

**First Day of
CSE142: Computer Architecture: Software
Perspective (2025 Summer Session II)**

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: htseng @ ucsd.edu
- Visiting Researcher @ Google, 01/2023—03/2023
 - Working for TensorFlow Lite
- PhD in **Computer Science**, University of California, San Diego, 2014
- Research Interests
 - Using ray-tracing accelerators for multiple domains of accelerators
 - General-purpose computing on AI/ML/NN accelerators
 - Intelligent storage devices & near-data processing
 - Or anything else fun
- Fun fact: Hung-Wei was once considering a career path as a singer but went back to academia due to the unsuccessful trial

A collage of three images. The top right image shows a man with glasses sitting at a desk in an office. The middle image is a purple graphic from a personality test showing a cartoon character and the text "Your personality type is: Architect INTJ-A". The bottom right image shows a man singing into a microphone on stage.

Your personality type is:
Architect
INTJ-A

Thought constitutes the greatness of man. Man is a reed, the feeblest thing in nature, but he is a thinking reed.

Blaise Pascal



Now, make sure you login to Poll Everywhere (through the App or the website) with your UCSD email



<https://www.pollev.com/hungweitseng> close in 1:30

What's your favorite topic in CS?





Why are you taking CSE142?

Why're you taking computer architecture?

20

The word cloud displays the following words and their approximate sizes:

- interested (large, blue)
- graduation (large, yellow)
- foundation (medium, brown)
- understanding (large, dark red)
- hardware (large, green)
- deeper (medium, blue)
- friends (medium, purple)
- challenge (medium, teal)
- ml (small, maroon)
- Core (large, blue)
- needed (medium, green)
- class (large, purple)
- tiny (medium, purple)
- modeling (medium, green)
- architecture (large, green)
- course (medium, brown)
- applications (medium, green)
- required (large, teal)
- professor (medium, purple)
- trying (medium, green)
- cool (medium, green)
- acceleration (large, purple)
- knowledge (medium, green)
- seem (medium, purple)
- know (medium, green)
- yes (medium, purple)
- performance (large, blue)

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

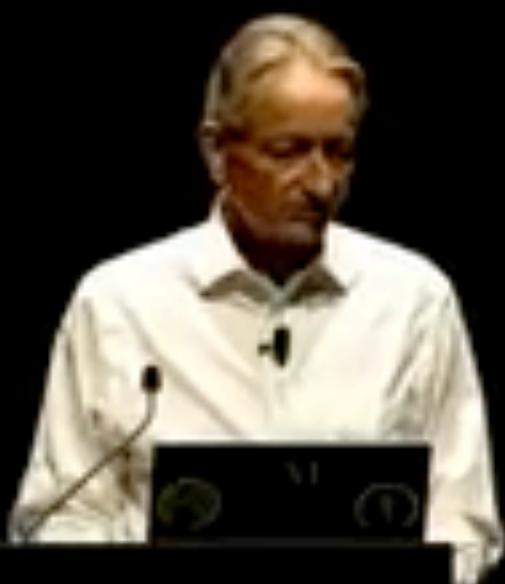
How important is computer architecture in computer science?

+ 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)



COMPUTER ORGANIZATION AND DESIGN MIPS EDITION

THE HARDWARE/SOFTWARE INTERFACE

SIXTH EDITION



DAVID A. PATTERSON & JOHN L. HENNESSY

Computer Architecture

Enables

Deep Learning

Computer Architects — Hidden Figures

BASED ON THE UNTOLD TRUE STORY

HIDDEN FIGURES



**Why should I care about
“Computer Architecture”**

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*

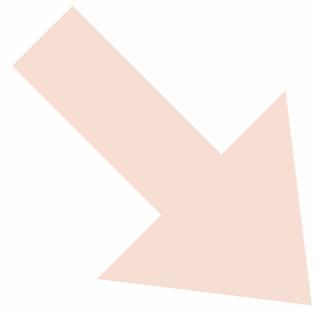
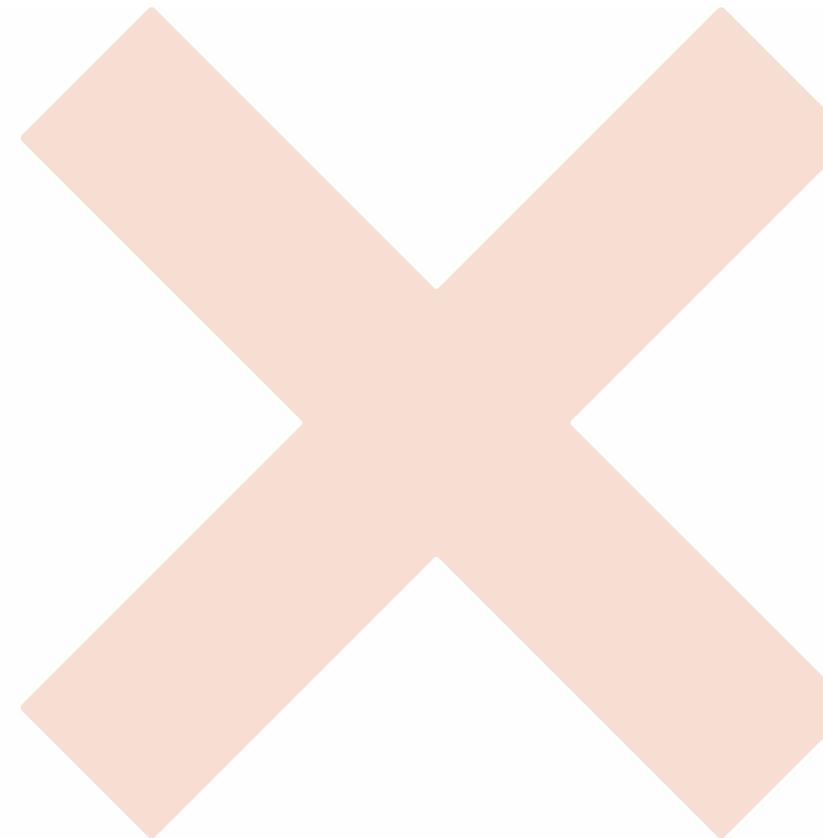
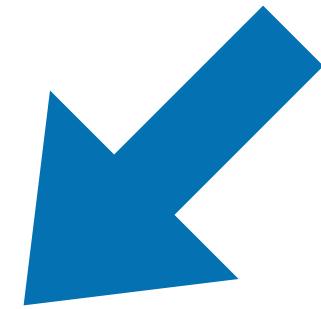
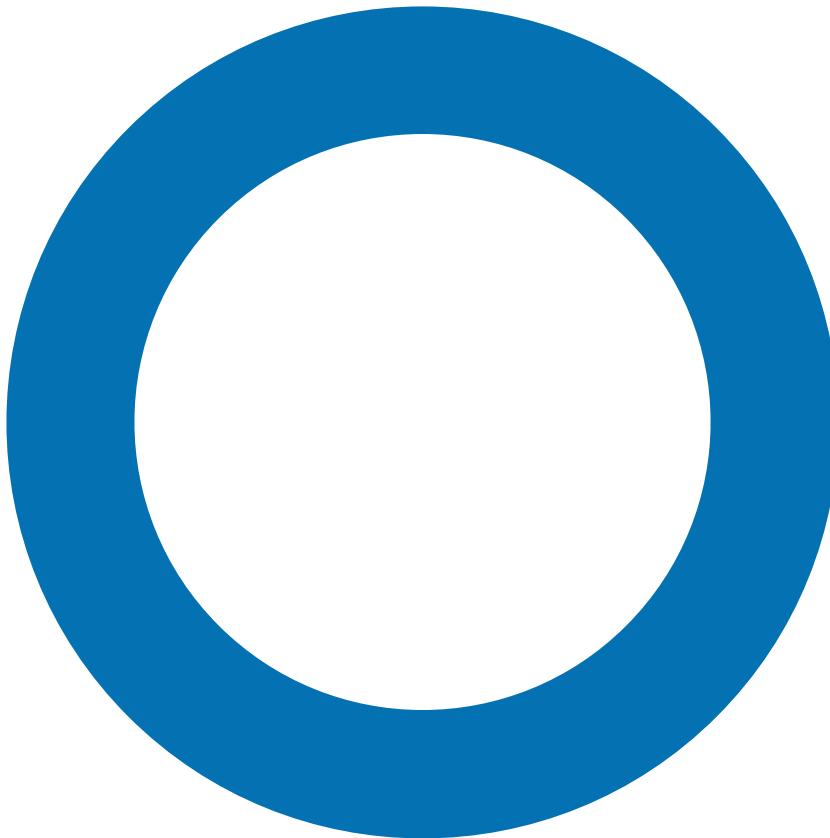
5 : the manner in which the components of a computer or computer system are organized and integrated

// *different program architectures*

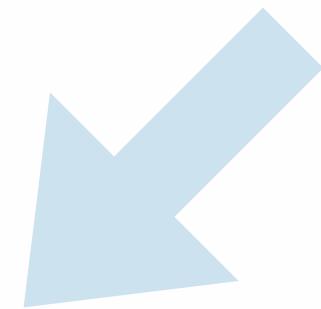
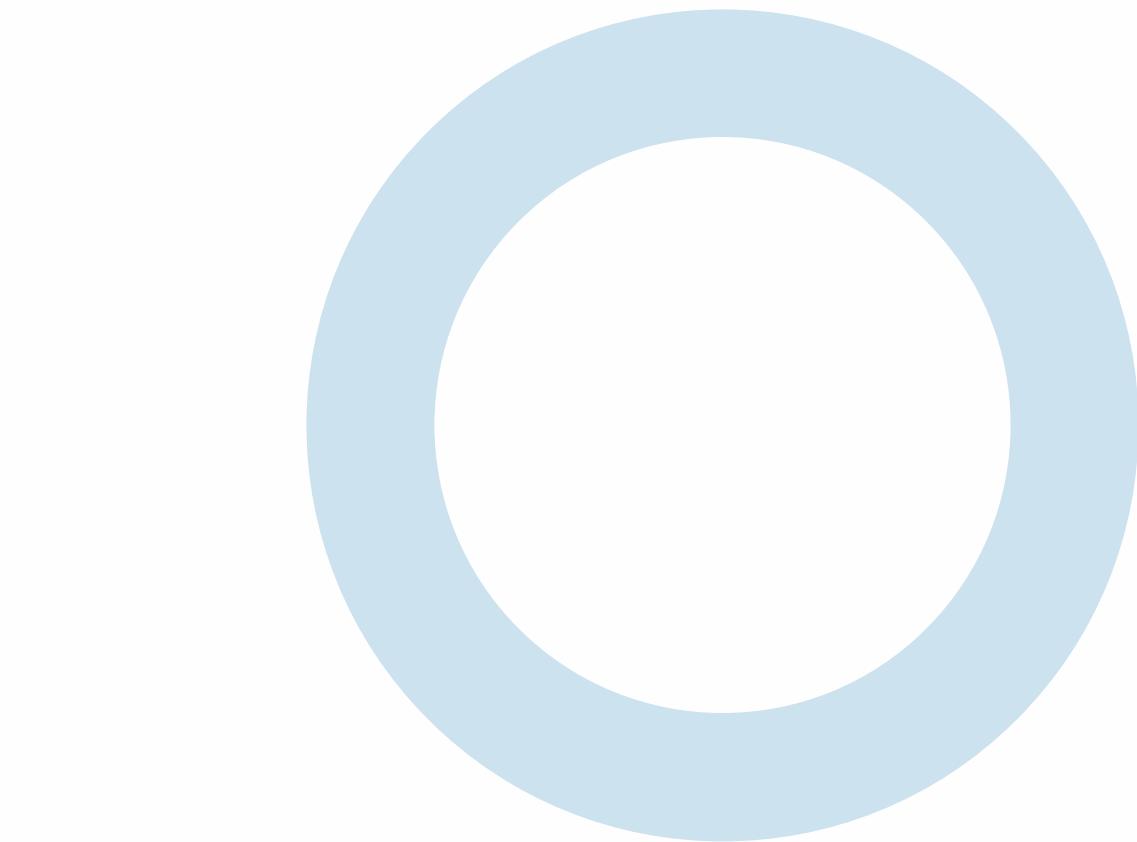
**The manner in which the components
of a computer or computer system are
organized and integrated**

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

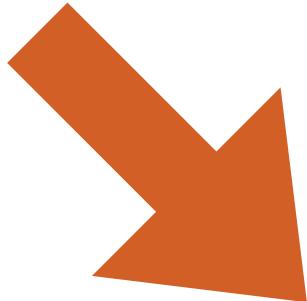
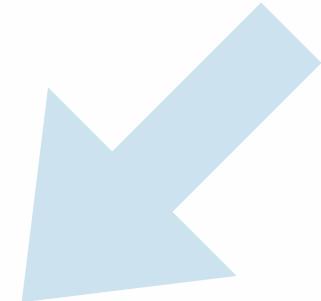
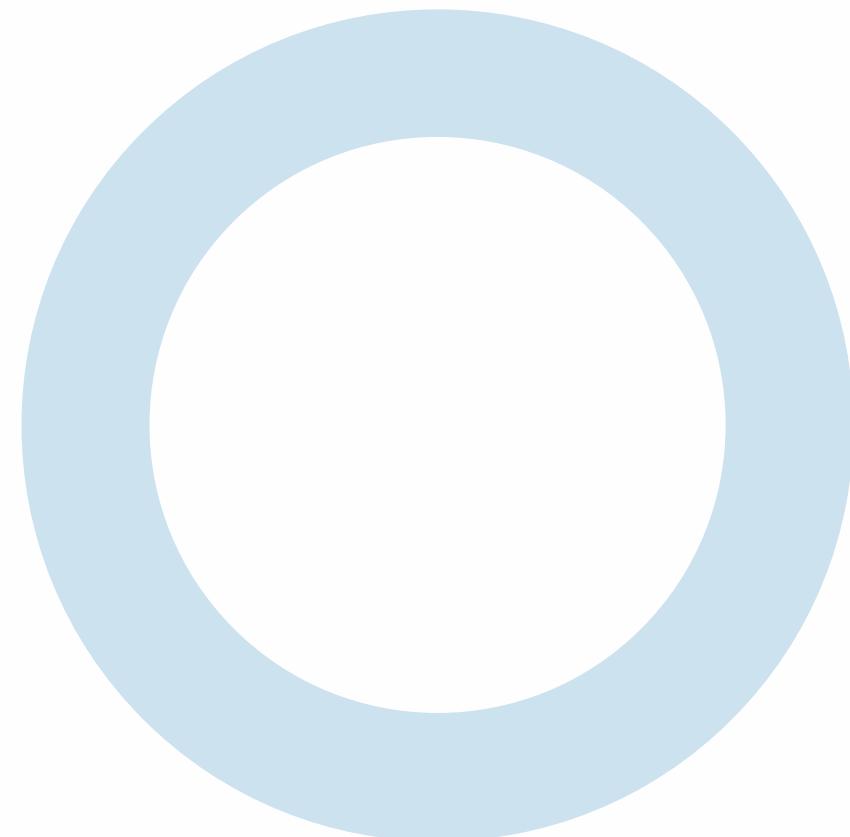
Processors and memories are essential for most modern general-purpose computers



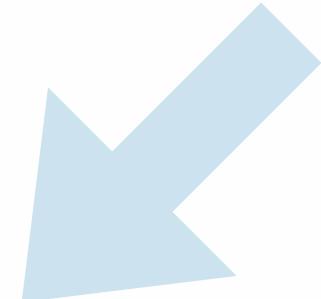
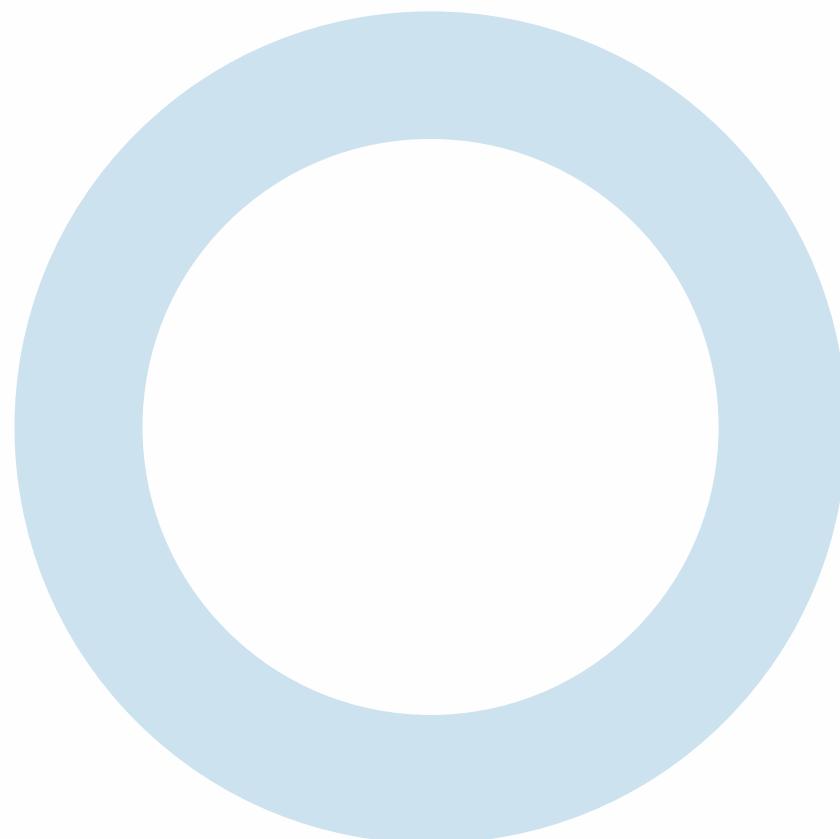
**On the same machine, programs with fewer lines
of code will have shorter execution time**



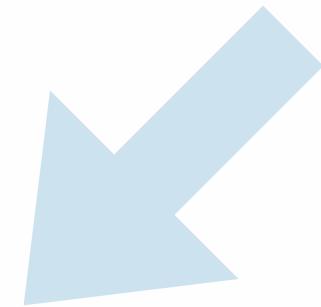
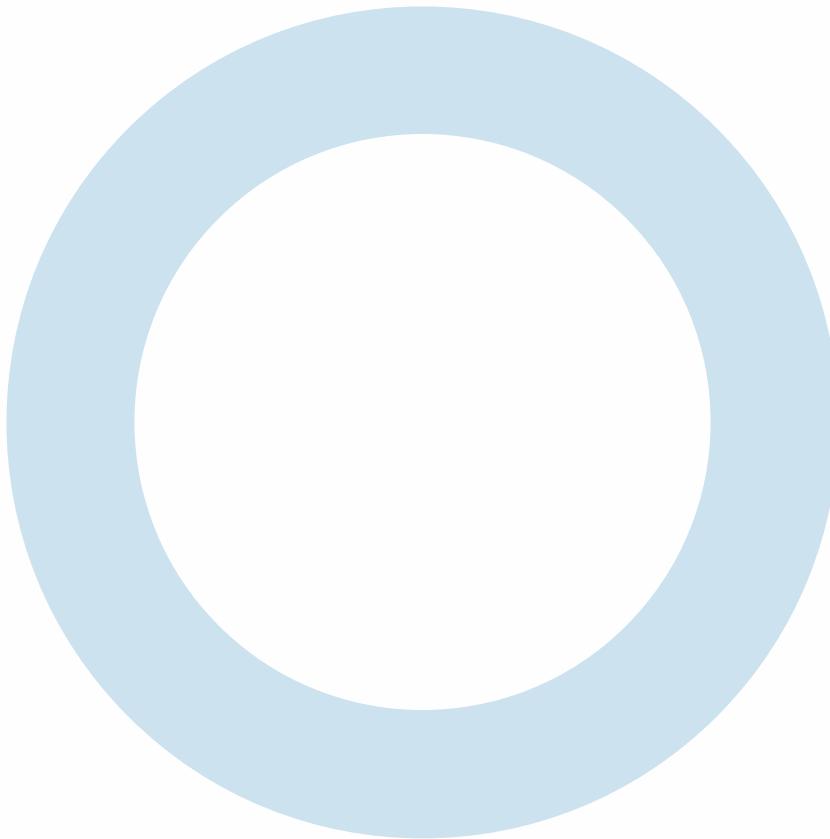
On the same machine, an algorithm with lower computational complexities will have shorter execution time when solving the same problem with the same input



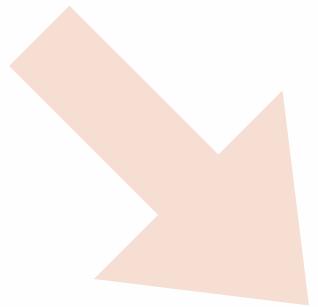
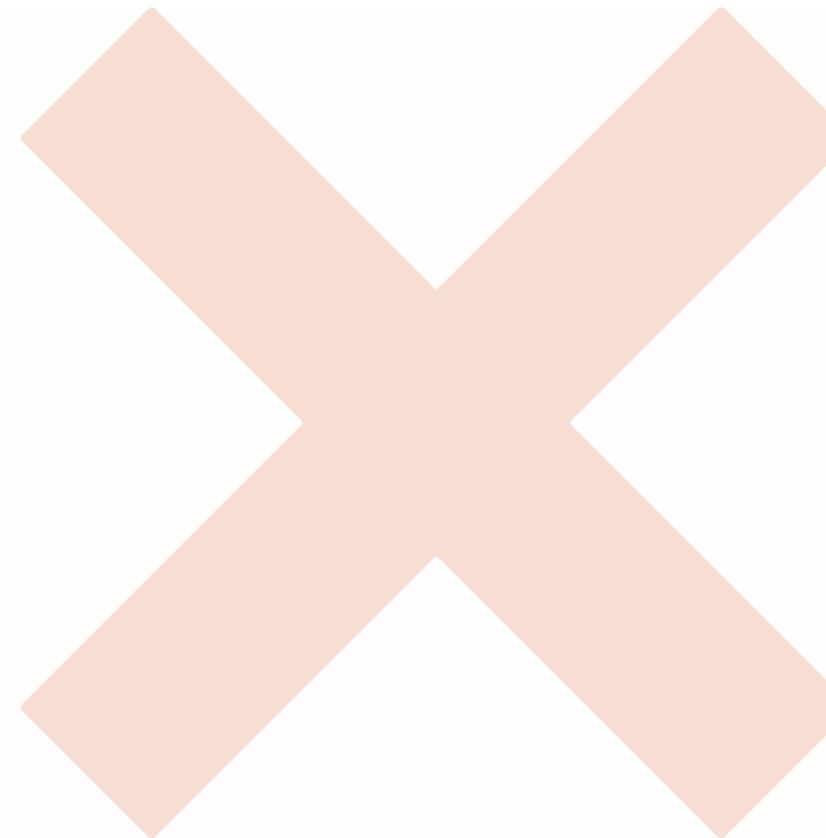
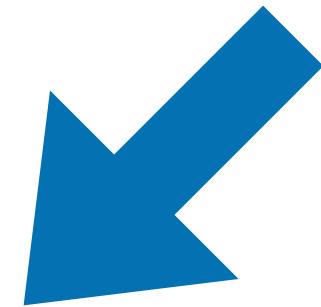
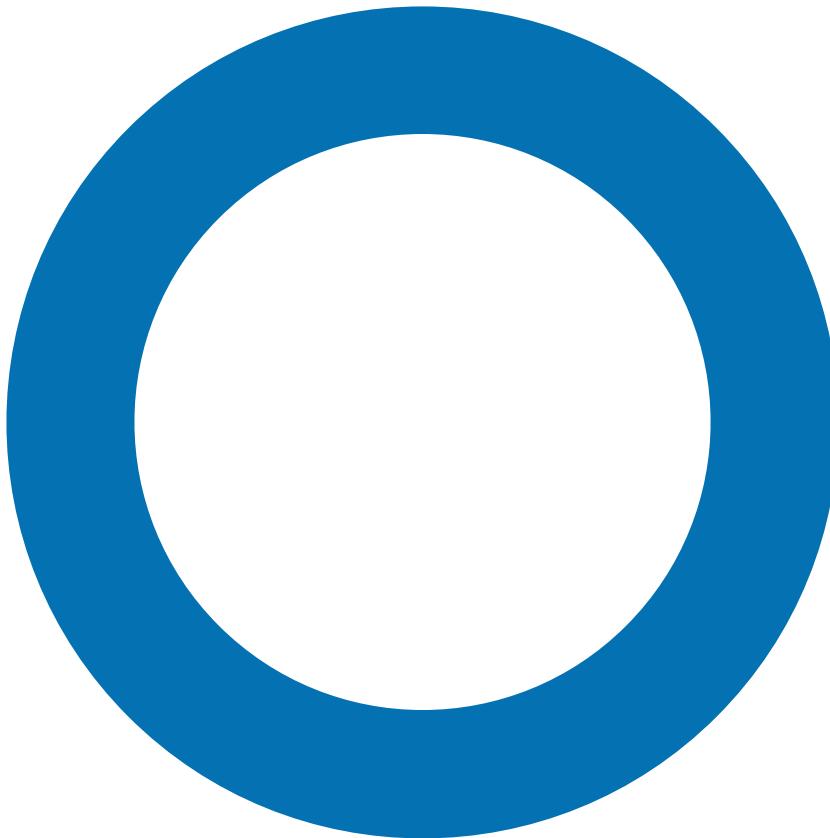
On the same machine, algorithm implementations with the same computational complexities will similar execution time when solving the same problem with the same input



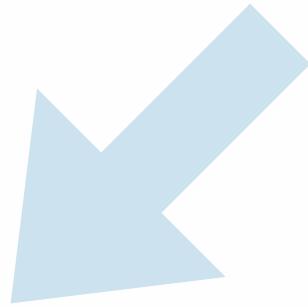
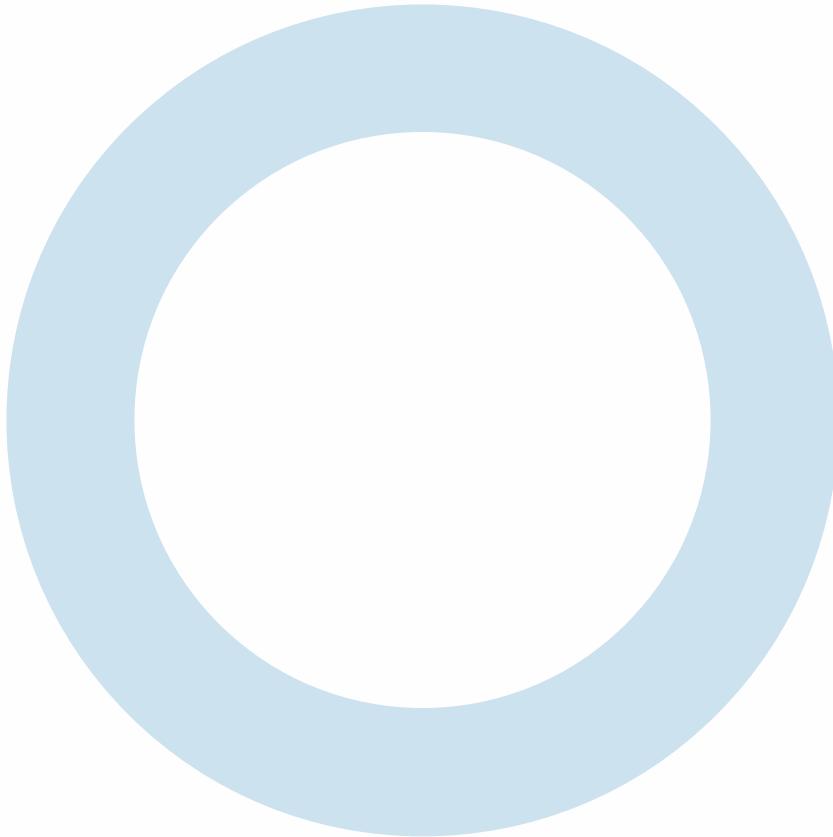
Leveraging more “bit-wise” operations in C code will make the program significantly faster



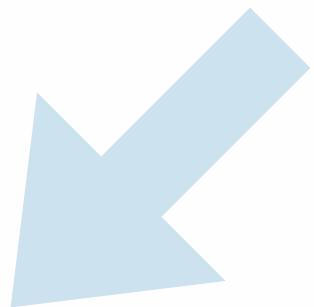
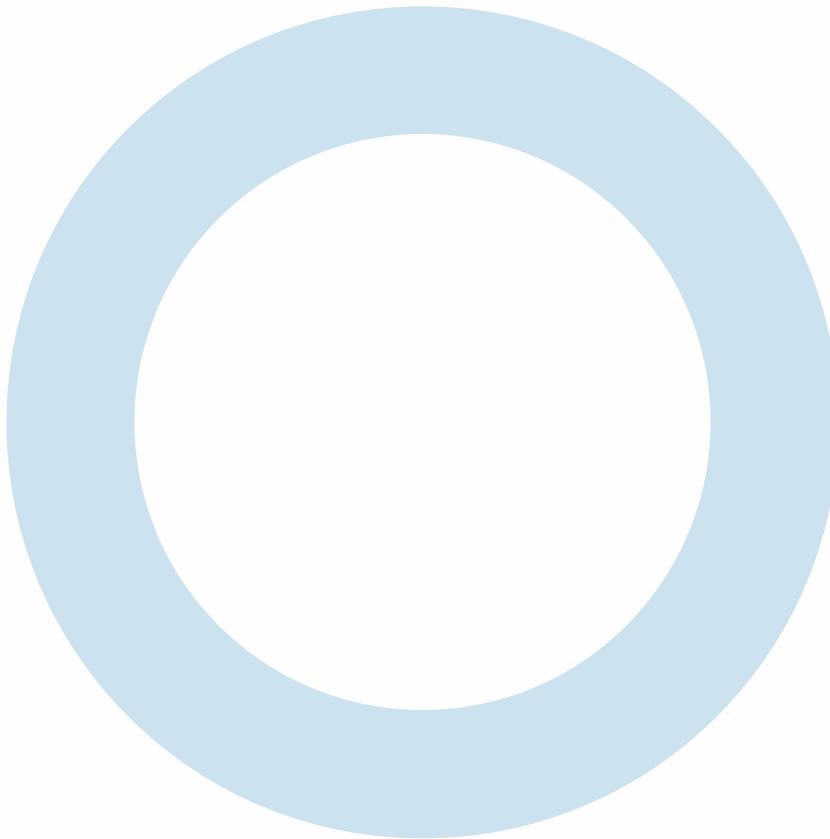
Algorithm complexity is less important if we have rich parallel processing capabilities



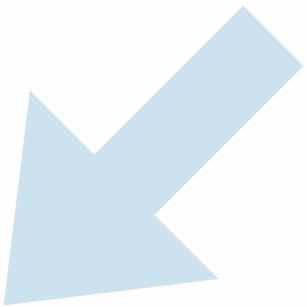
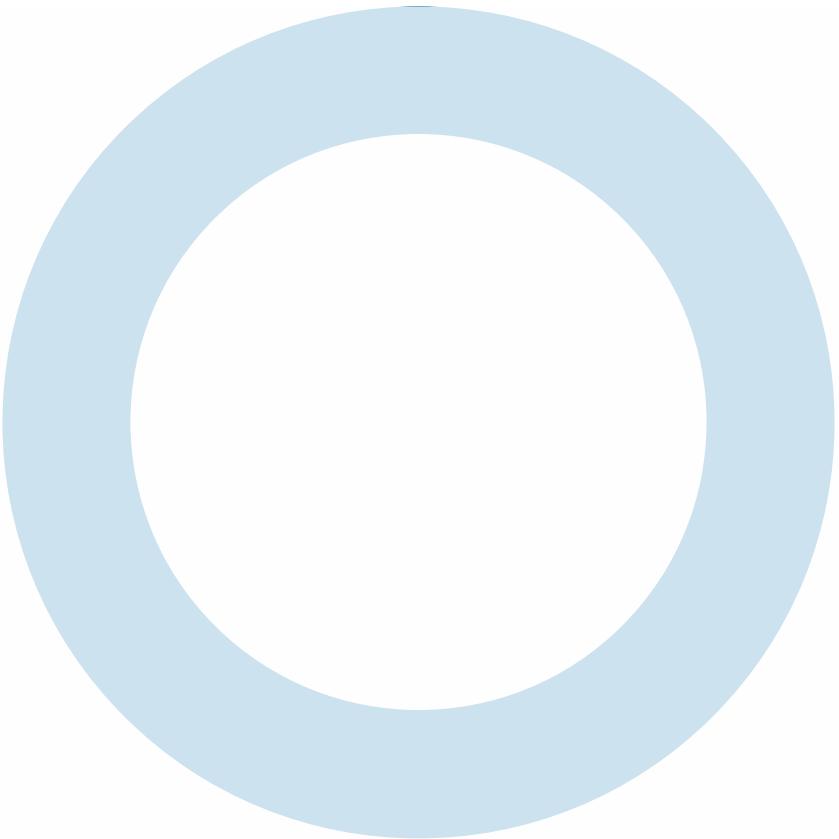
Moore's Law is currently discontinued



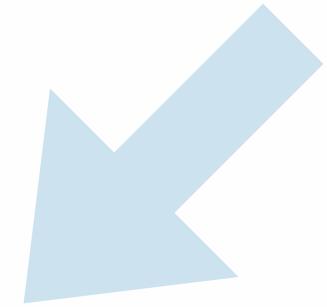
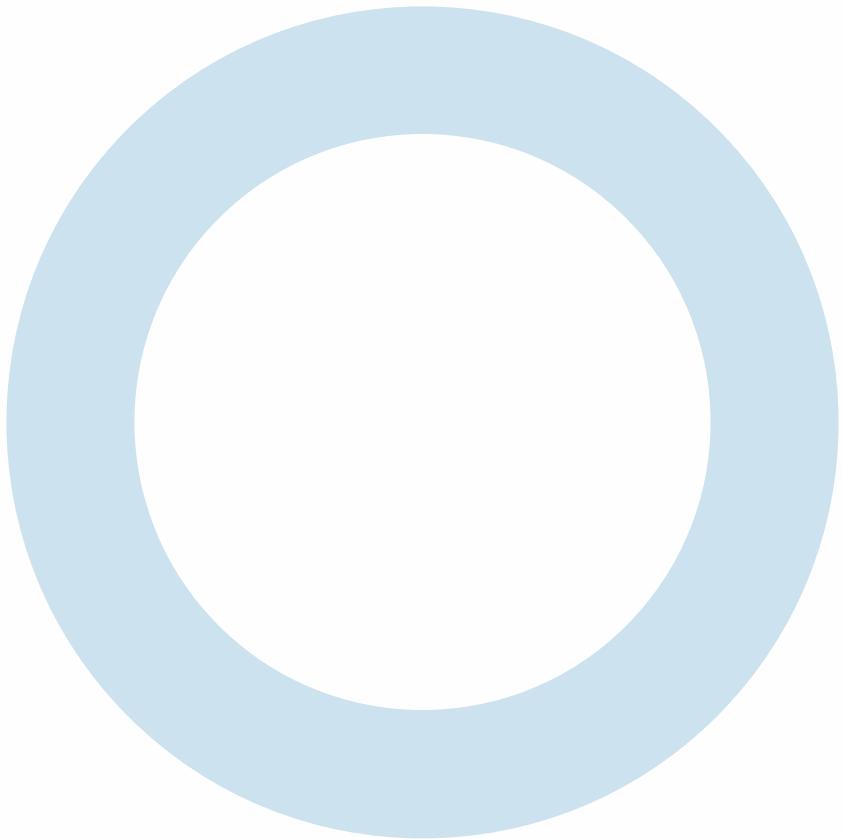
**The smaller size of a transistor,
the smaller power consumption of it**



The most recent GPUs can match the demand of efficient AI/ML training



**With GPUs/TPUs taking more roles,
CPU performance does not matter that much in applications**



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

main:
.LFB0:
endbr64
pushq %rbp
movq %rsp, %rbp
movl \$0, -8(%rbp)
movl \$0, -4(%rbp)
movl \$0, -8(%rbp)
jmp .L2

.L3:
movl -8(%rbp), %eax
cltq
leaq 0(%rax,4), %rdx
leaq A(%rip), %rax

Compiler

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

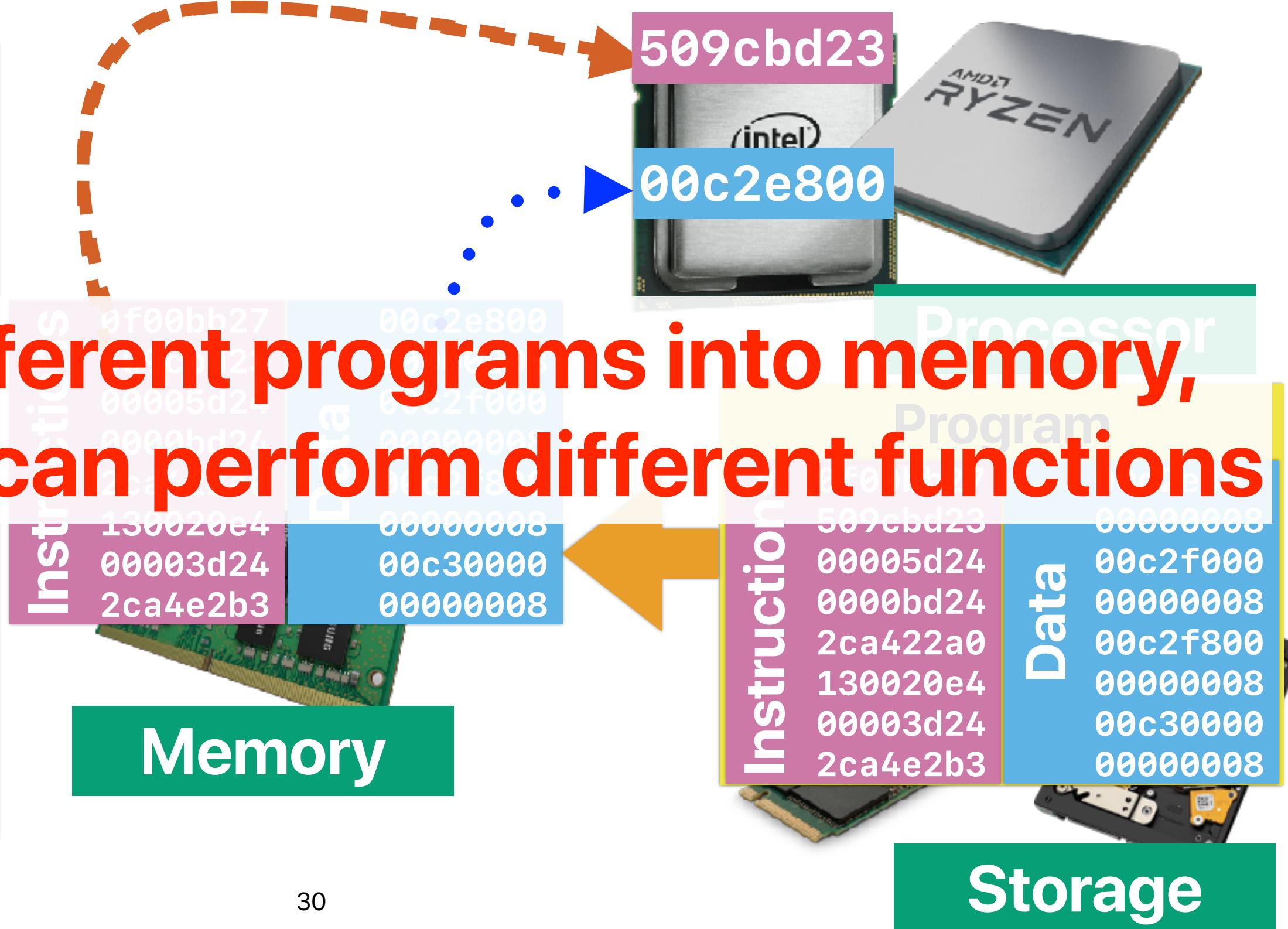
Instructions

Big Picture: the Von Neumann Architecture

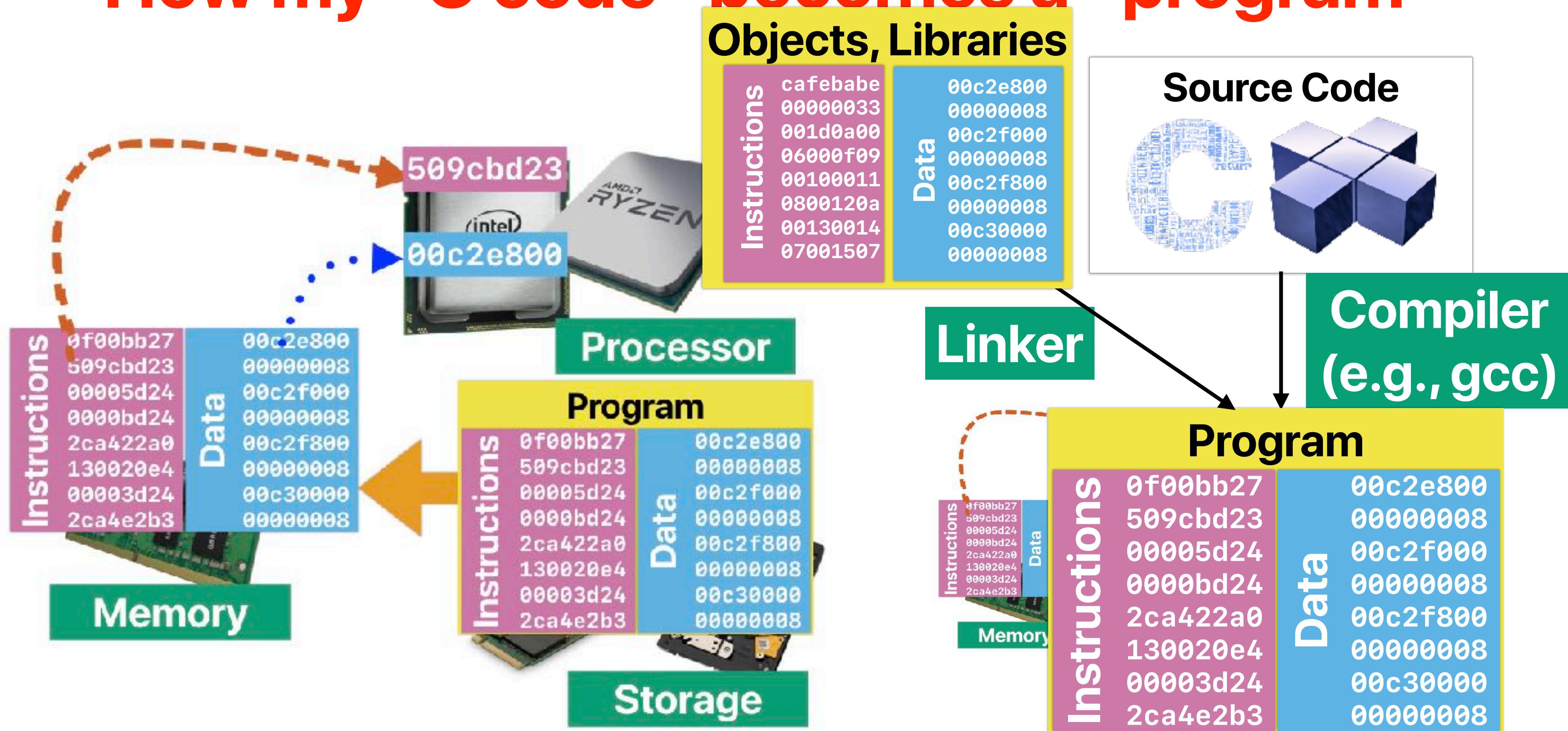
von Neumann Architecture



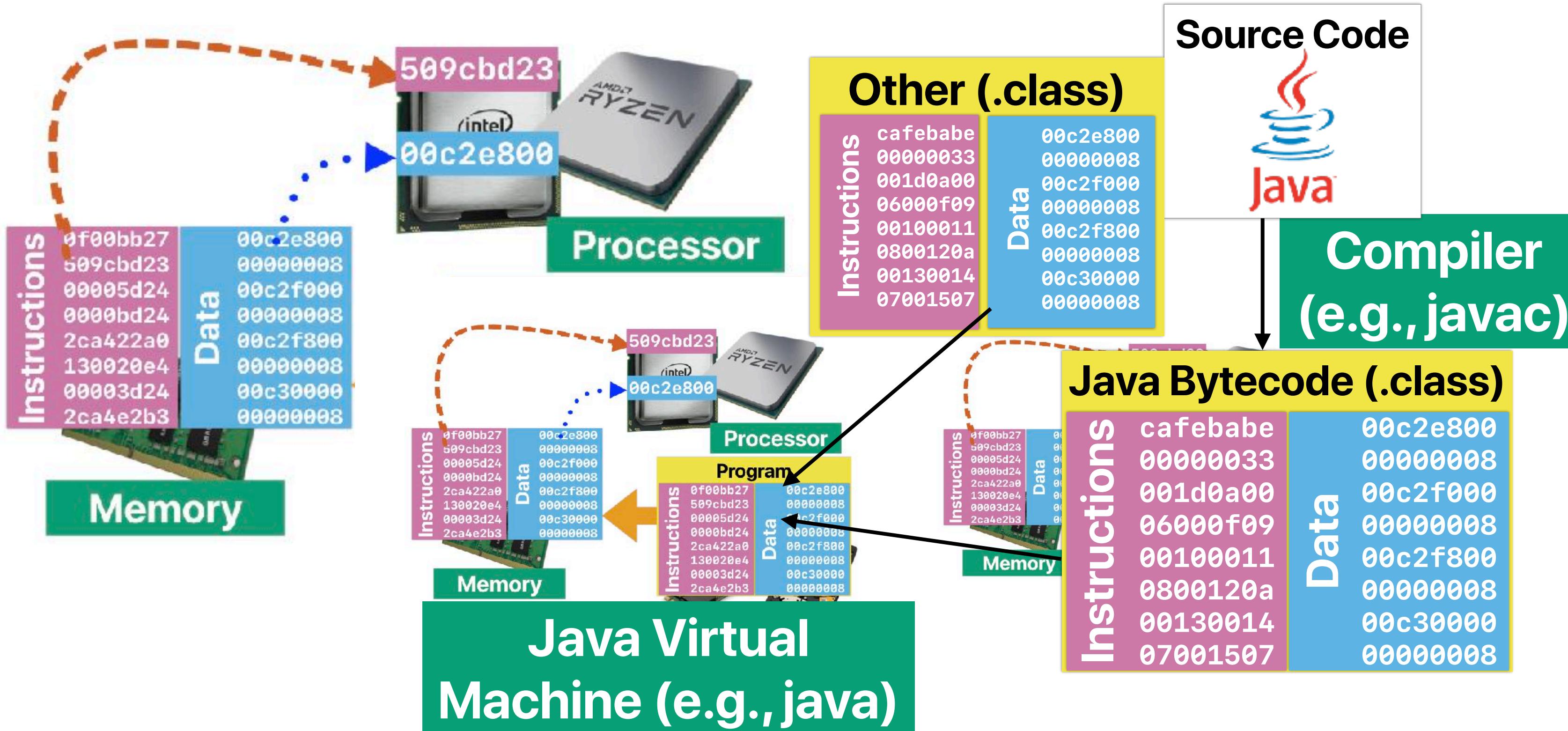
By loading different programs into memory,
your computer can perform different functions



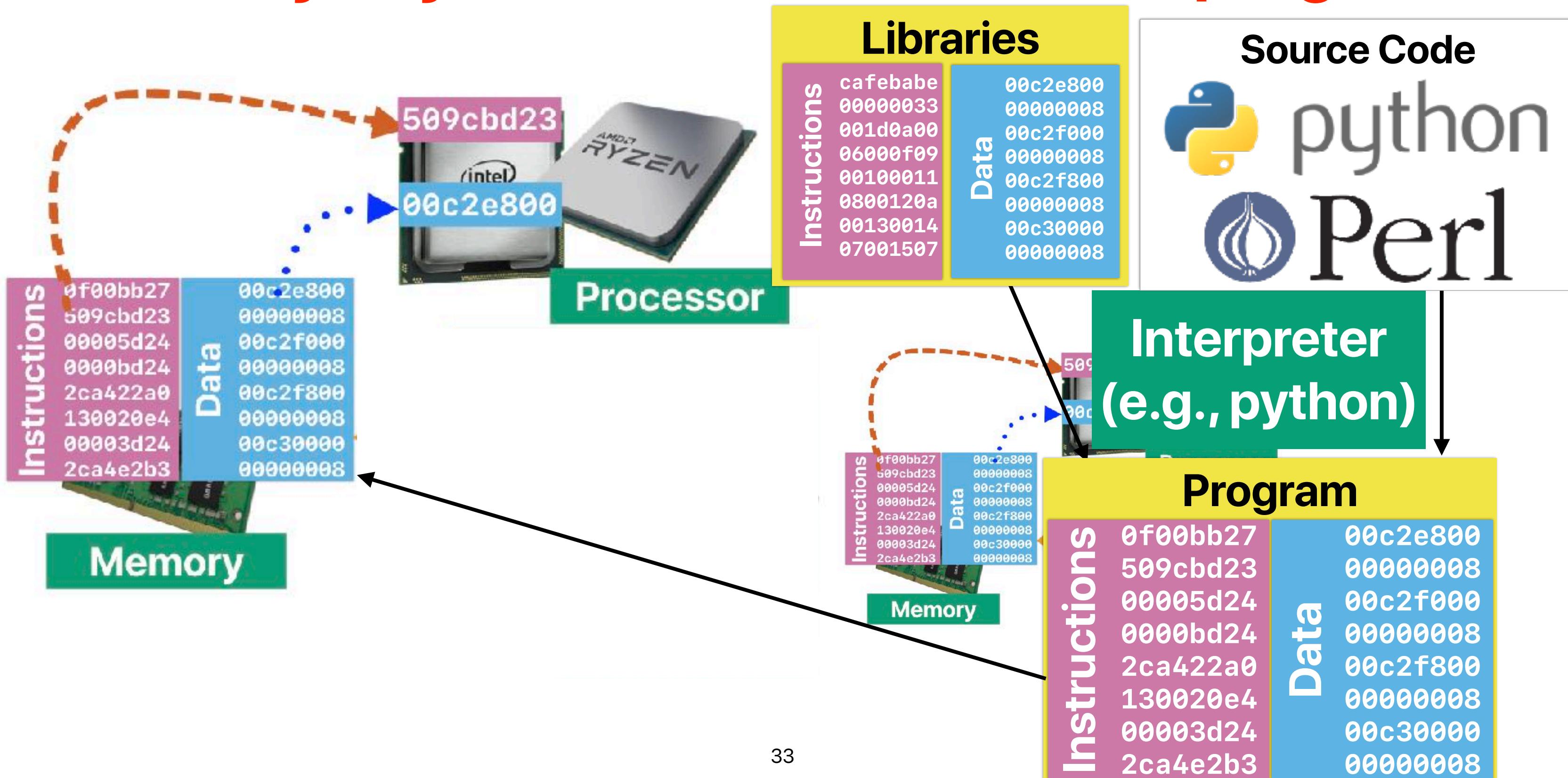
How my “C code” becomes a “program”



How my “Java code” becomes a “program”



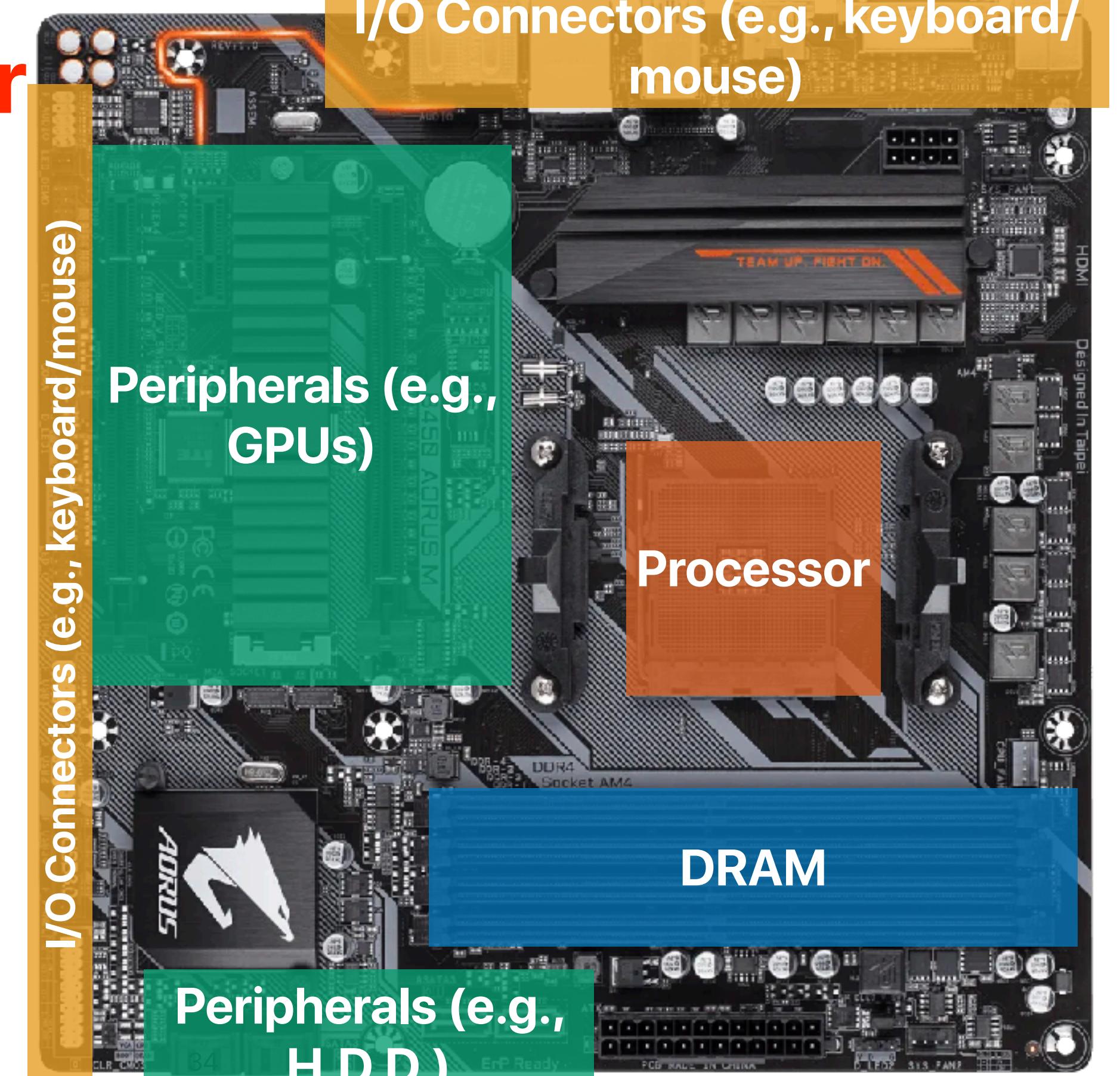
How my “Python code” becomes a “program”



Desktop Computer

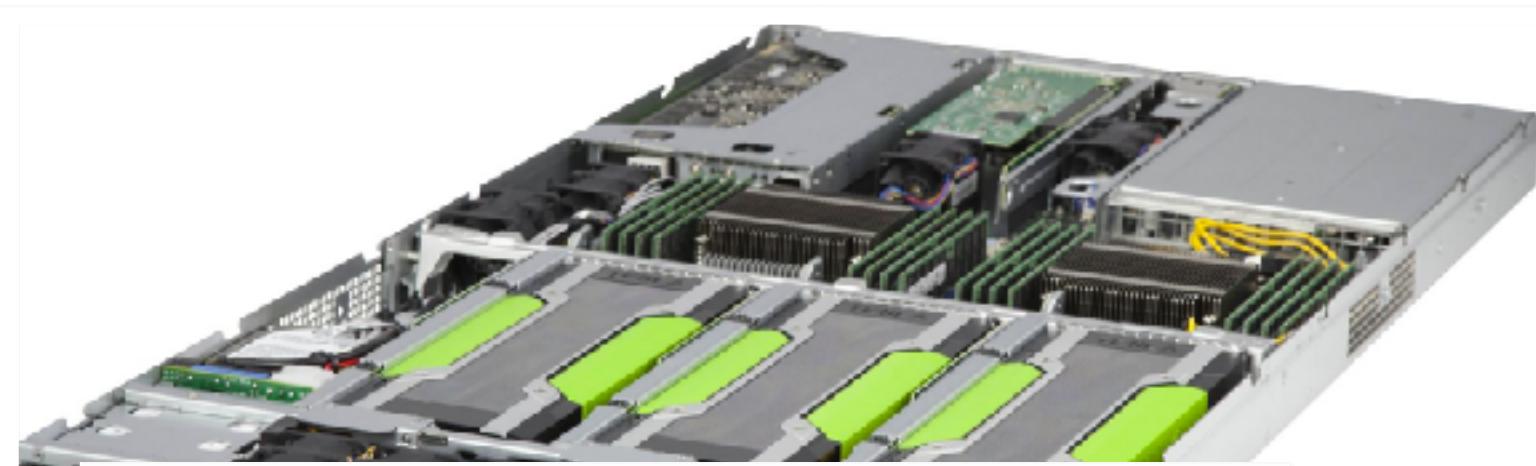


I/O Connectors (e.g., keyboard/mouse)



Server

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



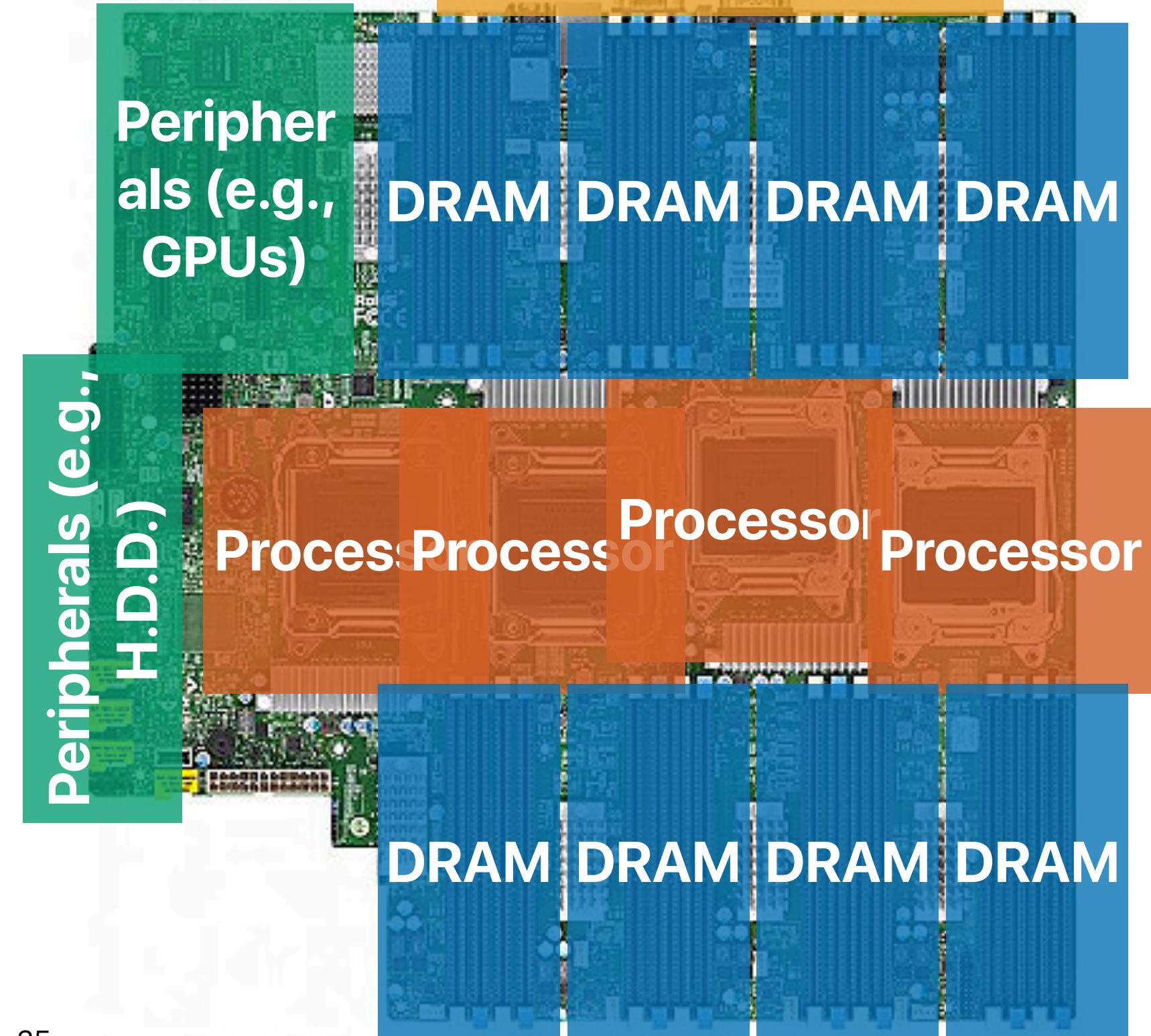
Peripherals (e.g., H.D.D.)

Peripherals (e.g., GPUs)

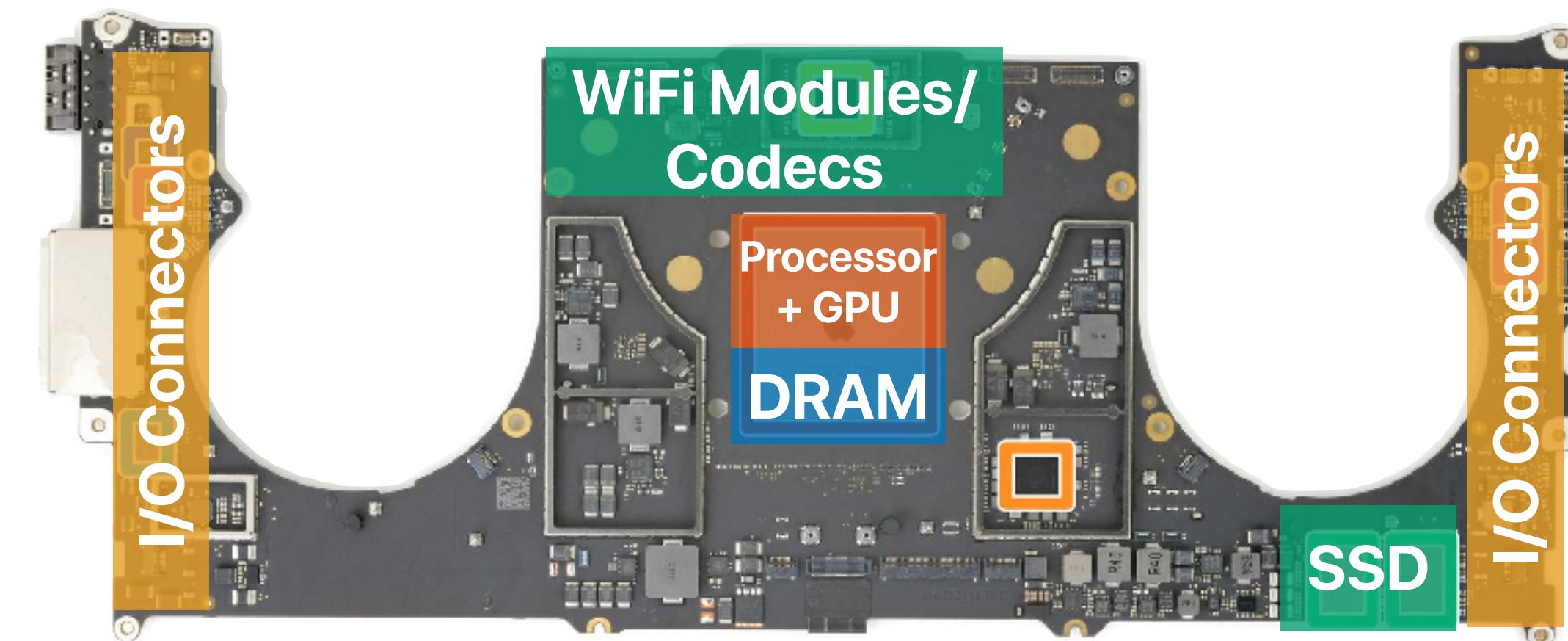
DRAM DRAM DRAM DRAM

Processor Processor Processor Processor

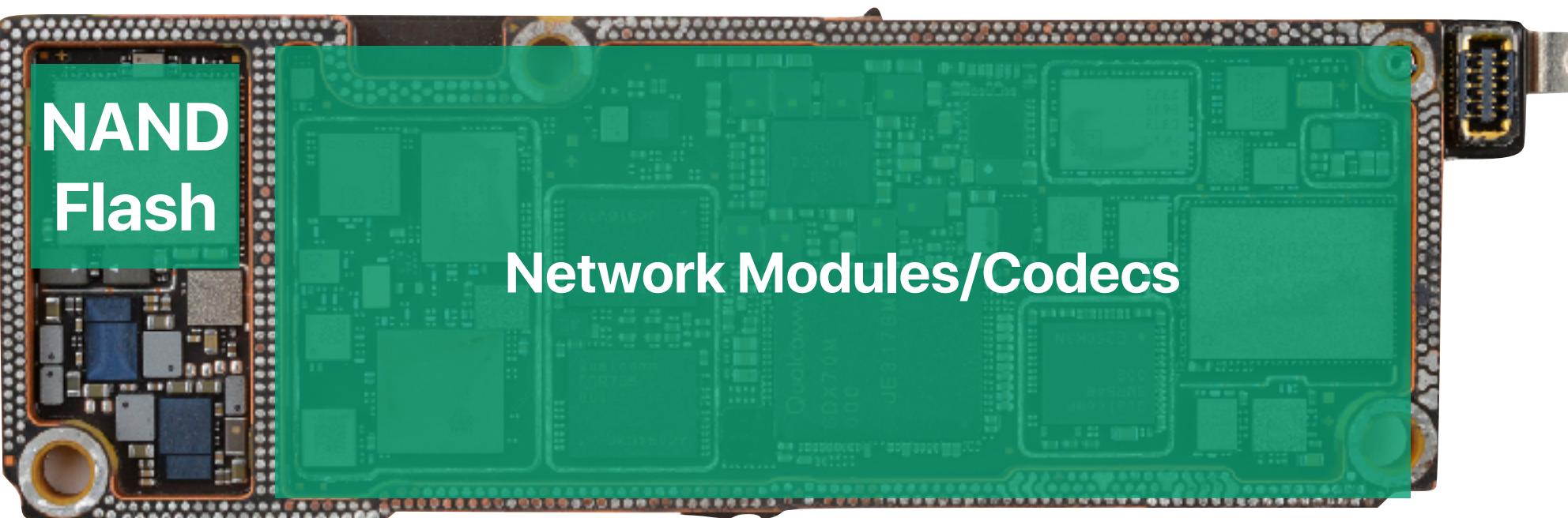
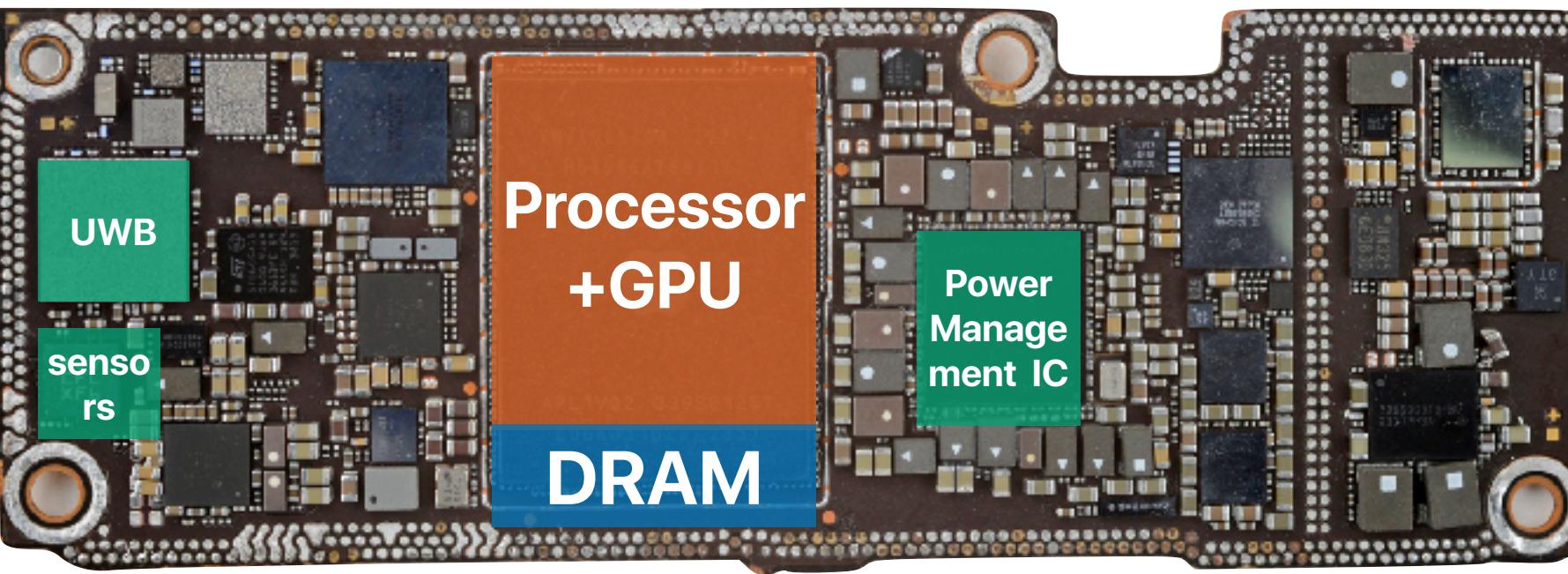
DRAM DRAM DRAM DRAM



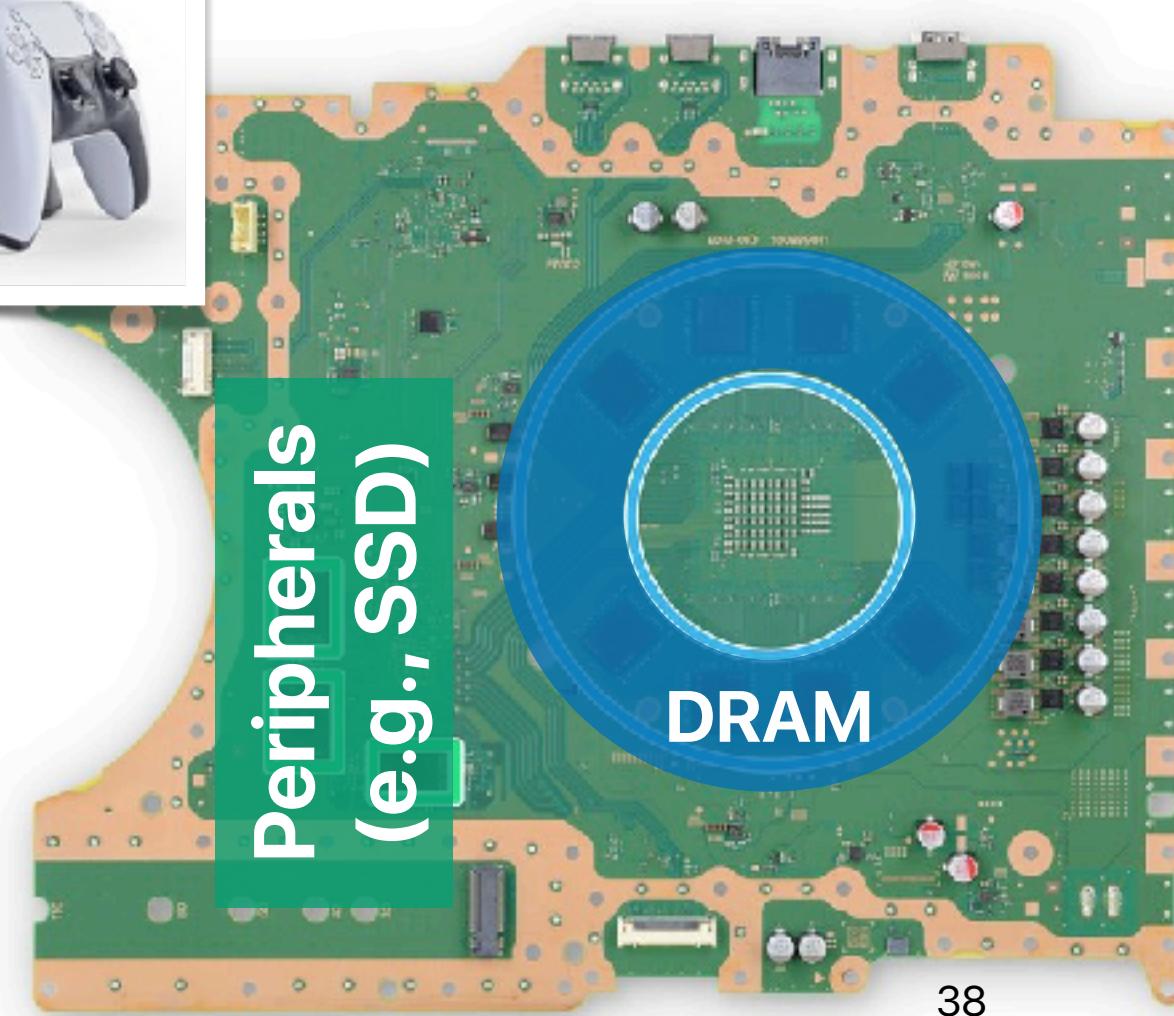
MacBook Pro M3



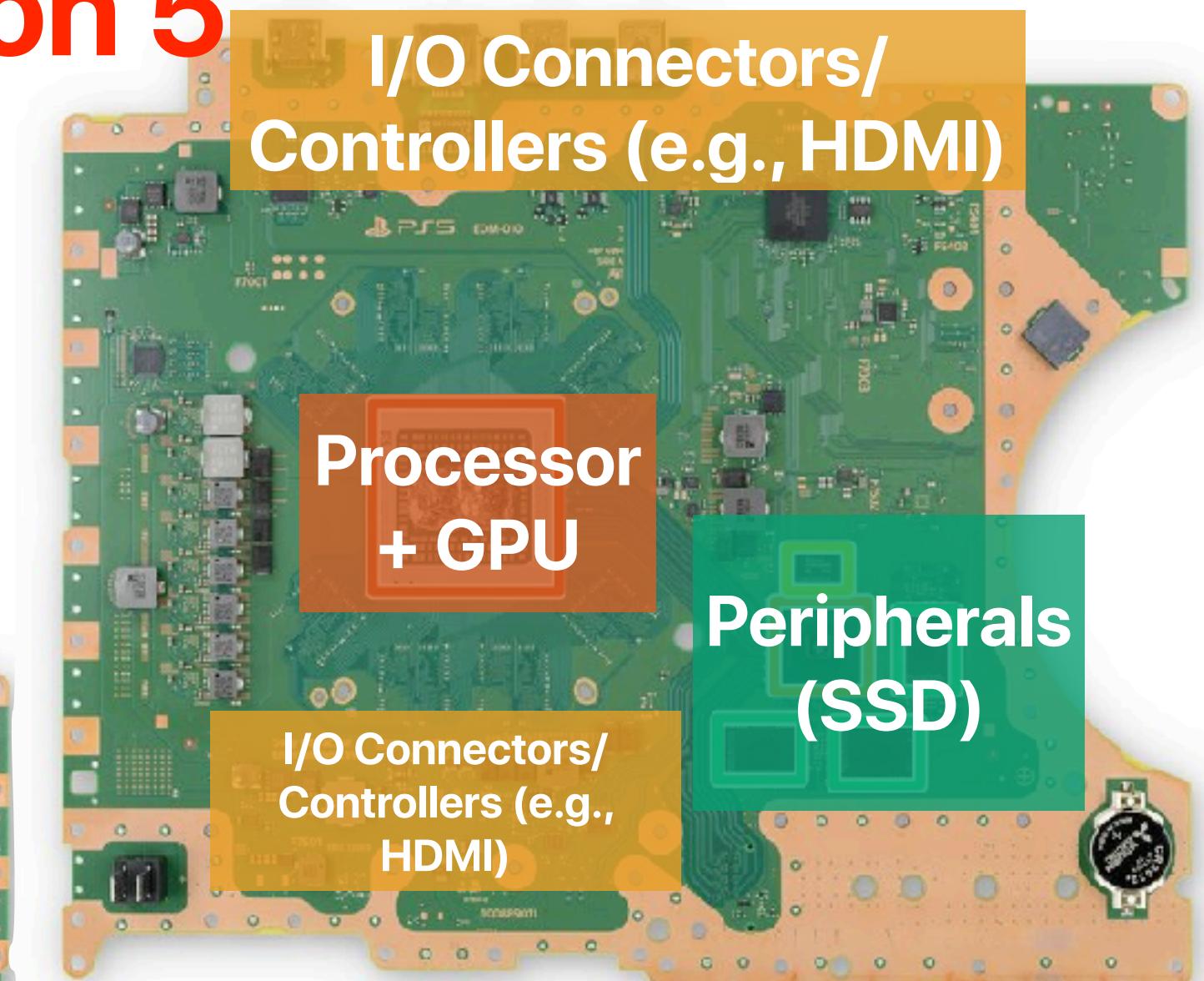
iPhone 15 Pro



Play Station 5



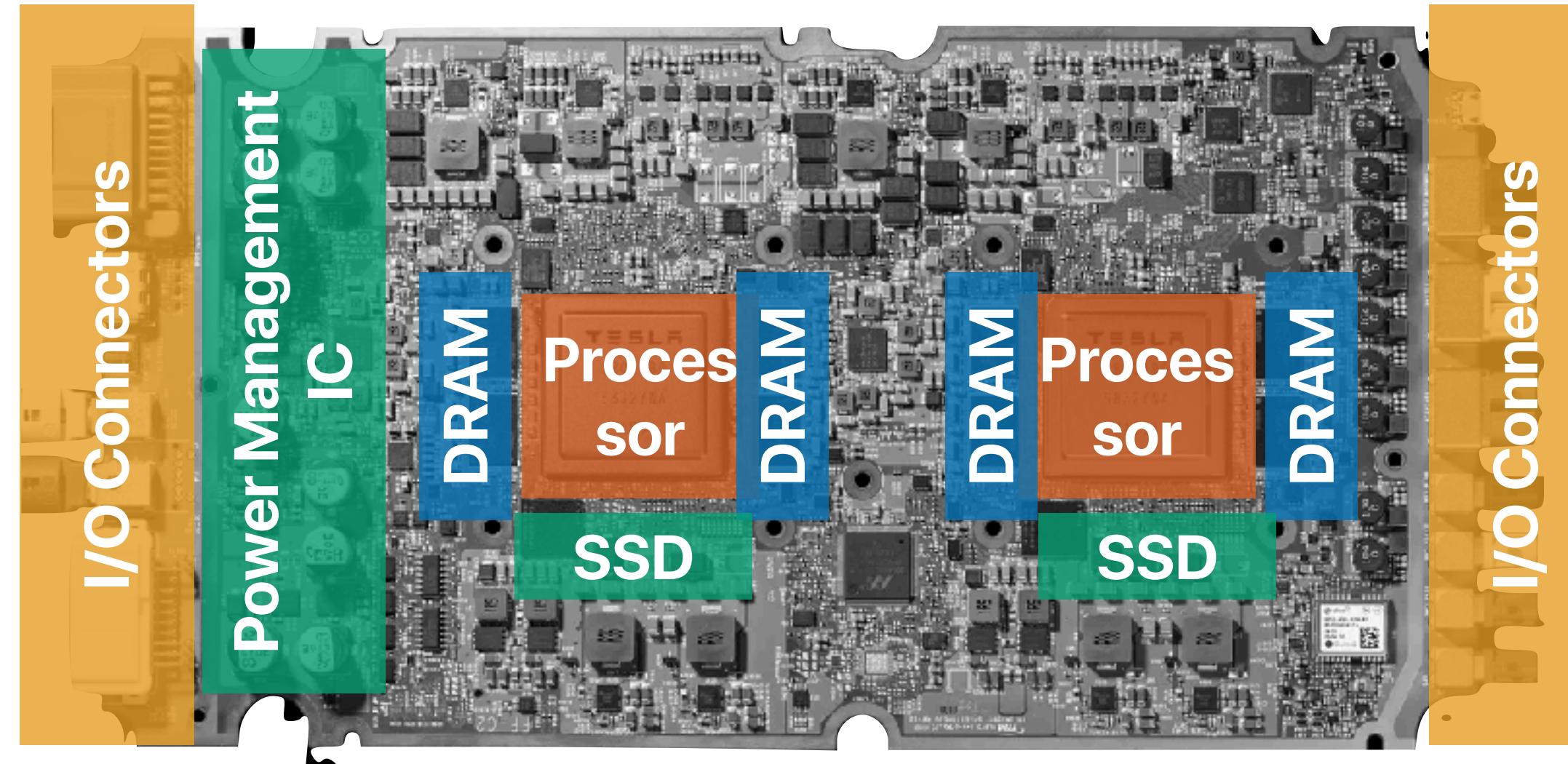
Peripherals
(e.g., SSD)



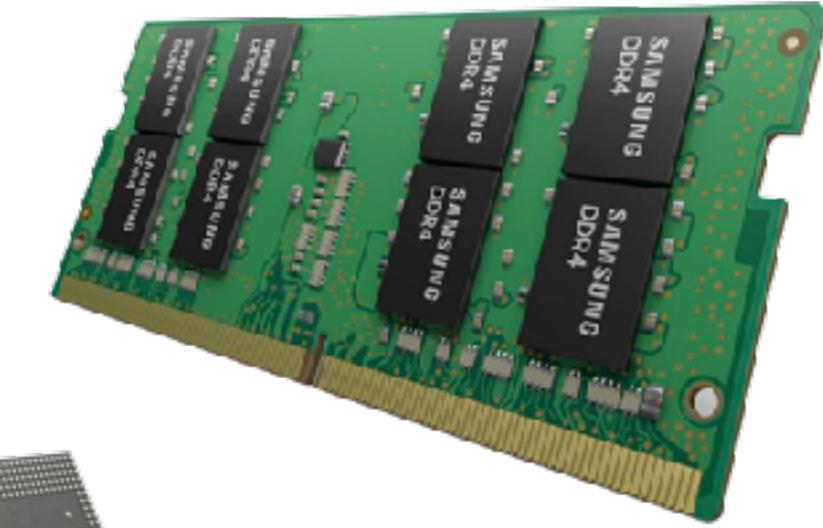
Nintendo Switch



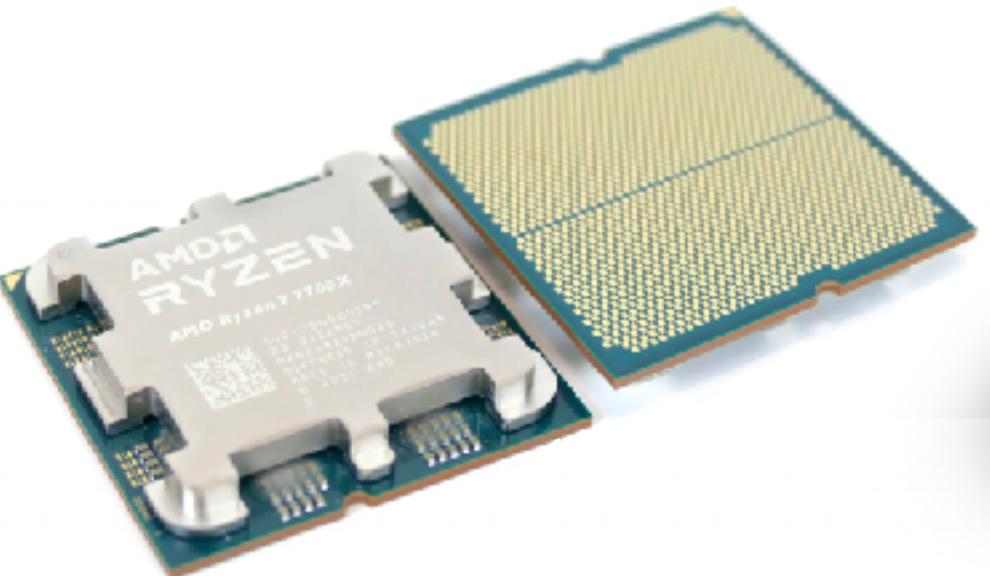
Tesla Model 3



Processors and memory modules are everywhere!

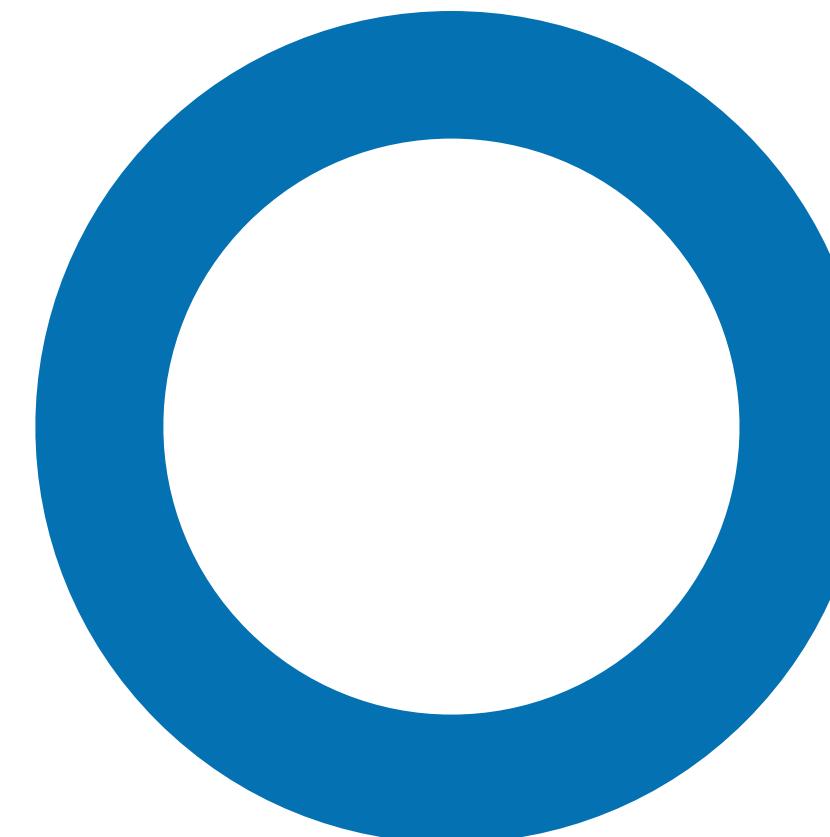


Processors



Memory

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



**How do “algorithm” and
“computational complexity” work on
modern computer architectures?**

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*)

On the same machine, programs with fewer lines of code 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];  
    }  
}
```

$O(n^2)$

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?

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

$O(n^2)$

A Lot Better!

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



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

main:
.LFB0:
endbr64
pushq %rbp
movq %rsp, %rbp
movl \$0, -8(%rbp)
movl \$0, -4(%rbp)
movl \$0, -8(%rbp)
jmp .L2

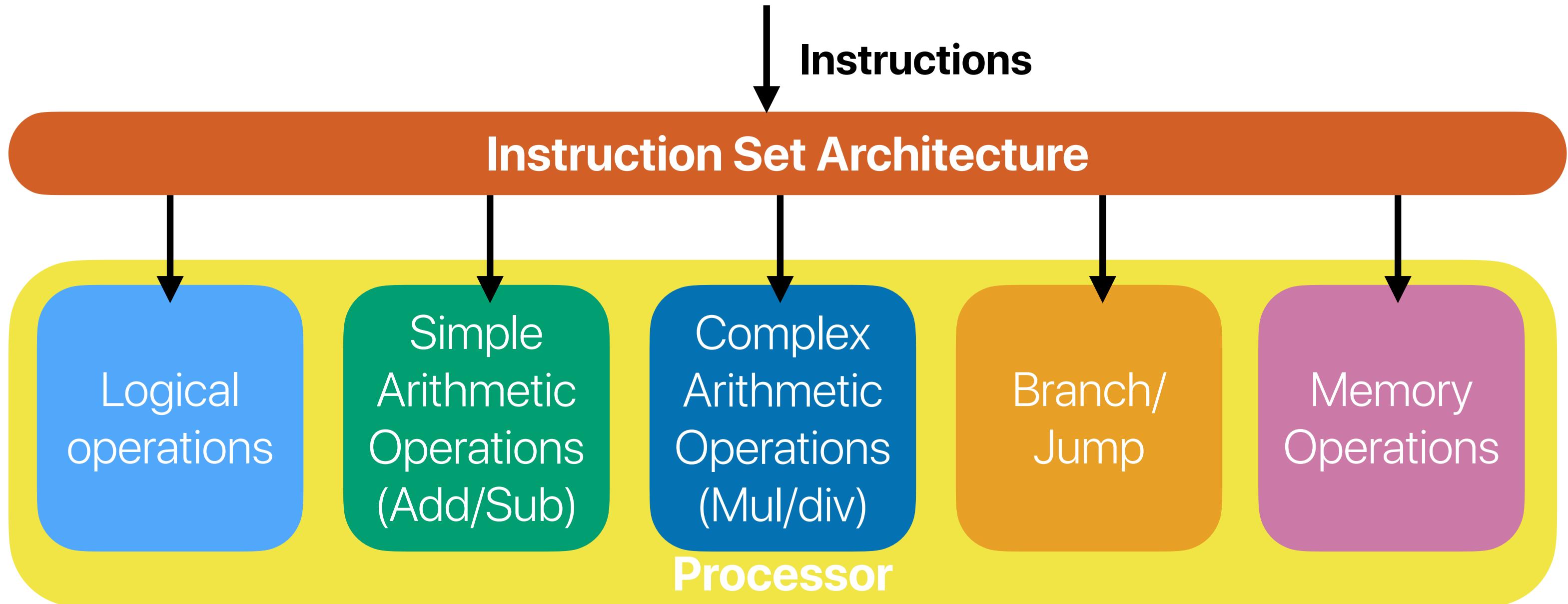
.L3:
movl -8(%rbp), %eax
cltq
leaq 0(%rax,4), %rdx
leaq A(%rip), %rax

Compiler

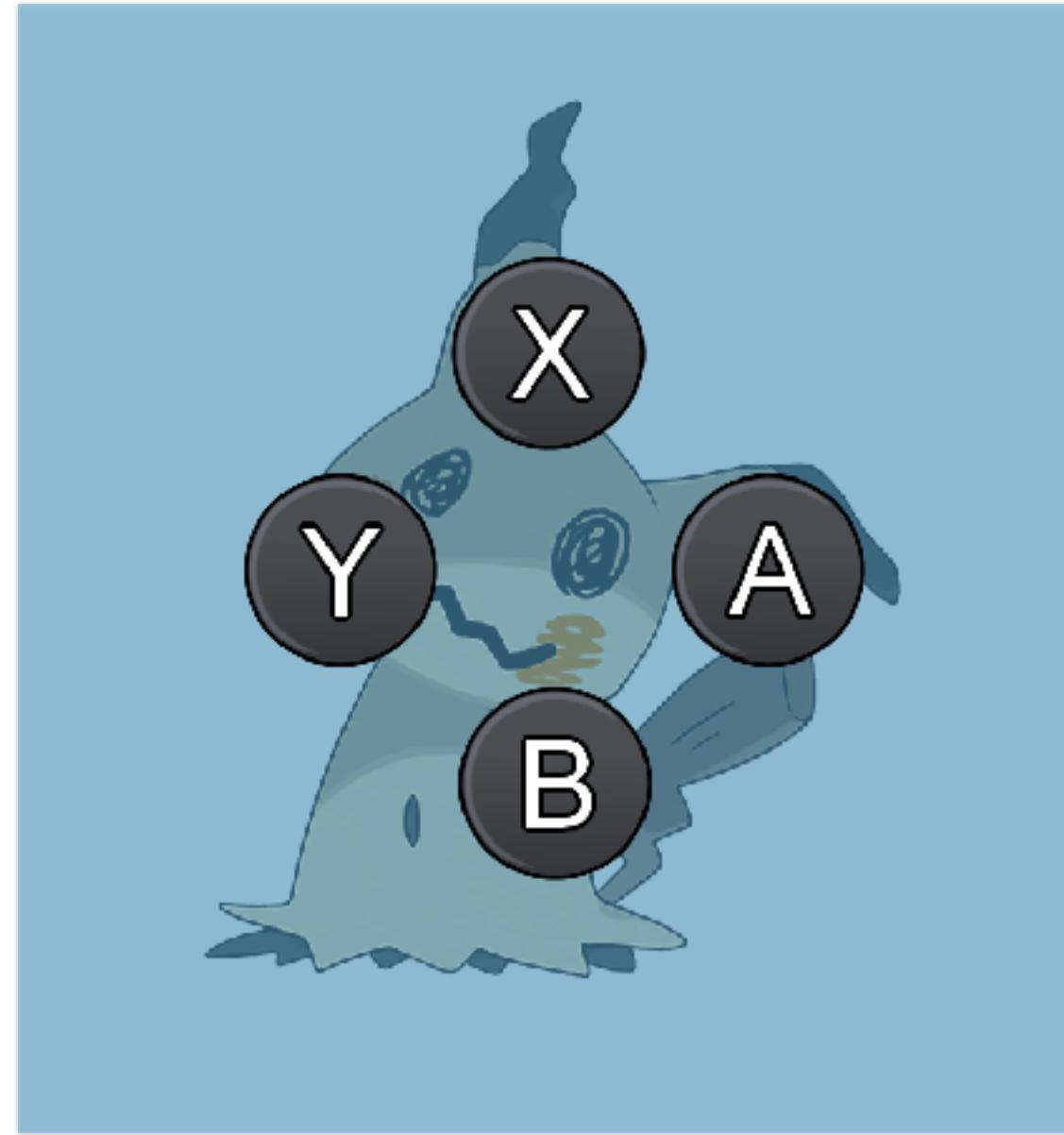
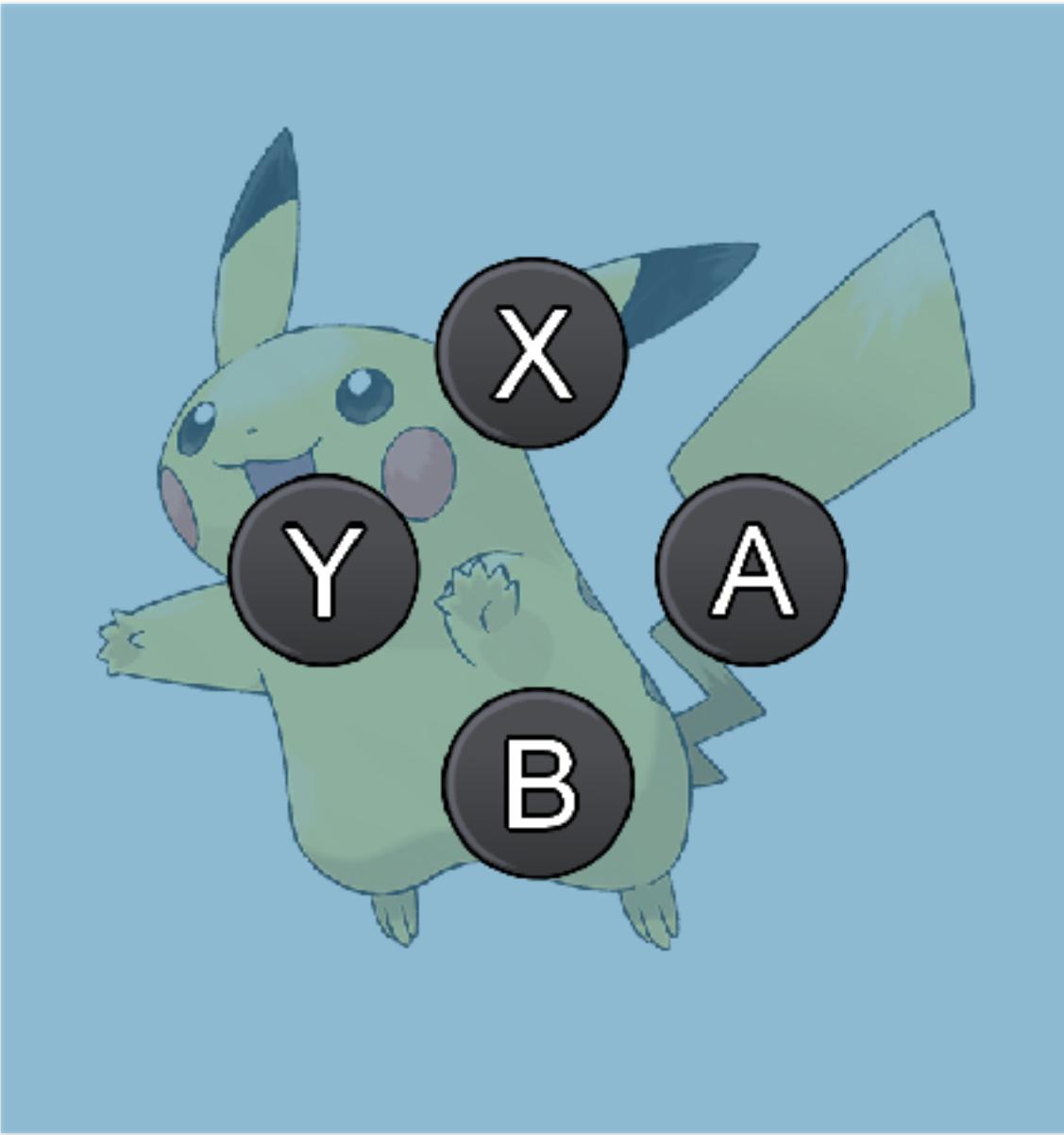
movl (%rdx,%rax),
%eax
addl %eax, -4(%rbp)
addl \$1, -8(%rbp)
.L2:
cmpl \$19, -8(%rbp)
jle .L3
movl \$0, %eax
popq %rbp
ret

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

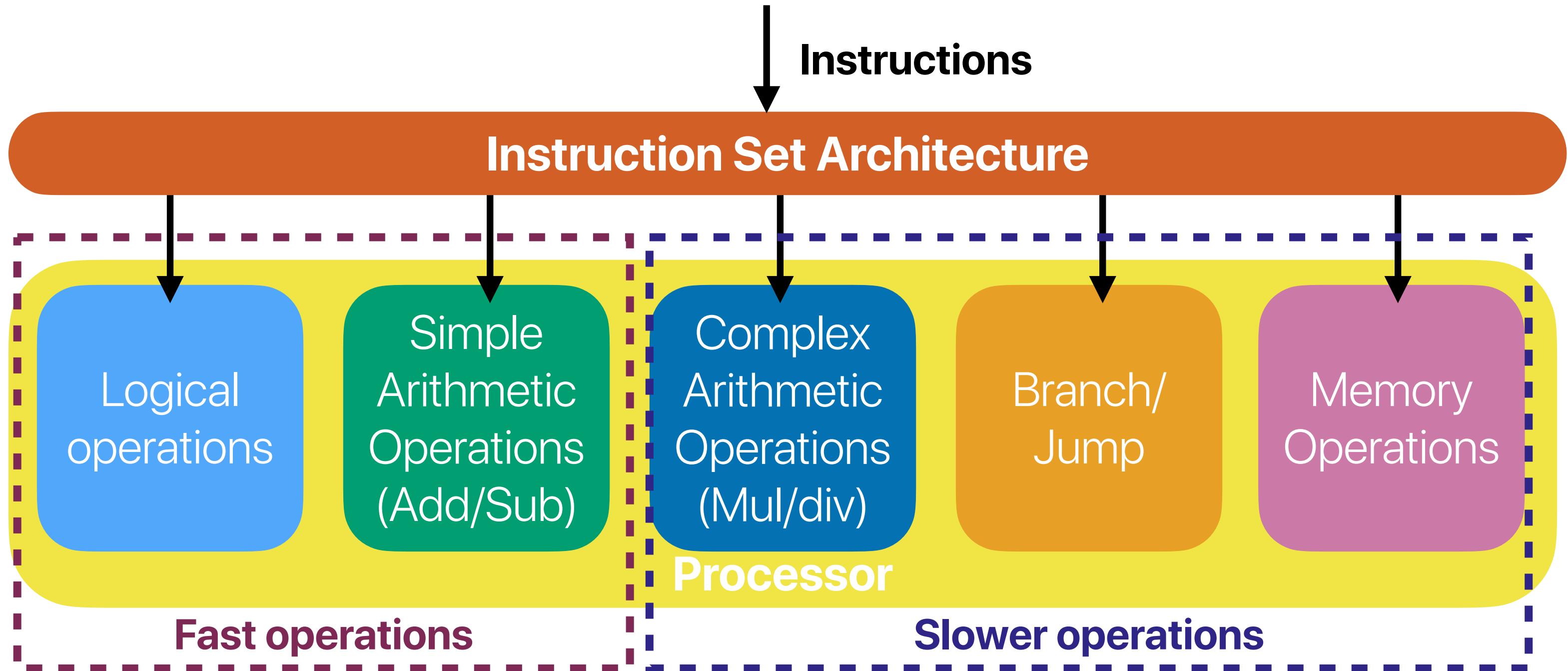
Microprocessor — a collection of functional units



ISA — the “abstraction” of processor features



Microprocessor — a collection of functional units



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;  
}
```

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

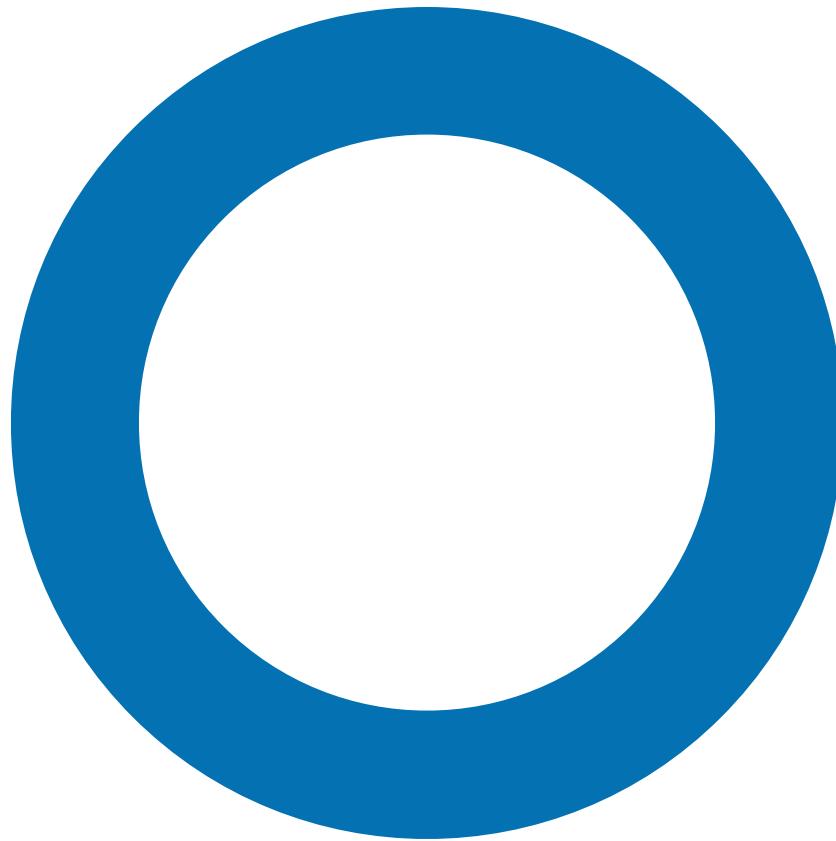
$$O(n \log_2 n)$$

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);  
                }  
            }  
        }  
    }  
}
```

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



Quicksort: $O(n \log_2 n)$ algorithm, but execution time goes up to 4 × when n is 2 ×

Bitonic sort: $O(n^2 \log_2 n)$ algorithm, but execution time does not change a lot when n is 2 ×

Quicksort: $O(n \log_2 n)$ algorithm, but underperforming bitonic sort whose complexity is $O(n^2 \log_2 n)$

Take-aways: Why CSE142?

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

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

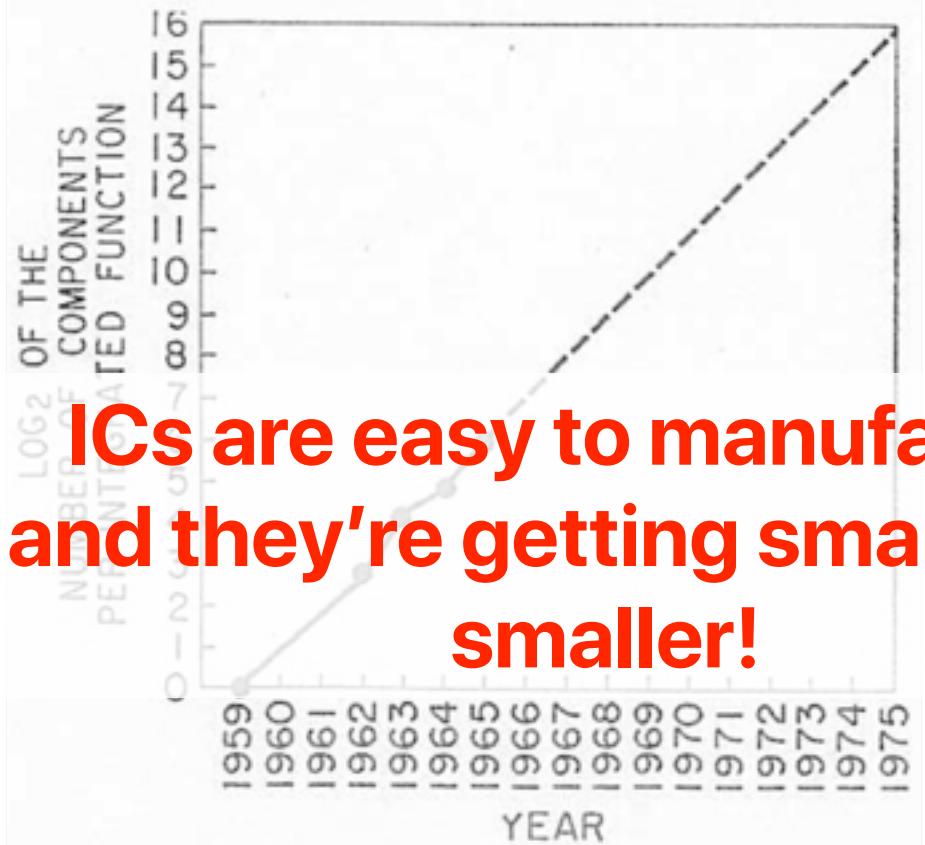
ICs are small

(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

units. The lack of large-value capacitors and

inductors makes it difficult to implement

linear circuitry in the integrated form.

In almost every case, however, the demonstrated 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 widely applicable

Reliability

coun

shed

or nev

and v

able

ollo,

he reli

s, tha

ilure as the

stablish

Heat problem

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

Moore's Law sets the pace

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

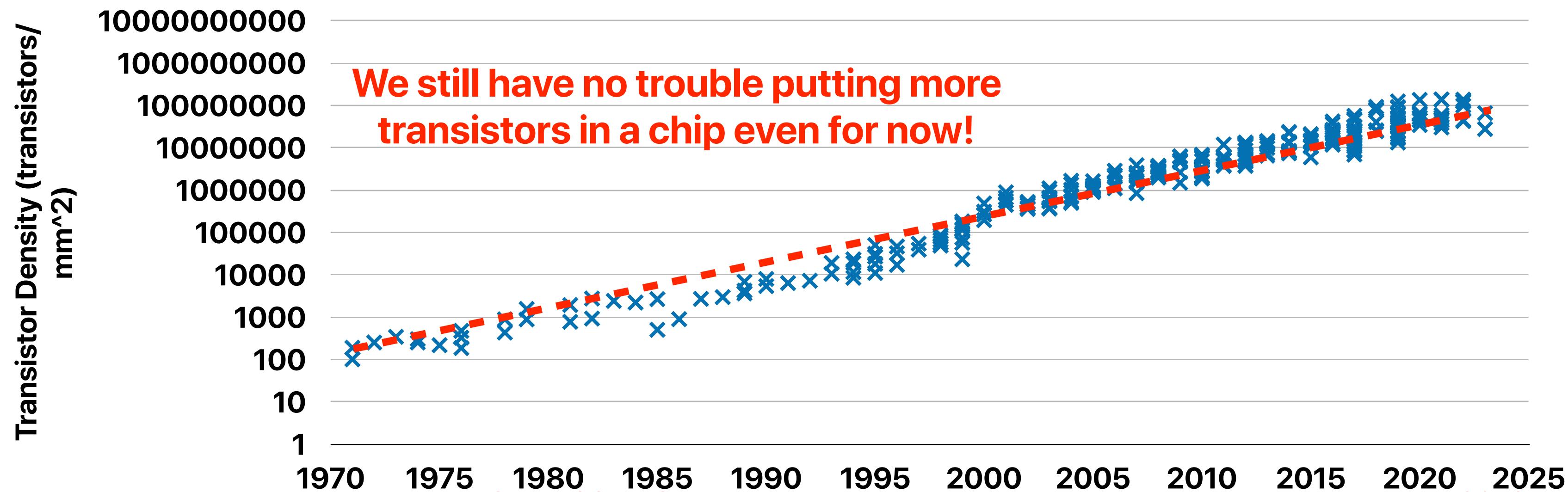
sign automation procedures could translate from any special engineering.

Designing ICs can be easy

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

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

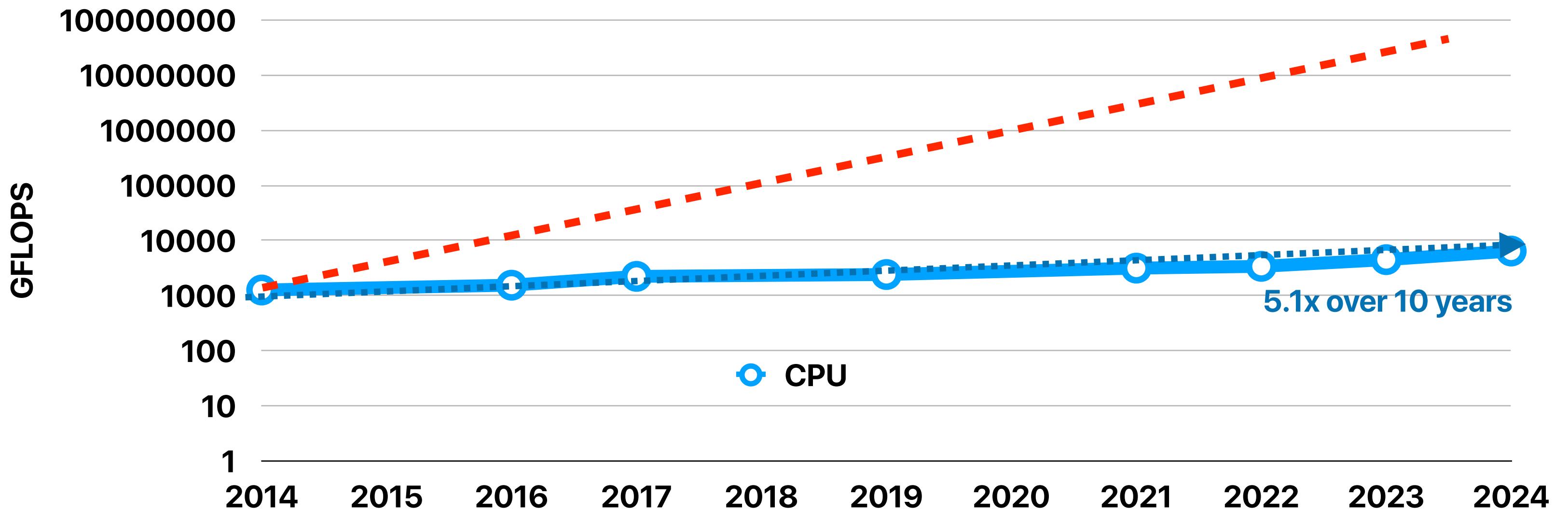


(1) Moore, G. E. (1965), 'Cramming more components onto integrated circuits', Electronics 38 (8).

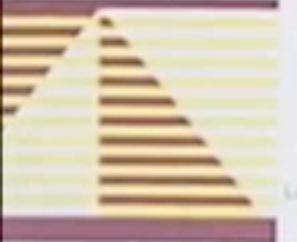
Moore's Law is currently discontinued



CPU Performance v.s. Moore's Law

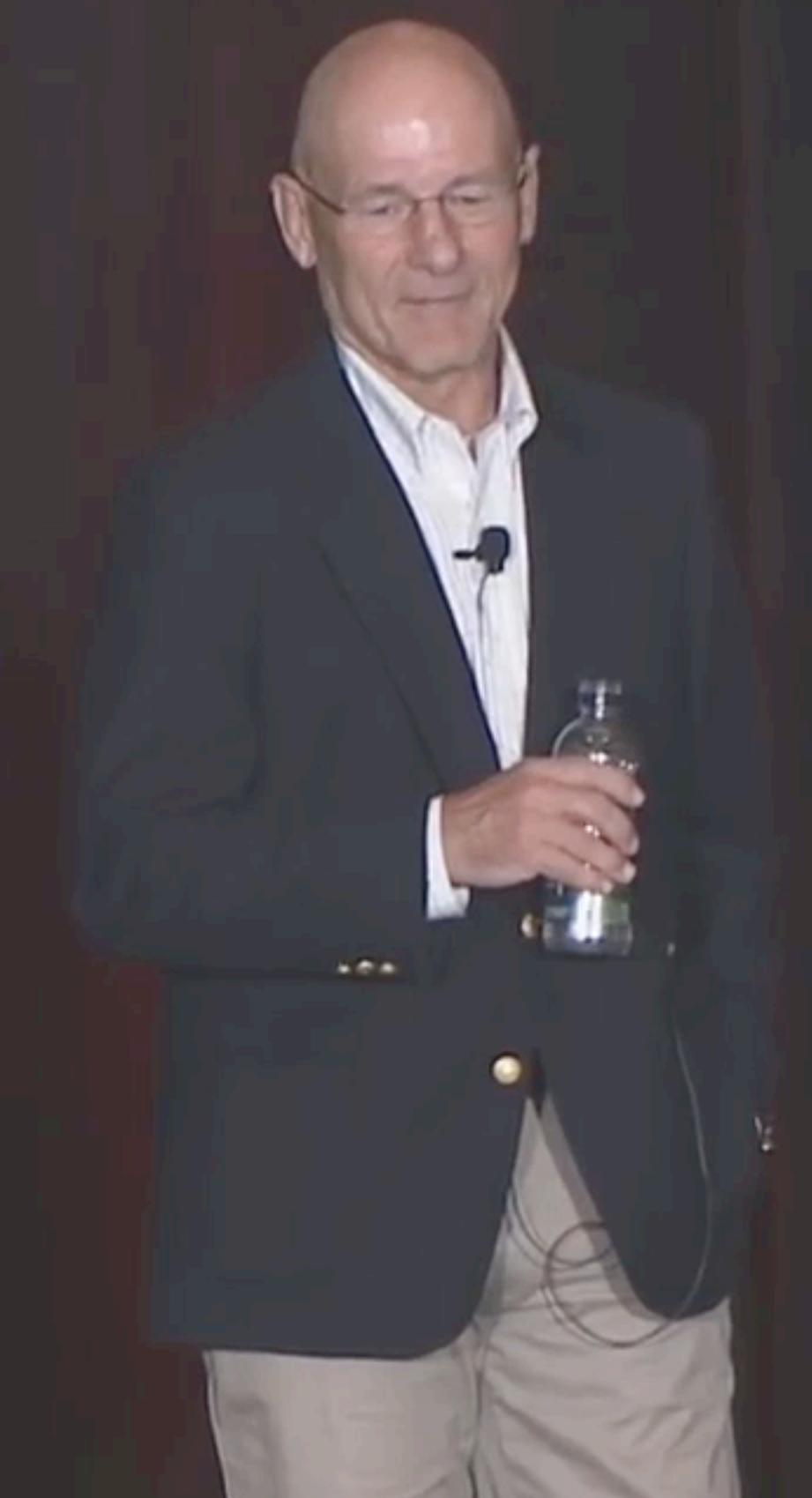
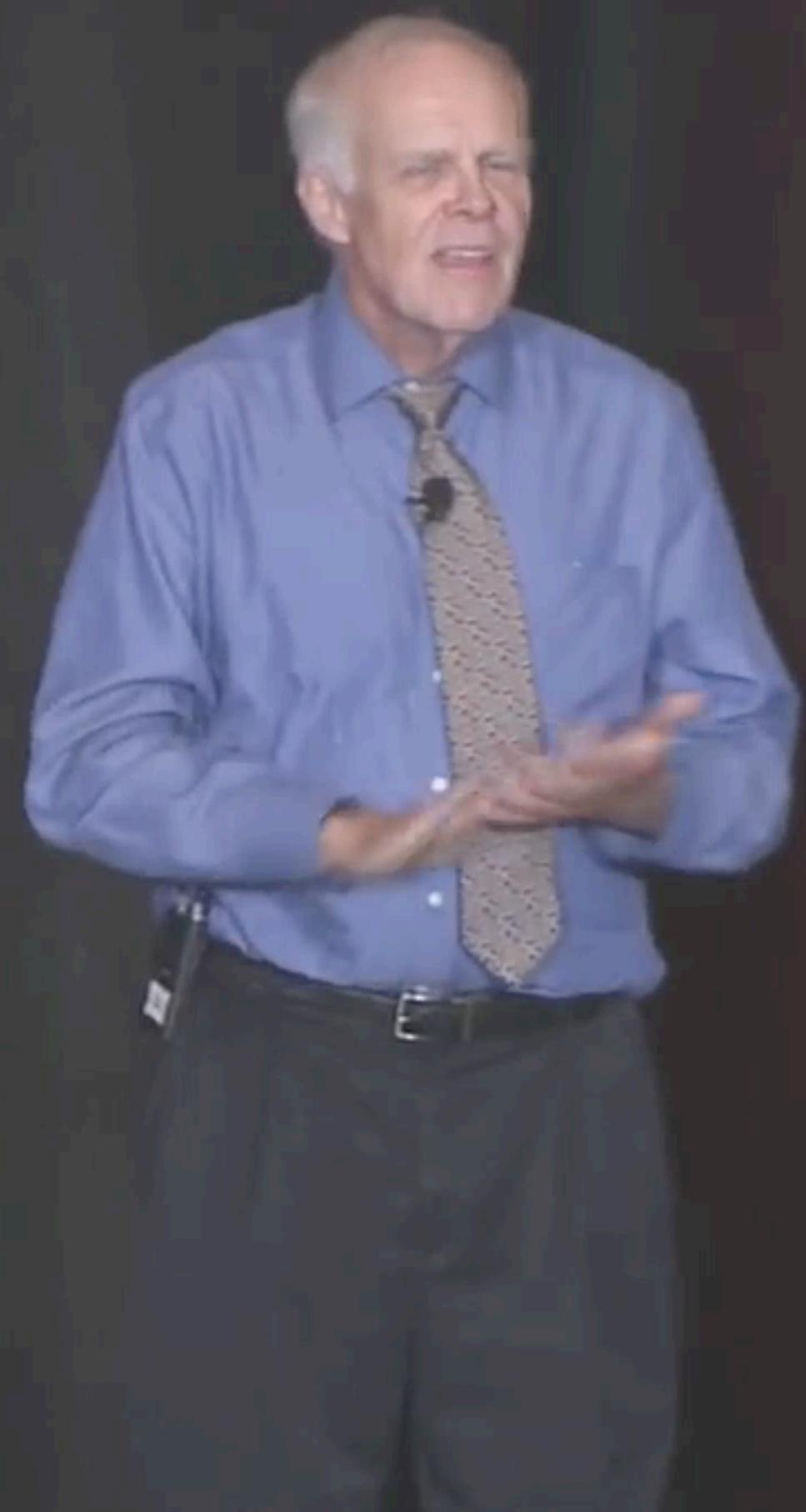


<https://ourworldindata.org/grapher/artificial-intelligence-training-computation>

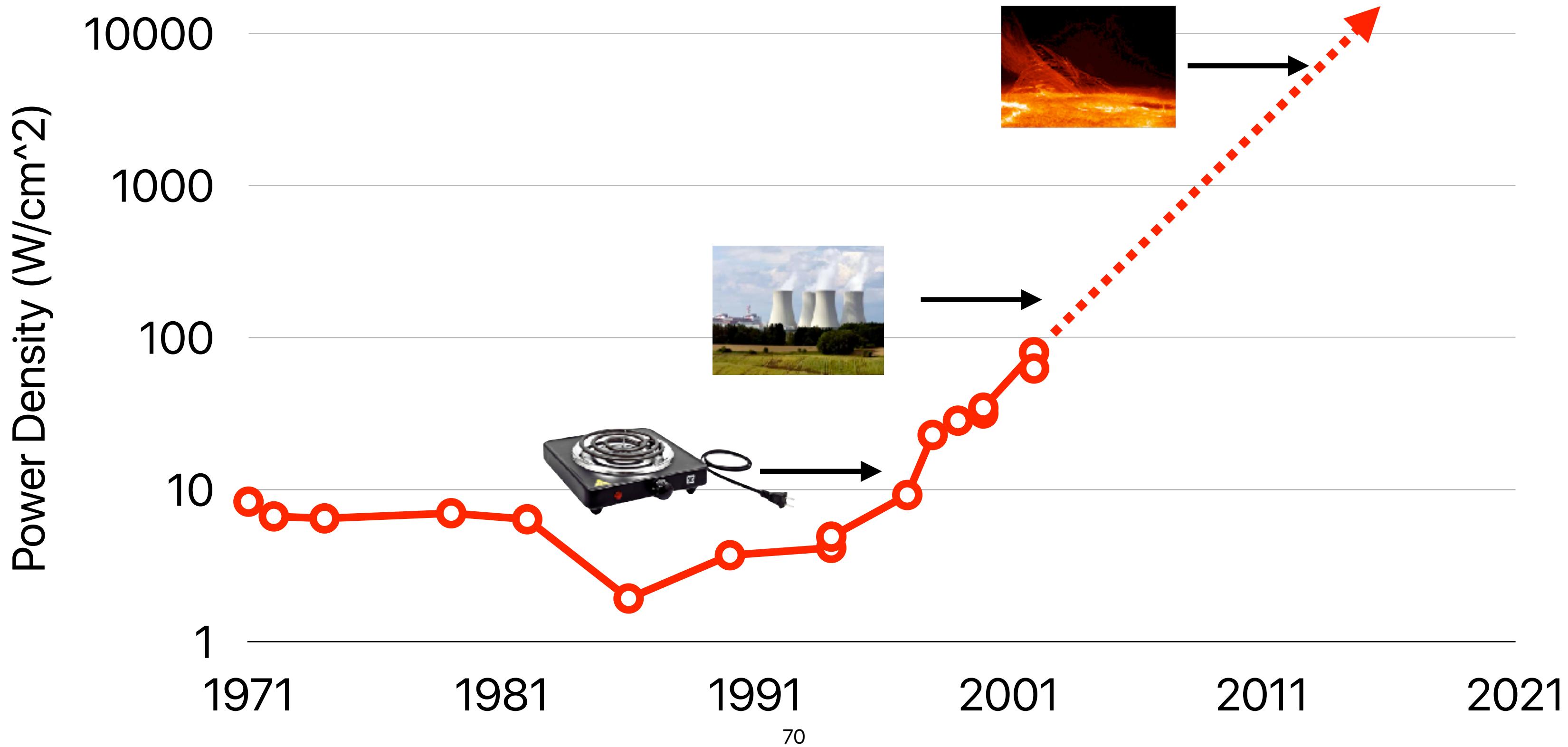


The 45th
ACM/IEEE
International
Symposium
on Computer
Architecture
Los Angeles, USA

ISCA 2018
uring Lecture



Power Density of Processors



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.

5.77 W/1B transistors

TSMC 4nm

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.

5.73 W/1B transistors

Intel 7nm

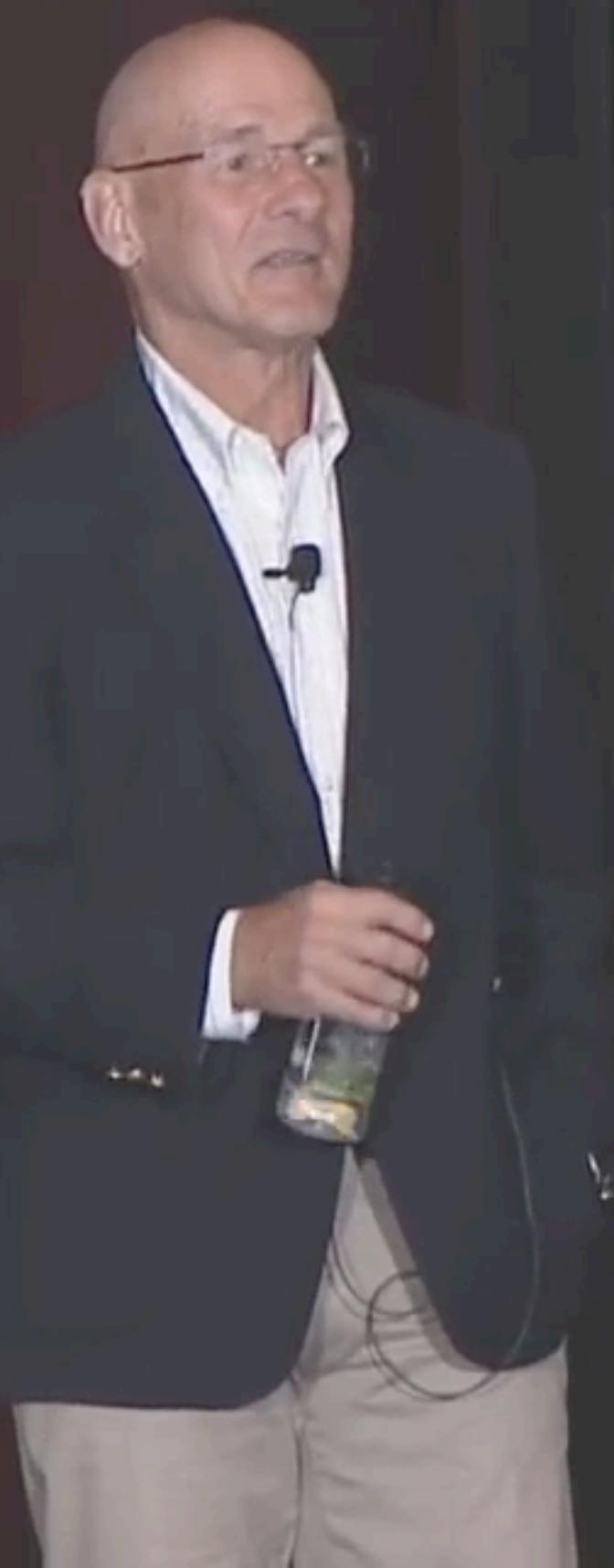
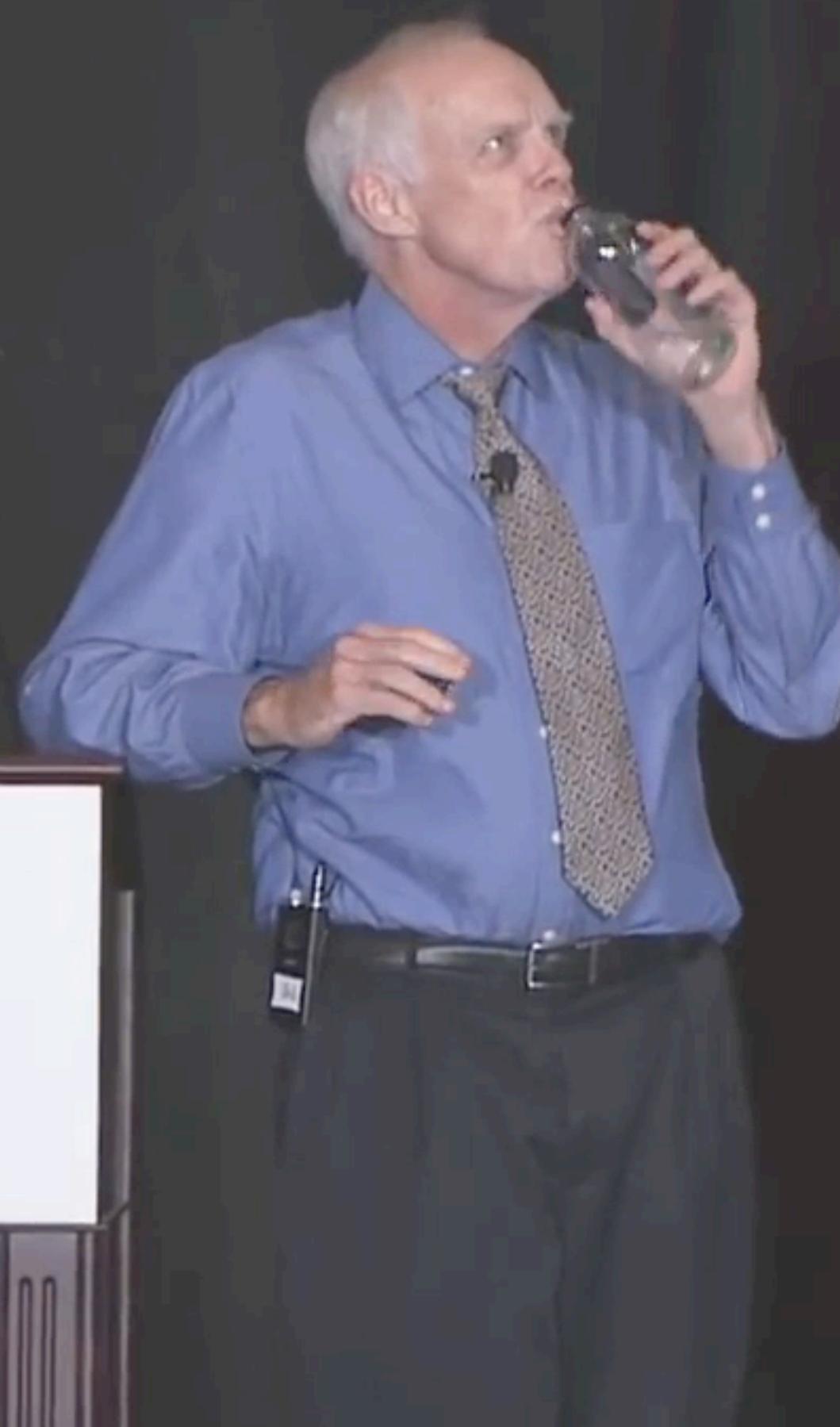
**The smaller size of a transistor,
the smaller power consumption of it**





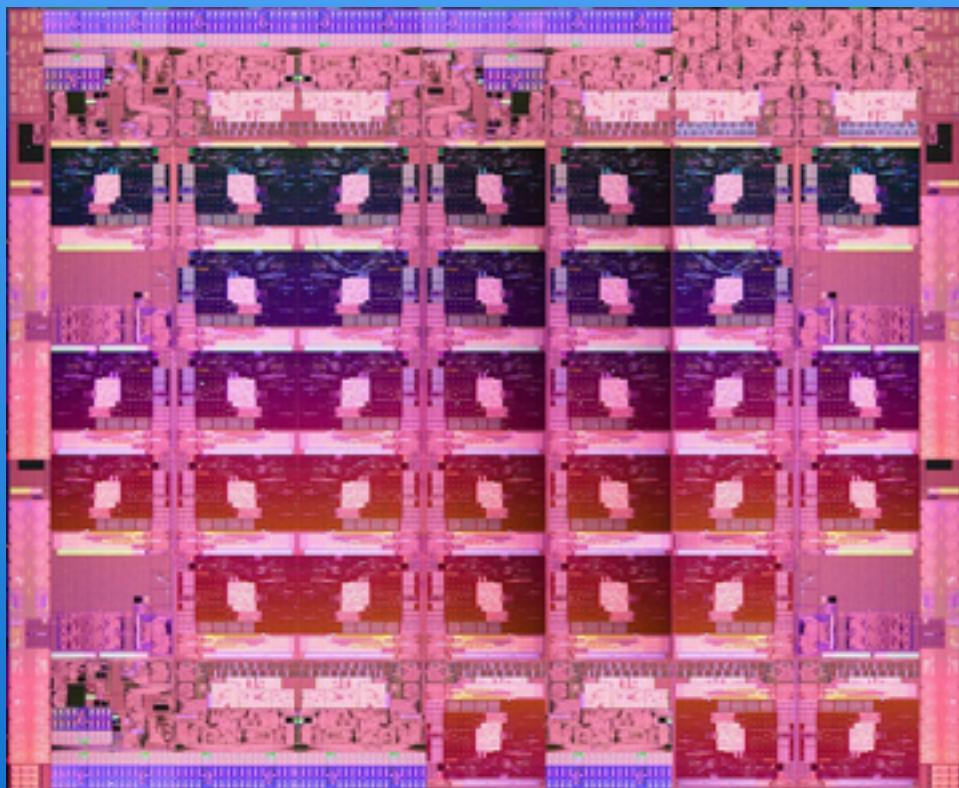
The 45th
ACM/IEEE
International
Symposium
on Computer
Architecture
Los Angeles, USA

A 2018
g Lecture



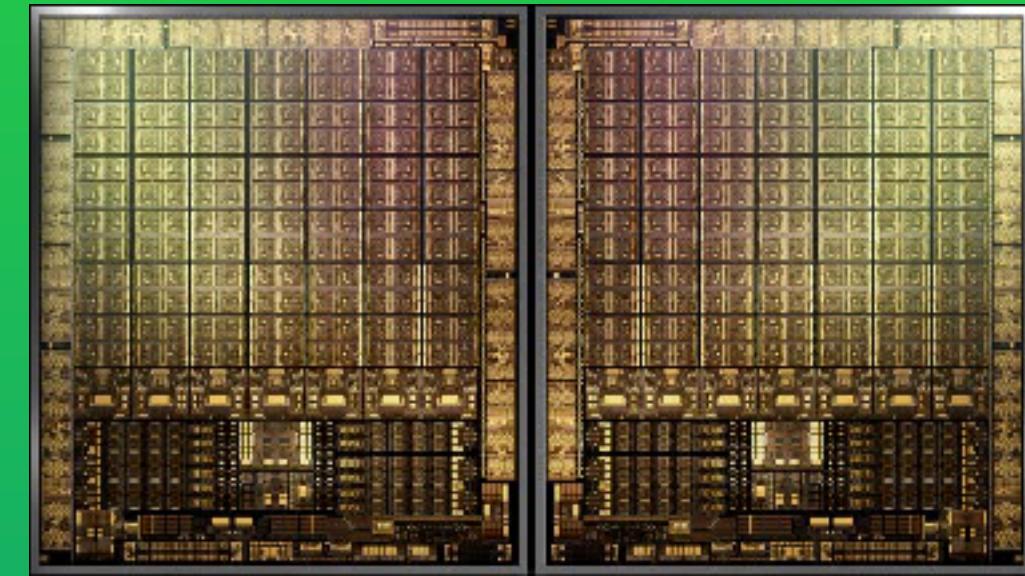
Alternatives to scaling single cores

Multicore Processors

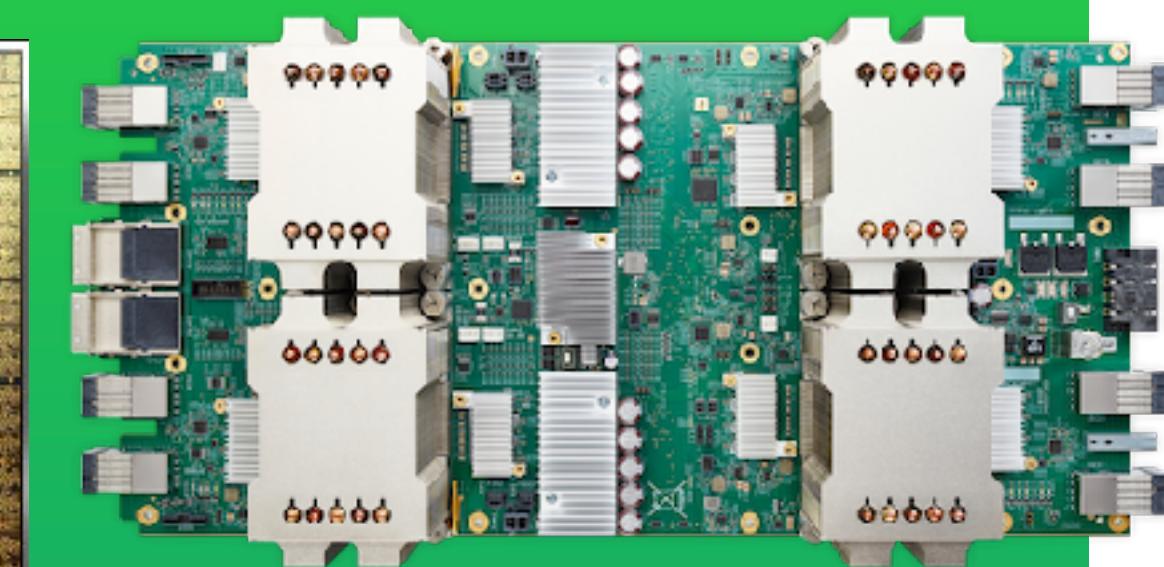


34-core Intel
Sapphire Rapids

Hardware Accelerators



NVIDIA H100
Graphic Processing Units

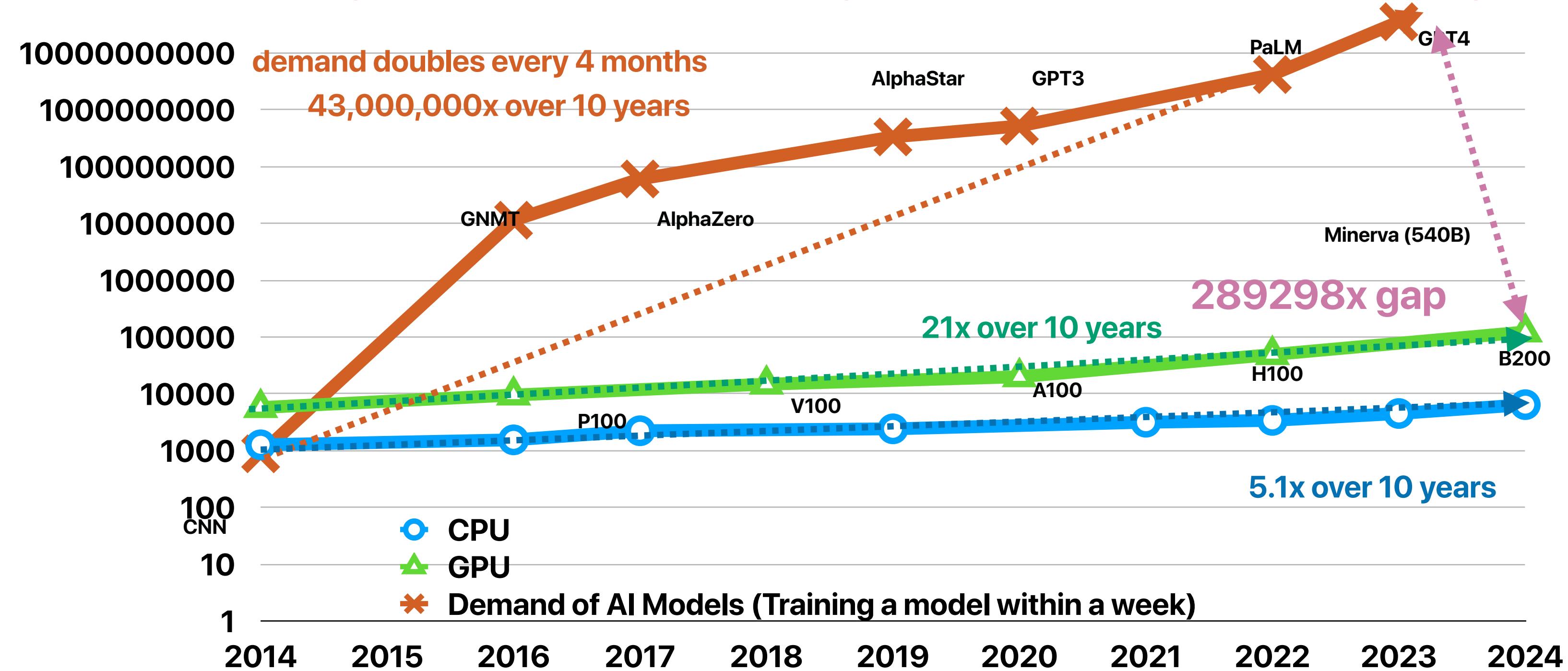


Google
Tensor Processing Units

Thread-level parallelism

Data-level parallelism

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



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

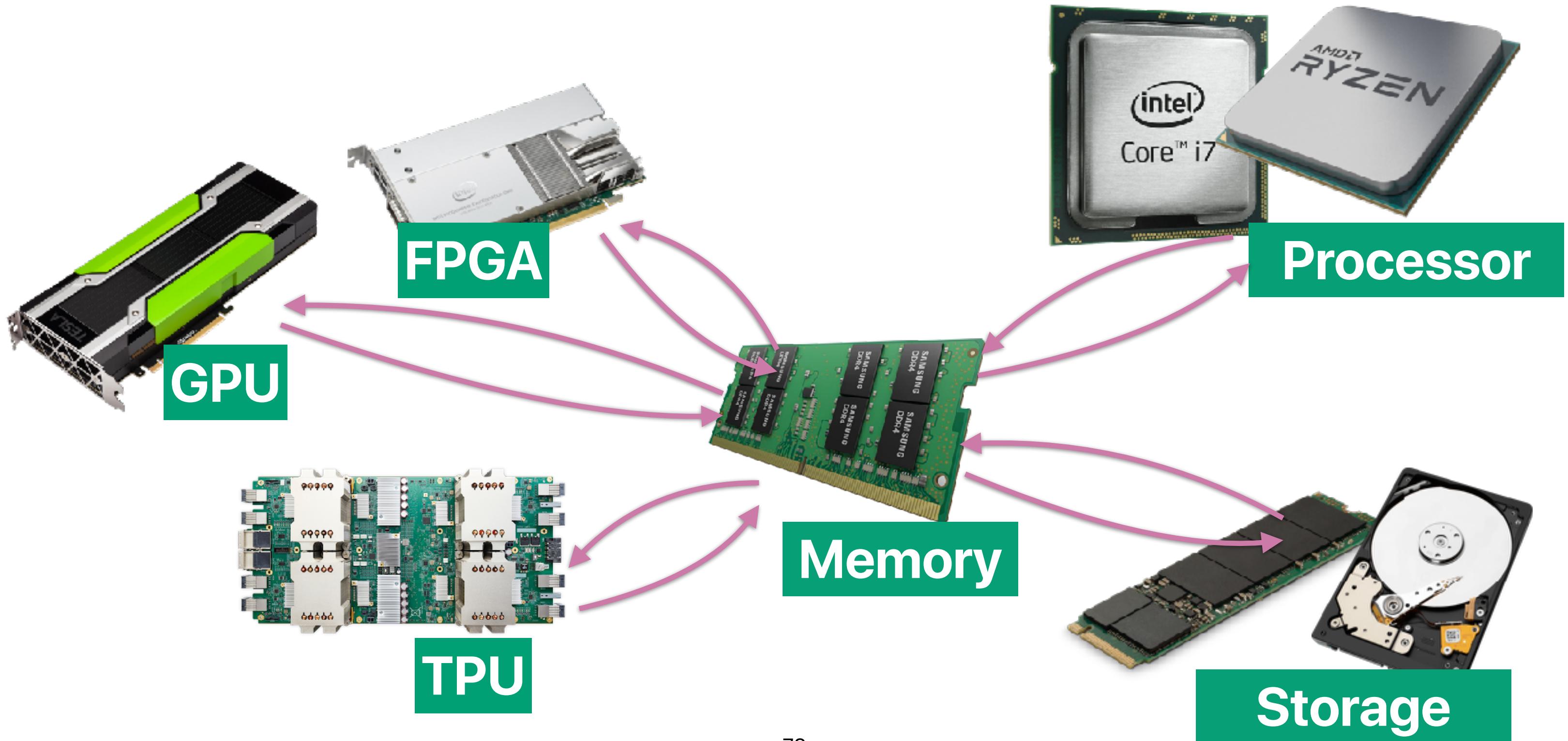
The most recent GPUs can match the demand of efficient AI/ML training



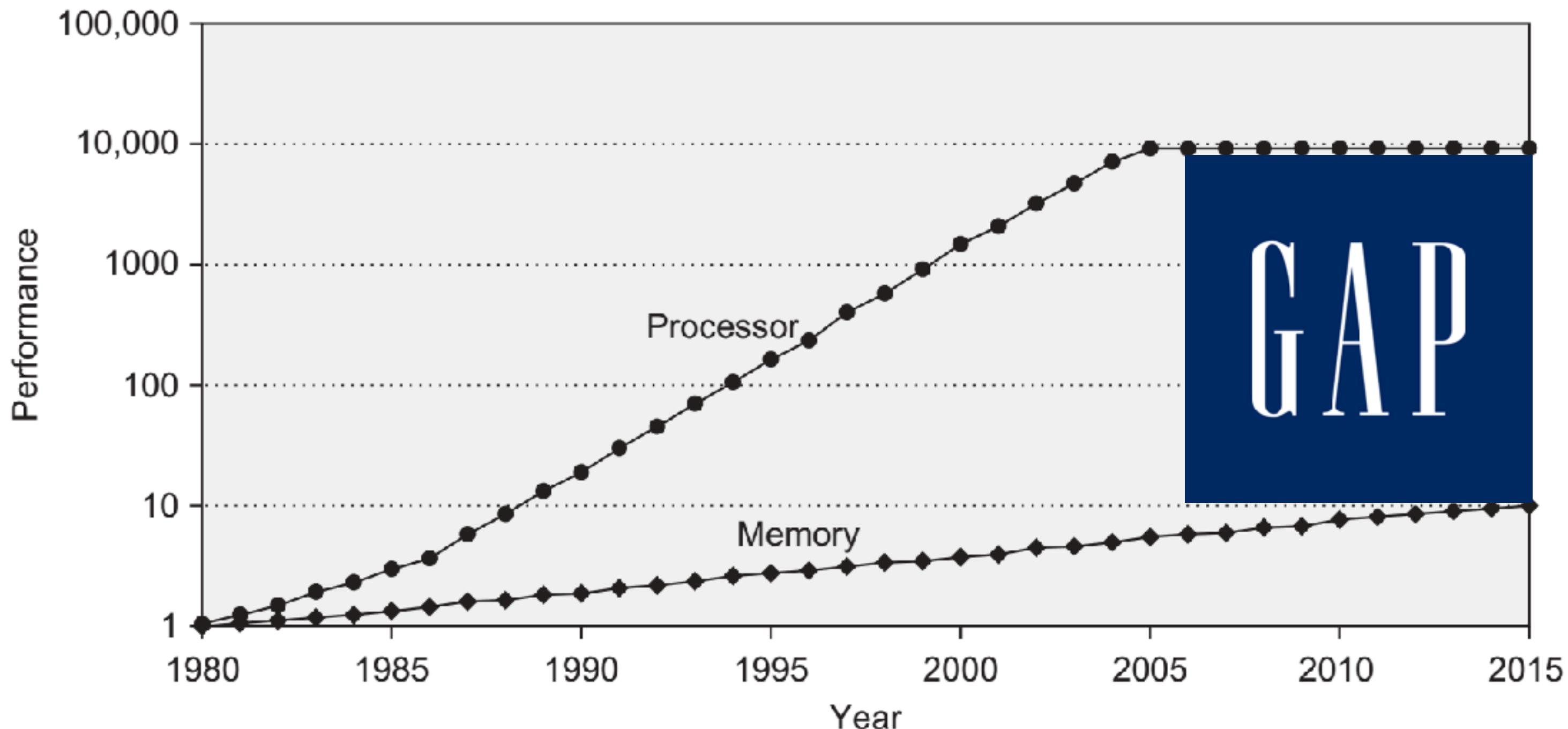
Take-aways: Why CSE142?

- 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!

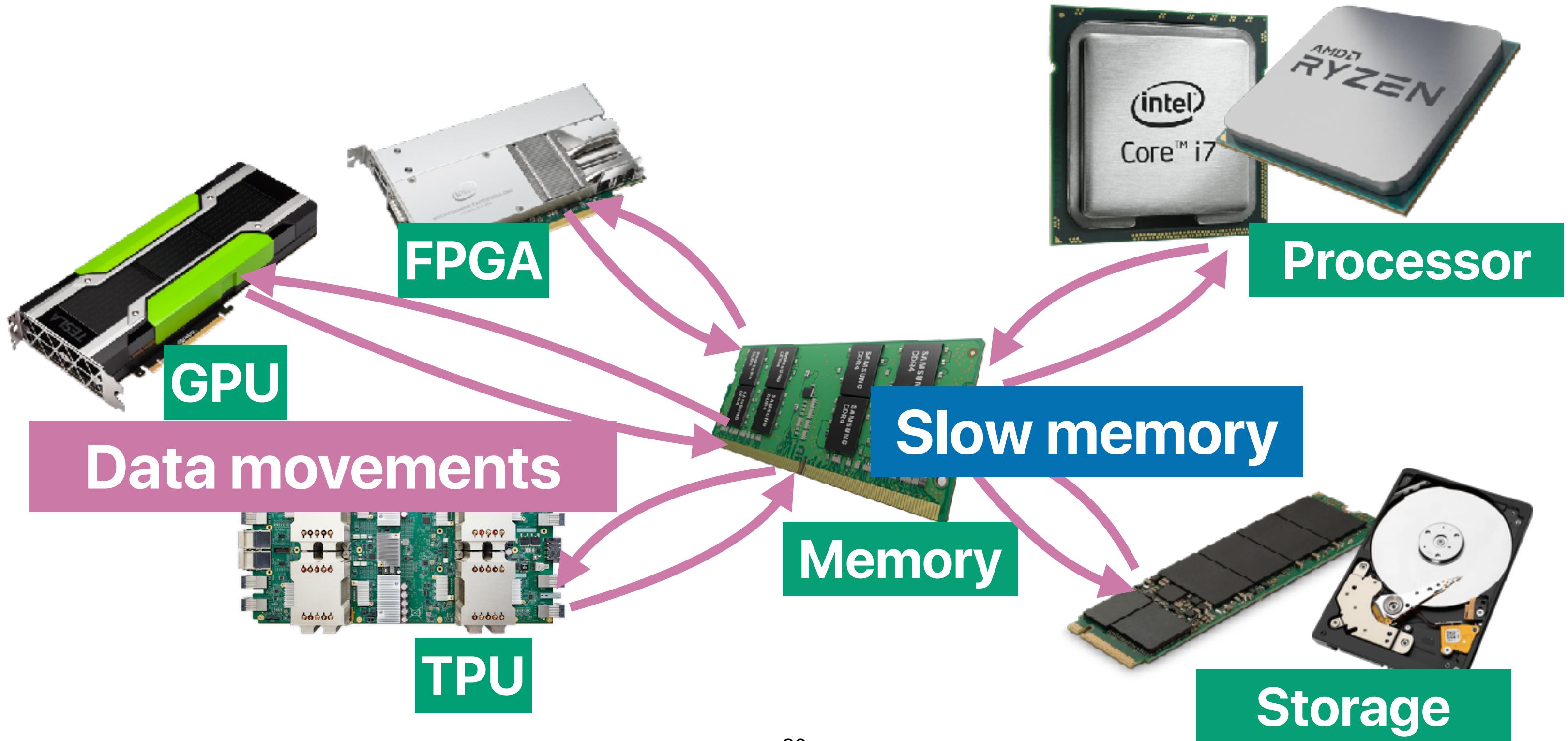
Heterogeneous Computer Architecture



Performance gap between Processor/Memory



Heterogeneous Computer Architecture

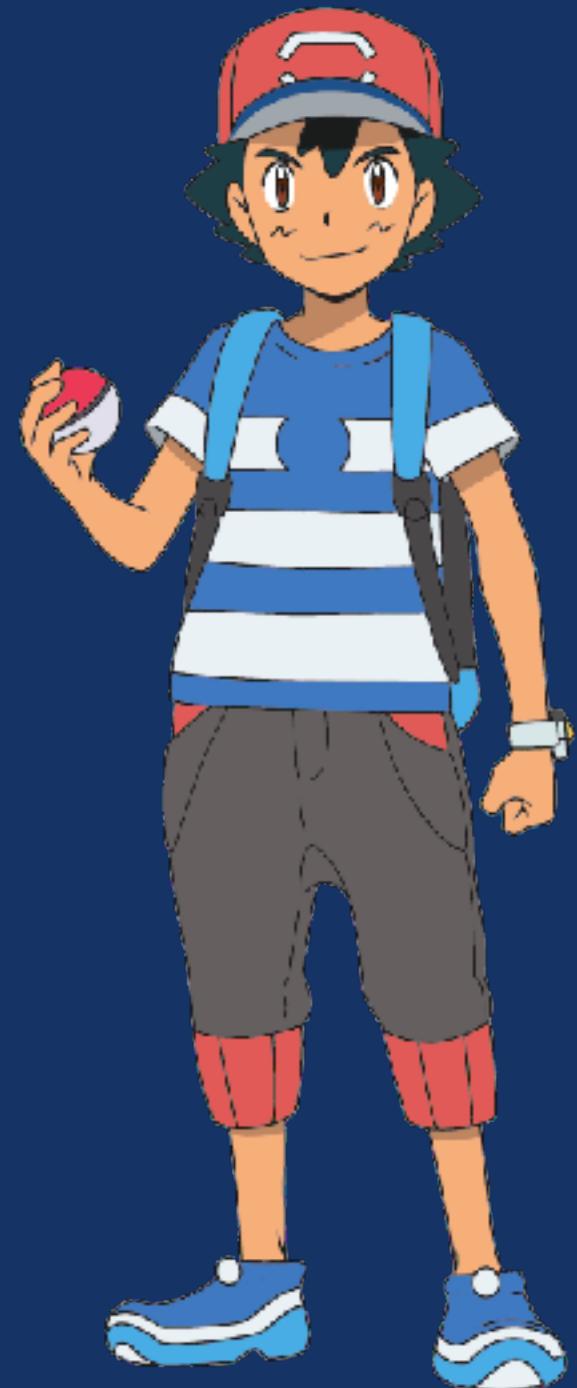


**With GPUs/TPUs taking more roles,
CPU performance does not matter that much in applications**



Take-aways: Why CSE142?

- 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



?????



Thinking about the washlet



Or a Tesla



CSE142'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 optimize programs and fulfill real-world demands

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

Date	Topic	Slides (Release)
8/4/2025	Introduction: the Big Picture of Computers	
8/5/2025	Performance: What affects performance?	
8/6/2025	Performance (II)	
8/7/2025	Performance (III)	
8/11/2025	Memory Hierarchy	
8/12/2025	Memory Hierarchy (II)	
8/13/2025	Memory Hierarchy (III)	
8/14/2025	Memory Hierarchy (IV)	
8/18/2025	Virtual Memory	
8/19/2025	Pipeline Processors	
8/20/2025	Pipelined processors (II)	
8/21/2025	Pipelined processors (III)	
8/25/2025		
8/26/2025	Pipelined processors (IV)	
8/27/2025	Parallelism (I): ILP	
8/28/2025	Parallelism (II): TLP	
9/2/2025	Parallelism (III): CMP	
9/3/2025	Parallelism (IV): SIMD and DSA	
9/4/2025	The Golden Age of Computer Architecture	
9/5/2025	Final Exam	

Performance

Memory

Processor

Parallelism

Learning eXperience

Most lectures today ...



I expect the lecture to be...



Peer instruction

- Before the lecture — You need to complete the required **reading**
- During the lecture — I'll bring in activities to ENGAGE you in exploring your understanding of the material
 - Popup questions
 - Individual **thinking** — use your clicker to express your opinion
 - Group discussion — **discuss** with your surroundings and use your clicker to express your group's opinion
 - Whole-classroom **discussion** — we would like to hear from you

Read

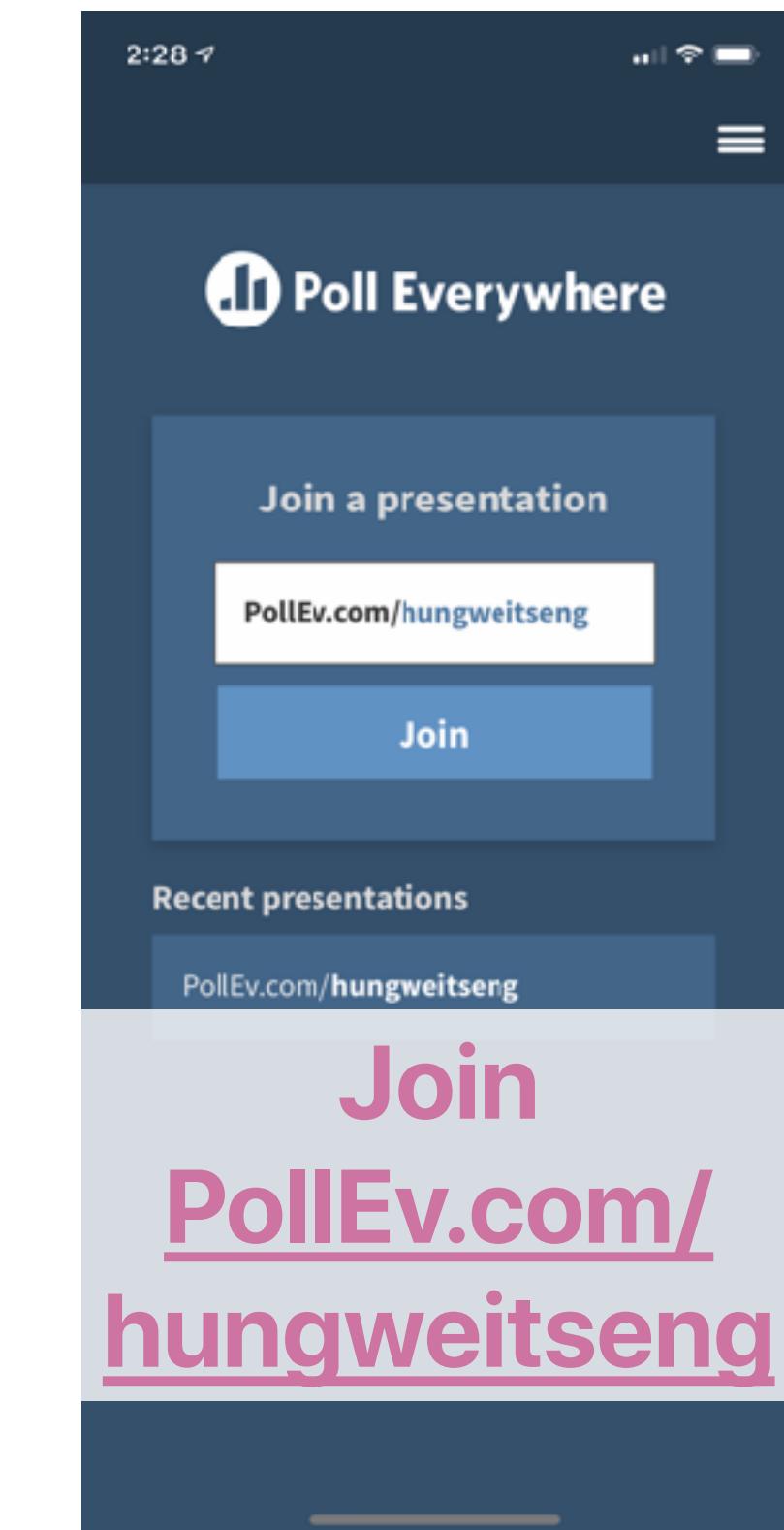
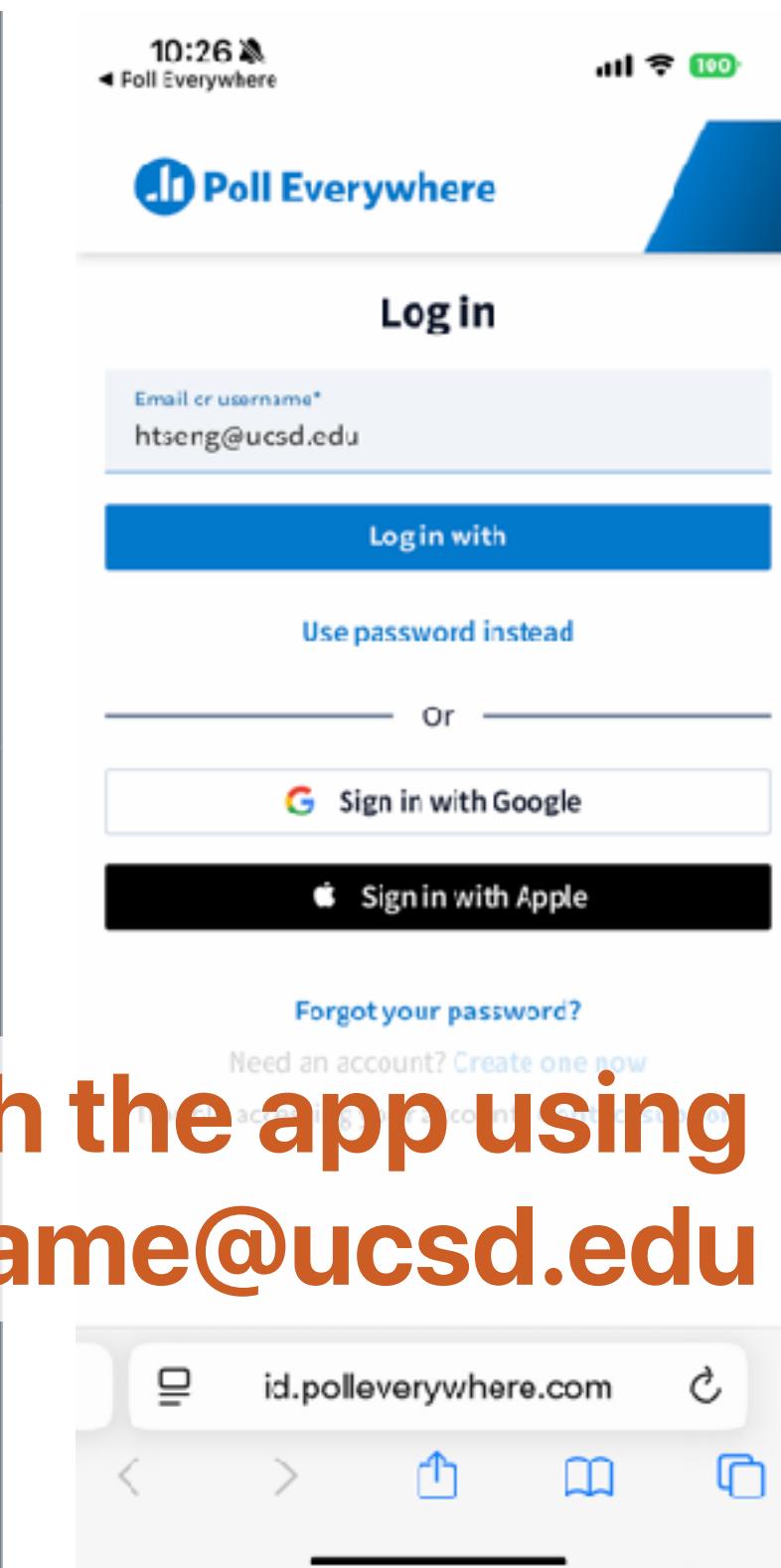
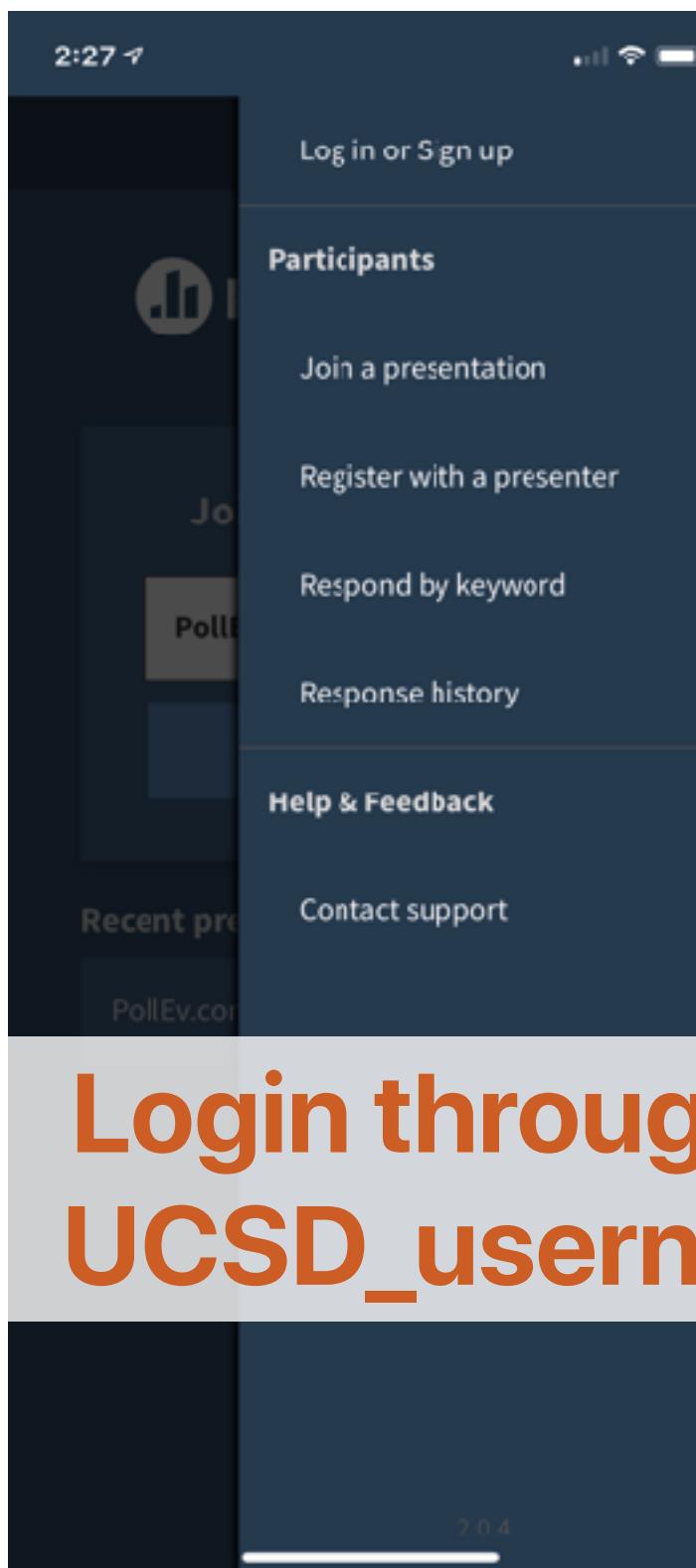
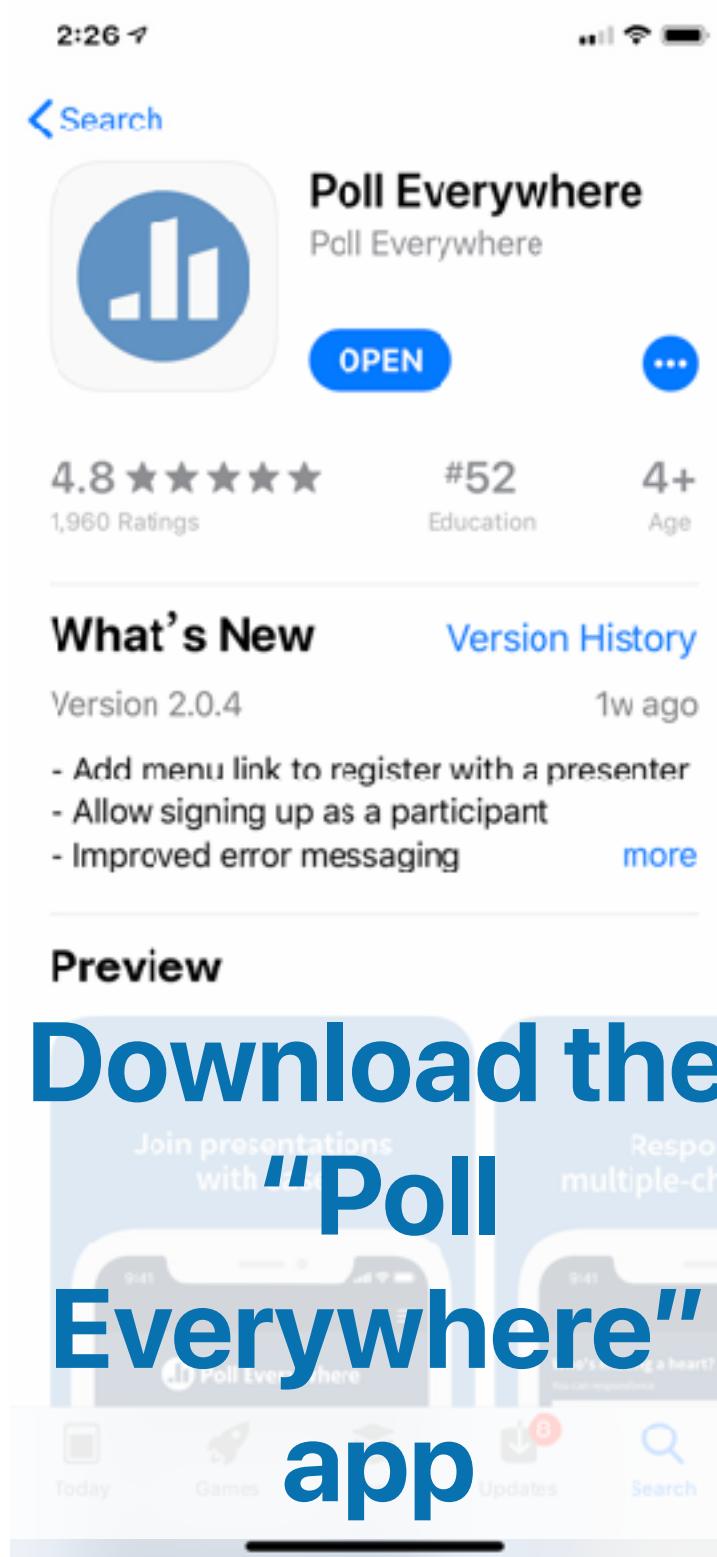
Think

Discuss

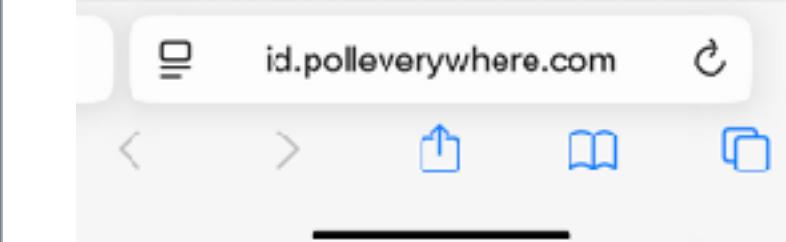
Before lectures: reading quizzes

- We need to prepare you for peer instruction activities and discussions!
- Reading assignments from
 - Randal E. Bryant and David R. O'Hallaron. Computer Systems: A Programmer's Perspective, 3/E
 - David Patterson and John Hennessy. Computer Organization and Design MIPS Edition: The Hardware/Software Interface. 6th Edition
 - Or other assigned materials
- Reading quizzes — 33% of your grades:
 - On Gradescope
 - Due before the lecture, usually twice a week. Check the schedule on our webpage
 - You will have two chances. We take the average
 - No time limitation until the deadline
 - No make up reading quizzes — we will drop probably your lowest two at least

About the time of the Lecture — Setup Poll Everywhere



Login through the app using
UCSD_username@ucsd.edu



Peer instruction

- I'll bring in 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

One set of assignments for two classes

- The best way to prepare the examines
- The same jupyter notebook as CSE142L
- Watch this before your first assignment!
<https://youtu.be/7PTzdq3I4E4>
- Submit through gradescope
 - You may use the escalab.org/datahub service to finish your assignment
 - You have to click the GitHub classroom link to begin with (will post the link of each assignment on the course webpage once released)
 - Complete the questions labeled as “CSE142&CSE142L” and “CSE142 Only”
 - Submit your GitHub repo contains the CSE142/CSE142L assignment lab
 - An autograder will grade your assignment immediately and you can resubmit before the deadline—the earlier you start, the higher chances you can get full credits
- The last CSE142L offering. If it's required for you, you should take this last chance

You'll notice that there are three kinds of questions: "CSE142&CSE142L", "CSE142 Only", and "CSE142L Only".

- **CSE142&CSE142L:** If you're submitting CSE142 assignments, you need to complete all CSE142&CSE142L question. You also need to complete all CSE142 Only question to get full credits.
- **CSE142 Only:** If you're submitting CSE142 assignment/lab, you need to complete all CSE142&CSE142L and CSE142 Only question.
- **CSE142L Only:** If you're submitting CSE142L assignment/lab, you need to complete all CSE142&CSE142L and CSE142L Only question.

Logistics

Grading Breakdown

	In-person session
Reading Quizzes	33.33% Drop lowest 2
Assignments	33.33% Drop lowest 1
Participation	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!
Exams	33.33%. Midterms & Finals using UCSD Testing Center

- 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: htseng @ ucsd.edu
- Visiting Researcher @ Google, 01/2023—03/2023
 - Working for TensorFlow Lite
- PhD in **Computer Science**, University of California, San Diego, 2014
- Research Interests
 - Using ray-tracing accelerators for multiple domains of accelerators
 - General-purpose computing on AI/ML/NN accelerators
 - Intelligent storage devices & near-data processing
 - Or anything else fun
- Fun fact: Hung-Wei was once considering a career path as a singer but went back to academia due to the unsuccessful trial

A collage of three images. The top right image shows a man with glasses sitting at a desk in an office. The middle image is a purple graphic from a personality test showing a cartoon character and the text "Your personality type is: Architect INTJ-A". The bottom right image shows a man singing into a microphone on stage.

Your personality type is:
Architect
INTJ-A

Thought constitutes the greatness of man. Man is a reed, the feeblest thing in nature, but he is a thinking reed.

Blaise Pascal

Teaching Assistant

- Hongyuan Jia
- Office hours:
M 9a-11a @ Zoom
F 9a-11a @ CSE B260A
- E-mail: [cse142_su25 @ escalab.org](mailto:cse142_su25@escalab.org)
- Fun fact: I went meteor watching last week and it seems like we can see meteors until August 12th



Tutor

- Andrew Lu
- Office hours:
Sat 1p-3p @ Zoom
Sun 11a-1p @ Zoom
- E-mail: [cse142_su25 @ escalab.org](mailto:cse142_su25@escalab.org)
- Fun fact: I am a pokemon nerd
(favorite pokemon is Greninja)



Tutor

- Owen Lam
- Office hours:
Th 9a-11a @ CSE B...
Sat 10a-12p @ Zoom
- E-mail: cse142_su25 @ escalab.org
- Fun fact: I am a proud matcha hater



Subscribe to our google calendar!

CSE 142L - Alexander's Office Hours (Remote) 11am – 2pm	CSE 142L Daniel's OH 9am – 2pm B270	CSE 142 - Peeyush's OH 9 – 11am B270	CSE 142L - Alexander's Office Hours (Remote) 9am – 12pm	CSE 142 - Peeyush's OH 9 – 11am B270A	CSE142 - Fucheng's OH 10am – 12pm B215
CSE142 (Lecture) 2 – 3:30pm	CSE142 (Lecture) 2 – 3:30pm	CSE142 (Lecture) 2 – 3:30pm	CSE142 - JC's OH 12 – 2pm	CSE 142L - Honghao OH 11am – 1pm https://ucsd.zoom.us/j/91799877409	CSE142L - Honghao OH 12 – 5pm https://ucsd.zoom.us/j/99877409
CSE 142 - JC's OH (Virtual) 3:30 – 5pm	CSE 142 - JC's OH (Virtual) 3:30 – 5pm	CSE142/L Office Hours (Hung-Wei 3:30pm, Computer Science and En	CSE142/L Office Hours (Hung-Wei 3:30pm, Computer Science and En	CSE142 (Lecture) 2 – 3:30pm	CSE142L - Yinong's OH 12 – 5pm CSE basement B250A
CSE142L (Lecture) 5 – 6pm	CSE 142 - Will's OH 5 – 8pm	CSE 142 - Will's OH 5 – 8pm	CSE 142 OH - Peeyush's OH 5pm, https://ucsd.zoom.us/j/91799877409	CSE142L (Lecture) 5 – 6pm	CSE142L - Honghao OH 7 – 9pm https://ucsd.zoom.us/j/91799877409
CSE 142 - Will's OH B250A 6 – 8pm					

Please be aware that TA/tutors are unionized and under contract. They are not supposed to work outside normal hours

Discussion sessions

- Fridays 12p-2p WLH 2204
- Covers hints for your assignments
- Highly recommended

CSE142: Computer Architecture: S Summer Session II)

The website

- Calendar
- Schedule
- Slides
 - Preview — for the ease of note taking
 - Release — the actual slides



Online: Zoom or <https://www.youtube.com/profusagi>

Lecture: MTuWTh 2:00p – 3:20p @ WLH 2205 or Zoom

Schedule and Slides Assignments

Hung-Wei Tseng

今天 2023年8月

週日	週一	週二	週三	週四	週五	週六
30	31	8月 1日	2	3	4	
6	7	8	9	10	11	
上午9:30 CSE 142 Ch 上午10點 Andrei office 上午10點 Andrei office 上午9:30 CSE 142 Ch 上午10點 Andrei office 上午10點 CSE142 Lec 下午2點 CSE142 Lect 上午11:30 CSE 142L F 上午10點 CSE142 Lec 下午1點 CSE 142L Hc 上午11:30 CSE 142L F 下午3:30 Hung-Wei's 下午2點 CSE142 Lect 下午2點 CSE142 Lect 下午5點 Andrei office 下午2點 CSE142 Lect 下午5點 CSE142 Lec 下午3:30 Hung-Wei's 下午4點 CSE142 Disc 下午8:30 CSE 142L EJ 下午5點 CSE142L Eij +2 更多						
13	14	15	16	17	18	
上午9:30 CSE 142 Ch 上午10點 Andrei office 上午10點 Andrei office 上午9:30 CSE 142 Ch 上午10點 Andrei office 上午10點 CSE142 Lec 下午2點 CSE142 Lect 上午11:30 CSE 142L F 上午10點 CSE142 Lec 下午1點 CSE 142L Hc 上午11:30 CSE 142L F 下午3:30 Hung-Wei's 下午2點 CSE142 Lect 下午2點 CSE142 Lect 下午5點 Andrei office 下午2點 CSE142 Lect 下午5點 CSE142 Lec 下午3:30 Hung-Wei's 下午4點 CSE142 Disc 下午8:30 CSE 142L EJ 下午5點 CSE142L Eij +2 更多						
20	21	22	23	24	25	
上午9:30 CSE 142 Ch 上午10點 Andrei office 上午10點 Andrei office 上午9:30 CSE 142 Ch 上午10點 Andrei office 上午10點 CSE142 Lec 下午2點 CSE142 Lect 上午11:30 CSE 142L F 上午10點 CSE142 Lec 下午1點 CSE 142L Hc 上午11:30 CSE 142L F 下午3:30 Hung-Wei's 下午2點 CSE142 Lect 下午2點 CSE142 Lect 下午5點 Andrei office 下午2點 CSE142 Lect 下午5點 CSE142 Lec 下午3:30 Hung-Wei's 下午4點 CSE142 Disc 下午8:30 CSE 142L EJ 下午5點 CSE142L Eij +2 更多						
27	28	29	30	31	9月 1日	
上午9:30 CSE 142 Ch 上午10點 Andrei office 上午10點 Andrei office 上午9:30 CSE 142 Ch 上午10點 Andrei office 上午10點 CSE142 Lec 下午2點 CSE142 Lect 上午11:30 CSE 142L F 上午10點 CSE142 Lec 下午1點 CSE 142L Hc 上午11:30 CSE 142L F 下午3:30 Hung-Wei's 下午2點 CSE142 Lect 下午2點 CSE142 Lect 下午5點 Andrei office 下午2點 CSE142 Lect 下午5點 CSE142 Lec 下午3:30 Hung-Wei's 下午4點 CSE142 Disc 下午8:30 CSE 142L EJ 下午5點 CSE142L Eij +2 更多						

活動顯示的時區：太平洋時間 - 洛杉磯

+ Google 日曆

This is summer session — everything is packed at 2x!

The image is a composite of three screens. The top screen shows a video player with the title "Now playing — Never say never (Justin Bieber)". The middle screen shows a Jupyter Notebook interface with code for creating a sine wave plot and the resulting plot. The bottom screen shows a playback speed control slider set to 2x.

Now playing — Never say never (Justin Bieber)

In [15]: # Enable interactive plot
%matplotlib notebook
%matplotlib notebook

In [16]: import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation

In [17]: fig, ax = plt.subplots()
line, = ax.plot([])
ax.set_xlim(0, 2*np.pi)
ax.set_ylim(-1.1, 1.1)

def animate(frame_num):
 x = np.sin(x + 2*np.pi * frame_num/100)
 line.set_data(x, y)
 return line

anim = FuncAnimation(fig, animate, frames=100, interval=20)
plt.show()

Figure 1

0.25
0.5
0.75
Normal
1.25
1.5
1.75

2

KEEP CALM AND ACE YOUR FINAL EXAMS

• 6/14/1471

• Cumul through

• No cut

• No

• 2 ques exam

An attention-grab
1. Starting with a
turnsp.
2. Starting with a fo
Your opinion of the book.
1. I enjoyed this book.
2. I did not like this book.
3. I would recommend this bo
A closing statement

nt 3

|| ▶ 🔍 0:14 / 1:28:59

Summary of course resources

- Lectures:
 - In-person @ **WLH 2204**
 - Repository on **Youtube**: <https://www.youtube.com/profusagi>
 - We do podcasts
- Schedule, slides, grades on **course webpage**:
<https://www.escalab.org/classes/cse142-2025su/>
- Discussion on **piazza**:
<https://piazza.com/class/md99z1w7qks3cb>
- Reading quizzes and assignments through **gradescope**:
<https://www.gradescope.com/courses/1068150>
- Working environment on <https://escalab.org/datahub>
- Office Hours & Locations
https://calendar.google.com/calendar/embed?src=c_373ea7ba1adb25dcb44c3a3d1cb62af934f7601955381cdc89116d91596ba4af%40group.calendar.google.com



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 UCSD 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 difficulty coming to the 50% of class, you can still earn all credits if you ace all assignments.
- 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 I need to take CSE142/L together?
 - Strongly encouraged
 - We have a combined CSE142/L assignments & lab reports this time. You just need to do a few more questions for each assignment and write two programming assignments this summer (the first PA is simply a toy one) and you can get CSE142L done.
- Do we have a canvas page?
 - No — 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/datahub
 - Please email htseng@ucsd.edu 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
- Do we have Zoom office hours
 - Yes, but only to our best efforts — I cannot force tutors/TA to do Zoom office hours
 - Please use piazza to ask questions!
- 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

Announcements

- Login to piazza, datahub, gradescope
 - Let us know if you have any issue
cse142_su25@escalab.org
- Reading quiz on Gradescope due **tomorrow before the lecture**
- Assignment #1 is released — working on datahub!
 - Watch this video before you start! <https://youtu.be/7PTzdq3I4E4>
- Check our website
 - Check the lab website if you have sometime as well!

Computer Science & Engineering

142

つづく

