P&S Processing-in-Memory

Data-Centric Architectures: Fundamentally Improving Performance and Energy

Dr. Juan Gómez Luna

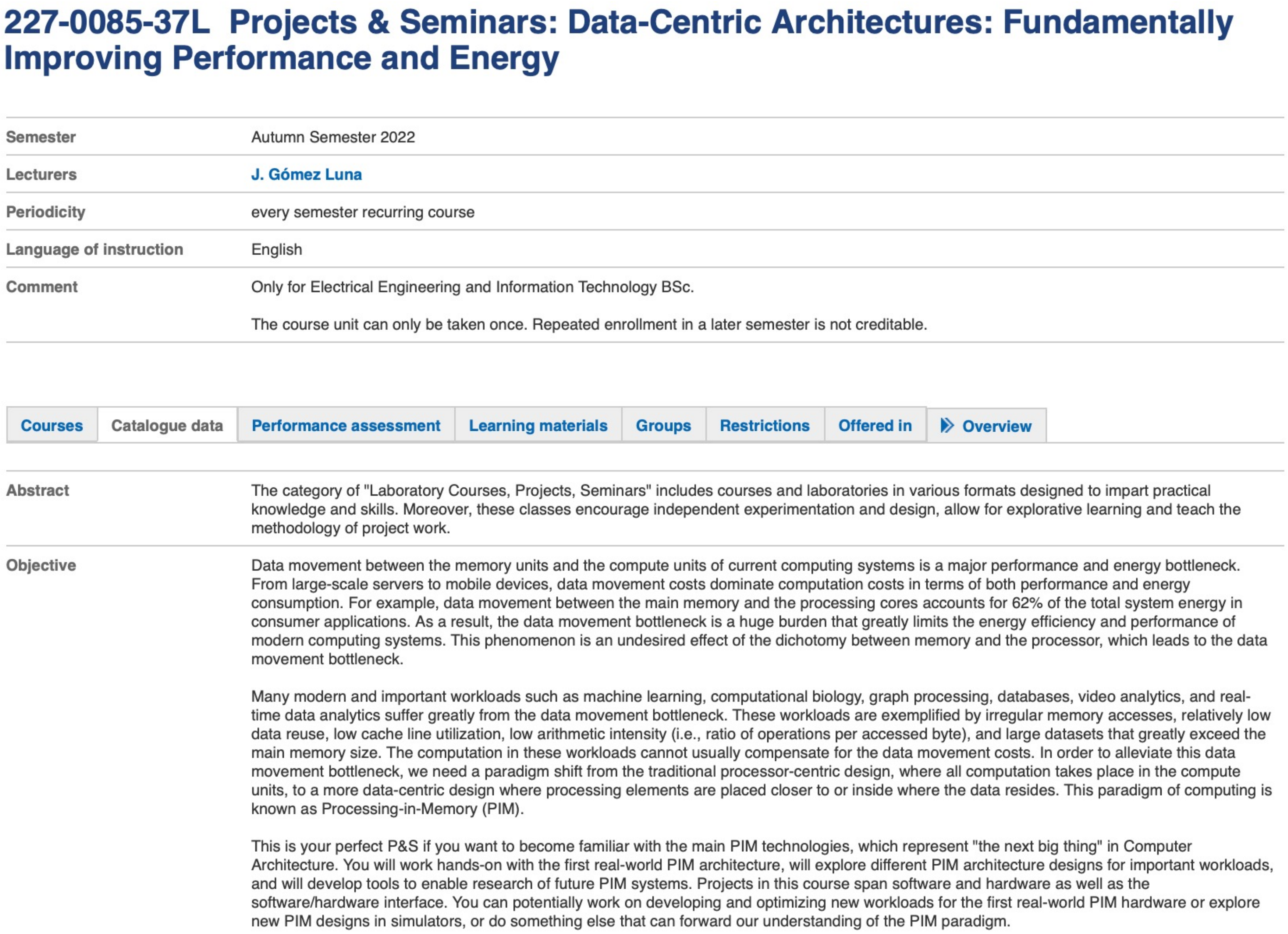
Prof. Onur Mutlu

ETH Zürich

Fall 2022

11 October 2022

P&S: Processing-in-Memory (I)



227-0085-37L Projects & Seminars: Data-Centric Architectures: Fundamentally Improving Performance and Energy

Semester	Autumn Semester 2022
Lecturers	J. Gómez Luna
Periodicity	every semester recurring course
Language of instruction	English
Comment	Only for Electrical Engineering and Information Technology BSc. The course unit can only be taken once. Repeated enrollment in a later semester is not creditable.

Abstract

The category of "Laboratory Courses, Projects, Seminars" includes courses and laboratories in various formats designed to impart practical knowledge and skills. Moreover, these classes encourage independent experimentation and design, allow for explorative learning and teach the methodology of project work.

Objective

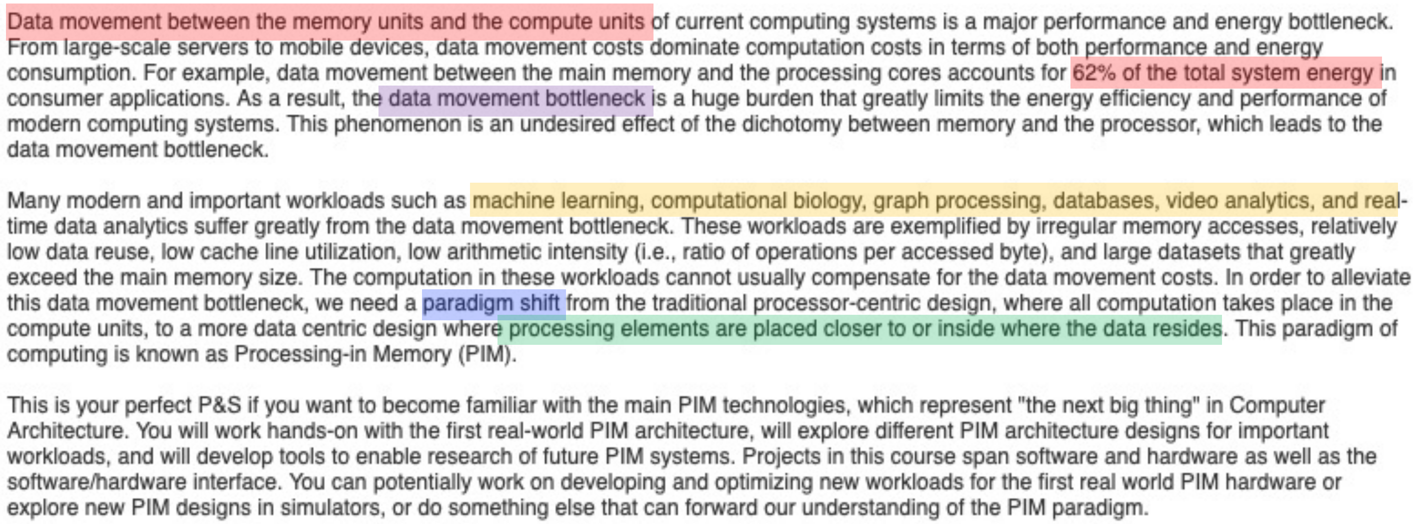
Data movement between the memory units and the compute units of current computing systems is a major performance and energy bottleneck. From large-scale servers to mobile devices, data movement costs dominate computation costs in terms of both performance and energy consumption. For example, data movement between the main memory and the processing cores accounts for 62% of the total system energy in consumer applications. As a result, the data movement bottleneck is a huge burden that greatly limits the energy efficiency and performance of modern computing systems. This phenomenon is an undesired effect of the dichotomy between memory and the processor, which leads to the data movement bottleneck.

Many modern and important workloads such as machine learning, computational biology, graph processing, databases, video analytics, and real-time data analytics suffer greatly from the data movement bottleneck. These workloads are exemplified by irregular memory accesses, relatively low data reuse, low cache line utilization, low arithmetic intensity (i.e., ratio of operations per accessed byte), and large datasets that greatly exceed the main memory size. The computation in these workloads cannot usually compensate for the data movement costs. In order to alleviate this data movement bottleneck, we need a paradigm shift from the traditional processor-centric design, where all computation takes place in the compute units, to a more data-centric design where processing elements are placed closer to or inside where the data resides. This paradigm of computing is known as Processing-in-Memory (PIM).

This is your perfect P&S if you want to become familiar with the main PIM technologies, which represent "the next big thing" in Computer Architecture. You will work hands-on with the first real-world PIM architecture, will explore different PIM architecture designs for important workloads, and will develop tools to enable research of future PIM systems. Projects in this course span software and hardware as well as the software/hardware interface. You can potentially work on developing and optimizing new workloads for the first real-world PIM hardware or explore new PIM designs in simulators, or do something else that can forward our understanding of the PIM paradigm.

http://www.vvz.ethz.ch/Vorlesungsverzeichnis/lerneinheit.view?semkez=2022W&ansicht=KATALOGDATEN&lerneinheitId=164152&lang=en

P&S: Processing-in-Memory (II)



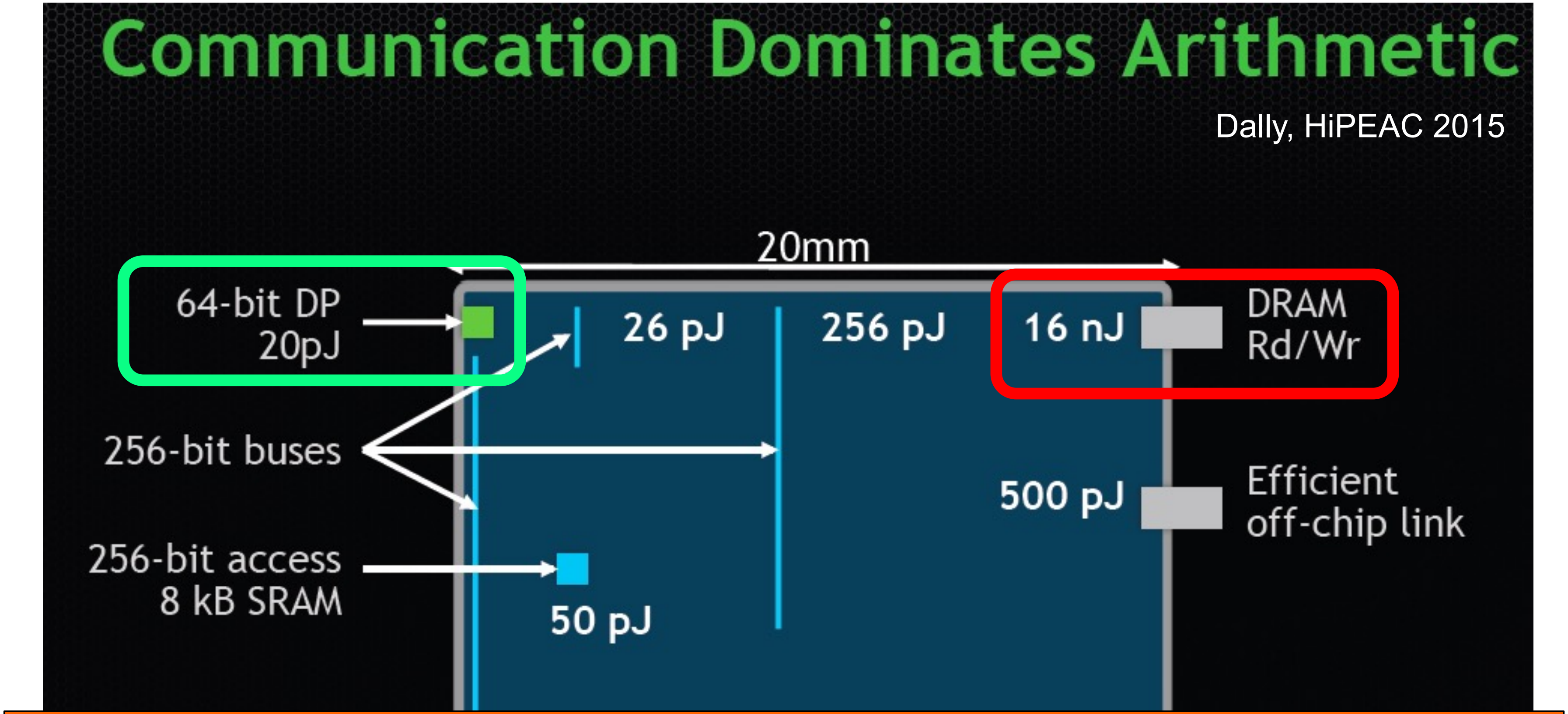
Data movement between the memory units and the compute units of current computing systems is a major performance and energy bottleneck. From large-scale servers to mobile devices, data movement costs dominate computation costs in terms of both performance and energy consumption. For example, data movement between the main memory and the processing cores accounts for 62% of the total system energy in consumer applications. As a result, the data movement bottleneck is a huge burden that greatly limits the energy efficiency and performance of modern computing systems. This phenomenon is an undesired effect of the dichotomy between memory and the processor, which leads to the data movement bottleneck.

Many modern and important workloads such as machine learning, computational biology, graph processing, databases, video analytics, and real-time data analytics suffer greatly from the data movement bottleneck. These workloads are exemplified by irregular memory accesses, relatively low data reuse, low cache line utilization, low arithmetic intensity (i.e., ratio of operations per accessed byte), and large datasets that greatly exceed the main memory size. The computation in these workloads cannot usually compensate for the data movement costs. In order to alleviate this data movement bottleneck, we need a paradigm shift from the traditional processor-centric design, where all computation takes place in the compute units, to a more data centric design where processing elements are placed closer to or inside where the data resides. This paradigm of computing is known as Processing-in Memory (PIM).

This is your perfect P&S if you want to become familiar with the main PIM technologies, which represent "the next big thing" in Computer Architecture. You will work hands-on with the first real-world PIM architecture, will explore different PIM architecture designs for important workloads, and will develop tools to enable research of future PIM systems. Projects in this course span software and hardware as well as the software/hardware interface. You can potentially work on developing and optimizing new workloads for the first real world PIM hardware or explore new PIM designs in simulators, or do something else that can forward our understanding of the PIM paradigm.

http://www.vvz.ethz.ch/Vorlesungsverzeichnis/lerneinheit.view?semkez=2022W&ansicht=KATALOGDATEN&lerneinheitId=164152&lang=en

Data Movement vs. Computation Energy



Communication Dominates Arithmetic

Dally, HiPEAC 2015

A memory access consumes ~1000X the energy of a complex addition

Goals of this P&S Course

P&S Processing-in-Memory: Contents

- We will introduce the data movement bottleneck, which is a major threat to high performance and energy efficiency of current computing systems

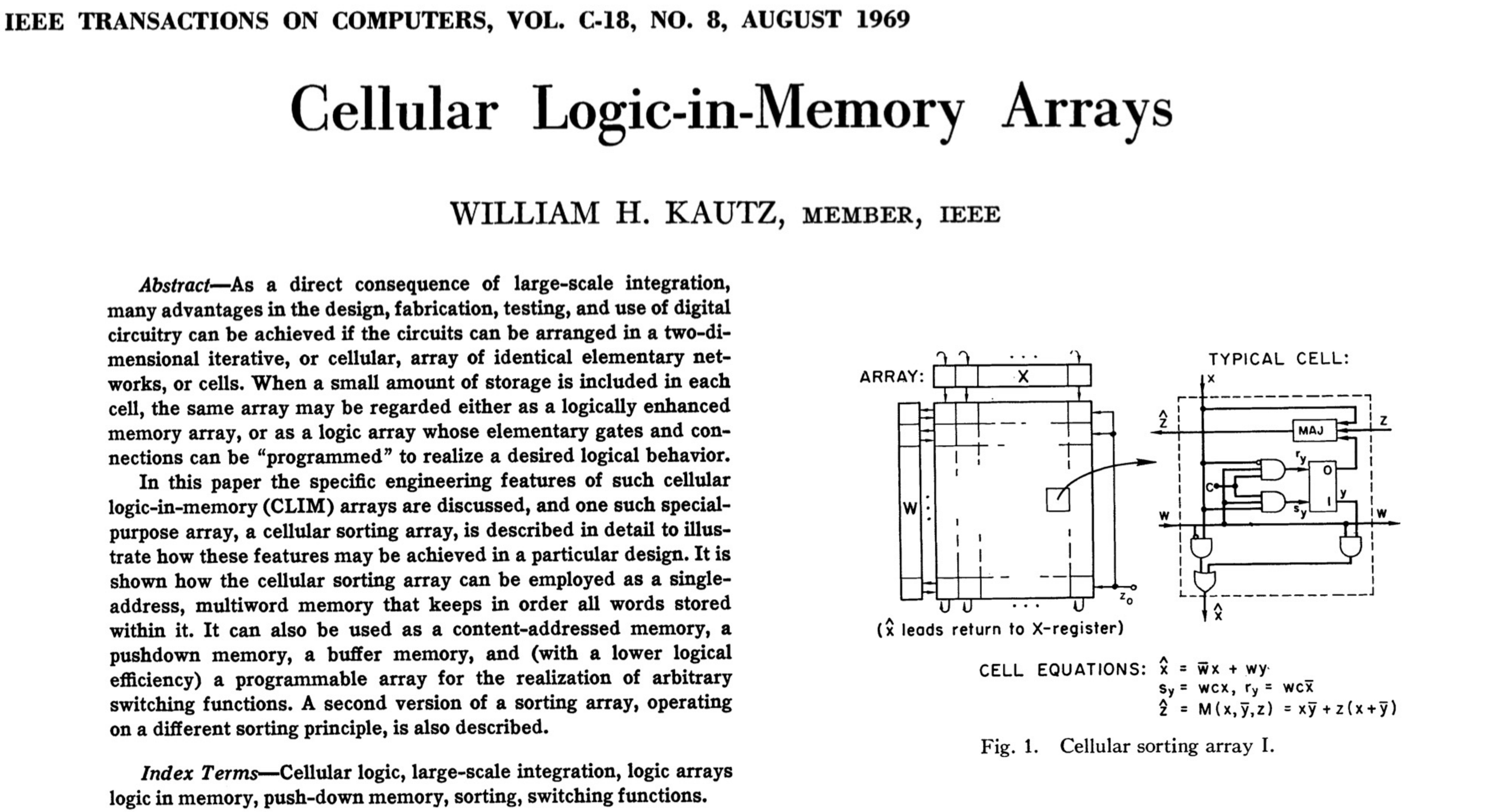
- You will learn what are key workload characteristics that make them more prone to the data movement bottleneck

- You will review traditional approaches to alleviating data movement and will get familiar with new research proposals and real systems: processing-in-memory solutions

- You will work hands-on: analyzing workloads, programming PIM architectures, simulating new PIM proposals, etc.

A +50-Year-Old Paradigm

- Kautz, "Cellular Logic-in-Memory Arrays", IEEE TC 1969



IEEE TRANSACTIONS ON COMPUTERS, VOL. C-18, NO. 8, AUGUST 1969

Cellular Logic-in-Memory Arrays

WILLIAM H. KAUTZ, MEMBER, IEEE

Abstract—As a direct consequence of large-scale integration, many advantages in the design, fabrication, testing, and use of digital circuitry can be achieved if the circuits can be arranged in a two-dimensional iterative, or cellular, array of identical elementary networks, or cells. When a small amount of storage is included in each cell, the same array may be regarded either as a logically enhanced memory array, or as a logic array whose elementary gates and connections can be "programmed" to realize a desired logical behavior.

In this paper the specific engineering features of such cellular logic-in-memory (CLIM) arrays are discussed, and one such special-purpose array, a cellular sorting array, is described in detail to illustrate how these features may be achieved in a particular design. It is shown how the cellular sorting array can be employed as a single-address, multiword memory that keeps in order all words stored within it. It can also be used as a content-addressed memory, a pushdown memory, a buffer memory, and (with a lower logical efficiency) a programmable array for the realization of arbitrary switching functions. A second version of a sorting array, operating on a different sorting principle, is also described.

Index Terms—Cellular logic, large-scale integration, logic arrays logic in memory, push-down memory, sorting, switching functions.

($\hat{x}$ leads return to X-register)

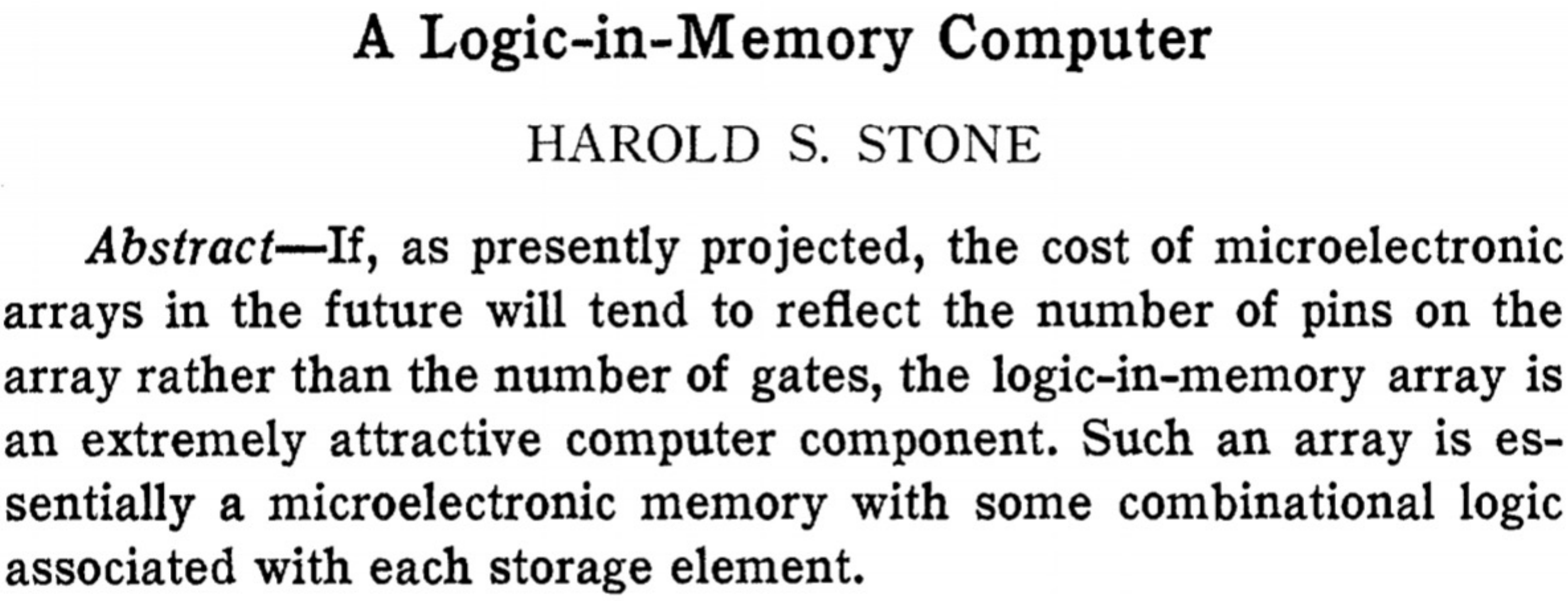
CELL EQUATIONS: $\hat{x} = \overline{w}x + wy$
$s_y = wcx, \; r_y = wc\overline{x}$
$\hat{z} = M(x, \overline{y}, z) = x\overline{y} + z(x + \overline{y})$

Fig. 1. Cellular sorting array I.

https://doi.org/10.1109/T-C.1969.222754

Processing in/near Memory: An Old Idea

- Stone, "A Logic-in-Memory Computer," IEEE TC 1970



A Logic-in-Memory Computer

HAROLD S. STONE

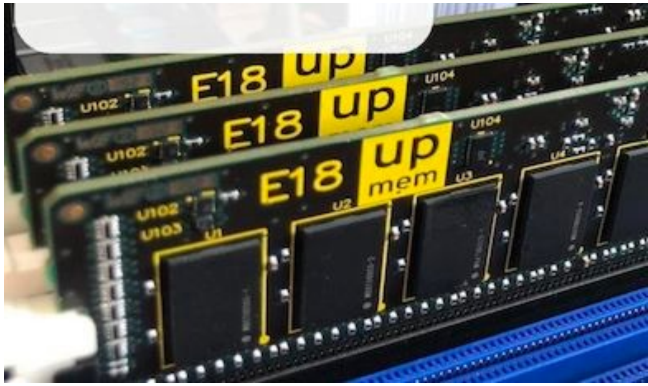
Abstract—If, as presently projected, the cost of microelectronic arrays in the future will tend to reflect the number of pins on the array rather than the number of gates, the logic-in-memory array is an extremely attractive computer component. Such an array is essentially a microelectronic memory with some combinational logic associated with each storage element.

https://doi.org/10.1109/TC.1970.5008902

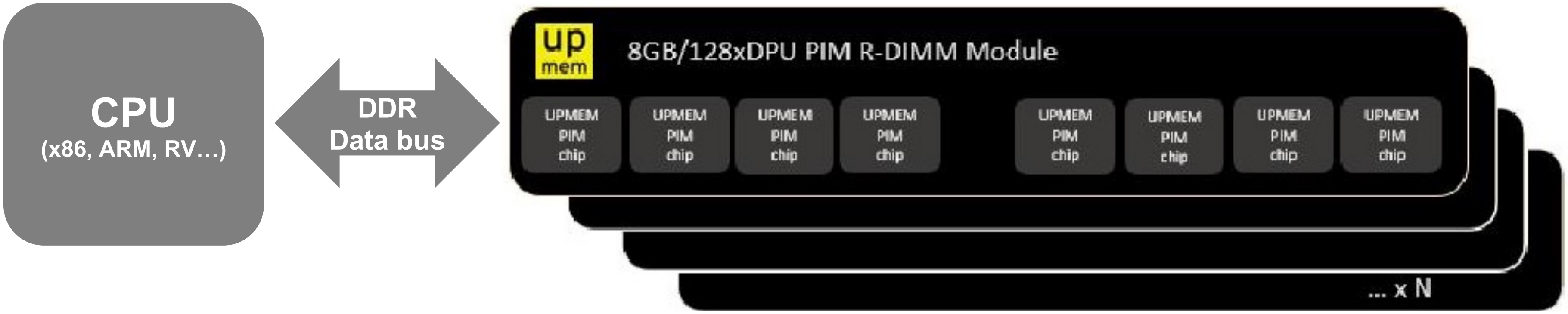
https://people.inf.ethz.ch/omutlu/pub/onur-MiM-Talk-IntelligentArchitecturesForIntelligentMachines-May-3-2021.pptx

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.

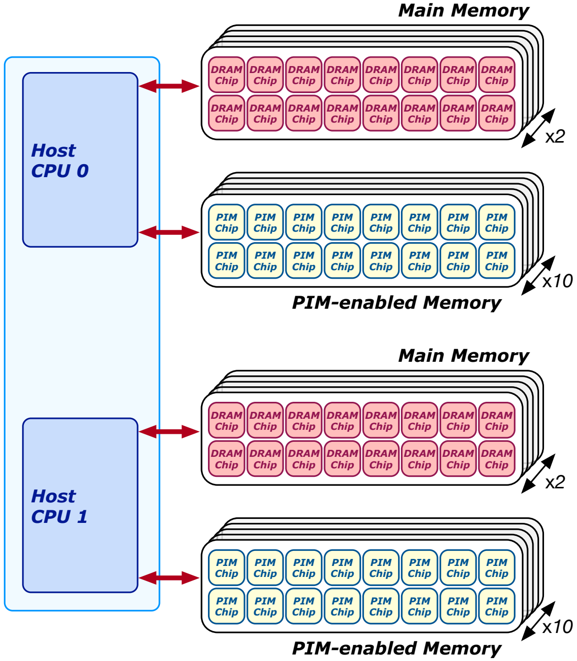


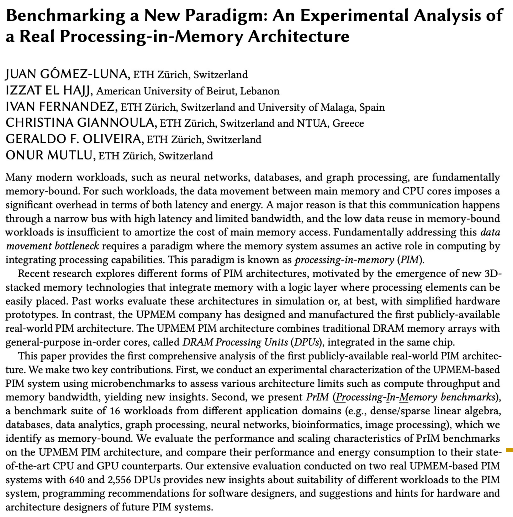
- Replaces **standard** DIMMs
  - DDR4 R-DIMM modules
    - 8GB+128 DPUs (16 PIM chips)
    - Standard 2x-nm DRAM process
  - **Large amounts of** compute & memory bandwidth



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/

2,560-DPU Processing-in-Memory System





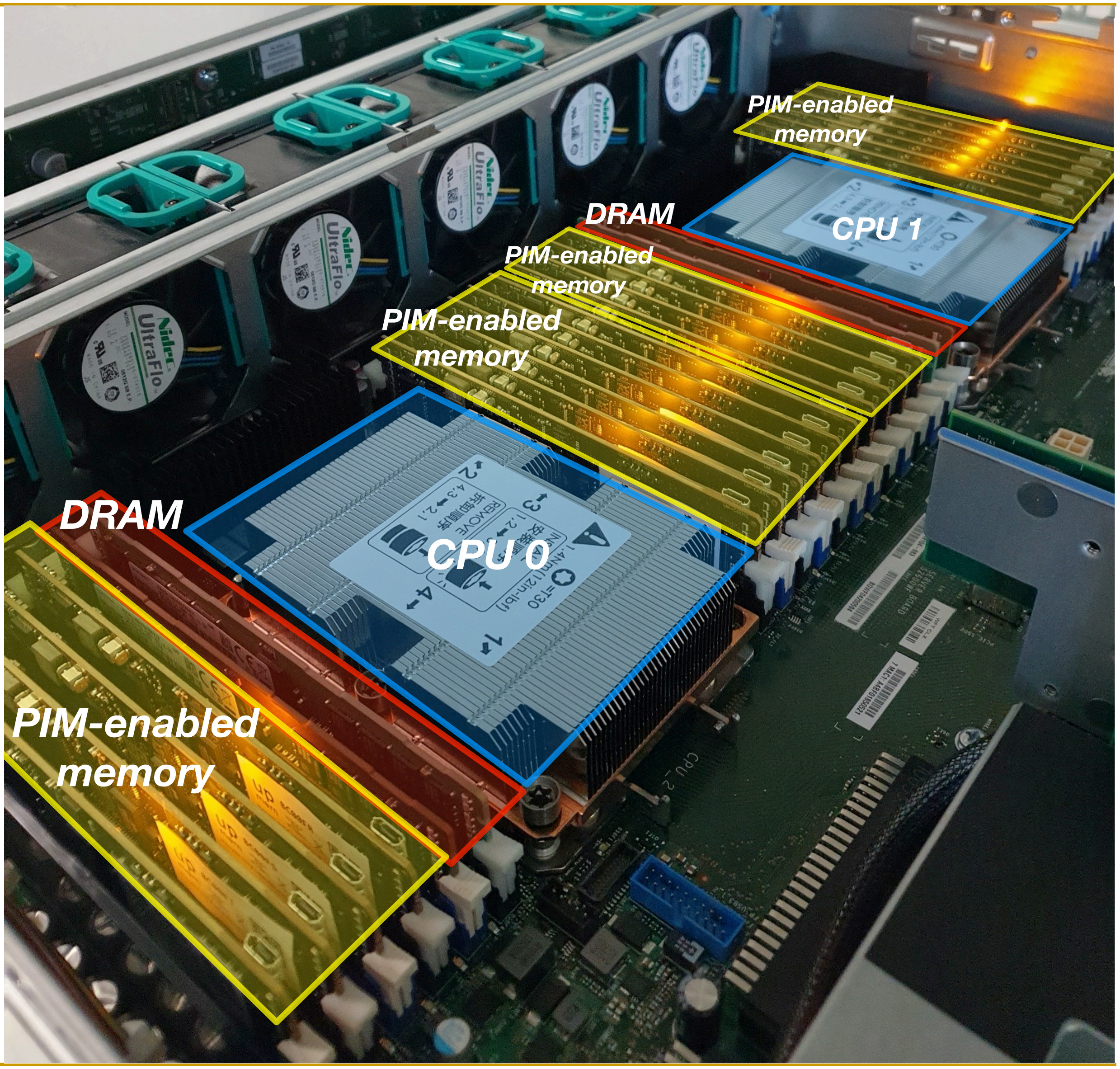
Benchmarking a New Paradigm: An Experimental Analysis of a Real Processing-in-Memory Architecture

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
IZZAT EL HAJJ, American University of Beirut, Lebanon
IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain
CHRISTINA GIANNOULA, ETH Zürich, Switzerland and NTUA, Greece
GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

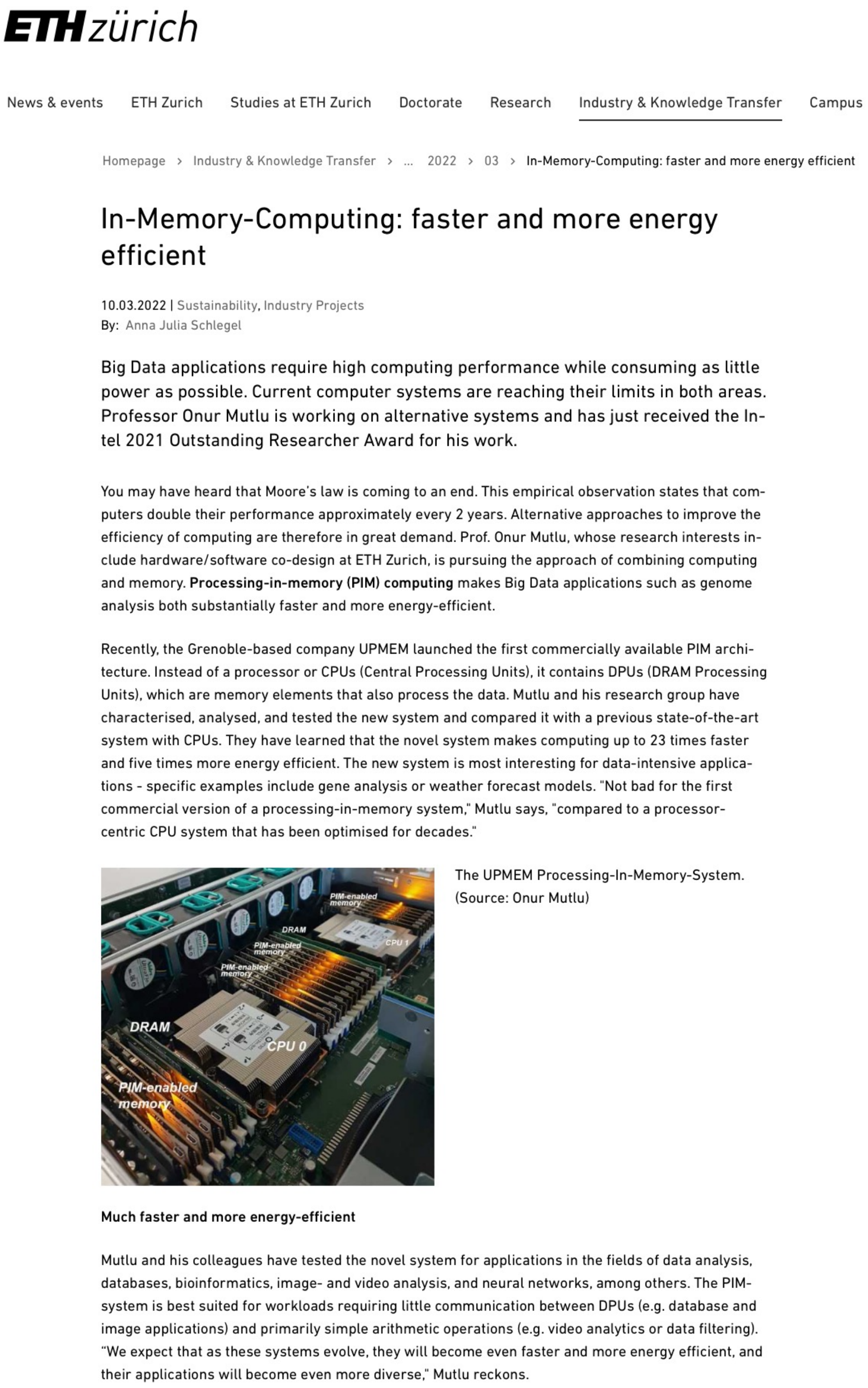
Many modern workloads, such as neural networks, databases, and graph processing, are fundamentally memory-bound. For such workloads, the data movement between main memory and CPU cores imposes a significant overhead in terms of both latency and energy. A major reason is that this communication happens through a narrow bus with high latency and limited bandwidth, and the low data reuse in memory-bound workloads is insufficient to amortize the cost of main memory access. Fundamentally addressing this *data movement bottleneck* requires a paradigm where the memory system assumes an active role in computing by integrating processing capabilities. This paradigm is known as *processing-in-memory* (*PIM*).

Recent research explores different forms of PIM architectures, motivated by the emergence of new 3D-stacked memory technologies that integrate memory with a logic layer where processing elements can be easily placed. Past works evaluate these architectures in simulation or, at best, with simplified hardware prototypes. In contrast, the UPMEM company has designed and manufactured the first publicly-available real-world PIM architecture. The UPMEM PIM architecture combines traditional DRAM memory arrays with general-purpose in-order cores, called *DRAM Processing Units* (*DPUs*), integrated in the same chip.

This paper provides the first comprehensive analysis of the first publicly-available real-world PIM architecture. We make two key contributions. First, we conduct an experimental characterization of the UPMEM-based PIM system using microbenchmarks to assess various architecture limits such as compute throughput and memory bandwidth, yielding new insights. Second, we present *PrIM* (*Processing-In-Memory benchmarks*), a benchmark suite of 16 workloads from different application domains (e.g., dense/sparse linear algebra, databases, data analytics, graph processing, neural networks, bioinformatics, image processing), which we identify as memory-bound. We evaluate the performance and scaling characteristics of PrIM benchmarks on the UPMEM PIM architecture, and compare their performance and energy consumption to their state-of-the-art CPU and GPU counterparts. Our extensive evaluation conducted on two real UPMEM-based PIM systems with 640 and 2,556 DPUs provides new insights about suitability of different workloads to the PIM system, programming recommendations for software designers, and suggestions and hints for hardware and architecture designers of future PIM systems.



https://arxiv.org/pdf/2105.03814.pdf



In-Memory-Computing: faster and more energy efficient

10.03.2022 | Sustainability, Industry Projects
By: Anna Julia Schlegel

Big Data applications require high computing performance while consuming as little power as possible. Current computer systems are reaching their limits in both areas. Professor Onur Mutlu is working on alternative systems and has just received the Intel 2021 Outstanding Researcher Award for his work.

You may have heard that Moore's law is coming to an end. This empirical observation states that computers double their performance approximately every 2 years. Alternative approaches to improve the efficiency of computing are therefore in great demand. Prof. Onur Mutlu, whose research interests include hardware/software co-design at ETH Zurich, is pursuing the approach of combining computing and memory. **Processing-in-memory (PIM) computing** makes Big Data applications such as genome analysis both substantially faster and more energy-efficient.

Recently, the Grenoble-based company UPMEM launched the first commercially available PIM architecture. Instead of a processor or CPUs (Central Processing Units), it contains DPUs (DRAM Processing Units), which are memory elements that also process the data. Mutlu and his research group have characterised, analysed, and tested the new system and compared it with a previous state-of-the-art system with CPUs. They have learned that the novel system makes computing up to 23 times faster and five times more energy efficient. The new system is most interesting for data-intensive applications - specific examples include gene analysis or weather forecast models. "Not bad for the first commercial version of a processing-in-memory system," Mutlu says, "compared to a processor-centric CPU system that has been optimised for decades."

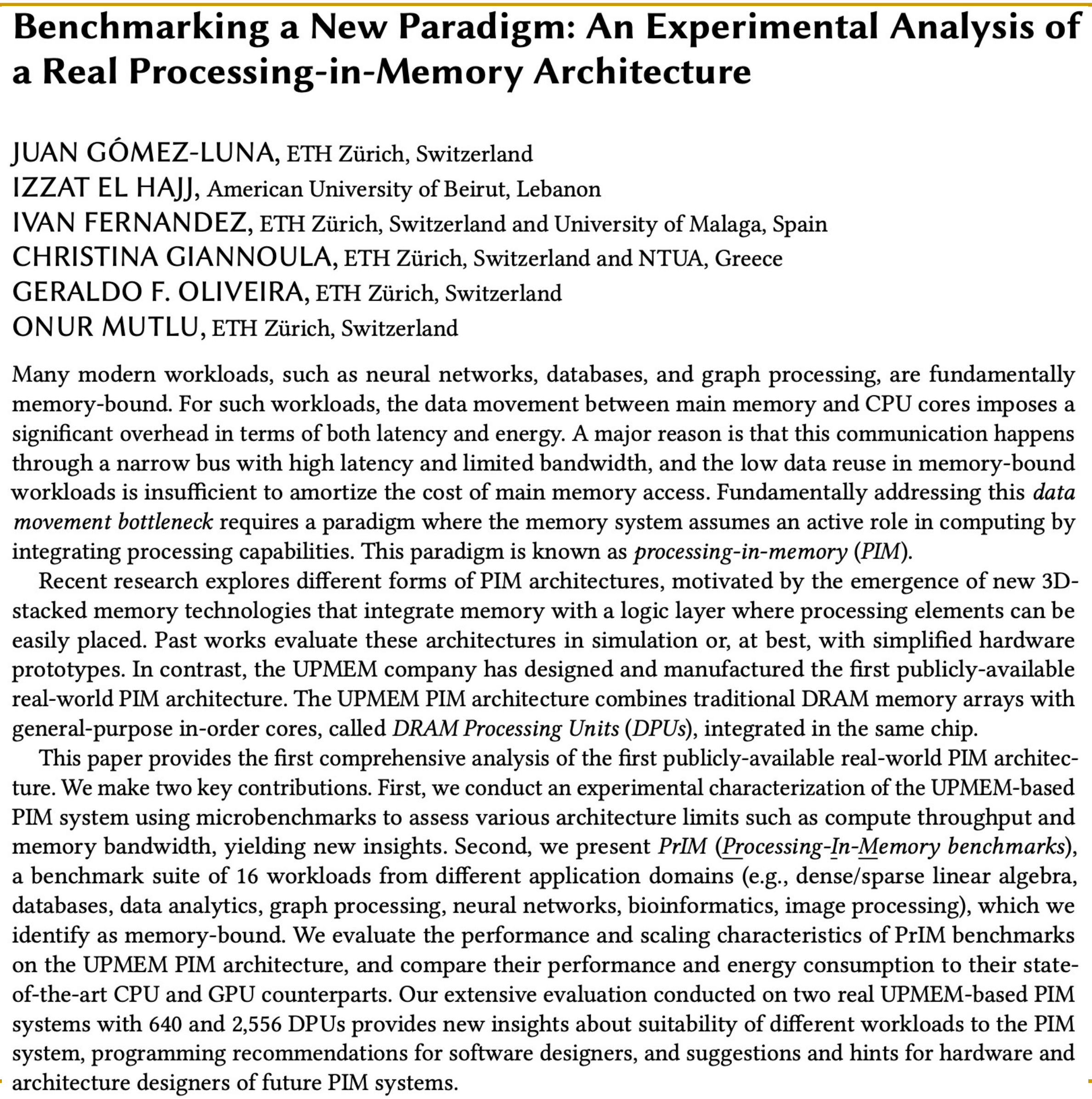
The UPMEM Processing-In-Memory-System. (Source: Onur Mutlu)

Much faster and more energy-efficient

Mutlu and his colleagues have tested the novel system for applications in the fields of data analysis, databases, bioinformatics, image- and video analysis, and neural networks, among others. The PIM-system is best suited for workloads requiring little communication between DPUs (e.g. database and image applications) and primarily simple arithmetic operations (e.g. video analytics or data filtering). "We expect that as these systems evolve, they will become even faster and more energy efficient, and their applications will become even more diverse," Mutlu reckons.

https://ethz.ch/en/industry/industry/news/data/2022/03/mehr-daten-schneller-und-energiesparender-verarbeiten.html

Experimental Analysis of the UPMEM PIM Engine



Benchmarking a New Paradigm: An Experimental Analysis of a Real Processing-in-Memory Architecture

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
IZZAT EL HAJJ, American University of Beirut, Lebanon
IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain
CHRISTINA GIANNOULA, ETH Zürich, Switzerland and NTUA, Greece
GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

Many modern workloads, such as neural networks, databases, and graph processing, are fundamentally memory-bound. For such workloads, the data movement between main memory and CPU cores imposes a significant overhead in terms of both latency and energy. A major reason is that this communication happens through a narrow bus with high latency and limited bandwidth, and the low data reuse in memory-bound workloads is insufficient to amortize the cost of main memory access. Fundamentally addressing this *data movement bottleneck* requires a paradigm where the memory system assumes an active role in computing by integrating processing capabilities. This paradigm is known as *processing-in-memory* (*PIM*).

Recent research explores different forms of PIM architectures, motivated by the emergence of new 3D-stacked memory technologies that integrate memory with a logic layer where processing elements can be easily placed. Past works evaluate these architectures in simulation or, at best, with simplified hardware prototypes. In contrast, the UPMEM company has designed and manufactured the first publicly-available real-world PIM architecture. The UPMEM PIM architecture combines traditional DRAM memory arrays with general-purpose in-order cores, called *DRAM Processing Units* (*DPUs*), integrated in the same chip.

This paper provides the first comprehensive analysis of the first publicly-available real-world PIM architecture. We make two key contributions. First, we conduct an experimental characterization of the UPMEM-based PIM system using microbenchmarks to assess various architecture limits such as compute throughput and memory bandwidth, yielding new insights. Second, we present *PrIM* (*Processing-In-Memory benchmarks*), a benchmark suite of 16 workloads from different application domains (e.g., dense/sparse linear algebra, databases, data analytics, graph processing, neural networks, bioinformatics, image processing), which we identify as memory-bound. We evaluate the performance and scaling characteristics of PrIM benchmarks on the UPMEM PIM architecture, and compare their performance and energy consumption to their state-of-the-art CPU and GPU counterparts. Our extensive evaluation conducted on two real UPMEM-based PIM systems with 640 and 2,556 DPUs provides new insights about suitability of different workloads to the PIM system, programming recommendations for software designers, and suggestions and hints for hardware and architecture designers of future PIM systems.

https://arxiv.org/pdf/2105.03814.pdf

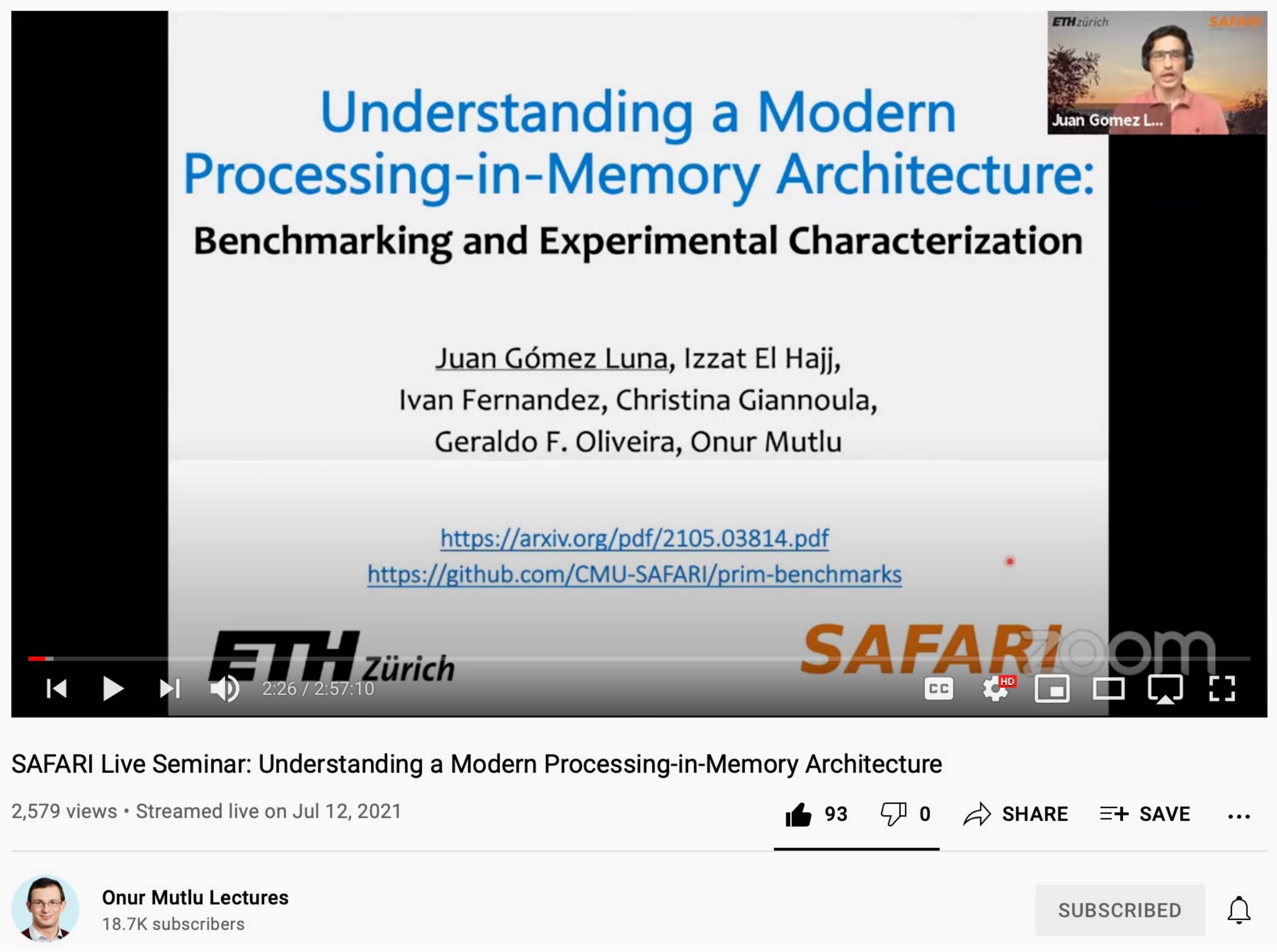
UPMEM PIM System Summary

- Juan Gomez-Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F. Oliveira, and Onur Mutlu,
  "Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-in-Memory Hardware"
  *Invited Paper at Workshop on Computing with Unconventional Technologies (**CUT**)*, Virtual, October 2021.
  [arXiv version]
  [PrIM Benchmarks Source Code]
  [Slides (pptx) (pdf)]
  [Talk Video (37 minutes)]
  [Lightning Talk Video (3 minutes)]

Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-in-Memory Hardware

Juan Gómez-Luna, *ETH Zürich*
Izzat El Hajj, *American University of Beirut*
Ivan Fernandez, *University of Malaga*
Christina Giannoula, *National Technical University of Athens*
Geraldo F. Oliveira, *ETH Zürich*
Onur Mutlu, *ETH Zürich*

Understanding a Modern PIM Architecture



Understanding a Modern Processing-in-Memory Architecture:

Benchmarking and Experimental Characterization

Juan Gómez Luna, Izzat El Hajj,
Ivan Fernandez, Christina Giannoula,
Geraldo F. Oliveira, Onur Mutlu

https://arxiv.org/pdf/2105.03814.pdf
https://github.com/CMU-SAFARI/prim-benchmarks

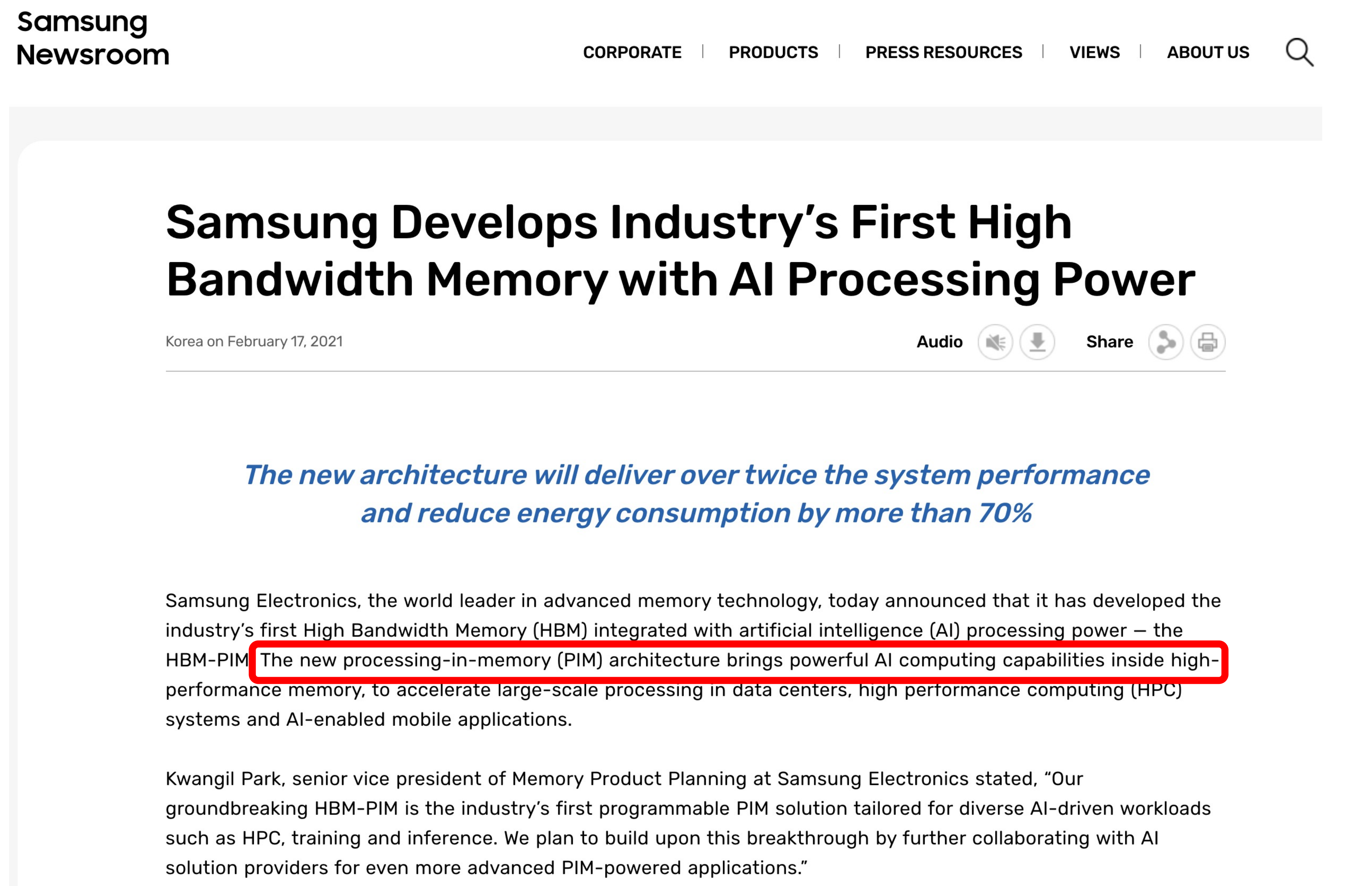
SAFARI Live Seminar: Understanding a Modern Processing-in-Memory Architecture

2,579 views • Streamed live on Jul 12, 2021

Onur Mutlu Lectures
18.7K subscribers

https://www.youtube.com/watch?v=D8Hjy2iU9l4&list=PL5Q2soXY2Zi_tOTAYm--dYByNPL7JhwR9

Samsung Function-in-Memory DRAM (2021)



Samsung Develops Industry's First High Bandwidth Memory with AI Processing Power

Korea on February 17, 2021

The new architecture will deliver over twice the system performance and reduce energy consumption by more than 70%

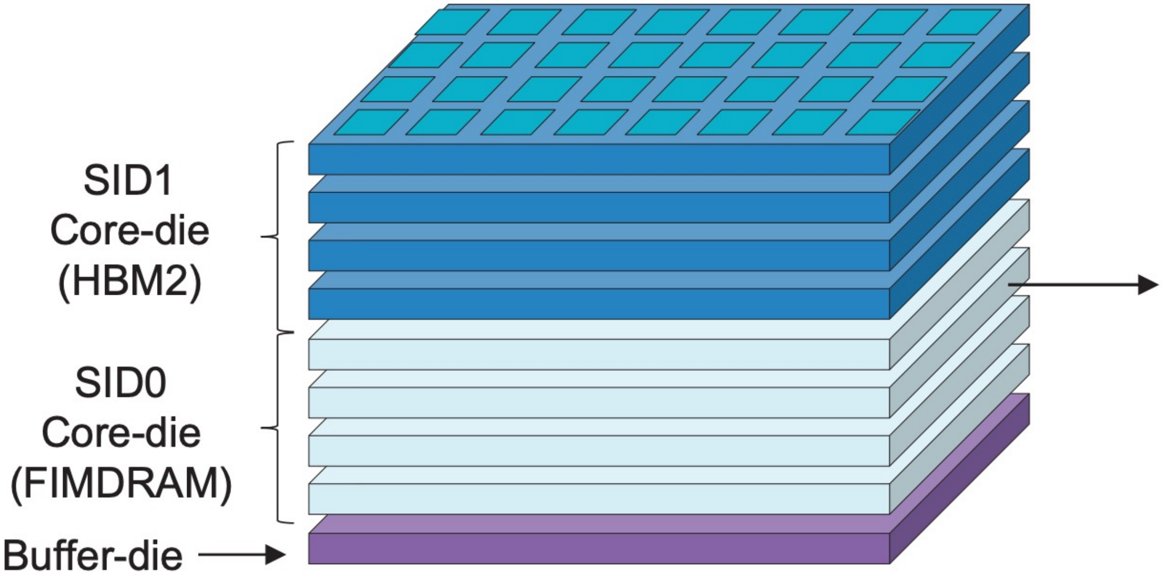
Samsung Electronics, the world leader in advanced memory technology, today announced that it has developed the industry's first High Bandwidth Memory (HBM) integrated with artificial intelligence (AI) processing power — the HBM-PIM. The new processing-in-memory (PIM) architecture brings powerful AI computing capabilities inside high-performance memory, to accelerate large-scale processing in data centers, high performance computing (HPC) systems and AI-enabled mobile applications.

Kwangil Park, senior vice president of Memory Product Planning at Samsung Electronics stated, "Our groundbreaking HBM-PIM is the industry's first programmable PIM solution tailored for diverse AI-driven workloads such as HPC, training and inference. We plan to build upon this breakthrough by further collaborating with AI solution providers for even more advanced PIM-powered applications."

https://news.samsung.com/global/samsung-develops-industrys-first-high-bandwidth-memory-with-ai-processing-power

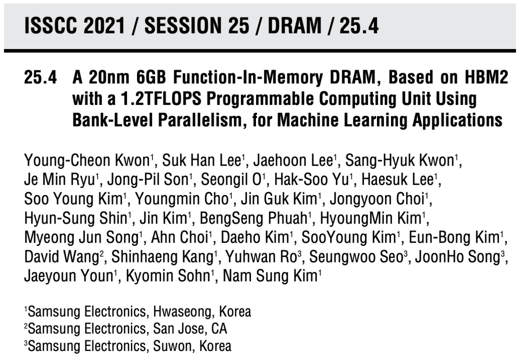
Samsung Function-in-Memory DRAM (2021)

- FIMDRAM based on HBM2



[3D Chip Structure of HBM with FIMDRAM]

Chip Specification
128DQ / 8CH / 16 banks / BL4
32 PCU blocks (1 FIM block/2 banks)
1.2 TFLOPS (4H)
FP16 ADD / Multiply (MUL) / Multiply-Accumulate (MAC) / Multiply-and- Add (MAD)



ISSCC 2021 / SESSION 25 / DRAM / 25.4

25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications

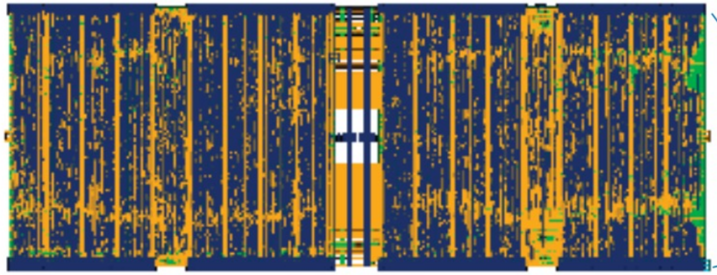
Young-Cheon Kwon[1], Suk Han Lee[1], Jaehoon Lee[1], Sang-Hyuk Kwon[1], Je Min Ryu[1], Jong-Pil Son[1], Seongil O[1], Hak-Soo Yu[1], Haesuk Lee[1], Soo Young Kim[1], Youngmin Cho[1], Jin Guk Kim[1], Jongyoon Choi[1], Hyun-Sung Shin[1], Jin Kim[1], BengSeng Phuah[1], HyoungMin Kim[1], Myeong Jun Song[1], Ahn Choi[1], Daeho Kim[1], SooYoung Kim[1], Eun-Bong Kim[1], David Wang[2], Shinhaeng Kang[1], Yuhwan Ro[3], Seungwoo Seo[3], JoonHo Song[3], Jaeyoun Youn[1], Kyomin Sohn[1], Nam Sung Kim[1]

[1]Samsung Electronics, Hwaseong, Korea
[2]Samsung Electronics, San Jose, CA
[3]Samsung Electronics, Suwon, Korea

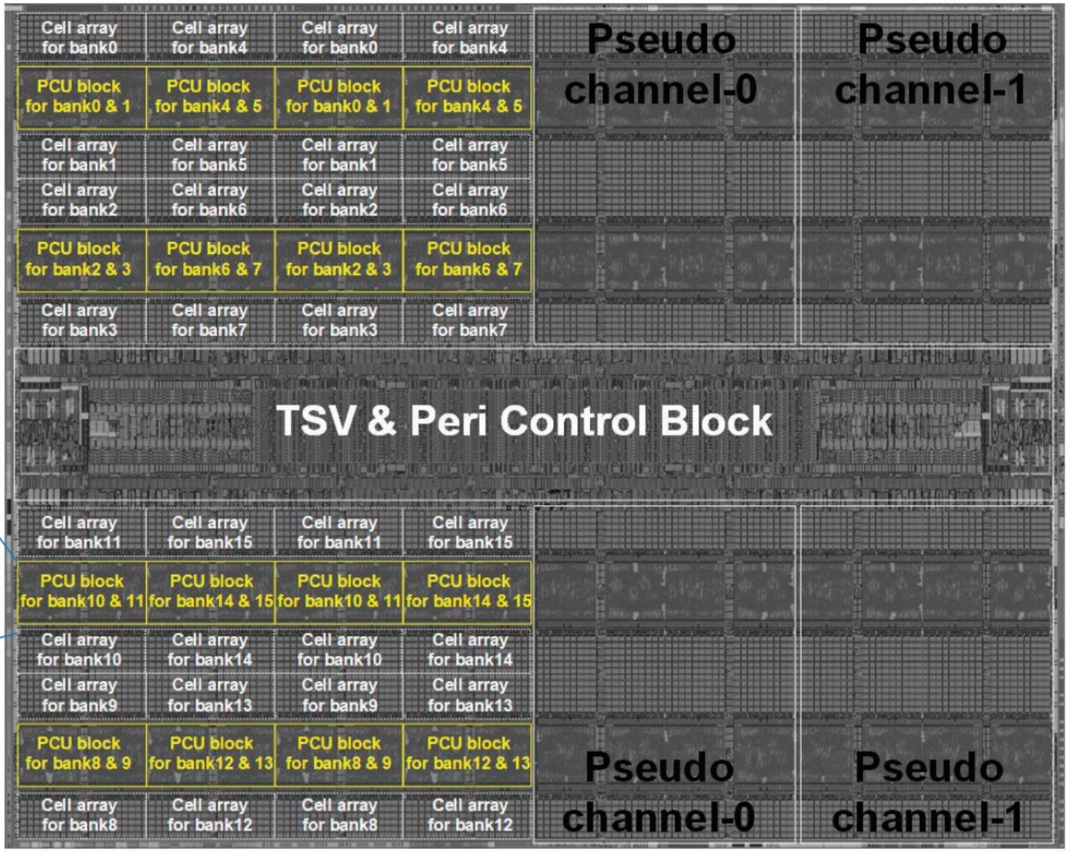
Samsung Function-in-Memory DRAM (2021)

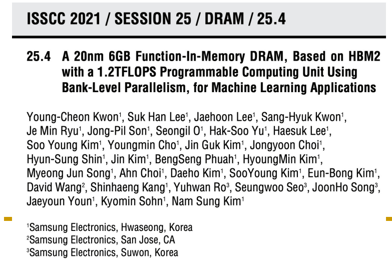
Chip Implementation

- Mixed design methodology to implement FIMDRAM
  - Full-custom + Digital RTL



[Digital RTL design for PCU block]





ISSCC 2021 / SESSION 25 / DRAM / 25.4

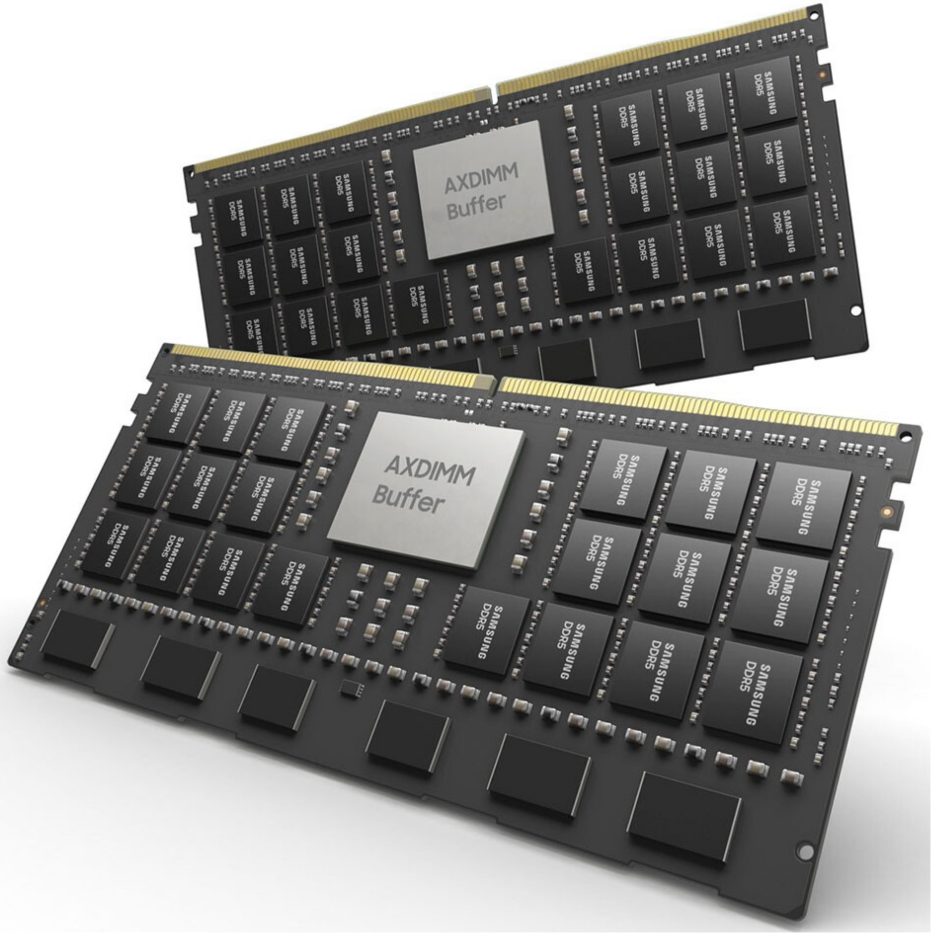
25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications

Young-Cheon Kwon[1], Suk Han Lee[1], Jaehoon Lee[1], Sang-Hyuk Kwon[1],
Je Min Ryu[1], Jong-Pil Son[1], Seongil O[1], Hak-Soo Yu[1], Haesuk Lee[1],
Soo Young Kim[1], Youngmin Cho[1], Jin Guk Kim[1], Jongyoon Choi[1],
Hyun-Sung Shin[1], Jin Kim[1], BengSeng Phuah[1], HyoungMin Kim[1],
Myeong Jun Song[1], Ahn Choi[1], Daeho Kim[1], SooYoung Kim[1], Eun-Bong Kim[1],
David Wang[2], Shinhaeng Kang[1], Yuhwan Ro[3], Seungwoo Seo[3], JoonHo Song[3],
Jaeyoun Youn[1], Kyomin Sohn[1], Nam Sung Kim[1]

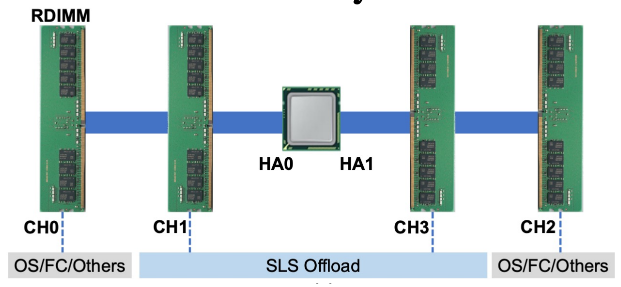
[1]Samsung Electronics, Hwaseong, Korea
[2]Samsung Electronics, San Jose, CA
[3]Samsung Electronics, Suwon, Korea

Samsung AxDIMM (2021)

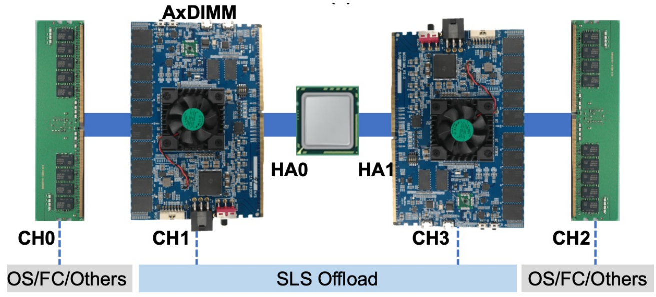
- DIMM-based PIM
  - DLRM recommendation system

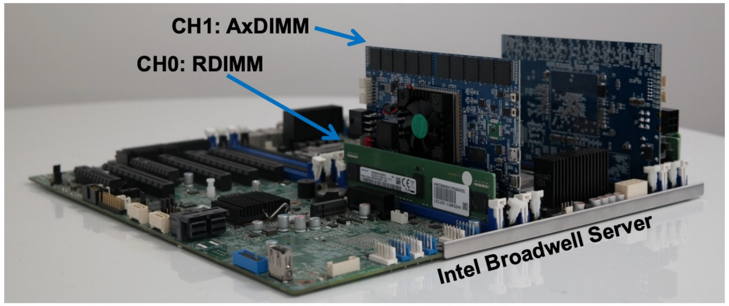


Baseline System



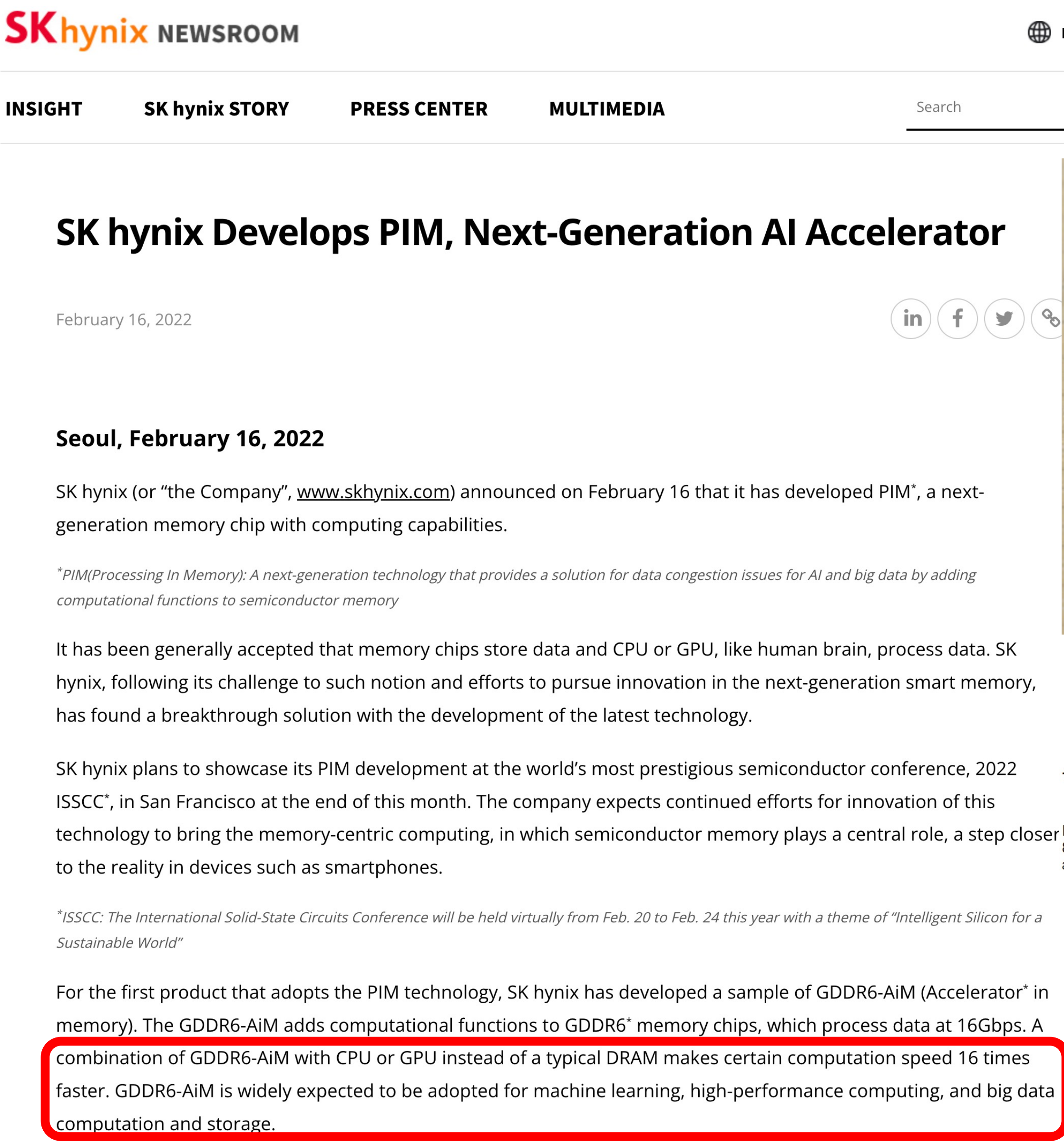
AxDIMM System





Ke et al. "Near-Memory Processing in Action: Accelerating Personalized Recommendation with AxDIMM", IEEE Micro (2021)

SK Hynix Accelerator-in-Memory (2022)



SK hynix Develops PIM, Next-Generation AI Accelerator

February 16, 2022

Seoul, February 16, 2022

SK hynix (or "the Company", www.skhynix.com) announced on February 16 that it has developed PIM*, a next-generation memory chip with computing capabilities.

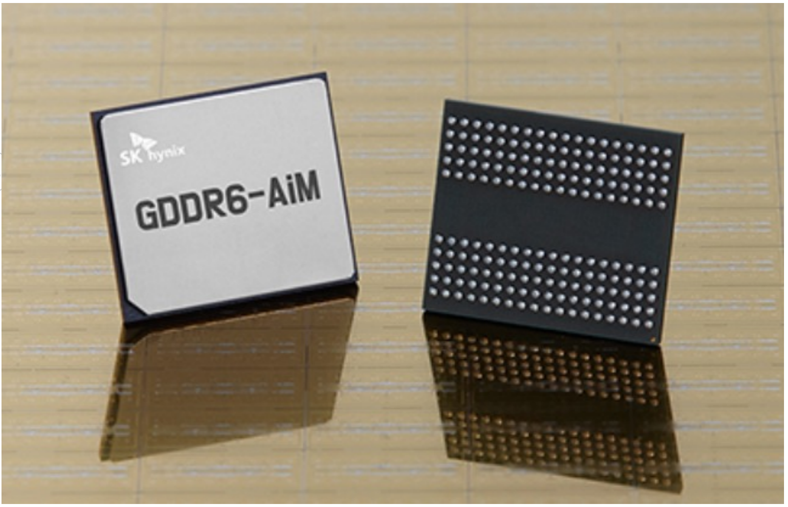
**PIM(Processing In Memory): A next-generation technology that provides a solution for data congestion issues for AI and big data by adding computational functions to semiconductor memory*

It has been generally accepted that memory chips store data and CPU or GPU, like human brain, process data. SK hynix, following its challenge to such notion and efforts to pursue innovation in the next-generation smart memory, has found a breakthrough solution with the development of the latest technology.

SK hynix plans to showcase its PIM development at the world's most prestigious semiconductor conference, 2022 ISSCC*, in San Francisco at the end of this month. The company expects continued efforts for innovation of this technology to bring the memory-centric computing, in which semiconductor memory plays a central role, a step closer to the reality in devices such as smartphones.

**ISSCC: The International Solid-State Circuits Conference will be held virtually from Feb. 20 to Feb. 24 this year with a theme of "Intelligent Silicon for a Sustainable World"*

For the first product that adopts the PIM technology, SK hynix has developed a sample of GDDR6-AiM (Accelerator* in memory). The GDDR6-AiM adds computational functions to GDDR6* memory chips, which process data at 16Gbps. A combination of GDDR6-AiM with CPU or GPU instead of a typical DRAM makes certain computation speed 16 times faster. GDDR6-AiM is widely expected to be adopted for machine learning, high-performance computing, and big data computation and storage.



11.1 A 1ynm 1.25V 8Gb, 16Gb/s/pin GDDR6-based Accelerator-in-Memory supporting 1TFLOPS MAC Operation and Various Activation Functions for Deep-Learning Applications

Seongju Lee, SK hynix, Icheon, Korea

In Paper 11.1, SK Hynix describes an 1ynm, GDDR6-based accelerator-in-memory with a command set for deep-learning operation. The 8Gb design achieves a peak throughput of 1TFLOPS with 1GHz MAC operations and supports major activation functions to improve accuracy.

https://news.skhynix.com/sk-hynix-develops-pim-next-generation-ai-accelerator/

Key Takeaways

- This P&S is aimed at improving your
  - Knowledge in Computer Architecture and Processing-in-Memory
  - Technical skills in programming parallel (PIM) architectures and CompArch simulation
  - Critical thinking and analysis
  - Interaction with a nice group of researchers
  - Familiarity with key research directions
  - Technical presentation of your project

Key Goal

(Learn how to) overcome

the data movement bottleneck

by programming, benchmarking,

exploring different designs of

the PIM computing paradigm

Prerequisites of the Course

- Digital Design and Computer Architecture (or equivalent course)
  - https://safari.ethz.ch/digitaltechnik/spring2021/doku.php?id=schedule
  - https://safari.ethz.ch/digitaltechnik/spring2022/doku.php?id=schedule

- Familiarity with C/C++ programming
  - FPGA implementation or GPU programming (desirable)

- Interest in
  - future computer architectures and computing paradigms
  - discovering why things do or do not work and solving problems
  - making systems efficient and usable

Course Info: Who Are We? (I)

- Onur Mutlu
  - Full Professor @ ETH Zurich ITET (INFK), 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
  - …

Course Info: Who Are We? (II)

- Lead Supervisor:
  - Dr. Juan Gómez Luna

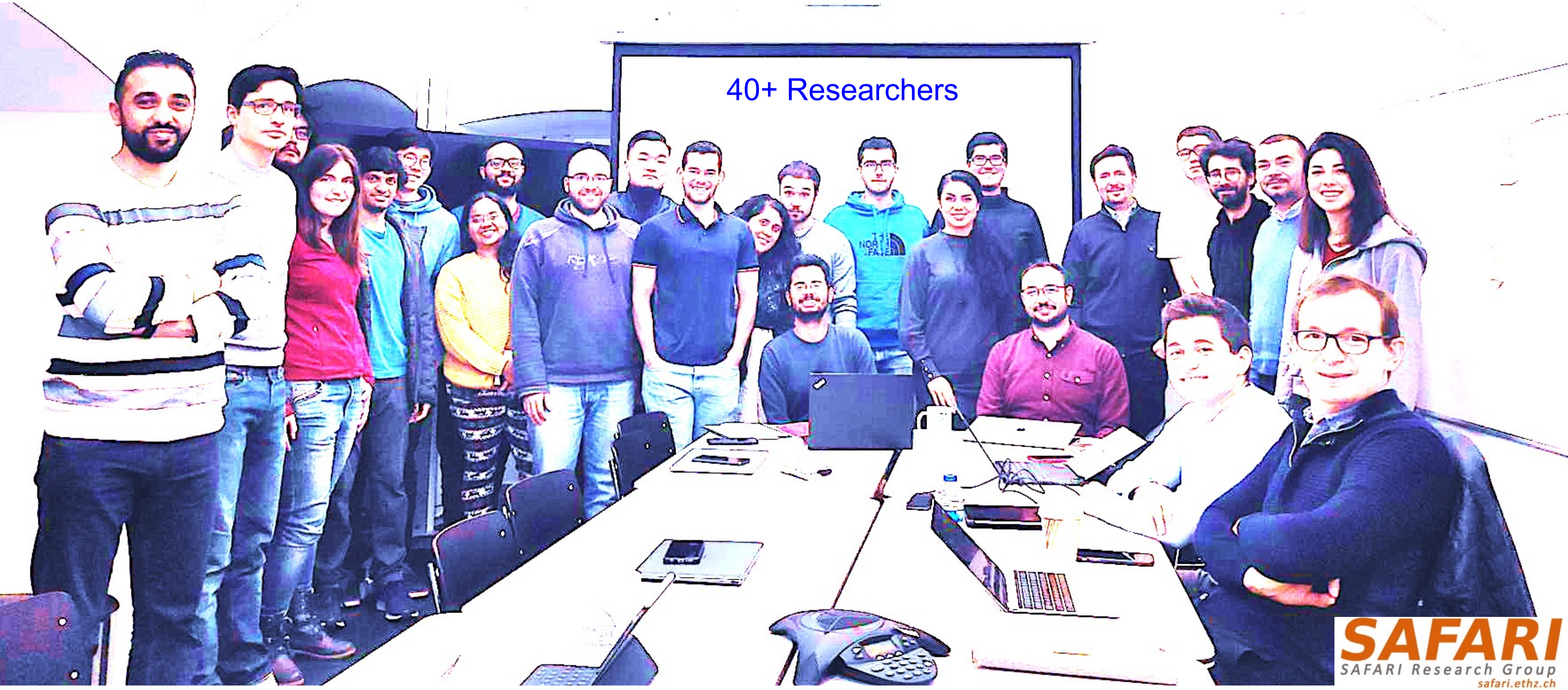
- Supervisors:
  - Dr. Haiyu Mao
  - Geraldo F. Oliveira
  - Konstantinos Kanellopoulos
  - Nika Mansouri Ghiasi

- Get to know us and our research
  - https://safari.ethz.ch/safari-group/

Onur Mutlu's SAFARI Research Group

Computer architecture, HW/SW, systems, bioinformatics, security, memory

https://safari.ethz.ch/safari-newsletter-january-2021/



Think BIG, Aim HIGH!

https://safari.ethz.ch

SAFARI Newsletter December 2021 Edition

- https://safari.ethz.ch/safari-newsletter-december-2021/

SAFARI

SAFARI Research Group

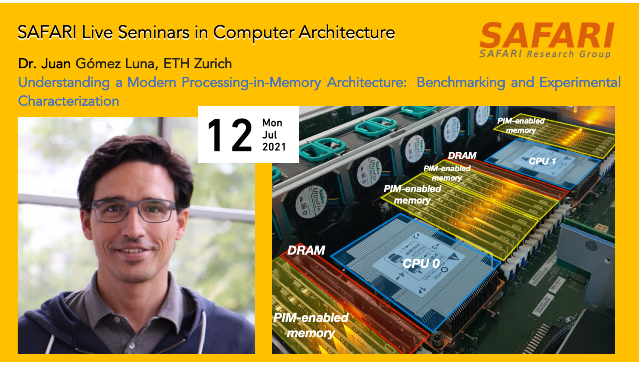
Think Big, Aim High

ETH zürich

View in your browser

December 2021

SAFARI Live Seminars (I)



SAFARI Live Seminars in Computer Architecture

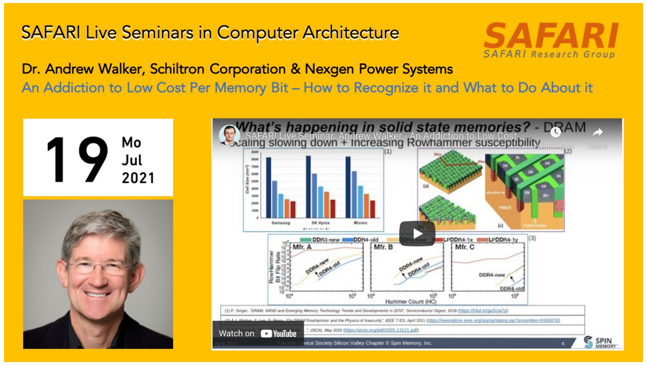
SAFARI
SAFARI Research Group

Dr. Juan Gómez Luna, ETH Zurich

Understanding a Modern Processing-in-Memory Architecture: Benchmarking and Experimental Characterization

12 Mon Jul 2021

PIM-enabled memory
DRAM
CPU 1
DRAM
PIM-enabled memory
PIM-enabled memory
DRAM
CPU 0
PIM-enabled memory



SAFARI Live Seminars in Computer Architecture

SAFARI
SAFARI Research Group

Dr. Andrew Walker, Schiltron Corporation & Nexgen Power Systems

An Addiction to Low Cost Per Memory Bit – How to Recognize it and What to Do About it

19 Mo Jul 2021

What's happening in solid state memories? - DRAM
Scaling slowing down + Increasing Rowhammer susceptibility

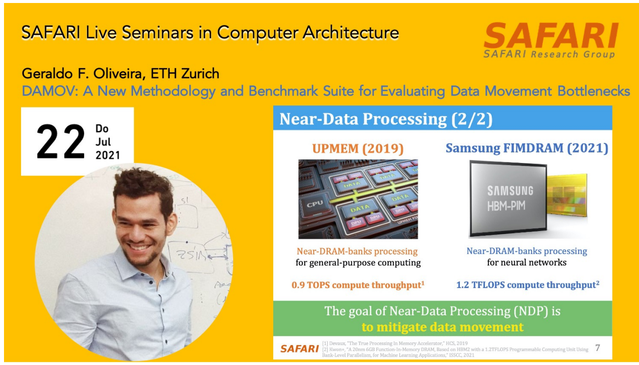
SAFARI Live Seminar: Andrew Walker - An Addiction to Low Cost ...

(1) P. Singer, "DRAM, NAND and Emerging Memory Technology Trends and Developments in 2019", Semiconductor Digest, 2019 (https://lnkd.in/guSiJa7p)

(2) A. J. Walker, S. Lee, D. Beery, "On DRAM Rowhammer and the Physics of Insecurity", IEEE T-ED, April 2021 (https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9366976)

(3) ..., (ISCA), May 2020 (https://arxiv.org/pdf/2005.13121.pdf)

April 2021 Electron Device Society Silicon Valley Chapter © Spin Memory, Inc. 4



SAFARI Live Seminars in Computer Architecture

SAFARI
SAFARI Research Group

Geraldo F. Oliveira, ETH Zurich

DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

22 Do Jul 2021

Near-Data Processing (2/2)

UPMEM (2019)

Near-DRAM-banks processing for general-purpose computing

0.9 TOPS compute throughput[1]

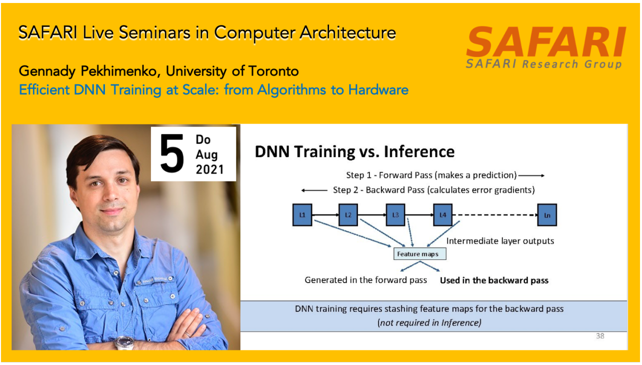
Samsung FIMDRAM (2021)

Near-DRAM-banks processing for neural networks

1.2 TFLOPS compute throughput[2]

The goal of Near-Data Processing (NDP) is to mitigate data movement

SAFARI [1] Devaux, "The True Processing In Memory Accelerator," HCS, 2019
[2] Kwon+, "A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications," ISSCC, 2021 7



SAFARI Live Seminars in Computer Architecture

SAFARI
SAFARI Research Group

Gennady Pekhimenko, University of Toronto

Efficient DNN Training at Scale: from Algorithms to Hardware

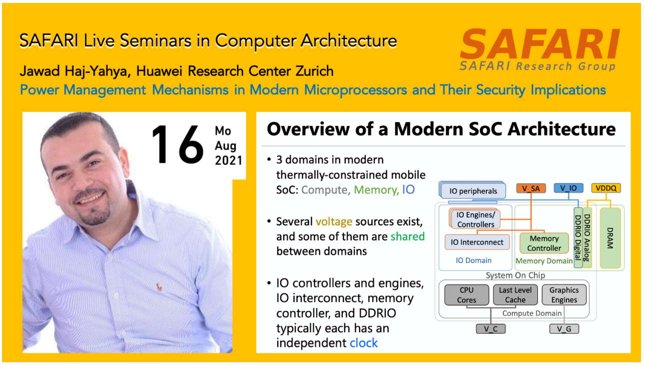
5 Do Aug 2021

DNN Training vs. Inference

Step 1 - Forward Pass (makes a prediction)
Step 2 - Backward Pass (calculates error gradients)
L1
L2
L3
L4
Ln
Intermediate layer outputs
Feature maps
Generated in the forward pass
Used in the backward pass

DNN training requires stashing feature maps for the backward pass
(not required in Inference)

38



SAFARI Live Seminars in Computer Architecture

SAFARI
SAFARI Research Group

Jawad Haj-Yahya, Huawei Research Center Zurich

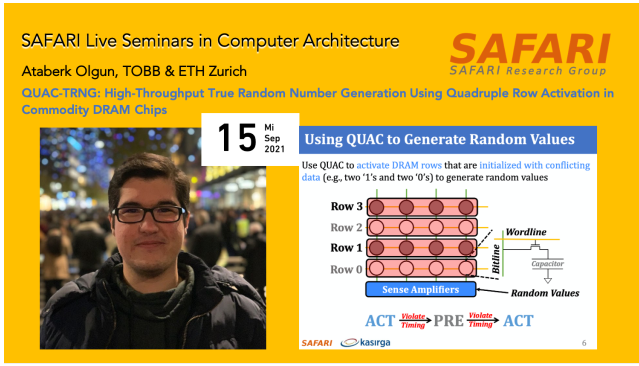
Power Management Mechanisms in Modern Microprocessors and Their Security Implications

16 Mo Aug 2021

Overview of a Modern SoC Architecture

- 3 domains in modern thermally-constrained mobile SoC: Compute, Memory, IO
- Several voltage sources exist, and some of them are shared between domains
- IO controllers and engines, IO interconnect, memory controller, and DDRIO typically each has an independent clock

IO peripherals
V_SA
V_IO
VDDQ
IO Engines/ Controllers
IO Interconnect
IO Domain
Memory Controller
Memory Domain
DDRIO Digital
DDRIO Analog
DRAM
System On Chip
CPU Cores
Last Level Cache
Graphics Engines
Compute Domain
V_C
V_G



SAFARI Live Seminars in Computer Architecture

SAFARI
SAFARI Research Group

Ataberk Olgun, TOBB & ETH Zurich

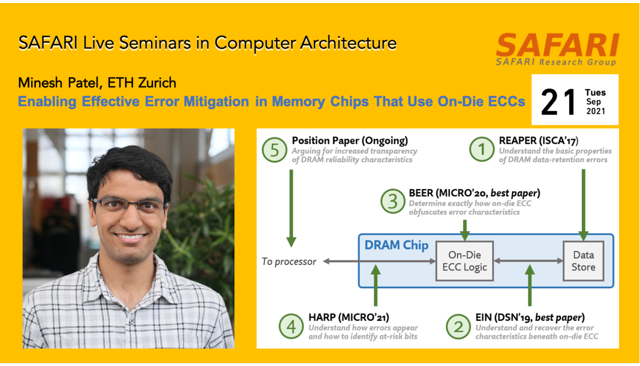
QUAC-TRNG: High-Throughput True Random Number Generation Using Quadruple Row Activation in Commodity DRAM Chips

15 Mi Sep 2021

Using QUAC to Generate Random Values

Use QUAC to activate DRAM rows that are initialized with conflicting data (e.g., two '1's and two '0's) to generate random values

SAFARI kasırga 6



SAFARI Live Seminars in Computer Architecture

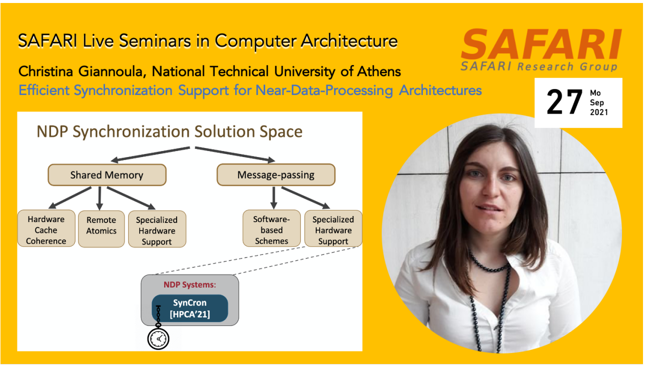
SAFARI
SAFARI Research Group

Minesh Patel, ETH Zurich

Enabling Effective Error Mitigation in Memory Chips That Use On-Die ECCs

21 Tues Sep 2021

5
Position Paper (Ongoing)
Arguing for increased transparency of DRAM reliability characteristics
1
REAPER (ISCA'17)
Understand the basic properties of DRAM data-retention errors
3
BEER (MICRO'20, best paper)
Determine exactly how on-die ECC obfuscates error characteristics
DRAM Chip
To processor
On-Die ECC Logic
Data Store
4
HARP (MICRO'21)
Understand how errors appear and how to identify at-risk bits
2
EIN (DSN'19, best paper)
Understand and recover the error characteristics beneath on-die ECC



SAFARI Live Seminars in Computer Architecture

SAFARI
SAFARI Research Group

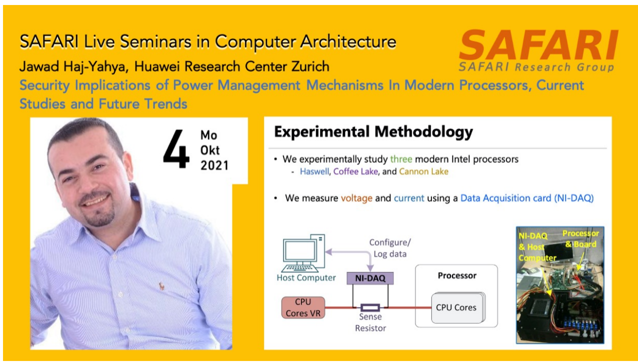
Christina Giannoula, National Technical University of Athens

Efficient Synchronization Support for Near-Data-Processing Architectures

27 Mo Sep 2021

NDP Synchronization Solution Space

Shared Memory
Message-passing
Hardware Cache Coherence
Remote Atomics
Specialized Hardware Support
Software-based Schemes
Specialized Hardware Support
NDP Systems:
SynCron [HPCA'21]



SAFARI Live Seminars in Computer Architecture

SAFARI
SAFARI Research Group

Jawad Haj-Yahya, Huawei Research Center Zurich

Security Implications of Power Management Mechanisms In Modern Processors, Current Studies and Future Trends

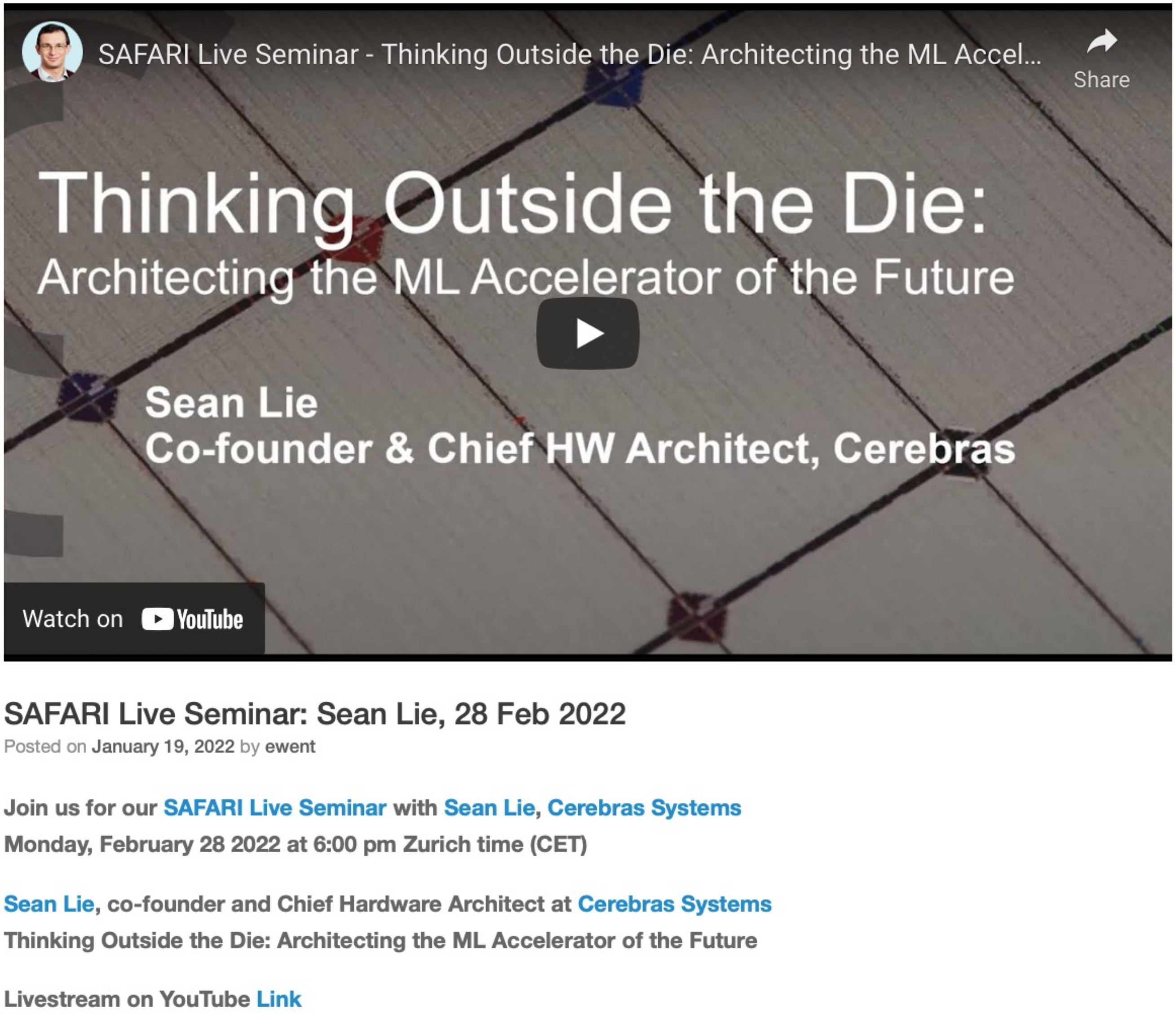
4 Mo Okt 2021

Experimental Methodology

- We experimentally study three modern Intel processors
  - Haswell, Coffee Lake, and Cannon Lake
- We measure voltage and current using a Data Acquisition card (NI-DAQ)

https://safari.ethz.ch/safari-seminar-series/

SAFARI Live Seminars (II)



SAFARI Live Seminar - Thinking Outside the Die: Architecting the ML Accel...

Thinking Outside the Die:
Architecting the ML Accelerator of the Future

Sean Lie
Co-founder & Chief HW Architect, Cerebras

SAFARI Live Seminar: Sean Lie, 28 Feb 2022

Posted on January 19, 2022 by ewent

Join us for our SAFARI Live Seminar with Sean Lie, Cerebras Systems
Monday, February 28 2022 at 6:00 pm Zurich time (CET)

Sean Lie, co-founder and Chief Hardware Architect at Cerebras Systems
Thinking Outside the Die: Architecting the ML Accelerator of the Future

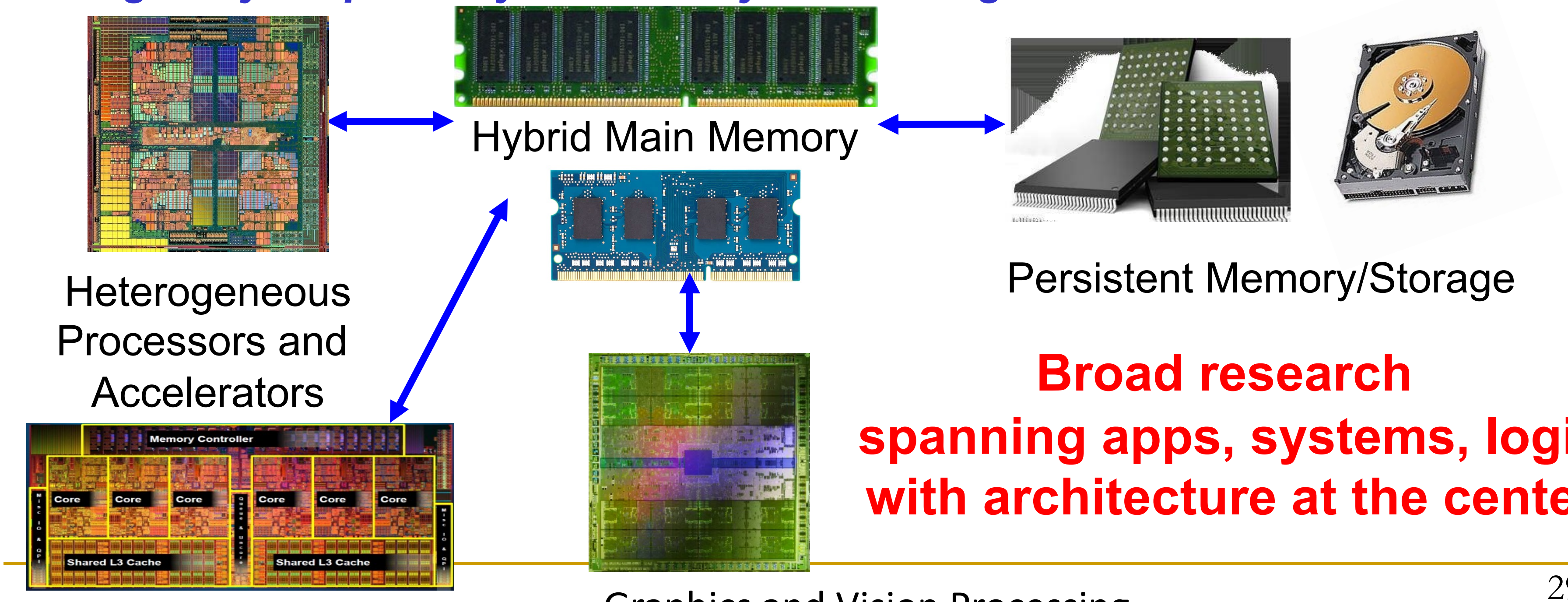
Livestream on YouTube Link

https://safari.ethz.ch/safari-live-seminar-sean-lie-28-feb-2022/

Current Research Focus Areas

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

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



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

Graphics and Vision Processing

Course Requirements and Expectations

- Attendance required for all meetings
- Study the learning materials
- Each student will carry out a hands-on project
  - Build, implement, code, and design with close engagement from the supervisors
- Participation
  - Ask questions, contribute thoughts/ideas
  - Read relevant papers

We will help in all projects!
If your work is really good, you may get it published!

Course Website

- https://safari.ethz.ch/projects_and_seminars/doku.php?id=processing_in_memory

- Useful information about the course

- Check your email frequently for announcements

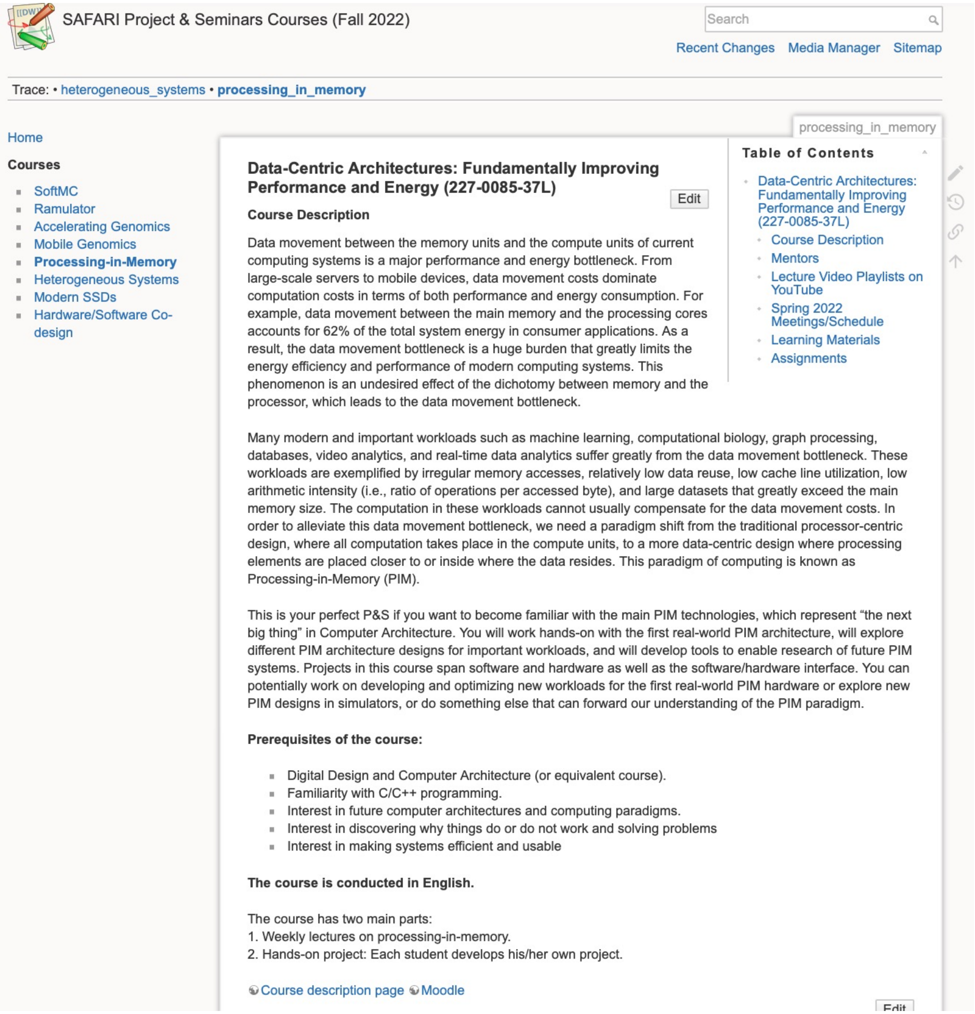
- We also have Moodle for Q&A

PIM Course (Current)

- **Fall 2022 Edition:**
  - https://safari.ethz.ch/projects_and_seminars/fall2022/doku.php?id=processing_in_memory

- **Youtube Livestream:**
  - https://youtube.com/playlist?list=PL5Q2soXY2Zi8KzG2CQYRNQOVD0GOBrnKy

- Project course
  - Taken by Bachelor’s/Master’s students
  - Processing-in-Memory lectures
  - Hands-on research exploration
  - Many research readings



SAFARI Project & Seminars Courses (Fall 2022)
Search
Recent Changes
Media Manager
Sitemap
Trace: • heterogeneous_systems • processing_in_memory
Home
Courses
SoftMC
Ramulator
Accelerating Genomics
Mobile Genomics
Processing-in-Memory
Heterogeneous Systems
Modern SSDs
Hardware/Software Co-design
processing_in_memory
Edit
Table of Contents
Data-Centric Architectures: Fundamentally Improving Performance and Energy (227-0085-37L)
Course Description
Mentors
Lecture Video Playlists on YouTube
Spring 2022 Meetings/Schedule
Learning Materials
Assignments

Data-Centric Architectures: Fundamentally Improving Performance and Energy (227-0085-37L)

Course Description

Data movement between the memory units and the compute units of current computing systems is a major performance and energy bottleneck. From large-scale servers to mobile devices, data movement costs dominate computation costs in terms of both performance and energy consumption. For example, data movement between the main memory and the processing cores accounts for 62% of the total system energy in consumer applications. As a result, the data movement bottleneck is a huge burden that greatly limits the energy efficiency and performance of modern computing systems. This phenomenon is an undesired effect of the dichotomy between memory and the processor, which leads to the data movement bottleneck.

Many modern and important workloads such as machine learning, computational biology, graph processing, databases, video analytics, and real-time data analytics suffer greatly from the data movement bottleneck. These workloads are exemplified by irregular memory accesses, relatively low data reuse, low cache line utilization, low arithmetic intensity (i.e., ratio of operations per accessed byte), and large datasets that greatly exceed the main memory size. The computation in these workloads cannot usually compensate for the data movement costs. In order to alleviate this data movement bottleneck, we need a paradigm shift from the traditional processor-centric design, where all computation takes place in the compute units, to a more data-centric design where processing elements are placed closer to or inside where the data resides. This paradigm of computing is known as Processing-in-Memory (PIM).

This is your perfect P&S if you want to become familiar with the main PIM technologies, which represent “the next big thing” in Computer Architecture. You will work hands-on with the first real-world PIM architecture, will explore different PIM architecture designs for important workloads, and will develop tools to enable research of future PIM systems. Projects in this course span software and hardware as well as the software/hardware interface. You can potentially work on developing and optimizing new workloads for the first real-world PIM hardware or explore new PIM designs in simulators, or do something else that can forward our understanding of the PIM paradigm.

Prerequisites of the course:

- Digital Design and Computer Architecture (or equivalent course).
- Familiarity with C/C++ programming.
- Interest in future computer architectures and computing paradigms.
- Interest in discovering why things do or do not work and solving problems
- Interest in making systems efficient and usable

The course is conducted in English.

The course has two main parts:
1. Weekly lectures on processing-in-memory.
2. Hands-on project: Each student develops his/her own project.

Course description page Moodle

Meeting 1: Learning Materials

- Required materials:

1. Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun,
"A Modern Primer on Processing in Memory"
Invited Book Chapter in Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann, Springer, to be published in 2021.
[Tutorial Video on "Memory-Centric Computing Systems" (1 hour 51 minutes)]

2. Onur Mutlu,
"Memory-Centric Computing"
Education Class at Embedded Systems Week (ESWEEK), Virtual, 9 October 2021.
[Slides (pptx) (pdf)]
[Abstract (pdf)]
[Talk Video (2 hours, including Q&A)]
[Invited Paper at DATE 2021]
["A Modern Primer on Processing in Memory" paper]

- Recommended materials:

3. Saugata Ghose, Amirali Boroumand, Jeremie S. Kim, Juan Gomez-Luna, and Onur Mutlu,
"Processing-in-Memory: A Workload-Driven Perspective"
Invited Article in IBM Journal of Research & Development, Special Issue on Hardware for Artificial Intelligence, November/December 2019. [Preliminary arXiv version]

4. Computation in Memory (Professor Onur Mutlu, lecture, Fall 2020).
(PDF) (PPT) Video

5. Near-data Processing (Professor Onur Mutlu, lecture, Fall 2020).
(PDF) (PPT) Video

6. Real Processing-in-DRAM with UPMEM (Dr. Juan Gomez Luna, SAFARI Live Seminar, July 2021).
"Benchmarking a New Paradigm: An Experimental Analysis of a Real Processing-in-Memory Architecture"
Preprint in arXiv, 9 May 2021. [arXiv preprint]
[PrIM Benchmarks Source Code]
[Slides (pptx) (pdf)]
[Long Talk Slides (pptx) (pdf)]
[Short Talk Slides (pptx) (pdf)]
[SAFARI Live Seminar Slides (pptx) (pdf)]
[SAFARI Live Seminar Video (2 hrs 57 mins)]
[Lightning Talk Video (3 minutes)]

Meeting 2 (March 15th)

- We will announce the projects and will give you some description about them
- We will give you a chance to select a project
- Then, we will have 1-1 meetings to match your interests, skills, and background with a suitable project
- It is important that you study the learning materials before our next meeting!

Next Meetings

- Individual meetings with your mentor/s

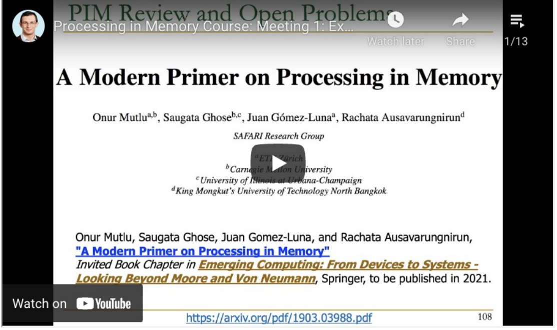
- Tutorials and short talks
  - PIM programming
  - Recent research works

- Presentation of your work

PIM Course (Spring 2022)

- **Spring 2022 Edition:**
  - https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=processing_in_memory
- **Youtube Livestream:**
  - https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi-841fUYYUK9EsXKhQKRPyX
- Project course
  - Taken by Bachelor's/Master's students
  - Processing-in-Memory lectures
  - Hands-on research exploration
  - Many research readings

https://www.youtube.com/onurmutlulectures



PIM Review and Open Problems

Processing in Memory Course: Meeting 1: Ex...

A Modern Primer on Processing in Memory

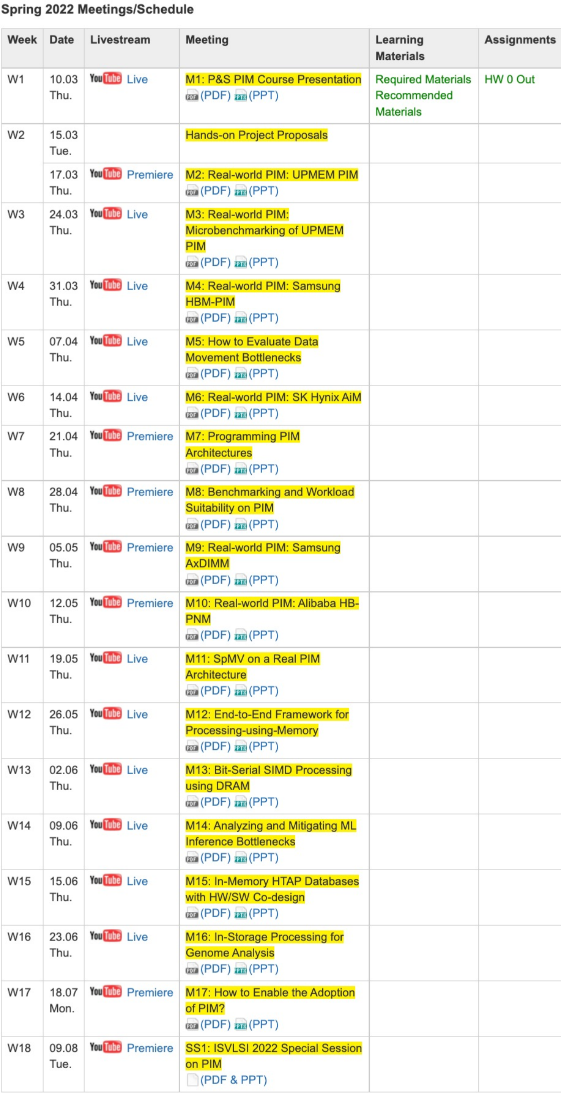
Onur Mutlu[a,b], Saugata Ghose[b,c], Juan Gómez-Luna[a], Rachata Ausavarungnirun[d]

SAFARI Research Group

[a]ETH Zürich
[b]Carnegie Mellon University
[c]University of Illinois at Urbana-Champaign
[d]King Mongkut's University of Technology North Bangkok

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun, **"A Modern Primer on Processing in Memory"** *Invited Book Chapter in* ***Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann***, Springer, to be published in 2021.

https://arxiv.org/pdf/1903.03988.pdf 108

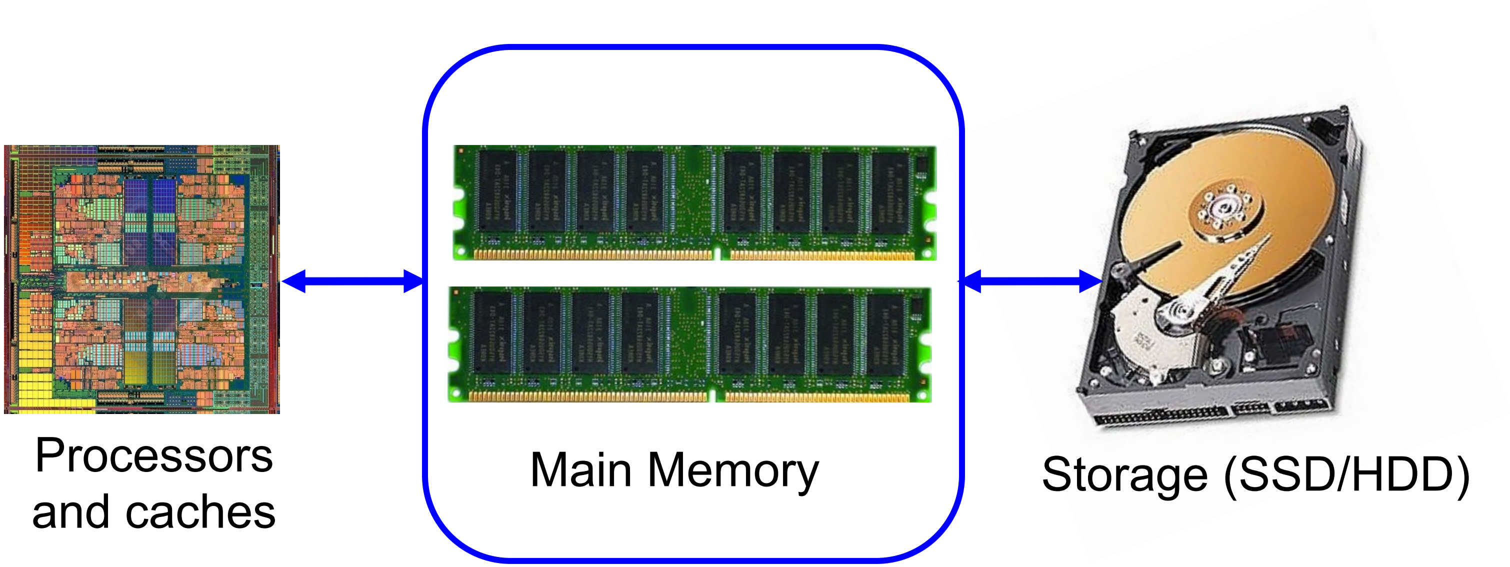


Spring 2022 Meetings/Schedule

Week	Date	Livestream	Meeting	Learning Materials	Assignments
W1	10.03 Thu.	YouTube Live	M1: P&S PIM Course Presentation (PDF) (PPT)	Required Materials Recommended Materials	HW 0 Out
W2	15.03 Tue.		Hands-on Project Proposals		
	17.03 Thu.	YouTube Premiere	M2: Real-world PIM: UPMEM PIM (PDF) (PPT)		
W3	24.03 Thu.	YouTube Live	M3: Real-world PIM: Microbenchmarking of UPMEM PIM (PDF) (PPT)		
W4	31.03 Thu.	YouTube Live	M4: Real-world PIM: Samsung HBM-PIM (PDF) (PPT)		
W5	07.04 Thu.	YouTube Live	M5: How to Evaluate Data Movement Bottlenecks (PDF) (PPT)		
W6	14.04 Thu.	YouTube Live	M6: Real-world PIM: SK Hynix AiM (PDF) (PPT)		
W7	21.04 Thu.	YouTube Premiere	M7: Programming PIM Architectures (PDF) (PPT)		
W8	28.04 Thu.	YouTube Premiere	M8: Benchmarking and Workload Suitability on PIM (PDF) (PPT)		
W9	05.05 Thu.	YouTube Premiere	M9: Real-world PIM: Samsung AxDIMM (PDF) (PPT)		
W10	12.05 Thu.	YouTube Premiere	M10: Real-world PIM: Alibaba HB-PNM (PDF) (PPT)		
W11	19.05 Thu.	YouTube Live	M11: SpMV on a Real PIM Architecture (PDF) (PPT)		
W12	26.05 Thu.	YouTube Live	M12: End-to-End Framework for Processing-using-Memory (PDF) (PPT)		
W13	02.06 Thu.	YouTube Live	M13: Bit-Serial SIMD Processing using DRAM (PDF) (PPT)		
W14	09.06 Thu.	YouTube Live	M14: Analyzing and Mitigating ML Inference Bottlenecks (PDF) (PPT)		
W15	15.06 Thu.	YouTube Live	M15: In-Memory HTAP Databases with HW/SW Co-design (PDF) (PPT)		
W16	23.06 Thu.	YouTube Live	M16: In-Storage Processing for Genome Analysis (PDF) (PPT)		
W17	18.07 Mon.	YouTube Premiere	M17: How to Enable the Adoption of PIM? (PDF) (PPT)		
W18	09.08 Tue.	YouTube Premiere	SS1: ISVLSI 2022 Special Session on PIM (PDF & PPT)		

An Introduction to Processing-in-Memory

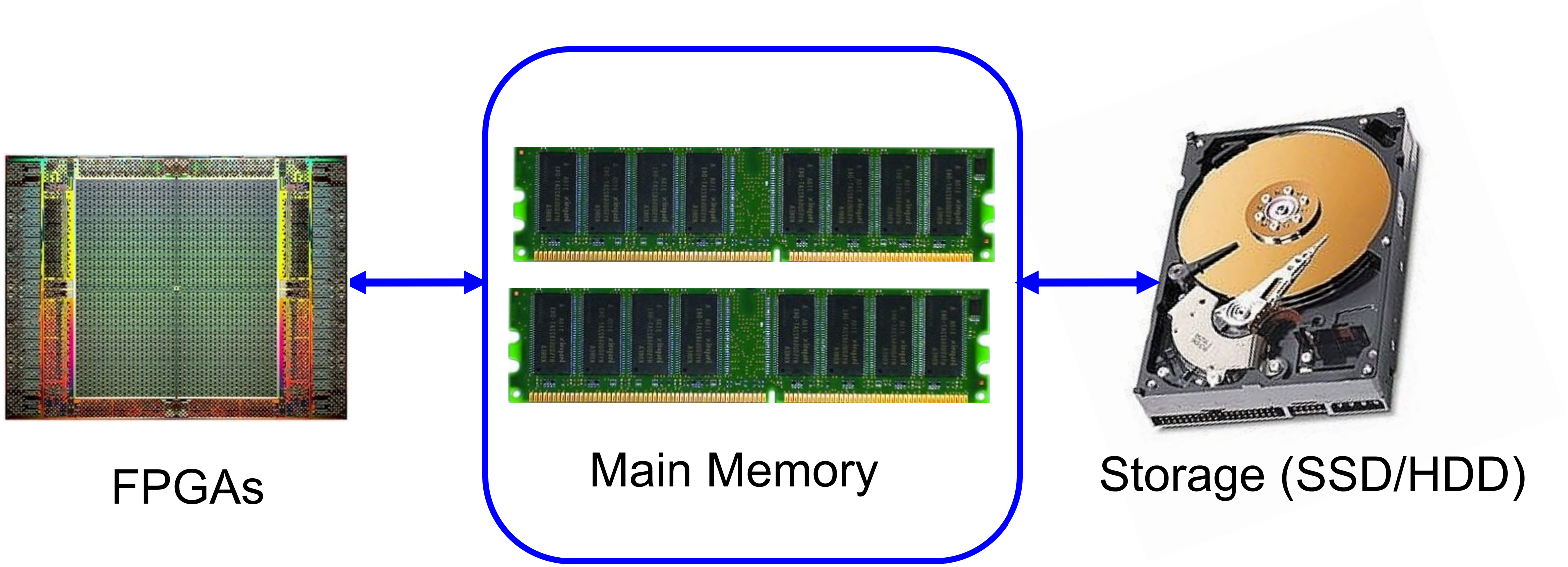
The Main Memory System



- Main memory is a critical component of all computing systems: server, mobile, embedded, desktop, sensor

- Main memory system must scale (in *size*, *technology*, *efficiency*, *cost*, and *management algorithms*) to maintain performance growth and technology scaling benefits

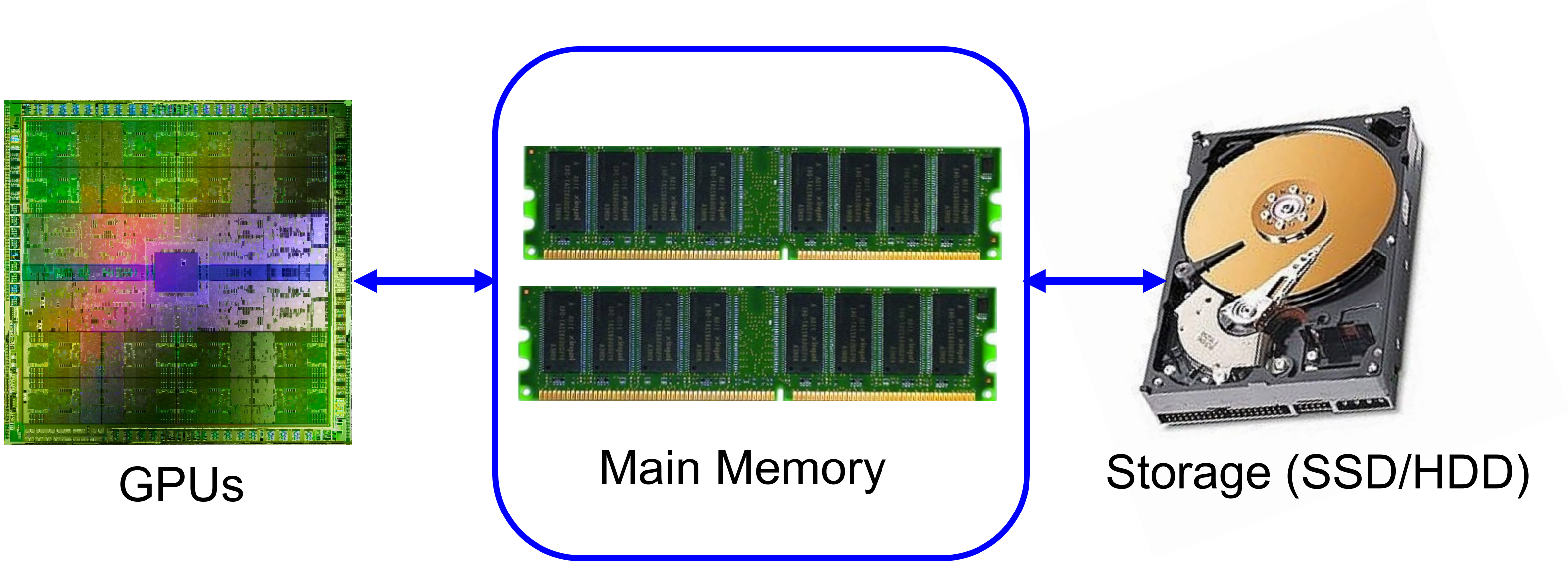
The Main Memory System



- Main memory is a critical component of all computing systems: server, mobile, embedded, desktop, sensor

- Main memory system must scale (in *size*, *technology*, *efficiency*, *cost*, and *management algorithms*) to maintain performance growth and technology scaling benefits

The Main Memory System



GPUs

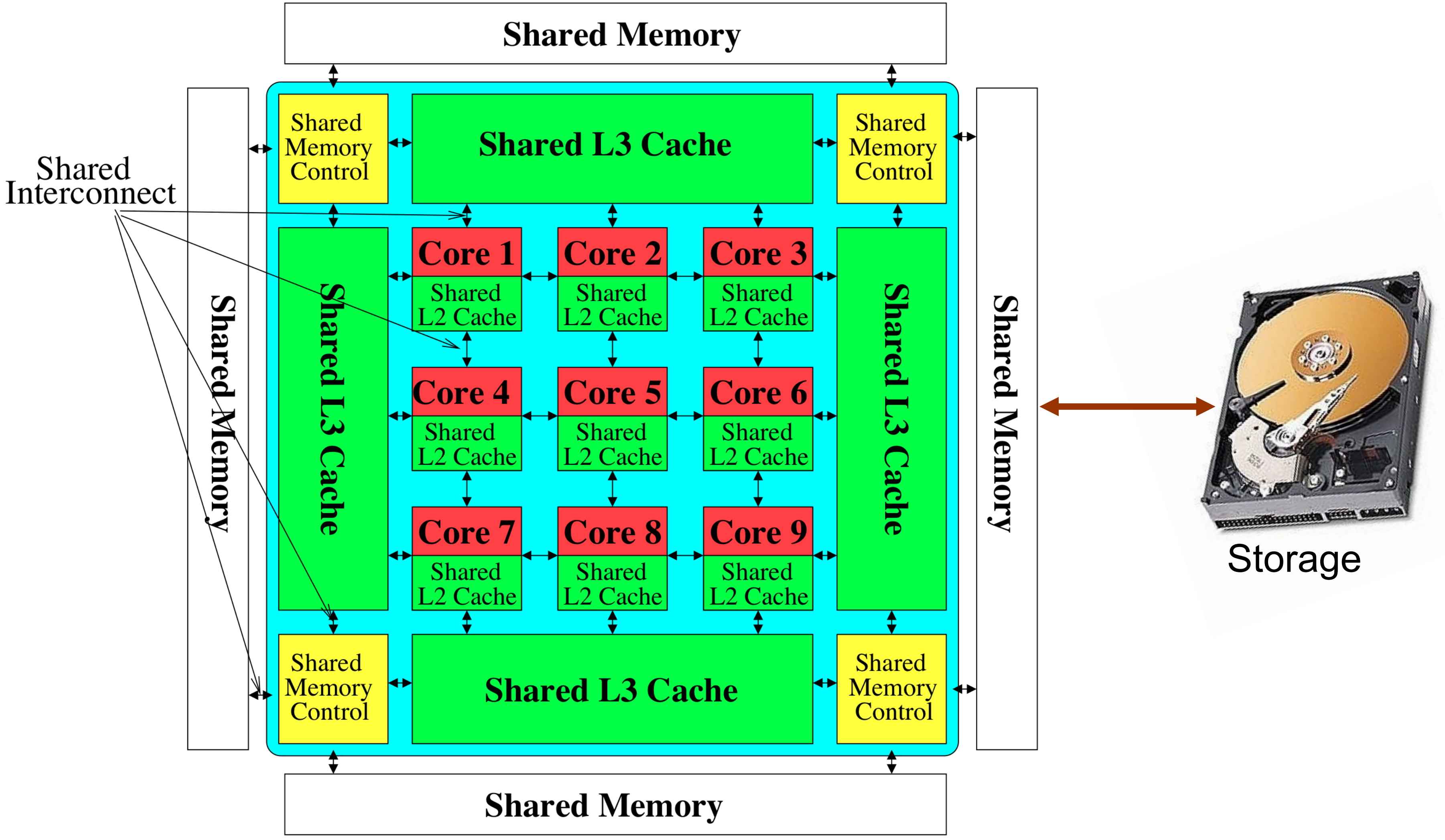
Main Memory

Storage (SSD/HDD)

- Main memory is a critical component of all computing systems: server, mobile, embedded, desktop, sensor

- Main memory system must scale (in *size*, *technology*, *efficiency*, *cost*, and *management algorithms*) to maintain performance growth and technology scaling benefits

Memory System: A ***Shared Resource*** View



Most of the system is dedicated to storing and moving data

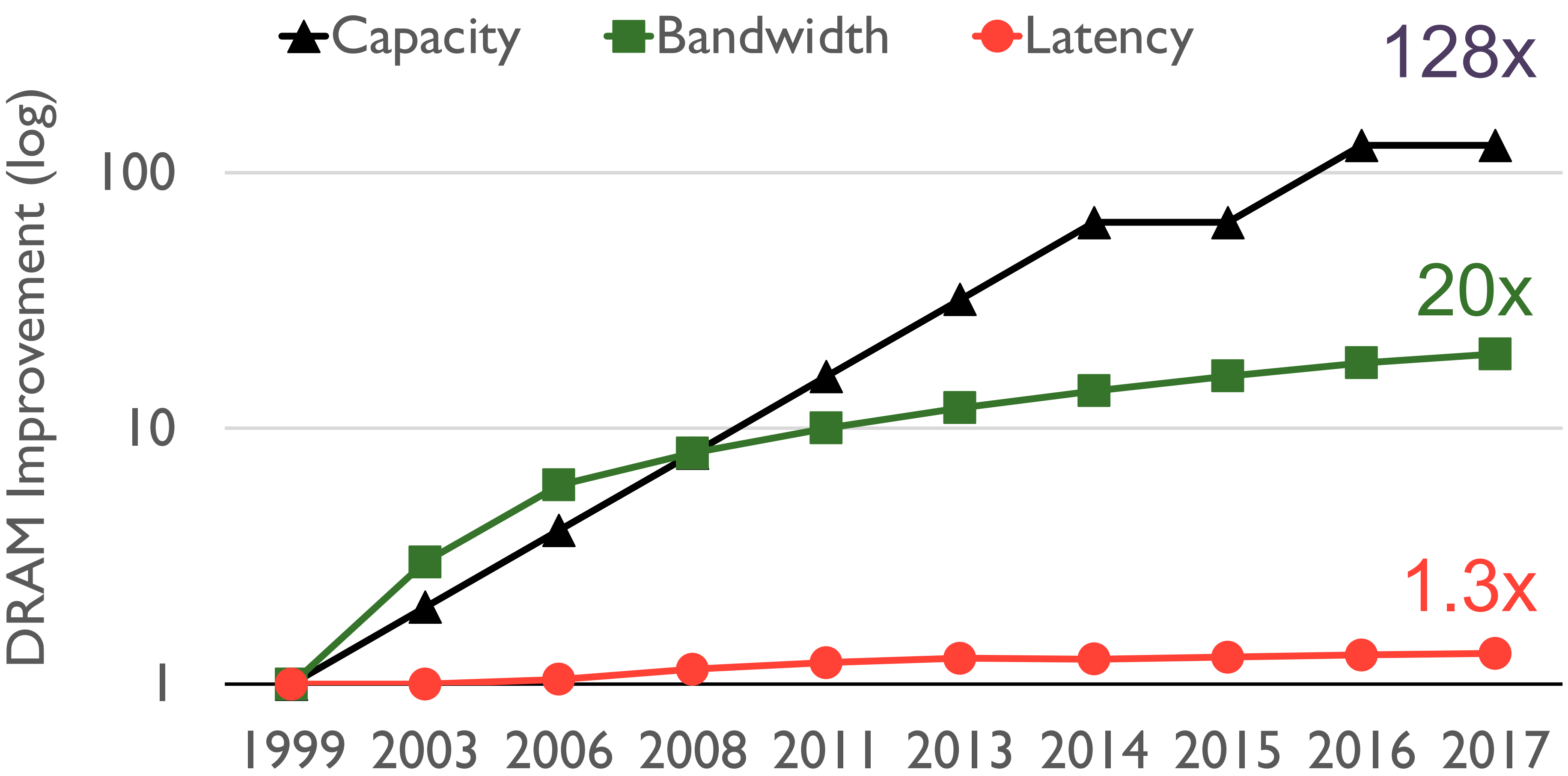
Three Key Systems Trends

1. Data access is a major bottleneck
   - Applications are increasingly data hungry

2. Energy consumption is a key limiter

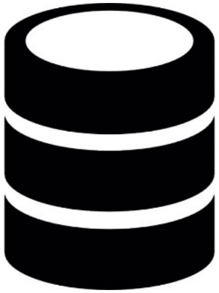
3. Data movement energy dominates compute
   - Especially true for off-chip to on-chip movement

Example: Capacity, Bandwidth & Latency



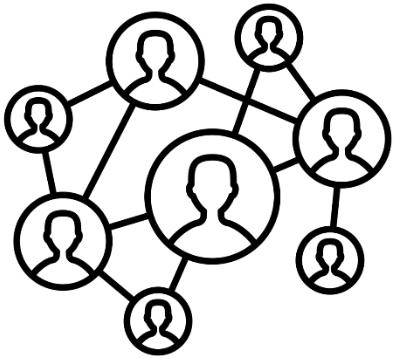
Memory latency remains almost constant

The Need for More Memory Performance



In-memory Databases

[Mao+, EuroSys'12;
Clapp+ (**Intel**), IISWC'15]

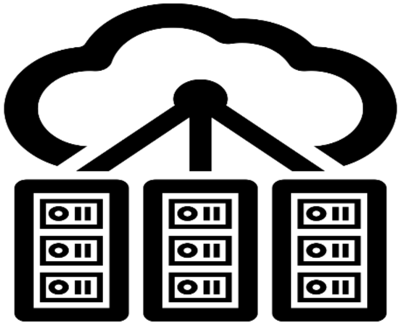


Graph/Tree Processing

[Xu+, IISWC'12; Umuroglu+, FPL'15]

In-Memory Data Analytics

[Clapp+ (**Intel**), IISWC'15;
Awan+, BDCloud'15]



Datacenter Workloads

[Kanev+ (**Google**), ISCA'15]

DRAM Latency Is Critical for Performance

In-memory Databases

Graph/Tree Processing

Long memory latency → performance bottleneck

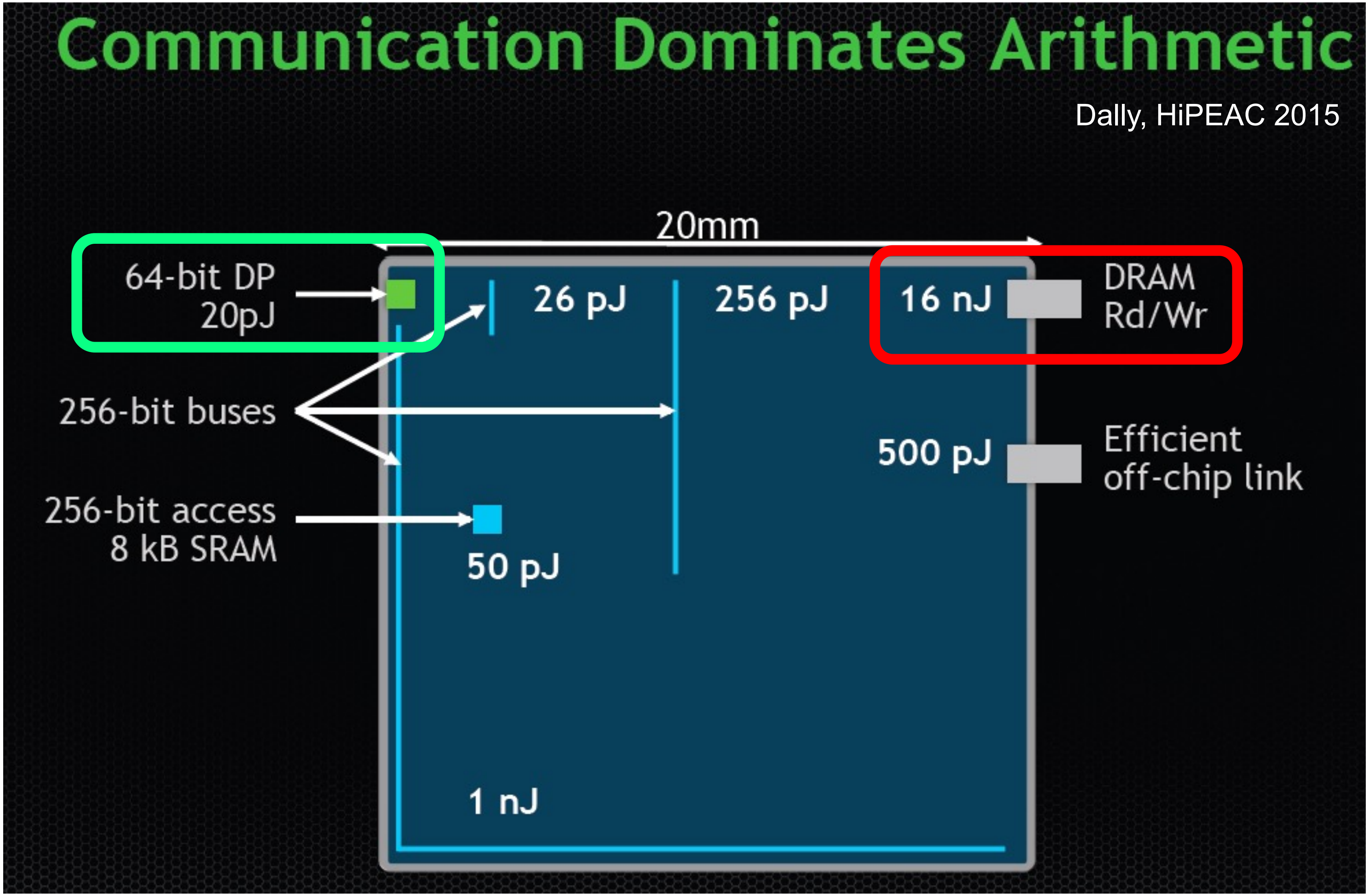
In-Memory Data Analytics

[Clapp+ (**Intel**), IISWC'15;
Awan+, BDCloud'15]

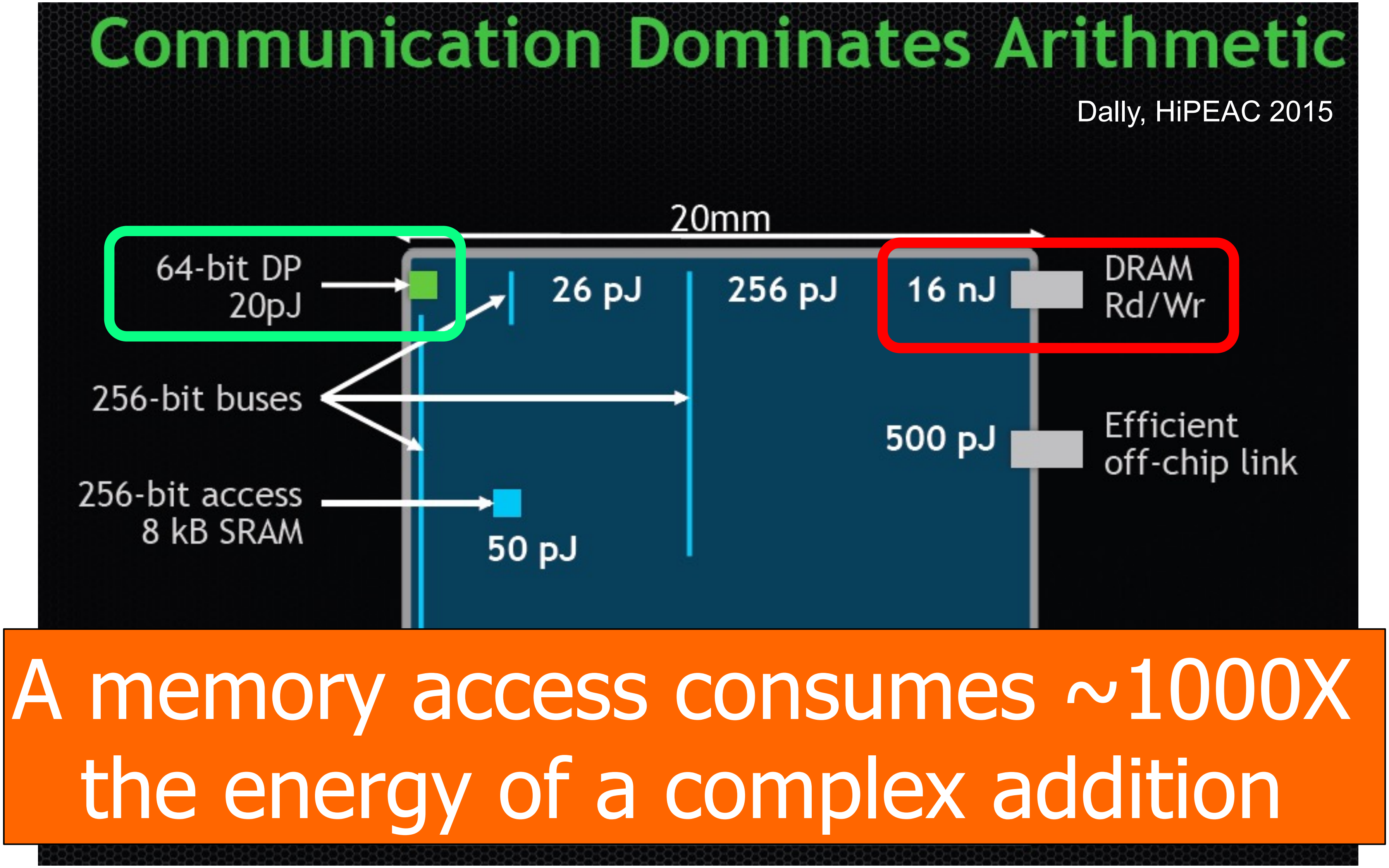
Datacenter Workloads

[Kanev+ (**Google**), ISCA'15]

The Energy Perspective



Data Movement vs. Computation Energy



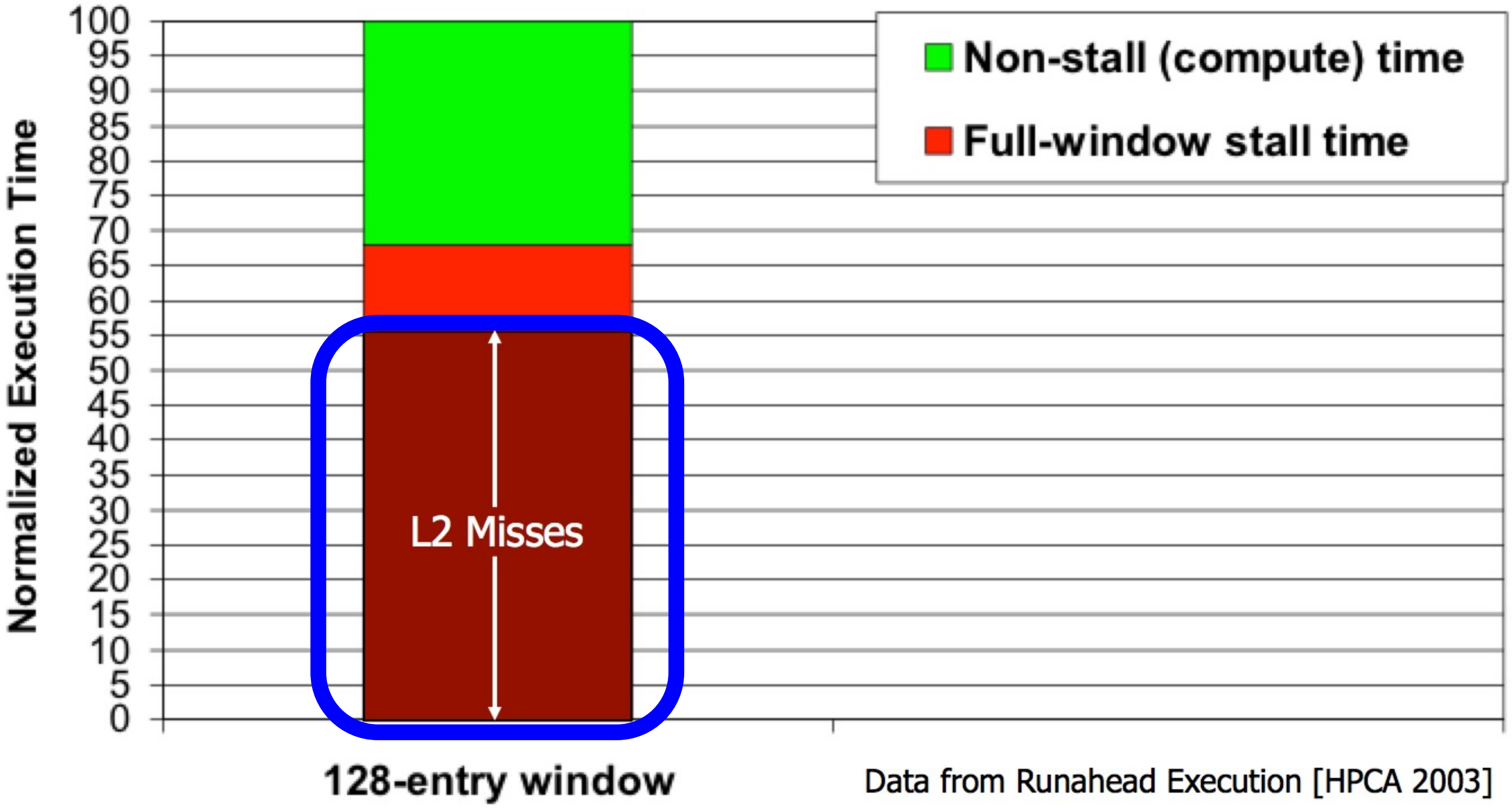
Communication Dominates Arithmetic

Dally, HiPEAC 2015

A memory access consumes ~1000X the energy of a complex addition

The Performance Perspective (1996-2005)

- "**It's the Memory, Stupid!**" (Richard Sites, MPR, 1996)

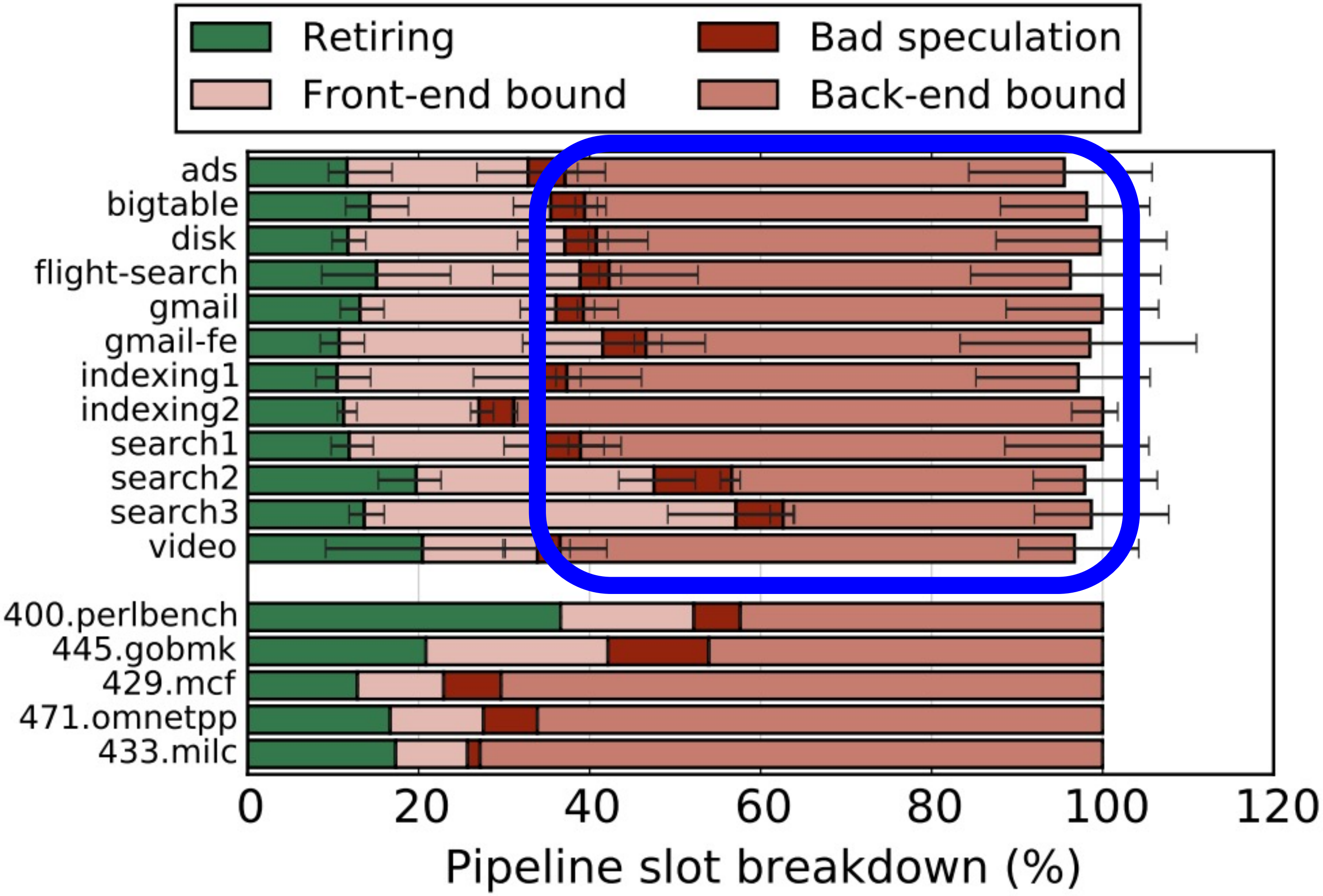


Data from Runahead Execution [HPCA 2003]

Mutlu+, "Runahead Execution: An Alternative to Very Large Instruction Windows for Out-of-Order Processors," HPCA 2003.

The Performance Perspective (Today)

- All of Google's Data Center Workloads (2015):



Kanev+, "Profiling a Warehouse-Scale Computer," ISCA 2015.

The Problem

Data access is the major performance and energy bottleneck

Our current

design principles

cause great energy waste

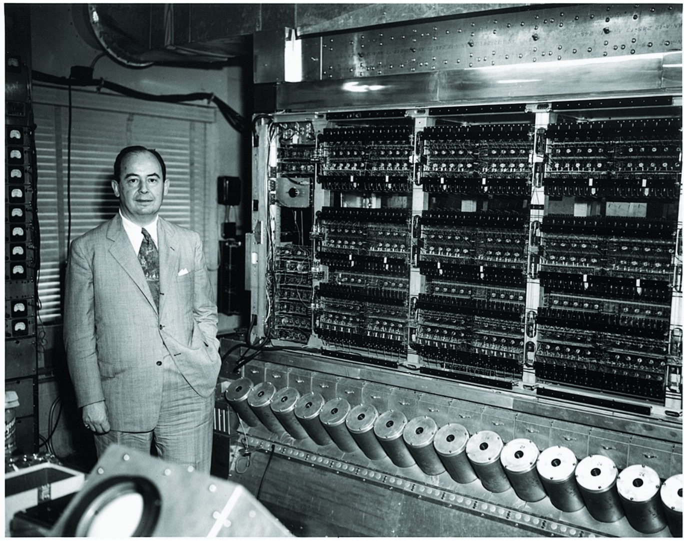
(and great performance loss)

The Problem

Processing of data
is performed
far away from the data

A Computing System

- Three key components
- Computation
- Communication
- Storage/memory



Burks, Goldstein, von Neumann, "Preliminary discussion of the logical design of an electronic computing instrument," 1946.

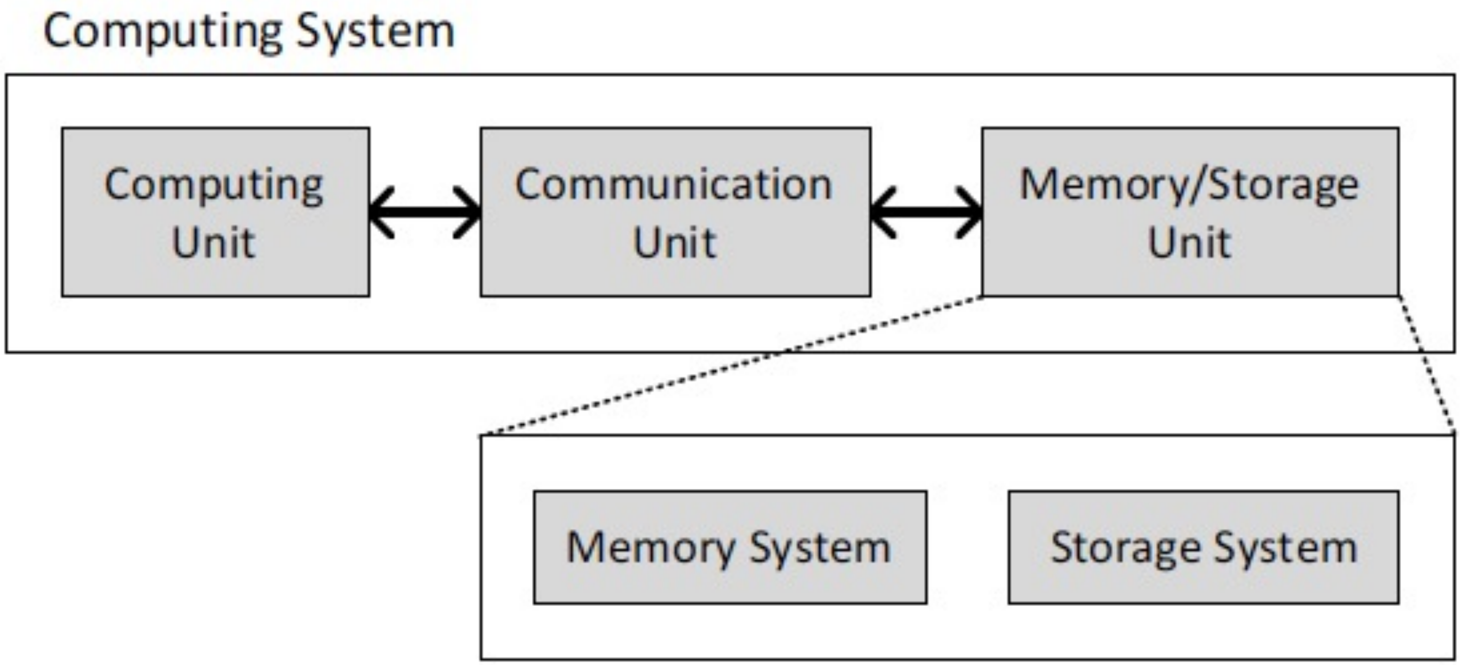
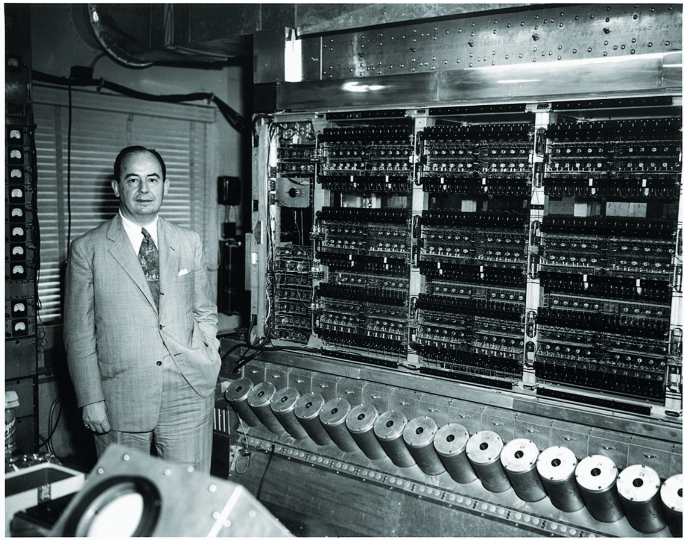


Image source: https://lbsitbytes2010.wordpress.com/2013/03/29/john-von-neumann-roll-no-15/

A Computing System

- Three key components
- Computation
- Communication
- Storage/memory



Burks, Goldstein, von Neumann, “Preliminary discussion of the logical design of an electronic computing instrument,” 1946.

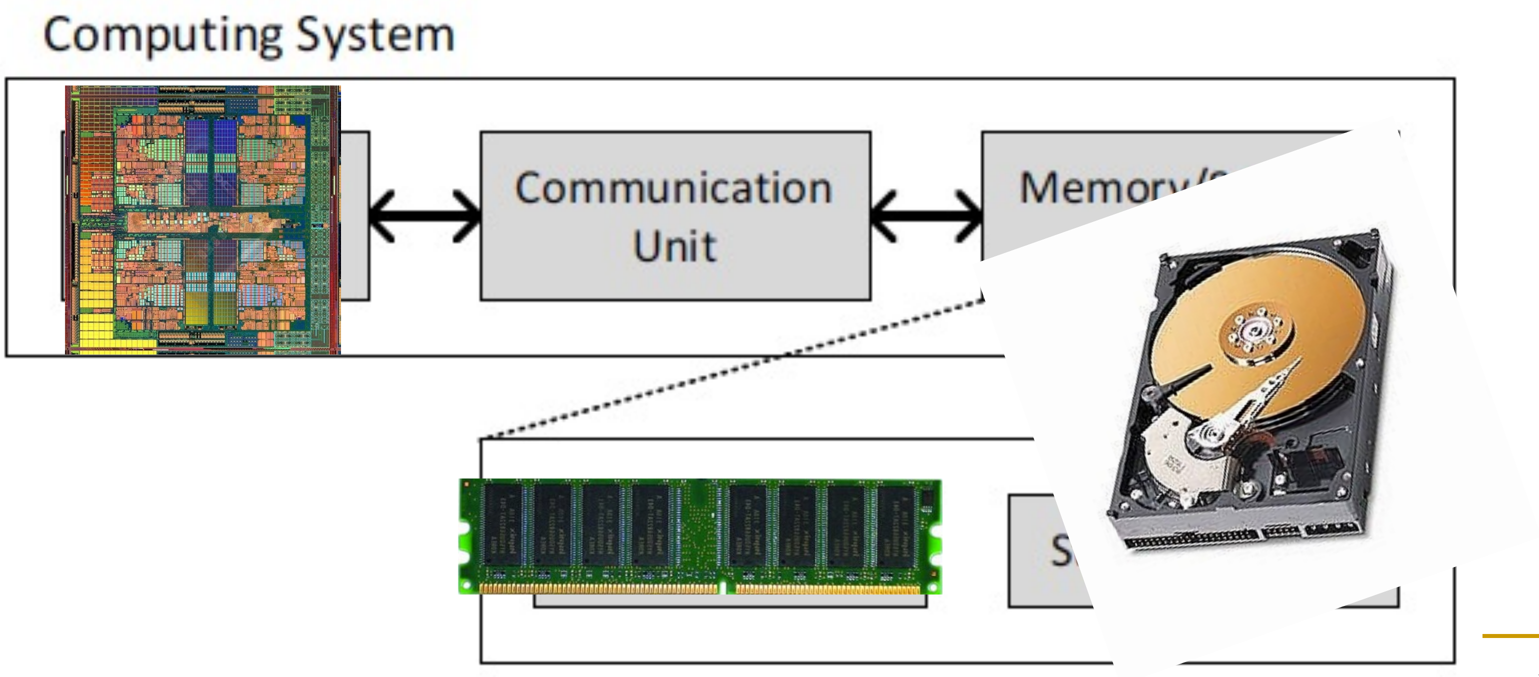
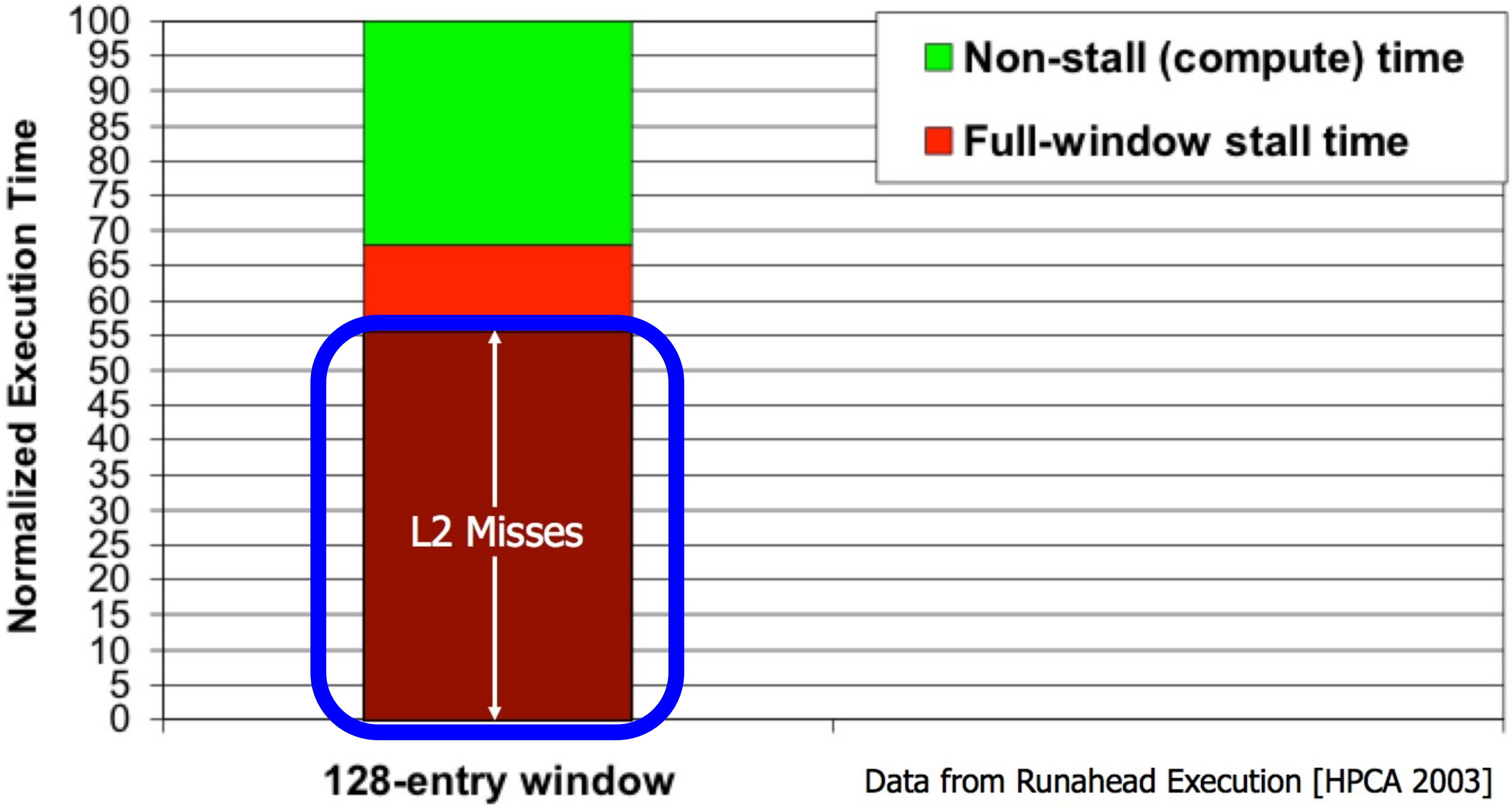


Image source: https://lbsitbytes2010.wordpress.com/2013/03/29/john-von-neumann-roll-no-15/

Yet …

- "**It's the Memory, Stupid!**" (Richard Sites, MPR, 1996)



Data from Runahead Execution [HPCA 2003]

Mutlu+, "Runahead Execution: An Alternative to Very Large Instruction Windows for Out-of-Order Processors," HPCA 2003.

Perils of Processor-Centric Design

- Grossly-imbalanced systems
  - Processing done only in **one place**
  - Everything else just stores and moves data: **data moves a lot**

  → Energy inefficient

  → Low performance

  → Complex

- Overly complex and bloated processor (and accelerators)
  - To tolerate data access from memory
  - Complex hierarchies and mechanisms

  → Energy inefficient

  → Low performance

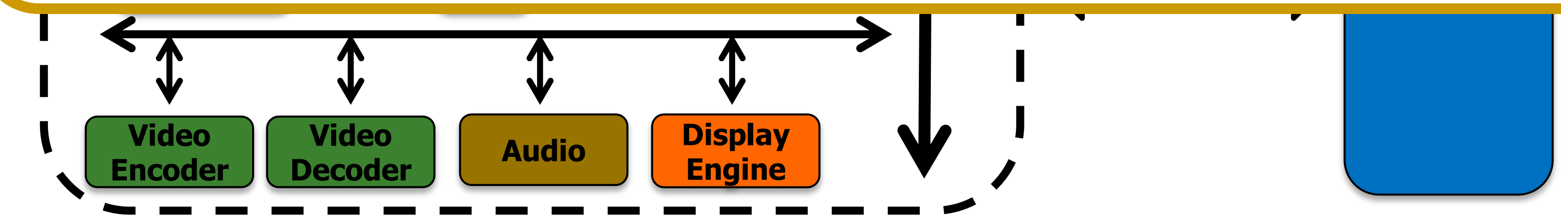
  → Complex

Data Movement in Computing Systems

- Data movement dominates performance and is a major system energy bottleneck
  - Comprises 41% of mobile system energy during web browsing*

Compute systems should be more data-centric

Processing-In-Memory proposes computing where it makes sense (where data resides)



*Reducing data Movement Energy via Online Data Clustering and Encoding (MICRO'16)

**Quantifying the energy cost of data movement for emerging smart phone workloads on mobile platforms (IISWC'14)

Energy Waste in Mobile Devices

- Amirali Boroumand, Saugata Ghose, Youngsok Kim, Rachata Ausavarungnirun, Eric Shiu, Rahul Thakur, Daehyun Kim, Aki Kuusela, Allan Knies, Parthasarathy Ranganathan, and Onur Mutlu, **"Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks"** *Proceedings of the 23rd International Conference on Architectural Support for Programming Languages and Operating Systems* (***ASPLOS***), Williamsburg, VA, USA, March 2018.

62.7% of the total system energy is spent on data movement

Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks

Amirali Boroumand[1] Saugata Ghose[1] Youngsok Kim[2]
Rachata Ausavarungnirun[1] Eric Shiu[3] Rahul Thakur[3] Daehyun Kim[4,3]
Aki Kuusela[3] Allan Knies[3] Parthasarathy Ranganathan[3] Onur Mutlu[5,1]

We Need A Paradigm Shift To …

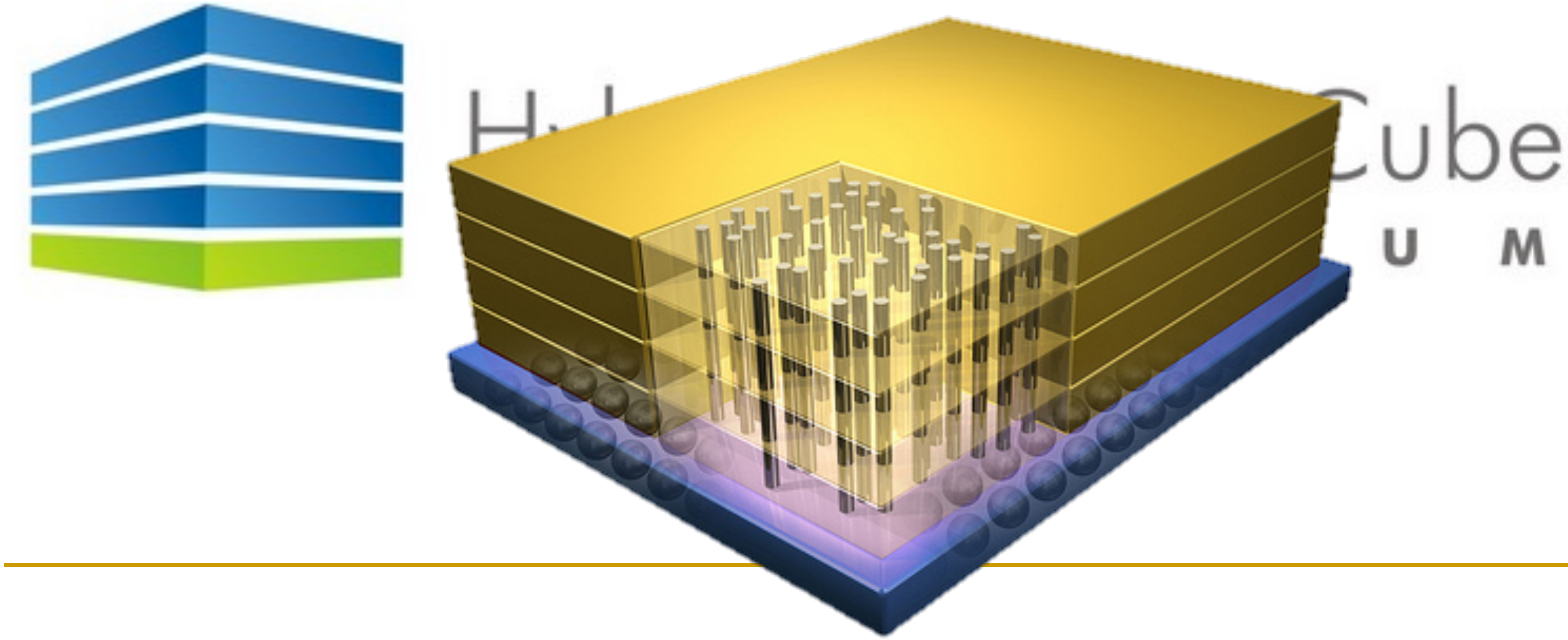
- Enable computation with minimal data movement

- Compute where it makes sense (where data resides)

- Make computing architectures more data-centric

Why In-Memory Computation Today?

- Pull from systems/applications for data-centric execution
- It can be practical today
  - 3D-stacked memories combine logic and memory functionality (relatively) tightly + industry open to new architectures



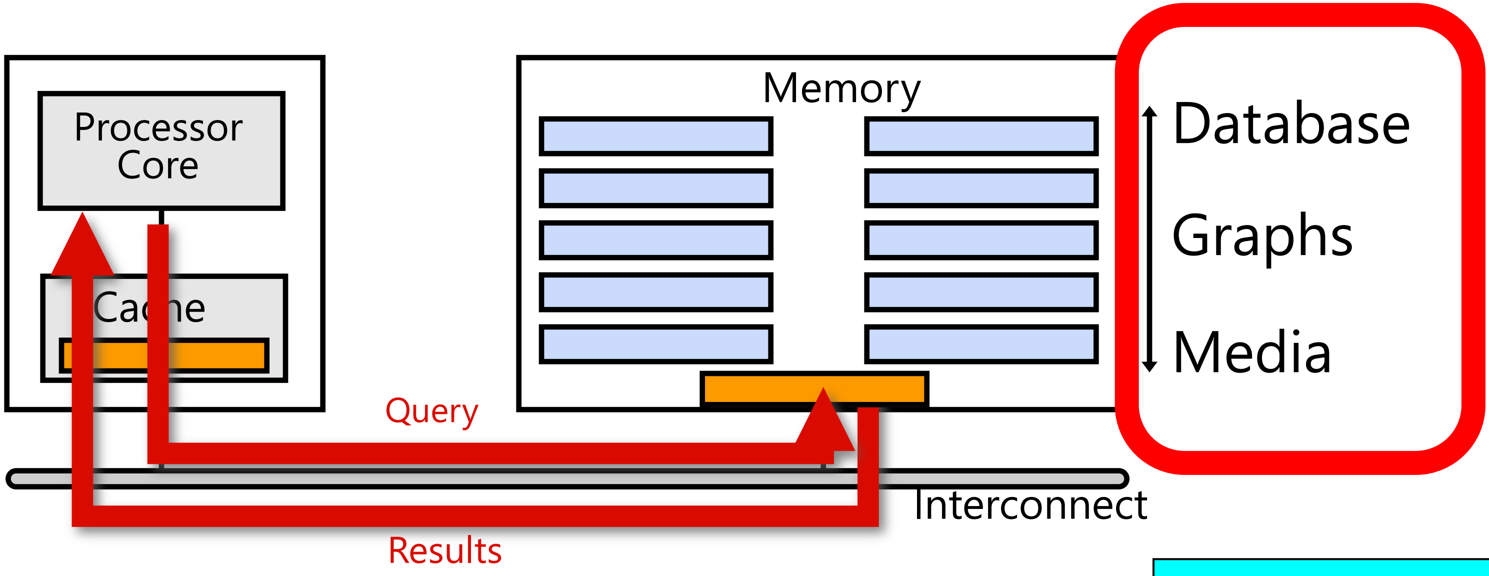
Challenge and Opportunity for Future

High Performance

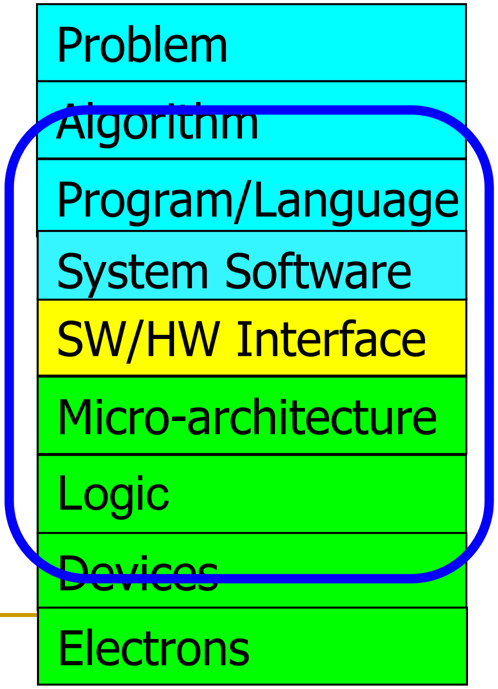
and

Energy Efficiency

Goal: Processing Inside Memory



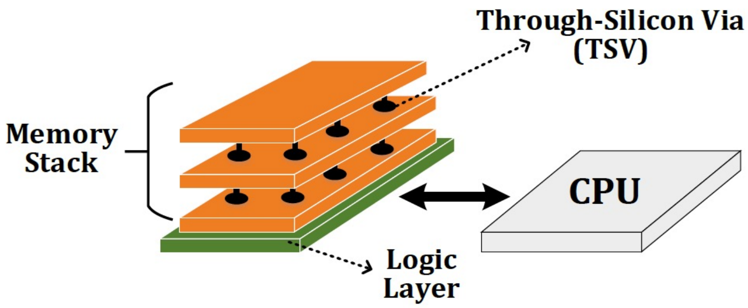
- Many questions… How do we design the:
  - compute-capable memory & controllers?
  - processor chip?
  - software and hardware interfaces?
  - system software and languages?
  - algorithms?



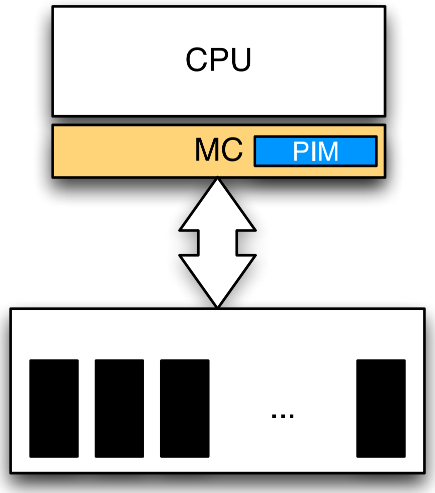
Processing In-Memory (PIM)

- Near-Data Processing or Processing In-Memory (PIM)
  - Move computation closer to where the data resides

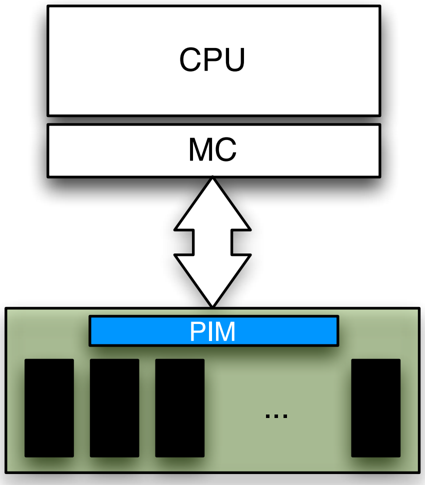
**Logic layer
3D stacked DRAM**



Memory controller

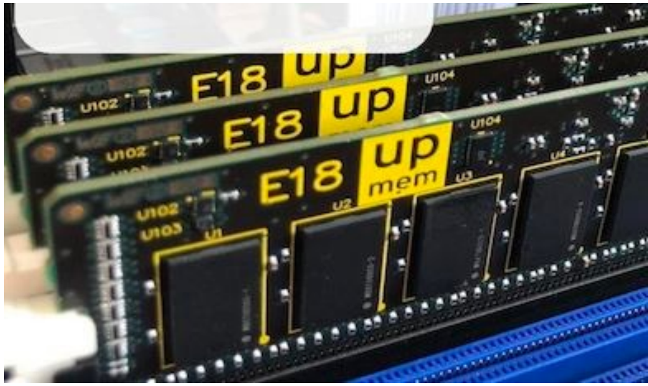


**Memory module
(DIMM)**

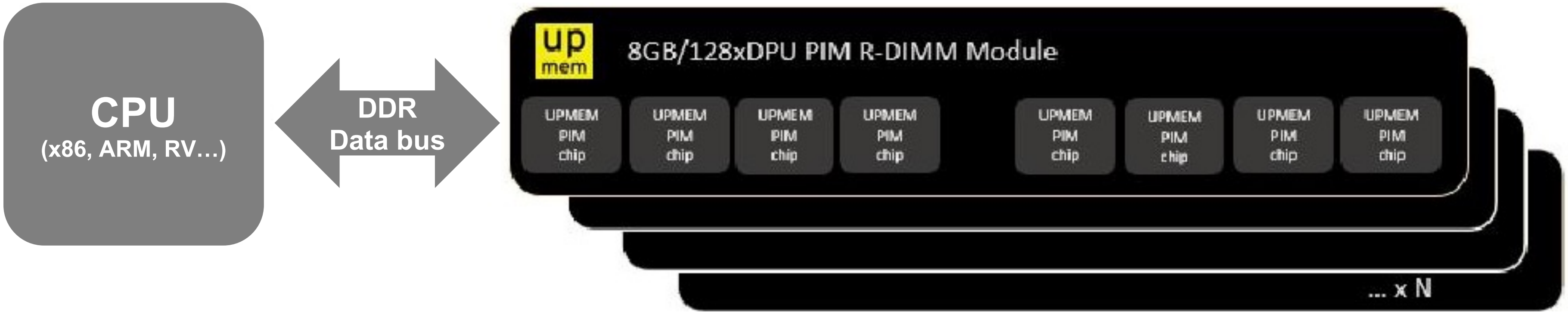


UPMEM Processing-in-DRAM Engine (2019)

- 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

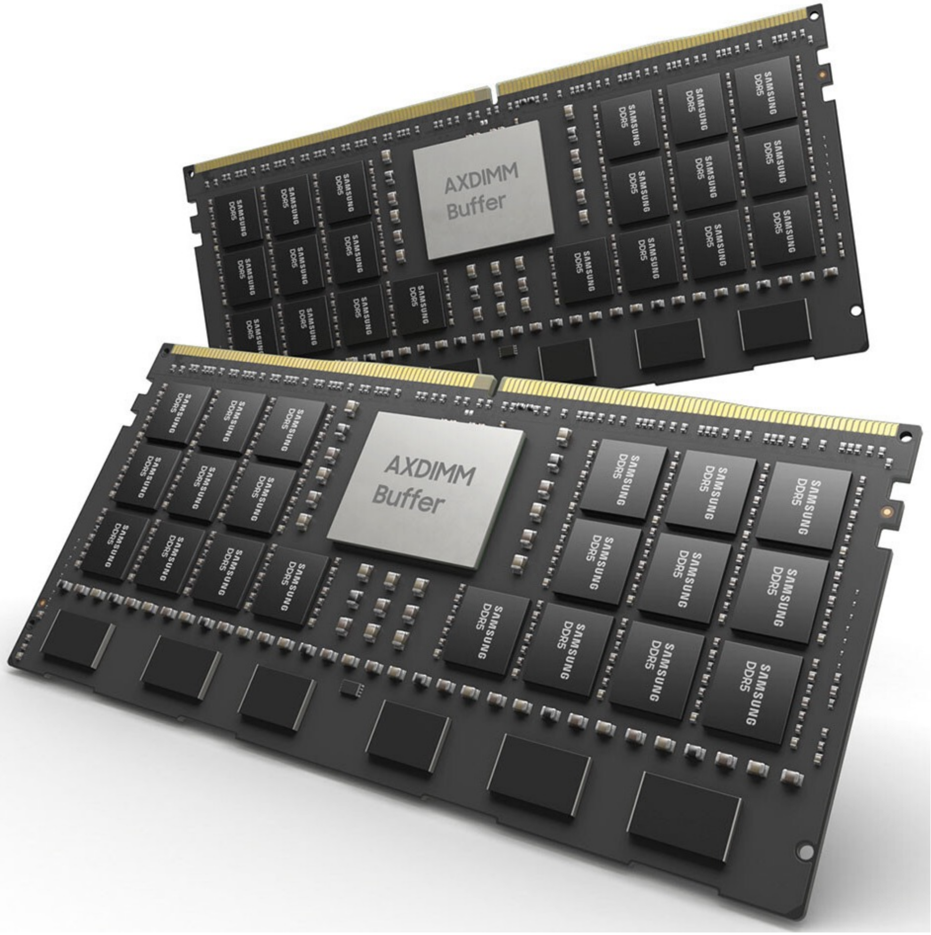
UPMEM PIM chip

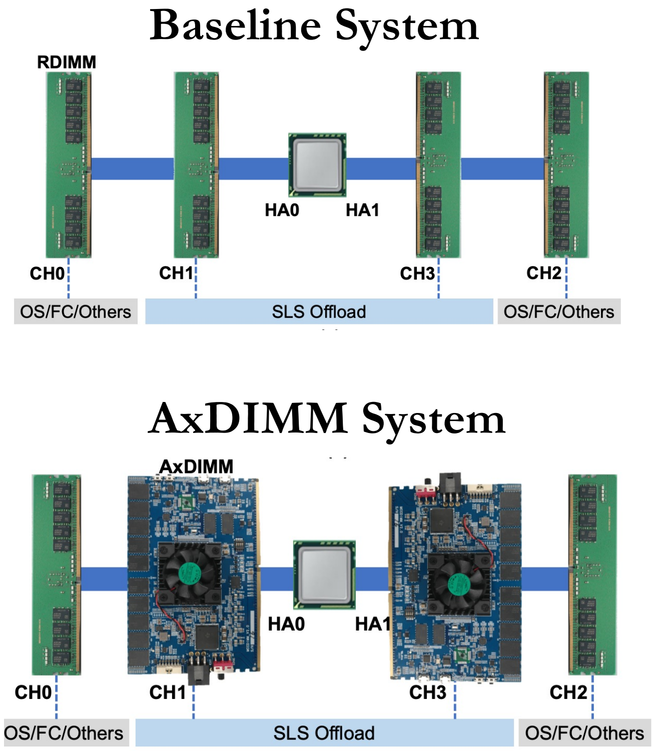
... x N

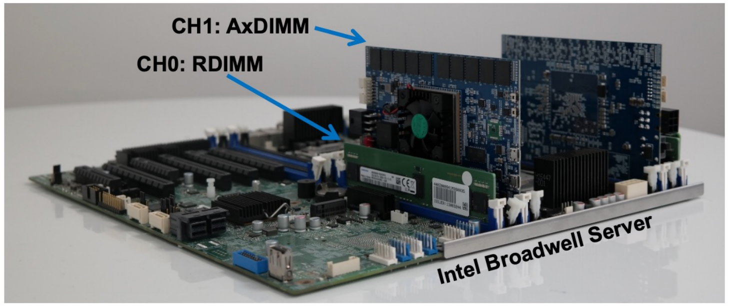
https://www.anandtech.com/show/14750/hot-chips-31-analysis-inmemory-processing-by-upmem
https://www.upmem.com/video-upmem-presenting-its-true-processing-in-memory-solution-hot-chips-2019/

Samsung AxDIMM (2021)

- DIMM-based PIM
  - DLRM recommendation system



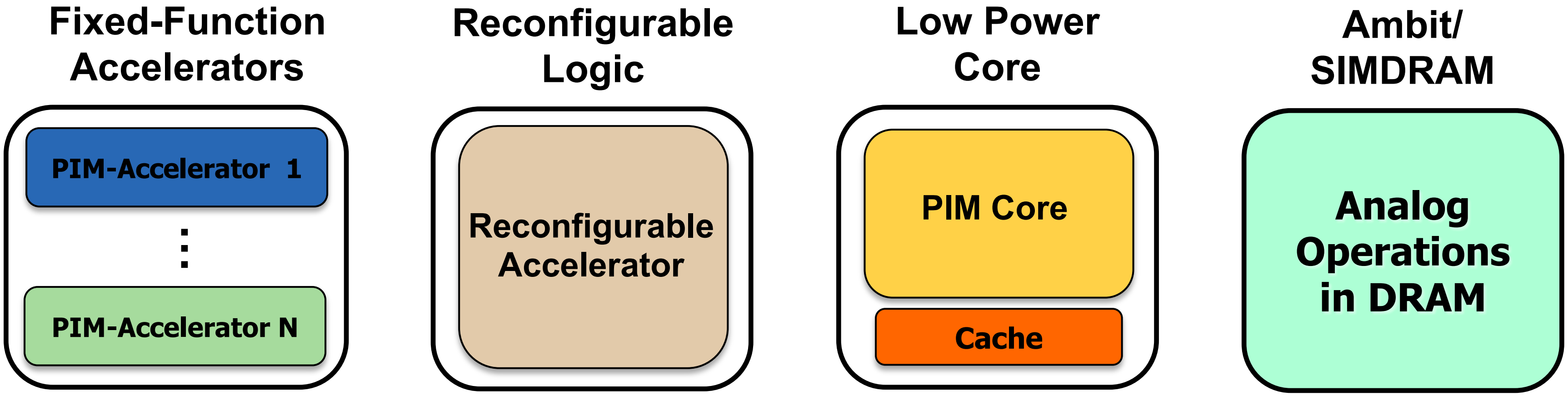




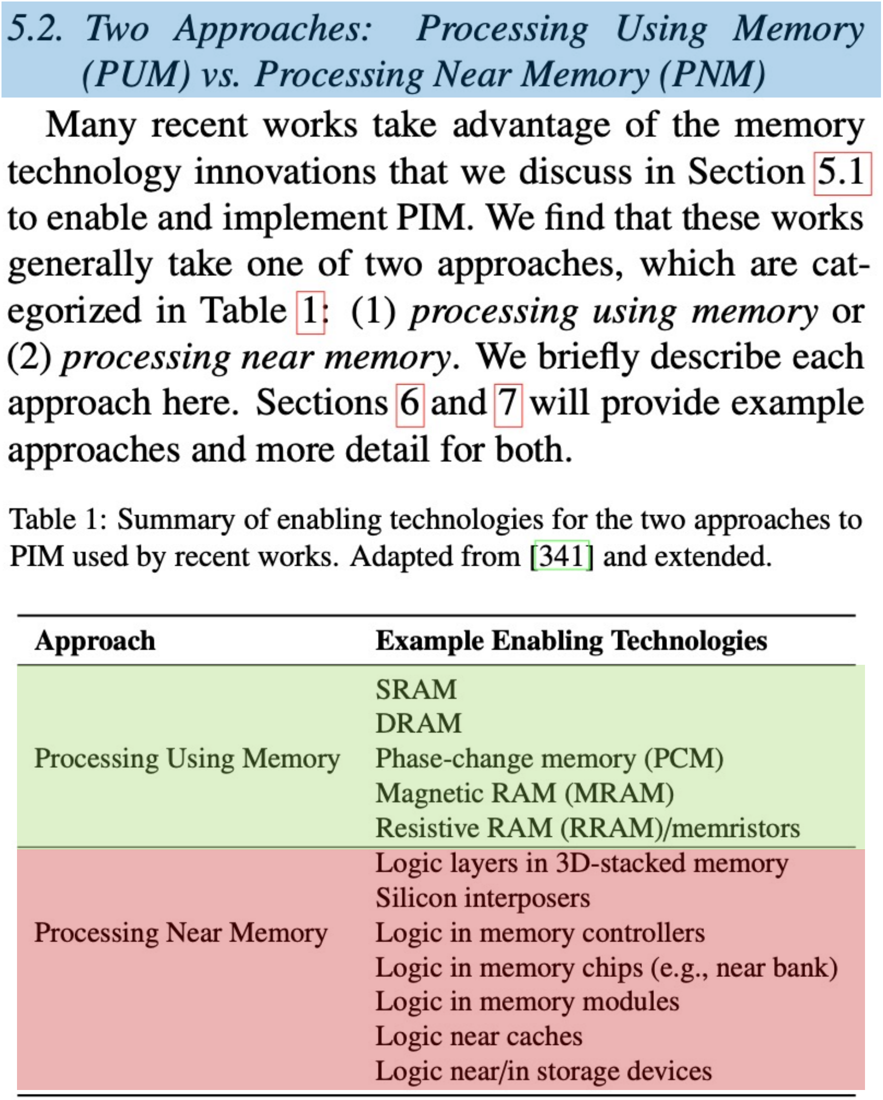
Ke et al. "Near-Memory Processing in Action: Accelerating Personalized Recommendation with AxDIMM", IEEE Micro (2021)

Possible Designs

- Fixed-function units
- Reconfigurable architectures
  - FPGAs, CGRA
- General-purpose programmable cores
  - E.g., ARM Cortex R-8, ARM Cortex A-35 (+SIMD units)
  - Possibility of running any workload
- Processing-using-memory:
  - Ambit: In-DRAM bulk bitwise operations (Seshadri+, MICRO'17)
  - SIMDRAM: End-to-end framework for SIMD in DRAM (Hajinazar+, ASPLOS'21)



Two PIM Approaches



5.2. *Two Approaches: Processing Using Memory (PUM) vs. Processing Near Memory (PNM)*

Many recent works take advantage of the memory technology innovations that we discuss in Section 5.1 to enable and implement PIM. We find that these works generally take one of two approaches, which are categorized in Table 1: (1) *processing using memory* or (2) *processing near memory*. We briefly describe each approach here. Sections 6 and 7 will provide example approaches and more detail for both.

Table 1: Summary of enabling technologies for the two approaches to PIM used by recent works. Adapted from [341] and extended.

Approach	**Example Enabling Technologies**
Processing Using Memory	SRAM DRAM Phase-change memory (PCM) Magnetic RAM (MRAM) Resistive RAM (RRAM)/memristors
Processing Near Memory	Logic layers in 3D-stacked memory Silicon interposers Logic in memory controllers Logic in memory chips (e.g., near bank) Logic in memory modules Logic near caches Logic near/in storage devices

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun,
"A Modern Primer on Processing in Memory"
Invited Book Chapter in ***Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann***, Springer, to be published in 2021.
[Tutorial Video on "Memory-Centric Computing Systems" (1 hour 51 minutes)]

https://people.inf.ethz.ch/omutlu/pub/ModernPrimerOnPIM_springer-emerging-computing-bookchapter21.pdf

Processing in Memory: Two Approaches

1. Processing using Memory
2. Processing near Memory

Agenda

- Major Trends Affecting Memory

- Processing in Memory: Two Directions
  - Processing-using-Memory (PuM)
    - Minimally Changing Memory Chips

  - Processing-near-Memory (PnM)
    - Exploiting 3D-Stacked Memory

Approach 1: Minimally Changing DRAM

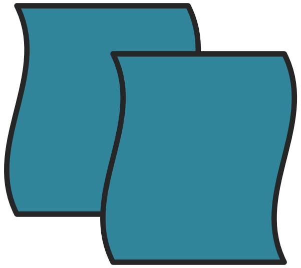
- DRAM has great capability to perform bulk data movement and computation internally with small changes
  - Can exploit internal bandwidth to move data
  - Can exploit analog computation capability
  - ...

- Examples: RowClone, In-DRAM AND/OR, Gather/Scatter DRAM
  - RowClone: Fast and Efficient In-DRAM Copy and Initialization of Bulk Data (Seshadri et al., MICRO 2013)
  - Fast Bulk Bitwise AND and OR in DRAM (Seshadri et al., IEEE CAL 2015)
  - Gather-Scatter DRAM: In-DRAM Address Translation to Improve the Spatial Locality of Non-unit Strided Accesses (Seshadri et al., MICRO 2015)
  - "Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology" (Seshadri et al., MICRO 2017)
  - "SIMDRAM: An End-to-End Framework for Bit-Serial SIMD Computing in DRAM" (Hajinazar et al., ASPLOS 2021)

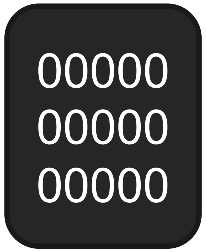
RowClone:
In-Memory Copy and Initialization

Starting Simple: Data Copy and Initialization

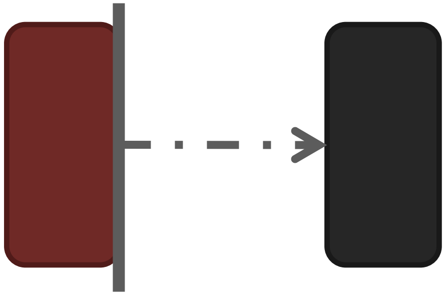
memmove & *memcpy*: 5% cycles in Google's datacenter [Kanev+ ISCA'15]



Forking

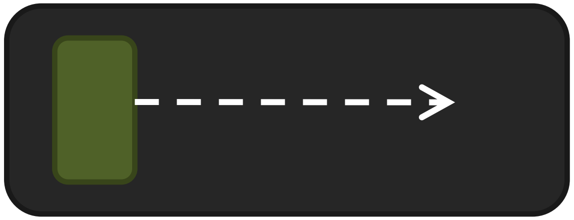


Zero initialization
(e.g., security)



Checkpointing

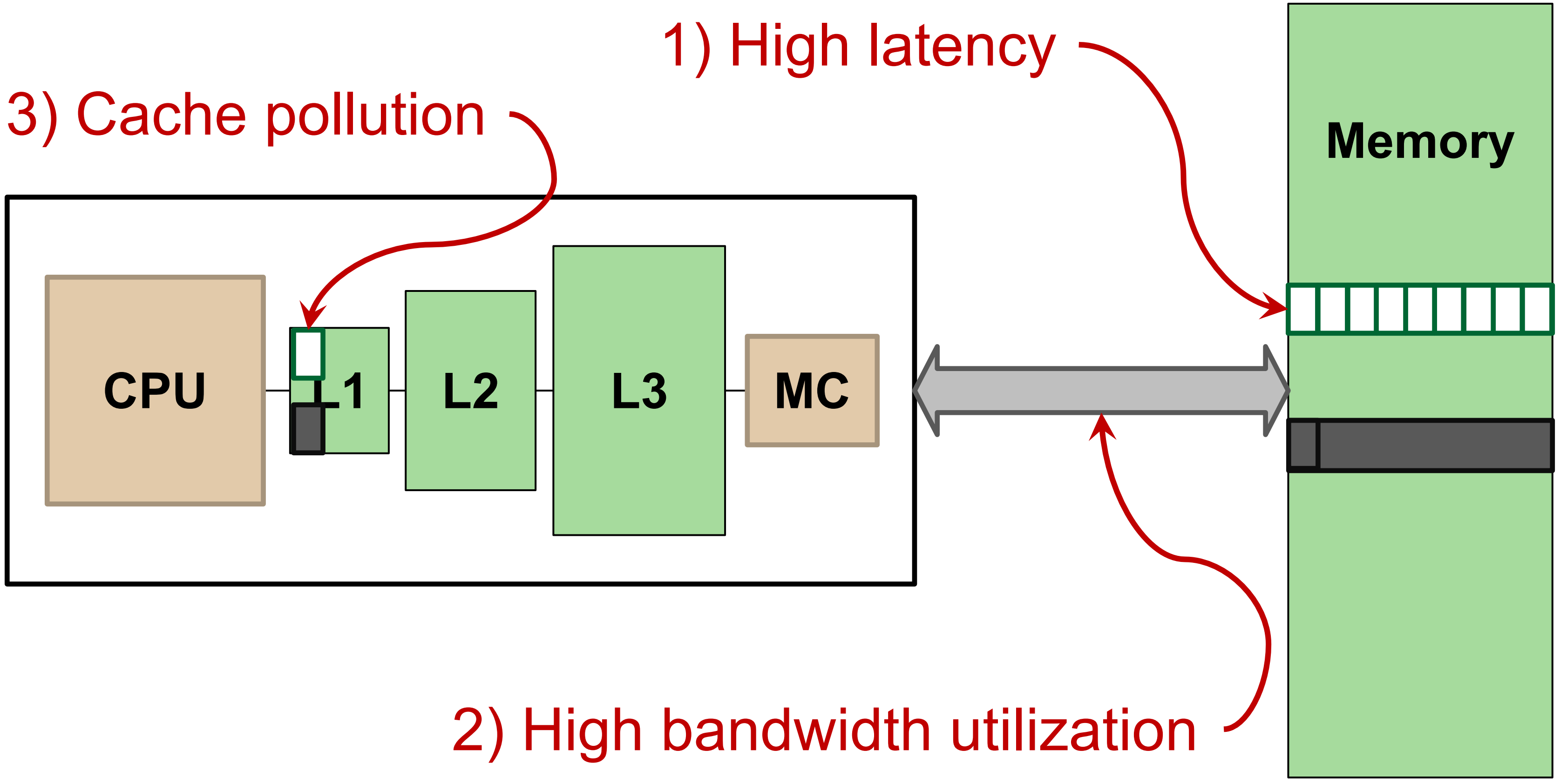
VM Cloning
Deduplication



Page Migration

• • •
Many more

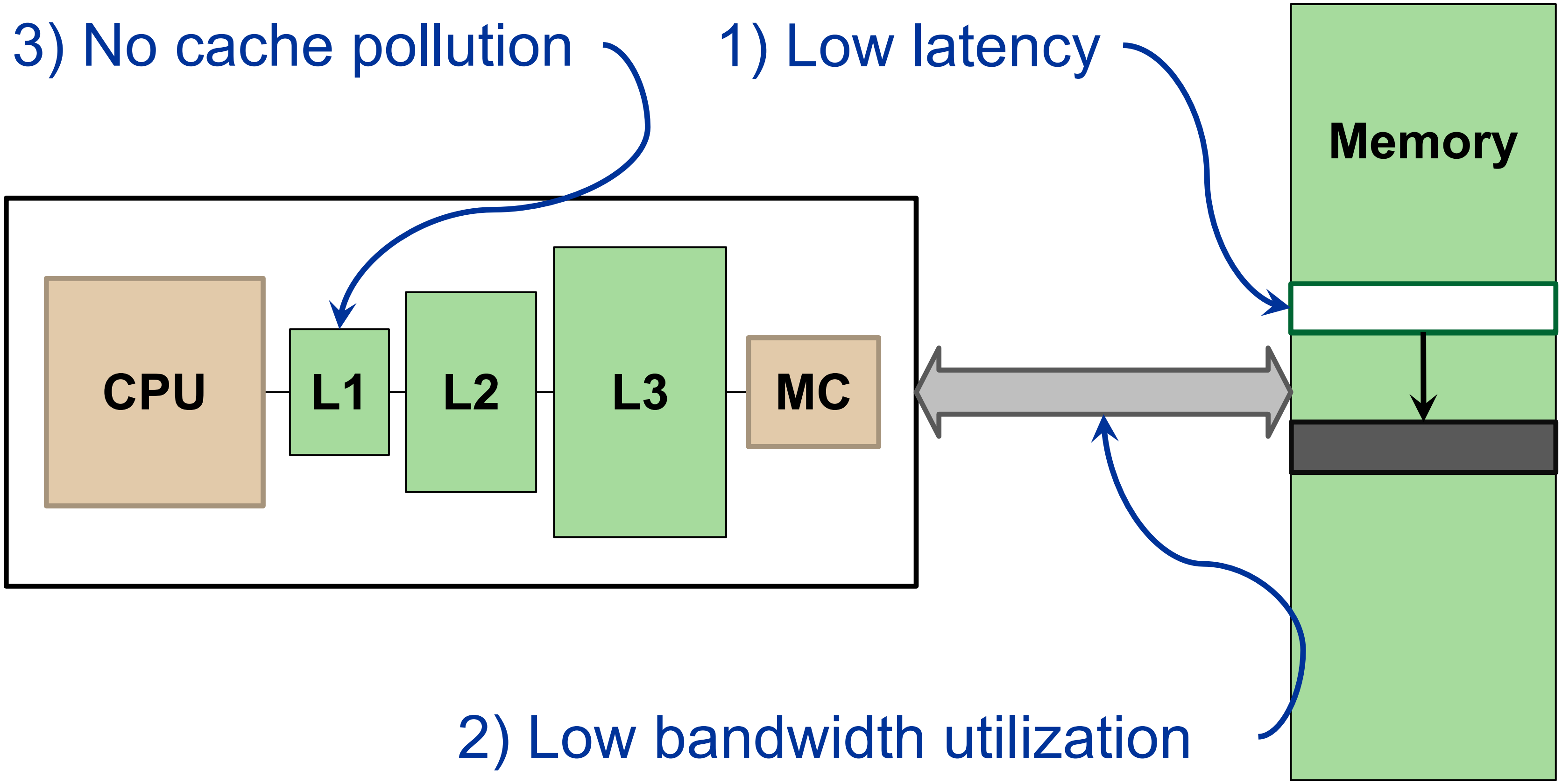
Today's Systems: Bulk Data Copy



4) Unwanted data movement

1046ns, 3.6uJ (for 4KB page copy via DMA)

Future Systems: In-Memory Copy



3) No cache pollution

1) Low latency

2) Low bandwidth utilization

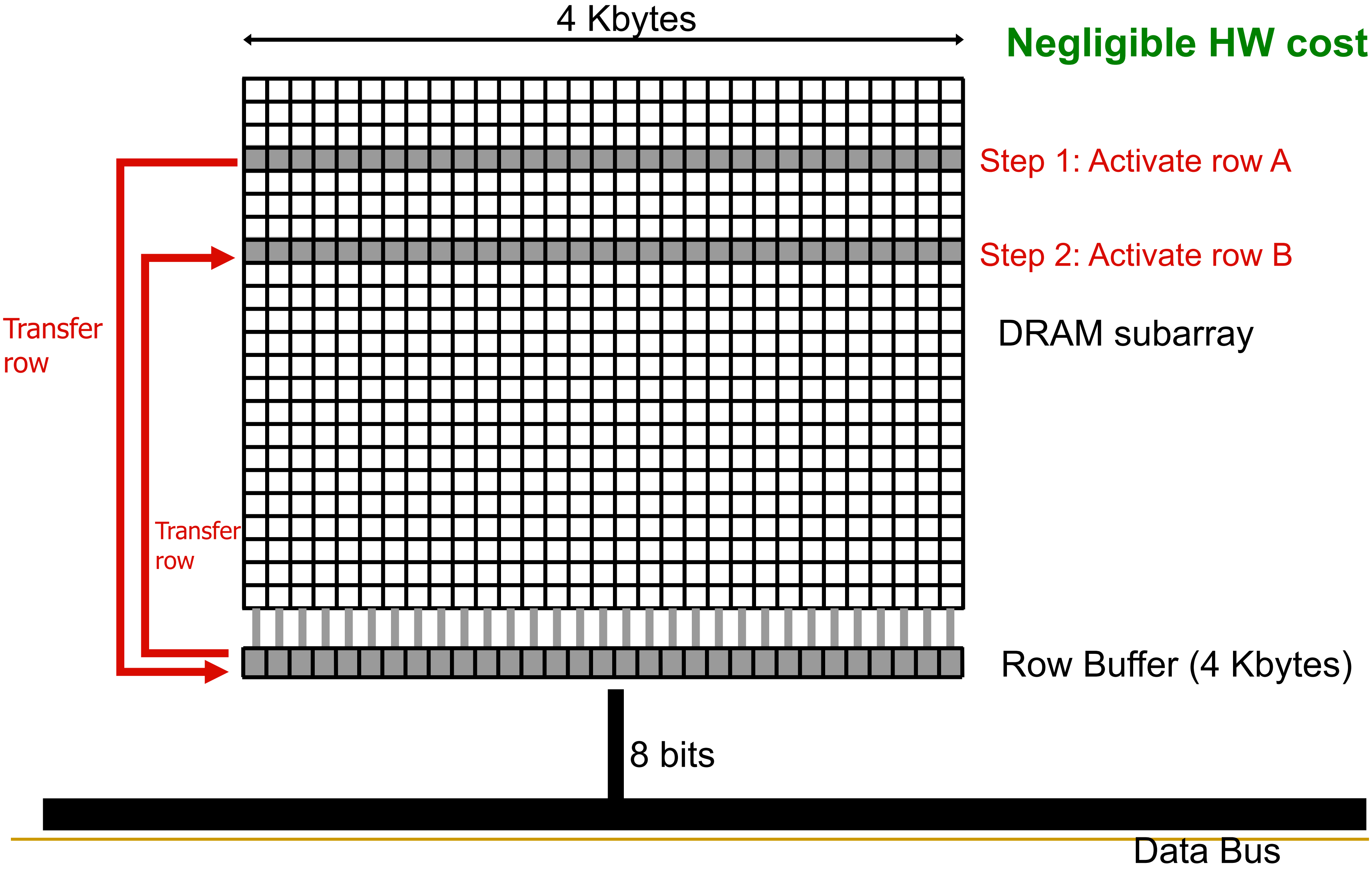
4) No unwanted data movement

1046ns, 3.6uJ → 90ns, 0.04uJ

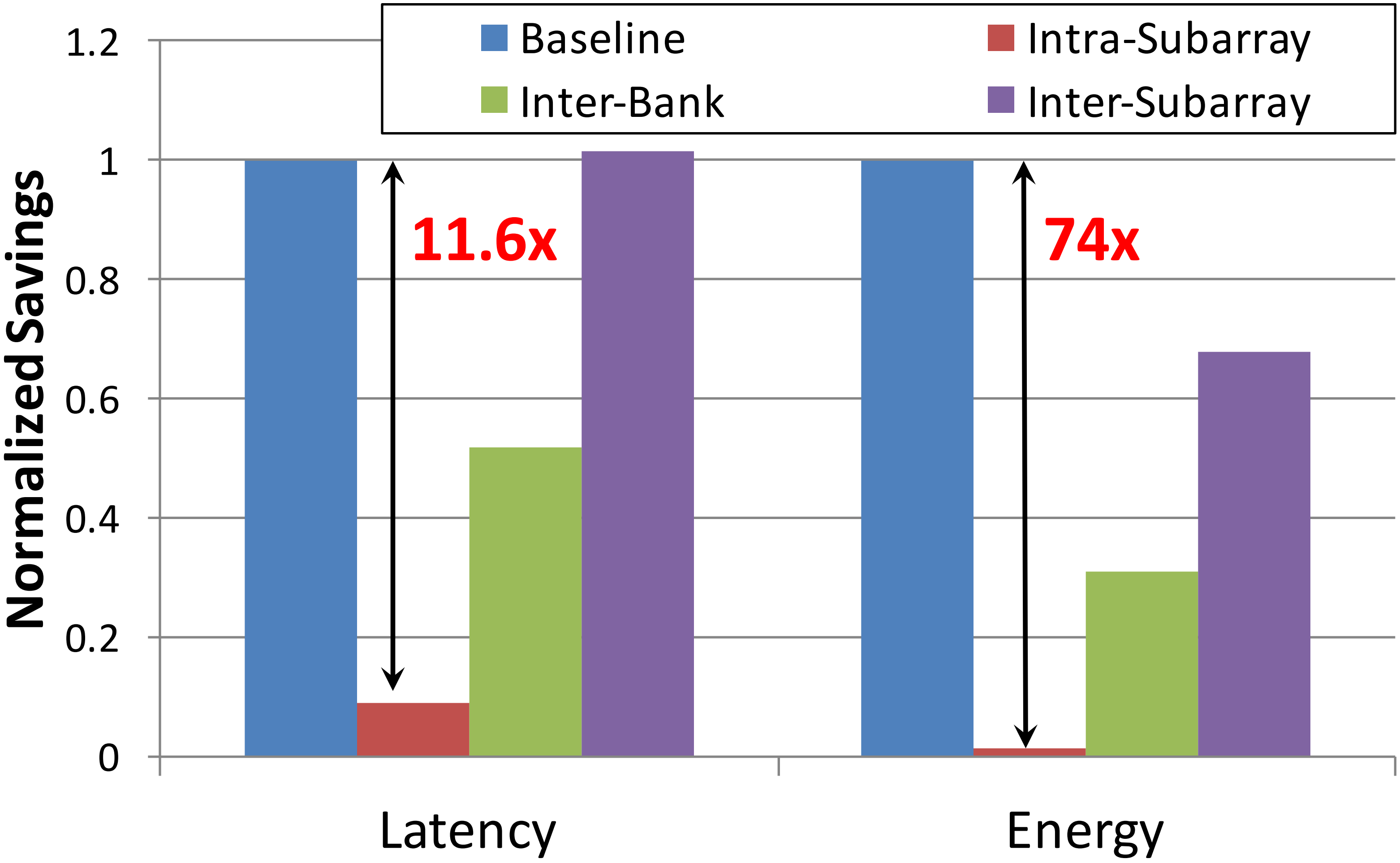
RowClone: In-DRAM Row Copy

Idea: Two consecutive ACTivates

Negligible HW cost



RowClone: Latency and Energy Savings



Seshadri et al., "RowClone: Fast and Efficient In-DRAM Copy and Initialization of Bulk Data," MICRO 2013.

More on RowClone

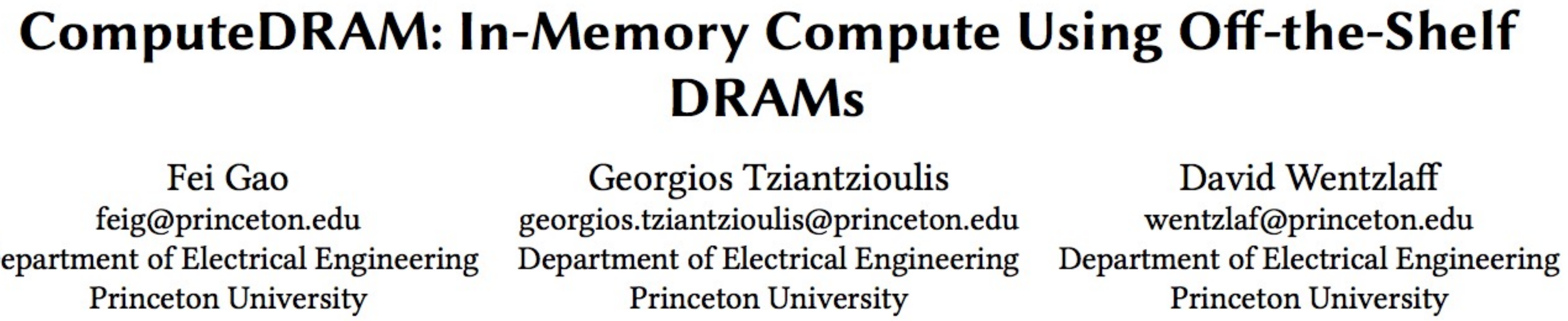
- Vivek Seshadri, Yoongu Kim, Chris Fallin, Donghyuk Lee, Rachata Ausavarungnirun, Gennady Pekhimenko, Yixin Luo, Onur Mutlu, Michael A. Kozuch, Phillip B. Gibbons, and Todd C. Mowry, **"RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization"** *Proceedings of the 46th International Symposium on Microarchitecture (**MICRO**)*, Davis, CA, December 2013. [Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)]

RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization

Vivek Seshadri vseshadr@cs.cmu.edu
Yoongu Kim yoongukim@cmu.edu
Chris Fallin* cfallin@c1f.net
Donghyuk Lee donghyuk1@cmu.edu
Rachata Ausavarungnirun rachata@cmu.edu
Gennady Pekhimenko gpekhime@cs.cmu.edu
Yixin Luo yixinluo@andrew.cmu.edu
Onur Mutlu onur@cmu.edu
Phillip B. Gibbons† phillip.b.gibbons@intel.com
Michael A. Kozuch† michael.a.kozuch@intel.com
Todd C. Mowry tcm@cs.cmu.edu

Carnegie Mellon University †Intel Pittsburgh

RowClone Demonstration in Real DRAM Chips



ComputeDRAM: In-Memory Compute Using Off-the-Shelf DRAMs

Fei Gao
feig@princeton.edu
Department of Electrical Engineering
Princeton University

Georgios Tziantzioulis
georgios.tziantzioulis@princeton.edu
Department of Electrical Engineering
Princeton University

David Wentzlaff
wentzlaf@princeton.edu
Department of Electrical Engineering
Princeton University

https://parallel.princeton.edu/papers/micro19-gao.pdf

Ambit: In-Memory Bulk Bitwise Operations

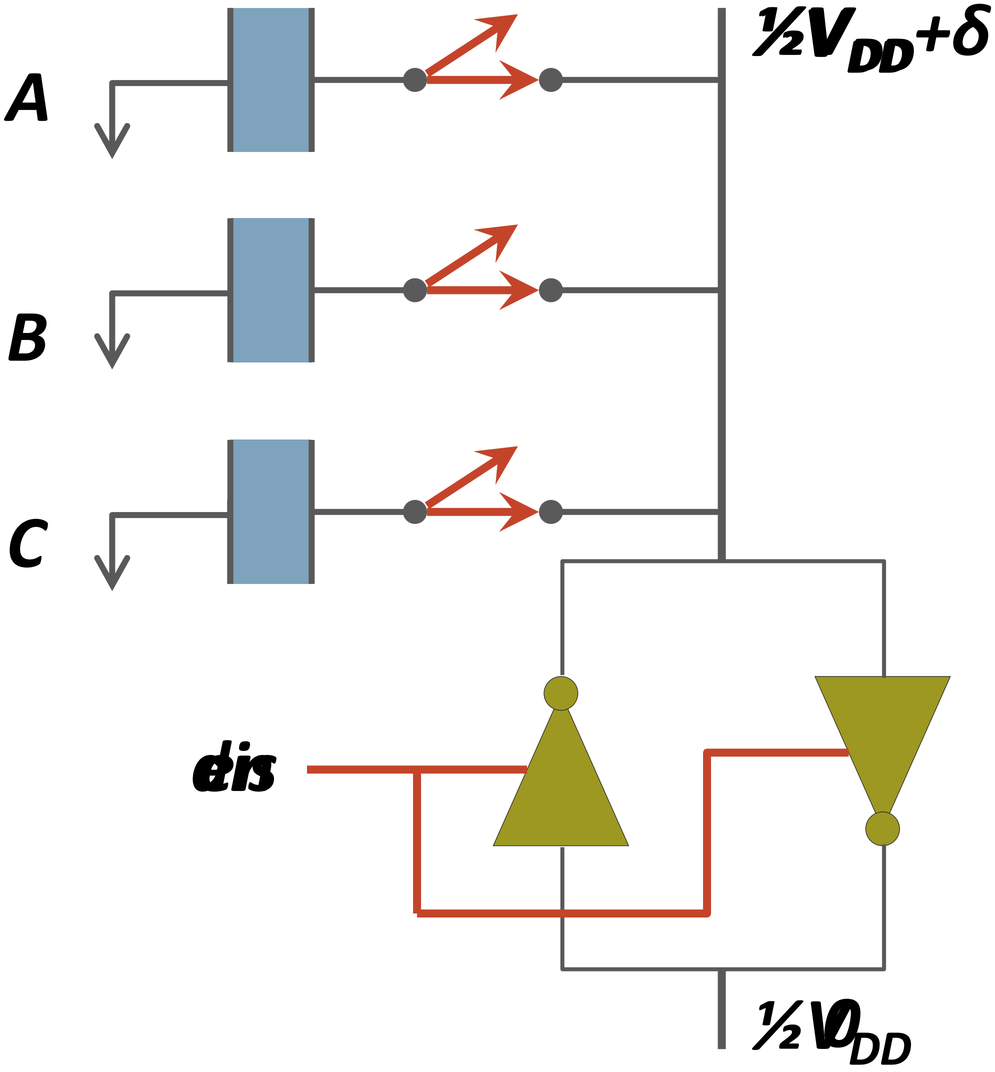
In-Memory Bulk Bitwise Operations

- We can support in-DRAM COPY, ZERO, AND, OR, NOT, MAJ
- At low cost

- Using analog computation capability of DRAM
  - Idea: activating multiple rows performs computation

- 30-60X performance and energy improvement
  - Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology," MICRO 2017.

In-DRAM AND/OR: Triple Row Activation



Final State

AB + BC + AC

C(A + B) + ~C(AB)

Seshadri+, "Fast Bulk Bitwise AND and OR in DRAM", IEEE CAL 2015.

In-DRAM Bulk Bitwise AND/OR Operation

- BULKAND A, B → C
- Semantics: Perform a bitwise AND of two rows A and B and store the result in row C

- R0 – reserved zero row, R1 – reserved one row
- D1, D2, D3 – Designated rows for triple activation

1. RowClone A into D1
2. RowClone B into D2
3. RowClone R0 into D3
4. ACTIVATE D1,D2,D3
5. RowClone Result into C

More on In-DRAM Bulk AND/OR

- Vivek Seshadri, Kevin Hsieh, Amirali Boroumand, Donghyuk Lee, Michael A. Kozuch, Onur Mutlu, Phillip B. Gibbons, and Todd C. Mowry,
  "Fast Bulk Bitwise AND and OR in DRAM"
  IEEE Computer Architecture Letters (***CAL***), April 2015.

Fast Bulk Bitwise AND and OR in DRAM

Vivek Seshadri*, Kevin Hsieh*, Amirali Boroumand*, Donghyuk Lee*,
Michael A. Kozuch†, Onur Mutlu*, Phillip B. Gibbons†, Todd C. Mowry*
*Carnegie Mellon University †Intel Pittsburgh

In-DRAM NOT: Dual Contact Cell

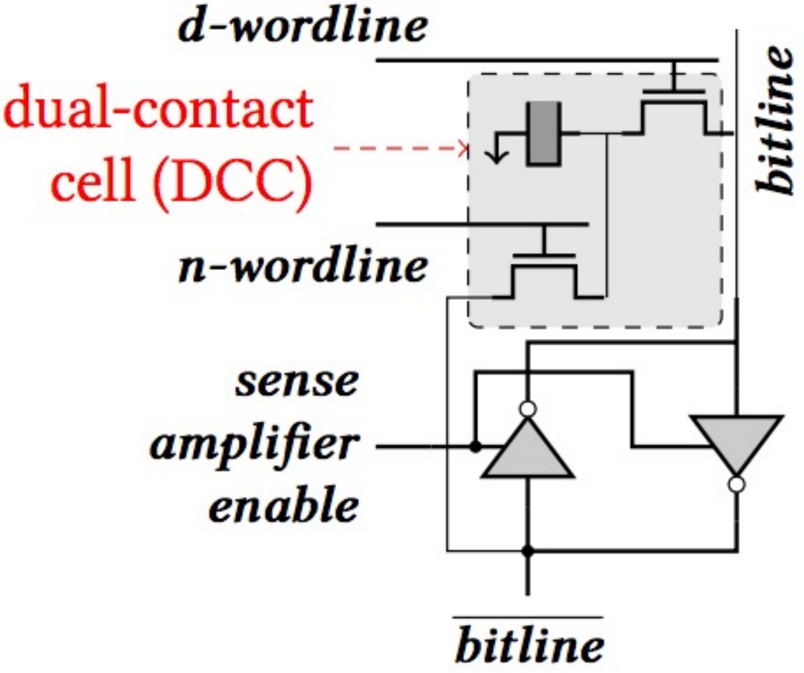
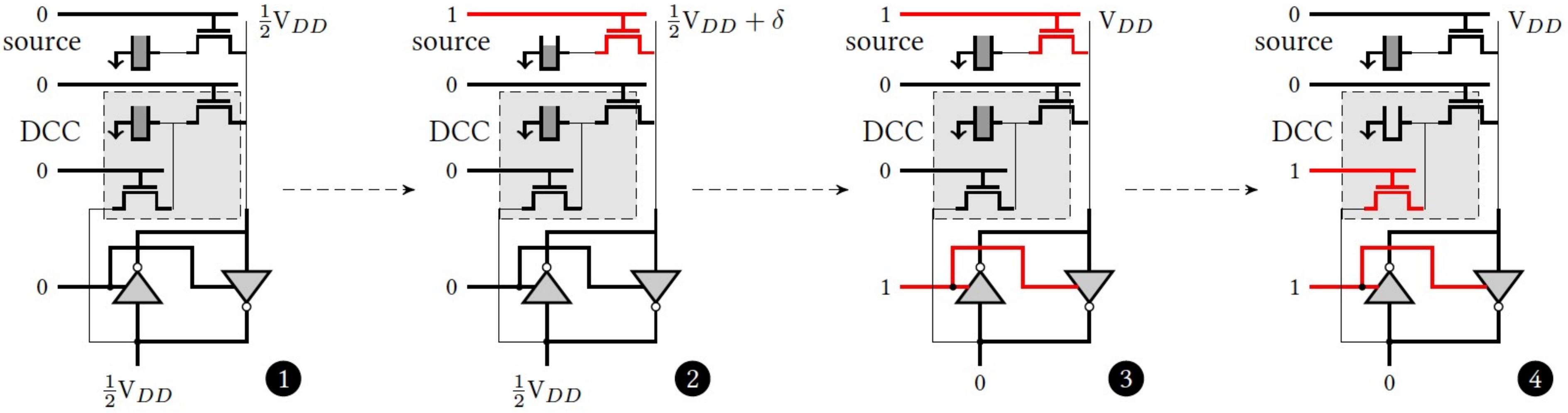


Figure 5: A dual-contact cell connected to both ends of a sense amplifier

Idea:
Feed the negated value in the sense amplifier into a special row

Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations using Commodity DRAM Technology," MICRO 2017

In-DRAM NOT Operation



Initial State

After Charge Sharing

Activated d-wordline

Activated n-wordline

Figure 5: Bitwise NOT using a dual contact capacitor

Seshadri+, “Ambit: In-Memory Accelerator for Bulk Bitwise Operations using Commodity DRAM Technology,” MICRO 2017

Performance: In-DRAM Bitwise Operations

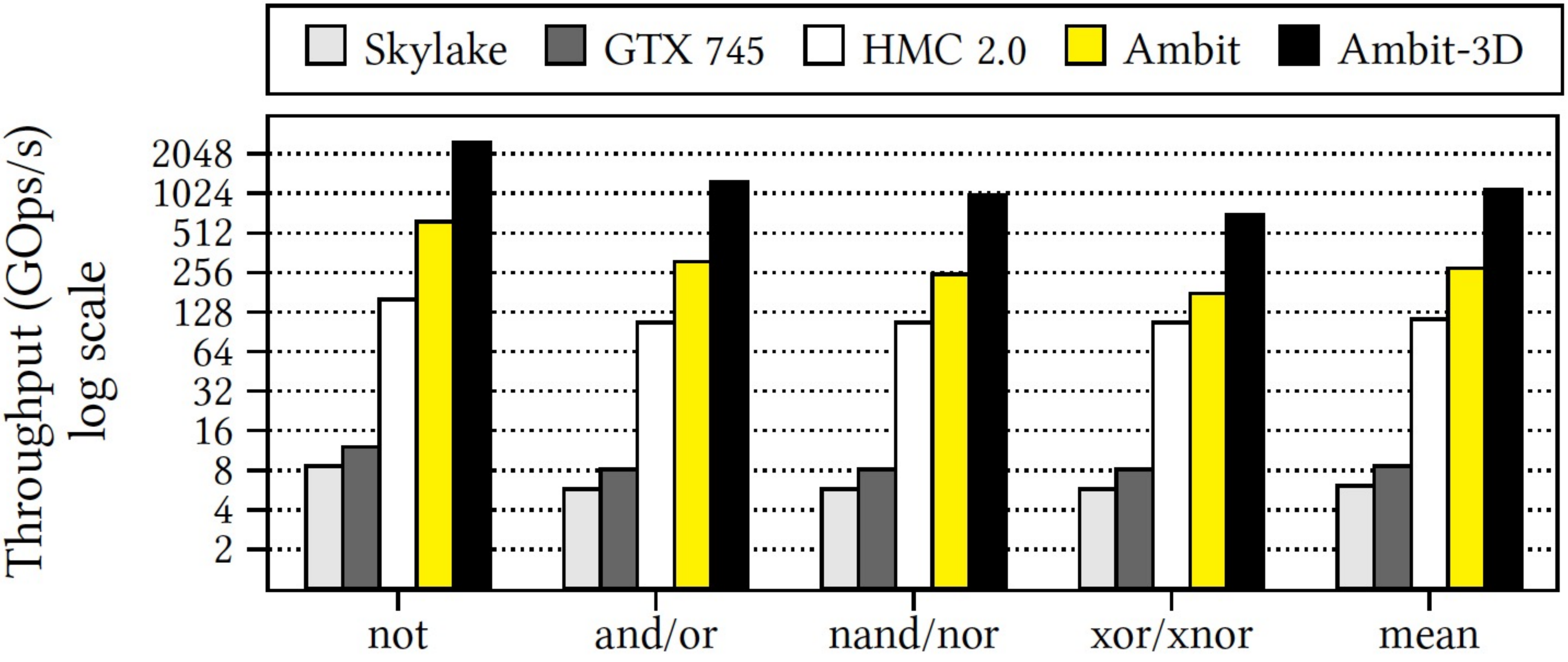
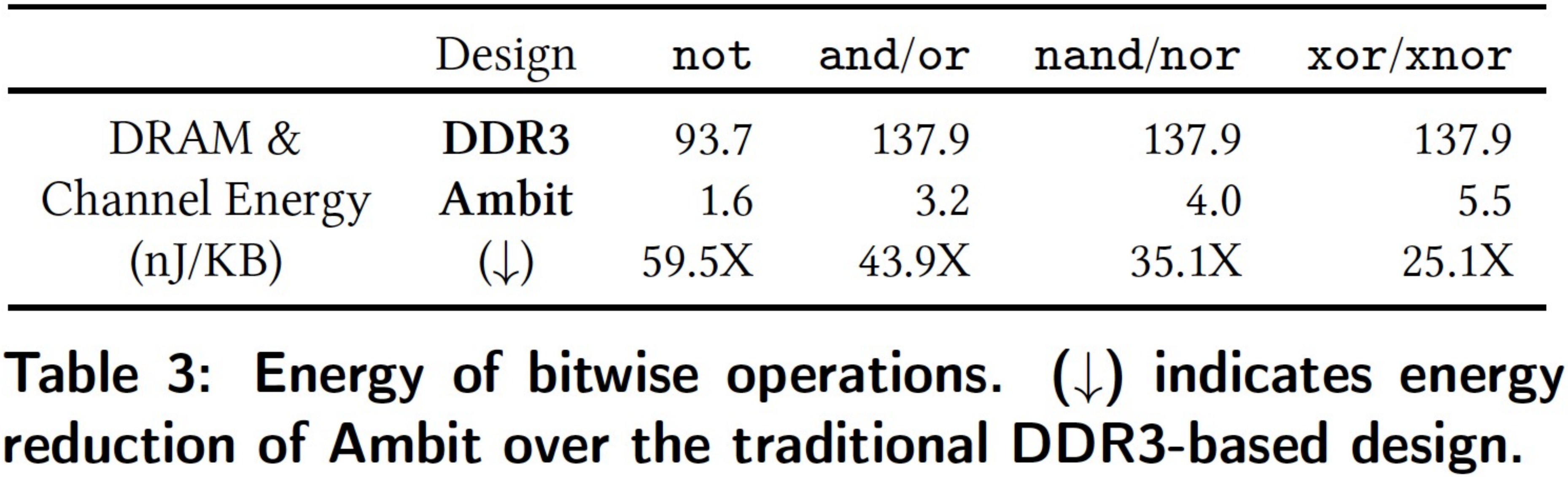


Figure 9: Throughput of bitwise operations on various systems.

Energy of In-DRAM Bitwise Operations



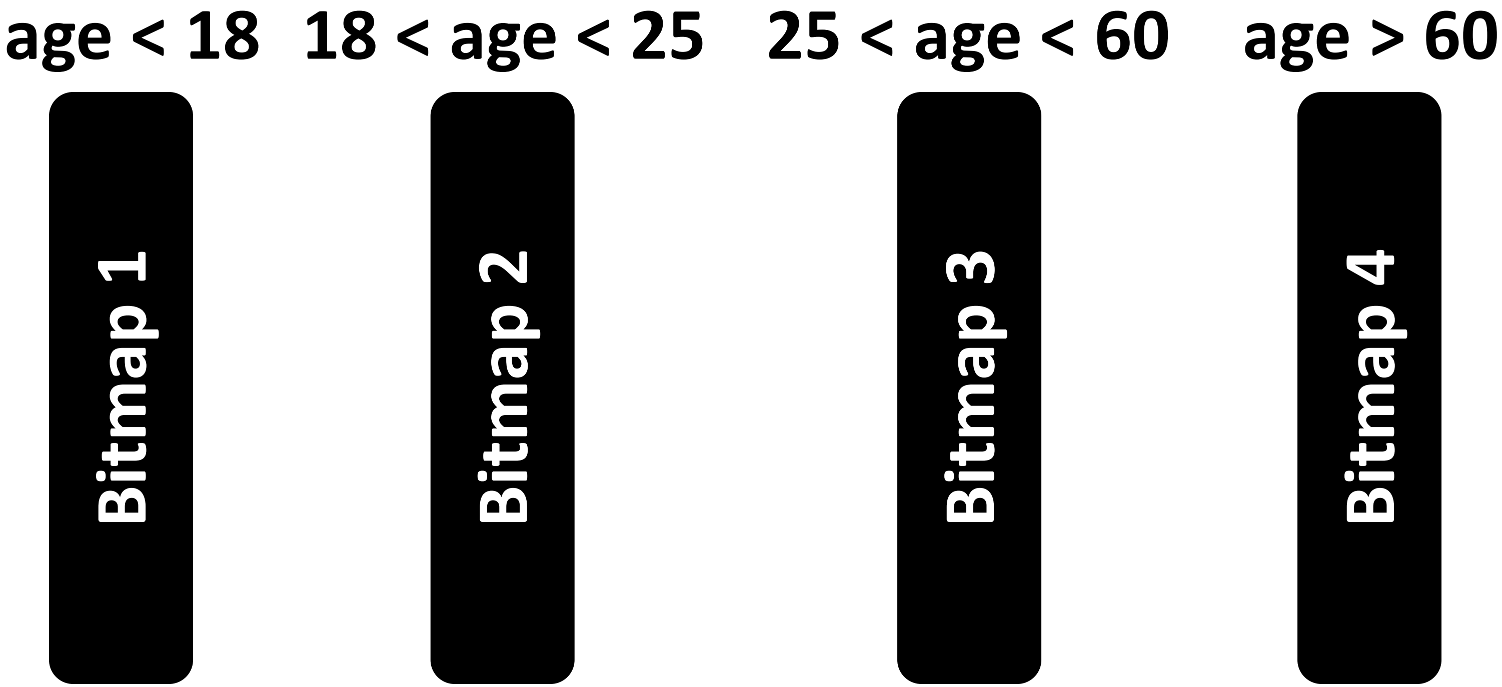
	Design	not	and/or	nand/nor	xor/xnor
DRAM & Channel Energy (nJ/KB)	**DDR3**	93.7	137.9	137.9	137.9
	Ambit	1.6	3.2	4.0	5.5
	(↓)	59.5X	43.9X	35.1X	25.1X

Table 3: Energy of bitwise operations. (↓) indicates energy reduction of Ambit over the traditional DDR3-based design.

Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations using Commodity DRAM Technology," MICRO 2017

Example Data Structure: Bitmap Index

- Alternative to B-tree and its variants
- Efficient for performing *range queries* and *joins*
- **Many bitwise operations to perform a query**



Performance: Bitmap Index on Ambit

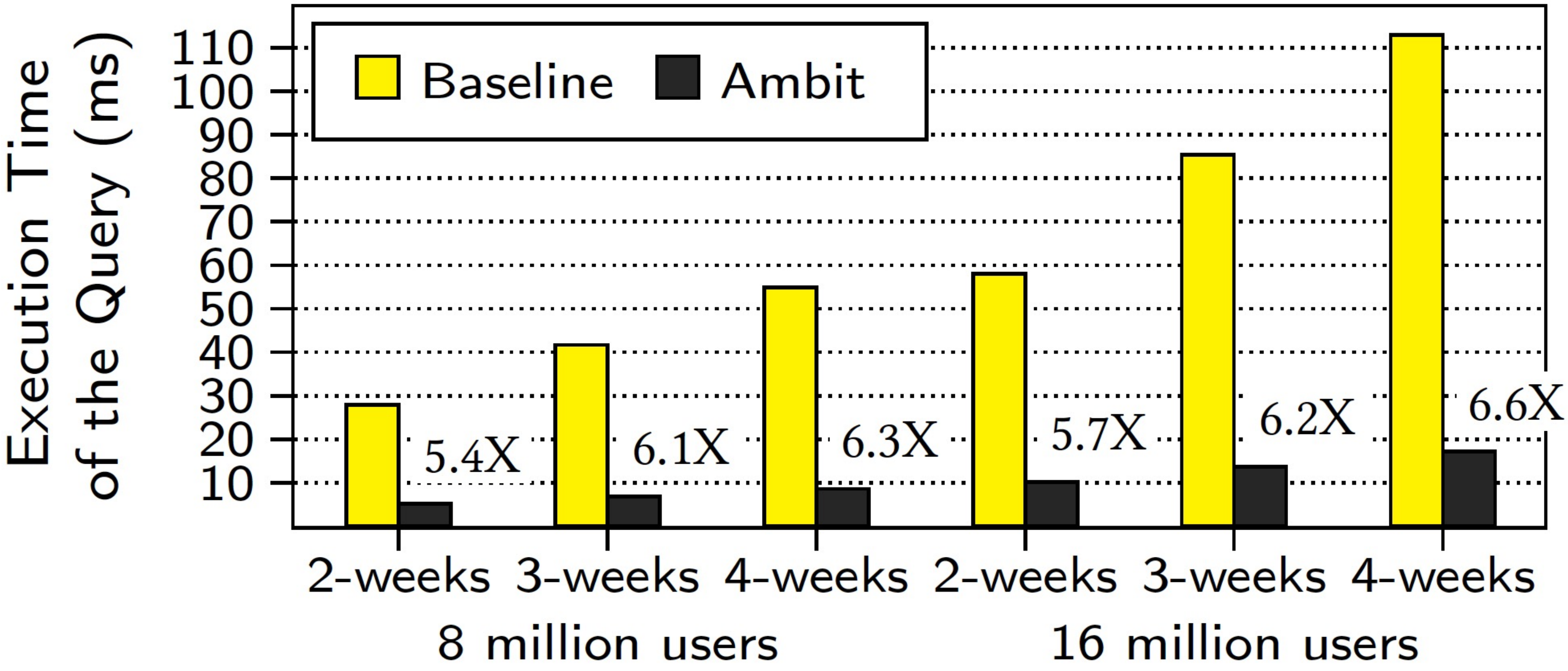
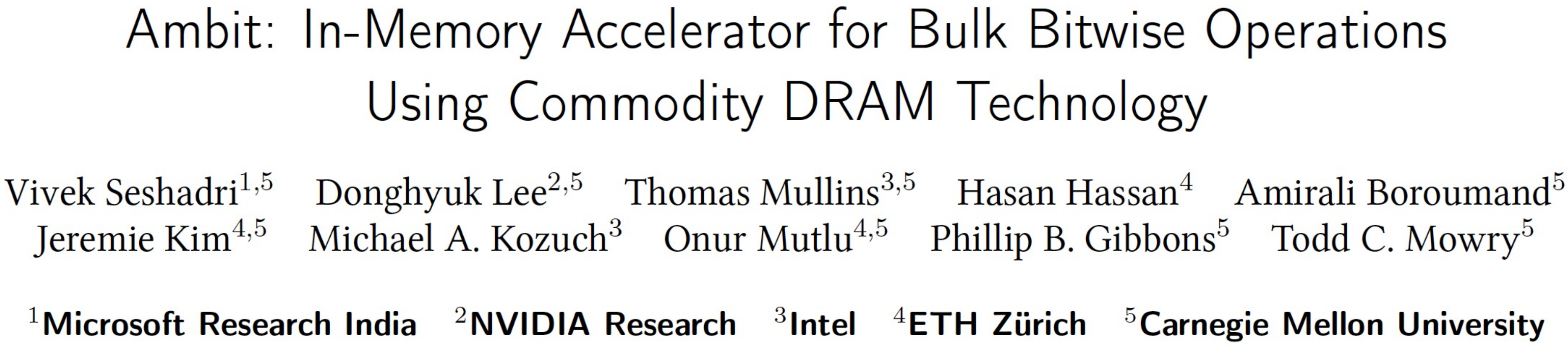


Figure 10: Bitmap index performance. The value above each bar indicates the reduction in execution time due to Ambit.

Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations using Commodity DRAM Technology," MICRO 2017

More on Ambit

- Vivek Seshadri et al., "**Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology**," MICRO 2017.



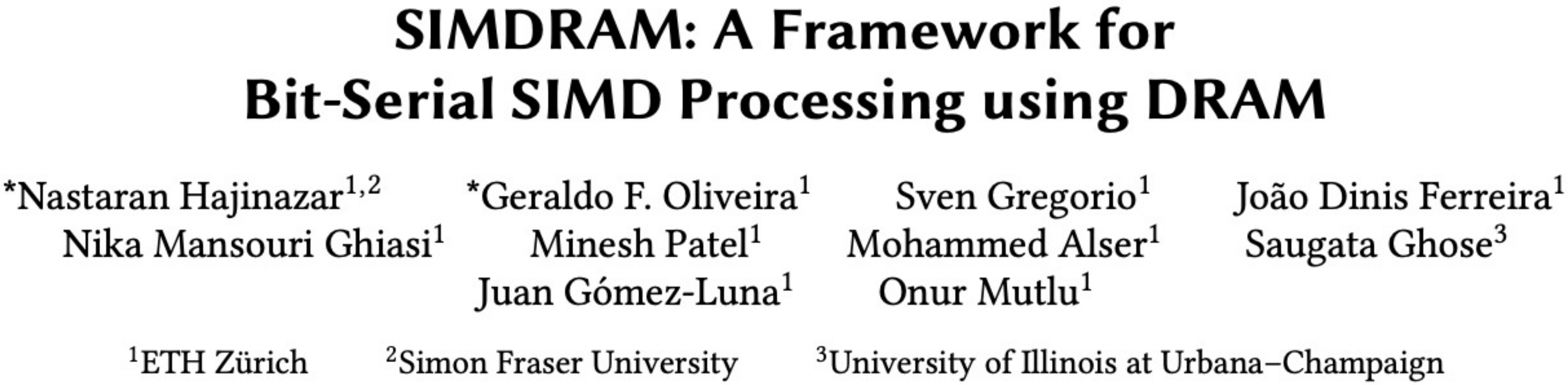
Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology

Vivek Seshadri[1,5] Donghyuk Lee[2,5] Thomas Mullins[3,5] Hasan Hassan[4] Amirali Boroumand[5]
Jeremie Kim[4,5] Michael A. Kozuch[3] Onur Mutlu[4,5] Phillip B. Gibbons[5] Todd C. Mowry[5]

[1]**Microsoft Research India** [2]**NVIDIA Research** [3]**Intel** [4]**ETH Zürich** [5]**Carnegie Mellon University**

SIMDRAM Framework

- Nastaran Hajinazar, Geraldo F. Oliveira, Sven Gregorio, Joao Dinis Ferreira, Nika Mansouri Ghiasi, Minesh Patel, Mohammed Alser, Saugata Ghose, Juan Gomez-Luna, and Onur Mutlu, **"SIMDRAM: An End-to-End Framework for Bit-Serial SIMD Computing in DRAM"** *Proceedings of the 26th International Conference on Architectural Support for Programming Languages and Operating Systems* (***ASPLOS***), Virtual, March-April 2021.
  [2-page Extended Abstract]
  [Short Talk Slides (pptx) (pdf)]
  [Talk Slides (pptx) (pdf)]
  [Short Talk Video (5 mins)]
  [Full Talk Video (27 mins)]



SIMDRAM: A Framework for Bit-Serial SIMD Processing using DRAM

*Nastaran Hajinazar[1,2] *Geraldo F. Oliveira[1] Sven Gregorio[1] João Dinis Ferreira[1]
Nika Mansouri Ghiasi[1] Minesh Patel[1] Mohammed Alser[1] Saugata Ghose[3]
Juan Gómez-Luna[1] Onur Mutlu[1]

[1]ETH Zürich [2]Simon Fraser University [3]University of Illinois at Urbana–Champaign

Agenda

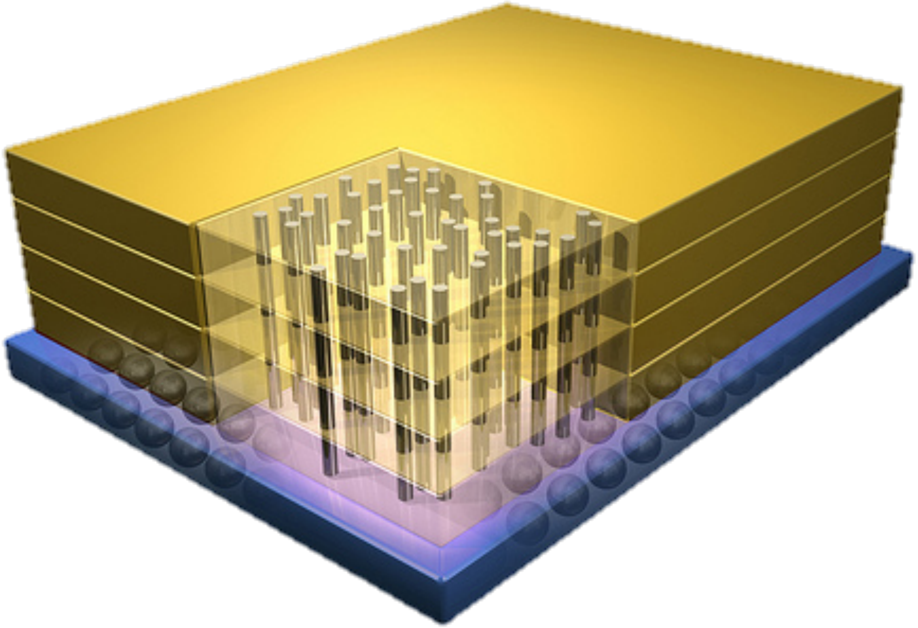
- Major Trends Affecting Memory

- Processing in Memory: Two Directions
  - Processing-using-Memory (PuM)
    - Minimally Changing Memory Chips

  - Processing-near-Memory (PnM)
    - Exploiting 3D-Stacked Memory

Approach 2: 3D-Stacked Logic+Memory

Hybrid Memory Cube
CONSORTIUM



Graph Processing

- Large graphs are everywhere (circa 2015)

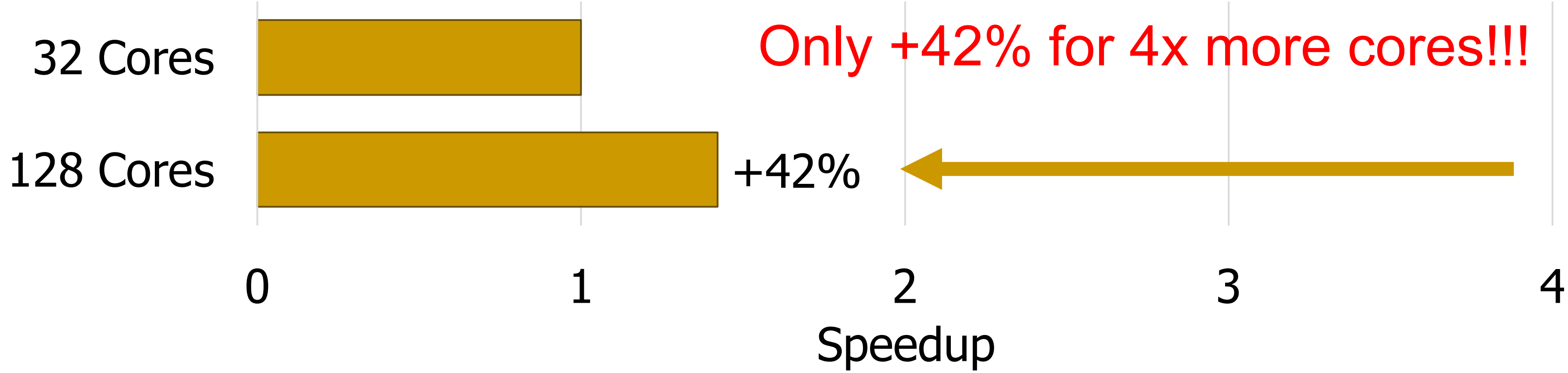
36 Million
Wikipedia Pages

1.4 Billion
Facebook Users

300 Million
Twitter Users

30 Billion
Instagram Photos

- Scalable large-scale graph processing is challenging

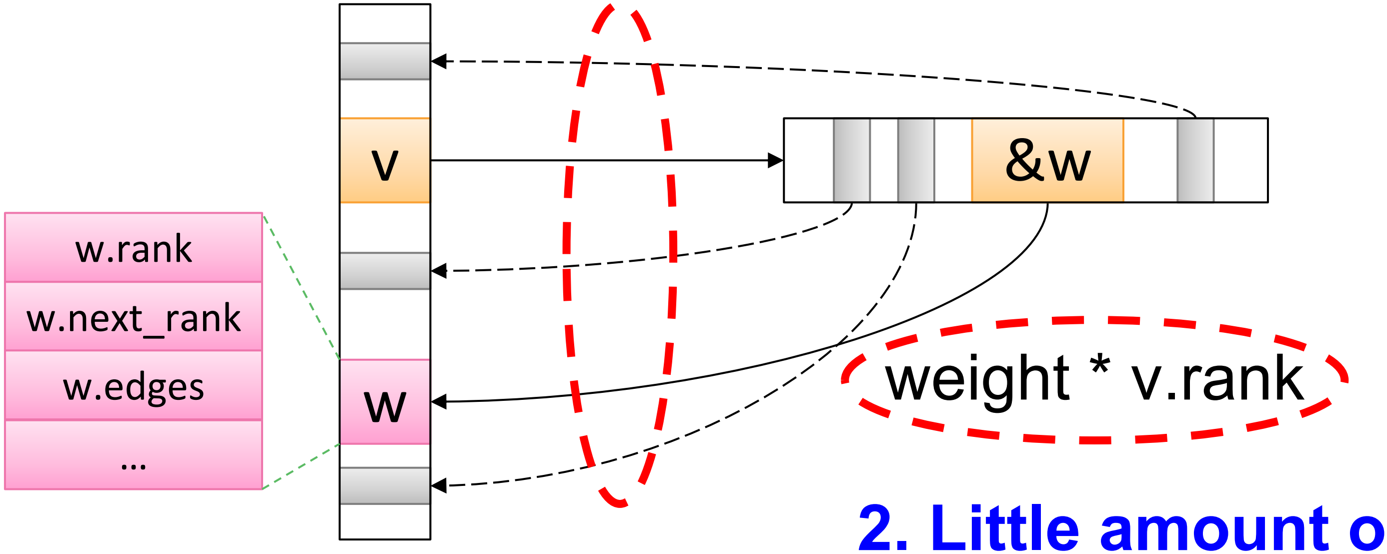


Only +42% for 4x more cores!!!

Key Bottlenecks in Graph Processing

PageRank algorithm (Page et al. 1999)

```
for (v: graph.vertices) {
    for (w: v.successors) {
        w.next_rank += weight * v.rank;
    }
}
```



1. Frequent random memory accesses

2. Little amount of computation

Two Key Questions in 3D-Stacked PIM

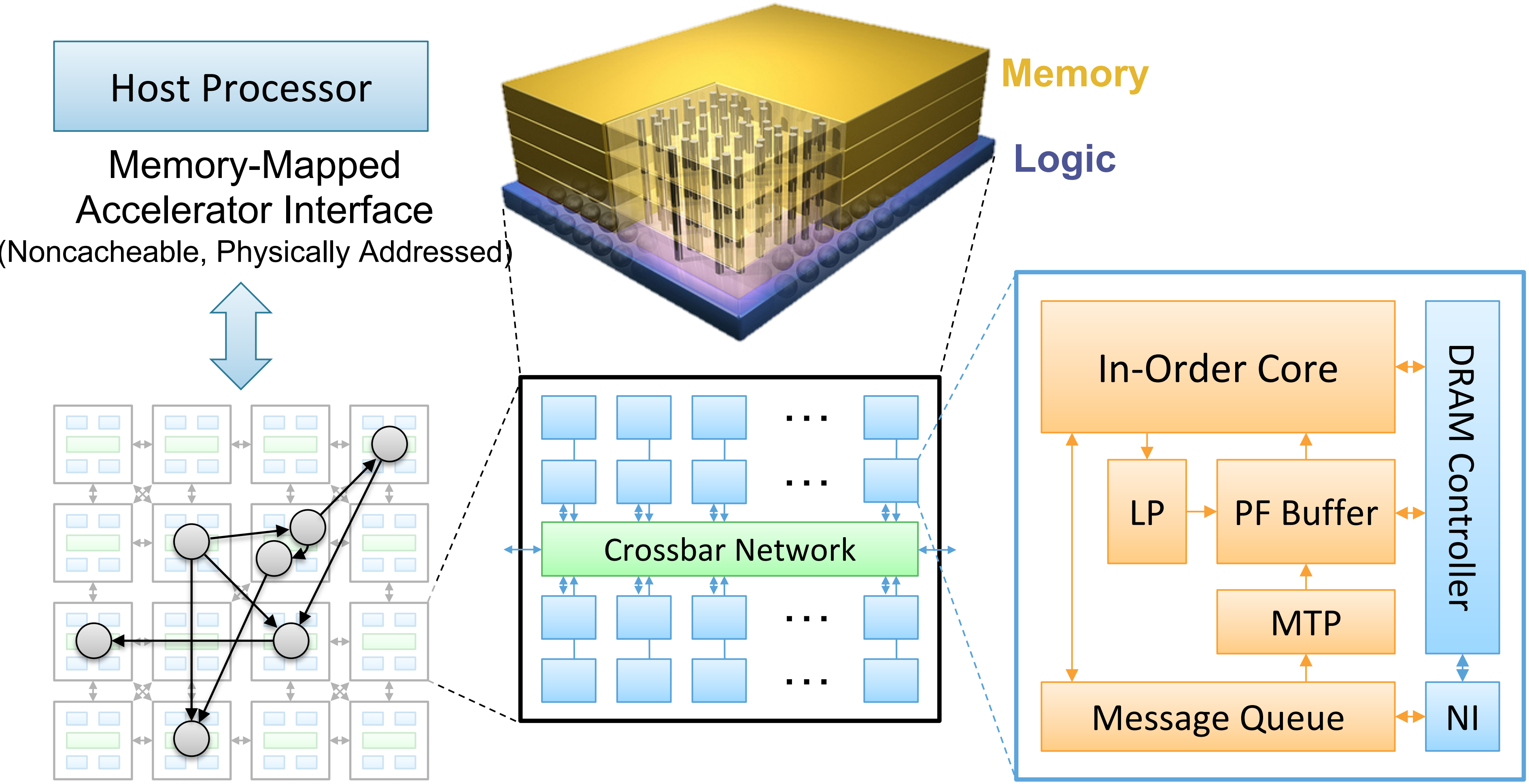
- How can we accelerate important applications if we use 3D-stacked memory as a coarse-grained accelerator?
  - what is the architecture and programming model?
  - what are the mechanisms for acceleration?

- What is the minimal processing-in-memory support we can provide?
  - without changing the system significantly
  - while achieving significant benefits

Tesseract: An In-Memory Accelerator for Graph Processing

Tesseract System for Graph Processing

Interconnected set of 3D-stacked memory+logic chips with simple cores



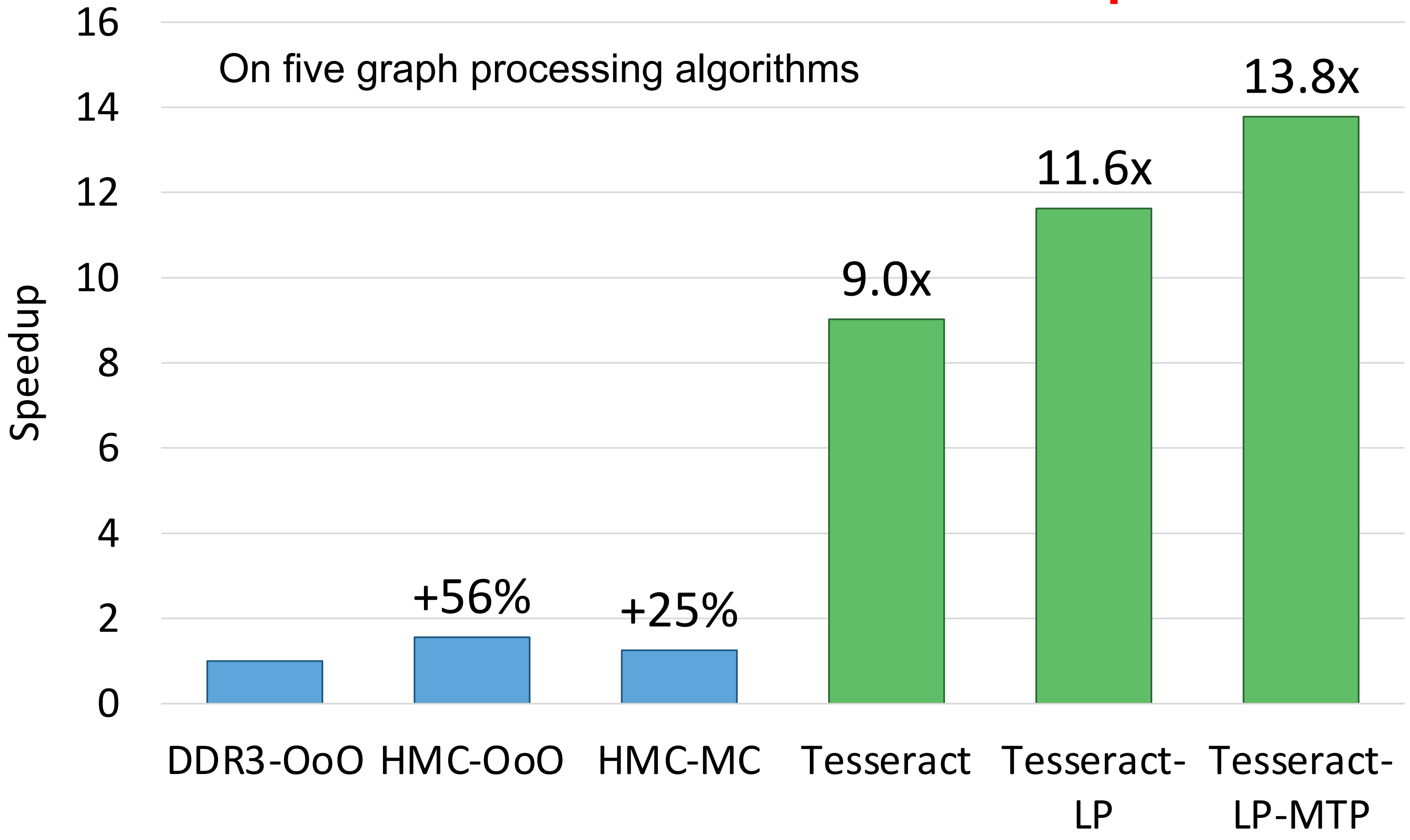
Ahn+, "A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing" ISCA 2015

Tesseract System for Graph Processing

- Evaluation on
  - DDR3 DRAM, computation on Out-of-Order (OoO) core
  - Hybrid Memory Cube (HMC) DRAM, computation on Out-of-Order (OoO) core
  - HMC DRAM, computation on the Memory Controller (MC)
  - Tesseract
    - With or without List Prefetching (LP)
    - With or without Message Triggered Prefetching (MTP), specified by the programmer

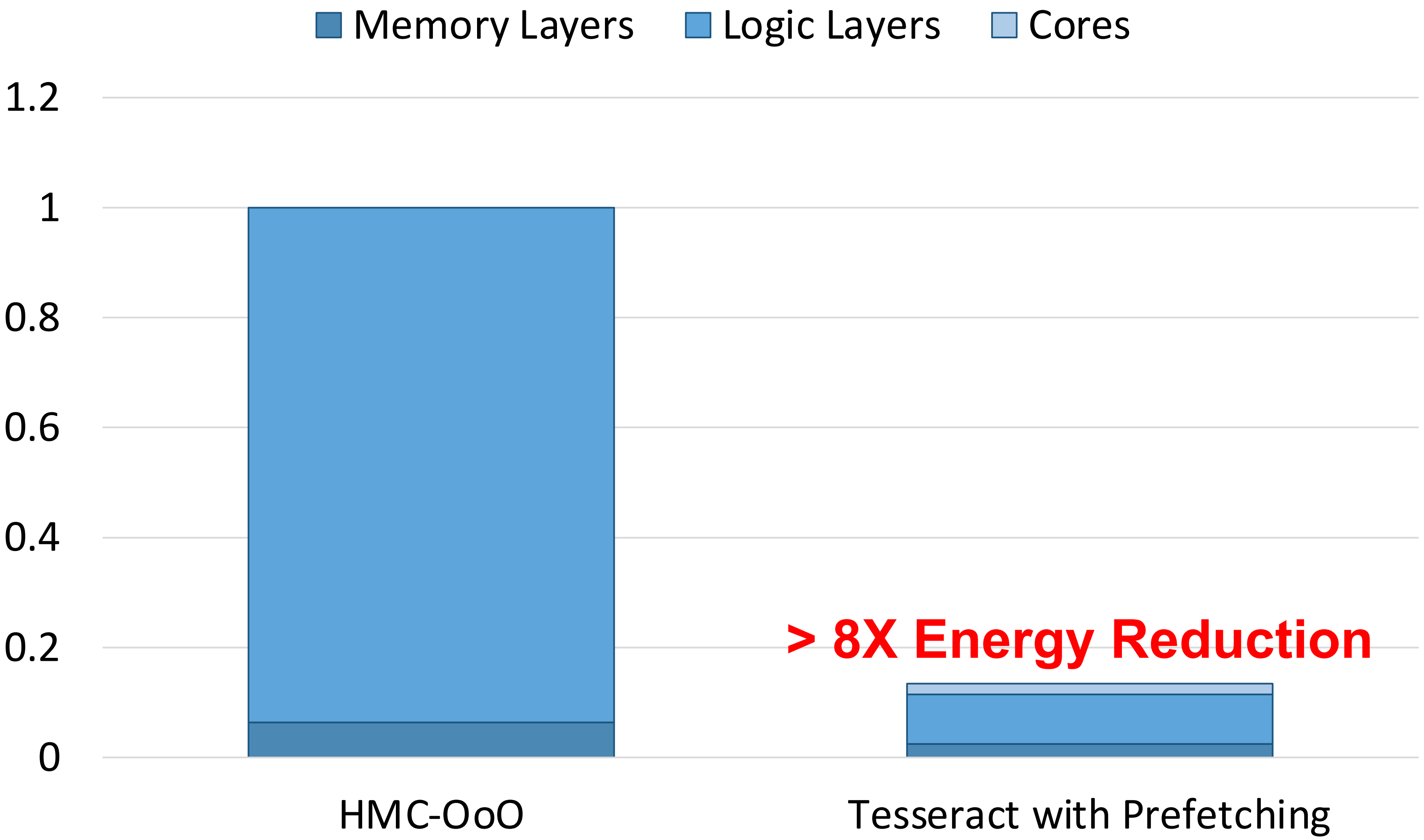
Tesseract Graph Processing Performance

>13X Performance Improvement



Ahn+, "A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing" ISCA 2015

Tesseract Graph Processing System Energy

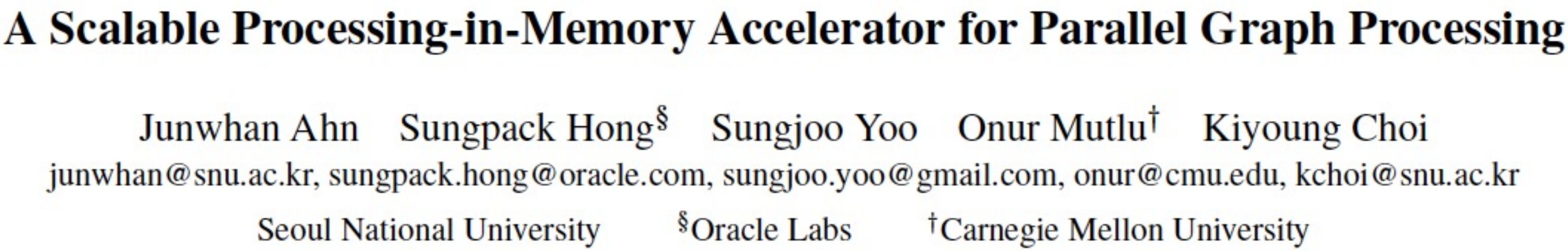


> 8X Energy Reduction

Ahn+, "A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing" ISCA 2015

More on Tesseract

- Junwhan Ahn, Sungpack Hong, Sungjoo Yoo, Onur Mutlu, and Kiyoung Choi,
  "A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing"
  *Proceedings of the 42nd International Symposium on Computer Architecture (**ISCA**)*, Portland, OR, June 2015.
  [Slides (pdf)] [Lightning Session Slides (pdf)]



A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing

Junwhan Ahn   Sungpack Hong[§]   Sungjoo Yoo   Onur Mutlu[†]   Kiyoung Choi
junwhan@snu.ac.kr, sungpack.hong@oracle.com, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr
Seoul National University   [§]Oracle Labs   [†]Carnegie Mellon University

Two Key Questions in 3D-Stacked PIM

- How can we accelerate important applications if we use 3D-stacked memory as a coarse-grained accelerator?
  - what is the architecture and programming model?
  - what are the mechanisms for acceleration?

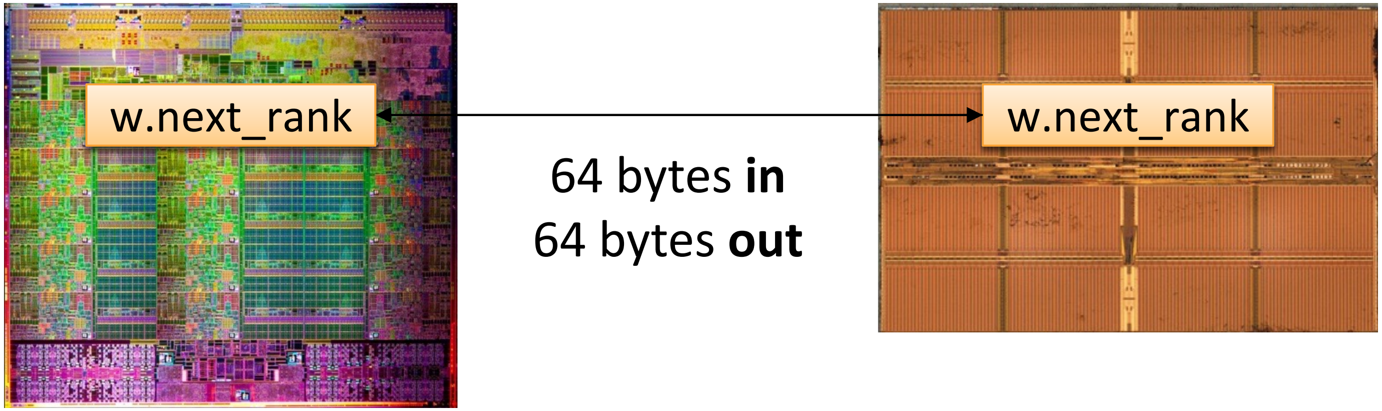
- What is the minimal processing-in-memory support we can provide?
  - without changing the system significantly
  - while achieving significant benefits

PIM-Enabled Instructions for Graph Processing

Simple PIM Operations as ISA Extensions (I)

PageRank algorithm (Page et al. 1999)

```
for (v: graph.vertices) {
    value = weight * v.rank;
    for (w: v.successors) {
        w.next_rank += value;
    }
}
```



Host Processor

Main Memory

64 bytes **in**
64 bytes **out**

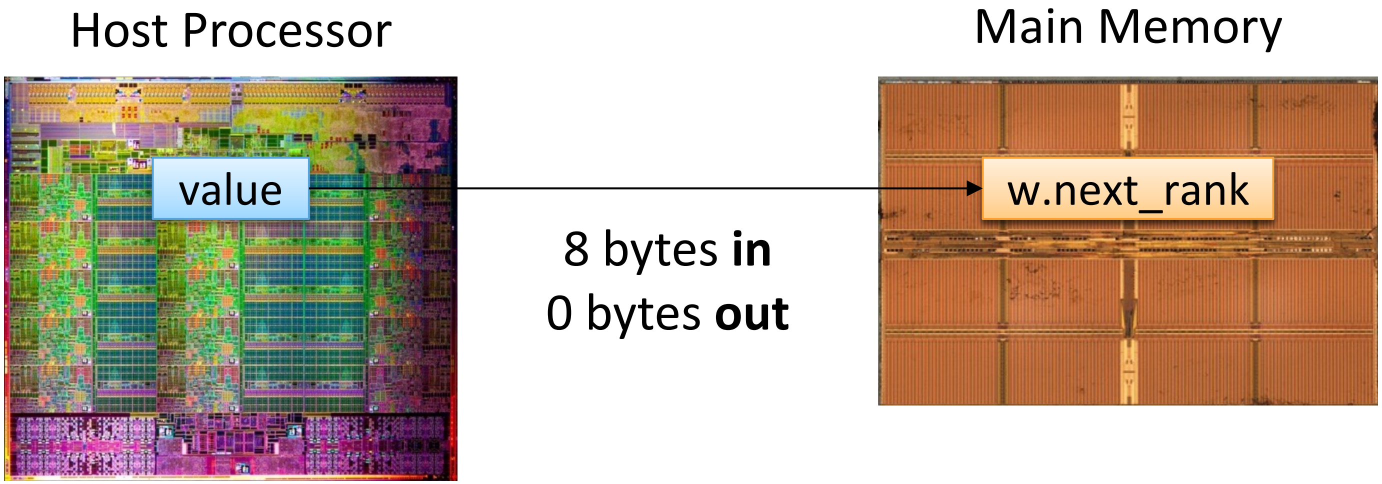
Conventional Architecture

Simple PIM Operations as ISA Extensions (II)

PageRank algorithm (Page et al. 1999)

```
for (v: graph.vertices) {
    value = weight * v.rank;
    for (w: v.successors) {
        __pim_add(&w.next_rank, value);
    }
}
```

```
pim.add r1, (r2)
```



8 bytes **in**
0 bytes **out**

In-Memory Addition

PEI: Benchmarks

- Graph processing
  - Average Teenage Follower (AT)
  - Breadth-First Search (BFS)
  - PageRank (PR)
  - Single-Source Shortest Path (SP)
  - Weakly Connected Components (WCC)

- Other benchmarks that can benefit from PEI
  - Data analytics
    - Hash Join (HJ)
    - Histogram (HG)
    - Radix Partitioning (RP)
  - Machine learning and data mining
    - Streamcluster (SC)
    - Support Vector Machine (SVM)

PEI: PIM-Enabled Instructions: Examples

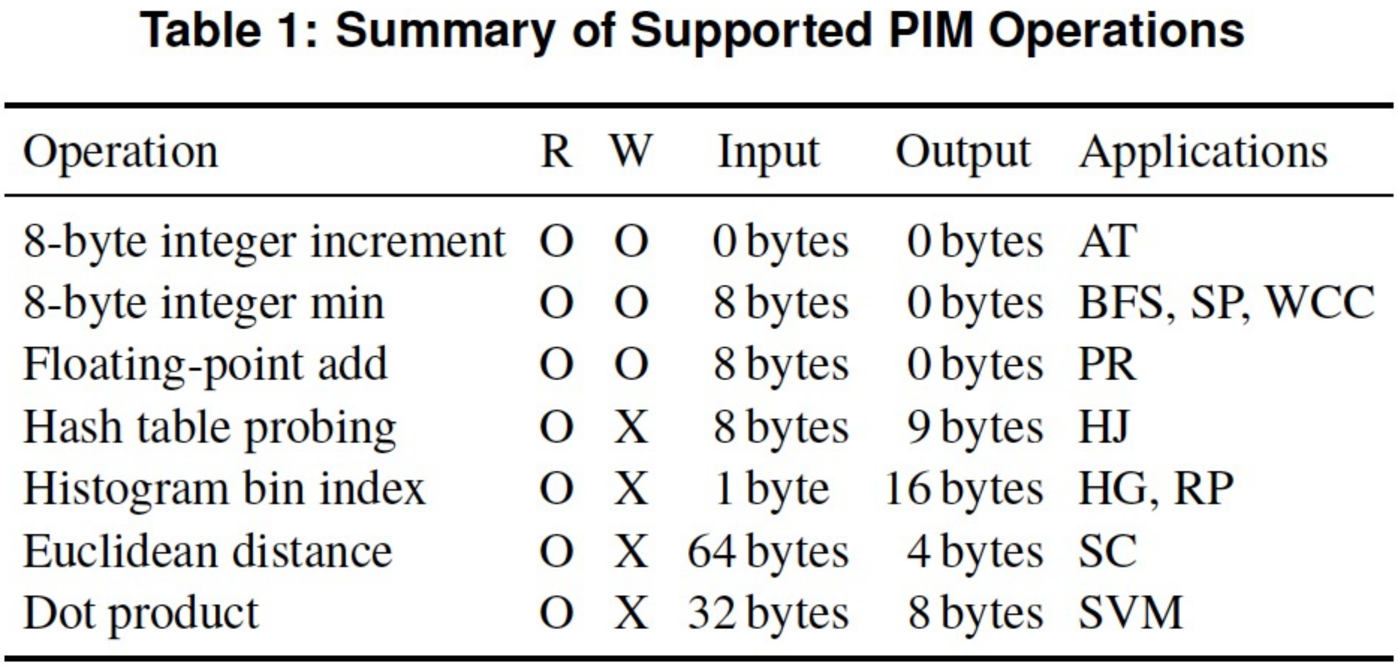
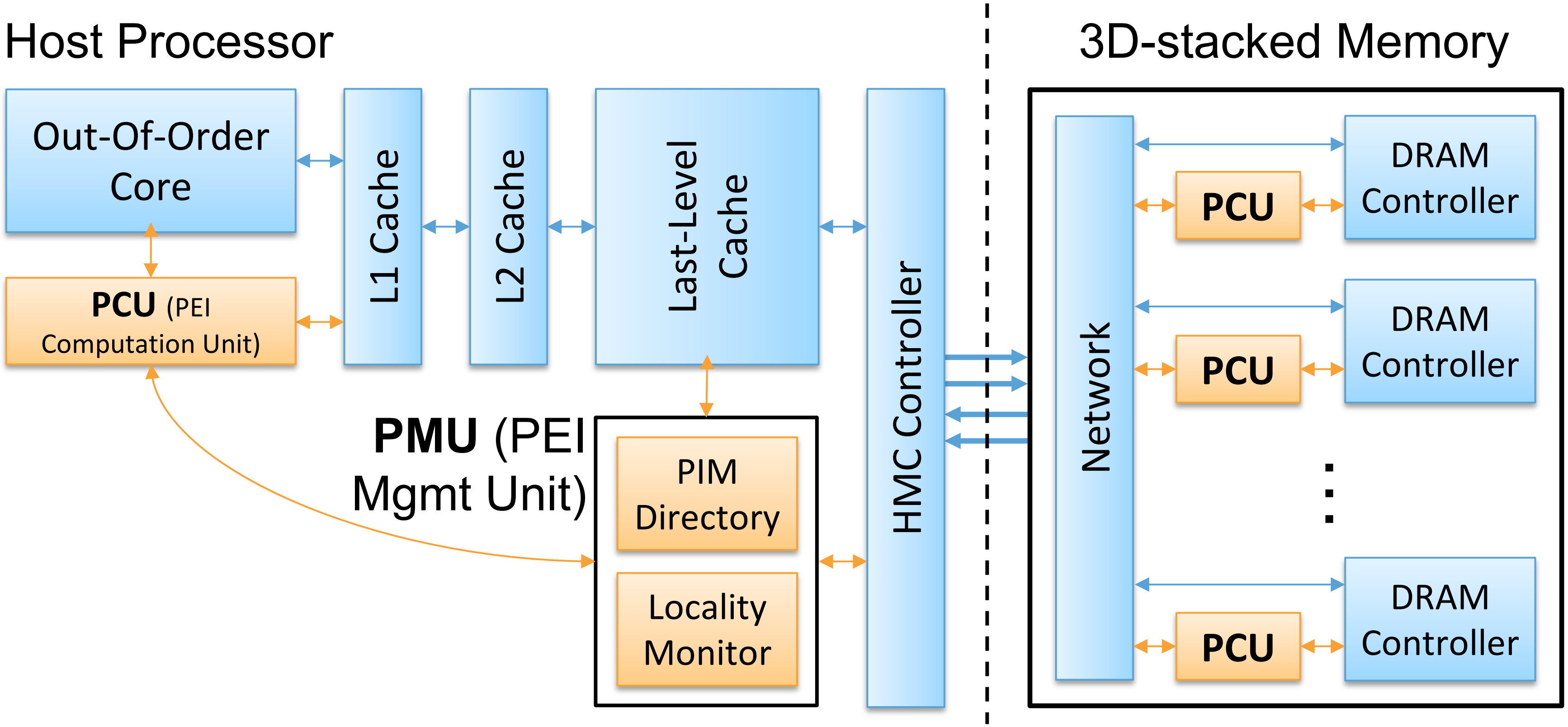


Table 1: Summary of Supported PIM Operations

Operation	R	W	Input	Output	Applications
8-byte integer increment	O	O	0 bytes	0 bytes	AT
8-byte integer min	O	O	8 bytes	0 bytes	BFS, SP, WCC
Floating-point add	O	O	8 bytes	0 bytes	PR
Hash table probing	O	X	8 bytes	9 bytes	HJ
Histogram bin index	O	X	1 byte	16 bytes	HG, RP
Euclidean distance	O	X	64 bytes	4 bytes	SC
Dot product	O	X	32 bytes	8 bytes	SVM

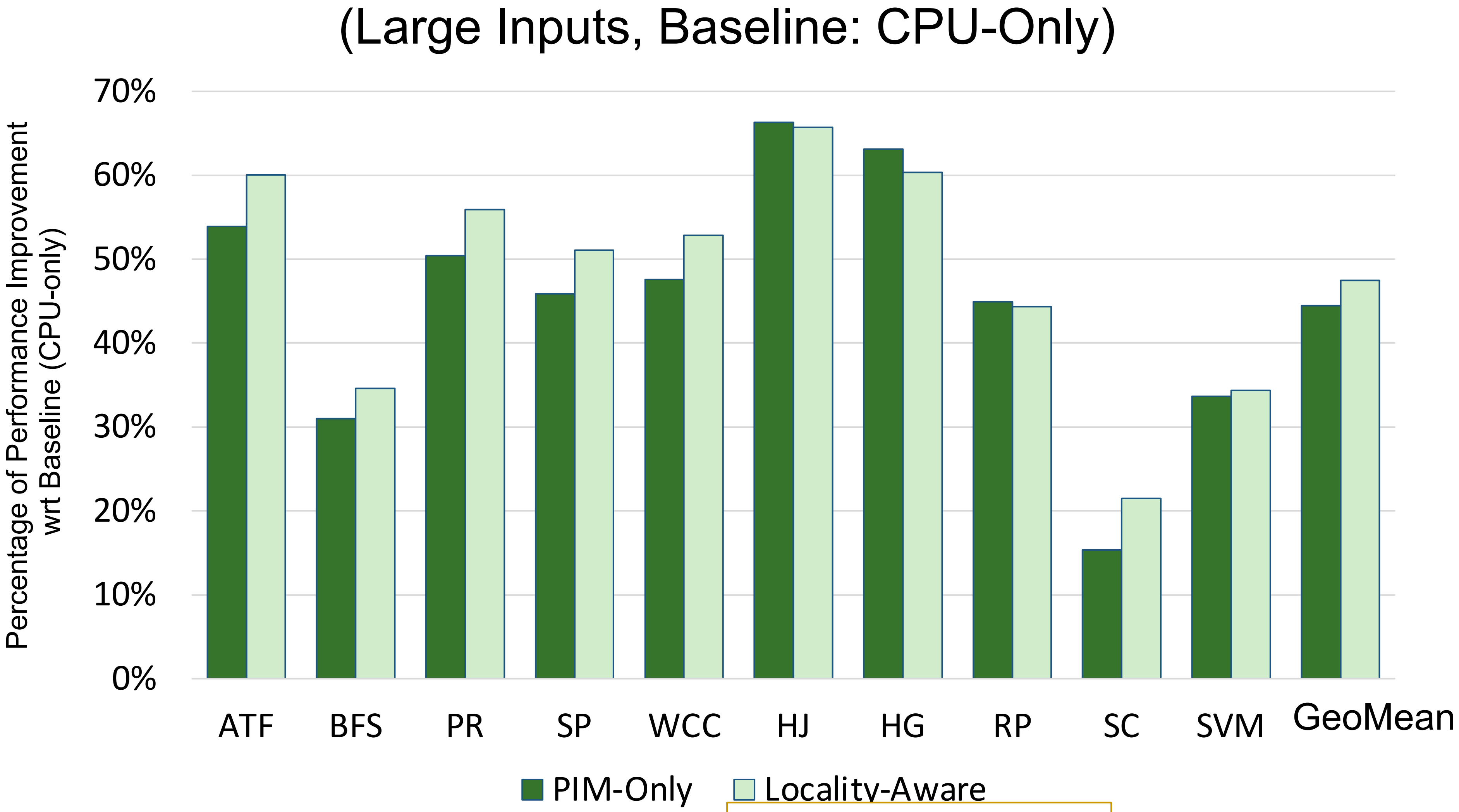
- Executed either in memory or in the processor: dynamic decision
  - Low-cost locality monitoring for a single instruction
- Cache-coherent, virtually-addressed, single cache block only
- Atomic between different PEIs
- *Not* atomic with normal instructions (use *pfence* for ordering)

Example PEI Microarchitecture



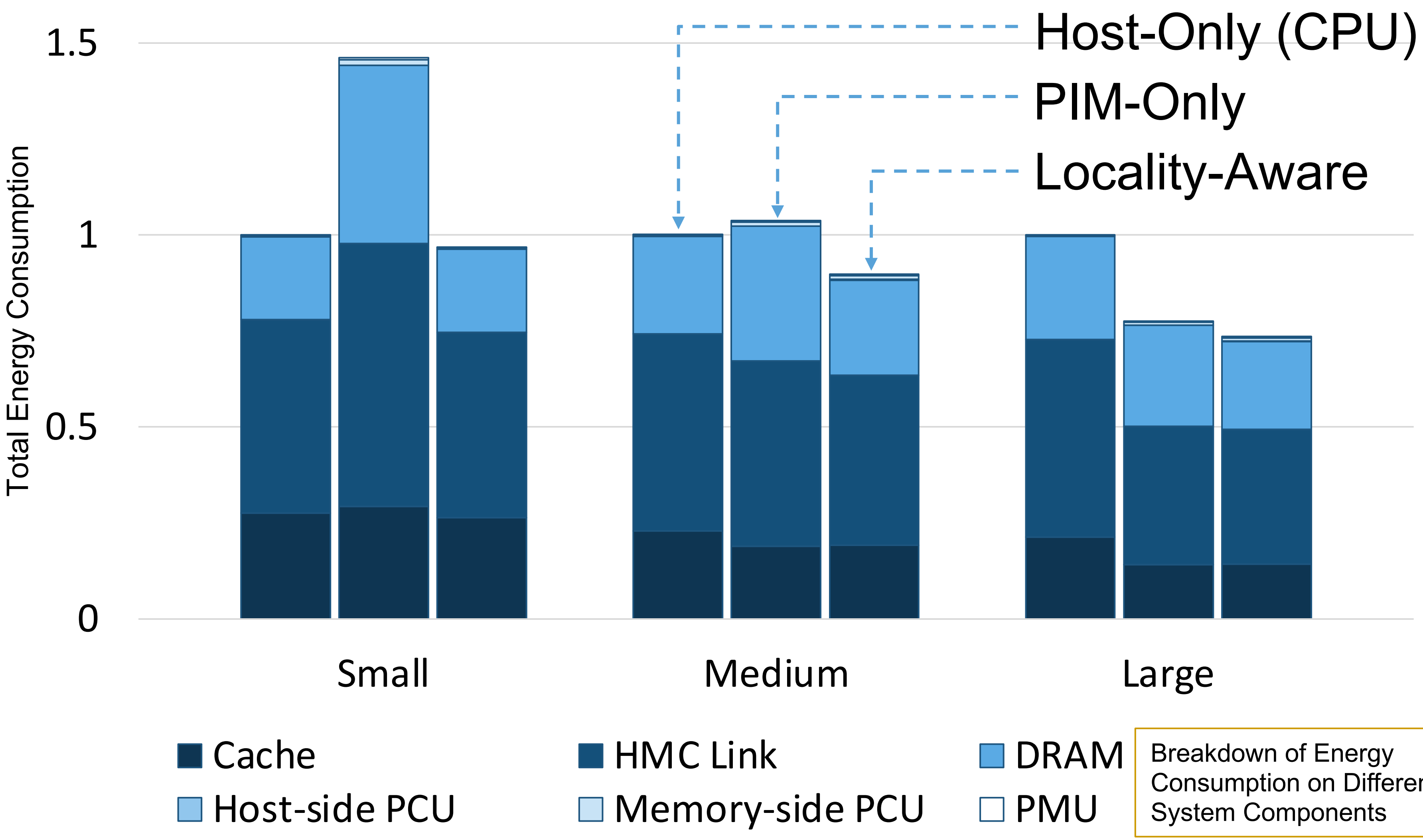
Example PEI uArchitecture

PEI Performance Delta: Large Data Sets



Locality-Aware = PIM or CPU depending on data location

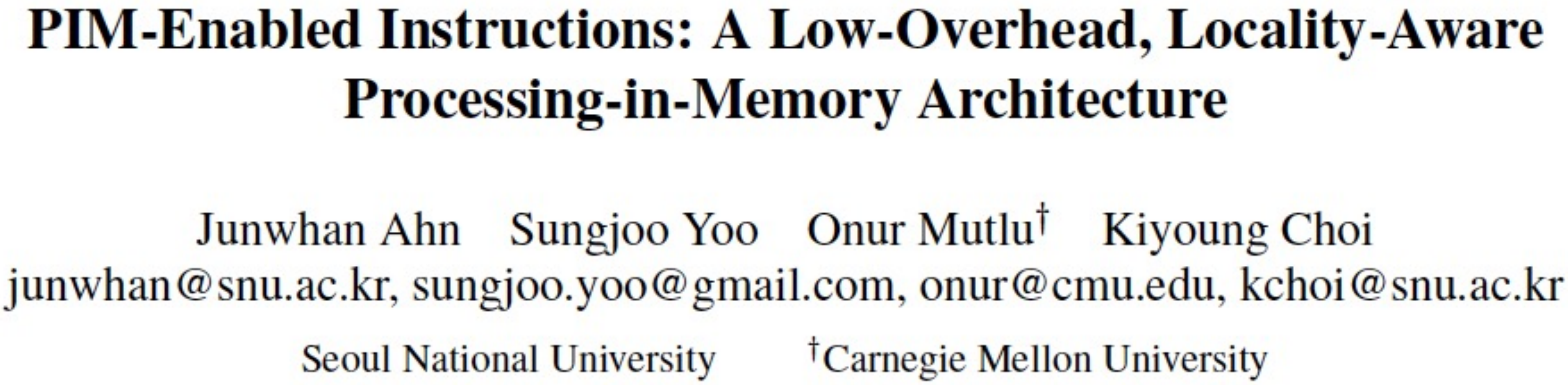
PEI Energy Consumption



Breakdown of Energy Consumption on Different System Components

More on PIM-Enabled Instructions

- Junwhan Ahn, Sungjoo Yoo, Onur Mutlu, and Kiyoung Choi, **"PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture"** *Proceedings of the 42nd International Symposium on Computer Architecture* (***ISCA***), Portland, OR, June 2015. [Slides (pdf)] [Lightning Session Slides (pdf)]



PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture

Junwhan Ahn Sungjoo Yoo Onur Mutlu[†] Kiyoung Choi
junwhan@snu.ac.kr, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr
Seoul National University [†]Carnegie Mellon University

Agenda

- Major Trends Affecting Memory

- Processing in Memory: Two Directions
  - Processing-using-Memory (PuM)
    - Minimally Changing Memory Chips
  - Processing-near-Memory (PnM)
    - Exploiting 3D-Stacked Memory

Eliminating the Adoption Barriers

How to Enable Adoption of Processing in Memory

Barriers to Adoption of PIM

1. Functionality of and applications & software for PIM

2. Ease of programming (interfaces and compiler/HW support)

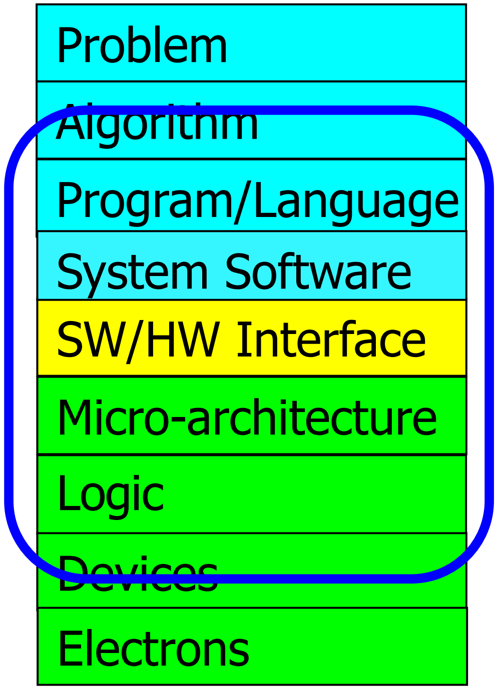
3. System support: coherence & virtual memory

4. Runtime and compilation systems for adaptive scheduling, data mapping, access/sharing control

5. Infrastructures to assess benefits and feasibility

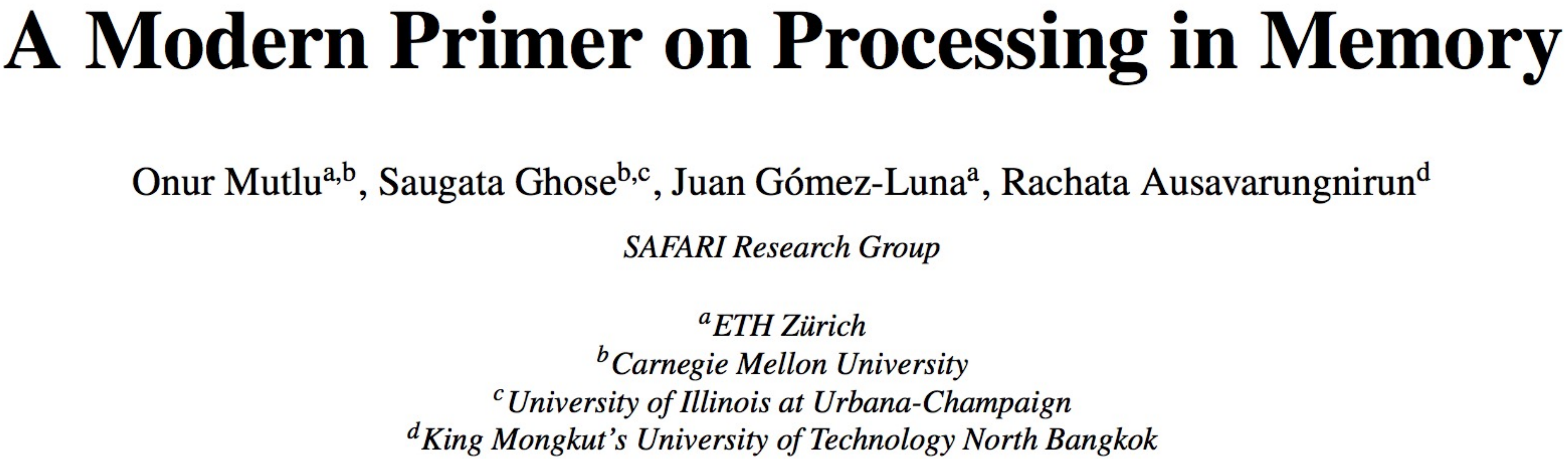
All can be solved with change of mindset

We Need to Revisit the Entire Stack



We can get there step by step

PIM Review and Open Problems



A Modern Primer on Processing in Memory

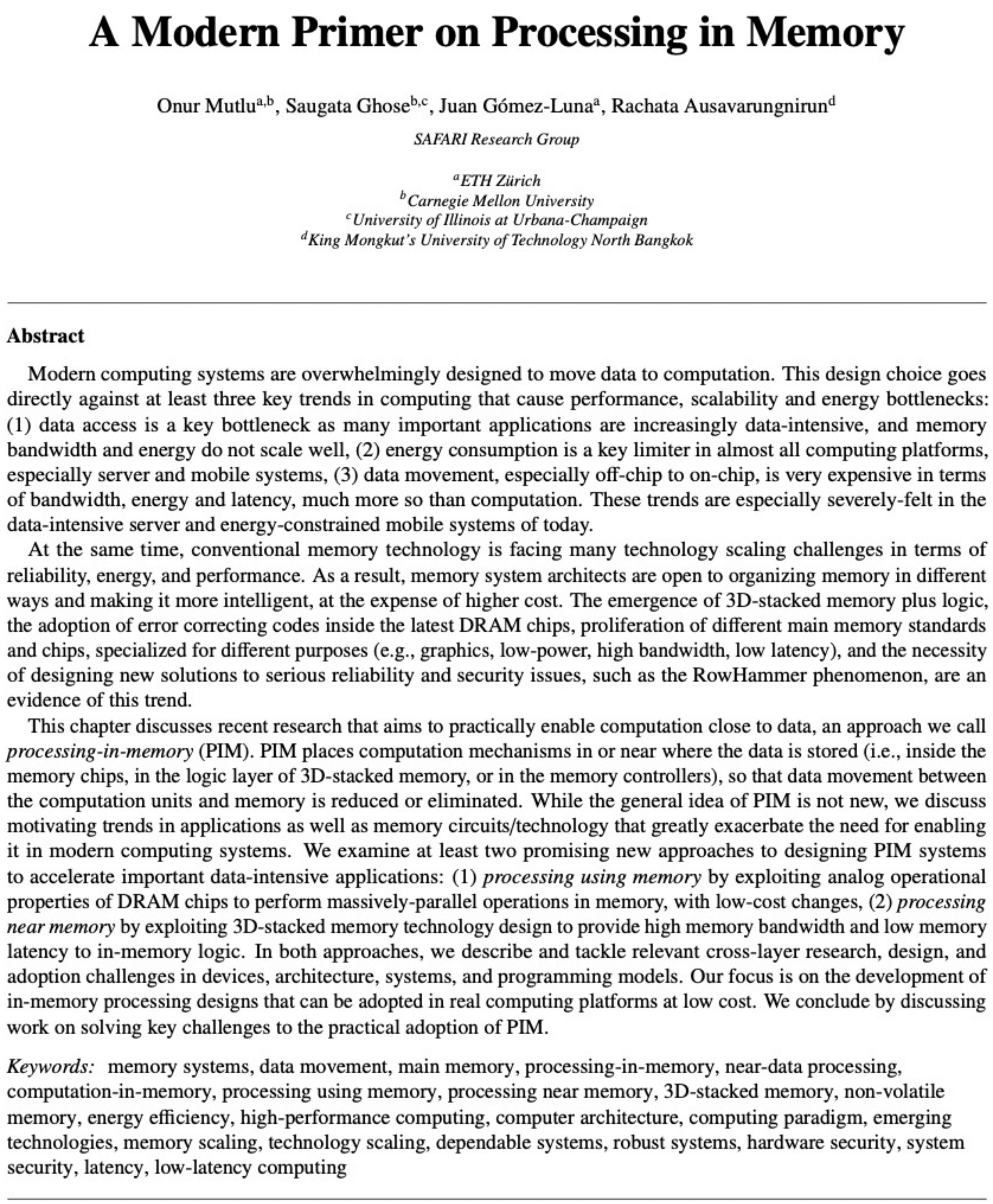
Onur Mutlu[a,b], Saugata Ghose[b,c], Juan Gómez-Luna[a], Rachata Ausavarungnirun[d]

SAFARI Research Group

[a] *ETH Zürich*
[b] *Carnegie Mellon University*
[c] *University of Illinois at Urbana-Champaign*
[d] *King Mongkut's University of Technology North Bangkok*

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun,
"A Modern Primer on Processing in Memory"
Invited Book Chapter in ***Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann***, Springer, to be published in 2021.

https://arxiv.org/pdf/1903.03988.pdf



A Modern Primer on Processing in Memory

Onur Mutlu[a,b], Saugata Ghose[b,c], Juan Gómez-Luna[a], Rachata Ausavarungnirun[d]

SAFARI Research Group

[a] *ETH Zürich*
[b] *Carnegie Mellon University*
[c] *University of Illinois at Urbana-Champaign*
[d] *King Mongkut's University of Technology North Bangkok*

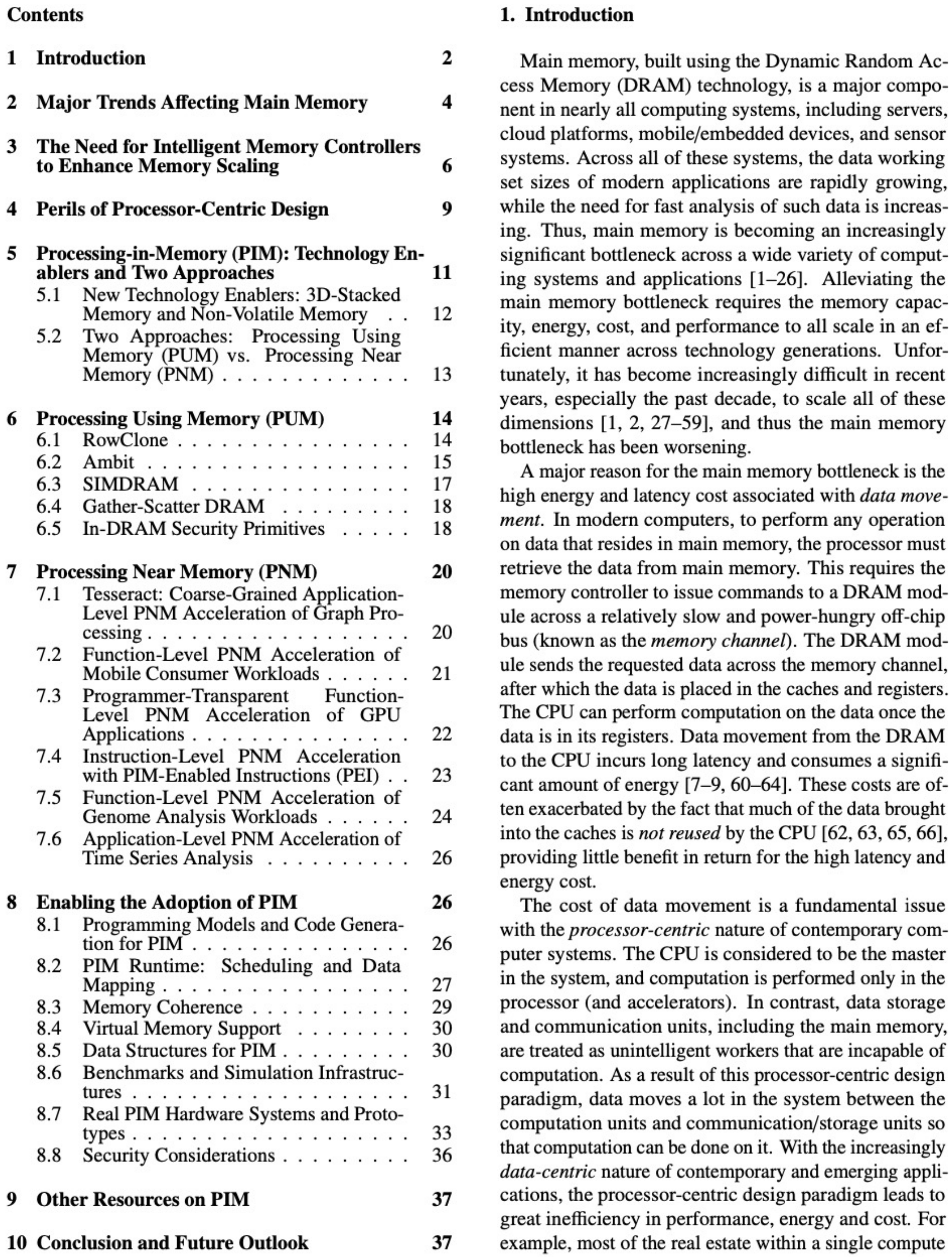
Abstract

Modern computing systems are overwhelmingly designed to move data to computation. This design choice goes directly against at least three key trends in computing that cause performance, scalability and energy bottlenecks: (1) data access is a key bottleneck as many important applications are increasingly data-intensive, and memory bandwidth and energy do not scale well, (2) energy consumption is a key limiter in almost all computing platforms, especially server and mobile systems, (3) data movement, especially off-chip to on-chip, is very expensive in terms of bandwidth, energy and latency, much more so than computation. These trends are especially severely-felt in the data-intensive server and energy-constrained mobile systems of today.

At the same time, conventional memory technology is facing many technology scaling challenges in terms of reliability, energy, and performance. As a result, memory system architects are open to organizing memory in different ways and making it more intelligent, at the expense of higher cost. The emergence of 3D-stacked memory plus logic, the adoption of error correcting codes inside the latest DRAM chips, proliferation of different main memory standards and chips, specialized for different purposes (e.g., graphics, low-power, high bandwidth, low latency), and the necessity of designing new solutions to serious reliability and security issues, such as the RowHammer phenomenon, are an evidence of this trend.

This chapter discusses recent research that aims to practically enable computation close to data, an approach we call *processing-in-memory* (PIM). PIM places computation mechanisms in or near where the data is stored (i.e., inside the memory chips, in the logic layer of 3D-stacked memory, or in the memory controllers), so that data movement between the computation units and memory is reduced or eliminated. While the general idea of PIM is not new, we discuss motivating trends in applications as well as memory circuits/technology that greatly exacerbate the need for enabling it in modern computing systems. We examine at least two promising new approaches to designing PIM systems to accelerate important data-intensive applications: (1) *processing using memory* by exploiting analog operational properties of DRAM chips to perform massively-parallel operations in memory, with low-cost changes, (2) *processing near memory* by exploiting 3D-stacked memory technology design to provide high memory bandwidth and low memory latency to in-memory logic. In both approaches, we describe and tackle relevant cross-layer research, design, and adoption challenges in devices, architecture, systems, and programming models. Our focus is on the development of in-memory processing designs that can be adopted in real computing platforms at low cost. We conclude by discussing work on solving key challenges to the practical adoption of PIM.

Keywords: memory systems, data movement, main memory, processing-in-memory, near-data processing, computation-in-memory, processing using memory, processing near memory, 3D-stacked memory, non-volatile memory, energy efficiency, high-performance computing, computer architecture, computing paradigm, emerging technologies, memory scaling, technology scaling, dependable systems, robust systems, hardware security, system security, latency, low-latency computing



Contents

1. Introduction

Main memory, built using the Dynamic Random Access Memory (DRAM) technology, is a major component in nearly all computing systems, including servers, cloud platforms, mobile/embedded devices, and sensor systems. Across all of these systems, the data working set sizes of modern applications are rapidly growing, while the need for fast analysis of such data is increasing. Thus, main memory is becoming an increasingly significant bottleneck across a wide variety of computing systems and applications [1–26]. Alleviating the main memory bottleneck requires the memory capacity, energy, cost, and performance to all scale in an efficient manner across technology generations. Unfortunately, it has become increasingly difficult in recent years, especially the past decade, to scale all of these dimensions [1, 2, 27–59], and thus the main memory bottleneck has been worsening.

A major reason for the main memory bottleneck is the high energy and latency cost associated with *data movement*. In modern computers, to perform any operation on data that resides in main memory, the processor must retrieve the data from main memory. This requires the memory controller to issue commands to a DRAM module across a relatively slow and power-hungry off-chip bus (known as the *memory channel*). The DRAM module sends the requested data across the memory channel, after which the data is placed in the caches and registers. The CPU can perform computation on the data once the data is in its registers. Data movement from the DRAM to the CPU incurs long latency and consumes a significant amount of energy [7–9, 60–64]. These costs are often exacerbated by the fact that much of the data brought into the caches is *not reused* by the CPU [62, 63, 65, 66], providing little benefit in return for the high latency and energy cost.

The cost of data movement is a fundamental issue with the *processor-centric* nature of contemporary computer systems. The CPU is considered to be the master in the system, and computation is performed only in the processor (and accelerators). In contrast, data storage and communication units, including the main memory, are treated as unintelligent workers that are incapable of computation. As a result of this processor-centric design paradigm, data moves a lot in the system between the computation units and communication/storage units so that computation can be done on it. With the increasingly *data-centric* nature of contemporary and emerging applications, the processor-centric design paradigm leads to great inefficiency in performance, energy and cost. For example, most of the real estate within a single compute

P&S Processing-in-Memory

Data-Centric Architectures: Fundamentally Improving Performance and Energy

Dr. Juan Gómez Luna

Prof. Onur Mutlu

ETH Zürich

Fall 2022

11 October 2022