Lecture 2
Benchmarks, Economics, and Technology

EEC 171 Parallel Architectures

John Owens
UC Davis

Credits

- © John Owens / UC Davis 2007–9.
- Thanks to many sources for slide material: Computer Organization and Design (Patterson & Hennessy) © 2005, Computer Architecture (Hennessy & Patterson) © 2007, Inside the Machine (Jon Stokes) © 2007, © Dan Connors / University of Colorado 2007, © Wen-Mei Hwu/David Kirk, University of Illinois 2007, © David Patterson / UCB 2003–6, © Mary Jane Irwin / Penn State 2005, © John Kubiatowicz / UCB 2002, © Krste Asinovic/Arvind / MIT 2002, © Morgan Kaufmann Publishers 1998.

Outline

- Benchmarks
- Economics
- 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

SPEC Benchmarks

SPEC2006 benchmark description	Benchmark name by SPEC generation				
	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	dealII				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			

3DMark06

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

TPC-C

- “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

- n warehouses
  - Each warehouse, 10 sales districts, terminal per s.d.
  - Each sales district, 3000 customers
- New order: 10 items, 10% chance from other w.house
- Other transactions: payment, order status, delivery, stock query
- TPC-C measures:
  - orders per second (tpmC)
  - Price-performance ($/tpmC)

Chip Economics

- Courtesy the Linley Group (Linley Gwennap) (*thanks!*)
- Scenario:
  - Building an ASIC in 0.18 µm technology (6? years old)
  - 50 mm² non-CPU, 5 mm² CPU, 55 mm² total per die
  - How much does it cost to build this chip?

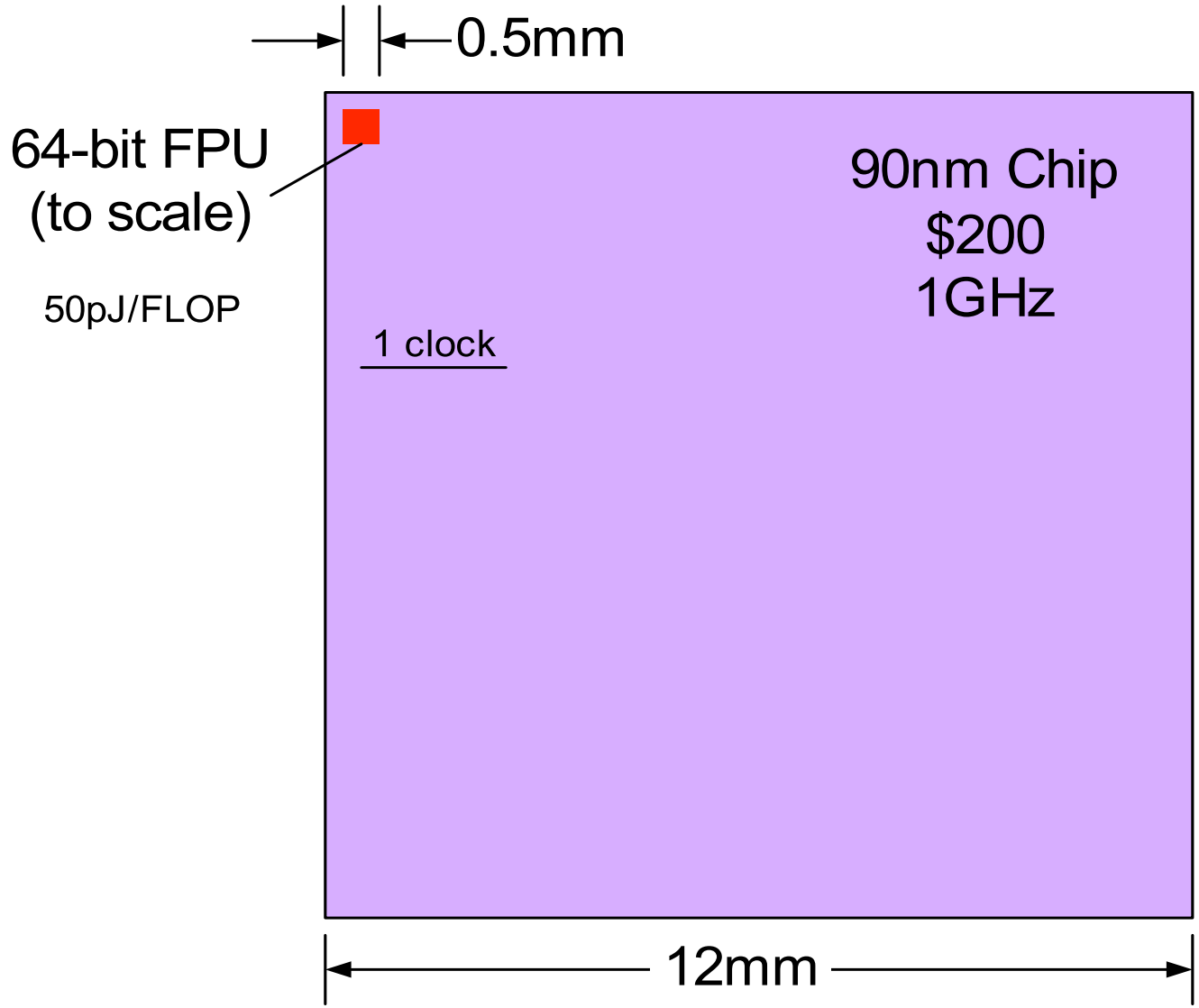
Wafer to Chips

- 200 mm^2 wafer
  - Costs $3400
- Gross dies/wafer: 523 dies can be cut per wafer
  - “Effective Area” fraction = 85%
- 0.5 defects per cm^2
- Yield: 80%
  - $1/(\text{defects} \times \text{die size} \times \text{effective area})^3$
- Net dies per wafer: 418
- Untested die cost: $8.14

Cost per Chip

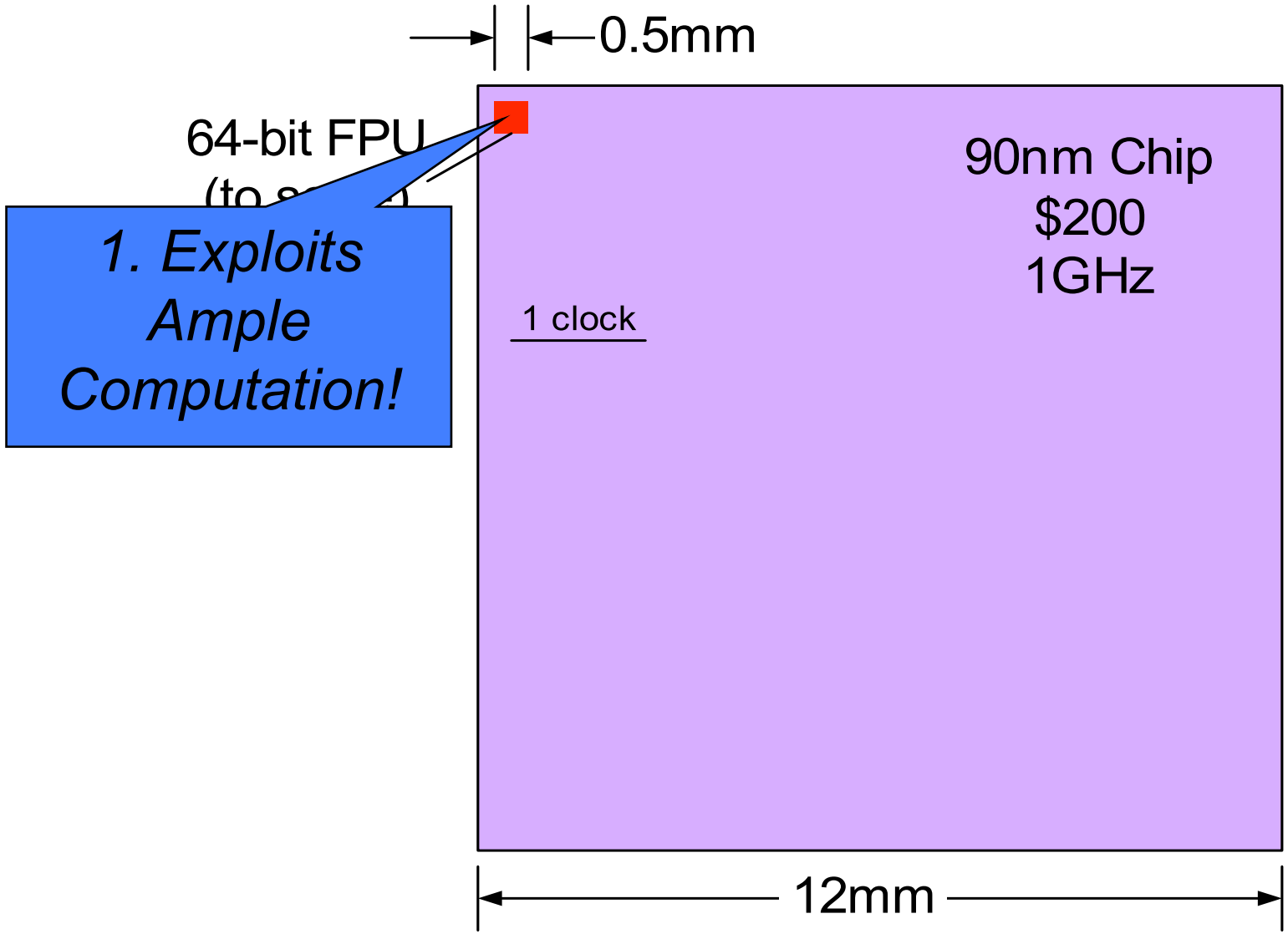
- $8.14 per untested die
- $1.00 test cost
- $0.50 packaging and assembly
- $2.00 package cost (BGA)
- 98% final test yield ($0.23 yield loss per chip)
- Total manufacturing cost : $11.87

Today's VLSI Capability

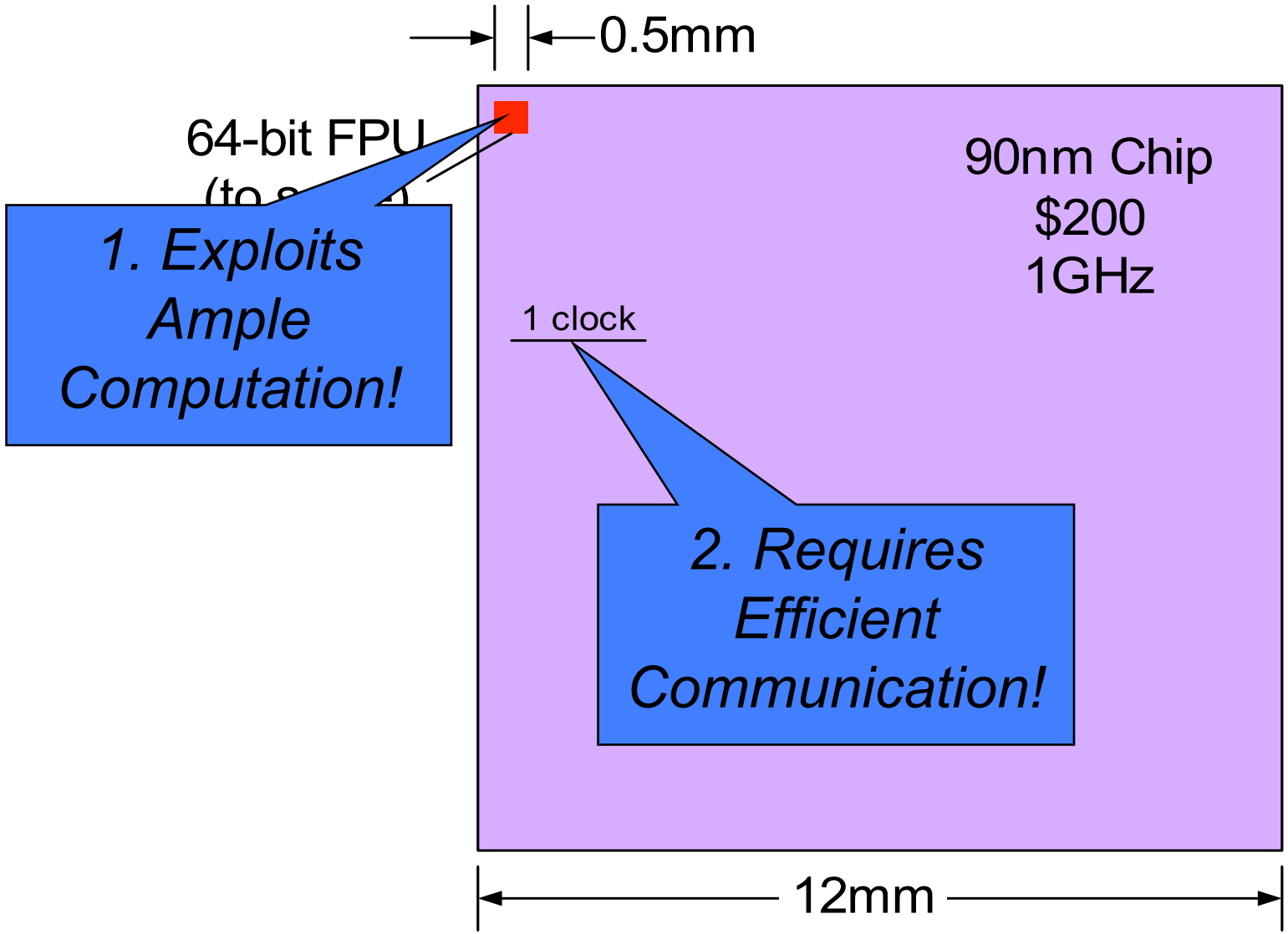


[courtesy Bill Dally]

Today's VLSI Capability



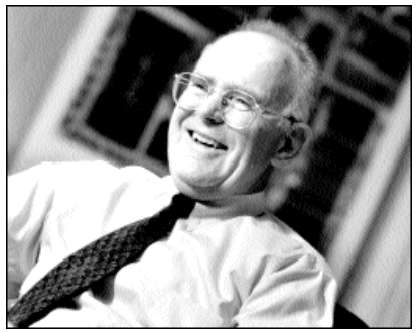
Today's VLSI Capability

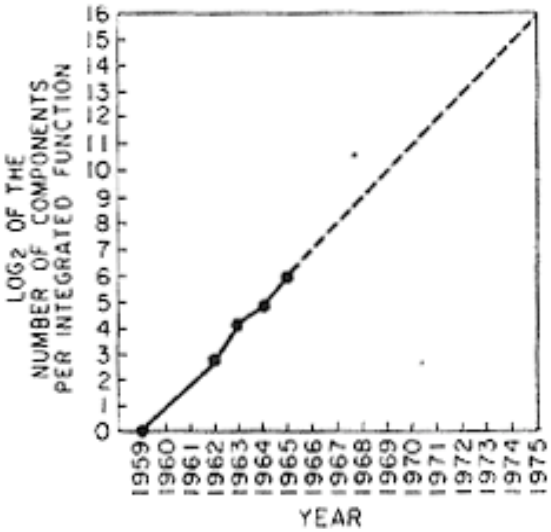


Moore's Law

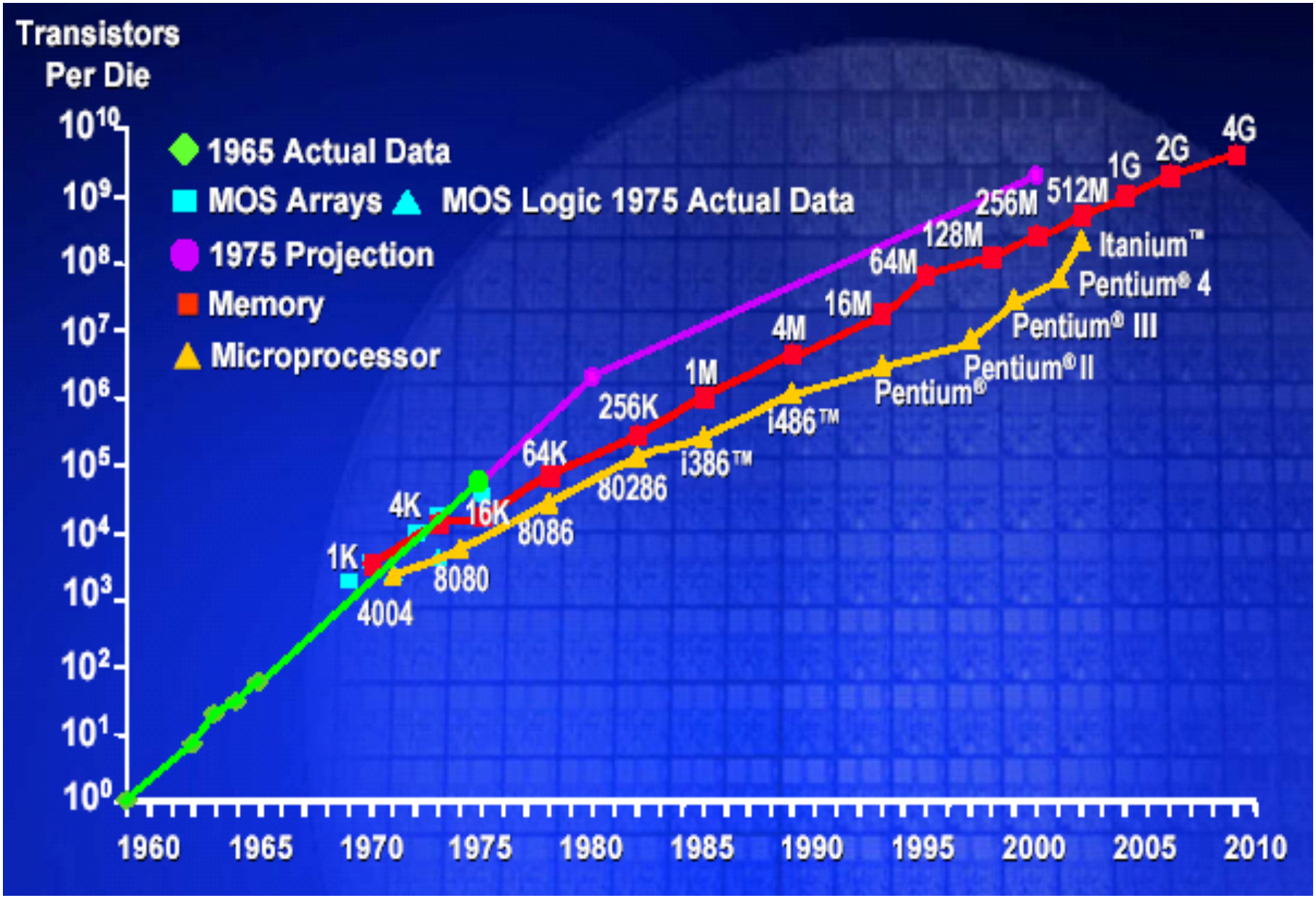
"The complexity for minimum component costs has increased at a rate of roughly a factor of two per year (see graph on next page). Certainly over the short term this rate can be expected to continue, if not to increase. Over the longer term, the rate of increase is a bit more uncertain, although there 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, Volume 38, Number 8, April 19, 1965





Intel historical data



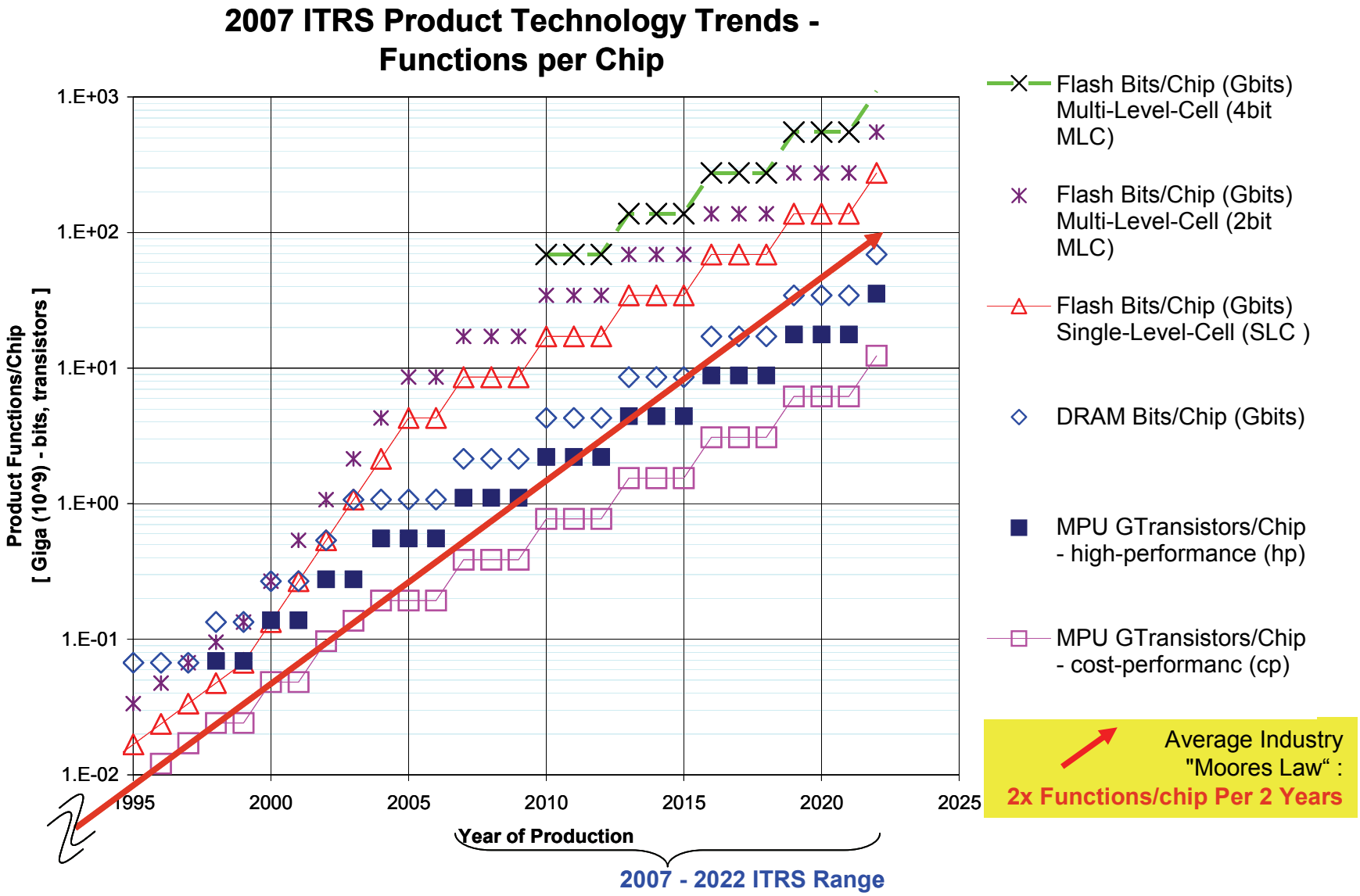
Semiconductor Scaling Rates

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

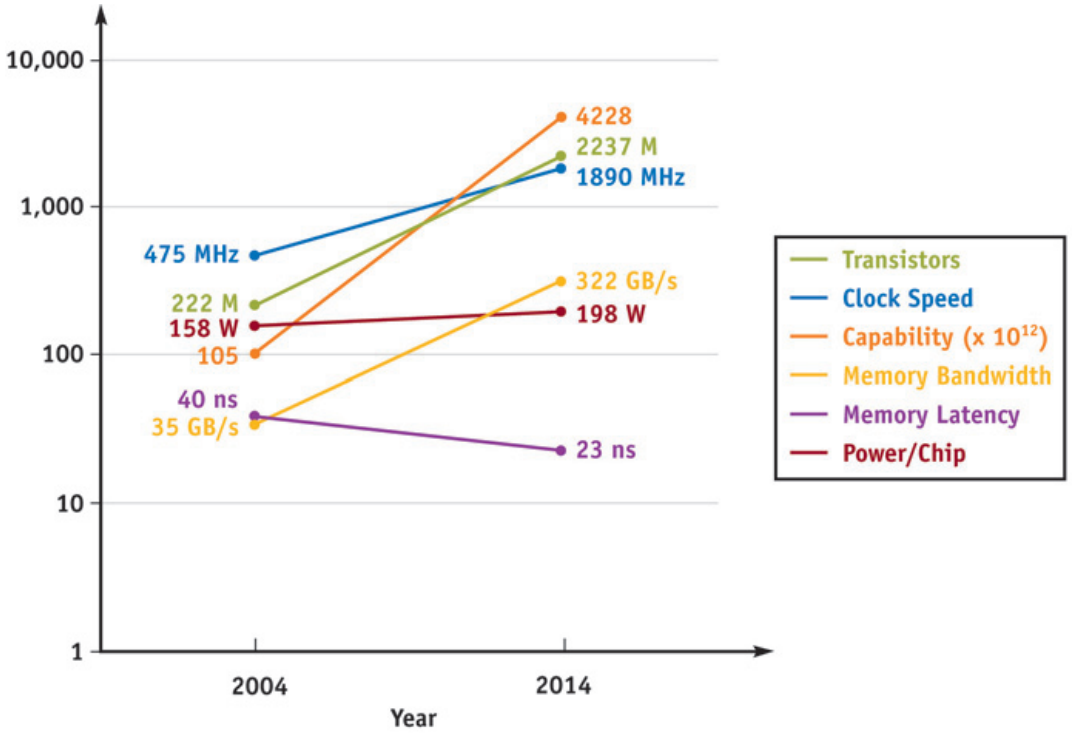
** Ignores multi-layer metal, 8-layers in 2001

Int'l Technology Roadmap for Semiconductors



Manufacturable solutions exist, and are being optimized
Manufacturable solutions are known
Interim solutions are known
Manufacturable solutions are NOT known

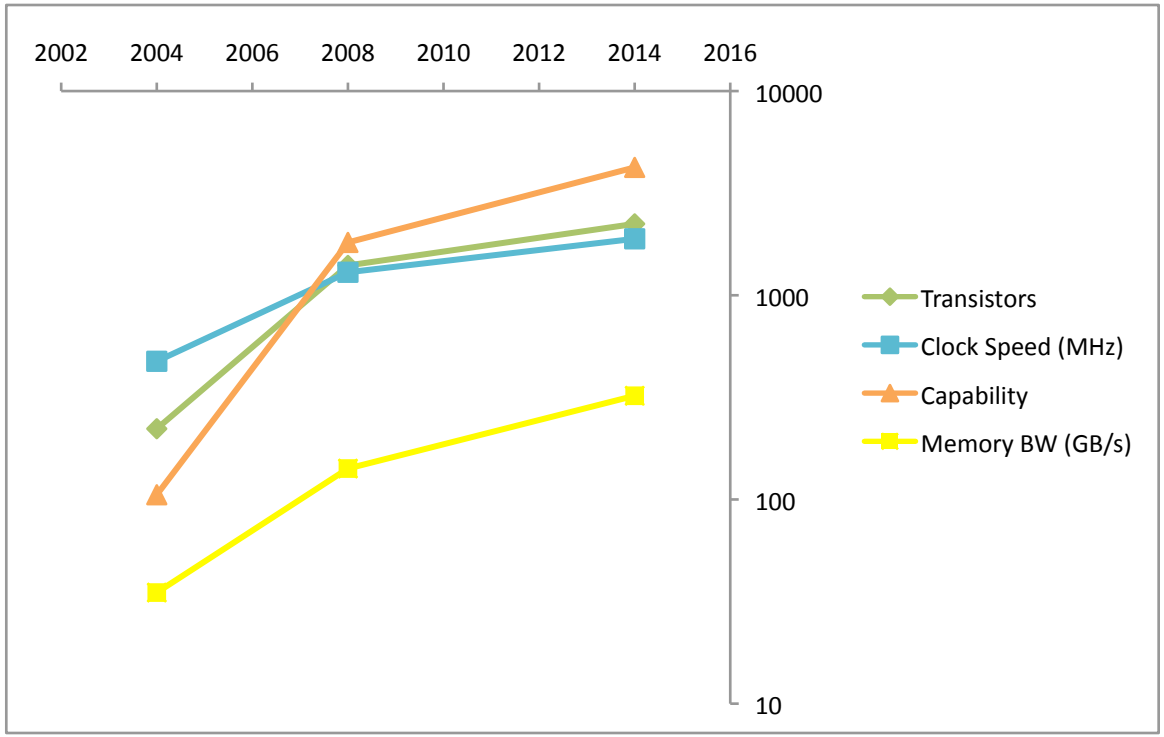
10-Year GPU Projection (2004–2014)

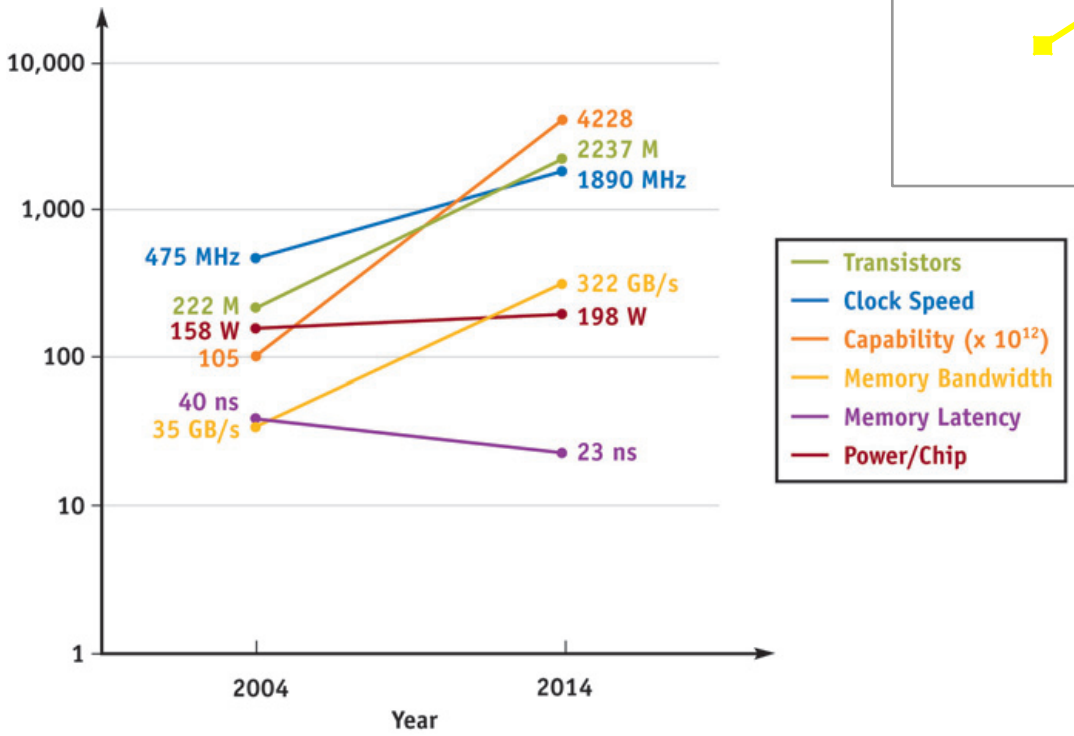


From Owens, *Streaming Architectures and Technology Trends*, GPU Gems 2, March 2005

- 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

2004, 2008 (NV GF280), 2014





Technology Theme

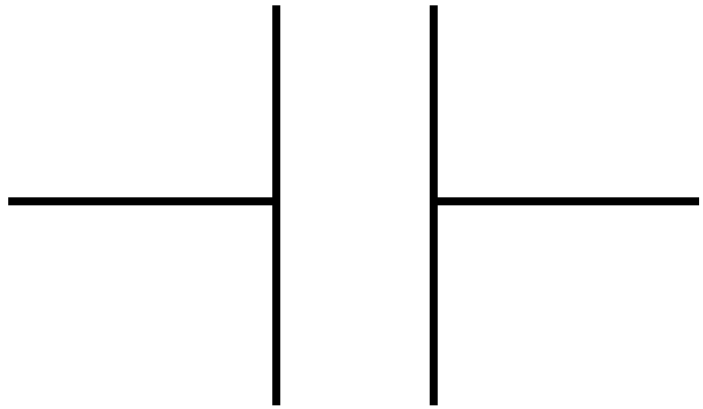
- *In a power-constrained world, we can't keep increasing performance by increasing clock speed.*

The Raisin Theory

- Observing the data—the laws of physics:
  - The silicon is not scaling in the same ways of the past:
    - Voltage scaling, the most significant contributor to power scaling, is decaying
    - We can no longer use the same architectural approaches for increased performance
      - At the same power:
        - The die gets smaller (a raisin)
        - The thermal density increases (a cooked raisin)

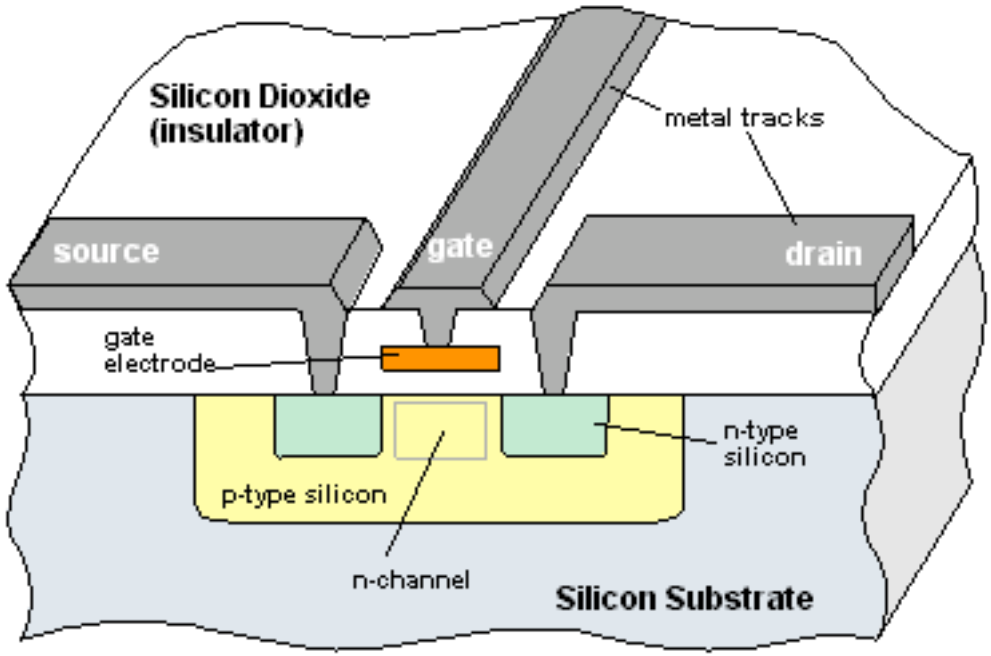
[courtesy Andy Riffel]

What's this?



- And how much energy does it take to charge it?

What's this?



From the Computer Desktop Encyclopedia

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”?

Historical scaling (0.7x)

- Power = Ceff [device] × # devices × Voltage² × MHz
  - Not talking about wires, leakage, etc.
- With process shrink, assuming constant die size:
  - Change in Ceff:
  - Change in number of devices:
  - Change in frequency:
  - Plug that all together:

Historical scaling (0.7x)

- Power = Ceff [device] × # devices × Voltage² × MHz
  - Not talking about wires, leakage, etc.
- With process shrink:
  - Change in Ceff:
  - 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 result

- Constant power from generation to generation
- What is our increase of (potential) performance from generation to generation?

Today's scaling

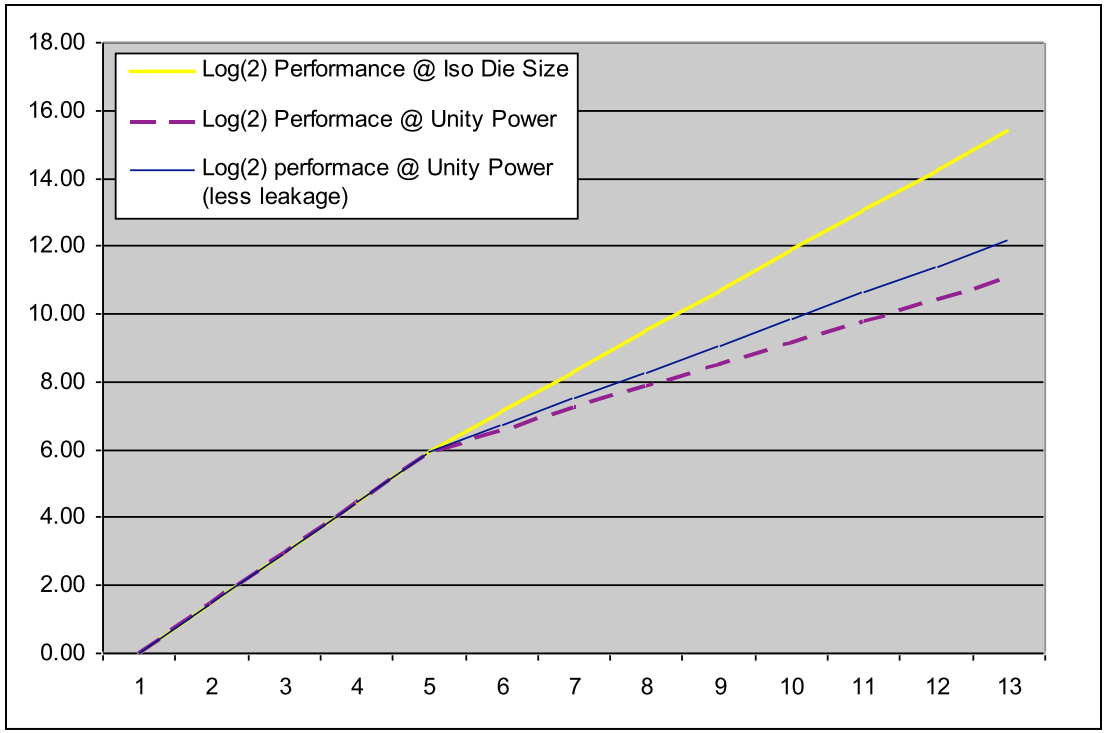
- Today, for 0.7x transistor scaling:
  - Ceff continues to scale (0.7x)
  - Device density continues to scale (2x)
  - Voltage does not scale as much (0.95x)
  - Frequency increases only by 1.2x
- Result: 2.3x performance, 1.46x power

How about wiring? (x=0.7)

- Capacitance: x
- Resistance of wire: 1/x
  - Length of wire: x
  - Cross-section of wire: $1/x^2$
- Resistance of constant-length wire: $1/x^2$ (+32%/year)
- Resistance of cross-chip wire: $1/x^3$ (+49%/year)
- RC/τ: delay of wire compared to device delay
  - Constant-length: +51%, cross-chip: +71%

Raisin Theory—The Real RHT

- Yellow line
  - 1st slope = 2.0X devices *
    1.40 MHz = 2.8x perf. scaling
  - 2nd slope = 1.9X devices *
    1.20 MHz = 2.3x perf. scaling
- Purple line
  - 1st slope = 2.0X devices *
    1.40 MHz = 2.8x perf. scaling
  - 2nd slope = 1.9X devices *
    0.82 MHz = 1.56x perf. scaling



The dashed purple line is performance gains at CONSTANT power

Why you are taking this class

2nd slope =

1.9X devices × 0.82 MHz =

1.56x perf. scaling

Technology Theme

- Microarchitectures have historically increased performance by increasing ILP and pipeline depth. Today, we can’t keep increasing performance at historical levels by these methods.

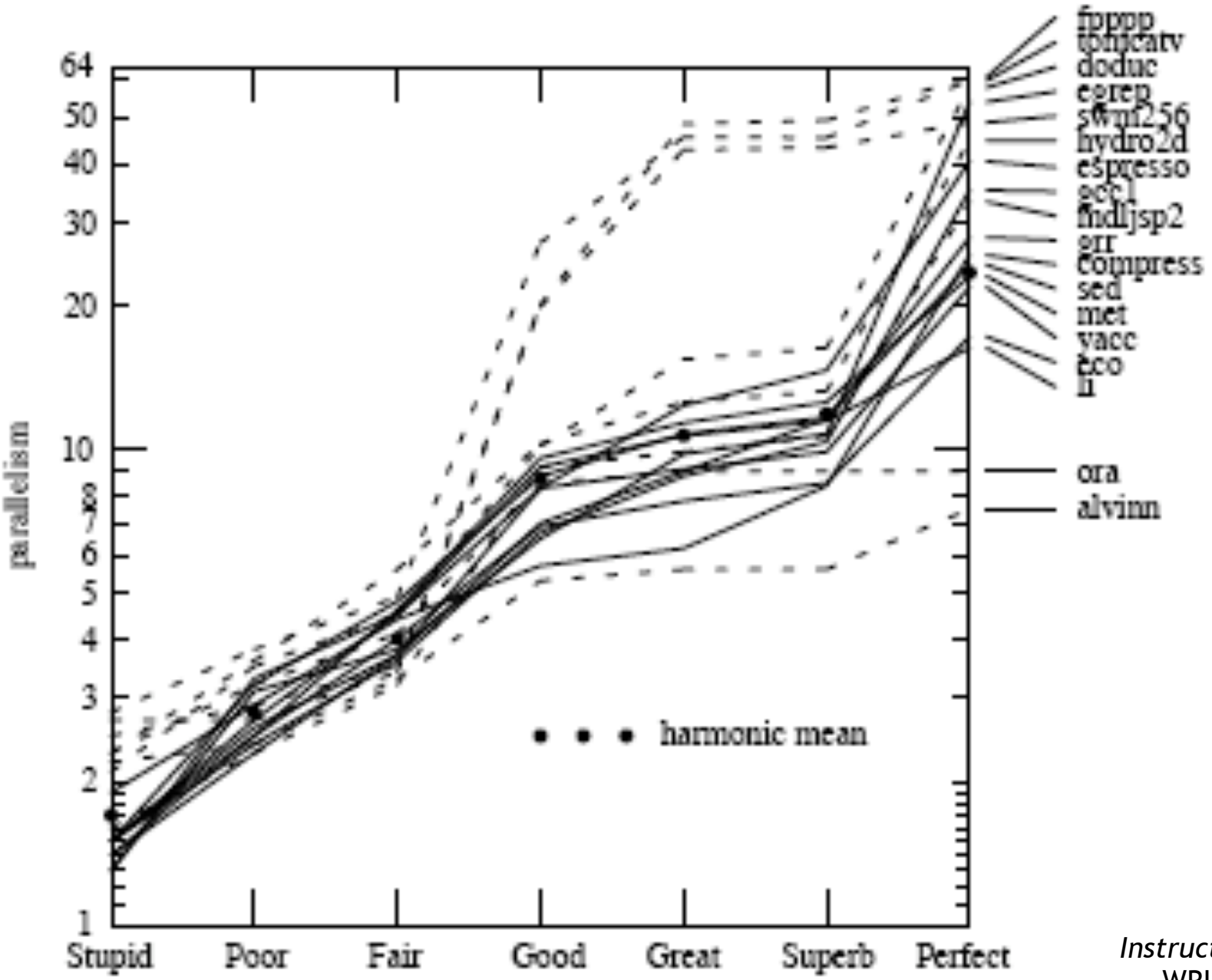
Why Do Processors Get Faster?

- Define “faster” as “more instructions per second”.
- Neglecting software improvements …
- 3 reasons:
  -
  -
  -

Microarchitectural Theme

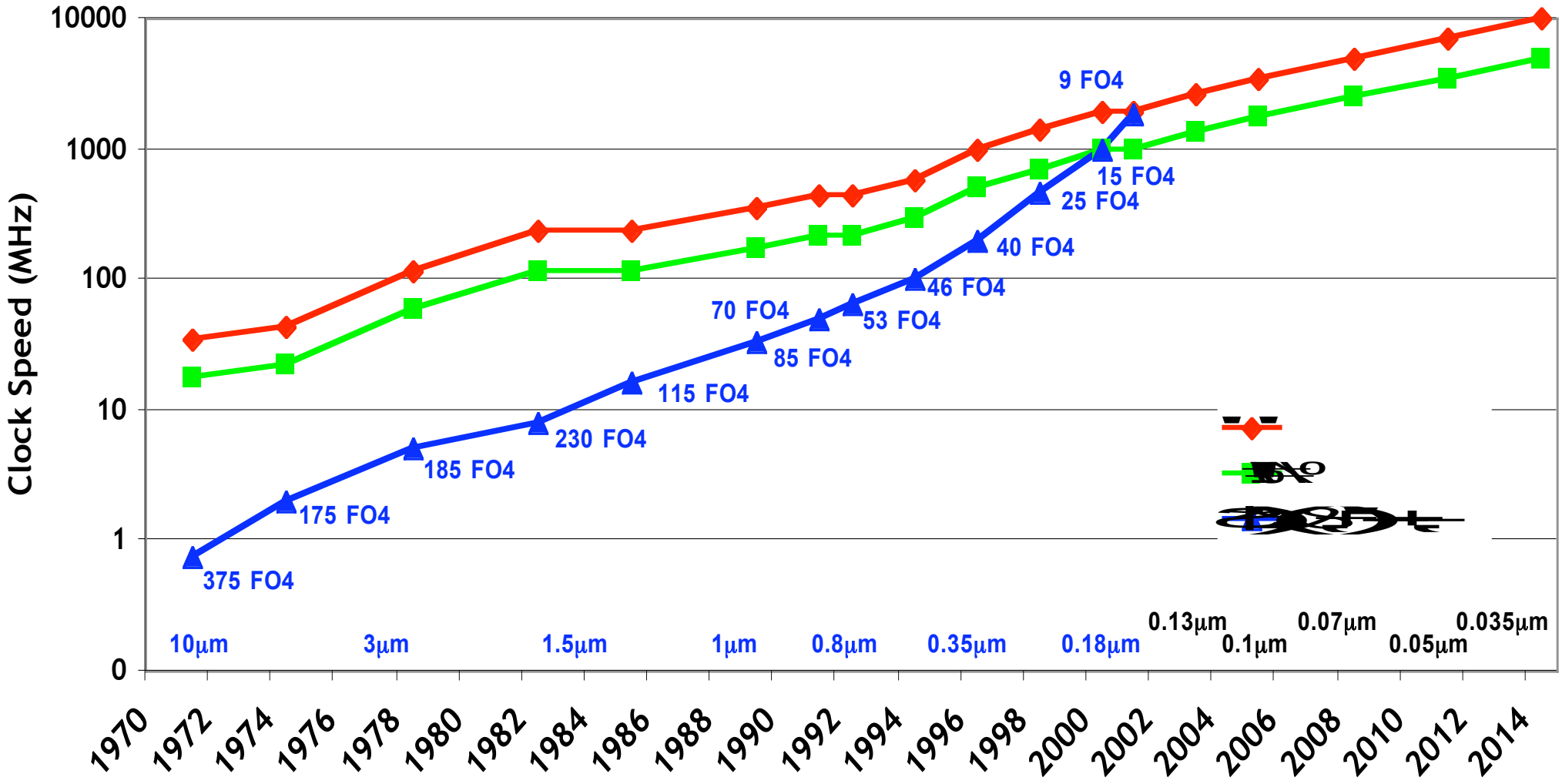
- Microarchitectures have historically increased performance by increasing ILP and pipeline depth. Today, we can’t keep increasing performance at historical levels by these methods.

Limits to Instruction Level Parallelism



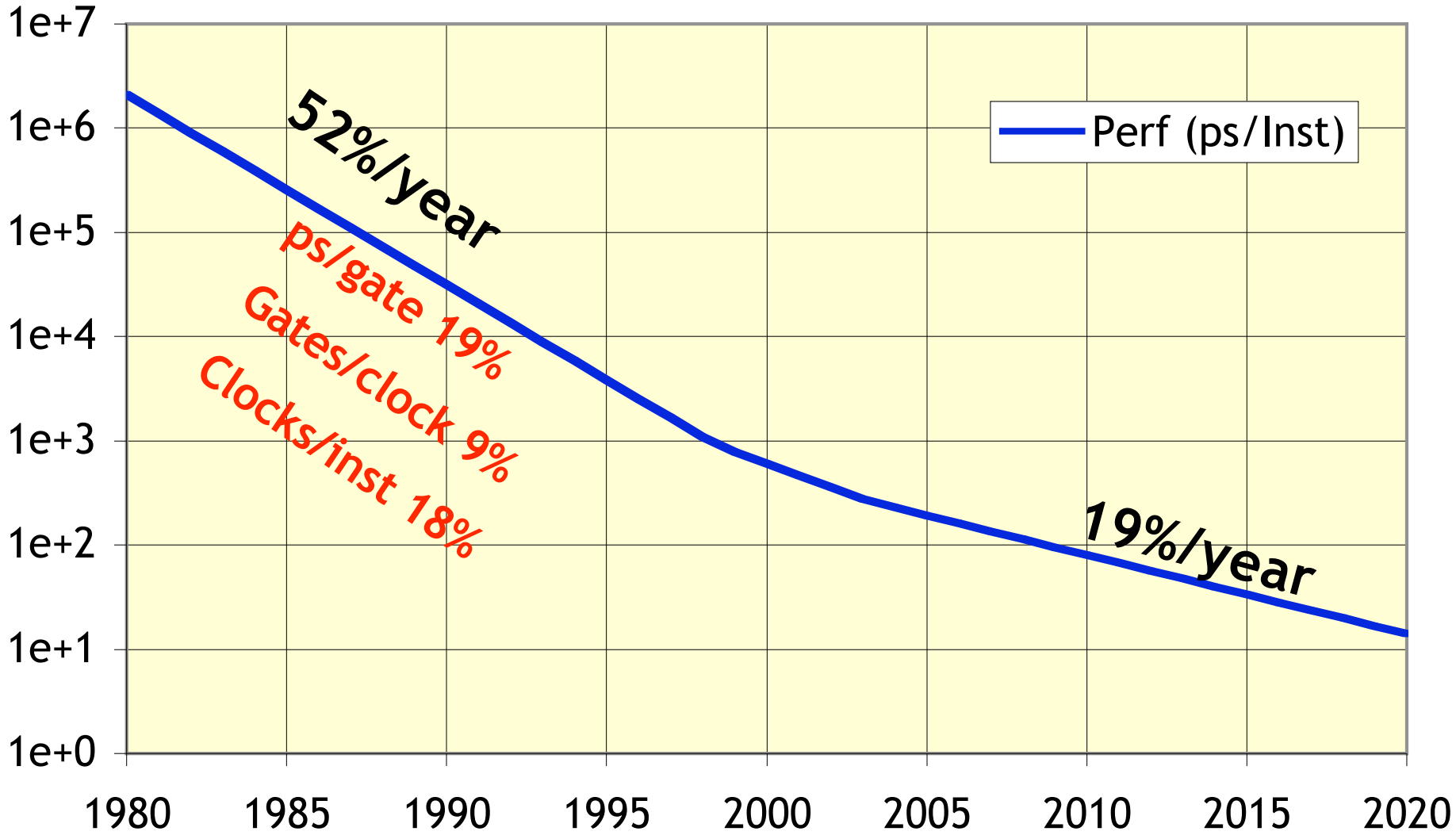
[David Wall, *Limits of Instruction-Level Parallelism*, WRL Research Report 93/6

Clock Scaling: Historical and Projected



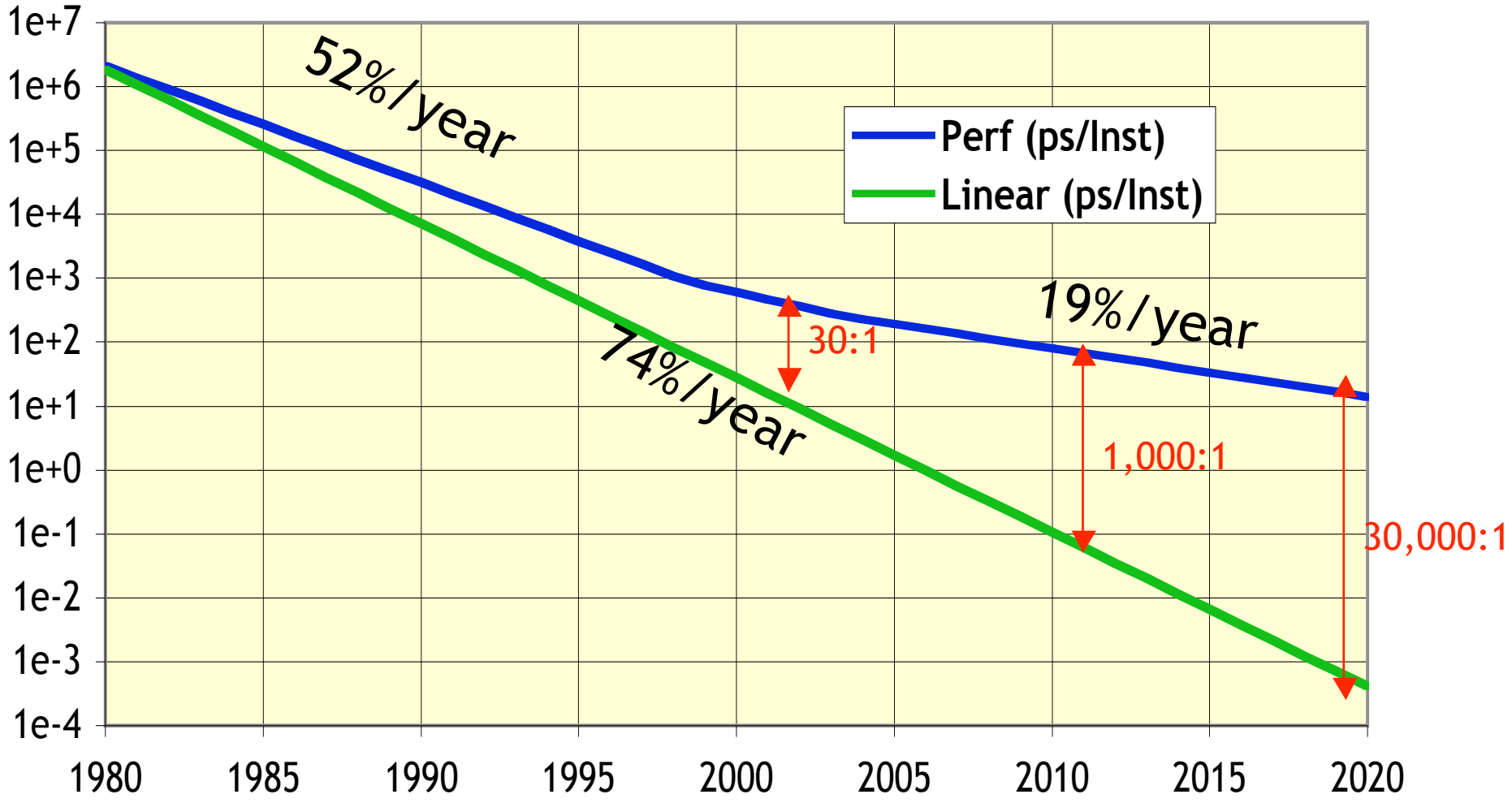
- "The Optimal Logic Depth Per Pipeline Stage is 6 to 8 FO4 Inverter Delays" (ISCA02) [courtesy Steve Keckler]

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 can’t do any of these things any more.

Technology rates of change

- Processor
  - logic capacity: about 30% per year
  - clock rate: about 20% per year
- Memory
  - DRAM capacity: about 60% per year (4x every 3 years)
  - Memory speed (latency): about 10% per year
  - Memory bandwidth: about 25% per year
  - Cost per bit: improves about 25% per year
- Disk
  - capacity: about 60% per year
  - Total use of data: 100% per 9 months!

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. I first saw this observation in Doug Cutting's talk, with Eric Baldeschwieler, at OSCON last year, where he worked through the numbers for updating a 1 terabyte database using the two paradigms B-Tree (seek-limited) and Sort/Merge (transfer-limited).

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

Disk Trends

The general point was well 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

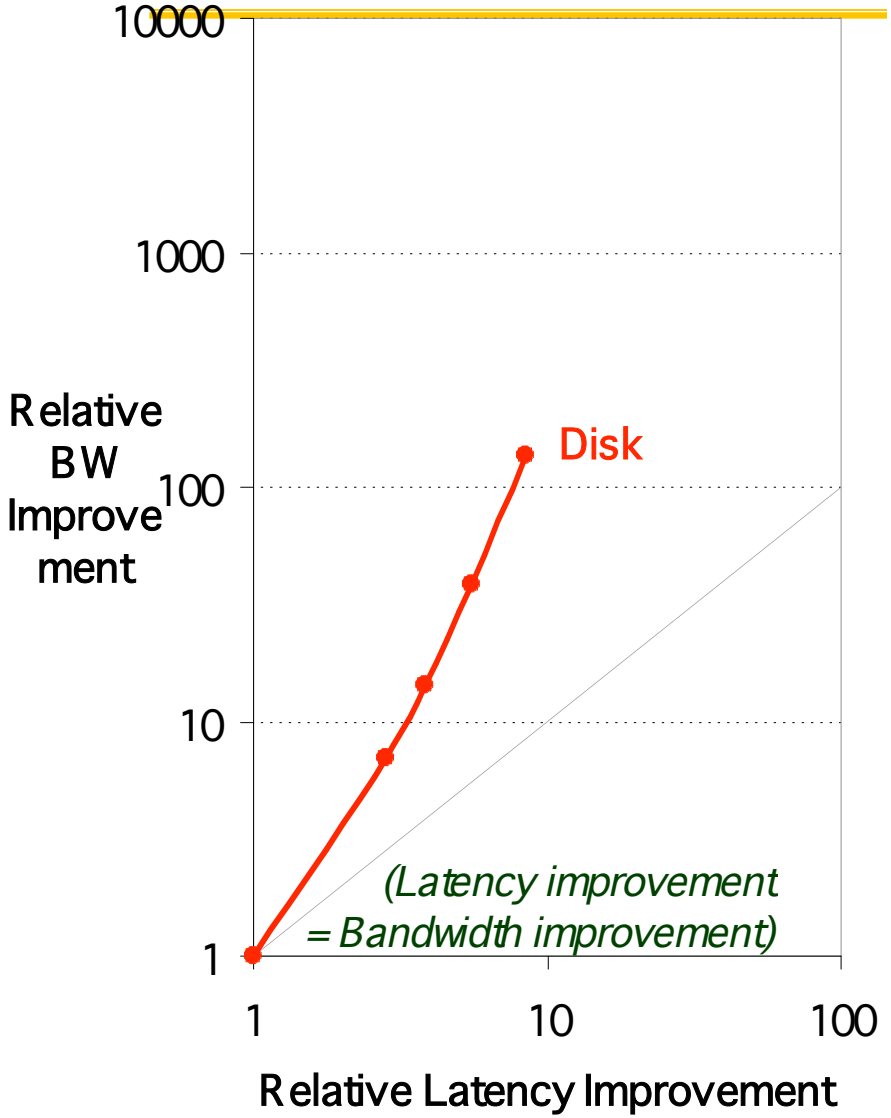
- Drill down into 4 technologies:
  - Disks; Memory; Network; Processors
- Compare ~1980 Archaic (Nostalgic) vs. ~2000 Modern (Newfangled)
  - Performance Milestones in each technology
- Compare for Bandwidth vs. Latency improvements in performance over time
- Bandwidth: number of events per unit time
  - E.g., M bits / second over network, M bytes / second from disk
- Latency: elapsed time for a single event
  - E.g., one-way network delay in microseconds,
    average disk access time in milliseconds

Disks: Archaic (Nostalgic) v. Modern (Newfangled)

- CDC Wren I, 1983
- 3600 RPM
- 0.03 GBytes capacity
- Tracks/Inch: 800
- Bits/Inch: 9550
- Three 5.25” platters
- Bandwidth: 0.6 MBytes/sec
- Latency: 48.3 ms
- Cache: none

- Seagate 373453, 2003
- 15000 RPM (4X)
- 73.4 GBytes (2500X)
- Tracks/Inch: 64000 (80X)
- Bits/Inch: 533,000 (60X)
- Four 2.5” platters (in 3.5” form factor)
- BW: 86 MB/sec (140X)
- Latency: 5.7 ms (8X)
- Cache: 8 MB

Disk Latency vs. Bandwidth



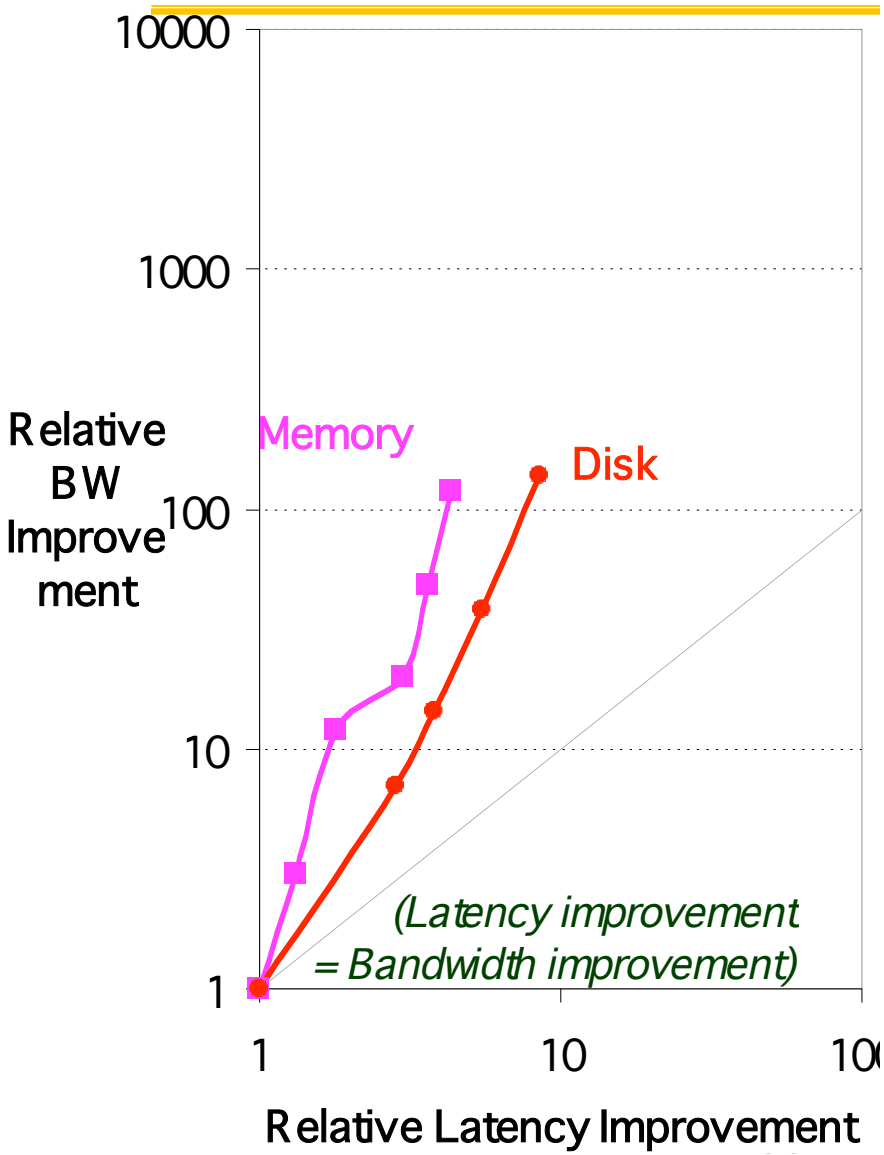
- Disk: 3600, 5400, 7200, 10000, 15000 RPM (8x, 143x)
- (latency = simple operation w/o contention; BW = best-case)

Memory: Archaic (Nostalgic) v. Modern (Newfangled)

- 1980 DRAM (asynchronous)
- 0.06 Mbits/chip
- 64,000 xtors, 35 mm²
- 16-bit data bus per module, 16 pins/chip
- 13 Mbytes/sec
- Latency: 225 ns
- (no block transfer)

- 2000 Double Data Rate Synchr. (clocked) DRAM
- 256 Mb/chip (4000X)
- 256M xtors, 204 mm²
- 64b data bus / DIMM, 66 pins/chip (4X)
- 1600 MB/sec (120X)
- Latency: 52 ns (4X)
- Block transfers (page mode)

Latency Lags Bandwidth (last ~20 years)



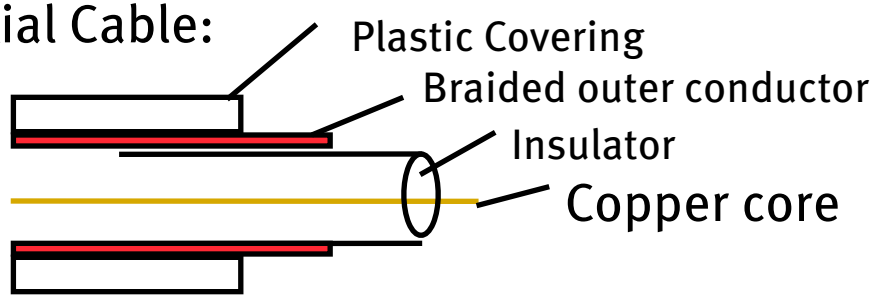
- Memory Module: 16 bit plain DRAM, Page Mode DRAM, 32b, 64b, SDRAM, DDR SDRAM (4x, 120x)
  - (latency = simple operation w/o contention; BW = best-case)

LANs: Archaic (Nostalgic) v. Modern (Newfangled)

- 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

Coaxial Cable:

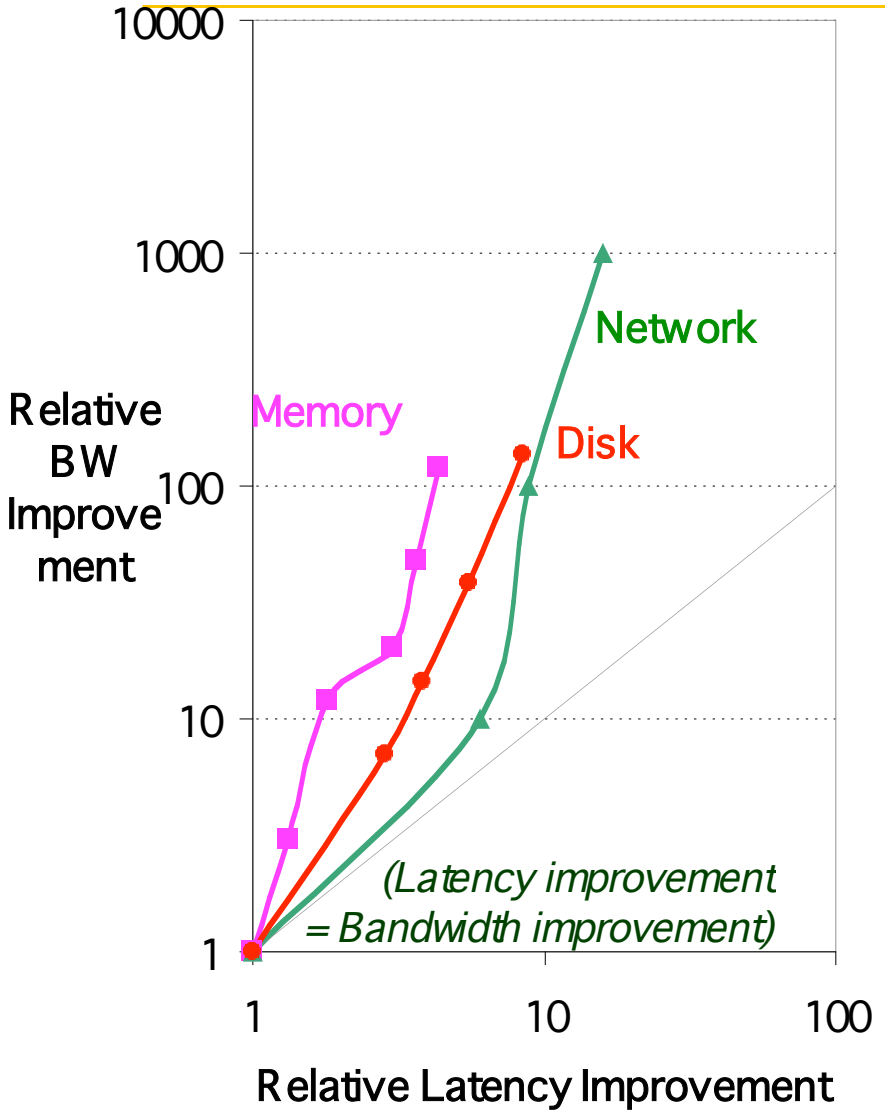


"Cat 5" is 4 twisted pairs in bundle

Twisted Pair:

Copper, 1mm thick,
twisted to avoid antenna effect

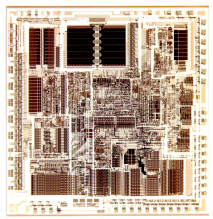
Latency Lags Bandwidth (last ~20 years)



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

CPUs: Archaic (Nostalgic) v. Modern (Newfangled)

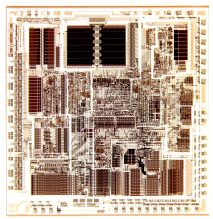
- 1982 Intel 80286
- 12.5 MHz
- 2 MIPS (peak)
- Latency 320 ns
- 134,000 xtors, 47 mm²
- 16-bit data bus, 68 pins
- Microcode interpreter, separate FPU chip
- (no caches)

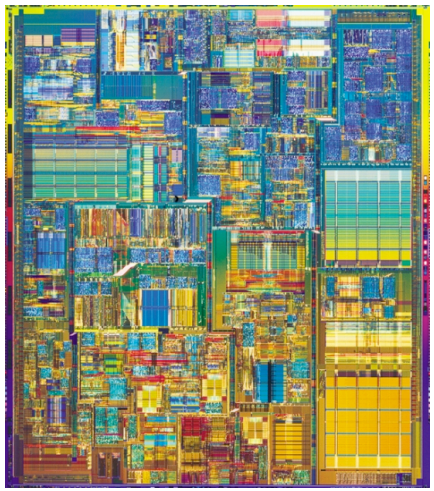


- 2001 Intel Pentium 4
- 1500 MHz (120X)
- 4500 MIPS (peak) (2250X)
- Latency 15 ns (20X)
- 42,000,000 xtors, 217 mm²
- 64-bit data bus, 423 pins
- 3-way superscalar, dynamic translate to RISC, Superpipelined (22 stage), Out-of-Order execution
- On-chip 8 KB Data caches, 96 KB Instr. Trace cache, 256 KB L2 cache

CPUs: Archaic (Nostalgic) v. Modern (Newfangled)

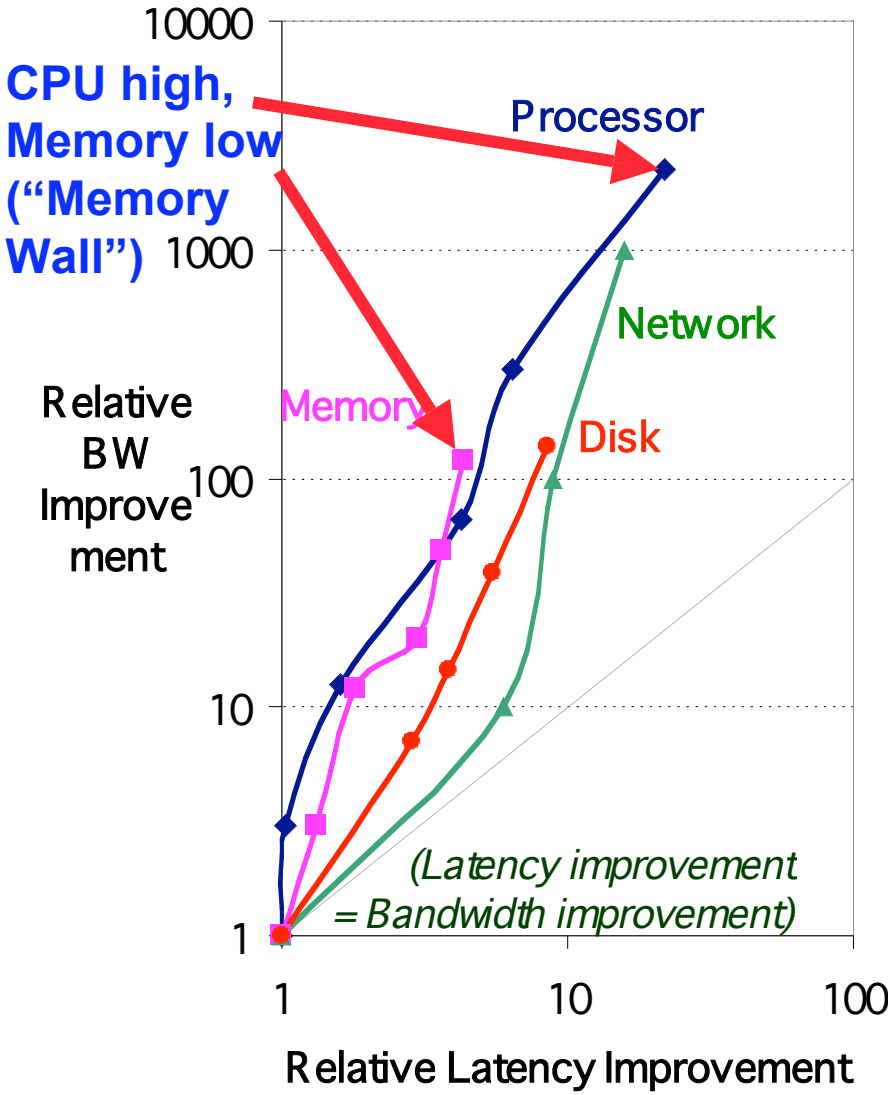
- 1982 Intel 80286
- 12.5 MHz
- 2 MIPS (peak)
- Latency 320 ns
- 134,000 xtors, 47 mm²
- 16-bit data bus, 68 pins
- Microcode interpreter, separate FPU chip
- (no caches)





- 2001 Intel Pentium 4
- 1500 MHz (120X)
- 4500 MIPS (peak) (2250X)
- Latency 15 ns (20X)
- 42,000,000 xtors, 217 mm²
- 64-bit data bus, 423 pins
- 3-way superscalar, dynamic translate to RISC, Superpipelined (22 stage), 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)

Rule of Thumb for Latency Lagging BW

- 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

6 Reasons Latency Lags Bandwidth

1. 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)

6 Reasons Latency Lags Bandwidth

- 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)

6 Reasons Latency Lags Bandwidth (cont'd)

2. 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?

6 Reasons Latency Lags Bandwidth (cont'd)

3. Bandwidth easier to sell ("bigger = better")
   - E.g., 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

6 Reasons Latency Lags Bandwidth (cont'd)

4. 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

6 Reasons Latency Lags Bandwidth (cont'd)

5. 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
6. Operating System overhead hurts latency more than Bandwidth
   - Long messages amortize overhead; overhead bigger part of short messages

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

Caching
Replication
Prediction

- 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