Lecture 04

Process Management

- A process needs resources (CPU, memory, files…) to do a task
- These resources are either given to the process when it is created, or allocated to it while it is running
- A **program** is a *passive* entity
- A **process** is an *active* entity, with a program counter giving the next instruction to execute
- The execution of a process must be sequential
- 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** specifying location of next instruction to execute in each thread
- Typically system has many processes, some user, some operating system (kernel) running concurrently on one or more CPUs

    - Concurrency by multiplexing the CPUs among the processes / threads
- Processes can execute concurrently by multiplexing the CPU
- In connection with process management, the OS is responsible for
    - Scheduling processes and threads on the CPUs
    - Creating and deleting both user & system processes
    - Suspending and resuming processes
    - Providing mechanisms for process synchronization
    - Providing mechanisms for process communication
    - Providing mechanisms for deadlock handling

PART TWO: PROCESS MANAGEMENT

- A process can be thought as a program in execution.
    - A process will need resources - such as CPU time, memory, files, and I/O devices - to accomplish its task.
    - These resources are allocated to the process either when it is created or while it is executed.
- A process is the unit of work in most systems.
- Systems consist of a collection of processes:
    - Operating-system processes execute system code
    - User processes execute user code
- All these processes may execute concurrently.
- Although traditionally a process contained only a single thread of control as it ran, most modern operating systems now support processes that have multiple threads.
- The operating system is responsible for the following activities in connection with process and thread management:
    - The creation and deletion of both user and system processes;
    - The scheduling of processes;
    - and the provision of mechanisms for synchronization, communication, and deadlock handling for processes.

Chapter 3: Process Concept

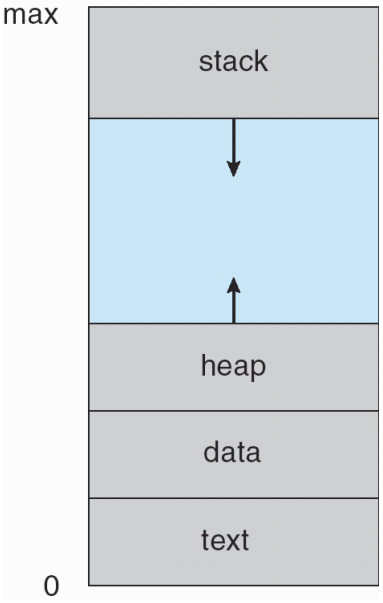
- **Objectives**:
    - To introduce the notion of a process - a program in execution, which forms the basis of all computation.
    - To describe the various features of processes, including scheduling, creation and termination, and communication.
    - To describe communication in client-server systems.

Process Concepts

- 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

The Process

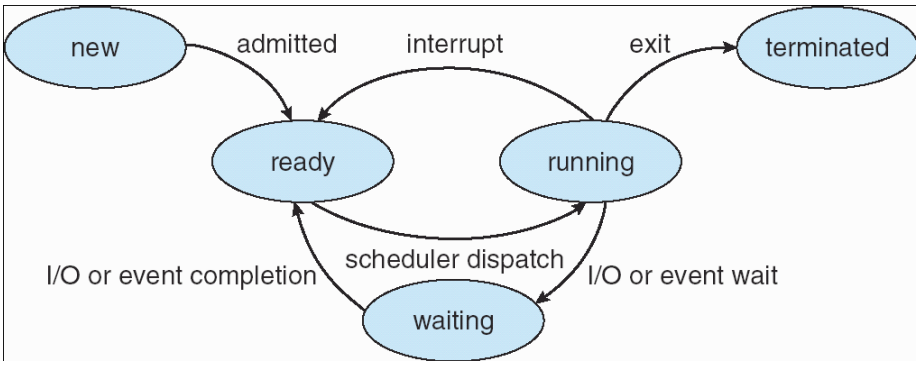
- Process = an active entity, with a program counter (to indicate the current activity), process stack (with temporary data), and a data section (with global variables)
- Text section = the program code
- If you run many copies of a program, each is a separate process (The text sections are equivalent, but the data sections vary)
- **Process**–a program in execution; process execution must progress in sequential fashion
- A process includes:
    - program counter
    - stack
    - data section
- A process in memory



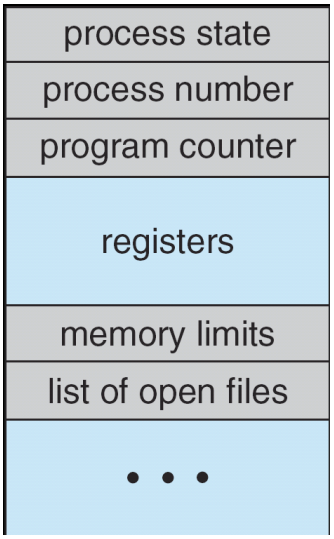
- p.102 give description of stack, heap, data and text areas

Process State

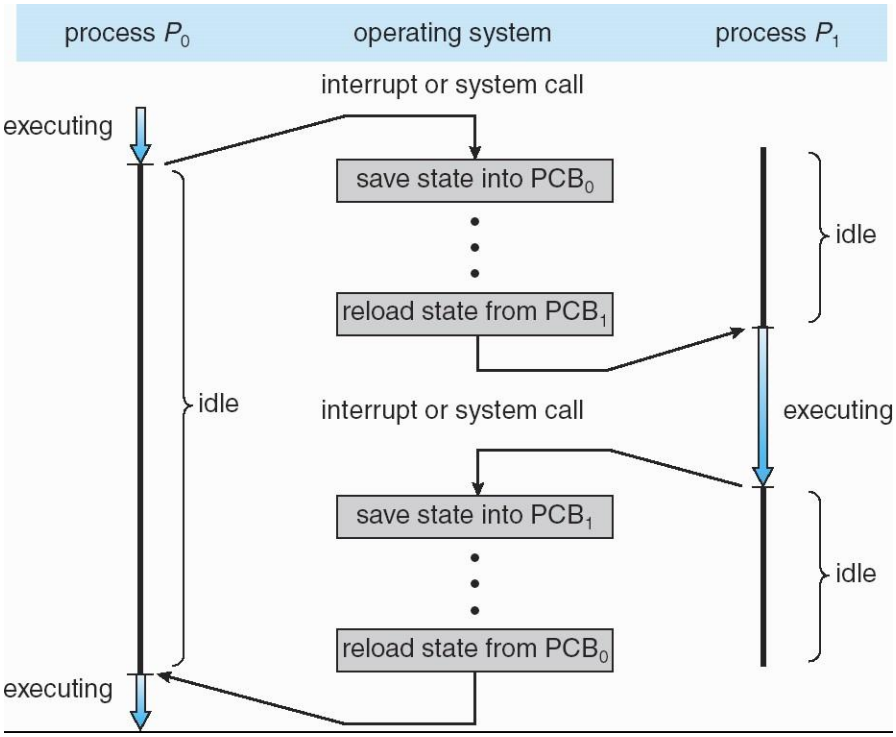
- Each process may be in one of the following states:
    - New (Process is being created)
    - Running (Instructions are being executed)
    - Waiting (Process is waiting for an event, e.g. I/O)
    - Ready (Process is waiting to be assigned to a processor)
    - Terminated (Process has finished execution)
- Only one process can be *running* on any processor at any instant



Process Control Block



- Contains information associated with a specific process:
    1. Process state (as above)
    2. Program counter (indicating the next instruction's address)
    3. CPU registers (Info must be saved when an interrupt occurs)



- CPU-scheduling info (includes process priority, pointers…)
- Memory-management info (includes value of base registers…)
- Accounting info (includes amount of CPU time used…)
- I/O status info (includes a list of I/O devices allocated…)

Threads

- Many OS's allow processes to perform more than one task at a time