Lecture #1

What is an Operating System?

⇨ A program that acts as an intermediary between a user of a computer and the computer hardware.

⇨ An operating System is a collection of system programs that together control the operations of a computer system.

Some examples of operating systems are UNIX, Mach, MS-DOS, MS-Windows, Windows/NT, Chicago, OS/2, MacOS, VMS, MVS, and VM.

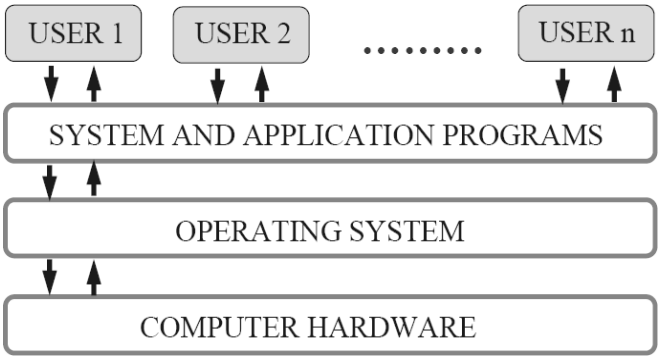
Operating system goals:

- 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 Components

1. **Hardware** – provides basic computing resources (CPU, memory, I/O devices).
2. **Operating system** – controls and coordinates the use of the hardware among the various application programs for the various users.
3. **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 programs).
4. **Users** (people, machines, other computers).

Abstract View of System Components



Operating System Definitions

Resource allocator – manages and allocates resources.

Control program – controls the execution of user programs and operations of I/O devices .

Kernel – The one program running at all times (all else being application programs).

Components of OS: OS has two parts. (1)Kernel.(2)Shell.

(1) Kernel is an active part of an OS i.e., it is the part of OS running at all times. It is a programs which can interact with the hardware. Ex: Device driver, dll files, system files etc.

(2) Shell is called as the command interpreter. It is a set of programs used to interact with the application programs. It is responsible for execution of instructions given to OS (called commands).

Operating systems can be explored from two viewpoints: the user and the system.

User View: From the user's point view, the OS is designed for one user to monopolize its resources, to maximize the work that the user is performing and for ease of use.

System View: From the computer's point of view, an operating system is a control program that manages the execution of user programs to prevent errors and improper use of the computer. It is concerned with the operation and control of I/O devices.

Lecture #2

Functions of Operating System:

Process Management

A *process* is a program in execution. A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task.

The operating system is responsible for the following activities in connection with process management.

- ✦ Process creation and deletion.
- ✦ process suspension and resumption.
- ✦ Provision of mechanisms for:
  - process synchronization
  - process communication

Main-Memory Management

Memory is a large array of words or bytes, each with its own address. It is a repository of quickly accessible data shared by the CPU and I/O devices.

Main memory is a volatile storage device. It loses its contents in the case of system failure.

The operating system is responsible for the following activities in connections with memory management:

- ♦ Keep track of which parts of memory are currently being used and by whom.
- ♦ Decide which processes to load when memory space becomes available.
- ♦ Allocate and de-allocate memory space as needed.

File Management

A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data.

The operating system is responsible for the following activities in connections with file management:

- ✦ File creation and deletion.
- ✦ Directory creation and deletion.
- ✦ Support of primitives for manipulating files and directories.
- ✦ Mapping files onto secondary storage.
- ✦ File backup on stable (nonvolatile) storage media.

I/O System Management

The I/O system consists of:

- ✦ A buffer-caching system
- ✦ A general device-driver interface
- ✦ Drivers for specific hardware devices

Secondary-Storage Management

Since main memory (*primary storage*) is volatile and too small to accommodate all data and programs permanently, the computer system must provide *secondary storage* to back up main memory.

Most modern computer systems use disks as the principle on-line storage medium, for both programs and data.

The operating system is responsible for the following activities in connection with disk management:

- ✦ Free space management
- ✦ Storage allocation
- ✦ Disk scheduling

Networking (Distributed Systems)

- ♦ A *distributed* system is a collection processors that do not share memory or a clock. Each processor has its own local memory.
- ♦ The processors in the system are connected through a communication network.
- ♦ Communication takes place using a *protocol.*
- ♦ A distributed system provides user access to various system resources.
- ♦ Access to a shared resource allows:
- ✦ Computation speed-up

- ✦ Increased data availability
- ✦ Enhanced reliability

Protection System

- ♦ *Protection* refers to a mechanism for controlling access by programs, processes, or users to both system and user resources.
- ♦ The protection mechanism must:

✦ distinguish between authorized and unauthorized usage.
✦ specify the controls to be imposed.
✦ provide a means of enforcement.

Command-Interpreter System

- Many commands are given to the operating system by control statements which deal with:

✦ process creation and management
✦ I/O handling
✦ secondary-storage management
✦ main-memory management
✦ file-system access
✦ protection
✦ networking

- The program that reads and interprets control statements is called variously:

✦ command-line interpreter
✦ shell (in UNIX)

- Its function is to get and execute the next command statement.

Operating-System Structures

- System Components
- Operating System Services
- System Calls
- System Programs
- System Structure
- Virtual Machines
- System Design and Implementation
- System Generation

Common System Components

- Process Management
- Main Memory Management
- File Management
- I/O System Management
- Secondary Management
- Networking
- Protection System
- Command-Interpreter System

Lecture #3

Evolution of OS:

1. Mainframe Systems

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 pack to monitor

2. Batch Processing Operating System:

- This type of OS accepts more than one jobs and these jobs are batched/ grouped together according to their similar requirements. This is done by computer operator. Whenever the computer becomes available, the batched jobs are sent for execution and gradually the output is sent back to the user.
- It allowed only one program at a time.
- This OS is responsible for scheduling the jobs according to priority and the resource required.

3. Multiprogramming Operating System:

- This type of OS is used to execute more than one jobs simultaneously by a single processor. it increases CPU utilization by organizing jobs so that the CPU always has one job to execute.
- The concept of multiprogramming is described as follows:
  - All the jobs that enter the system are stored in the job pool(in disc). The operating system loads a set of jobs from job pool into main memory and begins to execute.
  - During execution, the job may have to wait for some task, such as an I/O operation, to complete. In a multiprogramming system, the operating system simply switches to another job and executes. When that job needs to wait, the CPU is switched to *another* job, and so on.
  - When the first job finishes waiting and it gets the CPU back.
  - As long as at least one job needs to execute, the CPU is never idle.

  Multiprogramming operating systems use the mechanism of job scheduling and CPU scheduling.

3. Time-Sharing/multitasking Operating Systems

Time sharing (or multitasking) OS is a logical extension of multiprogramming. It provides extra facilities such as:

- Faster switching between multiple jobs to make processing faster.
- Allows multiple users to share computer system simultaneously.
- The users can interact with each job while it is running.

These systems use a concept of virtual memory for effective utilization of memory space. Hence, in this OS, no jobs are discarded. Each one is executed using virtual memory concept. It uses CPU scheduling, memory management, disc management and security management. Examples: CTSS, MULTICS, CAL, UNIX etc.

4. Multiprocessor Operating Systems

Multiprocessor operating systems are also known as parallel OS or tightly coupled OS. Such operating systems have more than one processor in close communication that sharing the computer bus, the clock and sometimes memory and peripheral devices. It executes multiple jobs at same time and makes the processing faster.

Multiprocessor systems have three main advantages:

- **Increased throughput:** By increasing the number of processors, the system performs more work in less time. The speed-up ratio with **N** processors is less than **N**.
- **Economy of scale:** Multiprocessor systems can save more money than multiple single-processor systems, because they can share peripherals, mass storage, and power supplies.
- **Increased reliability:** If one processor fails to done its task, then each of the remaining processors must pick up a share of the work of the failed processor. The failure of one processor will not halt the system, only slow it down.

The ability to continue providing service proportional to the level of surviving hardware is called **graceful degradation.** Systems designed for graceful degradation are called **fault tolerant.**

The multiprocessor operating systems are classified into two categories:

1. Symmetric multiprocessing system
2. Asymmetric multiprocessing system

- In symmetric multiprocessing system, each processor runs an identical copy of the operating system, and these copies communicate with one another as needed.
- In asymmetric multiprocessing system, a processor is called master processor that controls other processors called slave processor. Thus, it establishes master-slave relationship. The master processor schedules the jobs and manages the memory for entire system.

5. Distributed Operating Systems

- In distributed system, the different machines are connected in a network and each machine has its own processor and own local memory.
- In this system, the operating systems on all the machines work together to manage the collective network resource.
- It can be classified into two categories:
    1. Client-Server systems
    2. Peer-to-Peer systems

Advantages of distributed systems.

- Resources Sharing
- Computation speed up – load sharing
- Reliability
- Communications
- Requires networking infrastructure.
- Local area networks (LAN) or Wide area networks (WAN)

.

6. Desktop Systems/Personal Computer Systems

- The PC operating system is designed for maximizing user convenience and responsiveness. This system is neither multi-user nor multitasking.
- These systems include PCs running Microsoft Windows and the Apple Macintosh. The MS-DOS operating system from Microsoft has been superseded by multiple flavors of Microsoft Windows and IBM has upgraded MS-DOS to the OS/2 multitasking system.
- The Apple Macintosh operating system has been ported to more advanced hardware, and now includes new features such as virtual memory and multitasking.

7. Real-Time Operating Systems (RTOS)

- A real-time operating system (RTOS) is a multitasking operating system intended for applications with fixed deadlines (real-time computing). Such applications include some small embedded systems, automobile engine controllers, industrial robots, spacecraft, industrial control, and some large-scale computing systems.
- The real time operating system can be classified into two categories:
  1. hard real time system and 2. soft real time system.
- A **hard real-time** system guarantees that critical tasks be completed on time. This goal requires that all delays in the system be bounded, from the retrieval of stored data to the time that it takes the operating system to finish any request made of it. Such time constraints dictate the facilities that are available in hard real-time systems.
- A **soft real-time** system is a less restrictive type of real-time system. Here, a critical real-time task gets priority over other tasks and retains that priority until it completes. Soft real time system can be mixed with other types of systems. Due to less restriction, they are risky to use for industrial control and robotics.

Lecture #4

Operating System Services

Following are the five services provided by operating systems to the convenience of the users.

1. Program Execution

The purpose of computer systems is to allow the user to execute programs. So the operating system provides an environment where the user can conveniently run programs. Running a program involves the allocating and deallocating memory, CPU scheduling in case of multiprocessing.

2. I/O Operations

Each program requires an input and produces output. This involves the use of I/O. So the operating systems are providing I/O makes it convenient for the users to run programs.

3. File System Manipulation

The output of a program may need to be written into new files or input taken from some files. The operating system provides this service.

4. Communications

The processes need to communicate with each other to exchange information during execution. It may be between processes running on the same computer or running on the different computers. Communications can be occur in two ways: (i) shared memory or (ii) message passing

5. Error Detection

An error is one part of the system may cause malfunctioning of the complete system. To avoid such a situation operating system constantly monitors the system for detecting the errors. This relieves the user of the worry of errors propagating to various part of the system and causing malfunctioning.

Following are the three services provided by operating systems for ensuring the efficient operation of the system itself.

1. Resource allocation

When multiple users are logged on the system or multiple jobs are running at the same time, resources must be allocated to each of them. Many different types of resources are managed by the operating system.

2. Accounting

The operating systems keep track of which users use how many and which kinds of computer resources. This record keeping may be used for accounting (so that users can be billed) or simply for accumulating usage statistics.

3. Protection

When several disjointed processes execute concurrently, it should not be possible for one process to interfere with the others, or with the operating system itself. Protection involves ensuring that all access to system resources is controlled. Security of the system from outsiders is also important. Such security starts with each user having to authenticate him to the system, usually by means of a password, to be allowed access to the resources.

System Call:

- System calls provide an interface between the process and the operating system.
- System calls allow user-level processes to request some services from the operating system which process itself is not allowed to do.
- For example, for I/O a process involves a system call telling the operating system to read or write particular area and this request is satisfied by the operating system.

The following different types of system calls provided by an operating system:

Process control

- end, abort
- load, execute
- create process, terminate process
- get process attributes, set process attributes
- wait for time
- wait event, signal event
- allocate and free memory

File management

- create file, delete file
- open, close
- read, write, reposition
- get file attributes, set file attributes

Device management

- request device, release device
- read, write, reposition
- get device attributes, set device attributes
- logically attach or detach devices

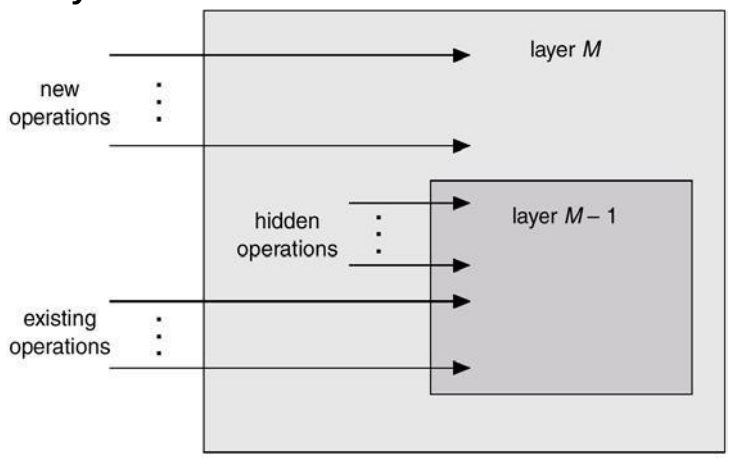
Information maintenance

- get time or date, set time or date
- get system data, set system data
- get process, file, or device attributes
- set process, file, or device attributes

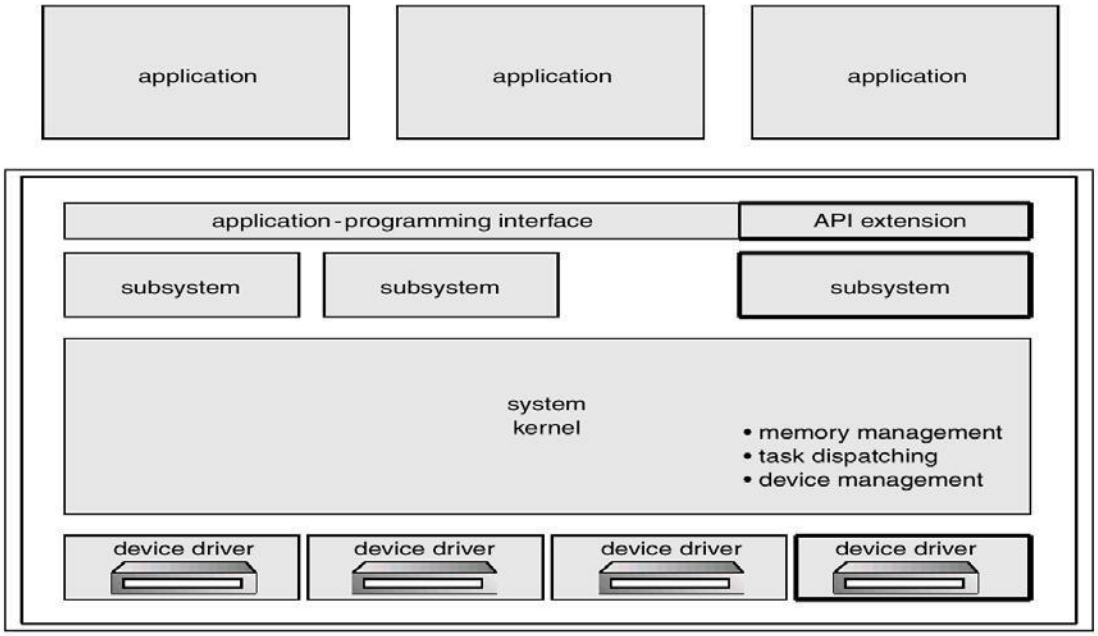
Communications

- create, delete communication connection
- send, receive messages
- transfer status information
- attach or detach remote devices

An Operating System Layer



OS/2 Layer Structure



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

Virtual Machines

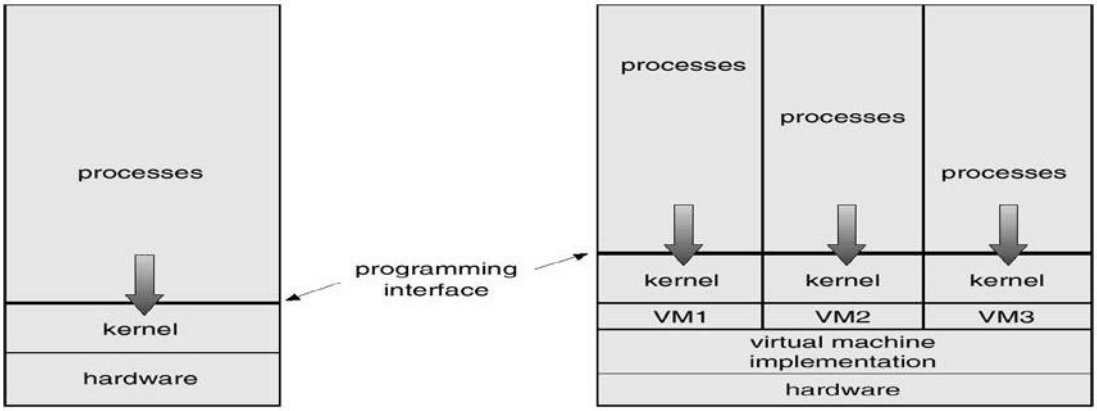
- A *virtual machine* takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware.
- A virtual machine provides an interface *identical* to the underlying bare hardware.
- The operating system creates the illusion of multiple processes, each executing on its own processor with its own (virtual) memory.
- The resources of the physical computer are shared to create the virtual machines.

✦ CPU scheduling can create the appearance that users have their own processor.

✦ Spooling and a file system can provide virtual card readers and virtual line printers.

✦ A normal user time-sharing terminal serves as the virtual machine operator's console.

- **System Models**



Non-virtual Machine Virtual Machine

- **Advantages/Disadvantages of Virtual Machines**
- The virtual-machine concept provides complete protection of system resources since each virtual
- machine is isolated from all other virtual machines. This isolation, however, permits no direct sharing of resources.
- A virtual-machine system is a perfect vehicle for operating-systems research and development. System development is done on the virtual machine, instead of on a physical machine and so does not disrupt normal system operation.
- The virtual machine concept is difficult to implement due to the effort required to provide an *exact* duplicate to the underlying machine.

System Generation (SYSGEN)

Operating systems are designed to run on any of a class of machines at a variety of sites with a variety of peripheral configurations. The system must then be configured or generated for each specific computer site, a process sometimes known as system generation (SYSGEN).

SYSGEN program obtains information concerning the specific configuration of the hardware system. To generate a system, we use a special program. The SYSGEN program reads from a given file, or asks the operator of the system for information concerning the specific configuration of the hardware system, or probes the hardware directly to determine what components are there.

The following kinds of information must be determined.

What CPU will be used? What options (extended instruction sets, floating point arithmetic, and so on) are installed? For multiple-CPU systems, each CPU must be described.

How much memory is available? Some systems will determine this value themselves by referencing memory location after memory location until an "illegal address" fault is generated. This procedure defines the final legal address and hence the amount of available memory.

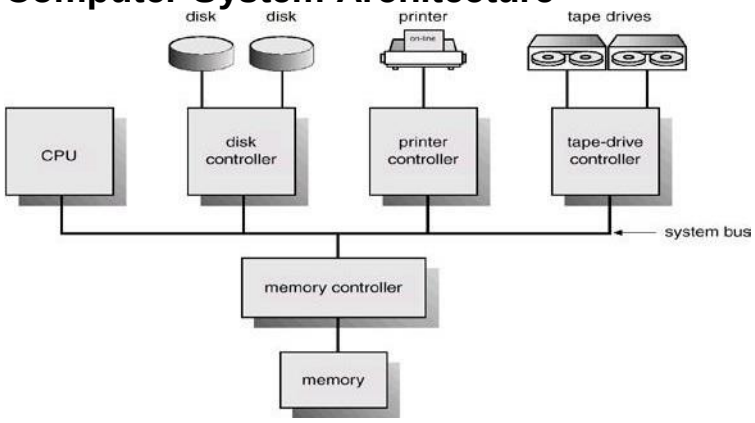
What devices are available? The system will need to know how to address each device (the device number), the device interrupt number, the device's type and model, and any special device characteristics.

What operating-system options are desired, or what parameter values are to be used? These options or values might include how many buffers of which sizes should be used, what type of CPU-scheduling algorithm is desired, what the maximum number of processes to be supported is.

Booting –The procedure of starting a computer by loading the kernel is known as booting the system. Most computer systems have a small piece of code, stored in ROM, known as the bootstrap program or bootstrap loader. This code is able to locate the kernel, load it into main memory, and start its execution. Some computer systems, such as PCs, use a two-step process in which a simple bootstrap loader fetches a more complex boot program from disk, which in turn loads the kernel.

Lecture #5

Computer-System Architecture



Computer-System Operation

- I/O devices and the CPU can execute concurrently.
- Each device controller is in charge of a particular device type.
- Each device controller has a local buffer.
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller.
- Device controller informs CPU that it has finished its operation by causing an *interrupt*.

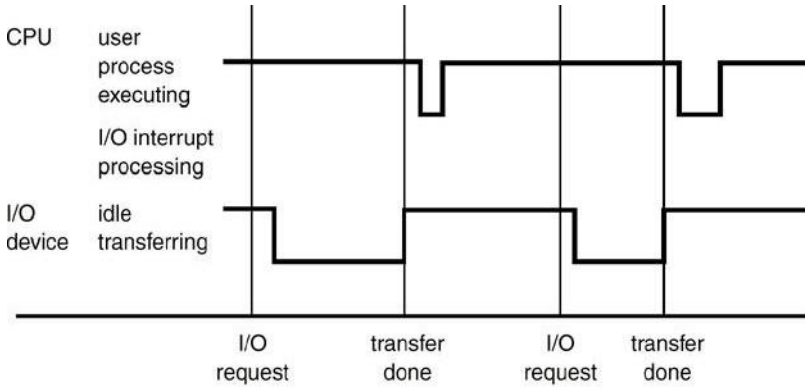
Common Functions of Interrupts

- 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

- 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

Interrupt Time Line For a Single Process Doing Output



I/O Structure

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.

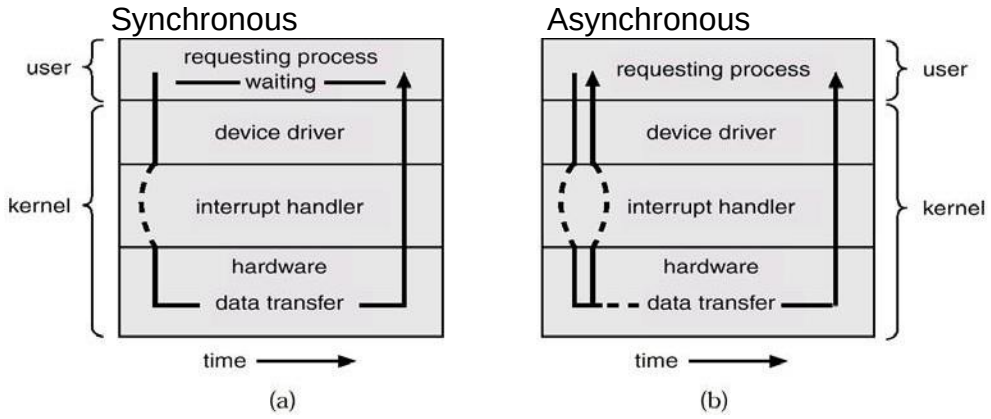
After I/O starts, control returns to user program without waiting for I/O completion.

✦ *System call* – request to the operating system to allow user to wait for I/O completion.

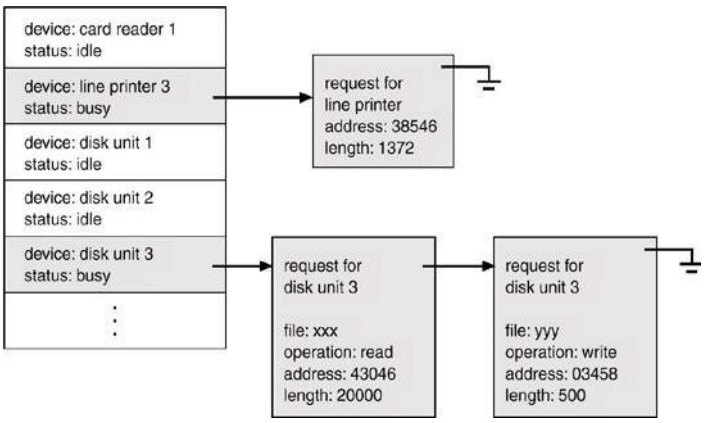
✦ *Device-status table* contains entry for each I/O device indicating its type, address, and state.

✦ Operating system indexes into I/O device table to determine device status and to modify table entry to include interrupt.

Two I/O Methods



Device-Status Table



Direct Memory Access Structure

- Used for high-speed I/O devices able to transmit information at close to memory speeds.
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention.
- Only on interrupt is generated per block, rather than the one interrupt per byte.

Storage Structure

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

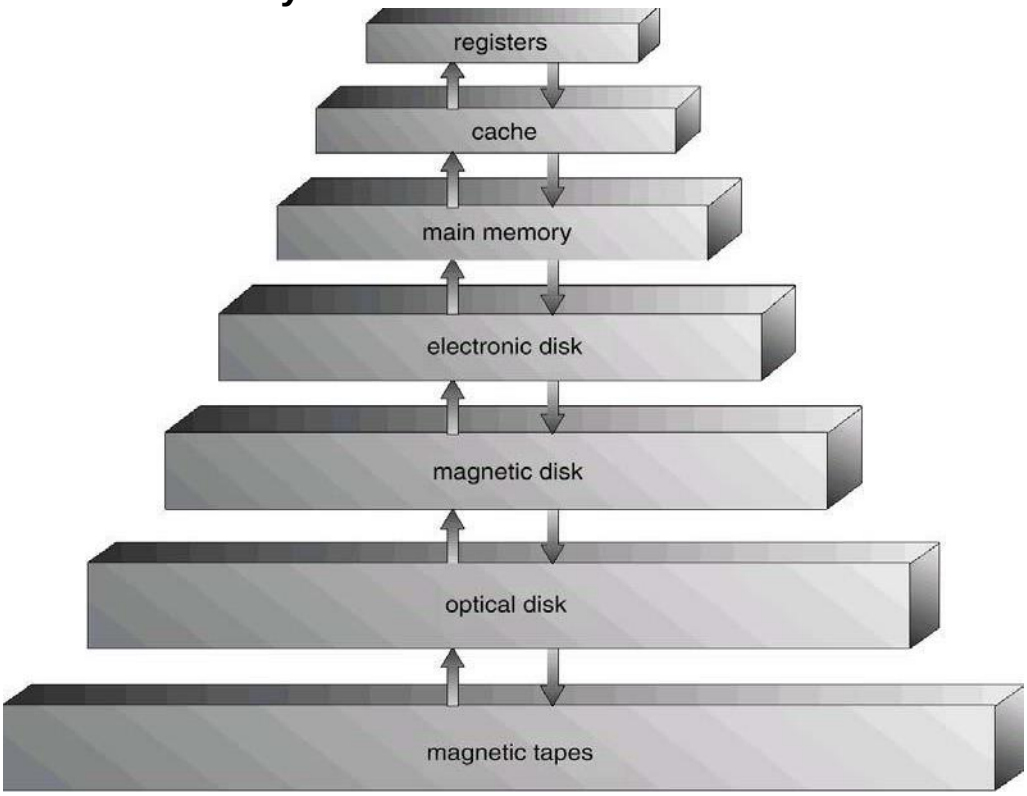
Storage systems organized in hierarchy.

✦ Speed
✦ Cost
✦ Volatility

Caching – copying information into faster storage system; main memory can be viewed as a last *cache* for secondary storage.

- Use of high-speed memory to hold recently-accessed data.
- Requires a *cache management* policy.
- Caching introduces another level in storage hierarchy.
- This requires data that is simultaneously stored in more than one level to be *consistent*.

Storage-Device Hierarchy



Hardware Protection

- Dual-Mode Operation

- I/O Protection
- Memory Protection
- CPU Protection

Dual-Mode Operation

- Sharing system resources requires operating system to ensure that an incorrect program cannot cause other programs to execute incorrectly.
- Provide hardware support to differentiate between at least two modes of operations.
- *User mode* – execution done on behalf of a user.
- *Monitor mode* (also *kernel mode* or *system mode*) – execution done on behalf of operating system.
- *Mode bit* added to computer hardware to indicate the current mode: monitor (0) or user (1).
- When an interrupt or fault occurs hardware switches to *Privileged instructions* can be issued only in monitor mode.

● monitor user ,Interrupt/fault ,set user mode

I/O Protection

- All I/O instructions are privileged instructions.
- Must ensure that a user program could never gain control of the computer in monitor mode (I.e.a user program that, as part of its execution, stores a new address in the interrupt vector).

Memory Protection

- Must provide memory protection at least for the interrupt vector and the interrupt service routines.
- In order to have memory protection, add two registers that determine the range of legal addresses a program may access:

✦ **Base register** – holds the smallest legal physical memory address.

✦ **Limit register** – contains the size of the range

- Memory outside the defined range is protected.

Hardware Protection

- When executing in monitor mode, the operating system has unrestricted access to both monitor and user's memory.
- The load instructions for the *base* and *limit* registers are privileged instructions.

CPU Protection

- *Timer* – interrupts computer after specified period to ensure operating system maintains control.

✦ Timer is decremented every clock tick.

✦ When timer reaches the value 0, an interrupt occurs.

- Timer commonly used to implement time sharing.
- Time also used to compute the current time.
- Load-timer is a privileged instruction.