ITP30002 Operating System

Introduction

(OSC: Chapters 1 and 2)

This lecture note is taken from the instructor's resource of Operating System Concept, 10/e and then partly edited/revised by Shin Hong.

What is an Operating System?

- Operating System refers to a program that acts as an intermediary between a user of a computer and the computer hardware

- Operating system goals:
  - execute application programs safely and efficiently
  - make developers write portable and extensible programs
  - assist end-users to operate computer systems conviently
  - improve utilization of hardware resources

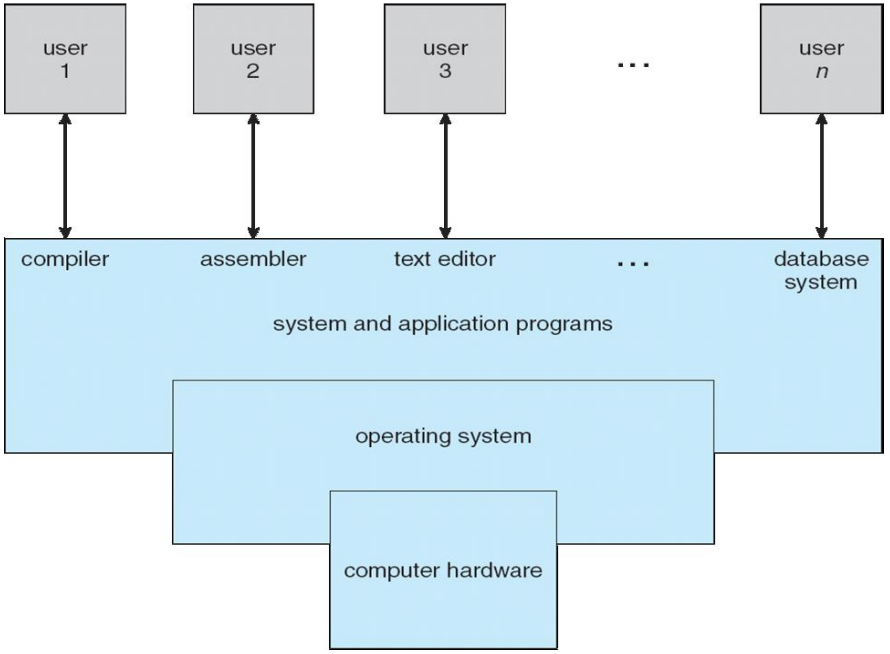
Computer System Structure - Components

- Hardware: provides primitive (fixed) computing functionalities
  - e.g., CPU, memory, I/O devices (interacting with human and other machines)

- Software
  - Application programs: define the ways in which the system resources are used to solve computing problems of the users
    - e.g., word processors, spreadsheet, web browsers
  - Operating system: controls and coordinates uses of hardware/software components by applications and users

Operating System Definition (1/3)

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



Operating System Definition (2/3)

- Various definitions
  - Everything a vendor ships when you order an operating system
- "The one program running at all times on the computer" is the **kernel**.
- Everything else is either
  - a system program (ships w/ the operating system), or
  - an application program.

Operating System Definition (3/3)

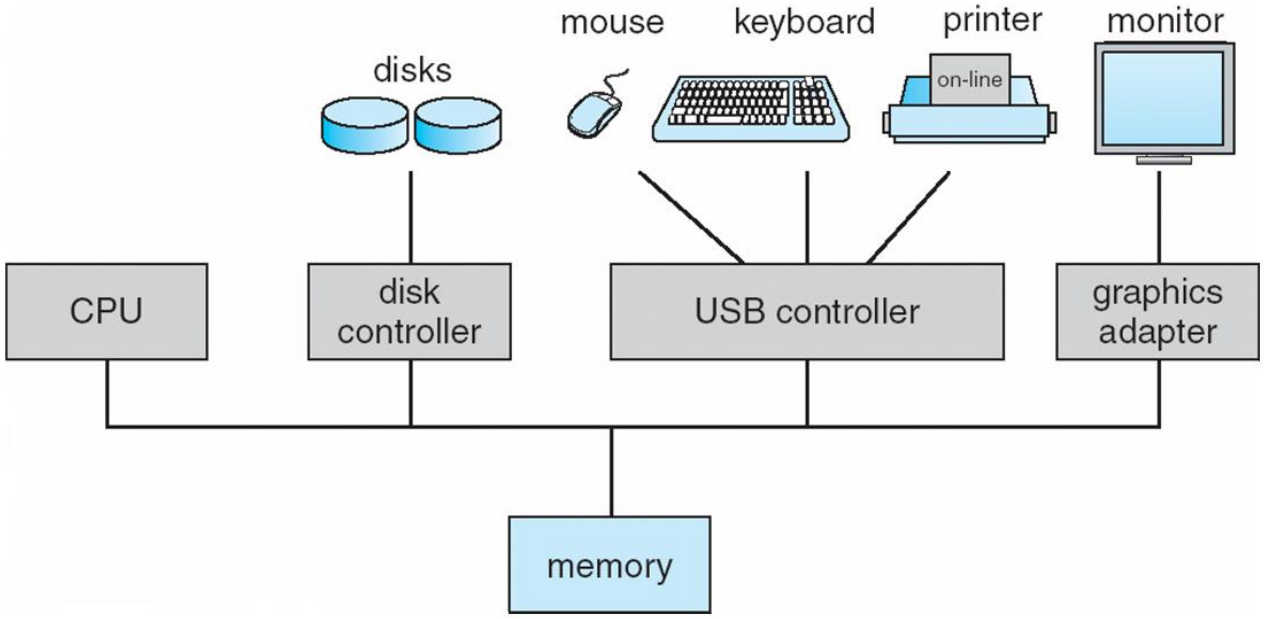
A definition of operating system

- OS is a program that provides abstractions of hardware components and application programs (software)
  - manage various hardware resources efficiently
  - support an application program to operate different hardware devices in a simple and consistent way
  - support interaction/communication among application programs in a simple and consistent way

- OS aims to offering a platform where different software systems are integrated into a new system conveniently

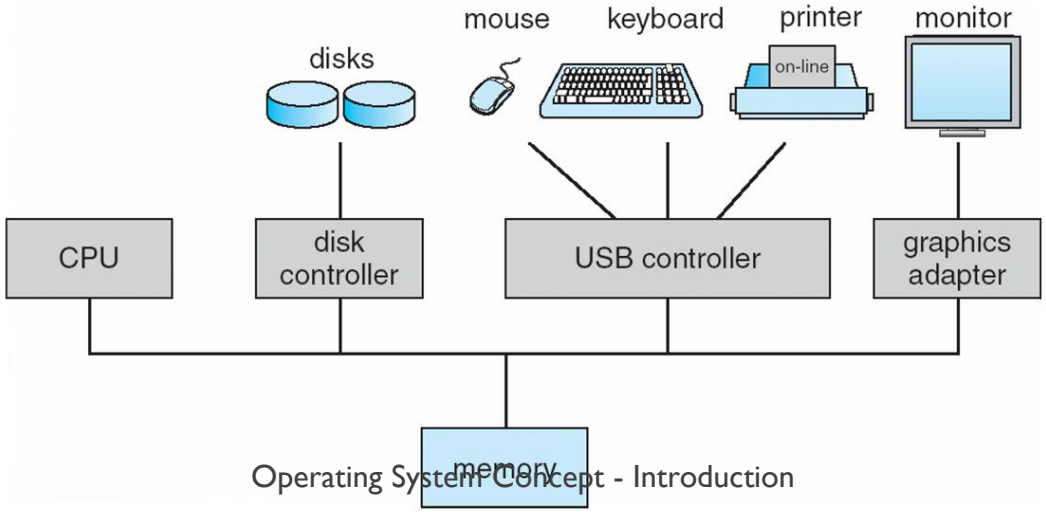
Computer System Organization

- 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



Computer-System Operation

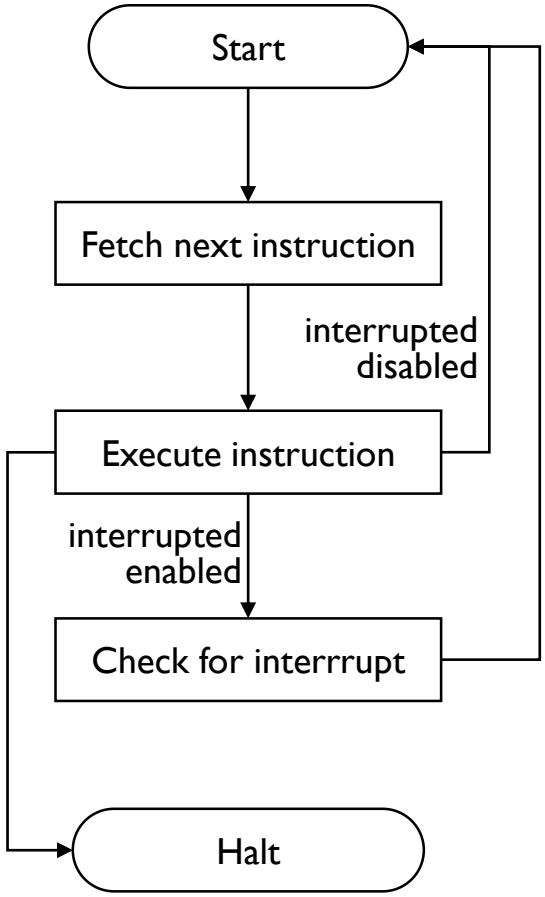
- I/O devices and the CPU can execute concurrently
- Each device controller is in charge of a particular device type
- Each device controller has a local buffer
- I/O is from the device to local buffer of controller
- Device controller informs CPU that it has finished its operation by causing an interrupt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



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
- A **trap** or **exception** is a software-generated interrupt caused either by an error or a user request
- An operating system is **interrupt driven**

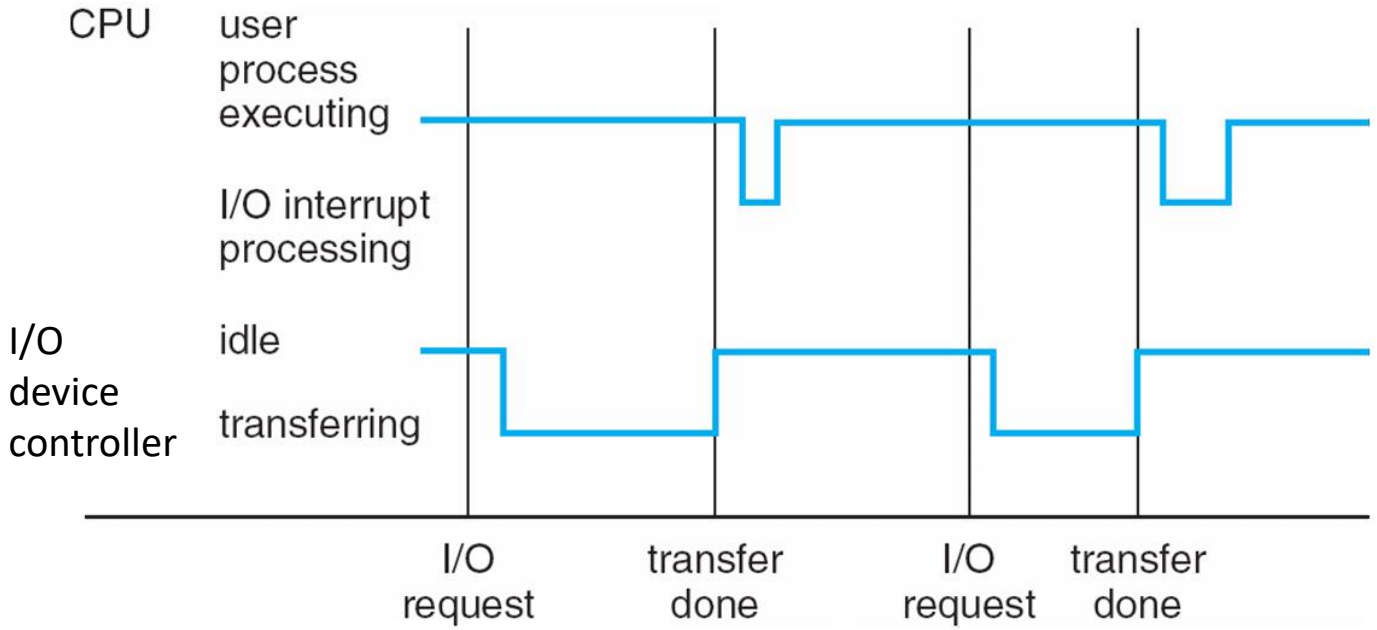
Instruction Cycle



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 code segments to determine what action should be taken for each type of interrupt

Interrupt Timeline



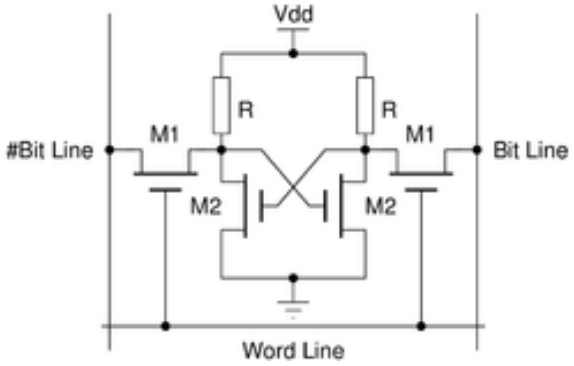
I/O Structure

- 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 requests the OS to allow user programs to wait for I/O completion
  - Device-status table contains entry for each I/O device indicating its type, address, and state
  - OS indexes into I/O device table to determine device status and to modify table entry to include interrupt

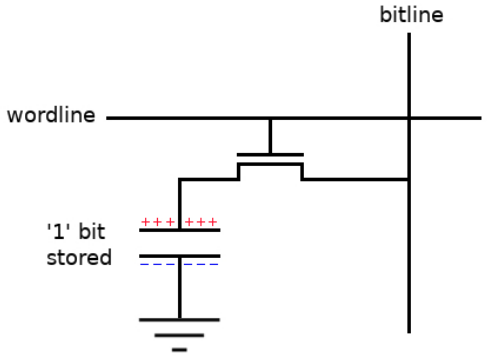
Storage Structure

- Main memory: only large storage media that the CPU can access directly
  - **Random access**
  - Typically **volatile**
- Secondary storage: extension of main memory that provides (usually) large and **nonvolatile** storage capacity
  - Hard 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
  - Solid-state device: faster than hard disks, nonvolatile
    - Various technologies
    - Becoming more popular

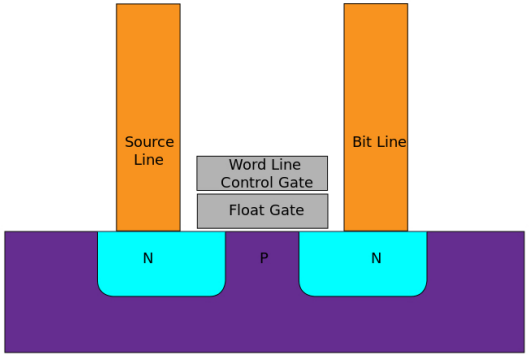
SRAM, DRAM and FLASH



<SRAM>



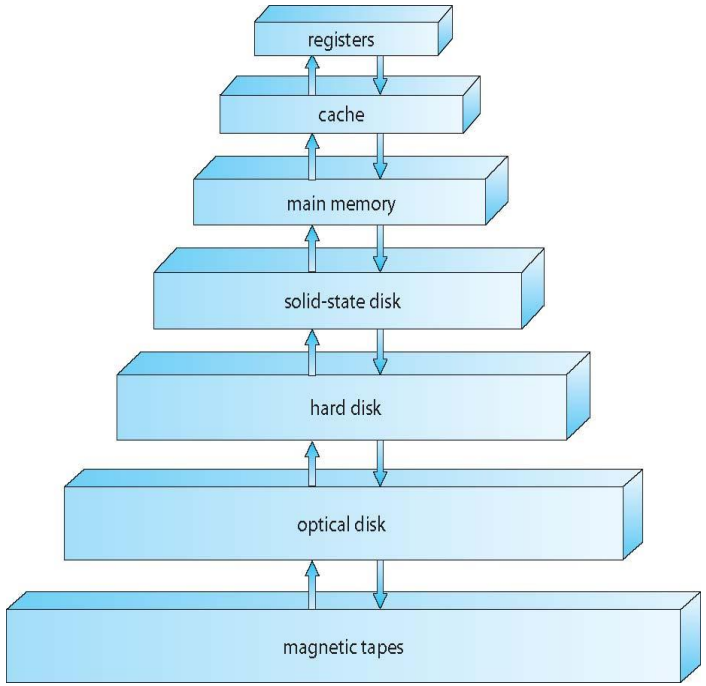
<DRAM>



<FLASH>

Storage Hierarchy

- Storage systems organized in hierarchy
  - Speed
  - Cost
  - Volatility
- Caching: copying information into faster storage system
  - main memory can be viewed as a cache for secondary storage
- Device Driver for each device controller to manage I/O
  - provides uniform interface between controller and kernel

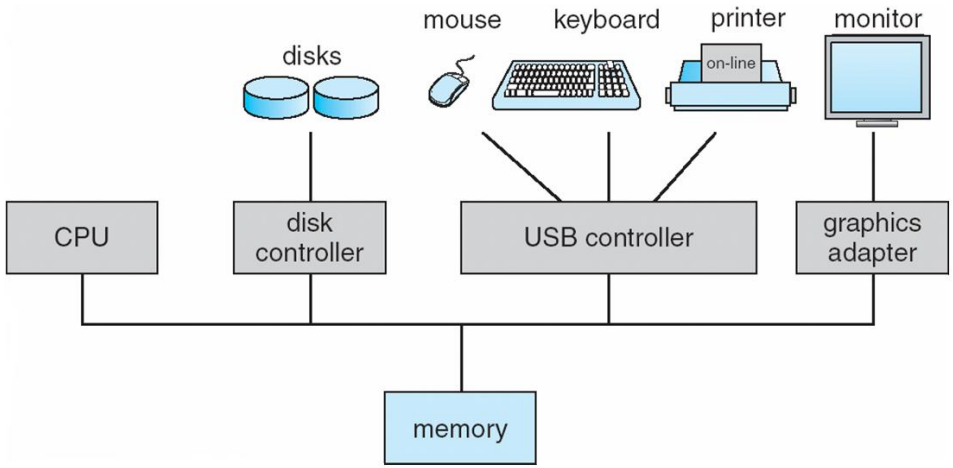


Caching

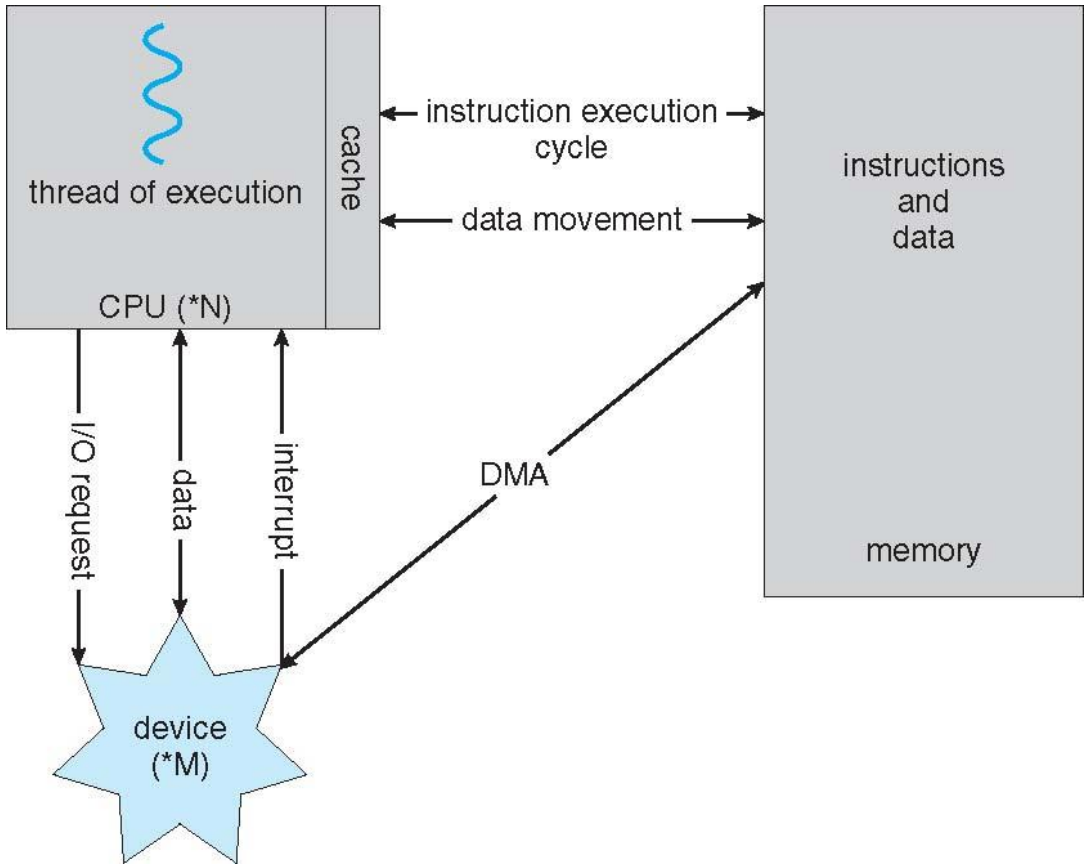
- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

Direct Memory Access (DMA) 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 directly to main memory without CPU intervention
- Only one interrupt is generated per block, rather than the one interrupt per byte



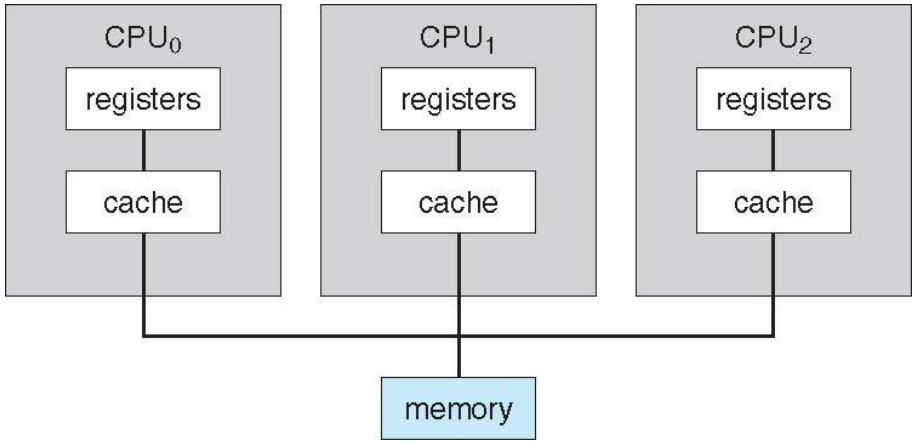
How a Modern Computer Works



A von Neumann architecture

Computer-System Architecture

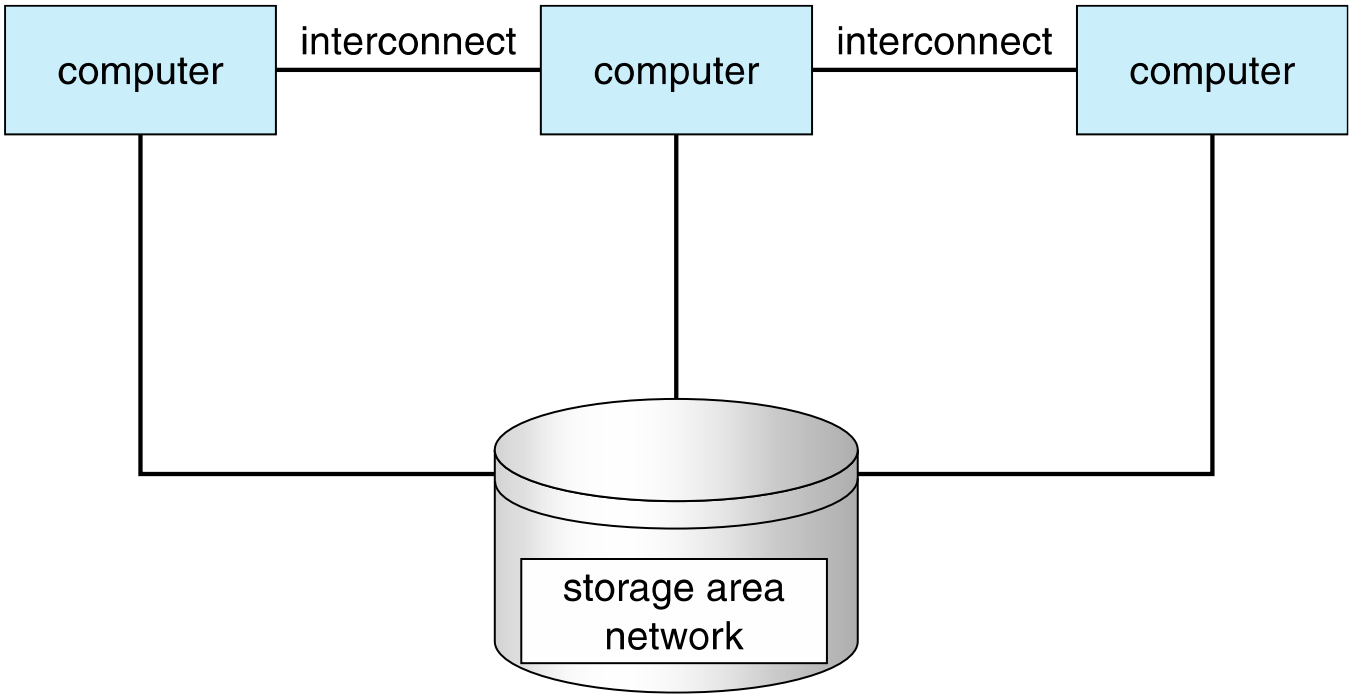
- Most systems use a single general-purpose processor
  - Most systems have special-purpose processors as well
- Multiprocessors systems growing in use and importance
  - Advantages include:
    1. Increased throughput
    2. Economy of scale
    3. Increased reliability – graceful degradation or fault tolerance
  - Two types:
    1. Asymmetric Multiprocessing – each processor is assigned a specie task.
    2. Symmetric Multiprocessing – each processor performs all tasks



Clustered Systems

- Like multiprocessor systems, but multiple systems working together
  - Usually sharing storage via a storage-area network (SAN)
  - Provides a high-availability service which survives failures
    - Asymmetric clustering has one machine in hot-standby mode
    - Symmetric clustering has multiple nodes running applications, monitoring each other
  - Some clusters are for high-performance computing (HPC)
    - Applications must be written to use parallelization
  - Some have distributed lock manager (DLM) to avoid conflicting operations

Clustered Systems



Operating System

Two primary purposes of OS

- To provide application programs with simple, uniform mechanisms for operating/manipulating different hardware devices
- To protect the hardware from misuse by runaway programs

Key abstractions

- process: an abstraction of a program execution
- memory (virtual memory): an abstraction of data as an array
- file: an abstraction of communication (I/O) as a stream

Operating System Structure

- **Multiprogramming** (**Batch system**) for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - Multiprogramming organizes jobs (code and data) so CPU always works
    - A subset of total jobs in system is kept in memory
    - One job selected and run via **job scheduling**
    - When it has to wait (for I/O for example), OS switches to another job

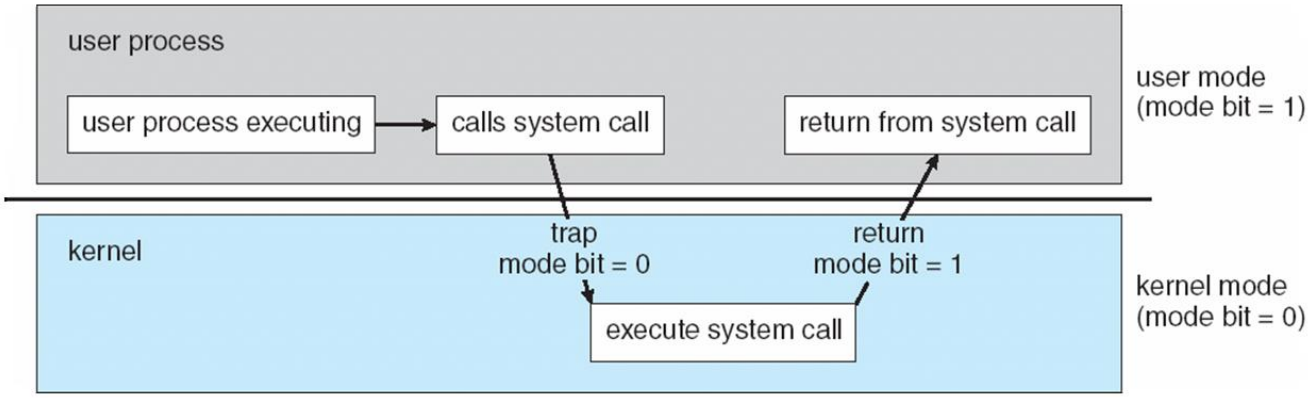
- **Timesharing** (**multitasking**) is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, **Response time** should be < 1 second
  - Each user has at least one program executing in memory ⇨**process**
  - If several jobs ready to run at the same time ⇨ **CPU scheduling**
  - **Virtual memory** allows processes not completely in memory
    - If processes don't fit in memory, **swapping** moves them in and out to run

Operating-System Operations (1/2)

- **Interrupt driven** (hardware and software)
  - Hardware interrupt by one of the devices
  - Software interrupt (**exception** or **trap):**
    - Software error (e.g., division by zero)
    - Request for operating system service
    - Other process problems include infinite loop, processes modifying each other or the operating system

Operating-System Operations (2/2)

- Dual-mode operation allows OS to protect itself and other system components
  - User mode and kernel mode
  - Mode bit provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as privileged, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user



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
  - Process executes instructions sequentially, one at a time, until completion
- 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
  - Concurrency by multiplexing the CPUs among the processes / threads

Process Management Activities

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

- To execute a program all (or part) of the instructions must be in memory
- All (or part) of the data that is needed by the program must be in memory.
- Memory management determines what is in memory and when
  - Optimizing CPU utilization and computer response to users
- 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
  - File abstracts physical properties to logical storage unit
  - Each medium is controlled by device (i.e., disk drive, tape drive)
    - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and directories
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

Performance of Various Levels of Storage

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

Movement between levels of storage hierarchy can be explicit or implicit

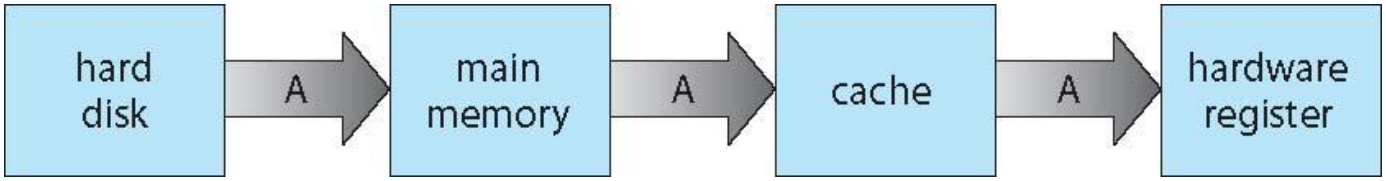
Performance of Various Levels of Storage

System Event	Actual Latency	Scaled Latency
One CPU cycle	0.4 ns	**1 s**
Level 1 cache access	0.9 ns	**2 s**
Level 2 cache access	2.8 ns	**7 s**
Level 3 cache access	28 ns	**1 min**
Main memory access (DDR DIMM)	~100 ns	**4 min**
Intel Optane memory access	<10 µs	**7 hrs**
NVMe SSD I/O	~25 µs	**17 hrs**
SSD I/O	50–150 µs	**1.5–4 days**
Rotational disk I/O	1–10 ms	**1–9 months**
Internet call: San Francisco to New York City	65 ms[3]	**5 years**
Internet call: San Francisco to Hong Kong	141 ms[3]	**11 years**

Gregg, Brendan. "Systems Performance: Enterprise and the Cloud." March 2015. www.brendangregg.com/sysperfbook.html. A CPU cycle refers to a single tick of a processor's internal clock. It is during the ticks of this clock that processors work their way through the pipeline of instructions awaiting computation.

Migration of data "A" from Disk to Register

- Multitasking environments must be careful to use most recent value, no 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

I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

Protection and Security

- **Protection**: any mechanism for controlling access of processes or users to resources defined by the OS
- **Security**: defense against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights