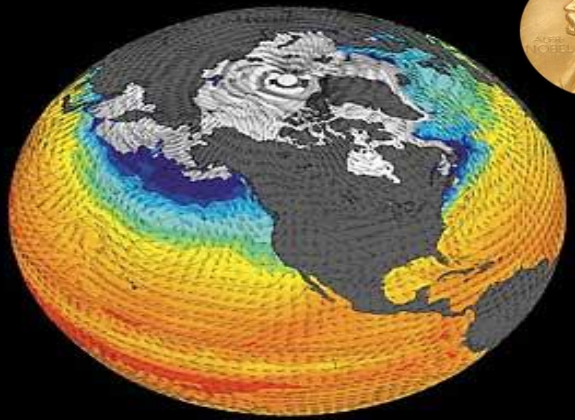
Programming Methodologies

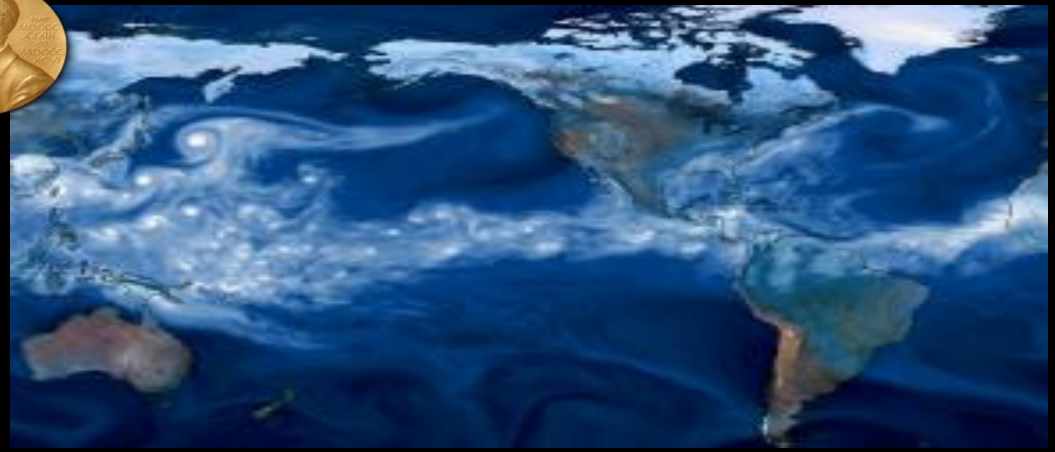
John Urbanic

Parallel Computing Scientist

Pittsburgh Supercomputing Center

FLOPS we need: Climate change analysis



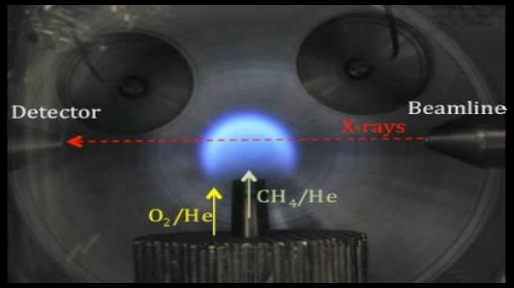


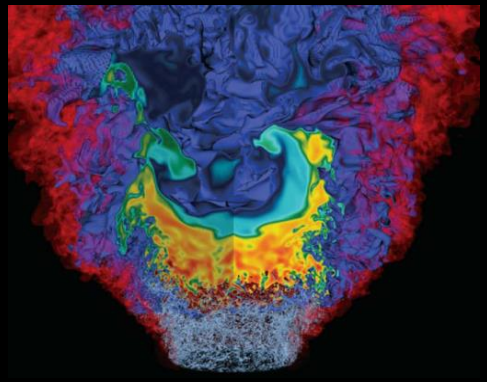
Simulations	Extreme data
• Cloud resolution, quantifying uncertainty, understanding tipping points, etc., will drive climate to exascale platforms • New math, models, and systems support will be needed	• "Reanalysis" projects need 100× more computing to analyze observations • Machine learning and other analytics are needed today for petabyte data sets • Combined simulation/observation will empower policy makers and scientists

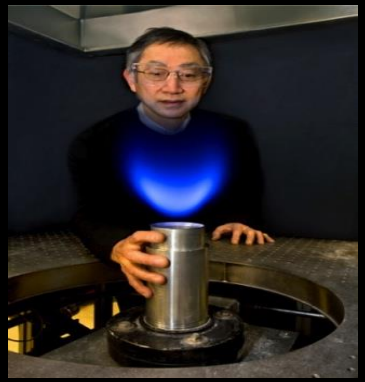
Courtesy Horst Simon, LBNL

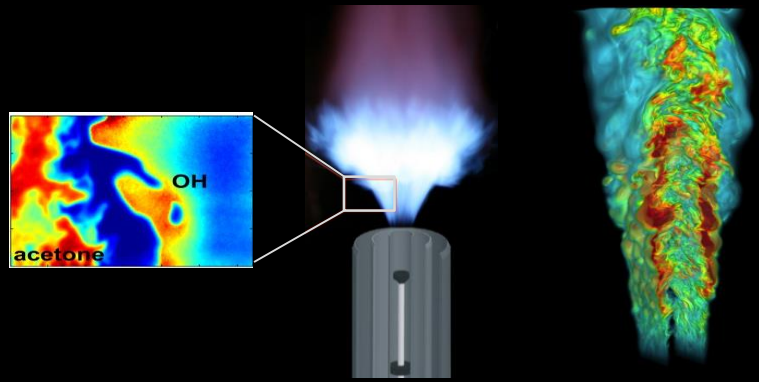
Exascale combustion simulations

- Goal: 50% improvement in engine efficiency
- Center for Exascale Simulation of Combustion in Turbulence (ExaCT)
  - Combines M&S and experimentation
  - Uses new algorithms, programming models, and computer science









Courtesy Horst Simon, LBNL

Modha Group at IBM Almaden

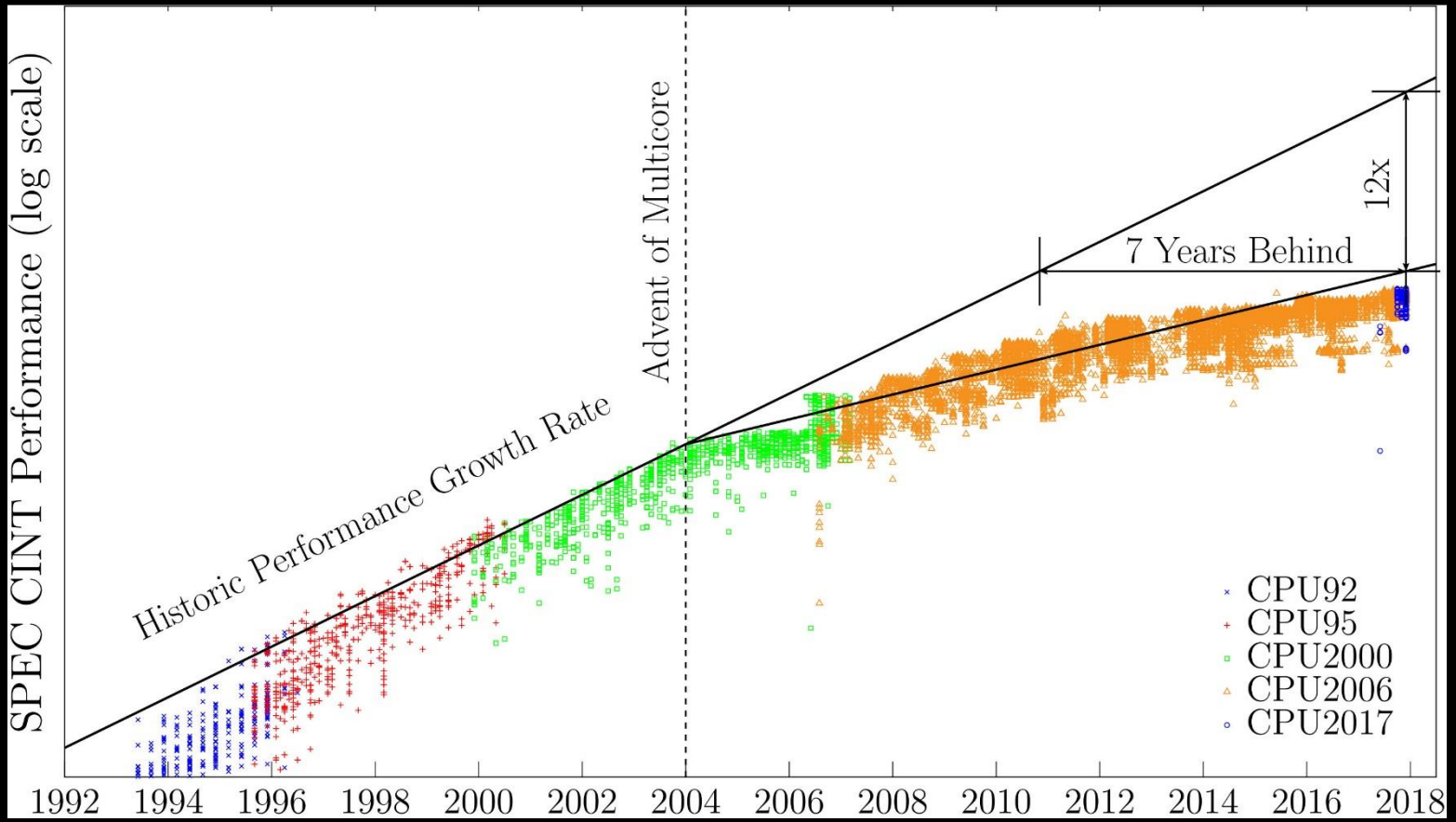
	Mouse	Rat	Cat	Monkey	Human
N:	16×10^6	56×10^6	763×10^6	2×10^9	22×10^9
S:	128×10^9	448×10^9	6.1×10^{12}	20×10^{12}	220×10^{12}

Recent simulations achieve unprecedented scale of $65*10^9$ neurons and $16*10^{12}$ synapses

Almaden	Watson	WatsonShaheen	LLNL Dawn	LLNL Sequoia
BG/L	BG/L	BG/P	BG/P	BG/Q
December, 2006	April, 2007	March, 2009	May, 2009	June, 2012

Courtesy Horst Simon, LBNL

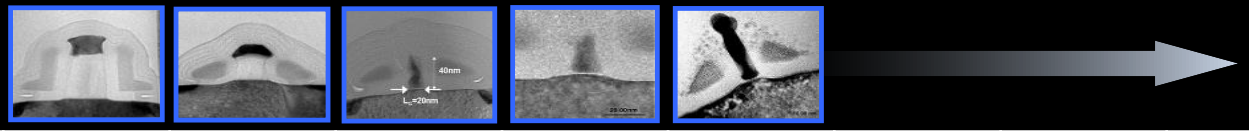
Moore's Law abandoned serial programming around 2004



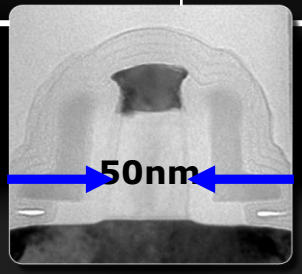
Courtesy Liberty Computer Architecture Research Group

Moore's Law is not to blame...

Intel process technology capabilities

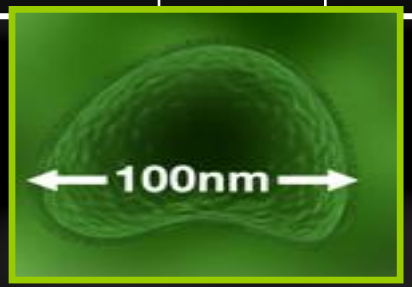


High Volume Manufacturing	2004	2006	2008	2010	2012	2014	2016	2018
Feature Size	90nm	65nm	45nm	32nm	22nm	16nm	11nm	8nm
Integration Capacity (Billions of Transistors)	2	4	8	16	32	64	128	256



Transistor for 90nm Process

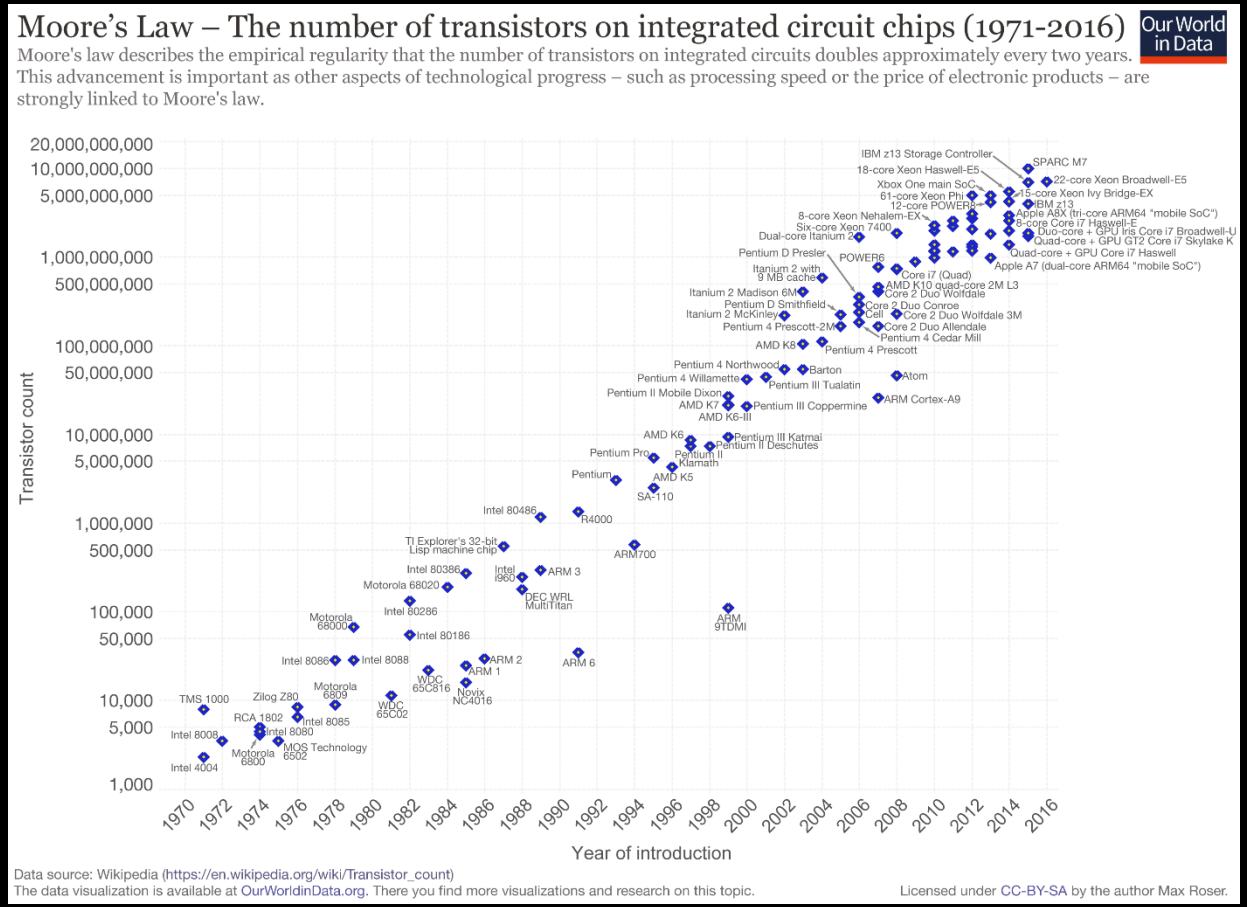
Source: Intel



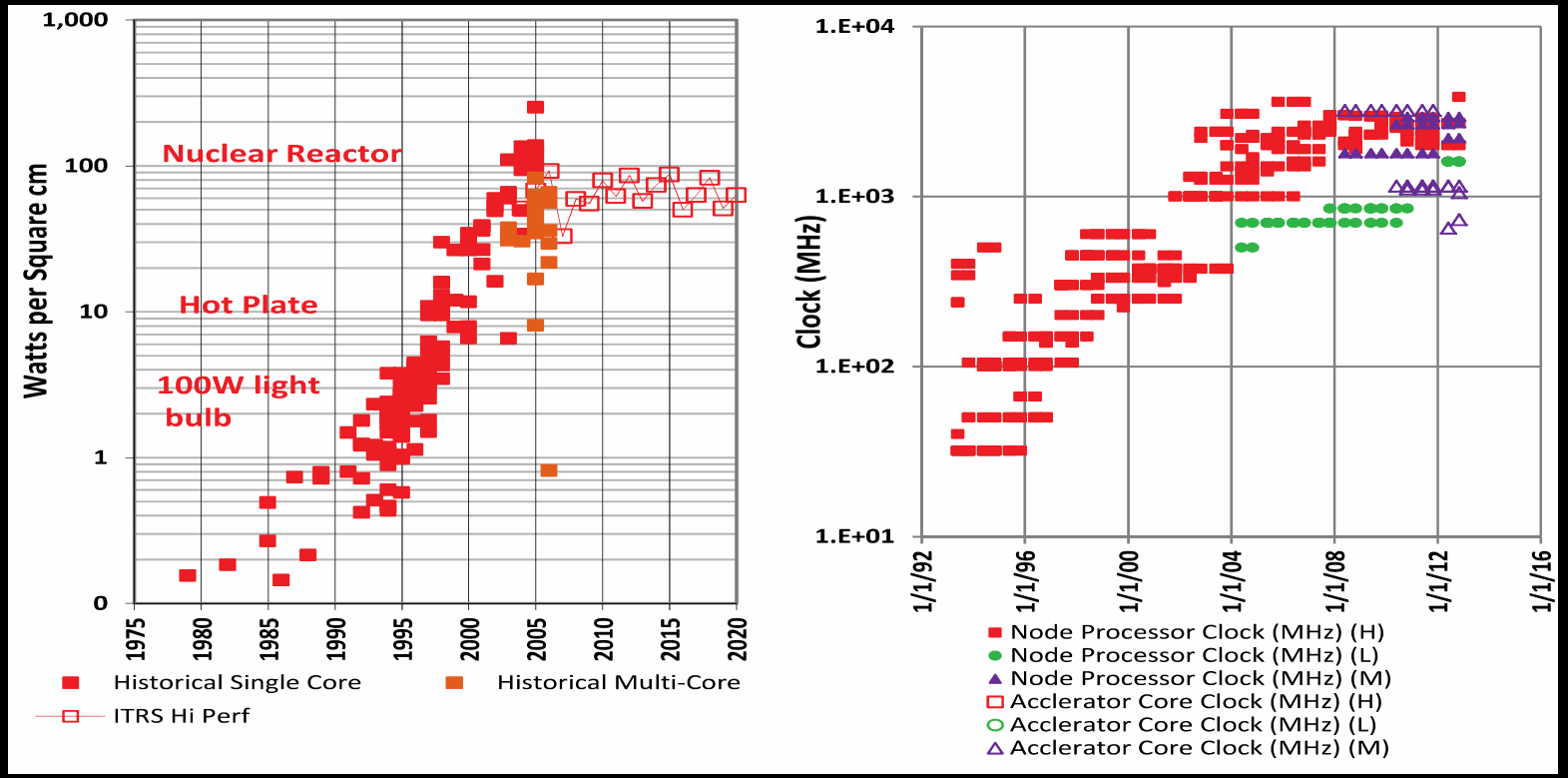
Influenza Virus

Source: CDC

At end of day, we keep using all those new transistors.



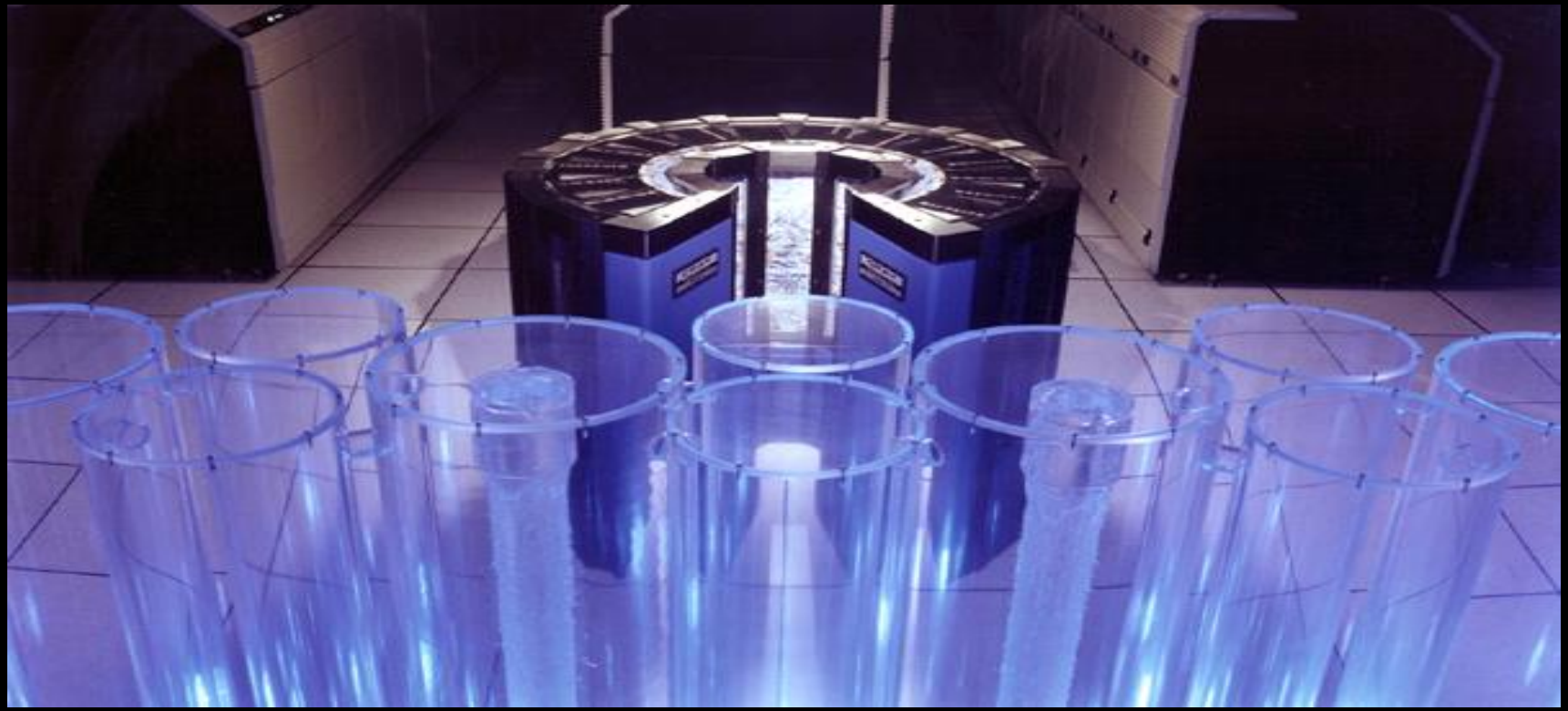
That Power and Clock Inflection Point in 2004… didn't get better.



Fun fact: At 100+ Watts and <1V, currents are beginning to exceed 100A at the point of load!

Courtesy Horst Simon, LBNL

Not a new problem, just a new scale…



Cray-2 with cooling tower in foreground, circa 1985

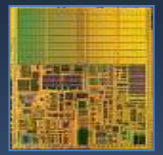
And how to get more performance from more transistors with the same power.

A 15% Reduction In Voltage Yields

RULE OF THUMB

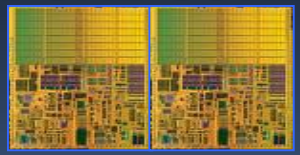
Frequency Reduction	Power Reduction	Performance Reduction
15%	45%	10%

SINGLE CORE



Area = 1
Voltage = 1
Freq = 1
Power = 1
Perf = 1

DUAL CORE

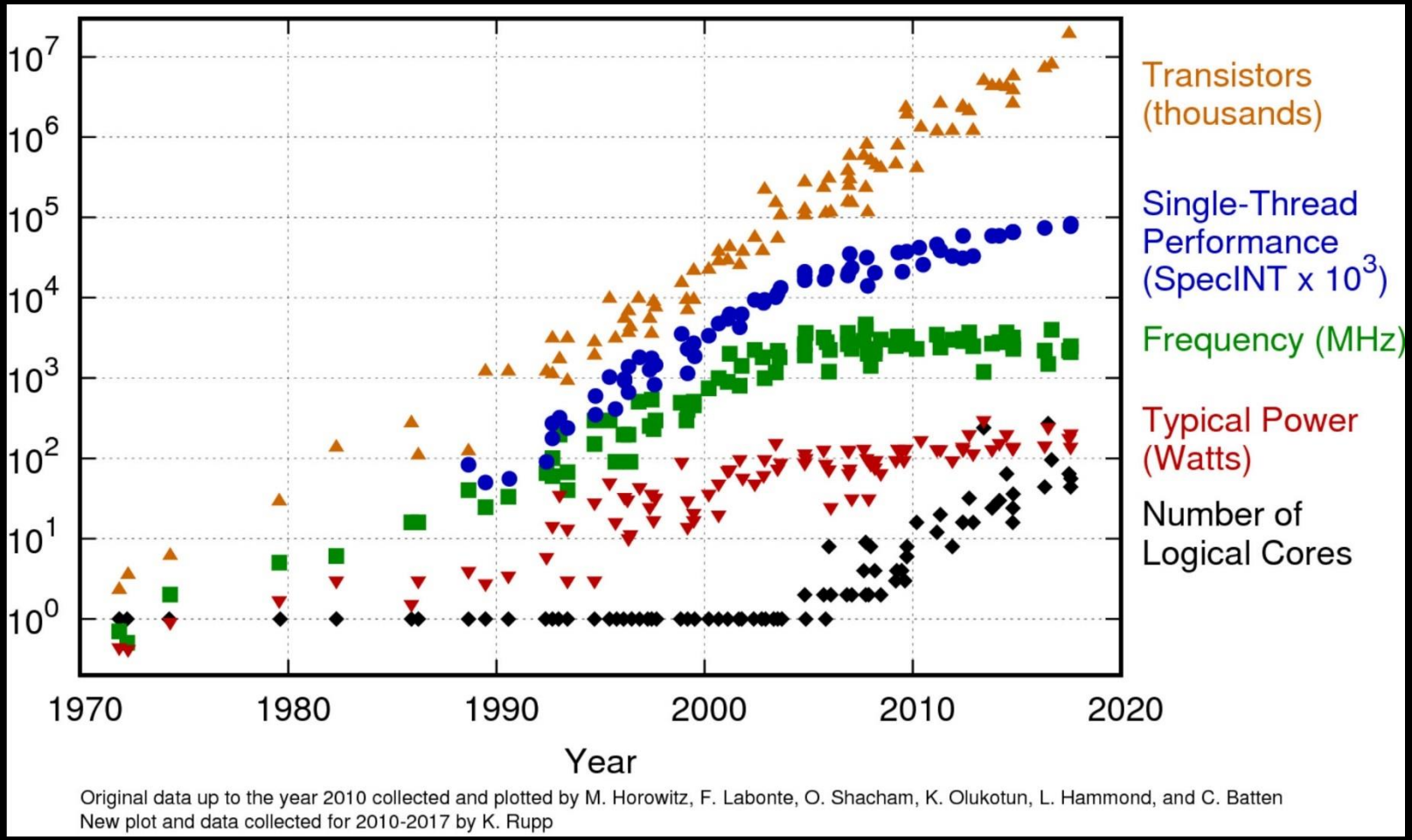


Area = 2
Voltage = 0.85
Freq = 0.85
Power = 1
Perf = ~1.8

Single Socket Parallelism: On your desktop

Processor	Year	Vector	Bits	SP FLOPs / core / cycle	Cores	FLOPs/cycle
Pentium III	1999	SSE	128	3	1	3
Pentium IV	2001	SSE2	128	4	1	4
Core	2006	SSE3	128	8	2	16
Nehalem	2008	SSE4	128	8	10	80
Sandybridge	2011	AVX	256	16	12	192
Haswell	2013	AVX2	256	32	18	576
KNC	2012	AVX512	512	32	64	2048
KNL	2016	AVX512	512	64	72	4608
Skylake	2017	AVX512	512	96	28	2688

Putting It All Together



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 2010-2017 by K. Rupp

MPPs (Massively Parallel Processors)

Distributed memory at largest scale. Shared memory at lower level.

Summit (ORNL)

- 122 PFlops Rmax and 187 PFlops Rpeak
- IBM Power 9, 22 core, 3GHz CPUs
- 2,282,544 cores
- NVIDIA Volta GPUs
- EDR Infiniband

Sunway TaihuLight (NSC, China)

- 93 PFlops Rmax and 125 PFlops Rpeak
- Sunway SW26010 260 core, 1.45GHz CPU
- 10,649,600 cores
- Sunway interconnect

Many Levels and Types of Parallelism

- Vector (SIMD)
- Instruction Level (ILP)
  - Instruction pipelining
  - Superscaler (multiple instruction units)
  - Out-of-order
  - Register renaming
  - Speculative execution
  - Branch prediction

Compiler
(not your problem)

OpenMP

- Multi-Core (Threads)
- SMP/Multi-socket

OpenACC

- Accelerators: GPU & MIC

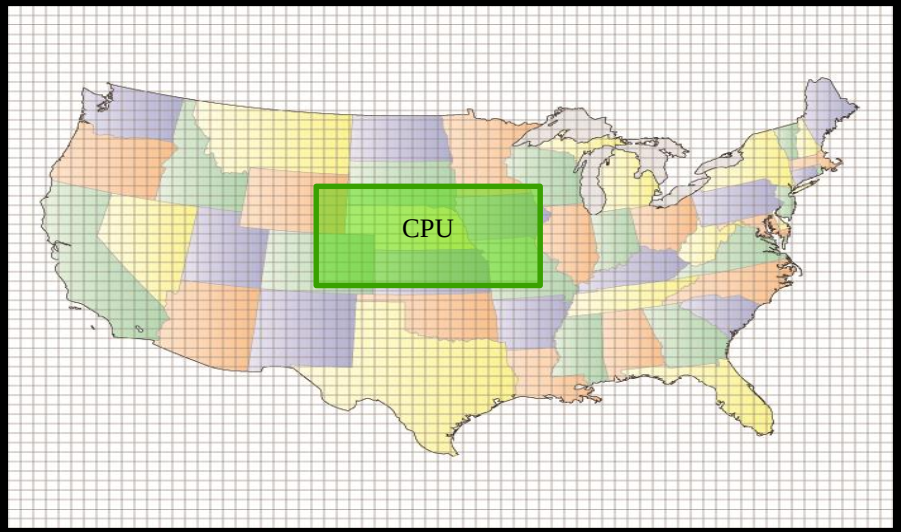
MPI

- Clusters
- MPPs

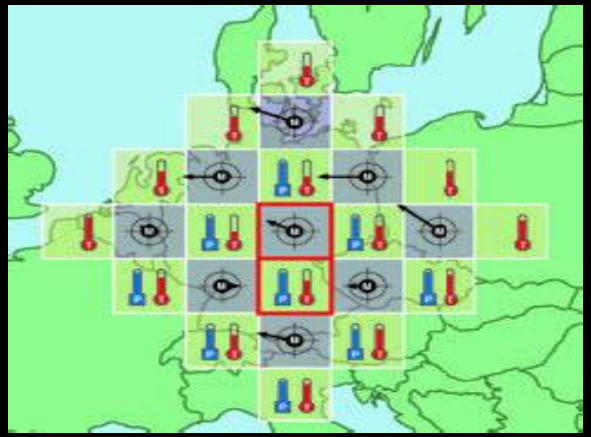
Also Important

- ASIC/FPGA/DSP
- RAID/IO

Prototypical Application: Serial Weather Model

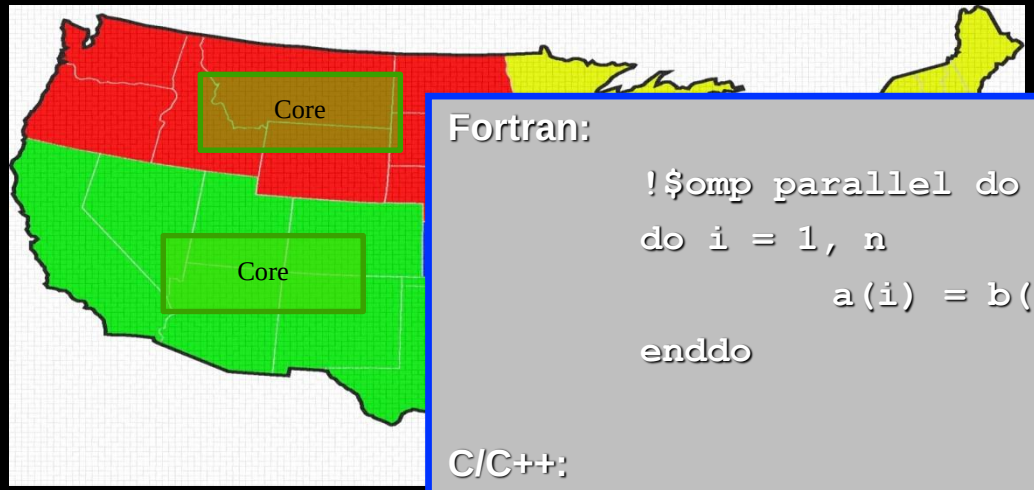


First parallel Weather Modeling algorithm: Richardson in 1917



Courtesy John Burkhardt, Virginia Tech

Weather Model: Shared Memory (OpenMP)



Four meteorologists in t

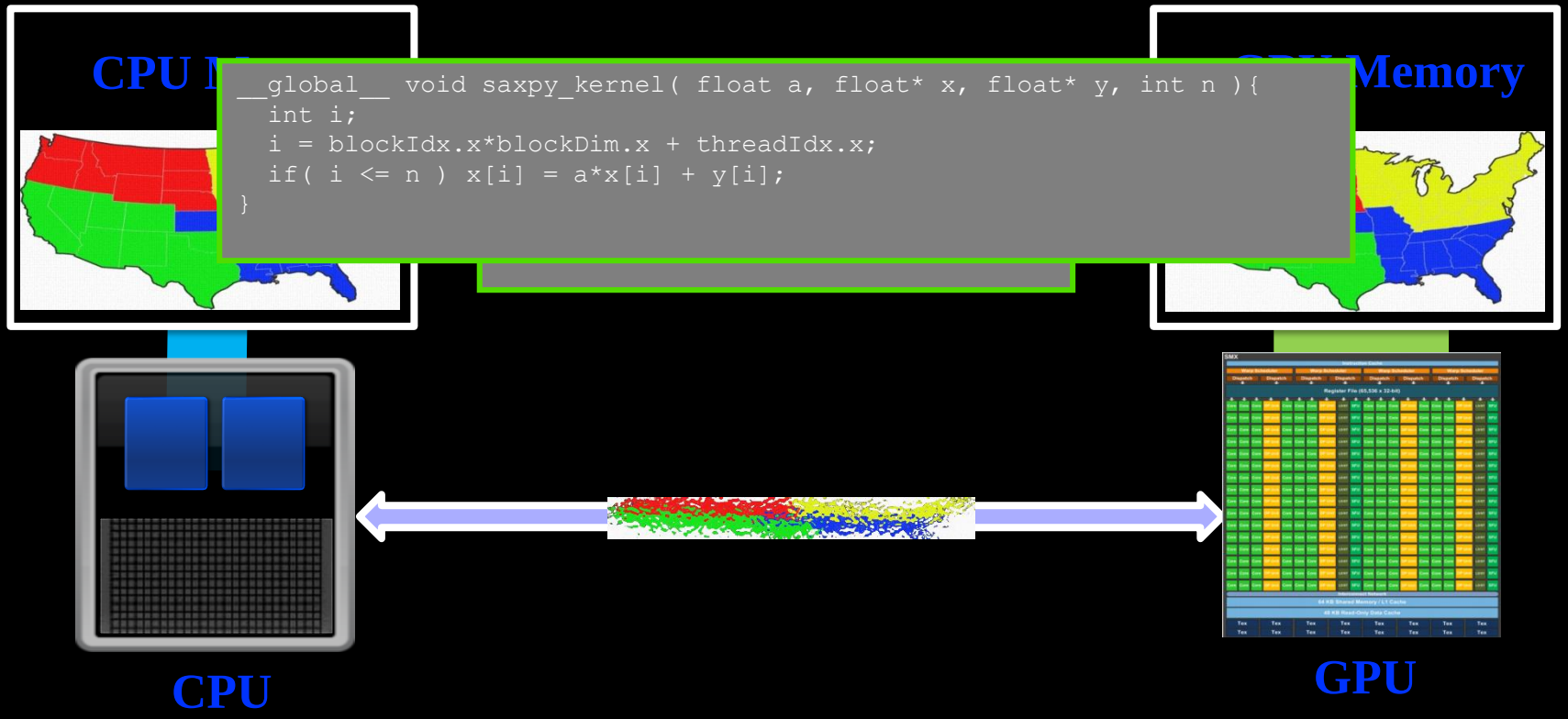
Fortran:

```
!$omp parallel do
do i = 1, n
        a(i) = b(i) + c(i)
enddo
```

C/C++:

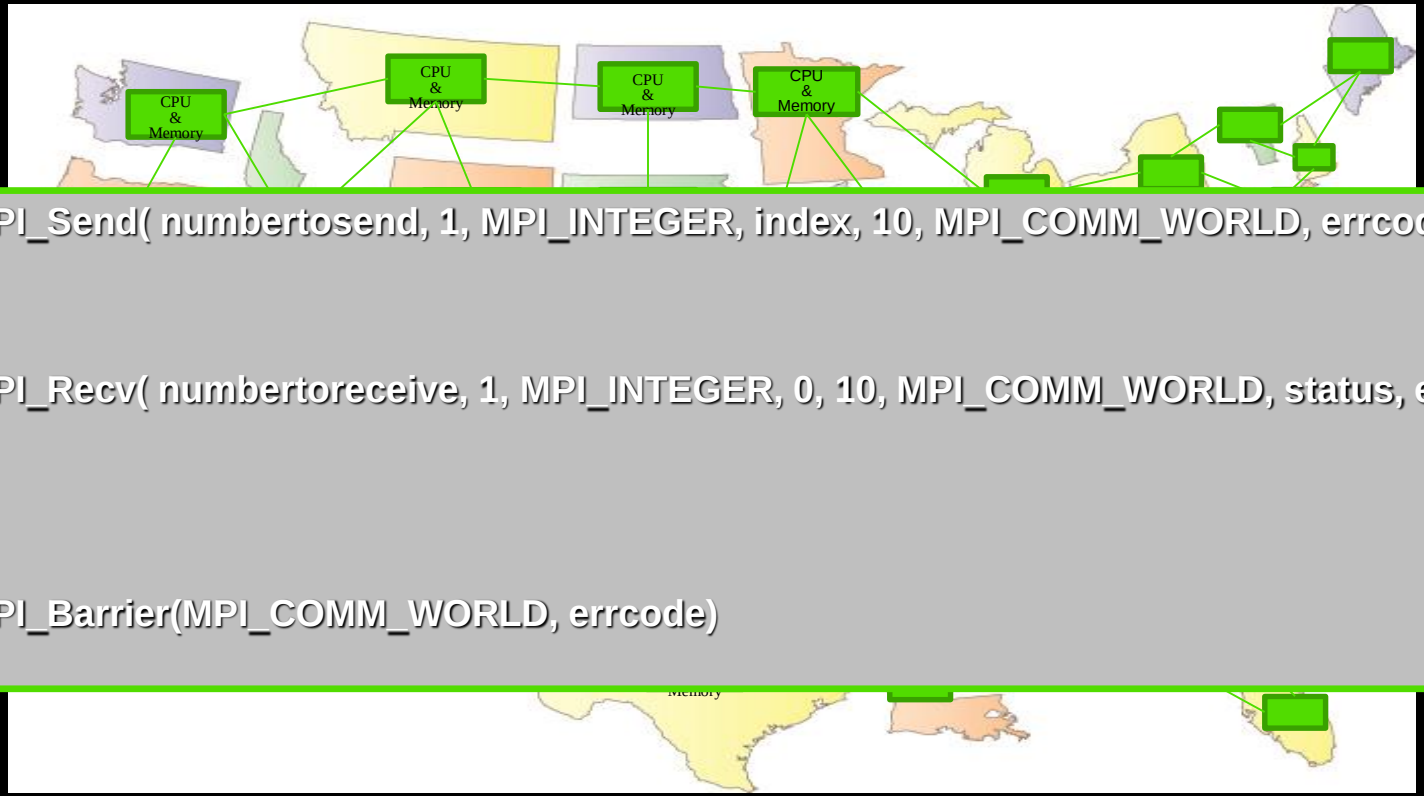
```
#pragma omp parallel for
for(i=1; i<=n; i++)
        a[i] = b[i] + c[i];
```

Weather Model: Accelerator (OpenACC)



1 meteorologists coordinating 1000 math savants using tin cans and a string.

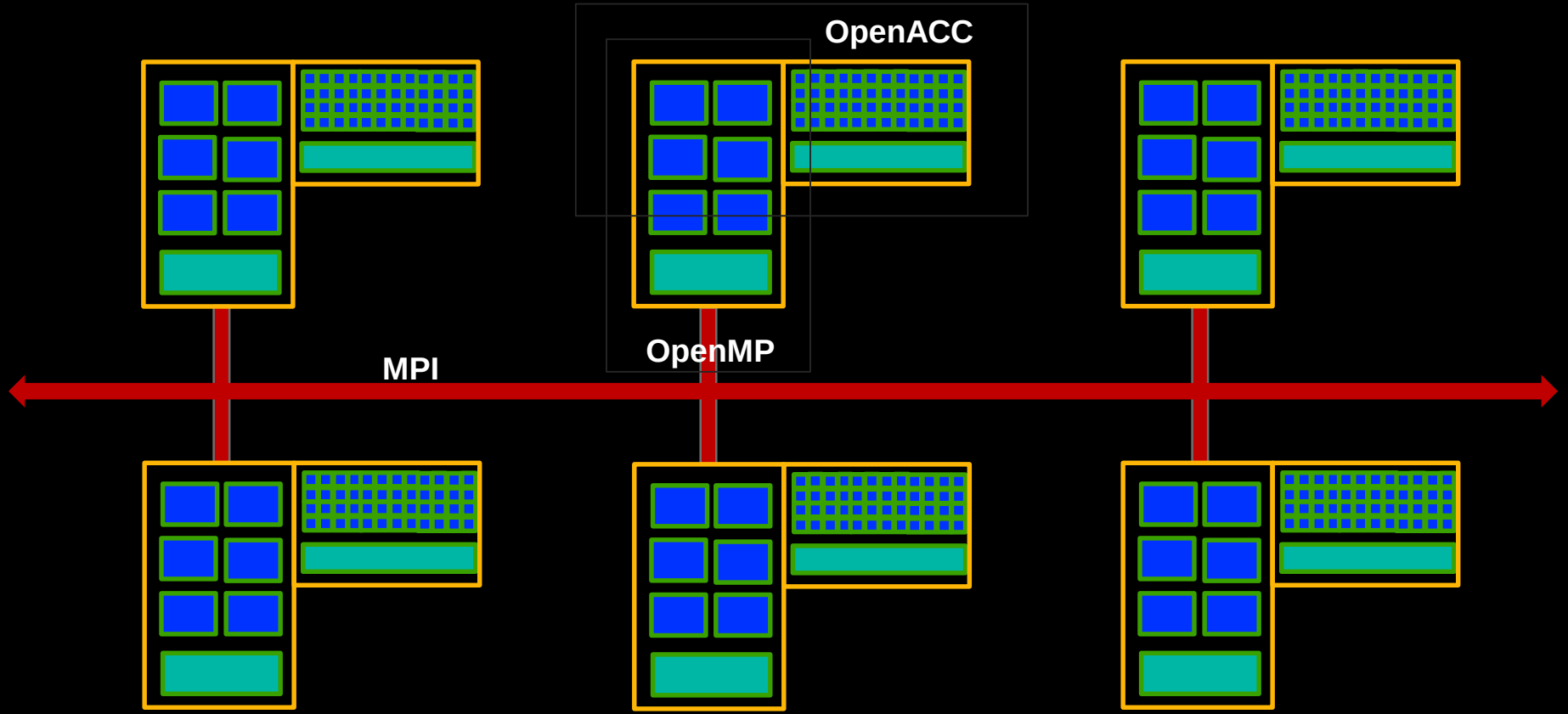
Weather Model: Distributed Memory (MPI)




```
call MPI_Send( numbertosend, 1, MPI_INTEGER, index, 10, MPI_COMM_WORLD, errcode)
.
.
call MPI_Recv( numbertoreceive, 1, MPI_INTEGER, 0, 10, MPI_COMM_WORLD, status, errcode)
.
.
.
call MPI_Barrier(MPI_COMM_WORLD, errcode)
.
```

50 meteorologists using a telegraph.

The pieces fit like this…



Top 10 Systems as of June 2019

#	Site	Manufacturer	Computer	CPU Interconnect [Accelerator]	Cores	Rmax (Tflops)	Rpeak (Tflops)	Power (MW)
1	DOE/SC/ORNL **United States**	IBM	Summit	Power9 22C 3.0 GHz Dual-rail Infiniband EDR NVIDIA V100	**2,414,592**	**148,600**	**200,794**	**10.1**
2	DOE/NNSA/LLNL **United States**	IBM	Sierra	Power9 3.1 GHz 22C Infiniband EDR NVIDIA V100	**1,572,480**	**94,640**	**125,712**	**7.4**
3	National Super Computer Center in Wuxi **China**	NRCPC	Sunway TaihuLight	Sunway SW26010 260C 1.45GHz	**10,649,600**	93,014	125,435	15.3
4	National Super Computer Center in Guangzhou **China**	NUDT	Tianhe-2 (MilkyWay-2)	Intel Xeon E5-2692 2.2 GHz TH Express-2 Intel Xeon Phi 31S1P	**4,981,760**	**61,444**	**100,678**	**18.4**
5	Texas Advanced Computing Center/Univ. of Texas **United States**	Dell	Frontera	Intel Xeon 8280 28C 2.7 GHz InfiniBand HDR	**448,448**	**23,516**	**38,745**	
6	Swiss National Supercomputing Centre (CSCS) **Switzerland**	Cray	Piz Daint Cray XC50	Xeon E5-2690 2.6 GHz Aries NVIDIA P100	**387,872**	**21,230**	**27,154**	**2.4**
7	DOE/NNSA/LANL/SNL **United States**	Cray	Trinity Cray XC40	Xeon E5-2698v3 2.3 GHz Aries Intel Xeon Phi 7250	**979,072**	**20,158**	**41,461**	**7.6**
8	AIST **Japan**	Fujitsu	AI Bridging Cloud Primergy	Xeon 6148 20C 2.4GHz InfiniBand EDR NVIDIA V100	**391,680**	**19,880**	**32,576**	**1.6**
9	Leibniz Rechenzentrum **Germany**	Lenovo	SuperMUC-NG	Xeon 8174 24C 3.1GHz Intel Omni-Path NVIDIA V100	**305,856**	**19,476**	**26,873**	
10	DOE/NNSA/LLNL **United States**	IBM	Lassen	Power9 22C 3.1 GHz InfiniBand EDR NVIDIA V100	**288,288**	**18,200**	**23,047**	

OpenACC is a first class API!

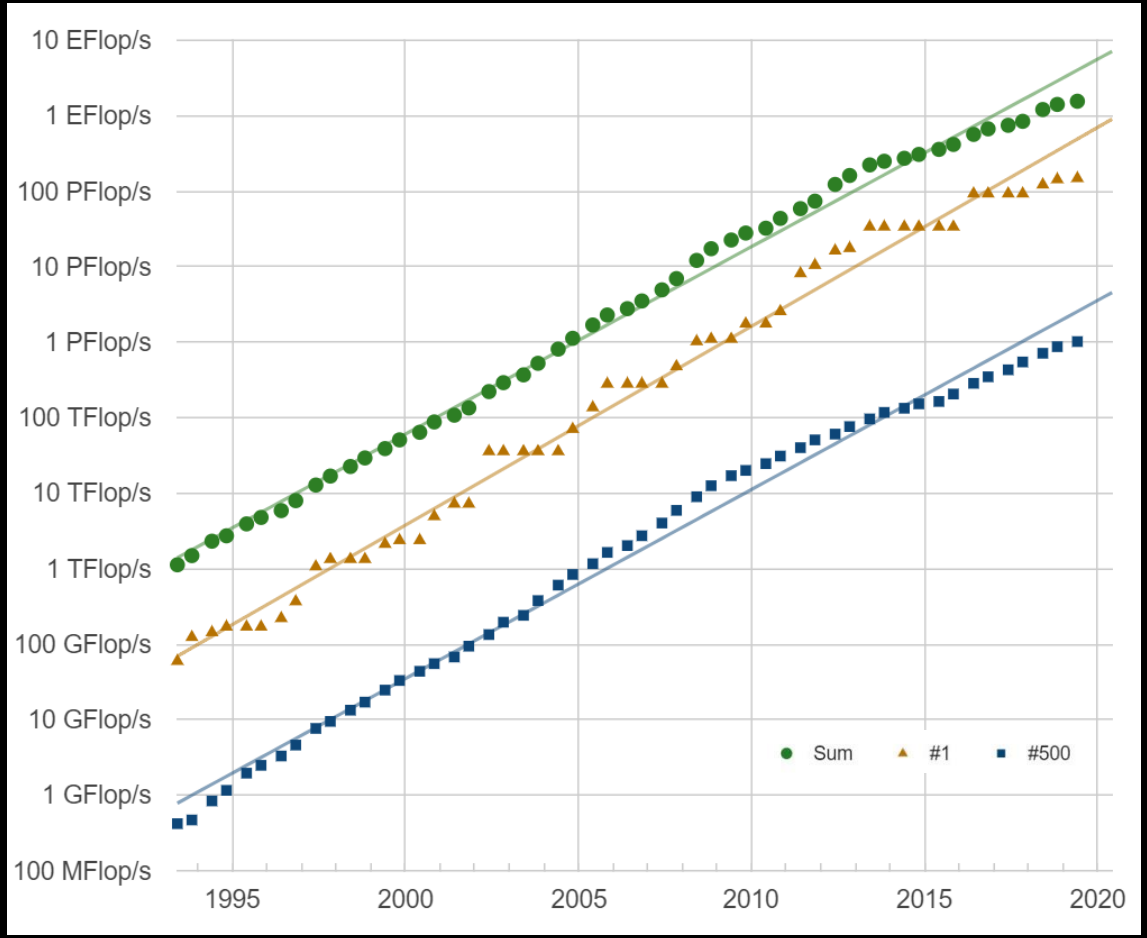
Other Paradigms?

- ✓ Message Passing
  - MPI
- ✓ Threads
  - OpenMP, OpenACC, CUDA
- ✓ Hybrid
  - MPI + OpenMP
- Data Parallel
  - Fortran90
- PGAS (Partitioned Global Address Space)
  - UPC, Coarray Fortran (Fortran 2008)
- Frameworks
  - Charm++

Today

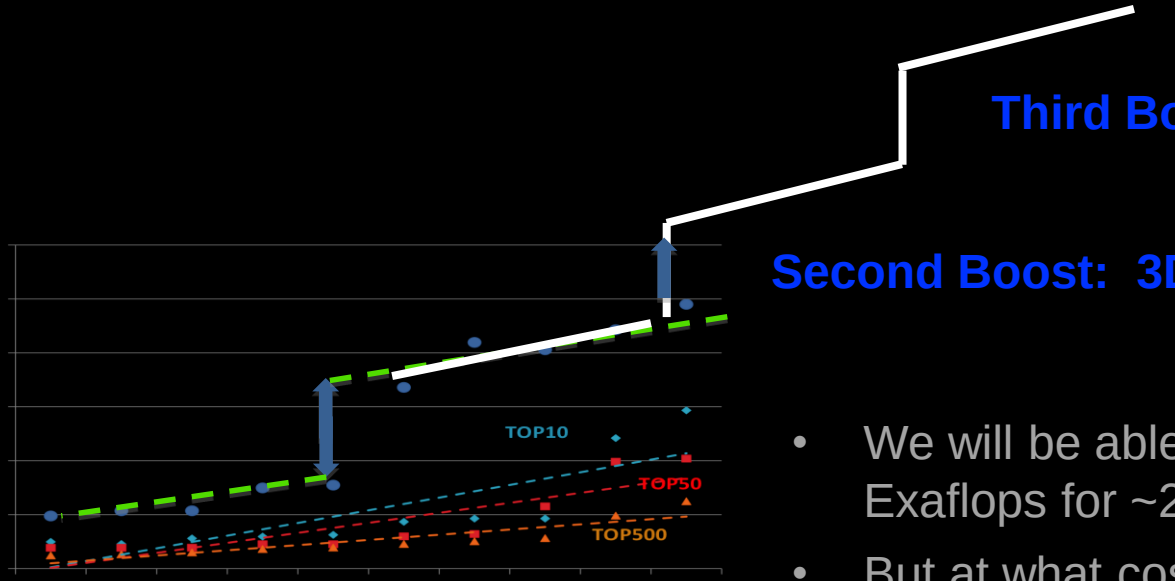
- Pflops computing fully established with more than 500 machines
- The field is thriving
- Interest in supercomputing is now worldwide, and growing in many new markets
- Exascale projects in many countries and regions

Sustaining Performance Improvements



TOP500

Two Additional Boosts to Improve Flops/Watt and Reach Exascale Target

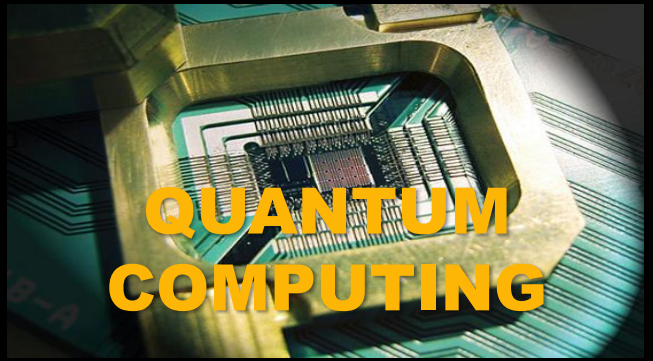


- We will be able to reach usable Exaflops for ~20 MW by 2024
- But at what cost?
- Will any of the other technologies give additional boosts after 2025?

Courtesy Horst Simon, LBNL

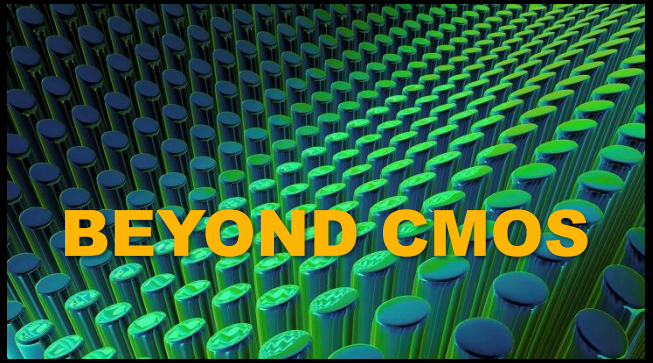
End of Moore's Law Will Lead to New Architectures

Non-von Neumann



ARCHITECTURE

von Neumann

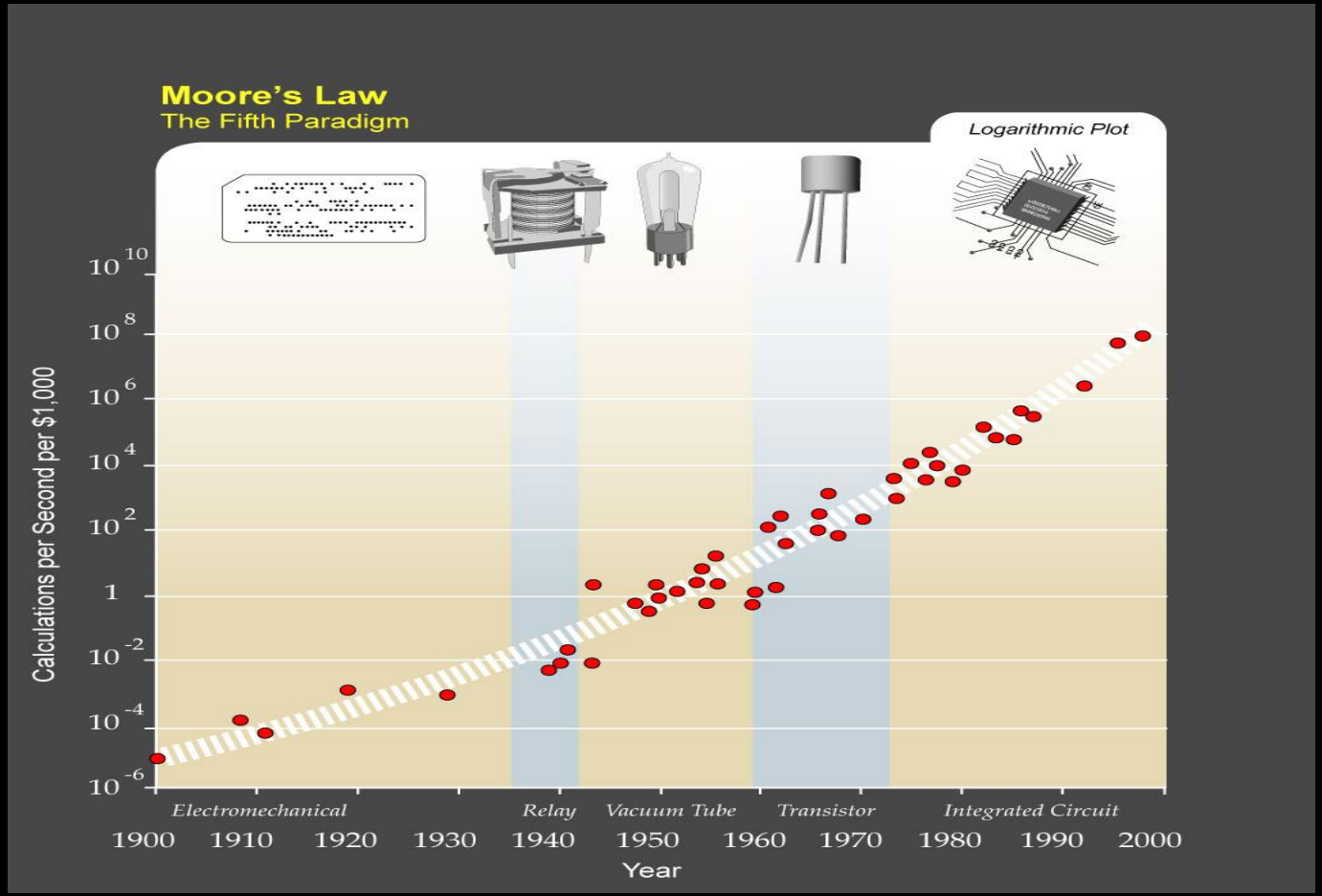


CMOS

Beyond CMOS

TECHNOLOGY

It would only be the 6th paradigm.



We can do better. We have a role model.

- Straight forward extrapolation results in a real time human brain scale simulation at about 1 - 10 Exaflop/s with 4 PB of memory
- Current predictions envision Exascale computers in 2022+ with a power consumption of at best 20 - 30 MW
- The human brain takes 20W
- Even under best assumptions in 2020 our brain will still be a million times more power efficient

Courtesy Horst Simon, LBNL

The Future and where you fit.

While the need is great, there is only a short list of serious contenders for 2020 exascale computing usability.

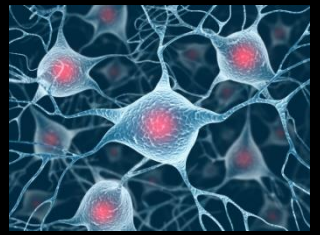
MPI 3.0 +X (MPI 3.0 specifically addresses exascale computing issues)

PGAS (partitioned global address space)

CAF (now in Fortran 2008!),UPC

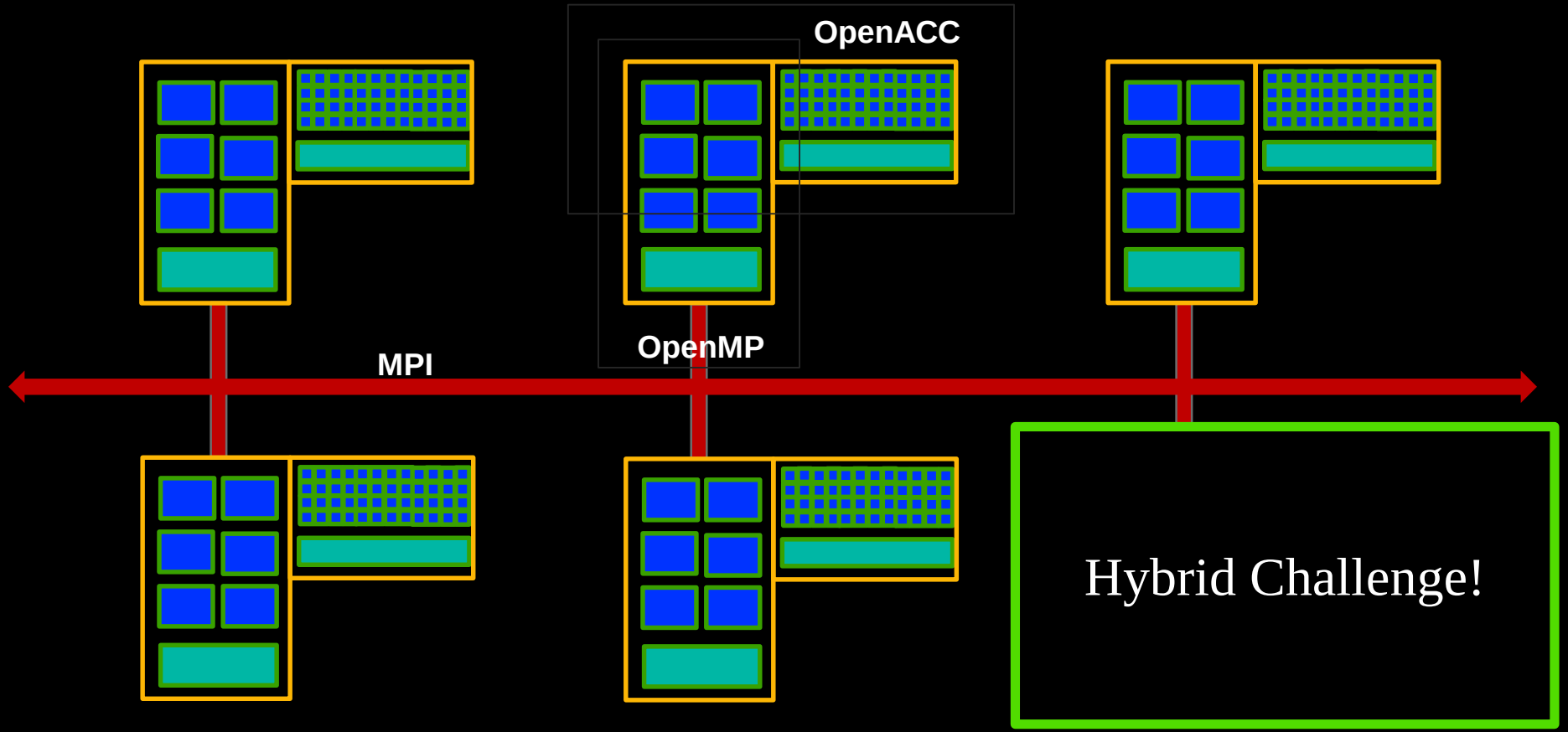
What about Big Data?

Deep Learning?



Thursday & Friday!

Again…



Appendix

Slides I had to ditch in the interest of time. However they are worthy of further discussion in this gathering. If some topic here catches your eye, or your ire, there are people all around you with related knowledge. I am also now an identified target.

Credits

- Horst Simon of LBNL
  - His many beautiful graphics are a result of his insightful perspectives
  - He puts his money where his mouth is: $2000 bet in 2012 that Exascale machine would not exist by end of decade
- Intel
  - Many datapoints flirting with NDA territory
- Top500.org
  - Data and tools
- Supporting cast:

Erich Strohmaier (LBNL)
Jack Dongarra (UTK)
Rob Leland (Sandia)
John Shalf (LBNL)
Scott Aronson (MIT)
Bob Lucas (USC-ISI)
John Kubiatowicz (UC Berkeley)
Dharmendra Modha and team(IBM)
Karlheinz Meier (Univ. Heidelberg)
Liberty Computer Architecture Research Group (Princeton)

Flops are free?

At exascale, >99% of power is consumed by moving operands across machine.

Does it make sense to focus on flops, or should we optimize around data movement?

To those that say the future will simply be Big Data:

“All science is either physics or stamp collecting.”

- Ernest Rutherford

Obstacles?

One of the many groups established to enable this outcome (the Advanced Scientific Computing Advisory Committee) puts forward this list of 10 technical challenges.

- Energy efficient circuit, power and cooling technologies.
- High performance interconnect technologies.
- Advanced memory technologies to dramatically improve capacity and bandwidth.
- Scalable system software that is power and resilience aware.
- Data management software that can handle the volume, velocity and diversity of data-storage
- Programming environments to express massive parallelism, data locality, and resilience.
- Reformulating science problems and refactoring solution algorithms for exascale.
- Ensuring correctness in the face of faults, reproducibility, and algorithm verification.
- Mathematical optimization and uncertainty quantification for discovery, design, and decision.
- Software engineering and supporting structures to enable scientific productivity.

It is not just “exaflops” – we are changing the whole computational model

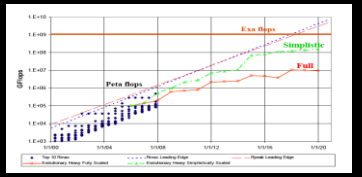
Current programming systems have WRONG optimization targets

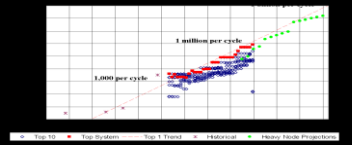
Old Constraints

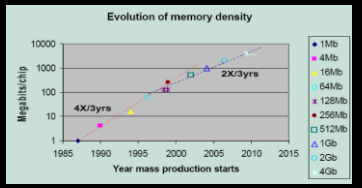
- **Peak clock frequency** ***as primary limiter for performance improvement***
- **Cost:** ***FLOPs*** **are biggest cost for system:** ***optimize for compute***
- **Concurrency: Modest growth of parallelism by adding nodes**
- **Memory scaling*: maintain byte per flop capacity and bandwidth***
- **Locality*: MPI+X model (uniform costs within node & between nodes)***
- **Uniformity:** ***Assume uniform system performance***
- **Reliability*: It’s the hardware’s problem***

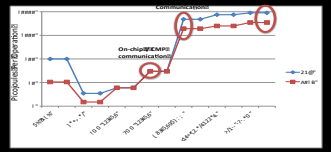
New Constraints

- **Power** *is primary design constraint for future HPC system design*
- **Cost:** *Data movement dominates: optimize to minimize data movement*
- **Concurrency:** *Exponential growth of parallelism within chips*
- **Memory Scaling:** *Compute growing 2x faster than capacity or bandwidth*
- **Locality***: must reason about data locality and possibly topology*
- **Heterogeneity**: *Architectural and performance non-uniformity increase*
- **Reliability**: *Cannot count on hardware protection alone*









Fundamentally breaks our current programming paradigm and computing ecosystem

Adapted from John Shalf

As a last resort, we ~~could~~ will learn to program again.

It has become a mantra of contemporary programming philosophy that developer hours are so much more valuable than hardware, that the best design compromise is to throw more hardware at slower code.

This might well be valid for some Java dashboard app used twice week by the CEO. But this has spread and results in…

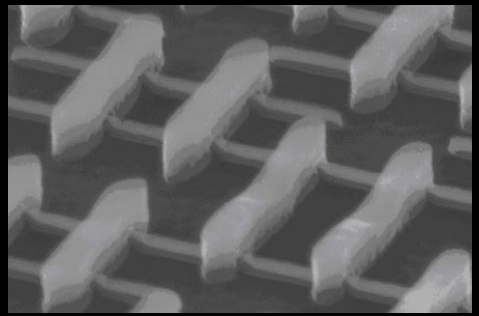
The common observation that a modern PC (or phone) seems to be more laggy than one from a few generations ago that had literally 1 thousandth the processing power.

Moore's Law has been the biggest enabler (or more accurately rationalization) for this trend. If Moore's Law does indeed end, then progress will require good programming.

No more garbage collecting, script languages. I am looking at you, Java, Python, Matlab,

…but our metrics are less clear.

After a while, “there was no one design rule that people could point to and say, ‘That defines the node name’ … The minimum dimensions are getting smaller, but I’m the first to admit that I can’t point to the one dimension that’s 32 nm or 22 nm or 14 nm. Some dimensions are smaller than the stated node name, and others are larger.”



Mark Bohr, Senior fellow at Intel.

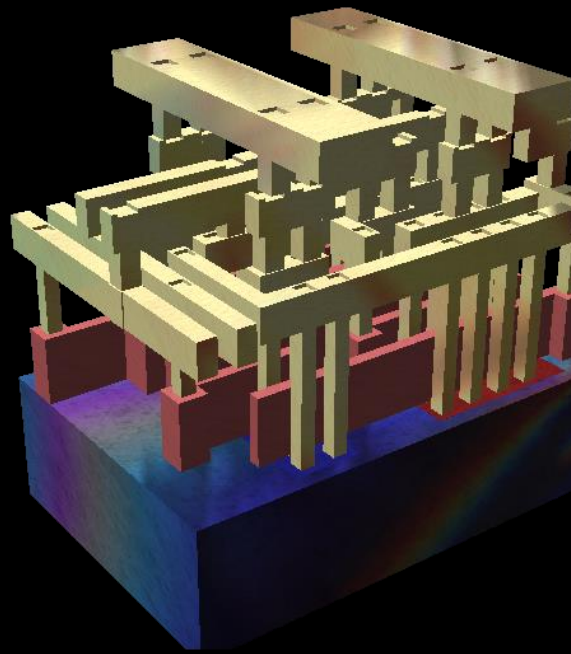
From *The Status of Moore's Law: It's Complicated* (IEEE Spectrum)

For a while thing were better than they appeared.

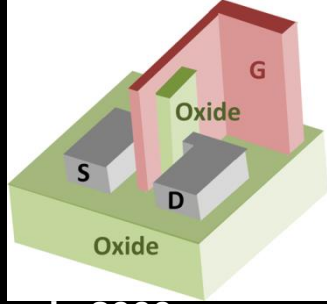
Intel’s 0.13-μm chips, which debuted in 2001, had transistor gates that were actually just 70 nm long. Nevertheless, Intel called them 0.13-μm chips because they were the next in line.

Manufacturers continued to pack the devices closer and closer together, assigning each successive chip generation a number about 70 percent that of the previous one.

A 30 percent reduction in both the x and y dimensions corresponds to a 50 percent reduction in the area occupied by a transistor, and therefore the potential to double transistor density on the chip.



Then new technologies carried the load.



After years of aggressive gate trimming, simple transistor scaling reached a limit in the early 2000s: Making a transistor smaller no longer meant it would be faster or less power hungry. So Intel, followed by others, introduced new technologies to help boost transistor performance.

- Strain engineering, adding impurities to silicon to alter the crystal, which had the effect of boosting speed without changing the physical dimensions of the transistor.
- New insulating and gate materials.
- And most recently, they rejiggered the transistor structure to create the more efficient FinFET, with a current-carrying channel that juts out of the plane of the chip.

The switch to FinFETs has made the situation even more complex. Intel’s 22-nm chips, the current state of the art, have FinFET transistors with gates that are 35 nm long but fins that are just 8 nm wide.

Now tradeoffs are stealing these gains.

The density and power levels on a state-of-the-art chip have forced designers to compensate by adding:

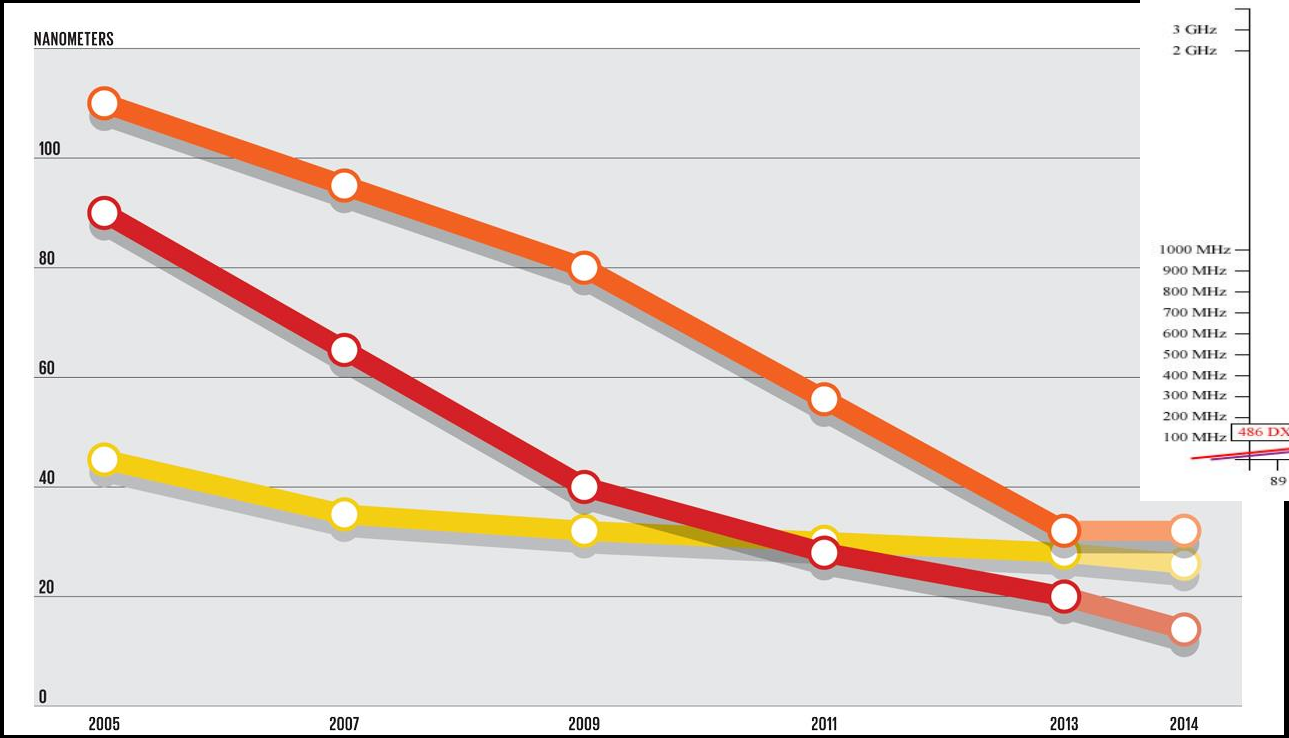
- **error-correction circuitry**
- **redundancy**
- **read- and write-boosting circuitry for failing static RAM cells**
- **circuits to track and adapt to performance variations**
- **complicated memory hierarchies to handle multicore architectures.**

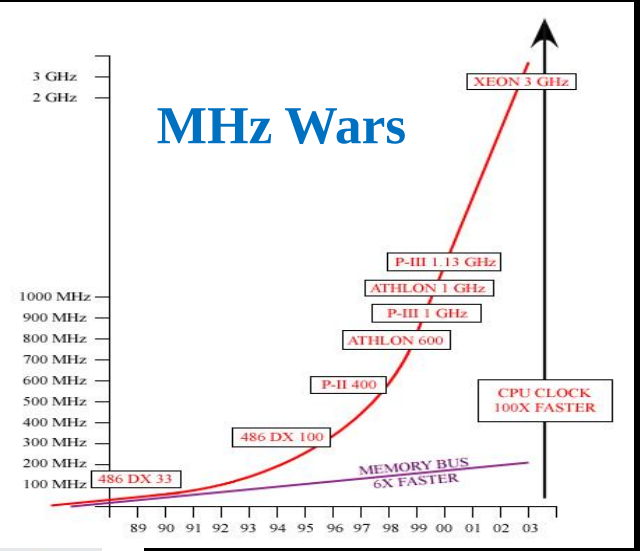
All of those extra circuits add area. Some analysts have concluded that when you factor those circuits in, chips are no longer twice as dense from generation to generation. One such analysis suggests, the density improvement over the past three generations, from 2007 on, has been closer to 1.6 than 2.

And cost per transistor has gone *up* for the first time ever:

- **2012 20M 28nm transistors/dollar**
- **2015 19M 16nm transistors/dollar**

Maybe they are even becoming “marketing”.





Global Foundries planned 2013 14nm chip introduction.

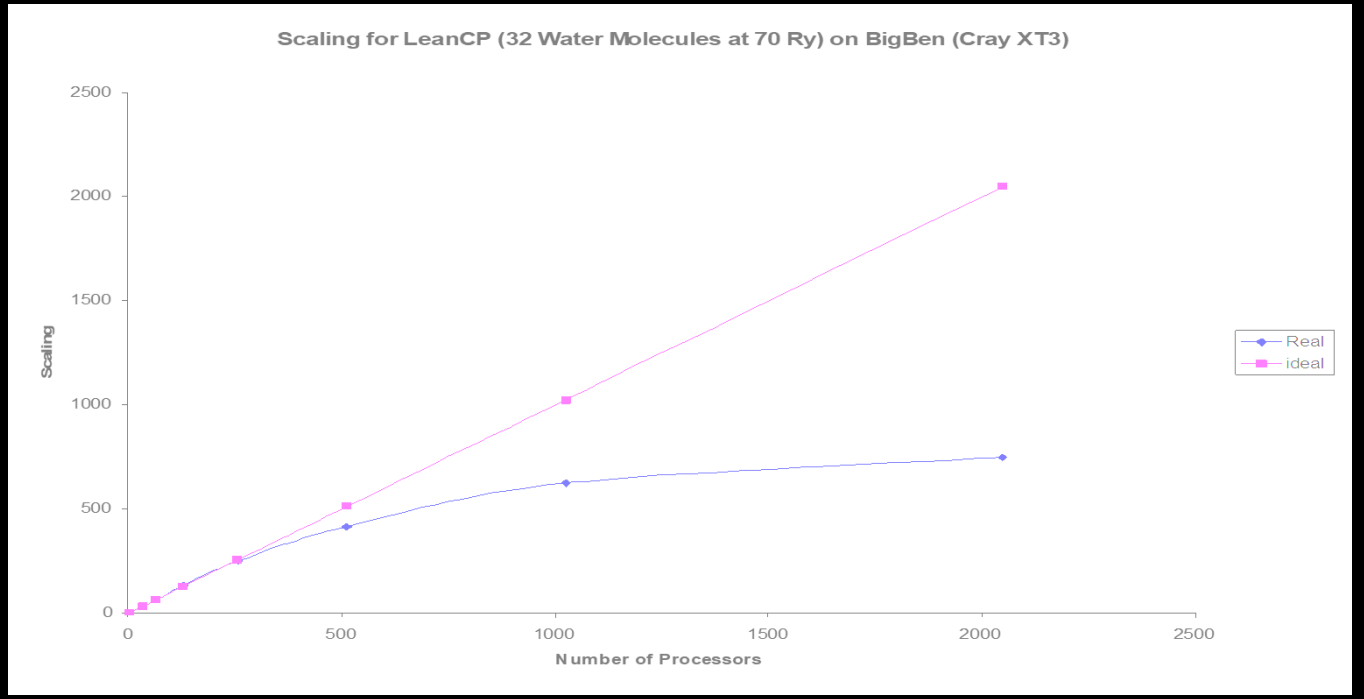
As reported by IEEE Spectrum

How parallel is a code?

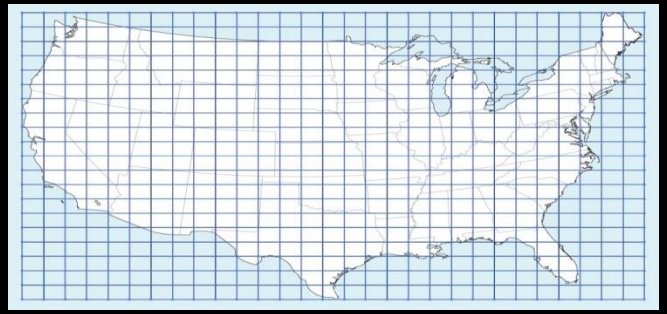
- **Parallel performance is defined in terms of scalability**

Strong Scalability
Can we get faster for a given problem size?

Weak Scalability
Can we maintain runtime as we scale up the problem?

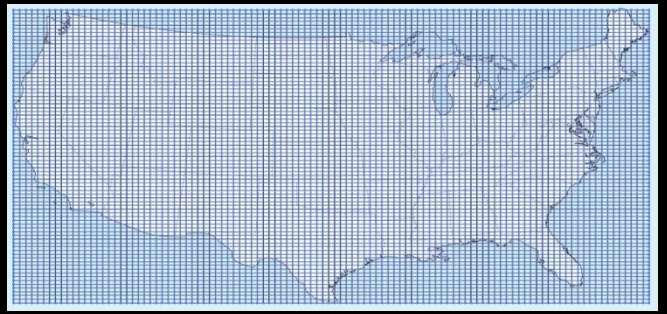


Weak vs. Strong scaling

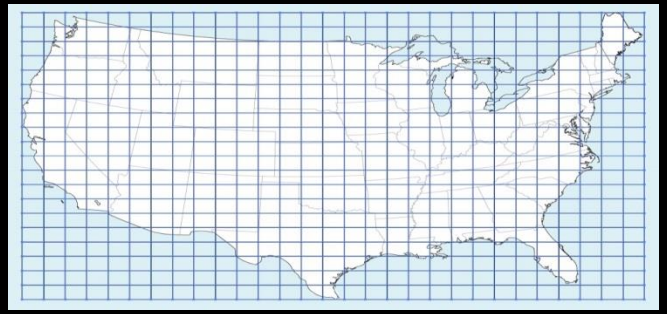


More Processors

Weak Scaling

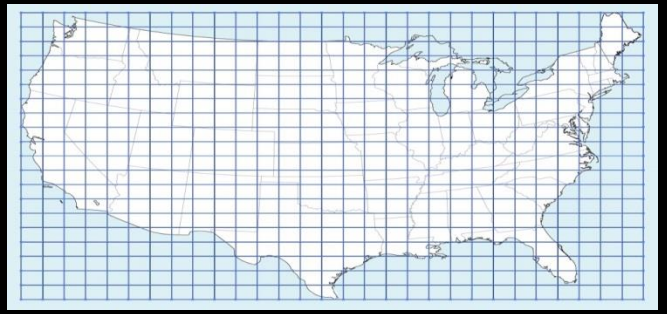


More accurate results



More Processors

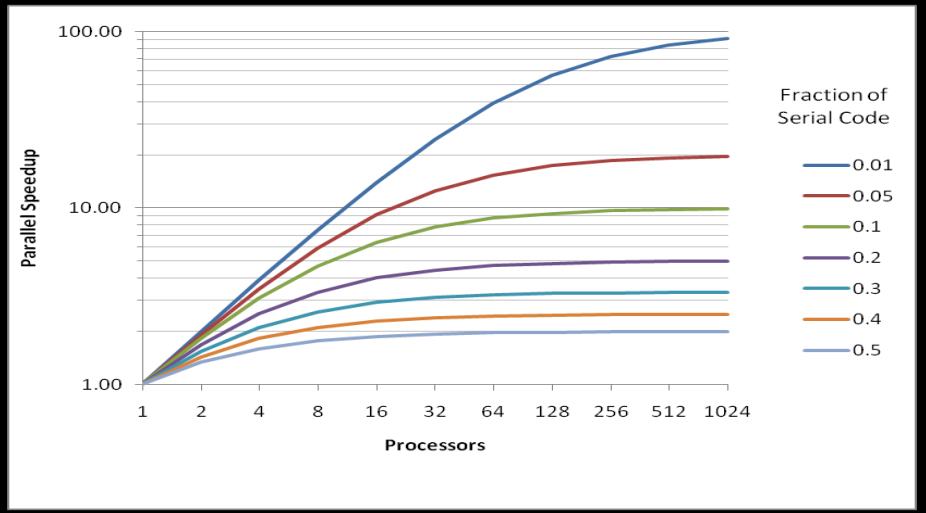
Strong Scaling



Faster results
(Tornado on way!)

Amdahl's Law

- If there is x% of serial component, speedup cannot be better than 100/x.
- If you decompose a problem into many parts, then the parallel time cannot be less than the largest of the parts.
- If the critical path through a computation is T, you cannot complete in less time than T, no matter how many processors you use .



MPI as an answer to an emerging problem ?!

In the Intro we mentioned that we are at a historic crossover where the cost of even on-chip data movement is surpassing the cost of computation.

MPI codes explicitly acknowledge and manage data locality and movement (communication).

Both this paradigm, and quite possible outright MPI, offer the only immediate solution. You and your programs may find a comfortable future in the world of massive multi-core.

This is a somewhat recent realization in the most *avant-garde* HPC circles. Amaze your friends with your insightful observations!