Process

- What is a Process?
- Process States and Life Cycle
- Process Scheduling

What is a Process?

- A process is a program in execution
  - A process is not the same as “program”
    - A program is a passive text of executable codes resides in disk.
    - A process is an active entity ripe for execution (must have a program counter, stack and data section).
    - Multiple people can run the same program, each running a copy of the same program text, but each is a distinct process.
  - Type (HP Unix):
    %ps shows all my processed with little detail
    %ps -fl more detail
    %ps -efl all processed with full detail
- User and OS processes
  - jobs (batch system), tasks (time shared system), process (generic)
Process (Continued..)

- The process must have (at least):
  - ID
  - Code of the program
  - Program’s static data
  - Program’s dynamic data
  - Content of Program Counter (PC)
  - Content of Stack Pointer (SP)
  - Content of Program Status Word (PSW)
  - Content of general purpose registers
  - CPU scheduling information
  - Memory management info (memory limits etc.)
  - Accounting information
  - I/O status information

---

Process Creation/ Termination

- Reasons for process creation:
  - New batch job
  - User starts a program
  - OS creates process to provide a service
  - Program starts another process

- Reasons for process termination:
  - Normal completion
  - Exceed time limit
  - I/O failure
  - Memory unavailable
  - Bounds violation
  - Protection error
  - Arithmetic error
  - Privileged instruction
  - Invalid instruction
  - Human intervention
  - Parent termination
  - Parent request
Process Execution

- Conceptual model of Processes executing:
  - Process A
  - Process B
  - Process C
  - Process D

- Actual interleaved execution of the 4 processes:
  - Process A
  - Process B
  - Process D
  - Process A
  - Process B
  - Process A
  - Process C
  - Process D
  - Process C

---

A Two State Process Model

- A Process is either “running” or “not running”

**State Diagram**

- Not running
- running
- New process Entry
- dispatch
- Exit
- pause

**Queuing Diagram**

- Enter
- queue
- dispatch
- CPU
- Exit
Process Waiting.....

- Some reasons why a process that might otherwise be running needs to wait:
  ♦ Wait for user to type the next key
  ♦ Wait for output to appear on the screen
  ♦ Program tried to read a file
  ♦ Netscape tried to follow a link (URL)

- OS Must distinguish between:
  ♦ Processes that are ready to run, and waiting for the time slice.
  ♦ Processes that are waiting for something to happen.

Five State Process Model

- States:
  ♦ New
  ♦ Running
  ♦ Ready
  ♦ Waiting
  ♦ Terminated

Scheduler dispatch

New

Running

Ready

Waiting

Terminated

admitted

Event done

interrupt

Event wait

exit
Process Control Block

- For Every process OS maintains a data structure that represents the process and its states

- Process ID
  - State
  - User IP owner
  - PC, SP, PSW and other registers
  - memory management info
  - list of open files
  - IO states
  - CPU scheduling (priority)

- pointer
- state
- Process number
- Program counter
- registers
- Memory limits
- List of open files
Ready Queue and Other I/O Device Queues

- **Ready queue**
  - Head: PCB7
  - Tail: PCB2

- **Tape 1 queue**
  - Head: PCB3
  - Tail: PCB1

- **Disk queue**
  - Head: PCB4
  - Tail: PCB5

- **Terminal queue**
  - Head: PCB6
  - Tail: PCB7

- **Tape 2 queue**
  - Head: PCB8
  - Tail: PCB9

---

Process Scheduling

- **Job queue**

- **Ready queue**

- **CPU**

- **I/O**
  - IO queue
  - IO request
  - Time expired
  - Fork a child

- **Child terminates**
  - Child executes
  - Interrupt occurs

- **Wait for interrupt**
Next Class..

- Operation on processes
- Cooperating processes
- Process Communication
- Threads