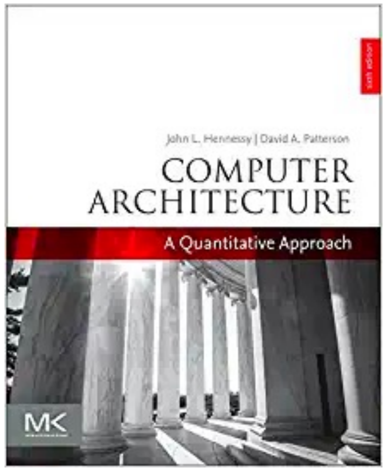


Chapter 1

Fundamentals of Quantitative Design and Analysis

Part 3: Benchmarking & Technology Trends

"I think it's fair to say that personal computers have become the most empowering tool we've ever created. They're tools of communication they're tools of creativity, and they can be shaped by their user." – Bill Gates, February 2004

Acknowledgements

- Thanks to many sources for slide material

  © 1990 Morgan Kaufmann Publishers, © 2001-present Elsevier
  Computer Architecture: A Quantitative Approach by J. Hennessy & D. Patterson

  © 1994 Morgan Kaufmann Publishers, © 2001-present Elsevier
  Computer Organization and Design by D. Patterson & J. Hennessy

  © 2002 K. Asinovic & Arvind, MIT

  © 2002 J. Kubiatowicz, University of California at Berkeley

  © 2006, © 2010 No Starch Press for Inside the Machine by J. Stokes

  © 2007 W.-M. Hwu & D. Kirk, University of Illinois & NVIDIA

  © 2007-2010 J. Owens, University of California at Davis

  © 2010 CRC Press for Introduction to Concurrency in Programming Languages by M. Sottile, T. Mattson, and C. Rasmussen

  © 2017, IBM POWER9 Processor Architecture by Sadasivam et al., IBM

  © 2016, © 2019 POWER9 Processor User's Manual, IBM

  © The OpenPOWER Foundation

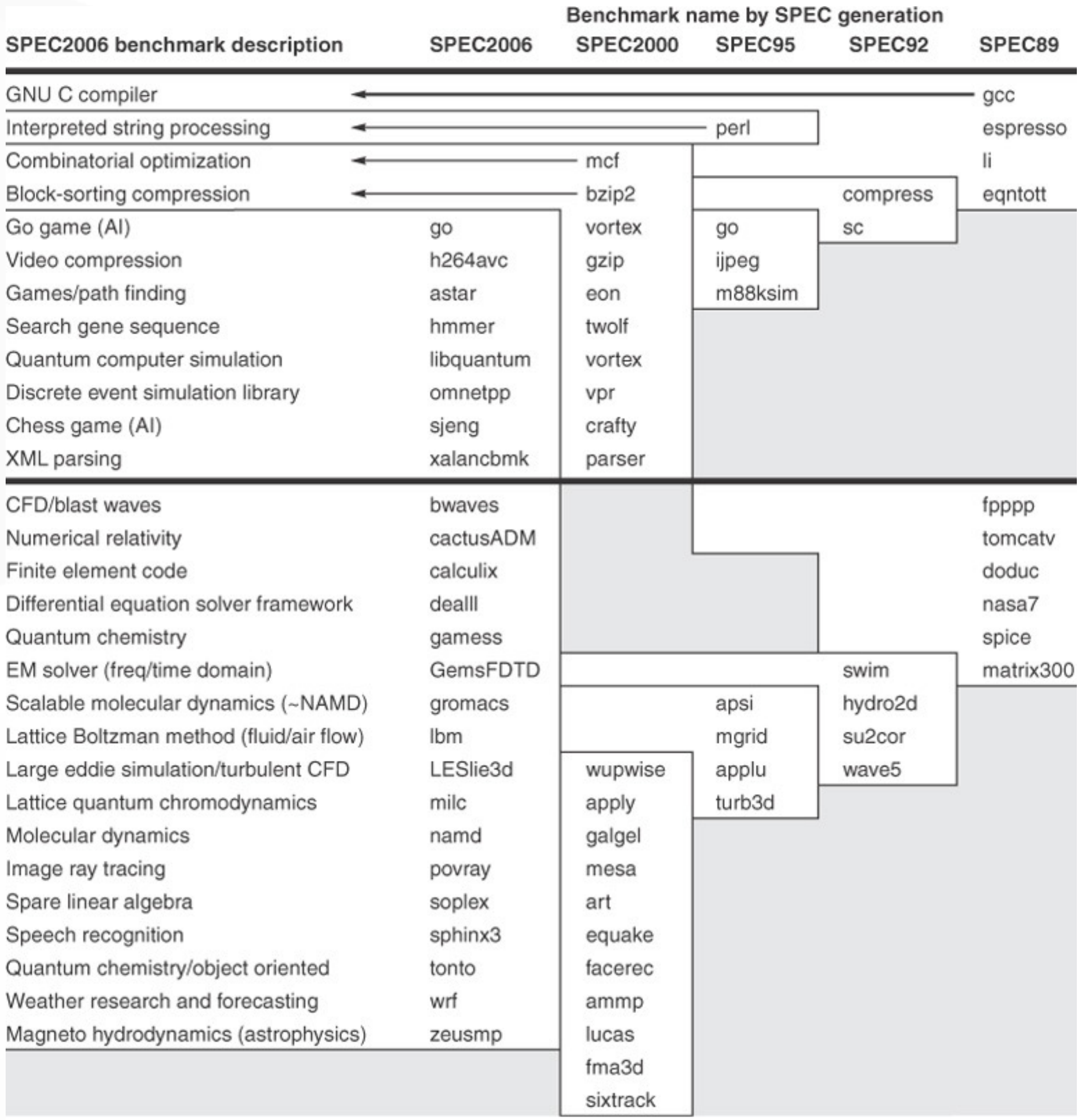
Outline

- Benchmarks
- Technology
  - In a power-constrained world, we can’t keep increasing performance by increasing clock speed.
- Architectural Trends
  - Microarchitectures have historically increased performance by increasing ILP and pipeline depth. Today, we can’t keep increasing performance at historical levels by these methods.
- Bandwidth vs. Latency

What makes a good benchmark?

What do you want to avoid in creating benchmarks?

SPEC Benchmarks



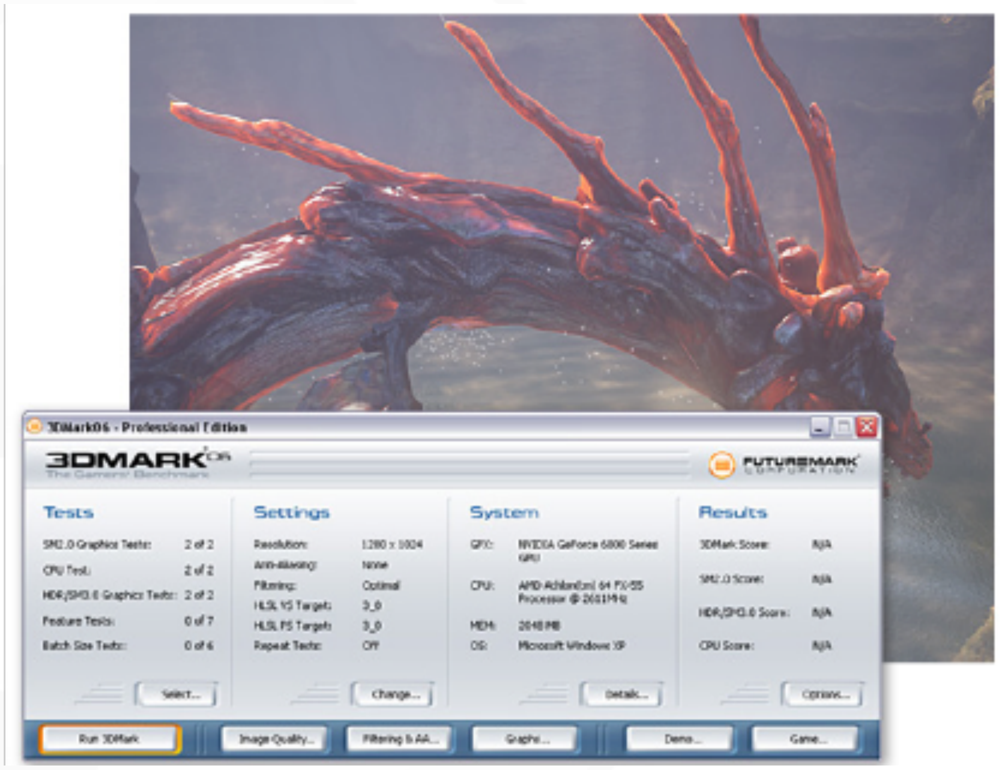
	Benchmark name by SPEC generation				
SPEC2006 benchmark description	SPEC2006	SPEC2000	SPEC95	SPEC92	SPEC89
GNU C compiler	←	←	←	←	gcc
Interpreted string processing	←	←	perl		espresso
Combinatorial optimization	←	mcf			li
Block-sorting compression	←	bzip2		compress	eqntott
Go game (AI)	go	vortex	go	sc	
Video compression	h264avc	gzip	ijpeg		
Games/path finding	astar	eon	m88ksim		
Search gene sequence	hmmer	twolf			
Quantum computer simulation	libquantum	vortex			
Discrete event simulation library	omnetpp	vpr			
Chess game (AI)	sjeng	crafty			
XML parsing	xalancbmk	parser			
CFD/blast waves	bwaves				fpppp
Numerical relativity	cactusADM				tomcatv
Finite element code	calculix				doduc
Differential equation solver framework	dealll				nasa7
Quantum chemistry	gamess				spice
EM solver (freq/time domain)	GemsFDTD			swim	matrix300
Scalable molecular dynamics (~NAMD)	gromacs		apsi	hydro2d	
Lattice Boltzman method (fluid/air flow)	lbm		mgrid	su2cor	
Large eddie simulation/turbulent CFD	LESlie3d	wupwise	applu	wave5	
Lattice quantum chromodynamics	milc	apply	turb3d		
Molecular dynamics	namd	galgel			
Image ray tracing	povray	mesa			
Spare linear algebra	soplex	art			
Speech recognition	sphinx3	equake			
Quantum chemistry/object oriented	tonto	facerec			
Weather research and forecasting	wrf	ammp			
Magneto hydrodynamics (astrophysics)	zeusmp	lucas			
		fma3d			
		sixtrack			

SPEC Discussion

- From SPECx to SPECy, why would we remove a benchmark of one type and add a benchmark of the same type?

- Can a particular problem be solved in more than one way? Example: Solving a system of equations.
  - Should a benchmark reflect “solve the problem using this particular algorithm” or “solve the problem using any algorithm”?

3DMark06 Benchmark



Tests

SM2.0 Graphics Tests:	2 of 2
CPU Test:	2 of 2
HDR/SM3.0 Graphics Tests:	2 of 2
Feature Tests:	0 of 7
Batch Size Tests:	0 of 6

Settings

Resolution:	1280 x 1024
Anti-Aliasing:	None
Filtering:	Optimal
HLSL VS Target:	3_0
HLSL PS Target:	3_0
Repeat Tests:	Off

System

GFX:	NVIDIA GeForce 6800 Series GPU
CPU:	AMD Athlon(tm) 64 FX-55 Processor @ 2601MHz
MEM:	2048 MB
OS:	Microsoft Windows XP

Results

3DMark Score:	N/A
SM2.0 Score:	N/A
HDR/SM3.0 Score:	N/A
CPU Score:	N/A

- Historically, becomes more CPU-bound over time
- Graphics (4) and CPU (2) benchmarks in 3DMark06

TPC-C Benchmark

“The difficulty in designing TPC benchmarks lies in reducing the diversity of operations found in a production application, while retaining its essential performance characteristics, namely, the level of system utilization and the complexity of its operations.”

Overview of the TPC Benchmark C:
The Order-Entry Benchmark

By Francois Raab, Walt Kohler, Amitabh Shah

TPC-C Overview

- *n* Warehouses
  - Each warehouse, 10 sales districts, terminal per sales district
  - Each sales district, 3000 customers
- New Order
  - 10 items, 10% chance from another warehouse
- Other Transactions
  - Payment, order status, delivery, stock query
- TPC-C Measures
  - Orders per second (tpmC)
  - Price-performance ($/tpmC)

More Benchmark Suites … Really?

- BabelStream, GPU-STREAM, STREAM
- ECP Proxy Apps
- FinanceBench
- Hetero-Mark
- Mantevo
- NPB-OCL
- OpenDwarfs
- Pannotia
- Parboil
- Polybench
- Rodinia
- SHOC
- ViennaCL

- **An Automated Framework for Characterizing and Subsetting GPGPU Workloads.**
  V. Adhinarayanan, W. Feng.
  In *Proc. of the IEEE Int'l Symp, on Performance Analysis of Systems and Software (ISPASS)*, Uppsala, Sweden, April 2016.
  Paper (PDF) Citations: [BibTeX XML PlainText]

Potential for

- Many research artifacts that extend the above.
- Many M.S. and even Ph.D. theses.

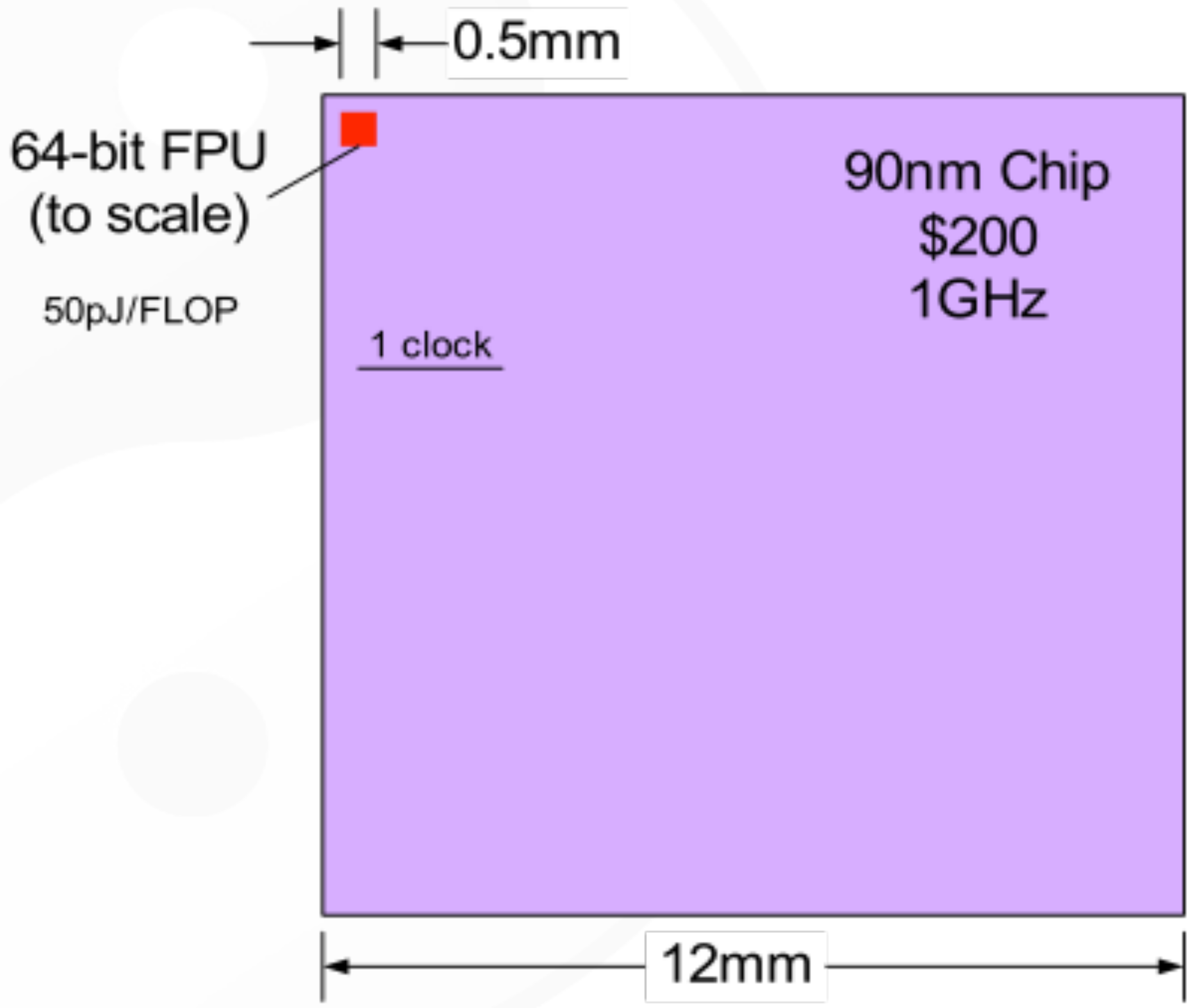
Outline

- Benchmarks
- Technology
  - In a power-constrained world, we can’t keep increasing performance by increasing clock speed.
- Architectural Trends
  - Microarchitectures have historically increased performance by increasing ILP and pipeline depth. Today, we can’t keep increasing performance at historical levels by these methods.
- Bandwidth vs. Latency

Technology

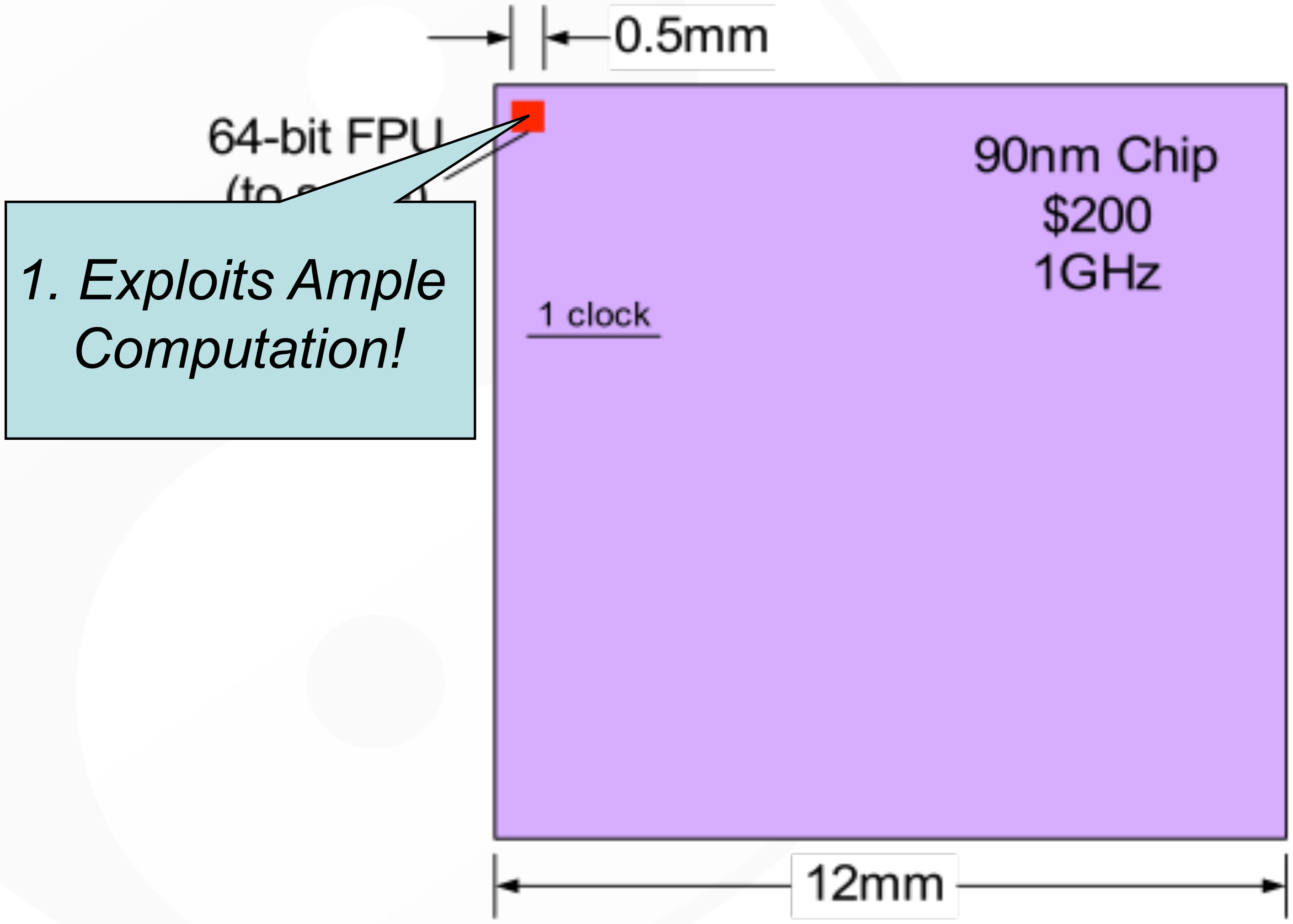
- As technology improves, what gets faster/bigger/better?
  - Gate delay
  - Chip size
  - Number of pins
  - Pin bandwidth
  - Memory bandwidth
  - Memory latency

VLSI Capability (2011)



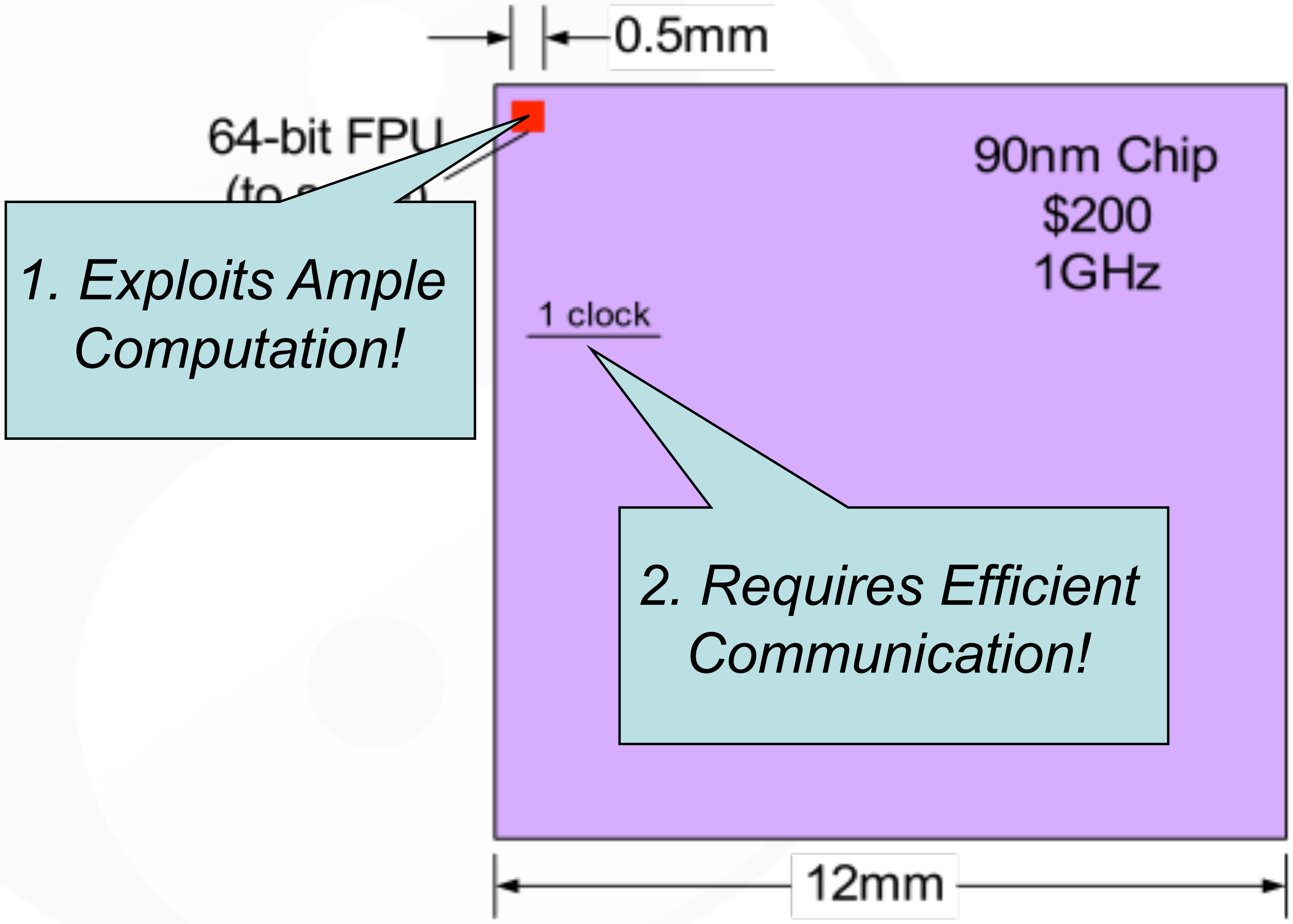
[Courtesy: Bill Dally]

Today's VLSI Capability



[Courtesy: Bill Dally]

Today's VLSI Capability

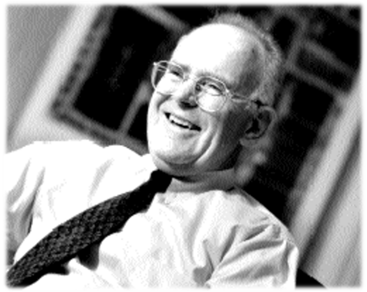

0.5mm
64-bit FPU
(to
90nm Chip
$200
1GHz
1 clock
12mm
1. Exploits Ample Computation!
2. Requires Efficient Communication!

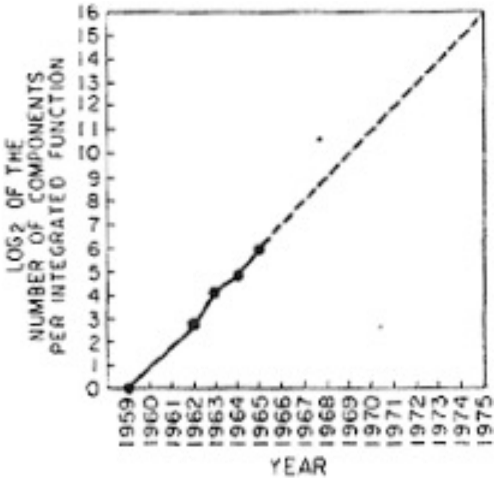
[Courtesy: Bill Dally]

Moore’s Law

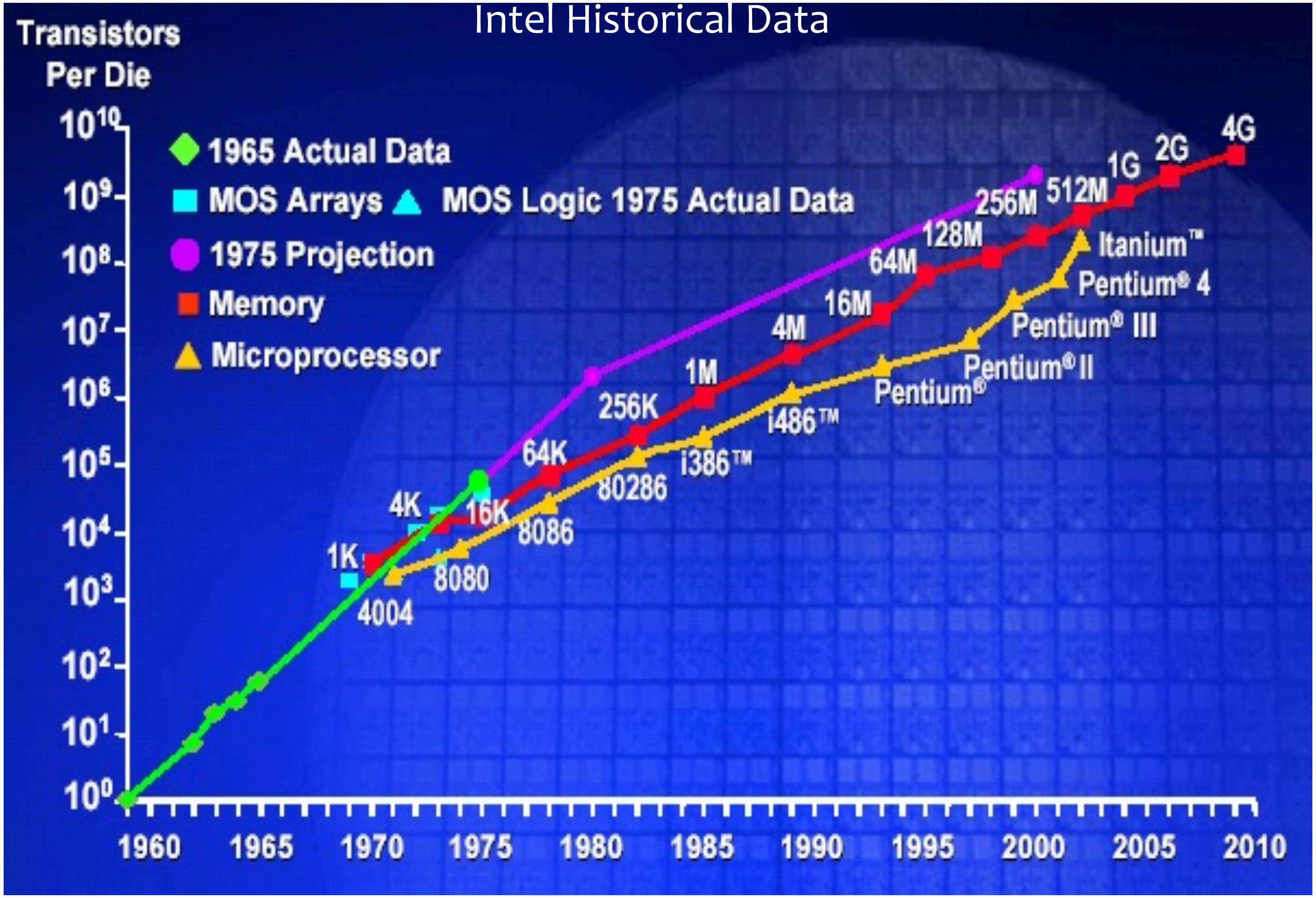
“The complexity for minimum component costs has increased at a rate of roughly a factor of two per year (see graph on next page). Certainly, over the short term this rate can be expected to continue, if not to increase. Over the longer term, the rate of increase is a bit more uncertain, although there is no reason to believe it will not remain nearly constant for at least 10 years. That means by 1975, the number of components per integrated circuit for minimum cost will be 65,000. I believe that such a large circuit can be built on a single wafer.”

“Cramming More Components onto Integrated Circuits” by Gordon E. Moore, Electronics, Vol. 38, No. 8, April 19, 1965.





Intel Historical Data

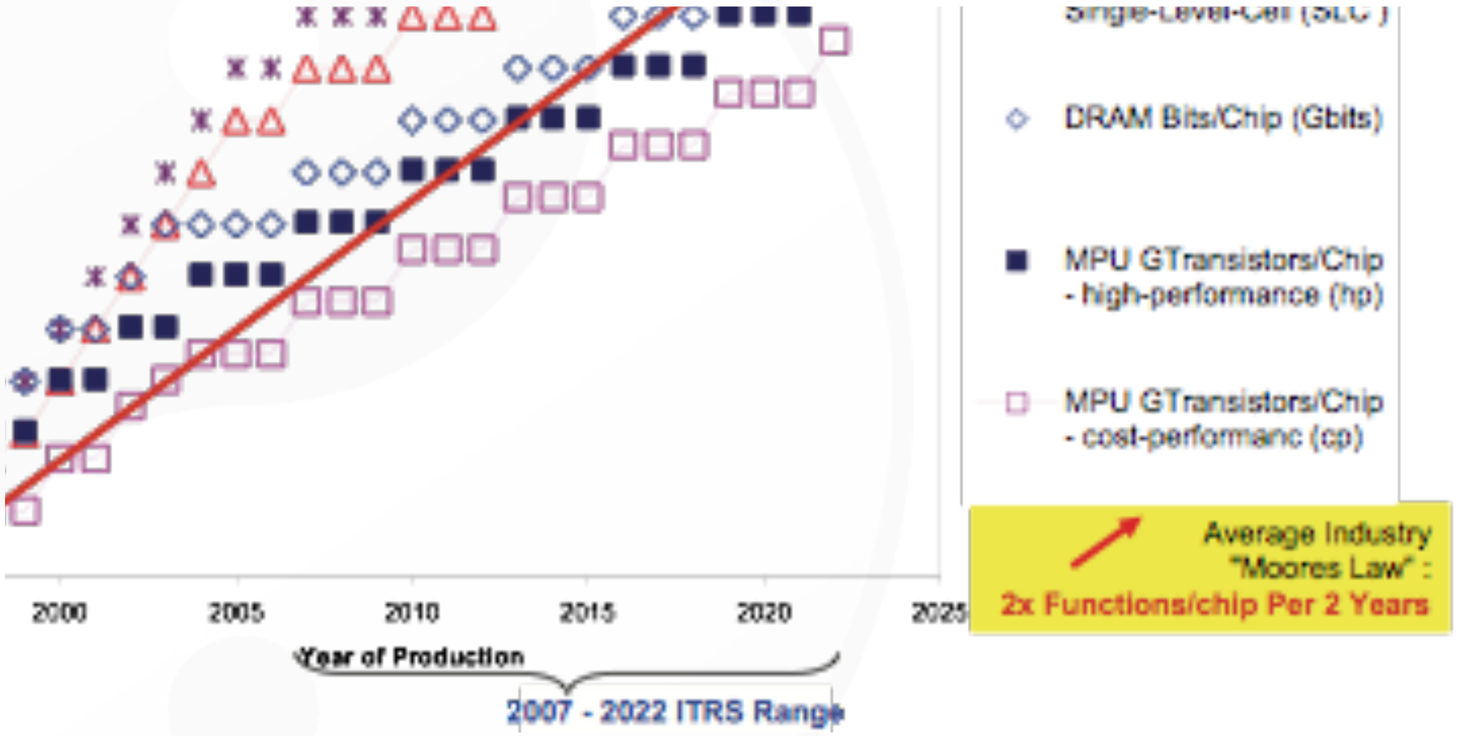


Semiconductor Scaling Rates

- From Digital Systems Engineering, Dally and Poulton, 1998

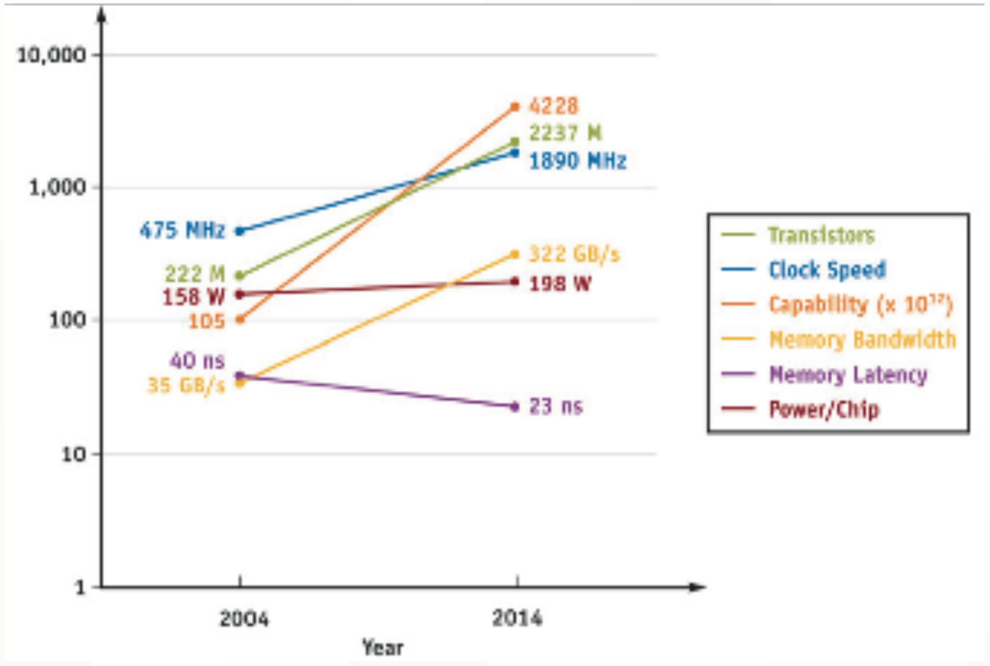
Parameter	Current Value	Yearly Factor	Years to Double (Half)
Moore's Law (grids on a die)**	1 B	1.49	1.75
Gate Delay	150 ps	0.87	(5)
Capability (grids / gate delay)		1.71	1.3
Device-length wire delay		1.00	
Die-length wire delay / gate delay		1.71	1.3
Pins per package	750	1.11	7
Aggregate off-chip bandwidth		1.28	3

** Ignores multi-layer metal, 8-layers in 2001



International Technology Roadmap for Semiconductors

10-Year GPU Projection (2004–14)



Source: Owens, "Streaming Architectures and Technology Trends," GPU Gems 2, March 2005.

- Transistors (NV40): 222M/2237M
- Clock speed (NV40, MHz): 475/1890
  - Capability: 105B/4228B
- Memory bandwidth (NV40, GB/s): 35/322
- Memory latency (RAS, ns): 40/23
- Power/chip (maximum, W): 158/198 (then flat?)
- Take-home point: Capability > mem bw > mem latency

Technology Theme

In a power-constrained world, we cannot keep increasing performance by increasing clock speed.

Process Generations

- Let’s assume one process generation to the next makes new transistors 0.7 times the size of the old transistors (“linear shrink”)
  - Recent example: 65 nm to 45 nm = 0.692
  - What is this “feature size”?
  - How does this relate to Moore’s Law?

Historical Scaling (0.7x)

- Power = C_{eff} [device] × # devices × voltage^2 × freq
  - Not talking about wires, leakage, etc.

- With process shrink, assuming constant die size:
  - Change in C_{eff}:
  - Change in number of devices:
  - Change in frequency:
  - Plug that all together:

Historical Scaling (0.7x)

- Power = C_{eff} [device] × # devices × voltage^2 × freq
  - Not talking about wires, leakage, etc.

- With process shrink, assuming constant die size:
  - Change in C_{eff}: 0.7x
  - Change in number of devices: 1/(0.7 × 0.7) = 2x
  - Change in frequency: 1/0.7x = 1.4x
  - Plug that all together: 2x

Historical Scaling (0.7x)

- Power = C_{eff} [device] × # devices × voltage^2 × freq
  - Not talking about wires, leakage, etc.

- With process shrink:
  - Change in C_{eff}:
  - Change in number of devices:
  - Change in frequency:
  - What else do we need to change to get constant power?
  - Plug that all together:

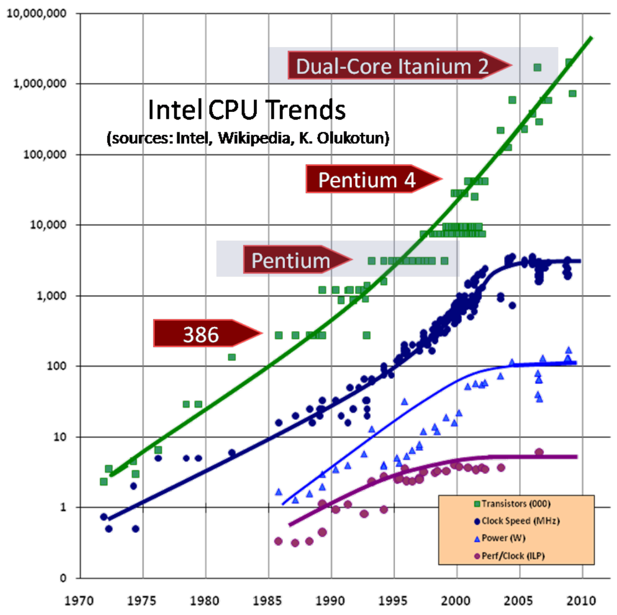
Historical Scaling (0.7x)

- Power = C_{eff} [device] × # devices × voltage^2 × freq
  - Not talking about wires, leakage, etc.

- With process shrink:
  - Change in C_{eff}: 0.7x
  - Change in number of devices: 1/(0.7 × 0.7) = 2x
  - Change in frequency: 1/0.7x = 1.4x
  - Change in voltage squared: (0.7 × 0.7)x = 0.5x
  - Plug that all together: 1x (constant power draw)

Historical Scaling Result

- Constant power from generation to generation
- What is our increase of (potential) performance from generation to generation?
  - 2x devices × 1.4x frequency = 2.8x performance at same power

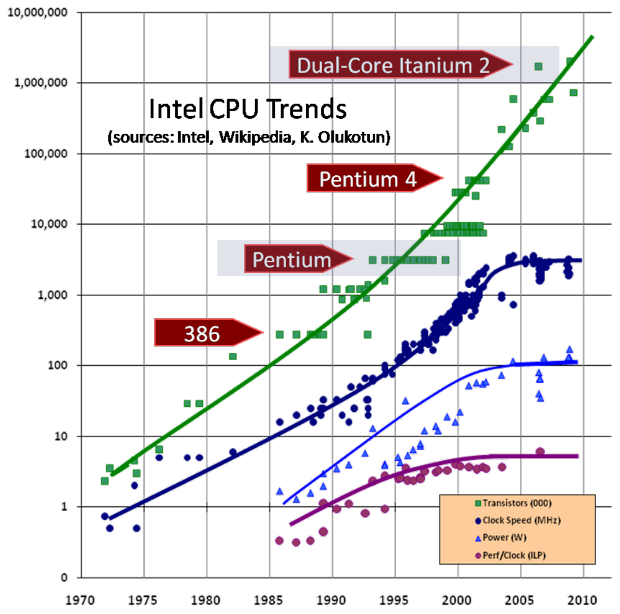


Dennard's Law:
As the dimensions of a device go down, so does its power consumption.

Scaling by the Mid-2000s?

- For 0.7x transistor scaling:
  - C_{eff} continues to scale (0.7x)
  - Device density continues to scale (2x)
  - Voltage does not scale as much (0.95x)
  - Frequency increases only by 1.2x
- Result: 2.4x performance, 1.5x power

Dennard's Law:
As the dimensions of a device go down,
so does its power consumption.



Outline

- Benchmarks
- Technology
  - In a power-constrained world, we can’t keep increasing performance by increasing clock speed.
- Architectural Trends
  - Microarchitectures have historically increased performance by increasing ILP and pipeline depth. Today, we can’t keep increasing performance at historical levels by these methods.
- Bandwidth vs. Latency

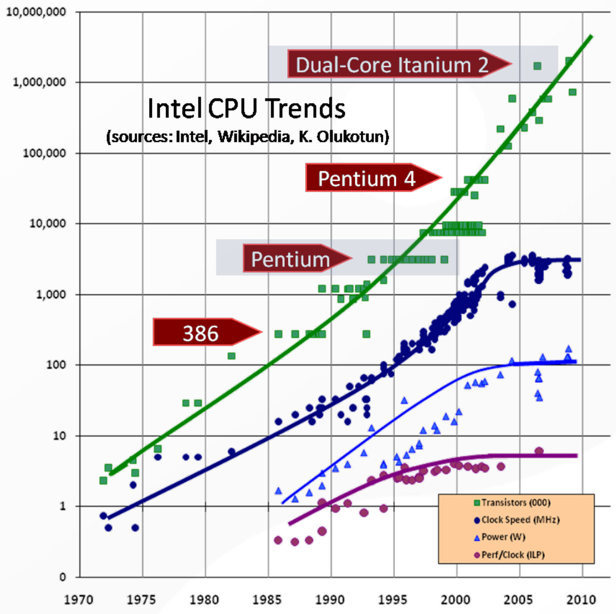
Architectural Trends

- Microarchitectures have historically increased performance by increasing ILP and pipeline depth.
- We have *not* been able to continue increasing performance at these historical levels by these methods.

Why Do Processors Get Faster?

- Define "faster" as "more instructions per second"
- Neglecting software improvements …
- 3 reasons:
  - More parallelism (or more work per pipeline stage)
    - Fewer clocks/instruction
  - Deeper pipelines
    - Fewer gates/clock
  - Transistors get faster (Moore's Law)
    - Fewer ps/gate

Limits to Instruction-Level Parallelism



	branch predict	*jump predict*	*reg renaming*	*alias analysis*
Stupid	none	none	none	none
Fair	infinite	infinite	256	inspection
Good	infinite	infinite	256	perfect
Great	infinite	infinite	perfect	perfect
Perfect	perfect	perfect	perfect	perfect

Figure 4. Five increasingly ambitious models.

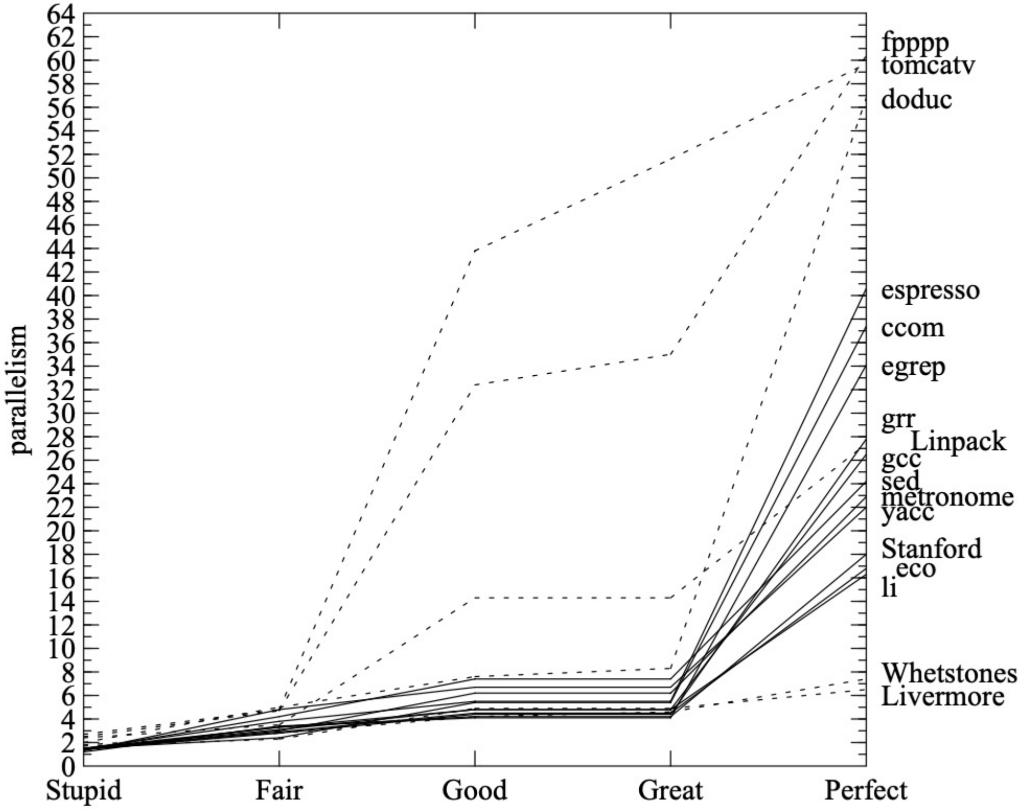
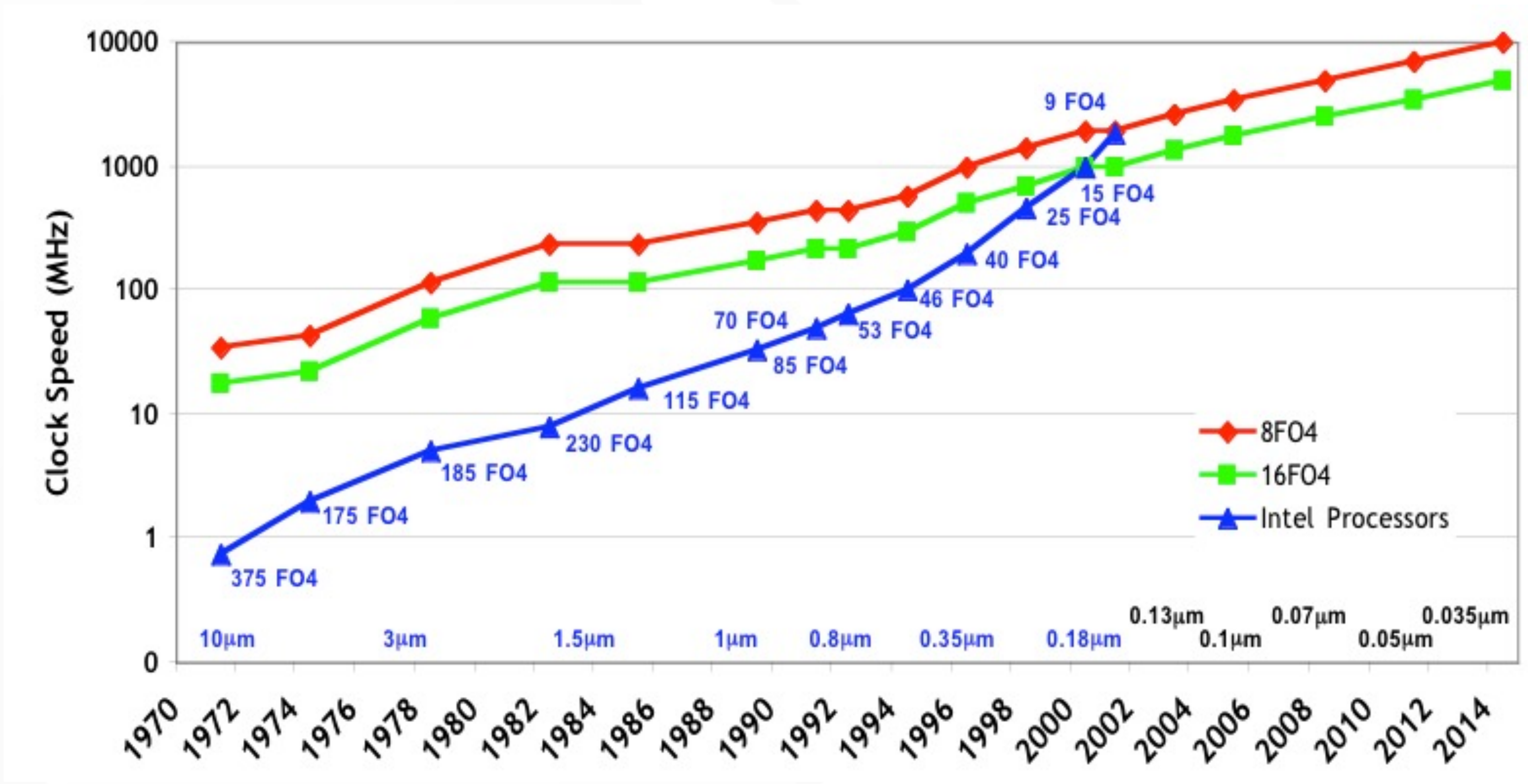


Figure 6. Parallelism under the five models.

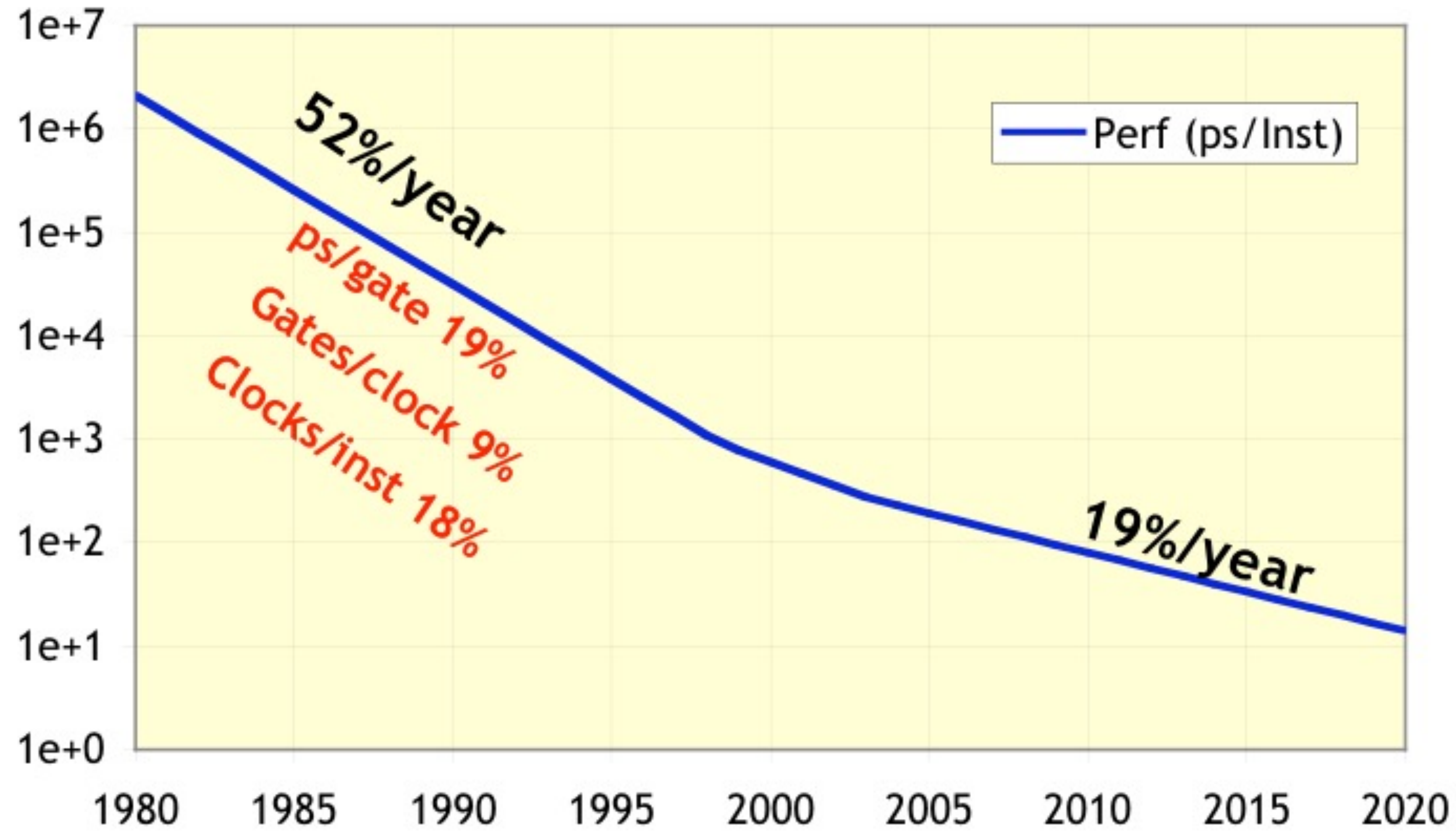
[David Wall, *Limits of Instruction-Level Parallelism*, WRL Research Report 93/6]

Clock Scaling: Historical & Projected



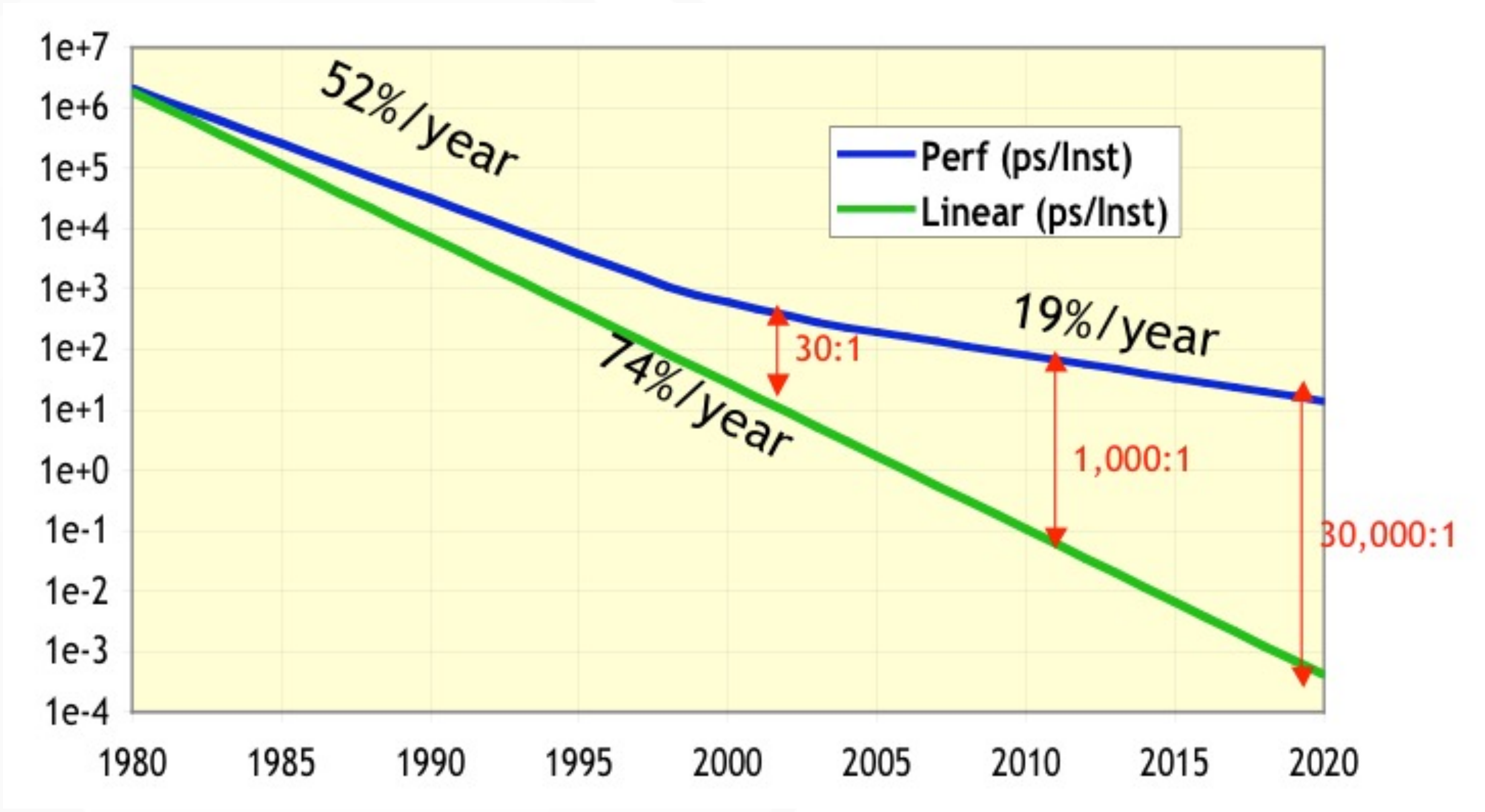
- "The Optimal Logic Depth Per Pipeline Stage is 6 to 8 FO4 Inverter Delays" (ISCA02) [courtesy Steve Keckler]

Microprocessor Scaling is Slowing



[courtesy of Bill Dally]

Future Potential is Large



- At the right-hand turn: 30:1
- 5 years: 1000:1

[courtesy of Bill Dally]

Summary

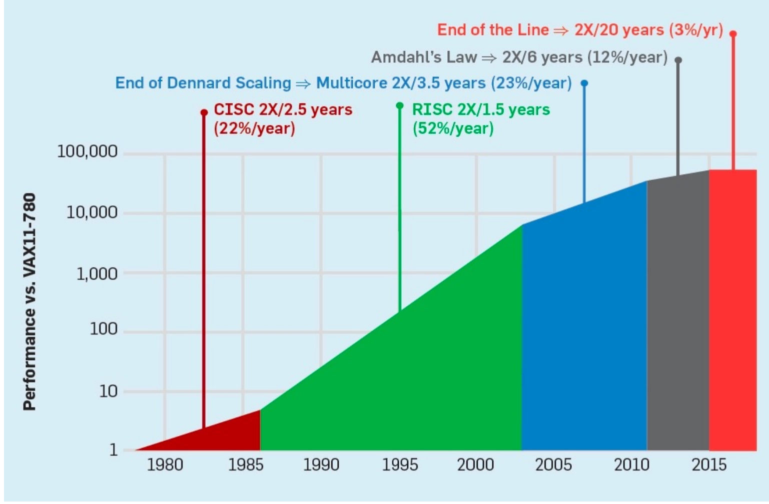
- Microprocessors have historically increased performance by
  - Increasing clock speed
  - Increasing pipeline depth
  - Increasing instruction-level parallelism (work per clock)
- We *CANNOT* continue improving performance by doing any of these things anymore.

Outline

- Benchmarks
- Technology
  - In a power-constrained world, we can’t keep increasing performance by increasing clock speed.
- Architectural Trends
  - Microarchitectures have historically increased performance by increasing ILP and pipeline depth. Today, we can’t keep increasing performance at historical levels by these methods.
- Bandwidth vs. Latency

Bandwidth vs. Latency

Technology Rates of …



- Processor
  - Logic Capacity: ~30% per year
  - Clock Rate: ~20% per year (2003-2011); ~10% per year (2011-2015); ~3% per year (2015-now)
- Memory
  - DRAM Capacity: ~60% per year (4x every 3 years)
  - Memory Speed: ~10% per year
  - Cost Per Bit: Improves ~25% per year
- Disk
  - Capacity: ~60% per year
  - Total Use of Data: ~100% per 9 months!
- Network
  - Bandwidth: 100%+ per year

Disk Trends

- MapReduce is a programming model for processing vast amounts of data. One of the reasons that it works so well is because it exploits a sweet spot of modern disk drive technology trends. In essence MapReduce works by repeatedly sorting and merging data that is streamed to and from disk at the transfer rate of the disk. Contrast this to accessing data from a relational database that operates at the seek rate of the disk (seeking is the process of moving the disk's head to a particular place on the disk to read or write data).
- So why is this interesting? Well, look at the trends in seek time and transfer rate. Seek time has grown at about 5% a year, whereas transfer rate at about 20%. Seek time is growing more slowly than transfer rate—so it pays to use a model that operates at the transfer rate, which is what MapReduce does.
- http://www.lexemetech.com/2008/03/disks-have-become-tapes.html

Disk Trends

- The general point summed up by Jim Gray in an interview in ACM Queue from 2003:

  ... programmers have to start thinking of the disk as a sequential device rather than a random access device.

- Or the more pithy …

  "Disks have become tapes." (Quoted by David DeWitt.)

- http://www.lexemetech.com/2008/03/disks-have-become-tapes.html

Tracking Technology: Performance Trends

- Drill down on four technologies
  - Disks; Memory; Network; Processors
  - ~1980 Archaic (Nostalgic) vs. ~2000 Modern (Newer)
    - Performance milestones in each technology
- Compare "Bandwidth vs. Latency" improvements in performance over time
- Define
  - Bandwidth: number of events per unit time
    - e.g., Mbits / second over network, MB / second from disk
  - Latency: elapsed time for a single event
    - e.g., one-way network delay in microseconds, average disk access time in milliseconds

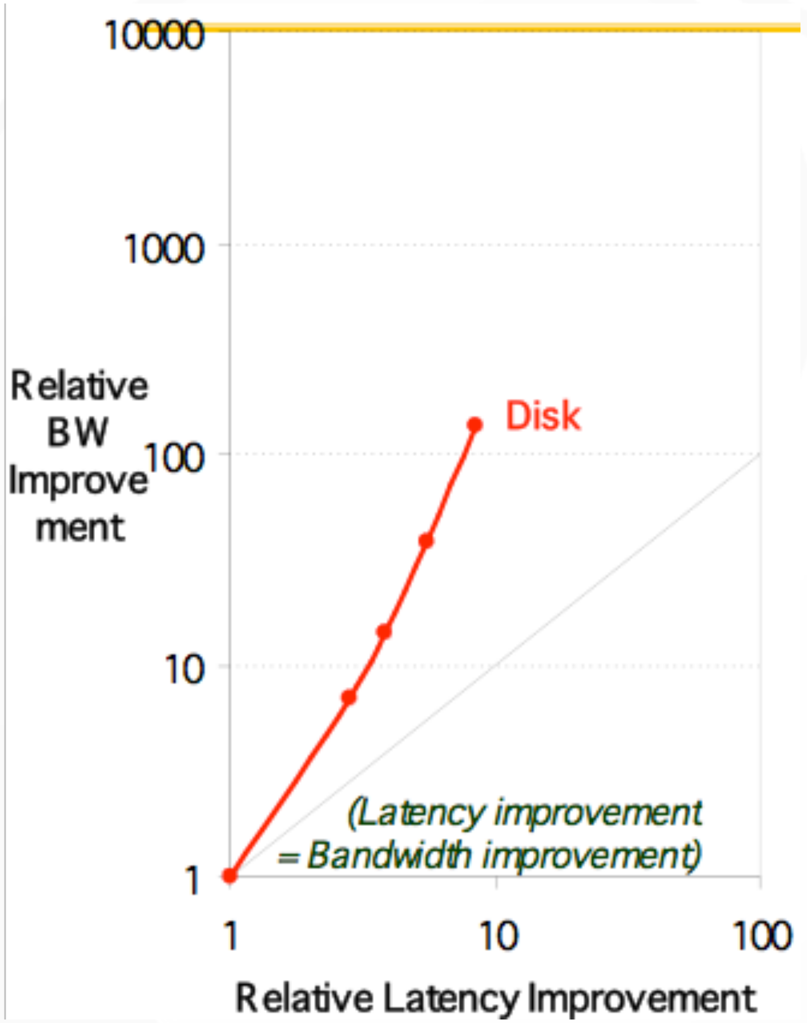
D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014

Disks

Archaic (Nostalgic) vs. Modern (Newer)

• CDC Wren I, 1983	• Seagate 373453, 2003	
• 3600 RPM	• 15000 RPM	(4X)
• 0.03 GBytes capacity	• 73.4 GBytes	(2500X)
• Tracks/Inch: 800	• Tracks/Inch: 64000	(80X)
• Bits/Inch: 9550	• Bits/Inch: 533,000	(60X)
• Three 5.25" platters	• Four 2.5" platters (in 3.5" form factor)	
• Bandwidth: 0.6 MBytes/sec	• BW: 86 MB/sec	(140X)
• Latency: 48.3 ms	• Latency: 5.7 ms	(8X)
• Cache: none	• Cache: 8 MB	

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014



Disk Latency vs. Bandwidth

- Disk: 3600, 5400, 7200, 10000, 15000 RPM (8x, 143x)
  - latency = simple operation w/o contention
  - bandwidth (BW) = best case

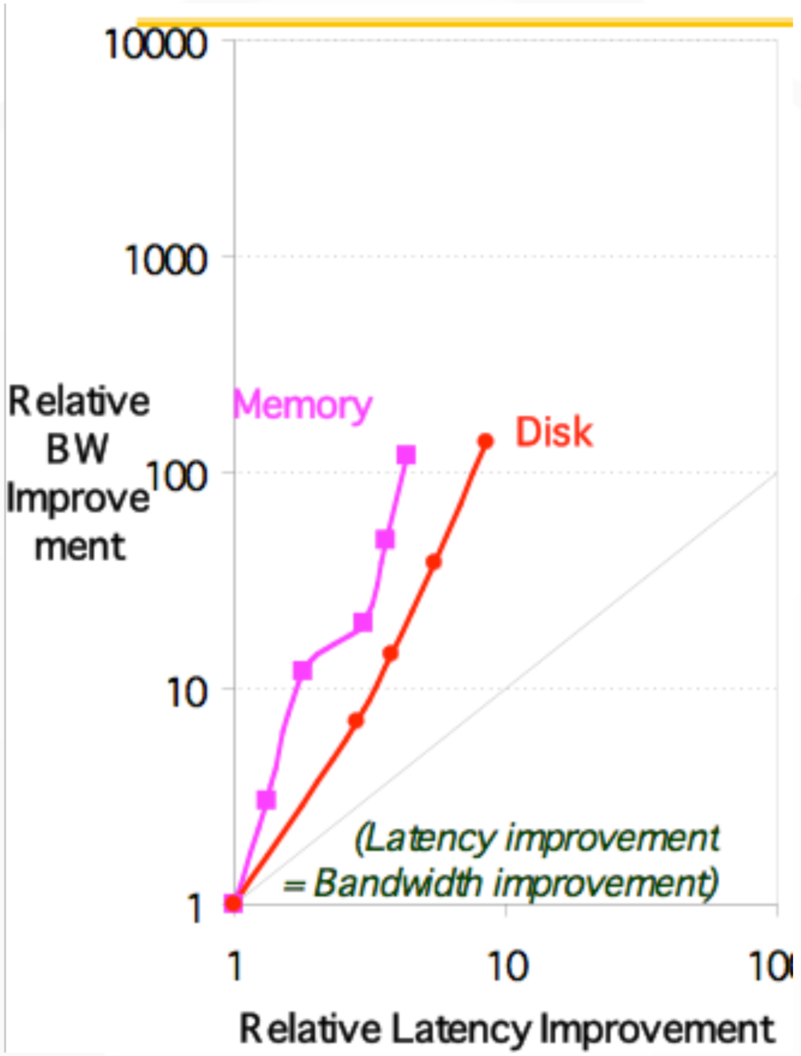
D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014

Memory:

Archaic (Nostalgic) vs. Modern (Newer)

• 1980 DRAM (asynchronous)	• 2000 Double Data Rate Synchr. (clocked) DRAM	
• 0.06 Mbits/chip	• 256 Mb/chip	(4000X)
• 64,000 xtors, 35 mm^2	• 256M xtors, 204 mm^2	
• 16b data bus/module, 16 pins/chip	• 64b data bus / DIMM, 66 pins/chip	(4X)
• 13 Mbytes/sec	• 1600 MB/sec	(120X)
• Latency: 225 ns	• Latency: 52 ns	(4X)
• (No block transfer)	• Block transfers (page mode)	

D. Patterson, “Latency Lags Bandwidth,” *Communications of the ACM*, October 2014



Latency Lags Bandwidth
(last ~20 years)

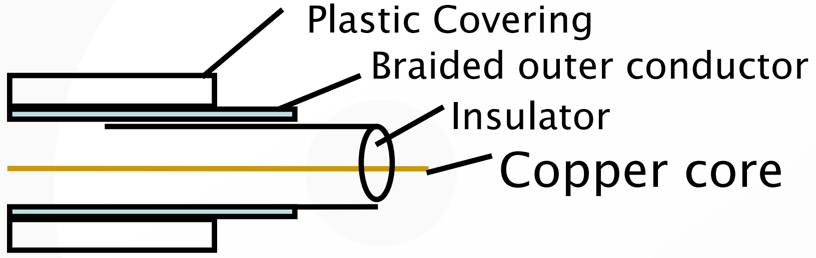
- Memory Module: 16-bit plain DRAM, Page Mode DRAM, 32b, 64b, SDRAM, DDR SDRAM (4x,120x)
  - latency = simple operation w/o contention
  - bandwidth (BW) = best case

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014

LANs

Archaic (Nostalgic) vs. Modern (Newer)

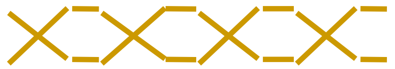
- Ethernet 802.3 (1978)
- 10 Mbits/s link speed
- Latency: 3000 µsec
- Shared media
- Coaxial cable



- Ethernet 802.3ae (2003)
- 10,000 Mbits/s (1000X) link speed
- Latency: 190 µsec (15X)
- Switched media
- Category 5 copper wire

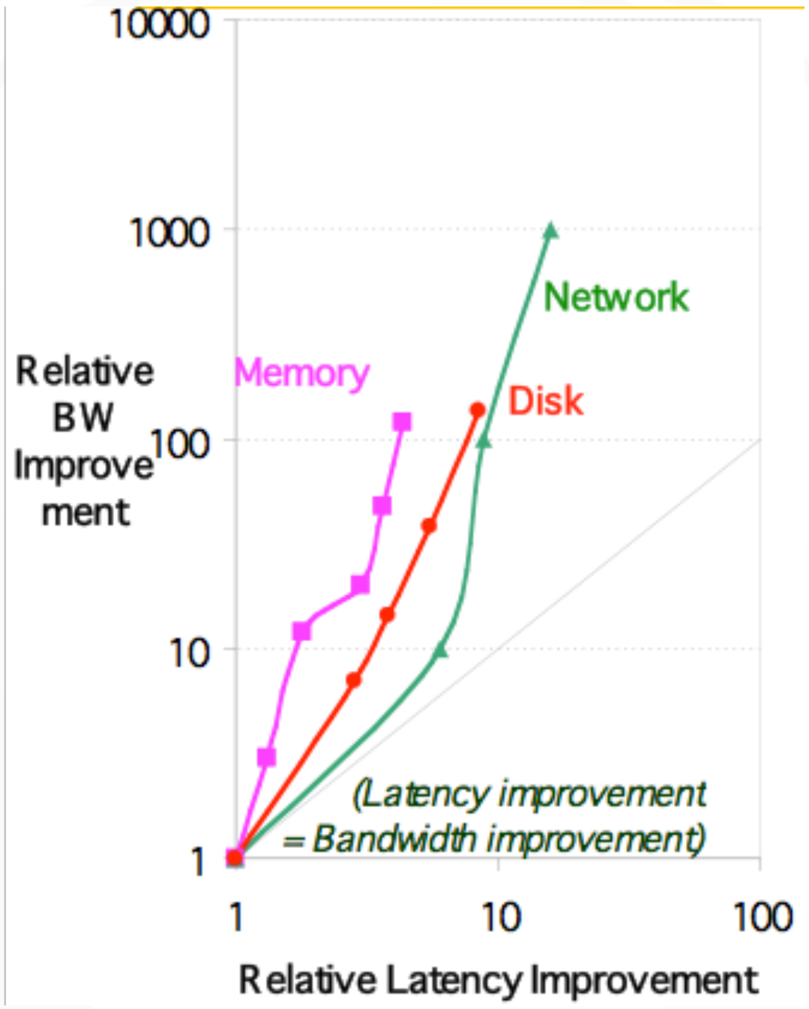
"Cat 5" is four twisted pairs in a bundle

Twisted Pair:



Copper, 1mm thick,
twisted to avoid antenna effect

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014



Latency Lags Bandwidth (last ~20 years)

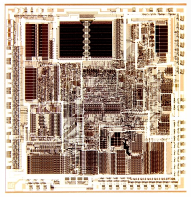
- Ethernet: 10Mb, 100Mb, 1000Mb, 10000 Mb/s (16x,1000x)
  - latency = simple operation w/o contention
  - bandwidth (BW) = best case

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014.

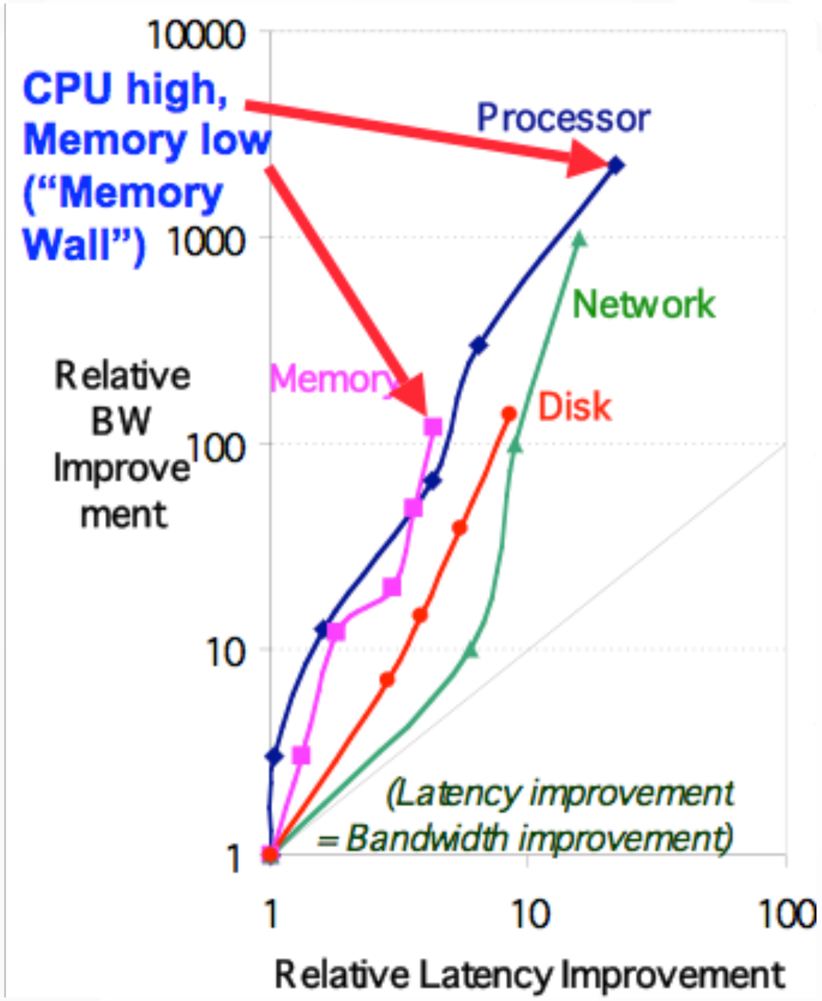
CPUs

Archaic (Nostalgic) v. Modern (Newer)

- 1982 Intel 80286
- 12.5 MHz
- 2 MIPS (peak)
- Latency 320 ns
- 134,000 xtors, 47 mm^2
- 16-bit data bus, 68 pins
- Microcode interpreter, separate FPU chip
- (no caches)



- 2001 Intel Pentium 4
- 1500 MHz (120X)
- 4500 MIPS (peak) (2250X)
- Latency 15 ns (20X)
- 42,000,000 xtors, 217 mm^2
- 64-bit data bus, 423 pins
- 3-way superscalar, dynamic translate to RISC, superpipelined (22 stages), out-of-order execution
- On-chip 8 KB data caches, 96 KB instr. trace cache, 256 KB L2 cache



Latency Lags Bandwidth
(last ~20 years)

- Processor: '286, '386, '486, Pentium, Pentium Pro, Pentium 4 (21x,2250x)

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014

Latency Lags Bandwidth

RULE OF THUMB

- In the time that bandwidth doubles, latency improves by no more than a factor of 1.2 to 1.4

  … and capacity improves faster than bandwidth

- Stated alternatively
  - Bandwidth improves by more than the square of the improvement in latency

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014

6 Reasons Latency Lags BW

- Moore’s Law helps BW more than latency
  - Faster transistors, more transistors, more pins help bandwidth

• MPU Transistors:	0.130 vs. 42 M xtors	(300X)
• DRAM Transistors:	0.064 vs. 256 M xtors	(4000X)
• MPU Pins:	68 vs. 423 pins	(6X)
• DRAM Pins:	16 vs. 66 pins	(4X)

D. Patterson, “Latency Lags Bandwidth,” *Communications of the ACM*, October 2014.

6 Reasons Latency Lags BW

- Moore's Law helps BW more than latency
  - Smaller, faster transistors but communicate over (relatively) longer lines: limits latency improvements
    - Feature size: 1.5 to 3 vs. 0.18 micron (8X,17X)
    - MPU Die Size: 35 vs. 204 mm^2 (ratio sqrt ⇒ 2X)
    - DRAM Die Size: 47 vs. 217 mm^2 (ratio sqrt ⇒ 2X)

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014.

6 Reasons Latency Lags BW

- Distance limits latency
  - Size of DRAM block ⇒ long bit and word lines ⇒ most of DRAM access time
  - Speed of light and computers on network
  - 1. & 2. explains linear latency vs. square BW?

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014.

6 Reasons Latency Lags BW

- Bandwidth easier to sell ("bigger = better")
  - Example
    - 10 Gbits/s Ethernet ("10 Gig") vs. 10 µsec latency Ethernet
  - 4400 MB/s DIMM ("PC4400") vs. 50 ns latency
  - Even if just marketing, customers now trained
  - Since bandwidth sells, more resources thrown at bandwidth, which further tips the balance

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014.

6 Reasons Latency Lags BW

- Latency helps BW, but not vice versa
  - Spinning disk faster improves both bandwidth and rotational latency
    - 3600 RPM ⇒ 15000 RPM = 4.2X
    - Average rotational latency: 8.3 ms ⇒ 2.0 ms
    - Things being equal, also helps BW by 4.2X
  - Lower DRAM latency ⇒ More access/second (higher bandwidth)
  - Higher linear density helps disk BW (and capacity), but not disk latency
    - 9,550 BPI ⇒ 533,000 BPI ⇒ 60X in BW

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014.

6 Reasons Latency Lags BW

- Bandwidth hurts latency
  - Queues help bandwidth, hurt latency (Queuing Theory)
  - Adding chips to widen a memory module increases bandwidth but higher fan-out on address lines may increase latency
- Operating system overhead hurts latency more than bandwidth
  - Long messages amortize overhead; overhead bigger part of short messages

D. Patterson, "Latency Lags Bandwidth," *Communications of the ACM*, October 2014.

Summary of Technology Trends

- For disk, LAN, memory, and microprocessor, bandwidth improves by square of latency improvement
  - In the time that bandwidth doubles, latency improves by no more than 1.2X to 1.4X
- Lag probably even larger in real systems, as bandwidth gains multiplied by replicated components
  - Multiple processors in a cluster or even in a chip
  - Multiple disks in a disk array
  - Multiple memory modules in a large memory
  - Simultaneous communication in switched LAN
- HW and SW developers should innovate assuming "Latency Lags Bandwidth"
  - If everything improves at the same rate, then nothing really changes. When rates vary, require real innovation