Introduction

CS 211: Computer Architecture
Fall 2020
Instructor: Jeff Ames

TAs:

- Priya Parikh
- Wenjie Qiu
- Yuwei Jin
- Gautam Singh
- Abu Awal Md Shoeb
- Kunpeng Song
- Fei Hua

Grader: TBA

Office hours: TBA
Web:

• Canvas
• Piazza

ilab
Textbooks

- Computer Systems: A Programmer’s Perspective
- Modern C
- The C Programming Language
Prereqs

- some math
- some algorithms and data structures
- one programming language (Java)
What you’ll learn

- Two more programming languages
  - C
  - Assembly
- Major hardware components in computer systems
- How hardware components are built from digital logic
- How programs are actually executed by the hardware
- The performance of programs
Expectations

- 7 programming assignments
- 6 quizzes
Expectations

• Attend lectures and recitations
• Read the assigned readings before lecture
• Read and think about the programming and homework assignments
• Ask questions
Expectations

- Start programming assignments early
- Don’t copy or cheat
- Late assignments will not be accepted
- Programming assignments to be handed in on Canvas
- Can hand in assignments multiple times
Collaboration

You learn by discussing with others.

But assignments should be your own work.

Department’s academic integrity policy: https://www.cs.rutgers.edu/academics/undergraduate/academic-integrity-policy

If you are having trouble with the course for any reason, come talk to us.
• Introduction
• C programming
• Information representation
• Assembly (x86) programming
• Memory hierarchy
• Digital logic
• Processor architecture
• 7 programming assignments
• Program in C and/or Assembly
  • Don’t wait until the last minute
  • Learn how to use tools
  • Don’t program/debug “by accident” or “by blind search”
• Will be done using the Instructional Lab
  • https://resources.cs.rutgers.edu/docs/computer-systems/student-systems/
Grading

- 700 points: programming assignments
- 300 points: quizzes
Why study architecture?

- Security
- Performance
- Understanding program execution
Main components

- CPU
- Memory
- Bus
- I/O devices
  - Mouse, keyboard, screen
  - Storage
  - Network
  - Graphics
Von Neumann model

Memory

Store both instructions and data
CPU function

FETCH[PC++]

DECODE

EXECUTE
```c
#include <stdio.h>
int main() {
    int x, y, temp;
    x=1; y=2;
    temp =x; x=y; y=temp;
    printf("%d %d %d\n",x,y,temp);
}
```

```
movl $1, -8(%ebp)
movl $2, -12(%ebp)
movl -8(%ebp), %eax
movl %eax, -16(%ebp)
movl -12(%ebp), %eax
movl %eax, -8(%ebp)
movl -16(%ebp), %eax
movl %eax, -12(%ebp)
movl -16(%ebp), %eax
movl %eax, 12(%esp)
movl -12(%ebp), %eax
movl %eax, 8(%esp)
movl -8(%ebp), %eax
movl %eax, 4(%esp)
```

```
7f454e46010101
00000000000000
00000200030001
00000000820408
34000000c40c00
000000003400
```
Von Neumann details

Diagram showing CPU components including PC, Register file, ALU, Bus interface, I/O bridge, Main memory, System bus, Memory bus, USB controller, Graphics adapter, Disk controller, Mouse, Keyboard, Display, Disk, I/O bus, Expansion slots for other devices such as network adapters.
• How to design and build the components
• How to design and build systems from the components
• In this class:
  • Understand basics of current components and systems
  • Understand how programs run on current systems
  • Understand how current architecture affect my high-level language programs
  • How can I make my program run faster?
Getting started

- set up your ilab account
- learn how to use the command-line
• cd: change directory
• ls: list
• mkdir: make directory
• rm: remove
• mv: move/rename
• pwd: show current directory