Preview

- Process Models
- Process Creation
- Process Termination
- Process State
- Process Implementation
The Processes

- We assume that there is one CPU in our system.

- Multiprogramming
  - several jobs are loaded in memory
  - illusion of parallelism – *Pseudo parallelism*

- CPU switches from one process to another based on chosen scheduling algorithm and process state.
The Processes

[Diagram of CPU with ALU, Registers, Control Unit, RAM with sections labeled Job1, Job2, Job3, and OS]
The Process Model

Two different points of view.

- Real Model – multiprogramming (rapidly switching back and forth)
- Conceptual Model – each process has its own virtual CPU, PC, registers, stack pointer
The Process Model

(a) One program counter

(b) Four program counters

(c) Process switch

Time
The Process Model

- With the CPU switching back and forth among the processes, its computation time will not be uniform.
- The computation time is also based on the process type – I/O bounded, CPU bounded.
Process Creation

Process Creation

- **System initialization** - when an OS is booted, typically several processes are created.
  - Foreground processes
  - Background processes (daemons)
- **Execution of a process creation system call by a running process** – a running process issues system calls to create one or more new processes.
- **A user creates a new process** – in interactive systems, users can start a program by typing a command or (double) clicking an icon.
- **Initiation of a batch job** – mainframe computer
Process Creation

- UNIX – system call **fork** creates a new process.
- In Window – system call **CreateProcess** creates a new process.
- Once a child process is created, both the parent and child have their own distinct address spaces.
A process terminate due to one of the following condition

- **Normal exit** – done with their work (voluntary).
- **Error exit** – the process discovers a fatal error – compiler tries to compile a program, there is no such a file (voluntary).
- **Fatal error** – error caused by the processor – a process tries to modify the memory location where other process is located (involuntary).
- **Killed by another process** – a deadlock has occurred. Kill the process call, resolve the deadlock (involuntary).
Process State

1. Process blocks for input
2. Scheduler picks another process
3. Scheduler picks this process
4. Input becomes available
To implement the process, operating system maintain each process’s information in its process table (or process control block PCB) when a running process’s state changes to block state or ready state, OS needs to save information for the process. When a process state changes to running state, OS needs to restore information for the process (from its process table) into CPU.
Process Implementation

Contents of a Process Table (or PCB)

- Process status – ready, running, blocked
- Program counter – address of next instructions for the process
- CPU registers – registers vary in number and type, depending on the computer architecture.
- CPU scheduling information – a process priority, pointers to the scheduling queues.
- Memory management information – the base and limit registers, the page tables, the segmentation tables.
- Accounting information – the amount of CPU and real time used, time limits, account numbers, process numbers,…
- I/O status information – list of I/O devices allocated to the process, list of open files, and so on …