Parallel Computing
&
Accelerators

John Urbanic

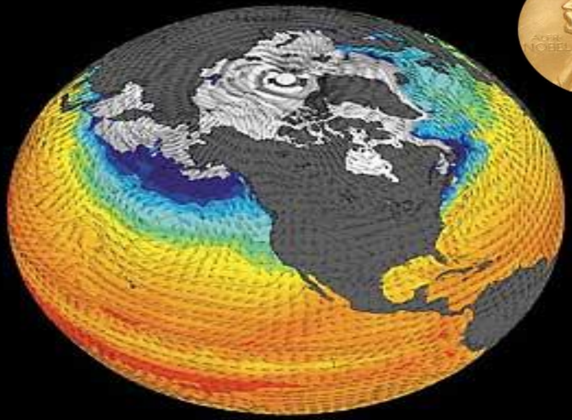
Parallel Computing Scientist
Pittsburgh Supercomputing Center

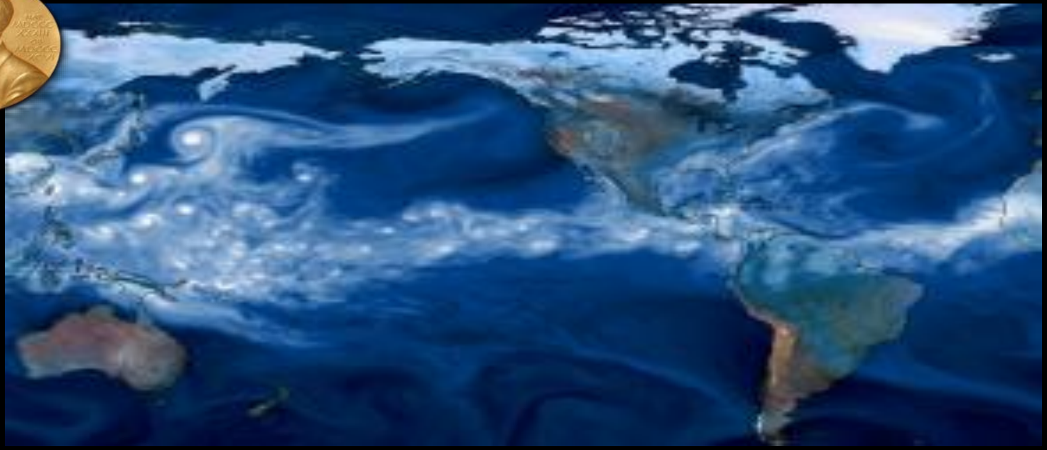
Distinguished Service Professor
Carnegie Mellon University

Purpose of this talk

This is the 50,000 ft. view of the parallel computing landscape. We want to orient you a bit before parachuting you down into the trenches to deal with OpenACC. The plan is that you walk away with a knowledge of not just OpenACC, but also where it fits into the world of High Performance Computing.

FLOPS we need: Climate change analysis





Simulations

- Cloud resolution, quantifying uncertainty, understanding tipping points, etc., will drive climate to exascale platforms
- New math, models, and systems support will be needed

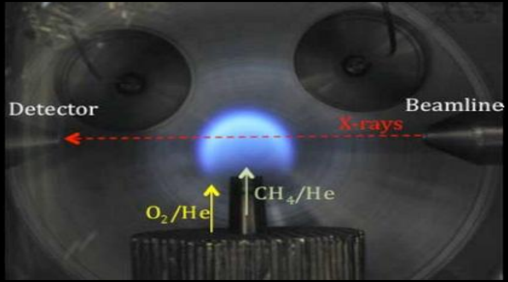
Extreme data

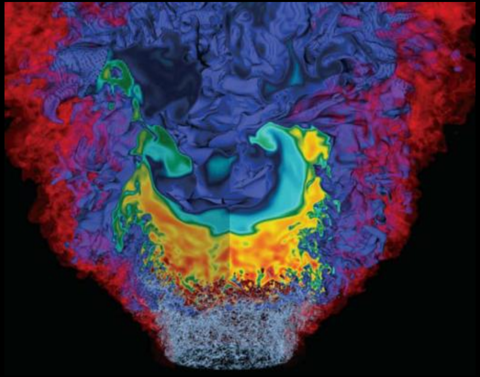
- “Reanalysis” projects need 100× more computing to analyze observations
- Machine learning and other analytics are needed today for petabyte data sets
- Combined simulation/observation will empower policy makers and scientists

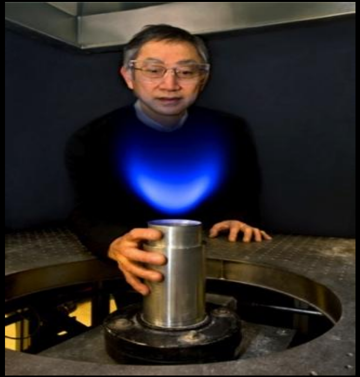
Courtesy Horst Simon, LBNL

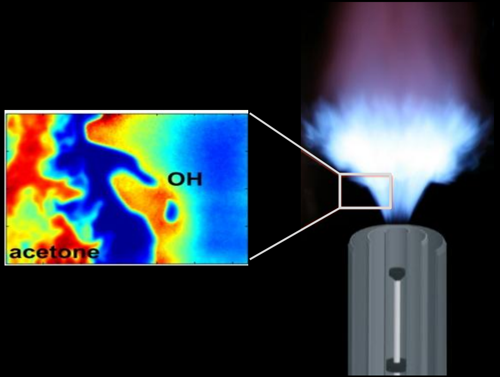
Exascale combustion simulations

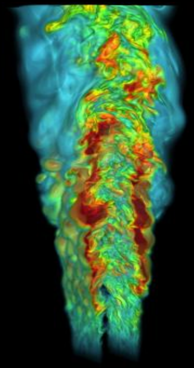
- Goal: 50% improvement in engine efficiency
- Center for Exascale Simulation of Combustion in Turbulence (ExaCT)
  - Combines M&S and experimentation
  - Uses new algorithms, programming models, and computer science







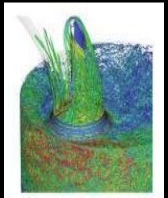


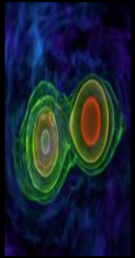


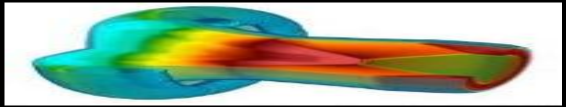
Courtesy Horst Simon, LBNL

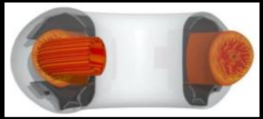
The list is long, and growing.

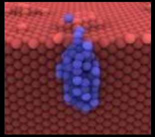
- Molecular-scale Processes: atmospheric aerosol simulations
- AI-Enhanced Science: predicting disruptions in tokomak fusion reactors
- Hypersonic Flight
- Modeling Thermonuclear X-ray Bursts: 3D simulations of a neutron star surface or supernovae
- Quantum Materials Engineering: electrical conductivity photovoltaic and plasmonic devices
- Physics of Fundamental Particles: mass estimates of the bottom quark
- Digital Cells

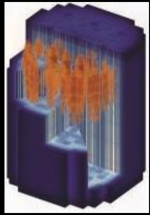


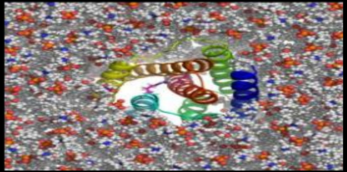


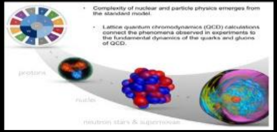












And many of you doubtless brought your own immediate research concerns. Great!

Welcome to The Exascale Era!

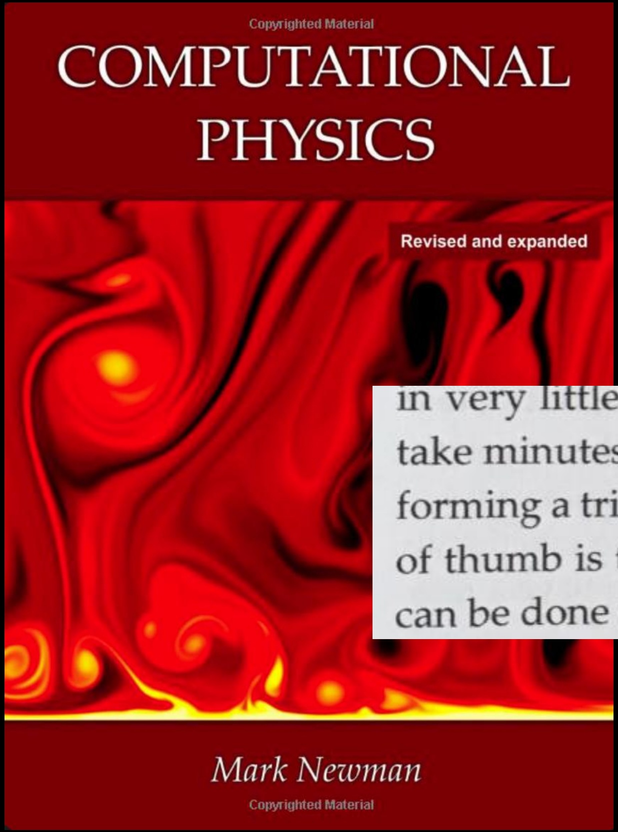
exa = 10^{18} = 1,000,000,000,000,000,000 = quintillion

64-bit precision floating point operations per second

23,800
Cray Red Storms
2004 (42 Tflops)

133,33
NVIDIA V100
(7.5 Tflops)

There may also be a Chinese machine, OceanLight, or 3-letter-agency machines on the scene.



in very little time. Performing a billion operations, on the other hand, could take minutes or hours, though it's still possible provided you are patient. Performing a trillion operations, however, will basically take forever. So a fair rule of thumb is that the calculations we can perform on a computer are ones that can be done with *about a billion operations or less.*

Where are those 10 or 12 orders of magnitude?

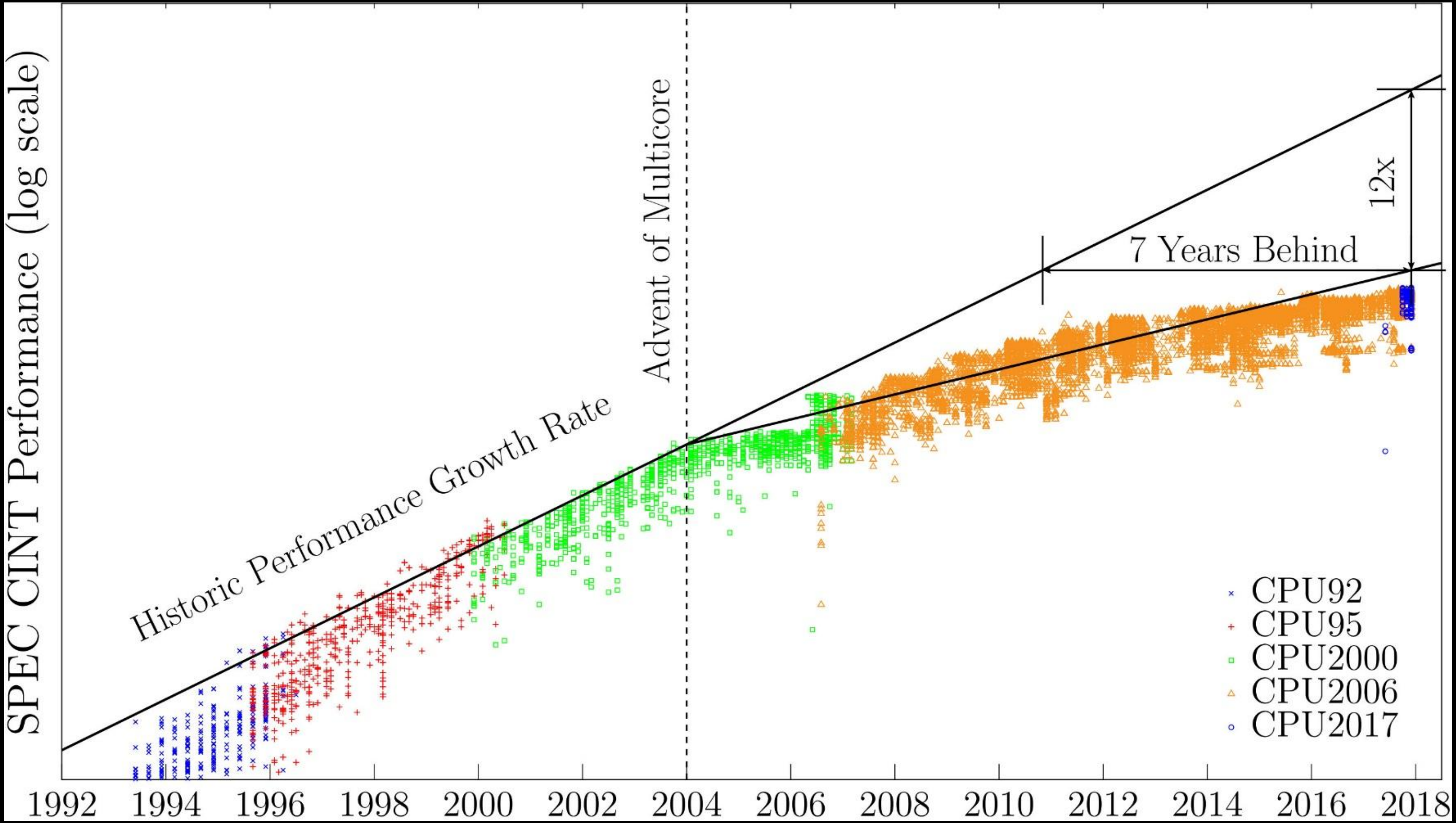
How do we get there from here?

BTW, that's a bigger gap than

vs.

IBM 709
12 kiloflops

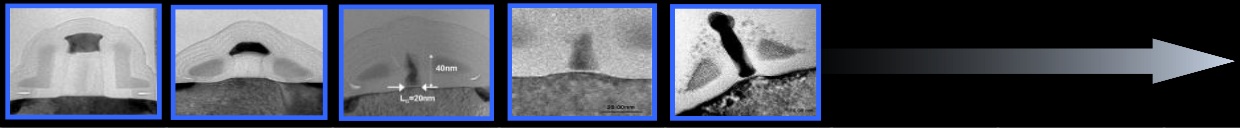
Moore's Law abandoned serial programming around 2004



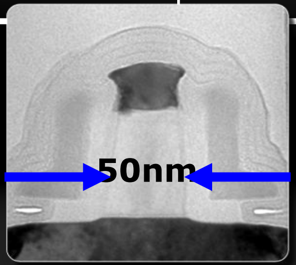
Courtesy Liberty Computer Architecture Research Group

But Moore's Law is only beginning to stumble now.

Intel process technology capabilities

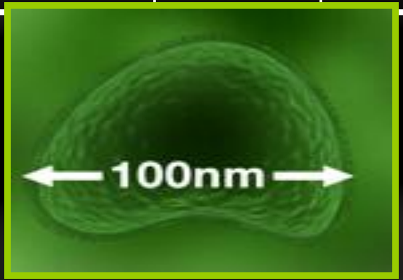


High Volume Manufacturing	2004	2006	2008	2010	2012	2014	2018	2021
Feature Size	90nm	65nm	45nm	32nm	22nm	14nm	10nm	7nm
Integration Capacity (Billions of Transistors)	2	4	8	16	32	64	128	256



Transistor for 90nm Process

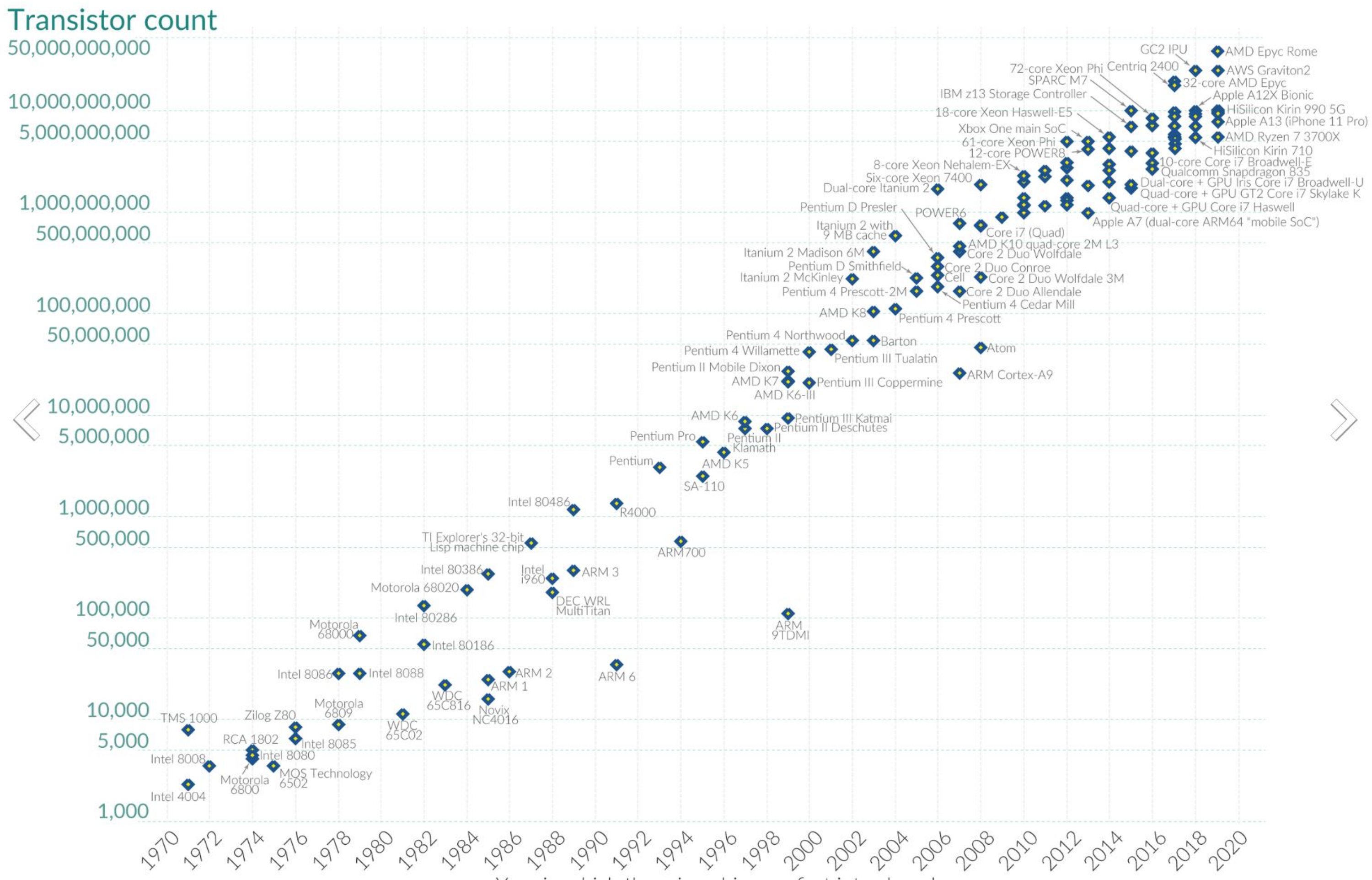
Source: Intel



Influenza Virus

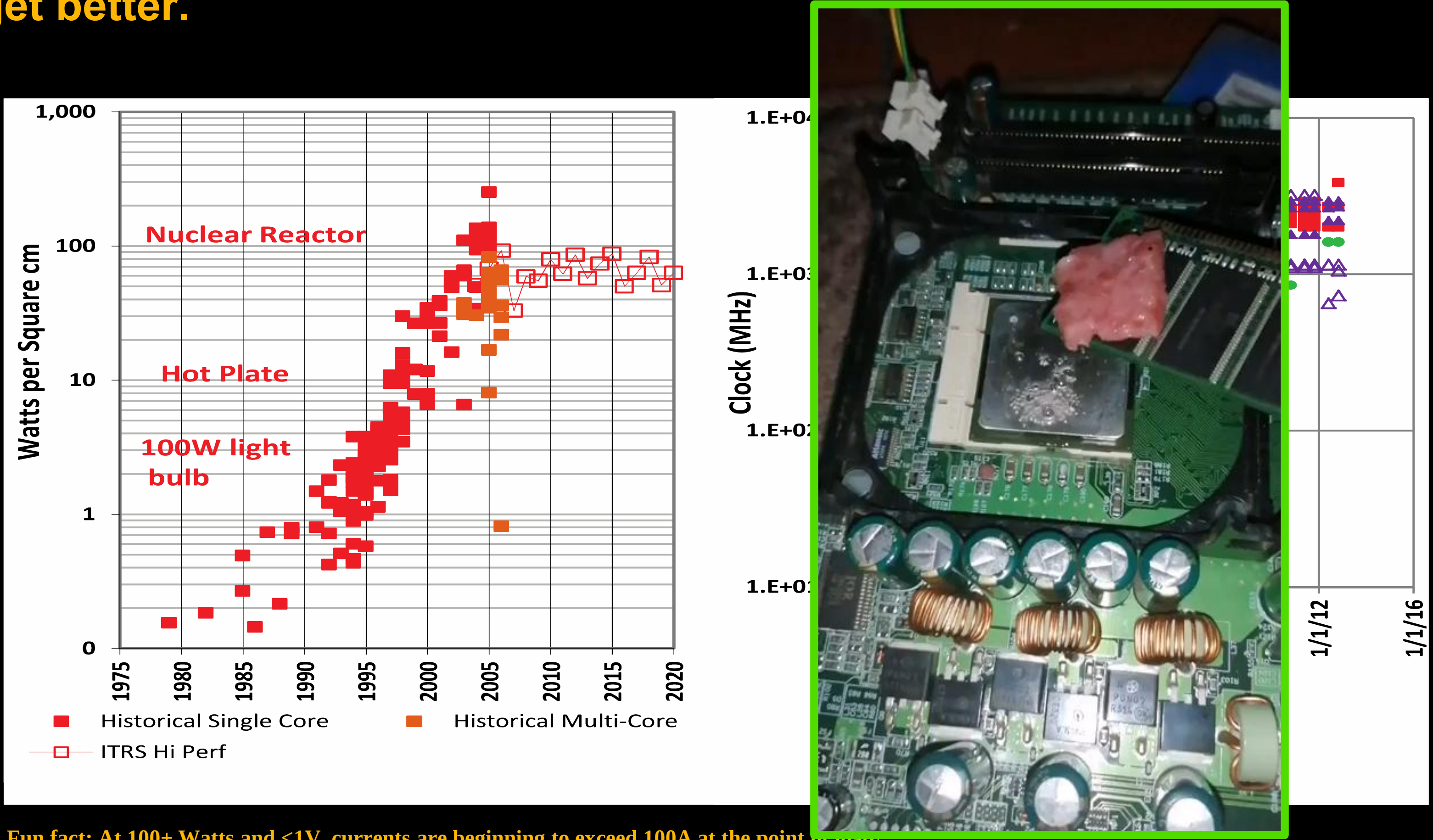
Source: CDC

And at end of day we keep using getting more transistors.



Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldinData.org – Research and data to make progress against the world's largest problems.

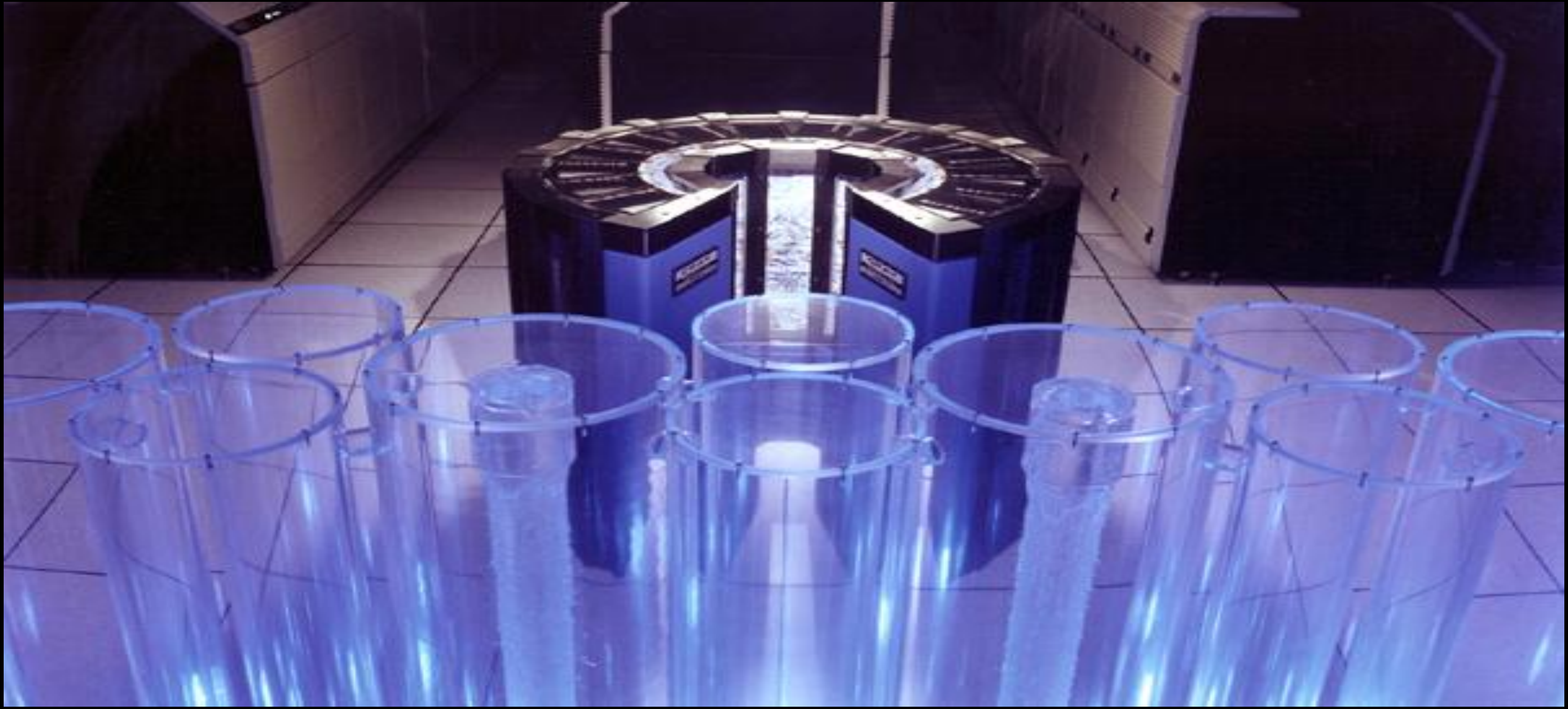
That Power and Clock Inflection Point in 2004…
didn't get better.



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

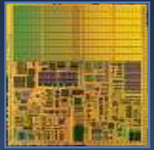
And how to get more performance from more transistors with the same power.

A 15% Reduction In Voltage Yields

RULE OF THUMB

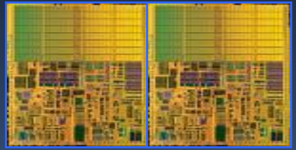
Frequency Reduction	Power Reduction	Performance Reduction
15%	45%	10%

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

Parallel Computing

One woman can make a baby in 9 months.

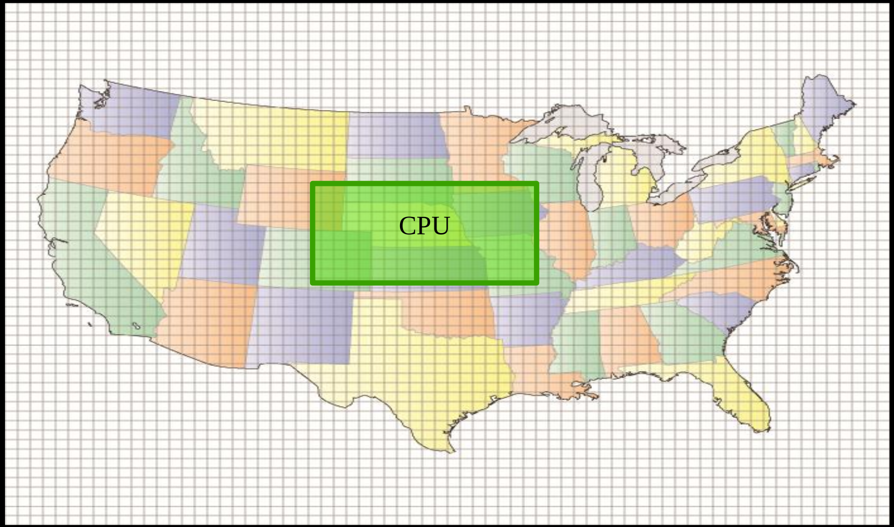
Can 9 women make a baby in 1 month?

But 9 women can make 9 babies in 9 months.

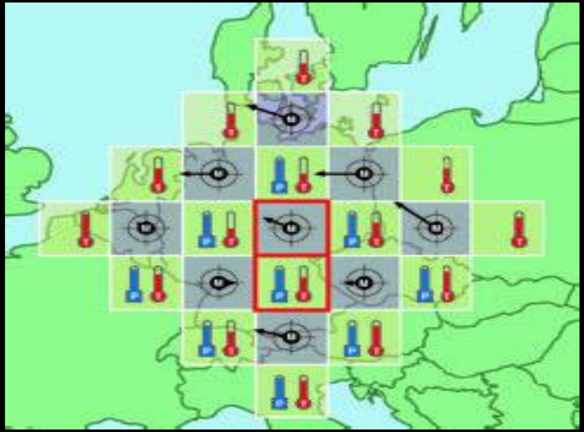
First two bullets are Brook's Law. From *The Mythical Man-Month.*

A must-read for serious project programmers that includes many other classics such as:
"What one programmer can do in one month, two programmers can do in two months."

Prototypical Application: Serial Weather Model

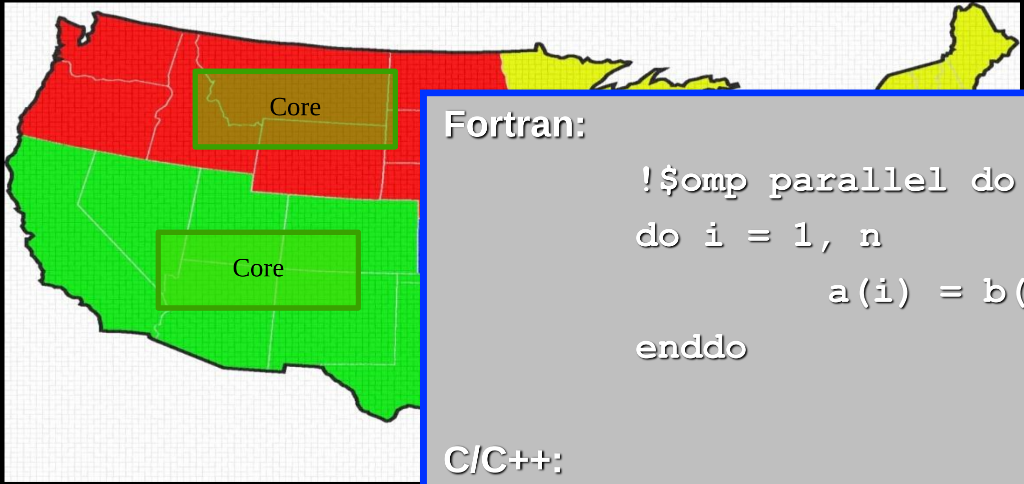


First Parallel Weather Modeling Algorithm: Richardson in 1917



Courtesy John Burkhardt, Virginia Tech

Weather Model: Shared Memory (OpenMP)



Fortran:

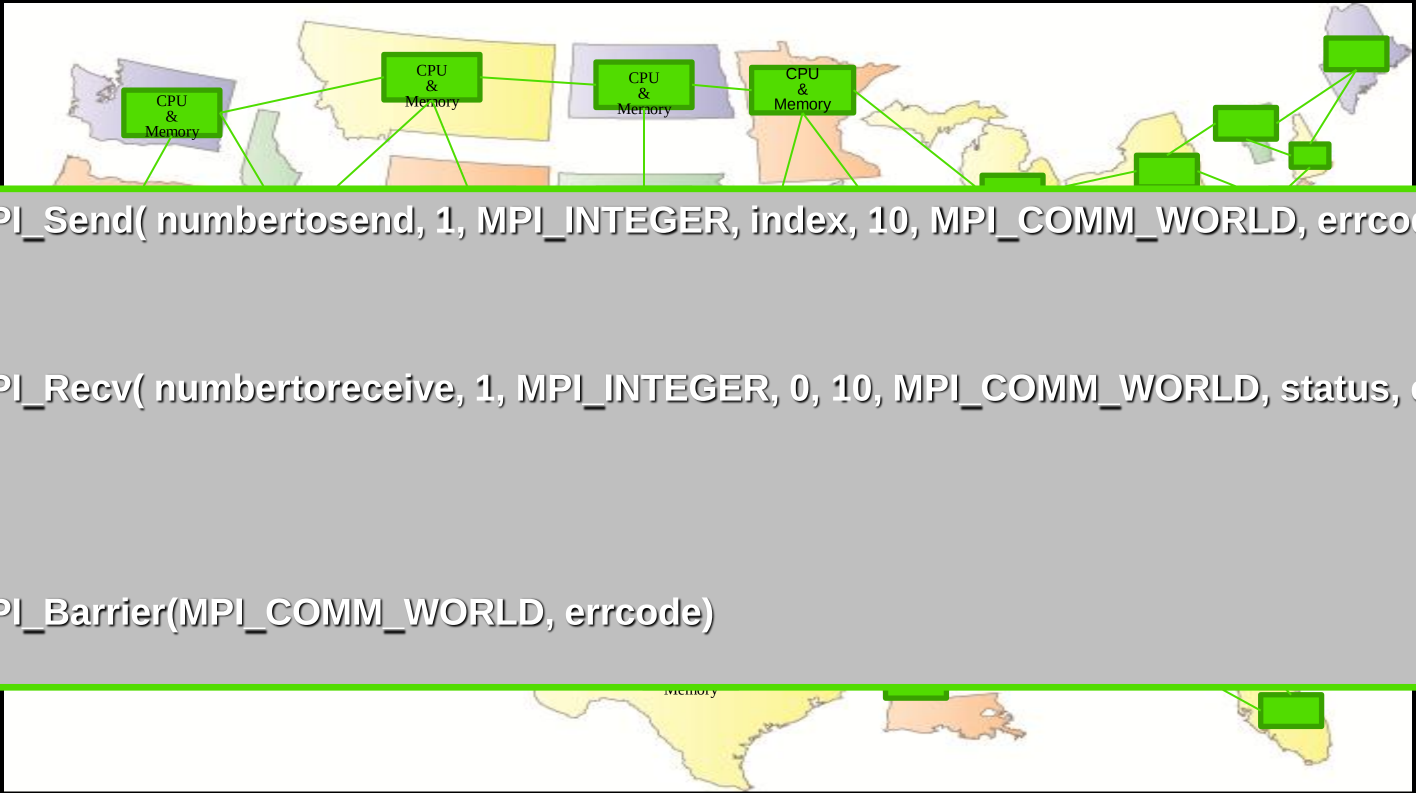
```
!$omp parallel do
do i = 1, n
        a(i) = b(i) + c(i)
enddo
```

C/C++:

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

Four meteorologists in th

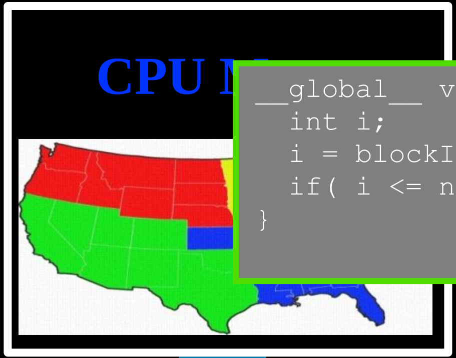
Weather Model: Distributed Memory (MPI)



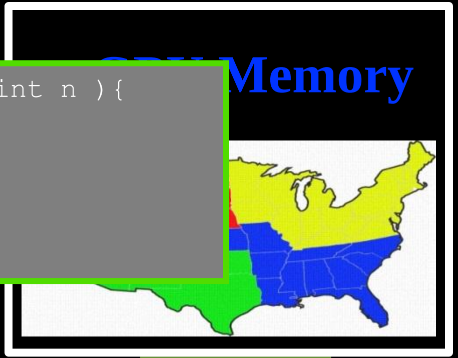

```
call MPI_Send( numbertosend, 1, MPI_INTEGER, index, 10, MPI_COMM_WORLD, errcode)
.
.
call MPI_Recv( numbertoreceive, 1, MPI_INTEGER, 0, 10, MPI_COMM_WORLD, status, errcode)
.
.
.
call MPI_Barrier(MPI_COMM_WORLD, errcode)
.
```

50 meteorologists using a telegraph.

Weather Model: Accelerator (OpenACC)



```
__global__ void saxpy_kernel( float a, float* x, float* y, int n ){
  int i;
  i = blockIdx.x*blockDim.x + threadIdx.x;
  if( i <= n ) x[i] = a*x[i] + y[i];
}
```



CPU

GPU

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

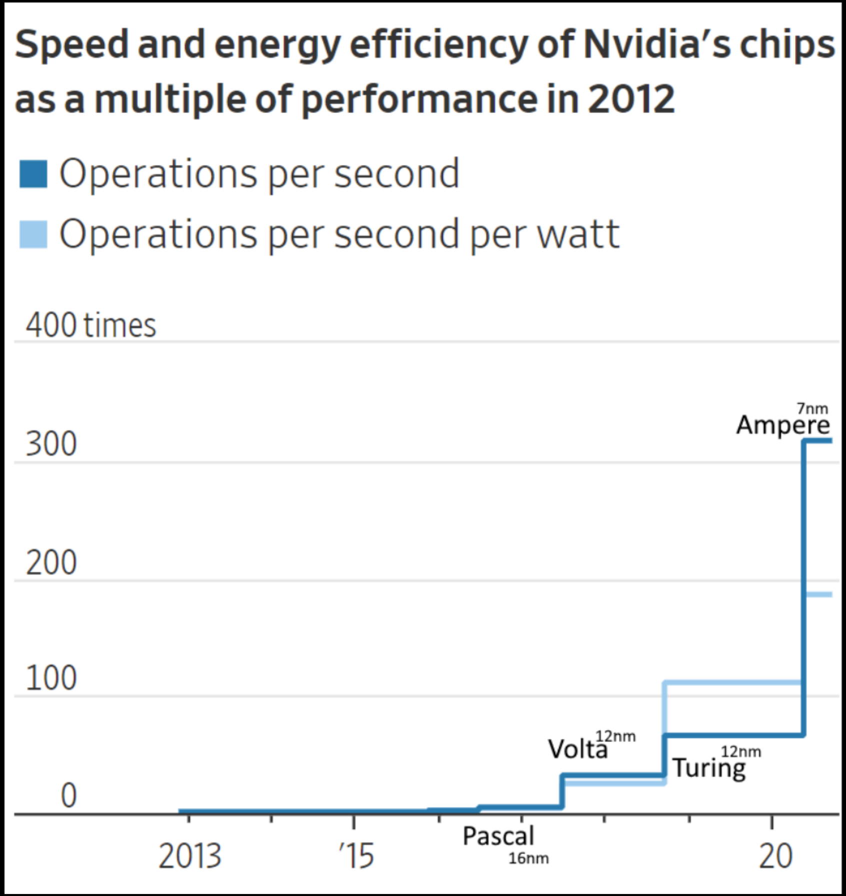
Huang's Law

An observation/claim made by Jensen Huang, CEO of Nvidia, at its 2018 GPU Technology Conference.

He observed that Nvidia’s GPUs were "25 times faster than five years ago" whereas Moore's law would have expected only a ten-fold increase.

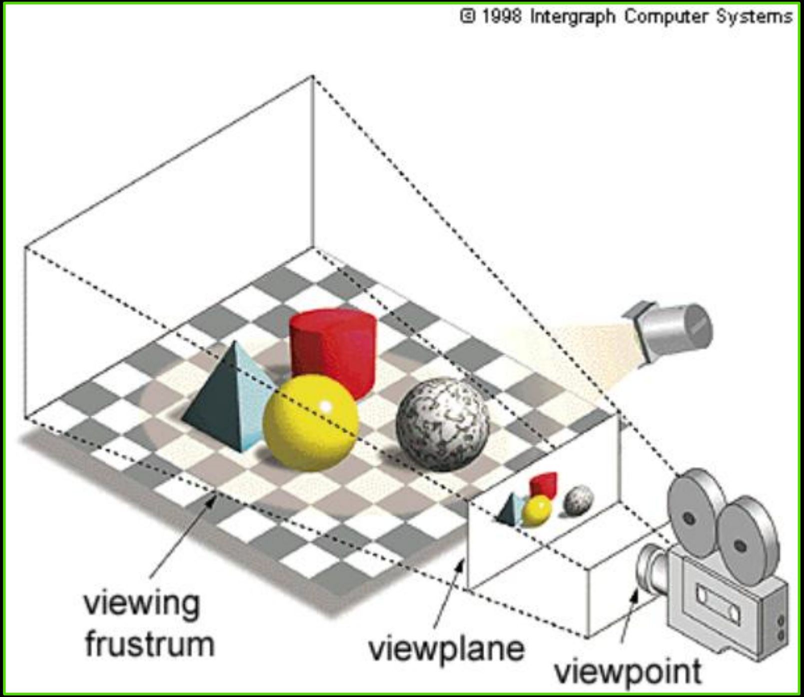
In 2006 Nvidia's GPU had a 4x performance advantage over other CPUs. In 2018 the Nvidia GPU was 20 times faster than a comparable CPU node: the GPUs were 1.7x faster each year. Moore’s law would predict a doubling every two years, however Nvidia's GPU performance was more than tripled every two years fulfilling Huang's law.

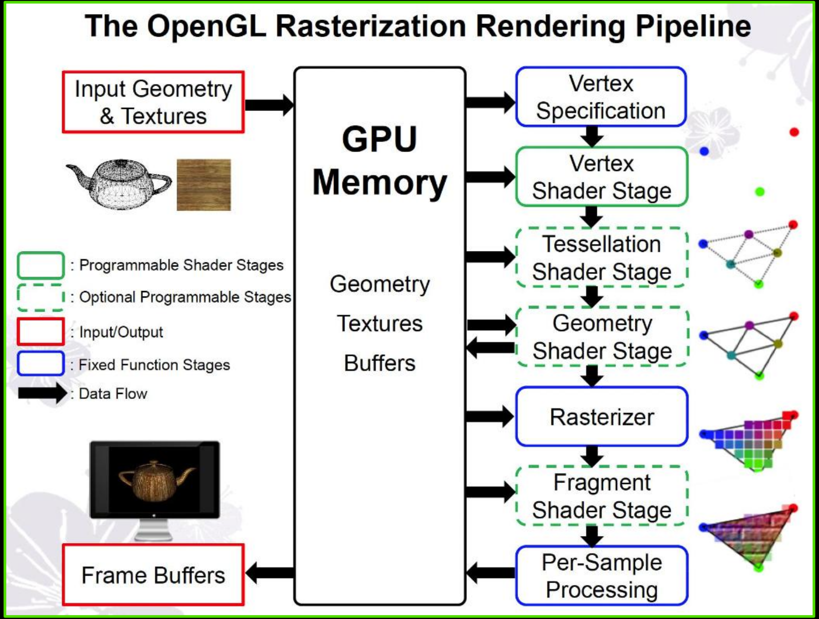
It is a little premature, and there are confounding factors at play, so most people haven't yet elevated this to the status of Moore's Law.



Source: NVIDIA

Why Video Gaming Cards?

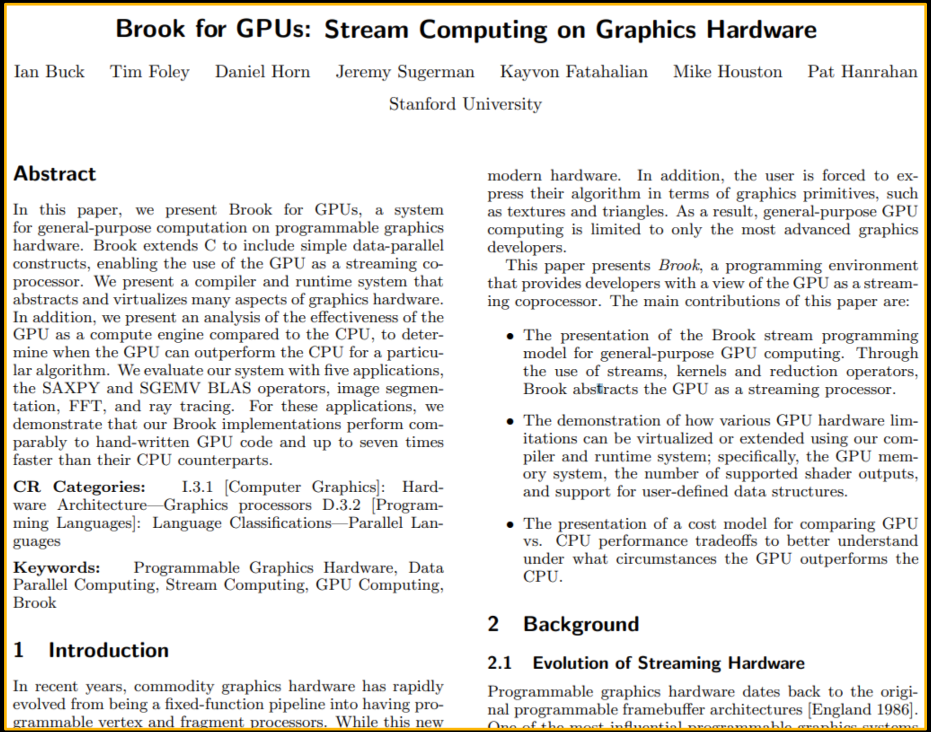




By the turn of the century, the video gaming market has already standardized around a few APIs for rendering 3D video games in real-time.

None of these looked anything like scientific computing.

Heroic Efforts



Brook for GPUs: Stream Computing on Graphics Hardware

Ian Buck Tim Foley Daniel Horn Jeremy Sugerman Kayvon Fatahalian Mike Houston Pat Hanrahan

Stanford University

Abstract

In this paper, we present Brook for GPUs, a system for general-purpose computation on programmable graphics hardware. Brook extends C to include simple data-parallel constructs, enabling the use of the GPU as a streaming coprocessor. We present a compiler and runtime system that abstracts and virtualizes many aspects of graphics hardware. In addition, we present an analysis of the effectiveness of the GPU as a compute engine compared to the CPU, to determine when the GPU can outperform the CPU for a particular algorithm. We evaluate our system with five applications, the SAXPY and SGEMV BLAS operators, image segmentation, FFT, and ray tracing. For these applications, we demonstrate that our Brook implementations perform comparably to hand-written GPU code and up to seven times faster than their CPU counterparts.

CR Categories: I.3.1 [Computer Graphics]: Hardware Architecture—Graphics processors D.3.2 [Programming Languages]: Language Classifications—Parallel Languages

Keywords: Programmable Graphics Hardware, Data Parallel Computing, Stream Computing, GPU Computing, Brook

1 Introduction

In recent years, commodity graphics hardware has rapidly evolved from being a fixed-function pipeline into having programmable vertex and fragment processors. While this new modern hardware. In addition, the user is forced to express their algorithm in terms of graphics primitives, such as textures and triangles. As a result, general-purpose GPU computing is limited to only the most advanced graphics developers.

This paper presents *Brook*, a programming environment that provides developers with a view of the GPU as a streaming coprocessor. The main contributions of this paper are:

- The presentation of the Brook stream programming model for general-purpose GPU computing. Through the use of streams, kernels and reduction operators, Brook abstracts the GPU as a streaming processor.
- The demonstration of how various GPU hardware limitations can be virtualized or extended using our compiler and runtime system; specifically, the GPU memory system, the number of supported shader outputs, and support for user-defined data structures.
- The presentation of a cost model for comparing GPU vs. CPU performance tradeoffs to better understand under what circumstances the GPU outperforms the CPU.

2 Background

2.1 Evolution of Streaming Hardware

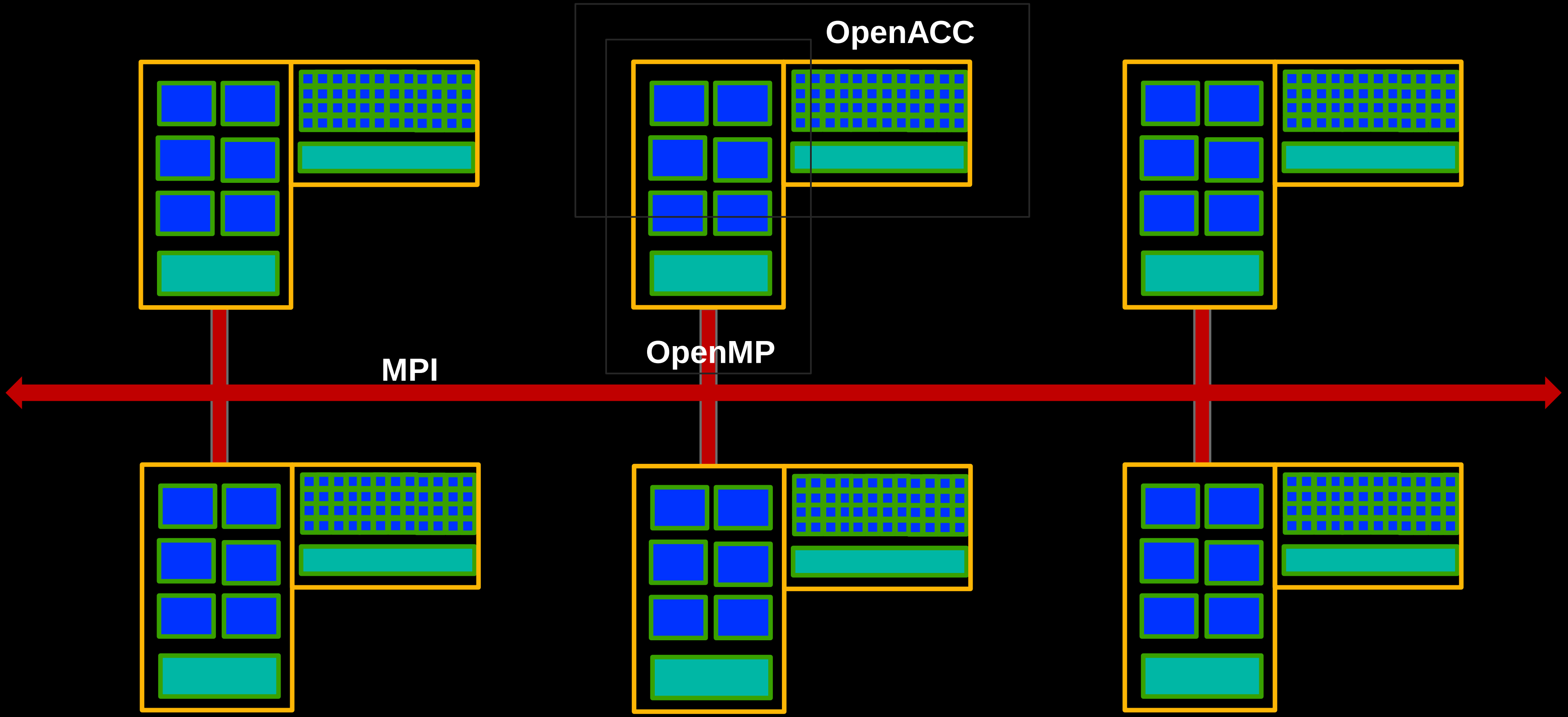
Programmable graphics hardware dates back to the original programmable framebuffer architectures [England 1986].

An API in 2004 first demonstrated the potential use of this latent floating point ability.

By 2007 NVIDIA supported a dedicated API for their own hardware.

Note that these early devices were not at all engineered for scientific computing and lacked several very fundamental capabilities. In particular EEC and double precision.

The pieces fit like this…



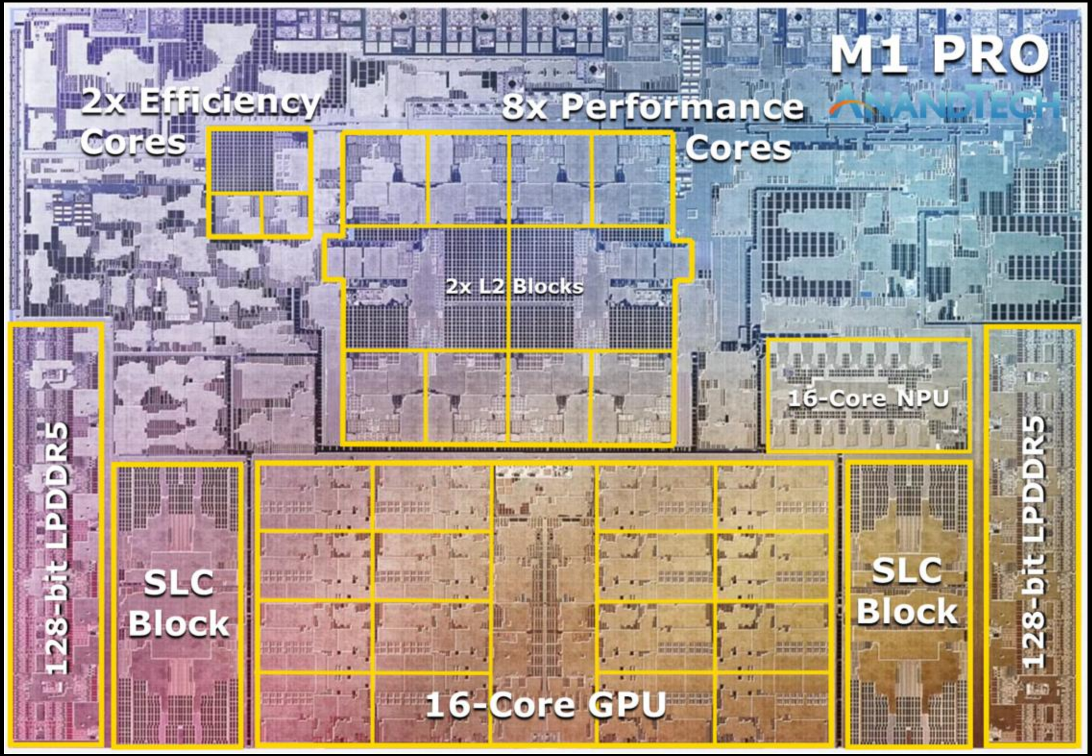
Top 10 Systems as of November 2024

#	Computer	Site	Manufacturer	CPU Interconnect [*Accelerator*]	Cores	Rmax (Pflops)	Rpeak (Pflops)	Power (MW)
1	**El Capitan**	Lawrence Livermore National Laboratory **United States**	HPE	AMD EPYC 24C 1.8GHz Slingshot-11 AMD Instinct MI300A	**11,039,616**	**1742**	**2746**	**30**
2	**Frontier**	Oak Ridge National Laboratory **United States**	HPE	AMD EPYC 64C 2GHz Slingshot-11 AMD Instinct MI250X	**9,066,176**	**1353**	**2055**	**25**
3	**Aurora**	Argonne National Laboratory **United States**	HPE	Intel Xeon Max 9470 52C 2.4GHz Slingshot-11 Intel Data Center GPU Max	**9,264,128**	**1012**	**1980**	**39**
4	**Eagle**	Microsoft **United States**	Microsoft	Intel Xeon 8480C 48C 2GHz Infiniband NDR NVIDIA H100	**1,123,200**	**561**	**846**	
5	**HPC6**	Eni S.p.A. **Italy**	HPE	AMD EPYC 64C 2GHz Slingshot-11 AMD Instinct MI250X	**3,143,520**	**477**	**606**	**8**
6	**Fugaku**	RIKEN Center for Computational Science **Japan**	Fujitsu	ARM 8.2A+ 48C 2.2GHz Torus Fusion Interconnect	**7,630,072**	**442**	**537**	**29**
7	**Alps**	Swiss National Supercomputing Center **Switzerland**	HPE	NVIDIA Grace 72C 3.1GHz Slingshot-11 *NVIDIA GH200*	**2,121,600**	**434**	**574**	**7**
8	**LUMI**	EuroHPC **Finland**	HPE	AMD EPYC 64C 2GHz Slingshot-11 AMD Instinct MI250X	**2,752,704**	**379**	**531**	**7**
9	**Leonardo**	EuroHPC **Italy**					**304**	**7**
10	**Tuolumne**	Lawrenc National					**288**	**3**

500	ThinkSystem SR590, Xeon Gold 5218 16C 2.3GHz, 10G Ethernet, Lenovo Service Provider T China	108,800	2.31	4.00

The word is *Heterogeneous*

And it's not just supercomputers. It's on your desk, and in your phone.



AnandTech

How much of this can you program?

We can do better. We have a role model.

- We hope to "simulate" a human brain in real time on one of these Exascale platforms with about 1 - 10 Exaflop/s and 4 PB of memory
- These newest Exascale computers use 20+ MW
- The human brain runs at 20W
- Our brain is a million times more power efficient!

Why you should be (extra) motivated.

- This parallel computing thing is no fad.
- The laws of physics are drawing this roadmap.
- If you get on board (the right bus), you can ride this trend for a long, exciting trip.

Let’s learn how to use these things!