Introduction

• Why are you taking this course?
  – You really liked the material in 311 and want to learn more?
  – The course time fit into your schedule well?
  – You needed upper level CS courses and chose this one at random?
  – All the courses you really wanted to take were filled?

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
• A great deal of jargon

Administrivia

• Class web pages are at http://www.cs.umd.edu/class/spring2009/cmsc411
  – and linked in from CS dept. class web pages
• Class accounts for project will be on CSIC Linux cluster
• First homework, for Unit 1, announced Tuesday, 2/3
• Start reading Ch. 1 of H&P
Syllabus

- More on web page
- Importance of doing 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

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

Chapter 1 of H&P

- Read Chapter 1
- Historical Perspective - Section 1.13
  - 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
  - Why does Konrad Zuse say 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

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

From Hennessy and Patterson, 4th edition

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
  = 2312 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.…”
- 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, Sun; 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
**COMPUTER ARCHITECTURE VS. INSTRUCTION SET ARCHITECTURE**

---

**Instruction Set Architecture: Critical Interface**

- Properties of a good abstraction
  - Lasts through many generations (portability)
  - Used in many different ways (generality)
  - Provides convenient functionality to higher levels
  - Permits an efficient implementation at lower levels

---

**Example: MIPS**

Programmable storage

- 2^32 x bytes
- 31 x 32-bit GPRs (R0=0)
- 32 x 32-bit FP regs (paired DP)

Data types?

- Format?
- Addressing Modes?

Arithmetic logical

- Add, AddU, Sub, SubU, And, Or, Xor, SLT, SLTU,
- AddI, AddIU, SLTI, SLTIU, AndI, OrI, XorI,
- SLL, SRL, SRA

Memory Access

- LB, LBU, LH, LHU, LW,
- SB, SH, SW

Control

- 32-bit instructions on word boundary
- J, JAL, JR, JALR
- BEq, BNE, BLEZ, BGTZ, BLTZ, BGEZ

---

**Instruction Set Architecture**

“... 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, Blaauw, and Brooks, 1964

---

**SOFTWARE**

- Organization of Programmable Storage
- Data Types & Data Structures: Encodings & Representations
- Instruction Formats
- Instruction (or Operation Code) Set
- Modes of Addressing and Accessing Data Items and Instructions
- Exceptional Conditions
ISA vs. Computer Architecture

- Old definition of computer architecture = instruction set design
  - Other aspects of computer design called implementation
  - Insinuates implementation is uninteresting or less challenging
- H&P’s view is computer architecture >> ISA
- Architect’s job much more than instruction set design; technical hurdles today more challenging than those in instruction set design
- Since instruction set design not where action is, some conclude computer architecture (using old definition) is not where action is
  - H&P disagree on conclusion
  - Agree that ISA not where action is (ISA in CA:AQA 4/e appendix)

Comp. Arch. is an Integrated Approach

- What really matters is the functioning of the complete system
  - hardware, runtime system, compiler, operating system, and application
  - In networking, this is called the “End to End argument”
- Computer architecture is not just about transistors, individual instructions, or particular implementations
  - E.g., Original RISC projects replaced complex instructions with a compiler + simple instructions

Computer Architecture is Design and Analysis

- Architecture is an iterative process:
  - Searching the space of possible designs
  - At all levels of computer systems

Costs

- From Figure 1.9 in H&P 3/e
- The cost of components in a $1000 PC in 2001 are:
  - CPU – 22%
  - Monitor – 19%
  - Hard drive – only 9%
  - DRAM – only 5% (for 128MB)
  - Software – 20% (OS & basic office suite)
Manufacture of DRAM and other chips

- Chips are manufactured on wafers - circular disks containing many dies (chips).
- The wafer is tested and chopped into dies.