Lecture Topics

- Today: Processes
  (Stallings, chapter 3.1-3.6)
- Next: continued

Announcements

- Consulting hours
- Self-Study Exercise #3
- Project #1 (due 1/26)
- Project #2 (due 2/2)
Processes

- All multiprogramming operating systems are built around the concept of the *process*

Main issues:
  - different process states, transitions among them
  - process description (data structures maintained)
  - process control (switching between processes)

OS Requirements

- Interleave the execution of multiple processes to maximize processor utilization while providing reasonable response time

- Allocate resources to processes, based on policies and priorities

- Support inter-process communication
Background

- Computer platform consists of a collection of hardware resources
- OS provides a convenient, feature-rich, secure, and consistent interface for applications
- OS provides a uniform, abstract representation of resources that can be requested and accessed by applications

OS Manages Execution of Applications

- Resources made available to multiple applications
- Processor is switched among multiple applications
- The processor and I/O devices used efficiently
Process Definitions

- An instance of a program running on a computer
- The entity that can be assigned to and executed on a processor
- A unit of activity characterized by the execution of a sequence of instructions, a current state, and an associated set of system resources

Process Elements

Process Control Block
- Identifier
- State
- Priority
- Program counter
- Memory pointers
- Context data
- I/O status info
- Accounting info
Process States

- Behavior of *process*: sequence of instructions executed by that process (trace)
- Behavior of *processor*: interleaved sequences of instructions from individual processes
- *Dispatcher* (part of OS) switches the processor from one process to another

---

Example

Assume no virtual memory: dispatcher and three processes are fully loaded into memory

Program counter currently points to an address in Process B
Assume timer interrupt after 6 clock cycles

Assume dispatcher is at addresses 100-105

Assume dispatcher selects next process using "round robin"
Process Creation

- Submission of a batch job
- User logs onto system
- Created by OS to provide a service (ex: control print job)
- One process creates another process (ex: parent process spawns child process)
Process Termination

- Batch job terminates
- User logs off system
- Application halts normally
- Error (ex: segmentation fault)
- Parent process terminates child process

Execution patterns

Execution of a process is usually a pattern of CPU burst followed by I/O burst (repeated)
Execution patterns

- CPU-bound processes have few bursts, but long bursts
- I/O-bound processes have many bursts, but short bursts
- Pattern varies within a process – depends on actions within the process

Most processes: many short CPU bursts, few long CPU bursts
Refinement: Five-State Process Model

Previous Example
Process Switch

- Process currently in Running state
  - save context (PC, other registers)
  - move PCB to appropriate queue
  - update PCB (state, accounting info)

- Process selected by OS
  - update PCB (state, accounting info)
  - move PCB out of Ready queue
  - restore context (PC, registers)

Queueing Models

(a) Single blocked queue
Suspended Processes

- Processor is faster than I/O devices, so all active processes could be waiting for I/O

- Swap some processes to disk to free up more memory (activate other processes)

- Two new states:
  - Blocked/Suspend
  - Ready/Suspend
Reasons for Process Suspension

- Swapping: OS needs to free up memory
- Interactive user request (ex: debugging)
- Timing (ex: periodic process)
- Parent process request (ex: suspend child)
- Other OS reasons (ex: misbehaving process)