CSCE 212: Introduction to Computer Architecture

Lecture 1: Introduction and Basics

Prof. Pooyan Jamshidi

University of South Carolina

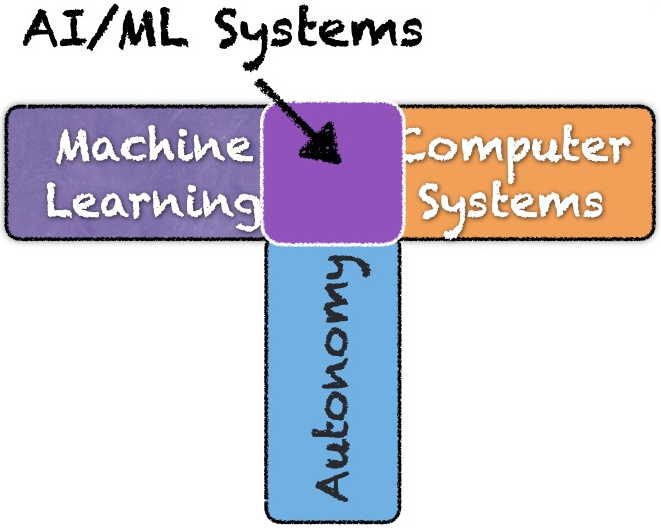
[These slides are mostly based on those of Onur Mutlu for the Computer Architecture Course at CMU]

Brief Self Introduction

- Pooyan Jamshidi
  - Assistant Professor @ UofSC CSE, since August 2018
  - Postdoc 2 @ Carnegie Mellon University
  - Postdoc 1 @ Imperial College London
  - PhD from Dublin City University
  - worked in software industry for 10 years (pre-PhD)
  - https://pooyanjamshidi.github.io/
- Research and Teaching in:
  - Machine Learning Systems = AI/ML + Computer Systems
  - Autonomous Robots = AI/ML + Robotics
  - Causal AI = Causal Inference, Causal Representation Learning
  - Neural Architectrures + Hardware Accelerators
  - Systems for ML (CSCE 585)
  - Autonomous and Adaptive Systems (NASA Autonomous Space Lander)
  - Causal Inference and Transfer Learning (ML Theory)
  - Adversarial Machine Learning (ATHENA, a defense framework for ML Systems)

Artificial Intelligence and Systems Laboratory (AISys Lab)

https://pooyanjamshidi.github.io/AISys/

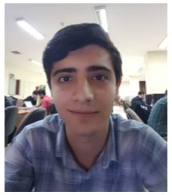



Ying Meng (PhD student)


Shuge Lei (PhD student)


Kimia Noorbakhsh (Undergrad)


Fatemeh Ghofrani (PhD student)


Hamed Damirchi (PhD student)


Shahriar Iqbal (PhD student)


Jianhai Su (PhD student)


M.A. Javidian (postdoc)


Abir Hossen (PhD student)


Mahdi Sharifi (PhD student)

Sponsors, thanks!


Lane Stanley (Intern)

x

Interested in AI/ML/Robotics Research?

- Am always looking for highly motivated students.
- Underrepresented in STEM/CS is a big plus for joining AISys

Course Information

- Course Website (under construction): https://pooyanjamshidi.github.io/csce212/
- Piazza (Communications): http://piazza.com/sc/spring2022/csce212
  - Discussion boards for each assignment and the course overall
    - PLEASE post questions on course material (don't be shy)
    - Answer others' questions - if you know the answer ;-)
    - Learn from others' questions and answers
    - Check it Often
- GradeScope (Assignments): https://www.gradescope.com/courses/354485
- I already added you to Piazza and GradeScope!

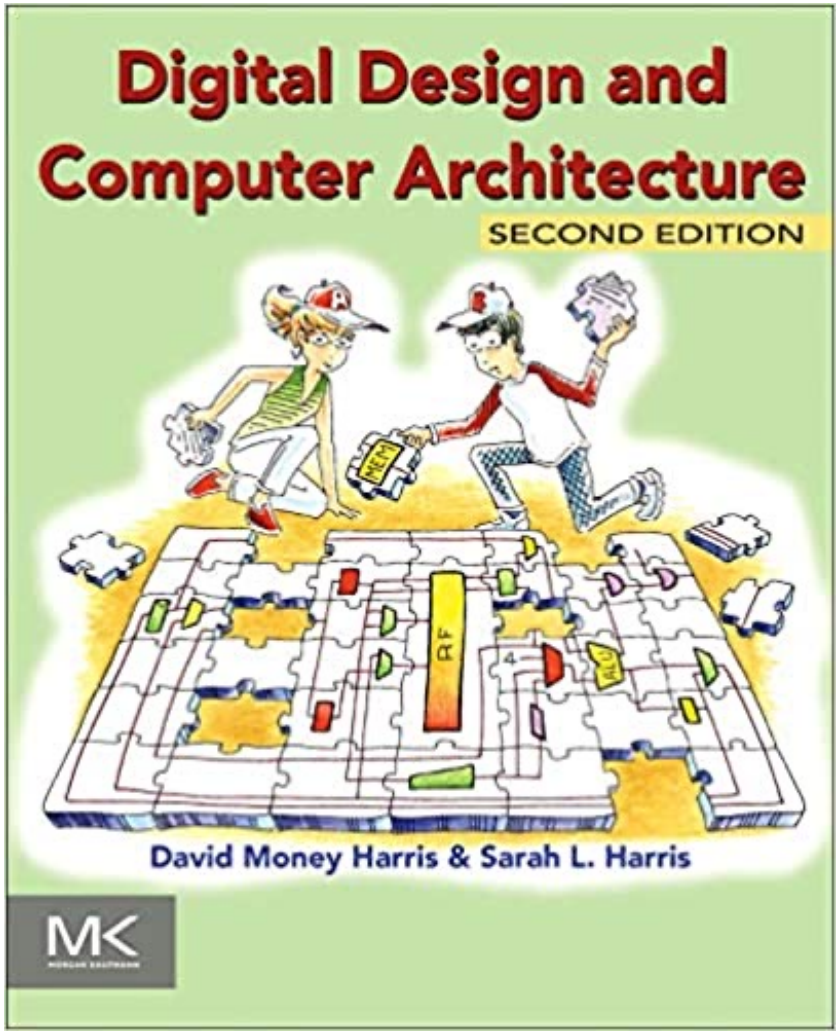
Course Info: TA and Communication

- Teaching Assistant
  - Ziyu Zhao
  - ZIYUZ@email.sc.edu
- Reach all of us at
  - Piazza
  - Please do not email to us unless there is an absolute necessity , e.g., you have a private question that cannot be shared with others, but again think twice ☺
  - Instead use Piazza, so others can benefit from your questions
  - Build a community and learn from each other via Piazza

Lectures

- Tuesdays and Thursdays, 4:25-5:40
- Zoom
- Will be recorded
- Attendance is for your benefit and is therefore important
- Please turn on your webcam if you have one in your computer

Textbook (Harris & Harris)



Highly recommend it, but not required, we will cover the materials that you will be responsible for…

- Chapter 6 (Architecture)
- Chapter 7 (Microarchitecture)
- Chapter 8 (Memory?)

Evaluation (subject to minor changes)

Quizzes	20%
Projects	30%
Midterm	25%
Final	25%

Key Research Directions in Computer Architecture at AISys

- Fundamentally Low-Latency and Predictable Neural Architectures
  - Neural Architecture Search
  - Hardware Accelerators
- Domain-Specific Architectures, e.g., Architectures for AI/ML
- Fundamentally Energy-Efficient Architectures
  - In-memory
  - Near-memory

What Will We Learn in This Course?

Answer

How Computers Work

(from the ground up)

Answer Continued

And Why We Care

Why Do We Have Computers?

Why Do We Do Computing?

Answer

To Solve Problems

Answer Reworded

To Gain Insight

Hamming, "Numerical Methods for Scientists and Engineers," 1962.

Answer Extended

To Enable
a Better Life & Future

How Does a Computer Solve Problems?

Answer

Orchestrating Electrons

In today's dominant technologies

How Do Problems Get Solved by Electrons?

So, I Hope You Are Here for This

CSCE 145/206

"C" as a model of computation

Programmer's view of how a computer system works

- How does an assembly program end up executing as digital logic?
- What happens in-between?
- How is a computer designed using logic gates and wires to satisfy specific goals?

Architect/microarchitect's view: How to design a computer that meets system design goals.

Choices critically affect both the SW programmer and the HW designer

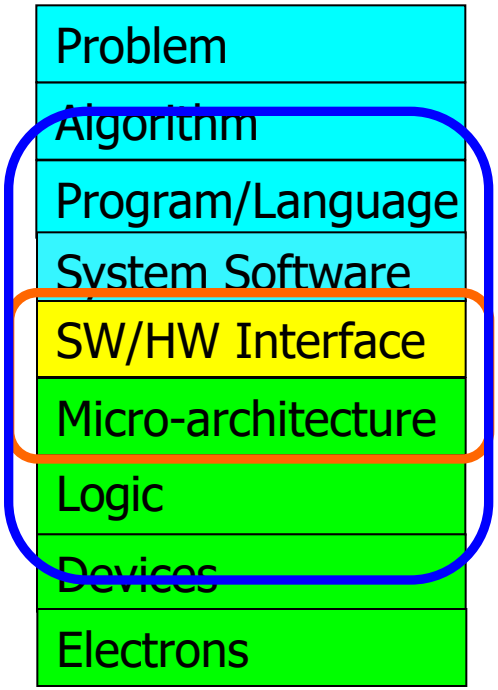
CSCE 211

HW designer's view of how a computer system works

Digital logic as a model of computation

The Transformation Hierarchy

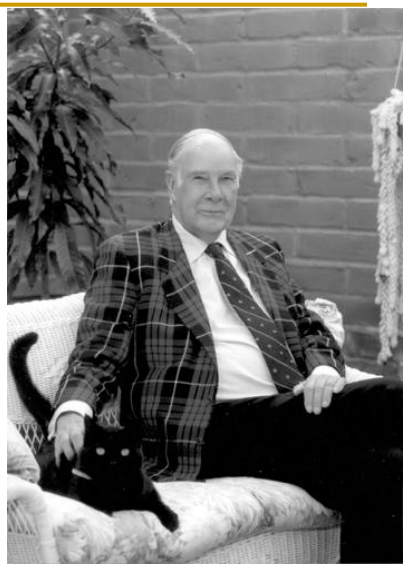
Computer Architecture (expanded view)



Computer Architecture (narrow view)

Levels of Transformation

"The purpose of computing is [to gain] insight" (*Richard Hamming*)
We gain and generate insight by solving problems
How do we ensure problems are solved by electrons?



Problem
Algorithm
Program/Language
Runtime System (VM, OS, MM)
ISA (Architecture)
Microarchitecture
Logic
Devices
Electrons

Algorithm

Step-by-step procedure that is **guaranteed to terminate** where **each step is precisely stated** and **can be carried out by a computer**

- **Finiteness**
- **Definiteness**
- **Effective computability**

Many algorithms for the same problem

ISA (Instruction Set Architecture)

Interface/contract between SW and HW.

What the programmer assumes hardware will satisfy.

Microarchitecture

An implementation of the ISA

Digital logic circuits

Building blocks of micro-arch (e.g., gates)

Computer Architecture

- is the science and art of designing computing platforms (hardware, interface, system SW, and programming model)

- to achieve a set of design goals
  - E.g., highest performance on earth on workloads X, Y, Z
  - E.g., longest battery life at a form factor that fits in your pocket with cost < $$$ CHF
  - E.g., best average performance across all known workloads at the best performance/cost ratio
  - …

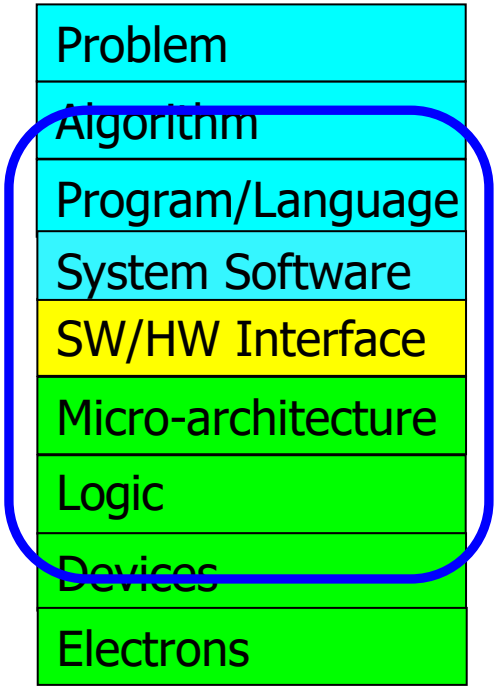
  - Designing a supercomputer is different from designing a smartphone → But, many fundamental principles are similar

Axiom

To achieve the highest energy efficiency and performance:

we must take the expanded view

of computer architecture



Co-design across the hierarchy: Algorithms to devices

Specialize as much as possible within the design goals

Different Platforms, Different Goals

Source: http://www.sia-online.org (semiconductor industry association)

Different Platforms, Different Goals

Source: https://iq.intel.com/5-awesome-uses-for-drone-technology/

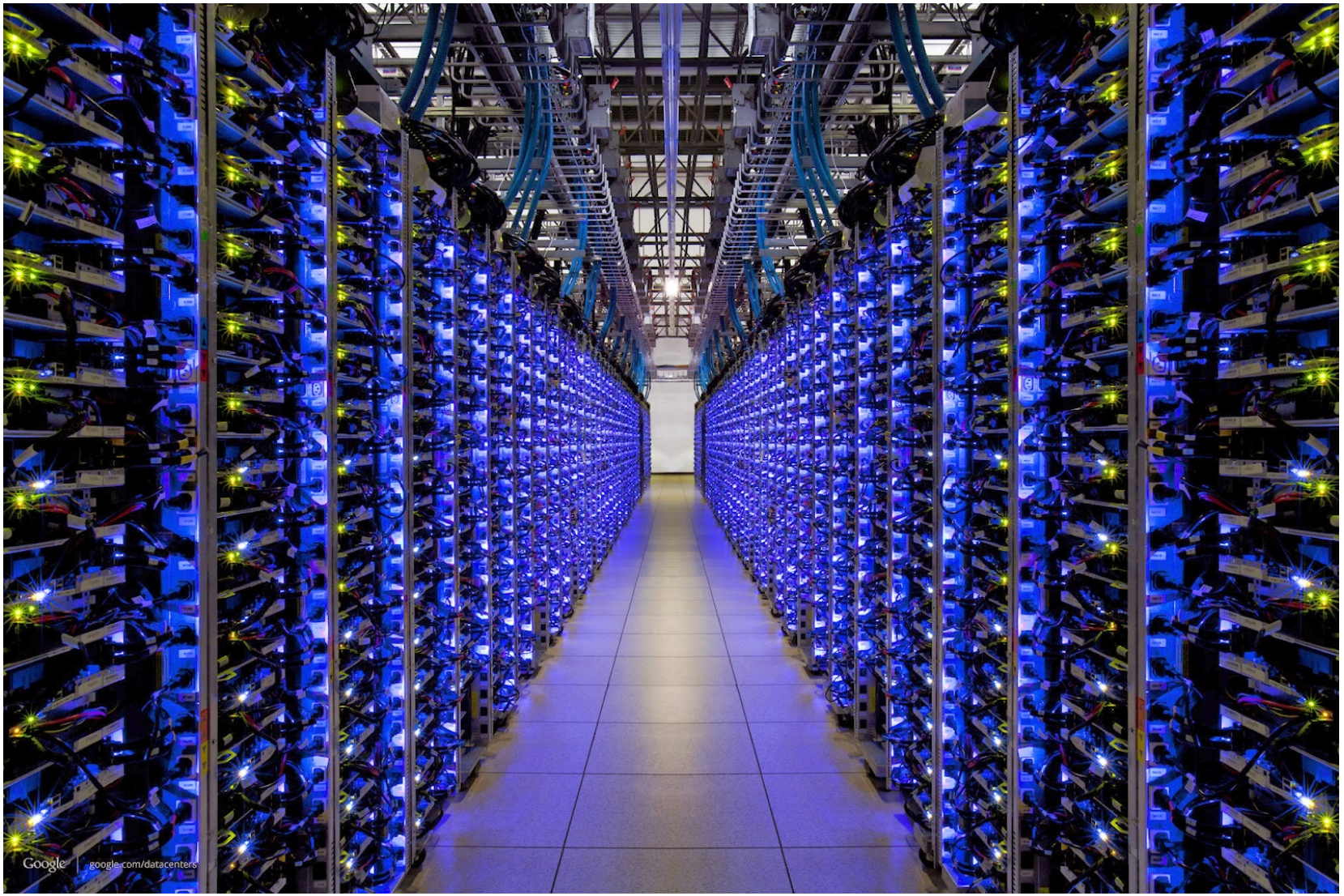
Different Platforms, Different Goals

Source: https://taxistartup.com/wp-content/uploads/2015/03/UK-Self-Driving-Cars.jpg

Different Platforms, Different Goals

Source: http://sm.pcmag.com/pcmag_uk/photo/g/google-self-driving-car-the-guts/google-self-driving-car-the-guts_dwx8.jpg

Different Platforms, Different Goals



Source: http://datacentervoice.com/wp-content/uploads/2015/10/data-center.jpg

Different Platforms, Different Goals

Source: https://fossbytes.com/wp-content/uploads/2015/06/Supercomputer-TIANHE2-china.jpg

Different Platforms, Different Goals

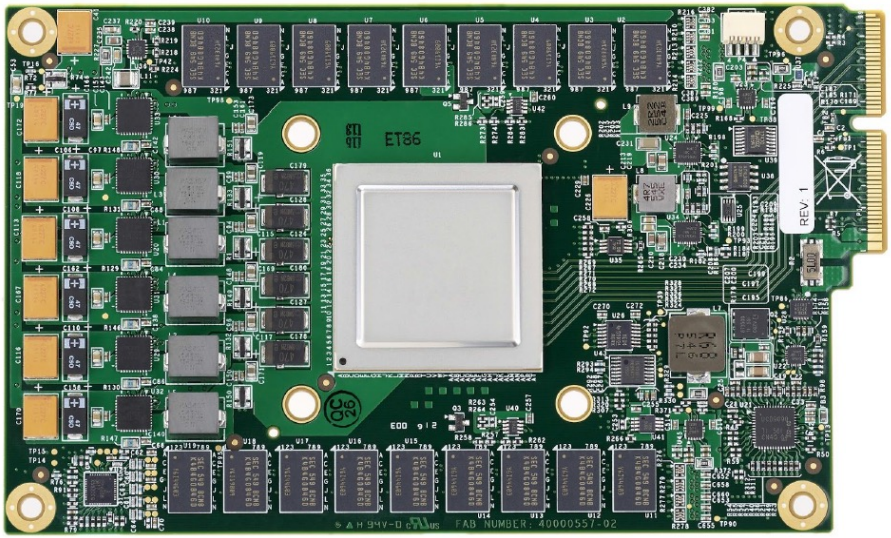


Figure 3. TPU Printed Circuit Board. It can be inserted in the slot for an SATA disk in a server, but the card uses PCIe Gen3 x16.

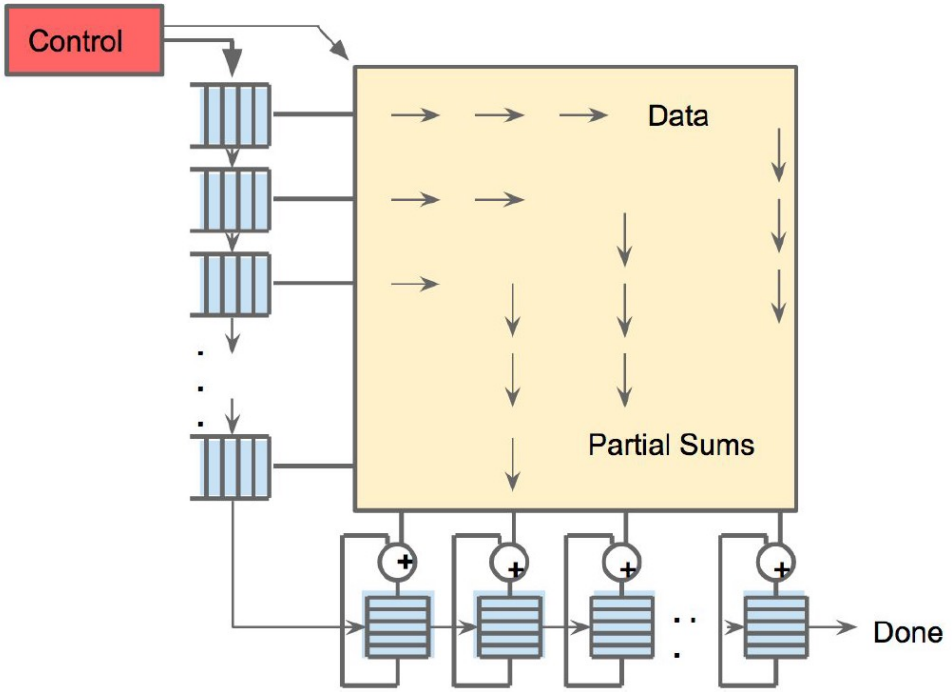
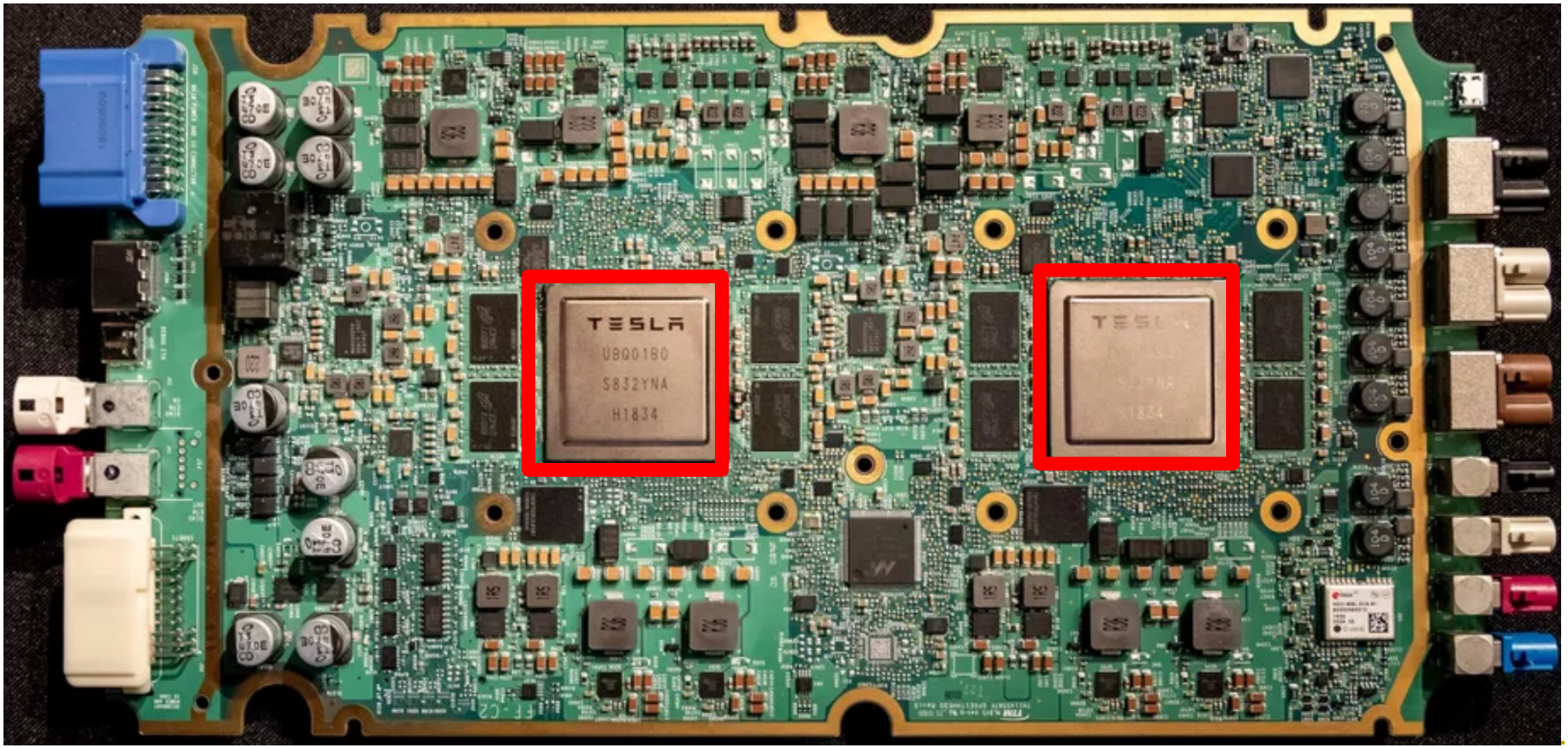


Figure 4. Systolic data flow of the Matrix Multiply Unit. Software has the illusion that each 256B input is read at once, and they instantly update one location of each of 256 accumulator RAMs.

Jouppi et al., "In-Datacenter Performance Analysis of a Tensor Processing Unit", ISCA 2017.

Different Platforms, Different Goals

- ML accelerator: 260 mm^2, 6 billion transistors, 600 GFLOPS GPU, 12 ARM 2.2 GHz CPUs.
- Two redundant chips for better safety.

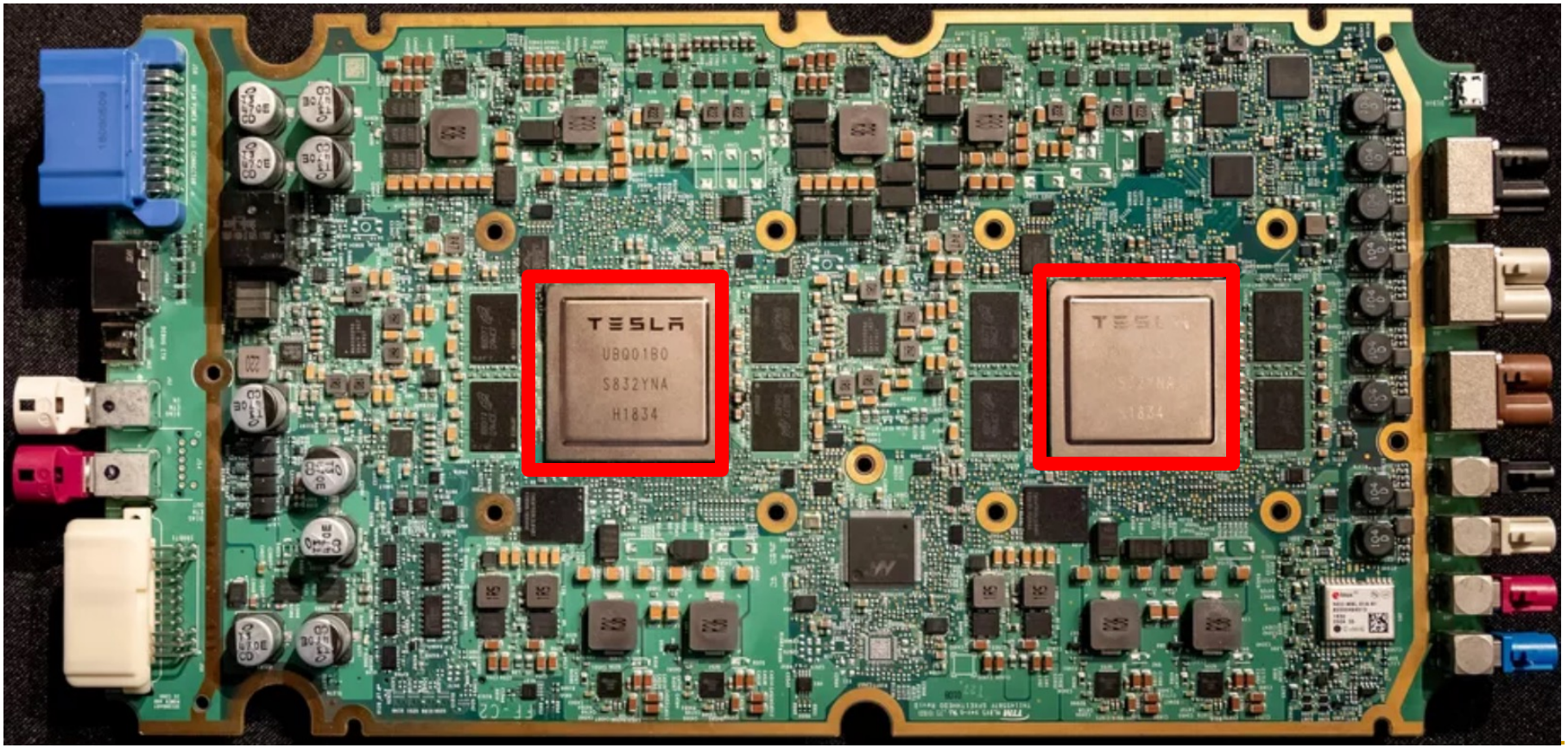


https://youtu.be/Ucp0TTmvqOE?t=4236

Many Interesting Things Are Happening Today in Computer Architecture

TESLA Full Self-Driving Computer (2019)

- ML accelerator: 260 mm^2, 6 billion transistors, 600 GFLOPS GPU, 12 ARM 2.2 GHz CPUs.
- Two redundant chips for better safety.



https://youtu.be/Ucp0TTmvqOE?t=4236

Google TPU Generation I (~2016)

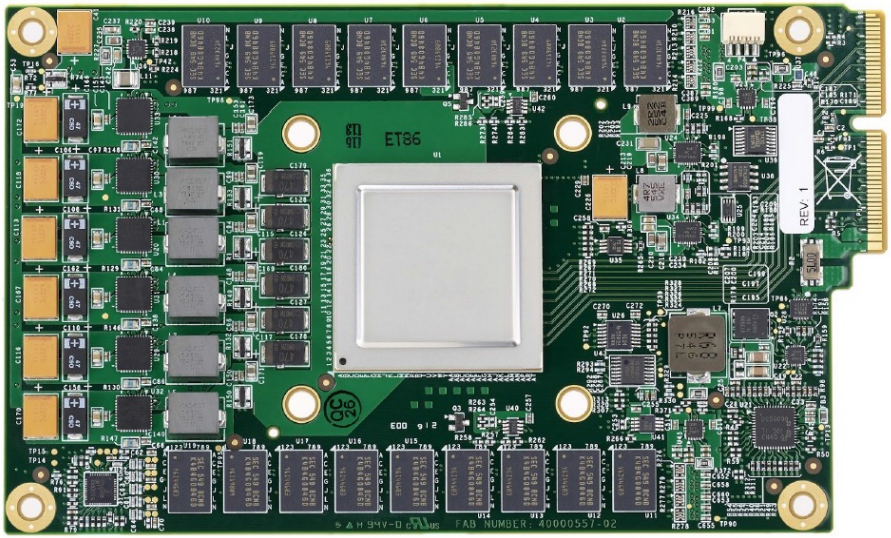


Figure 3. TPU Printed Circuit Board. It can be inserted in the slot for an SATA disk in a server, but the card uses PCIe Gen3 x16.

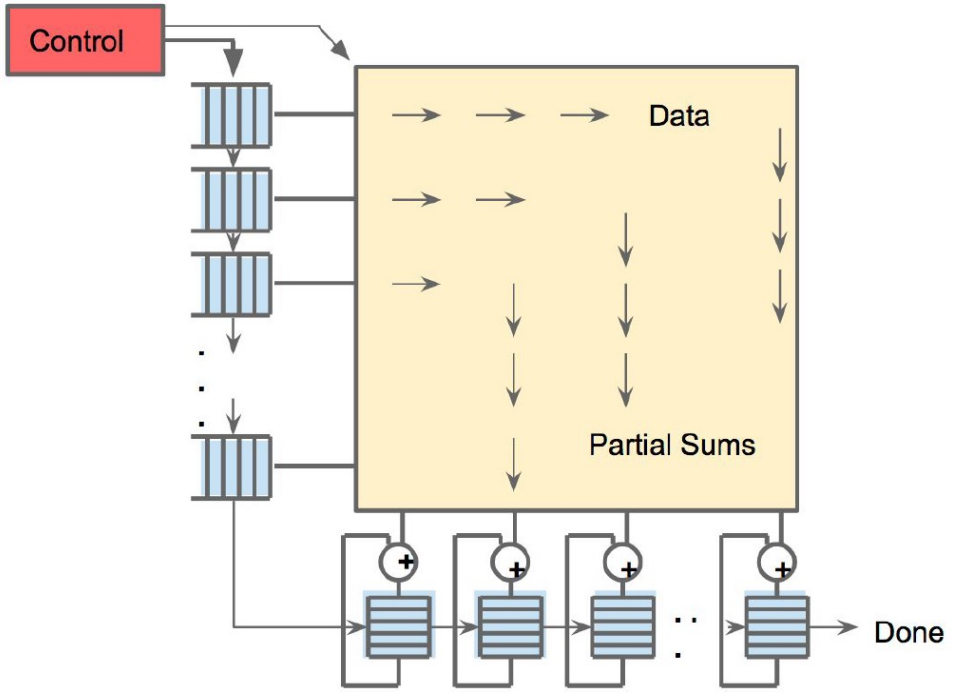
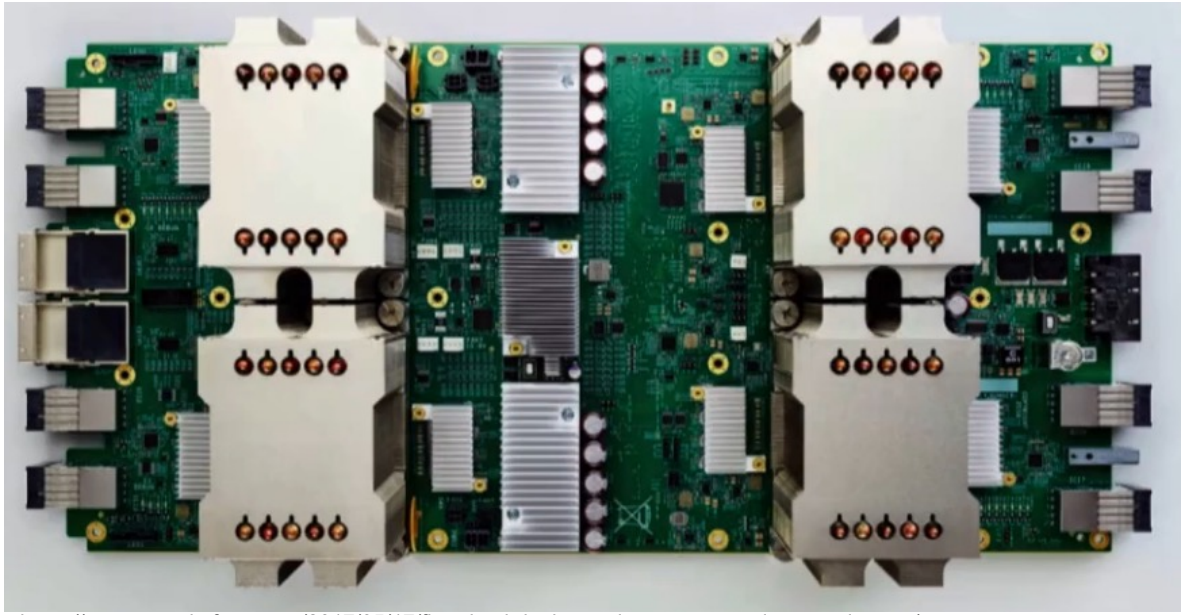


Figure 4. Systolic data flow of the Matrix Multiply Unit. Software has the illusion that each 256B input is read at once, and they instantly update one location of each of 256 accumulator RAMs.

Jouppi et al., "In-Datacenter Performance Analysis of a Tensor Processing Unit", ISCA 2017.

Google TPU Generation II (2017)



https://www.nextplatform.com/2017/05/17/first-depth-look-googles-new-second-generation-tpu/

4 TPU chips
vs 1 chip in TPU1

High Bandwidth Memory
vs DDR3

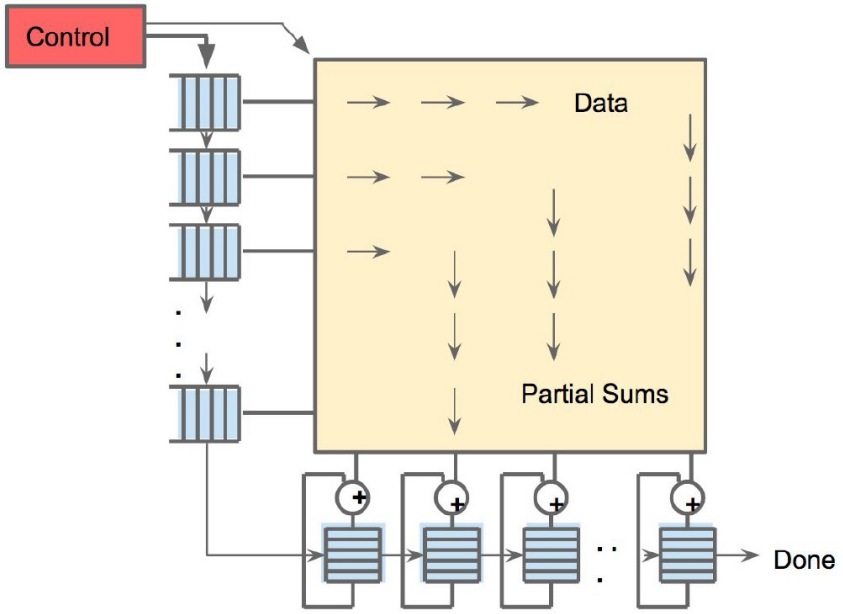
Floating point operations
vs FP16

45 TFLOPS per chip
vs 23 TOPS

Designed for training
and inference
vs only inference

An Example Modern Systolic Array: TPU (II)

As reading a large SRAM uses much more power than arithmetic, the matrix unit uses systolic execution to save energy by reducing reads and writes of the Unified Buffer [Kun80][Ram91][Ovt15b]. Figure 4 shows that data flows in from the left, and the weights are loaded from the top. A given 256-element multiply-accumulate operation moves through the matrix as a diagonal wavefront. The weights are preloaded, and take effect with the advancing wave alongside the first data of a new block. Control and data are pipelined to give the illusion that the 256 inputs are read at once, and that they instantly update one location of each of 256 accumulators. From a correctness perspective, software is unaware of the systolic nature of the matrix unit, but for performance, it does worry about the latency of the unit.



Jouppi et al., "In-Datacenter Performance Analysis of a Tensor Processing Unit", ISCA 2017.

An Example Modern Systolic Array: TPU (III)

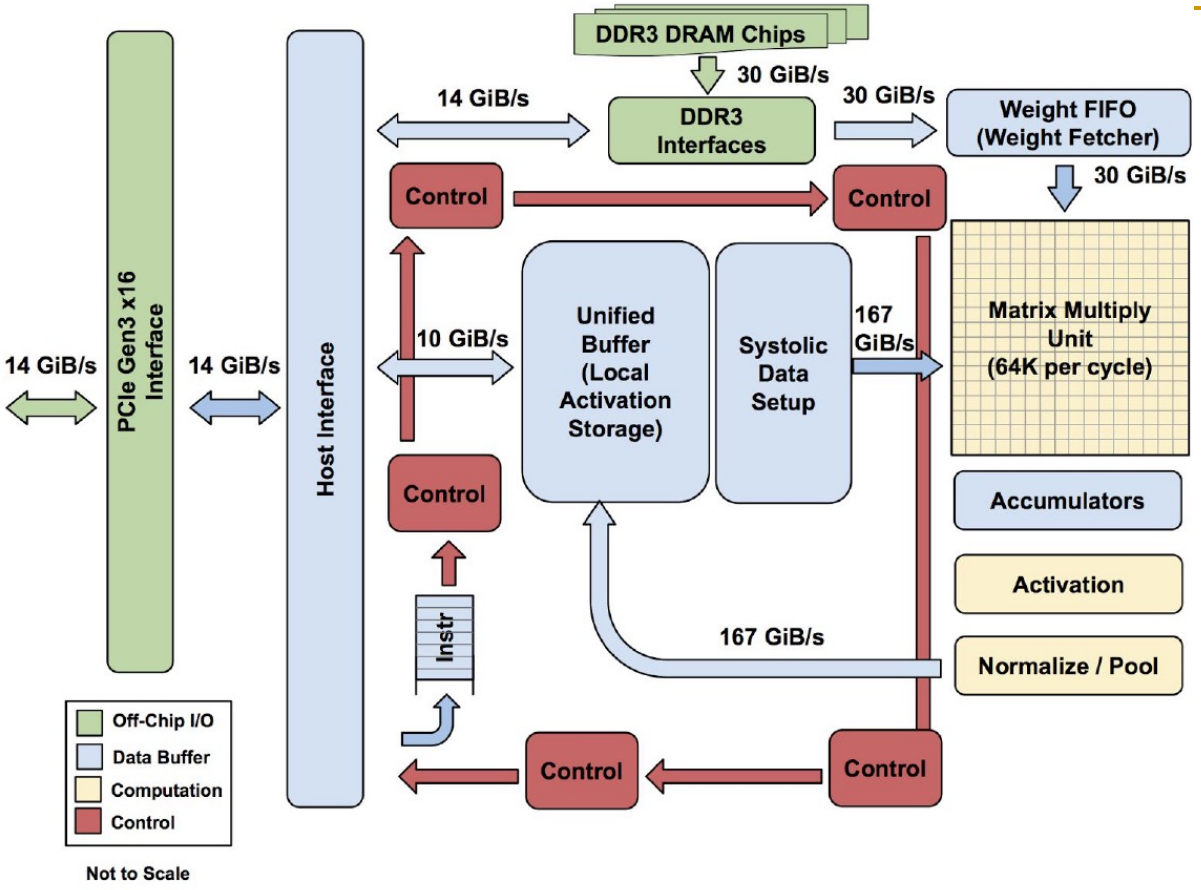


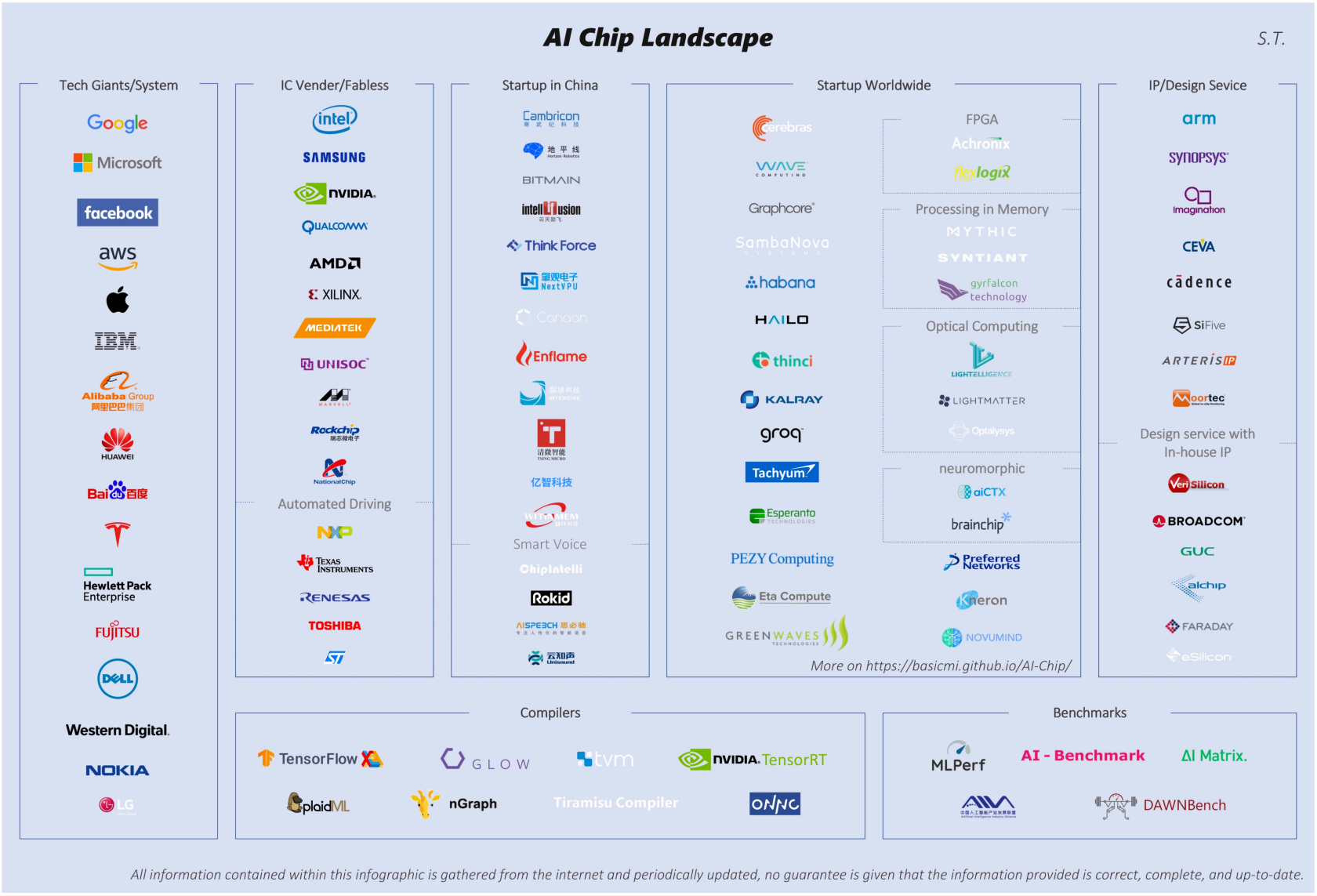
Figure 1. TPU Block Diagram. The main computation part is the yellow Matrix Multiply unit in the upper right hand corner. Its inputs are the blue Weight FIFO and the blue Unified Buffer (UB) and its output is the blue Accumulators (Acc). The yellow Activation Unit performs the nonlinear functions on the Acc, which go to the UB.

Many (Other) AI/ML Chips

- Alibaba
- Amazon
- Facebook
- Google
- Huawei
- Intel
- Microsoft
- NVIDIA
- Tesla
- Many Others and Many Startups…

- **Many More to Come…**

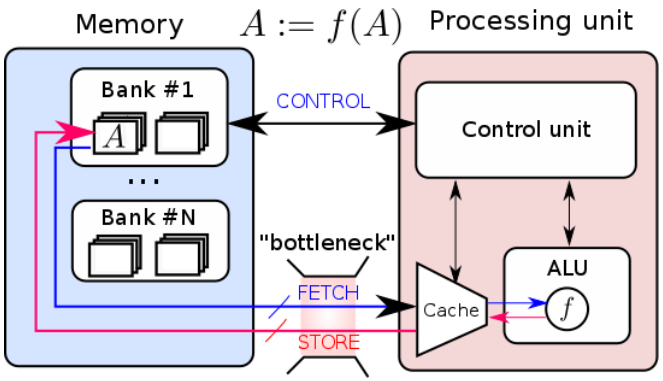
Many (Other) AI/ML Chips



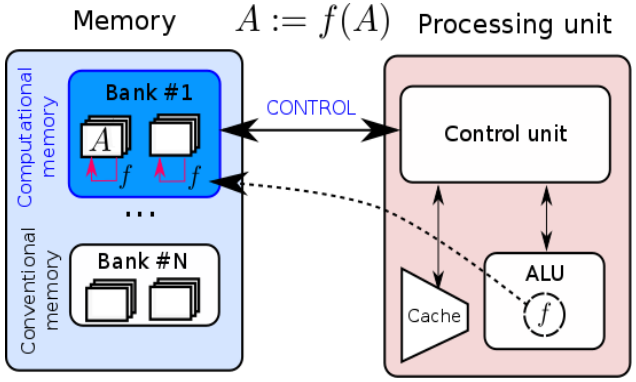
https://basicmi.github.io/AI-Chip/

In-memory computing

Processing unit & Conventional memory



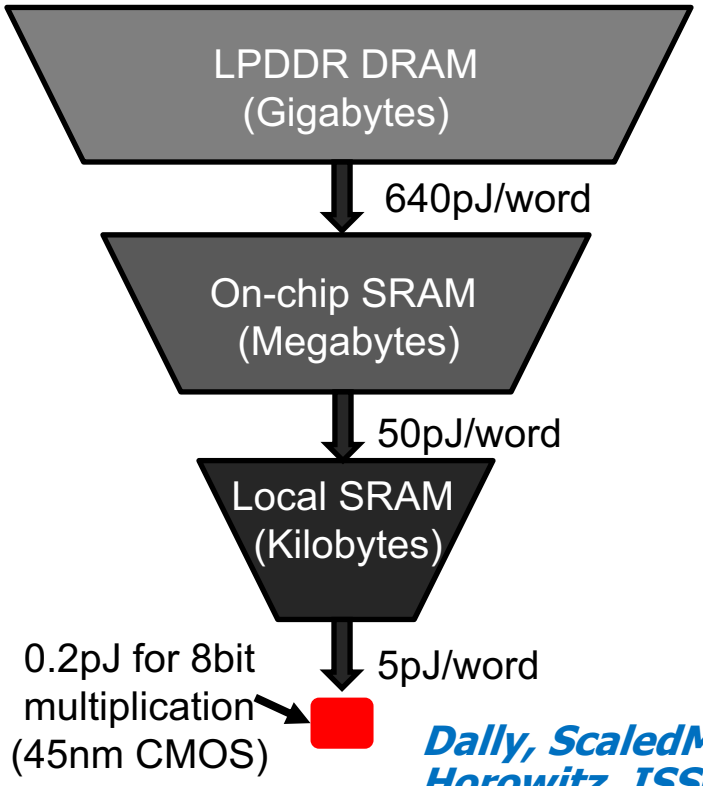
Processing unit & Computational memory



- Perform "certain" computational tasks **in place in memory**
- Achieved by exploiting **the physical attributes of the memory devices**, their **array level organization**, the **peripheral circuitry** as well as the **control logic**
- At **no point during computation**, the memory content is read back and **processed at the granularity of a single memory element**

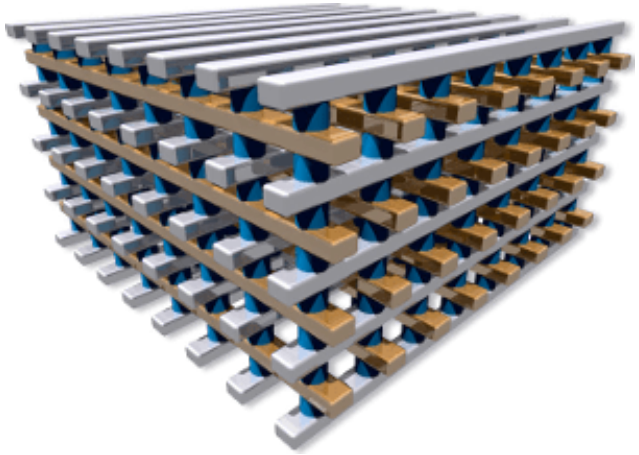
Why in-memory computing?

Reduce the cost of data motion



Dally, ScaledML (2019)
Horowitz, ISSCC (2014)

Reduce computational time complexity



Mostly from massive parallelism and analog way of computing
Reduce data movement

Sebastian et al., Nature Comm. (2017)
Di Ventra, Nature Phys. (2013)

Many Interesting Things Are Happening Today in Computer Architecture

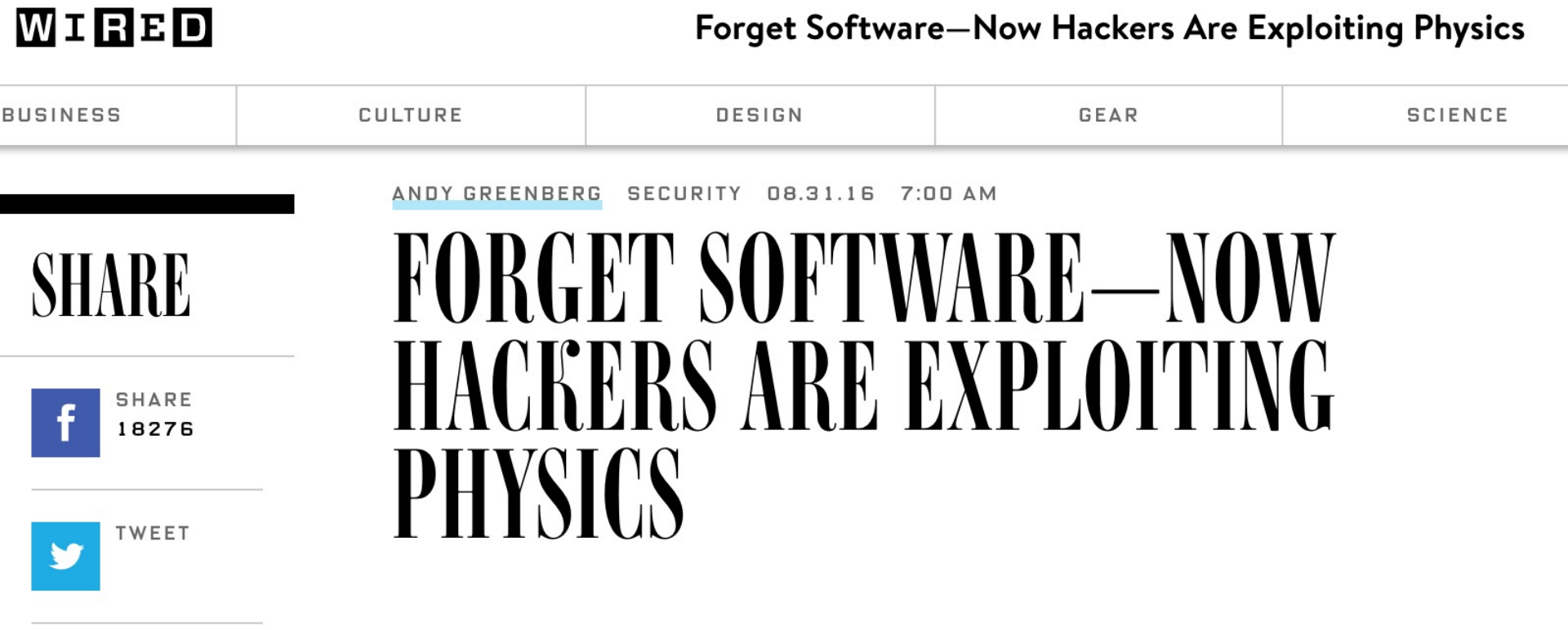
Many Interesting Things Are Happening Today in Computer Architecture

Reliability
and
Security

Security: RowHammer (2014)

The Story of RowHammer

- One can predictably induce bit flips in commodity DRAM chips
  - >80% of the tested DRAM chips are vulnerable
- First example of how a simple hardware failure mechanism can create a widespread system security vulnerability



WIRED

Forget Software—Now Hackers Are Exploiting Physics

BUSINESS | CULTURE | DESIGN | GEAR | SCIENCE

SHARE

SHARE 18276

TWEET

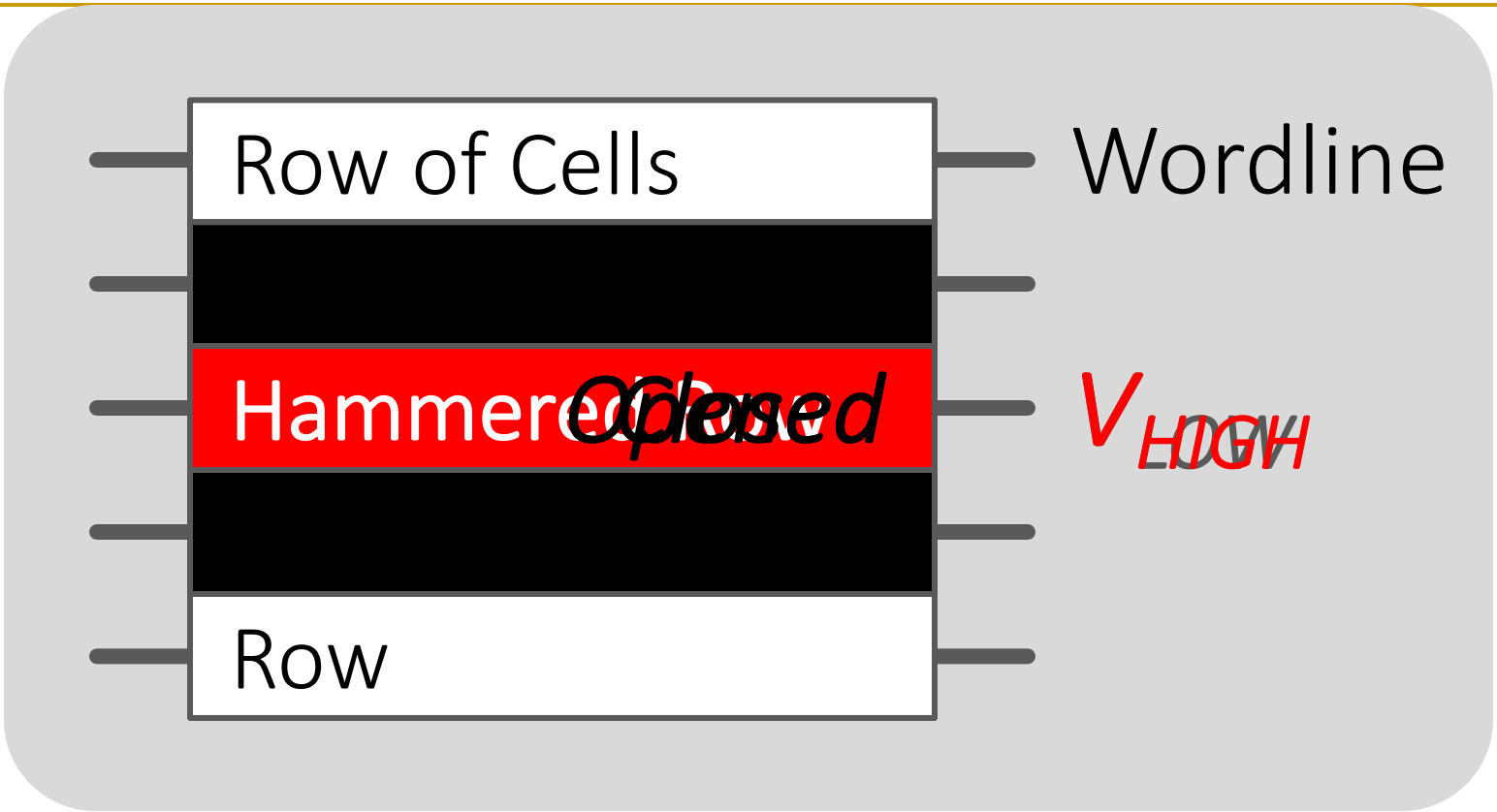
ANDY GREENBERG SECURITY 08.31.16 7:00 AM

FORGET SOFTWARE—NOW HACKERS ARE EXPLOITING PHYSICS

Security: RowHammer (2014)

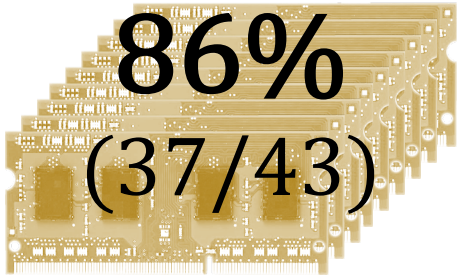
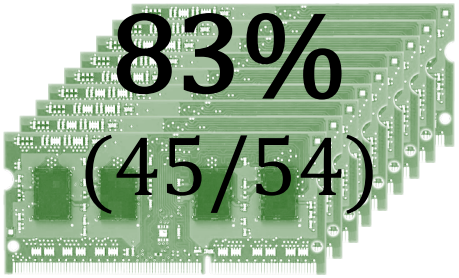
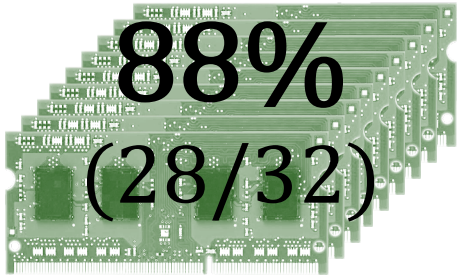
It's like breaking into an apartment by repeatedly slamming a neighbor's door until the vibrations open the door you were after

Modern DRAM is Prone to Disturbance Errors



Repeatedly reading a row enough times (before memory gets refreshed) induces disturbance errors in **adjacent rows** in most real DRAM chips you can buy today

Most DRAM Modules Are Vulnerable

A company	B company	C company
 86% (37/43)	 83% (45/54)	 88% (28/32)
Up to 1.0×10^{7} errors	Up to 2.7×10^{6} errors	Up to 3.3×10^{5} errors

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors, (Kim et al., ISCA 2014)

Security: Meltdown and Spectre (2018)

Source: J. Masters, Redhat, FOSDEM 2018 keynote talk.

Meltdown and Spectre

- Someone can steal secret data from the system even though
  - your program and data are perfectly correct and
  - your hardware behaves according to the specification and
  - there are no software vulnerabilities/bugs

- Why?
  - Speculative execution leaves traces of secret data in the processor's cache (internal storage)
    - It brings data that is not supposed to be brought/accessed if there was no speculative execution
  - A malicious program can inspect the contents of the cache to "infer" secret data that it is not supposed to access
  - A malicious program can actually force another program to speculatively execute code that leaves traces of secret data

More on Meltdown/Spectre Vulnerabilities

Project Zero

News and updates from the Project Zero team at Google

Wednesday, January 3, 2018

Reading privileged memory with a side-channel

Posted by Jann Horn, Project Zero

We have discovered that CPU data cache timing can be abused to efficiently leak information out of mis-speculated execution, leading to (at worst) arbitrary virtual memory read vulnerabilities across local security boundaries in various contexts.

Source: https://googleprojectzero.blogspot.ch/2018/01/reading-privileged-memory-with-side.html

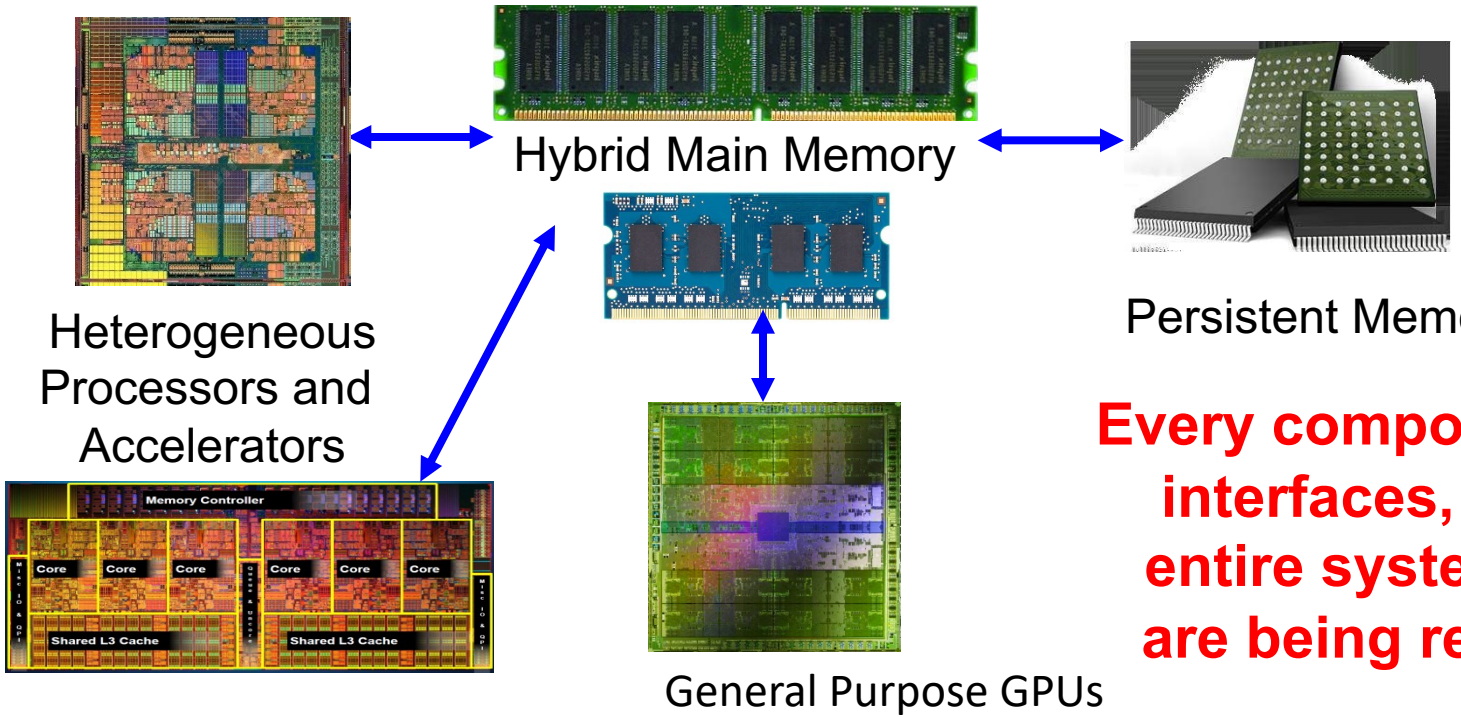
Many Interesting Things Are Happening Today in Computer Architecture

Many Novel Concepts Investigated Today

- New Computing Paradigms (Rethinking the Full Stack)
  - Processing in Memory, Processing Near Data
  - In-Memory Analog Computing
  - Neuromorphic Computing
  - Fundamentally Secure and Dependable Computers
- New Accelerators (Algorithm-Hardware Co-Designs)
  - Artificial Intelligence & Machine Learning
  - Graph Analytics
  - Genome Analysis
- New Memories and Storage Systems
  - Intelligent Memory

Computer Architecture Today

- Computing landscape is very different from 10-20 years ago
- Applications and technology both demand novel architectures



Every component and its interfaces, as well as entire system designs are being re-examined

Computer Architecture Today (II)

- You can revolutionize the way computers are built, if you understand both the hardware and the software (and change each accordingly)

- You can invent new paradigms for computation, communication, and storage

- Recommended book: Thomas Kuhn, "The Structure of Scientific Revolutions" (1962)
  - Pre-paradigm science: no clear consensus in the field
  - Normal science: dominant theory used to explain/improve things (business as usual); exceptions considered anomalies
  - Revolutionary science: underlying assumptions re-examined

Takeaways

- It is an exciting time to be understanding and designing computing architectures
- Many challenging and exciting problems in platform design
  - That no one has tackled (or thought about) before
  - That can have huge impact on the world's future
- Driven by huge hunger for data (Big Data), new applications (ML/AI, graph analytics, genomics), ever-greater realism, …
  - We can easily collect more data than we can analyze/understand
- Driven by significant difficulties in keeping up with that hunger at the technology layer
  - Five walls: Energy, reliability, complexity, security, scalability

Major High-Level Goals of This Course

In Computer Architecture

- Understand the basics
- Understand the principles (of design)
- Understand the precedents

- Based on such understanding:
  - learn how a modern computer works underneath
  - evaluate tradeoffs of different designs and ideas
  - implement a principled design (a simple microprocessor)
  - Hopefully enable you to develop novel, out-of-the-box designs

- The focus is on basics, principles, precedents, and how to use them to create/implement good designs, tradeoffs are important!

Why These Goals?

- Because you are here for a Computer Science degree

- Regardless of your future direction, learning the principles of computer architecture will be useful to
  - design better systems (software + hardware)
  - make better tradeoffs in design
  - understand why computers behave the way they do
  - solve problems better
  - think "in parallel"
  - think critically
  - …

I presume you all know the number systems?

- Binary Number
- Hexadecimal Numbers
- Bits, Bytes, Words
- least significant bit (lsb), most significant bit (msb)
- Least Significant Byte (LSB), Most Significant Byte (MSB)
- KB, MB, GB, TB
- Binary Addition
- Signed Binary Numbers