

# Chapter 1: Introduction

---

Objectives:

- To provide a grand tour of the major operating systems components
- To provide coverage of basic computer system organization

## What Operating Systems Do

- A program that acts as an intermediary between a user of a computer and the computer hardware
- Operating System goals
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
- Use the computer hardware in an efficient manner
- Four Components of a Computer System
  - Computer hardware
    - CPU
    - Memory
    - I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - system and application programs
    - Define the ways in which the system resources are used to solve the computing problems of the users
    - word processors, compilers, web browsers, database systems, video games
  - Users
    - People, machines, other computers
- OS Definition
- resource allocator
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- control program
  - Controls execution of programs to prevent errors and improper use of the computer

## Computer-System Organization

- 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
- 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
- I/O structure
- I/O methods
  - Synchronous
  - Asynchronous
- Device-Status Table
- Direct Memory Access Structure
- Storage Structure
  - registers
  - cache
  - Main memory
  - electronic disk
  - magnetic disk
  - optical disk
  - magnetic tapes
- Why Storage is organized in hierarchy
  - Speed
  - Cost
  - Volatility
- Caching

## Computer-System Architecture

### Operating-System Structure

- Multiprogramming needed for efficiency
  - Multi-user
  - organizes jobs
  - a subset of total jobs in system is kept in memory
  - job scheduling
  - job switching
- Timesharing (multitasking) is logic extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating interactive computing.
  - response time
  - process
  - CPU scheduling
  - swapping
  - virtual memory

### Operating-System Operations

- Interrupt driven by hardware
- software error or request creates exception or trap
- other process problems including infinite loop, ...
- Dual-mode operation allows OS to protect itself and other system components
  - User mode and kernel mode
  - Mode bit provided by hardware

- bit to distinguish when system is running user code or kernel code
- designated as privileged
- system call can change mode to kernel and return from call resets it to users.

## Process Management

- A process is a program in execution. It is a unit of work within the system. **Program** is a passive entity, **process** is an active entity.
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Process termination requires reclaim of any reusable resources
- Single-threaded process has one program counter specifying location of next instruction to execute
- Multi-threaded process has one program counter per thread
- Typically system has many processes, some user, some operating system running concurrently on one or more CPUs
- The operating system is responsible for the following activities in connection with process management:
  - Creating and deleting both user and system processes
  - Suspending and resuming processes
  - Providing mechanisms for process synchronization
  - Providing mechanisms for process communication
  - Providing mechanisms for deadlock handling

## Memory Management

- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management activities
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

## Storage Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit, **file**
  - Each medium is controlled by device (i.e., disk drive, tape drive)
- File-System management

## Protection and Security

- Protection – any mechanism for controlling access of processes or users to resources defined by the OS
- Security – defense of the system against internal and external attacks
- Systems generally first distinguish among users, to determine who can do what

## Computing Environments

- Traditional computer

- Office environment
  - Home networks
- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now servers, responding to requests generated by clients
- Peer-to-Peer Computing
  - Another model of distributed system
  - P2P does not distinguish clients and servers
- Web-Based Computing

## Chapter 2: Operating-System Structure

---

Objectives:

- To describe the services an operating system provides to users, processes, and other systems
- To discuss the various ways of structuring an operating system
- To explain how operating systems are installed and customized and how they boot

### Operating System Services

One set of operating-system services provides functions that are helpful to the user:

- User interface
- Program execution
- I/O operations
- File-system manipulation
- Communications
- Error detection
- Another set of OS functions exists for ensuring the efficient operation of the system itself via resource sharing
  - Resource allocation
  - Accounting
  - Protection and security

### System Calls

- Programming interface to the services provided by the OS
- Typically written in a high-level language (C or C++)
- Mostly accessed by programs via a high-level Application Program Interface (API) rather than direct system call use

### Types of System Calls

- Process control
- File management
- Device management
- Information maintenance
- Communications

### System Programs

System programs provide a convenient environment for program development and execution. They can be divided into:

- File manipulation
- Status information
- File modification
- Programming language support
- Program loading and execution
- Communications
- Application programs

## **Layered Approach**

### **Layered Operating System**

#### **Microkernel System Structure**

- Moves as much from the kernel into “user” space
- Communication takes place between user modules using message passing
- Benefits:
  - Easier to extend a microkernel
  - Easier to port the operating system to new architectures
  - More reliable (less code is running in kernel mode)
  - More secure
- Detriments:
  - Performance overhead of user space to kernel space communication

## **Modules**

- Most modern operating systems implement kernel modules
- Overall, similar to layers but with more flexible

### **Operating System Generation**

- Operating systems are designed to run on any of a class of machines
- SYSGEN program
- Booting
- Bootstrap program

## **Chapter 3: Processes**

---

### **Process Concept**

- An operating system executes a variety of programs:
  - Batch system
  - Time-shared systems
- Process – a program in execution; process execution must progress in sequential fashion
- A process includes:
  - program counter
  - stack
  - data section
- As a process executes, it changes state

- new
- running
- waiting
- ready
- terminated

## Process Control Block (PCB)

- Information associated with each process
  - Process state
  - Program counter
  - CPU registers
  - CPU scheduling information
  - Memory-management information
  - Accounting information
  - I/O status information

## Process Scheduling Queues

- Job queue
- Ready queue
- Device queues
- Processes migrate among the various queues

## Schedulers

- Long-term scheduler (or job scheduler)
- Short-term scheduler (or CPU scheduler)
- The long-term scheduler controls the degree of multiprogramming
- Processes can be described as either:
  - I/O-bound process
  - CPU-bound process

## Context Switch

## Process Creation

## Process Termination

- Process executes last statement and asks the operating system to delete it (exit)
- Parent may terminate execution of children processes (abort)

## Cooperating Processes

- Independent process cannot affect or be affected by the execution of another process
- Cooperating process can affect or be affected by the execution of another process
- Advantages of process cooperation
  - Information sharing
  - Computation speed-up
  - Modularity
  - Convenience

## Producer-Consumer Problem

- unbounded-buffer places no practical limit on the size of the buffer
- bounded-buffer assumes that there is a fixed buffer size

## Interprocess Communication (IPC)

- Direct Communication
- Indirect Communication

## Synchronization

- Blocking is considered synchronous
- Non-blocking is considered asynchronous

## Client-Server Communication

- Sockets
- Remote Procedure Calls
- Remote Method Invocation (Java)

# Chapter 4: Threads

---

## Benefits

- Responsiveness
- Resource Sharing
- Economy
- Utilization of MP Architectures

## User Threads

## Kernel Threads

## Multithreading Models

- Many-to-One
- One-to-One
- Many-to-Many

## Two-level Model

- Similar to M:M, except that it allows a user thread to be bound to kernel thread

## Threading Issues

- Semantics of fork() and exec() system calls
- Thread cancellation
  - Terminating a thread before it has finished
  - Two general approaches:
    - Asynchronous cancellation
    - Deferred cancellation
- Signal handling
- Thread pools
- Thread specific data

# Chapter 5: CPU Scheduling

---

## CPU Scheduler

- Selects from among the processes in memory that are ready to execute, and allocates the CPU to one of them
- CPU scheduling decisions may take place when a process:
  - Switches from running to waiting state
  - Switches from running to ready state
  - Switches from waiting to ready
  - Terminates
- Scheduling under 1 and 4 is non-preemptive
- All other scheduling is preemptive

## Dispatcher

### Scheduling Criteria

- CPU utilization
- Throughput
- Turnaround Time
- Waiting Time
- Average waiting time
- Response Time

### First-Come, First-Served (FCFS) Scheduling

### Shortest-Job-First (SJF) Scheduling

- Non-Preemptive SJF
- Preemptive SJF

### Priority Scheduling

### Round Robin (RR)

- Time Quantum

### Multilevel Queue Scheduling

### Multilevel Feedback Queue

- A process can move between the various queues; aging can be implemented this way
- Multilevel-feedback-queue scheduler defined by the following parameters:
  - number of queues
  - scheduling algorithms for each queue
  - method used to determine when to upgrade a process
  - method used to determine when to demote a process
  - method used to determine which queue a process will enter when that process needs service

### Multiple-Processor Scheduling



## Real-Time Scheduling

- Hard real-time systems – required to complete a critical task within a guaranteed amount of time
- Soft real-time computing – requires that critical processes receive priority over less fortunate ones

## Algorithm Evaluation

- Deterministic modeling
- Queueing models
- Implementation

# Chapter 6: Process Synchronization

---

## Solution to Critical-Section Problem

- Mutual Exclusion
- Progress Bounded Waiting

## Peterson's Solution

## Synchronization Hardware

### TestAndSet Instruction

### Swap Instruction

## Semaphore

- Binary semaphore
- Counting semaphore

## Deadlock and Starvation

## Classical Problems of Synchronization

- Bounded-Buffer Problem
- Readers and Writers Problem
- Dining-Philosophers Problem

## Monitors

# Chapter 7: Deadlocks

---

## The Deadlock Problem

- A set of blocked processes each holding a resource and waiting to acquire a resource held by another process in the set

## Deadlock Characterization

- Mutual exclusion
- Hold and wait
- No preemption
- Circular wait

## Resource Allocation Graph

- If graph contains no cycles
  - no deadlock
- If graph contains a cycle
  - if only one instance per resource type, then deadlock
  - if several instances per resource type, possibility of deadlock

## Methods for Handling Deadlocks

- Ensure that the system will never enter a deadlock state
- Allow the system to enter a deadlock state and then recover
- Ignore the problem and pretend that deadlocks never occur in the system

## Deadlock Prevention

- Mutual Exclusion – not required for sharable resources; must hold for non-sharable resources.
- Hold and Wait – must guarantee that whenever a process requests a resource, it does not hold any other resources.
  - Require process to request and be allocated all its resources before it begins execution, or allow process to request resources only when the process has none.
  - Low resource utilization; starvation possible.
- No Preemption
- Circular Wait – impose a total ordering of all resource types, and require that each process requests resources in an increasing order of enumeration.

## Deadlock Avoidance

- Safe State
- If a system is in safe state  $\Rightarrow$  no deadlocks.
- If a system is in unsafe state  $\Rightarrow$  possibility of deadlock.
- Avoidance  $\Rightarrow$  ensure that a system will never enter an unsafe state.

## Avoidance algorithms

- Single instance of a resource type. Use a resource-allocation graph
- Multiple instances of a resource type. Use the banker's algorithm

## Deadlock Detection

- Allow system to enter deadlock state
- Detection algorithm
  - the same as Avoidance algorithms
- Recovery scheme

## Recovery from Deadlock: Process Termination

- Abort all deadlocked processes
- Abort one process at a time until the deadlock cycle is eliminated
- In which order should we choose to abort
  - Priority of the process

- How long process has computed, and how much longer to completion
- Resources the process has used
- Resources process needs to complete
- How many processes will need to be terminated
- Is process interactive or batch

## Recovery from Deadlock: Resource Preemption

- Selecting a victim – minimize cost
- Rollback – return to some safe state, restart process for that state
- Starvation – same process may always be picked as victim, include number of rollback in cost factor

## Chapter 8: Memory Management

---

### Binding of Instructions and Data to Memory

- Address binding of instructions and data to memory addresses can happen at three different stages
  - Compile time
  - Load time
  - Execution time

### Memory-Management Unit (MMU)

- Hardware device that maps virtual to physical address

### Dynamic Loading

### Dynamic Linking

### Swapping

### Contiguous Allocation

### Dynamic Storage-Allocation Problem

- How to satisfy a request of size  $n$  from a list of free holes
  - First-fit
  - Best-fit
  - Worst-fit

### Fragmentation

- External Fragmentation
- Internal Fragmentation
- Reduce external fragmentation by compaction

### Paging

- Divide physical memory into fixed-sized blocks called **frames** (size is power of 2)
- Divide logical memory into blocks of same size called **pages**

### Effective Access Time

- Associative Lookup =  $\epsilon$  time unit

- Assume memory cycle time is 1 microsecond
- Hit ratio  $\alpha$  – percentage of times that a page number is found in the associative registers; ratio related to number of associative registers
- Effective Access Time (EAT)

$$\begin{aligned} \text{EAT} &= (1 + \epsilon)\alpha + (2 + \epsilon)(1 - \alpha) \\ &= 2 + \epsilon + \alpha \end{aligned}$$

## Memory Protection

### Structure of the Page Table

- Hierarchical Paging
- Hashed Page Tables
- Inverted Page Tables

## Chapter 9: Virtual Memory

---

- Virtual memory – separation of user logical memory from physical memory.
- Virtual memory can be implemented via:
  - Demand paging
  - Demand segmentation

### Demand Paging

- Bring a page into memory only when it is needed
  - Less I/O needed
  - Less memory needed
  - Faster response
  - More users
- Lazy swapper

### Valid-Invalid Bit

### Page Fault

### Performance of Demand Paging

- Page Fault Rate  $p \leq 1.0$
- Effective Access Time (EAT)

$$\text{EAT} = (1 - p)\alpha + p\beta$$

### Copy-on-Write

- Copy-on-Write (COW) allows both parent and child processes to initially share the same pages in memory
- If either process modifies a shared page, only then is the page copied

### Page Replacement

- First-In-First-Out (FIFO) Algorithm
  - Belady's Anomaly
- Optimal Algorithm

- Least Recently Used (LRU) Algorithm
  - Counter implementation
  - Stack implementation
- LRU Approximation Algorithms
  - Reference bit
  - Second chance
- Counting Algorithms
  - LFU Algorithm
  - MFU Algorithm

## Allocation of Frames

- Fixed Allocation
- Priority Allocation
- Global vs. Local Allocation
  - Global replacement – process selects a replacement frame from the set of all frames; one process can take a frame from another
  - Local replacement – each process selects from only its own set of allocated frames

## Thrashing

- Why does thrashing occur
  - size of locality > total memory size

## Working-Set Model

- $\Delta$  - working-set window - a fixed number of page references
- $WSS_i$  (working set of Process  $P_i$ ) = total number of pages referenced in the most recent  $\Delta$  (varies in time)
- $D = \sum WSS_i$  - total demand frames
- if  $D > m \rightarrow$  Thrashing
- Policy if  $D > m$ , then suspend one of the processes

## Other Issues -- Pre-paging

## Other Issues -- Page Size

## Other Issues – Program Structure

## Other Issues – I/O interlock

# Chapter 10: File-System Interface

---

- Contiguous logical address space

## Open File Locking

## File-system Organization

- Single-Level Directory
- Two-Level Directory
- Tree-Structured Directories

- Acyclic-Graph Directories

## **File System Mounting**

## **File Sharing**

## **Protection**

# **Chapter 11: File System Implementation**

---

## **Directory Implementation**

- Linear list
- Hash Table

## **Allocation Methods**

- Contiguous allocation
- Linked allocation
- Indexed allocation

## **Free-Space Management**

- Bit vector
- Linked list
- Indexed list

## **Recovery**

# **Chapter 12: Mass-Storage Systems**

---

- Magnetic disks
  - provide bulk of secondary storage of modern computers
  - Disks can be removable
  - Drive attached to computer via I/O bus
- Magnetic tape

## **Disk Scheduling**

- FCFS
- SSTF
- SCAN
- C-SCAN
- C-LOOK

## **Disk Management**

- Low-level formatting, or physical formatting — Dividing a disk into sectors that the disk controller can read and write

# **Chapter 13: I/O Systems**

---

## **I/O Hardware**

- Incredible variety of I/O devices

- Common concepts
  - Port
  - Bus (daisy chain or shared direct access)
  - Controller (host adapter)
- I/O instructions control devices
- Devices have addresses, used by
  - Direct I/O instructions
  - Memory-mapped I/O

## **Polling**

- Determines state of device
  - command-ready
  - busy
  - error
- Busy-wait cycle to wait for I/O from device

## **Interrupts**

- CPU Interrupt-request line triggered by I/O device
- Interrupt handler receives interrupts
- Maskable to ignore or delay some interrupts
- Interrupt vector to dispatch interrupt to correct handler
  - Based on priority
  - Some non-maskable

## **Direct Memory Access**

- Used to avoid programmed I/O for large data movement
- Requires DMA controller
- Bypasses CPU to transfer data directly between I/O device and memory

## **Application I/O Interface**

### **A Kernel I/O Structure**

### **Block and Character Devices**

- Block devices include disk drives
- Character devices include keyboards, mice, serial ports

### **Network Devices**

- Varying enough from block and character to have own interface
- Approaches vary widely (pipes, FIFOs, streams, queues, mailboxes)

### **Clocks and Timers**

- Programmable interval timer used for timings, periodic interrupts

### **Blocking and Nonblocking I/O**

- Blocking - process suspended until I/O completed
  - Easy to use and understand

- Insufficient for some needs
- Nonblocking - I/O call returns as much as available
  - User interface, data copy (buffered I/O)
  - Implemented via multi-threading
  - Returns quickly with count of bytes read or written
- Asynchronous - process runs while I/O executes
  - Difficult to use
  - I/O subsystem signals process when I/O completed

## **Two I/O Methods**

- Synchronous
- Asynchronous

## **Kernel I/O Subsystem**

- Scheduling
- Buffering - store data in memory while transferring between devices
- Caching - fast memory holding copy of data
- Spooling - hold output for a device
- Device reservation - provides exclusive access to a device

## **Device-status Table**

## **Error Handling**

- OS can recover from disk read, device unavailable, transient write failures
- Most return an error number or code when I/O request fails
- System error logs hold problem reports

## **I/O Protection**

- All I/O instructions defined to be privileged
- I/O must be performed via system calls

## **Kernel Data Structures**

## **I/O Requests to Hardware Operations**

## **Life Cycle of An I/O Request**

## **Streams**

## **Intercomputer Communications**

## **Device-Functionality Progression**