

Threads:

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 / heavyweight process has a single thread of control.

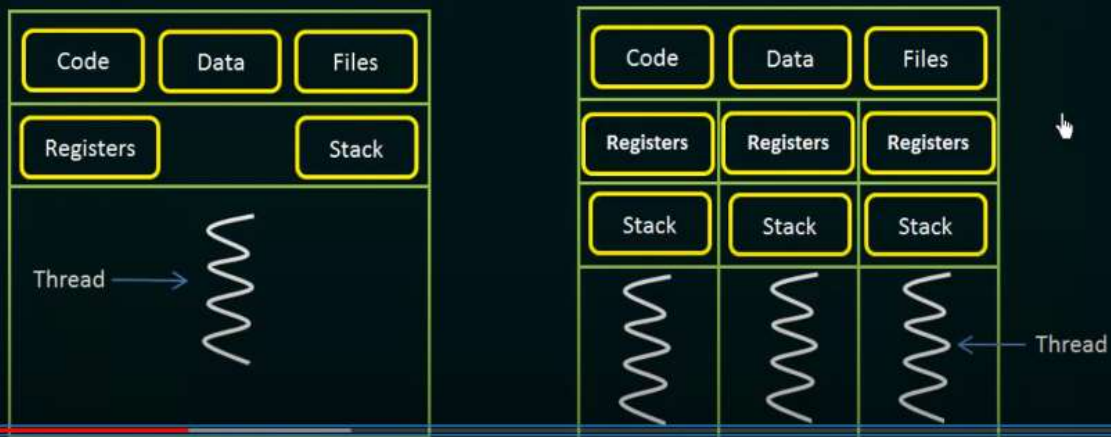
If a process has multiple threads of control, it can perform more than one task at a time.

Single And Multiple Threads:

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 / 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.



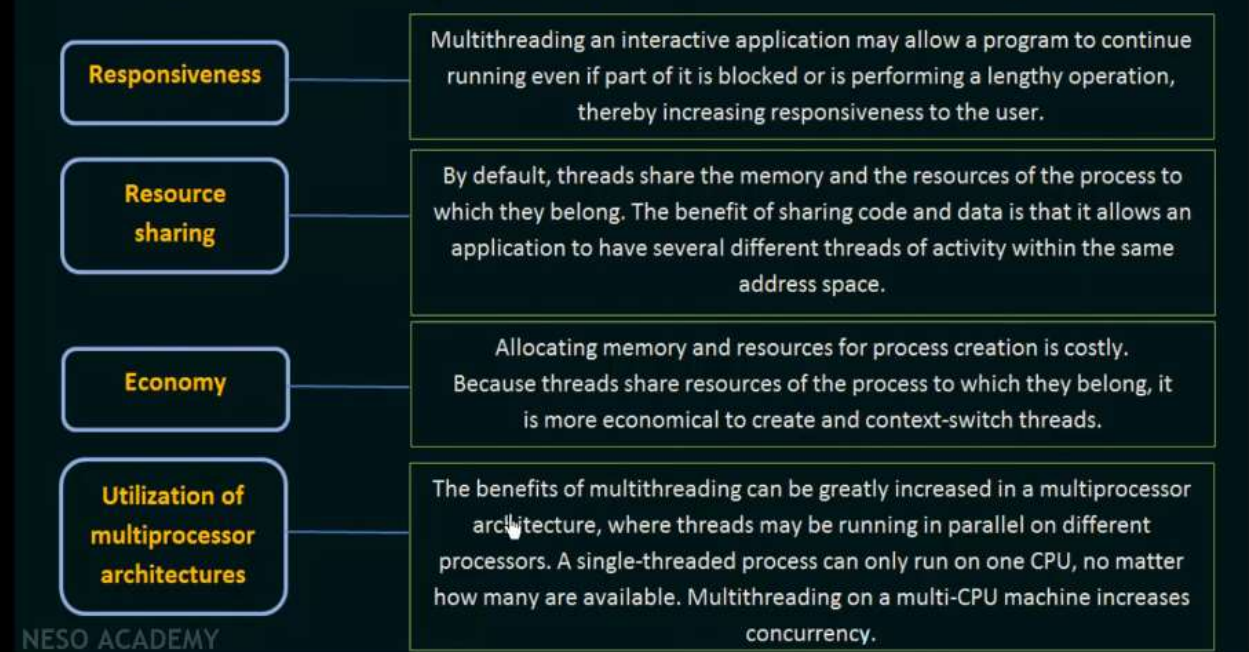
Single-threaded process

Multi-threaded process



Benefit of Multithreaded Programming:

The benefits of multithreaded programming can be broken down into four major categories:



Types of Threads:

Multithreading Models and Hyperthreading

Types of Threads:

- 1) **User Threads** - Supported above the kernel and are managed without kernel support.
- 2) **Kernel Threads** - Supported and managed directly by the operating system.

Ultimately, there must exist a relationship between user threads and kernel threads.

There are three common ways of establishing this relationship:

1. **Many-to-One Model**
2. **One-to-One Model**
3. **Many-to-Many Model**

Advantages and Disadvantages of User Level Threads:

Advantages of User Level Threads:

1. User-level threads are easier and faster to create than kernel-level threads. They can also be more easily managed.
2. User-level threads can be run on any operating system.
3. There are no kernel mode privileges required for thread switching in user-level threads

Disadvantage of User Level Thread:

1. In a pure ULT strategy, a multithreaded application cannot take advantage of multiprocessing.
2. The entire process is blocked if one user-level thread performs blocking operation.

Advantages and Disadvantages of Kernel Level Threads:

Advantage of Kernel Level Thread:

1. Multiple threads of the same process can be scheduled on different processors in kernel-level threads.
2. The kernel routines can also be multithreaded.
3. If a kernel-level thread is blocked, another thread of the same process can be scheduled by the kernel.

Disadvantages of Kernel Level Thread:

1. A mode switch to kernel mode is required to transfer control from one thread to another in a process.
2. Kernel-level threads are slower to create as well as manage as compared to user-level threads.

Many-to-One Model:

Many-to-One Model

The diagram illustrates the Many-to-One model. Four wavy lines, each labeled 'User Thread' with an arrow, converge at a single point and then lead to a single circle labeled 'K', which is labeled 'Kernel Thread' with an arrow. A mouse cursor points to the circle 'K'.

- Maps many user-level threads to one kernel thread.
- Thread management is done by the thread library in user space, so it is efficient.
- The entire process will block if a thread makes a blocking system call.
- Because only one thread can access the kernel at a time, multiple threads are unable to run in parallel on multiprocessors.

NESO ACADEMY

One-to-One Model:

One-to-One Model

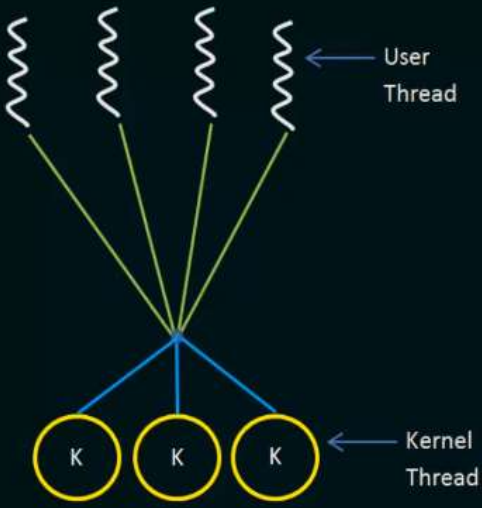
The diagram illustrates the One-to-One model. Four wavy lines, each labeled 'User Thread' with an arrow, are shown. Each wavy line is connected by a straight line to a separate circle labeled 'K', which is labeled 'Kernel Thread' with an arrow. A mouse cursor points to the fourth circle 'K'.

- Maps each user thread to a kernel thread.
- Provides more concurrency than the many-to-one model by allowing another thread to run when a thread makes a blocking system call;
- Also allows multiple threads to run in parallel on multiprocessors.
- Creating a user thread requires creating the corresponding kernel thread.
- Because the overhead of creating kernel threads can burden the performance of an application, most implementations of this model restrict the number of threads supported by the system.

NESO ACADEMY

Many-to-Many Model:

Many-to-Many Model



The diagram illustrates the Many-to-Many Model. At the top, four wavy lines represent User Threads. These threads converge at a central point. From this point, three straight lines diverge to three circles, each labeled with the letter 'K', representing Kernel Threads. Labels with arrows point to the wavy lines as 'User Thread' and the circles as 'Kernel Thread'.


- Multiplexes many user-level threads to a smaller or equal number of kernel threads.
- The number of kernel threads may be specific to either a particular application or a particular machine.
- Developers can create as many user threads as necessary, and the corresponding kernel threads can run in parallel on a multiprocessor.
- Also, when a thread performs a blocking system call, the kernel can schedule another thread for execution.

NESO ACADEMY

Hyperthreading OR Simultaneous Multithreading (SMT):

Hyperthreading
or
Simultaneous Multithreading (SMT)

Hyperthreaded systems allow their processor cores' resources to become multiple logical processors for performance.

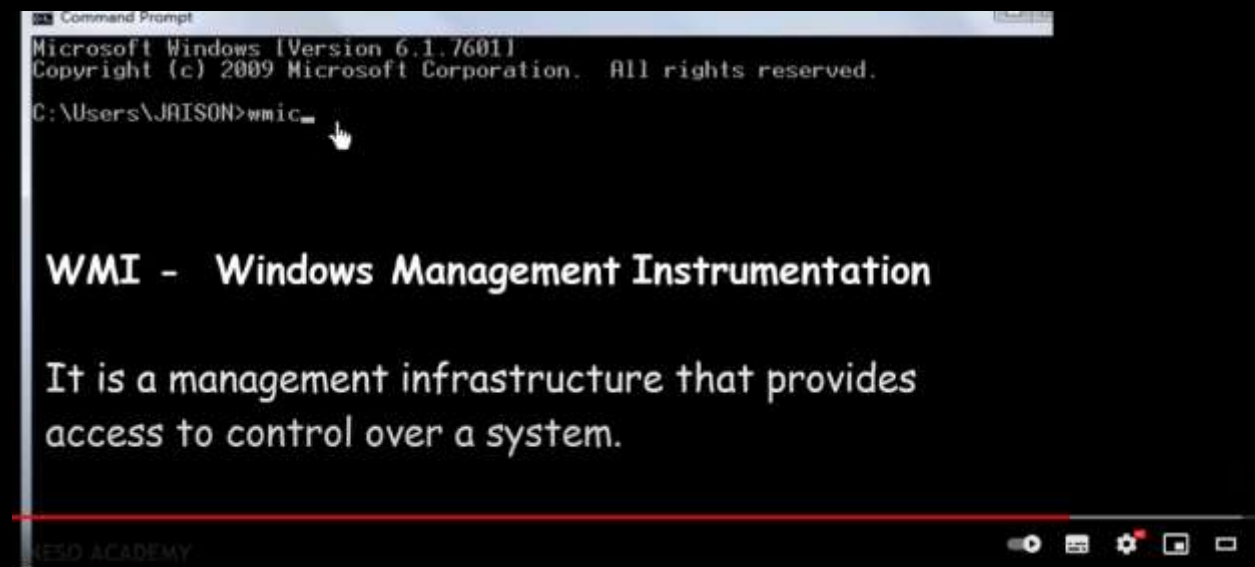


It enables the processor to execute two threads, or sets of instructions, at the same time. Since hyper-threading allows two streams to be executed in parallel, it is almost like having two separate processors working together.

NESO ACADEMY

Check if your computer support SMT:

- ➔ If the number of processor is equal to the number of logical processor that mean no hyper threading happening in the systems
- ➔ If the number of logical processor is greater than the number of physical processor that mean hyper threading happening in the systems



Steps:

1. wmic – Windows Management Instrumentations
2. CPU Get NumberOfCores
3. CPU Get NumberOfLogicalProcessors
4. CPU Get NumberOfCores, NumberOfLogicalProcessors

```
wmic:root\cli>NumbersOfLogicalProcessors
NumbersOfLogicalProcessors - Alias not found.
wmic:root\cli>CPU Get NumbersOfLogicalProcessors
Node - DESKTOP-QT8P0UO
ERROR:
Description = Invalid query

wmic:root\cli>CPU Get NumberOfCores, NumberOfLogicalProcessors
NumberOfCores  NumberOfLogicalProcessors
2              4

wmic:root\cli>
```