

Advanced Computer Architecture

Prof. Usagi a.k.a.
Hung-Wei Tseng



Instructor — Hung-Wei Tseng

- Associate Professor @ UC Riverside, 05/2019—
- Website: <https://intra.engr.ucr.edu/~htseng/>
- E-mail: cs203 @ escal.org
- Visiting Researcher @ Google, 01/2023—03/2023
 - Working for TensorFlow Lite
- PhD in **Computer Science**, University of California, San Diego, 2014
- Research Interests
 - General-purpose computing on AI/ML/NN/RayTracing accelerators
 - Or anything else fun — we have an OpenUVR project recently
- Fun fact: Hung-Wei was once considering a career path as a singer but went back to academia due to the unsuccessful trial





Now, make sure you login to Poll Everywhere (through the App or the website) with your UCR email (you're preregistered already)



Where did you spend your summer?

Where did you spend your holiday season? 0

Start the presentation to see live content. For screen share software, share the entire screen. Get help at pollev.com/app

What're your most favorite CS topics?

Your most favorite CS topic

0

Nobody has responded yet.

Hang tight! Responses are coming in.



Why are you taking CS203?

Why're you taking computer architecture?

0

Nobody has responded yet.

Hang tight! Responses are coming in.

chatgpt.com

RightNow Media ChatGPT

Get Plus

Ready when you are.

What's the most popular topic in computer science?

+ Tools

Create image Surprise me Code Summarize text More

gemini.google.com

RightNow Media Google Gemini

Gemini 2.5 Flash

What's the most popular topic in computer science?

請稍候...

請輸入 Gemini 提示詞

+ Deep Research Canvas 圖像

UC San Diego 對話不會用於改良 Google 模型。Gemini 可能會出錯，因此請查證回覆內容
個人隱私權與 Gemini

What do you care as a computer scientist?



Algorithms
Data Structures
Computer Architecture
Programming Languages
User Interfaces

chatgpt.com

RightNow Media Popular computer science topics

ChatGPT

What's the most popular topic in computer science?

The most popular topic in computer science can vary depending on the context (e.g., academia vs. industry), but as of now (2025), the **most popular and impactful topic** overall is:

Artificial Intelligence (AI) and Machine Learning (ML)

Why it's the most popular:

- **Widespread applications:** From natural language processing (like ChatGPT) to computer vision, recommendation systems, autonomous vehicles, and healthcare.
- **Massive research output:** AI/ML dominates conferences like NeurIPS, ICML, and CVPR.
- **Industry demand:** Companies across all sectors are investing heavily in AI talent and products.
- **Public interest:** AI technologies are shaping everyday tools and generating a lot of media attention.

What's the most important topic in computer science?

+ Tools

ChatGPT can make mistakes. Check important info.

gemini.google.com

RightNow Media Google Gemini

Gemini

2.5 Flash

- **Cybersecurity:** As our world becomes more connected, the need to protect data and systems is more critical than ever. Topics like post-quantum cryptography and zero-trust security models are particularly trending.
- **Data Science and Analytics:** With the explosion of "big data," there is a high demand for professionals who can analyze vast datasets to find patterns, make predictions, and inform business decisions.
- **Quantum Computing:** While still in its early stages, this field holds the potential to revolutionize computing by solving problems that are currently impossible for traditional computers.
- **Cloud Computing:** This continues to be a cornerstone of modern technology, enabling on-demand access to computing resources and services.

Overall, while traditional computer science topics like data structures and algorithms remain fundamental, the most dynamic and popular areas are those that address emerging technologies and their societal impact.

資料來源

What's the most important topic in computer science?

+ Deep Research Canvas 圖像

UC San Diego 對話不會用於改良 Google 模型。Gemini 可能會出錯，因此請查證回覆內容。個人隱私權與 Gemini 提交

chatgpt.com

RightNow Media Popular computer science topics

ChatGPT

systems.

Other contenders for "most important," depending on context:

Topic	Importance
Computer Architecture	Foundation of how computers work, needed for performance, hardware-software interaction.
Operating Systems	Fundamental for understanding how computers manage resources and run programs.
Artificial Intelligence	Arguably the most impactful today; transforming industries and society.
Cybersecurity	Crucial for trust, privacy, and protecting systems/data.
Theory of Computation	Defines the limits of what computers can do; essential for understanding CS at a deep level.
Programming Languages & Compilers	Core to building and running all software.
Databases	Critical for <input type="button" value="down"/> ing, retrieving, and managing data, which powers everything from apps to enterprises.

Ask anything

+ Tools

ChatGPT can make mistakes. Check important info.

gemini.google.com

RightNow Media Google Gemini

Gemini

2.5 Flash

almost every other area of computer science, including AI and software engineering.

- **Theory of Computation:** This field explores the fundamental capabilities and limitations of computers. It asks questions like: What can be computed? How efficiently can it be computed? This theoretical understanding underpins the entire discipline, even if it's not directly applied in day-to-day programming.
- **Computer Architecture and Operating Systems:** Understanding how a computer's hardware and software interact is vital. Computer architecture deals with the design of the computer system itself, while operating systems manage the hardware and software resources. These topics explain how programs are executed, how memory is managed, and how a computer functions at its most basic level.
- **Programming Languages:** Knowledge of programming languages and their underlying paradigms is essential. This includes understanding different types of languages (e.g., imperative, functional, object-oriented) and how they are used to write instructions for a computer.

The Most Impactful Areas

These are the topics that are currently shaping technology and have a

How important is computer architecture in computer science?

+ Deep Research Canvas 圖像

UC San Diego 對話不會用於改良 Google 模型。Gemini 可能會出錯，因此請查證回覆內容。
個人隱私權與 Gemini

The return of backpropagation

- Between 2005 and 2009 researchers (in Canada!) made several technical advances that enabled backpropagation to work better in feed-forward nets.
 - Unsupervised pre-training; random dropout of units; rectified linear units.
 - The technical details of these advances are very important to the researchers but they are not the main message.
 - The main message is that backpropagation now works amazingly well if you have two things:
 - a lot of labeled data
 - a lot of convenient compute power (e.g. GPUs)



Sixth Edition

John L. Hennessy | David A. Patterson

COMPUTER ARCHITECTURE

A Quantitative Approach



Computer Architecture

Enables

Deep Learning

Computer Architects — Hidden Figures

BASED ON THE UNTOLD TRUE STORY

HIDDEN FIGURES



What is “Computer Architecture”?

ChatGPT 3.5

threats, knowledge of security principles and cryptographic techniques is crucial for protecting data and systems from unauthorized access and attacks.

9. **Artificial Intelligence and Machine Learning:** AI and ML are transforming various industries and applications. Understanding concepts like neural networks, deep learning, reinforcement learning, and natural language processing is becoming increasingly important.

10. **Human-Computer Interaction (HCI):** HCI focuses on designing interactive systems that are user-friendly and efficient. Understanding user behavior, usability principles, and interaction design is essential for developing effective user interfaces.

These topics serve as foundational knowledge for computer scientists and are applicable across various domains and specialties within the field. Additionally, interdisciplinary areas such as bioinformatics, computational biology, and computational neuroscience are gaining importance, highlighting the interdisciplinary nature of computer science.

HT You
What is computer architecture?

ChatGPT

Message ChatGPT...

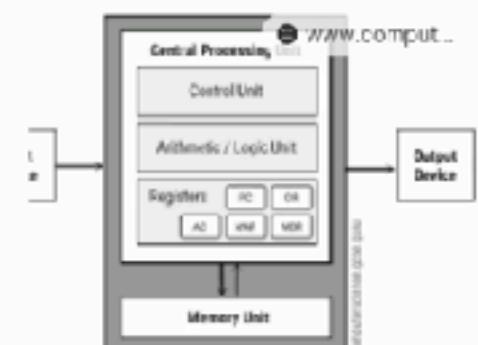
ChatGPT can make mistakes. Consider checking important information.

puter a

Gemini

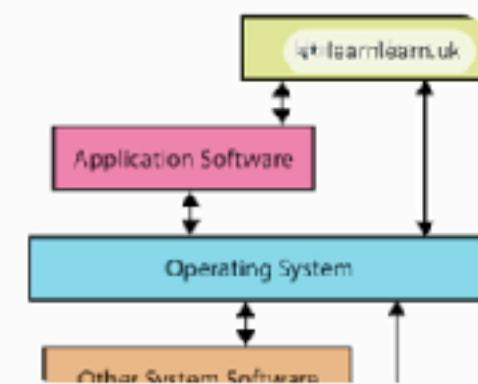
See the latest updates to the Gemini Apps Privacy Hub

well as the software that controls them. An understanding of computer architecture is essential for writing efficient programs and designing new computing systems.



Computer Architecture computer science

- Operating Systems: An operating system (OS) is the software that manages the hardware and software resources of a computer. It provides a platform for running other programs and applications. Operating systems are essential for the smooth operation of computers.



Operating Systems computer science

These are just a few of the most important topics in computer science. These foundational concepts are essential for understanding how computers work and for creating new technologies.

What is computer architecture?

Gemini may display inaccurate info, including about people, so double-check its responses. [Your privacy](#)

Submit

What's computer architecture?



architecture noun

ar·chi·tec·ture | \är-kuh-tek-chur \

Definition of *architecture*

1 : the art or science of building

specifically : the art or practice of designing buildings especially habitable ones

2 **a** : formation or construction resulting from design and skill

// the architecture of the garden

b : a unifying or coherent form or structure

// a novel that lacks architecture

3 : architectural product or work

// buildings that comprise the architecture of the square

4 : a method or style of building

// Gothic architecture

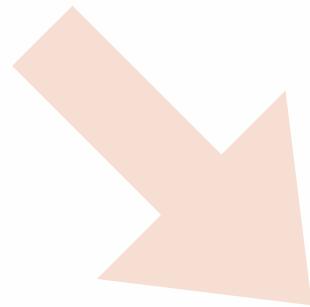
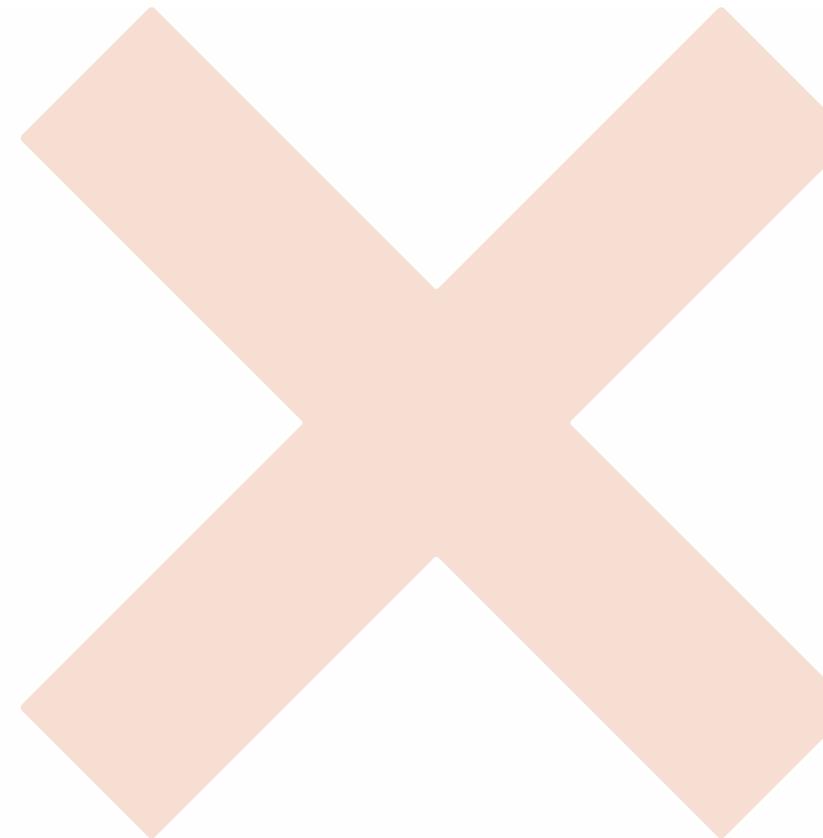
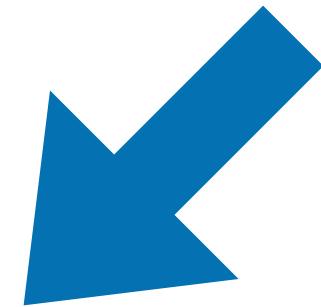
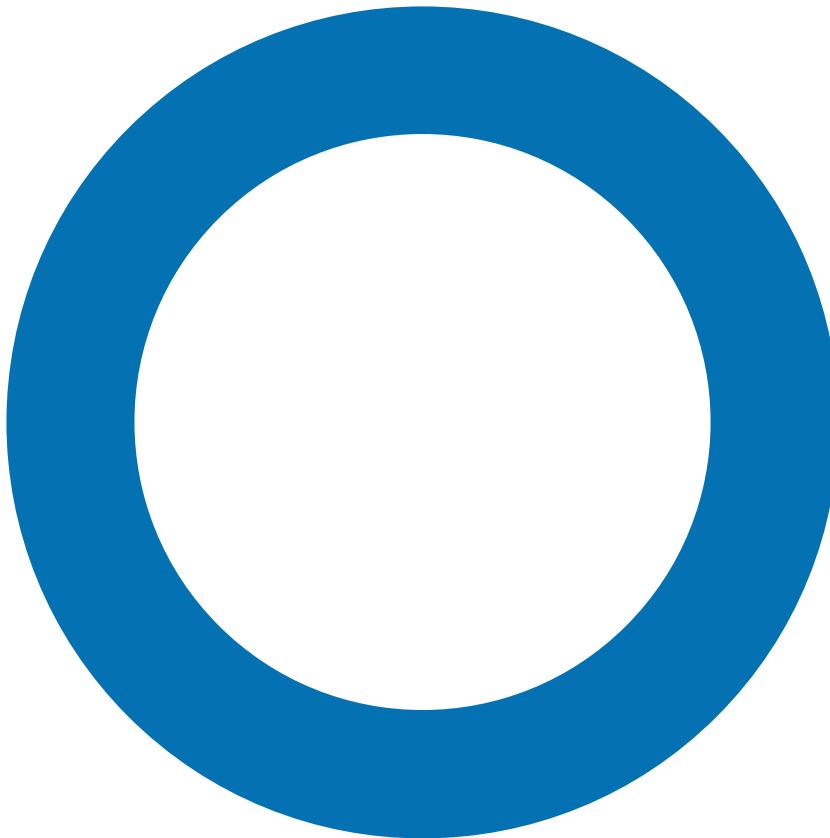
5 : the manner in which the components of a computer or computer system are organized and integrated

// different program architectures

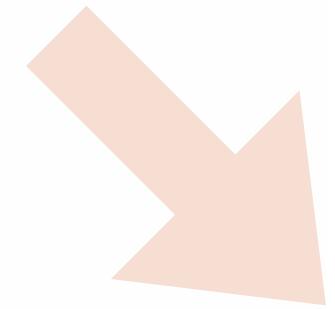
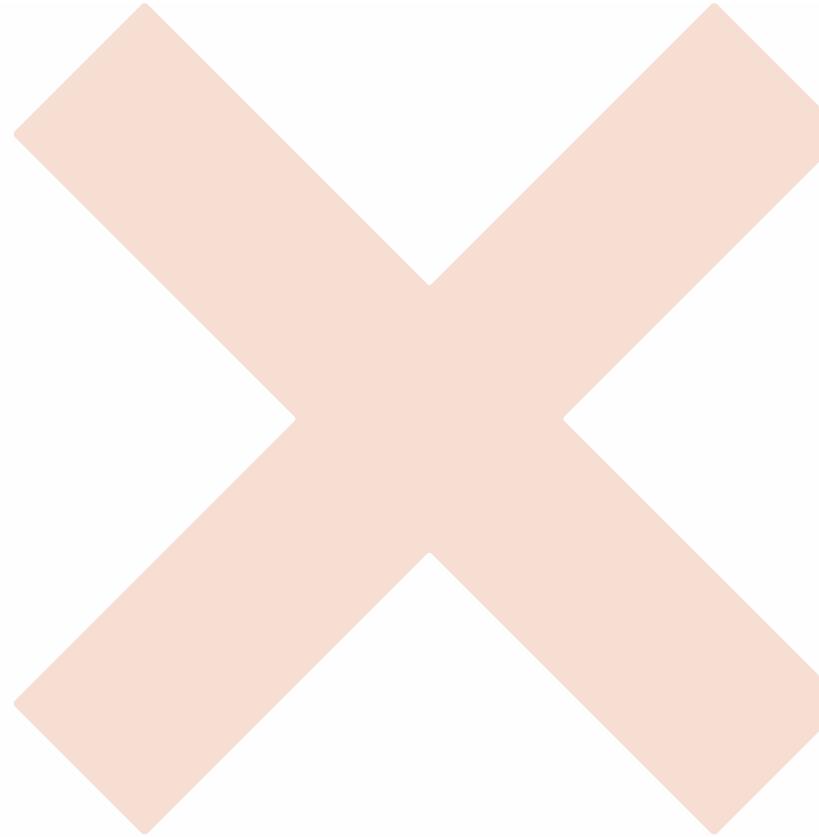
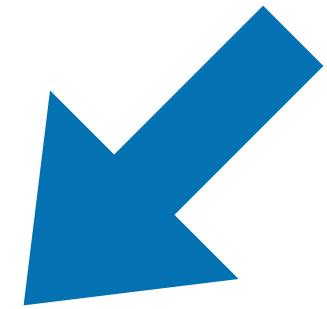
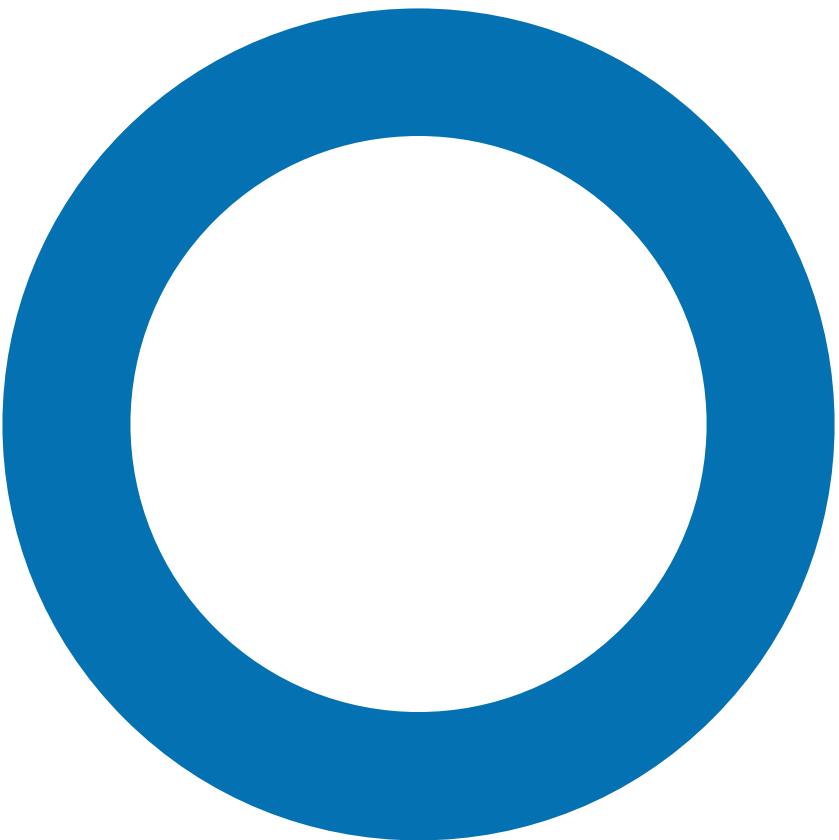
**The manner in which the components
of a computer or computer system are
organized and integrated**

**How much do we understand
“Computer Architectures” for now**

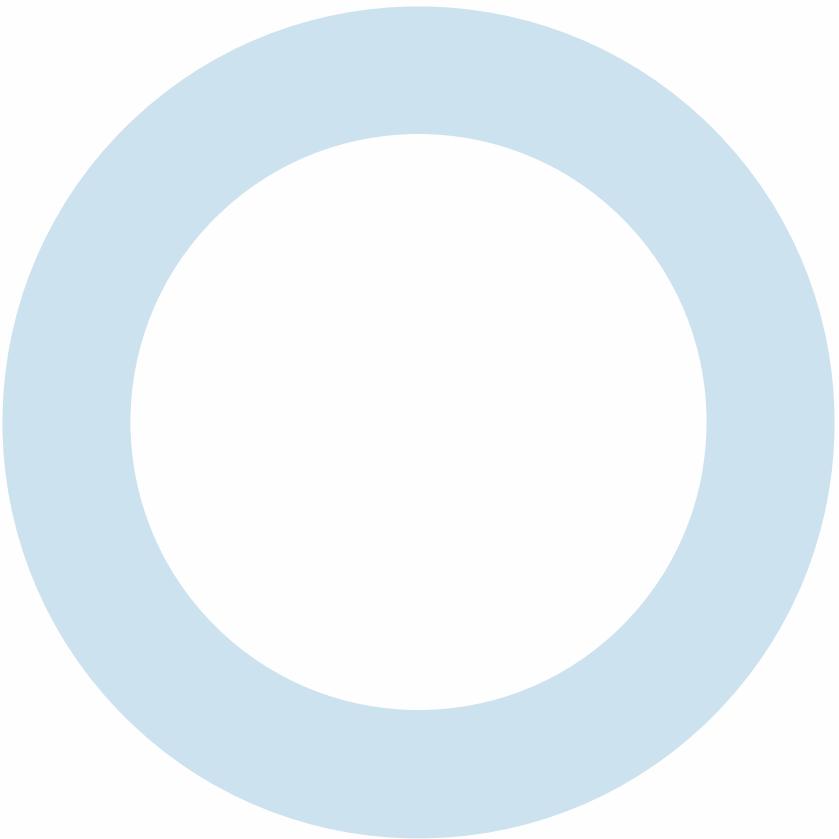
Processors and memories are essential for most modern general-purpose computers



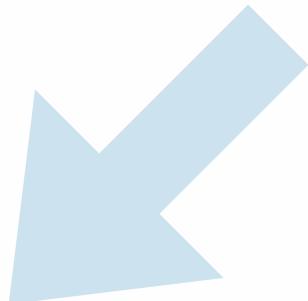
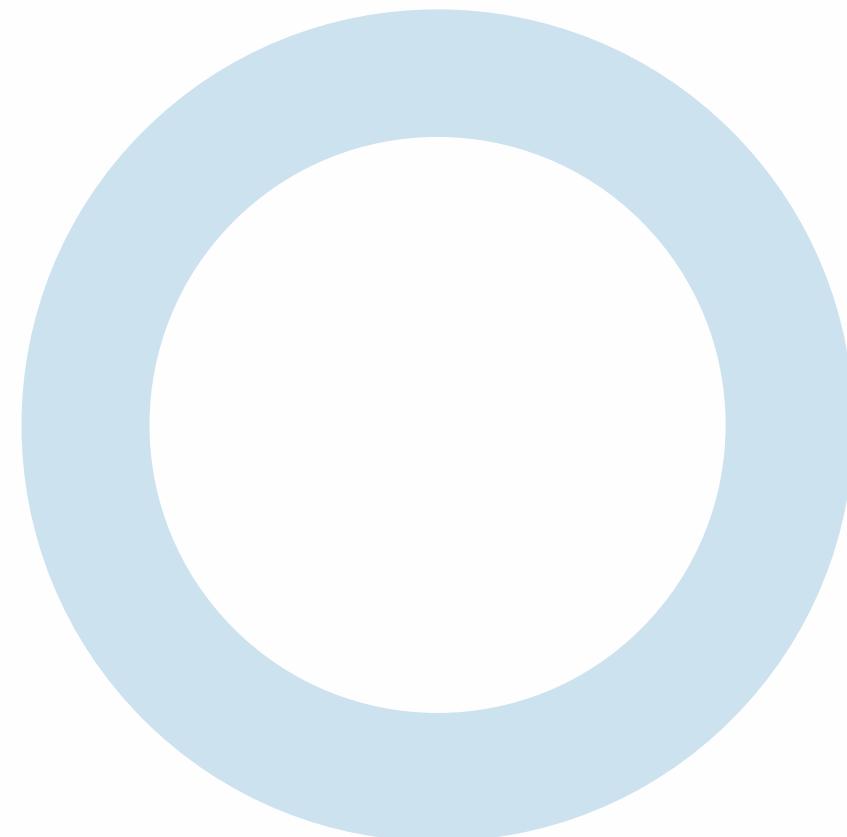
Moore's Law is still alive



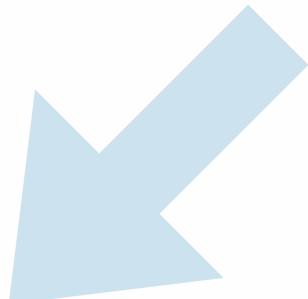
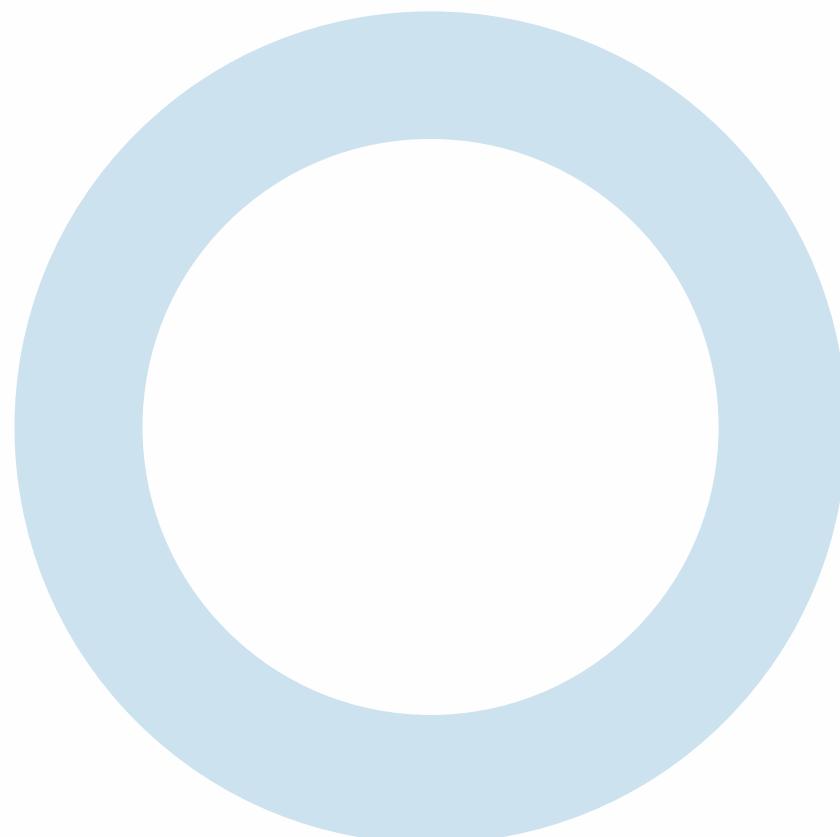
On the same machine, programs solving the same problem using fewer operations will have shorter execution time



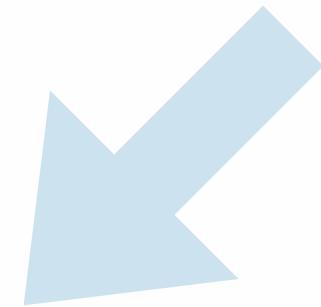
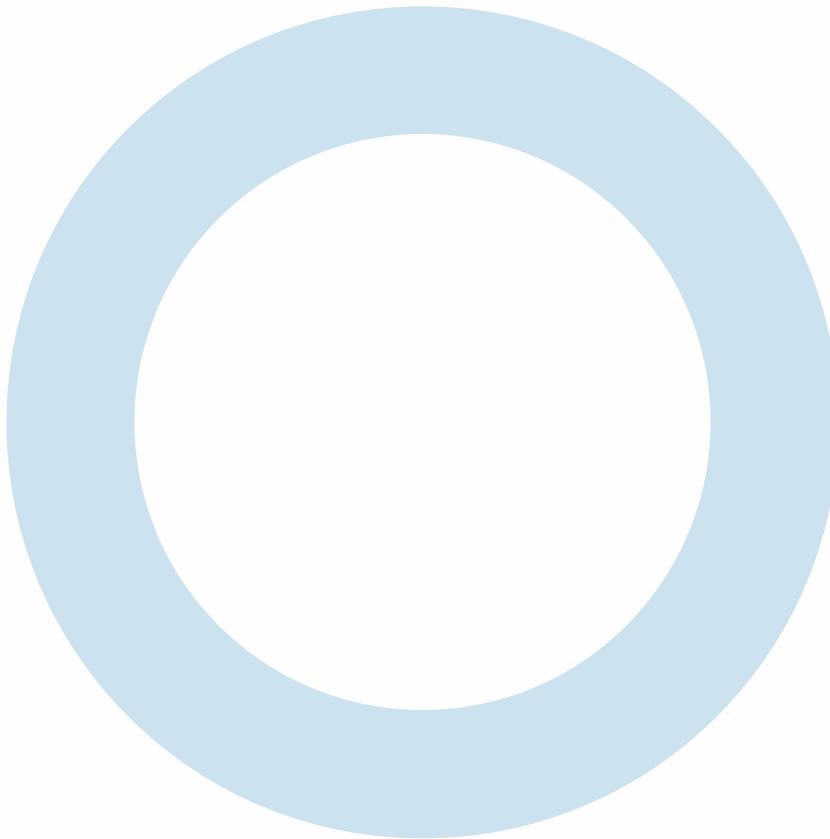
On the same machine, an algorithm with lower computational complexities will have shorter execution time when solving the same problem with the same input



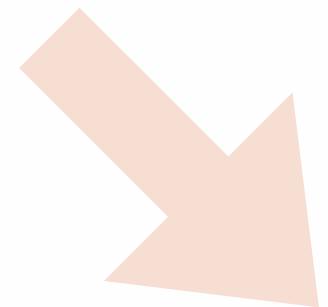
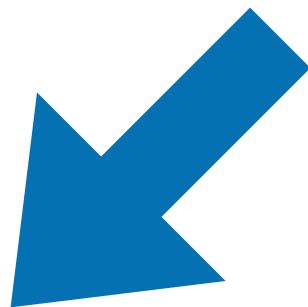
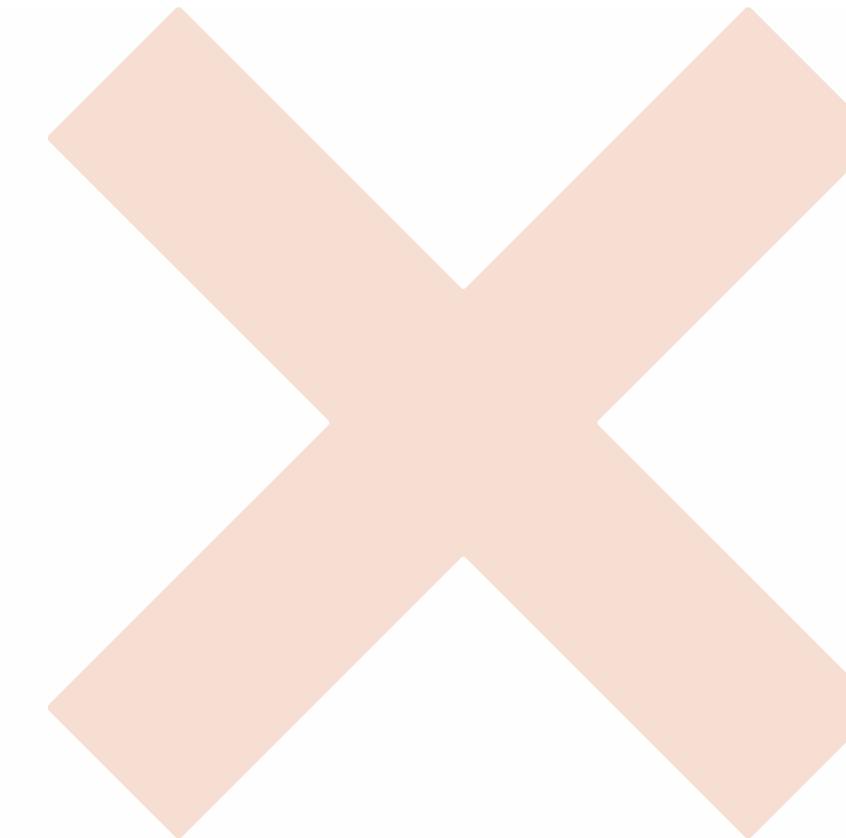
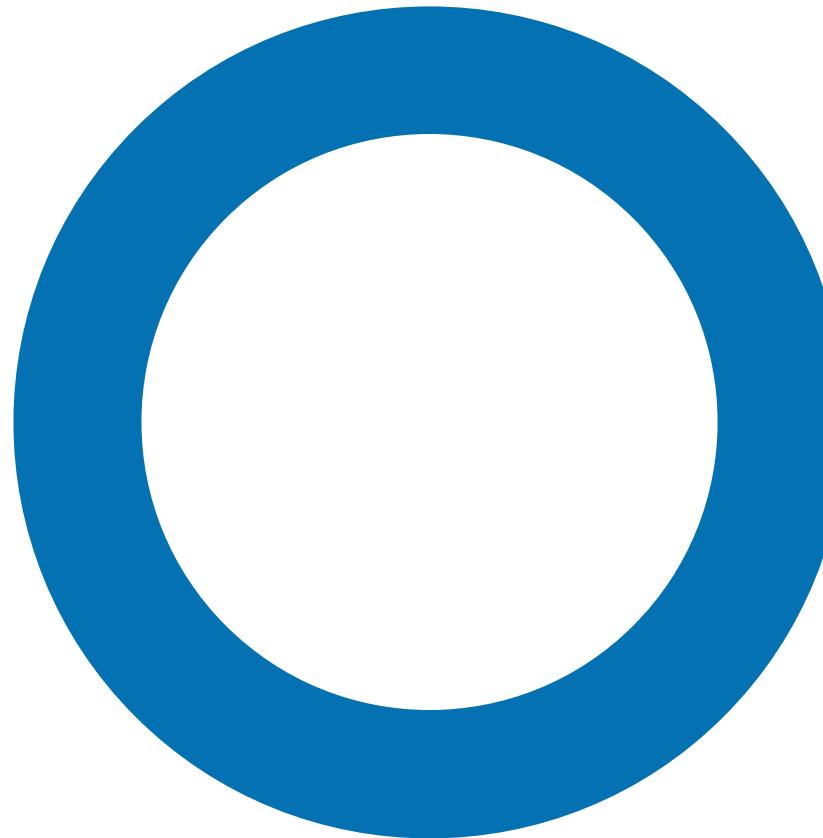
On the same machine, algorithm implementations with the same computational complexities will similar execution time when solving the same problem with the same input



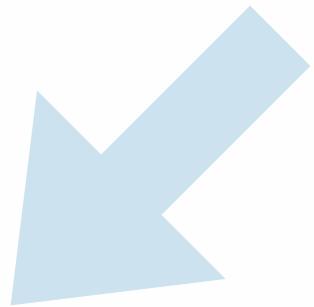
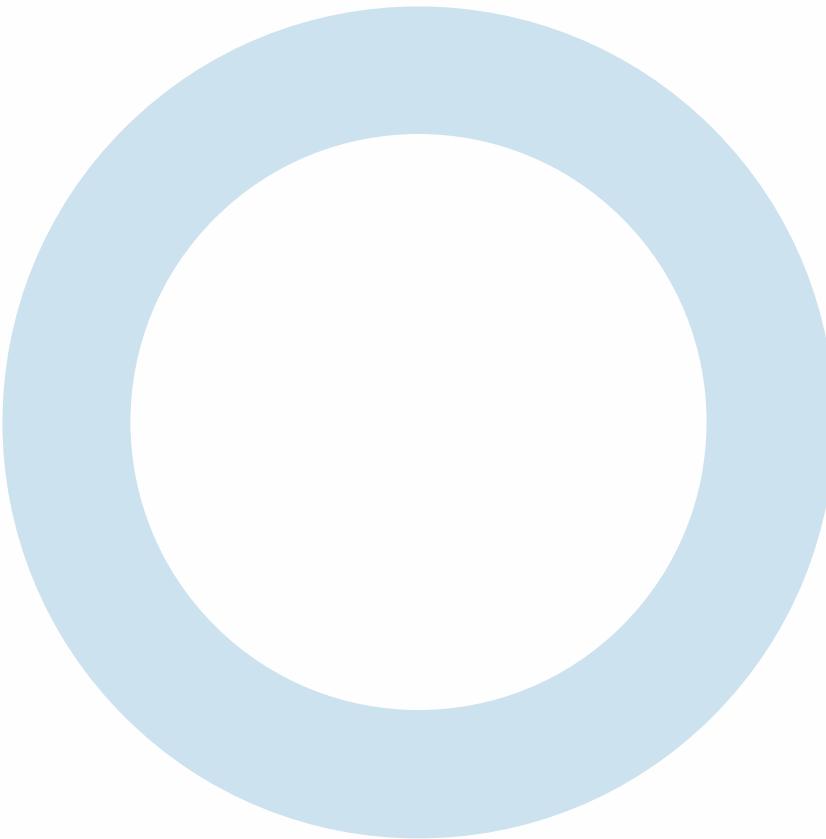
Leveraging more “bit-wise” operations in C code will make the program significantly faster



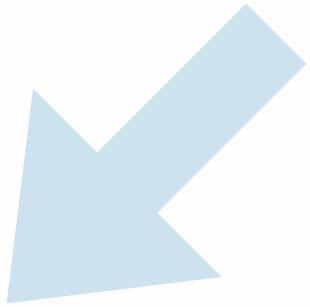
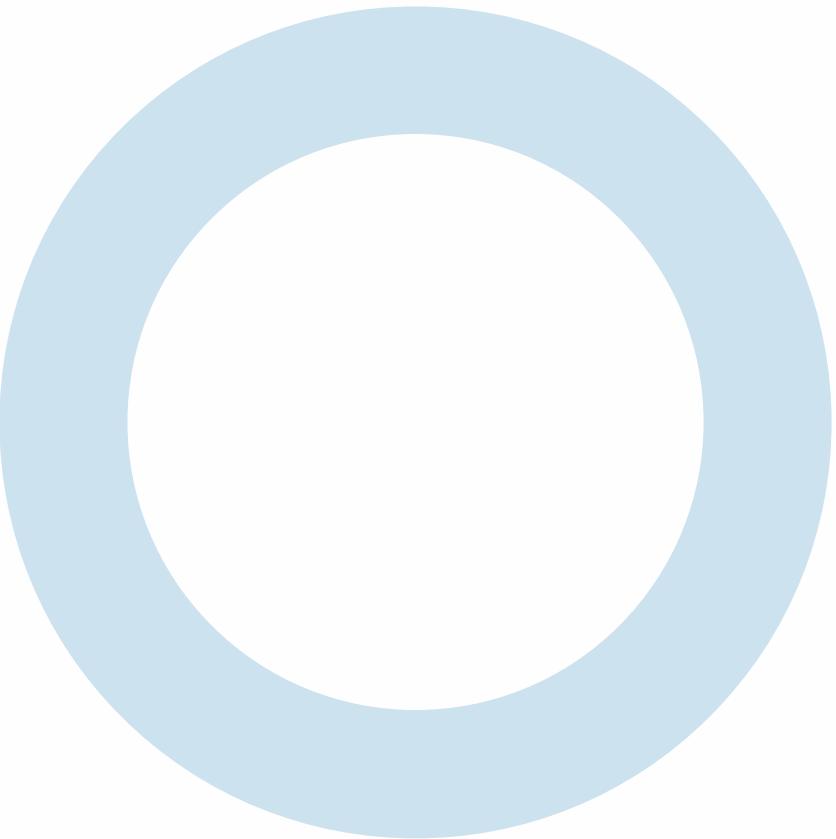
Algorithm complexity is less important if we have rich parallel processing capabilities



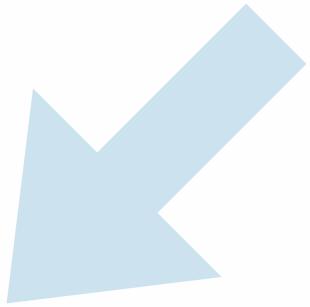
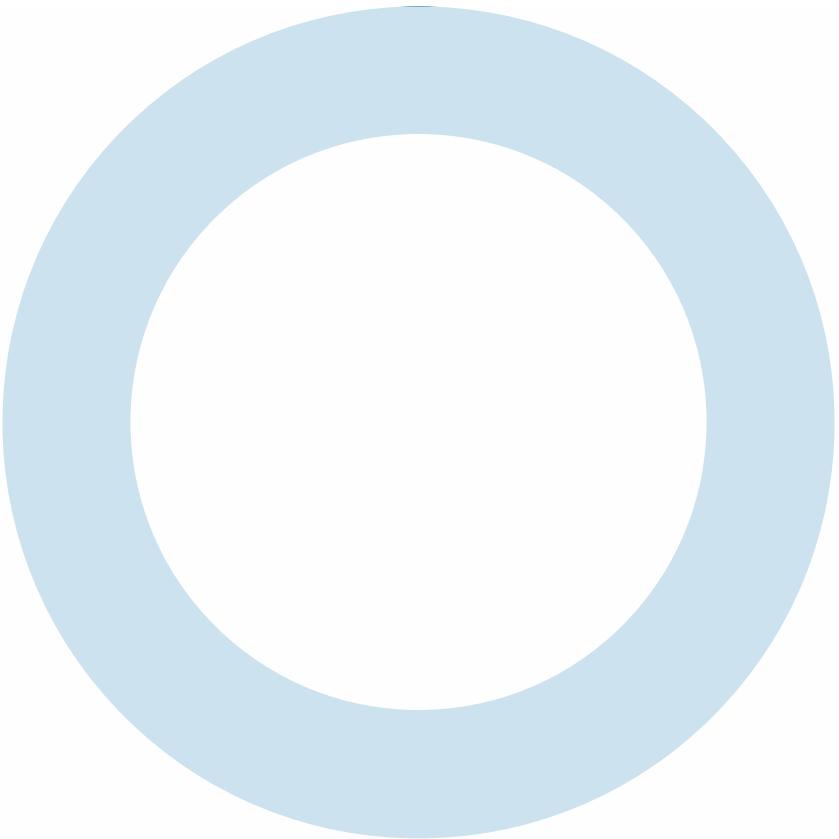
**The smaller size of a transistor,
the smaller power consumption of it**



GPUs are not limited by Moore's Law



**The performance growth of recent GPU architectures
can match the demand of efficient AI/ML training**



Isn't "algorithm" and "computational complexity" good enough?

Demo

```
if(option)
    std::sort(data, data + arraySize);      O(nlog2n)
for (unsigned c = 0; c < arraySize*1000; ++c) {
    int t = std::rand();
    if (data[c%arraySize] >= t)            O(n)
        sum++;
}
if option is set to 1: O(nlog2n)
```

otherwise, O(n): *O(n*)



Start the presentation to see live content. Still no live content? Install the app or get help at PollEv.com/app

On the same machine, programs solving the same problem using fewer operations will have shorter execution time



On the same machine, an algorithm with lower computational complexities will have shorter execution time when solving the same problem with the same input



Demo (2)

A

```
for(i = 0; i < ARRAY_SIZE; i++)
{
    for(j = 0; j < ARRAY_SIZE; j++)
    {
        c[i][j] = a[i][j]+b[i][j];
    }
}
```

B

```
for(j = 0; j < ARRAY_SIZE; j++)
{
    for(i = 0; i < ARRAY_SIZE; i++)
    {
        c[i][j] = a[i][j]+b[i][j];
    }
}
```

$O(n^2)$

Complexity

$O(n^2)$

Performance?



Start the presentation to see live content. Still no live content? Install the app or get help at PollEv.com/app

Demo (2)

A

```
for(i = 0; i < ARRAY_SIZE; i++)
{
    for(j = 0; j < ARRAY_SIZE; j++)
    {
        c[i][j] = a[i][j]+b[i][j];
    }
}
```

$O(n^2)$

A Lot Better!

B

```
for(j = 0; j < ARRAY_SIZE; j++)
{
    for(i = 0; i < ARRAY_SIZE; i++)
    {
        c[i][j] = a[i][j]+b[i][j];
    }
}
```

$O(n^2)$

Complexity

Performance?

Worse

On the same machine, algorithm implementations with the same computational complexities will similar execution time when solving the same problem with the same input



Demo (3) — Bitwise operations?

A

```
void regswap(int* a, int* b) {  
    int temp = *a;  
    *a = *b;  
    *b = temp;  
}
```

B

```
void xorswap(int* a, int* b) {  
    *a ^= *b;  
    *b ^= *a;  
    *a ^= *b;  
}
```



Start the presentation to see live content. Still no live content? Install the app or get help at PollEv.com/app

Leveraging more “bit-wise” operations in C code will make the program significantly faster



Demo (4) — quick sort v.s. bitonic sort on GPU

Quick Sort

 $O(n \log_2 n)$

Bitonic Sort

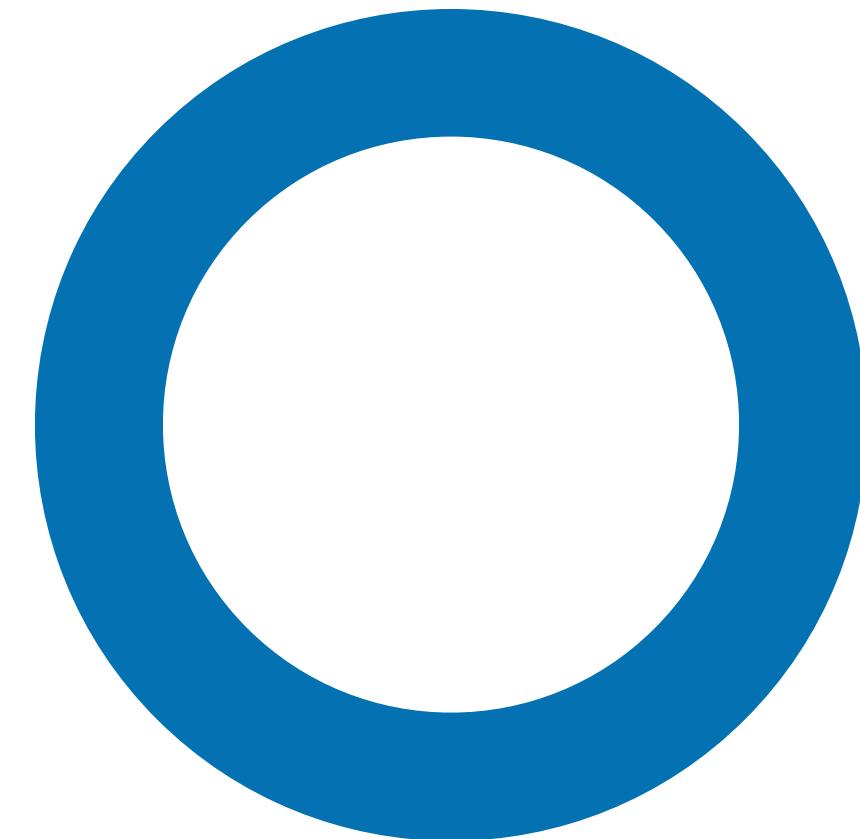
 $O(n \log_2^2 n)$

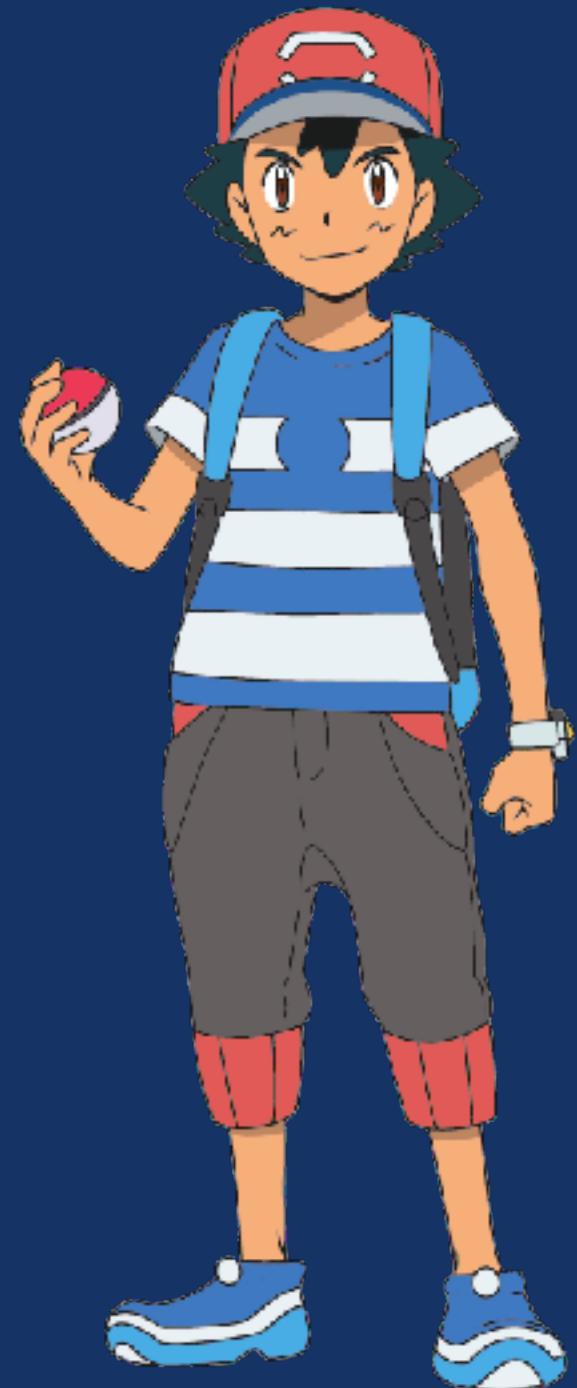
```
void BitonicSort() {  
    int i,j,k;  
  
    for (k=2; k<=N; k=2*k) {  
        for (j=k>>1; j>0; j=j>>1) {  
            for (i=0; i<N; i++) {  
                int ij=i^j;  
                if ((ij)>i) {  
                    if ((i&k)==0 && a[i] > a[ij])  
                        exchange(i,ij);  
                    if ((i&k)!=0 && a[i] < a[ij])  
                        exchange(i,ij);  
                }  
            }  
        }  
    }  
}
```



Start the presentation to see live content. Still no live content? Install the app or get help at PollEv.com/app

Algorithm complexity is less important if we have rich parallel processing capabilities





?????



Thinking about the washlet



Or a Tesla



Take-aways: Why CS203?

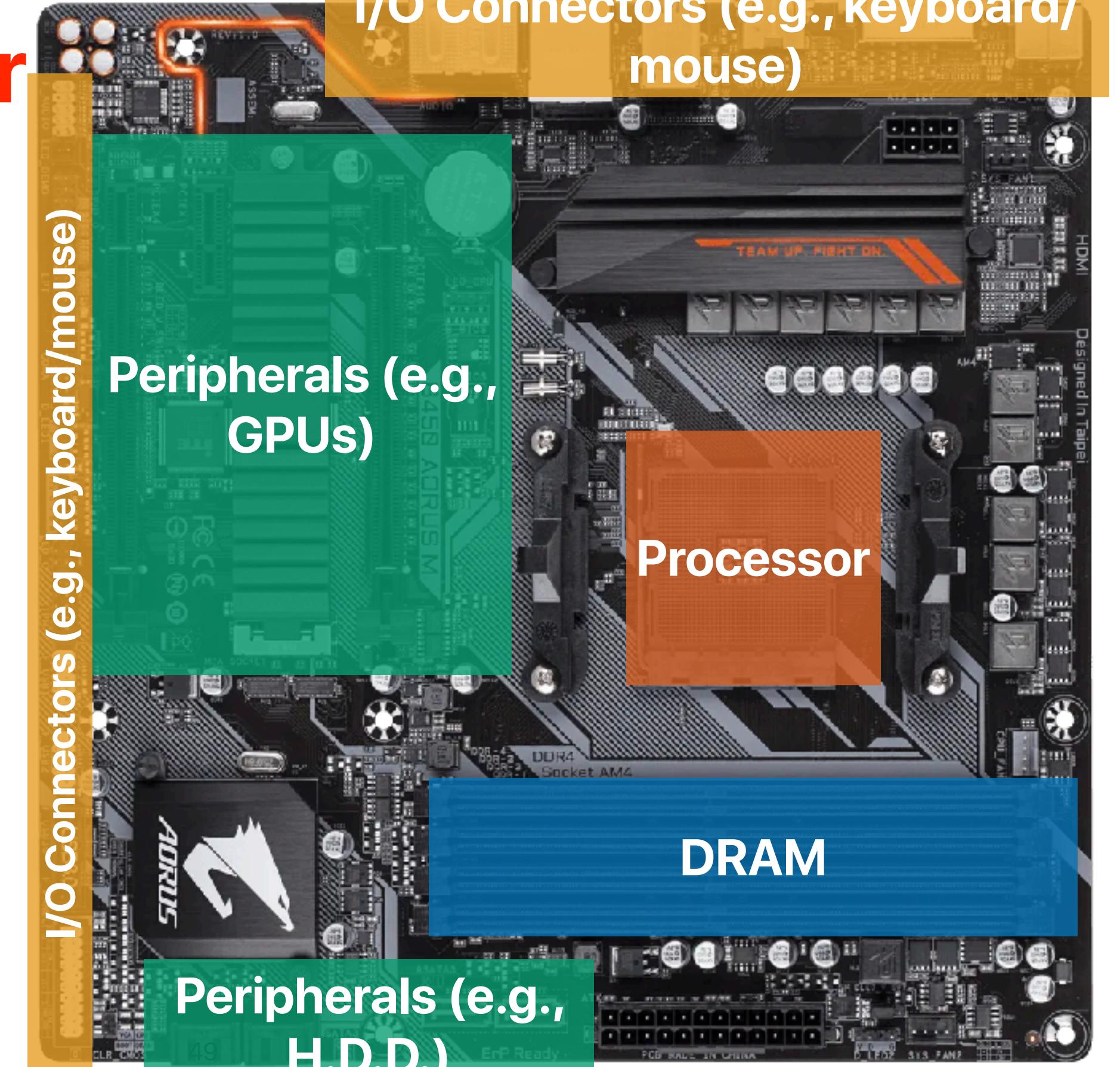
- Algorithm complexity does not work well on “real” computers

Big Picture of Computer Architecture: the Von Neumann Architecture

Desktop Computer

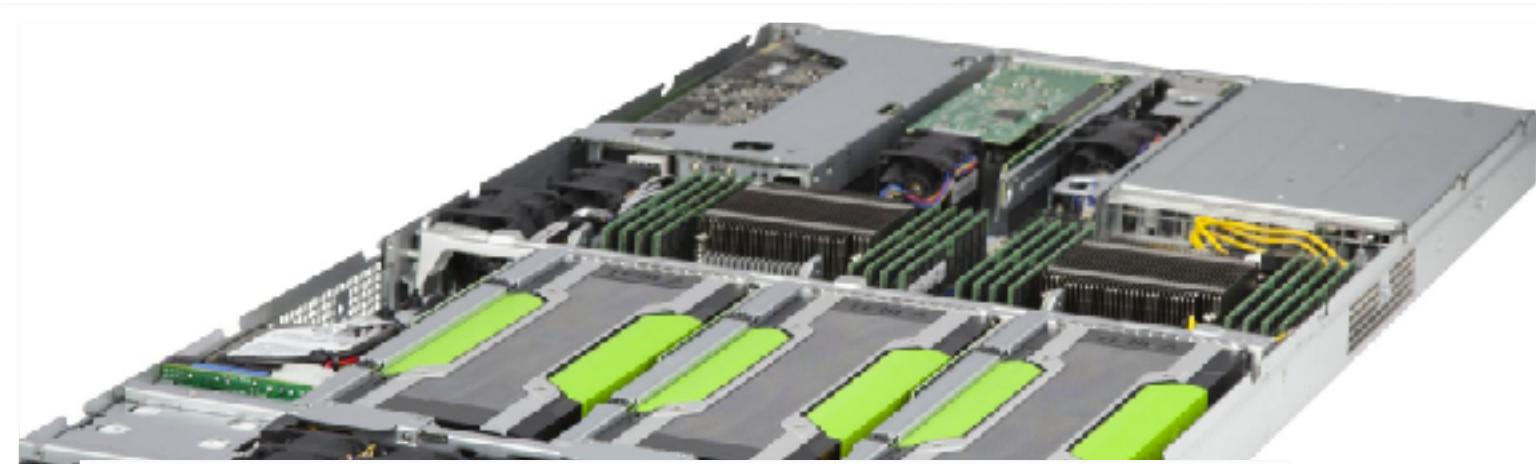


I/O Connectors (e.g., keyboard/mouse)



Server

I/O Connectors (e.g., keyboard/mouse)



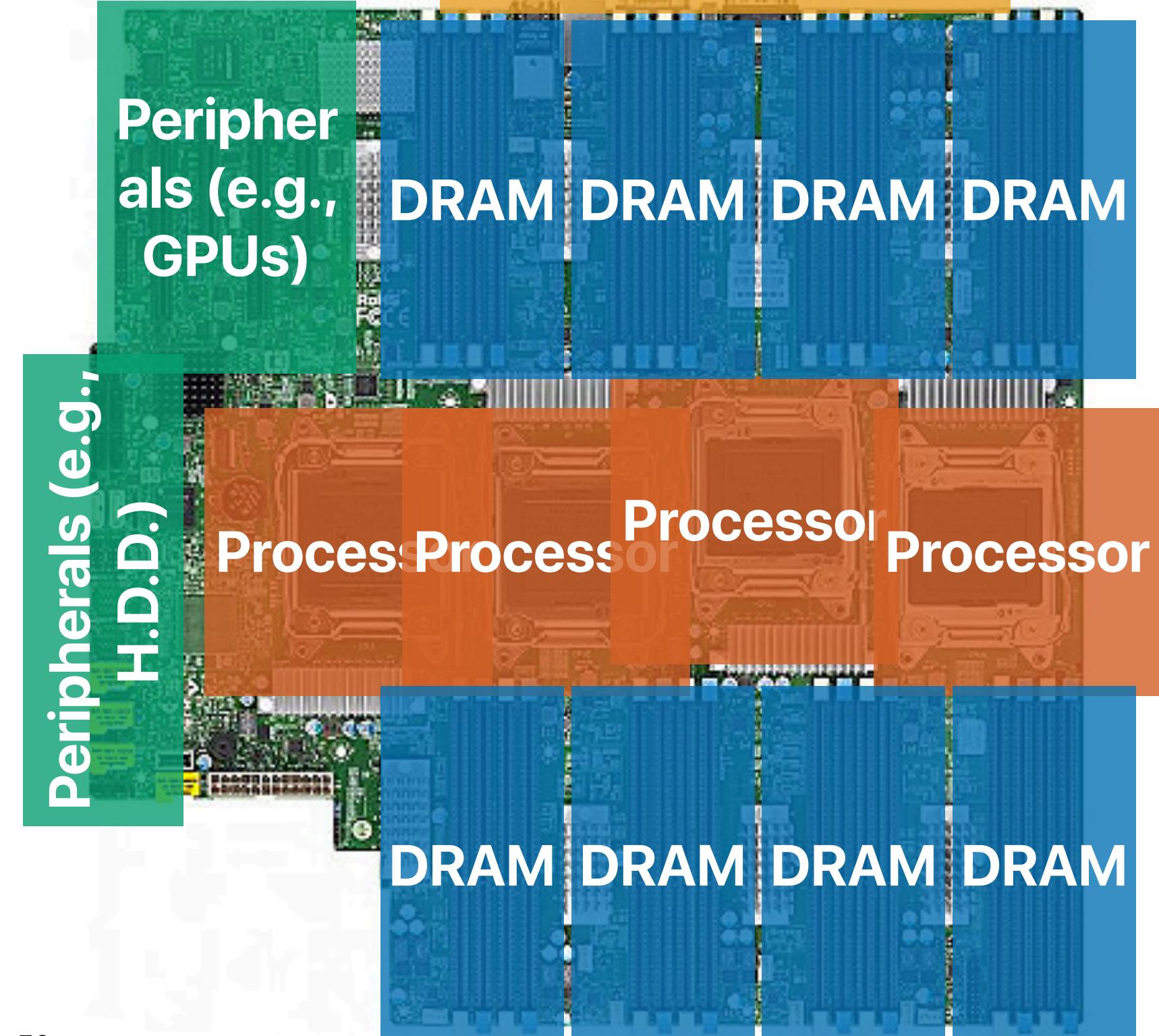
Peripherals (e.g., H.D.D.)

Peripherals (e.g., GPUs)

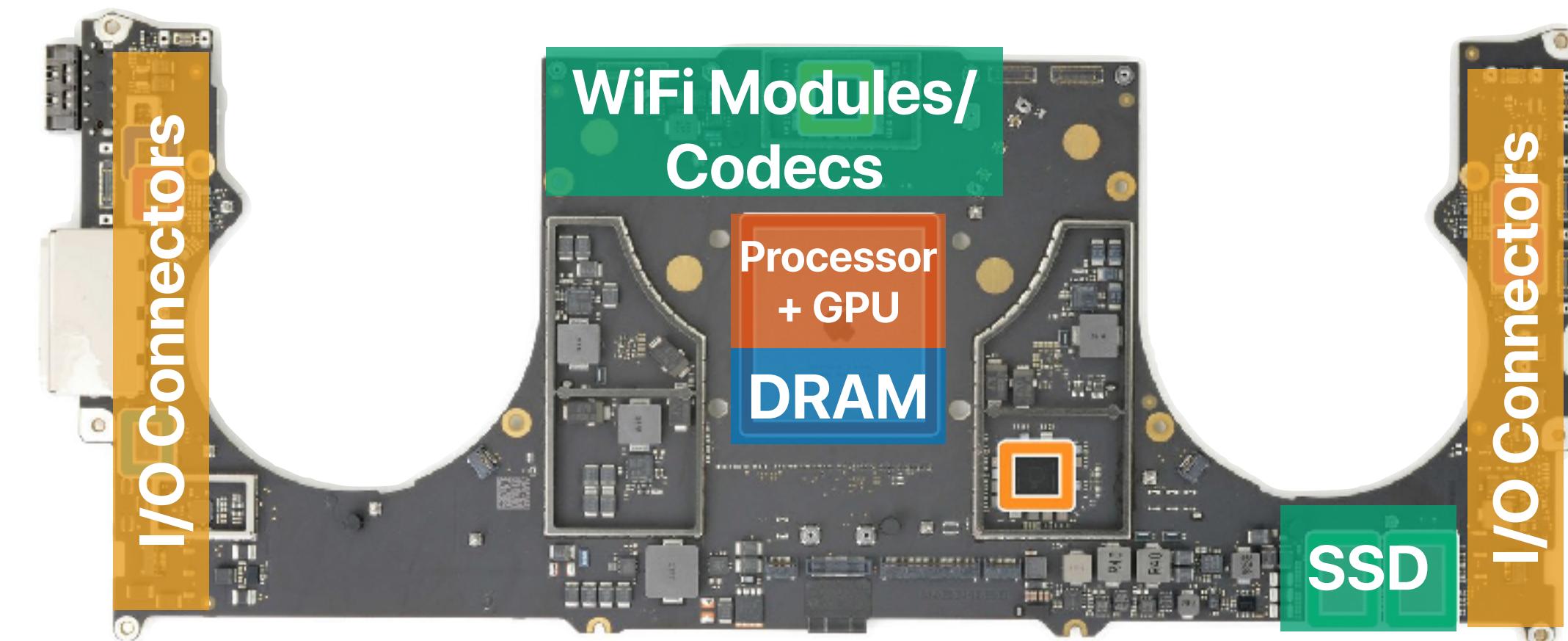
DRAM DRAM DRAM DRAM

Processor Processor Processor Processor

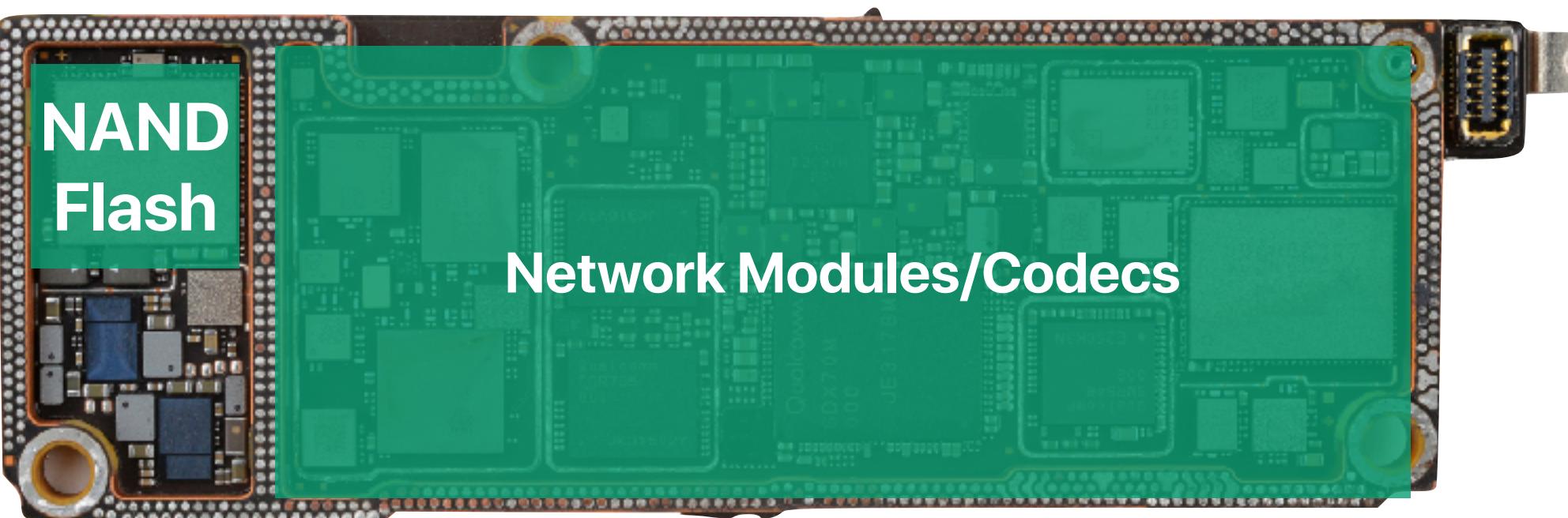
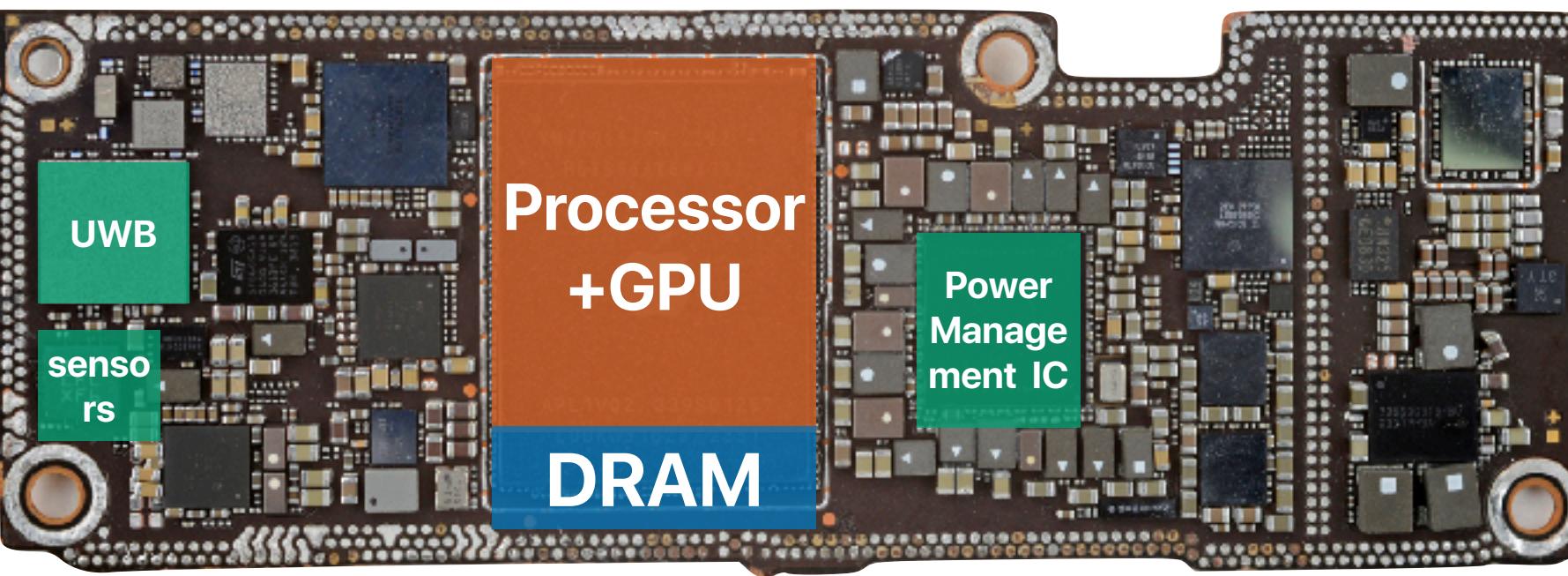
DRAM DRAM DRAM DRAM



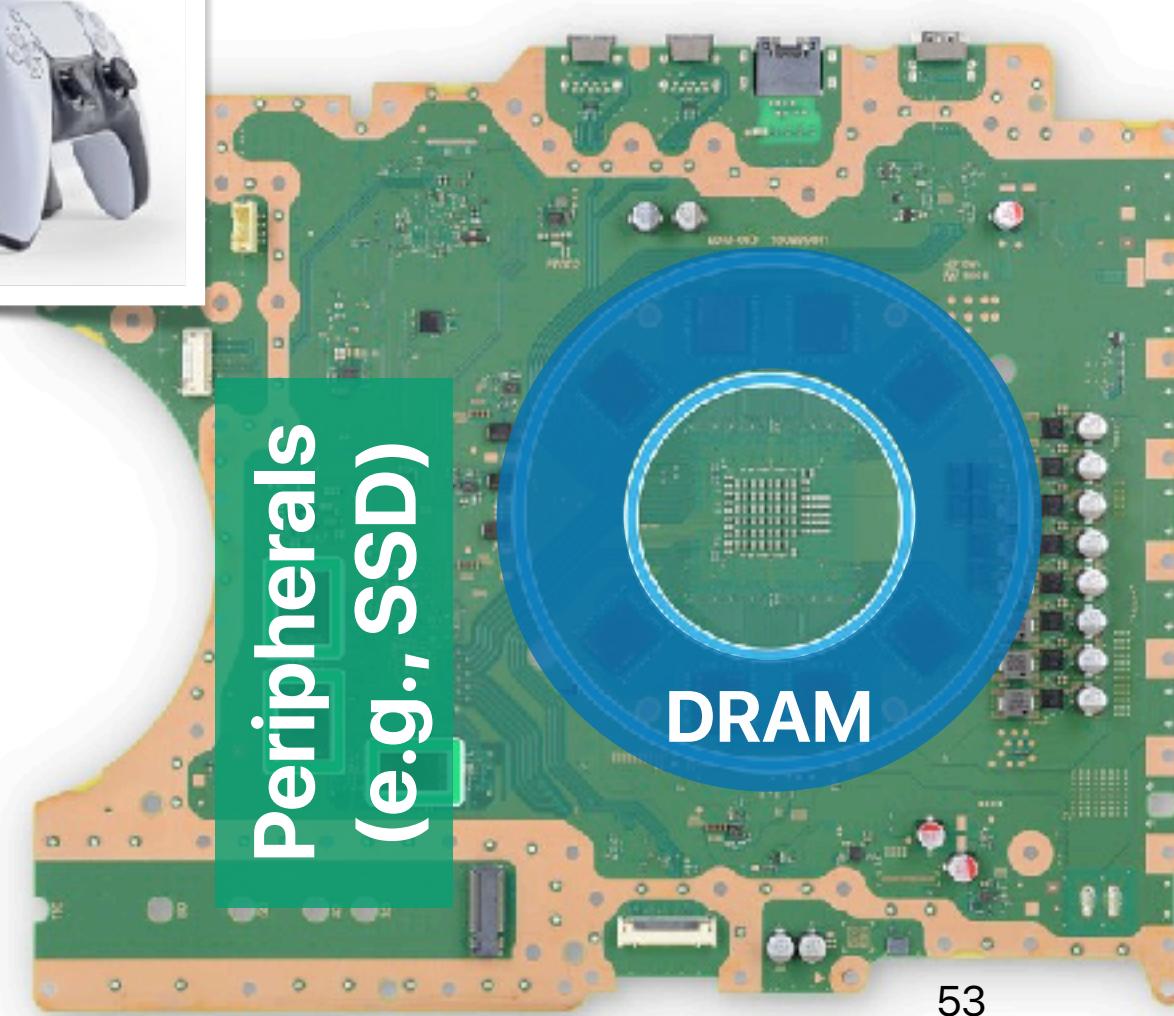
MacBook Pro M3



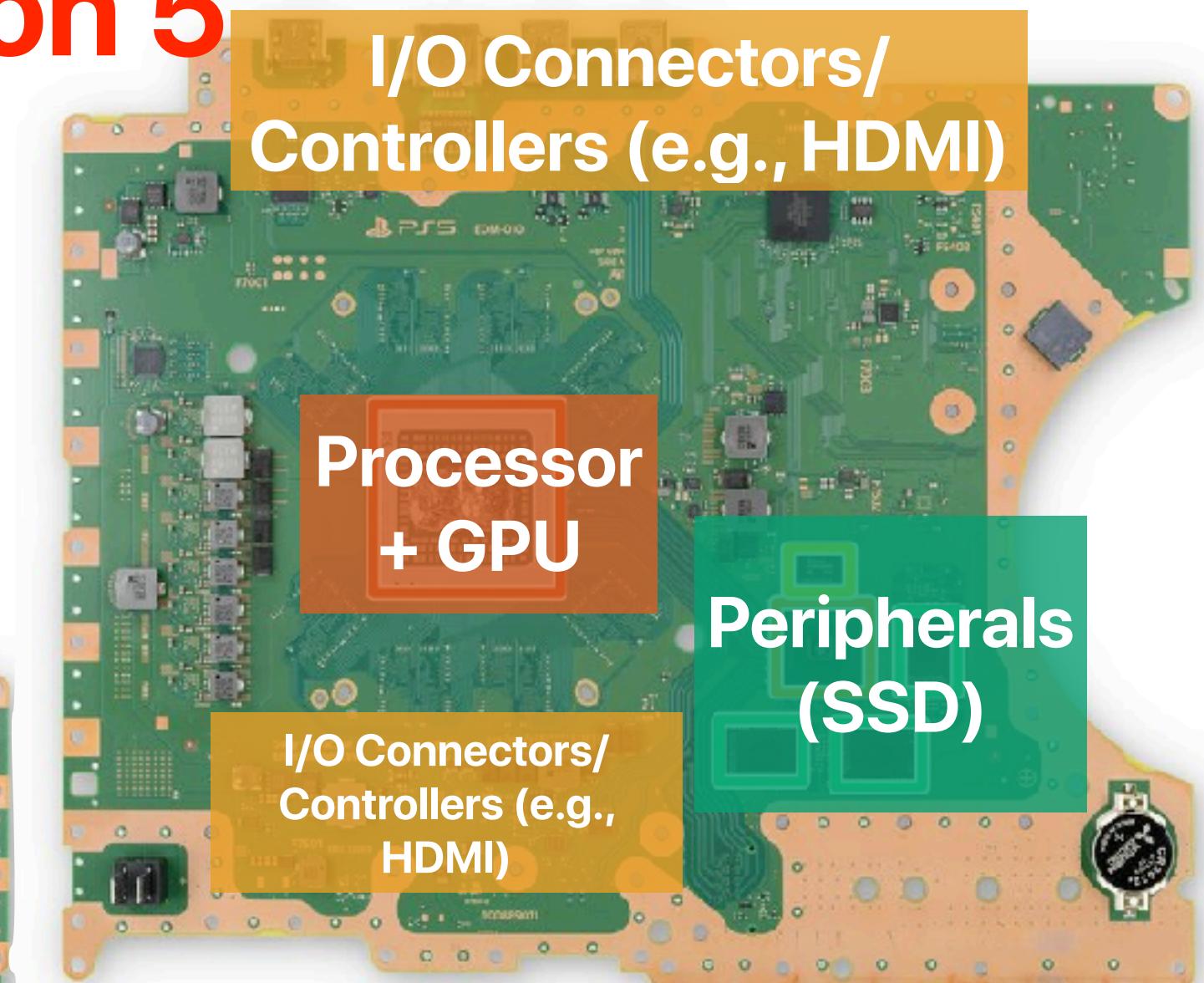
iPhone 15 Pro



Play Station 5



Peripherals
(e.g., SSD)



Processor
+ GPU

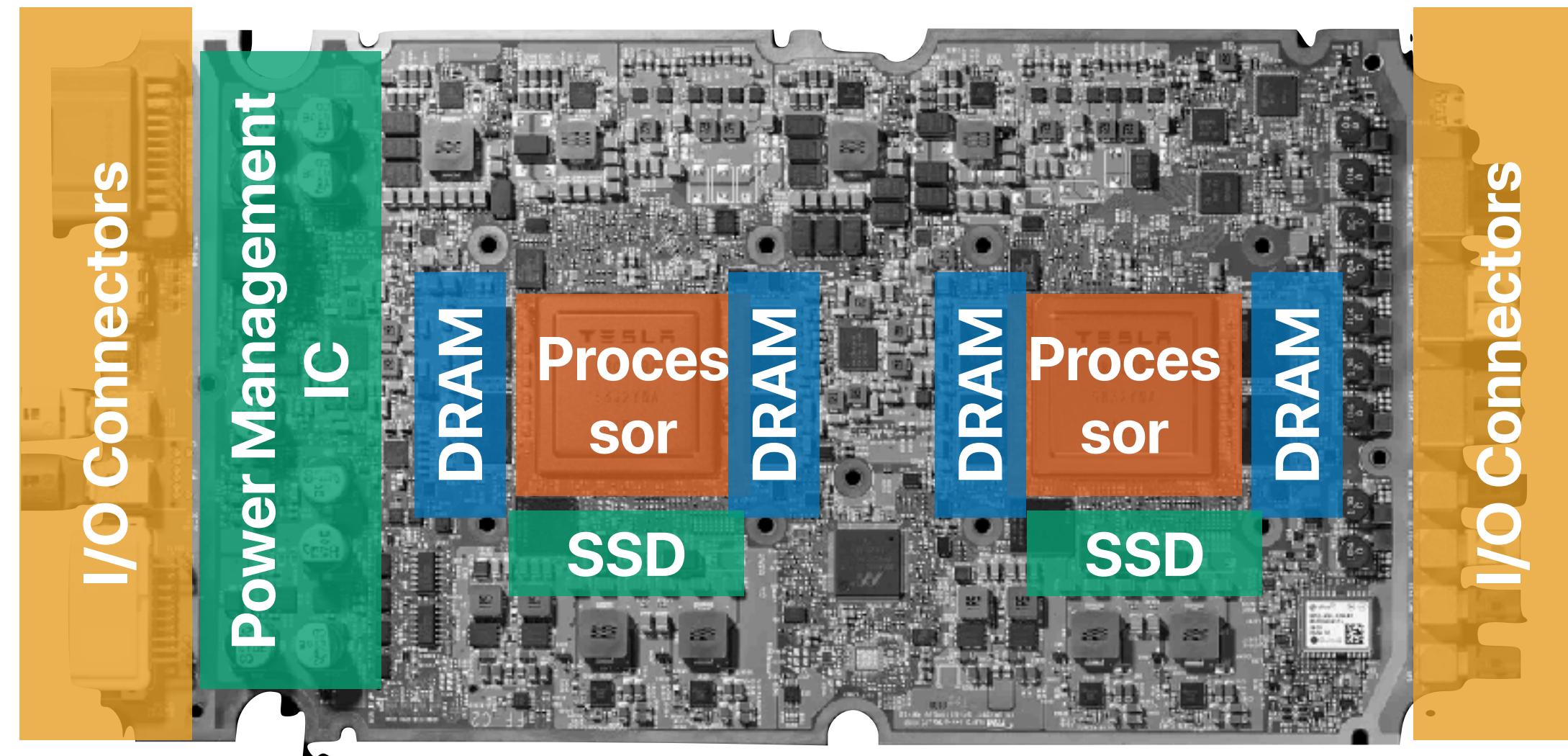
Peripherals
(SSD)

I/O Connectors/
Controllers (e.g.,
HDMI)

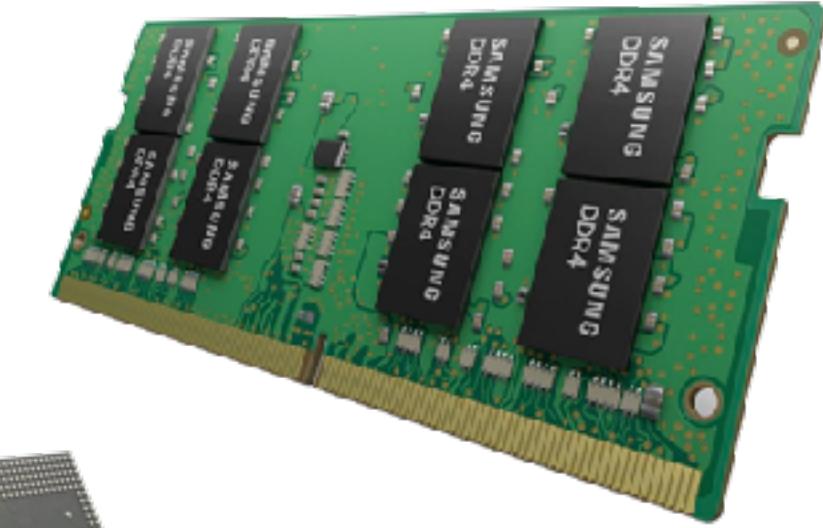
Nintendo Switch



Tesla Model 3



Processors and memory modules are everywhere!



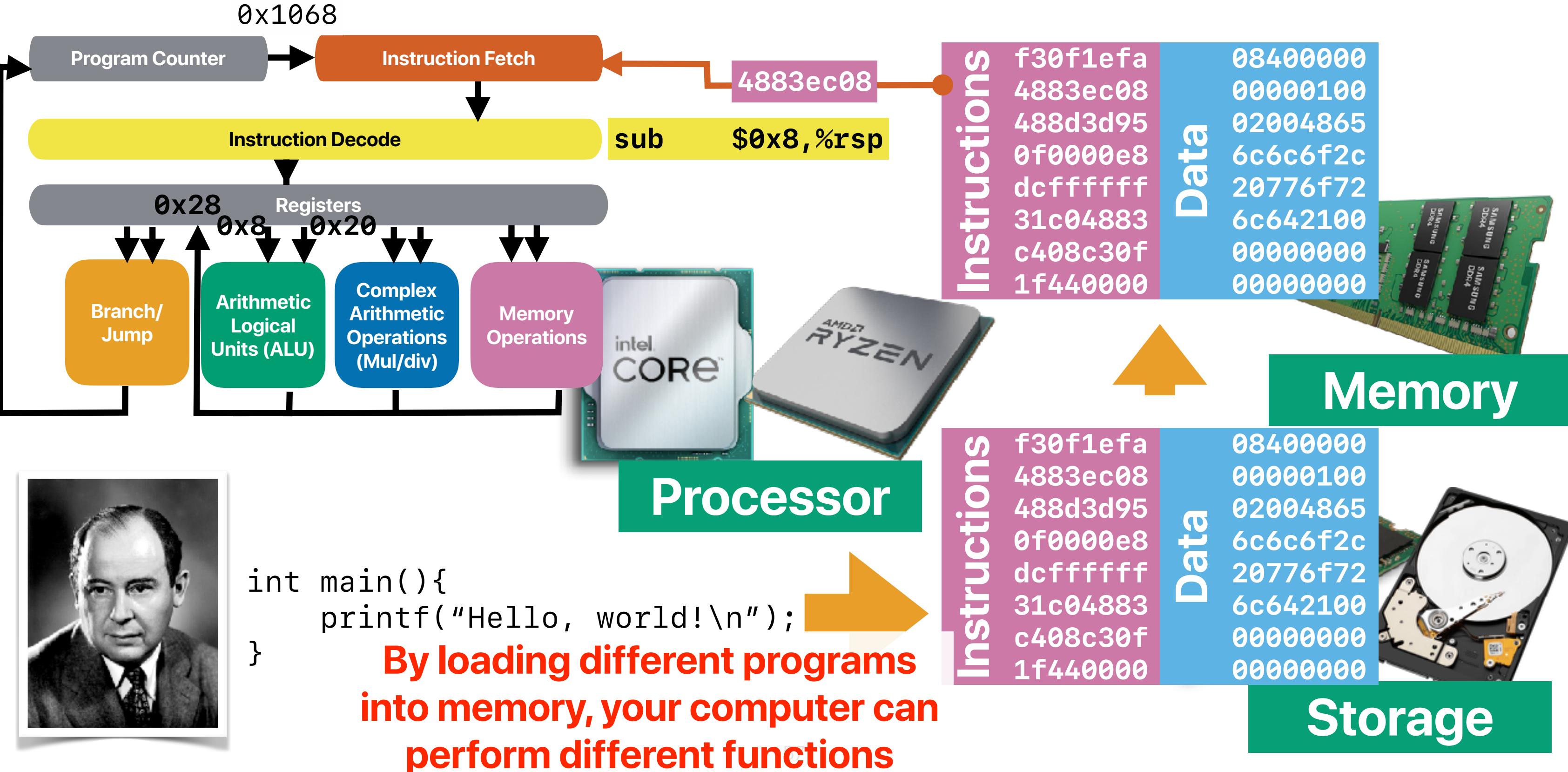
Processors



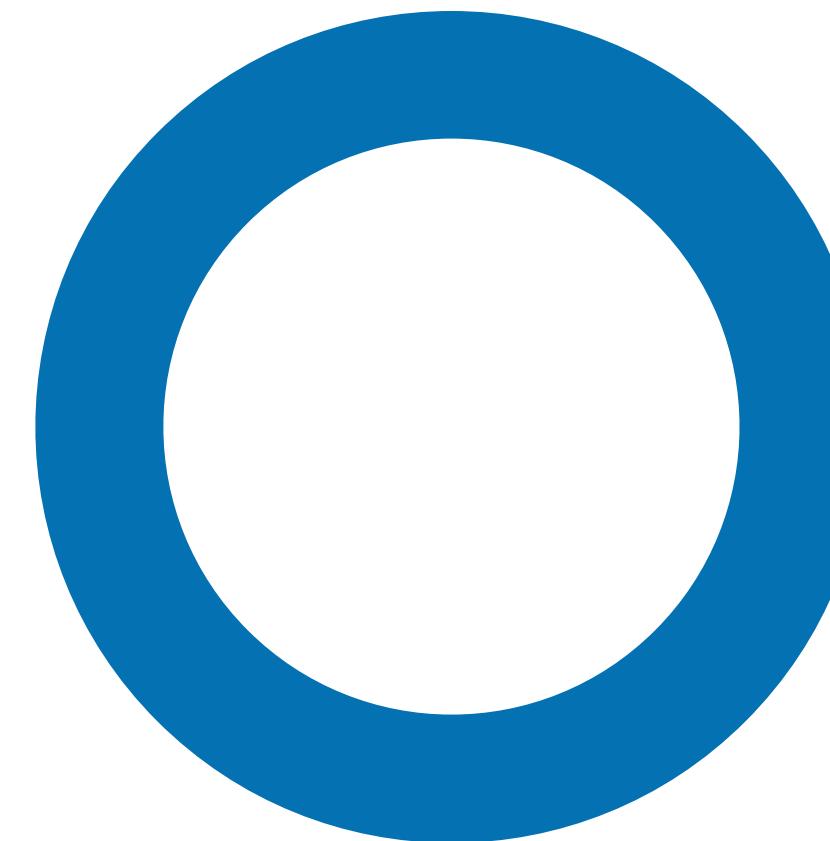
Memory



von Neumann architecture



Processors and memories are essential for most modern general-purpose computers



Start with this simple program in C

```
int A[] =  
{1,2,3,4,5,6,7,8,9,10,1,2,3,4  
,5,6,7,8,9,10};
```

Compiler

Contents of section .data:
0000 01000000 02000000 03000000 04000000
0010 05000000 06000000 07000000 08000000
0020 09000000 0a000000 0b000000 0c000000
0030 03000000 04000000 05000000 06000000
0040 07000000 08000000 09000000 0a000000

control flow
operations
logical operations

```
int main()  
{  
    int i=0, sum=0;  
    for(i = 0; i < 20; i++)  
    {  
        sum += A[i];  
    }  
    return 0;  
}
```

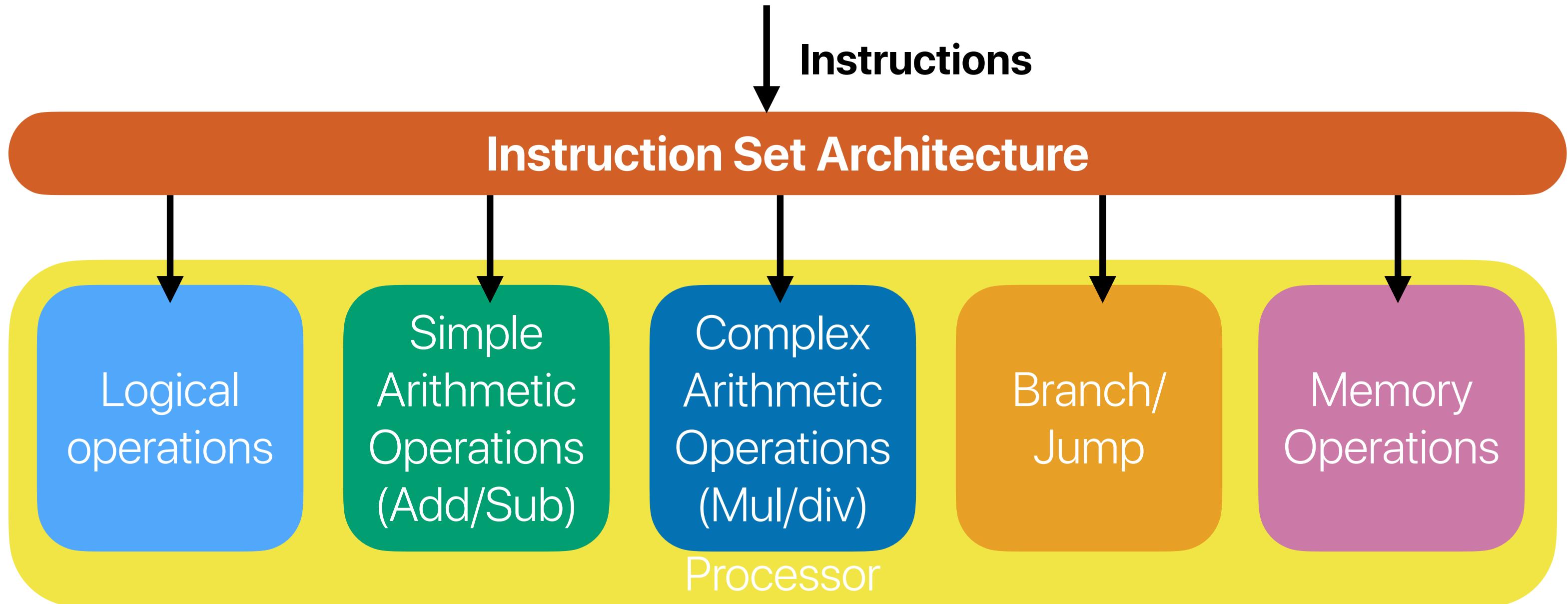
memory access
arithmetic operations

Compiler

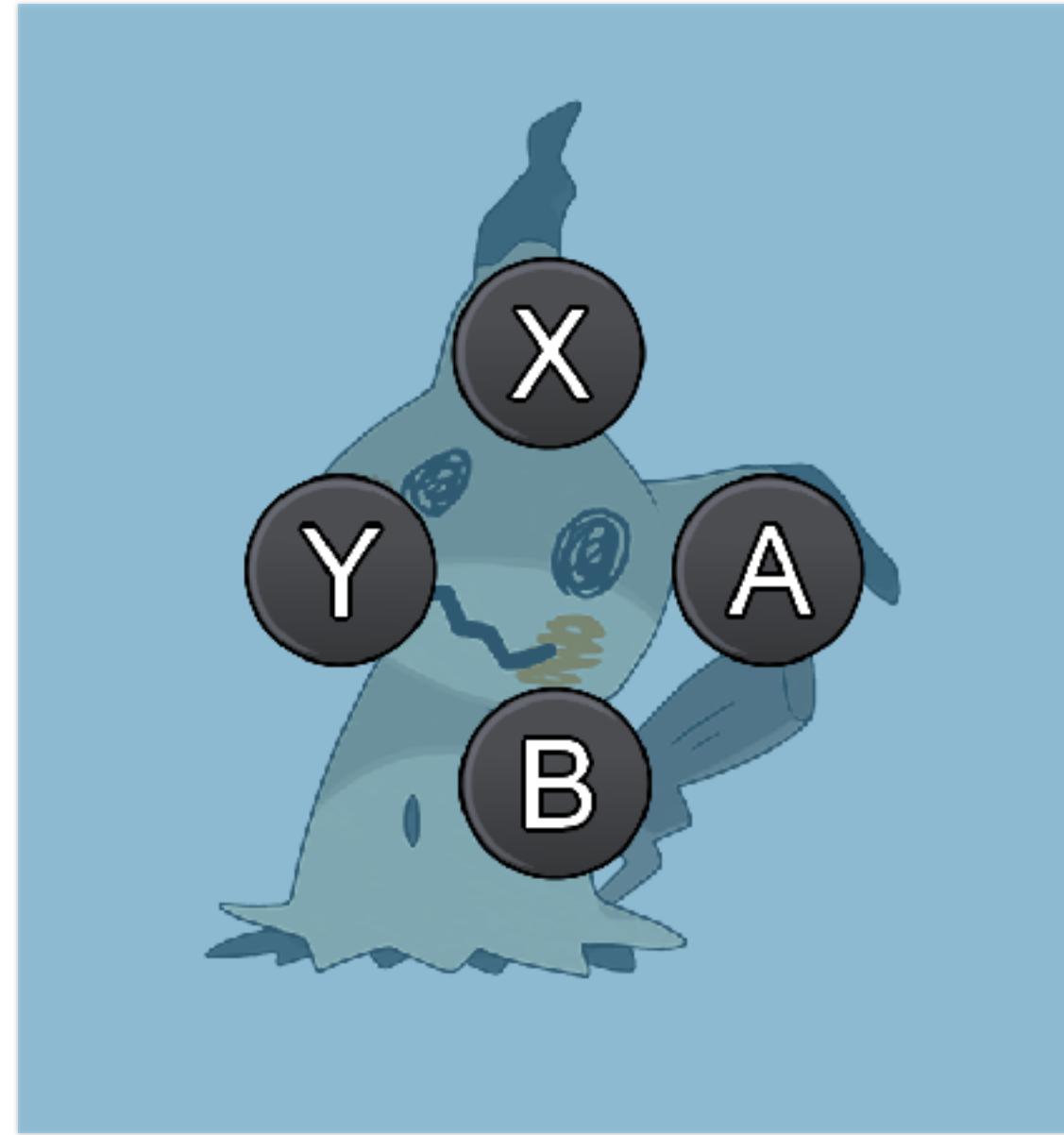
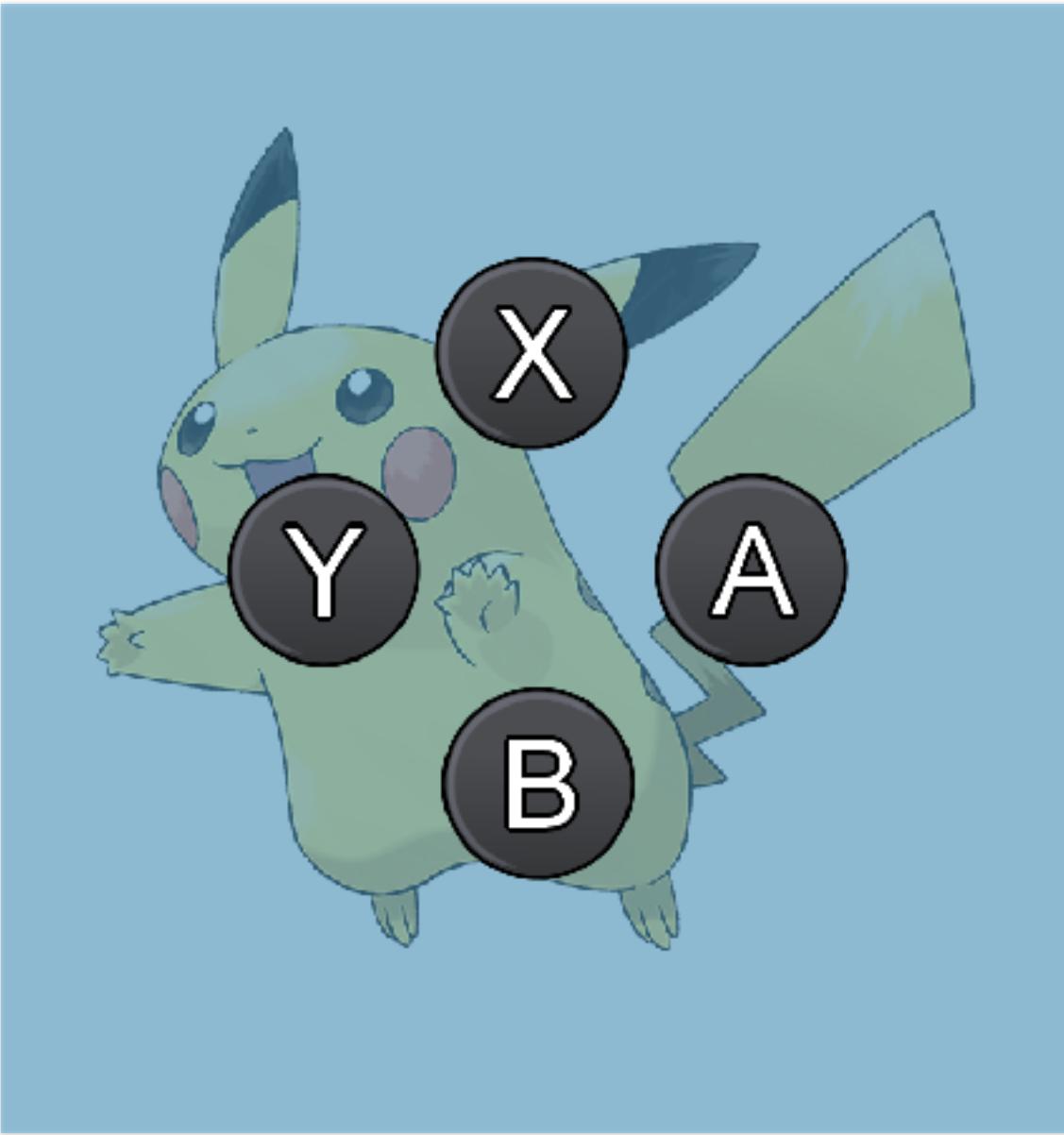
main:
.LFB0:
endbr64
pushq %rbp
movq %rsp, %rbp
movl \$0, -8(%rbp)
movl \$0, -4(%rbp)
movl \$0, -8(%rbp)
jmp .L2
.L2:
cmpl \$19, -8(%rbp)
jle .L3
movl \$0, %eax
popq %rbp
ret

Contents of section .text:
0000 f30f1efa 554889e5 c745f800 000000c7
0010 45fc0000 0000c745 f8000000 00eb1e8b
0020 45f84898 488d1405 00000000 488d0500
0030 0000008b 04020145 fc8345f8 01837df8
0040 137edcb8 00000000 5dc3

Microprocessor — a collection of functional units



ISA — the “abstraction” of processor features



Challenges of von Neumann Architecture

Moore's Law⁽¹⁾

- The number of transistors we can build in a fixed area of silicon doubles every 12 ~ 24 months.

Moore's Law⁽¹⁾

Present and future

By integrated electronics, I mean technologies which are referred to today as well as any additional result in electronics functions supplied as irreducible units. These technologies include the ability to miniaturize electronics equipment, increasingly complex electronic functions in space with minimum weight. Several evolved, including microassembly of individual components, thin-film and semiconductor integrated circuits.

Two-mil squares

With the dimensional tolerances already being employed in integrated circuits, isolated high-performance transistors can be built on centers two thousandths of an inch apart. Such a two-mil square can also contain several kilohms of resistance or

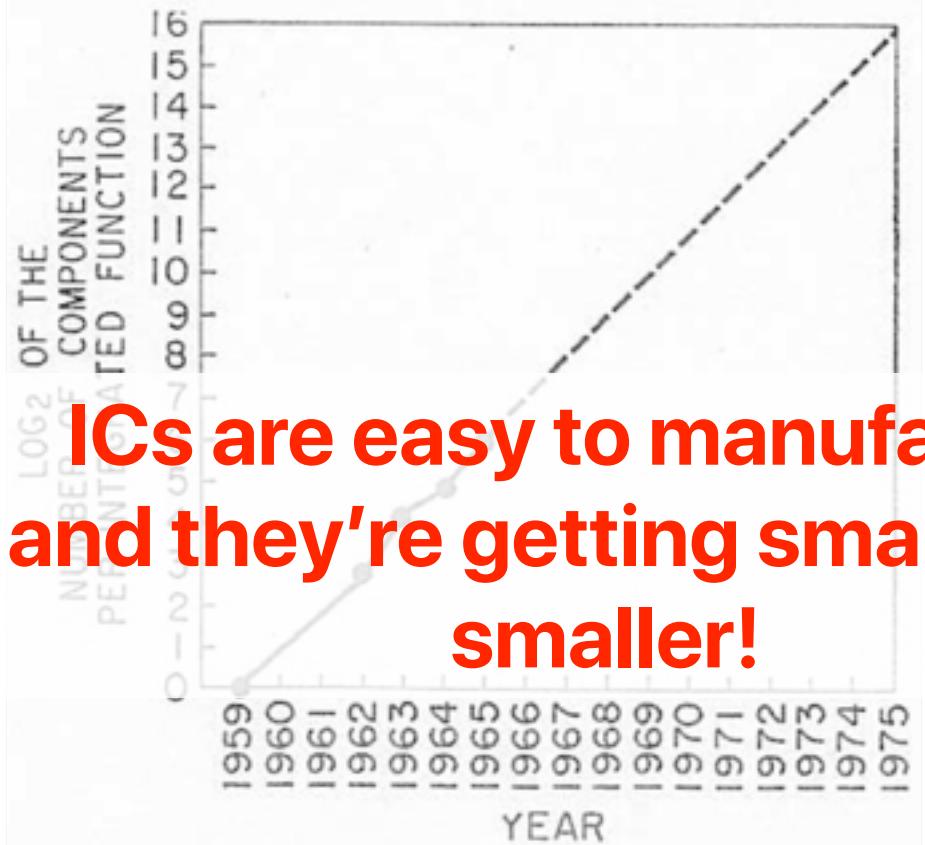
ICs are small

(1) Mo

The establishment

Increasing the yield

There is no fundamental obstacle to achieving device yields of 100%. At present, packaging costs so far exceed the cost of the semiconductor structure itself that there is no incentive to improve yields, but they can be raised as high as is economically justified. No barrier exists comparable to the thermodynamic equilibrium considerations



ICs are easy to manufacture and they're getting smaller and smaller!

Linear circuitry

Integration will not change linear systems as radically as digital systems. Still, a considerable degree of integration will be achieved with linear

circuits. The lack of large-value capacitors and

inductors makes it difficult to implement linear

systems. However, the lack of discrete compo-

nents—such as resistors, capacitors, and in-

ductors—makes it easier to implement linear

systems. The lack of discrete components makes

it easier to implement linear systems. The lack of discrete components makes it easier to imple-

ment linear systems. The lack of discrete com-

ponents onto integrated circuits', Electronics 38 (8).

Reliability count

In almost every field of electronics, ICs have demonstrated higher reliability than discrete components. At the same level of production—low compared to that of discrete components—it offers reduced systems cost, and in many systems improved performance has been realized.

ICs are more reliable

Heat problem

Will it be possible to remove the heat generated by tens of thousands of components in a single silicon chip?

Establish
Moore's
importance
historic

Day of reckoning

Clearly, we will be able to build such component-crammed equipment. Next, we ask under what circumstances we should do it. The total cost of making a particular system function must be minimized. To do so, we could amortize the engineering over several identical items, or evolve flexible techniques for the engineering of large functions so that no disproportionate expense need be borne by a particular array. Perhaps newly devised de-

sign automation procedures could translate from any special engineering.

Designing ICs can be easy

Moore's Law Alive and Well



RTX 4080 Why so expen\$ive?

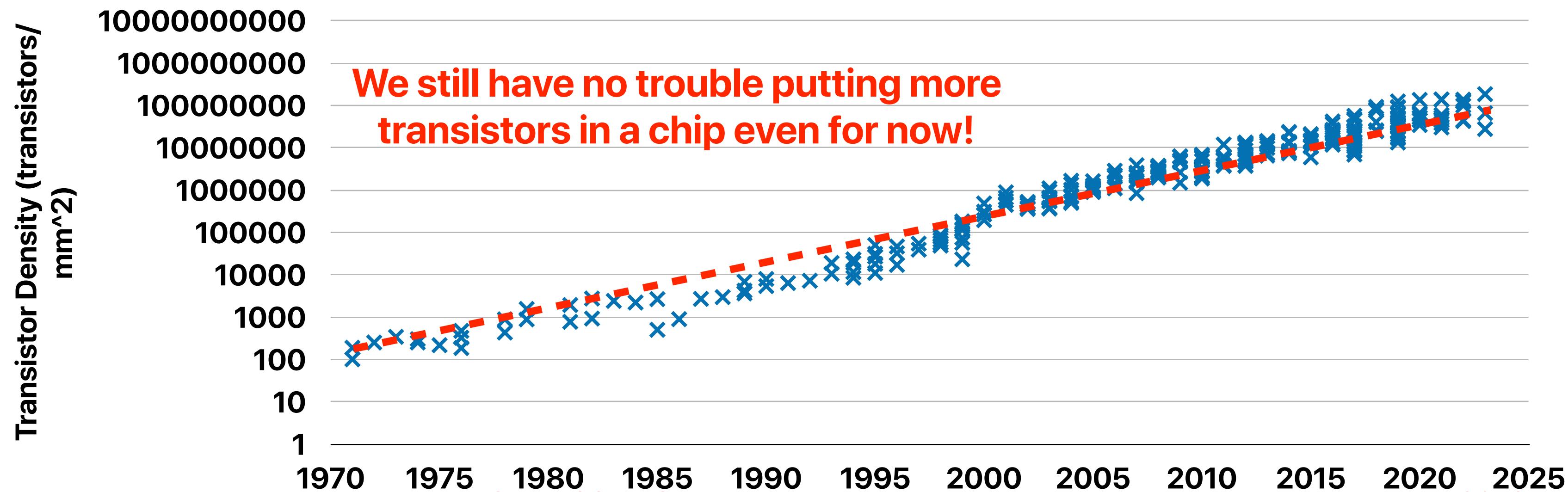


*"Moore's Law
Is Dead"*



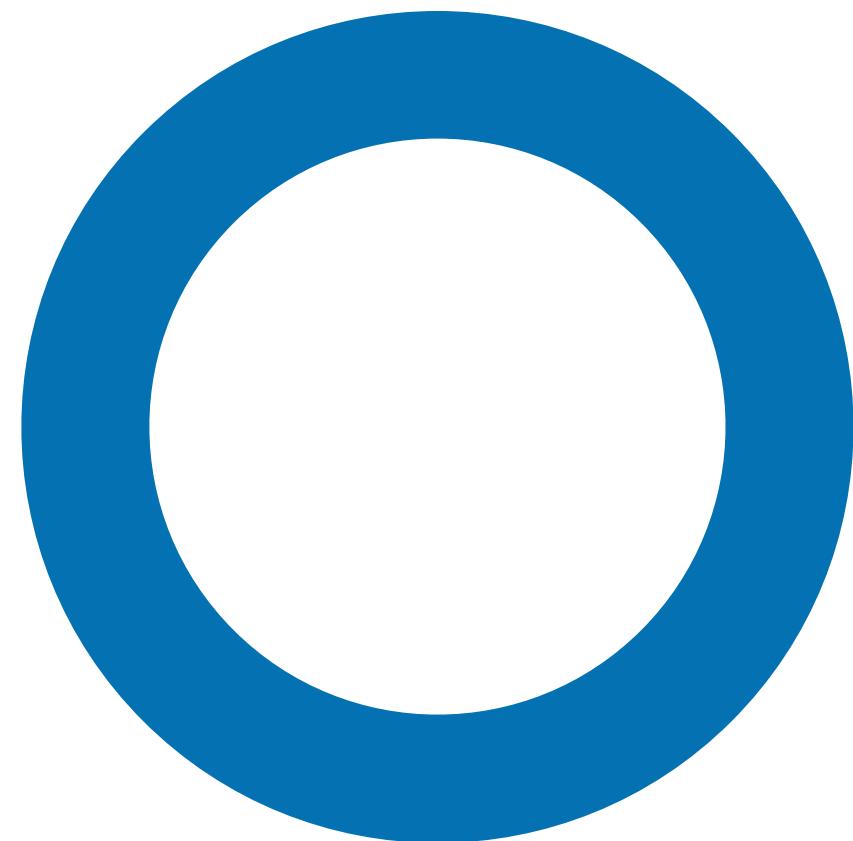
Moore's Law⁽¹⁾

- The number of transistors we can build in a fixed area of silicon doubles every 12 ~ 24 months.
- Moore's Law "was" the most important driver for historic CPU performance gains



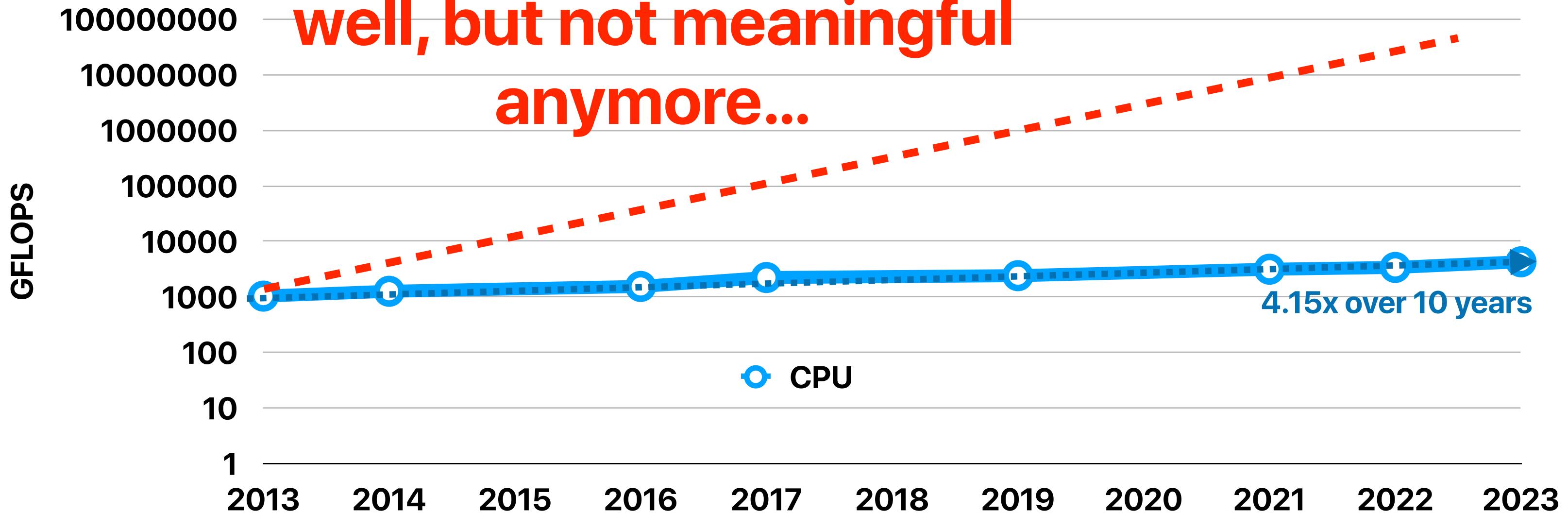
(1) Moore, G. E. (1965), 'Cramming more components onto integrated circuits', Electronics 38 (8).

Moore's Law is still alive

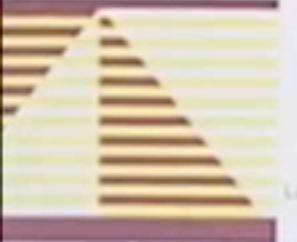


CPU Performance v.s. Moore's Law

Moore's Law is alive and
well, but not meaningful
anymore...

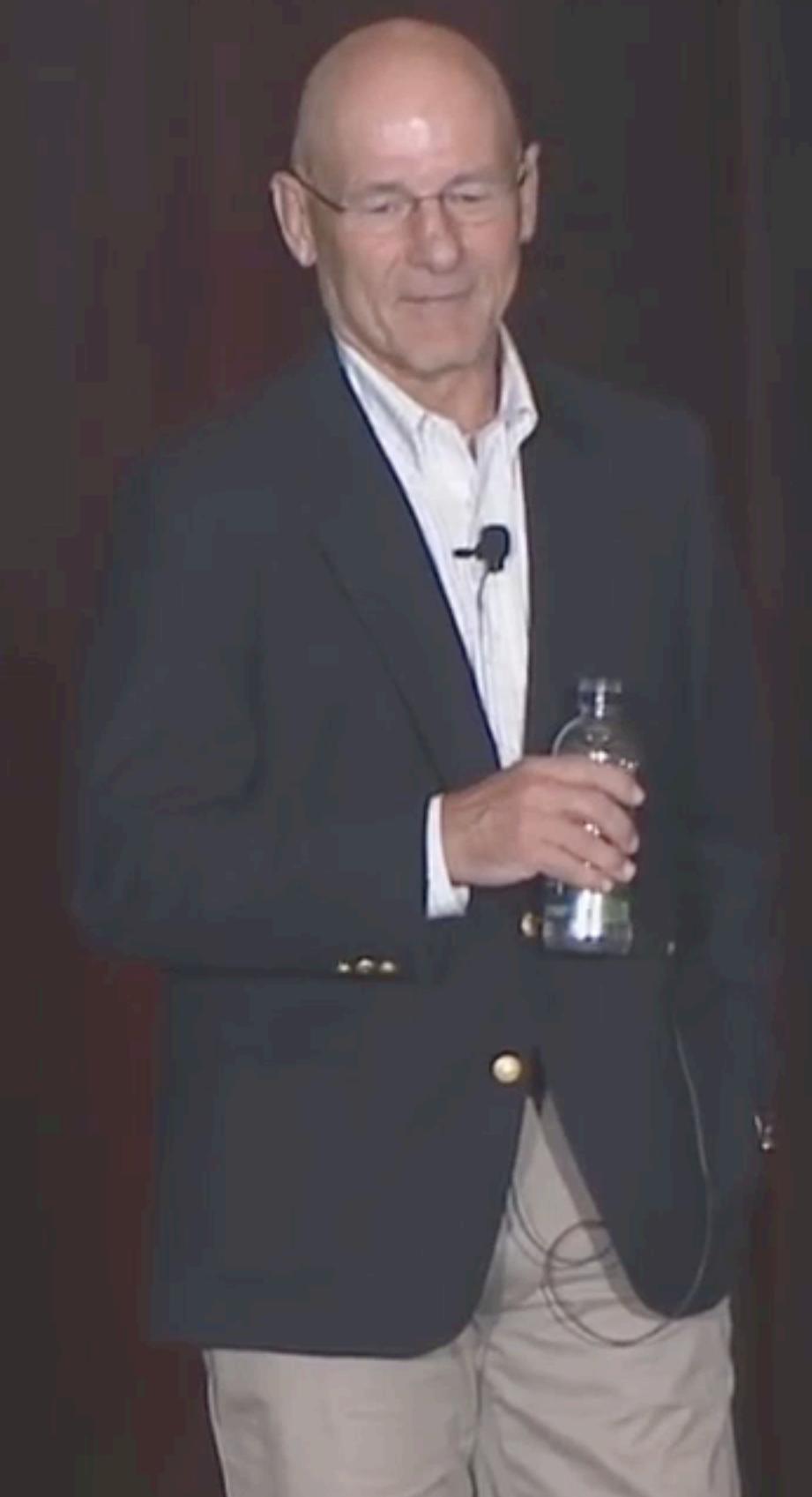
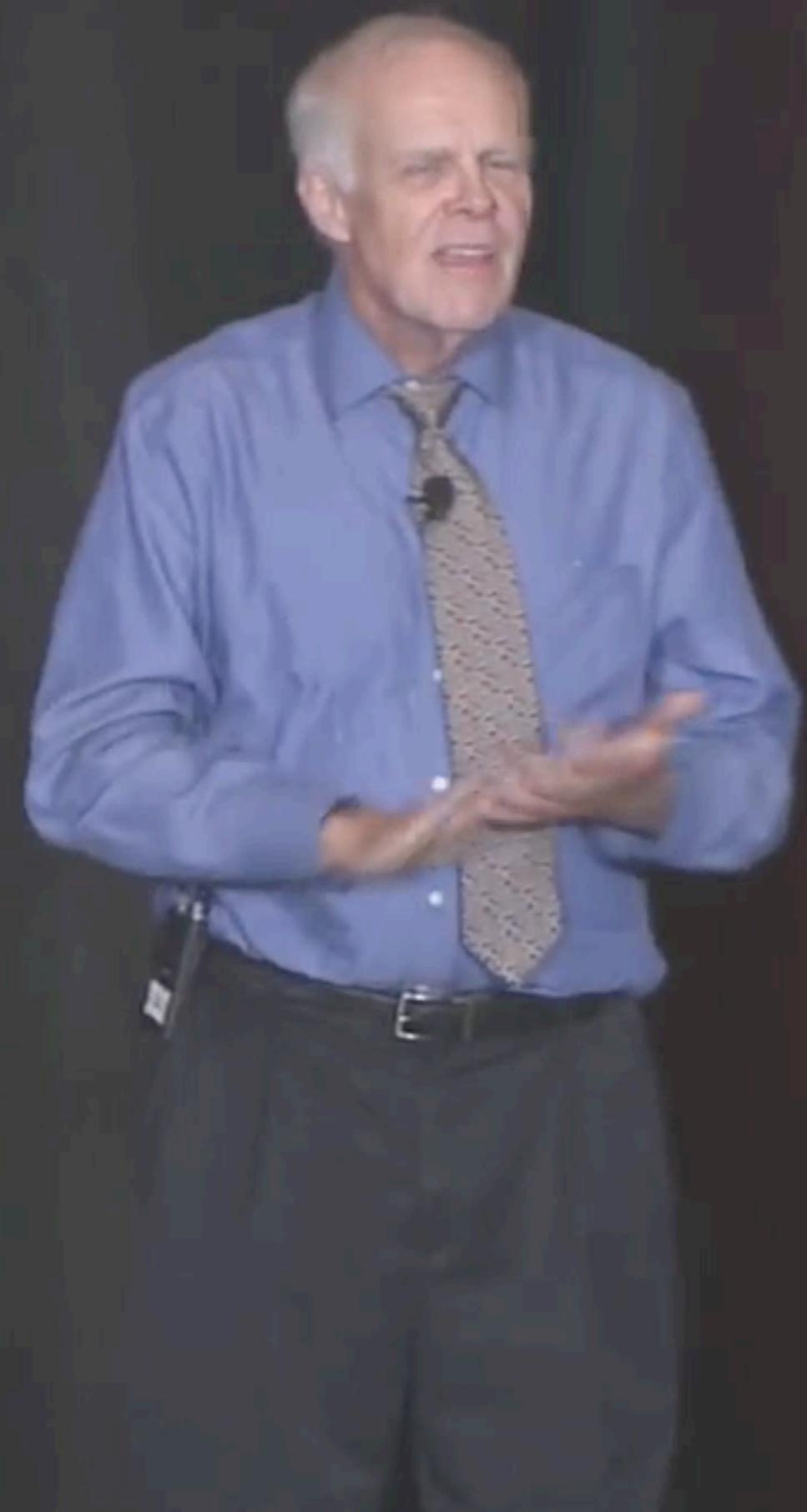


<https://ourworldindata.org/grapher/artificial-intelligence-training-computation>



The 45th
ACM/IEEE
International
Symposium
on Computer
Architecture
Los Angeles, USA

ISCA 2018
uring Lecture



Power consumption per transistor

GB200 GPU Is The Full Blackwell Specs, 500W More Power Than Hopper

During the launch, there was a particularly big confusion surrounding all the Blackwell GPU and platform variants. Jensen stated that Blackwell isn't a GPU, it's an entire platform & the platform has a range of products but they are still based on GPUs. As of right now, NVIDIA has announced three official Blackwell GPU variants.

These include the flagship and full-spec B200 which is being used by the [GB200 Superchip platforms](#). This chip has the highest-rated computing capabilities and has a maximum TDP of 1200W. This is 500 Watts more than the Hopper H100 which featured a 700W TDP. The entire Superchip is equipped with two of these B200 GPUs and a Grace CPU for up to 2700W power (1200W x 2 for B200 + 300W CPU/IO).

World's Most Powerful Chip — Packed with 208 billion transistors, Blackwell-architecture GPUs are manufactured using a custom-built 4NP TSMC process with two-reticle limit GPU dies connected by 10 TB/second chip-to-chip link into a single, unified GPU.

61 billion transistors

The XCC has [61 billion](#) transistors. The MCC die for Emerald Rapids has up to 32 cores exposed to the outside world, and probably has 36 cores in the design, again to improve yield. Dec 14, 2023



The Next Platform

<https://www.nextplatform.com> › Compute

[Intel "Emerald Rapids" Xeon SPs: A Little More Bang, A Little ...](#)

The top-line Emerald Rapids chip, the 5th gen Xeon 8592+, will have 64 cores, an improvement from the 60 cores in Sapphire Rapids. The chip will operate at a 1.9GHz frequency that can max out at 3.9GHz in turbo mode. It has 320MB of cache, draws [350](#) watts of power, and fits into two-socket systems. It costs a whopping \$11,600.

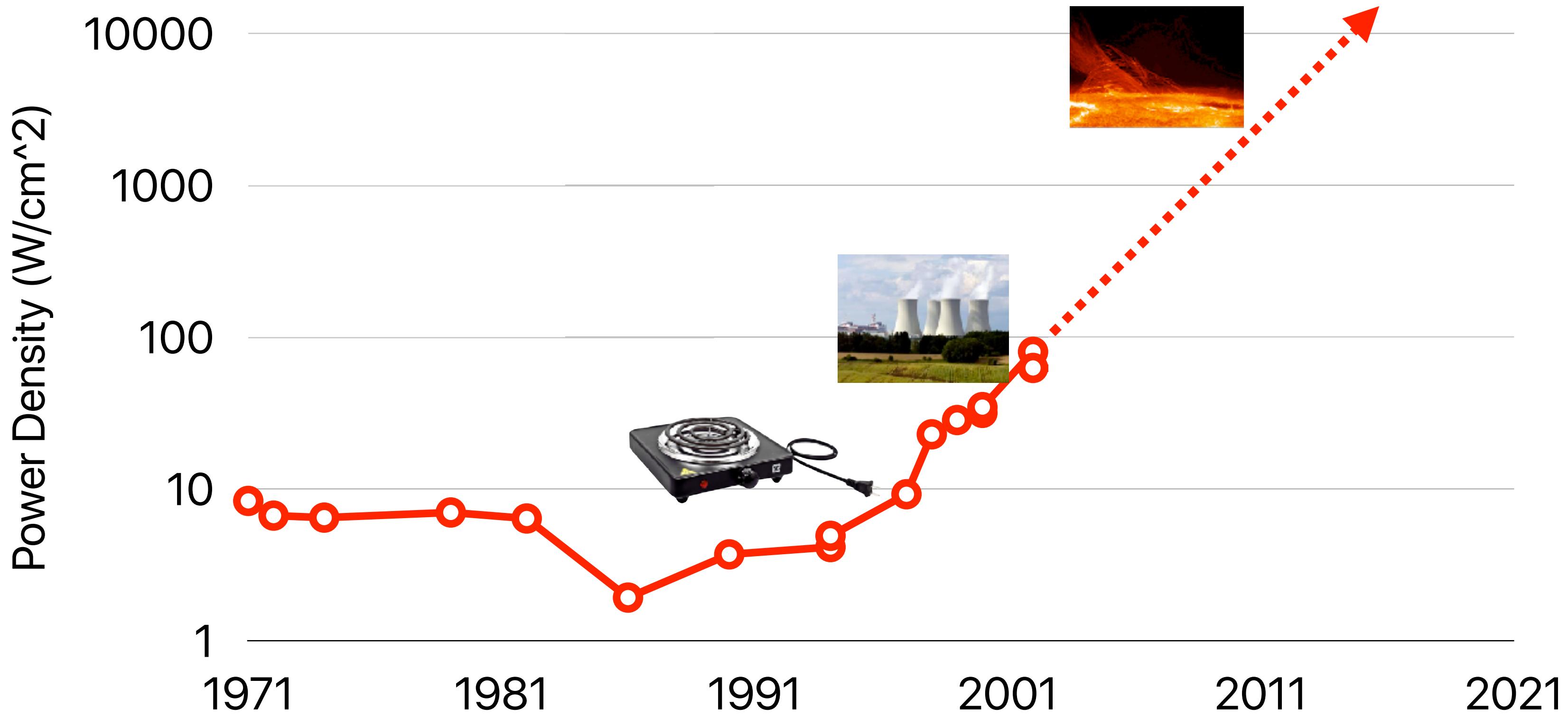
5.77 W/1B transistors

TSMC 4nm

5.73 W/1B transistors

Intel 7nm

Power Density of Processors

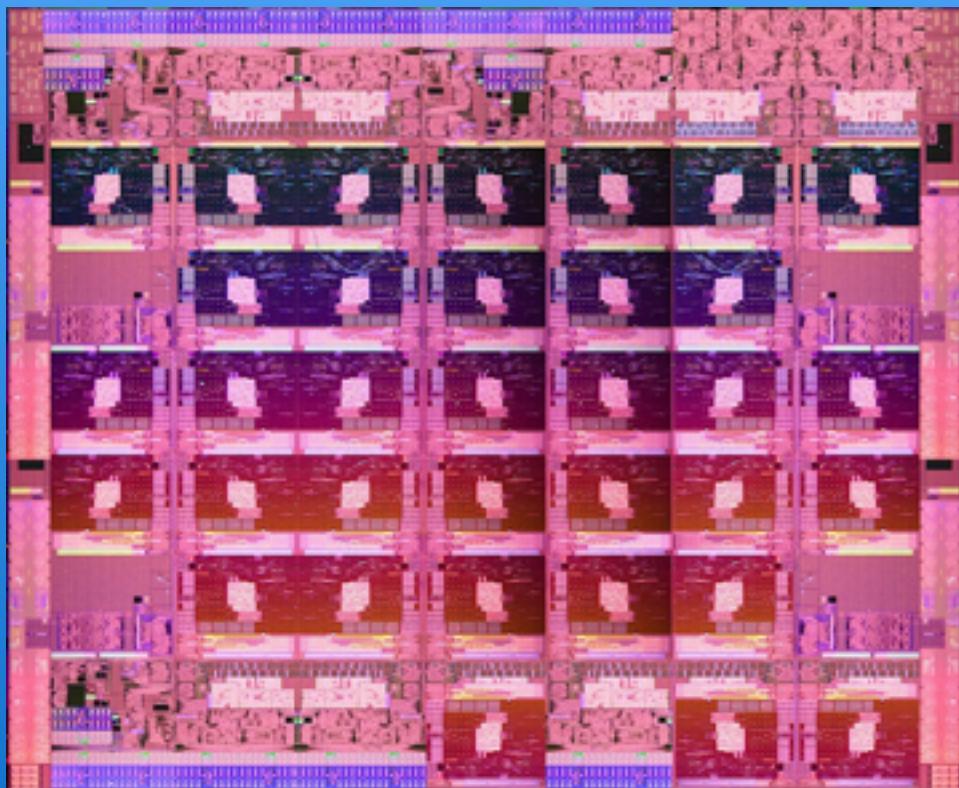


**The smaller size of a transistor,
the smaller power consumption of it**



Alternatives to scaling single cores

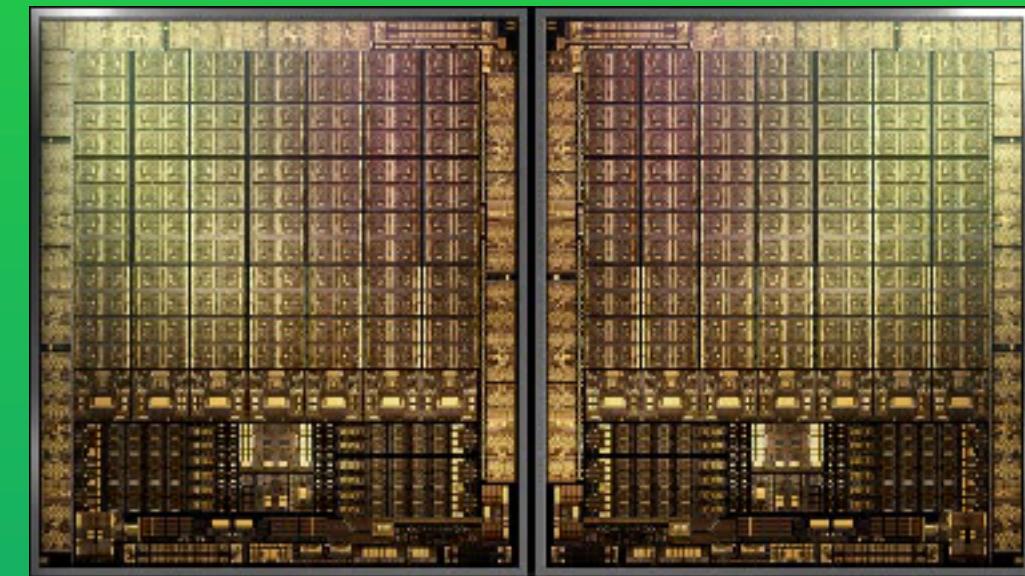
Multicore Processors



34-core Intel
Sapphire Rapids

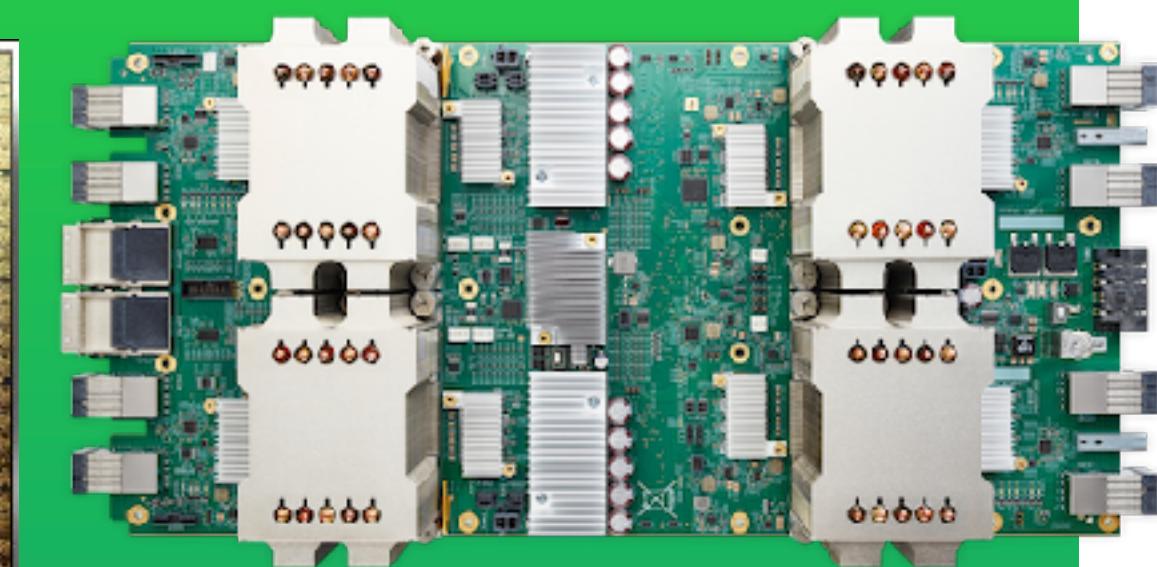
Thread-level parallelism

Hardware Accelerators



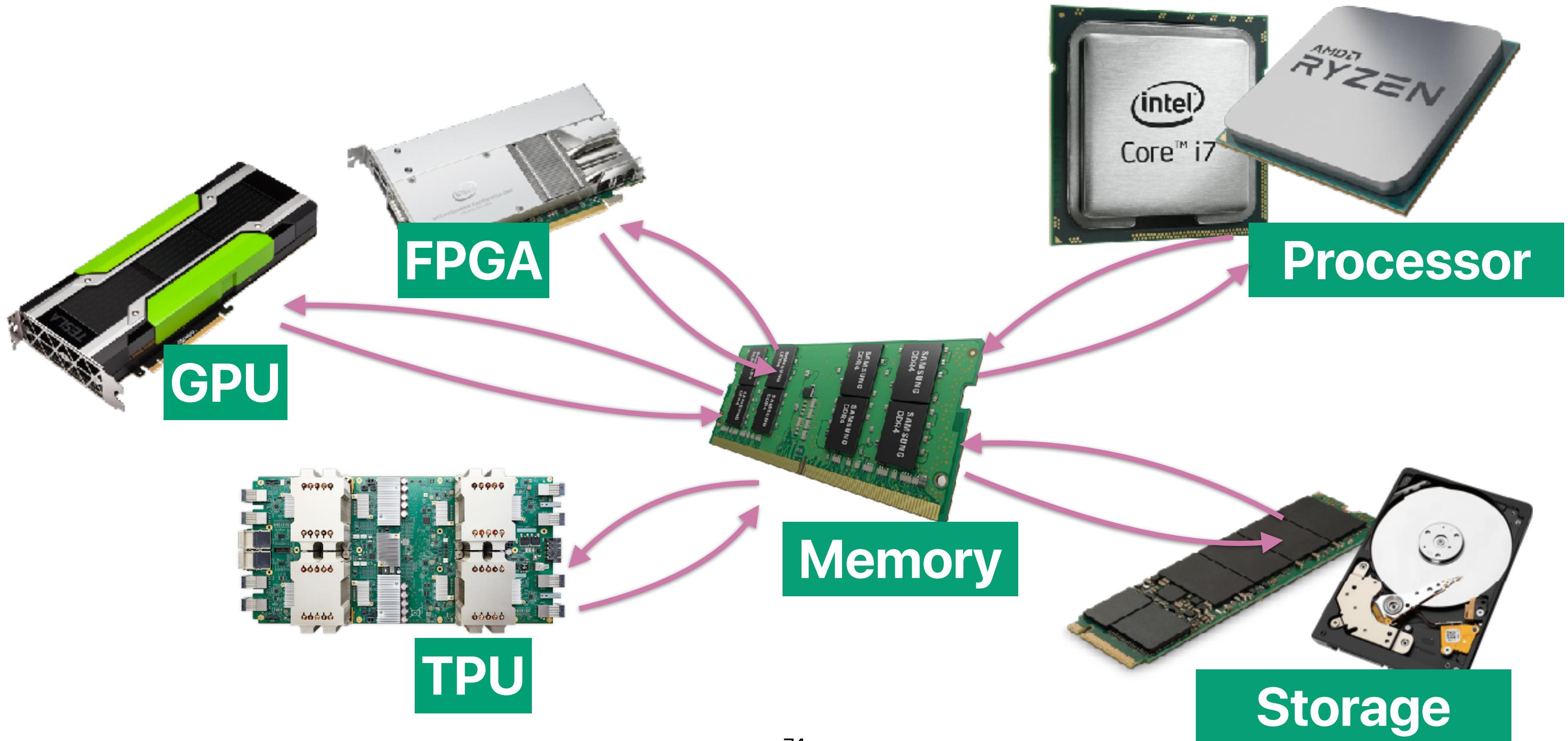
NVIDIA H100
Graphic Processing Units

Data-level parallelism



Google
Tensor Processing Units

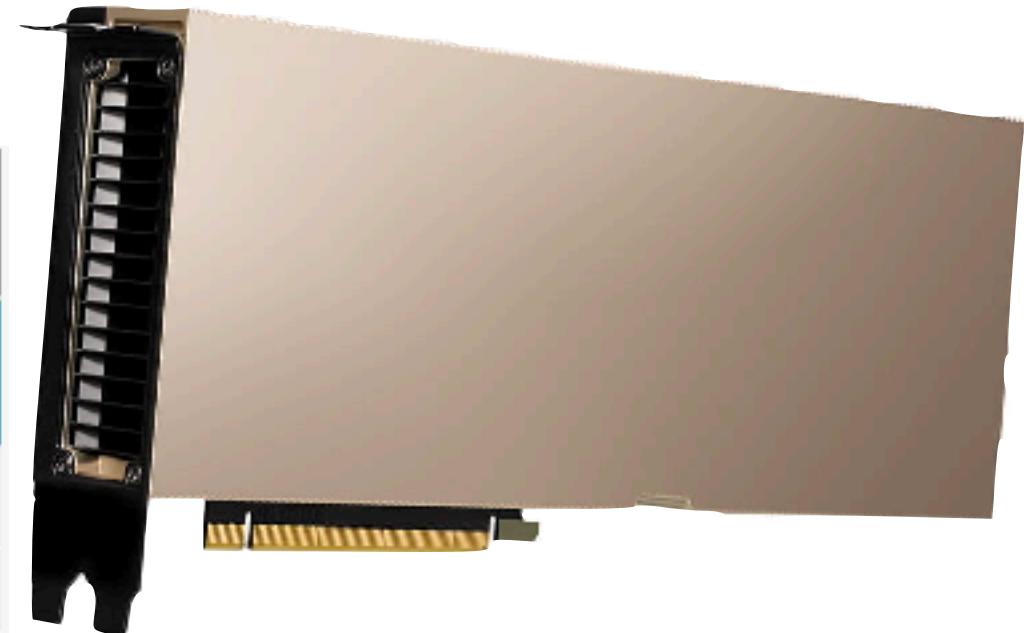
Heterogeneous Computer Architecture



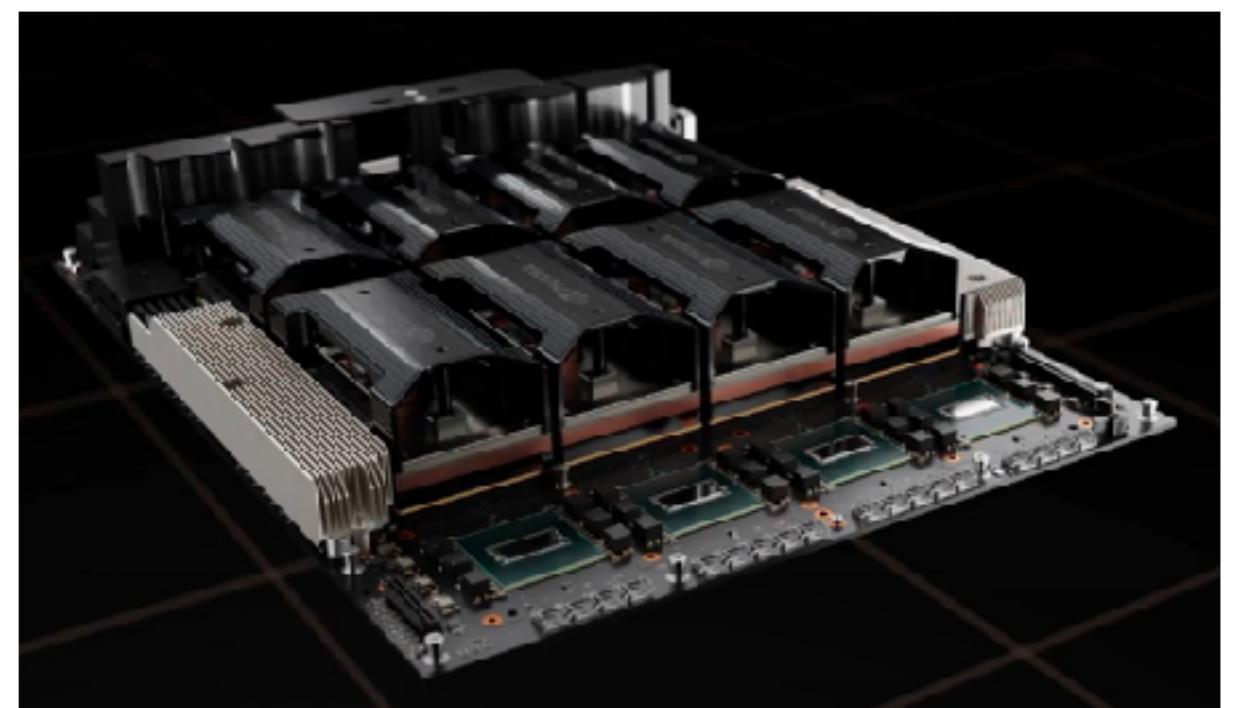
But still limited by the power consumption

	H100	A100 (80GB)	V100
NVIDIA Accelerator Specification Comparison			
	H100	A100 (80GB)	V100
FP32 CUDA Cores	16896	6912	5120
Tensor Cores	528	432	640
GPU	GH100 (814mm ²)	GA100 (826mm ²)	GV100 (815mm ²)
Transistor Count	80B	1.46x 1.75x	54.2B
TDP	700W	400W	21.1B 300W/350W
Manufacturing Process	TSMC 4N	TSMC 7N	TSMC 12nm FFN
Interface	SXM5	SXM4	SXM2/SXM3
Architecture	Hopper	Ampere	Volta
Transistor Count	80B	54.2B	21.1B
TDP	700W	400W	300W/350W
Manufacturing Process	TSMC 4N	TSMC 7N	TSMC 12nm FFN
Interface	SXM5	SXM4	SXM2/SXM3
Architecture	Hopper	Ampere	Volta

B Transistors B Transistors



<https://www.workstationspecialist.com/product/nvidia-tesla-a100/>



<https://www.servethehome.com/wp-content/uploads/2022/03/NVIDIA-GTC-2022-H100-in-HGX-H100.jpg>

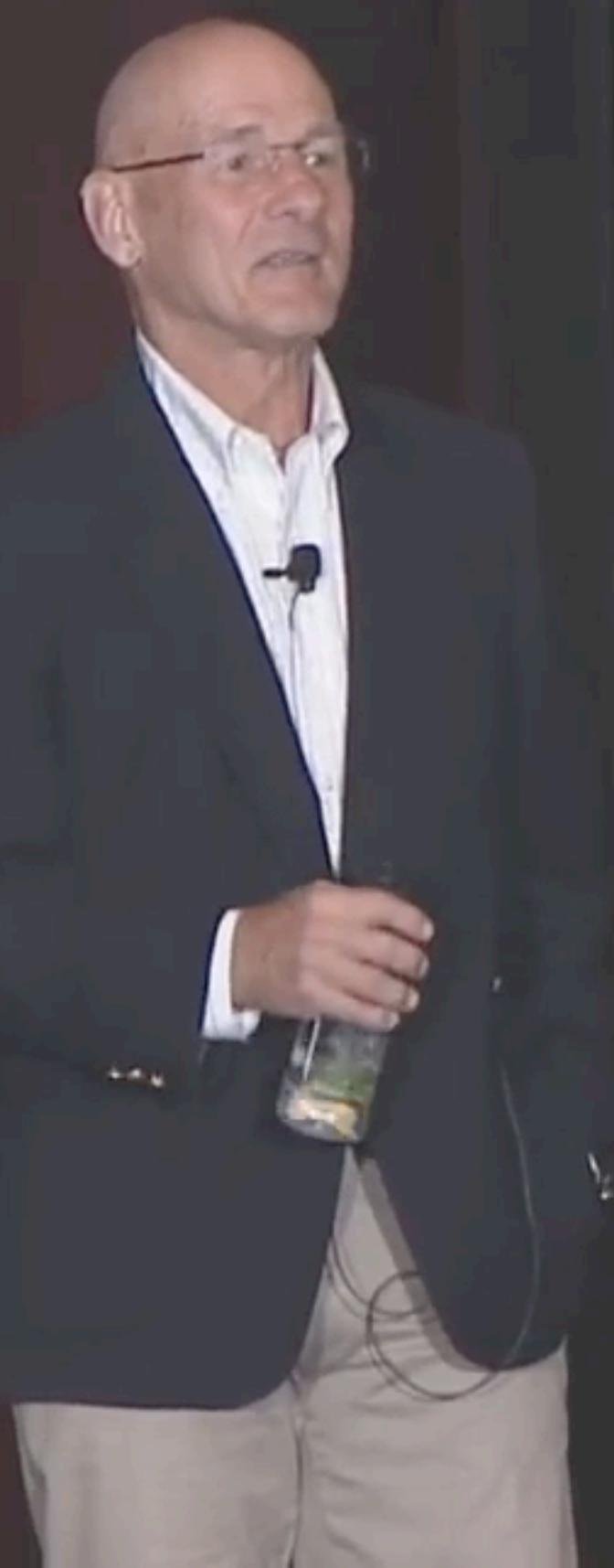
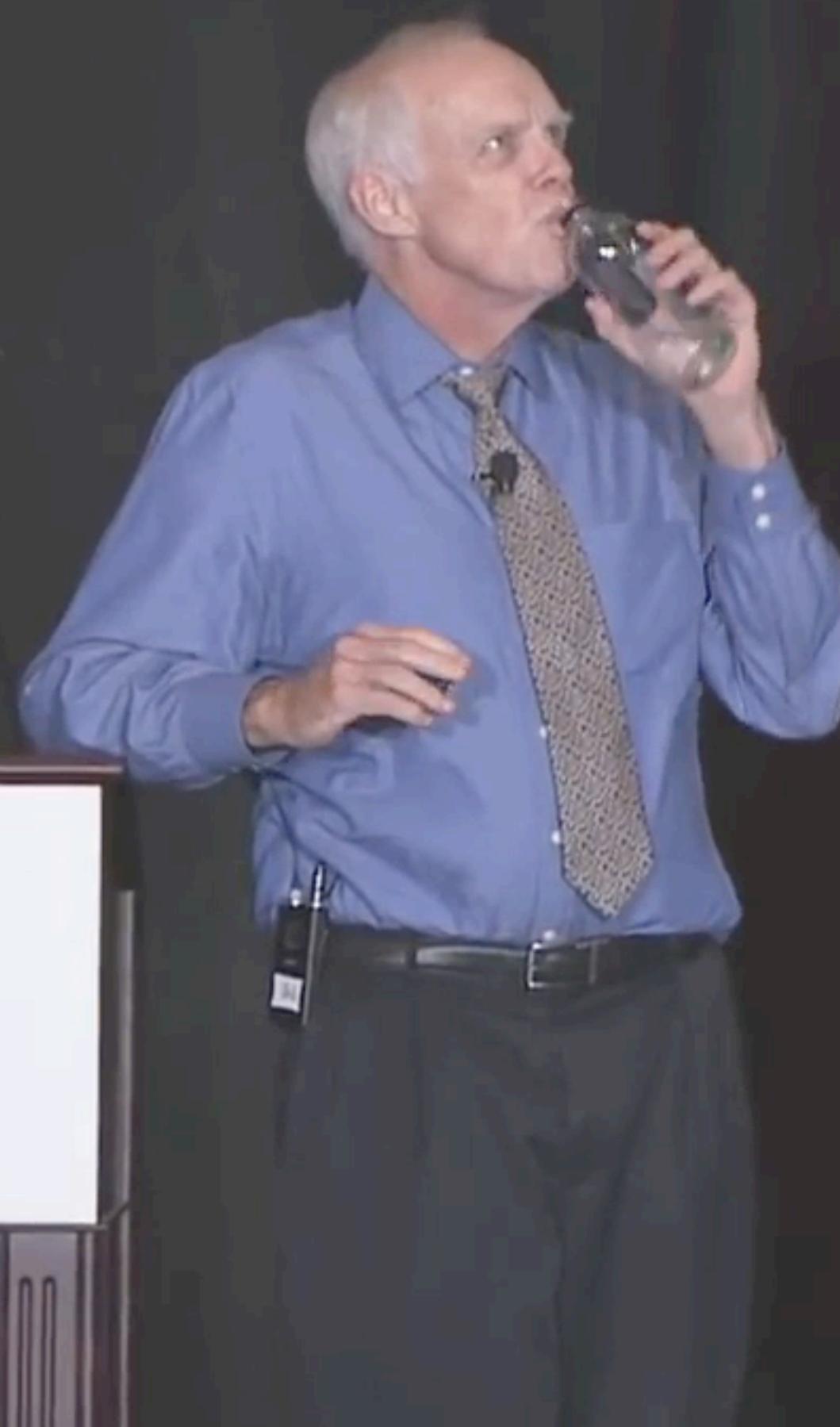
GPUs are not limited by Moore's Law



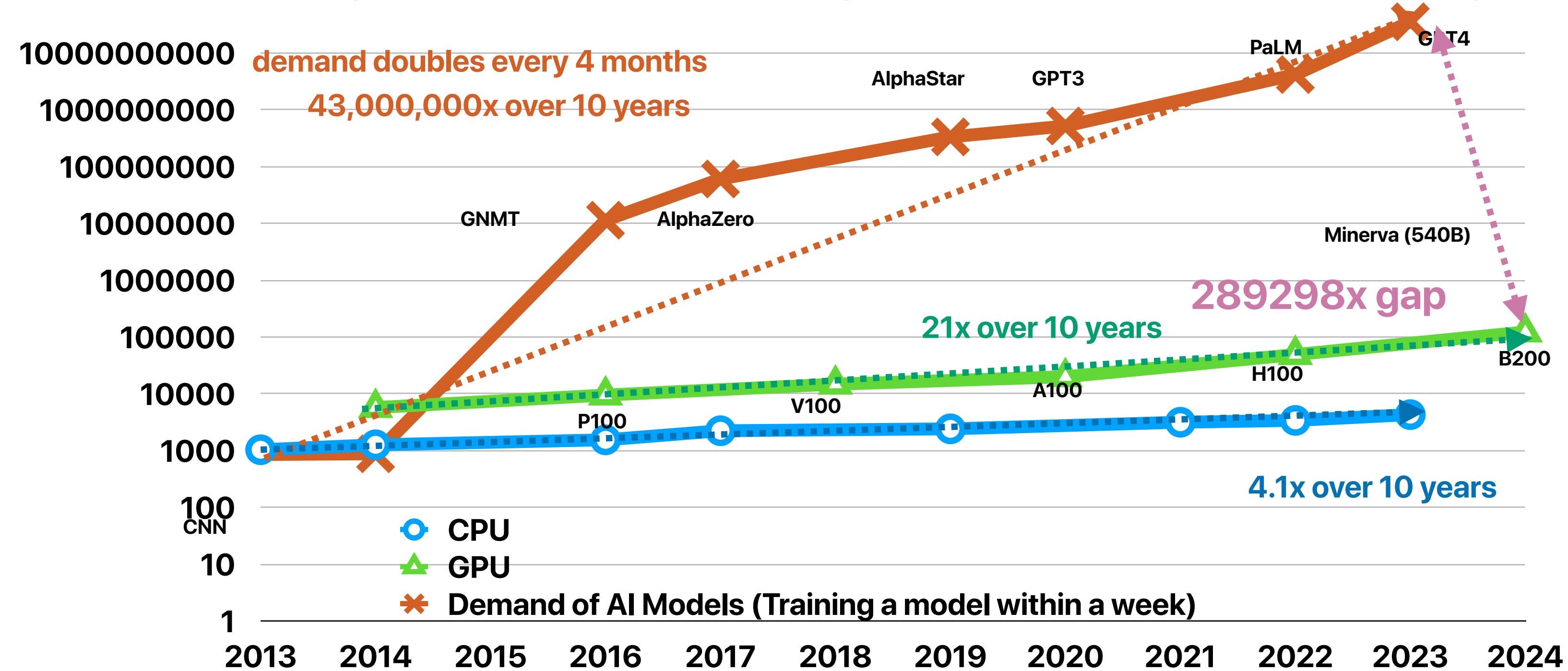


The 45th
ACM/IEEE
International
Symposium
on Computer
Architecture
Los Angeles, USA

A 2018
g Lecture



Mis-matching AI/ML demand and general-purpose processing



<https://ourworldindata.org/grapher/artificial-intelligence-training-computation>

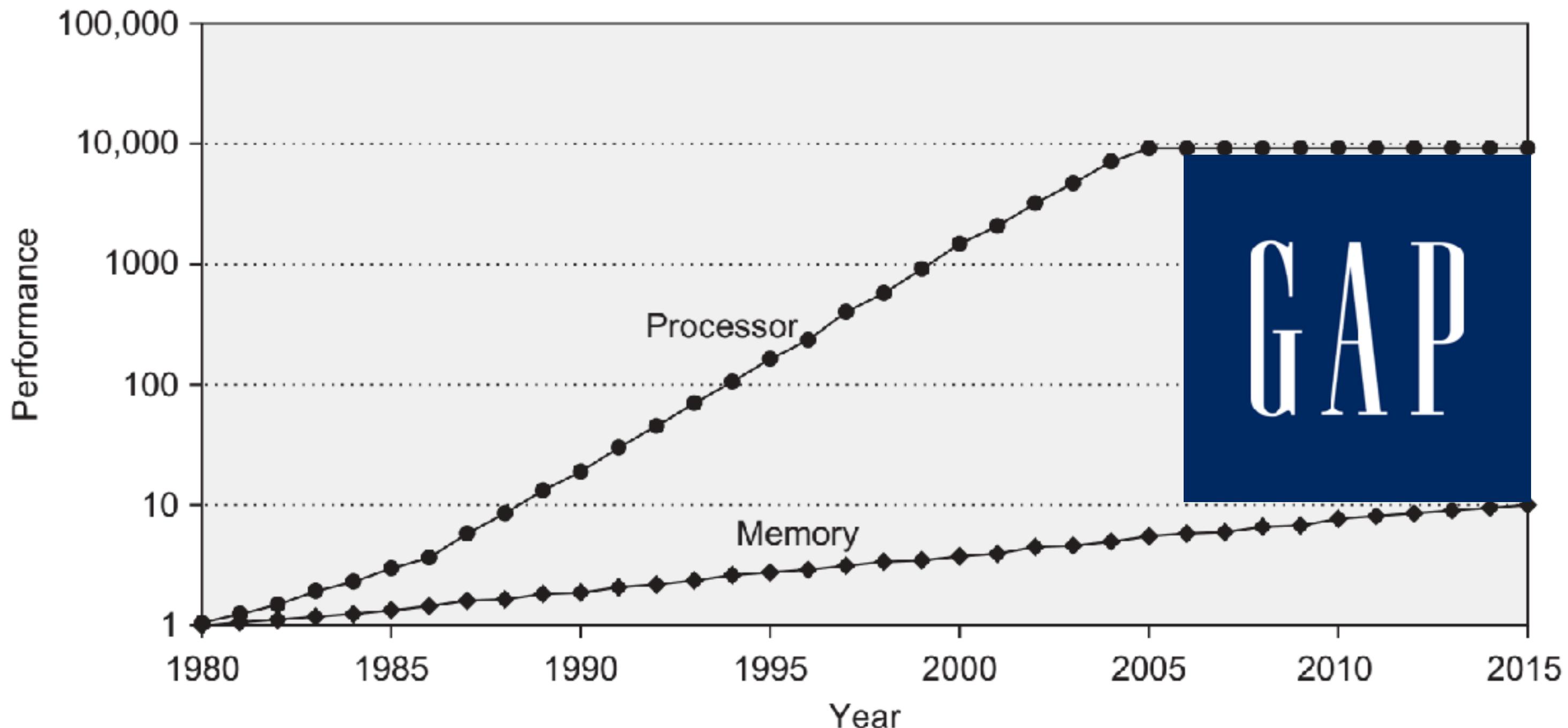
**The performance growth of recent GPU architectures
can match the demand of efficient AI/ML training**



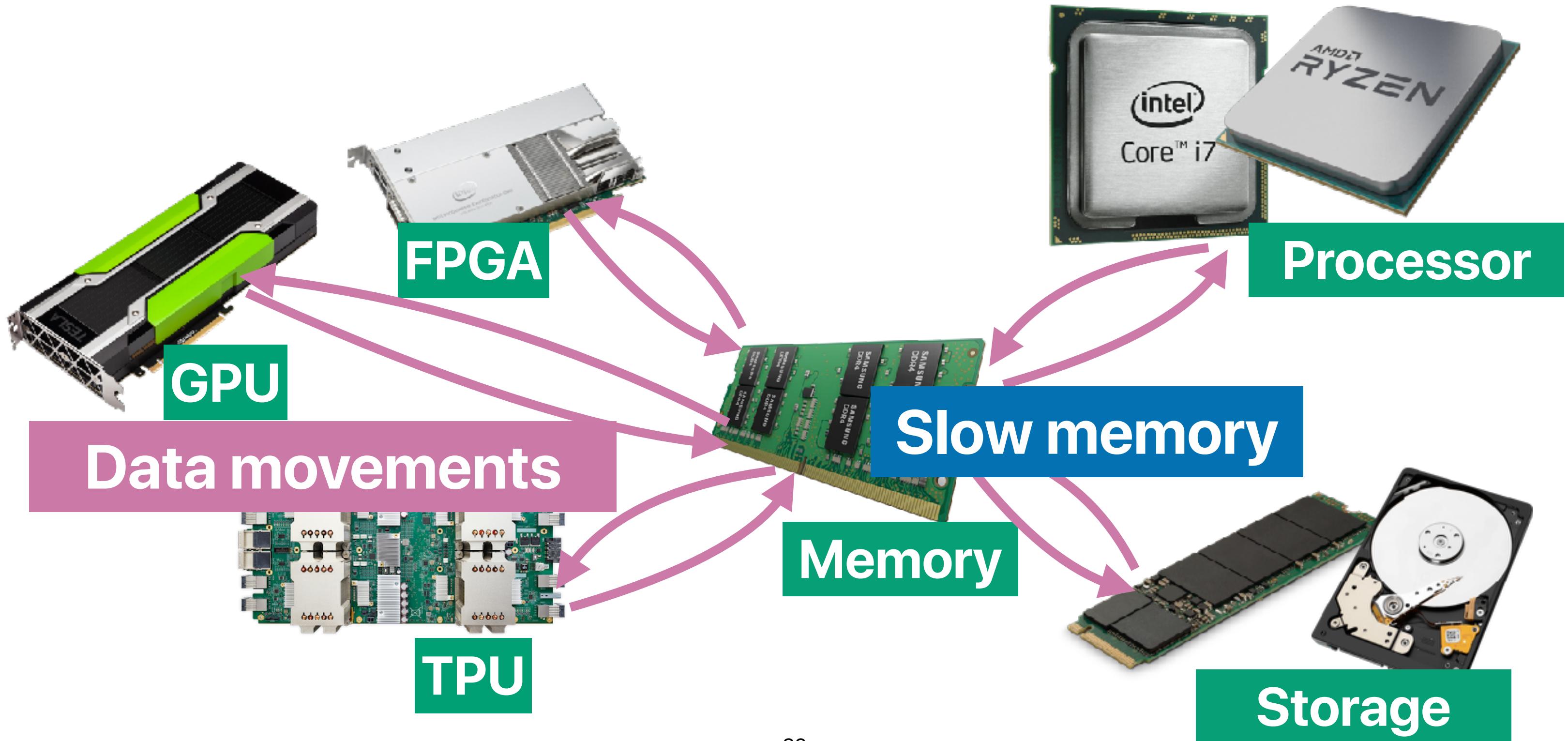
Take-aways: Why CS203?

- Algorithm complexity does not work well on “real” computers
- Processors/memories are essential for modern computer systems but their performance improves slowly in recent decades
 - Moore’s Law continues, but Dennard Scaling discontinues
 - Cannot catch up the demand of applications — programmers need to do something!

Performance gap between Processor/Memory



Heterogeneous Computer Architecture



Take-aways: Why CS203?

- Algorithm complexity does not work well on “real” computers
- Processors/memories are essential for modern computer systems but their performance improves slowly in recent decades
 - Moore’s Law continues, but Dennard Scaling discontinues
 - Cannot catch up the demand of applications — programmers need to do something!
- We have to rethink about programming as computers become more parallel, heterogeneous, application-specific

CS203's vision and missions

Vision

- Be programmers who write efficient programs that save time, power, energy, carbon footprint.

Missions

- Using the knowledge of computer architecture and the general guidelines to identify performance problems and design solutions to tackle the challenges of real-world applications

Heterogeneous Computer Architecture

- ## Performance
- Performance measurement
 - What affects performance
 - Amdahl's Law
 - Metrics

- ## Memory
- Memory hierarchy
 - Hardware optimizations
 - Software optimizations

- ## Processor
- Pipelining
 - OoO Execution
 - Branch predictions
 - Software optimizations

- ## Parallelism
- Parallel hardware
 - Thread-level
 - Data-level
 - Accelerators
 - Software optimizations

TPU

Storage

Tentative Schedule

	Topic	Due Dates
1/7/2025	Introduction	
1/9/2025	Performance Evaluation (I)	Reading Quiz #1
1/14/2025	Performance Evaluation (II)	Reading Quiz #2
1/16/2025	Performance Evaluation (III)	Programming Assignment #1 due
1/21/2025	Memory Hierarchy (1): The Basics	Reading Quiz #3
1/23/2025	Memory Hierarchy (2)	Assignment 1 due
1/28/2025	Memory Hierarchy (3): Optimizing Cache Performance Applications	Reading Quiz #4
1/30/2025	Memory Hierarchy (4): Programmer's optimizations	
2/4/2025	Virtual Memory	Reading Quiz #5
2/6/2025	Virtual Memory (II)	Assignment 2 due
2/11/2025	Basic Processor Design & Branch Prediction	Reading Quiz #6
2/13/2025	Advanced Branch Prediction	Assignment 3 due & Programming Assignment #2
2/18/2025	OOO Scheduling	Reading Quiz #7
2/20/2025	OOO Scheduling	
2/25/2025	OOO Scheduling	Reading Quiz #8
2/27/2025	SMT & Chip Multiprocessors	Assignment 4 due
3/4/2025	Midterm	
3/6/2025	Chip Multiprocessors	
3/11/2025	Dark Silicon	Reading Quiz #9
3/13/2025	TPU, FPGA	Assignment #5
3/20/2025 7p-10p	Final Exam	

Computer Science Graduates Face Worst Job Market in Decades

Computer science graduates are struggling to find jobs as the unemployment rate hits 6.1% in 2025. Check how AI, mass layoffs, and oversupply have created the worst CS job market in decades.



Written by
Kaustubh Saini



Edited by
Jaya Muvania



Reviewed by
Kaivan Dave

Updated on
Aug 18, 2025

Read time
5 Min Read



entrepreneur.com/business-news/college-majors-with-the-lowest-unem...

Entrepreneur

Business News

[Grow Your Biz Watch EntrepreneurTV Now](#)

Canva

All your work in one design

Start designing

Docs Sheets Whiteboard Presentation Social

These Are the College Majors With the Lowest Unemployment Rates — and Philosophy Ranks Higher Than Computer Science

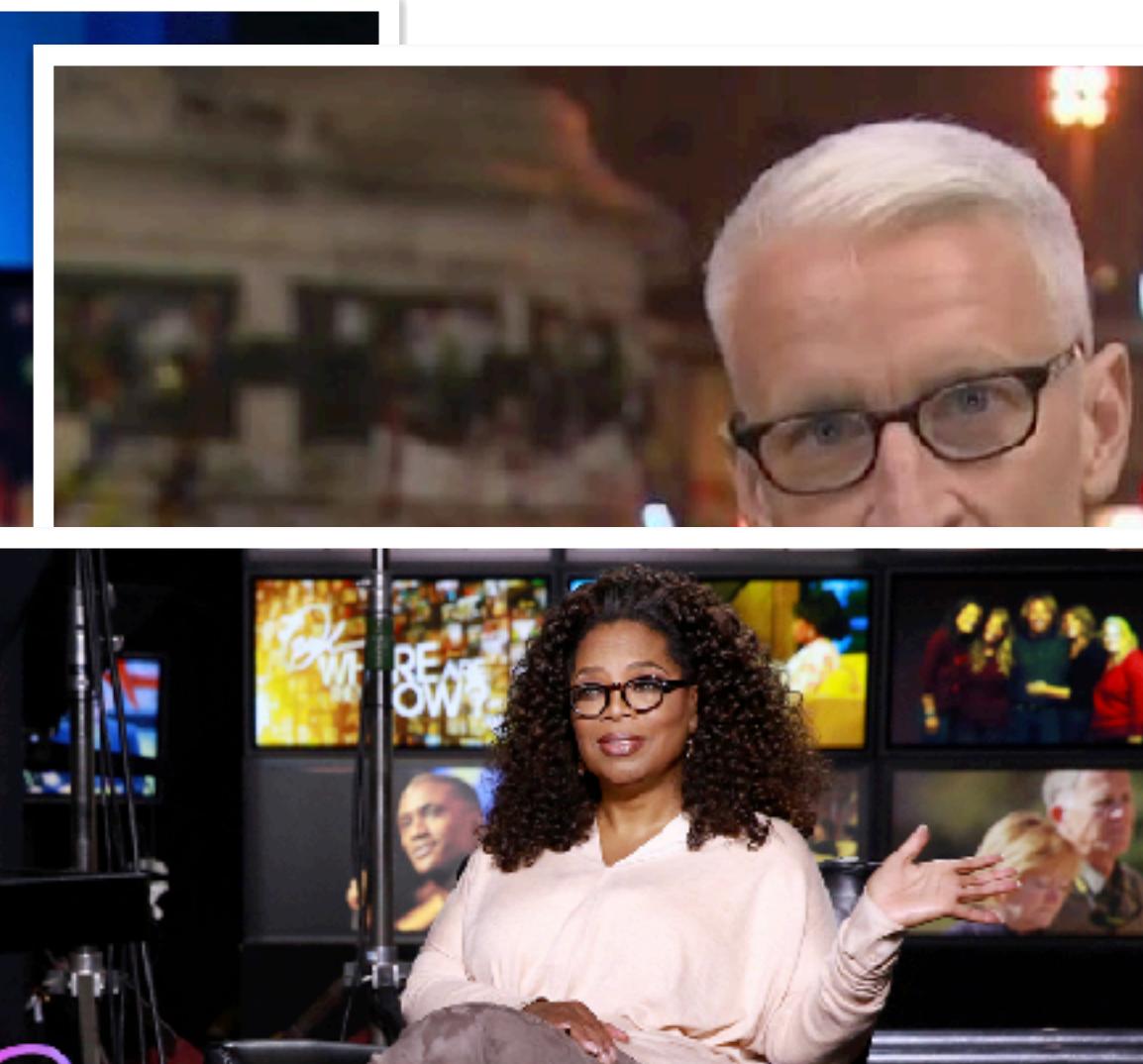
An analysis of employment data from the Federal Reserve Bank of New York shows that some humanities majors rank higher than STEM majors in employment prospects.

What skills will CS203 give you?

- The ability to practice performance programming on large-scale projects
- The ability to analyze/diagnose performance issues in real programs/systems
- The ability to think thoroughly on each concept
- The ability to answer a question properly/precisely with insightful details

Learning eXperience

Most lectures today ...



E SAME JOB **ON CNN TONIGHT**

Paris
4:02 AM

LIVE
CNN
7:02 PM PT

I expect the lecture to be...



Read

Think

Discuss

Learn

Practice

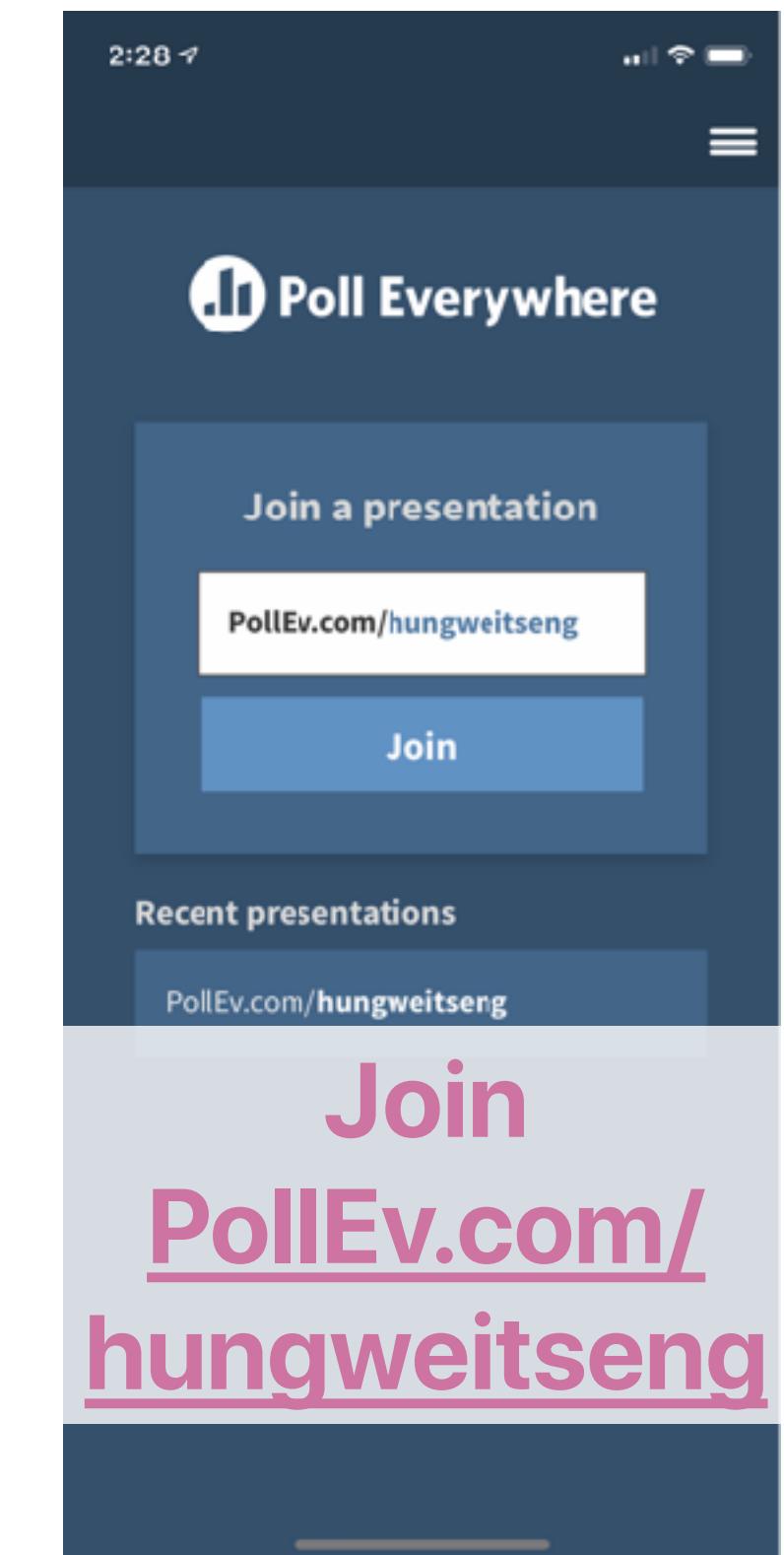
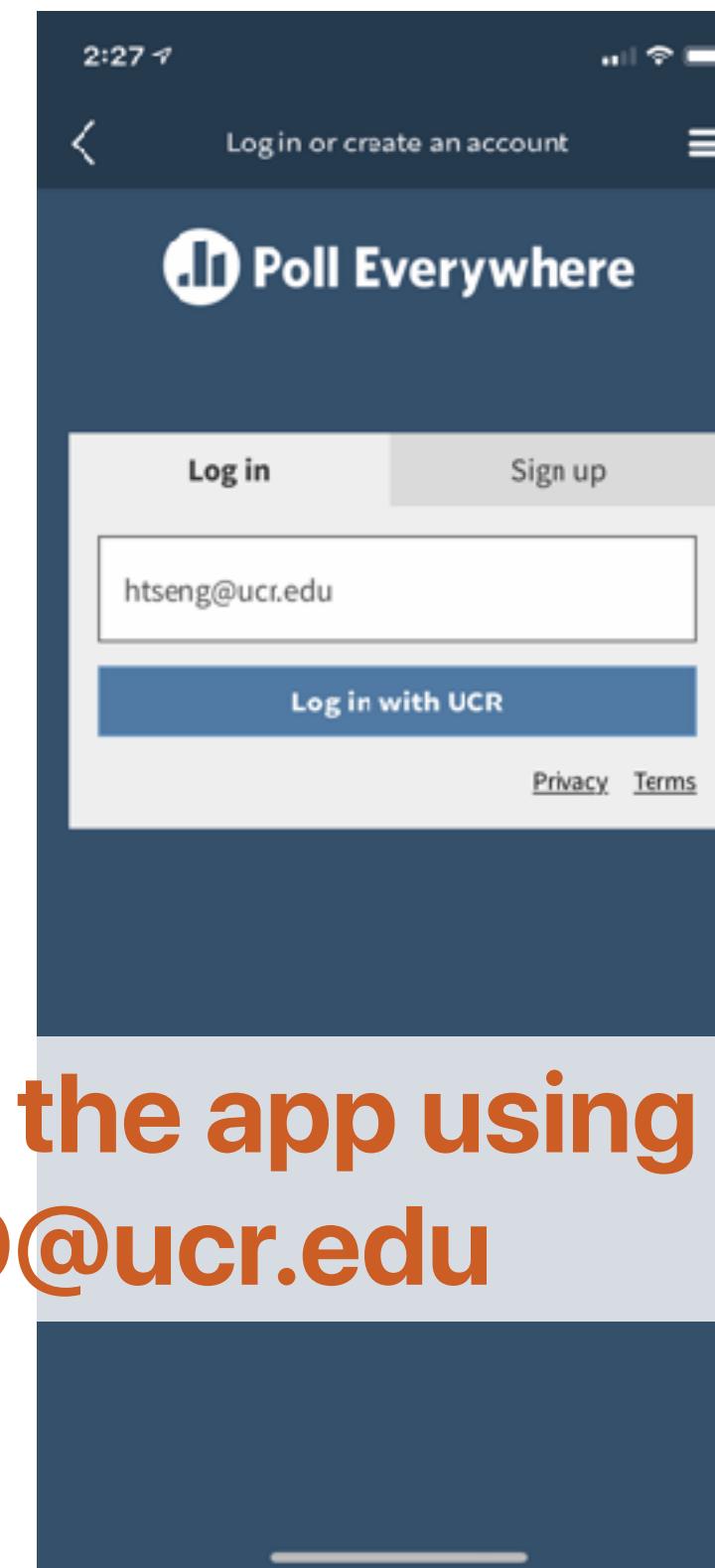
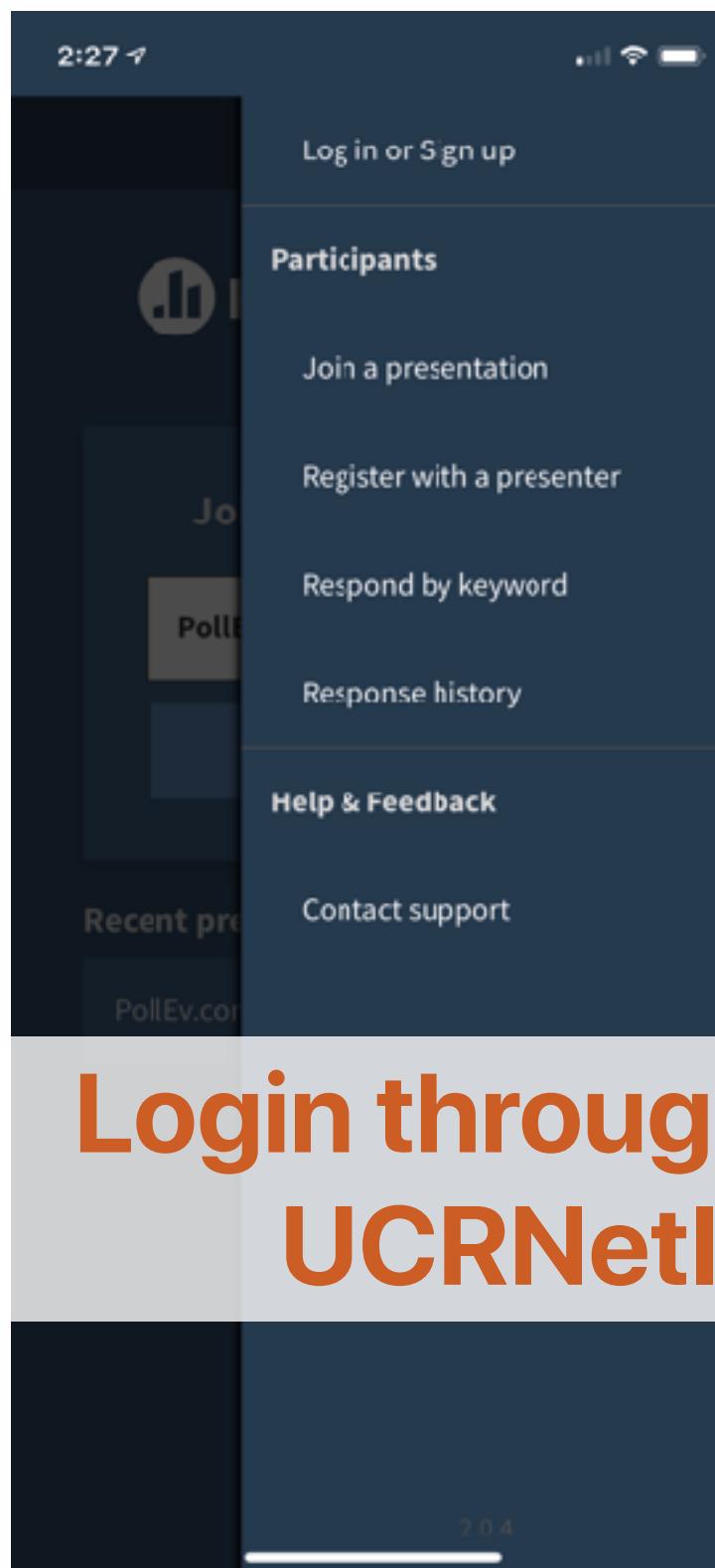
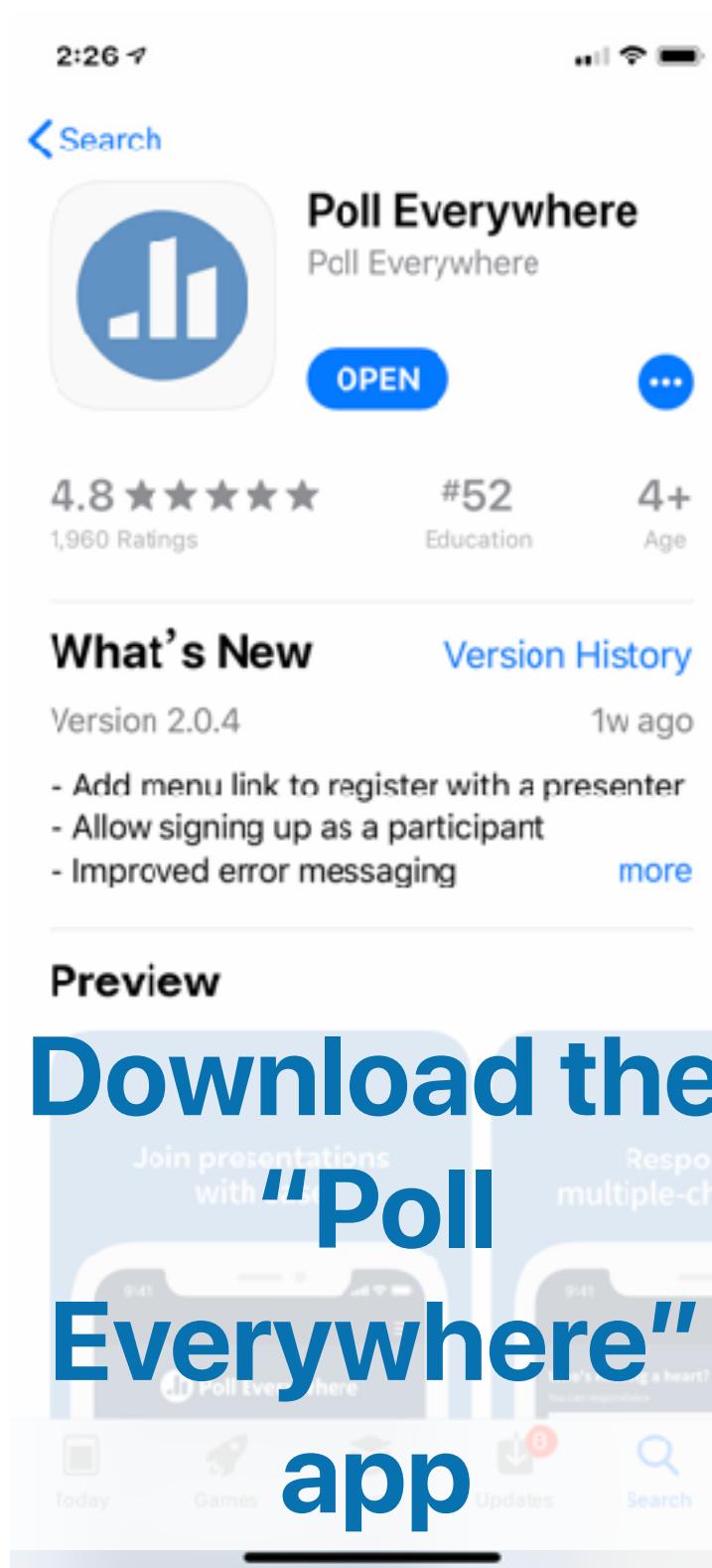
Our method

- Textbook & Papers
- Reading quiz on gradescope— due periodically before entering a new topic
- We will have polls to encourage you think!
 - Let you practice
 - Bring out misconceptions
- We will have polls to encourage you think!
 - Let you practice
 - Bring out misconceptions
- We will learn more after thinking about those questions!
- We will have assignments help review what you learned during lectures!
- We will practice learned concepts into lab experiences!

Before lectures: reading quizzes

- We need to prepare you for peer instruction activities and discussions!
- Reading assignments from
 - Patterson & Hennessy, Computer Architecture: A Quantitative Approach, David Patterson & John Hennessy, Morgan Kaufmann, 6th Edition
 - Assigned research papers/documents
- Reading quizzes — 20% of your grades:
 - On prairielearn (the same interface used for midterm and final)
 - Due before the lecture, usually once a week. Check the schedule on our webpage
 - No time limitation until the deadline
 - No make up reading quizzes — we will drop probably your lowest two

About the time of the Lecture — Setup Poll Everywhere



Login through the app using
UCRNetID@ucr.edu

Peer instruction questions

- The activities to ENGAGE you in exploring your understanding of the material
 - Let you practice
 - Bring out misconceptions
 - Let us LEARN from each other about difficult parts.
- You will be GET CREDIT for your efforts to learn in class
 - By answering questions with **Poll Everywhere**
 - Answer **50%** of the **clicker questions** in class, get a **full-credit assignment**
 - Typically more than 50% of questions are individual thinking questions as individual thinking comes first
 - If you don't feel comfortable to talk with others, you can still get full credits if you made choices on all individual thinking questions

Assignments

- Measure/Analysis/Programing using Jupyterhub/Jupyter notebooks
- Watch this before your first assignment!
https://youtu.be/m7OoY8y_lsk
- Submit through gradescope
 - You have to click the GitHub classroom link to begin with (will post the link of each assignment on the course webpage once released)
 - You should use the escala.org/datahub service to finish your assignment
 - Submit your GitHub repo contains the CS203 assignment through Gradescope
 - An autograder will grade your assignment immediately and you can resubmit as many times as you want before the deadline—the earlier you start, the higher chances you can get full credits
 - There is no regrading after the deadline

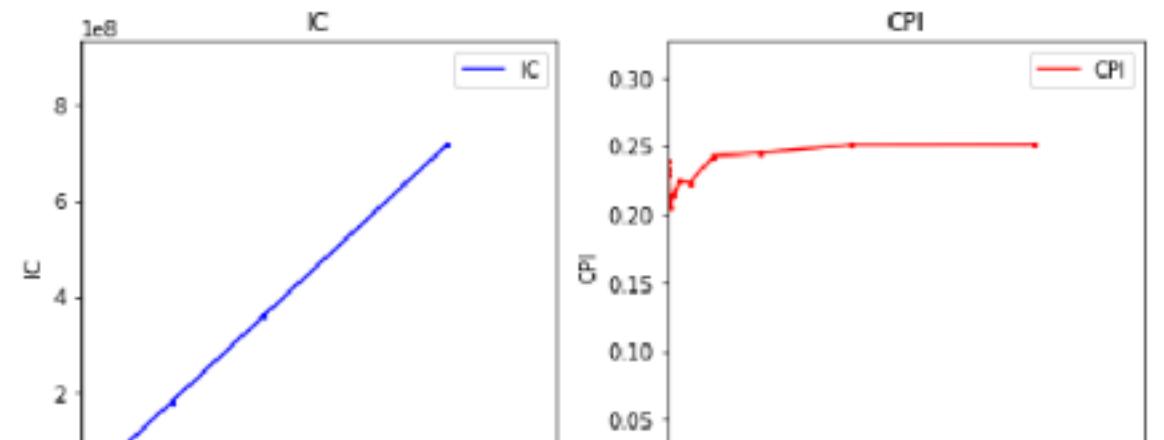
The total execution time is

$$ET = IC \times CPI \times CT = 5602640 \times 0.20$$

Now, let's plot the result and verify your prediction and see how ev

```
[9]: df=render_csv("array_size.csv", columns=columns, average=True)
plotPE(df=df, lines=True, what=[('size', "IC"), ("size", "CPI")])
display_mono(df)
```

	size	ET	IC	CPI	M
0	10000	0.000042	703477.840000	0.238585	3995.2118
1	20000	0.000083	1404491.760000	0.233374	3970.8904
2	40000	0.000160	2803566.080000	0.228001	3983.0703
3	80000	0.000287	5604217.320000	0.204585	3988.8586
4	160000	0.000600	11204847.920000	0.213550	3988.0081
5	320000	0.001257	22406745.080000	0.223729	3989.3769
6	640000	0.002506	44812777.640000	0.222840	3985.2323
7	1280000	0.005459	89617166.120000	0.242636	3983.1877
8	2560000	0.011036	179228929.520000	0.244669	3973.7334
9	5120000	0.022575	358452827.360000	0.250542	3978.1896
10	10240000	0.045183	716890078.920000	0.250808	3979.4289



Computer-based quizzes/examines

- Prairielearn as the testing platform
- Midterm and final
 - Schedule your examines at <https://us.prairietest.com/pt>
 - Taking place in UCR testing center with dedicated proctor
 - Bring your ID with you
 - You should book a practice examine to get familiar with the environment
 - Both midterm & final forbid the use of
 - Notes
 - Cheatsheets
 - Textbooks
 - Calculators (the computer provides)
 - Any outside materials

The screenshot shows the PrairieLearn platform interface. At the top, there's a navigation bar with 'PrairieLearn' logo, 'Home', 'CSE 142', and 'CSE142_2025_Summer'. Below the navigation is a toolbar with icons for 'Files' (selected), 'Preview', 'Settings', 'Statistics', and 'Issues'. On the left, there's a sidebar with various icons. The main content area displays a quiz question: "What are some academic integrity violations?" with a "View..." button. The question asks: "Which of the following are academic integrity violations in this course?". There are seven options, each with an unchecked checkbox:

- (a) Using Co-pilot
- (b) Using AI tools on the homeworks.
- (c) Submitting in-class responses for another student.
- (d) Collaborating with another student on a Quiz
- (e) Discussing how to approach a homework problem or PA with another student.
- (f) Submitting in-class responses when you are not in class.
- (g) Collaborating on the homeworks.

A note at the bottom says "Select all possible options that apply." with a help icon.

Logistics

Grading Breakdown

	Percentage	Notes
Reading Quizzes	20%	Drop the lowest 2
Assignments	30%	Drop the lowest 1
Participation	As part of assignments	Count as one assignment. Get full credit if you show up on 50% of PI questions If you don't show up, please do well on every assignment!
Midterm	20%	UCR Testing Center — please schedule your slots between 10/27/2025–10/31/2025 with UCR Testing Center as soon as possible
Final	30%	UCR Testing Center — please schedule your slots between 12/5/2025—12/9/2025 with UCR Testing Center as soon as possible

- Review the course website for policies
- Check your grades https://www.escalab.org/my_grades/

Instructor — Hung-Wei Tseng

- Associate Professor @ UC Riverside, 05/2019—
- Website: <https://intra.engr.ucr.edu/~htseng/>
- E-mail: cs203 @ escal.org
- Visiting Researcher @ Google, 01/2023—03/2023
 - Working for TensorFlow Lite
- PhD in **Computer Science**, University of California, San Diego, 2014
- Research Interests
 - General-purpose computing on AI/ML/NN/RayTracing accelerators
 - Or anything else fun — we have an OpenUVR project recently
- Fun fact: Hung-Wei was once considering a career path as a singer but went back to academia due to the unsuccessful trial



Teaching Assistant

- Hongrui Zhang
- Office hours: TuW 4p-5p @ WCH 110 & Thu 4p-5p
@ zoom
- E-mail: cs203 @ escalab.org
- Fun fact:

Subscribe to our google calendar!

Oct 2024



MON	TUE	WED	THU
30	1	2	3
	CS203 Lecture 11am – 12:20pm		CS203 Lecture 11am – 12:20pm
		Hung-Wei's Office Hour 2 – 4pm	
	TA office hour 4 – 5pm		TA office hour 4 – 6pm

The website

- <https://www.escalab.org/classes/cs203-2025fa/>
- Calendar
- Schedule
- Slides
 - Preview — for the ease of note taking
 - Release — the actual slides



escalab.org/classes/cs203-2025wi/ Extreme Scale Computer Architecture Lab... Customize 22 New Edit Page Purge SG Cache

CS203: Advanced Computer Architecture (2025)

Lecture: TuTh 2p – 3:20p
Where: Watkins Hall | Room 1101

Schedule and Slides Assignments Logistics

Instructor

Hung-Wei Tseng
email: cs203 @ escalab.org
Office Hours: TBA

Teaching Assistant

TBA
email: cs203 @ escalab.org
Office Hours: TBA

Other important links

- Assignment & Reading quizzes submission through
- Discussion Forum on Discord: Please find the link in your course agreement on Gradescope.
- Lecture recordings: [Youtube Channel](#)
- Calendar: [Google Calendar](#)

今天 < > 2025年1月 月

週日	周一	周二	周三	周四	周五	周六
29	30	31	1月 1日	2	3	4
5	6	7	8	9	10	11

● 上午11點 HH214
△ 下午1點 HH214

Summary of course resources

- Lectures:
 - In-person @ WCH 143
 - Repository on **Youtube**: <https://www.youtube.com/profusagi>
- Schedule, slides, grades on **course webpage**:
<https://www.escalab.org/classes/cs203-2025fa/>
- Discussion on **discord** — link in **gradescope's course agreement**
- Reading quizzes and assignments through **gradescope**:
<https://www.gradescope.com/courses/928407>
- Working environment on <https://escalab.org/databut>
- Office Hours & Locations
 - Hung-Wei Tseng — W 2p-4p WCH 406



Academic Honesty

- Don't cheat.
 - Cheating on a test will get you an F in the class and no option to drop, and a visit with your college dean.
 - Cheating on homework means you don't have to turn them in any more, but you don't get points either. You will also take at least 25% penalty on the exam grades.
- Copying solutions of the internet or a solutions manual is cheating
 - They are incorrect sometimes
- Review the UCR student handbook
- When in doubt, ask.

Frequently asked questions (FAQs)

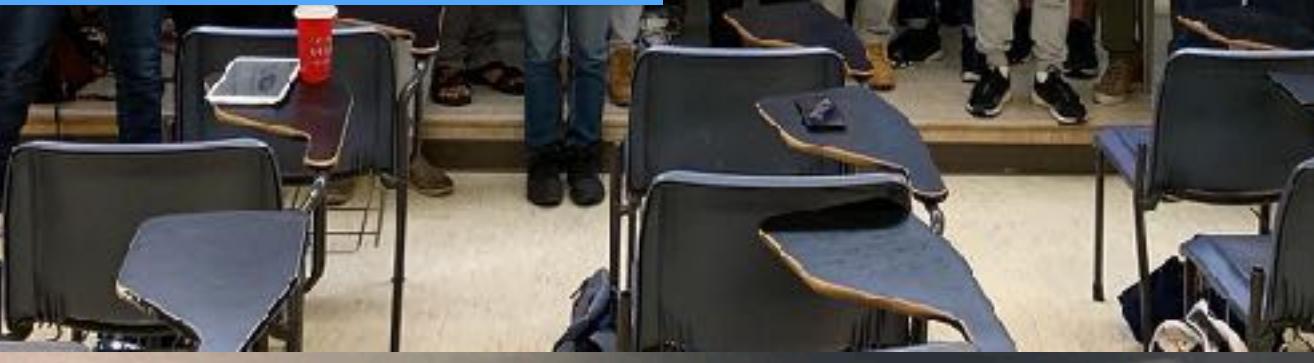
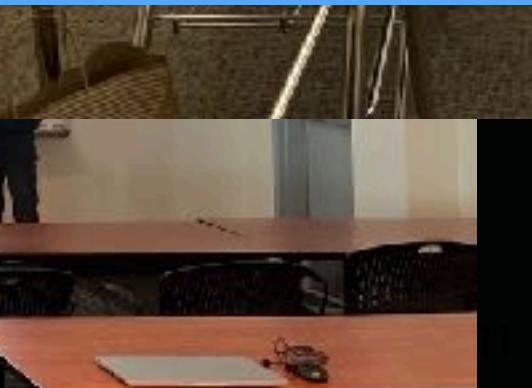
- Do I need to attend every lecture?
 - Attending the lecture is never “required”, but strongly encouraged
 - In-person students have better learning outcome even though given the same difficulty of midterm/final examines based on the experience last year
 - We encourage you to participate in in-class learning activities (peer instructions) through giving you a “full-credit assignment” if you can answer 50% of the PI questions.
 - If you have difficulties coming to the 50% of class, you can still earn all credits if you ace all assignments.
 - Please **do not** email the staff regarding individual instances of no show
- Do we podcast the class online?
 - We record all lectures and post them on the same day as the lecture on YouTube
<https://www.youtube.com/profusagi>
- Are examines in-person?
 - Yes — for fairness and logistics regarding proctoring

FAQs (cont.)

- When will I know my assignment grade?
 - A few minutes after your submission — everything is automatically graded
- How many times can I submit my assignment?
 - As many times as you want before the deadline
 - Start early and get feedback early — more chances to ace the assignments
- Do we have an eLearn/canvas page?
 - No — eLearn/canvas is broken in many aspects and the grade book is never accurate
 - Please refer to the class webpage for the most comprehensive links

FAQs (cont.)

- I cannot login escalab.org/databhub
 - Please email cs203@escalab.org as soon as possible (M-F 9a-5p)
- I cannot login Gradescope
 - You need to officially enrolled to have access to the Gradescope of the class — I cannot help you unfortunately
- Are textbooks required?
 - Yes
- What if I need help during the weekend?
 - Unfortunately, we don't guarantee we can answer you questions either through piazza or e-mail during the weekend — you should plan your schedule carefully



2025 Fall

Announcements



- Check our website for all resources
- Login to datahub, gradescope, discord, prairielearn
 - Let us know if you have any issue — cs203@escalab.org
- Course agreement/Reading quiz on prairielearn due **next Tuesday before the lecture**
 - Unlimited time before Tuesday's lecture — one trial only, no make-ups
- Please make sure your gradescope account is associated with your UCRNetID@ucr.edu — you cannot receive any credit if there is a mismatch
- Programming assignment #1 are released — working on datahub!
 - Watch this video before you start! https://youtu.be/m7OoY8y_lsk

Computer Science & Engineering

203

つづく

