Review: (Write on the board before class)
Quantitative principles of computer design
1) Make the common case fast; optimize for the common case
2) Amdahl's Law: $\textit{Speed-up} = \dfrac{1}{(1-\text{Fraction}_{\text{enhanced}}) + \dfrac{\text{Fraction}_{\text{enhanced}}}{\text{Speed-up}_{\text{enhanced}}}}$
3) Principle of Locality: 90/10 rule; temporal & spatial

1) + 3) => memory hierarchy: CPU register,cache,main memory,disks
Today complete Ch1 & Start Ch 2
Definitions:
Performance: throughput/bandwidth vs. response time/latency/execution time **Ask class for examples?**

What is faster? "X is n% faster than Y"=> 1 + n/100 = Perfx/Perfy = Timey/Timex (slower than confusing; try not to use in this class)

Job of a computer designer
1) functional requirements; review figure 1.4
2) balancing hardware and software; only advantage of hardware is speed
software wins ease of design, replication costs, ease of upgrade
=> complixity => design time
3) designing to last through trends
Logic: capacity & speed 2X / 3yrs;
DRAM capacity 4X every 3 years, cycle time slowly (1/3 decrease in 10 yrs); Disks: capacity 2X every 3 years, access time slowly slowly (1/3 decrease in 10)

Address space: 1.5 to 2X in program size, using1/2 to 1 bit of address/yr
assembly lang => HLL; compiler technology advancing with time
programming langauges: every 10 years

Balance: Case/Amdahl rule of thumb: 1MIPS=1MB memory =1Mb I/O

Pitfalls and Fallacies: why include
Pitfall: comparing performance working machine today vs when your machine works (*ignoring progress when projecting relative performance*)
25% improvement in performance per year
1 year late, better add 25% to performance? 1.25^3 = 2X/3 yrs (1.5^3 =3.4)
(reason projects are cancelled)
Fallacy: HW is always faster than SW
If perform exactly same algorithm
frequently instructions are very general to handle all cases
MVC: 50% move a single byte; LOAD;STORE 2x as fast as MVC on 370
Class know of examples?

Class stuff:

Lecture style
Distrubte Survey again
No discussion sections; individual times on projects instead
First home work assignment: 1.1 through 1.6, and 1.8 **and 2.1-2.7/ in 533 Evans by Noon(too many students)**

Tapes available 1 or 2 days after lecture; in 205? McLaughlin
Book not ready: first portion available at Copymats on Bancroft for about $15
Books delivered to class when avilable (about march 1)
Will take checks in class for difference between book and copy
Mention quizzes, term projects, Grades: about 30%/category, 10% participation

Continue Lecture
Review: Job of computer designer
Basic Function, Lasting through trends, balance HW & software

Chapter 2: Performance and Cost, the fundamental evalutation of computers
Historically costs ignored, because hard to describe
Embarassing, but performance summary is worse than automobiles
supercomputers characterized as MFLOPS rate or MIPS rate
single number of performance
Targets: high performance (Cray YMP), low cost (PC clone), cost/performance (workstation);
Be sure to know target & evaluate as target; comparing cost/perf of Cray YMP or IBM PC clone may not matter; but if workstation is as fast as Cray or as cheap as PC, then cost and performance matter

Peformance
Time command of Unix: 90u, 10s, 2:30, 67%
user CPU time, system CPU time, elapsed time, % user of elapsed
Terms: elapsed time = reponse time on an unloaded system,
Ignore system time (it varies) or add to user CPU time; usually quote CPU time

Time in clocks

$$\text{CPU time} = \text{CPU clock cycles for a program} * \text{Clock cycle time}$$

$$\text{CPI} = \frac{\text{CPU clock cycles for a program}}{\text{Instruction count}}$$

$$\text{CPU time} = \text{Instruction count} * \text{CPI} * \text{Clock cycle time}$$

The fundamental equation; if remember nothing else from class remember Amdahl's Law and CPU time formula;

Only thing that matters is time; making any of these terms smaller by itself does not mean faster

Clipper: 30 ns clock vs. 60 ns clock of Sun 4/280; IC about the same, yet 4/280 takes less time for programs
CDC 6600 vs B5500: IC 50% higher, but 550% faster
(also size != speed; code size 220% larger)

80386 vs. 68030: clock rate is 25MHx vs 33 MHz; CPI is 4 vs. 6; what must relative IC be for equal performance? (about 1.13)
do in class?

Time is measure of machines, not code size, not CPI, not cycle time, not IC!

Go through example using CPU time equation

Summarize: Job of computer designer: functions, trends, HW v. SW, perf eq.

Notes to lecturer: Amdahl's Law + locality worked well, showing how to use Amdahl's Law.

Functional Requirements

Application Area
- Special Purpose
- General Purpose
- Scientific
- Commercial

Level of Software Compatibility
- Programming Language
- Assembly Language
- Object Code or Binary Compatible

Operating System Requirements
- Size of address space
- Memory management
- Protection
- Context Switch
- Interrupts and Traps

Standards
- Floating Point
- I/O Bus
- Operating Systems
- Networks
- Programming Languages