1. Introduction

ECE30021/ITP30002 Operating Systems

Agenda

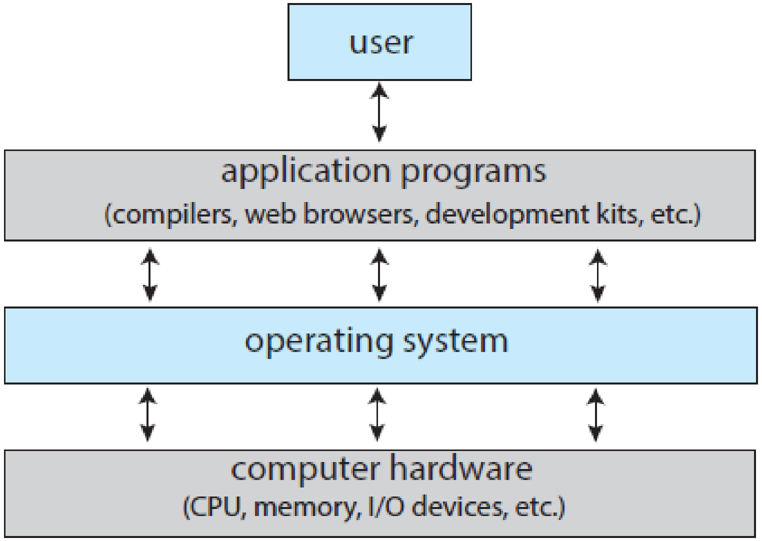
- Definitions of operating system
- Computer system organization and operation
- Computer system architecture
- Operating system structure and operation
- Core components of OS
- Computing environments

What is an Operating System?

- There is no completely adequate definition for OS
- Roughly, an OS is …
  - Intermediary between the user and H/W
  - Kernel + additional programs
    - Kernel: core of OS that runs at all times
      - System / application programs are not included
    - Middleware: a set of software frameworks that provide additional services to app developers (e.g. database, multimedia, graphics)
- 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

Components of Computer System

- Computer system
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - 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



What Operating Systems Do?

- OS provides environment
  - Performs no useful function by itself.
  - However, the user can do something easily in the environment provided by OS.

- OS manages system resources
  - Let the users and the programs share the system resources in time and space.
  - Let the user use the computer hardware in an efficient manner.

User View

- Varies according to the interface being used

System	Major design issues
Single-user system (PC)	Ease of use, Performance, security
Mobile devices	Touch screen, Voice recognition interface, Battery life
Multi-user system (Mainframe, minicomputer)	Resource utilization, Information exchange
Network-based system (workstation/server)	Individual usability vs. resource, utilization
Embedded computers	Little or no user view (numeric keypads + indicator lights)

System View

- An OS is a resource allocator
  - Manages all resources
    - CPU, memory, file-storage, I/O device, …
  - Decides between conflicting requests for efficient and fair resource use
- An OS is a control program
  - Controls execution of programs to prevent errors and improper use of the computer

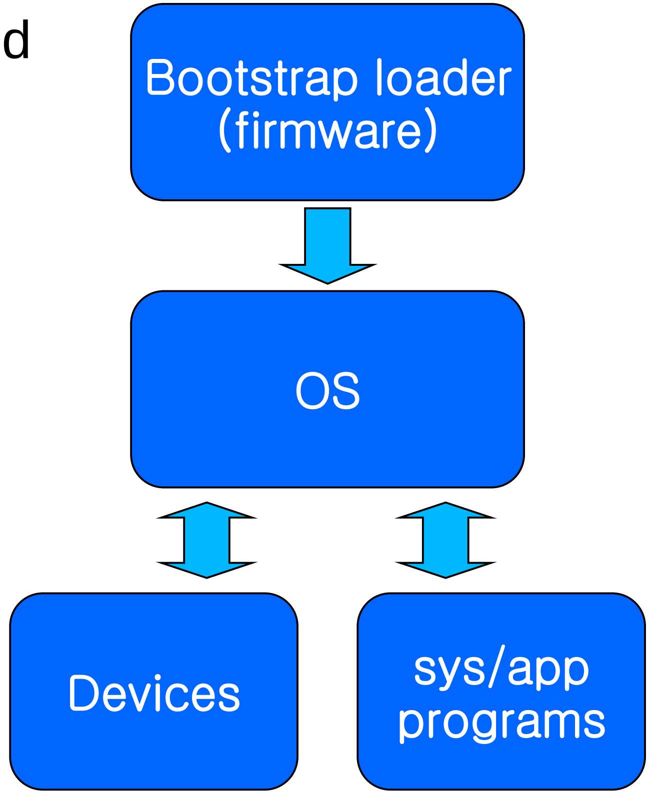
Agenda

- Definitions of operating system
- **Computer system organization and operation**
- Computer system architecture
- Operating system structure and operation
- Core components of OS
- Computing environments

Computer System Operation

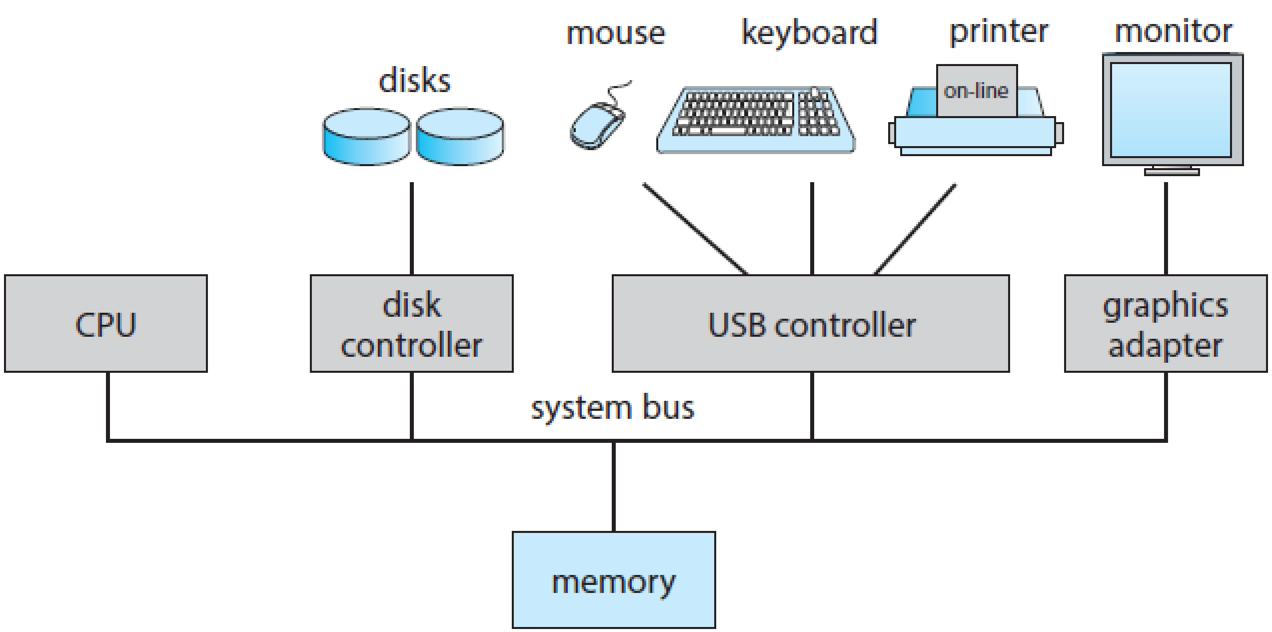
- What happens when a computer starts running?
  1. Bootstrap program (firmware) is executed
     - Diagnose and initialize system
     - Load and execute OS kernel
       - Bootstrap loader
  2. OS kernel
     - Boots (init)
     - **Waits for some event**
     - **Handles event**

→ Modern operating systems are **interrupt driven programs**



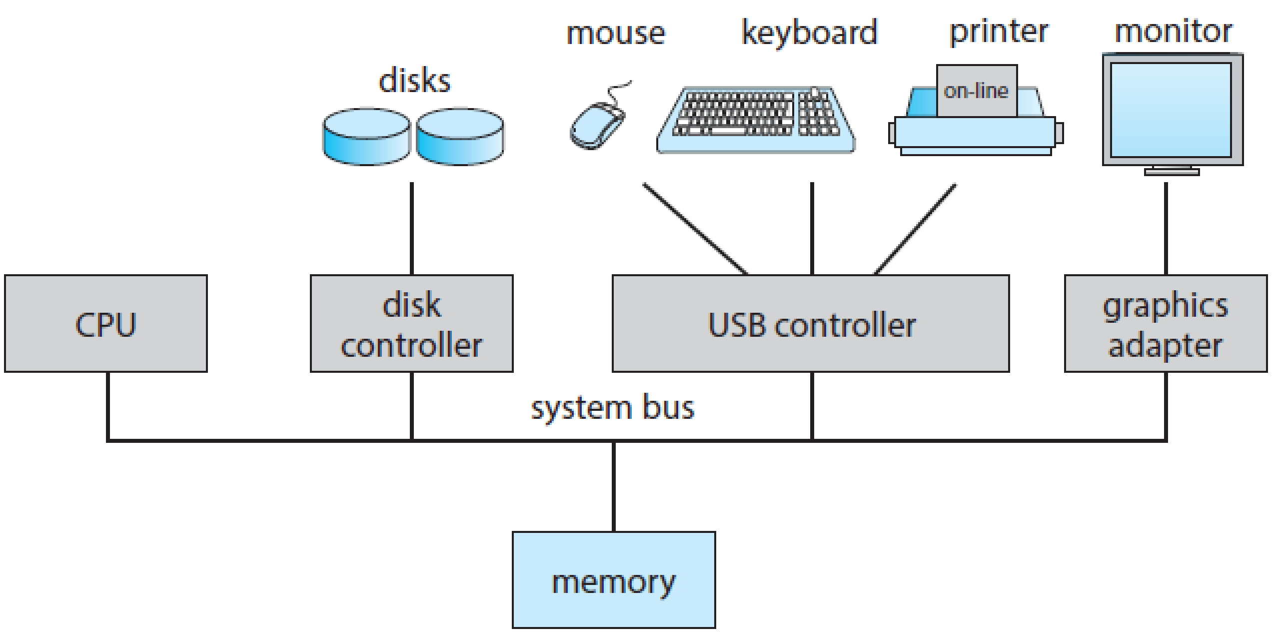
Modern Computer System

- Common bus: a subsystem that transfers data between computer components
- CPU, memory and device controllers are connected to a common bus
- I/O devices and the CPU can execute concurrently.
  - Competing for memory → synchronization issue



Modern Computer System (cont'd)

- 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.



Interrupt

- **Interrupt**: an asynchronous signal from hardware or software indicating the need for attention
  - Supported by H/W
  - Each type of interrupt is associated with a number (IRQ number)
  - Handled by **interrupt handler**

- Separate segments of code determine what action should be taken for each type of interrupt.
  - Table of interrupt handler: **interrupt vector**

- A mechanism to process event
  - H/W interrupt
    - Sending signal to CPU
  - S/W interrupt
    - System call (= monitor call): request from program
      - I/O access, memory allocation, …
    - Exception
      - Divide by zero, invalid memory access, I/O exception, …

Intel Processor Event–Vector Table

- 0~31: non-maskable, 32~255: maskable

vector number	description
0	divide error
1	debug exception
2	null interrupt
3	breakpoint
4	INTO-detected overflow
5	bound range exception
6	invalid opcode
7	device not available
8	double fault
9	coprocessor segment overrun (reserved)
10	invalid task state segment
11	segment not present
12	stack fault
13	general protection
14	page fault
15	(Intel reserved, do not use)
16	floating-point error
17	alignment check
18	machine check
19–31	(Intel reserved, do not use)
32–255	maskable interrupts

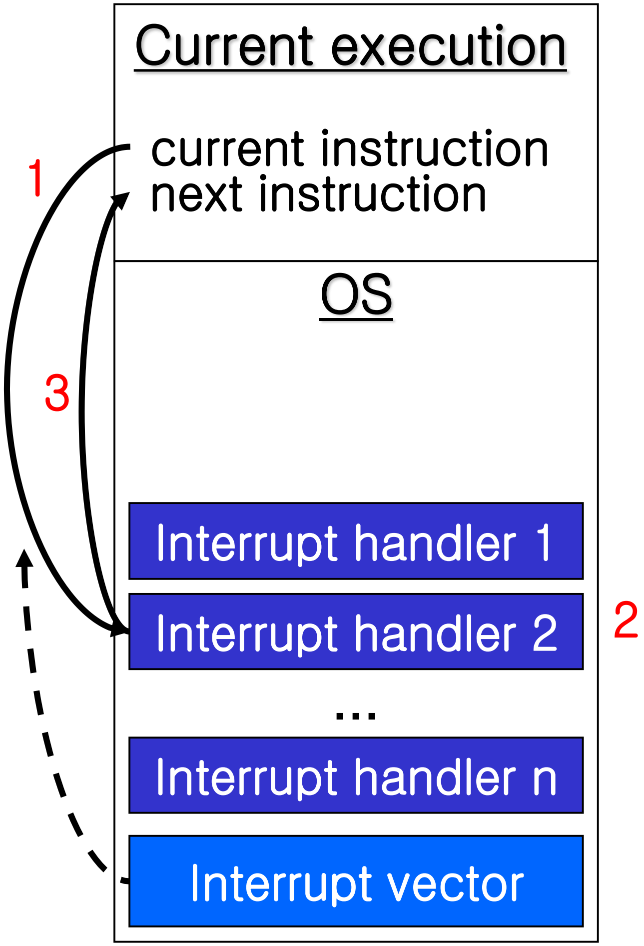
Hardware Process

- Execution cycle of CPU
  - PC: Program Counter
  - IRQ: Interrupt ReQuest
  - ISR: Interrupt Service Routine

```
InterruptRequest = FALSE;          // interrupt request line
...
while(haltFlag not set during execution){
    IR = memory[PC];      // Fetch into the IR
    PC++;                 // Point the PC at the next instruction
    execute(IR);          // Execute the current instruction
    if(InterruptRequest) {  // Interrupt the current process
        save_PC();        // Save the PC
        restore_PC(IRQ);  // Branch to the ISR for this IRQ
    }
}
```

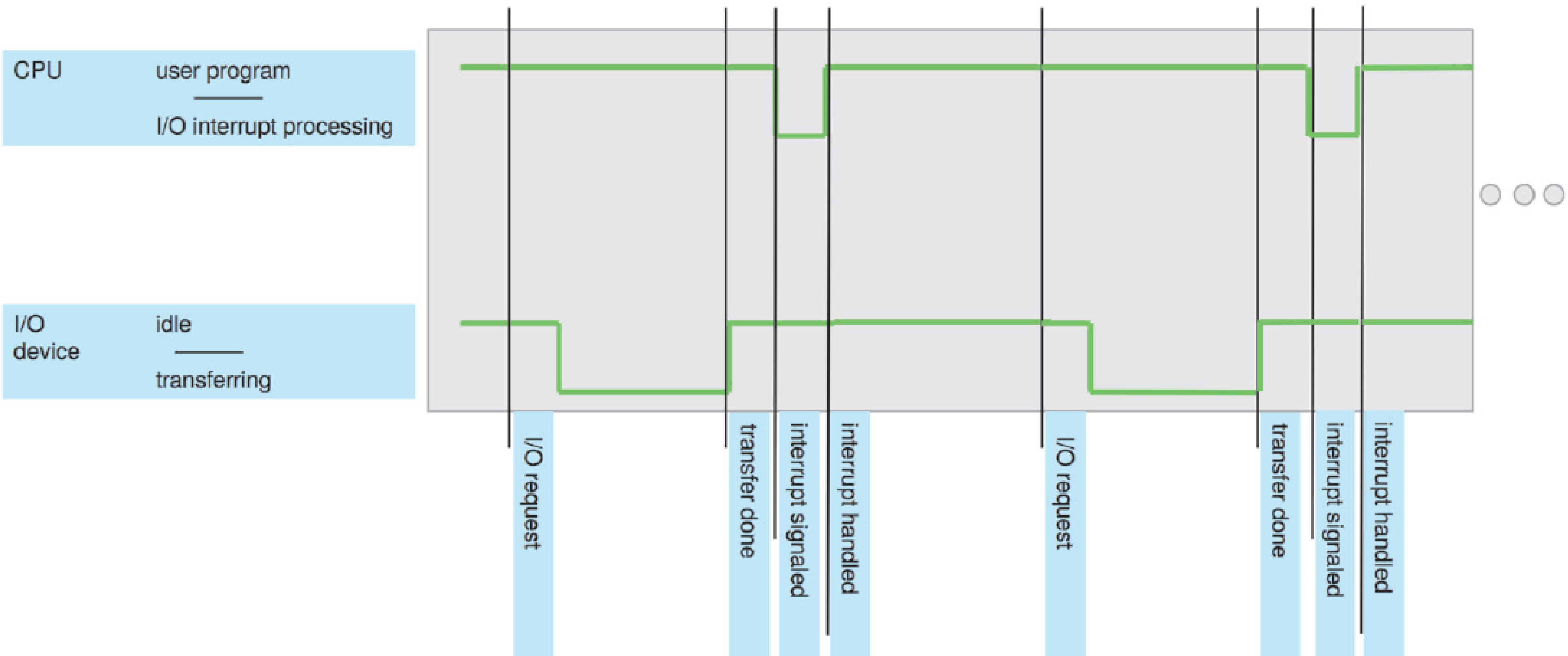
Interrupt Mechanism

- Interrupt handling
  1. CPU stops current work and transfers execution to interrupt handler
     - Interrupt vector: table of interrupt handlers for each types interrupt
  2. Interrupt is handled by corresponding handler
  3. Return to the interrupted program
  - Before interrupt handler is invoked, necessary information should be saved (return address, state)



Interrupt-based I/O

1. CPU sends request and continue current process or do another job.
2. When data transfer is done, I/O device interrupts

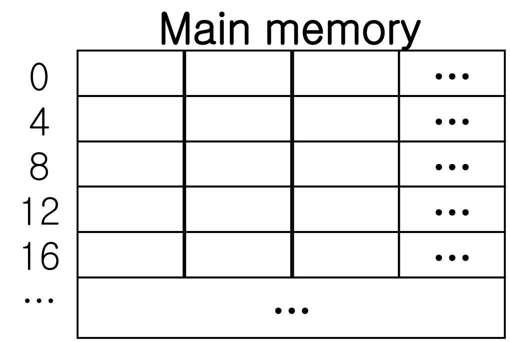


Instruction-Execution Cycle

- Fetches an instruction from memory and stores that instruction in the instruction register.
- The instruction is then decoded and may cause operands to be fetched from memory and stored in some internal register.
- After the instruction on the operands has been executed, the result may be stored back in memory.

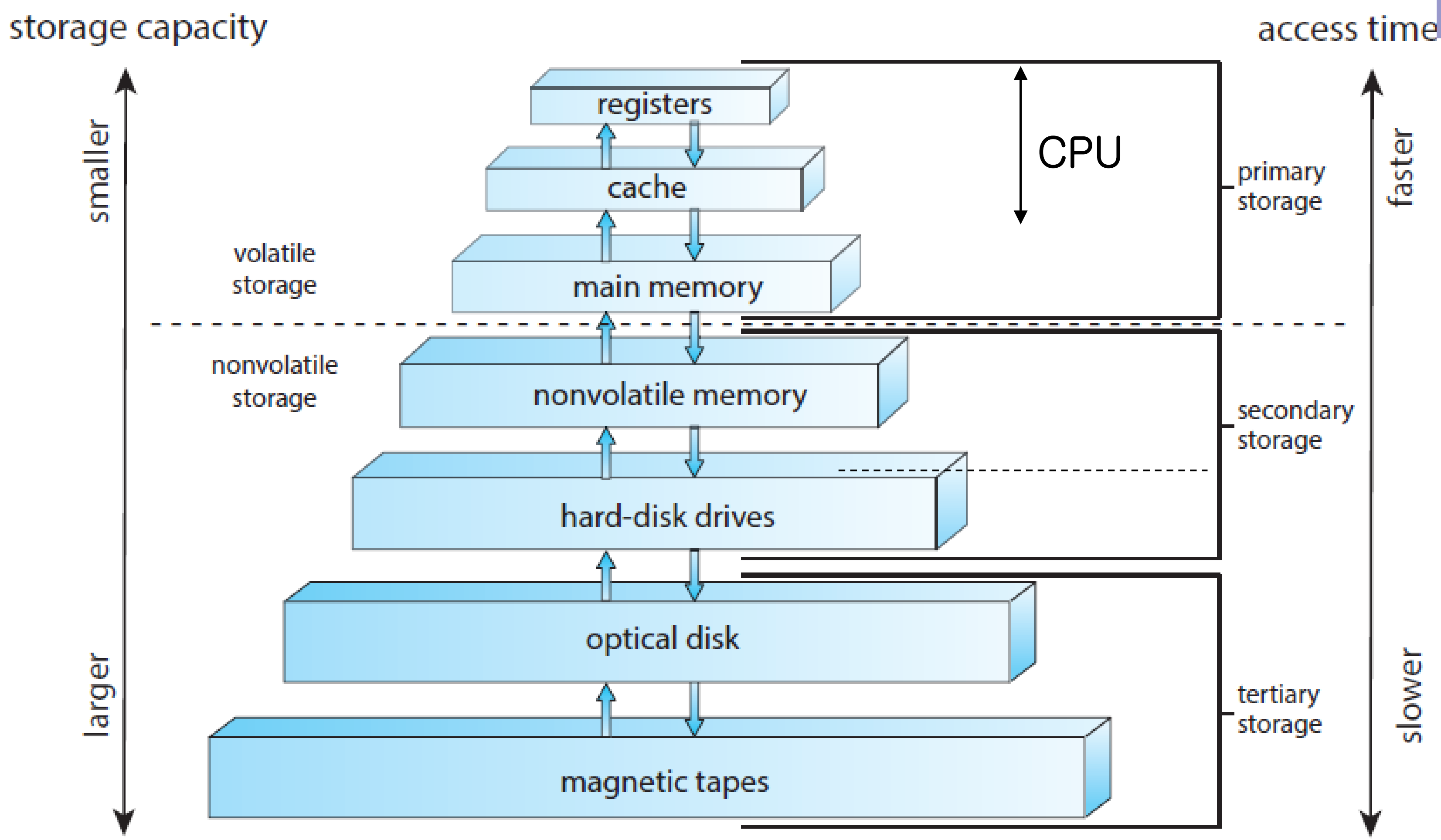
Main Memory

- Register
  - Small, fast, memory in CPU
- Main memory
  - Array of cells to store data or instruction
  - Each cell is identified by its address.
  - Fast, volatile



- All forms of memory provide an array of bytes. Each byte has its own address.
  - **load** instruction moves a byte or word from main memory to an internal register within the CPU, whereas the
  - **store** instruction moves the content of a register to main memory

Storage Structure



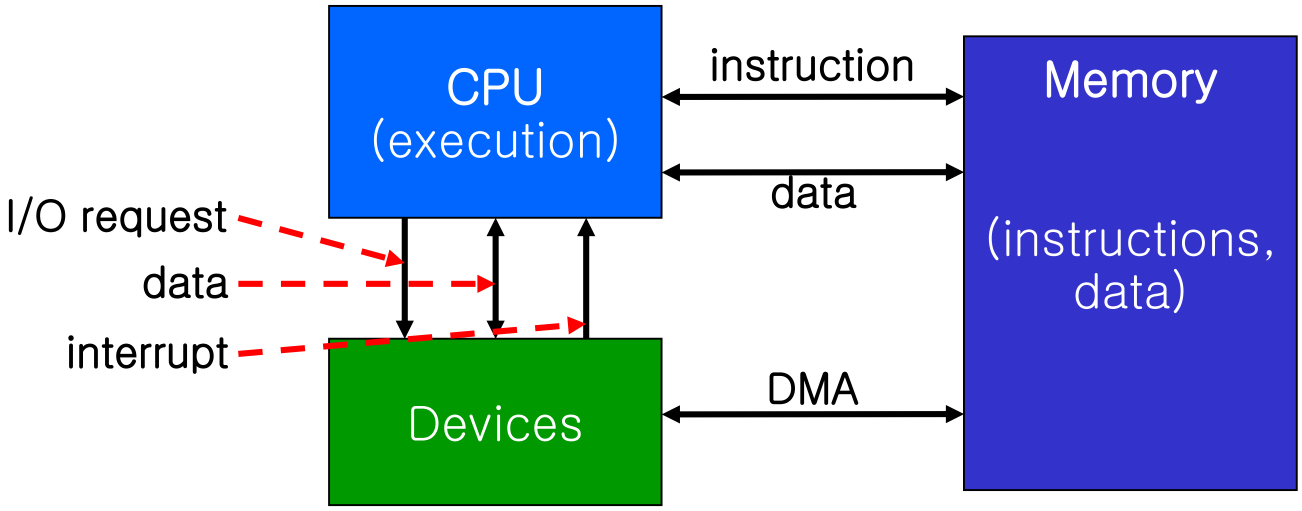
* Electronic disk: SSD, flash memory, NVRAM

Size and Speed of Storages

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

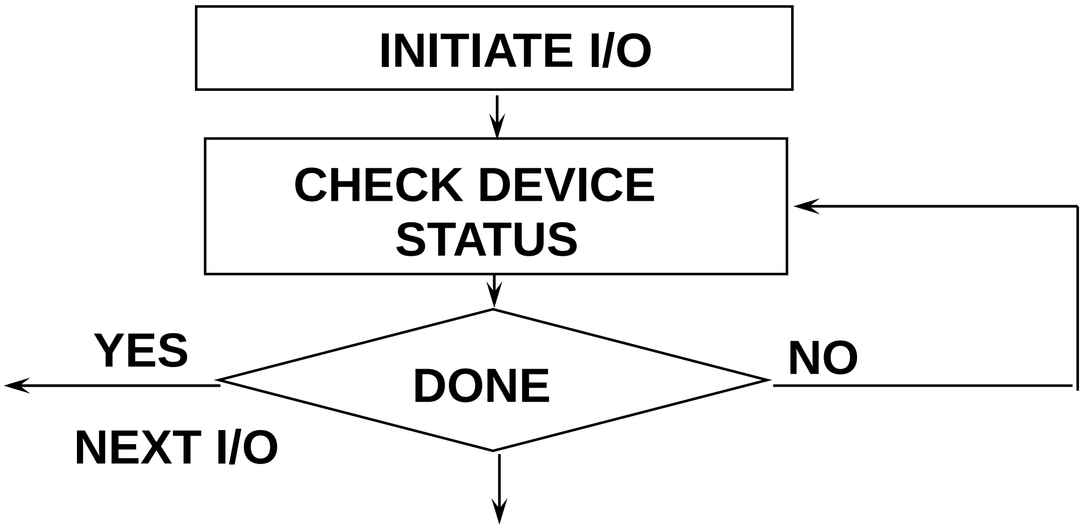
I/O Device Access

- Old systems
  - Busy waiting
- Modern systems
  - Interrupt-driven I/O
  - DMA (Direct memory Access)
    - For large bulk of data



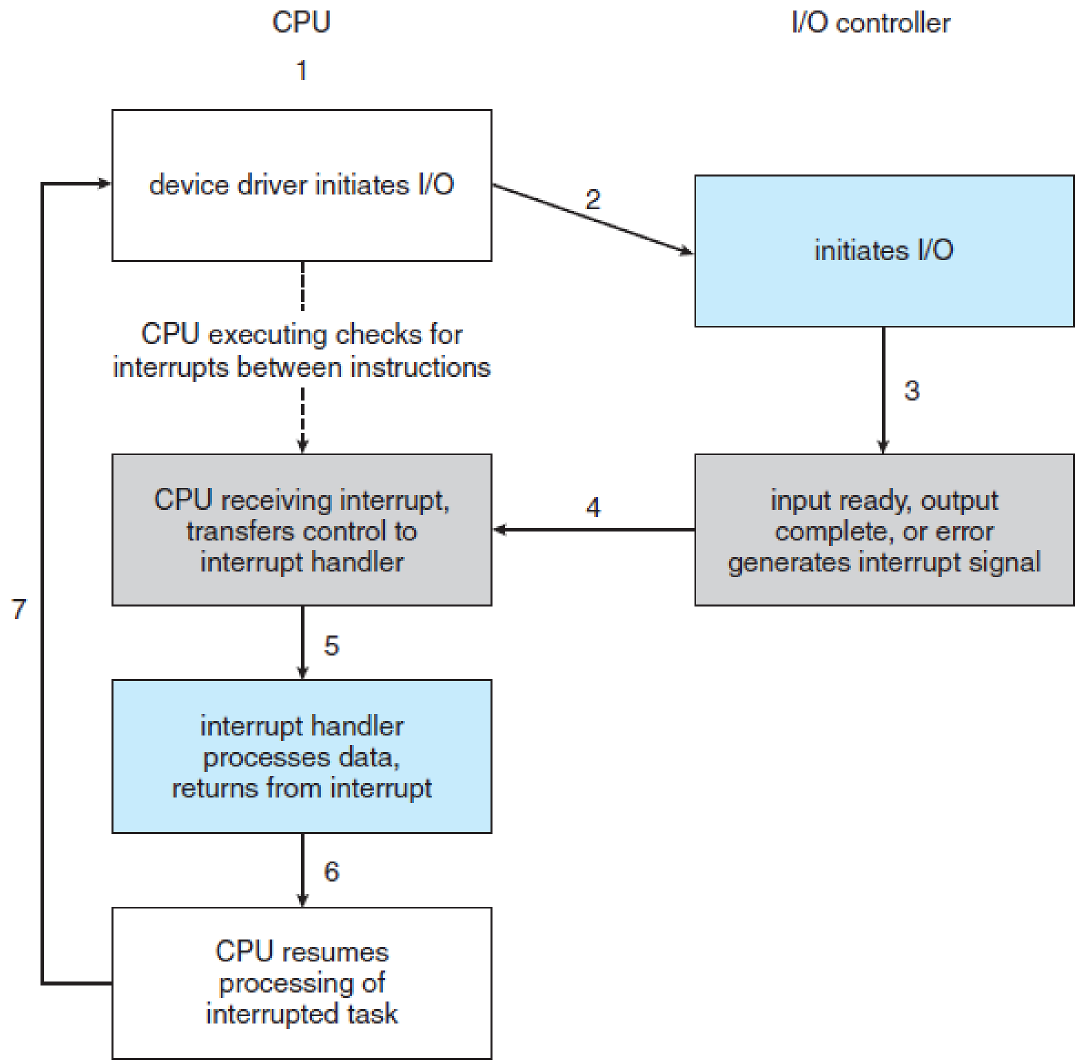
I/O Device Access

- Busy waiting (old system)
  - CPU checks devices status periodically



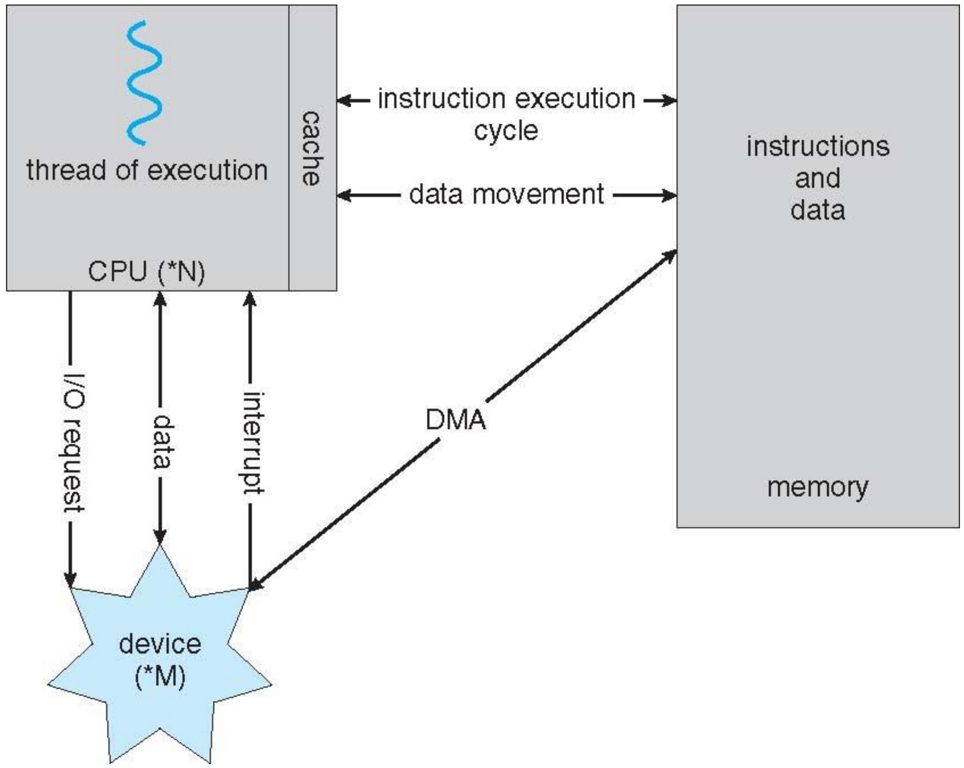
→ Problem: CPU does nothing but waits for I/O device

Interrupt-driven I/O Cycle



I/O Device Access

- DMA (Direct Memory Access)
  - Device controller directly transfers large bulk of data into main memory
  - CPU doesn't have to be involved in data transfer



Agenda

- Definitions of operating system
- Computer system organization and operation
- **Computer system architecture**
- Operating system structure and operation
- Core components of OS
- Computing environments

Computer System Architecture

- Single processor systems
- Multi-processor systems
- Clustered systems

Single-Processor systems

- Single general-purpose main CPU (usually single core)

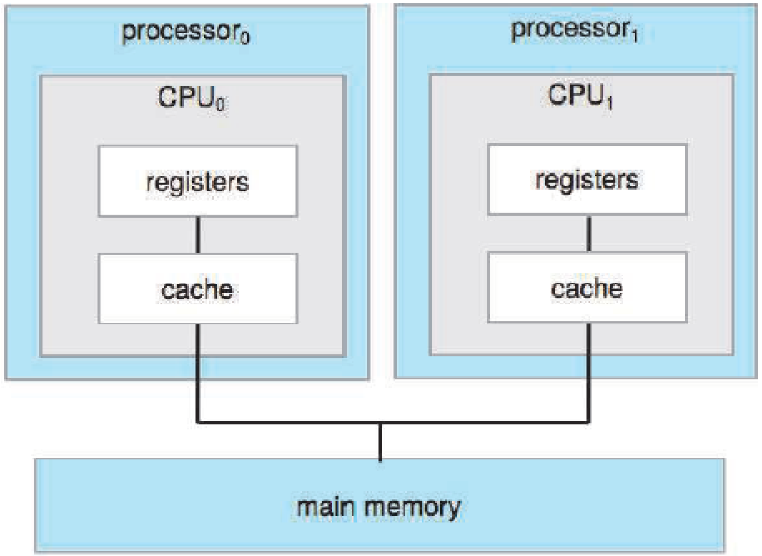
- Some special-purpose processors for devices
  - Not for user process

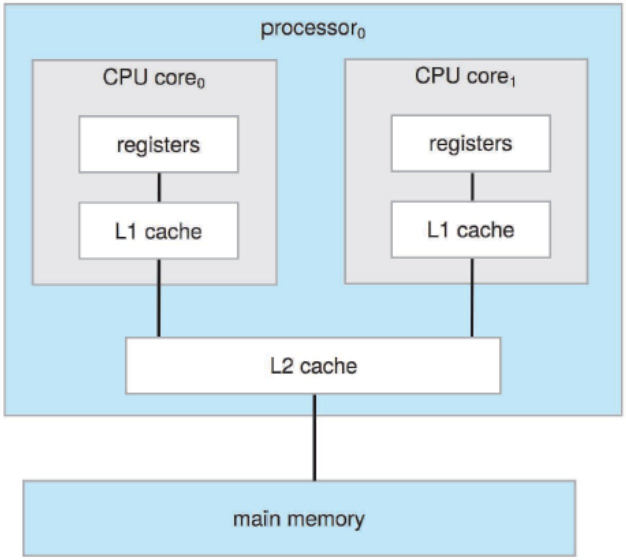
  Ex) processor of device controller

- Advantage of single processor system
  - Simple
  - Cheap

Multi-Processor Systems

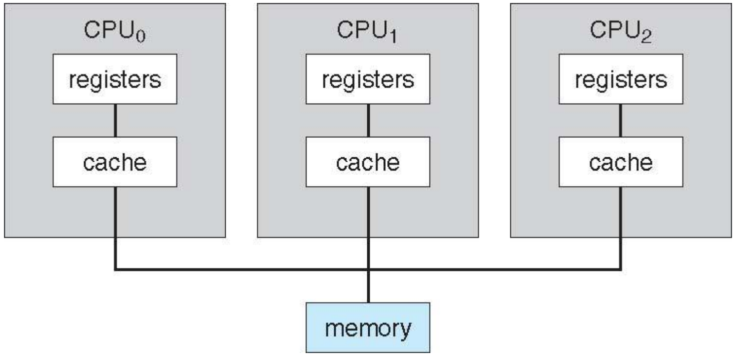
- Two or more processors in close communication
  - Tightly-coupled system
    - Shared bus, clock, memory, peripheral
- Usually, includes multi-core systems
  - Core: the component that executes instructions and registers for storing data locally



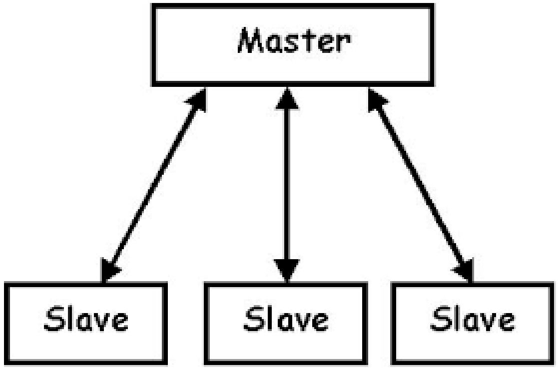


Multi-Processor Systems

- Categories
  - Symmetric multiprocessing (SMP)
    - All processors are peers

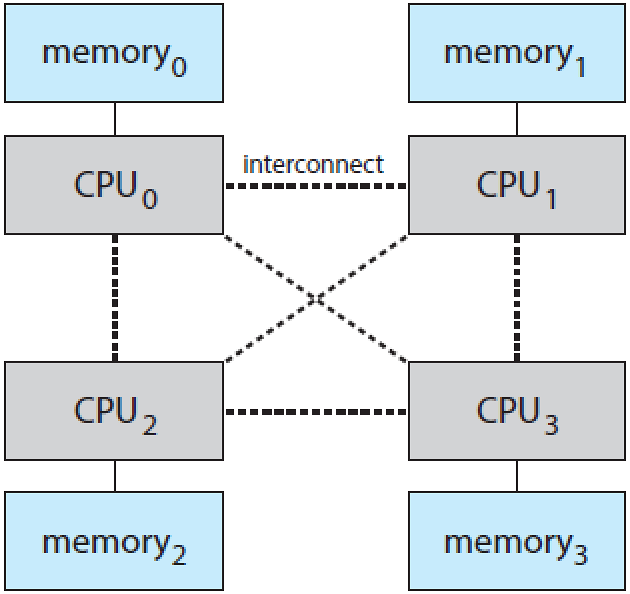


  - Asymmetric multiprocessing
    - Master-slave relation
      - The master schedules and allocates the slaves.



NUMA System

- Non-uniform memory access
  - A group of CPUs, each of which has its own local memory that is accessed via a small, fast local bus.

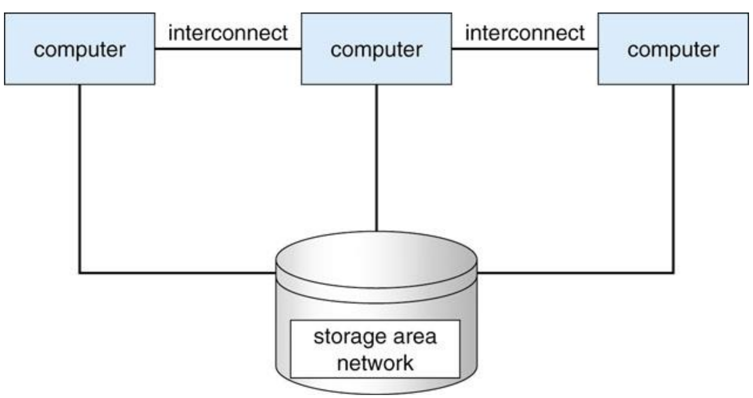


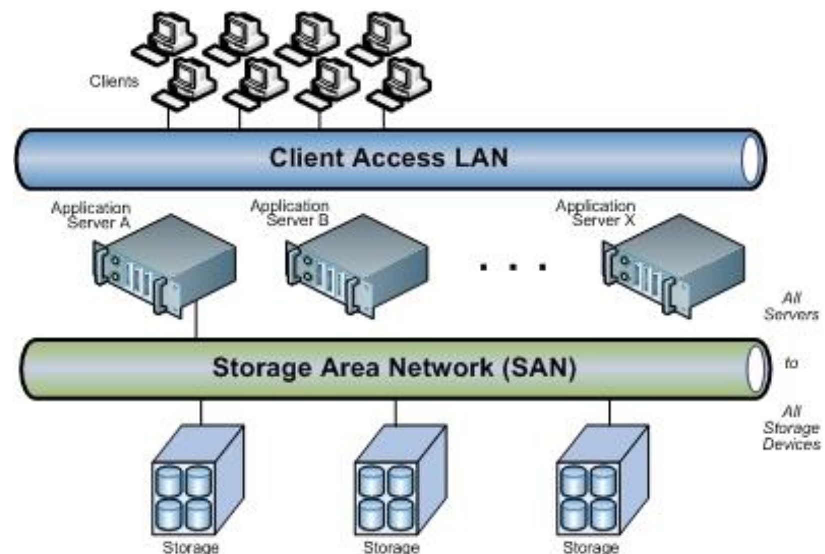
Multi-Processor Systems

- Advantages of multiprocessor system
  - Increased throughput
    - N processors → speed up by N times (ideally)
      - But, usually some overhead
    - Performance improvement is related to parallelism
  - Economy of scale (compared with clustered systems)
    - Memory, peripherals can be shared

Clustered Systems

- Multiple systems working together
  - Loosely coupled system
  - Usually sharing storage via a storage-area network (SAN)
    - SAN: a dedicated network that provides access to consolidated, block level data storage.
      - Composed of hosts, switches storage elements
  - Some have distributed lock manager (DLM) to avoid conflicting operations





Purposes of Clustered Systems

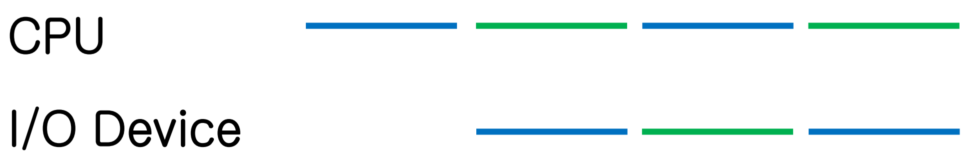
- Provides **high-availability** service which survives failures
  - Asymmetric clustering has one machine in hot-standby mode
  - Symmetric clustering has multiple nodes running applications, monitoring each other
  - Increased reliability
    - Graceful degradation: ability to continue service proportional to the level of surviving H/W
    - Fault tolerant: the property of a system that continues operating properly in the event of failure of some of its parts.
- Some clusters are for **high-performance** computing (HPC)
  - Applications must be written to use parallelization

Agenda

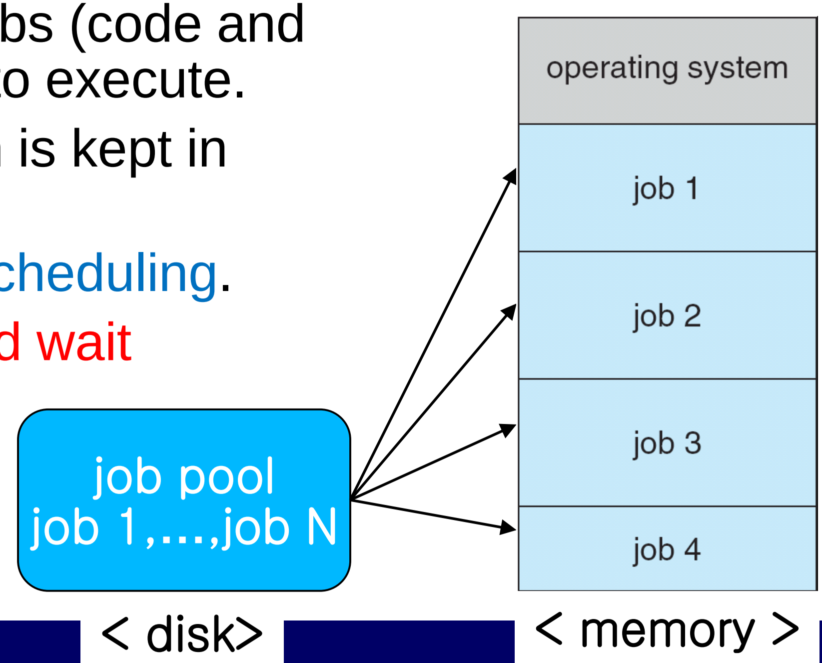
- Definitions of operating system
- Computer system organization and operation
- Computer system architecture
- **Operating system structure and operation**
- Core components of OS
- Computing environments

Operating System Structure

- Motivation: single user cannot keep CPU and I/O devices busy at all times

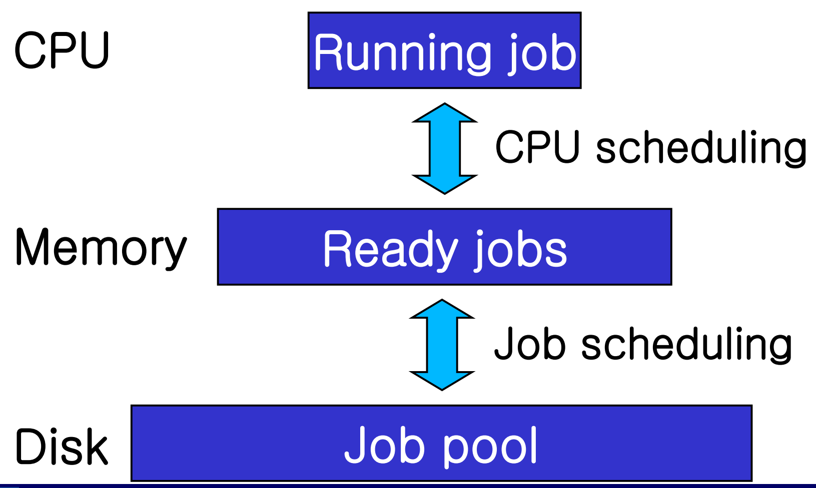


- **Multiprogramming**
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute.
  - A subset of total jobs in system is kept in memory
  - One job selected and run via scheduling.
  - OS switches jobs if a job should wait



Operating System Structure

- **Timesharing (multitasking)**: logical extension of multiporgramming in which CPU switches jobs so frequently that users can interact with each job while it is running, creating interactive computing.
  - **Response time** should be < 1 second
  - Each user has at least one program executing(process) in memory
  - **CPU scheduler** selects a job that is ready to run



Virtual Memory

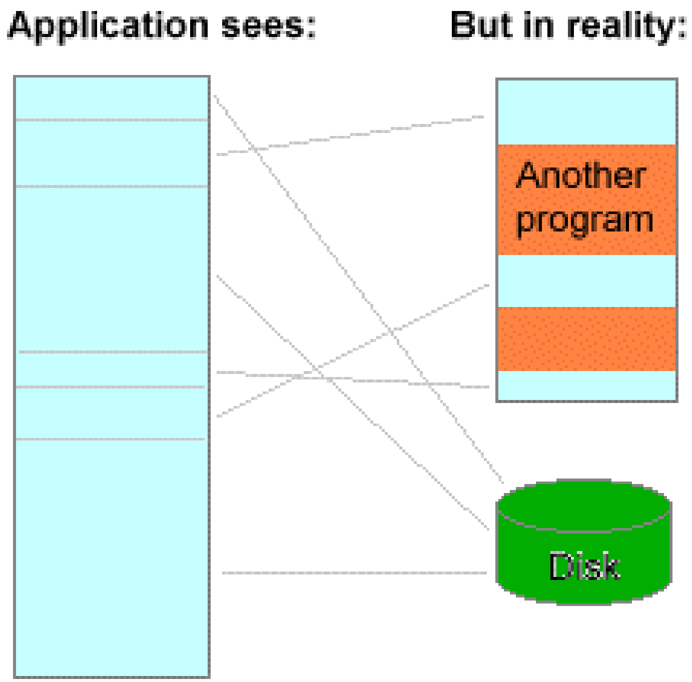
- Main memory is not enough to accommodate all processes on multiprogramming/multitasking system.

➔ Remedies

- **Swapping** moves memory contents in and out to run.
- **Virtual memory** allows execution of processes not completely in memory.

Virtual Memory

- Virtual memory is a computer system technique which gives an application program the impression that it has contiguous working memory, while in fact it is physically fragmented and may even overflow on to disk storage.
  - "Using disk space to extend physical memory size" + alpha



Operating System Operations

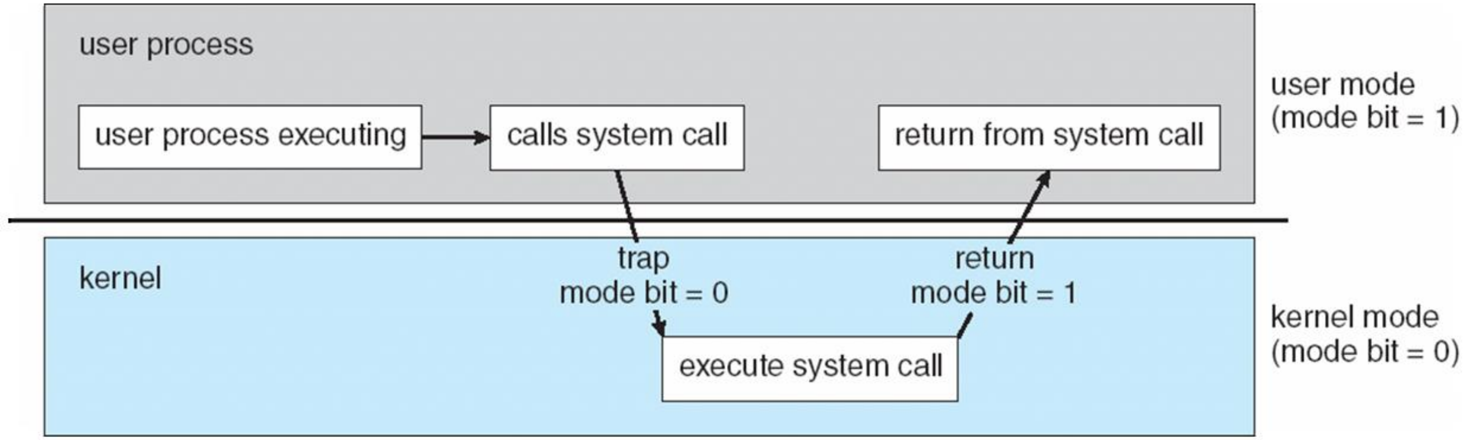
- Modern OS's are interrupt-driven programs
  - Events are signaled by interrupts, which are handled by interrupt handlers.
  - **H/W and S/W resources are shared.**

  ➔ **Problem: An error from a process can corrupt whole system**

- Essential requirement for multi-user OS: Error of a program should not affect to other program
  - Dangerous instructions ➔ **dual mode operation**
    - See 80x86 instruction set
  - Maintaining control over CPU
    - Preventing long execution of user process ➔ **timer**

Operating System Operations

- Dual-mode operation (requires H/W support)
  - User mode
    - User defined code (application)
    - **Privileged instructions, which can cause harm to other system, are prohibited**
      - I/O instruction, timer instruction, …
      - Privileged instruction can be invoked only through OS system call.
  - Kernel mode (supervisor mode, system mode, privileged mode)
    - OS code
    - Privileged instructions are permitted



Operating System Operations

- Advantage of dual-mode operation
  - Errors violating mode can be detected by H/W and handled by OS
    - Abnormal termination without system halt
    - Error message
    - Memory is dumped

Timer

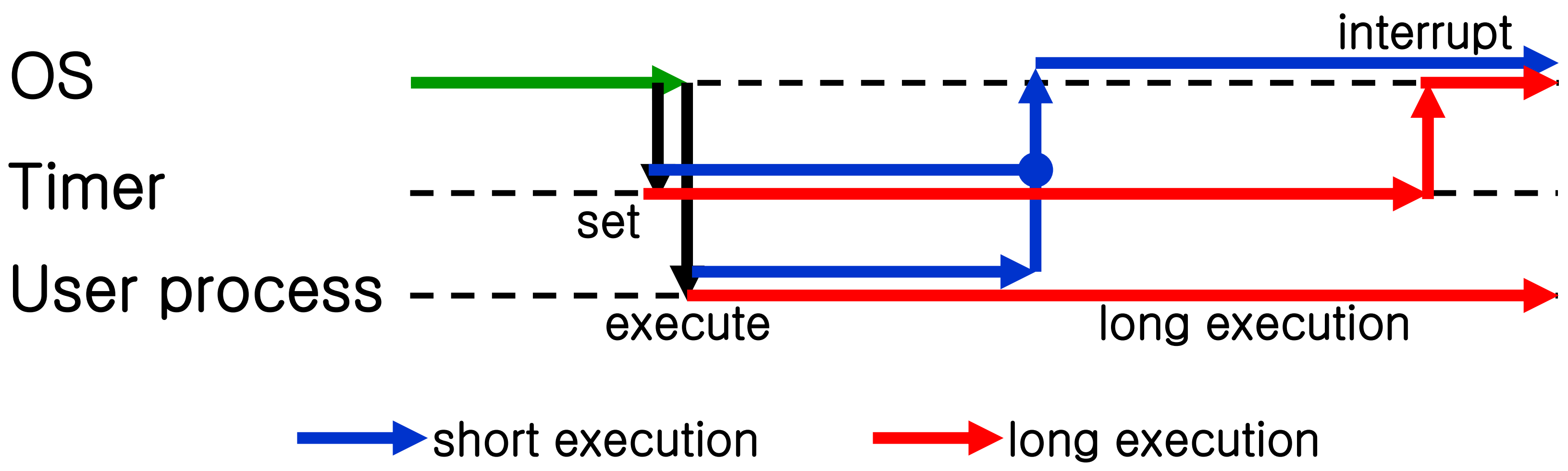
- We must ensure that the operating system maintains control over the CPU.
  - We cannot allow a user program to get stuck in an infinite loop or to fail to call system services and never return control to the operating system.

- Timer
  - Can be set to interrupt the computer after a specified period.
  - The period may be fixed (e.g. 1/60 second) or variable (e.g. 1~1000 msec.).
  - A **variable timer** is generally implemented by a fixed-rate clock and a counter.
    - The operating system sets the counter.
    - Every time the clock ticks, the counter is decremented.
    - When the counter reaches 0, an interrupt occurs.

Timer

- Protecting from long execution of a program
  - Before OS gives control to user process, timer is set to interrupt after pre-defined duration
  - If timer interrupt occurs, OS can take control to handle each case appropriately

  * Note: Time counter can be modified by only privileged instruction



Multi-Mode Operation

- Intel processors have four separate **protection rings**,
  - Ring 0: kernel mode
  - Ring 1 and 2: could be used for various operating-system services, but in practice, rarely used.
  - Ring 3: user mode.
- ARMv8 systems have seven modes.
- CPUs that support virtualization frequently have a separate mode to indicate when the **virtual machine manager** (**VMM**) is in control of the system.
  - VMM has more privileges than user processes but fewer than the kernel.

Agenda

- Definitions of operating system
- Computer system organization and operation
- Computer system architecture
- Operating system structure and operation
- **<u>Core components of OS</u>**
- Computing environments

Core Components of OS

- Process management
- Memory management
- Storage management
- Protection and security

Process Management

- **Process**: program in execution (active entity)
  - Job, time-shared program
  - Active program with **required resources**

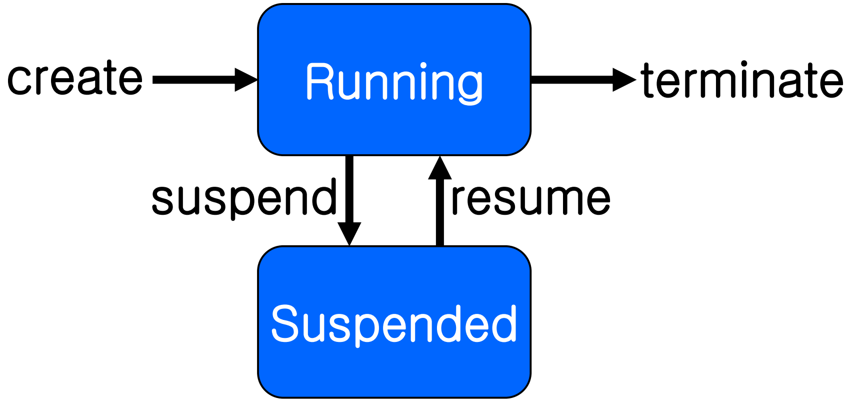
    Ex) word-processor running on PC, …
  - Single threaded process has **a program counter**.
    - Multiple instances of a program: separate processes
  - Unit of work
    - Operating system processes, user processes

Process Management

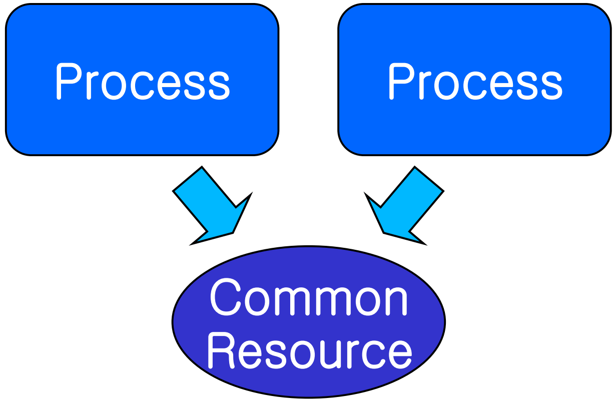
- Terminologies similar to process
  - **Thread**: a way for a program to split itself into two or more simultaneously running task
    - Basic unit of CPU utilization
    - Smaller unit than process
    - Each thread has its own ID, program counter, register set, stack,…
    - Major resources are shared
  - **Task**: an execution path through address space
    - A set of program instructions that is loaded in memory
    - In some context, such as Linux, task means process or both process and thread

Process Management

- Process management by OS
  - Creating/deleting processes
  - Suspending/resuming process
  - Process synchronization



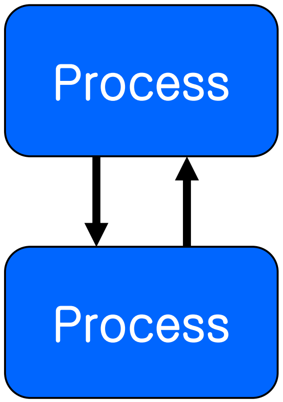
< process life cycle >



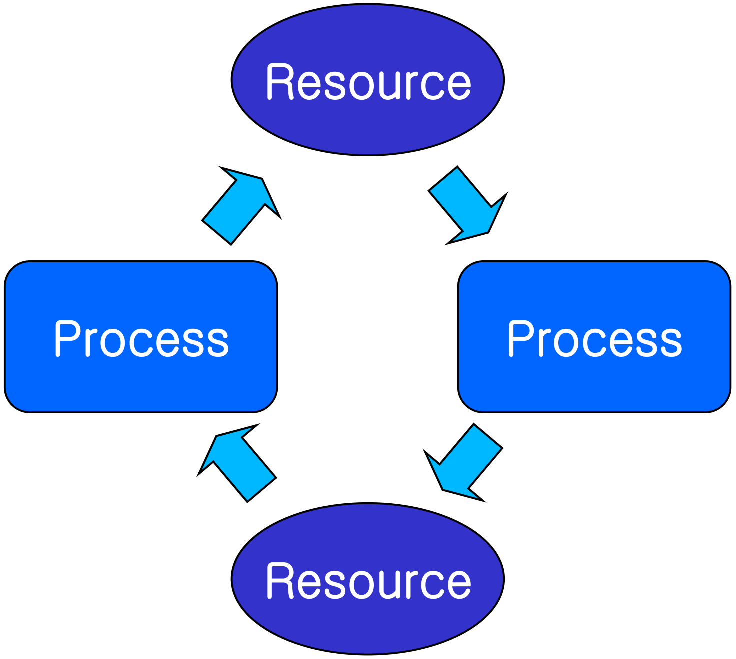
< process synchronization >

Process Management

- Process management by OS
  - Process communication
  - Deadlock handling



< process communication >



< dead lock >

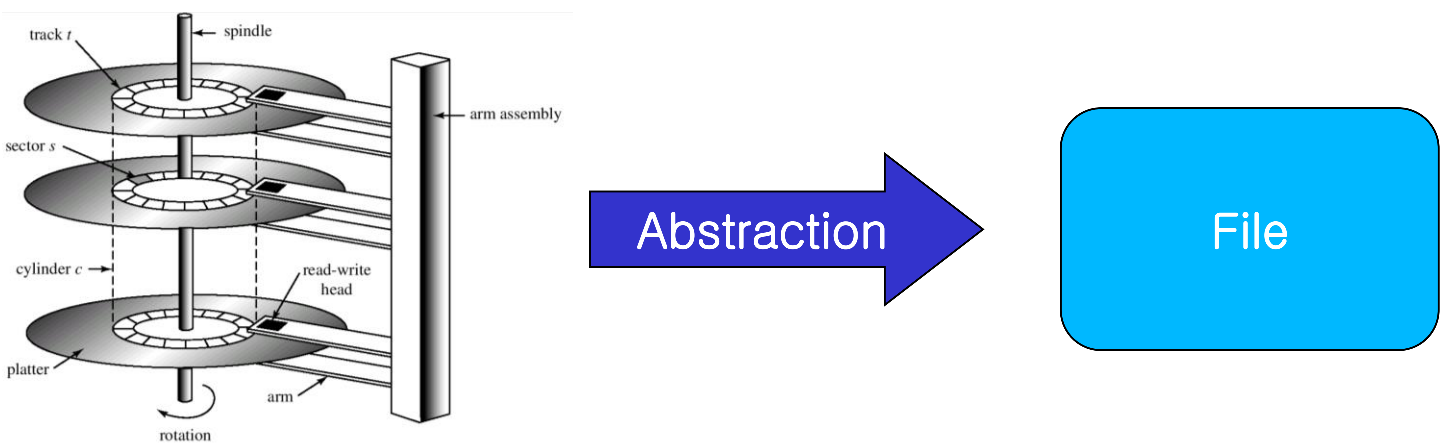
Memory Management

- Main memory is central to operation of modern system
  - The only large storage that CPU can address/access directly.
  - Repository of data shared by CPU and I/O devices.

- Memory management by OS
  - Keeping track of which part of memory is occupied by whom
  - Allocating/deallocating memory space
  - Deciding which process and data to move in/out of memory

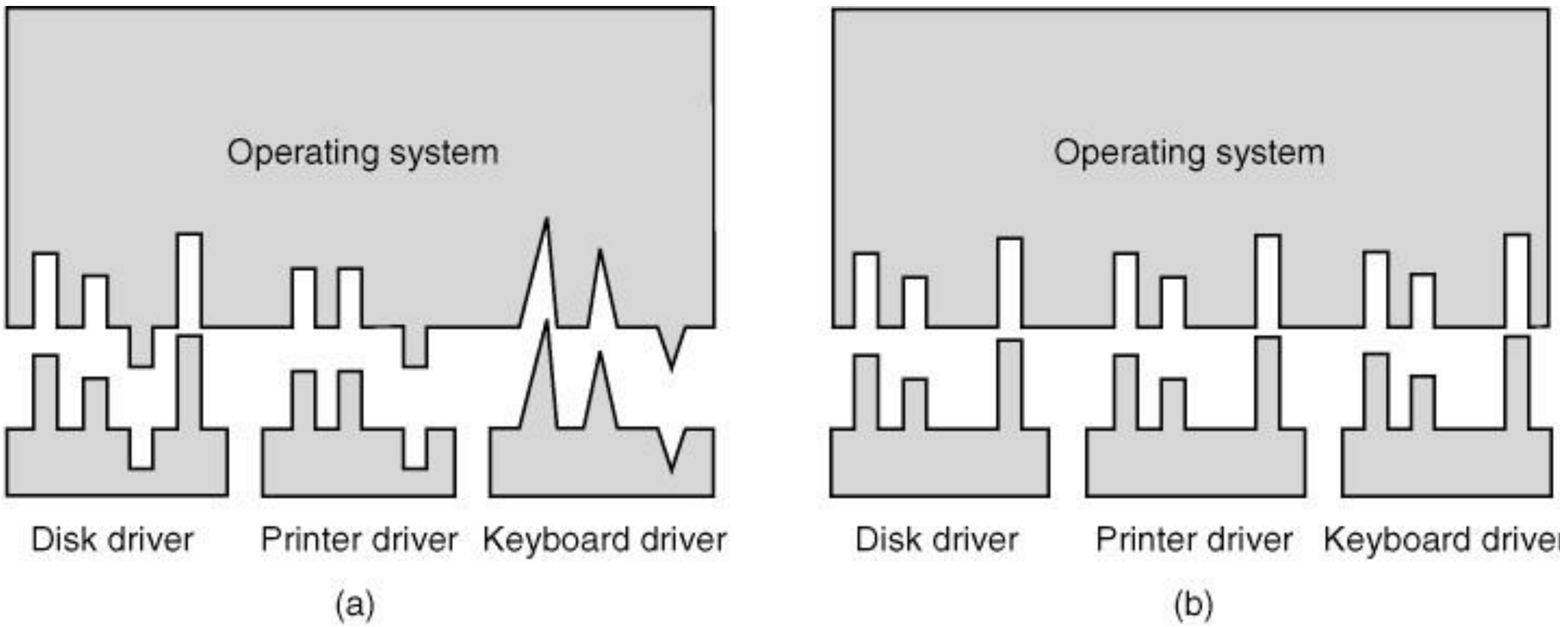
Storage Management

- Abstraction of physical storage into logical **file**
  - Physical storage: magnetic disk, optical disk, magnetic tape, …
    - Various in speed, capacity, transfer rate, access method
  - File: logical storage unit abstracted by OS



Device Driver

- Device drivers provides uniform interfaces



Storage Management

- File-system management
  - Creating/deleting files and directories
  - Supporting primitives for manipulating files/directories
  - Mapping files into secondary storage
  - Backup files on stable storage media

Mass-Storage Management

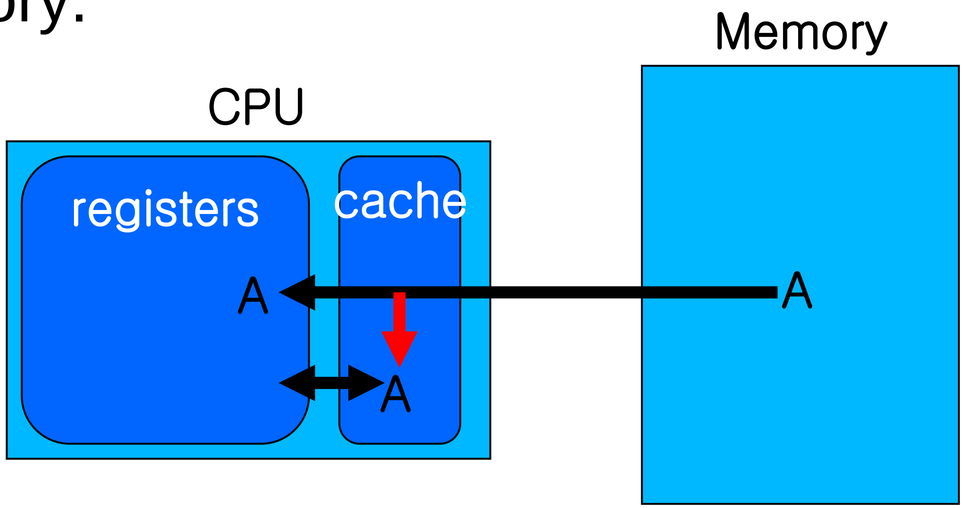
- The operating system implements the abstract concept of a file by managing mass-storage media.

- Mass-storage management
  - Free-space management
  - Storage allocation
  - Disk scheduling

Cashing

- Temporal copy of used information into **small and faster storage** for next access
  - Based on locality of references
- When a particular piece of information is needed,
  - First, check if it is in cache → *faster than original source*
  - If so, use it, otherwise, use information from original source

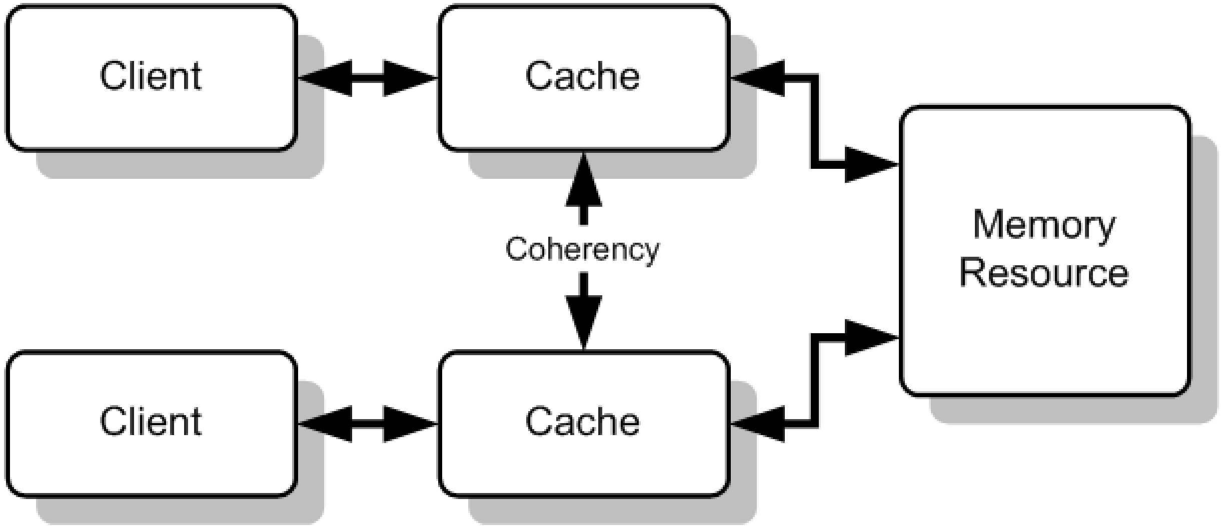
cf. Main memory can be viewed as a cache for secondary memory.



Cache Coherence

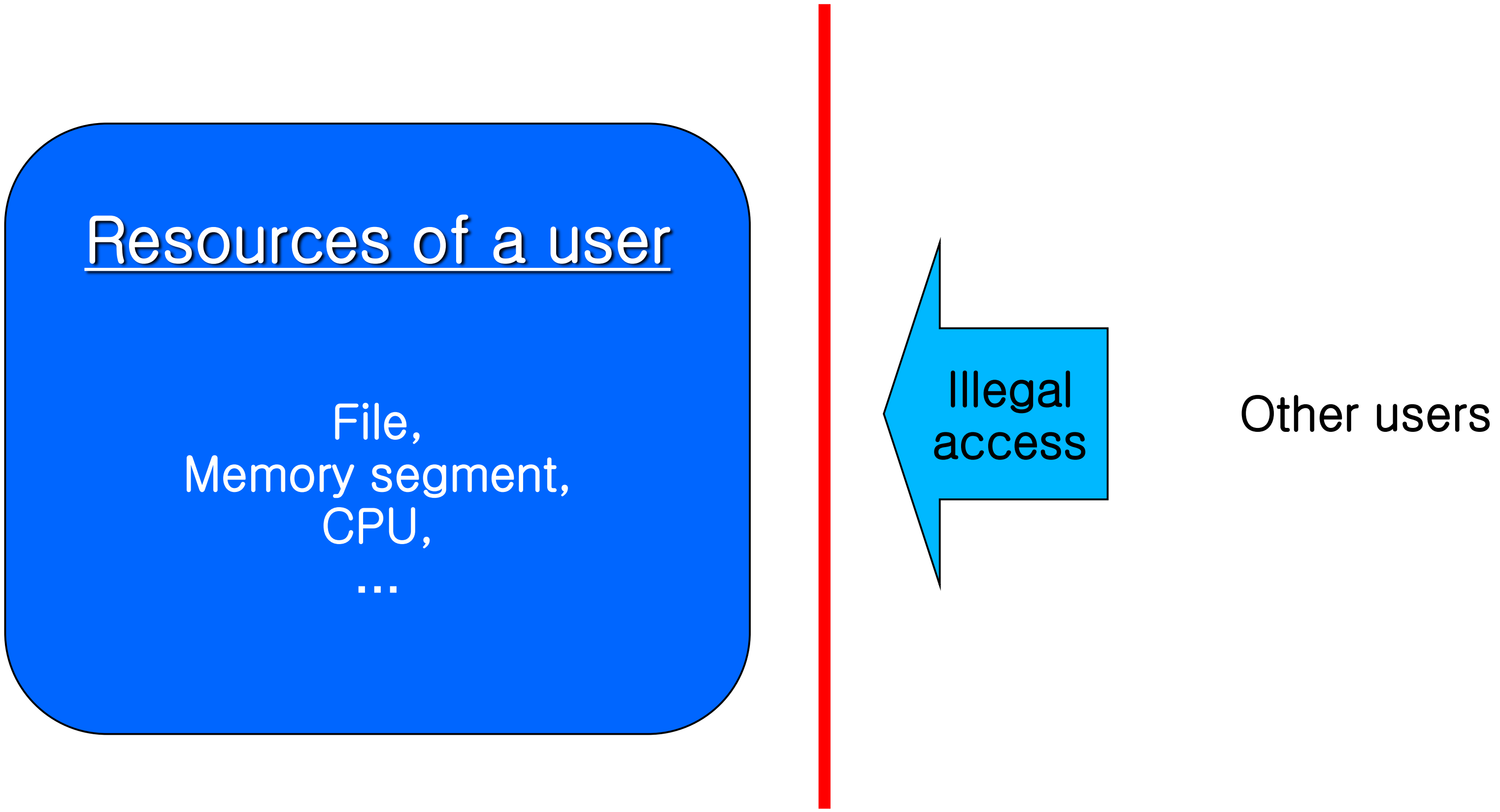
- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache

- Cache coherence: integrity of data stored in local caches of a shared resource.



Protection and Security

- Major issue in multi-user, multi-processing system
  - Authorization mechanism for file, memory segment, CPU, and other resources



Protection and Security

- **Protection**: any mechanism for controlling access of processes or users to resources
  - Unique User ID, group ID are assigned to all processes and resources
- Protection is not sufficient as authentication can be stolen
- **Security**: defense against external and internal attack to acquire authentication illegally
  - Security issue is very important in recent environment

Agenda

- Definitions of operating system
- Computer system organization and operation
- Computer system architecture
- Operating system structure and operation
- Core components of OS
- **Computing environments**

Computing Environments – Traditional

- Stand-alone general purpose machines
- But blurred as most systems interconnect with others (i.e., the Internet)
- **Portals** provide web access to internal systems
- **Network computers** (**thin clients**) are like Web terminals
- Mobile computers interconnect via **wireless networks**
- Networking becoming ubiquitous – even home systems use **firewalls** to protect home computers from Internet attacks

Computing Environments – Mobile

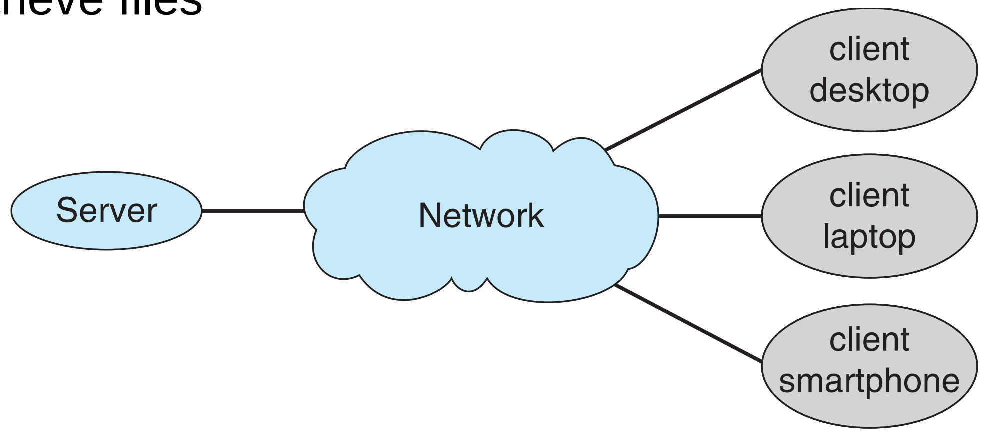
- Handheld smartphones, tablets, etc
- What is the functional difference between them and a “traditional” laptop?
- Extra feature – more OS features (GPS, gyroscope)
- Allows new types of apps like ***augmented reality***
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders are **Apple iOS** and **Google Android**

Computing Environments – Distributed

- Distributed computiing
  - Collection of separate, possibly heterogeneous, systems networked together
    - **Network** is a communications path, **TCP/IP** most common
      - **Local Area Network** (**LAN**)
      - **Wide Area Network** (**WAN**)
      - **Metropolitan Area Network** (**MAN**)
      - **Personal Area Network** (**PAN**)
  - **Network Operating System** provides features between systems across network
    - Communication scheme allows systems to exchange messages
    - Illusion of a single system

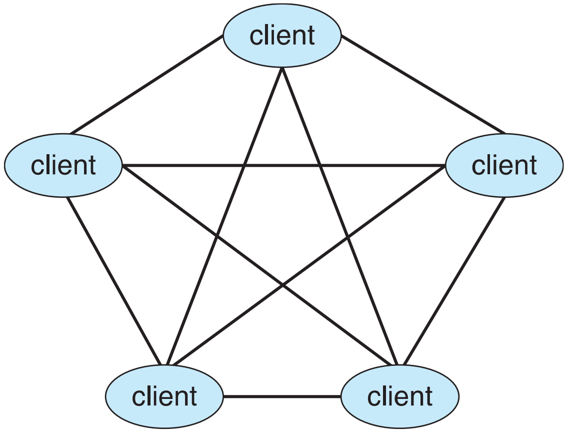
Computing Environments – Client-Server

- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now **servers**, responding to requests generated by **clients**
    - **Compute-server system** provides an interface to client to request services (i.e., database)
    - **File-server system** provides interface for clients to store and retrieve files



Computing Environments – Peer-to-Peer

- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - May each act as client, server or both
  - Node must join P2P network
    - Registers its service with central lookup service on network, or
    - Broadcast request for service and respond to requests for service via ***discovery protocol***
  - Examples include Napster and Gnutella, **Voice over IP** (**VoIP**) such as Skype



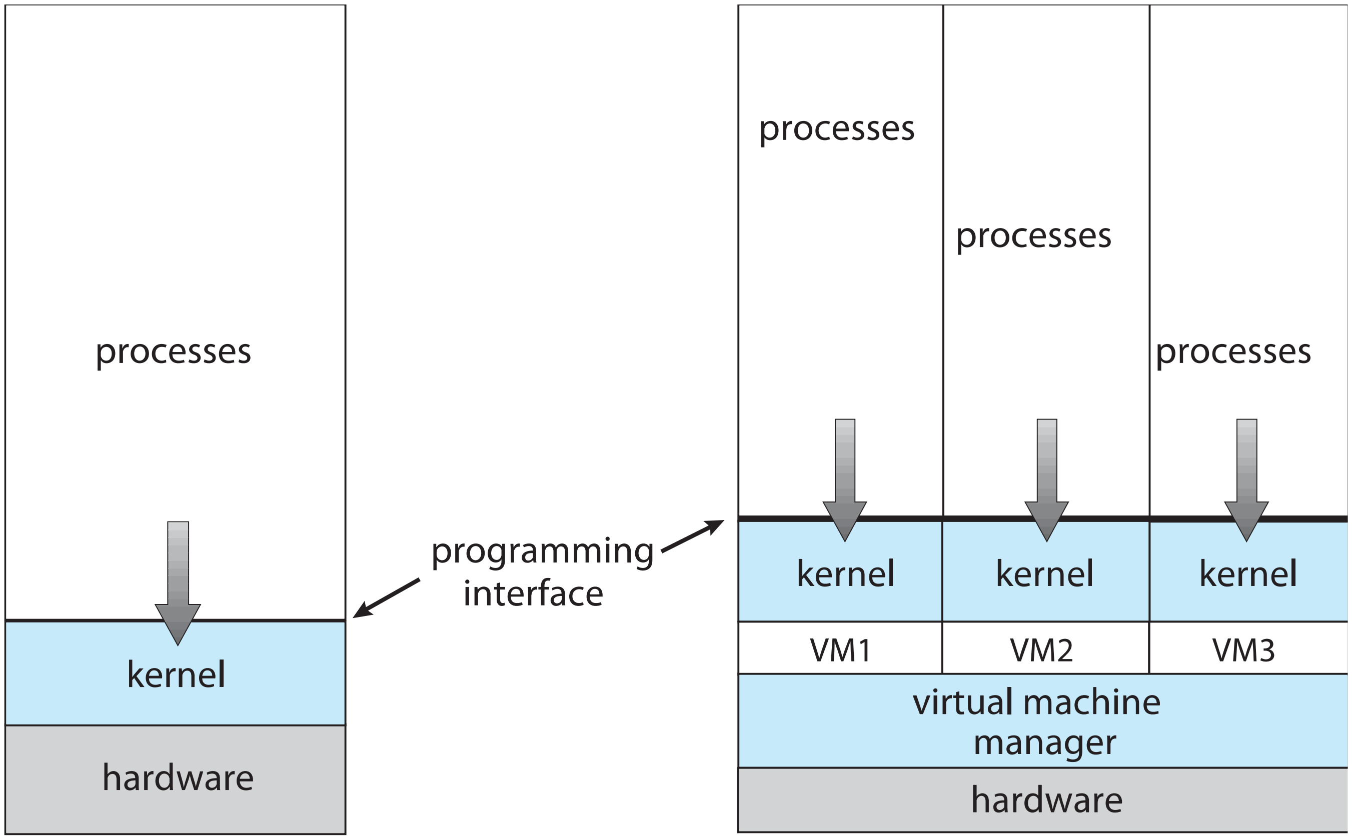
Computing Environments – Virtualization

- Allows operating systems to run applications within other OS
  - Vast and growing industry
- **Emulation** used when source CPU type different from target type (i.e. PowerPC to Intel x86)
  - Generally slowest method
  - When computer language not compiled to native code – **Interpretation**
- **Virtualization** – OS natively compiled for CPU, running **guest** OS also natively compiled
  - Consider VMware running WinXP guests, each running applications, all on native WinXP **host** OS
  - **VMM** (virtual machine manager) provides virtualization services

Computing Environments – Virtualization

- Use cases involve laptops and desktops running multiple OSes for exploration or compatibility
  - Apple laptop running Mac OS X host, Windows as a guest
  - Developing apps for multiple OSes without having multiple systems
  - QA testing applications without having multiple systems
  - Executing and managing compute environments within data centers
- VMM can run natively, in which case they are also the host
  - There is no general purpose host then (VMware ESX and Citrix XenServer)

Computing Environments – Virtualization



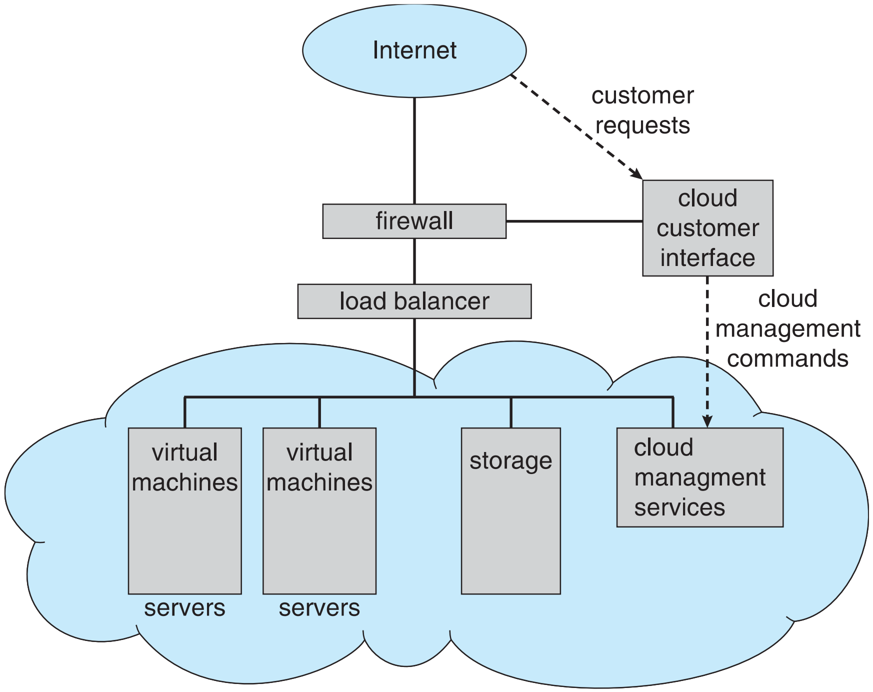
(a) (b)

Computing Environments – Cloud Computing

- Delivers computing, storage, even apps as a service across a network
- Logical extension of virtualization because it uses virtualization as the base for it functionality.
  - Amazon **EC2** has thousands of servers, millions of virtual machines, petabytes of storage available across the Internet, pay based on usage
- Many types
  - **Public cloud** – available via Internet to anyone willing to pay
  - **Private cloud** – run by a company for the company's own use
  - **Hybrid cloud** – includes both public and private cloud components
  - Software as a Service (**SaaS**) – one or more applications available via the Internet (i.e., word processor)
  - Platform as a Service (**PaaS**) – software stack ready for application use via the Internet (i.e., a database server)
  - Infrastructure as a Service (**IaaS**) – servers or storage available over Internet (i.e., storage available for backup use)

Computing Environments – Cloud Computing

- Cloud computing environments composed of traditional OSes, plus VMMs, plus cloud management tools
  - Internet connectivity requires security like firewalls
  - Load balancers spread traffic across multiple applications



Computing Environments – Real-Time Embedded Systems

- Real-time embedded systems most prevalent form of computers
  - Vary considerable, special purpose, limited purpose OS, **real-time OS**
  - Use expanding
- Many other special computing environments as well
  - Some have OSes, some perform tasks without an OS
- Real-time OS has well-defined fixed time constraints
  - Processing ***must*** be done within constraint
  - Correct operation only if constraints met

Open-Source Operating Systems

- Operating systems made available in source-code format rather than just binary **closed-source**
- Counter to the **copy protection** and **Digital Rights Management (DRM)** movement
- Started by **Free Software Foundation (FSF)**, which has "copyleft" **GNU Public License (GPL)**
- Examples include **GNU/Linux** and **BSD UNIX** (including core of **Mac OS X**), and many more
- Can use VMM like VMware Player (Free on Windows), Virtualbox (open source and free on many platforms - http://www.virtualbox.com)
  - Use to run guest operating systems for exploration