
**Super Computing and Distributed Systems Camp**
**Catay, Santander, Colombia**
**August 15-22, 2010**

# Overview: Multicore Architectures

*Gilberto Díaz*
gilberto@ula.ve

Universidad de Los Andes


**Super Computing and Distributed Systems Camp**
**Catay, Santander, Colombia**
**August 15-22, 2010**

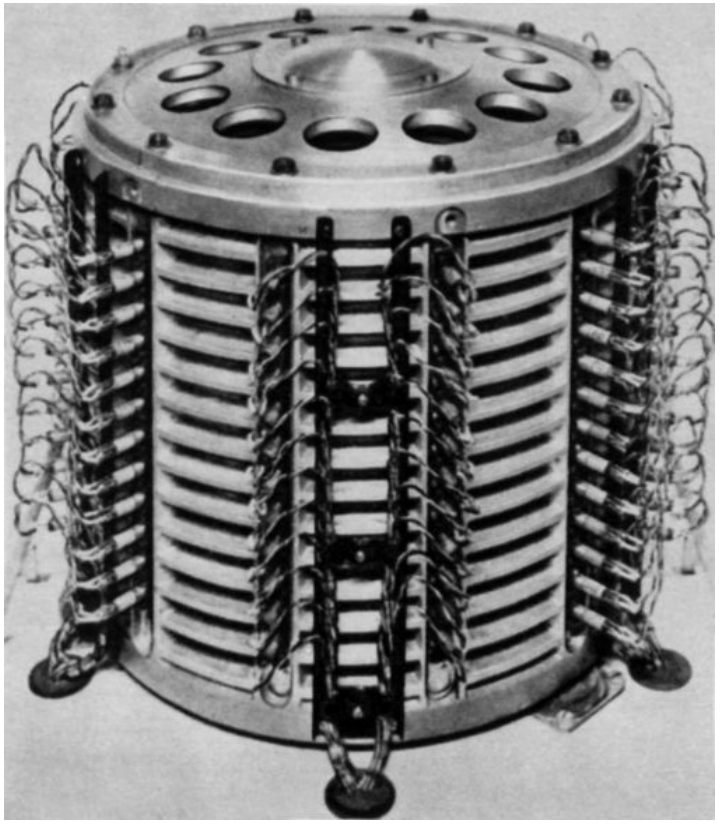
# Technological Advances in Processors

*Gilberto Díaz*
gilberto@ula.ve

Universidad de Los Andes

# *Technological advances in Processors*

In the early days of computing the CPU generally ran slower than its memory.



# *Technological advances in Processors*

In 70s, the performance lines crossed because developers start to add some advance features to processors in order to get higher performance.

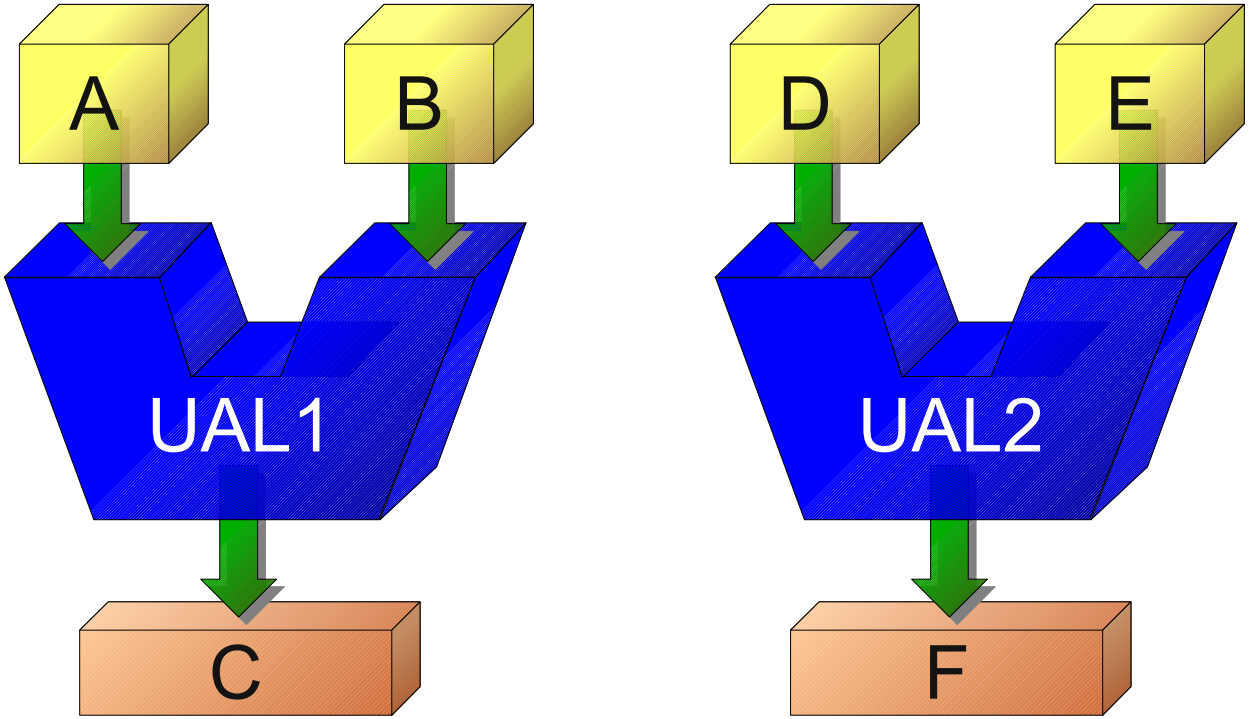
But, CPUs, increasingly starved for data.

Next, some of those features are described.

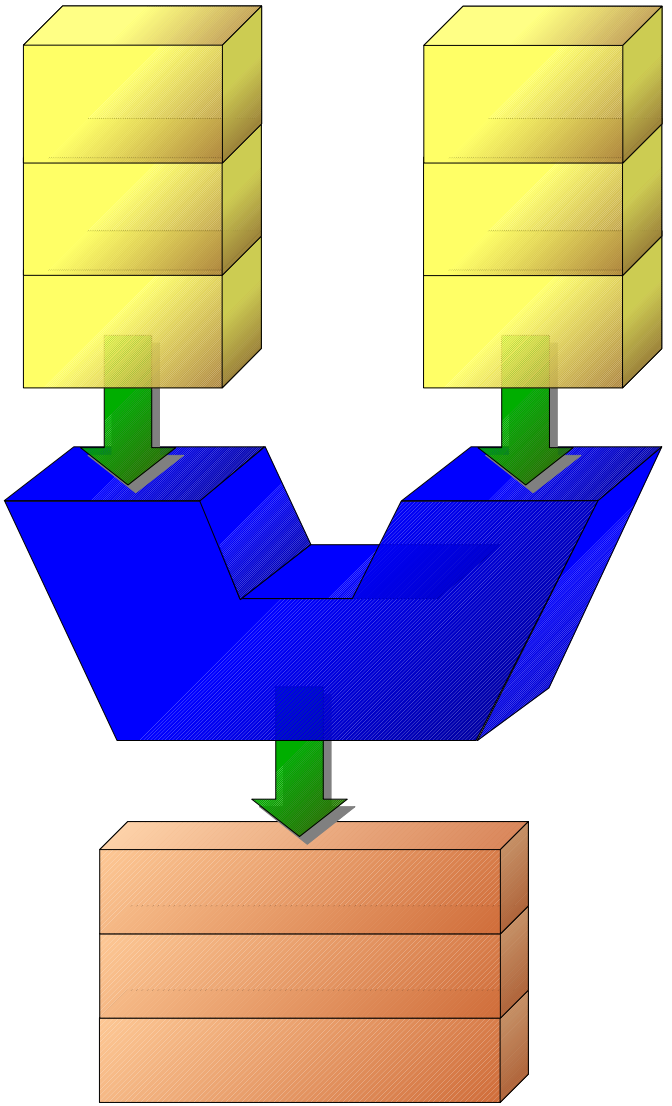
# Technological advances in Processors

More than one aritmethic logic unit

(super scalar)

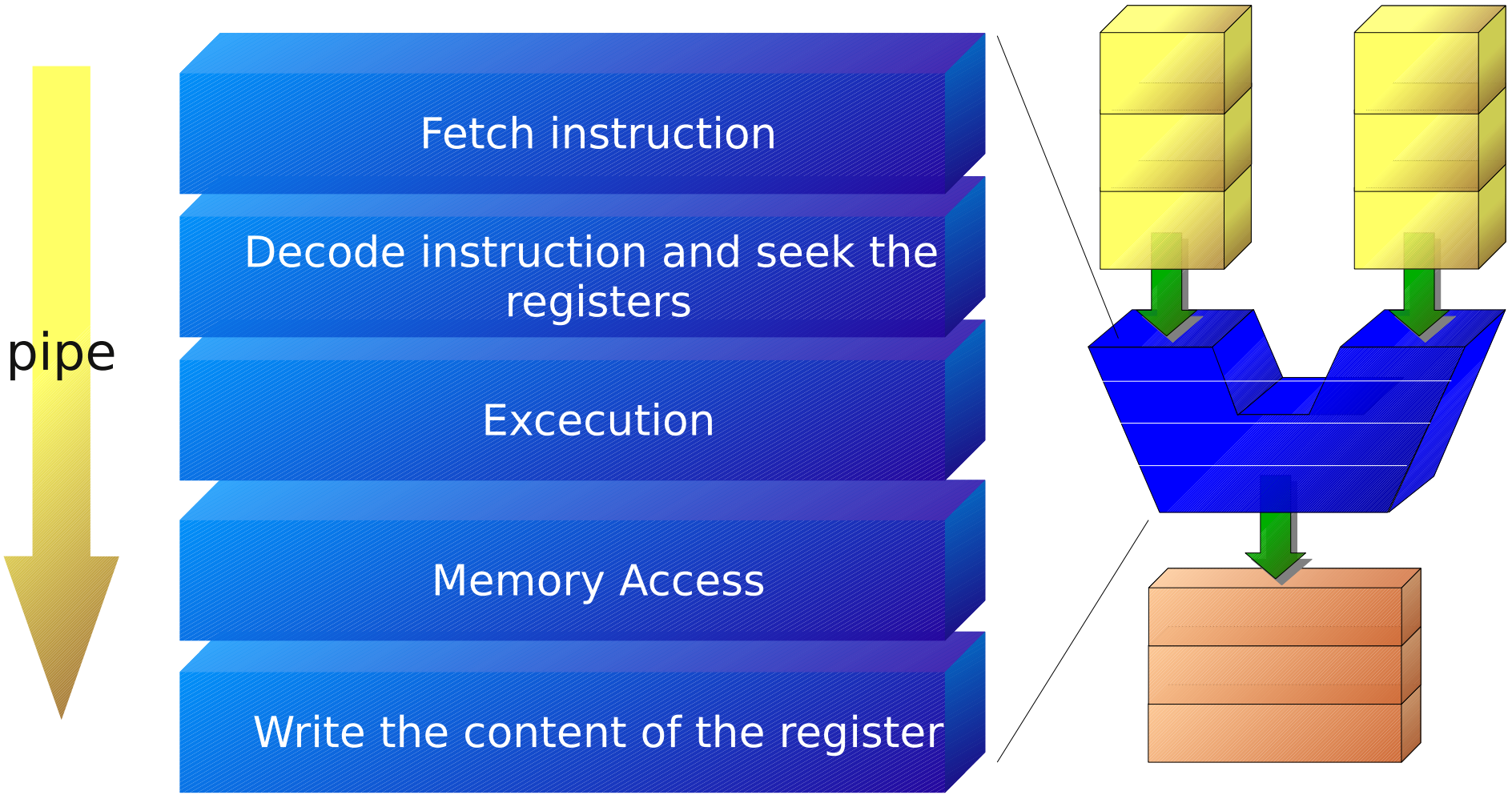


# Vectorial Processing



# Technological advances in Processors

## Pipelining



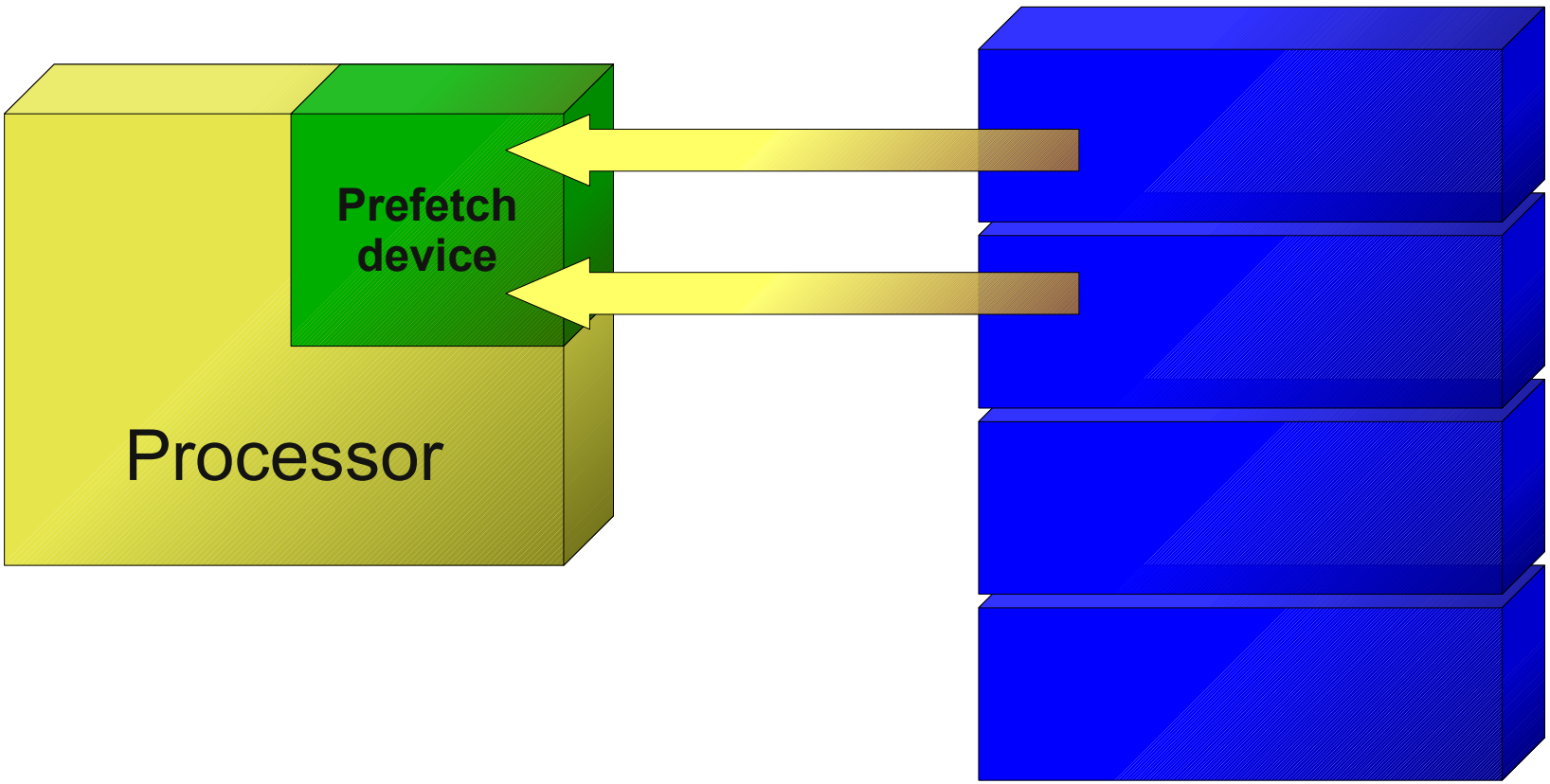
## Pipelining

Pentium 4 20 stages

Pentium D 31 stages

# Technological advances in Processors

- Prefetching instructions and data

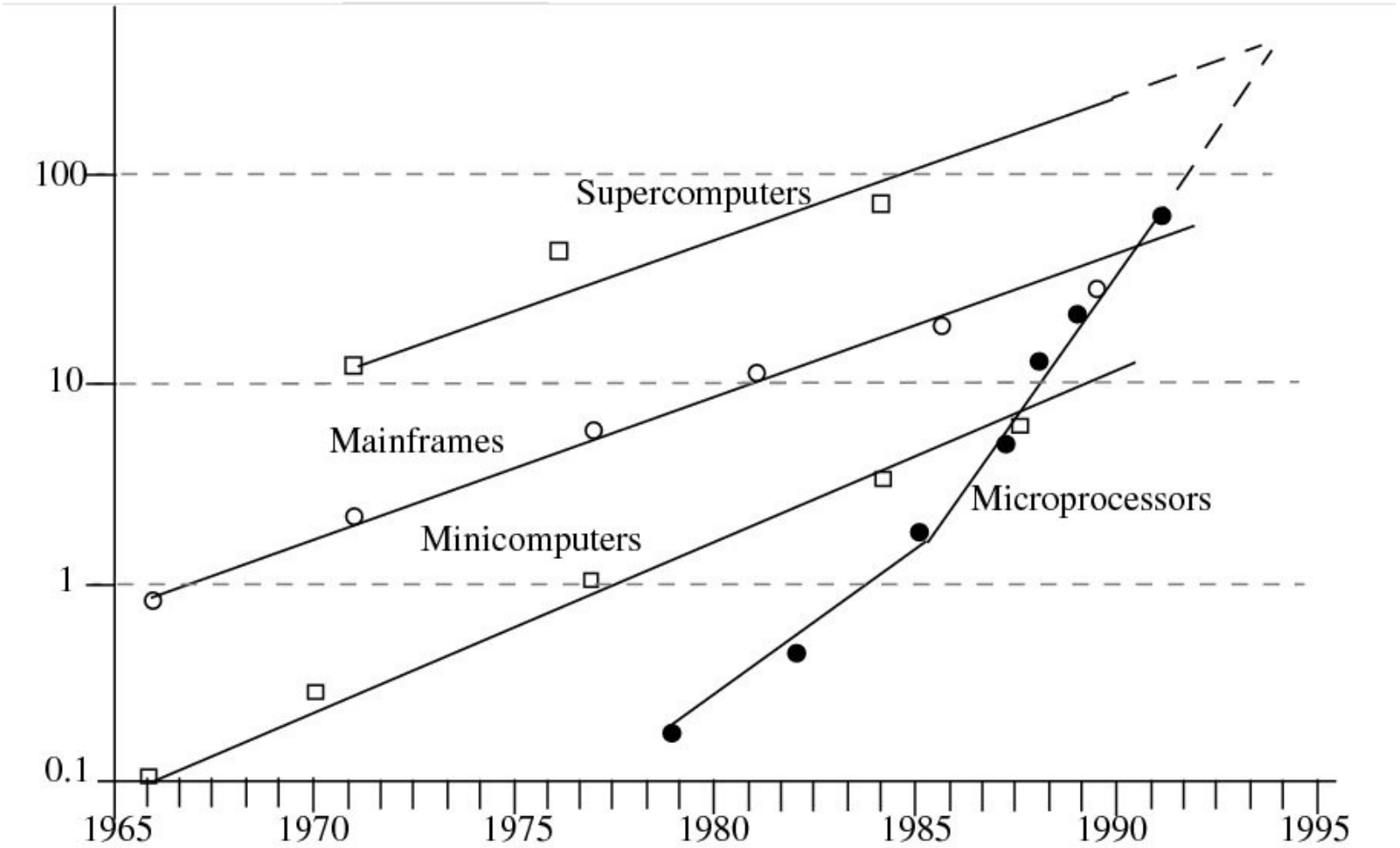


# Technological advances in Processors

- Instructions excecution in advance

```
if ( a == b ){
    Instruction A;
    Instruction B;
}
else{
    Instruction C;
    Instruction D;
}
```

# Technological advances in Processors




Source: “Parallel Computer Architecture” Culler

# *Technological advances in Processors*

In the 80s and 90s developers start to add mechanisms to feed processors with instructions and data in a faster way.

Source: “Parallel Computer Architecture” Culler


**Super Computing and Distributed Systems Camp**
**Catay, Santander, Colombia**
**August 15-22, 2010**

# Multicore Processors

*Gilberto Díaz*
gilberto@ula.ve

Universidad de Los Andes


# Multicores Processors

Chip-making technologies reached the physical limits of the technology.

Therefore, another way to improving a computer's performance is to add extra processors, as in symmetric multiprocessing designs, which have been popular in servers and workstations since the early 1990s.

# Multicores Processors

At the begining, multiprocessor machines were developed. They were conformed by several processors sharing the same memory space.

Every processor was built in a separate silicon chip.

# Multicores Processors

In computing, symmetric multiprocessing or SMP involves a multiprocessor computer hardware architecture where two or more identical processors are connected to a single shared main memory and are controlled by a single OS instance.

# Multicores Processors

Processors were originally developed with only one core. The core is the part of the processor that actually performs the reading and executing of the instruction.

# Multicores Processors

A multi-core processor is a processing system composed of two or more independent cores. One can describe it as an integrated circuit to which two or more individual processors (called cores in this sense) have been attached.

# Multicores Processors

In 2005, the first personal computer dual-core processors were announced and as of 2009 dual-core and quad-core processors are widely used in servers, workstations and PCs.

# Memory Paradigms

*Gilberto Díaz*
gilberto@ula.ve

Universidad de Los Andes

# Memory Paradigms

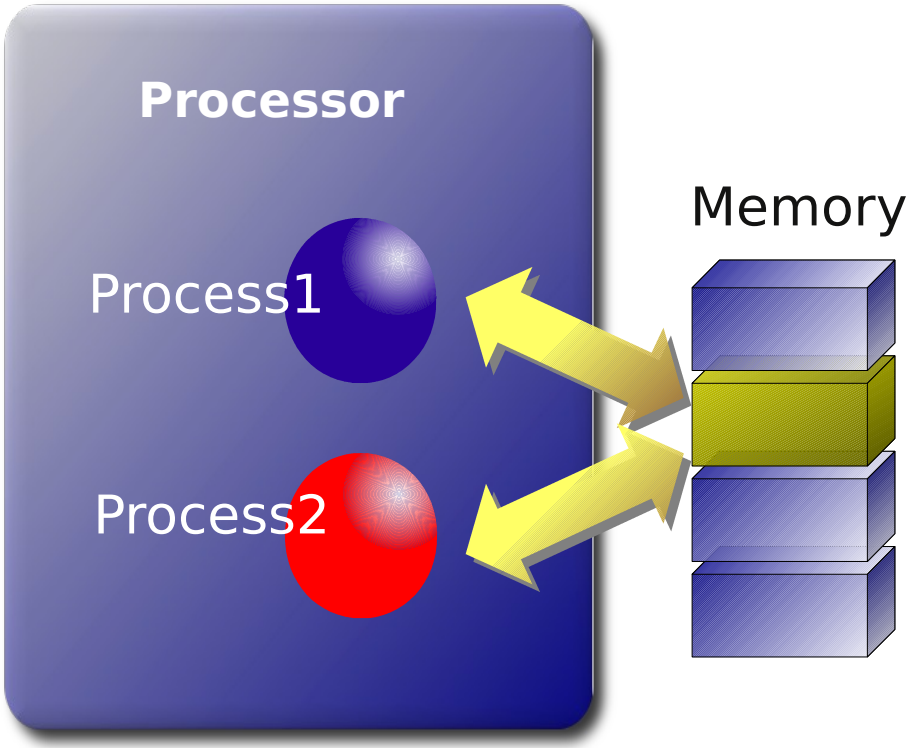
According to memory management, parallel systems are classified in 3 categories:

- Shared Memory
- Distributed Memory
- Distributed Shared Memory

# Memory Paradigms

## Shared Memory

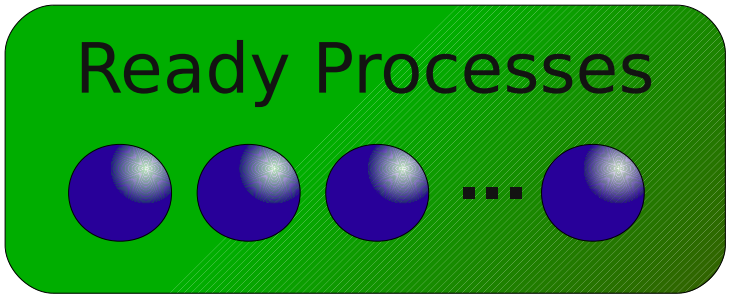
Processes can access the same physical memory location. They can run in the same processor. (shared time technique).



# Memory Paradigms

## Shared Time

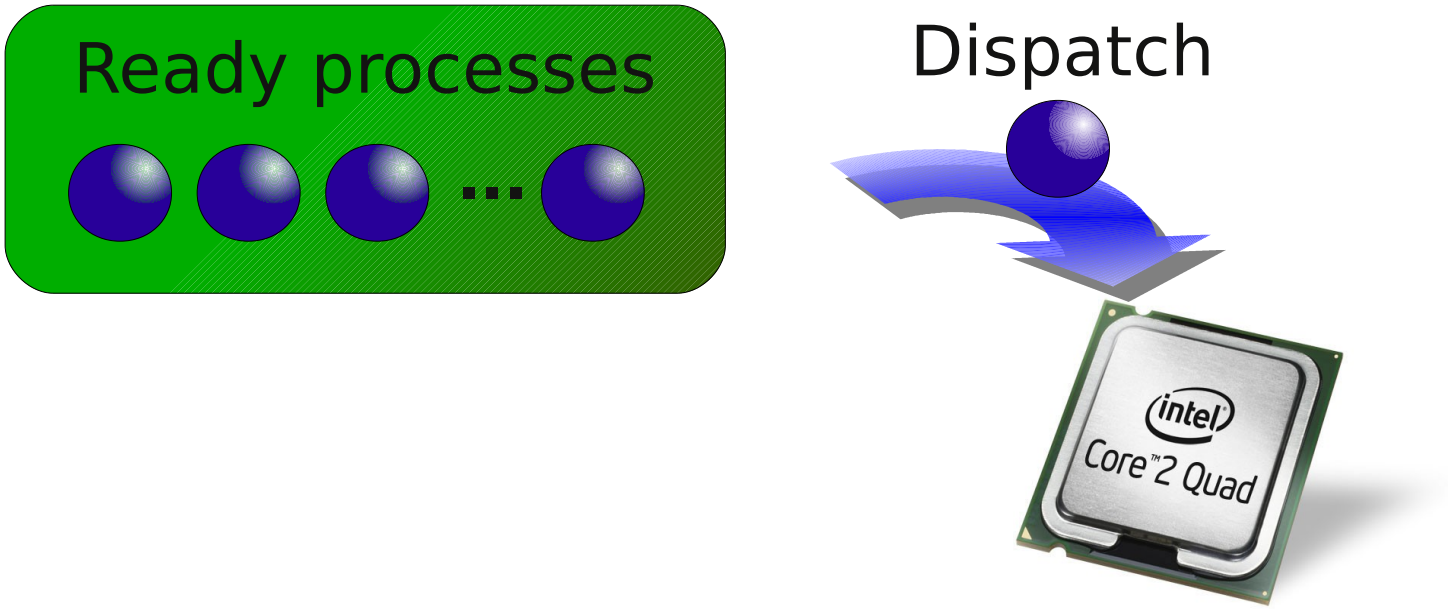
Multi task Operating Systems utilize a technique named shared time to excecute processes.




# Memory Paradigms

## Shared Time

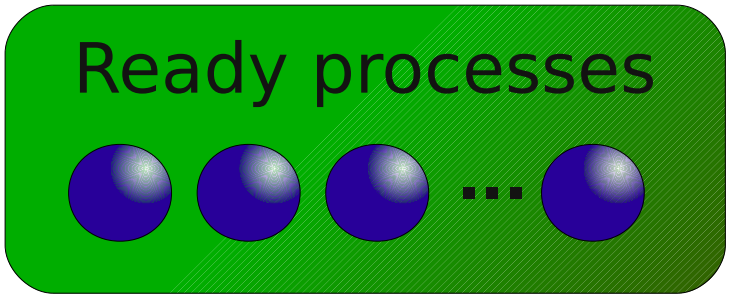
Operating system dispatch the first process of the ready processes queue.



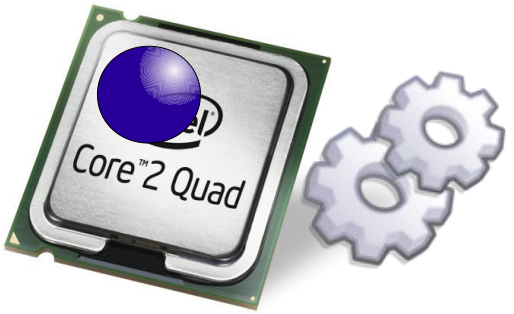
# Memory Paradigms

## Shared Time

Process begins its excecution



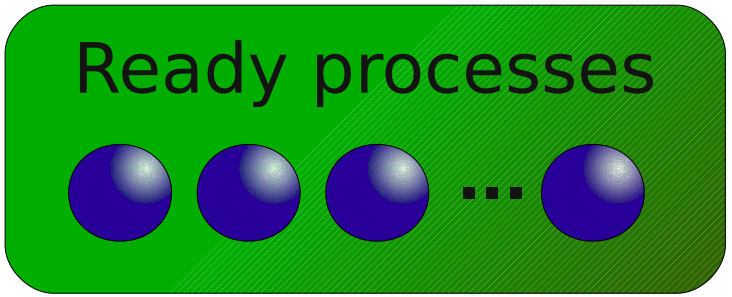

Running



# Memory Paradigms

## Shared Time

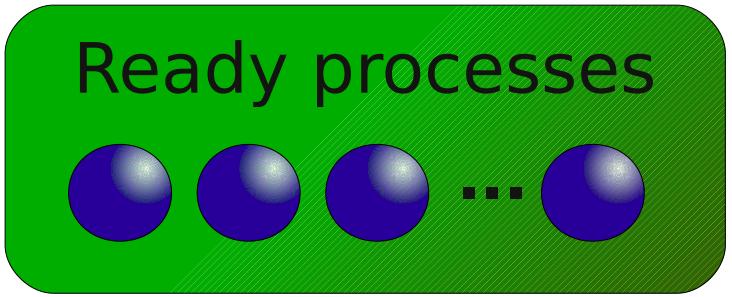
O.S. asigns a fixed time (*quantum*) to each process to run.




# Memory Paradigms

## Shared Time

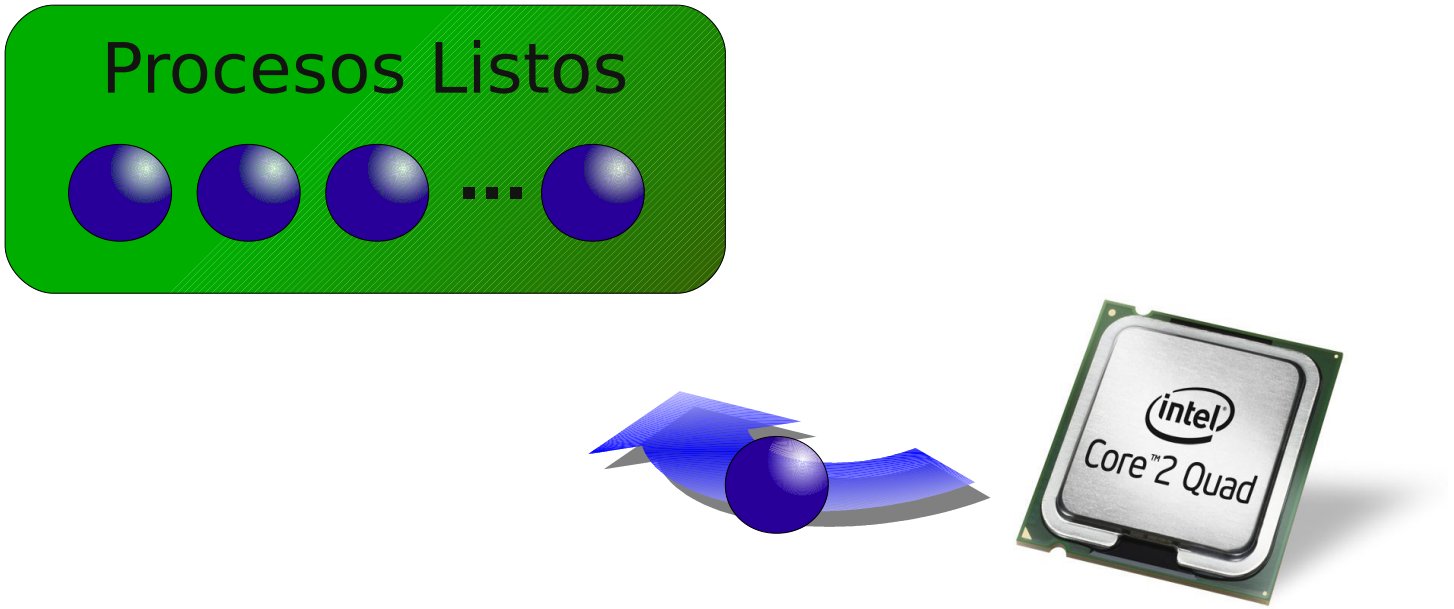
Process runs until it finish.




# Memory Paradigms

## Shared Time

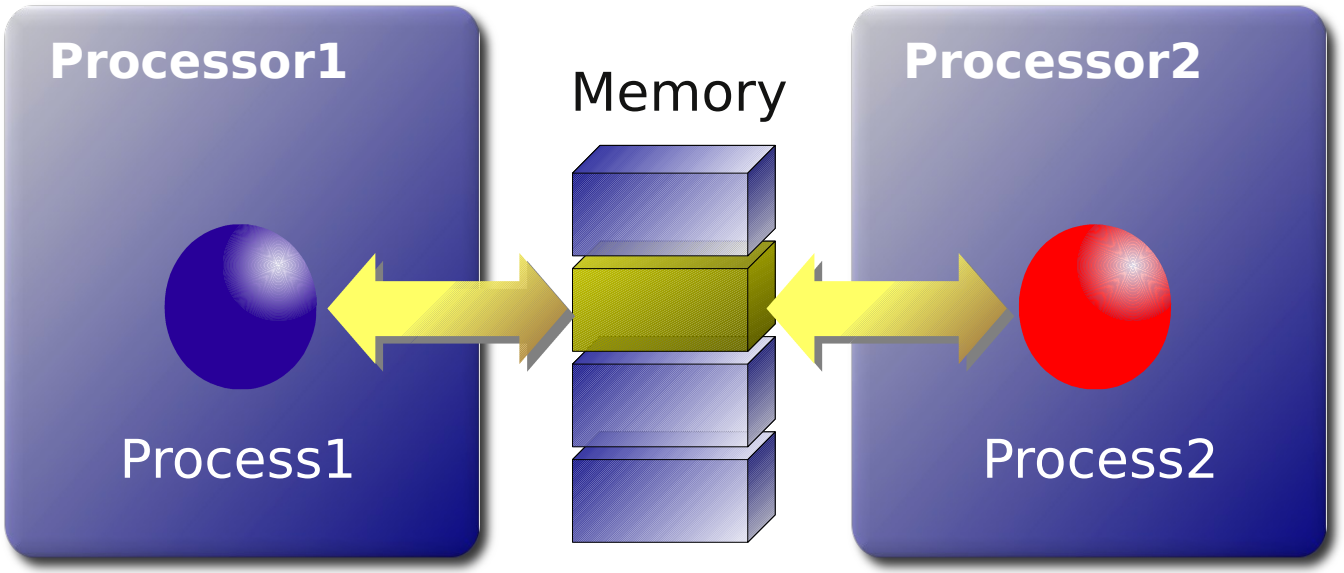
Or until *quantum* expires. In this case the process comes back to the queue.



# Memory Paradigms

## Shared Memory

Processes can be excecuted in different processors in the same computer.



# Memory Paradigms

## Shared memory

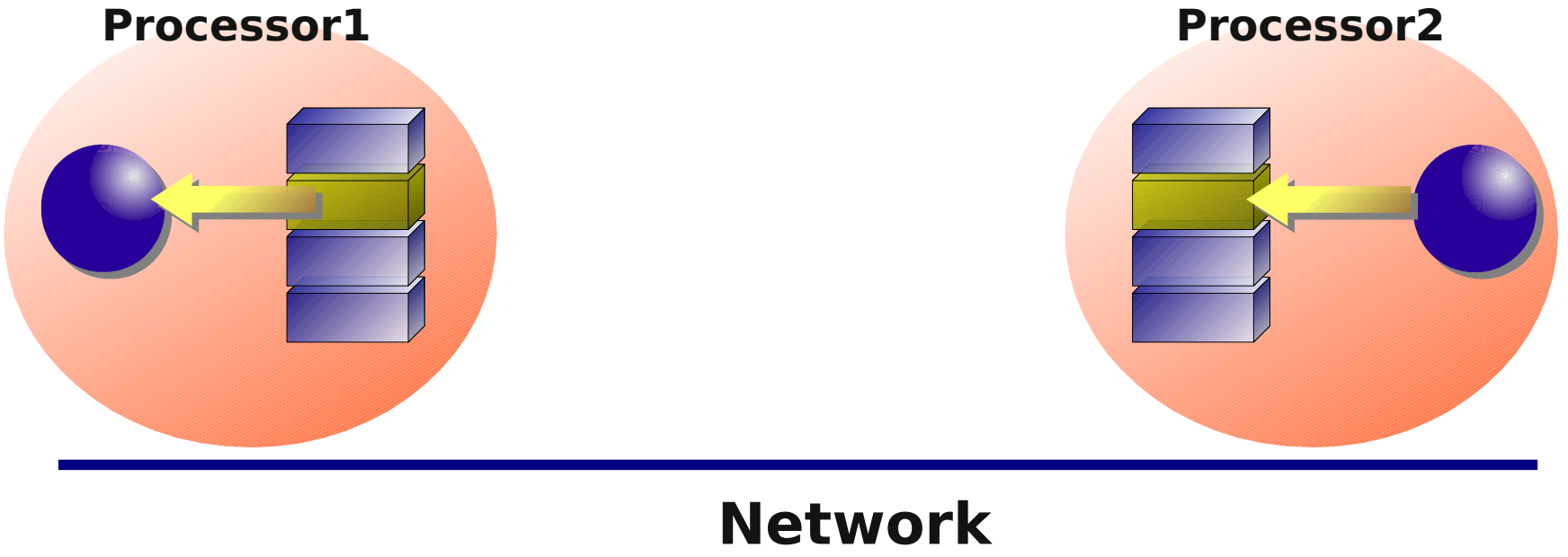
We have to take care of data consistency.

To do that we need to use semaphores, mutex, barriers, etc.

# Memory Paradigms

## Distributed Memory

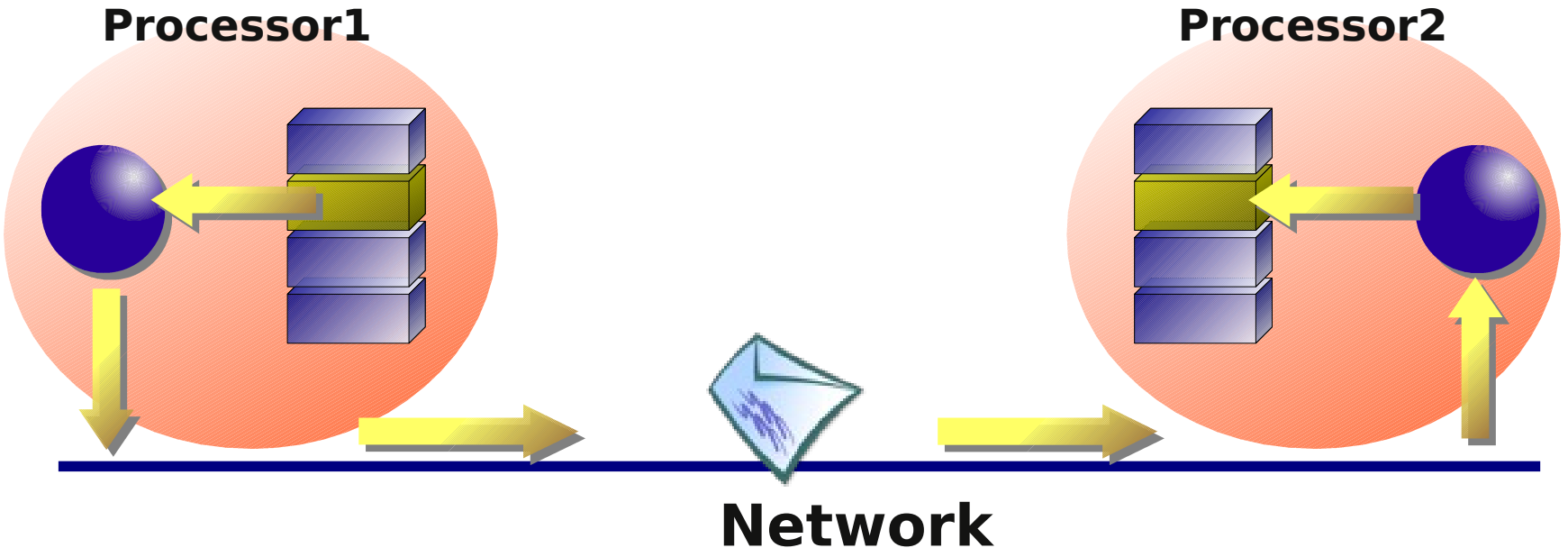
In this category each processor has its own memory.



# Memory Paradigms

## Distributed Memory

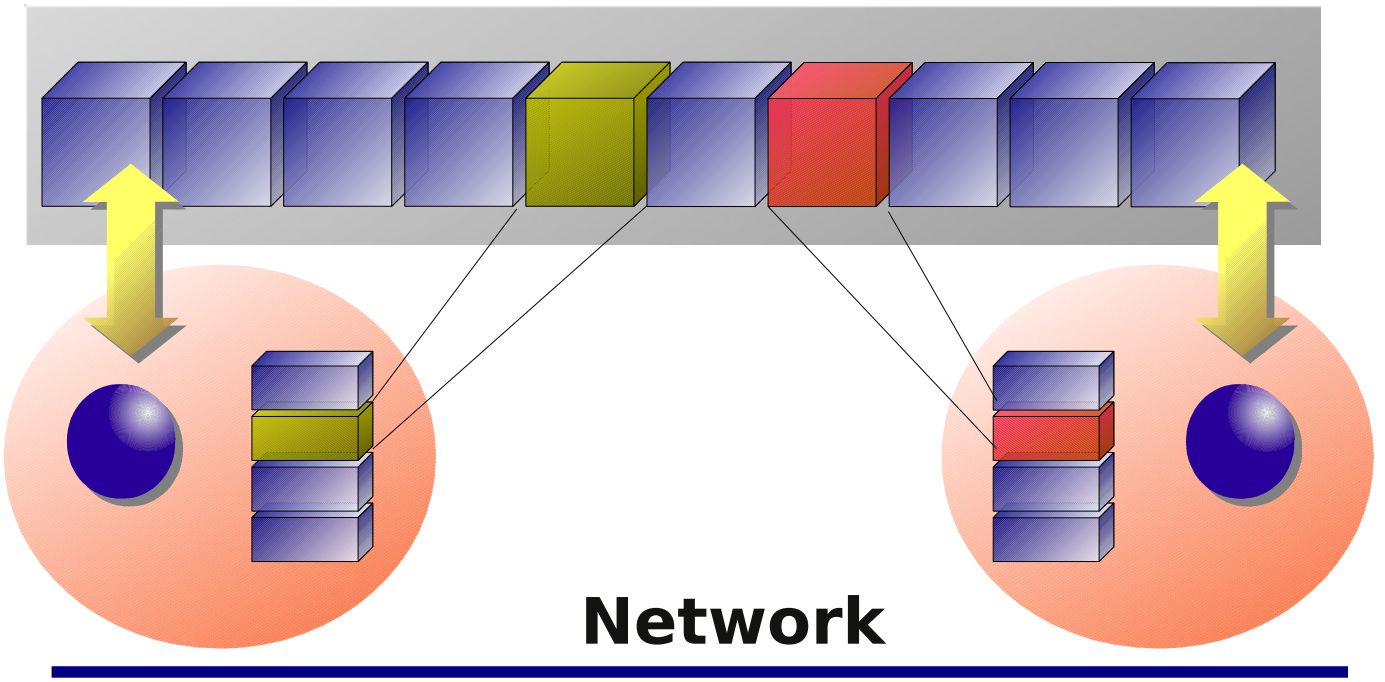
Computers request data as needed. The information is  transfered from one computer to the other using messages.



# Memory Paradigms

## Distributed Shared Memory

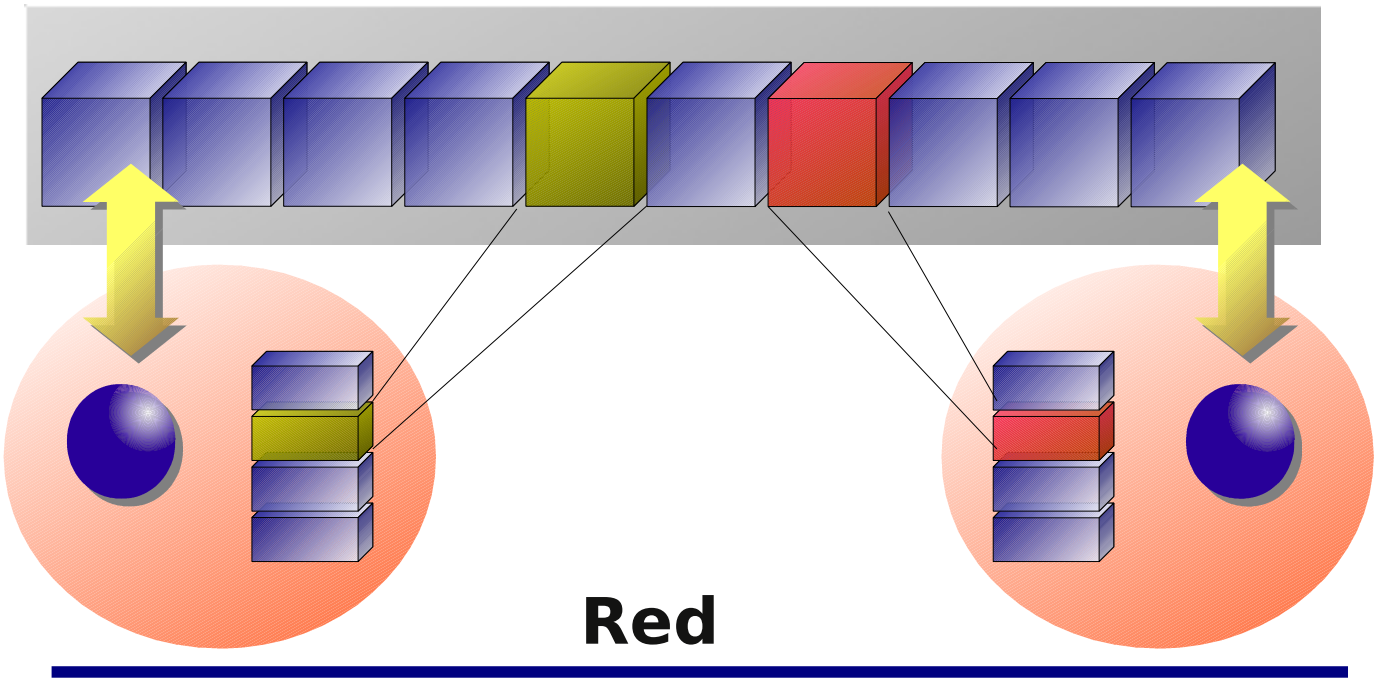
A software layer is used to simulate a virtual shared memory space. Each node provides a portion of its own memory.



# Memory Paradigms

## Distributed Shared Memory

In general, this software layer is implemented as rutines libraries: LINDA, munin, etc.




**Super Computing and Distributed Systems Camp**
**Catay, Santander, Colombia**
**August 15-22, 2010**

# Shared Memory Basics

*Gilberto Díaz*
gilberto@ula.ve

Universidad de Los Andes


# Shared Memory Paradigm

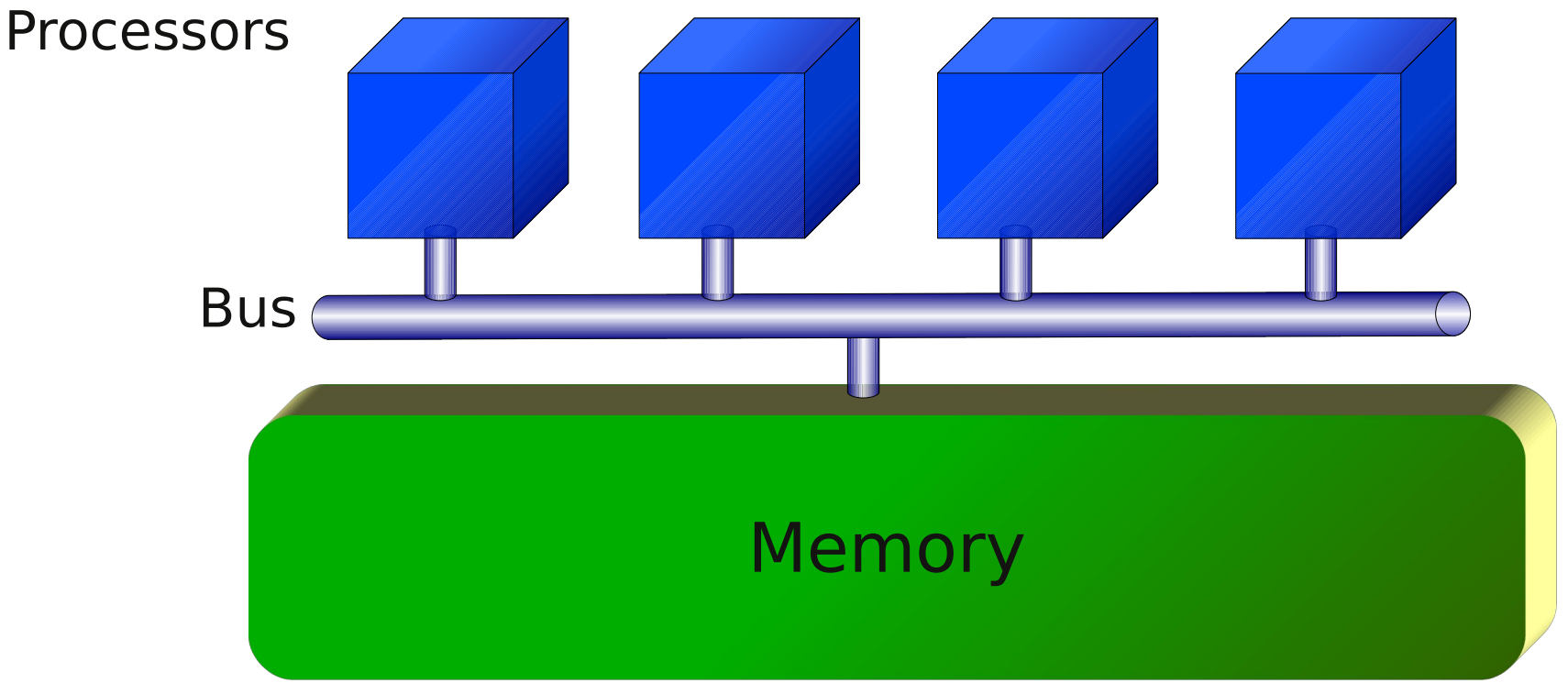
In the Shared Memory paradigm there are different categories:

- Uniform Memory Access (UMA)
- Non Uniform Memory Access (NUMA)
- Cache Coherent Non-Uniform Memory Access (ccNUMA)
- Cache Only Memory Access (COMA)

# Shared Memory Paradigm

## Uniform Memory Access (UMA)

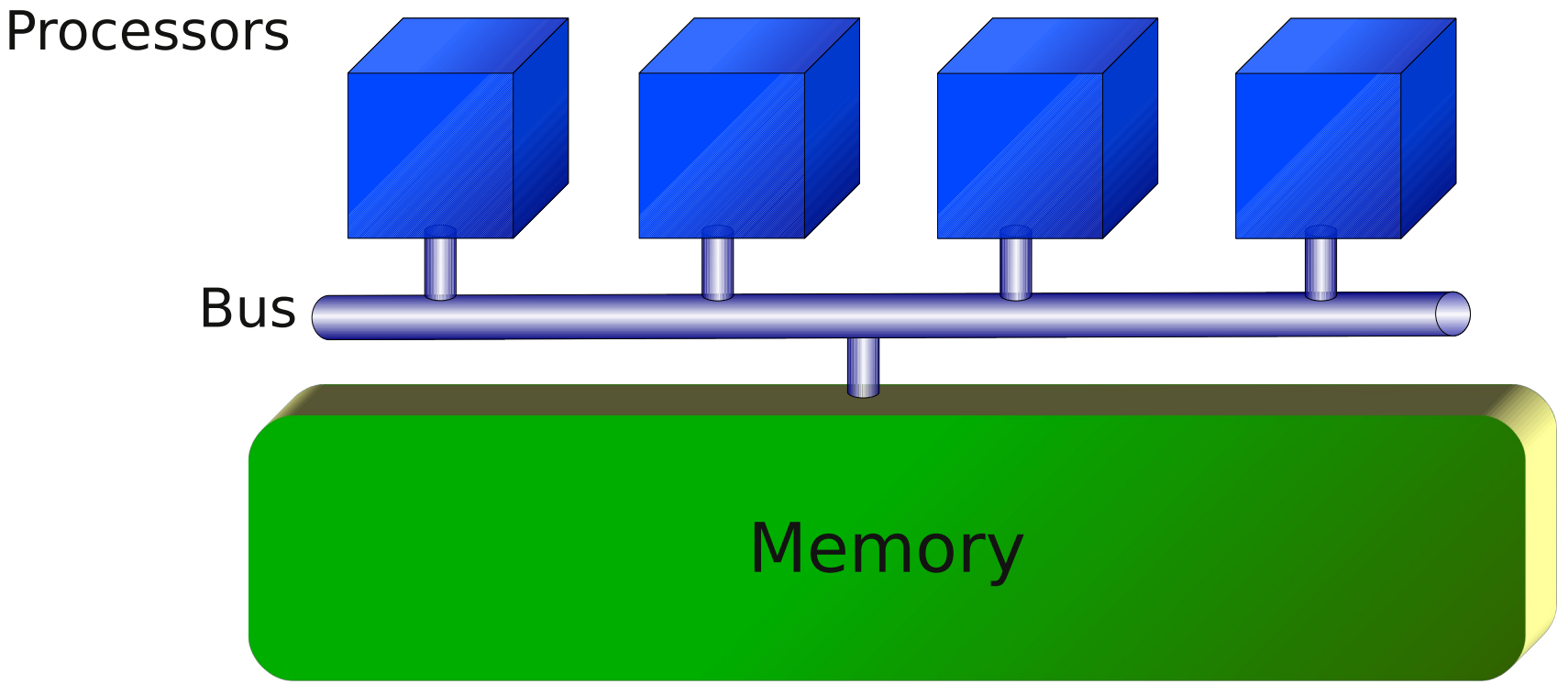
All processors have direct access to a common block of memory.



# Shared Memory Paradigm

## Uniform Memory Access (UMA)

All locations in memory are equidistant in terms of access time.



# Shared Memory Paradigm

There are some caveats:

- Bus could be a bottle neck.
- More than one processor could need the data placed in the same memory location.

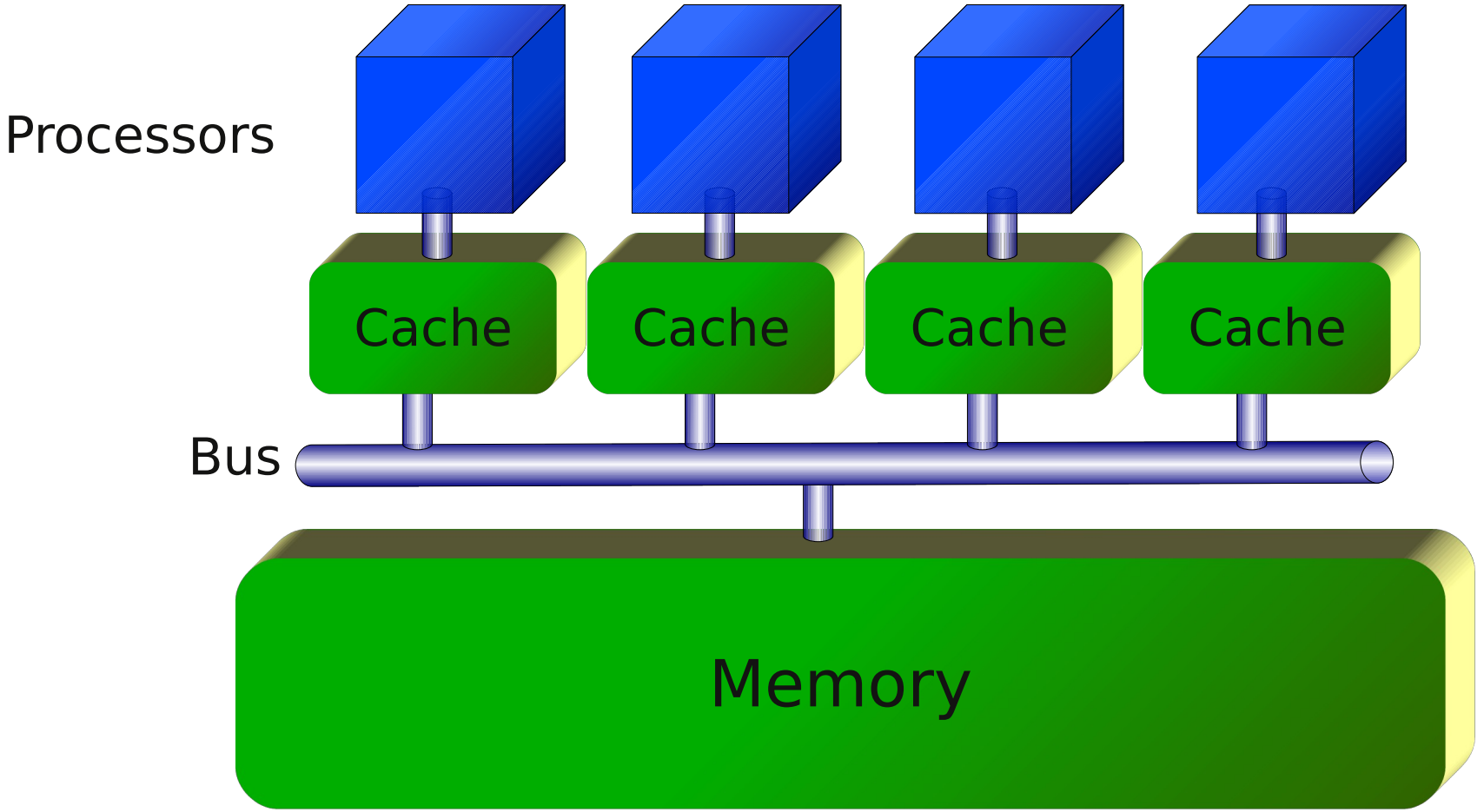
# Memoria Shared Memory Paradigm

- One of the mechanisms added by developers to keep busy the processors was faster memory placed near of them (cache memory)

# Shared Memory Paradigm

## Non Uniform Memory Access (NUMA)

Each processor has its own local memory

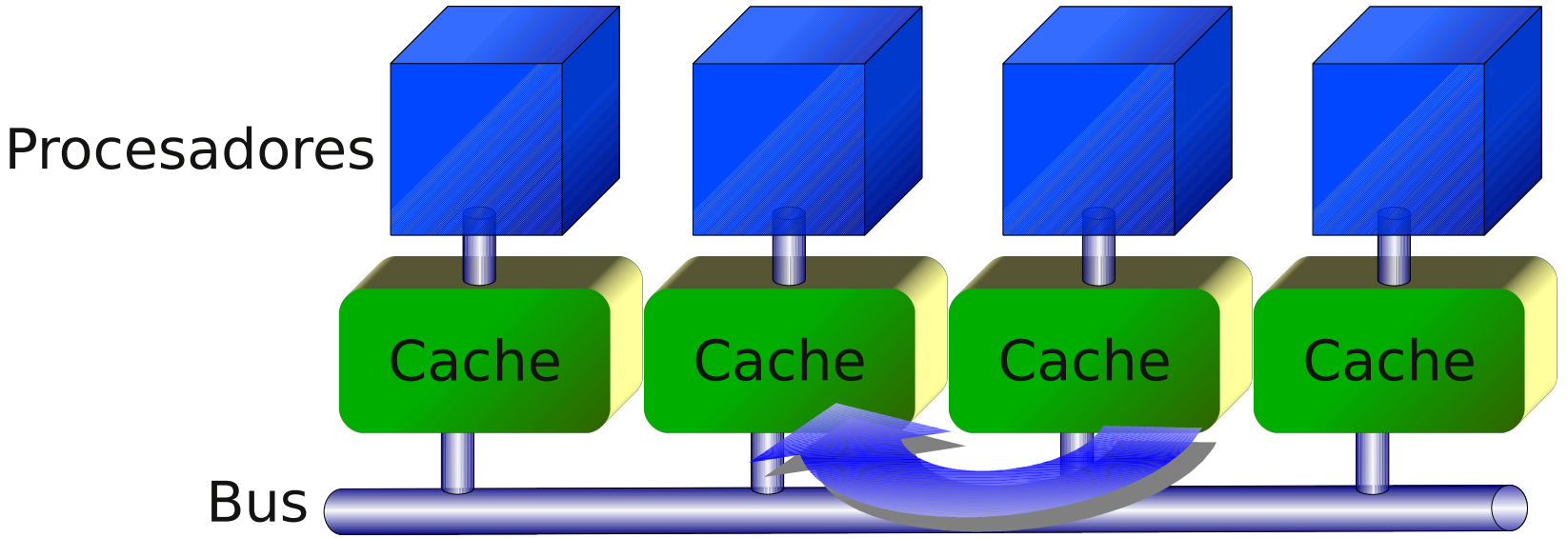


# Shared Memory Paradigm

- Some memory regions are located in different buses, therefore, terms like local memory and remote memory are used.

# Shared Memory Paradigm

- Una vez más, más de un proceso puede necesitar el mismo dato y para gestionar esto una máquina NUMA incluye hardware especializado que mueve la información desde una memoria cache a otra.



# Modelos de Memoria

- Sin embargo, esta operación puede afectar seriamente el rendimiento de una aplicación.

- Por esto, el incremento en la acelearción de la ejecución de un proceso, debido al uso de NUMA, depende en gran medida de la naturaleza de la aplicación.

# Modelos de Memoria

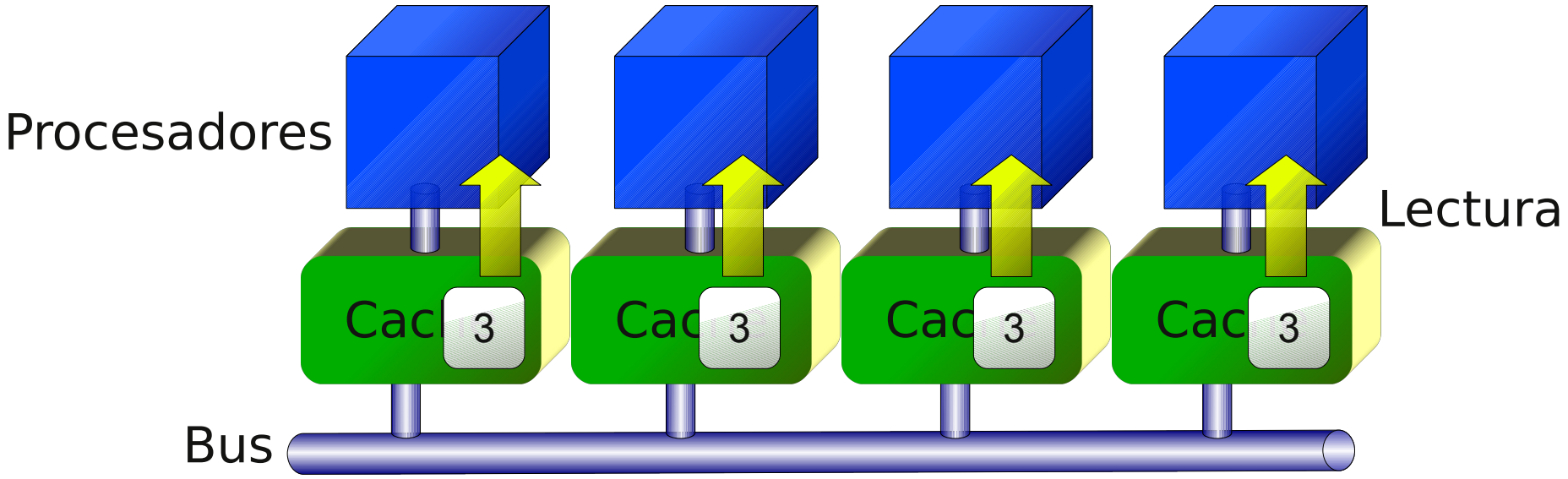
- Controlar la ubicación de los procesos o hebras a priori puede mejorar el rendimiento de la ejecución.

- Para esto el usuario debe aprovechar su conocimiento de la arquitectura de la máquina para hacer una asignación adecuada.

# Modelos de Memoria

## Cache Coherence NUMA (ccNUMA)

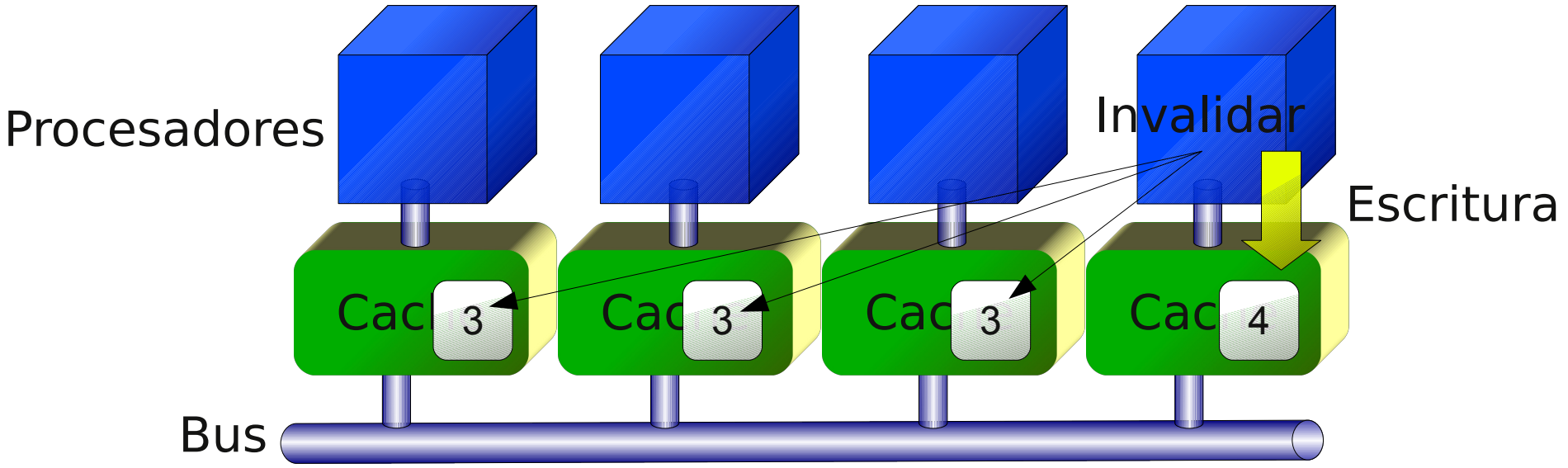
En los casos en que más de un proceso utilice un mismo dato, se debe mantener la consistencia entre las múltibples copias.



# Modelos de Memoria

## Cache Coherence NUMA (ccNUMA)

Cuando uno de los procesadores cambia una copia, se debe invalidar el resto.

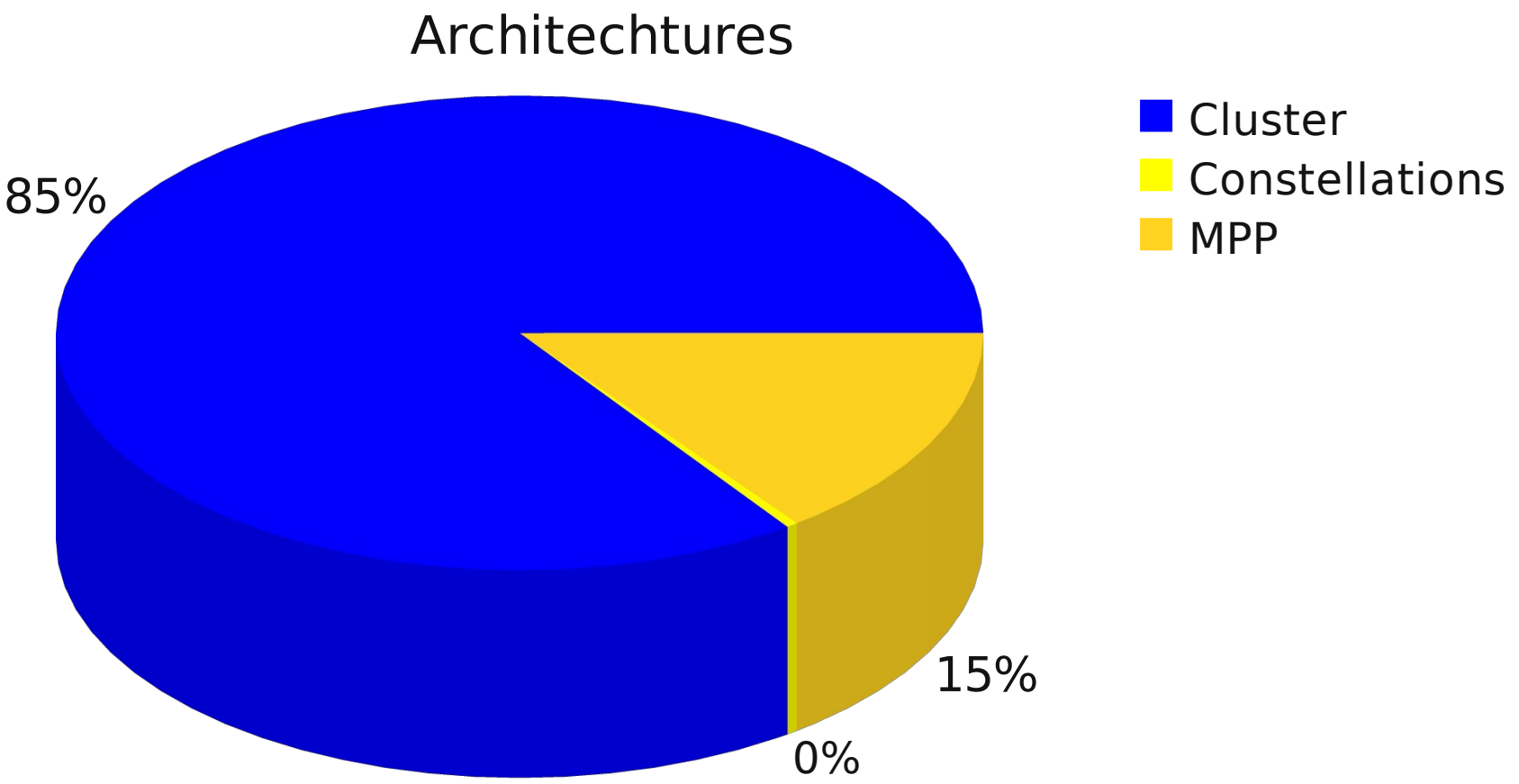


# Current Multicore Platforms

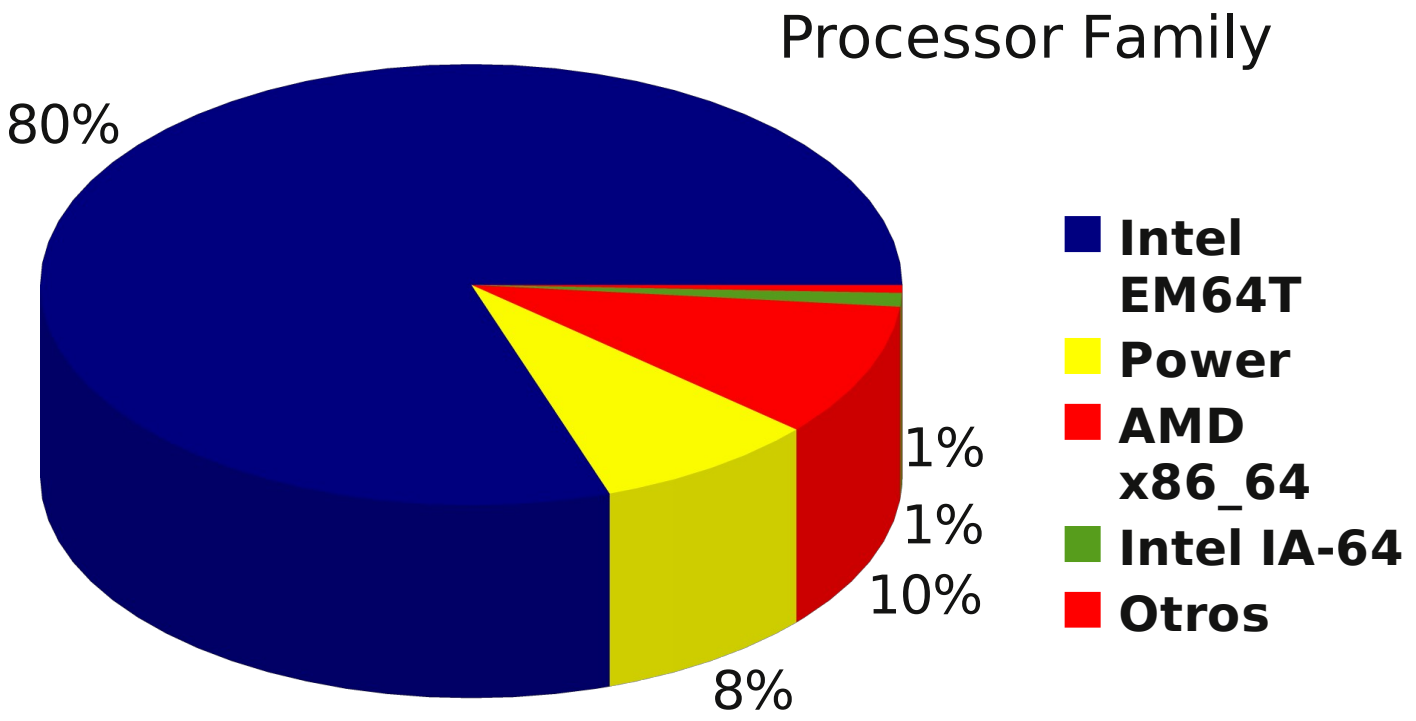
*Gilberto Díaz*
gilberto@ula.ve

Universidad de Los Andes

# top500



# top500


Processor Family
80%
1%
1%
10%
8%
Intel EM64T
Power
AMD x86_64
Intel IA-64
Otros

# top500

First place in Top500 is based on

AMD x86_64 Opteron Six Core

# top500

AMD x86_64 Opteron 12 Core:5
AMD x86_64 Opteron Dual Core:6
AMD x86_64 Opteron Quad Core:31
AMD x86_64 Opteron Six Core:7
Intel EM64T Xeon 32xx (Kentsfield):1
Intel EM64T Xeon 51xx (Woodcrest):10
Intel EM64T Xeon 52xx (Wolfdale):1
Intel EM64T Xeon 53xx (Clovertown):14
Intel EM64T Xeon 73xx (Tigerton):1

# top500

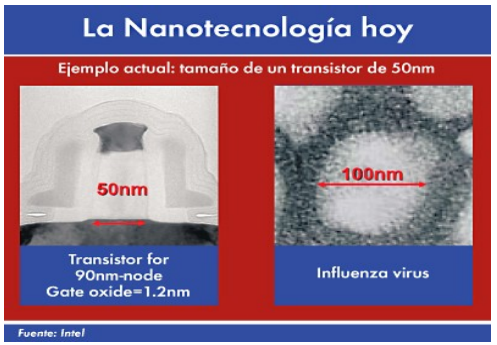
Intel EM64T Xeon 75xx (Nehalem-EX):2
Intel EM64T Xeon E54xx (Harpertown):122
Intel EM64T Xeon E55xx (Nehalem-EP):77
Intel EM64T Xeon EM64T:1
Intel EM64T Xeon L54xx (Harpertown):47
Intel EM64T Xeon L55xx (Nehalem-EP):11
Intel EM64T Xeon L56xx (Westmere-EP):1
Intel EM64T Xeon X54xx (Harpertown):11
Intel EM64T Xeon X55xx (Nehalem-EP):96
Intel EM64T Xeon X56xx (Westmere-EP):6

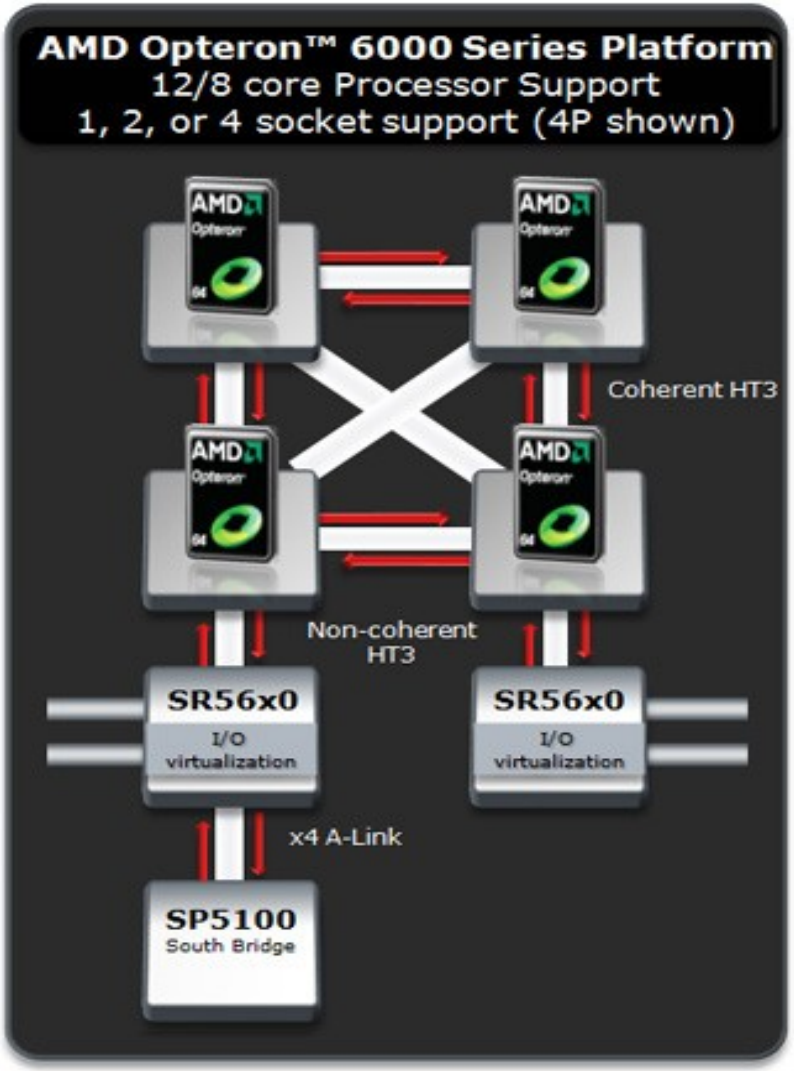
# top500

Intel IA-64 Itanium 2 Dual Core:1
Intel IA-64 Itanium2 Montecito Dual Core:4
NEC:2
POWER5:1
POWER5+:1
POWER6:18
PowerPC 440:5
PowerPC 450:10
PowerPC 970:1
PowerXCell 8i:6
SPARC64 VII:2

# AMD Opteron Processors 6000 Series

- Up to 8/12 cores 2.3 GHz.
- 45nm CMOS technology.
- 128 KB L1 cache.
- 512 KB L2 cache.
- 12 MB shared L3 cache.



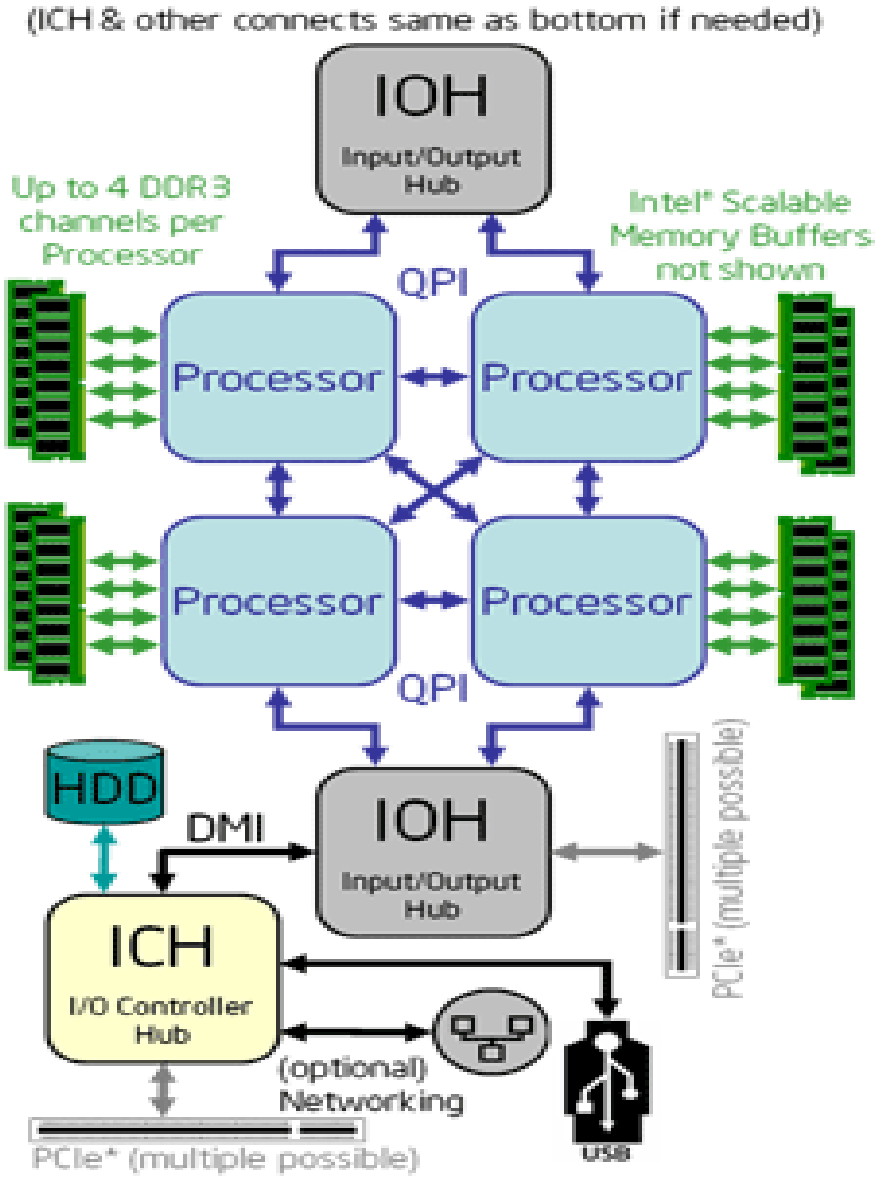



# AMD Opteron Processors 6000 Series

- Integrated Memory Controller 1800 MHz.
- Quad 16-bit HyperTransport™ 3 technology (HT3) links, up to 6.4 GT/s per link.
- AMD SR56x0 chipset with I/O Virtualization and PCIe® 2.0. Supports I/O level virtualization which provides direct control of devices by a VM
- Quad-Channel U/RDDR3 & LV U/RDDR3.

# Intel Xeon Processor X7560

- 8 cores 2.6 GHz
- 32nm technology
- 32KB L1 cache
- 256 KB L2 cache
- 24MB L3 cache



# Intel Xeon Processor X7560

- Intel Hyper-Threading Technology (16 threads)
- Intel Quick Path Interconnection 6.4 GT/s
- Intel Virtualization Technology (VT-x).