The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

# Chapter 3: Processes

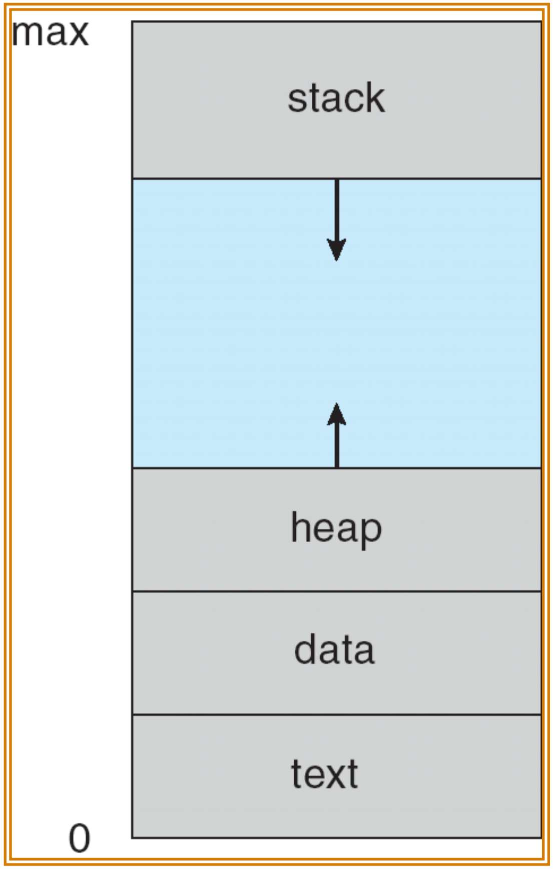
# Process Concept

- An operating system executes a variety of programs:
  - Batch system – jobs
  - Time-shared systems – user programs or tasks
- Textbook uses the terms *job* and *process* almost interchangeably
- Process – a program in execution; process execution must progress in sequential fashion

# Typical Elements of a Process Image

- **Text section** (user program code)
- **Data section** (global variables)
- **Stack**
  - User Stack (temporary data such as function parameters, return addresses, local variables)
  - Kernel Stack (register values, parameters and calling addresses for system calls)
- **Heap** (memory that is dynamically allocated during process run time)
- **Process Control Block** (PCB)

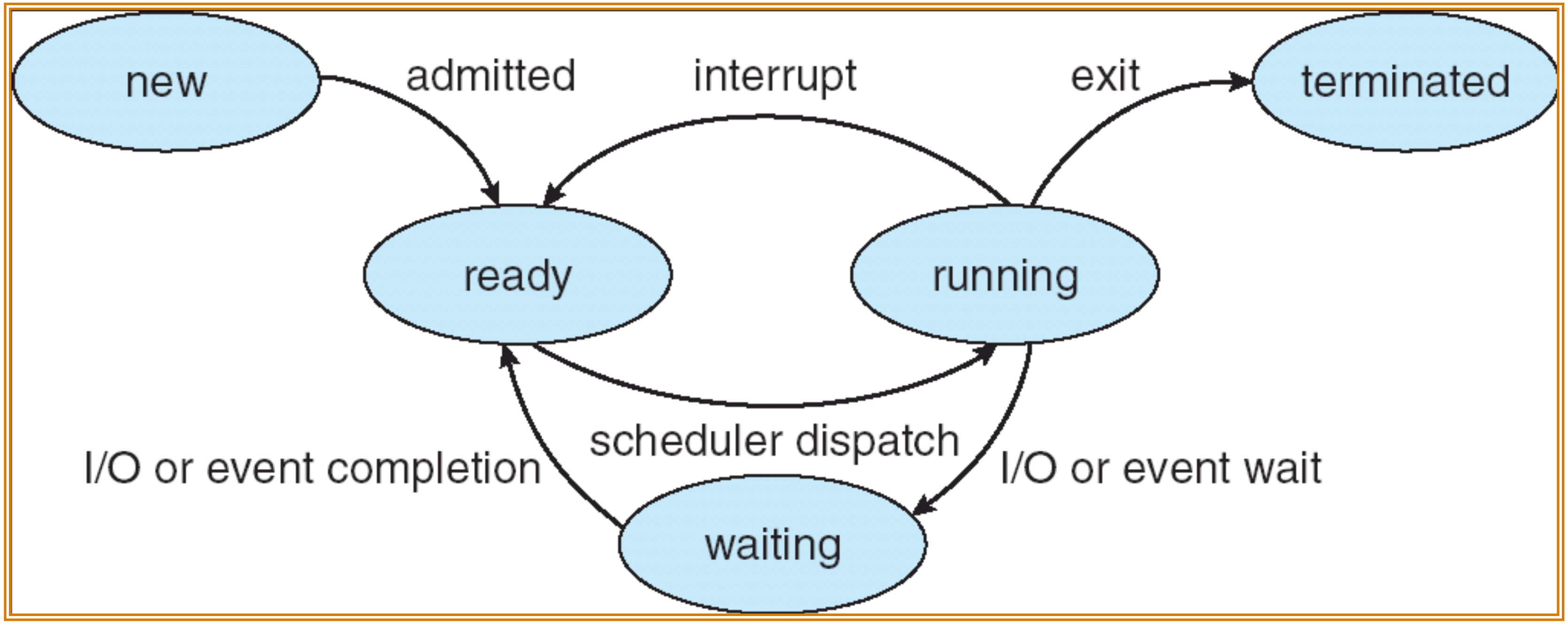
# Process in Memory



# Process State

- As a process executes, it changes *state*
  - **new**:  The process is being created
  - **running**:  Instructions are being executed
  - **waiting**:  The process is waiting for some event to occur
  - **ready**:  The process is waiting to be assigned to a process
  - **terminated**:  The process has finished execution

# Diagram of Process State

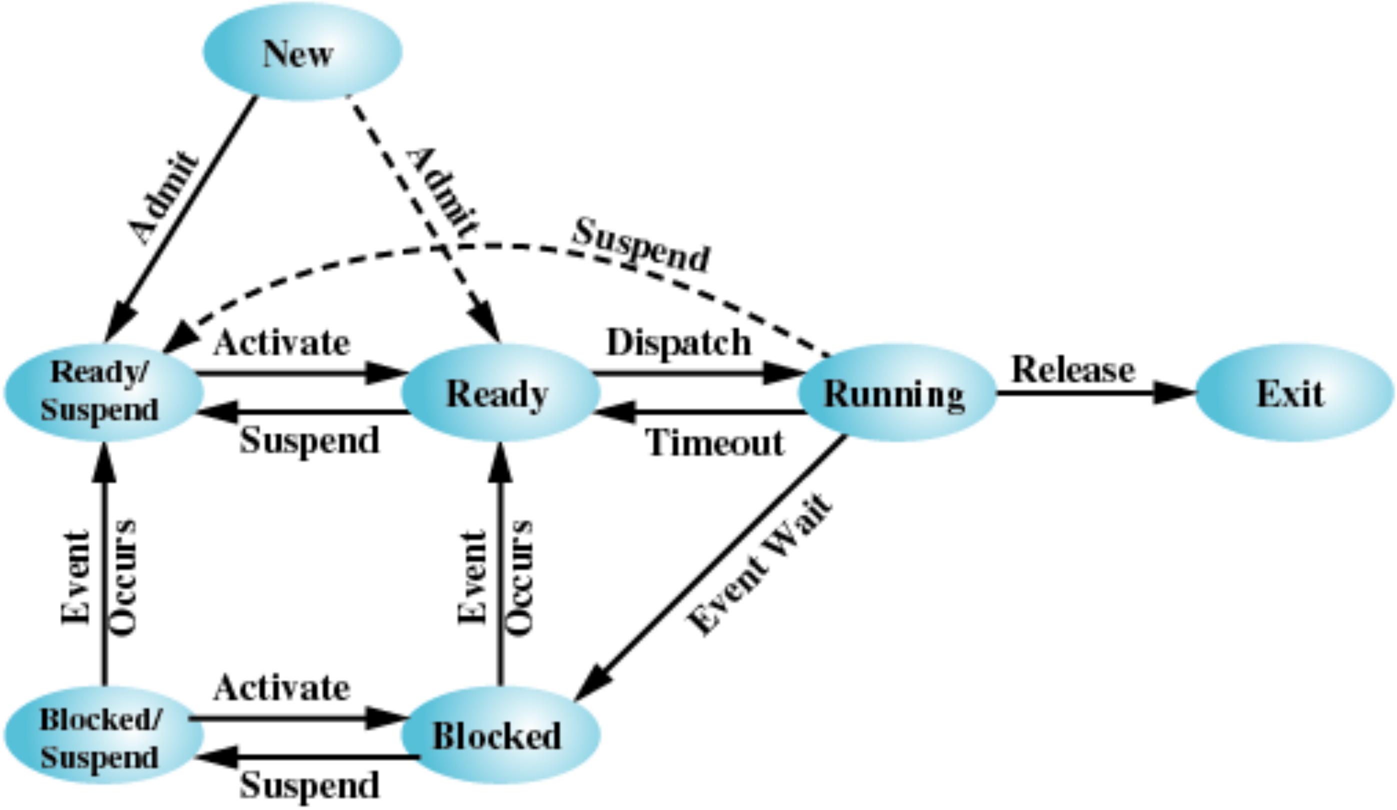


# Need for Swapping

- Processor is faster than I/O; so all processes could be waiting for I/O
- Even with multiprogramming, a processor could be idle most of the time.
- Swapping: moving part or all of a process from main memory to disk (to free up more memory)
  - blocked state becomes suspend state when swapped to disk
  - suspended queue : a queue of existing processes that have been temporarily kicked out of main memory, or suspended.

** **two new states**: ready-suspended & blocked-suspended

# Two Suspend States

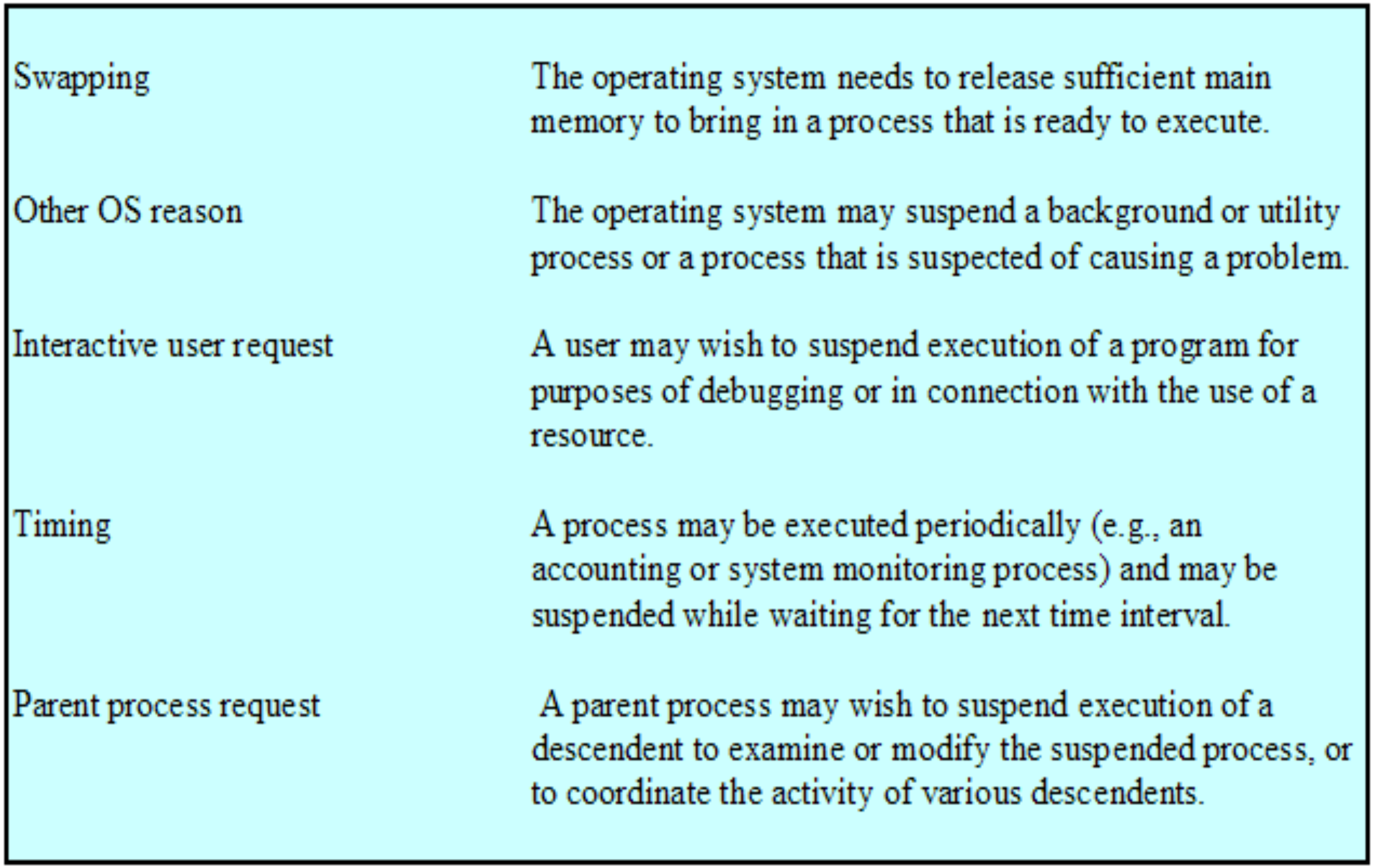


(b) With Two Suspend States

*Blocked state = Waiting state in the previous diagram*

*** *Operating Systems: Internals and Design Principles, William Stallings*

# Reasons for Process Suspension

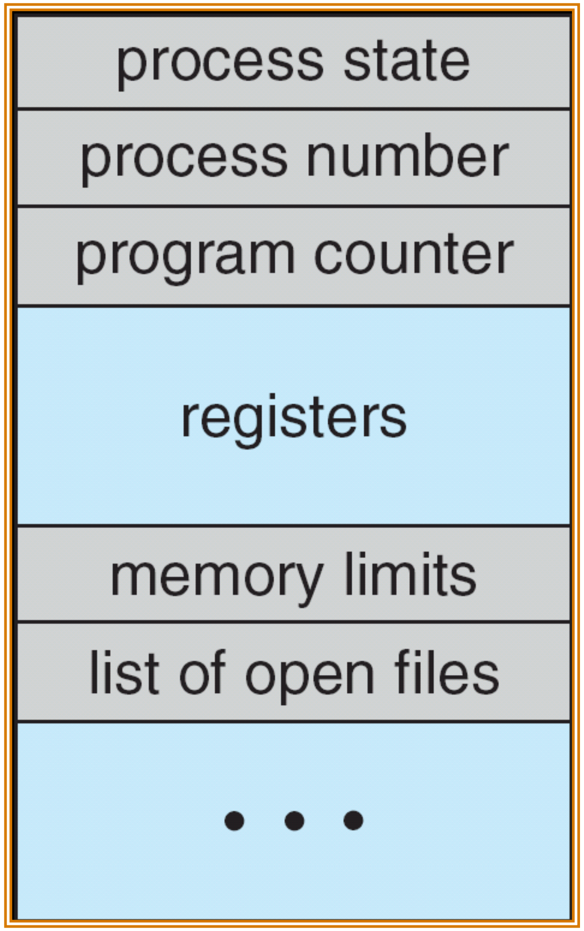


| | |
|---|---|
| Swapping | The operating system needs to release sufficient main memory to bring in a process that is ready to execute. |
| Other OS reason | The operating system may suspend a background or utility process or a process that is suspected of causing a problem. |
| Interactive user request | A user may wish to suspend execution of a program for purposes of debugging or in connection with the use of a resource. |
| Timing | A process may be executed periodically (e.g., an accounting or system monitoring process) and may be suspended while waiting for the next time interval. |
| Parent process request | A parent process may wish to suspend execution of a descendent to examine or modify the suspended process, or to coordinate the activity of various descendents. |

# Process Control Block (PCB)

- Information associated with each process
  - Process state
  - Program counter
  - CPU registers
  - CPU scheduling information
  - Memory-management information
  - Accounting information
  - I/O status information

# Process Control Block (PCB)



process state

process number

program counter

registers

memory limits

list of open files

• • •

# Process Control Block

- Information is classified into three categories:
  a. Process identification,
  b. Processor state information
  c. Process control information

## a. Process identification

- Numeric identifiers that may be stored with the process control block include
  - Identifier of this process
  - Identifier of the process that created this process (parent process)
  - User identifier

# b. Processor State Information

- User-Visible Registers
  - A user-visible register is one that may be referenced by means of the machine language that the processor executes while in user mode.
- Control and Status Registers
  - *Program counter:* Contains the address of the next instruction to be fetched
  - *Condition codes:* Result of the most recent arithmetic or logical operation (e.g., sign, zero, carry, equal, overflow)
  - *Status information:* Includes interrupt enabled / disabled flags, execution mode
- Stack Pointers

# c. Process Control Information

- Scheduling and State Information
  - *Process state*
  - *Priority:* One or more fields may be used to describe the scheduling priority of the process.
  - *Scheduling-related information:* This will depend on the scheduling algorithm used.
  - *Event:* Identity of event the process is awaiting before it can be resumed
- Data Structuring
  - A process may be linked to other process in a queue, ring, or some other structure.
  - all processes in a waiting state for a particular priority level may be linked in a queue.

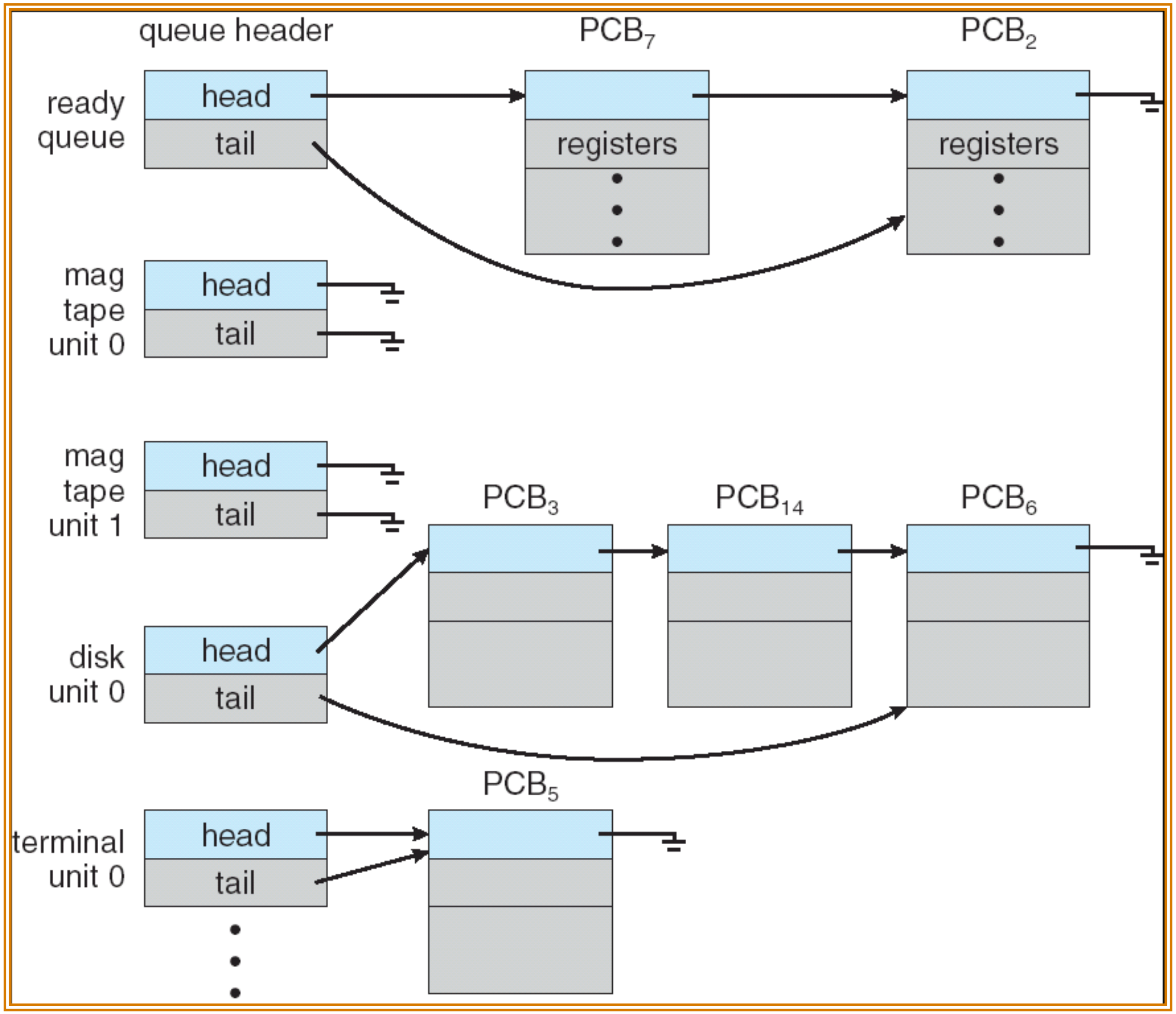
# c. Process Control Information (cont.)

- Interprocess Communication
- Process Privileges
- Memory Management
  - This section may include pointers to segment and/or page tables that describe the virtual memory assigned to this process.
- Resource Ownership and Utilization
  - Resources controlled by the process may be indicated, such as opened files. A history of utilization of the processor or other resources may also be included.

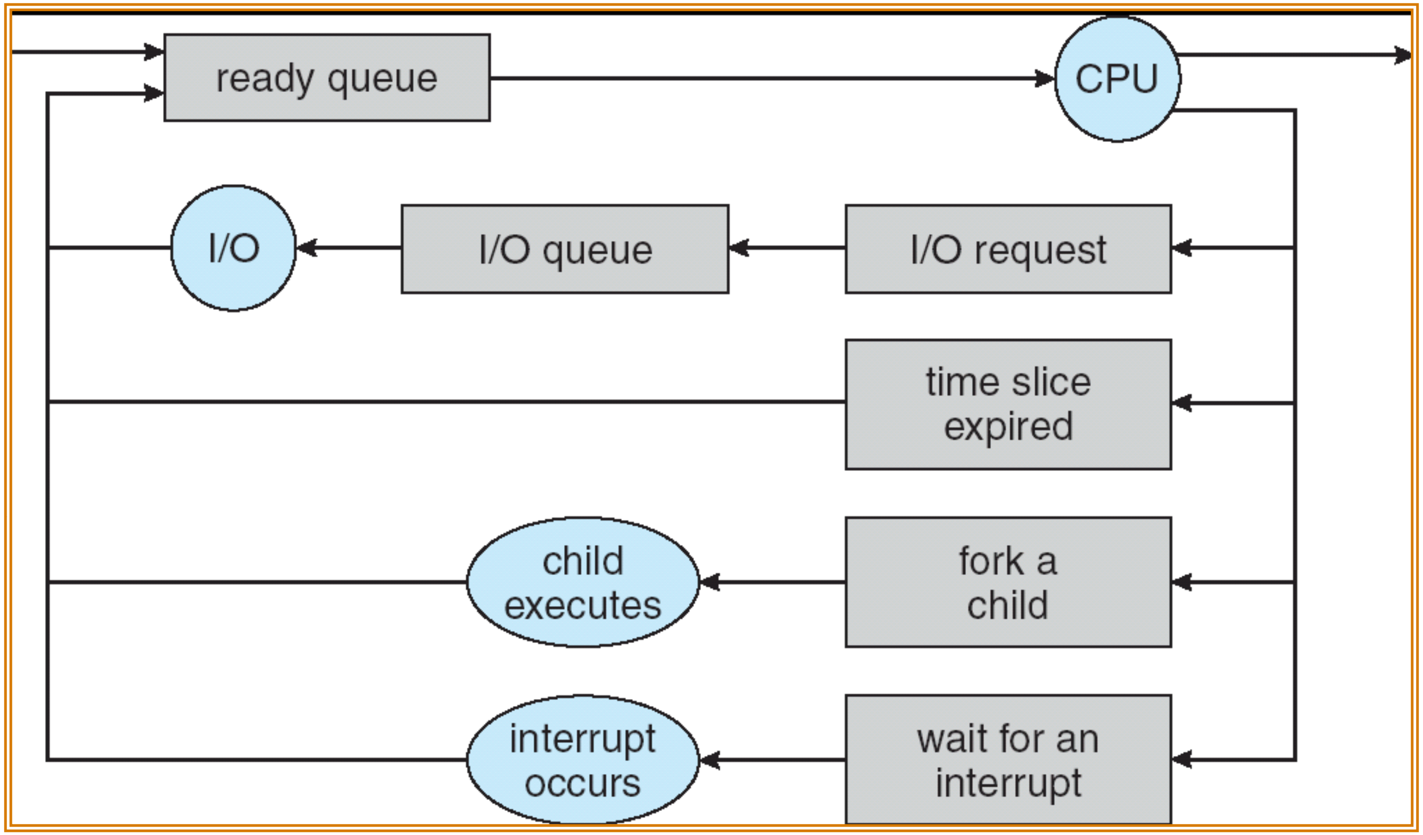
# Process Scheduling Queues

- **Job queue** – set of all processes in the system
- **Ready queue** – set of all processes residing in main memory, ready and waiting to execute
- **Device queues** – set of processes waiting for an I/O device
- Processes migrate among the various queues

# Ready Queue And Various I/O Device Queues



# Representation of Process Scheduling



# Schedulers

- **Long-term scheduler** (or job scheduler) – selects which processes should be brought from the pool (typically a disk) into the ready queue
  - Does not exist in Linux or Windows.
- **Short-term scheduler** (or CPU scheduler) – selects which process should be executed next and allocates CPU

# Schedulers (Cont.)

- Short-term scheduler is invoked very frequently (milliseconds) (must be fast)
- Long-term scheduler is invoked  less  frequently (seconds, minutes) (may be slow)
- The long-term scheduler controls the *degree of multiprogramming*
- Processes can be described as either:
  - **I/O-bound process** – spends more time doing I/O than computations, many short CPU bursts
  - **CPU-bound process** – spends more time doing computations; few very long CPU bursts

# Context Switch

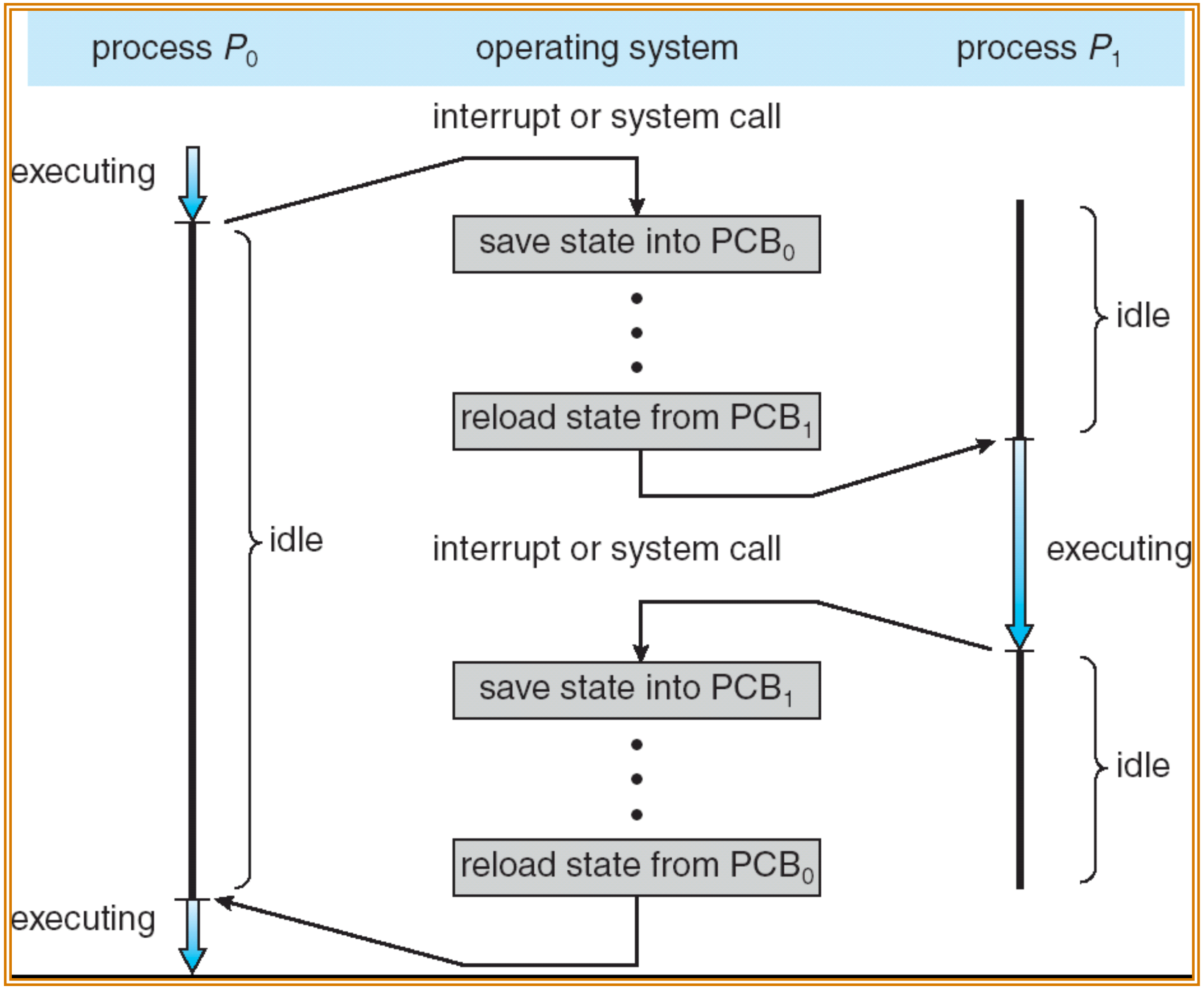
- When CPU switches to another process, the system must save the state of the old process and load the saved state for the new process
- Context-switch time is overhead; the system does no useful work while switching
- Time dependent on hardware support

# Steps in a Process Switch

- Save context of processor including PC and other registers
- Update the process control block of the process that is currently in the running state
  - (changing the state of the process)
- Move process control block to appropriate queue
  - (ready; blocked; ready/suspend)
- Select another process for execution
- Update the process control block of the process selected (change state to running)
- Update memory-management data structures
- Restore the context of the processor

** with the selected process when it was last switched out

# Context Switch



# Process Creation

- Parent process create children processes, which, in turn create other processes, forming a tree of processes
- Resource sharing
  - Parent and children share all resources
  - Children share subset of parent’s resources
  - Parent and child share no resources
- Execution
  - Parent and children execute concurrently
  - Parent waits until children terminate

# Process Creation (Cont.)

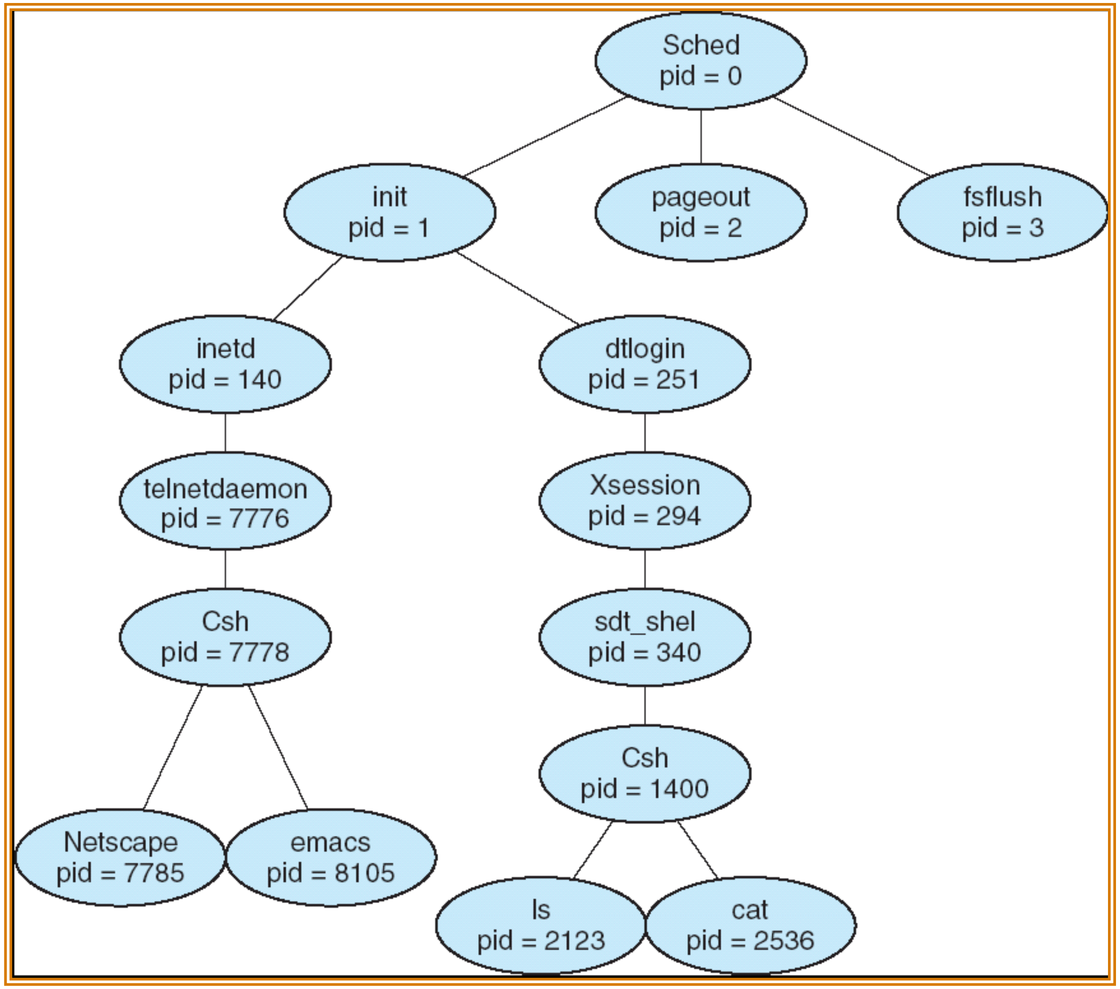
- Address space
  - Child duplicates the address space of the parent.
  - Child has a program loaded into it.
- UNIX examples
  - **fork** system call creates new process.
  - **exec** system call used after a **fork** to replace the process' memory space with a new program.

# Steps of  Process Creation

- Assign a unique process identifier
  - A new entry is added to the primary process table
- Allocate space for the process
- Initialize process control block
- Set up appropriate linkages
  - add new process to linked list  for scheduling queue

    ***  put in  ready  list  or  ready/suspend list
- Create or expand other data structures
  - Ex: maintain an accounting file

# A tree of processes on a typical Solaris



# Process Termination

- Process executes last statement and asks the operating system to delete it (**exit** system call)
  - Output data (return status) from child to parent (**wait** system call)
  - Process' resources are deallocated by operating system
- Parent may terminate execution of children processes
  - Child has exceeded allocated resources
  - Task assigned to child is no longer required
  - parent is exiting and OS not allow child to continue if its parent terminates
    - If parent terminates, all children are terminated by OS (*cascading termination)*