Chapter 4: Threads

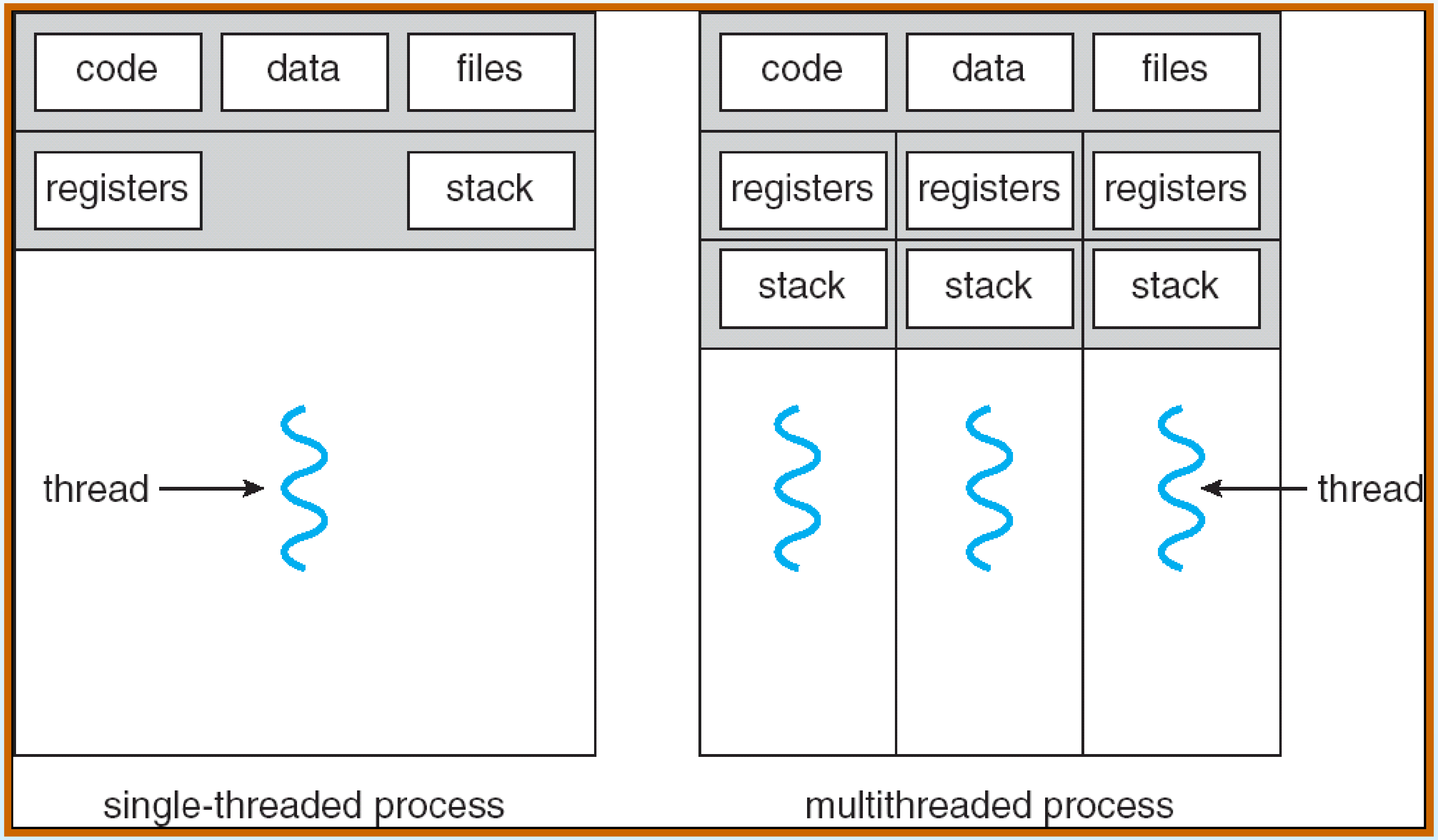
Chapter 4: Threads

- Overview
- Multithreading Models
- POSIX Pthreads
- Windows XP Threads
- Java Threads
- Threading Issues
- Implementation of LinuxThreads

A Thread

- Has
  - Thread ID
  - Program counter
  - Register set
  - Stack
- Shares with other threads belonging to the same process
  - Code section
  - Data section
  - Other resources
    - Such as open files, signals

Single and Multithreaded Processes



Benefits

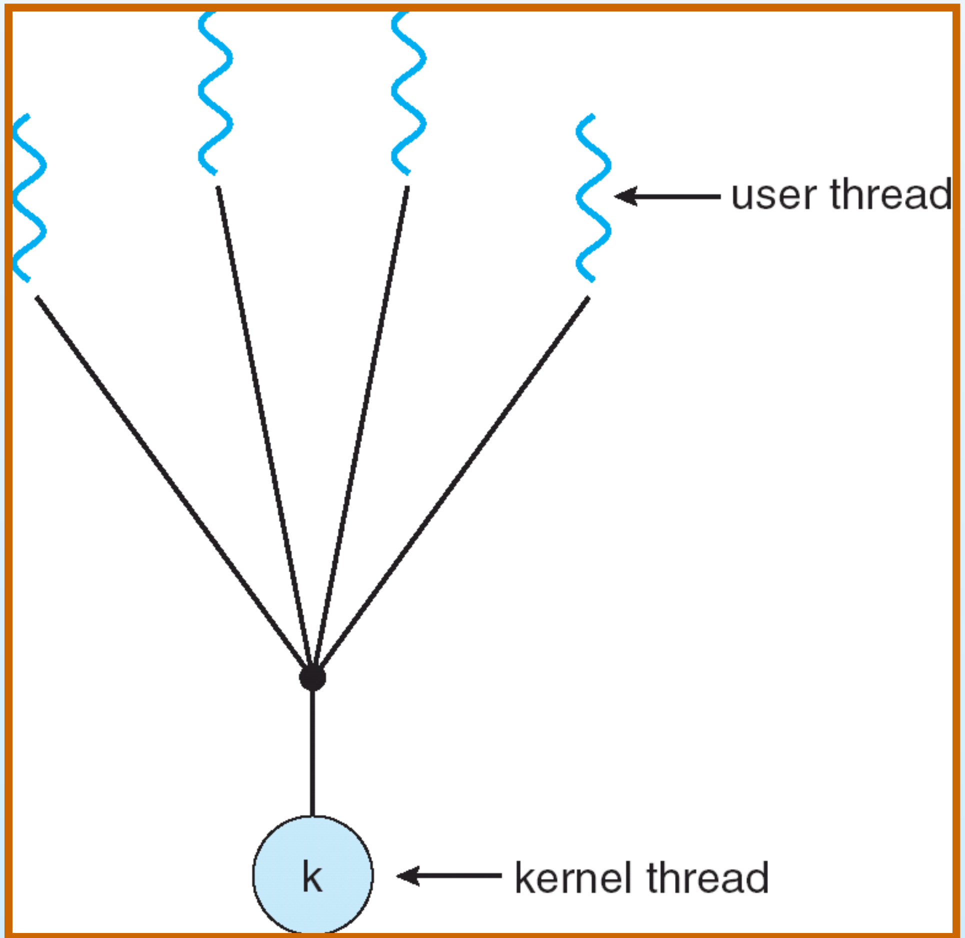
- Responsiveness
- Resource Sharing
- Economy
- Utilization of Multiprocessor Architectures

Threads

- User threads
  - Thread management done by user-level threads library
- Kernel threads
  - Supported by the Kernel
  - Examples
    - Windows XP/2000
    - Solaris
    - Linux
    - Tru64 UNIX
    - Mac OS X

User Threads

- Many user-level threads mapped to single kernel thread



User Threads

- Good
  - Efficient
- Bad
  - One blocked system call blocks all threads
  - Unable to run in parallel on multiprocessors
- Examples:
  - Solaris Green Threads
  - GNU Portable Threads

Thread Libraries

- POSIX Pthreads
- Win32 threads
- Java threads

Pthreads

- A POSIX standard (IEEE 1003.1c) API
- Common in UNIX operating systems (Solaris, Linux, Mac OS X)
- API:
  - pthread_create()
  - pthread_exit()
  - ……

Windows XP Threads

- API:
  - CreateThread()
  - ExitThread()
  - ……

Java Threads

- Java threads are managed by the JVM
  - Use or not use the thread library provided by OS
- Java threads may be created by:
  - Extending Thread class
  - Implementing the Runnable interface

Threading Issues

- Thread cancellation
- Thread pools
- Thread specific data
- Scheduler activations

Thread Cancellation

- Terminating a thread before it has finished
- Two general approaches:
  - Terminates the target thread immediately
  - The target thread periodically checks if it should be cancelled

Thread Pools

- Create a number of threads in a pool where they await work
- Advantages:
  - Usually slightly faster to service a request with an existing thread than create a new thread
  - Allows the number of threads in the application(s) to be bound to the size of the pool

Thread Specific Data

- Allows each thread to have its own copy of data
- Win32, Pthreads and java provide support

Implementation of LinuxThreads

- LinuxThreads is implemented by `clone()` system call
  - `fork()` is implemented by `clone()` too
  - So Linux calls both of process and thread *task*
  - Linux threads are also called Light-Weight Process
- `clone()`'s flags:
  - CLONE_FS: file-system information is shared
  - CLONE_FILES: the set of open files is shared
  - CLONE_VM: The same memory space is shared
- Thanks to `struct task_struct` who use pointers instead of storing data

End of Chapter 4