CSC 411

Computer Organization (Fall 2024)
Lecture 1: Logistics

Prof. Marco Alvarez, University of Rhode Island

What is this course about?

- A comprehensive exploration of fundamental principles in computer organization
  - the fascinating interplay between hardware and software that powers modern computing systems, from transistors to high-level programming
- Topics covered
  - instruction sets, assembly language, processor design, the memory hierarchy, and performance optimization, with a particular focus on the **RISC-V architecture**

How computers work?

A computing system

	Problem
	Algorithm
	Program / Language
	System Software (OS, VM)
ISA (Instruction Set Architecture)	SW / HW Interface
Implementation of the ISA	Microarchitecture
Building blocks (gates)	Logic
	Devices
	Electrons

Computer architecture

- What is computer architecture
  - the science of designing and implementing computing systems
  - primary emphasis on the HW/SW interface, particularly the instruction set architecture, processor design, and memory systems
- Key design goals
  - maximizing performance (execute tasks as quickly as possible)
  - optimizing energy efficiency (while still meeting performance)
  - balancing cost and performance
- Design priorities vary by system type
  - e.g., supercomputers emphasize throughput, smartphones prioritize energy efficiency
  - despite varying priorities, **fundamental principles** apply across different system architectures

Why study computer architecture?

- Understand current/future capabilities of computing systems
  - why computers work as they do (principles and design decisions)
- Develop better software
  - best system programmers understand all abstraction levels and the underlying hardware
    - write more efficient code, debug complex issues, exploit hardware characteristics
- Optimizing performance across the entire stack
  - writing well tuned software by understanding the relationship between hardware and software
- Establish foundation for further work
  - prepare for advanced study and career opportunities

Why learning Assembly?

- Critical for understanding machine-level execution
  - provides a low-level perspective on how computers execute instructions
- Key advantages
  - enhanced debugging skills
  - optimizing program performance and identifying sources of program inefficiency
  - implementing system software that leverages hardware capabilities
  - enhanced security against malware

To illustrate the potential gains from performance engineering, consider multiplying two 4096-by-4096 matrices. Here is the four-line kernel of Python code for matrix-multiplication:

```
for i in xrange(4096):
    for j in xrange(4096):
        for k in xrange(4096):
            C[i][j] += A[i][k] * B[k][j]
```

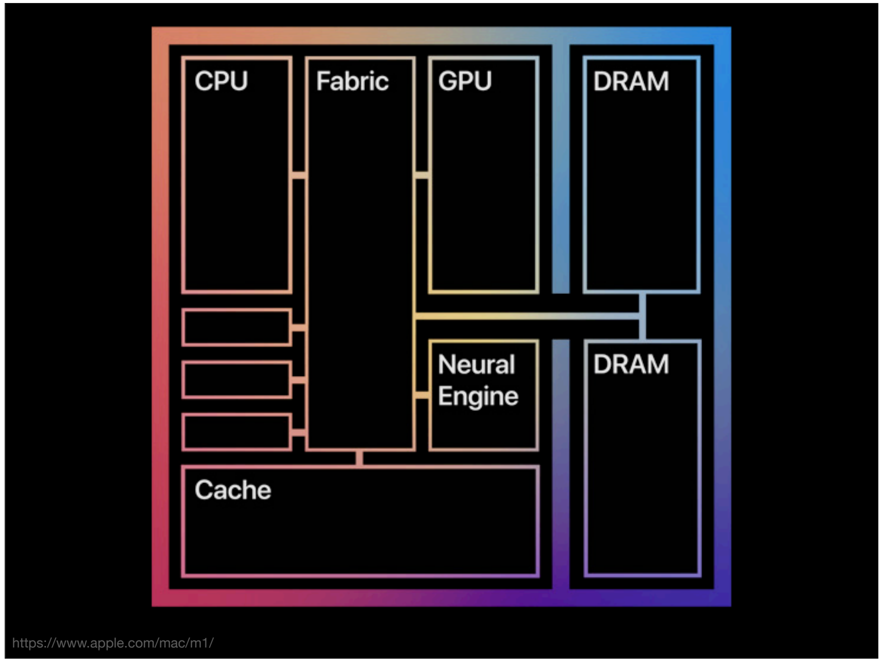
Version	Implementation	Running time (s)	GFLOPS	Absolute speedup	Relative speedup	Fraction of peak (%)
1	Python	25,552.48	0.005	1	—	0.00
2	Java	2,372.68	0.058	11	10.8	0.01
3	C	542.67	0.253	47	4.4	0.03
4	Parallel loops	69.80	1.969	366	7.8	0.24
5	Parallel divide and conquer	3.80	36.180	6,727	18.4	4.33
6	plus vectorization	1.10	124.914	23,224	3.5	14.96
7	plus AVX intrinsics	0.41	337.812	62,806	2.7	40.45

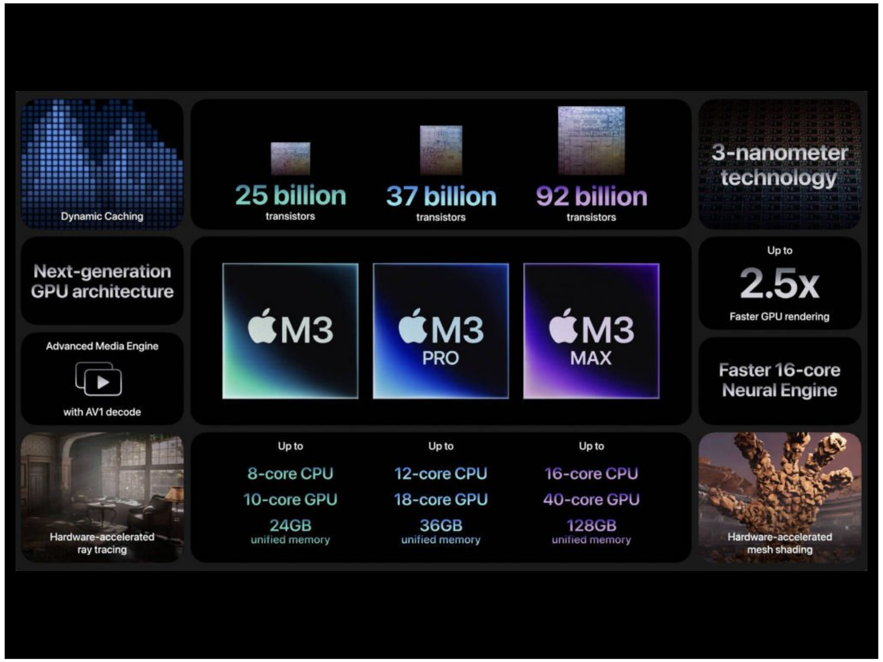
From: “There’s plenty of room at the Top: What will drive computer performance after Moore’s law? “

Modern computer architecture

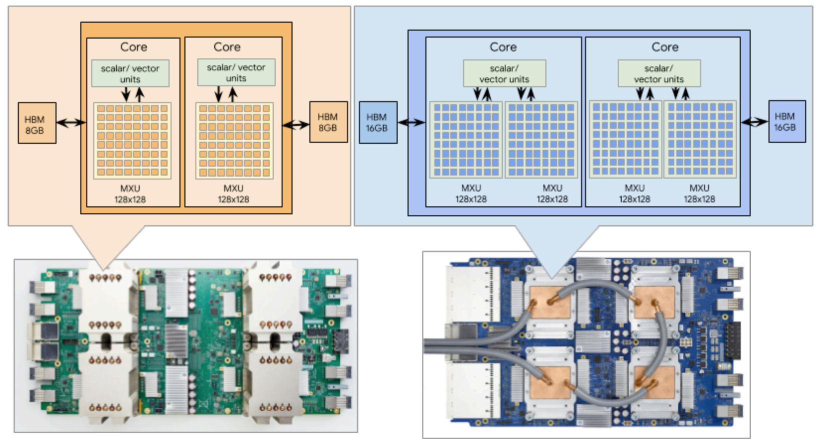
- Key strategies for higher performance and efficiency
  - **co-design** across the hierarchy (bring algorithms to devices)
  - **specialize** as much as possible for targeted efficiency
- Outlook
  - fundamental building blocks and principles persist
  - emergence of novel architectures
    - evolving and powerful landscape
  - exciting era for computer architecture

Examples



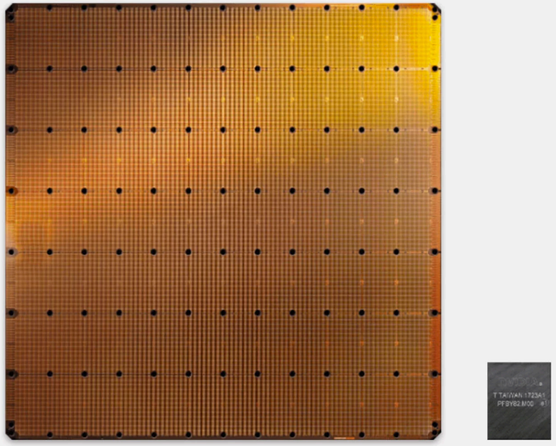


TPU (Tensor Processing Unit)



TPU v2 - 4 chips, 2 cores per chip

TPU v3 - 4 chips, 2 cores per chip



Cerebras WSE-2
46,225mm² Silicon
2.6 Trillion transistors

Largest GPU
826mm² Silicon
54.2 Billion transistors

https://www.cerebras.net/product-chip/

Cerebras's Wafer Scale ML Engine

Course organization

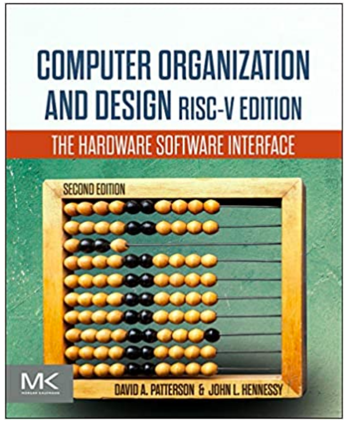
Course website

- URL
  - https://homepage.cs.uri.edu/~malvarez/teaching/csc-411
  - Syllabus
  - Schedule
  - Presentations
  - Resources

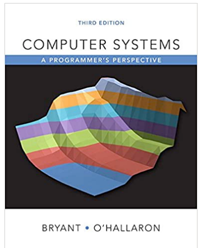
Course information

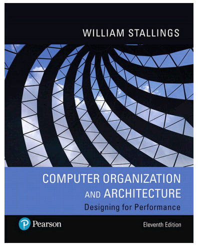
- Lectures
  - MW 4:30-5:45p
- Lab
  - T 5-5:50p (via Zoom)
- Team
  - Marco Alvarez, Instructor
  - Emily Light
  - Liam Cannon
- Office Hours
  - TBA

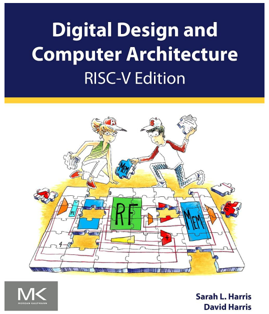
Recommended textbooks



Required







Support tools

Ed Discussion: Academic discussion, polls, quizzes.

Gradescope: Assignment submission and grading.

Zoom: Virtual labs and office hours.

Grading

- Homework assignments (15%)
- Midterm exam (30%)
- Technical presentation (20%)
- Final exam (35%)

Coursework

- Homework assignments
  - problem sets and programming tasks (primarily C and RISC-V)
  - individual work, however discussions and collaboration are allowed
    - you **must write your own** code and solutions
  - late submissions **NOT accepted**
    - ample time given to complete (6-9 days)
    - start early and use office hours for guidance and feedback
- Exams
  - in-person and open-book (printed materials only)
  - no electronic devices allowed
  - mix of multiple-choice, short answer, and problem-solving questions
  - designed to test understanding, not just memorization

Coursework

- Technical presentation
  - team work (3 members), diving deep into an advanced topic
    - specific topics provided by instructor, covering cutting-edge areas in computer organization
    - prepare 20-minute presentation to share your insights/findings
  - enhance your communication and collaboration skills
  - contribute to the class' collective knowledge
  - **outstanding presentations** will receive extra credit

Academic integrity

- Assignments
  - collaboration is encouraged for discussing ideas, but each student/team must submit their own unique solutions
  - sharing/copying solutions from peers is **prohibited**
  - using uncredited AI-generated content **prohibited**
- AI and LLMs
  - AI tools (e.g., ChatGPT, Gemini, Claude, GitHub Copilot) can be used to enhance learning through brainstorming, concept exploration, and strategy development
    - students must critically evaluate and fully understand any AI-generated content used in their work
    - all AI-assisted work **must be cited** in submissions
  - AI tools are designed to support students' learning, NOT to replace independent problem-solving and critical thinking
  - seek instructor guidance if uncertain about appropriate AI tool usage

How to succeed?

- Class attendance
  - students are expected to **attend all lectures and labs**, which are held **synchronously** and will not be recorded
  - regular attendance is linked to higher grades and better comprehension of course material
- Active participation
  - students should engage critically by asking questions, participating in discussions, and thinking deeply about the material
    - during online labs, cameras should be turned on
  - **laptops and cellphones are NOT permitted** unless being used for note-taking
  - students are advised to utilize Ed and attend office hours for additional support
  - `Aim for excellence in your work, rather than simply chasing a good grade`