Computer Systems Organization

Jinyang Li

Slides are based on Tiger Wang’s class
Why study CSO?
Your career path in the next few years

graduation

interview

programmer
Your career path in the next few years

- Graduation
- Interview
- Apply for graduate school
- Programmer
- Programmer & researcher
Your career path in the next few years

graduation

interview

graduate school

startup

programmer & researcher

programmer

Hire / become
Your career path in the next few years

Be able to
Hire / become

programmer

programmer & researcher

start-up

lawyer

graduate school

interview

graduation
Your career path in the next few years

- graduation
- interview
- graduate school
- programmer & researcher
- programmer
- startup
- lawyer
- enjoy life

Be able to
Hire / become

Your career path in the next few years...
Your career path in the next few years

graduation

interview

graduate school

programmer & researcher

Hire / become

Be able to

Enjoy life

Be in a relationship with

programmer

startup

lawyer
Taking CSO will affect each step in the path!
For Graduation

Required class
  – For CS Major
  – Also for CS minor 😞

Prepare for your later system classes
  – Operating Systems, Compilers, Networks, Computer Architecture, Distributed Systems
For Interview

This class adds to your CV
– C Programming, UNIX, X86 Assembly …

Interview related topics
– Basic knowledge of Array, String, Bit Manipulation
Some exercises in this class are derived from the real interview questions!

Our text books are considered as the bibles of job interview.
For Graduate School Application

This class adds to your CV

- A

Research related topics

- Performance optimization
  - Memory layout, code optimization, memory allocation, concurrent programming

- Security
  - Buffer Overflow
Startup
The life you imagine
CEO
CTO
CFO
COO
Startup

Your real life: full stack programmer

Server
Website
Phone’s App
Optimizations
My lawyer friend

Take >10 hours each day to extract information from the documents
My lawyer friend

I want to study programming.
My lawyer friend

I want to study programming.

Ok, you need to study CSO first.
My lawyer friend

I want to study programming.

Ok, you need to study CSO first.

Hmm..., I want to marry a programmer.
My lawyer friend

I want to study programming.

Ok, you need to study CSO first.

Hmm…, I want to marry a programmer.

Ok, you need to study CSO first.
My lawyer friend

I want to study programming.
Ok, you need to study CSO first.

Hmm…, I want to marry a programmer.

Ok, you need to study CSO first.

…The user is offline
Have you heard of the Meltdown attack?

No. Is it bad?

Meltdown lets an attacker read another process’ address space!

What is an address space?

He does not know anything about computers...

Sorry I have to run now, bye!
For Programming

Understand how your program runs on the hardware

– Why it fails
– Why it is slow
Why it fails?

What is the result of 1000,000 * 1000,000?
Why it fails?

What is the result of 1000,000 * 1000,000 ?

Expected answer: 1000,000,000,000 (1 trillion)
Why it fails?

What is the result of 1000,000 * 1000,000 ?

Expected answer: 1000,000,000,000 (1 trillion)

```c
int main()
{
    int a = 1000000;
    int b = 1000000;
    int r = a * b;
    printf("result is %d\n", r);
    return 0;
}
```
Why it is slow?

Example Matrix Multiplication

Both implementations have exactly the same operations count ($2n^3$)

Throughput

Best code (K. Goto)

Triple loop

160x

matrix size
What is CSO about?
Computer System Organization

- System Fan
- Floppy
- Heat Sink
- Power Supply
- Hard Drive
- Optical Drive
- Motherboard
- Processors (CPU)
- RAM Modules
Computer System Organization

- System Fan
- Power Supply
- Motherboard
- Processors (CPU)
- RAM Modules
- Floppy
- Heat Sink
- Hard Drive
- Optical Drive

Printed Circuit
Computer System Organization

- System Fan
- Floppy
- Blackberry
- Hard Disk
- Printed Circuit
- Power Supply
- Adobe
- Optical Drive
- talk
- Motherboard
- Processor (CPU)
- RAM Modules
- DirectX
- Microsoft
Layered Organization

Software

Hardware
Layered Organization

**Software**

**Hardware**

Transistors  Diodes  Resistors
Layered Organization

**Software**

**Hardware**

- Logical Circuits, Flip-Flops, Gates
- Transistors
- Diodes
- Resistors
Layered Organization

**Software**

**Hardware**

CPU, Memory, Disk

Logical Circuits, Flip-Flops, Gates

Transistors, Diodes, Resistors
Layered Organization

Hardware

Software

CPU
Memory
I/O

Logical Circuits, Flip-Flops, Gates, ...

Transistors, Diodes, Resistors, ...
Layered Organization

**Hardware**
- Transistors, Diodes, Resistors, …

**Software**
- System Software (OS, compiler, VM…)
- Logical Circuits, Flip-Flops, Gates, …
Layered Organization

User Applications

System Software (OS, compiler, VM…)

Software

Hardware

CPU

Memory

I/O

Logical Circuits, Flip-Flops, Gates, …

Transistors, Diodes, Resistors, …
Layered Organization

User Applications

System Software

Software

Hardware

- CPU
- Memory
- I/O

- Logical Circuits, Flip-Flops, Gates, …
- Transistors, Diodes, Resistors, …

User App

Operating System

Compilers
Abstraction

User Applications

System Software

Software

Hardware

Abstract Interface

Transistors, Diodes, Resistors, …

Logical Circuits, Flip-Flops, Gates, …

I/O

Memory

CPU

Operating System

Compilers
The Scope of This Class

Hardware
- Transistors
- Diodes
- Resistors
- Logical Circuits
- Flip-Flops
- Gates
- ...
The Scope of This class

Focus on abstract interfaces exposed by

– CPU and Memory
– Operating System, Compilers

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Assembly, Virtual memory, Interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Software</td>
<td>C Programming, OS Service, Memory Management, Concurrent Programming</td>
</tr>
<tr>
<td>Software</td>
<td>Operating Systems and Compilers</td>
</tr>
<tr>
<td></td>
<td>CPU and Memory</td>
</tr>
</tbody>
</table>
Schedule

http://news.cs.nyu.edu/~jinyang/fa18-cso/schedule.html

overview
bit, byte and int
float point
[C] basics, bitwise operator, control flow
[C] scopes rules, pointers, arrays
[C] structs, mallocs
[C] large program (linked list)

C Programming
Schedule

http://news.cs.nyu.edu/~jinyang/fa18-cso/schedule.html

overview
bit, byte and int
float point
[C] basics, bitwise operator, control flow
[C] scopes rules, pointers, arrays
[C] structs, mallocs
[C] large program (linked list)
Machine Prog: ISA, Compile, movq
Machine Prog: Control Code (condition, jump instruction)
Machine Prog: Array allocation and access
Machine Prog: Procedure calls
Machine Prog: Structure, Memory Layout
Machine Prog: Buffer Overflow
Code optimizations
Schedule

http://news.cs.nyu.edu/~jinyang/fa18-cso/schedule.html

Overview
- bit, byte and int
- float point
- [C] basics, bitwise operator, control flow
- [C] scopes rules, pointers, arrays
- [C] structs, mallocs
- [C] large program (linked list)

C Programming
- Machine Prog: ISA, Compile, movq
- Machine Prog: Control Code (condition, jump instruction)
- Machine Prog: Array allocation and access
- Machine Prog: Procedure calls
- Machine Prog: Structure, Memory Layout
- Machine Prog: Buffer Overflow
- Code optimizations

Assembly (X86)

Virtual Memory
- Virtual memory: Address Spaces/Translation, Goal
- Virtual memory: Page table/physical to virtual
- Process
http://news.cs.nyu.edu/~jinyang/fa18-cso/schedule.html

overview
bit, byte and int
float point
[C] basics, bitwise operator, control flow
[C] scopes rules, pointers, arrays
[C] structs, mallocs
[C] large program (linked list)

Machine Prog: ISA, Compile, movq
Machine Prog: Control Code (condition, jump instruction)
Machine Prog: Array allocation and access
Machine Prog: Procedure calls
Machine Prog: Structure, Memory Layout
Machine Prog: Buffer Overflow
Code optimizations
Virtual memory: Address Spaces/Translation, Goal
Virtual memory: Page table/physical to virtual
Process
Dynamic Memory Allocation I: malloc, free
Dynamic Memory Allocation II: design allocator
Dynamic Memory Allocation III: further optimization
Schedule

http://news.cs.nyu.edu/~jinyang/fa18-cso/schedule.html

overview
bit, byte and int
float point
[C] basics, bitwise operator, control flow
[C] scopes rules, pointers, arrays
[C] structs, mallocs
[C] large program (linked list)
Machine Prog: ISA, Compile, movq
Machine Prog: Control Code (condition, jump instruction)
Machine Prog: Array allocation and access
Machine Prog: Procedure calls
Machine Prog: Structure, Memory Layout
Machine Prog: Buffer Overflow
Code optimizations
Virtual memory: Address Spaces/ Translation, Goal
Virtual memory: Page table/physical to virtual
Process
Dynamic Memory Allocation I: malloc, free
Dynamic Memory Allocation II: design allocator
Dynamic Memory Allocation III: further optimization
Concurrent Programming I: thread, race
Concurrent Programming II: lock
Concurrent Programming III: conditional variable
Concurrent Programming IV: Other primitives
Course Perspective

Most Systems Courses are Builder-Centric

– Computer Architecture
  • Design pipelined processor in Verilog

– Operating Systems
  • Implement large portions of operating system

– Compilers
  • Write compiler for simple language

– Networking
  • Implement and simulate network protocols
Course Perspective

Most Systems Courses are Builder-Centric

– Computer Architecture
  • Design pipelined processor in Verilog
– Operating Systems
  • Implement large portions of operating system
– Compilers
  • Write compiler for simple language
– Networking
  • Implement and simulate network protocols
This course is **programmer-centric**

– Understanding of underlying system makes a more effective programmer
– Bring out the hidden hacker in everyone
To be a happy programmer, you should

Attend
- Lectures (M/W 3:30-4:45pm) Wav 366
- Recitation (Th 12:30-1:45 pm) CIWW 312
  • In-class exercises provide hands-on instruction

Do
- 5 Programming labs
- Recitation exercises

Pass
- Quiz 1 (10/3)
- Quiz 2 (10/31)
- Final exam

Starts tomorrow! Bring Laptop to class

Due on Thu 11pm.
Grade Breakdown

Recitation and Exercises 15%
Labs 40%
Quiz-1 10%
Quiz-2 15%
Final 20%

Bonus I: lecture and piazza participation 5%
Bonus II: extra-credit lab questions (points vary)
Is CSO going to be hard?
Statistics from last term: quiz 1
Statistics from last term: final
Statistics from this term: major vs. minor

- CS: 60.5%
- Undecided: 26.3%
- Others: 13.2%

- 23 CS majors
- 10 undecided
- 5 others
Statistics from this term:
student level

- Freshman: 5.3%
- Sophomore: 36.8%
- Junior: 50.0%
- Senior: 7.9%
Time to work hard

We (the course staff) are here to help
We are here to help

Jinyang Li
Lecturer

Lingfan Yu
Recitation Leader
Head grader

Jingyu Liu
Grader

cso-staff@cs.nyu.edu
Before Class

Read the related sections in the text books

http://csapp.cs.cmu.edu

Reserved at NYU library
Be Active In Class

Raise your hand at any time
  – Ask me to repeat, repeat and repeat
  – Ask questions
  – Answer questions from me or others

Have discussion and make friends with each others
After Class

Finish all labs / exercises
  – By yourself

Attend the recitations
  – Any issue of doing labs or exercises

Getting help
  – Office hour, Piazza
Policies

You must work alone on all assignments
  – You may post questions on Piazza,
  – You are encouraged to answer others’ questions, but refrain from explicitly giving away solutions.

Labs & Exercises
  – Assignments due at 11:00pm on the due date
  – Everybody has 5 grace days (incremented at half day granularity)
  – Zero score after the due
Class Info

http://news.cs.nyu.edu/~jinyang/fa18-cso/

Recitation starts tomorrow, bring your laptop to class
Integrity and Collaboration Policy

We will enforce the policy strictly.

1. The work that you turn in must be yours
2. You must acknowledge your influences
3. You must not look at, or use, solutions from prior years or the Web, or seek assistance from the Internet
4. You must take reasonable steps to protect your work
   – You must not publish your solutions
5. If there are inexplicable discrepancies between exam and lab performance, we will over-weight the exam and interview you.

Do not turn in labs/exercises that are not yours
You won’t fail because of one missing lab
Integrity and Collaboration Policy

We will enforce this policy strictly and report violators to the department and Dean.