1964-2014 celebrating Golden Jubilee

BHARATI VIDYAPEETH PUNE

Multithreading

ECE-1
Year IV
Embedded Systems

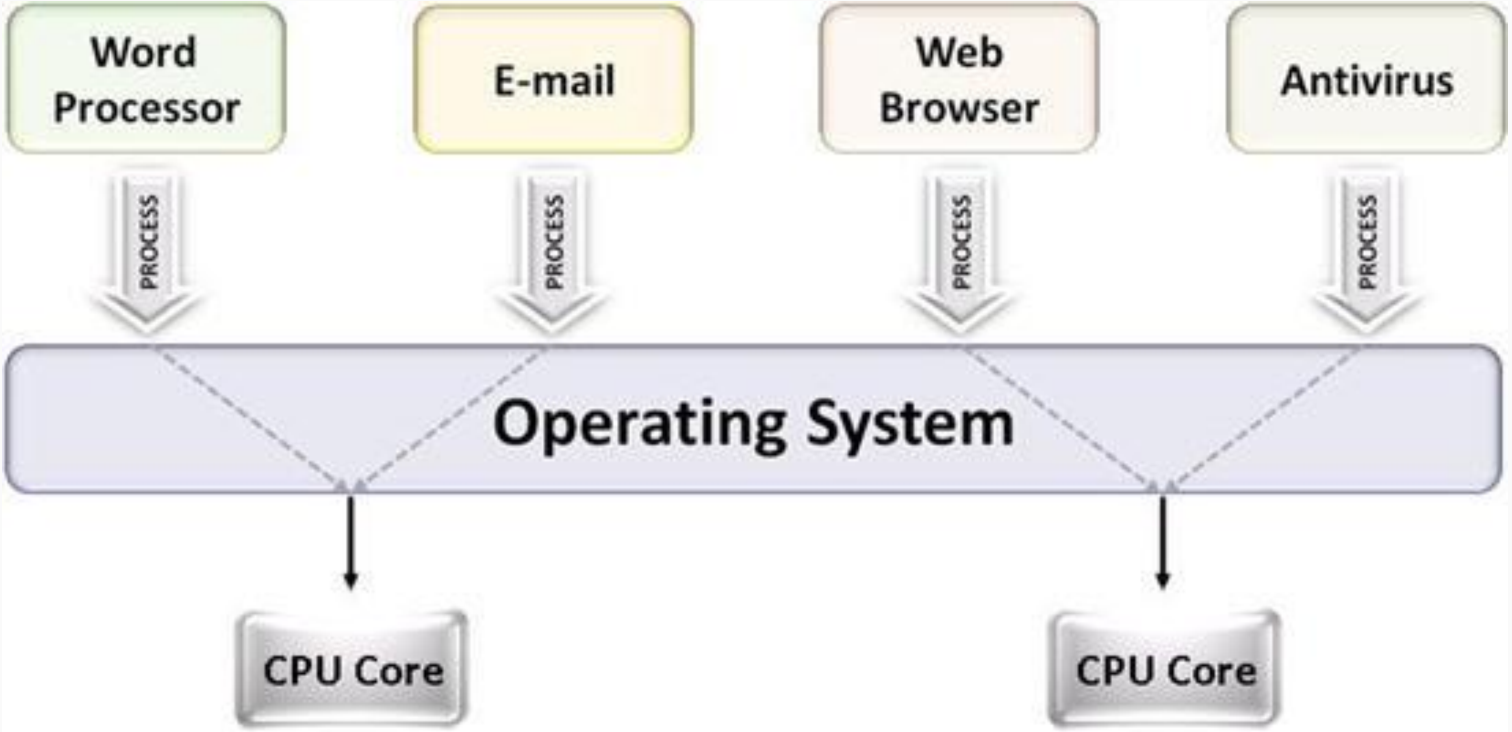
By:-
Chetan Chawla (025)
Divyansh Malhotra (027)
Himanshu (034)
Ishani Janveja (036)

Intro

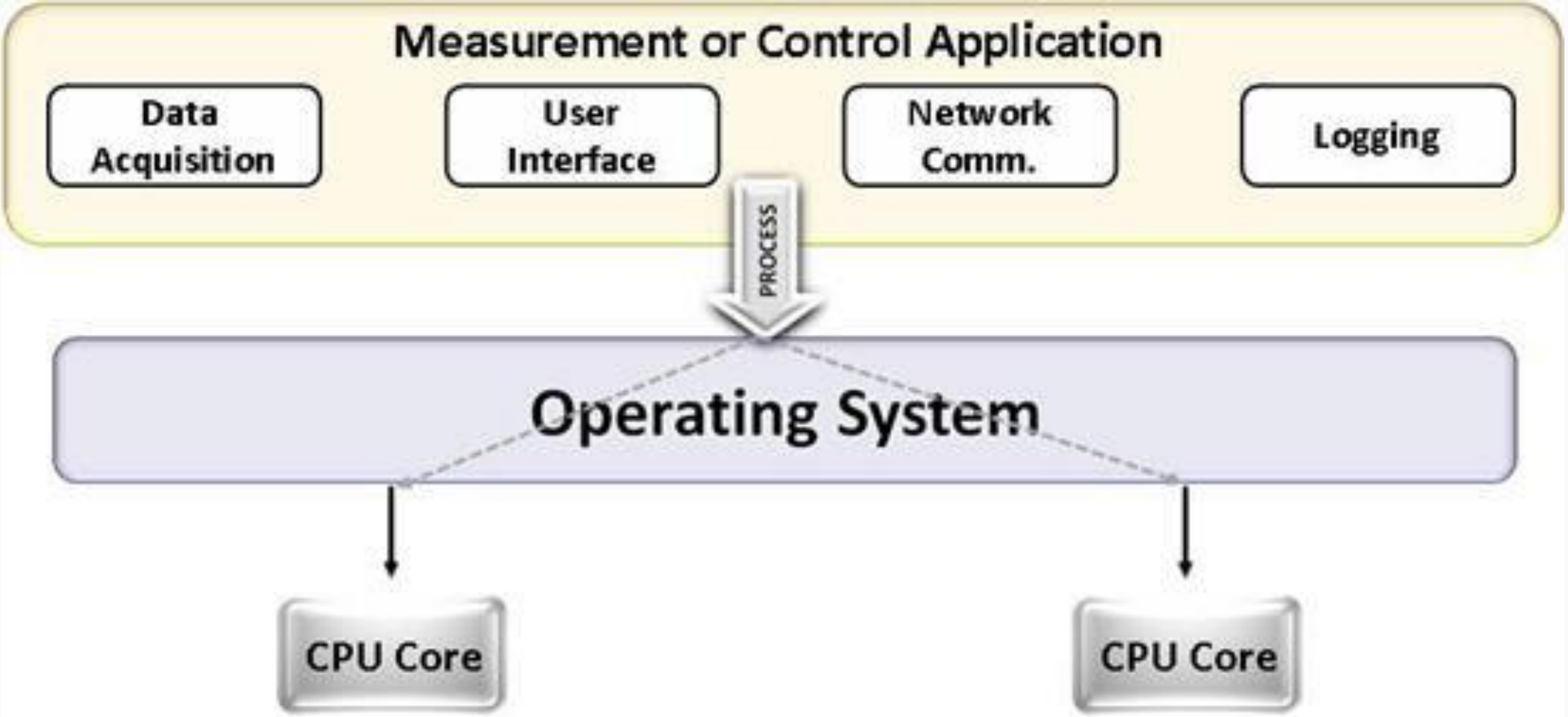
In multithreading you can subdivide specific operations within a single application into individual threads/tasks/processes. Each of the threads can run in parallel. The OS divides processing time not only among different applications, but also among each thread within an application.

Applications that take advantage of multithreading have numerous benefits, including the following:

- More efficient CPU use
- Better system reliability
- Improved performance on multiprocessor computers

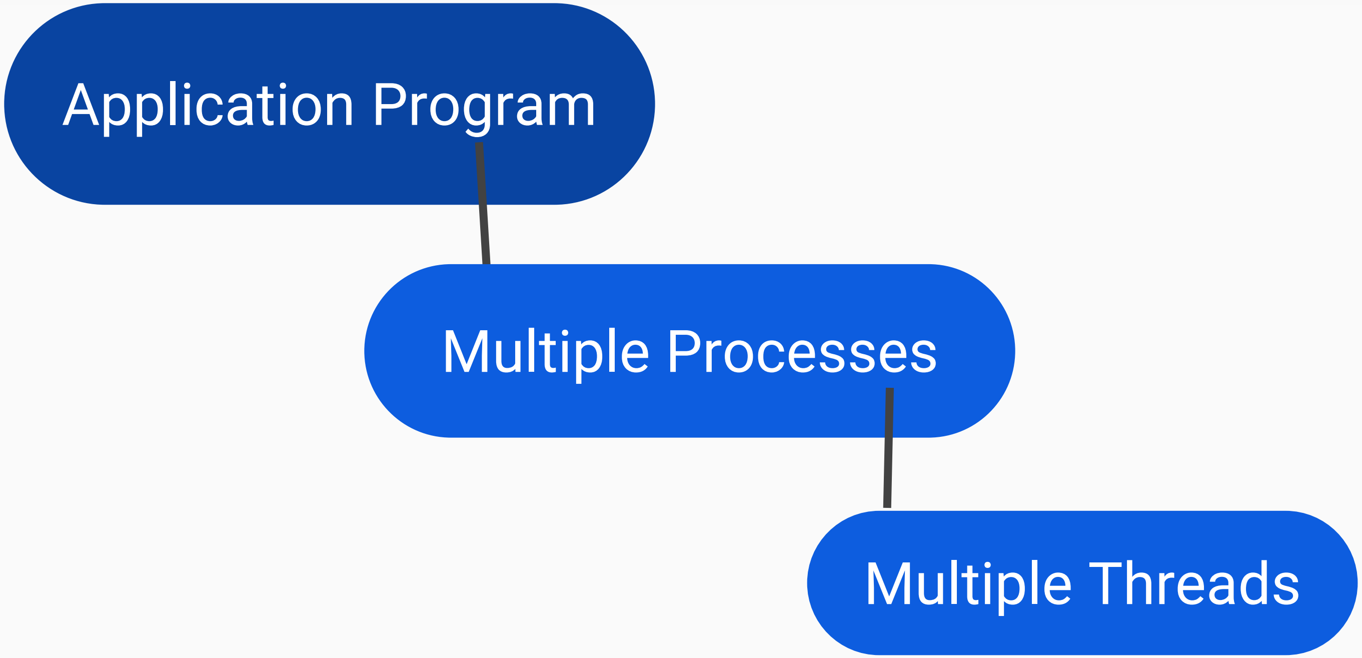


Multi-tasking

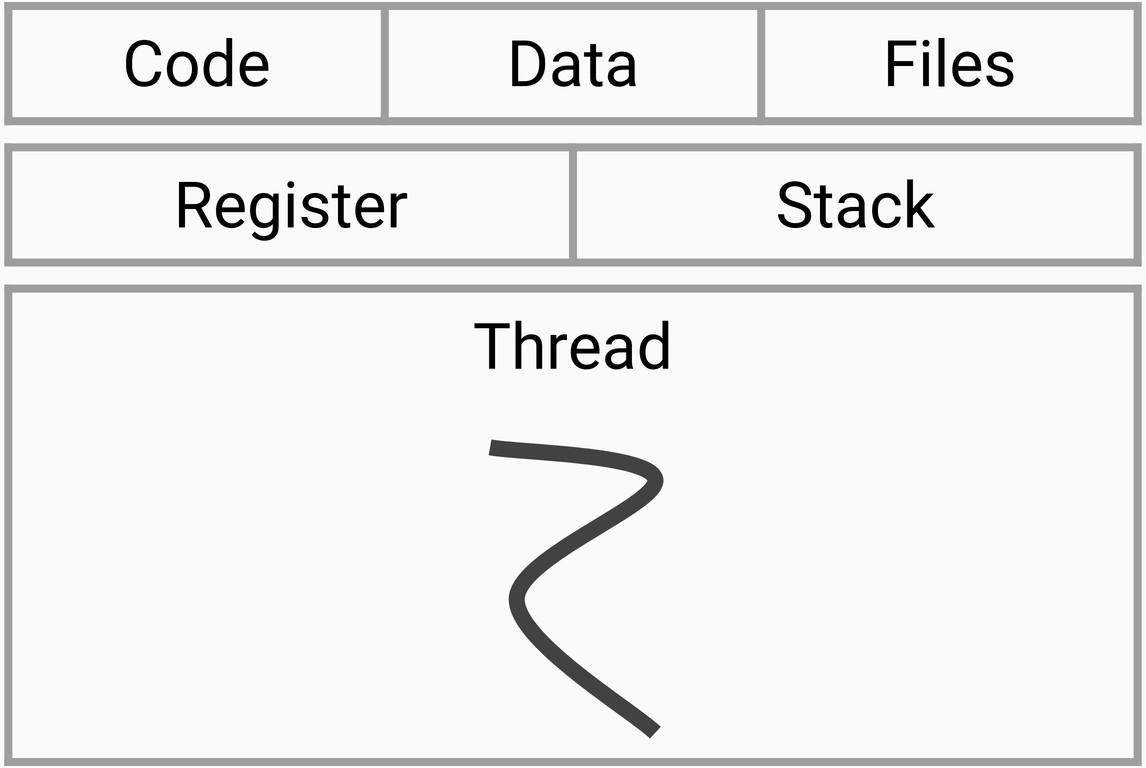


Multi-threading

Relation between processes and threads



Single Threaded Process



Code

Data

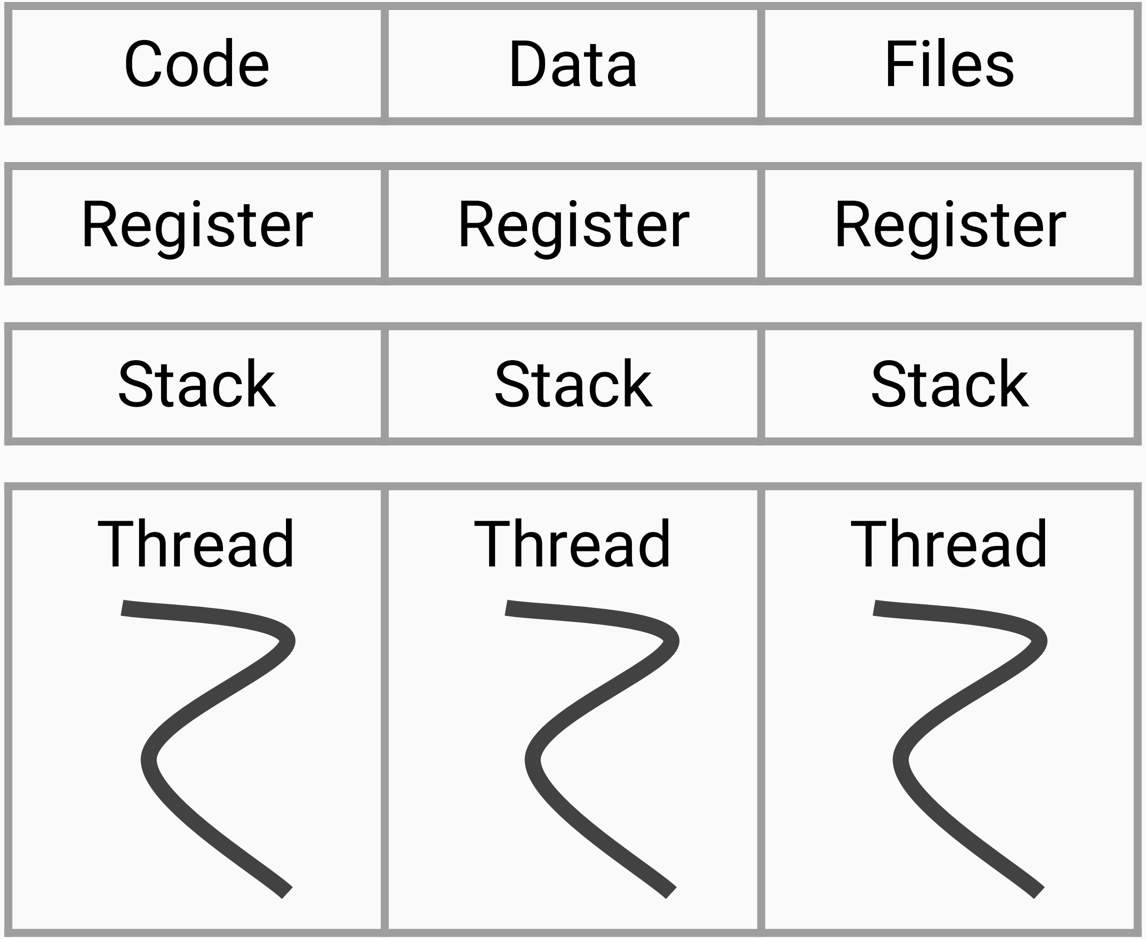
Files

Register

Stack

Thread

Multiple Threaded Process



Code	Data	Files
Register	Register	Register
Stack	Stack	Stack
Thread	Thread	Thread

User Threads

- Thread management in user space
- Invisible to Kernel; no support or management
- Blocks entire process on getting blocked
- Thus, limited benefits

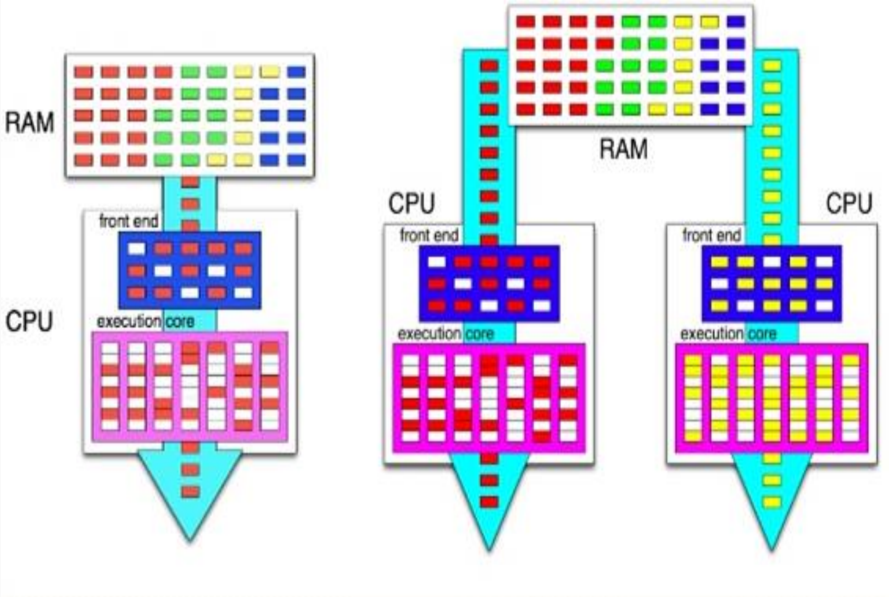
Kernel Threads

- Managed directly by Operating System
- OS executes full control of these threads and create Light Weight Processes (LWPs)
- Supported in Linux, Windows XP/2000 onwards, Solaris, Mac OS, etc.

Types of Multithreading

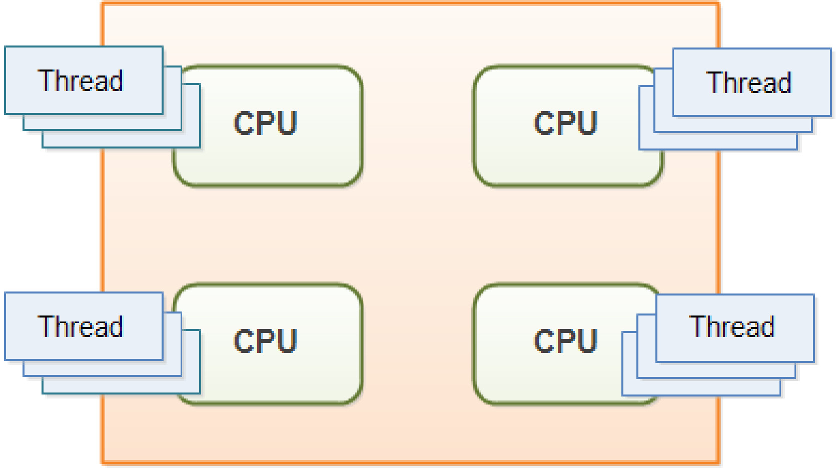
Multithreading can be executed using:

- Multiple Processors
- Single Processor
  - Coarse-grained
  - Fine-grained
  - Simultaneous



Multithreading with Multiple Processors

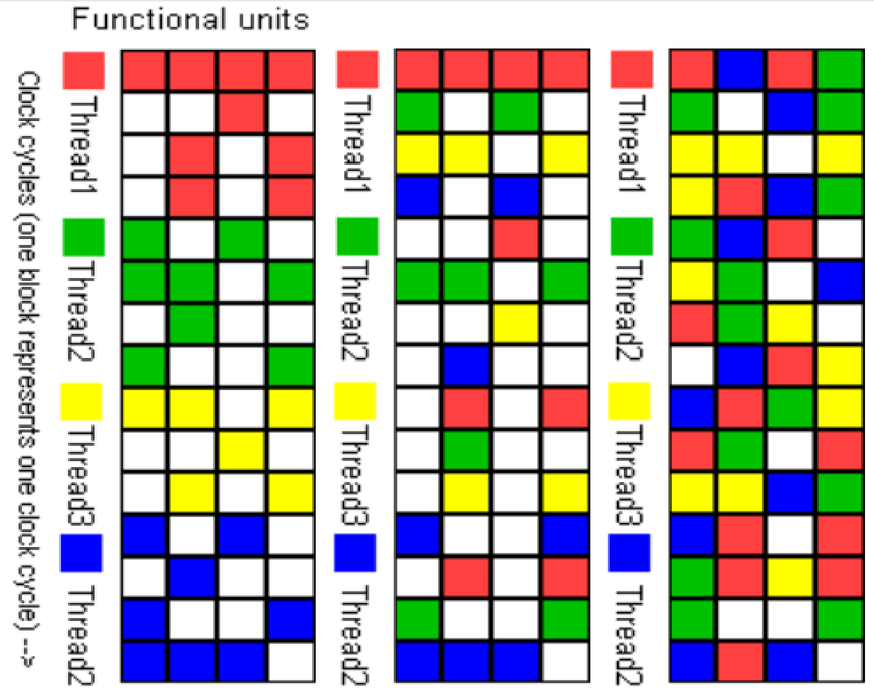
- In case of multiple processors, one thread runs on each processor.
- All threads may either share local memory or have a separate local memory storage connected through an interconnected network.



Multithreading with a Single Processor

<u>Thread Scheduling</u>

- **Coarse Grained Multithreading** - Thread switching on heavy stalls
- **Fine Grained Multithreading** - Thread switching on every clock cycle.
- **Simultaneous Multithreading** - instructions from multiple threads in single clock cycle.

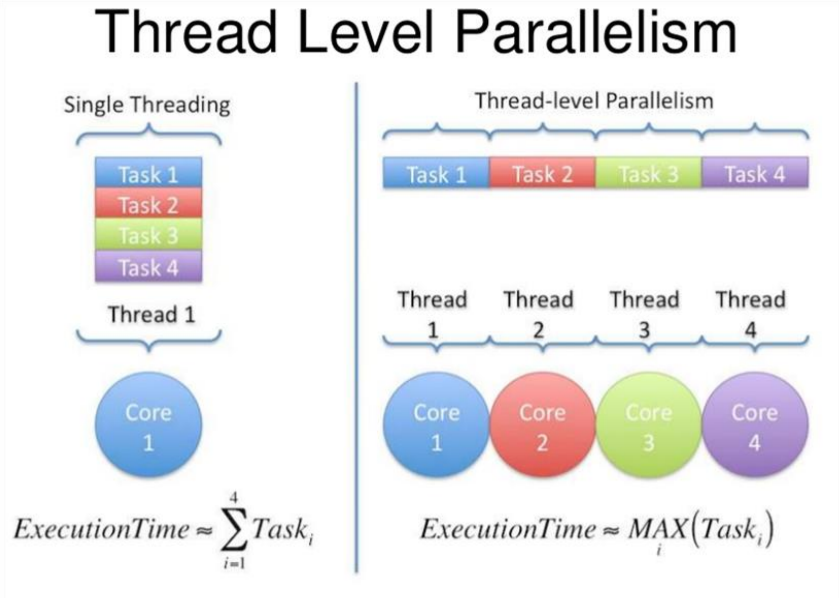


We can use multi threading as:-

- Thread-Level parallelism
- Data-Level parallelism

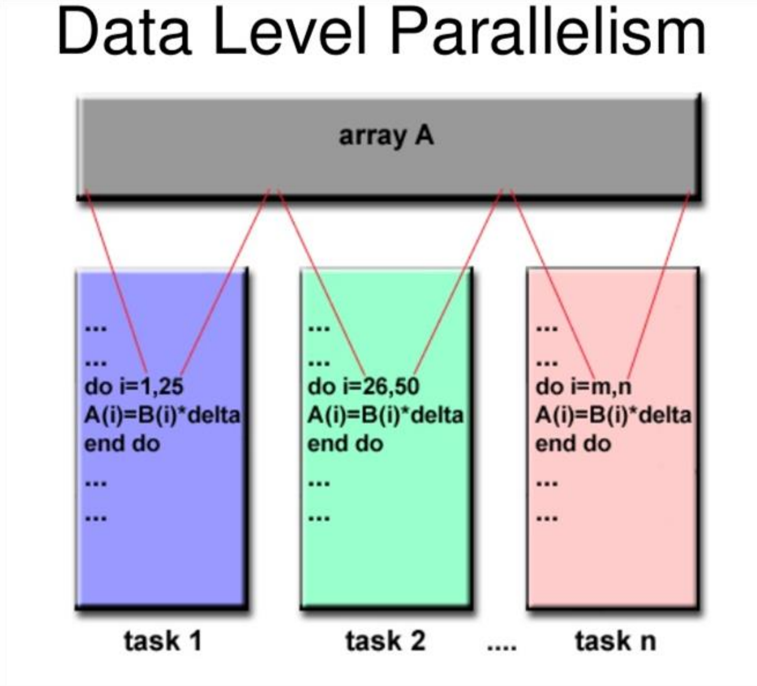
We can use multi threading as

Thread-Level Parallelism, **TLP** is a software capability that allows high-end programs such as a database or web application to work with multiple threads at the same time. Programs that support this ability can do a lot more even under high levels of work loads.



Data Level Parallelism

Data parallelism is parallelization across multiple processors in **parallel** computing environments. It focuses on distributing the **data** across different nodes, which operate on the **data** in **parallel**. It can be applied on regular **data** structures like arrays and matrices by working on each element in **parallel**.



Data Level Parallelism

```
...
...
do i=1,25
A(i)=B(i)*delta
end do
...
...
```

```
...
...
do i=26,50
A(i)=B(i)*delta
end do
...
...
```

```
...
...
do i=m,n
A(i)=B(i)*delta
end do
...
...
```

Multithreading parallelizes processes to fasten up our application execution in time efficient manner.

Thanks for listening

For questions, mail at:
chid.bvcoe@gmail.com