

ECE 152 Introduction to Computer Architecture

Intro and Overview
Copyright 2008 Daniel J. Sorin
Duke University

Slides are derived from work by
Amir Roth (Penn) and Alvy Lebeck (Duke)
Spring 2008

Instructor

- Professor: Daniel J. Sorin
 - Office: Hudson Hall 209C
 - Email: sorin@ee.duke.edu
 - subject of all emails to me must begin with ECE152
 - Office Hours: TBD
 - If you have class at both of these times, email me to meet
- No graduate TA for this course

Undergrad Teaching Assistants

- Undergraduate TAs (UTAs):
 - Mike Bauer (meb26@ee.duke.edu)
 - Tuesdays, 10:30-11:30am @ Teer basement
 - Pat Eibl (pje2@ee.duke.edu)
 - TBD
 - Phillip Ethier (pde@ee.duke.edu)
 - TBD
 - Andrew Waterman (asw9@ee.duke.edu)
 - Mondays 4:15-5:30pm @ 202A Hudson
- Will help with
 - Answering email questions about homeworks and project
 - Holding office hours to help with CAD software
- Will NOT bail you out at 3am when deadline is at 10am

Course Website

- Course Web Page
<http://www.ee.duke.edu/~sorin/ece152/>
- Lecture slides available on web before or shortly after class
 - Print them out and bring them with you to class
 - Value (just reading slides) << Value (attending class)
 - Missing class = missing important course material
- Most important info for course is on website
 - Please check it before emailing me or TAs
- You are required to monitor web page
 - Homework and project assignments will appear on web page
 - I will announce them as well

Google Group for ECE 152

- I will communicate with you via email
 - You must check your email
- I have also set up a Google group and invited all of you
 - And the TAs!
 - If you have questions about homeworks or project parts, please post them there – then everyone can see the answer(s) posted there by me, a TA, or your fellow classmate

Textbook

- Text: *Computer Organization & Design* (Patterson & Hennessy)
 - 3rd edition of the textbook
 - You are expected to complete the assigned readings
- We will not cover chapters in the textbook in a strictly linear fashion

Workload

- Readings from textbook
- Homework assignments (performed in groups of 2)
 - Pencil and paper problems
 - Small programming problems
- Project (performed in groups of 2)
 - Building the Duke152/16 computer in real hardware!
 - Programming the Duke152/16 computer you built
 - You will choose project partners, and I will ensure that group grading is done fairly by end-of-semester questionnaires
 - Project will be broken up into smaller parts to make it more manage-able
 - Project Part 1: Building a register file and finite state machine, due: Weds, Jan 23

Grading

- Grade breakdown
 - Homework 30%
 - Project 25% (graded as a group, but fairly)
 - Midterm Exam 20%
 - Final Exam 25%
- I strongly believe in partial credit
 - Please explain your answers to get as much credit as possible
- Late homework policy
 - 10% reduction for each day late
 - No credit after the homework is graded and handed back
- Assignments take a lot of time, so start them early
 - Yes, this means you! And you and you and especially you.

Academic Misconduct

- Academic Misconduct
 - Refer to Duke Honor Code (link to it off course website)
 - Studying together in groups is encouraged
 - Homework and project must be in groups of 2
 - You must choose a different partner for each homework
 - You will choose one partner for all parts of project
 - Common examples of cheating:
 - Running out of time and using someone else's output
 - Borrowing code from someone who took course before
 - Using solutions found on the Web
 - Person asks to borrow solution "just to take a look"
- **I will not tolerate any academic misconduct!**
 - Historically, this course has "led the league" in cases of academic misconduct that have led to suspensions and expulsions
 - Software for detecting cheating is very, very good ... and I use it

Goals of This Course

- By end of semester, you will
 - Know how computers work
 - What's inside a computer?
 - How do computers run programs written in C, C++, Java, Matlab, etc.?
 - Design your own computer, program it, and build it in real hardware!
 - Understand the engineering tradeoffs to be made in the design and implementation of different types of computers
- If, at any point, it's not clear why I'm talking about some topic, please ask!

Outline of Introduction

- Administrivia
- **What is a computer?**
- What is computer architecture?
- Why are there different types of computers?
- How do we tell computers what to do?

Reading Assignment

- Patterson & Hennessy
 - Chapter 1
 - This is a short and relatively easy-to-read chapter
 - **Please read it such that afterwards you'd feel comfortable teaching the material to an ECE 52 student**
- For those of you who haven't done digital logic design in a while, you may want to go back to your ECE 52 textbook or skim Appendix B of Patterson & Hennessy
 - Digital logic design is a pre-requisite for this course, but I understand if some of you haven't done this in a while
 - You should be able to design combination logic and finite state machines – otherwise, you're going to have a very rough semester

What is a Computer?

- A computer is just a digital system
 - Consists of combinational and sequential logic
 - One big, honking finite state machine
 - A computer would be a very exciting ECE 52 project
- Seriously, it's just a digital system
- Yes, but what does this digital system do?
 - Whatever you tell it to do! No more, no less
- A computer just does what software tells it to do
 - Software is a series of **instructions**
- **ICQ (In-Class Question): What instructions does a computer need?**

Computers Execute Instructions

- What kinds of instructions are there?
 - Arithmetic: add, subtract, multiply, divide, etc.
 - Read/write from memory
 - Conditional: if condition, then jump to other part of program
 - What other kinds of instructions might be useful?
- How do we represent instructions?
 - Digitally! With strings of zeros and ones
 - **Remember: a computer is just a digital system!**
- So how do computers run programs in Java or C/C++ or Matlab or whatever whippersnappers use these days?
 - None of us write programs in binary (zeros and ones) ...
 - We'll get to this in a few minutes

Instruction Sets

- Computers can only execute instructions that are in their specific machine language
- Every **type** of computer has a different **instruction set** that it understands
 - Intel (and AMD) IA-32 (x86): Pentium4, Pentium CoreDuo, AMD Athlon and Opteron, AMD Dual Core, etc.
 - Intel IA-64: Itanium, Itanium 2
 - PowerPC: In Cell Processor and old Apple Macs
 - SPARC: In computers from Sun Microsystems
 - MIPS: MIPS R10000 → **this is the example used in the textbook**
- Note: no computer executes Java or C++
 - Not even Matlab

Outline of Introduction

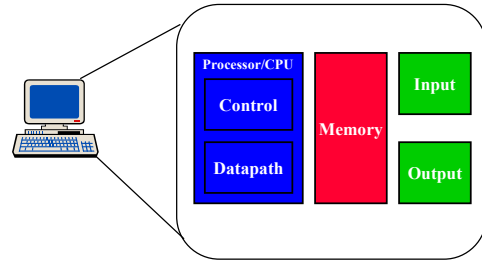
- Administrivia
- What is a computer?
- **What is computer architecture?**
- Why are there different types of computers?
- How do we tell computers what to do?

Hint: It Doesn't Involve Skyscrapers ...

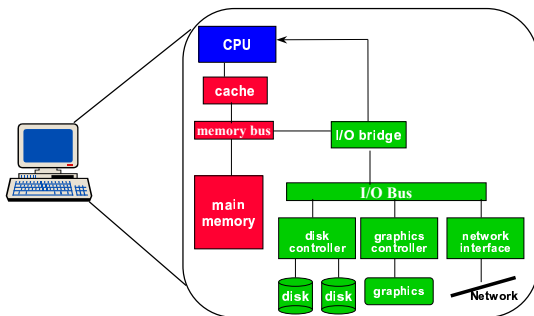
- Strictly speaking, a **computer architecture** specifies what the hardware looks like (its interface), so that we can write software to run on it
 - Exactly what instructions does it have
 - Number of register storage locations it has
 - And more that we'll learn about later in semester
- Important point: there are many, many different ways to build digital systems that provide the same interface to software
 - There are many **microarchitectures** that conform to same architecture
 - Some are better than others! If you don't believe me, I'll trade you my original Intel Pentium for your Intel Core2
- **ICQ: So what's inside one of these digital systems?**

The Inside of a Computer

- The Five Classic Components of a Computer

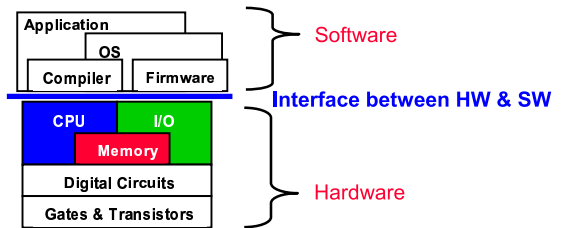


System Organization



What Is ECE 152 All About?

- Architecture = interface between hardware and software



- ECE 152 = design of CPU, memory, and I/O

Outline of Introduction

- Administrivia
- What is a computer?
- What is computer architecture?
- Why are there different types of computers?
- How do we tell computers what to do?

Differences Between Computers

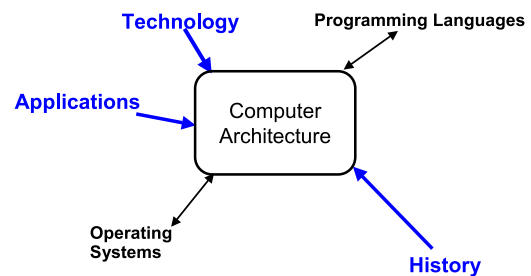
- We have different computers for different purposes
- Some can achieve performance needed for high-performance gaming
 - E.g., Cell Processor in Xbox
- Others can achieve decent enough performance for laptop without using too much power
 - E.g., Intel Pentium M (for Mobile)
- And yet others can function reliably enough to be trusted with the control of your car's brakes

ICQ: What computers do you use?
ICQ: Which of those computers do you own?

Kinds of Computers

- "Traditional" personal computers
 - Laptop, desktop
- Less-traditional personal computers
 - iPhone, Blackberry, iPod, Xbox, Wii, etc.
- Hidden "big" computers
 - Mainframes and servers for business, science, government
 - Google has tens of thousands of computers (that you don't see)
- Hidden embedded computers
 - Controllers for cars, airplanes, ATMs, toasters, DVD players, etc.
 - Far and away the largest market for computers!
- Other kinds of computers??

Forces on Computer Architecture



Outline of Introduction

- Administrivia
- What is a computer?
- What is computer architecture?
- Why are there different types of computers?
- How do we tell computers what to do?