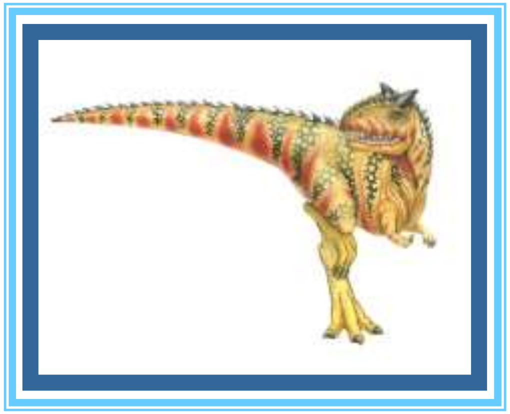
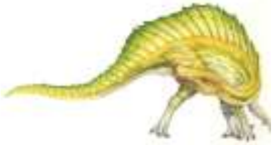
Chapter 4: Threads



Chapter 4: Threads

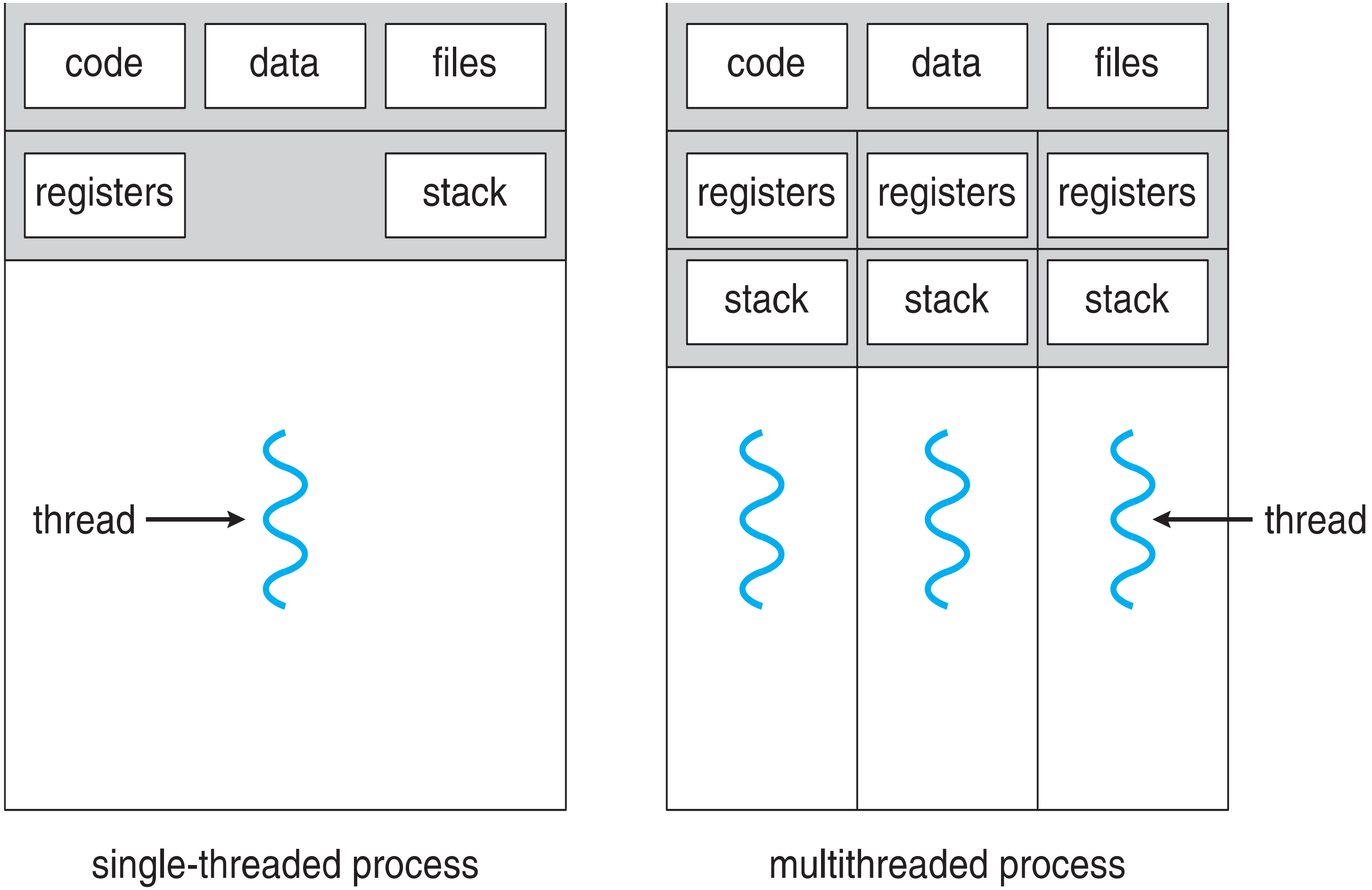
- Overview
- Multicore Programming
- Multithreading Models
- Thread Libraries



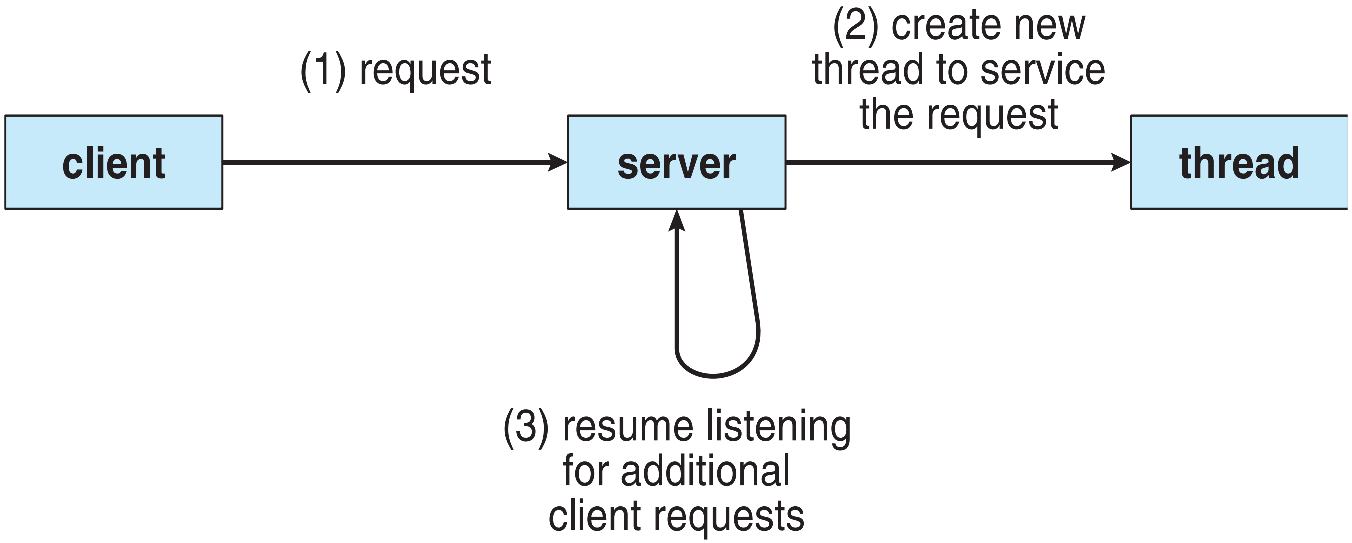
Motivation

- Most modern applications are multithreaded
- **Threads** run within application
- Thread—a fundamental unit of CPU utilization that forms the basis of multithreaded computer systems
- Multiple tasks with the application can be implemented by separate threads
  - Update display
  - Fetch data
  - Spell checking
  - Answer a network request
- Process creation is **heavy-weight** while thread creation is **light-weight**
- Can **simplify code**, **increase efficiency**
- Kernels are generally multithreaded

Single and Multithreaded Processes



Multithreaded Server Architecture


(1) request
(2) create new thread to service the request
client
server
thread
(3) resume listening for additional client requests

Benefits

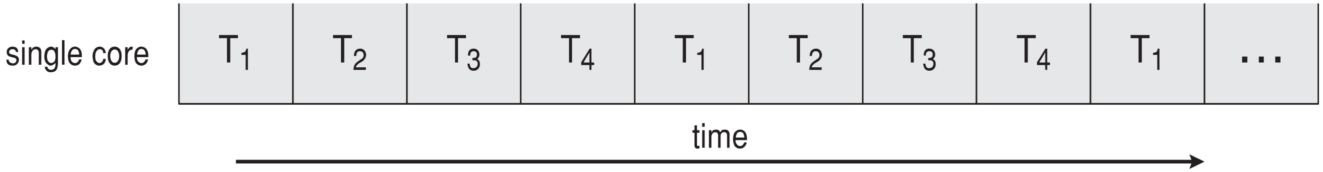
- **Responsiveness –** may allow continued execution if part of process is blocked, especially important for user interfaces
- **Resource Sharing –** threads share resources of process, easier than shared memory or message passing
- **Economy –** cheaper than process creation, thread switching lower overhead than context switching
- **Scalability –** process can take advantage of multiprocessor architectures

Multicore Programming

- **Parallelism** implies a system can perform more than one task simultaneously
- **Concurrency** supports more than one task making progress
  - Single processor / core, scheduler providing concurrency

Concurrency vs. Parallelism

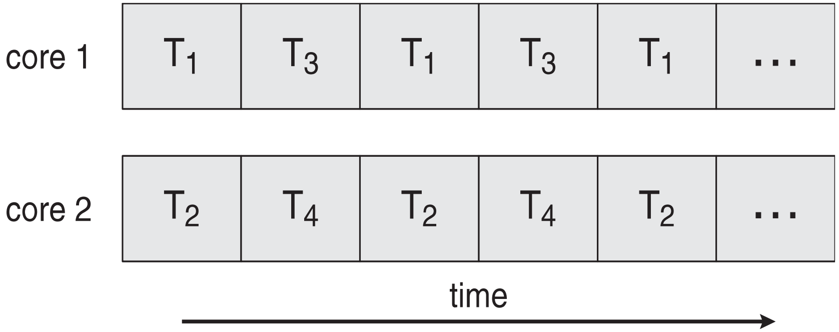
- **Concurrent execution on single-core system:**



single core | T_1 | T_2 | T_3 | T_4 | T_1 | T_2 | T_3 | T_4 | T_1 | . . .

time

- **Parallelism on a multi-core system:**



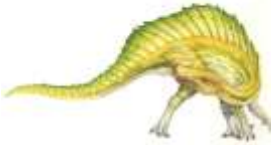
core 1 | T_1 | T_3 | T_1 | T_3 | T_1 | . . .

core 2 | T_2 | T_4 | T_2 | T_4 | T_2 | . . .

time

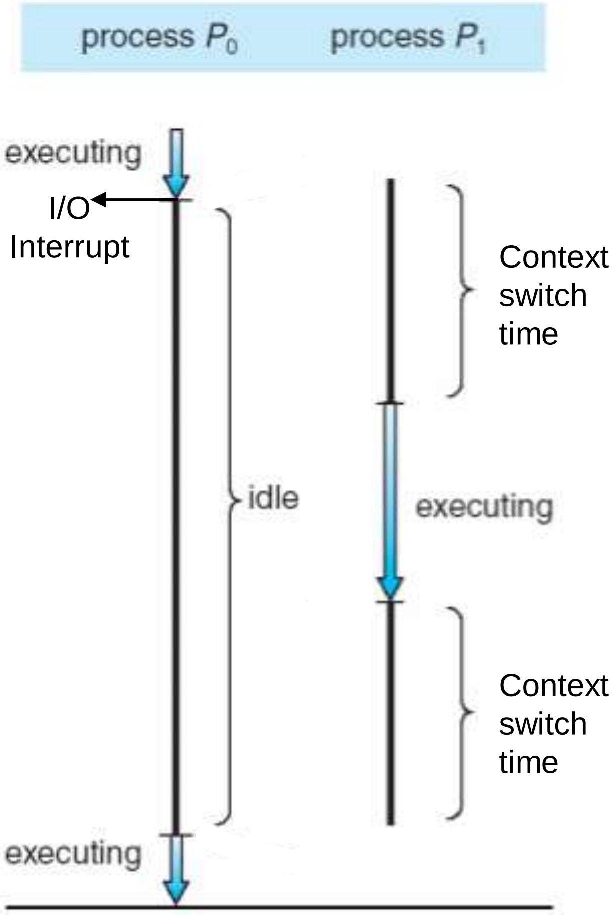
Multicore Programming (Cont.)

- Types of parallelism
  - **Data parallelism** – distributes subsets of the same data across multiple cores, same operation on each
  - **Task parallelism** – distributing threads across cores, each thread performing unique operation



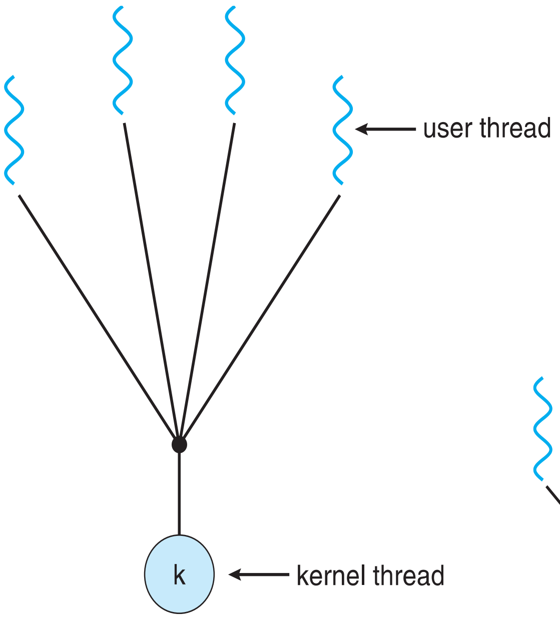
Multithreading in single-core machines

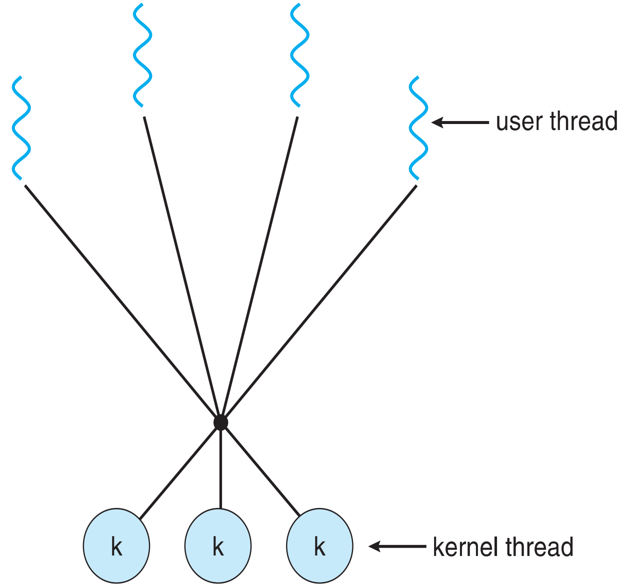
- P0 has requested some data from the disk, so it invoked a system call.
- During this time, the CPU is idle and not used.
- If this idle time > 2* context switch time to P1:
  - Switching to P1 is useful.
  - Otherwise, it is not worth it.
- Context switch time for thread < context switch for process
  - Because threads share the same address space, it takes less time to save, reload thread context.
  - So idle time can be used more efficiently.

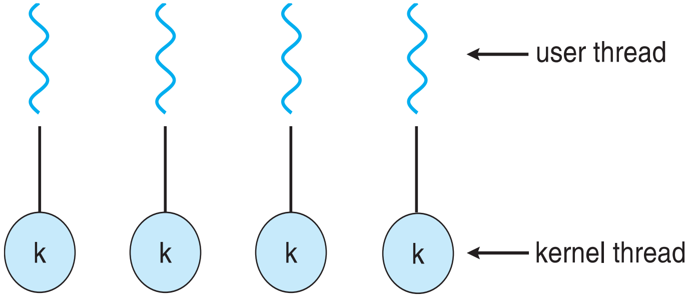


Multithreading Models

- Many-to-One
- One-to-One
- Many-to-Many







Thread Libraries

- **Thread library** provides programmer with API for creating and managing threads

- **Pthreads**
- **Java threads** are managed by the JVM
- Thread Pools
- **OpenMP:** It is a set of compiler directives and an API for C, C++, FORTRAN
- Grand Central Dispatch

End of Chapter 4