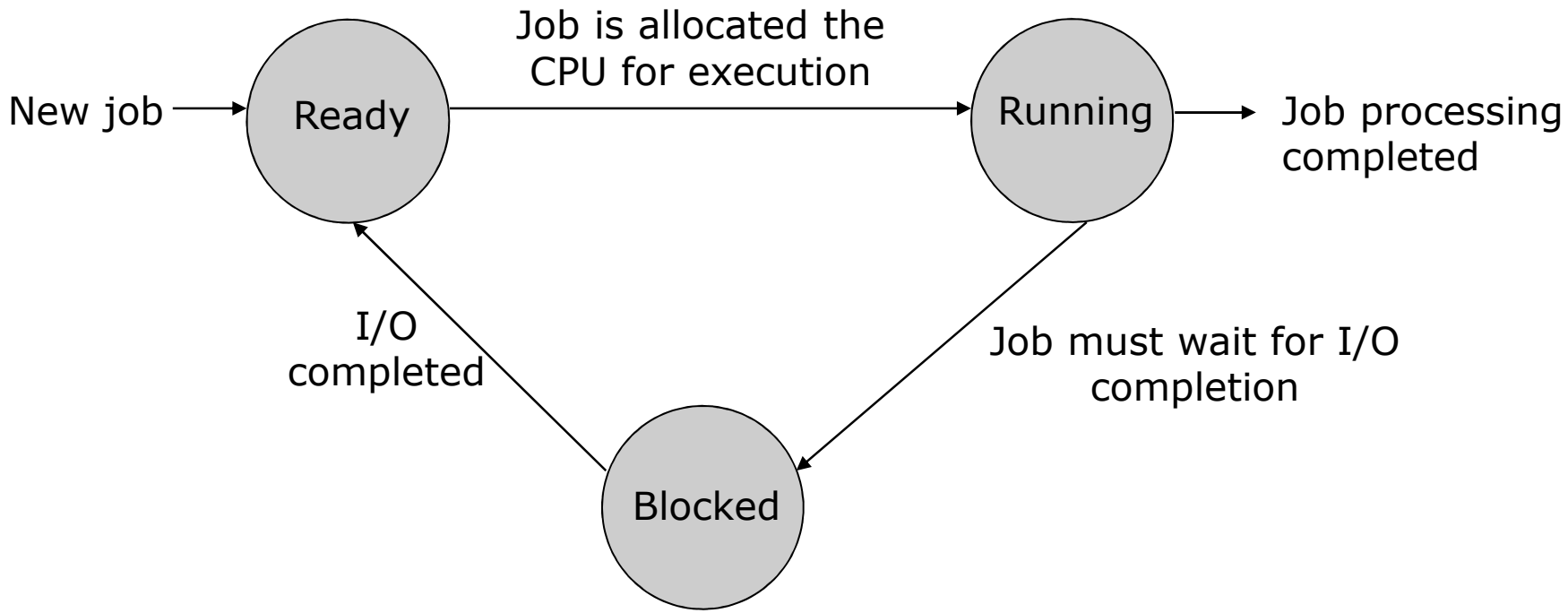
(Continued on next slide…)

Multiprogramming



A typical scenario of jobs in a multiprogramming system

Multiprogramming



Three different states of jobs in main memory in a multiprogramming system

Requirements of Multiprogramming Systems

- Large memory
- Memory protection
- Job status preservation
- Proper job mix (CPU and I/O bound jobs)
- CPU scheduling

Process Control Block (PCB)

process identifier
process state
program counter
values of various CPU registers
accounting and scheduling information
I/O status information
⋮

PCB is used to preserve the job status of each loaded process in a multiprogramming system

Multitasking

- Multitasking is single-user variation of multiprogramming concept
- Both refer to the same concept of a system's capability to work concurrently on more than one task
- Some authors prefer to use the term multiprogramming for multi-user systems and multitasking for single-user systems
- Multitasking eases user operation and saves lots of time when a user has to switch between two or more applications while performing a job
- *Multiprogramming* is interleaved execution of multiple jobs in a multi-user system, while *multitasking* is interleaved execution of multiple jobs in a single-user system

Multithreading

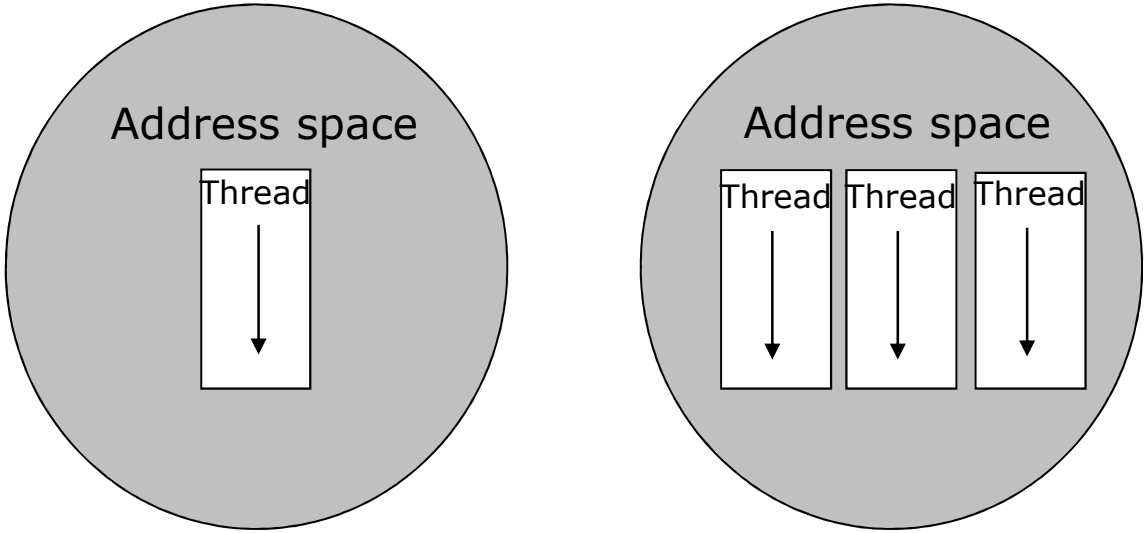
- *Threads* are a popular way to improve application performance
- In traditional operating systems, the basic unit of CPU utilization is a process
- Each process has its own program counter, its own register states, its own stack, and its own address space
- In operating systems with threads facility, the basic unit of CPU utilization is a thread
- A process consists of an address space and one or more threads of control

(Continued on next slide…)

Multithreading

- Each thread of a process has its own program counter, its own register states, and its own stack
- All the threads of a process share the same address space
- All threads of a process also share the same set of operating system resources
- Threads are often referred to as *lightweight processes* and traditional processes are referred to as *heavyweight processes*

Multithreading System



(a) Single-threaded and (b) multithreaded processes. A single-threaded process corresponds to a process of a traditional operating system. [Reproduced with permission, from the book titled Distributed Operating Systems: Concepts and Design by Pradeep K. Sinha. © 1997 IEEE, USA].

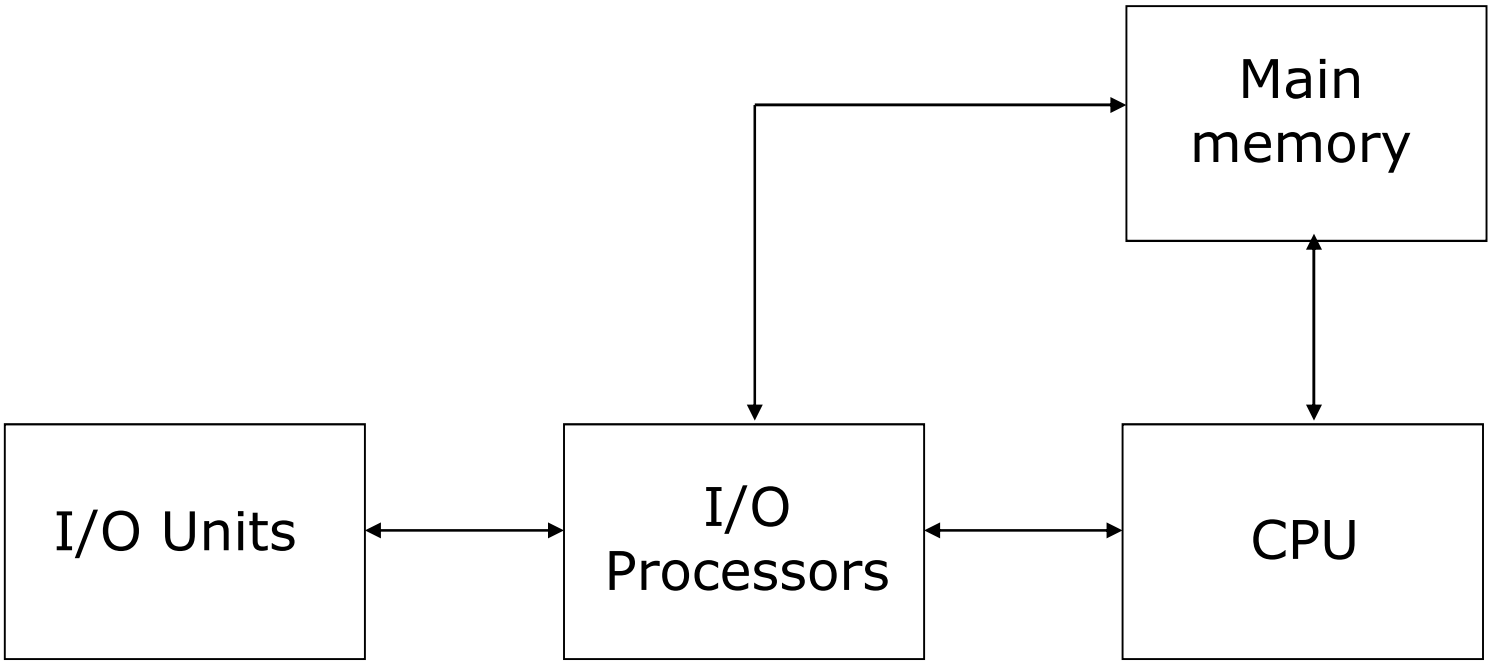
Motivations for Using Threads

- Overhead involved in creating a new process is considerably greater than that for creating a new thread within a process
- New thread uses the address space of its process
- Overhead involved in CPU switching among peer threads is very small as compared to CPU switching among processes
- Resources are shared more efficiently among multiple threads of a process than among multiple processes
- Users find the threads model more intuitive for application programming

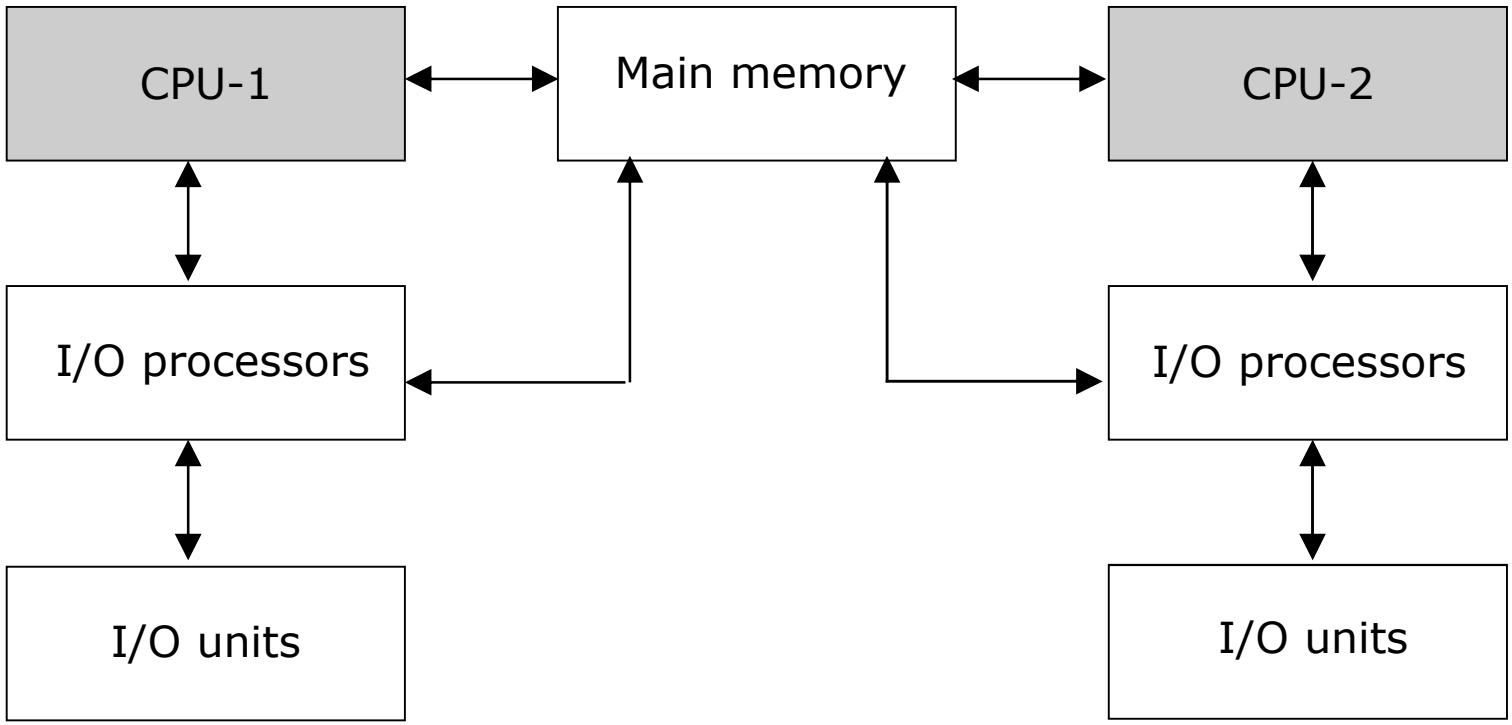
Multiprocessing

- System with two or more CPUs having ability to execute multiple processes concurrently
- Multiple CPUs are used to process either instructions from different and independent programs or different instructions from the same program simultaneously
- Types of multiprocessing:
  - *Tightly-coupled:* Single system-wide primary memory shared by all processors
  - *Loosely-coupled:* Each processor has its own local memory

CPU, Memory, and I/O Processors of a Computer System



Multiprocessing System



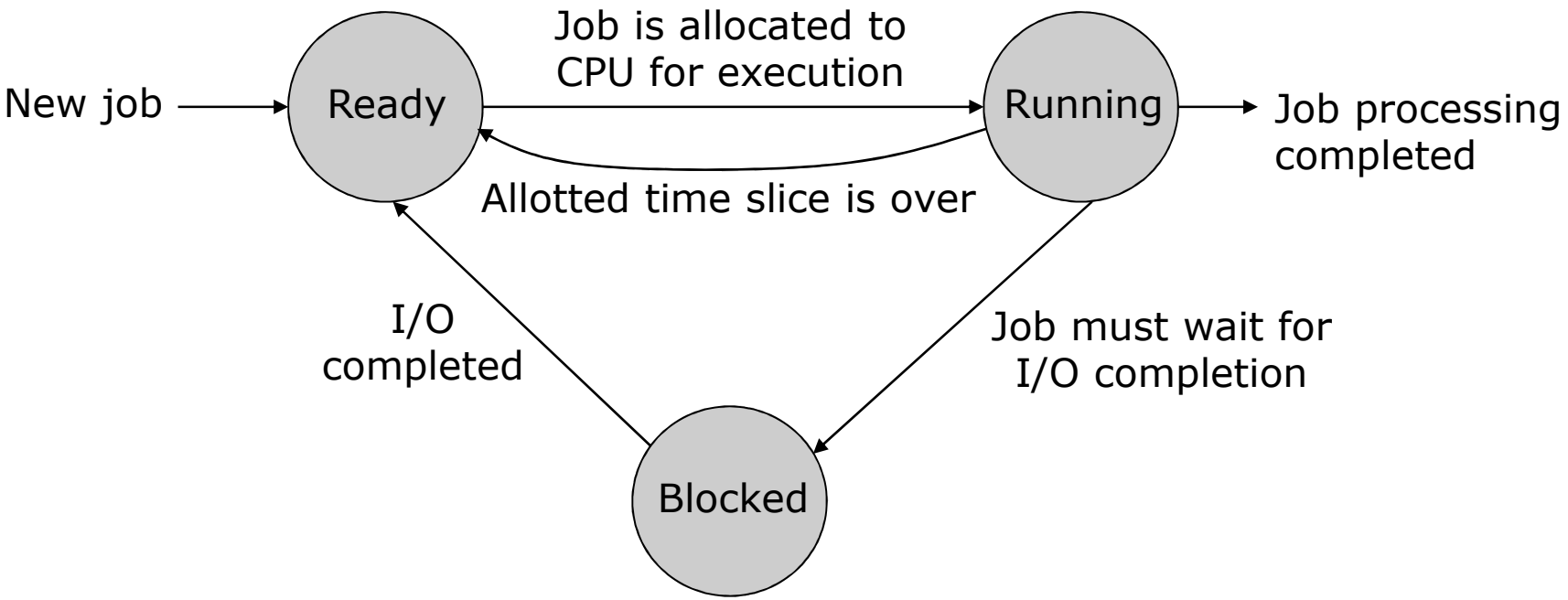
Difference between Multiprogramming and Multiprocessing

- *Multiprocessing* is simultaneous execution of two or more processes by a computer system having more than one CPU
- *Multiprogramming* is interleaved execution of two or more processes by a single-CPU system
- Multiprogramming involves execution of a portion of one program, then a portion of another, etc., in brief consecutive periods
- Multiprocessing involves simultaneous execution of several program segments of the same or different programs

Time-sharing

- Simultaneous interactive use of a computer system by many users in such a way that each one feels that he/she is the sole user of the system
- Many user terminals are connected to the same computer simultaneously
- Uses multiprogramming with a special CPU scheduling algorithm
- Short period during which a user process gets to use CPU is known as time slice, time slot, or quantum
- CPU is taken away from a running process when the allotted time slice expires

Process State Diagram for a Time-Sharing System



Process state diagram for a time-sharing system

Requirements of Time-sharing Systems

- Time-sharing systems require following additional hardware and software
  - Number of user terminals
  - Large memory to support multiprogramming
  - Memory protection mechanism to prevent a job's instructions and data from other jobs
  - Job status preservation mechanism to preserve a job's status information when the operating system takes away CPU from it, and restores this information back
  - Special CPU scheduling algorithm that allocates CPU for a short period one-by-one to each user process
  - Interrupt mechanism to send an interrupt signal to CPU after every time slice

Advantages of Time-sharing Systems

- Reduces CPU idle time
- Provides advantages of quick response time
- Offers good computing facility to small users

Memory Management

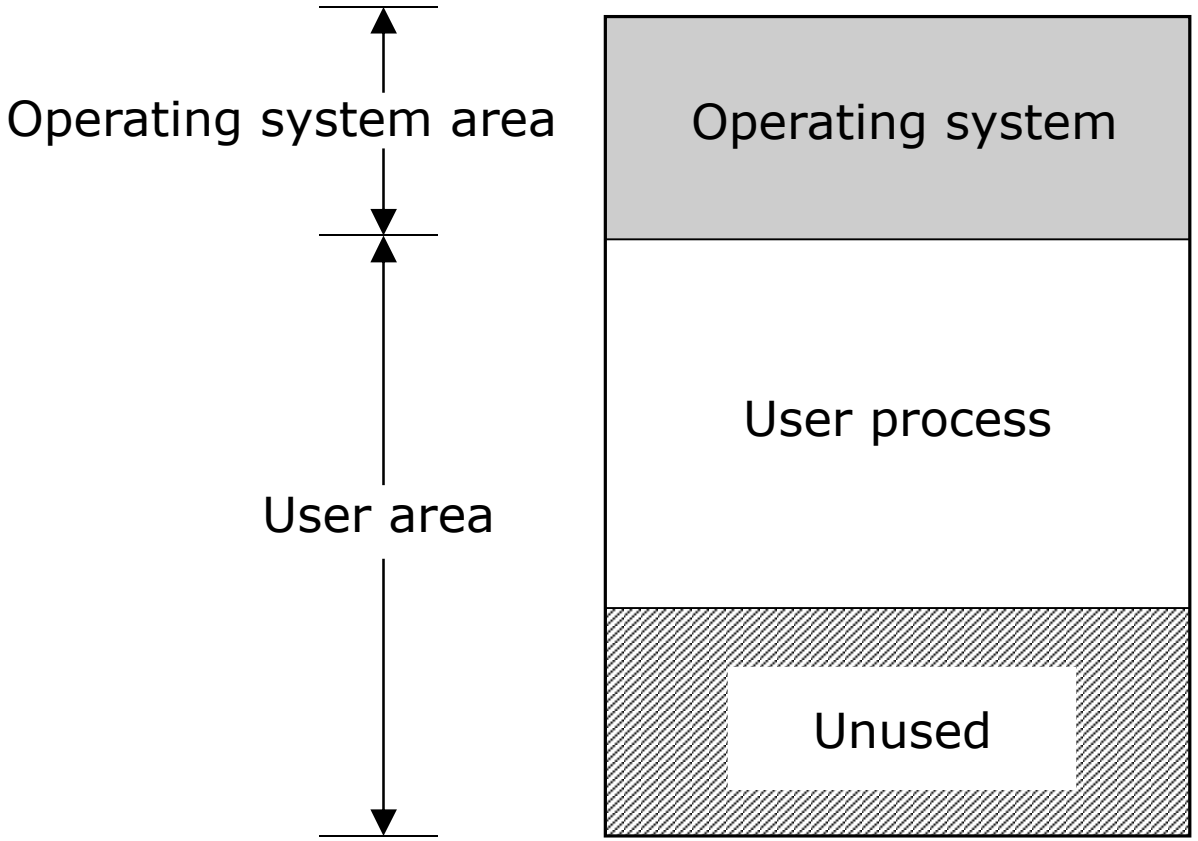
Memory Management

- Memory is important resource of a computer system that must be properly managed for the overall system performance
- Memory management module:
  - Keeps track of parts of memory in use and parts not in use
  - Allocates memory to processes as needed and deallocates when no longer needed

Uniprogramming Memory Model

- Used in systems that process one job only at a time, and all system resources are available exclusively for the job until it completes
- Simple and easy to implement
- Does not lead to proper utilization of the main memory as unoccupied memory space by the currently active user process remains unused
- Used only on very small or dedicated computer systems

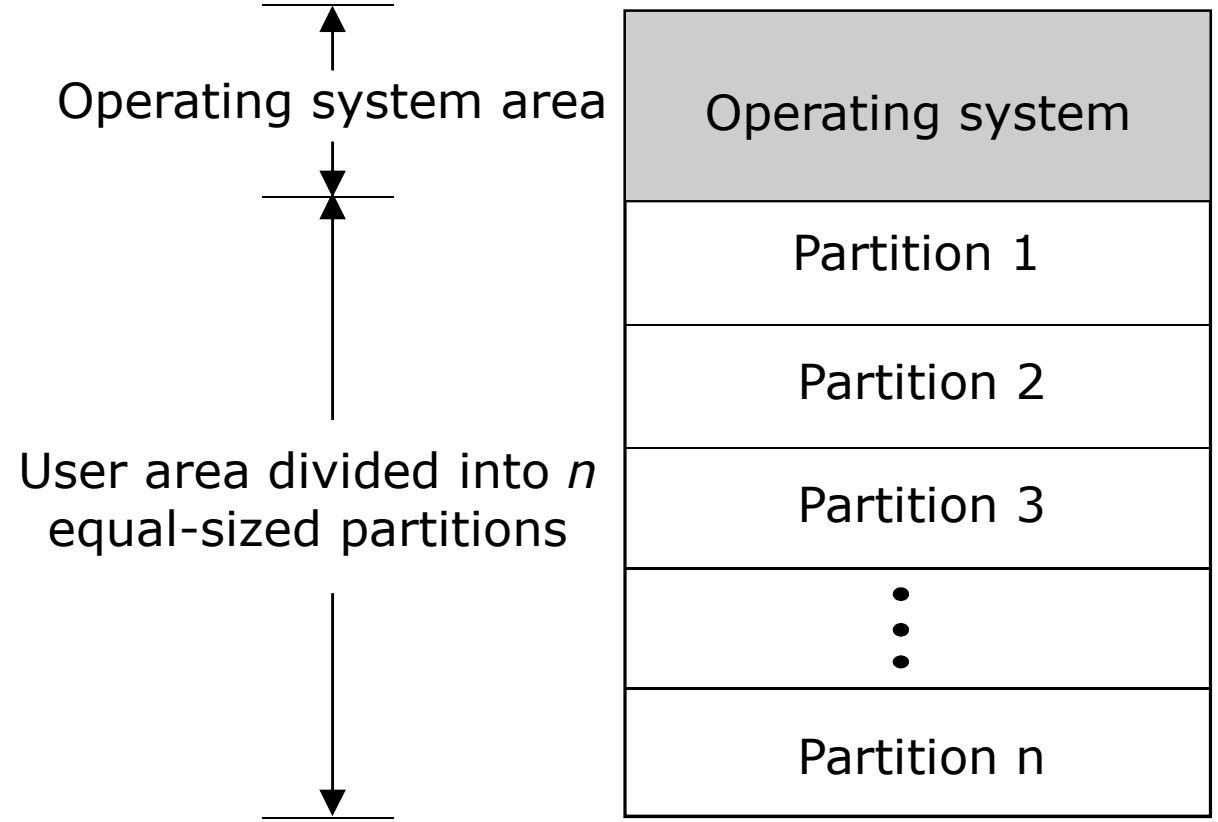
Uniprogramming Memory Model



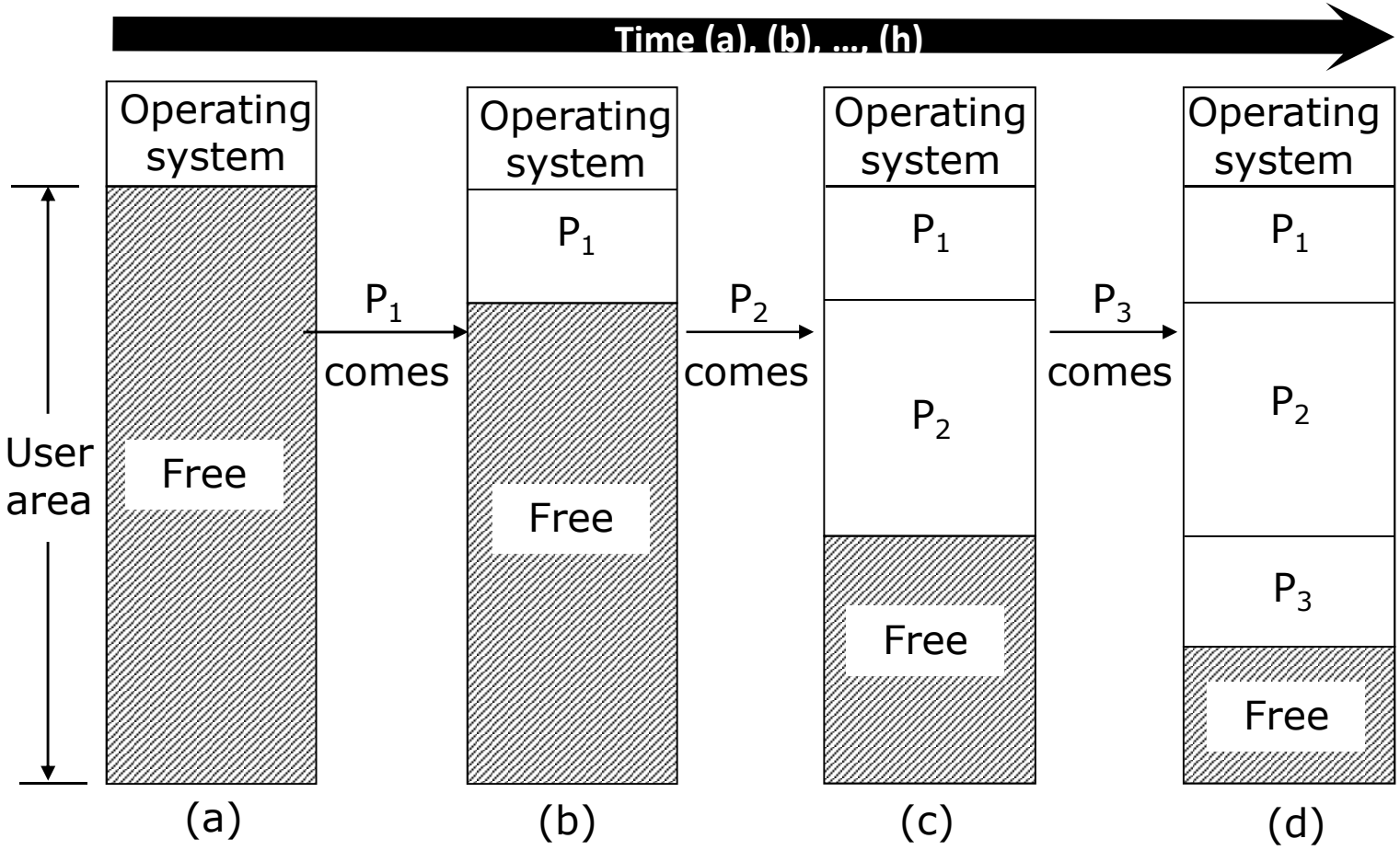
Multiprogramming Memory Models

- In a multiprogramming system, multiple user processes can reside simultaneously in main memory
- Two memory management schemes used to facilitate this are:
  - *Multiprogramming with fixed number of memory partitions*: User area of the memory is divided into a number of fixed-sized partitions
  - *Multiprogramming with variable number of memory partitions*: Number, size and location of the partitions vary dynamically as processes come and go

Multiprogramming with Fixed Number of Memory Partition

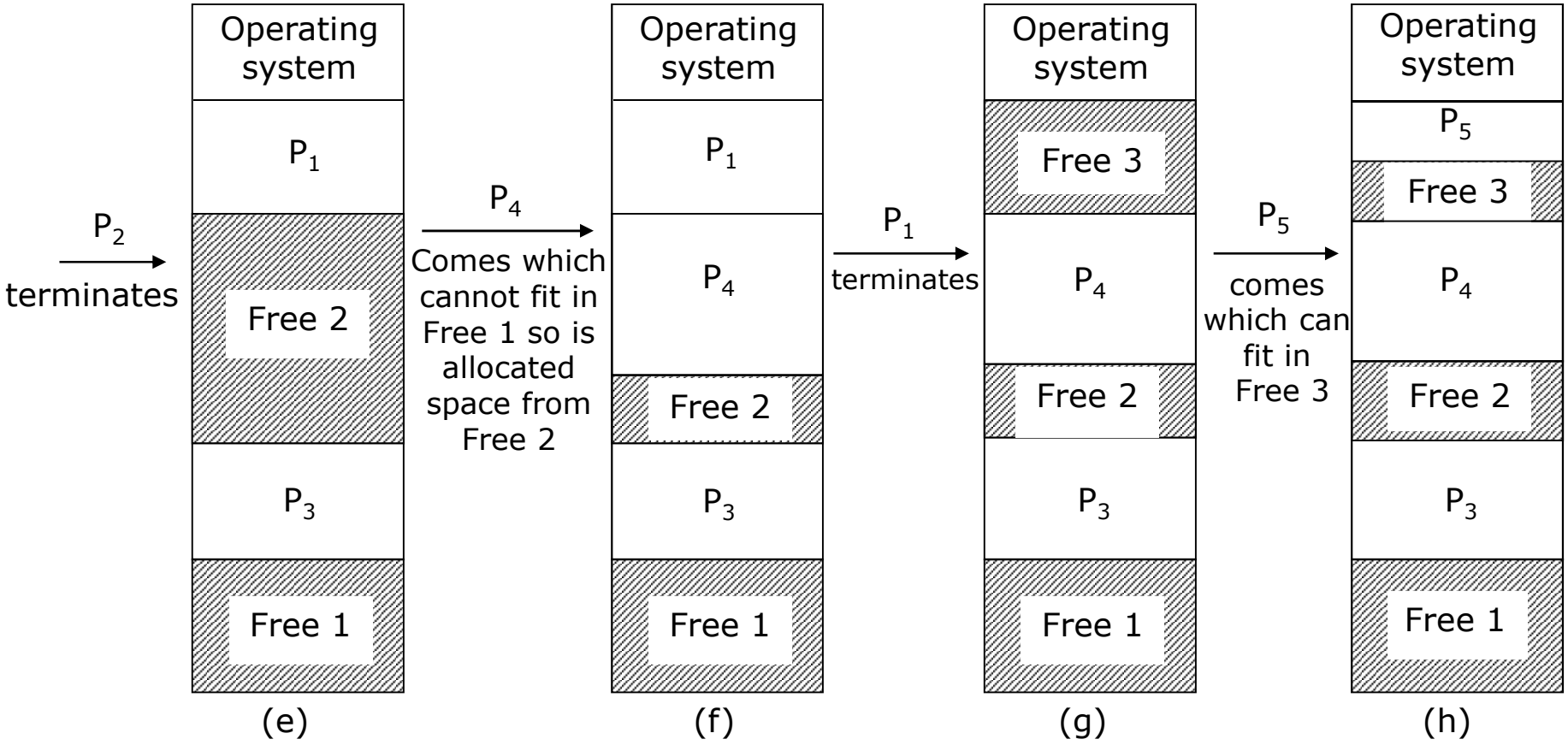


Multiprogramming with Fixed Number of Memory Partition



The number, size, and location of the partitions vary dynamically as processes come and go. (contd…)

Multiprogramming with Variable Number of Memory Partitions



The number, size, and location of the partitions vary dynamically as processes come and go.

Virtual Memory

- Conventional memory management schemes suffer from two main limitations
  - Operating system cannot load a process until sufficient free memory for loading the entire process becomes available
  - Operating system cannot load a process if main memory size is less than the total memory required
- *Virtual memory* is a memory management scheme that overcomes these limitations by allowing execution of a process without the need to load the process in main memory completely

How is Virtual Memory Realized?

- On-line secondary storage
  - On-line secondary storage device having much larger capacity than main memory
- Swapping
  - *Swapping* is the process of transferring a block of data from on-line secondary storage to main memory or vice-versa
- Demand paging
  - Instead of loading an entire process before its execution can start, the operating system uses a swapping algorithm
  - When operating system needs to swap to continue a process's execution, it invokes a page-replacement algorithm to create one for the accessed page
  - Page replacement deals with selecting a page that is residing in memory but is not in use currently

(Continued on next slide…)

How is Virtual Memory Realized?

- *Virtual memory* is often described as a hierarchy of two storage systems – one is a low-cost, large-capacity, low-speed system (on-line disk storage), and the other is a high-cost, small-capacity, high-speed system (main memory)

Advantages of Virtual Memory

- Provides a large virtual memory to programmers on a system having smaller physical memory
- Enables execution of a process on a system whose main memory size is less than the total memory required by the process
- Enables a process's execution to be started even when sufficient free memory for loading the entire process is not available
- Makes programming easier as programmers no longer need to worry about the memory size limitations
- Often leads to less I/O activity resulting in better throughput, turnaround time, and response time

Disadvantages of Virtual Memory

- Difficult to implement because it requires algorithms to support demand paging
- If used carelessly, it may substantially decrease performance due to high page fault rate

Dr. Sinha's

File Management

File Management

- A **file** is a collection of related information
- Every file has a name, its data and attributes
- File's name uniquely identifies it in the system and is used by its users to access it
- File's data is its contents
- File's attributes contain information such as date & time of its creation, date & time of last access, date & time of last update, its current size, its protection features, etc.
- File management module of an operating system takes care of file-related activities such as structuring, accessing, naming, sharing, and protection of files

File Access Methods

- Sequential access files
  - Operating systems use sequential access files for storage of files on sequential access storage media
  - A process can read the bytes or records in the file in the order in which they are stored
- Random access files
  - Operating systems use random access files for storage of files on random access storage media
  - Applications can access the contents of a random access file randomly, irrespective of the order in which the bytes or records are stored

File Operations (Examples)

File operation	Usage
Create	Is used to create a new file.
Delete	Is used to delete an existing file that is no longer needed.
Open	Is used to open an existing file when a user wants to start using it.
Close	Is used to close a file when the user has finished using it.
Read	Is used to read data stored in a file.
Write	Is used to write new data in a file.
Seek	Is used with random access files to first position read/write pointer to a specific place in file so that data can be read from, or written to, that position.
Get attributes	Is used to access the attributes of a file.
Set attributes	Is used to change user-settable attributes (such as, protection mode) of a file.
Rename	Is used to change name of an existing file.
Copy	Is used to create a copy of a file, or to copy a file to an I/O device, such as a printer.

File Naming

File naming deals with the rules for naming files in an operating system. This may include such rules as:

- Maximum number of characters that a file name may have
- Special characters allowed in a file name
- Distinction between upper case and lower case letters
- Multi-part file names allow file extensions to be part of a file name. File extensions indicate something about the file and its content
- Used by applications to check for the intended type of file before operating on it

File Extensions (Example)

File extension	Its meaning
.bas	Basic source program file
.c	C source program file
.ftn	Fortran source program file
.pas	Pascal source program file
.obj	Object file (compiler output, not yet linked)
.bin	Executable binary program file
.lib	Library of .obj files used by the linker
.dat	Data file
.hlp	Text file for HELP command
.man	Online manual page file

(Continued on next slide…)

File Extensions (Example)

File extension	Its meaning
.txt	General text file
.bak	Backup file
.doc	Microsoft word document file
.wav	Microsoft windows sound file
.wk4	Lotus 1-2-3 spreadsheet file
.xls	Microsoft Excel spreadsheet file
.jpg	JPEG graphics file
.gif	GIF graphics file

Dr. Sinha's

Device Management

Controlling I/O Devices

- Computer uses device controllers to connect I/O devices to it
- Each device controller is in charge of and controls a set of devices of a specific type
- Device controller maintains some local buffer storage and is responsible for moving data between an I/O device that it controls and its local buffer storage
- Device controller also has a few registers that it uses for communicating with CPU
- These registers are part of the regular memory address space
- This scheme is called *memory-mapped I/O*

(Continued on next slide…)

Controlling I/O Devices

- Two methods to transfer data from the controller's local buffer to the appropriate memory area of the computer are:
- **Non-DMA transfer**
  - As soon as transfer of data from input device to the controller's local buffer is complete, the controller sends an interrupt signal to CPU
  - CPU then stops what it is doing currently, and transfers control of execution to the starting address of the service routine, which handles the interrupt
  - Interrupt service routine transfers the data from local buffer of the device controller to main memory

(Continued on next slide…)

Controlling I/O Devices

- **DMA transfer**
  - When the operating system prepares for data transfer operation, it writes the relevant commands and their associated parameters into the controller's registers
  - After the controller has read the data from the device into its buffer, it copies the data one byte or word at a time from its buffer into main memory at the specified memory address
  - It does not involve the CPU
  - Device controller sends an interrupt to CPU only after it completes copying the entire data

Simple and Easy User Interface to I/O Devices

- Operating systems provide simple and easy user interface to all I/O devices
- They achieve this by organizing the software for using I/O devices as a series of layers
- Lower layers hide the internal details
- Upper layers present a nice, clean, uniform interface to the users

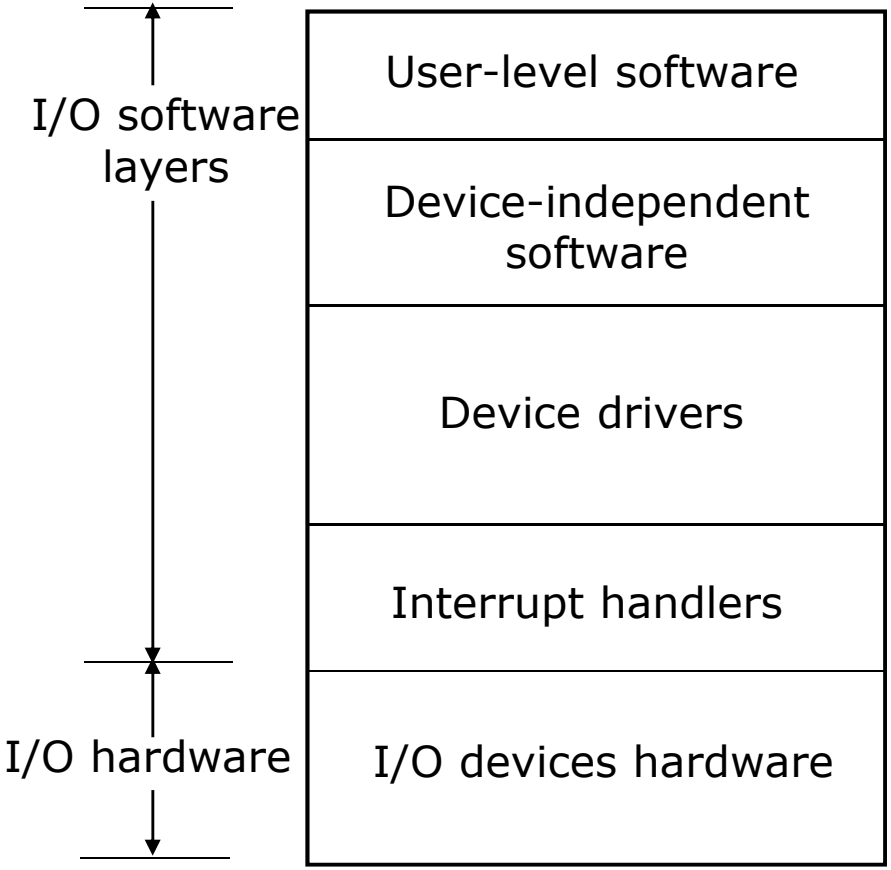
(Continued on next slide…)

Simple and Easy User Interface to I/O Devices

- Operating systems provide simple and easy user interface to all I/O devices
- They achieve this by organizing the software for using I/O devices as a series of layers
- Lower layers hide the internal details
- Upper layers present a nice, clean, uniform interface to the users

(Continued on next slide…)

Layers of I/O System



Supports standard I/O system calls as library procedures.

Performs I/O functions that are common to all devices and maps symbolic device names to the proper device driver.

Converts I/O requests into suitable commands for the appropriate device controller and writes the relevant commands and their associated parameters into the controller's registers.

Causes interrupt to wakeup the device driver after it completes data transfer.

Performs actual data I/O.

Dr. Sinha's

Security

Security

- Deals with protecting the various resources and information of a computer system against destruction and unauthorized access
- **External security:** Deals with securing computer against external factors such as fires, floods, earthquakes, stolen disks/tapes, etc. by maintaining adequate backup, using security guards, allowing access to sensitive information to only trusted employees/users, etc.
- **Internal security:** Deals with user authentication, access control, and cryptography mechanisms

Security

- **User authentication:** Deals with the problem of verifying the identity of a user (person or program) before permitting access to the requested resource
- **Access Control:** Once authenticated, access control mechanisms prohibit a user/process from accessing those resources/information that he/she/it is not authorized to access
- **Cryptography:** Means of encrypting private information so that unauthorized access cannot use information

Command Interpretation

Command Interpretation

- Provides a set of commands using which the user can give instructions to the computer for getting some job done by it
- Commands supported by the command interpretation module are known as **system calls**

(Continued on next slide…)

Command Interpretation

Two types of user interfaces supported by various operating systems are:

- **Command-line interface:** User gives instructions to the computer by typing the commands
- **Graphical User Interface (GUI):** User gives commands to the system by selecting icon or menu item displayed on the screen with the use of a point-and-draw device

Dr. Sinha's

OS Capability Enhancement Software

OS Capability Enhancement Software

- Perform several tasks of routine nature, frequently needed by users but are not provided as part of the OS
- They are primarily grouped into three categories:
  - **Translating programs:** Translate a source program into an object program
  - **Library programs:** Consist of frequently used functions and operations
  - **Utility programs:** Assist users with system maintenance tasks such as disk formatting, data compression, data backups, antivirus utilities

Dr. Sinha's

Some Popular Operating Systems

UNIX OS

- Developed in the early 1970s at Bell Laboratories by Ken Thompson and Dennis Ritchie
- Written in C high-level language, hence, highly portable
- Multi-user, time-sharing OS
- Used on a wide variety of computers ranging from notebook computers to super computers
- Especially prevalent on RISC workstations such as those from Sun Microsystems, Hewlett-Packard, IBM, and Silicon Graphics
- Structured in three layers – kernel, shell, and utilities

MS-DOS

- Stands for Microsoft Disk Operating System.
- Single-user OS for IBM and IBM-compatible personal computers (PC)
- Structured in three layers – BIOS (Basic Input Output System), kernel, and shell
- Very popular in the 1980s. Not in much use now after development of Microsoft Windows OS in 1990s

Microsoft Windows

- Developed by Microsoft to overcome limitations of MS-DOS operating system
- Single-user, multitasking OS
- Native interface is a GUI
- Designed to be not just an OS but also a complete operating environment
- OS of choice for most PCs after 1990

Microsoft Windows NT

- Multi-user, time-sharing OS developed by Microsoft
- Designed to have UNIX-like features so that it can be used for powerful workstations, network, and database servers
- Supports multiprogramming and is designed to take advantage of multiprocessing on systems having multiple processors
- Native interface is a GUI
- Built-in networking and communications features
- Provides strict system security
- Rich set of tools for software development

Linux

- Open-source OS enhanced and backed by thousands of programmers world-wide
- Multi-tasking, multiprocessing OS, originally designed to be used in PCs
- Name "Linux" is derived from its inventor Linus Torvalds
- Several Linux distributions available (Red Hat, SuSE). Difference in distribution is mostly set of tools, number and quality of applications, documentation, support, and service

Keywords/Phrases

- Access control
- Batch processing
- Command interpretation
- Command-line interface (CLI)
- CPU-bound jobs
- Cryptography
- Demand paging
- Device manageemnt
- External security
- File
- File attributes
- File extensions
- File management
- Graphical User Interface (GUI)
- I/O-bound jobs
- Internal security
- Job
- Job control language (JCL)
- Library programs
- Lightweight processes
- Linux
- Loosely coupled system
- Memory management
- Memory partition
- Microsoft Windows
- Microsoft Windows NT
- MS-DOS
- Multiprocessing
- Multiprogramming
- Multiprogramming with fixed tasks (MFT)
- Multiprogramming with variable tasks (MVT)
- Multitasking
- Multithreading
- Operating system
- Process
- Process Control Block (PCB)
- Process management
- Random access files
- Response time
- Security

(Continued on next slide…)

Keywords/Phrases

- Sequential access files
- Swapping
- Throughput
- Tightly coupled system
- Time-sharing
- Time slice
- Time slot
- Translating programs
- Turnaround time
- Uniprogramming system
- Unix
- User authentication
- Utility programs
- Virtual machine
- Virtual memory