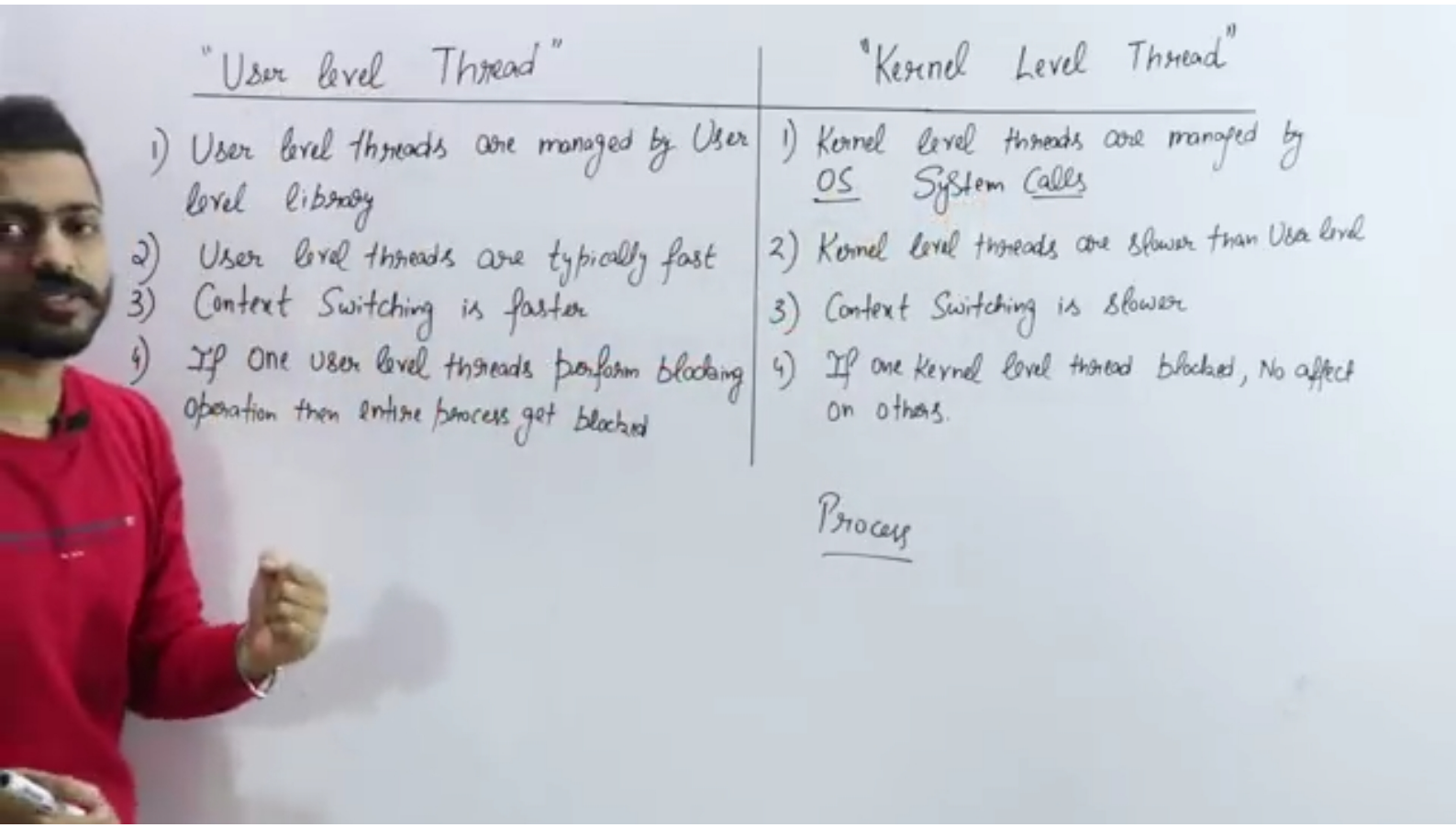
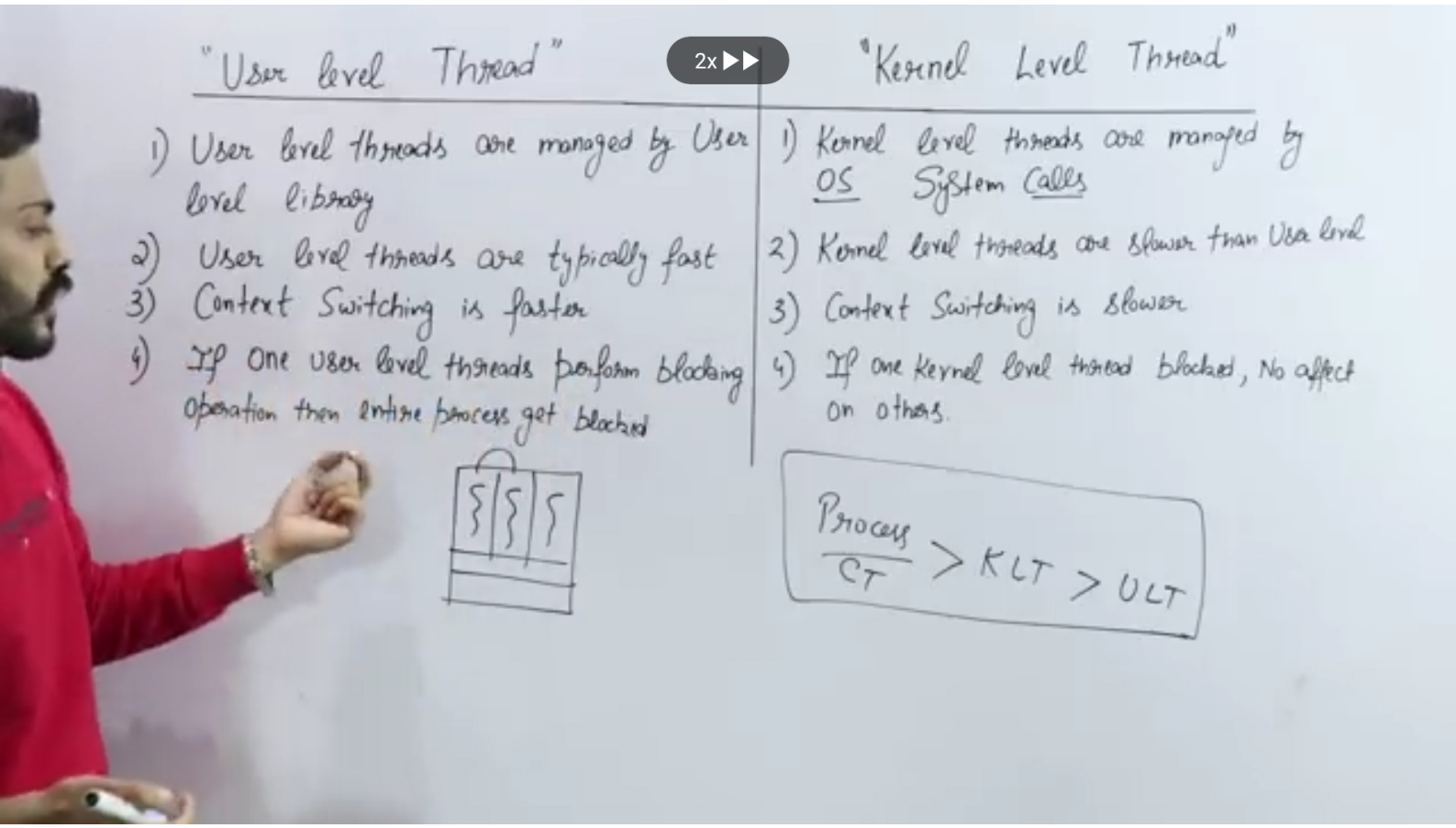
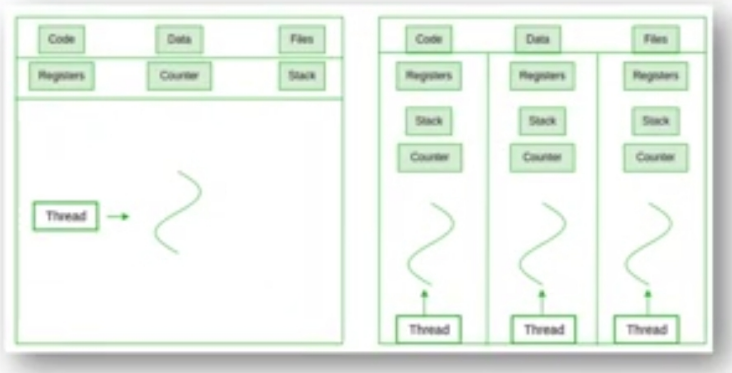

"User level Thread"
1) User level threads are managed by User level library
2) User level threads are typically fast
3) Context Switching is faster
4) If one user level threads perform blocking operation then entire process get blocked
"Kernel Level Thread"
1) Kernel level threads are managed by OS System Calls
2) Kernel level threads are slower than User level
3) Context Switching is slower
4) If one kernel level thread blocked, No affect on others.
Process


"User level Thread"
2x
"Kernel Level Thread"
1) User level threads are managed by User level library
2) User level threads are typically fast
3) Context Switching is faster
4) If one user level threads perform blocking operation then entire process get blocked
1) Kernel level threads are managed by OS System Calls
2) Kernel level threads are slower than User level
3) Context Switching is slower
4) If one kernel level thread blocked, No affect on others.
Process > KLT > ULT
CT

Operating System

Thread
Types of Threads
Multithreading

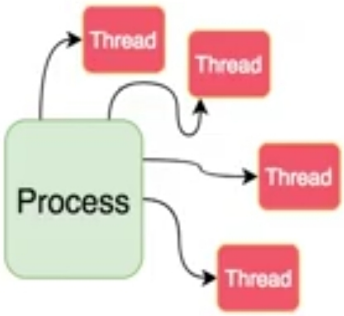


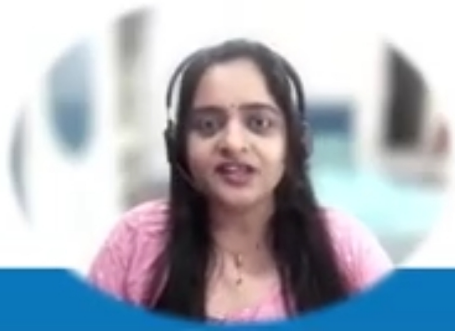
Easy Talk with Real Examples

CS & IT Tutorials by Vrushali

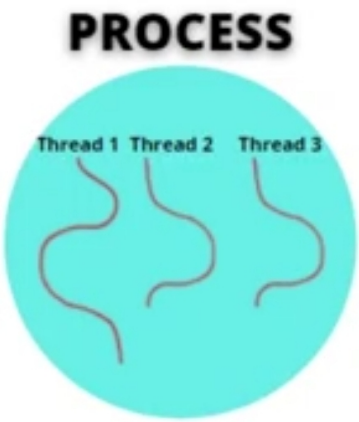
What is Thread?

- Thread is the smallest unit of execution in a process.
- A process can have one or many threads working together.





Why we need Thread?



1. Improved Performance :

- Threads take advantage of multi-core CPUs.
- Each thread can run on a separate core, making processing faster.

Example: Photoshop, WhatsApp, Video editor etc.

2. Responsiveness in Applications :

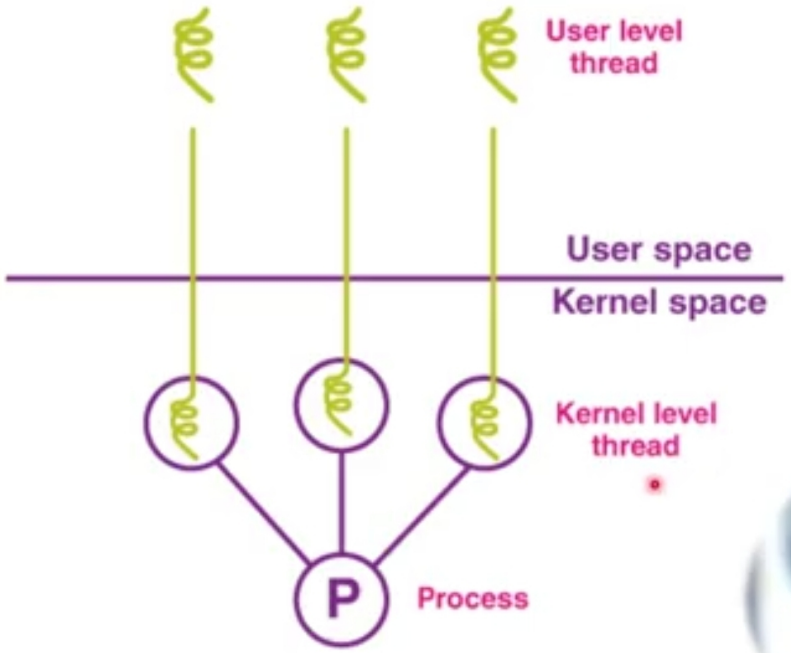
- If one task takes time (like downloading), it won't freeze the entire app.

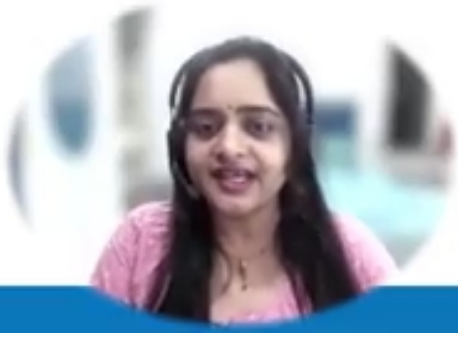
Example: Mobile System.

3. Efficient Resource Sharing :

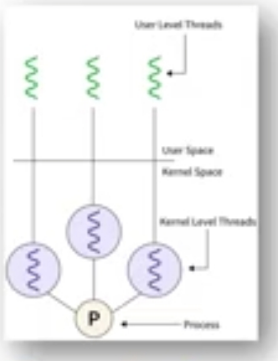
- All threads in a process share the same memory.
- Switching between threads is faster and cheaper than switching between processes.

Types of Thread





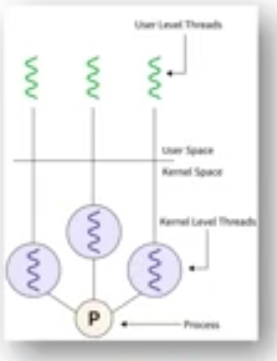
1. User Level Thread



- It's created and managed entirely by user-level programs, libraries or runtime environments (pthread in C or Thread class in Java).
- They are often implemented using standard programming language features.
- Without requiring any special support from the operating system.

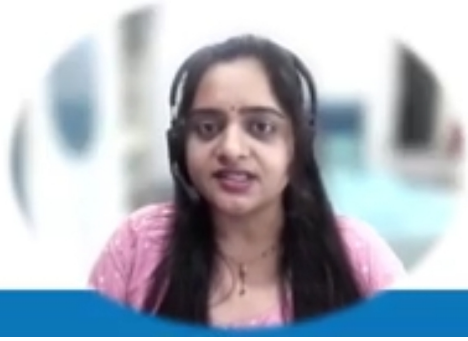
✓**Examples:** Banking Application, Chat Application, E-commerce Website, Games in Java etc.

1. User Level Thread

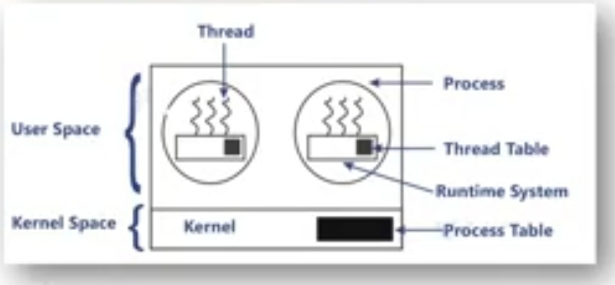


- It's created and managed entirely by user-level programs, libraries or runtime environments (pthread in C or Thread class in Java).
- They are often implemented using standard programming language features.
- Without requiring any special support from the operating system.

✓**Examples:** Banking Application, Chat Application, E-commerce Website, Games in Java etc.



Advantages & Disadvantages of User Level Thread

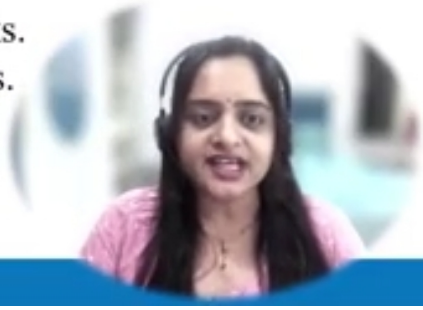


➢Advantages of User Level Thread

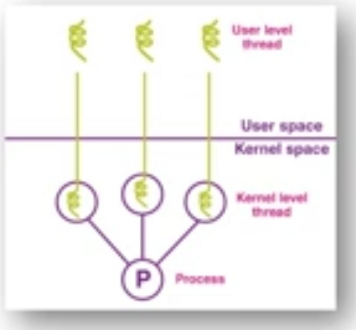
1. **Fast creation** – Threads are created quickly.
2. **Quick switching** – Switching between threads is fast.
3. **Portable** – Can run on any system.
4. **Low overhead** – Uses less memory and resources.
5. **Easy to manage** – Controlled by the program using libraries.

➢Disadvantages of User Level Thread

1. **One blocks all** – If one thread is blocked, all threads stop.
2. **Limited performance** – Not suitable for heavy or CPU-bound tasks.
3. **Poor I/O handling** – Blocking I/O operations can freeze all threads.



2. Kernel Level Thread



- It managed directly by the operating system kernel.
- OS controls lifecycle, scheduling and resource allocation of these threads.
- Kernel-level threads use system call() to interact with OS.

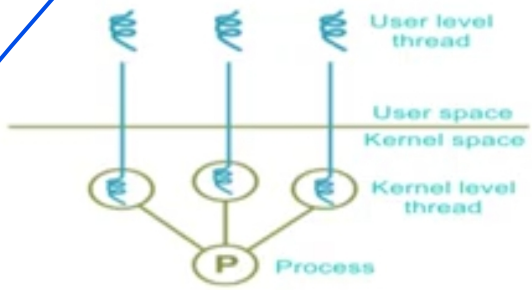
➢Example:

1. When you *open a website*, your request goes to a web server (Apache, Nginx or Tomcat).
2. These servers use kernel-level threads to handle multiple users.
3. Each user request opening a page is handled by a separate thread.

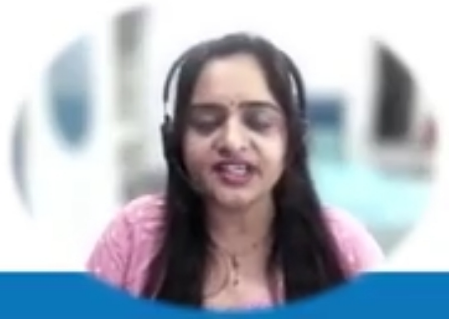
- OS assign different threads to different CPU cores for running parallel.
- They keep server faster & more responsive.

Advantages & Disadvantages of Kernel Level Thread

➢Advantages of Kernel Level Thread

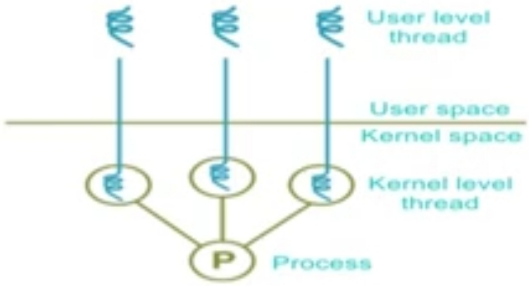


1. **True parallelism** – Runs on multiple CPU cores.
2. **No blocking issue** – One thread blocks, others still run.
3. **Better CPU use** – OS distributes threads efficiently.
4. **Easy to debug** – OS tracks each thread.
5. **Great for multitasking apps** – Like servers and real-time systems.



Advantages & Disadvantages of Kernel Level Thread

➢Advantages of Kernel Level Thread



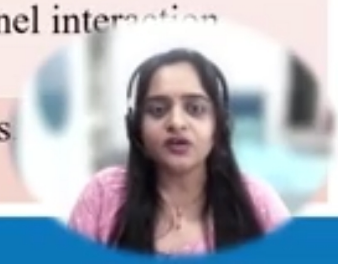
1. **True parallelism** – Runs on multiple CPU cores.
2. **No blocking issue** – One thread blocks, others still run.
3. **Better CPU use** – OS distributes threads efficiently.
4. **Easy to debug** – OS tracks each thread.
5. **Great for multitasking apps** – Like servers and real-time systems.

➢Disadvantages of Kernel Level Thread

1. **More overhead** – Uses more memory/resources.
2. **Complex** – Needs OS/kernel support.
3. **System calls needed** – Makes it a bit slower than user-level threads

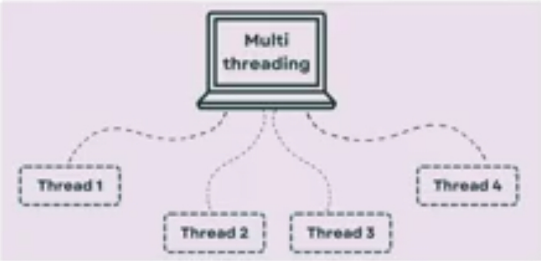
User Level Thread *VS* Kernel Level Thread

No.	User Level Thread	Kernel Level Thread
1	Threads managed **by user space libraries.**	Threads managed **directly by the OS kernel**.
2	OS is **not aware** of user threads.	OS is **fully aware** of kernel threads.
3	**Fast** – no need for kernel mode switch	**Slower** – involves mode switch between user and kernel
4	If one thread blocks (e.g., I/O), **all threads block**	If one thread blocks, **others can continue**
5	**More portable** – doesn't depend on OS	**Less portable** – tightly coupled with OS
6	Thread creation is **Very fast**, done in user space	**Slower**, needs kernel interaction
7	Ex. Java threads, Python threads (CPython)	Ex. Linux kernel threads

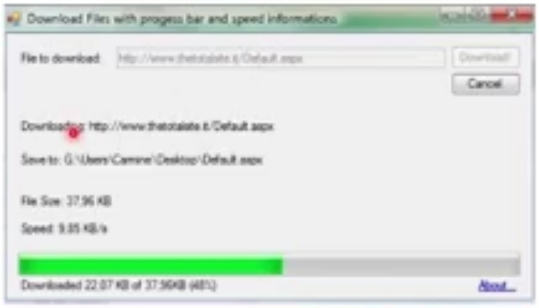


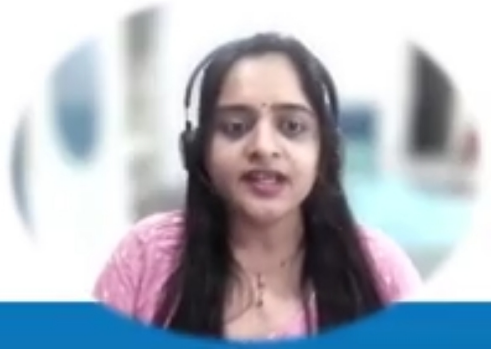
About Multithreading

- It is programming concept where multiple threads run simultaneously in a single program.
- It helps in doing multiple tasks at the same time.
- It makes programs faster and more efficient.



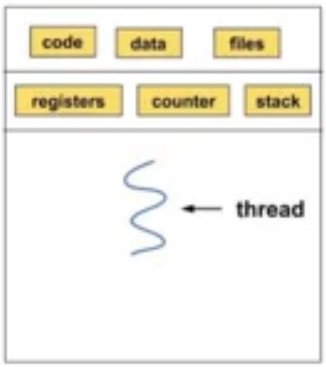
Example: Downloading File





Single Threaded Process

- It is a program that has only one thread of execution.
- Means program can do only one task at a time, step by step.

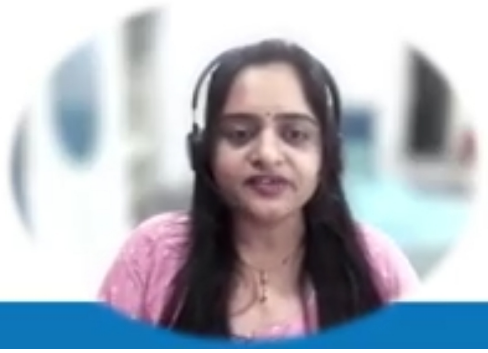


Single-threaded process

➢ **Components of Single Thread:** **Calculate Addition of Two Numbers**

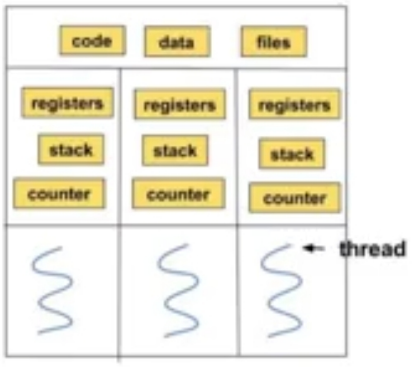
1. **Code :** Program Instructions
2. **Data :** Variables, arrays, constants
3. **Files :** Any files the process uses (read/write)
4. **Registers :** Temporary, fast-access CPU storage
5. **Counter :** Keep track of executed instruction
6. **Stack :** Stores function calls, parameters, return values

Example: Calculator, Notepad, Command Line Program, File Compression, Simple Game etc



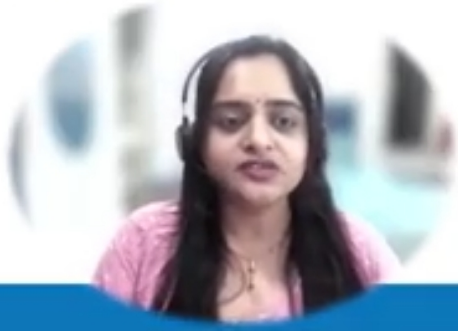
Multi Threaded Process

- It is a program that has multiple thread of execution parallelly.
- Each thread perform different task.
- All threads share the same memory (code, data, files).
- But have separate stack and registers.
- It improves speed, responsiveness and efficient CPU use.



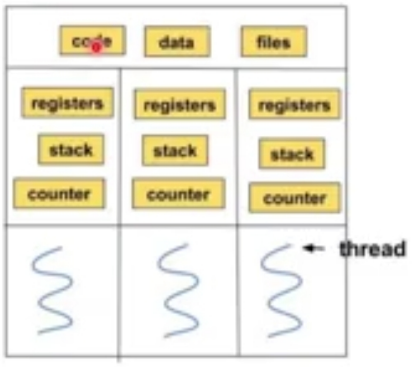
Multi-threaded process

Example: Web Browser, Video Editing Software, Games, IDE, Chat Apps



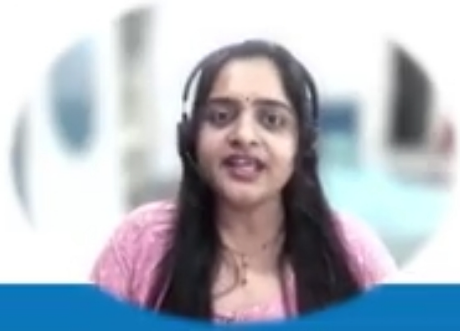
Multi Threaded Process

- It is a program that has multiple thread of execution parallelly.
- Each thread perform different task.
- All threads share the same memory (code, data, files).
- But have separate stack and registers.
- It improves speed, responsiveness and efficient CPU use.



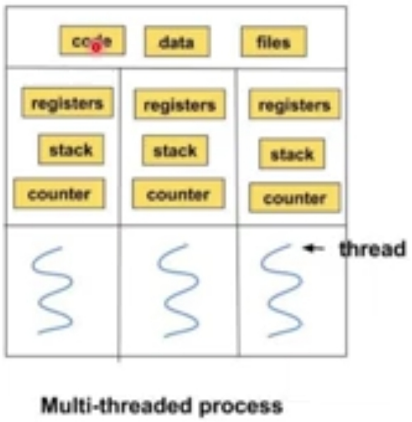
Multi-threaded process

Example: Web Browser, Video Editing Software, Games, IDE, Chat Apps



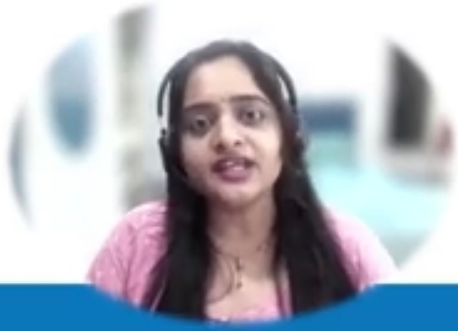
Multi Threaded Process

- It is a program that has multiple thread of execution parallelly.
- Each thread perform different task.
- All threads share the same memory (code, data, files).
- But have separate stack and registers.
- It improves speed, responsiveness and efficient CPU use.



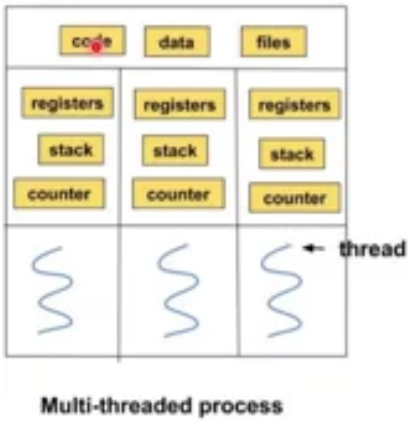
Multi-threaded process

Example: Web Browser, Video Editing Software, Games, IDE, Chat Apps



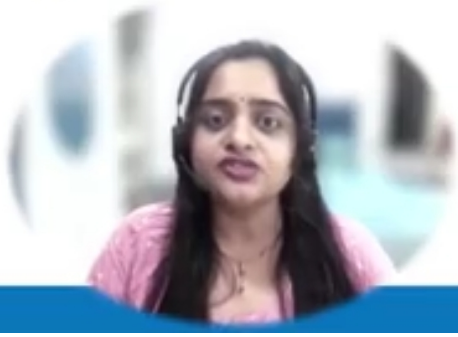
Multi Threaded Process

- It is a program that has multiple thread of execution parallelly.
- Each thread perform different task.
- All threads share the same memory (code, data, files).
- But have separate stack and registers.
- It improves speed, responsiveness and efficient CPU use.



Multi-threaded process

Example: Web Browser, Video Editing Software, Games, IDE, Chat Apps



- **Kernel Level Threads:**

2x ▶▶

1. Kernel Level Threads are supported within the kernel of the operating system.

2. All modern operating system support kenrel level threads.

3. They allow the kernel to perform multiple tasks.

- **Kernel Level Threads:**

1. Kernel Level Threads are supported within the kernel of the operating system.

2. All modern operating system support kenrel level threads.

3. They allow the kernel to perform multiple tasks.

- **User Level Threads:**

1. User Level Threads are implemented in user level libraries instead of system call.

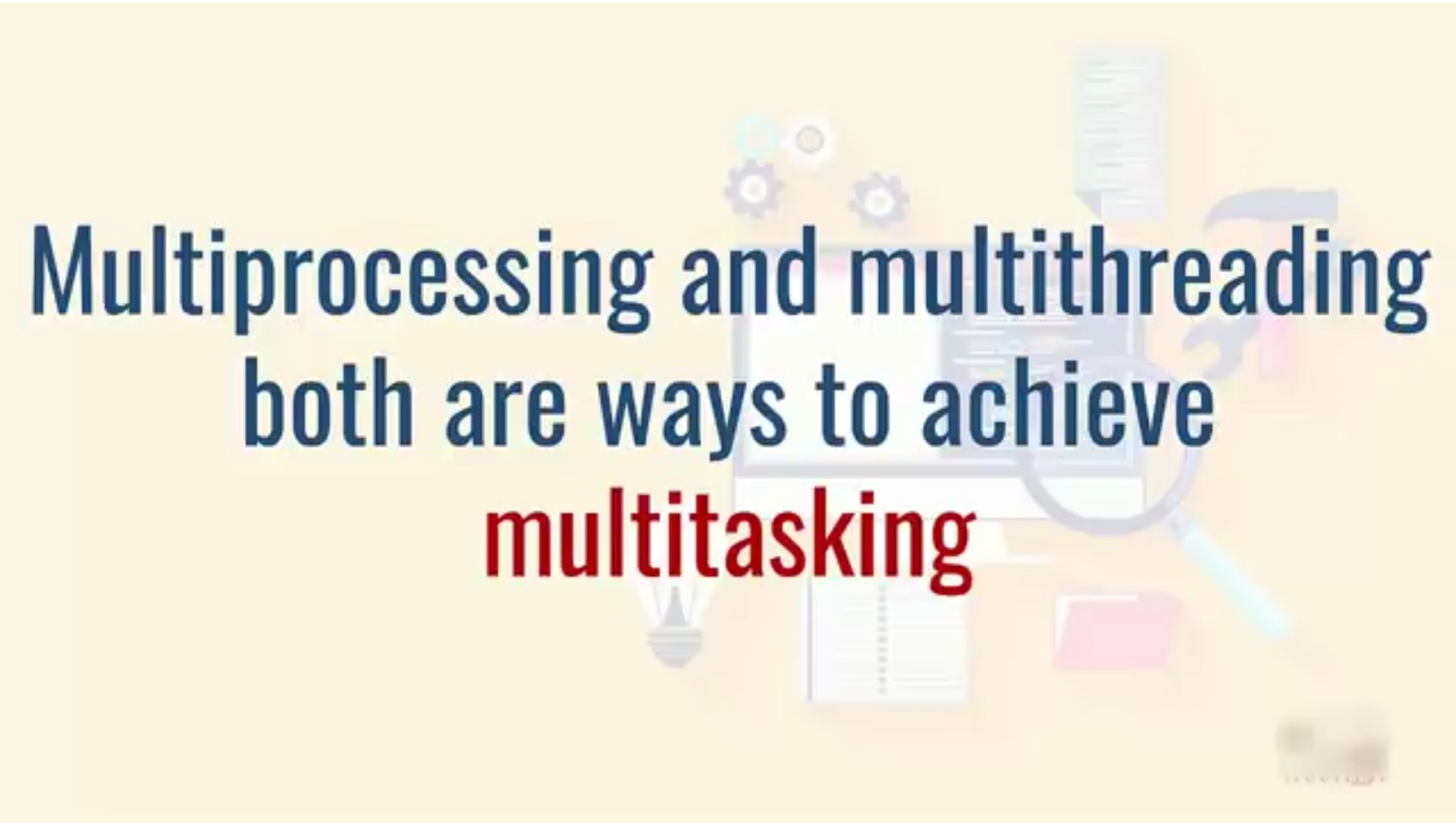
2. The thread switching does not need to call operating system.

3. It does not cause interrupt to the kernel.

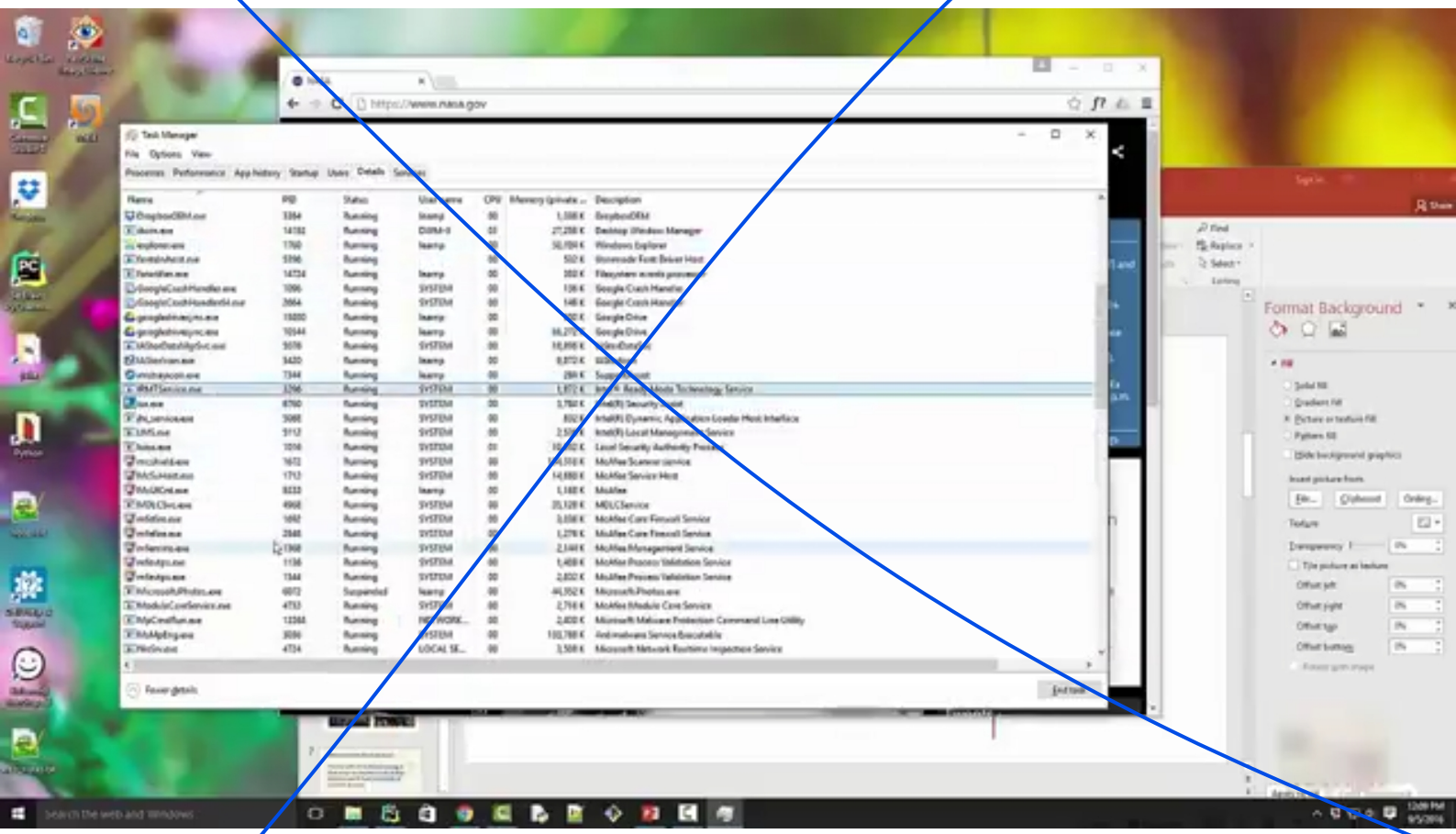
4. The kernel knows nothing about user level threads.

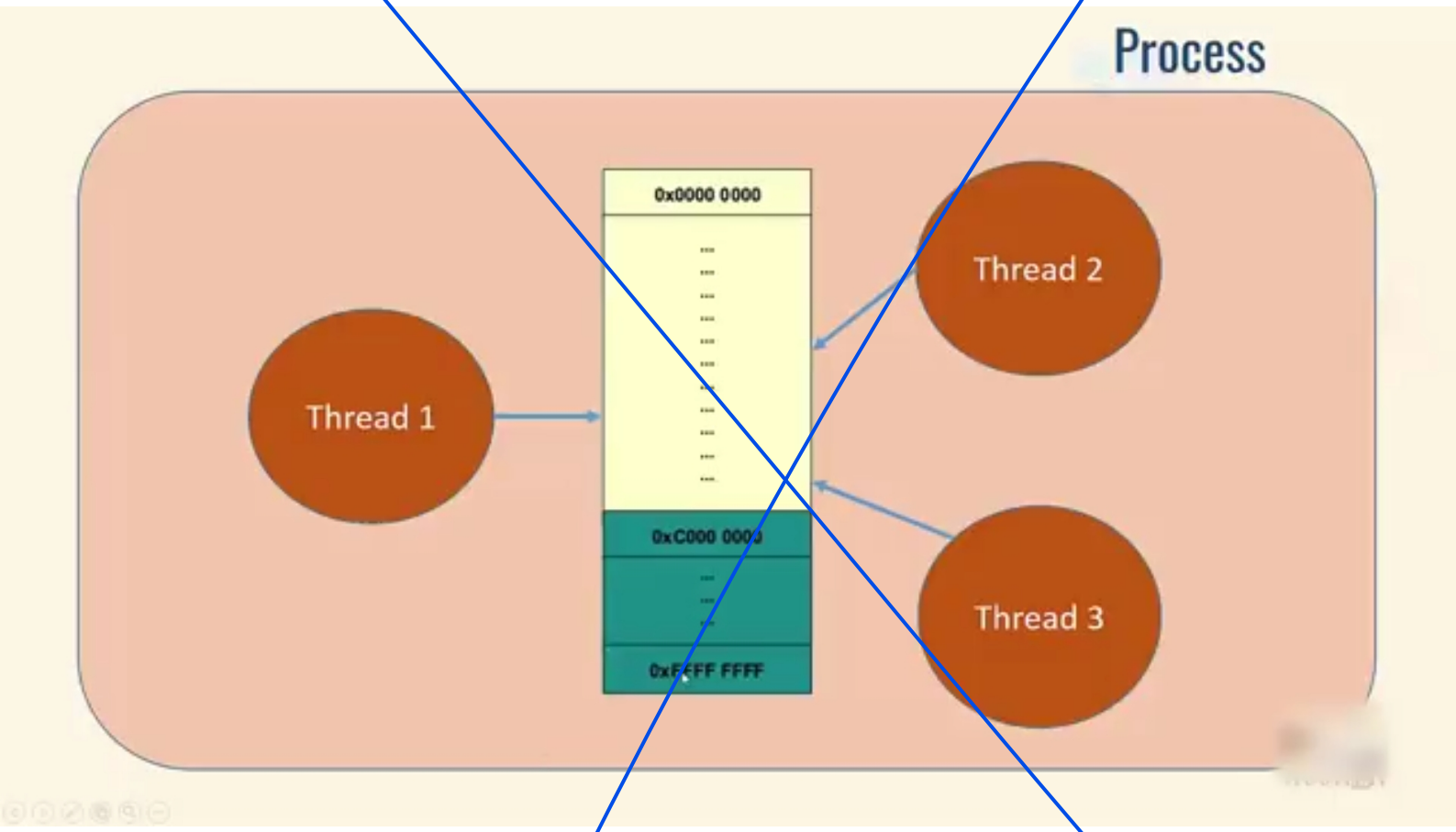
5. It manages these threads as single threaded processes.

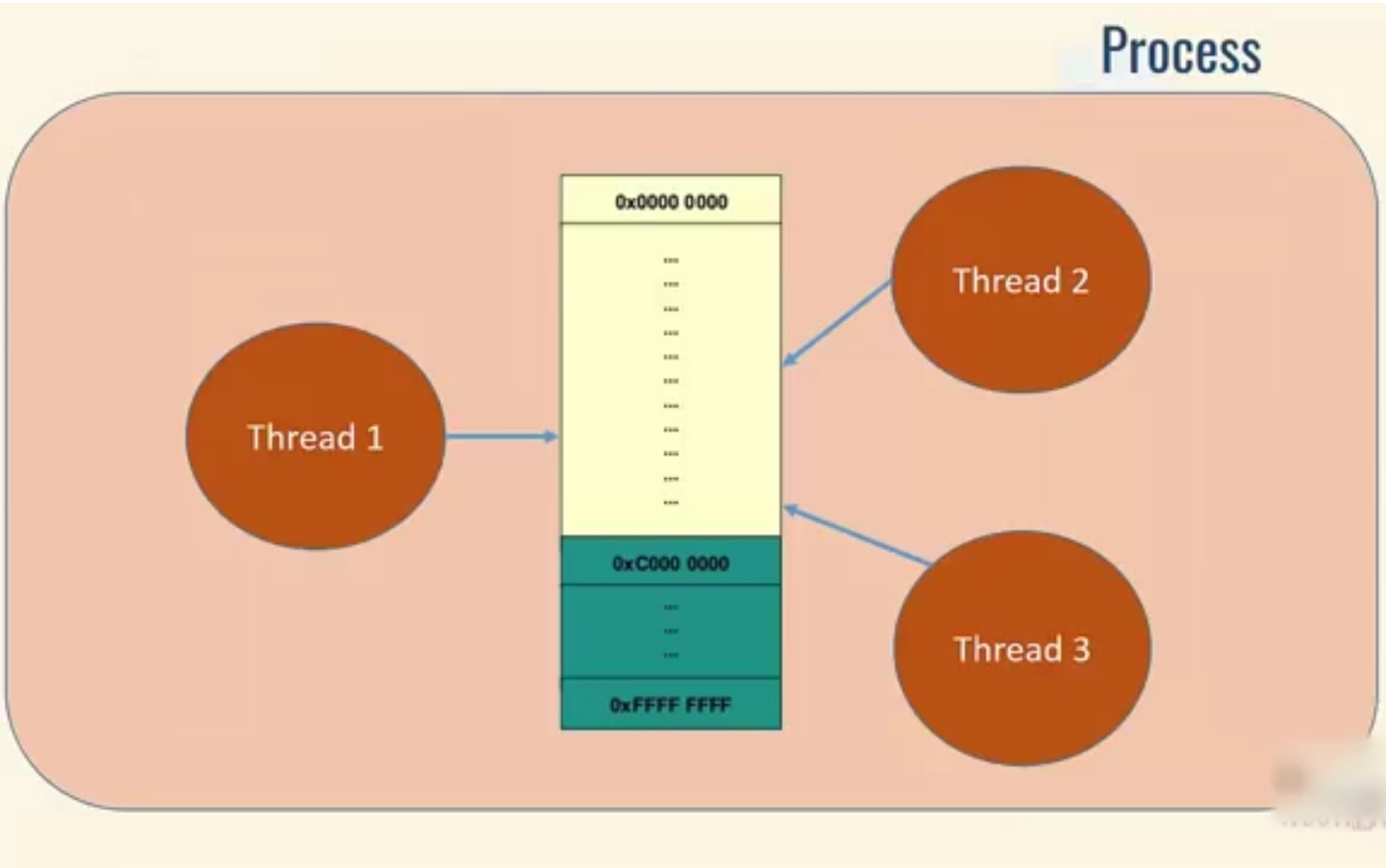
6. The user level threads are very fast.

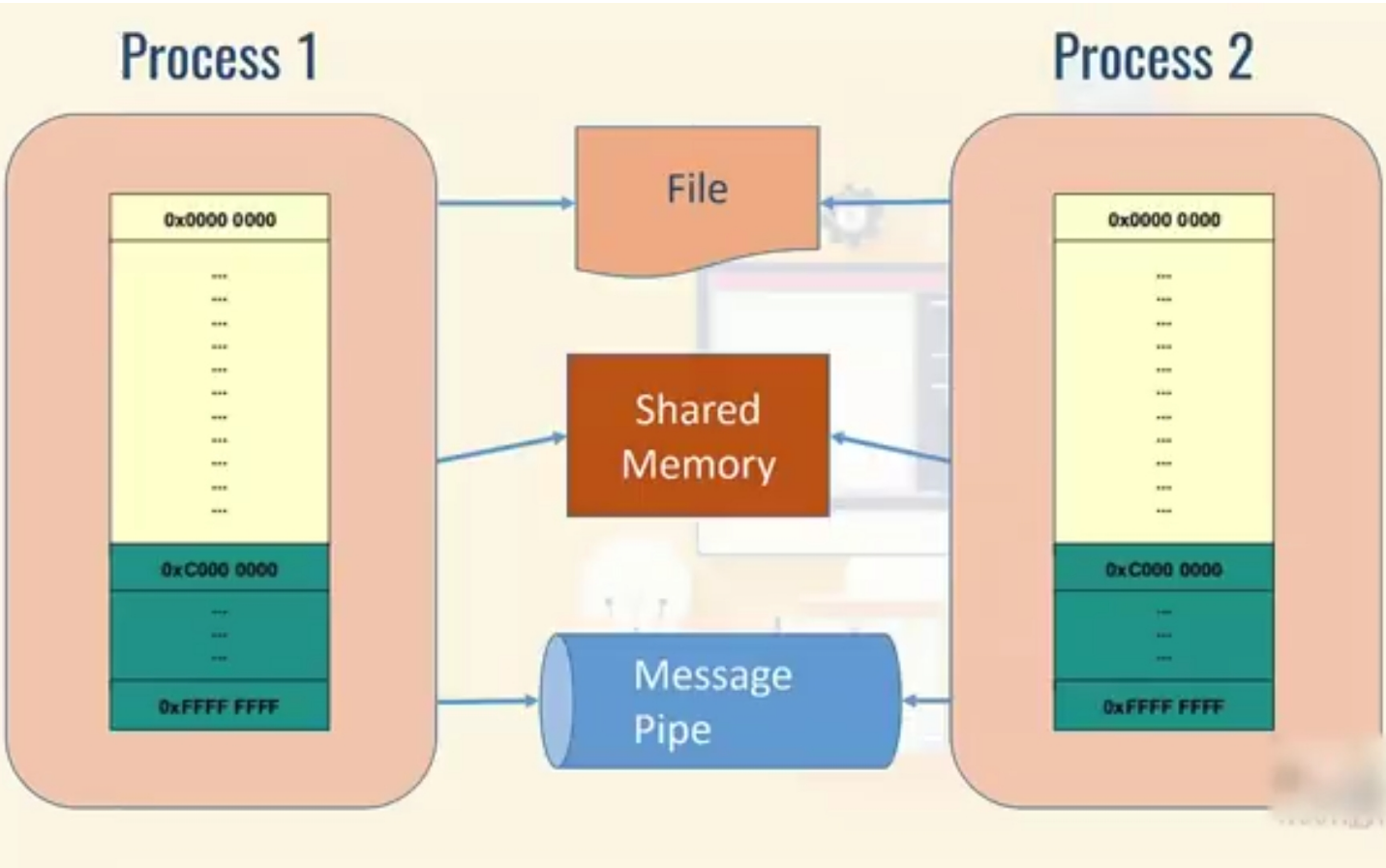

Multiprocessing and multithreading
both are ways to achieve
multitasking


Cooking
food
Take
care
of baby
Phone
call


Task Manager
Format Background


Process
0x0000 0000
Thread 1
Thread 2
Thread 3
0xC000 0000
0xFFFF FFFF


Process
0x0000 0000
Thread 1
Thread 2
Thread 3
0xC000 0000
0xFFFF FFFF


Process 1
Process 2
File
Shared Memory
Message Pipe
0x0000 0000
0xC000 0000
0xFFFF FFFF
0x0000 0000
0xC000 0000
0xFFFF FFFF

Multitasking and Multithreading

Multithreading :

Executing multiple threads simultaneously

Single process → subset

Multiple threads

Execute simultaneously

eg word → Spellcheck (2)

→ Type (1)

Save time

Multitasking :

Multitasking and Multithreading

Multithreading :

Executing multiple threads simultaneously

Single process → Subset

Multiple threads

Execute simultaneously

eg word → Spellcheck (2)

→ Type (1)

Save time

CPU switches between the threads frequently.

Multitasking :

Multiple task executed simultaneously

Using Time-sharing

CPU switches between processes frequently

Multitasking and Multithreading

Multithreading :

Executing multiple threads simultaneously

Single process → Subset

Multiple threads

Execute simultaneously

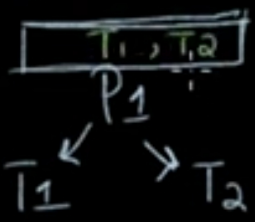
eg word → Spellcheck (2)

→ Type (1)

Save time

CPU switches between the threads frequently.

all threads of a process shares the memory



CPU

→ Maximum Utilize

Multitasking :

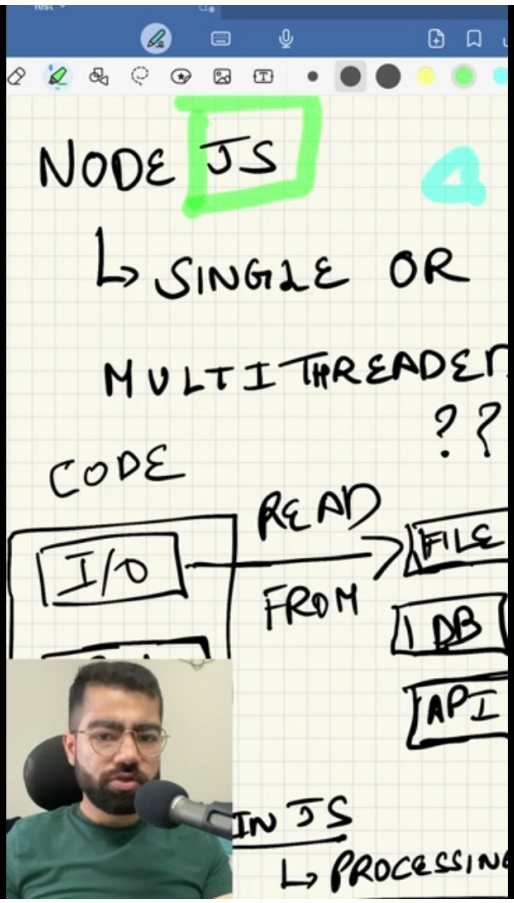
Multiple task executed simultaneously

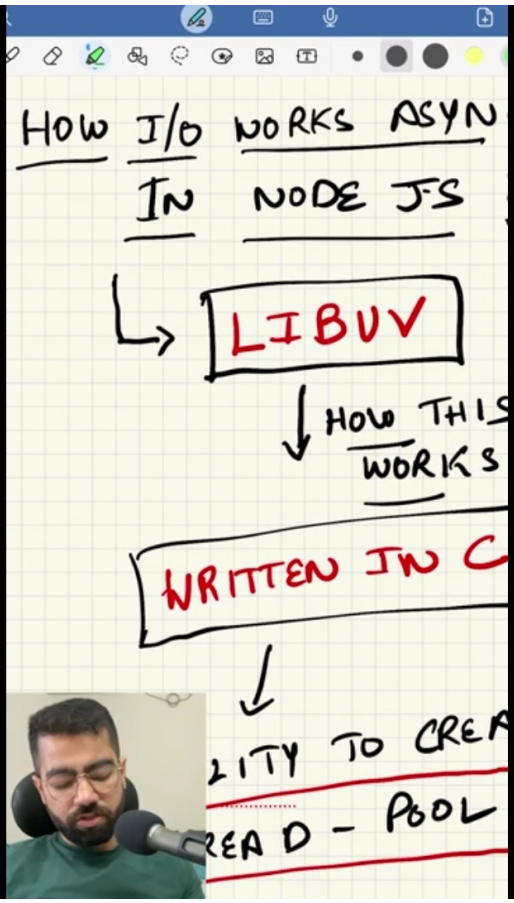
Using Time-sharing

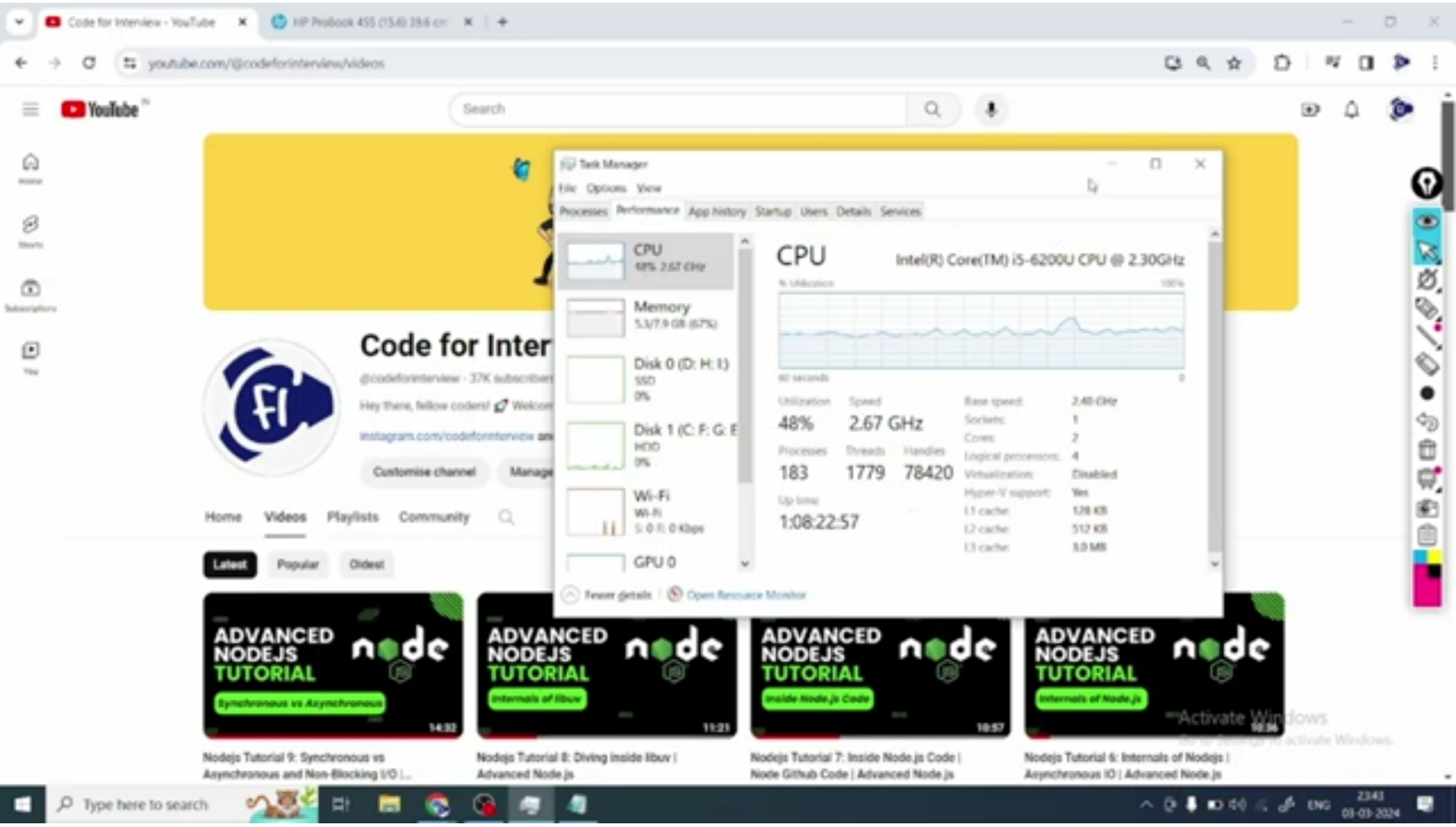
CPU switches between processes frequently

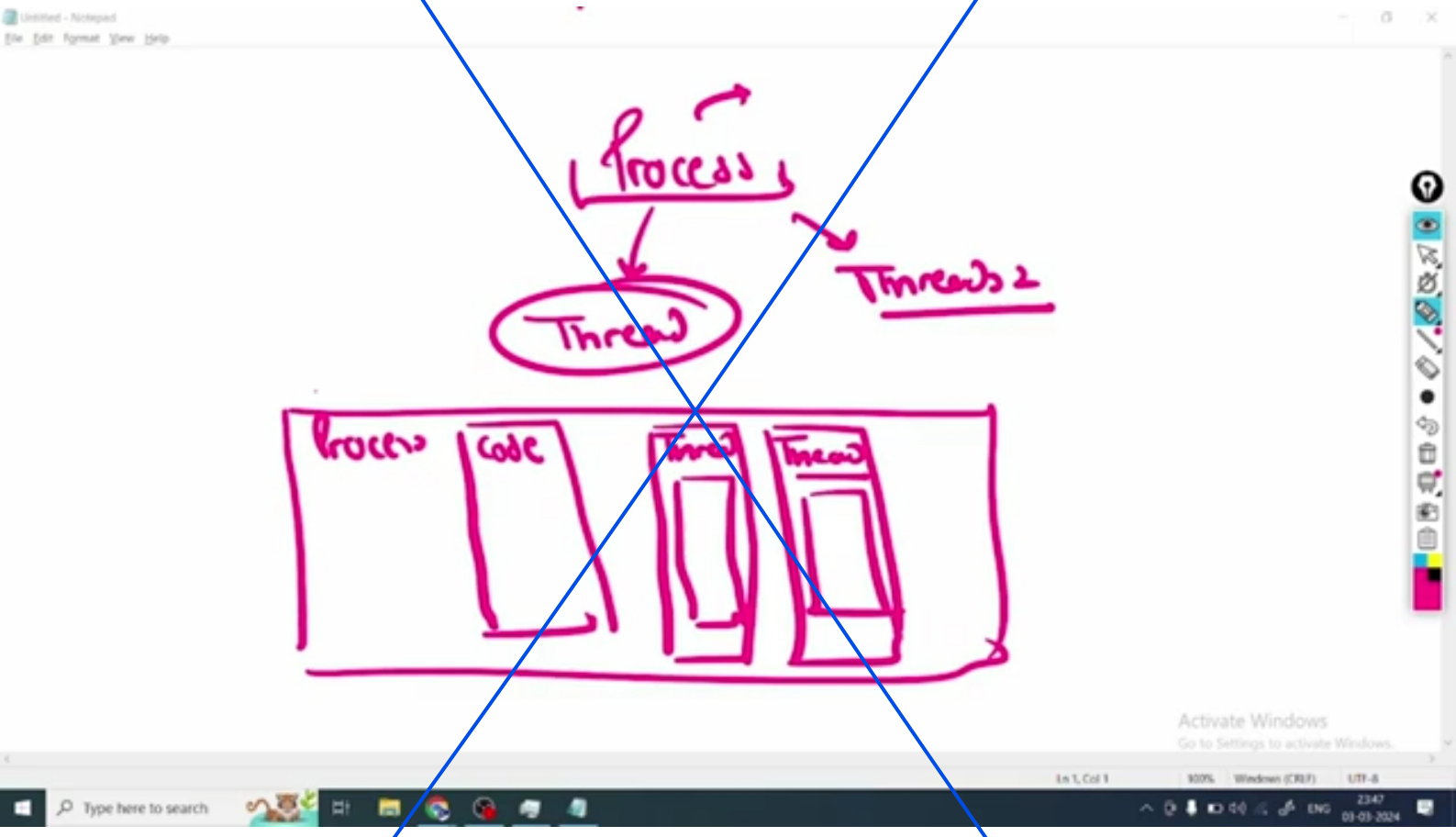
Save Time

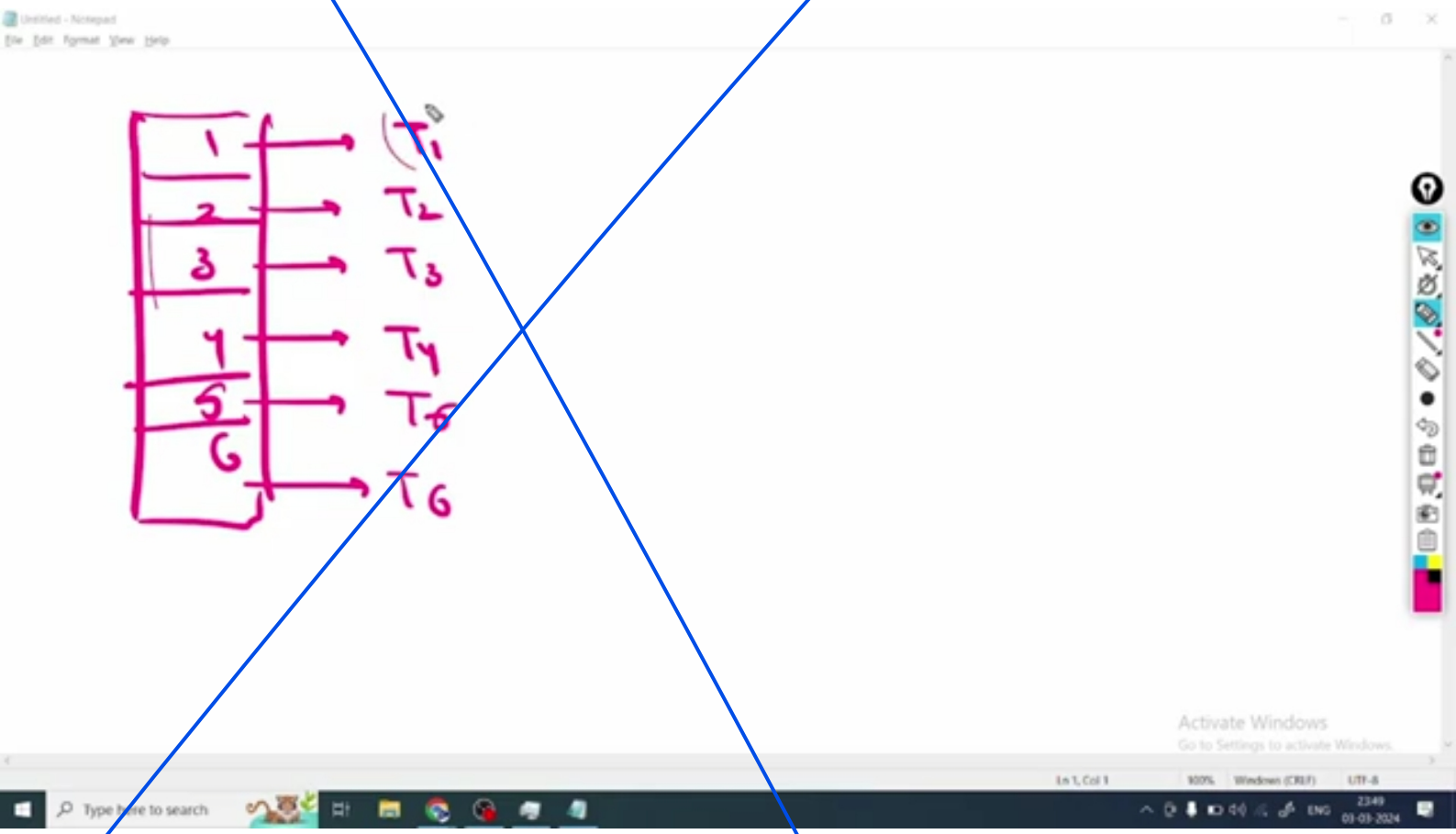
Allocate separate memory

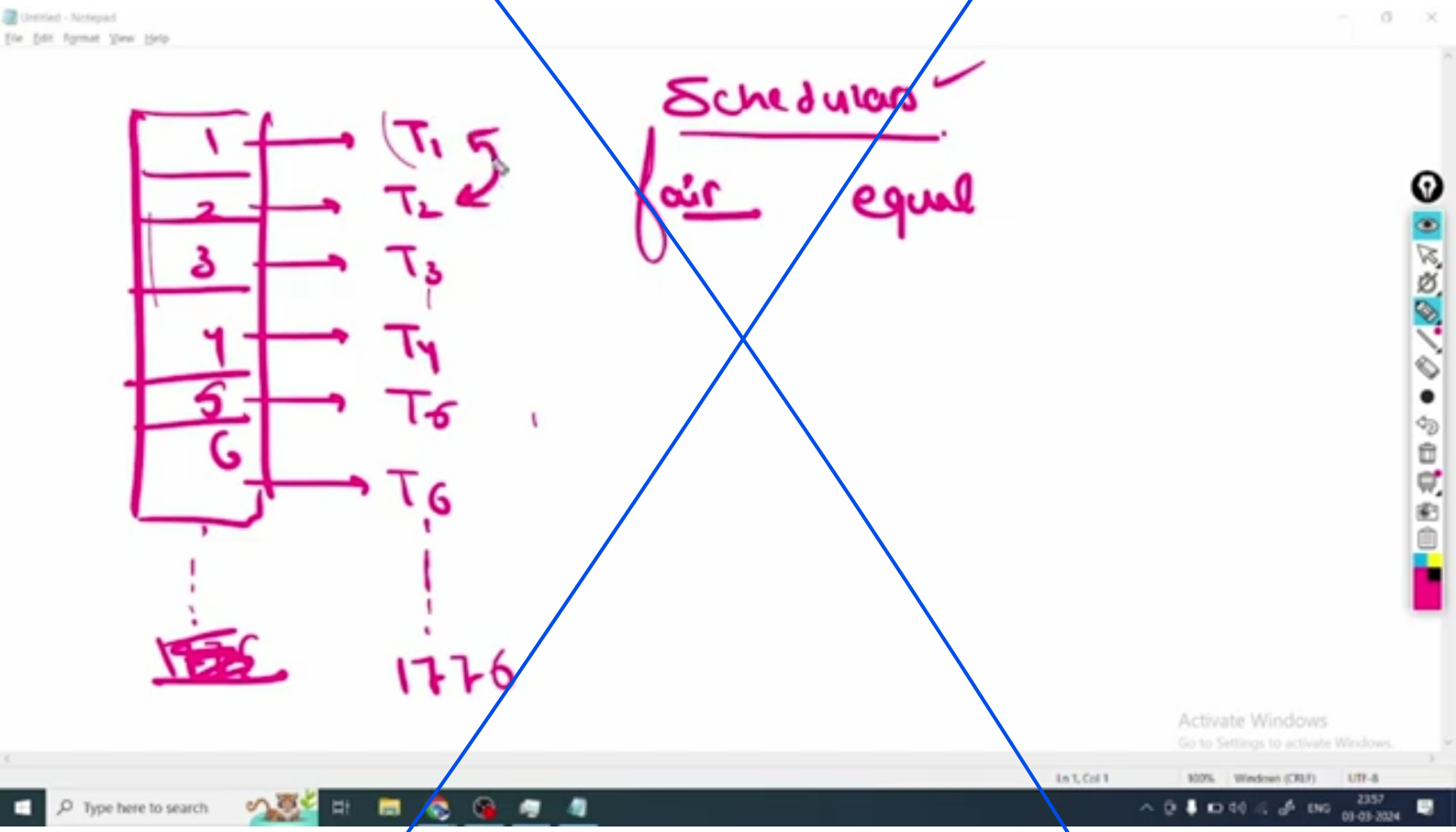

NODE JS
↳ SINGLE OR
MULTITHREADED
??
CODE
I/O
READ
FROM
FILE
DB
API
IN JS
↳ PROCESSING

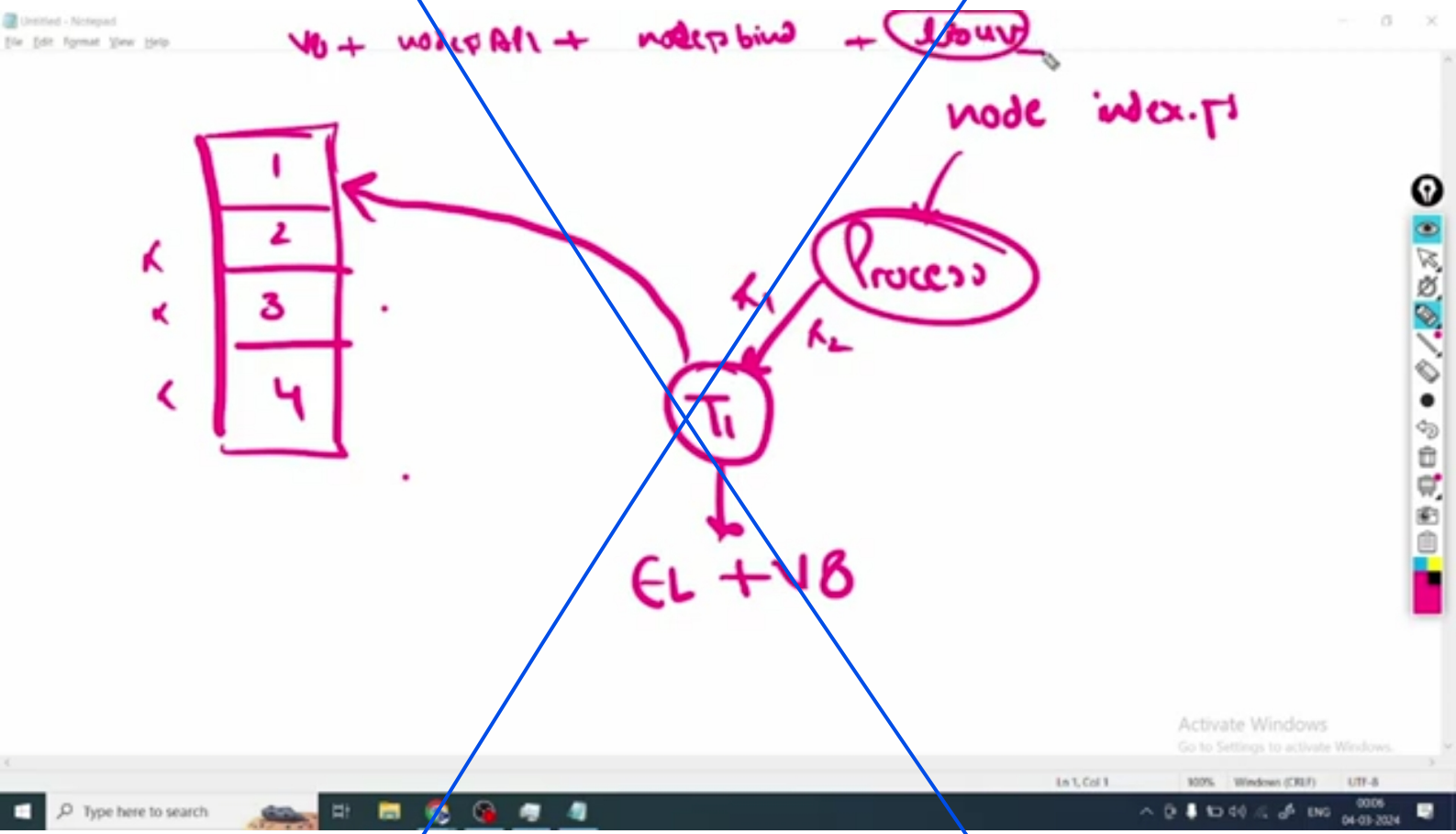

HOW I/O WORKS ASYN
IN NODE JS
LIBUV
HOW THIS
WORKS
WRITTEN IN C
ITY TO CREA
READ - POOL

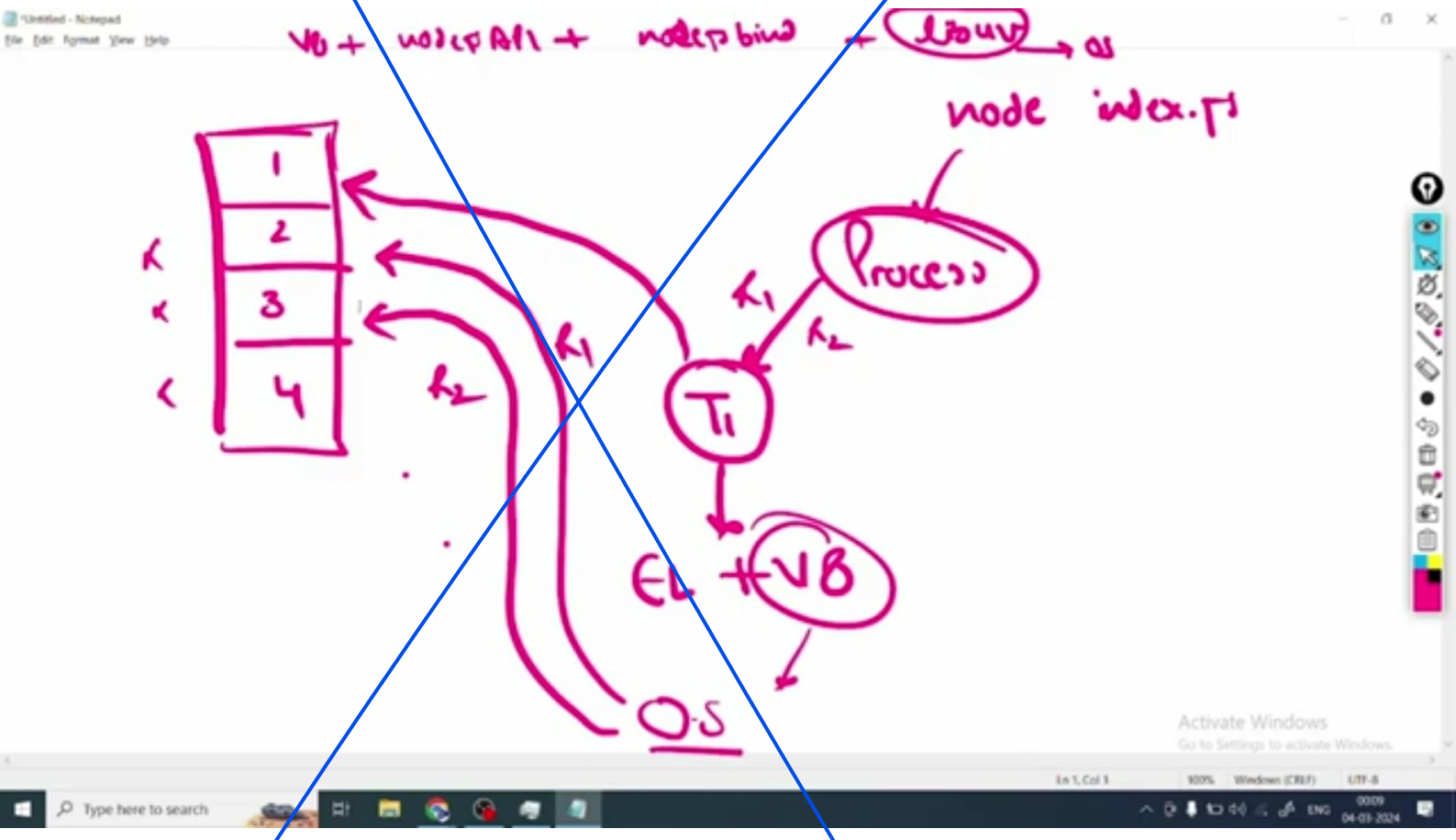

Code for Interview - YouTube
youtube.com/@codeforinterview/videos
YouTube
Search
Task Manager
File Options View
Processes Performance App history Startup Users Details Services
CPU
48% 2.67 GHz
Memory
5.3/7.9 GB (67%)
Disk 0 (D: H: I:)
SSD
0%
Disk 1 (C: F: G: E
HDD
0%
Wi-Fi
Wi-Fi
S: 0 R: 0 Kbps
GPU 0
Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz
Utilization
48%
Speed
2.67 GHz
Processes
183
Threads
1779
Handles
78420
Up time
1:08:22:57
Base speed: 2.40 GHz
Sockets: 1
Cores: 2
Logical processors: 4
Virtualization: Disabled
Hyper-V support: Yes
L1 cache: 128 KB
L2 cache: 512 KB
L3 cache: 3.0 MB
Fewer details
Open Resource Monitor
Code for Inter
@codeforinterview · 37K subscribers
Hey there, fellow coders!
instagram.com/codeforinterview
Customise channel
Home
Videos
Playlists
Community
Latest
Popular
Oldest
ADVANCED NODEJS TUTORIAL
Synchronous vs Asynchronous
14:32
Nodejs Tutorial 9: Synchronous vs Asynchronous and Non-Blocking I/O |...
ADVANCED NODEJS TUTORIAL
Internals of libuv
11:21
Nodejs Tutorial 8: Diving inside libuv | Advanced Node.js
ADVANCED NODEJS TUTORIAL
Inside Node.js Code
10:57
Nodejs Tutorial 7: Inside Node.js Code | Node Github Code | Advanced Node.js
ADVANCED NODEJS TUTORIAL
Internals of Node.js
10:06
Nodejs Tutorial 6: Internals of Nodejs | Asynchronous IO | Advanced Node.js
Activate Windows
Type here to search
ENG
23:43
03-03-2024

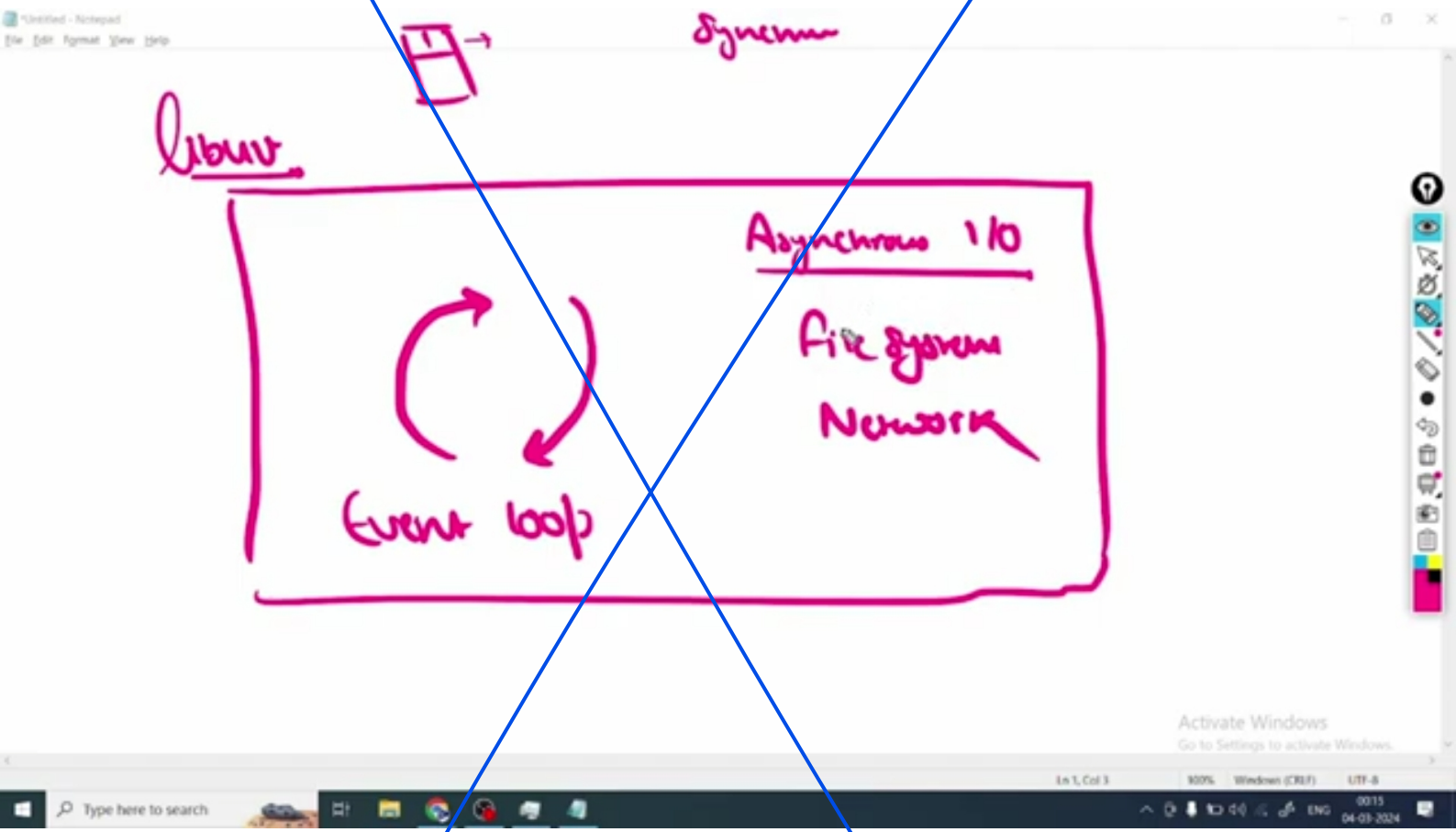

Untitled - Notepad
File Edit Format View Help
Process
Threads 2
Thread
Process
code
Thread
Thread
Activate Windows
Go to Settings to activate Windows.
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8
Type here to search
ENG
23:47
03-03-2024

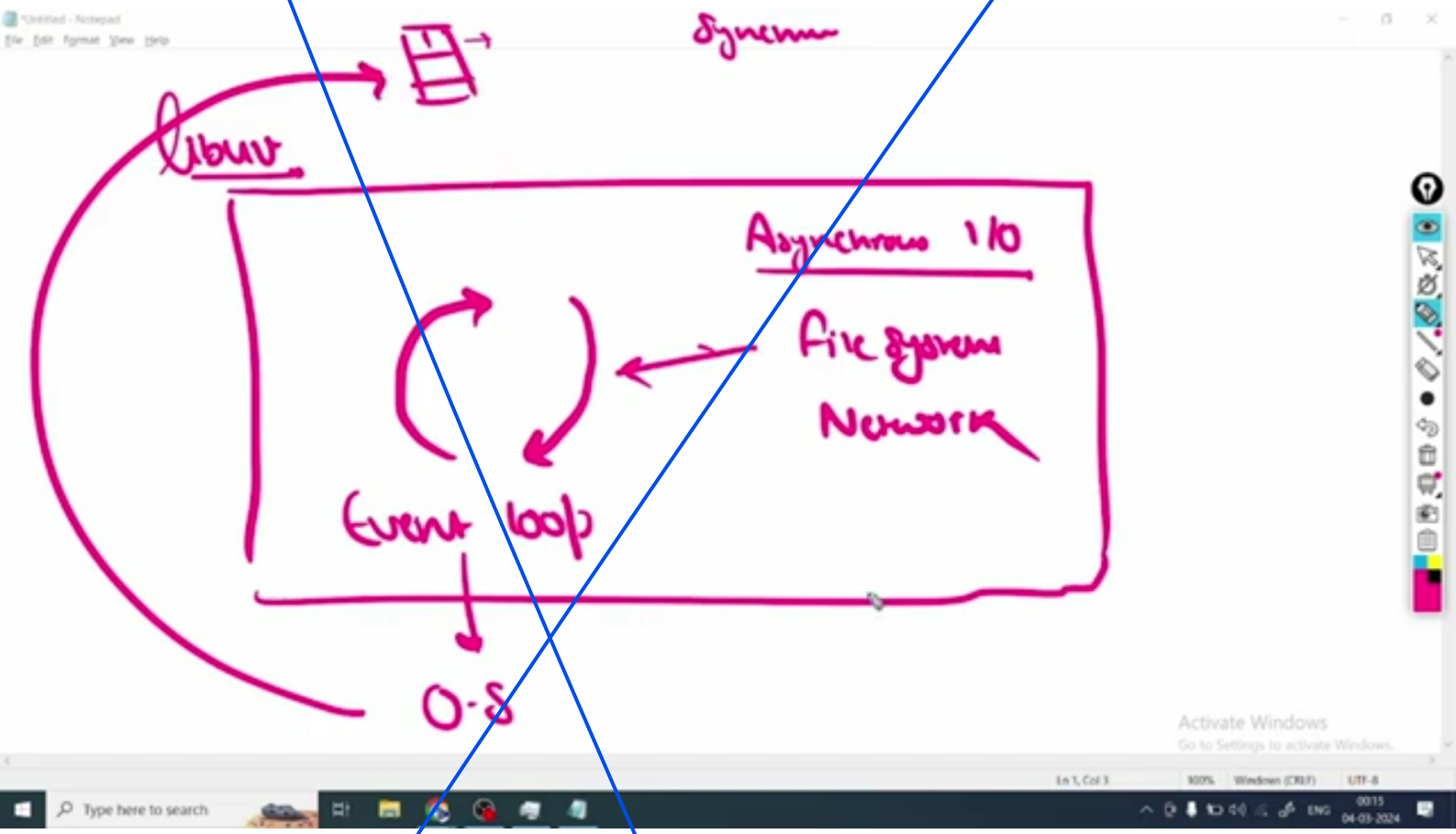

Untitled - Notepad
File Edit Format View Help
1
2
3
4
5
6
T1
T2
T3
T4
T5
T6
Activate Windows
Go to Settings to activate Windows.
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8
Type here to search
ENG
23:49
03-03-2024

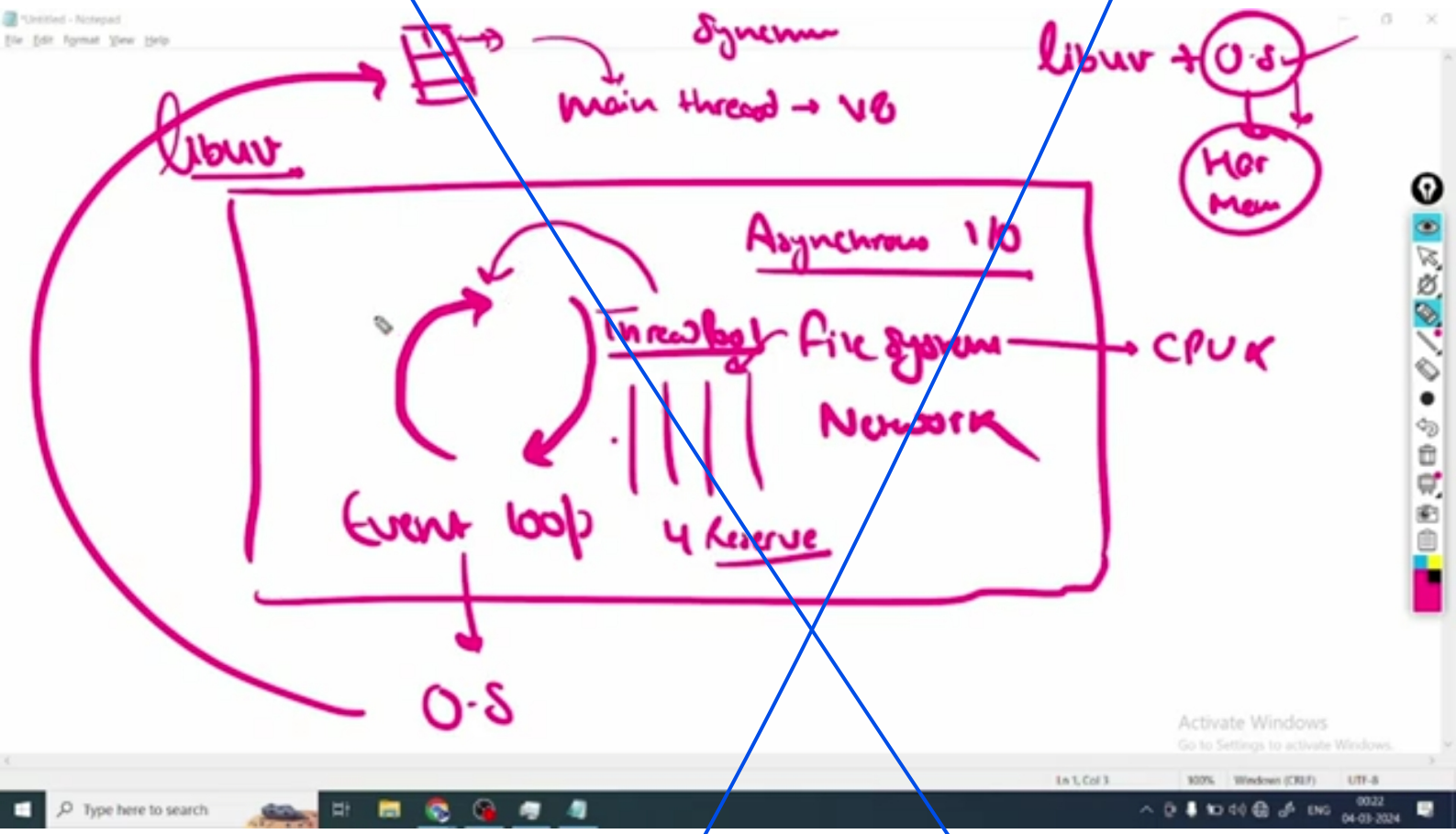

Scheduler
fair
equal
1
2
3
4
5
6
T1
T2
T3
T4
T5
T6
1776
Activate Windows
Type here to search


node index
1
2
3
4
Process
T_1
EL + VB


node index.js
1
2
3
4
Process
T1
V8
OS


System
libuv
Asynchronous I/O
File System
Network
Event loop


Libuv
Async I/O
File System
Network
Event loop
O.S

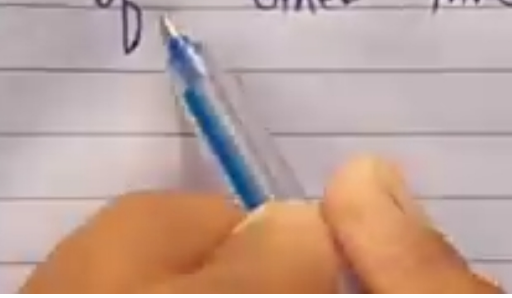

Synchronous
main thread → V8
libuv
libuv → O.S
Net
Mem
Asynchronous I/O
Threadpool
File system
CPU
Network
Event loop
4 Reserve
O.S

V.V.I

Multithreading

Q. What is multithreading ? With example.

Ans → multithreading is a process to execute multiple threads at the same time without dependency of other threads called multithreading.



of other threads called multithreading.

main() {

deposit()

withdraw()

chuck balls()

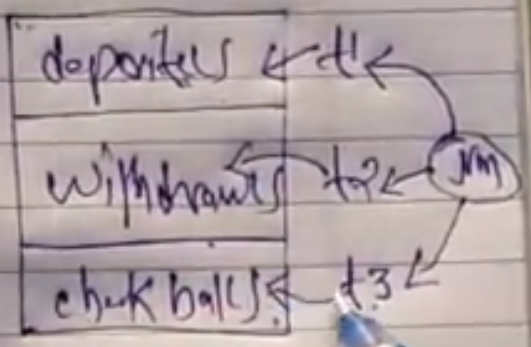
}

Q. What is thread?

of other threads called multithreading.

main()

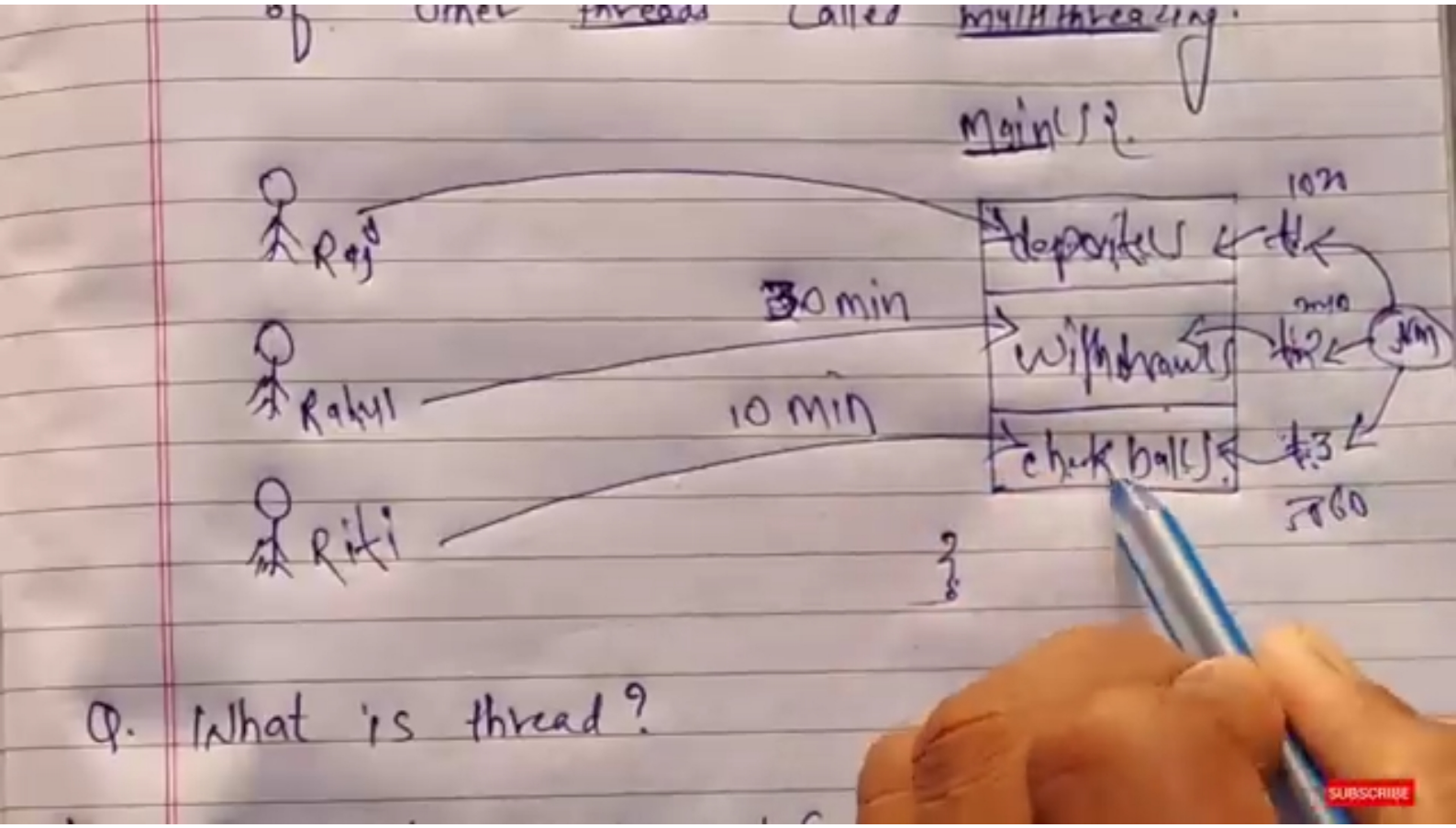
30 min



}

Q. What is thread?

of other threads called Multithreading.



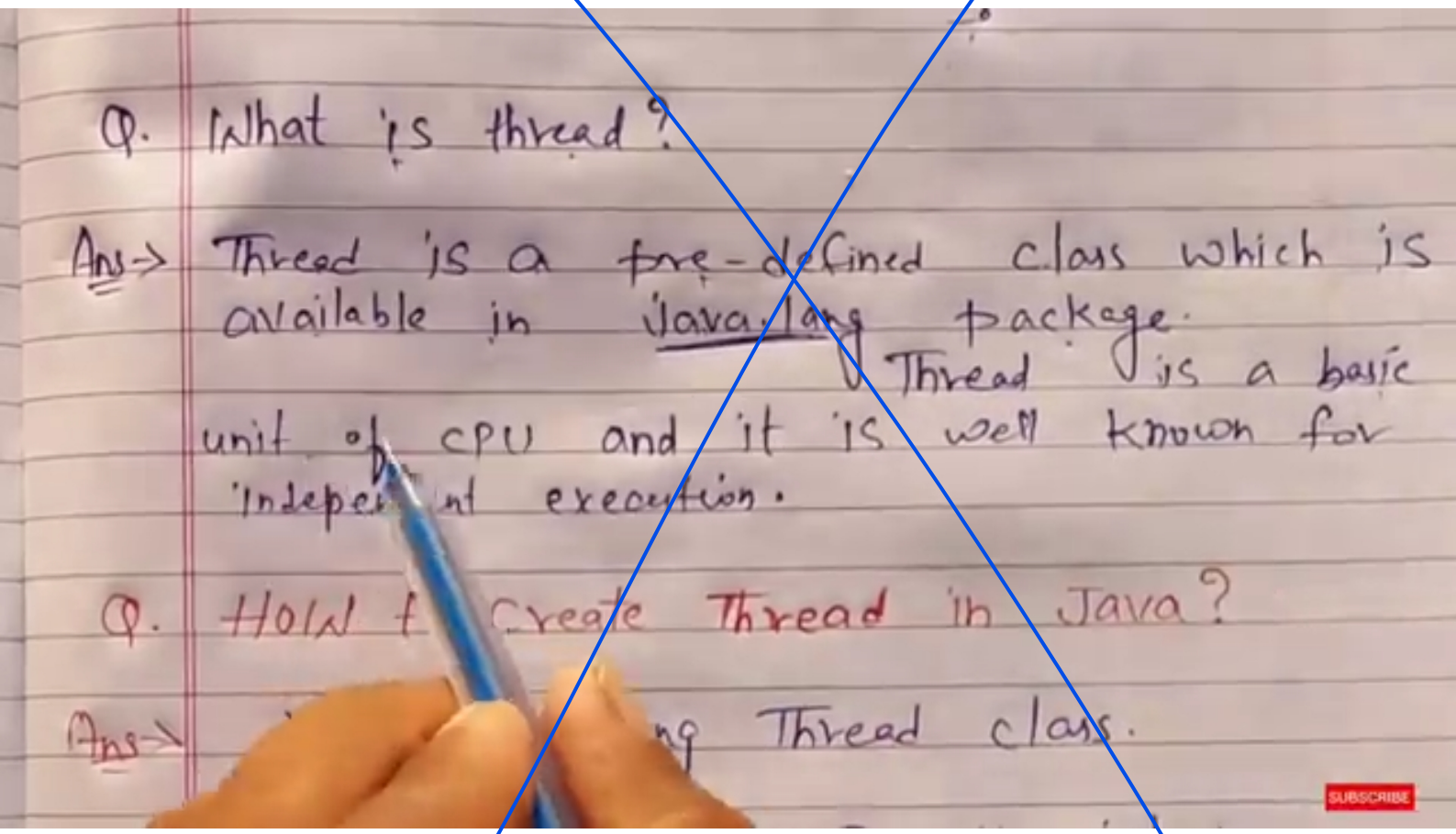
Q. What is thread?

Q. What is thread?

Ans-> Thread is a pre-defined class which is available in java.lang package.
Thread is a basic unit of CPU and it is well known for independent execution.

Q. How to create Thread in Java?

Ans-> ...ng Thread class.



Q. What is thread?

Ans-> Thread is a pre-defined class which is available in java.lang package.
Thread is a basic unit of CPU and it is well known for independent execution.

Q. How to Create Thread in Java?

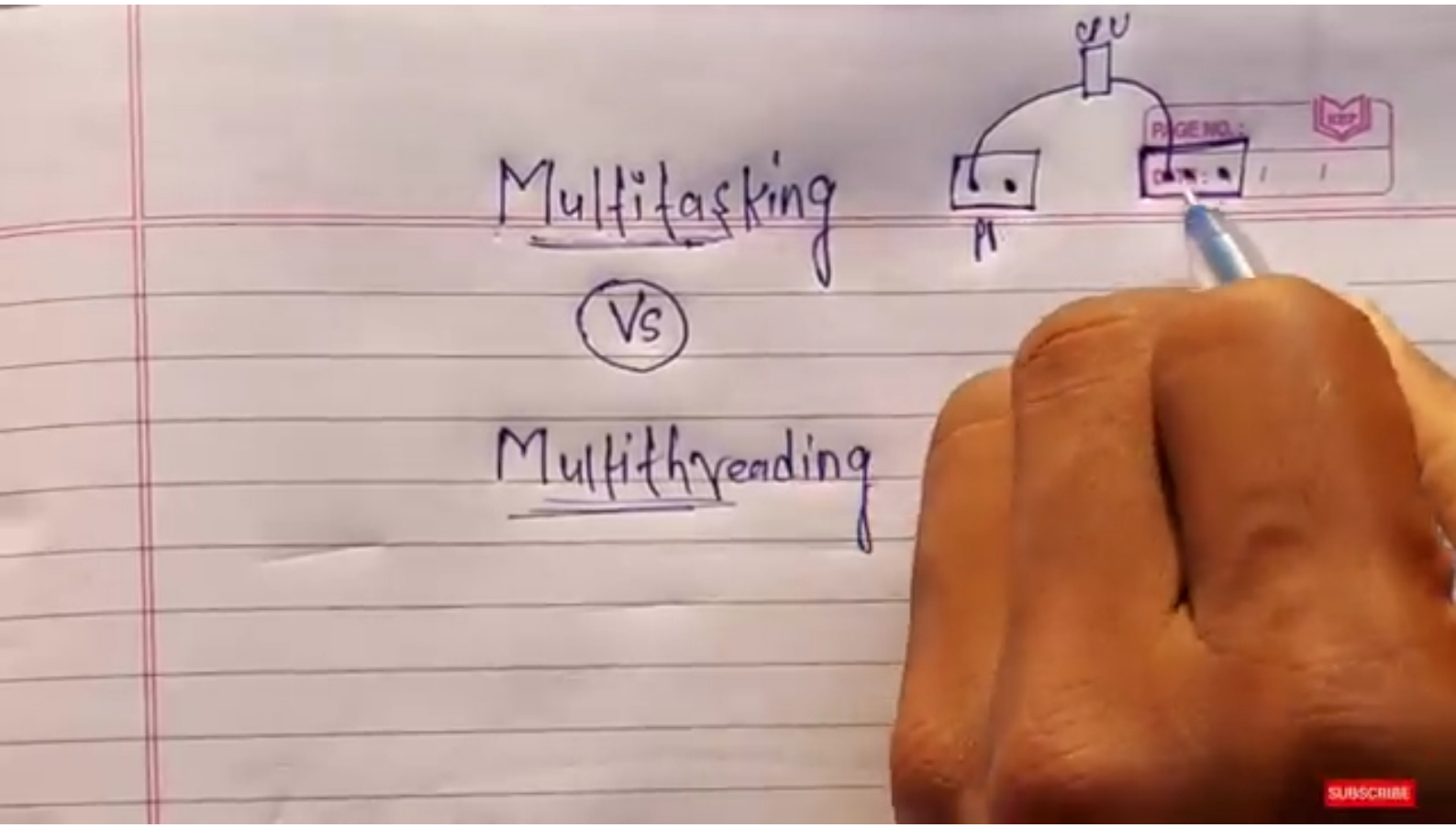
Ans-> 1) By extending T

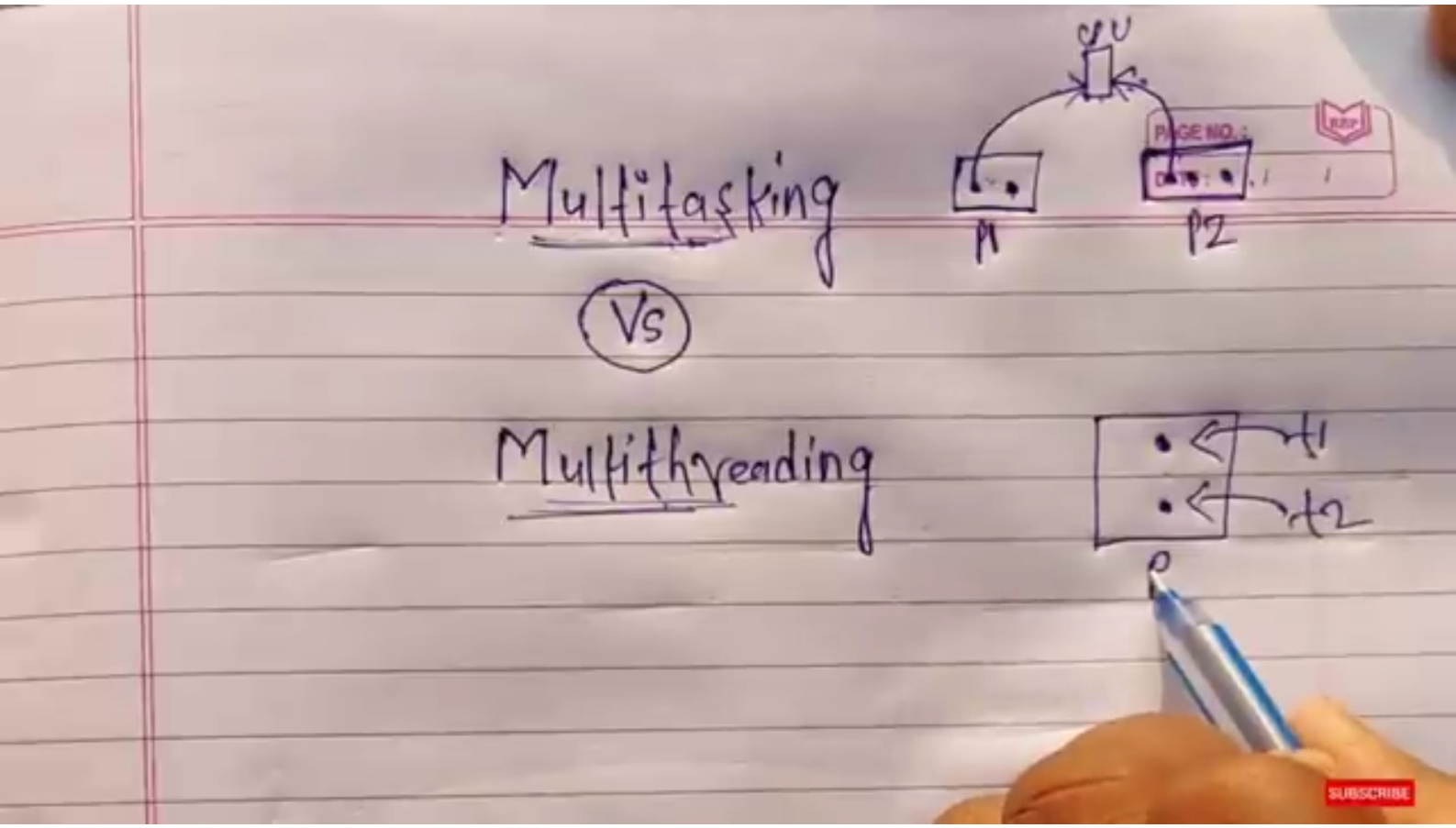
Ans-> Thread is a pre-defined class which is available in java.lang package.
Thread is a basic unit of CPU and it is well known for independent execution.

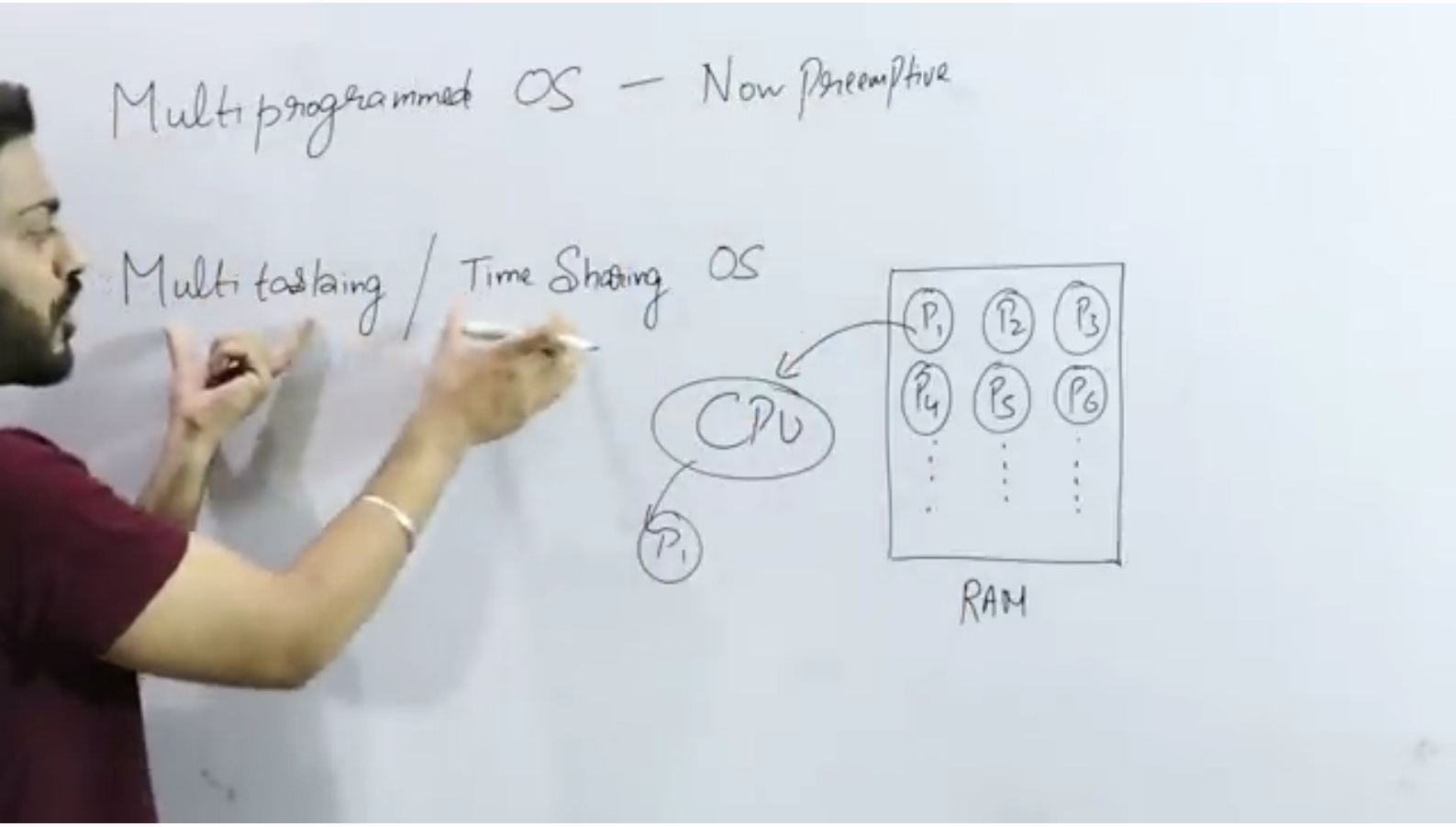
Q. How to create Thread in Java?

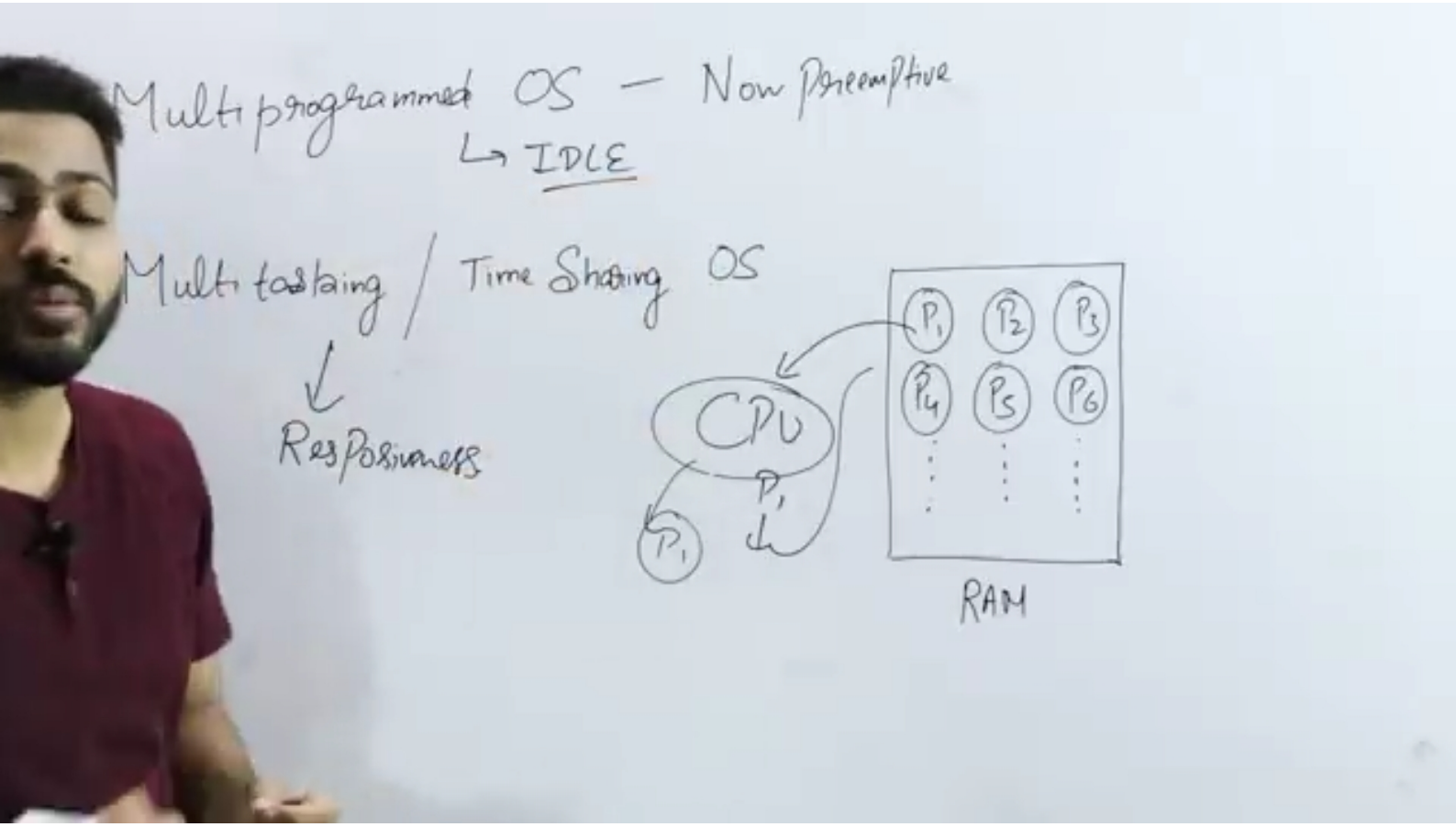
Ans-> 1) By extending Thread class.

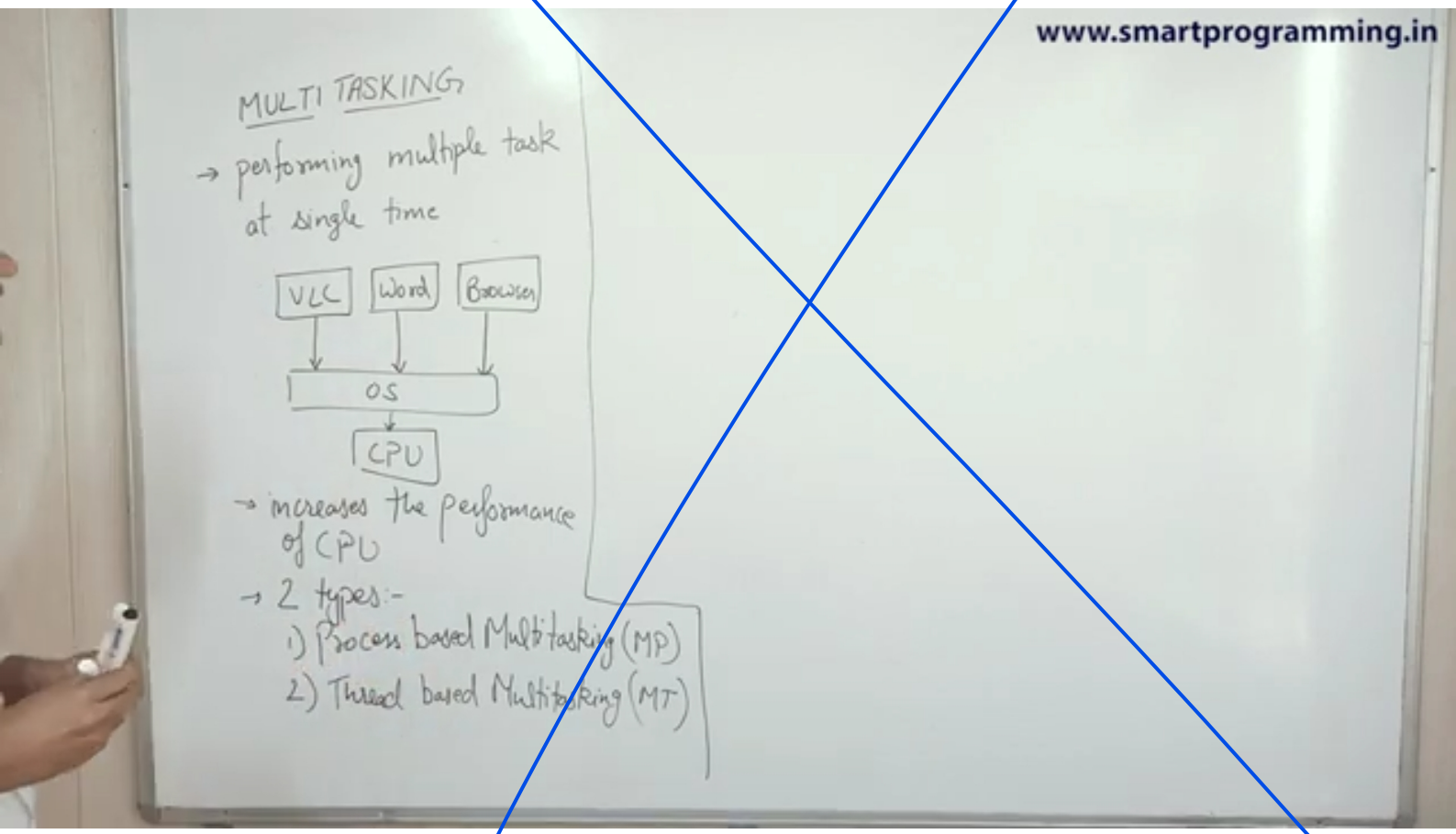
ii) By implementing Runnable interface.


Multitasking
Vs
Multithreading
CPU
P1
SUBSCRIBE


Multitasking
Vs
Multithreading
CPU
PAGE NO.
P1
P2
t1
t2
SUBSCRIBE

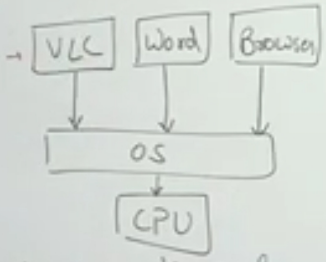

Multiprogrammed OS — Now Preemptive
Multitasking / Time Sharing OS
CPU
P1
P2
P3
P4
P5
P6
P1
RAM


Multiprogrammed OS — Now Preemptive
↳ IDLE
Multitasking / Time Sharing OS
↓
Responsiveness
CPU
P1
P1
P2
P3
P4
P5
P6
RAM


www.smartprogramming.in
MULTI TASKING
→ performing multiple task at single time
VLC
Word
Browser
OS
CPU
→ increases the performance of CPU
→ 2 types:-
1) Process based Multitasking (MP)
2) Thread based Multitasking (MT)

MULTI TASKING

Def → performing multiple task at single time

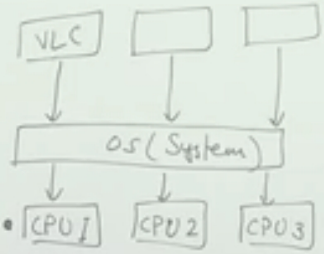


Use → increases the performance of CPU

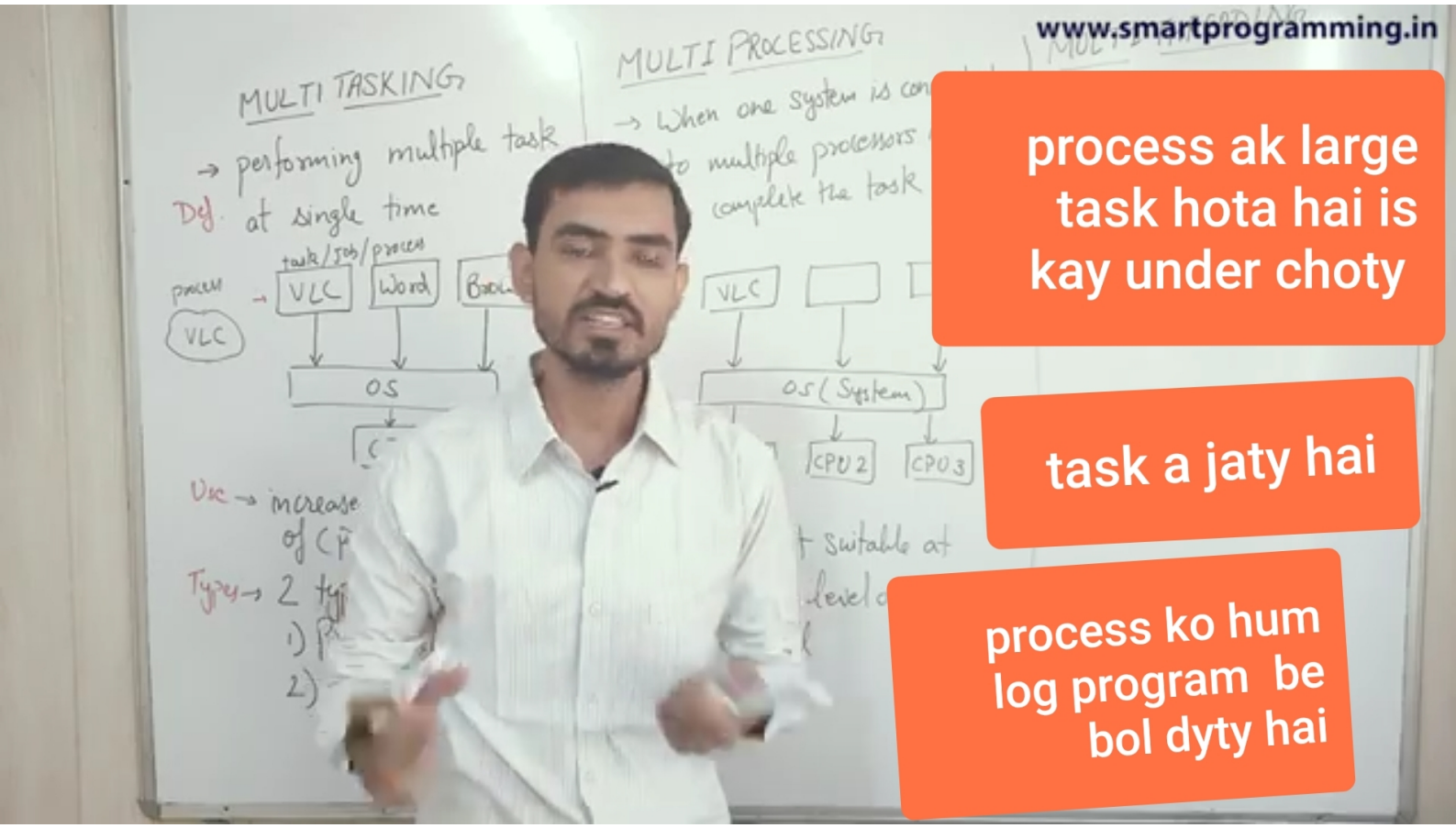
Types → 2 types:-
1) Process based Multitasking (MP)
2) Thread based Multitasking (MT)

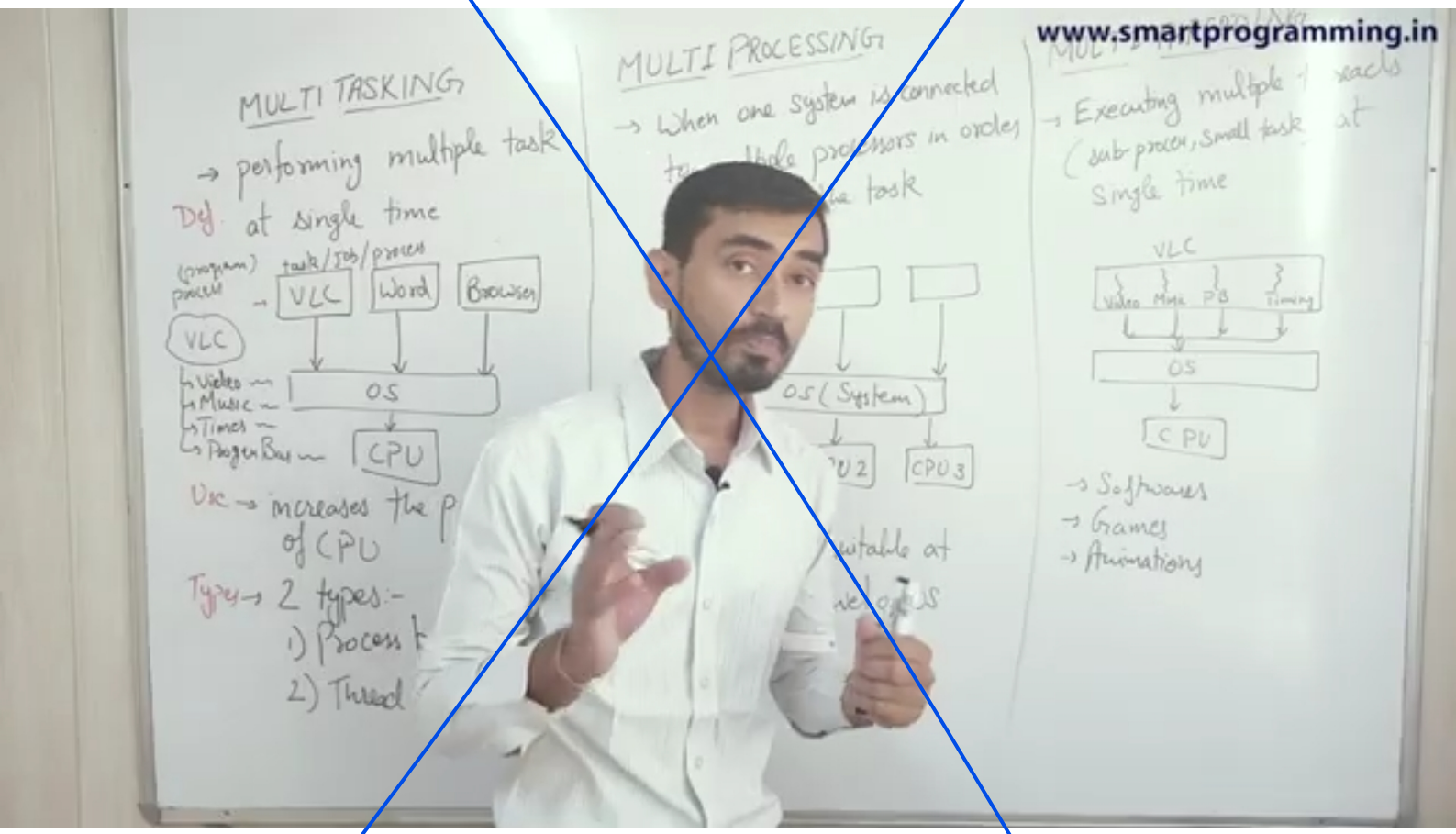
MULTI PROCESSING

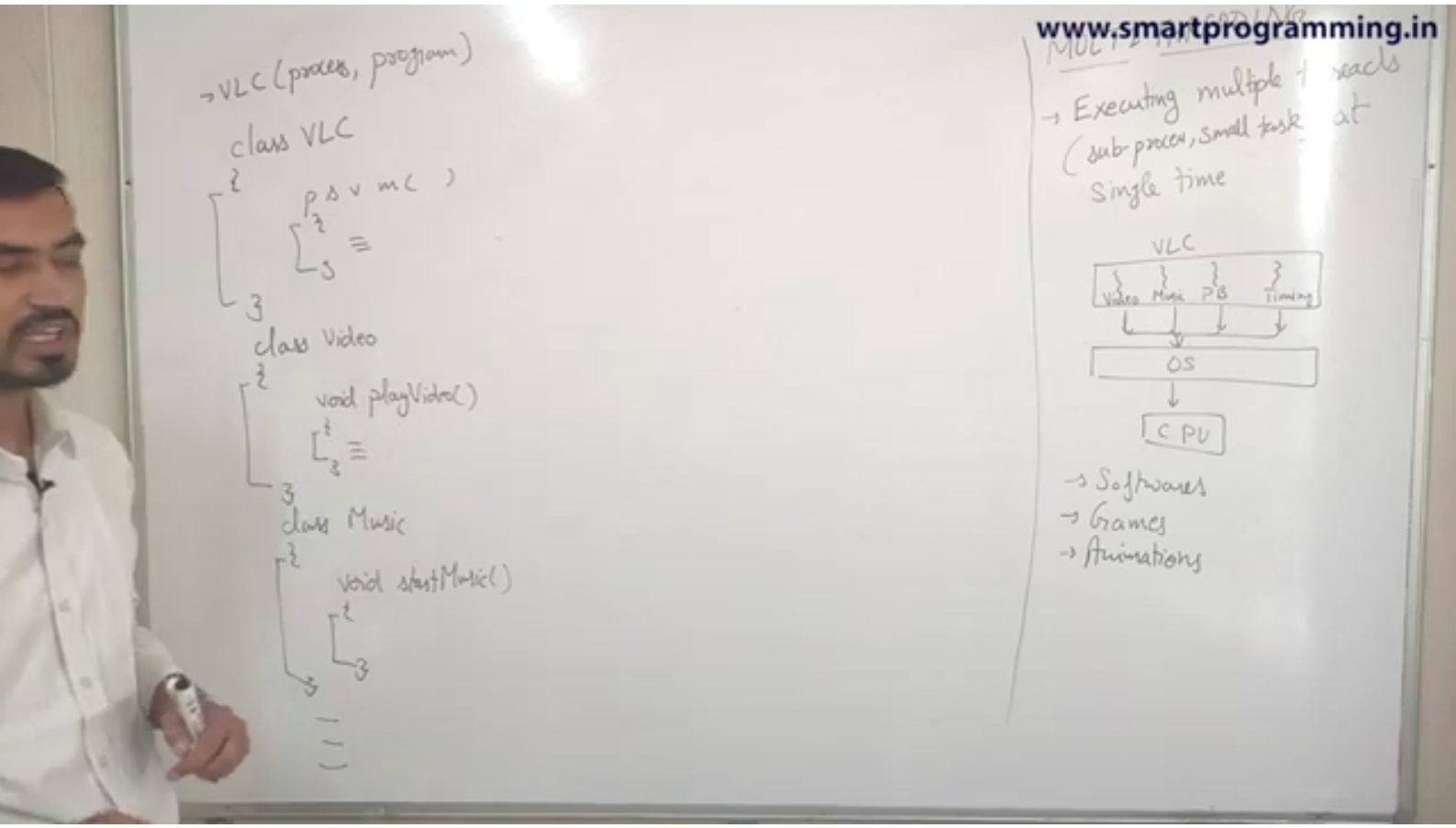
→ When one system is connected to multiple processors in order to complete the task

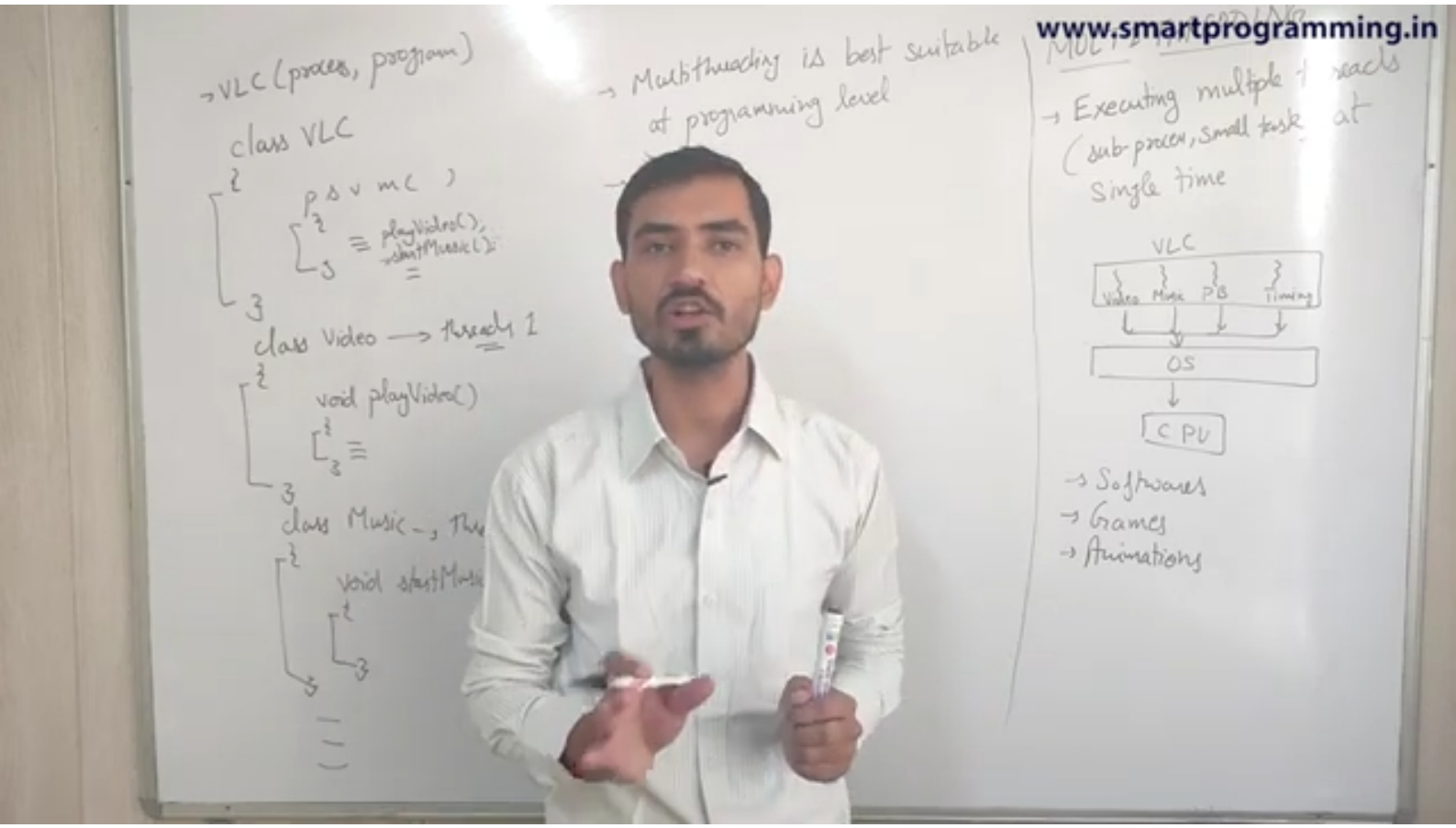


→ It is best suitable at system level or OS level


www.smartprogramming.in
MULTI TASKING
→ performing multiple task
Def. at single time
MULTI PROCESSING
→ When one system is con
to multiple processors
complete the task
VLC
Word
OS
VLC
OS (System)
CPU 2
CPU 3
process ak large task hota hai is kay under choty
task a jaty hai
process ko hum log program be bol dyty hai

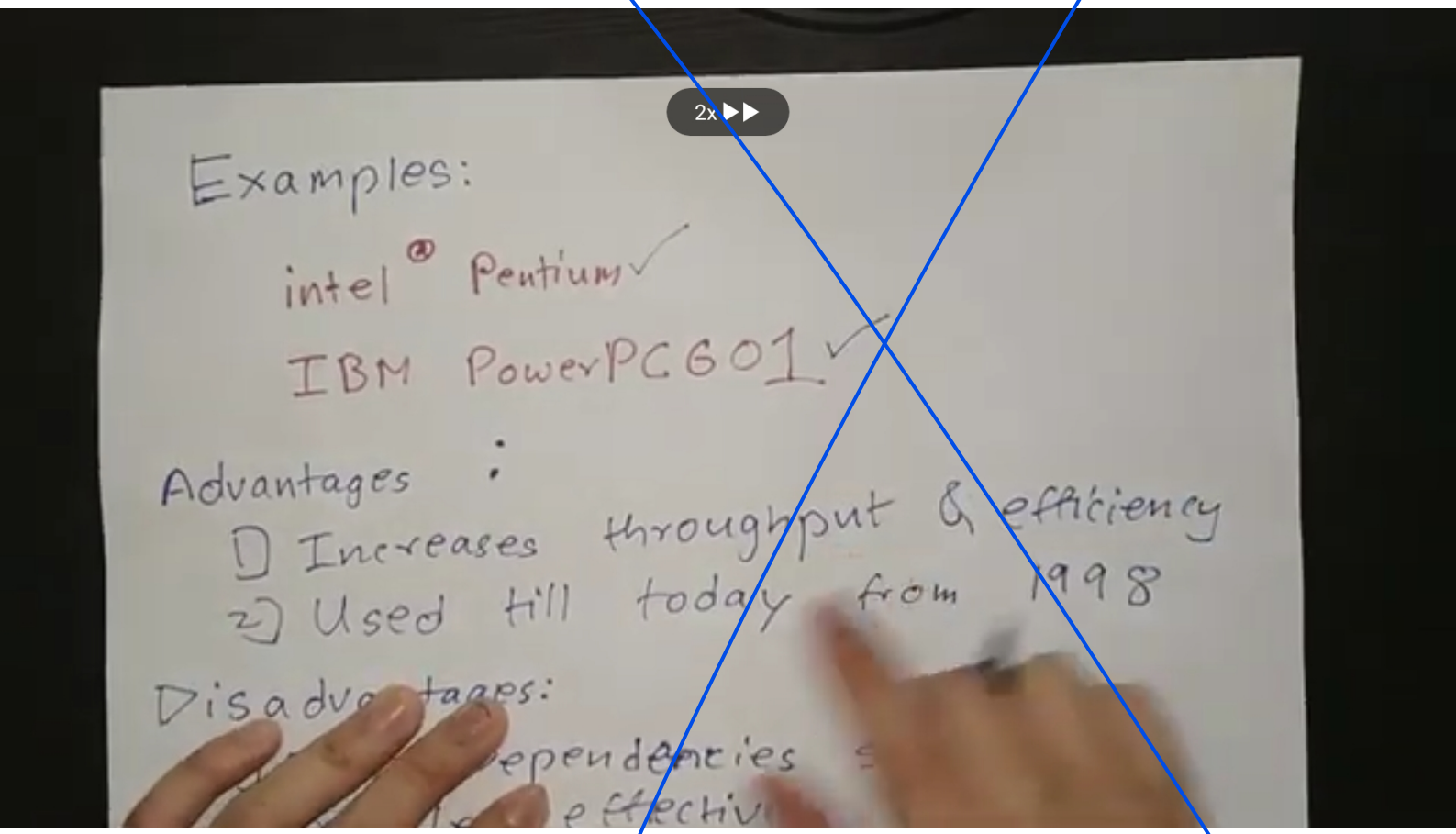

www.smartprogramming.in
MULTI TASKING
→ performing multiple task
Def at single time
(program) process
task/job/proces
VLC
Word
Browser
VLC
Video
Music
Timer
ProgressBar
OS
CPU
Use → increases the p
of CPU
Type → 2 types:-
1) Process
2) Thread
MULTI PROCESSING
→ When one system is connected
task
OS (System)
CPU 2
CPU 3
suitable at
VLC
Video
Music
PB
Timing
OS
CPU
→ Softwares
→ Games
→ Animations

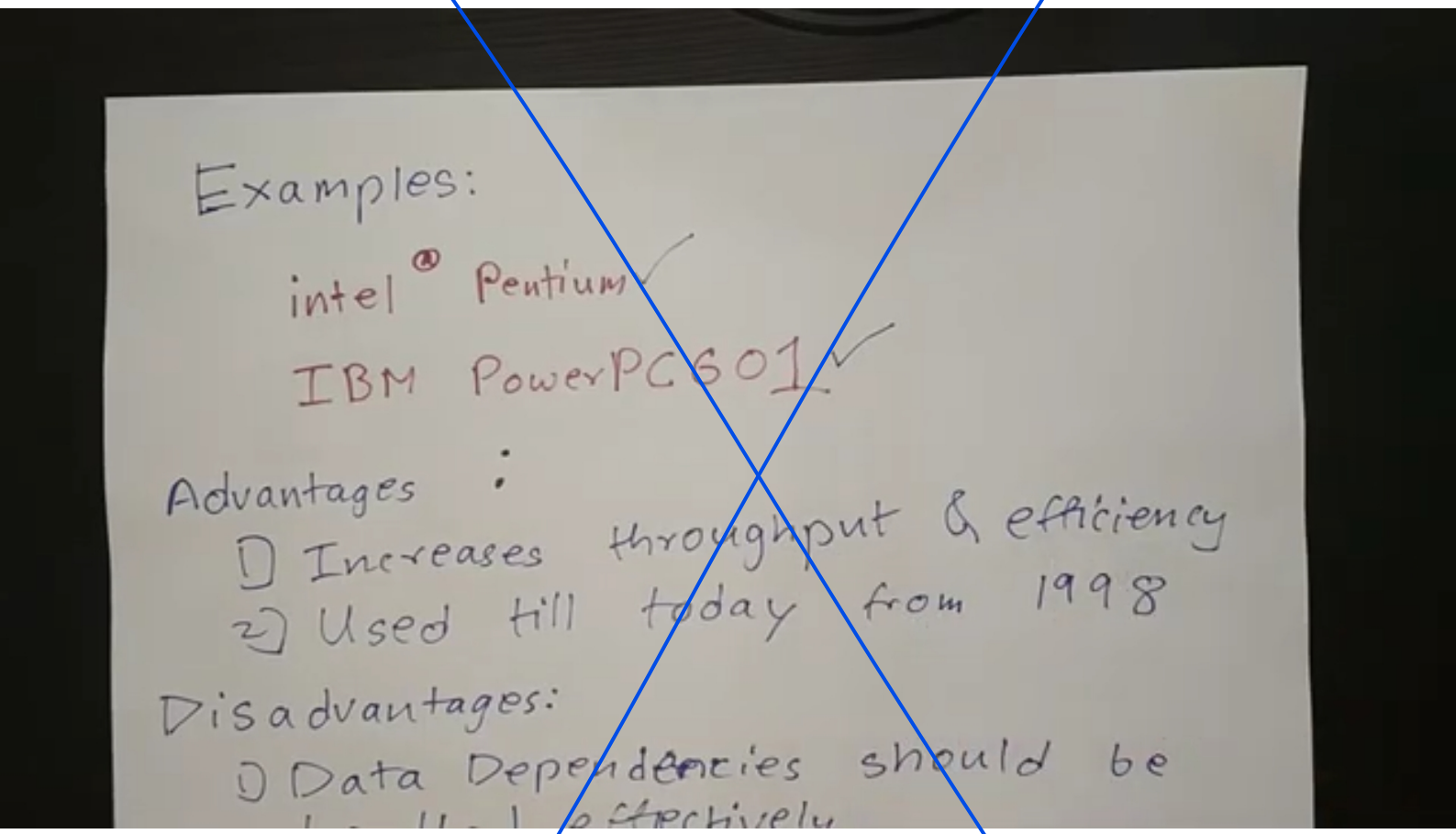

www.smartprogramming.in
→ VLC (process, program)
class VLC
{
psvm()
{
}
}
class Video
{
void playVideo()
{
}
}
class Music
{
void startMusic()
{
}
}
→ Executing multiple threads (sub-process, small task) at single time
VLC
Video
Music
PB
Timing
OS
CPU
→ Softwares
→ Games
→ Animations


www.smartprogramming.in
→ VLC (process, program)
class VLC
class Video → thread 1
void playVideo()
→ Multithreading is best suitable at programming level
→ Executing multiple threads (sub-process, small task) at Single time
VLC
Video
Music
PB
Timing
OS
CPU
→ Softwares
→ Games
→ Animations


www.smartprogramming.in
2x
→ VLC (proces, program)
class VLC
class Video
void playVid
class Mi
void
→ is best suitable
at programming level
→ Java provides predefined API
for multithreading
→ Executing multiple threads
(sub-proces, small task at
Single time
VLC
Video
Music
PB
Timing
OS
CPU
→ Softwares
→ Games
→ Animations


www.smartprogramming.in
2x
→ VLC (process, program)
class VLC
→ is best suitable at programming level
→ Java provides predefined API for multithreading
Thread, Runnable,
ThreadGroup,
Concurrency, Thread Pool
→ Executing multiple tasks (sub-process, small task) at Single time
VLC
Video
Music
PB
Timing
OS
CPU
→ Softwares
→ Games
→ Animations


2x
Examples:
intel® Pentium
IBM PowerPC601
Advantages :
1) Increases throughput & efficiency
2) Used till today from 1998
Disadvantages:


Examples:
intel® Pentium
IBM PowerPC601
Advantages :
1) Increases throughput & efficiency
2) Used till today from 1998
Disadvantages:
1) Data Dependencies should be

Examples:

intel® Pentium ✓

IBM PowerPC 601 ✓

Advantages :

1) Increases throughput & efficiency
2) Used till today from 1998

Disadvantages:

1) Data Dependencies should be handled effectively, 1unit time = more than

Superscalar

- 1st invented in 1987
- Superscalar processor executes multiple independent instructions in parallel.
- Common instructions (arithmetic, load/store etc) can be initiated simultaneously and executed independently.
- Applicable to both RISC & CISC, but usually in RISC

Why Superscalar

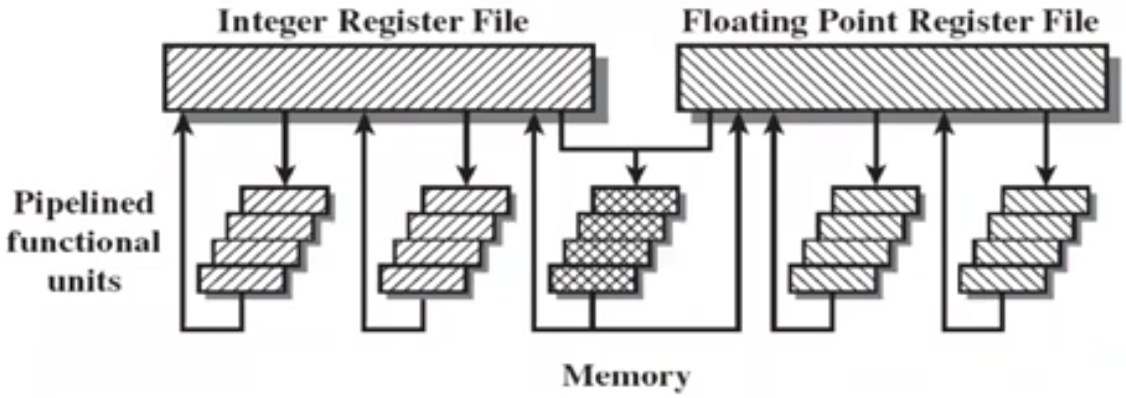
- Most operations are on scalar quantities
- Superscalar was designed to improve the performance of these operations by executing them concurrently in multiple pipelines

Superscalar Performance

- In superscalar multiple independent instruction pipelines are used. Each pipeline consists of multiple stages, so that each pipeline can handle multiple instructions at a time.
- A superscalar processor typically fetches multiple instructions at a time and then attempts to find nearby instructions that are independent of one another and can therefore be executed in parallel.
- If the input to one instruction depends on the output of a preceding instruction, then the latter instruction cannot complete execution at the same time or before the former instruction.

General Superscalar Organization

- The configuration below, supports the parallel execution of two integer operations, two floating point operations and one memory operation.



Super Pipeline

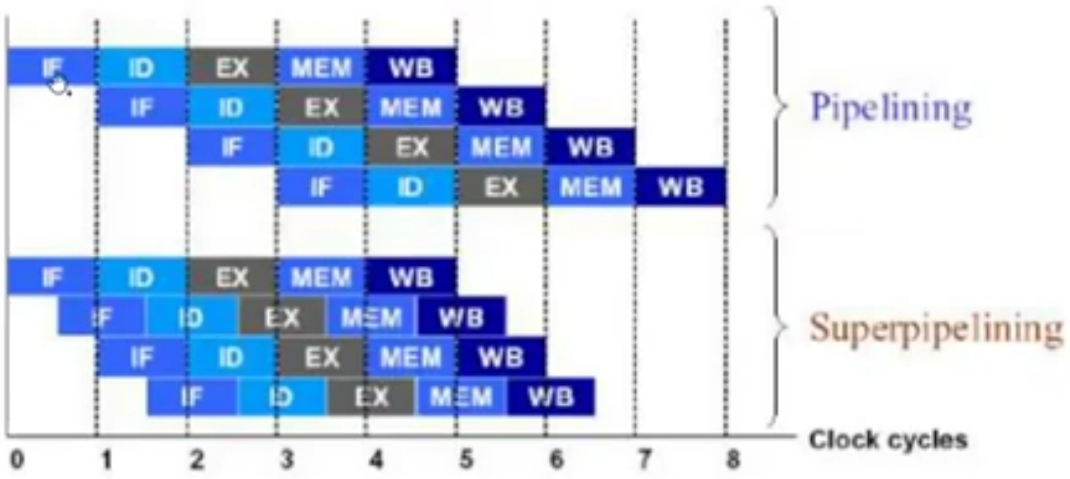
- Super-pipeline is an alternative approach to achieve greater performance
- 1st invented in 1988
- Many pipeline stages need less than half a clock cycle

Super pipeline cont...

- Super-pipelining is the breaking of stages of a given pipeline into smaller stages (thus making the pipeline deeper) in an attempt to shorten the clock period and thus enhancing the instruction throughput by keeping more and more instructions in flight at a time.

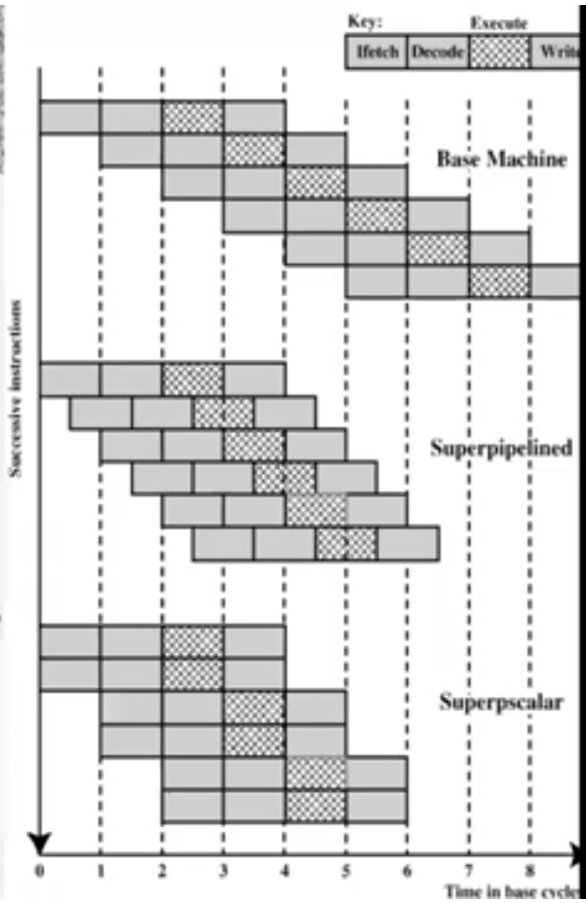
Super pipeline Performance

- The performance is shown below in the figure:



Superscalar versus super-pipeline

- Simple pipeline system performs only one pipeline stage per clock cycle
- Super-pipeline system is capable of performing two pipeline stages per clock cycle
- Superscalar performs only one pipeline stage per clock cycle in each parallel pipeline



Super pipeline benefit & drawback

- Benefits:

The major benefit of super-pipelining is the increase in the number of instructions which can be in the pipeline at one time and hence the level of parallelism.

- Drawbacks:

The larger number of instructions "in flight" (*i.e.,* in some part of the pipeline) at any time, increases the potential for data dependencies to introduce stalls.

Limitations of superscalar

- The superscalar approach depends on the ability to execute multiple instructions in parallel. The term **Instruction-level parallelism** refers to the degree to which the instructions of a program can be executed in parallel.

- A combination of compiler-based optimization and hardware techniques can be used to maximize instruction-level parallelism.

Limitations to parallelism

Fundamental limitations to parallelism

- True data dependency
- Procedural dependency
- Resource conflicts
- Output dependency
- Anti-dependency

Data dependency problem

- The major problem of executing multiple instructions in a scalar program is the handling of data dependencies. If data dependencies are not effectively handled, it is difficult to achieve an execution rate of more than one instruction per clock cycle.

Superscalar Architecture

1) Scalar Architecture
2) Vector Architecture
3) Pipeline Architecture
4) Superscalar Architecture

POWERCERT VIDEOS

HYPER-THREADING

Technology developed by Intel

PowerCert

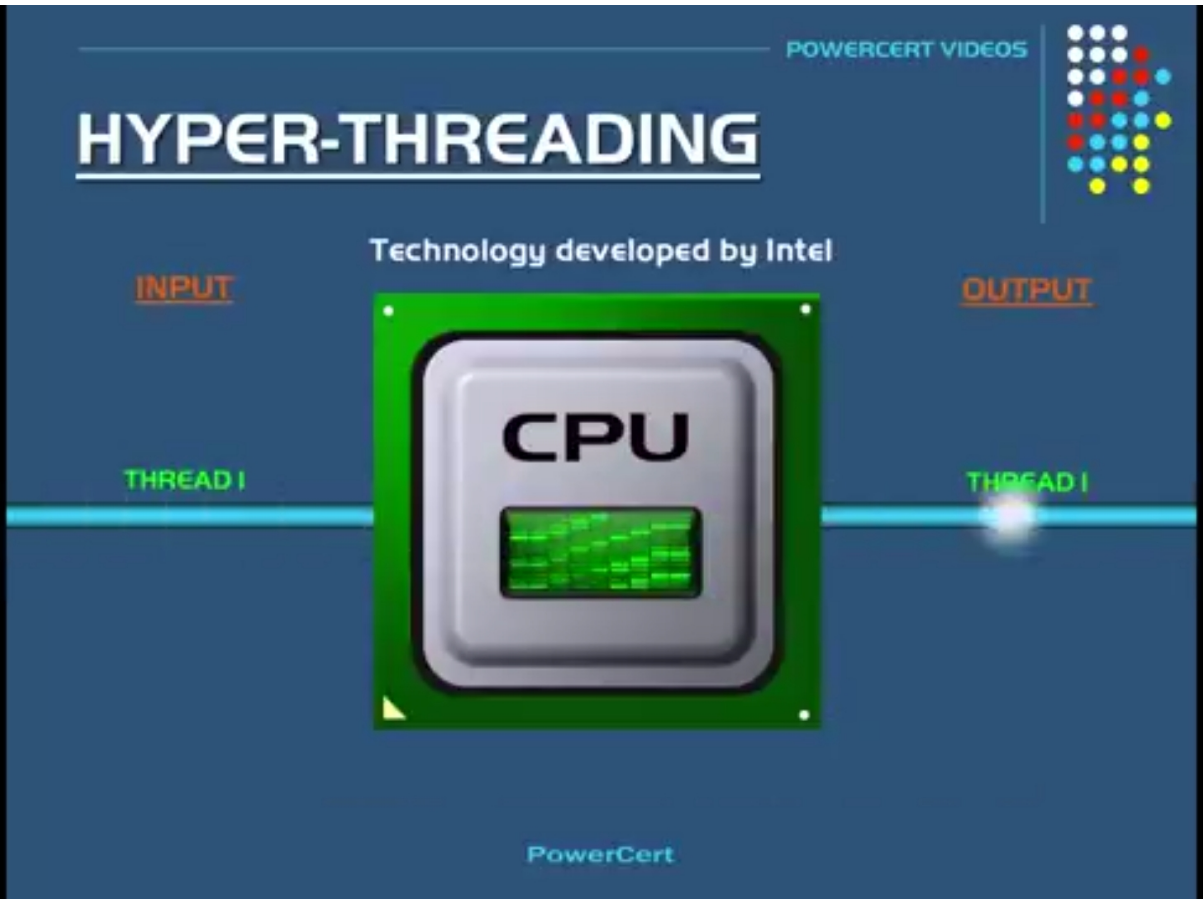
POWERCERT VIDEOS

HYPER-THREADING

Technology developed by Intel

Increases the performance of the CPU cores.

PowerCert


POWERCERT VIDEOS
HYPER-THREADING
Technology developed by Intel
INPUT
OUTPUT
CPU
THREAD I
THREAD I
PowerCert

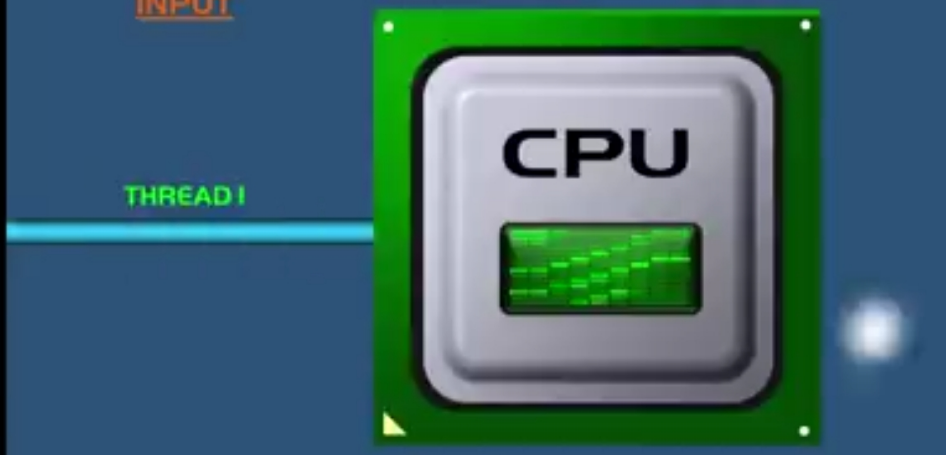
POWERCERT VIDEOS

HYPER-THREADING

Technology developed by Intel

INPUT

OUTPUT



Enables multiple threads to be run by each core to make the CPU run more efficiently.

PowerCert

POWERCERT VIDEOS

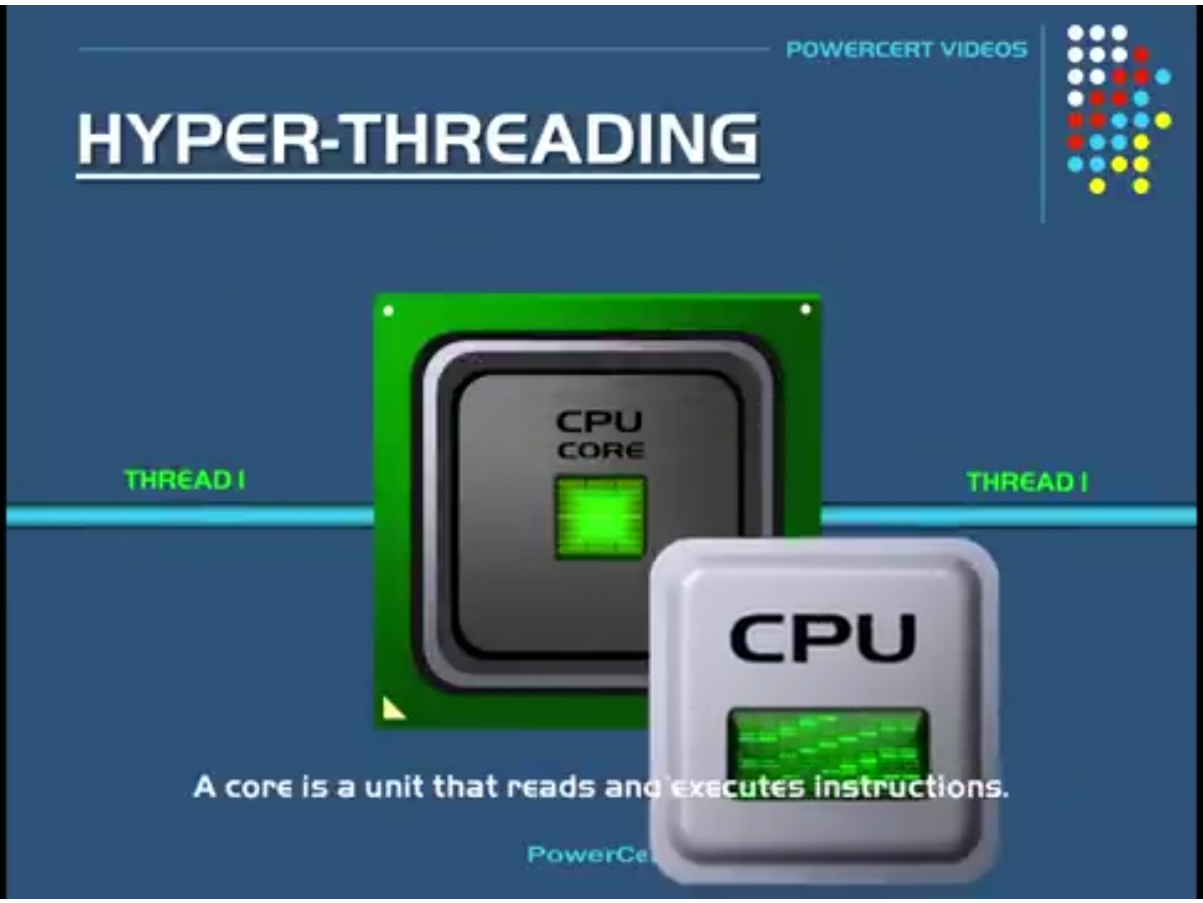
HYPER-THREADING

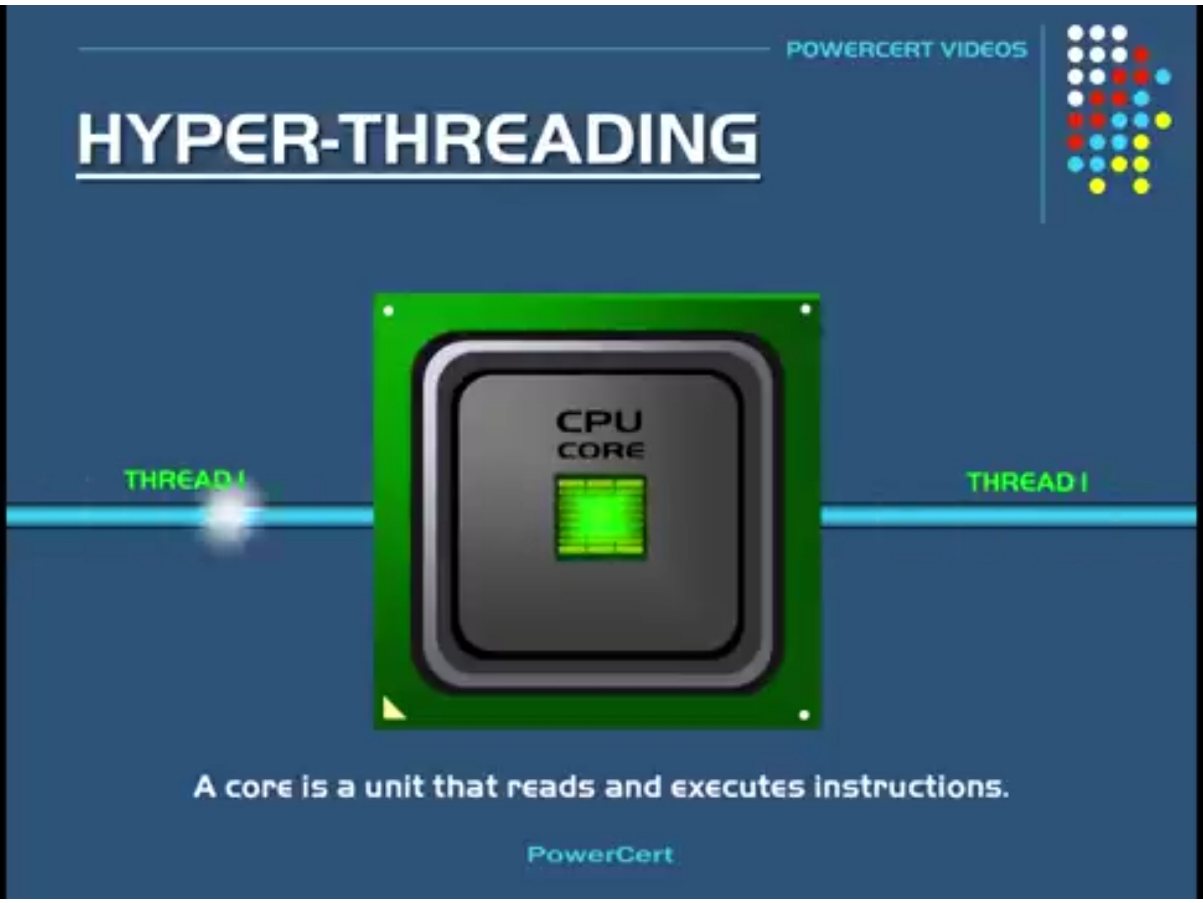
THREAD 1

THREAD 1

A core is a unit that reads and executes instructions.

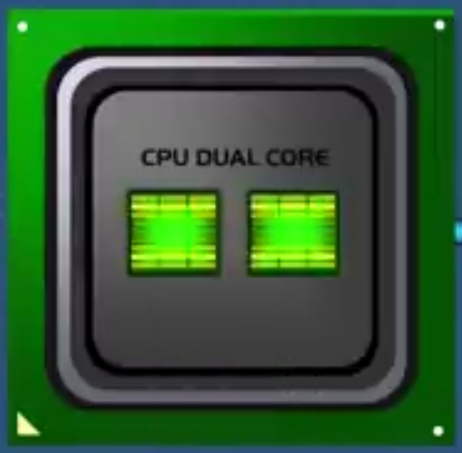
PowerCert


POWERCERT VIDEOS
HYPER-THREADING
CPU
CORE
THREAD I
THREAD I
CPU
A core is a unit that reads and executes instructions.


POWERCERT VIDEOS
HYPER-THREADING
CPU
CORE
THREAD 1
THREAD 1
A core is a unit that reads and executes instructions.
PowerCert

POWERCERT VIDEOS

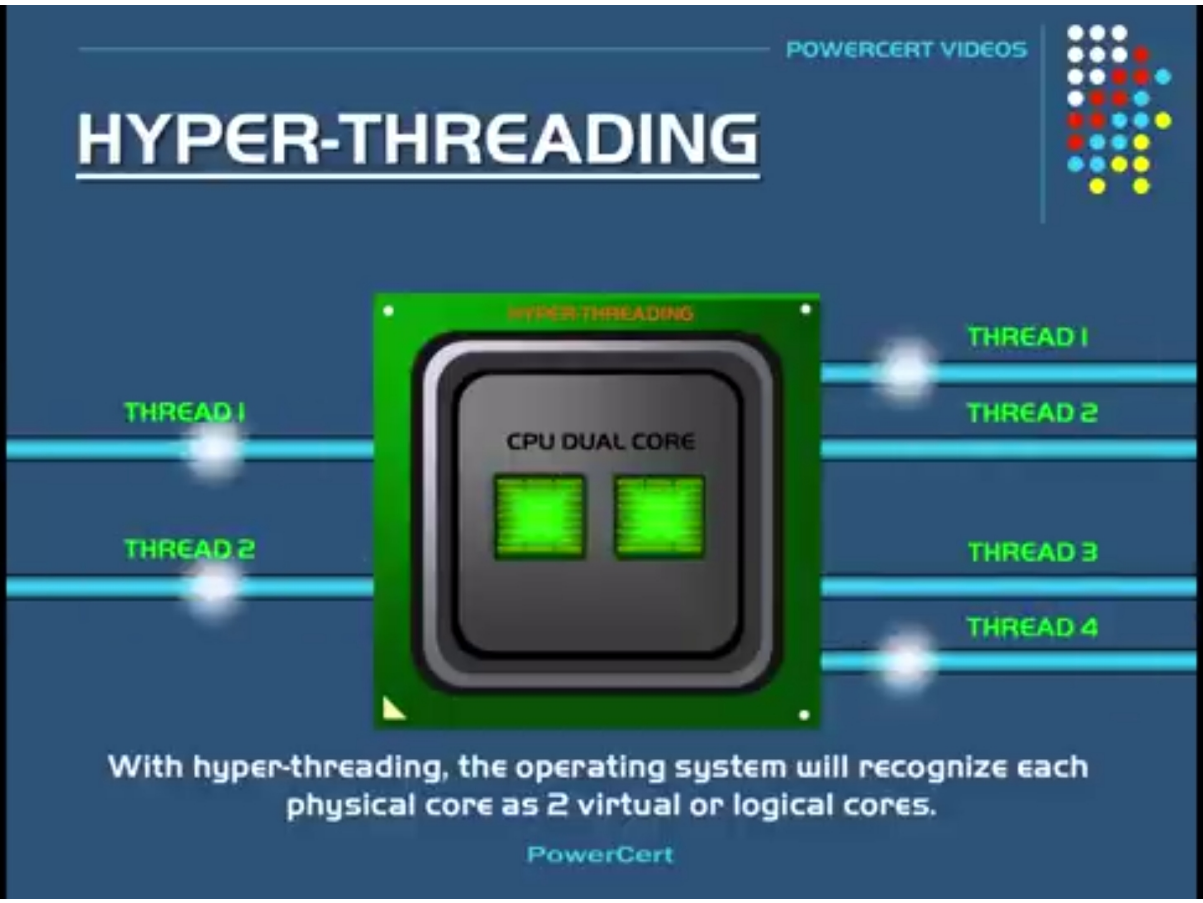
HYPER-THREADING



THREAD I

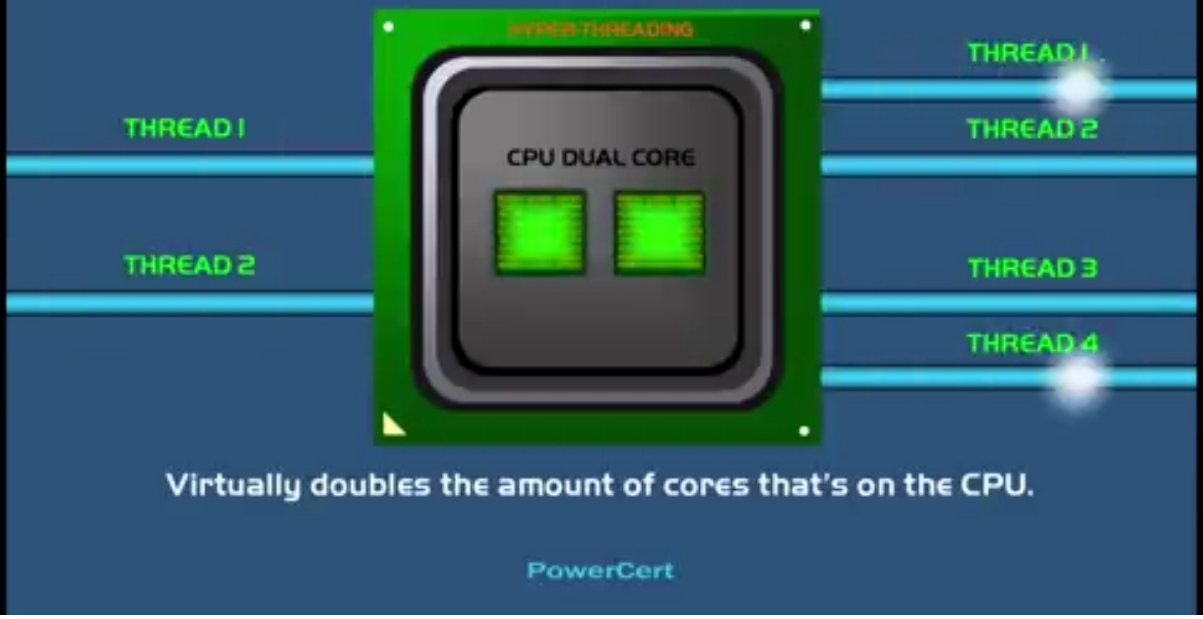
THREAD I

PowerCert


POWERCERT VIDEOS
HYPER-THREADING
HYPER THREADING
CPU DUAL CORE
THREAD 1
THREAD 2
THREAD 1
THREAD 2
THREAD 3
THREAD 4
With hyper-threading, the operating system will recognize each physical core as 2 virtual or logical cores.
PowerCert

POWERCERT VIDEOS

HYPER-THREADING

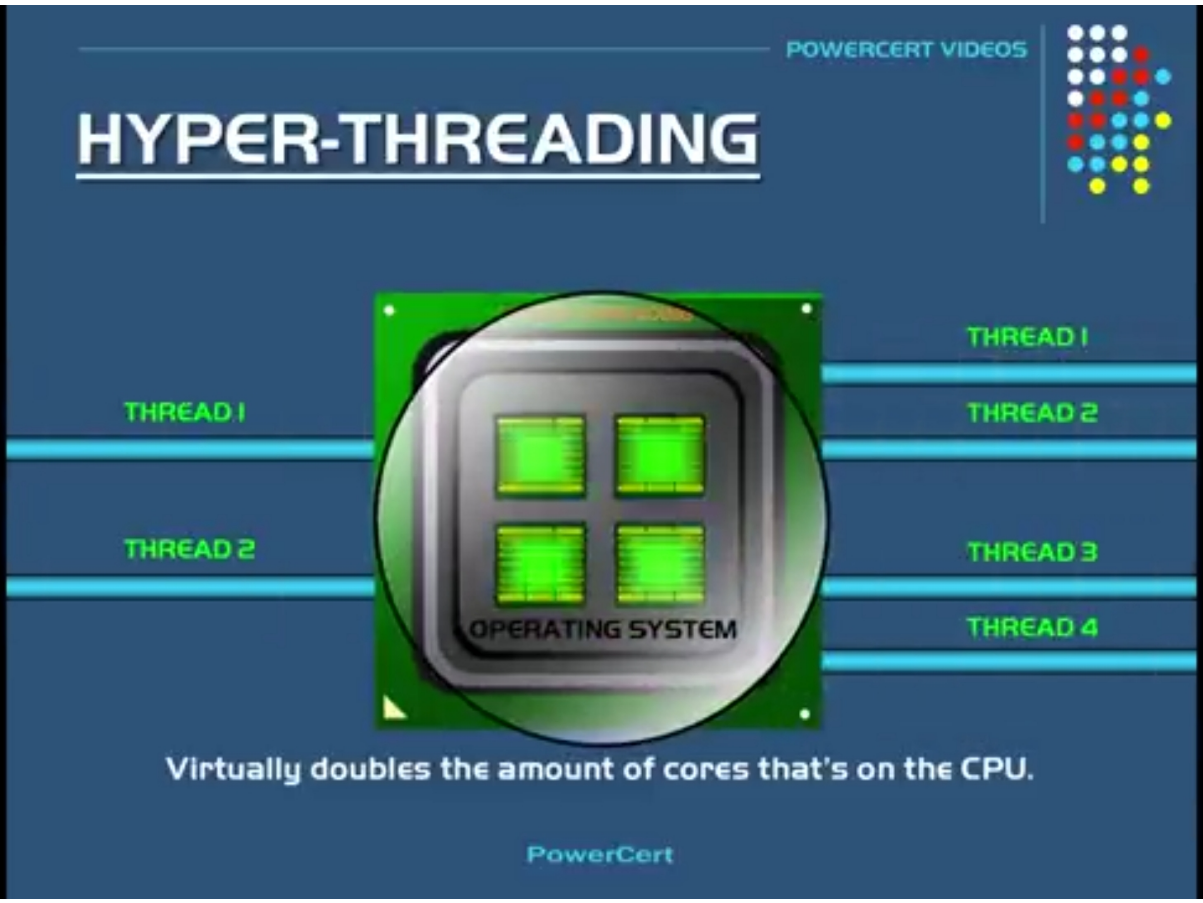


Virtually doubles the amount of cores that's on the CPU.

PowerCert

POWERCERT VIDEOS

HYPER-THREADING

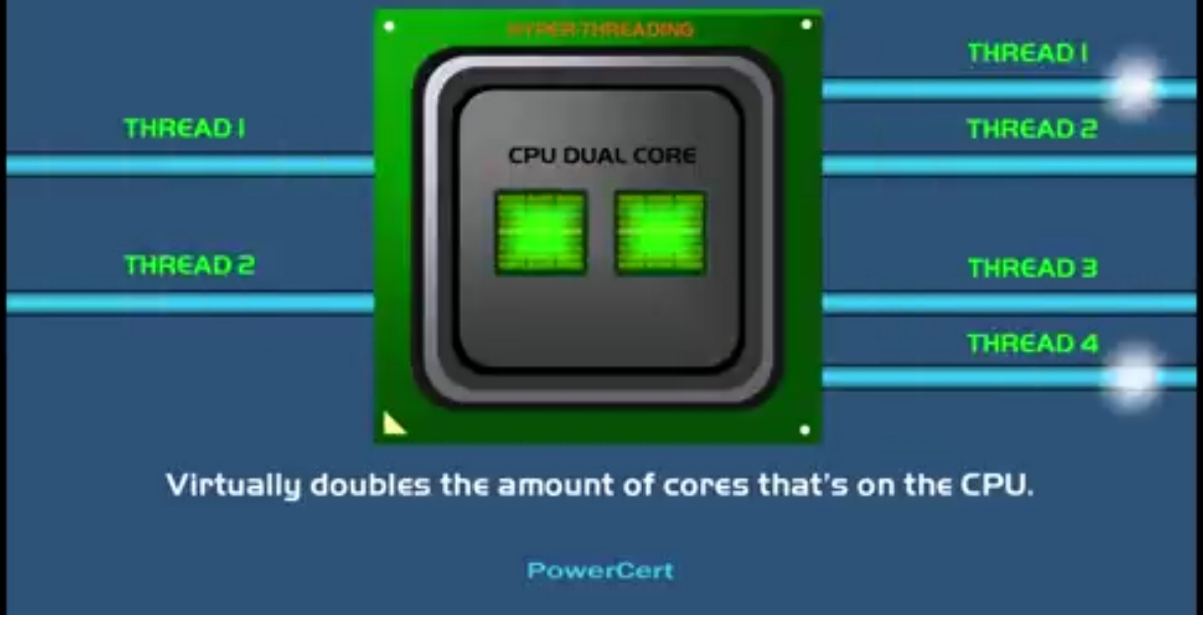


Virtually doubles the amount of cores that's on the CPU.

PowerCert

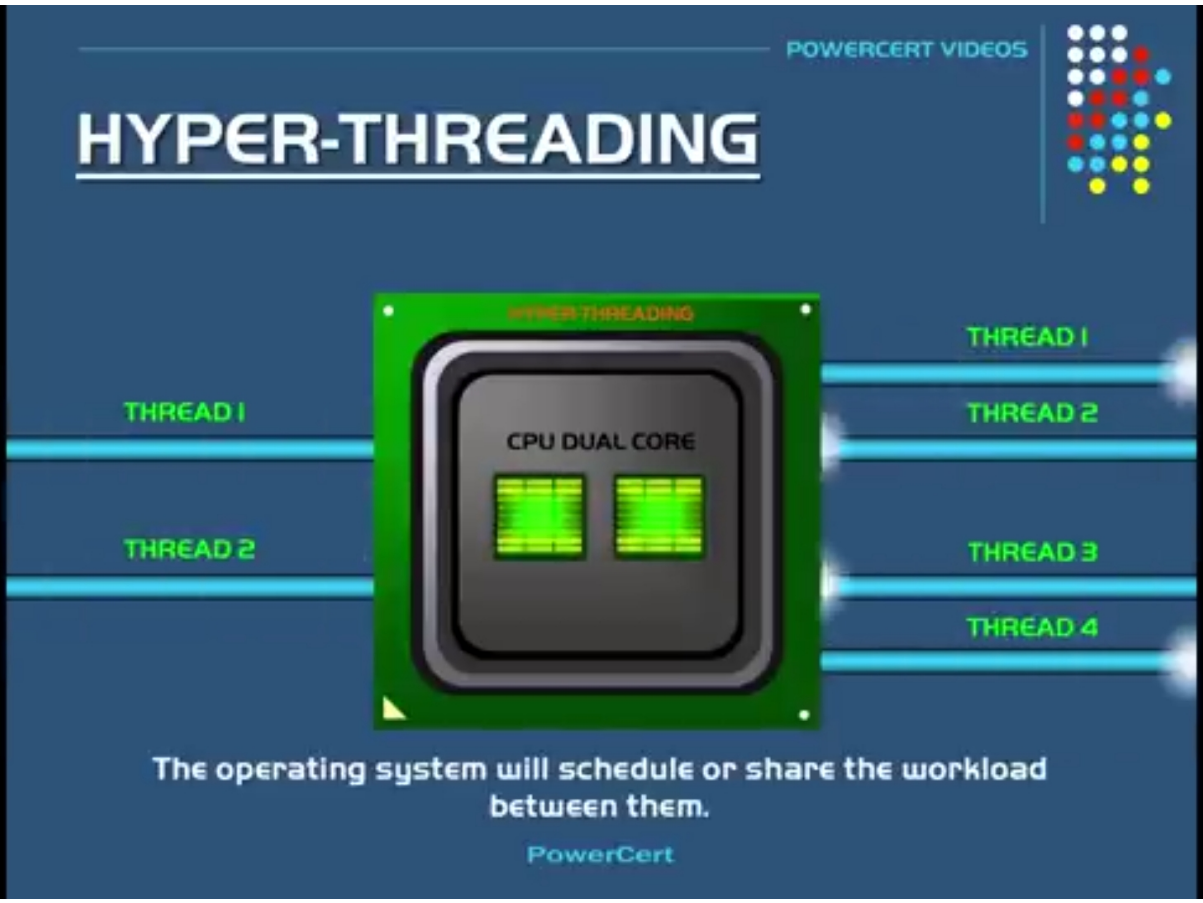
POWERCERT VIDEOS

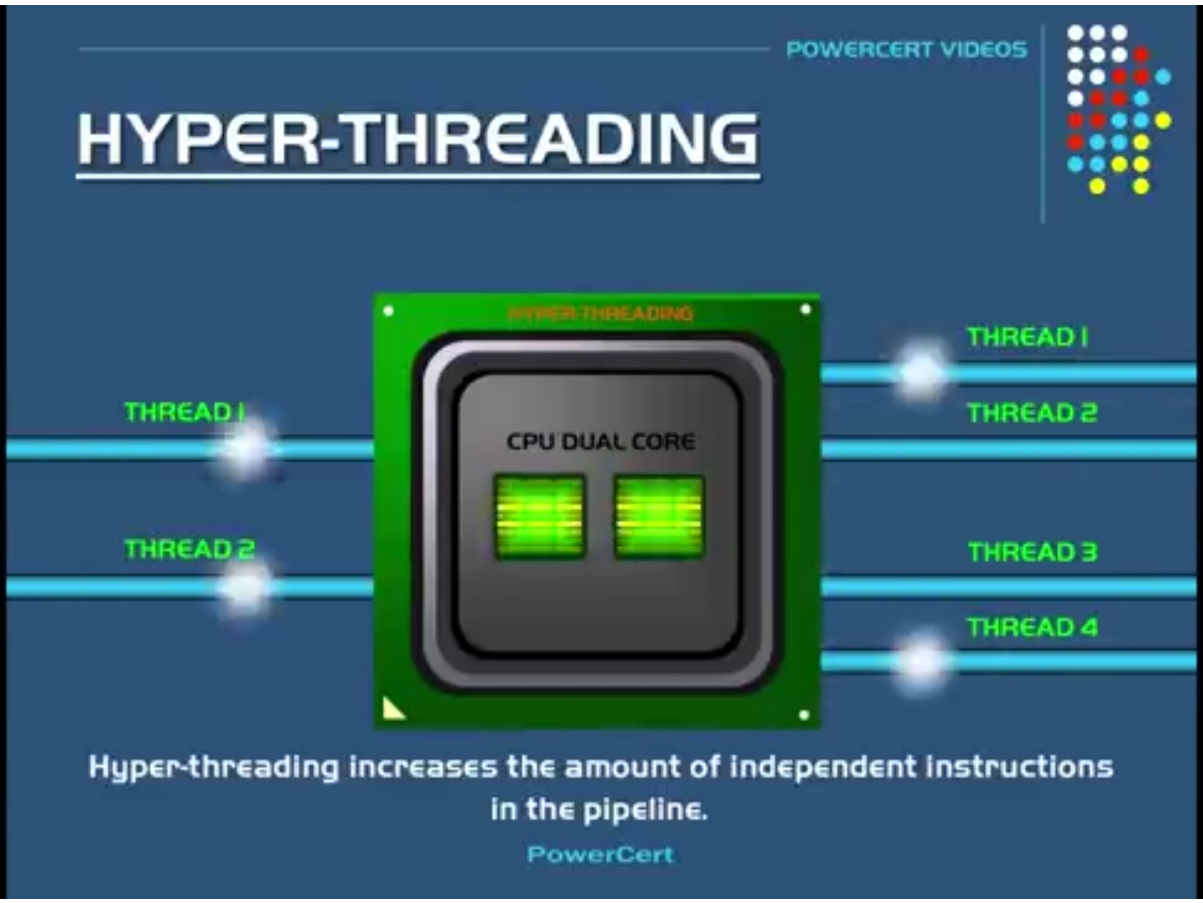
HYPER-THREADING

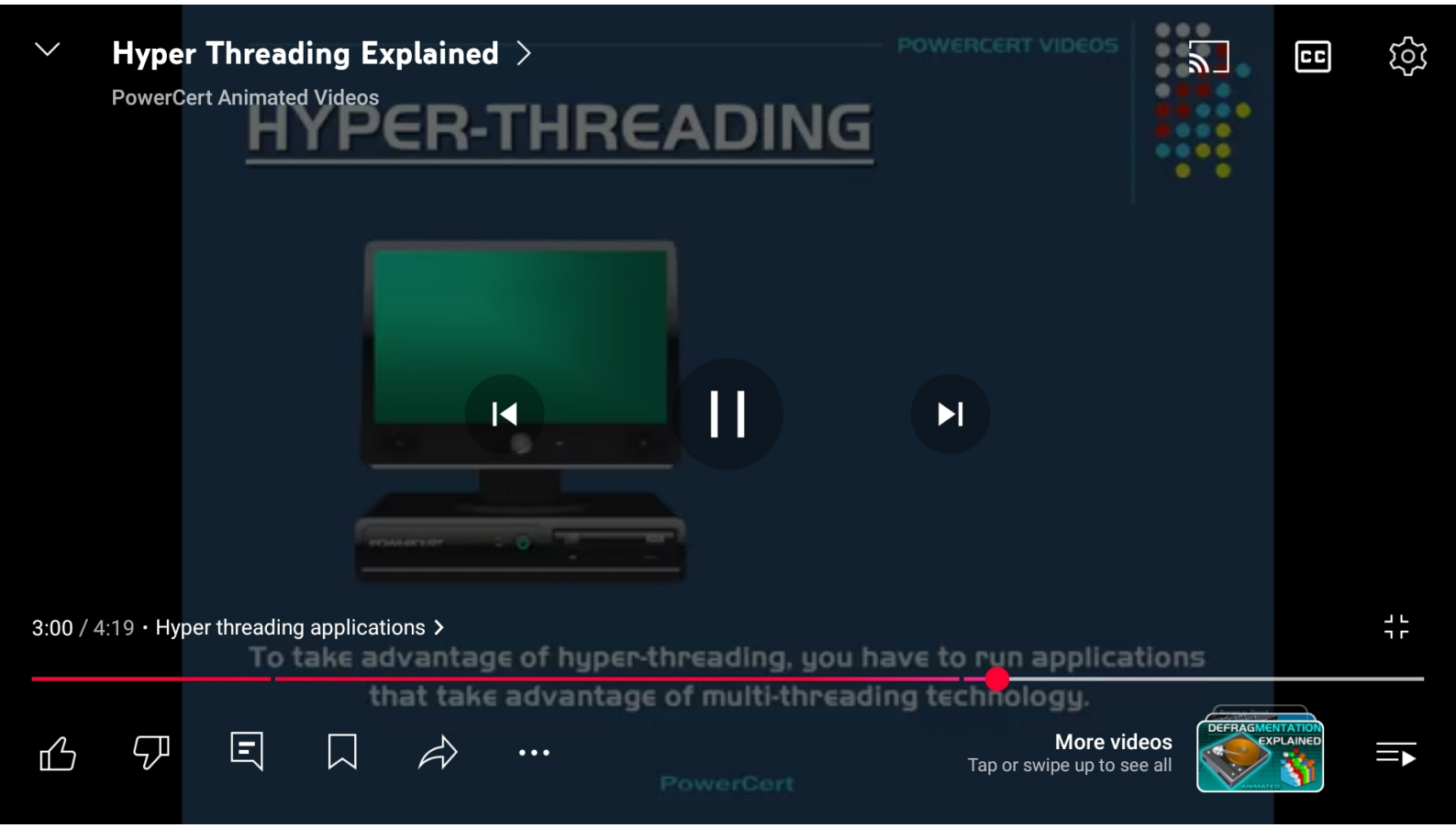


Virtually doubles the amount of cores that's on the CPU.

PowerCert


POWERCERT VIDEOS
HYPER-THREADING
HYPER THREADING
CPU DUAL CORE
THREAD 1
THREAD 2
THREAD 1
THREAD 2
THREAD 3
THREAD 4
The operating system will schedule or share the workload between them.
PowerCert


POWERCERT VIDEOS
HYPER-THREADING
HYPER-THREADING
CPU DUAL CORE
THREAD 1
THREAD 2
THREAD 1
THREAD 2
THREAD 3
THREAD 4
Hyper-threading increases the amount of independent instructions in the pipeline.
PowerCert


Hyper Threading Explained
PowerCert Animated Videos
POWERCERT VIDEOS
HYPER-THREADING
3:00 / 4:19 • Hyper threading applications
To take advantage of hyper-threading, you have to run applications that take advantage of multi-threading technology.
PowerCert
More videos
Tap or swipe up to see all
DEFRAGMENTATION EXPLAINED

POWERCERT VIDEOS

HYPER-THREADING

Example: Video editing & encoding.

To take advantage of hyper-threading, you have to run applications that take advantage of multi-threading technology.

PowerCert

POWERCERT VIDEOS

HYPER-THREADING

Example: Video editing & encoding.

To take advantage of hyper-threading, you have to run applications that take advantage of multi-threading technology.

PowerCert

POWERCERT VIDEOS

HYPER-THREADING

First introduced by Intel in 2002.

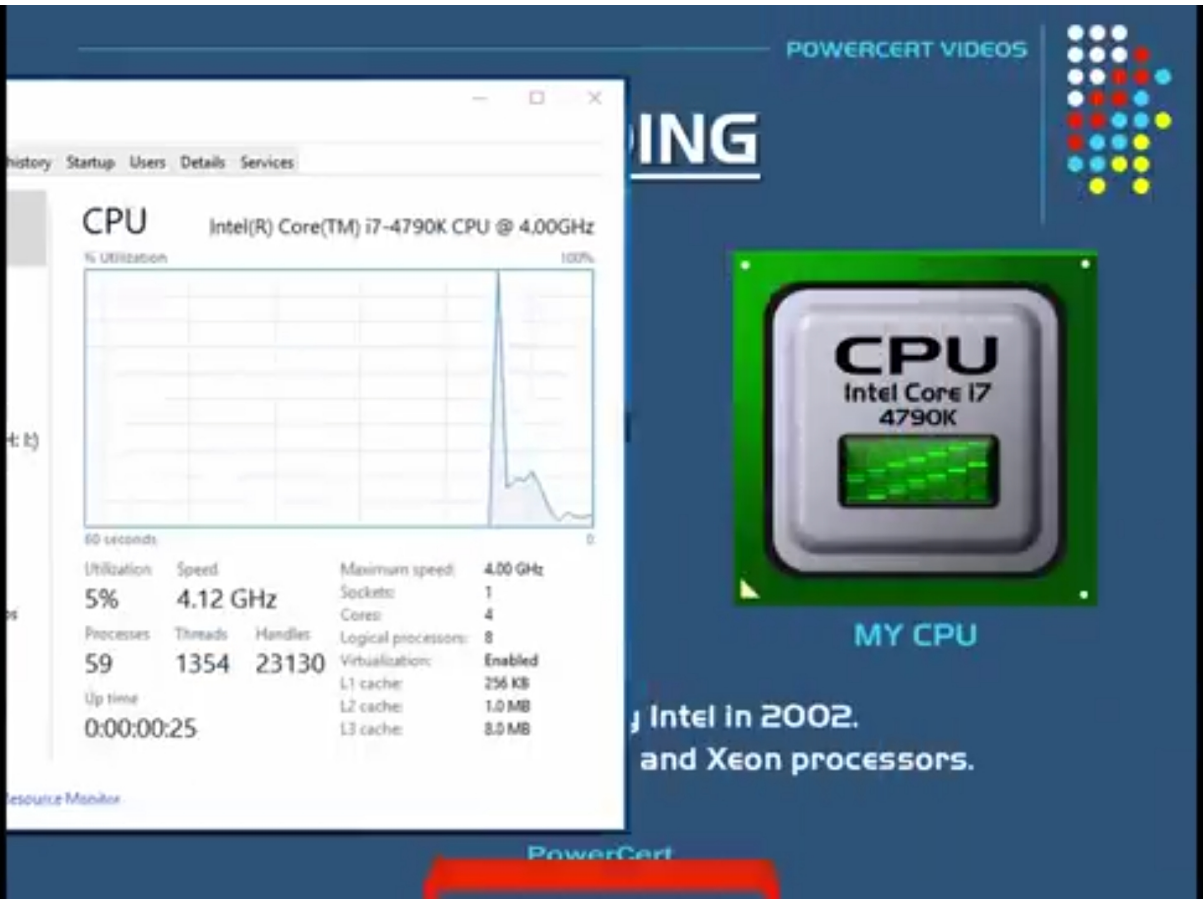
PowerCert

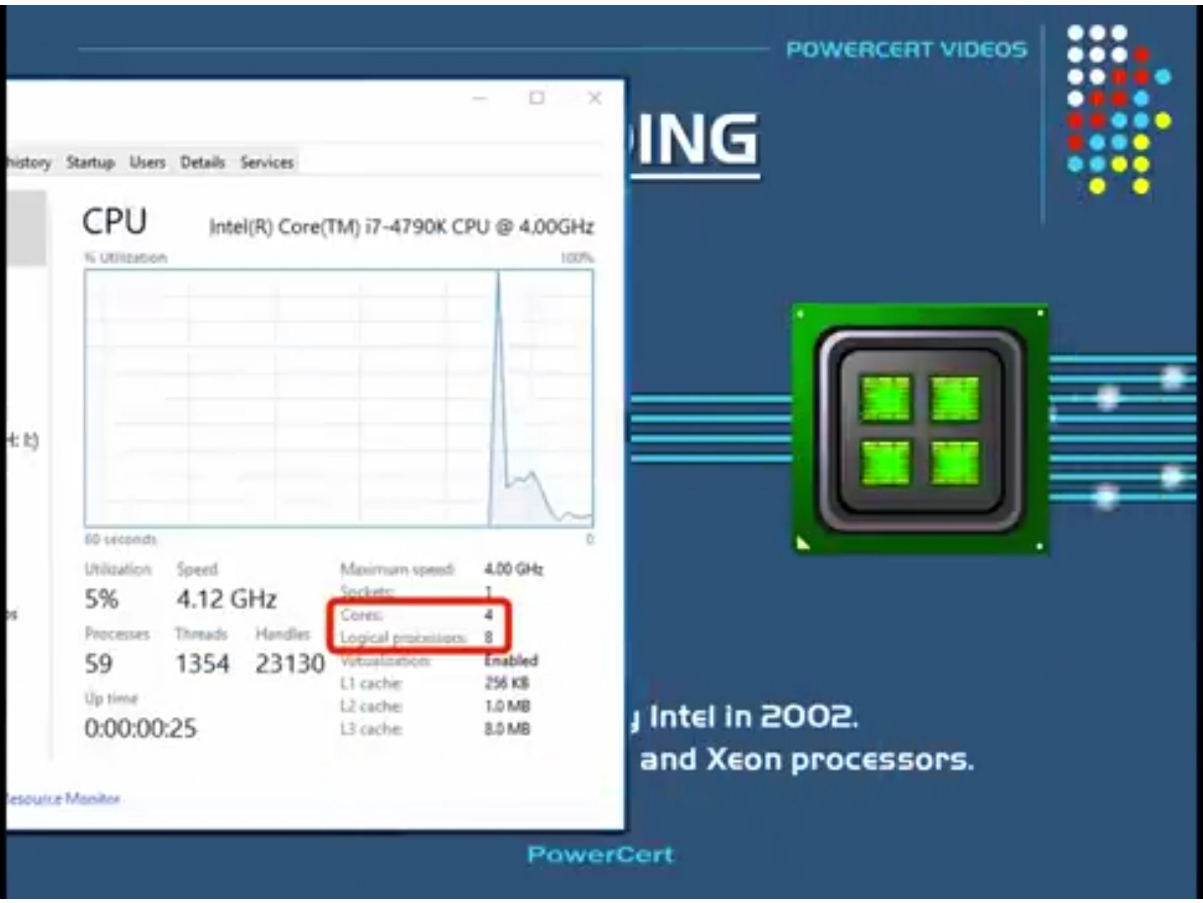
POWERCERT VIDEOS

HYPER-THREADING

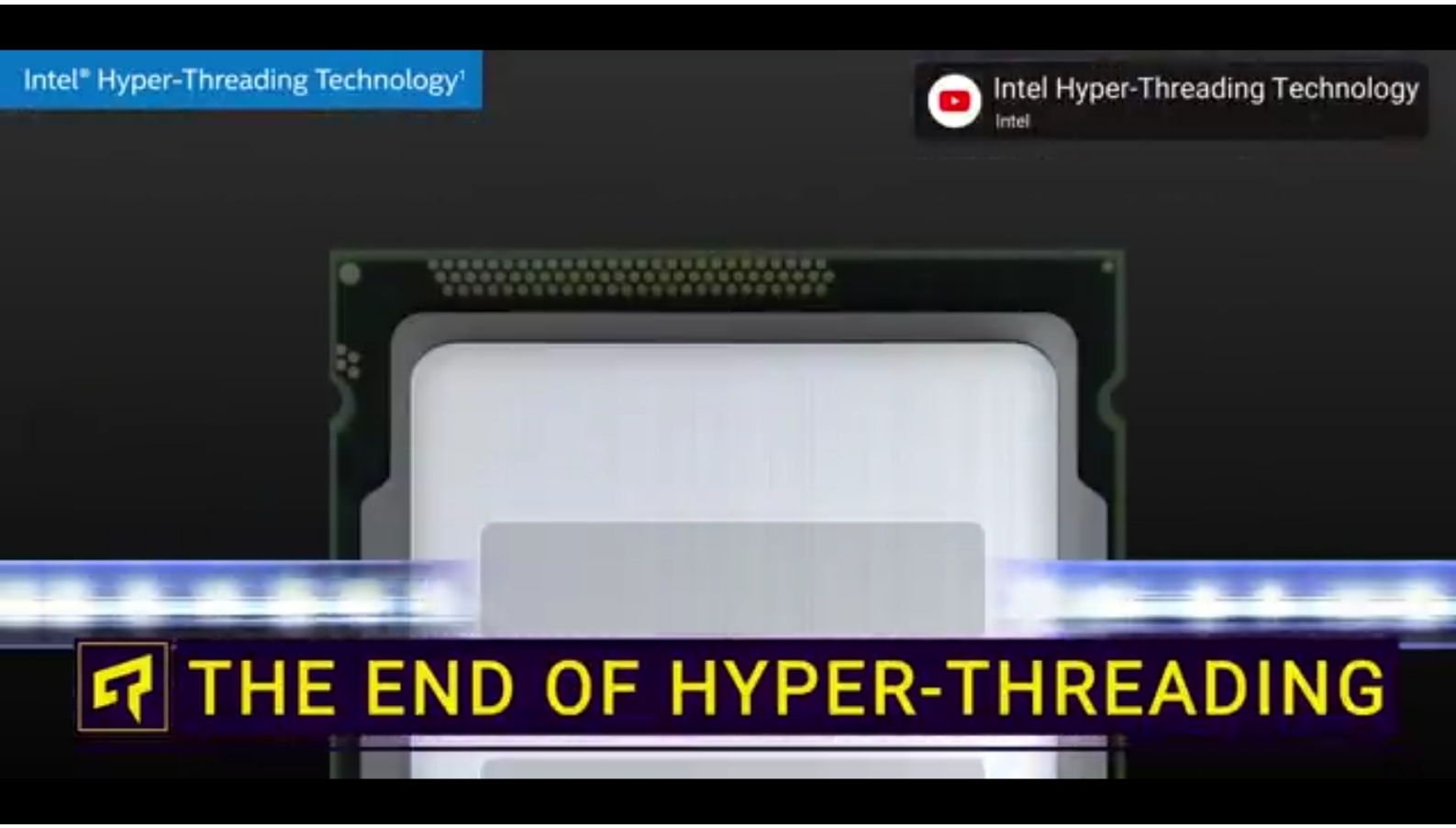
First introduced by Intel in 2002.
Debuted on the Pentium 4 and Xeon processors.

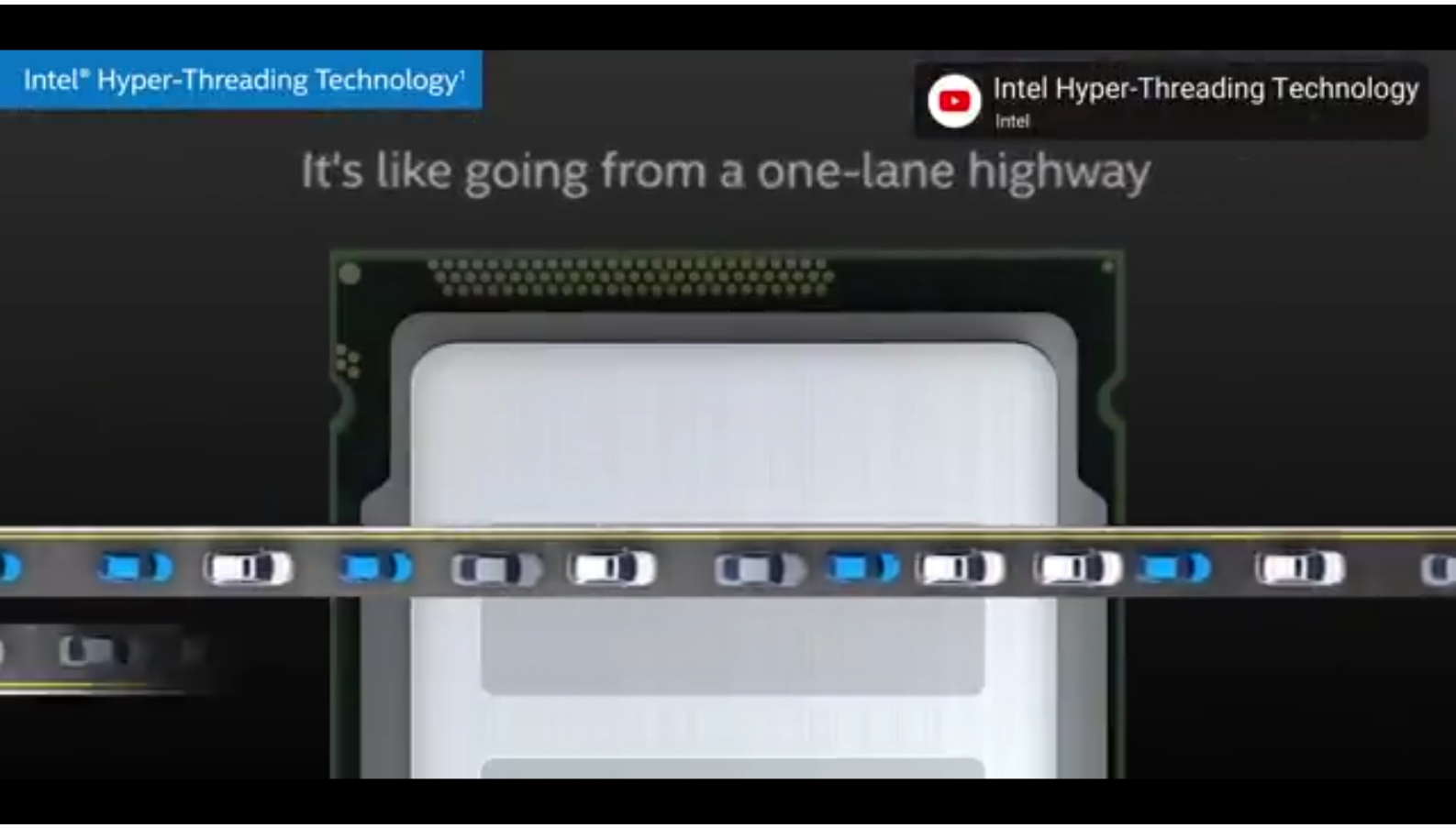
PowerCert

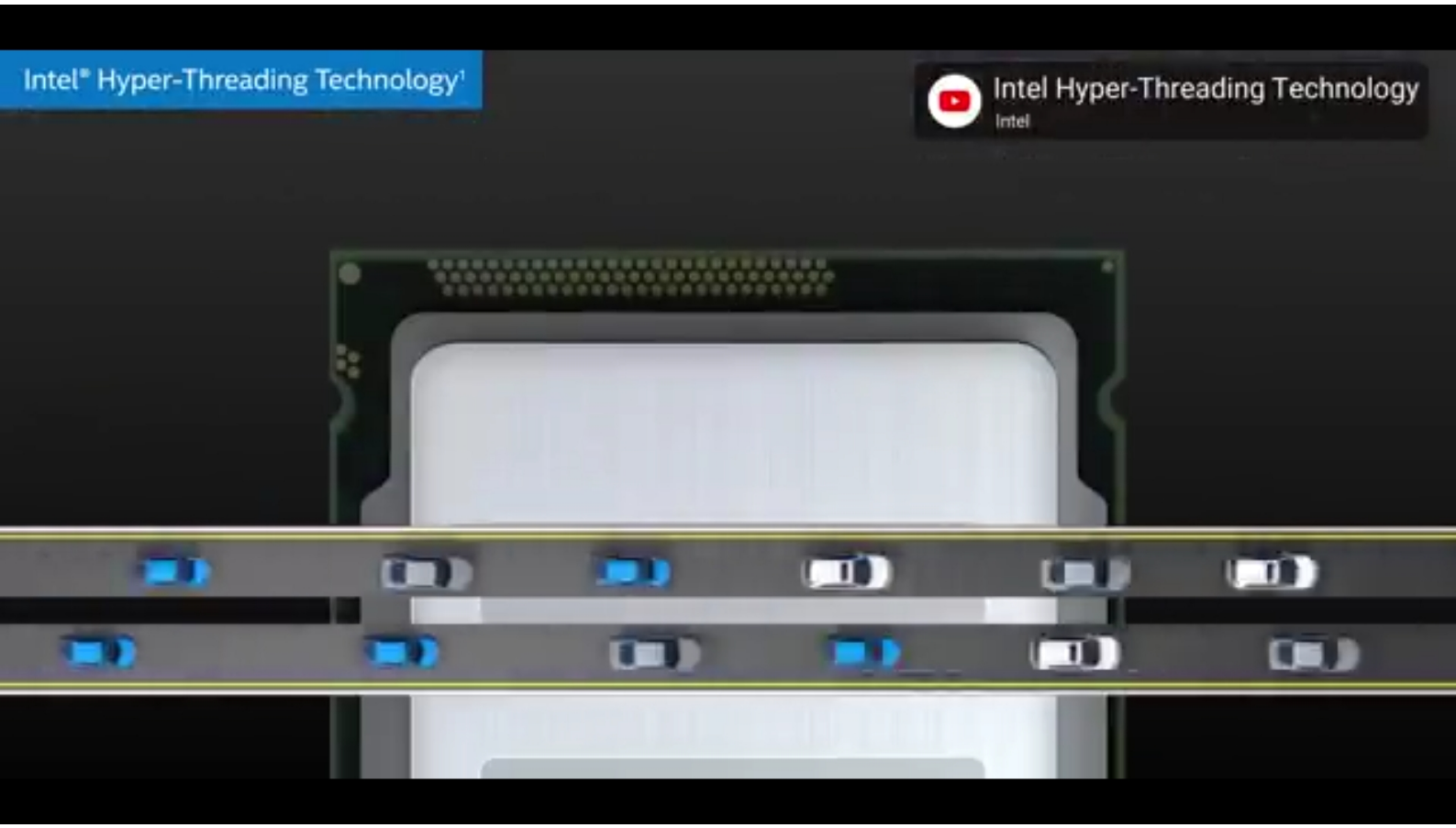

POWERCERT VIDEOS
ING
History Startup Users Details Services
CPU
Intel(R) Core(TM) i7-4790K CPU @ 4.00GHz
% Utilization
100%
60 seconds
0
Utilization
5%
Speed
4.12 GHz
Processes
59
Threads
1354
Handles
23130
Up time
0:00:00:25
Maximum speed: 4.00 GHz
Sockets: 1
Cores: 4
Logical processors: 8
Virtualization: Enabled
L1 cache: 256 KB
L2 cache: 1.0 MB
L3 cache: 8.0 MB
Resource Monitor
CPU
Intel Core i7
4790K
MY CPU
Intel in 2002.
and Xeon processors.
PowerCert

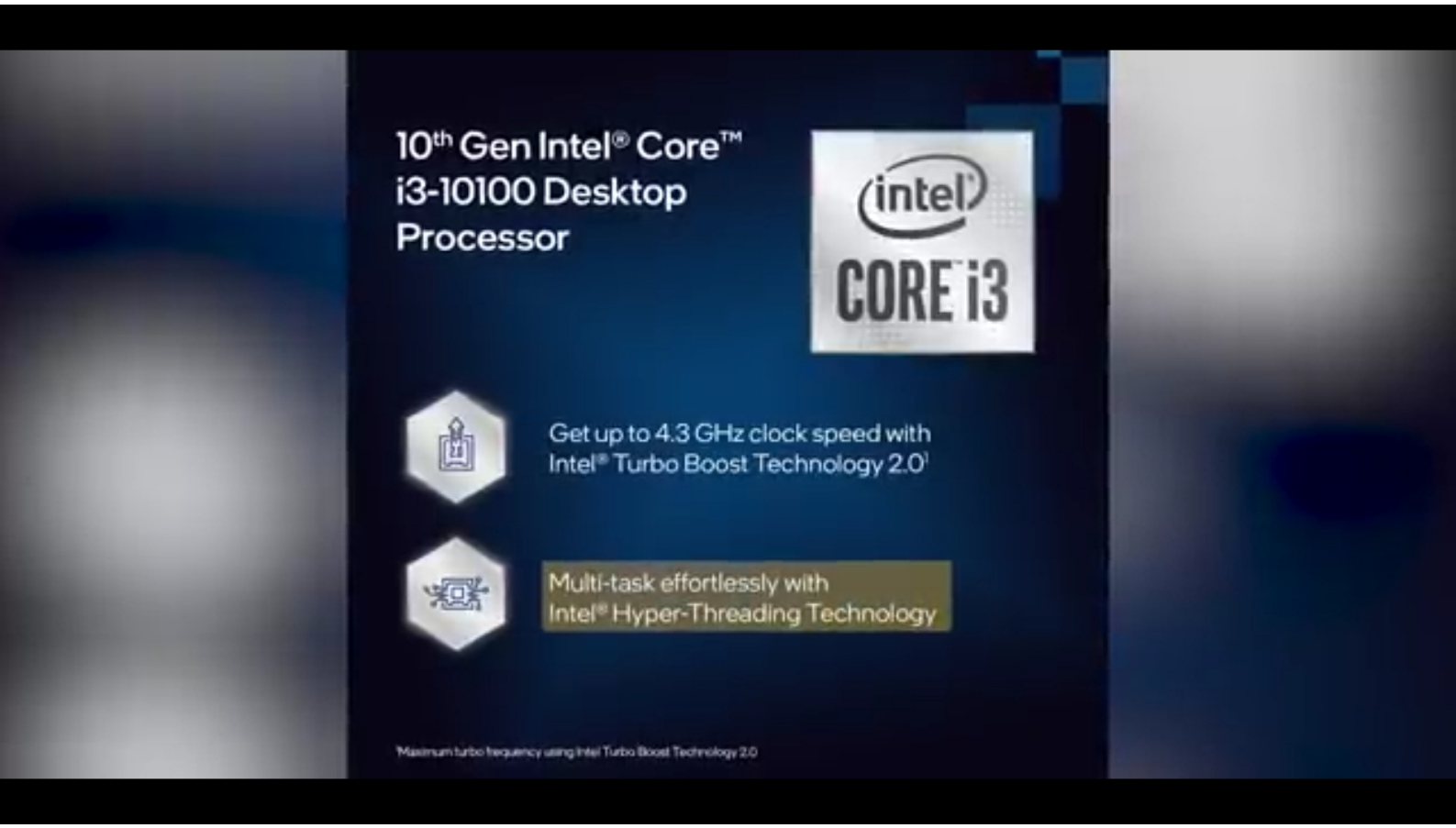

POWERCERT VIDEOS
ING
Startup
Users
Details
Services
CPU
Intel(R) Core(TM) i7-4790K CPU @ 4.00GHz
% Utilization
100%
60 seconds
0
Utilization
5%
Speed
4.12 GHz
Processes
59
Threads
1354
Handles
23130
Up time
0:00:00:25
Maximum speed: 4.00 GHz
Cores: 4
Logical processors: 8
Virtualization: Enabled
L1 cache: 256 KB
L2 cache: 1.0 MB
L3 cache: 8.0 MB
Intel in 2002.
and Xeon processors.
PowerCert


Intel® Hyper-Threading Technology¹
Intel Hyper-Threading Technology
Intel
Increases the performance of your processor cores.
THE END OF HYPER-THREADING


Intel® Hyper-Threading Technology¹
Intel Hyper-Threading Technology
Intel
THE END OF HYPER-THREADING


Intel® Hyper-Threading Technology¹
Intel Hyper-Threading Technology
Intel
It's like going from a one-lane highway


Intel® Hyper-Threading Technology¹
Intel Hyper-Threading Technology
Intel


10th Gen Intel® Core™
i3-10100 Desktop
Processor
intel
CORE i3
Get up to 4.3 GHz clock speed with
Intel® Turbo Boost Technology 2.0[1]
Multi-task effortlessly with
Intel® Hyper-Threading Technology
[1]Maximum turbo frequency using Intel Turbo Boost Technology 2.0

https://www.extremetech.com/computing/intels-arrow-lake-cpus-will-allegedly-ditch-hyper-threading-leak

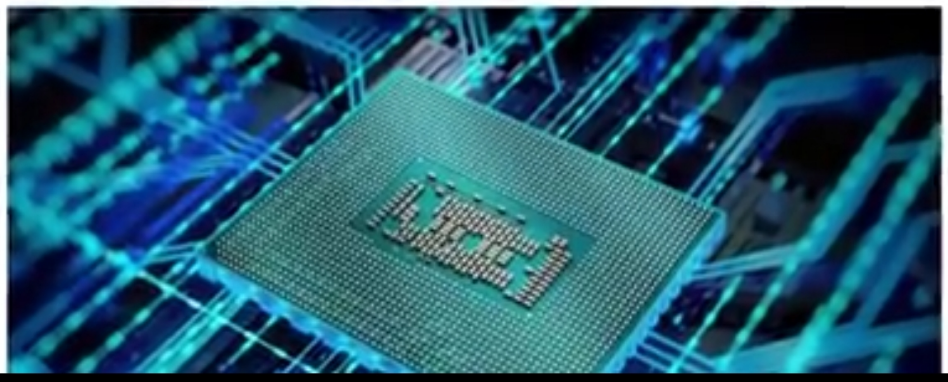
EXTREMETECH Aerospace Cars Computing Defense Electronics Energy Gaming Phones Science Security

Home > Computing

Intel's Arrow Lake CPUs Will Allegedly Ditch Hyper-Threading: Leak

The company's 15th Generation CPUs will be a whole different kettle of fish.

By Josh Norem January 22, 2024



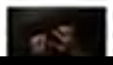
ExtremeTech Newsletter

Subscribe Today to get the latest ExtremeTech news delivered right to your inbox.

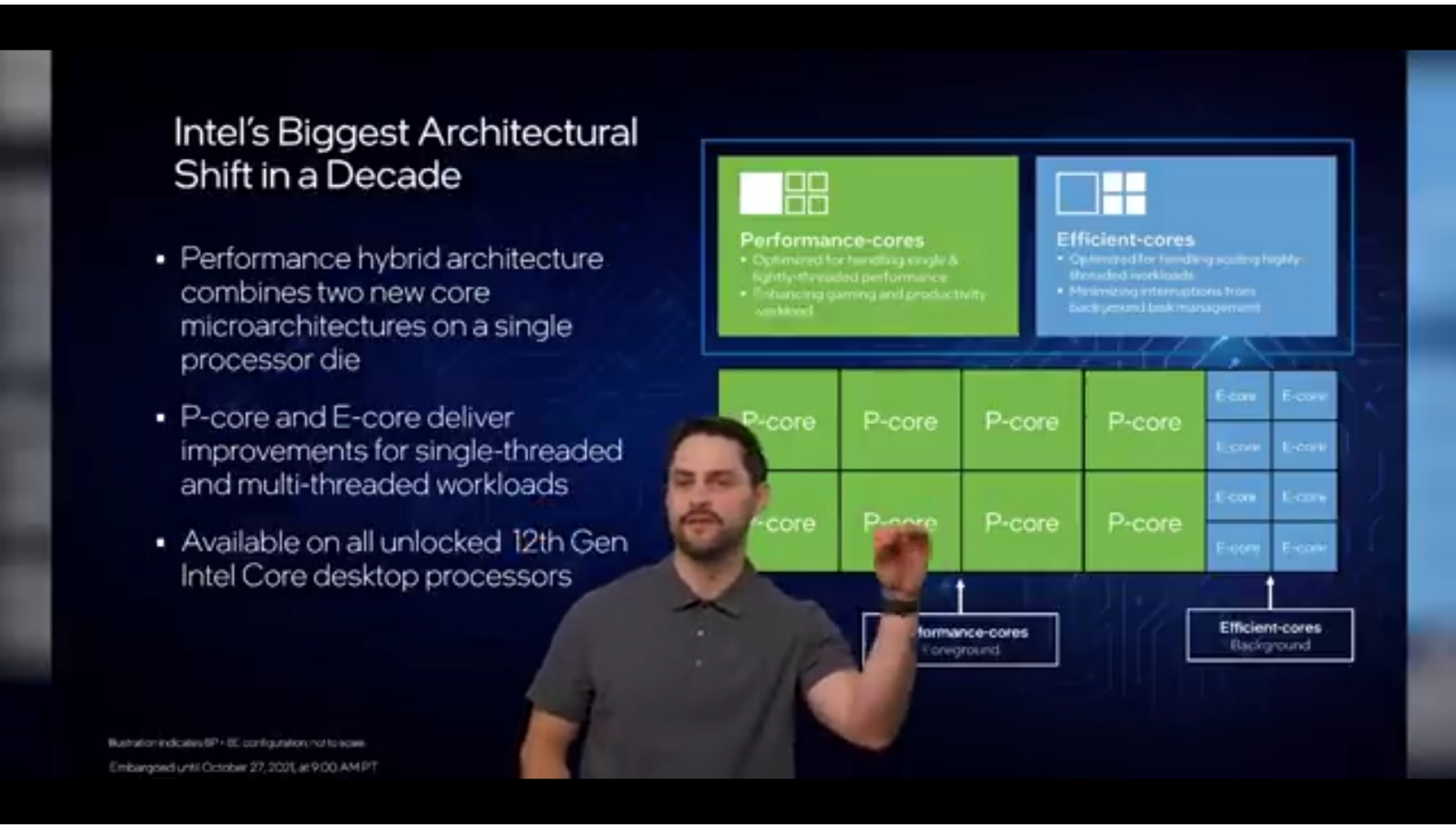
Email Address... Sign Up

This newsletter may contain advertising, deals, or affiliate links. Subscribing to a newsletter indicates your consent to our Terms of use and Privacy Policy. You may unsubscribe from the newsletter at any time.

More Articles



Former Police Officer Using AI to Generate Fake News Stories, Says Dispatch Among Appearances

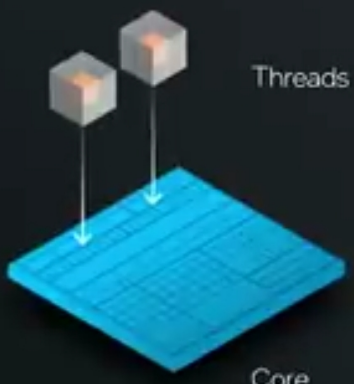

Intel's Biggest Architectural Shift in a Decade
• Performance hybrid architecture combines two new core microarchitectures on a single processor die
• P-core and E-core deliver improvements for single-threaded and multi-threaded workloads
• Available on all unlocked 12th Gen Intel Core desktop processors
Performance-cores
Efficient-cores
P-core
E-core
Efficient-cores
Background
Illustration indicates 8P + 8E configuration; not to scale
Embargoed until October 27, 2021, at 9:00 AM PT

Hyper-Threading Benefits

Improving IPC within the same core area footprint

Hyper-Threading

on latest P-cores



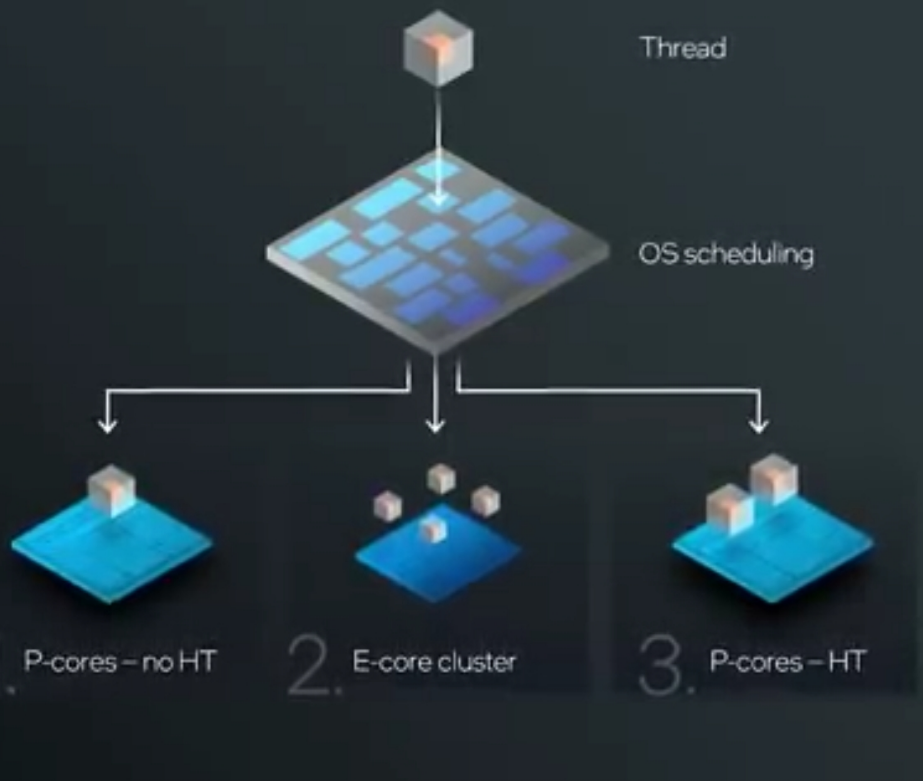
+30%
IPC
(or throughput)

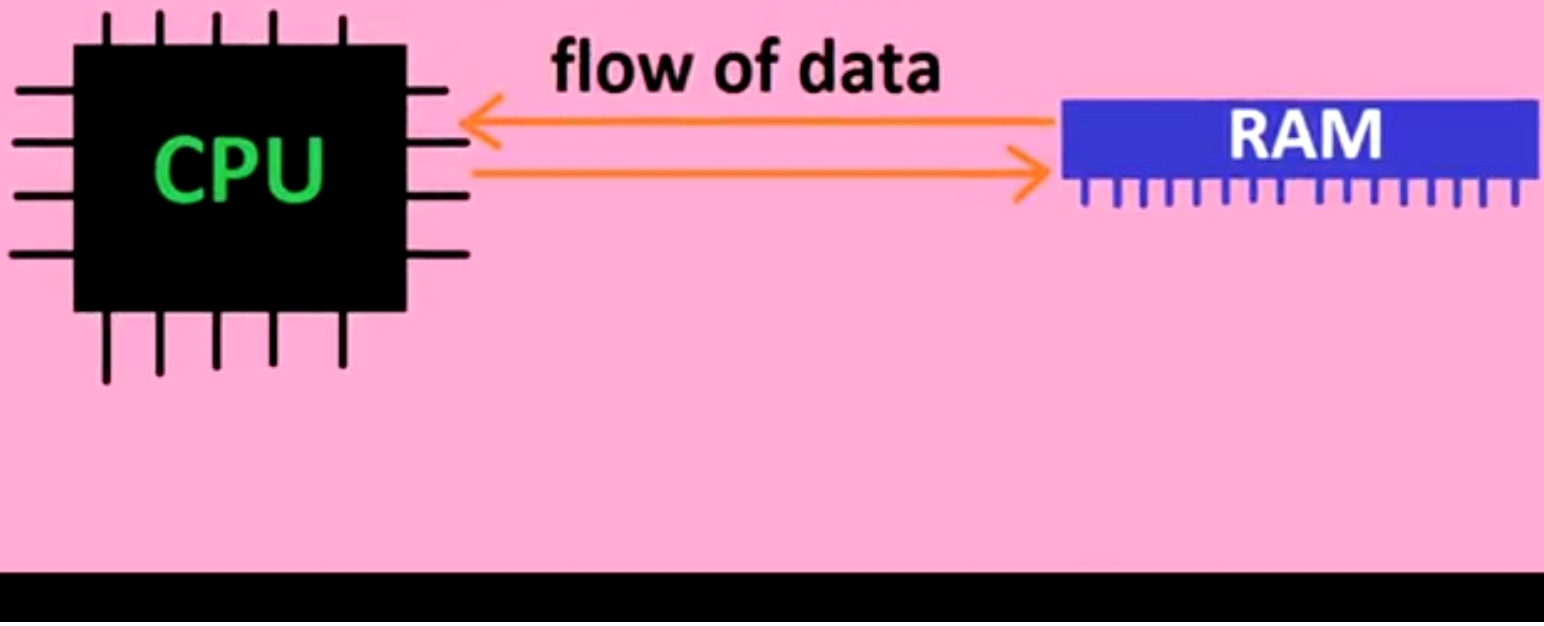
+20%
Cdyn
(power at same V/F)

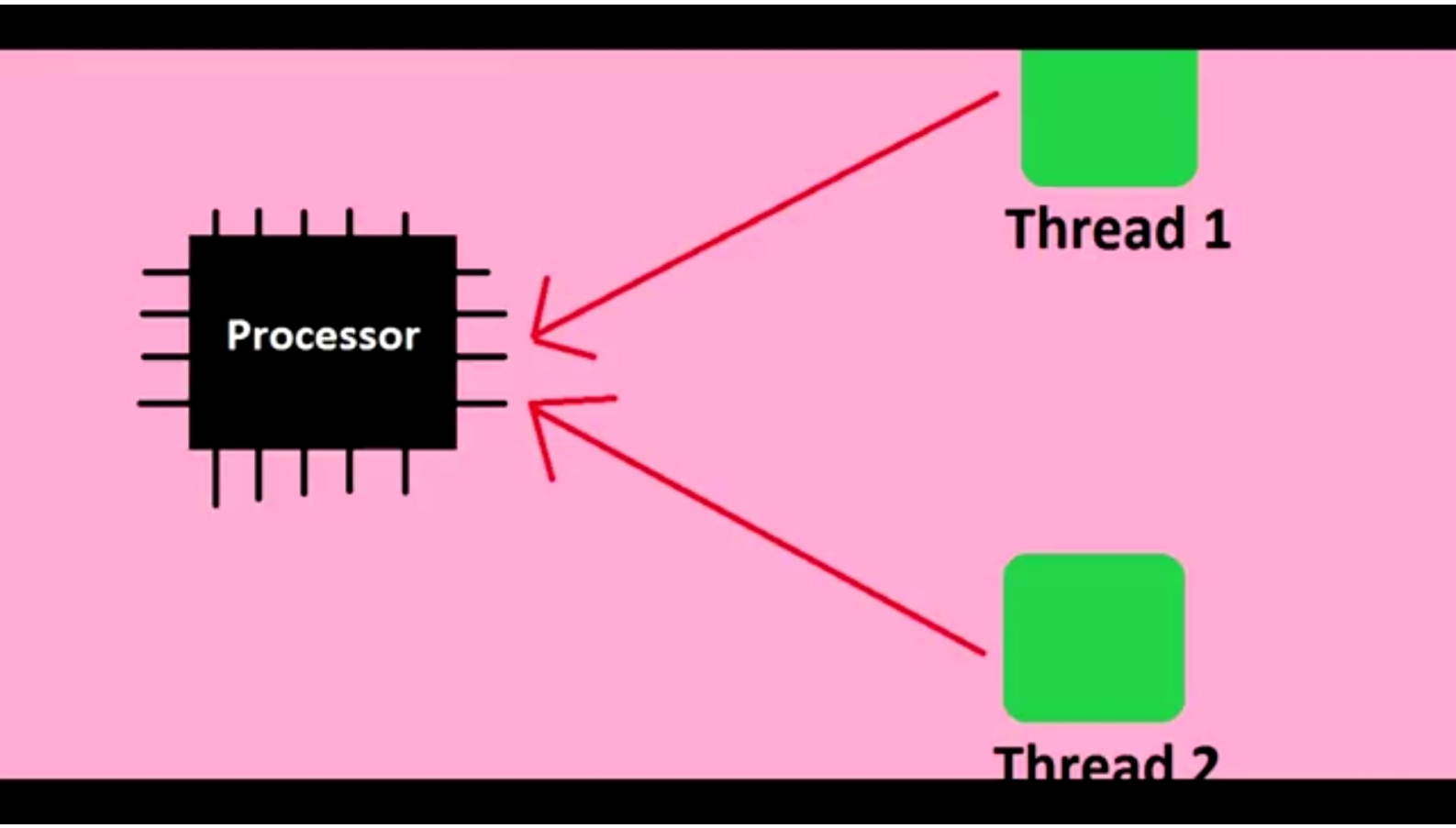
*Projected architecture representation of best-case benefits of Hyper-Threading feature on vs. off on a latest generation P-core

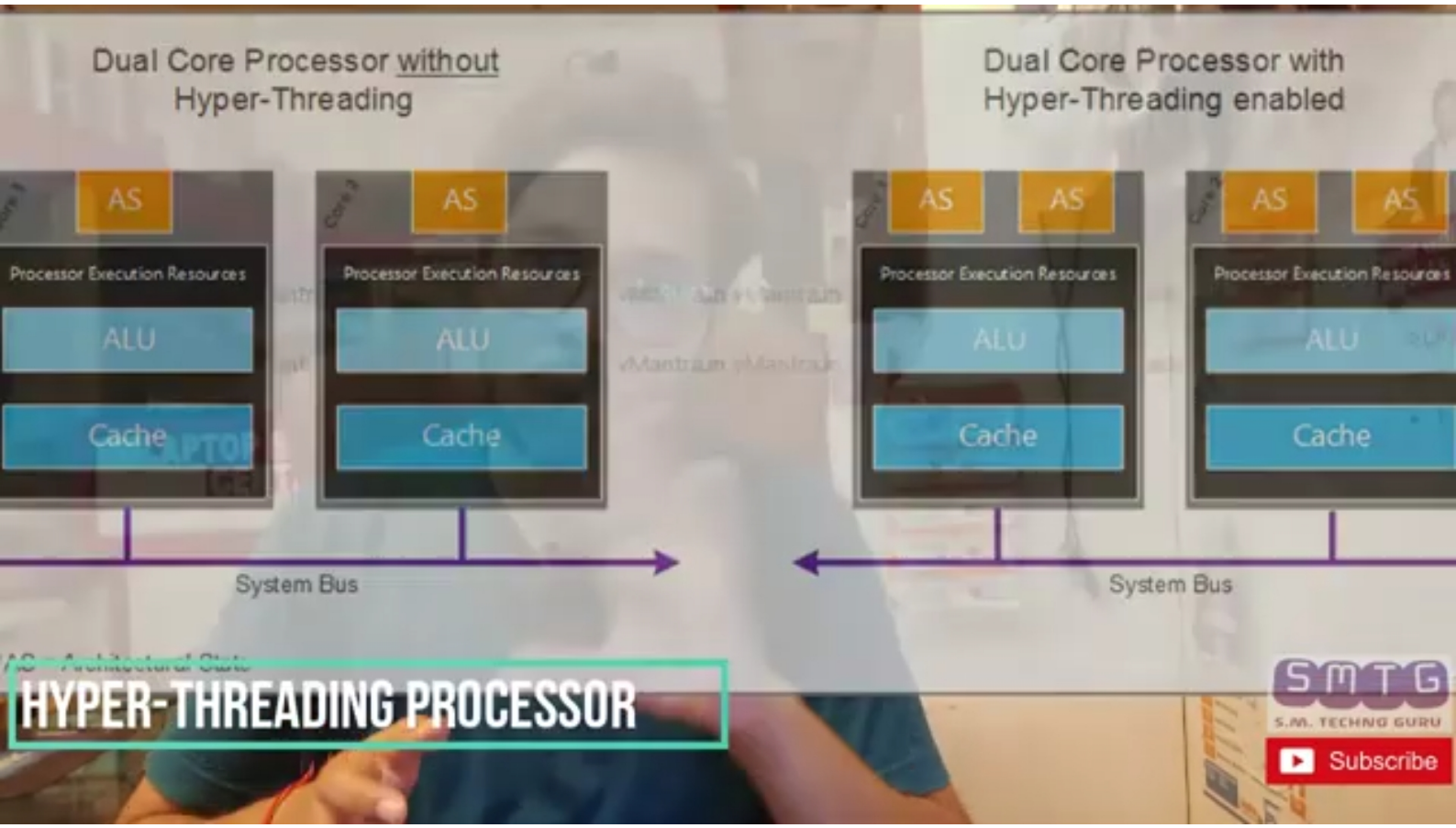
intel Tech Tour '24

Typical Scheduling on Hybrid Client

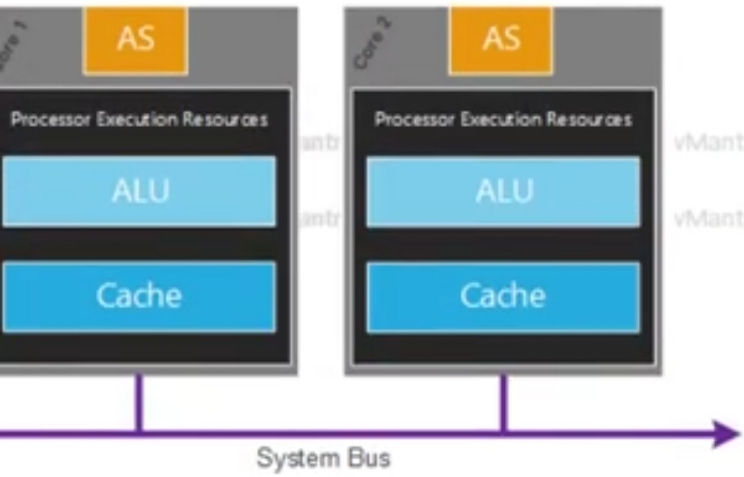



CPU
flow of data
RAM

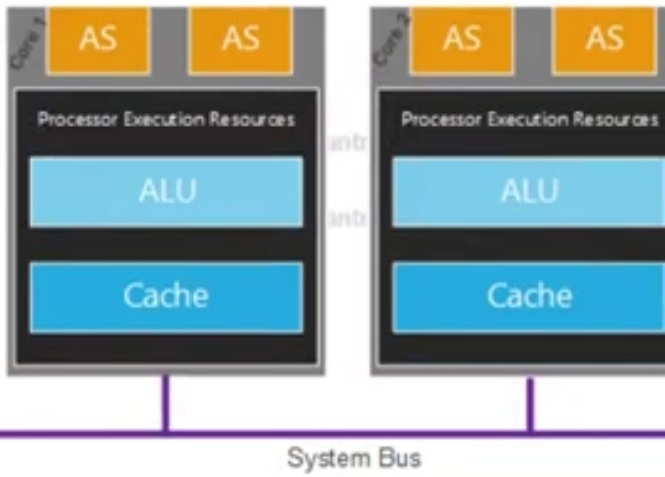

Thread 1
Processor
Thread 2


Dual Core Processor without Hyper-Threading
AS
Processor Execution Resources
ALU
Cache
AS
Processor Execution Resources
ALU
Cache
System Bus
Dual Core Processor with Hyper-Threading enabled
AS
AS
Processor Execution Resources
ALU
Cache
AS
AS
Processor Execution Resources
ALU
Cache
System Bus
HYPER-THREADING PROCESSOR
SMTG
S.M. TECHNO GURU
Subscribe

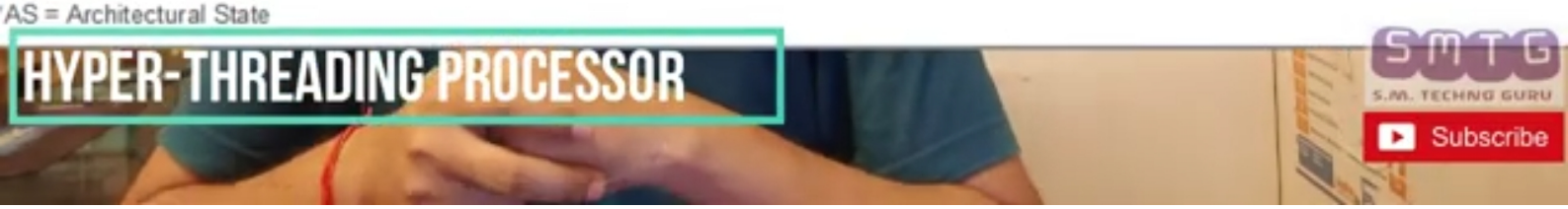
Dual Core Processor without Hyper-Threading

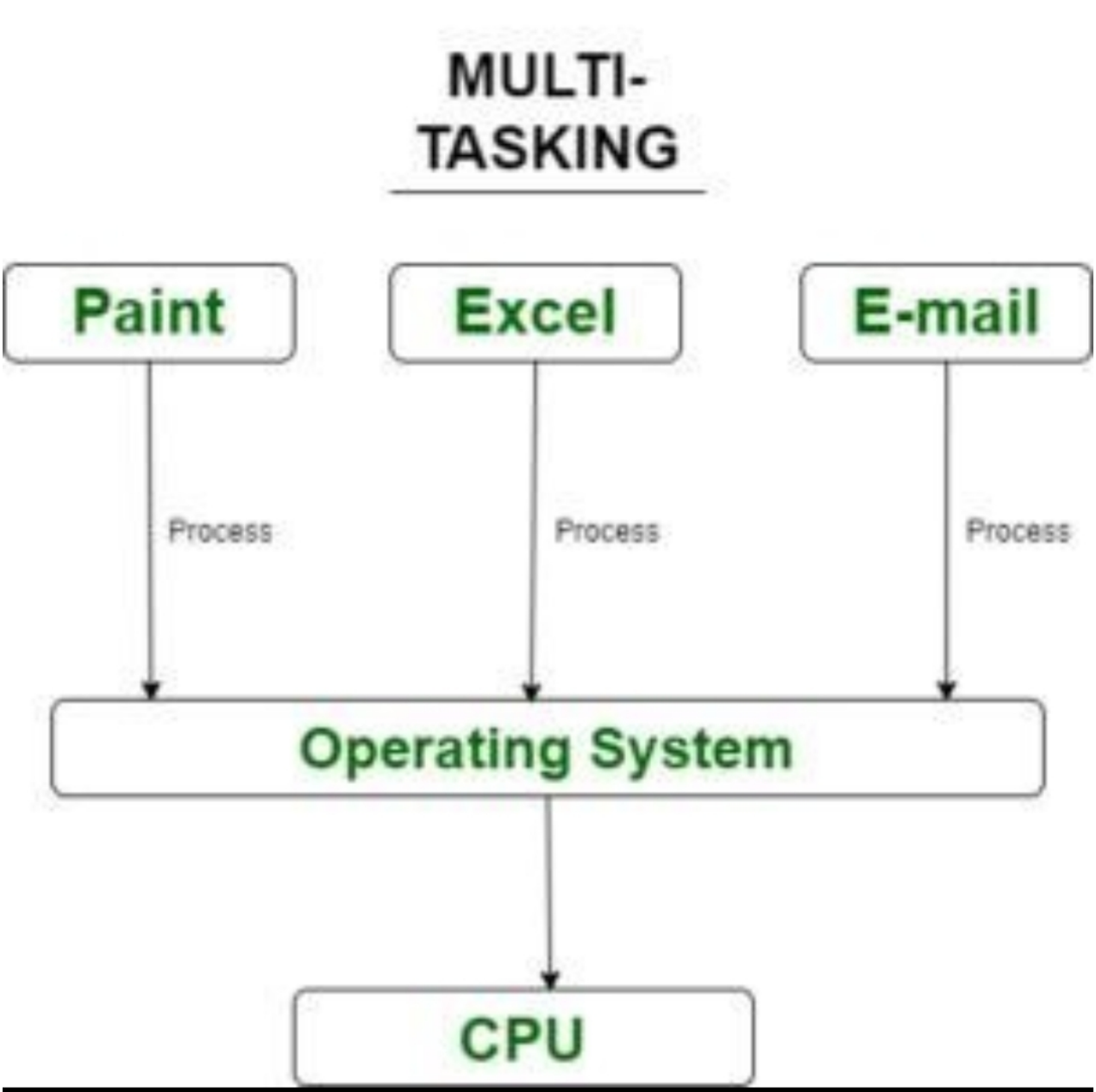


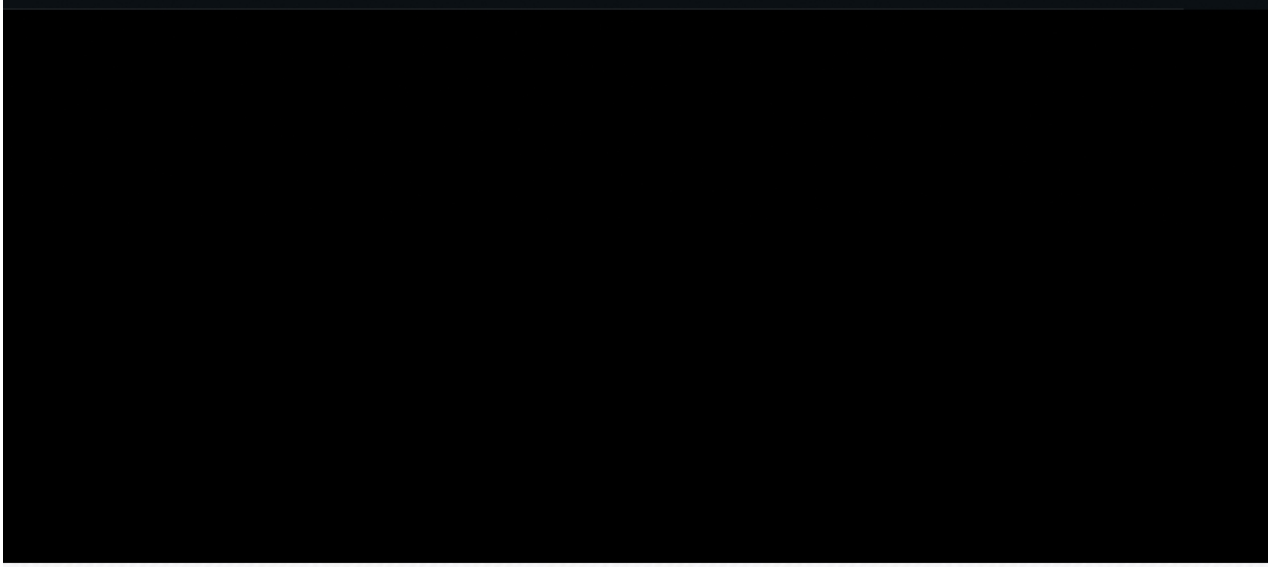
Dual Core Processor with Hyper-Threading enabled

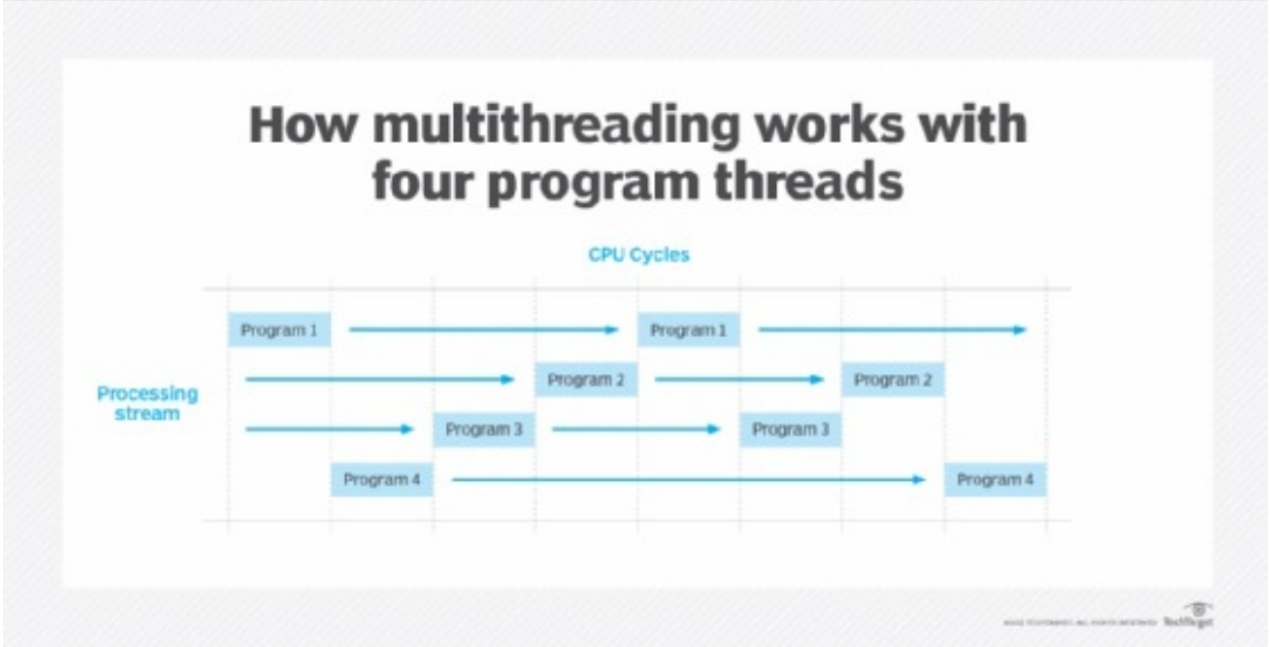


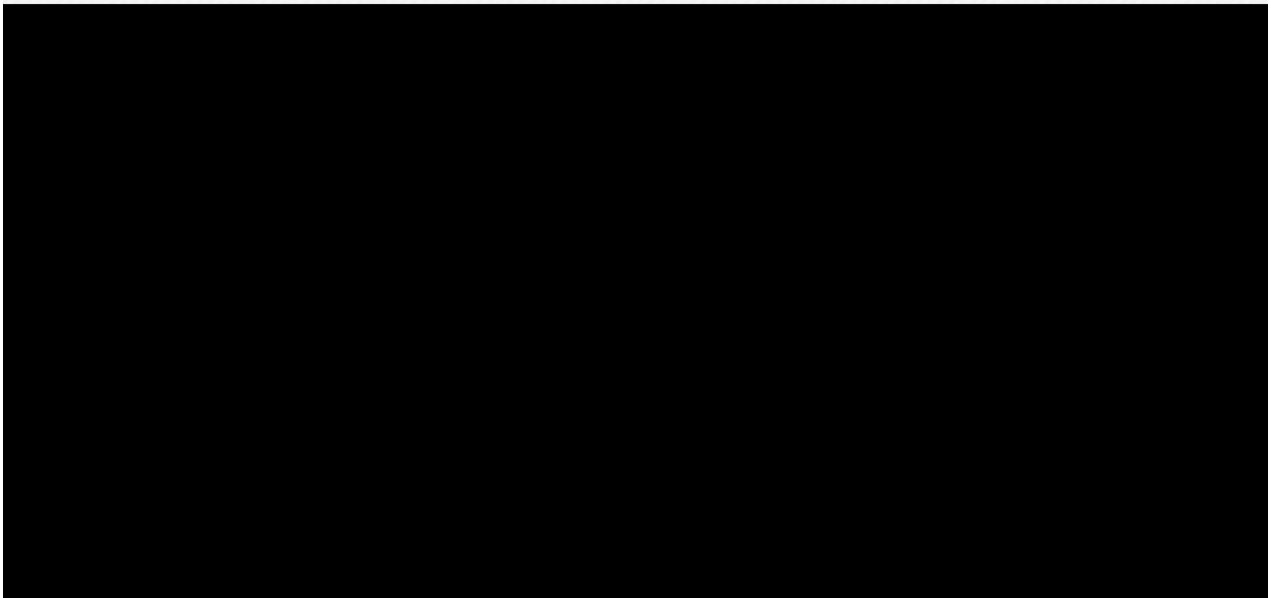
AS = Architectural State

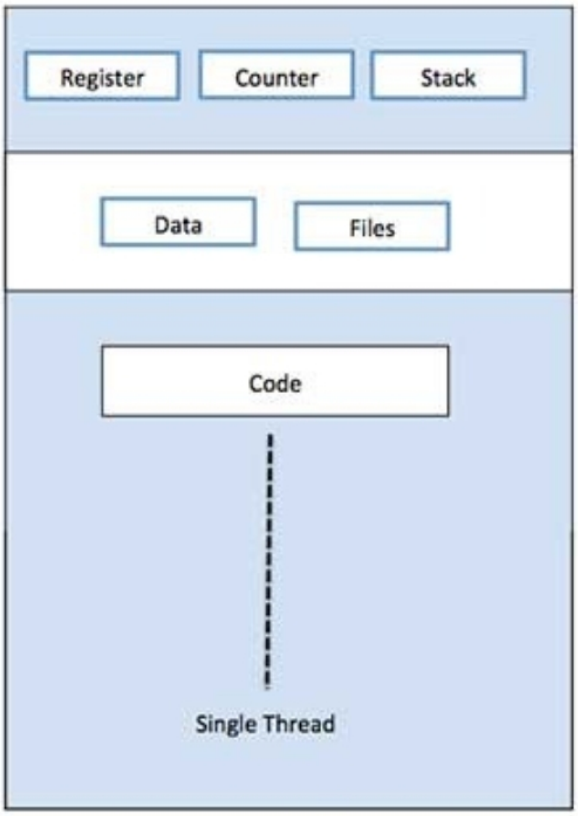




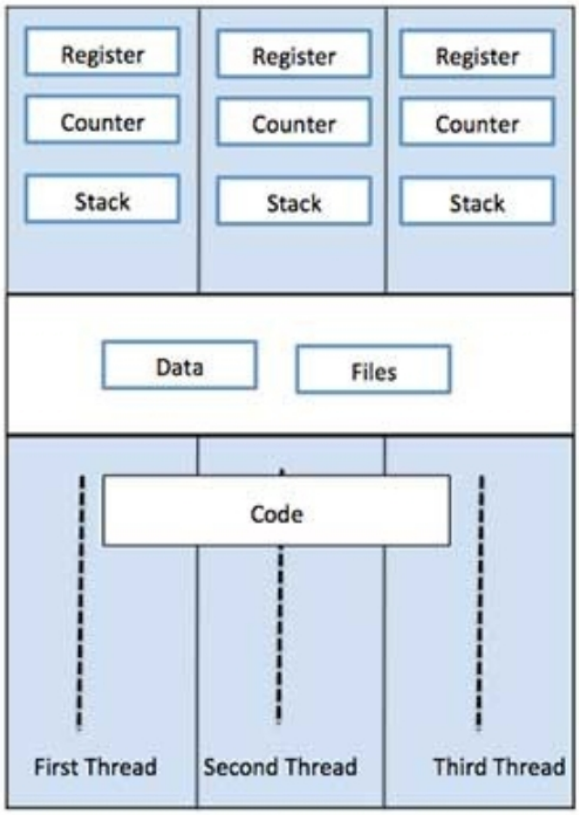



How multithreading works with four program threads
CPU Cycles
Processing stream
Program 1
Program 1
Program 2
Program 2
Program 3
Program 3
Program 4
Program 4

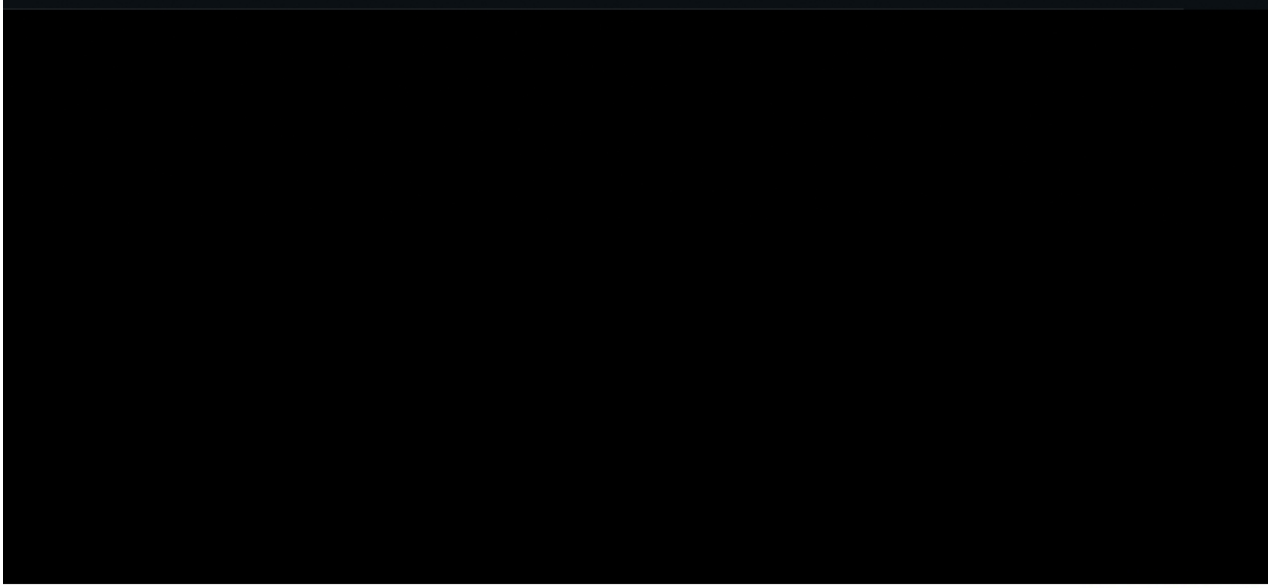


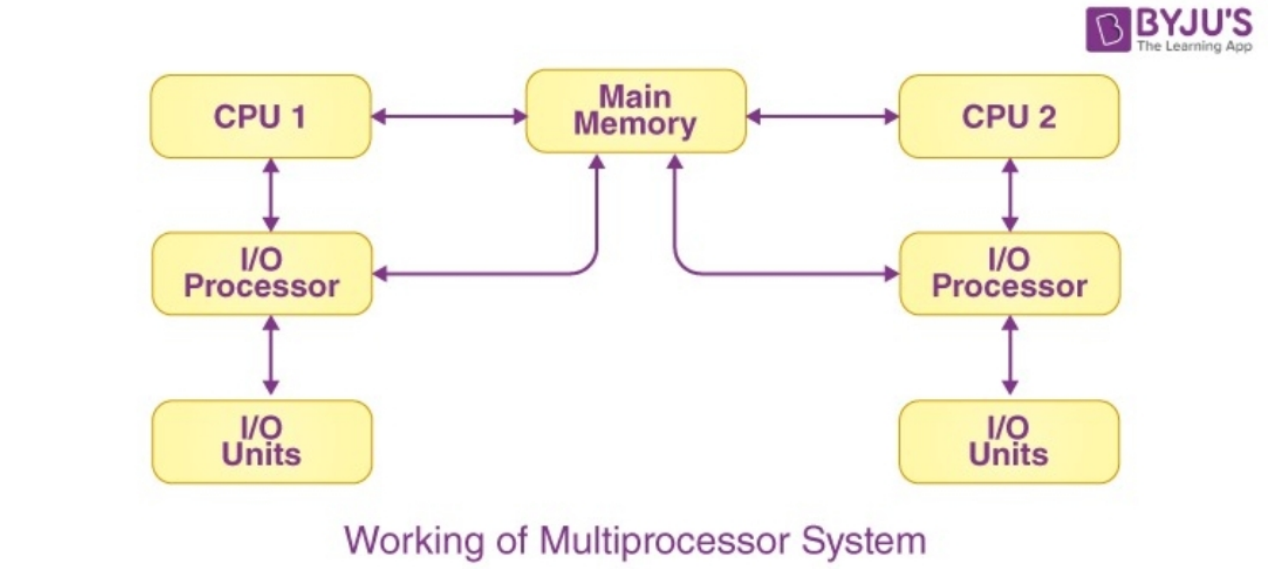


Single Process P with single thread

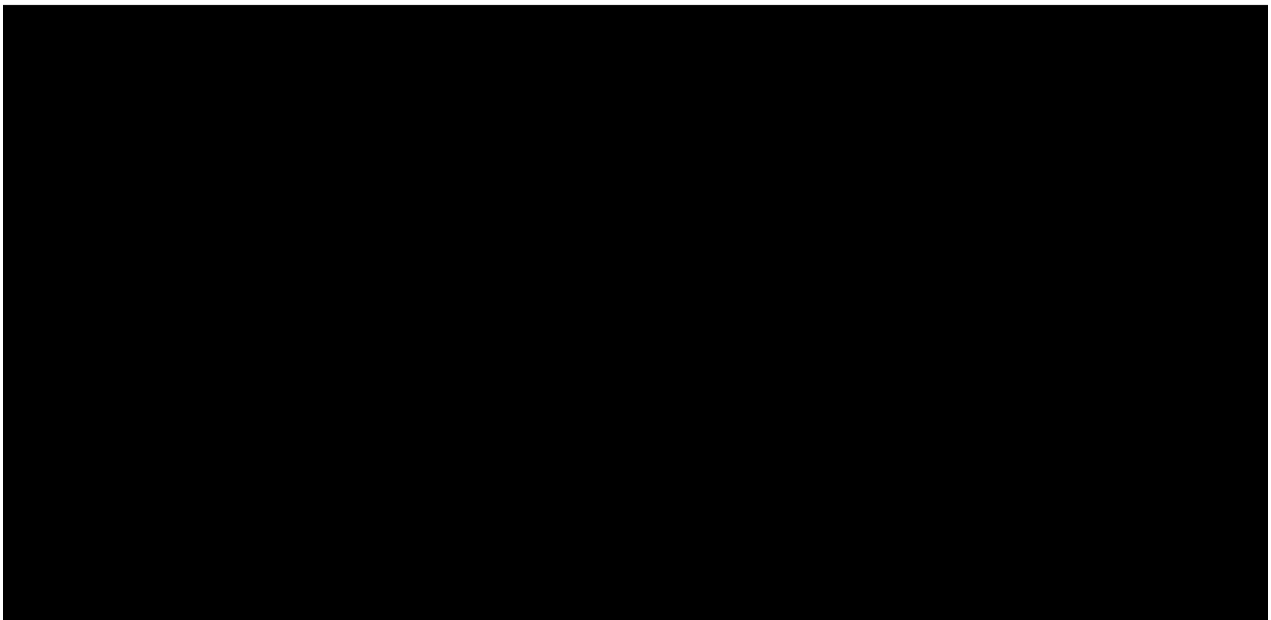


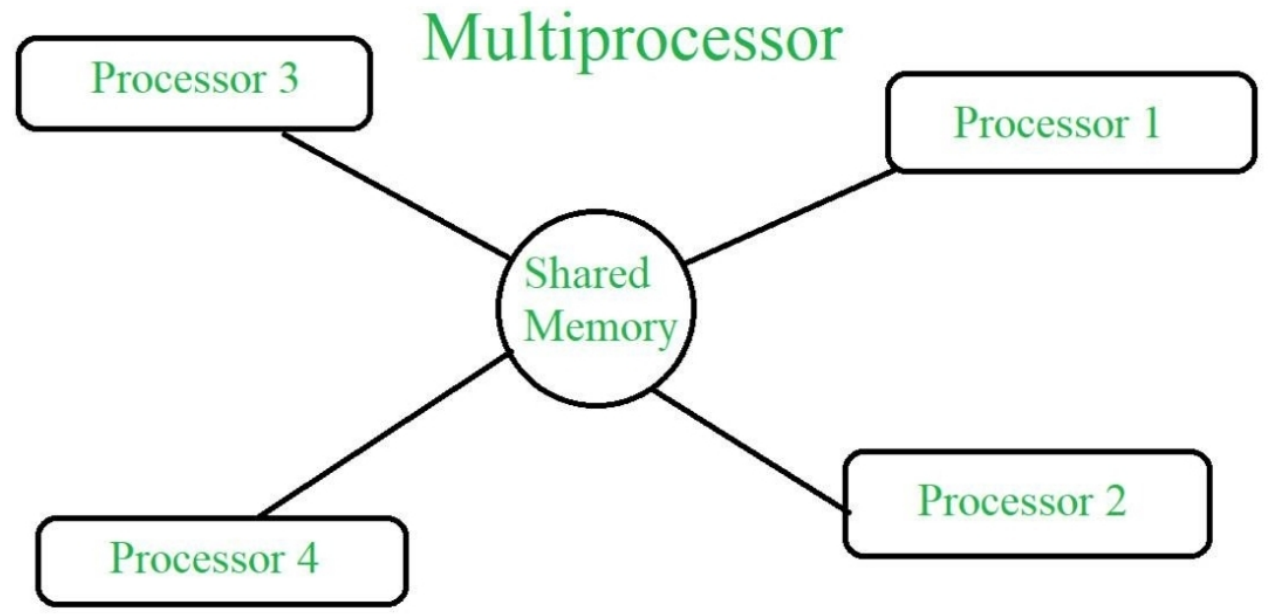
Single Process P with three threads

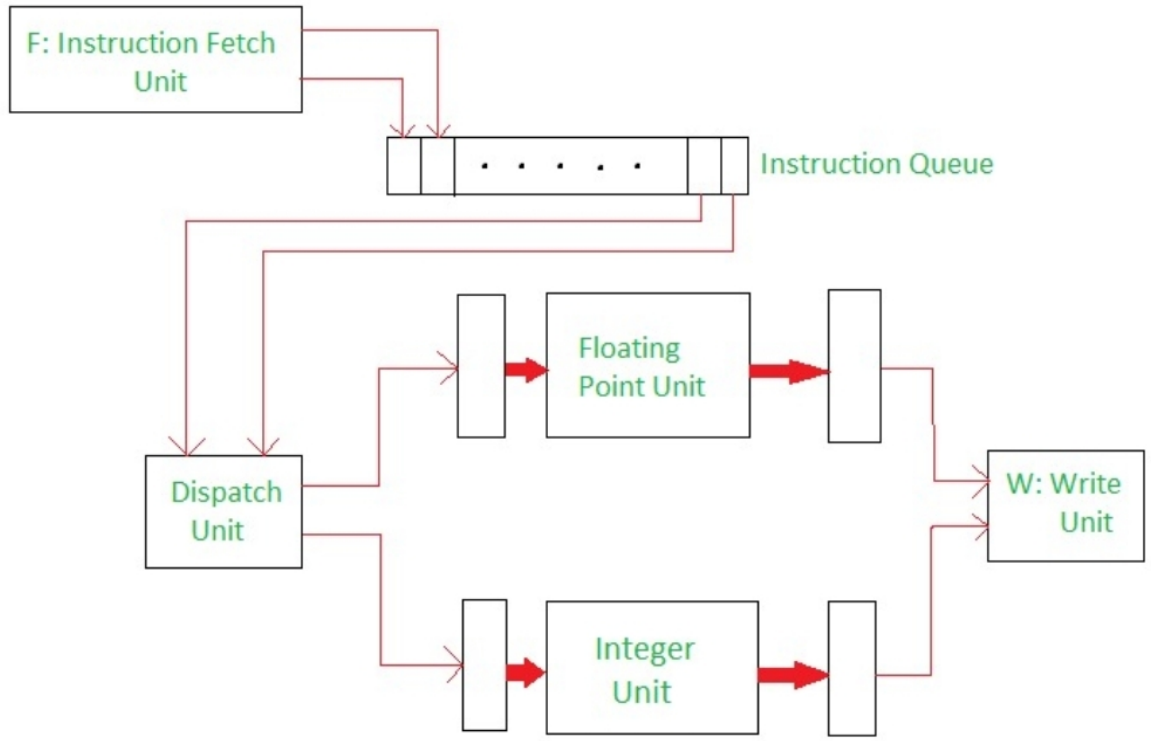




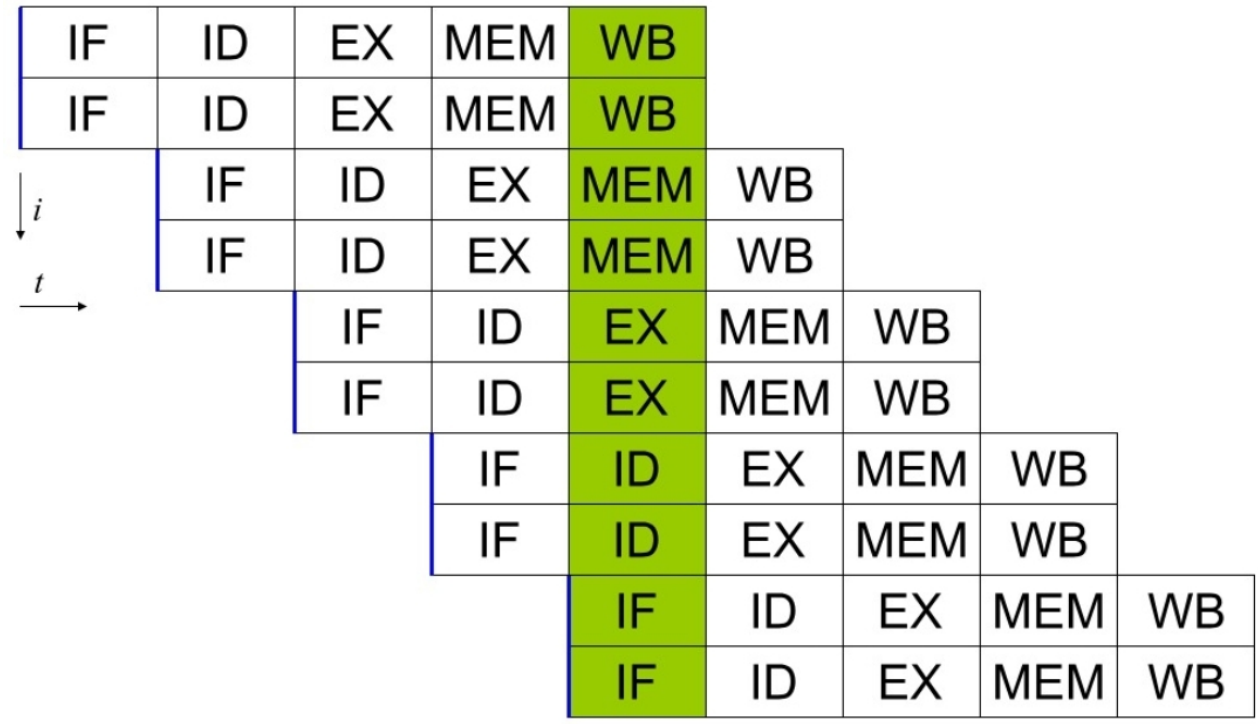
Working of Multiprocessor System

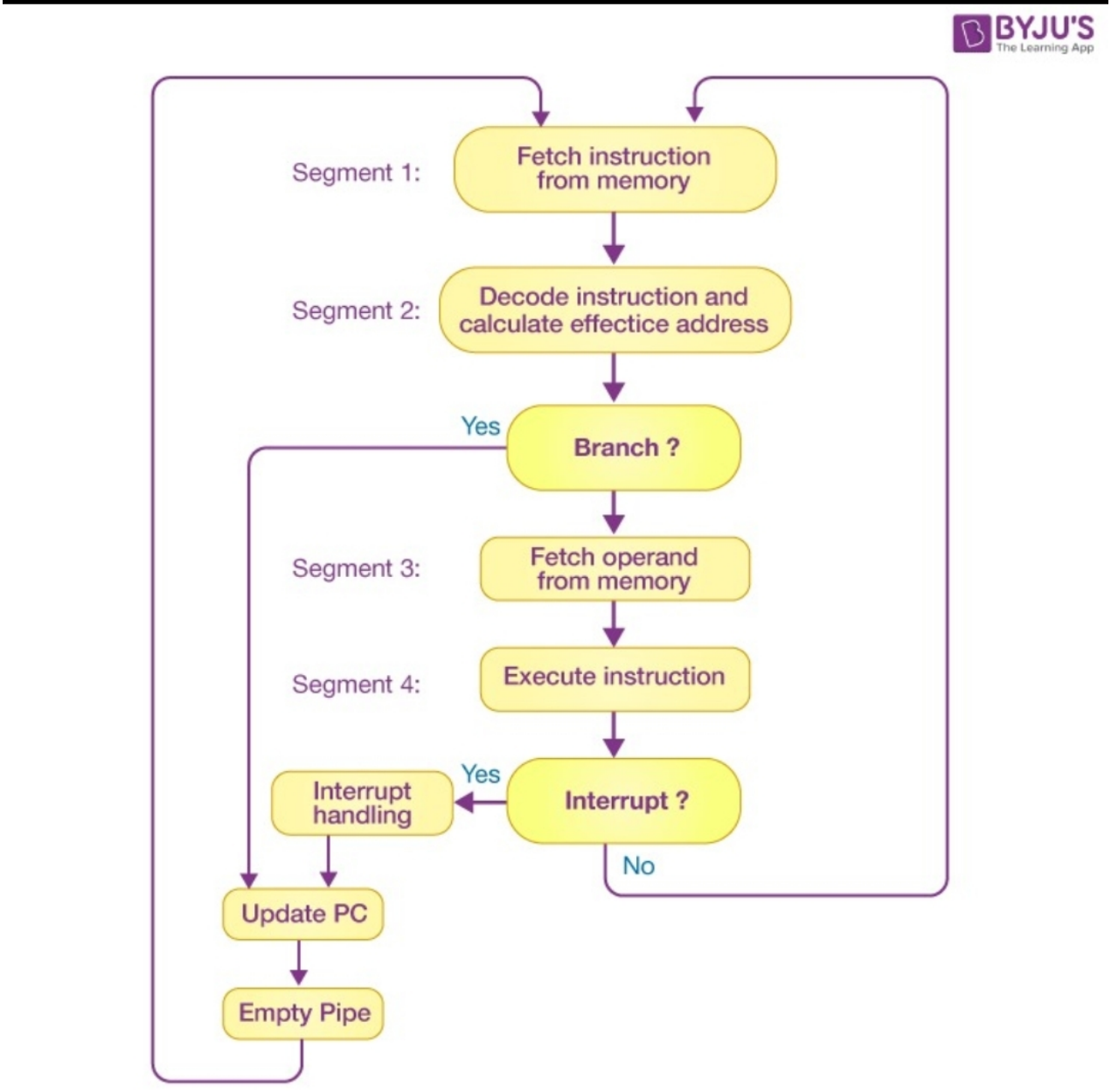


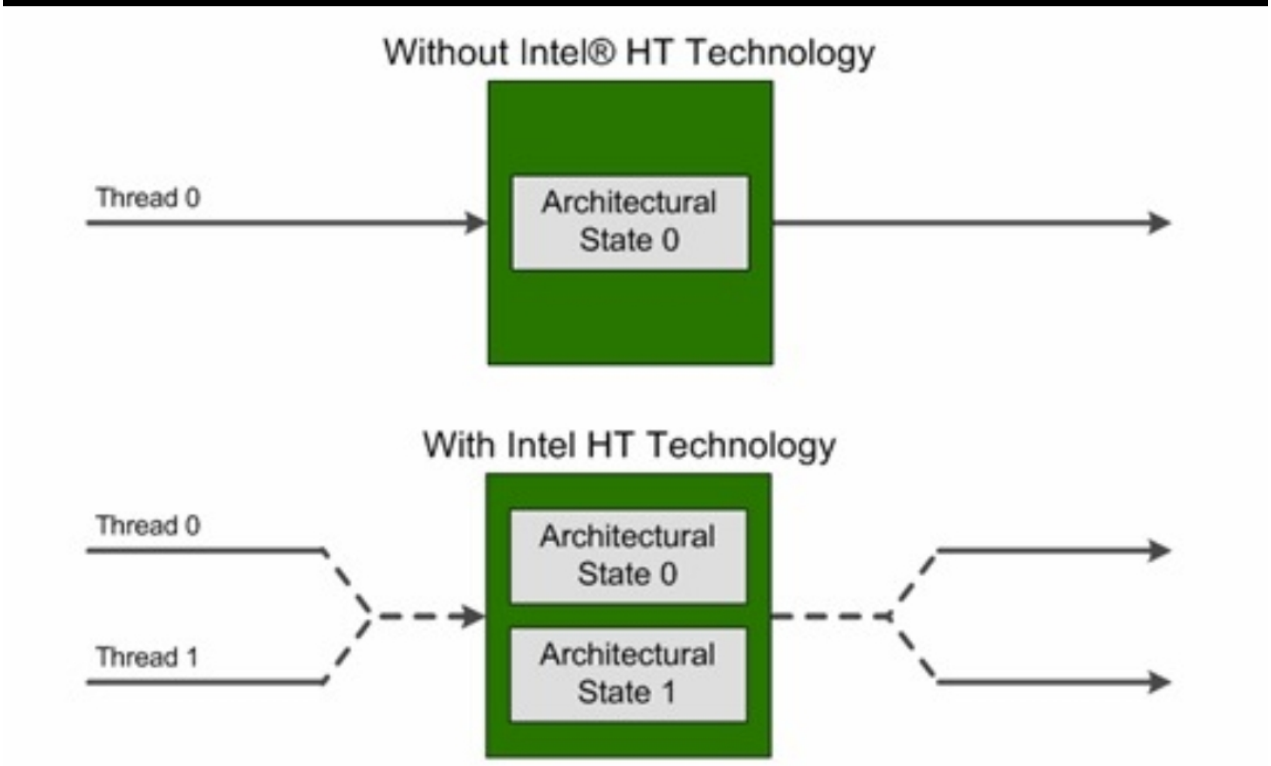




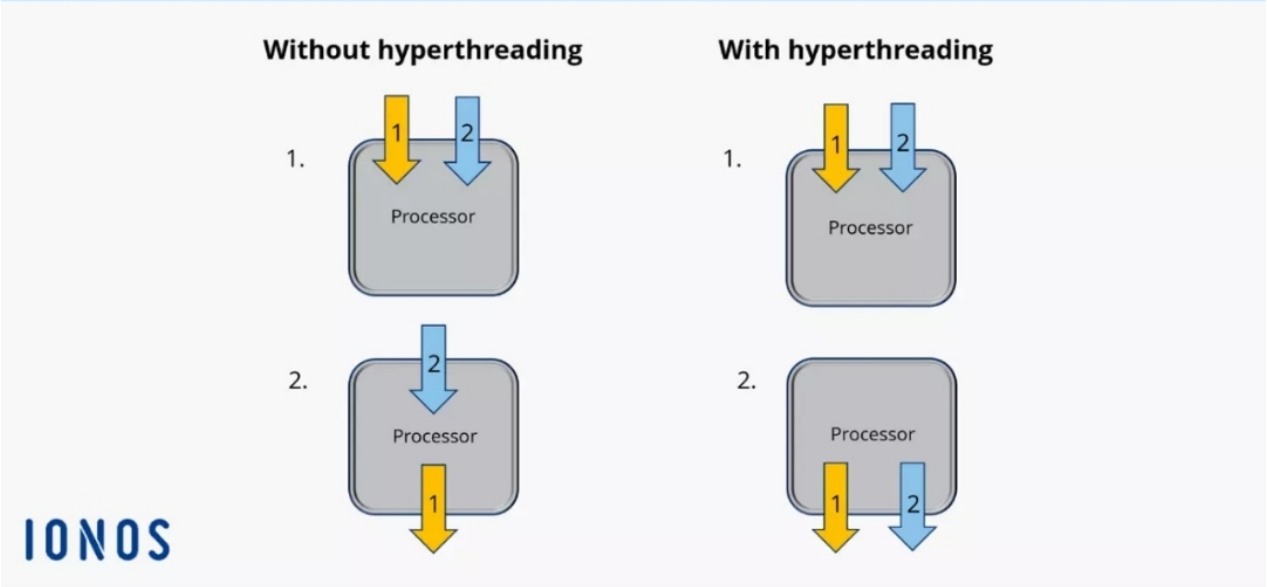
Processor with Two Execution Units

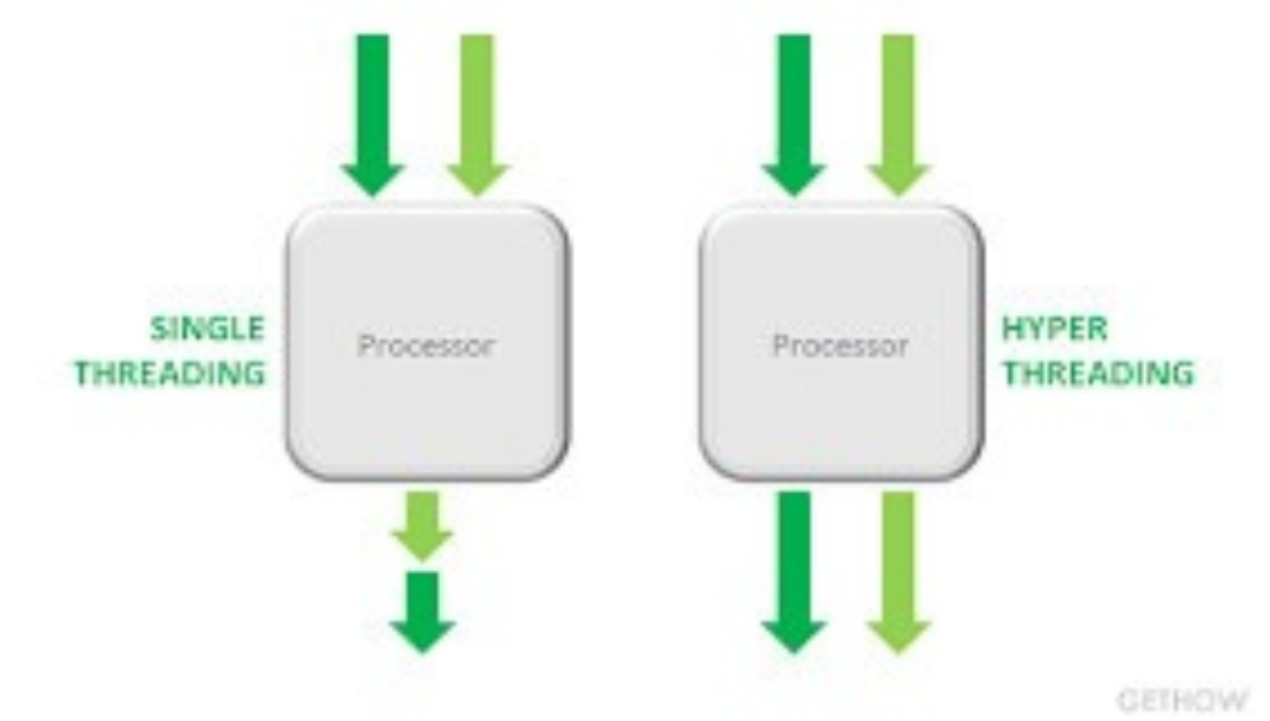






Hyperthreading technology





PROCESS AND THREAD

Process: a program in execution is called process.

Thread: is the segment of process means a process can have multiple threads and these threads are contained within a process.

User level vs kernel level thread

User level thread: executes in user space code and can call kernel space. If user thread need some thing, it will call into kernel

Kernel level thread: executes only kernel code and isn't associated with a user space process.

PROCESS VS THREAD

PROCESS VS THREAD

- Process is program under execution
- Heave weight
- More time required for context switching
- Need more resources
- More time for creation
- More time for termination

- Thread is a segment of process
- Light weight
- Less time required for context switching
- Need less resources
- Less time for creation
- Less time for termination

PROCESS VS THREAD

- Process doesn't share data with each other
- System call is involved
- Process is isolated
- Process has its own PCB, Stack and address space

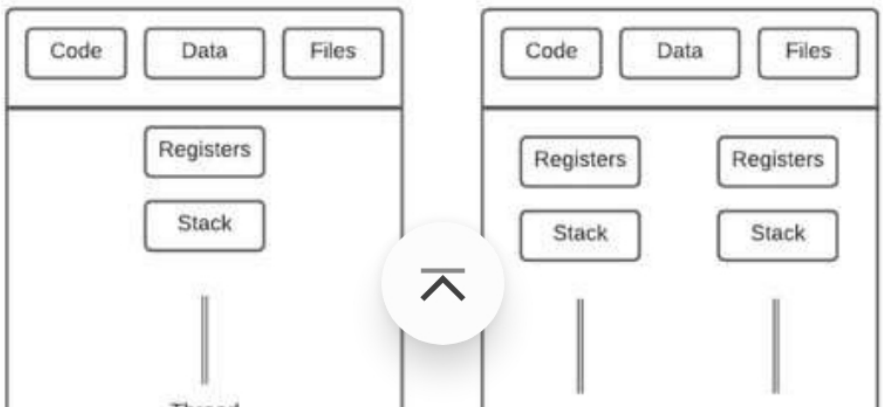
- Threads share data with each other.
- No system call is involved
- Threads share memory.
- Threads has its own TCB, Stack and Address space.

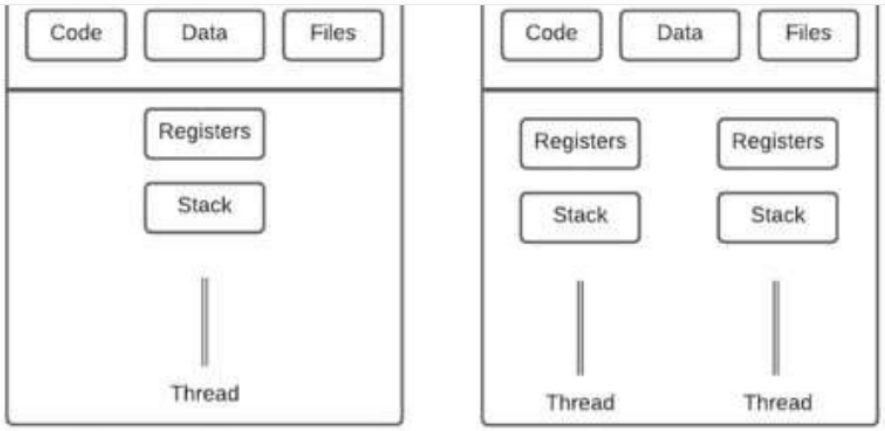
Threads versus Processes

Option 1: Partitioning data set between threads/processes

Process and Thread, EXAMPLE

In a **word processor,** a thread may *check spelling and grammar* while another process user input while yet another third thread loads images from the hard drive and fourth does periodic automatic backups of the file being edited.





Single-threaded Process

Multi-threaded Process

Multitasking vs multithreading

Multitasking allow to run more process concurrently

While, multithreading allows single process to perform multiple tasks in a single process

Both were introduced to improve performance and responsiveness of computing system.

MULTITHREADED PROGRAMMING

Concurrent execution of threads, these threads could run on single processor.

Is the ability of single CPU to provide multiple threads of execution concurrently, supported by operating system

Its aims to increase utilization of single core by using thread level parallelism.

It allows multiple request to be satisfied simultaneously with having to service requests sequentially.

MULTITHREADED PROGRAMMING - ADVANTAGES

Responsiveness: if one thread complete its execution, then its output can be immediately returned.

Faster context switching: context switch time is lower, process context switching requires more overhead from the CPU.

Effective utilization of multiprocessor: Use of multiple threads in a single process make process execution faster. Similarly multiple threads can scheduled on multi processors

MULTITHREADED PROGRAMMING - advantages

Resource Sharing: resources like code, date, files can be shared among all threads within process. Stack and registers can't be shared among threads

Communication: communication between multiple thread is faster.

Increase throughput of the system: each thread's function is considered as one job. Then number of jobs completed per unit time is increased.

BASIS FOR COMPARISON	MULTIPROCESSING	MULTITHREADING
Basic	Multiprocessing a CPUs to increase computing power.	Multithreading creates multiple threads of a single process to increase computing power.

BASIS FOR COMPARISON	MULTIPROCESSING	MULTITHREADING
Basic	Multiprocessing adds CPUs to increase computing power.	Multithreading creates multiple threads of a single process to increase computing power.
Execution	Multiple processes are executed concurrently.	Multiple threads of a single process are executed concurrently.

Scaler processors

- a processors that process only one data item at a time i.e. SISD Processor
- While, Vector processor is a single instruction operates simultaneously on multiple items i.e. SIMD

Super scaler processors

Executes more than one instruction during a clock cycle by simultaneously, dispatching multiple instructions on different functional units on the processors.

Remember, Processors is single but one CPU has different functional units. Such as ALU, Bit shifter

14/22

Super scaler processors

Executes more than one instruction during a clock cycle by simultaneously, dispatching multiple instructions on different functional units on the processors.

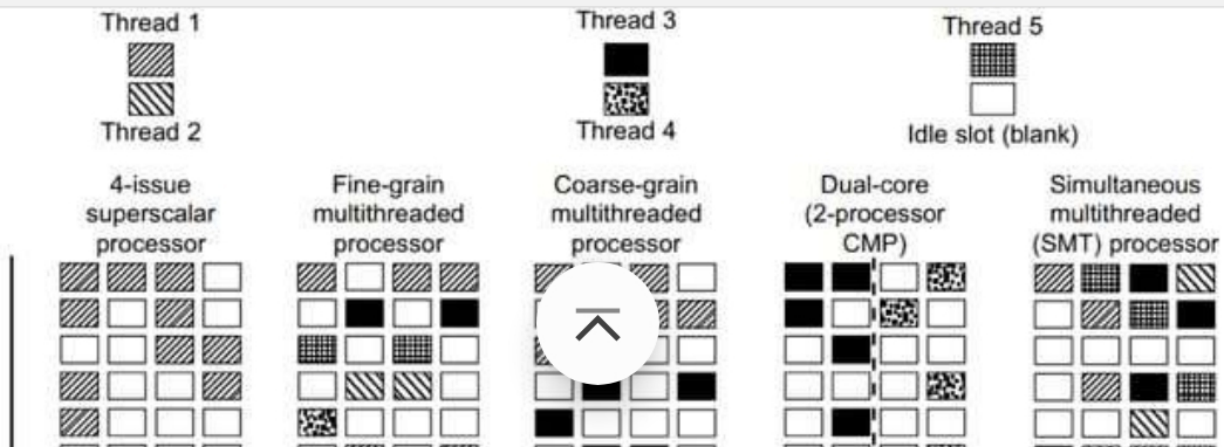
Remember, Processors is single but one CPU has different functional units. Such as ALU, Bit shifter or a multiplier

15/22

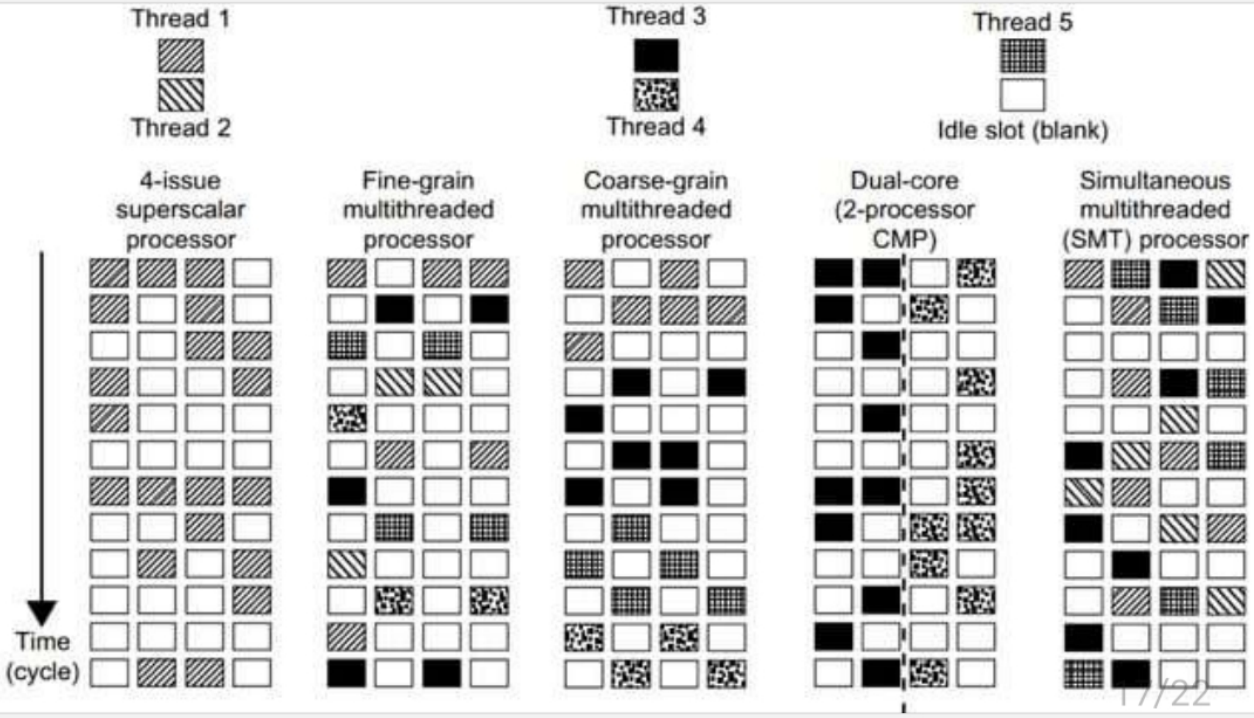
Super scaler processors

It therefore allows more throughput, i.e. the number of instructions can be executed per unit time

16/22



16/22



17/22

Intel hyper threading

Uses processor resources more efficiently, enabling multiple threads to run on each core.

which increases processor throughput, improving overall performance

Latest (HTT) HT Technology are

- ***Intel® Core™ Processor Family***
- ***Intel® Core™ vPro ™ Processor Family***
- ***Intel® Core™ M Processor Family***
- ***Intel® Xeon R Processor Family***

18/22

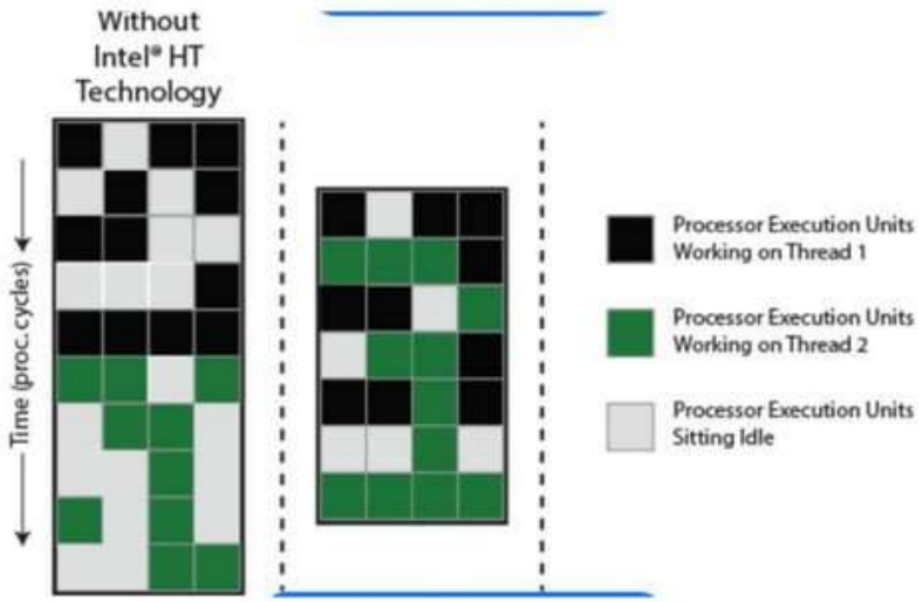
Intel hyper threading

Main function of hyper-threading is to

Intel hyper threading

Main function of hyper-threading is to increase the number of independent instructions in the pipeline,

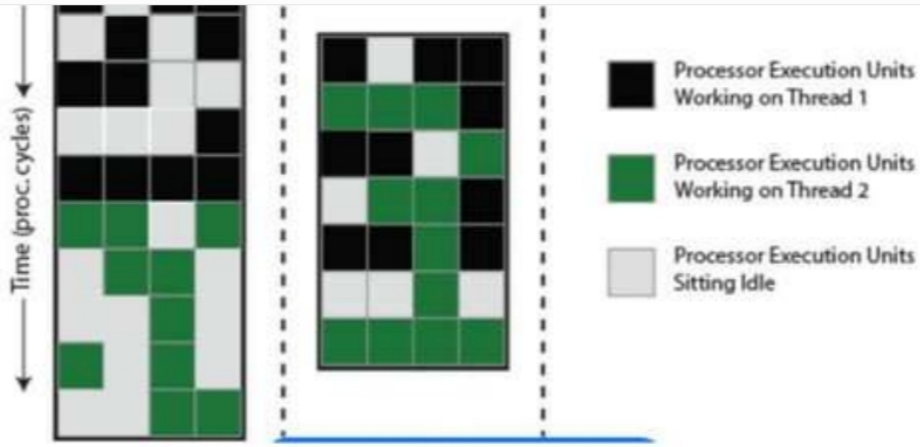
It takes advantages of superscalar architecture in which multiple instructions operate on separate data in parallel



HTT: ADVANTAGES /DISADVANTAGES

Advantages:

- make the best use of each CPU Core.
- Allow running more bac nd data without interruption
- Multithreading enables design and development of



HTT: ADVANTAGES /DISADVANTAGES

Advantages:

- make the best use of each CPU Core.
- Allow running more background data without interruption
- Multithreading enables design and development of smaller chips by eliminating the need for more cores

HTT: ADVANTAGES /DISADVANTAGES

Disadvantages:

Technically, doesn't double the performance but maximize its performance.

Some software doesn't support multithreading, so software implementation is required.

Generate more heat, to avoid performance degradation and overheating, cooling system must be implemented