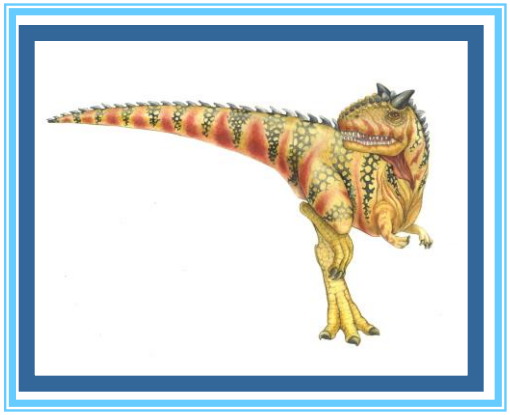
Chapter 3: Processes



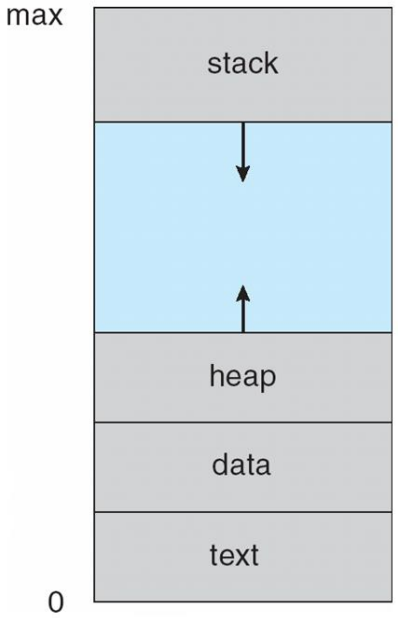
Chapter 3: Processes

- Process Concept
- Process Scheduling
- Operations on Processes
- Interprocess Communication
- Examples of IPC Systems
- Communication in Client-Server Systems

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
- A process includes (among other things):
  - program counter
  - stack
  - data section (globals)
  - text (program)
  - heap

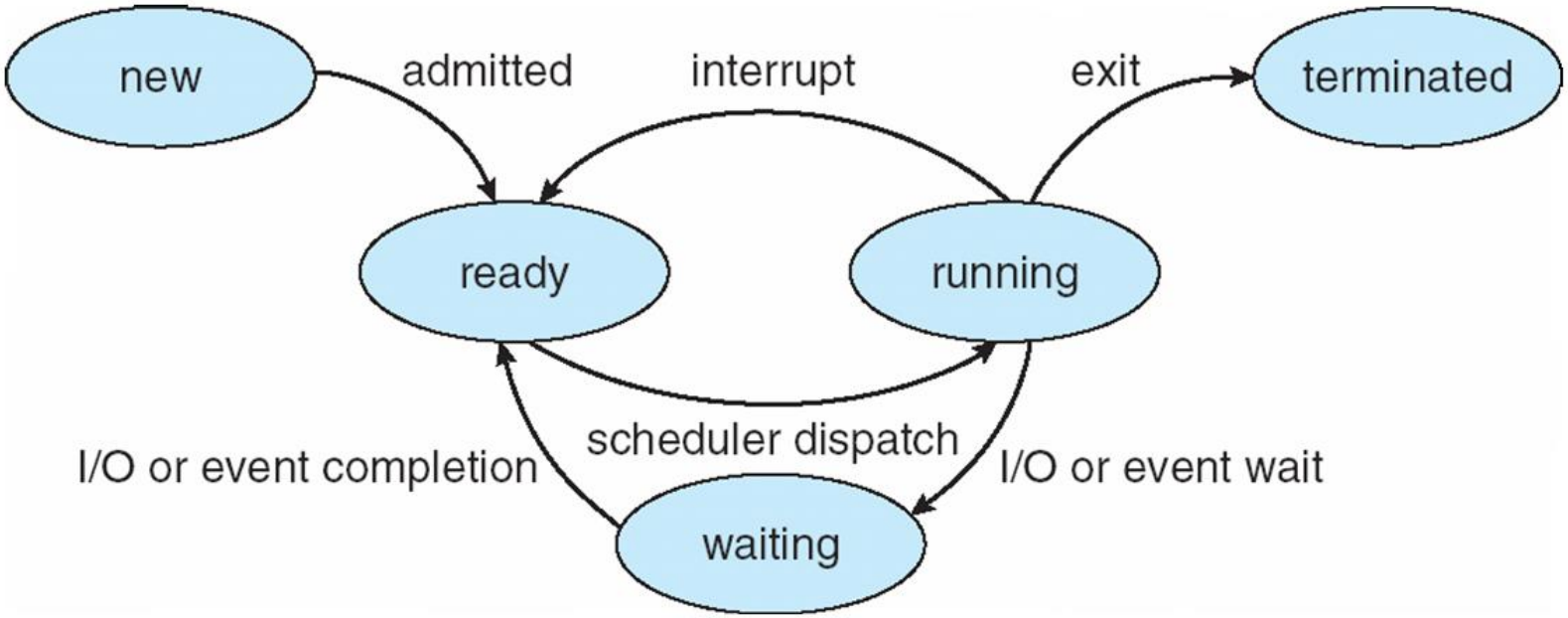
Process in Memory

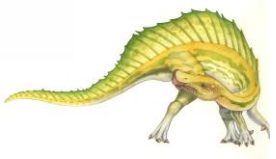


Process State

- As a process executes, it changes *state*

1. How many processes can be in each state?



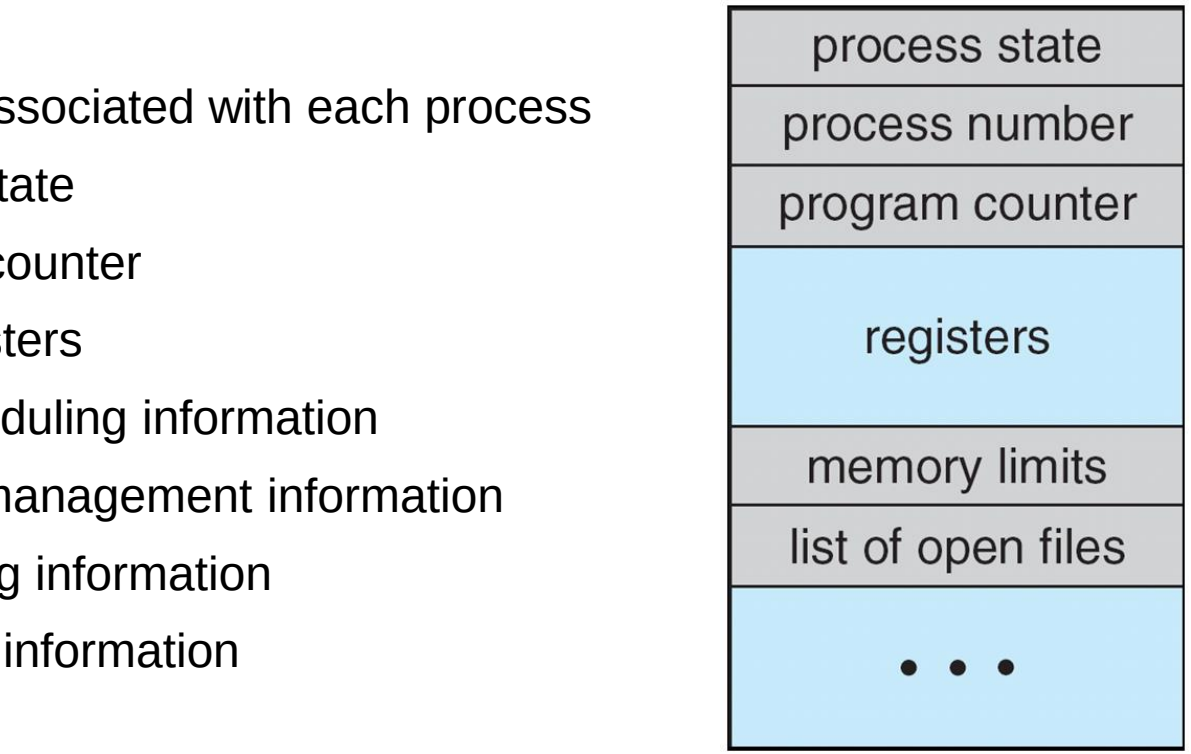


Process Control Block (PCB)

Information associated with each process

- Process state
- Program counter
- CPU registers
- CPU scheduling information
- Memory-management information
- Accounting information
- I/O status information

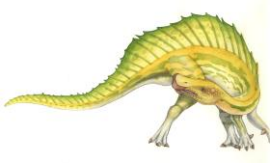
1. Who owns this data structure?



Process Scheduling

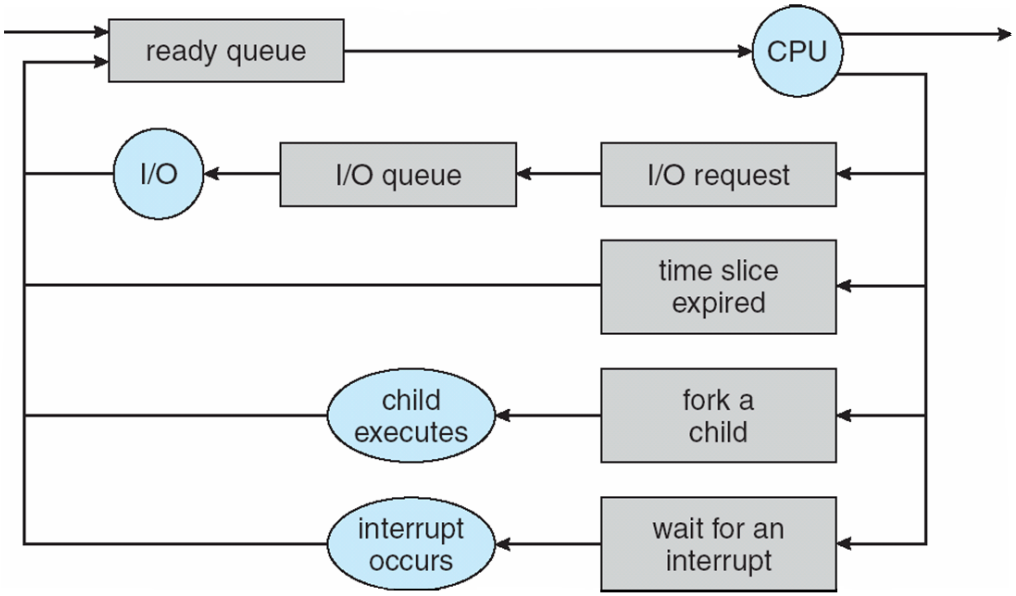
- Processes fall into one of two types
- I/O Bound (e.g.
- CPU Bound (e.g.

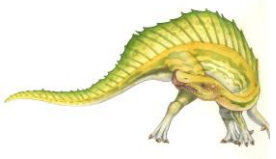
- New processes go into the ready queue
- After a process is allocated the CPU
  - it executes for a while and eventually quits
  - is interrupted
  - waits for the completion of an I/O request



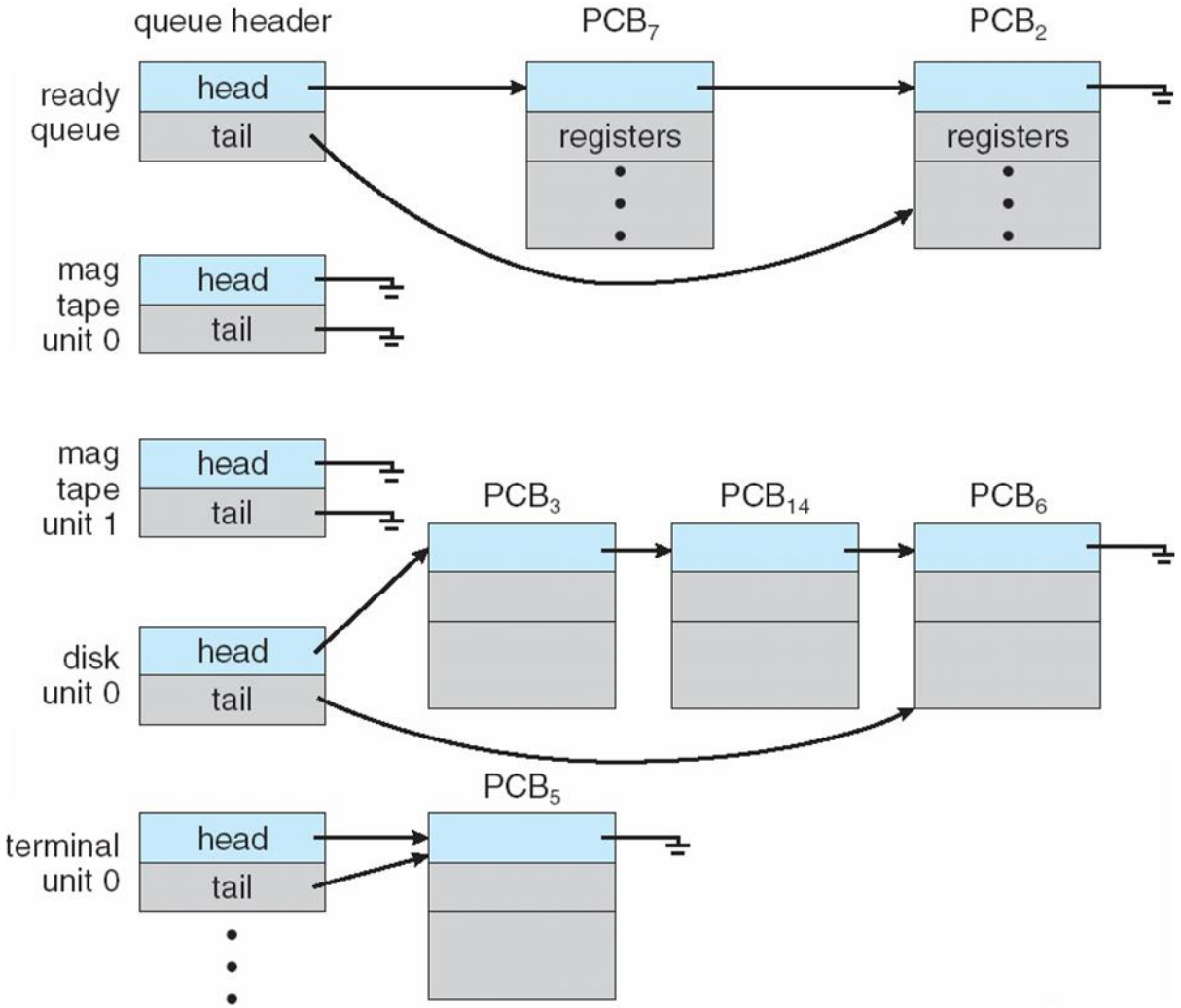
Schedulers

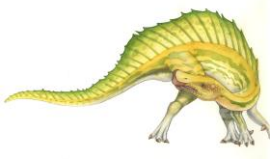
- **Long-term scheduler** (or job scheduler)
  - selects which processes should be brought into the ready queue
    - i.e loads the process into memory for execution
  - must select a good mix of I/O bound and CPU bound processes
- **Short-term scheduler** (or CPU scheduler) – selects which process should be executed next and allocates CPU





Ready Queue And Various I/O Device Queues



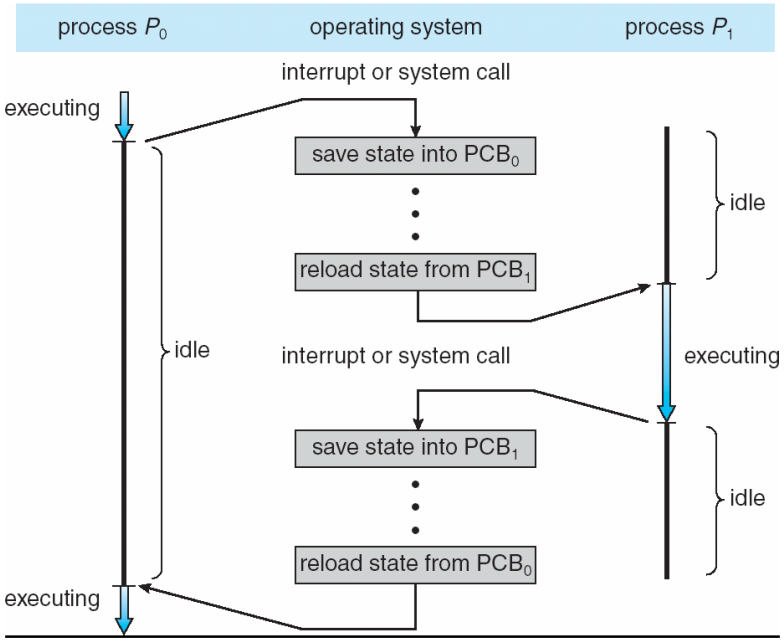


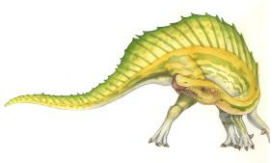
Context Switch

1. What is the context of a process?

2. What is a context switch?

3. Does useful work happen for the user during a context switch? Explain.

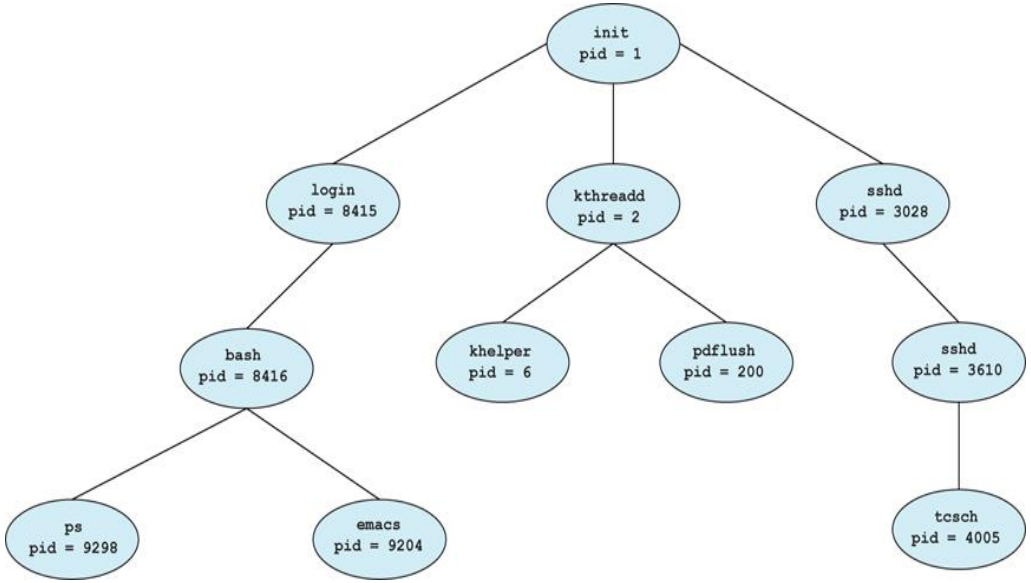


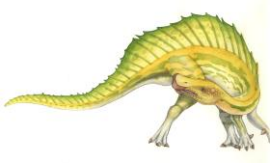


Process Creation

- During program execution, a process may create several new processes
  - Process tree
  - Process Id
  - Parent Process
  - Child Process

1. Explain the login process
2. How are there 2 processes created by bash





Process Creation

- Address space
  - Child duplicate of 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

Process Creation

```
/* This code works on Zeus! What happens????? */
int main()
{
        pid_t  pid;
        int value = 0;
        value = 9;
        /* fork another process */
        pid = fork();
        fprintf(stderr,"The value: %d", value);
        if (pid < 0) { /* error occurred */
         fprintf(stderr, "Fork Failed");
         exit (1);
        }
        else if (pid == 0) { /* child process */
         execlp("/bin/ls", "ls", NULL);
        }
        else { /* parent process */
         wait (NULL); /* parent will wait for the child to complete */
         printf ("Child Complete");
         exit (0);
        }
}       /* page 118 of Silberschatz */
```