Computer Architecture

Yipeng Huang

Rutgers University

January 18, 2022
Table of contents

Curiosity
  Computer science
  Computer systems

Community
  The students
  The instructors

Learning
  Preview of syllabus
  Preview of assignments

Expectations
  Accessing the class & resources
  Lecture and short quizzes
  Programming assignments
  Recitation code review study groups
  Academic honesty and integrity

A New Golden Age for Computer Architecture
Warm up questions

1. What are some fundamental concepts in computer science?
2. How does a computer work?

▶ slido.com
▶ #342433
What this class is about: abstractions

Intermediate courses in computer science:

**CS 112: Data structures**
learned about how to store data and manipulate data with algorithms

**CS 205/206: Discrete structures**
learn about the discrete and continuous mathematics governing computer science

**CS 211: Computer architecture**
learn about the abstractions that make programs run on computer building blocks

**CS 213: Software methodology**
learn how to organize complex programs

**CS 214: Systems programming**
learn how to interact with the operating system
What are the parts of a computer?

1. What are the parts of a computer?
   ▶ slido.com
   ▶ #342433
What are the parts of a computer?

- Central Processing Unit
- Registers
- Caches
- Memory
- File Systems
- Network
- Screen
- User interfaces
- GPU
- FPGA
- ASIC
- Circuit boards
- Chips
- Integrated circuits
- Logic gates
- Transistors
What are desirable properties for computers?

1. What are desirable properties for computers?
What are desirable properties for computers?

- Correctness
- Performance
- Efficiency
- Security and Privacy
- Fairness
- Positively impacts human condition
Important computing abstractions

Abstractions
A way to hide the details of an underlying system so you (users & programmers) can be more creative.

Low-level programming
C, assembly language, machine code, instruction set architecture

The memory hierarchy
File system, main memory, caches, data representations

Digital logic
Pipelines, registers, flip flops, arithmetic units, gates
Table of contents

Curiosity
   Computer science
   Computer systems

Community
   The students
   The instructors

Learning
   Preview of syllabus
   Preview of assignments

Expectations
   Accessing the class & resources
   Lecture and short quizzes
   Programming assignments
   Recitation code review study groups
   Academic honesty and integrity

A New Golden Age for Computer Architecture
The students

Take a look around to meet your fellow students.

As of today, 226 registered students

- ≈ 10 seniors, ≈ 55 juniors, ≈ 100 sophomores, ≈ 55 first-years
- ≈ 10 BAIT, ≈ 40 CS, ≈ 10 finance, ≈ 5 ITI, ≈ 10 mathematics, ≈ 130 undeclared, ≈ 10 prebusiness
Welcome all to class

- We welcome in this class diverse backgrounds and viewpoints spanning various dimensions: race, national origin, gender, sexuality, disability status, class, religious beliefs.
- We will treat each other with respect and strive to create a safe environment to exchange questions and ideas.
The instructors

Prof. Yipeng Huang
yipeng.huang@rutgers.edu

Teaching assistants

- Song Wen (PhD TA)
- Naishal Patel (MS recitation PTL)
- Eshaan Gandhi (undergraduate recitation PTL)

I am expecting one more recitation PTL and two graders to be joining the team soon.
My research is in abstractions that allow us to use novel computer architectures such as quantum and analog computers.

- I am looking for students who want to pursue research projects.
- In the fall I teach CS 583—Quantum Computing: Programs and Systems
- Worked with DARPA to investigate feasibility of using analog electronic circuits for scientific computation.
Table of contents

Curiosity
   Computer science
   Computer systems

Community
   The students
   The instructors

Learning
   Preview of syllabus
   Preview of assignments

Expectations
   Accessing the class & resources
   Lecture and short quizzes
   Programming assignments
   Recitation code review study groups
   Academic honesty and integrity

A New Golden Age for Computer Architecture
What this class is about

Course objective
Sustain and enhance your (the student’s) interest and confidence in computer science.

Specific learning goals
Throughout the course, students will learn about important computing abstractions such as low-level programming, the memory hierarchy, and digital logic via case studies that are representative of real-world computer systems.
Low-level programming: C

Learn a new and foundational programming language.

```c
int main() {
    printf("Hello, World!");
    return 0;
}
```
Data representations

```c
void show_squares()
{
    int x;
    for (x = 5; x <= 500000; x += 10)
        printf("x = %d x^2 = %d\n", x, x*x);
}
```

<table>
<thead>
<tr>
<th>x</th>
<th>x^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>50</td>
<td>2500</td>
</tr>
<tr>
<td>500</td>
<td>250000</td>
</tr>
<tr>
<td>5000</td>
<td>2500000</td>
</tr>
<tr>
<td>50000</td>
<td>-1794967296</td>
</tr>
<tr>
<td>500000</td>
<td>891896832</td>
</tr>
<tr>
<td>5000000</td>
<td>-1004630016</td>
</tr>
</tbody>
</table>

- Numbers are represented using a finite word size
- Operations can overflow when values too large
  - But behavior still has clear, mathematical properties

Figure: Credit: Computer Systems: A Programmer’s Perspective
Low-level programming: assembly

Study the interface between software and hardware.

MOV [ESI+EAX], CL ; Move the contents of CL into the byte at address ESI+EAX
MOV DS, DX ; Move the contents of DX into segment register DS
The memory hierarchy

**Computer Memory Hierarchy**

- **Small size, small capacity**
  - Power on
  - Immediate term
- **Random access memory**
  - Fast, affordable

- **Medium size, medium capacity**
  - Power on
  - Very short term
- **Flash/USB memory**
  - Slower, cheap

- **Small size, large capacity**
  - Power off
  - Short term
- **Hard drives**
  - Slow, very cheap

- **Large size, very large capacity**
  - Power off
  - Mid term
- **Tape backup**
  - Very slow, affordable

**Figure: Credit: wikimedia**
The memory hierarchy

```c
void copyji(int src[2048][2048],
            int dst[2048][2048])
{
    int i, j;
    for (j = 0; j < 2048; j++)
        for (i = 0; i < 2048; i++)
            dst[i][j] = src[i][j];
}
```

4.3 ms  
81.8 ms

19 times slower!

(Measured on 2 GHz Intel Core i7 Haswell)

- Hierarchical memory organization
- Performance depends on access patterns
  - Including how step through multi-dimensional array

Figure: Credit: Computer Systems: A Programmer’s Perspective
Preview of assignments

Students will apply essential knowledge about computer systems to modify and create new low-level software and hardware implementations via hands-on programming exercises.

1. A new language: C
2. Review of data structures and algorithms in C
3. Everything is a number, numbers are bits and bytes
4. How programs are represented and run in computers
5. How to create the illusion of fast and big memory
6. How to build computers from simple logic
Table of contents

Curiosity
  Computer science
  Computer systems

Community
  The students
  The instructors

Learning
  Preview of syllabus
  Preview of assignments

Expectations
  Accessing the class & resources
  Lecture and short quizzes
  Programming assignments
  Recitation code review study groups
  Academic honesty and integrity

A New Golden Age for Computer Architecture
Accessing the class & resources

Canvas
Announcements, lecture slides, videos, quizzes, assignments, submissions.
https://rutgers.instructure.com/courses/160141

Textbooks
- Modern C: https://gustedt.gitlabpages.inria.fr/modern-c/
Lecture and short quizzes

Lectures
- First four sessions are virtual. Hybrid in-person and livestream in February. If overall attendance and engagement suffers, we would go in-person only if safe and permitted.
- It benefits your learning to attend live, ask questions, and keep up.
- Videos will be posted within one day of lecture to YouTube, access link on Canvas.

Short quizzes
- Ensure that you keep up with the class and check on basic concepts from previous week, and to collect feedback.
- 30 minutes for quiz, max two attempts, open for set time window.
Programming assignments

ilab

▶ All students need access to ilab to compile, run, and test programs in Linux.
▶ If you do not have access, sign up immediately: https://services.cs.rutgers.edu/accounts/activate/activate

Piazza

▶ Ask all questions if possible on Piazza.
▶ If you send the instructor or any of the TAs an email that is better addressed on Piazza, we will kindly ask you to repost your question on Piazza and we will answer it there.
▶ Sign up now: https://piazza.com/class/kydc7isau027cs
Programming assignments

Automatic compiling, testing, and grading

- It is important that you carefully follow the specified output formats so that the testing framework can validate your program.
- New for this class: more incremental points, stricter validation, and more feedback from the grading system.

Submit on Canvas

- Start early.
- You can submit as many times as you wish.
- We will not accept late assignments; deadline will be enforced by Canvas.
Recitation code review study groups

Goals
- Give stylistic code review and feedback (avoid coding to satisfy autograder).
- Boost recitation attendance and give structure to recitation sections.

Mechanics
- Teams of \( \approx 5 \) students.
- Review and discuss code from previous assignment.
- As a team, present findings in 5 minute short summary.
- Recitation TAs have full discretion to award a portion of assignment grades for participating.
Importance of writing your own code

### EFFECTIVE SORTS

**DEFINE HALFHEARTED_MERGESORT(LIST):**
```plaintext
IF LENGTH(LIST) < 2:
    RETURN LIST

P = INT((LENGTH(LIST) / 2))
A = HALFHEARTED_MERGESORT(LIST[:P])
B = HALFHEARTED_MERGESORT(LIST[P:])
// UMMMNNN
RETURN[A, B] // HERE. SORRY.
```

**DEFINE FAST_BOGOSORT(LIST):**
```plaintext
// AN OPTIMIZED BOGOSORT
// RUNS IN O(NLOG(N))
FOR N FROM 1 TO LOG(LENGTH(LIST)):
    SHUFFLE(LIST);
    IF ISORTED(LIST):
        RETURN LIST
RETURN “KERNEL PAGE FAULT (ERROR CODE: 2)"
```

---

### INEFFECTIVE SORTS

**DEFINE JOE_BORKED_QUICKSORT(LIST):**
```plaintext
OK SO YOU CHOOSE A PIVOT
THEN DIVIDE THE LIST IN HALF
FOR EACH HALF:
    CHECK TO SEE IF IT'S SORTED
    NO WAY, IT DOESN'T MATTER
    COMBINE ELEMNT TO THE PIVOT
    THE BIGGER ONES GO IN A NEW LIST
    THE SMALLER ONES GO INTO UM
    THE SECOND LIST FROM BEFORE
    HANG ON, LET ME NAME THE LISTS
    THIS IS LIST A
    THE NEW ONE IS LIST B
    PUT THE BIG ONES INTO LIST B
    NOW TAKE THE SECOND LIST
    CALL IT LIST, UM, A2
    WHICH ONE WAS THE PIVOT IN?
    SCRATCH ALL THAT
    IT JUST RECURSIVELY CALLS ITSELF
    UNTIL BOTH LISTS ARE EMPTY
    RIGHT?
    NOT EMPTY, BUT YOU KNOW WHAT I MEAN
    AM I ALLOWED TO USE THE STANDARD LIBRARIES?
```

**DEFINE PROMISSE(LIST):**
```plaintext
IF ISORTED(LIST):
    RETURN LIST
FOR N FROM 1 TO 10000:
    PIVOT = RANDOM(0, LENGTH(LIST))
    LIST = LIST[PIVOT:]+LIST[:PIVOT]
    IF ISORTED(LIST):
        RETURN LIST
    IF ISORTED(LIST):
        RETURN LIST
    IF ISORTED(LIST):
        // THIS CAN'T BE HAPPENING
        RETURN LIST
    IF ISORTED(LIST):
        // COME ON COME ON
        RETURN LIST
    // OH JEEZ
    // I'M GONNA BE IN SO MUCH TROUBLE
    LIST = []
    SYSTEM("SHUTDOWN -H +5")
    SYSTEM("RM -RF ./")
    SYSTEM("RM -RF ")+"
    SYSTEM("RM -RF .")
    SYSTEM("RD /S /Q O:*") // PORTABILITY
    RETURN [1, 2, 3, 4, 5]
```
Academic honesty and integrity

Study and practice programming to learn

▶ You are encouraged to discuss the homework with your classmates on Piazza.
▶ You are encouraged to research and study concepts online.

Importance of writing your own code

▶ But, you must not disclose your code or see your classmates’ code.
▶ You cannot look at answers online that are obviously specific to this class.
▶ Finding your own solution and writing and debugging your own code is vital to your learning. Copying someone else’s code short-circuits this process.
▶ We will use automatic tools to detect identical or similar submissions.

Rutgers Academic Integrity Policy

▶ https://nbprovost.rutgers.edu/academic-integrity-students
▶ Every offense will be reported to office of student conduct.
Table of contents

Curiosity
  Computer science
  Computer systems

Community
  The students
  The instructors

Learning
  Preview of syllabus
  Preview of assignments

Expectations
  Accessing the class & resources
  Lecture and short quizzes
  Programming assignments
  Recitation code review study groups
  Academic honesty and integrity

A New Golden Age for Computer Architecture
Learning goal

At the end of this course, students should have the preliminary skills to design and evaluate solutions involving the computer software-hardware interface to address new problems.

1https://cacm.acm.org/magazines/2019/2/234352-a-new-golden-age-for-computer-architecture/fulltext
History: computer architecture abstractions drove digital revolution

<table>
<thead>
<tr>
<th>Year</th>
<th>1940s</th>
<th>1950s</th>
<th>1960s</th>
<th>1970s</th>
<th>1980s</th>
<th>1990s</th>
<th>2000s</th>
<th>2010s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analog continuous-time computing</td>
<td>Analog computers for rocket and artillery controllers.</td>
<td>Analog computers for field problems.</td>
<td>1st transistorized analog computer.</td>
<td>Analog-digital hybrid computers.</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Architectural abstraction milestones**

- Stored program computer.
- Microprogramming.
- Instruction set architecture.
- Reduced instruction set computers.

Transistor scaling and architectural abstractions drive digital revolution, make analog alternatives irrelevant.

**Figure:** Emerging Architectures for Humanity’s Grand Challenges, Yipeng Huang
History: computer architecture abstractions drove digital revolution

Figure: Credit: A New Golden Age for Computer Architecture
Present: power constraints driving diverse computer architectures

Figure: Credit: A New Golden Age for Computer Architecture
Present: power constraints driving diverse computer architectures

<table>
<thead>
<tr>
<th>Year</th>
<th>Analog continuous-time computing</th>
<th>Digital discrete-time computing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1940s</td>
<td>Analog computers for rocket and artillery controllers.</td>
<td>Turing’s <em>Bomba</em>.</td>
</tr>
<tr>
<td>1950s</td>
<td>Analog computers for field problems.</td>
<td>1st transistorized digital computer.</td>
</tr>
<tr>
<td>1960s</td>
<td>1st transistorized analog computer.</td>
<td>Moore’s law projection for transistor scaling.</td>
</tr>
<tr>
<td>1980s</td>
<td>...</td>
<td>VLSI democratized.</td>
</tr>
<tr>
<td>1990s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2000s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2010s</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure:** Emerging Architectures for Humanity’s Grand Challenges, Yipeng Huang

Transistor scaling and architectural abstractions drive digital revolution, make analog alternatives irrelevant.

Scaling challenges drive heterogenous architectures.

Heterogenous architectures

- Cloud FPGAs: Microsoft Catapult, Amazon F1.
- GPUs introduced: Nvidia introduces CUDA.
- CPUs go multicore.
- FPGAs introduced.
- End of Dennard’s scaling.

ASICS: Google TPUs, DE Shaw Research Anton.
Present: a rapidly evolving and influential field of study

Heterogeneity
Multicore CPUS, GPUS, FPGAs, ASICs, TPUs

Energy conservation
Laptop and phone battery life, datacenter energy consumption

Security
Spectre / Meltdown

Virtualization
Docker, Amazon AWS
Present: a rapidly evolving and influential field of study

CS 211 lays foundations for many areas of computer systems and science

- Internet technology
- Security
- Programming Languages and Compilers
- Systems Programming
- Database Implementation
- Operating Systems
- Distributed Systems
- Parallel Programming
Future: post-Moore’s Law computer architectures

- Quantum chemistry & high energy physics
  - Quantum circuit computers

- High dimensional nonlinear optimization
  - Quantum annealers

- Computational neuroscience & pattern recognition
  - Analog neuromorphic networks

- Fluid dynamics & plasma physics
  - Analog continuous-time accelerators

Image sources: Wikimedia.org
Future: post-Moore’s Law computer architectures

Open challenges in emerging architectures:

Problem abstractions
- How do you accurately solve big problems?

Programming abstractions
- Can you borrow ideas from conventional computing?

Architecture abstractions
- How to interface with the unconventional hardware?

Multicore CPUs, GPUs, FPGAs, ASICs, analog, quantum, etc.

New and extreme workload challenges

Limitations in transistor scaling

Nonlinear scientific computation
Quantum simulation & optimization

Dennard’s scaling already ended
Moore’s law increasingly costly to sustain

Image sources: Lanyon and Whitfield et al., 2010