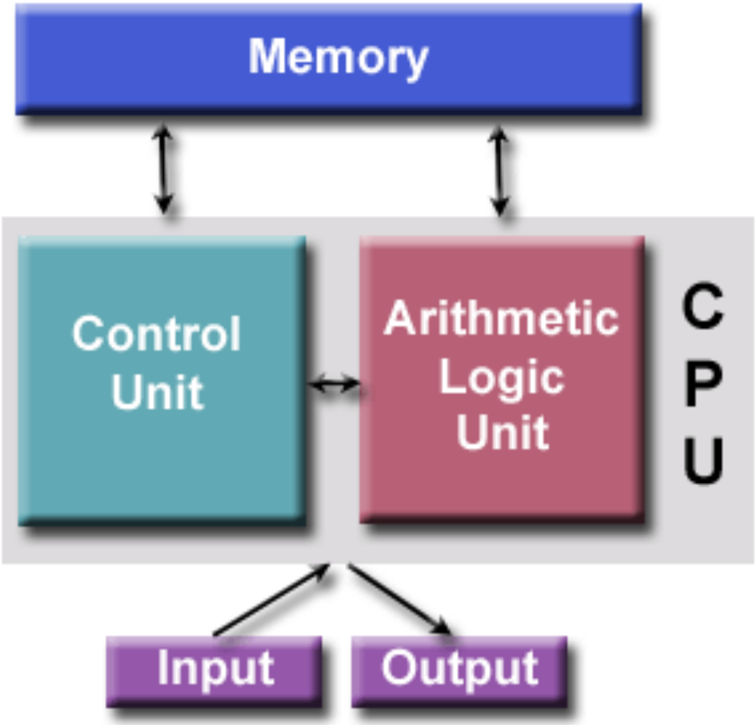
ATM 265, Spring 2019
Lecture 7
Parallelism and Supercomputing
April 22, 2019

Paul A. Ullrich (HH 251)
paullrich@ucdavis.edu

Slides are based on Richard Loft's talk on parallelism and supercomputing from the DCMIP workshop (2012 and 2016)

Von Neumann Architecture



This is the basic building block of modern computers

- **Memory** stores programs and data
  - Programs and data are interchangable
- **Control unit** sequentially controls arithmetic unit
- **Arithmetic unit (AMU)** does the calculations / mathematics (contains floating point unit, FPU)
- **Input/output** connects to the outside world (hard drive, tape drive, monitor, etc.)

Processor Trends

Why did we stop developing single-chip supercomputers?

Around 2000, chip designers realized…

Power wall: Clock speed couldn't continue to rise due to power density

Memory wall: Memory system speed couldn't keep up with processor speed

Instruction Level Parallelism (ILP) wall: Benefit of adding processor "smarts" reached point of diminishing returns

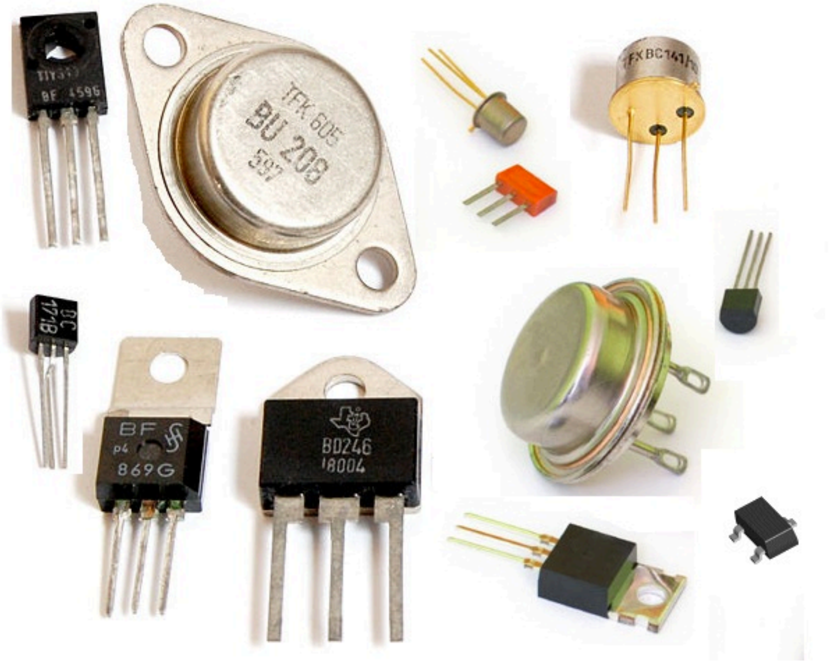
Solution: Break up monolithic single processor into multiple computing regions (referred to as **cores**)

Transistors

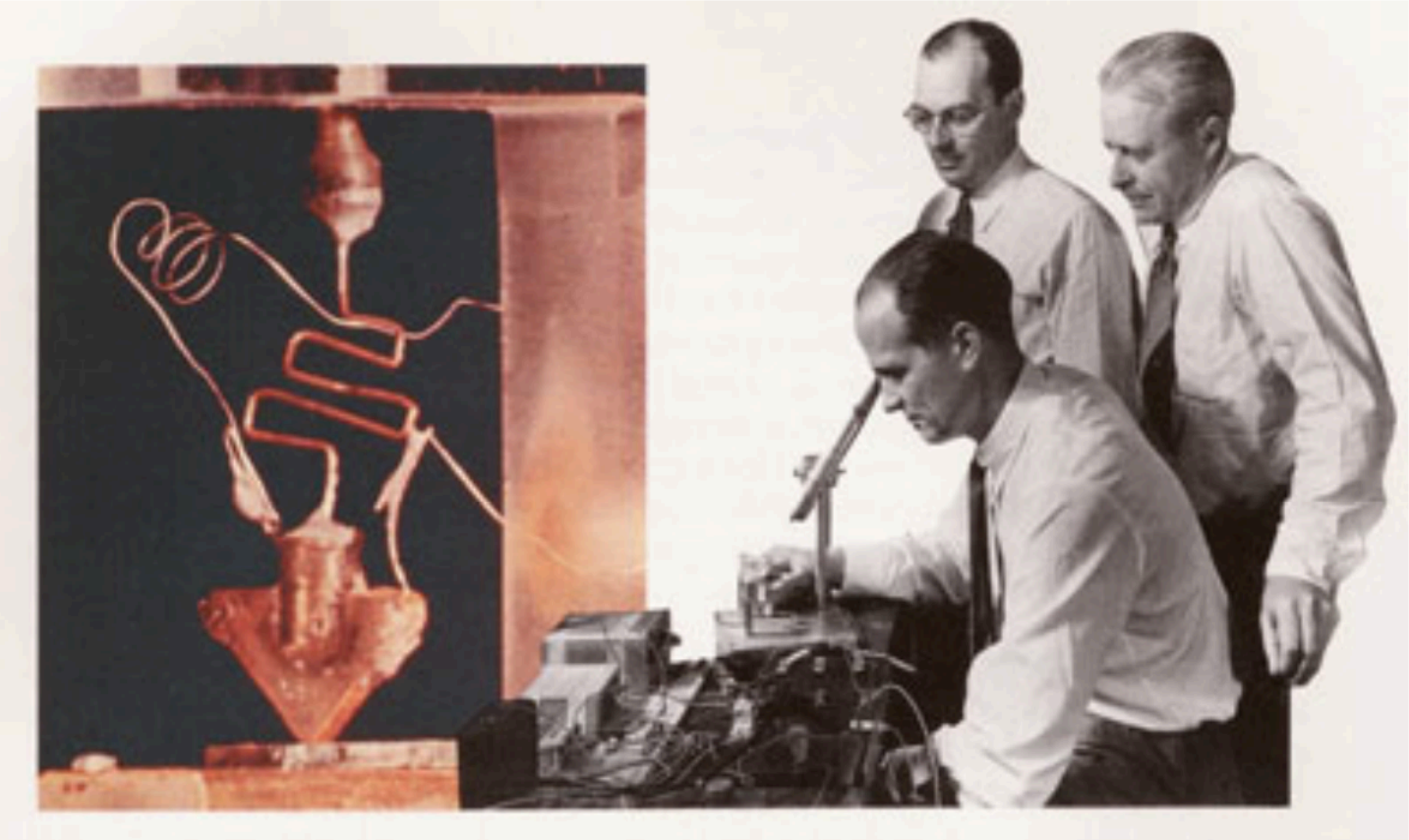
A semiconductor device used to amplify or switch electronic signals and electrical power.

A voltage or current applied to one pair of the transistor's terminals controls the current through another pair of terminals.

An AND or OR gate required 3 transistors. A NAND or NOR gate requires 2 transistors.



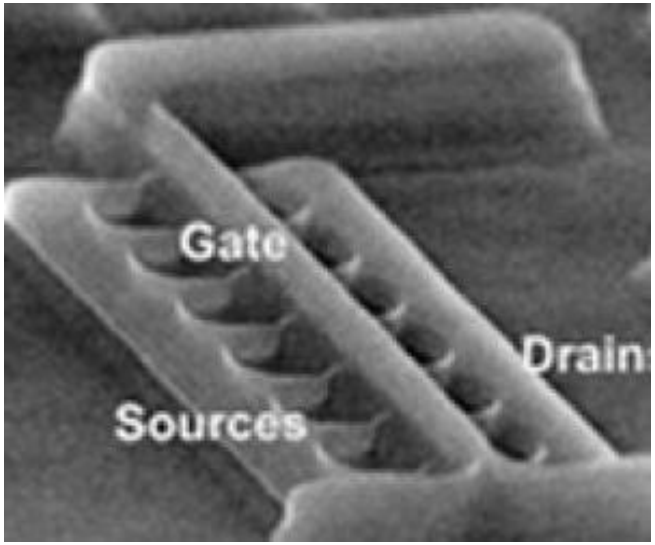
Transistors in 1947 (~5cm across)

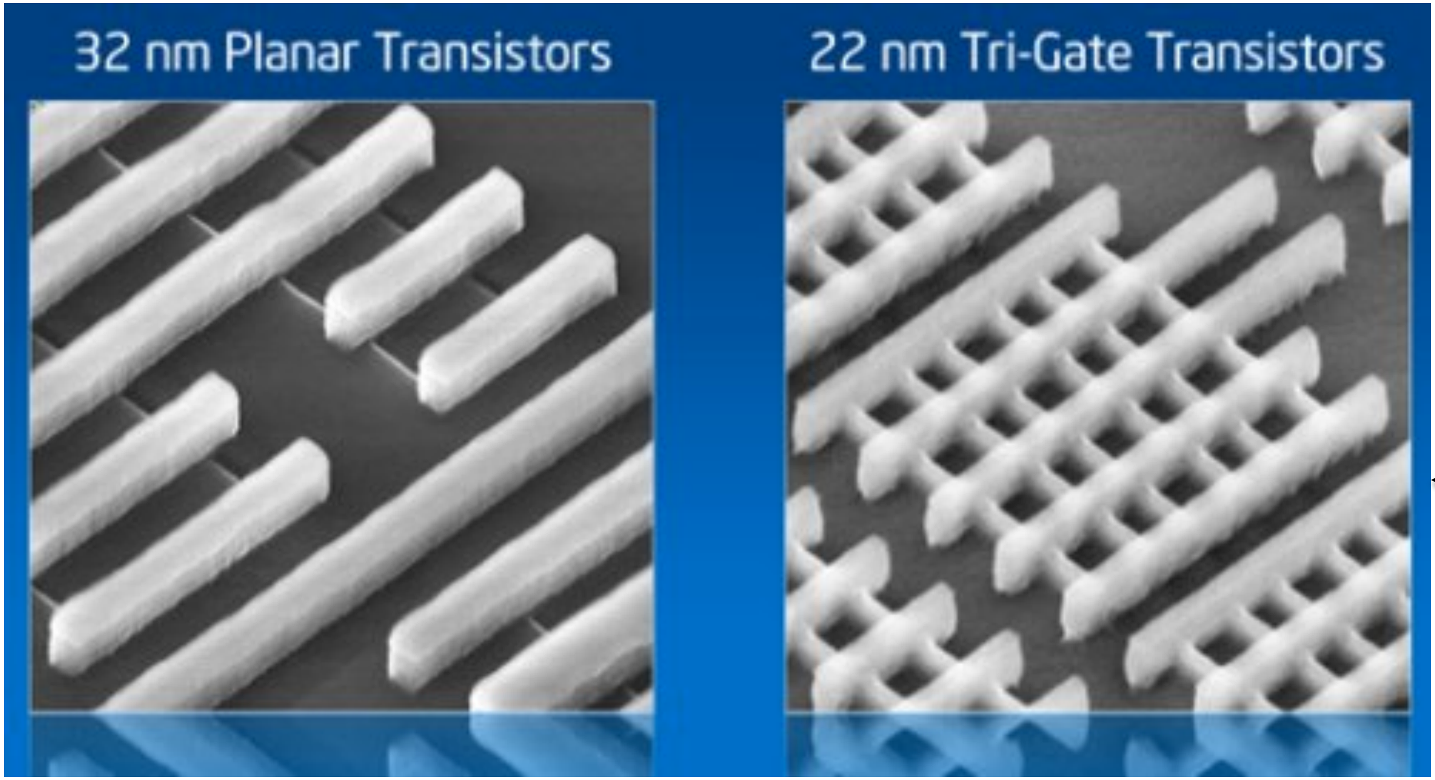


William Shockley (1910-1989), John Bardeen (1908-1991), Walter Brattain (1902-1987)

Modern Transistors (~200 atoms across)

So small that quantum tunneling is a problem!

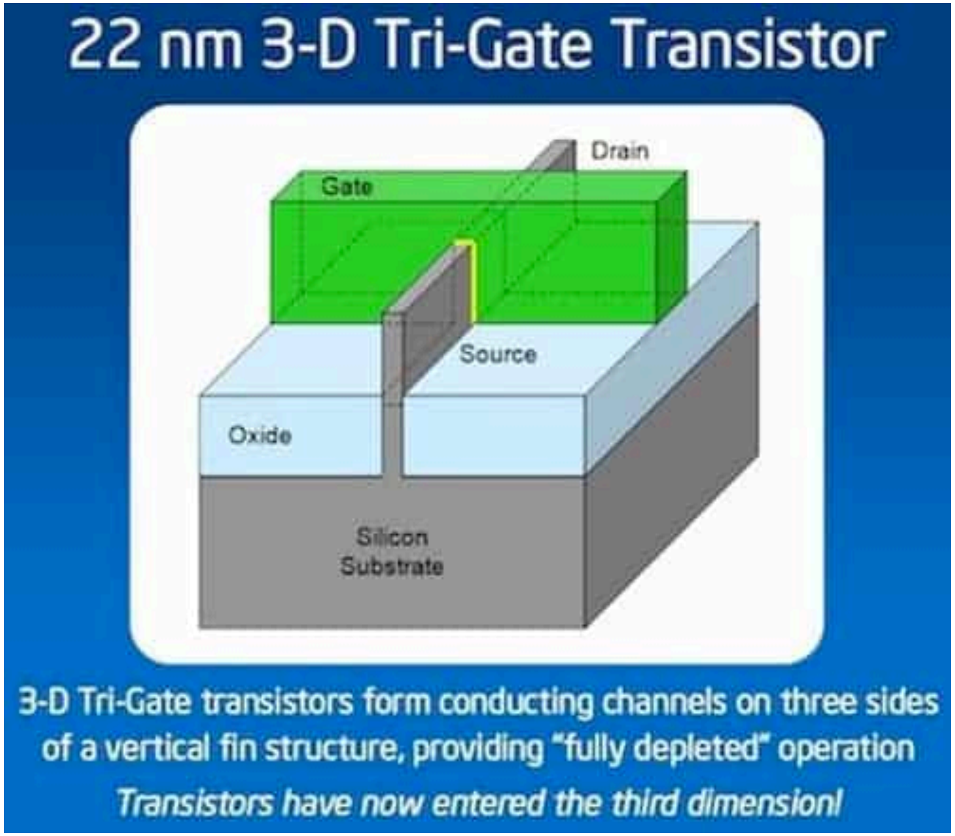



32 nm Planar Transistors

22 nm Tri-Gate Transistors

Intel Haswell Processor

Modern Transistors (~200 atoms across)

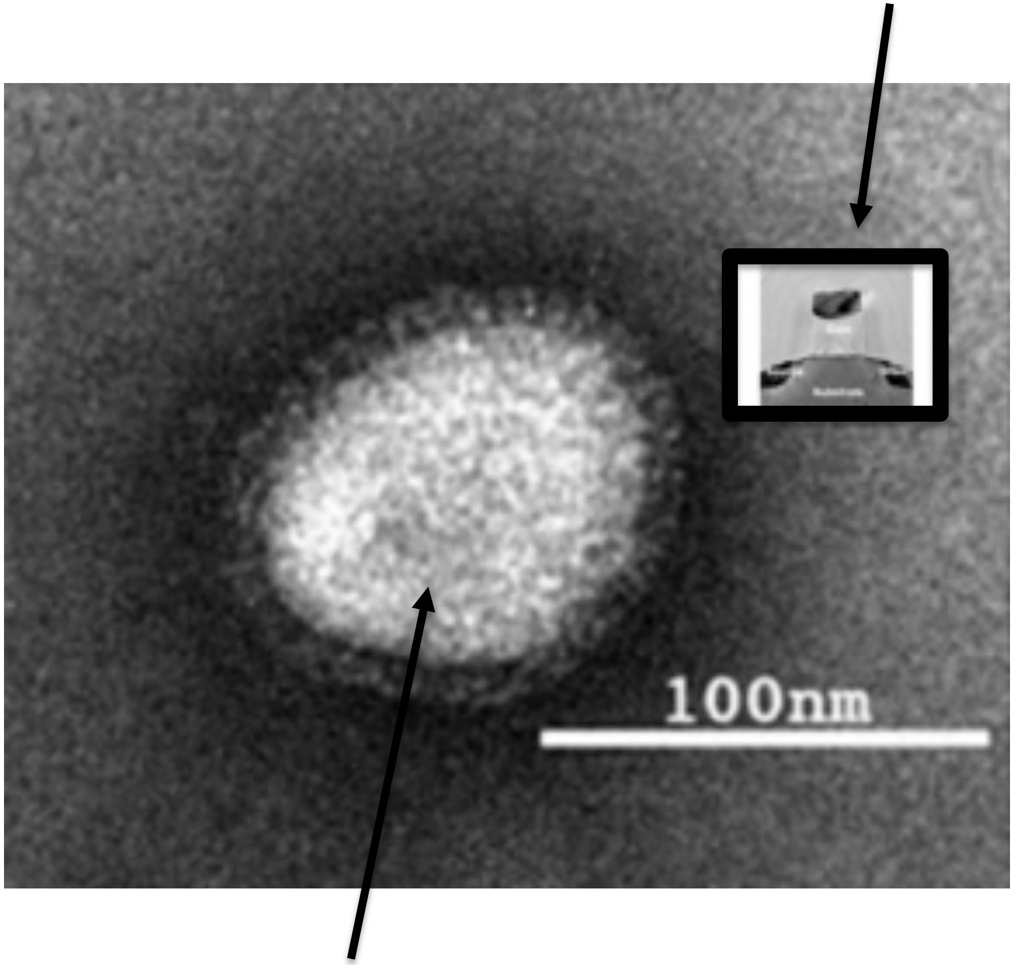


22 nm 3-D Tri-Gate Transistor

3-D Tri-Gate transistors form conducting channels on three sides of a vertical fin structure, providing "fully depleted" operation

Transistors have now entered the third dimension!

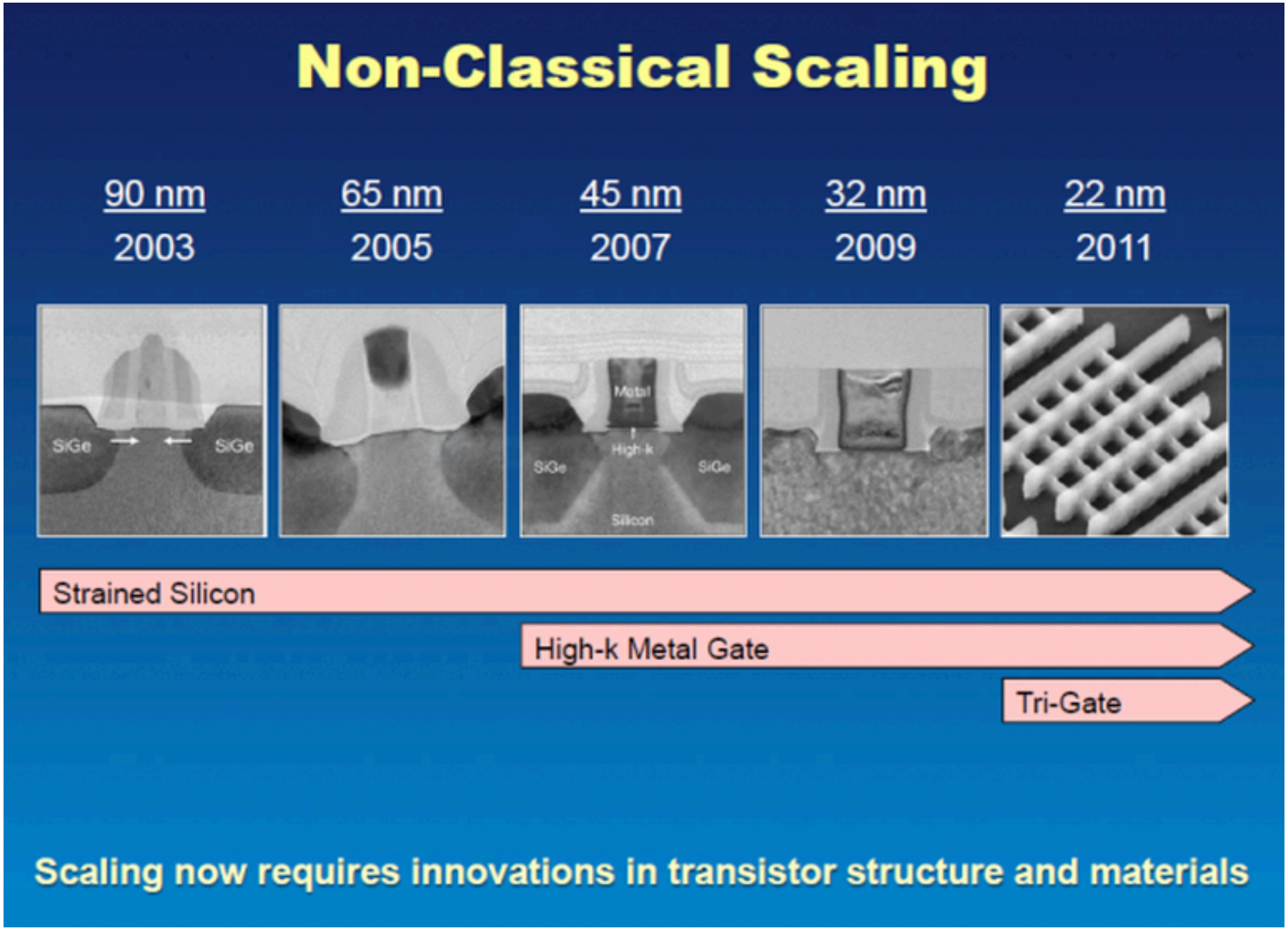
Modern Transistors



Transistor (22nm)

Influenza Virus

Modern Transistors



Non-Classical Scaling

90 nm	65 nm	45 nm	32 nm	22 nm
2003	2005	2007	2009	2011

SiGe
SiGe
Metal
High-k
SiGe
SiGe
Silicon

Strained Silicon

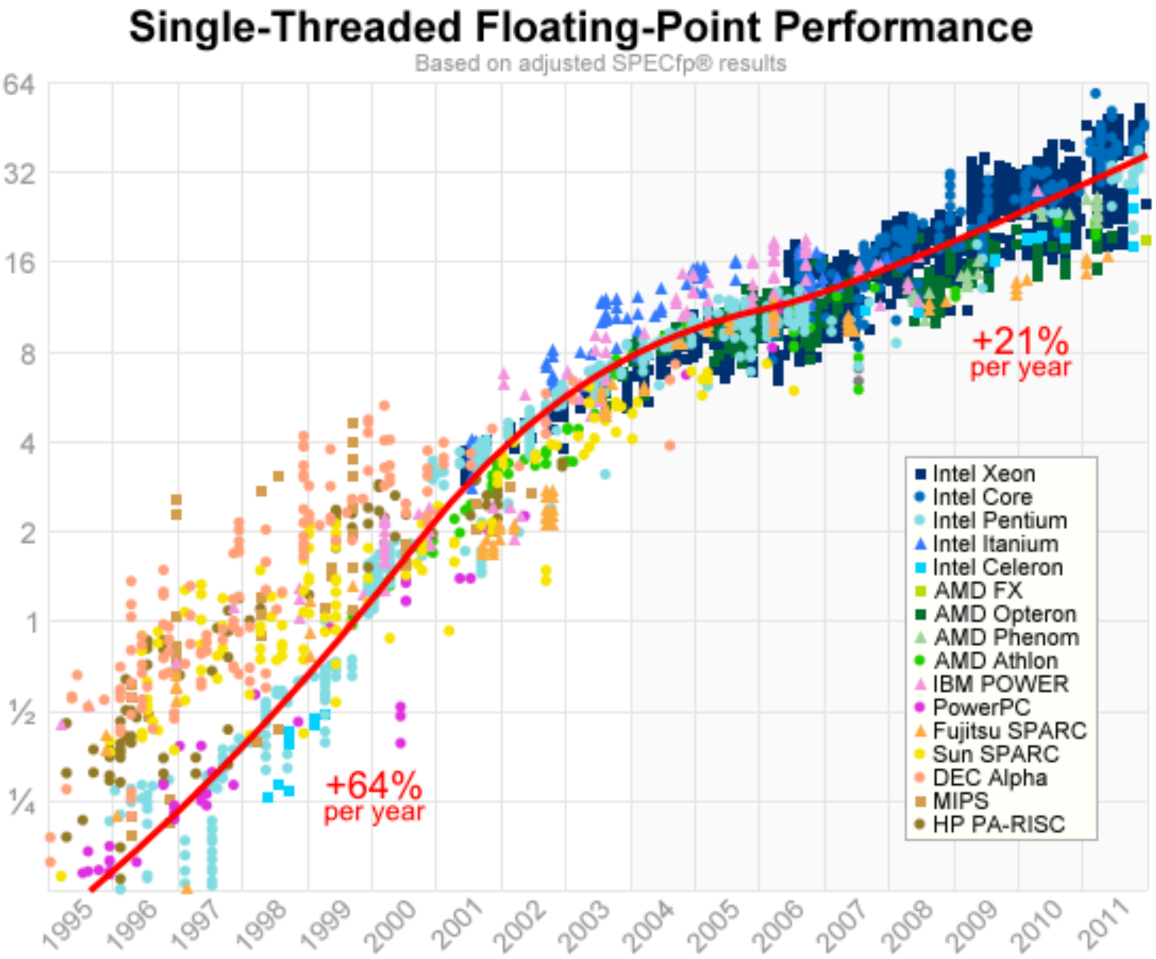
High-k Metal Gate

Tri-Gate

Scaling now requires innovations in transistor structure and materials

Processor Trends

Clock speeds stopped improving and single thread improvements slowed.



Moore's Law

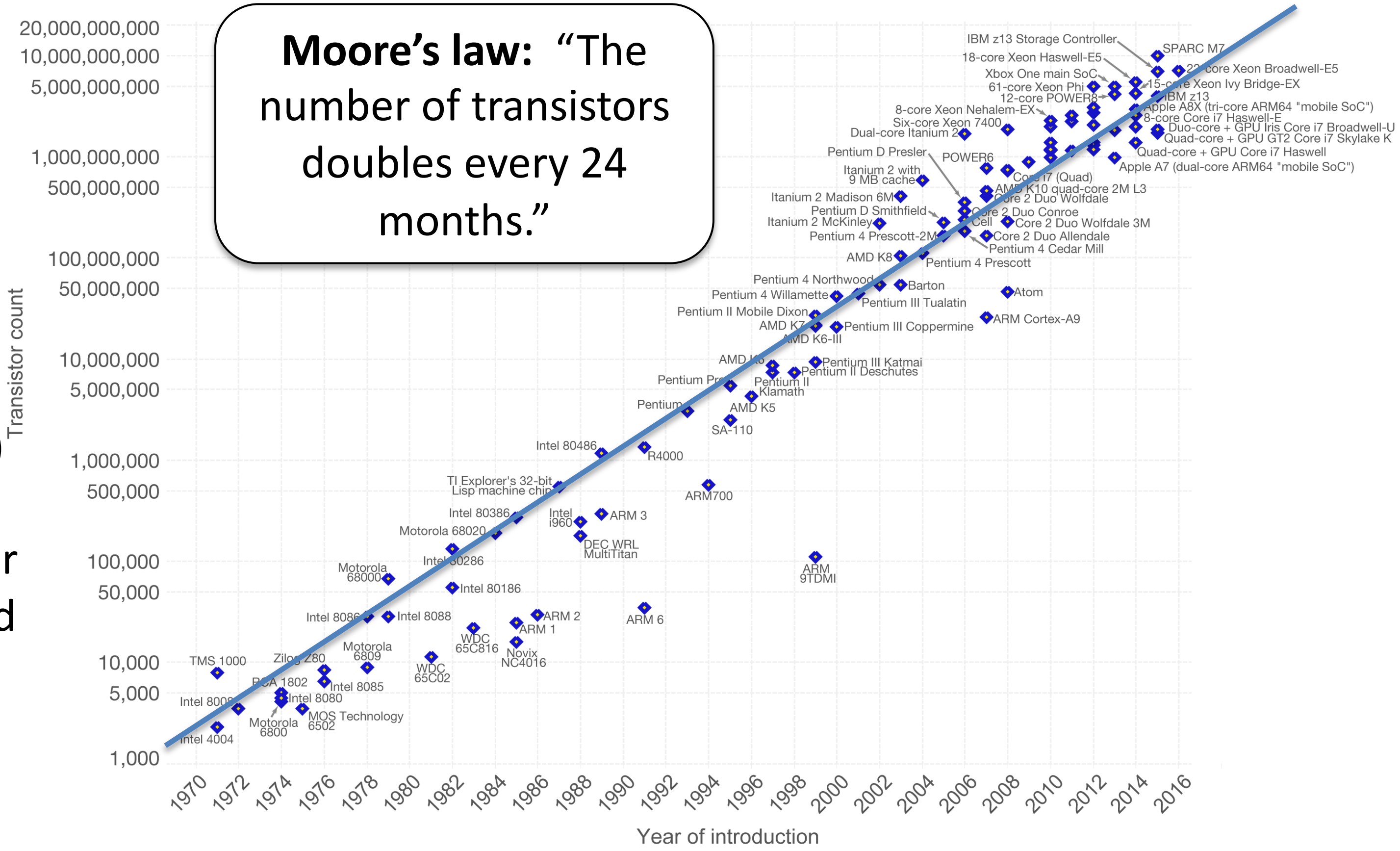
Not a physical law; more of a self-fulfilling prophesy.

Law **does not state** "performance doubles every 18 months." This statement is due to House (House's Law) This "law" no longer holds since processor clock speeds stopped increasing.

Moore's Law – The number of transistors on integrated circuit chips (1971-2016)

Our World in Data

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore's law.

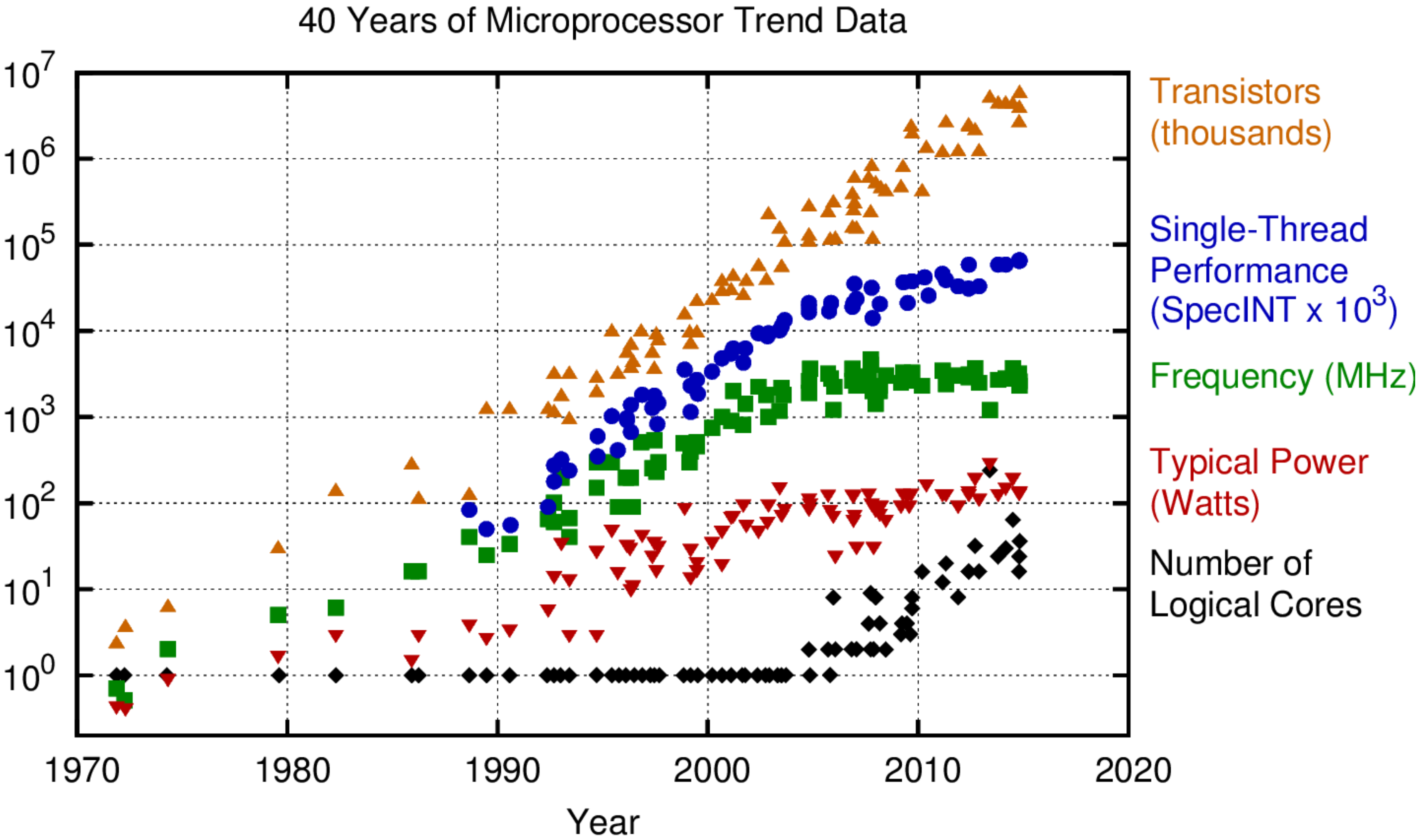


Moore's law: "The number of transistors doubles every 24 months."

Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Source: http://letshavetheconversation.blogspot.com

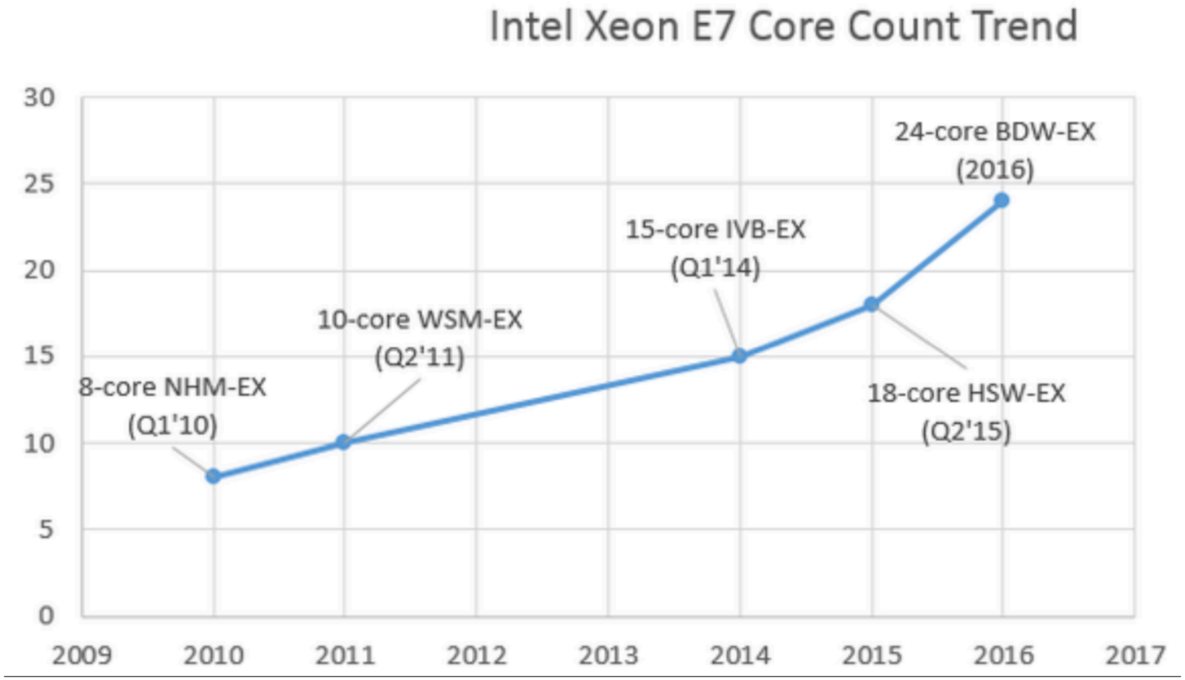
Processor Trends



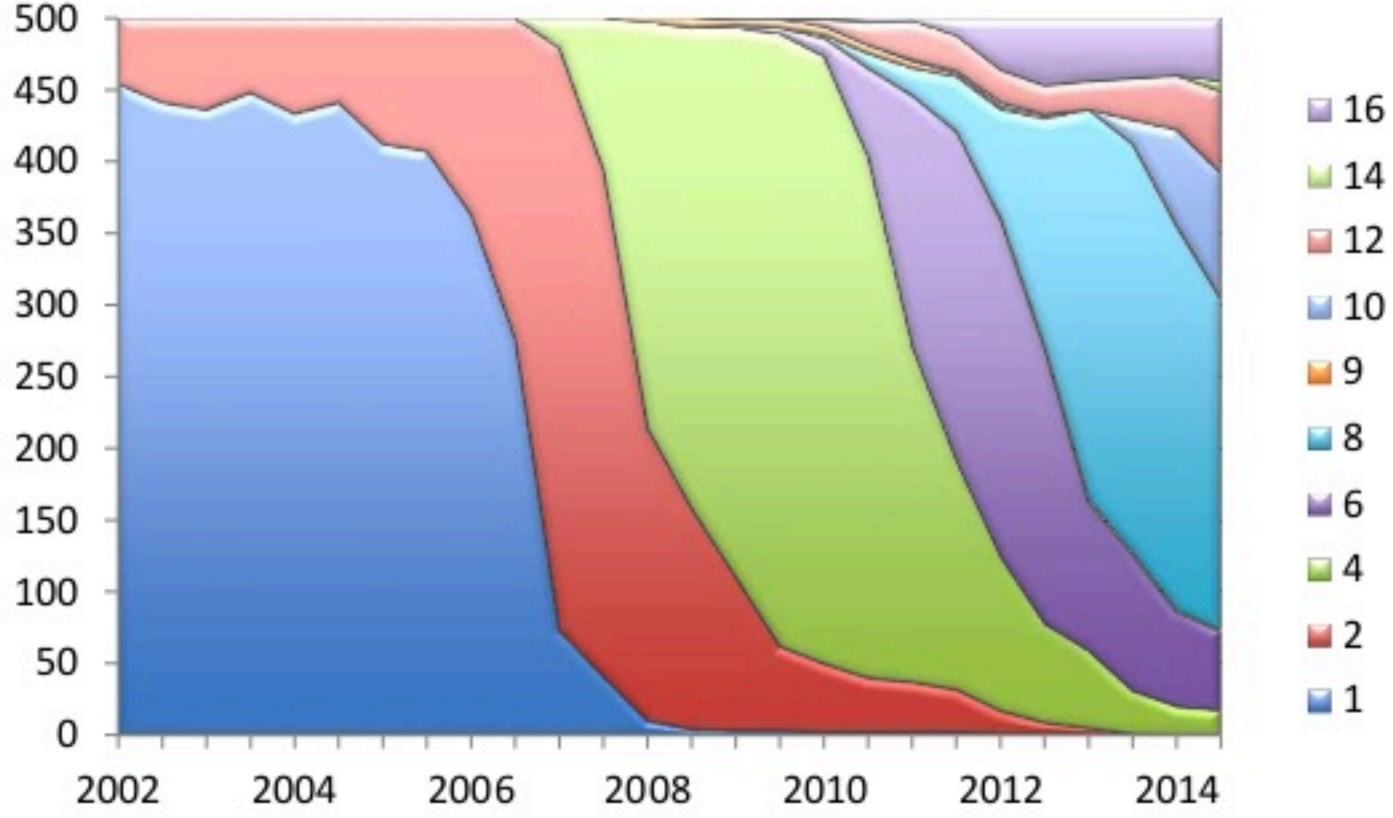
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

Many-core Evolutionary Approach

- Hold clock speed approximately constant
- Hold core complexity approximately constant
- Gradually add more cores on the die as transistors shrink
- Gradually add Single-Instruction Multi-Data (SIMD) features (vector units)



Cores per Socket on Top500 Machines



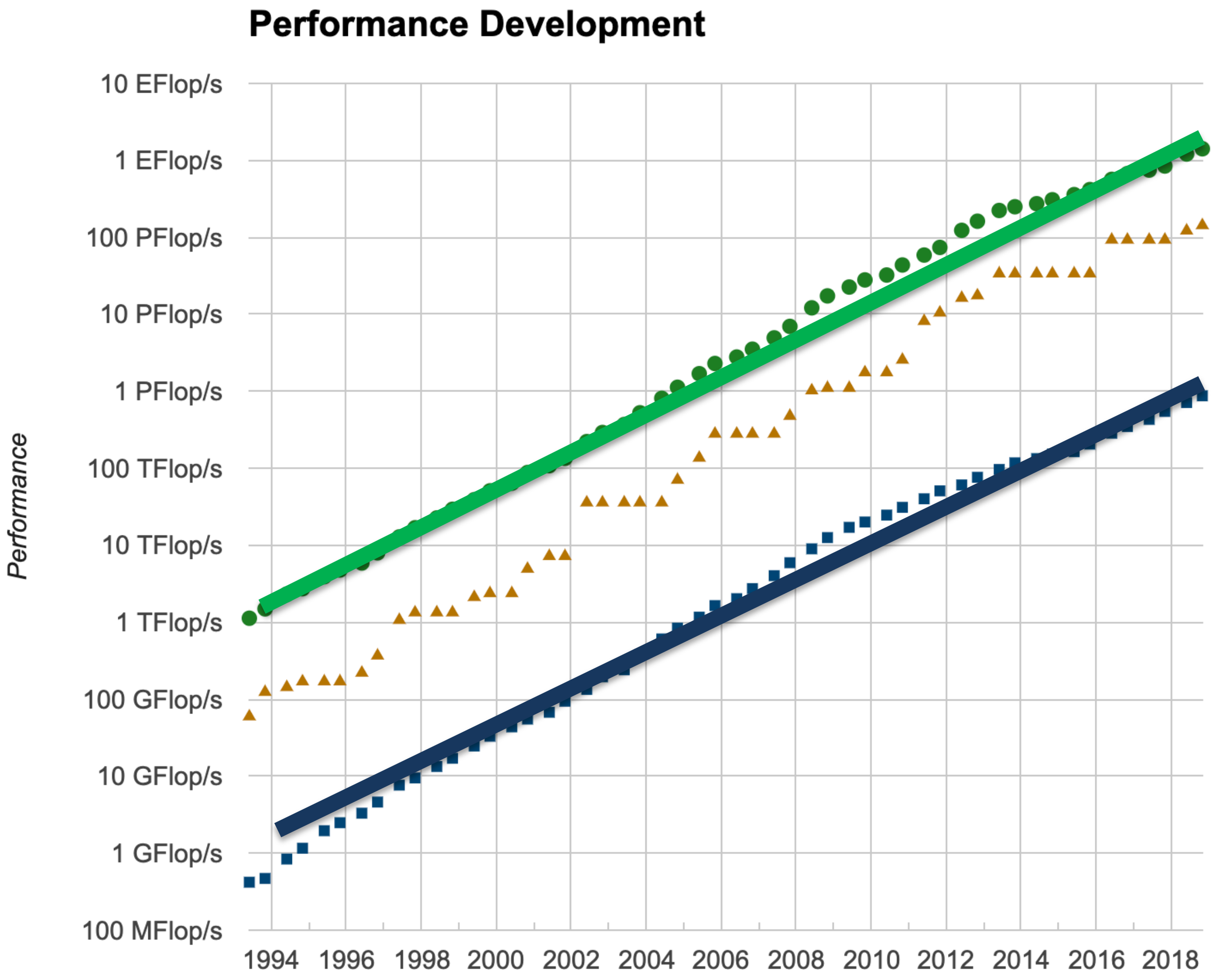
https://www.top500.org/statistics/overtime/

Source: https://www.computer.org/cms/Computer.org/dl/mags/co/2015/11/figures/mco20151100423.gif

Computing Power

The computing power of modern supercomputing systems are growing at an exponential pace.

This motivates the need for atmospheric models that can harness these large-scale parallel systems.



Note logarithmic scale → Exponential growth!

Computing Power

The computing power of modern supercomputing systems are growing at an exponential pace.

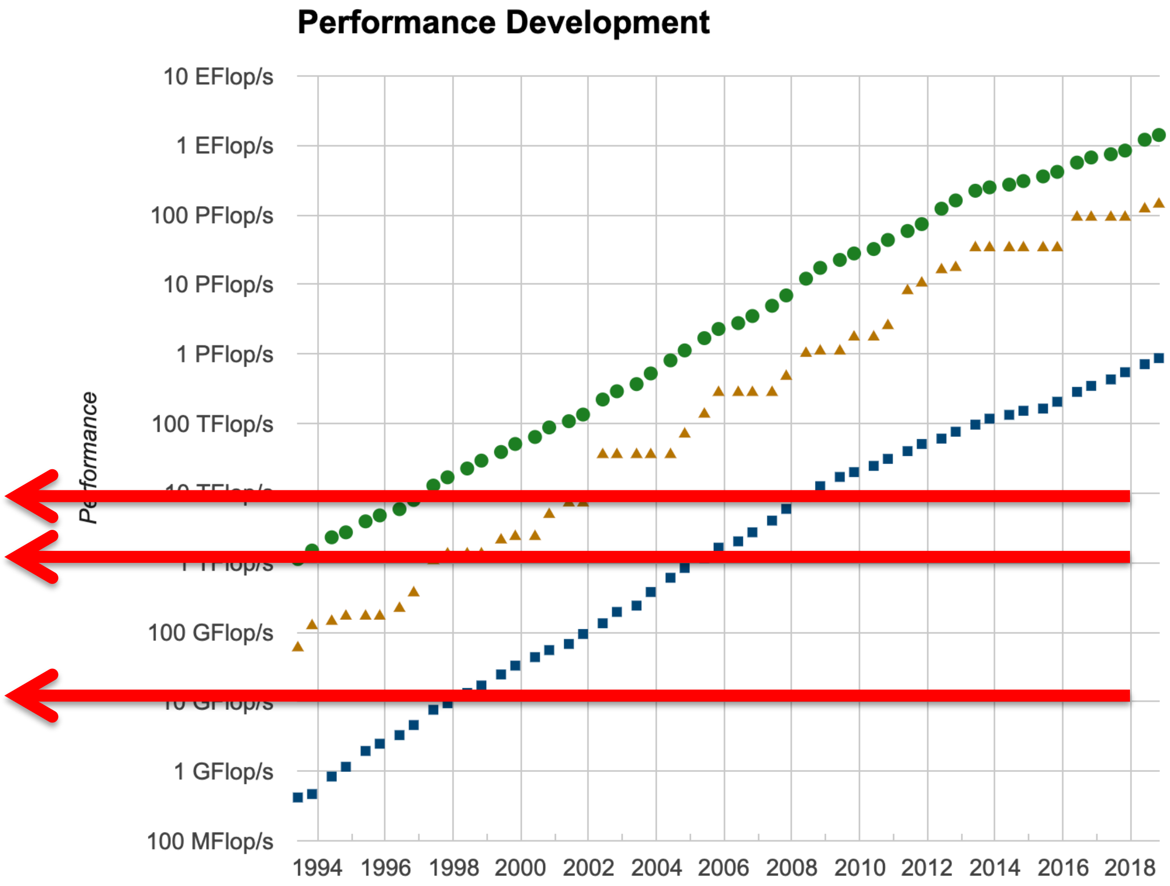
<u>**GPUs**</u>

Nvidia Titan X (Pascal)

GeForce GTX 960M

<u>**CPUs**</u>

Intel(R) Core(TM) i7-4870HQ (this MacBook)



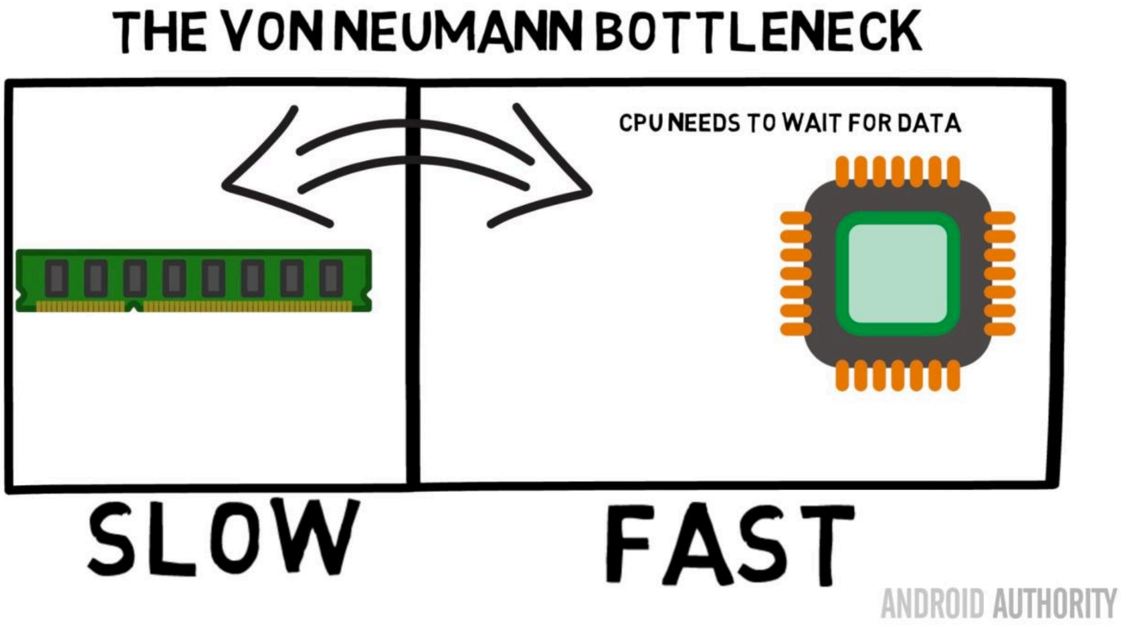
Note logarithmic scale → Exponential growth!

Problems: The Memory Wall

Although processors have marched forward, memory access times have not kept pace.

This has led to a memory access bottleneck that will only get worse into the future.

Need more parallelism with fewer memory accesses and more computation (FLOPs are free model).

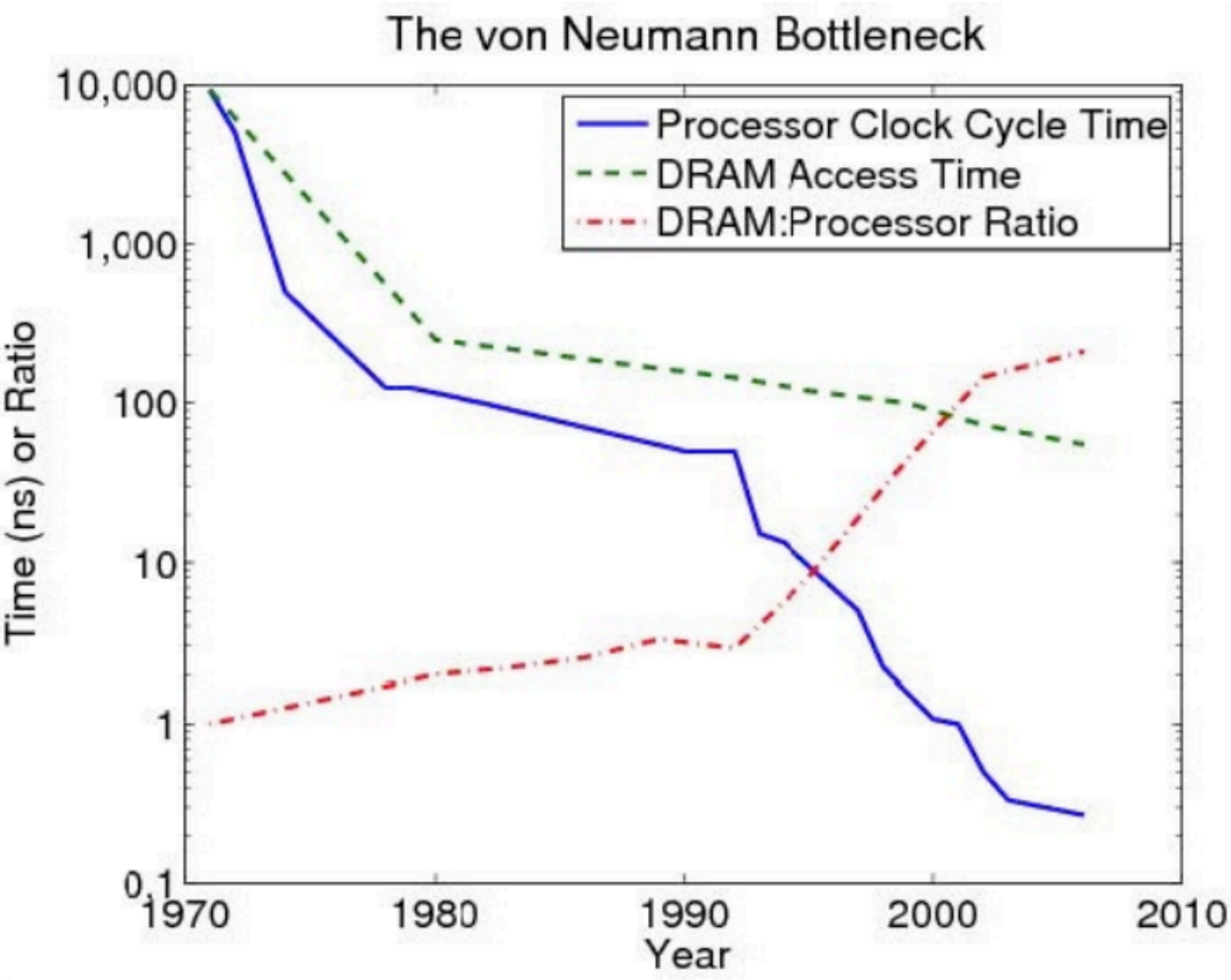


Problems: The Memory Wall

Although processors have marched forward, memory access times have not kept pace.

This has led to a memory access bottleneck that will only get worse into the future.

Need more parallelism with fewer memory accesses and more computation (FLOPs are free model).



Flynn’s Classical Taxonomy

SISD

Single Instruction, Single Data

SIMD

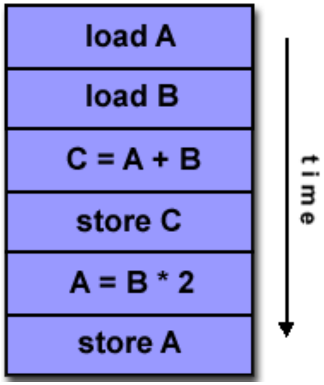
Single Instruction, Multiple Data

~~**MISD**~~

~~Multiple Instruction~~, Single Data

MIMD

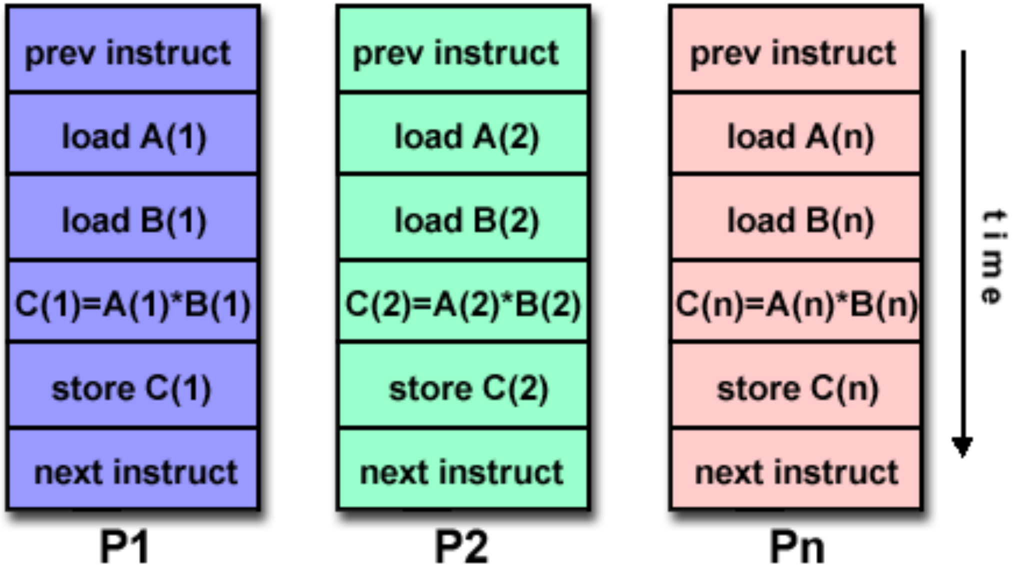
Multiple Instruction, Multiple Data



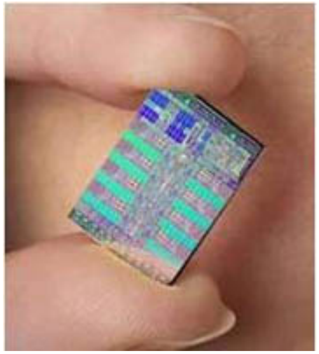
SISD (Single Instruction Single Data)

Typical serial (non-parallel) computer

Flynn's Classical Taxonomy

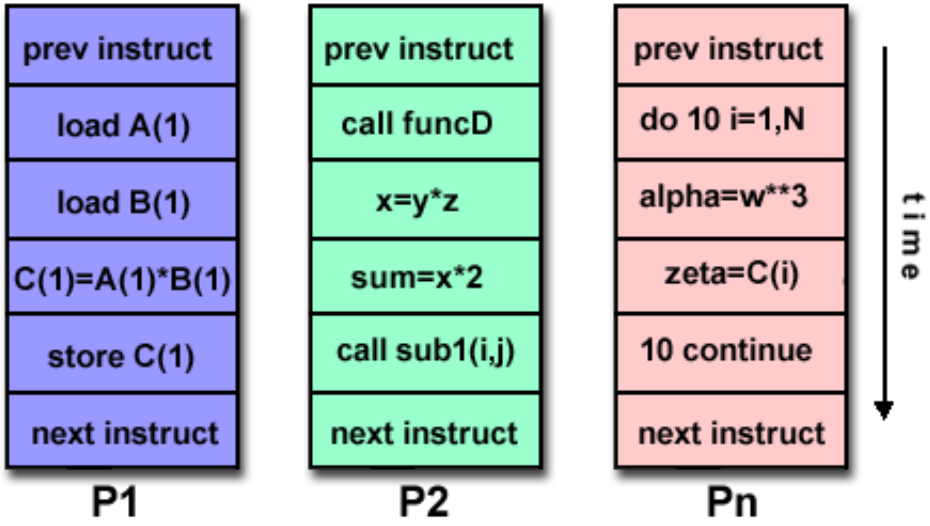


SIMD (Single Instruction Single Data) All processing units execute the same instruction over one cycle



Example: Vector supercomputing systems, Cell processors (Graphical Processing Units)

Flynn's Classical Taxonomy

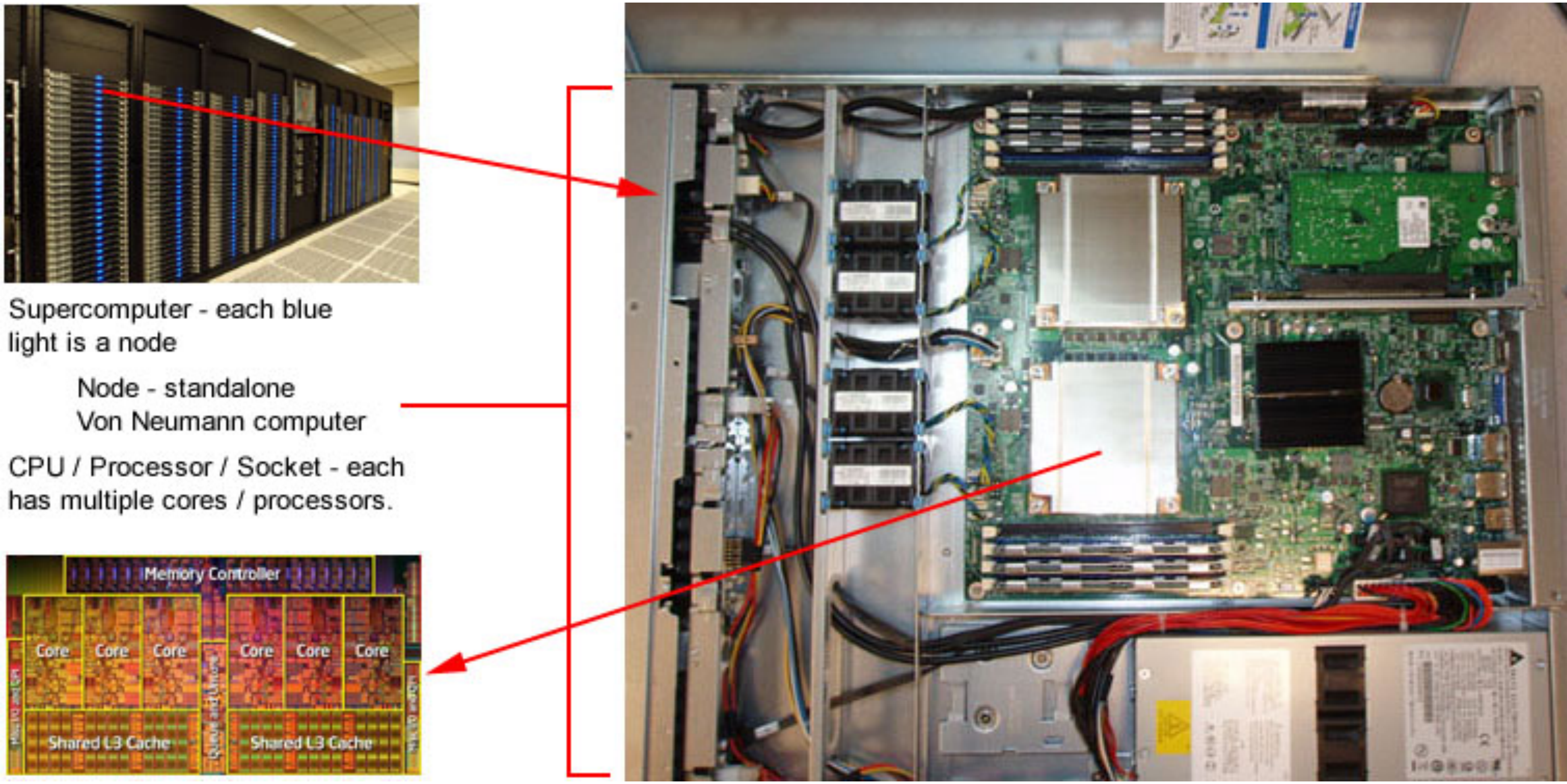


MIMD (Multiple Instruction Multiple Data)
Most general type of parallel computer
Every processor can execute a different instruction stream. Every processor may be working on a different data.

Examples: Yellowstone, Agri, etc.

MIMD systems can have SIMD sub-components as well.

Terminology



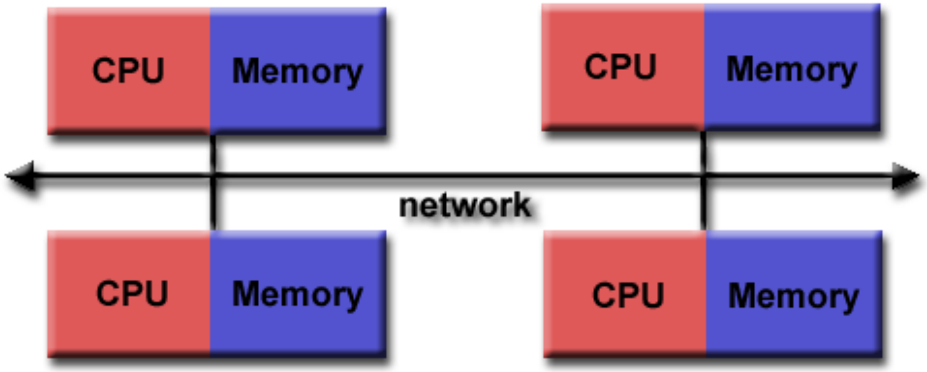
Supercomputer - each blue light is a node

Node - standalone Von Neumann computer

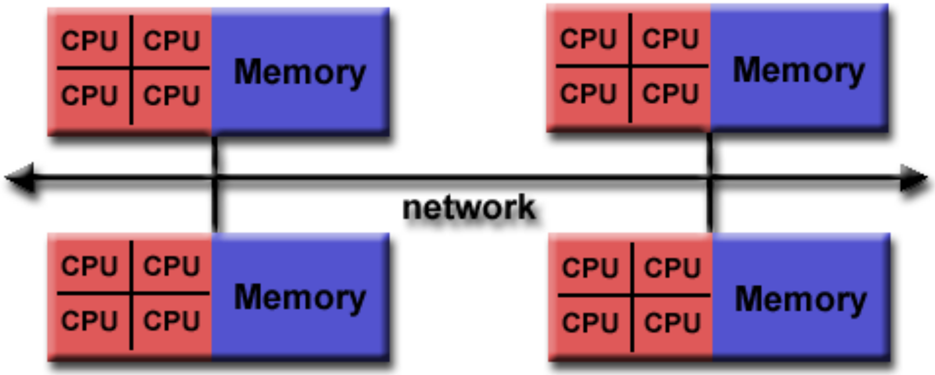
CPU / Processor / Socket - each has multiple cores / processors.

Computer Architecture

Distributed memory systems have memory allocated for each CPU.

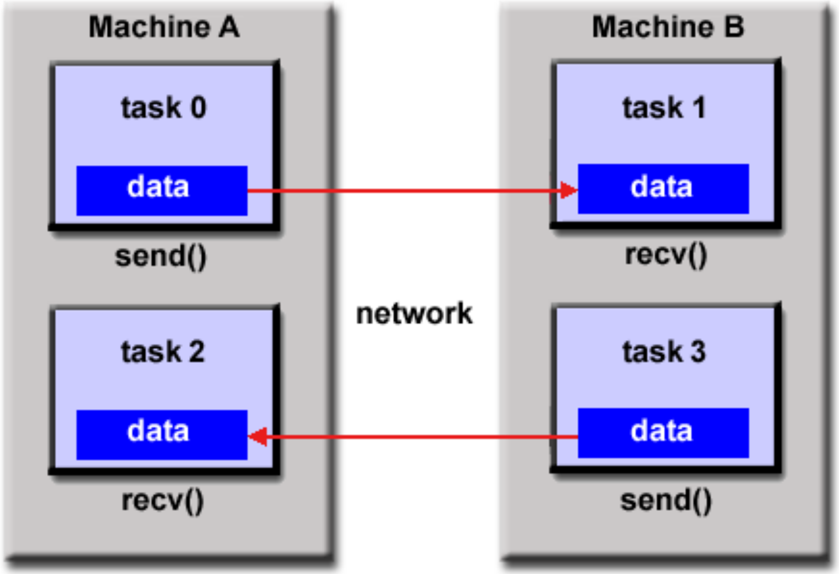


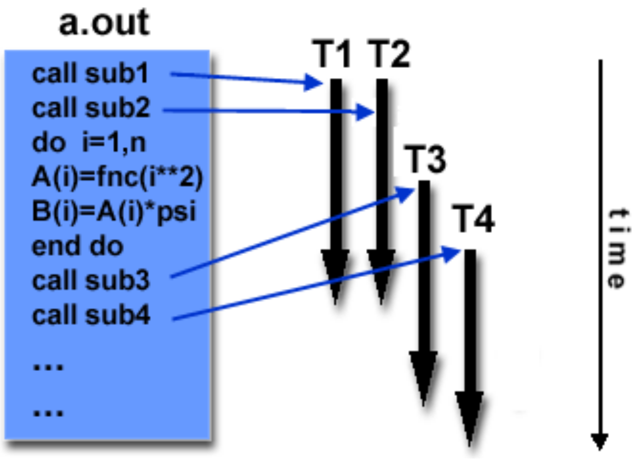
Most modern supercomputers use a **hybrid distributed-shared memory** structure, with blocks of memory shared between local CPUs.



Programming Models

Message passing model (for example, Message Passing Interface / MPI). Communication is via messages.

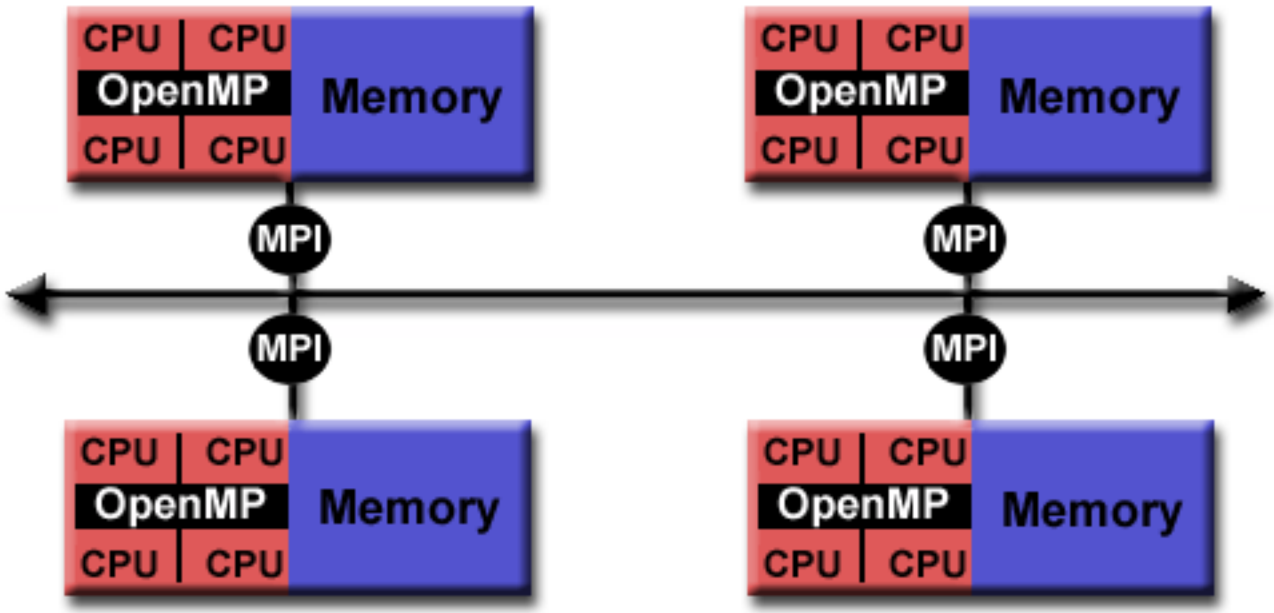




Threads model (for example, OpenMP) This model is for shared memory programming. Multiple concurrent execution paths are spread among processors (for example iterations in a loop)

Programming Models

The **hybrid model** combines MPI and OpenMP implementations to deal with the hybrid distributed / shared memory architecture.



What makes parallel programs SLOWER?

Starvation

- Underutilization of available compute power due to inadequate distribution of work or heterogeneities in the system.

Latency

- Delays during communication or I/O caused by physical hardware limitations.

Overhead of Parallelization

- The costs of managing parallel computation and synchronizing resources.

Waiting for Contention Resolution

- Delays due to lack of availability of a shared resource.

Energy Conservation Costs

- Resource constraints, heterogeneities and complexity caused by power management.

Resiliency Costs

- Down time due to hardware failure and the overheads associated with failure-tolerance.

Source: albrecht-durer.org

Alternative Programming Models

The development of alternative programming models has been a major focus in computer science, in order to devise means to expose greater parallelism.

High-Level Libraries

i.e. HPX, Legion, PaRSEC

Threading Libraries

i.e. PPL

Global Address Spaces

i.e. GASNet, UPC++

Data Models

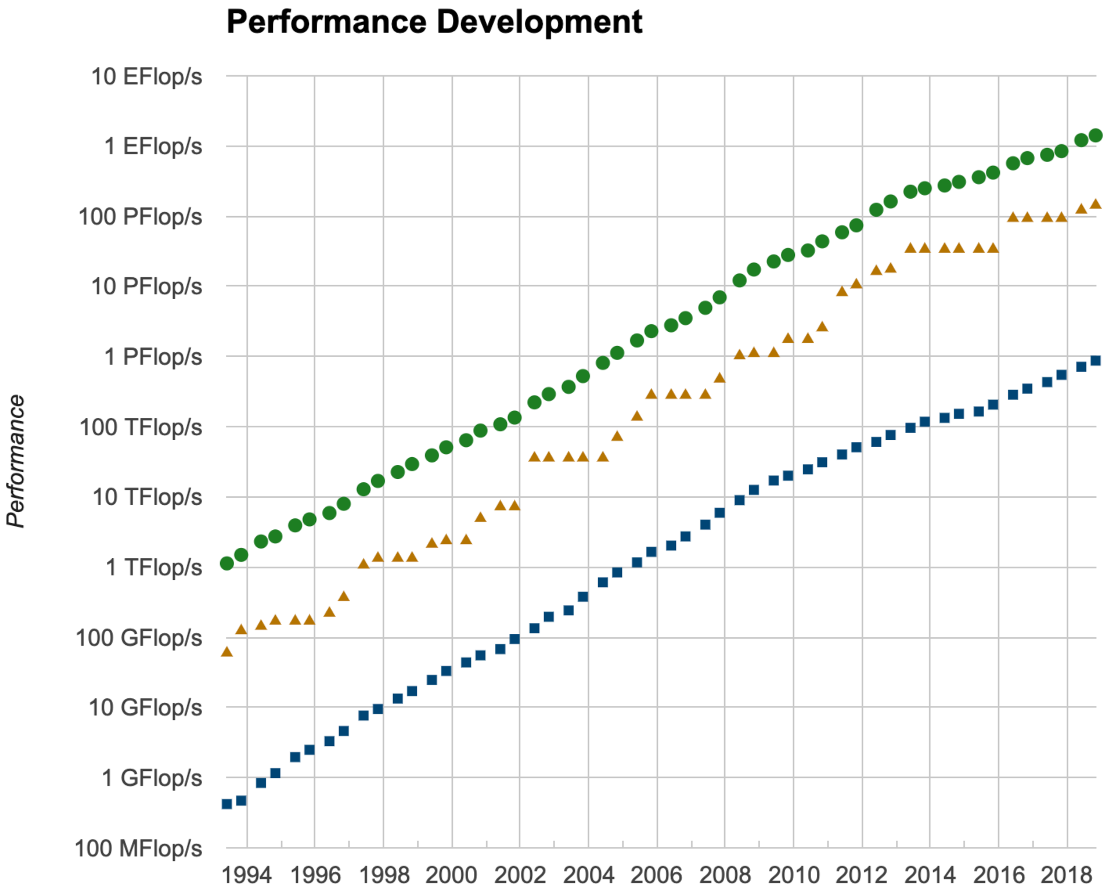
i.e. OCR, Kokkos

Computing Power vs. Resolution

Computational power doubles approximately every 1.2 years.

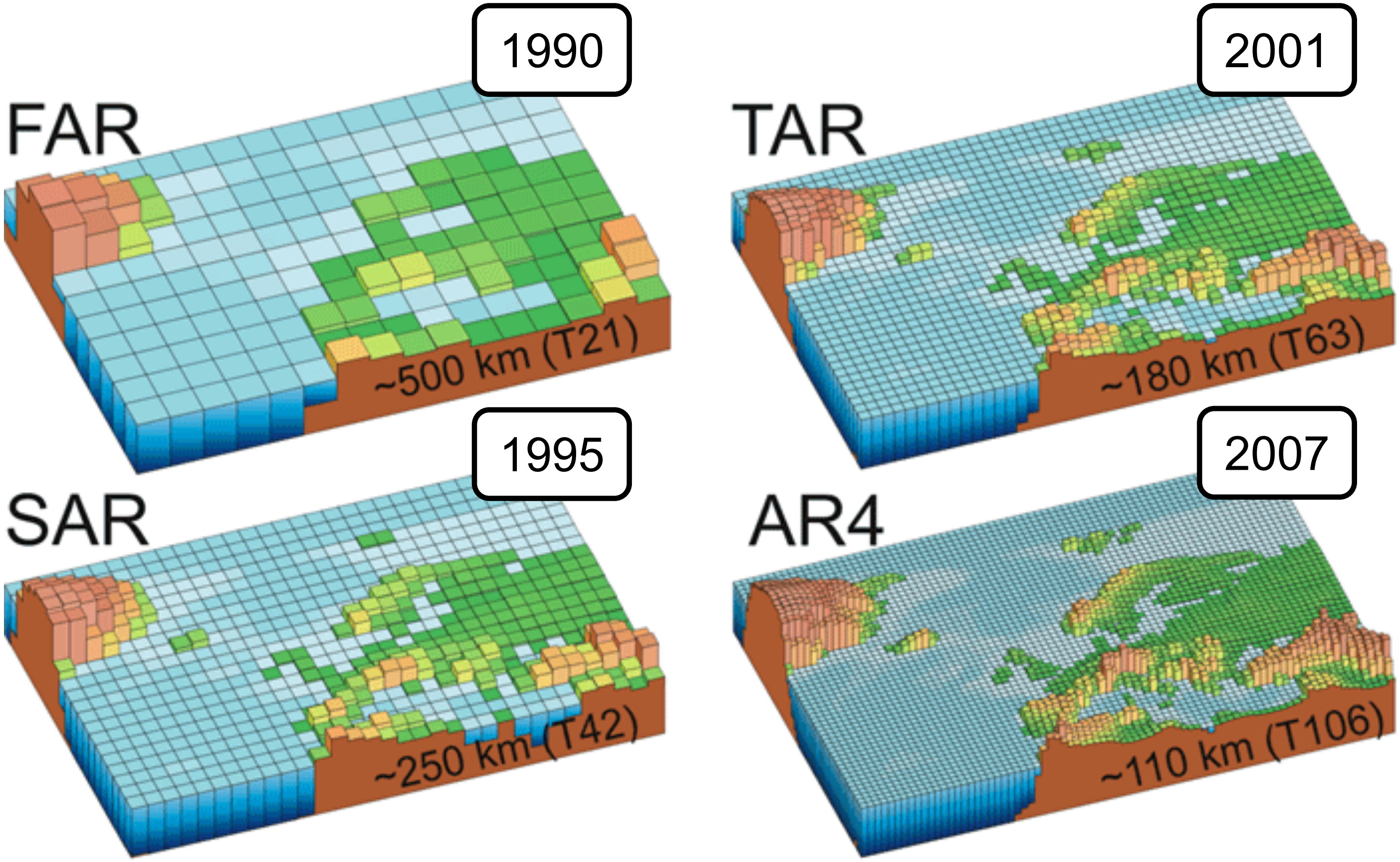
To obtain a factor of 2 horizontal refinement, numerical models require 8x the computational power.

Doubling of horizontal resolution every 3.6 years?



In reality slower due to increased vertical resolution, more tracers and more complicated parameterizations

Climate Model Resolution



AR5 (2013) included some model simulations at ~50km, but most runs were at 110km.

AR6 (2019) will rely on CMIP6 runs, which include many runs at 25-50km global resolution.

Next-Generation Modeling

Locality and grid uniformity

- Key to achieving scalable applications
- Local refinement and global solvers (implicit methods) are problematic
- Balancing computational work equally over multiple processors is difficult

Flexible expression of parallelism

- Hybrid MPI and OpenMP structure
- Easy to extend to SIMD co-processors such as GPUs

Small memory footprint per thread

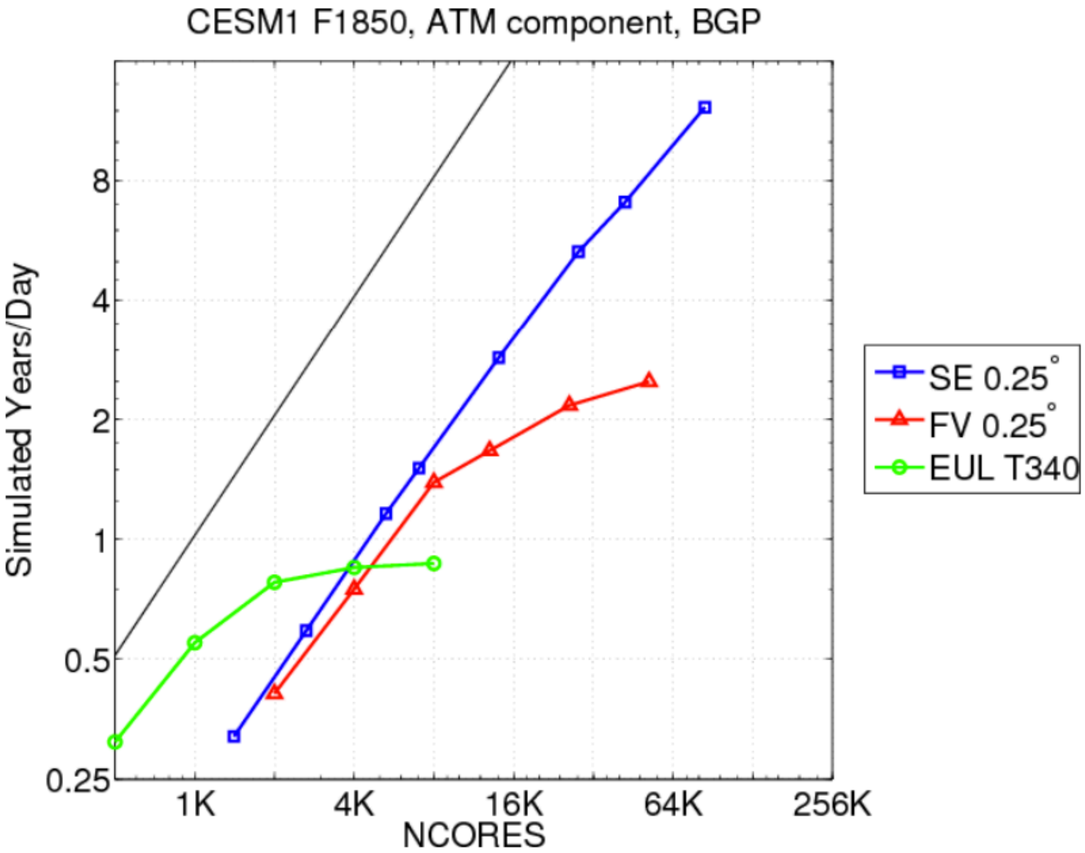
- Choose numerical methods that require a lot of computing, but little memory
- Compact numerical methods – Spectral Element or Discontinuous Galerkin?

Computing Power

CAM4 0.25° (28km) scalability
IBM GB/P "Intrepid"

Spectral element dynamical core (using a local finite-element method) achieves near-perfect scalability to 1 element per core (86000 cores) with peak performance 12.2 SYPD

Full CESM runs ~ 50% slower due to other components

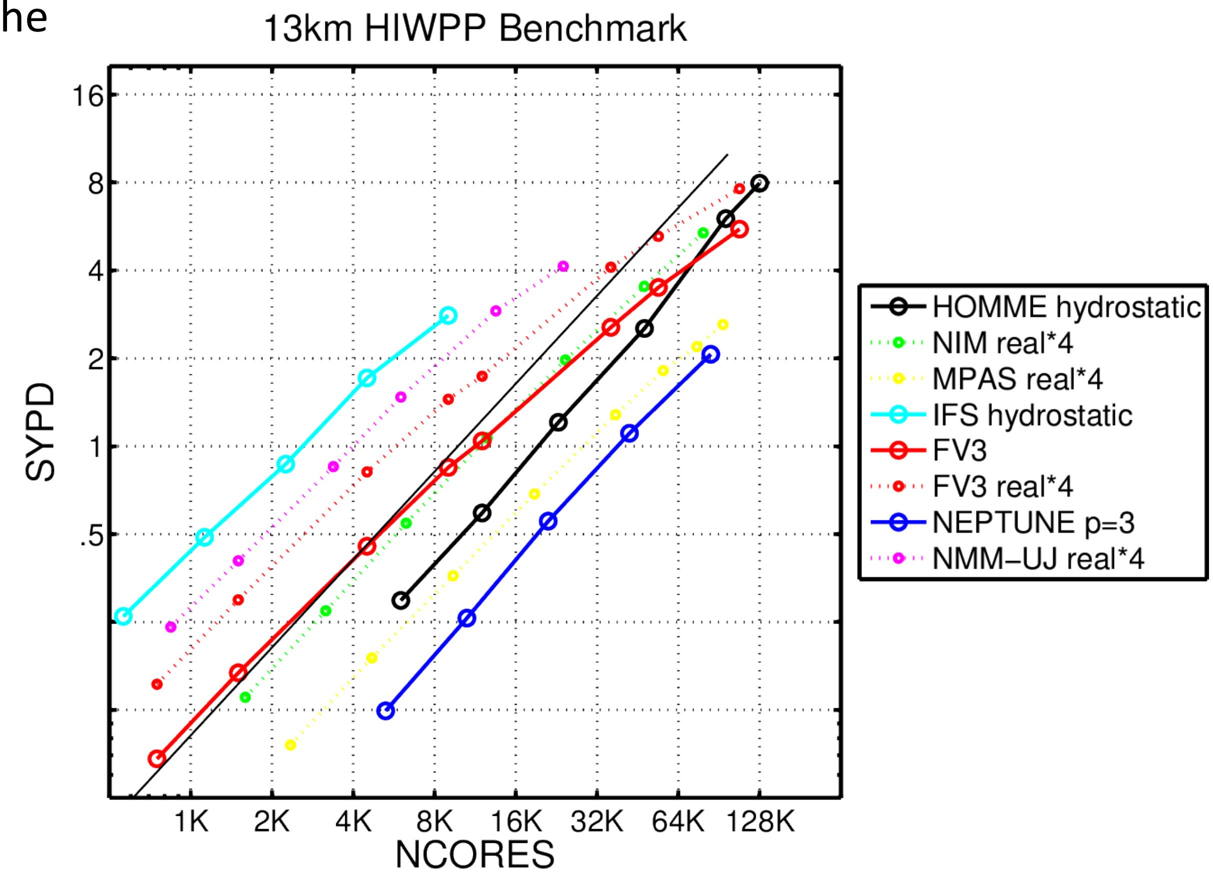


Source: Mark Taylor, correspondence

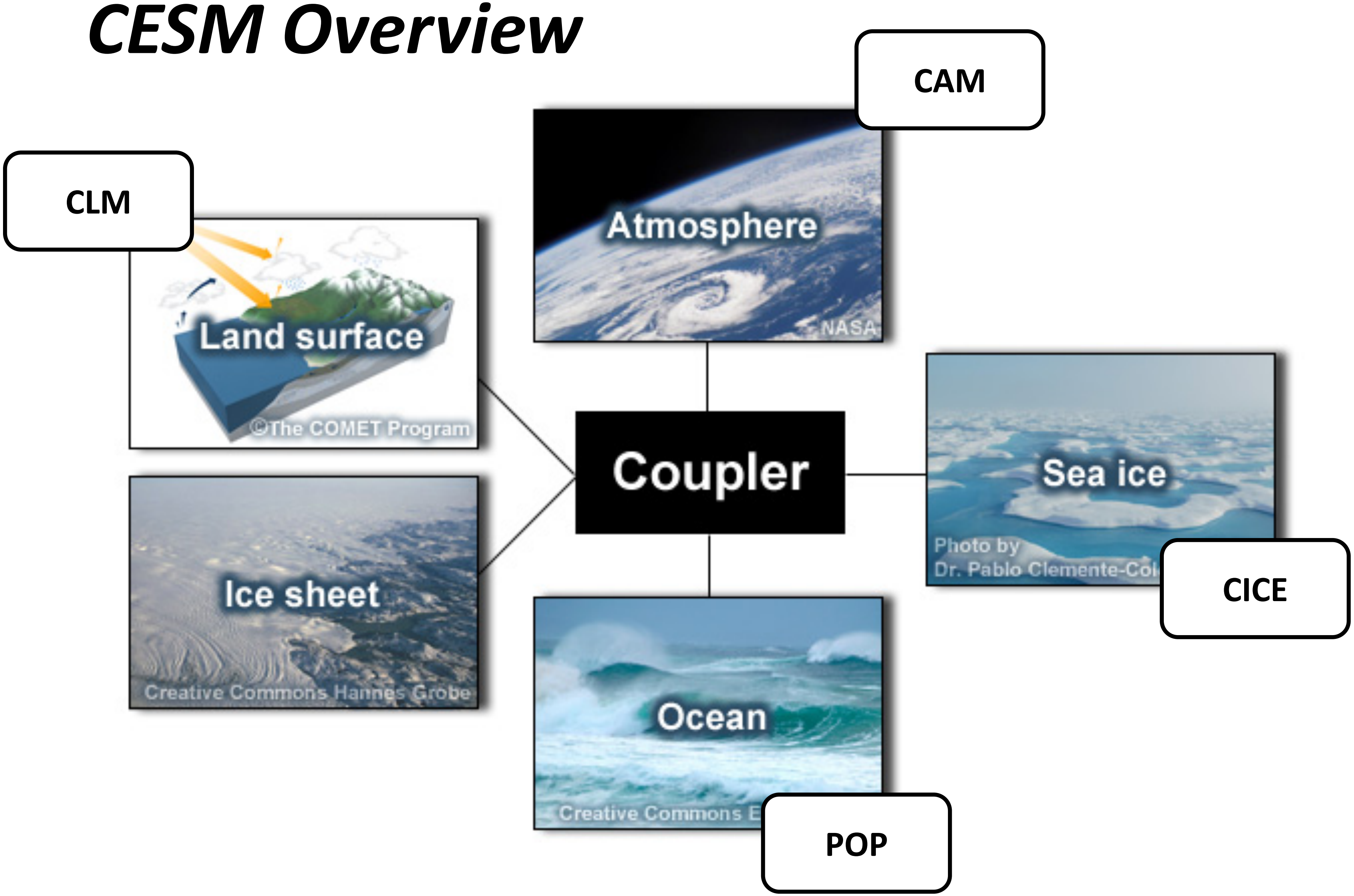
Parallel Scalability: Global Models

Figure: Recent results from the HIWPP intercomparison, showing many of the leading global atmospheric modeling systems.

Note that much of the spread in performance can be purely attributed to the amount of time spent optimizing each code.

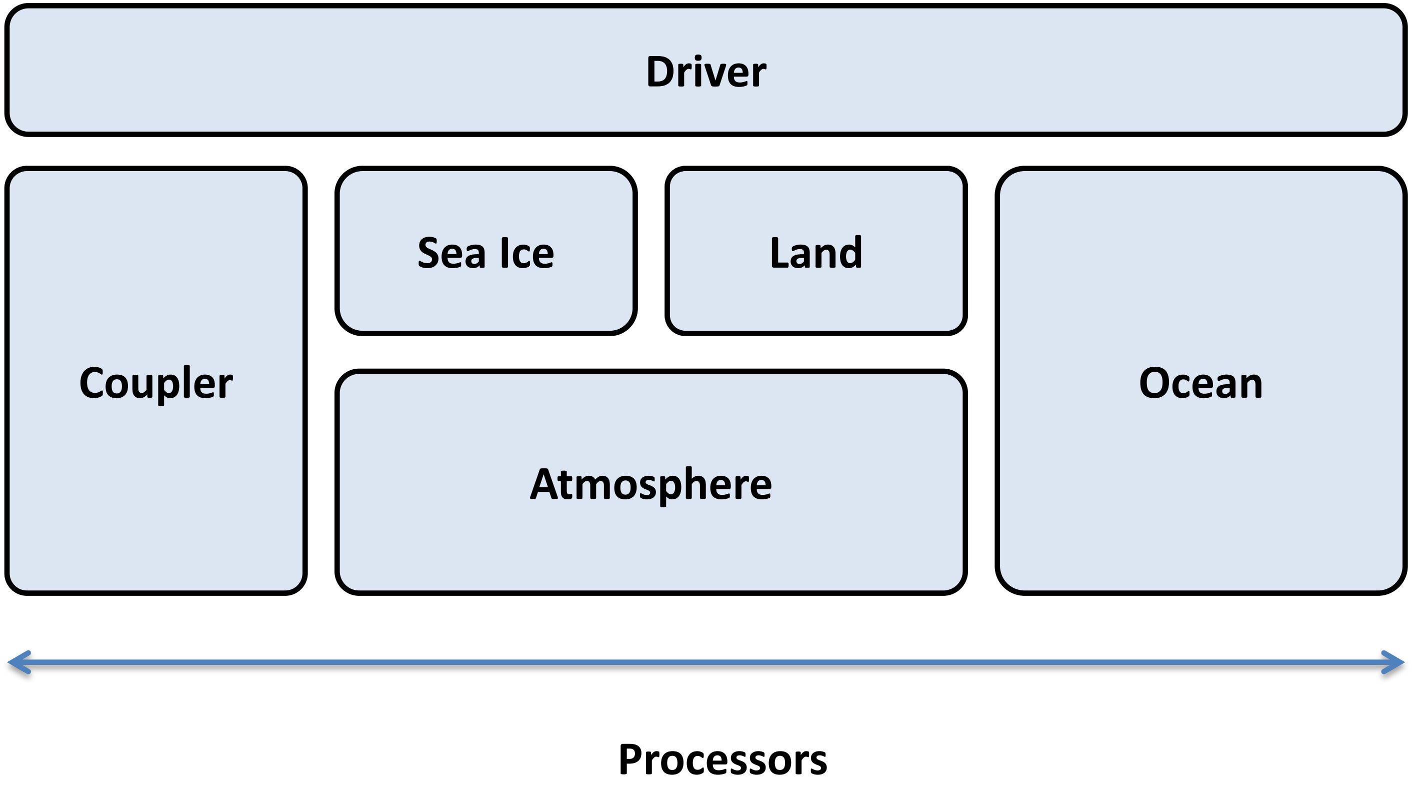


CESM Overview



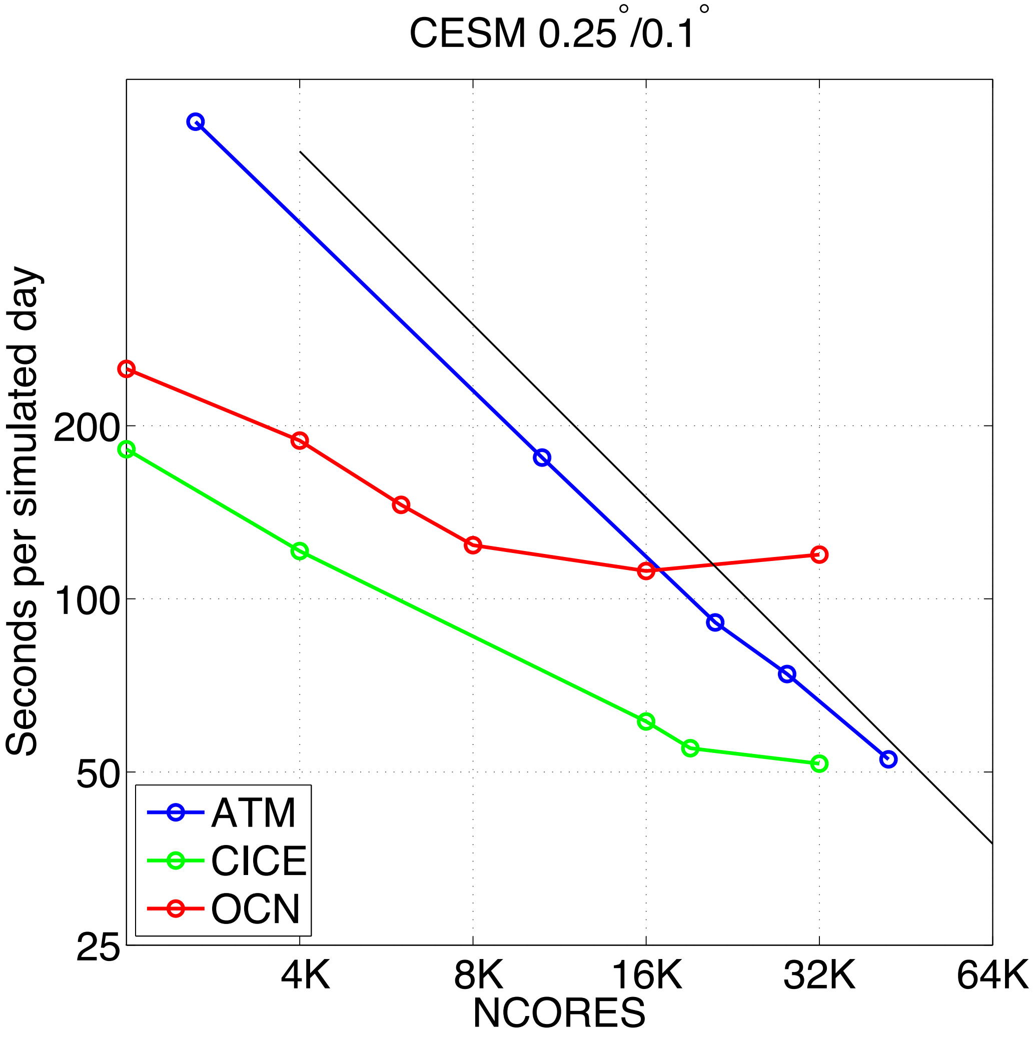
CESM Overview

Improved locality of atmosphere / ocean processes will be important going forward, leading to a more even distribution of work.



Parallel Scalability: CESM Components

Figure: Coupled model performance tends to be worse than individual model performance. Incompressibility in the ocean adds additional communication cost which drives down parallelism.



Ideas to improve parallel performance in CESM have primarily focused on ensembles rather than improving the architecture to expose additional parallelism.

Maybe this is ok?