Lecture 1: Course Introduction, Technology Trends, Performance

Professor Alvin R. Lebeck
Computer Science 220
Fall 2001

Administrative

- **Office Hours**
  - Office: D304 LSRC
  - Hours: Mon 10:00-11:00 Thurs 2:00-3:00 or by appointment (email)
  - email: alvy@cs.duke.edu
  - Phone: 660-6551
- **Teaching Assistant**
  - Fareed Zaffar
  - Office: D125 LSRC
  - Hours: Tuesday 10:00-11:00, Wednesday 1:00-2:00
  - email: fareed@cs.duke.edu
  - Phone: 660-6576

Administrative (Grading)

- **30% Homeworks**
  - 6 Homeworks
  - 5 points per day late, for first 10 days
  - Always do the homework (better late than never)
- **30% Examinations (Midterm + Final)**
- **30% Research Project (work in pairs)**
- **10% Class Participation**
- **This course requires hard work.**

Administrative (Continued)

- **Midterm Exam: In class (75 min) Closed book**
- **Final Exam: (3 hours) closed book**
- **This is a "Quals" Course.**
  - Quals pass based on Midterm and Final exams only

Administrative (Continued)

- **Course Web Page**
  - http://www.cs.duke.edu/courses/fall01/cps220
  - Lectures posted there after class (pdf)
  - Homework posted there
- **Course News Group**
  - duke.cs.cps220
  - Use it to 1) read announcements/comments on class or homework, 2) ask questions (help), 3) communicate with each other
- **Need Duke CS account**
  - Duke ID, ACPUB account name (see HW #0)

SPIDER: Systems Seminar

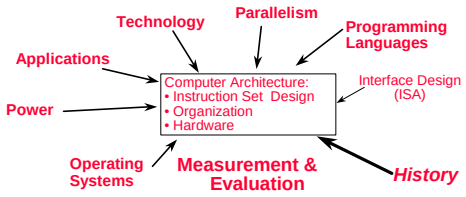
- **Systems & Architecture Seminar**
  - Wednesdays 3:45-5:00 in D344
  - duke.cs.os-research (spider newsgroup)
- **Presentations on current work**
  - Practice talks for conferences
  - Discussion on recent papers
  - Your own research
- **Why you should go?**
  - If you want to work in Systems/Architecture…
  - Good time to practice public speaking in front of friendly crowd
  - Learn about current topics

Assignment

- Homework #0 (Background, due Thursday)
- Read Chapters 1 & 2

CPS 220 Course Focus

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



Related Courses

Prerequisites

- CPS 104: Basic Machine Organization
- CPS 110: Basic Operating System Functions

- This course: focus on why, analysis, evaluation
  - Cost/performance
  - Power budget

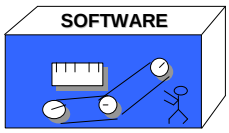
Follow on Courses

- CPS 221: Advanced Computer Architecture II
  - Parallel computer architecture

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

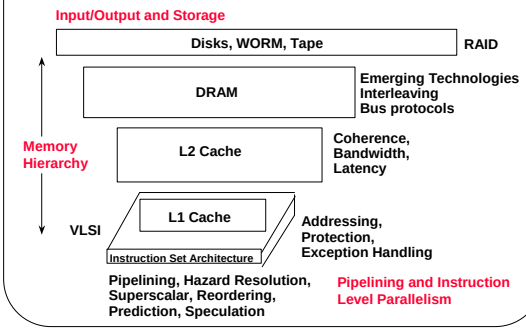


Topic Coverage

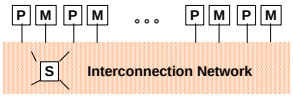
Textbook: Hennessy and Patterson, *Computer Architecture: A Quantitative Approach*, 2nd Ed., 1995.

- Fundamentals of Computer Architecture (Chapter 1)
- Instruction Set Architecture (Chapter 2, Appendix C&D)
- Pipelining (Chapter 3)
- Advanced Pipelining and ILP (Chapter 4)
- Memory Hierarchy (Chapter 5)
- Input/Output and Storage (Chapter 6)
- Networks and Interconnection Technology (Chapter 7)
- Multiprocessors (Chapter 8)
- Vectors (Apendix)
- New Architectures/trends (papers)
- Power (papers)

Computer Architecture Topics



Computer Architecture Topics (CPS 221)



Processor-Memory-Switch

Multiprocessors
Networks and Interconnections

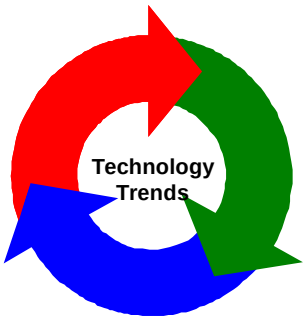
Shared Memory, Message Passing, Data Parallel

Network Interfaces

Topologies, Routing, Bandwidth, Latency, Reliability

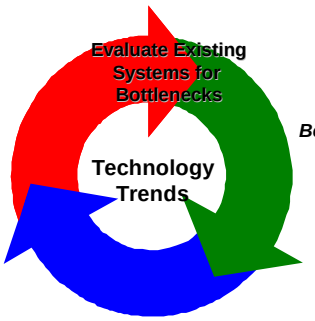
© Alvin R. Lebeck 2001

Computer Engineering Methodology



© Alvin R. Lebeck 2001

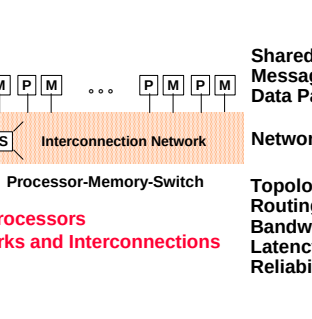
Computer Engineering Methodology



Benchmarks

© Alvin R. Lebeck 2001

Computer Engineering Methodology



Benchmarks

Workloads

Simulate New Designs and Organizations

© Alvin R. Lebeck 2001

Computer Engineering Methodology

Benchmarks

Workloads

Simulate New Designs and Organizations

© Alvin R. Lebeck 2001

Context for Designing New Architectures

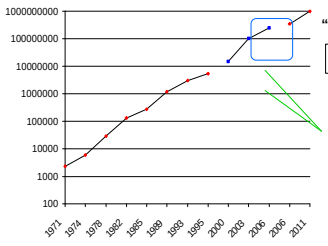
- Application Area
  - Special Purpose (e.g., DSP) / General Purpose
  - Scientific (FP intensive) / Commercial (Mainframe)
  - Portable (Power matters)
- Level of Software Compatibility
  - Object Code/Binary Compatible (cost HW vs. SW; IBM S/360)
  - Assembly Language (dream to be different from binary)
  - Programming Language; Why not?

© Alvin R. Lebeck 2001

Context for Designing New Architectures

- **OS Requirements for General Purpose Apps**
  - Size of Address Space
  - Memory Management/Protection
  - Context Switch
  - Interrupts and Traps
  - Communication
- **Standards: Innovation vs. Competition**
  - IEEE 754 Floating Point
  - I/O Bus
  - Networks
  - Operating Systems / Programming Languages ...

Technology Trends: Microprocessor Capacity



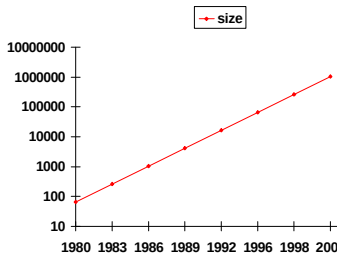
"Graduation Window"

Intel
Digital

Pentium Pro: 5.5 million
Sparc Ultra: 5.2 million
PowerPC 620: 6.9 million
Alpha 21164: 9.3 million
Alpha 21264: 15 million
Pentium III: 28 million
Pentium 4: 42 million
Alpha 21364: 100 million
Alpha 21464: 250 million

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

DRAM Capacity (single chip)

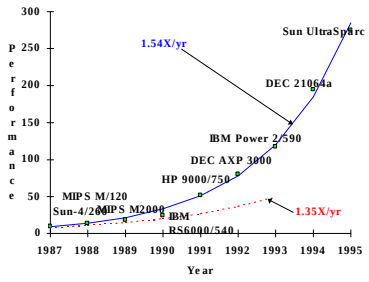


year	size	cyc time
1980	64 Kb	250 ns
1983	256 Kb	220 ns
1986	1 Mb	190 ns
1989	4 Mb	165 ns
1992	16 Mb	145 ns
1996	64Mb	104 ns
1998	256Mb	
2002	1Gb	

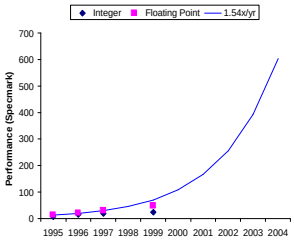
Technology Trends (Summary)

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

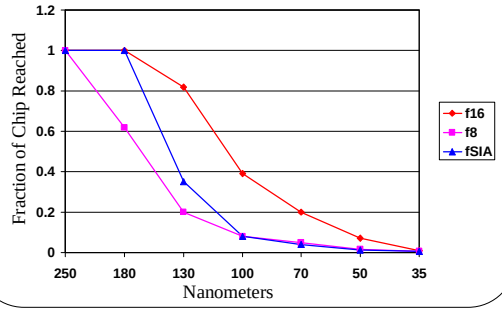
Processor Performance



Alpha SPECint and SPECfp

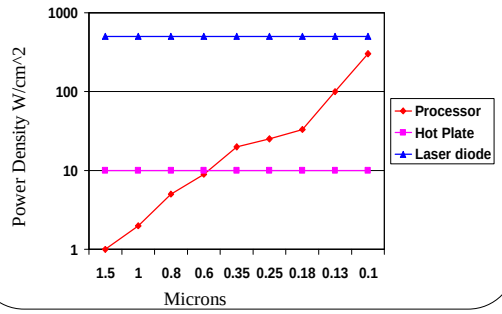


Chip Area Reachable in One Clock Cycle



© Alvin R. Lebeck 2001

Power Density



© Alvin R. Lebeck 2001

Processor Perspective

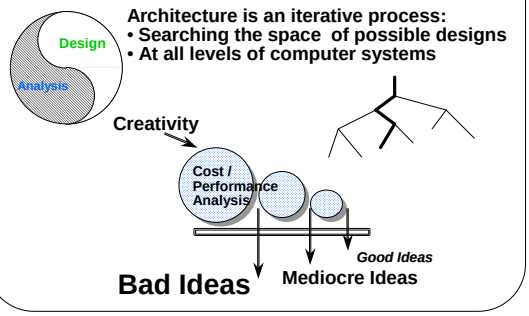
- Putting performance growth in perspective:

	Pentium-III Personal Comp.	Cray YMP Supercomputer
Year	1998	1988
MIPS	> 400 MIPS	< 50 MIPS
Linpack	140 MFLOPS	160 MFLOPS
Cost	$3,000	$1M ($1.6M in 1994$)
Clock	400 MHz	167 MHz
Cache	512 KB	0.25 KB
Memory	128 MB	256 MB

- 1988 supercomputer in 1998 personal computer!

© Alvin R. Lebeck 2001

Measurement and Evaluation



© Alvin R. Lebeck 2001

Measurement Tools

- How do I evaluate an idea?
- Performance, Cost, Die Area, Power Estimation
- Benchmarks, Traces, Mixes
- Simulation (many levels)
  - ISA, RT, Gate, Circuit
- Queuing Theory
- Rules of Thumb
- Fundamental Laws

- Question: What is "better" Boeing 747 or Concorde?

© Alvin R. Lebeck 2001

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 Concorde	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

© Alvin R. Lebeck 2001

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

Performance Terminology

"X is n% faster than Y" means:

$$\frac{\text{ExTime(Y)}}{\text{ExTime(X)}} = \frac{\text{Performance(X)}}{\text{Performance(Y)}} = 1 + \frac{n}{100}$$

$$n = \frac{100(\text{Performance(X)} - \text{Performance(Y)})}{\text{Performance(Y)}}$$

Example: Y takes 15 seconds to complete a task, X takes 10 seconds. What % faster is X?

Example

$$\frac{\text{ExTime(Y)}}{\text{ExTime(X)}} = \frac{15}{10} = \frac{1.5}{1.0} = \frac{\text{Performance (X)}}{\text{Performance (Y)}}$$

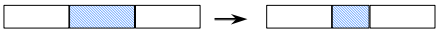
$$n = \frac{100\ (1.5\ -\ 1.0)}{1.0}$$

$$n = 50\%$$

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 instruction execution time is FP

ExTime_{new} =

$\text{Speedup}_{overall}$ =

Amdahl's Law

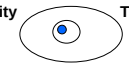
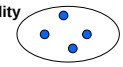
- Floating point instructions improved to run 2X; but only 10% of actual instruction execution time is FP

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

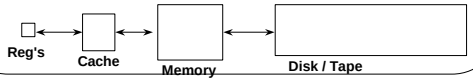
$$Speedup_{overall} = \frac{1}{0.95} = 1.053$$

Corollary: Make The Common Case Fast

- All instructions require an instruction fetch, only a fraction require a data fetch/store.
  - Optimize instruction access over data access
- Programs exhibit *locality*

Spatial Locality  Temporal Locality 

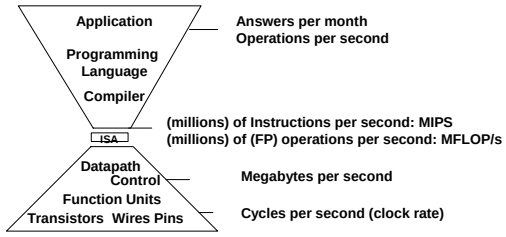
- Access to small memories is faster
  - Provide a *storage hierarchy* such that the most frequent accesses are to the smallest (closest) memories.



Occam's Toothbrush

- The simple case is usually the most frequent and the easiest to optimize!
- Do simple, fast things in hardware and be sure the rest can be handled correctly in software

Metrics of Performance



Aspects of CPU Performance

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

	Instr. Cnt	CPI	Clock Rate
Program			
Compiler			
Instr. Set			
Organization			
Technology			

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

Marketing Metrics

$$MIPS = \frac{\text{Instruction Count}}{\text{Time}} \times 10^6 = \frac{\text{Clock Rate}}{\text{CPI}} \times 10^6$$

- Machines with different instruction sets ?
- Programs with different instruction mixes ?
  - Dynamic frequency of instructions
- Uncorrelated with performance

$$MFLOPS = \frac{\text{FP Operations}}{\text{Time}} \times 10^6$$

- Machine dependent
- Often not where time is spent

Normalized:	
add,sub,compare,mult	1
divide, sqrt	4
exp, sin, . . .	8

Cycles Per Instruction

"Average Cycles Per Instruction"

$$CPI = \frac{\text{CPU time} \times \text{Clock Rate}}{\text{Instruction Count}} = \frac{\text{Cycles}}{\text{Instruction Count}}$$

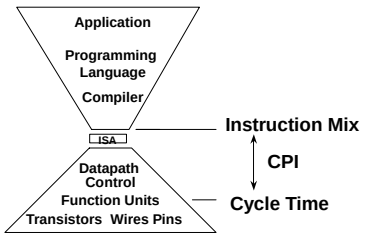
$$\text{CPU time} = \text{Cycle Time} \times \sum_{i=1}^{n} CPI_i \times I_i$$

"Instruction Frequency"

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

Invest Resources where time is Spent!

Organizational Trade-offs



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

Example

Add register / memory operations to traditional RISC:
- One source operand in memory
- One source operand in register
- Cycle count of 2

Branch cycle count to increase to 3.

What fraction of the loads must be eliminated for this to pay off?

Base Machine (Reg / Reg)

Op	Freq	Cycles
ALU	50%	1
Load	20%	2
Store	10%	2
Branch	20%	2

Next Time

- Benchmarks
- Performance Metrics
- Cost
- Instruction Set Architectures

TODO

- Read Chapters 1 & 2
- Do Homework #0