Computer Architecture (DAT 105)

Credit points: 7.5 EHC

Course Examiner and Lecturer: Per Stenstrom

Teaching Assistant:
Muhammad Waqar Azhar
Fazeleh Hoseini
Evangelos Vasilakis

Department of Computer Science & Engineering
Chalmers University of Technology

What is Computer Architecture All About?

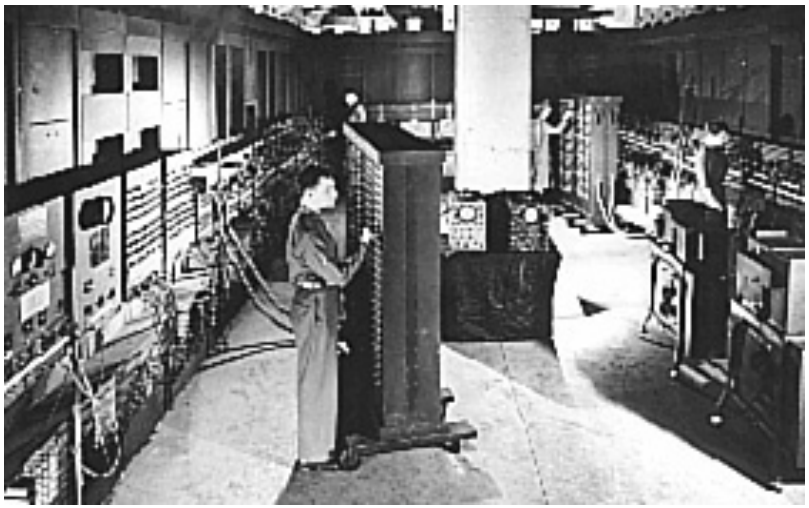
A Fascinating Success Story

Intel i9 (2017)
~ 30 billion ops/s,
~ 7 billion transistors

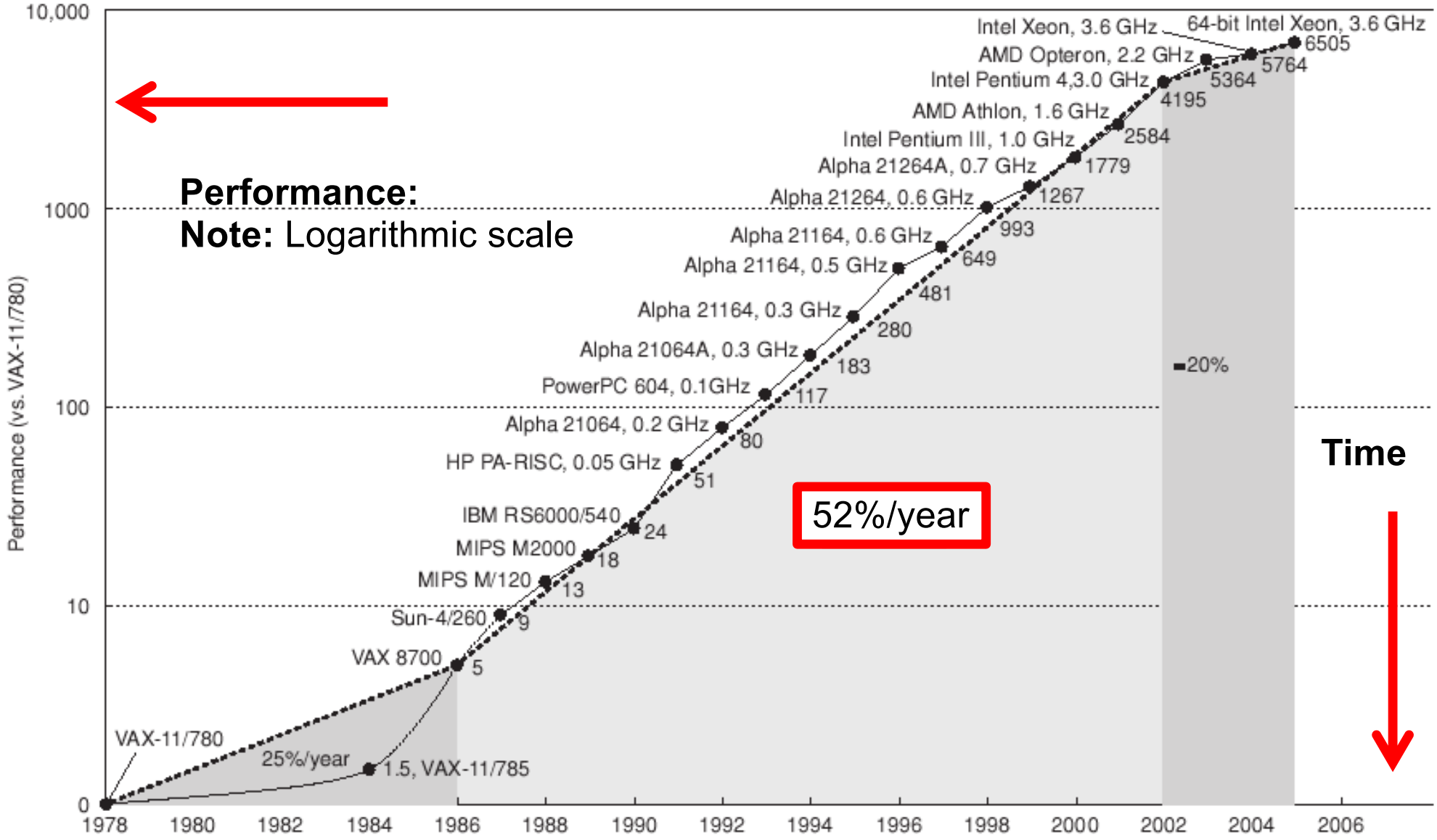
~ 3.7 cm

ENIAC (1946):
~ 1000 ops/s 18000 vacuum tubes

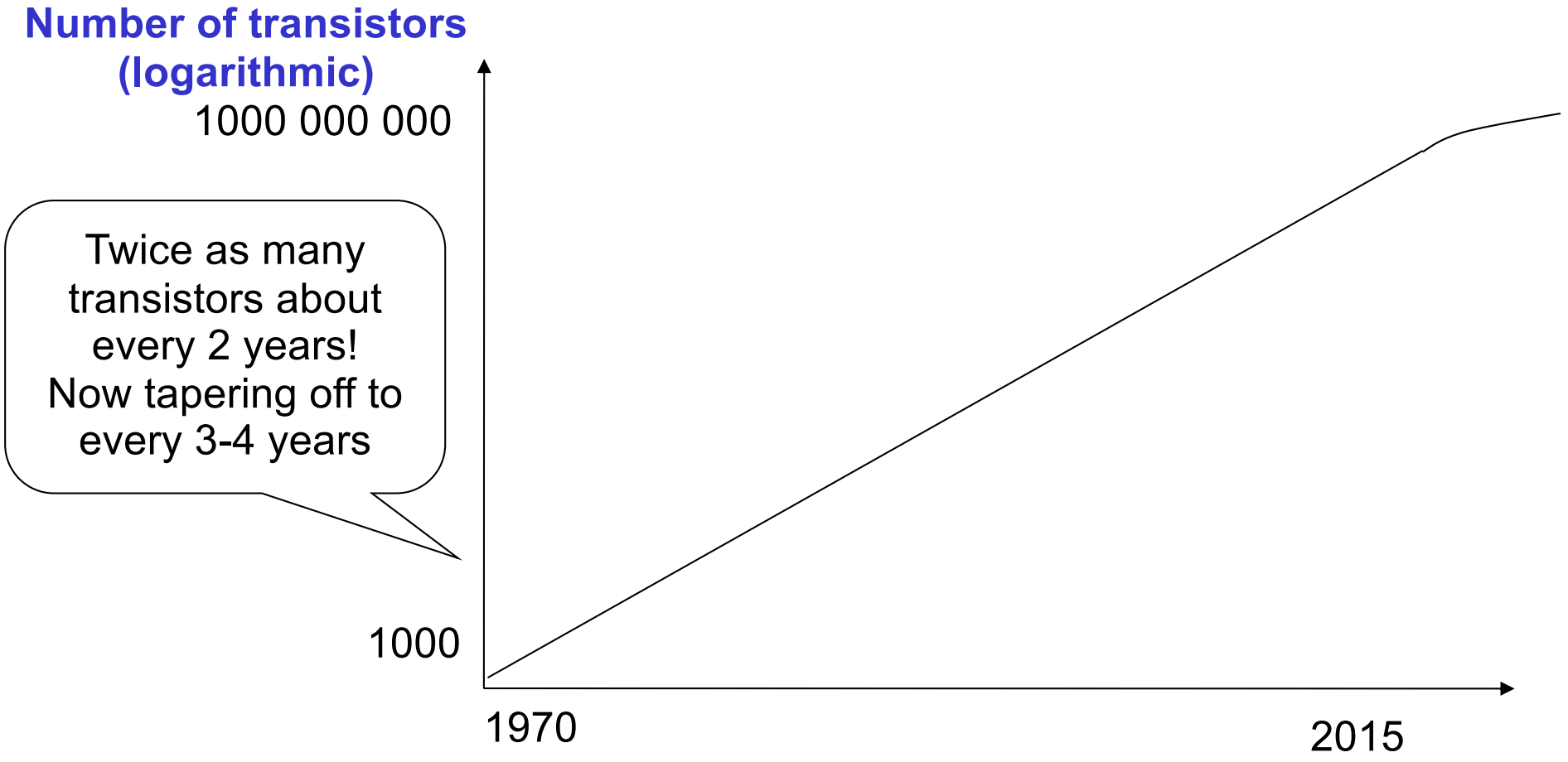
~ 10 m



Millions of times faster, smaller, and more cost effective

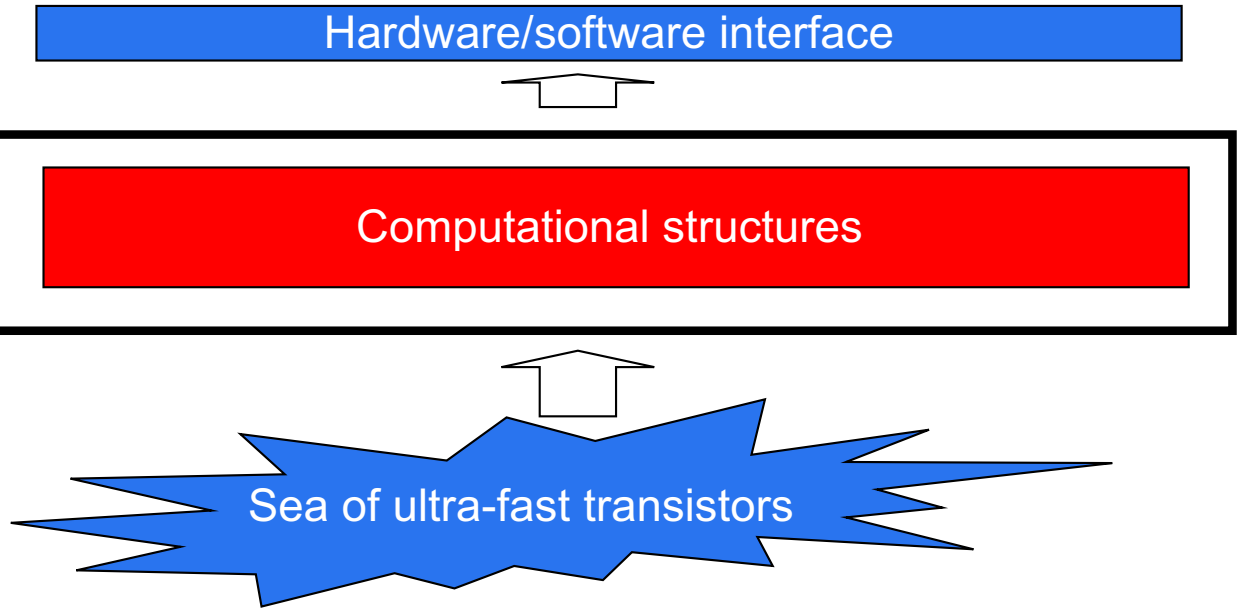


Moore's Law



- How can we tame all these transistors to compute?

Computer Architecture



The engineering discipline of computer design

The hardware/software interface

- Instruction Set Architecture (ISA)
- Computer organization
- Hardware design

Quiz 1.1

Compare ENIAC with Intel Core I9. Which of the following statements are true

A. Intel I9 is at least 20 million times faster than ENIAC
B. Intel I9 is at most 2 million times faster than ENIAC
C. Core I9 contains at least 300 thousand more switching elements than ENIAC
D. Core I9 is at least 70000 times smaller than ENIAC
E. Core I9 is at least 1 million times smaller than ENIAC

How is the Course Organized?

Learning Objectives

After the course you should:

- **master** fundamental **concepts and terminology**
- **understand** **design principles of processors**
- **understand** **design principles of memory hierarchies**
- **understand** **design principles of multicore microprocessors**
- **be able to use** **modeling methods** to assess the impact design alternatives have on **performance/energy**

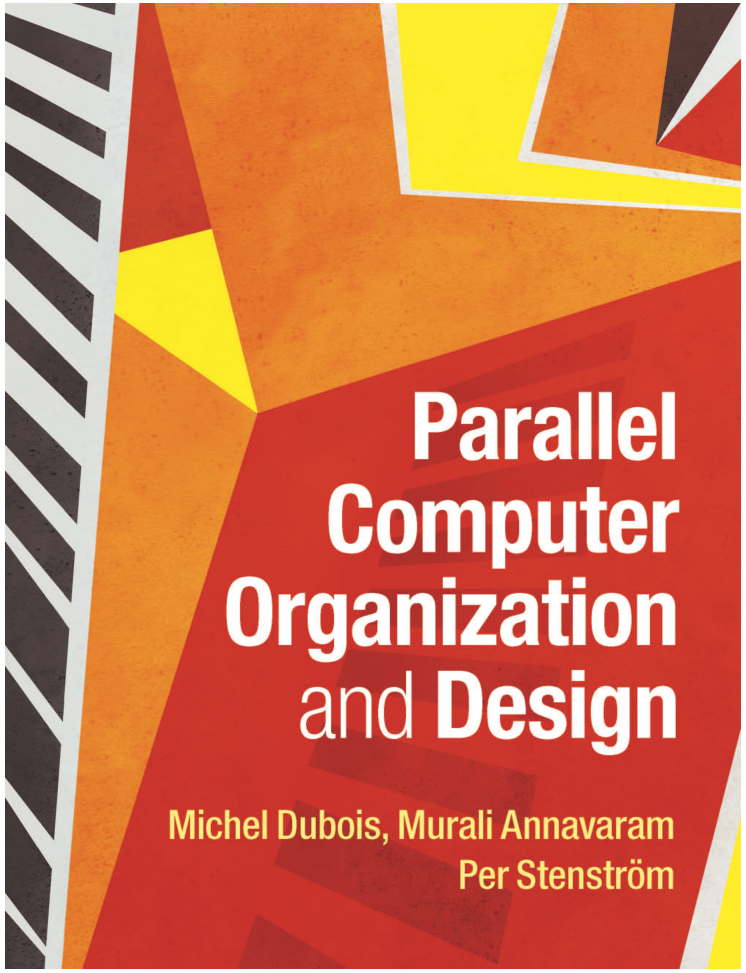
Course Organization

- 7 lectures & 2 optional crash-course lectures
- 2 real-stuff presentations
- 7 problem solution sessions
- 3 laboratory sessions/assignments (reports)

Lectures: Flipped-classroom teaching. Interaction. Active learning!

Preparation: Watch available youtube lectures before class (starting next week)!!!!

Textbook



- Book is available for sale at Cremona
- Only selected chapters of the book
- Book is also used in the other courses in the Computer Architecture track.

Course Contents – Lectures

1. **Introduction:** Components of a computer, technology trends, performance principles and metrics

   Opt 1 (Basics of modern processor design: Pipelining concepts)

   Do you know what a *RAW hazard* is? *Delayed branch*? *Stalling*?

   Opt 2 (Basics of memory hierarchies: Caches & Interconnects)

   Do you know what a *memory hierarchy is*? *Set associativity*? Miss *penalty*?

2-4. **Techniques to uncover ILP:** Overcoming limitations of instruction ordering, name dependencies, and control flow: Static instruction scheduling, out-of-order execution, branch prediction, speculative execution

5. **Memory hierarchy concepts.** Cache and virtual memory. Inclusion, lockup-free caches, prefetching etc.
6. **Techniques to uncover TLP:** Thread-level parallelism, multicore-multithreaded architectures, cache coherence
7. **Very Large Instruction Word (VLIW) computers.** Static techniques for uncovering ILP, trace scheduling, predicated execution.

Course Schedule

TimeEdit

Course information:

- At Canvas
- Course Schedule (Under Syllabus tab)
- Course material (under Files tab)
  - Video lectures
  - Flipped classroom material
  - Exercises
  - Laboratory PMs

Keep an eye on updates at Canvas regularly!!!

- Real-time information about course logistics

Examination

- **Laboratory project assignment**
  - Brief written report required for each assignment
- **Quizzes at each lecture**
  - Based on youtube lectures
  - 3 correct => 4 bonus points for the exam (for higher grades only)
- **Real-stuff studies**
  - Learn how principles are used in commercial machines
- **Written exam**
  - Typically on concepts and problem solving. Examples of old exams will be distributed. **Date: October 28, 2019.**

Course Evaluation

- **We want to improve the course continuously**
  - Does the course start at the right level?
  - How good is the teaching material (textbook, problems, lab manuals)?
  - How well do the instructors perform?
  - What is the pace in the course?

Please give us feedback; we use it to improve the course – in real-time, and for coming generations of students

Course representatives:

- MPSOF johaaro@student.chalmers.se Johan Aronsson
- UTBYTE sebastianfrank95@gmail.com Sebastian Frank
- TKDAT grahn.lovisa@gmail.com Lovisa Gran
- MPHPC romanroibu@gmail.com Roman Roibu
- MPHPC theocharistr@gmail.com Theocharis Triantafyllidis

Lecture 1

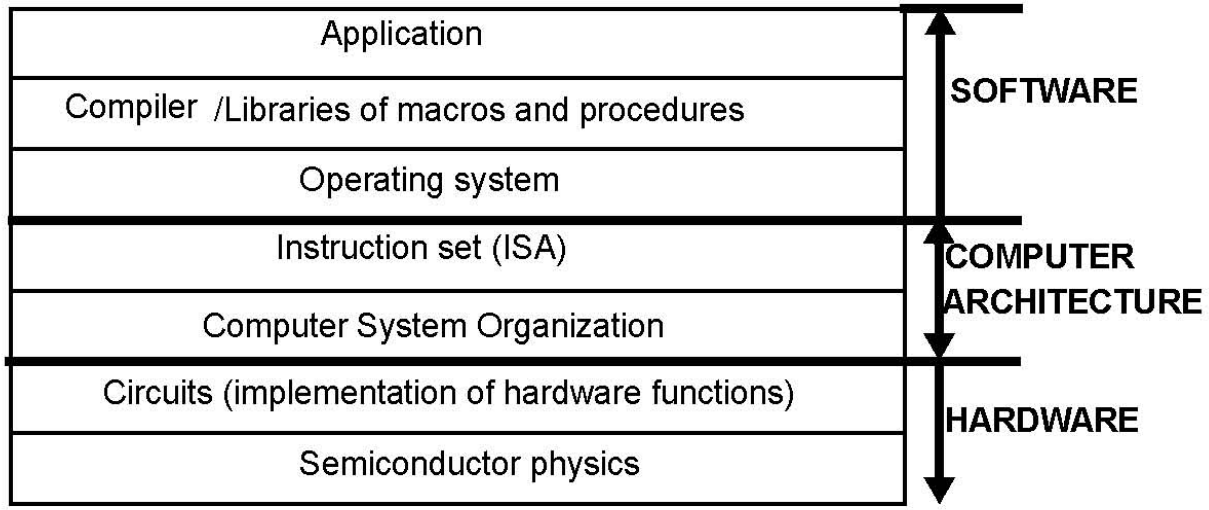
Fundamentals of Computer Architecture

Components of a Computer, Trends and Parallelism (Ch 1.1 – 1.3, 1.5)

What is Computer Architecture?

- **Old definition:** Instruction Set Architecture (ISA)
- **Today's definition:** Hardware organization of computers (including ISA)

The Transformation Hierarchy:



The role of a computer architect: To make design trade-offs across the HW/SW interface to meet functional, performance and cost requirements

A PC Computer Organization

The organization of a modern PC:

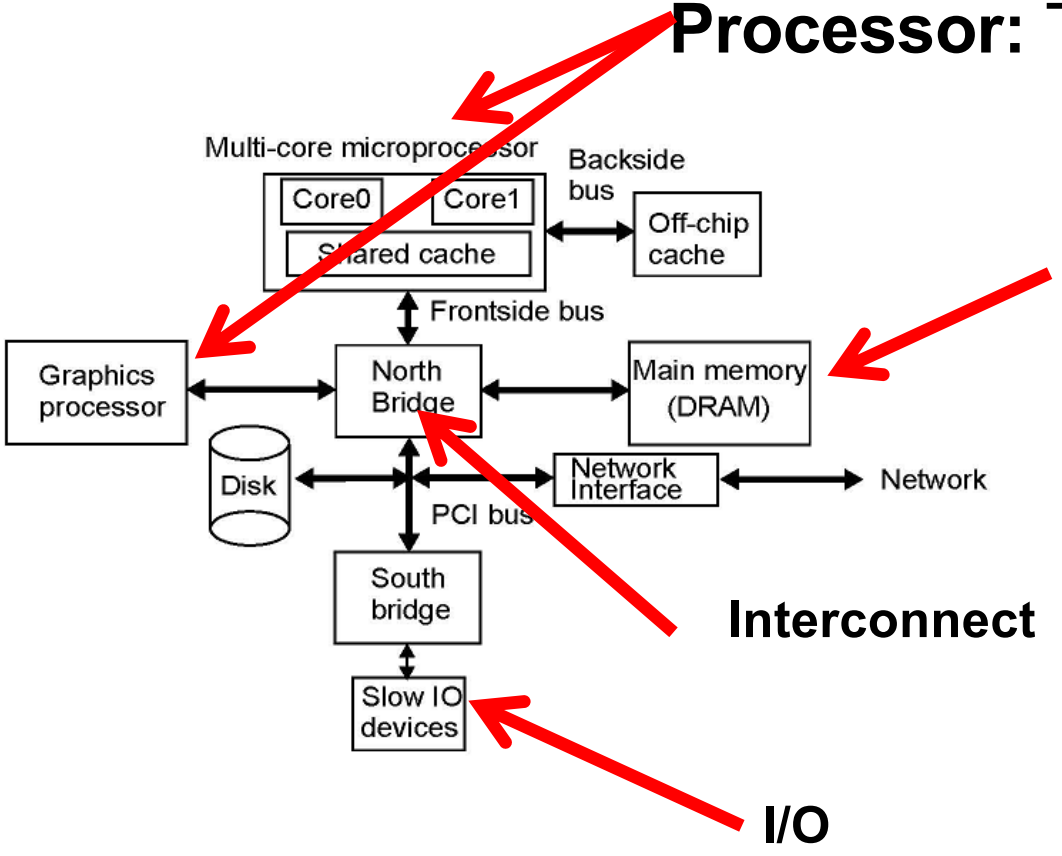


Fig 1.3
Dubois et al.
Parallel Computer
Organization and Design

A High-end Computer Organization

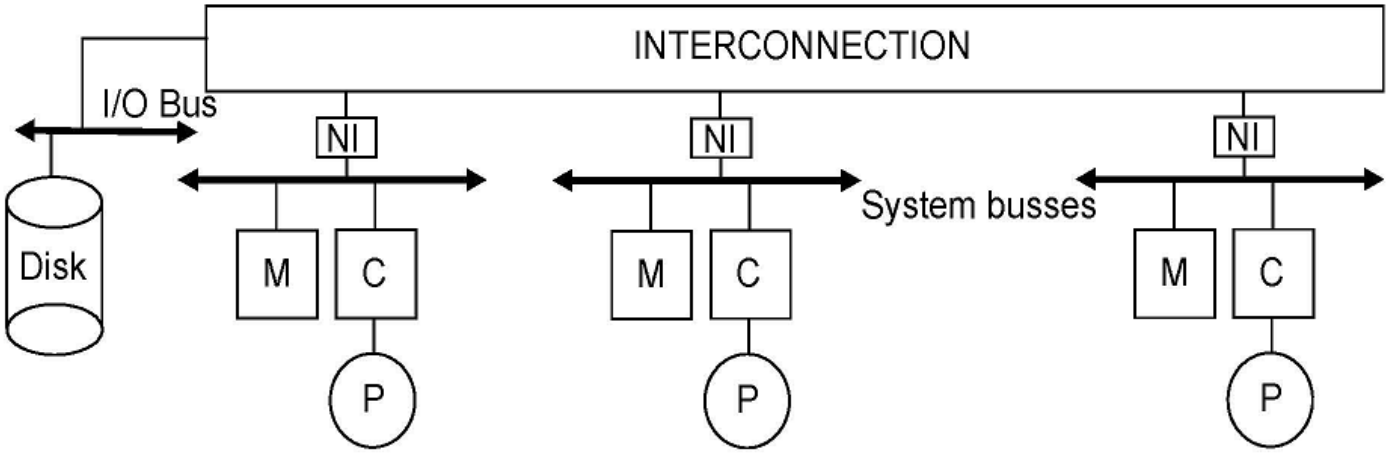


Fig 1.4
Dubois et al.
Parallel Computer
Organization and Design

Main components:

Processor (P), memory system (M and C), I/O, and networks

Processor Architecture 1(2)

- **Clock rates of microprocessors have increased exponentially**

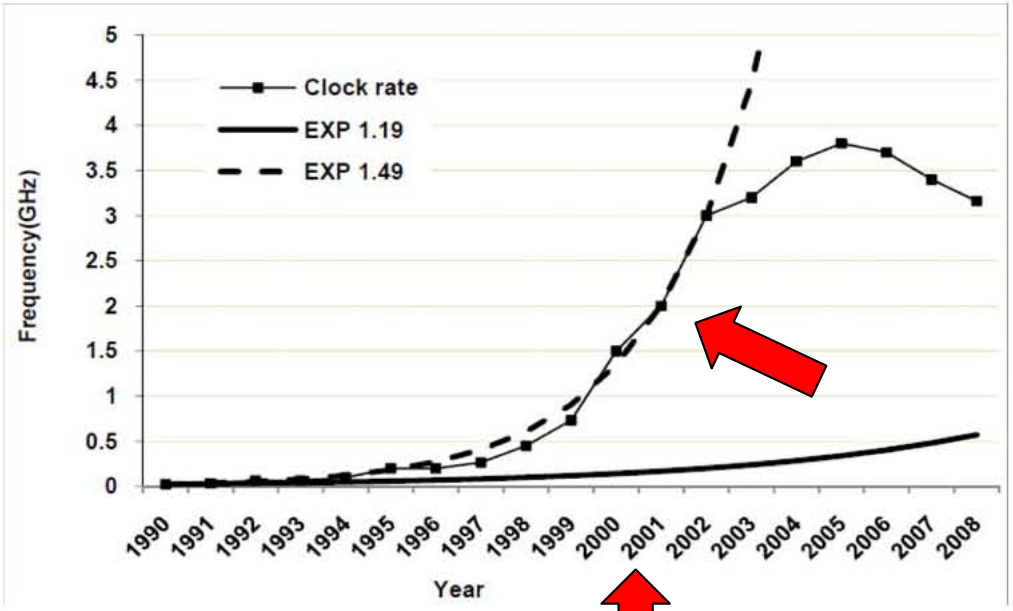


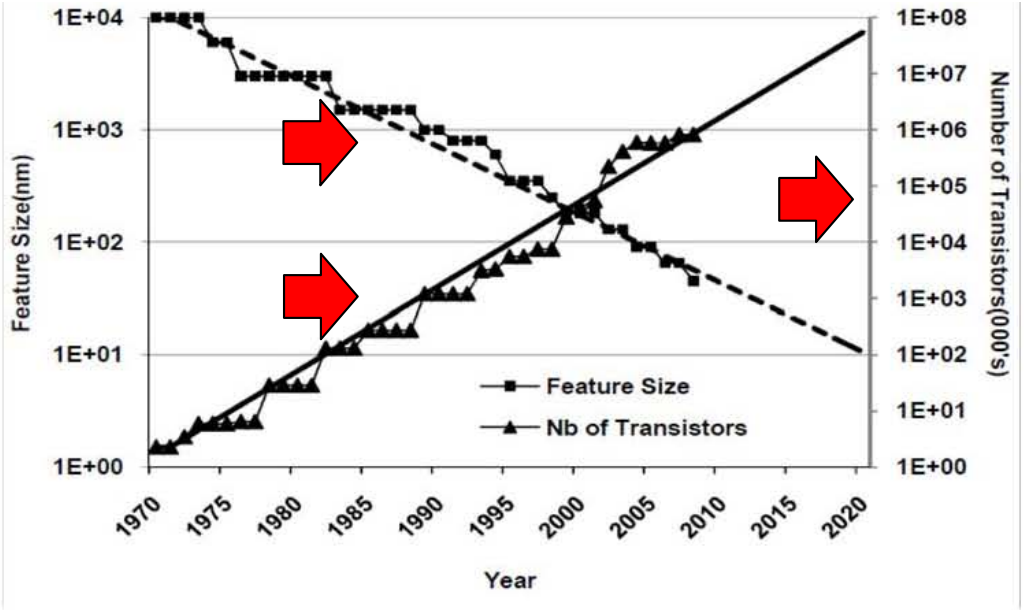
Fig 1.5
Dubois et al.
Parallel Computer
Organization and Design

Key factors:

- **Technology improvements only (19% annual improv.) Deeper pipelines + architecture innovations**
- **Altogether 49% up until 2002**

Processor Architecture 2(2)

Computer architects take advantage of the growing number of transistors:



- New process every two year. Feature size reduced by 30% in every new process
- Transistor count doubles every 2 years (Moore's law)

- A sandbox to play in so to speak
- How do we use 100B transistors?

This trend is tapering off

Memory Systems

- Main memory speed grows slower than processor speed (“Memory wall”)
- A multi-level memory hierarchy tries to realize a memory that is big, fast and cheap
- It works surprisingly well due to the Principle of locality

Memory	Size	Marginal Cost	Cost per MB	Access time
L2 Cache(on chip)	1MB	$20/MB	$20	5nsec
Main Memory	1GB	$50/GB	5c	200nsec
Disk	500GB	$100/500GB	0.02c	5msec

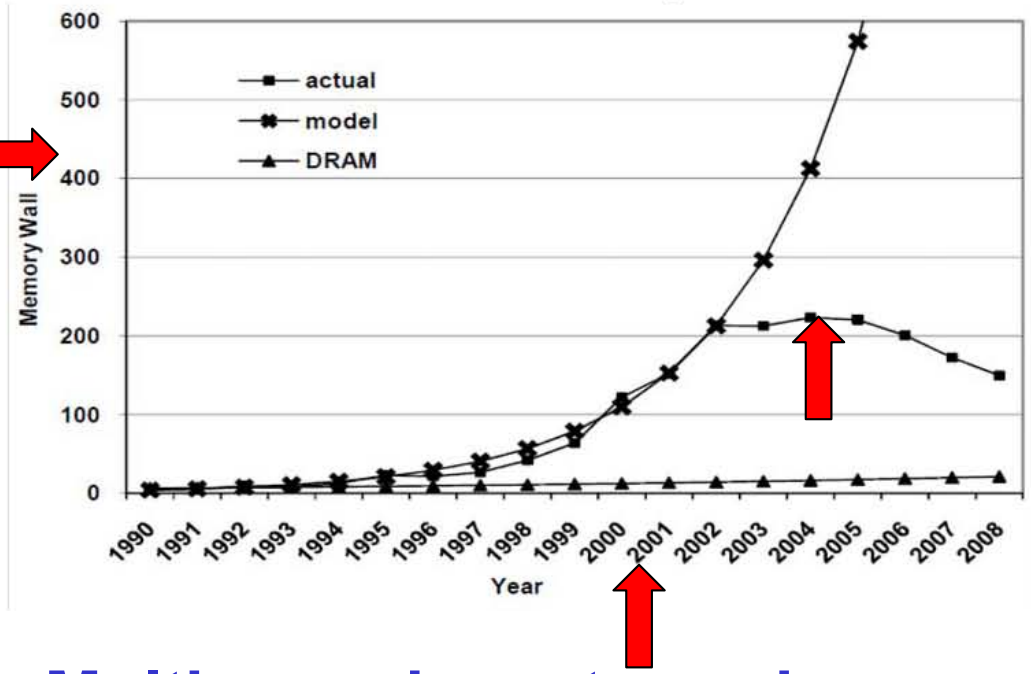
Table shows the cost and speed of memories in a PC (2008)

The Memory Wall

Microprocessor speed has increased by 50% every year

- DRAM performance has improved by only 7% per year
- DRAM density keeps increasing by 4X every 3 years

Trends have changed dramatically in the last years



DRAM: 1.07 CGR
Memory wall = memory_cycle/processor_. cycle

- In 1990: 4 (25MHz,150ns)
- In 2002: ~ 200
- > 2002: Is tapering off/reversing

Multicores have turned memory wall into a bandwidth wall

Parallelism in Architectures 1(2)

Scalar processors do operations on single (not vector) operands

- A typical scalar instruction

Example) ADD O1,O2,O3 /O2+O3==>O1

Multiple scalar instructions can be executed at a time:

- Pipelining, superscalars, superpipelining

These approaches exploit ***Instruction-Level Parallelism*** *(ILP)* exposed in a single thread/process execution

Chip multiprocessors (or multicores) exploit parallelism across threads running in parallel on different processors/cores: (***Thread-Level Parallelism*** TLP)

Parallelism in Architectures 2(2)

Vector and array processors

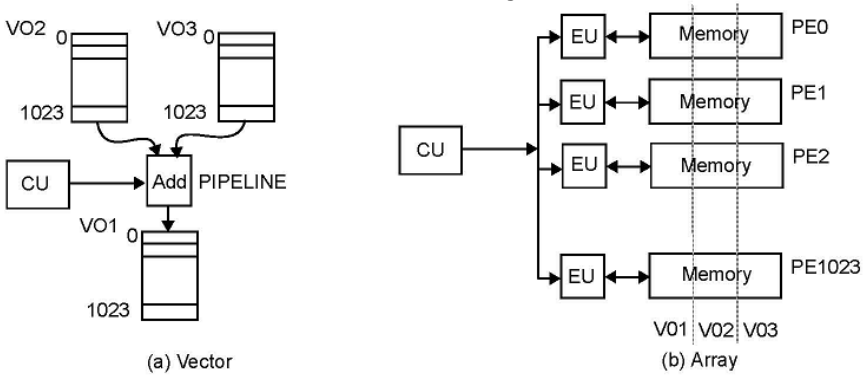
- Example vector instruction

  VADD VO1,VO2,VO3 /VO2+VO3==>VO1

- VOk is a vector of scalar components
- Equivalent to computing

  VO2[i]+VO3[i]==>VO1[i], i=0,1,..,N

Vectors are executed by pipelines or parallel arrays



Performance Metrics and Evaluation (Ch 1.4)

Performance Metrics (Measures)

Metric #1: Time to complete a task (Texe): Execution time, response time, latency

- "X is N times faster than Y" means Texe(Y)/Texe(X) = N
- The major metric used in this course

Metric #2: Number of tasks per time unit (e.g., hr, s, ns)

- The throughput for X is N times higher than Y if
  Throughput(X)/Throughput(Y) = N

NOTE: Does not always lead to same design options

Example of unreliable metrics: MIPS and MFLOPS

Execution time is the ultimate measure of performance

=>Benchmarking

Which Programs to Choose?

Early unsuccessful attempts

Real programs:

- **Problems:** Porting, complexity and not easy to interpret results

Kernels

- Computationally intense piece of real program
- **Problem:** Does not capture impact of entire program

Toy benchmarks:

- **Problem:** Unclear link to reality

Synthetic benchmarks

- **Problem:** Inherently unreal

Benchmark Suites

Commonly used benchmark suites

- SPEC: Standard Performance Evaluation Corporation
- Scientific/Engineering/General Purpose
- Integer and floating point intense program suites
- New editions in 1995, 1998, 2000, 2006 and 2017
- TPC benchmarks (transaction processing)
- Embedded benchmarks (EEMBC)
- Media benchmarks

Reporting Performance

Given a set of N programs

(Weighted) arithmetic mean: $\sum_i T_i/N$ **or** $\sum_i T_i \times W_i$

Speedup over a reference machine R: $S_i = \frac{T_{R,i}}{T_i}$

Geometric mean of speedup: $\bar{S} = \sqrt[N]{\prod_{i=1}^{N} S_i}$

Problem with Arithmetic Mean (Lower is better)

	Program 1 [s]	Program 2 [s]	Program 3 [s]
Machine 1	1	2	10
Machine 2	2	3	4
Reference	10	24	60

	Arithmetic mean [s]
Machine 1	4.3
Machine 2	3
Reference	31.3

Outliers!

Geometric Mean (Higher is better)

	Program 1 [s]	Program 2 [s]	Program 3 [s]
Machine 1	1	2	10
Machine 2	2	3	4
Reference	10	24	60

	S for P1	S for P2	S for P3	Geom. mean
Machine 1	10/1=10	24/2=12	60/10=6	8.96
Machine 2	10/2=5	24/3=8	60/4=15	8.43
Reference				

Used in performance reports (eg SPEC)

Quiz 1.2

The execution times for programs P1, P2 and P3: 1, 4 and 8 s on machine M1 and 2, 8, 16 s on a reference machine R.

Which of the following statements are correct?

A) The average execution time of M1 is 4.3 s

B) The average execution time of R is 8.6 s

C) M1 is (8.6/4.3 =) 2 X faster than R

D) R is (8.6/4.3 =) 2 X faster than M1

E) Using R as a reference, the geomentric mean of speedup is 2

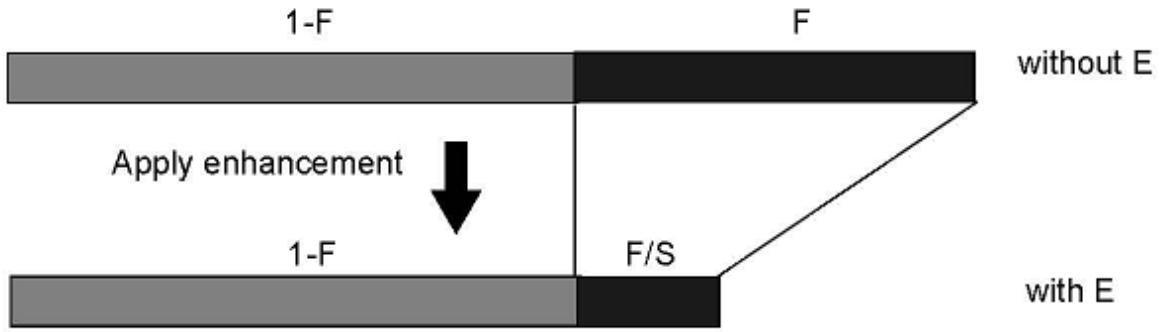
Fundamental Performance Equations for CPUs

$$Texe = IC \times CPI \times Tc$$

- **IC:** Depends on program, compiler and ISA.
- **CPI:** Depends on ***instruction mix***, ISA, and implementation
- **Tc:** Depends on implementation and technology

When processor executes more than one instruction per clock use: **Texe = (IC X Tc)/IPC**

Amdahl's Law



Enhancement ***E*** accelerates a fraction ***F*** of the task by a factor *S*

$$T_{exe}(withE) = T_{exe}(withoutE) X \left[(1 - F) + \frac{F}{S} \right]$$

$$Speedup(E) = \frac{T_{exe}(withoutE)}{T_{exe}(withE)} = \frac{1}{(1 - F) + \frac{F}{S}}$$

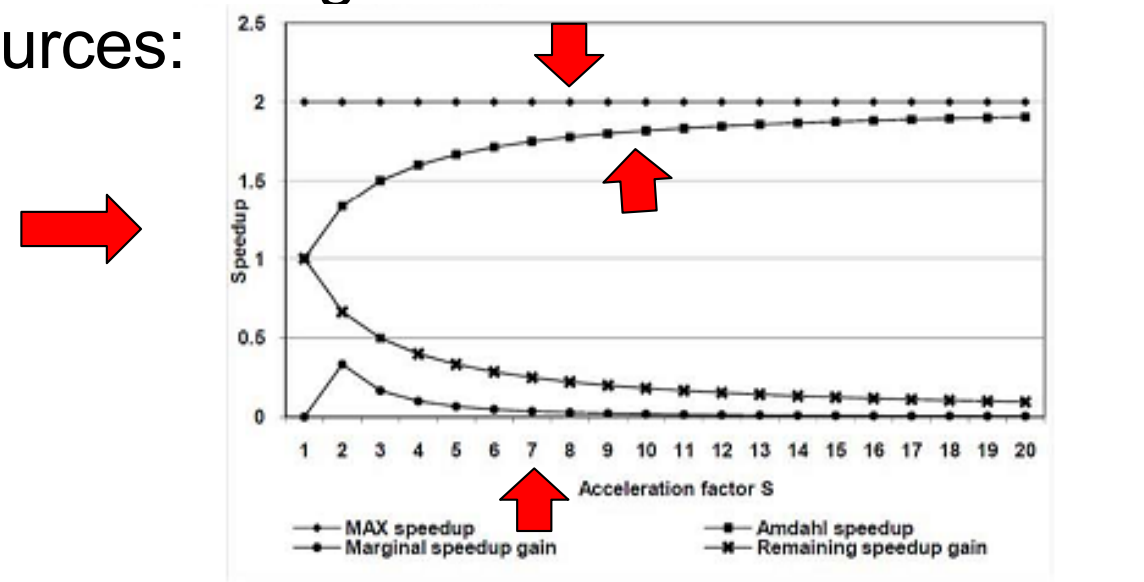
Lessons Learned from Amdahl's Law

LESSON #1: Improvement is limited by the fraction of the execution time that cannot be enhanced:

$$SPEEDUP(E) < \frac{1}{1-F}$$

LESSON #2: Optimize the common case; for example, deal with rare cases in software.

LESSON #3: Law of diminishing returns: Gains diminish as we add more resources:



F=0.5

Power 1(2)

Total Power = Dynamic + Static (leakage)

$$P_{dynamic} = \alpha C V^2 f$$

$$P_{static} = V I_{sub} \propto V e^{-k\frac{V_T}{T}}$$

Dynamic power favors parallel processing over higher clock frequency. Dynamic power roughly proportional to frequency^3

- **Example)**
  - Replicate a single core four times: 4X speedup and 4X dynamic power
  - Increase clock frequency 4X: 4X speedup but 64X dynamic power

Static power: Circuits leak independent of frequency

Power 2(2)

Power/energy are critical problems

- Power (immediate energy dissipation) must be dissipated
- Otherwise temperature goes up which affects performance, correctness, and may eventually destroy the circuit
- Effect on the supply of power to the chip
- Energy (depends on power and speed)
- Costly; a global problem
- Problematic for battery-operated devices

Quiz 1.3

The execution time of a program on a single processor is 100s. 50% of that execution has a speedup linear to the number of processors

Which of the following statements are correct?

A) The execution time of the program on two processors is 50 s

B) The execution time of the program on two processors is 75 s

C) The execution time of the program can never go below 50 s regardless of the number of processors

D) The execution time of the program with 100 processors is 1 s

What you should know by now

- Definition of computer architecture and the role of a computer architect
- Components of a computer organization and how trends affect them (Moore's law, Memory wall…)
- Parallelism in computer architecture: ILP and TLP
- Quantitative metrics and methods for reporting performance:
  - Benchmarking
  - Execution time and speedup (arithmetic, harmonic, geometric etc)
  - Amdahl's Law