CS333: Computer Architecture
University of Virginia Computer Science
Fall 2006
Michele Co

Class 1: Introduction

Menu
• Course Structure
• Course Goals
• First Assignment
• Course Admin (add class/change section)

Course Structure and Expectations

Staff
• Me: Michele Co
  – 228A Olsson Hall, 982-2688
  – Office Hours: (tentative) posted on course website
• TAs
  – Timothy Hnat (grad)
  – Krasimira Krapitanova (grad)
  – Ugrad TAs TBA
  – Office Hours: TBA

Contacting Us
• Office hours! Finalized times to be posted soon.
• cs333@cs.virginia.edu
  – We will try to respond within 24 hours. Please get in touch with us again if you hear no response within that time frame.
• Anonymous Feedback

Contacting You
• Information (relevant notes, announcements, etc.) will be posted on class website:
• Class Email
Meetings
- Lectures: 3 per week
  - Will include material not in the book
  - Many classes will use slides and notes
  - Some classes will not
- Lab meetings: 1 section per week
  - Won’t meet each week. In-lab activities will be announced in advance

Grading Criteria (Subject to Change)
- 2 Midterms (50%)
- Homework and Lab (20%)
- Final Exam (20%)
- Class Participation (10%)

Expectations: Honor
- Everyone will be expected to follow what is on the course pledge
- Exact content will be determined before next week based on your survey responses
  - I hope to place a large burden on the honor system, but won’t if students think this puts honest students at a disadvantage

Course Goals

CS333 Roadmap
- Levels of Abstraction
  - Application
  - High Level Languages
  - Assembly language
  - Architecture
  - Gate-level logic
  - Electronics
  - Semiconductor
- Real World Physics

Course Goal 1
- Learn to think about the machine from the
  - Architecture level
  - Assembly language level
  - Logic level
- Understand relationships and interactions between levels
Course Goal 2

Be able to understand key computer organization concepts.

Course Goal 3

Understand the processor design process and tradeoffs made to achieve particular design goals.

General Overview

Terminology

- **Machine Language**
  - Fundamental instructions expressed as 0s and 1s
- **Assembly Language**
  - Human-readable equivalent of machine language
- **Assembler**
  - Converts assembly language program to machine language
- **Stored program**
  - Program can be stored on computer with its data and be manipulated as if it were data

Machine and Assembly Language

- The assembler converts assembly language to machine language. You must also know how to do this.

<table>
<thead>
<tr>
<th>MC68000 Assembly Language</th>
<th>Machine Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVE.W D4, D5</td>
<td>0011 101 000 000 100</td>
</tr>
<tr>
<td>ADDI.W #9, D2</td>
<td>00000001 10 111 100</td>
</tr>
<tr>
<td></td>
<td>0000 0000 0000 1001</td>
</tr>
</tbody>
</table>

**Table 1.2 Two Motorola MC68000 instructions**

Assembly Programmer’s View

- **Instruction set architecture (ISA)**
  - Collection of all possible operations on a machine
  - Includes:
    - instruction set
    - machine memory
    - programmer-accessible registers
- **Tools**
  - Assembler
  - Linker
  - Debugger
Who Uses Assembly Language?

- Machine designers
  - Tradeoffs of new designs
- Compiler writers
  - Generate correct code
  - Optimized code to efficiently use resources
- Microprocessor vendors
  - Enhancements (MMX, SSE extensions, etc.)

Architect’s Goals

- Design
  - ISA: programming utility, optimum implementation performance
  - Machine: best implementation of instructions
- Performance
  - Time to completion
  - Energy-efficiency

CS333 Roadmap

Levels of Abstraction

- Application
- High Level Languages
- Assembly language
- Architecture
- Gate-level logic
- Electronics
- Semiconductor

Real World Physics

Architect’s View

- Functional building blocks and their interconnections
  - ISA (interface)
  - CPU and memory
  - Buses
- Tools
  - Benchmark programs
  - Simulators, emulators
  - Register Transfer Notation

Logic Designer’s View

- Implementation domain
  - set of hardware devices.
  - Ex., Logic gate implementation domain – VLSI on silicon, TTL, PLAs, etc.
Logic Designer’s Tools

- Computer aided design tools
  - Logic design and simulation packages
  - Printed circuit layout tools
  - IC (integrated circuit) design and layout tools
- Logic analyzers and oscilloscopes
- Hardware development system

Assignments

- Beginning of Course Survey
  - Linked from course web site
  - Submit by Saturday, August 26 by 11:59 pm
- Text: Read Chapter 1
- Be familiar with answers to 1.9, 1.10, 1.13, 1.16, 1.19, 1.21
- Friday: More detail on general purpose machines