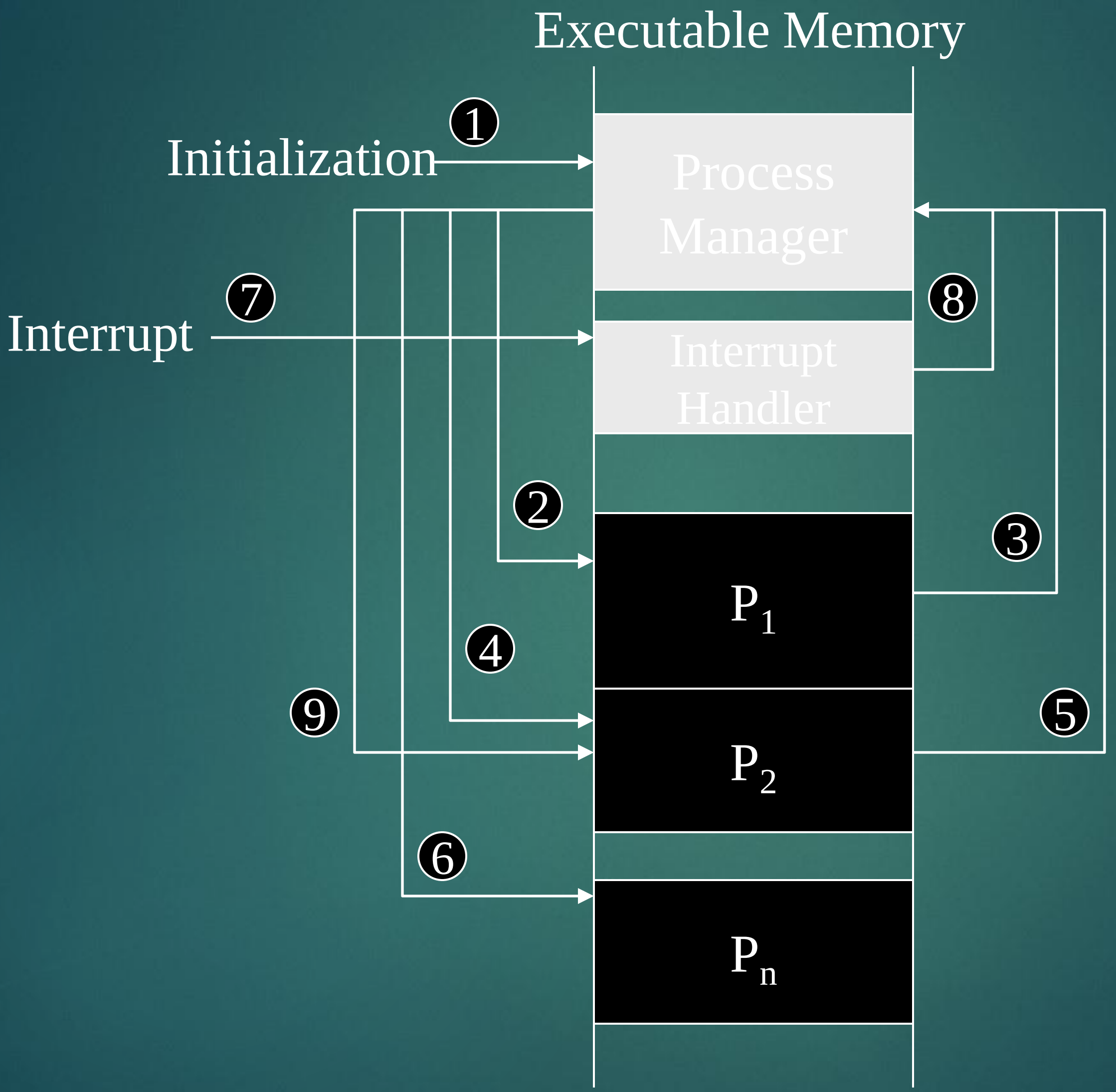
Lecture 3

PROCESS MODEL

Context Switching



When to Switch a Process

- Clock interrupt
  - process has executed for the maximum allowable time slice
- I/O interrupt
- Memory fault
  - memory address is in virtual memory so it must be brought into main memory

When to Switch a Process

- Trap
  - error or exception occurred
  - may cause process to be moved to Exit state
- Supervisor call
  - such as file open

Process Creation

- Assign a unique process identifier
- Allocate space for the process
- Initialize process control block
- Set up appropriate linkages
    - Ex: add new process to linked list used for scheduling queue
- Create or expand other data structures
    - Ex: maintain an accounting file

Change of Process State

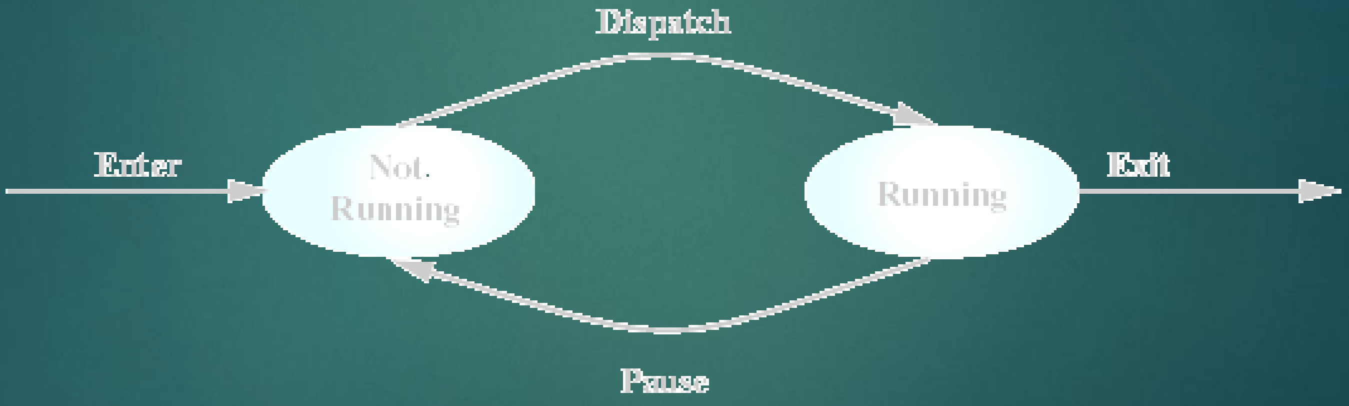
- Save context of processor including program counter and other registers
- Update the process control block of the process that is currently in the Running state
- Move process control block to appropriate queue – ready; blocked; ready/suspend
- Select another process for execution

Change of Process State

- Update the process control block of the process selected
- Update memory-management data structures
- Restore context of the selected process

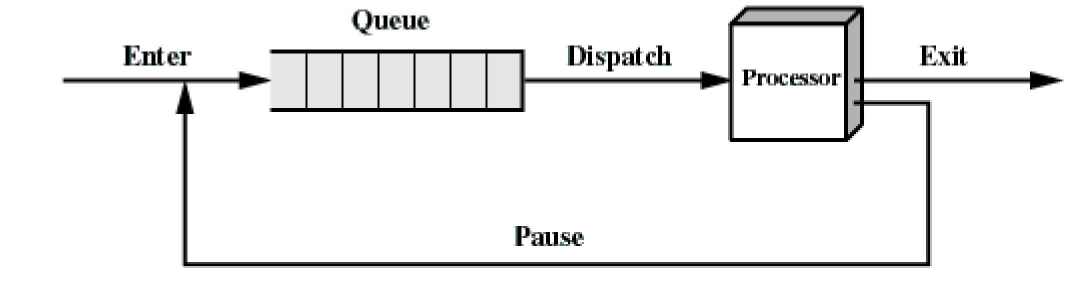
Two-State Process Model

- Process may be in one of two states
  - Running
  - Not-running



(a) State transition diagram

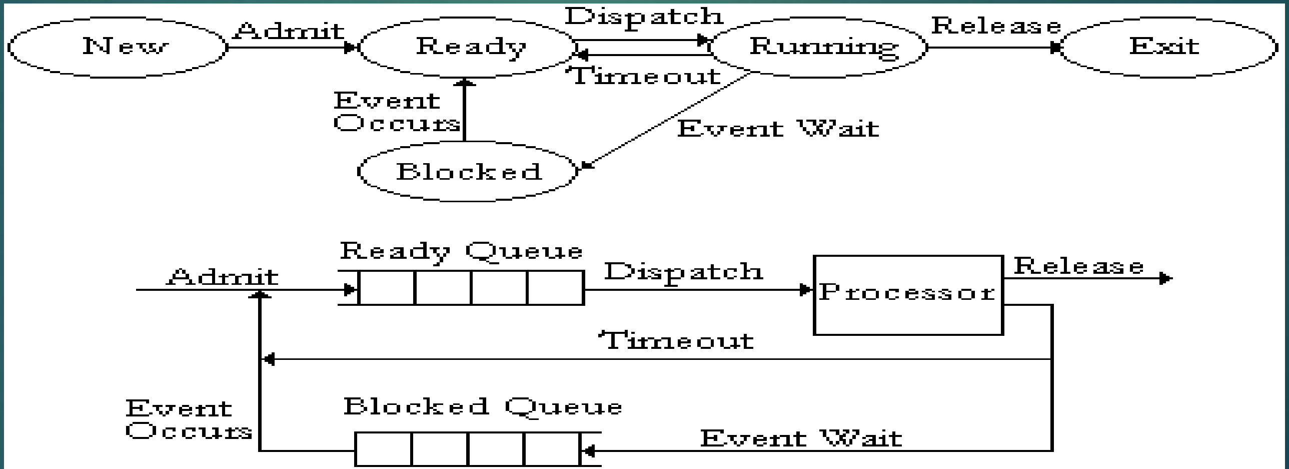
Not-Running Process in a Queue



(b) Queuing diagram

Five-state Model

- Processes may be waiting for I/O
- Use additional states:
  - *Running*: currently being run
  - *Ready*: ready to run
  - *Blocked*: waiting for an event (I/O)
  - *New*: just created, not yet admitted to set of runnable processes
  - *Exit*: completed/error exit



Five-state Model

- May have separate waiting queues for each event Transitions:
  - Null → New – Process is created
  - New → Ready – O.S. is ready to handle another process (Memory, CPU)
  - Ready → Running – Select another process to run
  - Running → Exit – Process has terminated
  - Running → Ready – End of time slice or higher-priority process is ready
  - Running → Blocked – Process is waiting for an event (I/O, Synchronization)
  - Blocked → Ready – The event a process is waiting for has occurred, can continue
  - Ready → Exit – Process terminated by O.S. or parent
  - Blocked → Exit – Same reasons

Threads and processes

- Most modern OS's (Mach, Chorus, NT, modern UNIX) therefore support two entities:
  - the process, which defines the address space and general process attributes (such as open files, etc.)
  - the thread, which defines a sequential execution stream within a process
- A thread is bound to a single process / address space
  - address spaces, however, can have multiple threads executing within them
  - sharing data between threads is cheap: all see the same address space
  - creating threads is cheap too!
- Threads become the unit of scheduling
  - processes / address spaces are just containers in which threads execute