B649
Graduate Computer Architecture

Lec 2 - Introduction

Slides derived from David Patterson

Review from last lecture

- Computer Architecture >> instruction sets
- Computer Architecture skill sets are different
  - 5 Quantitative principles of design
  - Quantitative approach to design
  - Solid interfaces that really work
  - Technology tracking and anticipation
- Computer Science at the crossroads from sequential to parallel computing
  - Salvation requires innovation in many fields, including computer architecture

Review: Computer Architecture brings

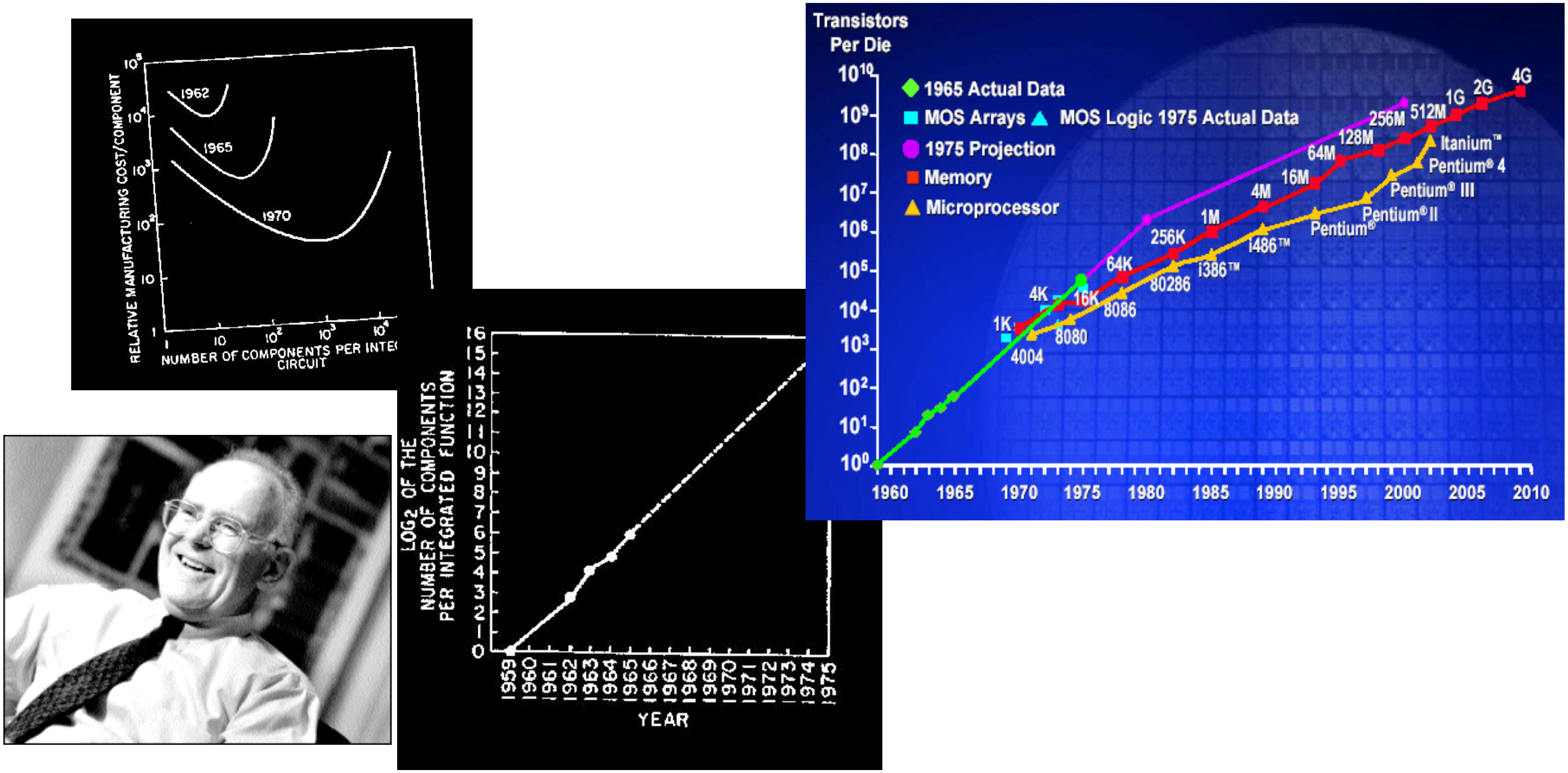
- Other fields often borrow ideas from architecture
- Quantitative Principles of Design
  1. Take Advantage of Parallelism
  2. Principle of Locality
  3. Focus on the Common Case
  4. Amdahl’s Law
  5. The Processor Performance Equation
- Careful, quantitative comparisons
  - Define, quantify, and summarize relative performance
  - Define and quantify relative cost
  - Define and quantify dependability
  - Define and quantify power
- Culture of anticipating and exploiting advances in technology
- Culture of well-defined interfaces that are carefully implemented and thoroughly checked

Outline

- Review
- Technology Trends: Culture of tracking, anticipating and exploiting advances in technology
- Careful, quantitative comparisons:

1. Define, quantify, and summarize relative performance
2. Define and quantify relative cost
3. Define and quantify dependability
4. Define and quantify power

Moore’s Law: 2X transistors / “year”



- “Cramming More Components onto Integrated Circuits”
  - Gordon Moore, Electronics, 1965
- # on transistors / cost-effective integrated circuit double every N months (12 ≤ N ≤ 24)

Tracking Technology Performance Trends

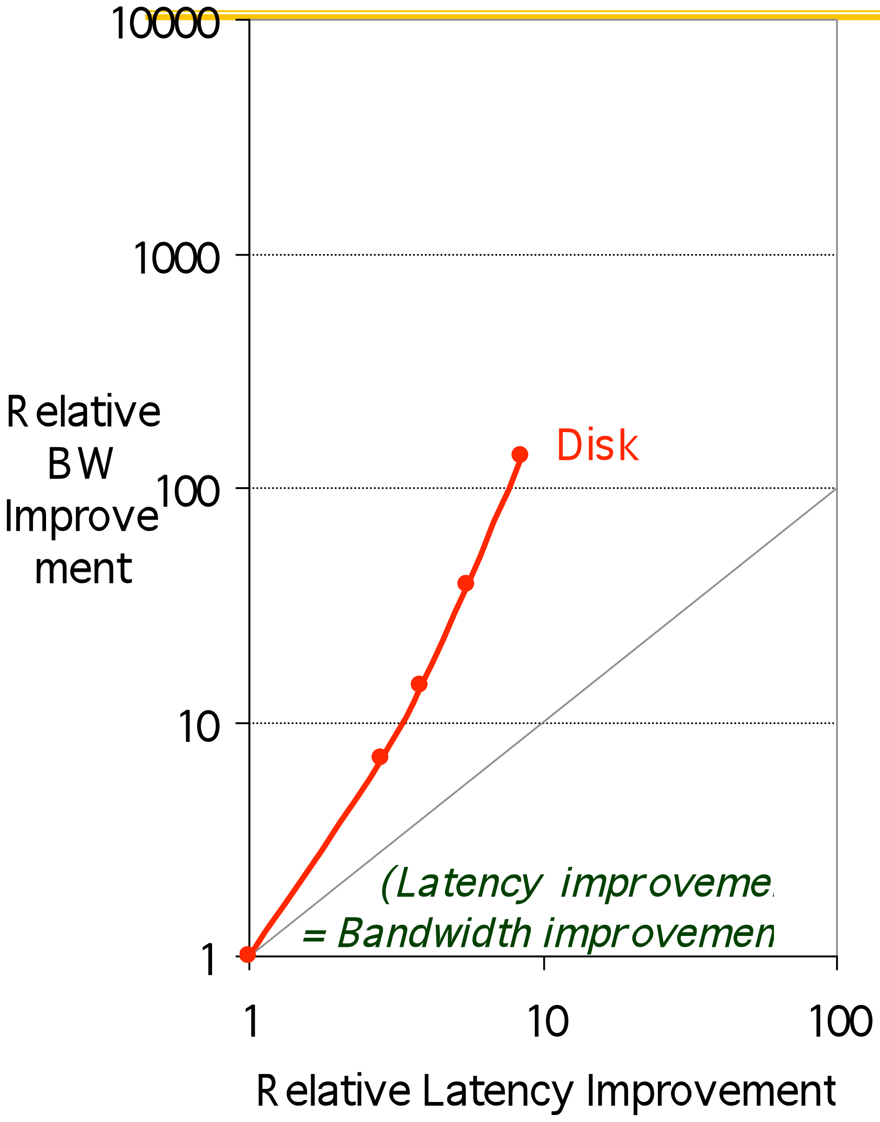
- **Drill down into 4 technologies:**
  - Disks,
  - Memory,
  - Network,
  - Processors
- **Compare ~1980 Archaic (Nostalgic) vs. ~2000 Modern (Newfangled)**
  - Performance Milestones in each technology
- **Compare for Bandwidth vs. Latency improvements in performance over time**
- **Bandwidth: number 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)
- Bandwidth:
  86 MBytes/sec (140X)
- Latency: 5.7 ms (8X)
- Cache: 8 MBytes

Latency Lags Bandwidth (for last ~20 years)



- **Performance Milestones**

- **Disk: 3600, 5400, 7200, 10000, 15000 RPM** **(8x, 143x)**

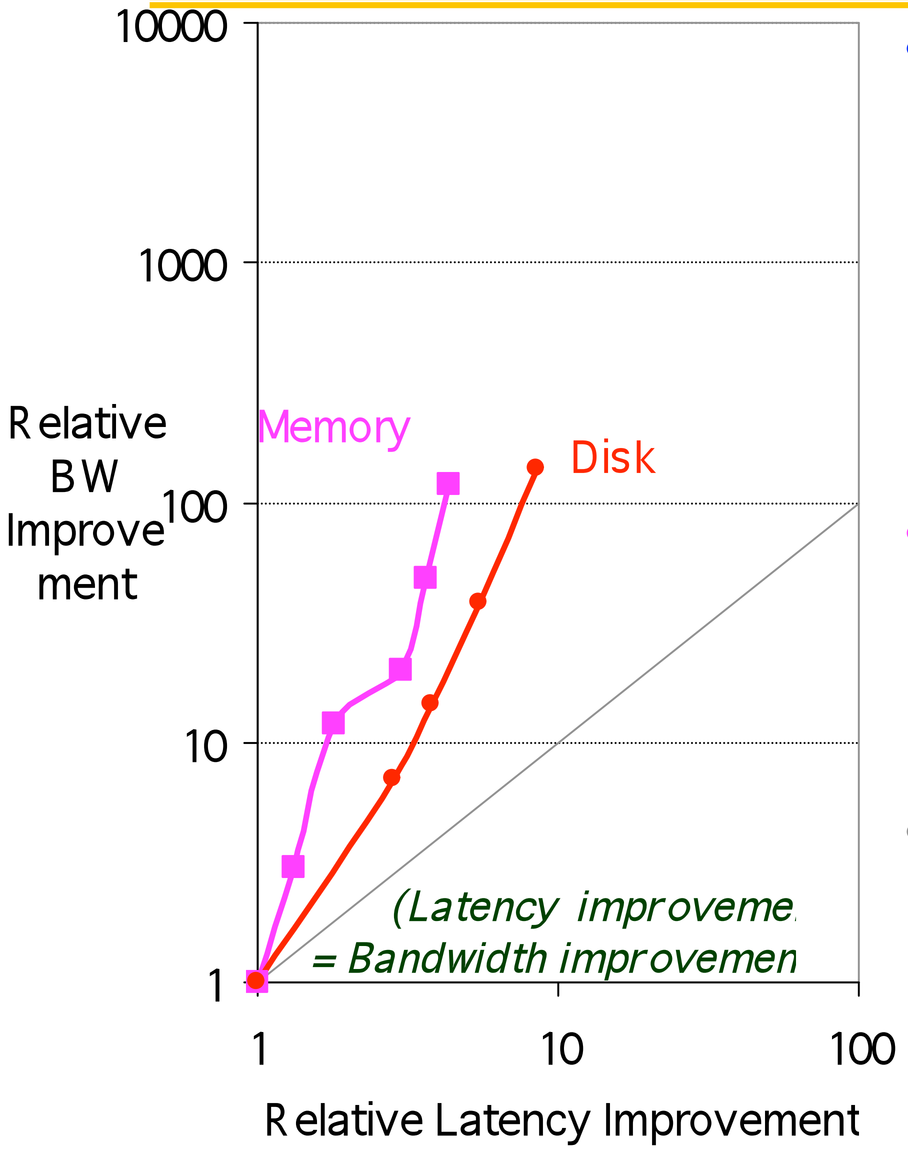
  (latency = simple operation w/o contention
  BW = best-case)

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

- 1980 DRAM (asynchronous)
- 0.06 Mbits/chip
- 64,000 xtors, 35 mm^2
- 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
- 256.00 Mbits/chip (4000X)
- 256,000,000 xtors, 204 mm^2
- 64-bit data bus per DIMM, 66 pins/chip (4X)
- 1600 Mbytes/sec (120X)
- Latency: 52 ns (4X)
- Block transfers (page mode)

Latency Lags Bandwidth (last ~20 years)



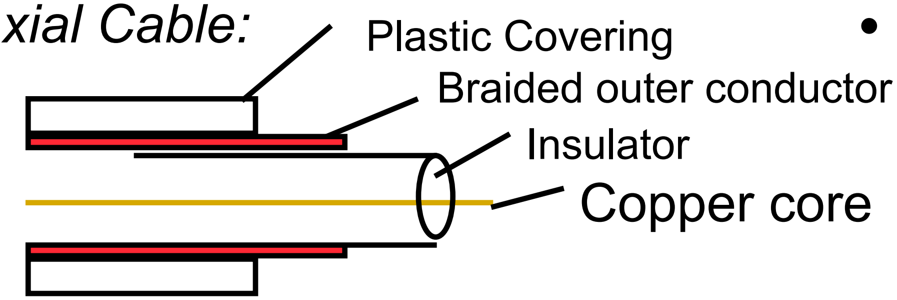
- **Performance Milestones**
- **Memory Module: 16bit plain DRAM, Page Mode DRAM, 32b, 64b, SDRAM, DDR SDRAM** **(4x,120x)**
- **Disk: 3600, 5400, 7200, 10000, 15000 RPM** **(8x, 143x)**

(latency = simple operation w/o contention
BW = best-case)

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

- Ethernet 802.3
- Year of Standard: 1978
- 10 Mbits/s
  link speed
- Latency: 3000 µsec
- Shared media
- Coaxial cable

Coaxial Cable:

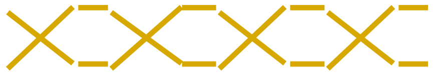


- Ethernet 802.3ae
- Year of Standard: 2003
- 10,000 Mbits/s

  (1000X)
  link speed
- Latency: 190 µsec (15X)
- Switched media
- Category 5 copper wire

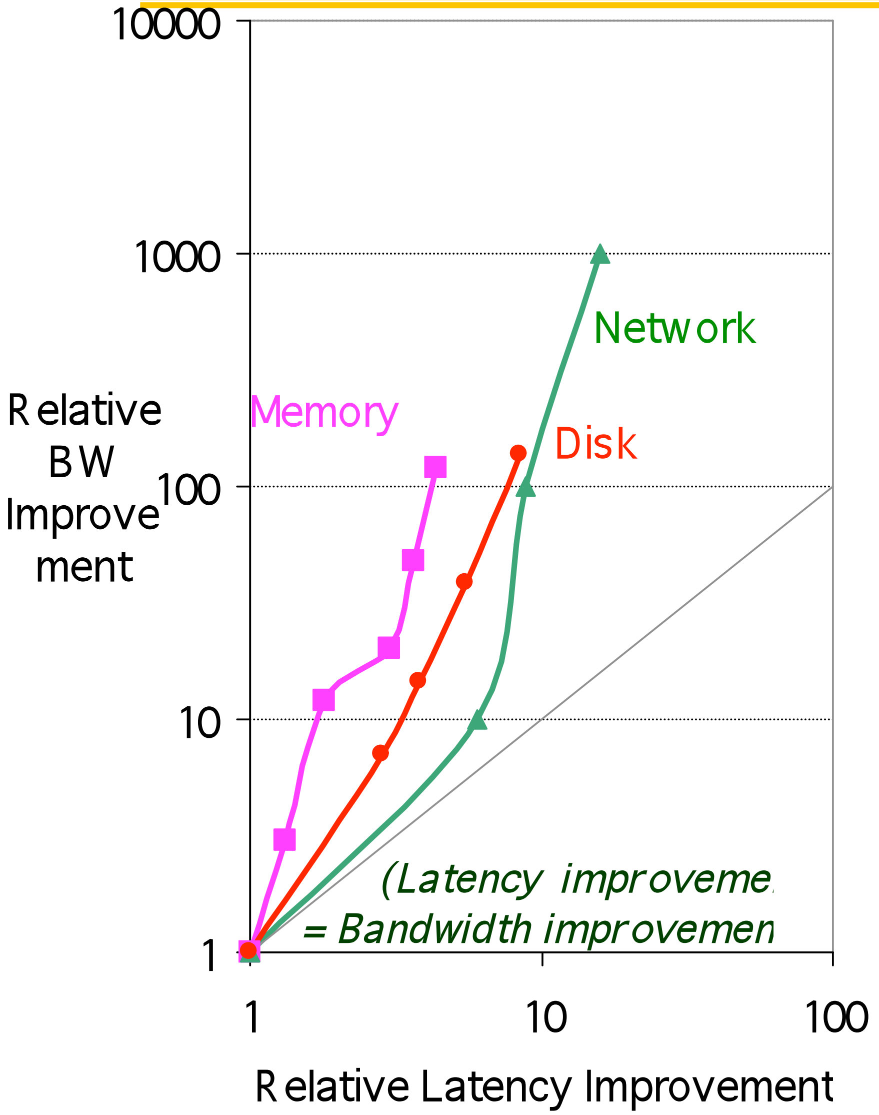
"Cat 5" is 4 twisted pairs in bundle

Twisted Pair:



Copper, 1mm thick,
twisted to avoid antenna effect

Latency Lags Bandwidth (last ~20 years)



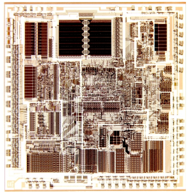
- **Performance Milestones**

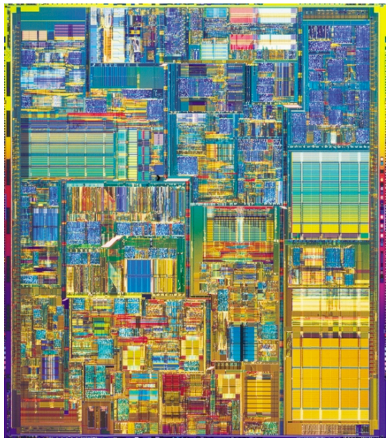
- **Ethernet: 10Mb, 100Mb, 1000Mb, 10000 Mb/s** (16x,1000x)
- **Memory Module: 16bit plain DRAM, Page Mode DRAM, 32b, 64b, SDRAM, DDR SDRAM** (4x,120x)
- **Disk: 3600, 5400, 7200, 10000, 15000 RPM** (8x, 143x)

(latency = simple operation w/o contention
BW = best-case)

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

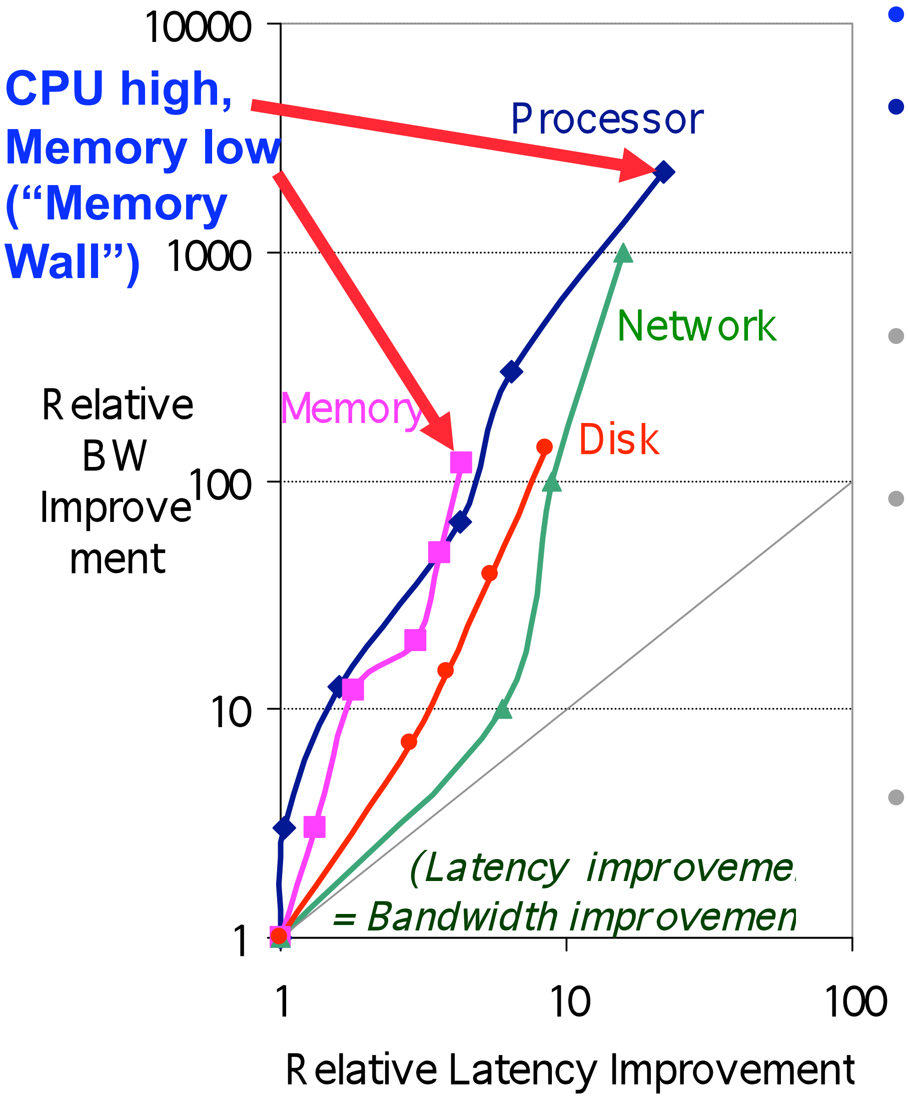
- 1982 Intel 80286
- 12.5 MHz
- 2 MIPS (peak)
- Latency 320 ns
- 134,000 xtors, 47 mm^2
- 16-bit data bus, 68 pins
- Microcode interpreter, separate FPU chip
- (no caches)





- 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 stage), Out-of-Order execution
- On-chip 8KB Data caches, 96KB Instr. Trace cache, 256KB L2 cache

Latency Lags Bandwidth (last ~20 years)



CPU high, Memory low ("Memory Wall")

- **Performance Milestones**
- **Processor: '286, '386, '486, Pentium, Pentium Pro, Pentium 4** (21x,2250x)
- **Ethernet: 10Mb, 100Mb, 1000Mb, 10000 Mb/s** (16x,1000x)
- **Memory Module: 16bit plain DRAM, Page Mode DRAM, 32b, 64b, SDRAM, DDR SDRAM** (4x,120x)
- **Disk : 3600, 5400, 7200, 10000, 15000 RPM** (8x, 143x)

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

Computers in the News

- “Intel loses market share in own backyard,” By Tom Krazit, CNET News.com, 1/18/2006
- “Intel's share of the U.S. retail PC market fell by 11 percentage points, from 64.4 percent in the fourth quarter of 2004 to 53.3 percent. … Current Analysis' market share numbers measure U.S. retail sales only, and therefore exclude figures from Dell, which uses its Web site to sell directly to consumers. …
  AMD chips were found in 52.5 percent of desktop PCs sold in U.S. retail stores during that period.”
- Technical advantages of AMD Opteron/Athlon vs. Intel Pentium 4 as we’ll see in this course.

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)**
   - **Smaller, faster transistors but communicate over (relatively) longer lines: limits latency**
     - » **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?**

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**
- **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**

Outline

- Review
- Technology Trends: Culture of tracking, anticipating and exploiting advances in technology
- Careful, quantitative comparisons:

1. Define and quantity power
2. Define and quantity dependability
3. Define, quantity, and summarize relative performance
4. Define and quantity relative cost

Define and quantify power (1 / 2)

- For CMOS chips, traditional dominant energy consumption has been in switching transistors, called *dynamic power*

$$Power_{dynamic} = 1/2 \times CapacitiveLoad \times Voltage^2 \times FrequencySwitched$$

- For mobile devices, energy better metric

$$Energy_{dynamic} = CapacitiveLoad \times Voltage^2$$

- For a fixed task, slowing clock rate (frequency switched) reduces power, but not energy
- Capacitive load a function of number of transistors connected to output and technology, which determines capacitance of wires and transistors
- Dropping voltage helps both, so went from 5V to 1V
- To save energy & dynamic power, most CPUs now turn off clock of inactive modules (e.g. Fl. Pt. Unit)

Example of quantifying power

- Suppose 15% reduction in voltage results in a 15% reduction in frequency. What is impact on dynamic power?

$$
\begin{aligned}
Power_{dynamic} &= 1/2 \times CapacitiveLoad \times Voltage^2 \times FrequencySwitched \\
&= 1/2 \times .85 \times CapacitiveLoad \times (.85 \times Voltage)^2 \times FrequencySwitched \\
&= (.85)^3 \times OldPower_{dynamic} \\
&\approx 0.6 \times OldPower_{dynamic}
\end{aligned}
$$

Define and quantify power (2 / 2)

- Because leakage current flows even when a transistor is off, now *static power* important too

$$Power_{static} = Current_{static} \times Voltage$$

- Leakage current increases in processors with smaller transistor sizes
- Increasing the number of transistors increases power even if they are turned off
- In 2006, goal for leakage is 25% of total power consumption; high performance designs at 40%
- Very low power systems even gate voltage to inactive modules to control loss due to leakage

Outline

- Review
- Technology Trends: Culture of tracking, anticipating and exploiting advances in technology
- Careful, quantitative comparisons:

1. Define and quantity power
2. Define and quantity dependability
3. Define, quantity, and summarize relative performance
4. Define and quantity relative cost

Define and quantify dependability (1/3)

- How decide when a system is operating properly?
- Infrastructure providers now offer Service Level Agreements (SLA) to guarantee that their networking or power service would be dependable
- Systems alternate between 2 states of service with respect to an SLA:

1. Service accomplishment, where the service is delivered as specified in SLA
2. Service interruption, where the delivered service is different from the SLA

- Failure = transition from state 1 to state 2
- Restoration = transition from state 2 to state 1

Define and quantify dependability (2/3)

- *Module reliability* = measure of continuous service accomplishment (or time to failure).
  2 metrics

1. *Mean Time To Failure* (*MTTF*) measures Reliability
2. *Failures In Time* (*FIT*) = 1/MTTF, the rate of failures
   - Traditionally reported as failures per billion hours of operation

- *Mean Time To Repair* (*MTTR*) measures Service Interruption
  - *Mean Time Between Failures* (*MTBF*) = MTTF+MTTR
- *Module availability* measures service as alternate between the 2 states of accomplishment and interruption (number between 0 and 1, e.g. 0.9)
- *Module availability = MTTF / (MTTF + MTTR)*

Example calculating reliability

- If modules have *exponentially distributed lifetimes* (age of module does not affect probability of failure), overall failure rate is the sum of failure rates of the modules
- Calculate FIT and MTTF for 10 disks (1M hour MTTF per disk), 1 disk controller (0.5M hour MTTF), and 1 power supply (0.2M hour MTTF):

$$FailureRate =$$

$$MTTF =$$

Example calculating reliability

- If modules have *exponentially distributed lifetimes* (age of module does not affect probability of failure), overall failure rate is the sum of failure rates of the modules
- Calculate FIT and MTTF for 10 disks (1M hour MTTF per disk), 1 disk controller (0.5M hour MTTF), and 1 power supply (0.2M hour MTTF):

$$
\begin{aligned}
FailureRate &= 10 \times (1/1{,}000{,}000) + 1/500{,}000 + 1/200{,}000 \\
&= 10 + 2 + 5/1{,}000{,}000 \\
&= 17/1{,}000{,}000 \\
&= 17{,}000 FIT \\
MTTF &= 1{,}000{,}000{,}000/17{,}000 \\
&\approx 59{,}000 hours
\end{aligned}
$$

Outline

- Review
- Technology Trends: Culture of tracking, anticipating and exploiting advances in technology
- **Careful, quantitative comparisons:**

1. Define and quantify power
2. Define and quantify dependability
3. **Define, quantify, and summarize relative performance**
4. **Define and quantify relative cost**

Definition: Performance

- Performance is in units of things per sec
  - bigger is better
- If we are primarily concerned with response time

$$\text{performance}(x) = \frac{1}{\text{execution\_time}(x)}$$

" X is n times faster than Y" means

$$n = \frac{\text{Performance}(X)}{\text{Performance}(Y)} = \frac{\text{Execution\_time}(Y)}{\text{Execution\_time}(X)}$$

Performance: What to measure

- Usually rely on benchmarks vs. real workloads
- To increase predictability, collections of benchmark applications, called *benchmark suites*, are popular
- SPECCPU: popular desktop benchmark suite
  - CPU only, split between integer and floating point programs
  - SPECint2000 has 12 integer, SPECfp2000 has 14 integer pgms
  - SPECCPU2006 to be announced Spring 2006
  - SPECSFS (NFS file server) and SPECWeb (WebServer) added as server benchmarks
- Transaction Processing Council measures server performance and cost-performance for databases
  - TPC-C Complex query for Online Transaction Processing
  - TPC-H models ad hoc decision support
  - TPC-W a transactional web benchmark
  - TPC-App application server and web services benchmark

How Summarize Suite Performance (1/5)

- Arithmetic average of execution time of all pgms?
  - But they vary by 4X in speed, so some would be more important than others in arithmetic average
- Could add a weights per program, but how pick weight?
  - Different companies want different weights for their products
- SPECRatio: Normalize execution times to reference computer, yielding a ratio proportional to performance =

$$\frac{\text{time on reference computer}}{\text{time on computer being rated}}$$

How Summarize Suite Performance (2/5)

- **If program SPECRatio on Computer A is 1.25 times bigger than Computer B, then**

$$1.25 = \frac{SPECRatio_A}{SPECRatio_B} = \frac{\dfrac{ExecutionTime_{ref\ erence}}{ExecutionTime_A}}{\dfrac{ExecutionTime_{ref\ erence}}{ExecutionTime_B}}$$

$$= \frac{ExecutionTime_B}{ExecutionTime_A} = \frac{Perf\ ormane_A}{Perf\ ormane_B}$$

- **Note that when comparing 2 computers as a ratio, execution times on the reference computer drop out, so choice of reference computer is irrelevant**

How Summarize Suite Performance (3/5)

- Since ratios, proper mean is geometric mean (SPECRatio unitless, so arithmetic mean meaningless)

$$GeometricMean = \sqrt[n]{\prod_{i=1}^{n} SPECRatio_i}$$

1. Geometric mean of the ratios is the same as the ratio of the geometric means
2. Ratio of geometric means
   = Geometric mean of performance ratios
   ⇒ choice of reference computer is irrelevant!

- These two points make geometric mean of ratios attractive to summarize performance

How Summarize Suite Performance (4/5)

- Does a single mean well summarize performance of programs in benchmark suite?
- Can decide if mean a good predictor by characterizing variability of distribution using standard deviation
- Like geometric mean, geometric standard deviation is multiplicative rather than arithmetic
- Can simply take the logarithm of SPECRatios, compute the standard mean and standard deviation, and then take the exponent to convert back:

$$GeometricMean = \exp\left(\frac{1}{n} \times \sum_{i=1}^{n} \ln\left(SPECRatio_i\right)\right)$$

$$GeometricStDev = \exp\left(StDev\left(\ln\left(SPECRatio_i\right)\right)\right)$$

How Summarize Suite Performance (5/5)

- Standard deviation is more informative if know distribution has a standard form
  - *bell-shaped normal distribution*, whose data are symmetric around mean
  - *lognormal distribution*, where logarithms of data--not data itself--are normally distributed (symmetric) on a logarithmic scale
- For a lognormal distribution, we expect that

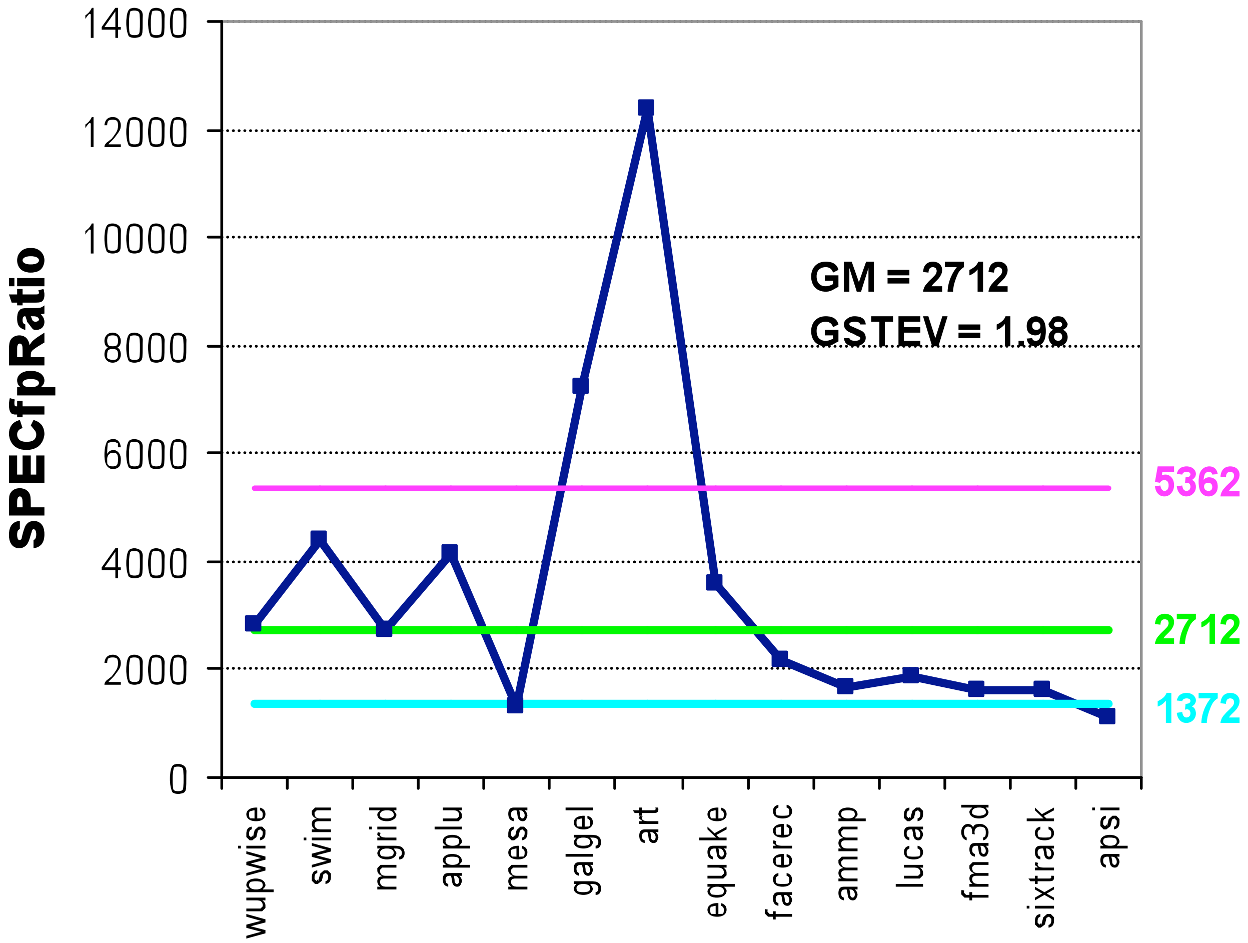
68% of samples fall in range $\left[mean / gstdev, mean \times gstdev \right]$

95% of samples fall in range $\left[mean / gstdev^2 , mean \times gstdev^2 \right]$

- Note: Excel provides functions EXP(), LN(), and STDEV() that make calculating geometric mean and multiplicative standard deviation easy

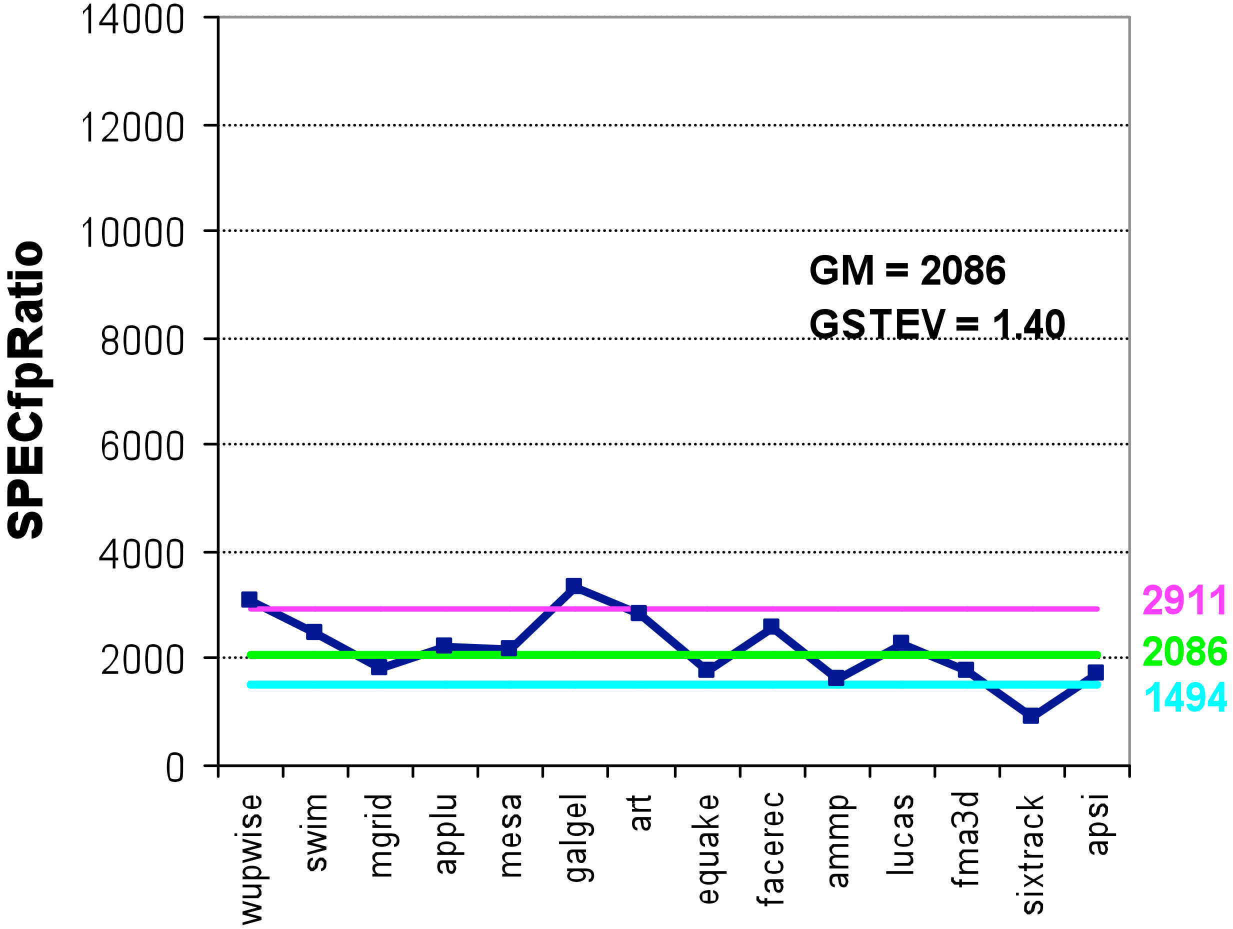
Example Standard Deviation (1/2)

- **GM and multiplicative StDev of SPECfp2000 for Itanium 2**



Example Standard Deviation (2/2)

- **GM and multiplicative StDev of SPECfp2000 for AMD Athlon**



Comments on Itanium 2 and Athlon

- Standard deviation of 1.98 for Itanium 2 is much higher-- vs. 1.40--so results will differ more widely from the mean, and therefore are likely less predictable
- Falling within one standard deviation:
  - 10 of 14 benchmarks (71%) for Itanium 2
  - 11 of 14 benchmarks (78%) for Athlon
- Thus, the results are quite compatible with a lognormal distribution (expect 68%)

And in conclusion …

- Tracking and extrapolating technology part of architect's responsibility
- Expect Bandwidth in disks, DRAM, network, and processors to improve by at least as much as the square of the improvement in Latency
- Quantify dynamic and static power
  - Capacitance x Voltage^2 x frequency, Energy vs. power
- Quantify dependability
  - Reliability (MTTF, FIT), Availability (99.9…)
- Quantify and summarize performance
  - Ratios, Geometric Mean, Multiplicative Standard Deviation
- Read Appendix A