Lecture 1: CS/ECE 3810 Introduction

- Today's topics:
  - Why computer organization is important
  - Logistics
  - Modern trends

Why Computer Organization

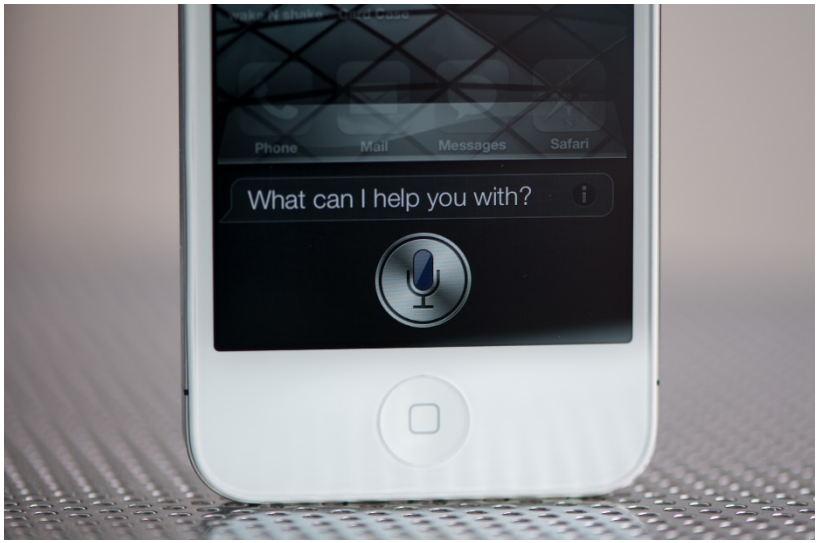


Image credits: uber, extremetech, anandtech

Why Computer Organization

Image credits: gizmodo

Why Computer Organization

- Embarrassing if you are a BS in CS/CE and can't make sense of the following terms: DRAM, pipelining, cache hierarchies, I/O, virtual memory, …
- Embarrassing if you are a BS in CS/CE and can't decide which processor to buy: 4.4 GHz Intel Core i9 or
  4.7 GHz AMD Ryzen 9 (reason about performance/power)
- Obvious first step for chip designers, compiler/OS writers
- Will knowledge of the hardware help you write better and more secure programs?

Must a Programmer Care About Hardware?

- Must know how to reason about program performance and energy and security
- Memory management: if we understand how/where data is placed, we can help ensure that relevant data is nearby
- Thread management: if we understand how threads interact, we can write smarter multi-threaded programs

→ Why do we care about multi-threaded programs?

Example

200x speedup for matrix vector multiplication

- Data level parallelism: 3.8x
- Loop unrolling and out-of-order execution: 2.3x
- Cache blocking: 2.5x
- Thread level parallelism: 14x

Further, can use accelerators to get an additional 100x.

Key Topics

- Moore's Law, power wall
- Use of abstractions
- Assembly language
- Computer arithmetic
- Pipelining
- Using predictions
- Memory hierarchies
- Accelerators
- Reliability and Security

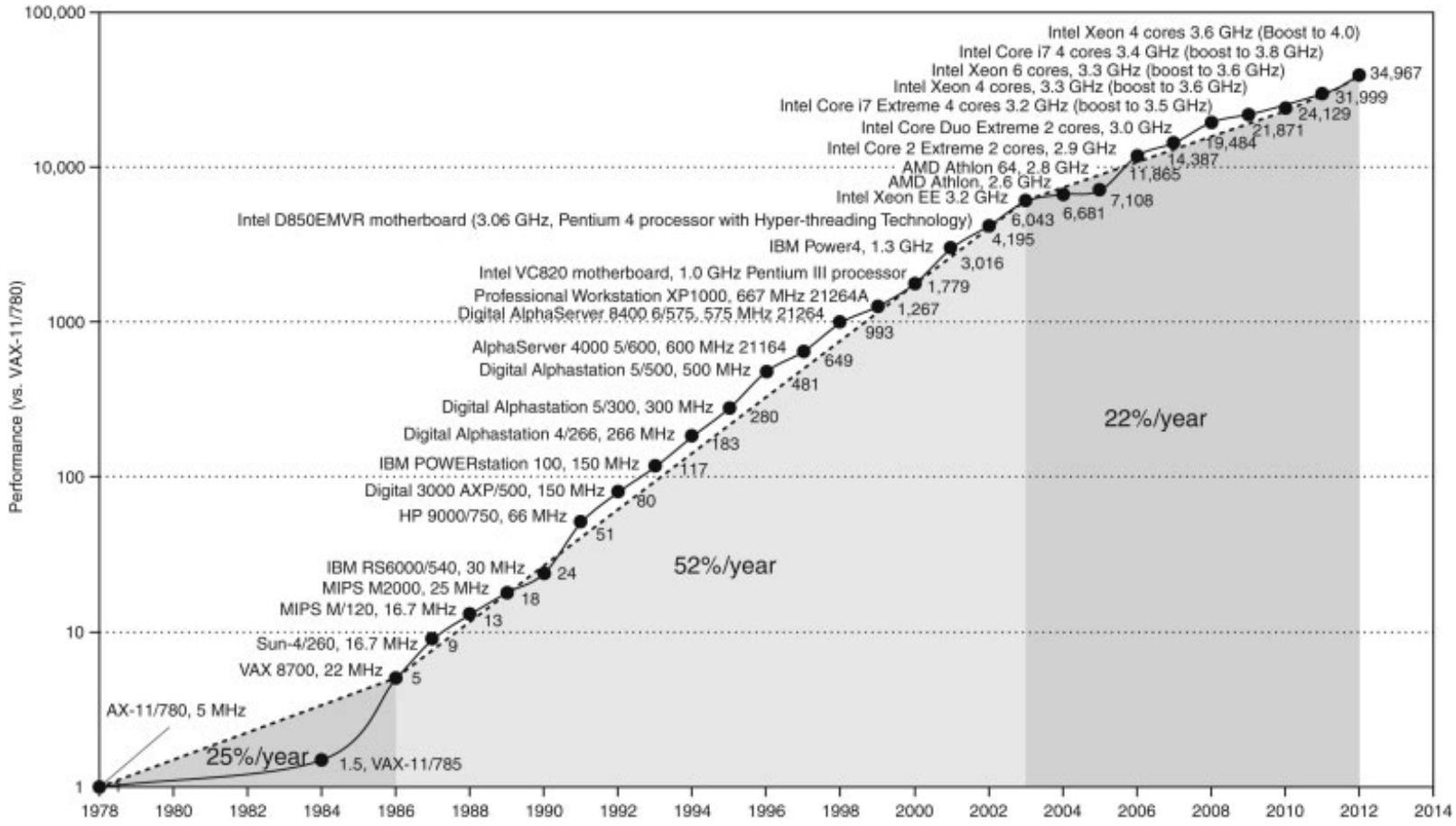
Logistics

- See class web-page
  https://www.cs.utah.edu/~rajeev/cs3810
- COVID reminders – follow university guidelines – lectures also available on Zoom, but not recorded
- TAs and office hours: TBA
- Most communication on Canvas; email me directly to set up office hours, or meet me right after class
- Textbook: Computer Organization – HW/SW Interface, Patterson and Hennessy, 5th or 6th edition

Course Organization

- 30% midterm, 40% final, 30% assignments
- ~10 assignments – you may skip two; assignments due at the start of class (upload on Canvas)
- Co-operation policy: you may discuss – you may not see someone else's written matter when writing your solution
- Exams are open-notes
- Print slides just before class
- Screencast YouTube videos

Microprocessor Performance

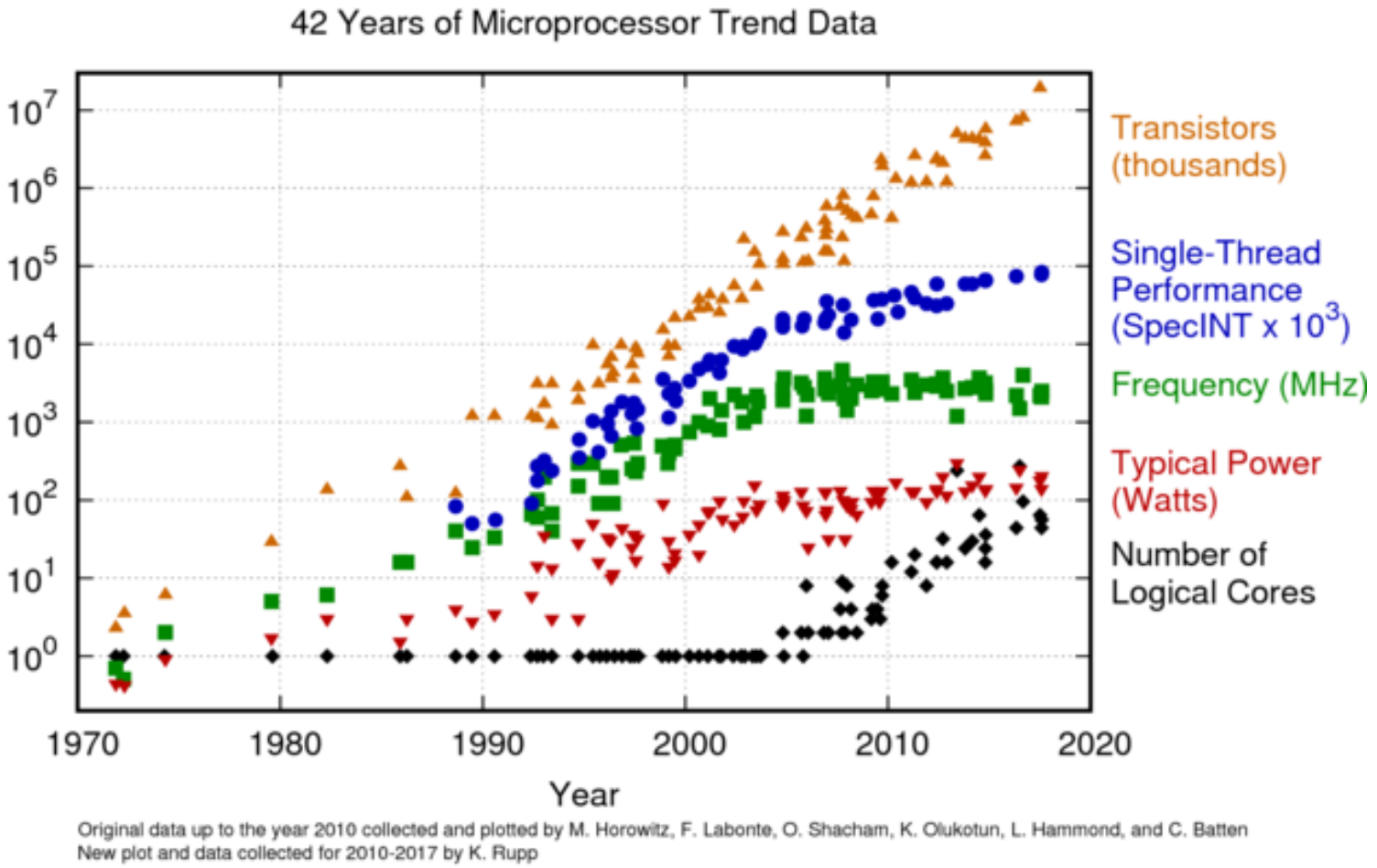


Source: H&P Textbook

50% improvement every year!!

What contributes to this improvement?

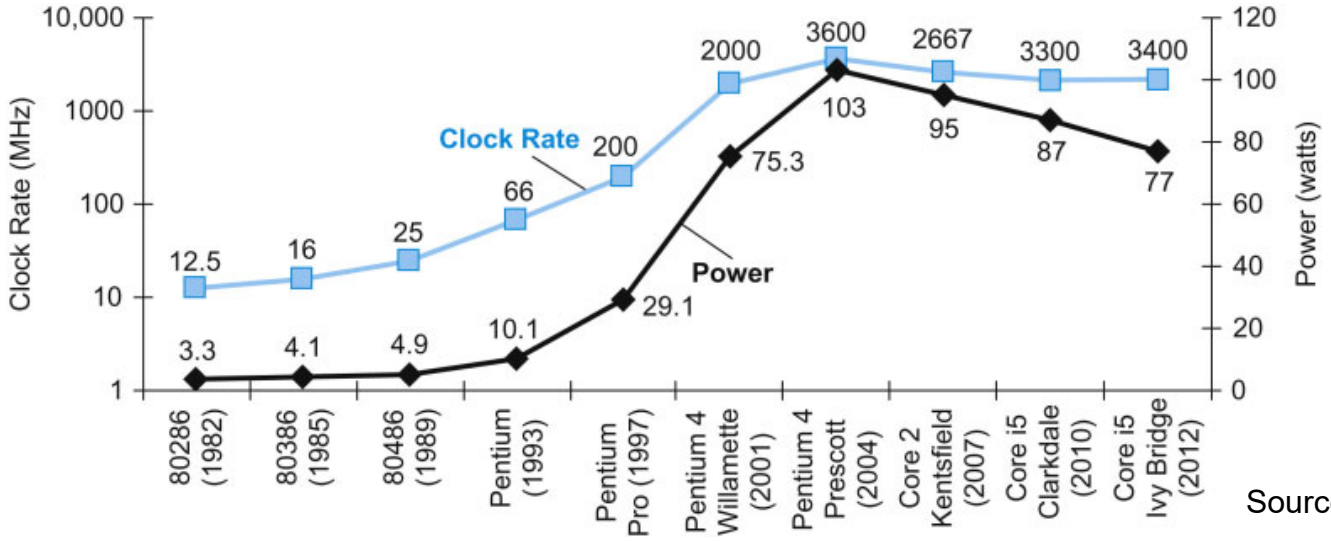
Microprocessor Performance



Source: karlrupp.net

Power Consumption Trends

- Dyn power α activity x capacitance x voltage^2 x frequency
- Voltage and frequency are somewhat constant now, while capacitance per transistor is decreasing and number of transistors (activity) is increasing
- Leakage power is also rising (function of #trans and voltage)



Source: H&P Textbook

Summary

- Increasing frequency led to power wall in early 2000s
- Frequency has stagnated since then
- End of voltage (Dennard) scaling in early 2010s
- Has led to dark silicon and dim silicon (occasional turbo)

Important Trends

- Running out of ideas to improve single thread performance
- Power wall makes it harder to add complex features
- Power wall makes it harder to increase frequency
- Additional performance provided by: more cores, occasional spikes in frequency, accelerators

Important Trends

- Historical contributions to performance:
  1. Better processes (faster devices) ~20%
  2. Better circuits/pipelines ~15%
  3. Better organization/architecture ~15%

In the future, bullet-2 will help little and bullet-1 will eventually disappear!

	Pentium	P-Pro	P-II	P-III	P-4	Itanium	Montecito
Year	1993	95	97	99	2000	2002	2005
Transistors	3.1M	5.5M	7.5M	9.5M	42M	300M	1720M
Clock Speed	60M	200M	300M	500M	1500M	800M	1800M

Moore's Law in action

At this point, adding transistors to a core yields little benefit

What Does This Mean to a Programmer?

- Today, one can expect only a 20% annual improvement; the improvement is even lower if the program is not multi-threaded
  - A program needs many threads
  - The threads need efficient synchronization and communication
  - Data placement in the memory hierarchy is important
  - Accelerators should be used when possible

Challenges for Hardware Designers

- Find efficient ways to
  - improve single-thread performance and energy
  - improve data sharing
  - boost programmer productivity
  - manage the memory system
  - build accelerators for important kernels
  - provide security

Next Class

- Topics: Trends, Performance, MIPS instruction set architecture (Chapter 2)

- Visit the class web-page
  https://www.cs.utah.edu/~rajeev/cs3810