Computer Architecture

Lecture 1: Introduction and Basics

Prof. Onur Mutlu

ETH Zürich

Fall 2019

19 September 2019

Brief Self Introduction

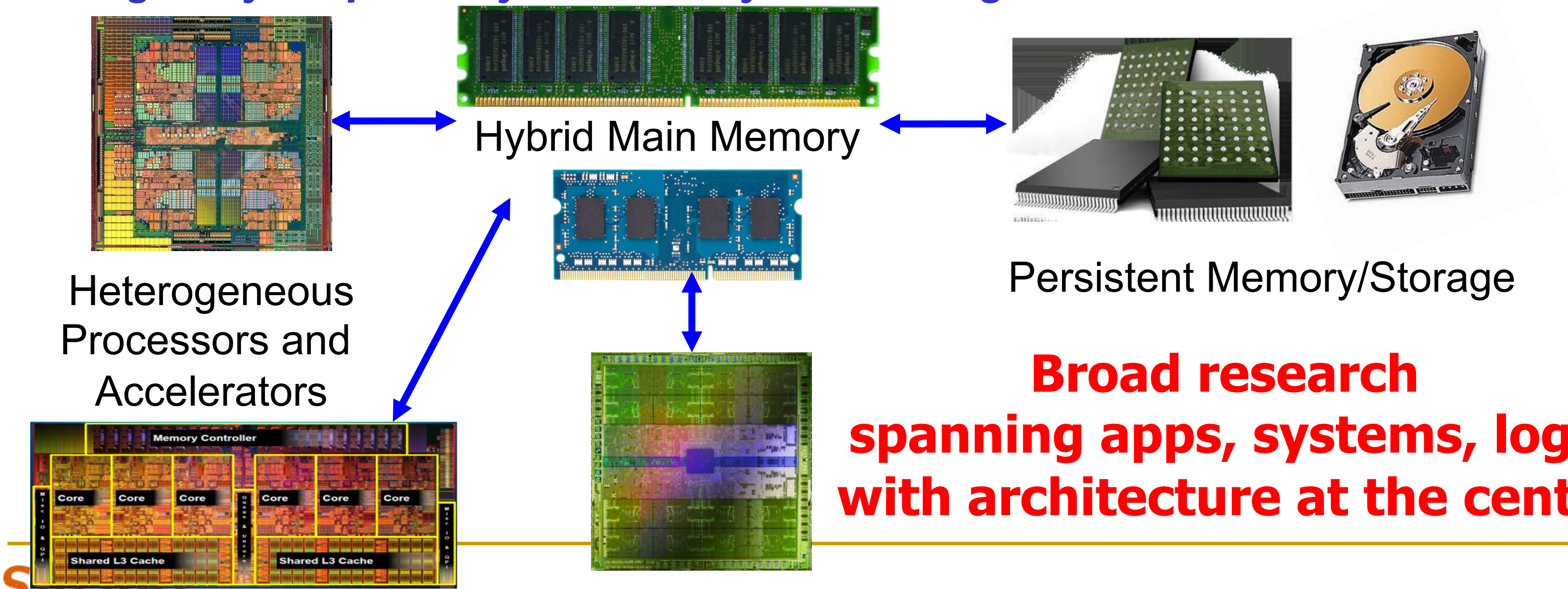
- Onur Mutlu
  - Full Professor @ ETH Zurich CS (EE), since September 2015
  - Strecker Professor @ Carnegie Mellon University ECE/CS, 2009-2016, 2016-…
  - PhD from UT-Austin, worked at Google, VMware, Microsoft Research, Intel, AMD
  - https://people.inf.ethz.ch/omutlu/
  - omutlu@gmail.com (Best way to reach me)
  - https://people.inf.ethz.ch/omutlu/projects.htm

- Research and Teaching in:
  - Computer architecture, computer systems, hardware security, bioinformatics
  - Memory and storage systems
  - Hardware security, safety, predictability
  - Fault tolerance
  - Hardware/software cooperation
  - Architectures for bioinformatics, health, medicine
  - …

Current Research Focus Areas

Research Focus: **Computer architecture, HW/SW, bioinformatics, security**

- *Memory and storage (DRAM, flash, emerging), interconnects*
- *Heterogeneous & parallel systems, GPUs, systems for data analytics*
- *System/architecture interaction, new execution models, new interfaces*
- *Hardware security, energy efficiency, fault tolerance, performance*
- *Genome sequence analysis & assembly algorithms and architectures*
- *Biologically inspired systems & system design for bio/medicine*



Hybrid Main Memory

Heterogeneous Processors and Accelerators

Persistent Memory/Storage

Graphics and Vision Processing

Broad research spanning apps, systems, logic with architecture at the center

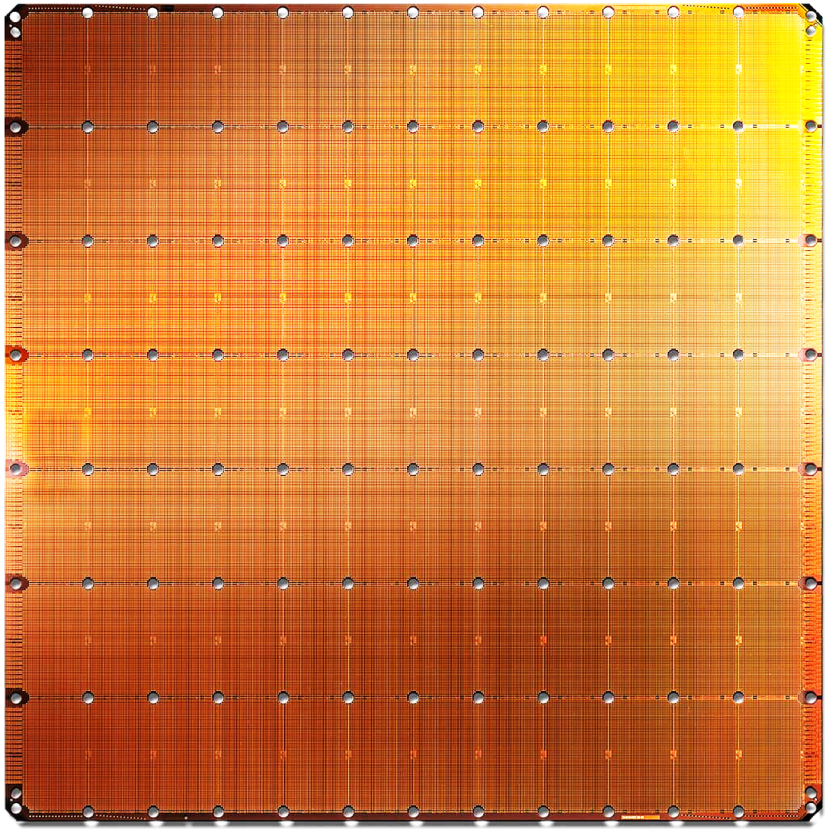
Many Interesting Things Are Happening Today in Computer Architecture

Intel Optane Persistent Memory (2019)

- Non-volatile main memory
- Based on 3D-XPoint Technology

https://www.storagereview.com/intel_optane_dc_persistent_memory_module_pmm

Cerebras's Wafer Scale Engine (2019)



- The largest ML accelerator chip
- 400,000 cores

Cerebras WSE
1.2 Trillion transistors
46,225 mm^2

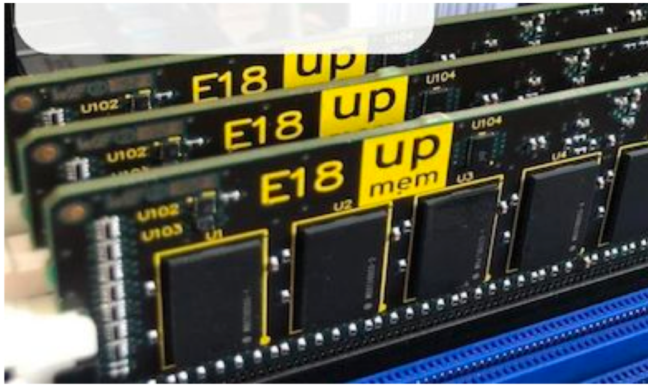
Largest GPU
21.1 Billion transistors
815 mm^2
NVIDIA TITAN V

https://www.anandtech.com/show/14758/hot-chips-31-live-blogs-cerebras-wafer-scale-deep-learning

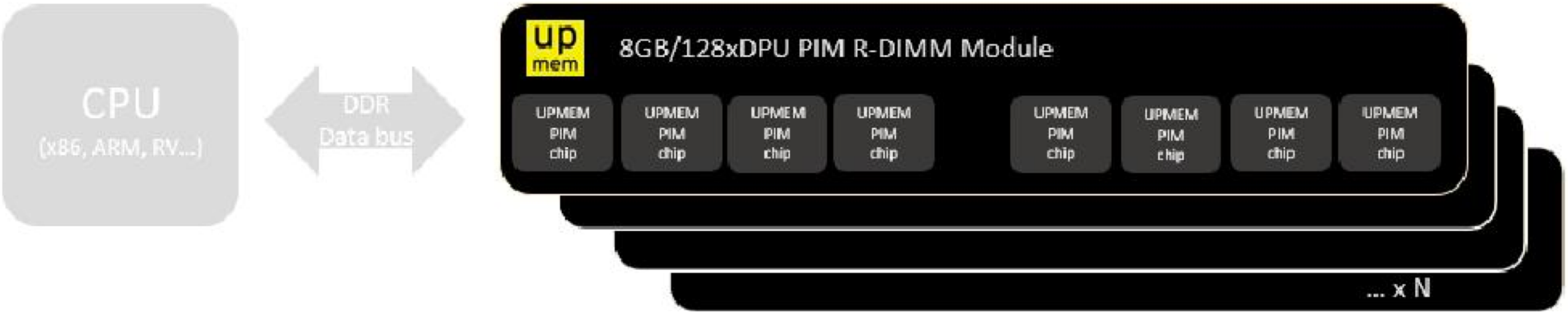
https://www.cerebras.net/cerebras-wafer-scale-engine-why-we-need-big-chips-for-deep-learning/

UPMEM Processing-in-DRAM Engine (2019)

- Processing in DRAM Engine
- Includes **standard DIMM modules**, with a **large number of DPU processors** combined with DRAM chips.



- Replaces **standard** DIMMs
  - DDR4 R-DIMM modules
    - 8GB+128 DPUs (16 PIM chips)
    - Standard 2x-nm DRAM process
  - **Large amounts of** compute & memory bandwidth

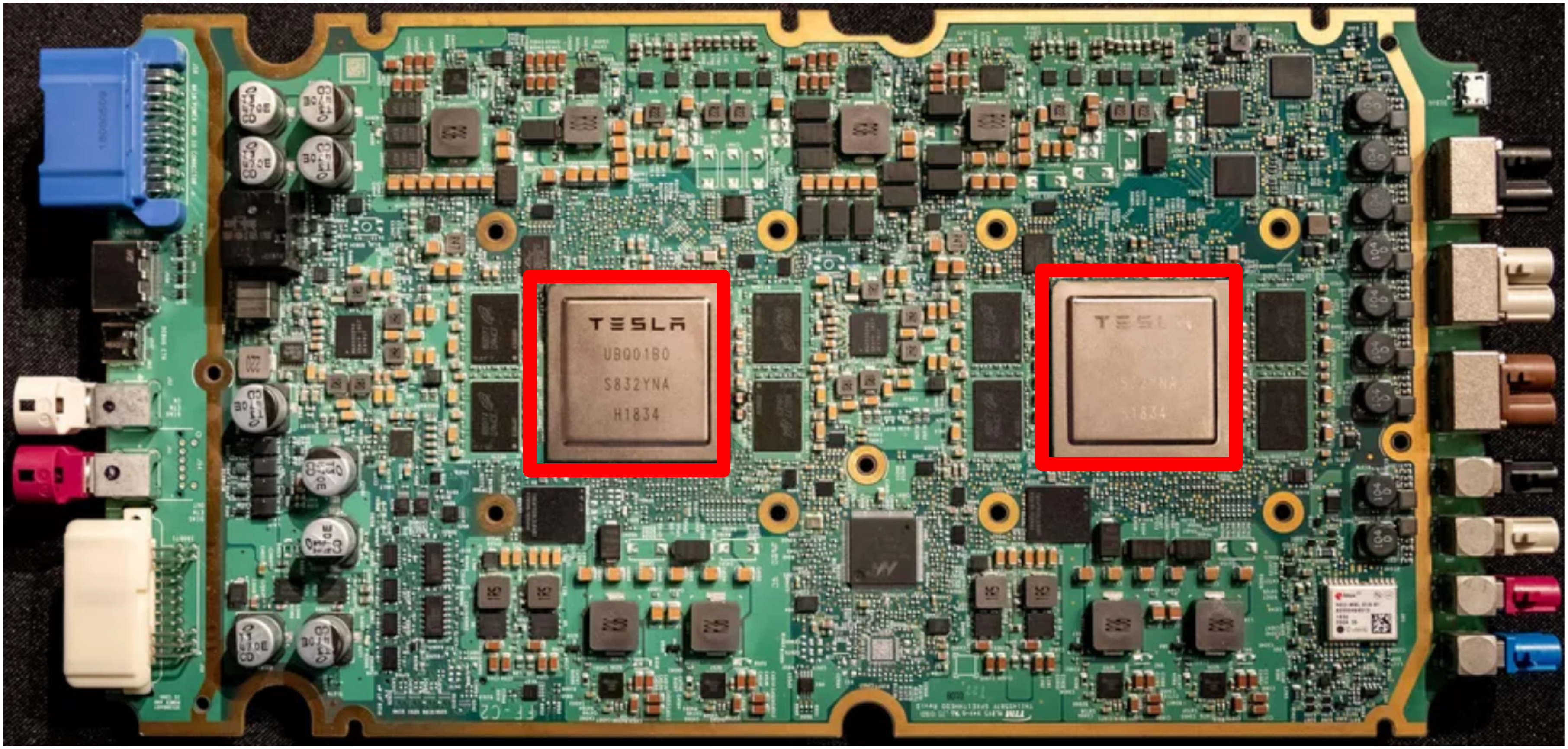


CPU
(x86, ARM, RV...)
DDR
Data bus
up mem
8GB/128xDPU PIM R-DIMM Module
UPMEM PIM chip
UPMEM PIM chip
UPMEM PIM chip
UPMEM PIM chip
UPMEM PIM chip
UPMEM PIM chip
UPMEM PIM chip
UPMEM PIM chip
... x N

https://www.anandtech.com/show/14750/hot-chips-31-analysis-inmemory-processing-by-upmem
https://www.upmem.com/video-upmem-presenting-its-true-processing-in-memory-solution-hot-chips-2019/

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)

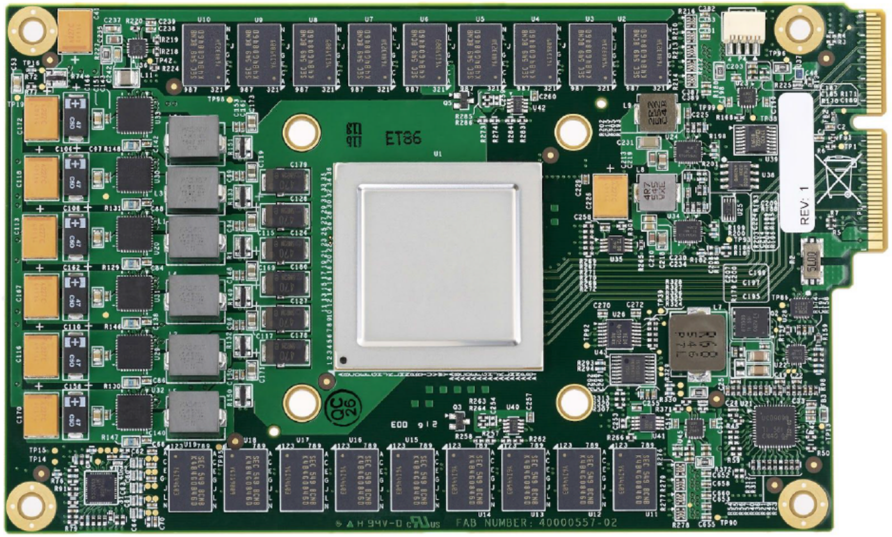


Figure 3. TPU Printed Circuit Board. It can be inserted in the slot for an SATA disk in a server, but the card uses PCIe Gen3 x16.

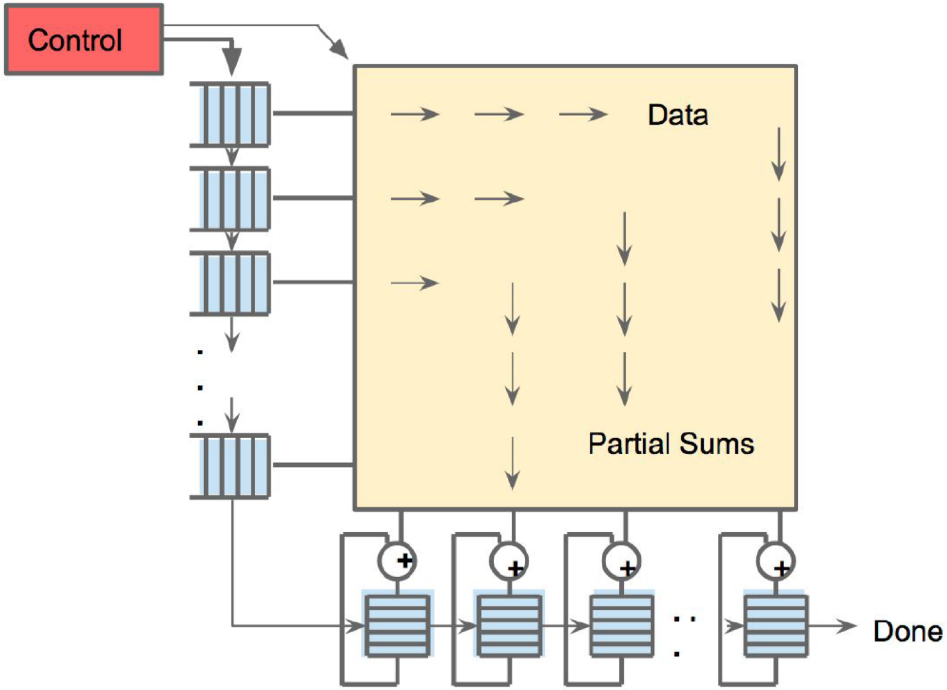
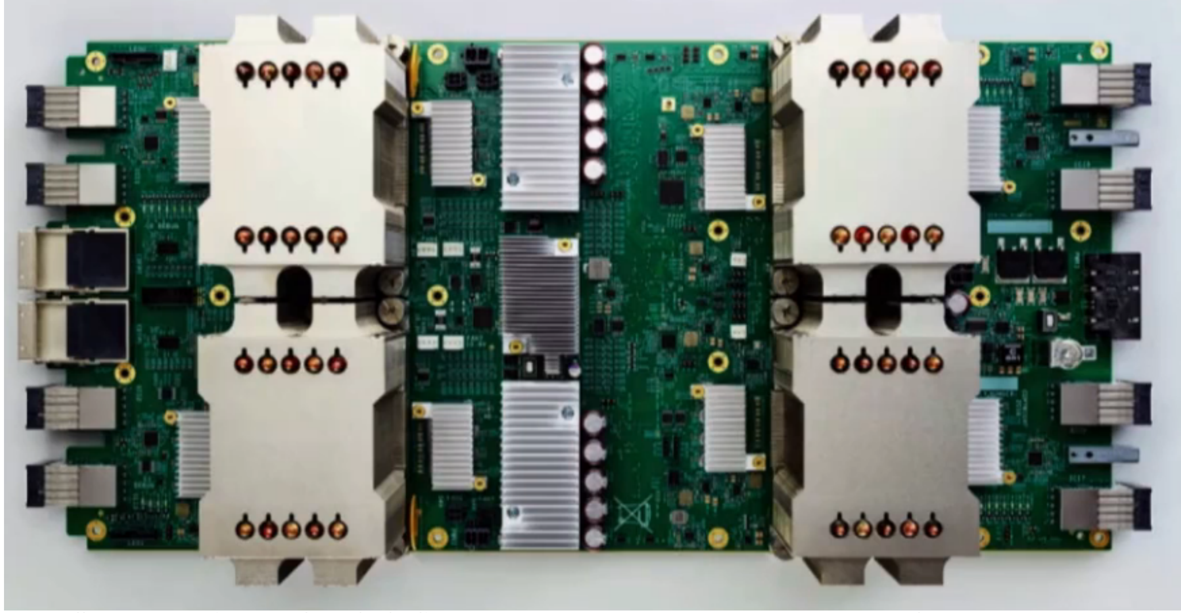


Figure 4. Systolic data flow of the Matrix Multiply Unit. Software has the illusion that each 256B input is read at once, and they instantly update one location of each of 256 accumulator RAMs.

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

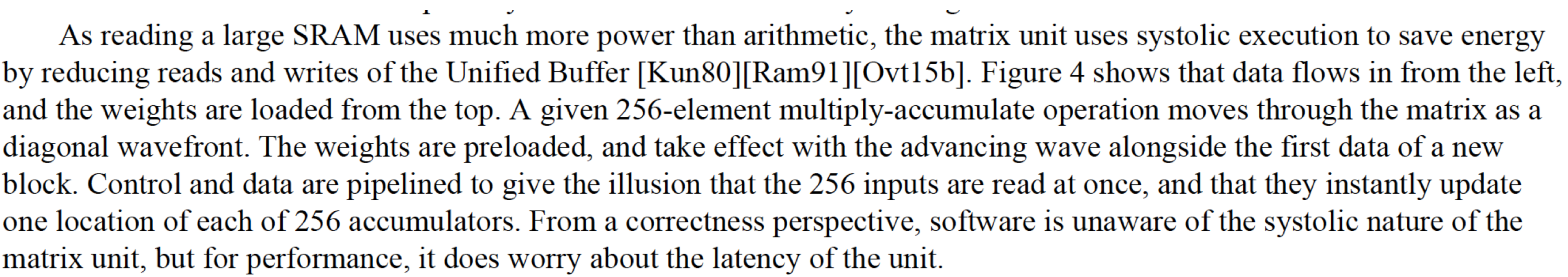
High Bandwidth Memory
vs DDR3

Floating point operations
vs FP16

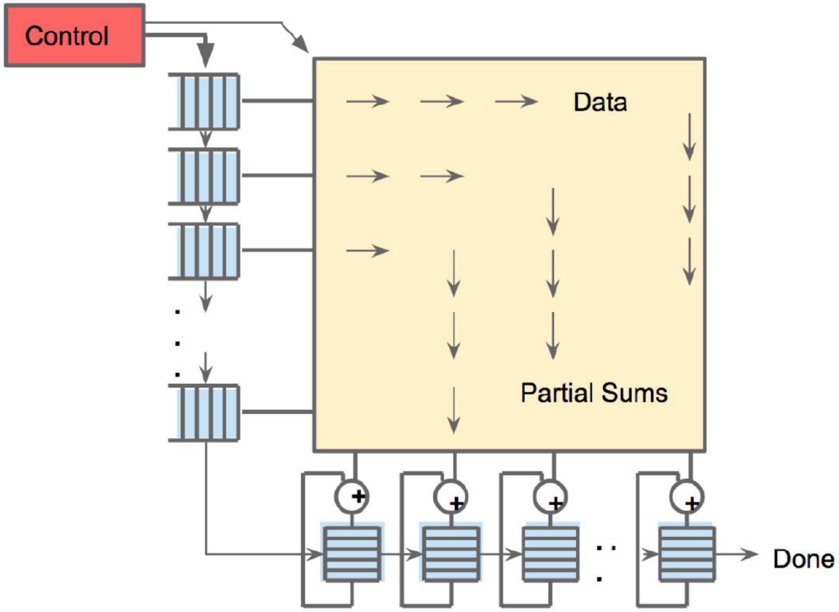
45 TFLOPS per chip
vs 23 TOPS

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)

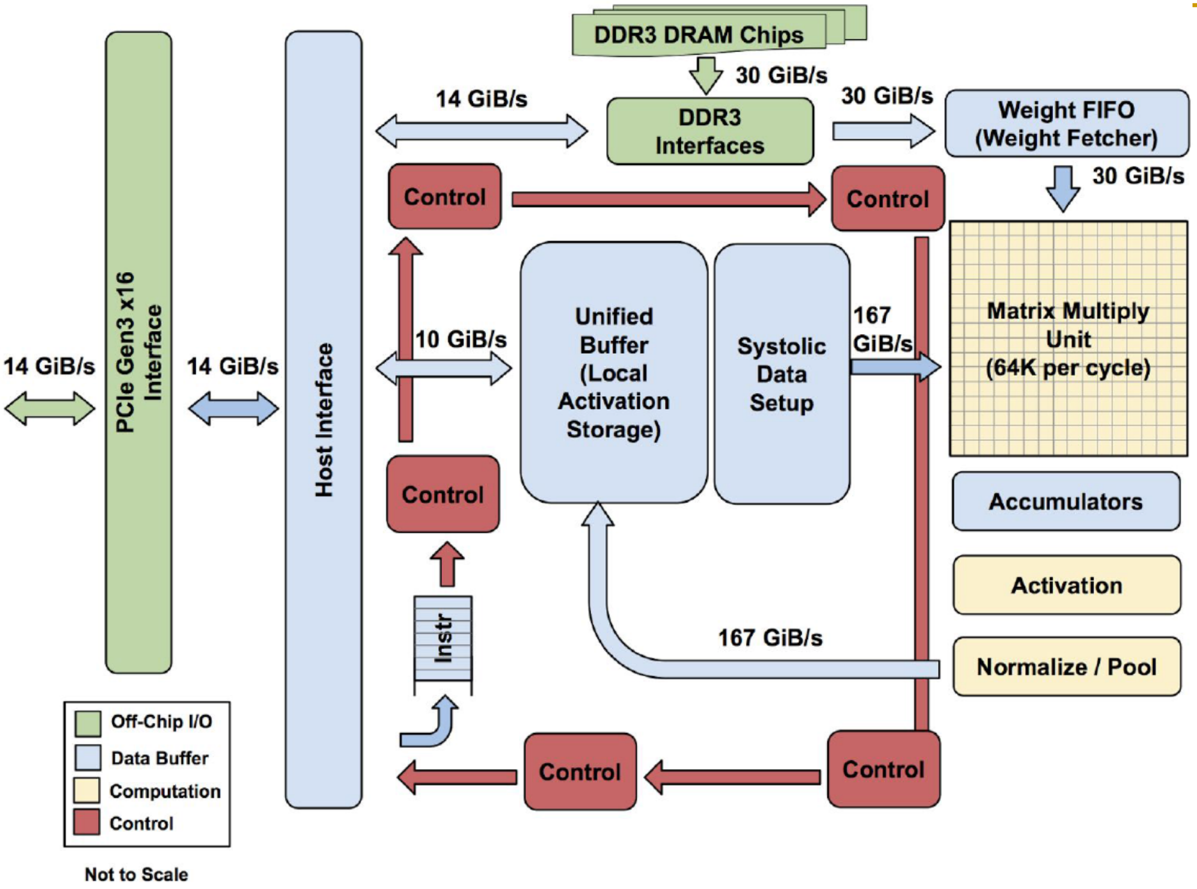


Figure 1. TPU Block Diagram. The main computation part is the yellow Matrix Multiply unit in the upper right hand corner. Its inputs are the blue Weight FIFO and the blue Unified Buffer (UB) and its output is the blue Accumulators (Acc). The yellow Activation Unit performs the nonlinear functions on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

Many Concepts Being Investigated Today

- New Computing Paradigms
    - Processing in Memory
    - Neuromorphic Computing

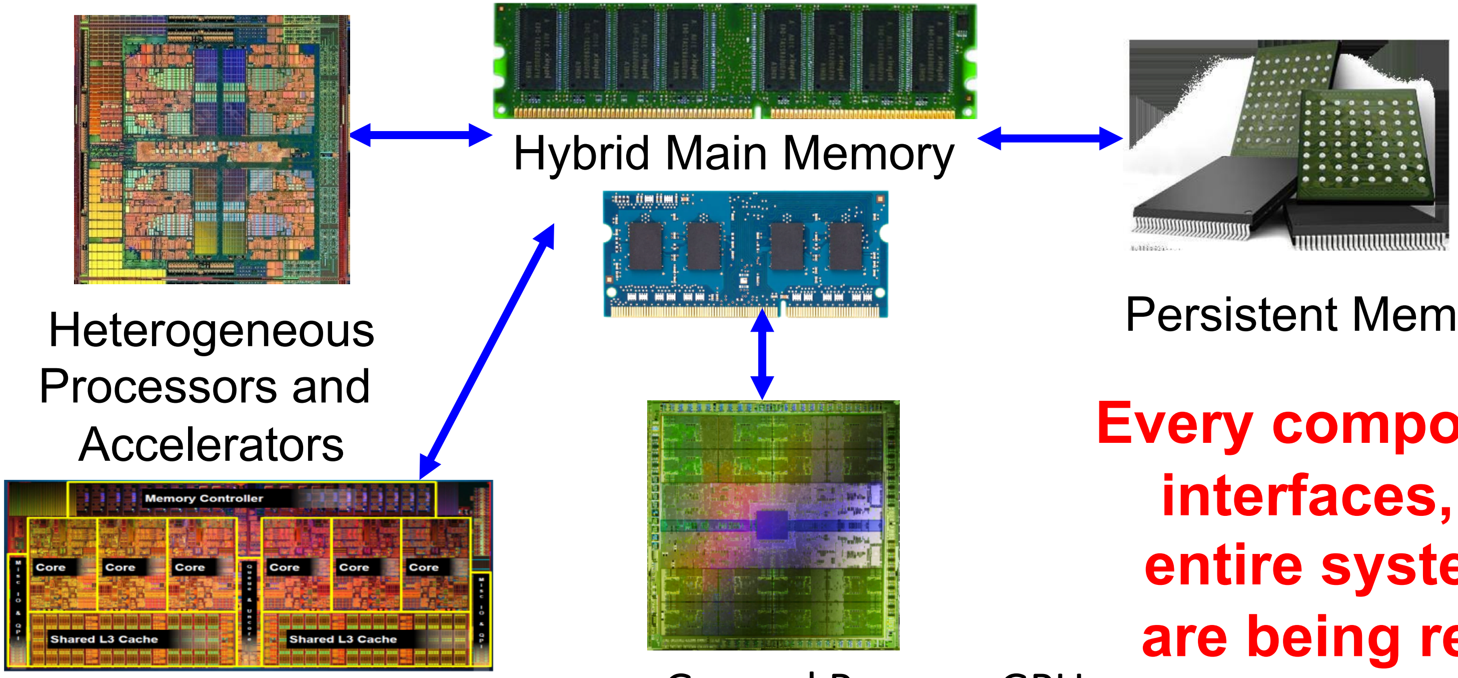
- New Accelerators
    - Machine Learning
    - Graph Analytics
    - Genome Analysis

- New Systolic Architectures

- New Memories

Computer Architecture Today

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



Hybrid Main Memory

Persistent Memory/Storage

Heterogeneous Processors and Accelerators

General Purpose GPUs

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

Computer Architecture Today (II)

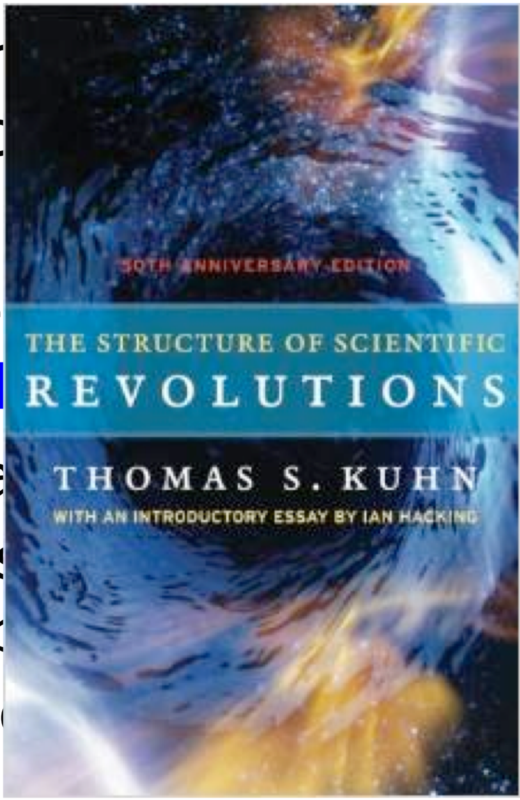
- You can revolutionize the way computers are built, if you understand both the hardware and the software (and change each accordingly)

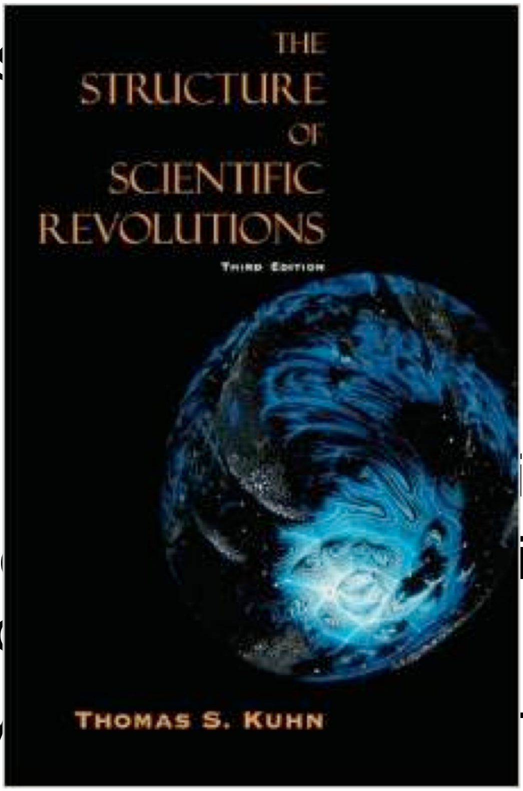
- You can invent new paradigms for computation, communication, and storage

- 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

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)


50TH ANNIVERSARY EDITION
THE STRUCTURE OF SCIENTIFIC
REVOLUTIONS
THOMAS S. KUHN
WITH AN INTRODUCTORY ESSAY BY IAN HACKING

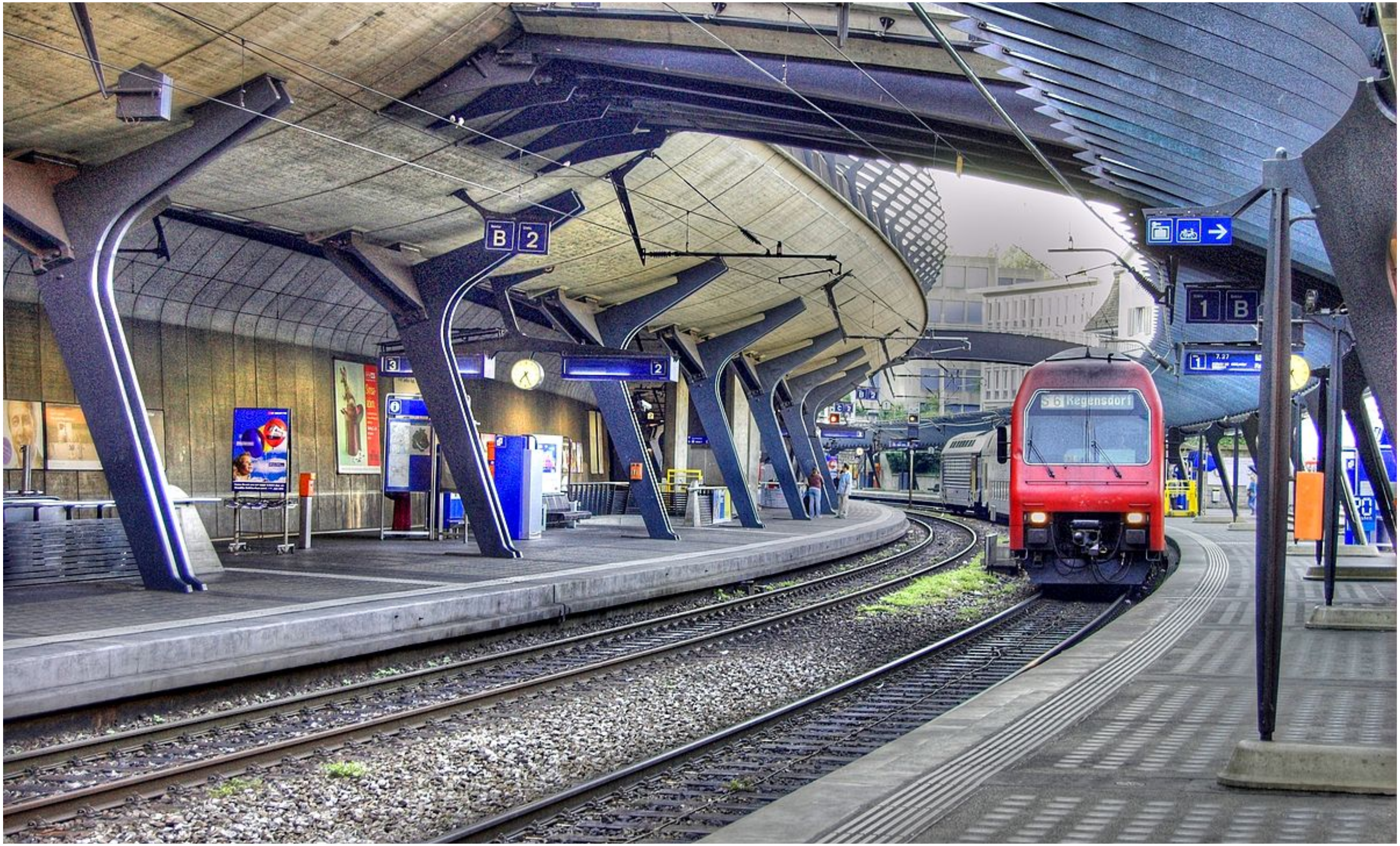


- You can ir ms
  communic e

- Recomme as ure of
  Scientific I)
  - Pre-para ea ield
  - Normal s ne improve
    things (b ex anomalies
  - Revoluti rly examined

Let’s Start with Some Fundamentals

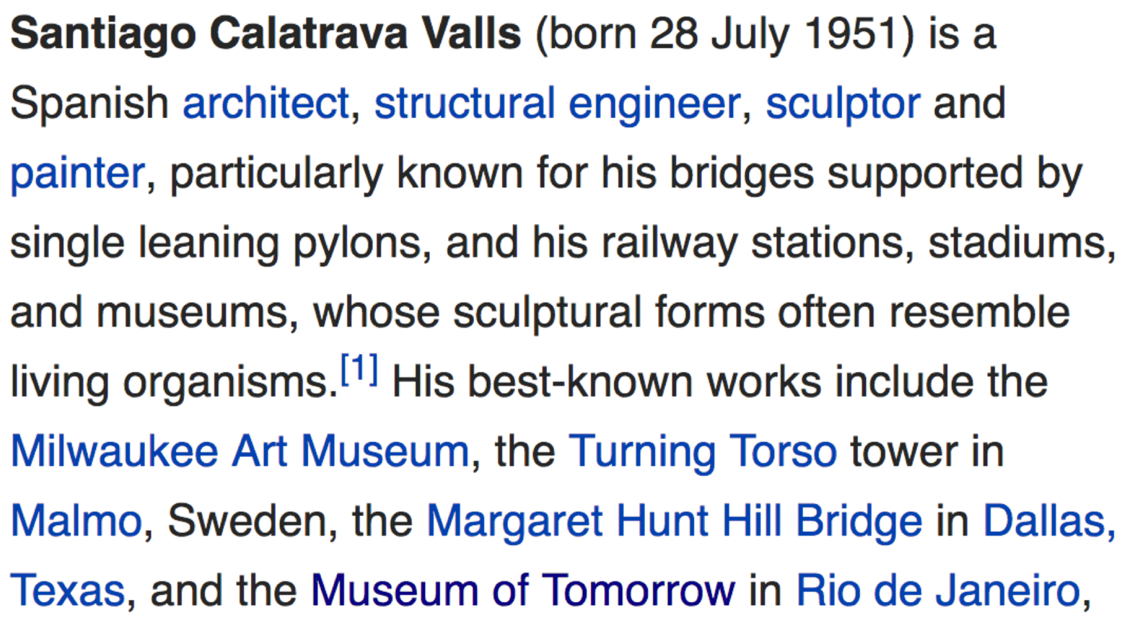
Question: What Is This?



Source: By Toni_V, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=4087256

Answer: The First Major Piece of a Famous Architect

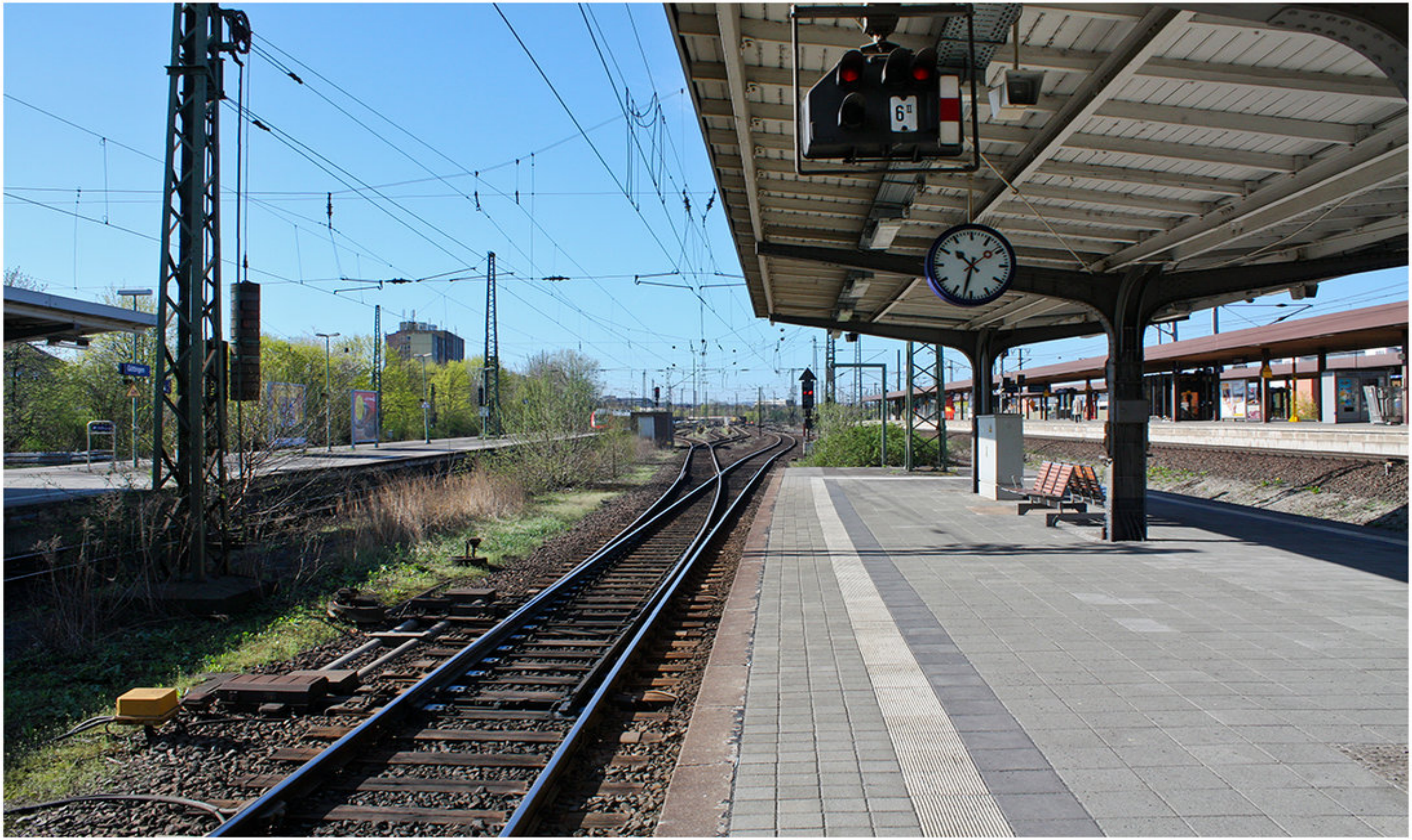
- **Bahnhof Stadelhofen:** “The train station has several of the features that became signatures of his work; straight lines and right angles are rare.“
- ETH Alumnus, PhD in Civil Engineering



Santiago Calatrava Valls (born 28 July 1951) is a Spanish architect, structural engineer, sculptor and painter, particularly known for his bridges supported by single leaning pylons, and his railway stations, stadiums, and museums, whose sculptural forms often resemble living organisms.[1] His best-known works include the Milwaukee Art Museum, the Turning Torso tower in Malmo, Sweden, the Margaret Hunt Hill Bridge in Dallas, Texas, and the Museum of Tomorrow in Rio de Janeiro,

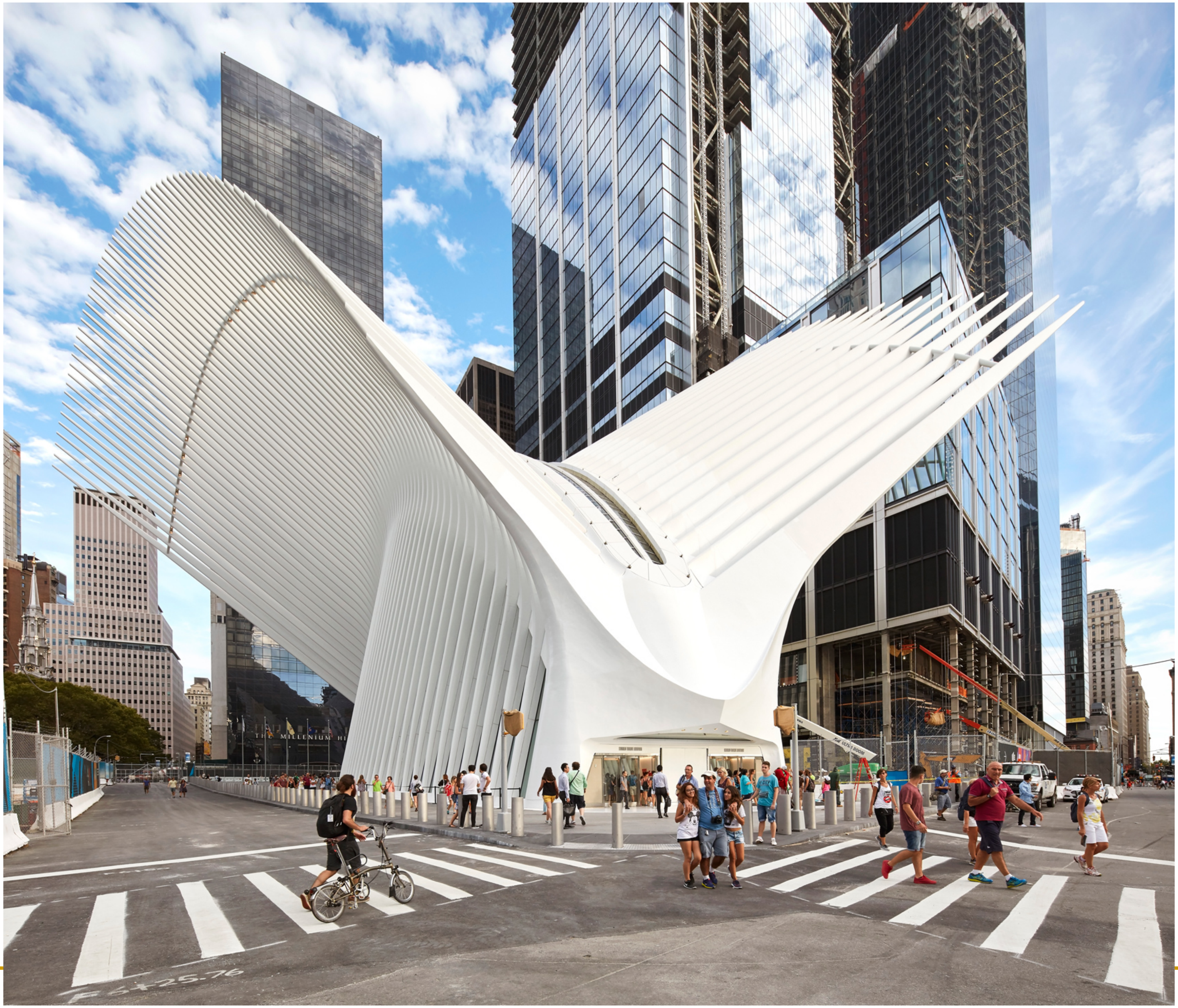
Source: By 準建築人手札網站 Forgemind ArchiMedia - Flickr: IMG_2489.JPG, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=31493356, https://en.wikipedia.org/wiki/Santiago_Calatrava

Compare To This



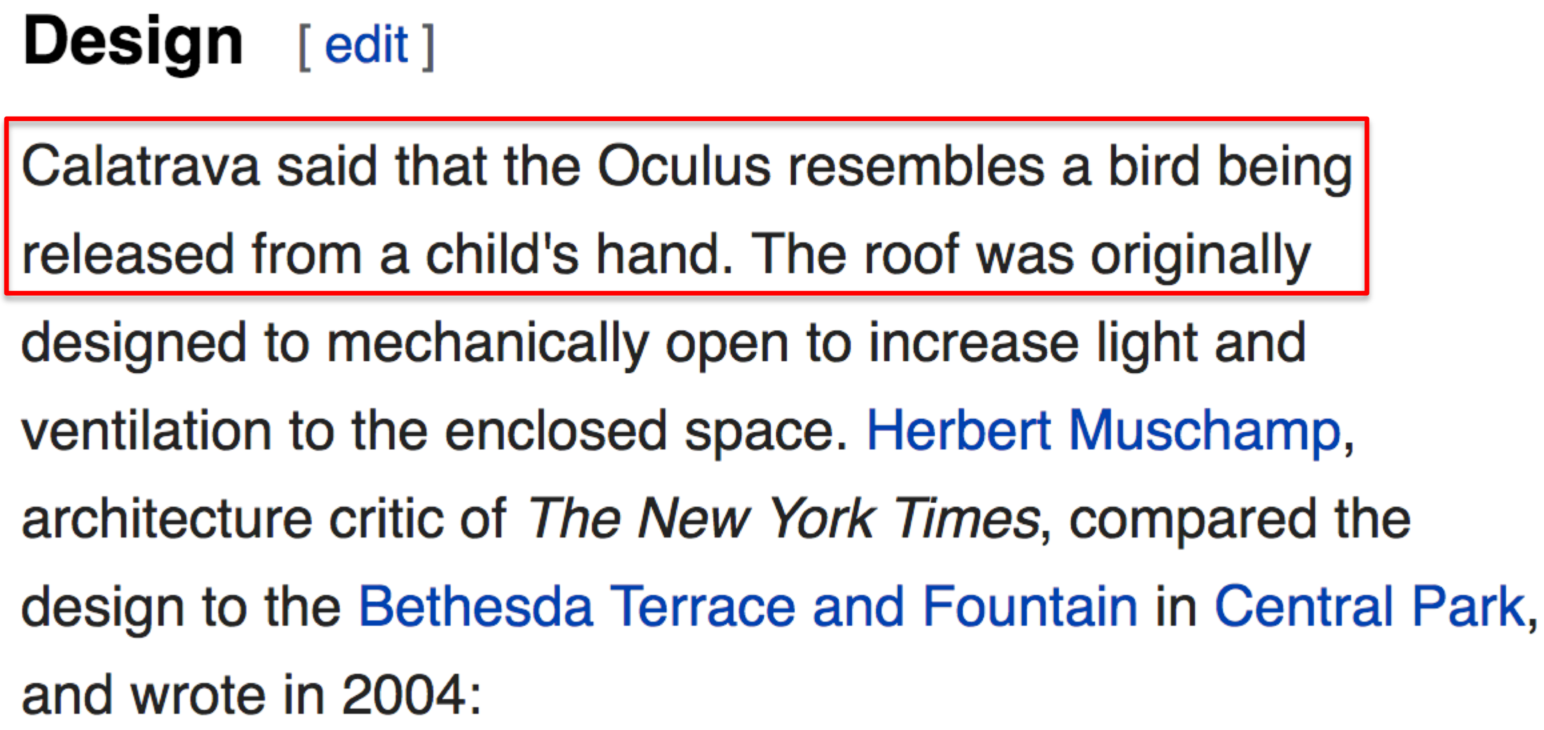
Source: http://cookiemagik.deviantart.com/art/Train-station-207266944

Question 2: What Is This?



Source: https://www.dezeen.com/2016/08/29/santiago-calatrava-oculus-world-trade-center-transportation-hub-new-york-photographs-hufton-crow/

Answer: Masterpiece of a Famous Architect

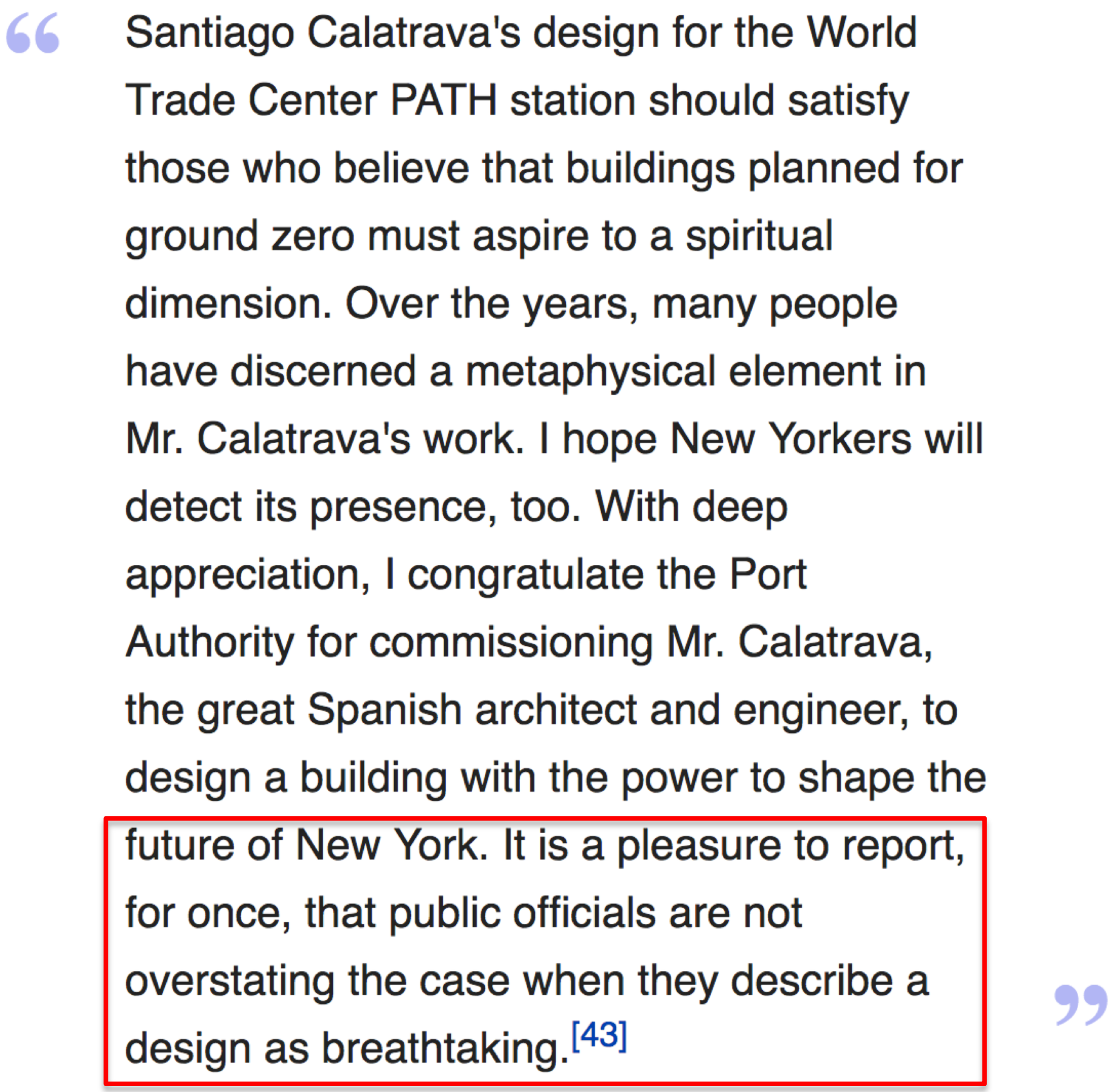


Design [edit]

Calatrava said that the Oculus resembles a bird being released from a child's hand. The roof was originally designed to mechanically open to increase light and ventilation to the enclosed space. Herbert Muschamp, architecture critic of *The New York Times*, compared the design to the Bethesda Terrace and Fountain in Central Park, and wrote in 2004:

Source: https://en.wikipedia.org/wiki/World_Trade_Center_station_(PATH)

Strengths and Praise



> Santiago Calatrava's design for the World Trade Center PATH station should satisfy those who believe that buildings planned for ground zero must aspire to a spiritual dimension. Over the years, many people have discerned a metaphysical element in Mr. Calatrava's work. I hope New Yorkers will detect its presence, too. With deep appreciation, I congratulate the Port Authority for commissioning Mr. Calatrava, the great Spanish architect and engineer, to design a building with the power to shape the future of New York. It is a pleasure to report, for once, that public officials are not overstating the case when they describe a design as breathtaking.[43]

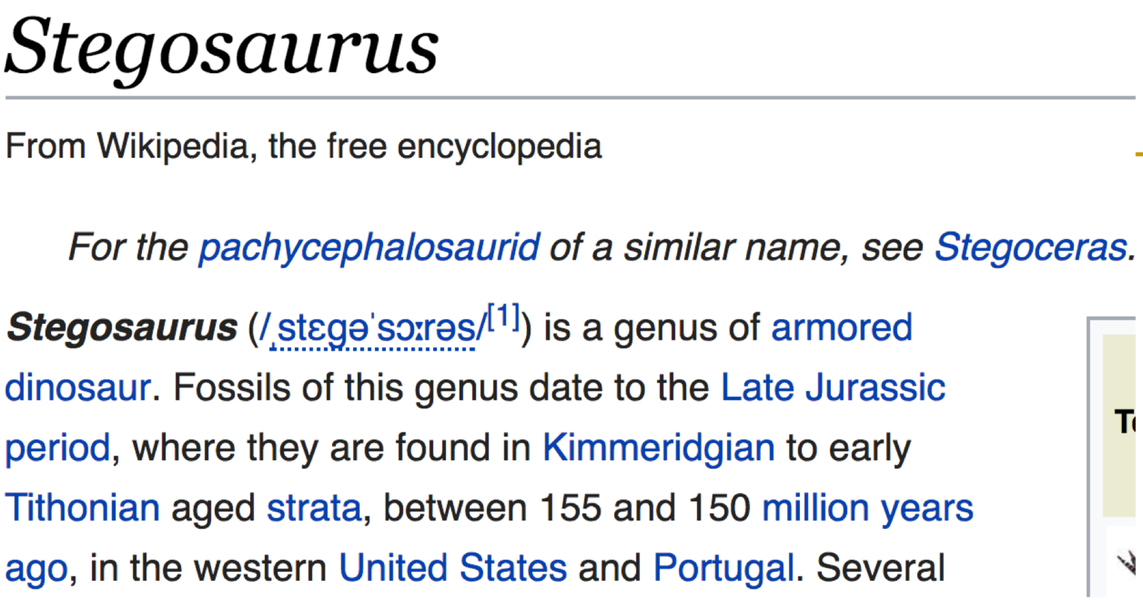
Source: https://en.wikipedia.org/wiki/World_Trade_Center_station_(PATH)

Design Constraints and Criticism

However, Calatrava's original soaring spike design was scaled back because of security issues. The *New York Times* observed in 2005:

> In the name of security, Santiago Calatrava's bird has grown a beak. Its ribs have doubled in number and its wings have lost their interstices of glass.... [T]he main transit hall, between Church and Greenwich Streets, will almost certainly lose some of its delicate quality, while gaining structural expressiveness. It may now evoke a slender stegosaurus more than it does a bird.[45]

Source: https://en.wikipedia.org/wiki/World_Trade_Center_station_(PATH)

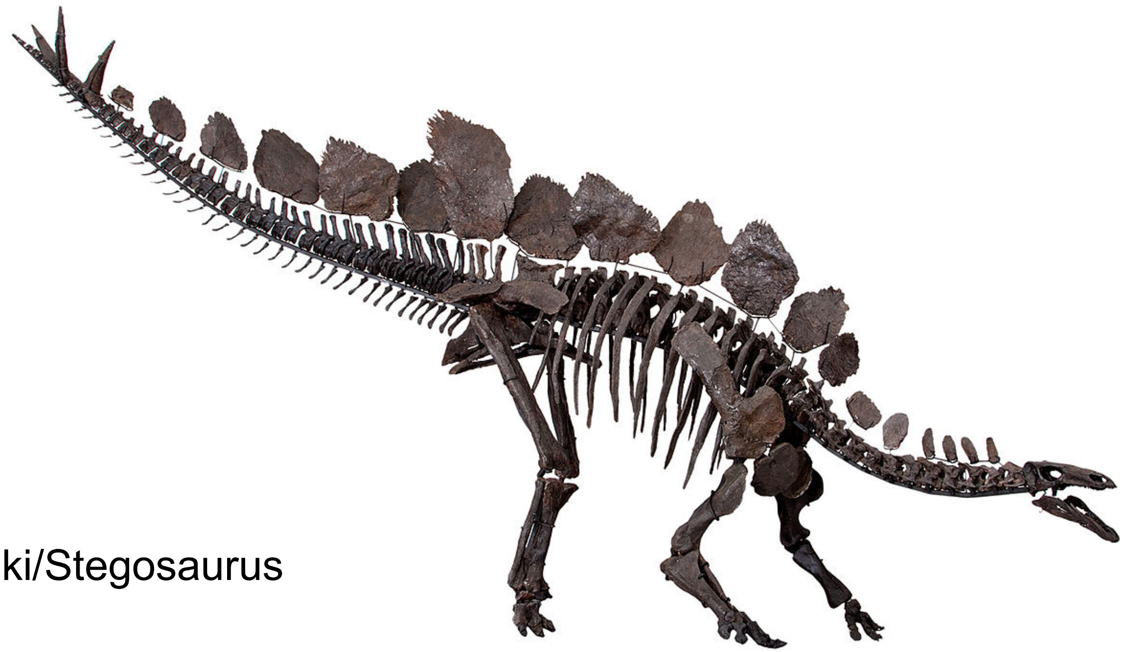


Stegosaurus

From Wikipedia, the free encyclopedia

For the pachycephalosaurid of a similar name, see Stegoceras.

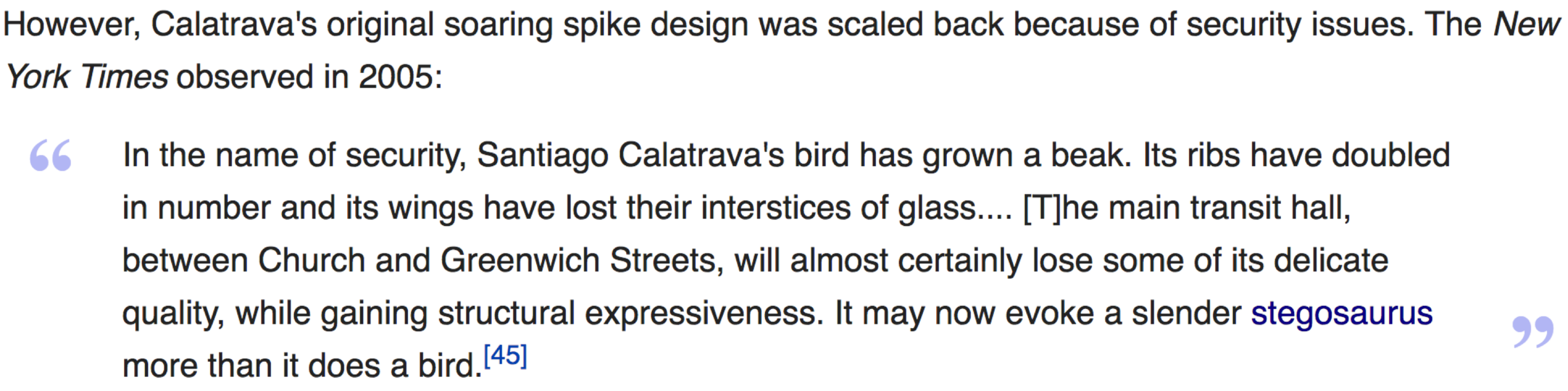
Stegosaurus (/ˌstɛɡəˈsɔːrəs/[1]) is a genus of armored dinosaur. Fossils of this genus date to the Late Jurassic period, where they are found in Kimmeridgian to early Tithonian aged strata, between 155 and 150 million years ago, in the western United States and Portugal. Several



Source: https://en.wikipedia.org/wiki/Stegosaurus

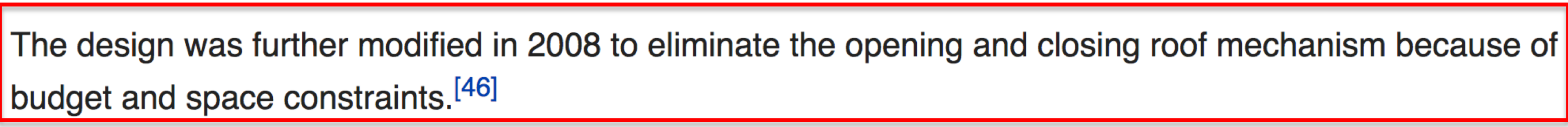
Susannah Maidment et al. & Natural History Museum, London - Maidment SCR, Brassey C, Barrett PM (2015) The Postcranial Skeleton of an Exceptionally Complete Individual of the Plated Dinosaur Stegosaurus stenops (Dinosauria: Thyreophora) from the Upper Jurassic Morrison Formation of Wyoming, U.S.A. PLoS ONE 10(10): e0138352. doi:10.1371/journal.pone.0138352

Design Constraints: Noone is Immune

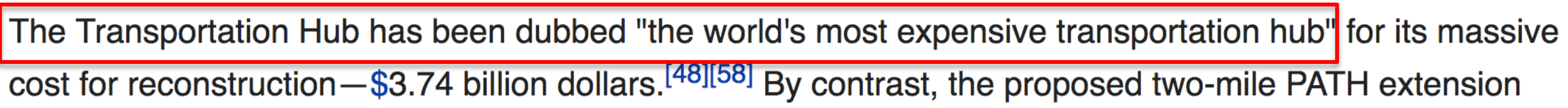


However, Calatrava's original soaring spike design was scaled back because of security issues. The *New York Times* observed in 2005:

> In the name of security, Santiago Calatrava's bird has grown a beak. Its ribs have doubled in number and its wings have lost their interstices of glass.... [T]he main transit hall, between Church and Greenwich Streets, will almost certainly lose some of its delicate quality, while gaining structural expressiveness. It may now evoke a slender stegosaurus more than it does a bird.[45]



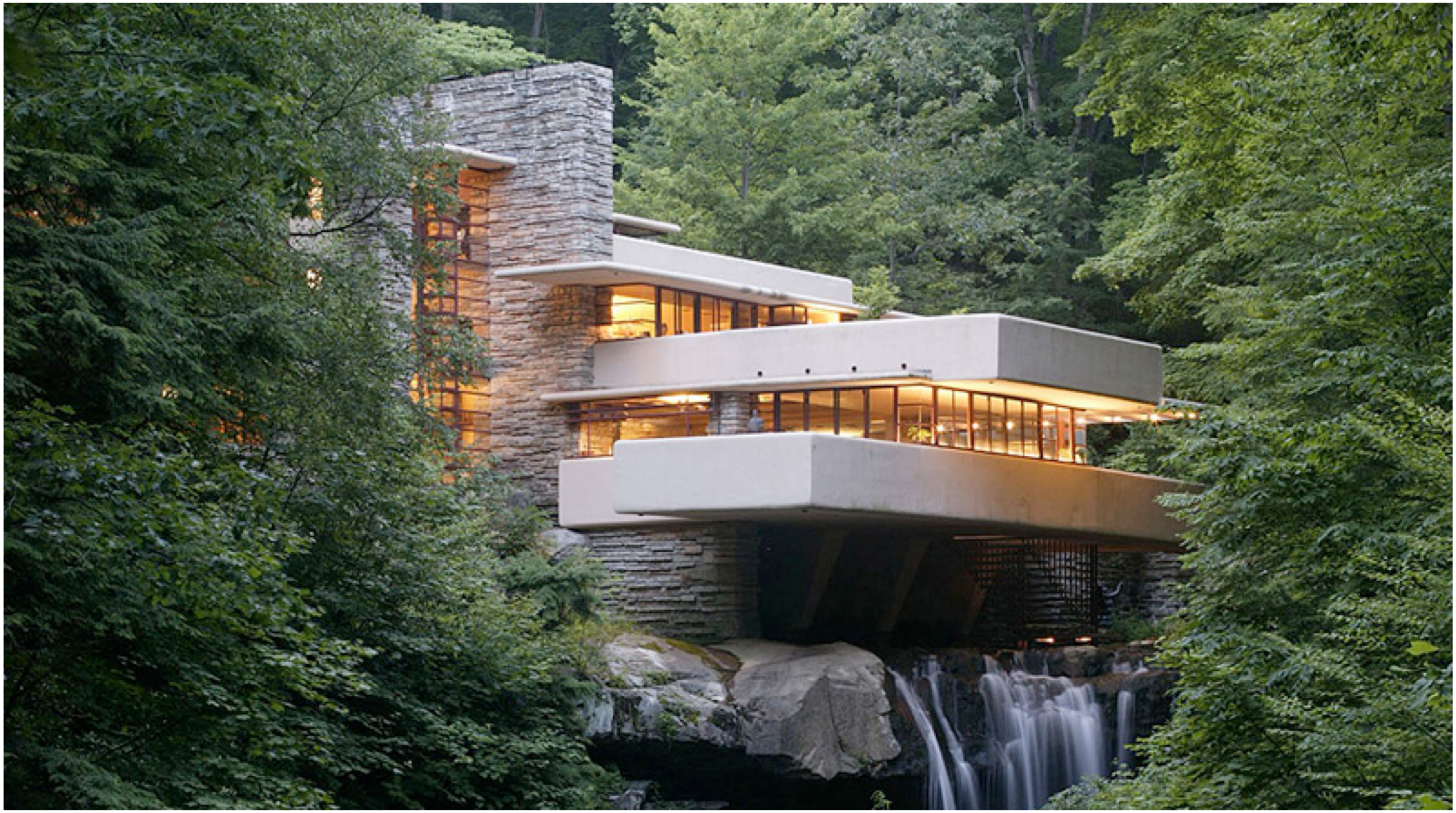
The design was further modified in 2008 to eliminate the opening and closing roof mechanism because of budget and space constraints.[46]



The Transportation Hub has been dubbed "the world's most expensive transportation hub" for its massive cost for reconstruction—$3.74 billion dollars.[48][58] By contrast, the proposed two-mile PATH extension

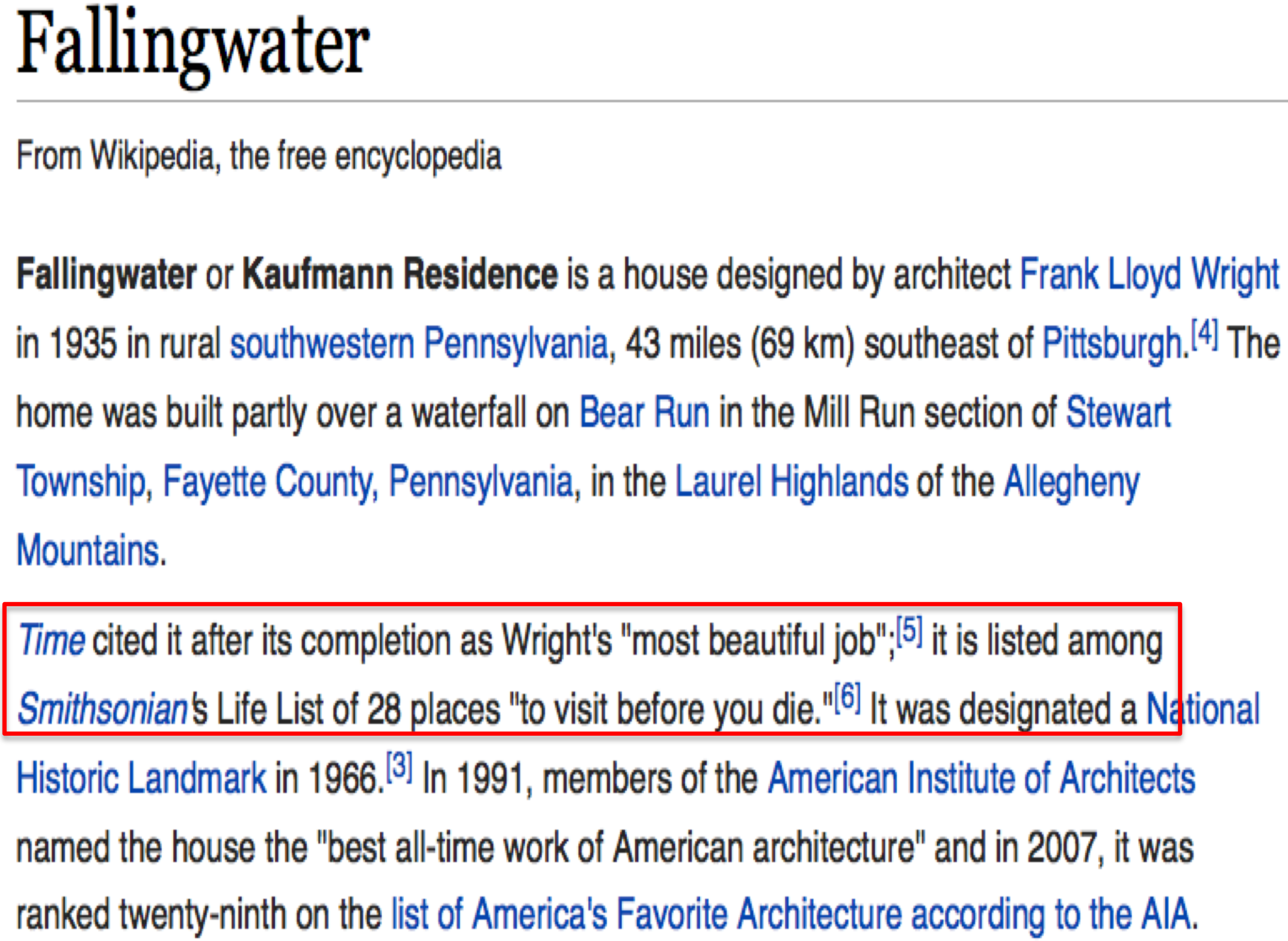
Source: https://en.wikipedia.org/wiki/World_Trade_Center_station_(PATH)

Question: What Is This?



www.GreatBuildings.com

Answer: Masterpiece of Another Famous Architect

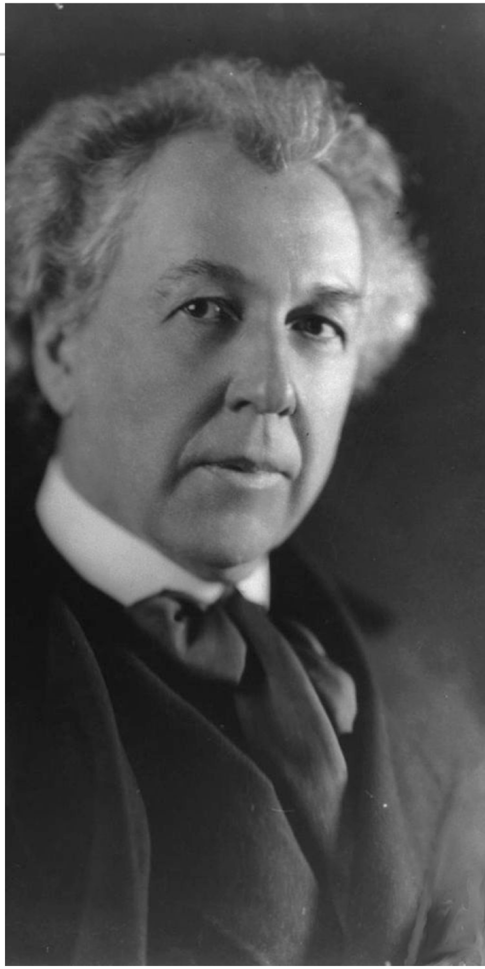


Fallingwater

From Wikipedia, the free encyclopedia

Fallingwater or **Kaufmann Residence** is a house designed by architect Frank Lloyd Wright in 1935 in rural southwestern Pennsylvania, 43 miles (69 km) southeast of Pittsburgh.[4] The home was built partly over a waterfall on Bear Run in the Mill Run section of Stewart Township, Fayette County, Pennsylvania, in the Laurel Highlands of the Allegheny Mountains.

Time cited it after its completion as Wright's "most beautiful job";[5] it is listed among *Smithsonian*'s Life List of 28 places "to visit before you die."[6] It was designated a National Historic Landmark in 1966.[3] In 1991, members of the American Institute of Architects named the house the "best all-time work of American architecture" and in 2007, it was ranked twenty-ninth on the list of America's Favorite Architecture according to the AIA.



Source: https://en.wikipedia.org/wiki/Fallingwater

Your First Comp Arch Assignment

- Go and visit Bahnhof Stadelhofen
  - Extra credit: Repeat for Oculus
  - Extra+ credit: Repeat for Fallingwater

- Appreciate the beauty & out-of-the-box and creative thinking
- Think about tradeoffs in the design of the Bahnhof
  - Strengths, weaknesses, goals of design
- Derive principles on your own for good design and innovation

- Due date: **Any time during this course**
  - Later during the course is better
  - Apply what you have learned in this course
  - Think out-of-the-box

But First, Today's First Assignment

- Find The Differences Of This and That

Find The Differences of This and That

This

Source: By Toni_V from Zurich, Switzerland - Stadelhofen2, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=4087256

That

Source: http://cookiemagik.deviantart.com/art/Train-station-207266944

Many Tradeoffs Between Two Designs

- You can list them after you complete the first assignment…

Aside: Evaluation Criteria for the Designs

- Functionality (Does it meet the specification?)
- Reliability
- Space requirement
- Cost
- Expandability
- Comfort level of users
- Happiness level of users
- Aesthetics
- …

- How to evaluate goodness of design is always a critical question.

A Key Question

- How was Calavatra able to design especially his key buildings?
- Can have many guesses
  - (Ultra) hard work, perseverance, dedication (over decades)
  - Experience
  - Creativity, Out-of-the-box thinking
  - A good understanding of past designs
  - Good judgment and intuition
  - Strong skill combination (math, architecture, art, engineering, …)
  - Funding ($$$$), luck, initiative, entrepreneurialism
  - Strong understanding of and commitment to fundamentals
  - Principled design
  - …
- (You will be exposed to and hopefully develop/enhance many of these skills in this course)

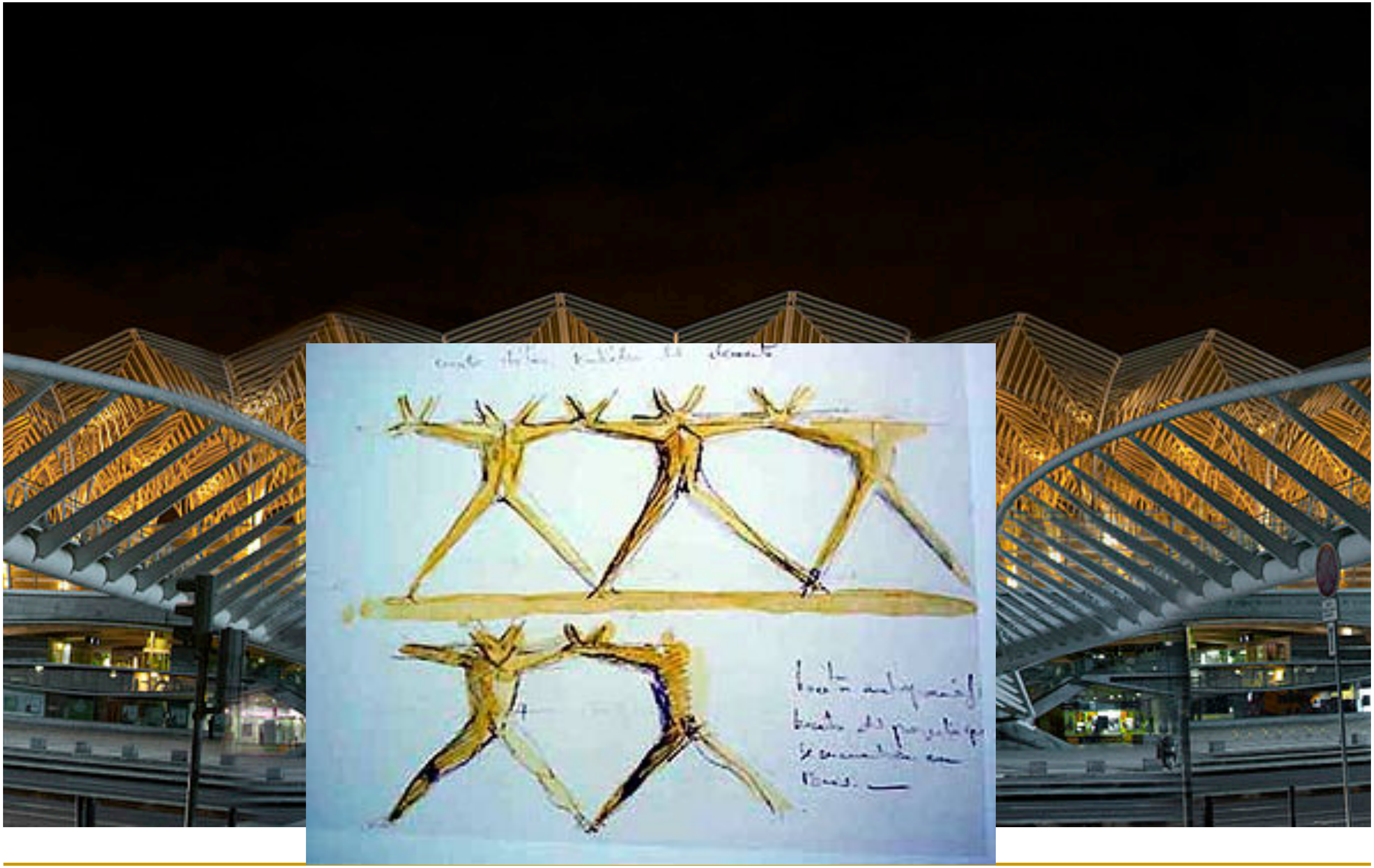
Principled Design

- "To me, there are two overriding principles to be found in nature which are most appropriate for building:
  - one is the optimal use of material,
  - the other the capacity of organisms to change shape, to grow, and to move."
  - *Santiago Calatrava*

- "Calatrava's constructions are inspired by natural forms like plants, bird wings, and the human body."

Source: http://www.arcspace.com/exhibitions/unsorted/santiago-calatrava/

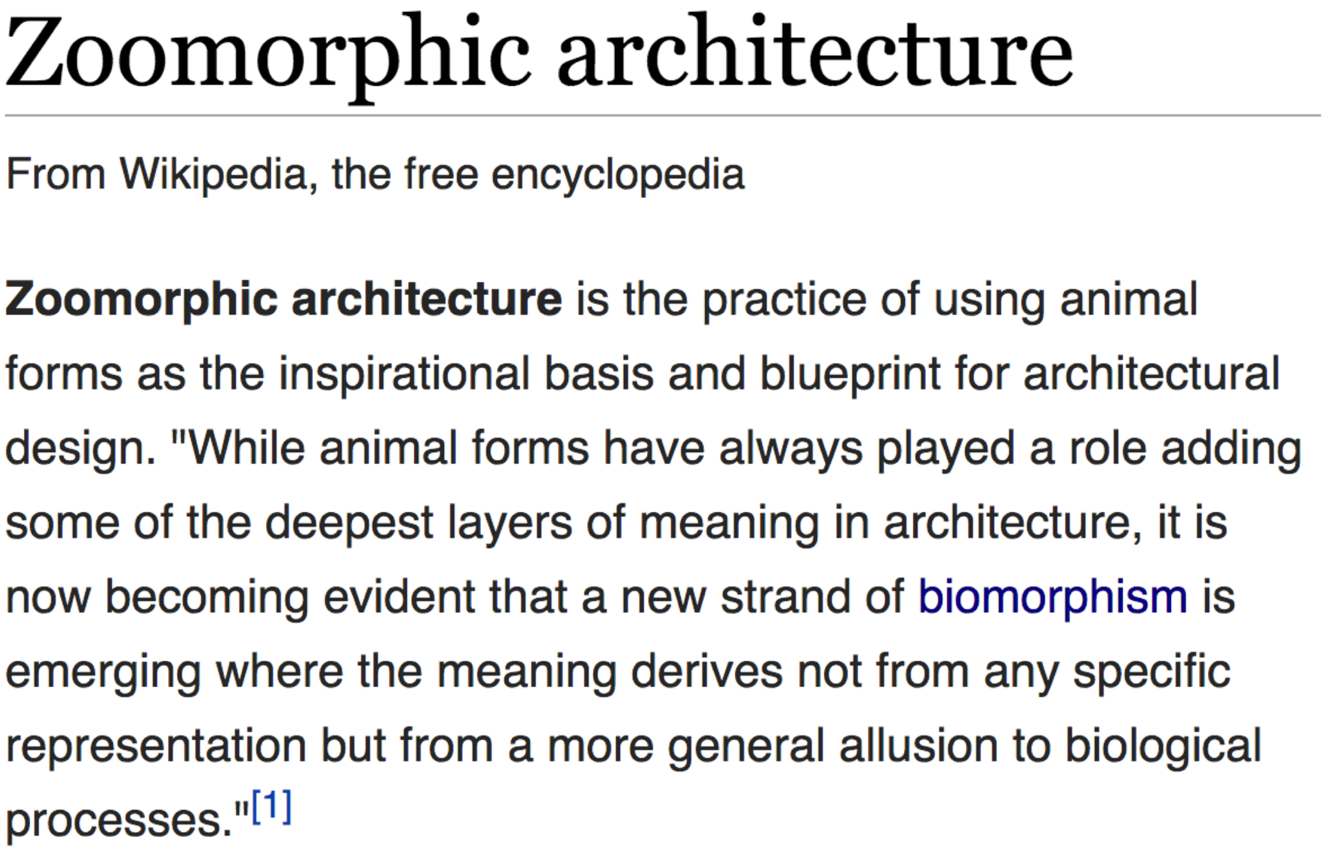
Gare do Oriente, Lisbon, Revisited



Source: By Martín Gómez Tagle - Lisbon, Portugal, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=13764903

Source: http://www.arcspace.com/exhibitions/unsorted/santiago-calatrava/

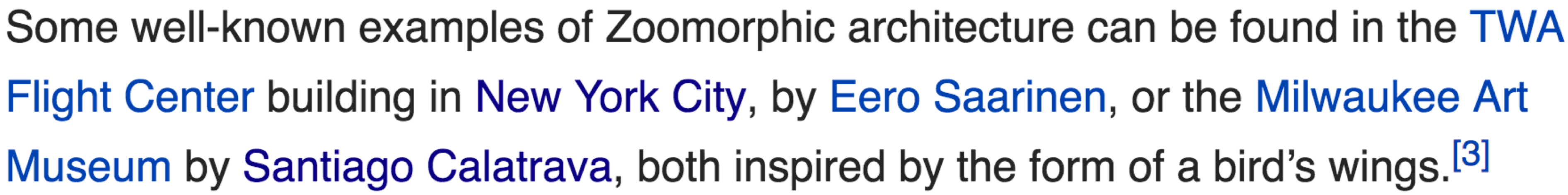
A Principled Design



Zoomorphic architecture

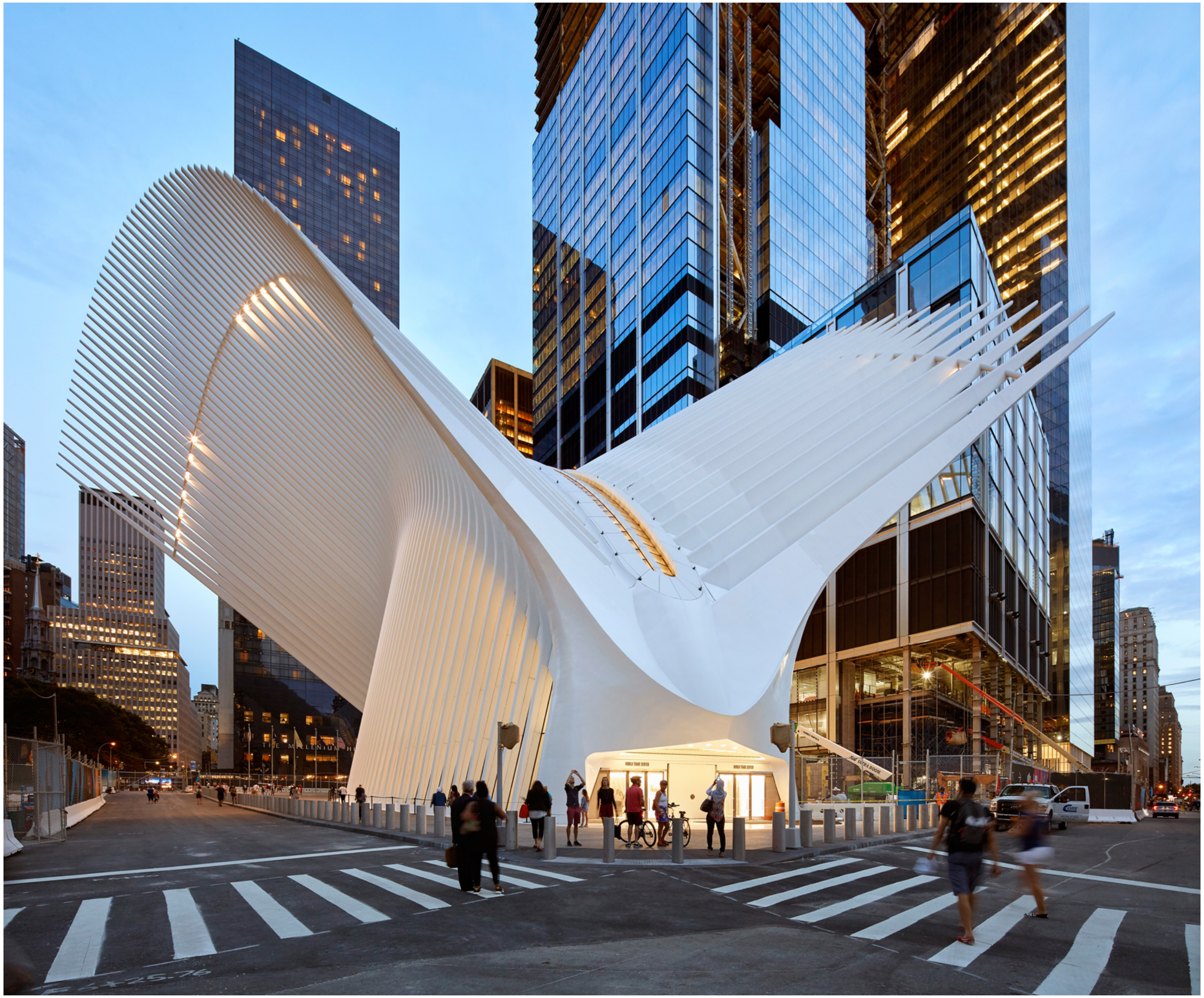
From Wikipedia, the free encyclopedia

Zoomorphic architecture is the practice of using animal forms as the inspirational basis and blueprint for architectural design. "While animal forms have always played a role adding some of the deepest layers of meaning in architecture, it is now becoming evident that a new strand of biomorphism is emerging where the meaning derives not from any specific representation but from a more general allusion to biological processes."[1]



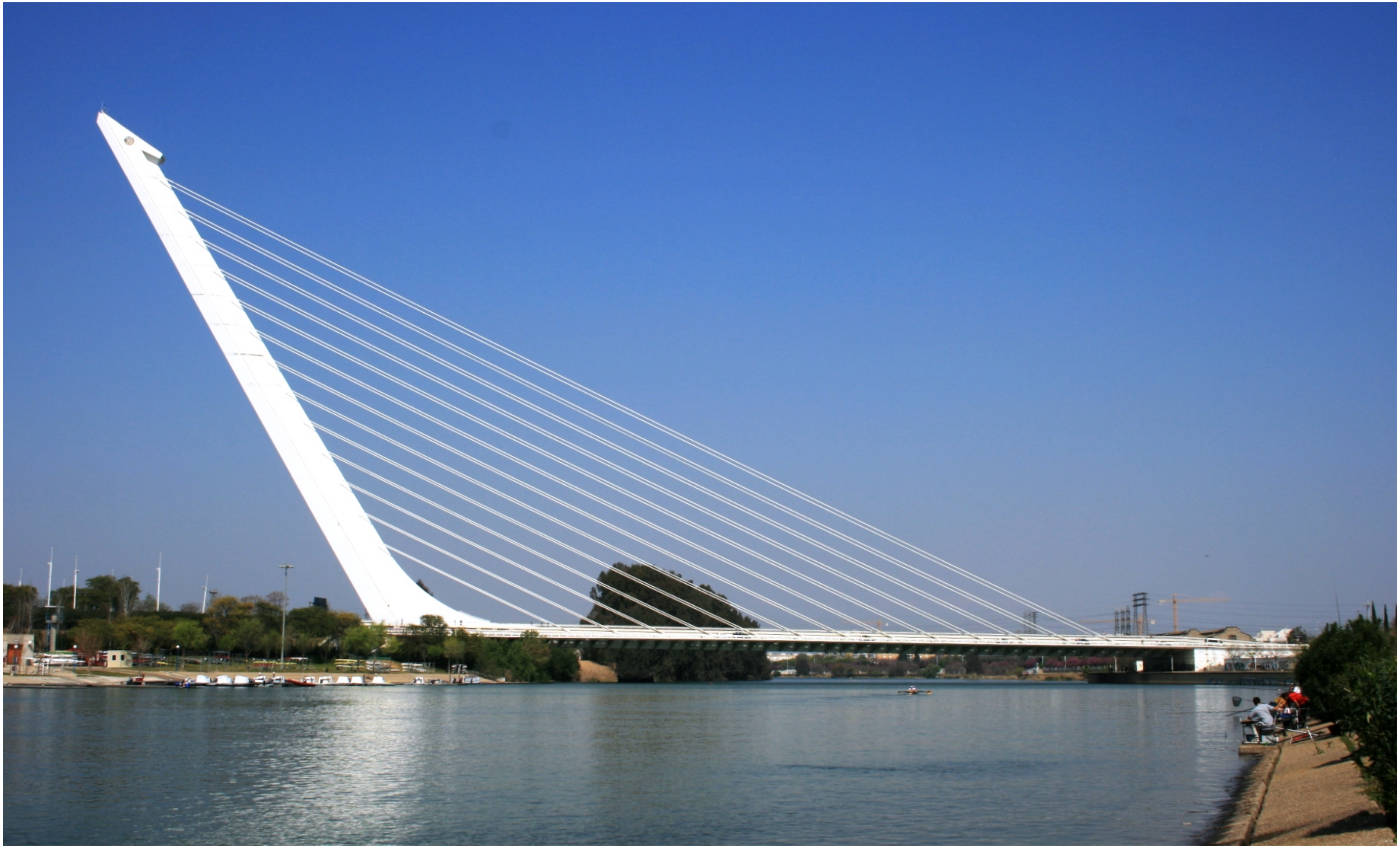
Some well-known examples of Zoomorphic architecture can be found in the TWA Flight Center building in New York City, by Eero Saarinen, or the Milwaukee Art Museum by Santiago Calatrava, both inspired by the form of a bird's wings.[3]

What Does This Remind You Of?



Source: https://www.dezeen.com/2016/08/29/santiago-calatrava-oculus-world-trade-center-transportation-hub-new-york-photographs-hufton-crow/

What About This?



Source: De Galván - Puente del Alamillo.jpg on Enciclopedia.us.es, GFDL, https://commons.wikimedia.org/w/index.php?curid=15026095

A Quote from The Other Famous Architect

- "architecture […] based upon principle, and not upon precedent" (Frank Lloyd Wright)

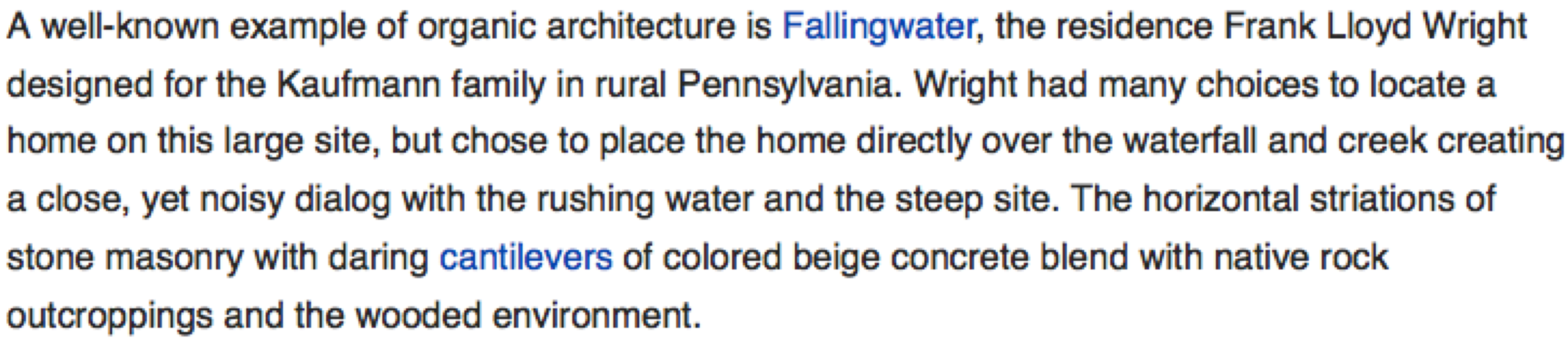
Source: http://www.fallingwater.org/

A Principled Design

Organic architecture

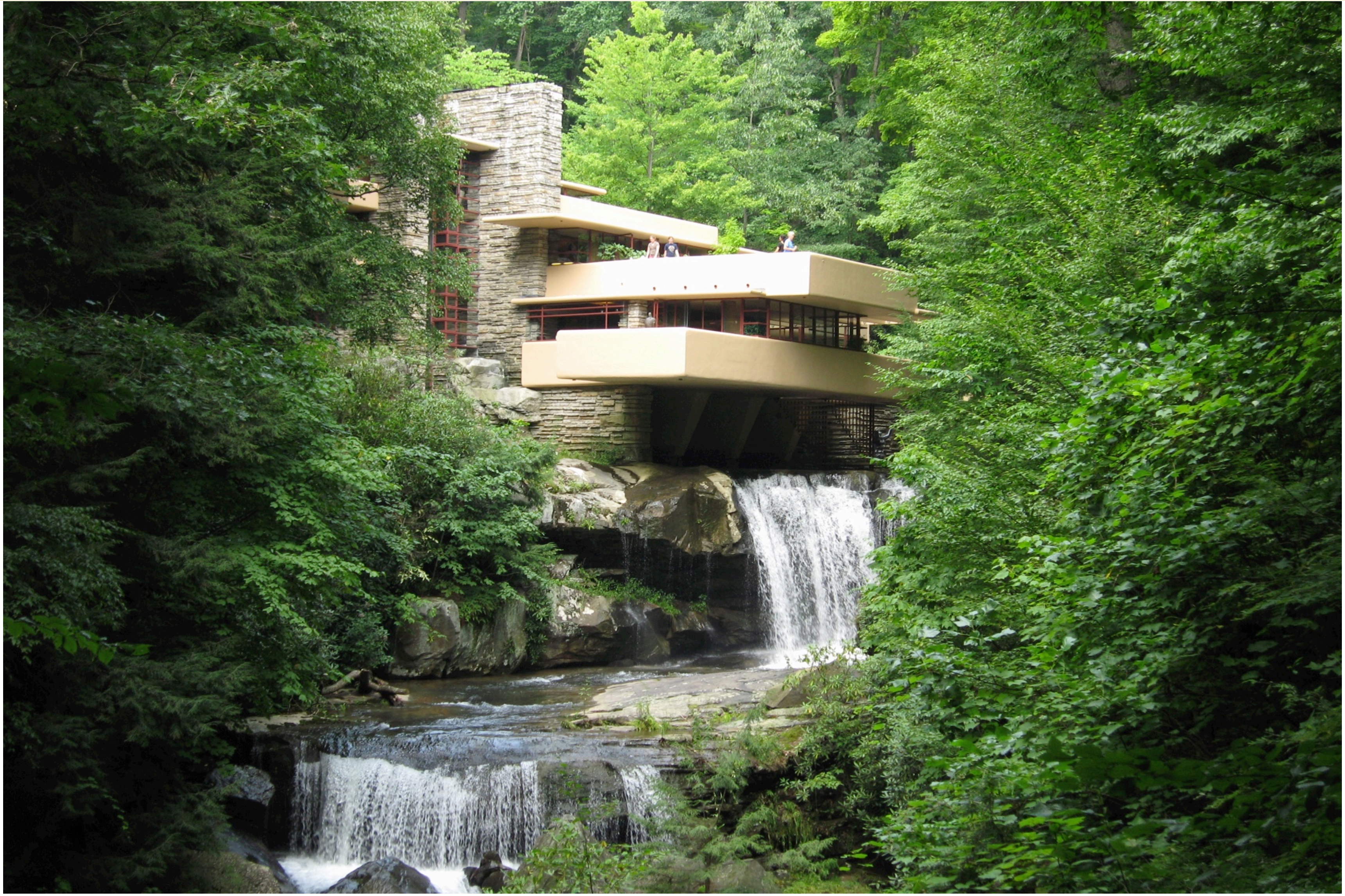
From Wikipedia, the free encyclopedia

Organic architecture is a philosophy of architecture which promotes harmony between human habitation and the natural world through design approaches so sympathetic and well integrated with its site, that buildings, furnishings, and surroundings become part of a unified, interrelated composition.



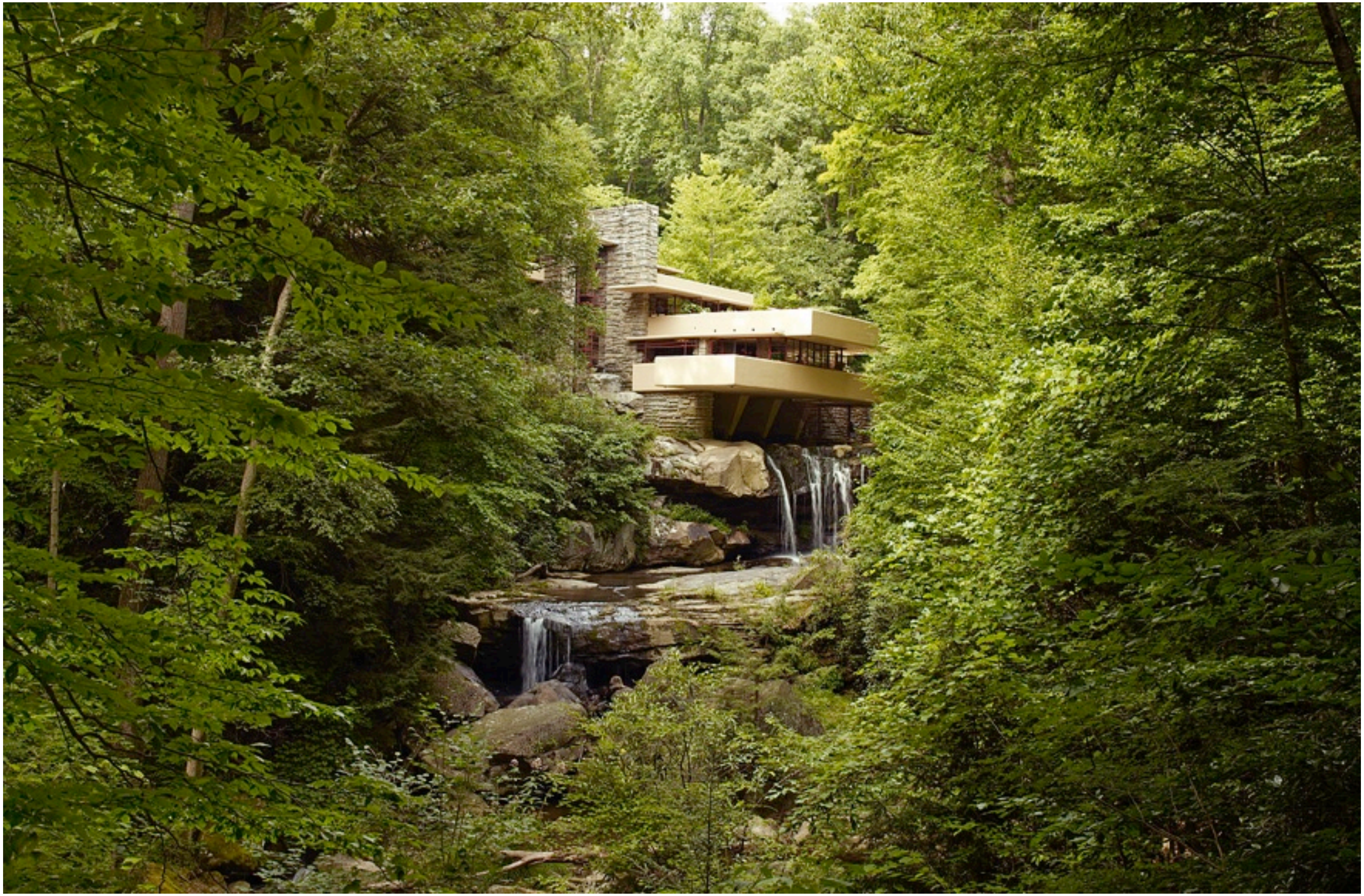
A well-known example of organic architecture is Fallingwater, the residence Frank Lloyd Wright designed for the Kaufmann family in rural Pennsylvania. Wright had many choices to locate a home on this large site, but chose to place the home directly over the waterfall and creek creating a close, yet noisy dialog with the rushing water and the steep site. The horizontal striations of stone masonry with daring cantilevers of colored beige concrete blend with native rock outcroppings and the wooded environment.

Another View



Source: https://roadtrippers.com/stories/falling-water

Yet Another View



Source: By Carol M. Highsmith - http://www.loc.gov/pictures/collection/highsm/item/2010630255/, Public Domain, https://commons.wikimedia.org/w/index.php?curid=29385254

www.GreatBuildings.com

Major High-Level Goals of This Course

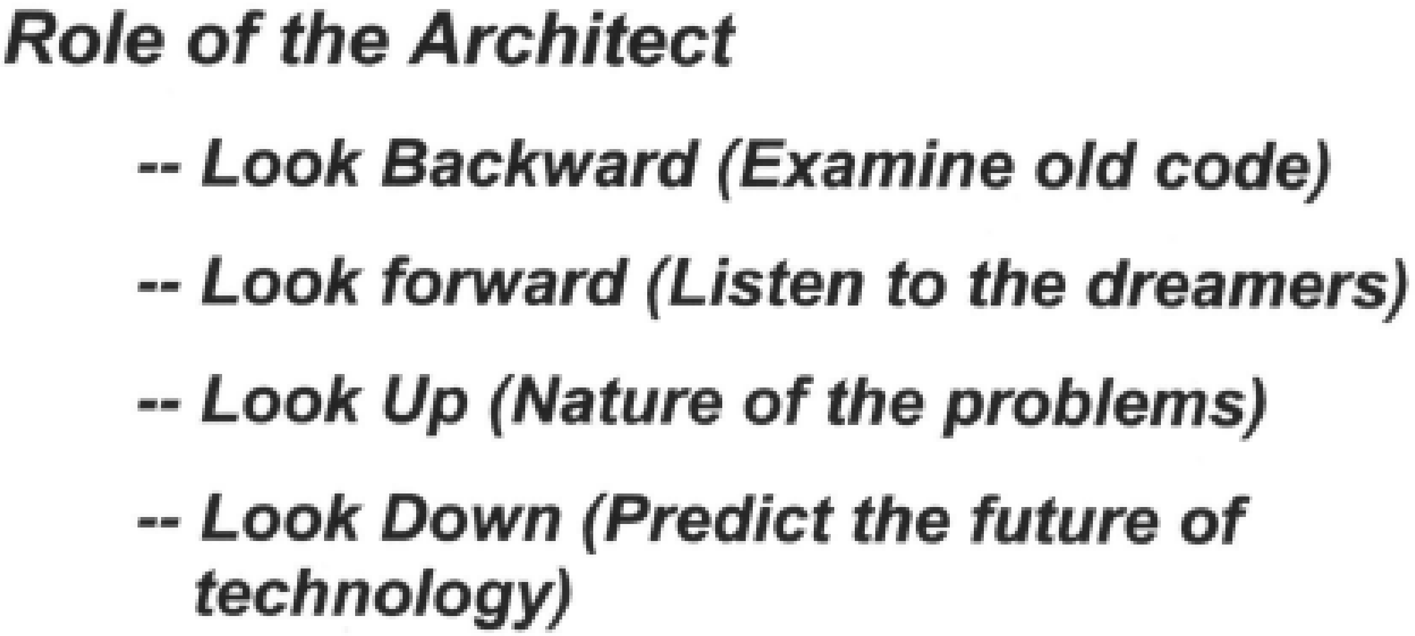
- Understand the principles
- Understand the precedents

- Based on such understanding:
  - Enable you to evaluate tradeoffs of different designs and ideas
  - Enable you to develop principled designs
  - Enable you to develop novel, out-of-the-box designs

- The focus is on:
  - Principles, precedents, and how to use them for new designs

- In Computer Architecture

Role of the (Computer) Architect



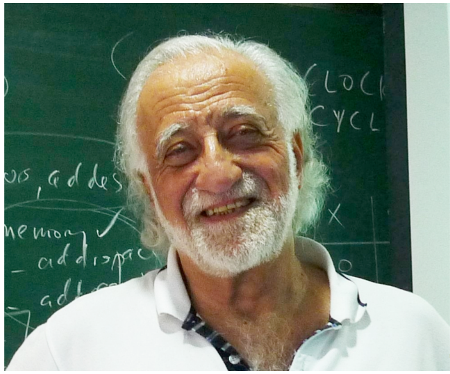
Role of the Architect

-- Look Backward (Examine old code)

-- Look forward (Listen to the dreamers)

-- Look Up (Nature of the problems)

-- Look Down (Predict the future of technology)



from Yale Patt's lecture notes

Role of The (Computer) Architect

- Look backward (to the past)
  - Understand tradeoffs and designs, upsides/downsides, past workloads. Analyze and evaluate the past.
- Look forward (to the future)
  - Be the dreamer and create new designs. Listen to dreamers.
  - Push the state of the art. Evaluate new design choices.
- Look up (towards problems in the computing stack)
  - Understand important problems and their nature.
  - Develop architectures and ideas to solve important problems.
- Look down (towards device/circuit technology)
  - Understand the capabilities of the underlying technology.
  - Predict and adapt to the future of technology (you are designing for N years ahead). Enable the future technology.

Takeaways

- Being an architect is not easy
- You need to consider **many** things in designing a new system + have good intuition/insight into ideas/tradeoffs

- But, it is fun and can be very rewarding
- And, enables a great future
  - E.g., many scientific and everyday-life innovations would not have been possible without architectural innovation that enabled very high performance systems
  - E.g., your mobile phones
  - E.g., self-driving vehicles

- This course will enable you to become a good computer architect

So, I Hope You Are Here for This

Systems Prog.

"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

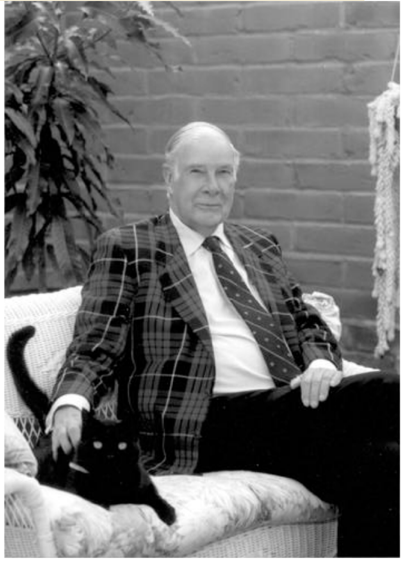
HW designer's view of how a computer system works

Digital logic as a model of computation

Digital Design

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?

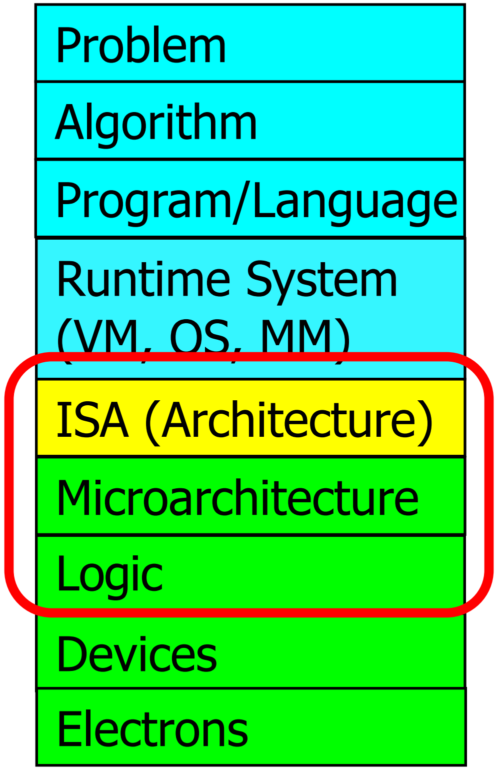


Algorithm

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



Problem

Algorithm

Program/Language

Runtime System (VM, OS, MM)

ISA (Architecture)

Microarchitecture

Logic

Devices

Electrons

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)

Aside: A Famous Work By Hamming

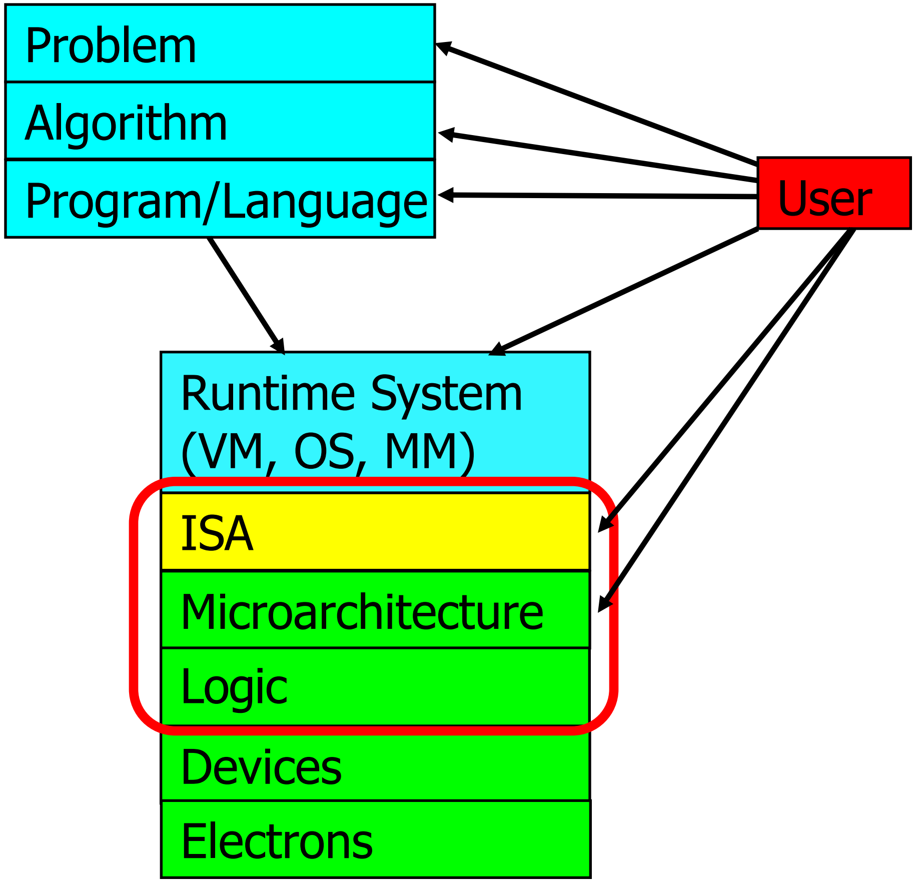
- Hamming, "Error Detecting and Error Correcting Codes," Bell System Technical Journal 1950.

- Introduced the concept of Hamming distance
  - number of locations in which the corresponding symbols of two equal-length strings is different
- Developed a theory of codes used for error detection and correction

- Also see:
  - Hamming, "You and Your Research," Talk at Bell Labs, 1986.
  - http://www.cs.virginia.edu/~robins/YouAndYourResearch.html

Levels of Transformation, Revisited

- A user-centric view: computer designed for users



- The entire stack should be optimized for user

The Power of Abstraction

- Levels of transformation create abstractions
  - Abstraction: A higher level only needs to know about the interface to the lower level, not how the lower level is implemented
  - E.g., high-level language programmer does not really need to know what the ISA is and how a computer executes instructions

- Abstraction improves productivity
  - No need to worry about decisions made in underlying levels
  - E.g., programming in Java vs. C vs. assembly vs. binary vs. by specifying control signals of each transistor every cycle

- Then, why would you want to know what goes on underneath or above?

Crossing the Abstraction Layers

- As long as everything goes well, not knowing what happens underneath (or above) is not a problem.

- What if
  - The program you wrote is running slow?
  - The program you wrote does not run correctly?
  - The program you wrote consumes too much energy?
  - Your system just shut down and you have no idea why?
  - Someone just compromised your system and you have no idea how?

- What if
  - The hardware you designed is too hard to program?
  - The hardware you designed is too slow because it does not provide the right primitives to the software?

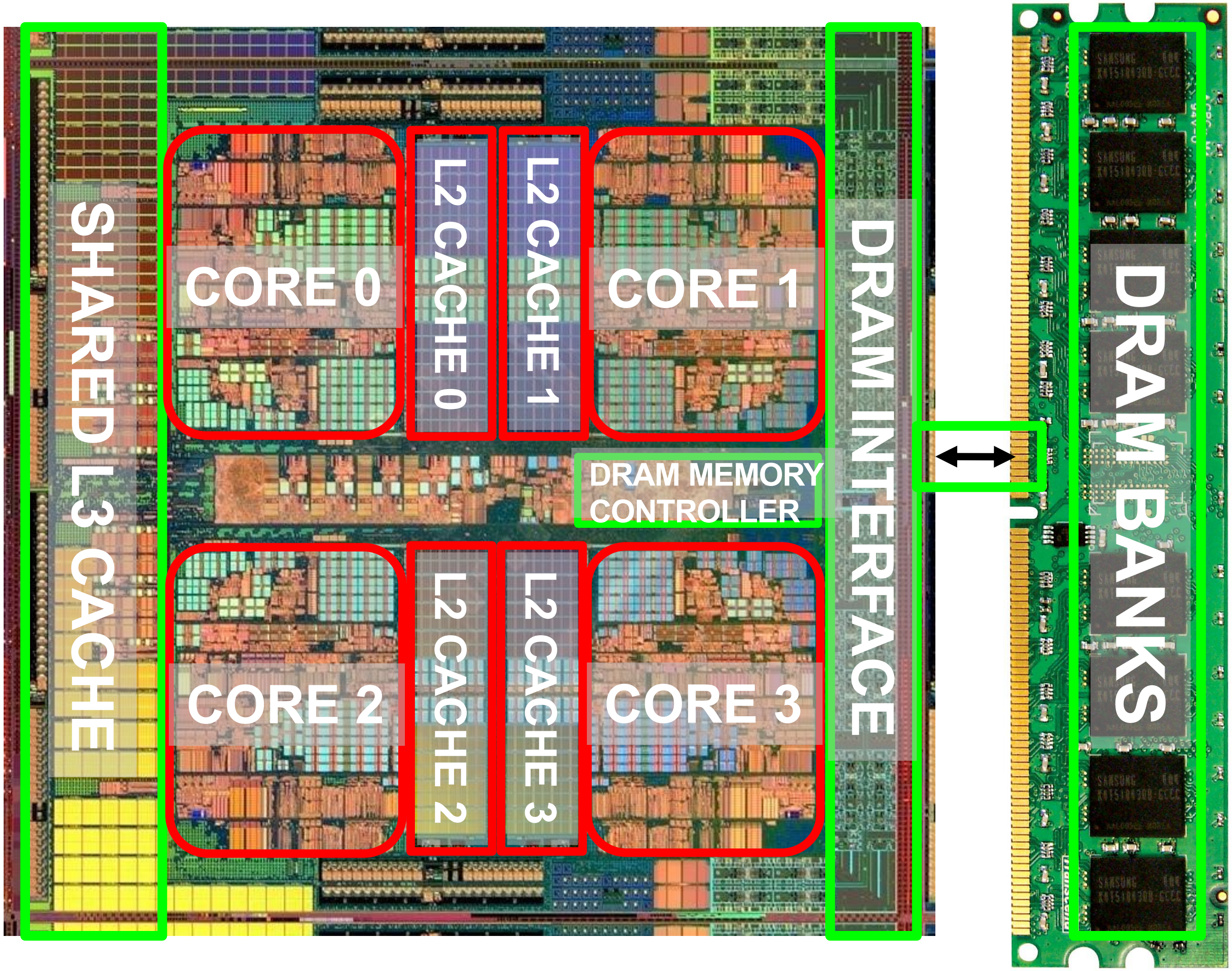
- What if
  - You want to design a much more efficient and higher performance system?

Crossing the Abstraction Layers

- Two key goals of this course are
  - to understand how a processor works underneath the software layer and how decisions made in hardware affect the software/programmer
  - to enable you to be comfortable in making design and optimization decisions that cross the boundaries of different layers and system components

An Example: Multi-Core Systems

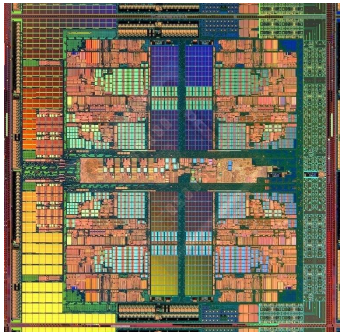
Multi-Core Chip



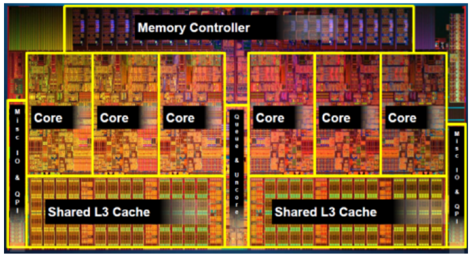
*Die photo credit: AMD Barcelona

A Trend: Many Cores on Chip

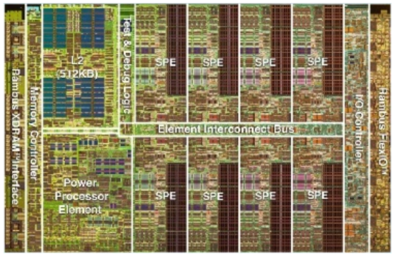
- Simpler and lower power than a single large core
- Parallel processing on single chip → faster, new applications



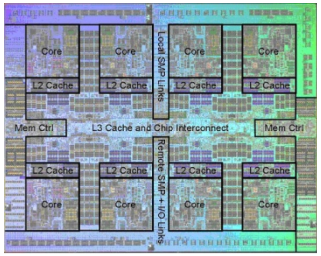
AMD Barcelona
4 cores



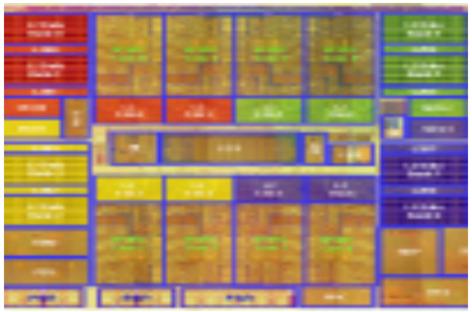
Intel Core i7
8 cores



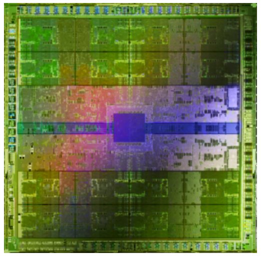
IBM Cell BE
8+1 cores



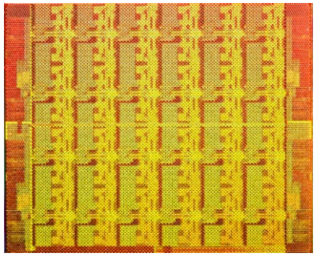
IBM POWER7
8 cores



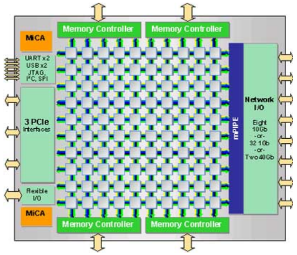
Sun Niagara II
8 cores



Nvidia Fermi
448 “cores”



Intel SCC
48 cores, networked



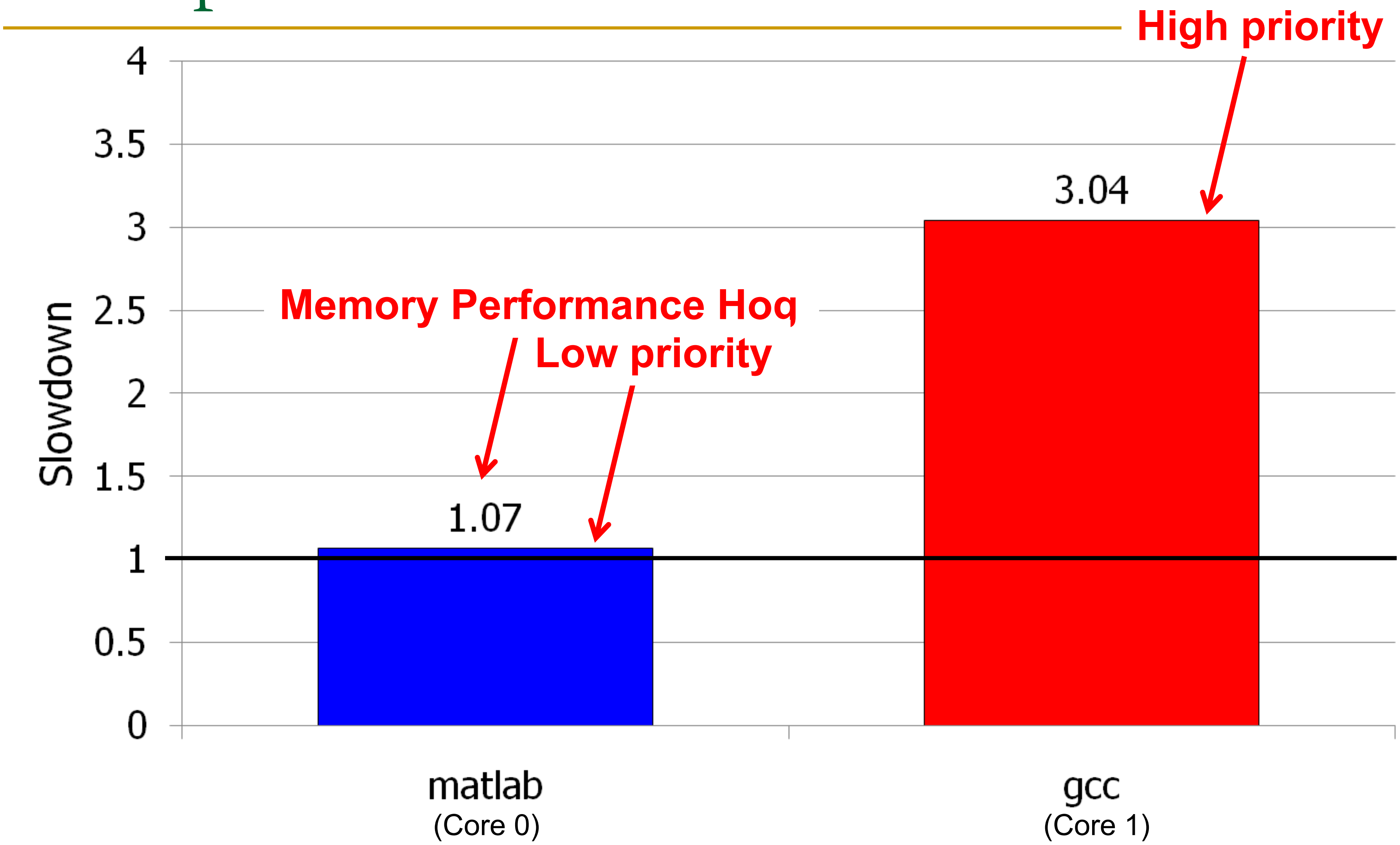
Tilera TILE Gx
100 cores, networked

Many Cores on Chip

- What we want:
  - N times the system performance with N times the cores

- What do we get today?

Unexpected Slowdowns in Multi-Core



Moscibroda and Mutlu, "Memory performance attacks: Denial of memory service in multi-core systems," USENIX Security 2007.

Three Questions

- Can you figure out why the applications slow down if you do not know the underlying system and how it works?

- Can you figure out why there is a disparity in slowdowns if you do not know how the system executes the programs?

- Can you fix the problem without knowing what is happening "underneath"?

Three Questions: Rephrased & Concise

- Why is there any slowdown?

- Why is there a disparity in slowdowns?

- How can we solve the problem if we do not want that disparity?

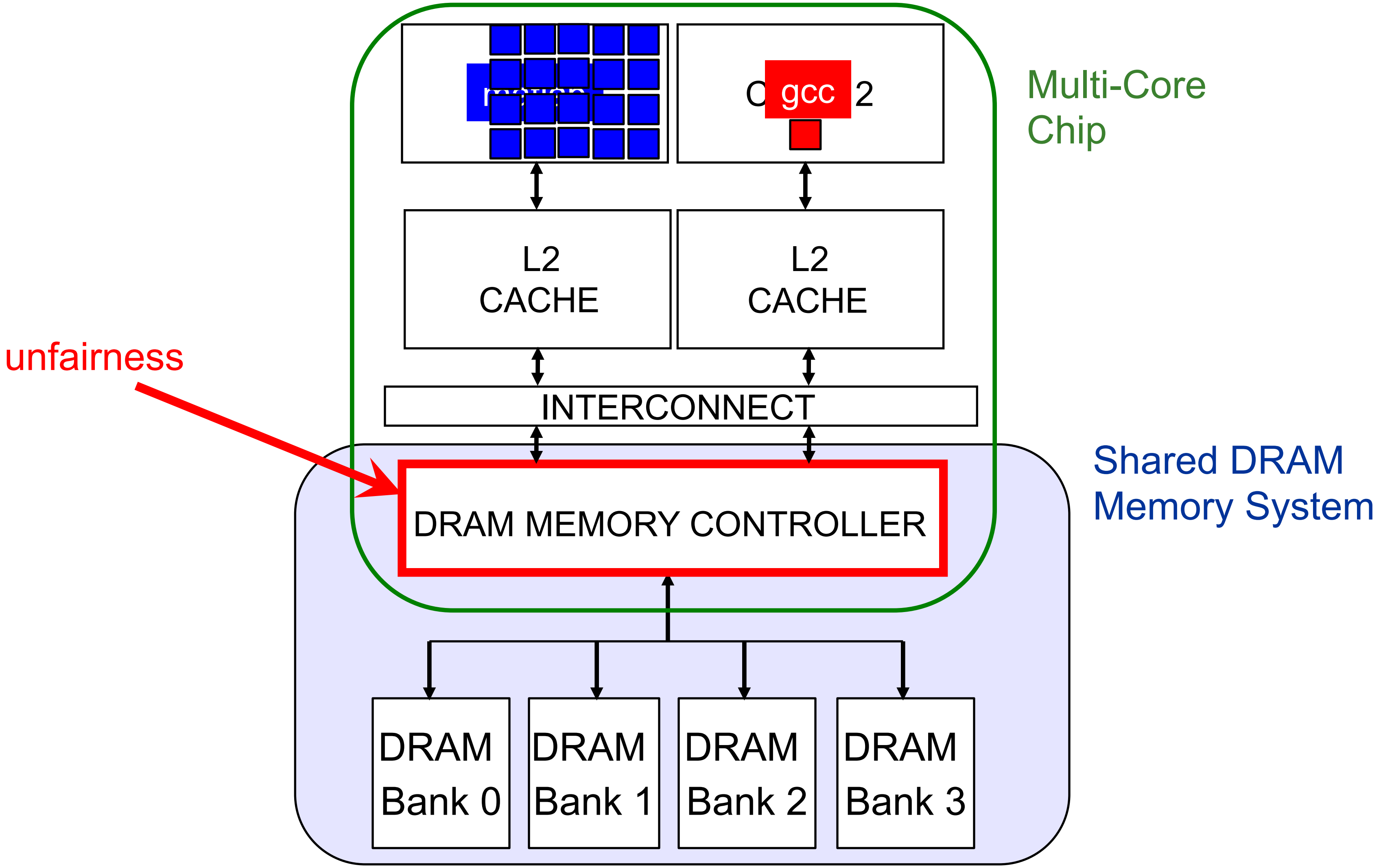
Why Is This Important?

- We want to execute applications in parallel in multi-core systems → consolidate more and more (for efficiency)
  - Cloud computing
  - Mobile phones
  - Automotive systems

- We want to mix different types of applications together
  - those requiring QoS guarantees (e.g., video, pedestrian detection)
  - those that are important but less so
  - those that are less important

- We want the system to be controllable **and** high performance

Why the Disparity in Slowdowns?



C gcc 2

L2 CACHE

L2 CACHE

INTERCONNECT

DRAM MEMORY CONTROLLER

DRAM Bank 0

DRAM Bank 1

DRAM Bank 2

DRAM Bank 3

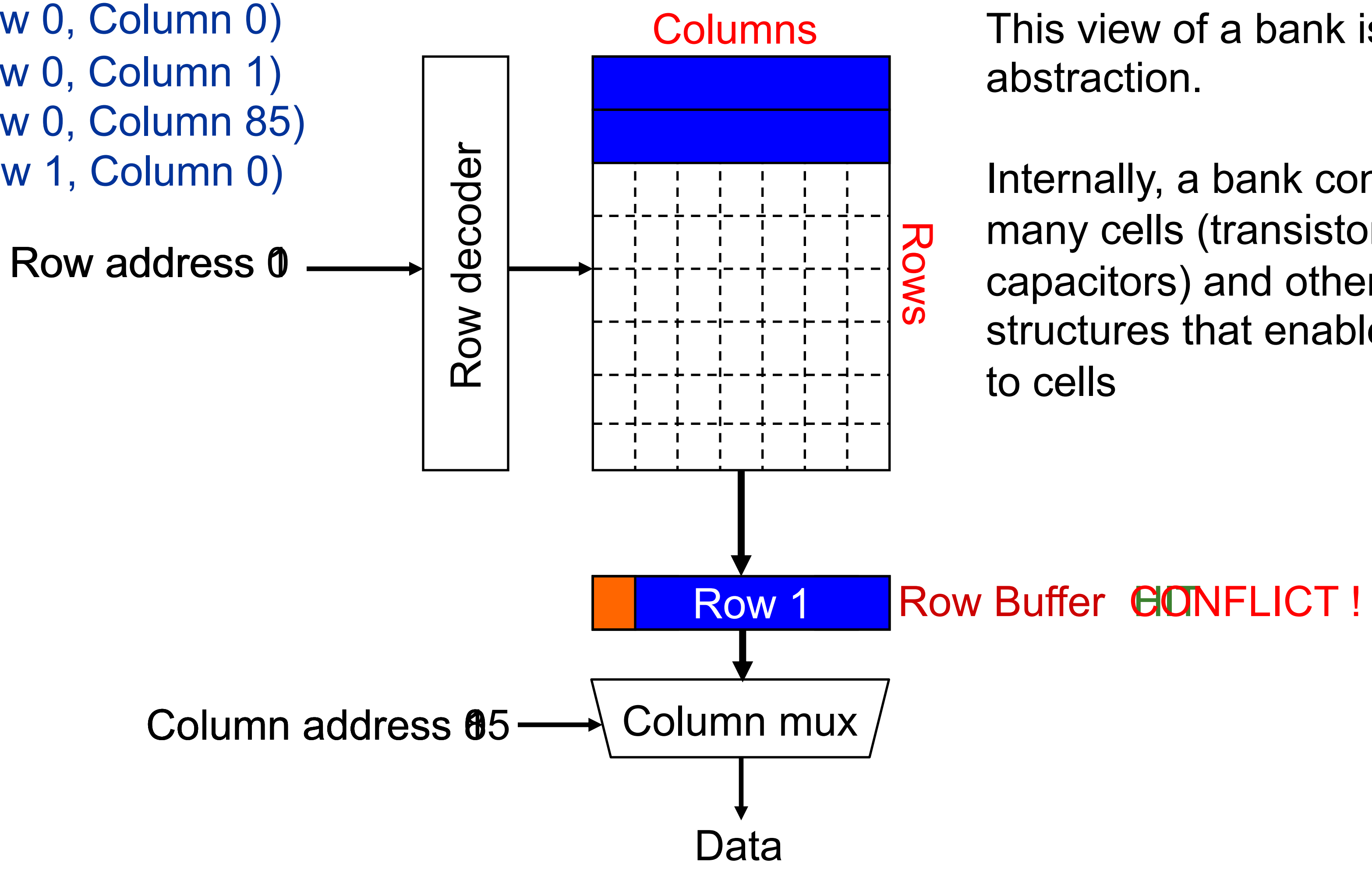
Multi-Core Chip

Shared DRAM Memory System

unfairness

Digging Deeper: DRAM Bank Operation

Access Address:
(Row 0, Column 0)
(Row 0, Column 1)
(Row 0, Column 85)
(Row 1, Column 0)



This view of a bank is an abstraction.

Internally, a bank consists of many cells (transistors & capacitors) and other structures that enable access to cells

DRAM Controllers

- A row-conflict memory access takes significantly longer than a row-hit access

- Current controllers take advantage of this fact

- Commonly used scheduling policy (FR-FCFS) [Rixner 2000]*

  (1) Row-hit first: Service row-hit memory accesses first

  (2) Oldest-first: Then service older accesses first

- This scheduling policy aims to maximize DRAM throughput

*Rixner et al., "Memory Access Scheduling," ISCA 2000.
*Zuravleff and Robinson, "Controller for a synchronous DRAM …," US Patent 5,630,096, May 1997.

The Problem

- Multiple applications share the DRAM controller
- DRAM controllers designed to maximize DRAM data throughput

- DRAM scheduling policies are unfair to some applications
  - Row-hit first: unfairly prioritizes apps with high row buffer locality
    - Threads that keep on accessing the same row
  - Oldest-first: unfairly prioritizes memory-intensive applications

- DRAM controller vulnerable to denial of service attacks
  - Can write programs to exploit unfairness

A Memory Performance Hog

```
// initialize large arrays A, B

for (j=0; j<N; j++) {
    index = j*linesize;
    A[index] = B[index];
    …
}
```

streaming (in sequence)

```
// initialize large arrays A, B

for (j=0; j<N; j++) {
    index = rand();
    A[index] = B[index];
    …
}
```

random

STREAM

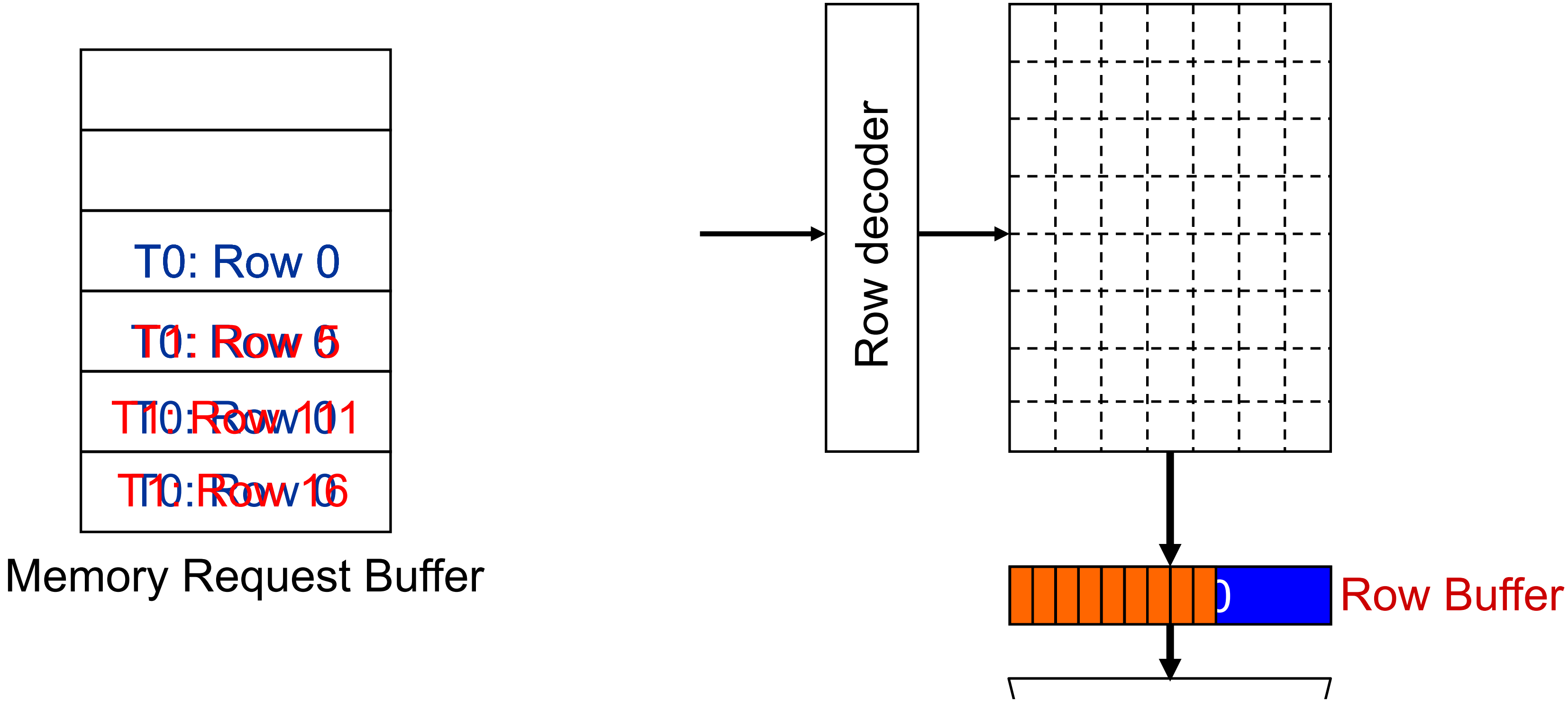
- Sequential memory access
- Very high row buffer locality (96% hit rate)
- Memory intensive

RANDOM

- Random memory access
- Very low row buffer locality (3% hit rate)
- Similarly memory intensive

Moscibroda and Mutlu, “Memory Performance Attacks,” USENIX Security 2007.

What Does the Memory Hog Do?



Row size: 8KB, request size: 64B

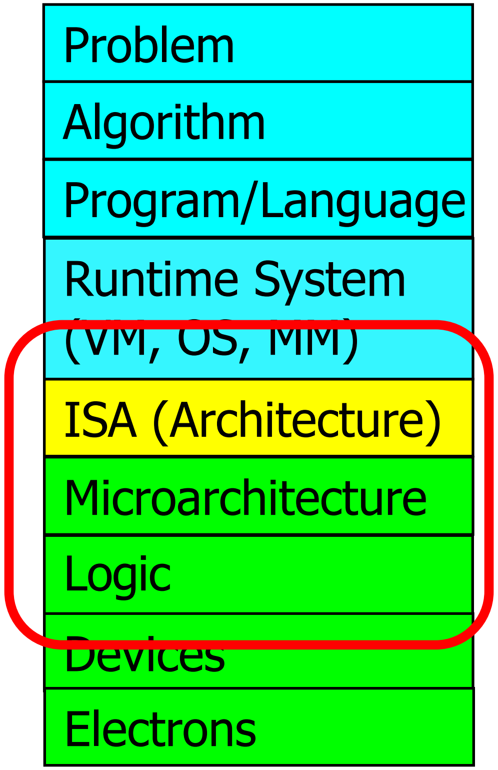
128 (8KB/64B) requests of STREAM serviced before a single request of RANDOM

Moscibroda and Mutlu, "Memory Performance Attacks," USENIX Security 2007.

Now That We Know What Happens Underneath

- How would you solve the problem?

- What is the right place to solve the problem?
  - Programmer?
  - System software?
  - Compiler?
  - Hardware (Memory controller)?
  - Hardware (DRAM)?
  - Circuits?

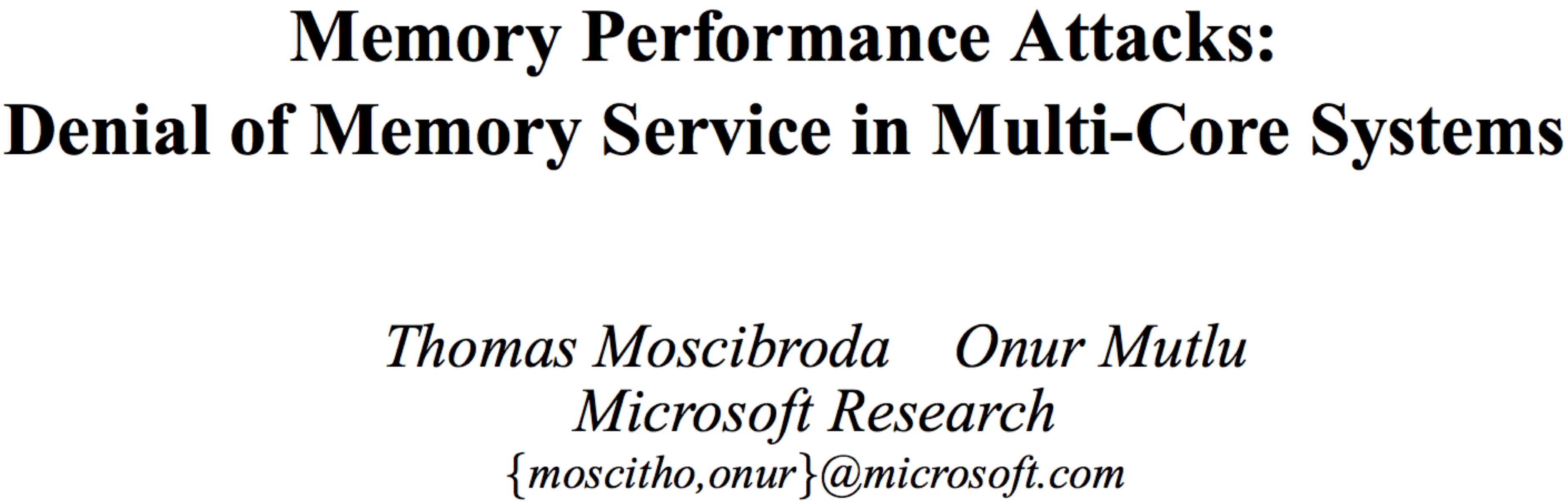


- Two other goals of this course:
  - Enable you to think critically
  - Enable you to think broadly

Reading on Memory Performance Attacks

- Thomas Moscibroda and Onur Mutlu,
  "Memory Performance Attacks: Denial of Memory Service in Multi-Core Systems"
  *Proceedings of the 16th USENIX Security Symposium (**USENIX SECURITY**)*, pages 257-274, Boston, MA, August 2007. Slides (ppt)

- One potential reading for your Homework 1 assignment



Memory Performance Attacks:
Denial of Memory Service in Multi-Core Systems

Thomas Moscibroda   Onur Mutlu
Microsoft Research
{moscitho,onur}@microsoft.com

If You Are Interested … Further Readings

- Onur Mutlu and Thomas Moscibroda,
  "Stall-Time Fair Memory Access Scheduling for Chip Multiprocessors"
  Proceedings of the 40th International Symposium on Microarchitecture (***MICRO***), pages 146-158, Chicago, IL, December 2007. Slides (ppt)

- Onur Mutlu and Thomas Moscibroda,
  "Parallelism-Aware Batch Scheduling: Enhancing both Performance and Fairness of Shared DRAM Systems"
  Proceedings of the 35th International Symposium on Computer Architecture (***ISCA***) [Slides (ppt)]

- Sai Prashanth Muralidhara, Lavanya Subramanian, Onur Mutlu, Mahmut Kandemir, and Thomas Moscibroda,
  "Reducing Memory Interference in Multicore Systems via Application-Aware Memory Channel Partitioning"
  Proceedings of the 44th International Symposium on Microarchitecture (***MICRO***), Porto Alegre, Brazil, December 2011. Slides (pptx)

Takeaway

Breaking the abstraction layers (between components and transformation hierarchy levels)

and knowing what is underneath

enables you to **understand** and **solve** problems

Computer Architecture
Lecture 1: Introduction and Basics

Prof. Onur Mutlu

ETH Zürich

Fall 2019

19 September 2019

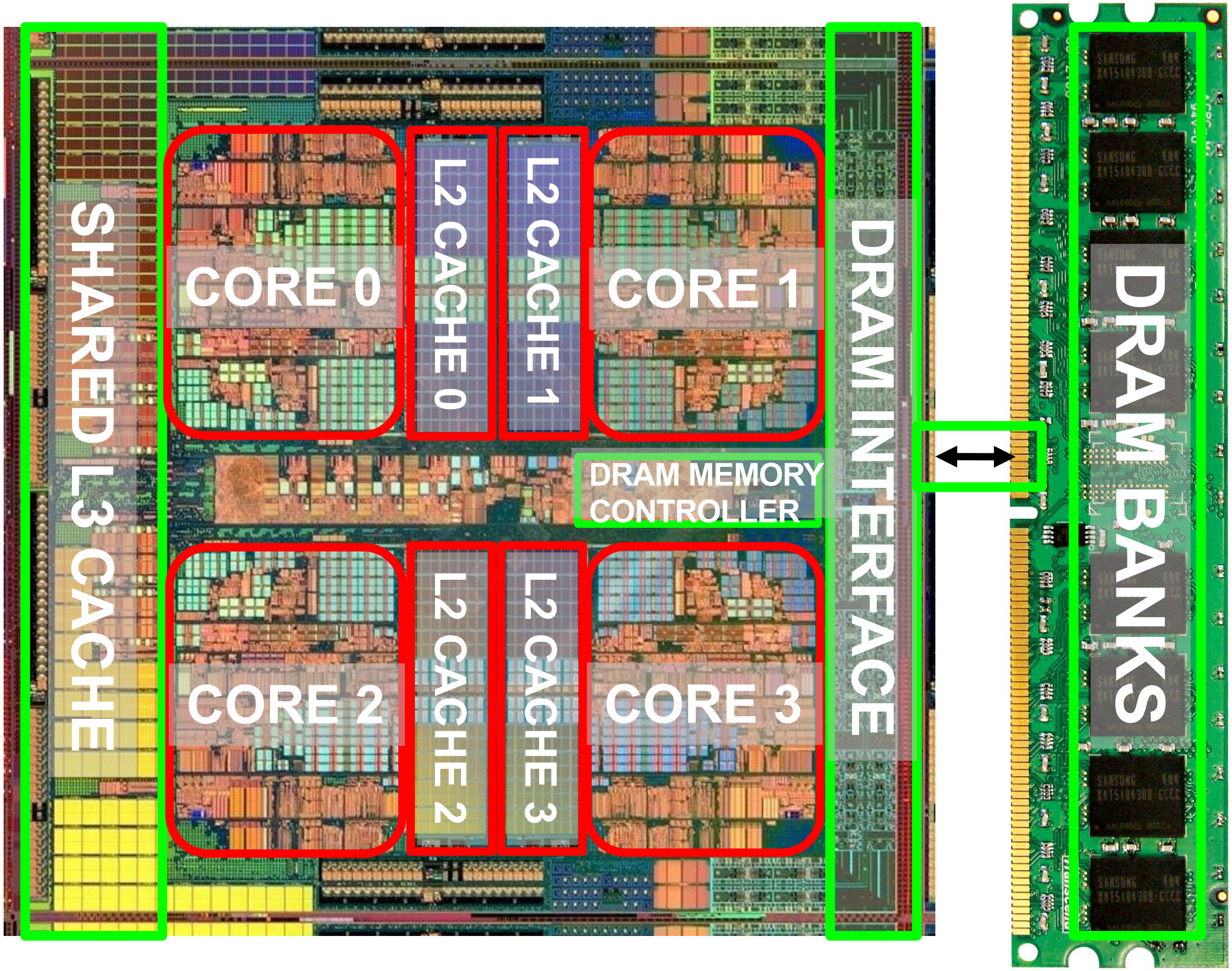
We Did Not Cover These Slides. They Are For Your Benefit.

Another Example

- DRAM Refresh

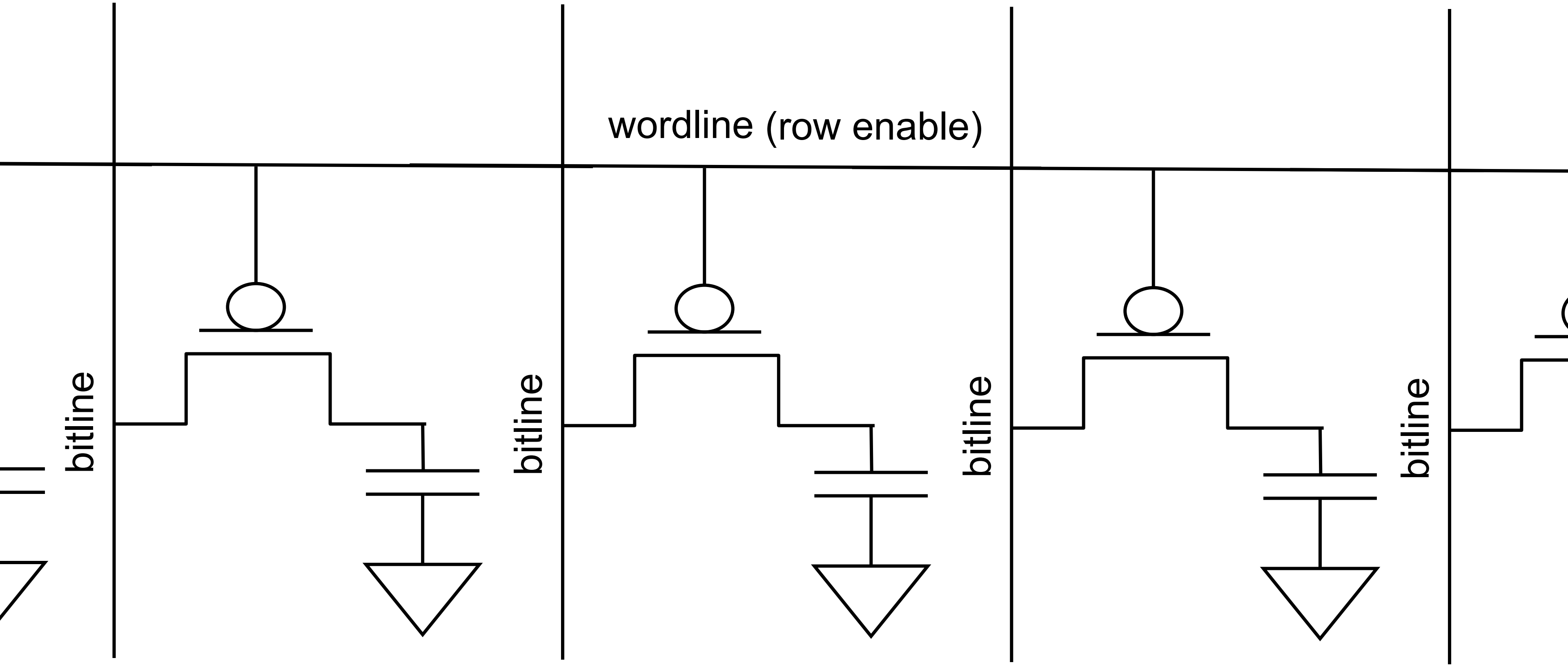
DRAM in the System

Multi-Core Chip



*Die photo credit: AMD Barcelona

A DRAM Cell



- A DRAM cell consists of a capacitor and an access transistor
- It stores data in terms of charge in the capacitor
- A DRAM chip consists of (10s of 1000s of) rows of such cells

DRAM Refresh

- DRAM capacitor charge leaks over time

- The memory controller needs to refresh each row periodically to restore charge
  - Activate each row every N ms
  - Typical N = 64 ms

- Downsides of refresh

  -- Energy consumption: Each refresh consumes energy

  -- Performance degradation: DRAM rank/bank unavailable while refreshed

  -- QoS/predictability impact: (Long) pause times during refresh

  -- Refresh rate limits DRAM capacity scaling

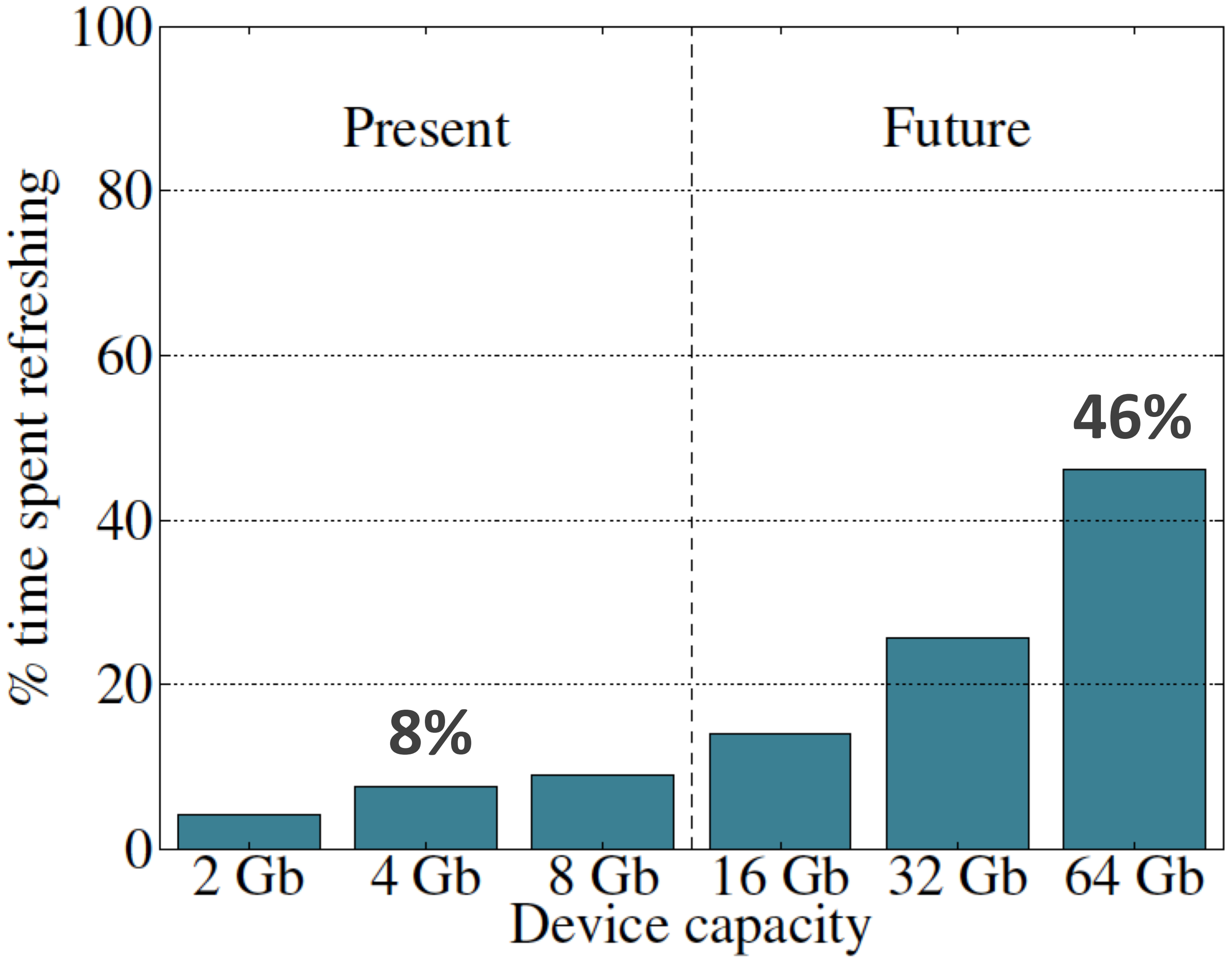
First, Some Analysis

- Imagine a system with 8 ExaByte DRAM (2^63 bytes)
- Assume a row size of 8 KiloBytes (2^13 bytes)

- How many rows are there?
- How many refreshes happen in 64ms?
- What is the total power consumption of DRAM refresh?
- What is the total energy consumption of DRAM refresh during a day?

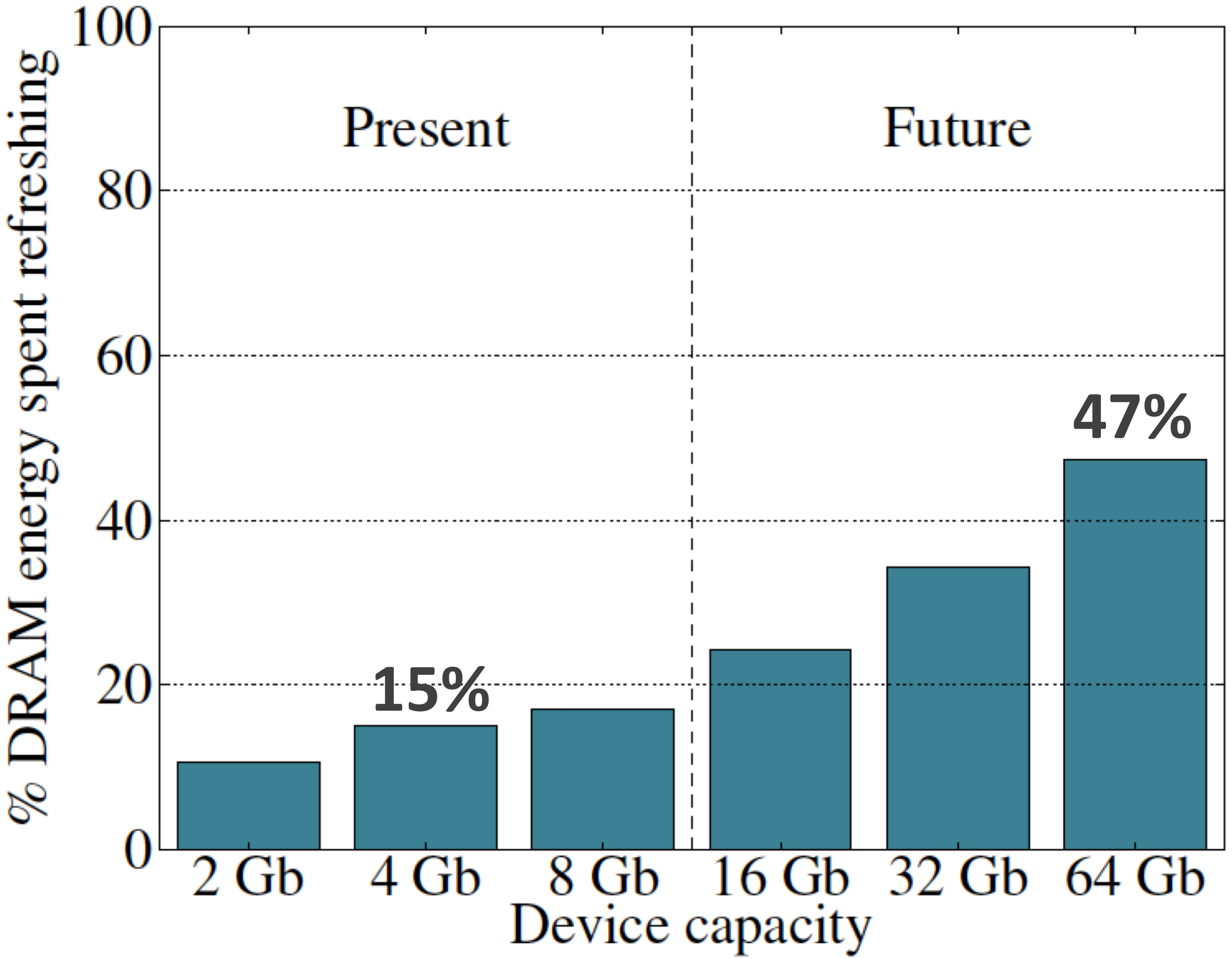
- A good exercise…
- Brownie points from me if you do it...

Refresh Overhead: Performance



Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

Refresh Overhead: Energy



Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

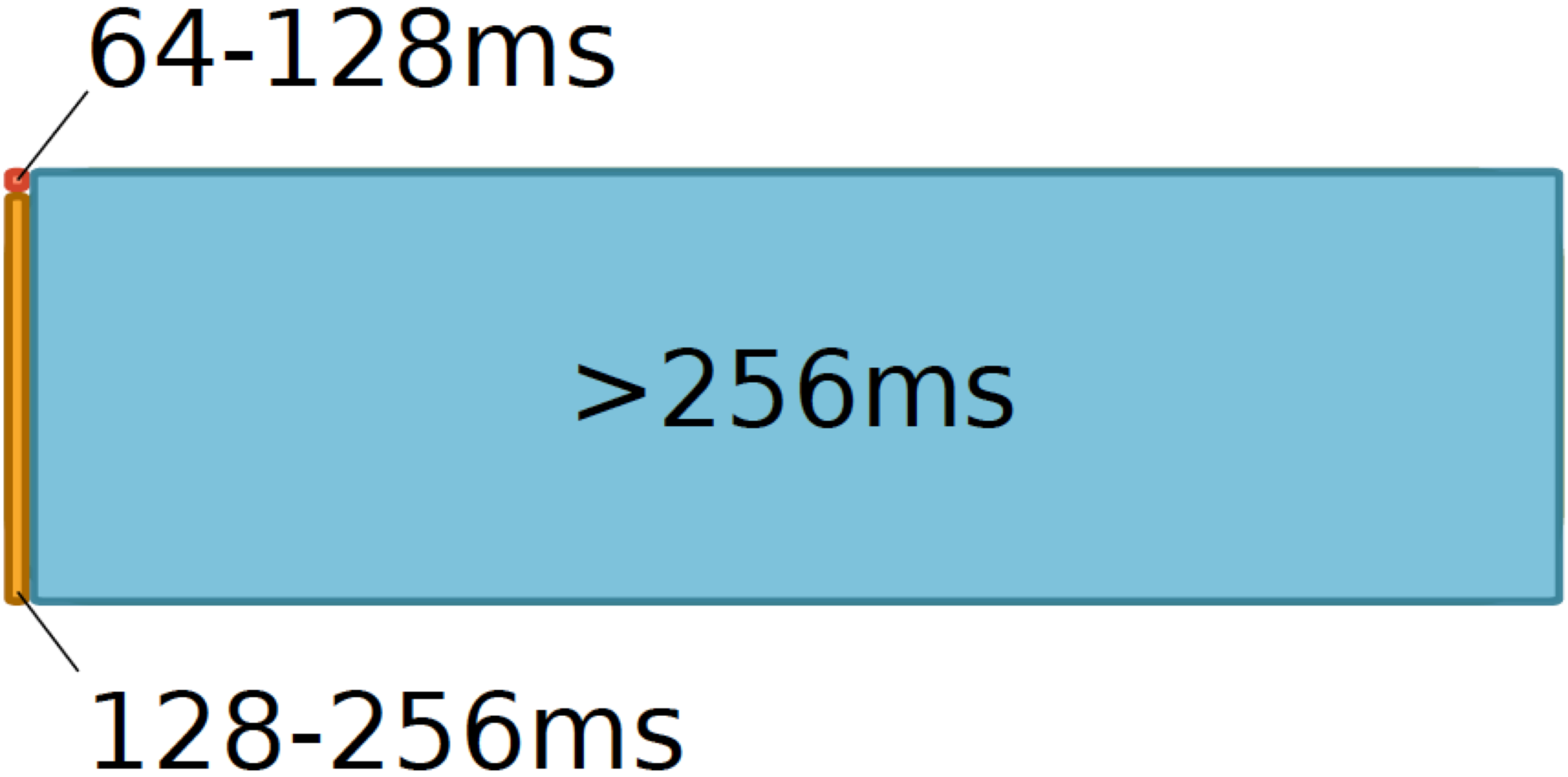
How Do We Solve the Problem?

- Observation: All DRAM rows are refreshed every 64ms.

- Critical thinking: Do we need to refresh all rows every 64ms?

- What if we knew what happened underneath (in DRAM cells) and exposed that information to upper layers?

Underneath: Retention Time Profile of DRAM



Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

Aside: Why Do We Have Such a Profile?

- Answer: Manufacturing is not perfect
- Not all DRAM cells are exactly the same
- Some are more leaky than others
- This is called Manufacturing Process Variation

Opportunity: Taking Advantage of This Profile

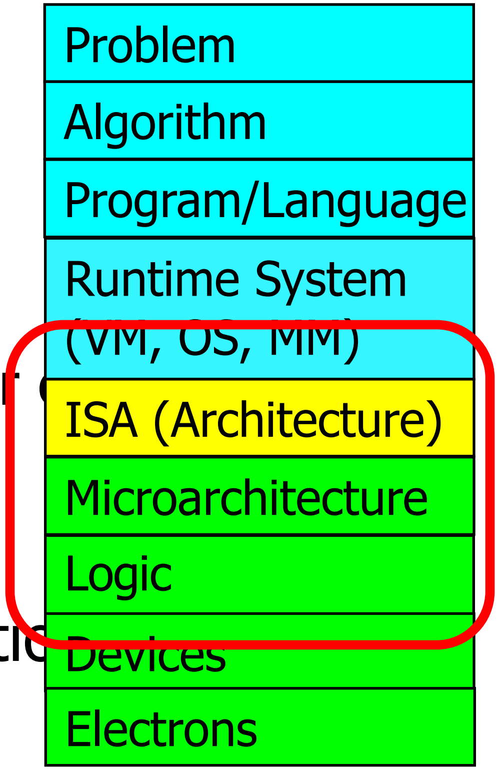
- Assume we know the retention time of each row exactly

- What can we do with this information?

- Who do we expose this information to?

- How much information do we expose?
  - Affects hardware/software overhead, power c verification complexity, cost

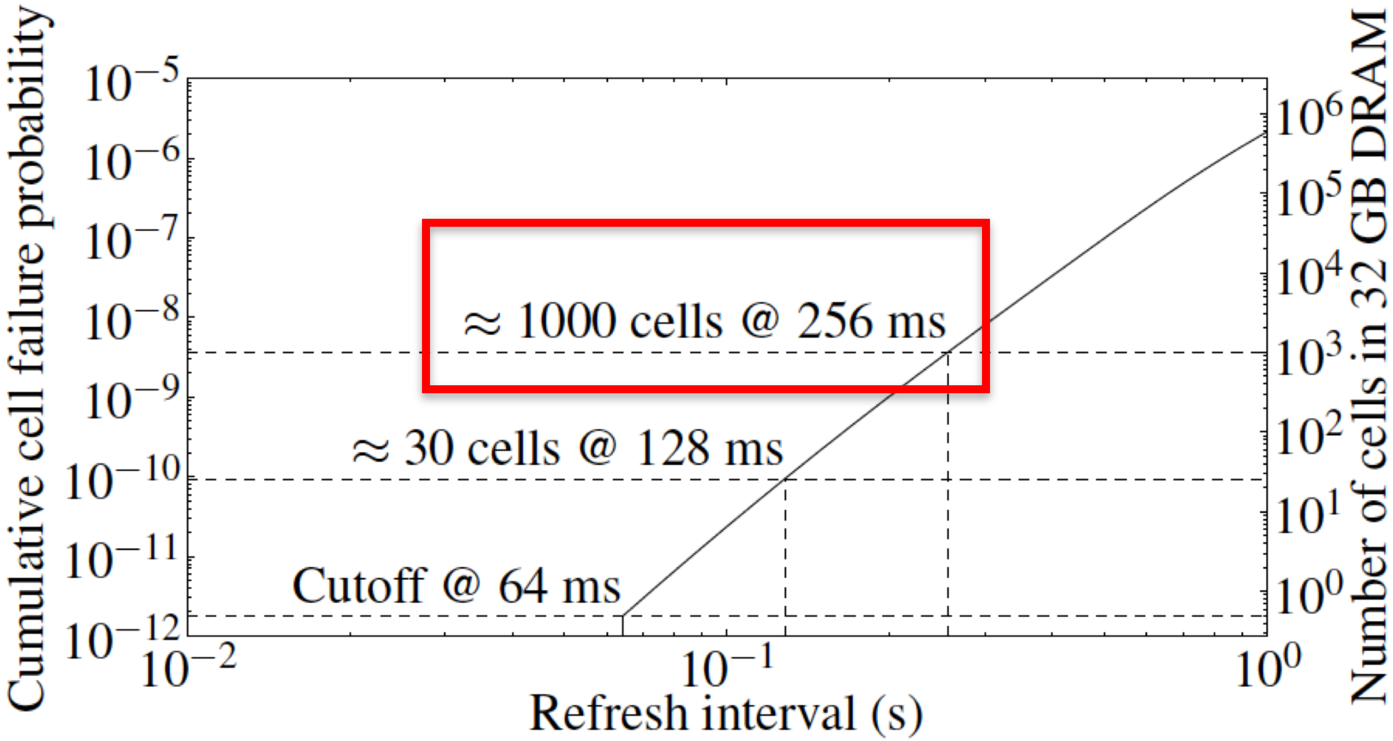
- How do we determine this profile informatio
  - Also, who determines it?



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

Retention Time of DRAM Rows

- Observation: Overwhelming majority of DRAM rows can be refreshed much less often without losing data



Key Idea of RAIDR: Refresh weak rows more frequently, all other rows less frequently

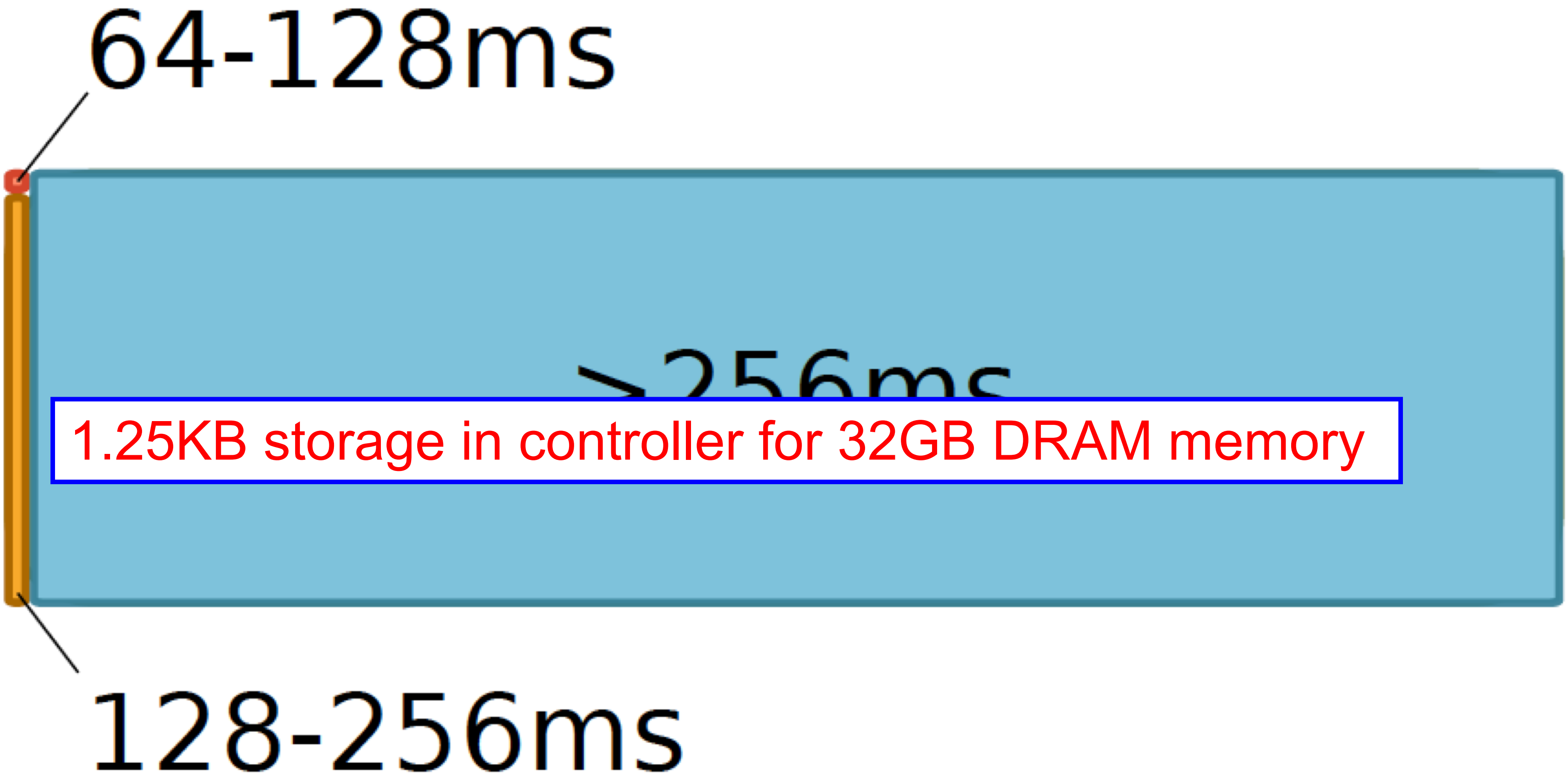
Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

RAIDR: Eliminating Unnecessary DRAM Refreshes

Liu, Jaiyen, Veras, Mutlu,
RAIDR: Retention-Aware Intelligent DRAM Refresh
ISCA 2012.

RAIDR: Mechanism

1. Profiling: Identify the retention time of all DRAM rows



1.25KB storage in controller for 32GB DRAM memory

→ check the bins to determine refresh rate of a row

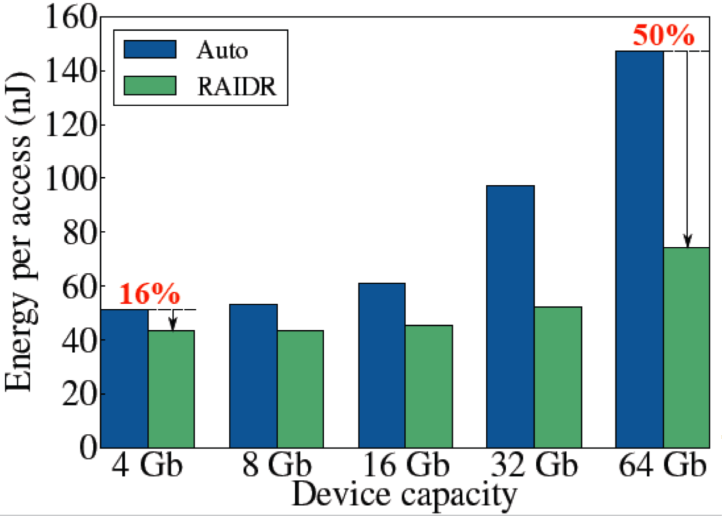
Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

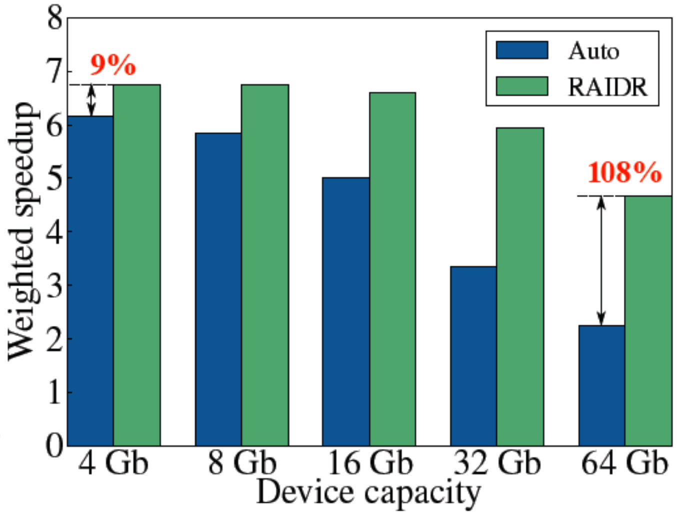
RAIDR: Results and Takeaways

- System: 32GB DRAM, 8-core; Various workloads

- RAIDR hardware cost: 1.25 kB (2 Bloom filters)
- Refresh reduction: 74.6%
- Dynamic DRAM energy reduction: 16%
- Idle DRAM power reduction: 20%
- Performance improvement: 9%

- Benefits increase as DRAM scales in density

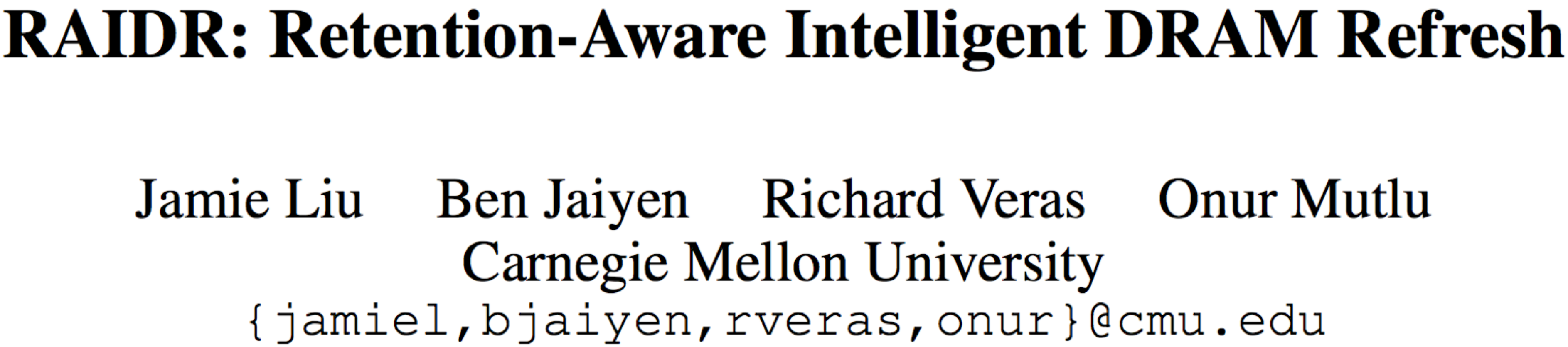




Reading on RAIDR

- Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu, **"RAIDR: Retention-Aware Intelligent DRAM Refresh"** *Proceedings of the 39th International Symposium on Computer Architecture (**ISCA**)*, Portland, OR, June 2012. Slides (pdf)

- One potential reading for your Homework 1 assignment



RAIDR: Retention-Aware Intelligent DRAM Refresh

Jamie Liu   Ben Jaiyen   Richard Veras   Onur Mutlu
Carnegie Mellon University
{jamiel,bjaiyen,rveras,onur}@cmu.edu

If You Are Interested … Further Readings

- Onur Mutlu,
  "Memory Scaling: A Systems Architecture Perspective"
  *Technical talk at MemCon 2013 (**MEMCON**)*, Santa Clara, CA, August 2013.
  Slides (pptx) (pdf) Video

- Kevin Chang, Donghyuk Lee, Zeshan Chishti, Alaa Alameldeen, Chris Wilkerson, Yoongu Kim, and Onur Mutlu,
  "Improving DRAM Performance by Parallelizing Refreshes with Accesses"
  *Proceedings of the 20th International Symposium on High-Performance Computer Architecture (**HPCA**)*, Orlando, FL, February 2014. Slides (pptx) (pdf)

Takeaway 1

Breaking the abstraction layers (between components and transformation hierarchy levels)

and knowing what is underneath

enables you to **understand** and **solve** problems

Takeaway 2

Cooperation between
multiple components and layers
can enable
more effective
solutions and systems

Digging Deeper: Making RAIDR Work

"Good ideas are a dime a dozen"

"Making them work is oftentimes the real contribution"

Recall: RAIDR: Mechanism

1. Profiling: Identify the retention time of all DRAM rows

→ can be done at design time or during operation

2. Binning: Store rows into bins by retention time

→ use Bloom Filters for efficient and scalable storage

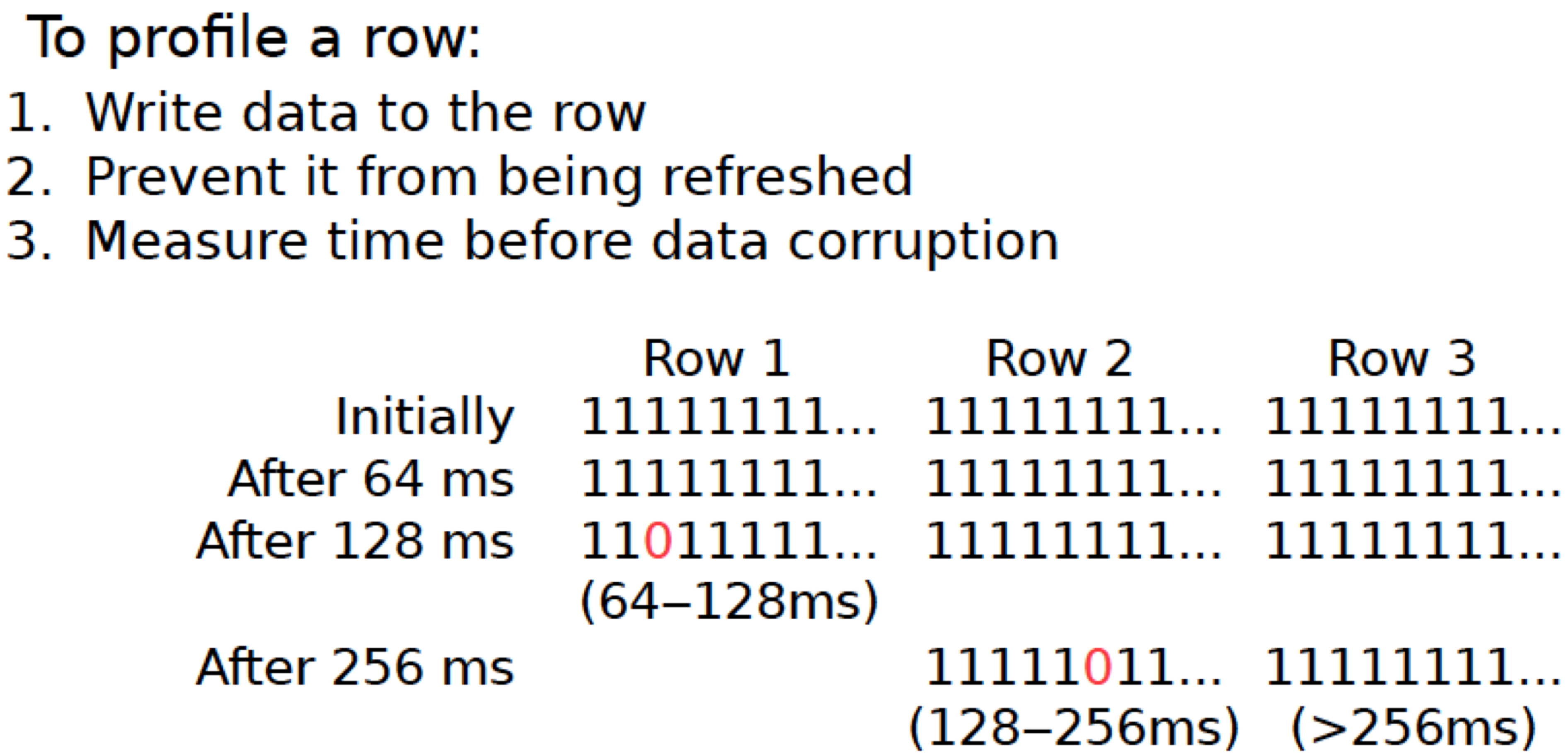
1.25KB storage in controller for 32GB DRAM memory

3. Refreshing: Memory controller refreshes rows in different bins at different rates

→ check the bins to determine refresh rate of a row

Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

1. Profiling



To profile a row:

1. Write data to the row
2. Prevent it from being refreshed
3. Measure time before data corruption

	Row 1	Row 2	Row 3
Initially	11111111...	11111111...	11111111...
After 64 ms	11111111...	11111111...	11111111...
After 128 ms	11011111...	11111111...	11111111...
	(64–128ms)		
After 256 ms		11111011...	11111111...
		(128–256ms)	(>256ms)

DRAM Retention Time Profiling

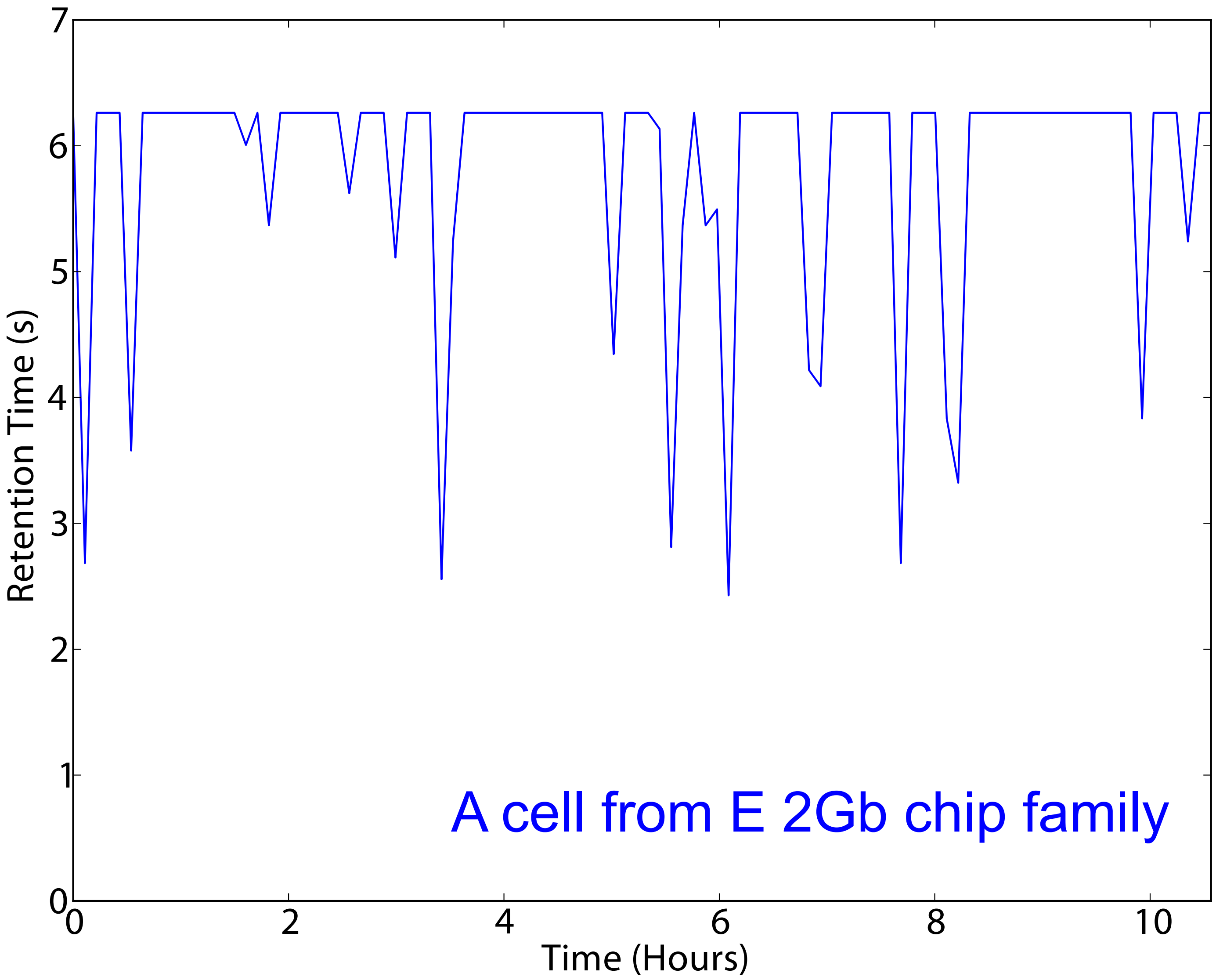
- Q: Is it really this easy?
- A: Ummm, not really…

Two Challenges to Retention Time Profiling

- Data Pattern Dependence (DPD) of retention time

- Variable Retention Time (VRT) phenomenon

An Example VRT Cell



VRT: Implications on Profiling Mechanisms

- Problem 1: There does not seem to be a way of determining if a cell exhibits VRT without actually observing a cell exhibiting VRT
  - VRT is a memoryless random process **[Kim+ JJAP 2010]**

- Problem 2: VRT complicates retention time profiling by DRAM manufacturers
  - Exposure to very high temperatures can induce VRT in cells that were not previously susceptible

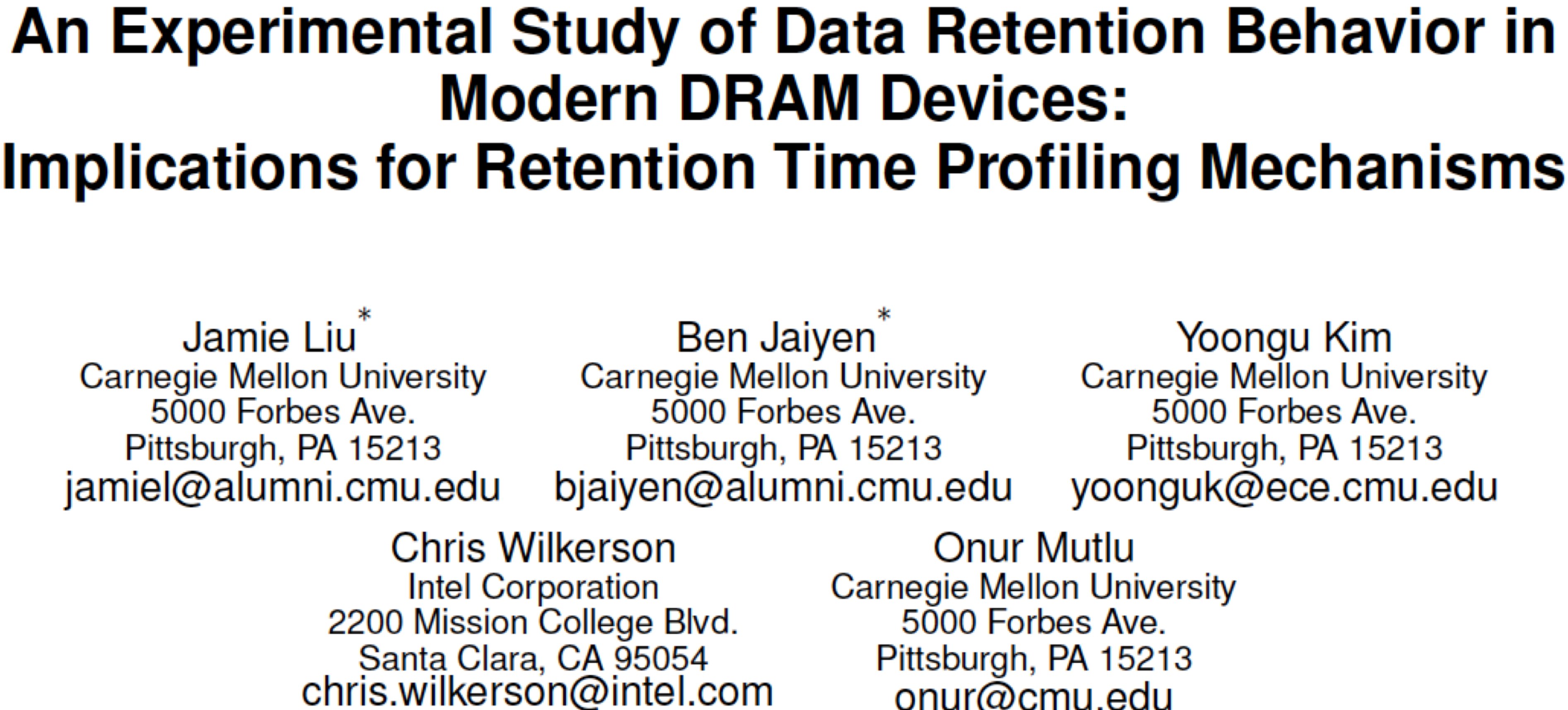
    → can happen during soldering of DRAM chips

    → manufacturer's retention time profile may not be accurate

- One option for future work: Use ECC to continuously profile DRAM online while aggressively reducing refresh rate
  - Need to keep ECC overhead in check

More on DRAM Retention Analysis

- Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, and Onur Mutlu, **"An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms"** *Proceedings of the 40th International Symposium on Computer Architecture (**ISCA**)*, Tel-Aviv, Israel, June 2013. Slides (ppt) Slides (pdf)



An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms

Jamie Liu*
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
jamiel@alumni.cmu.edu

Ben Jaiyen*
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
bjaiyen@alumni.cmu.edu

Yoongu Kim
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
yoonguk@ece.cmu.edu

Chris Wilkerson
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054
chris.wilkerson@intel.com

Onur Mutlu
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
onur@cmu.edu

Finding DRAM Retention Failures

- How can we reliably find the retention time of all DRAM cells?

- Goals: so that we can
  - Make DRAM reliable and secure
  - Make techniques like RAIDR work
    → improve performance and energy

Mitigation of Retention Issues [SIGMETRICS'14]

- Samira Khan, Donghyuk Lee, Yoongu Kim, Alaa Alameldeen, Chris Wilkerson, and Onur Mutlu,
"The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study"
*Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (**SIGMETRICS**)*, Austin, TX, June 2014. [Slides (pptx) (pdf)] [Poster (pptx) (pdf)] [Full data sets]

The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study

Samira Khan[†][*]
samirakhan@cmu.edu

Donghyuk Lee[†]
donghyuk1@cmu.edu

Yoongu Kim[†]
yoongukim@cmu.edu

Alaa R. Alameldeen[*]
alaa.r.alameldeen@intel.com

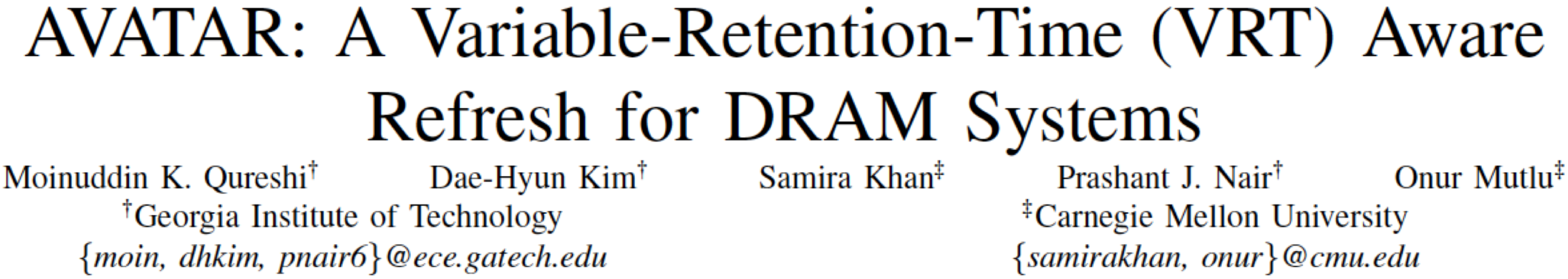
Chris Wilkerson[*]
chris.wilkerson@intel.com

Onur Mutlu[†]
onur@cmu.edu

[†]Carnegie Mellon University [*]Intel Labs

Handling Variable Retention Time [DSN'15]

- Moinuddin Qureshi, Dae Hyun Kim, Samira Khan, Prashant Nair, and Onur Mutlu, **"AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems"**
*Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Rio de Janeiro, Brazil, June 2015.
[Slides (pptx) (pdf)]

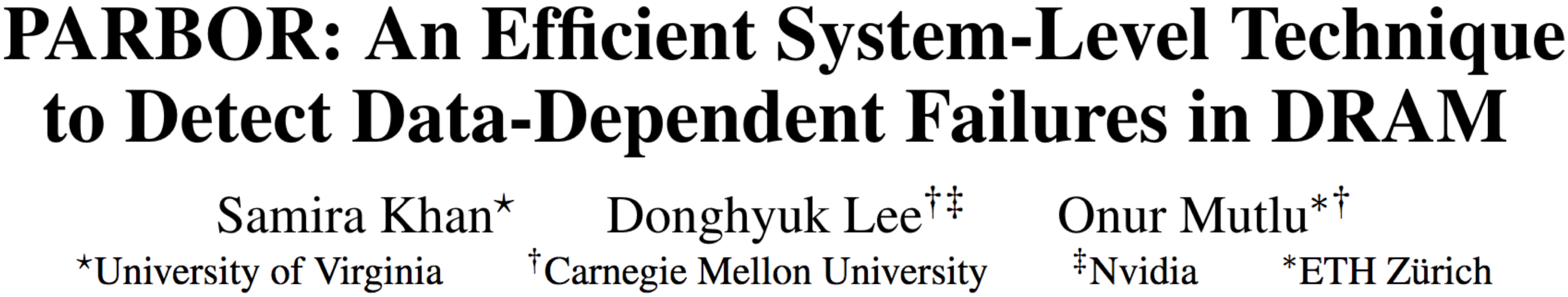


AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems

Moinuddin K. Qureshi[†] Dae-Hyun Kim[†] Samira Khan[‡] Prashant J. Nair[†] Onur Mutlu[‡]
[†]Georgia Institute of Technology
{moin, dhkim, pnair6}@ece.gatech.edu
[‡]Carnegie Mellon University
{samirakhan, onur}@cmu.edu

Handling Data-Dependent Failures [DSN'16]

- Samira Khan, Donghyuk Lee, and Onur Mutlu,
  "PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM"
  *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Toulouse, France, June 2016.
  [Slides (pptx) (pdf)]



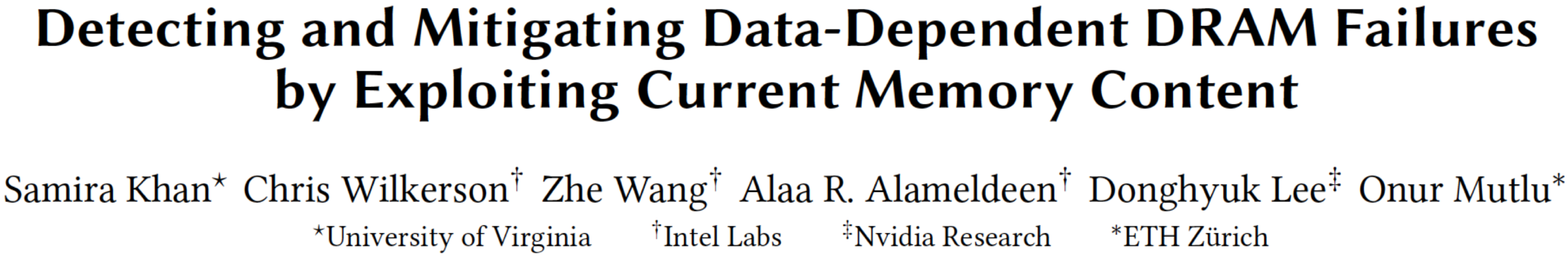
PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM

Samira Khan[⋆] Donghyuk Lee[†‡] Onur Mutlu[*†]

[⋆]University of Virginia [†]Carnegie Mellon University [‡]Nvidia [*]ETH Zürich

Handling Data-Dependent Failures [MICRO'17]

- Samira Khan, Chris Wilkerson, Zhe Wang, Alaa R. Alameldeen, Donghyuk Lee, and Onur Mutlu,
"Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content"
Proceedings of the 50th International Symposium on Microarchitecture (***MICRO***), Boston, MA, USA, October 2017.
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)]



Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content

Samira Khan[⋆] Chris Wilkerson[†] Zhe Wang[†] Alaa R. Alameldeen[†] Donghyuk Lee[‡] Onur Mutlu[*]
[⋆]University of Virginia [†]Intel Labs [‡]Nvidia Research [*]ETH Zürich

Handling Both DPD and VRT [ISCA'17]

- Minesh Patel, Jeremie S. Kim, and Onur Mutlu,
  "The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions"
  *Proceedings of the 44th International Symposium on Computer Architecture (**ISCA**)*, Toronto, Canada, June 2017.
  [Slides (pptx) (pdf)]
  [Lightning Session Slides (pptx) (pdf)]

- First experimental analysis of (mobile) LPDDR4 chips
- Analyzes the complex tradeoff space of retention time profiling
- Idea: enable fast and robust profiling at higher refresh intervals & temperatures

**The Reach Profiler (REAPER):
Enabling the Mitigation of DRAM Retention Failures
via Profiling at Aggressive Conditions**

Minesh Patel[§‡] Jeremie S. Kim[‡§] Onur Mutlu[§‡]
[§]ETH Zürich [‡]Carnegie Mellon University

2. Binning

- How to efficiently and scalably store rows into retention time bins?
- Use Hardware Bloom Filters [Bloom, CACM 1970]

Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

Bloom Filter

- [Bloom, CACM 1970]
- Probabilistic data structure that compactly represents set membership (presence or absence of element in a set)

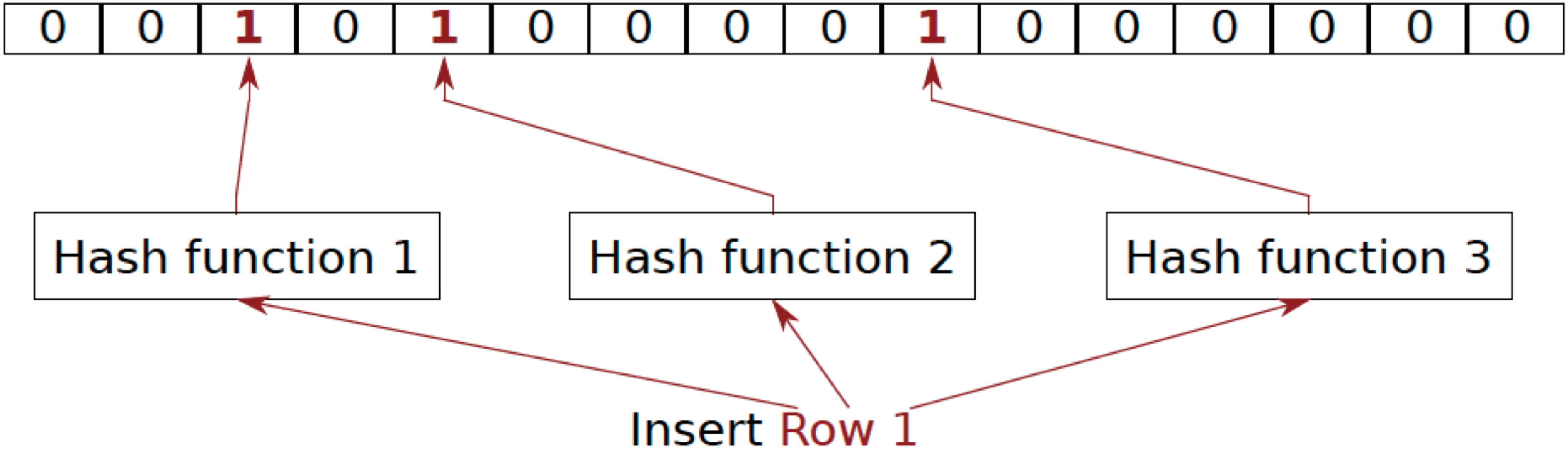
- Non-approximate set membership: Use 1 bit per element to indicate absence/presence of each element from an element space of N elements
- Approximate set membership: use a much smaller number of bits and indicate each element's presence/absence with a subset of those bits
  - Some elements map to the bits other elements also map to

- Operations: 1) insert, 2) test, 3) remove all elements

Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

Bloom Filter Operation Example

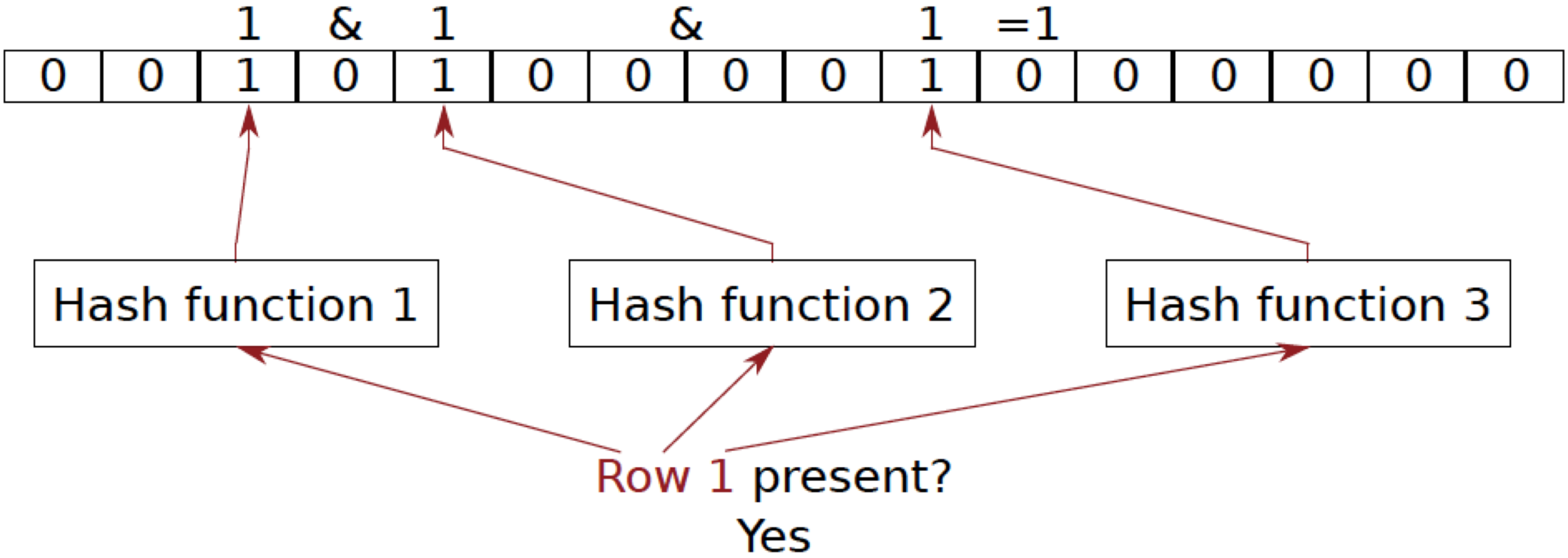
Example with 64–128ms bin:



Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

Bloom Filter Operation Example

Example with 64–128ms bin:



1 & 1 & 1 =1

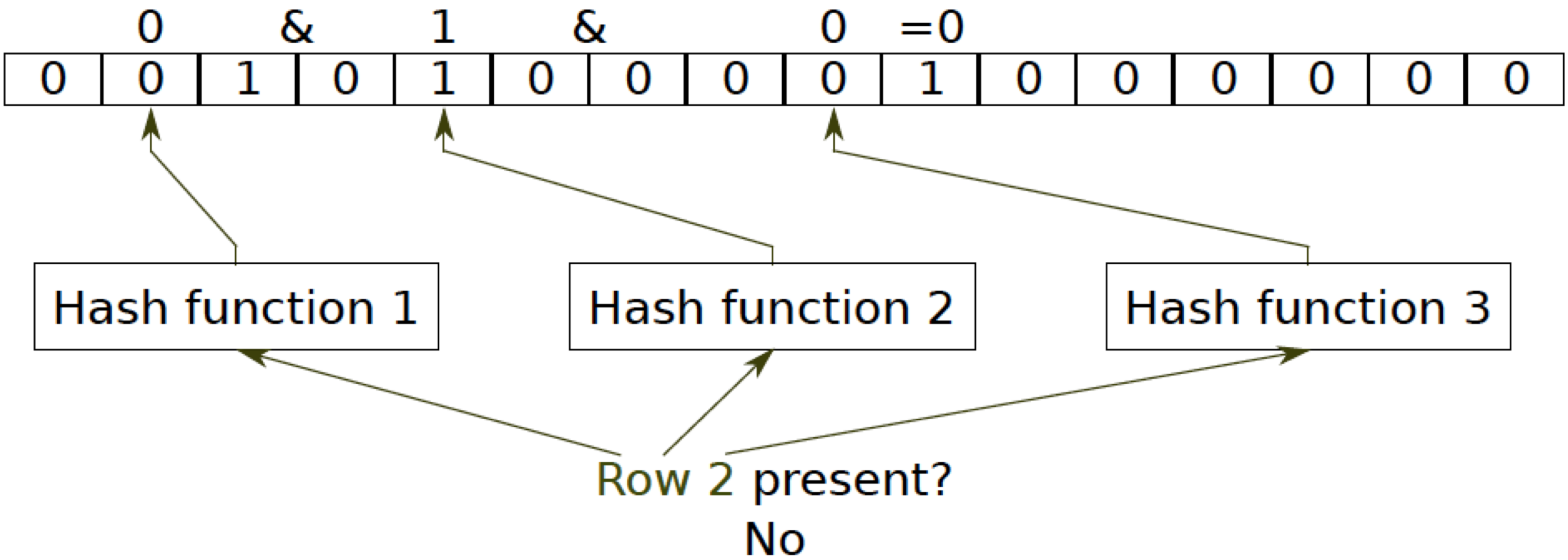
0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0

Row 1 present?

Yes

Bloom Filter Operation Example

Example with 64–128ms bin:



0 & 1 & 0 =0

0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0

Hash function 1

Hash function 2

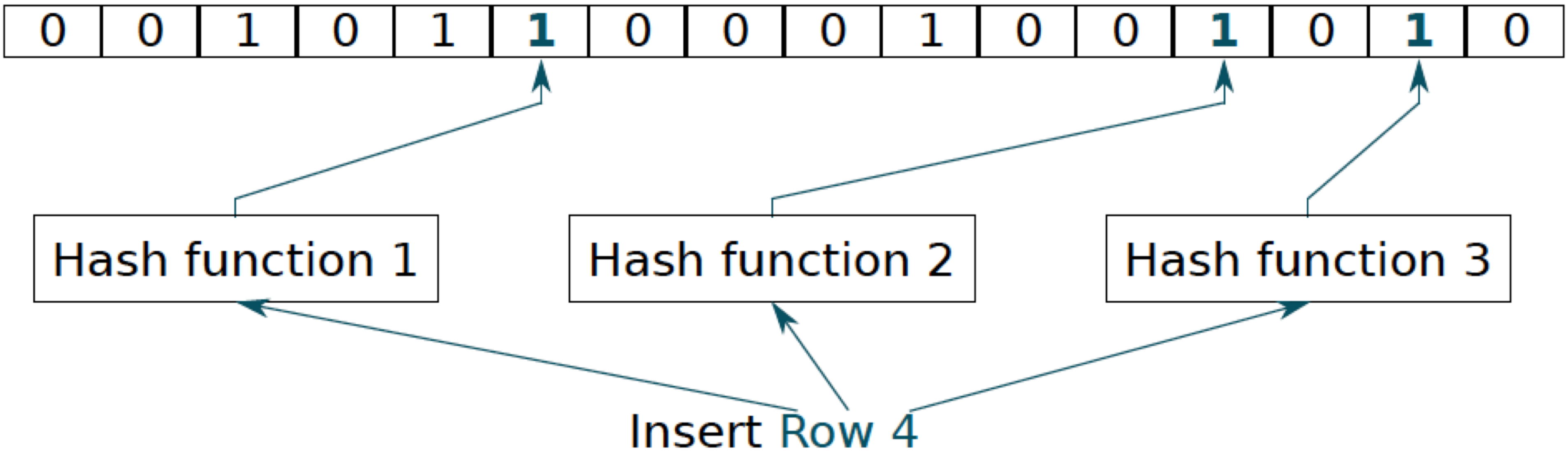
Hash function 3

Row 2 present?

No

Bloom Filter Operation Example

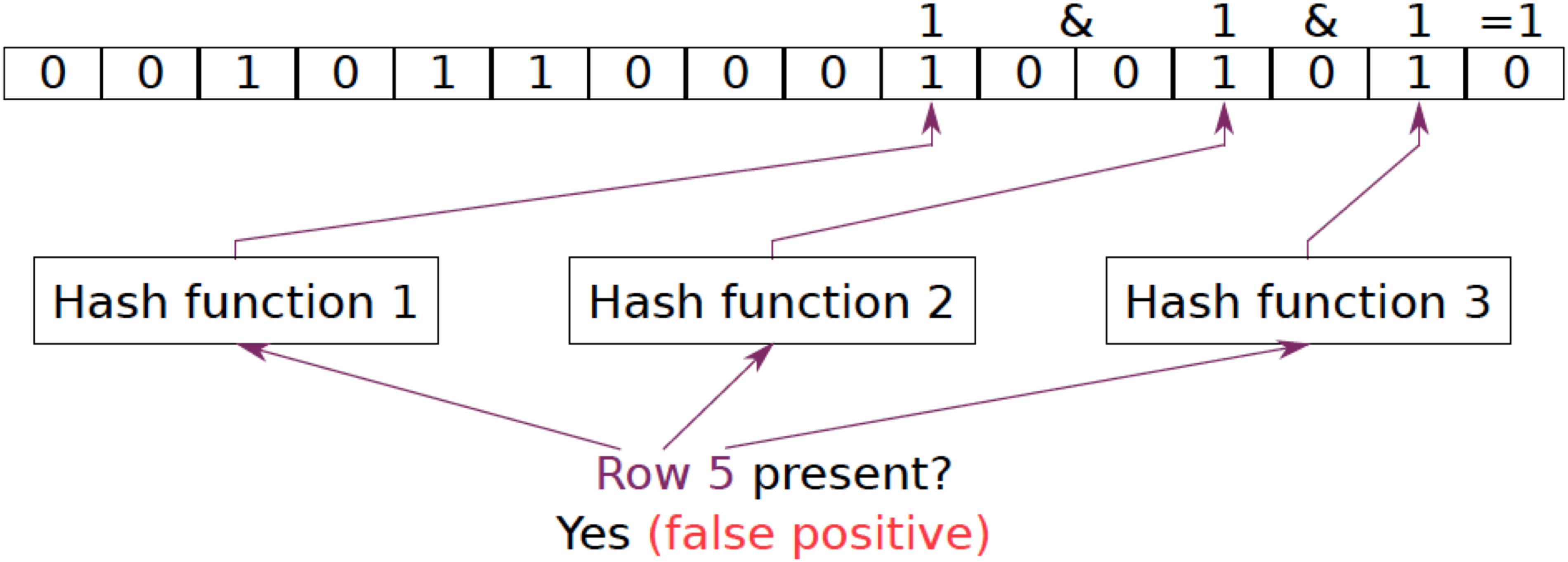
Example with 64–128ms bin:



0	0	1	0	1	**1**	0	0	0	1	0	0	**1**	0	**1**	0

Bloom Filter Operation Example

Example with 64–128ms bin:



1 & 1 & 1 =1

0	0	1	0	1	1	0	0	0	1	0	0	1	0	1	0

Hash function 1

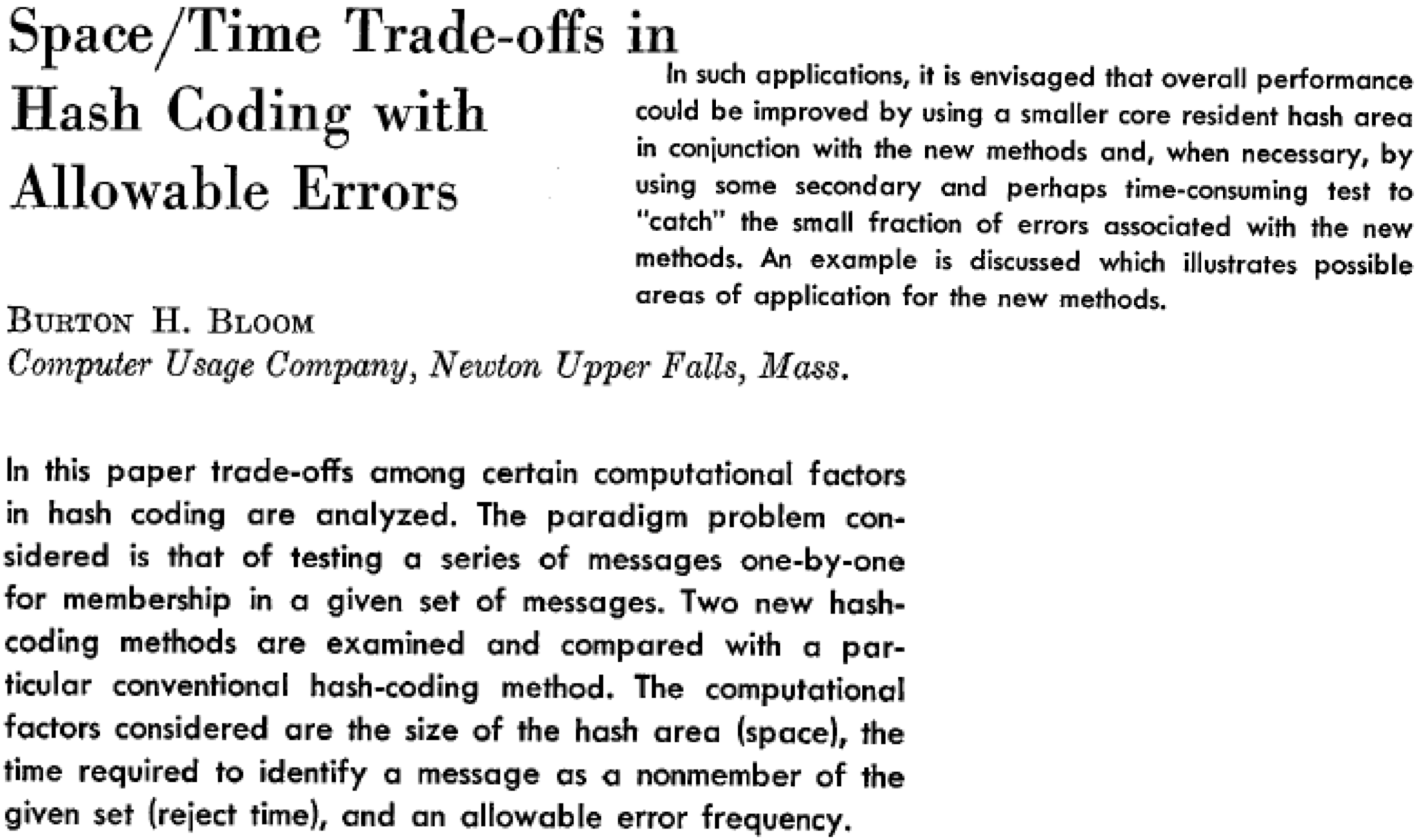
Hash function 2

Hash function 3

Row 5 present?

Yes (false positive)

Bloom Filters



Space/Time Trade-offs in Hash Coding with Allowable Errors

Burton H. Bloom
Computer Usage Company, Newton Upper Falls, Mass.

In this paper trade-offs among certain computational factors in hash coding are analyzed. The paradigm problem considered is that of testing a series of messages one-by-one for membership in a given set of messages. Two new hash-coding methods are examined and compared with a particular conventional hash-coding method. The computational factors considered are the size of the hash area (space), the time required to identify a message as a nonmember of the given set (reject time), and an allowable error frequency.

In such applications, it is envisaged that overall performance could be improved by using a smaller core resident hash area in conjunction with the new methods and, when necessary, by using some secondary and perhaps time-consuming test to "catch" the small fraction of errors associated with the new methods. An example is discussed which illustrates possible areas of application for the new methods.

Bloom, “Space/Time Trade-offs in Hash Coding with Allowable Errors”, CACM 1970.

Bloom Filters: Pros and Cons

- Advantages

  + Enables storage-efficient representation of set membership

  + Insertion and testing for set membership (presence) are fast

  + No false negatives: If Bloom Filter says an element is not present in the set, the element must not have been inserted

  + Enables tradeoffs between time & storage efficiency & false positive rate (via sizing and hashing)

- Disadvantages

  -- False positives: An element may be deemed to be present in the set by the Bloom Filter but it may never have been inserted

  Not the right data structure when you cannot tolerate false positives

Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

Benefits of Bloom Filters as Refresh Rate Bins

- False positives: a row may be declared present in the Bloom filter even if it was never inserted
  - Not a problem: Refresh some rows more frequently than needed

- No false negatives: rows are never refreshed less frequently than needed (no correctness problems)

- Scalable: a Bloom filter never overflows (unlike a fixed-size table)

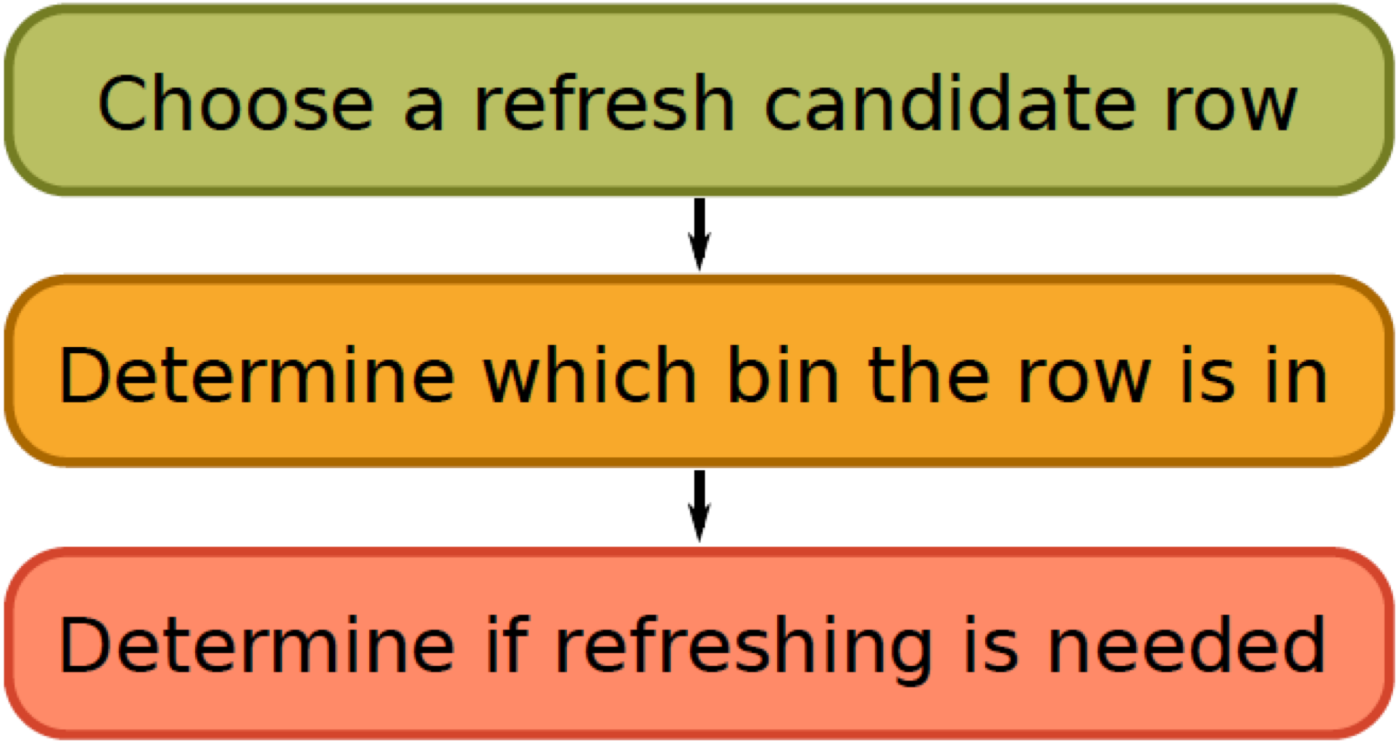
- Efficient: No need to store info on a per-row basis; simple hardware → 1.25 KB for 2 filters for 32 GB DRAM system

Use of Bloom Filters in Hardware

- Useful when you can tolerate false positives in set membership tests

- See the following recent examples for clear descriptions of how Bloom Filters are used
  - Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.
  - Seshadri et al., "The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing," PACT 2012.

3. Refreshing (RAIDR Refresh Controller)

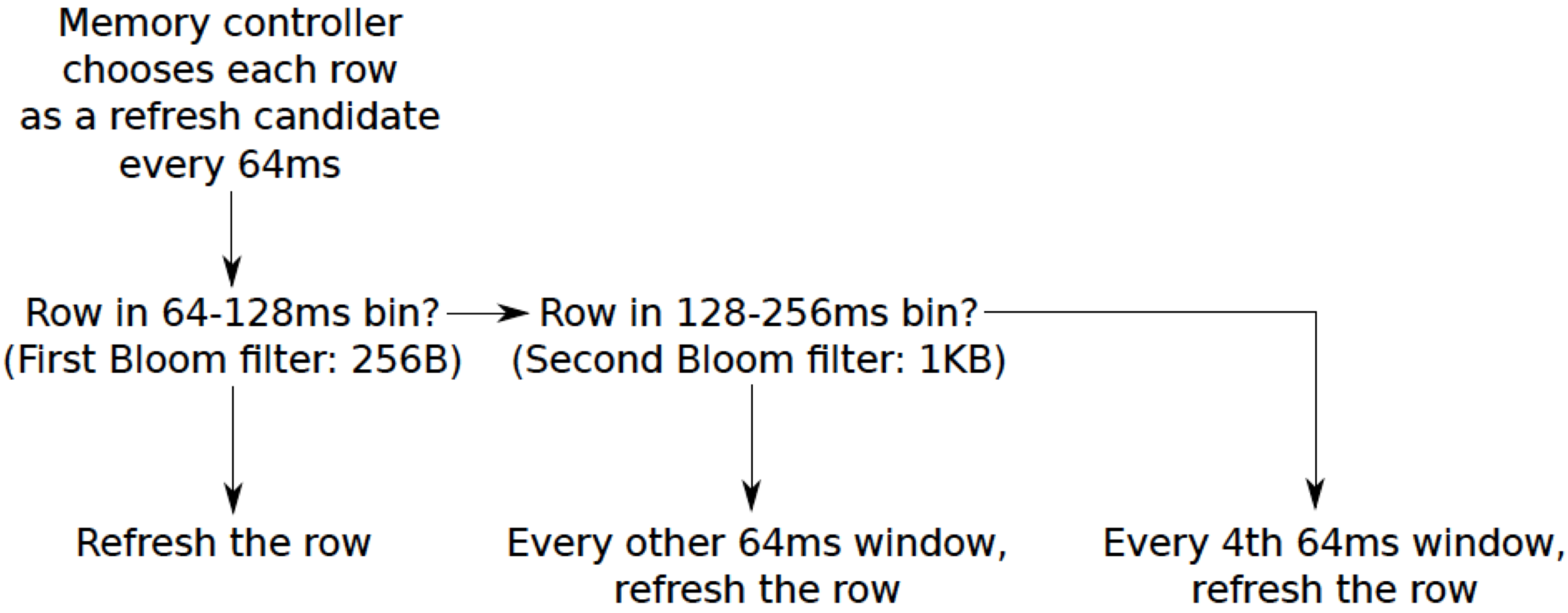


Choose a refresh candidate row

Determine which bin the row is in

Determine if refreshing is needed

3. Refreshing (RAIDR Refresh Controller)



Memory controller chooses each row as a refresh candidate every 64ms

Row in 64-128ms bin? (First Bloom filter: 256B)

Row in 128-256ms bin? (Second Bloom filter: 1KB)

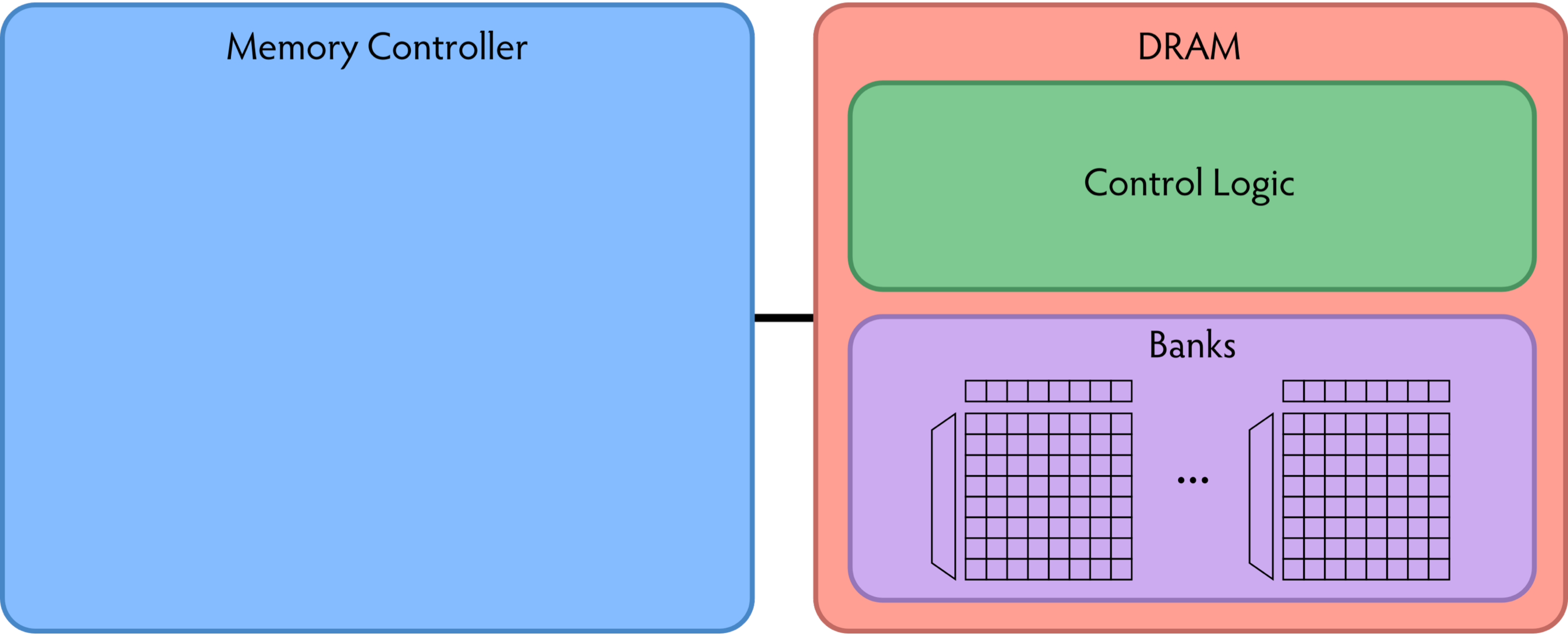
Refresh the row

Every other 64ms window, refresh the row

Every 4th 64ms window, refresh the row

Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

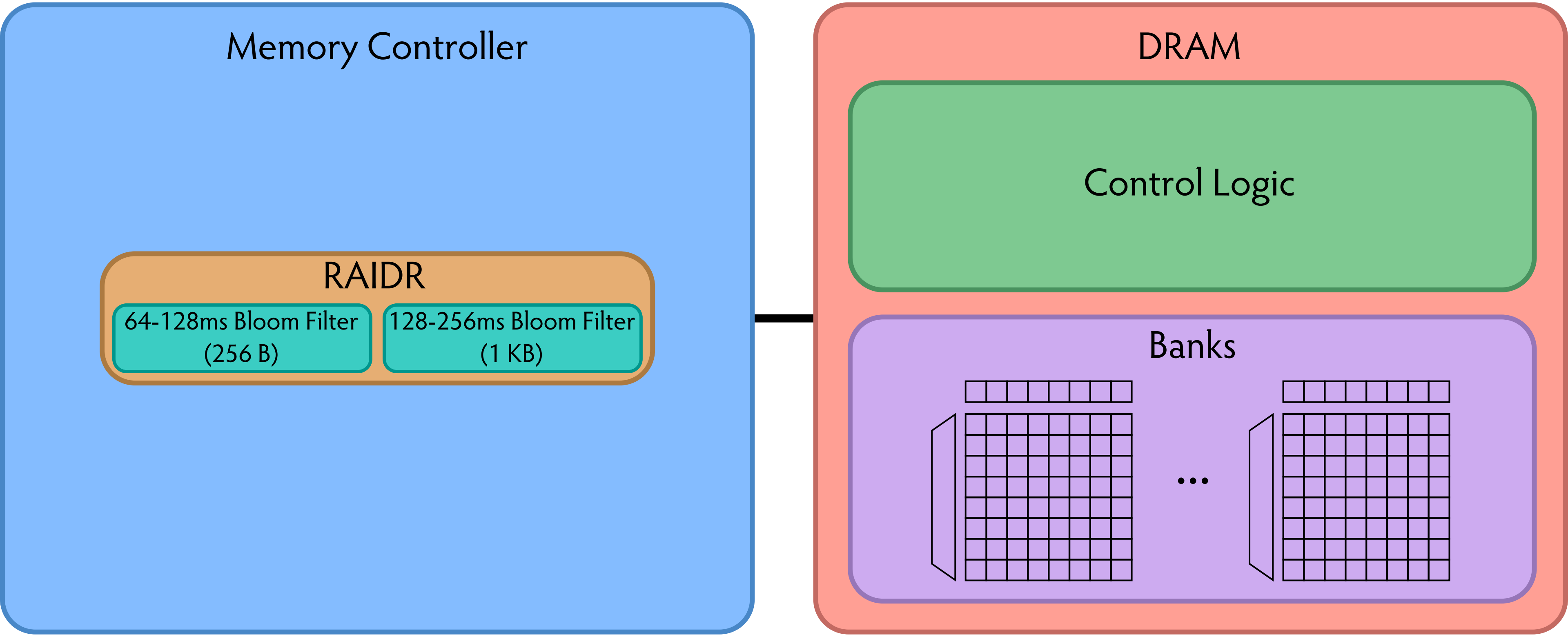
RAIDR: Baseline Design



Refresh control is in DRAM in today's auto-refresh systems

RAIDR can be implemented in either the controller or DRAM

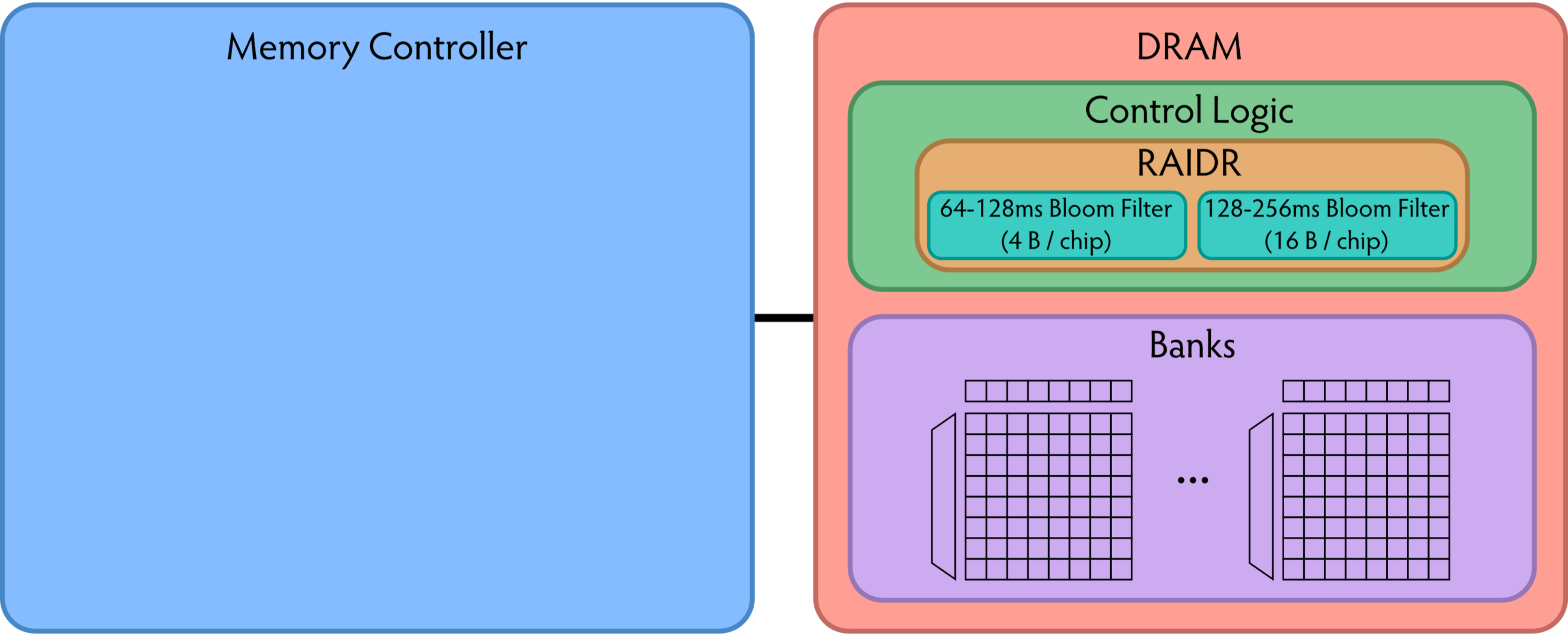
RAIDR in Memory Controller: Option 1



Overhead of RAIDR in DRAM controller:

1.25 KB Bloom Filters, 3 counters, additional commands issued for per-row refresh (all accounted for in evaluations)

RAIDR in DRAM Chip: Option 2



Overhead of RAIDR in DRAM chip:

Per-chip overhead: 20B Bloom Filters, 1 counter (4 Gbit chip)

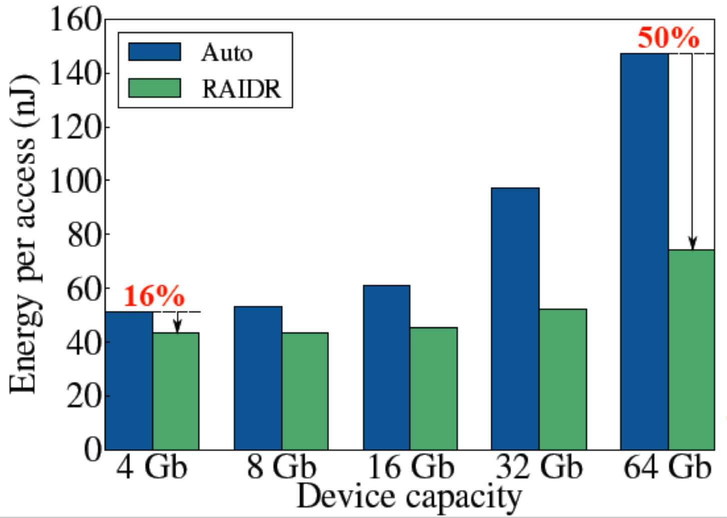
Total overhead: 1.25KB Bloom Filters, 64 counters (32 GB DRAM)

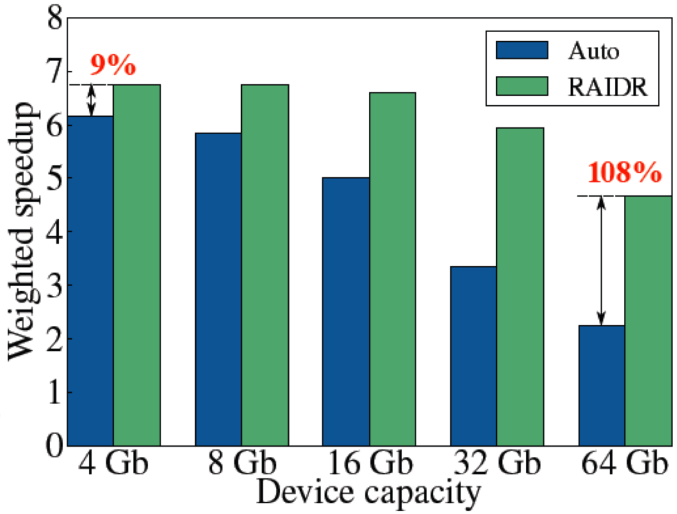
RAIDR: Results and Takeaways

- System: 32GB DRAM, 8-core; SPEC, TPC-C, TPC-H workloads

- RAIDR hardware cost: 1.25 kB (2 Bloom filters)
- Refresh reduction: 74.6%
- Dynamic DRAM energy reduction: 16%
- Idle DRAM power reduction: 20%
- Performance improvement: 9%

- Benefits increase as DRAM scales in density





DRAM Refresh: More Questions

- What else can you do to reduce the impact of refresh?

- What else can you do if you know the retention times of rows?

- How can you accurately measure the retention time of DRAM rows?

- Recommended reading:
  - Liu et al., "An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms," ISCA 2013.

Industry Is Writing Papers About It, Too

DRAM Process Scaling Challenges

❖ Refresh

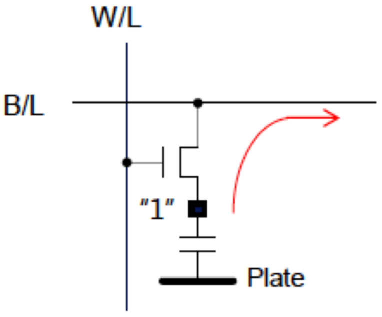
- Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance
- Leakage current of cell access transistors increasing

❖ tWR

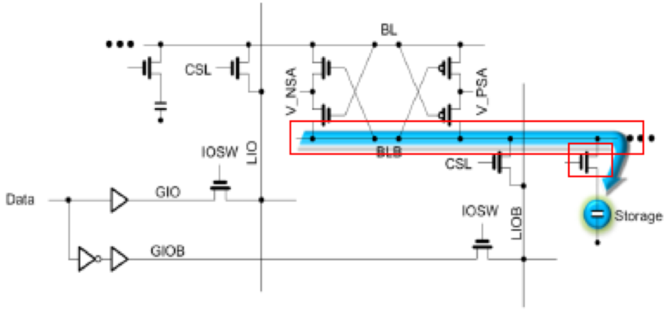
- Contact resistance between the cell capacitor and access transistor increasing
- On-current of the cell access transistor decreasing
- Bit-line resistance increasing

❖ VRT

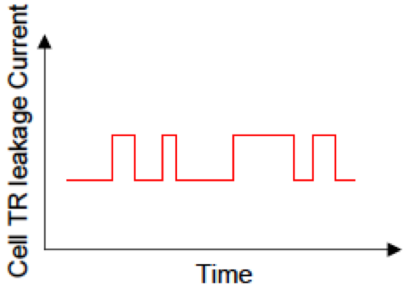
- Occurring more frequently with cell capacitance decreasing



Refresh

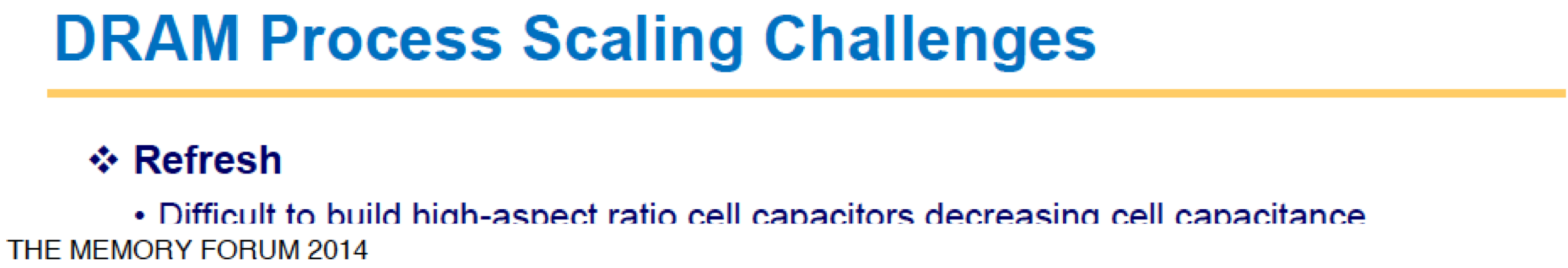


tWR



VRT

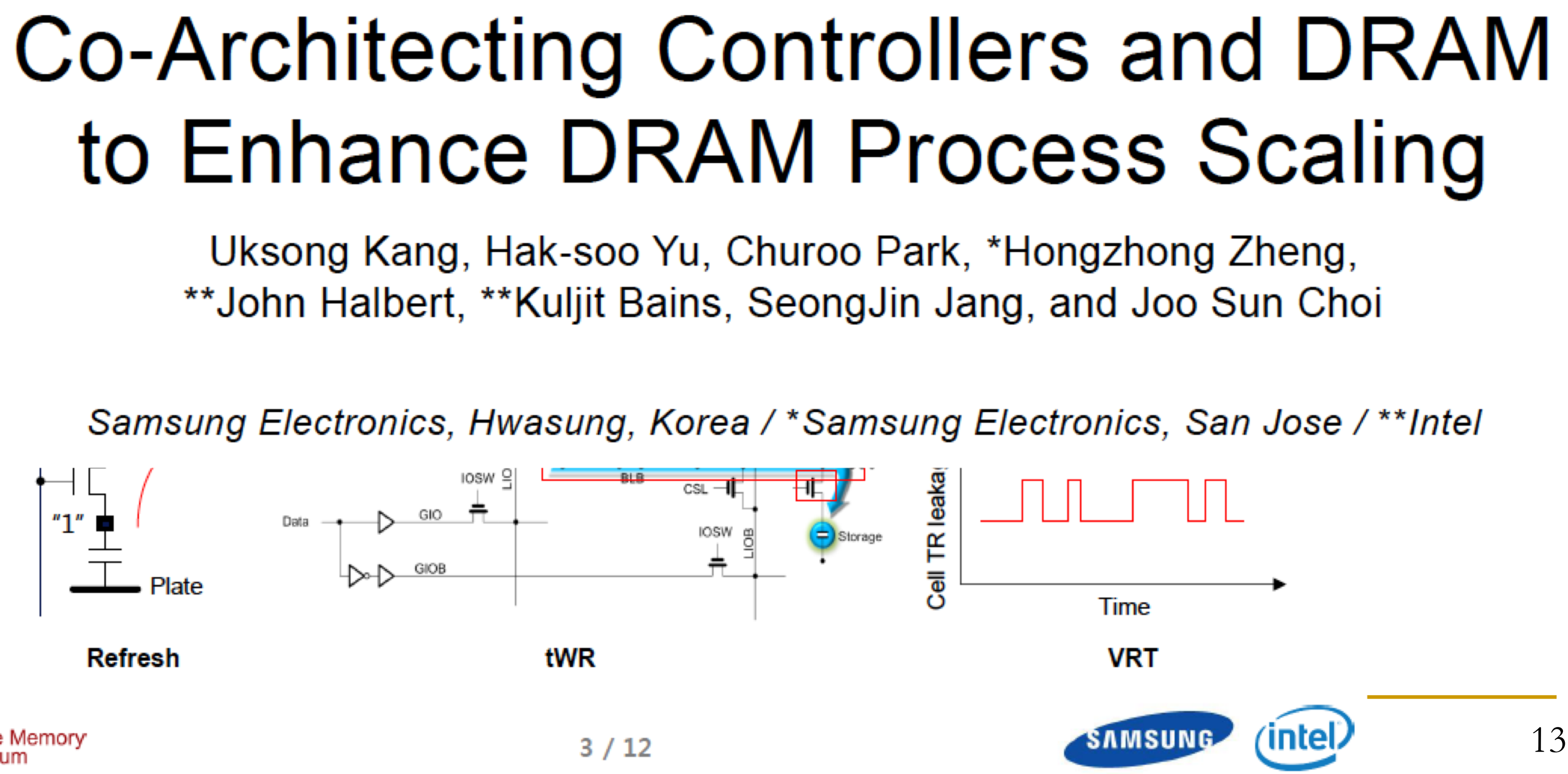
Call for Intelligent Memory Controllers



DRAM Process Scaling Challenges

- ❖ **Refresh**
  - Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance

THE MEMORY FORUM 2014



Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling

Uksong Kang, Hak-soo Yu, Churoo Park, *Hongzhong Zheng,
**John Halbert, **Kuljit Bains, SeongJin Jang, and Joo Sun Choi

*Samsung Electronics, Hwasung, Korea / *Samsung Electronics, San Jose / **Intel*

Refresh **tWR** **VRT**

The Memory Forum

3 / 12

SAMSUNG intel

We Will Dig Deeper More In This Course

"Good ideas are a dime a dozen"

"Making them work is oftentimes the real contribution"