CMSC 411
Computer Systems Architecture
Lecture 1
Computer Design and Evaluation

Alan Sussman
als@cs.umd.edu

Slides also contributed by Pete Keleher, Chau-Wen Tseng

Administrivia

- Class web pages are at http://www.cs.umd.edu/class/spring2014/cmsc411-0201
  - and linked in from CS dept. class web pages
- Class accounts for projects are on CSIC Linux cluster
- First homework, for Unit 1, announced Tuesday, 2/4
- Start reading Ch. 1 of H&P

What can you expect to learn?

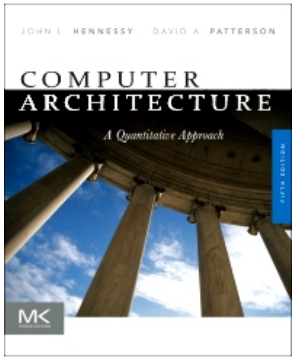
- What to look for in buying a PC (brag to parents and friends!)
- How computer architecture affects programming style
- How programming style affect computer architecture
- How processors/disks/memory work
- How processors exploit instruction/thread parallelism
- A great deal of jargon

Rules and reminders

- All course rules in syllabus
- Do the homework!!!
- Lecture notes available on the web – after class
- I'll post homework (and exam) answers after the due date – password protected, and you'll get email with the password – account name cmsc411
- Keep ahead of your work
  - Get help as soon as you need it
    - » Office hours, Piazza (email as a last resort)
- Don't disturb other students in class
  - Keep cell phones quiet
  - No laptops in class

The Textbook – H&P

- Everyone complains about it
- Virtually everyone uses it
- You can handle it, but you have to work at it – *do the reading*
- Through lecture notes, other references, etc., I'll try to help you put it all together
- Appendices available on book web site at http://booksite.mkp.com/9780123838728



Chapter 1 of H&P

- Read Chapter 1
- Historical Perspective – Appendix L.2 of H&P
  - Computers as we know them are roughly 60 years old
  - The *von Neumann machine* model that underlies computer design is only partially von Neumann's
  - Konrad Zuse says he had "the bad luck of being too early"?
    - » Optional: Read his own recollections in TR 180 of ETH, Zürich, http://www.inf.ethz.ch/research/disstechreps/techreports/show?serial=180&lang=en (contains both German and English)
  - No one was able to successfully patent the idea of a stored-program computer, much to the dismay of Eckert and Mauchly

Early development steps

- Make input and output easier than wiring circuit boards and reading lights
- Make programming easier by developing higher level programming languages, so that users did not need to use binary machine code instructions
  - First compilers in late 1950's, for Fortran and Cobol
- Develop storage devices

Later development steps

- Faster
- More storage
- Cheaper
- Networking and parallel computing
- Better user interfaces
- Ubiquitous applications
- Development of standards

Perspective: An example

- Most powerful computer in 1988: CRAY Y-MP
- 1993: a desktop workstation (IBM Power-2) matched its power at less than 10% of the cost
- How did this happen?
  - hardware improvements, e.g., squeezing more circuits into a smaller area
  - improvements in instruction-set design, e.g., making the machine faster on a small number of frequently used instructions
  - improvements in compilation, e.g., optimizing code to reduce memory accesses and make use of faster machine instructions

Perspective: An example (cont.)

- Most powerful computer in 1988: CRAY Y-MP
  - 8 vector processors (~0.3 GFlops each)
  - 32 bit registers, 512 MB SRAM
- 2013: Intel Core i7
  - 4 cores (~45 GFlops each)
  - 64 bit registers, 8 MB L3 cache
- How did this happen?
  - Even more hardware improvements, e.g., squeezing more circuits into a smaller area
  - improvements in instruction-level parallelism, e.g., executing more instructions at the same time to improve performance

COMPUTER ARCHITECTURE AT A CROSSROADS

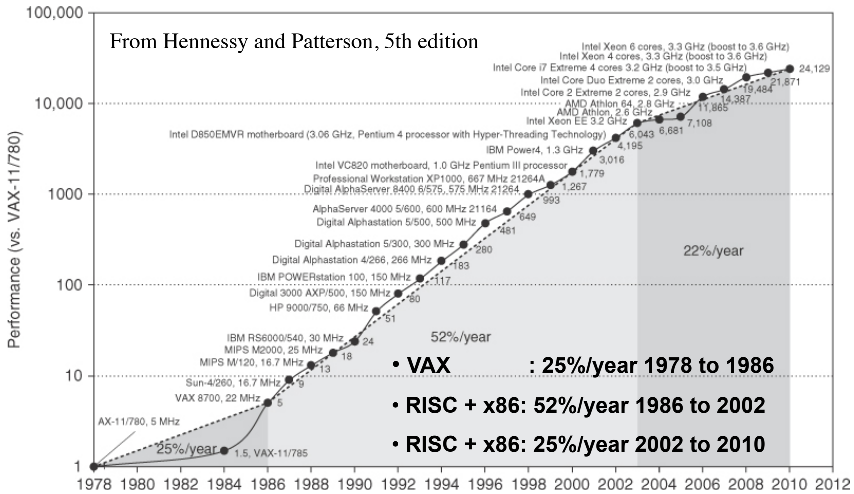
Crossroads: Conventional Wisdom in Comp. Arch

- Old Conventional Wisdom: Power is free, Transistors expensive
- New Conventional Wisdom: "Power wall" Power expensive, transistors free (Can put more on chip than can afford to turn on)
- Old CW: Sufficiently increasing Instruction Level Parallelism (ILP) via compilers, innovation (Out-of-order, speculation, VLIW, …)
- New CW: "ILP wall" law of diminishing returns on more HW for ILP
- Old CW: Multiplies are slow, Memory access is fast
- New CW: "Memory wall" Memory slow, multiplies fast (200 clock cycles to DRAM memory, 4 clocks for multiply)
- Old CW: Uniprocessor performance 2X / 1.5 yrs
- New CW: Power Wall + ILP Wall + Memory Wall = Brick Wall
  - Uniprocessor performance now 2X / 5(?) yrs

⇒ Sea change in chip design: multiple "cores" (2X processors per chip / ~ 2 years)

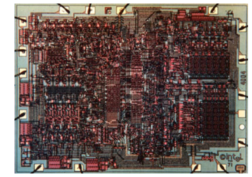
» More simpler processors are more power efficient

Crossroads: Uniprocessor Performance

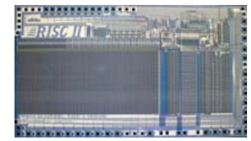


Sea Change in Chip Design

- Intel 4004 (1971): 4-bit processor, 2312 transistors, 0.4 MHz, 10 micron PMOS, 11 mm² chip



- RISC II (1983): 32-bit, 5 stage pipeline, 40,760 transistors, 3 MHz, 3 micron NMOS, 60 mm² chip



- 125 mm² chip, 0.065 micron CMOS = RISC II+FPU+Icache+Dcache
  - RISC II shrinks to ~ 0.02 mm² at 65 nm
  - Caches via DRAM or 1 transistor SRAM (www.t-ram.com) ?
  - Proximity Communication via capacitive coupling at > 1 TB/s ? (Ivan Sutherland @ Sun / Berkeley)
- **Processor is the new transistor?**

Multiprocessors - Déjà vu all over again?

- Multiprocessors imminent in 1970s, '80s, '90s, …
- "… today's processors … are nearing an impasse as technologies approach the speed of light.."

  David Mitchell, *The Transputer: The Time Is Now* (1989)
- Transputer was premature
  ⇒ Custom multiprocessors strove to lead uniprocessors
  ⇒ Procrastination rewarded: 2X seq. perf. / 1.5 years
- "We are dedicating all of our future product development to multicore designs. … This is a sea change in computing"

  Paul Otellini, President, Intel (2004)
- Difference is all microprocessor companies switch to multiprocessors (AMD, Intel, IBM; all new Apples >= 2 CPUs)
  ⇒ Procrastination penalized: 2X sequential perf. / 5 yrs
  ⇒ Biggest programming challenge: 1 to 2 CPUs

Problems with Sea Change

- Algorithms, Programming Languages, Compilers, Operating Systems, Architectures, Libraries, … not ready to supply Thread Level Parallelism or Data Level Parallelism for 1000 CPUs / chip,
- Architectures not ready for 1000 CPUs / chip
  - Unlike Instruction Level Parallelism, cannot be solved just by computer architects and compiler writers alone, but also cannot be solved *without* participation of computer architects
- This 5th Edition of textbook Computer Architecture: A Quantitative Approach explores shift from Instruction Level Parallelism to Thread Level Parallelism / Data Level Parallelism