CSCI 3753
Operating Systems

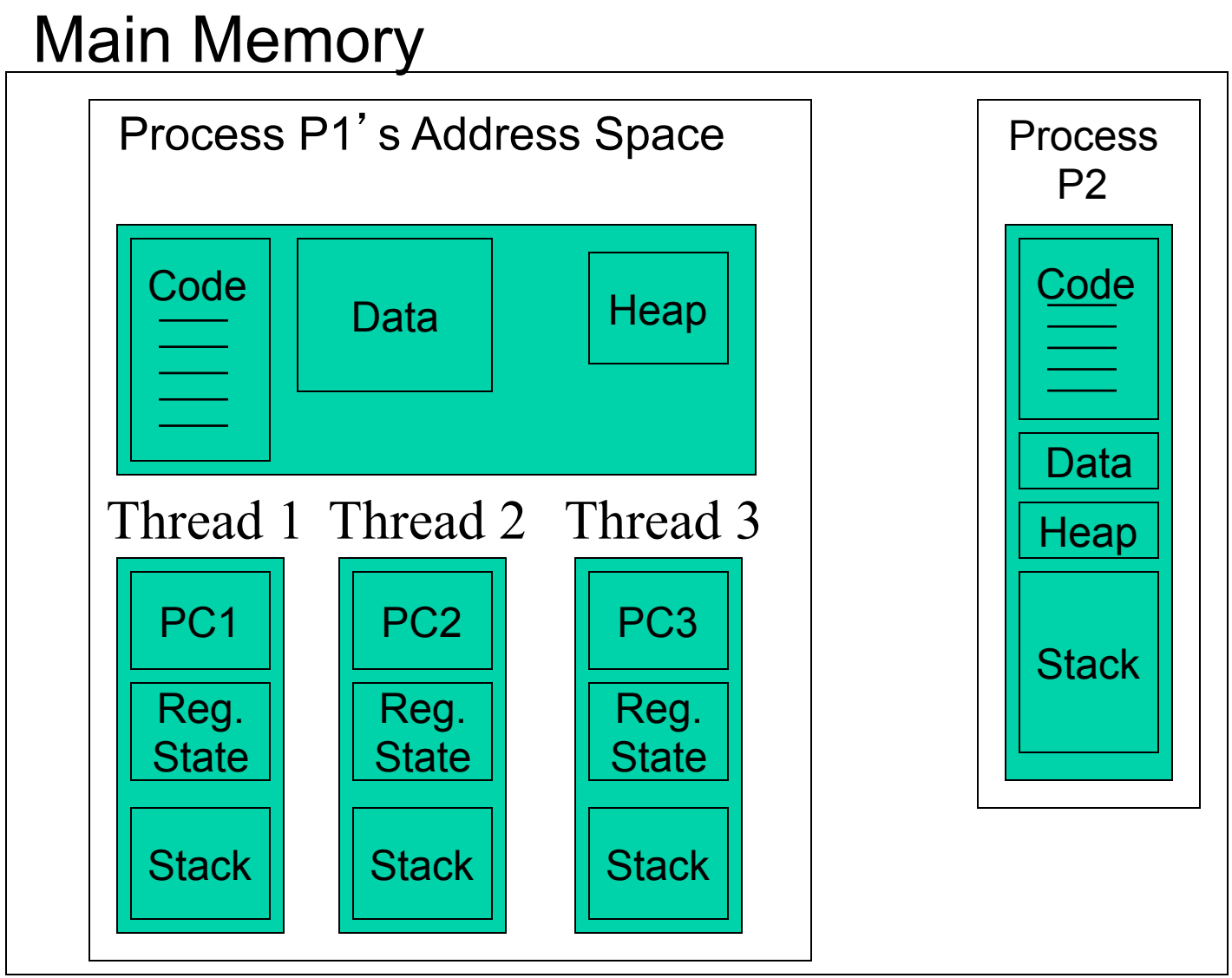
Threads

Lecture Notes By
Shivakant Mishra
Computer Science, CU-Boulder
Last Update: 02/07/14

Threads

- A thread is a logical flow or unit of execution that runs within the context of a process
  - has its own program counter (PC), register state, and stack
  - shares the memory address space with other threads in the same process
    - share the same code and data and resources (e.g. open files)
  - Thread are also called *lightweight* processes

Threads



- Process P1 is *multithreaded*
- Process P2 is single threaded
- The OS is *multiprogrammed*
- If there is preemptive timeslicing, the system is *multitasked*

Applications, Processes, and Threads

- Application = Σ_i $Processes_i$
  - e.g. a server could be split into multiple processes, each one dedicated to a specific task (UI, computation, communication, etc.)
- $Process_j$ = Σ $Threads_j$
  - e.g. a Web server process could spawn a thread to handle each new http request for a Web page
- An application could thus consist of many processes and threads

Threads

- Why would you want multithreaded processes?
  - Reduced context switch overhead
    - In Solaris, context switching between processes is 5x slower than switching between threads
    - Don't have to save/restore context, including base and limit registers and other MMU registers, also TLB cache doesn't have to be flushed
  - Shared resources => less memory consumption => more threads can be supported, especially for a scalable system, e.g. Web server must handle thousands of connections
  - Inter-thread communication is easier and faster than inter-process communication – threads share the same memory space, so just read/write from/to the same memory location!

- Threads allow interactive programs to be more responsive, especially useful for designing user interfaces

User-Space & Kernel Threads

- User-space threads are created without any kernel support
- User space threads are usually cooperatively multitasked, i.e. user threads within a process voluntarily give up the CPU to each other
- OS is unaware of user-space threads – only sees user-space processes
- User-space thread library
  - provides interface to create, delete threads in the same process
  - If one user space thread blocks, the entire process blocks in a many-to-one scenario (see text)
- User space thread also called a *fiber*

User-Space & Kernel Threads (2)

- *Kernel threads* are supported by the OS
- kernel sees threads and schedules at the granularity of threads
- Most modern OSs like Linux, Mac OS X, Win XP support kernel threads
- Mapping of user-level threads to kernel threads is usually one-to-one, e.g. Linux and Windows, but could be many-to-one, or many-to-many
- Win32 thread library is a kernel-level thread library

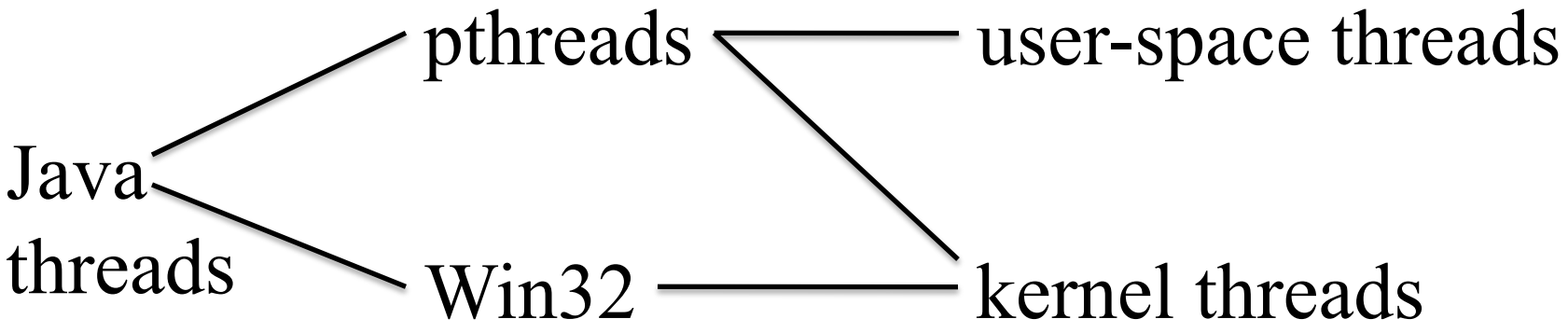
Pthreads

- *pthreads* is a POSIX threading API
- implementations of pthreads API differ underneath the API; could be user space threads; there is also pthreads support for kernel threads as well

- Details in recitation

User-Space & Kernel Threads (3)

- Java thread library is running in Java VM on top of host OS, so on Windows it's implemented on top of Win32 threading, while on Linux/Unix it's implemented on top of pthreads
- Possible scenarios:



Thread Pool

- Potential problems with threads
  - Time to create a thread is still significant
  - If there is no bound on the # of concurrent threads, they may exhaust resources – CPU and memory
- Thread Pool
  - Create a number of threads at process startup. These threads remain blocked waiting for work
  - When a thread is needed, e.g. a server receives a new request, a thread in the pool is woken up to service that request
  - After the thread completes the service, it again blocks, waiting for new work

Thread Pool

- If no thread is currently available, the server waits until one becomes available
- Benefits
  - Servicing a request is faster – no need to create new thread
  - Bound on the maximum number of concurrent threads
- Threads also have some issues: Please read Section 4.6 (page 183) from the textbook