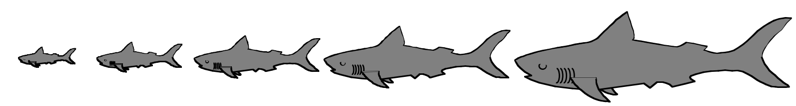
Lectures 1 : Review of Technology Trends, Cost/Performance, and Instruction Sets

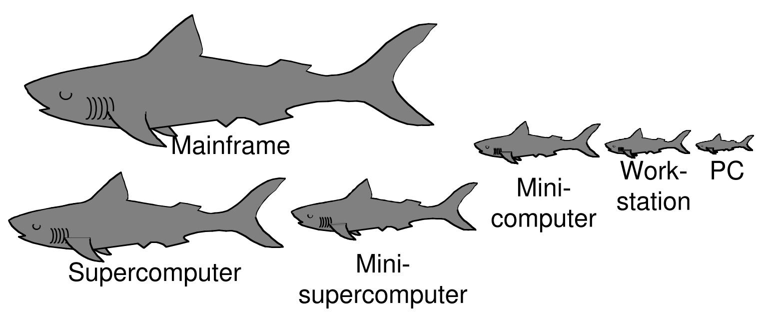
Prof. David A. Patterson
Computer Science 252
Fall 1996

Original Food Chain Picture

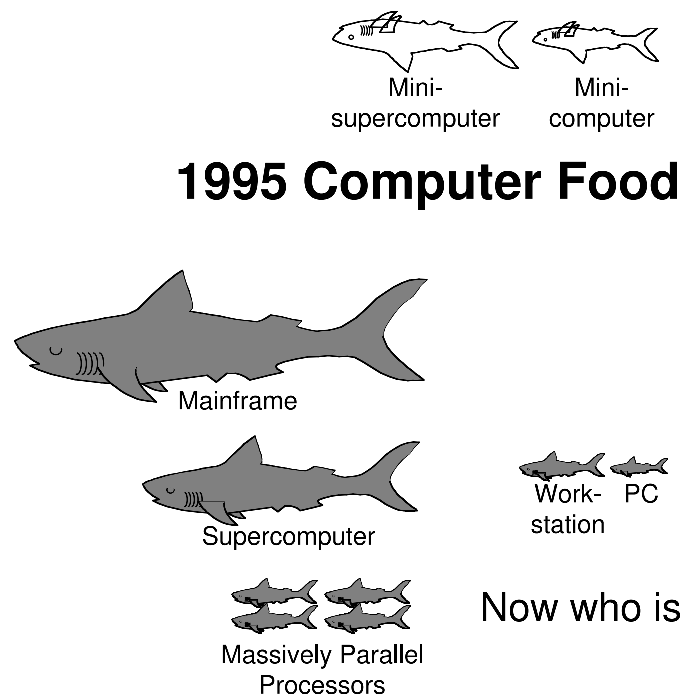


Big Fishes Eating Little Fishes

1985 Computer Food Chain



1995 Computer Food Chain

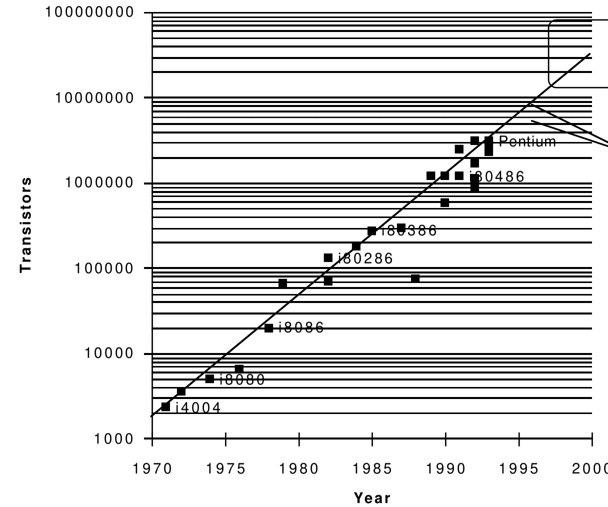


Now who is eating whom?

Why Such Change in 10 years?

- Function
  - Rise of networking/local interconnection technology
- Performance
  - Technology Advances
    - » CMOS VLSI dominates older technologies (TTL, ECL) in cost *AND* performance
  - Computer architecture advances improves low-end
    - » RISC, superscalar, RAID, …
- Price: Lower costs due to …
  - Simpler development
    - » CMOS VLSI: smaller systems, fewer components
  - Higher volumes
    - » CMOS VLSI : same dev. cost 10,000 vs. 10,000,000 units
  - Lower margins by class of computer, due to fewer services

Technology Trends: Microprocessor Capacity

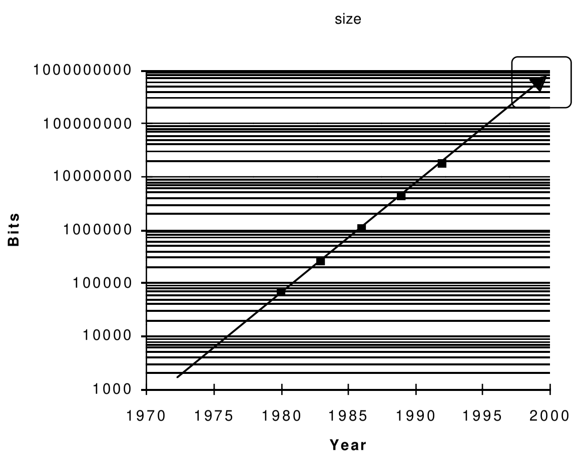


"Graduation Window"

Pentium Pro: 5.5 million
PowerPC 620: 6.9 million
Alpha 21164: 9.3 million
Sparc Ultra: 5.2 million

CMOS improvements:
- **Die size: 2X every 3 yrs**
- **Line width: halve / 7 yrs**

Memory Capacity (Single Chip DRAM)

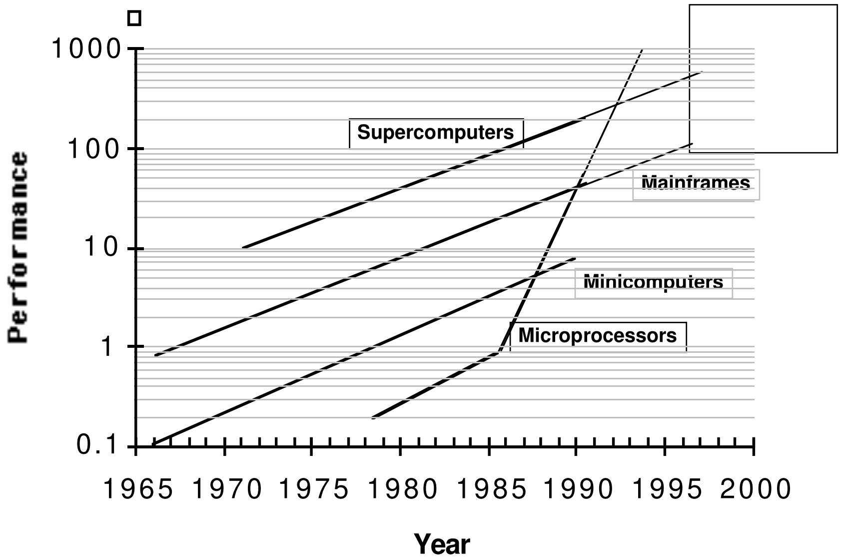


year	size	cyc time
1980	64 Kb	250 ns
1983	256 Kb	220 ns
1986	1 Mb	190 ns
1989	4 Mb	165 ns
1992	16 Mb	145 ns
1996	64 Mb	??

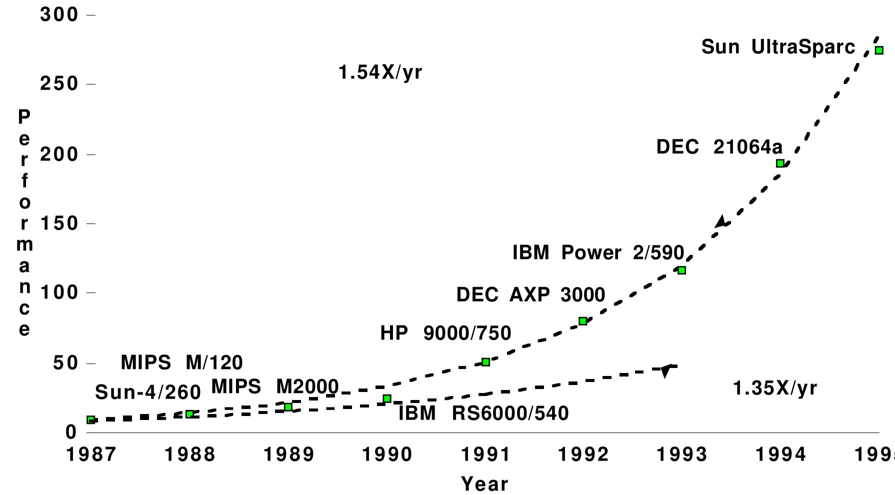
Technology Trends (Summary)

	Capacity	Speed
Logic	2x in 3 years	2x in 3 years
DRAM	4x in 3 years	1.4x in 10 years
Disk	4x in 3 years	1.4x in 10 years

Processor Performance Trends



Processor Performance

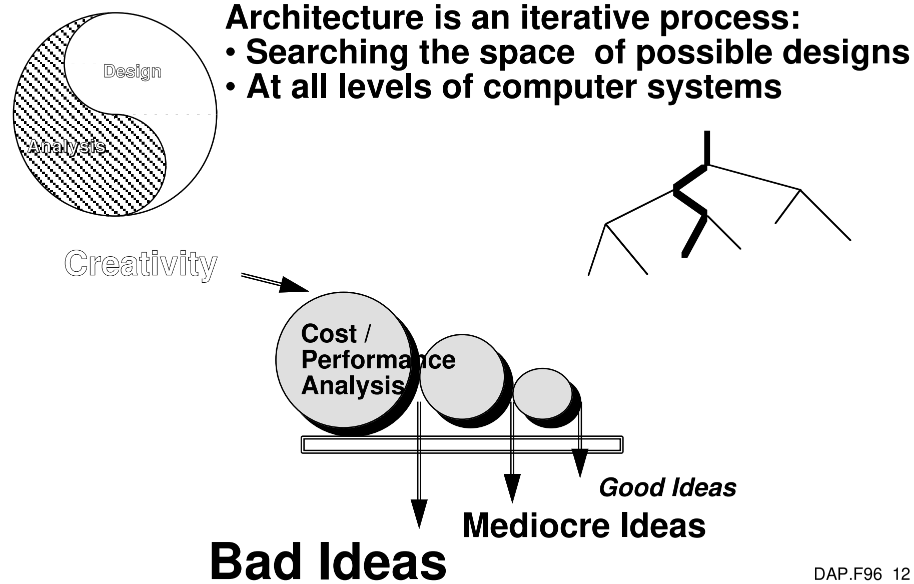


Performance Trends (Summary)

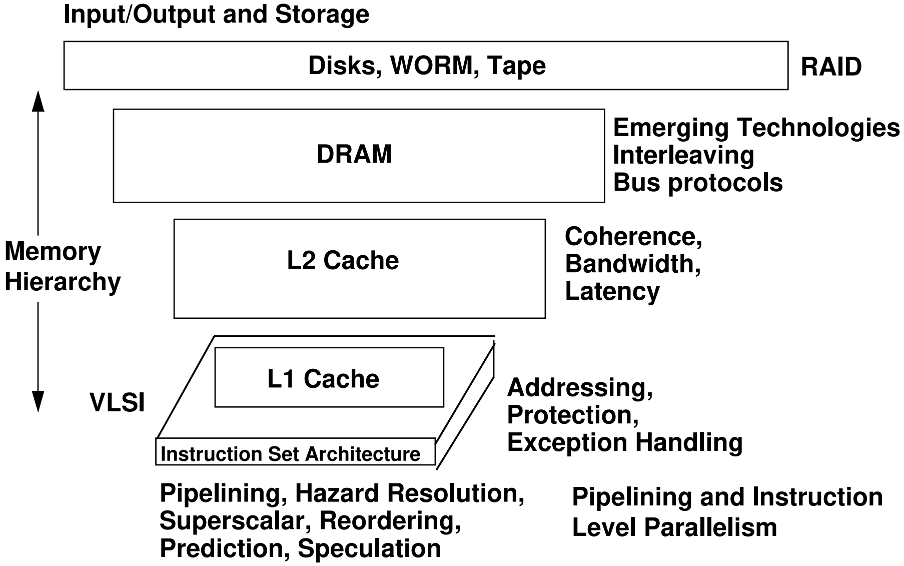
- Workstation performance (measured in Spec Marks) improves roughly 50% per year (2X every 18 months)
- Improvement in cost performance estimated at 70% per year

Measurement and Evaluation

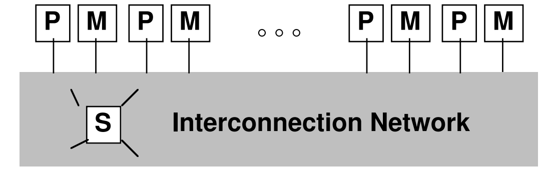
Architecture is an iterative process:
- Searching the space of possible designs
- At all levels of computer systems



Computer Architecture Topics



Computer Architecture Topics



Processor-Memory-Switch

Multiprocessors
Networks and Interconnections

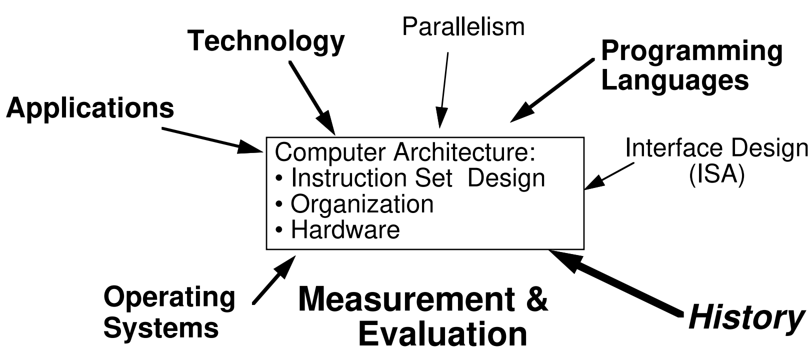
Shared Memory,
Message Passing,
Data Parallel

Network Interfaces

Topologies,
Routing,
Bandwidth,
Latency,
Reliability

CS 252 Course Focus

Understanding the design techniques, machine structures, technology factors, evaluation methods that will determine the form of computers in 21st Century



Topic Coverage

Textbook: Hennessy and Patterson, *Computer Architecture: A Quantitative Approach*, 2nd Ed., 1995.

- **Review: Fundamentals of Computer Architecture (Chapter 1), Instruction Set Architecture (Chapter 2), Pipelining (Chapter 3)**
- **Pipelining and Instructional Level Parallelism (Chapter 4)**
- **Memory Hierarchy (Chapter 5)**
- **Input/Output and Storage (Chapter 6)**
- **Networks and Interconnection Technology (Chapter 7)**
- **Multiprocessors (Chapter 8 + Culler book draft Chapter 1)**
- **Research: NOW, Reconfigurable MPer, DRAM+MPer, Wireless**

CS252: Staff

Instructor: David A. Patterson
Office: 635 Soda Hall, 642-6587 patterson@cs
Office Hours: Mon 1- 2, Wed 2-3 or by appt.
(Contact Patric Bodin, 643-7066, bodin@cs, 634 Soda)

T. A: Rich Fromm
Office: 479 Soda Hall, 642-9669 rfromm @cs
TA Office Hours TBD

Class: Wed, Fri 12:40:00 - 2:00:00 203 McLaughlin

Discussion: Tue. 2-3 B1 North Gate Hall, Thu. 3-4 10 Wellman Hall

Text: Computer Architecture: A Quantitative Approach, Second Edition (1996) (≥ second printing)

Web page: http://http.cs.berkeley.edu/~patterson/252/
Lectures available online <11AM day of lecture

Newsgroup: ucb.class.c252

Lecture style

- 1-Minute Review
- 20-Minute Lecture
- 3- Minute Administrative Matters
- 25-Minute Lecture
- 5-Minute Break (water, stretch)
- 25-Minute Lecture
- 1-Minute Summary
- I'll come to class early & stay after to answer questions

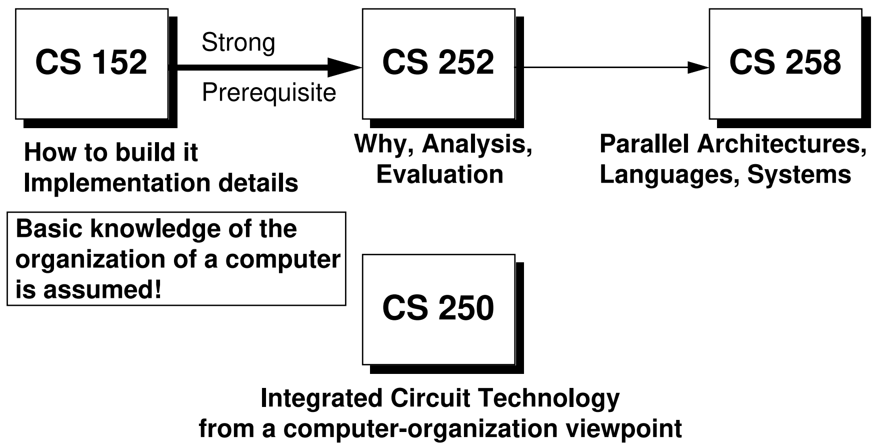
Course Style

- Reduce the pressure of taking quizes
  - <u>Only 2 Graded Quizes: Wednesday Oct 9 and Wed. Nov 20</u>
  - Our goal: test knowledge vs. speed writing
  - Both mid-term quizes can bring summary sheets
  - 3 hrs to take 1.5-hr test (5-8 PM, Sibley Auditorium)
  - Last chance Q&A: during class time day of exam
- Students/Staff meet over free pizza:
  Wed Oct 9 (8 PM) and Wed Nov 20 (8 PM)

Grading

- 10% Class Participation
- 30% Homeworks (work in pairs)
- 30% Examinations (2 Midterms)
- 30% Research Project (work in pairs)
  - pick topic
  - meet 3 times with faculty/TA to see progress
  - give oral presentation
  - give poster session
  - written report like conference paper
  - ≈ 3 weeks work full time for 2 people
  - Opportunity to do "research in the small" to help make transition from good student to research colleague

Related Courses



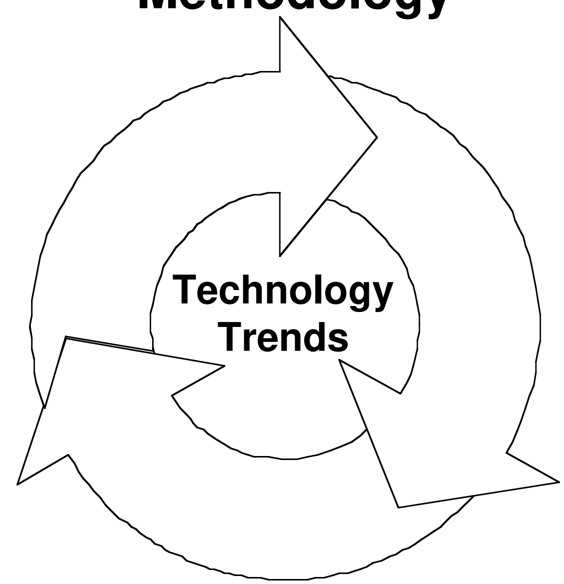
Coping with CS 252

- My last CS 252 = my worst teaching experience
- Too many students with too varied background?
- Last time 60 students:
  - To give proper attention to projects (as well as homeworks and quizes), I can handle up to 36 students
- Limiting Number of Students
  - First priority is first year CS/ EECS grad students
  - Second priority is N-th year CS/ EECS grad students
  - Third priority is College of Engineering grad students
  - Fourth priority is CS/EECS undegraduate seniors
  - All other categories
- If not this semester, 252 is offered regularily; CS 250 satisfies CS requirement

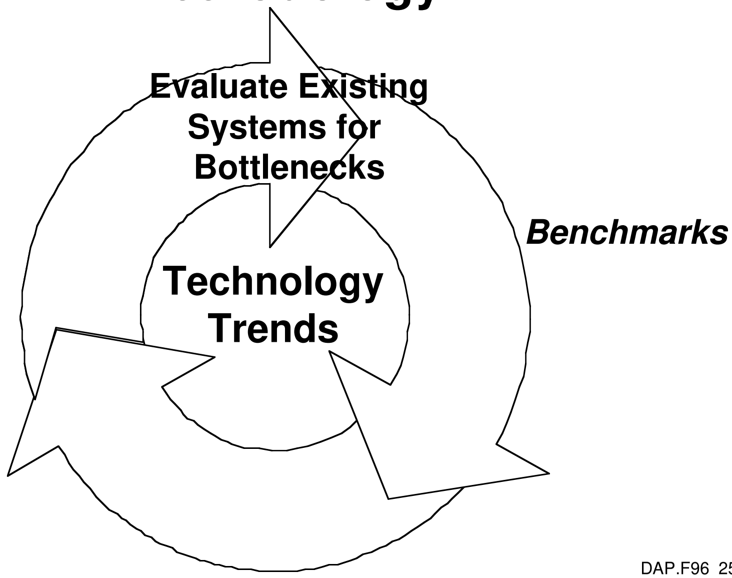
Coping with CS 252

- Students with too varied background?
  - In past, CS grad students took written prelim exams on undergraduate material in hardware, software, and theory
  - 1st 5 weeks reviewed background, helped 252, 262, 270
  - Prelims were dropped => some unprepared for CS 252?
- In class exam on Wednesday September 3
  - Doesn't affect grade, only admission into class
  - 2 grades: Admitted or audit/take CS 152 1st (same time in 306)
  - Improve your experience if recapture common background
- Review: Chapters 1- 3, CS 152 home page, maybe "Computer Organization and Design (COD)"
  - Chapters 1 to 8 of COD if never took prerequisite
  - If did take a class, be sure COD Chapters 2, 6, 7 are familiar
  - Copies in Bechtel Library on 2-hour reserve

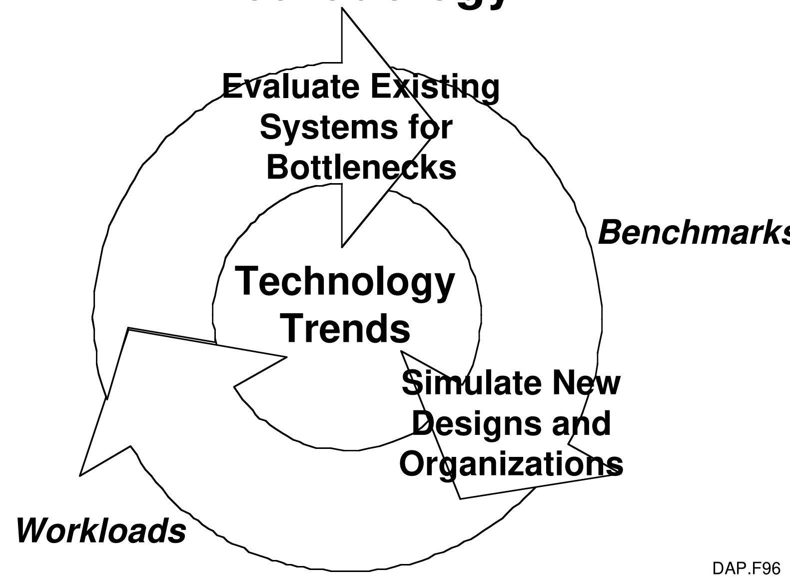
Computer Engineering Methodology



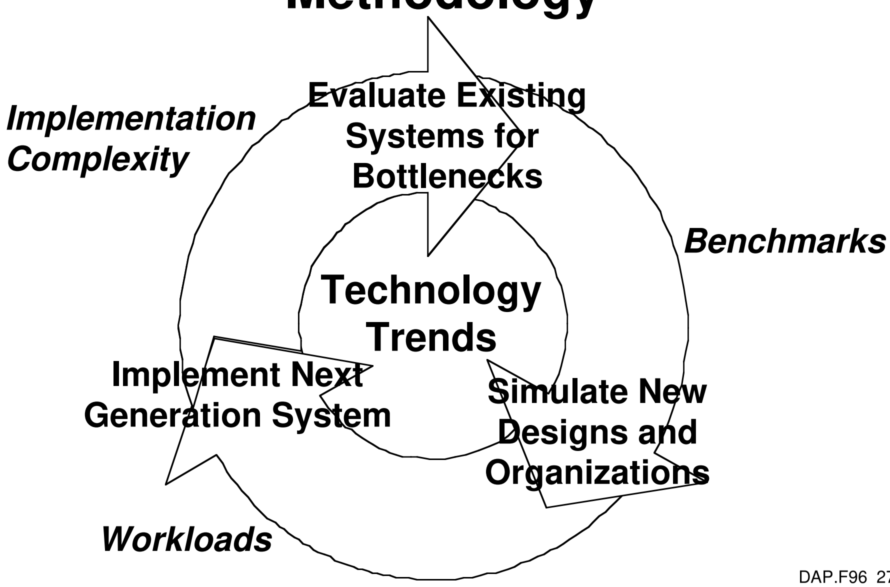
Computer Engineering Methodology



Computer Engineering Methodology



Computer Engineering Methodology



Measurement Tools

- Benchmarks, Traces, Mixes
- Cost, delay, area, power estimation
- Simulation (many levels)
  - ISA, RT, Gate, Circuit
- Queuing Theory
- Rules of Thumb
- Fundamental Laws

The Bottom Line: Performance (and Cost)

Plane	DC to Paris	Speed	Passengers	Throughput (pmph)
Boeing 747	6.5 hours	610 mph	470	286,700
BAD/Sud Concodre	3 hours	1350 mph	132	178,200

- **Time to run the task (ExTime)**
  - **Execution time, response time, latency**
- **Tasks per day, hour, week, sec, ns … (Performance)**
  - **Throughput, bandwidth**

The Bottom Line: Performance (and Cost)

"X is n times faster than Y" means

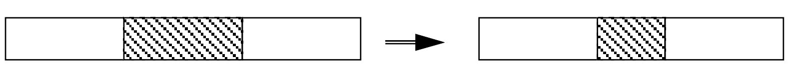
$$\frac{\text{ExTime(Y)}}{\text{ExTime(X)}} = \frac{\text{Performance(X)}}{\text{Performance(Y)}}$$

- **Speed of Concorde vs. Boeing 747**
- **Throughput of Boeing 747 vs. Concorde**

Amdahl's Law

Speedup due to enhancement E:

$$\text{Speedup(E)} = \frac{\text{ExTime w/o E}}{\text{ExTime w/ E}} = \frac{\text{Performance w/ E}}{\text{Performance w/o E}}$$



Suppose that enhancement E accelerates a fraction F of the task by a factor S, and the remainder of the task is unaffected, then:

ExTime(E) =

Speedup(E) =

Amdahl's Law

$$\text{ExTime}_{new} = \text{ExTime}_{old} \times \left[(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}} \right]$$

$$\text{Speedup}_{overall} = \frac{\text{ExTime}_{old}}{\text{ExTime}_{new}} = \frac{1}{(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}}}$$

Amdahl's Law

- Floating point instructions improved to run 2X; but only 10% of actual instructions are FP

$ExTime_{new}$ =

$Speedup_{overall}$ =

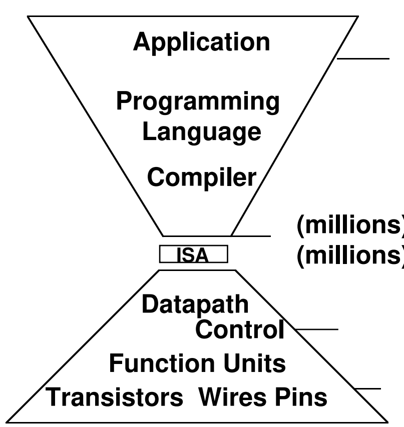
Amdahl's Law

- Floating point instructions improved to run 2X; but only 10% of actual instructions are FP

$$ExTime_{new} = ExTime_{old} \times (0.9 + .1/2) = 0.95 \times ExTime_{old}$$

$$Speedup_{overall} = \frac{1}{0.95} = 1.053$$

Metrics of Performance



Answers per month
Operations per second

(millions) of Instructions per second: MIPS
(millions) of (FP) operations per second: MFLOP/s

Megabytes per second

Cycles per second (clock rate)

Aspects of CPU Performance

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

	Inst Count	CPI	Clock Rate
Program	X		
Compiler	X	(X)	
Inst. Set.	X	X	
Organization		X	X
Technology			X

Cycles Per Instruction

"Average Cycles per Instruction"

CPI = (CPU Time * Clock Rate)/ Instruction Count
= Cycles / Instruction Count

$$\text{CPU time} = \text{CycleTime} * \sum_{i=1}^{n} CPI_i * I_i$$

"Instruction Frequency"

$$CPI = \sum_{i=1}^{n} CPI_i * F_i \quad \text{where } F_i = \frac{I_i}{\text{Instruction Count}}$$

Invest Resources where time is Spent!

Example: Calculating CPI

Base Machine (Reg / Reg)

Op	Freq	Cycles	CPI(i)	(% Time)
ALU	50%	1	.5	(33%)
Load	20%	2	.4	(27%)
Store	10%	2	.2	(13%)
Branch	20%	2	.4	(27%)
			1.5	

Typical Mix

SPEC: System Performance Evaluation Cooperative

- First Round 1989
  - 10 programs yielding a single number ("SPECmarks")
- Second Round 1992
  - SPECInt92 (6 integer programs) and SPECfp92 (14 floating point programs)
    - » Compiler Flags unlimited. March 93 of DEC 4000 Model 610:

```
spice: unix.c:/def=(sysv,has_bcopy,"bcopy(a,b,c)=
       memcpy(b,a,c)"
wave5: /ali=(all,dcom=nat)/ag=a/ur=4/ur=200
nasa7: /norecu/ag=a/ur=4/ur2=200/lc=blas
```

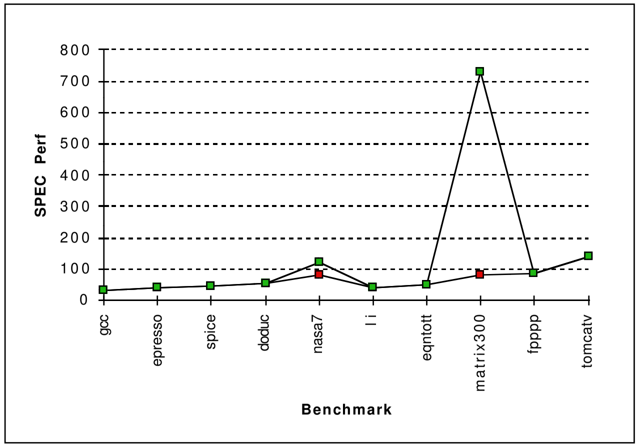
- Third Round 1995
  - new set of programs: "benchmarks useful for 3 years" SPECint95 (8 integer programs) and SPECfp95 (10 floating point)
  - Single flag setting for all programs: SPECint_base95, SPECfp_base95

How to Summarize Performance

- Arithmetic mean (weighted arithmetic mean) tracks execution time: $\sum(T_i)/n$ or $\sum(W_i*T_i)$
- Harmonic mean (weighted harmonic mean) of rates (e.g., MFLOPS) tracks execution time: $n/\sum(1/R_i)$ or $n/\sum(W_i/R_i)$
- Normalized execution time is handy for scaling performance
- But do not take the arithmetic mean of normalized execution time, use the geometric mean ($\prod(R_i)$^1/n)

SPEC First Round

- One program: 99% of time in single line of code
- New front-end compiler could improve dramatically



Impact of Means on SPECmark89 for IBM 550

	Ratio to VAX:		Time:		Weighted Time:	
Program	*Before*	*After*	*Before*	*After*	*Before*	*After*
gcc	30	29	49	51	8.91	9.22
espresso	35	34	65	67	7.64	7.86
spice	47	47	510	510	5.69	5.69
doduc	46	49	41	38	5.81	5.45
nasa7	78	144	258	140	3.43	1.86
li	34	34	183	183	7.86	7.86
eqntott	40	40	28	28	6.68	6.68
matrix300	78	730	58	6	3.43	0.37
fpppp	90	87	34	35	2.97	3.07
tomcatv	33	138	20	19	2.01	1.94
Mean	54	72	124	108	54.42	49.99
	Geometric		*Arithmetic*		*Weighted Arith.*	
	Ratio	1.33	Ratio	1.16	Ratio	1.09

Performance Evaluation

- For better or worse, benchmarks shape a field
- Good products created when have:
  - Good benchmarks
  - Good ways to summarize performance
- Given sales is a function in part of performance relative to competition, investment in improving product as reported by performance summary
- If benchmarks/summary inadequate, then choose between improving product for real programs vs. improving product to get more sales;
  Sales almost always wins!
- Ex. time is the measure of computer performance!
- What about cost?

5 minute Class Break

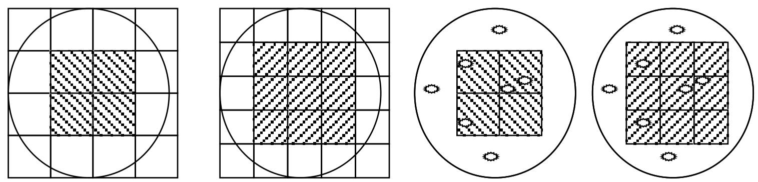
- 80 minutes straight is too long for me to lecture (12:40:00 – 2:00:00):
  - ≈ 1 minute: review last time & motivate this lecture
  - ≈ 20 minute lecture
  - ≈ 3 minutes: discuss class manangement
  - ≈ 25 minutes: lecture
  - 5 minutes: break
  - ≈25 minutes: lecture
  - ≈1 minute: summary of today's important topics

Integrated Circuits Costs

$$\text{IC cost} = \frac{\text{Die cost} + \text{Testing cost} + \text{Packaging cost}}{\text{Final test yield}}$$

$$\text{Die cost} = \frac{\text{Wafer cost}}{\text{Dies per Wafer} * \text{Die yield}}$$

$$\text{Dies per wafer} = \frac{\pi * (\text{Wafer\_diam} / 2)^2}{\text{Die Area}} - \frac{\pi * \text{Wafer\_diam}}{\sqrt{2 * \text{Die Area}}} - \text{Test dies}$$



$$\text{Die Yield} = \text{Wafer yield} * \left\{1 + \frac{\text{Defects\_per\_unit\_area} * \text{Die\_Area}}{\alpha}\right\}^{-\alpha}$$

Die Cost goes roughly with die area^4

Real World Examples

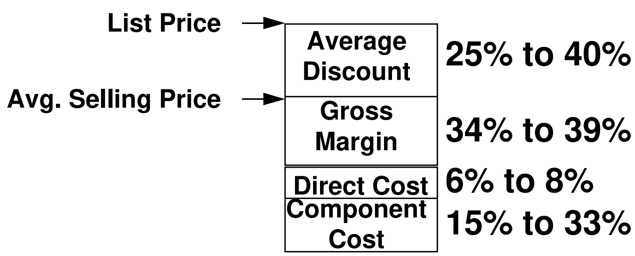
Chip	Metal layers	Line width	Wafer cost	Defect /cm^2	Area mm^2	Dies/ wafer	Yield	Die Cost
386DX	2	0.90	$900	1.0	43	360	71%	$4
486DX2	3	0.80	$1200	1.0	81	181	54%	$12
PowerPC 601	4	0.80	$1700	1.3	121	115	28%	$53
HP PA 7100	3	0.80	$1300	1.0	196	66	27%	$73
DEC Alpha	3	0.70	$1500	1.2	234	53	19%	$149
SuperSPARC	3	0.70	$1700	1.6	256	48	13%	$272
Pentium	3	0.80	$1500	1.5	296	40	9%	$417

– From "Estimating IC Manufacturing Costs," by Linley Gwennap, *Microprocessor Report*, August 2, 1993, p. 15

Cost/Performance

What is Relationship of Cost to Price?

- **Component Costs**
- **Direct Costs** (add 25% to 40%) recurring costs: labor, purchasing, scrap, warranty
- **Gross Margin** (add 82% to 186%) nonrecurring costs: R&D, marketing, sales, equipment maintenance, rental, financing cost, pretax profits, taxes
- **Average Discount** to get List Price (add 33% to 66%): volume discounts and/or retailer markup



Chip Prices (August 1993)

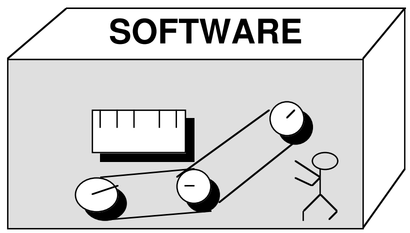
- Assume purchase 10,000 units

Chip	Area mm^2	Mfg. cost	Price	Multi-plier	Comment
386DX	43	$9	$31	3.4	Intense Competition
486DX2	81	$35	$245	7.0	No Competition
PowerPC 601	121	$77	$280	3.6	
DEC Alpha	234	$202	$1231	6.1	Recoup R&D?
Pentium	296	$473	$965	2.0	Early in shipments

Computer Architecture Is …

the attributes of a [computing] system as seen by the programmer, i.e., the conceptual structure and functional behavior, as distinct from the organization of the data flows and controls the logic design, and the physical implementation.

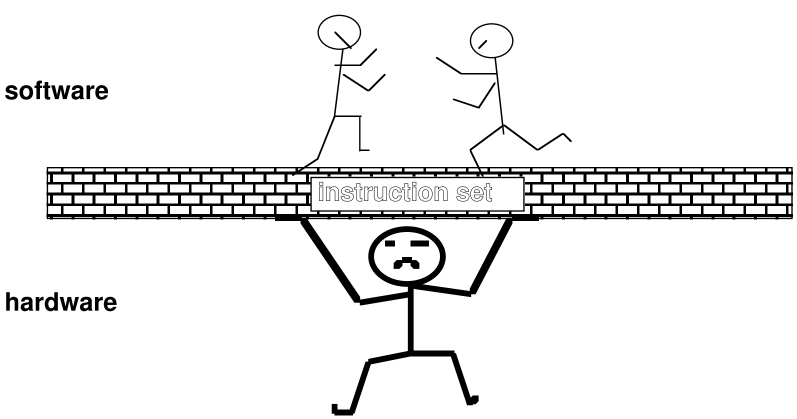
Amdahl, Blaaw, and Brooks, 1964



Computer Architecture’s Changing Definition

- **1950s to 1960s: Computer Architecture Course Computer Arithmetic**
- **1970s to mid 1980s: Computer Architecture Course Instruction Set Design, especially ISA appropriate for compilers**
- **1990s: Computer Architecture Course Design of CPU, memory system, I/O system, Multiprocessors**

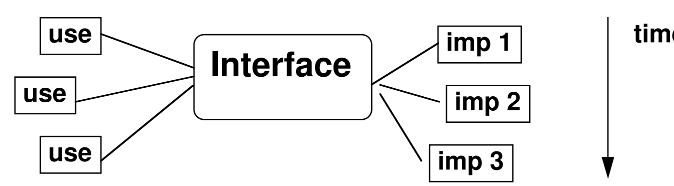
Instruction Set Architecture (ISA)



Interface Design

A good interface:

- **Lasts through many implementations (portability, compatability)**
- **Is used in many differeny ways (generality)**
- **Provides convenient functionality to higher levels**
- **Permits an efficient implementation at lower levels**



Evolution of Instruction Sets

Single Accumulator (EDSAC 1950)

Accumulator + Index Registers
(Manchester Mark I, IBM 700 series 1953)

Separation of Programming Model
from Implementation

High-level Language Based
(B5000 1963)

Concept of a Family
(IBM 360 1964)

General Purpose Register Machines

Complex Instruction Sets
(Vax, Intel 432 1977-80)

Load/Store Architecture
(CDC 6600, Cray 1 1963-76)

RISC
(Mips,Sparc,HP-PA,IBM RS6000, . . .1987)

Evolution of Instruction Sets

- Major advances in computer architecture are typically associated with landmark instruction set designs
  - Ex: Stack vs GPR (System 360)
- Design decisions must take into account:
  - technology
  - machine organization
  - programming langauges
  - compiler technology
  - operating systems
- And they in turn influence these

A "Typical" RISC

- 32-bit fixed format instruction (3 formats)
- 32 32-bit GPR (R0 contains zero, DP take pair)
- 3-address, reg-reg arithmetic instruction
- Single address mode for load/store: base + displacement
  - no indirection
- Simple branch conditions
- Delayed branch

see: SPARC, MIPS, HP PA-Risc, DEC Alpha, IBM PowerPC, CDC 6600, CDC 7600, Cray-1, Cray-2, Cray-3

Example: MIPS

Register-Register

31 26	25 21	20 16	15 11	10 6	5 0
Op	Rs1	Rs2	Rd		Opx

Register-Immediate

31 26	25 21	20 16	15 0
Op	Rs1	Rd	immediate

Branch

31 26	25 21	20 16	15 0
Op	Rs1	Rs2/Opx	immediate

Jump / Call

31 26	25 0
Op	target

Summary, #1

- **Designing to Last through Trends**

	Capacity	Speed
Logic	2x in 3 years	2x in 3 years
DRAM	4x in 3 years	1.4x in 10 years
Disk	4x in 3 years	1.4x in 10 years

- **Time to run the task**
  - Execution time, response time, latency
- **Tasks per day, hour, week, sec, ns, …**
  - Throughput, bandwidth
- **"X is n times faster than Y" means**

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

Summary, #2

- **Amdahl's Law:**

$$\text{Speedup}_{overall} = \frac{\text{ExTime}_{old}}{\text{ExTime}_{new}} = \frac{1}{(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}}}$$

- **CPI Law:**

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

- **Execution time is the REAL measure of computer performance!**
- **Good products created when have:**
  - Good benchmarks
  - Good ways to summarize performance
- **Die Cost goes roughly with die area^4**

Summary, #3: Price vs. Cost