CSE 410
Computer Systems

Hal Perkins
Spring 2010
Lecture 1 – Course Introduction
Reading and References

• Reading
  – Computer Organization and Design, Patterson and Hennessy
    • Chapter 1, sec. 1.1-1.3, 1.5 (background)
    • Chapter 2, sec. 2.1-2.5
Administrative

• Instructor:
  – Hal Perkins
  – perkins@cs.washington.edu, CSE548
• TAs:
  – Michael Ratanapintha
  – Euzel Villaneuva
• All class info is on the web site
Class Overview

• Provide an introduction to the inner workings of computer systems
• Levels of abstraction
  – bits, bytes, assembly language
  – operating system concepts
  – higher level languages - C, C++, Java, …
  – application programs
Goal

• You will understand
  – what is actually happening when a computer system is running application programs
• So that you will be able to
  – make good design choices as a developer, project manager, or system customer
  – calibrate your hype-o-meter with facts
Main Topics

• The hardware / software interface
  – the elements of a computer system
  – what parts are visible to the software
  – instruction set architecture (ISA)
  – what happens inside the CPU

• Operating systems
  – services an OS performs for an application
  – design of various OS components
  – OS mechanisms and policies
Course Mechanics

• 3 Lectures/week
• Homeworks most weeks
  – Written problems, small programming exercises
• Office hours tba, scattered through week
  – Use them!
• Online discussion board to stay in touch between classes / office hours
Homework & Exams

- \( \approx 6-7 \) assignments (50%)
- Midterm, tentatively Fri. April 30 or Mon. May 3 (20%)
- Final, Tue. June 8, 2:30 (25%)
- Participation, citizenship, etc. (5%)

- Late policy: 4 “late days”, at most 2 on any single assignment, counted in 24 hour chunks, otherwise no late assignments.
  - Save late days for later!
Academic Integrity

- Policy on the course web. **Read it!**
- Do your own work – always explain any unconventional action on your part
- I trust you completely
- I have no sympathy for trust violations – nor should you
- Honest work is the most important feature of a university. It shows respect for your colleagues and yourself.
What’s a Computer?

- For our purposes (for now)...

```
main memory

I/O bus

hard disk  video / camera  optical drive  serial ports  network interface

processor/memory bus

processor
```
Architecture and Organization

• Architecture (the boxes)
  – defines elements and interfaces between layers
  – ISA (Instruction Set Architecture): instructions, registers, addressing – programmer’s view of the hardware

• Organization / Implementation (inside the boxes)
  – components and connections
  – how instructions are implemented in hardware
  – many different organizations can implement a single architecture
  – One organization can support multiple architectures(!)
Instruction set architectures

- Interface between hardware and software
  - abstraction: hide HW complexity from the software through a set of simple operations and devices

  add, mul, and, lw, ...
Computer Architecture

• Specification of how to program a specific computer family
  – what instructions are available?
  – how are the instructions formatted into bits?
  – how many registers and what is their function?
  – how is memory addressed?
  – how does I/O work?
Architecture Families

- IBM 360, 370, … (the first computer family)
- PowerPC 601, 603, …
- DEC VAX, PDP-11
- Intel x86: 286, 386, 486, Pentium, P4, Core…
- Intel IA64 Itanium
- MIPS R2000, R3000, R4000, R5000, …
- SUN Sparc
- ARM family
MIPS

- In this class, we'll use the MIPS instruction set architecture (ISA) to illustrate concepts in assembly language and machine organization
  - Of course, the concepts are not MIPS-specific
  - MIPS is just convenient because it is real, yet simple (unlike x86)
- The MIPS ISA is still used in many places today. Primarily in embedded systems, like:
  - Various routers from Cisco
  - Game machines like the Nintendo 64 and Sony Playstation 2
Roadmap

• To start: learn to program at the architecture / instruction set / memory level
  – Information representation (bits, bytes, …)
  – MIPS assembly language programming
• Then look at some of the core implementation issues
  – Pipelining
  – Memory hierarchy (caches, virtual memory)
• Hardware from the programmer’s perspective:
  – How does my code run?
  – Why is it fast or slow?
How to Succeed in CSE 410

• Remember the big picture
  – What are we trying to accomplish, and why?
• Read the textbooks
  – They’re clear, well-organized, and well-written (particularly P&H). Work through the examples and try some exercises on your own. Read the “Real Stuff” and “Historical Perspective” sections.
• Talk to each other
  – You can learn a lot from other students, both by asking and answering questions.
• Help us help you
  – Come to lectures and office hours. Use the discussion board. Ask lots of questions! Check out the web pages.