CS343 - Operating Systems

Module-2E

Introduction to Threads

Dr. John Jose

Assistant Professor

Department of Computer Science & Engineering

Indian Institute of Technology Guwahati, Assam.

http://www.iitg.ac.in/johnjose/

Session Outline

- ❖ Process vs Threads
- ❖ Thread model
- ❖ Multithreaded programs
- ❖ User and Kernel threads
- ❖ Multithread mapping models

Concept of Threads

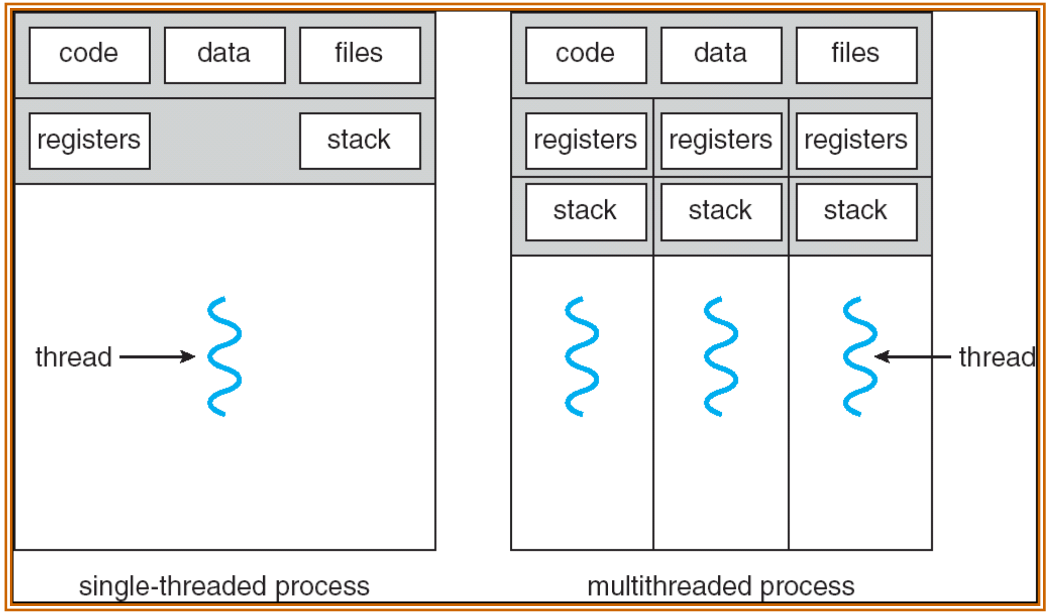
- Thread is a flow of control within a process.
  - single-threaded process, multi-threaded process.
- It is a basic unit of CPU utilization, which comprise
  - a thread ID, program counter, register set, stack.
- Shares with other threads belonging to the same process its code section, data section, and other OS resources (open files and signal)
- If a process has multiple threads of control, it can perform more than one task at a time.

The Thread Model

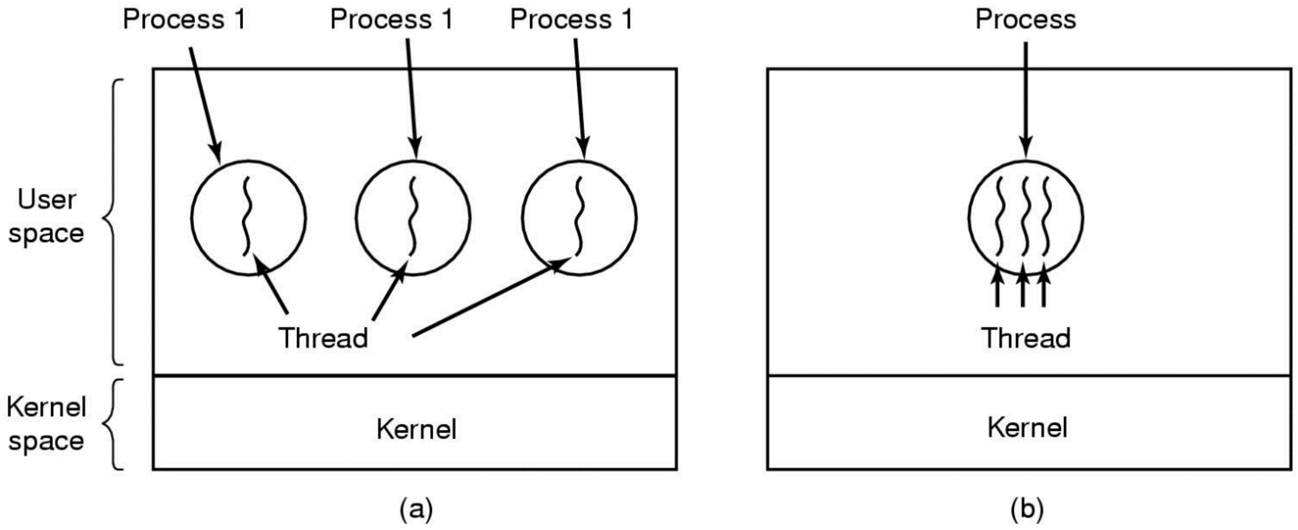
- ❖ Items shared by all threads in a process
- ❖ Items private to each thread

Per process items
Address space
Global variables
Open files
Child processes
Pending alarms
Signals and signal handlers
Accounting information

Per thread items
Program counter
Registers
Stack
State



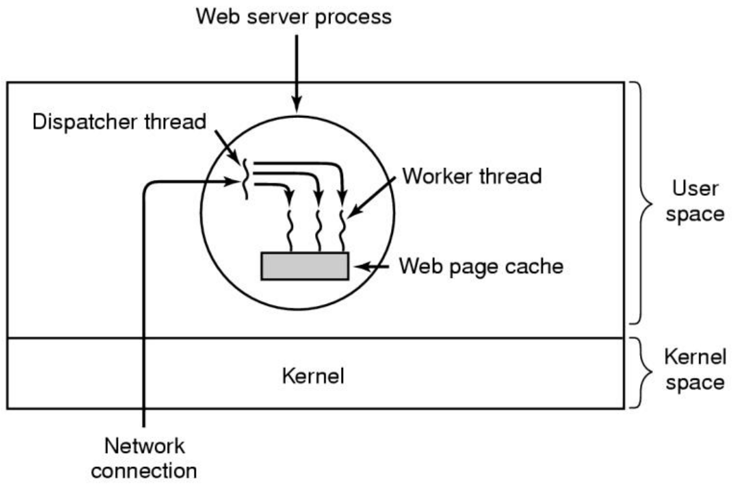
The Thread Model



Three processes each with one thread Vs One process with three threads

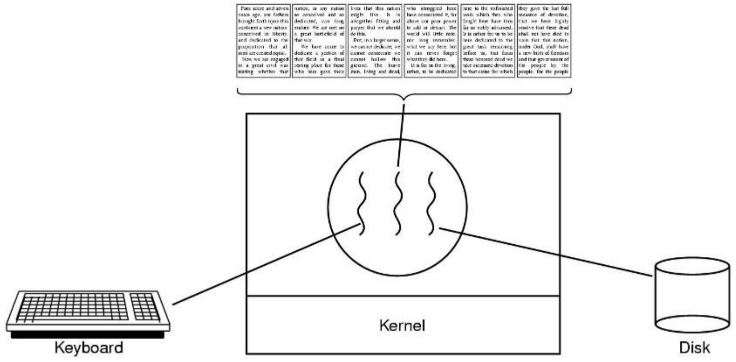
Multi-threaded programs

- ❖ Many software packages that run on modern OS are multi-threaded.
- ❖ A web browser might have
  - ❖ One thread display images or text
  - ❖ Another thread retrieves data from the network



Multi-threaded programs

- ❖ Many software packages that run on modern OS are multi-threaded.
- ❖ A word processor may have
  - ❖ A thread for displaying graphics
  - ❖ Another thread for responding to keystrokes form the user
  - ❖ A third thread for performing spelling and grammar checking



Multi-threaded programs

- Types of Web Server
  - Single-threaded web server: a client might have to wait for its request to be serviced.
  - Multi-processes web server: used before threads become popular, much overhead in creating a new process.
  - Multi-threaded web server: less overhead in thread creation, concurrent service to multiple client.
- Many OS kernels are now multi-threaded
  - Several threads operates in the kernel
  - Each thread performs a specific task, such as managing devices or interrupt handling.

Benefits of multi-threaded programming

- ❖ Responsiveness
  - ❖ Multithreading an interactive application may allow a program to continue running even if part of it is blocked or doing a lengthy operation.
- ❖ Resource Sharing
  - ❖ Threads share the memory and the resources of the process to which they belong.
- ❖ Economy
  - ❖ Because threads in a process shares the resources, it is more economical to create and context-switch threads.
- ❖ Utilization of Multi-Processor Architectures
  - ❖ Threads may be running in parallel on different processors.

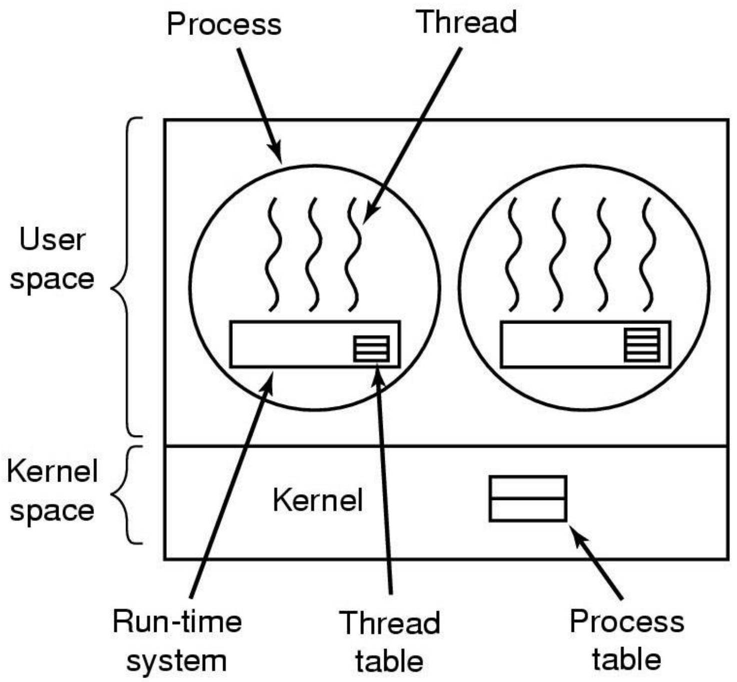
Two types of threads

- **User Thread**
  - User-level thread are threads that are visible to the programmer and are unknown to the kernel.
  - User thread are supported above the kernel and are managed without kernel support.
- Thread management done by user-level threads library
- Three primary thread libraries:
  - POSIX Pthreads
  - Win32 threads
  - Java threads

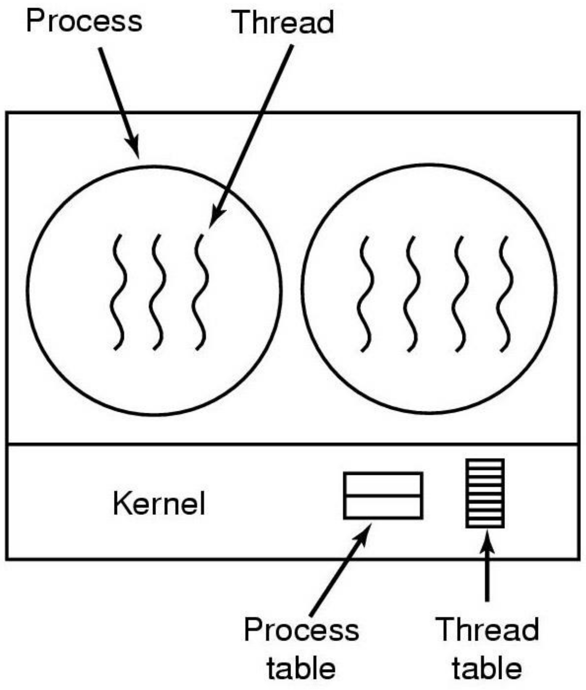
Two types of threads

- **Kernel Thread**
  - OS kernel supports and manages kernel-level threads
  - The threads are supported and managed directly by the operating system.
- Examples
  - Windows 10
  - Solaris
  - Linux
  - Tru64 UNIX
  - Mac OS X

Implementing Threads in User Space



A user-level threads package



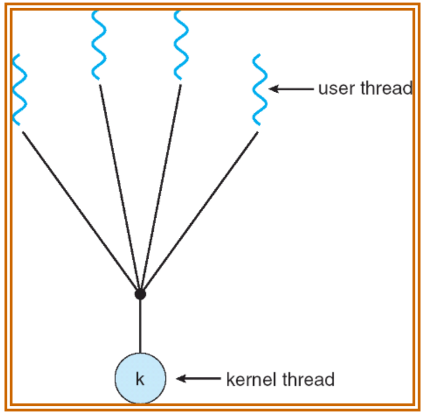
A threads package managed by the kernel

Multithreading Models

- ❖ A Relationship between user threads and kernel threads.
  - ❖ Many-to-One
  - ❖ One-to-One
  - ❖ Many-to-Many
  - ❖ Two Level Model

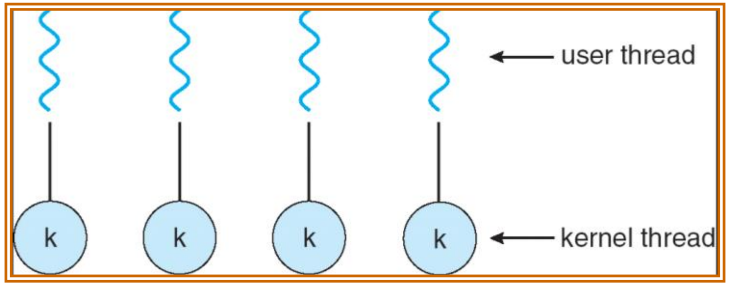
Many-to-One

- Many user-level threads mapped to single kernel thread
  - Thread management is done by the thread library in user space
  - Can create as many user threads as you wish.
  - The entire process will block when a thread makes a blocking system call.
  - Even on multiprocessors, threads are unable to run in parallel
- Examples:
  - Solaris Green Threads
  - GNU Portable Threads



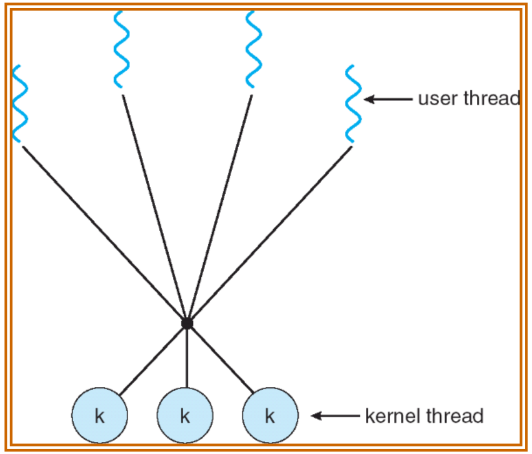
One-to-One

- Each user-level thread maps to a kernel thread
  - Provides more concurrency than the many-to-one model
  - Allows another thread to run when a thread is in blocking system call
  - Creating a user thread requires creating the corresponding kernel thread. (overhead)
  - The number of threads a process can create is smaller than many-to-one model. (careful not to create too many thread)
- Examples
  - Windows NT/XP/2000
  - Linux
  - Solaris 9 and later



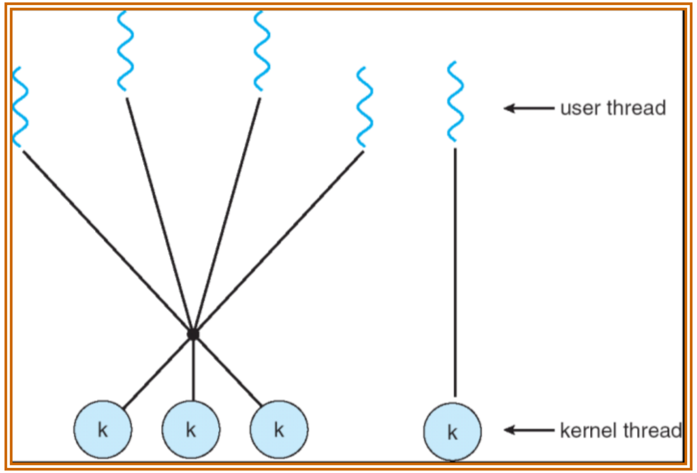
Many-to-Many Model

- ❖ Allows many user level threads to be mapped to smaller or equal kernel threads
  - ❖ Allows the OS to create a sufficient number of kernel threads
  - ❖ The number of kernel threads may be specific to either a application or machine
- ❖ Examples
  - ❖ Solaris prior to version 9
  - ❖ Windows NT/2000 with the ThreadFiber package



Two-Level Model

- One popular variation on many-to-many model
    - Similar to Many-to-Many model,
    - Many user-level threads are multiplexed to a smaller or equal number of kernel threads
    - But it allows a user thread to be **bound** to a kernel thread
- Examples
    - IRIX
    - HP-UX
    - Tru64 UNIX
    - Solaris 8 and earlier



Thank you

johnjose@iitg.ac.in

http://www.iitg.ac.in/johnjose/