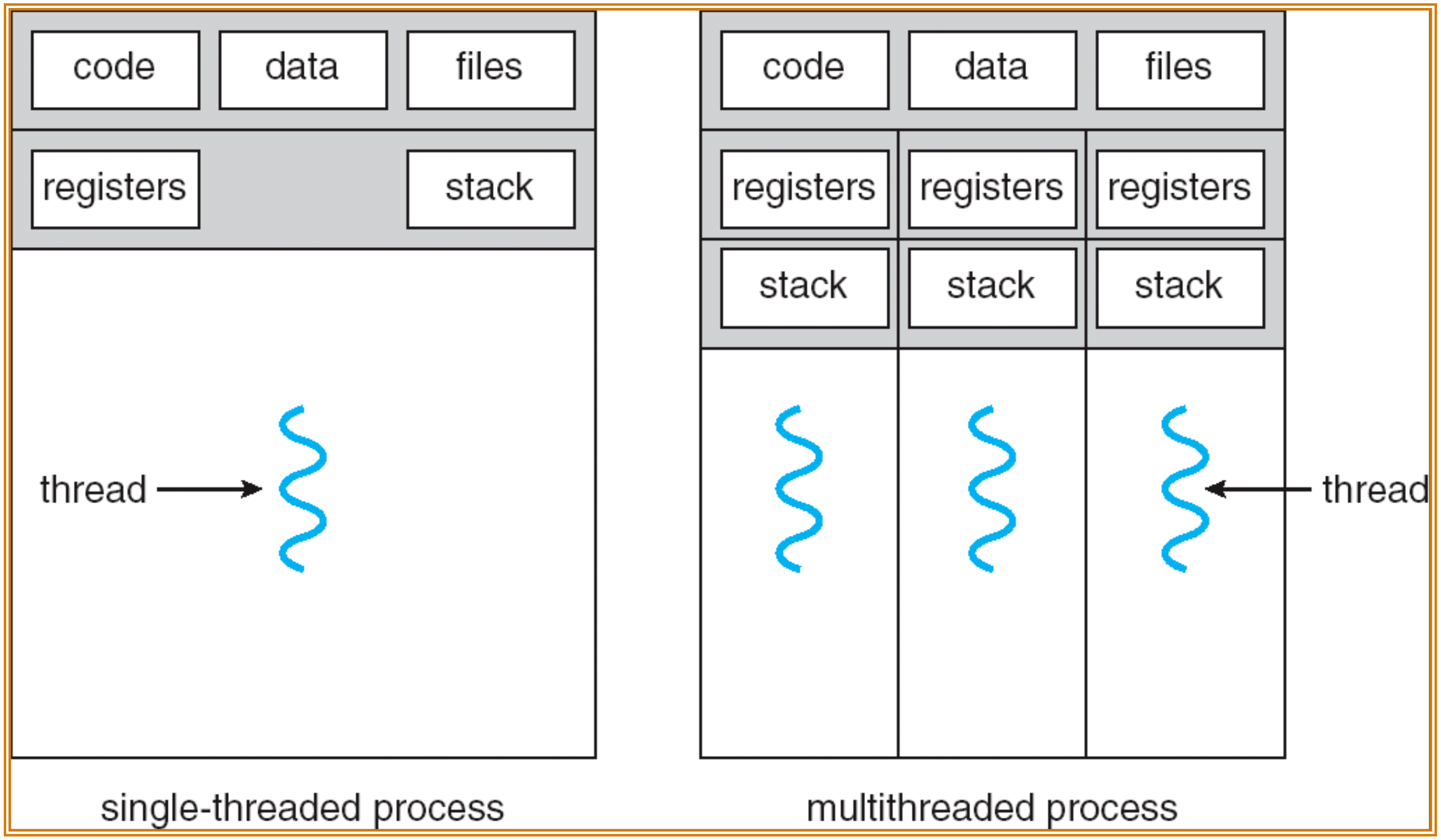
So far….

- Firmware identifies hardware devices present
- OS bootstrap process: uses the list created by firmware and loads driver modules for each detected hardware. Initializes internal data structures (PCB, device queue for each device)
- Each process can have one or more threads
- Processes can be in Wait (for resources), Ready (waiting for processor) and Run states.
- Next: scheduling next process from Wait to Run

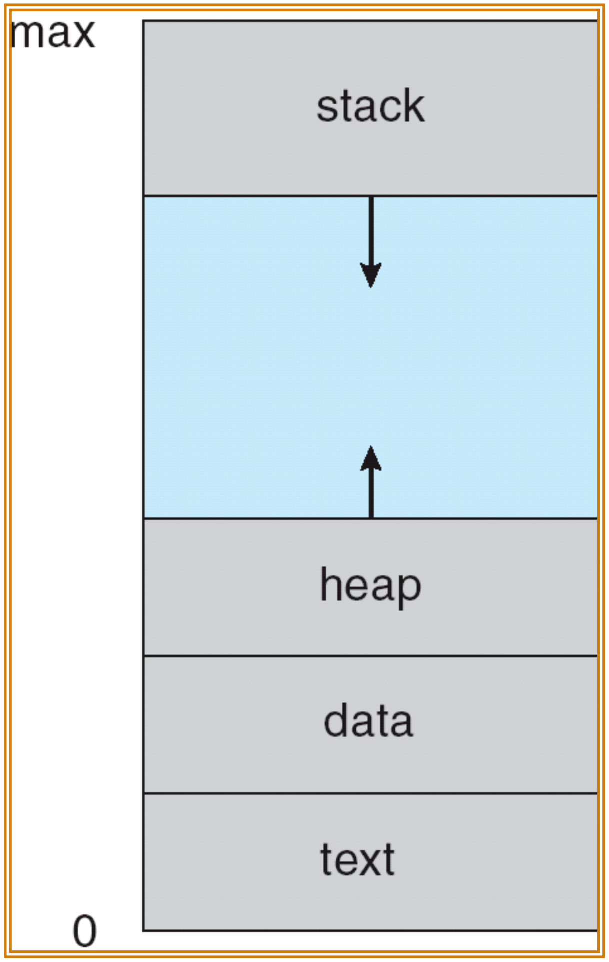
Clarification re: multi-threaded process



From Lecture 5

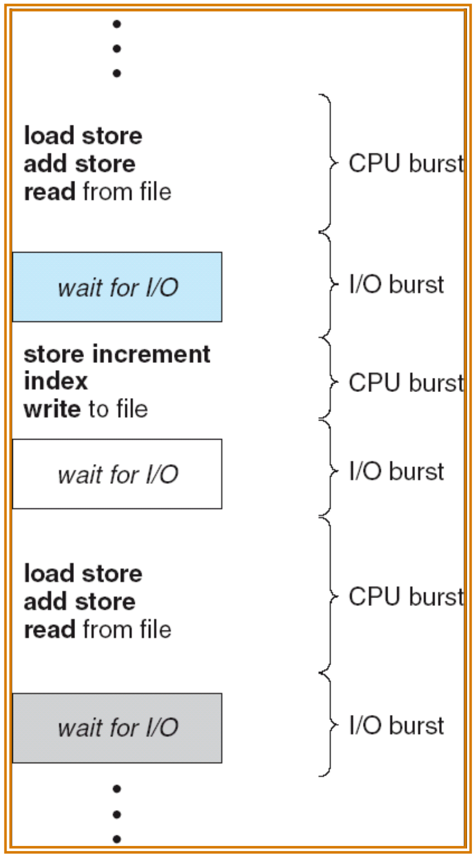
From Lecture 4

- Local variables are stored in stack
- Malloc() memory is stored in heap
- Compilers create and manage these locations. They request a certain amount of stack during program loading from the OS
  - The specific format depends on the program structure (e.g. ELF)



Scheduling basics

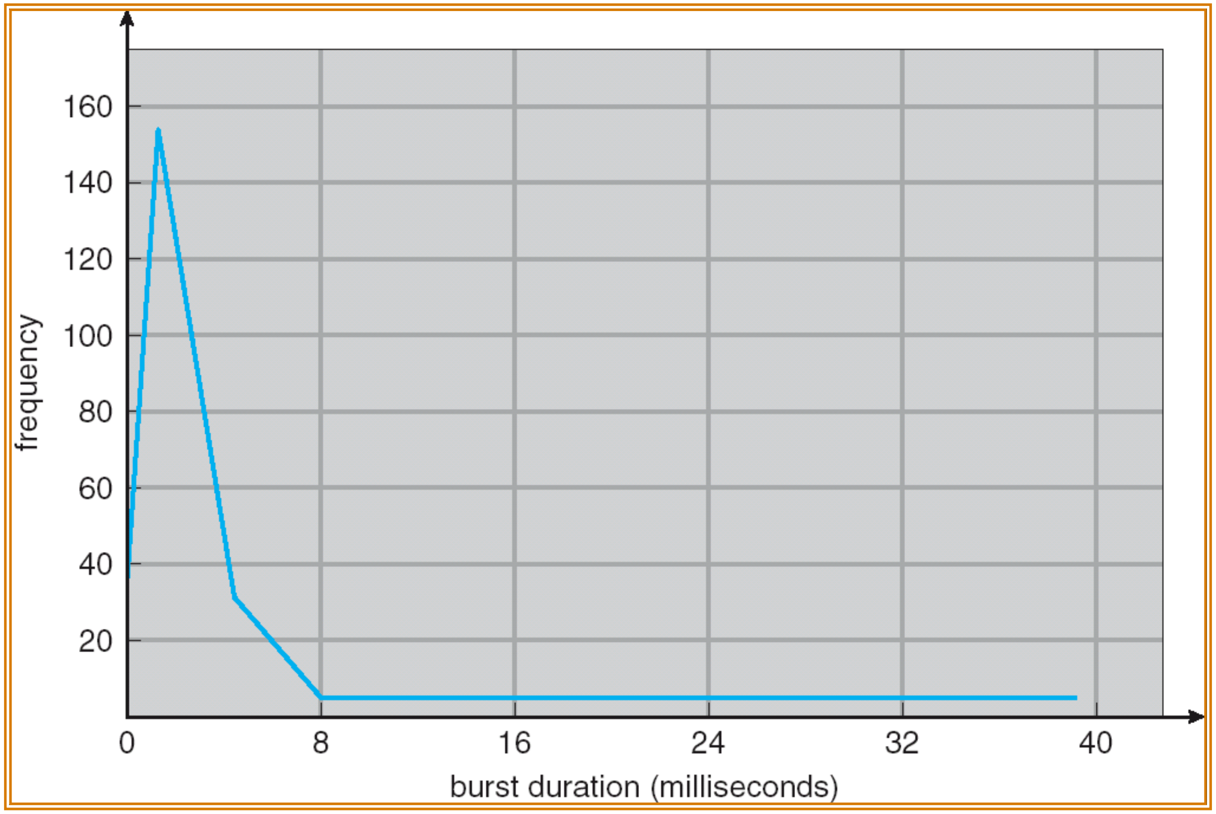
CPU–I/O Burst Cycle – Process execution consists of a cycle of CPU execution and I/O wait



- CPU scheduling depends on the observation that processes cycle between CPU execution and I/O wait.

Histogram of CPU-burst Times

Typical CPU-burst duration



CPU bursts are short lived

CPU Scheduler

- Selects from among the processes in memory that are ready to execute, and allocates the CPU to one of them
- CPU scheduling decisions may take place when a process:
  1. Switches from running to waiting state (e.g. I/O request)
  2. Switches from running to ready state (e.g. Interrupt)
  3. Switches from waiting to ready (e.g. I/O completion)
  4. Terminates
- Scheduling under 1 and 4 is *non-preemptive (cooperative)*
- All other scheduling is *preemptive* - have to deal with possibility that operations (system calls) may be incomplete

Dispatcher

- Dispatcher module gives control of the CPU to the process selected by the short-term scheduler; this involves:
  - switching context
  - switching to user mode
  - jumping to the proper location in the user program to restart that program
- Dispatch latency – time it takes for the dispatcher to stop one process and start another running
  - Should be as low as possible

Scheduling Criteria

- CPU utilization (max) – keep the CPU as busy as possible
- Throughput (max) – # of processes that complete their execution per time unit
- Turnaround time (min) – amount of time to execute a particular process
- Waiting time (min) – amount of time a process has been waiting in the ready queue
- Response time (min) – amount of time it takes from when a request was submitted until the first response is produced, not output (for time-sharing environment)
- In typical OS, we optimize each to various degrees depending on what we are optimizing the OS

Optimization criteria

- Max CPU utilization
- Max throughput
- Min turnaround time
- Min waiting time
- Min response time

- Analysis using Gantt chart (illustrates when processes complete)

First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_1 , P_2 , P_3
  The Gantt Chart for the schedule is:

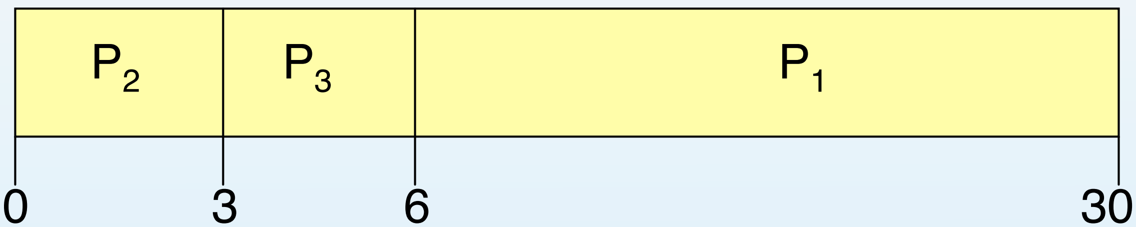
- Waiting time for $P_1 = 0$; $P_2 = 24$; $P_3 = 27$
- Average waiting time: (0 + 24 + 27)/3 = 17

FCFS Scheduling (Cont.)

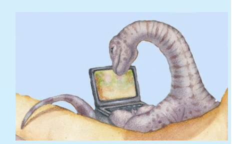
- Suppose that the processes arrive in the order

  P2 , P3 , P1

- The Gantt chart for the schedule is:



- Waiting time for P1 = 6; P2 = 0; P3 = 3
- Average waiting time: (6 + 0 + 3)/3 = 3
- Much better than previous case
- Convoy effect short process behind long process



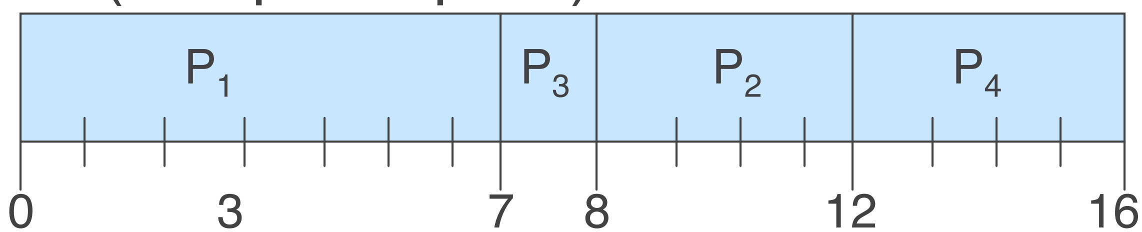
Shortest-Job-First (SJR) Scheduling

- Associate with each process the length of its next CPU burst. Use these lengths to schedule the process with the shortest time
- Two schemes:
  - nonpreemptive – once CPU given to the process, it cannot be preempted until completes its CPU burst
  - preemptive – if a new process arrives with CPU burst length less than remaining time of current executing process, preempt. This scheme is know as the Shortest-Remaining-Time-First (SRTF)
- SJF is optimal – gives minimum average waiting time for a given set of processes

Example of Non-Preemptive SJF

Process	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (non-preemptive)

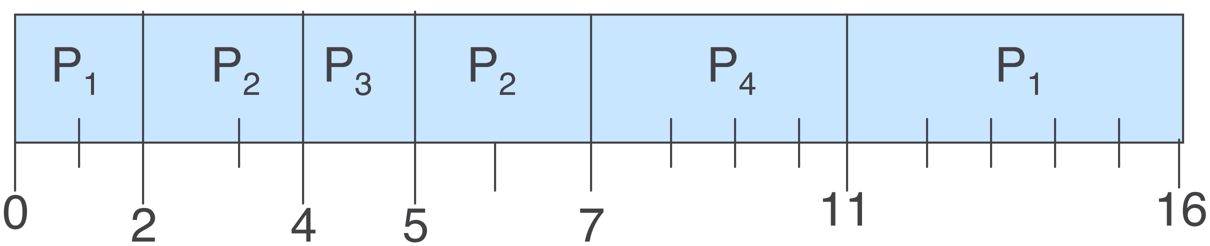


- Average waiting time = (0 + 6 + 3 + 7)/4 = 4

Example of Preemptive SJF

Process	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (preemptive)



- Average waiting time = (9 + 1 + 0 +2)/4 = 3