

Foundations of High Performance Computing

Lecture 2: HPC hardware&software

“High Performance Computing Module”



DATA SCIENCE &
ARTIFICIAL INTELLIGENCE



SCIENTIFIC &
DATA-INTENSIVE COMPUTING

Agenda

Why HPC is parallel ? 

Serial Computers

Moore law/Dennard Scaling

Parallel computers

HPC

=

PARALLEL
COMPUTING

L> SOFTWARE SITE

HPC

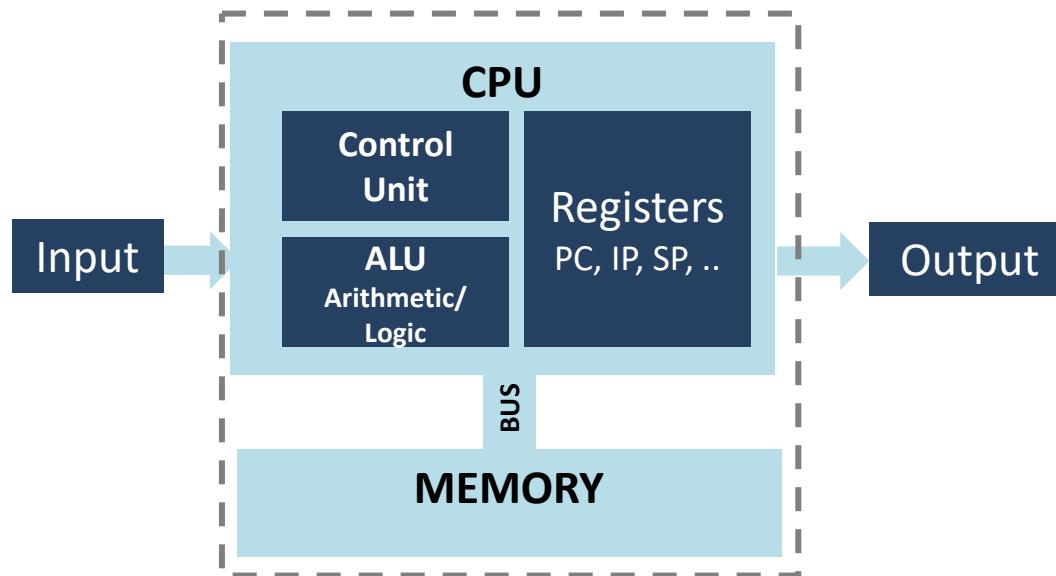
=

PARALLEL
COMPUTERS

↳ HARDWARE SITE

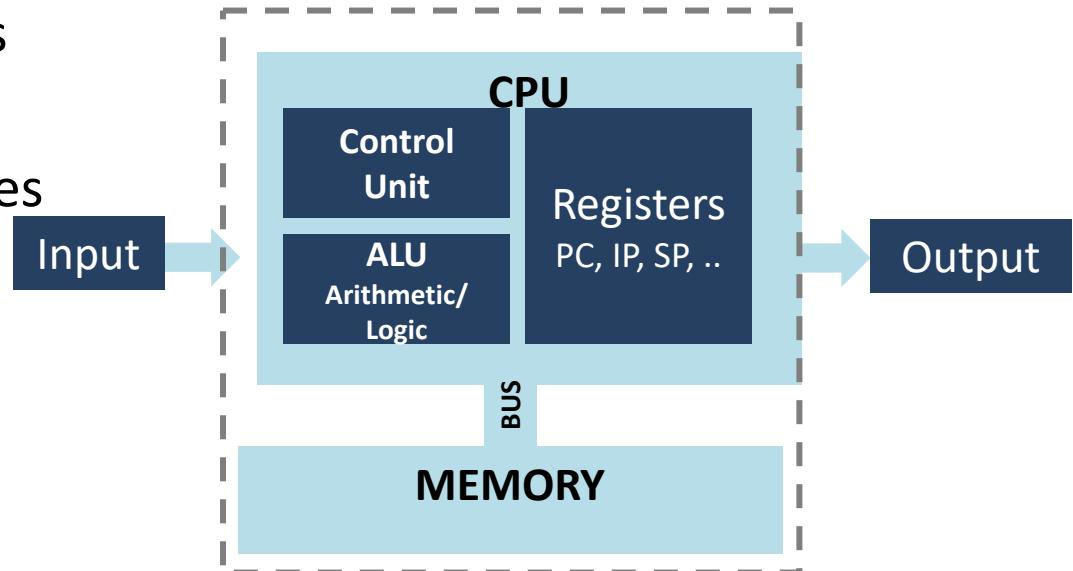
What is a serial computer ?

- Von Neumann architecture (the fundamental model)



Von Neumann architecture:

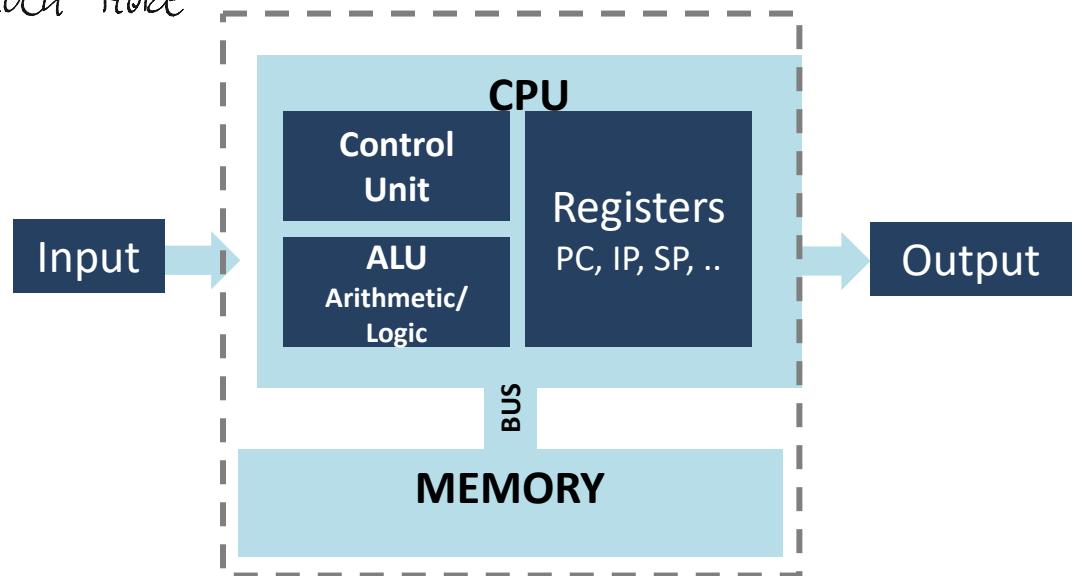
- There is only one process unit (CPU)
 - Control Unit: processes instructions
 - ALU: math and logic operations
 - Register: store data



Von Neumann architecture:

- 1 instruction is executed at a time
- memory is “flat”:
 - access on any location has always the same cost
 - access to memory has the same cost than op execution

→ NOW IS MUCH MORE



NOT ANYMORE TRUE
ON REAL MACHINE

Agenda

Why HPC is parallel ? 

Serial Computers 

Moore law/Dennard Scaling

Parallel computers

Moore Law

- Typically stated as: “Performance doubles every X months”
*JUST AN EXTRAPOLATION
OF THIS ONE ↴*
- Actually, closer to: “Number of transistors per unit cost doubles every X months”

The original Moore Law

The complexity for minimum component costs has increased at a rate of roughly a factor of two per year. [...]

Over the longer term, the rate of increase is a bit more uncertain, although there is no reason to believe it will not remain nearly constant for at least 10 years.

-- Gordon Moore,
Electronics, 1965

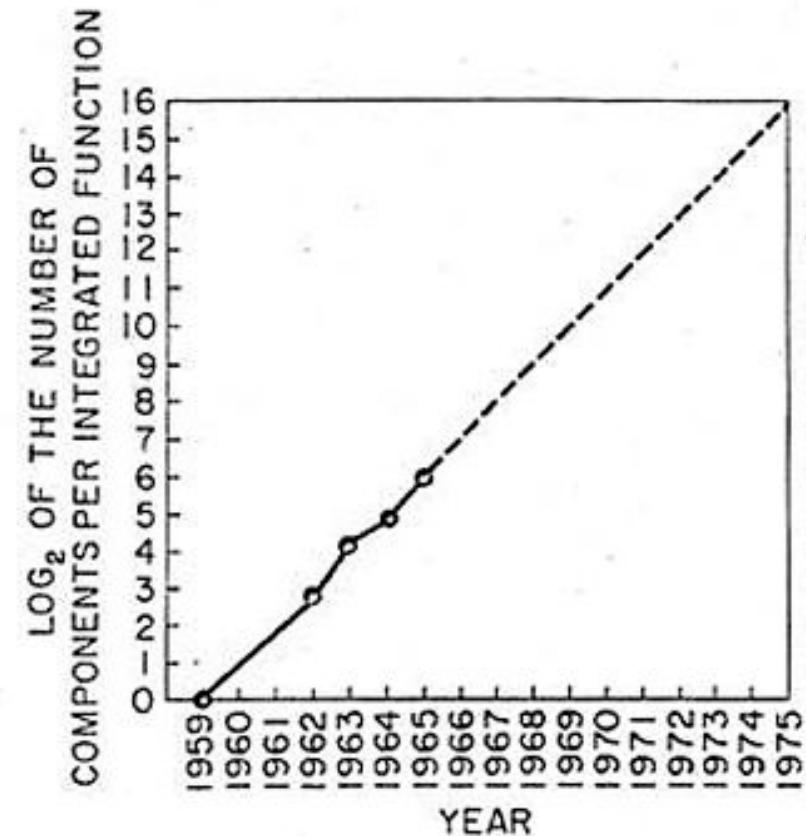


Fig. 2 Number of components per integrated function for minimum cost per component extrapolated vs time.

Why is Moore's Law connected
with processor performance?

MUCHO TRANSISTOR
MUCHO POWER

Dennard Scaling: From Moore's Law to performance

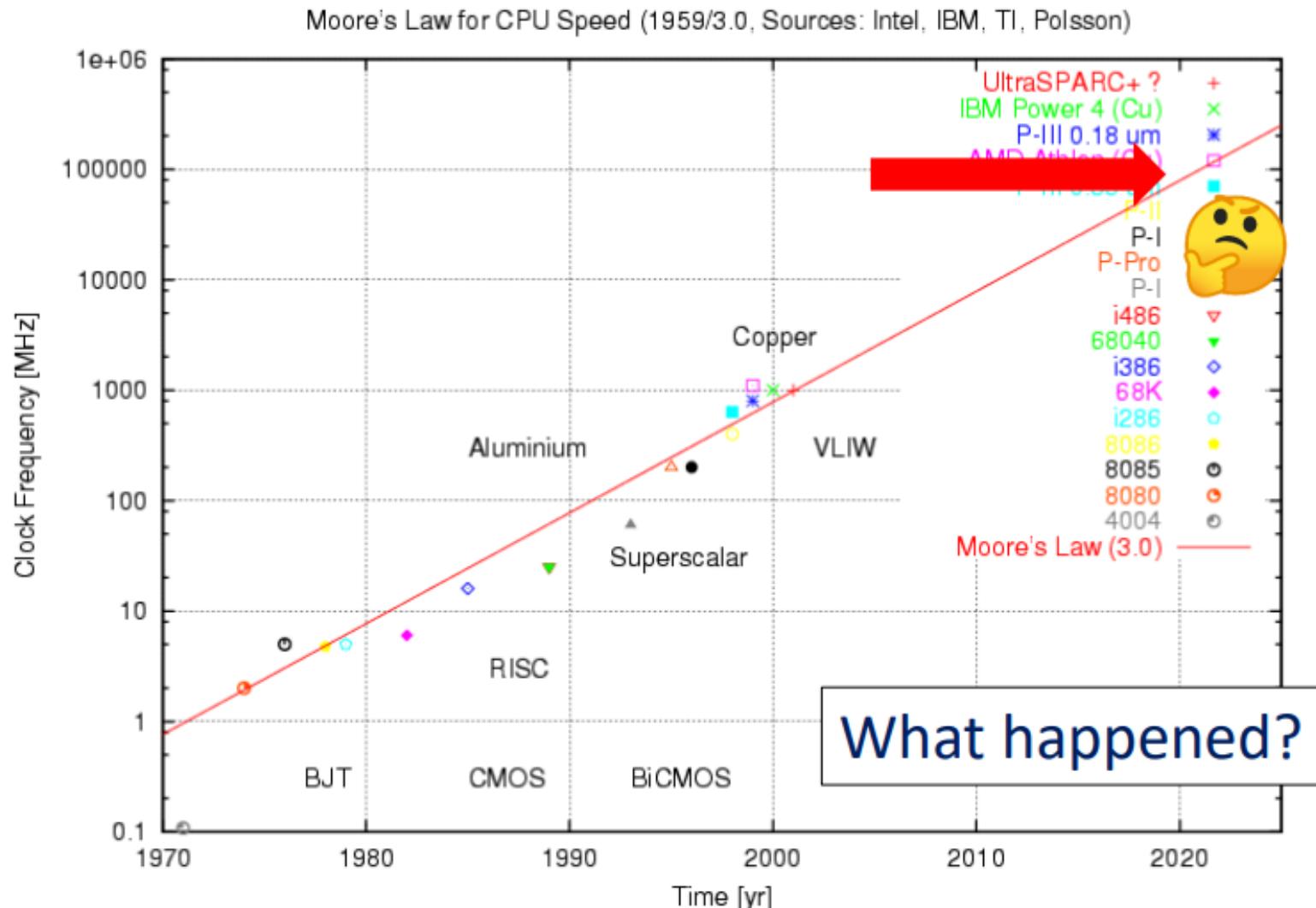
- “*Power density stays constant as transistors get smaller*”

Robert H. Dennard, 1974

- Intuitively:
 - Smaller transistors → shorter propagation delay → faster frequency
 - Smaller transistors → smaller capacitance → lower voltage
- $$Power \propto Capacitance \times Voltage^2 \times Frequency$$

Moore's law → Faster performance @ Constant power!

Single-core performance scaling



End of Dennard Scaling

- Even with smaller transistors, we cannot continue reducing power..

↳ QUANTUM EFFECTS

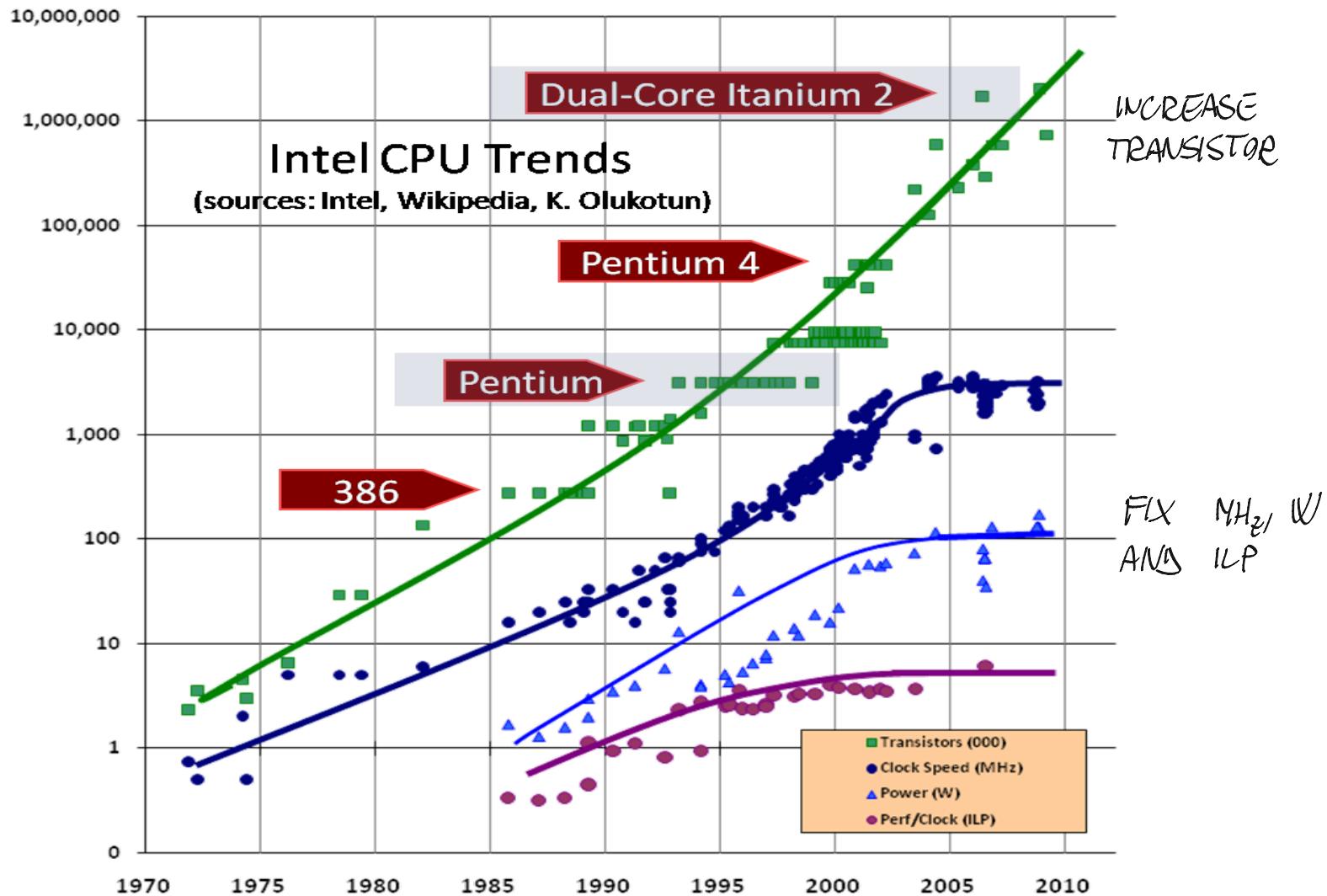
And now ?

CAN NO LONGER
DISSIPATE THE HEAT



- 2 options:
 - Increase power (when increase frequency)
 - Stop frequency scaling... (THE MHz)

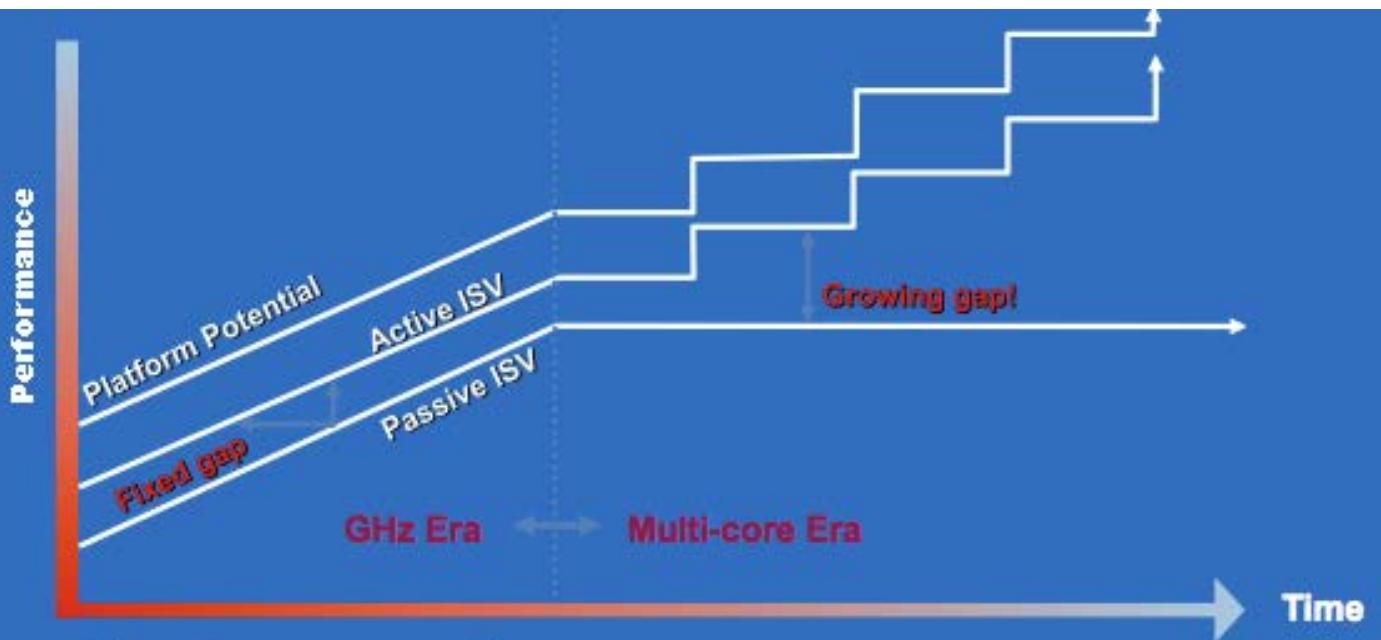
(original) Moore law still valid...



No more “free lunch” from 2006...

- Single core performance scaling ended.
 - Performance no longer depend on hardware scaling (i.e increase in frequency)
- Solution 1: the software solution
 - Write efficient software to make the efficient use of hardware resources (*ALWAYS A GOOD PRACTICE*)
 - “Performance engineering” software, using hardware knowledge

An old picture from Intel..



"Parallelism for Everyone"

Parallelism changes the game

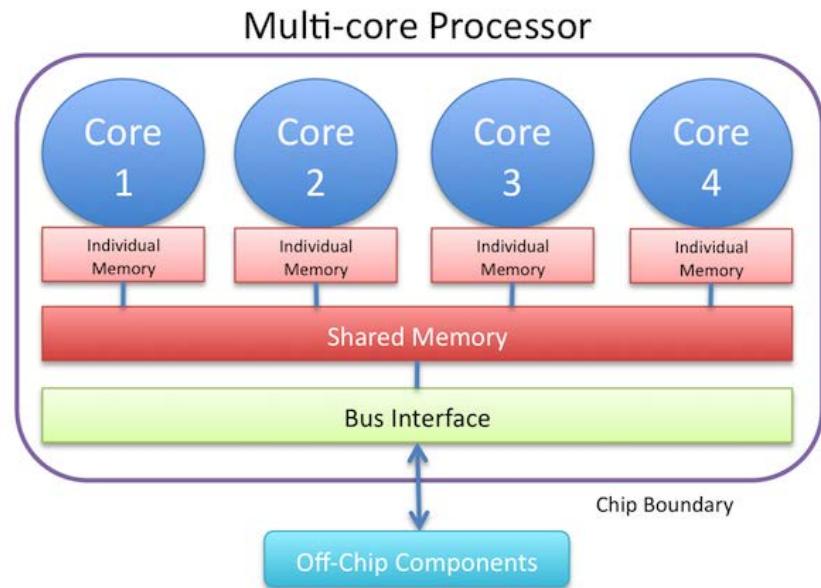
- A large percentage of people who provide applications are going to have to care about parallelism in order to match the capabilities of their competitors.

No more “free lunch” from 2006...

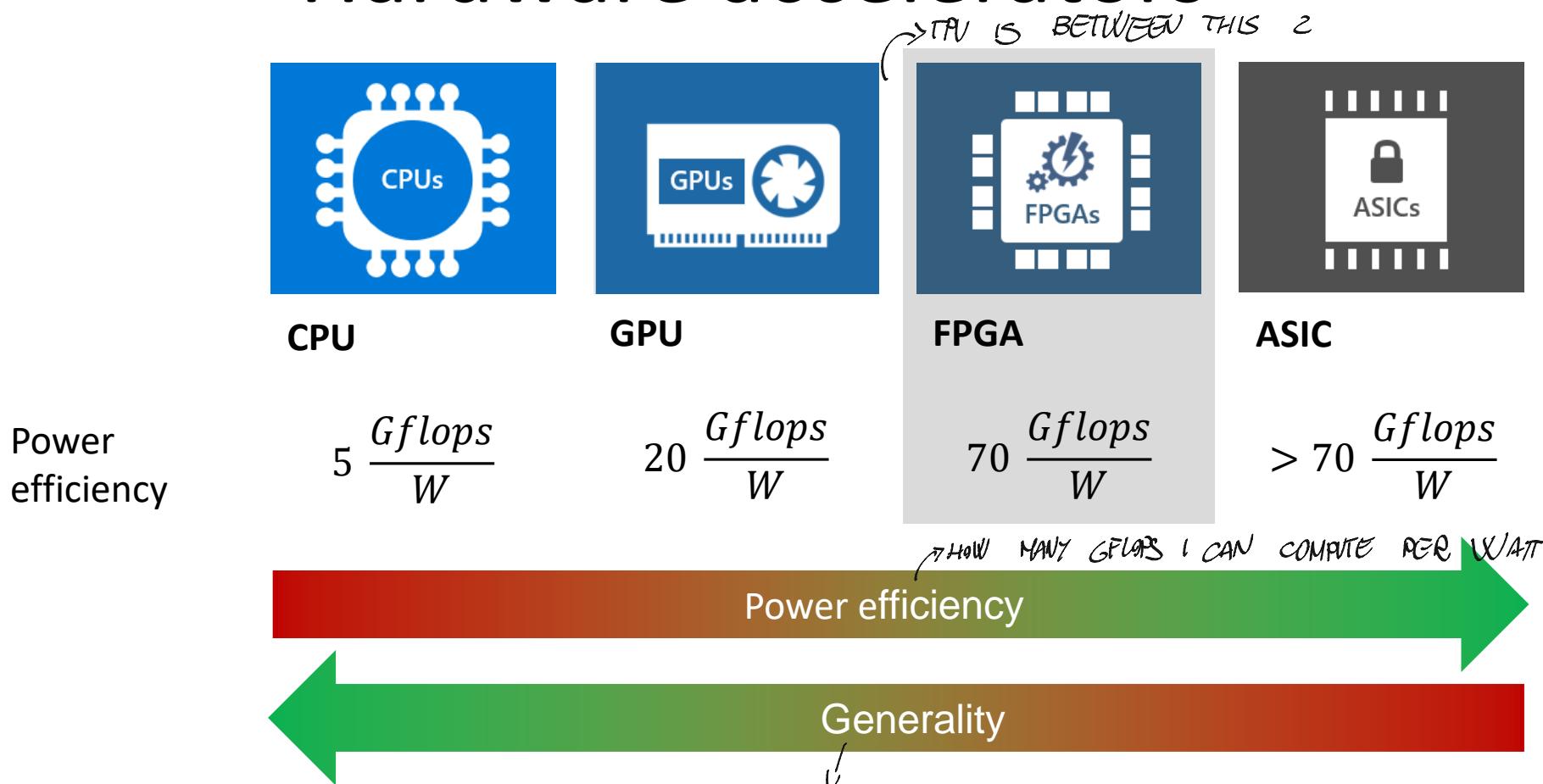
- Solution 2: specialized architectural solution
 - Chip space is now cheap, but power is expensive
 - Stop depending on more complex general-purpose cores
 - Use space to build heterogeneous systems, with compute engines well-suited for each application

CPU are multicore processor

- Because of power, heat dissipation, tendency is to actually lower clock frequency but pack more computing cores onto a chip.
- These cores will share some resources, e.g. memory, network, disk, etc but are **still capable** of independent calculations



Hardware accelerators



Images: <https://www.microsoft.com/en-us/research/video/inside-microsoft-fpga-based-configurable-cloud/>

Numbers: https://h2rc.cse.sc.edu/2015/burger_keynote.pdf

Agenda

Why HPC is parallel ? 

Serial Computers 

Moore law/Dennard Scaling 

Parallel computers

PARALLELISM IS
EVERYWHERE

even in your
laptop..

Parallel Computers

- Flynn Taxonomy (1966): may help us in classifying them:
 - Data Stream
 - Instruction Stream

		Instruction stream	
		Single	Multiple
Data stream	Single	SISD	MISD
	Multiple	SIMD	MIMD

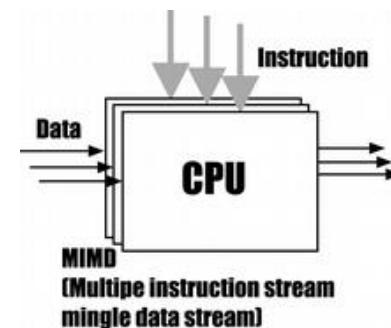
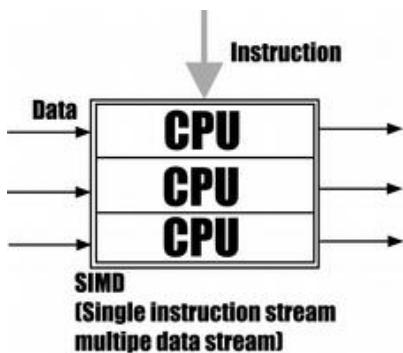
Comments

- Flynn taxonomy does not help too much nowadays with modern HPC infrastructure
 - CPU and computers are changed too much in the last 60 years
- However, SIMD and MIMD concepts are still used HPC hardware

SINGLE INSTRUCTION
/ MULTIPLE DATA

MULTIPLE INSTRUCTION
/ MULTIPLE DATA

AVX LIBRARY

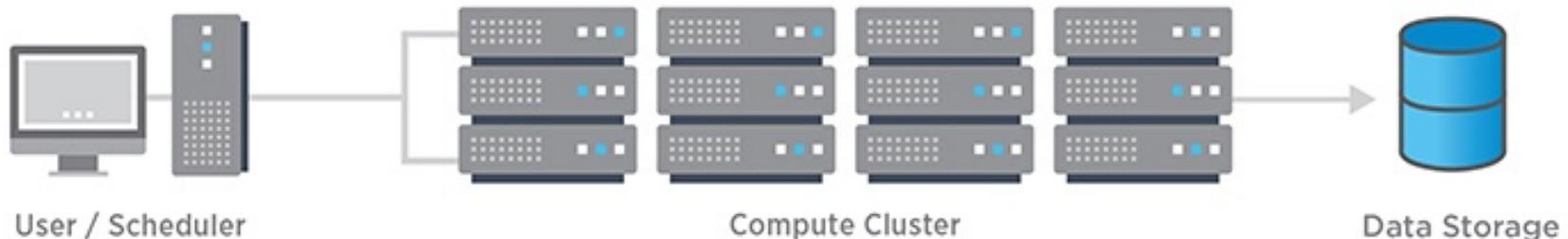


Comments (2)

	HW level	SW level
SISD	A Von Neumann CPU	no parallelism at all
MISD	On a superscalar CPU, different ports executing different <i>read</i> on the same data	<ul style="list-style-type: none">• ILP on same data;• Multiple tasks or threads operating on the same data
SIMD	Any vector-capable hardware, the vector registers on a core, a GPU, a vector processor, an FPGA, ...	data parallelism through vector instructions and operations
MIMD	Every multi-core/processor system; on a superscalar CPUs, different ports executing different ops on different data	<ul style="list-style-type: none">• ILP on different data;• Multiple tasks or threads executing different code on different data.

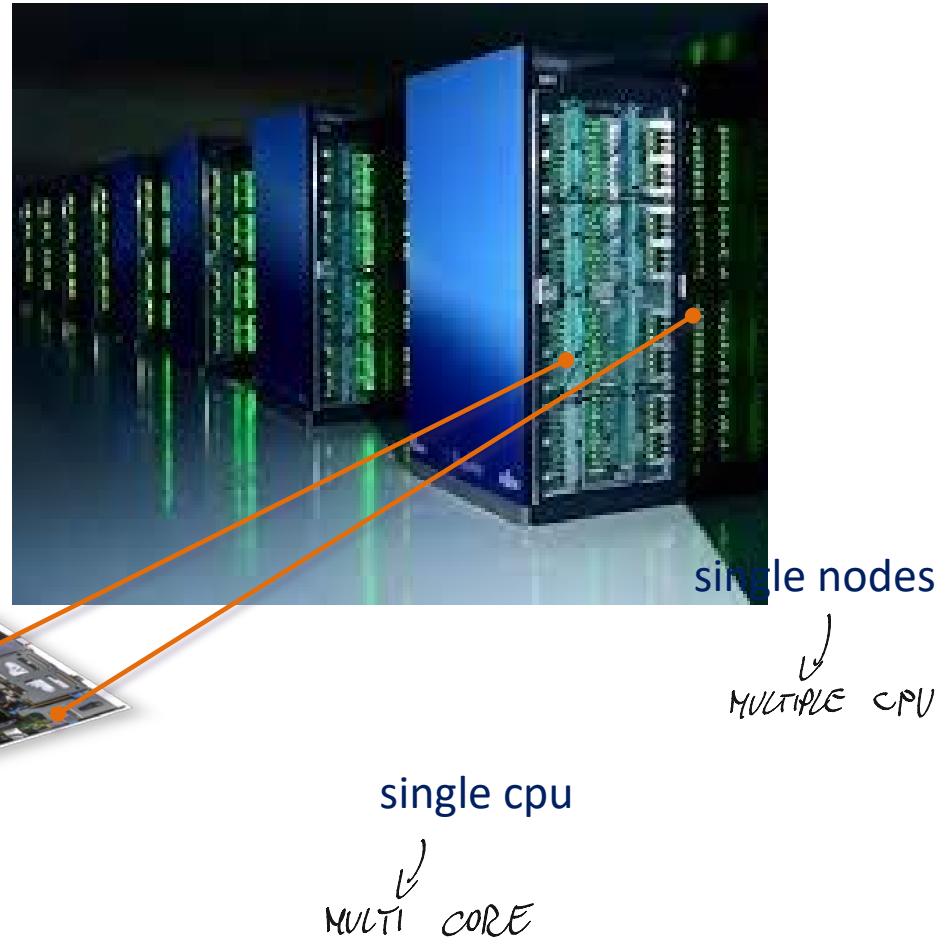
Essential component of a HPC cluster

- Several computers (nodes)
 - often in special cases (1U) for easy mounting in a rack
- One or more networks (interconnects) to hook the nodes together
- Some kind of storage
- A login/access node..



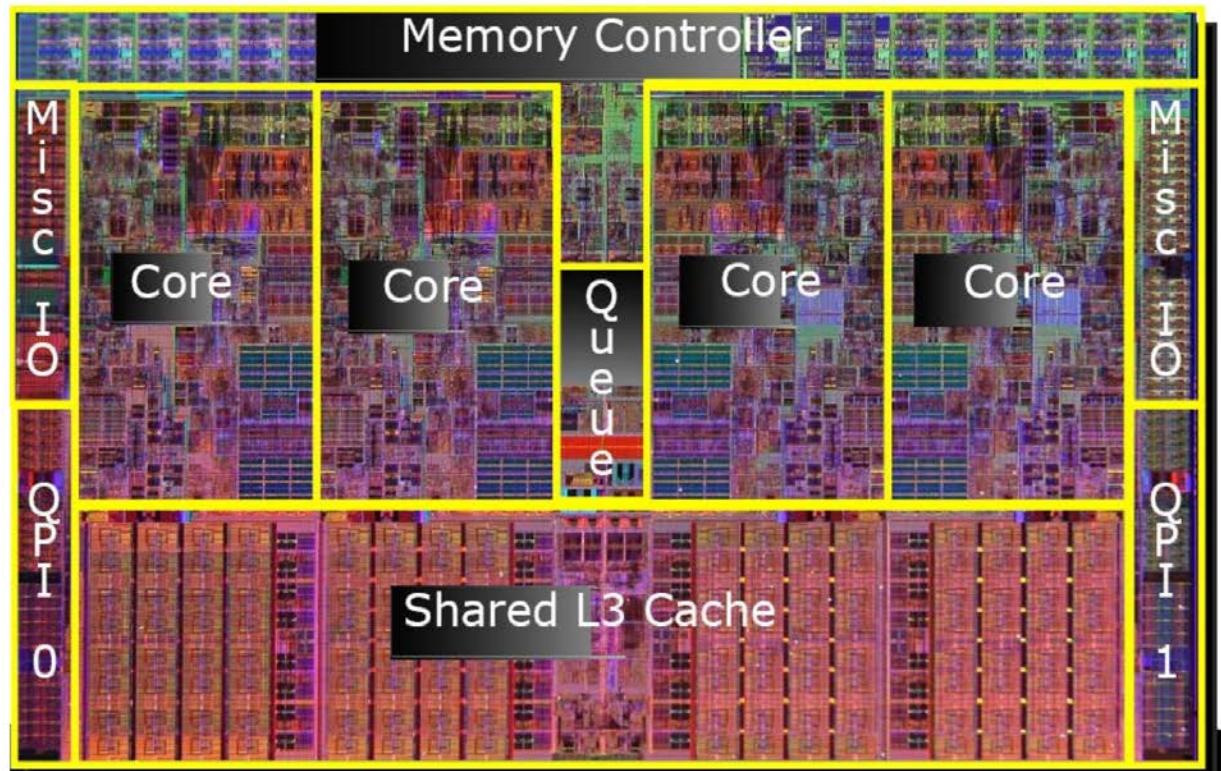
The hardware behind HPC

interconnected
racks of connected
nodes



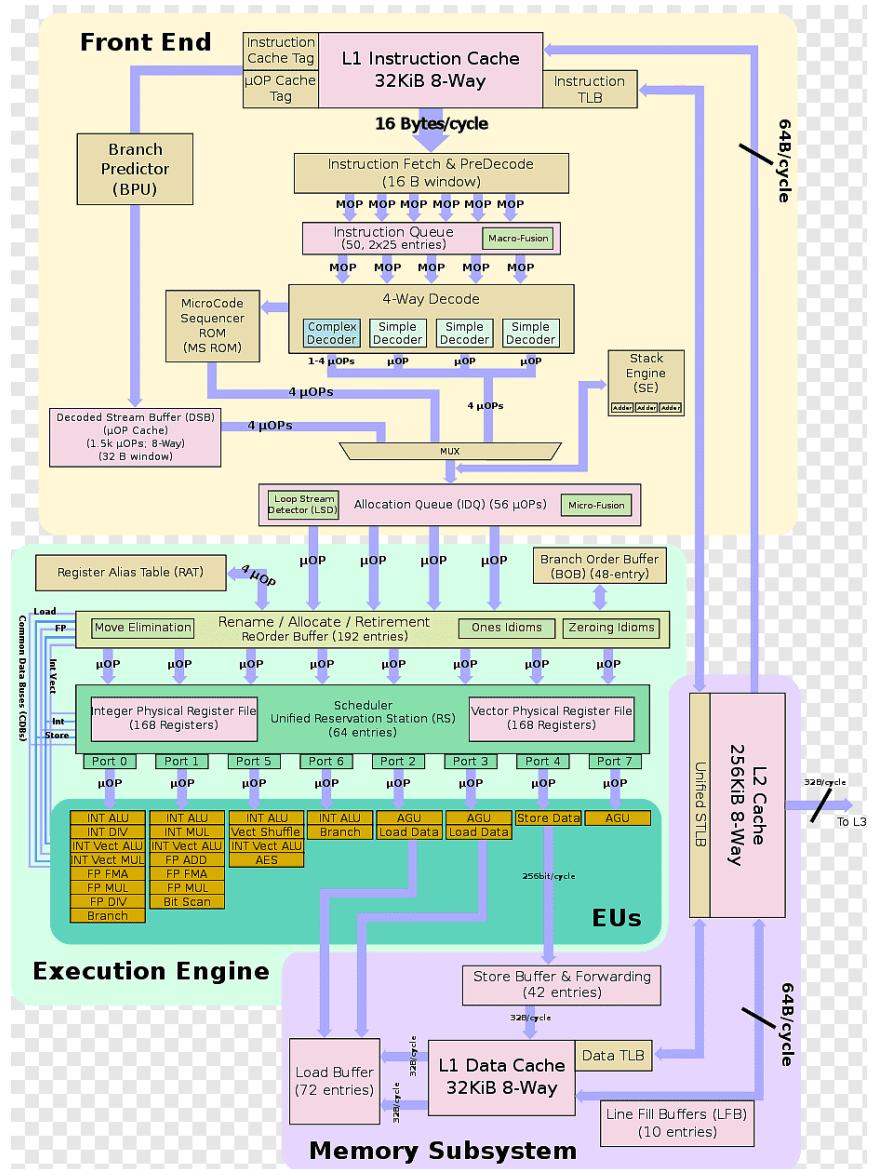
Single CPU topology

**Modern
CPUs**
are multi- (or
many-) cores



Core : definition

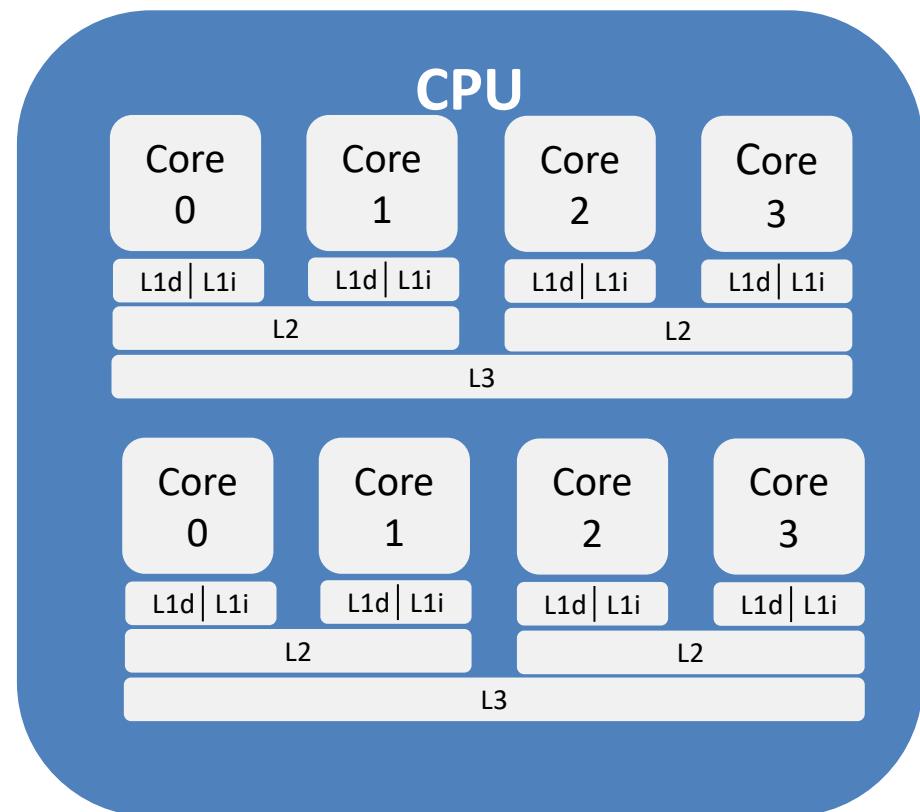
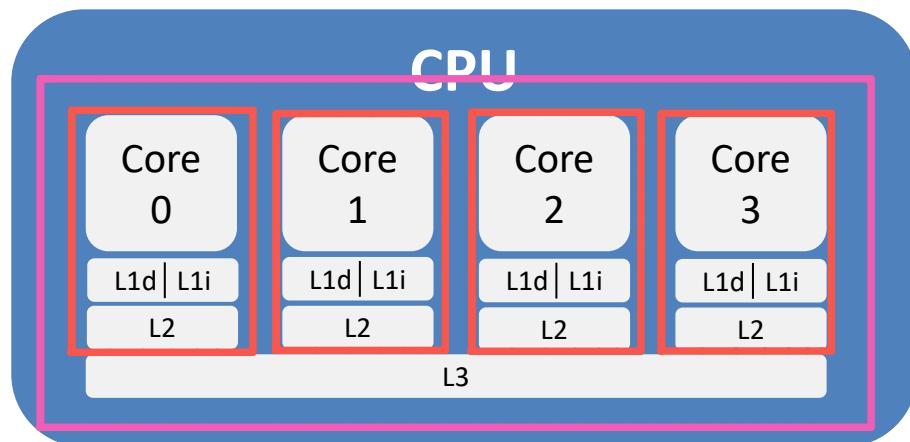
- A core is the smallest unit of computing, having one or more (hardware/software) threads and is responsible for executing instructions.



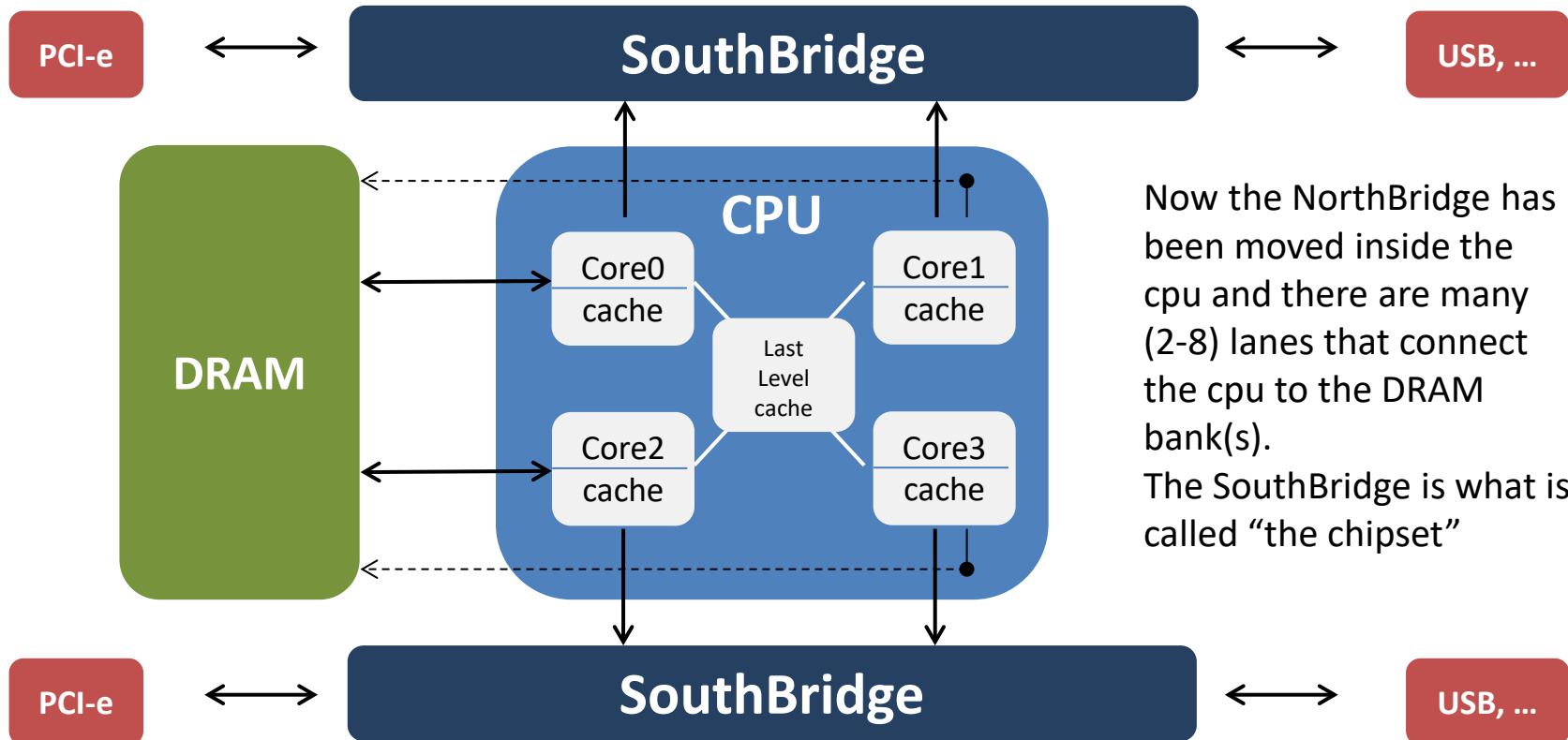
Single CPU topology

- Cache hierarchy can have different topologies

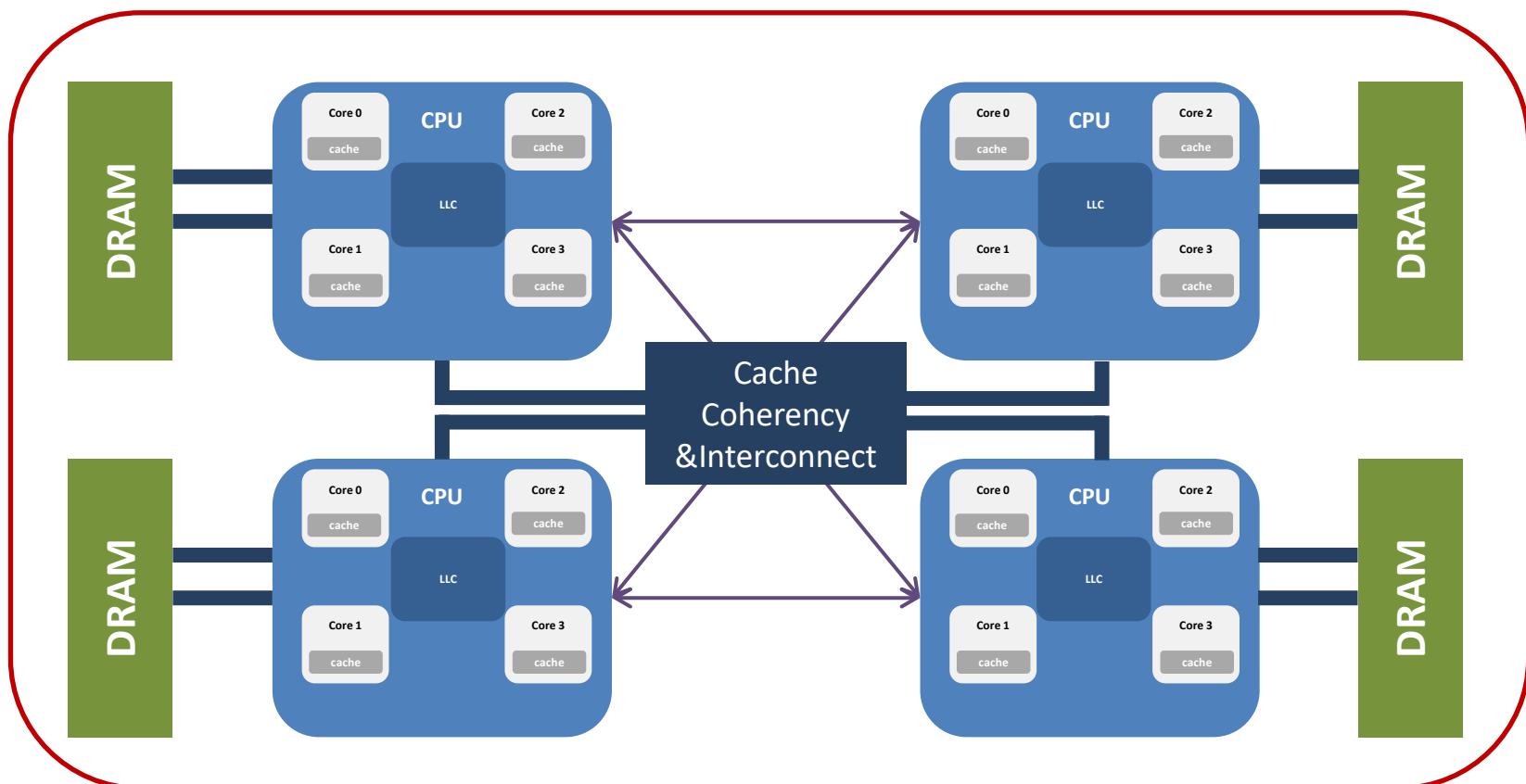
$L1d \rightarrow DATA$
 $L1i \rightarrow INSTRUCTION$



Modern CPU layout



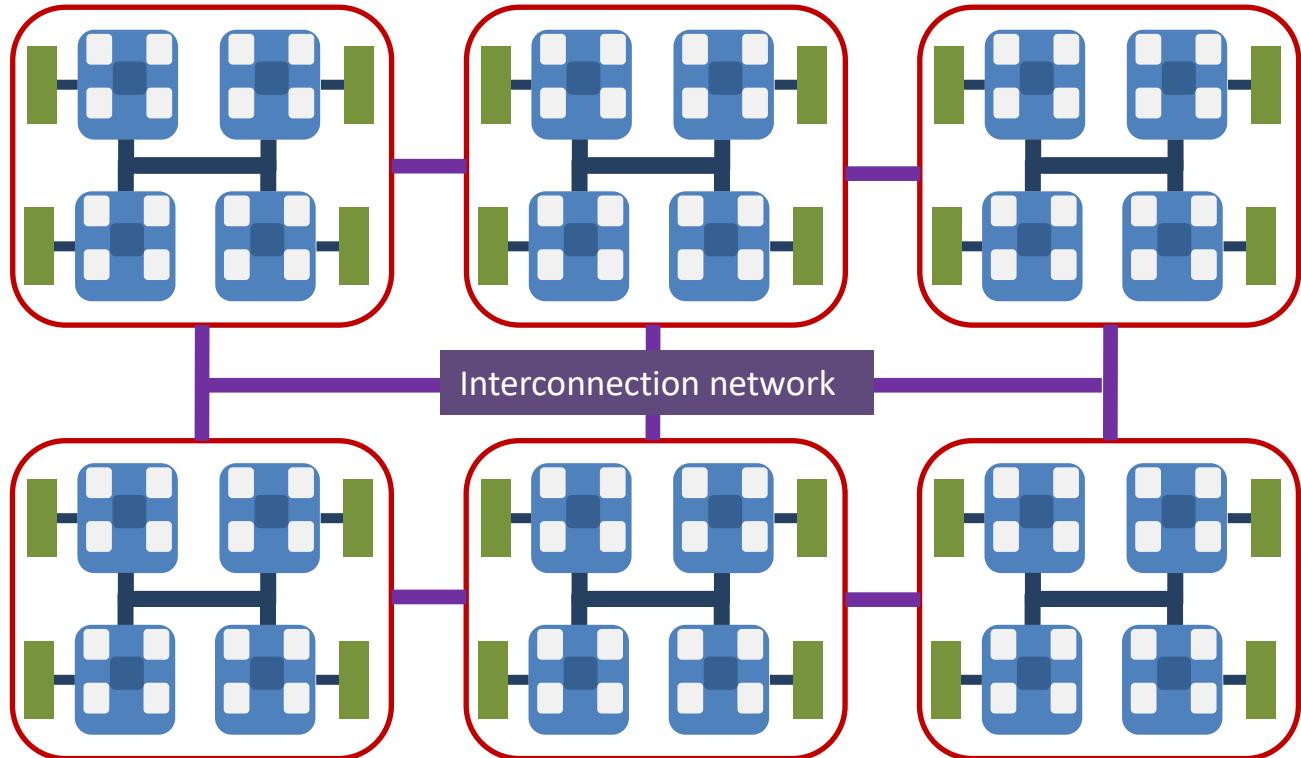
Node topology



The overall topology

CLUSTER OF COMPUTING NODES

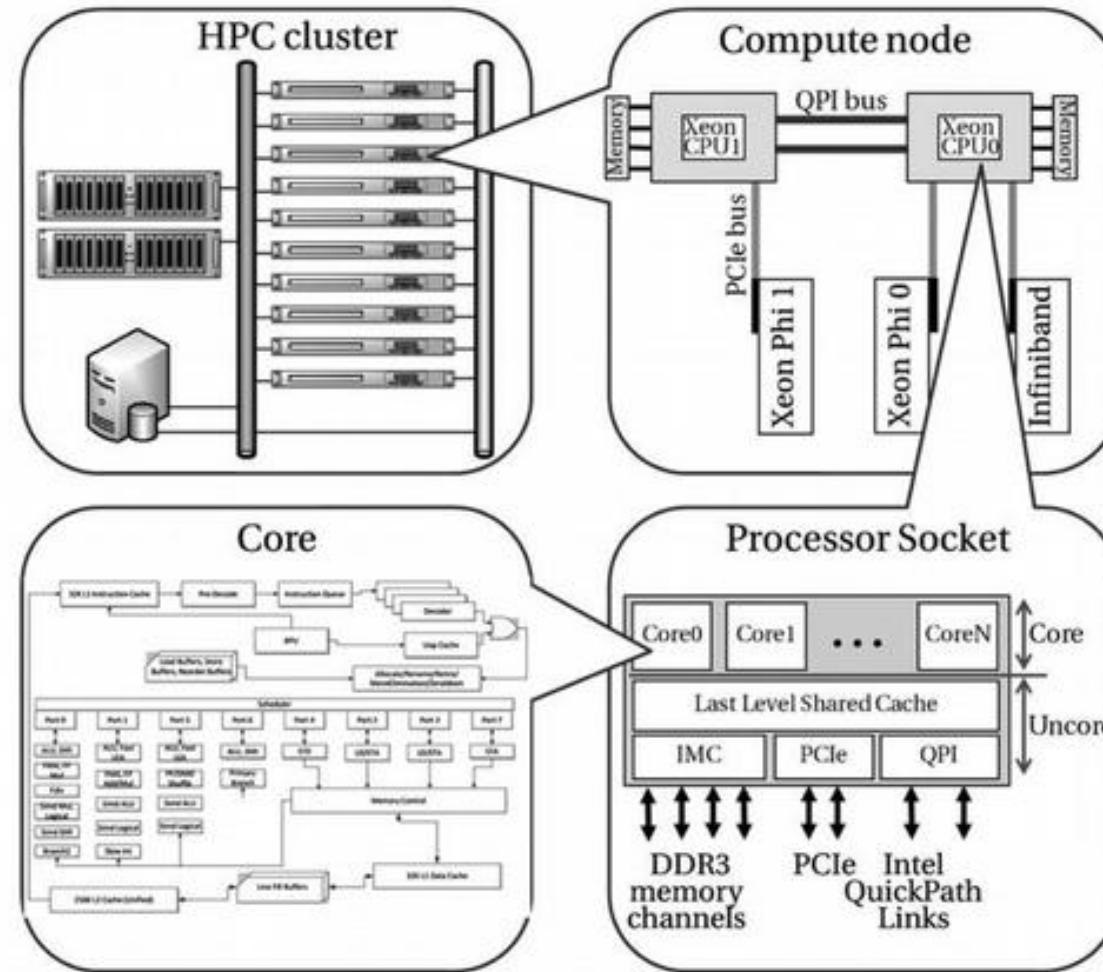
Note: there are many different topologies for the interconnection network.



Network cluster classification

- HIGH SPEED NETWORK
 - parallel computation
 - low latency /high bandwidth
 - Usual choices: Infiniband...
- I/O NETWORK
 - I/O requests (NFS and/or parallel FS)
 - latency not fundamental/ good bandwidth
 - GIGABIT could be ok /10Gb and/or Infiniband better
- In band Management network
 - management traffic of all services (LRMS/NFS/software etc..)
- Out of band Management network:
 - Remote control of nodes and any other device

The building blocks of a HPC infrastructure (cluster)

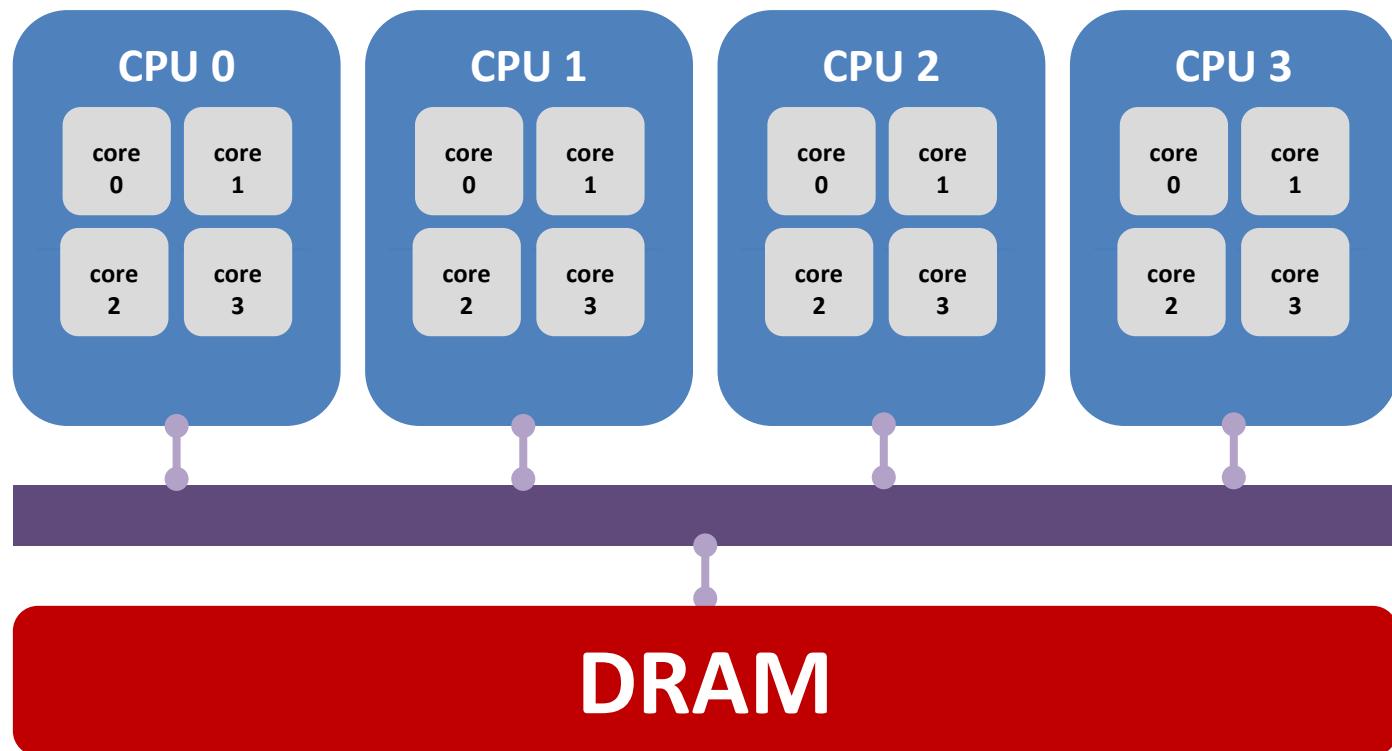


What about memory ?

- Note that on a supercomputer there is a hybrid approach as for the memory placement:
- the memory on a single nodes can be accessed directly by all the cores on that node, meaning that memory access is a “read/write” instructions irrespectively of what exact memory bank it refers to.
This is called **shared-memory**.
- when you use many nodes at a time, a process can not directly access the memory on a different node. It need to issue a request for that, not a read/write instruction.
That is named **distributed memory**.
- These are hardware concepts, i.e. they describe how the memory is physically accessible. However, they do also refer to programming paradigms, as we'll see in a while.

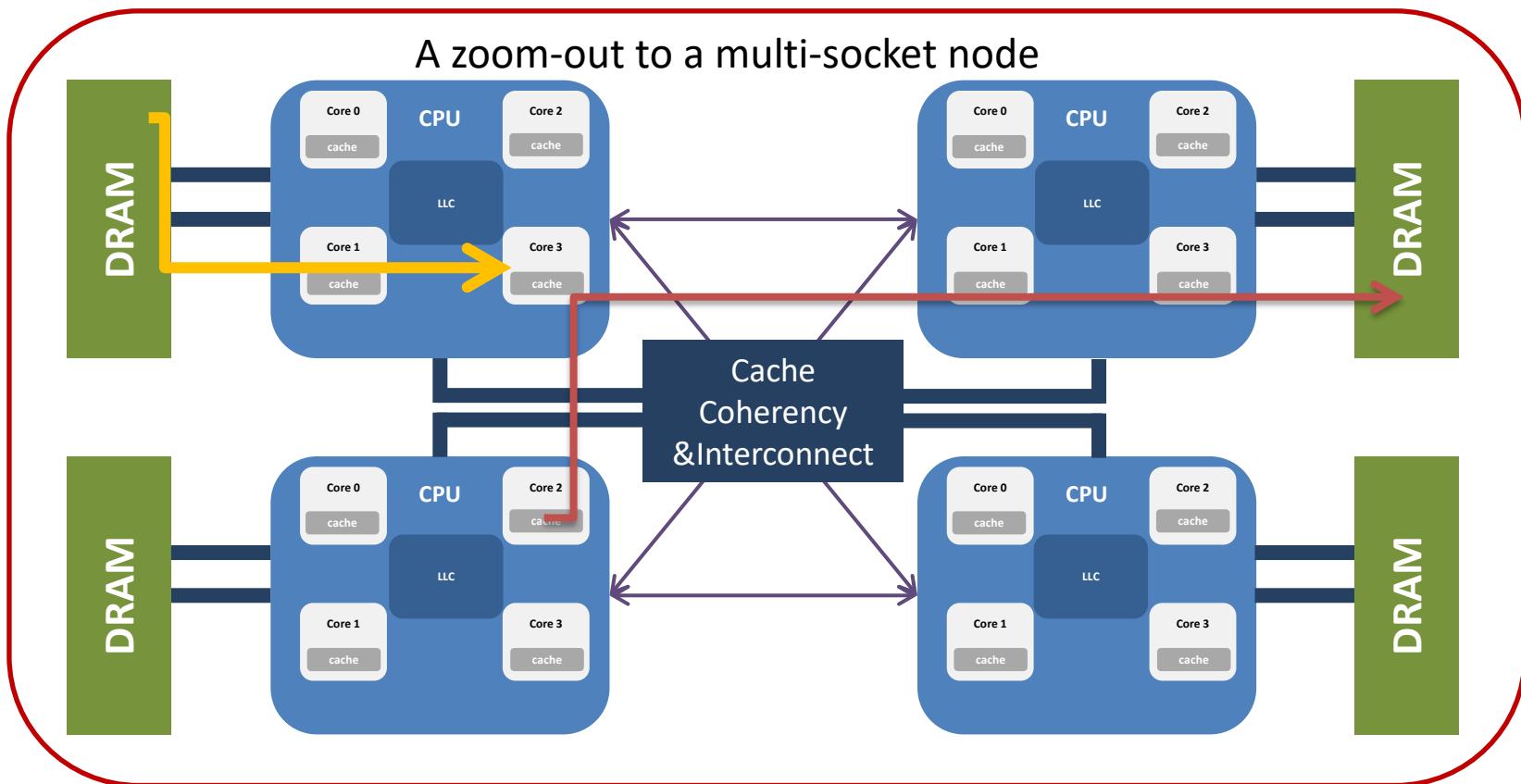
Shared memory: UMA

Uniform memory access (UMA): Each processor has uniform access to memory. Also known as symmetric multiprocessors (**SMP**)

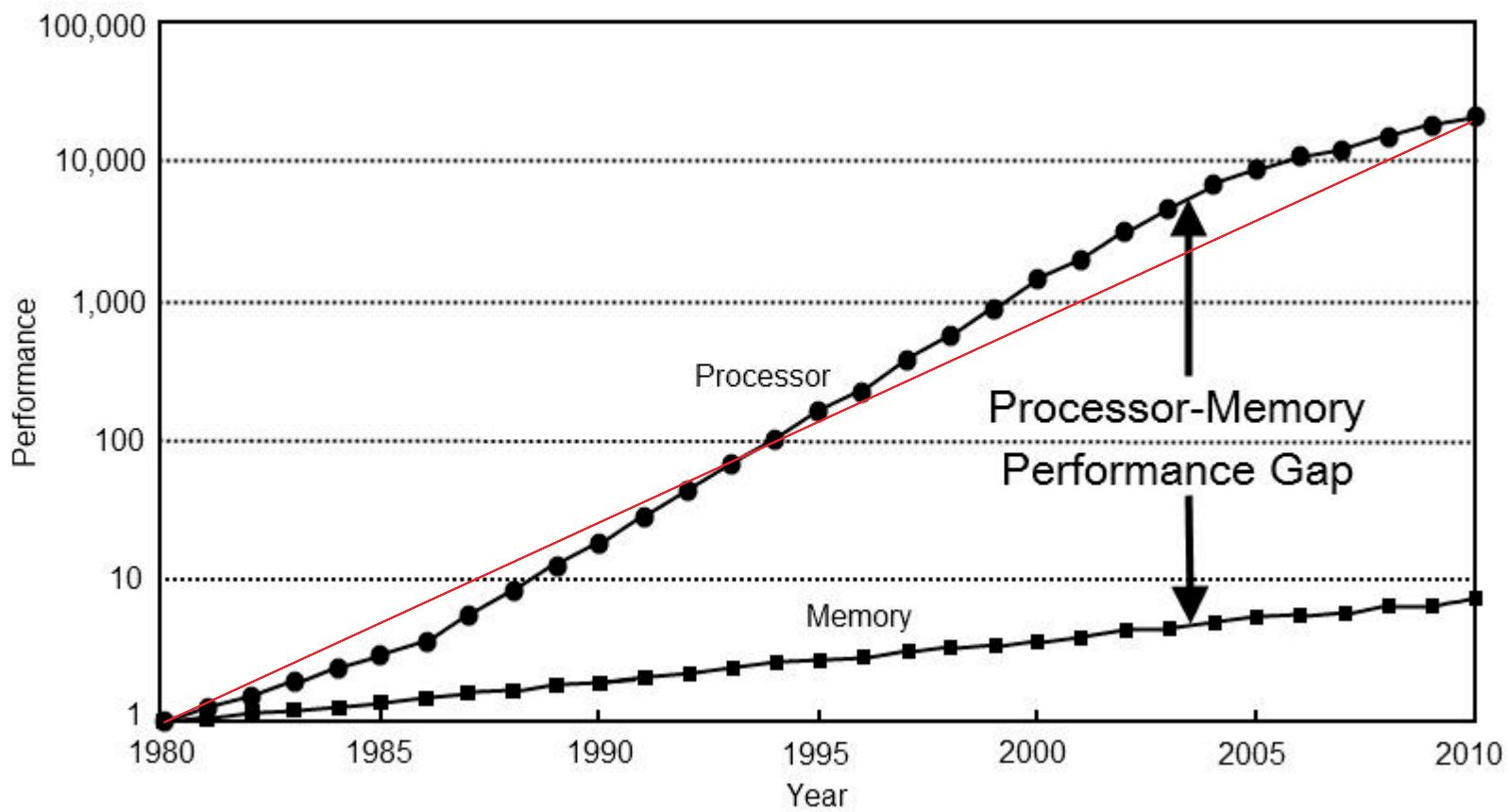


Shared memory: NUMA

Non-uniform memory access (NUMA): Time for memory access depends on location of data. Local access is faster than non-local access.



Memory wall problem



Challenges for multicore

- Relies on effective exploitation of multiple-thread parallelism
 - Need for parallel computing model and parallel programming model
- Aggravates **memory wall problem**
 - Memory bandwidth
 - Way to get data out of memory banks
 - Way to get data into multi-core processor array
 - Memory latency
 - Cache sharing

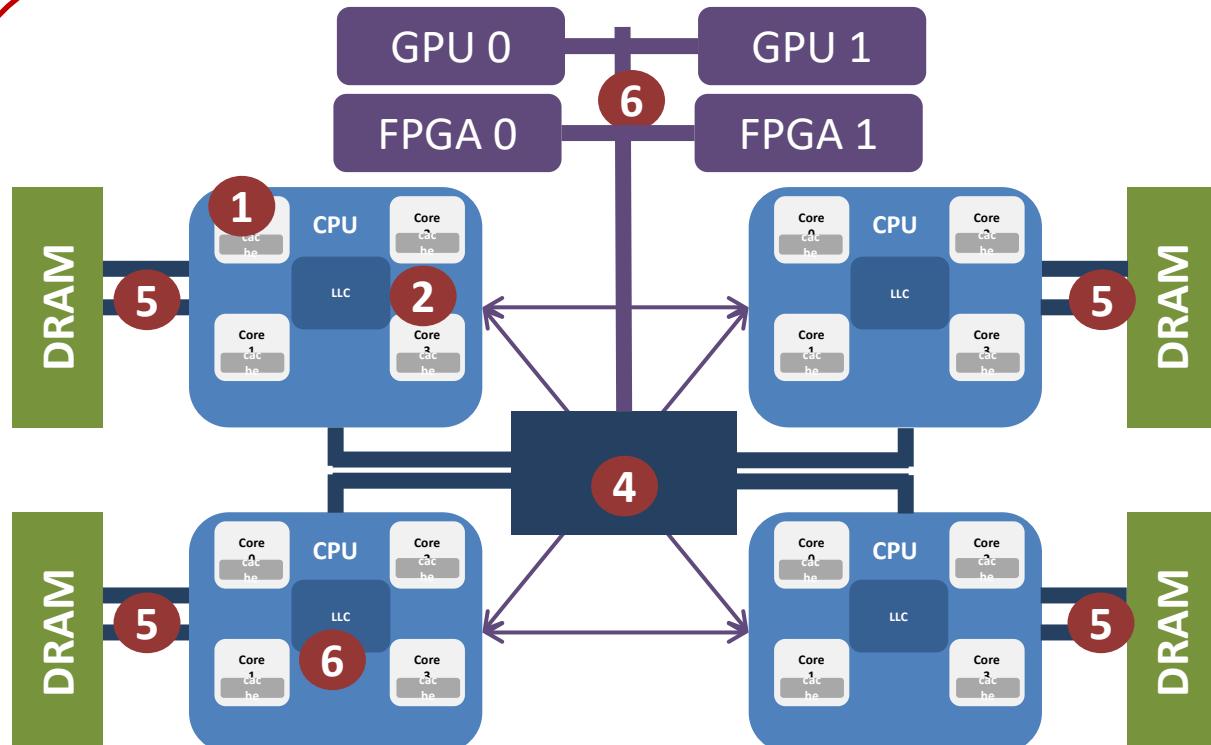
a little bit of jargon..

- Multiprocessor = server with more than 1 CPU
- Multicore = a CPU with more than 1 core
- Processor = CPU = socket

BUT SOMETIMES:

- Processor = core
- a process for each processor (i.e. each core)

Parallelism within a HPC node



- Parallel resources
 - ILP/SIMD units (2)
 - Cores (3)
 - Inner cache levels (3)
 - Socket/ccNuma domains (4)
 - Multiple accelerator (5)
 - MULTIPLE GPU (6)

All done

Why HPC is parallel ? 

Serial Computers 

Moore law/Dennard Scaling 

Parallel computers 