Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

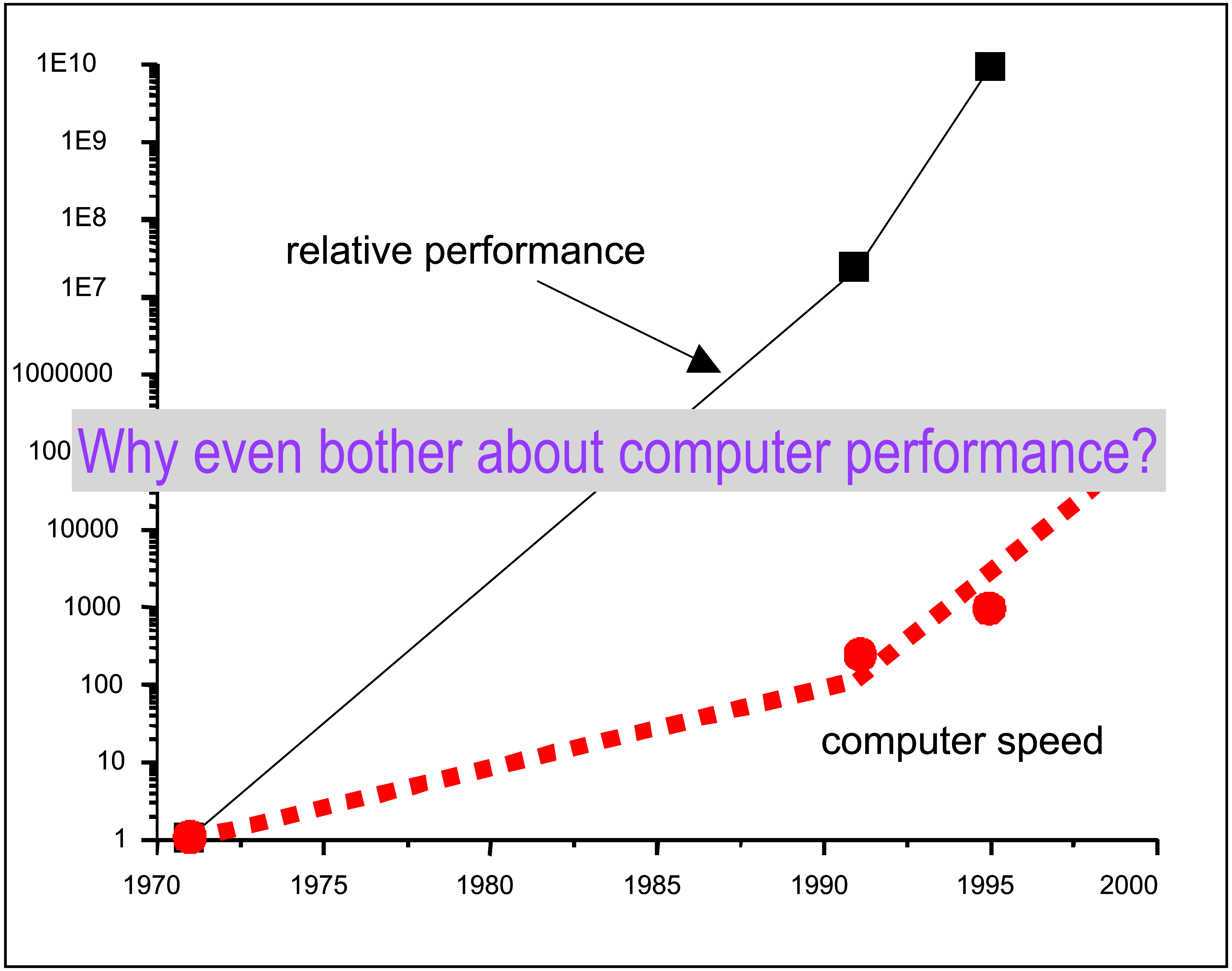
CSCS
Swiss National Supercomputing Centre

High-Performance Computing & Simulations in Quantum Many-Body Systems – PART I

Thomas Schulthess
schulthess@phys.ethz.ch

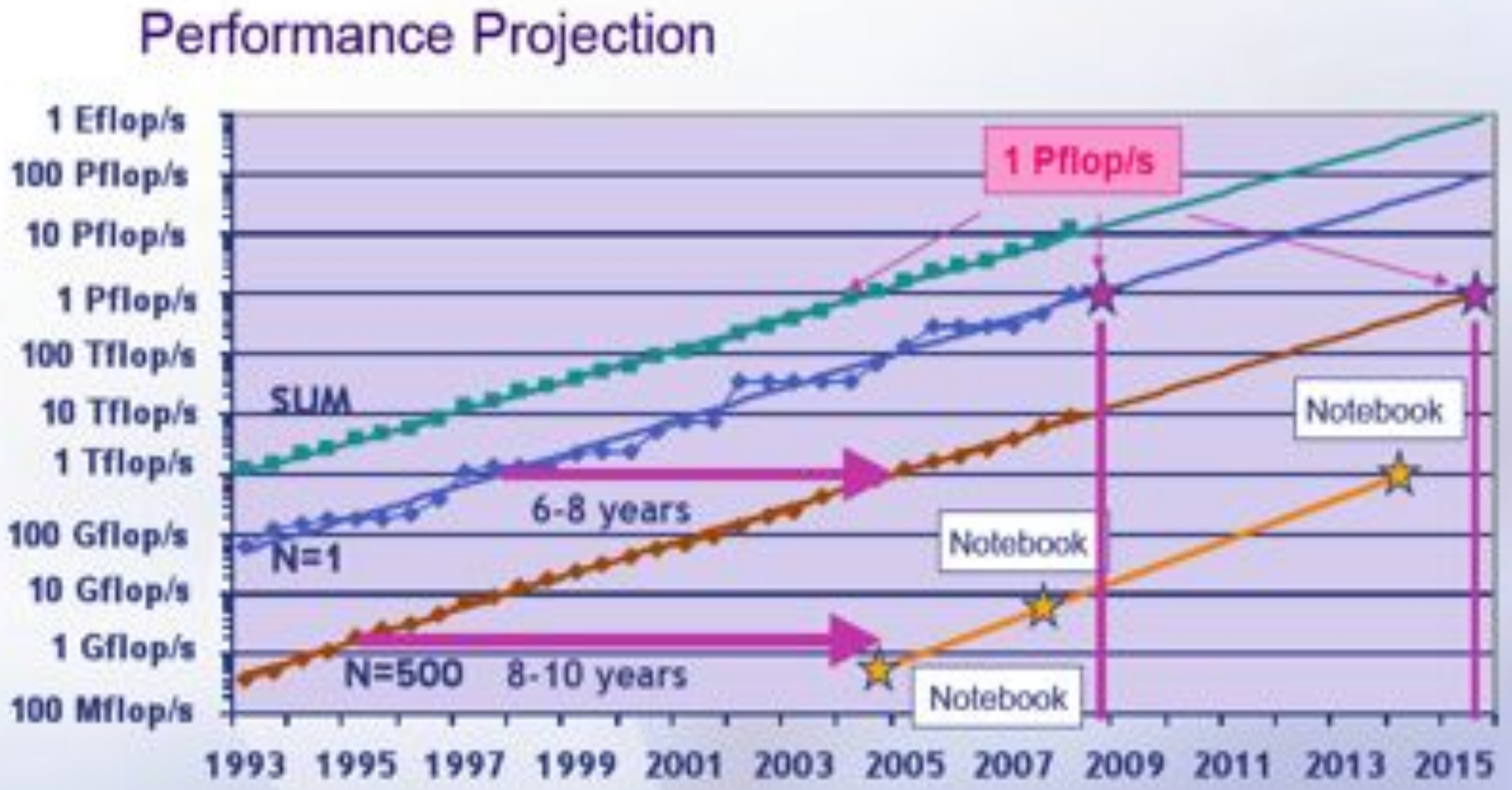
What exactly is high-performance computing?

Why even bother about computer performance?



Source: David P. Landau

Performance Development and Projection



Application performance seems to keep up with supercomputing systems performance (!)

~100 Kilowatts → ← ~5 Megawatts → ← 20-30 MW →

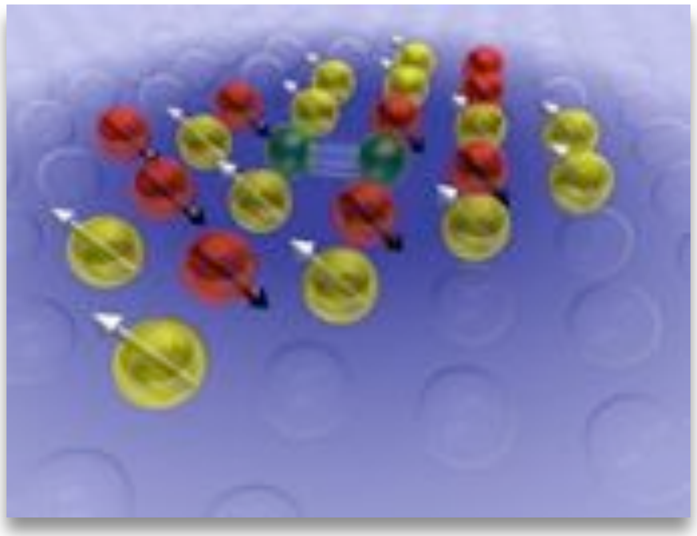
~1 Exaflop/s

100 millions or billion processing cores (!)

1 Gigaflop/s
Cray YMP
8 processors

1.02 Teraflop/s
Cray T3E
1’500 processors

1.35 Petaflop/s
Cray XT5
150’000 processors



1988 — 1998 — 2008 — 2018

First sustained GFlop/s
Gordon Bell Prize 1989

First sustained TFlop/s
Grondon Bell Prize 1998

First sustained PFlop/s
Gordon Bell Prize 2008

Another 1,000x in sustained performance increase (?)

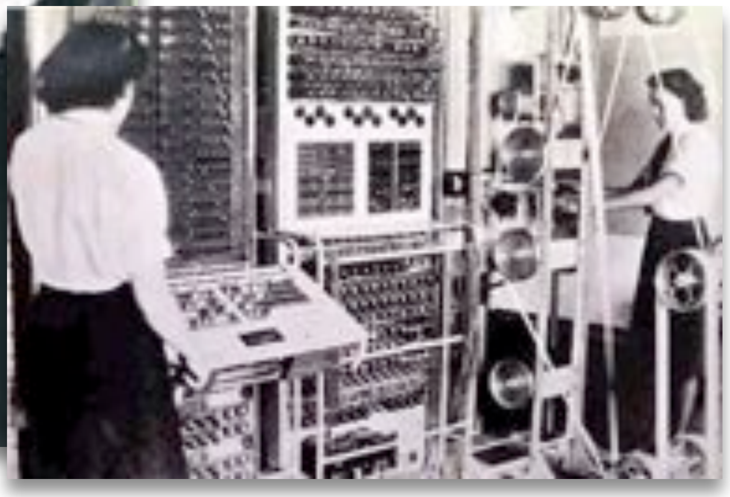
Plan for this lecture

- What is HPC and why worry?
- Historic background of scientific computing – how we came to where we are today
- Bottleneck and complexities of today's processors
- Parallel computers and parallel programming models
- Extrapolating Moore's Law into the future – why condensed matter physicists could be interested in computing

Electronic computing: the beginnings

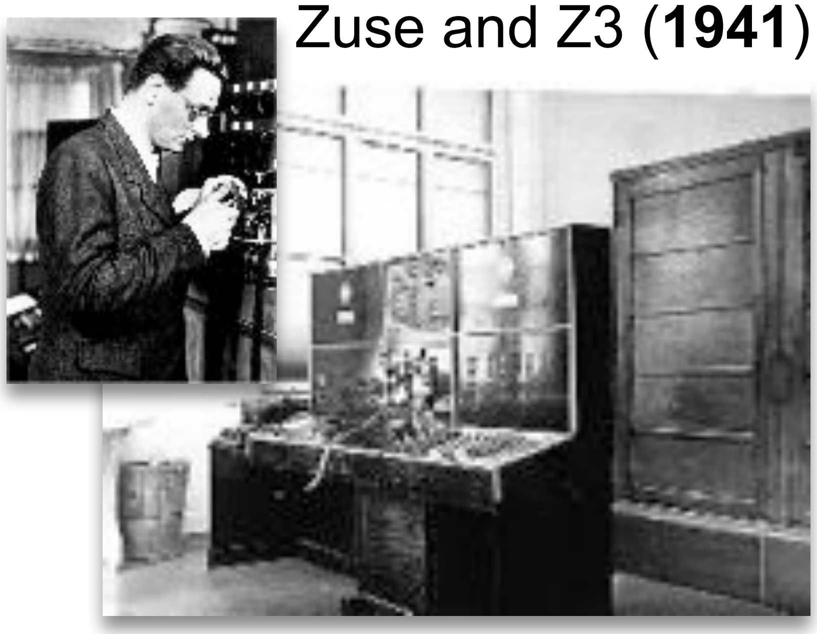
1939-42: Atanasoff-Berry Computer - Iowa State Univ.

1938: Konrad Zuse’s Z1 - Germany



1943/44: Colossus Mark 1&2 - Britain

Zuse and Z3 (**1941**)



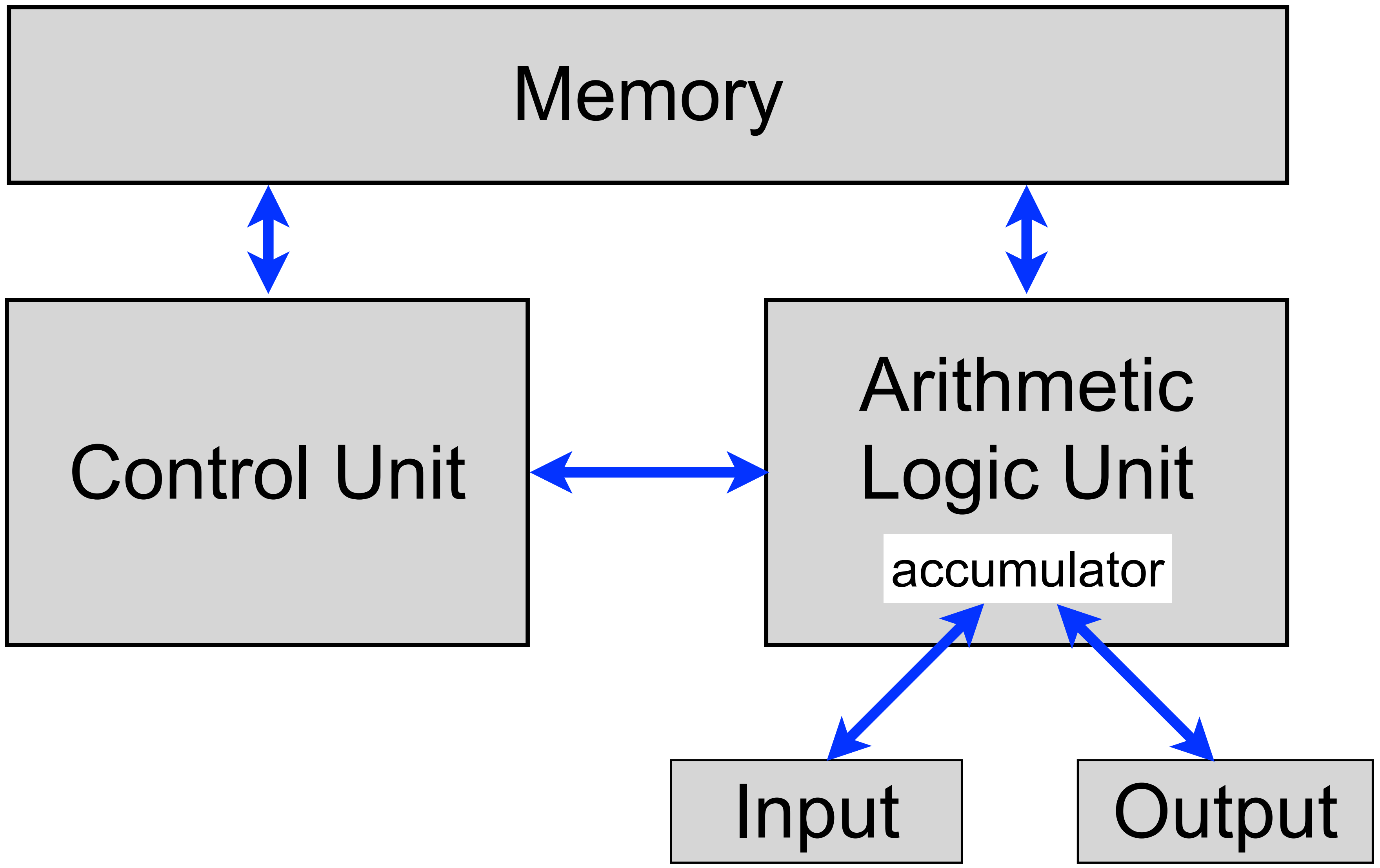
Z4 @ ETH (1950-54)

1945-51: UNIVAC I
Eckert & Mauchly - “first commercial computer”

1945: John von Neumann report that defines the “von Neuman” architecture

Von Neumann Architecture:

Invented by Eckert and Mauchly
Discussed in report by von Neumann (1945)



stored-program concept = general purpose computing machine

Since the dawn of High-performance computing:
Supercomputing at U.S. Dept. of Energy laboratories

1946: ENIAC

1952: MANIAC I

1957: MANIAC II

...

1974: Cray 1 - vector architecture

...

1987: nCUBE 10 (SNL) - MPP architecture

1993: Intel Paragon (SNL)

1993: Cray T3D

...

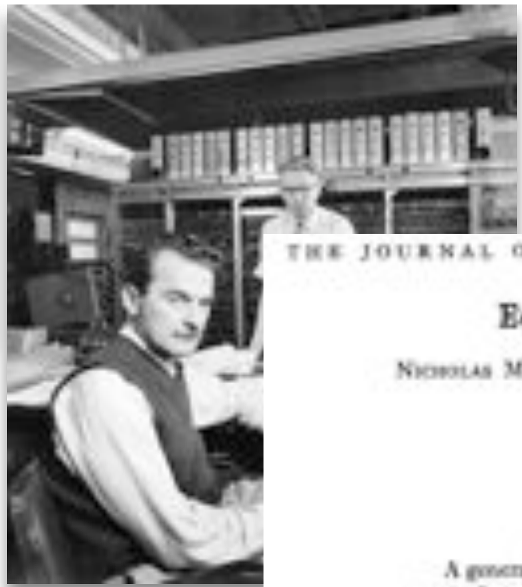
2004: IBM BG/L (LLNL)

2005: Cray Redstorm/XT3 (S

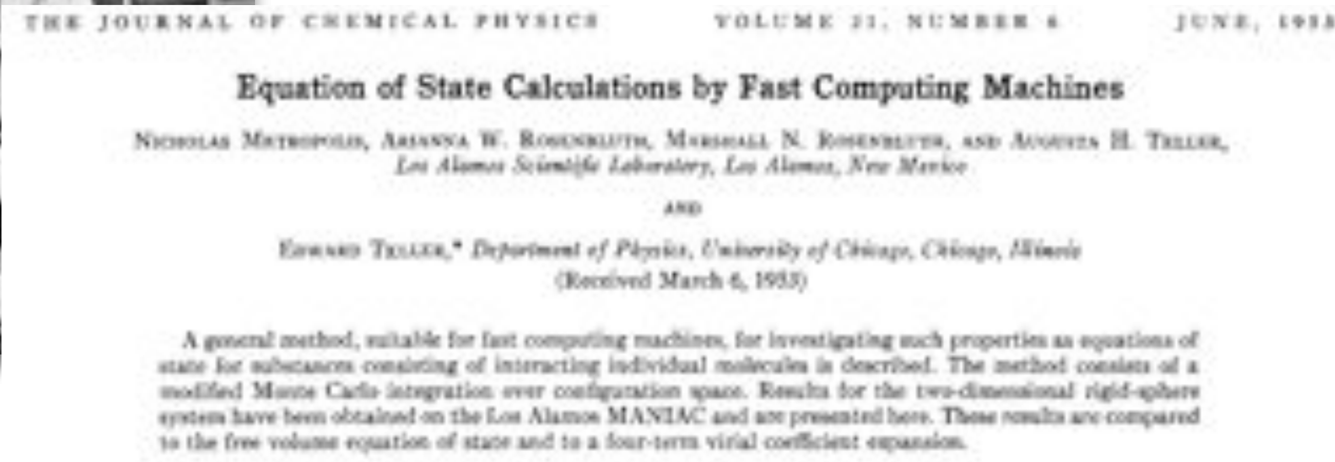
2007: IBM BG/P (ANL)

2008: IBM “Roadrunner”

2008: Cray XT5 (ORNL)



Nicholas Metropolis, physicists & leader of group in LANL’s T Division that designed MANIAC I & II



THE JOURNAL OF CHEMICAL PHYSICS VOLUME 21, NUMBER 6 JUNE, 1953

Equation of State Calculations by Fast Computing Machines

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, and Augusta H. Teller,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

and

Edward Teller,* *Department of Physics, University of Chicago, Chicago, Illinois*
(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

2002:
Japanese Earth Simulator - Sputnik shock of HPC

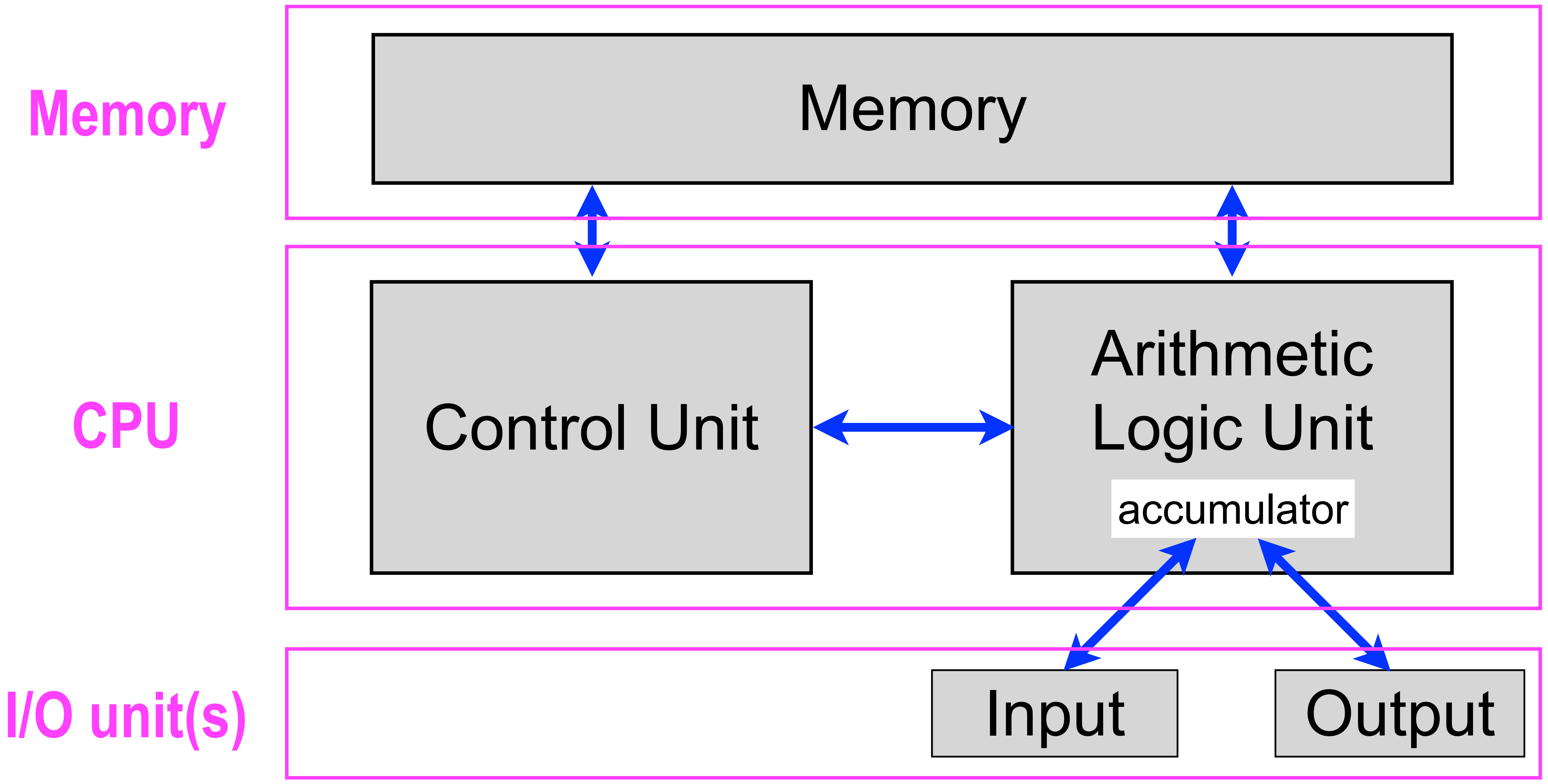
Peak: 1.382 TF/s
Quad-Core AMD Freq.: 2.3 GHz
150,176 compute cores
Memory: 300 TB

Today’s important types of processor architectures

- Scalar processor: process one data item (integer / floating point number) at a time
- Vector processor: a single instruction operates on many data items simultaneously
- Typical processor today: “pipelined superscalar”
  - Superscalar: simultaneously dispatch multiple instruction to redundant functional units (multiplier or adder)
  - Pipeline: set of processing elements connected in a series
  - Example: 2 multiplies and two add per cycle
    (4 floating point operations per cycle)

The good news: by and large compiler-level optimization will take care of this complexity

Von Neumann Architecture:



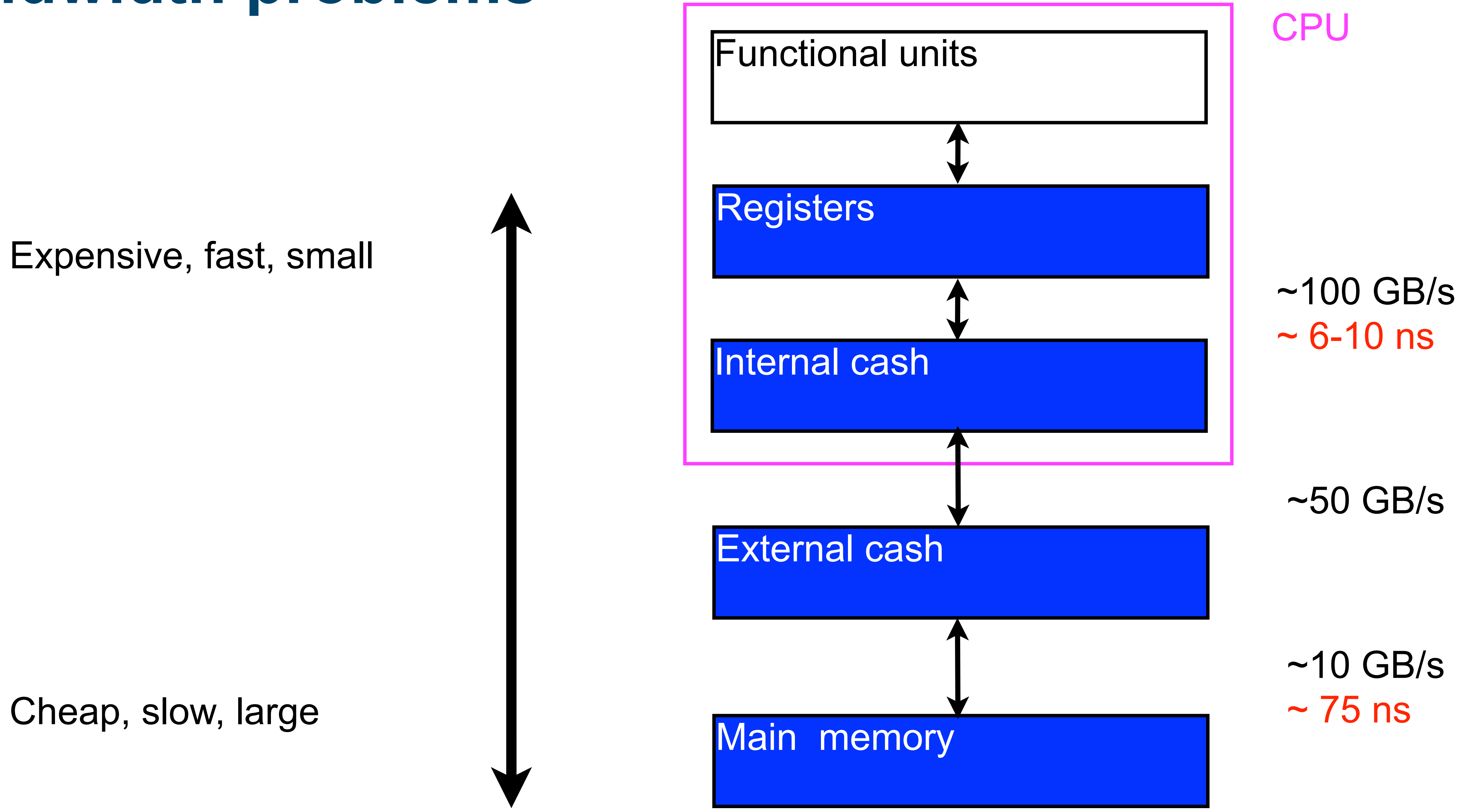
stored-program concept = general purpose computing machine

Computers in the past and today

	1970s (*)	my laptop	improvement
clock (CPU)	6 MHz	~2GHz	300 x
Flop/s	6 MFlop/s	~8 GFlop/s	10^3 x
RAM	128kB	8GB	10^6 x
Mem. latency	850ns	~100ns	20 x

(*) Charles Thacker's computer in the 1970s

Memory hierarchy to work around latency and bandwidth problems



Moore’s Law is still alive and well

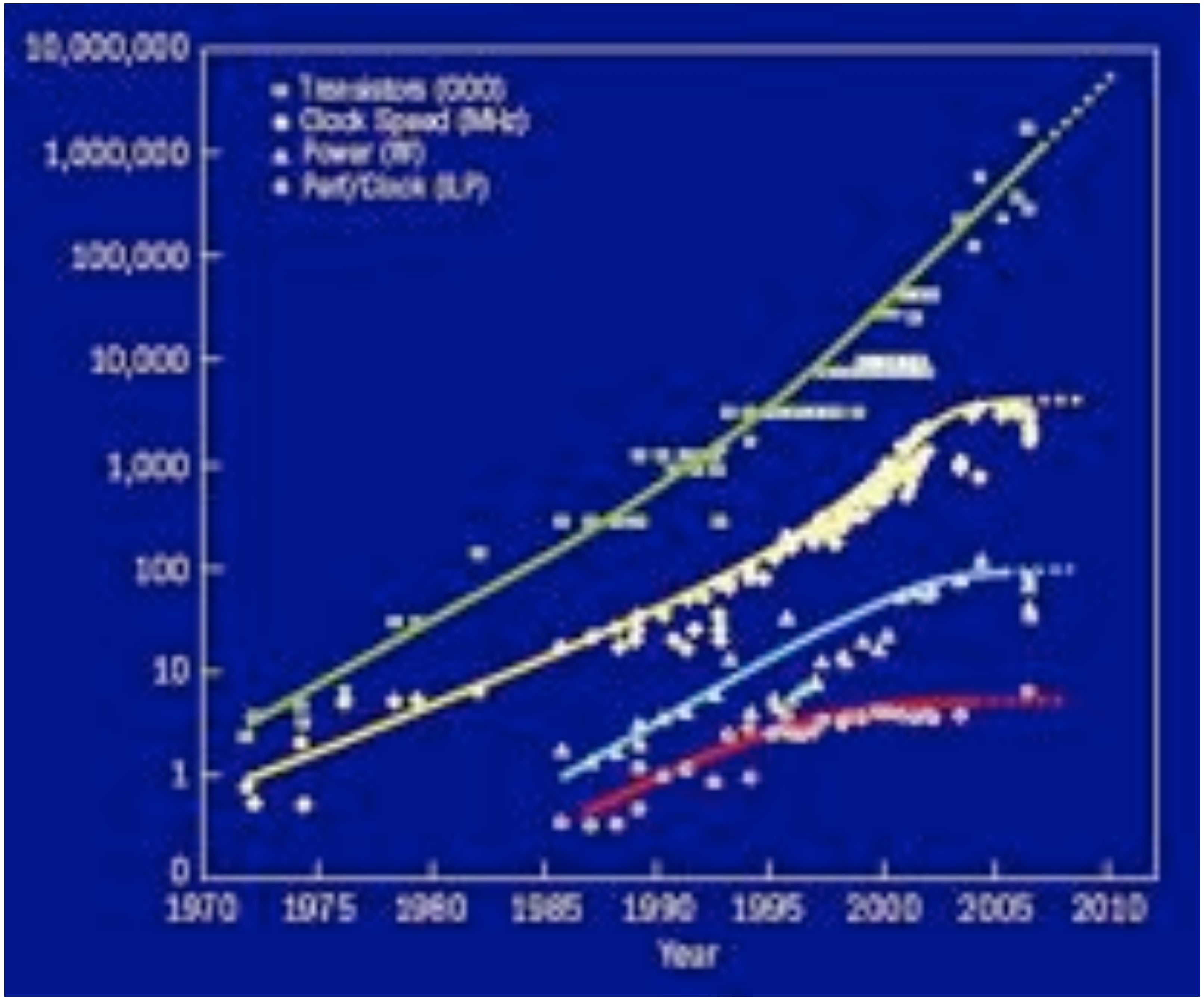
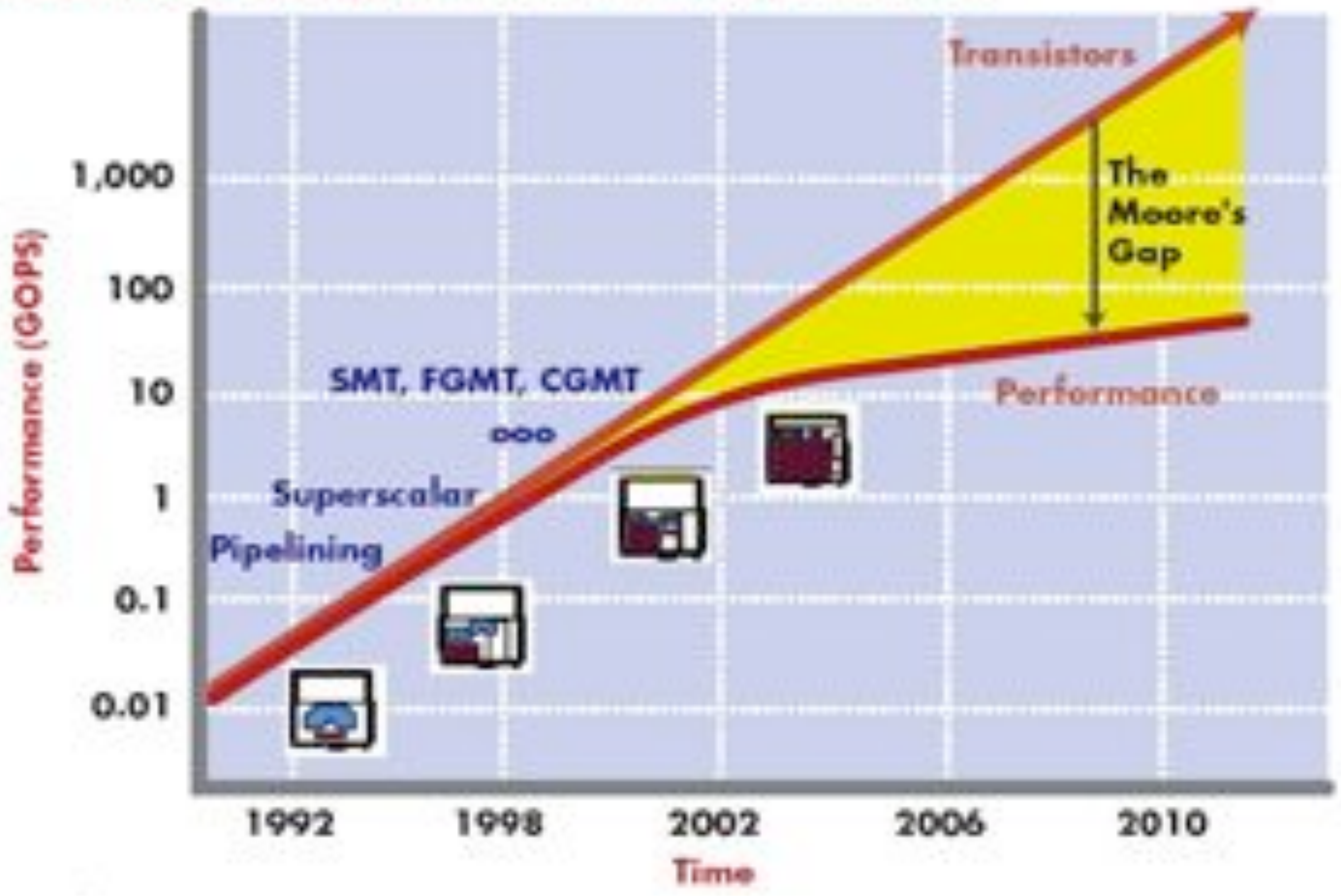


illustration: A. Tovey, source: D. Patterson, UC Berkeley

Single processor performance is no longer tracking Moore's Law



Performance increase due to exploding number of processing cores

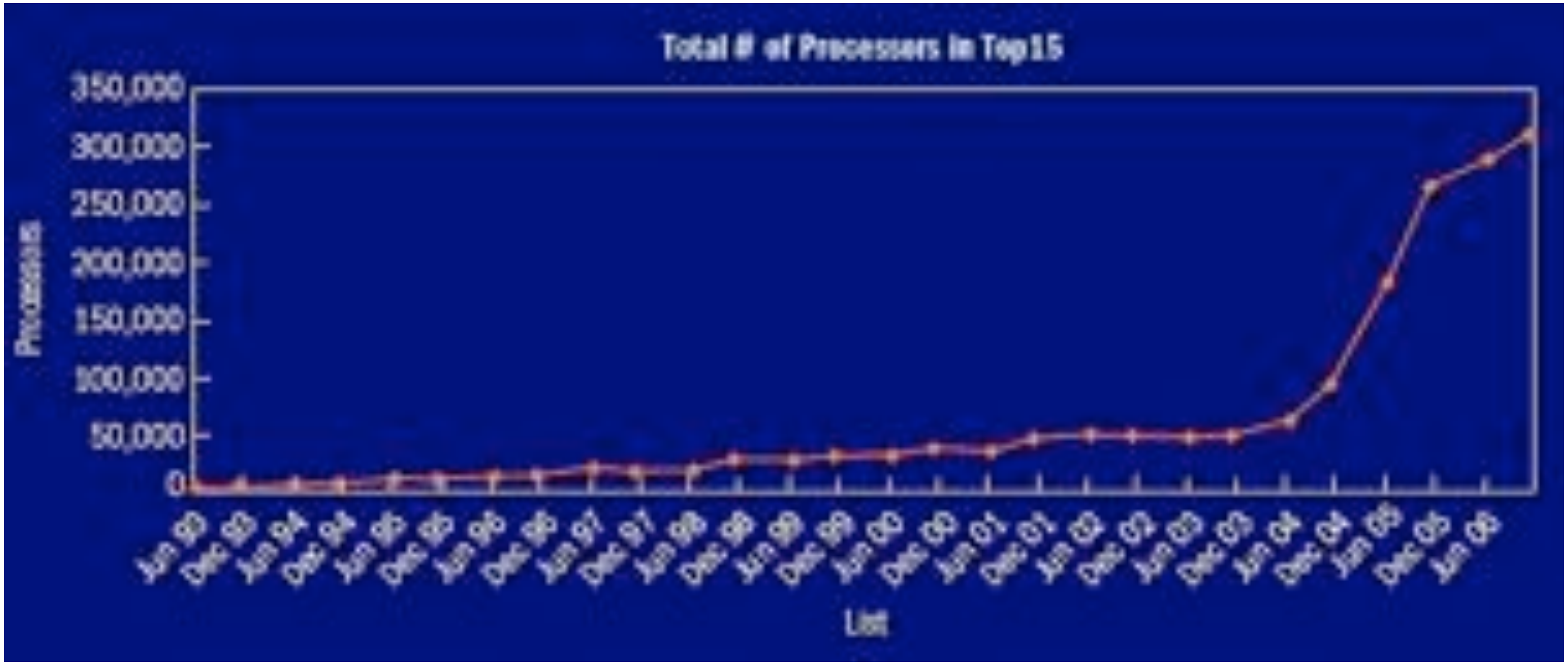
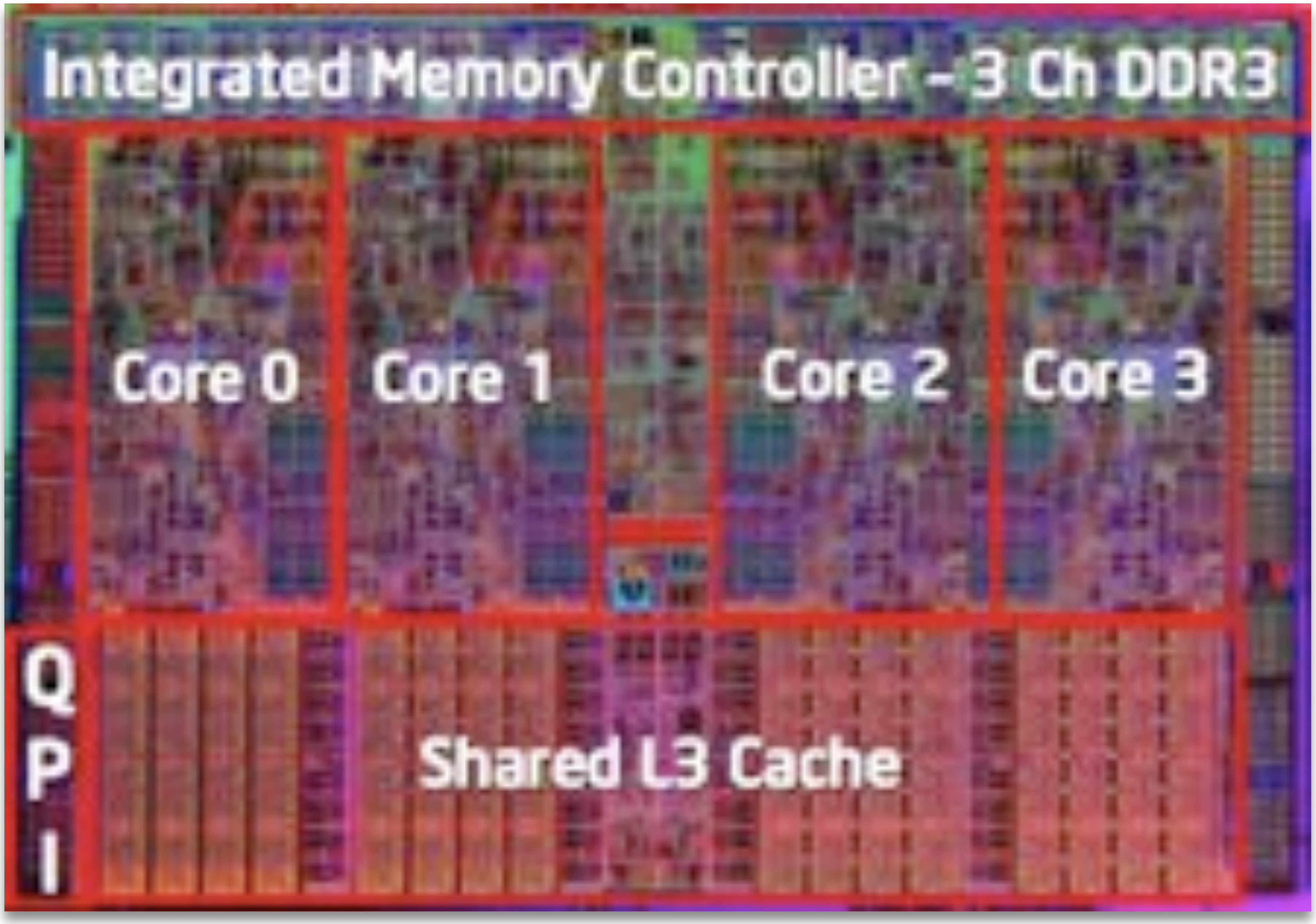
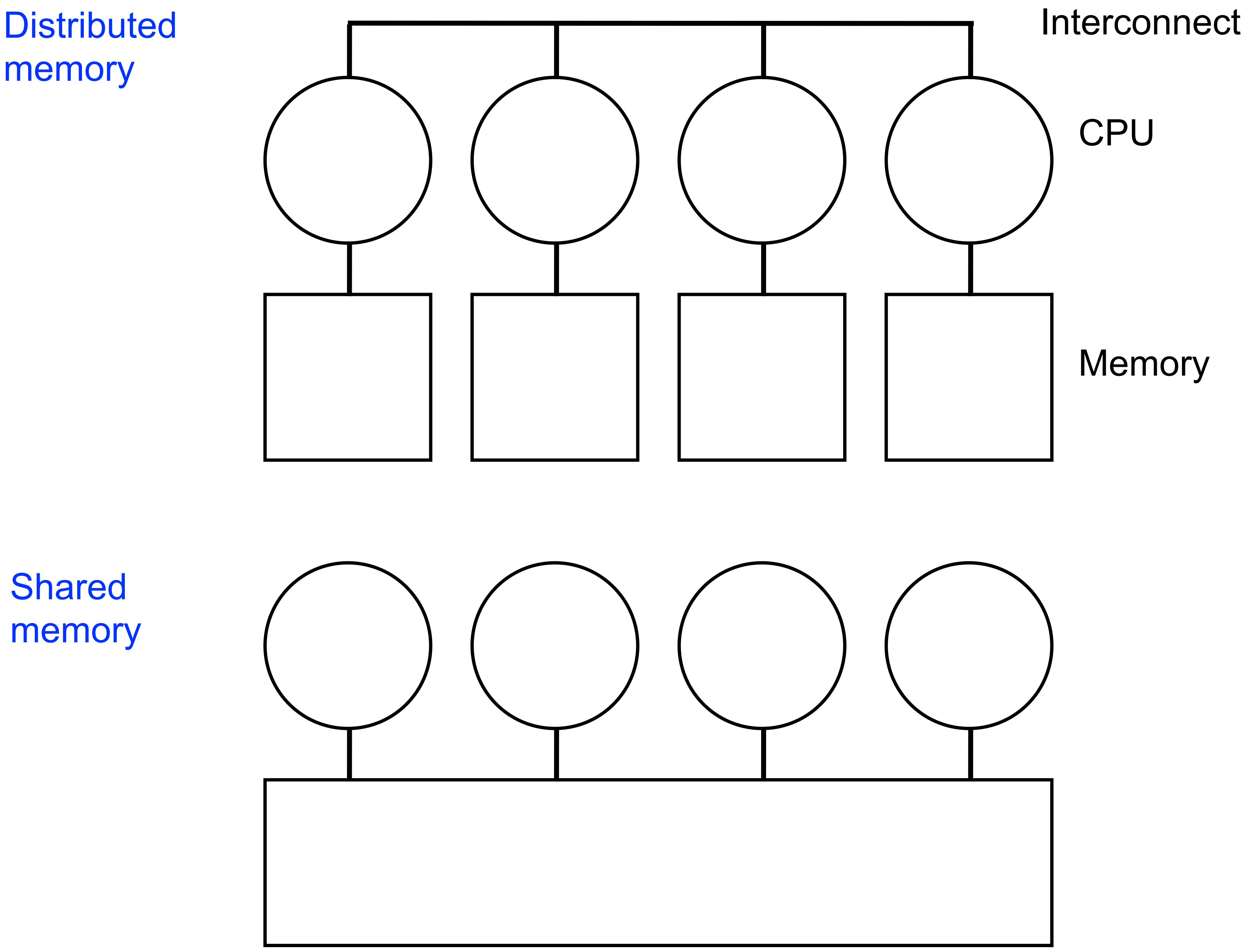


illustration: A. Tovey, source: D. Patterson, UC Berkeley

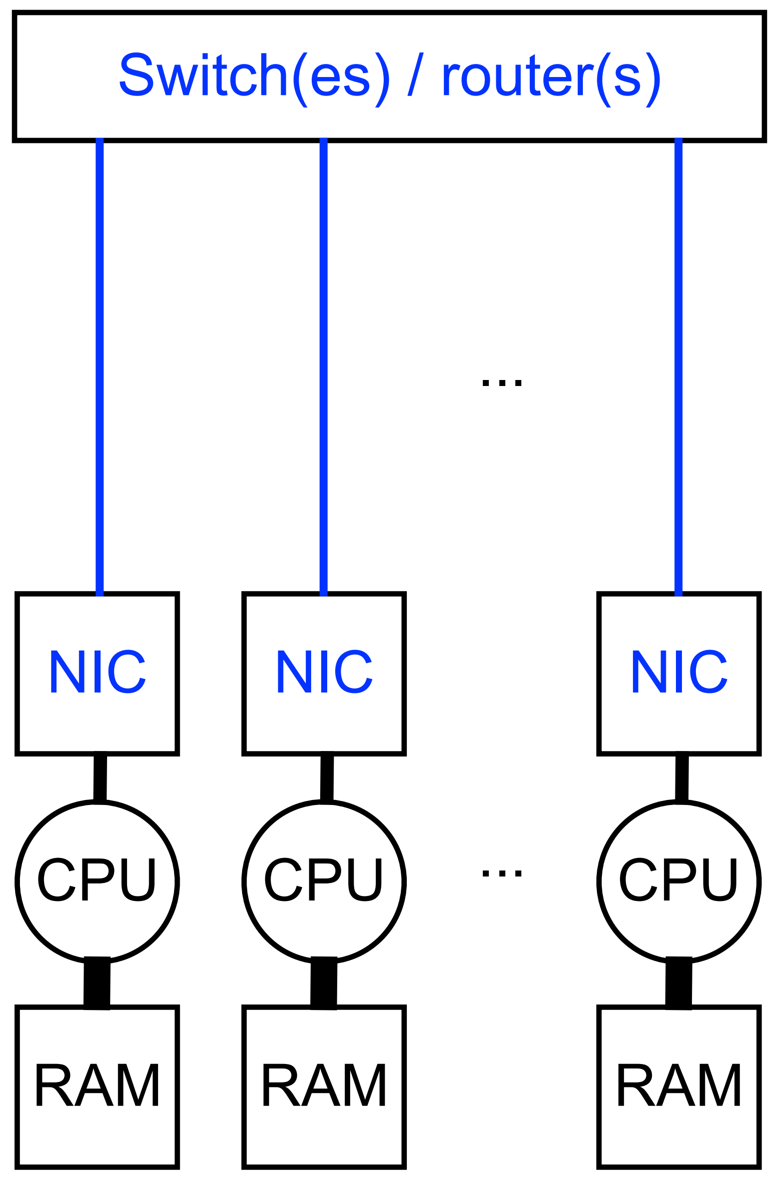
Multi-core processors (since middle of decade)

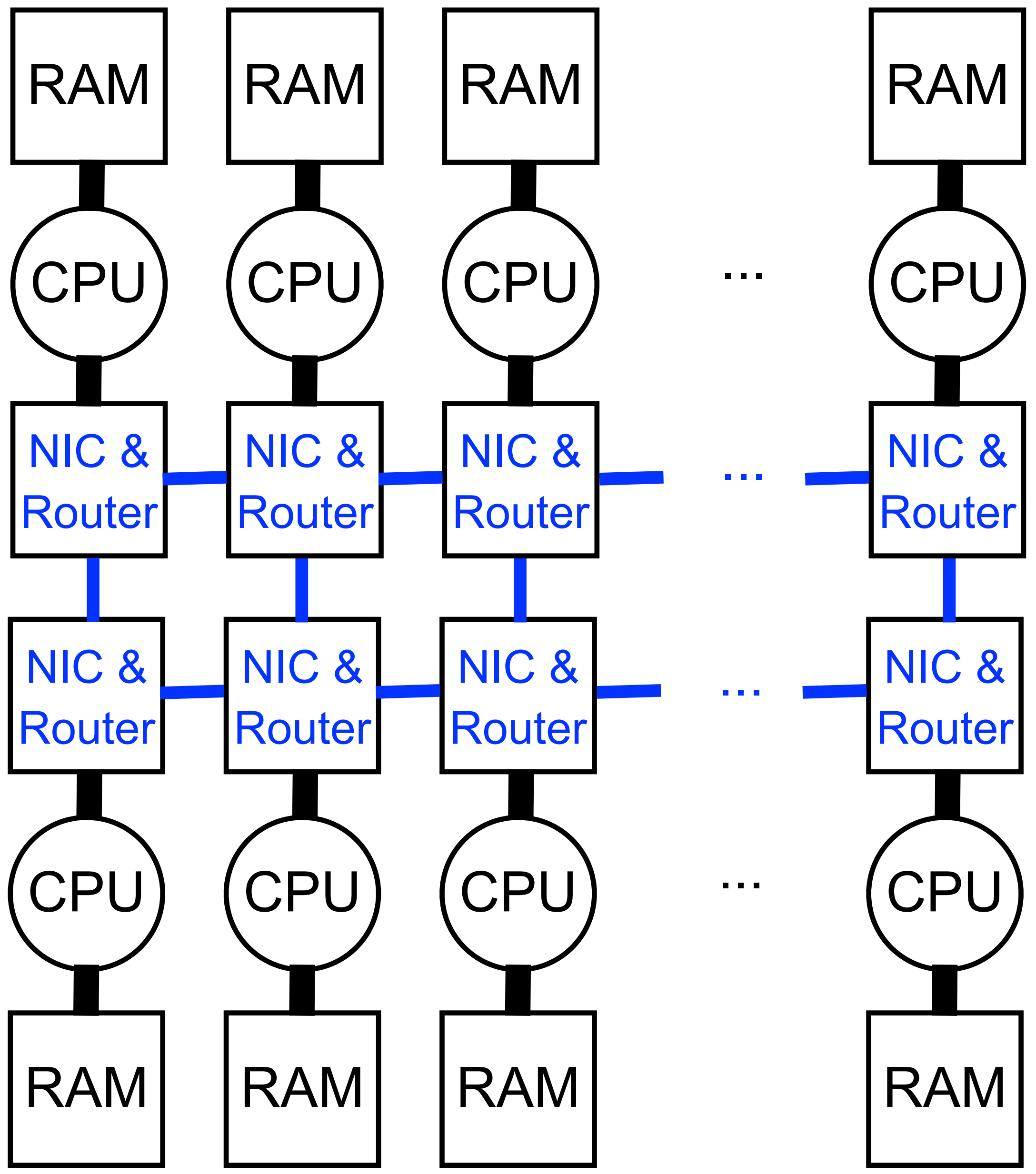


Distributed vs. shared memory architecture

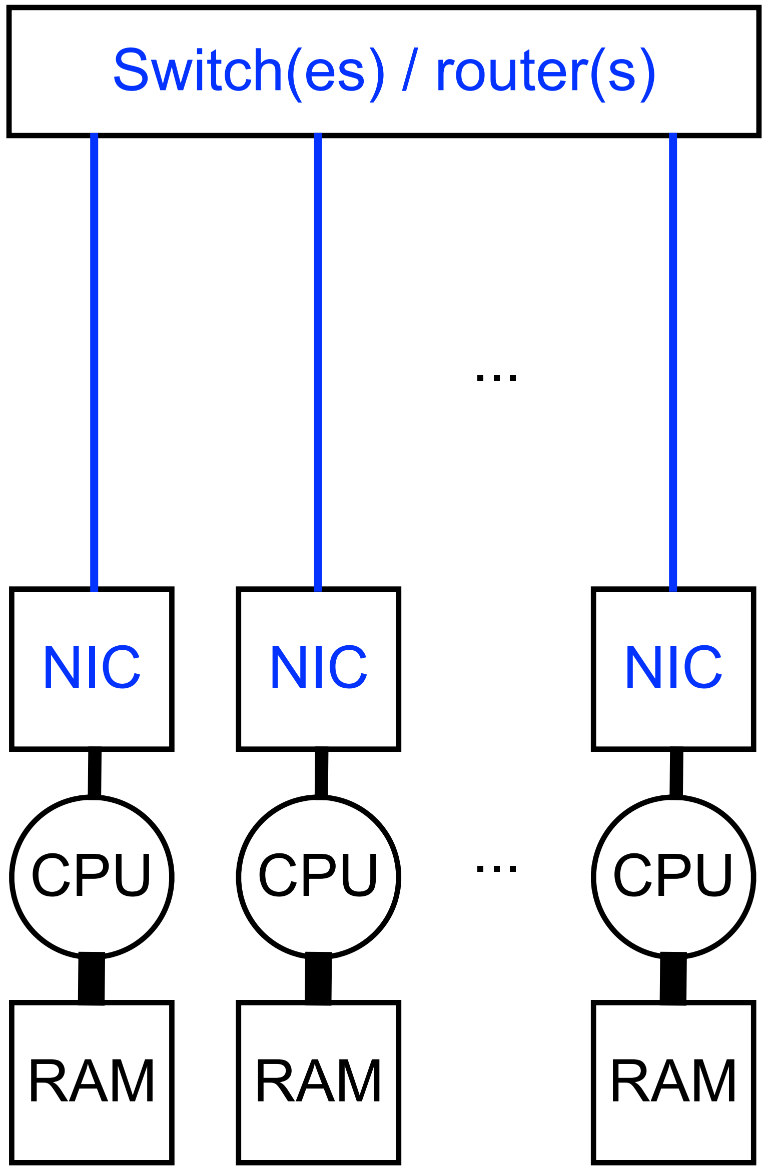


Interconnect types on massively parallel processing (MPP) systems – distributed memory





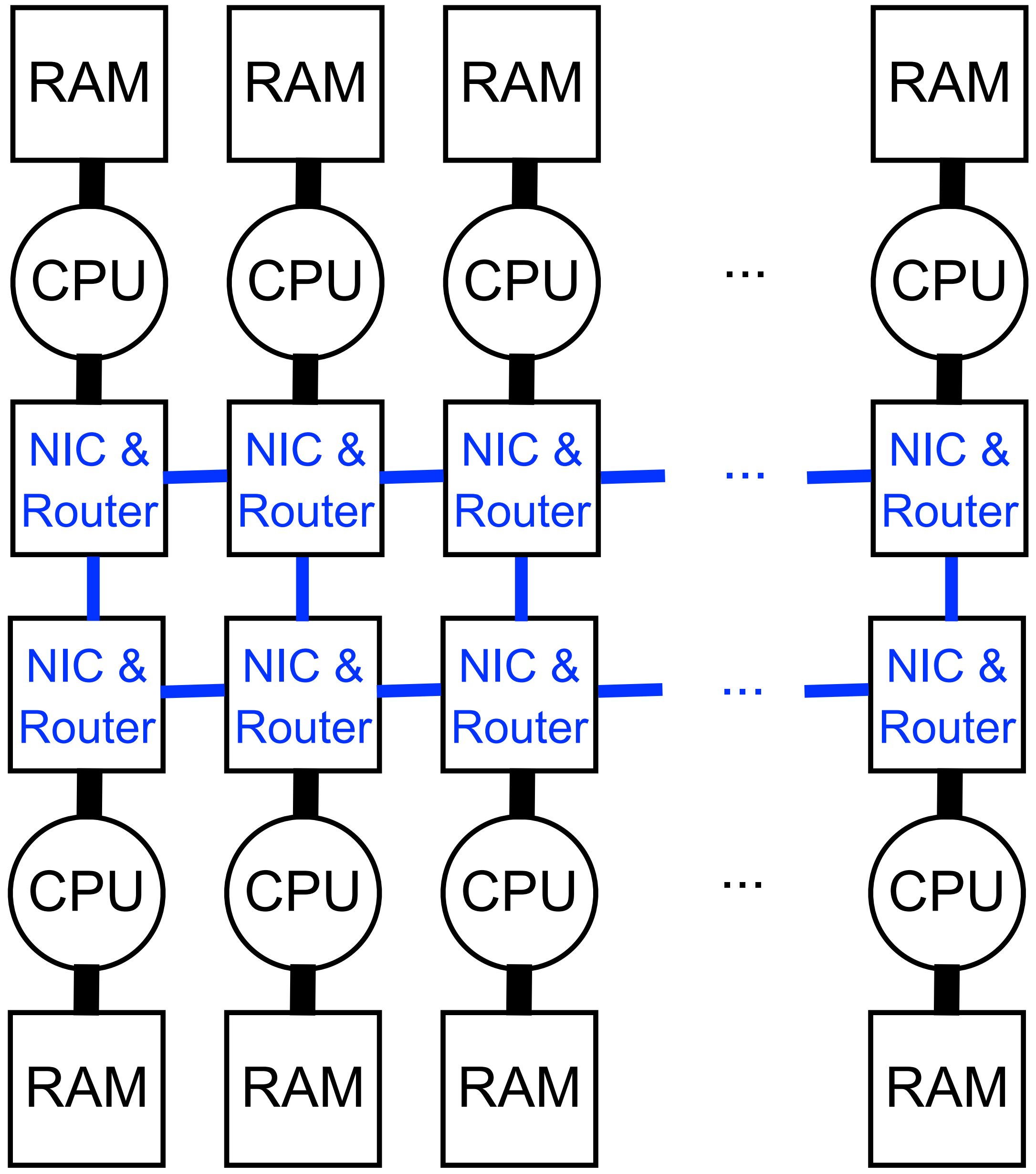
Infiniband networks (ethernet) – separating compute partition from router/switch



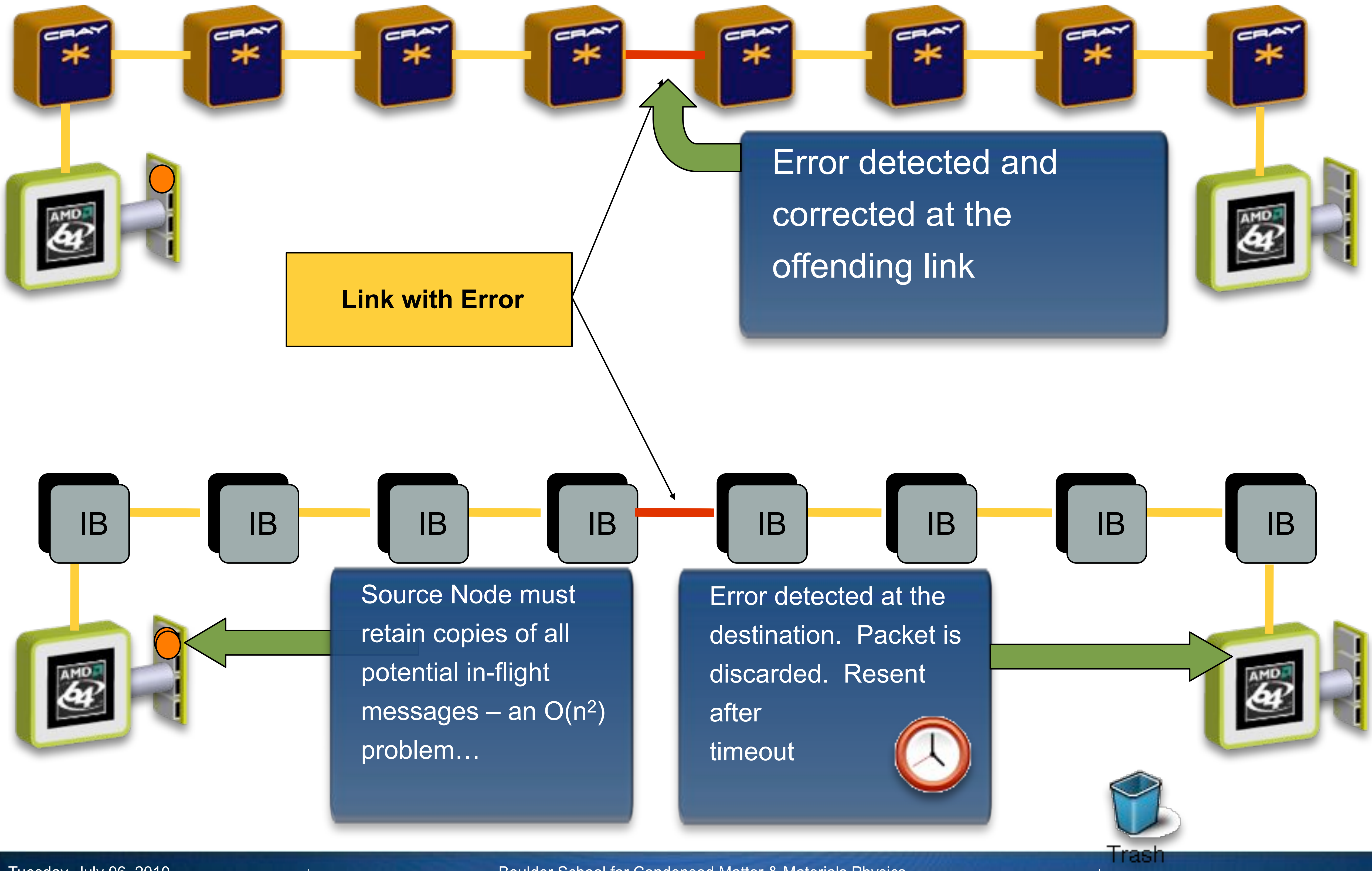
- Open / commodity network
- More flexibility with topology (usually fat tree; but hyper cube, dragon fly, etc. also possible)
- Scales to only up to 10^4 nodes (ideal for small clusters)
- Latency can be as low as microsecond
- Bandwidth not as high as proprietary

Proprietary network – Integrated router/NIC (network interconnect chip) and compute node

- Proprietary networks (today)
  - IBM BG/P – torus + fat tree
  - Cray Seastar (XT5) – torus
  - Cray Gemini (XE6) – torus
- Reliable and scales to 100k nodes
- Higher bandwidth (similar to PDIe)
- Latency slightly high than infiniband



Complexity of interconnect

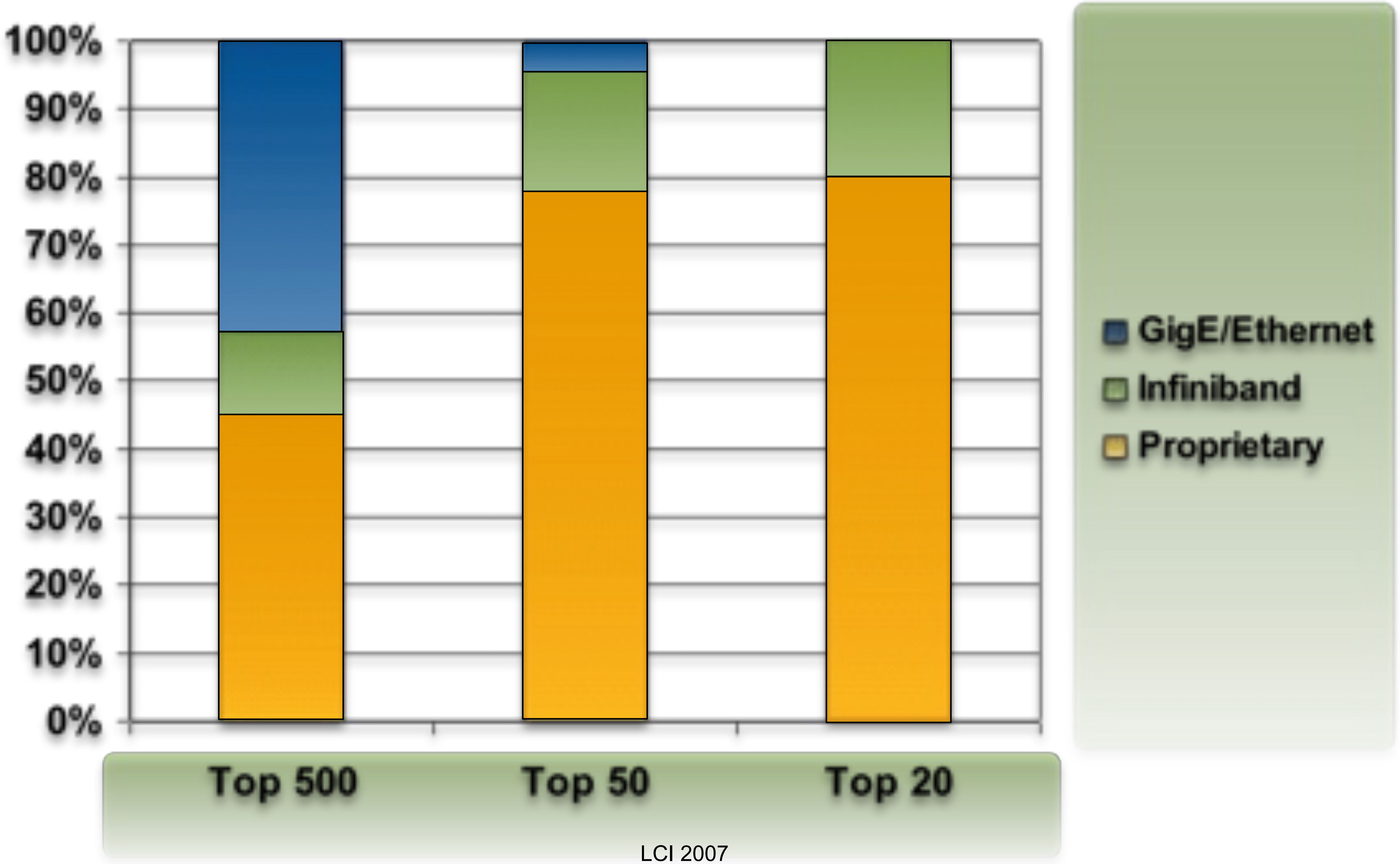


Error detected and corrected at the offending link

Source Node must retain copies of all potential in-flight messages – an O(n^2) problem…

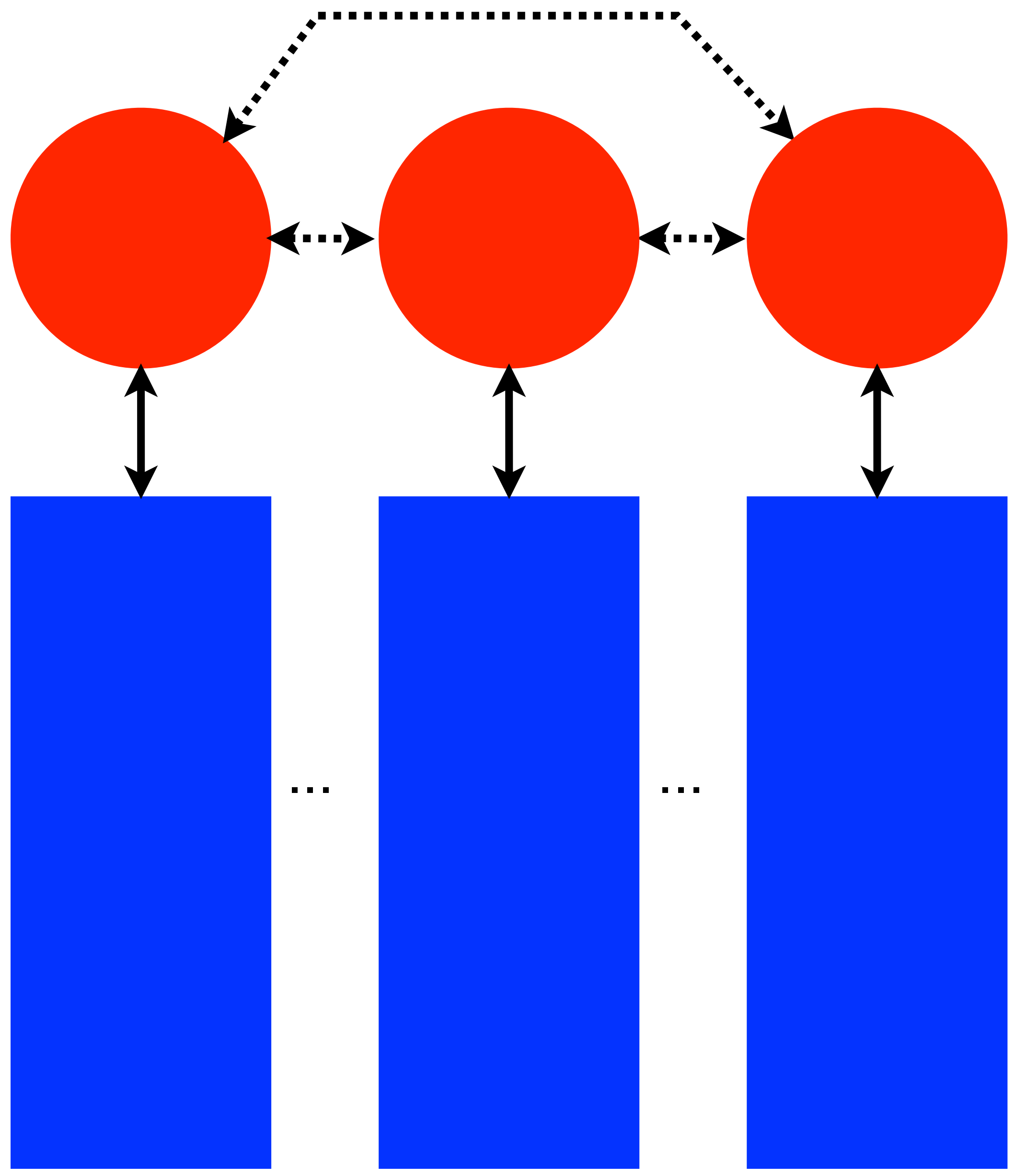
Error detected at the destination. Packet is discarded. Resent after timeout

Interconnects in the TOP500 systems



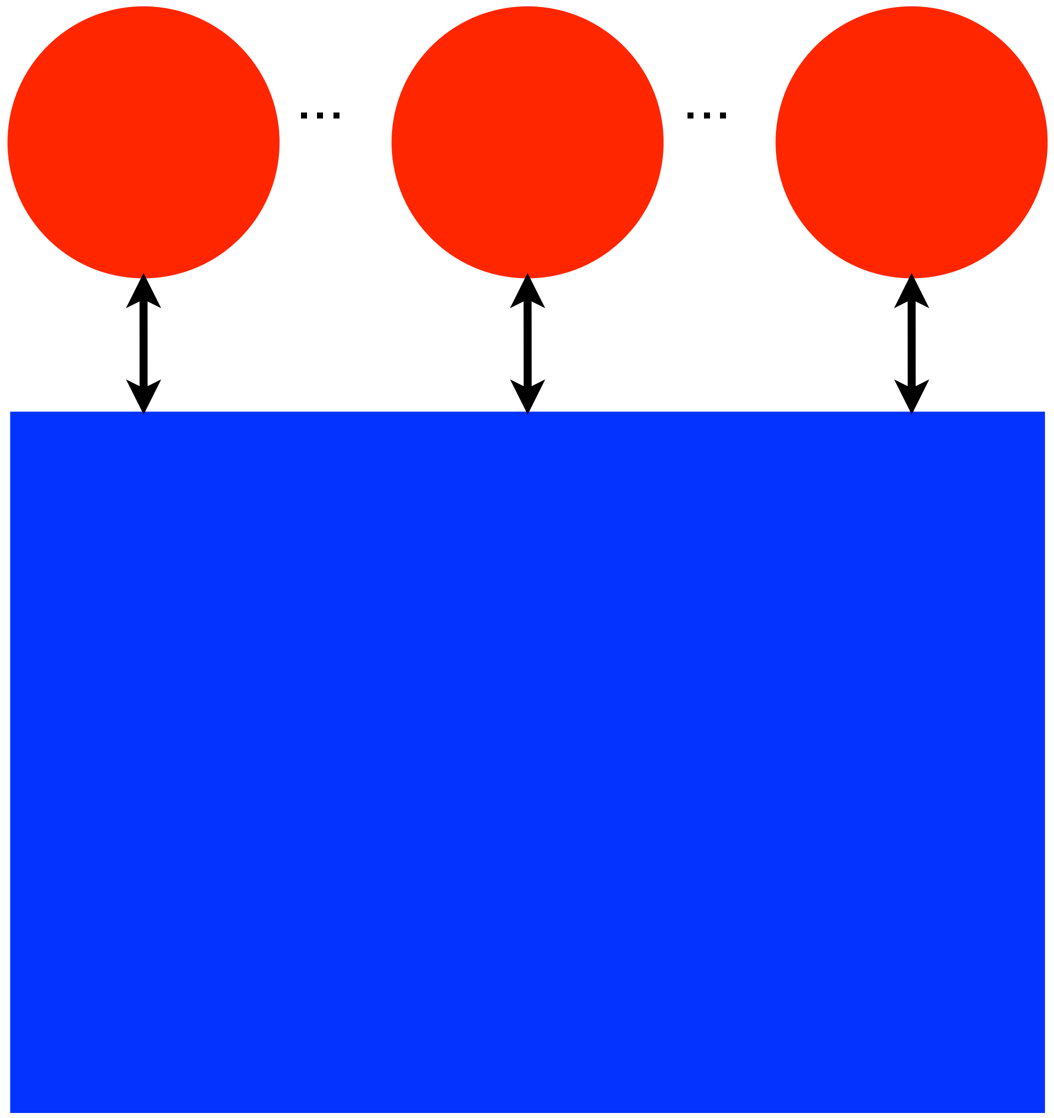
LCI 2007

Programming models (I): message passing



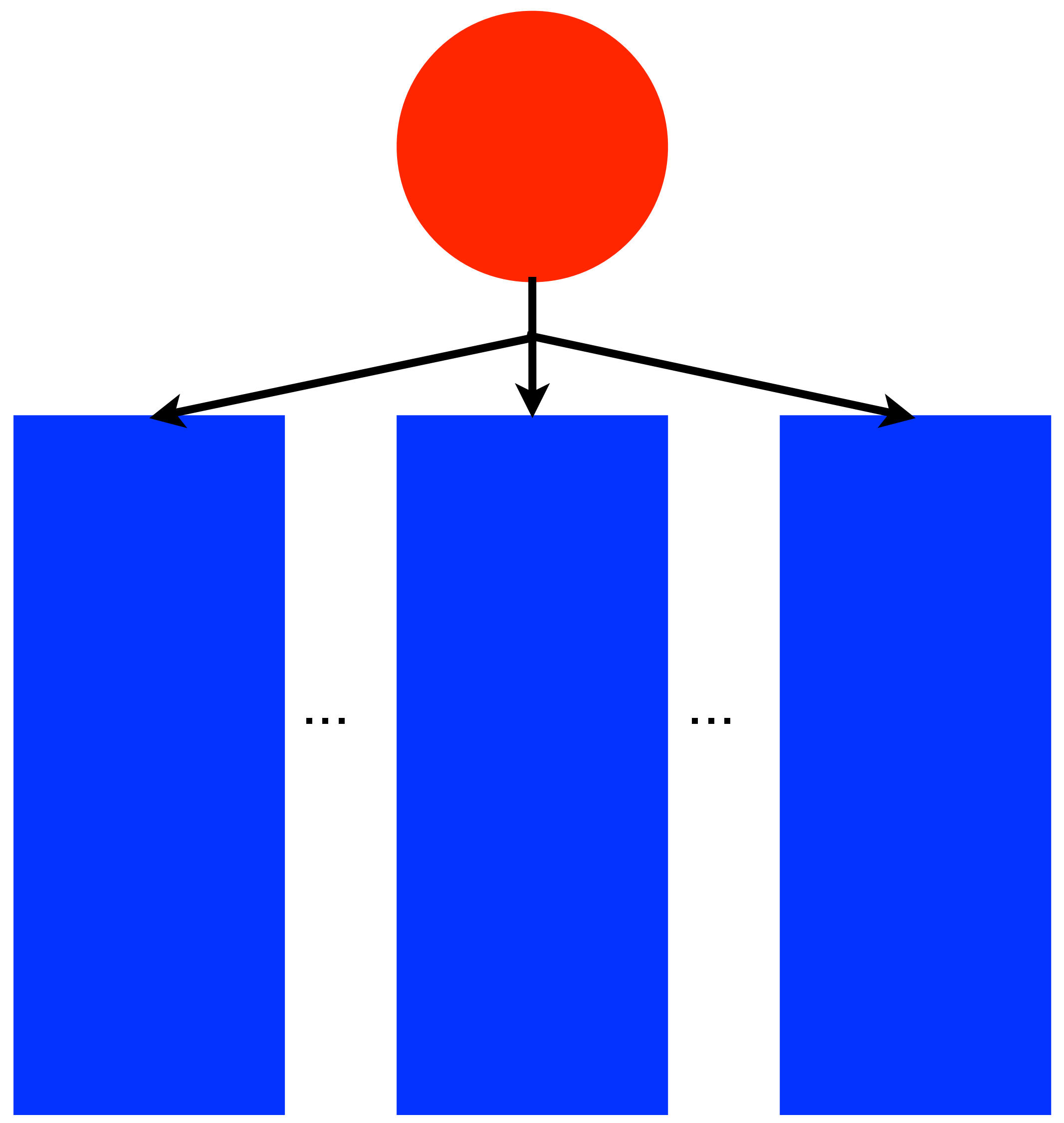
- Concurrent sequential processes cooperating on the same task
- Each process has own private space
- Communication is two-sided through send and receive
  - Large overhead!
- Lots of flexibility in decomposing large problems, however, provides only fragmented view of the problem
  - All burden is placed on the programmer to maintain global view
- Examples are message passing libraries like MPI or PVM

Programming models (II): shared memory



- Multiple independent threads operate on same shared address space
- Easy to use since there is only one type of memory access
  - One-sided remote access (low overhead)
- Application view remains integrated (global view)
- Shared memory hardware doesn't scale (local & remote memory access)
- It is difficult to exploit inherent data locality - degradation of performance!
- Examples are OpenMP or Pthreads
  - Compiler directive used with C, Fortran, ...

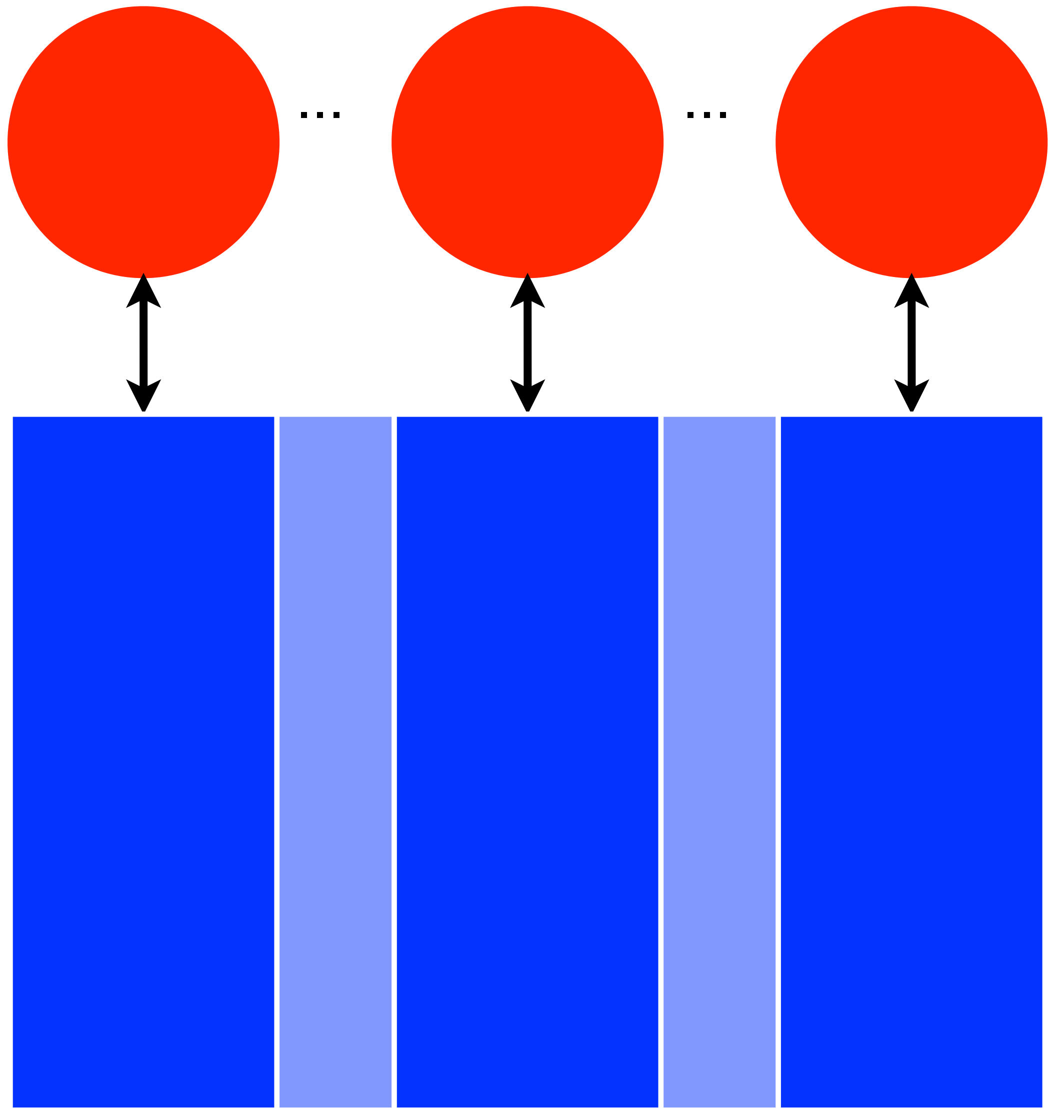
Programming models (III): data parallel



- Concurrent processing of many data elements in the same manner
- Executing only one process (on many processors)
- Major drawback: does not permit independent branching
  - Not good for problems that are rich in functional parallelism
- Popular examples are C* and HPF
  - Revived today with GPGPU & CUDA

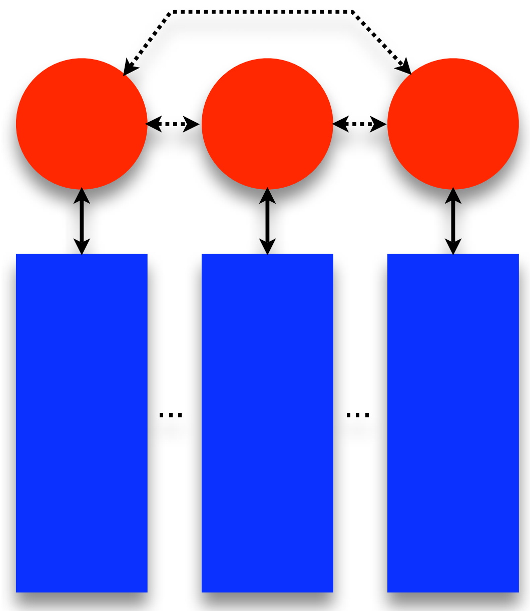
Programming models (IV): distributed shared memory

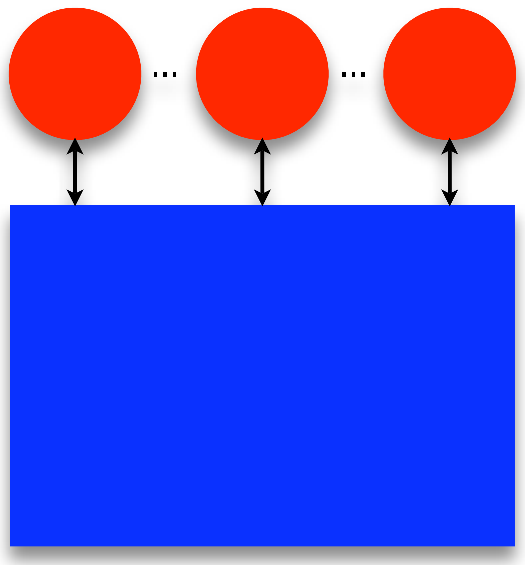
Also called partitioned global address space (PGAS) model

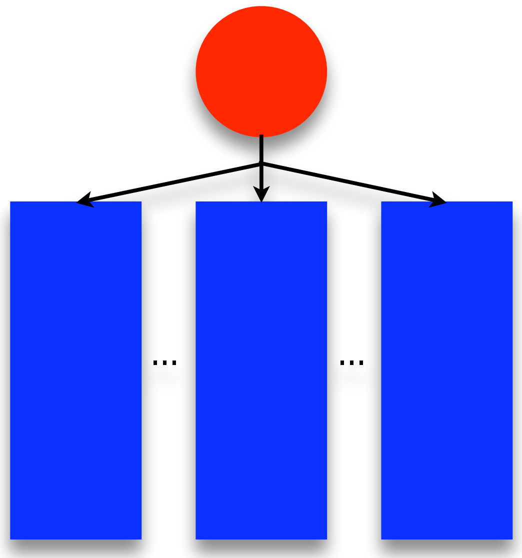


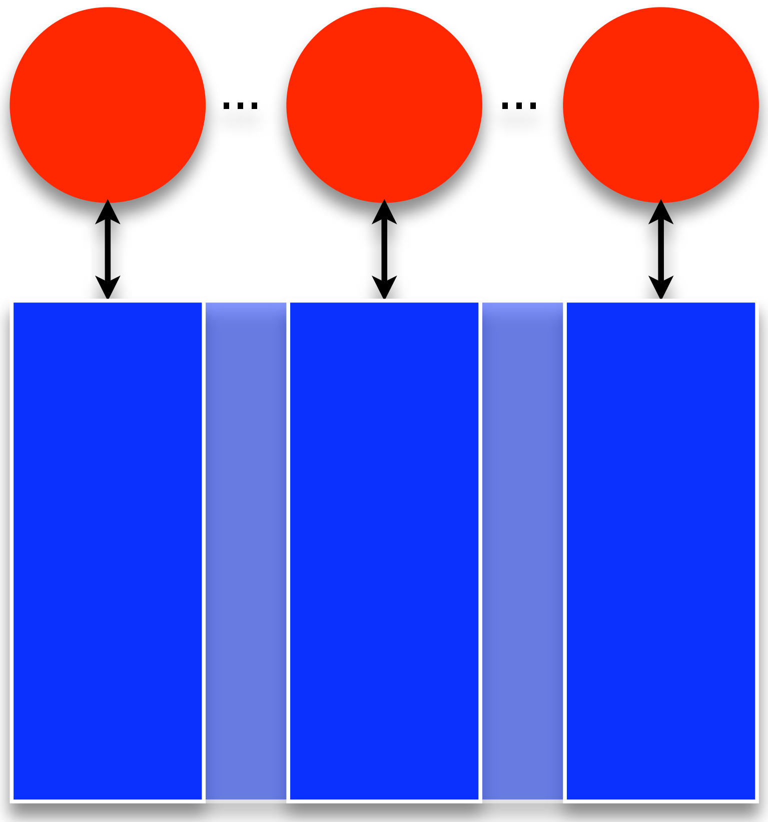
- Independed threads operate in shared memory space
  - preserve global view of program
- Shared space is locally partitioned among threads
  - allows exploiting data locality
- "Single program multiple data stream" (SPMD) execution
  - independent forking (functional parallelism)
- Popular examples: UPC and co-Array Fortran; or Global-Array library
- May still not have the same flexibility as Message Passing Model

Distributed shared memory or PGAS: keeping the best from all other models









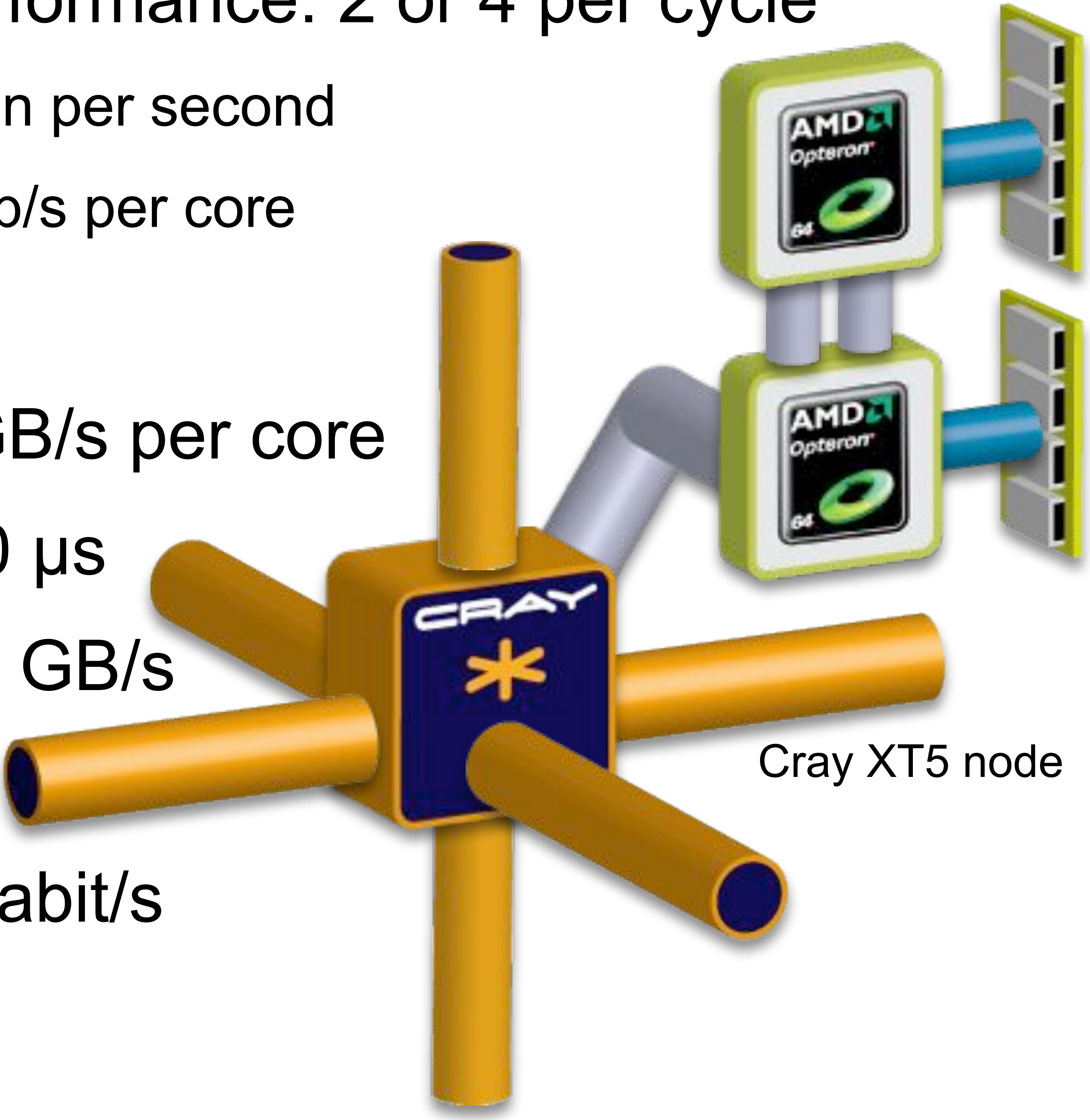
Hardware optimized PGAS:

Cray XE6 with Gemini interconnect – fall 2010 @ NERSC and Edinburgh (first XE6 cabinet @ CSCS since June 2010)

IBM BG/Q with new interconnect – late 2011 @ LLNL; 2012 ANL & Julich

Aspects of performance - typical values in 2009/10

- Floating point (integer) performance: 2 or 4 per cycle
  - Flop/s = floating point operation per second
  - 2.4 GHz processors: 9.6 GFlop/s per core
- Memory latency: ~50 ns
- Memory bandwidth: ~10 GB/s per core
- Interconnect latency ~1-10 µs
- Network bandwidth: ~5-10 GB/s
- Disk access time ~ ms
- I/O bandwidth (disk) ~ Gigabit/s



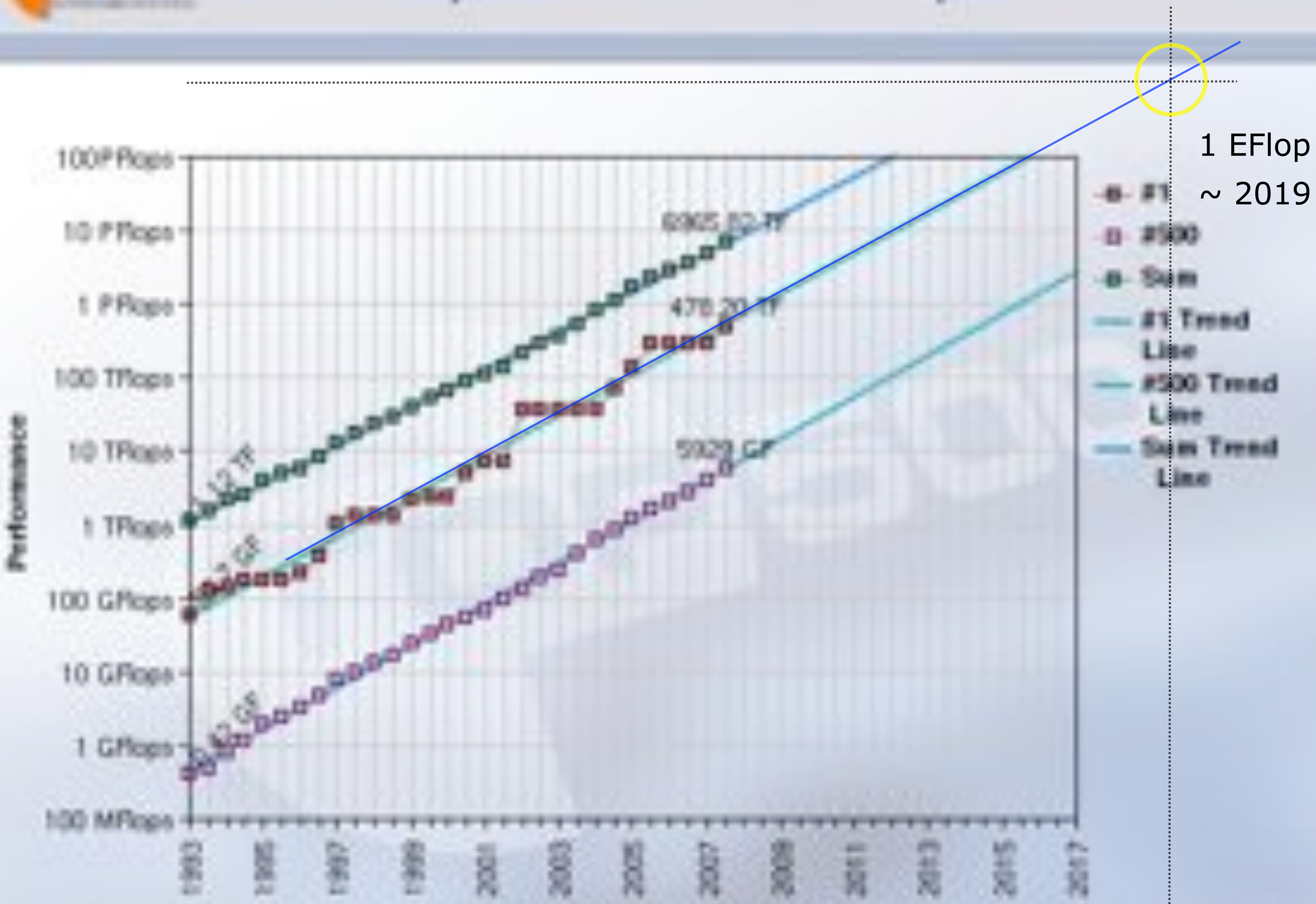
Cray XT5 node

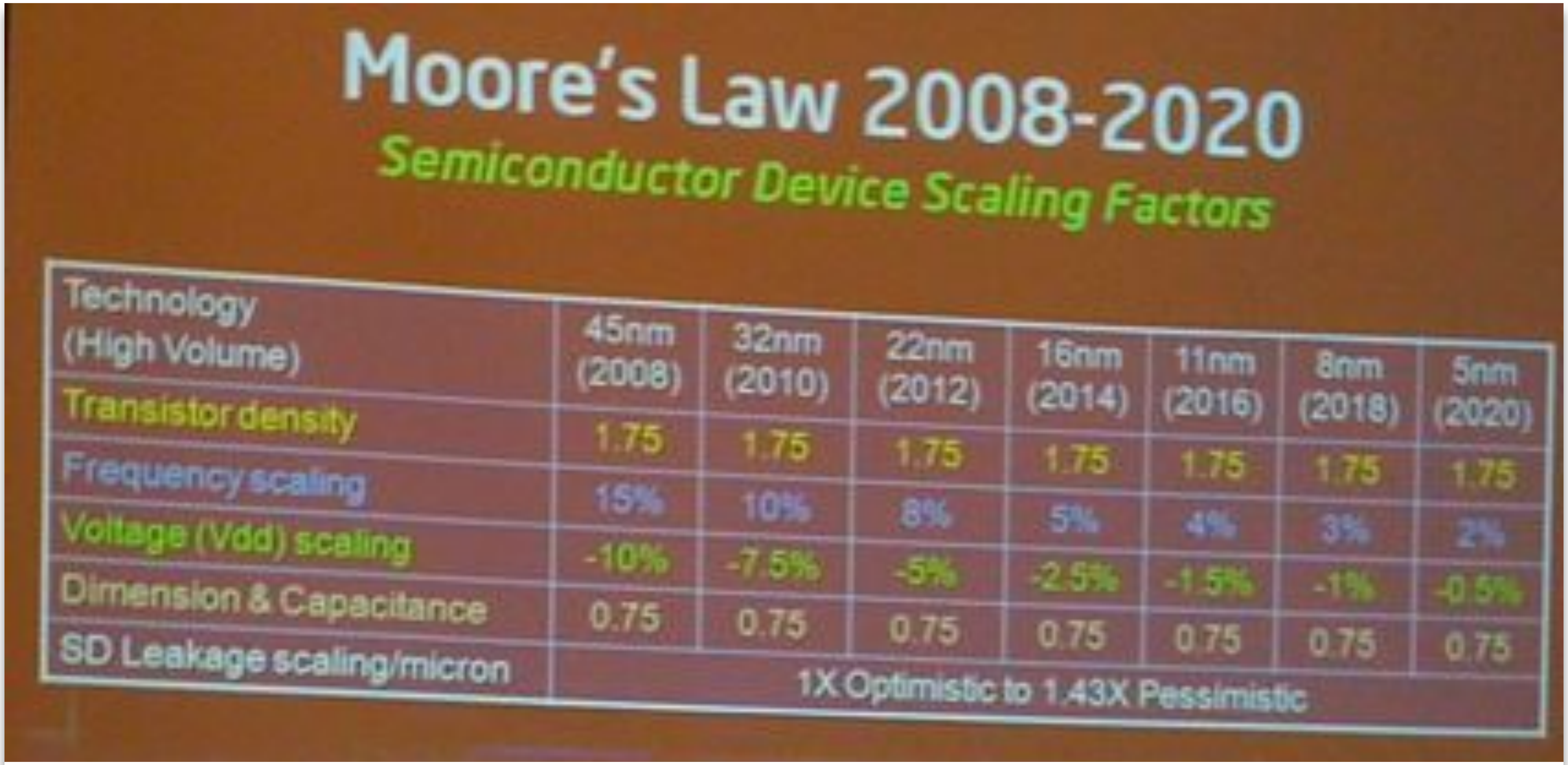
Summary: Brutal fact of modern HPC

- Mind boggling numbers of processing processing units
- Processor complexity (multi-core, heterogeneous, memory)
- Interconnect is a non-trivial part of the HPC system
- Parallel programming model characterized by memory model
  - Shared memory (OpenMP), distributed memory (MPI), data parallel (HPF)
  - Distributed shared memory (PGAS such as UPC, CAF, Global Array)
- Accessing memory is prohibitively expensive compared to the cost of floating point operations
  - 1960s: transistors were expensive, memory access was cheap
  - today: transitors are cheap, memory access is expensive

Key aspect of programming in HPC systems:
All about managing resources

Projected Performance Development





Moore's Law 2008-2020

Semiconductor Device Scaling Factors

Technology (High Volume)	45nm (2008)	32nm (2010)	22nm (2012)	16nm (2014)	11nm (2016)	8nm (2018)	5nm (2020)
Transistor density	1.75	1.75	1.75	1.75	1.75	1.75	1.75
Frequency scaling	15%	10%	8%	5%	4%	3%	2%
Voltage (Vdd) scaling	-10%	-7.5%	-5%	-2.5%	-1.5%	-1%	-0.5%
Dimension & Capacitance	0.75	0.75	0.75	0.75	0.75	0.75	0.75
SD Leakage scaling/micron	1X Optimistic to 1.43X Pessimistic						

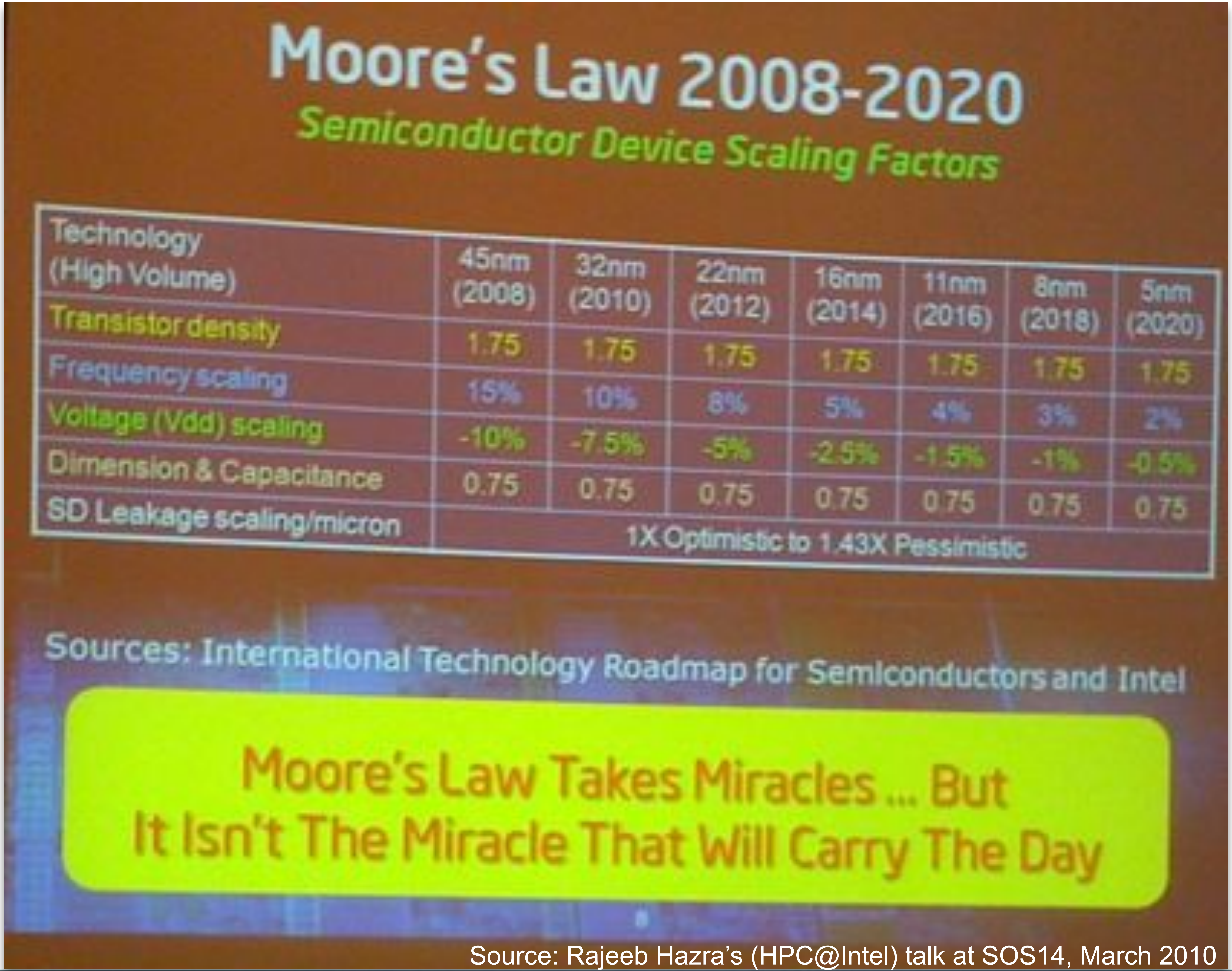
1000 fold increase in performance in 10 years:

> previously: double transistor density every 18 months = 100X in 10 years
>
> frequency increased

> now: "only" 1.75X transistor density every 2 years = 16X in 10 years
>
> frequency almost the same

Need to make up a factor 60 somewhere else

Source: Rajeeb Hazra's (HPC@Intel) talk at SOS14, March 2010



Moore's Law 2008-2020

Semiconductor Device Scaling Factors

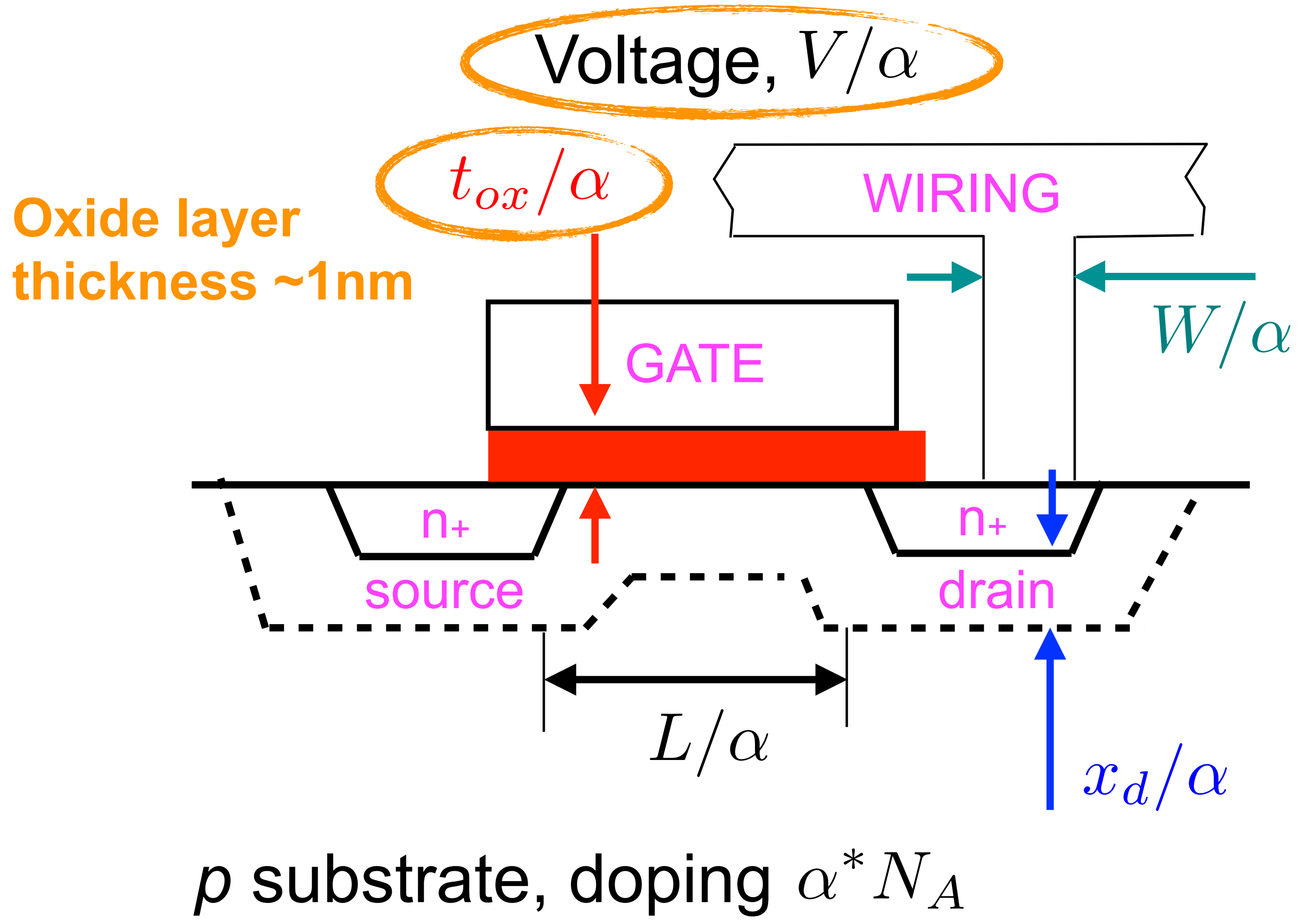
Technology (High Volume)	45nm (2008)	32nm (2010)	22nm (2012)	16nm (2014)	11nm (2016)	8nm (2018)	5nm (2020)
Transistor density	1.75	1.75	1.75	1.75	1.75	1.75	1.75
Frequency scaling	15%	10%	8%	5%	4%	3%	2%
Voltage (Vdd) scaling	-10%	-7.5%	-5%	-2.5%	-1.5%	-1%	-0.5%
Dimension & Capacitance	0.75	0.75	0.75	0.75	0.75	0.75	0.75
SD Leakage scaling/micron	1X Optimistic to 1.43X Pessimistic						

Sources: International Technology Roadmap for Semiconductors and Intel

Moore's Law Takes Miracles ... But It Isn't The Miracle That Will Carry The Day

Source: Rajeeb Hazra's (HPC@Intel) talk at SOS14, March 2010

Limits of CMOS scaling



SCALING

~~Voltage:~~	~~V/α~~
~~Oxide:~~	~~t_{ox}/α~~
Wire width:	W/α
Gate Width:	L/α
Diffusion:	x_d/α
Substrate:	$\alpha^* N_A$

CONSEQUENCE:

Higher density:	$\sim \alpha^2$
Higher speed:	$\sim \alpha$
Power/ckt:	$\sim 1/\alpha^2$
~~Power density:~~	~~$\sim$ constant~~

The power challenge today is a precursor of more physical limitations in scaling – atomic limit!

Opportunity for outsiders: A major disruption must happen before 2025

- Current CMOS technology will reach its physical limits by the end of the decade and will seise to scale
- The other physical limitations is the speed of light
  - Light travels 30cm in 1ns – this is several cycles!
  - Supercomputers can't just become larger as they did the past decade
- Enthusiasm for GPGPU and hybrid systems indicates that a change in architecture is happening
  - But this is still within current thinking and paradigm of digital computers

Huge opportunities for new materials, devices and condensed matter physics!