Scheduling Policies

Lecture 13

Klara Nahrstedt

CS241 Administrative

- Read Chapter 9. 1 and 9.2 Stallings
- SMP3 is ON this week (2/12-2/19)

Content of This Lecture

- Why CPU Scheduling?
- Basic scheduling algorithms
  - FIFO (FCFS)
  - Shortest job first
  - Round Robin
  - Priority Scheduling

- Goals:
  - Understand how your program is executed on the machine together with other programs
  - SMP4

Review: State Process Model

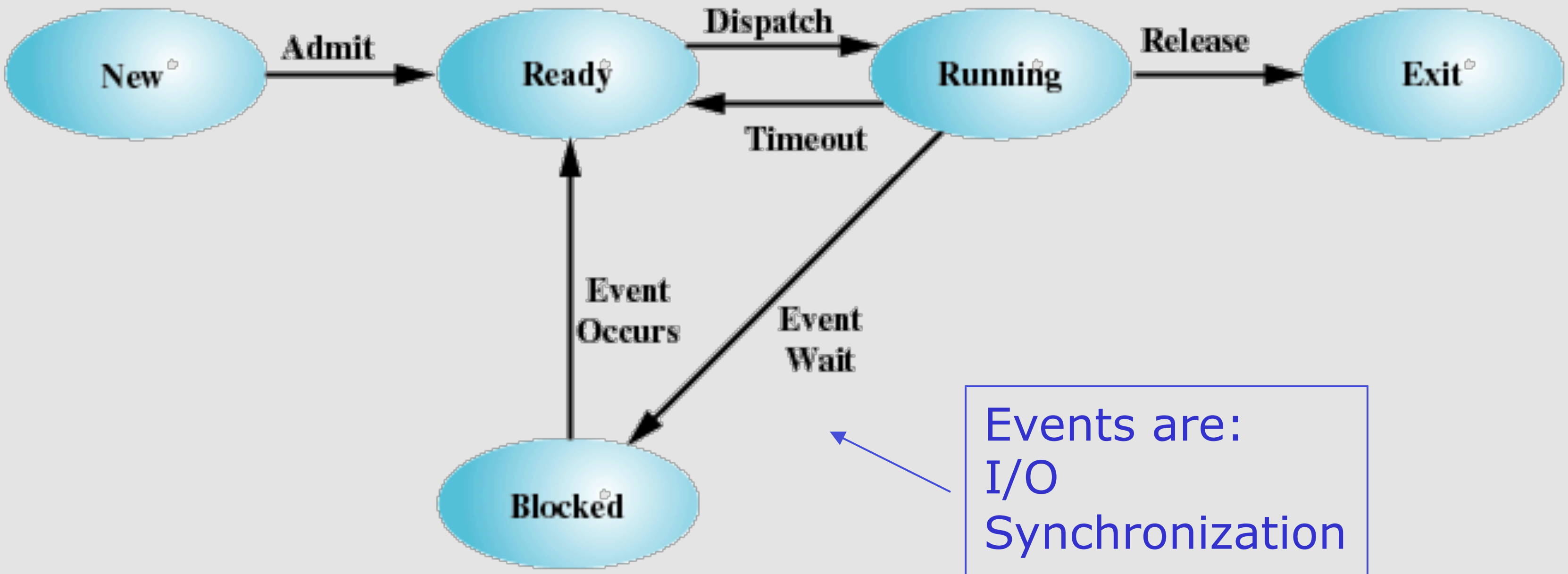


Figure 3.6 Five-State Process Model

Events are:
I/O
Synchronization

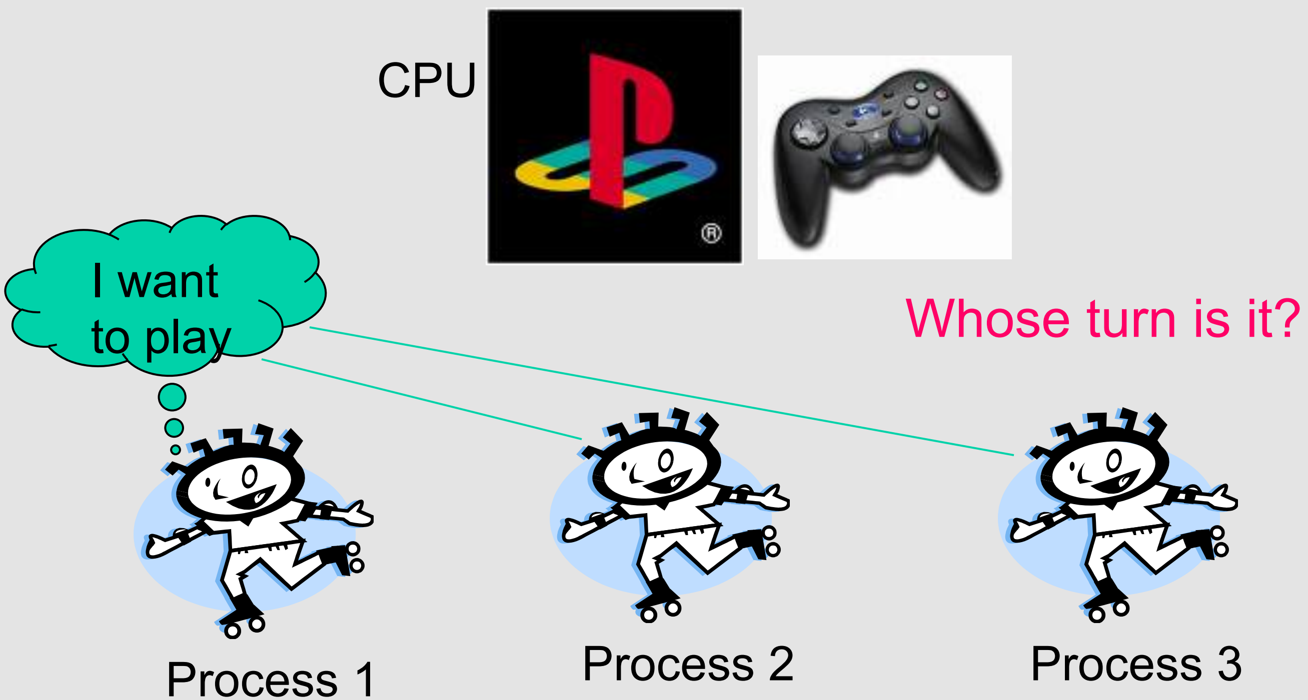
OS Representation of Process via Process Control Block (PCB)

Process management	**Memory management**	**File management**
Registers	Pointer to text segment	Root directory
Program counter	Pointer to data segment	Working directory
Program status word	Pointer to stack segment	File descriptors
Stack pointer		User ID
Process state		Group ID
Priority		
Scheduling parameters		
Process ID		
Parent process		
Process group		
Signals		
Time when process started		
CPU time used		
Children's CPU time		
Time of next alarm		

Fields of a process table entry

Process Scheduling

Deciding which process/thread should occupy the resource (CPU, disk, etc)



Process Scheduling

Objective of multiprogramming – maximal CPU utilization, i.e., have always a process running

Objective of time-sharing – switch CPU among processes frequently enough so that users can interact with a program which is running

Need:

Scheduling Mechanisms:

- Context Switching between Processes
- Queueing Capabilities

Scheduling Policies:

- Which process to select for running at CPU ?

Context Switch

Switch CPU from one process to another

Performed by scheduler

It includes:

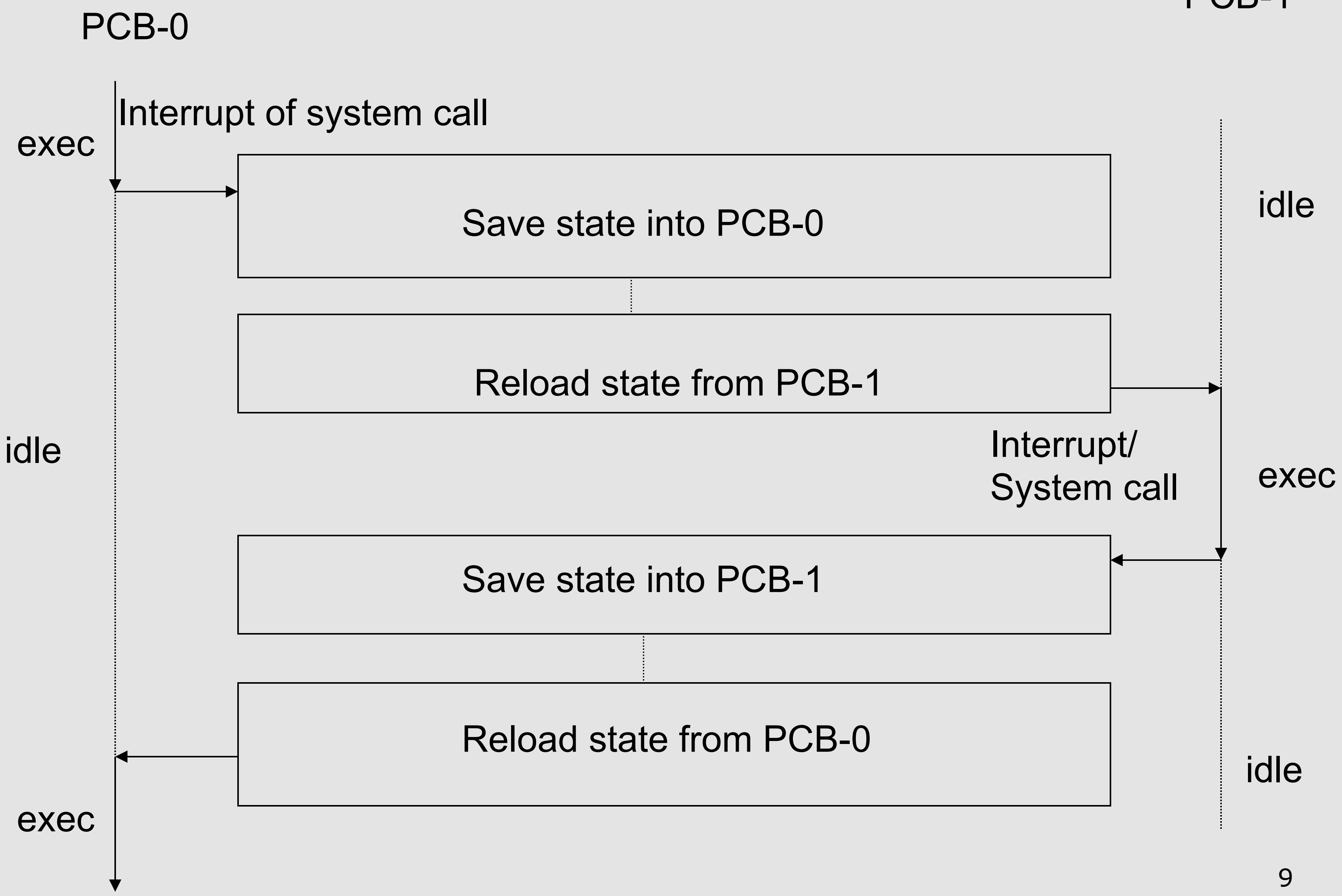
- save PCB state of the old process;
- load PCB state of the new process;
- Flush memory cache;
- Change memory mapping (TLB);

Context switch is expensive (1-1000 microseconds)

- No useful work is done (pure overhead)
- Can become a bottleneck
- Real life analogy?

Need hardware support

Process Context Switch



When to schedule?

A new process starts

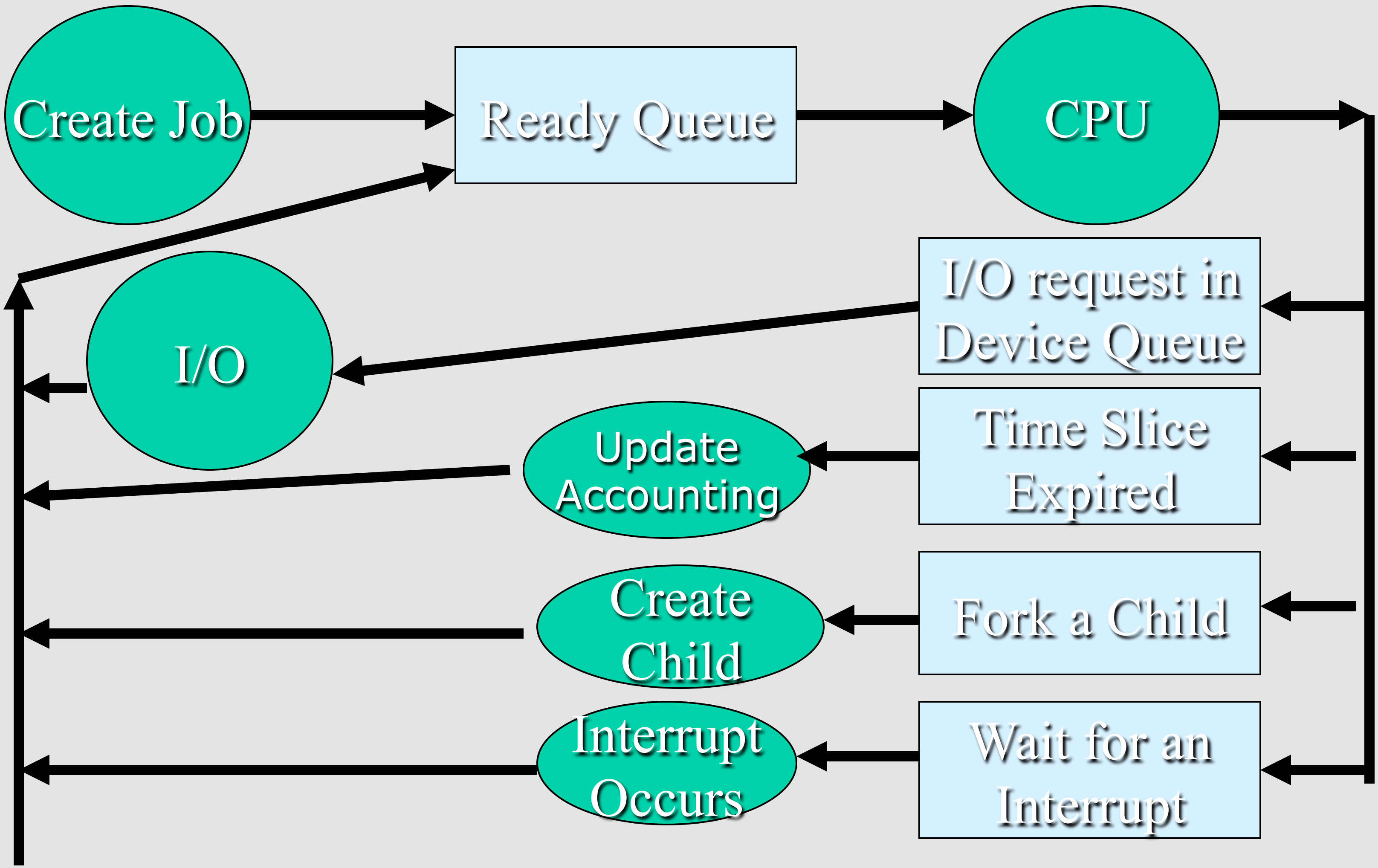
The running process exits

The running process is blocked

I/O interrupt (some processes will be ready)

Clock interrupt (every 10 milliseconds)

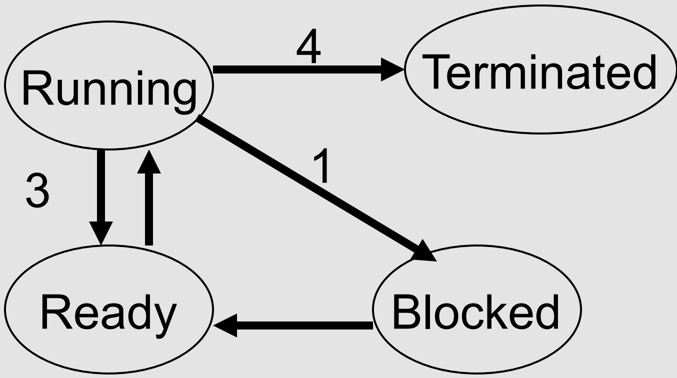
Queuing Diagram for Processes



Preemptive vs. Non-preemptive

Non-preemptive scheduling:

- The running process keeps the CPU until it voluntarily gives up the CPU
  - process exits
  - switches to blocked state
  - 1and 4 only (no 3)



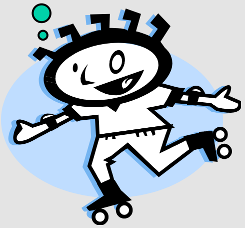
Preemptive scheduling:

- The running process can be interrupted and must release the CPU (can be forced to give up CPU)

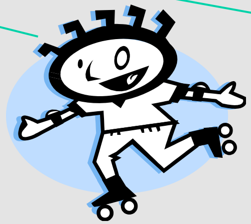
What are the scheduling objectives?

I want to ride it

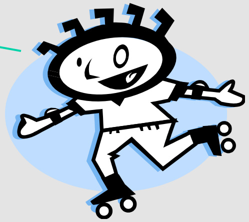
Whose turn is it?



Process 1



Process 2



Process 3

Scheduling Objectives

Fair (nobody cries)

Priority (lady first)

Efficiency (make best use of equipment)

Encourage good behavior (good boy/girl)

Support heavy loads (degrade gracefully)

Adapt to different environments (interactive, real-time, multi-media)

Performance Criteria

Fairness

Efficiency: keep resources as busy as possible

Throughput: # of processes that completes in unit time

Turnaround Time (also called elapse time)

- amount of time to execute a particular process from the time its entered

Waiting Time

- amount of time process has been waiting in ready queue

Response Time

- amount of time from when a request was first submitted until first response is produced.
- predictability and variance

Policy Enforcement:

- seeing that stated policy is carried out

Proportionality:

- meet users' expectation

Meeting Deadlines: avoid losing data

Process Profiles

I/O – Bound

- Does too much I/O to keep CPU busy

CPU – Bound

- Does too much computation to keep I/O busy

Process Mix

- Scheduling should load balance between I/O bound and CPU-bound processes
- Ideal would be to run all equipment at 100% utilization but that would not necessarily be good for response time

CPU Scheduler

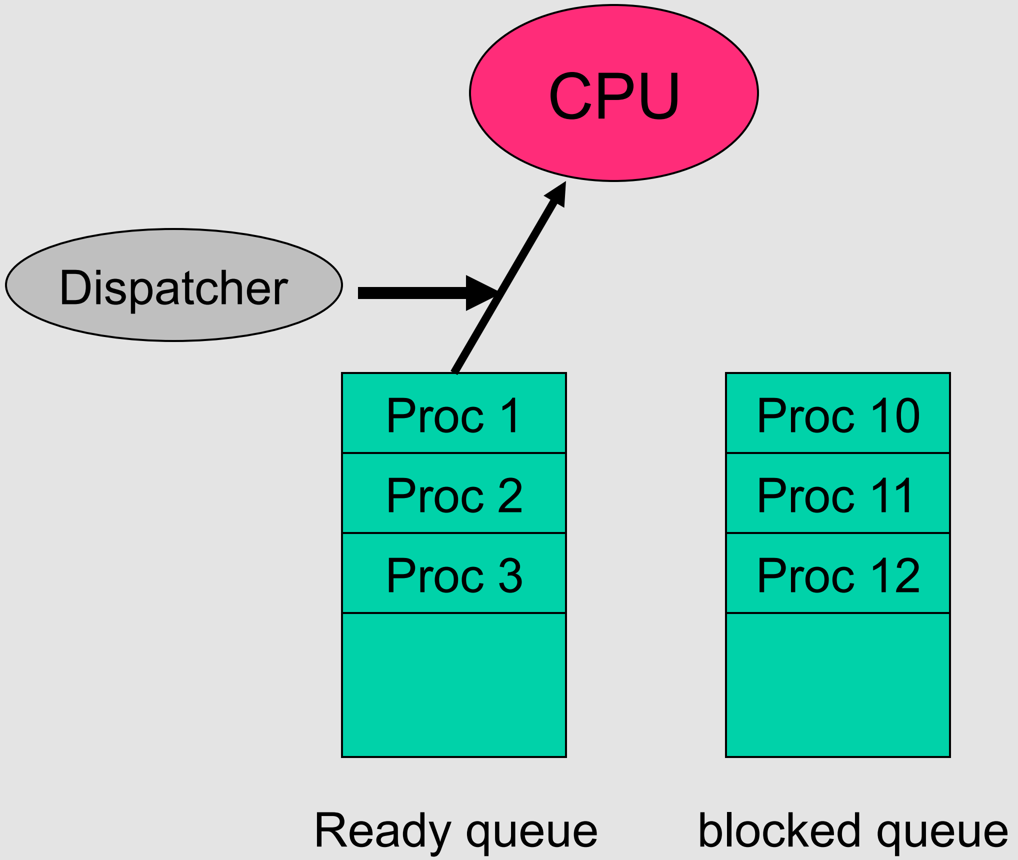
Proc 1: 14 time units

Proc2: 8 time units

Proc3: 8 time units

Dispatcher

Preemptive vs.
 non-preemptive



Simple Processor Scheduling Algorithms

- Batch systems
  - First Come First Serve (FCFS)
  - Shortest Job First
- Interactive Systems
  - Round Robin
  - Priority Scheduling
  - …

First Come First Serve (FCFS)

Process that requests the CPU FIRST is allocated the CPU FIRST.

Also called FIFO

Preemptive or Non-preemptive?

Used in Batch Systems

Real life analogy?

- Fast food restaurant

Implementation

- FIFO queues
- A new process enters the tail of the queue
- The schedule selects from the head of the queue.

Performance Metric: **Average Waiting Time**.

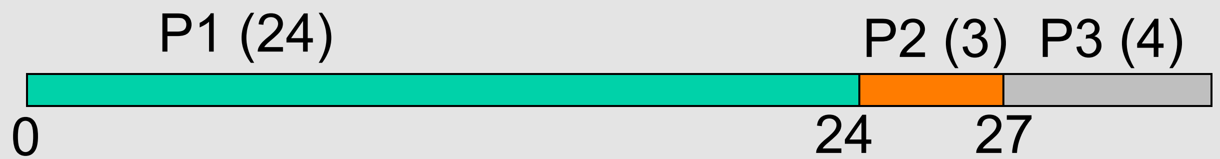
Given Parameters:

- Burst Time (in ms), Arrival Time and Order

FCFS Example

Process	Duration	Order	Arrival Time
P1	24	1	0
P2	3	2	0
P3	4	3	0

The final schedule:



P1 waiting time: 0
P2 waiting time: 24
P3 waiting time: 27

The average waiting time:
(0+24+27)/3 = 17

What if P1 arrives at time 2

Problems with FCFS

- Non-preemptive
- Not optimal AWT
- Cannot utilize resources in parallel:
  - Assume 1 process CPU bounded and many I/O bounded processes
  - result: Convoy effect, low CPU and I/O Device utilization
  - Why?

Summary

Why Scheduling?

Scheduling objectives

Scheduling Algorithms

- FCFS (FIFO)