CS 61C: Great Ideas in Computer Architecture (Machine Structures)

Instructors:
Randy H. Katz
David A. Patterson
http://inst.eecs.Berkeley.edu/~cs61c/fa10

Agenda

- Great Ideas in Computer Architecture
- Administrivia
- Technology Break
- How to Talk to a Computer

Old School CS61c

New School CS61c
Great Ideas in Computer Architecture

- Layers of Representation/Interpretation
- Moore’s Law
- Principle of Locality/Memory Hierarchy
- Parallelism
- Performance Improvement

Levels of Representation/Interpretation

High Level Language Program (e.g., C)
Compiler
Assembly Language Program (e.g., MIPS)
Assembler
Machine Language Program (MIPS)

Machine Interpretation
- Hardware Architecture Description (e.g., block diagrams)
- Logic Circuit Description (Circuit Schematic Diagrams)

Moore’s Law

Predicts: 2X Transistors / chip every 2 years
Gordon Moore
Intel Co-founder
B.S. Cal 1950!

Principle of Locality/Memory Hierarchy

Parallelism

Jane
Research, Composition, Typing
Sue
Research, Composition, Typing
Tom
Research, Composition, Typing
Performance Improvement

- Matching application to underlying hardware to exploit:
  - Locality
  - Parallelism
  - Special hardware features, like specialized instructions (e.g., matrix manipulation)

- Latency
  - How long to set the problem up
  - How much faster does it execute once it gets going
  - It is all about ‘me to finish’

Agenda

- Great Ideas in Computer Architecture
- Administrivia
- Technology Break
- How to Talk to a Computer

Course Information

- Instructors:
  - Randy Katz, Dave Patterson
- Teaching Assistants:
  - Andrew Gearhart, Michael Greenbaum, Conor Hughes, Charles Reiss
- Textbooks:
  - Patterson, Hennessy, Computer Organization and Design, 4th Edition
- Course Web: http://inst.eecs.Berkeley.edu/~cs61c/fa10
- Google Group: 61CFall2010UCB

Course Organization

- Grading
  - Class Participation (5%)
  - Homework (5%)
  - Labs (20%)
  - Projects (40%)
    - Computer Instruction Set Simulator
    - Data Parallelism (Map-Reduce on EC2)
    - Computer Processor Design (Logsim)
    - Performance Tuning of a Parallel Application (partnered)
  - Midterm (10%)
  - Final (20%)

Course Effort Participation

- Altruism

- Effort
  - Participation
  - Altruism

- Altruism

- Effort
  - Participation
  - Altruism
The Rules (and we really mean it!)

Agenda

- Great Ideas in Computer Architecture
- Administrivia
- Technology Break
- How to Talk to a Computer

Agenda

- Great Ideas in Computer Architecture
- Administrivia
- Technology Break
- How to Talk to a Computer

Speaking to a Computer

- A word a computer understands called an instruction
- Vocabulary of all words a computer understands are called an instruction set
- Different computers may have different vocabularies (instruction sets)
  - Computer vocabulary used in iPhone is not same as computer vocabulary used in Macbook laptop
  - Or the same vocabulary
    - iPhone and iPad computers have same instruction set
  - Why not all the same? Why not all different? What might be pros and cons?

Instruction Set in this Class

- MIPS
  - Invented by my co-author John Hennessy
  - Standard instruction set for networking gear
- Elegant example of instruction sets called Reduced Instruction Set Computer (RISC)
  - RISC: Limited vocabulary in number of words and in complexity of those words
- Most popular RISC is ARM (Advanced RISC Machine)
  - 5 billion in 2010, in every cell phone (iPhone, iPad, iPod, ...)
- Another popular Instruction Set is used in Macbook and in PCs: Intel 80x86 in Core i3, Core i5, Core i7
  - 1/20th as 80x86s made in 2010 (0.3 billion)

Example MIPS Instructions

- Every computer does arithmetic
- Tell a computer to do addition?
  
  add a, b, c

  - Add b to c and put sum into a
- Needs 3 operands:
  2 sources + 1 destination for sum
- Always just 1 operation per MIPS instruction
- How write same operation in Java?
Guess the other MIPS instructions

- Subtract c from b and put difference in a?
- Multiply b by c and put product in a?
- Divide b by c and put quotient in a?

Example Instructions

- MIPS instructions are inflexible, rigid:
  - Just 1 arithmetic operation per instruction,
  - Always with 3 operands
- How write this Java operation in MIPS?
  \[ a = b + c + d + e \]

Comments in MIPS

- Can add comments to MIPS instruction by putting # that continues to end of line of text
  add a, b, c # b + c is placed in a.
  add a, a, d # b + c + d is now in a.
  add a, a, e # b + c + d + e is in a.

Java to MIPS

- Operators almost identical
- What is MIPS code that performs same as?
  \[ a = b + c; \]
  \[ d = a - e; \]
- What is MIPS code that performs same as?
  \[ f = (g + h) - (i + j); \]

Peer Instruction

- For a given function, which programming language likely takes the most lines of code?
  I. Scheme
  II. Java
  III. MIPS instructions

Put the three representations in order
A. Most to least: I, II, III  B. I, III, II
C. II, III, I  D. II, I, III
E. III, II, I  F. III, I, II
Summary

• Computer words called instructions
• Computer vocabulary is instruction set
• MIPS is example instruction set in this class
• Rigid format: 1 operation, 2 source operands, 1 destination
  – add, sub, mult, div, and, or, shl, shr
• Like other “programming languages,” has comments to help programmer
• Simple mapping from arithmetic expression in Java to MIPS instructions