Computer Architecture

Yipeng Huang

Rutgers University

January 16, 2024

Table of contents

- Curiosity
  - Computer science
  - Computer systems
- Community
  - The students
  - The instructors
- Learning
  - Preview of syllabus
- Expectations
  - Accessing the class & resources
  - Lecture and short quizzes
  - Midterm and final exams
  - Programming assignments
  - Academic honesty and integrity
- A New Golden Age for Computer Architecture
- Accessing iLab Linux machines

Warm up questions

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

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 and network

What are the parts of a computer?

1. What are the parts of a computer?

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
- Expectations
  - Accessing the class & resources
  - Lecture and short quizzes
  - Midterm and final exams
  - Programming assignments
  - Academic honesty and integrity
- A New Golden Age for Computer Architecture
- Accessing iLab Linux machines

The students

Take a look around to meet your fellow students.

As of today, 244 registered students. From prior year distributions:

- ≈ 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

- Khyati Doshi (MS recitation PTL)
- Nate Blum (undergraduate recitation PTL)
- Pedro Torres (undergraduate recitation PTL)

Prof. Yipeng Huang

http://cs.rutgers.edu/~yh804

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.
- I also teach CS 558—Quantum Computing: Programs and Systems
- Worked with DARPA to investigate feasibility of using analog electronic circuits for scientific computation.
- PhD Dissertation: Hybrid Analog-Digital Co-Processing for Scientific Computation.

Table of contents

- Curiosity
  - Computer science
  - Computer systems
- Community
  - The students
  - The instructors
- Learning
  - Preview of syllabus
- Expectations
  - Accessing the class & resources
  - Lecture and short quizzes
  - Midterm and final exams
  - Programming assignments
  - Academic honesty and integrity
- A New Golden Age for Computer Architecture
- Accessing iLab Linux machines

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, data representations, 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.

```
#include <stdio.h>

int main() {

    printf("Hello, World!");

    return 0;

}
```

https://www.tiobe.com/tiobe-index/

Data representations

```
void show_squares()
{
  int x;
  for (x = 5; x <= 5000000; x*=10)
    printf("x = %d x^2 = %d\n", x, x*x);
}
```

x	=	5	x^2	=	25
x	=	50	x^2	=	2500
x	=	500	x^2	=	250000
x	=	5000	x^2	=	25000000
x	=	50000	x^2	=	-1794967296
x	=	500000	x^2	=	891896832
x	=	5000000	x^2	=	-1004630016

- 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 EAX, [EBX]    ; Move the 4 bytes in memory at the address cont

MOV [ESI+EAX], CL ; Move the contents of CL into the byte at addr

MOV DS, DX        ; Move the contents of DX into segment register
```

The memory hierarchy

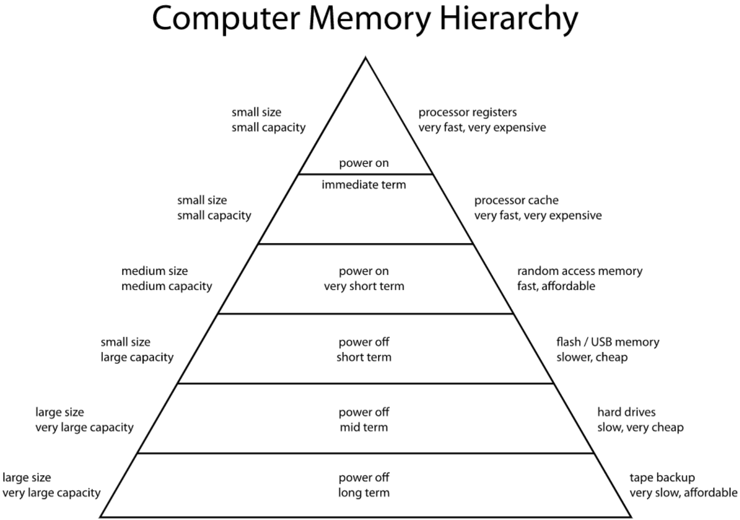


Figure: Credit: wikimedia

The memory hierarchy

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

4.3 ms

```
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];
}
```

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

Table of contents

Curiosity

- Computer science
- Computer systems

Community

- The students
- The instructors

Learning

- Preview of syllabus

Expectations

- Accessing the class & resources
- Lecture and short quizzes
- Midterm and final exams
- Programming assignments
- Academic honesty and integrity

A New Golden Age for Computer Architecture

Accessing iLab Linux machines

Accessing the class & resources

Canvas

Announcements, lecture slides, videos, quizzes, assignments, submissions.

`https://rutgers.instructure.com/courses/261416`

Long-range syllabus

Reading and programming assignment plan.

`https://yipenghuang.com/teaching/2024-spring-211/`

Textbooks

- Modern C: `https://gustedt.gitlabpages.inria.fr/modern-c/`
- Bryant and O'Hallaron. Computer Systems: A Programmer's Perspective. Prentice Hall. 3rd edition.

Lecture and short quizzes

Lectures

- In-person attendance here in SERC Room 111 is expected. If you are sick or have an emergency, use the Zoom livestream and recording.
- It benefits your learning to attend live, ask questions, and keep up.
- Videos will be posted within one day, access link on Canvas.

Short quizzes (5% of course grade)

- 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.

Midterm and final exams

Midterm exam (10% of course grade)

- Tuesday after spring break, March 19, in class session time.
- Will be on C programming and data representations.
- Best way to prepare will be to master the programming assignments and quizzes.

Final exam (15% of course grade)

- Tuesday, May 7 12:00pm to 3:00pm.
- Standard final exam block time assigned by the registrar.
- Cumulative of whole course.
- Best way to prepare will be to master the programming assignments and quizzes.

Programming assignments (70% of course grade)

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

0. Setting up your programming environment (1.25% of course grade)
1. A new language: C (6.25% of course grade)
2. Review of data structures and algorithms in C (12.5% of course grade)
3. Everything is a number, numbers are bits and bytes (12.5% of course grade)
4. How programs are represented and run in computers (12.5% of course grade)
5. How to create the illusion of fast and big memory (12.5% of course grade)
6. How to build computers from simple logic (12.5% of course grade)

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/rutgers/spring2024/01198211050８`

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.

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 ≈ 5 students.
- Review and discuss code from previous assignment.
- As a team, present findings in 5 minute short summary.

Importance of writing your own code

Figure: Credit: xkcd.com/1185

Academic honesty and integrity

Study and practice programming to *learn*

- We adopt the collaboration guidelines table set by CS 112 Data Structures `https://ds.cs.rutgers.edu/`
- 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
- Expectations
  - Accessing the class & resources
  - Lecture and short quizzes
  - Midterm and final exams
  - Programming assignments
  - Academic honesty and integrity
- A New Golden Age for Computer Architecture
- Accessing iLab Linux machines

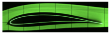
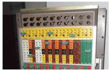
A New Golden Age for Computer Architecture[1]

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.

[1] https://cacm.acm.org/magazines/2019/2/234352-a-new-golden-age-for-computer-architecture/fulltext

History: computer architecture abstractions drove digital revolution

	1940s	1950s	1960s	1970s	1980s	1990s	2000s	2010s
Analog continuous-time computing	Analog computers for rocket and artillery controllers.	Analog computers for field problems. 	1st transistorized analog computer. 	Analog-digital hybrid computers.	...			
Digital discrete-time computing	Turing's *Bomba*.	1st transistorized digital computer.	Moore's law projection for transistor scaling.	Dennard's scaling for transistor power density.	VLSI democratized.			
	Stored program computer.	Microprogramming.	Instruction set architecture.	Reduced instruction set computers.	**Architecture abstraction milestones**			

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

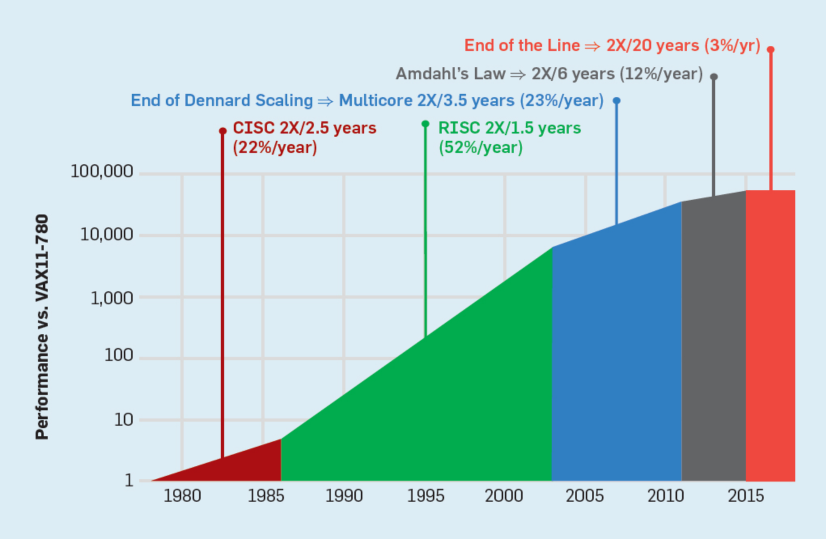


Figure: Credit: A New Golden Age for Computer Architecture

Present: power constraints driving diverse computer architectures

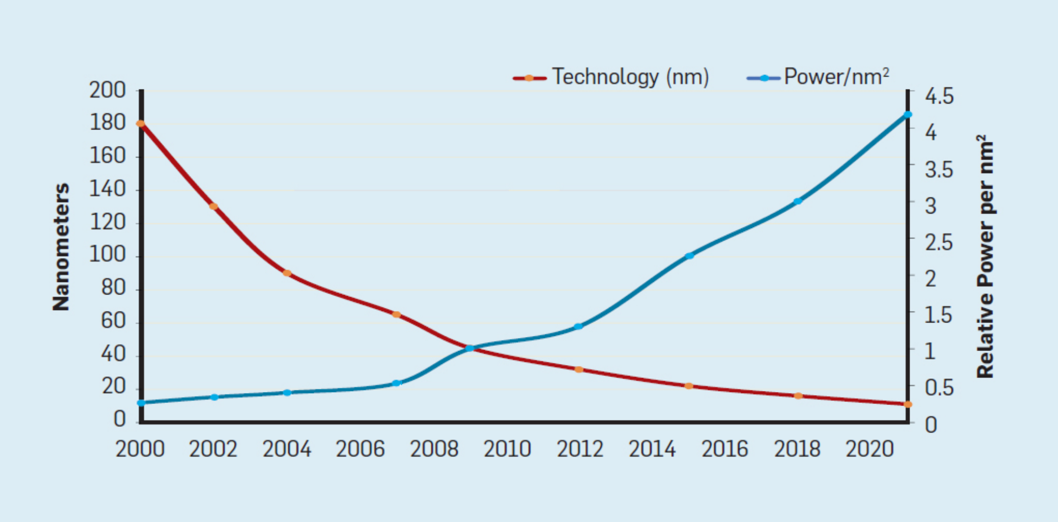
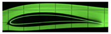
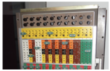


Figure: Credit: A New Golden Age for Computer Architecture

Present: power constraints driving diverse computer architectures

	1940s	1950s	1960s	1970s	1980s	1990s	2000s	2010s
Analog continuous-time computing	Analog computers for rocket and artillery controllers.	Analog computers for field problems. 	1st transistorized analog computer. 	Analog-digital hybrid computers.	...			
Digital discrete-time computing	Turing's *Bomba*.	1st transistorized digital computer.	Moore's law projection for transistor scaling.	Dennard's scaling for transistor power density.	VLSI democratized.	FPGAs introduced.	End of Dennard's scaling. CPUs go multicore.	Cloud FPGAs: Microsoft Catapult. Amazon F1.
	Stored program computer.	Microprogram ming.	Instruction set architecture.	Reduced instruction set computers.		GPUs introduced.	Nvidia introduces CUDA.	ASICs: Google TPUs. DE Shaw Research Anton.

Heterogenous architectures

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

Scaling challenges drive heterogenous architectures

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

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

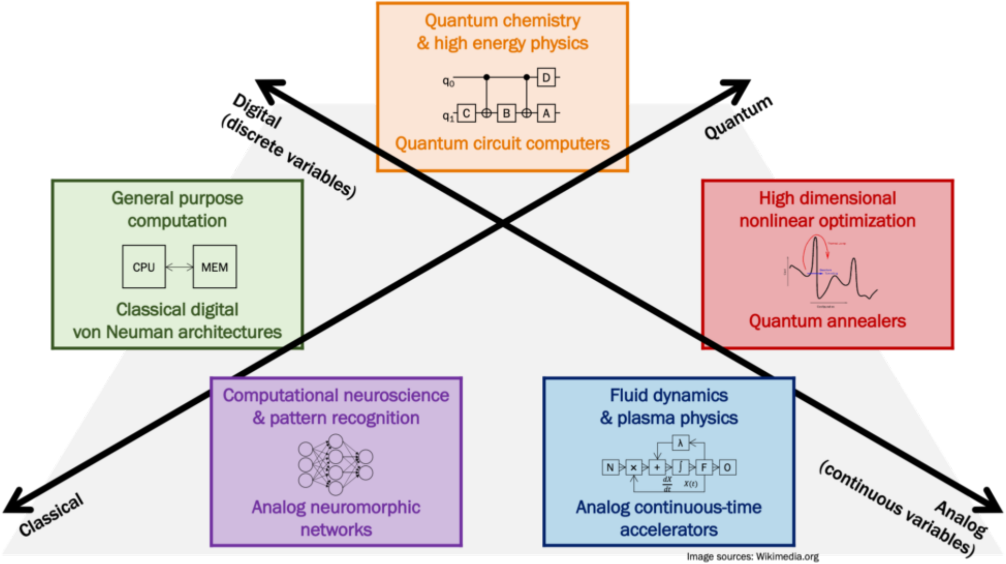
Docker, Amazon AWS

Present: a rapidly evolving and influential field of study

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

- Internet technology
- Security
- Distributed systems
- Database implementation
- Parallel programming
- Operating systems
- Systems programming
- Programming languages and compilers

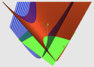
Future: post-Moore's Law computer architectures

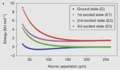


Future: post-Moore's Law computer architectures

Nonlinear scientific computation

Quantum simulation & optimization





New and extreme workload challenges

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

Limitations in transistor scaling

Dennard's scaling already ended

Moore's law increasingly costly to sustain

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?

Image sources: Lanyon and Whitfield et al., 2010

Table of contents

- Curiosity
  - Computer science
  - Computer systems
- Community
  - The students
  - The instructors
- Learning
  - Preview of syllabus
- Expectations
  - Accessing the class & resources
  - Lecture and short quizzes
  - Midterm and final exams
  - Programming assignments
  - Academic honesty and integrity
- A New Golden Age for Computer Architecture
- Accessing iLab Linux machines

Why use Linux?

Do you have Linux? Trick question...

Why use Linux?

- Stable
- Open source
- Flexible: all form factors (wearables, IoT, Raspberry Pi, Roku, Android, laptops, iLab, web hosting, warehouse-scale datacenters)
- A critical piece of infrastructure for practicing computer science

Key steps to get going

1. Activate account: `https://services.cs.rutgers.edu/accounts/`
2. Familiarize yourself with CS department infrastructure: `https://resources.cs.rutgers.edu/docs/new-users/beginners-info/`
3. Use what you are familiar with to log onto iLab remotely. Command line: Windows command line, macOS, terminal, PuTTy. Graphical: X2Go... `https://resources.cs.rutgers.edu/docs/other/working-at-home/`
4. Use what you are familiar with to move files. SCP, Filezilla, Cyberduck... `https://resources.cs.rutgers.edu/docs/file-storage/accessing-files-remotely/`
5. Use what you are familiar with to edit files. Vim, Emacs, other text editors, VS Code...