SNS COLLEGE OF TECHNOLOGY

Coimbatore-35.
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

COURSE NAME : 23CST202 – OPERATING SYSTEMS

II YEAR/ IV SEMESTER

UNIT – I OVERVIEW AND PROCESS MANAGEMENT

Topic: Multithreading Models

Dr.V.Savitha

Associate Professor

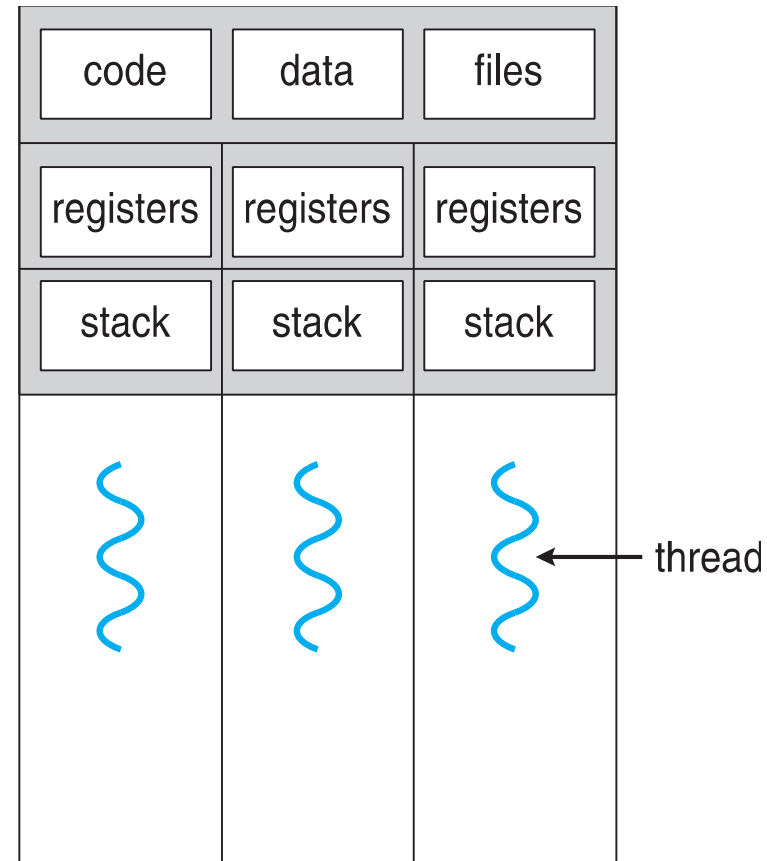Department of Computer Science and Engineering

Threads

- A thread is a basic unit of CPU utilization; it comprises a thread ID, a program counter, a register set, and a stack.

- It shares with other threads belonging to the same process its code section, data section, and other operating-system resources, such as open files and signals.

- A traditional (or heavyweight:) process has a single thread of control. If a process has multiple threads of control, it can perform more than one task at a time.

Single and Multithreaded Processes
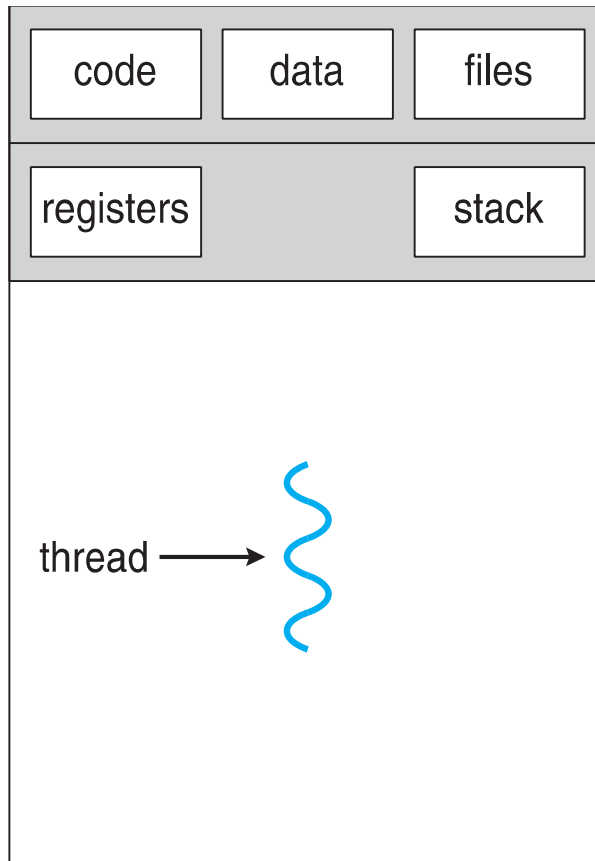
single-threaded process   multithreaded process

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
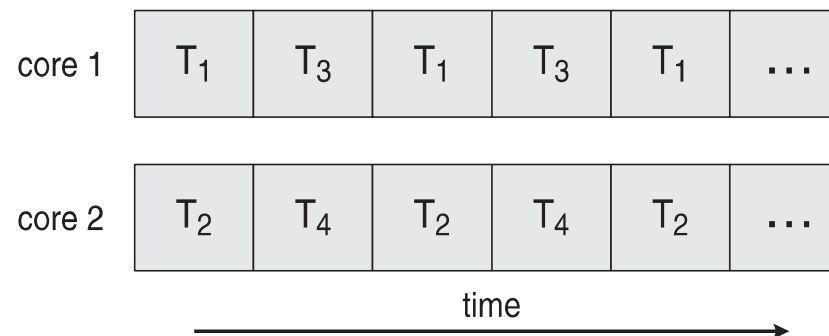
Concurrency vs. Parallelism

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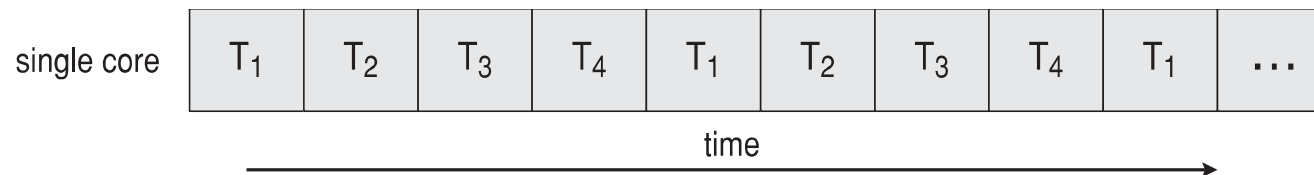
- **Parallelism on a multi-core system:**

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 →

User Threads and Kernel Threads

- **User threads** - management done by user-level threads library
- Three primary thread libraries:
  - POSIX **Pthreads**
  - Windows threads
  - Java threads
- **Kernel threads** - Supported by the Kernel
- Examples – virtually all general purpose operating systems, including:
  - Windows
  - Solaris
  - Linux
 - Tru64 UNIX
  - Mac OS X

Multithreading Models

- Many-to-One

- One-to-One

- Many-to-Many

19CSB201-Operating Systems/Unit-I/Dr.V.Savitha

Many-to-One

- Many user-level threads mapped to single kernel thread

- One thread blocking causes all to block

- Multiple threads may not run in parallel on multicore system because only one may be in kernel at a time

- Few systems currently use this model

- Examples:

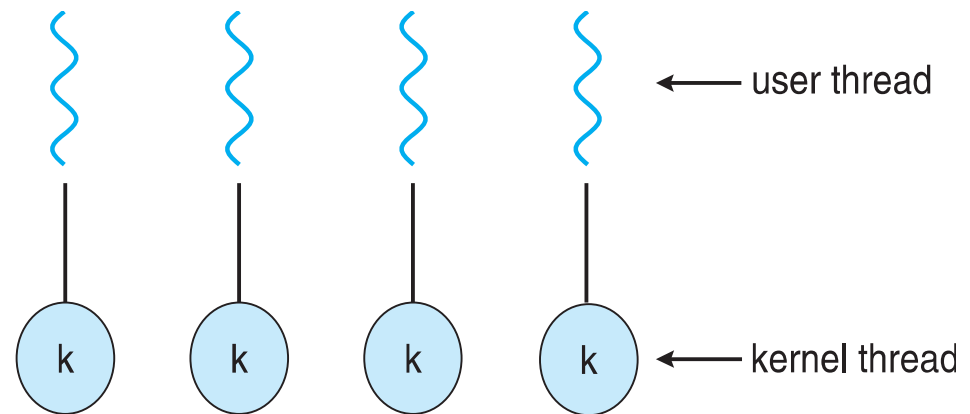 - **Solaris Green Threads**

  - **GNU Portable Threads**

← user thread

k ← kernel thread

19CSB201-Operating Systems/Unit-I/Dr.V.Savitha
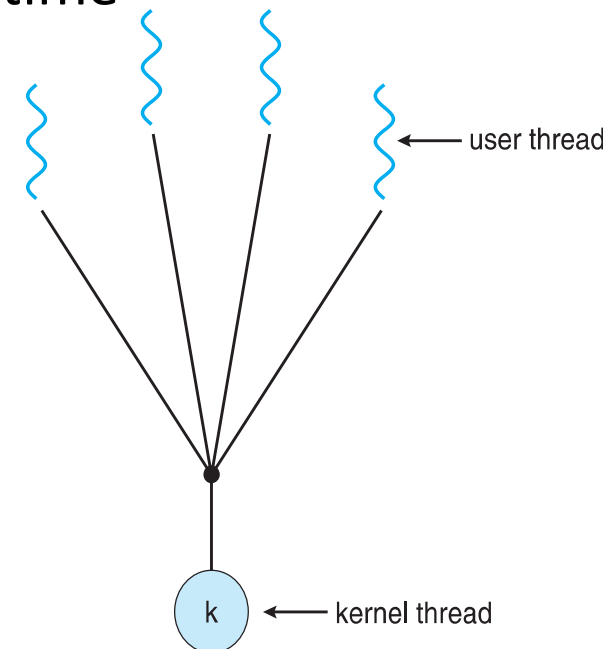
One-to-One

- Each user-level thread maps to kernel thread
- Creating a user-level thread creates a kernel thread
- More concurrency than many-to-one
- Number of threads per process sometimes restricted due to overhead
- Examples
  - Windows
  - Linux
  - Solaris 9 and later

← user thread

k   k   k   k ← kernel thread

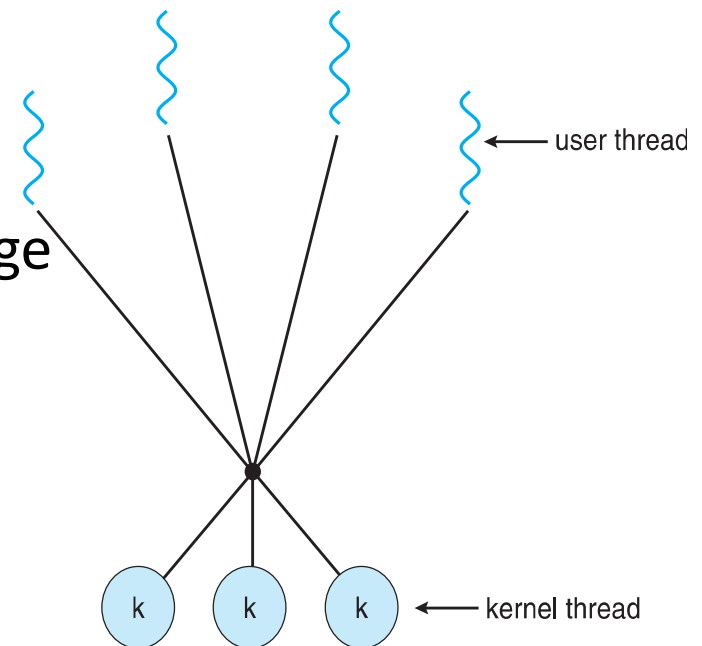19CSB201-Operating Systems/Unit-I/Dr.V.Savitha

www.snsgroups.com

Many-to-Many Model

- Allows many user level threads to be mapped to many kernel threads

- Allows the operating system to create a
  sufficient number of kernel threads

- Solaris prior to version 9

- Windows with the ThreadFiber package

Two-level Model

- Similar to M:M, except that it allows a user thread to be **bound** to kernel thread

- Examples
  - IRIX
  - HP-UX
  - Tru64 UNIX
  - Solaris 8 and earlier