Operating Systems & Systems Programming
Module 1
Introduction to Operating System

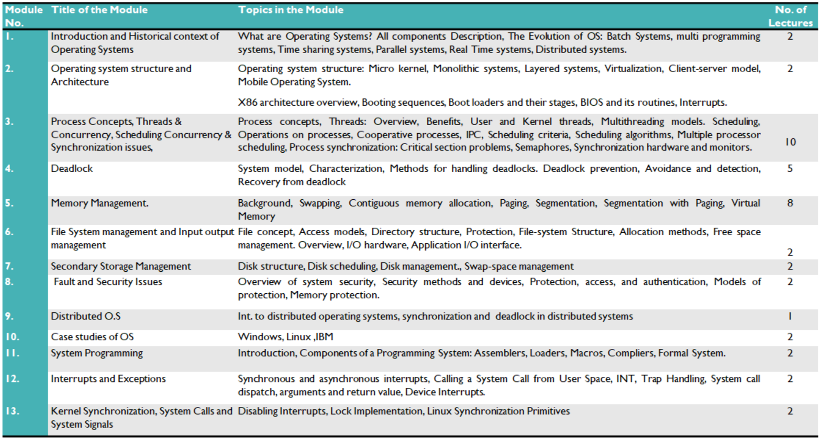
Dr. Vikash

Jaypee Institute of Information Technology, Noida

Overview

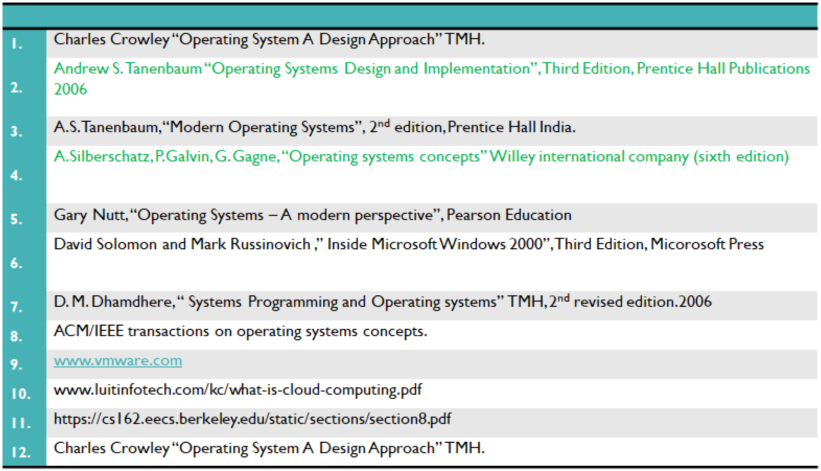
1. Course Description
2. Introduction to Operating System
3. Component Description
4. Evolution of Operating System

Course Description



Module No.	Title of the Module	Topics in the Module	No. of Lectures
1.	Introduction and Historical context of Operating Systems	What are Operating Systems? All components Description, The Evolution of OS: Batch Systems, multi programming systems, Time sharing systems, Parallel systems, Real Time systems, Distributed systems.	2
2.	Operating system structure and Architecture	Operating system structure: Micro kernel, Monolithic systems, Layered systems, Virtualization, Client-server model, Mobile Operating System. X86 architecture overview, Booting sequences, Boot loaders and their stages, BIOS and its routines, Interrupts.	2
3.	Process Concepts, Threads & Concurrency, Scheduling Concurrency & Synchronization issues,	Process concepts, Threads: Overview, Benefits, User and Kernel threads, Multithreading models. Scheduling, Operations on processes, Cooperative processes, IPC, Scheduling criteria, Scheduling algorithms, Multiple processor scheduling, Process synchronization: Critical section problems, Semaphores, Synchronization hardware and monitors.	10
4.	Deadlock	System model, Characterization, Methods for handling deadlocks. Deadlock prevention, Avoidance and detection, Recovery from deadlock	5
5.	Memory Management.	Background, Swapping, Contiguous memory allocation, Paging, Segmentation, Segmentation with Paging, Virtual Memory	8
6.	File System management and Input output management	File concept, Access models, Directory structure, Protection, File-system Structure, Allocation methods, Free space management. Overview, I/O hardware, Application I/O interface.	2
7.	Secondary Storage Management	Disk structure, Disk scheduling, Disk management., Swap-space management	2
8.	Fault and Security Issues	Overview of system security, Security methods and devices, Protection, access, and authentication, Models of protection, Memory protection.	2
9.	Distributed O.S	Int. to distributed operating systems, synchronization and deadlock in distributed systems	1
10.	Case studies of OS	Windows, Linux ,IBM	2
11.	System Programming	Introduction, Components of a Programming System: Assemblers, Loaders, Macros, Compliers, Formal System.	2
12.	Interrupts and Exceptions	Synchronous and asynchronous interrupts, Calling a System Call from User Space, INT, Trap Handling, System call dispatch, arguments and return value, Device Interrupts.	2
13.	Kernel Synchronization, System Calls and System Signals	Disabling Interrupts, Lock Implementation, Linux Synchronization Primitives	2

Reference Materials



1.	Charles Crowley "Operating System A Design Approach" TMH.
2.	Andrew S. Tanenbaum "Operating Systems Design and Implementation", Third Edition, Prentice Hall Publications 2006
3.	A.S. Tanenbaum, "Modern Operating Systems", 2nd edition, Prentice Hall India.
4.	A. Silberschatz, P. Galvin, G. Gagne, "Operating systems concepts" Willey international company (sixth edition)
5.	Gary Nutt, "Operating Systems – A modern perspective", Pearson Education
6.	David Solomon and Mark Russinovich ," Inside Microsoft Windows 2000", Third Edition, Micorosoft Press
7.	D. M. Dhamdhere, " Systems Programming and Operating systems" TMH, 2nd revised edition. 2006
8.	ACM/IEEE transactions on operating systems concepts.
9.	www.vmware.com
10.	www.luitinfotech.com/kc/what-is-cloud-computing.pdf
11.	https://cs162.eecs.berkeley.edu/static/sections/section8.pdf
12.	Charles Crowley "Operating System A Design Approach" TMH.

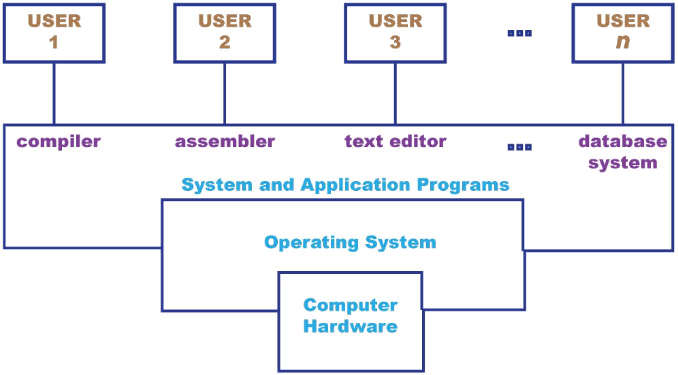
Operating System

- An OS is a program that acts an intermediary between the user of a computer and 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.

Computer System Structure

- Basically computer system can be divided into four major components:
  1. **Hardware** – Facilitates basic computer resources Like, CPU, Memory, I/O, Devices
  2. **Operating System** – A software program to provide control and coordination among various applications and end users.
  3. **Application Programs** – A user define program to solve the computing problem by utilizing the computer resources e.g., word processors, compilers, web browsers, database systems, video games etc.
  4. **Users** – The entity which actually uses the whole system e.g., people, machine, peer computers etc.

Computer System Components



Role of Operating Systems

Definition

The operating system controls the hardware and coordinates its use among the various application programs for various users.

- The operating system provides the means for proper use of the resources the the operation of the computer system.
- It like a governing body that is not directly involved in any computing function. It simply provides and environment within which other programs can do useful work.
- "The one program running at all times on the computer" is the **kernel**. Everything else is either a system program (ships with the operating system) or an application program

User Perception towards Operating Systems

The user's view of the computer varies according to the interface.

- **Single User Computers:** The computer systems that are designed for one user to monopolize its resources. The goal is to maximize the work (or play) that user is performing. The operating system is designed mostly for **ease to use** and **good performance** e.g., PC, workstations etc.
- **Multi User Computers:** These type of computer systems share its resource and may exchange information. The operating system in such case is designed to maximize resource utilization to assure that all available CPU time, memory, and I/O are used efficiently and no individual users takes more than their air share.

User Perception towards Operating Systems

- **Handheld Computers:** These type of systems are resource constrained, optimized for usablity and battery e.g., Smartphones and tablets.
- **Embedded Computers:** This type of systems are designed to run without human intervention to perform the several tasks e.g., computers in home devices and auto-mobiles etc.

System Perception towards Operating Systems

From the computer's point of view, the operating system is the program most initially involved with hardware.

- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer

Operating Systems : A Common Perceptive

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).

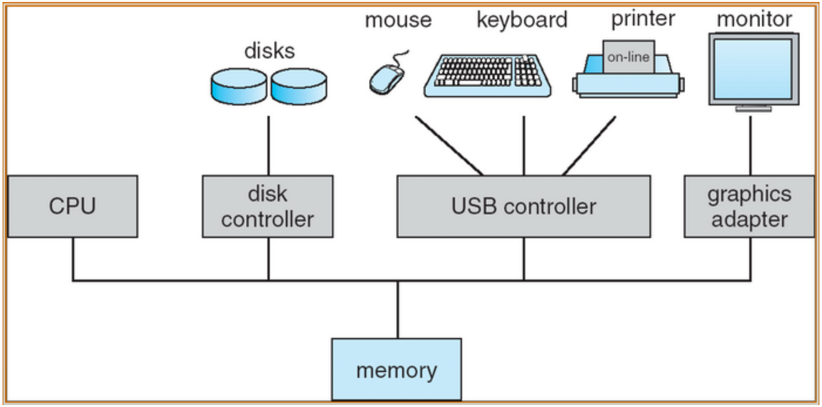
Goals of an Operating System

- 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.
- Provide isolation, security and protection among user programs.
- Improve overall system reliability
    - error confinement, fault tolerance, reconfiguration.

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

General Computer System Architecture



Computer Startup

- Bootstrap program is loaded at power-up or reboot
    - Typically stored in ROM or EPROM, generally known as firmware.
    - Initializes all aspects of system
    - Loads operating system kernel and starts execution.

Operating System Spectrum

- Monitors and Small Kernels
  - special purpose and embedded systems, real-time systems
- Batch and multiprogramming
- Timesharing
  - workstations, servers, minicomputers, timeframes
- Transaction systems
- Personal Computing Systems
- Mobile Platforms, devices (of all sizes)

First Ever Operating System – Bare Machine (1950's)

This is the era where **Hardware** is expensive and **Human** is cheaper as compare computer to hardware.

- Structure: Large machines run from console and single user system
- Early Software: Assemblers, compilers, linkers, loaders, device drivers, libraries of common subroutines.
- Secure execution
- Inefficient use of expensive resources: Low CPU utilization, high setup time.

Simple Batch Systems (1960s)

- Reduce setup time by batching jobs with similar requirements.
- Add a card reader, Hire an operator
  - User is NOT the operator
  - Automatic job sequencing
    - Forms a rudimentary OS.
  - Resident Monitor
    - Holds initial control, control transfers to job and then back to monitor.
  - Problem
    - Need to distinguish job from job and data from program.

Supervisor/Operator Control

- Secure monitor that controls job processing
    - Special cards indicate what to do.
    - User program prevented from performing I/O
- Separate user from computer
    - User submits card deck
    - cards put on tape
    - tape processed by operator
    - output written to tape
    - tape printed on printer
- Problems
    - Long turnaround time - up to 2 DAYS!!!
    - Low CPU utilization

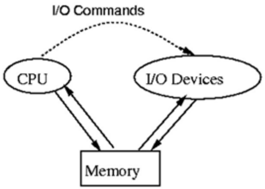
Batch Systems - Issues

Solutions to speed up I/O:

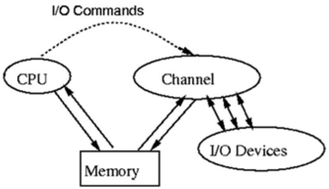
- Offline Processing
  - load jobs into memory from tapes, card reading and line printing are done offline.
- Spooling
  - Use disk (random access device) as large storage for reading as many input files as possible and
  - storing output files until output devices are ready to accept them.
  - Allows overlap - I/O of one job with computation of another.
  - Introduces notion of a job pool that allows OS choose next job to run so as to increase CPU utilization.

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

Beginning of ERA where hardware getting cheaper and Human getting his value place to live quality of life.

- 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.
- 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.

Personal Computing Systems

Hardware is as cheap as it is utilized as a resource for Human to avail computing services.

- Single user systems, portable.
- I/O devices - keyboards, mice, display screens, small printers.
- Laptops and palmtops, Smart cards, Wireless devices.
- Single user systems may not need advanced CPU utilization or protection features.
- Advantages:
  - user convenience, responsiveness, ubiquitous

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

Hardware – very cheap ; Human – very expensive

.

- 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 timeliness
- 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.

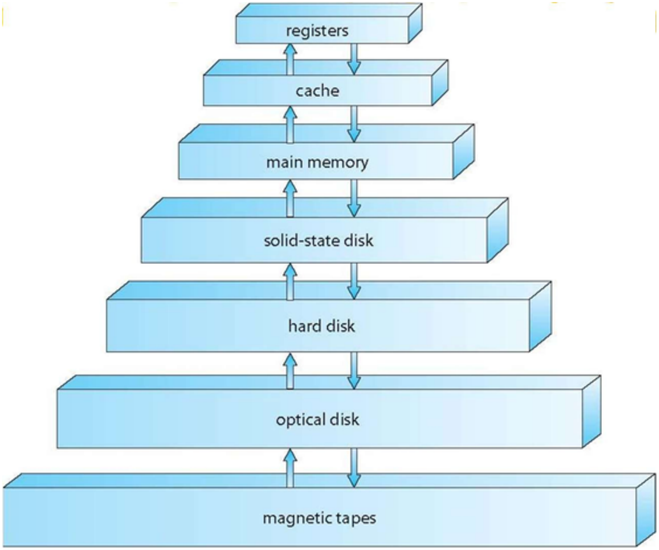
Storage Structure for Computers

- Main memory – 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.

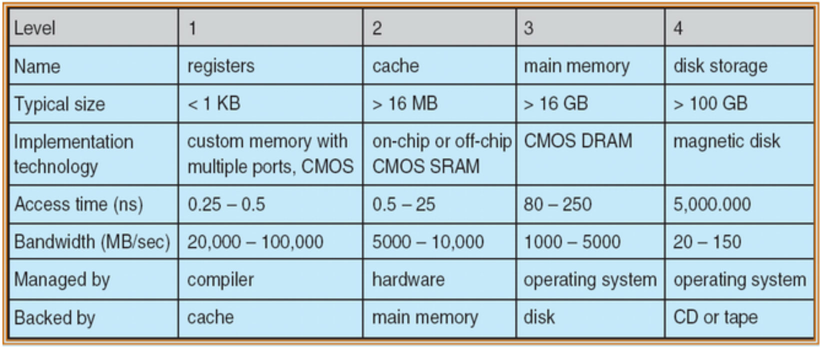
Storage Hierarchy

- Storage systems are organized in a hierarchy based on **1. Speed, 2. Cost, 3. Volatility**
- Caching - process of copying information into faster storage system; main memory can be viewed as fast cache for secondary storage.

Storage Device Hierarchy

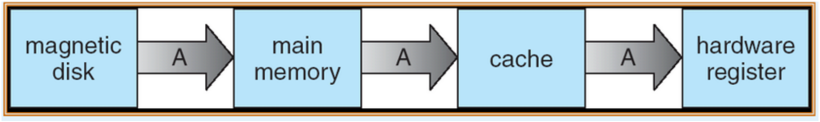


Performance of Various Levels of Storage



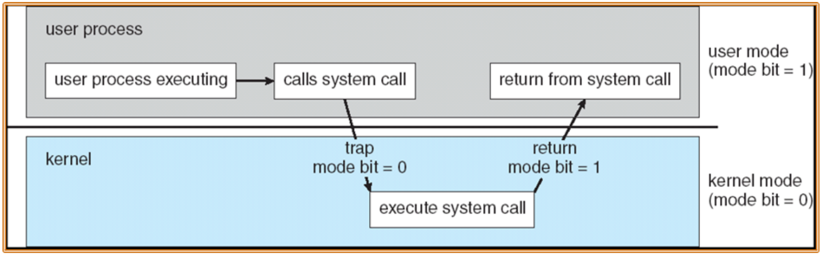
Level	1	2	3	4
Name	registers	cache	main memory	disk storage
Typical size	< 1 KB	> 16 MB	> 16 GB	> 100 GB
Implementation technology	custom memory with multiple ports, CMOS	on-chip or off-chip CMOS SRAM	CMOS DRAM	magnetic disk
Access time (ns)	0.25 – 0.5	0.5 – 25	80 – 250	5,000.000
Bandwidth (MB/sec)	20,000 – 100,000	5000 – 10,000	1000 – 5000	20 – 150
Managed by	compiler	hardware	operating system	operating system
Backed by	cache	main memory	disk	CD or tape

Migration of Integer A from Disk to Register

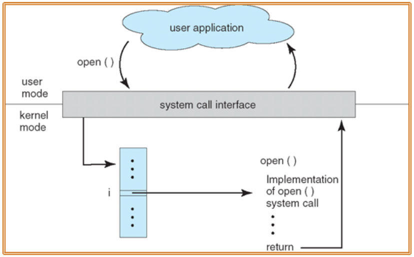


- Multitasking environments must be careful to use most recent value, not matter where it is stored in the storage hierarchy
- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
    - Several copies of a datum can exist
    - Various solutions covered in Chapter 17

System Calls (Transition from User to Kernel Mode)



System Calls



- Interface between applications and the OS.
  - Application uses an assembly instruction to trap into the kernel
  - Some higher level languages provide wrappers for system calls (e.g., C)
- System calls pass parameters between an and OS via registers or memory, memory tables or stack.
- Linux has about 300 system calls
  - read(), write(), open(), close(), fork(), exec(), ioctl(),..

Timer – Need of Clock in computer

To prevent process to be in infinite loop (process hogging resources), a timer is used, which is an hardware device.

- Timer is a counter that is decremented by the physical clock.
- Timer is set to interrupt the computer after some time period.
- Operating system sets the counter (privileged instructed).
- When counter reaches the value zero, and interrupt is generated.
- The OS sets up the value of the counter before scheduling a process to regain control or terminate program that exceeds allotted time.

OS Task: Process Management

- Process - fundamental concept in OS
    - Process is an instance of a program in execution.
    - Process needs resources - CPU time, memory, files/data and I/O devices.
- OS is responsible for the following process management activities.
    - Process creation and deletion
    - Process suspension and resumption
    - Process synchronization and interprocess communication
    - Process interactions - deadlock detection, avoidance and correction

OS Task: Memory Management

- Main Memory is an array of addressable words or bytes that is quickly accessible.
- Main Memory is volatile.
- OS is responsible for:
  - Allocate and deallocate memory to processes.
  - Managing multiple processes within memory - keep track of which parts of memory are used by which processes. Manage the sharing of memory between processes.
  - Determining which processes to load when memory becomes available.

OS Task: Secondary Storage and I/O Management

- Since primary storage (i.e., main memory) is expensive and volatile, secondary storage is required for backup.
- Disk is the primary form of secondary storage.
    - OS performs storage allocation, free-space management, etc.
- I/O system in the OS consists of
    - Device driver interface that abstracts device details
    - Drivers for specific hardware devices

OS Task: File System Management

- File is a collection of related information - represents programs and data.
- OS is responsible for
  - File creation and deletion
  - Directory creation and deletion
  - Supporting primitives for file/directory manipulation.
  - Mapping files to disks (secondary storage).

OS Task: Protection and Security

- Protection mechanisms control access of processes to user and system resources.
- Protection mechanisms must:
  - Distinguish between authorized and unauthorized use.
  - Specify access controls to be imposed on use.
  - Provide mechanisms for enforcement of access control.

OS Task: File System Management

- File is a collection of related information - represents programs and data.
- OS is responsible for
  - File creation and deletion
  - Directory creation and deletion
  - Supporting primitives for file/directory manipulation.
  - Mapping files to disks (secondary storage).

Thank You!!!