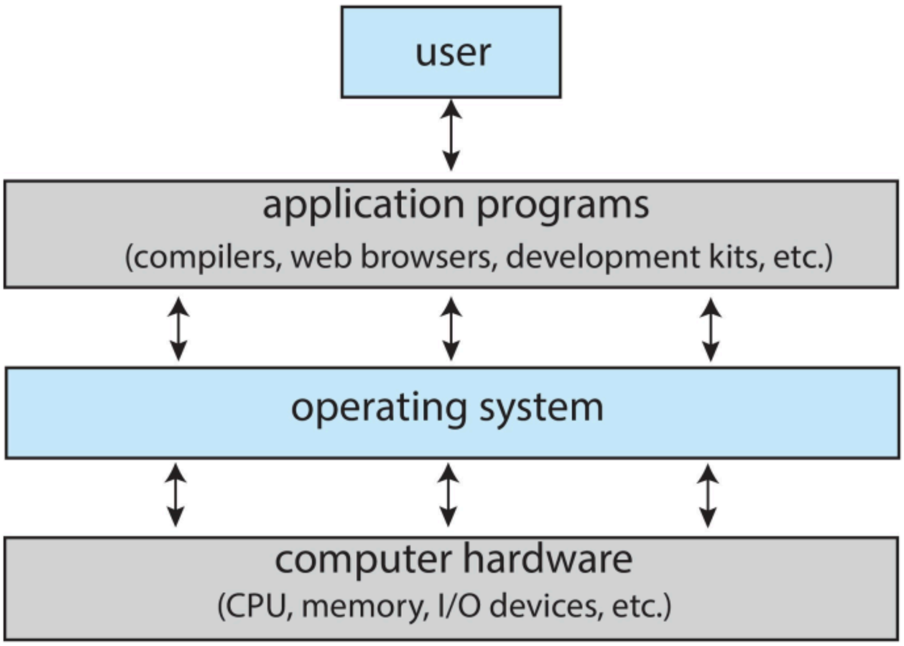
In HIS name

OS

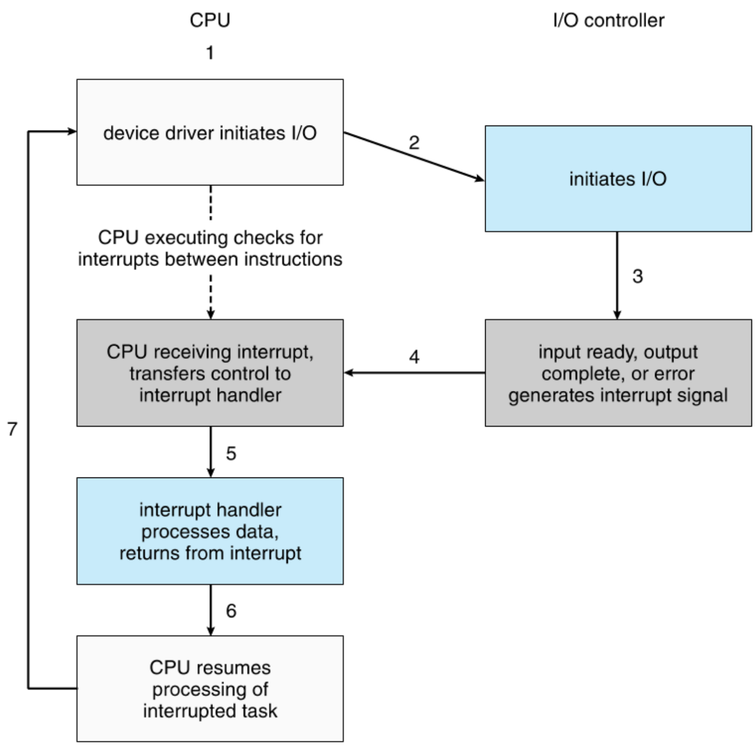
Chapter 1

- Operating system:
    - CPU, memory, I/O devices
    - Controls and coordinates use of hardware among various
    - One purpose of OS is to hide peculiarities of hardware devices from the user

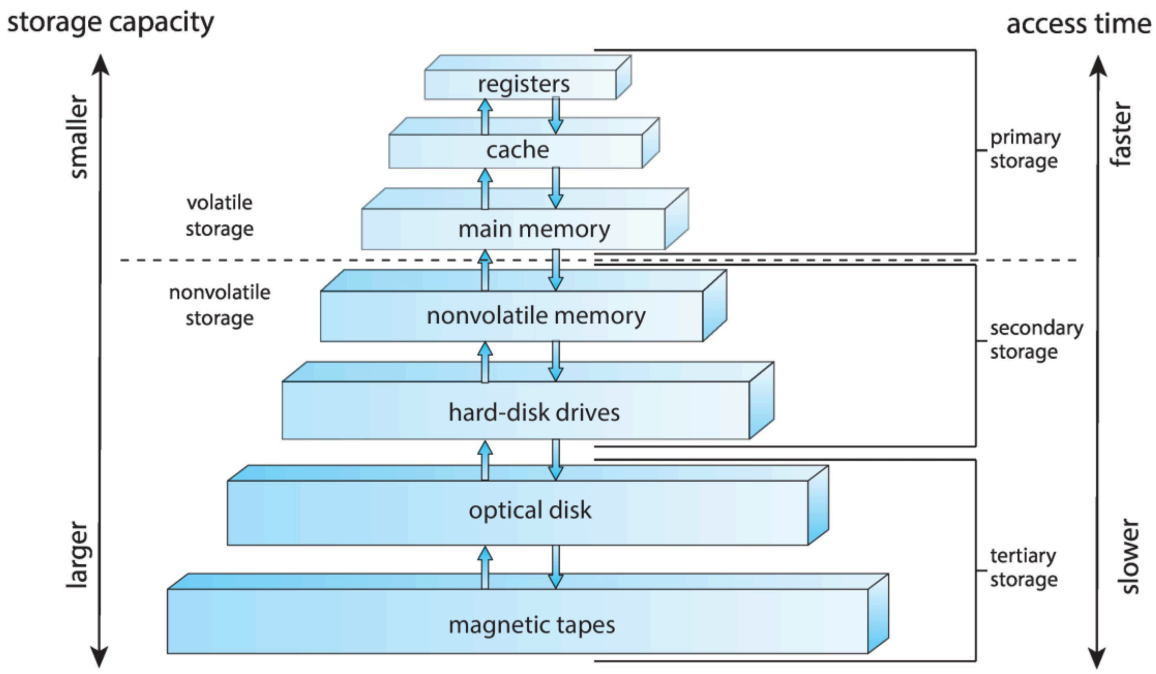
- Operating system is a resource allocator and control program making efficient use of HW and managing execution of user programs



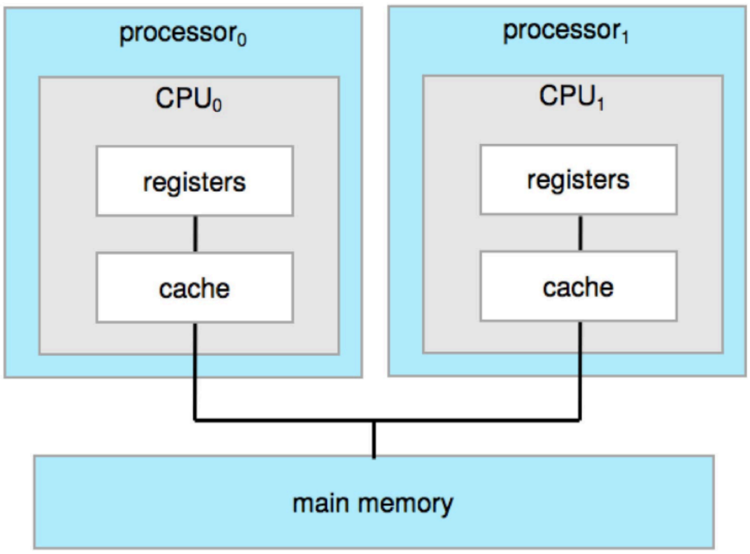
- "The one program running at all times on the computer" is the kernel, part of the operating system
- Others are:
  - system program (ships with the operating system, but not part of the kernel) , or
  - an application program, all programs not associated with the operating system
- Shares memory : One or more CPUs, device controllers connect through common bus providing access to shared memory
- Interrupt:
  - Device controller informs CPU that it has finished its operation by causing an interrupt
  - 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

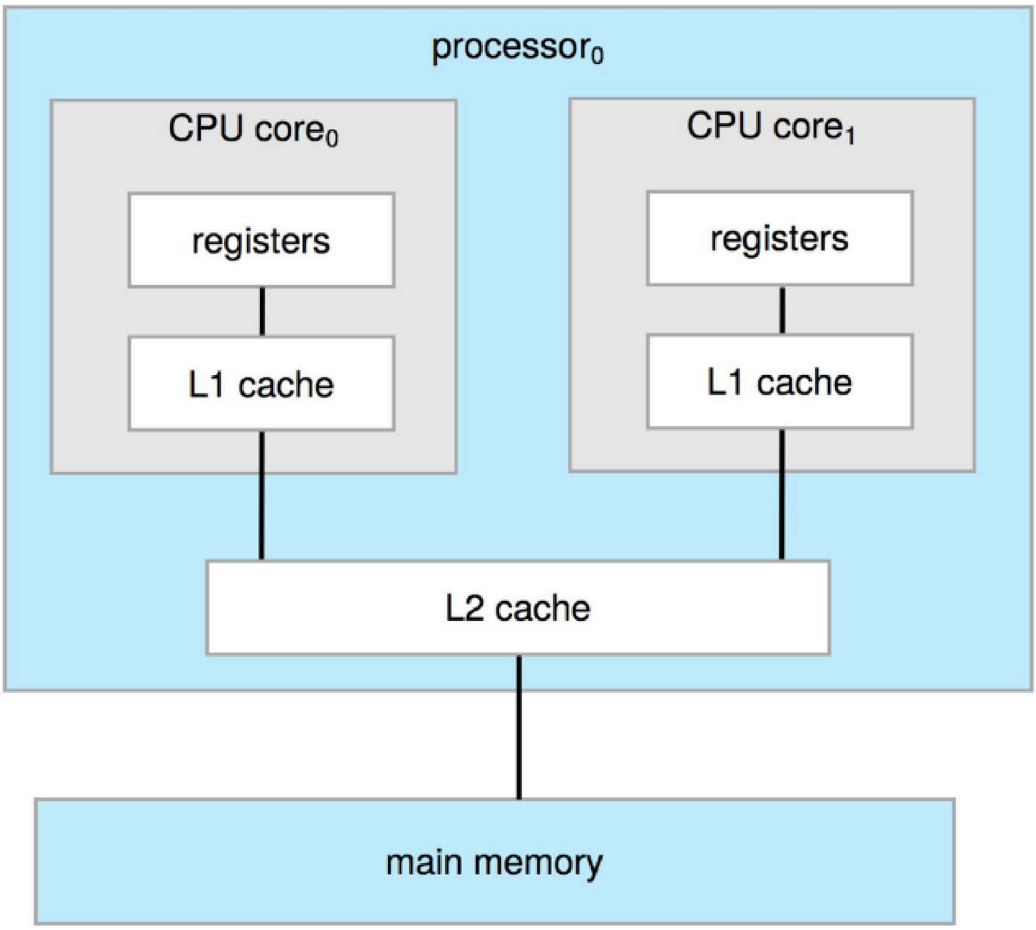


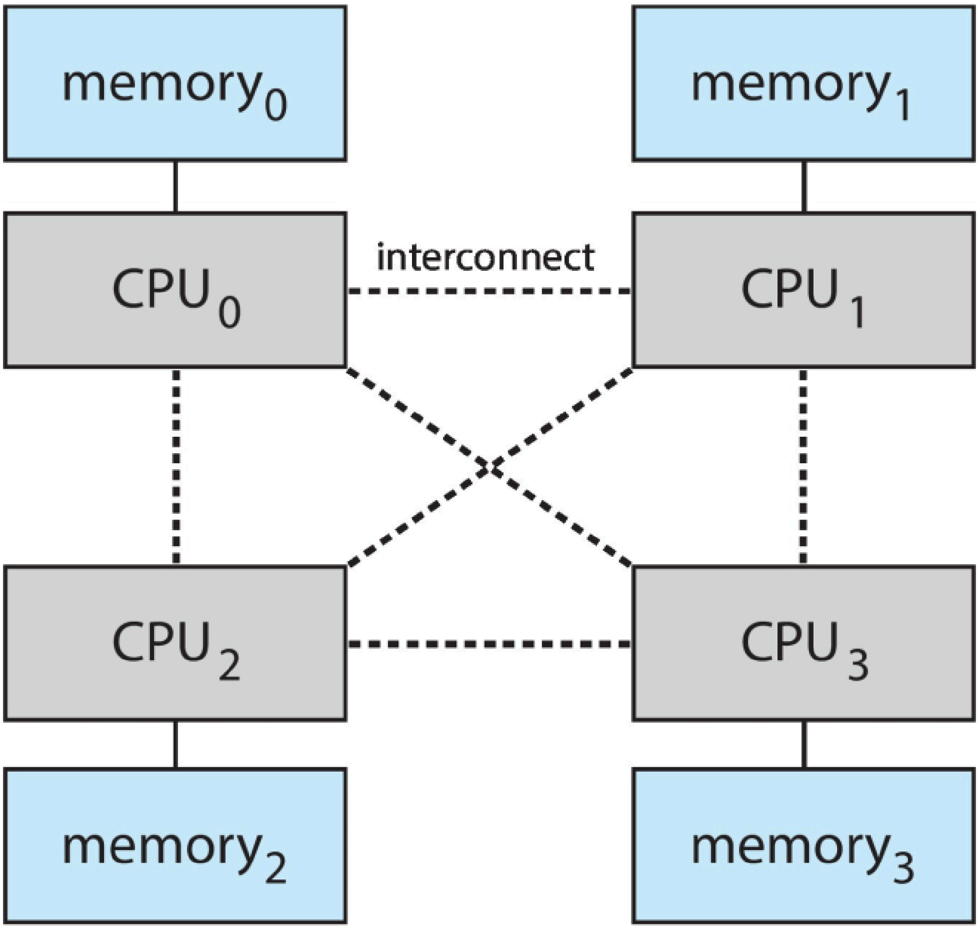
- ➔ Memory:
    - ◆ Random access
    - ◆ Typically random-access memory in the form of Dynamic Random-access Memory (DRAM)
    - ◆ Typically volatile
- ➔ Secondary storage : extension of main memory that provides large nonvolatile storage capacity
- ➔ Hard Disk Drives (HDD) : rigid metal covered with 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
- ➔ 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
- ➔ Storage systems organized in hierarchy:
    - ◆ Speed Volatility Cost



- Direct memory access:
  - 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 one interrupt is generated per block, rather than the one interrupt per byte
- Multiprocess
  - Increased throughput
  - Economy of scale
  - Increased reliability – fault tolerance
  - Types:
    - Asymmetric Multiprocessing – each processor is assigned a specific task.
    - Symmetric Multiprocessing – each processor performs all tasks


processor$_0$
CPU$_0$
registers
cache
processor$_1$
CPU$_1$
registers
cache
main memory

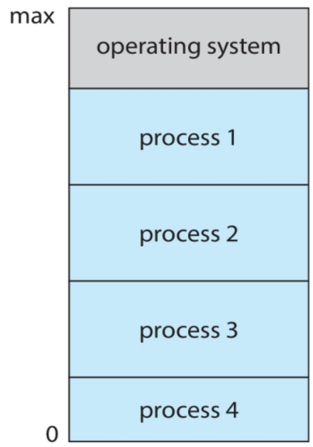

processor$_0$
CPU core$_0$
registers
L1 cache
CPU core$_1$
registers
L1 cache
L2 cache
main memory



- 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
- Bootstrap program : simple code to initialize the system, load the kernel
  - Kernel loads

- Starts system daemons (services provided outside of the kernel)

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

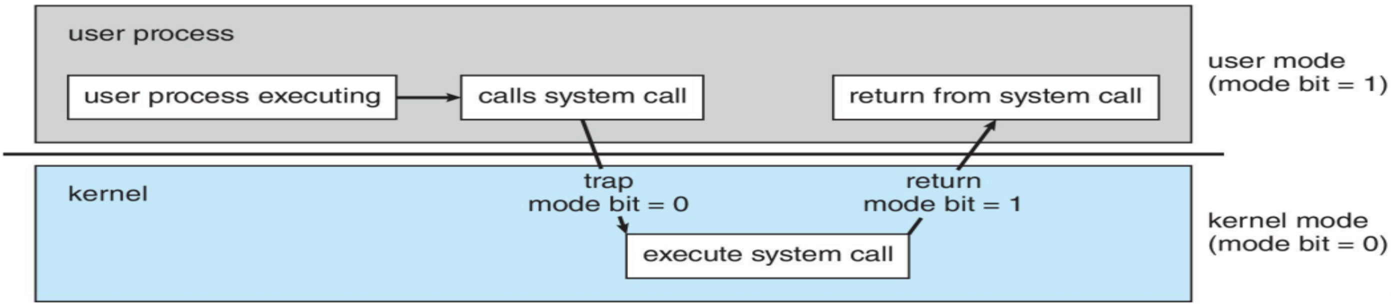


- Multiprogramming (Batch system) : needed for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - A subset of total jobs in system is kept in memory
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - One job selected and run via job 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
  - if several jobs ready to run at the same time -> CPU scheduling
  - Virtual memory allows execution of processes not completely in memory

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

- multi-mode operations:
  - i.e. virtual machine manager (VMM) mode for guest VMs



- Timer to prevent infinite loop:
  - Keep a counter that is decremented by the physical clock
  - Operating system set the counter (privileged instruction)
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time
- Multi-threaded process has one program counter per thread
- The operating system is responsible for the following activities in connection with process management:
  - Creating and deleting both user and system processes
  - Providing mechanisms for process synchronization
  - Suspending and resuming processes
  - Providing mechanisms for process communication
  - Providing mechanisms for deadlock handling
- 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
- Register -> compiler
- Cache -> hardware
- Memory, disk, tertiary storage -> OS
- Multiprocessor environment must provide cache coherency in

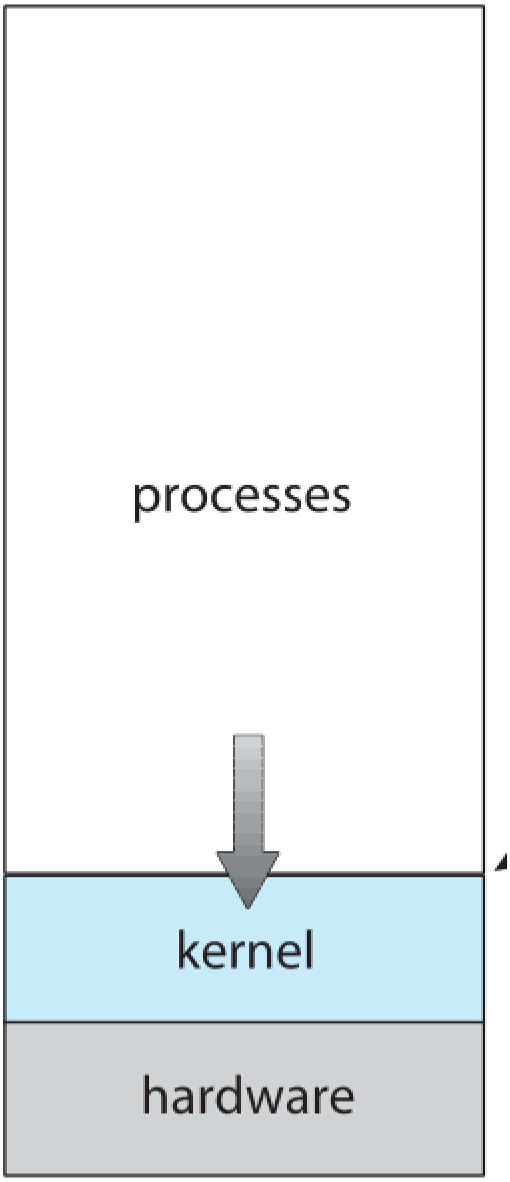
hardware such that all CPUs have the most recent value in their Cache

- 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 : any mechanism for controlling access of processes or users to resources defined by the OS
- Security : defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Virtualization
  - Allows operating systems to run applications within other OSes

Emulation : used when source CPU type different from target

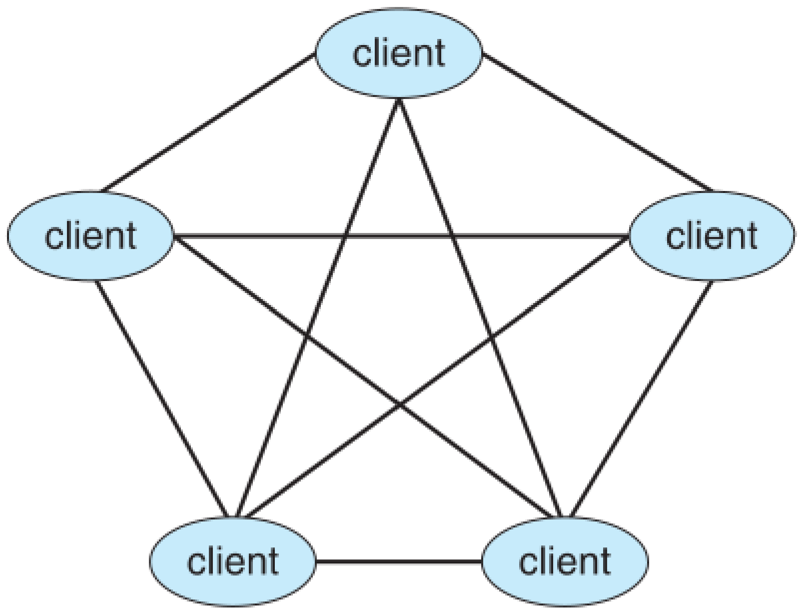
- type (i.e. PowerPC to Intel x86)
  - Generally slowest method
  - When computer language not compiled to native code – Interpretation
- Virtualization : OS natively compiled for CPU, running guest OSes also natively compiled
  - Consider VMware running WinXP guests, each running applications, all on native WinXP host OS
  - VMM (virtual machine Manager) provides virtualization services
- Benefits of Virtualizations:
  - Developing apps for multiple OSes without having multiple systems

- QA testing applications without having multiple systems Executing and managing compute environments within data centers



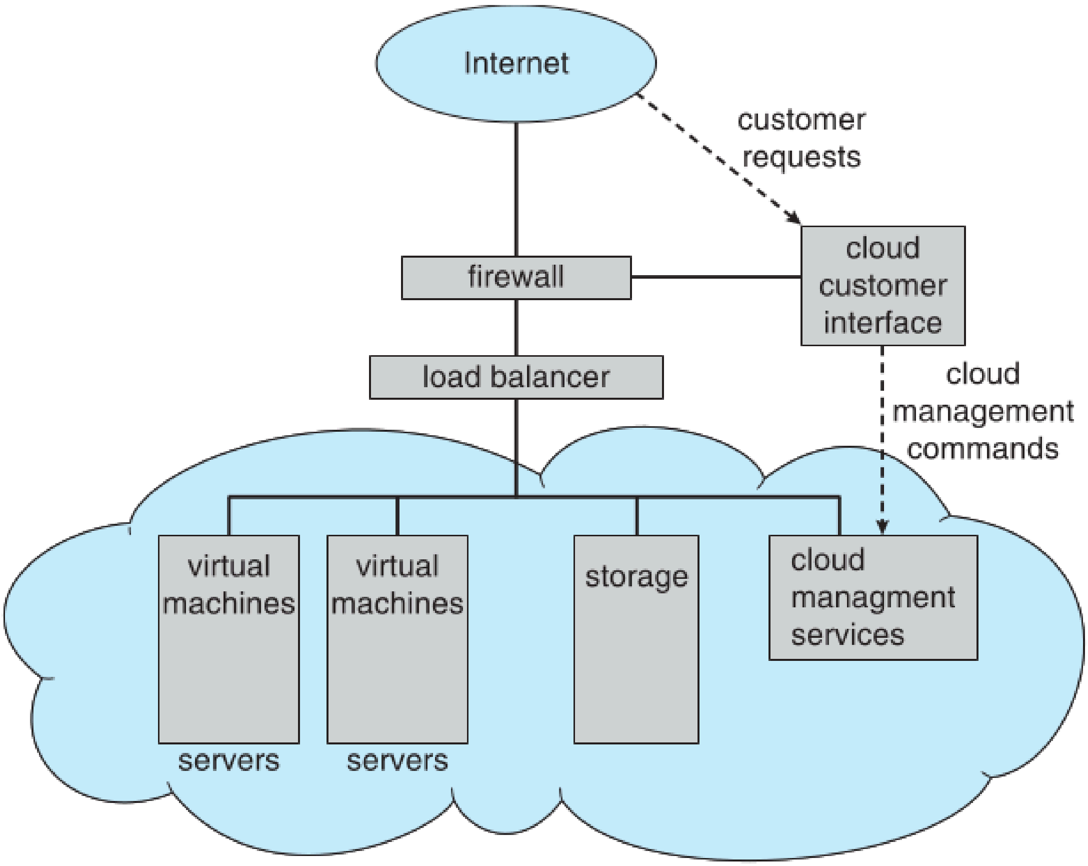
- Distributed computing
    - Collection of separate, possibly heterogeneous, systems networked together
- Compute-server system “
    - provides an interface to client to request services (i.e., database)
- File-server system :
    - provides interface for clients to store and retrieve files

- 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
  - 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



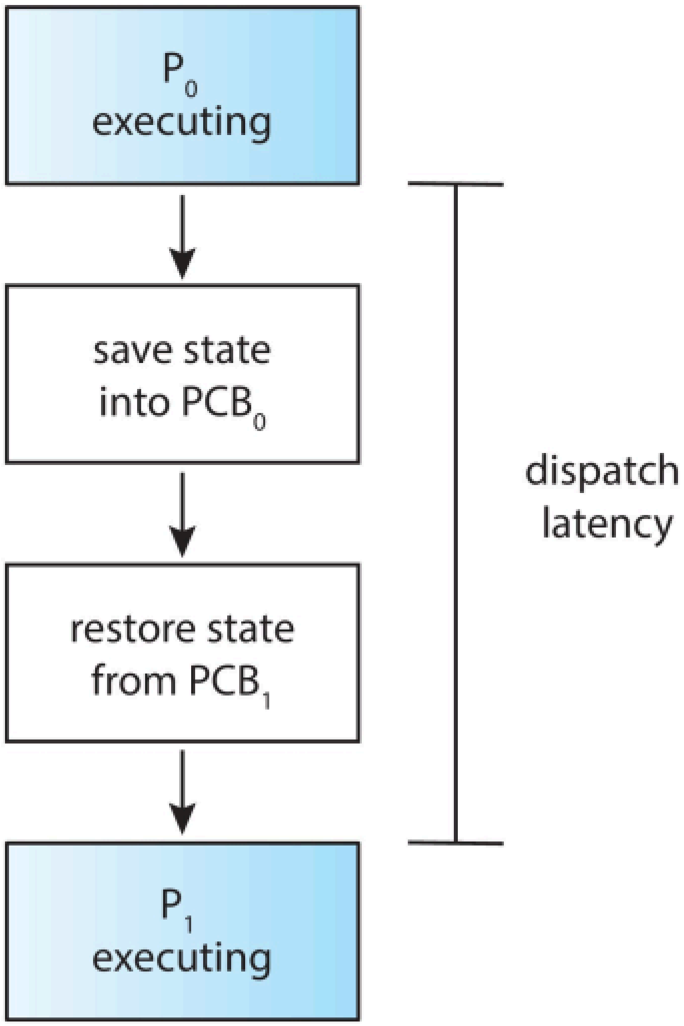
- Delivers computing, storage, even apps as a service across a network
- Logical extension of virtualization because it uses virtualization as the base for its functionality.
  - Public cloud : available via Internet to anyone willing to pay
  - Hybrid cloud – includes both public and private cloud components
  - Private cloud – run by a company for the company's own use
- 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)
➔ Load balancers spread traffic across multiple applications



Q5

➔ Dispatcher module:
  - switching context
  - switching to user mode
➔ Dispatch latency : time it takes for the dispatcher to stop one process and start another running

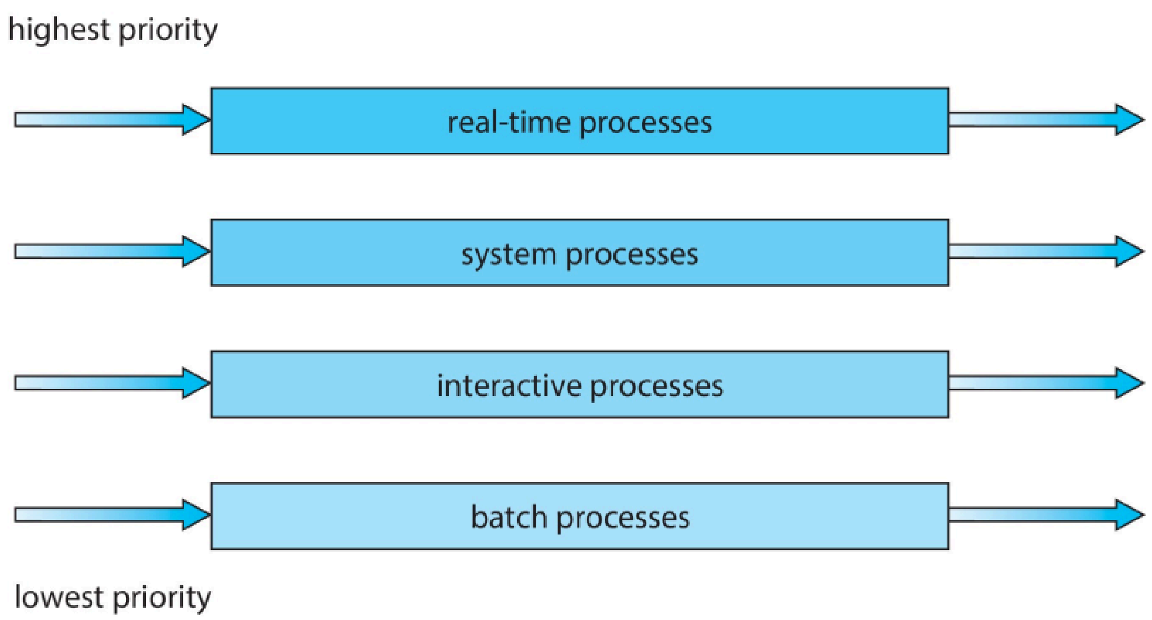


- ➔ CPU utilization (Max)– keep the CPU as busy as possible
- ➔ Throughput (Max)– # of processes that complete their execution per time unit
- ➔ Turnaround time (Min)– amount of time to execute a particular process
- ➔ Waiting time (Min)– amount of time a process has been waiting in the ready queue
- ➔ Response time (Min)– amount of time it takes from when request was submitted until the first response is produced, not output (for time-sharing environment)

- FCFS = FIFO
- Convoy effect - short process behind long process
- Consider one CPU-bound and many I/O-bound processes

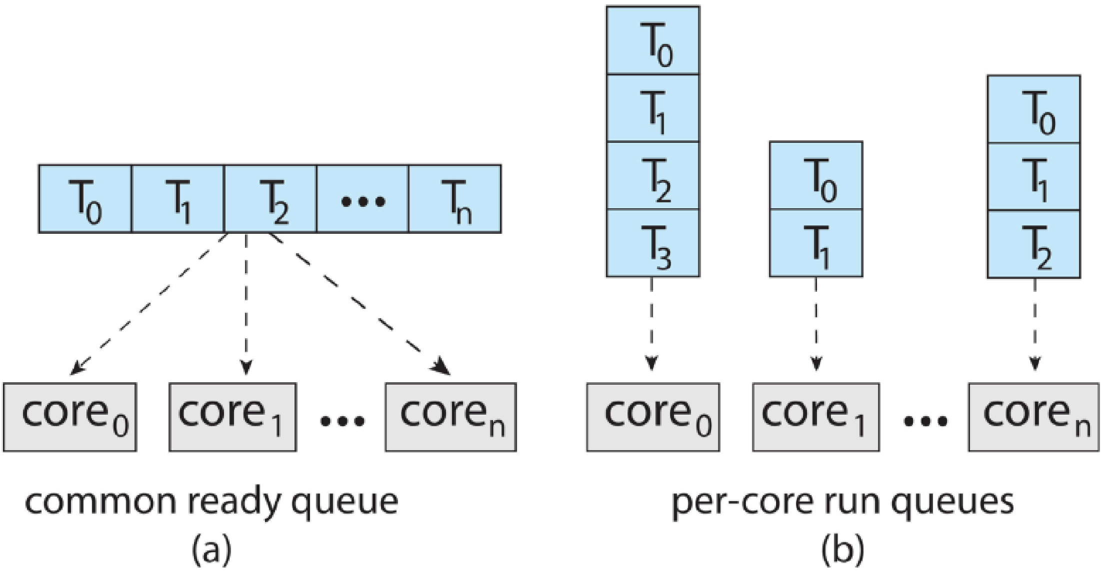
- SJF (Shortest job first)is optimal – gives minimum average waiting time
- The difficulty is knowing the length of the next CPU request
- t_n = actual length of n th CPU burst
- 2. τ_{n+1} = predicted value for the next CPU burst
- 3. α , $0 \le \alpha \le 1$
- 4. Define : $\tau_{n=1} = \alpha t_n + (1 - \alpha)\tau_n$.
- Commonly, α set to ½
- Preemptive version called shortest-remaining-time-first
- Shortest-remaining-time-first = preemptive sjf (on arrival of processes)
- Round Robin
  - q large ⇒ FIFO
  - q small ⇒ q must be large with respect to context switch,otherwise overhead is too high
  - Typically, higher average waiting time
  - Typically, higher average turnaround than SJF
  - q should be large compared to context switch time
  - q usually 10ms to 100ms, context switch < 10 usec
  - 80% of CPU bursts should be shorter than q
- Priority Scheduling = (smallest integer ≡ highest priority)
  - Problem ≡ Starvation – low priority processes may never execute
  - Solution ≡ Aging – as time progresses increase the priority of the process
- Priority with RR = Processes with the same priority run round-robin

- ➔ MLFQ = have separate queues for each priority.



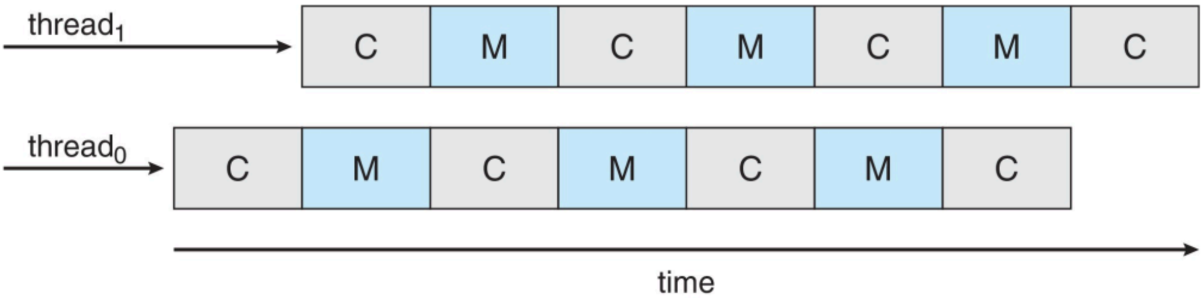
- ◆ number of queues
- ◆ method used to determine when to upgrade a process
- ◆ method used to determine which queue a process will enter when that process needs service
- ◆ scheduling algorithms for each queue
- ◆ method used to determine when to demote a process

- ➔ Symmetric multiprocessing (SMP) is where each processor is self scheduling.
  - ◆ All threads may be in a common ready queue (a)
  - ◆ Each processor may have its own private queue of threads (b)

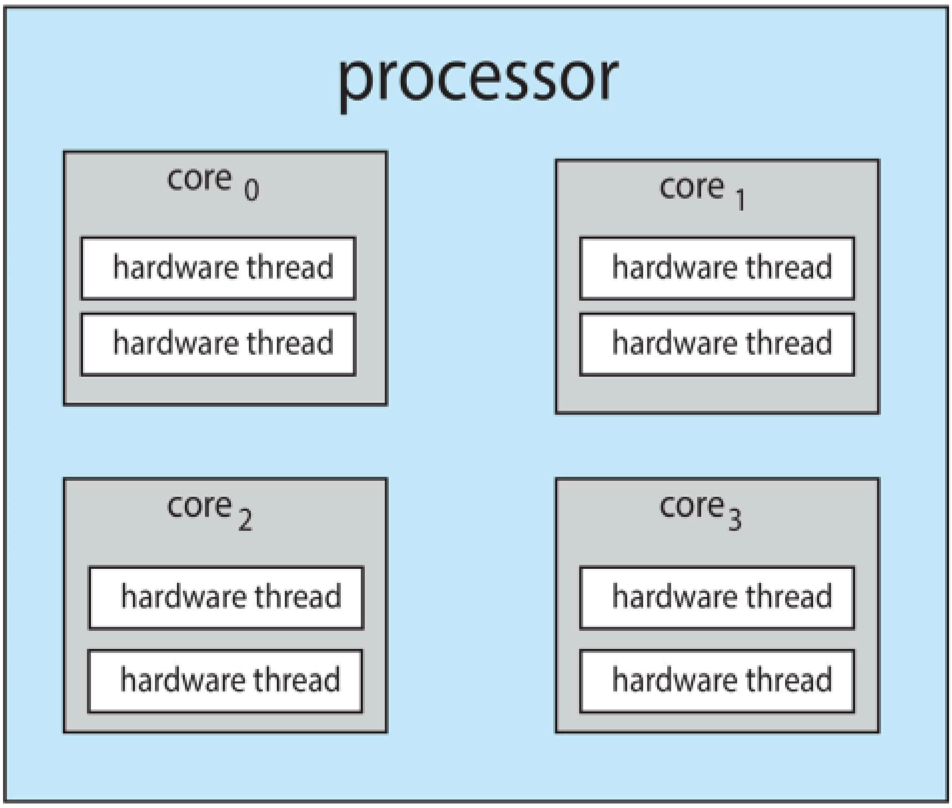


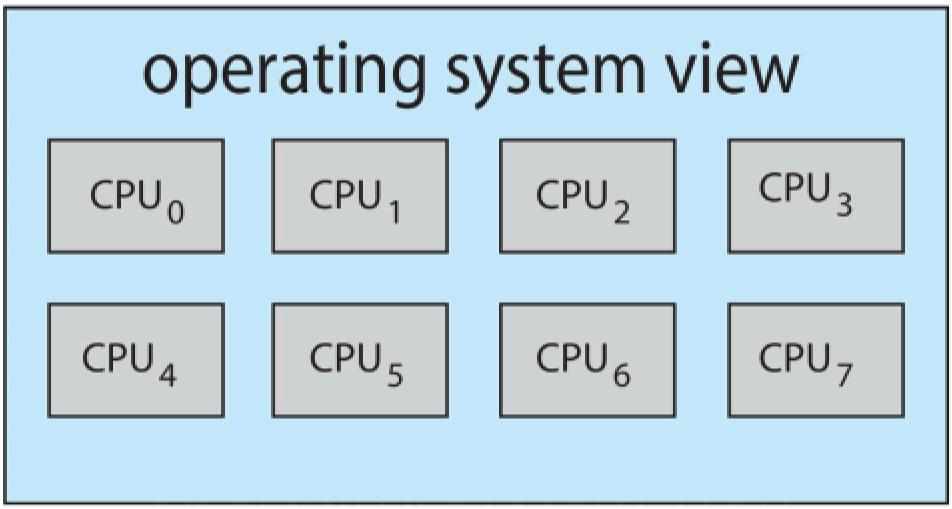
➔ Place multiple processor cores on same
➔ physical chip
➔ Faster and consumes less power
➔ Multiple threads per core also growing

➔ If one thread has a memory stall, switch to another thread!



➔ Chip-multithreading (CMT)
➔ assigns each core multiple hardware threads. (hyperthreading.)
➔ On a quad-core system with 2 hardware threads per core, the operating system sees 8 logical processors.





- ➔ If SMP, need to keep all CPUs loaded for efficiency
- ➔ **Push migration** – periodic task checks load on each processor, and if found pushes task from overloaded CPU to other CPUs
- ➔ **Load balancing** attempts to keep workload evenly distributed
- ➔ **Pull migration** – idle processors pulls waiting task from busy processor

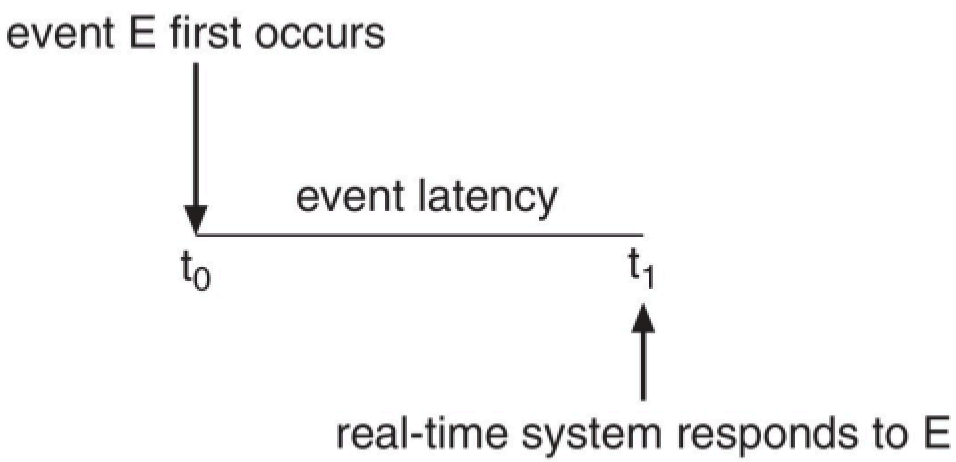
- ➔ Thread loses the contents of what it had in the cache of the processor it was moved off of.

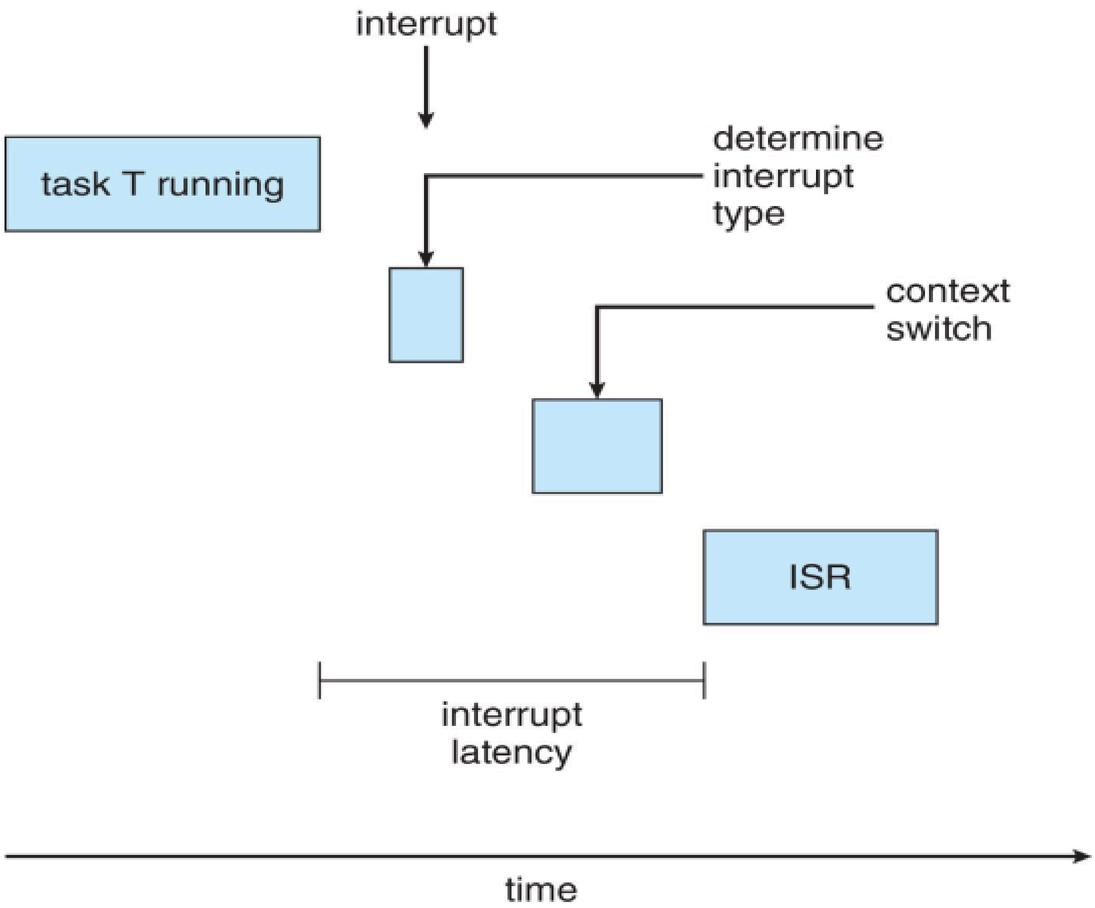
- ➔ **Soft affinity** – the operating system attempts to keep a thread running on the same processor, but no guarantees.
- ➔ **Hard affinity** – allows a process to specify a set of processors it may run on.

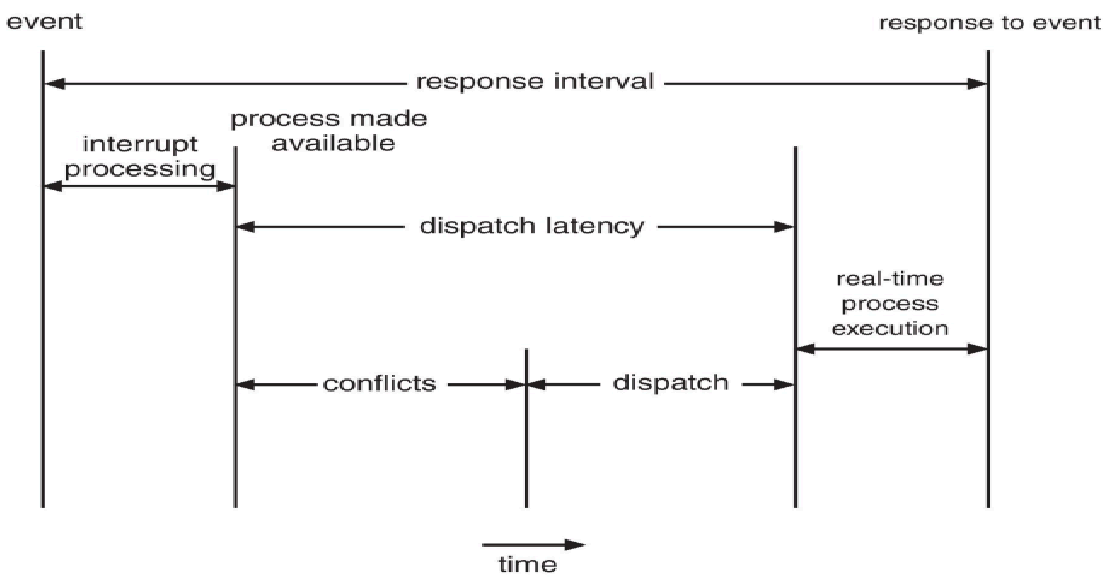
- ➔ **NUMA-aware** OSes will assign memory closest to the CPU the thread is running on.

- ➔ **Real time**
  - ◆ Soft real-time systems – Critical real-time tasks have the highest priority, but no guarantee as to when tasks will be scheduled
  - ◆ Hard real-time systems – task must be serviced by its deadline
- ➔ Event latency – the amount of time that elapses from when an event occurs to when it is serviced.

- ➔ 1.Interrupt latency – time from arrival of interrupt to start of routine that services interrupt
- ➔ 2.Dispatch latency – time for schedule to take current process off CPU and switch to another
- ➔ 1.Preemption of any process running in kernel mode
- ➔ 2.Release by low- priority process of resources needed by high- priority processes




interrupt
task T running
determine
interrupt
type
context
switch
ISR
interrupt
latency
time


event
response to event
response interval
process made
available
interrupt
processing
dispatch latency
real-time
process
execution
conflicts
dispatch
time

- preemptive, priority-based scheduling
- only guarantees soft real-time
- hard real-time must also provide ability to meet deadlines

- Rate-Monotonic
  - Rate of periodic task is 1/p
  - Shorter periods = higher priority;
- EDF (Earliest deadline first)
  - the earlier the deadline, the higher the priority;

- **Algorithm Evaluation**
  - How to select a CPU-scheduling algorithm for an OS?
- Deterministic modeling:
  - Type of analytic evaluation
  - Takes a particular predetermined workload and defines the
  - performance of each algorithm for that workload
  - Simple and fast, but requires exact numbers for input, applies only to those inputs

- Queueing Models:
  - Describes the arrival of processes, and CPU and I/O bursts probabilistically
  - Commonly exponential, and described by mean
  - Computes average throughput, utilization, waiting time, utilization, average queue length, average wait time, etc.
  - Knowing arrival rates and service rates
-

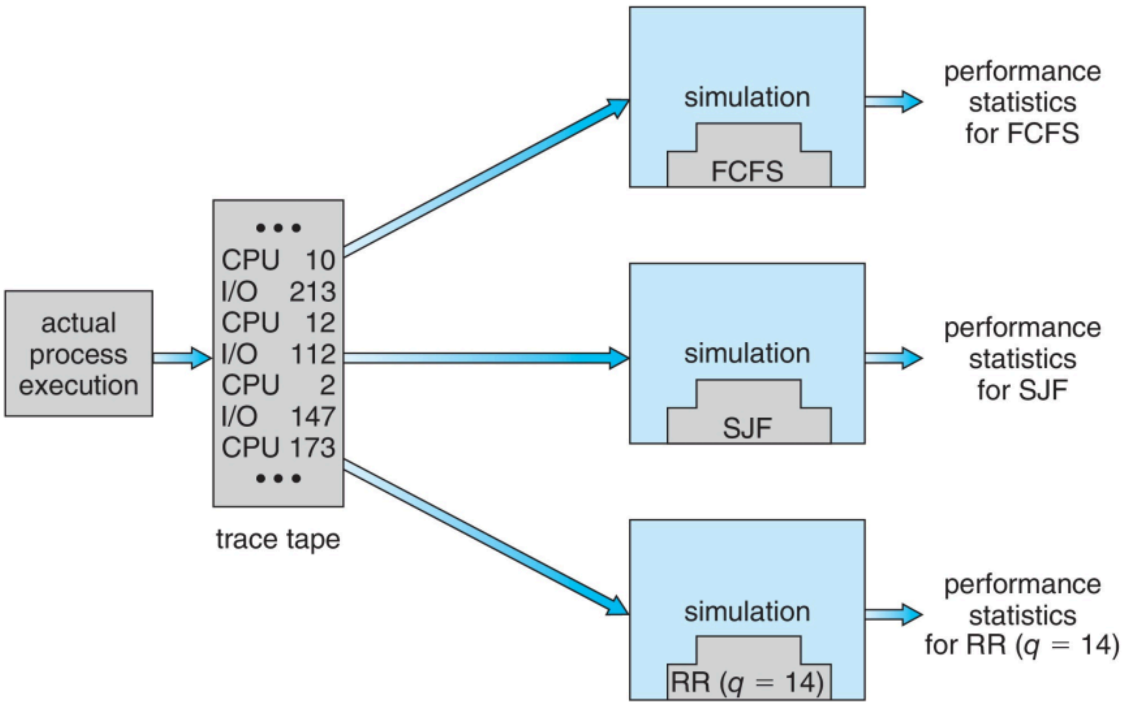
- n = average queue length
- λ = average arrival rate into queue
- W = average waiting time in queue
- Little's law – in the steady state, processes leaving queue must
- equal processes arriving, thus:

- $n = \lambda \times W$

- Simulations
  - Queueing models limited
  - Simulations more accurate
  - Programmed model of computer system
  - Gather statistics, indicating algorithm performance
  - Clock is a variable
  - Data to drive simulation gathered via
    - Random number generator according to probabilities
    - Distributions defined mathematically or empirically
    - Trace tapes record sequences of real events in real systems

  - Even simulations have limited accuracy
  - Just implement new scheduler and test in real systems
    - High cost, high risk
    - Environments vary

◆ Most flexible schedulers can be modified per-site or



per-system