Lecture Topics

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

Announcements

- Self-Study Exercise #3
- Project #1 (due 9/13)
- Project #2 (due 9/20)
OS and Processes

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

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

Definitions of "Process"

- 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
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 switched among multiple applications
- Processor and I/O devices used efficiently

Processes

- A process consists of (at least):
  - code and data for executing program
  - program counter (PC) and status register (PSW)
  - general-purpose registers
  - set of OS resources (open files, signals received, network connections, …)

- In other words, everything the OS needs to run the program -- or to re-start it, if it’s interrupted at some point
## Process Control Block

**All info about process**
- Identifier
- State
- Priority
- Program counter
- Memory pointers
- Context data
- I/O status info
- Accounting info

### PCB assigned to process when created

- OS maintains info in PCB while process is in existence
- OS moves PCBs between queues to manage system
**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
Example (cont)

Assume timer interrupt after 6 clock cycles

Assume dispatcher is at addresses 100-105

Assume dispatcher selects next process using "round robin"
Two-State Process Model

Process Creation

- Submission of a batch job
- User logs onto system
- OS creates 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: 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

![Five-State Process Model Diagram]

Previous Example

<table>
<thead>
<tr>
<th>Time</th>
<th>Process A</th>
<th>Process B</th>
<th>Process C</th>
<th>Dispatcher</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-10</td>
<td>Running</td>
<td>Ready</td>
<td>Blocked</td>
<td>Running</td>
</tr>
<tr>
<td>10-20</td>
<td>Ready</td>
<td>Blocked</td>
<td>Ready</td>
<td>Ready</td>
</tr>
<tr>
<td>20-30</td>
<td>Blocked</td>
<td>Ready</td>
<td>Ready</td>
<td>Running</td>
</tr>
<tr>
<td>30-40</td>
<td>Ready</td>
<td>Blocked</td>
<td>Ready</td>
<td>Ready</td>
</tr>
<tr>
<td>40-50</td>
<td>Blocked</td>
<td>Ready</td>
<td>Blocked</td>
<td>Running</td>
</tr>
</tbody>
</table>

Legend:
- = Running
- = Ready
- = Blocked
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
Queueing Models

(b) Multiple blocked queues